

科学技術計算ライブラリ
ASL ユーザーズガイド
< 基本機能編 第5分冊 >

はしがき

本書は、科学技術計算ライブラリ ASL (Advanced Scientific Library) の概念、機能、利用方法などについて説明したものです。

当製品に対応する説明書は7分冊からなっており、構成は次のとおりです。このうち本書は、基本機能第5分冊について記述したものです。

基本機能 第1分冊

章	タイトル	内 容
1	使用の手引き	本説明書の構成、各項目の見方、および使用上の制限事項などの説明
2	格納モードの変換	配列データの格納モードの変換に関するサブルーチンのアルゴリズム、使用方法および使用例の説明
3	基本行列演算	行列の基本演算に関するサブルーチンのアルゴリズム、使用方法および使用例の説明
4	固有値・固有ベクトル	実行列、複素行列、実対称行列、エルミート行列、実対称バンド行列、実対称3重対角行列、実対称スパース行列、エルミートスパース行列の標準固有値問題および実行列、実対称行列、エルミート行列、実対称バンド行列の一般化固有値問題に関するサブルーチンのアルゴリズム、使用方法および使用例の説明

基本機能 第2分冊

章	タイトル	内 容
1	使用の手引き	本説明書の構成、各項目の見方、および使用上の制限事項などの説明
2	連立1次方程式(直接法)	実行列、複素行列、正値対称行列、実対称行列、エルミート行列、実バンド行列、正値対称バンド行列、実3重対角行列、実上三角行列、実下三角行列の連立1次方程式に関するサブルーチンのアルゴリズム、使用方法および使用例の説明

基本機能 第3分冊

章	タイトル	内 容
1	使用の手引き	本説明書の構成, 各項目の見方, および使用上の制限事項などの説明
2	フーリエ変換とその応用	1次元, 2次元および3次元の複素ならびに実フーリエ変換, 1次元, 2次元および3次元の畳み込み, 相関, パワー・スペクトル解析, ウェーブレット変換およびラプラス逆変換に関するサブルーチンのアルゴリズム, 使用方法および使用例の説明

基本機能 第4分冊

章	タイトル	内 容
1	使用の手引き	本説明書の構成, 各項目の見方, および使用上の制限事項などの説明
2	微分方程式とその応用	〔常微分方程式初期値問題〕 連立高階, 陰的連立, 行列型, スティフ問題の連立高階, 連立1階, 高階常微分方程式 〔常微分方程式境界値問題〕 連立高階, 連立1階, 高階, 線形高階, 線形2階常微分方程式 〔積分方程式〕 第2種フレドホルム型, 第1種ボルテラ型積分方程式 〔偏微分方程式〕 2次元および3次元の非同次ヘルムホルツ方程式 に関するサブルーチンのアルゴリズム, 使用方法および使用例の説明
3	数値微分	1変数関数および多変数関数の数値微分に関するサブルーチンのアルゴリズム, 使用方法および使用例の説明
4	数値積分	有限区間, 半無限区間, 全無限区間, 2次元有限区間, 多次元有限区間の数値積分に関するサブルーチンのアルゴリズム, 使用方法および使用例の説明
5	補間・近似	補間, 曲面補間, 最小二乗近似, 最小二乗曲面近似, チェビシェフ近似に関するサブルーチンのアルゴリズム, 使用方法および使用例の説明
6	スプライン関数	3次スプライン, 双3次スプラインおよびB-スプラインを用いた補間, 平滑化, 数値微分, 数値積分に関するサブルーチンのアルゴリズム, 使用方法および使用例の説明

基本機能 第 5 分冊

章	タイトル	内 容
1	使用の手引き	本説明書の構成, 各項目の見方, および使用上の制限事項などの説明
2	特殊関数	ベッセル関数, 変形ベッセル関数, 球ベッセル関数, ベッセル関数に関連した関数, ガンマ関数, ガンマ関数に関連した関数, 楕円関数, 初等関数の不定積分, ルジャンドル陪関数, 直交多項式, その他の特殊関数に関するサブルーチンのアルゴリズム, 使用方法および使用例の説明
3	ソート・順位付け	ソート, 順位付けに関するサブルーチンの使用方法および使用例の説明
4	方程式の根	代数方程式, 非線形方程式, 連立非線形方程式の根に関するサブルーチンのアルゴリズム, 使用方法および使用例の説明
5	極値問題・最適化	制約なし関数の極小化, 制約なし関数二乗和の極小化, 制約付き 1 変数関数の極小化, 制約付き多変数関数の最小化, 最短路問題に関するサブルーチンのアルゴリズム, 使用方法および使用例の説明

基本機能 第 6 分冊

章	タイトル	内 容
1	使用の手引き	本説明書の構成, 各項目の見方, および使用上の制限事項などの説明
2	乱数の検定	一様乱数の検定, 分布乱数の検定に関するサブルーチンの使用方法および使用例の説明
3	確率分布	連続分布, 離散分布に関するサブルーチンの使用方法および使用例の説明
4	基礎統計量	基礎統計量, 分散共分散, 相関係数に関するサブルーチンの使用方法および使用例の説明
5	推定と検定	区間推定, 検定に関するサブルーチンの使用方法および使用例の説明
6	分散分析・実験計画	1 元配置, 2 元配置, 多元配置, 乱塊法, グレコ・ラテン方格法, 累積法に関するサブルーチンの使用方法および使用例の説明
7	ノンパラメトリック検定	χ^2 分布による検定, その他分布による検定に関するサブルーチンの使用方法および使用例の説明
8	多変量解析	主成分分析, 因子分析, 正準相関分析, 判別分析, クラスタ分析に関するサブルーチンの使用方法および使用例の説明
9	時系列分析	自己相関・相互相関, 自己共分散・相互共分散, 平滑化・需要予測に関するサブルーチンの使用方法および使用例の説明
10	回帰分析	線形回帰, 非線形回帰に関するサブルーチンの使用方法および使用例の説明

共有メモリ並列機能

章	タイトル	内 容
1	使用の手引き	本説明書の構成, 各項目の見方, および使用上の制限事項などの説明
2	基本行列演算	実行列および複素行列の積を求めるサブルーチンのアルゴリズム, 使用方法の説明
3	連立 1 次方程式 (直接法)	実行列, 複素行列, 実対称行列, エルミート行列の連立 1 次方程式 (直接法) に関するサブルーチンのアルゴリズム, 使用方法および使用例の説明
4	連立 1 次方程式 (反復法)	実正値対称スパース行列, 実対称スパース行列, 実非対称スパース行列の連立 1 次方程式 (反復法) に関するサブルーチンのアルゴリズム, 使用方法および使用例の説明
5	固有値・固有ベクトル	実対称行列およびエルミート行列の固有値問題に関するサブルーチンのアルゴリズム, 使用方法および使用例の説明
6	フーリエ変換とその応用	1次元, 2次元および3次元の複素ならびに実フーリエ変換, 2次元および3次元の畳み込み, 相関, パワー・スペクトル解析に関するサブルーチンのアルゴリズム, 使用方法および使用例の説明
7	ソート	ソートに関するサブルーチンの使用方法および使用例の説明

2023年 3月 ASL 付属説明書 3.0.0-230301

- 備考 (1) 本書に説明しているすべての機能は, プログラムプロダクトであり, ASL 1.1 に対応しています.
- (2) 製品名などの固有名詞は, 各メーカーの登録商標または商標です.
- (3) 本ライブラリは, 最新の数値計算技法を取り入れ, 開発されたものです. 従って, 最新の技術を維持する目的から, 改良または新しく追加されたサブルーチンが, 既存のサブルーチンの機能を包含し, かつ, これまで以上の高速性能が得られる場合には, 既存のサブルーチンを削除することもあります.

目次

第 1 章	使用の手引	1
1.1	概説	1
1.1.1	科学技術計算ライブラリ ASL の概要	1
1.1.2	ASL の特長	1
1.2	ライブラリの種類	2
1.3	マニュアルについて	3
1.3.1	『概要』	3
1.3.2	サブルーチン説明文の構成	3
1.3.3	各項目の内容	3
1.4	サブルーチン名	7
1.5	注意事項	9
第 2 章	特殊関数	11
2.1	概要	11
2.1.1	使用上の注意	12
2.1.2	使用しているアルゴリズム	13
2.1.2.1	ベッセル関数	13
2.1.2.2	変形ベッセル関数	18
2.1.2.3	球ベッセル関数	26
2.1.2.4	ベッセル関数に関連した関数	29
2.1.2.5	ガンマ関数	32
2.1.2.6	ガンマ関数に関連した関数	35
2.1.2.7	楕円関数と楕円積分	36
2.1.2.8	初等関数の不定積分	42
2.1.2.9	ルジャンドル陪関数	47
2.1.2.10	直交多項式	48
2.1.2.11	整数次マシュー関数	49
2.1.2.12	ランジュバン関数	51
2.1.2.13	ガウス・ルジャンドル積分公式	51
2.1.2.14	ベッセル関数の零点	53
2.1.2.15	第 2 種ベッセル関数の正零点	55
2.1.2.16	正定値 2 次形式 $x^2 + ay^2$ のゼータ関数	55
2.1.2.17	ディログ関数	56
2.1.2.18	デバイ関数	56
2.1.2.19	正規化された球面調和関数	57
2.1.2.20	実変数フルビッツゼータ関数	58
2.1.2.21	誤差関数に関連した関数	60

2.1.2.22	係数算出法	63
2.1.2.23	関連した特殊関数の計算方法	63
2.1.3	参考文献	65
2.2	ベッセル関数	67
2.2.1	WIBJ0X, VIBJ0X 第1種0次ベッセル関数	67
2.2.2	WIBY0X, VIBY0X 第2種0次ベッセル関数	69
2.2.3	WIBJ1X, VIBJ1X 第1種1次ベッセル関数	71
2.2.4	WIBY1X, VIBY1X 第2種1次ベッセル関数	73
2.2.5	DIBJNX, RIBJNX 第1種整数次ベッセル関数	75
2.2.6	DIBYNX, RIBYNX 第2種整数次ベッセル関数	77
2.2.7	DIBJMX, RIBJMX 第1種実数次ベッセル関数	79
2.2.8	DIBYMX, RIBYMX 第2種実数次ベッセル関数	82
2.2.9	ZIBJNZ, CIBJNZ 複素変数第1種整数次ベッセル関数	85
2.2.10	ZIBYNZ, CIBYNZ 複素変数第2種整数次ベッセル関数	87
2.3	ベッセル関数の零点	90
2.3.1	DIZBS0, RIZBS0 第1種0次ベッセル関数の正零点	90
2.3.2	DIZBS1, RIZBS1 第1種1次ベッセル関数の正零点	92
2.3.3	DIZBSN, RIZBSN 第1種整数次ベッセル関数の正零点	94
2.3.4	DIZBYN, RIZBYN 第2種整数次ベッセル関数の正零点	96
2.3.5	DIZBSL, RIZBSL $aJ_0(\alpha) + xJ_1(\alpha)$ の正零点	98
2.4	変形ベッセル関数	100
2.4.1	WIBI0X, VIBI0X 第1種0次変形ベッセル関数	100
2.4.2	WIBK0X, VIBK0X 第2種0次変形ベッセル関数	102
2.4.3	WIBI1X, VIBI1X 第1種1次変形ベッセル関数	104
2.4.4	WIBK1X, VIBK1X 第2種1次変形ベッセル関数	106

2.4.5	DIBINX, RIBINX	
	第1種整数次変形ベッセル関数	108
2.4.6	DIBKNX, RIBKNX	
	第2種整数次変形ベッセル関数	110
2.4.7	DIBIMX, RIBIMX	
	第1種実数次変形ベッセル関数	112
2.4.8	DIBKMX, RIBKMX	
	第2種実数次変形ベッセル関数	115
2.4.9	ZIBINZ, CIBINZ	
	複素変数第1種整数次変形ベッセル関数	118
2.4.10	ZIBKNZ, CIBKNZ	
	複素変数第2種整数次変形ベッセル関数	120
2.5	球ベッセル関数	123
2.5.1	DIBSJM, RIBSJM	
	第1種整数次球ベッセル関数	123
2.5.2	DIBSYN, RIBSYN	
	第2種整数次球ベッセル関数	125
2.5.3	DIBSIN, RIBSIN	
	第1種整数次変形球ベッセル関数	127
2.5.4	DIBSKN, RIBSKN	
	第2種整数次変形球ベッセル関数	129
2.6	ベッセル関数に関連した関数	131
2.6.1	ZIBH1N, CIBH1N	
	第1種ハンケル関数	131
2.6.2	ZIBH2N, CIBH2N	
	第2種ハンケル関数	133
2.6.3	DIBBER, RIBBER	
	ケルビン関数 $\text{ber}_n(x)$	135
2.6.4	DIBBEI, RIBBEI	
	ケルビン関数 $\text{bei}_n(x)$	137
2.6.5	DIBKER, RIBKER	
	ケルビン関数 $\text{ker}_n(x)$	139
2.6.6	DIBKEI, RIBKEI	
	ケルビン関数 $\text{kei}_n(x)$	141
2.6.7	WIBH0X, VIBH0X	
	0次ストループ関数	143
2.6.8	WIBH1X, VIBH1X	
	1次ストループ関数	145
2.6.9	WIBHY0, VIBHY0	
	0次ストループ関数とベッセル関数の差	147
2.6.10	WIBHY1, VIBHY1	
	1次ストループ関数とベッセル関数の差	149
2.6.11	DIBAIX, RIBAIX	
	エアリ関数 $\text{Ai}(x)$	151

2.6.12	DIBBIX, RIBBIX エアリ関数 $Bi(x)$	153
2.6.13	DIBRID, RIBRID エアリ関数の導関数 $Ai'(x)$	155
2.6.14	DIBBID, RIBBID エアリ関数の導関数 $Bi'(x)$	157
2.7	ガンマ関数	159
2.7.1	WIGAMX, VIGAMX 実変数ガンマ関数	159
2.7.2	WIGLGX, VIGLGX 実変数対数ガンマ関数	162
2.7.3	DIGIG1, RIGIG1 第 1 種不完全ガンマ関数	164
2.7.4	DIGIG2, RIGIG2 第 2 種不完全ガンマ関数	166
2.7.5	ZIGAMZ, CIGAMZ 複素変数ガンマ関数	168
2.7.6	ZIGLGZ, CIGLGZ 複素変数対数ガンマ関数	170
2.8	ガンマ関数に関連した関数	172
2.8.1	WIGDIG, VIGDIG ディガンマ関数	172
2.8.2	WIGBET, VIGBET ベータ関数	174
2.9	楕円関数と楕円積分	176
2.9.1	WIECI1, VIECI1 第 1 種完全楕円積分	176
2.9.2	WIECI2, VIECI2 第 2 種完全楕円積分	178
2.9.3	DIEII1, RIEII1 第 1 種不完全楕円積分	180
2.9.4	DIEII2, RIEII2 第 2 種不完全楕円積分	182
2.9.5	DIEII3, RIEII3 不完全変形楕円積分	184
2.9.6	DIEII4, RIEII4 ワイエルシュトラス型の不完全楕円積分	186
2.9.7	WIEJAC, VIEJAC ヤコビの楕円関数	188
2.9.8	WIENMQ, VIENMQ ノーム q および完全楕円積分	190
2.9.9	WIETHE, VIETHE 楕円テータ関数	193

2.9.10	WIEJZT, VIEJZT ヤコビのゼータ関数	196
2.9.11	WIEJEP, VIEJEP ヤコビのエプシロン関数	198
2.9.12	WIEJTE, VIEJTE ヤコビのテータ関数	200
2.9.13	WIEPAI, VIEPAI パイ関数	202
2.10	初等関数の不定積分	205
2.10.1	WIEXP, VIEXP 指数積分	205
2.10.2	WILOG, VILOG 対数積分	208
2.10.3	DIISIN, RIISIN 正弦積分	210
2.10.4	DIICOS, RIICOS 余弦積分	212
2.10.5	WIIFSI, VIIFSI フレネル正弦積分	214
2.10.6	WIIFCO, VIIFCO フレネル余弦積分	216
2.10.7	WIIDAW, VIIDAW ドーソン積分	218
2.10.8	WIICND, VIICND 正規分布関数	220
2.10.9	WIICNC, VIICNC 余正規分布関数	222
2.11	誤差関数に関連した関数	224
2.11.1	WIERRF, VIERRF 誤差関数	224
2.11.2	WIERFC, VIERFC 余誤差関数	226
2.11.3	DIIERF, RIIERF 余誤差関数の逆関数	228
2.11.4	JIIERF, IIIERF 複素変数の誤差関数	230
2.12	ルジャンドル陪関数	232
2.12.1	DILEG1, RILEG1 第1種ルジャンドル陪関数	232
2.12.2	DILEG2, RILEG2 第2種ルジャンドル陪関数	235
2.13	直交多項式	237
2.13.1	DIOPLE, RIOPLE ルジャンドル多項式	237

2.13.2	DIZGLW, RIZGLW ガウス・ルジャンドル積分公式	239
2.13.3	DIOPLA, RIOPLA ラゲール多項式	242
2.13.4	DIOPHE, RIOPHE エルミート多項式	244
2.13.5	DIOPCH, RIOPCH チェビシェフ多項式	246
2.13.6	DIOPC2, RIOPC2 第 2 種チェビシェフ関数	248
2.13.7	DIOPGL, RIOPGL 一般ラゲール多項式	250
2.14	マシユー関数	252
2.14.1	DIMTCE, RIMTCE 整数次マシユー関数 $ce_n(x, q)$	252
2.14.2	DIMTSE, RIMTSE 整数次マシユー関数 $se_n(x, q)$	255
2.15	その他の関数	258
2.15.1	WIXSPS, VIXSPS ディログ関数	258
2.15.2	WIDBEY, VIDBEY デバイ関数	261
2.15.3	WINPLG, VINPLG 球面調和関数	263
2.15.4	WIXSLA, VIXSLA ランジュバン関数	266
2.15.5	WIXZTA, VIXZTA フルビッツゼータ関数	268
2.15.6	DIXEPS, RIXEPS 正定値 2 次形式 $x^2 + ay^2$ のゼータ関数	270
第 3 章	ソート・順位付け	273
3.1	概要	273
3.1.1	使用しているアルゴリズム	274
3.1.1.1	ソート	274
3.1.1.2	データ列の順位付け	275
3.1.1.3	上位 N 件の抽出	275
3.1.1.4	ソート済みデータ列のマージ	275
3.1.1.5	ソート済みペアデータ列のマージ	275
3.1.2	参考文献	276
3.2	ソート	277
3.2.1	DSSTA1, RSSTA1 データ列のソート	277

3.2.2	DSSTA2, RSSTA2 ペアデータ列のソート	280
3.3	順位付け	284
3.3.1	DSSTRA, RSSTRA データ列の順位付け	284
3.3.2	DSSTPT, RSSTPT 上位 N 件の抽出	287
3.4	マージ	290
3.4.1	DSMGON, RSMGON ソート済みデータ列のマージ	290
3.4.2	DSMGPA, RSMGPA ソート済みペアデータ列のマージ	294
第 4 章	方程式の根	299
4.1	概要	299
4.1.1	使用上の注意	300
4.1.2	使用しているアルゴリズム	302
4.1.2.1	実係数代数方程式の根	302
4.1.2.2	複素係数代数方程式の根	311
4.1.2.3	実関数の根 (初期値指定) (導関数定義必要)	313
4.1.2.4	実関数の根 (初期値指定) (導関数定義不要)	315
4.1.2.5	実関数の根 (区間指定) (導関数定義不要)	317
4.1.2.6	実関数の全根 (区間指定) (導関数定義不要)	317
4.1.2.7	複素関数の根 (初期値指定) (導関数定義不要)	318
4.1.2.8	連立非線形方程式の根 (ヤコビ行列定義任意)	319
4.1.2.9	連立非線形方程式の根 (ヤコビ行列定義不要)	320
4.1.3	参考文献	323
4.2	代数方程式	324
4.2.1	DLARHA, RLARHA 実係数代数方程式の根	324
4.2.2	ZLACHA, CLACHA 複素係数代数方程式の根	327
4.3	非線形方程式	330
4.3.1	DLNRDS, RLNRDS 実関数の根 (初期値指定) (導関数定義必要)	330
4.3.2	DLNRIS, RLNRIS 実関数の根 (初期値指定) (導関数定義不要)	333
4.3.3	DLNRSS, RLNRSS 実関数の根 (区間指定) (導関数定義不要)	336
4.3.4	DLNRSA, RLNRSA 実関数の全根 (区間指定) (導関数定義不要)	339
4.3.5	ZLNCIS, CLNCIS 複素関数の根 (初期値指定) (導関数定義不要)	342
4.4	連立非線形方程式	345

4.4.1	DLSRDS, RLSRDS 連立非線形方程式の根 (ヤコビ行列定義任意)	345
4.4.2	DLSRIS, RLSRIS 連立非線形方程式の根 (ヤコビ行列定義不要)	350
第 5 章	極値問題・最適化	353
5.1	概要	353
5.1.1	使用上の注意	355
5.1.2	使用しているアルゴリズム	356
5.1.2.1	1 変数関数の極小化	356
5.1.2.2	多変数関数の極小化	357
5.1.2.3	非線形最小二乗法	358
5.1.2.4	制約付き多変数線形関数の最小化 (線形制約)	360
5.1.2.5	0-1 変数を含む線形制約付き多変数線形関数の最小化	367
5.1.2.6	ネットワーク上の流れに対する費用の最小化	370
5.1.2.7	プロジェクトの日程計画に対する費用の最小化	372
5.1.2.8	供給地から需要地への輸送費用の最小化	374
5.1.2.9	制約付き多変数凸型 2 次関数の最小化 (線形制約)	374
5.1.2.10	多変数広義凸型 2 次関数の最小化 (線形制約)	376
5.1.2.11	制約無し 0-1 多変数 2 次関数の最小化	378
5.1.2.12	制約付き多変数関数の最小化	382
5.1.2.13	ネットワーク上の 2 節点間の距離の最小化	383
5.1.3	参考文献	385
5.2	制約なし 1 変数関数の極小化	386
5.2.1	DMUUSN, RMUUSN 1 変数関数の極小化	386
5.3	制約なし多変数関数の極小化	389
5.3.1	DMUMQN, RMUMQN 多変数関数の極小化 (導関数定義不要)	389
5.3.2	DMUMQG, RMUMQG 多変数関数の極小化 (導関数定義必要)	392
5.4	制約なし関数二乗和の極小化	396
5.4.1	DMUSSN, RMUSSN 非線形最小二乗法 (導関数定義不要)	396
5.5	制約付き 1 変数関数の極小化	399
5.5.1	DMCUSN, RMCUSN 1 変数関数の極小化 (区間指定)	399
5.6	制約付き多変数線形関数の最小化 (線形計画)	402
5.6.1	DMCLSN, RMCLSN 多変数線形関数の最小化 (線形制約)	402
5.6.2	DMCLAF, RMCLAF 多変数線形関数の最小化 (実不規則スパース行列で与えられる線形制約)	407
5.6.3	DMCLMZ, RMCLMZ 0-1 変数を含む線形制約付き多変数線形関数の最小化 (混合 0-1 計画)	416

5.6.4	DMCLMC, RMCLMC	
	ネットワーク上の流れに対する費用の最小化 (最小費用流問題)	422
5.6.5	DMCLCP, RMCLCP	
	プロジェクトの日程計画に対する費用の最小化 (日程計画問題)	427
5.6.6	DMCLTP, RMCLTP	
	供給地から需要地への輸送費用の最小化 (輸送問題)	433
5.7	多変数 2 次関数の最小化 (2 次計画)	439
5.7.1	DMCQSN, RMCQSN	
	多変数凸型 2 次関数の最小化 (線形制約)	439
5.7.2	DMCQLM, RMCQLM	
	多変数広義凸型 2 次関数の最小化 (線形制約)	444
5.7.3	DMCQAZ, RMCQAZ	
	制約無し 0-1 多変数 2 次関数の最小化 (0-1 無制約 2 次計画問題)	449
5.8	制約付き多変数関数の最小化 (非線形計画)	455
5.8.1	DMSQPM, RMSQPM	
	制約付き多変数関数の最小化 (非線形制約)	455
5.9	ネットワーク上の距離の最短化 (最短路問題)	460
5.9.1	DMSP1M, RMSP1M	
	ネットワーク上のある節点から他のすべての節点までの距離の最小化	460
5.9.2	DMSPMM, RMSPMM	
	ネットワーク上の全 2 節点間の距離の最小化	464
5.9.3	DMSP11, RMSP11	
	ネットワーク上の 2 節点間の距離の最小化	467
付 録 A 用語説明		471
付 録 B ASL で使用している計算機依存定数		473
B.1	誤差判定のための単位	473
B.2	浮動小数点データの値の最大値・最小値	473

第 1 章 使用の手引

1.1 概説

1.1.1 科学技術計算ライブラリ ASL の概要

科学技術計算ライブラリ ASL (Advanced Scientific Library) は、数値解析プログラムの作成を強力に支援する数学ライブラリである。ASL では広範な数値解析分野で頻出するプログラムを提供しており、それらは VE(Vector Engine) 上で優れた実行速度と精度を実現するための高度な最適化が適用されている。ASL を用いることによって、難解な数値計算アルゴリズムの詳細に煩わされることなく高度な数値解析プログラムを作成することができ、数値解析プログラム開発の生産性を大幅に改善することができる。

ASL は、基本機能、共有メモリ並列機能で構成される。機能分類と本マニュアルの分冊との対応を表 1-1 に示す。

表 1-1 ASL の機能分類

機能分類	分冊
基本機能	第 1~6 分冊
共有メモリ並列機能	第 7 分冊

1.1.2 ASL の特長

ASL の特長は、次のとおりである。

- (1) ハードウェア性能を十分発揮できるように設計しており、コンパイラの最適化機能を用いて作成した。
- (2) 行列を扱うサブルーチンでは、行列の種類 (対称行列、エルミート行列など) に応じて最適に処理を行えるように、専用のサブルーチンをそれぞれ提供している。一般に、専用のサブルーチンを用いて処理を行った方が、処理性能を向上したり、必要なメモリ容量を節約したりすることができる。
- (3) 処理手順に従ってモジュール化を行い、コンポーネントサブルーチンごとの信頼性向上に努めるとともに、システム全体の効率化、信頼性向上を図った。
- (4) サブルーチンを利用した後の エラーインディケータの番号が体系的に決めてあるので、エラー情報を把握しやすい。

1.2 ライブラリの種類

ASL には、32 ビット整数型ライブラリと 64 ビット整数型ライブラリがある。32 ビット整数型ライブラリに含まれるサブルーチンの整数型の引数は、32 ビット (4 バイト) 整数型である。一方、64 ビット整数型ライブラリに含まれるサブルーチンの整数型の引数は、64 ビット (8 バイト) 整数型である。また、サブルーチンの実数型の引数によってサブルーチン名が異なる。サブルーチン名については、1.4 を参照のこと。

表 1-2 ASL で提供しているライブラリの種類

変数の大きさ (バイト)		引数の型宣言文	通称	ライブラリの種類
整数型	実数型			
4	8	INTEGER(4) REAL(8)	32 ビット整数型倍精度 サブルーチン	32 ビット整数型ライブラリ (リンクオプション: -lasl_sequential)
4	4	INTEGER(4) REAL(4)	32 ビット整数型単精度 サブルーチン	
8	8	INTEGER(8) REAL(8)	64 ビット整数型倍精度 サブルーチン	64 ビット整数型ライブラリ (リンクオプション: -lasl_sequential_i64)
8	4	INTEGER(8) REAL(4)	64 ビット整数型単精度 サブルーチン	

(注 1) 機能によっては、4 種類全てをサポートしているとは限らない。その場合、個別の説明の注意事項の欄に記述するので注意されたい。

(注 2) INTEGER(4) および REAL(4) で型宣言する場合、“(4)” は省略可。

1.3 マニュアルについて

ここでは本マニュアルの第2章以降の構成について述べる。

第2章以降は ASL で用いられるサブルーチンとその機能, 使用方法の説明を行う。

1.3.1 『概要』

各章の第1節では, 概要として各サブルーチンの効果的な使用法, 採用した手法およびそのアルゴリズム, 注意事項などについて述べてある。

1.3.2 サブルーチン説明文の構成

各章の第2節では, サブルーチンごとに以下の順で説明している。

- (1) 機能
- (2) 使用法
- (3) 引数
- (4) 制限条件
- (5) エラーインディケータ
- (6) 注意事項
- (7) 使用例

各項目は次に述べる原則に従って記述されている。

1.3.3 各項目の内容

(1) 機能

この項目では, サブルーチンの目的とする機能について簡単に述べてある。

(2) 使用法

この項目では, サブルーチン名とその引数の順序について記述してある。
引数の並べ方は, 原則として次のように決められている。

CALL サブルーチン名 (入力引数, 入出力引数, 出力引数, ISW, ワーク, IERR)

ここで, ISW は処理の手順を指定するための入力引数であり, IERR は エラーインディケータである。ただし, 入力引数と入出力引数の順序が逆の場合もある。さらに次の規則にしたがっている。

- 配列は重要度に応じてできるだけ左方によせる。
- 配列名に続けて配列の大きさをそえる。同じ大きさをもつ配列が複数個あるときは, その最初の配列名に続けてその大きさを引数として与え, 2 番目以降の配列からは, その大きさは引数として与えない。

(3) 引数

(2) 項で記述された引数について、順番に説明されている。その形式は以下のように統一されている。

引数	型	大きさ	入出力	内容
(a)	(b)	(c)	(d)	(e)

(a) 引数

引数が記載されている。

(b) 型

引数のデータの型を示す。次の略記号のいずれかに示されている。

I : 整数型

D : 倍精度実数型

R : 単精度実数型

Z : 倍精度複素数型

C : 単精度複素数型

整数型の引数には 64 ビット整数型と 32 ビット整数型とがある。サブルーチンの整数型引数が 64 ビット整数型であるのか 32 ビット整数型であるのかは、そのサブルーチンが 64 ビット整数型であるか 32 ビット整数型であるか、つまりライブラリの種類によって決められる (1.2 参照)。ユーザプログラムにおいて引数の型を宣言する際は、32 ビット整数型の引数は `INTEGER(4)`、64 ビット整数型の引数は `INTEGER(8)` を用いて宣言する必要がある。

(c) 大きさ

指定された引数の必要な大きさを示す。2 以上を指定した場合には、このサブルーチンを利用したプログラム側で、その必要な領域を確保しなければならない。

1 : 変数であることを示す。

N : 要素が N 個の 1 次元配列であることを示す。この配列が指定された直後にその大きさを示す引数 N が定義される。ただし大きさ N が以前に定義された配列の大きさを規定している場合には省略される。このほかに数値のみにて指定する場合や、 $3 \times N$ や $N + M$ のように、積または和の形で表記する場合もある。

M, N : M 行 N 列の 2 次元の配列であることを示す。この配列が指定される前にこの M と N が定義されていない場合は、この配列の直後にその大きさを示す引数 M または N が定義される。

(d) 入出力

引数の内容説明が入力時であるか出力時であるかを示す。

i. 「入力」とだけある場合 :

このサブルーチンを利用したプログラムに制御がもどったときに、引数の入力時の情報は保存されている。入力時の情報は特に断らない限り、利用者が与えなければならない。

ii. 「出力」とだけある場合 :

引数には、サブルーチン内で計算された結果が出力される。入力時には何も入れなくてよい。

iii. 「入力」と「出力」の両方に説明がある場合 :

サブルーチンに制御がわたる前とサブルーチンから制御がもどった後で、この引数の内容に変化がある場合である。入力時の情報は特に断らない限り、利用者が与えなければならない。

iv. 「ワーク」とある場合 :

サブルーチン内で演算を行うときに利用する領域であることを示す。サブルーチンを利用するプログラム側で、指定された大きさの作業領域を確保しなければならない。なお、次の計算に流用するために、作業領域の内容を保存しておく必要がある場合がある。

(e) 内容

入力時あるいは出力時に、引数が保持している情報について説明される。

- 「引数」の説明の例を次に示す。

例 実行列の LU 分解と条件数を求めるサブルーチン (DBGMLC, RBGMLC) の使用法は以下のとおりである。

倍精度サブルーチン:

CALL DBGMLC (A, LNA, N, IPVT, COND, W1, IERR)

単精度サブルーチン:

CALL RBGMLC (A, LNA, N, IPVT, COND, W1, IERR)

この場合の引数の説明は次のようになる。

表 1-3 引数の例

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	A	$\begin{Bmatrix} D \\ R \end{Bmatrix}$ 注	LNA, N	入力	実行列 A(2次元配列型)
				出力	A = LU と分解した時の単位上三角行列 U および下三角行列 L
2	LNA	I	1	入力	配列 A の整合寸法
3	N	I	1	入力	行列 A の次数 n
4	IPVT	I	N	出力	ピボット情報 IPVT(i): i 段目の処理において行 i と交換した行の番号
5	COND	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出力	条件の逆数
6	W1	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	N	ワーク	作業領域
7	IERR	I	1	出力	エラーインディケータ

このサブルーチンを利用するには、まず、引数として使用する配列 A, IPVT および W1 を、呼び出し元の利用者プログラム側でアロケートする必要がある。それらはそれぞれ、 $\begin{Bmatrix} \text{倍精度} \\ \text{単精度} \end{Bmatrix}$ 注 実数型で大きさ

(LNA, N), 整数型で大きさ N, $\begin{Bmatrix} \text{倍精度} \\ \text{単精度} \end{Bmatrix}$ 実数型で大きさ N の配列である。

また、64 ビット整数版を利用する場合には、整数型引数 (LNA, N, IPVT, IERR) はすべて INTEGER ではなく INTEGER(8) を用いて宣言する必要がある。

注 DBGMLC のときには倍精度実数型 (略記号 D), RBGMLC のときには実数型 (略記号 R) で宣言することを意味する。以下、本文中で特に断らない限り中括弧 {} 等の使用法は、同様の扱いとする。

このサブルーチンを使用するときには、A、LNA および N にデータを格納しておかなければならない。サブルーチン内では、与えられた行列の LU 分解と条件数の算出が行われ、結果が配列 A と変数 COND に格納される。また、後続サブルーチンで利用するため、ピボット情報 IPVT に格納される。

IERR は、入力データや処理途中の異常を利用者に知らせるための引数であり、正常の場合は 0 にセットされる。

なお、W1 はサブルーチン内でのみ使用する作業領域であるので、入力時および出力時の内容は特に意味をもたない。

(4) 制限条件

サブルーチンの引数の制限範囲を明確にしてある。

(5) エラーインディケータ

各サブルーチンには、エラーインディケータが出力引数として設けられている。このエラーインディケータは、IERR という変数名に統一されており、引数表の最後におかれている。各サブルーチンはサブルーチン内でエラー検出を行い、その結果を IERR に設定する。IERR の値の意味は、次の 5 段階に分かれている。

表 1-4 エラーインディケータの出力値区分

レベル	IERR の値	意 味	処 理 内 容
正 常	0	正常終了した。	結果は保証される。
警 告	1000 ~ 2999	ある条件のもとで一応の処理が終了した。	条件付きで結果は保証される。
異 常	3000 ~ 3499	引数が制限条件に違反したために処理が打ち切られた。	結果は保証されない。
	3500 ~ 3999	得られた結果がある検定条件を満足しなかった。	得られた結果を返す (結果は保証されない)。
	4000 以上	処理の途中で致命的なエラーが発見された。通常は処理を打ち切る。	結果は保証されない。

(6) 注意事項

サブルーチンを使用するときの注意点およびあいまいな点を明確にしてある。

(7) 使用例

サブルーチンの使い方の一例を載せてある。なお複数のサブルーチンを組み合わせて一つの例としてある場合もあるので注意されたい。出力結果は、32 ビット整数版での結果であり、コンパイラや組み込み関数の変更などにより丸め誤差の範囲で異なる場合がある。

本説明書に記載されている使用例のプログラムはソースコードの形で「ASL ユーザーズガイド」に収録されている。入力データも (もし存在する場合は) 「ASL ユーザーズガイド」に収録されている。コンパイラを用いて使用例のソースコードから実行形式ファイルを作成する場合には、ライブラリ本体とリンクする必要がある。

1.4 サブルーチン名

ASL の基本機能のサブルーチン名は、6桁のアルファニューメリック記号の集まりである。また、サブルーチン名の各記号にはそれぞれ意味を持ち、図 1-1 で表される。利用時には、計算用途に合わせてサブルーチン名を指定する必要がある。

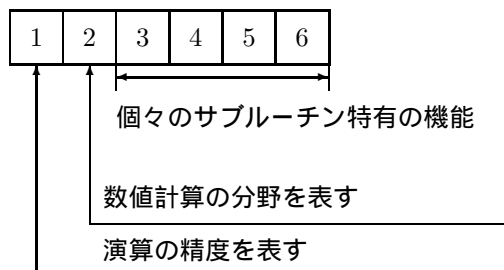


図 1-1 サブルーチン名の構成要素

図 1-1 の“1”：演算の精度を表す。基本機能編で使用される文字は、次の 8 種類である。

- D, W 倍精度実数型演算
- R, V 単精度実数型演算
- Z, J 倍精度複素数型演算
- C, I 単精度複素数型演算

ただし、上記の複素数型とは必ずしも引数の型が複素数型であることを意味しない。

図 1-1 の“2”：計算の分野を表す。現在、ASL では次の文字が使用されている。

文字	計算の分野	分冊
A	格納モードの変換	1
	基本行列演算	1, 7
B	連立 1 次方程式 (直接法)	2, 7
C	固有値・固有ベクトル	1, 7
F	フーリエ変換とその応用	3, 7
	時系列分析	6
G	スプライン関数	4
H	数値積分	4
I	特殊関数	5
J	乱数の検定	6
K	常微分方程式初期値問題	4
L	方程式の根	5
M	極値問題・最適化	5
N	近似・回帰分析	4, 6
O	常微分方程式境界値問題, 積分方程式, 偏微分方程式	4
P	補間	4
Q	数値微分	4

文字	計算の分野	分冊
S	ソート・順位付け	5, 7
X	基本行列演算	1
	連立1次方程式(反復法)	7
1	確率分布	6
2	標本統計	6
3	推定と検定	6
4	分散分析・実験計画	6
5	ノンパラメトリック検定	6
6	多変量解析	6

図1-1の“3”～“6”：これらの文字で、個々のサブルーチンに特有の機能を表す。

1.5 注意事項

- (1) 単精度版ではなく、倍精度版を標準として利用する方がよい。精度が高いことに加え、倍精度版の方が単精度版に比べて安定的に解が求まる場合 (特に固有値・固有ベクトル) が多い。
- (2) 演算例外の抑止はメインプログラム側で行う必要がある。ASL のサブルーチンでは、コンパイラの演算例外の抑止に関して、ユーザのメインプログラムのコンパイルパラメータの指示に従うように設定してある。
- (3) 扱う演算桁数を越える精度を期待することはできない。たとえば倍精度演算の (仮数部の) 演算桁数は 10 進 15 桁程度であるが、ここで数学的に 1 となるような値を計算した場合、 10^{-15} 程度の誤差は必ず発生する。これを抑制する方法として、任意桁数演算のような多倍長演算のエミュレートが考えられるが、この場合、たとえば円周率のような定数や関数近似の定数なども都度計算する必要が生じるので、通常の演算と比較して計算効率は悪くなる。
- (4) 数学的に解が存在しないような問題の解を得ることはできない。たとえば、数学的に特異な (または特異に近い) 行列を係数に持つ連立 1 次方程式の解を精度良く求めることは原理的にできない。なお、数値計算上は、数学的に特異な行列と特異に近い行列とを厳密に区別することはできない。もちろん、たとえば、条件数の計算値が設定した基準値以上であれば特異とみなすというようなことはいつでも可能である。
- (5) 浮動小数点例外 (オーバフローなど) をおこすようなデータを与えた場合、正常な計算結果を期待することはできない。ただし、反復計算で残差の加算等を行った場合に発生する浮動小数点アンダフローなどはこの限りではない。
- (6) 数値計算で扱う問題 (特に反復法を計算手法とする問題) では、与えるデータによっては解が精度良く求められない場合や全く求まらない場合がある。このような場合は、問題自体を見直して、解が求まるような問題に変更するなどの処置を講じる必要がある。たとえば、スパース行列を係数とする連立 1 次方程式を解く場合に、専用のサブルーチンで解が得られないときでも、密行列用のサブルーチンを用いることで解が得られる場合がある。
- (7) 解が複数ある問題を解く場合、実行するマシンや OS、用いるコンパイラ等で実行結果が見掛け上異なる場合がある。たとえば、固有値問題を解いた場合に得られる固有ベクトルがこれに相当する。
- (8) “[非推奨]” と表示のあるサブルーチンは、今後廃止予定の機能である。より高速な実装が ASL 統合インタフェースで提供されているので、そちらを利用されたい。

第 2 章 特殊関数

2.1 概要

本章では特殊関数の関数値を求めるサブルーチンについて説明する。特殊関数の計算方法には、

- テーラー展開や漸近展開による方法
- 近似式による方法
- 連分数による方法
- 漸化式による方法

などがあるが、ここでは区間を分割し最も最適と考えられる計算方法を各区間に対し適用する方法をとった。

2.1.1 使用上の注意

- (1) 第2種ベッセル関数 $N_\nu(z)$ や第2種球ベッセル関数 $n_\nu(z)$ はここでの $Y_\nu(z), y_\nu(z)$ と同じものである。
- (2) 実数次や整数次の各種ベッセル関数は、引数 z や次数 ν が大きいほど、計算時間が多くかかり、一般に $|\nu| < 1000.0, |z| < 1000.0$ とすることが望ましい。
- (3) 第1種のベッセル、変形ベッセル、球ベッセル、変形球ベッセル関数で次数が1ずつ変化する連続した値を得たいときは、高い次数の連続する2個の値をこのサブルーチンより求め、あとは高い次数から順次次数を減ずる方向に漸化式を用いて計算すると良い。
- (4) デイログ関数 $Li_2(x)$ ・デバイ関数 $F_D(x)$ ・球面調和関数 $P_n^{*m}(x)$ はベクトル処理・共通な係数・漸化式を用いて値が求められることにより、個別に計算するよりはまとめて計算する方が効率的である。
- (5) $J_{-\nu}(x), I_{-\nu}(x), Y_{-\nu}(x), i_{-n}(x), Y_{-n}(x)$ は漸化式を用いて計算する。漸化式は各サブルーチンの注意事項を参照。
- (6) 半奇数次のベッセル関数や変形ベッセル関数は球ベッセル関数を用いる方が効率的である。

$$J_{n+\frac{1}{2}}(x) = \sqrt{\frac{2x}{\pi}} \cdot j_n(x), Y_{n+\frac{1}{2}}(x) = \sqrt{\frac{2x}{\pi}} \cdot y_n(x)$$

$$I_{n+\frac{1}{2}}(x) = \sqrt{\frac{2x}{\pi}} \cdot i_n(x), K_{n+\frac{1}{2}}(x) = \sqrt{\frac{2x}{\pi}} \cdot k_n(x)$$

2.1.2 使用しているアルゴリズム

2.1.2.1 ベッセル関数

(1) 第1種0次, 1次ベッセル関数 $J_0(x), J_1(x)$

① $x < 0.0$ のとき

$J_0(x) = J_0(-x), J_1(x) = -J_1(-x)$ として $J_0(-x), J_1(-x)$ について以下に述べる方法を適用する.

② $0.0 \leq x < 4.0$ のとき次の式を最良近似式化して求める.

$$J_0(x) = \sum_{k=0}^{\infty} \frac{(-1)^k}{(k!)^2} \left(\frac{x}{2}\right)^{2k}$$

$$J_1(x) = \sum_{k=0}^{\infty} \frac{(-1)^k}{k!(k+1)!} \left(\frac{x}{2}\right)^{2k+1}$$

③ $4.0 \leq x \leq 8.0$ のとき次の式を最良近似式化して求める.

$$J_0(x) = \sum_{k=0}^{\infty} \frac{J_0^{(k)}(6)}{k!} (x-6)^k$$

$$J_1(x) = \sum_{k=0}^{\infty} \frac{J_1^{(k)}(6)}{k!} (x-6)^k$$

($J_0^{(k)}(6), J_1^{(k)}(6)$ は $x = 6.0$ での k 階微分値)

最良近似式化については 2.1.2.22 に記述されている. また, $J_0^{(k)}(6), J_1^{(k)}(6)$ 計算法は文献 (19) に記述されている.

④ $x > 8.0$ のとき次の漸近展開式で求める.

$$J_0(x) \text{ または } J_1(x) = \frac{P \cos(\phi) - Q \sin(\phi)}{\sqrt{x}}$$

ここで

$J_0(x)$ のとき

$$P = \frac{\sum_{n=0}^m a_n^{(1)} \left(\frac{8}{x}\right)^{2n}}{\sum_{n=0}^{m'} b_n^{(1)} \left(\frac{8}{x}\right)^{2n}}$$

$$\phi = x - \frac{\pi}{4}$$

$J_1(x)$ のとき

$$Q = \frac{\sum_{n=0}^m c_n^{(2)} \left(\frac{8}{x}\right)^{2n}}{\sum_{n=0}^{m'} d_n^{(2)} \left(\frac{8}{x}\right)^{2n}}$$

$$\phi = x - \frac{3}{4}\pi$$

である. 係数 $a_n^{(1)}, a_n^{(2)}, b_n^{(1)}, b_n^{(2)}, c_n^{(1)}, c_n^{(2)}, d_n^{(1)}, d_n^{(2)}$ は文献 (2) に記述されている.

(2) 第2種0次, 1次ベッセル関数 $Y_0(x), Y_1(x)$ ($x > 0.0$)

① $0.0 < x < 4.0$ のとき次の式を最良近似式化して求める.

$$Y_0(x) = \frac{2}{\pi} \left[J_0(x) \left\{ \log \left(\frac{x}{2} \right) + \gamma \right\} - \sum_{k=1}^{\infty} \left\{ \frac{(-1)^k}{(k!)^2} \left(\frac{x}{2} \right)^{2k} \sum_{m=1}^k \frac{1}{m} \right\} \right]$$

$$Y_1(x) = \frac{2}{\pi} \left[J_1(x) \left\{ \log \left(\frac{x}{2} \right) + \gamma \right\} \right]$$

$$+ \frac{2}{\pi} \left[-\frac{1}{x} - \frac{1}{2} \sum_{k=0}^{\infty} \left\{ \frac{(-1)^k}{k!(k+1)!} \left(\frac{x}{2} \right)^{2k+1} \left(\sum_{m=1}^k \frac{1}{m} + \sum_{m=1}^{k+1} \frac{1}{m} \right) \right\} \right]$$

ここで γ はオイラーの定数で $0.5772 \dots$ である.

② $4.0 \leq x \leq 8.0$ のとき次の式を最良近似式化して求める.

$$Y_0(x) = \sum_{k=0}^{\infty} \frac{Y_0^{(k)}(6)}{k!} (x-6)^k$$

$$Y_1(x) = \sum_{k=0}^{\infty} \frac{Y_1^{(k)}(6)}{k!} (x-6)^k$$

最良近似式化については 2.1.2.22 に記述されている. ここで, $Y_0(6), Y_1(6)$ の k 階微分値は, ベッセル関数を $Z_k(x)$ として,

$$Z'_k(x) = Z_{k-1}(x) - \frac{k}{x} Z_k(x)$$

$$Z_1^{(k)}(x) = -Z_0^{(k+1)}(x)$$

より漸化式を作り求める. この方法については文献 (19) に詳細が記述されている.

③ $x > 8.0$ のとき次の漸近展開式で求める.

$$Y_0(x) \text{ または } Y_1(x) = \frac{P \sin(\phi) + Q \cos(\phi)}{\sqrt{x}}.$$

ここで P, Q, ϕ は $J_0(x), J_1(x)$ の場合と同じである. 係数 $a_n^{(1)}, a_n^{(2)}, b_n^{(1)}, b_n^{(2)}$ は文献 (2) に記述されている.

(3) 第 1 種整数次ベッセル関数 $J_n(x)$

$J_n(x)$ は x, n の符号に応じて $J_{|n|}(|x|)$ を用いて表 2-1 のように表せる.

表 2-1 x, n の符号と $J_n(x)$ の計算方法

—	$n < 0$	$n \geq 0$
$x < 0.0$	$J_{ n }(x)$	$(-1)^n J_{ n }(x)$
$x \geq 0.0$	$(-1)^n J_{ n }(x)$	$J_{ n }(x)$

したがって以下に, $J_{|n|}(|x|)$ の計算法について記述する. 表記を簡略化するため, $|n| \rightarrow n, |x| \rightarrow x$ と表す.

① $n = 0, 1$ のときは $J_0(x), J_1(x)$ を利用する.

② $n \geq 2$ のとき

(a) $0.0 \leq x^2 < \left\{ \begin{array}{l} \text{倍精度: } 0.1n + 0.4 \\ \text{単精度: } 1.9n + 7.6 \end{array} \right\}$ で $n < \left\{ \begin{array}{l} \text{倍精度: } 170 \\ \text{単精度: } 34 \end{array} \right\}$ 場合

$$J_n(x) = \left(\frac{x}{2} \right)^n \sum_{k=0}^9 \frac{(-1)^k \left(\frac{x}{2} \right)^{2k}}{k!(n+k)!}$$

のべき級数展開で求める.

(b) $x^2 \geq \left\{ \begin{array}{l} \text{倍精度: } 0.1n + 0.4 \\ \text{単精度: } 1.9n + 7.6 \end{array} \right\}$ または $n \geq \left\{ \begin{array}{l} \text{倍精度: } 170 \\ \text{単精度: } 34 \end{array} \right\}$ の場合

i. $x \leq n$ ならば

$T_{n+k+1} = 0.0$ とし、次の漸化式を用いて連分数近似におけるパラメータ T_i ($i = n, n+1, \dots, n+k-1, n+k$) を計算する.

$$T_i = \frac{x^2}{2i - T_{i+1}} \quad (i = n+1, \dots, n+k-1, n+k)$$

$$T_n = \frac{x}{2n - T_{n+1}}$$

(T_n は $\frac{J_n(x)}{J_{n-1}(x)}$ の値)

k の値は表 2-2 に示すようにとる.

表 2-2 k の値

単精度演算	倍精度演算
$k = \lfloor \frac{(3.0+1.5\sqrt{n})x}{n} + 1.5 \rfloor$	$k = \lfloor \frac{(3.0+2.8\sqrt{n})x}{n} + 5.2 \rfloor$

$B_n = T_n, B_{n-1} = 1.0$ とし、次の漸化式を用いて B_i ($i = 1, 2, \dots, n-2$) を計算する.

$$B_{i-1} = \frac{2i}{x} B_i - B_{i+1} \quad (i = 2, \dots, n-2, n-1)$$

(B_1 は $\frac{J_1(x)}{J_{n-1}(x)}$ の値)

$J_0(x) \simeq 0$ または $J_1(x) \simeq 0$ のとき精度が低下するので次の処理を行う.

$$K = \lfloor 1.27324x + 3 \rfloor \pmod{4}$$

として

$K = 0$ または 3 のとき

$$B = \frac{2B_1}{x} - B_2, \quad J = J_0(x)$$

$K = 1$ または 2 のとき

$$B = B_1, \quad J = J_1(x)$$

得られた T_n と B と J の値より

$$J_n(x) = \frac{T_n J}{B}$$

とする.

ii. $x > n$ ならば

$J_1(x), J_0(x)$ を初期値とし、漸化式

$$J_{k+1}(x) = \frac{2k}{x} J_k(x) - J_{k-1}(x) \quad (k = 1, 2, \dots, n-1)$$

を用いて $J_n(x)$ を求める.

(4) 第 2 種整数次ベッセル関数 $Y_n(x)$ ($x > 0.0$)

① $n < 0$ のとき

$Y_n(x) = (-1)^n Y_{-n}(x)$ として $Y_{-n}(x)$ について以下に述べる方法を適用する.

② $n = 0, 1$ のときは $Y_0(x), Y_1(x)$ を利用する.

③ $n \geq 2$ のときは

$Y_1(x), Y_0(x)$ を初期値とし、漸化式

$$Y_{k+1}(x) = \frac{2k}{x} Y_k(x) - Y_{k-1}(x) \quad (k = 1, 2, \dots, n-1)$$

を用いて $Y_n(x)$ を求める.

(5) 第 1 種実数次ベッセル関数 $J_\nu(x)$

① ν が整数のとき $n = \nu$ として整数次ベッセル関数のサブルーチンを利用する.

② ν が整数でない実数のとき ($x > 0.0, \nu > 0.0$)

(a) $0.0 < x^2 < \left\{ \begin{array}{l} \text{倍精度: } 0.1\nu + 0.4 \\ \text{単精度: } 1.9\nu + 7.6 \end{array} \right\}$ で $\nu < \left\{ \begin{array}{l} \text{倍精度: } 170.0 \\ \text{単精度: } 34.0 \end{array} \right\}$ の場合

$$J_\nu(x) = \left(\frac{x}{2}\right)^\nu \sum_{k=0}^9 \frac{(-1)^k \left(\frac{x}{2}\right)^{2k}}{k! \Gamma(\nu + k + 1)}$$

のべき級数展開で求める.

(b) $x^2 \geq \left\{ \begin{array}{l} \text{倍精度: } 0.1\nu + 0.4 \\ \text{単精度: } 1.9\nu + 7.6 \end{array} \right\}$ または $\nu \geq \left\{ \begin{array}{l} \text{倍精度: } 170.0 \\ \text{単精度: } 34.0 \end{array} \right\}$ の場合

i. $x > \left\{ \begin{array}{l} \text{倍精度: } 30.0 \\ \text{単精度: } 15.0 \end{array} \right\}$ かつ $x \geq 0.55\nu^2$ ならば

$$J_\nu(x) = \sqrt{\frac{2}{\pi x}} (P \cos(\phi) - Q \sin(\phi))$$

の漸近展開式で求める. ここで

$$P = 1 + \sum_{k=1}^m (-1)^k \frac{(4\nu^2 - 1^2)(4\nu^2 - 3^2) \cdots (4\nu^2 - (4k - 1)^2)}{(2k)!(8x)^{2k}}$$

$$Q = \sum_{k=0}^{m'} (-1)^k \frac{(4\nu^2 - 1^2)(4\nu^2 - 3^2) \cdots (4\nu^2 - (4k + 1)^2)}{(2k + 1)!(8x)^{2k+1}}$$

$$\phi = x - \left(\frac{\nu}{2} + \frac{1}{4}\right)\pi$$

(m, m' は末項がそれ以前の計算値に影響しなくなるまでの値である.)

ii. 上述以外のとき次の漸化式 (逆行法) より求める.

δ を ν の小数部, n をその整数部, M を十分大きい値, a を正の最小値 (浮動小数点データの内部表現であらわすことのできる最も 0.0 に近い正の値) とする.

$F_{\delta+M+1}(x) = 0, F_{\delta+M}(x) = a$ を初期値とし

$$F_{\delta+k-1}(x) = \frac{2(\delta+k)}{x} F_{\delta+k}(x) - F_{\delta+k+1}(x) \quad (k = M, M-1, \dots, 1)$$

として

$$J_\nu(x) = \frac{F_{\delta+n}(x) \left(\frac{x}{2}\right)^\delta}{\sum_{n=0}^{\lfloor \frac{M}{2} \rfloor} \frac{(\delta+2m)\Gamma(\delta+m)}{m!} F_{\delta+2m}(x)}$$

M の値は x, ν の値より近似式を使って決める.

(6) 第 2 種実数次ベッセル関数 $Y_\nu(x)$ ($x > 0.0$)

① ν が整数のとき $n = \nu$ として整数次ベッセル関数のサブルーチンを利用する.

② ν が整数でない実数のとき ($\nu > 0.0$)

ν を整数部 n と小数部 δ に分解する.

(a) $0.0 < x \leq 4.0$ の場合

$$Y_\nu(x) = \frac{J_\nu(x) \cos(\nu\pi) - J_{-\nu}(x)}{\sin(\nu\pi)}$$

に $J_\nu(x), J_{-\nu}(x)$ のべき級数展開式を代入し, 項をまとめ, 桁落ちが生ずる部分を最良近似式化する吉田・二宮の方法で計算する. 以下にその内容を示す (参考文献 (5) 参照).

i. $0.0 < \nu \leq 0.5$ では

$$Y_\nu(x) = \sum_{k=0}^{\infty} - \left(-\frac{x^2}{4}\right)^k \frac{\tilde{A}_k(\nu) + \tilde{B}_k(\nu)}{\sin(\nu\pi)}$$

より計算する. $\tilde{A}_k(\nu)$ は

$$\tilde{A}_0(\nu) = (\nu - \nu_0) \sum_{k=0}^M P_k^{(1)} \nu^k \quad (\nu_0 = 0.221521 \dots)$$

を初期値として, 漸化式

$$\tilde{A}_k(\nu) = \frac{1}{k!} \left\{ \frac{1}{\Gamma(k-\nu)} + \frac{\cos(\nu\pi)}{\Gamma(k+\nu)} \right\} + \tilde{A}_{k-1}(\nu)$$

$$(k+\nu)(k-\nu)$$

で計算する. $\tilde{B}_k(\nu)$ は

$$\tilde{B}_k(\nu) = \frac{1}{k!} \left\{ \frac{\phi_1}{\Gamma(k+1-\nu)} + \frac{\phi_2 \cos(\nu\pi)}{\Gamma(k+1+\nu)} \right\}$$

として計算する. ここで, $(\frac{x}{2})^\nu < 0.5$ または $(\frac{x}{2})^\nu > 2.0$ のときは,

$$\phi_1 = \frac{(\frac{x}{2})^{-\nu} - 1}{\nu}, \quad \phi_2 = \frac{1 - (\frac{x}{2})^\nu}{\nu}$$

とし, また $0.5 \leq (\frac{x}{2})^\nu \leq 2.0$ のときは,

$$\phi_1 = -f(-\nu \log(\frac{x}{2})) \log(\frac{x}{2}), \quad \phi_2 = -f(\nu \log(\frac{x}{2})) \log(\frac{x}{2})$$

とする. $f(t)$ は $\frac{e^t - 1}{t}$ の関数の最良近似式を用いて計算する.

ii. $0.5 < \nu \leq 1.5$ ならば

A. $\delta \leq 0.5$ のとき $\delta = \delta + 1$, $n = n - 1$ として $0.5 < \delta \leq 1.5$ の範囲で計算する.

B. $0.5 < \delta \leq 1.5$ のとき

$$Y_\delta(x) = - \frac{\frac{2^{1+\alpha}}{x^{1+\alpha}\Gamma(1-\alpha)} + \sum_{k=0}^{\infty} \left\{ \frac{x}{2} \left(-\frac{x^2}{4} \right)^k (\tilde{C}_k(\alpha) + \tilde{D}_k(\alpha)) \right\}}{\sin(\alpha\pi)}$$

とし ($\alpha = \delta - 1$ とする), $\tilde{C}_k(\alpha)$ や $\tilde{D}_k(\alpha)$ を $0.0 < \nu \leq 0.5$ のときと同様に最良近似式を利用して求める.

さらに

$$Y_{\delta+1}(x) = \frac{-\frac{2^\alpha \{4(\alpha+1) + x^2\}}{x^{\alpha+2}\Gamma(1-\alpha)} + \sum_{k=0}^{\infty} \left\{ \left(-\frac{x^2}{4} \right)^{k+1} (\tilde{E}_k(\alpha) + \tilde{F}_k(\alpha)) \right\}}{\sin(\alpha\pi)}$$

とし, $\tilde{E}_k(\alpha)$ や $\tilde{F}_k(\alpha)$ を $0.0 < \nu \leq 0.5$ のときと同様に最良近似式を利用して求める.

iii. $\nu > 1.5$ ならば

ii. で得られた $Y_\delta(x)$ と $Y_{\delta+1}(x)$ から, 漸化式

$$Y_{k+\delta+1}(x) = \frac{2(k+\delta)}{x} Y_{k+\delta}(x) - Y_{k+\delta-1}(x) \quad (k = 1, 2, \dots, n-1)$$

を用いて $Y_\nu(x)$ を得る.

(b) $4.0 < x \leq \begin{cases} \text{倍精度: 30.0} \\ \text{単精度: 15.0} \end{cases}$ の場合

i. $0.0 < \nu < 2.0$ ならば

δ または $1 - \delta$ が $\sqrt{\text{誤差判定のための単位}/4}$ より小なる場合

n または $n + 1$ を次数とする整数次ベッセル関数で近似する.

漸化式 (逆行法) により $J_\delta(x)$ と $J_{\delta+1}(x)$ を求める. また $t = 1 - \delta$ とし,

同様に $J_t(x)$, $J_{t+1}(x)$ を求め,

$$J_{-\delta}(x) = \frac{2(1-\delta)}{x} J_t(x) - J_{t+1}(x)$$

として $J_{-\delta}(x)$ を計算する. ((5) $J_\nu(x)$ 参照).

$$Y_\delta(x) = \frac{J_\delta(x) \cos(\delta\pi) - J_{-\delta}(x)}{\sin(\delta\pi)}$$

$$Y_{\delta+1}(x) = \frac{J_{\delta+1}(x) \cos(\delta\pi) - \frac{2\delta J_{-\delta}(x)}{x} - J_{\delta}(x)}{\sin(\delta\pi)}$$

として $Y_{\delta}(x)$ と $Y_{\delta+1}(x)$ を得る.

ii. $\nu \geq 2.0$ ならば

i. 得られた $Y_{\delta}(x)$ と $Y_{\delta+1}(x)$ より漸化式

$$Y_{k+\delta+1}(x) = \frac{2(k+\delta)}{x} Y_{k+\delta}(x) - Y_{k+\delta-1}(x) \quad (k = 1, 2, \dots, n-1)$$

を用いて $Y_{\nu}(x)$ を得る.

(c) $x > \begin{cases} \text{倍精度: 30.0} \\ \text{単精度: 15.0} \end{cases}$ の場合

i. $x \leq 0.55\nu^2$ ならば

$$Y_{\nu}(x) = \sqrt{\frac{2}{\pi x}} (P \sin(\phi) + Q \cos(\phi))$$

の漸近展開式で求める. ここで P, Q, ϕ は (5) $J_{\nu}(x)$ の計算時と同様にして求める.

ii. $x > 0.55\nu^2$ ならば

$m = \lfloor \sqrt{\frac{x}{0.55}} \rfloor - 1$ とする.

$Y_{m+\delta}(x)$ と $Y_{m+\delta+1}(x)$ を上の漸近展開式で求め, 漸化式

$$Y_{m+\delta+k+1}(x) = \frac{2(m+\delta+k)}{x} Y_{m+\delta+k}(x) - Y_{m+\delta+k-1}(x) \quad (k = 1, 2, \dots, n-m-1)$$

により $Y_{\nu}(x)$ を得る.

(7) 複素変数第 1 種整数次ベッセル関数 $J_n(z)$

$$J_n(z) = (-i)^n I_n(iz) \quad (i = \sqrt{-1})$$

より求める.

(8) 複素変数第 2 種整数次ベッセル関数 $Y_n(z)$ ($|z| > 0.0$)

① $n < 0$ のとき

$Y_n(z) = (-1)^n Y_{-n}(z)$ として $Y_{-n}(z)$ について以下に述べる方法を適用する.

② $n \geq 0$ のとき

z の虚部が負ならば $Y_n(\bar{z}) = \overline{Y_n(z)}$ として求める.

$$Y_n(z) = i^{n+1} I_n(-iz) - \frac{2}{\pi} (-i)^n K_n(-iz) \quad (i = \sqrt{-1})$$

より求める.

2.1.2.2 変形ベッセル関数

(1) 第 1 種 0 次, 1 次変形ベッセル関数 $I_0(x), I_1(x)$

$x < 0.0$ のとき

$I_0(x) = I_0(-x), I_1(x) = -I_1(-x)$ として $I_0(-x), I_1(-x)$ について以下に述べる方法を適用する.

・単精度

① $0.0 \leq x \leq 3.75$ のとき次の式を最良近似式化して求める.

$$I_0(x) = \sum_{k=0}^{\infty} \frac{1}{(k!)^2} \left(\frac{x}{2}\right)^{2k}$$

$$I_1(x) = \sum_{k=0}^{\infty} \frac{1}{k!(k+1)!} \left(\frac{x}{2}\right)^{2k+1}$$

② $x > 3.75$ のとき次の近似式で求める.

$$I_0(x) = \frac{\sum_{n=0}^8 a_n^{(1)} \left(\frac{3.75}{x}\right)^n}{e^{-x}\sqrt{x}}$$

$$I_1(x) = \frac{\sum_{n=0}^8 a_n^{(2)} \left(\frac{3.75}{x}\right)^n}{e^{-x}\sqrt{x}}$$

係数 $a_n^{(1)}, a_n^{(2)}$ は文献 (1) に記述されている.

・倍精度

① $0.0 \leq x \leq 8.0$ のとき次の式を最良近似式化して求める.

$$I_0(x) = \sum_{k=0}^{\infty} \frac{1}{(k!)^2} \left(\frac{x}{2}\right)^{2k}$$

$$I_1(x) = \sum_{k=0}^{\infty} \frac{1}{k!(k+1)!} \left(\frac{x}{2}\right)^{2k+1}$$

② $8.0 < x < 24.0$ のとき, 次の近似式で求める.

$$I_0(x) = \frac{\sum_{n=0}^{28} a_n^{(3)} x^{-n}}{e^{-x}\sqrt{x}}$$

$$I_1(x) = \frac{\sum_{n=0}^{28} a_n^{(4)} x^{-n}}{e^{-x}\sqrt{x}}$$

③ $x \geq 24.0$ のとき, 次の式を最良近似化して求める.

$$I_0(x) = \frac{e^x}{\sqrt{2\pi x}} \sum_{k=0}^{\infty} \frac{1^2 \cdot 1^2 \cdot 3^2 \cdots (2k-1)^2}{k!(8x)^k}$$

$$I_1(x) = \frac{e^x}{\sqrt{2\pi x}} \left\{ 1 + \sum_{k=1}^{\infty} \frac{(-3) \cdot 5 \cdots ((2k-1)^2 - 4)}{k!(8x)^k} \right\}$$

係数 $a_n^{(3)}, a_n^{(4)}$ は文献 (21) を参考にテレスコーピング計算をして求めた.
最良近似式化については 2.1.2.22 に記述されている.

(2) 第 2 種 0 次, 1 次変形ベッセル関数 $K_0(x), K_1(x)$ ($x > 0.0$)

・単精度

① $0.0 < x \leq 2.0$ のとき次の式を最良近似式化して求める.

$$K_0(x) = -\gamma + \sum_{k=1}^{\infty} \frac{1}{(k!)^2} \left(\frac{x}{2}\right)^{2k} \left\{ \left(\sum_{m=1}^k \frac{1}{m}\right) - \gamma \right\} - \log\left(\frac{x}{2}\right) \left\{ \sum_{k=0}^{\infty} \frac{1}{(k!)^2} \left(\frac{x}{2}\right)^{2k} \right\}$$

$$K_1(x) = \frac{1 + \sum_{k=0}^{\infty} \frac{1}{k!(k+1)!} \left(\frac{x}{2}\right)^{2k+2} \left(2\gamma - \sum_{m=1}^k \frac{1}{m} - \sum_{m=1}^{k+1} \frac{1}{m}\right)}{x} + \log\left(\frac{x}{2}\right) \sum_{k=0}^{\infty} \frac{1}{k!(k+1)!} \left(\frac{x}{2}\right)^{2k+1}$$

(ただし, $\sum_{m=1}^0 \frac{1}{m} = 0$)

② $x > 2.0$ のとき次の近似式で求める.

$$K_0(x) = \frac{e^{-x}}{\sqrt{x}} \sum_{n=0}^6 a_n^{(1)} \left(\frac{2}{x}\right)^n$$

$$K_1(x) = \frac{e^{-x}}{\sqrt{x}} \sum_{n=0}^6 a_n^{(2)} \left(\frac{2}{x}\right)^n$$

係数 $a_n^{(1)}, a_n^{(2)}$ は文献 (1) に記述されている.

・倍精度

① $0.0 < x \leq 2.0$ のとき次の式を最良近似式化して求める.

$$K_0(x) = -\gamma + \sum_{k=1}^{\infty} \frac{1}{(k!)^2} \left(\frac{x}{2}\right)^{2k} \left\{ \left(\sum_{m=1}^k \frac{1}{m}\right) - \gamma \right\} - \log\left(\frac{x}{2}\right) \left\{ \sum_{k=0}^{\infty} \frac{1}{(k!)^2} \left(\frac{x}{2}\right)^{2k} \right\}$$

$$K_1(x) = \frac{1 + \sum_{k=0}^{\infty} \frac{1}{k!(k+1)!} \left(\frac{x}{2}\right)^{2k+2} \left(2\gamma - \sum_{m=1}^k \frac{1}{m} - \sum_{m=1}^{k+1} \frac{1}{m}\right)}{x} + \log\left(\frac{x}{2}\right) \sum_{k=0}^{\infty} \frac{1}{k!(k+1)!} \left(\frac{x}{2}\right)^{2k+1}$$

(ただし, $\sum_{m=1}^0 \frac{1}{m} = 0$)

ここで γ はオイラーの定数で $0.5772\dots$ である.

② $2.0 < x \leq 5.0$ のとき次の式を最良近似式化して求める.

$$K_0(x) = \sum_{k=0}^{\infty} \frac{K_0^{(k)}(3.5)}{k!} (x - 3.5)^k$$

$$K_1(x) = \sum_{k=0}^{\infty} \frac{K_1^{(k)}(3.5)}{k!} (x - 3.5)^k$$

ここで $K_0^{(k)}(3.5), K_1^{(k)}(3.5)$ は $x = 3.5$ での k 階微分値である. また $K_0^{(k)}(3.5), K_1^{(k)}(3.5)$ 計算法は文献 (19) を参考に次のように計算できる.

$$t_0 = 1, t_1 = 0, t_2 = 0, u_0 = 0, u_1 = 1$$

$$v_0 = 0, v_1 = 0, w_0 = 0, w_1 = 0, w_2 = -1$$

$$i = 3, x = 3.5$$

$$\text{反復} \left[\begin{array}{l} t'_j = v_j - u_j \quad (j = 0, \dots, i-2) \\ t'_{i-1} = 0, t'_i = 0 \\ u'_0 = w_0 - t_0 \\ u'_j = w_j - t_j - u_{j-1} \quad (j = 1, \dots, i-1) \\ t_j = t'_j \quad (j = 0, \dots, i), u_j = u'_j \quad (j = 0, \dots, i-1) \\ K_0^{(i)}(x) = \sum_{j=0}^{i-2} t_j x^{-j} K_0(x) + \sum_{j=0}^{i-1} u_j x^{-j} K_1(x) \\ v_{j+1} = -j t_j \quad (j = 1, i-2) \\ w_{j+1} = -j u_j \quad (j = 1, i-1) \\ i = i + 1 \end{array} \right.$$

$$K_1^{(i)}(x) = -K_0^{(i+1)}(x), K_0''(x) = K_0(x) + \frac{K_1(x)}{x}, K_0'(x) = -K_1(x)$$

③ $5.0 < x < 24.0$ のとき

$$K_0(x) = \frac{e^{-x}}{\sqrt{x}} \sum_{n=0}^{18} a_n^{(3)} x^{-n}$$

$$K_1(x) = \frac{e^{-x}}{\sqrt{x}} \sum_{n=0}^{18} a_n^{(4)} x^{-n}$$

係数 $a_n^{(3)}, a_n^{(4)}$ は文献 (21) を参考にテレスコーピング計算をして求めた。

④ $x \geq 24.0$ のとき、次の式を最良近似式化して求める。

$$K_0(x) = \sqrt{\frac{\pi}{2x}} e^{-x} \sum_{k=0}^{\infty} (-1)^k \frac{1^2 \cdot 1^2 \cdot 3^2 \cdots (2k-1)^2}{k!(8x)^k}$$

$$K_1(x) = \sqrt{\frac{\pi}{2x}} e^{-x} \left\{ 1 + \sum_{k=1}^{\infty} \frac{3 \cdot (-5) \cdots (4 - (2k-1)^2)}{k!(8x)^k} \right\}$$

最良近似式化については、2.1.2.22 に記述されている。

(3) 第 1 種整数次変形ベッセル関数 $I_n(x)$

$I_n(x)$ は x, n の符号に応じて $I_{|n|}(|x|)$ を用いて表 2-3 のように表せる。したがって以下に、 $I_{|n|}(|x|)$ の計算法

表 2-3 x, n の符号と $I_n(x)$ の計算方法

—	$n < 0$	$n \geq 0$
$x < 0.0$	$(-1)^n I_{ n }(x)$	$(-1)^n I_{ n }(x)$
$x \geq 0.0$	$I_{ n }(x)$	$I_{ n }(x)$

について記述する。表記を簡略化するために $|n| \rightarrow n, |x| \rightarrow x$ と表す。

① $n = 0, 1$ のときは $I_0(x), I_1(x)$ を利用する。

② $n \geq 2$ のとき

(a) $0.0 \leq x^2 < \left\{ \begin{array}{l} \text{倍精度: } 0.1n + 0.4 \\ \text{単精度: } 1.9n + 7.6 \end{array} \right\}$ で $n < \left\{ \begin{array}{l} \text{倍精度: } 170 \\ \text{単精度: } 34 \end{array} \right\}$ の場合

$$I_n(x) = \left(\frac{x}{2}\right)^n \sum_{k=0}^9 \frac{\left(\frac{x}{2}\right)^{2k}}{k!(n+k)!}$$

のべき級数展開で求める。

(b) $x^2 \geq \left\{ \begin{array}{l} \text{倍精度: } 0.1n + 0.4 \\ \text{単精度: } 1.9n + 7.6 \end{array} \right\}$ または $n \geq \left\{ \begin{array}{l} \text{倍精度: } 170 \\ \text{単精度: } 34 \end{array} \right\}$ の場合

i. $x \leq n$ ならば

$T_{n+k+1} = 0.0$ とし、次の漸化式を用いて連分数近似におけるパラメータ T_i ($i = n, n+1, \dots, n+k-1, n+k$) を計算する。

$$T_i = \frac{x^2}{2i + T_{i+1}} \quad (i = n+1, \dots, n+k-1, n+k)$$

$$T_n = \frac{x}{2n + T_{n+1}}$$

(T_n は $\frac{I_n(x)}{I_{n-1}(x)}$ の値)

k の値は表 2-4 に示すようにとる。 $B_n = T_n, B_{n-1} = 1.0$ とし、次の漸化式を用いて B_i ($i = 1, 2, \dots, n-2$) を計算する。

$$B_{i-1} = \frac{2i}{x} B_i + B_{i+1} \quad (i = 2, \dots, n-2, n-1)$$

表 2-4 k の値

単精度演算	倍精度演算
$k = \lfloor \frac{(4.6+0.5\sqrt{n})x}{n} + 2.0 \rfloor$	$k = \lfloor \frac{(6.0+1.2\sqrt{n})x}{n} + 6.0 \rfloor$

$(B_1$ は $\frac{I_1(x)}{I_{n-1}(x)}$ の値)

得られた T_n と B_1 と $I_1(x)$ の値より

$$I_n(x) = \frac{T_n I_1(x)}{B_1}$$

とする。

ii. $x > n$ ならば

a を正の最小値 (浮動小数点データの内部表現であらわすことのできる最も 0.0 に近い正の値), M を n より十分大きい値とする。

$G_{M+1}(x) = 0.0, G_M(x) = a$ を初期値とし

$$G_{k-1}(x) = \frac{2k}{x} G_k(x) + G_{k+1}(x) \quad (k = M, M-1, \dots, 1)$$

として

$$I_n(x) = \frac{G_n(x)e^x}{\sum_{m=0}^M \varepsilon_m G_m(x)} \quad (\varepsilon_0 = 1, \varepsilon_m = 2 \quad (m \geq 1))$$

とする。

(4) 第 2 種整数次変形ベッセル関数 $K_n(x)$ ($x > 0.0$)

① $n < 0$ のとき

$K_n(x) = K_{-n}(x)$ として $K_{-n}(x)$ について以下に述べる方法を適用する。

② $n = 0, 1$ のときは $K_0(x), K_1(x)$ を利用する。

③ $n \geq 2$ のときは

$K_1(x), K_0(x)$ を初期値とし, 漸化式

$$K_{k+1}(x) = \frac{2k}{x} K_k(x) + K_{k-1}(x) \quad (k = 1, 2, \dots, n-1)$$

を用いて $K_n(x)$ を求める。

(5) 第 1 種実数次変形ベッセル関数 $I_\nu(x)$

① ν が整数のとき $n = \nu$ として整数次変形ベッセル関数のサブルーチンを利用する。

② ν が整数でない実数のとき ($x > 0.0, \nu > 0.0$)

(a) $0.0 \leq x^2 < \left\{ \begin{array}{l} \text{倍精度: } 0.1\nu + 0.4 \\ \text{単精度: } 1.9\nu + 7.6 \end{array} \right\}$ で $\nu < \left\{ \begin{array}{l} \text{倍精度: } 170.0 \\ \text{単精度: } 34.0 \end{array} \right\}$ の場合

$$I_\nu(x) = \left(\frac{x}{2}\right)^\nu \sum_{k=0}^9 \frac{\left(\frac{x}{2}\right)^{2k}}{k! \Gamma(\nu + k + 1)}$$

のべき級数展開で求める。

(b) $x^2 \geq \left\{ \begin{array}{l} \text{倍精度: } 0.1\nu + 0.4 \\ \text{単精度: } 1.9\nu + 7.6 \end{array} \right\}$ または $\nu \geq \left\{ \begin{array}{l} \text{倍精度: } 170.0 \\ \text{単精度: } 34.0 \end{array} \right\}$ の場合

i. $x > \left\{ \begin{array}{l} \text{倍精度: } 30.0 \\ \text{単精度: } 15.0 \end{array} \right\}$ かつ $x \geq 0.55\nu^2$ ならば

$$I_\nu(x) = \frac{e^x}{\sqrt{2\pi x}} \sum_{k=0}^m (-1)^k \frac{(4\nu^2 - 1^2)(4\nu^2 - 3^2) \dots (4\nu^2 - (2k-1)^2)}{k! (8x)^k}$$

の漸近展開式で求める.

(m は末項がそれ以前の計算値に影響しなくなるまでの値である.)

ii. x が前項以外のとき次の漸化式 (逆行法) より求める.

δ を ν の小数部, n をその整数部, M を十分大きい値, a を正の最小値 (浮動小数点データの内部表現であらわすことのできる最も 0.0 に近い正の値) とする.

$G_{\delta+M+1}(x) = 0.0, G_{\delta+M}(x) = a$ を初期値とし,

$$G_{\delta+k-1}(x) = \frac{2(\delta+k)}{x} G_{\delta+k}(x) + G_{\delta+k+1}(x) \quad (k = M, M-1, \dots, 1)$$

として

$$I_\nu(x) = \frac{1}{2} \left(\frac{x}{2}\right)^\delta \frac{\Gamma(2\delta+1)}{\Gamma(\delta+1)} e^x \frac{G_{n+\delta}(x)}{\sum_{k=0}^M \frac{(\delta+k)\Gamma(2\delta+k)}{k!} G_{\delta+k}(x)}$$

M の値は x, ν の値より近似式を使って決める.

(6) 第 2 種実数変形ベッセル関数 $K_\nu(x)$ ($x > 0.0$)

① $\nu < 0.0$ のとき

$K_\nu(x) = K_{-\nu}(x)$ として $K_{-\nu}(x)$ について以下に述べる方法を適用する.

② ν が整数のとき $n = \nu$ として整数変形ベッセル関数のサブルーチンを利用する.

③ ν が整数でない実数のとき

x が小さい場合は

$$K_\nu(x) = \frac{\pi I_{-\nu}(x) - I_\nu(x)}{2 \sin(\nu\pi)}$$

に $I_\nu(x), I_{-\nu}(x)$ の級数展開を代入し, 項をまとめ, 桁落ちが生ずる部分を最良近似化する吉田・二宮の方法で計算する.

x が大きい場合は τ -method による $K_n(x)$ の計算法を $K_\nu(x)$ 計算に拡張した方法を利用する. こうして $0.0 \leq \nu \leq 2.5$ の区間の $K_\nu(x)$ を計算し, $\nu > 2.5$ については, 漸化式

$$K_{\nu+1}(x) = \frac{2\nu}{x} K_\nu(x) + K_{\nu-1}(x)$$

により求める. 以下にその内容を示す (文献 (8) 参照)

(a) $0.0 \leq \nu \leq 0.5$ の場合

i. $x < -0.75\nu^2 + 0.0235\nu + 0.778$ のとき

$$K_\nu(x) = \frac{\sum_{k=0}^{\infty} \left\{ \left(\frac{x}{2}\right)^{2k} (\tilde{A}_k(\nu) + \tilde{B}_k(\nu)) \right\}}{\frac{\pi}{2} \sin(\nu\pi)}$$

により計算する. ただし, $\tilde{A}_k(\nu)$ は $\tilde{A}_0(\nu) = \nu \sum_{k=0}^M p_k^{(1)} \nu^{2k}$, $\tilde{A}_1(\nu) = \nu \sum_{k=0}^M q_k^{(1)} \nu^{2k}$ とし,

$\tilde{A}_0(\nu), \tilde{A}_1(\nu)$ を初期値として, $k \geq 2$ について, 漸化式

$$\tilde{A}_k(\nu) = \frac{\nu \left\{ \frac{1}{\Gamma(k-\nu)} + \frac{1}{\Gamma(k+\nu)} \right\} + \tilde{A}_{k-1}(\nu)}{(k+\nu)(k-\nu)}$$

から求める.

$\tilde{B}_k(\nu)$ は

$$\begin{aligned} \phi_1 &= \left(\frac{x}{2}\right)^{-\nu} - 1, \quad \phi_2 = 1 - \left(\frac{x}{2}\right)^\nu \\ & \left(\left(\frac{x}{2}\right)^\nu < 0.5 \text{ または } \left(\frac{x}{2}\right)^\nu > 2.0 \text{ のとき}\right) \end{aligned}$$

$$\phi_1 = f(-\nu \log(\frac{x}{2})), \phi_2 = -f(\nu \log(\frac{x}{2}))$$

$$(0.5 \leq (\frac{x}{2})^\nu \leq 2.0 \text{ のとき})$$

とし, $f(t)$ を $e^t - 1$ の関数の最良近似式を用いて計算し,

$$\tilde{B}_k(\nu) = \frac{1}{k!} \left(\frac{\phi_1}{\Gamma(k+1-\nu)} + \frac{\phi_2}{\Gamma(k+1+\nu)} \right)$$

として求める. p_k, q_k の係数は文献 (8) を参照して計算できる.

ii. $x \geq -0.75\nu^2 + 0.0235\nu + 0.778$ のとき

$$K_\nu(x) = \sqrt{\frac{1}{x}} e^{-x} \frac{\sum_{i=0}^m \left(\frac{1}{x}\right)^i \left\{ \sum_{j=0}^i b_{ij} (\nu^2)^j \right\}}{\sum_{i=0}^m \left(\frac{1}{x}\right)^i e_i \Psi_i}$$

の τ -method の式により計算する. ただし, b_{ij}, e_i の係数は文献 (20) を参照して計算できる.

また, Ψ_i は,

$$\Psi_0 = 1, \Psi_i = \prod_{l=0}^{i-1} \left\{ \nu^2 - \left(m - l + \frac{1}{2}\right)^2 \right\} \quad (i \geq 1)$$

として求める. ($e_i = \sqrt{\frac{2}{\pi}} \frac{(m-i)! p_{m,m-i}^*}{(m+1)! 2^i}$ で $p_{m,m-i}^*$ は, ずらしルジャンドル多項式の係数)

(b) $0.5 < \nu \leq 2.5$ の場合

ν を整数部 n と小数部 δ に分解する. $\alpha = \delta - 1$ とする.

$\delta \leq 0.5$ のときは $\delta = \delta + 1, n = n - 1, \alpha = \alpha + 1$ とする.

i. K_δ の計算を以下の A. または B. の方法で行う.

A. $x < -0.675\delta^2 + 1.973\delta - 0.12$ のとき

$$K_\delta(x) = - \frac{\frac{2^{1+\alpha}}{x^{1+\alpha}\Gamma(1-\alpha)} + \sum_{k=0}^{\infty} \left\{ \left(\frac{x}{2}\right)^{2k+1} (\tilde{C}_k(\alpha) + \tilde{D}_k(\alpha)) \right\}}{\frac{\pi}{2} \sin(\alpha\pi)}$$

とし, $\tilde{C}_k(\alpha)$ や $\tilde{D}_k(\alpha)$ を $\nu \leq 0.5$ の場合と同様に最良近似式を利用して求める.

B. $x \geq -0.675\delta^2 + 1.973\delta - 0.12$ のとき

$\nu \leq 0.5$ で x が大きい場合と同様に τ -method により $K_\delta(x)$ を計算する.

ii. $K_{\delta+1}$ の計算を以下の A. または B. の方法で行う.

A. $x < -0.277(\delta+1)^2 + 1.817(\delta+1) - 0.94$ のとき

$$K_{\delta+1}(x) = \frac{\frac{2^\alpha \alpha (4\alpha + 5)}{x^\alpha \Gamma(1-\alpha)} + \sum_{k=0}^{\infty} \left\{ \left(\frac{x}{2}\right)^{2k+2} (\tilde{E}_k(\alpha) + \tilde{F}_k(\alpha)) \right\}}{\frac{\pi}{2} \sin(\alpha\pi)}$$

とし, $\tilde{E}_k(\alpha)$ や $\tilde{F}_k(\alpha)$ を $\nu \leq 0.5$ の場合と同様に最良近似式を利用して求める.

B. $x \geq -0.277(\delta+1)^2 + 1.817(\delta+1) - 0.94$ のとき

$\nu \leq 0.5$ で x が大きい場合と同様に τ -method により $K_{\delta+1}(x)$ を計算する.

(c) $\nu > 2.5$ の場合

上記で得られた $K_\delta(x)$ と $K_{\delta+1}(x)$ から漸化式

$$K_{k+\delta+1}(x) = \frac{2(k+\delta)}{x} K_{k+\delta}(x) + K_{k+\delta-1}(x) \quad (k = 1, 2, \dots, n-1)$$

を用いて $K_\nu(x)$ を計算する.

(7) 複素変数第 1 種整数次変形ベッセル関数 $I_n(z)$

① $n < 0$ のとき

$I_n(z) = I_{-n}(z)$ として $I_{-n}(z)$ について以下に述べる方法を適用する.

② $n \geq 0$ のとき

(a) z の実部が負ならば $I_n(z) = (-1)^n I_n(-z)$ として $I_n(-z)$ について以下に述べる方法を適用する.

(b) z の実部が正であり,

i. $|z|^2 < \begin{cases} \text{倍精度: } 0.1n + 0.4 \\ \text{単精度: } 1.9n + 7.6 \end{cases}$ ならば

$$I_n(z) = \left(\frac{z}{2}\right)^n \sum_{k=0}^9 \frac{\left(\frac{z}{2}\right)^{2k}}{k!(n+k)!}$$

のべき級数展開で求める.

ii. $|z|^2 \geq \begin{cases} \text{倍精度: } 0.1n + 0.4 \\ \text{単精度: } 1.9n + 7.6 \end{cases}$ の場合

A. $\Re(z) > 100.0$ または $|\Im(z)| > 100.0$ で $n \leq 15$ ならば漸近展開式

$$I_n(z) = \frac{e^z}{\sqrt{2\pi z}} \sum_{k=0}^m (-1)^k \frac{(4n^2 - 1^2)(4n^2 - 3^2) \cdots (4n^2 - (2n-1)^2)}{k!(8z)^k} \\ + \frac{i(-1)^n e^{-z}}{\sqrt{2\pi z}} \sum_{k=0}^{m'} \frac{(4n^2 - 1^2)(4n^2 - 3^2) \cdots (4n^2 - (2n-1)^2)}{k!(8z)^k}$$

(m, m' は末項がそれ以前の計算値に影響しなくなるまでの値である.)

B. それ以外のとき次の漸化式 (逆行法) より求める.

十分大きい M をとり, a を正の最小値 (浮動小数点データの内部表現であらわすことのできる最も 0.0 に近い正の値) とする.

$G_{M+1}(z) = 0.0, G_M(z) = a$ を初期値とし,

$$G_{k-1}(z) = \frac{2k}{z} G_k(z) + G_{k+1}(z) \quad (k = M, M-1, \dots, 1)$$

として

$$I_n(z) = \frac{G_n(z)e^z}{\sum_{m=0}^M \varepsilon_m G_m(z)} \quad (\varepsilon_0 = 1, \varepsilon_m = 2 \ (m \geq 1))$$

M の値は z, n の値より近似式を使って決める (文献 (6), (7) 参照).

(8) 複素変数第 2 種整数次変形ベッセル関数 $K_n(z)$ ($|z| > 0.0$)

① $\Re(z) < 0$ 負のとき

$$K_n(z) = (-1)^n K_n(-z) + \pi i I_n(-z) \cdot (\Im(-z) \text{ の符号})$$

として $K_n(-z)$ について以下に述べる方法を適用する.

② $\Re(z) > 0$ のとき

(a) $n < 0$ ならば

$K_n(z) = K_{-n}(z)$ として $K_{-n}(z)$ について以下に述べる方法を適用する.

(b) $0 \leq n < 2$ であり

i. $|\Im(z)| < \begin{cases} \text{倍精度: } -4.0\Re(z) + 8.0 \\ \text{単精度: } -2.25\Re(z) + 4.5 \end{cases}$ のとき

$$K_0(z) = -\{\gamma + \log\left(\frac{z}{2}\right)\} I_0(z) + \sum_{k=1}^n \frac{\left(\frac{z}{2}\right)^{2k}}{(k!)^2} \left(\sum_{m=1}^k \frac{1}{m}\right)$$

$$K_1(z) = \frac{\frac{1}{z} - I_1(z)K_0(z)}{I_0(z)}$$

(ここで, γ はオイラーの定数) により $K_0(z), K_1(z)$ を計算する.

ii. $|\Im(z)| \geq \begin{cases} \text{倍精度: } -4.0\Re(z) + 8.0 \\ \text{単精度: } -2.25\Re(z) + 4.5 \end{cases}$ のとき

$$K_n(z) = \sqrt{\frac{1}{z}} e^{-z} \frac{\sum_{k=0}^m c_k z^{k-m}}{\sum_{k=0}^m d_k z^{k-m}}$$

の τ -method により $K_0(z)$ と $K_1(z)$ を計算する。 $K_0(z)$ および $K_1(z)$ に対応する係数 c_k, d_k は文献 (9) 参照。

(c) $n \geq 2$ の場合, $K_0(z)$ と $K_1(z)$ より漸化式

$$K_{k+1}(z) = \frac{2k}{z} K_k(z) + K_{k-1}(z) \quad (k = 1, 2, \dots, n-1)$$

にて求める。

2.1.2.3 球ベッセル関数

(1) 第 1 種整数次球ベッセル関数 $j_n(x)$ ($x \geq 0.0$)

① $n < 0$ のとき

$$j_n(x) = (-1)^n y_{-n-1}(x).$$

より求める。

② $n \geq 0$ のとき

(a) $x^2 < 0.1n + 0.47$ で $n < 35$ のとき

$$j_n(x) = x^n \sum_{k=0}^9 \frac{(-\frac{x^2}{2})^k}{k!(2n+2k+1)!!}$$

のべき級数展開で求める。

(b) $x^2 \geq 0.1n + 0.47$ または $n \geq 35$ のとき

i. $n = 0, 1$ の場合

$$j_0(0) = 1, j_1(0) = 0,$$

$$j_0(x) = \frac{\sin(x)}{x}, j_1(x) = \frac{\sin(x) - x \cos(x)}{x^2}$$

により計算する。

ii. $n \geq 2$ の場合

A. $x \leq n$ ならば

$T_{n+k+1} = 0.0$ とし, 次の漸化式を用いて連分数近似におけるパラメータ T_i ($i = n, n+1, \dots, n+k-1, n+k$) を計算する。

$$T_i = \frac{x^2}{2i - T_{i+1} + 1} \quad (i = n+1, \dots, n+k-1, n+k)$$

$$T_n = \frac{x}{2n - T_{n+1} + 1}$$

(T_n は $\frac{j_n(x)}{j_{n-1}(x)}$ の値)

k の値は表 2-5 に示すようにとる。

表 2-5 k の値

単精度演算	倍精度演算
$k = \lfloor \frac{(3.0+1.5\sqrt{n})x}{n} + 1.5 \rfloor$	$k = \lfloor \frac{(3.0+2.8\sqrt{n})x}{n} + 5.2 \rfloor$

$B_n = T_n, B_{n-1} = 1.0$ とし, 次の漸化式を用いて B_i ($i = 1, 2, \dots, n-2$) を計算する.

$$B_{i-1} = \frac{2i+1}{x} B_i - B_{i+1} \quad (i = 2, \dots, n-2, n-1)$$

$j_0(x) \simeq 0$ または $j_1(x) \simeq 0$ のとき精度が低下するので次の処理を行う.

$$K = [1.27324x] \pmod{4}$$

として

K が 1 または 2 のとき

$$B = \frac{3B_1}{x} - B_2, \quad j = j_0(x)$$

K が 0 または 3 のとき

$$B = B_1, \quad j = j_1(x)$$

得られた T_n と B と j の値より

$$j_n(x) = \frac{T_n j}{B}$$

とする.

B. $x > n$ ならば

$j_1(x), j_0(x)$ を初期値とし, 漸化式

$$j_{k+1}(x) = \frac{2k+1}{x} j_k(x) - j_{k-1}(x) \quad (k = 1, 2, \dots, n-1)$$

を用いて $j_n(x)$ を求める.

(2) 第 2 種整数次球ベッセル関数 $y_n(x)$ ($x > 0.0$)

① $n < 0$ のとき

$y_n(x) = (-1)^{n+1} j_{-n-1}(x)$ より求める.

② $n \geq 0$ のとき

(a) $x \leq 0.41$ ならば

$$y_n(x) = -\frac{(2n-1)!!}{x^{n+1}} \left\{ 1 + \sum_{k=1}^9 \frac{\left(-\frac{x^2}{2}\right)^k}{k!(1-2n)(3-2n)\cdots(2k-1-2n)} \right\}$$

のべき級数展開で求める.

(b) $x > 0.41$ ならば

i. $n = 0, 1$ の場合

$$y_0(x) = -\frac{\cos(x)}{x}, \quad y_1(x) = -\frac{\cos(x) + x \sin(x)}{x^2}$$

により計算する.

ii. $n \geq 2$ の場合

$y_1(x), y_0(x)$ を初期値とし, 漸化式

$$y_{k+1}(x) = \frac{2k+1}{x} y_k(x) - y_{k-1}(x) \quad (k = 1, 2, \dots, n-1)$$

を用いて $y_n(x)$ を求める.

(3) 第 1 種整数次変形球ベッセル関数 $i_n(x)$ ($n \geq 0, x \geq 0.0$)

① $x^2 < 0.1n + 0.47$ で $n \leq 30$ のとき

$$i_n(x) = x^n \sum_{k=0}^9 \frac{\left(\frac{x^2}{2}\right)^k}{k!(2n+2k+1)!!}$$

のべき級数展開で求める.

② $x^2 \geq 0.1n + 0.47$ または $n > 30$ のとき

(a) $n = 0, 1$ の場合

$$i_0(x) = \frac{\sinh(x)}{x}$$

$$i_1(x) = \frac{x \cosh(x) - \sinh(x)}{x^2} = \frac{e^x(x-1) + e^{-x}(x+1)}{2x^2}$$

により計算する.

(b) $n \geq 2$ の場合

i. $x \leq n$ ならば

$T_{n+k+1} = 0.0$ とし, 次の漸化式を用いて連分数近似におけるパラメータ T_i ($i = n, n+1, \dots, n+k-1, n+k$) を計算する.

$$T_i = \frac{x^2}{2i + T_{i+1} + 1} \quad (i = n+1, \dots, n+k-1, n+k)$$

$$T_n = \frac{x}{2n + T_{n+1} + 1}$$

(T_n は $\frac{i_n(x)}{i_{n-1}(x)}$ の値)

k の値は表 2-6 に示すようにとる.

表 2-6 k の値

単精度演算	倍精度演算
$k = \lfloor \frac{(4.6+0.5\sqrt{n})x}{n} + 2.0 \rfloor$	$k = \lfloor \frac{(6.0+1.2\sqrt{n})x}{n} + 6.0 \rfloor$

$B_n = T_n, B_{n-1} = 1.0$ とし, 次の漸化式を用いて B_k ($k = 1, 2, \dots, n-2$) を計算する.

$$B_{k-1} = \frac{2k+1}{x} B_k + B_{k+1} \quad (k = 2, \dots, n-2, n-1)$$

(B_1 は $\frac{i_1(x)}{i_{n-1}(x)}$ の値)

得られた T_n と B_1 と $i_1(x)$ の値より

$$i_n(x) = \frac{T_n i_1(x)}{B_1}$$

とする.

ii. $x > n$ ならば

a を正の最小値 (浮動小数点データの内部表現であらわすことのできる最も 0.0 に近い正の値), M を n より十分大きな値とする.

$G_{M+1}(x) = 0.0, G_M(x) = a$ を初期値とし,

$$G_{k-1}(x) = \frac{2k+1}{x} G_k(x) + G_{k+1}(x) \quad (k = M, M-1, \dots, 1)$$

として

$$i_n(x) = \frac{G_n(x)e^x}{2 \sum_{m=0}^M (m+0.5)G_m(x)}$$

とする.

(4) 第 2 種整数次変形球ベッセル関数 $k_n(x)$ ($x > 0.0$)

① $n < 0$ のとき

$k_n(x) = k_{-n}(x)$ として $k_{-n}(x)$ について以下に述べる方法を適用する.

② $n \geq 0$ のとき

(a) $n = 0, 1$ の場合

$$k_0(x) = \frac{\pi e^{-x}}{2x}, \quad k_1(x) = \frac{\pi e^{-x}}{2x} \left(1 + \frac{1}{x}\right)$$

により計算する.

(b) $n \geq 2$ の場合

$k_1(x), k_0(x)$ を初期値とし, 漸化式

$$k_{i+1}(x) = \frac{2i+1}{x}k_i(x) + k_{i-1}(x) \quad (i = 1, 2, \dots, n-1)$$

を用いて $k_n(x)$ を求める.

2.1.2.4 ベッセル関数に関連した関数

(1) 第 1 種, 第 2 種整数次ハンケル関数 $H_n^{(1)}(z), H_n^{(2)}(z)$

$$H_n^{(1)}(z) = J_n(z) + iY_n(z)$$

$$H_n^{(2)}(z) = J_n(z) - iY_n(z)$$

より求める.

(2) ケルビン関数 $\text{ber}_n(x), \text{bei}_n(x)$

① $n < 0$ または, $x < 0.0$ のとき

まず, $\text{ber}_{|n|}(|x|), \text{bei}_{|n|}(|x|)$ を② $n \geq 0$ かつ $x \geq 0.0$ のときで述べる方法によって計算し, 次に以下の関係を用いて $\text{ber}_n(x), \text{bei}_n(x)$ を求める.

(a) n か x のどちらかが負のとき

$$\text{ber}_n(x) = (-1)^n \text{ber}_{|n|}(|x|), \quad \text{bei}_n(x) = (-1)^n \text{bei}_{|n|}(|x|)$$

(b) n も x も負のとき

$$\text{ber}_n(x) = \text{ber}_{|n|}(|x|), \quad \text{bei}_n(x) = \text{bei}_{|n|}(|x|)$$

② $n \geq 0$ かつ $x \geq 0.0$ のとき

(a) $x \leq \begin{cases} \text{ber}_n(x) : 5.65 + 0.25n \\ \text{bei}_n(x) : 6.92 + 0.25n \end{cases}$ とき, 次の近似式で求める.

$$\text{ber}_n(x) \simeq \sum_{k=0}^m \frac{\cos\{\frac{1}{4}(3n+2k)\pi\}}{k!(n+k)!} \left(\frac{x}{2}\right)^{n+2k}$$

$$\text{bei}_n(x) \simeq \sum_{k=0}^m \frac{\sin\{\frac{1}{4}(3n+2k)\pi\}}{k!(n+k)!} \left(\frac{x}{2}\right)^{n+2k}$$

ただし, $n = 0$ のときは $\text{ber}(x), \text{bei}(x)$ とともに上の式を最良近似式化して求める. 最良近似式化については 2.1.2.22 に記述されている.

(b) $x \geq \begin{cases} \text{倍精度} : 20.0 \\ \text{単精度} : 10.0 \end{cases}$ かつ $x \geq 0.5n^2$ のとき

$$\text{ber}_n(x) \simeq \frac{e^{\frac{x}{\sqrt{2}}}}{\sqrt{2\pi x}} \{P \cos(\Psi) + Q \sin(\Psi)\}$$

$$\text{bei}_n(x) \simeq \frac{e^{\frac{x}{\sqrt{2}}}}{\sqrt{2\pi x}} \{P \sin(\Psi) - Q \cos(\Psi)\}$$

の漸近展開式で求める.

ここで

$$P = 1 + \sum_{k=1}^m (-1)^k \cos\left(\frac{k\pi}{4}\right) \frac{(4n^2-1^2)(4n^2-3^2)\cdots(4n^2-(2k-1)^2)}{k!(8x)^k}$$

$$Q = \sum_{k=1}^m (-1)^k \sin\left(\frac{k\pi}{4}\right) \frac{(4n^2-1^2)(4n^2-3^2)\cdots(4n^2-(2k-1)^2)}{k!(8x)^k}$$

$$\Psi = \frac{x}{\sqrt{2}} + \left(\frac{n}{2} - \frac{1}{8}\right)\pi$$

ただし, $n = 0$ のときは, P, Q ともに最良近似式化して求める. 最良近似式化については 2.1.2.22 に記述されている. (m は末項がそれ以前の計算値に影響しなくなるころの値である.)

(c) (a), (b) 以外は, 複素変数第 1 種整数次ベッセル関数 $J_n(z)$ より

$$\text{ber}_n(x) = \Re \left(J_n \left(-\frac{x}{\sqrt{2}}, \frac{x}{\sqrt{2}} \right) \right)$$

$$\text{bei}_n(x) = \Im \left(J_n \left(-\frac{x}{\sqrt{2}}, \frac{x}{\sqrt{2}} \right) \right)$$

とする.

(3) ケルビン関数 $\text{ker}_n(x), \text{kei}_n(x)$ ($x > 0.0$)

① $n < 0$ のとき

まず, $\text{ker}_{|n|}(x), \text{kei}_{|n|}(x)$ を③ $n > 0$ のときで述べる方法によって計算し, 次に以下の関係を用いて $\text{ker}_n(x), \text{kei}_n(x)$ を求める.

$$\text{ker}_n(x) = (-1)^n \text{ker}_{|n|}(x), \quad \text{kei}_n(x) = (-1)^n \text{kei}_{|n|}(x)$$

② $n = 0$ で $x \leq \begin{cases} \text{ker}(x) : 5.77 \\ \text{kei}(x) : 6.56 \end{cases}$ のとき

$$\text{ker}(x) = -\log\left(\frac{x}{2}\right) \text{ber}(x) + \frac{\pi}{4} \text{bei}(x) + \sum_{n=0}^{\infty} \left\{ \frac{(-1)^n}{((2n)!)^2} \left(\sum_{s=1}^{2n} \frac{1}{s} - \gamma \right) \left(\frac{x}{2}\right)^{4n} \right\}$$

$$\text{kei}(x) = -\log\left(\frac{x}{2}\right) \text{bei}(x) - \frac{\pi}{4} \text{ber}(x) + \sum_{n=0}^{\infty} \left\{ \frac{(-1)^n}{((2n+1)!)^2} \left(\sum_{s=1}^{2n} \frac{1}{s} - \gamma \right) \left(\frac{x}{2}\right)^{4n+2} \right\}$$

を最良近似式化して求める (γ : オイラーの定数: $0.57721566\dots$). 最良近似式化については 2.1.2.22 に記述されている.

③ $n > 0$ のとき

(a) $x \geq \begin{cases} \text{倍精度} : 20.0 \\ \text{単精度} : 10.0 \end{cases}$ かつ $x \geq 0.5n^2$ のとき

$$\text{ker}_n(x) \simeq \sqrt{\frac{\pi}{2x}} e^{-x\sqrt{2}} \{P \cos(\Psi) - Q \sin(\Psi)\}$$

$$\text{kei}_n(x) \simeq \sqrt{\frac{\pi}{2x}} e^{-x\sqrt{2}} \{-P \sin(\Psi) - Q \cos(\Psi)\}$$

の漸近展開式で求める.

ここで

$$P = 1 + \sum_{k=1}^m \cos\left(\frac{k\pi}{4}\right) \frac{(4n^2 - 1^2)(4n^2 - 3^2)\cdots(4n^2 - (2k-1)^2)}{k!(8x)^k}$$

$$Q = \sum_{k=1}^m \sin\left(\frac{k\pi}{4}\right) \frac{(4n^2 - 1^2)(4n^2 - 3^2)\cdots(4n^2 - (2k-1)^2)}{k!(8x)^k}$$

$$\Psi = \frac{x}{\sqrt{2}} + \left(\frac{n}{2} - \frac{1}{8}\right)\pi$$

ただし, $n = 0$ のときは, P, Q ともに最良近似式化して求める. 最良近似式化については 2.1.2.22 に記述されている. (m は末項がそれ以前の計算値に影響しなくなるころの値である.)

(b) (a) 以外は, 複素変数第 2 種整数次変形ベッセル関数 $K_n(z)$ より, $K_n\left(\frac{x}{\sqrt{2}}, \frac{x}{\sqrt{2}}\right)$ の実部を A , 虚部を B とし, $\frac{n}{4}$ の余りが

i. 0 ならば

$$\text{ker}_n(x) = A, \quad \text{kei}_n(x) = B$$

ii. 1 ならば

$$\ker_n(x) = B, \text{ kei}_n(x) = -A$$

iii. 2 ならば

$$\ker_n(x) = -A, \text{ kei}_n(x) = -B$$

iv. 3 ならば

$$\ker_n(x) = -B, \text{ kei}_n(x) = A$$

とする.

(4) ストループ関数 $\mathbf{H}_0(x), \mathbf{H}_1(x), \mathbf{H}_0(x) - Y_0(x), \mathbf{H}_1(x) - Y_1(x)$

① $x < 0.0$ のとき

$\mathbf{H}_0(x) = -\mathbf{H}_0(-x), \mathbf{H}_1(x) = \mathbf{H}_1(-x)$ として $\mathbf{H}_0(-x), \mathbf{H}_1(-x)$ について以下に述べる方法を適用する.
ただし, ベッセル関数との差 $\mathbf{H}_0(x) - Y_0(x), \mathbf{H}_1(x) - Y_1(x)$ はエラーとする.

② $0.0 \leq x \leq 8.0$ のとき次の式を最良近似式化して求める.

$$\begin{aligned} \mathbf{H}_0(x) &= \frac{2}{\pi} \sum_{k=0}^{\infty} \frac{(-1)^k x^{2k+1}}{((2k+1)!!)^2} \\ \mathbf{H}_1(x) &= \frac{2}{\pi} \sum_{k=1}^{\infty} \frac{(-1)^k (2k+1)x^{2k}}{((2k+1)!!)^2} \end{aligned}$$

最良近似式化については 2.1.2.22 に記述されている.

$\mathbf{H}_0(x) - Y_0(x), \mathbf{H}_1(x) - Y_1(x)$ は, これにより得られた $\mathbf{H}_0(x), \mathbf{H}_1(x)$ より第 2 種ベッセル関数 $Y_0(x), Y_1(x)$ を減算して求める.

③ $x > 8.0$ のとき次の近似式で求める.

$$\begin{aligned} \mathbf{H}_0(x) - Y_0(x) &= \frac{1}{x} \sum_{k=0}^n a_k^{(1)} \left(\frac{1}{x}\right)^{2n} & n &= \begin{cases} \text{倍精度: 28} \\ \text{単精度: 6} \end{cases} \\ \mathbf{H}_1(x) - Y_1(x) &= \sum_{k=0}^n a_k^{(2)} \left(\frac{1}{x}\right)^{2n} & n &= \begin{cases} \text{倍精度: 25} \\ \text{単精度: 5} \end{cases} \end{aligned}$$

係数 $a_n^{(1)}, a_n^{(2)}$ は文献 (21) を参考に近似式をテレスコーピング計算して求めた.

$\mathbf{H}_0(x), \mathbf{H}_1(x)$ は, これより得られたベッセル関数との差の値に $Y_0(x), Y_1(x)$ を加算して求める.

(5) エアリ関数とその導関数 $\text{Ai}(x), \text{Bi}(x), \text{Ai}'(x), \text{Bi}'(x)$

$\zeta = \frac{2}{3}|x|^{\frac{3}{2}}$ とする.

① $x < \begin{cases} -3.8315472 & (\text{Ai}(x), \text{Bi}(x) \text{ のとき}) \\ -5.2414828 & (\text{Ai}'(x), \text{Bi}'(x) \text{ のとき}) \end{cases}$:

$$\text{Ai}(x) = \frac{\sqrt{-x}}{3} \left\{ \frac{4}{3\zeta} J_{\frac{2}{3}}(\zeta) - J_{\frac{5}{3}}(\zeta) + J_{\frac{1}{3}}(\zeta) \right\}$$

$$\text{Bi}(x) = \sqrt{\frac{-x}{3}} \left\{ \frac{4}{3\zeta} J_{\frac{2}{3}}(\zeta) - J_{\frac{5}{3}}(\zeta) - J_{\frac{1}{3}}(\zeta) \right\}$$

$$\text{Ai}'(x) = \frac{x}{3} \left\{ \frac{2}{3\zeta} J_{\frac{1}{3}}(\zeta) - J_{\frac{4}{3}}(\zeta) - J_{\frac{2}{3}}(\zeta) \right\}$$

$$\text{Bi}'(x) = \frac{-x}{\sqrt{3}} \left\{ \frac{2}{3\zeta} J_{\frac{1}{3}}(\zeta) - J_{\frac{4}{3}}(\zeta) + J_{\frac{2}{3}}(\zeta) \right\}$$

② $\begin{cases} -3.8315472 & (\text{Ai}(x), \text{Bi}(x) \text{ のとき}) \\ -5.2414828 & (\text{Ai}'(x), \text{Bi}'(x) \text{ のとき}) \end{cases} \leq x < \begin{cases} 3.8315472 & (\text{Ai}(x), \text{Bi}(x) \text{ のとき}) \\ 5.2414828 & (\text{Ai}'(x), \text{Bi}'(x) \text{ のとき}) \end{cases}$

$$C_1 = \frac{3^{-\frac{2}{3}}}{\Gamma(\frac{2}{3})}$$

$$C_2 = \frac{3^{-\frac{1}{3}}}{\Gamma(\frac{1}{3})}$$

$$f(x) = 1 + \frac{1}{3!}x^3 + \frac{1 \cdot 4}{6!}x^6 + \frac{1 \cdot 4 \cdot 7}{9!}x^9 + \dots$$

$$g(x) = x + \frac{2}{4!}x^4 + \frac{2 \cdot 5}{7!}x^7 + \frac{2 \cdot 5 \cdot 8}{10!}x^{10} + \dots$$

$x < 0$ と $x \geq 0$ の区間に分割して,

$$\text{Ai}(x) = C_1 f(x) - C_2 g(x)$$

$$\text{Bi}(x) = \sqrt{3}(C_1 f(x) + C_2 g(x))$$

を最良近似式化して求める.

$\text{Ai}'(x)$ は, $\text{Ai}(x)$ の近似式を微分した式を作り, これを最良近似式化して求める. $\text{Bi}'(x)$ は, $\text{Bi}(x)$ の近似式を微分した式を作り, これを最良近似式化して求める.

最良近似式化については, 2.1.2.22 に記述されている.

$$\textcircled{3} \quad x \geq \begin{cases} 3.8315472 & (\text{Ai}(x), \text{Bi}(x) \text{ のとき}) \\ 5.2414828 & (\text{Ai}'(x), \text{Bi}'(x) \text{ のとき}) \end{cases}$$

$$\text{Ai}(x) = e^{-\zeta} x^{-\frac{1}{4}} \sum_{k=0}^{m1} a_k^{(1)} \left(\frac{1}{\zeta}\right)^k$$

$$\text{Bi}(x) = e^{\zeta} x^{-\frac{1}{4}} \sum_{k=0}^{m2} a_k^{(2)} \left(\frac{1}{\zeta}\right)^k + \sqrt{\frac{x}{3}} I_{\frac{5}{3}}(\zeta)$$

$$\text{Ai}'(x) = e^{-\zeta} x^{\frac{1}{4}} \sum_{k=0}^{m3} a_k^{(3)} \left(\frac{1}{\zeta}\right)^k$$

$$\text{Bi}'(x) = e^{\zeta} x^{\frac{1}{4}} \sum_{k=0}^{m4} a_k^{(4)} \left(\frac{1}{\zeta}\right)^k + \frac{x}{\sqrt{3}} I_{\frac{4}{3}}(\zeta)$$

係数 $a_k^{(1)} \sim a_k^{(4)}$ は文献 (21) を参考に, 新しく求めなおした.

2.1.2.5 ガンマ関数

(1) 実変数ガンマ関数 $\Gamma(x)$ および実変数対数ガンマ関数 $\log_e(\Gamma(x))$

① $x \geq 11.0$ のとき以下の対数ガンマ関数の漸近展開より求める.

$$\log_e(\Gamma(x)) \sim \left(x - \frac{1}{2}\right) \log_e x - x + \frac{1}{2} \log_e(2\pi) + \sum_{n=1}^{\infty} \frac{B_{2n}}{(2n-1)(2n)x^{2n-1}}$$

$$\Gamma(x) = \exp(\log_e(\Gamma(x)))$$

② $0 < x < 11.0$ のとき, 最良有理式近似を用いる. z を x の小数部分として

$$\Gamma(z) = \left(\sum_{k=0}^6 a_k z^k\right) / \left(\sum_{k=0}^7 b_k z^k\right)$$

これを, 関数等式により, $\Gamma(x)$ に移す.

③ $x < 0.0$ のとき

$$\Gamma(x) = \frac{\pi}{-x \sin(\pi x) \Gamma(-x)}$$

$$\log_e(\Gamma(x)) = \log_e \pi - \log_e((-x) \sin(\pi x)) - \log_e(\Gamma(-x))$$

を用いる.

(2) 複素変数ガンマ関数 $\Gamma(z)$ および複素変数対数ガンマ関数 $\log(\Gamma(z))$

- ① $\Im(z) = 0.0$ のときは実変数対数ガンマ関数を利用する.
- ② $\Re(z) = 0.0$ で $|\Im(z)| > 12.0$ のとき以下の漸近展開より求める.

$$\Re(\log_e(\Gamma(z))) \sim \frac{1}{2}(\log_e(2\pi) - \pi\Im(z) - \log_e(\Im(z)))$$

$$\Im(\log_e(\Gamma(z))) \sim \Im(z) \log_e(\Im(z)) - \Im(z) - \frac{1}{4}\pi - \sum_{n=1}^{\infty} \frac{(-1)^{n-1} B_{2n}}{(2n-1)(2n)(\Im(z))^{2n-1}}$$

ここで, B_n はベルヌーイ数である.

- ③ $\Im(z) < 0.0$ のとき

$$\log_e(\Gamma(z)) = \log_e \pi - \log_e(-z) \sin(\pi z) - \log_e(\Gamma(-z))$$

であるので, 以下の方法を $\log_e(\Gamma(-z))$ に適用して計算する.

- ④ $\Im(z) > 0.0$ のとき

- (a) $|\Re(z)| < 11.0$ のとき

$$\log_e(\Gamma(z)) = \log_e(\Gamma(n+z)) - \log_e((n-1+z)(n-2+z)\cdots(z))$$

であるので, 以下の方法を $\log_e(\Gamma(n+z))$ に適用して計算する. ただし, $n = \lceil 12.0 - \Re(z) \rceil$ である.

- (b) $|\Re(z)| \geq 11.0$ のとき以下の漸近展開より求める.

$$\log_e(\Gamma(z)) \sim (z - \frac{1}{2}) \log_e z - z + \frac{1}{2} \log_e(2\pi) + \sum_{n=1}^{\infty} \frac{B_{2n}}{(2n-1)(2n)z^{2n-1}}$$

- ⑤ 複素変数ガンマ関数 $\Gamma(z)$ は, $\exp(\log_e(\Gamma(z)))$ として計算する.

(3) 第1種不完全ガンマ関数 $\gamma(\nu, x)$, ($\nu \geq 0.0, x \geq 0.0$)

- ① $x = 0.0$ のとき $\gamma(\nu, x) = 0.0$ とする.
- ② $x \leq \nu - 0.5$ または $x \leq 3.5$ のとき
 $\nu \leq \frac{e^{3.5}}{\text{最大値}}$ のとき

$$\gamma(\nu, x) = \frac{1}{\nu}$$

その他のとき

$$\gamma(\nu, x) = \frac{e^{\nu \log(x) - x} P}{\nu}$$

ここで,

$$P = 1 + \sum_{k=1}^m \frac{x^k}{(\nu+1)(\nu+2)\cdots(\nu+k)}$$

(m は末項がそれ以前の計算値に影響しなくなるまでの項である)

- ③ ①, ②以外 のとき,

$$\gamma(\nu, x) = \Gamma(\nu) - \Gamma(\nu, x)$$

とする.

ここで, $\Gamma(\nu, x)$ は, 第2種不完全ガンマ関数である.

(4) 第2種不完全ガンマ関数 $\Gamma(\nu, x)$ ($\nu \geq 0.0, x \geq 0.0$)

- ① $\nu \leq \frac{1.0}{\text{最大値}}$ のとき, 指数積分のサブルーチンを利用して, $-\text{Ei}(-x)$ を計算する.

② $x = 0.0$ のとき, $\Gamma(\nu, x) = \Gamma(\nu)$ とする.

③ ν が整数で $x > 0.015\nu^2$ のとき,

$$\Gamma(\nu, x) = e^{(\nu-1)\log(x)-x} \left\{ 1 + \sum_{k=1}^m \frac{(\nu-1)(\nu-2)\cdots(\nu-k)}{x^k} \right\}$$

(m は末項がそれ以前の計算値に影響しなくなるまで $m \leq \nu + 2$ までの項である)

④ $x \leq \nu - 0.5$ のとき

$\Gamma(\nu, x) = \Gamma(\nu) - \gamma(\nu, x)$ とする.

⑤ $x \geq \begin{cases} \text{倍精度 : 49.0} \\ \text{単精度 : 23.0} \end{cases}$ のとき,

ν が整数で $x \geq 0.015\nu^2$ のときと同じ処理を行う.

⑥ $\nu < 1.0$ のとき

(a) $x < 0.36\nu + 0.85$ のとき,

$$\Gamma(\nu, x) = (\nu - 1) \frac{\sum_{k=0}^M p_k^{(2)} \nu^k}{\sum_{k=0}^M q_k^{(2)} \nu^k} - \frac{e^{\nu \log(x)} - 1}{\nu} - x^\nu \sum_{k=1}^{M'} \frac{(-1)^k x^k}{k!(k+\nu)} \quad \left(M' : \begin{cases} \text{倍精度 : 21} \\ \text{単精度 : 12} \end{cases} \right)$$

(b) $x \leq 1.5\nu + 2.1$ のとき,

$$\Gamma(\nu, x) = \Gamma(1 + \nu) e^{-x} \sum_{k=0}^{N'} (x^k \tilde{A}_k(\nu) + x^k \frac{\phi(\nu, x)}{\Gamma(k+1+\nu)}) \quad \left(N' : \begin{cases} \text{倍精度 : 23} \\ \text{単精度 : 14} \end{cases} \right)$$

ここで

$$\tilde{A}_0(\nu) = (1 + \nu)(\tilde{A}_1(\nu) - 1)$$

$$\tilde{A}_1(\nu) \simeq \sum_{k=0}^M p_k^{(1)} \nu^k$$

として

$$\tilde{A}_k(\nu) = \frac{1}{k+\nu} \left\{ \tilde{A}_{k-1}(\nu) + \frac{1}{k!} \right\}$$

$$\phi(\nu, x) = -\frac{e^{\nu \log(x)} - 1}{\nu}$$

(c) 上記以外のとき,

$$\Gamma(\nu, x) = e^{\nu \log(x)-x} \cdot \frac{1}{|x|} + \frac{\infty}{\Phi} \frac{n-\nu}{|1|} + \frac{n}{|x|}$$

ここで, 連分数の項数は, $\left\{ \begin{array}{l} \text{倍精度 : } \lfloor \frac{120}{x} + 5 \rfloor \\ \text{単精度 : } \lfloor \frac{25}{x-0.25} + 2 \rfloor \end{array} \right\}$ とする.

⑦ ① から ⑥ 以外のとき,

(a) $x \leq 5.6$ のとき, $\alpha = \nu - \lfloor \nu \rfloor$, $\mu = \alpha + 1.0$ として,

$$\Gamma(\mu, x) = \Gamma(\mu) e^{-x} \left[1 + \alpha \sum_{k=0}^{M''} (x^{k+1} \tilde{C}_k(\alpha) + x^{k+1} \frac{\phi(\alpha, x)}{\Gamma(k+1+\mu)}) \right] \quad \left(M'' : \begin{cases} \text{倍精度 : 20} \\ \text{単精度 : 12} \end{cases} \right)$$

$$\tilde{C}_0(\alpha) = \tilde{A}_1(\alpha)$$

として,

$$\tilde{C}_k(\alpha) = \frac{1}{k+\nu} \left\{ \tilde{C}_{k-1}(\alpha) + \frac{1}{(k+1)!} \right\}$$

$$\phi(\alpha, x) = -\frac{e^{\alpha \log(x)} - 1}{\alpha}$$

- $\nu > 2.0$ のとき,

$m = \nu - \mu$ として,

$$\Gamma(\nu, x) = e^{(\nu-1)\log(x)-x} \left\{ 1 + \sum_{k=1}^{m-1} \frac{(\nu-1)(\nu-2)\cdots(\nu-k)}{x^k} \right\} + (\nu-1)(\nu-2)\cdots(\nu-k)\Gamma(\mu, x)$$

- $\nu \leq 2.0$ のとき,

$$\Gamma(\nu, x) = \Gamma(\mu, x)$$

(b) (a) 以外 のとき,

$$\Gamma(\nu, x) = e^{(\nu-1)\log(x)-x} \left\{ 1 + \sum_{k=1}^{m-1} \frac{(\nu-1)(\nu-2)\cdots(\nu-k)}{x^k} \right\} + (\nu-1)(\nu-2)\cdots(\nu-m)e^{(\nu-m)\log(x)-x} \cdot \left[\frac{1}{x} + \frac{\infty}{n-1} \left(\frac{n - (\nu-m)}{1} + \frac{n}{x} \right) \right]$$

(m は ν の整数部)

係数 $p_k^{(1)}, p_k^{(2)}, q_k^{(2)}$ は文献 (22) に記述されている。

2.1.2.6 ガンマ関数に関連した関数

(1) デイガンマ関数 $\Psi(x)$ ($x < 0.0$ のとき $x \neq$ 整数)

① $x < -2.0$ のとき

$$\Psi(x) = \log(1-x) - \frac{1}{2(1-x)} + \frac{\sum_{k=0}^m a_k^{(2)}(1-x)^{-2k}}{\sum_{k=0}^m b_k^{(2)}(1-x)^{-2k}} - \pi \cot(\pi x)$$

を計算する。

② $-2.0 < x < 0.5$ のとき

$$\Psi(x) = (1-c-x) \frac{\sum_{k=0}^m a_k^{(1)}(1-x)^k}{\sum_{k=0}^m b_k^{(1)}(1-x)^k} - \pi \cot(\pi x)$$

を計算する。

③ $0.5 \leq x \leq 3.0$ のとき

$$\Psi(x) = (x-c) \frac{\sum_{k=0}^m a_k^{(1)}x^k}{\sum_{k=0}^m b_k^{(1)}x^k}$$

を計算する。

④ $x > 3.0$ のとき

$$\Psi(x) = \log(x) - \frac{1}{2x} + \frac{\sum_{k=0}^m a_k^{(2)}x^{-2k}}{\sum_{k=0}^m b_k^{(2)}x^{-2k}}$$

を計算する.

なお, $c = 1.46163214496836234126$ とし, 係数 $a_k^{(1)}, b_k^{(1)}, a_k^{(2)}, b_k^{(2)}$ は文献 (13) に記述されている.

(2) ベータ関数 $B(p, q)$ ($p > 0.0, q > 0.0$)

ベータ関数は, つぎの式により定義される.

$$B(p, q) = \int_0^1 x^{p-1}(1-x)^{q-1} dx = B(q, p) \quad (p, q > 0)$$

p と q の値により, 以下のように計算する.

① $p < 12.0$ かつ $q < 12.0$ のときガンマ関数を使い, つぎの式で計算する.

$$B(p, q) = \frac{\Gamma(p)\Gamma(q)}{\Gamma(p+q)}$$

② $p < 12.0$ かつ $q \geq 12.0$ のとき, ガンマ関数とスターリングの公式から計算する.

$$B(p, q) = \Gamma(p)e^{(q-\frac{1}{2})\log(q)+\Phi(q)-(p+q-\frac{1}{2})\log(p+q)+p-\Phi(p+q)}$$

ここで $\Phi(x)$ は,

$$\Phi(x) = \sum_{n=1}^{\infty} \frac{(-1)^{n-1} B_n}{(2n)(2n-1)x^{2n-1}}$$

を最良近似式化して求める (B_n はベルヌーイ数).

最良近似式化については 2.1.2.22 に記述されている.

③ $p \geq 12.0$ かつ $q < 12.0$ のときは $B(p, q) = B(q, p)$ より $p < 12.0$ かつ $q \geq 12.0$ のときと同じ.

④ $p \geq 12.0$ かつ $q \geq 12.0$ のときはスターリングの公式から計算する.

$$B(p, q) = e^{(p-\frac{1}{2})\log(p)+(q-\frac{1}{2})\log(q)-(p+q-\frac{1}{2})\log(p+q)+\log(\sqrt{2\pi})+\Phi(p)+\Phi(q)-\Phi(p+q)}$$

2.1.2.7 楕円関数と楕円積分

(1) 第 1 種, 第 2 種完全楕円積分 $K(m), E(m)$ ($0.0 \leq m \leq 1.0$)

① $0.0 \leq m \leq 0.25$ のとき

$$K(m) = \frac{\pi}{2} \sum_{k=0}^{\infty} \left(\frac{(2k-1)!!}{(2k)!!} \right)^2 m^k$$

$$E(m) = \frac{\pi}{2} \left\{ 1 - \sum_{k=1}^{\infty} \left(\frac{(2k-1)!!}{(2k)!!} \right)^2 \frac{m^k}{2k-1} \right\}$$

を最良近似式化して求める.

最良近似式化については 2.1.2.22 に記述されている.

② $0.25 < m < 1.0$ のとき

$$K(m) = \sum_{k=0}^l \left\{ a_k^{(1)} (1-m)^k \right\} - \log(1-m) \sum_{k=0}^l \left\{ b_k^{(1)} (1-m)^k \right\}$$

$$E(m) = \sum_{k=0}^l \left\{ a_k^{(2)} (1-m)^k \right\} - \log(1-m) \sum_{k=0}^l \left\{ b_k^{(2)} (1-m)^k \right\}$$

を計算する.

③ $m = 1.0$ のとき, $K(m) = +\infty, E(m) = 1.0$ とする.

係数 $a_k^{(1)}, b_k^{(1)}, a_k^{(2)}, b_k^{(2)}$ は文献 (2) に記述されている.

(2) 第1種, 第2種不完全楕円積分 $F(x, m), E(x, m)$ ($0.0 \leq x \leq 1.0, 0.0 \leq m \leq 1.0$)

① $x = 0.0$ のとき, $F(0.0, m) = E(0.0, m) = 0.0$

② $x = 1.0$ のとき, 完全楕円積分より

$$F(1.0, m) = K(m)$$

$$E(1.0, m) = E(m)$$

とする.

③ ①, ②以外

(a) $m = 0.0$ のとき

$$F(x, 0.0) = \sin^{-1}(x)$$

$$E(x, 0.0) = \sin^{-1}(x)$$

とする.

(b) $0.0 < m < 1.0$ のとき

$a_0 = 1.0, b_0 = \sqrt{1-m}, c_0 = \sqrt{m}$ を初期値として

$$a_{k+1} = \frac{a_k + b_k}{2}$$

$$b_{k+1} = \sqrt{a_k b_k}$$

$$c_{k+1} = \frac{a_k - b_k}{2} \quad (k = 0, 1, \dots)$$

を求める. 収束判定は,

$$c_N < a_k \cdot (\text{誤差判定のための単位}).$$

とする. $\phi_0 = \sin^{-1}(x)$ とし,

$$\tan(\phi_{k+1} - \phi_k) - \frac{b_k}{a_k} \tan(\phi_k) = 0 \quad (k = 0, \dots, N-1)$$

を $\phi_{k+1} (> \phi_k)$ についてニュートン法で解きながら, ϕ_1, \dots, ϕ_N を求める.

$$F(x, m) = \frac{\phi_N}{2^N a_N}$$

$$E(x, m) = \left(1 - \frac{1}{2} \sum_{k=0}^N 2^k c_k^2\right) F(x, m) + \sum_{k=1}^N c_k \sin(\phi_k)$$

を計算する. 他の方を用いて, 第1種不完全楕円積分 $F(x, m)$ を求めることもできる. ランデン変換 (母数 $m = k^2$ を, 以下の k_1^2 に換える変換) により,

$$k_1 = \frac{1 - \sqrt{1-m}}{1 + \sqrt{1-m}}, \quad m_1 = k_1^2, \quad m = k^2, \quad x_1 = \frac{2}{1 + \sqrt{1-mx^2}} \frac{x}{1 + k_1}$$

とすると, $F(x, m) = (1 + k_1)F(x_1, m_1)$ であるから, $X_0 = x, K_0 = k, (m = k^2, k > 0)$ として, $n = 0, 1, \dots$ について逐次,

$$K_{n+1} = \frac{K_n^2}{(1 + \sqrt{1 - K_n^2})^2}, \quad X_{n+1} = \frac{2}{1 + \sqrt{1 - K_n^2 X_n^2}} \frac{X_n}{1 + K_{n+1}}$$

とすると,

$$F(x, m) = (1 + K_1)(1 + K_2) \cdots (1 + K_{n+1})F(X_{n+1}, K_{n+1}^2)$$

であるが, $K_{n+1} \leq K_n^2$ などにより, K_{n+1} は非常に速く 0 に収束し, $F(X_{n+1}, K_{n+1}^2)$ は $\sin^{-1} X_{n+1}$ に等しいとみなすことができる.

ガウスの算術幾何平均の方法は, ランデン変換の方法と本質的には同じである.

(c) $m = 1.0$ のとき

$$F(x, m) = \text{artanh}(x)$$

$$E(x, m) = x$$

とする.

(3) 不完全変形楕円積分, ワイエルシュトラス型の不完全楕円積分

母数 $m(0 \leq m < 1)$ の第 1 種不完全楕円積分 $F(r, m)$ および第 2 種不完全楕円積分 $E(r, m)$ を, 各々

$$F(r, m) = \int_0^r \frac{dx}{\sqrt{(1-x^2)(1-mx^2)}},$$

$$E(r, m) = \int_0^r \sqrt{\frac{1-mx^2}{1-x^2}} dx$$

で定義する. ただし, $0 \leq r < 1$.

① 不完全変形楕円積分

$m \geq 0, a, b$ と $p \geq 0$ に対して, 不完全変形楕円積分

$$\int_0^p \frac{a+bt^2}{1+t^2} \frac{dt}{\sqrt{(1+t^2)(1+mt^2)}}$$

は以下の手順で求める.

ただし, $m = 0, 1$ である場合, すなわち不完全変形楕円積分が退化し初等関数で表現できる場合は対数関数または逆正接関数によって求める.

(a) m が 1 に近くない場合で, $m > 1$ ならば, 置換積分によって $m < 1$ の場合 (b) に帰着する.

(b) m が 1 に近くない場合で, $m < 1$ の場合, $p = r/\sqrt{1-r^2}$ とおいて

$$\int_0^p \frac{a+bt^2}{1+t^2} \frac{dt}{\sqrt{(1+t^2)(1+mt^2)}} = \frac{b-ma}{1-m} F(r, 1-m) + \frac{a-b}{1-m} E(r, 1-m)$$

(c) m が 1 に近いときは, $p = \tan \alpha, 0 \leq \alpha < \pi/2$ と置いて,

$$\int_0^p \frac{a+bt^2}{1+t^2} \frac{dt}{\sqrt{(1+t^2)(1+mt^2)}} = \int_0^\alpha \frac{a+(b-a)\sin^2 u}{\sqrt{1-(1-m)\sin^2 u}} du$$

を用いる. $I_n(\alpha) = \int_0^\alpha \sin^{2n} u du$ は漸化式

$$(2n+2)I_{n+1}(\alpha) = (2n+1)I_n(\alpha) - \sin^{2n+1} \alpha \cos \alpha, n \geq 0$$

によって求め, 被積分関数の展開にべき級数展開

$$\frac{1}{\sqrt{1-v}} = 1 + \sum_{n=1}^{\infty} \frac{I_n(\pi/2)}{\pi/2} v^n, |v| < 1$$

を利用する.

② ワイエルシュトラス型の不完全楕円積分

不完全楕円積分が退化し初等関数で表現できる場合は対数関数または逆正接関数によって表示する. それ以外の場合は, $z > y > x > 0$ の場合を取り扱えばよく

$$v_1 = \sqrt{\frac{(z-x)p}{1+zp}}, v_2 = \sqrt{\frac{z-x}{z}}, m = \frac{z-y}{z-x}$$

とにおいて

$$\frac{1}{2} \int_0^{1/p} \frac{dt}{\sqrt{(t+x)(t+y)(t+z)}} = \frac{1}{\sqrt{z-x}} (F(v_2, m) - F(v_1, m))$$

である. $x = 0$ のときは不完全変形楕円積分を用いれば良い.

(4) ヤコビの楕円関数 $\text{sn}(u, m), \text{cn}(u, m), \text{dn}(u, m)$ ($0.0 \leq m \leq 1.0$)

① $u = 0.0$ のとき

$$\text{sn}(0.0, m) = 0.0, \text{cn}(0.0, m) = \text{dn}(0.0, m) = 1.0$$

② $m = 1.0$ のとき

$$\begin{aligned} \operatorname{sn}(u, 1.0) &= \tanh(u) \\ \operatorname{cn}(u, 1.0) &= \operatorname{dn}(u, 1.0) = \operatorname{sech}(u) \end{aligned}$$

③ $m = 0.0$ のとき

$$\begin{aligned} \operatorname{sn}(u, 0.0) &= \sin(u) \\ \operatorname{cn}(u, 0.0) &= \cos(u) \\ \operatorname{dn}(u, 0.0) &= 1.0 \end{aligned}$$

④ ①, ②, ③以外

$a_0 = 1.0, b_0 = \sqrt{1-m}$ を初期値として

$$\begin{aligned} a_{k+1} &= \frac{a_k + b_k}{2} \\ b_{k+1} &= \sqrt{a_k b_k} \\ c_{k+1} &= \frac{a_k - b_k}{2} \quad (k = 0, 1, \dots) \end{aligned}$$

を求める。収束判定は、

$$c_N < a_{N-1} \cdot (\text{誤差判定のための単位}).$$

とする。

$$\begin{aligned} K(m) &= \frac{\pi}{2a_N} \\ y_N &= \frac{a_N}{\sin(a_N u)} \\ y_{N-1} &= y_N + \frac{a_N c_N}{y_N} \\ &\vdots \\ y_0 &= y_1 + \frac{a_1 c_1}{y_1} \end{aligned}$$

を求め

$$\begin{aligned} \operatorname{sn}(u, m) &= \frac{1}{y_0} \\ \operatorname{cn}(u, m) &= \begin{cases} \sqrt{1 - \left(\frac{1}{y_0}\right)^2} & (\text{mod}(\lfloor \frac{|u|}{K(m)} \rfloor, 4) = 0 \text{ or } 3 \text{ のとき}) \\ -\sqrt{1 - \left(\frac{1}{y_0}\right)^2} & (\text{mod}(\lfloor \frac{|u|}{K(m)} \rfloor, 4) = 1 \text{ or } 2 \text{ のとき}) \end{cases} \\ \operatorname{dn}(u, m) &= \sqrt{1 - m \left(\frac{1}{y_0}\right)^2} \end{aligned}$$

なお、 $\lfloor \frac{u}{K(m)} \rfloor$ と $\frac{u}{K(m)}$ の差が 0.03125 以下のときは、 $\operatorname{sn}(u, m) \simeq 1.0$ で $\sqrt{1 - \operatorname{sn}^2(u, m)}$ の計算で大きなケタ落ちが生じるため、次式により求める。

$$\begin{aligned} v &= \frac{u}{2K(m)} \\ \operatorname{sn}(u, m) &= \frac{\vartheta_1(v, q)}{\sqrt[4]{m} \vartheta_4(v, q)} \\ \operatorname{cn}(u, m) &= \frac{\vartheta_2(v, q) \sqrt[4]{\frac{1-m}{m}}}{\vartheta_4(v, q)} \\ \operatorname{dn}(u, m) &= \frac{\vartheta_3(v, q) \sqrt[4]{(1-m)}}{\vartheta_4(v, q)} \end{aligned}$$

(q は、次項ノーム q 参照)

(5) ノーム q ($0.0 \leq m \leq 1.0$)

$m > 0.5$ ならば、

$m = 1 - m$ とし、最終的に q と q' , $K(m)$ と $K(m')$, $E(m)$ と $E(m')$ の値を、それぞれ入れ替える。

① $m = 0.0$ のとき、

$$\begin{aligned} q &= 0.0 \\ q' &= 1.0 \\ K(m) &= \frac{\pi}{2} \\ K(m') &= (\text{最大値}) \\ E(m) &= \frac{\pi}{2} \\ E(m') &= 1.0 \end{aligned}$$

② ① 以外のとき、

$$\varepsilon = \frac{0.5m}{(2(1 + \sqrt[4]{1-m}(1 + \sqrt{1-m}) + \sqrt{1-m}) - m)}$$

とおくと、

$$q = c_4\varepsilon^{13} + c_3\varepsilon^9 + c_2\varepsilon^5 + c_1\varepsilon$$

(ただし、倍精度では $c_1 = 1, c_2 = 2, c_3 = 15, c_4 = 150$, 単精度では $c_1 = 1, c_2 = 2, c_3 = 0, c_4 = 0$)

$$q' = \exp\left(\frac{\pi^2}{\log(q)}\right)$$

δ を

$$\delta = (d_4q^9 + d_3q^4 + d_2q + d_1)^2$$

(ただし、倍精度では $d_1 = 1, d_2 = 2, d_3 = 1, d_4 = 1$ 単精度では $d_1 = 1, d_2 = 2, d_3 = 1, d_4 = 0$) とおくと、

$$\begin{aligned} K(m) &= \frac{\pi}{2}\delta \\ K(m') &= (-0.5)\log(q)\delta \\ E(m) &= \left(\frac{\pi}{6\delta}\right)(1 + (2-m)\delta^2 - (e_7q^{14} + e_6q^{12} + e_5q^{10} + e_4q^8 + e_3q^6 + e_2q^4 + e_1q^2)) \\ E(m') &= K(m')\left(1 - \frac{E(m)}{K(m)}\right) + \frac{1}{\delta} \end{aligned}$$

(ただし、倍精度では $e_1 = 24, e_2 = 72, e_3 = 96, e_4 = 168, e_5 = 144, e_6 = 288, e_7 = 192$ 単精度では $e_1 = 24, e_2 = 72, e_3 = 96, e_4 = 0, e_5 = 0, e_6 = 0, e_7 = 0$)

(6) 楕円テータ関数 $\vartheta_i(v, q)$ ($0 \leq i \leq 4, 0.0 \leq q \leq 1.0$)

$i = 0$ のときは $i = 4$ とし、 $i = 2, 3$ のときはそれぞれ $v = v + 0.5, v = v - 0.5$ としておく。

① $i = 1, 2$ のとき

$s = \text{SIGN}(v), v = |v|$ とおき、

$v = v - \text{INT}(v), \text{INT}(v)$ が奇数のとき $s = -s$ とする。

i) $q = 0.0$ または $v = 0.0$ ならば、

$$\vartheta(v, q) = 0.0$$

とする。

ii) $q \leq 0.125$ ならば,

$$sw = \sin(\pi \cdot v)$$

$$s3w = (-4 \cdot sw^2 + 3) \cdot sw$$

$$s5w = ((16 \cdot sw^2 - 20) \cdot sw^2 + 5) \cdot sw$$

とおくと,

・単精度

$$\vartheta(v, q) = 2 \cdot s \cdot \sqrt[4]{q} \cdot ((s5w \cdot q^4 - s3w) \cdot q^2 + sw)$$

・倍精度

$$s7w = (((-64 \cdot sw^2 + 112) \cdot sw^2 - 56) \cdot sw^2 + 7) \cdot sw$$

$$s9w = (((256 \cdot (1 - sw^2) - 448) \cdot (1 - sw^2) + 240) \cdot (1 - sw^2) - 40) \cdot (1 - sw^2) + 1) \cdot sw$$

とおくと,

$$\vartheta_i(v, q) = 2 \cdot s \cdot \sqrt[4]{q} \cdot (((s9w \cdot q^8 - s7w) \cdot q^6 + s5w) \cdot q^4 - s3w) \cdot q^2 + sw)$$

iii) $q > 0.125$ のとき

$v > 0.5$ ならば $v = 1.0 - v$ とする.

$$PLQ = \pi^2 \cdot (1/\log(q))$$

$$wQ = \exp(PLQ)$$

$$w = -\exp(2 \cdot v \cdot PLQ)$$

とおくと,

・単精度

$$\vartheta_i(v, q) = s \cdot \sqrt{\pi \cdot (-1/\log(q))} \cdot \exp((v^2 - v + 0.25) \cdot PLQ) \cdot (((wQ^6 \cdot w + wQ^2) \cdot w + 1) \cdot w + 1 + (wQ^4/w + 1) \cdot (wQ^2/w))$$

・倍精度

$$\vartheta_i(v, q) = s \cdot \sqrt{\pi \cdot (-1/\log(q))} \cdot \exp((v^2 - v + 0.25) \cdot PLQ) \cdot (((wQ^{12} \cdot w + wQ^6) \cdot w + wQ^2) \cdot w + 1) \cdot w + 1 + ((wQ^8/w + wQ^2) \cdot (wQ^2/w) + 1) \cdot (wQ^2/w))$$

② $i = 3, 4$ のとき,

$v = |v| - INT(|v|)$ とする.

i) $q = 0.0$ ならば,

$$\vartheta_i(v, q) = 1.0$$

とする.

ii) $q \leq 0.125$ ならば,

$$cw = \cos(2 \cdot \pi \cdot v)$$

$$c2w = 2 \cdot cw^2 - 1$$

$$c3w = (4 \cdot cw^2 - 3) \cdot cw$$

とおくと,

・単精度

$$\vartheta_i(v, q) = 2 \cdot ((-c3w \cdot q^5 + c2w) \cdot q^3 - cw) \cdot q + 1$$

・倍精度

$$c4w = (8 \cdot cw^2 - 8) \cdot cw^2 + 1$$

とおくと,

$$\vartheta_i(v, q) = 2 \cdot (((c4w \cdot q^7 - c3w) \cdot q^5 + c2w) \cdot q^3 - cw) \cdot q + 1$$

iii) $q > 0.125$ のとき, $v > 0.5$ ならば $v = 1.0 - v$ とする.

$$PLQ = \pi^2 \cdot (1/\log(q))$$

$$wQ = \exp(PLQ)$$

$$w = \exp(2 \cdot v \cdot PLQ)$$

とくと、

・単精度

$$\vartheta_i(v, q) = \sqrt{\pi \cdot (-1/\log(q))} \cdot \exp((v^2 - v + 0.25) \cdot PLQ) \cdot (((wQ^6 \cdot w + wQ^2) \cdot w + 1) \cdot w + 1 + (wQ^4/w + 1) \cdot (wQ^2/w))$$

・倍精度

$$\vartheta_i(v, q) = \sqrt{\pi \cdot (-1/\log(q))} \cdot \exp((v^2 - v + 0.25) \cdot PLQ) \cdot (((wQ^{12} \cdot w + wQ^6) \cdot w + wQ^2) \cdot w + 1) \cdot w + 1 + ((wQ^6/w + wQ^2) \cdot (wQ^2/w) + 1) \cdot (wQ^2/w)$$

(7) ヤコビのゼータ関数 $Z(u)$ ($0.0 \leq m \leq 1.0$)

m より $K(m), K(m')$, ノーム q , 補ノーム q' を求める.

$$v = u/(2 \cdot K(m))$$

$$s = \text{SIGN}(v), v = |v| - \text{INT}(|v|), w = 2\pi \cdot v \text{ として}$$

① $q \leq 0.125$ のとき

・単精度

$$Z(u) = \frac{\pi \cdot 2q}{K(m)} \cdot \frac{\sin(w) - 2q^3 \cdot \sin(2w) + 3q^8 \cdot \sin(3w)}{1 - 2q(\cos(w) - q^3 \cdot \cos(2w) + q^8 \cdot \cos(3w))} \cdot s$$

・倍精度

$$Z(u) = \frac{\pi \cdot 2q}{K(m)} \cdot \frac{\sin(w) - 2q^3 \cdot \sin(2w) + 3q^8 \cdot \sin(3w) - 4q^{15} \cdot \sin(4w)}{1 - 2q(\cos(w) - q^3 \cdot \cos(2w) + q^8 \cdot \cos(3w) - q^{15} \cdot \cos(4w))} \cdot s$$

② $q > 0.125$ のとき

$$z = \exp(-\pi u/K(m')) \text{ として}$$

・単精度

$$Z(u) = \frac{\pi}{2k(m')} \cdot \left(\frac{-5q'^6 z^5 - 3q'^2 z^4 - z^3 + z^2 + 3q'^2 z + 5q'^6}{q'^6 z^5 + q'^2 z^4 + z^3 + z^2 + q'^2 z + q'^6} - 2v \right) \cdot s$$

・倍精度

$$Z(u) = \frac{\pi}{2k(m')} \cdot \left(\frac{(z^3 - z^4) + 3q'^2(z^2 - z^5) + 5q'^6(z - z^6) + 7q'^{12}(1 - z^7)}{(z^3 + z^4) + q'^2(z^2 + z^5) + q'^6(z + z^6) + q'^{12}(1 + z^7)} - 2v \right) \cdot s$$

③ $q > \begin{cases} \text{倍精度 } 0.8 \\ \text{単精度 } 0.6 \end{cases}$ としても、 $v \leq 0.5$ のとき

$$Z(u) = \tanh(u) - 2v$$

(8) ヤコビのエプシロン関数 $E(u)$ ($0.0 \leq m \leq 1.0$)

$$E(u) = Z(u) + \frac{E(m)}{K(m)} \cdot u \text{ (} E(m), K(m) \text{ は、ノーム } q \text{ により計算する。)}$$

(9) ヤコビのテータ関数 $\Theta(u)$ ($0.0 \leq m \leq 1.0$)

$$\Theta(u) = \vartheta_4(u/2 \cdot K(m), q) \text{ (} q, K(m) \text{ はノーム } q, \vartheta_4 \text{ はテータ関数により計算する。)}$$

ただし、 $m = 1.0$ のときは $\Theta(u) = 0.0$ とする。

(10) パイ関数 $\Pi(u, \alpha)$ ($0.0 \leq m < 1.0$)

$$\Pi(u, \alpha) = \frac{1}{2} \log \frac{\Theta(u-\alpha)}{\Theta(u+\alpha)} + u \cdot Z(\alpha)$$

2.1.2.8 初等関数の不定積分

(1) 指数積分 $\overline{\text{Ei}}(x), \text{Ei}(-x)$ ($x > 0.0$)

$x < 0.0$ のときは $\text{Ei}(x)$, $x > 0.0$ のときは $\overline{\text{Ei}}(x)$ を計算する。以下にその方法を述べる。

① $x < -4.0$ のとき

$$\text{Ei}(x) = \frac{e^x}{x} \left\{ 1 + \left(-\frac{1}{x} \right) \frac{\sum_{k=0}^m a_k^{(1)} \left(-\frac{1}{x} \right)^k}{\sum_{k=0}^m b_k^{(1)} \left(-\frac{1}{x} \right)^k} \right\}$$

② $-4.0 \leq x < -1.0$ のとき

$$\text{Ei}(x) = -e^x \frac{\sum_{k=0}^m a_k^{(2)} \left(-\frac{1}{x} \right)^k}{\sum_{k=0}^m b_k^{(2)} \left(-\frac{1}{x} \right)^k}$$

③ $-1.0 \leq x < 0.0$ のとき

$$\text{Ei}(x) = \log(-x) + \gamma + \sum_{n=1}^{\infty} \frac{(-x)^n}{n!n}$$

を最良近似式化して求める (γ はオイラーの定数).
最良近似式化については 2.1.2.22 に記述されている.

④ $0.0 < x \leq 1.0$ のとき

$$\overline{\text{Ei}}(x) = \log(x) + \gamma + \sum_{n=1}^{\infty} \frac{x^n}{n!n}$$

を最良近似式化して求める.
最良近似式化については 2.1.2.22 に記述されている.

⑤ $1.0 < x \leq 6.0$ のとき

$$\overline{\text{Ei}}(x) = \log\left(\frac{x}{x_0}\right) + (x - x_0) \frac{\sum_{k=0}^m a_k^{(3)} x^k}{\sum_{k=0}^m b_k^{(3)} x^k}$$

($x_0 = 0.37250741078136663446$)

⑥ $6.0 < x \leq 12.0$ のとき

$$\overline{\text{Ei}}(x) = \frac{e^x}{x} \left(a_0^{(4)} + \frac{m}{\Phi} \frac{b_{k-1}^{(4)}}{a_k^{(4)} + x} \right)$$

を計算する.

⑦ $12.0 < x \leq 24.0$ のとき

$$\overline{\text{Ei}}(x) = \frac{e^x}{x} \left(a_0^{(5)} + \frac{m}{\Phi} \frac{b_{k-1}^{(5)}}{a_{k-1}^{(5)} + x} \right)$$

⑧ $x > 24.0$ のとき

$$\overline{\text{Ei}}(x) = \frac{e^x}{x} \left\{ 1 + \frac{1}{x} \left(a_0^{(6)} + \frac{m}{\Phi} \frac{b_{k-1}^{(6)}}{a_k^{(6)} + x} \right) \right\}$$

係数 $a_k^{(1)} \sim a_k^{(6)}, b_k^{(1)} \sim b_k^{(6)}$ は文献 (14), (15) に記述されている. また, $a_k^{(3)}, b_k^{(3)}$ は, ずらしチェビシェフ多項式の係数をテレスコーピングして求める.

(2) 対数積分 $\text{Li}(x)$ ($x \geq 1.0$)

指数積分を利用して

$$\text{Li}(x) = \overline{\text{Ei}}(\log(x))$$

を計算する.

(対数積分 $\text{Li}(x)$ は $\text{li}(x)$ とも書かれる.)

(3) 正弦積分 $\text{Si}(x)$

① $x < 0.0$ のとき

$\text{Si}(x) = -\text{Si}(-x)$ として $\text{Si}(-x)$ について以下に述べる方法を適用する.

② $x \geq 0.0$ のとき

i) $x \geq \left\{ \begin{array}{l} \text{倍精度} : 2^{50} \cdot \pi \\ \text{単精度} : 2^{18} \cdot \pi \end{array} \right\}$ の場合

$\text{Si}(x) = \frac{\pi}{2}$ とする.

ii) i) 以外のとき

イ) $x \leq 5.0$ のとき

$$\text{Si}(x) = \sum_{k=0}^{\infty} \frac{(-1)^k \cdot x^{2k+1}}{(2k+1) \cdot (2k+1)!}$$

を最良近似式化して求める.

最良近似式化については 2.1.2.22 に記述されている.

ロ) $x \geq 42.0$ のとき

$\text{Si}(x) = \frac{\pi}{2} - f(x) \cdot \cos(x) - g(x) \cdot \sin(x)$ で計算する.

ここで $f(x), g(x)$ は次の計算式を最良近似式化して求める.

$$f(x) = \sum_{k=0}^{\infty} \frac{(-1)^k \cdot (2k)!}{x^{2k+1}}$$

$$g(x) = \sum_{k=0}^{\infty} \frac{(-1)^k \cdot (2k+1)!}{x^{2k+2}}$$

最良近似式については 2.1.2.22 に記述されている.

ハ) $5.0 < x < 42.0$ のとき

$$Q = \text{Si}(x_n) \left(\begin{array}{l} x_n \text{ は } x \text{ に最も近い整数値で } \text{Si}(x_n) \text{ は } x_n \text{ における正弦積分} \\ \text{の真値} \end{array} \right)$$

$z = x - x_n$ とする.

a) $|z| < (\text{誤差判定のための単位}) \cdot x$ ならば

$\text{Si}(x) = \text{Si}(x_n)$ とする.

b) その他のとき

$$f^{(J)}(x_n) = \{(\sin(x_n))^{(J-1)} - (J-1) \cdot f^{(J-1)}(x_n)\} / x_n$$

$$QQ = f^{(J)}(x_n) \cdot Z^J / J!$$

$$Q = Q + QQ \quad (J = 1, 2, 3, \dots)$$

を、 $|QQ| < \|Q \cdot (\text{誤差判定のための単位})\|$ になるまで繰返し、

$$\text{Si}(x) = Q$$

とする.

(4) 余弦積分 $\text{Ci}(x)$ ($x \geq 0.0$)

① $x \leq 2.0$ のとき

$$\text{Ci}(x) = \gamma + \log(x) + \sum_{k=1}^{\infty} \frac{(-1)^k \cdot x^{2k}}{2k \cdot (2k)!}$$

$$(\gamma = 0.57721566490153286061)$$

を最良近似式化して求める.

最良近似式化については 2.1.2.22 に記述されている.

② $x \geq 42.0$ のとき

$$\text{Ci}(x) = f(x) \cdot \sin(x) - g(x) \cdot \cos(x)$$

ここで, $f(x), g(x)$ は, (35) 正弦積分と同じ.

③ $2.0 < x < 42.0$ のとき

$Q = \text{Ci}(x_n)$ (x_n は x に最も近い整数値で $\text{Ci}(x_n)$ は x_n における余弦積分の真値)

$z = x - x_n$ とする.

イ) $|z| < (\text{誤差判定のための単位}) \cdot x$ ならば

$\text{Ci}(x) = \text{Ci}(x_n)$ とする.

ロ) その他のとき

$$f^{(1)}(x_n) = (\cos(x_n) - 1)/x_n$$

$$Q = Q + f^{(1)}(x_n) \cdot z$$

とおき,

$$f^{(J)}(x_n) = \{(\cos(x_n))^{(J-1)} - (J-1) \cdot f^{(J-1)}(x_n)\}/x_n$$

$$QQ = f^{(J)}(x_n) \cdot z^J / J!$$

$$Q = Q + QQ \quad (J = 2, 3, \dots)$$

を, $|QQ| < |Q \cdot (\text{誤差判定のための単位})|$ になるまで繰り返し,

$$\text{Ci}(x) = Q + \log\left(\frac{x}{x_n}\right)$$

とする.

(5) フレネル積分 $S(x), C(x)$

次の式で定義されるフレネル積分を計算する.

$$S(x) = \int_0^x \sin\left(\frac{\pi}{2} \cdot t^2\right) dt$$

$$C(x) = \int_0^x \cos\left(\frac{\pi}{2} \cdot t^2\right) dt$$

① $x < 0.0$ のとき

$S(x) = -S(-x), C(x) = -C(-x)$ として $S(-x), C(-x)$ について以下に述べる方法を適用する.

② $x \geq 0.0$ のとき

i) $x \leq 1.6$ のとき

$$S(x) = \sum_{k=0}^{\infty} \frac{(-1)^k \cdot \left(\frac{\pi}{2}\right)^{2k+1}}{(2k+1)! \cdot (4k+3)} \cdot x^{4k+3}$$

$$C(x) = \sum_{k=0}^{\infty} \frac{(-1)^k \cdot \left(\frac{\pi}{2}\right)^{2k}}{(2k)! \cdot (4k+1)} \cdot x^{4k+1}$$

を最良近似式化して求める.

最良近似式化については 2.1.2.22 に記述されている.

ii) $x \geq 5.0$ のとき

$$S(x) = \frac{1}{2} - \left\{ f(x) \cdot \cos\left(\frac{\pi}{2} \cdot x^2\right) + g(x) \cdot \sin\left(\frac{\pi}{2} \cdot x^2\right) \right\}$$

$$C(x) = \frac{1}{2} + \left\{ f(x) \cdot \sin\left(\frac{\pi}{2} \cdot x^2\right) - g(x) \cdot \cos\left(\frac{\pi}{2} \cdot x^2\right) \right\}$$

ここで, $f(x), g(x)$ は, 次の計算式を最良近似式化して求める.

$$f(x) = x \cdot \sum_{k=0}^m \frac{(-1)^k \cdot (4k-1)!!}{(\pi \cdot x^2)^{2k+1}}$$

$$g(x) = x \cdot \sum_{k=0}^m \frac{(-1)^k \cdot (4k+1)!!}{(\pi \cdot x^2)^{2k+2}}$$

最良近似式化については 2.1.2.22 に記述されている.

iii) $1.6 < x < 5.0$ のとき

$Y = \frac{\pi}{2} \cdot x^2$ で変換し,

$$S(x) = \frac{1}{2} - \{f(Y) \cdot \cos(Y) + g(Y) \cdot \sin(Y)\}$$

$$C(x) = \frac{1}{2} + \{f(Y) \cdot \sin(Y) - g(Y) \cdot \cos(Y)\}$$

ここで, $f(Y), g(Y)$ は,

単精度:

$$f(Y) = \frac{1}{x} \sum_{k=0}^7 a_k^{(1)} \left(\frac{4}{Y}\right)^k$$

$$g(Y) = \frac{1}{x} \sum_{k=0}^8 b_k^{(1)} \left(\frac{4}{Y}\right)^k$$

倍精度:

$$f(Y) = \frac{1}{x} \sum_{k=0}^{34} a_k^{(2)} \left(\frac{4}{Y}\right)^k$$

$$g(Y) = \frac{1}{x} \sum_{k=0}^{34} b_k^{(2)} \left(\frac{4}{Y}\right)^k$$

係数 $a_k^{(1)}, b_k^{(1)}, a_k^{(2)}, b_k^{(2)}$ はテレスコーピングにより求めた.

(6) ドーソン積分 $F(x)$

ドーソン積分の値は次の式で定義される.

$$F(x) = e^{-x^2} \int_0^x e^{t^2} dt$$

① $x \geq 0.0$ のとき

$F(x) = -F(-x)$ として $F(-x)$ について以下に述べる方法を適用する.

② $x \geq 0.0$ のとき

i) $x \leq 2.5$ ならば

$$F(x) = x \frac{\sum_{k=0}^m a_k^{(1)} x^{2k}}{\sum_{k=0}^m b_k^{(1)} x^{2k}}$$

ii) $2.5 < x \leq 3.5$ ならば

$$F(x) = \frac{1}{x} \cdot \left(a_0^{(2)} + \frac{m}{\Phi} \frac{b_{k-1}^{(2)}}{a_k^{(2)} + x^2} \right)$$

iii) $3.5 < x \leq 5.0$ ならば

計算式は ii) と同じで係数 $a_k^{(3)}, b_k^{(3)}$ とする

iv) $5.0 < x \leq 1/\varepsilon$ ならば (ε : 誤差判定のための単位)

$$F(x) = \frac{1}{2x} \cdot \left\{ 1 + x^{-2} \cdot \left(a_0^{(4)} + \frac{m}{\Phi} \frac{b_{k-1}^{(4)}}{a_k^{(4)} + x^2} \right) \right\}$$

v) $x > 1/\varepsilon$ ならば

$$F(x) = \frac{1}{2x}$$

係数 $a_k^{(1\sim4)}, b_k^{(1\sim4)}$ は文献 (15) に記述されている.

(7) 正規分布関数, 余正規分布関数 $\Phi(x), \Psi(x)$

次の式で計算する.

$$\Phi(x) = \frac{1}{2} \cdot \text{Erf}(x/\sqrt{2})$$

$$\Psi(x) = \frac{1}{2} \cdot \operatorname{Erfc}(x/\sqrt{2})$$

ここに Erf, Erfc は、誤差関数 および 補誤差関数 (余誤差関数と呼ぶこともある) で、これらの定義および計算方法については 2.1.2.21 を参照のこと。

2.1.2.9 ルジャンドル陪関数

(1) 第 1 種ルジャンドル陪関数 $P_n^m(x)$

① 正規化されたものについては、球面調和関数の項を参照のこと。

② $n < 0$ のとき

$P_n^m(x) = P_{-n-1}^m(x)$ として $P_{-n-1}^m(x)$ について以下に述べる方法を適用する。

③ $m < 0$ かつ $n \geq 0$ のとき

③ または ⑤ に述べる方法によって $P_n^{|m|}(x)$ を求め次の関係式より $P_n^m(x)$ を計算する。

i) $|x| \leq 1.0$ ならば

$$P_n^m(x) = (-1)^m \frac{(n - |m|)!}{(n + |m|)!} P_n^{|m|}(x)$$

ii) $|x| > 1.0$ ならば

$$P_n^m(x) = \frac{(n - |m|)!}{(n + |m|)!} P_n^{|m|}(x)$$

④ $m > n \geq 0$ のとき

$$P_n^m(x) = 0.0$$

⑤ $m = 0$ のとき、 $P_n^m(x)$ はルジャンドル多項式 $P_n(x)$ に等しいので $P_0(x) = 1, P_1(x) = x$ を初期値として次の漸化式を計算し $P_n^m(x)$ を求める。

$$P_k(x) = \frac{2k-1}{k} \cdot x \cdot P_{k-1}(x) - \frac{k-1}{k} \cdot P_{k-2}(x) \quad (k = 2, \dots, n)$$

⑥ $n \geq m > 0$ のとき

i) $n = 1$ で $x \geq \sqrt{1/\varepsilon}$ ならば

$$P_n^m(x) = x$$

ii) $n = m$ ならば

$$P_n^m(x) = (2n - 1)!! \cdot (\sqrt{|1 - x^2|})^m$$

iii) $n = m + 1$ ならば

$$P_n^m(x) = (2n - 1)!! \cdot x \cdot (\sqrt{|1 - x^2|})^m$$

iv) $n \geq m + 2$ ならば

$$F_m = (2m - 1)!!, F_{m+1} = (2m + 1)!! \cdot x = F_m \cdot (2m + 1) \cdot x$$

を初期値に、次の漸化式を計算し $P_n^m(x)$ を求める。

$$F_k = \frac{2k-1}{k-m} \cdot x \cdot F_{k-1} - \frac{k+m-1}{k-m} \cdot F_{k-2} \quad (k = m + 2, \dots, n)$$

$$P_n^m(x) = (\sqrt{|1 - x^2|})^m \cdot F_n$$

(2) 第 2 種ルジャンドル陪関数 $Q_n^m(x)$ ($n \geq 0, |x| = 1.0$)

① $m < 0$ のとき

② または ③ に述べる方法によって $Q_n^{|m|}$ を求め、次の関係式より $Q_n^m(x)$ を計算する。

i) $|x| < 1.0$ ならば

$$Q_n^m(x) = (-1)^m \frac{(n - |m|)!}{(n + |m|)!} Q_n^{|m|}(x)$$

ii) $|x| > 1.0$ ならば

$$Q_n^m(x) = \frac{(n - |m|)!}{(n + |m|)!} Q_n^{|m|}(x)$$

② $|x| < 1.0$ かつ $m \geq 0$ のとき

$Q_0^0(x) = \operatorname{artanh}(x)$, $Q_1^0(x) = x \cdot Q_0^0(x) - 1$ を初期値に次の漸化式を実行し,

$Q_n^0(x), Q_{n-1}^0(x)$ を求める ($m = 0$ のときは, これで打ち切る).

$$Q_{k+1}^0(x) = \{(2k+1) \cdot x \cdot Q_k^0(x) - k \cdot Q_{k-1}^0(x)\} / (k+1) \quad (k = 1, 2, \dots, n-1)$$

次に, $Q_n^1(x) = \{-n \cdot x \cdot Q_n^0(x) + n \cdot Q_{n-1}^0(x)\} / \sqrt{1-x^2}$ により $Q_n^1(x)$ を求める.

なお, $n = 0$ のときは, $Q_n^1(x) = -1/\sqrt{1-x^2}$ とする ($m = 1$ のときは, これで打ち切る).

さらに次の漸化式を計算し, $Q_n^m(x)$ を求める.

$$Q_n^{k+2}(x) = 2 \cdot (k+1) \cdot (x/\sqrt{1-x^2}) \cdot Q_n^{k+1}(x) - (n-k) \cdot (n+k+1) \cdot Q_n^k(x) \quad (k = 0, 1, \dots, m-2)$$

③ $|x| > 1.0$ かつ $m \geq 0$ のとき

i) $x > \sqrt{n+2}$ および $n \geq m$ のとき

イ) $n = m = 0$ のとき

$$Q_n^m(x) = \operatorname{artanh}(1/x)$$

ロ) $|x| > \sqrt{(n+0.5)/\varepsilon}$ のとき

$$Q_n^m(x) = (-1)^m \frac{(n+m)!}{(2n+1)!!} x^{-n-1}$$

ハ) その他のとき次の級数展開式を計算する.

$$Q_n^m(x) = (-1)^m (x^2 - 1)^{\frac{m}{2}} \frac{(n+m)!}{(2n+1)!!} x^{-n-m-1} \cdot \left(1 + \sum_{k=1}^{\infty} \frac{(n+m+1) \cdot (n+m+2) \cdots (n+m+2 \cdot k)}{2k!! \cdot (2n+3) \cdot (2n+5) \cdots (2n+2k+1)} \cdot \frac{1}{x^{2k}} \right)$$

ここで Σ 計算は, 末項が初項に対し十分小さくなるまで計算する.

ii) i) 以外のとき

まず, $Q_0^0(x)$ を次式より求める.

$$Q_0^0(x) = \operatorname{artanh}(1/x)$$

($n = m = 0$ のときは, これで打ち切る.)

次に

$$f = \sum_{k=n}^{\infty} \frac{1}{(k+1) \cdot P_k(x) \cdot P_{k+1}(x)}$$

として級数展開式を計算し, $Q_n^0(x), Q_{n-1}^0(x)$ を次式より求める.

$$Q_n^0(x) = P_n(x) \cdot f$$

$$Q_{n-1}^0(x) = P_{n-1}(x) \cdot f + \frac{1}{n \cdot P_n(x)}$$

ここで, $P_k(x)$ は 2.1.2.10 (1) に示すアルゴリズムを用いて解く.

($m = 0$ のときは, これで打ち切る.)

次に,

$$Q_n^1(x) = \frac{n \cdot x \cdot Q_n^0(x) - n \cdot Q_{n-1}^0(x)}{\sqrt{x^2 - 1}}$$

より $Q_n^1(x)$ を求める.

なお, $n = 0$ のときは $Q_n^1(x) = -1/\sqrt{x^2 - 1}$ とする

($m = 1$ のときは, これで打ち切る).

さらに次の漸化式を計算し, $Q_n^m(x)$ を求める.

$$Q_n^{k+2}(x) = -2(k+1)(x/\sqrt{x^2-1})Q_n^{k+1}(x) + (n-k)(n+k+1)Q_n^k(x) \quad (k = 0, 1, \dots, m-2)$$

2.1.2.10 直交多項式

(1) ルジャンドル多項式 $P_n(x)$ ($n \geq 0$)

$P_0(x) = 1.0, P_1(x) = x$ を初期値に次の漸化式を計算し $P_n(x)$ を求める.

$$P_k(x) = \frac{2k-1}{k} \cdot x \cdot P_{k-1}(x) - \frac{k-1}{k} \cdot P_{k-2}(x) \quad (k = 2, 3, \dots, n)$$

(2) ラゲール多項式 $L_n(x)$ ($n \geq 0$)

$L_0(x) = 1.0, L_1(x) = 1.0 - x$ を初期値に次の漸化式を計算し $L_n(x)$ を求める.

$$L_k(x) = \frac{(2k-x-1)}{k} \cdot L_{k-1}(x) - \frac{(k-1)}{k} \cdot L_{k-2}(x) \quad (k = 2, 3, \dots, n)$$

(3) エルミート多項式 $H_n(x)$ ($n \geq 0$)

$H_0(x) = 1.0, H_1(x) = 2 \cdot x$ を初期値に次の漸化式を計算し $H_n(x)$ を求める.

$$H_k(x) = 2 \cdot x \cdot H_{k-1}(x) - 2 \cdot (k-1) \cdot H_{k-2}(x) \quad (k = 2, 3, \dots, n)$$

(4) チェビシエフ多項式 $T_n(x)$ ($n \geq 0$)

$T_0(x) = 1.0, T_1(x) = x$ を初期値に次の漸化式を計算し $T_n(x)$ を求める.

$$T_k(x) = 2 \cdot x \cdot T_{k-1}(x) - T_{k-2}(x) \quad (k = 2, 3, \dots, n)$$

(5) 第2種チェビシエフ関数 $U_n(x)$ ($n \geq 0, |x| \leq 1.0$)

$$U_0(x) = 0.0,$$

$$U_1(x) = \sqrt{1-x^2} \text{ として,}$$

$$U_{n+1}(x) = 2 \cdot x \cdot U_n(x) - U_{n-1}(x) \text{ を計算する.}$$

(6) 一般ラゲール多項式 $L_n^{(\alpha)}(x)$ ($n \geq 0$)

$$L_0^{(\alpha)}(x) = 1.0,$$

$$L_1^{(\alpha)}(x) = (1+\alpha) - x \text{ として,}$$

$$L_n^{(\alpha)}(x) = ((2 \cdot n + \alpha - x - 1) \cdot L_{n-1}^{(\alpha)}(x) + (1 - \alpha - n) \cdot L_{n-2}^{(\alpha)}(x)) / n$$

を計算する.

2.1.2.11 整数次マシュー関数

以下のような常微分方程式を考える:

$$\frac{d^2y}{dx^2} + (a - 2q \cos 2x)y = 0.$$

実数のパラメータ q に対して, 周期 2π の偶関数解 $y = y(x)$ をもつような実数 $a = c_0, c_1, \dots$ の全体を

$$c_0 < c_1 < c_2 < \dots$$

とする. また, 周期 2π の奇関数解をもつような実数 $a = s_1, s_2, \dots$ の全体を

$$s_1 < s_2 < s_3 < \dots$$

とする. $a = c_n (n \geq 0)$ に対して, 周期偶関数解を以下の条件で正規化する:

$$\int_0^{2\pi} ce_n(x, q)^2 dx = \pi.$$

この周期偶関数解 $ce_n(x, q)$ は, 整数次マシュー関数 $ce_n(x, q)$ と呼ばれている.

同様に, $a = s_n (n \geq 1)$ のときの周期奇関数解を以下の条件で正規化する:

$$\int_0^{2\pi} se_n(x, q)^2 dx = \pi.$$

この周期奇関数解 $se_n(x, q)$ も, 整数次マシュー関数 $se_n(x, q)$ と呼ばれている.

$ce_n(x, q), se_n(x, q)$ は, 以下のフーリエ展開で与えられることが知られている:

(1) n が偶数の場合の $ce_n(x, q)$ ($s=0$ とする)

$$ce_n(x, q) = \sum_{r=0}^{\infty} X_r \cos(2rx).$$

(2) n が奇数の場合の $se_n(x, q)$ ($s=1$ とする)

$$se_n(x, q) = \sum_{r=0}^{\infty} X_r \sin((2r+1)x).$$

(3) n が奇数の場合の $ce_n(x, q)$ ($s=2$ とする)

$$ce_n(x, q) = \sum_{r=0}^{\infty} X_r \cos((2r+1)x).$$

(4) n が正の偶数の場合の $se_n(x, q)$ ($s=3$ とする)

$$se_n(x, q) = \sum_{r=0}^{\infty} X_r \sin((2r+2)x).$$

これらは、 n 次のマシュー関数と呼ばれている。展開係数 X_r を求めるために、これらの式を常微分方程式に代入すると、以下のような実対称 3 重対角行列 $A_s(q)$ の固有値問題に帰着する。

$$A_s(q) = \begin{bmatrix} K_s^2 + d_s q & R_s q & & & & & & & \\ & R_s q & (2 + K_s)^2 & q & & 0 & & & \\ & & q & (4 + K_s)^2 & q & & & & \\ & 0 & & q & (6 + K_s)^2 & \ddots & & & \\ & & & & \ddots & \ddots & & & \\ & & & & & & \ddots & & \ddots \end{bmatrix}$$

ここで、

$$(1) R_0 = \sqrt{2}, R_1 = R_2 = R_3 = 1$$

$$(2) K_0 = 0, K_1 = K_2 = 1, K_3 = 2$$

$$(3) d_0 = 0, d_1 = -1, d_2 = 1, d_3 = 0$$

で、 $s=0, 1, 2, 3$ である。この s は、計算すべきマシュー関数の種類 (つまり、 $ce_n(x, q)$ もしくは $se_n(x, q)$) および次数 n の奇偶で定まる。 $A_s(q)$ の固有ベクトル $[R_s X_0, X_1, X_2, X_3, \dots]^t$ は、大きさを 1 とする。また、整数 l を

(1) $ce_n(x, q)$ を求めるとき

$$l = 1 + [n/2]$$

(2) $se_n(x, q)$ を求めるとき

$$l = n/2 \quad (n \text{ が偶数のとき})$$

$$l = (n+1)/2 \quad (n \text{ が奇数のとき})$$

としたとき、小さい方から l 番目の固有値に対応する固有ベクトルの向きは、 $X_l > 0$ となるようにとる。

この手法によって得られたマシュー関数 $ce_n(x, q)$, $se_n(x, q)$ は、 $q \rightarrow 0$ のとき

$$(1) ce_0(x, q) \rightarrow \frac{1}{\sqrt{2}}$$

$$(2) ce_n(x, q) \rightarrow \cos nx \quad (n = 1, 2, \dots)$$

$$(3) se_n(x, q) \rightarrow \sin nx \quad (n = 1, 2, \dots)$$

と収束する。

2.1.2.12 ランジュバン関数

(1) $x < 0.0$ のとき

$L(x) = -L(-x)$ として $x > 0.0$ の範囲で計算する.

(2) $x \geq 0.0$ のとき

i) $x \leq 1.5$ のとき

$$L(x) = x \cdot \sum_{k=1}^{\infty} (-1)^{k-1} \left\{ \frac{2^{2k} \cdot B_k \cdot x^{2k+2}}{(2k)!} \right\} \text{ を最良近似式化して求める.}$$

最良近似式化については 2.1.2.22 に記述されている.

ii) $1.5 < x < \begin{cases} \text{倍精度: 45.0} \\ \text{単精度: 20.0} \end{cases}$ のとき

$$L(x) = \frac{2 \cdot e^{-2 \cdot x}}{1 - e^{-2 \cdot x}} - \frac{1}{x} + 1$$

iii) $x \geq \begin{cases} \text{倍精度: 45.0} \\ \text{単精度: 20.0} \end{cases}$ のとき

$$L(x) = 1 - \frac{1}{x}$$

2.1.2.13 ガウス・ルジャンドル積分公式

(1) ガウス・ルジャンドル積分公式

次数が $2N - 1$ 以下の任意の多項式 $F(x)$ に対し, 定数 b_0, b_1, \dots, b_{N-1} を適当にとれば, N 次ルジャンドル多項式 $P_N(x)$ で $F(x)$ を割った余りを $P_j(x) (j = 0, 1, \dots, N - 1)$ の線型結合で表示して

$$F(x) = P_N(x)Q(x) + b_0 + \sum_{j=1}^{N-1} \sqrt{2j+1} b_j P_j(x)$$

($Q(x)$ は次数が $N - 1$ 以下の多項式) とすることができる. ここで, 次数が $N - 1$ 以下の任意の多項式 $Q(x)$ は $P_j(x) (j = 0, 1, \dots, N - 1)$ の線型結合で表示され, 直交性

$$\int_{-1}^{+1} P_N(x) P_j(x) dx = 0 \quad (j = 0, 1, \dots, N - 1)$$

により

$$\int_{-1}^{+1} P_N(x) Q(x) dx = 0.$$

さらに,

$$\int_{-1}^{+1} P_j(x) dx = 0 \quad (j = 1, 2, \dots, N - 1)$$

に注意して, $F(x)$ の表示式から

$$\int_{-1}^{+1} F(x) dx = 2b_0.$$

$P_N(x)$ の N 個の零点を $\alpha_k (k = 1, 2, \dots, N)$ とする. $F(x)$ の表示式に $x = \alpha_k$ を代入して

$$F(\alpha_k) = b_0 + \sum_{j=1}^{N-1} \sqrt{2j+1} b_j P_j(\alpha_k) \quad (k = 1, 2, \dots, N)$$

である. いま, $k = 1, 2, \dots, N$ について

$$d_k^2 = \sum_{n=0}^{N-1} (2n+1)P_n^2(\alpha_k)$$

となる d_k^2 をとると, $F(\alpha_k)$ の上記表示と

$$\sum_{k=1}^N d_k^{-2} P_j(\alpha_k) = 0 \quad (j = 1, 2, \dots, N-1),$$

$$\sum_{k=1}^N d_k^{-2} = 1$$

により

$$b_0 = \sum_{k=1}^N d_k^{-2} F(\alpha_k)$$

したがって, $\alpha_k (k = 1, 2, \dots, N)$ において, そこでの値 $F(\alpha_k)$ が積分すべき関数 $f(x)$ の値と一致するように多項式 $F(x)$ を選べば, $2N-1$ 次の積分公式

$$\begin{aligned} \int_{-1}^{+1} f(x) dx &\simeq \int_{-1}^{+1} F(x) dx = 2b_0 = 2 \sum_{k=1}^N d_k^{-2} F(\alpha_k) \\ &= 2 \sum_{k=1}^N d_k^{-2} f(\alpha_k) \end{aligned}$$

がえられる. ここで, 次数が $2N-1$ 以下の多項式 $F(x)$ は N 個の零点における値に対する条件を満たしていればよい. すなわち, $F(x)$ の選び方には最大 N 個の自由度がある. この積分公式は, 被積分関数 $f(x)$ を多項式 $F(x)$ で近似出来ない場合 (例えば振動する関数) には次数を十分に大きくとる等して, 適用しなければならない.

(2) ルジャンドル多項式の零点

$X_n = \sqrt{2n+1}P_n(\alpha)$ とおく. α が $P_N(\alpha) = 0$ をみたせば, $a_n = n/\sqrt{4n^2-1}$ として

$$\begin{aligned} a_1 X_1 &= \alpha X_0 \\ a_2 X_2 + a_1 X_0 &= \alpha X_1 \\ a_3 X_3 + a_2 X_1 &= \alpha X_2 \\ &\vdots \\ a_{N-1} X_{N-1} + a_{N-2} X_{N-3} &= \alpha X_{N-2} \\ a_{N-1} X_{N-2} &= \alpha X_{N-1} \end{aligned}$$

が成り立ち, ルジャンドル多項式の零点が単根であることから, これらは対角成分が0, 副対角成分が a_1, a_2, \dots, a_{N-1} である実対称3重対角行列の固有値として得られる. 高次数のルジャンドル多項式では, すべての零点が開区間 $(-1,1)$ に密集して存在するため, ニュートン法で零点を求めるかわりに, この実対称3重対角行列の固有値問題を無平方根QR法で解くことにより, N 個の零点すべてを確実に求めることができる.

2.1.2.14 ベッセル関数の零点

- (1) ベッセル関数を含む超越方程式 $aJ_0(\alpha) + \alpha J_1(\alpha) = 0$ の正の解
超越方程式

$$aJ_0(\alpha) + \alpha J_1(\alpha) = 0$$

の正の解 α は以下の手順で求める.

この方法で超越方程式の正の解を $M=50$ 個程度まで求めることができる.

(単精度では $M \leq 24$ すなわち $N \leq 192$ である.)

パラメータ a は, $a=0$, または, $10^{-10} \leq |a| \leq 10^4$ が, 適用範囲である.

1. 近似値を有限要素法によって求める.

微分方程式

$$u''(r) + u'(r)/r + \alpha^2 u(r) = 0$$

の境界条件

$$u'(1) = au(1)$$

を満たす非自明な解 $u(r) (0 \leq r \leq 1)$ で $u(+0)$ が有限なもの, α が $aJ_0(\alpha) + \alpha J_1(\alpha) = 0$ を満たすときに限り存在することができて, $u(r) = CJ_0(\alpha r)$ である.

$r = 0, 1/N, 2/N, \dots, (N-1)/N, 1$ に節点を取り, 各節点において $w_j(j/N) = 1, w_j(k/N) = 0 (k \neq j)$ をみたす基底関数 $w_j(r)$ を考える. この w_j を使って, $u(r)$ を

$$u(r) = \sum_{j=0}^N u(j/N)w_j(r)$$

と基底関数補間する. 重み付き残差法を適用して

$$\int_0^1 w_j(r)(u''(r) + u'(r)/r + \alpha^2 u(r))rdr = 0$$

が得られる. さらに 2 回微分を含む項には部分積分を用いて変形すれば, これは積分関係式

$$\int_0^1 w'_j(r)u'(r)rdr - aw_j(1)u(1) = \alpha^2 \int_0^1 w_j(r)u(r)rdr$$

となる. これに基底関数補間を代入すれば

$$AX = \alpha^2 BX, \quad X = (u(0), u(1/N), \dots, u(1))^t,$$

となる. ここで, A, B は,

$$A_{i,j} = \int_0^1 w'_j(r)w'_i(r)rdr - \delta_{i,N}\delta_{j,N}a,$$

$$B_{i,j} = \int_0^1 w_j(r)w_i(r)rdr$$

である. よって, $aJ_0(\alpha) + \alpha J_1(\alpha) = 0$ の解 $\alpha > 0$ の近似値は, 一般化固有値問題 $AX = \beta BX$ の正の固有値 β を求めて, $\sqrt{\beta}$ とすることで得られる.

2. バイセクション法およびニュートン法による解の改良

$N = LM$ とする. ここで, L は近似倍率であり, 8 程度の値が望ましい.

一般化固有値の正の値の平方根は, 求める超越方程式の正の解の近似値である.

E を

$$E = 0(a < 0)$$

$$E = \alpha(0)^2(a \geq 0)$$

で定義する, ここで $\alpha(0)$ は $J_0(x)$ の最小正零点である.

以下, 一般化固有値のうちで E よりも大きいものを昇順に $\beta_1^*, \beta_2^*, \dots, \beta_M^*$ とする.

ここで, E 以下の固有値が 2 個以上存在しないことを検査する.

また, β_j^* は重根でないことを検査する.

① バイセクション法

もし, 近似解 $\alpha^* = \sqrt{\beta_j^*}$ ($j \geq 2$) が,

$$|aJ_0(\alpha^*) + \alpha^* J_1(\alpha^*)| < 0.1 * (|J_0(\alpha^*)| + |J_1(\alpha^*)|)$$

を満たすならば, ニュートン法による近似解の改良処理に進む.

この条件は, j が 10 以下程度なら成り立つ. しかし, j が大きいときは重み付き残差法の近似誤差によって

この条件がなりたたない. したがって, 後者の場合は, バイセクション法を用いて近似解 α^* を改良する.

バイセクション法では, 近似解 α^* が, 解に近づくにつれ $|aJ_0(\alpha^*) + \alpha^* J_1(\alpha^*)|$ は小さくなるという性質を利用する.

j 番目の解のみを含む区間 $(\sqrt{\beta_j^* - \delta}, \sqrt{\beta_j^* + \delta})$ を 2 分割し, 分割された区間の端点の 1 つを α^* に置き換える. ここで, δ は β_j^* の近似誤差程度の正数とする. α^* が上記の条件を満たすまで区間の分割を繰り返した後, ニュートン法による近似解の改良処理に進む.

② ニュートン法

ニュートン法により, $x = \alpha^*$ を $|aJ_0(x) + xJ_1(x)| < e$ を満たすまで反復改良する. ここで,

$e=10^{-11}$ 倍精度

$e=10^{-4}$ 単精度

(2) 整数次ベッセル関数 $J_{m+1}(x)$ の正零点

$m=1, 2, \dots$ のとき

$$J_m(x) = x^m F_m(x)$$

なる正則関数 $F_m(x)$ を定めることができる. これは

$$F_m''(x) + (2m + 1)F_m'(x)/x + F_m(x) = 0$$

を満たす. また, $x^m F_m'(x) = -J_{m+1}(x)$. よって, アルゴリズム (1) の方法を

$$A_{i,j} = \int_0^1 w_j'(r)w_i'(r)r^{2m+1} dr$$

$$B_{i,j} = \int_0^1 w_j(r)w_i(r)r^{2m+1} dr$$

に対して適用する. ただし, 正の値であることを判定する場合の基準 E を

$$E = 10^{-3} \text{ (倍精度)}$$

$$E = 10^{-1} \text{ (単精度)}$$

とし, バイセクション法の誤差判定は,

$$|J_{m+1}(\alpha^*)| \leq 0.001$$

を用いている.

(3) 第 0 次, 第 1 次のベッセル関数 J_0, J_1 の正零点

$J_1(x)$ の零点は, アルゴリズム (1) で $a=0$ としたときの超越方程式の正の零点として得られる. a を十分大きな値にしたとき, 超越方程式 $aJ_0(\alpha) + \alpha J_1(\alpha) = 0$ の解は $J_0(x)$ の零点の近似値となる. したがって, $J_0(x)$ の零点は, 十分大きな a についてアルゴリズム (1) を適用することで得られる.

2.1.2.15 第 2 種ベッセル関数の正零点

① 窓区間 $[x_1, x_1 + 2\pi)$ ($x_1 > 0$) における第 2 種ベッセル関数 $Y_n(x)$ の正零点を以下の手順で求める.

(1) 各区間における正零点

初期値 $x = x_0$ を窓区間の中点にとる.

$F(x) = -Y_n(x) \cos x + J_n(x) \sin x$, $G(x) = Y_n(x) \sin x + J_n(x) \cos x$ とおき, ベクトル $(F(x), G(x))$ の偏角を α とする.

$$\cos \alpha = F(x)/S$$

$$\sin \alpha = G(x)/S$$

$$S = \sqrt{(F(x))^2 + (G(x))^2}$$

ここで, α は, この窓区間に入るように定める. 零点の近似値の改良値 x_{i+1} は, ベクトル $(F(x_i), G(x_i))$ の偏角で与えられる. この改良を反復する.

(2) 正零点の存在判定

もし, この反復改良が収束しないときは, この窓区間には正零点が存在しないとしてよい.

② 窓区間を 2π ずつずらした各区間について, 以上の手順を得られた正零点の個数が指定値になるまで反復する.

2.1.2.16 正定値 2 次形式 $x^2 + ay^2$ のゼータ関数

正定値 2 次形式 $x^2 + ay^2$ のゼータ関数

$$Z(s, a) = \sum_{(m,n) \in \mathbb{Z}^2, (m,n) \neq (0,0)} (m^2 + an^2)^{-s} \quad (s > 1)$$

から, この関数の極を消すための関数を引くことにより, 解析接続は

$$\begin{aligned} & Z(s, a) - \frac{\pi^s a^{-s/2}}{\Gamma(s)(s-1)} \\ &= \frac{\pi^s a^{-s/2}}{\Gamma(s)} \sum_{(m,n) \in \mathbb{Z}^2, (m,n) \neq (0,0)} (A_{m,n}^{-s} \int_{A_{m,n}}^{\infty} e^{-t} t^{s-1} dt + A_{m,n}^{s-1} \int_{A_{m,n}}^{\infty} e^{-t} t^{-s} dt) - \frac{\pi^s a^{-s/2}}{\Gamma(s+1)} \\ &= \frac{\pi^s a^{-s/2}}{\Gamma(s)} \sum_{(m,n) \in \mathbb{Z}^2, (m,n) \neq (0,0)} (A_{m,n}^{-s} \Gamma(s, A_{m,n}) + A_{m,n}^{s-1} \Gamma(1-s, A_{m,n})) - \frac{\pi^s a^{-s/2}}{\Gamma(s+1)} \end{aligned}$$

である。ここに, $A_{m,n} = \pi(\sqrt{am^2 + n^2}/\sqrt{a})$.

和をとるべき範囲は, $A_{m,n} < 40$ をみたす整数対 $(m, n) \neq (0, 0)$ の全部である。

2.1.2.17 ディログ関数

$$Li_2(x) = - \int_0^x \log|t-1| \frac{dt}{t}$$

の値は以下のとおりに求める。

- ① $x = 1$ のとき, 関数値を $\pi^2/6$ とする。
- ② $x > 1$ のとき, 関数値を $\pi^2/3 - \frac{1}{2}(\log x)^2 - Li_2(\frac{1}{x})$ とする。 $Li_2(\frac{1}{x})$ の値は以下によって求める。
- ③ $0 \leq x < 1$ のとき,
 - (1) もし, $x > \frac{1}{2}$ ならば, $Li_2(x) = \pi^2/6 - \log(1-x)\log x - Li_2(1-x)$ によって関数値を計算する。 $Li_2(1-x)$ は, (2) によって得られる。
 - (2) もし, $x \leq \frac{1}{2}$ ならば, $Li_2(x)$ は

$$Li_2(x) = \alpha - \frac{1}{4}\alpha^2 - \sum_{k=1}^8 \frac{c_{2k}}{2k+1} \alpha^{2k+1}$$

となる。ここで, $\alpha = -\log(1-x)$, c_{2k} はベルヌーイ数である。

この関係式から関数値 $Li_2(x)$ を求める。

2.1.2.18 デバイ関数

デバイ関数 $F_D(x)$ は, 以下で定義される関数である。

$$F_D(x) = \frac{3}{x^3} \int_0^x \frac{e^{t^4}}{(e^t - 1)^2} dt$$

これは,

$$F_D(x) = \frac{12}{x^3} \int_0^x \frac{t^3}{e^t - 1} dt - \frac{3x}{e^x - 1}$$

となる。この関数値は, x の範囲に応じ, 以下の 2 種類の方法を用いて計算される。

- ① $0 \leq x \leq \log 2$ のとき

$$F_D(x) = -\frac{3x}{e^x - 1} + 12\left(\frac{1}{3} - \frac{x}{8} - \sum_{k=1}^{\infty} \frac{c_{2k}}{2k+3} x^{2k}\right) = 1 + \sum_{k=1}^{\infty} \frac{3(2k-1)}{2k+3} c_{2k} x^{2k}$$

により求める。ここで, c_{2k} はベルヌーイ数である。 k についての級数和は, 加算項が誤差判定定数 (付録 B 参照) 以下になるまで計算する。

- ② $x > \log 2$ のとき

$$F_D(x) = \frac{12}{x^3} F(x) - \frac{3x}{e^x - 1},$$

$$F(x) = \frac{\pi^4}{15} + S_1(y) + S_2(y) \log y + S_3(y) (\log y)^2 + y (\log y)^3,$$

$$S_j(y) = \sum_{k=1}^{\infty} b_{k,j} y^k \quad (j = 1, 2, 3)$$

により求める。ここで、 $y = -\log(1 - e^{-x})$ であり、 k についての級数和は加算項が誤差判定定数 (付録 B 参照) 以下になるまで計算する。係数 $b_{k,j}$ は、以下の手順で求める。

$F(x) = \int_0^x t^3 dt / (e^t - 1)$ について、パラメータ $y (0 < y < \log 2)$ を用いると、

$$F(x) = \int_y^{\infty} (-\log(1 - e^{-u}))^3 du = \int_0^{\infty} - \int_0^y = I_1 - I_2$$

ここで、

$$I_1 = \int_0^{\infty} = F(\infty) = \frac{\pi^4}{15},$$

$$I_2 = \int_0^y = \int_0^y (\log(\frac{u}{1 - e^{-u}}) - \log u)^3 du.$$

さらに、被積分関数の含む $\log(\frac{u}{1 - e^{-u}})$ を級数展開

$$\log(\frac{u}{1 - e^{-u}}) = \frac{u}{2} + \sum_{k=1}^{\infty} \frac{c_{2k}}{2k} u^{2k}$$

で置き換え、べき級数と $\log u$ の低次べきの積和に変型し、項別積分の係数 $b_{k,j}$ が得られる。ここで、 c_{2k} はベルヌーイ数である。

2.1.2.19 正規化された球面調和関数

実数 $x (-1 \leq x \leq 1)$ に対して正規化された球面調和関数 (正規化されたルジャンドル陪関数) $P_n^{*m}(x)$

$$P_n^{*m}(x) = \frac{1}{4\pi i^m} A_{n,m} \int_{-\pi}^{\pi} (x + i\sqrt{1 - x^2} \cos \psi)^n \cos(m\psi) d\psi$$

を求める。ただし、 $i = \sqrt{-1}$,

$$A_{n,0} = \sqrt{\frac{2n+1}{\pi}}$$

$$A_{n,m} = \sqrt{\frac{2(2n+1)(n-m)!(n+m)!}{\pi(n!)^2}} \quad (1 \leq m \leq n)$$

である。

積分項を計算するために、以下のフーリエ級数展開を用いる ($x = \cos \theta, 0 \leq \theta \leq \pi$ とする):

$$(\cos \theta + i \sin \theta \cos \psi)^n = \sum_{m=0}^n i^m C_{n,m}(\theta) \cos(m\psi)$$

ここで、

$$I_{n,m} = \frac{1}{i^m \sin^m \theta} \int_{-\pi}^{\pi} (\cos \theta + i \sin \theta \cos \psi)^n \cos(m\psi) d\psi$$

とおくと、フーリエ展開係数 $C_{n,m}(\theta)$ は、以下の漸化式によって得られる:

$$I_{n,n} = \frac{\pi}{2^{n-1}}, \quad I_{n,n+1} = 0$$

$$I_{n,m-2} = \frac{2}{m-n-2} \left\{ -(m-1) \cos \theta I_{n,m-1} + (m+n) \frac{\sin^2 \theta}{2} I_{n,m} \right\}$$

$$C_{n,m}(\theta) = \frac{(2 - \delta_{m,0}) \sin^m \theta}{2\pi} I_{n,m}$$

上述の手順によって得られたフーリエ展開係数 $C_{n,m}(\theta)$ を用いると、次数 n の正規化された球面調和関数 (正規化されたルジャンドル陪関数) $P_n^{*m}(\cos \theta)$ は、以下ようになる。

$$P_n^{*m}(\cos \theta) = \frac{C_{n,m}(\theta)}{h_{n,m}}$$

ここで、係数 $h_{n,m}$ は

$$g_{n,0} = 1, g_{n,m} = g_{n,m-1} \left(1 + \frac{m}{n}\right)^{-1} \left(1 - \frac{m-1}{n}\right) \quad (m = 1, 2, \dots)$$

$$h_{n,m} = 2\sqrt{\frac{(2 - \delta_{m,0})\pi g_{n,m}}{2n+1}}$$

である。

2.1.2.20 実変数フルビッツゼータ関数

(1) 定義

フルビッツゼータ関数 $\zeta(s, a)$ は、実領域 $s > -1$ かつ $s \neq 1$ で、パラメータ $a > 0$ に対して、以下のように定義される関数である：

$$\zeta(s, a) = \frac{a^{-s+1}}{s-1} + \frac{a^{-s}}{2} + \int_0^\infty -s\psi(x)(x+a)^{-s-1} dx.$$

ここで、 $\psi(x)$ は x の小数部分から $1/2$ を引いて定義される x の周期関数 (周期 1) である。

(a) $s > 1$ のときの定義

フルビッツゼータ関数は以下ようになる：

$$\begin{aligned} \zeta(s, a) &= \int_0^\infty (x+a)^{-s} dx + \frac{a^{-s}}{2} + \int_0^\infty -s\psi(x)(x+a)^{-s-1} dx \\ &= \sum_{n=0}^\infty \left\{ \int_n^{n+1} (x+a)^{-s} dx + \int_n^{n+1} -s\psi(x)(x+a)^{-s-1} dx \right\} + \frac{a^{-s}}{2} \\ &= \sum_{n=0}^\infty \left\{ \int_n^{n+1} \frac{d}{dx} (\psi(x)(x+a)^{-s}) dx \right\} + \frac{a^{-s}}{2} \\ &= \sum_{n=0}^\infty \left\{ \frac{1}{2}(n+a)^{-s} + \frac{1}{2}(n+1+a)^{-s} \right\} + \frac{a^{-s}}{2} \\ &= \sum_{n=0}^\infty (n+a)^{-s}. \end{aligned}$$

したがって、 $s > 1$ について、フルビッツゼータ関数は有限値に収束する無限級数和 $\sum_{n=0}^\infty (n+a)^{-s}$ に一致する。

(b) $s > -1$ のときの定義

無限積分

$$\int_0^\infty -s\psi(x)(x+a)^{-s-1} dx$$

は、 $s > -1$ に対しても収束する。このことは、交代級数の部分

$$\sum_{m=0}^M \int_{\frac{m}{2}}^{\frac{m+1}{2}} -s\psi(x)(x+a)^{-s-1} dx$$

が $M \rightarrow \infty$ のとき収束することから明らかである。したがって、この範囲においてもフルビッツゼータ関数が定義される。

以上より、フルビッツゼータ関数

$$\zeta(s, a) = \frac{a^{-s+1}}{s-1} + \frac{a^{-s}}{2} + \int_0^\infty -s\psi(x)(x+a)^{-s-1}dx$$

は, $s > 1$ について定義された無限級数和 $\sum_{n=0}^\infty (n+a)^{-s}$ の定義域を拡張した関数である. また,

$$\zeta(s, a) - \frac{1}{s-1} = \int_0^1 \{\zeta(s, a) - \zeta(s, x+1)\}dx$$

は, $s = 1$ の場合も含め, $s > -1$ において成り立つ積分表示

$$\int_a^1 x^{-s}dx + \frac{a^{-s}}{2} + \int_0^\infty -s\psi(x)(x+a)^{-s-1}dx$$

をもつ.

(2) 計算のアルゴリズム

(a) 計算に用いる展開式

自然数 $N = 10$ として, これを固定する. フルビッツゼータ関数は, 以下のように展開できる.

$$\zeta(s, a) = \sum_{n=0}^{N-1} (n+a)^{-s} + Z(s, a) + \frac{(N+a)^{-s}}{2} + R(s, a)$$

ここに,

$$Z(s, a) = \frac{(N+a)^{-s+1}}{s-1},$$

$$R(s, a) = \int_N^\infty -s(x+a)^{-s-1}\psi(x)dx$$

である. この展開式は, フルビッツゼータ関数 $\zeta(s, a) = \sum_{n=0}^{N-1} (n+a)^{-s} + \sum_{n=N}^\infty (n+a)^{-s}$ の右辺第 2 和に, 積分表示を用いれば得られる.

(b) $R(s, a)$ の計算

$R(s, a)$ の値は, 次の近似式で計算する. (近似誤差: 単精度 10^{-9} , 倍精度 10^{-18})

$$R(s, a) = \sum_{j=0}^{m-2} (-s) \cdots (-s-j)(N+a)^{-s-1-j}\psi_{2+j}(0)$$

ここで, $\psi_k(x)$ は, 周期 1 の周期関数であり,

$$-\psi(x) = \psi'_2(x), \quad -\psi_2(x) = \psi'_3(x), \quad -\psi_3(x) = \psi'_4(x), \quad \cdots$$

$$\int_0^1 \psi_k(x)dx = 0$$

を満たすように定める.

$R(s, a)$ の手順は単精度と倍精度とでは異なる. また, s の値によっても異なる.

i. 単精度実数版のとき

$s > 10$ ならば $R(s, a) = 0$ とする.

$0 \leq s \leq 10$ ならば $m = 10$ として $R(s, a)$ の近似式を用いる.

ii. 倍精度実数版のとき

$s > 20$ ならば $R(s, a) = 0$ とする.

$10 < s \leq 20$ ならば $m = 10$ として $R(s, a)$ の近似式を用いる.

$5 < s \leq 10$ ならば $m = 15$ として $R(s, a)$ の近似式を用いる.

$0 \leq s \leq 5$ ならば $m = 20$ として $R(s, a)$ の近似式を用いる.

$\psi_k(0)$ はベルヌーイ数 B_l により

$$\psi_{2j+1}(0) = 0$$

$$\psi_{2j}(0) = (-1)^j \frac{B_j}{(2j)!}$$

と表され、任意の実数 x と t ($|t| < 2\pi$) に対して

$$\frac{te^{x_1 t}}{e^t - 1} = 1 - \sum_{k=1}^{\infty} \psi_k(x) (-t)^k$$

が成り立つ。ここに x_1 は x の小数部分である。ここで、ベルヌーイ多項式 $\psi_k(x)$ は

$$(-1)^{\frac{k+1}{2}} 2 \sum_{n=1}^{\infty} \frac{\sin(2\pi n x)}{(2\pi n)^k} \quad (k : \text{奇数})$$

$$(-1)^{\frac{k}{2}} 2 \sum_{n=1}^{\infty} \frac{\cos(2\pi n x)}{(2\pi n)^k} \quad (k : \text{偶数})$$

とフーリエ展開される。これより、絶対値が $(2\pi)^{-k}$ 程度であることがわかる。

(c) 特異点処理

フルビッツゼータ関数は $s = 1$ に極を持つので、 s が 1 に等しいときは関数値を求めることができない。しかし、 $\zeta(s, a) - 1/(s - 1)$ は整関数として定義できるので、フルビッツゼータ関数 $\zeta(s, a)$ の展開式の中の $Z(s, a)$ を $|s - 1| < 10^{-4}$ (単精度), $|s - 1| < 10^{-8}$ (倍精度) のときに限り、

$$\frac{(N + a)^{-s+1} - 1}{s - 1}$$

の近似式 (理論精度は少なくとも:単精度 10^{-9} , 倍精度 10^{-18})

$$-\log(N + a) \left(1 - \frac{1}{2} \log(N + a)(s - 1) \left(1 - \frac{1}{3} \log(N + a)(s - 1)\right)\right)$$

に置き換える。ここで、

$$\lim_{s \rightarrow 1} \left(\zeta(s, a) - \frac{1}{s - 1} \right) = -\frac{\Gamma'(a)}{\Gamma(a)}$$

である。

2.1.2.21 誤差関数に関連した関数

(1) 複素変数の誤差関数 $e^{-z^2} \operatorname{Erfc}(-iz)$

複素変数の誤差関数 $e^{-z^2} \operatorname{Erfc}(-iz)$ は、定積分

$$e^{-z^2} \int_0^z e^{w^2} dw$$

の値から求めることができる。

(a) $\Im(z)$ の絶対値が小さく 10^{-4} 以下のとき

$\Re(z)$ の特定の値 (0 もこれらの 1 つ) に対して分母が小さくなる可能性を避けるため、

$$f(z) = f_0 + f_1(z - z_0) + f_2(z - z_0)^2/2 + f_3(z - z_0)^3/6$$

で求める。ここに、 $f(z) = e^{-z^2} + \frac{2i}{\sqrt{\pi}} D_w(z)$ ($D_w(z)$ はドーソン積分) で、

$$f_0 = f(z_0), f_1 = -2z_0 f_0 + \frac{2i}{\sqrt{\pi}}, f_2 = -2z_0 f_1 - 2f_0, f_3 = -2z_0 f_2 - 4f_1, z_0 = \Re(z)$$

である。

(b) $\Im(z)$ が負のとき

$F(z) = e^{-z^2} \operatorname{Erfc}(-iz)$ に対して

$$F(z) + F(-z) = 2e^{-z^2}$$

であることを用いて、虚数部が正の場合に帰着させる。

(c) $\Im(z)$ が正のとき

$z = \beta = R + iI$ (R, I は実数で, $I > 0$) とする。

$$f(x) = \int_{-\infty}^{\infty} e^{-xt^2} / (t + \beta) dt = \int_0^{\infty} 2\beta e^{-xt^2} / (\beta^2 - t^2) dt \quad (x \geq 0)$$

とおく。 $x > 0$ のとき、

$$(e^{\beta^2 x} f(x))' = (\sqrt{\pi}\beta / \sqrt{x}) e^{\beta^2 x}$$

$$e^{\beta^2 x} f(x) - f(0) = \sqrt{\pi}\beta \int_0^x \frac{e^{\beta^2 t}}{\sqrt{t}} dt$$

これより、

$$\int_0^{\beta} e^{w^2} dw = \frac{1}{2\sqrt{\pi}} (e^{\beta^2} f(1) - f(0))$$

さらに、 $f(0) = \int_0^{\infty} 2\beta / (\beta^2 - t^2) dt$ において、積分路を $[0, \infty)$ から $[0, i\beta\infty)$ に換えて、留数を考慮すれば、

$$e^{-\beta^2} \int_0^{\beta} e^{w^2} dw = \frac{1}{2\sqrt{\pi}} \left(\int_{-\infty}^{\infty} e^{-t^2} / (t + \beta) dt + \pi i e^{-\beta^2} \right)$$

である。

次にポアソンの和公式を適用する。

$$g(x) = h \sum_{n=-\infty}^{\infty} \frac{e^{-h^2(n+x)^2}}{h(n+x) + \beta}$$

とおくと、 $g(x)$ は周期 1 の周期関数で、フーリエ級数に展開される:

$$\begin{aligned} \int_0^1 g(x) e^{-2\pi i n x} dx &= \int_{-\infty}^{\infty} h \frac{e^{-h^2 x^2}}{hx + \beta} e^{-2\pi i n x} dx \\ &= \int_{-\infty}^{\infty} \frac{e^{-x^2}}{x + \beta} e^{-2\pi i n x/h} dx \\ &= \int_{-\infty}^{\infty} \frac{e^{-(x+\pi i n/h)^2}}{x + \beta} dx e^{-\pi^2 n^2/h^2} \end{aligned}$$

ここで、留数を考慮すると、

$$\begin{aligned} &= e^{-\pi^2 n^2/h^2} \int_{-\infty}^{\infty} \frac{e^{-x^2}}{x + \beta - \pi i n/h} dx \quad (\pi n/h < I) \\ &= e^{-\pi^2 n^2/h^2} \left(\int_{-\infty}^{\infty} \frac{e^{-x^2}}{x + \beta - \pi i n/h} dx - 2\pi i e^{-(\beta + \pi i n/h)^2} \right) \quad (\pi n/h > I) \end{aligned}$$

となる。そこで、 $\pi/h > \sqrt{\log(10^{16})}$ とすれば、 10^{-16} のオーダーより小さい項を無視して、

$$h \sum_{n=-\infty}^{\infty} \frac{e^{-h^2 n^2}}{hn + \beta} = \int_{-\infty}^{\infty} \frac{e^{-x^2}}{x + \beta} dx - 2\pi i \sum_{\pi n/h > I} \exp(2\pi i n \beta/h - \beta^2)$$

が得られる。この両辺の無限和も少ない項で収束する。 $e^{-\beta^2} \operatorname{Erfc}(-i\beta)$ は

$$\frac{i}{\pi} \int_{-\infty}^{\infty} \frac{e^{-x^2}}{x + \beta} dx$$

である。

(2) 余誤差関数の逆関数

余誤差関数 $\operatorname{Erfc}(x)$ は、以下のように定義される。

$$y = \operatorname{Erfc}(x) = \frac{2}{\sqrt{\pi}} \int_x^{\infty} e^{-t^2} dt$$

ここで、 $y \geq 0$ である。

この逆関数 $y = \operatorname{Erfc}^{-1}(x)$ を求めるために、ニュートン法を用いて

$$f(y) = \left(\frac{2}{\sqrt{\pi}} \int_y^{\infty} e^{-t^2} dt - x \right) e^{y^2} = 0$$

を満たすような y を計算する。なお、この $f(y)$ の微分は、 $f'(y) = 2yf(y) - 2/\sqrt{\pi}$ である。 y の初期値は、 x の範囲によって以下のようにとる。

(a) $x \geq \operatorname{Erfc}(2)$ のとき

初期値 y_{init} は、 $y_{init} = 0$ とする。

(b) $x < \operatorname{Erfc}(2)$ のとき

初期値 y_{init} は、与えられた x に対して以下の式を用いて計算する。

$$y_1 = 1/(\sqrt{\pi}x), y_2 = \sqrt{\log y_1}, y_3 = 1/(\sqrt{\pi}xy_2), y_{init} = \sqrt{\log y_3}$$

この手順によって得られた初期値 y_{init} は、

$$\frac{\sqrt{\pi}}{2}x = e^{-y^2} \frac{1}{2y} (1 + O(\frac{1}{y^2}))$$

を満たす y の近似値である。

(3) 誤差関数 Erf , 余誤差関数 Erfc

(a) $|x| \leq 0.476563$ のとき

i. 余誤差関数 $\operatorname{Erfc}(x) = 1 - \operatorname{Erf}(x)$ とする。

ii. 誤差関数 $\operatorname{Erf}(x)$ は、以下による:

$$\operatorname{Erf}(x) = \frac{e^{-x^2}}{\sqrt{\pi}} \frac{2x}{1 - \frac{2x^2}{3 + \frac{4x^2}{5 - \frac{6x^2}{7 + \frac{8x^2}{9 - \dots}}}}}$$

(b) $|x| > 0.476563$ のとき

i. 余誤差関数 $\operatorname{Erfc}(x)$ は、以下の手順による。

$0 \leq x \leq 8.0$ のとき、最良近似式による。

$8.0 < x \leq 26.628736$ のとき

$$\operatorname{Erfc}(x) = \frac{e^{-x^2}}{\sqrt{\pi}} \frac{1.0}{x + \frac{0.5}{x + \frac{1.0}{x + \frac{1.5}{x + \dots}}}}$$

$x > 26.628736$ のとき、 $\operatorname{Erfc}(x) = 0$ とする。

x が負のときは、 $\operatorname{Erfc}(x) = 2 - \operatorname{Erfc}(-x)$ とする。

ii. 誤差関数 $\operatorname{Erf}(x) = 1 - \operatorname{Erfc}(x)$ とする。

2.1.2.22 係数算出法

本ライブラリのサブルーチンは、精度を落さずに計算量を減らす技法を採用している (文献 (18) 参照)。

2.1.2.23 関連した特殊関数の計算方法

(1) 第 3 種不完全楕円積分

$$\begin{aligned} \Pi(\varphi; c, m) &= \int_0^\varphi \frac{d\theta}{(1 + c \sin^2 \theta) \sqrt{1 - m \sin^2 \theta}} \\ &= \int_0^x \frac{dt}{(1 + c t^2) \sqrt{(1 - t^2)(1 - mt^2)}} \quad (x = \sin \varphi) \\ &= \int_0^u \frac{dw}{1 + c \cdot \operatorname{sn}^2 w} \quad (x = \operatorname{sn} u) \end{aligned}$$

これを Π 関数で表すと、次式になる。

$$\Pi(\varphi; c, m) = u + \frac{\operatorname{sn} \alpha}{\operatorname{cn} \alpha \cdot \operatorname{dn} \alpha} \cdot \Pi(u, \alpha)$$

ここで α は $\operatorname{sn}^2 \alpha = -\frac{c}{m}$ となる α とする。また $u = F(x, m) = F(\sin \varphi, m)$ である。(参考文献 (3), (17) 参照)

また、別の方法として

- $-m < c < 0$ のとき

$$\begin{aligned} \epsilon &= \sqrt{-\frac{c}{m}} \\ \beta &= \frac{\pi}{2} \cdot F(\epsilon, m) / K(m) \\ v &= \frac{\pi}{2} \cdot F(x, m) / K(m) \quad (x = \sin \varphi) \\ \delta_1 &= \sqrt{\frac{-c}{(1+c)(m+c)}} \end{aligned}$$

$$\Pi(\varphi; c, m) = \delta_1 \left[-\frac{1}{2} \log \frac{\vartheta_4(v + \beta)}{\vartheta_4(v - \beta)} + v \cdot \frac{\vartheta_1'(\beta)}{\vartheta_1(\beta)} \right]$$

ここで、 q を母数 m に対するノーム q として

$$\frac{1}{2} \log \frac{\vartheta_4(v + \beta)}{\vartheta_4(v - \beta)} = 2 \sum_{s=1}^{\infty} \frac{q^s}{s \cdot (1 - q^{2s})} \cdot \sin(2 \cdot s \cdot v) \cdot \sin(2 \cdot s \cdot \beta)$$

$$\frac{\vartheta_1'(\beta)}{\vartheta_1(\beta)} = \cot \beta + 4 \sum_{s=1}^{\infty} \frac{q^{2s}}{1 - 2 \cdot q^{2s} \cdot \cos(2 \cdot \beta) + q^{4s}} \cdot \sin(2 \cdot \beta)$$

Σ 計算は、末項が初項に対し十分小さくなるまで計算する。

- $c < -1$ のとき

$$N = \frac{m}{c}, p_1 = \sqrt{(-c - 1)(1 + \frac{m}{c})} \text{ として}$$

$$\Pi(\varphi; c, m) = -\pi(\varphi; N, m) + F(x, m) + \frac{1}{2p_1} \log \left[\frac{\Delta(\varphi) + p_1 \tan \varphi}{\Delta(\varphi) - p_1 \tan \varphi} \right]$$

$\Delta(\varphi) = \sqrt{1 - m \sin^2 \varphi}$, $\Pi(\varphi; N, m)$ は $-m < c < 0$ のときの計算を利用

- $-1 < c < -m$ のとき

$$\begin{aligned} \epsilon &= \sqrt{\frac{1+c}{1-m}} \\ \beta &= \frac{\pi}{2} \cdot F(\epsilon, 1-m) / K(m) \\ v &= \frac{\pi}{2} \cdot F(x, m) / K(m) \end{aligned}$$

$$\delta_2 = \sqrt{\frac{c}{(1+c)(m+c)}}$$

$$\lambda = a \tan(\tanh \beta \cdot \tan v) + 2 \sum_{s=1}^{\infty} \frac{(-1)^{s-1} \cdot q^{2s}}{s \cdot (1 - q^{2s})} \cdot \sin(2 \cdot s \cdot v) \cdot \sinh(2 \cdot s \cdot \beta)$$

$$\mu = \frac{\sum_{s=1}^{\infty} s q^{s^2} \sinh(2 \cdot s \cdot \beta)}{1 + 2 \sum_{s=1}^{\infty} q^{s^2} \cosh(2 \cdot s \cdot \beta)}$$

$$\Pi(\varphi; c, m) = \delta_2 \cdot (\lambda - 4 \cdot \mu \cdot v)$$

・ $c > 0.0$ のとき

$$N = -\frac{m+c}{1+c}$$

$$p_2 = \sqrt{\frac{c \cdot (m+c)}{1+c}}$$

$$\Pi(\varphi; c, m) = \left\{ \sqrt{(1+N) \cdot \left(1 + \frac{m}{N}\right)} \cdot \Pi(\varphi; N, m) + \frac{m \cdot F(x, m)}{p_2} \right. \\ \left. + a \tan\left(\frac{1}{2} p_2 \cdot \sin\left(\frac{2 \cdot \varphi}{\Delta(\varphi)}\right)\right) \right\} / \sqrt{(1+c) \left(1 + \frac{m}{c}\right)}$$

$\Delta(\varphi) = \sqrt{1 - m \sin^2 \varphi}$, $\Pi(\varphi; N, m)$ は $-m < c < 0.0$ のときの計算を利用 (参考文献 (1) 参照)

(2) ホイマンのラムダ関数

$$\Lambda_0(\varphi \setminus \alpha) = \frac{2}{\pi} \{K(\alpha)E(\varphi \setminus 90^\circ - \alpha) - (K(\alpha) - E(\alpha)) \cdot F(\varphi \setminus 90^\circ - \alpha)\}$$

このとき $m = \sin^2 \alpha$ とし

$$K(\alpha) = K(m)$$

$$E(\varphi \setminus 90^\circ - \alpha) = E(\sin \varphi, 1 - m)$$

$$E(\alpha) = E(m)$$

$$F(\varphi \setminus 90^\circ - \alpha) = F(\sin \varphi, 1 - m)$$

として求める。ここで K, E, F は楕円積分である。(参考文献 (1) 参照)

(3) ルジャンドル関数

$x > 1.0$ のとき

$$P_{-1/2}(x) = \frac{2}{\pi} \sqrt{\frac{2}{x+1}} K\left(\frac{x-1}{x+1}\right)$$

$$P_{1/2}(x) = \frac{2}{\pi} \sqrt{x - \sqrt{x^2 - 1}} E\left(\frac{2\sqrt{x^2 - 1}}{x + \sqrt{x^2 - 1}}\right)$$

$$Q_{-1/2}(x) = \sqrt{\frac{2}{x+1}} K\left(\frac{2}{x+1}\right)$$

$$Q_{1/2}(x) = x \sqrt{\frac{2}{x+1}} K\left(\frac{2}{x+1}\right) - \sqrt{2(x+1)} E\left(\frac{2}{x+1}\right)$$

$x \leq 1.0$ のとき

$$P_{-1/2}(x) = \frac{2}{\pi} K\left(\frac{1-x}{2}\right)$$

$$P_{1/2}(x) = \frac{2}{\pi} \left\{ 2E\left(\frac{1-x}{2}\right) - K\left(\frac{1-x}{2}\right) \right\}$$

$$Q_{-1/2}(x) = K\left(\frac{1+x}{2}\right)$$

$$Q_{1/2}(x) = K\left(\frac{1+x}{2}\right) - 2E\left(\frac{1+x}{2}\right)$$

ここで E, K は完全楕円積分である。

2.1.3 参考文献

- (1) Abramowitz, M. and Stegun, I. A. 編, “Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables”, Dover Publications, Inc. (1965).
- (2) Hart, J. F. 編, “Computer approximation”, John Wiley and Sons (1968).
- (3) 山内二郎, 宇野利雄, 一松信編, “電子計算機のための数値計算法 II, III”, 培風館 (1967, 1972).
- (4) 森口繁一, 宇田川 久, 一松信, “数学公式 III—特殊函数—”, 岩波全書, (1960).
- (5) 吉田年雄, 二宮市三, “ x が小さい場合のベッセル関数 $Y_\nu(x)$ の数値計算”, 情報処理学会論文誌, Vol. 23, No. 3, pp. 269–303 (1982).
- (6) 吉田年雄他, “漸化式を用いる複素変数のベッセル関数 $I_n(z)$ の数値計算”, 情報処理, Vol. 14, No. 1, pp. 23–29 (1973).
- (7) 吉田年雄, “複素変数のベッセル関数副プログラムについて – $I_n(z)$ 及び $J_n(z)$ –”, 名古屋大学大型計算機センターニュース, Vol. 5, No. 3, pp. 179–185 (1974).
- (8) 吉田年雄, 二宮市三, “ x が小さい場合の変形ベッセル関数 $K_\nu(x)$ の数値計算”, 情報処理学会論文誌, Vol. 21, No. 3, pp. 238–245 (1980).
- (9) 吉田年雄, 二宮市三, “ τ –method による複素変数のベッセル関数 $K_n(z)$ の数値計算”, 情報処理, Vol. 14, No. 8, pp. 569–575 (1973).
- (10) 渡部力, 名取亮, 小国力編, “第3版数値解析とFORTRAN”, 丸善, (1983).
- (11) 井阪秀高, “連分数と漸化式を用いるベッセル関数 $J_\nu(x), I_\nu(x)$ の近似計算”, 情報処理学会全国大会講演論文集, Vol. 35, pp. 11–12 (1987).
- (12) Cody W. J. , Strecok A. J. and Thacher H. C. , “Chebyshev approximations for the PSI function”, Math. Comp. , Vol. 27, No. 121, pp. 123–127 (1973).
- (13) Cody W. J. and Thacher H. C. , “Rational Chebyshev approximations for the exponential integral $E_1(x)$ ”, Math. Comp. , Vol. 22, pp. 641–649 (1968).
- (14) Cody W. J. and Thacher H. C. , “Rational Chebyshev approximations for the exponential integral $E_1(x)$ ”, Math. Comp. , Vol. 23, pp. 289–303 (1969).
- (15) Cody W. J. , Paciorek K. A. and Thacher H. C. Jr. , “Chebyshev approximations for Dawson’s integral”, Math. Comp. , Vol. 24, pp. 171–178 (1970).
- (16) 多谷虎男, “ベッセル関数と弾性波動理論”, 山海堂, (1986).
- (17) 安藤四郎, “楕円積分, 楕円関数入門”, 日新出版, (1970).
- (18) 浜田穂積, “最良近似システム”, 情報処理学会研究報告, 数値解析研究会資料, 6, (1983).
- (19) 井阪秀高, “ x が中間領域の場合のベッセル関数の近似計算”, 情報処理学会全国大会講演論文集, Vol. 37, pp. 47–48 (1988).
- (20) 吉田年雄, 二宮市三, “ベッセル関数 $K_\nu(x)$ の計算法”, 情報処理学会全国大会講演論文集, Vol. 20, pp. 427–428 (1979).

- (21) Luke Y. L. , “The special functions and their approximations, II”, Academic Press, (1969).
- (22) 吉田年雄, 二宮市三, “ x が小さい場合の不完全ガンマ関数 $\Gamma(v, x)$ の数値計算”, 情報処理学会論文誌, Vol. 23, No. 5, pp. 522–528 (1982).

2.2 ベッセル関数

2.2.1 WIBJ0X, VIBJ0X

第1種0次ベッセル関数

(1) 機能

$x = X_i$ に対して0次の第1種ベッセル関数

$$J_0(x) = \frac{1}{\pi} \int_0^\pi \cos(x \sin(t)) dt$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL WIBJ0X (NV, XI, XO, IERR)

単精度サブルーチン:

CALL VIBJ0X (NV, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力データ数
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	入 力	変数値 X_i
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	出 力	$J_0(X_i)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(b) $|XI(i)| \leq M$

ここで, $M = \{ \text{倍精度: } 2^{50}\pi, \text{単精度: } 2^{18}\pi \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
3000+i	XI(i) が制限条件 (b) を満足しなかった.	

(6) 注意事項

- (a) 第1種
- ν
- 次ベッセル関数
- $J_\nu(z)$
- はベッセルの微分方程式

$$z^2 \frac{d^2 w}{dz^2} + z \frac{dw}{dz} + (z^2 - \nu^2)w = 0$$

の基本解であり、

$$J_\nu(z) = \left(\frac{z}{2}\right)^\nu \sum_{m=0}^{\infty} \frac{(-1)^m}{m! \Gamma(m + \nu + 1)} \left(\frac{z}{2}\right)^{2m}$$

で定義される。

- (b) 第1種ベッセル関数は第1種円柱関数とも呼ばれる。

(7) 使用例

- (a) 問題

$J_0(x)$ の値を $x = 0.0, 0.1, 0.2, \dots, 0.9$ について求める。

- (b) 主プログラム

```

PROGRAM EIBJOX
IMPLICIT REAL(8)(A-H,O-Z)
PARAMETER (NV=10)
REAL(8) XI(NV), XO(NV)
CHARACTER*6 CNAME, CFNC
PARAMETER( CNAME='WIBJOX', CFNC=' JO' )
!
DNV=NV
DO 1000 I=1,NV
  XI(I)=(I-1)/DNV
1000 CONTINUE
!
CALL WIBJOX( NV, XI, XO, IERR )
!
WRITE(6,6000) CNAME
WRITE(6,6100)
DO 2000 I=1,NV
  WRITE(6,6200) I,XI(I)
2000 CONTINUE
!
WRITE(6,6300)
WRITE(6,6400) IERR
DO 3000 I=1,NV
  WRITE(6,6500) CFNC,XI(I), XO(I)
3000 CONTINUE
STOP
!
6000 FORMAT(1X,'*** ',A6,' *')
6100 FORMAT(1X,'*** INPUT *')
6200 FORMAT(1X,'XI(',I2,')=',F10.6 )
6300 FORMAT(1X,'*** OUTPUT *')
6400 FORMAT(1X,'IERR=',I5 )
6500 FORMAT(1X,A6,'(',F10.6,')=',F10.6 )
END

```

- (c) 出力結果

```

*** WIBJOX *
*** INPUT *
XI( 1)= 0.000000
XI( 2)= 0.100000
XI( 3)= 0.200000
XI( 4)= 0.300000
XI( 5)= 0.400000
XI( 6)= 0.500000
XI( 7)= 0.600000
XI( 8)= 0.700000
XI( 9)= 0.800000
XI(10)= 0.900000
*** OUTPUT *
IERR= 0
JO( 0.000000)= 1.000000
JO( 0.100000)= 0.997502
JO( 0.200000)= 0.990025
JO( 0.300000)= 0.977626
JO( 0.400000)= 0.960398
JO( 0.500000)= 0.938470
JO( 0.600000)= 0.912005
JO( 0.700000)= 0.881201
JO( 0.800000)= 0.846287
JO( 0.900000)= 0.807524

```

2.2.2 WIBY0X, VIBY0X 第2種0次ベッセル関数

(1) 機能

$x = X_i$ に対して0次の第2種ベッセル関数

$$Y_0(x) = -\frac{2}{\pi} \int_0^\infty \cos(x \cosh(t)) dt \quad (x > 0.0)$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL WIBY0X (NV, XI, XO, IERR)

単精度サブルーチン:

CALL VIBY0X (NV, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力データ数
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	入 力	変数値 X_i
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	出 力	$Y_0(X_i)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(b) $XI(i) \geq 0.0$

(c) $XI(i) \leq M$

ここで, $M = \{ \text{倍精度: } 2^{50}\pi, \text{単精度: } 2^{18}\pi \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
2000	$XI(i) = 0.0$ (overflow)	$XO(i) = (\text{最小値})$ を返す.
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
3000+i	$XI(i)$ が制限条件 (b) または (c) を満足しなかった.	

(6) 注意事項

- (a) 第 2 種
- ν
- 次ベッセル関数
- $Y_\nu(z)$
- はベッセルの微分方程式

$$z^2 \frac{d^2 w}{dz^2} + z \frac{dw}{dz} + (z^2 - \nu^2)w = 0$$

の基本解であり,

$$Y_\nu(z) = \frac{J_\nu(z) \cos \nu\pi - J_{-\nu}(z)}{\sin \nu\pi}$$

で定義される. ただし, ν が整数 n に等しい場合は,

$$Y_n(z) = \lim_{\nu \rightarrow n} Y_\nu(z)$$

とする.

- (b) 第 2 種ベッセル関数は第 2 種円柱関数とも呼ばれる.
 (c) ノイマン関数 $N_\nu(z)$ は, 第 2 種ベッセル関数 $Y_\nu(z)$ と同じものである.

(7) 使用例

- (a) 問題

$Y_0(x)$ の値を $x = 0.1, 0.2, \dots, 1.0$ について求める.

- (b) 主プログラム

```

PROGRAM EIBY0X
IMPLICIT REAL(8)(A-H,O-Z)
PARAMETER (NV=10)
REAL(8) XI(NV), XO(NV)
CHARACTER*6 CNAME, CFNC
PARAMETER( CNAME='WIBY0X', CFNC=' YO' )
!
DNV=NV
DO 1000 I=1,NV
  XI(I)=I/DNV
1000 CONTINUE
!
CALL WIBY0X( NV, XI, XO, IERR )
!
WRITE(6,6000) CNAME
WRITE(6,6100)
DO 2000 I=1,NV
  WRITE(6,6200) I,XI(I)
2000 CONTINUE
!
WRITE(6,6300)
WRITE(6,6400) IERR
DO 3000 I=1,NV
  WRITE(6,6500) CFNC,XI(I), XO(I)
3000 CONTINUE
STOP
!
6000 FORMAT(1X,'*** ',A6,' *')
6100 FORMAT(1X,'*** INPUT *')
6200 FORMAT(1X,'XI(',I2,')=',F10.6 )
6300 FORMAT(1X,'*** OUTPUT *')
6400 FORMAT(1X,'IERR=',I5 )
6500 FORMAT(1X,A6,'(',F10.6,')=',F10.6 )
END

```

- (c) 出力結果

```

*** WIBY0X *
*** INPUT *
XI( 1)= 0.100000
XI( 2)= 0.200000
XI( 3)= 0.300000
XI( 4)= 0.400000
XI( 5)= 0.500000
XI( 6)= 0.600000
XI( 7)= 0.700000
XI( 8)= 0.800000
XI( 9)= 0.900000
XI(10)= 1.000000
*** OUTPUT *
IERR= 0
YO( 0.100000)= -1.534239
YO( 0.200000)= -1.081105
YO( 0.300000)= -0.807274
YO( 0.400000)= -0.606025
YO( 0.500000)= -0.444519
YO( 0.600000)= -0.308510
YO( 0.700000)= -0.190665
YO( 0.800000)= -0.086802
YO( 0.900000)= 0.005628
YO( 1.000000)= 0.088257

```

2.2.3 WIBJ1X, VIBJ1X 第1種1次ベッセル関数

(1) 機能

$x = X_i$ に対して1次の第1種ベッセル関数

$$J_1(x) = \frac{1}{\pi} \int_0^\pi \cos(x \sin(t) - t) dt$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL WIBJ1X (NV, XI, XO, IERR)

単精度サブルーチン:

CALL VIBJ1X (NV, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力データ数
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	入 力	変数値 X_i
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	出 力	$J_1(X_i)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(b) $|XI(i)| \leq M$

ここで, $M = \{ \text{倍精度: } 2^{50}\pi, \text{ 単精度: } 2^{18}\pi \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
3000+i	XI(i) が制限条件 (b) を満足しなかった.	

(6) 注意事項

- (a) 第 1 種
- ν
- 次ベッセル関数
- $J_\nu(z)$
- はベッセルの微分方程式

$$z^2 \frac{d^2 w}{dz^2} + z \frac{dw}{dz} + (z^2 - \nu^2)w = 0$$

の基本解であり,

$$J_\nu(z) = \left(\frac{z}{2}\right)^\nu \sum_{m=0}^{\infty} \frac{(-1)^m}{m! \Gamma(m + \nu + 1)} \left(\frac{z}{2}\right)^{2m}$$

で定義される.

- (b) 第 1 種ベッセル関数は第 1 種円柱関数とも呼ばれる.

(7) 使用例

- (a) 問題

$J_1(x)$ の値を $x = 0.0, 0.1, 0.2, \dots, 0.9$ について求める.

- (b) 主プログラム

```

PROGRAM EIBJ1X
IMPLICIT REAL(8)(A-H,O-Z)
PARAMETER (NV=10)
REAL(8) XI(NV), XO(NV)
CHARACTER*6 CNAME, CFNC
PARAMETER( CNAME='WIBJ1X', CFNC=' J1' )
!
DNV=NV
DO 1000 I=1,NV
  XI(I)=(I-1)/DNV
1000 CONTINUE
!
CALL WIBJ1X( NV, XI, XO, IERR )
!
WRITE(6,6000) CNAME
WRITE(6,6100)
DO 2000 I=1,NV
  WRITE(6,6200) I,XI(I)
2000 CONTINUE
!
WRITE(6,6300)
WRITE(6,6400) IERR
DO 3000 I=1,NV
  WRITE(6,6500) CFNC,XI(I), XO(I)
3000 CONTINUE
STOP
!
6000 FORMAT(1X,'*** ',A6,' *')
6100 FORMAT(1X,'*** INPUT *')
6200 FORMAT(1X,'XI(',I2,')=',F10.6 )
6300 FORMAT(1X,'*** OUTPUT *')
6400 FORMAT(1X,'IERR=',I5 )
6500 FORMAT(1X,A6,'(',F10.6,')=',F10.6 )
END

```

- (c) 出力結果

```

*** WIBJ1X *
*** INPUT *
XI( 1)= 0.000000
XI( 2)= 0.100000
XI( 3)= 0.200000
XI( 4)= 0.300000
XI( 5)= 0.400000
XI( 6)= 0.500000
XI( 7)= 0.600000
XI( 8)= 0.700000
XI( 9)= 0.800000
XI(10)= 0.900000
*** OUTPUT *
IERR= 0
J1( 0.000000)= 0.000000
J1( 0.100000)= 0.049938
J1( 0.200000)= 0.099501
J1( 0.300000)= 0.148319
J1( 0.400000)= 0.196027
J1( 0.500000)= 0.242268
J1( 0.600000)= 0.286701
J1( 0.700000)= 0.328996
J1( 0.800000)= 0.368842
J1( 0.900000)= 0.405950

```

2.2.4 WIBY1X, VIBY1X 第2種1次ベッセル関数

(1) 機能

$x = X_i$ に対して1次の第2種ベッセル関数

$$Y_1(x) = \frac{1}{\pi} \int_0^\pi \sin(x \sin(t) - t) dt - \frac{1}{\pi} \int_0^\infty e^{-x \sinh(t)} [e^t - e^{-t}] dt$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL WIBY1X (NV, XI, XO, IERR)

単精度サブルーチン:

CALL VIBY1X (NV, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力データ数
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	入 力	変数値 X_i
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	出 力	$Y_1(X_i)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(b) $XI(i) \geq 0.0$

(c) $XI(i) \leq M$

ここで, $M = \{ \text{倍精度: } 2^{50}\pi, \text{単精度: } 2^{18}\pi \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
2000+i	$XI(i) \leq 1.0/(\text{最大値})$ (overflow)	XO(i) =(最小値) を返す.
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
3000+i	XI(i) が制限条件 (b) または (c) を満足しなかった.	

(6) 注意事項

- (a) 第 2 種
- ν
- 次ベッセル関数
- $Y_\nu(z)$
- はベッセルの微分方程式

$$z^2 \frac{d^2 w}{dz^2} + z \frac{dw}{dz} + (z^2 - \nu^2)w = 0$$

の基本解であり,

$$Y_\nu(z) = \frac{J_\nu(z) \cos \nu\pi - J_{-\nu}(z)}{\sin \nu\pi}$$

で定義される. ただし, ν が整数 n に等しい場合は,

$$Y_n(z) = \lim_{\nu \rightarrow n} Y_\nu(z)$$

とする.

- (b) 第 2 種ベッセル関数は第 2 種円柱関数とも呼ばれる.
 (c) ノイマン関数 $N_\nu(z)$ は, 第 2 種ベッセル関数 $Y_\nu(z)$ と同じものである.

(7) 使用例

- (a) 問題

$Y_1(x)$ の値を $x = 0.1, 0.2, \dots, 1.0$ について求める.

- (b) 主プログラム

```

PROGRAM EIBY1X
IMPLICIT REAL(8)(A-H,O-Z)
PARAMETER (NV=10)
REAL(8) XI(NV), XO(NV)
CHARACTER*6 CNAME, CFNC
PARAMETER( CNAME='WIBY1X', CFNC=' Y1' )
!
DNV=NV
DO 1000 I=1,NV
  XI(I)=I/DNV
1000 CONTINUE
!
CALL WIBY1X( NV, XI, XO, IERR )
!
WRITE(6,6000) CNAME
WRITE(6,6100)
DO 2000 I=1,NV
  WRITE(6,6200) I,XI(I)
2000 CONTINUE
!
WRITE(6,6300)
WRITE(6,6400) IERR
DO 3000 I=1,NV
  WRITE(6,6500) CFNC,XI(I), XO(I)
3000 CONTINUE
STOP
!
6000 FORMAT(1X,'*** ',A6,' *')
6100 FORMAT(1X,'*** INPUT *')
6200 FORMAT(1X,'XI(',I2,')=',F10.6 )
6300 FORMAT(1X,'*** OUTPUT *')
6400 FORMAT(1X,'IERR=',I5 )
6500 FORMAT(1X,A6,'(',F10.6,')=',F10.6 )
END

```

- (c) 出力結果

```

*** WIBY1X *
*** INPUT *
XI( 1)= 0.100000
XI( 2)= 0.200000
XI( 3)= 0.300000
XI( 4)= 0.400000
XI( 5)= 0.500000
XI( 6)= 0.600000
XI( 7)= 0.700000
XI( 8)= 0.800000
XI( 9)= 0.900000
XI(10)= 1.000000
*** OUTPUT *
IERR= 0
Y1( 0.100000)= -6.458951
Y1( 0.200000)= -3.323825
Y1( 0.300000)= -2.293105
Y1( 0.400000)= -1.780872
Y1( 0.500000)= -1.471472
Y1( 0.600000)= -1.260391
Y1( 0.700000)= -1.103250
Y1( 0.800000)= -0.978144
Y1( 0.900000)= -0.873127
Y1( 1.000000)= -0.781213

```


2.2.5 DIBJNX, RIBJNX

第 1 種整数次ベッセル関数

(1) 機能

整数次の第 1 種ベッセル関数

$$J_n(x) = \frac{1}{\pi} \int_0^\pi \cos(x \sin(t) - nt) dt$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL DIBJNX (N, XI, XO, IERR)

単精度サブルーチン:

CALL RIBJNX (N, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER(4)} \\ 64 \text{ ビット整数版では INTEGER(8)} \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入 力	次数 n
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 x
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	$J_n(x)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $|XI| \leq M$

ここで, $M = \{ \text{倍精度: } 2^{50}\pi, \text{単精度: } 2^{18}\pi \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1000	$ n (\log_e \frac{x}{N} - M_1) > M_2$ であった. (注意事項 (c) 参照) ($XI \neq 0.0, N \neq 0$)(underflow)	XO = 0.0 を返す.
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.

(6) 注意事項

(a) $J_n(x)$ は, x や n が大きいほど計算時間がかかる. 一般に $|N| < 1000, |XI| < 1000.0$ とすることが望ましい.

(b) $J_n(x), J_{n+1}(x), J_{n+2}(x) \dots$ を同時に求める場合は, 漸化式を用いて順次求めた方がこのサブルーチンを幾度も呼ぶよりも速い.

ただし, n を増大させていくと不安定になるので, n が減少する方向へ適用する.

漸化式:

$$J_{n-1}(x) = \frac{2n}{x} J_n(x) - J_{n+1}(x)$$

(c) このサブルーチンで IERR=1000 となるときの M_1, M_2 の値は以下の通りである.

$$M_1 = 0.3068, M_2 = \{ \text{倍精度: } 709.7827, \text{単精度: } 88.72284 \}$$

(d) 第1種 ν 次ベッセル関数 $J_\nu(z)$ はベッセルの微分方程式

$$z^2 \frac{d^2 w}{dz^2} + z \frac{dw}{dz} + (z^2 - \nu^2) w = 0$$

の基本解であり,

$$J_\nu(z) = \left(\frac{z}{2}\right)^\nu \sum_{m=0}^{\infty} \frac{(-1)^m}{m! \Gamma(m + \nu + 1)} \left(\frac{z}{2}\right)^{2m}$$

で定義される.

(e) 第1種ベッセル関数は第1種円柱関数とも呼ばれる.

(7) 使用例

(a) 問題

$J_n(x)$ の値を $n = 5, x = 1.5$ について求める.

(b) 入力データ

N = 5, XI = 1.5

(c) 主プログラム

```
PROGRAM BIBJNX
! *** EXAMPLE OF DIBJNX ***
IMPLICIT REAL(8) (A-H,O-Z)
READ (5,*) N
READ (5,*) XI
WRITE(6,1000) N,XI
CALL DIBJNX(N,XI,XO,IERR)
WRITE(6,2000) IERR,XO
1000 FORMAT(' ',/,/,5X,'*** DIBJNX ***',/,/,6X,'** INPUT **',&
/,/,8X,'N = ',I3,5X,'XI = ',F6.2 )
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF JN(X)',/,/,10X,'XO = ',D18.10)
END
```

(d) 出力結果

```
*** DIBJNX ***
** INPUT **
N = 5    XI = 1.50

** OUTPUT**
IERR = 0
VALUE OF JN(X)
XO = 0.1799421767D-02
```

2.2.6 DIBYNX, RIBYNX 第2種整数次ベッセル関数

(1) 機能

整数次の第2種ベッセル関数

$$Y_n(x) = \frac{1}{\pi} \int_0^\pi \sin(x \sin(t) - nt) dt - \frac{1}{\pi} \int_0^\infty e^{-x \sinh(t)} [e^{nt} + (-1)^n e^{-nt}] dt$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL DIBYNX (N, XI, XO, IERR)

単精度サブルーチン:

CALL RIBYNX (N, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER(4)} \\ 64 \text{ ビット整数版では INTEGER(8)} \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入 力	次数 n
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 x
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	$Y_n(x)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $XI \geq 0.0$

(b) $XI \leq M$

ここで, $M = \{ \text{倍精度: } 2^{50}\pi, \text{単精度: } 2^{18}\pi \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
2000	$XI \leq 2.0/$ (最大値) または $ n (\log_e \frac{ n }{x} - M_1) > M_2$ (注意事項 (c) 参照) ($XI \neq 0.0, N \neq 0$) (overflow)	$N \geq 0$ のとき $XO =$ (最小値) $N < 0$ のとき $XO =$ (最小値) $\times (-1)^N$ を返す.
3000	制限条件 (a) または (b) を満足しなかった.	処理を打ち切る.

(6) 注意事項

- (a) $Y_n(x)$ は, x や n が大きいほど計算時間がかかる. 一般に $|N| < 1000$, $XI < 1000.0$ とすることが望ましい.
- (b) $Y_n(x), Y_{n+1}(x), Y_{n+2}(x) \dots$ を同時に求める場合は, 漸化式を用いて順次求めた方が, このサブルーチンを幾度も使用するよりも速い.

漸化式:

$$Y_{n+1}(x) = \frac{2n}{x} Y_n(x) - Y_{n-1}(x)$$

- (c) このサブルーチンで IERR=2000 となるときの M_1, M_2 の値は以下の通りである.

$$M_1 = 0.3068, M_2 = \{ \text{倍精度: } 709.7827, \text{単精度: } 88.72284 \}$$

- (d) 第 2 種 ν 次ベッセル関数 $Y_\nu(z)$ はベッセルの微分方程式

$$z^2 \frac{d^2 w}{dz^2} + z \frac{dw}{dz} + (z^2 - \nu^2) w = 0$$

の基本解であり,

$$Y_\nu(z) = \frac{J_\nu(z) \cos \nu\pi - J_{-\nu}(z)}{\sin \nu\pi}$$

で定義される. ただし, ν が整数 n に等しい場合は,

$$Y_n(z) = \lim_{\nu \rightarrow n} Y_\nu(z)$$

とする.

- (e) 第 2 種ベッセル関数は第 2 種円柱関数とも呼ばれる.
- (f) ノイマン関数 $N_\nu(z)$ は, 第 2 種ベッセル関数 $Y_\nu(z)$ と同じものである.

(7) 使用例

- (a) 問題

$Y_n(x)$ の値を $n = 5$, $x = 1.5$ について求める.

- (b) 入力データ

$N = 5$, $XI = 1.5$

- (c) 主プログラム

```

PROGRAM BIBYNX
! *** EXAMPLE OF DIBYNX ***
IMPLICIT REAL(8) (A-H,O-Z)
READ (5,*) N
READ (5,*) XI
WRITE (6,1000) N,XI
CALL DIBYNX(N,XI,X0,IERR)
WRITE (6,2000) IERR,X0
1000 FORMAT(' ',/,/,5X,'*** DIBYNX ***',/,/,6X,'** INPUT **',&
/,/,8X,'N =',I3,5X,'XI =',F6.2)
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF YN(X)',/,/,10X,'X0 = ',D18.10)
END

```

- (d) 出力結果

```

*** DIBYNX ***
** INPUT **
N = 5      XI = 1.50

** OUTPUT**
IERR = 0
VALUE OF YN(X)
X0 = -0.3719030840D+02

```

2.2.7 DIBJMX, RIBJMX 第 1 種実数次ベッセル関数

(1) 機能

実数次の第 1 種ベッセル関数

$$J_\nu(x) = \frac{1}{\pi} \int_0^\pi \cos(x \sin(t) - \nu t) dt - \frac{\sin(\pi\nu)}{\pi} \int_0^\infty e^{-x \sinh(t)} e^{-\nu t} dt$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL DIBJMX (R, XI, XO, IERR)

単精度サブルーチン:

CALL RIBJMX (R, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER(4)} \\ 64 \text{ ビット整数版では INTEGER(8)} \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	R	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	次数 ν
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 x
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	$J_\nu(x)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) R が整数値と一致するとき

$$|R| \leq M_1$$

$$|XI| \leq M_2$$

(b) R が整数値と一致しないとき

$$0 < R \leq M_1$$

$$0 < XI \leq M_2$$

ここで, $M_1 = \{ \text{倍精度: } 2^{31}, \text{ 単精度: } 2^{31} \}$,

$M_2 = \{ \text{倍精度: } 2^{50}\pi, \text{ 単精度: } 2^{18}\pi \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1000	$\nu(\log_e \frac{x}{e} - M_3) > M_4$ であった. (注意事項 (e) 参照) ($XI \neq 0.0, R \neq 0.0$) (underflow) 注 ν が整数値と一致するときは $ \nu $, $ x $ に対して判定	XO = 0.0 を返す.
3000	制限条件 (a) または (b) を満足しなかった.	処理を打ち切る.

(6) 注意事項

(a) $J_\nu(x)$ は, x や ν が大きいほど計算時間がかかる. 一般に $R < 1000.0, XI < 1000.0$ とすることが望ましい.

(b) 半奇数次の場合は, 球ベッセル関数を利用する方がよい.

$$J_{n+\frac{1}{2}}(x) = \sqrt{\frac{2x}{\pi}} j_n(x)$$

(c) $\nu < 0$ で ν が整数値と一致しないときは, このサブルーチンでは計算ができない. したがって漸化式を利用して求める.

(d) $J_\nu(x), J_{\nu+1}(x), J_{\nu+2}(x) \dots$ を同時に求める場合は, 漸化式を用いて順次求めた方が, このサブルーチンを幾度も使用するよりも速い. ただし, ν を増大させていくと不安定となるので, ν が減少する方向へ適用する.

漸化式:

$$J_{\nu-1}(x) = \frac{2\nu}{x} J_\nu(x) - J_{\nu+1}(x)$$

(e) このサブルーチンで IERR=1000 となるときの M_3, M_4 の値は以下の通りである.

$$M_3 = 0.3068, M_4 = \{ \text{倍精度: } 709.7827, \text{単精度: } 88.72284 \}$$

(f) 第 1 種 ν 次ベッセル関数 $J_\nu(z)$ はベッセルの微分方程式

$$z^2 \frac{d^2 w}{dz^2} + z \frac{dw}{dz} + (z^2 - \nu^2)w = 0$$

の基本解であり,

$$J_\nu(z) = \left(\frac{z}{2}\right)^\nu \sum_{m=0}^{\infty} \frac{(-1)^m}{m! \Gamma(m + \nu + 1)} \left(\frac{z}{2}\right)^{2m}$$

で定義される.

(g) 第 1 種ベッセル関数は第 1 種円柱関数とも呼ばれる.

(7) 使用例

(a) 問題

$J_\nu(x)$ の値を $\nu = 3.3, x = 1.5$ について求める。

(b) 入力データ

R = 3.3, XI = 1.5

(c) 主プログラム

```
PROGRAM BIBJMX
! *** EXAMPLE OF DIBJMX ***
IMPLICIT REAL(8) (A-H,O-Z)
READ (5,*) R
READ (5,*) XI
WRITE (6,1000) R,XI
CALL DIBJMX(R,XI,XO,IERR)
WRITE (6,2000) IERR,XO
1000 FORMAT(' ',/,/,5X,'*** DIBJMX ***',/,/,6X,'** INPUT **',&
/,/,8X,'R = ',F6.2,5X,'XI = ',F6.2)
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF JM(X)',/,/,10X,'XO = ',D18.10)
END
```

(d) 出力結果

```
*** DIBJMX ***
** INPUT **
R = 3.30    XI = 1.50

** OUTPUT**
IERR =      0
VALUE OF JM(X)
XO = 0.3827927999D-01
```

2.2.8 DIBYMX, RIBYMX

第 2 種実数次ベッセル関数

(1) 機能

実数次の第 2 種ベッセル関数

$$Y_\nu(x) = \frac{1}{\pi} \int_0^\pi \sin(x \sin(t) - \nu t) dt - \frac{1}{\pi} \int_0^\infty e^{-x \sinh(t)} [e^{\nu t} + \cos(\pi \nu) e^{-\nu t}] dt$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL DIBYMX (R, XI, XO, IERR)

単精度サブルーチン:

CALL RIBYMX (R, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	R	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	次数 ν
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 x
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	$Y_\nu(x)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) R が整数値と一致するとき

$$|R| \leq M_1$$

(b) R が整数値と一致しないとき

$$0 < R \leq M_1$$

ここで, $M_1 = \{ \text{倍精度: } 2^{31}, \text{単精度: } 2^{31} \}$

(c) $XI \geq 0.0$

(d) $XI \leq M_2$

ここで, $M_2 = \{ \text{倍精度: } 2^{50}\pi, \text{単精度: } 2^{18}\pi \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
2000	$XI \leq 2.0 / (\text{最大値})$ または $\nu(\log_e \frac{\nu}{x} - M_3) > M_4$ であった. (注意事項 (e) 参照) ($XI \neq 0.0, R \neq 0$) (overflow) 注 ν が整数値と一致するときは $ \nu $ に対して判定	$XO = (\text{最小値})$ を返す. 注 ν が整数で負のときは $XO = (-1)^{\nu+1} \times (\text{最大値})$ を返す.
3000	制限条件 (a), (b) または (c) を満足しなかった.	処理を打ち切る.

(6) 注意事項

- (a) $Y_\nu(x)$ は, x や ν が大きいほど計算時間がかかる. 一般に $R < 1000.0, XI < 1000.0$ とすることが望ましい.
 (b) 半奇数次の場合は, 球ベッセル関数を利用するのが良い.

$$Y_{n+\frac{1}{2}}(x) = \sqrt{\frac{2x}{\pi}} y_n(x)$$

- (c) $\nu < 0$ で ν が整数値と一致しないときは, このサブルーチンでは計算ができない. したがって, 漸化式を利用し求める.

$$Y_{\nu-1}(x) = \frac{2\nu}{x} Y_\nu(x) - Y_{\nu+1}(x)$$

- (d) $Y_\nu(x), Y_{\nu+1}(x), Y_{\nu+2}(x) \dots$ を同時に求める場合は, 漸化式を用いて順次求めた方が, このサブルーチンを幾度も使用するよりも速い.

- (e) このサブルーチンで IERR=2000 となるときの M_3, M_4 の値は以下の通りである.

$$M_3 = 0.3068, M_4 = \{ \text{倍精度: } 709.7827, \text{単精度: } 88.72284 \}$$

- (f) 第2種 ν 次ベッセル関数 $Y_\nu(z)$ はベッセルの微分方程式

$$z^2 \frac{d^2 w}{dz^2} + z \frac{dw}{dz} + (z^2 - \nu^2) w = 0$$

の基本解であり,

$$Y_\nu(z) = \frac{J_\nu(z) \cos \nu\pi - J_{-\nu}(z)}{\sin \nu\pi}$$

で定義される. ただし, ν が整数 n に等しい場合は,

$$Y_n(z) = \lim_{\nu \rightarrow n} Y_\nu(z)$$

とする.

- (g) 第2種ベッセル関数は第2種円柱関数とも呼ばれる.

- (h) ノイマン関数 $N_\nu(z)$ は, 第2種ベッセル関数 $Y_\nu(z)$ と同じものである.

(7) 使用例

(a) 問題

$Y_\nu(x)$ の値を $\nu = 3.3$, $x = 1.5$ について求める。

(b) 入力データ

$R = 3.3$, $XI = 1.5$

(c) 主プログラム

```

PROGRAM BIBYMX
! *** EXAMPLE OF DIBYMX ***
IMPLICIT REAL(8) (A-H,O-Z)
READ (5,*) R
READ (5,*) XI
WRITE (6,1000) R,XI
CALL DIBYMX(R,XI,XO,IERR)
WRITE (6,2000) IERR,XO
1000 FORMAT(' ',/,/,5X,'*** DIBYMX ***',/,/,6X,'** INPUT **',&
/,/,8X,'R = ',F6.2,5X,'XI = ',F6.2 )
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF YM(X)',/,/,10X,'XO = ',D18.10)
END

```

(d) 出力結果

```

*** DIBYMX ***
** INPUT **
R = 3.30    XI = 1.50

** OUTPUT**
IERR =      0
VALUE OF YM(X)
XO = -0.2895226697D+01

```

2.2.9 ZIBJNZ, CIBJNZ 複素変数第 1 種整数次ベッセル関数

(1) 機能

複素変数で整数次の第 1 種ベッセル関数

$$J_n(z) = \frac{1}{\pi} \int_0^\pi \cos(z \sin(t) - nt) dt$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL ZIBJNZ (N, ZI, ZO, IERR)

単精度サブルーチン:

CALL CIBJNZ (N, ZI, ZO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER(4)} \\ 64 \text{ ビット整数版では INTEGER(8)} \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入 力	次数 n
2	ZI	$\begin{Bmatrix} Z \\ C \end{Bmatrix}$	1	入 力	変数値 z
3	ZO	$\begin{Bmatrix} Z \\ C \end{Bmatrix}$	1	出 力	$J_n(z)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $|\Im(ZI)| \leq M_1$

ここで, $M_1 = \{ \text{倍精度: } 709.7827, \text{ 単精度: } 88.72284 \}$

(b) $|ZI| \leq M_2$

ここで, $M_2 = \{ \text{倍精度: } 2^{50}\pi, \text{ 単精度: } 2^{18}\pi \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1000	$ n (\log_e \frac{ n }{z} - M_3) > M_4$ (注意事項 (c) 参照) ($ ZI \neq 0.0, N \neq 0$) (underflow)	ZO = (0.0, 0.0) を返す.
3000	制限条件 (a) または (b) を満足しなかった.	処理を打ち切る.

(6) 注意事項

- (a) $J_n(z)$ は、 $|z|$ や n が大きいほど時間がかかり、一般に $|N| < 1000$, $|ZI| < 1000.0$ とすることが望ましい。
- (b) $J_n(z), J_{n+1}(z), J_{n+2}(z) \cdots$ を同時に求める場合は、漸化式を用いて順次求めた方が、このサブルーチンを幾度も使用するよりも速い。ただし、 n を増大させていくと不安定となるので、 n が減少する方向へ適用する。

漸化式:

$$J_{n-1}(z) = \frac{2n}{z} J_n(z) - J_{n+1}(z)$$

- (c) このサブルーチンで IERR=1000 となるとき M_3, M_4 の値は以下の通りである。

$$M_3 = 0.3068, M_4 = \{ \text{倍精度: } 709.7827, \text{単精度: } 88.72284 \}$$

- (d) 第1種 ν 次ベッセル関数 $J_\nu(z)$ はベッセルの微分方程式

$$z^2 \frac{d^2 w}{dz^2} + z \frac{dw}{dz} + (z^2 - \nu^2) w = 0$$

の基本解であり、

$$J_\nu(z) = \left(\frac{z}{2}\right)^\nu \sum_{m=0}^{\infty} \frac{(-1)^m}{m! \Gamma(m + \nu + 1)} \left(\frac{z}{2}\right)^{2m}$$

で定義される。

- (e) 第1種ベッセル関数は第1種円柱関数とも呼ばれる。

(7) 使用例

- (a) 問題

$J_n(z)$ の値を $n = 3, z = 1 + 2\sqrt{-1}$ について求める。

- (b) 入力データ

$N = 3, ZI = (1.0, 2.0)$

- (c) 主プログラム

```
PROGRAM AIBJNZ
! *** EXAMPLE OF ZIBJNZ ***
IMPLICIT COMPLEX(8) (A-H,O-Z)
READ (5,*) N
READ (5,'(D6.1,D6.1)') ZI
WRITE(6,1000) N,ZI
CALL ZIBJNZ(N,ZI,ZO,IERR)
WRITE(6,2000) IERR,ZO
1000 FORMAT(' ',/,/,5X,'*** ZIBJNZ ***',/,/,6X,'** INPUT **',&
/,/,8X,'N =',I3,5X,'ZI = (',F6.2,',',F6.2,',')')
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF JN(Z)',/,/,10X,'ZO = (',D18.10,',',D18.10,',')')
END
```

- (d) 出力結果

```
*** ZIBJNZ ***
** INPUT **
N = 3    ZI = ( 1.00 , 2.00 )

** OUTPUT**
IERR =    0
VALUE OF JN(Z)
ZO = ( -0.2810396668D+00 , 0.1717506200D-01 )
```

2.2.10 ZIBYNZ, CIBYNZ 複素変数第2種整数次ベッセル関数

(1) 機能

複素変数で整数次の第2種ベッセル関数

$$Y_n(z) = \frac{1}{\pi} \int_0^\pi \sin(z \sin(t) - nt) dt - \frac{1}{\pi} \int_0^\infty e^{-z \sinh(t)} [e^{nt} + (-1)^n e^{-nt}] dt$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL ZIBYNZ (N, ZI, ZO, IERR)

単精度サブルーチン:

CALL CIBYNZ (N, ZI, ZO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER(4)} \\ 64 \text{ ビット整数版では INTEGER(8)} \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入 力	次数 n
2	ZI	$\begin{Bmatrix} Z \\ C \end{Bmatrix}$	1	入 力	変数値 z
3	ZO	$\begin{Bmatrix} Z \\ C \end{Bmatrix}$	1	出 力	$Y_n(z)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $|ZI| > 0.0$

(b) $|\Im(ZI)| \leq M_1$

ここで, $M_1 = \{ \text{倍精度: } 709.7827, \text{ 単精度: } 88.72284 \}$

(c) $|ZI| \leq M_2$

ここで, $M_2 = \{ \text{倍精度: } 2^{50}\pi, \text{ 単精度: } 2^{18}\pi \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a), (b) または (c) を満足しなかつた.	処理を打ち切る.
4000	$ ZI \leq 2.0 /$ (最大値) または $ n (\log_e \frac{ n }{ z } - M_3) > M_4$ (注意事項 (d) 参照) ($ ZI \neq 0.0, N \neq 0$)	

(6) 注意事項

- (a) 第2種ベッセル関数 $N_n(z)$ は, $Y_n(z)$ と同じものである.
- (b) $Y_n(z)$ は, $|z|$ や n が大きいほど計算時間がかかる. 一般に $|N| < 1000, |ZI| < 1000.0$ とすることが望ましい.
- (c) $Y_n(z), Y_{n+1}(z), Y_{n+2}(z) \dots$ を同時に求める場合は, 漸化式を用いて順次求めた方が, このサブルーチンを幾度も使用するよりも速い.

漸化式:

$$Y_{n+1}(z) = \frac{2n}{z} Y_n(z) - Y_{n-1}(z)$$

- (d) このサブルーチンで IERR=4000 となるときの M_3, M_4 の値は以下の通りである.

$$M_3 = 0.3068, M_4 = \{ \text{倍精度: } 709.7827, \text{単精度: } 88.72284 \}$$

- (e) 第2種 ν 次ベッセル関数 $Y_\nu(z)$ はベッセルの微分方程式

$$z^2 \frac{d^2 w}{dz^2} + z \frac{dw}{dz} + (z^2 - \nu^2) w = 0$$

の基本解であり,

$$Y_\nu(z) = \frac{J_\nu(z) \cos \nu\pi - J_{-\nu}(z)}{\sin \nu\pi}$$

で定義される. ただし, ν が整数 n に等しい場合は,

$$Y_n(z) = \lim_{\nu \rightarrow n} Y_\nu(z)$$

とする.

- (f) 第2種ベッセル関数は第2種円柱関数とも呼ばれる.
- (g) ノイマン関数 $N_\nu(z)$ は, 第2種ベッセル関数 $Y_\nu(z)$ と同じものである.

(7) 使用例

- (a) 問題

$Y_n(z)$ の値を $n = 3, z = 1 + 2\sqrt{-1}$ について求める.

- (b) 入力データ

$$N = 3, ZI = (1.0, 2.0)$$

- (c) 主プログラム

```

PROGRAM AIBYNZ
! *** EXAMPLE OF ZIBYNZ ***
IMPLICIT COMPLEX(8) (A-H,O-Z)
READ (5,*) N
READ (5, '(D6.1,D6.1)') ZI
WRITE(6,1000) N,ZI
CALL ZIBYNZ(N,ZI,ZO,IERR)

```

```

WRITE(6,2000) IERR,ZO
1000 FORMAT(' ',/,/,5X,'*** ZIBYNYZ ***',/,/,6X,'** INPUT **',&
/,/,8X,'N =',I3,5X,'ZI = (',F6.2,',',F6.2,', )' )
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF YN(Z)',/,/,10X,'ZO = (',D18.10,',',D18.10,', )' )
END

```

(d) 出力結果

```

*** ZIBYNYZ ***
** INPUT **
  N = 3      ZI = ( 1.00 , 2.00 )

** OUTPUT**
IERR =      0
VALUE OF YN(Z)
  ZO = ( 0.2901532942D+00 , -0.2121187705D+00 )

```

2.3 ベッセル関数の零点

2.3.1 DIZBS0, RIZBS0

第1種0次ベッセル関数の正零点

(1) 機能

第1種0次ベッセル関数 $J_0(x)$ の正の零点を求める.

(2) 使用法

倍精度サブルーチン:

CALL DIZBS0 (N, Z, IERR)

単精度サブルーチン:

CALL RIZBS0 (N, Z, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入力	正の零点の個数
2	Z	$\begin{cases} D \\ R \end{cases}$	N	出力	第0次の第1種ベッセル関数の正の零点 (小さい方から順に入る)
3	IERR	I	1	出力	エラーインディケータ

(4) 制限条件

(a) $1 \leq N \leq 50$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.

(6) 注意事項

なし

(7) 使用例

(a) 問題

$N = 20$ として、 $J_0(x)$ の 20 番目までの正の零点を求める。

(b) 主プログラム

```

PROGRAM BIZBS0
IMPLICIT REAL(8) (A-H,O-Z)
PARAMETER(N=20)
REAL(8) Z(N)
CALL DIZBS0(N,Z,IERR)
WRITE(6,60)
WRITE(6,80)
WRITE(6,90) N
WRITE(6,100)
WRITE(6,110) 0,IERR
DO 1000 I=1,N
CALL DIBJOX(Z(I),Y,IERR)
WRITE(6,6000) I,0,Z(I),Y
1000 CONTINUE
STOP
60 FORMAT(1X,'*** DIZBS0 ***',/,/)
80 FORMAT(1X,' *** INPUT *** ',/,/)
90 FORMAT(1X,' N= ',I3,/,/)
100 FORMAT(1X,' *** OUTPUT *** ',/,/)
110 FORMAT(1X,'ORDER = ',I2,' IERR = ',I4,/,/)
6000 FORMAT(1X,I2,' TH ZERO OF J',I2,' ',F13.10,' ERR=',E10.3)
END

```

(c) 出力結果

```

*** DIZBS0 ***

*** INPUT ***

N= 20

*** OUTPUT ***

ORDER = 0 IERR = 0

1 TH ZERO OF J 0 2.4048255577 ERR= 0.000E+00
2 TH ZERO OF J 0 5.5200781103 ERR=-0.555E-16
3 TH ZERO OF J 0 8.6537279129 ERR=-0.855E-16
4 TH ZERO OF J 0 11.7915344390 ERR= 0.561E-15
5 TH ZERO OF J 0 14.9309177085 ERR= 0.314E-16
6 TH ZERO OF J 0 18.0710639679 ERR= 0.206E-16
7 TH ZERO OF J 0 21.2116366299 ERR= 0.235E-15
8 TH ZERO OF J 0 24.3524715307 ERR=-0.287E-15
9 TH ZERO OF J 0 27.4934791320 ERR=-0.163E-15
10 TH ZERO OF J 0 30.6346064684 ERR=-0.461E-16
11 TH ZERO OF J 0 33.7758202136 ERR=-0.170E-15
12 TH ZERO OF J 0 36.9170983537 ERR=-0.458E-15
13 TH ZERO OF J 0 40.0584257646 ERR=-0.374E-15
14 TH ZERO OF J 0 43.1997917132 ERR= 0.223E-15
15 TH ZERO OF J 0 46.3411883717 ERR= 0.411E-15
16 TH ZERO OF J 0 49.4826098974 ERR= 0.353E-16
17 TH ZERO OF J 0 52.6240518411 ERR= 0.292E-15
18 TH ZERO OF J 0 55.7655107550 ERR=-0.231E-15
19 TH ZERO OF J 0 58.9069839261 ERR=-0.845E-16
20 TH ZERO OF J 0 62.0484691902 ERR=-0.862E-16

```

2.3.2 DIZBS1, RIZBS1

第1種1次ベッセル関数の正零点

(1) 機能

第1種1次ベッセル関数 $J_1(x)$ の正の零点を求める.

(2) 使用法

倍精度サブルーチン:

CALL DIZBS1 (N, Z, IERR)

単精度サブルーチン:

CALL RIZBS1 (N, Z, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入 力	正の零点の個数
2	Z	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	N	出 力	第1次の第1種ベッセル関数の正の零点(小さい方から順に入る)
3	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $1 \leq N \leq 50$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.

(6) 注意事項

なし

(7) 使用例

(a) 問題

$N = 20$ として、 $J_1(x)$ の 20 番目までの正の零点を求める。

(b) 主プログラム

```

PROGRAM BIZBS1
IMPLICIT REAL(8) (A-H,O-Z)
PARAMETER(N=20)
REAL(8) Z(N)
CALL DIZBS1(N,Z,IERR)
WRITE(6,60)
WRITE(6,80)
WRITE(6,90) N
WRITE(6,100)
WRITE(6,110) 1,IERR
DO 1000 I=1,N
CALL DIBJ1X(Z(I),Y,IERR)
WRITE(6,6000) I,1,Z(I),Y
1000 CONTINUE
STOP
60 FORMAT(1X,'*** DIZBS1 ***',/,/)
80 FORMAT(1X,' *** INPUT *** ',/,/)
90 FORMAT(1X,' N= ',I3,/,/)
100 FORMAT(1X,' *** OUTPUT *** ',/,/)
110 FORMAT(1X,'ORDER = ',I2,' IERR = ',I4,/,/)
6000 FORMAT(1X,I2,' TH ZERO OF J',I2,' ',F13.10,' ERR=',E10.3)
END

```

(c) 出力結果

```

*** DIZBS1 ***

*** INPUT ***

N= 20

*** OUTPUT ***

ORDER = 1 IERR = 0

1 TH ZERO OF J 1 3.8317059702 ERR= 0.000E+00
2 TH ZERO OF J 1 7.0155866698 ERR= 0.555E-16
3 TH ZERO OF J 1 10.1734681351 ERR= 0.892E-16
4 TH ZERO OF J 1 13.3236919363 ERR= 0.159E-15
5 TH ZERO OF J 1 16.4706300509 ERR= 0.186E-15
6 TH ZERO OF J 1 19.6158585105 ERR=-0.281E-15
7 TH ZERO OF J 1 22.7600843806 ERR=-0.247E-15
8 TH ZERO OF J 1 25.9036720876 ERR= 0.777E-16
9 TH ZERO OF J 1 29.0468285349 ERR=-0.187E-15
10 TH ZERO OF J 1 32.1896799110 ERR= 0.356E-15
11 TH ZERO OF J 1 35.3323075501 ERR= 0.384E-15
12 TH ZERO OF J 1 38.4747662348 ERR= 0.337E-16
13 TH ZERO OF J 1 41.6170942128 ERR=-0.511E-16
14 TH ZERO OF J 1 44.7593189977 ERR=-0.161E-15
15 TH ZERO OF J 1 47.9014608872 ERR= 0.306E-15
16 TH ZERO OF J 1 51.0435351836 ERR=-0.192E-15
17 TH ZERO OF J 1 54.1855536411 ERR= 0.132E-15
18 TH ZERO OF J 1 57.3275254379 ERR=-0.490E-16
19 TH ZERO OF J 1 60.4694578453 ERR=-0.125E-15
20 TH ZERO OF J 1 63.6113566985 ERR=-0.785E-16

```

2.3.3 DIZBSN, RIZBSN

第1種整数次ベッセル関数の正零点

(1) 機能

第1種整数次ベッセル関数 $J_m(x)$ の正の零点を求める.

(2) 使用法

倍精度サブルーチン:

CALL DIZBSN (N,M,LF, Z, WORK, IERR)

単精度サブルーチン:

CALL RIZBSN (N,M,LF, Z, WORK, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入 力	正の零点の個数
2	M	I	1	入 力	次数 m
3	LF	I	1	入 力	近似倍率
4	Z	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	N	出 力	正の零点 (小さい方から順に入る)
5	WORK	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	内容参照	ワーク	作業領域 大きさ: $2 \times (LF \times N + 1) \times (LF \times N + 2)$
6	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $1 \leq N \leq 50$ ($M = -1, 0, +1$) または $1 \leq N$ (それ以外)

(b) $LF \geq 1$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) または (b) を満足しなかった.	処理を打ち切る.
3500	反復改良による解の改良ができなかった.	
3600	固有値問題の解が求められなかった.	

(6) 注意事項

- (a) N は 50 程度を上限とすべきである。
- (b) 反復改良は $LF \times N$ 回で打ち切られる。さらに、この値は反復改良の初期値を求める際に解かれる固有値問題の次数としても使われる。この値の大きさが十分ではない場合、反復改良に使われる初期値の近似精度が悪くなり、 $IERR=3500,3600$ が発生する原因となる。逆に、この値が大きすぎると、反復改良に使われる初期値を求める計算にかかる処理時間が増大する。
目安として、M が 10 程度の場合は $N \times LF$ を 24 以上、M が 18 程度の場合は $N \times LF$ を 30 以上にとるとよい。
- (c) $M=-1,0,+1$ のときは、数値テーブルを参照して計算しているので、処理時間は比較的になる。

(7) 使用例

(a) 問題

$N=20, M=10$ として、 $J_{10}(x)$ の 20 番目までの正の零点を求める。

(b) 主プログラム

```

PROGRAM BIZBSN
IMPLICIT REAL(8) (A-H,O-Z)
PARAMETER(N=20)
REAL(8) Z(N), WORK(2*(1+8*N)*(2+8*N))
II=10
CALL DIZBSN(N,II,8,Z,WORK,IERR)
WRITE(6,10)
WRITE(6,20)
WRITE(6,30) N,II
WRITE(6,40)
WRITE(6,50) II,IERR
DO 1000 I=1,N
CALL DIBJNX(II, Z(I), DERR, IERR)
WRITE(6,6000) I,II ,Z(I),DERR
1000 CONTINUE
STOP
10 FORMAT(1X, ' *** DIZBSN *** ',/,/)
20 FORMAT(1X, ' *** INPUT *** ',/,/)
30 FORMAT(1X, ' N= ',I3, ' M= ',I3,/,/)
40 FORMAT(1X, ' *** OUTPUT *** ',/,/)
50 FORMAT(1X, 'ORDER = ',I3, ' IERR = ',I4,/,/)
6000 FORMAT(1X,I2, ' TH ZERO OF J ',I2, ' ',F13.10,&
' ERR = ',E13.3)
END

```

(c) 出力結果

```

*** DIZBSN ***

*** INPUT ***

N= 20 M= 10

*** OUTPUT ***

ORDER = 10 IERR = 0

1 TH ZERO OF J10 14.4755006866 ERR = -0.167E-15
2 TH ZERO OF J10 18.4334636670 ERR = 0.194E-15
3 TH ZERO OF J10 22.0469853647 ERR = 0.194E-15
4 TH ZERO OF J10 25.5094505542 ERR = -0.264E-15
5 TH ZERO OF J10 28.8873750635 ERR = 0.139E-16
6 TH ZERO OF J10 32.2118561997 ERR = -0.319E-15
7 TH ZERO OF J10 35.4999092054 ERR = 0.444E-15
8 TH ZERO OF J10 38.7618070179 ERR = 0.194E-15
9 TH ZERO OF J10 42.0041902367 ERR = 0.555E-16
10 TH ZERO OF J10 45.2315741035 ERR = 0.250E-15
11 TH ZERO OF J10 48.4471513873 ERR = -0.555E-16
12 TH ZERO OF J10 51.6532516682 ERR = 0.167E-15
13 TH ZERO OF J10 54.8516190760 ERR = -0.160E-15
14 TH ZERO OF J10 58.0435879282 ERR = -0.694E-16
15 TH ZERO OF J10 61.2301979773 ERR = -0.340E-15
16 TH ZERO OF J10 64.4122724129 ERR = 0.354E-15
17 TH ZERO OF J10 67.5904720737 ERR = -0.354E-15
18 TH ZERO OF J10 70.7653339962 ERR = -0.607E-15
19 TH ZERO OF J10 73.9372993818 ERR = 0.461E-15
20 TH ZERO OF J10 77.1067342469 ERR = -0.163E-15

```

2.3.4 DIZBYN, RIZBYN

第 2 種整数次ベッセル関数の正零点

(1) 機能

第 2 種整数次ベッセル関数 $Y_m(x)$ の正の零点を求める.

(2) 使用法

倍精度サブルーチン:

CALL DIZBYN (N, M, Z, NCONV, IERR)

単精度サブルーチン:

CALL RIZBYN (N, M, Z, NCONV, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入 力	正の零点の個数
2	M	I	1	入 力	次数 m
3	Z	$\begin{cases} D \\ R \end{cases}$	N	出 力	正の零点 (小さい方から順に入る.)
4	NCONV	I	1	出 力	最大反復回数
5	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $N \geq 1$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
4000	最大反復回数以内で収束しなかった.	

(6) 注意事項

(a) 次数の絶対値が 5 以上の第 2 種ベッセル関数の零点を求めるときは, 倍精度版を使用する.

(7) 使用例

(a) 問題

$Y_{10}(x)$ の 20 番目までの正零点を求める.

(b) 入力データ

N=20, M=10

(c) 主プログラム

```

PROGRAM BIZBYN
! *** EXAMPLE OF DIZBYN ***
IMPLICIT NONE
!
INTEGER N
INTEGER M, NCONV, IERR, I
PARAMETER( N = 20 )
REAL(8) Z(N), DERR
!
M = 10
!
WRITE(6,6000) N,M
!
CALL DIZBYN(N,M,Z,NCONV,IERR)
!
WRITE(6,6010) IERR, NCONV, M
DO 100 I=1,N
CALL DIBYNX(M, Z(I), DERR, IERR)
IF(IERR.NE.0) WRITE(6,6020)
WRITE(6,6030) I,M ,Z(I),DERR
100 CONTINUE
!
STOP
6000 FORMAT(/,&
1X,'*** DIZBYN ***',/,/,&
1X,' ** INPUT **',/,/,&
1X,' N= ',I3,' M= ',I3,/)
6010 FORMAT(/,&
1X,' ** OUTPUT **',/,/,&
1X,' IERR =',I5,/,/,&
1X,' NCONV = ',I4,/,/,&
1X,' NO ',Y',I2,&
,'( ZERO POINT ) VALUE OF YM(X)',/)
6020 FORMAT(1X,' ** ERROR IN DIBYNX',/)
6030 FORMAT(1X,5X,I2,' Y',I2,'( ',F13.10,' ) ',D13.3)
END

```

(d) 出力結果

```

*** DIZBYN ***
** INPUT **
N= 20      M= 10

** OUTPUT **
IERR = 0
NCONV = 29

NO  Y10( ZERO POINT )      VALUE OF YM(X)
1   Y10( 12.1289277044 )    -0.335D-11
2   Y10( 16.5222843948 )    -0.468D-12
3   Y10( 20.2659845012 )    0.695D-13
4   Y10( 23.7916697195 )    0.145D-12
5   Y10( 27.2065688816 )    -0.144D-13
6   Y10( 30.5550200110 )    0.247D-13
7   Y10( 33.8596838727 )    -0.322D-14
8   Y10( 37.1336497603 )    0.178D-13
9   Y10( 40.3851175938 )    -0.357D-14
10  Y10( 43.6195330856 )    0.541D-15
11  Y10( 46.8406766306 )    0.798D-15
12  Y10( 50.0512658519 )    0.352D-14
13  Y10( 53.2533105567 )    0.125D-15
14  Y10( 56.4483324889 )    -0.101D-14
15  Y10( 59.6375070056 )    0.194D-14
16  Y10( 62.8217575527 )    -0.196D-14
17  Y10( 66.0018204937 )    0.330D-14
18  Y10( 69.1782907026 )    -0.251D-13
19  Y10( 72.3516543205 )    0.351D-14
20  Y10( 75.5223127385 )    -0.410D-14

```

2.3.5 DIZBSL, RIZBSL

 $aJ_0(\alpha) + xJ_1(\alpha)$ の正零点

(1) 機能

任意の実パラメータ a を与えて、ベッセル関数を含む方程式 $aJ_0(\alpha) + \alpha J_1(\alpha) = 0$ の正の解 α を求める。

(2) 使用法

倍精度サブルーチン:

CALL DIZBSL (N, A, LF, Z, WORK, IERR)

単精度サブルーチン:

CALL RIZBSL (N, A, LF, Z, WORK, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入 力	正の解の個数
2	A	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	パラメータ a
3	LF	I	1	入 力	近似倍率
4	Z	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	N	出 力	正の解 α (小さい方から順に入る)
5	WORK	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	内容参照	ワーク	作業領域 大きさ: $2 \times (LF \times N + 1) \times (LF \times N + 2)$
6	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $N \geq 1$

(b) $LF \geq 1$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) または (b) を満足しなかった.	処理を打ち切る.
3500	反復改良による解の改良ができなかった.	
3600	固有値問題の解が求められなかった.	

(6) 注意事項

- (a) パラメータ a (入力値 A) の有効な適用範囲は $10^{-10} \leq |a| \leq 10^4$ および $a = 0$ である.
- (b) N は 50 程度を上限とすべきである.
- (c) 反復改良は $LF \times N$ 回で打ち切られる.
- (d) LF は 8 程度が望ましい.

(7) 使用例

(a) 問題

$a = -\beta \frac{J_1(\beta)}{J_0(\beta)}$ ($\beta = 2.304780$), $N=20$, $LF=8$ に対して, $aJ_0(\alpha) + \alpha J_1(\alpha) = 0$ の正の解 α を求める.

(b) 主プログラム

```
PROGRAM BIZBSL
IMPLICIT REAL(8) (A-H,O-Z)
PARAMETER(N=20)
REAL(8) Z(N), WORK(2*(1+8*N)*(2+8*N))
!
A=2.304780D0
CALL DIBJOX(A,F,IERR)
CALL DIBJ1X(A,D,IERR)
A=-D*A/F
WRITE(6,10)
WRITE(6,20)
WRITE(6,6000) A
CALL DIZBSL(N,A,8,Z,WORK,IERR)
WRITE(6,30)
WRITE(6,40) IERR
DO 1000 J=1,N
CALL DIBJOX(Z(J),F,IERR)
CALL DIBJ1X(Z(J),D,IERR)
P=A*F+D*Z(J)
WRITE(6,6100) J,Z(J),P
1000 CONTINUE
STOP
10 FORMAT(1X,' *** DIZBSL *** ',/,/)
20 FORMAT(1X,' *** INPUT *** ',/,/)
30 FORMAT(1X,' *** OUTPUT *** ',/,/)
40 FORMAT(1X,' IERR = ',I4,/,/)
6000 FORMAT(1X,' INPUT A = ',F10.6,/,/)
6100 FORMAT(1X,' ALPHA(',I2,') = ',F10.6,' ERR= ',E11.4)
END
```

(c) 出力結果

```
*** DIZBSL ***

*** INPUT ***

INPUT A = -23.456847

*** OUTPUT ***

IERR = 0

ALPHA( 1)= 2.304780 ERR= 0.0000E+00
ALPHA( 2)= 5.293488 ERR= 0.1332E-14
ALPHA( 3)= 8.306597 ERR= -0.4885E-14
ALPHA( 4)= 11.333025 ERR= -0.1243E-13
ALPHA( 5)= 14.371644 ERR= 0.6217E-14
ALPHA( 6)= 17.421959 ERR= -0.5773E-14
ALPHA( 7)= 20.483189 ERR= 0.4441E-14
ALPHA( 8)= 23.554295 ERR= 0.1332E-14
ALPHA( 9)= 26.634127 ERR= 0.0000E+00
ALPHA(10)= 29.721552 ERR= 0.5773E-14
ALPHA(11)= 32.815519 ERR= -0.3553E-14
ALPHA(12)= 35.915098 ERR= 0.1421E-13
ALPHA(13)= 39.019482 ERR= -0.1865E-13
ALPHA(14)= 42.127984 ERR= 0.1510E-13
ALPHA(15)= 45.240020 ERR= -0.4441E-15
ALPHA(16)= 48.355101 ERR= -0.6661E-14
ALPHA(17)= 51.472814 ERR= 0.1554E-13
ALPHA(18)= 54.592810 ERR= 0.1732E-13
ALPHA(19)= 57.714796 ERR= 0.2309E-13
ALPHA(20)= 60.838523 ERR= 0.1199E-13
```

2.4 変形ベッセル関数

2.4.1 WIBI0X, VIBI0X

第1種0次変形ベッセル関数

(1) 機能

$x = X_i$ に対して0次の第1種変形ベッセル関数

$$I_0(x) = \frac{1}{\pi} \int_0^\pi e^{x \cos(t)} dt$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL WIBI0X (NV, XI, XO, IERR)

単精度サブルーチン:

CALL VIBI0X (NV, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER(4)} \\ 64 \text{ ビット整数版では INTEGER(8)} \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力データ数
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	入 力	変数値 X_i
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	出 力	$I_0(X_i)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
2000	制限条件 (a) を満足しなかった.	処理を打ち切る.
2000+i	$ XI(i) > M$ であった. (注意事項 (a) 参照) (overflow)	XO(i) = (最大値) を返す.

(6) 注意事項

- (a) このサブルーチンで IERR=2000 となるときの M の値は以下の通りである.

$M = \{ \text{倍精度: 713.067, 単精度: 90.978} \}$

- (b) 第1種 ν 次変形ベッセル関数 $I_\nu(z)$ は変形されたベッセルの微分方程式

$$z^2 \frac{d^2 w}{dz^2} + z \frac{dw}{dz} - (z^2 + \nu^2)w = 0$$

の特殊解であり,

$$I_{\pm\nu}(z) = e^{\mp\sqrt{-1}\pi/2} J_{\pm\nu}(\sqrt{-1}z)$$

で定義される.

(7) 使用例

- (a) 問題

$I_0(x)$ の値を $x = 0.0, 0.1, \dots, 0.9$ について求める.

- (b) 主プログラム

```

PROGRAM EIBIOX
IMPLICIT REAL(8)(A-H,O-Z)
PARAMETER (NV=10)
REAL(8) XI(NV), XO(NV)
CHARACTER*6 CNAME, CFNC
PARAMETER( CNAME='WIBIOX', CFNC=' IO' )
!
DNV=NV
DO 1000 I=1,NV
  XI(I)=(I-1)/DNV
1000 CONTINUE
!
CALL WIBIOX( NV, XI, XO, IERR )
!
WRITE(6,6000) CNAME
WRITE(6,6100)
DO 2000 I=1,NV
  WRITE(6,6200) I,XI(I)
2000 CONTINUE
!
WRITE(6,6300)
WRITE(6,6400) IERR
DO 3000 I=1,NV
  WRITE(6,6500) CFNC,XI(I), XO(I)
3000 CONTINUE
STOP
!
6000 FORMAT(1X,'*** ',A6,' *')
6100 FORMAT(1X,'*** INPUT *')
6200 FORMAT(1X,'XI(',I2,')=',F10.6 )
6300 FORMAT(1X,'*** OUTPUT *')
6400 FORMAT(1X,'IERR=',I5 )
6500 FORMAT(1X,A6,'(',F10.6,')=',F10.6 )
END

```

- (c) 出力結果

```

*** WIBIOX *
*** INPUT *
XI( 1)= 0.000000
XI( 2)= 0.100000
XI( 3)= 0.200000
XI( 4)= 0.300000
XI( 5)= 0.400000
XI( 6)= 0.500000
XI( 7)= 0.600000
XI( 8)= 0.700000
XI( 9)= 0.800000
XI(10)= 0.900000
*** OUTPUT *
IERR= 0
IO( 0.000000)= 1.000000
IO( 0.100000)= 1.002502
IO( 0.200000)= 1.010025
IO( 0.300000)= 1.022627
IO( 0.400000)= 1.040402
IO( 0.500000)= 1.063483
IO( 0.600000)= 1.092045
IO( 0.700000)= 1.126303
IO( 0.800000)= 1.166515
IO( 0.900000)= 1.212985

```

2.4.2 WIBK0X, VIBK0X

第 2 種 0 次変形ベッセル関数

(1) 機能

$x = X_i$ に対して 0 次の第 2 種変形ベッセル関数

$$K_0(x) = \int_0^{\infty} e^{-x \cosh(t)} dt$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL WIBK0X (NV, XI, XO, IERR)

単精度サブルーチン:

CALL VIBK0X (NV, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力データ数
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	入 力	変数値 X_i
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	出 力	$K_0(X_i)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(b) $XI(i) \geq 0.0$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1000	$XI(i) > M$ であった. (注意事項 (a) 参照) (underflow)	$XO(i) = 0.0$ を返す.
2000	$XI(i) = 0.0$ (overflow)	$XO(i) = (\text{最大値})$ を返す.
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
3000+i	$XI(i)$ が制限条件 (b) を満足しなかった.	

(6) 注意事項

- (a) このサブルーチンで IERR=1000 となるときの M の値は以下の通りである.

$M = \{ \text{倍精度: 705.117, 単精度: 85.114} \}$

- (b) 第2種 ν 次変形ベッセル関数 $K_\nu(z)$ は変形されたベッセルの微分方程式

$$z^2 \frac{d^2 w}{dz^2} + z \frac{dw}{dz} - (z^2 + \nu^2)w = 0$$

の特殊解であり,

$$K_\nu(z) = \frac{\pi}{2} \frac{I_{-\nu}(z) - I_\nu(z)}{\sin \nu\pi}$$

で定義される. ただし, ν が整数 n に等しい場合は,

$$K_n(z) = \lim_{\nu \rightarrow n} K_\nu(z)$$

とする.

(7) 使用例

- (a) 問題

$K_0(x)$ の値を $x = 0.1, 0.2, \dots, 1.0$ について求める.

- (b) 主プログラム

```

PROGRAM EIBK0X
IMPLICIT REAL(8)(A-H,O-Z)
PARAMETER (NV=10)
REAL(8) XI(NV), XO(NV)
CHARACTER*6 CNAME, CFNC
PARAMETER( CNAME='WIBK0X', CFNC=' KO' )
!
DNV=NV
DO 1000 I=1,NV
  XI(I)=I/DNV
1000 CONTINUE
!
CALL WIBK0X( NV, XI, XO, IERR )
!
WRITE(6,6000) CNAME
WRITE(6,6100)
DO 2000 I=1,NV
  WRITE(6,6200) I,XI(I)
2000 CONTINUE
!
WRITE(6,6300)
WRITE(6,6400) IERR
DO 3000 I=1,NV
  WRITE(6,6500) CFNC,XI(I), XO(I)
3000 CONTINUE
STOP
!
6000 FORMAT(1X,'*** ',A6,' *')
6100 FORMAT(1X,'*** INPUT *')
6200 FORMAT(1X,'XI(',I2,')=',F10.6 )
6300 FORMAT(1X,'*** OUTPUT *')
6400 FORMAT(1X,'IERR=',I5 )
6500 FORMAT(1X,A6,'(',F10.6,')=',F10.6 )
END

```

- (c) 出力結果

```

*** WIBK0X *
*** INPUT *
XI( 1)= 0.100000
XI( 2)= 0.200000
XI( 3)= 0.300000
XI( 4)= 0.400000
XI( 5)= 0.500000
XI( 6)= 0.600000
XI( 7)= 0.700000
XI( 8)= 0.800000
XI( 9)= 0.900000
XI(10)= 1.000000
*** OUTPUT *
IERR= 0
KO( 0.100000)= 2.427069
KO( 0.200000)= 1.752704
KO( 0.300000)= 1.372460
KO( 0.400000)= 1.114529
KO( 0.500000)= 0.924419
KO( 0.600000)= 0.777522
KO( 0.700000)= 0.660520
KO( 0.800000)= 0.565347
KO( 0.900000)= 0.486730
KO( 1.000000)= 0.421024

```

2.4.3 WIBI1X, VIBI1X

第 1 種 1 次変形ベッセル関数

(1) 機能

$x = X_i$ に対して 1 次の第 1 種変形ベッセル関数

$$I_1(x) = \frac{1}{\pi} \int_0^\pi e^{x \cos(t)} \cos(t) dt$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL WIBI1X (NV, XI, XO, IERR)

単精度サブルーチン:

CALL VIBI1X (NV, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\left\{ \begin{array}{l} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{array} \right\}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力データ数
2	XI	$\left\{ \begin{array}{l} D \\ R \end{array} \right\}$	NV	入 力	変数値 X_i
3	XO	$\left\{ \begin{array}{l} D \\ R \end{array} \right\}$	NV	出 力	$I_1(X_i)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
2000	制限条件 (a) を満足しなかった.	処理を打ち切る.
2000+i	$ XI(i) > M$ であった. (注意事項 (a) 参照) (overflow)	$XI(i) \geq 0.0$ のとき $XO(i) = (\text{最大値})$ $XI(i) < 0.0$ のとき $XO(i) = -(\text{最大値})$

(6) 注意事項

- (a) このサブルーチンで IERR=2000 となるときの M の値は以下の通りである.

$M = \{ \text{倍精度: 713.067, 単精度: 90.978} \}$

- (b) 第1種 ν 次変形ベッセル関数 $I_\nu(z)$ は変形されたベッセルの微分方程式

$$z^2 \frac{d^2 w}{dz^2} + z \frac{dw}{dz} - (z^2 + \nu^2)w = 0$$

の特殊解であり,

$$I_{\pm\nu}(z) = e^{\mp\sqrt{-1}\pi/2} J_{\pm\nu}(\sqrt{-1}z)$$

で定義される.

(7) 使用例

- (a) 問題

$I_1(x)$ の値を $x = 0.0, 0.1, 0.2, \dots, 0.9$ について求める.

- (b) 主プログラム

```

PROGRAM EIBI1X
IMPLICIT REAL(8)(A-H,O-Z)
PARAMETER (NV=10)
REAL(8) XI(NV), XO(NV)
CHARACTER*6 CNAME, CFNC
PARAMETER( CNAME='WIBI1X', CFNC=' I1' )
!
DNV=NV
DO 1000 I=1,NV
  XI(I)=(I-1)/DNV
1000 CONTINUE
!
CALL WIBI1X( NV, XI, XO, IERR )
!
WRITE(6,6000) CNAME
WRITE(6,6100)
DO 2000 I=1,NV
  WRITE(6,6200) I,XI(I)
2000 CONTINUE
!
WRITE(6,6300)
WRITE(6,6400) IERR
DO 3000 I=1,NV
  WRITE(6,6500) CFNC,XI(I), XO(I)
3000 CONTINUE
STOP
!
6000 FORMAT(1X,'*** ',A6,' *')
6100 FORMAT(1X,'*** INPUT *')
6200 FORMAT(1X,'XI(',I2,')=',F10.6 )
6300 FORMAT(1X,'*** OUTPUT *')
6400 FORMAT(1X,'IERR=',I5 )
6500 FORMAT(1X,A6,'(',F10.6,')=',F10.6 )
END

```

- (c) 出力結果

```

*** WIBI1X *
*** INPUT *
XI( 1)= 0.000000
XI( 2)= 0.100000
XI( 3)= 0.200000
XI( 4)= 0.300000
XI( 5)= 0.400000
XI( 6)= 0.500000
XI( 7)= 0.600000
XI( 8)= 0.700000
XI( 9)= 0.800000
XI(10)= 0.900000
*** OUTPUT *
IERR= 0
I1( 0.000000)= 0.000000
I1( 0.100000)= 0.050063
I1( 0.200000)= 0.100501
I1( 0.300000)= 0.151694
I1( 0.400000)= 0.204027
I1( 0.500000)= 0.257894
I1( 0.600000)= 0.313704
I1( 0.700000)= 0.371880
I1( 0.800000)= 0.432865
I1( 0.900000)= 0.497126

```

2.4.4 WIBK1X, VIBK1X

第 2 種 1 次変形ベッセル関数

(1) 機能

$x = X_i$ に対して 1 次の第 2 種変形ベッセル関数

$$K_1(x) = \int_0^{\infty} e^{-x \cosh(t)} \cosh(t) dt$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL WIBK1X (NV, XI, XO, IERR)

単精度サブルーチン:

CALL VIBK1X (NV, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力データ数
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	入 力	変数値 X_i
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	出 力	$K_1(X_i)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(b) $XI(i) \geq 0.0$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1000	$XI(i) > M$ であった. (注意事項 (a) 参照) (underflow)	$XO(i) = 0.0$ を返す.
2000	$XI(i) \leq 1.0/(\text{最大値})$ (overflow)	$XO(i) = (\text{最大値})$ を返す.
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
3000+i	$XI(i)$ が制限条件 (b) を満足しなかった.	

(6) 注意事項

- (a) このサブルーチンで IERR=1000 となるときの M の値は以下の通りである.

$M = \{ \text{倍精度: 705.117, 単精度: 85.114} \}$

- (b) 第2種 ν 次変形ベッセル関数 $K_\nu(z)$ は変形されたベッセルの微分方程式

$$z^2 \frac{d^2 w}{dz^2} + z \frac{dw}{dz} - (z^2 + \nu^2)w = 0$$

の特殊解であり,

$$K_\nu(z) = \frac{\pi I_{-\nu}(z) - I_\nu(z)}{2 \sin \nu \pi}$$

で定義される. ただし, ν が整数 n に等しい場合は,

$$K_n(z) = \lim_{\nu \rightarrow n} K_\nu(z)$$

とする.

(7) 使用例

- (a) 問題

$K_1(x)$ の値を $x = 0.1, 0.2, \dots, 1.0$ について求める.

- (b) 主プログラム

```

PROGRAM EIBK1X
IMPLICIT REAL(8)(A-H,O-Z)
PARAMETER (NV=10)
REAL(8) XI(NV), XO(NV)
CHARACTER*6 CNAME, CFNC
PARAMETER( CNAME='WIBK1X', CFNC=' K1' )
!
DNV=NV
DO 1000 I=1,NV
  XI(I)=I/DNV
1000 CONTINUE
!
CALL WIBK1X( NV, XI, XO, IERR )
!
WRITE(6,6000) CNAME
WRITE(6,6100)
DO 2000 I=1,NV
  WRITE(6,6200) I,XI(I)
2000 CONTINUE
!
WRITE(6,6300)
WRITE(6,6400) IERR
DO 3000 I=1,NV
  WRITE(6,6500) CFNC,XI(I), XO(I)
3000 CONTINUE
STOP
!
6000 FORMAT(1X,'*** ',A6,' *')
6100 FORMAT(1X,'*** INPUT *')
6200 FORMAT(1X,'XI( ',I2,')=',F10.6 )
6300 FORMAT(1X,'*** OUTPUT *')
6400 FORMAT(1X,'IERR=',I5 )
6500 FORMAT(1X,A6,'(',',F10.6,')=',F10.6 )
END

```

- (c) 出力結果

```

*** WIBK1X *
*** INPUT *
XI( 1)= 0.100000
XI( 2)= 0.200000
XI( 3)= 0.300000
XI( 4)= 0.400000
XI( 5)= 0.500000
XI( 6)= 0.600000
XI( 7)= 0.700000
XI( 8)= 0.800000
XI( 9)= 0.900000
XI(10)= 1.000000
*** OUTPUT *
IERR= 0
K1( 0.100000)= 9.853845
K1( 0.200000)= 4.775973
K1( 0.300000)= 3.055992
K1( 0.400000)= 2.184354
K1( 0.500000)= 1.656441
K1( 0.600000)= 1.302835
K1( 0.700000)= 1.050284
K1( 0.800000)= 0.861782
K1( 0.900000)= 0.716534
K1( 1.000000)= 0.601907

```

2.4.5 DIBINX, RIBINX

第 1 種整数次変形ベッセル関数

(1) 機能

整数次の第 1 種変形ベッセル関数

$$I_n(x) = \frac{1}{\pi} \int_0^\pi e^{x \cos(t)} \cos(nt) dt$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL DIBINX (N, XI, XO, IERR)

単精度サブルーチン:

CALL RIBINX (N, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\left\{ \begin{array}{l} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{array} \right\}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入 力	次数 n
2	XI	$\left\{ \begin{array}{l} D \\ R \end{array} \right\}$	1	入 力	変数値 x
3	XO	$\left\{ \begin{array}{l} D \\ R \end{array} \right\}$	1	出 力	$I_n(x)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $|XI| \leq M$

ここで, $M = \{ \text{倍精度: } 713.067, \text{単精度: } 90.978 \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1000	$ n (\log_e \frac{x}{n} - M_1) > M_2$ であった. (注意事項 (c) 参照) ($XI \neq 0.0, N \neq 0$) (underflow)	XO = 0.0 を返す.
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.

(6) 注意事項

- (a) $I_n(x)$ は, x や n が大きいほど計算時間がかかる. 一般に $|N| < 1000, |XI| < 1000.0$ とすることが望ましい.
- (b) $I_n(x), I_{n+1}(x), I_{n+2}(x) \cdots$ を同時に求める場合は, 漸化式を用いて順次求めた方が, このサブルーチンを幾度も使用するよりも速い. ただし, n を増大させていくと不安定となるので, n が減少する方向へ適用する.

漸化式

$$I_{n-1}(x) = \frac{2n}{x} I_n(x) + I_{n+1}(x)$$

- (c) このサブルーチンで IERR=1000 となるときの M_1, M_2 の値は以下の通りである.

$$M_1 = 0.3068, M_2 = \{ \text{倍精度: } 709.7827, \text{単精度: } 88.72284 \}$$

- (d) 第 1 種 ν 次変形ベッセル関数 $I_\nu(z)$ は変形されたベッセルの微分方程式

$$z^2 \frac{d^2 w}{dz^2} + z \frac{dw}{dz} - (z^2 + \nu^2) w = 0$$

の特殊解であり,

$$I_{\pm\nu}(z) = e^{\mp\sqrt{-1}\pi/2} J_{\pm\nu}(\sqrt{-1}z)$$

で定義される.

(7) 使用例

- (a) 問題

$I_n(x)$ の値を $n = 5, x = 1.5$ について求める.

- (b) 入力データ

N = 5, XI = 1.5

- (c) 主プログラム

```

PROGRAM BIBINX
! *** EXAMPLE OF DIBINX ***
IMPLICIT REAL(8) (A-H,O-Z)
READ (5,*) N
READ (5,*) XI
WRITE (6,1000) N,XI
CALL DIBINX(N,XI,XO,IERR)
WRITE (6,2000) IERR,XO
1000 FORMAT(' ',/,/,5X,'*** DIBINX ***',/,/,6X,'** INPUT **',&
/,/,8X,'N =',I3,5X,'XI =',F6.2 )
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF IN(X)',/,/,10X,'XO = ',D18.10)
END
    
```

- (d) 出力結果

```

*** DIBINX ***
** INPUT **
N = 5    XI = 1.50

** OUTPUT**
IERR =    0
VALUE OF IN(X)
XO = 0.2170559569D-02
    
```

2.4.6 DIBKNX, RIBKNX

第 2 種整数次変形ベッセル関数

(1) 機能

整数次の第 2 種変形ベッセル関数

$$K_n(x) = \int_0^{\infty} e^{-x \cosh(t)} \cosh(nt) dt$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL DIBKNX (N, XI, XO, IERR)

単精度サブルーチン:

CALL RIBKNX (N, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: { 32 ビット整数版では INTEGER(4)
R:単精度実数型 C:単精度複素数型 { 64 ビット整数版では INTEGER(8) }

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入 力	次数 n
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 x
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	$K_n(x)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $XI \geq 0.0$

(b) $XI \leq M$

ここで, $M = \{ \text{倍精度: } 705.117, \text{単精度: } 85.114 \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
2000	$XI \leq 2.0/$ (最大値) または $ n (\log_e \frac{ n }{x} - M_1) > M_2$ (注意事項 (c) 参照) ($XI \neq 0.0, N \neq 0$) (overflow)	XO =(最大値) を返す.
3000	制限条件 (a) または (b) を満足しなかった.	処理を打ち切る.

(6) 注意事項

- (a) $K_n(x)$ は, x や n が大きいほど計算時間がかかる. 一般に $|N| < 1000$, $XI < 1000.0$ とすることが望ましい.
- (b) $K_n(x), K_{n+1}(x), K_{n+2}(x) \dots$ を同時に求める場合は, 漸化式を用いて順次求めた方が, このサブルーチンを幾度も使用するよりも速い.

漸化式

$$K_{n+1}(x) = \frac{2n}{x}K_n(x) + K_{n-1}(x)$$

- (c) このサブルーチンで IERR=2000 となるときの M_1, M_2 の値は以下の通りである.

$M_1 = 0.3068, M_2 = \{ \text{倍精度: } 709.7827, \text{単精度: } 88.72284 \}$

- (d) 第2種 ν 次変形ベッセル関数 $K_\nu(z)$ は変形されたベッセルの微分方程式

$$z^2 \frac{d^2 w}{dz^2} + z \frac{dw}{dz} - (z^2 + \nu^2)w = 0$$

の特殊解であり,

$$K_\nu(z) = \frac{\pi I_{-\nu}(z) - I_\nu(z)}{2 \sin \nu \pi}$$

で定義される. ただし, ν が整数 n に等しい場合は,

$$K_n(z) = \lim_{\nu \rightarrow n} K_\nu(z)$$

とする.

(7) 使用例

- (a) 問題

$K_n(x)$ の値を $n = 5, x = 1.5$ について求める.

- (b) 入力データ

$N = 5, XI = 1.5$

- (c) 主プログラム

```
PROGRAM BIBKNX
! *** EXAMPLE OF DIBKNX ***
IMPLICIT REAL(8) (A-H,O-Z)
READ (5,*) N
READ (5,*) XI
WRITE(6,1000) N,XI
CALL DIBKNX(N,XI,XO,IERR)
WRITE(6,2000) IERR,XO
1000 FORMAT(' ',/,/,5X,'*** DIBKNX ***',/,/,6X,'** INPUT **',&
/,/,8X,'N =',I3,5X,'XI =',F6.2 )
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR =',I5,&
/,/,8X,'VALUE OF KN(X)',/,/,10X,'XO =',D18.10)
END
```

- (d) 出力結果

```
*** DIBKNX ***
** INPUT **
N = 5    XI = 1.50

** OUTPUT**
IERR = 0
VALUE OF KN(X)
XO = 0.4406778116D+02
```

2.4.7 DIBIMX, RIBIMX

第 1 種実数変形ベッセル関数

(1) 機能

整数次の第 1 種変形ベッセル関数

$$I_\nu(x) = \frac{1}{\pi} \int_0^\pi e^{x \cos(t)} \cos(\nu t) dt - \frac{\sin(\pi\nu)}{\pi} \int_0^\infty e^{-x \cosh(t) - \nu t} dt$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL DIBIMX (R, XI, XO, IERR)

単精度サブルーチン:

CALL RIBIMX (R, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: { 32 ビット整数版では INTEGER(4) }
 R:単精度実数型 C:単精度複素数型 { 64 ビット整数版では INTEGER(8) }

項番	引数名	型	大きさ	入出力	内 容
1	R	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	次数 ν
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 x
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	$I_\nu(x)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) R が整数値と一致するとき

$$|R| \leq M_1$$

$$|XI| \leq M_2$$

(b) R が整数値と一致しないとき

$$0 < R \leq M_1$$

$$0 < XI \leq M_2$$

ここで, $M_1 = \{ \text{倍精度: } 2^{31}, \text{ 単精度: } 2^{31} \}$,

$M_2 = \{ \text{倍精度: } 713.067, \text{ 単精度: } 90.978 \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1000	$\nu(\log_e \frac{x}{M_3} - M_3) > M_4$ であった. (注意事項 (e) 参照) ($XI \neq 0.0, R \neq 0.0$) (underflow) 注 ν が整数値と一致するときは $ \nu $, $ x $ に対して判定	XO = 0.0 を返す.
3000	制限条件 (a) または (b) を満足しなかった.	処理を打ち切る.

(6) 注意事項

- (a) $I_\nu(x)$ は, x や ν が大きいほど計算時間がかかる. 一般に $R < 1000.0, XI < 1000.0$ とすることが望ましい.
 (b) 半奇数次の場合は, 球ベッセル関数を利用する方が良い.

$$I_{n+\frac{1}{2}}(x) = \sqrt{\frac{2x}{\pi}} i_n(x)$$

- (c) $\nu < 0$ で ν が整数値と一致しないときは, このサブルーチンでは計算できない. したがって, 漸化式を利用して求める.
 (d) $I_\nu(x), I_{\nu+1}(x), I_{\nu+2}(x) \dots$ を同時に求める場合は, 漸化式を用いて順次求めた方がこのサブルーチンを幾度も使用するよりも速い.
 ただし, ν を増大させていくと不安定となるので, ν が減少する方向へ適用する.

漸化式

$$I_{\nu-1}(x) = \frac{2\nu}{x} I_\nu(x) + I_{\nu+1}(x)$$

- (e) このサブルーチンで IERR=1000 となるときの M_3, M_4 の値は以下の通りである.
 $M_3 = 0.3068, M_4 = \{ \text{倍精度: } 709.7827, \text{単精度: } 88.72284 \}$
 (f) 第 1 種 ν 次変形ベッセル関数 $I_\nu(z)$ は変形されたベッセルの微分方程式

$$z^2 \frac{d^2 w}{dz^2} + z \frac{dw}{dz} - (z^2 + \nu^2) w = 0$$

の特殊解であり,

$$I_{\pm\nu}(z) = e^{\mp\sqrt{-1}\pi/2} J_{\pm\nu}(\sqrt{-1}z)$$

で定義される.

(7) 使用例

(a) 問題

$I_\nu(x)$ の値を $\nu = 3.3$, $x = 1.5$ について求める.

(b) 入力データ

R = 3.3, XI = 1.5

(c) 主プログラム

```

PROGRAM BIBIMX
! *** EXAMPLE OF DIBIMX ***
IMPLICIT REAL(8) (A-H,O-Z)
READ (5,*) R
READ (5,*) XI
WRITE (6,1000) R,XI
CALL DIBIMX(R,XI,XO,IERR)
WRITE (6,2000) IERR,XO
1000 FORMAT(' ',/,/,5X,'*** DIBIMX ***',/,/,6X,'** INPUT **',&
/,/,8X,'R = ',F6.2,5X,'XI = ',F6.2)
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF IM(X)',/,/,10X,'XO = ',D18.10)
END

```

(d) 出力結果

```

*** DIBIMX ***
** INPUT **
R = 3.30    XI = 1.50

** OUTPUT**
IERR =      0
VALUE OF IM(X)
XO = 0.4973088526D-01

```


2.4.8 DIBKMX, RIBKMX 第 2 種実数次変形ベッセル関数

(1) 機能

実数次の第 2 種変形ベッセル関数

$$K_\nu(x) = \int_0^\infty e^{-x \cosh(t)} \cosh(\nu t) dt$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL DIBKMX (R, XI, XO, IERR)

単精度サブルーチン:

CALL RIBKMX (R, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	R	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	次数 ν
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 x
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	$K_\nu(x)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $|R| \leq M_1$

ここで, $M_1 = \{ \text{倍精度: } 2^{31}, \text{ 単精度: } 2^{31} \}$

(b) $XI \geq 0.0$

(c) $XI \leq M_2$

ここで, $M_2 = \{ \text{倍精度: } 705.117, \text{ 単精度: } 85.114 \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
2000	$XI \leq 2.0$ / (最大値) または $\nu(\log_e \frac{\nu}{x} - M_3) > M_4$ であった. (注意事項 (d) 参照) ($XI \neq 0.0$, $R \neq 0.0$) (overflow)	XO =(最大値) を返す.
3000	制限条件 (a), (b) または (c) を満足しなかつた.	処理を打ち切る.

(6) 注意事項

- (a) $K_\nu(x)$ は, x または ν が大きいほど計算時間がかかる. 一般に $R < 1000.0$, $XI < 1000.0$ とすることが望ましい.
- (b) 半奇数次の場合は, 球ベッセル関数を利用する.

$$K_{n+\frac{1}{2}}(x) = \sqrt{\frac{2x}{\pi}} k_n(x)$$

- (c) $K_\nu(x), K_{\nu+1}(x), K_{\nu+2}(x) \cdots$ を同時に求める場合は, 漸化式を用いて順次求めた方が, このサブルーチンを反復使用するよりも速い.

漸化式

$$K_{\nu+1}(x) = \frac{2\nu}{x} K_\nu(x) + K_{\nu-1}(x)$$

- (d) このサブルーチンで IERR=2000 となるときの M_3, M_4 の値は以下の通りである.

$$M_3 = 0.3068, M_4 = \{ \text{倍精度: } 709.7827, \text{単精度: } 88.72284 \}$$

- (e) 第 2 種 ν 次変形ベッセル関数 $K_\nu(z)$ は変形されたベッセルの微分方程式

$$z^2 \frac{d^2 w}{dz^2} + z \frac{dw}{dz} - (z^2 + \nu^2) w = 0$$

の特殊解であり,

$$K_\nu(z) = \frac{\pi}{2} \frac{I_{-\nu}(z) - I_\nu(z)}{\sin \nu\pi}$$

で定義される. ただし, ν が整数 n に等しい場合は,

$$K_n(z) = \lim_{\nu \rightarrow n} K_\nu(z)$$

とする.

(7) 使用例

(a) 問題

$K_\nu(x)$ の値を $\nu = 3.3, x = 1.5$ について求める.

(b) 入力データ

R = 3.3, XI = 1.5

(c) 主プログラム

```
PROGRAM BIBKMX
! *** EXAMPLE OF DIBKMX ***
IMPLICIT REAL(8) (A-H,O-Z)
READ (5,*) R
READ (5,*) XI
WRITE (6,1000) R,XI
CALL DIBKMX(R,XI,XO,IERR)
WRITE (6,2000) IERR,XO
1000 FORMAT(' ',/,/,5X,'*** DIBKMX ***',/,/,6X,'** INPUT **',&
/,/,8X,'R = ',F6.2,5X,'XI = ',F6.2)
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF KM(X)',/,/,10X,'XO = ',D18.10)
END
```

(d) 出力結果

```
*** DIBKMX ***
** INPUT **
R = 3.30    XI = 1.50

** OUTPUT**
IERR =      0
VALUE OF KM(X)
XO = 0.2759863620D+01
```

2.4.9 ZIBINZ, CIBINZ

複素変数第 1 種整数次変形ベッセル関数

(1) 機能

複素変数で整数次の第 1 種変形ベッセル関数

$$I_n(z) = \frac{1}{\pi} \int_0^\pi e^{z \cos(t)} \cos(nt) dt$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL ZIBINZ (N, ZI, ZO, IERR)

単精度サブルーチン:

CALL CIBINZ (N, ZI, ZO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入 力	次数 n
2	ZI	$\begin{Bmatrix} Z \\ C \end{Bmatrix}$	1	入 力	変数値 z
3	ZO	$\begin{Bmatrix} Z \\ C \end{Bmatrix}$	1	出 力	$I_n(z)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $|ZI| \leq M_1$

ここで, $M_1 = \{ \text{倍精度: } 2^{50}\pi, \text{ 単精度: } 2^{18}\pi \}$

(b) $|\Re(ZI)| \leq M_2$

ここで, $M_2 = \{ \text{倍精度: } 709.7827, \text{ 単精度: } 88.72284 \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1000	$ n (\log_e \frac{ n }{ z } - M_3) > M_4$ (注意事項 (c) 参照) ($ ZI \neq 0.0, N \neq 0$) (underflow)	ZO = (0.0, 0.0) を返す.
3000	制限条件 (a) または (b) を満足しなかった.	処理を打ち切る.

(6) 注意事項

(a) $I_n(z)$ は, $|z|$ や n が大きいほど計算時間がかかる. 一般に $|N| < 1000, |ZI| < 1000.0$ とすることが望ましい.

- (b) $I_n(z), I_{n+1}(z), I_{n+2}(z) \dots$ を同時に求める場合は、漸化式を用いて順次求めた方が、このサブルーチンを幾度も使用するよりも速い。

ただし、 n を増大させていくと不安定になるので、 n が減少する方向へ適用する。

漸化式

$$I_{n-1}(z) = \frac{2n}{z} I_n(z) + I_{n+1}(z)$$

- (c) このサブルーチンで IERR=1000 となるときの M_3, M_4 の値は以下の通りである。

$M_3 = 0.3068, M_4 = \{ \text{倍精度: } 709.7827, \text{単精度: } 88.72284 \}$

- (d) 第 1 種 ν 次変形ベッセル関数 $I_\nu(z)$ は変形されたベッセルの微分方程式

$$z^2 \frac{d^2 w}{dz^2} + z \frac{dw}{dz} - (z^2 + \nu^2) w = 0$$

の特殊解であり、

$$I_{\pm\nu}(z) = e^{\mp\sqrt{-1}\pi/2} J_{\pm\nu}(\sqrt{-1}z)$$

で定義される。

(7) 使用例

- (a) 問題

$I_n(z)$ の値を $n = 3, z = 1 + 2\sqrt{-1}$ について求める。

- (b) 入力データ

$N = 3, ZI = (1.0, 2.0)$

- (c) 主プログラム

```
PROGRAM AIBINZ
! *** EXAMPLE OF ZIBINZ ***
IMPLICIT COMPLEX(8) (A-H,O-Z)
READ (5,*) N
READ (5, '(D6.1,D6.1)') ZI
WRITE(6,1000) N,ZI
CALL ZIBINZ(N,ZI,ZO,IERR)
WRITE(6,2000) IERR,ZO
1000 FORMAT(' ',/,/,5X,'*** ZIBINZ ***',/,/,6X,'** INPUT **',&
/,/,8X,'N = ',I3,5X,'ZI = (',F6.2,', ',F6.2,', )')
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF IN(Z)',/,/,10X,'ZO = (',D18.10,', ',D18.10,', )')
END
```

- (d) 出力結果

```
*** ZIBINZ ***
** INPUT **
N = 3    ZI = ( 1.00 , 2.00 )

** OUTPUT**
IERR =    0
VALUE OF IN(Z)
ZO = ( -0.1753534440D+00 , -0.8243079895D-01 )
```

2.4.10 ZIBKNZ, CIBKNZ

複素変数第 2 種整数次変形ベッセル関数

(1) 機能

複素変数で整数次の第 2 種変形ベッセル関数

$$K_n(z) = \int_0^{\infty} e^{-z \cosh(t)} \cosh(nt) dt$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL ZIBKNZ (N, ZI, ZO, IERR)

単精度サブルーチン:

CALL CIBKNZ (N, ZI, ZO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入 力	次数 n
2	ZI	$\begin{Bmatrix} Z \\ C \end{Bmatrix}$	1	入 力	変数値 z
3	ZO	$\begin{Bmatrix} Z \\ C \end{Bmatrix}$	1	出 力	$K_n(z)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $|ZI| > 0.0$

(b) $|ZI| \leq M_1$

ここで, $M_1 = \{ \text{倍精度: } 2^{50}\pi, \text{ 単精度: } 2^{18}\pi \}$

(c) $|\Re(ZI)| \leq M_2$

ここで, $M_2 = \{ \text{倍精度: } 709.7827, \text{ 単精度: } 88.72284 \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a), (b) または (c) を満足しなかつた.	処理を打ち切る.
4000	$ ZI \leq 2.0/(\text{最大値})$ または $ n (\log_e \frac{ n }{ z } - M_3) > M_4$ (注意事項 (c) 参照) ($ ZI \neq 0.0, N \neq 0$)	

(6) 注意事項

(a) $K_n(z)$ は, $|z|$ や n が大きいほど計算時間がかかる. 一般に $|N| < 1000, |ZI| < 1000.0$ とすることが望ましい.

(b) $K_n(z), K_{n+1}(z), K_{n+2}(z) \dots$ を同時に求める場合は, 漸化式を用いて順次求めた方がこのサブルーチンを幾度も使用するよりも速い.

漸化式

$$K_{n+1}(z) = \frac{2n}{z} K_n(z) + K_{n-1}(z)$$

(c) このサブルーチンで IERR=4000 となるときの M_3, M_4 の値は以下の通りである.

$$M_3 = 0.3068, M_4 = \{ \text{倍精度: } 709.7827, \text{単精度: } 88.72284 \}$$

(d) 第 2 種 ν 次変形ベッセル関数 $K_\nu(z)$ は変形されたベッセルの微分方程式

$$z^2 \frac{d^2 w}{dz^2} + z \frac{dw}{dz} - (z^2 + \nu^2) w = 0$$

の特殊解であり,

$$K_\nu(z) = \frac{\pi}{2} \frac{I_{-\nu}(z) - I_\nu(z)}{\sin \nu\pi}$$

で定義される. ただし, ν が整数 n に等しい場合は,

$$K_n(z) = \lim_{\nu \rightarrow n} K_\nu(z)$$

とする.

(7) 使用例

(a) 問題

$K_n(z)$ の値を $n = 3$, $z = 1 + 2\sqrt{-1}$ について求める.

(b) 入力データ

$N = 3$, $ZI = (1.0, 2.0)$

(c) 主プログラム

```

PROGRAM AIBKNZ
! *** EXAMPLE OF ZIBKNZ ***
IMPLICIT COMPLEX(8) (A-H,O-Z)
READ (5,*) N
READ (5,'(D6.1,D6.1)') ZI
WRITE (6,1000) N,ZI
CALL ZIBKNZ(N,ZI,ZO,IERR)
WRITE (6,2000) IERR,ZO
1000 FORMAT(' ',/,/,5X,'*** ZIBKNZ ***',/,/,6X,'** INPUT **',&
/,/,8X,'N =',I3,5X,'ZI = (',F6.2,',',F6.2,')')
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF KN(Z)',/,/,10X,'ZO = (',D18.10,',',D18.10,')')
END

```

(d) 出力結果

```

*** ZIBKNZ ***
** INPUT **
N = 3      ZI = ( 1.00 , 2.00 )

** OUTPUT**
IERR = 0
VALUE OF KN(Z)
ZO = ( -0.6814364280D+00 , 0.6251546546D+00 )

```


2.5 球ベッセル関数

2.5.1 DIBSJNI, RIBSJNI

第 1 種整数次球ベッセル関数

(1) 機能

整数次の第 1 種球ベッセル関数

$$j_n(x) = \sqrt{\frac{\pi}{2x}} J_{n+\frac{1}{2}}(x)$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL DIBSJNI (N, XI, XO, IERR)

単精度サブルーチン:

CALL RIBSJNI (N, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: { 32 ビット整数版では INTEGER(4) }
 R:単精度実数型 C:単精度複素数型 { 64 ビット整数版では INTEGER(8) }

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入力	次数 n
2	XI	{ D } { R }	1	入力	変数値 x
3	XO	{ D } { R }	1	出力	$j_n(x)$ の値
4	IERR	I	1	出力	エラーインディケータ

(4) 制限条件

(a) $XI \geq 0.0$

(b) $XI \leq M$

ここで, $M = \{ \text{倍精度: } 2^{50}\pi, \text{単精度: } 2^{18}\pi \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1000	$ n (\log_e \frac{ n }{x} - M_1) > M_2$ であった. (注意事項 (c) 参照) ($XI \neq 0.0, N \neq 0$) (underflow or overflow)	$N \geq 0$ のとき $XO = 0.0$ $N < 0$ のとき $XO = (-1)^{N+1} \times (\text{最大値})$.
3000	制限条件 (a) または (b) を満足しなかった.	処理を打ち切る.

(6) 注意事項

- (a) $j_n(x)$ は x や n が大きいほど計算時間がかかる。一般に $|N| < 1000$, $XI < 1000.0$ とすることが望ましい。
- (b) $j_n(x), j_{n+1}(x), j_{n+2}(x) \dots$ を同時に求める場合は、漸化式を用いて順次求めた方が、このサブルーチンを幾度も使用するよりも速い。

ただし、 n を増大させていくと不安定となるので、 n が減少する方向へ適用する。

漸化式

$$j_{n-1}(x) = \frac{2n+1}{x} j_n(x) - j_{n+1}(x)$$

- (c) このサブルーチンで IERR=1000 となるときの M_1, M_2 の値は以下の通りである。

$M_1 = 0.3068, M_2 = \{ \text{倍精度: } 709.7827, \text{単精度: } 88.72284 \}$

- (d) 第 1 種 n 次球ベッセル関数 $j_n(z)$ は微分方程式

$$z^2 \frac{d^2 w}{dz^2} + 2z \frac{dw}{dz} + \{z^2 - n(n+1)\} w = 0 \quad (n = 0, \pm 1, \pm 2, \dots)$$

の特殊解であり、

$$j_n(z) = \sqrt{\frac{\pi}{2z}} J_{n+\frac{1}{2}}(z)$$

で定義される。

(7) 使用例

- (a) 問題

$j_n(x)$ の値を $n = 5, x = 1.5$ について求める。

- (b) 入力データ

$N = 5, XI = 1.5$

- (c) 主プログラム

```

PROGRAM BIBSJM
! *** EXAMPLE OF DIBSJM ***
IMPLICIT REAL(8) (A-H,O-Z)
READ (5,*) N
READ (5,*) XI
WRITE(6,1000) N,XI
CALL DIBSJM(N,XI,XO,IERR)
WRITE(6,2000) IERR,XO
1000 FORMAT(' ',/,/,5X,'*** DIBSJM ***',/,/,6X,'** INPUT **',&
/,/,8X,'N = ',I3,5X,'XI = ',F6.2 )
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF SPHERICAL JN(X)',/,/,10X,'XO = ',D18.10)
END

```

- (d) 出力結果

```

*** DIBSJM ***
** INPUT **
N = 5    XI = 1.50

** OUTPUT**
IERR = 0
VALUE OF SPHERICAL JN(X)
XO = 0.6696205963D-03

```

2.5.2 DIBSYN, RIBSYN 第 2 種整数次球ベッセル関数

(1) 機能

整数次の第 2 種球ベッセル関数

$$y_n(x) = \sqrt{\frac{\pi}{2x}} Y_{n+\frac{1}{2}}(x)$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL DIBSYN (N, XI, XO, IERR)

単精度サブルーチン:

CALL RIBSYN (N, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入 力	次数 n
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 x
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	$y_n(x)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $XI \geq 0.0$

(b) $XI \leq M$

ここで, $M = \{ \text{倍精度: } 2^{50}\pi, \text{単精度: } 2^{18}\pi \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
2000	$XI \leq 2.0 / (\text{最大値})$ または $ n (\log_e \frac{ n }{x} - M_1) > M_2$ (注意事項 (c) 参照) ($XI \neq 0.0, N \neq 0$) (overflow or underflow)	$N \geq 0$ のとき $XO = (\text{最小値})$ $N < 0$ のとき $XO = 0.0$
3000	制限条件 (a) または (b) を満足しなかった.	処理を打ち切る.

(6) 注意事項

- (a) $y_n(x)$ は, x や n が大きいほど計算時間がかかる. 一般に $|N| < 1000$, $XI < 1000.0$ とすることが望ましい.
- (b) $y_n(x), y_{n+1}(x), y_{n+2}(x) \cdots$ を同時に求める場合は, 漸化式を用いて順次求めた方が, このサブルーチンを幾度も使用するよりも速い.

漸化式

$$y_{n+1}(x) = \frac{2n+1}{x} y_n(x) - y_{n-1}(x)$$

- (c) このサブルーチンで IERR=2000 となるときの M_1, M_2 の値は以下の通りである.

$$M_1 = 0.3068, M_2 = \{ \text{倍精度: } 709.7827, \text{単精度: } 88.72284 \}$$

- (d) 第 2 種 n 次球ベッセル関数 $y_n(z)$ は微分方程式

$$z^2 \frac{d^2 w}{dz^2} + 2z \frac{dw}{dz} + \{z^2 - n(n+1)\} w = 0 \quad (n = 0, \pm 1, \pm 2, \cdots)$$

の特殊解であり,

$$y_n(z) = \sqrt{\frac{\pi}{2z}} Y_{n+\frac{1}{2}}(z)$$

で定義される.

- (e) 球ノイマン関数 $n_n(z)$ は, 第 2 種球ベッセル関数 $y_n(z)$ と同じものである.

(7) 使用例

- (a) 問題

$y_n(x)$ の値を $n = 5, x = 1.5$ について求める.

- (b) 入力データ

$N = 5, XI = 1.5$

- (c) 主プログラム

```

PROGRAM BIBSYN
! *** EXAMPLE OF DIBSYN ***
IMPLICIT REAL(8) (A-H,O-Z)
READ (5,*) N
READ (5,*) XI
WRITE (6,1000) N,XI
CALL DIBSYN(N,XI,XO,IERR)
WRITE (6,2000) IERR,XO
1000 FORMAT(' ',/,/,5X,'*** DIBSYN ***',/,/,6X,'** INPUT **',&
/,/,8X,'N =',I3,5X,'XI =',F6.2)
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF SPHERICAL YN(X)',/,/,10X,'XO = ',D18.10)
END

```

- (d) 出力結果

```

*** DIBSYN ***
** INPUT **
N = 5    XI = 1.50

** OUTPUT**
IERR = 0
VALUE OF SPHERICAL YN(X)
XO = -0.9423611009D+02

```

2.5.3 DIBSIN, RIBSIN 第 1 種整数次変形球ベッセル関数

(1) 機能

整数次の第 1 種変形球ベッセル関数

$$i_n(x) = \sqrt{\frac{\pi}{2x}} I_{n+\frac{1}{2}}(x)$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL DIBSIN (N, XI, XO, IERR)

単精度サブルーチン:

CALL RIBSIN (N, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入 力	次数 n
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 x
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	$i_n(x)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $N \geq 0$

(b) $XI \geq 0.0$

(c) $XI \leq M$

ここで, $M = \{ \text{倍精度: } 709.7827, \text{単精度: } 88.72284 \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1000	$n(\log_e \frac{n}{x} - M_1) > M_2$ であった. (注意事項 (d) 参照) ($XI \neq 0.0, N \neq 0$) (underflow)	XO = 0.0 を返す.
3000	制限条件 (a), (b) または (c) を満足しなかった.	処理を打ち切る.

(6) 注意事項

- (a) $i_n(x)$ は, x や n が大きいほど計算時間がかかる. 一般に $N < 1000, XI < 1000.0$ とすることが望ましい.
- (b) $n < 0$ のときは, このサブルーチンでは計算できない. 従って, 漸化式を利用して求める.
- (c) $i_n(x), i_{n+1}(x), i_{n+2}(x) \dots$ を同時に求める場合は, 漸化式を用いて順次求めた方が, このサブルーチンを幾度も使用するよりも速い.
ただし, n を増大させていくと不安定になるので, n が減少する方向へ適用する.
漸化式

$$i_{n-1}(x) = \frac{2n+1}{x} i_n(x) + i_{n+1}(x)$$

- (d) このサブルーチンで IERR=1000 となるときの M_1, M_2 の値は以下の通りである.
 $M_1 = 0.3068, M_2 = \{ \text{倍精度: } 709.7827, \text{単精度: } 88.72284 \}$
- (e) 第1種 n 次変形球ベッセル関数 $i_n(z)$ は微分方程式

$$z^2 \frac{d^2 w}{dz^2} + 2z \frac{dw}{dz} - \{z^2 + n(n+1)\} w = 0 \quad (n = 0, \pm 1, \pm 2, \dots)$$

の特殊解であり,

$$i_n(z) = \sqrt{\frac{\pi}{2z}} J_{n+\frac{1}{2}}(z)$$

で定義される.

(7) 使用例

(a) 問題

$i_n(x)$ の値を $n = 5, x = 1.5$ について求める.

(b) 入力データ

$N = 5, XI = 1.5$

(c) 主プログラム

```
PROGRAM BIBSIN
! *** EXAMPLE OF DIBSIN ***
IMPLICIT REAL(8) (A-H,O-Z)
READ (5,*) N
READ (5,*) XI
WRITE (6,1000) N,XI
CALL DIBSIN(N,XI,XO,IERR)
WRITE (6,2000) IERR,XO
1000 FORMAT(' ',/,/,5X,'*** DIBSIN ***',/,/,6X,'** INPUT **',&
/,/,8X,'N =',I3,5X,'XI =',F6.2 )
2000 FORMAT(' ',/,/,8X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF SPHERICAL IN(X)',/,/,10X,'XO = ',D18.10)
END
```

(d) 出力結果

```
*** DIBSIN ***
** INPUT **
N = 5      XI = 1.50

** OUTPUT**
IERR =     0
VALUE OF SPHERICAL IN(X)
XO = 0.7961612655D-03
```

2.5.4 DIBSKN, RIBSKN 第 2 種整数次変形球ベッセル関数

(1) 機能

整数次の第 2 種変形球ベッセル関数

$$k_n(x) = \sqrt{\frac{\pi}{2x}} K_{n+\frac{1}{2}}(x)$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL DIBSKN (N, XI, XO, IERR)

単精度サブルーチン:

CALL RIBSKN (N, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER(4)} \\ 64 \text{ ビット整数版では INTEGER(8)} \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入 力	次数 n
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 x
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	$k_n(x)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $XI \geq 0.0$

(b) $XI \leq M$

ここで, $M = \{ \text{倍精度: } 702.293, \text{単精度: } 83.364 \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
2000	$XI \leq 2.0/$ (最大値) または $ n (\log_e \frac{ n }{x} - M_1) > M_2$ (注意事項 (c) 参照) ($XI \neq 0.0, N \neq 0$) (overflow)	$XO =$ (最大値) を返す.
3000	制限条件 (a) または (b) を満足しなかった.	処理を打ち切る.

(6) 注意事項

- (a) $k_n(x)$ は, x や n が大きいほど計算時間がかかる. 一般に $|N| < 1000$, $XI < 1000.0$ とすることが望ましい.
- (b) $k_n(x), k_{n+1}(x), k_{n+2}(x) \dots$ を同時に求める場合は, 漸化式を用いて順次求めた方が, このサブルーチンを幾度も使用するよりも速い.

漸化式

$$k_{n+1}(x) = \frac{2n+1}{x} k_n(x) + k_{n-1}(x)$$

- (c) このサブルーチンで IERR=2000 となるときの M_1, M_2 の値は以下の通りである.

$$M_1 = 0.3068, M_2 = \{ \text{倍精度: } 709.7827, \text{単精度: } 88.72284 \}$$

- (d) 第2種 n 次変形球ベッセル関数 $k_n(z)$ は微分方程式

$$z^2 \frac{d^2 w}{dz^2} + 2z \frac{dw}{dz} - \{z^2 + n(n+1)\} w = 0 \quad (n = 0, \pm 1, \pm 2, \dots)$$

の特殊解であり,

$$k_n(z) = \sqrt{\frac{\pi}{2z}} K_{n+\frac{1}{2}}(z)$$

で定義される.

(7) 使用例

- (a) 問題

$k_n(x)$ の値を $n = 5, x = 1.5$ について求める.

- (b) 入力データ

$N = 5, XI = 1.5$

- (c) 主プログラム

```
PROGRAM BIBSKN
! *** EXAMPLE OF DIBSKN ***
IMPLICIT REAL(8) (A-H,O-Z)
READ (5,*) N
READ (5,*) XI
WRITE (6,1000) N,XI
CALL DIBSKN(N,XI,XO,IERR)
WRITE (6,2000) IERR,XO
1000 FORMAT(' ',/,/,5X,'*** DIBSKN ***',/,/,6X,'** INPUT **',&
/,/,8X,'N =',I3,5X,'XI =',F6.2 )
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF SPHERICAL KN(X)',/,/,10X,'XO = ',D18.10)
END
```

- (d) 出力結果

```
*** DIBSKN ***
** INPUT **
   N = 5      XI = 1.50

** OUTPUT**
IERR =      0
VALUE OF SPHERICAL KN(X)
   XO = 0.1152469739D+03
```


2.6 ベッセル関数に関連した関数

2.6.1 ZIBH1N, CIBH1N

第1種ハンケル関数

(1) 機能

第1種整数次ハンケル関数

$$H_n^{(1)}(z) = -\frac{2\sqrt{-1}}{\pi} e^{-\sqrt{-1}n\pi/2} \int_0^\infty e^{\sqrt{-1}z \cosh(t)} \cosh(nt) dt \quad (0 < \arg z < \pi)$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL ZIBH1N (N, ZI, ZO, IERR)

単精度サブルーチン:

CALL CIBH1N (N, ZI, ZO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER(4)} \\ 64 \text{ ビット整数版では INTEGER(8)} \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入 力	次数 n
2	ZI	$\begin{cases} Z \\ C \end{cases}$	1	入 力	変数値 z
3	ZO	$\begin{cases} Z \\ C \end{cases}$	1	出 力	$H_n^{(1)}(z)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $|ZI| > 0.0$

(b) $|\Im(ZI)| \leq M_1$

ここで, $M_1 = \{ \text{倍精度: } 709.7827, \text{単精度: } 88.72284 \}$

(c) $|ZI| \leq M_2$

ここで, $M_2 = \{ \text{倍精度: } 2^{50}\pi, \text{単精度: } 2^{18}\pi \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a), (b) または (c) を満足しなかつた.	処理を打ち切る.
4000	$ ZI \leq 2.0 /$ (最大値) または $ n (\log_e \frac{ n }{ z } - M_3) > M_4$ (注意事項 (a) 参照)	

(6) 注意事項

(a) このサブルーチンで IERR=4000 となるときの M_3, M_4 の値は以下の通りである.

$$M_3 = 0.3068, M_4 = \{ \text{倍精度: } 709.7827, \text{単精度: } 88.72284 \}$$

(b) 第1種 ν 次ハンケル関数 $H_\nu^{(1)}(z)$ はベッセルの微分方程式

$$z^2 \frac{d^2 w}{dz^2} + z \frac{dw}{dz} + (z^2 - \nu^2)w = 0$$

の特殊解であり,

$$H_\nu^{(1)}(z) = -\frac{1}{\pi} \int_{L_1} e^{-\sqrt{-1}z \sin \tau + \sqrt{-1}\nu \tau} d\tau$$

で定義される. ここで, 積分路 L_1 は $(0, -\infty) \rightarrow (0, 0) \rightarrow (-\pi, 0) \rightarrow (-\pi, \infty)$ ととる.

(c) 第1種ならびに第2種ハンケル関数は第3種ベッセル関数(第3種円柱関数)とも呼ばれる.

(7) 使用例

(a) 問題

$H_n^{(1)}(z)$ の値を $n = 3, z = 1 + 2\sqrt{-1}$ について求める.

(b) 入力データ

$$N = 3, ZI = (1.0, 2.0)$$

(c) 主プログラム

```

PROGRAM AIBH1N
! *** EXAMPLE OF ZIBH1N ***
IMPLICIT COMPLEX(8) (A-H,O-Z)
READ (5,*) N
READ (5,'(D6.1,D6.1)') ZI
WRITE (6,1000) N,ZI
CALL ZIBH1N(N,ZI,ZO,IERR)
WRITE (6,2000) IERR,ZO
1000 FORMAT(' ',/,/,5X,'*** ZIBH1N ***',/,/,6X,'** INPUT **',&
/,/,8X,'N =',I3,5X,'ZI = (',F6.2,',',F6.2,',')')
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF H1N(Z)',&
/,/,10X,'ZO = (',D18.10,',',D18.10,',')')
END

```

(d) 出力結果

```

*** ZIBH1N ***
** INPUT **
N = 3    ZI = ( 1.00 , 2.00 )

** OUTPUT**
IERR =    0
VALUE OF H1N(Z)
ZO = ( -0.6892089637D-01 , 0.3073283562D+00 )

```

2.6.2 ZIBH2N, CIBH2N 第2種ハンケル関数

(1) 機能

第2種整数次ハンケル関数

$$H_n^{(2)}(z) = \frac{2\sqrt{-1}}{\pi} e^{\sqrt{-1}n\pi/2} \int_0^\infty e^{-\sqrt{-1}z \cosh(t)} \cosh(nt) dt \quad (0 < \arg z < \pi)$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL ZIBH2N (N, ZI, ZO, IERR)

単精度サブルーチン:

CALL CIBH2N (N, ZI, ZO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER(4)} \\ 64 \text{ ビット整数版では INTEGER(8)} \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入 力	次数 n
2	ZI	$\begin{Bmatrix} Z \\ C \end{Bmatrix}$	1	入 力	変数値 z
3	ZO	$\begin{Bmatrix} Z \\ C \end{Bmatrix}$	1	出 力	$H_n^{(2)}(z)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $|ZI| > 0.0$

(b) $|\Im(ZI)| \leq M_1$

ここで, $M_1 = \{ \text{倍精度: } 709.7827, \text{単精度: } 88.72284 \}$

(c) $|ZI| \leq M_2$

ここで, $M_2 = \{ \text{倍精度: } 2^{50}\pi, \text{単精度: } 2^{18}\pi \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a), (b) または (c) を満足しなかつた.	処理を打ち切る.
4000	$ ZI \leq 2.0 /$ (最大値) または $ n (\log_e \frac{ n }{ z } - M_3) > M_4$ (注意事項 (a) 参照)	

(6) 注意事項

(a) このサブルーチンで IERR=4000 となるときの M_3, M_4 の値は以下の通りである.

$$M_3 = 0.3068, M_4 = \{ \text{倍精度: } 709.7827, \text{単精度: } 88.72284 \}$$

(b) 第2種 ν 次ハンケル関数 $H_\nu^{(2)}(z)$ はベッセルの微分方程式

$$z^2 \frac{d^2 w}{dz^2} + z \frac{dw}{dz} + (z^2 - \nu^2)w = 0$$

の特殊解であり,

$$H_\nu^{(2)}(z) = -\frac{1}{\pi} \int_{L_2} e^{-\sqrt{-1}z \sin \tau + \sqrt{-1}\nu \tau} d\tau$$

で定義される. ここで, 積分路 L_2 は $(\pi, \infty) \rightarrow (\pi, 0) \rightarrow (0, 0) \rightarrow (0, -\infty)$ ととる.

(c) 第1種ならびに第2種ハンケル関数は第3種ベッセル関数 (第3種円柱関数) とも呼ばれる.

(7) 使用例

(a) 問題

$H_n^{(2)}(z)$ の値を $n = 3, z = 1 + 2\sqrt{-1}$ について求める.

(b) 入力データ

$$N = 3, ZI = (1.0, 2.0)$$

(c) 主プログラム

```

PROGRAM AIBH2N
! *** EXAMPLE OF ZIBH2N ***
IMPLICIT COMPLEX(8) (A-H,O-Z)
READ (5,*) N
READ (5,'(D6.1,D6.1)') ZI
WRITE(6,1000) N,ZI
CALL ZIBH2N(N,ZI,ZO,IERR)
WRITE(6,2000) IERR,ZO
1000 FORMAT(' ',/,/,5X,'*** ZIBH2N ***',/,/,6X,'** INPUT **',&
/,/,8X,'N =',I3,5X,'ZI = (',F6.2,',',F6.2,',')')
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF H2N(Z)',&
/,/,10X,'ZO = (',D18.10,',',D18.10,',')')
END

```

(d) 出力結果

```

*** ZIBH2N ***
** INPUT **
N = 3    ZI = ( 1.00 , 2.00 )

** OUTPUT**
IERR =    0
VALUE OF H2N(Z)
ZO = ( -0.4931584373D+00 , -0.2729782322D+00 )

```

2.6.3 DIBBER, RIBBER

ケルビン関数 $\text{ber}_n(x)$

(1) 機能

ケルビン関数

$$\text{ber}_n(x) = \Re\{J_n(xe^{3\sqrt{-1}\pi/4})\}$$

の値を求める.

(2) 使用法

倍精度サブルーチン:

CALL DIBBER (N, XI, XO, IERR)

単精度サブルーチン:

CALL RIBBER (N, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入 力	次数 n
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 x
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	$\text{ber}_n(x)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $|XI| \leq M$

ここで, $M = \{ \text{倍精度: } 1003.784, \text{ 単精度: } 125.473 \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1000	$ n (\log_e \frac{ n }{ x } - M_1) > M_2$ であった. (注意事項 (a) 参照) (underflow)	XO = 0.0 を返す.
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.

(6) 注意事項

(a) このサブルーチンで IERR=1000 となるときの M_1, M_2 の値は以下の通りである.

$$M_1 = 0.3068, M_2 = \{ \text{倍精度: } 709.7827, \text{単精度: } 88.72284 \}$$

(b) $w = \text{ber}_\nu(x) + \sqrt{-1} \text{bei}_\nu(x), \text{ber}_{-\nu}(x) + \sqrt{-1} \text{bei}_{-\nu}(x), \text{ker}_\nu(x) + \sqrt{-1} \text{kei}_\nu(x), \text{ker}_{-\nu}(x) + \sqrt{-1} \text{kei}_{-\nu}(x)$ は微分方程式

$$x^2 \frac{d^2 w}{dx^2} + x \frac{dw}{dx} - (\sqrt{-1} x^2 + \nu^2) w = 0$$

の解である.

(7) 使用例

(a) 問題

$\text{ber}_n(x)$ の値を $n = 3, x = 1.0$ について求める.

(b) 入力データ

N = 3, XI = 1.0

(c) 主プログラム

```
PROGRAM BIBBER
! *** EXAMPLE OF DIBBER ***
IMPLICIT REAL(8) (A-H,O-Z)
READ (5,*) N
READ (5,*) XI
WRITE (6,1000) N,XI
CALL DIBBER(N,XI,XO,IERR)
WRITE (6,2000) IERR,XO
1000 FORMAT(' ',/,/,5X,'*** DIBBER ***',/,/,6X,'** INPUT **',&
/,/,8X,'N = ',I3,5X,'XI = ',F6.2 )
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF BERNX',/,/,10X,'XO = ',D18.10)
END
```

(d) 出力結果

```
*** DIBBER ***
** INPUT **
N = 3    XI = 1.00

** OUTPUT**
IERR = 0
VALUE OF BERNX
XO = 0.1378798405D-01
```

2.6.4 DIBBEI, RIBBEI

ケルビン関数 $\text{bei}_n(x)$

(1) 機能

ケルビン関数

$$\text{bei}_n(x) = \Im\{J_n(xe^{3\sqrt{-1}\pi/4})\}$$

の値を求める.

(2) 使用法

倍精度サブルーチン:

CALL DIBBEI (N, XI, XO, IERR)

単精度サブルーチン:

CALL RIBBEI (N, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入 力	次数 n
2	XI	$\begin{cases} D \\ R \end{cases}$	1	入 力	変数値 x
3	XO	$\begin{cases} D \\ R \end{cases}$	1	出 力	$\text{bei}_n(x)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $|XI| \leq M$

ここで, $M = \{ \text{倍精度: } 1003.784, \text{ 単精度: } 125.473 \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1000	$ n (\log_e \frac{ n }{ x } - M_1) > M_2$ であった. (注意事項 (a) 参照) (underflow)	XO = 0.0 を返す.
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.

(6) 注意事項

(a) このサブルーチンで IERR=1000 となるときの M_1, M_2 の値は以下の通りである.

$$M_1 = 0.3068, M_2 = \{ \text{倍精度: } 709.7827, \text{単精度: } 88.72284 \}$$

(b) $w = \text{ber}_\nu(x) + \sqrt{-1} \text{bei}_\nu(x), \text{ber}_{-\nu}(x) + \sqrt{-1} \text{bei}_{-\nu}(x), \text{ker}_\nu(x) + \sqrt{-1} \text{kei}_\nu(x), \text{ker}_{-\nu}(x) + \sqrt{-1} \text{kei}_{-\nu}(x)$ は微分方程式

$$x^2 \frac{d^2 w}{dx^2} + x \frac{dw}{dx} - (\sqrt{-1}x^2 + \nu^2)w = 0$$

の解である.

(7) 使用例

(a) 問題

$\text{bei}_n(x)$ の値を $n = 3, x = 1.0$ について求める.

(b) 入力データ

N = 3, XI = 1.0

(c) 主プログラム

```
PROGRAM BIBBEI
! *** EXAMPLE OF DIBBEI ***
IMPLICIT REAL(8) (A-H,O-Z)
READ (5,*) N
READ (5,*) XI
WRITE (6,1000) N,XI
CALL DIBBEI(N,XI,XO,IERR)
WRITE (6,2000) IERR,XO
1000 FORMAT(' ',/,/,5X,'*** DIBBEI ***',/,/,6X,'** INPUT **',&
/,/,8X,'N = ',I3,5X,'XI = ',F6.2 )
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF BEINX',/,/,10X,'XO = ',D18.10)
END
```

(d) 出力結果

```
*** DIBBEI ***
** INPUT **
N = 3    XI = 1.00

** OUTPUT**
IERR = 0
VALUE OF BEINX
XO = 0.1562876861D-01
```


2.6.5 DIBKER, RIBKER

ケルビン関数 $\ker_n(x)$

(1) 機能

ケルビン関数

$$\ker_n(x) = \Re\{e^{-\sqrt{-1}n\pi/2} K_n(xe^{\sqrt{-1}\pi/4})\}$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL DIBKER (N, XI, XO, IERR)

単精度サブルーチン:

CALL RIBKER (N, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER(4)} \\ 64 \text{ ビット整数版では INTEGER(8)} \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入 力	次数 n
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 x
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	$\ker_n(x)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $0.0 < XI \leq M$

ここで, $M = \{ \text{倍精度: } 1003.784, \text{ 単精度: } 125.473 \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
4000	$XI \leq 2.0$ / (最大値) または $ n (\log_e \frac{ n }{x} - M_1) > M_2$ であった. (注意事項 (a) 参照)	

(6) 注意事項

(a) このサブルーチンで IERR=4000 となるときの M_1, M_2 の値は以下の通りである.

$$M_1 = 0.3068, M_2 = \{ \text{倍精度: } 709.7827, \text{単精度: } 88.72284 \}$$

(b) $w = \text{ber}_\nu(x) + \sqrt{-1} \text{bei}_\nu(x), \text{ber}_{-\nu}(x) + \sqrt{-1} \text{bei}_{-\nu}(x), \text{ker}_\nu(x) + \sqrt{-1} \text{kei}_\nu(x), \text{ker}_{-\nu}(x) + \sqrt{-1} \text{kei}_{-\nu}(x)$ は微分方程式

$$x^2 \frac{d^2 w}{dx^2} + x \frac{dw}{dx} - (\sqrt{-1} x^2 + \nu^2) w = 0$$

の解である.

(7) 使用例

(a) 問題

$\ker_n(x)$ の値を $n = 3, x = 1.0$ について求める.

(b) 入力データ

$N = 3, XI = 1.0$

(c) 主プログラム

```
PROGRAM BIBKER
! *** EXAMPLE OF DIBKER ***
IMPLICIT REAL(8) (A-H,O-Z)
READ (5,*) N
READ (5,*) XI
WRITE (6,1000) N,XI
CALL DIBKER(N,XI,XO,IERR)
WRITE (6,2000) IERR,XO
1000 FORMAT(' ',/,/,5X,'*** DIBKER ***',/,/,6X,'** INPUT **',&
/,/,8X,'N =',I3,5X,'XI =',F6.2 )
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF KERNX',/,/,10X,'XO = ',D18.10)
END
```

(d) 出力結果

```
*** DIBKER ***
** INPUT **
N = 3    XI = 1.00

** OUTPUT**
IERR = 0
VALUE OF KERNX
XO = 0.4887273882D+01
```

2.6.6 DIBKEI, RIBKEI

ケルビン関数 $\text{kei}_n(x)$

(1) 機能

ケルビン関数

$$\text{kei}_n(x) = \Im\{e^{-\sqrt{-1}n\pi/2} K_n(xe^{\sqrt{-1}\pi/4})\}$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL DIBKEI (N, XI, XO, IERR)

単精度サブルーチン:

CALL RIBKEI (N, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER(4)} \\ 64 \text{ ビット整数版では INTEGER(8)} \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入 力	次数 n
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 x
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	$\text{kei}_n(x)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $0.0 < XI \leq M$

ここで, $M = \{ \text{倍精度: } 1003.784, \text{ 単精度: } 125.473 \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
4000	$XI \leq 2.0/$ (最大値) または $ n (\log_e \frac{ n }{x} - M_1) > M_2$ であった. (注意事項 (a) 参照)	

(6) 注意事項

(a) このサブルーチンで IERR=4000 となるときの M_1, M_2 の値は以下の通りである.

$$M_1 = 0.3068, M_2 = \{ \text{倍精度: } 709.7827, \text{単精度: } 88.72284 \}$$

(b) $w = \text{ber}_\nu(x) + \sqrt{-1} \text{bei}_\nu(x), \text{ber}_{-\nu}(x) + \sqrt{-1} \text{bei}_{-\nu}(x), \text{ker}_\nu(x) + \sqrt{-1} \text{kei}_\nu(x), \text{ker}_{-\nu}(x) + \sqrt{-1} \text{kei}_{-\nu}(x)$ は微分方程式

$$x^2 \frac{d^2 w}{dx^2} + x \frac{dw}{dx} - (\sqrt{-1} x^2 + \nu^2) w = 0$$

の解である.

(7) 使用例

(a) 問題

$\text{kei}_n(x)$ の値を $n = 3, x = 1.0$ について求める.

(b) 入力データ

$N = 3, XI = 1.0$

(c) 主プログラム

```
PROGRAM BIBKEI
! *** EXAMPLE OF DIBKEI ***
IMPLICIT REAL(8) (A-H,O-Z)
READ (5,*) N
READ (5,*) XI
WRITE (6,1000) N,XI
CALL DIBKEI(N,XI,XO,IERR)
WRITE (6,2000) IERR,XO
1000 FORMAT(' ',/,/,5X,'*** DIBKEI ***',/,/,6X,'** INPUT **',&
/,/,8X,'N =',I3,5X,'XI =',F6.2 )
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF KEINX',/,/,10X,'XO = ',D18.10)
END
```

(d) 出力結果

```
*** DIBKEI ***
** INPUT **
N = 3    XI = 1.00

** OUTPUT**
IERR = 0
VALUE OF KEINX
XO = -0.6269710887D+01
```

2.6.7 WIBH0X, VIBH0X 0 次ストループ関数

(1) 機能

$x = X_i$ に対して 0 次のストループ関数

$$H_0(x) = \frac{2}{\pi} \int_0^{\frac{\pi}{2}} \sin(x \cos(t)) dt$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL WIBH0X (NV, XI, XO, IERR)

単精度サブルーチン:

CALL VIBH0X (NV, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力データ数
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	入 力	変数値 X_i
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	出 力	$H_0(X_i)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(b) $XI(i) \leq M$ ここで, $M = \{ \text{倍精度: } 2^{50}\pi, \text{単精度: } 2^{18}\pi \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
3000+i	XI(i) が制限条件 (b) を満足しなかった.	

(6) 注意事項

- (a)
- ν
- 次ストループ関数を
- $H_\nu(z)$
- とすると次の漸化式が成立する.

$$H_{\nu-1}(z) + H_{\nu+1}(z) = \frac{2\nu}{z} H_\nu(z) + \frac{\left(\frac{z}{2}\right)^\nu}{\sqrt{\pi}\Gamma\left(\nu + \frac{3}{2}\right)}$$

- (b) 微分方程式

$$z^2 \frac{d^2 w}{dz^2} + z \frac{dw}{dz} + (z^2 - \nu^2)w = \frac{4\left(\frac{z}{2}\right)^{\nu+1}}{\sqrt{\pi}\Gamma\left(\nu + \frac{1}{2}\right)}$$

の一般解は

$$w = aJ_\nu(z) + bY_\nu(z) + H_\nu(z)$$

である. ここで, $J_\nu(z)$, $Y_\nu(z)$ は第 1 種ならびに第 2 種ベッセル関数, a, b は定数である.

(7) 使用例

- (a) 問題

 $H_0(x)$ の値を $x = 0.0, 0.1, 0.2, \dots, 0.9$ について求める.

- (b) 主プログラム

```

PROGRAM EIBHOX
IMPLICIT REAL(8)(A-H,O-Z)
PARAMETER (NV=10)
REAL(8) XI(NV), XO(NV)
CHARACTER*6 CNAME, CFNC
PARAMETER( CNAME='WIBHOX', CFNC=' HO' )
!
DNV=NV
DO 1000 I=1,NV
  XI(I)=(I-1)/DNV
1000 CONTINUE
!
CALL WIBHOX( NV, XI, XO, IERR )
!
WRITE(6,6000) CNAME
WRITE(6,6100)
DO 2000 I=1,NV
  WRITE(6,6200) I,XI(I)
2000 CONTINUE
!
WRITE(6,6300)
WRITE(6,6400) IERR
DO 3000 I=1,NV
  WRITE(6,6500) CFNC,XI(I), XO(I)
3000 CONTINUE
STOP
!
6000 FORMAT(1X,'*** ',A6,' *')
6100 FORMAT(1X,'*** INPUT *')
6200 FORMAT(1X,'XI(',I2,')=',F10.6 )
6300 FORMAT(1X,'*** OUTPUT *')
6400 FORMAT(1X,'IERR=',I5 )
6500 FORMAT(1X,A6,'(',F10.6,')=',F10.6 )
END

```

- (c) 出力結果

```

*** WIBHOX *
*** INPUT *
XI( 1)= 0.000000
XI( 2)= 0.100000
XI( 3)= 0.200000
XI( 4)= 0.300000
XI( 5)= 0.400000
XI( 6)= 0.500000
XI( 7)= 0.600000
XI( 8)= 0.700000
XI( 9)= 0.800000
XI(10)= 0.900000
*** OUTPUT *
IERR= 0
HO( 0.000000)= 0.000000
HO( 0.100000)= 0.063591
HO( 0.200000)= 0.126759
HO( 0.300000)= 0.189083
HO( 0.400000)= 0.250150
HO( 0.500000)= 0.309556
HO( 0.600000)= 0.366911
HO( 0.700000)= 0.421842
HO( 0.800000)= 0.473994
HO( 0.900000)= 0.523035

```

2.6.8 WIBH1X, VIBH1X 1 次ストループ関数

(1) 機能

$x = X_i$ に対して 1 次ストループ関数

$$H_1(x) = \frac{2x}{\pi} \int_0^{\frac{\pi}{2}} \sin(x \cos(t)) \sin^2(t) dt$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL WIBH1X (NV, XI, XO, IERR)

単精度サブルーチン:

CALL VIBH1X (NV, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力データ数
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	入 力	変数値 X_i
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	出 力	$H_1(X_i)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(b) $XI(i) \leq M$

ここで, $M = \{ \text{倍精度: } 2^{50}\pi, \text{単精度: } 2^{18}\pi \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
3000+i	XI(i) が制限条件 (b) を満足しなかった.	

(6) 注意事項

- (a)
- ν
- 次ストループ関数を
- $H_\nu(z)$
- とすると次の漸化式が成立する.

$$H_{\nu-1}(z) + H_{\nu+1}(z) = \frac{2\nu}{z} H_\nu(z) + \frac{\left(\frac{z}{2}\right)^\nu}{\sqrt{\pi}\Gamma\left(\nu + \frac{3}{2}\right)}$$

- (b) 微分方程式

$$z^2 \frac{d^2 w}{dz^2} + z \frac{dw}{dz} + (z^2 - \nu^2)w = \frac{4\left(\frac{z}{2}\right)^{\nu+1}}{\sqrt{\pi}\Gamma\left(\nu + \frac{1}{2}\right)}$$

の一般解は

$$w = aJ_\nu(z) + bY_\nu(z) + H_\nu(z)$$

である. ここで, $J_\nu(z)$, $Y_\nu(z)$ は第 1 種ならびに第 2 種ベッセル関数, a, b は定数である.

(7) 使用例

- (a) 問題

 $H_1(x)$ の値を $x = 0.0, 0.1, 0.2, \dots, 0.9$ について求める.

- (b) 主プログラム

```

PROGRAM EIBH1X
IMPLICIT REAL(8)(A-H,O-Z)
PARAMETER (NV=10)
REAL(8) XI(NV), XO(NV)
CHARACTER*6 CNAME, CFNC
PARAMETER( CNAME='WIBH1X', CFNC=' H1' )
!
DNV=NV
DO 1000 I=1,NV
  XI(I)=(I-1)/DNV
1000 CONTINUE
!
CALL WIBH1X( NV, XI, XO, IERR )
!
WRITE(6,6000) CNAME
WRITE(6,6100)
DO 2000 I=1,NV
  WRITE(6,6200) I,XI(I)
2000 CONTINUE
!
WRITE(6,6300)
WRITE(6,6400) IERR
DO 3000 I=1,NV
  WRITE(6,6500) CFNC,XI(I), XO(I)
3000 CONTINUE
STOP
!
6000 FORMAT(1X,'*** ',A6,' *')
6100 FORMAT(1X,'*** INPUT *')
6200 FORMAT(1X,'XI(',I2,')=',F10.6 )
6300 FORMAT(1X,'*** OUTPUT *')
6400 FORMAT(1X,'IERR=',I5 )
6500 FORMAT(1X,A6,'(',F10.6,')=',F10.6 )
END

```

- (c) 出力結果

```

*** WIBH1X *
*** INPUT *
XI( 1)= 0.000000
XI( 2)= 0.100000
XI( 3)= 0.200000
XI( 4)= 0.300000
XI( 5)= 0.400000
XI( 6)= 0.500000
XI( 7)= 0.600000
XI( 8)= 0.700000
XI( 9)= 0.800000
XI(10)= 0.900000
*** OUTPUT *
IERR= 0
H1( 0.000000)= 0.000000
H1( 0.100000)= 0.002121
H1( 0.200000)= 0.008466
H1( 0.300000)= 0.018984
H1( 0.400000)= 0.033593
H1( 0.500000)= 0.052174
H1( 0.600000)= 0.074580
H1( 0.700000)= 0.100632
H1( 0.800000)= 0.130122
H1( 0.900000)= 0.162817

```


2.6.9 WIBHY0, VIBHY0 0 次ストループ関数とベッセル関数の差

(1) 機能

$x = X_i$ に対して 0 次ストループ関数と第 2 種 0 次ベッセル関数の差

$$H_0(x) - Y_0(x) = \frac{2}{\pi} \int_0^\infty e^{-xt} (1+t^2)^{-\frac{1}{2}} dt$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL WIBHY0 (NV, XI, XO, IERR)

単精度サブルーチン:

CALL VIBHY0 (NV, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\left\{ \begin{array}{l} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{array} \right\}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力データ数
2	XI	$\left\{ \begin{array}{l} D \\ R \end{array} \right\}$	NV	入 力	変数値 X_i
3	XO	$\left\{ \begin{array}{l} D \\ R \end{array} \right\}$	NV	出 力	$H_0(X_i) - Y_0(X_i)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(b) $XI(i) \geq 0.0$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
2000	$XI(i) = 0.0$ (overflow)	$XO(i) = (\text{最大値})$ を返す.
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
3000+i	$XI(i)$ が制限条件 (b) を満足しなかった.	

(6) 注意事項

(a) 以下の関係が成立する.

$$H_\nu(z) - Y_\nu(z) = \frac{2 \left(\frac{z}{2}\right)^\nu}{\sqrt{\pi}\Gamma(\nu + \frac{1}{2})} \int_0^\infty e^{-zt}(1+t^2)^{\nu-\frac{1}{2}} dt \quad (|\arg z| < \frac{\pi}{2})$$

(7) 使用例

(a) 問題

 $H_0(x) - Y_0(x)$ の値を $x = 0.1, 0.2, \dots, 1.0$ について求める.

(b) 主プログラム

```

PROGRAM EIBHYO
IMPLICIT REAL(8)(A-H,O-Z)
PARAMETER (NV=10)
REAL(8) XI(NV), XO(NV)
CHARACTER*6 CNAME, CFNC
PARAMETER( CNAME='WIBHYO', CFNC='HO-YO' )
!
DNV=NV
DO 1000 I=1,NV
  XI(I)=I/DNV
1000 CONTINUE
!
CALL WIBHYO( NV, XI, XO, IERR )
!
WRITE(6,6000) CNAME
WRITE(6,6100)
DO 2000 I=1,NV
  WRITE(6,6200) I,XI(I)
2000 CONTINUE
!
WRITE(6,6300)
WRITE(6,6400) IERR
DO 3000 I=1,NV
  WRITE(6,6500) CFNC,XI(I), XO(I)
3000 CONTINUE
STOP
!
6000 FORMAT(1X,'*** ',A6,' *')
6100 FORMAT(1X,'*** INPUT *')
6200 FORMAT(1X,'XI(',I2,')=',F10.6 )
6300 FORMAT(1X,'*** OUTPUT *')
6400 FORMAT(1X,'IERR=',I5 )
6500 FORMAT(1X,A6,'(',',F10.6,')=',F10.6 )
END

```

(c) 出力結果

```

*** WIBHYO *
*** INPUT *
XI( 1)= 0.100000
XI( 2)= 0.200000
XI( 3)= 0.300000
XI( 4)= 0.400000
XI( 5)= 0.500000
XI( 6)= 0.600000
XI( 7)= 0.700000
XI( 8)= 0.800000
XI( 9)= 0.900000
XI(10)= 1.000000
*** OUTPUT *
IERR= 0
HO-YO( 0.100000)= 1.597830
HO-YO( 0.200000)= 1.207864
HO-YO( 0.300000)= 0.996357
HO-YO( 0.400000)= 0.856174
HO-YO( 0.500000)= 0.754075
HO-YO( 0.600000)= 0.675421
HO-YO( 0.700000)= 0.612507
HO-YO( 0.800000)= 0.560797
HO-YO( 0.900000)= 0.517407
HO-YO( 1.000000)= 0.480400

```

2.6.10 WIBHY1, VIBHY1 1 次ストループ関数とベッセル関数の差

(1) 機能

$x = X_i$ に対して 1 次ストループ関数と第 2 種 1 次ベッセル関数の差

$$H_1(x) - Y_1(x) = \frac{2x}{\pi} \int_0^\infty e^{-xt}(1+t^2)^{\frac{1}{2}} dt$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL WIBHY1 (NV, XI, XO, IERR)

単精度サブルーチン:

CALL VIBHY1 (NV, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力データ数
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	入 力	変数値 X_i
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	出 力	$H_1(X_i) - Y_1(X_i)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(b) $XI(i) \geq 0.0$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
2000	$XI(i) \leq 1.0/(\text{最大値})$ (overflow)	$XO(i) = (\text{最大値})$ を返す.
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
3000+i	$XI(i)$ が制限条件 (b) を満足しなかった.	

(6) 注意事項

(a) 以下の関係が成立する.

$$H_\nu(z) - Y_\nu(z) = \frac{2 \left(\frac{z}{2}\right)^\nu}{\sqrt{\pi}\Gamma(\nu + \frac{1}{2})} \int_0^\infty e^{-zt}(1+t^2)^{\nu-\frac{1}{2}} dt \quad (|\arg z| < \frac{\pi}{2})$$

(7) 使用例

(a) 問題

 $H_1(x) - Y_1(x)$ の値を $x = 0.1, 0.2, \dots, 1.0$ について求める.

(b) 主プログラム

```

PROGRAM EIBHY1
IMPLICIT REAL(8)(A-H,O-Z)
PARAMETER (NV=10)
REAL(8) XI(NV), XO(NV)
CHARACTER*6 CNAME, CFNC
PARAMETER( CNAME='WIBHY1', CFNC=' H1-Y1' )
!
DNV=NV
DO 1000 I=1,NV
  XI(I)=I/DNV
1000 CONTINUE
!
CALL WIBHY1( NV, XI, XO, IERR )
!
WRITE(6,6000) CNAME
WRITE(6,6100)
DO 2000 I=1,NV
  WRITE(6,6200) I,XI(I)
2000 CONTINUE
!
WRITE(6,6300)
WRITE(6,6400) IERR
DO 3000 I=1,NV
  WRITE(6,6500) CFNC,XI(I), XO(I)
3000 CONTINUE
STOP
!
6000 FORMAT(1X,'*** ',A6,' *')
6100 FORMAT(1X,'*** INPUT *')
6200 FORMAT(1X,'XI(',I2,')=',F10.6 )
6300 FORMAT(1X,'*** OUTPUT *')
6400 FORMAT(1X,'IERR=',I5 )
6500 FORMAT(1X,A6,'(',F10.6,')=',F10.6 )
END

```

(c) 出力結果

```

*** WIBHY1 *
*** INPUT *
XI( 1)= 0.100000
XI( 2)= 0.200000
XI( 3)= 0.300000
XI( 4)= 0.400000
XI( 5)= 0.500000
XI( 6)= 0.600000
XI( 7)= 0.700000
XI( 8)= 0.800000
XI( 9)= 0.900000
XI(10)= 1.000000
*** OUTPUT *
IERR= 0
H1-Y1( 0.100000)= 6.461072
H1-Y1( 0.200000)= 3.332291
H1-Y1( 0.300000)= 2.312089
H1-Y1( 0.400000)= 1.814465
H1-Y1( 0.500000)= 1.523646
H1-Y1( 0.600000)= 1.334971
H1-Y1( 0.700000)= 1.203882
H1-Y1( 0.800000)= 1.108267
H1-Y1( 0.900000)= 1.035944
H1-Y1( 1.000000)= 0.979670

```

2.6.11 DIBAIX, RIBAIX エアリ関数 $Ai(x)$

(1) 機能

エアリ関数

$$Ai(x) = \begin{cases} \pi^{-1} \sqrt{\frac{x}{3}} K_{\frac{1}{3}}\left(\frac{2}{3}|x|^{\frac{3}{2}}\right) & (x \geq 0.0) \\ \frac{1}{3} \sqrt{|x|} [J_{\frac{1}{3}}\left(\frac{2}{3}|x|^{\frac{3}{2}}\right) + J_{-\frac{1}{3}}\left(\frac{2}{3}|x|^{\frac{3}{2}}\right)] & (x < 0.0) \end{cases}$$

の値を求める.

(2) 使用法

倍精度サブルーチン:

CALL DIBAIX (XI, XO, IERR)

単精度サブルーチン:

CALL RIBAIX (XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 x
2	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	$Ai(x)$ の値
3	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $XI \geq -M$

ここで, $M = \{ \text{倍精度: } (3 \times 2^{49} \times \pi)^{2/3}, \text{ 単精度: } (3 \times 2^{17} \times \pi)^{2/3} \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1000	$XI > M_1$ であった. (注意事項 (a) 参照)	XO = 0.0 を返す.
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.

(6) 注意事項

- (a) このサブルーチンで IERR=1000 となるときの
- M_1
- の値は以下の通りである。

$$M_1 = \{ \text{倍精度: 103.8, 単精度: 25.3} \}$$

- (b) 微分方程式

$$\frac{d^2 w}{dz^2} - zw = 0$$

の 1 次独立な解の組は $\{Ai(z), Bi(z)\}$, $\{Ai(z), Ai(ze^{\frac{2\sqrt{-1}\pi}{3}})\}$, $\{Ai(z), Ai(ze^{-\frac{2\sqrt{-1}\pi}{3}})\}$ である。

(7) 使用例

- (a) 問題

$x = -5.3$ におけるエアリ関数 $Ai(x)$ の値を求める。

- (b) 入力データ

$$XI = -5.3$$

- (c) 主プログラム

```

PROGRAM BIBAIX
! *** EXAMPLE OF DIBAIX ***
IMPLICIT REAL(8) (A-H,O-Z)
INTEGER IERR
!
READ(5,*) XI
WRITE(6,1000)
WRITE(6,2000) XI
CALL DIBAIX(XI,X0,IERR)
WRITE(6,3000)
WRITE(6,4000) IERR
WRITE(6,5000) X0
STOP
!
1000 FORMAT(' ',/,5X,'*** DIBAIX ***',/,&
6X,'** INPUT **')
2000 FORMAT(9X,'VALUE OF VARIABLE X = ',F4.1)
3000 FORMAT(' ',/,/,6X,'** OUTPUT **')
4000 FORMAT(9X,'IERR = ',I4)
5000 FORMAT(9X,'VALUE OF AIRY FUNCTION AI(X) = ',D17.10)
END

```

- (d) 出力結果

```

*** DIBAIX ***
** INPUT **
VALUE OF VARIABLE X = -5.3

** OUTPUT **
IERR = 0
VALUE OF AIRY FUNCTION AI(X) = 0.1825679311D+00

```

2.6.12 DIBBIX, RIBBIX エアリ関数 Bi(x)

- (1) 機能
エアリ関数

$$\text{Bi}(x) = \begin{cases} \sqrt{\frac{x}{3}} [I_{-\frac{1}{3}}(\frac{2}{3}|x|^{\frac{3}{2}}) + I_{\frac{1}{3}}(\frac{2}{3}|x|^{\frac{3}{2}})] & (x \geq 0.0) \\ \sqrt{\frac{|x|}{3}} [J_{-\frac{1}{3}}(\frac{2}{3}|x|^{\frac{3}{2}}) - J_{\frac{1}{3}}(\frac{2}{3}|x|^{\frac{3}{2}})] & (x < 0.0) \end{cases}$$

の値を求める。

- (2) 使用法
倍精度サブルーチン:
CALL DIBBIX (XI, XO, IERR)
単精度サブルーチン:
CALL RIBBIX (XI, XO, IERR)

- (3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入力	変数値 x
2	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出力	Bi(x) の値
3	IERR	I	1	出力	エラーインディケータ

- (4) 制限条件

(a) $XI \geq -M$

ここで, $M = \{ \text{倍精度: } (3 \times 2^{49} \times \pi)^{2/3}, \text{単精度: } (3 \times 2^{17} \times \pi)^{2/3} \}$

- (5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
2000	XI > M_1 であった. (注意事項 (a) 参照) (overflow)	XO =(最大値) を返す.
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.

(6) 注意事項

- (a) このサブルーチンで IERR=2000 となるときの
- M_1
- の値は以下の通りである。

$$M_1 = \{ \text{倍精度: } 104.266, \text{単精度: } 20.066 \}$$

- (b) 微分方程式

$$\frac{d^2 w}{dz^2} - zw = 0$$

の 1 次独立な解の組は $\{Ai(z), Bi(z)\}$, $\{Ai(z), Ai(ze^{\frac{2\sqrt{-1}\pi}{3}})\}$, $\{Ai(z), Ai(ze^{-\frac{2\sqrt{-1}\pi}{3}})\}$ である。

(7) 使用例

- (a) 問題

$x = -5.3$ におけるエアリ関数 $Bi(x)$ の値を求める。

- (b) 入力データ

$$XI = -5.3$$

- (c) 主プログラム

```

PROGRAM BIBBIX
! *** EXAMPLE OF DIBBIX ***
IMPLICIT REAL(8) (A-H,O-Z)
INTEGER IERR
!
READ(5,*) XI
WRITE(6,1000)
WRITE(6,2000) XI
CALL DIBBIX(XI,X0,IERR)
WRITE(6,3000)
WRITE(6,4000) IERR
WRITE(6,5000) X0
STOP
!
1000 FORMAT(' ',/,5X,'*** DIBBIX ***',/,&
6X,'** INPUT **')
2000 FORMAT(9X,'VALUE OF VARIABLE X = ',F4.1)
3000 FORMAT(' ',/,/,6X,'** OUTPUT **')
4000 FORMAT(9X,'IERR = ',I4)
5000 FORMAT(9X,'VALUE OF AIRY FUNCTION BI(X) = ',D17.10)
END

```

- (d) 出力結果

```

*** DIBBIX ***
** INPUT **
VALUE OF VARIABLE X = -5.3

** OUTPUT **
IERR = 0
VALUE OF AIRY FUNCTION BI(X) = -0.3237160767D+00

```


2.6.13 DIBAIID, RIBAIID エアリ関数の導関数 $Ai'(x)$

(1) 機能

エアリ関数の導関数

$$Ai'(x) = \begin{cases} -\frac{x}{\sqrt{3\pi}} K_{\frac{2}{3}}\left(\frac{2}{3}|x|^{\frac{3}{2}}\right) & (x \geq 0.0) \\ \frac{x}{3} [J_{-\frac{2}{3}}\left(\frac{2}{3}|x|^{\frac{3}{2}}\right) - J_{\frac{2}{3}}\left(\frac{2}{3}|x|^{\frac{3}{2}}\right)] & (x < 0.0) \end{cases}$$

の値を求める.

(2) 使用法

倍精度サブルーチン:

CALL DIBAIID (XI, XO, IERR)

単精度サブルーチン:

CALL RIBAIID (XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 x
2	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	$Ai'(x)$ の値
3	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $XI \geq -M$

ここで, $M = \{ \text{倍精度: } (3 \times 2^{49} \times \pi)^{2/3}, \text{ 単精度: } (3 \times 2^{17} \times \pi)^{2/3} \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1000	$XI > M_1$ であった. (注意事項 (a) 参照) (underflow)	$XO = 0.0$ を返す.
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.

(6) 注意事項

(a) このサブルーチンで IERR=1000 となるときの M_1 の値は以下の通りである.

$$M_1 = \{ \text{倍精度: } 104.1, \text{単精度: } 25.7 \}$$

(7) 使用例

(a) 問題

$x = -5.3$ におけるエアリ関数 $Ai(x)$ の導関数 $Ai'(x)$ の値を求める.

(b) 入力データ

$$XI = -5.3$$

(c) 主プログラム

```

PROGRAM BIBAIID
! *** EXAMPLE OF DIBAIID ***
IMPLICIT REAL(8) (A-H,O-Z)
INTEGER IERR
!
  READ(5,*) XI
  WRITE(6,1000)
  WRITE(6,2000) XI
  CALL DIBAIID(XI,XO,IERR)
  WRITE(6,3000)
  WRITE(6,4000) IERR
  WRITE(6,5000) XO
  STOP
!
1000 FORMAT(' ',/,5X,'*** DIBAIID ***',/,&
6X,'** INPUT **')
2000 FORMAT(9X,'VALUE OF VARIABLE X = ',F4.1)
3000 FORMAT(' ',/,/,6X,'** OUTPUT **')
4000 FORMAT(9X,'IERR = ',I4)
5000 FORMAT(9X,'VALUE OF DERIVED AIRY FUNCTION AI'(X) = ',D17.10)
END

```

(d) 出力結果

```

*** DIBAIID ***
** INPUT **
  VALUE OF VARIABLE X = -5.3

** OUTPUT **
  IERR =      0
  VALUE OF DERIVED AIRY FUNCTION AI'(X) =  0.7545754199D+00

```

2.6.14 DIBBID, RIBBID エアリ関数の導関数 $Bi'(x)$

(1) 機能

エアリ関数の導関数

$$Bi'(x) = \begin{cases} \frac{x}{\sqrt{3}} [I_{-\frac{2}{3}}(\frac{2}{3}|x|^{\frac{3}{2}}) + I_{\frac{2}{3}}(\frac{2}{3}|x|^{\frac{3}{2}})] & (x \geq 0.0) \\ -\frac{x}{\sqrt{3}} [J_{-\frac{2}{3}}(\frac{2}{3}|x|^{\frac{3}{2}}) + J_{\frac{2}{3}}(\frac{2}{3}|x|^{\frac{3}{2}})] & (x < 0.0) \end{cases}$$

の値を求める.

(2) 使用法

倍精度サブルーチン:

CALL DIBBID (XI, XO, IERR)

単精度サブルーチン:

CALL RIBBID (XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 x
2	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	$Bi'(x)$ の値
3	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $XI \geq -M$

ここで, $M = \{ \text{倍精度: } (3 \times 2^{49} \times \pi)^{2/3}, \text{単精度: } (3 \times 2^{17} \times \pi)^{2/3} \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
2000	$XI > M_1$ であった. (注意事項 (a) 参照) (overflow)	XO =(最大値) を返す.
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.

(6) 注意事項

(a) このサブルーチンで IERR=2000 となるときの M_1 の値は以下の通りである.

$$M_1 = \{ \text{倍精度: } 104.266, \text{単精度: } 20.066 \}$$

(7) 使用例

(a) 問題

$x = -5.3$ におけるエアリ関数 $Bi(x)$ の導関数 $Bi'(x)$ の値を求める.

(b) 入力データ

$$XI = -5.3$$

(c) 主プログラム

```

PROGRAM BIBBID
! *** EXAMPLE OF DIBBID ***
IMPLICIT REAL(8) (A-H,O-Z)
INTEGER IERR
!
  READ(5,*) XI
  WRITE(6,1000)
  WRITE(6,2000) XI
  CALL DIBBID(XI,XO,IERR)
  WRITE(6,3000)
  WRITE(6,4000) IERR
  WRITE(6,5000) XO
  STOP
!
1000 FORMAT(' ',/,5X,'*** DIBBID ***',/,&
6X,'** INPUT **')
2000 FORMAT(9X,'VALUE OF VARIABLE X = ',F4.1)
3000 FORMAT(' ',/,/,6X,'** OUTPUT **')
4000 FORMAT(9X,'IERR = ',I4)
5000 FORMAT(9X,'VALUE OF DERIVED AIRY FUNCTION BI'(X) = ',D17.10)
END

```

(d) 出力結果

```

*** DIBBID ***
** INPUT **
  VALUE OF VARIABLE X = -5.3

** OUTPUT **
  IERR =      0
  VALUE OF DERIVED AIRY FUNCTION BI'(X) =  0.4055569409D+00

```

2.7 ガンマ関数

2.7.1 WIGAMX, VIGAMX

実変数ガンマ関数

(1) 機能

$x = X_i$ に対して実変数ガンマ関数

$$\Gamma(x) = \int_0^{\infty} e^{-t} t^{x-1} dt$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL WIGAMX (NV, XI, XO, IERR)

単精度サブルーチン:

CALL VIGAMX (NV, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力データ数
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	入 力	変数値 X_i
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	出 力	$\Gamma(X_i)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(b) XI(i) は負の整数値または 0.0 と一致しないこと

(c) $XI(i) \geq -M$

ここで, $M = \{ \text{倍精度: } 170.4, \text{単精度: } 34.0 \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
2000	$ XI(i) \leq 1.0/(\text{最大値})$ または $XI(i) > M_1$ であった. (注意事項 (c) 参照) (overflow)	(最大値) を返す.
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
3000+i	$XI(i)$ が制限条件 (b) または (c) を満足しな かった.	

(6) 注意事項

(a) IERR = 3000 + i で制限条件 (b) を満足しないときは, $XI(i)$ が負の整数値に近い値でない限り, $\Gamma(X_i)$ の値は 0.0 に非常に近い値である.

(b) $x = -n$ ($n = 0, 1, 2, \dots$) はガンマ関数 $\Gamma(x)$ の, 有理型関数としての単純極で,

$$\lim_{x \rightarrow -n} \frac{1}{\Gamma(x)} = 0 \quad (n = 0, 1, 2, \dots)$$

であるので, ゼロや負の整数値に近い点でのガンマ関数の計算値について高い精度を期待することはできない.

(c) このサブルーチンで IERR=2000 となるときの M_1 の値は以下の通りである.

$$M_1 = \{ \text{倍精度: } 171.4, \text{単精度: } 35.0 \}$$

(d) ガンマ関数 $\Gamma(z)$ は第 2 種 Euler 積分とも呼ばれる. また, $\Gamma(z+1) = z\Gamma(z) = z!$ であるので階乗関数と呼ばれることもある.

(7) 使用例

(a) 問題

$\Gamma(x)$ の値を $x = 0.1, 0.2, \dots, 0.9, 1.0$ について求める.

(b) 主プログラム

```

PROGRAM EIGAMX
IMPLICIT REAL(8) (A-H,O-Z)
PARAMETER (NV=10)
REAL(8) XI(NV), XO(NV)
CHARACTER*6 CNAME, CFNC
PARAMETER( CNAME='WIGAMX', CFNC=' GAMMA' )
!
DNV=NV
DO 1000 I=1,NV
  XI(I)=I/DNV
1000 CONTINUE
!
CALL WIGAMX( NV, XI, XO, IERR )
!
WRITE(6,6000) CNAME
WRITE(6,6100)
DO 2000 I=1,NV
  WRITE(6,6200) I,XI(I)
2000 CONTINUE
!
WRITE(6,6300)
WRITE(6,6400) IERR
DO 3000 I=1,NV
  WRITE(6,6500) CFNC,XI(I), XO(I)
3000 CONTINUE
STOP
!
6000 FORMAT(1X,'*** ',A6,' *')
6100 FORMAT(1X,'*** INPUT *')
6200 FORMAT(1X,'XI(',I2,',')=',F10.6 )
6300 FORMAT(1X,'*** OUTPUT *')
6400 FORMAT(1X,'IERR=',I5 )
6500 FORMAT(1X,A6,'(',',F10.6,',')=',F10.6 )
END

```

(c) 出力結果

```
*** WIGAMX *
*** INPUT *
XI( 1)= 0.100000
XI( 2)= 0.200000
XI( 3)= 0.300000
XI( 4)= 0.400000
XI( 5)= 0.500000
XI( 6)= 0.600000
XI( 7)= 0.700000
XI( 8)= 0.800000
XI( 9)= 0.900000
XI(10)= 1.000000
*** OUTPUT *
IERR= 0
GAMMA( 0.100000)= 9.513508
GAMMA( 0.200000)= 4.590844
GAMMA( 0.300000)= 2.991569
GAMMA( 0.400000)= 2.218160
GAMMA( 0.500000)= 1.772454
GAMMA( 0.600000)= 1.489192
GAMMA( 0.700000)= 1.298055
GAMMA( 0.800000)= 1.164230
GAMMA( 0.900000)= 1.068629
GAMMA( 1.000000)= 1.000000
```

2.7.2 WIGLGX, VIGLGX 実変数対数ガンマ関数

(1) 機能

$x = X_i$ に対して実変数対数ガンマ関数 $\log_e(\Gamma(x))$ の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL WIGLGX (NV, XI, XO, IERR)

単精度サブルーチン:

CALL VIGLGX (NV, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER(4)} \\ 64 \text{ ビット整数版では INTEGER(8)} \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力データ数
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	入 力	変数値 X_i
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	出 力	$\log_e(\Gamma(X_i))$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(b) $XI(i)$ は負の整数値または 0.0 と一致しないこと

(c) $XI(i) > -M$

ここで, $M = \{ \text{倍精度: } 2^{50}, \text{単精度: } 2^{18} \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1000	$\Gamma(XI(i))$ が負であった.	$ \Gamma(XI(i)) $ に対して自然対数を取り値を返す.
2000	$XI(i) > M_1$ であった. (注意事項 (b) 参照) (overflow)	(最大値) を返す.
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
3000+i	$XI(i)$ が制限条件 (b) または (c) を満足しなかった.	

(6) 注意事項

(a) ガンマ関数値は一般に $\exp(XO(i))$ として得られるが, IERR = 1000 のときは, ある i に対して $-\exp(XO(i))$ として得られる.

- (b) このサブルーチンで IERR=2000 となるときの M_1 の値は以下の通りである。
 $M_1 = \{ \text{倍精度: } 2.545 \times 10^{305}, \text{単精度: } 4.08 \times 10^{36} \}$

(7) 使用例

(a) 問題

$\log_e(\Gamma(x))$ の値を $x = 0.1, 0.2, \dots, 0.9, 1.0$ について求める。

(b) 主プログラム

```

PROGRAM EIGLGX
IMPLICIT REAL(8)(A-H,O-Z)
PARAMETER (NV=10)
REAL(8) XI(NV), XO(NV)
CHARACTER*6 CNAME, CFNC
PARAMETER( CNAME='WIGLGX', CFNC='LGAMMA' )
!
DNV=NV
DO 1000 I=1,NV
  XI(I)=I/DNV
1000 CONTINUE
!
CALL WIGLGX( NV, XI, XO, IERR )
!
WRITE(6,6000) CNAME
WRITE(6,6100)
DO 2000 I=1,NV
  WRITE(6,6200) I,XI(I)
2000 CONTINUE
!
WRITE(6,6300)
WRITE(6,6400) IERR
DO 3000 I=1,NV
  WRITE(6,6500) CFNC,XI(I), XO(I)
3000 CONTINUE
STOP
!
6000 FORMAT(1X,'*** ',A6,' *')
6100 FORMAT(1X,'*** INPUT *')
6200 FORMAT(1X,'XI(',I2,')=',F10.6 )
6300 FORMAT(1X,'*** OUTPUT *')
6400 FORMAT(1X,'IERR=',I5 )
6500 FORMAT(1X,A6,'(',F10.6,')=',F10.6 )
END

```

(c) 出力結果

```

*** WIGLGX *
*** INPUT *
XI( 1)= 0.100000
XI( 2)= 0.200000
XI( 3)= 0.300000
XI( 4)= 0.400000
XI( 5)= 0.500000
XI( 6)= 0.600000
XI( 7)= 0.700000
XI( 8)= 0.800000
XI( 9)= 0.900000
XI(10)= 1.000000
*** OUTPUT *
IERR= 0
LGAMMA( 0.100000)= 2.252713
LGAMMA( 0.200000)= 1.524064
LGAMMA( 0.300000)= 1.095798
LGAMMA( 0.400000)= 0.796678
LGAMMA( 0.500000)= 0.572365
LGAMMA( 0.600000)= 0.398234
LGAMMA( 0.700000)= 0.260867
LGAMMA( 0.800000)= 0.152060
LGAMMA( 0.900000)= 0.066376
LGAMMA( 1.000000)= -0.000000

```

2.7.3 DIGIG1, RIGIG1

第1種不完全ガンマ関数

(1) 機能

第1種不完全ガンマ関数

$$\gamma(\nu, x) = \int_0^x e^{-t} t^{\nu-1} dt$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL DIGIG1 (V, XI, XO, IERR)

単精度サブルーチン:

CALL RIGIG1 (V, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: { 32ビット整数版では INTEGER(4) }
 R:単精度実数型 C:単精度複素数型 { 64ビット整数版では INTEGER(8) }

項番	引数名	型	大きさ	入出力	内 容
1	V	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 ν
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 x
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	$\gamma(\nu, x)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $XI \geq 0.0$ (b) $V \geq 0.0$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1000	$\nu \log_e(x) < -M_1$ であった. (注意事項 (b) 参照) (underflow)	XO = 0.0 を返す.
2000	$V \leq 1.0/(\text{最大値})$ (overflow)	XO =(最大値) を返す.
3000	制限条件 (a) または (b) を満足しなかった.	処理を打ち切る.
4000	XI > 1.0 で, 次のいずれかが成立した. <ul style="list-style-type: none"> • $\nu > (M_2 + x)/\log_e(x)$ • $x > x_m$ かつ $\nu > \nu_m$ (注意事項 (c) 参照)	処理を打ち切る. (注意事項 (a) 参照)

(6) 注意事項

- (a) IERR = 4000 のときは, $\gamma(\nu, x)$ の値はほぼ (最大値) である.
- (b) このサブルーチンで IERR=1000 となるときの M_1 の値は以下の通りである.
 $M_1 = \{ \text{倍精度: } 708.396, \text{単精度: } 87.336 \}$
- (c) このサブルーチンで IERR=4000 となるときの x_m, ν_m, M_2 の値は以下の通りである.
 $x_m = \{ \text{倍精度: } 171.0, \text{単精度: } 35.0 \},$
 $\nu_m = \{ \text{倍精度: } 171.4, \text{単精度: } 35.0 \},$
 $M_2 = \{ \text{倍精度: } 709.782, \text{単精度: } 88.722 \}$

(7) 使用例

- (a) 問題
 $\gamma(\nu, x)$ の値を $\nu = 4.0, x = 3.0$ について求める.
- (b) 入力データ
 $V = 4.0, XI = 3.0$
- (c) 主プログラム

```

PROGRAM BIGIG1
! *** EXAMPLE OF DIGIG1 ***
IMPLICIT REAL(8) (A-H,O-Z)
READ (5,*) V
READ (5,*) XI
WRITE (6,1000) V,XI
CALL DIGIG1(V,XI,XO,IERR)
WRITE (6,2000) IERR,XO
1000 FORMAT(' ',/,/,5X,'*** DIGIG1 ***',/,/,6X,'** INPUT **',&
/,/,8X,'V =',F6.2,5X,'XI =',F6.2)
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF INCOMPLETE GAMMA FUNCTION OF THE FIRST KIND',&
/,/,10X,'XO = ',D18.10)
END
    
```

(d) 出力結果

```

*** DIGIG1 ***
** INPUT **
V = 4.00    XI = 3.00

** OUTPUT**
IERR =      0
VALUE OF INCOMPLETE GAMMA FUNCTION OF THE FIRST KIND
XO = 0.2116608667D+01
    
```

2.7.4 DIGIG2, RIGIG2

第2種不完全ガンマ関数

(1) 機能

第2種不完全ガンマ関数

$$\Gamma(\nu, x) = \int_x^{\infty} e^{-t} t^{\nu-1} dt = e^{-x} \int_0^{\infty} e^{-t} (x+t)^{\nu-1} dt$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL DIGIG2 (V, XI, XO, IERR)

単精度サブルーチン:

CALL RIGIG2 (V, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: { 32ビット整数版では INTEGER(4) }
 R:単精度実数型 C:単精度複素数型 { 64ビット整数版では INTEGER(8) }

項番	引数名	型	大きさ	入出力	内 容
1	V	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 ν
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 x
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	$\Gamma(\nu, x)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $XI \geq 0.0$ (b) $0.0 \leq V \leq M$ ここで, $M = \{ \text{倍精度: } 171.4, \text{単精度: } 35.0 \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1000	$XI > V$ で $(\nu - 1) \log_e(x) - x < -M_1$ であった. (注意事項 (b) 参照) (underflow)	XO=0.0 を返す.
2000	$V = 0.0$ かつ $XI = 0.0$ (overflow)	XO =(最大値) を返す.
3000	制限条件 (a) または (b) を満足しなかった.	処理を打ち切る. (注意事項 (a) 参照)
4000	$XI \leq V - 0.5$ のとき $\nu \log_e(x) > (M_2 + x)$ または $x > x_m$ であった. (注意事項 (c) 参照)	処理を打ち切る.

(6) 注意事項

- (a) IERR = 3000 で ν が大きな値のときは, $\Gamma(\nu, x)$ の値はほぼ (最大値) である.
- (b) このサブルーチンで IERR=1000 となるときの M_1 の値は以下の通りである.
 $M_1 = \{ \text{倍精度: } 708.396, \text{単精度: } 87.336 \}$
- (c) このサブルーチンで IERR=4000 となるときの x_m, M_2 の値は以下の通りである.
 $x_m = \{ \text{倍精度: } 171.0, \text{単精度: } 35.0 \},$
 $M_2 = \{ \text{倍精度: } 709.7827, \text{単精度: } 88.72284 \}$

(7) 使用例

(a) 問題

$\Gamma(\nu, x)$ の値を $\nu = 4.0, x = 3.0$ について求める.

(b) 入力データ

$V = 4.0, XI = 3.0$

(c) 主プログラム

```

PROGRAM BIGIG2
! *** EXAMPLE OF DIGIG2 ***
IMPLICIT REAL(8) (A-H,O-Z)
READ (5,*) V
READ (5,*) XI
WRITE(6,1000) V,XI
CALL DIGIG2(V,XI,XO,IERR)
WRITE(6,2000) IERR,XO
1000 FORMAT(' ',/,/,5X,'*** DIGIG2 ***',/,/,6X,'** INPUT **',&
/,/,8X,'V =',F6.2,5X,'XI =',F6.2)
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF INCOMPLETE GAMMA FUNCTION OF THE SECOND KIND',&
/,/,10X,'XO = ',D18.10)
END

```

(d) 出力結果

```

*** DIGIG2 ***
** INPUT **
V = 4.00    XI = 3.00

** OUTPUT**
IERR =      0
VALUE OF INCOMPLETE GAMMA FUNCTION OF THE SECOND KIND
XO = 0.3883391333D+01

```

2.7.5 ZIGAMZ, CIGAMZ

複素変数ガンマ関数

(1) 機能

複素変数ガンマ関数

$$\Gamma(z) = \int_0^{\infty} e^{-t} t^{z-1} dt \quad (\Re\{z\} > 0)$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL ZIGAMZ (ZI, ZO, IERR)

単精度サブルーチン:

CALL CIGAMZ (ZI, ZO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	ZI	$\begin{Bmatrix} Z \\ C \end{Bmatrix}$	1	入 力	変数値 z
2	ZO	$\begin{Bmatrix} Z \\ C \end{Bmatrix}$	1	出 力	$\Gamma(z)$ の値
3	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) ZI は負の整数値または 0.0 と一致しないかつ $|ZI| > 1.0/(\text{最大値})$ であること。(b) $-M_1 \leq \Re(ZI) \leq M_2$ ここで, $M_1 = \{ \text{倍精度: } 170.4, \text{ 単精度: } 34.0 \}$, $M_2 = \{ \text{倍精度: } 171.4, \text{ 単精度: } 35.0 \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) または (b) を満足しなかった.	処理を打ち切る. (注意事項 (a) 参照)

(6) 注意事項

(a) $\Re(ZI) < -M_1$ のときは, IERR = 3000 が出力されるが, $\Gamma(z)$ の値は ZI が負の整数値に近い値でない限り 0.0 に非常に近い値である.

ここで, $M_1 = \{ \text{倍精度: 170.4, 単精度: 34.0} \}$

(b) $z = -n$ ($n = 0, 1, 2, \dots$) はガンマ関数 $\Gamma(z)$ の特異点すなわち

$$\lim_{z \rightarrow -n} \frac{1}{\Gamma(z)} = 0 \quad (n = 0, 1, 2, \dots)$$

であるので, ゼロや負の整数値に近い点でのガンマ関数の計算値について高い精度を期待することはできない.

(c) ガンマ関数 $\Gamma(z)$ は第 2 種 Euler 積分とも呼ばれる. また, $\Gamma(z+1) = z\Gamma(z) = z!$ であるので階乗関数と呼ばれることもある.

(7) 使用例

(a) 問題

$\Gamma(z)$ の値を $z = 1 + 2\sqrt{-1}$ について求める.

(b) 入力データ

ZI = (1.0, 2.0)

(c) 主プログラム

```

PROGRAM AIGAMZ
! *** EXAMPLE OF ZIGAMZ ***
IMPLICIT COMPLEX(8) (A-H,O-Z)
READ (5, '(D6.1,D6.1)') ZI
WRITE (6,1000) ZI
CALL ZIGAMZ(ZI,ZO,IERR)
WRITE (6,2000) IERR,ZO
1000 FORMAT(' ',/,/,5X,'*** ZIGAMZ ***',/,/,6X,'** INPUT **',&
/,/,8X,'ZI = (',F6.2,',',F6.2,',')')
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF GAMMA FUNCTION OF COMPLEX VARIABLE',&
/,/,10X,'ZO = (',D18.10,',',D18.10,',')')
END

```

(d) 出力結果

```

*** ZIGAMZ ***
** INPUT **
   ZI = ( 1.00 , 2.00 )

** OUTPUT**
IERR =      0
VALUE OF GAMMA FUNCTION OF COMPLEX VARIABLE
   ZO = ( 0.1519040027D+00 , 0.1980488016D-01 )

```

2.7.6 ZIGLGZ, CIGLGZ

複素変数対数ガンマ関数

(1) 機能

複素変数対数ガンマ関数 $\log_e(\Gamma(z))$ の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL ZIGLGZ (ZI, ZO, IERR)

単精度サブルーチン:

CALL CIGLGZ (ZI, ZO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	ZI	$\begin{Bmatrix} Z \\ C \end{Bmatrix}$	1	入力	変数値 z
2	ZO	$\begin{Bmatrix} Z \\ C \end{Bmatrix}$	1	出力	$\log_e(\Gamma(z))$ の値
3	IERR	I	1	出力	エラーインディケータ

(4) 制限条件

(a) ZI は負の整数値または 0.0 と一致しないこと。

(b) $\Re(ZI) > -M_1$

ここで, $M_1 = \{ \text{倍精度: } 2^{50}, \text{ 単精度: } 2^{18} \}$

(c) $|\Im(ZI)|, \Re(ZI) \leq M_2$

ここで, $M_2 = \{ \text{倍精度: } 2.545 \times 10^{305}, \text{ 単精度: } 4.08 \times 10^{36} \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1000	$\Gamma(z)$ が負の実数であった.	$\log_e(\Gamma(z)) + \pi\sqrt{-1}$ の値を返す.
3000	制限条件 (a), (b) または (c) を満足しなかつた.	処理を打ち切る.

(6) 注意事項

- (a) ガンマ関数値は, $\exp(ZO)$ として得られるが, $IERR = 1000$ のときは, $\exp(\Re(ZO))$ としてガンマ関数値を求めたほうがよい.
- (b) 複素変数対数ガンマ関数 $\log_e(\Gamma(z))$ は無限多価関数であり, それらは $2\pi\sqrt{-1}$ の整数倍の差を持つ. このサブルーチンでは $-\pi < \Im\{\log_e(\Gamma(z))\} \leq \pi$ となるように定義した主値を計算する.

(7) 使用例

(a) 問題

$\log_e(\Gamma(z))$ の値を $z = 1 + 2\sqrt{-1}$ について求める.

(b) 入力データ

ZI = (1.0, 2.0)

(c) 主プログラム

```

PROGRAM AIGLGZ
! *** EXAMPLE OF ZIGLGZ ***
IMPLICIT COMPLEX(8) (A-H,O-Z)
READ (5, '(D6.1,D6.1)') ZI
WRITE(6,1000) ZI
CALL ZIGLGZ(ZI,ZO,IERR)
WRITE(6,2000) IERR,ZO
1000 FORMAT(' ',/,/,5X,'*** ZIGLGZ ***',/,/,6X,'** INPUT **',&
/,/,8X,'ZI = (',F6.2,' ',F6.2,' )')
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF LOGARITHMIC GAMMA FUNCTION OF COMPLEX VARIABLE',&
/,/,10X,'ZO = (',D18.10,' ',D18.10,' )')
END
    
```

(d) 出力結果

```

*** ZIGLGZ ***
** INPUT **
   ZI = ( 1.00 , 2.00 )

** OUTPUT**
IERR =      0
VALUE OF LOGARITHMIC GAMMA FUNCTION OF COMPLEX VARIABLE
   ZO = ( -0.1876078786D+01 , 0.1296463163D+00 )
    
```

2.8 ガンマ関数に関連した関数

2.8.1 WIGDIG, VIGDIG

ディガンマ関数

(1) 機能

$x = X_i$ に対してディガンマ関数 (プシー関数)

$$\psi(x) = \frac{\Gamma'(x)}{\Gamma(x)} = \frac{d}{dx} \log_e(\Gamma(x))$$

の値を求める.

(2) 使用法

倍精度サブルーチン:

CALL WIGDIG (NV, XI, XO, IERR)

単精度サブルーチン:

CALL VIGDIG (NV, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER(4)} \\ 64 \text{ ビット整数版では INTEGER(8)} \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力データ数
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	入 力	変数値 X_i
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	出 力	$\psi(X_i)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(b) XI(i) は負の整数値または 0.0 と一致しないこと.

(c) $XI(i) > -M$

ここで, $M = \{ \text{倍精度: } 2^{50}, \text{単精度: } 2^{18} \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
3000+i	XI(i) が制限条件 (b) または (c) を満足しなかった.	

(6) 注意事項

(a) ディガンマ関数 $\psi(x)$ は

$$\psi(x+1) - \psi(x) = \frac{1}{x}, \quad \psi(1) = -\gamma, \quad \lim_{n \rightarrow \infty} \{\psi(x+n) - \psi(1+n)\} = 0$$

の解である。ここで、 γ は Euler の定数:

$$\gamma = \lim_{n \rightarrow \infty} \left(1 + \frac{1}{2} + \cdots + \frac{1}{n} - \log_e n \right)$$

(b) ディガンマ関数 $\psi(x)$ の導関数 $\psi'(x)$, $\psi''(x)$, $\psi'''(x)$ はそれぞれ、トリガンマ関数, テトラガンマ関数, ペンタガンマ関数と呼ばれる。また、これらは総称してポリガンマ関数と呼ばれる。

(7) 使用例

(a) 問題

 $\psi(x)$ の値を $x = 0.1, 0.2, \dots, 0.9, 1.0$ について求める。

(b) 主プログラム

```

PROGRAM EIGDIG
IMPLICIT REAL(8)(A-H,O-Z)
PARAMETER (NV=10)
REAL(8) XI(NV), XO(NV)
CHARACTER*6 CNAME, CFNC
PARAMETER( CNAME='WIGDIG', CFNC='DGAMMA' )
!
DNV=NV
DO 1000 I=1,NV
  XI(I)=I/DNV
1000 CONTINUE
!
CALL WIGDIG( NV, XI, XO, IERR )
!
WRITE(6,6000) CNAME
WRITE(6,6100)
DO 2000 I=1,NV
  WRITE(6,6200) I,XI(I)
2000 CONTINUE
!
WRITE(6,6300)
WRITE(6,6400) IERR
DO 3000 I=1,NV
  WRITE(6,6500) CFNC,XI(I), XO(I)
3000 CONTINUE
STOP
!
6000 FORMAT(1X,'*** ',A6,' *')
6100 FORMAT(1X,'*** INPUT *')
6200 FORMAT(1X,'XI(',I2,')=',F10.6 )
6300 FORMAT(1X,'*** OUTPUT *')
6400 FORMAT(1X,'IERR=',I5 )
6500 FORMAT(1X,A6,'(',F10.6,')=',F10.6 )
END

```

(c) 出力結果

```

*** WIGDIG *
*** INPUT *
XI( 1)= 0.100000
XI( 2)= 0.200000
XI( 3)= 0.300000
XI( 4)= 0.400000
XI( 5)= 0.500000
XI( 6)= 0.600000
XI( 7)= 0.700000
XI( 8)= 0.800000
XI( 9)= 0.900000
XI(10)= 1.000000
*** OUTPUT *
IERR= 0
DGAMMA( 0.100000)=-10.423755
DGAMMA( 0.200000)= -5.289040
DGAMMA( 0.300000)= -3.502524
DGAMMA( 0.400000)= -2.561385
DGAMMA( 0.500000)= -1.963510
DGAMMA( 0.600000)= -1.540619
DGAMMA( 0.700000)= -1.220024
DGAMMA( 0.800000)= -0.965009
DGAMMA( 0.900000)= -0.754927
DGAMMA( 1.000000)= -0.577216

```

2.8.2 WIGBET, VIGBET

ベータ関数

(1) 機能

$p = X_i, q = Y_i$ に対してベータ関数

$$B(p, q) = \int_0^1 t^{p-1} (1-t)^{q-1} dt$$

の値を求める.

(2) 使用法

倍精度サブルーチン:

CALL WIGBET (NV, P, Q, XO, IERR)

単精度サブルーチン:

CALL VIGBET (NV, P, Q, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力データ数
2	P	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	入 力	変数値 X_i
3	Q	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	入 力	変数値 Y_i
4	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	出 力	$B(X_i, Y_i)$ の値
5	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(b) $2.0/(\text{最大値}) < P(i) \leq M$

(c) $2.0/(\text{最大値}) < Q(i) \leq M$

ここで, $M = \{ \text{倍精度: } 1.2 \times 10^{305}, \text{ 単精度: } 2.0 \times 10^{36} \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
2000	$P(i) > 1000.0$, または $Q(i) > 1000.0$	処理を終了するが, 得られた値の精度が低い.
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
3000+i	$P(i)$ および $Q(i)$ が制限条件 (b) または (c) を満足しなかった.	

(6) 注意事項

- (a) ベータ関数 $B(p, q)$ は第 1 種 Euler 積分とも呼ばれる。
 (b) ベータ関数 $B(p, q)$ はガンマ関数 $\Gamma(z)$ を用いて次のように表すことができる。

$$B(p, q) = \frac{\Gamma(p)\Gamma(q)}{\Gamma(p+q)}$$

(7) 使用例

(a) 問題

$B(p, q)$ の値を $p = 0.1, 0.2, \dots, 0.9, 1.0, q = 0.5$ について求める。

(b) 主プログラム

```

PROGRAM EIGBET
IMPLICIT REAL(8)(A-H,O-Z)
PARAMETER (NV=10)
REAL(8) XI(NV), YI(NV), XO(NV)
CHARACTER*6 CNAME, CFNC
PARAMETER( CNAME='WIGBET', CFNC=' BETA' )
!
DNV=NV
DO 1000 I=1,NV
  XI(I)=I/DNV
  YI(I)=0.5DO
1000 CONTINUE
!
CALL WIGBET( NV, XI, YI, XO, IERR )
!
WRITE(6,6000) CNAME
WRITE(6,6100)
DO 2000 I=1,NV
  WRITE(6,6200) I,XI(I)
2000 CONTINUE
!
WRITE(6,6300)
WRITE(6,6400) IERR
DO 3000 I=1,NV
  WRITE(6,6500) CFNC,XI(I), XO(I)
3000 CONTINUE
STOP
!
6000 FORMAT(1X,'*** ',A6,' *')
6100 FORMAT(1X,'*** INPUT *')
6200 FORMAT(1X,'XI(',I2,')=',F10.6 )
6300 FORMAT(1X,'*** OUTPUT *')
6400 FORMAT(1X,'IERR=',I5 )
6500 FORMAT(1X,A6,'(',F10.6,')=',F10.6 )
END

```

(c) 出力結果

```

*** WIGBET *
*** INPUT *
XI( 1)= 0.100000
XI( 2)= 0.200000
XI( 3)= 0.300000
XI( 4)= 0.400000
XI( 5)= 0.500000
XI( 6)= 0.600000
XI( 7)= 0.700000
XI( 8)= 0.800000
XI( 9)= 0.900000
XI(10)= 1.000000
*** OUTPUT *
IERR= 0
BETA( 0.100000)= 11.323087
BETA( 0.200000)= 6.268653
BETA( 0.300000)= 4.554443
BETA( 0.400000)= 3.679094
BETA( 0.500000)= 3.141593
BETA( 0.600000)= 2.774502
BETA( 0.700000)= 2.505796
BETA( 0.800000)= 2.299288
BETA( 0.900000)= 2.134760
BETA( 1.000000)= 2.000000

```

2.9 楕円関数と楕円積分

2.9.1 WIECI1, VIECI1

第1種完全楕円積分

(1) 機能

$m = X_i$ に対して第1種完全楕円積分

$$K(m) = \int_0^1 \frac{dt}{\sqrt{(1-t^2)(1-mt^2)}} = \int_0^{\frac{\pi}{2}} \frac{1}{\sqrt{1-m\sin^2\theta}} d\theta$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL WIECI1 (NV, RM, XO, IERR)

単精度サブルーチン:

CALL VIECI1 (NV, RM, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力データ数
2	RM	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	入 力	母数 X_i
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	出 力	$K(X_i)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(b) $0.0 \leq RM(i) \leq 1.0$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
2000	$RM(i) = 1.0$ (overflow)	XO(i) =(最大値) を返す.
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
3000+i	RM(i) が制限条件 (b) を満足しなかった.	

(6) 注意事項

- (a) 第1種完全楕円積分が $K(k) = \int_0^{\frac{\pi}{2}} \frac{1}{\sqrt{1-k^2 \sin^2 \theta}} d\theta$ と表されているときは, RM(i) に k^2 の値を入力すること。
- (b) $K(m)$, $E(m)$ を共に求めたい場合は 2.9.8 $\left\{ \begin{array}{l} \text{WIENMQ} \\ \text{VIENMQ} \end{array} \right\}$ を利用した方が効率が良い。

(7) 使用例

(a) 問題

$K(m)$ の値を $m = 0.0, 0.1, 0.2, \dots, 0.9$ について求める。

(b) 主プログラム

```

PROGRAM EIECI1
IMPLICIT REAL(8)(A-H,O-Z)
PARAMETER (NV=10)
REAL(8) XI(NV), XO(NV)
CHARACTER*6 CNAME, CFNC
PARAMETER( CNAME='WIECI1', CFNC=' K' )
!
DNV=NV
DO 1000 I=1,NV
XI(I)=(I-1)/DNV
1000 CONTINUE
!
CALL WIECI1( NV, XI, XO, IERR )
!
WRITE(6,6000) CNAME
WRITE(6,6100)
DO 2000 I=1,NV
WRITE(6,6200) I,XI(I)
2000 CONTINUE
!
WRITE(6,6300)
WRITE(6,6400) IERR
DO 3000 I=1,NV
WRITE(6,6500) CFNC,XI(I), XO(I)
3000 CONTINUE
STOP
!
6000 FORMAT(1X,'*** ',A6,' *')
6100 FORMAT(1X,'*** INPUT *')
6200 FORMAT(1X,'XI(',I2,')=',F10.6 )
6300 FORMAT(1X,'*** OUTPUT *')
6400 FORMAT(1X,'IERR=',I5 )
6500 FORMAT(1X,A6,'(',F10.6,')=',F10.6 )
END

```

(c) 出力結果

```

*** WIECI1 *
*** INPUT *
XI( 1)= 0.000000
XI( 2)= 0.100000
XI( 3)= 0.200000
XI( 4)= 0.300000
XI( 5)= 0.400000
XI( 6)= 0.500000
XI( 7)= 0.600000
XI( 8)= 0.700000
XI( 9)= 0.800000
XI(10)= 0.900000
*** OUTPUT *
IERR= 0
K( 0.000000)= 1.570796
K( 0.100000)= 1.612441
K( 0.200000)= 1.659624
K( 0.300000)= 1.713889
K( 0.400000)= 1.777519
K( 0.500000)= 1.854075
K( 0.600000)= 1.949568
K( 0.700000)= 2.075363
K( 0.800000)= 2.257205
K( 0.900000)= 2.578092

```

2.9.2 WIECI2, VIECI2 第 2 種完全楕円積分

(1) 機能

$m = X_i$ に対して第 2 種完全楕円積分

$$E(m) = \int_0^1 \sqrt{(1-t^2)(1-mt^2)} dt = \int_0^{\frac{\pi}{2}} \sqrt{1-m \sin^2 \theta} d\theta$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL WIECI2 (NV, RM, XO, IERR)

単精度サブルーチン:

CALL VIECI2 (NV, RM, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: { 32 ビット整数版では INTEGER(4)
R:単精度実数型 C:単精度複素数型 { 64 ビット整数版では INTEGER(8) }

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力データ数
2	RM	{ D } { R }	NV	入 力	母数 X_i
3	XO	{ D } { R }	NV	出 力	$E(X_i)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(b) $0.0 \leq RM(i) \leq 1.0$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
3000+i	RM(i) が制限条件 (b) を満足しなかった.	

(6) 注意事項

- (a) 第2種完全楕円積分が $E(k) = \int_0^{\frac{\pi}{2}} \sqrt{1 - k^2 \sin^2 \theta} d\theta$ と表されているときは, RM(i) に k^2 の値を入力すること.
- (b) $E(m)$, $K(m)$ を共に求めたい場合は 2.9.8 $\left\{ \begin{array}{l} \text{WIENMQ} \\ \text{VIENMQ} \end{array} \right\}$ を利用した方が効率が良い.

(7) 使用例

(a) 問題

$E(m)$ の値を $m = 0.0, 0.1, 0.2, \dots, 0.9$ について求める.

(b) 主プログラム

```

PROGRAM EIECI2
IMPLICIT REAL(8)(A-H,O-Z)
PARAMETER (NV=10)
REAL(8) XI(NV), XO(NV)
CHARACTER*6 CNAME, CFNC
PARAMETER( CNAME='WIECI2', CFNC=' E' )
!
DNV=NV
DO 1000 I=1,NV
  XI(I)=(I-1)/DNV
1000 CONTINUE
!
CALL WIECI2( NV, XI, XO, IERR )
!
WRITE(6,6000) CNAME
WRITE(6,6100)
DO 2000 I=1,NV
  WRITE(6,6200) I,XI(I)
2000 CONTINUE
!
WRITE(6,6300)
WRITE(6,6400) IERR
DO 3000 I=1,NV
  WRITE(6,6500) CFNC,XI(I), XO(I)
3000 CONTINUE
STOP
!
6000 FORMAT(1X,'*** ',A6,' *')
6100 FORMAT(1X,'*** INPUT *')
6200 FORMAT(1X,'XI(',I2,')=',F10.6 )
6300 FORMAT(1X,'*** OUTPUT *')
6400 FORMAT(1X,'IERR=',I5 )
6500 FORMAT(1X,A6,'(',F10.6,')=',F10.6 )
END

```

(c) 出力結果

```

*** WIECI2 *
*** INPUT *
XI( 1)= 0.000000
XI( 2)= 0.100000
XI( 3)= 0.200000
XI( 4)= 0.300000
XI( 5)= 0.400000
XI( 6)= 0.500000
XI( 7)= 0.600000
XI( 8)= 0.700000
XI( 9)= 0.800000
XI(10)= 0.900000
*** OUTPUT *
IERR= 0
E( 0.000000)= 1.570796
E( 0.100000)= 1.530758
E( 0.200000)= 1.489035
E( 0.300000)= 1.445363
E( 0.400000)= 1.399392
E( 0.500000)= 1.350644
E( 0.600000)= 1.298428
E( 0.700000)= 1.241671
E( 0.800000)= 1.178490
E( 0.900000)= 1.104775

```

2.9.3 DIEI1, RIEI1

第 1 種不完全楕円積分

(1) 機能

第 1 種不完全楕円積分

$$F(x, m) = \int_0^x \frac{1}{\sqrt{(1-t^2)(1-mt^2)}} dt = \int_0^\psi \frac{1}{\sqrt{1-m\sin^2\theta}} d\theta \quad (\text{ただし, } x = \sin\psi)$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL DIEI1 (XI, RM, XO, IERR)

単精度サブルーチン:

CALL RIEI1 (XI, RM, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入力	変数値 x
2	RM	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入力	母数 m
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出力	$F(x, m)$ の値
4	IERR	I	1	出力	エラーインディケータ

(4) 制限条件

(a) $0.0 \leq XI \leq 1.0$ (b) $0.0 \leq RM \leq 1.0$ (注意事項 (c) 参照)

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
2000	XI = 1.0 かつ RM = 1.0 (overflow)	XO =(最大値) を返す.
3000	制限条件 (a) または (b) を満足しなかった.	処理を打ち切る.
4000	ガウスの算術幾何平均法またはニュートン法が収束しなかった.	

(6) 注意事項

(a) 第1種不完全楕円積分が $F(x, k) = \int_0^{\psi} \frac{1}{\sqrt{1 - k^2 \sin^2 \theta}} d\theta$ と表されているときは, RM に k^2 の値を入力すること.

(b) 第1種不完全楕円積分は

$$F(\varphi \setminus \alpha) = F(\varphi | m) = \int_0^{\varphi} \frac{1}{\sqrt{1 - \sin^2 \alpha \sin^2 \theta}} d\theta \quad (\text{ただし, } m = \sin^2 \alpha)$$

と表される場合がある.

(c) 母数 $m < 0.0$ のときは 2.9.5 $\left\{ \begin{array}{l} \text{DIEI3} \\ \text{RIEI3} \end{array} \right\}$ を用いる.

(7) 使用例

(a) 問題

$F(x, m)$ の値を $x = 0.3, m = 0.5$ について求める.

(b) 入力データ

XI = 0.3, RM = 0.5

(c) 主プログラム

```

PROGRAM BIEI1
! *** EXAMPLE OF DIEI1 ***
IMPLICIT REAL(8) (A-H,O-Z)
READ (5,*) XI
READ (5,*) RM
WRITE (6,1000) XI, RM
CALL DIEI1(XI, RM, XO, IERR)
WRITE (6,2000) IERR, XO
1000 FORMAT(' ', /, /, 5X, '*** DIEI1 ***', /, /, 6X, '** INPUT **', &
/, /, 8X, 'XI =', F6.2, 5X, 'RM =', F6.2 )
2000 FORMAT(' ', /, /, 6X, '** OUTPUT**', /, /, 8X, 'IERR = ', I5, &
/, /, 8X, 'VALUE OF F(X,M)', /, /, 10X, 'XO =', D18.10)
END
    
```

(d) 出力結果

```

*** DIEI1 ***
** INPUT **
   XI =  0.30    RM =  0.50

** OUTPUT**
IERR =      0
VALUE OF F(X,M)
   XO =  0.3070549305D+00
    
```

2.9.4 DIEII2, RIEII2

第 2 種不完全楕円積分

(1) 機能

第 2 種不完全楕円積分

$$E(x, m) = \int_0^x \sqrt{\frac{1 - mt^2}{1 - t^2}} dt = \int_0^\psi \sqrt{1 - m \sin^2 \theta} d\theta \quad (\text{ただし, } x = \sin \psi)$$

の値を求める.

(2) 使用法

倍精度サブルーチン:

CALL DIEII2 (XI, RM, XO, IERR)

単精度サブルーチン:

CALL RIEII2 (XI, RM, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入力	変数値 x
2	RM	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入力	母数 m
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出力	$E(x, m)$ の値
4	IERR	I	1	出力	エラーインディケータ

(4) 制限条件

(a) $0.0 \leq XI \leq 1.0$ (b) $0.0 \leq RM \leq 1.0$ (注意事項 (c) 参照)

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) または (b) を満足しなかった.	処理を打ち切る.
4000	ガウスの算術幾何平均法またはニュートン法が収束しなかった.	

(6) 注意事項

(a) 第2種不完全楕円積分が $E(x, k) = \int_0^{\psi} \sqrt{1 - k^2 \sin^2 \theta} d\theta$ と表されているときは, RM に k^2 の値を入力すること.

(b) 第2種不完全楕円積分は

$$E(\varphi|\alpha) = E(u|m) = \int_0^{\varphi} \sqrt{1 - \sin^2 \alpha \sin^2 \theta} d\theta \quad (\text{ただし, } m = \sin^2 \alpha, \quad \sin \varphi = \text{sn } u)$$

と表される場合がある. $E(u|m)$ の値を u をパラメータとして計算したい場合には 2.9.11 $\left\{ \begin{array}{l} \text{WIEJEP} \\ \text{VIEJEP} \end{array} \right\}$ を用いる.

(c) 母数 $m < 0.0$ のときは 2.9.5 $\left\{ \begin{array}{l} \text{DIEI3} \\ \text{RIEI3} \end{array} \right\}$ を用いる.

(7) 使用例

(a) 問題

$E(x, m)$ の値を $x = 0.3, m = 0.5$ について求める.

(b) 入力データ

XI = 0.3, RM = 0.5

(c) 主プログラム

```
PROGRAM BIEI2
! *** EXAMPLE OF DIEI2 ***
IMPLICIT REAL(8) (A-H,O-Z)
READ (5,*) XI
READ (5,*) RM
WRITE (6,1000) XI, RM
CALL DIEI2(XI, RM, XO, IERR)
WRITE (6,2000) IERR, XO
1000 FORMAT(' ', /, /, 5X, '*** DIEI2 ***', /, /, 6X, '** INPUT **', &
/, /, 8X, 'XI =', F6.2, 5X, 'RM =', F6.2)
2000 FORMAT(' ', /, /, 6X, '** OUTPUT**', /, /, 8X, 'IERR =', I5, &
/, /, 8X, 'VALUE OF E(X,M)', /, /, 10X, 'XO =', D18.10)
END
```

(d) 出力結果

```
*** DIEI2 ***
** INPUT **
   XI =  0.30    RM =  0.50

** OUTPUT**
IERR =      0
VALUE OF E(X,M)
   XO =  0.3023628305D+00
```

2.9.5 DIEI3, RIEI3 不完全変形楕円積分

(1) 機能

実数 $m \geq 0, a, b$ と $x \geq 0$ に対して, 不完全変形楕円積分

$$f(x, m, a, b) \equiv \int_0^x \frac{a + bt^2}{\sqrt{(1+t^2)^3 (1+mt^2)}} dt$$

の値を求める.

(2) 使用法

倍精度サブルーチン:

CALL DIEI3 (X, DM, A, B, Y, IERR)

単精度サブルーチン:

CALL RIEI3 (X, DM, A, B, Y, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER(4)} \\ 64 \text{ ビット整数版では INTEGER(8)} \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	X	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 x
2	DM	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	母数値 m
3	A	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	$a + bt^2$ の係数 a
4	B	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	$a + bt^2$ の係数 b
5	Y	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	不完全変形楕円積分 $f(x, m, a, b)$ の値
6	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $X \geq 0$

(b) $DM \geq 0$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) または (b) を満足しなかった.	処理を打ち切る.

(6) 注意事項

(a) 不完全楕円積分

$$F(r, m) = \int_0^r \frac{dt}{\sqrt{(1-t^2)(1-mt^2)}}, E(r, m) = \int_0^r \sqrt{\frac{1-mt^2}{1-t^2}} dt$$

と、因数 $1 + mt^2$ が違い、制限 $m < 1$ がない。

(b) 第1種不完全楕円積分 $\int_0^\psi \sqrt{(1-a\sin^2\theta)^{-1}} d\theta$ は $f(\tan\psi, 1-a, 1, 1)$ である。ここで $0 \leq a < 1$ であるが、第1種不完全楕円積分を $a < 1$ に拡張して定義できる。

(c) 第2種不完全楕円積分 $\int_0^\psi \sqrt{1-a\sin^2\theta} d\theta$ は $f(\tan\psi, 1-a, 1, 1-a)$ である。ここで $0 \leq a < 1$ であるが、第2種不完全楕円積分を $a < 1$ に拡張して定義できる。

(7) 使用例

(a) 問題

$x = 1.0, m = 3.0, a = 4.0, b = 2.0$ に対して不完全変形楕円積分の値を求める。

(b) 入力データ

X=1.0, DM=3.0, A=4.0, B=2.0

(c) 主プログラム

```

PROGRAM DIEI13
! *** EXAMPLE OF DIEI13 ***
IMPLICIT NONE
!
INTEGER IERR
REAL(8) X,DM,A,B,Y
DATA X/1.000000,DM/3.000000,A/4.000000,B/2.000000/
!
WRITE(6,6000) X, DM, A, B
!
CALL DIEI13(X, DM, A, B, Y, IERR)
!
WRITE(6,6010) IERR
WRITE(6,6020) Y
STOP
!
6000 FORMAT(/,&
1X,'*** DIEI13 ***',/,&
1X,' ** INPUT **',/,&
1X,' X= ',F10.7,' DM= ',F10.7,' A= ',F10.7,&
1X,' B= ',F10.7,/)
6010 FORMAT(/,&
1X,' ** OUTPUT **',/,&
1X,' IERR =',I5,/)
6020 FORMAT(1X,' Y= ',F10.7)
END

```

(d) 出力結果

```

*** DIEI13 ***
** INPUT **
X= 1.0000000 DM= 3.0000000 A= 4.0000000 B= 2.0000000

** OUTPUT **
IERR = 0
Y= 2.5140399

```

2.9.6 DIEI4, RIEI4

ワイエルシュトラス型の不完全楕円積分

(1) 機能

正数 x, y, z と実数 $p \geq 0$ に対して、ワイエルシュトラス型の不完全楕円積分

$$f(x, y, z, p) \equiv \frac{1}{2} \int_0^{1/p} \frac{dt}{\sqrt{(t+x)(t+y)(t+z)}}$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL DIEI4 (X, Y, Z, P, DI, IERR)

単精度サブルーチン:

CALL RIEI4 (X, Y, Z, P, DI, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	X	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 x
2	Y	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 y
3	Z	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 z
4	P	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	上端値の逆数 p (注意事項 (a) 参照)
5	DI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	不完全楕円積分 $f(x, y, z, p)$ の値 (注意事項 (b) 参照)
6	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $X, Y, Z > 0.0$

(b) $P \geq 0.0$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) または (b) を満足しなかった.	処理を打ち切る.

(6) 注意事項

- (a) 変数
- p
- を 0.0 にするとワイエルシュトラス型の完全楕円積分

$$\frac{1}{2} \int_0^{\infty} \frac{dt}{\sqrt{(t+x)(t+y)(t+z)}}$$

が DI に代入される.

- (b)
- $f(x, y, z, p)$
- を定義する定積分の上端が
- $1/p$
- であることに注意されたい (前項参照).

- (c)
- $f(x, y, z, p)$
- は,
- x, y, z
- について対称である.

- (d)
- $h \geq 0$
- のとき,

$$f(x+h, y+h, z+h, p) = \frac{1}{2} \int_h^{1/p+h} \frac{dt}{\sqrt{(t+x)(t+y)(t+z)}}$$

である.

(7) 使用例

- (a) 問題

$x = 1.0, y = 3.0, z = 4.0, p = 2.0$ に対してワイエルシュトラス型の不完全楕円積分の値を求める.

- (b) 入力データ

X=1.0, DM=3.0, A=4.0, B=2.0

- (c) 主プログラム

```

PROGRAM BIEII4
! *** EXAMPLE OF DIEII4 ***
IMPLICIT NONE
!
INTEGER IERR
REAL(8) X,Y,Z,P,DI
DATA X/1.DO/,Y/3.DO/,Z/4.DO/,P/2.DO/
!
WRITE(6,6000) X, Y, Z, P
!
CALL DIEII4(X, Y, Z, P, DI, IERR)
!
WRITE(6,6010) IERR
WRITE(6,6020) DI
!
STOP
6000 FORMAT(/,&
1X,'*** DIEII4 ***',/,/,&
1X,' ** INPUT **',/,/,&
1X,' X= ',F10.7,' Y= ',F10.7,' Z= ',F10.7,&
', P= ',F10.7,/)
6010 FORMAT(/,&
1X,' ** OUTPUT **',/,/,&
1X,' IERR =',I5,/)
6020 FORMAT(1X,' DI= ',F10.7)
END

```

- (d) 出力結果

```

*** DIEII4 ***
** INPUT **
X= 1.0000000 Y= 3.0000000 Z= 4.0000000 P= 2.0000000

** OUTPUT **
IERR = 0
DI= 0.0607049

```

2.9.7 WIEJAC, VIEJAC

ヤコビの楕円関数

(1) 機能

$u = X_i$ に対してヤコビの楕円関数 $\text{sn}(u, m)$, $\text{cn}(u, m)$, $\text{dn}(u, m)$ の値を求める.

ここで, u, m は $u = F(x, m)$: 第 1 種不完全楕円積分とした場合,

$\text{sn}(u, m) = \sin \psi = x$, $\text{cn}(u, m) = \cos \psi$, $\text{dn}(u, m) = \sqrt{1 - m \sin^2 \psi}$ となるように定める.

(2) 使用法

倍精度サブルーチン:

CALL WIEJAC (NV, UI, RM, SN, CN, DN, IERR)

単精度サブルーチン:

CALL VIEJAC (NV, UI, RM, SN, CN, DN, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\left\{ \begin{array}{l} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{array} \right\}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力データ数
2	UI	$\left\{ \begin{array}{l} D \\ R \end{array} \right\}$	NV	入 力	変数値 X_i
3	RM	$\left\{ \begin{array}{l} D \\ R \end{array} \right\}$	1	入 力	母数 m
4	SN	$\left\{ \begin{array}{l} D \\ R \end{array} \right\}$	NV	出 力	$\text{sn}(X_i, m)$ の関数値
5	CN	$\left\{ \begin{array}{l} D \\ R \end{array} \right\}$	NV	出 力	$\text{cn}(X_i, m)$ の関数値
6	DN	$\left\{ \begin{array}{l} D \\ R \end{array} \right\}$	NV	出 力	$\text{dn}(X_i, m)$ の関数値
7	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(b) $0.0 \leq RM \leq 1.0$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
3001	制限条件 (b) を満足しなかった.	

(6) 注意事項

- (a) このサブルーチンは第1種不完全楕円積分 u を $u = \int_0^\psi \frac{d\theta}{\sqrt{1-m\sin^2\theta}}$ と表したときに $\text{sn}(u, m) = \sin \psi$ となるように $\text{sn}(u, m)$, $\text{cn}(u, m)$, $\text{dn}(u, m)$ を求めるが、
 $u = \int_0^\psi \frac{d\theta}{\sqrt{1-k^2\sin^2\theta}}$ と表したときに $\text{sn}(u, k) = \sin \psi$ となるように $\text{sn}(u, k)$, $\text{cn}(u, k)$, $\text{dn}(u, k)$ を求めるときは、RM に k^2 の値を入力すること。一般には $\text{sn}(u, m)$, $\text{cn}(u, m)$, $\text{dn}(u, m)$ はそれぞれ $\text{sn}(u|m)$, $\text{cn}(u|m)$, $\text{dn}(u|m)$ と表される。

(7) 使用例

(a) 問題

$m=0.5$ として $\text{sn}(u, m)$, $\text{cn}(u, m)$, $\text{dn}(u, m)$ の値を $u = 0.0, 0.1, 0.2, \dots, 0.9$ について求める。

(b) 主プログラム

```

PROGRAM EIEJAC
IMPLICIT REAL(8) (A-H,O-Z)
PARAMETER (NV=10)
REAL(8) XI(NV), XO(NV,3)
CHARACTER*6 CNAME, CFNC
PARAMETER( CNAME='WIEJAC', CFNC='S C DN', RM=0.5D0)
!
DNV=NV
DO 1000 I=1,NV
  XI(I)=(I-1)/DNV
1000 CONTINUE
!
CALL WIEJAC( NV, XI, RM, XO(1,1),XO(1,2),XO(1,3), IERR )
!
WRITE(6,6000) CNAME
WRITE(6,6100)
DO 2000 I=1,NV
  WRITE(6,6200) I,XI(I)
2000 CONTINUE
!
WRITE(6,6300)
WRITE(6,6400) IERR
DO 3000 I=1,NV
  WRITE(6,6500) CFNC,XI(I), XO(I,1),XO(I,2),XO(I,3)
3000 CONTINUE
STOP
!
6000 FORMAT(1X,'*** ',A6,' *')
6100 FORMAT(1X,'*** INPUT *')
6200 FORMAT(1X,'XI(',I2,',)=',F10.6 )
6300 FORMAT(1X,'*** OUTPUT *')
6400 FORMAT(1X,'IERR=',I5 )
6500 FORMAT(1X,A6,'(',',F10.6,')=',',3F10.6 )
END

```

(c) 出力結果

```

*** WIEJAC *
*** INPUT *
XI( 1)= 0.000000
XI( 2)= 0.100000
XI( 3)= 0.200000
XI( 4)= 0.300000
XI( 5)= 0.400000
XI( 6)= 0.500000
XI( 7)= 0.600000
XI( 8)= 0.700000
XI( 9)= 0.800000
XI(10)= 0.900000
*** OUTPUT *
IERR= 0
S C DN( 0.000000)= -0.000000 1.000000 1.000000
S C DN( 0.100000)= 0.099751 0.995012 0.997509
S C DN( 0.200000)= 0.198022 0.980198 0.990148
S C DN( 0.300000)= 0.293413 0.955986 0.978241
S C DN( 0.400000)= 0.384672 0.923053 0.962296
S C DN( 0.500000)= 0.470750 0.882266 0.942972
S C DN( 0.600000)= 0.550831 0.834617 0.921028
S C DN( 0.700000)= 0.624340 0.781153 0.897273
S C DN( 0.800000)= 0.690935 0.722917 0.872528
S C DN( 0.900000)= 0.750478 0.660895 0.847580

```

2.9.8 WIENMQ, VIENMQ

ノーム q および完全楕円積分

(1) 機能

$m = X_i$ に対して,

$$\text{ノーム } q = e^{-\pi K(m')/K(m)},$$

$$\text{補ノーム } q' = e^{-\pi K(m)/K(m')} \text{ (ただし, } m' = 1 - m),$$

第1種完全楕円積分 $K(m)$, $K(m')$ および

第2種完全楕円積分 $E(m)$, $E(m')$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL WIENMQ (NV, RM, Q, QD, K, KD, E, ED, IERR)

単精度サブルーチン:

CALL VIENMQ (NV, RM, Q, QD, K, KD, E, ED, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\left\{ \begin{array}{l} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{array} \right\}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力データ数
2	RM	$\left\{ \begin{array}{l} D \\ R \end{array} \right\}$	NV	入 力	母数 $m = X_i$
3	Q	$\left\{ \begin{array}{l} D \\ R \end{array} \right\}$	NV	出 力	ノーム q の値
4	QD	$\left\{ \begin{array}{l} D \\ R \end{array} \right\}$	NV	出 力	補ノーム q' の値
5	K	$\left\{ \begin{array}{l} D \\ R \end{array} \right\}$	NV	出 力	第1種完全楕円積分 $K(m)$ の値
6	KD	$\left\{ \begin{array}{l} D \\ R \end{array} \right\}$	NV	出 力	第1種完全楕円積分 $K(m')$ の値
7	E	$\left\{ \begin{array}{l} D \\ R \end{array} \right\}$	NV	出 力	第2種完全楕円積分 $E(m)$ の値
8	ED	$\left\{ \begin{array}{l} D \\ R \end{array} \right\}$	NV	出 力	第2種完全楕円積分 $E(m')$ の値
9	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(b) $0.0 \leq RM(i) \leq 1.0$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
2000	RM(i) = 0.0 または 1.0 (overflow)	RM(i) = 0.0 のとき KD(i) =(最大値) を返す. RM(i) = 1.0 のとき K(i) =(最大値) を返す.
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
3000+i	RM(i) が制限条件 (b) を満足しなかった.	

(6) 注意事項

(a) 第 1 種完全楕円積分 $K(m)$ または第 2 種完全楕円積分 $E(m)$ の値のみを求めればよい場合は

2.9.1 $\left\{ \begin{matrix} \text{WIECI1} \\ \text{VIECI1} \end{matrix} \right\}$, 2.9.2 $\left\{ \begin{matrix} \text{WIECI2} \\ \text{VIECI2} \end{matrix} \right\}$ を利用した方が効率がよい.

(7) 使用例

(a) 問題

ノーム q , 補ノーム q' , 完全楕円積分 $K(m)$, $E(m)$,

$m' = 1 - m$ に対する完全楕円積分 $K(m')$, $E(m')$ を母数 $m = 0.1, 0.2, \dots, 0.9$ について求める.

(b) 主プログラム

```

PROGRAM EIENMQ
IMPLICIT REAL(8) (A-H,O-Z)
PARAMETER (NV=9)
REAL(8) XI(NV), XO(NV,6)
CHARACTER*6 CNAME, CFNC
PARAMETER( CNAME='WIENMQ', CFNC='PARAM' )
!
DNV=NV+1
DO 1000 I=1,NV
  XI(I)=I/DNV
1000 CONTINUE
!
CALL WIENMQ( NV, XI, XO(1,1),XO(1,2),XO(1,3),&
             XO(1,4),XO(1,5),XO(1,6), IERR)
!
WRITE(6,6000) CNAME
WRITE(6,6100)
DO 2000 I=1,NV
  WRITE(6,6200) I,XI(I)
2000 CONTINUE
!
WRITE(6,6300)
WRITE(6,6400) IERR
DO 3000 I=1,NV
  WRITE(6,6500) CFNC,XI(I), XO(I,1),XO(I,2),XO(I,3),&
                XO(I,4),XO(I,5),XO(I,6)
3000 CONTINUE
STOP
!
6000 FORMAT(1X,'*** ',A6,' *')
6100 FORMAT(1X,'*** INPUT *')
6200 FORMAT(1X,'XI(',I2,')=',F10.6 )
6300 FORMAT(1X,'*** OUTPUT *')
6400 FORMAT(1X,'IERR=',I5 )
6500 FORMAT(1X,A6,'(',F10.6,')=',F10.6 )
END

```

(c) 出力結果

```

*** WIENMQ *
*** INPUT *
XI( 1)= 0.100000
XI( 2)= 0.200000
XI( 3)= 0.300000
XI( 4)= 0.400000
XI( 5)= 0.500000
XI( 6)= 0.600000
XI( 7)= 0.700000
XI( 8)= 0.800000
XI( 9)= 0.900000
*** OUTPUT *
IERR= 0

```

PARAM(0.100000)=	0.006585	0.140173	1.612441	2.578092	1.530758	1.104775
PARAM(0.200000)=	0.013943	0.099274	1.659624	2.257205	1.489035	1.178490
PARAM(0.300000)=	0.022277	0.074690	1.713889	2.075363	1.445363	1.241671
PARAM(0.400000)=	0.031883	0.057020	1.777519	1.949568	1.399392	1.298428
PARAM(0.500000)=	0.043214	0.043214	1.854075	1.854075	1.350644	1.350644
PARAM(0.600000)=	0.057020	0.031883	1.949568	1.777519	1.298428	1.399392
PARAM(0.700000)=	0.074690	0.022277	2.075363	1.713889	1.241671	1.445363
PARAM(0.800000)=	0.099274	0.013943	2.257205	1.659624	1.178490	1.489035
PARAM(0.900000)=	0.140173	0.006585	2.578092	1.612441	1.104775	1.530758

2.9.9 WIETHE, VIETHE

楕円テータ関数

(1) 機能

$v = X_j$ に対してヤコビの楕円テータ関数 $\vartheta_i(v, q)$ の値を求める.

$$\vartheta_0(v, q) = \vartheta_4(v, q) = 1 + 2 \sum_{n=1}^{\infty} (-1)^n q^{n^2} \cos(2n\pi v)$$

$$\vartheta_1(v, q) = 2q^{1/4} \sum_{n=0}^{\infty} (-1)^n q^{n(n+1)} \sin((2n+1)\pi v)$$

$$\vartheta_2(v, q) = 2q^{1/4} \sum_{n=0}^{\infty} q^{n(n+1)} \cos((2n+1)\pi v)$$

$$\vartheta_3(v, q) = 1 + 2 \sum_{n=1}^{\infty} q^{n^2} \cos(2n\pi v)$$

(2) 使用法

倍精度サブルーチン:

CALL WIETHE (NV, I, V, Q, XO, IERR)

単精度サブルーチン:

CALL VIETHE (NV, I, V, Q, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力データ数
2	I	I	1	入 力	次数 i
3	V	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	入 力	変数値 X_j
4	Q	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	ノーム q
5	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	出 力	$\vartheta_i(X_j, q)$ の値
6	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(b) $0 \leq I \leq 4$

(c) $0 \leq Q < 1.0$

(d) $|V(j)| < M$

ここで, $M = \{ \text{倍精度: } 2^{31}, \text{単精度: } 2^{18} \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a),(b) または (c) を満足しなかつた.	処理を打ち切る.
4000+j	V(j) が制限条件 (d) を満足しなかった.	

(6) 注意事項

(a) $\vartheta_0(v, q) = \vartheta_4(v, q)$ である.

(b) 第 1 種不完全楕円積分 $F(x, m)$ の引数 (x, m) に対応する値を求める場合は, 2.9.3 $\left\{ \begin{matrix} \text{DIEI1} \\ \text{RIEI1} \end{matrix} \right\}$ から $u = F(x, m)$ を, 2.9.8 $\left\{ \begin{matrix} \text{WIENMQ} \\ \text{VIENMQ} \end{matrix} \right\}$ から ノーム q と第 1 種完全楕円積分 $K(m)$ とをそれぞれ求め, $v = \frac{u}{2K(m)}$ としてこのサブルーチンを適用する.

(c) $\vartheta'_4(v, q)$ は $\vartheta_4(v, q) = 2Z(u)K(m)\vartheta_4(v, q)$ により求めることができる.

(7) 使用例

(a) 問題

$\vartheta_i(v, q)$ の値を $i = 3, v = 0.0, 0.1, 0.2, \dots, 0.9, q = 0.5$ について求める.

(b) 主プログラム

```

PROGRAM EIETHE
IMPLICIT REAL(8)(A-H,O-Z)
PARAMETER (NV=10)
REAL(8) XI(NV), XO(NV)
CHARACTER*6 CNAME, CFNC
PARAMETER( CNAME='WIETHE', CFNC=' JTH3' )
!
DNV=NV
DO 1000 I=1,NV
  XI(I)=(I-1)/DNV
1000 CONTINUE
!
II=3
Q = 0.5D0
CALL WIETHE( NV, II, XI, Q, XO, IERR )
!
WRITE(6,6000) CNAME
WRITE(6,6100)
DO 2000 I=1,NV
  WRITE(6,6200) I,XI(I)
2000 CONTINUE
!
WRITE(6,6300)
WRITE(6,6400) IERR
DO 3000 I=1,NV
  WRITE(6,6500) CFNC,XI(I), XO(I)
3000 CONTINUE
STOP
!
6000 FORMAT(1X,'*** ',A6,' *')
6100 FORMAT(1X,'*** INPUT *')
6200 FORMAT(1X,'XI(',I2,')=',F10.6 )
6300 FORMAT(1X,'*** OUTPUT *')
6400 FORMAT(1X,'IERR=',I5 )
6500 FORMAT(1X,A6,'(',F10.6,')=',F10.6 )
END

```

(c) 出力結果

```

*** WIETHE *
*** INPUT *
XI( 1)= 0.000000
XI( 2)= 0.100000
XI( 3)= 0.200000
XI( 4)= 0.300000
XI( 5)= 0.400000
XI( 6)= 0.500000
XI( 7)= 0.600000
XI( 8)= 0.700000
XI( 9)= 0.800000
XI(10)= 0.900000

```


*** OUTPUT *

IERR= 0
JTH3(0.000000)= 2.128937
JTH3(0.100000)= 1.846412
JTH3(0.200000)= 1.204739
JTH3(0.300000)= 0.593025
JTH3(0.400000)= 0.230793
JTH3(0.500000)= 0.121124
JTH3(0.600000)= 0.230793
JTH3(0.700000)= 0.593025
JTH3(0.800000)= 1.204739
JTH3(0.900000)= 1.846412

2.9.10 WIEJZT, VIEJZT ヤコビのゼータ関数

(1) 機能

$u = X_i$ に対してヤコビのゼータ関数

$$Z(u) = \frac{\Theta'(u)}{\Theta(u)} \quad (\text{ただし, } \Theta(u) = \vartheta_4(v, q) = \vartheta_4(u/2K(m), q))$$

の値を求める.

(2) 使用法

倍精度サブルーチン:

CALL WIEJZT (NV, UI, RM, XO, IERR)

単精度サブルーチン:

CALL VIEJZT (NV, UI, RM, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER(4)} \\ 64 \text{ ビット整数版では INTEGER(8)} \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力データ数
2	UI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	入 力	変数値 X_i (第 1 種不完全楕円積分値 $F(x, m)$)
3	RM	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	母数 m
4	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	出 力	$Z(X_i)$ の値
5	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(b) $0.0 \leq RM \leq 1.0$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
3001	制限条件 (b) を満足しなかった.	

(6) 注意事項

(a) $u = 2K(m)v$ により u は v と $K(m)$ から求めることができる.

(7) 使用例

(a) 問題

$Z(u)$ の値を $u = 0.0, 0.1, 0.2, \dots, 0.9$, 母数 $m = 0.5$ について求める.

(b) 主プログラム

```

PROGRAM EIEJZT
IMPLICIT REAL(8) (A-H,O-Z)
PARAMETER (NV=10)
REAL(8) XI(NV), XO(NV)
CHARACTER*6 CNAME, CFNC
PARAMETER( CNAME='WIEJZT', CFNC=' Z' )
!
DNV=NV
DO 1000 I=1,NV
  XI(I)=(I-1)/DNV
1000 CONTINUE
!
RM = 0.5D0
CALL WIEJZT( NV, XI, RM, XO, IERR )
!
WRITE(6,6000) CNAME
WRITE(6,6100)
DO 2000 I=1,NV
  WRITE(6,6200) I,XI(I)
2000 CONTINUE
!
WRITE(6,6300)
WRITE(6,6400) IERR
DO 3000 I=1,NV
  WRITE(6,6500) CFNC,XI(I), XO(I)
3000 CONTINUE
STOP
!
6000 FORMAT(1X,'*** ',A6,' *')
6100 FORMAT(1X,'*** INPUT *')
6200 FORMAT(1X,'XI(',I2,')=',F10.6 )
6300 FORMAT(1X,'*** OUTPUT *')
6400 FORMAT(1X,'IERR=',I5 )
6500 FORMAT(1X,A6,'(',F10.6,')=',F10.6 )
END

```

(c) 出力結果

```

*** WIEJZT *
*** INPUT *
XI( 1)= 0.000000
XI( 2)= 0.100000
XI( 3)= 0.200000
XI( 4)= 0.300000
XI( 5)= 0.400000
XI( 6)= 0.500000
XI( 7)= 0.600000
XI( 8)= 0.700000
XI( 9)= 0.800000
XI(10)= 0.900000
*** OUTPUT *
IERR= 0
Z( 0.000000)= -0.000000
Z( 0.100000)= 0.026987
Z( 0.200000)= 0.052988
Z( 0.300000)= 0.077076
Z( 0.400000)= 0.098434
Z( 0.500000)= 0.116389
Z( 0.600000)= 0.130442
Z( 0.700000)= 0.140276
Z( 0.800000)= 0.145748
Z( 0.900000)= 0.146872

```

2.9.11 WIEJEP, VIEJEP ヤコビのエプシロン関数

(1) 機能

$u = X_i$ に対してヤコビのエプシロン関数

$$E(u|m) = \int_0^x \sqrt{\frac{1-mt^2}{1-t^2}} dt = \int_0^u \operatorname{dn}^2(t) dt \quad \left(\text{ただし, } u = \int_0^x \frac{dt}{\sqrt{(1-t^2)(1-mt^2)}} \right)$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL WIEJEP (NV, UI, RM, XO, IERR)

単精度サブルーチン:

CALL VIEJEP (NV, UI, RM, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力データ数
2	UI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	入 力	変数値 X_i (第 1 種不完全楕円積分値 $F(x, m)$)
3	RM	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	母数 m
4	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	出 力	$E(X_i m)$ の値
5	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(b) $0.0 \leq RM \leq 1.0$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
3001	制限条件 (b) を満足しなかった.	

(6) 注意事項

- (a) ヤコビのエプシロン関数 $E(u|m)$ は第2種不完全楕円積分 $E(\varphi|\alpha)$ と同じもので以下の関係がある。

$$E(\varphi|\alpha) = E(u|m) = \int_0^\varphi \sqrt{1 - \sin^2 \alpha \sin^2 \theta} d\theta \quad (\text{ただし, } m = \sin^2 \alpha, \quad \sin \varphi = \text{sn } u)$$

(2.9.2) $\left\{ \begin{array}{l} \text{WIECI2} \\ \text{VIECI2} \end{array} \right\}$ 参照

(7) 使用例

- (a) 問題

$E(u|m)$ の値を $u = 0.0, 0.1, 0.2, \dots, 0.9$, 母数 $m = 0.5$ について求める。

- (b) 主プログラム

```

PROGRAM EIEJEP
IMPLICIT REAL(8)(A-H,O-Z)
PARAMETER (NV=10)
REAL(8) XI(NV), XO(NV)
CHARACTER*6 CNAME, CFNC
PARAMETER( CNAME='WIEJEP', CFNC=' E' )
!
DNV=NV
DO 1000 I=1,NV
  XI(I)=(I-1)/DNV
1000 CONTINUE
!
RM = 0.5D0
CALL WIEJEP( NV, XI, RM, XO, IERR )
!
WRITE(6,6000) CNAME
WRITE(6,6100)
DO 2000 I=1,NV
  WRITE(6,6200) I,XI(I)
2000 CONTINUE
!
WRITE(6,6300)
WRITE(6,6400) IERR
DO 3000 I=1,NV
  WRITE(6,6500) CFNC,XI(I), XO(I)
3000 CONTINUE
STOP
!
6000 FORMAT(1X,'*** ',A6,' *')
6100 FORMAT(1X,'*** INPUT *')
6200 FORMAT(1X,'XI(',I2,')=',F10.6 )
6300 FORMAT(1X,'*** OUTPUT *')
6400 FORMAT(1X,'IERR=',I5 )
6500 FORMAT(1X,A6,'(',F10.6,')=',F10.6 )
END

```

- (c) 出力結果

```

*** WIEJEP *
*** INPUT *
XI( 1)= 0.000000
XI( 2)= 0.100000
XI( 3)= 0.200000
XI( 4)= 0.300000
XI( 5)= 0.400000
XI( 6)= 0.500000
XI( 7)= 0.600000
XI( 8)= 0.700000
XI( 9)= 0.800000
XI(10)= 0.900000
*** OUTPUT *
IERR=
E( 0.000000)= 0.000000
E( 0.100000)= 0.099834
E( 0.200000)= 0.198682
E( 0.300000)= 0.295618
E( 0.400000)= 0.389823
E( 0.500000)= 0.480625
E( 0.600000)= 0.567526
E( 0.700000)= 0.650208
E( 0.800000)= 0.728526
E( 0.900000)= 0.802498

```

2.9.12 WIEJTE, VIEJTE ヤコビのテータ関数

(1) 機能

$u = X_i$ に対してヤコビのテータ関数 $\Theta(u) = \vartheta_4(v, q) = \vartheta_4(u/2K(m), q)$ の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL WIEJTE (NV, UI, RM, XO, IERR)

単精度サブルーチン:

CALL VIEJTE (NV, UI, RM, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER(4)} \\ 64 \text{ ビット整数版では INTEGER(8)} \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力データ数
2	UI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	入 力	変数値 X_i (第 1 種不完全楕円積分値 $F(x, m)$)
3	RM	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	母数 m
4	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	出 力	$\Theta(X_i)$ の値
5	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(b) $0.0 \leq RM \leq 1.0$

(c) $\frac{|UI(i)|}{2K(m)} < M$

ここで, $M = \{ \text{倍精度: } 2^{31}, \text{単精度: } 2^{18} \}$, $K(m)$ は $m = RM$ での第 1 種完全楕円積分値。

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
3001	制限条件 (b) を満足しなかった.	
4000+i	UI(i) が制限条件 (c) を満足しなかった.	

(6) 注意事項

- (a) $\Theta'(u)$ の値は $Z(u)$ の値を求める 2.9.10 $\left\{ \begin{array}{l} \text{WIEJZT} \\ \text{VIEJZT} \end{array} \right\}$ とこのサブルーチンを用いて以下の式から求めることができる。

$$\Theta'(u) = Z(u)\Theta(u)$$

(7) 使用例

(a) 問題

$\Theta(u)$ の値を $u = 0.0, 0.1, 0.2, \dots, 0.9$, 母数 $m = 0.5$ について求める。

(b) 主プログラム

```

PROGRAM EIEJTE
IMPLICIT REAL(8) (A-H,O-Z)
PARAMETER (NV=10)
REAL(8) XI(NV), XO(NV)
CHARACTER*6 CNAME, CFNC
PARAMETER( CNAME='WIEJTE', CFNC='JTHETA' )
!
DNV=NV
DO 1000 I=1,NV
  XI(I)=(I-1)/DNV
1000 CONTINUE
!
RM = 0.5D0
CALL WIEJTE( NV, XI, RM, XO, IERR )
!
WRITE(6,6000) CNAME
WRITE(6,6100)
DO 2000 I=1,NV
  WRITE(6,6200) I,XI(I)
2000 CONTINUE
!
WRITE(6,6300)
WRITE(6,6400) IERR
DO 3000 I=1,NV
  WRITE(6,6500) CFNC,XI(I), XO(I)
3000 CONTINUE
STOP
!
6000 FORMAT(1X,'*** ',A6,' *')
6100 FORMAT(1X,'*** INPUT *')
6200 FORMAT(1X,'XI(',I2,')=',F10.6 )
6300 FORMAT(1X,'*** OUTPUT *')
6400 FORMAT(1X,'IERR=',I5 )
6500 FORMAT(1X,A6,'(',F10.6,')=',F10.6 )
END

```

(c) 出力結果

```

*** WIEJTE *
*** INPUT *
XI( 1)= 0.000000
XI( 2)= 0.100000
XI( 3)= 0.200000
XI( 4)= 0.300000
XI( 5)= 0.400000
XI( 6)= 0.500000
XI( 7)= 0.600000
XI( 8)= 0.700000
XI( 9)= 0.800000
XI(10)= 0.900000
*** OUTPUT *
IERR= 0
JTHETA( 0.000000)= 0.913579
JTHETA( 0.100000)= 0.914816
JTHETA( 0.200000)= 0.918493
JTHETA( 0.300000)= 0.924504
JTHETA( 0.400000)= 0.932677
JTHETA( 0.500000)= 0.942777
JTHETA( 0.600000)= 0.954517
JTHETA( 0.700000)= 0.967560
JTHETA( 0.800000)= 0.981533
JTHETA( 0.900000)= 0.996035

```

2.9.13 WIEPAI, VIEPAI パイ関数

(1) 機能

$u = X_i$ に対してパイ関数

$$\Pi(u, \alpha) = m \operatorname{sn} \alpha \operatorname{cn} \alpha \operatorname{dn} \alpha \int_0^u \frac{\operatorname{sn}^2 t dt}{1 - m \operatorname{sn}^2 \alpha \operatorname{sn}^2 t}$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL WIEPAI (NV, UI, ALF, RM, XO, IERR)

単精度サブルーチン:

CALL VIEPAI (NV, UI, ALF, RM, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力データ数
2	UI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	入 力	変数値 X_i (第 1 種不完全楕円積分値 $F(x, m)$)
3	ALF	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 α
4	RM	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	母数 m
5	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	出 力	$\Pi(X_i, \alpha)$ の値
6	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(b) $0.0 \leq RM < 1.0$

(c) $|\frac{|UI(i)| - ALF}{2K(m)}| < M$

ここで, $M = \{ \text{倍精度: } 2^{31}, \text{単精度: } 2^{18} \}$, $K(m)$ は $m = RM$ での第 1 種完全楕円積分値.

(d) $|\frac{ALF}{2K(m)}| < M$

ここで, M および $K(m)$ は制限条件 (c) に同じ.

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
3001	制限条件 (b) を満足しなかった.	
4000	制限条件 (d) を満足しなかった.	
4000+i	UI(i) が制限条件 (c) を満足しなかった.	

(6) 注意事項

(a) パイ関数 $\Pi(u, \alpha)$ はヤコビのテータ関数 $\Theta(u)$ を用いて次のように表せる.

$$\Pi(u, \alpha) = u \frac{\Theta'(\alpha)}{\Theta(\alpha)} + \frac{1}{2} \log_e \frac{\Theta(u - \alpha)}{\Theta(u + \alpha)}$$

(b) 第 3 種不完全楕円積分 $F(z)$ は

$$F(z) = \int_0^z \frac{dz}{(1 - a^2 z^2) \sqrt{(1 - z^2)(1 - k^2 z^2)}} = \frac{\operatorname{sn} \alpha}{\operatorname{cn} \alpha \operatorname{dn} \alpha} \Pi(u, \alpha) + u$$

と表せる. ただし, $z = \operatorname{sn} u$, $a^2 = k^2 \operatorname{sn}^2 \alpha$ である.

(7) 使用例

(a) 問題

$\Pi(u, \alpha)$ の値を $u = 0.0, 0.1, 0.2, \dots, 0.9$, $\alpha = 1.7$, 母数 $m = 0.5$ について求める.

(b) 主プログラム

```

PROGRAM EIEPAI
IMPLICIT REAL(8) (A-H,O-Z)
PARAMETER (NV=10)
REAL(8) XI(NV), XO(NV)
CHARACTER*6 CNAME, CFNC
PARAMETER( CNAME='WIEPAI', CFNC='JACPAI' )
!
DNV=NV
DO 1000 I=1,NV
  XI(I)=(I-1)/DNV
1000 CONTINUE
!
RM = 0.5D0
ALF= 1.7D0
CALL WIEPAI( NV, XI, ALF, RM, XO, IERR )
!
WRITE(6,6000) CNAME
WRITE(6,6100)
DO 2000 I=1,NV
  WRITE(6,6200) I,XI(I)
2000 CONTINUE
!
WRITE(6,6300)
WRITE(6,6400) IERR
DO 3000 I=1,NV
  WRITE(6,6500) CFNC,XI(I), XO(I)
3000 CONTINUE
STOP
!
6000 FORMAT(1X,'*** ',A6,' *')
6100 FORMAT(1X,'*** INPUT *')
6200 FORMAT(1X,'XI(',I2,',')=',F10.6 )
6300 FORMAT(1X,'*** OUTPUT *')
6400 FORMAT(1X,'IERR=',I5 )
6500 FORMAT(1X,A6,'(',',F10.6,')= ',F10.6 )
END

```

(c) 出力結果

```

*** WIEPAI *
*** INPUT *
XI( 1)= 0.000000
XI( 2)= 0.100000
XI( 3)= 0.200000
XI( 4)= 0.300000
XI( 5)= 0.400000
XI( 6)= 0.500000
XI( 7)= 0.600000
XI( 8)= 0.700000

```

```
XI( 9)= 0.800000
XI(10)= 0.900000
*** OUTPUT *
IERR= 0
JACPAI( 0.000000)= 0.000000
JACPAI( 0.100000)= 0.000013
JACPAI( 0.200000)= 0.000103
JACPAI( 0.300000)= 0.000346
JACPAI( 0.400000)= 0.000821
JACPAI( 0.500000)= 0.001601
JACPAI( 0.600000)= 0.002762
JACPAI( 0.700000)= 0.004373
JACPAI( 0.800000)= 0.006501
JACPAI( 0.900000)= 0.009204
```

2.10 初等関数の不定積分

2.10.1 WIIEXP, VIIEXP

指数積分

(1) 機能

$x = X_i$ に対して指数積分

$$\overline{\text{Ei}}(x) = P \int_{-\infty}^x \frac{e^t}{t} dt \quad (x > 0.0)$$

$$\text{Ei}(x) = - \int_{-x}^{\infty} \frac{e^{-t}}{t} dt = \int_{-\infty}^x \frac{e^t}{t} dt \quad (x < 0.0)$$

の値を求める。ただし、 P は Cauchy の主値を表す。

(2) 使用法

倍精度サブルーチン:

CALL WIIEXP (NV, XI, XO, IERR)

単精度サブルーチン:

CALL VIIEXP (NV, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\left\{ \begin{array}{l} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{array} \right\}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力データ数
2	XI	$\left\{ \begin{array}{l} D \\ R \end{array} \right\}$	NV	入 力	変数値 X_i
3	XO	$\left\{ \begin{array}{l} D \\ R \end{array} \right\}$	NV	出 力	$X_i > 0.0$: $\overline{\text{Ei}}(X_i)$ の値 $X_i < 0.0$: $\text{Ei}(X_i)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1000	XI(i) < -M ₁ であった. (注意事項 (b) 参照) (underflow)	XO(i) = 0.0 を返す.
2000	XI(i) = 0.0 (overflow)	XO(i) =(最小値) を返す.
2100	XI(i) > M ₂ であった. (注意事項 (c) 参照) (overflow)	XO(i) =(最大値) を返す.
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.

(6) 注意事項

(a) $x = XI(i) > 0.0$ のときは $\overline{Ei}(x) = P \int_{-\infty}^x \frac{e^t}{t} dt$, $x = XI(i) < 0.0$ のときは $Ei(x) = - \int_{-x}^{\infty} \frac{e^{-t}}{t} dt = \int_{-\infty}^x \frac{e^t}{t} dt$ を計算する.

(b) このサブルーチンで IERR = 1000 となるときの M₁ の値は以下の通りである.

$$M_1 = \{ \text{倍精度: } 702.0, \text{ 単精度: } 83.0 \}$$

(c) このサブルーチンで IERR = 2100 となるときの M₂ の値は以下の通りである.

$$M_2 = \{ \text{倍精度: } 709.782, \text{ 単精度: } 88.722 \}$$

(d) 指数積分を

$$E_1(z) = \int_z^{\infty} \frac{e^t}{t} dt \quad (|\arg z| < \pi)$$

と定義する場合もある. この場合 $E_1(x) = -Ei(-x)$ ($x > 0$) となる.

(7) 使用例

(a) 問題

$\overline{Ei}(x)$ の値を $x = 0.1, 0.2, \dots, 1.0$ について求める.

(b) 主プログラム

```

PROGRAM EIIEXP
IMPLICIT REAL(8) (A-H,O-Z)
PARAMETER (NV=10)
REAL(8) XI(NV), XO(NV)
CHARACTER*6 CNAME, CFNC
PARAMETER( CNAME='WIIEXP', CFNC=' Ei' )
!
DNV=NV
DO 1000 I=1,NV
  XI(I)=I/DNV
1000 CONTINUE
!
CALL WIIEXP( NV, XI, XO, IERR )
!
WRITE(6,6000) CNAME
WRITE(6,6100)
DO 2000 I=1,NV
  WRITE(6,6200) I,XI(I)
2000 CONTINUE
!
WRITE(6,6300)
WRITE(6,6400) IERR
DO 3000 I=1,NV
  WRITE(6,6500) CFNC,XI(I), XO(I)
3000 CONTINUE
STOP
!
6000 FORMAT(1X,'*** ',A6,' *')
6100 FORMAT(1X,'*** INPUT *')
6200 FORMAT(1X,'XI(',I2,')=',F10.6 )
6300 FORMAT(1X,'*** OUTPUT *')
6400 FORMAT(1X,'IERR=',I5 )
6500 FORMAT(1X,A6,'(',F10.6,')=',F10.6 )
END

```

(c) 出力結果

```
*** WIIEXP *
*** INPUT *
XI( 1)= 0.100000
XI( 2)= 0.200000
XI( 3)= 0.300000
XI( 4)= 0.400000
XI( 5)= 0.500000
XI( 6)= 0.600000
XI( 7)= 0.700000
XI( 8)= 0.800000
XI( 9)= 0.900000
XI(10)= 1.000000
*** OUTPUT *
IERR= 0
  Ei( 0.100000)= -1.622813
  Ei( 0.200000)= -0.821761
  Ei( 0.300000)= -0.302669
  Ei( 0.400000)=  0.104765
  Ei( 0.500000)=  0.454220
  Ei( 0.600000)=  0.769881
  Ei( 0.700000)=  1.064907
  Ei( 0.800000)=  1.347397
  Ei( 0.900000)=  1.622812
  Ei( 1.000000)=  1.895118
```

2.10.2 WILOG, VILOG

対数積分

(1) 機能

$x = X_i$ に対して対数積分

$$\text{Li}(x) = \int_0^x \frac{1}{\log_e(t)} dt$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL WILOG (NV, XI, XO, IERR)

単精度サブルーチン:

CALL VILOG (NV, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力データ数
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	入 力	変数値 X_i
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	出 力	$\text{Li}(X_i)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(b) $XI(i) \geq 1.0$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
2000	$XI(i) = 1.0$ (overflow)	$XO(i) = (\text{最小値})$ を返す.
2100	$\log(XI(i)) > M_2$ であった. (注意事項 (a) 参照)	$XO(i) = (\text{最大値})$ を返す.
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
3000+i	$XI(i)$ が制限条件 (b) を満足しなかった.	

(6) 注意事項

(a) このサブルーチンで $IERR = 2100$ となるときの M_2 の値は以下の通りである.

$M_2 = \{ \text{倍精度: } 709.782, \text{ 単精度: } 88.722 \}$

(7) 使用例

(a) 問題

$\text{Li}(x)$ の値を $x = 1.1, 1.2, \dots, 2.0$ について求める.

(b) 主プログラム

```

PROGRAM EILOG
IMPLICIT REAL(8) (A-H,O-Z)
PARAMETER (NV=10)
REAL(8) XI(NV), XO(NV)
CHARACTER*6 CNAME, CFNC
PARAMETER( CNAME='WILOG', CFNC=' Li' )
!
DNV=NV
DO 1000 I=1,NV
  XI(I)=1.DO+I/DNV
1000 CONTINUE
!
CALL WILOG( NV, XI, XO, IERR )
!
WRITE(6,6000) CNAME
WRITE(6,6100)
DO 2000 I=1,NV
  WRITE(6,6200) I,XI(I)
2000 CONTINUE
!
WRITE(6,6300)
WRITE(6,6400) IERR
DO 3000 I=1,NV
  WRITE(6,6500) CFNC,XI(I), XO(I)
3000 CONTINUE
STOP
!
6000 FORMAT(1X,'*** ',A6,' *')
6100 FORMAT(1X,'*** INPUT *')
6200 FORMAT(1X,'XI(',I2,',')=',F10.6 )
6300 FORMAT(1X,'*** OUTPUT *')
6400 FORMAT(1X,'IERR=',I5 )
6500 FORMAT(1X,A6,'(',',F10.6,')=',F10.6 )
END

```

(c) 出力結果

```

*** WILOG *
*** INPUT *
XI( 1)= 1.100000
XI( 2)= 1.200000
XI( 3)= 1.300000
XI( 4)= 1.400000
XI( 5)= 1.500000
XI( 6)= 1.600000
XI( 7)= 1.700000
XI( 8)= 1.800000
XI( 9)= 1.900000
XI(10)= 2.000000
*** OUTPUT *
IERR= 0
Li( 1.100000)= -1.675773
Li( 1.200000)= -0.933787
Li( 1.300000)= -0.480178
Li( 1.400000)= -0.144991
Li( 1.500000)= 0.125065
Li( 1.600000)= 0.353748
Li( 1.700000)= 0.553744
Li( 1.800000)= 0.732637
Li( 1.900000)= 0.895327
Li( 2.000000)= 1.045164

```

2.10.3 DIISIN, RIISIN 正弦積分

- (1) 機能
正弦積分

$$\text{Si}(x) = \int_0^x \frac{\sin t}{t} dt$$

の値を求める。

- (2) 使用法
倍精度サブルーチン:
CALL DIISIN (XI, XO, IERR)
単精度サブルーチン:
CALL RIISIN (XI, XO, IERR)

- (3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: { 32ビット整数版では INTEGER(4) }
R:単精度実数型 C:単精度複素数型 { 64ビット整数版では INTEGER(8) }

項番	引数名	型	大きさ	入出力	内 容
1	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 x
2	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	$\text{Si}(x)$ の値
3	IERR	I	1	出 力	エラーインディケータ

- (4) 制限条件
なし
- (5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	

- (6) 注意事項
なし

(7) 使用例

(a) 問題

$Si(x)$ の値を $x = 1.0$ について求める.

(b) 入力データ

XI = 1.0

(c) 主プログラム

```

PROGRAM BIISIN
! *** EXAMPLE OF DIISIN ***
IMPLICIT REAL(8) (A-H,O-Z)
READ (5,*) XI
WRITE (6,1000) XI
CALL DIISIN(XI,XO,IERR)
WRITE (6,2000) IERR,XO
1000 FORMAT(' ',/,/,5X,'*** DIISIN ***',/,/,6X,'** INPUT **',&
/,/,8X,'XI = ',F6.2 )
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF SI(X)',/,/,10X,'XO = ',D18.10)
END

```

(d) 出力結果

```

*** DIISIN ***
** INPUT **
XI = 1.00

** OUTPUT**
IERR = 0
VALUE OF SI(X)
XO = 0.9460830704D+00

```

2.10.4 DIICOS, RIICOS 余弦積分

(1) 機能

余弦積分

$$\text{Ci}(x) = - \int_x^{\infty} \frac{\cos t}{t} dt$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL DIICOS (XI, XO, IERR)

単精度サブルーチン:

CALL RIICOS (XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 x
2	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	$\text{Ci}(x)$ の値
3	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $0.0 \leq \text{XI} \leq M$ ここで, $M = \{ \text{倍精度: } 2^{50}\pi, \text{単精度: } 2^{18}\pi \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
2000	XI = 0.0 (overflow)	XO =(最小値) を返す.
3000	制限条件 (a) を満足しなかった.	処理を打ち切る. (注意事項 (a) 参照)

(6) 注意事項

(a) IERR = 3000 で XI が十分大きいときは, $Ci(x)$ の値は 0.0 に非常に近い値である.

(7) 使用例

(a) 問題

$Ci(x)$ の値を $x = 1.0$ について求める.

(b) 入力データ

XI = 1.0

(c) 主プログラム

```

PROGRAM BIICOS
! *** EXAMPLE OF DIICOS ***
IMPLICIT REAL(8) (A-H,O-Z)
READ (5,*) XI
WRITE (6,1000) XI
CALL DIICOS(XI,X0,IERR)
WRITE(6,2000) IERR,X0
1000 FORMAT(' ',/,/,5X,'*** DIICOS ***',/,/,6X,'** INPUT **',&
/,/,8X,'XI = ',F6.2 )
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF CI(X)',/,/,10X,'X0 = ',D18.10)
END

```

(d) 出力結果

```

*** DIICOS ***
** INPUT **
  XI =   1.00

** OUTPUT**
  IERR =     0
  VALUE OF CI(X)
    XO =  0.3374039229D+00

```

2.10.5 WIIFSI, VIIFSI フレネル正弦積分

(1) 機能

$x = X_i$ に対してフレネル正弦積分

$$S(x) = \int_0^x \sin\left(\frac{\pi}{2}t^2\right)dt$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL WIIFSI (NV, XI, XO, IERR)

単精度サブルーチン:

CALL VIIFSI (NV, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力データ数
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	入 力	変数値 X_i
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	出 力	$S(X_i)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(b) $|XI(i)| \leq M$

ここで, $M = \{ \text{倍精度: } 47453132.0, \text{単精度: } 724.07734 \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
3000+i	XI(i) が制限条件 (b) を満足しなかった.	処理を打ち切る. (注意事項 (b) 参照)

(6) 注意事項

(a) フレネル正弦積分が $S(x) = \frac{1}{\sqrt{2\pi}} \int_0^x \frac{\sin t}{\sqrt{t}} dt$ と表されているときは, XI(i) に $\sqrt{\frac{2x}{\pi}}$ の値を入力すること.

(b) IERR=3000+i で $|x|$ が十分大きいときは ($x = XI(i)$), $S(x)$ の値は $0.5(XI(i) > 0.0$ のとき) または $-0.5(XI(i) < 0.0$ のとき) に非常に近い値である.

(7) 使用例

(a) 問題

$S(x)$ の値を $x = 0.0, 0.1, \dots, 0.9$ について求める.

(b) 主プログラム

```

PROGRAM EIIFSI
IMPLICIT REAL(8) (A-H,O-Z)
PARAMETER (NV=10)
REAL(8) XI(NV), XO(NV)
CHARACTER*6 CNAME, CFNC
PARAMETER( CNAME='WIIFSI', CFNC=' S' )
!
DNV=NV
DO 1000 I=1,NV
  XI(I)=(I-1)/DNV
1000 CONTINUE
!
CALL WIIFSI( NV, XI, XO, IERR )
!
WRITE(6,6000) CNAME
WRITE(6,6100)
DO 2000 I=1,NV
  WRITE(6,6200) I,XI(I)
2000 CONTINUE
!
WRITE(6,6300)
WRITE(6,6400) IERR
DO 3000 I=1,NV
  WRITE(6,6500) CFNC,XI(I), XO(I)
3000 CONTINUE
STOP
!
6000 FORMAT(1X,'*** ',A6,' *')
6100 FORMAT(1X,'*** INPUT *')
6200 FORMAT(1X,'XI(',I2,',)=',F10.6 )
6300 FORMAT(1X,'*** OUTPUT *')
6400 FORMAT(1X,'IERR=',I5 )
6500 FORMAT(1X,A6,'(',F10.6,')=',F10.6 )
END

```

(c) 出力結果

```

*** WIIFSI *
*** INPUT *
XI( 1)= 0.000000
XI( 2)= 0.100000
XI( 3)= 0.200000
XI( 4)= 0.300000
XI( 5)= 0.400000
XI( 6)= 0.500000
XI( 7)= 0.600000
XI( 8)= 0.700000
XI( 9)= 0.800000
XI(10)= 0.900000
*** OUTPUT *
IERR= 0
S( 0.000000)= 0.000000
S( 0.100000)= 0.000524
S( 0.200000)= 0.004188
S( 0.300000)= 0.014117
S( 0.400000)= 0.033359
S( 0.500000)= 0.064732
S( 0.600000)= 0.110540
S( 0.700000)= 0.172136
S( 0.800000)= 0.249341
S( 0.900000)= 0.339776

```

2.10.6 WIIFCO, VIIFCO フレネル余弦積分

(1) 機能

$x = X_i$ に対してフレネル余弦積分

$$C(x) = \int_0^x \cos\left(\frac{\pi}{2}t^2\right)dt$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL WIIFCO (NV, XI, XO, IERR)

単精度サブルーチン:

CALL VIIFCO (NV, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER(4)} \\ 64 \text{ ビット整数版では INTEGER(8)} \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力データ数
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	入 力	変数値 X_i
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	出 力	$C(X_i)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(b) $|XI(i)| \leq M$

ここで, $M = \{ \text{倍精度: } 47453132.0, \text{単精度: } 724.07734 \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
3000+i	XI(i) が制限条件 (b) を満足しなかった.	処理を打ち切る. (注意事項 (b) 参照)

(6) 注意事項

(a) フレネル余弦積分が $C(x) = \frac{1}{\sqrt{2\pi}} \int_0^x \frac{\cos t}{\sqrt{t}} dt$ と表されているときは, XI(i) に $\sqrt{\frac{2x}{\pi}}$ の値を入力すること.

(b) IERR = 3000 + i で $|x|$ が十分大きいときは ($x = XI(i)$), $C(x)$ の値は $0.5(XI(i) > 0.0$ のとき) または $-0.5(XI(i) < 0.0$ のとき) に非常に近い値である.

(7) 使用例

(a) 問題

$C(x)$ の値を $x = 0.0, 0.1, \dots, 0.9$ について求める.

(b) 主プログラム

```

PROGRAM EIIFCO
IMPLICIT REAL(8) (A-H,O-Z)
PARAMETER (NV=10)
REAL(8) XI(NV), XO(NV)
CHARACTER*6 CNAME, CFNC
PARAMETER( CNAME='WIIFCO', CFNC=' C' )
!
DNV=NV
DO 1000 I=1,NV
  XI(I)=(I-1)/DNV
1000 CONTINUE
!
CALL WIIFCO( NV, XI, XO, IERR )
!
WRITE(6,6000) CNAME
WRITE(6,6100)
DO 2000 I=1,NV
  WRITE(6,6200) I,XI(I)
2000 CONTINUE
!
WRITE(6,6300)
WRITE(6,6400) IERR
DO 3000 I=1,NV
  WRITE(6,6500) CFNC,XI(I), XO(I)
3000 CONTINUE
STOP
!
6000 FORMAT(1X,'*** ',A6,' *')
6100 FORMAT(1X,'*** INPUT *')
6200 FORMAT(1X,'XI(',I2,')=',F10.6 )
6300 FORMAT(1X,'*** OUTPUT *')
6400 FORMAT(1X,'IERR=',I5 )
6500 FORMAT(1X,A6,'(',F10.6,')=',F10.6 )
END

```

(c) 出力結果

```

*** WIIFCO *
*** INPUT *
XI( 1)= 0.000000
XI( 2)= 0.100000
XI( 3)= 0.200000
XI( 4)= 0.300000
XI( 5)= 0.400000
XI( 6)= 0.500000
XI( 7)= 0.600000
XI( 8)= 0.700000
XI( 9)= 0.800000
XI(10)= 0.900000
*** OUTPUT *
IERR= 0
C( 0.000000)= 0.000000
C( 0.100000)= 0.099998
C( 0.200000)= 0.199921
C( 0.300000)= 0.299401
C( 0.400000)= 0.397481
C( 0.500000)= 0.492344
C( 0.600000)= 0.581095
C( 0.700000)= 0.659652
C( 0.800000)= 0.722844
C( 0.900000)= 0.764823

```

2.10.7 WIIDAW, VIIDAW ドーソン積分

(1) 機能

$x = X_i$ に対してドーソン積分

$$e^{-x^2} \int_0^x e^{t^2} dt$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL WIIDAW (NV, XI, XO, IERR)

単精度サブルーチン:

CALL VIIDAW (NV, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力データ数
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	入 力	変数値 X_i
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	出 力	$e^{-X_i^2} \int_0^{X_i} e^{t^2} dt$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.

(6) 注意事項

なし

(7) 使用例

(a) 問題

$e^{-x^2} \int_0^x e^{t^2} dt$ の値を $x = 0.0, 0.1, \dots, 0.9$ について求める.

(b) 主プログラム

```

PROGRAM EIIDAW
IMPLICIT REAL(8)(A-H,O-Z)
PARAMETER (NV=10)
REAL(8) XI(NV), XO(NV)
CHARACTER*6 CNAME, CFNC
PARAMETER( CNAME='WIIDAW', CFNC='DAWSON' )
!
DNV=NV
DO 1000 I=1,NV
    XI(I)=(I-1)/DNV
1000 CONTINUE
!
CALL WIIDAW( NV, XI, XO, IERR )
!
WRITE(6,6000) CNAME
WRITE(6,6100)
DO 2000 I=1,NV
    WRITE(6,6200) I,XI(I)
2000 CONTINUE
!
WRITE(6,6300)
WRITE(6,6400) IERR
DO 3000 I=1,NV
    WRITE(6,6500) CFNC,XI(I), XO(I)
3000 CONTINUE
STOP
!
6000 FORMAT(1X,'*** ',A6,' *')
6100 FORMAT(1X,'*** INPUT *')
6200 FORMAT(1X,'XI(',I2,')=',F10.6 )
6300 FORMAT(1X,'*** OUTPUT *')
6400 FORMAT(1X,'IERR=',I5 )
6500 FORMAT(1X,A6,'(',F10.6,')=',F10.6 )
END
    
```

(c) 出力結果

```

*** WIIDAW *
*** INPUT *
XI( 1)= 0.000000
XI( 2)= 0.100000
XI( 3)= 0.200000
XI( 4)= 0.300000
XI( 5)= 0.400000
XI( 6)= 0.500000
XI( 7)= 0.600000
XI( 8)= 0.700000
XI( 9)= 0.800000
XI(10)= 0.900000
*** OUTPUT *
IERR= 0
DAWSON( 0.000000)= 0.000000
DAWSON( 0.100000)= 0.099336
DAWSON( 0.200000)= 0.194751
DAWSON( 0.300000)= 0.282632
DAWSON( 0.400000)= 0.359943
DAWSON( 0.500000)= 0.424436
DAWSON( 0.600000)= 0.474763
DAWSON( 0.700000)= 0.510504
DAWSON( 0.800000)= 0.532102
DAWSON( 0.900000)= 0.540724
    
```

2.10.8 WIICND, VIICND 正規分布関数

(1) 機能

$x = X_i$ に対して正規分布関数

$$\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_0^x e^{-\frac{t^2}{2}} dt$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL WIICND (NV, XI, XO, IERR)

単精度サブルーチン:

CALL VIICND (NV, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER(4)} \\ 64 \text{ ビット整数版では INTEGER(8)} \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力データ数
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	入 力	変数値 X_i
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	出 力	$\Phi(X_i)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.

(6) 注意事項

なし

(7) 使用例

(a) 問題

$\Phi(x)$ の値を $x = 0.0, 0.1, \dots, 0.9$ について求める.

(b) 主プログラム

```

PROGRAM EIICND
IMPLICIT REAL(8) (A-H,O-Z)
PARAMETER (NV=10)
REAL(8) XI(NV), XO(NV)
CHARACTER*6 CNAME, CFNC
PARAMETER( CNAME='WIICND', CFNC=' PHI' )
!
DNV=NV
DO 1000 I=1,NV
  XI(I)=(I-1)/DNV
1000 CONTINUE
!
CALL WIICND( NV, XI, XO, IERR )
!
WRITE(6,6000) CNAME
WRITE(6,6100)
DO 2000 I=1,NV
  WRITE(6,6200) I,XI(I)
2000 CONTINUE
!
WRITE(6,6300)
WRITE(6,6400) IERR
DO 3000 I=1,NV
  WRITE(6,6500) CFNC,XI(I), XO(I)
3000 CONTINUE
STOP
!
6000 FORMAT(1X,'*** ',A6,' *')
6100 FORMAT(1X,'*** INPUT *')
6200 FORMAT(1X,'XI(',I2,')=',F10.6 )
6300 FORMAT(1X,'*** OUTPUT *')
6400 FORMAT(1X,'IERR=',I5 )
6500 FORMAT(1X,A6,'(',F10.6,')=',F10.6 )
END

```

(c) 出力結果

```

*** WIICND *
*** INPUT *
XI( 1)= 0.000000
XI( 2)= 0.100000
XI( 3)= 0.200000
XI( 4)= 0.300000
XI( 5)= 0.400000
XI( 6)= 0.500000
XI( 7)= 0.600000
XI( 8)= 0.700000
XI( 9)= 0.800000
XI(10)= 0.900000
*** OUTPUT *
IERR= 0
PHI( 0.000000)= 0.000000
PHI( 0.100000)= 0.039828
PHI( 0.200000)= 0.079260
PHI( 0.300000)= 0.117911
PHI( 0.400000)= 0.155422
PHI( 0.500000)= 0.191462
PHI( 0.600000)= 0.225747
PHI( 0.700000)= 0.258036
PHI( 0.800000)= 0.288145
PHI( 0.900000)= 0.315940

```

2.10.9 WIICNC, VIICNC 余正規分布関数

(1) 機能

$x = X_i$ に対して余正規分布関数

$$\Psi(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{-\frac{t^2}{2}} dt$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL WIICNC (NV, XI, XO, IERR)

単精度サブルーチン:

CALL VIICNC (NV, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力データ数
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	入 力	変数値 X_i
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	出 力	$\Psi(X_i)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1000	XI(i) > M であった. (注意事項 (a) 参照) (underflow)	XO(i) = 0.0 を返す.
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.

(6) 注意事項

(a) このサブルーチンで IERR=1000 となるときの M の値は以下の通りである.

$M = \{ \text{倍精度: } 38.485, \text{ 単精度: } 13.0 \}$

(7) 使用例

(a) 問題

$\Phi(x)$ の値を $x = 0.0, 0.1, 0.2, \dots, 0.9$ について求める.

(b) 主プログラム

```

PROGRAM EIICNC
IMPLICIT REAL(8) (A-H,O-Z)
PARAMETER (NV=10)
REAL(8) XI(NV), XO(NV)
CHARACTER*6 CNAME, CFNC
PARAMETER( CNAME='WIICNC', CFNC=' PSI' )
!
DNV=NV
DO 1000 I=1,NV
  XI(I)=(I-1)/DNV
1000 CONTINUE
!
CALL WIICNC( NV, XI, XO, IERR )
!
WRITE(6,6000) CNAME
WRITE(6,6100)
DO 2000 I=1,NV
  WRITE(6,6200) I,XI(I)
2000 CONTINUE
!
WRITE(6,6300)
WRITE(6,6400) IERR
DO 3000 I=1,NV
  WRITE(6,6500) CFNC,XI(I), XO(I)
3000 CONTINUE
STOP
!
6000 FORMAT(1X,'*** ',A6,' *')
6100 FORMAT(1X,'*** INPUT *')
6200 FORMAT(1X,'XI(',I2,')=',F10.6 )
6300 FORMAT(1X,'*** OUTPUT *')
6400 FORMAT(1X,'IERR=',I5 )
6500 FORMAT(1X,A6,'(',F10.6,')=',F10.6 )
END

```

(c) 出力結果

```

*** WIICNC *
*** INPUT *
XI( 1)= 0.000000
XI( 2)= 0.100000
XI( 3)= 0.200000
XI( 4)= 0.300000
XI( 5)= 0.400000
XI( 6)= 0.500000
XI( 7)= 0.600000
XI( 8)= 0.700000
XI( 9)= 0.800000
XI(10)= 0.900000
*** OUTPUT *
IERR= 0
PSI( 0.000000)= 0.500000
PSI( 0.100000)= 0.460172
PSI( 0.200000)= 0.420740
PSI( 0.300000)= 0.382089
PSI( 0.400000)= 0.344578
PSI( 0.500000)= 0.308538
PSI( 0.600000)= 0.274253
PSI( 0.700000)= 0.241964
PSI( 0.800000)= 0.211855
PSI( 0.900000)= 0.184060

```

2.11 誤差関数に関連した関数

2.11.1 WIERRF, VIERRF

誤差関数

(1) 機能

$x = X_i$ に対して、誤差関数 $\text{Erf}(x)$ の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL WIERRF (NV, XV, YV, IERR)

単精度サブルーチン:

CALL VIERRF (NV, XV, YV, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入力	入力値の個数
2	XV	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	入力	x_i
3	YV	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	出力	$\text{Erf}(x_i)$
4	IERR	I	1	出力	エラーインディケータ

(4) 制限条件

(a) $NV > 0$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.

(6) 注意事項

なし

(7) 使用例

(a) 問題

x_i を、関数値が $1 - 0.1^i$ となるように与えて、 $i = 1, 2, \dots, 10$ に対して $1 - \text{Erf}(x_i)$ の値を求める。

(b) 主プログラム

```

PROGRAM EIERRF
IMPLICIT REAL(8) (A-H,O-Z)
INTEGER NV
PARAMETER( NV = 10 )
REAL(8) XV(NV), YV(NV)
ITER=0
DO 1000 I=1,NV
CALL DIERRF(1.D-1**I, XV(I), ITER, IERR)
IF(IERR.GT.0) STOP
1000 CONTINUE
WRITE(6,10)
WRITE(6,20)
CALL WIERRF(NV, XV, YV, IERR)
DO 2000 I=1,NV
WRITE(6,6000) I, XV(I)
2000 CONTINUE
WRITE(6,30)
WRITE(6,40) IERR
DO 3000 I=1,NV
WRITE(6,6500) I, 1.DO - YV(I)
3000 CONTINUE
STOP
10 FORMAT(1X,' *** WIERRF *** ',/,/)
20 FORMAT(1X,' *** INPUT *** ',/,/)
30 FORMAT(1X,/,/,', *** OUTPUT *** ',/,/)
40 FORMAT(1X,' IERR = ',I4,/,/)
6000 FORMAT(1X,I2,' TH INPUT VALUES = ',E15.7)
6500 FORMAT(1X,I2,' TH OUTPUT VALUES = ',E15.7)
END

```

(c) 出力結果

```

*** WIERRF ***

*** INPUT ***

1 TH INPUT VALUES = 0.1163087E+01
2 TH INPUT VALUES = 0.1821386E+01
3 TH INPUT VALUES = 0.2326754E+01
4 TH INPUT VALUES = 0.2751064E+01
5 TH INPUT VALUES = 0.3123413E+01
6 TH INPUT VALUES = 0.3458911E+01
7 TH INPUT VALUES = 0.3766563E+01
8 TH INPUT VALUES = 0.4052237E+01
9 TH INPUT VALUES = 0.4320005E+01
10 TH INPUT VALUES = 0.4572825E+01

*** OUTPUT ***

IERR = 0

1 TH OUTPUT VALUES = 0.1000000E+00
2 TH OUTPUT VALUES = 0.1000000E-01
3 TH OUTPUT VALUES = 0.1000000E-02
4 TH OUTPUT VALUES = 0.1000000E-03
5 TH OUTPUT VALUES = 0.1000000E-04
6 TH OUTPUT VALUES = 0.1000000E-05
7 TH OUTPUT VALUES = 0.1000000E-06
8 TH OUTPUT VALUES = 0.1000000E-07
9 TH OUTPUT VALUES = 0.1000000E-08
10 TH OUTPUT VALUES = 0.1000000E-09

```

2.11.2 WIERFC, VIERFC

余誤差関数

(1) 機能

$x = X_i$ に対して、余誤差関数 $\text{Erfc}(x)$ の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL WIERFC (NV, XV, YV, IERR)

単精度サブルーチン:

CALL VIERFC (NV, XV, YV, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力値の個数
2	XV	$\begin{cases} D \\ R \end{cases}$	NV	入 力	x_i
3	YV	$\begin{cases} D \\ R \end{cases}$	NV	出 力	$\text{Erfc}(x_i)$
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV > 0$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.

(6) 注意事項

なし

(7) 使用例

(a) 問題

x_i を、関数値が 0.1^i となるように与えて、 $i = 1, 2, \dots, 10$ に対して $\text{Erfc}(x_i)$ の値を求める。

(b) 主プログラム

```

PROGRAM EIERFC
IMPLICIT REAL(8) (A-H,O-Z)
INTEGER NV
PARAMETER( NV = 10 )
REAL(8) XV(NV), YV(NV)
DO 1000 I=1,NV
  CALL DIERF(1.D-1**I, XV(I), 0, IERR)
  IF(IERR.GT.0) STOP
1000 CONTINUE
WRITE(6,10)
WRITE(6,20)
CALL WIERFC(NV, XV, YV, IERR)
DO 2000 I=1,NV
WRITE(6,6000) I, XV(I)
2000 CONTINUE
WRITE(6,30)
WRITE(6,40) IERR
DO 3000 I=1,NV
WRITE(6,6500) I, YV(I)
3000 CONTINUE
STOP
10 FORMAT(1X,' *** WIERFC *** ',/,/)
20 FORMAT(1X,' *** INPUT *** ',/,/)
30 FORMAT(1X,/,/,', *** OUTPUT *** ',/,/)
40 FORMAT(1X,' IERR = ',I4,/,/)
6000 FORMAT(1X,I2,' TH INPUT VALUES = ',E15.7)
6500 FORMAT(1X,I2,' TH OUTPUT VALUES = ',E15.7)
END

```

(c) 出力結果

```

*** WIERFC ***

*** INPUT ***

1 TH INPUT VALUES = 0.1163087E+01
2 TH INPUT VALUES = 0.1821386E+01
3 TH INPUT VALUES = 0.2326754E+01
4 TH INPUT VALUES = 0.2751064E+01
5 TH INPUT VALUES = 0.3123413E+01
6 TH INPUT VALUES = 0.3458911E+01
7 TH INPUT VALUES = 0.3766563E+01
8 TH INPUT VALUES = 0.4052237E+01
9 TH INPUT VALUES = 0.4320005E+01
10 TH INPUT VALUES = 0.4572825E+01

*** OUTPUT ***

IERR = 0

1 TH OUTPUT VALUES = 0.1000000E+00
2 TH OUTPUT VALUES = 0.1000000E-01
3 TH OUTPUT VALUES = 0.1000000E-02
4 TH OUTPUT VALUES = 0.1000000E-03
5 TH OUTPUT VALUES = 0.1000000E-04
6 TH OUTPUT VALUES = 0.1000000E-05
7 TH OUTPUT VALUES = 0.1000000E-06
8 TH OUTPUT VALUES = 0.1000000E-07
9 TH OUTPUT VALUES = 0.1000000E-08
10 TH OUTPUT VALUES = 0.1000000E-09

```

2.11.3 DIERF, RIERF 余誤差関数の逆関数

(1) 機能

余誤差関数 Erfc の逆関数を求める。すなわち、 $0 < x \leq 1$ に対して $\text{Erfc}(y) = x$ となる y を求める。

(2) 使用法

倍精度サブルーチン:

CALL DIERF (X, Y, ITER, IERR)

単精度サブルーチン:

CALL RIERF (X, Y, ITER, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	X	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入力	x
2	Y	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出力	y
3	ITER	I	1	入力	最大反復回数 (既定値:10)(注意事項 (d) 参照)
4	IERR	I	1	出力	エラーインディケータ

(4) 制限条件

(a) $0 < X \leq 1$

(b) $\text{ITER} \geq 1$ (既定値にするために 0 または負の値を入力する場合は除く)

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1000	$X=1$ であった.	$Y=0$ とする.
3000	$X > 1$ であった.	処理を打ち切る.
3500	$X \leq 0$ であった.	
4000	反復回数が最大反復回数でも収束しなかつた.	

(6) 注意事項

- (a) $\text{Erfc}(y) = x$ を満たす y を $0 < x \leq 1$ に対して求めるので、 $2 > x > 1$ に対しては、

$$\frac{2}{\sqrt{\pi}} \int_{-y}^{\infty} e^{-t^2} dt = 2 - x$$

を用いて、 $y = -\text{Erfc}^{-1}(2 - x)$ とすればよい。

- (b) $\text{Erfc}^{-1}(0) = \infty$, $\text{Erfc}^{-1}(2) = -\infty$ である。
 (c) $\text{Erf}(x)$ の逆関数は $\text{Erfc}^{-1}(1 - x)$ である。
 (d) 最大反復回数は、ITER が 0 または負のときは、10 に設定される。

(7) 使用例

- (a) 問題

$\text{Erf}(y) = 0.9999$ となる y を求め、結果を確かめる。

- (b) 主プログラム

```

PROGRAM BIIERF
IMPLICIT REAL(8) (A-H,O-Z)
PARAMETER(ONE=1.D0)
X=ONE-0.9999D0
WRITE(6,90)
WRITE(6,91)
WRITE(6,5000) X
CALL DIERF(X, Y, 0, IERR)
WRITE(6,92)
WRITE(6,93) IERR
CALL WIERRF(1, Y, Z, IERR)
IF(IERR.NE.0) STOP
WRITE(6,6000) ONE-X, Y, Z
STOP
90 FORMAT(1X,' *** DIERF *** ',/,/)
91 FORMAT(1X,' *** INPUT *** ',/,/)
92 FORMAT(1X,' *** OUTPUT *** ',/,/)
93 FORMAT(1X,' IERR= ',I4,/,/)
5000 FORMAT(1X,' X = ',E15.7,/,/)
6000 FORMAT(1X,' 1-X=',E15.7,' OUTPUT Y=',&
           E15.7,&
           ' TEST VALUE =',E15.7)
END
    
```

- (c) 出力結果

```

*** DIERF ***

*** INPUT ***

X =  0.1000000E-03

*** OUTPUT ***

IERR=  0

1-X=  0.9999000E+00 OUTPUT Y=  0.2751064E+01 TEST VALUE =  0.9999000E+00
    
```

2.11.4 JIJERF, IIJERF 複素変数の誤差関数

(1) 機能

$z=Z_i$ に対して、複素変数の誤差関数 $e^{-z^2} \operatorname{Erfc}(-iz)$ を求める。

(2) 使用法

倍精度サブルーチン:

CALL JIJERF (NV, Z, W, IERR)

単精度サブルーチン:

CALL IIJERF (NV, Z, W, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力値の個数
2	Z	$\begin{Bmatrix} Z \\ C \end{Bmatrix}$	NV	入 力	z
3	W	$\begin{Bmatrix} Z \\ C \end{Bmatrix}$	NV	出 力	$e^{-z^2} \operatorname{Erfc}(-iz)$
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV > 0$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
4000	オーバフローが発生した. (注意事項 (b) 参照)	

(6) 注意事項

(a) 虚数部が 0 に近いときは、若干精度が落ちる (倍精度では 10^{-10} 程度の誤差).

(b) $\Im(z)$ が負かつ $-\Re(z^2)$ が $\log(\text{正の最大値})$ 程度のときは、オーバフローが起きる.

(7) 使用例

(a) 問題

積分路 $|z - 3i| = 1$ について、コーシーの積分表示を確かめる。

(b) 主プログラム

```

PROGRAM KIIERF
IMPLICIT COMPLEX(8) (A-H,O-Z)
PARAMETER( N=100 , Z3=(0.DO,3.DO) )
PARAMETER( Z1=(0.DO,1.DO) )
REAL(8) R1, RI1, R3, RI3
CHARACTER*10 HEAD1,HEAD2
COMPLEX(8) ZV(N+1), WV(N+1)
HEAD1='CAUCHY : '
HEAD2='TRUE : '
PAI=ACOS(-1.DO)
S=(0.DO,0.DO)
WRITE(6,10)
WRITE(6,20)
WRITE(6,30) N+1
DO 1000 I=1,100
  ZV(I)=Z3+EXP(2*PAI*Z1*I/100.DO)
1000 CONTINUE
  ZV(N+1)=Z3
  WRITE(6,40)
  CALL JIIERF(N+1, ZV, WV, IERR)
  DO 2000 I=1,100
    S=S+WV(I)/100.DO
2000 CONTINUE
  W3=WV(N+1)
  R1=S
  RI1=S/Z1
  R3=W3
  RI3=W3/Z1
  WRITE(6,6000) HEAD1,R1,RI1
  WRITE(6,6000) HEAD2,R3,RI3
  STOP
10 FORMAT(1X,' *** JIIERF *** ',/,/)
20 FORMAT(1X,' *** INPUT *** ',/,/)
30 FORMAT(1X,' NV = ',I3,/,/)
40 FORMAT(1X,' *** OUTPUT *** ',/,/)
6000 FORMAT(1X,A10,5X,E15.7,5X,E15.7,/,/)
END

```

(c) 出力結果

```

*** JIIERF ***

*** INPUT ***

NV = 101

*** OUTPUT ***

CAUCHY :          0.1790012E+00          -0.7589415E-18

TRUE   :          0.1790012E+00          0.6251103E-19

```

2.12 ルジャンドル陪関数

2.12.1 DILEG1, RILEG1

第1種ルジャンドル陪関数

(1) 機能

第1種ルジャンドル陪関数

$$P_n^m(x) = (|1-x^2|)^{\frac{m}{2}} \frac{d^m P_n(x)}{dx^m}$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL DILEG1 (N, M, XI, XO, IERR)

単精度サブルーチン:

CALL RILEG1 (N, M, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER(4)} \\ 64 \text{ ビット整数版では INTEGER(8)} \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入力	次数 n
2	M	I	1	入力	陪数 m
3	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入力	変数値 x
4	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出力	$P_n^m(x)$ の値
5	IERR	I	1	出力	エラーインディケータ

(4) 制限条件

(a) $M < 0$ のとき $|M| \leq N$ ($N < 0$ なら $|M| \leq -N - 1$)

(b) $|M| \leq N$ ($N < 0$ なら $|M| \leq -N - 1$) のとき

$|M| < M_1$

ここで, $M_1 = \{ \text{倍精度: 150, 単精度: 27} \}$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
2000	$ XI > (\text{最大値})^{1/N}/2$ および $N \geq 2$ (overflow)	$XI \geq 0.0$ のとき $XO = (\text{最大値})$ $XI < 0.0$ のとき $XO = (\text{最大値}) \times (-1)^N$ を返す.
3000	制限条件 (a) または (b) を満足しなかった.	処理を打ち切る.
4000	演算の途中でオーバフローが発生した.	

(6) 注意事項

(a) $|x| > 1.0$ のときは,

$$P_n^m(x) = (x^2 - 1.0)^{\frac{m}{2}} \frac{d^m P_n(x)}{dx^m}$$

(Hobson のルジャンドル陪関数) を計算し, $|x| \leq 1.0$ のときは,

$$P_n^m(x) = (1.0 - x^2)^{\frac{m}{2}} \frac{d^m P_n(x)}{dx^m}$$

(Ferrers のルジャンドル陪関数) を計算する.

(b) このサブルーチンは精度を保つため, 内部で倍長演算を用いている.

(c) $|x| \leq 1.0$ のとき,

$$P_n^m(x) = (-1)^m (1.0 - x^2)^{\frac{m}{2}} \frac{d^m P_n(x)}{dx^m}$$

と定義する場合や, $(n - m)!$ をかけて陪関数と定義する場合もあるので注意を要する.

(d) 次数 n が大きく, 多くの XI についての値を求める場合は, 2.15.3 $\left\{ \begin{array}{l} \text{WINPLG} \\ \text{VINPLG} \end{array} \right\}$ を使用すると良い. 第1種ルジャンドル陪関数 $P_n^m(x)$ と正規化された球面調和関数 $P_n^{*m}(x)$ の間の関係は,

$$P_n^{*0}(x) = \sqrt{\frac{2n+1}{4\pi}} P_n^0(x) \quad (-1 \leq x \leq 1),$$

$$P_n^{*m}(x) = \sqrt{\frac{2n+1}{4\pi}} \sqrt{2 \frac{(n-m)!}{(n+m)!}} P_n^m(x) \quad (-1 \leq x \leq 1; m = 1, 2, \dots, n)$$

である. m が大きいときは, 第1種ルジャンドル陪関数の絶対値が階乗的に大きくなることに注意する必要がある.

(7) 使用例

(a) 問題

$P_n^m(x)$ の値を $n = 4, m = 2, x = 0.8$ について求める.

(b) 入力データ

$N = 4, M = 2, XI = 0.8$

(c) 主プログラム

```

PROGRAM BILEG1
! *** EXAMPLE OF DILEG1 ***
IMPLICIT REAL(8) (A-H,O-Z)
READ (5,*) N
READ (5,*) M
READ (5,*) XI
WRITE (6,1000) N,M,XI
CALL DILEG1(N,M,XI,XO,IERR)
WRITE (6,2000) IERR,XO
1000 FORMAT(' ',/,/,5X,'*** DILEG1 ***',/,/,6X,'** INPUT **',&
/,/,8X,'N = ',I3,5X,'M = ',I3,5X,'XI = ',F6.2 )
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF PNM(X)',/,/,10X,'XO = ',D18.10)
END

```

(d) 出力結果

```

*** DILEG1 ***

** INPUT **

N = 4      M = 2      XI = 0.80

** OUTPUT**

IERR = 0

VALUE OF PNM(X)

XO = 0.9396000000D+01

```


2.12.2 DILEG2, RILEG2 第2種ルジャンドル陪関数

(1) 機能

第2種ルジャンドル陪関数

$$Q_n^m(x) = (|1-x^2|)^{\frac{m}{2}} \frac{d^m Q_n(x)}{dx^m}$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL DILEG2 (N, M, XI, XO, IERR)

単精度サブルーチン:

CALL RILEG2 (N, M, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: { 32ビット整数版では INTEGER(4) }
 R:単精度実数型 C:単精度複素数型 { 64ビット整数版では INTEGER(8) }

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入力	次数 n
2	M	I	1	入力	倍数 m
3	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入力	変数値 x
4	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出力	$Q_n^m(x)$ の値
5	IERR	I	1	出力	エラーインディケータ

(4) 制限条件

- (a) $N \geq 0$
- (b) $M < 0$ のとき $|M| \leq N$
- (c) $|XI| \neq 1.0$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a), (b) または (c) を満足しなかつた.	処理を打ち切る.
4000	演算の途中でオーバフローが発生した.	
4100	級数展開式の計算が収束しなかつた.	

(6) 注意事項

(a) $|x| > 1.0$ のときは,

$$Q_n^m(x) = (x^2 - 1.0)^{\frac{m}{2}} \frac{d^m Q_n(x)}{dx^m}$$

(Hobson のルジャンドル陪関数) を計算し, $|x| \leq 1.0$ のときは,

$$Q_n^m(x) = (1.0 - x^2)^{\frac{m}{2}} \frac{d^m Q_n(x)}{dx^m}$$

(Ferrers のルジャンドル陪関数) を計算する.

(b) このサブルーチンは精度を保つため, 内部で倍長演算を用いている.

(c) $|x| > 1.0$ または $|x| < 1.0$ のときに本定義の $Q_n^m(x)$ に $(-1)^m$ をかけて定義する場合や, $(n - m)!$ をかけて陪関数と定義する場合もあるので注意を要する.

(7) 使用例

(a) 問題

 $Q_n^m(x)$ の値を $n = 4$, $m = 2$, $x = 1.8$ について求める.

(b) 入力データ

N = 4, M = 2, XI = 1.8

(c) 主プログラム

```

PROGRAM BILEG2
! *** EXAMPLE OF DILEG2 ***
IMPLICIT REAL(8) (A-H,O-Z)
READ (5,*) N
READ (5,*) M
READ (5,*) XI
WRITE(6,1000) N,M,XI
CALL DILEG2(N,M,XI,XO,IERR)
WRITE(6,2000) IERR,XO
1000 FORMAT(' ',/,5X,'*** DILEG2 ***',/,/,6X,'** INPUT **',&
/,/,8X,'N = ',I3,5X,'M = ',I3,5X,'XI = ',F6.2 )
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF QNM(X)',/,/,10X,'XO = ',D18.10)
END

```

(d) 出力結果

```

*** DILEG2 ***
** INPUT **
   N =   4      M =   2      XI =   1.80

** OUTPUT**
IERR =       0
VALUE OF QNM(X)
XO =   0.7031257577D-01

```

2.13 直交多項式

2.13.1 DIOPLE, RIOPLE ルジャンドル多項式

(1) 機能

ルジャンドル多項式

$$P_i(x) = \frac{1}{2^i i!} \frac{d^i}{dx^i} (x^2 - 1)^i \quad (i = 0, 1, \dots, n)$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL DIOPLE (N, XI, XO, IERR)

単精度サブルーチン:

CALL RIOPLE (N, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入 力	最高次数 n
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 x
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	0 : N	出 力	$P_i(x)$ の値 ($i = 0, 1, \dots, n$)
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $N \geq 0$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
4000	演算の途中でオーバフローが発生した.	

(6) 注意事項

- (a) このサブルーチンは精度を保つため、内部で倍長演算を用いている。

(7) 使用例

(a) 問題

$P_n(x)$ の値を $n = 3, x = 0.8$ について求める。

(b) 入力データ

$N = 3, XI = 0.8$

(c) 主プログラム

```
PROGRAM BIOPLE
! *** EXAMPLE OF DIOPLE ***
IMPLICIT REAL(8) (A-H,O-Z)
DIMENSION XO(0:3)
READ (5,*) N
READ (5,*) XI
WRITE (6,1000) N,XI
CALL DIOPLE(N,XI,XO,IERR)
WRITE (6,2000) IERR,XO(3)
1000 FORMAT(' ',/,/,5X,'*** DIOPLE ***',/,/,6X,'** INPUT **',&
/,/,8X,'N = ',I3,5X,'XI = ',F6.2 )
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF PN(X)',/,/,10X,'XO = ',D18.10)
END
```

(d) 出力結果

```
*** DIOPLE ***
** INPUT **
N = 3    XI = 0.80

** OUTPUT**
IERR = 0
VALUE OF PN(X)
XO = 0.80000000000D-01
```

2.13.2 DIZGLW, RIZGLW ガウス・ルジャンドル積分公式

(1) 機能
 (高次数の) ガウス・ルジャンドル積分公式の積分点および重みを計算する.

(2) 使用法
 倍精度サブルーチン:
 CALL DIZGLW (N, Z, W, WORK, IERR)
 単精度サブルーチン:
 CALL RIZGLW (N, Z, W, WORK, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入 力	次数 n
2	Z	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	N	出 力	ルジャンドル多項式 $P_n(x)$ の零点 (積分点) (小さい方から順に入る)
3	W	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	N	出 力	ガウス・ルジャンドル積分公式の重み (零点の小さい方から順に入る)
4	WORK	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	N	ワーク	作業領域
5	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $N \geq 1$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1000	$N=1$ であった.	処理を続ける.
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
4000	積分点を求める処理で収束しなかった.	
5000	ルジャンドル多項式の計算の途中でオーバーフローが発生した.	

(6) 注意事項

(a) ルジャンドル多項式 $P_n(x)$ の零点はこのサブルーチンの内部で計算される。

(b) ガウス・ルジャンドル積分公式は

$$\int_{-1}^1 f(x)dx = \sum_{j=1}^n w_j f(z_j)$$

である。ここに、 z_j, w_j は $P_n(x)$ の零点とそれに対応する重みである。積分区間が $[-1, 1]$ である場合に直接に適用できるが、これ以外の積分区間については置換積分により $[-1, 1]$ に写すこと。

(c) ガウス・ルジャンドル積分公式を、振動する関数の積分に用いることは避けることが望ましい。これを例を挙げて示すために定積分

$$\int_{-1}^1 5e^{-25x^2} \cos(10Ax)dx$$

を考える。この被積分関数は両端で急激に減少する。両端を無限区間に換えて計算すると、この積分の値は $\sqrt{\pi}e^{-A^2}$ となる。また、この区間変換による誤差は 10^{-12} 程度である。分点の数を 24 としてガウス・ルジャンドル積分公式を用いると、 $A=0.3$ では 0.3×10^{-7} 程度の誤差である。 $A=2.4$ では結果値-0.0004898 が得られるが、真の値は 0.005585 程度である。このように $\cos(24x)$ 程度の振動項があるだけでも積分誤差を生じる。

(d) このサブルーチンは、高次数のガウス・ルジャンドル積分公式についても適用できる。

(7) 使用例

(a) 問題

$N=24$ として、定積分 $\int_{-1}^1 \frac{1}{1+x+x^2} dx = \frac{\pi}{\sqrt{3}}$ を求める。

(b) 主プログラム

```

PROGRAM BIZGLW
IMPLICIT REAL(8) (A-H,O-Z)
PARAMETER(N=24)
REAL(8)&
  Z(N), W(2*N)
!
  WRITE(6,10)
  WRITE(6,20)
  WRITE(6,30)N
  CALL DIZGLW(N, Z, W, W(N+1), IERR)
  WRITE(6,40)
  WRITE(6,50) IERR
  WRITE(6,60) N
  WRITE(6,4000)
  DO 500 I=1,N
  WRITE(6,5000) I, Z(I), W(I)
500 CONTINUE
  D=0
  DO 1000 I=1,N
  X=Z(I)
  F= 1.DO/(1+X+X*X)
  D=D+W(I)*F
1000 CONTINUE
  S=2.DO*ASIN(1.DO)/SQRT(3.DO)
  WRITE(6,70) N
  WRITE(6,6000) D
  WRITE(6,7000) S, D-S
  STOP
10 FORMAT(1X,' *** DIZGLW *** ',/,/)
20 FORMAT(1X,' *** INPUT *** ',/,/)
30 FORMAT(1X,' N = ',I3,/,/)
40 FORMAT(1X,' *** OUTPUT *** ',/,/)
50 FORMAT(1X,' IERR= ',I4,/,/)
60 FORMAT(1X,' TABLE FOR ',I3,' POINTS FORMULA',/,/)
70 FORMAT(1X,' GAUSS-LEGENDRE ',I3,' POINTS FORMULA',/,/)
4000 FORMAT(1X,' NO ',10X,' ZERO POINT',15X,' WEIGHT ',/,/)
5000 FORMAT(1X,' I4, 5X, E20.10,5X,E20.10)
6000 FORMAT(1X,' COMPUTED VALUE= ',E20.10,/,/)
7000 FORMAT(1X,' TRUE VALUE= ',E20.10,' ERR = ',E20.10)
END

```

(c) 出力結果

```

*** DIZGLW ***

*** INPUT ***

N = 24

*** OUTPUT ***

IERR= 0

TABLE FOR 24 POINTS FORMULA

NO          ZERO POINT          WEIGHT

 1      -0.9951872200E+00      0.1234122980E-01
 2      -0.9747285560E+00      0.2853138863E-01
 3      -0.9382745520E+00      0.4427743882E-01
 4      -0.8864155270E+00      0.5929858492E-01
 5      -0.8200019860E+00      0.7334648141E-01
 6      -0.7401241916E+00      0.8619016153E-01
 7      -0.6480936519E+00      0.9761865210E-01
 8      -0.5454214714E+00      0.1074442701E+00
 9      -0.4337935076E+00      0.1155056681E+00
10      -0.3150426797E+00      0.1216704729E+00
11      -0.1911188675E+00      0.1258374563E+00
12      -0.6405689286E-01      0.1279381953E+00
13      0.6405689286E-01      0.1279381953E+00
14      0.1911188675E+00      0.1258374563E+00
15      0.3150426797E+00      0.1216704729E+00
16      0.4337935076E+00      0.1155056681E+00
17      0.5454214714E+00      0.1074442701E+00
18      0.6480936519E+00      0.9761865210E-01
19      0.7401241916E+00      0.8619016153E-01
20      0.8200019860E+00      0.7334648141E-01
21      0.8864155270E+00      0.5929858492E-01
22      0.9382745520E+00      0.4427743882E-01
23      0.9747285560E+00      0.2853138863E-01
24      0.9951872200E+00      0.1234122980E-01

GAUSS-LEGENDRE 24 POINTS FORMULA

COMPUTED VALUE= 0.1813799364E+01

TRUE VALUE= 0.1813799364E+01 ERR = -0.3774758284E-14
    
```

2.13.3 DIOPLA, RIOPLA ラゲール多項式

(1) 機能

ラゲール多項式

$$L_i(x) = \frac{e^x}{i!} \frac{d^i}{dx^i} (e^{-x} x^i) \quad (i = 0, 1, \dots, n)$$

の値を求める.

(2) 使用法

倍精度サブルーチン:

CALL DIOPLA (N, XI, XO, IERR)

単精度サブルーチン:

CALL RIOPLA (N, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入 力	最高次数 n
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 x
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	0 : N	出 力	$L_i(x)$ の値 ($i = 0, 1, \dots, n$)
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $N \geq 0$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
4000	演算の途中でオーバフローが発生した.	

(6) 注意事項

- (a) このサブルーチンは精度を保つため、内部で倍長演算を用いている。

(7) 使用例

(a) 問題

$L_n(x)$ の値を $n = 3, x = 0.8$ について求める。

(b) 入力データ

N = 3, XI = 0.8

(c) 主プログラム

```

PROGRAM BIOPLA
! *** EXAMPLE OF DIOPLA ***
IMPLICIT REAL(8) (A-H,O-Z)
DIMENSION XO(0:3)
READ (5,*) N
READ (5,*) XI
WRITE (6,1000) N,XI
CALL DIOPLA(N,XI,XO,IERR)
WRITE (6,2000) IERR,XO(3)
1000 FORMAT(' ',/,/,5X,'*** DIOPLA ***',/,/,6X,'** INPUT **',&
/,/,8X,'N =',I3,5X,'XI =',F6.2 )
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF LN(X)',/,/,10X,'XO = ',D18.10)
END

```

(d) 出力結果

```

*** DIOPLA ***
** INPUT **
  N =  3    XI =  0.80

** OUTPUT**
IERR =      0
VALUE OF LN(X)
  XO = -0.52533333333D+00

```

2.13.4 DIOPHE, RIOPHE エルミート多項式

(1) 機能

エルミート多項式

$$H_i(x) = (-1)^i e^{x^2} \frac{d^i}{dx^i} (e^{-x^2}) \quad (i = 0, 1, \dots, n)$$

の値を求める.

(2) 使用法

倍精度サブルーチン:

CALL DIOPHE (N, XI, XO, IERR)

単精度サブルーチン:

CALL RIOPHE (N, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\left\{ \begin{array}{l} 32 \text{ ビット整数版では INTEGER(4)} \\ 64 \text{ ビット整数版では INTEGER(8)} \end{array} \right\}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入 力	最高次数 n
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 x
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	0 : N	出 力	$H_i(x)$ の値 ($i = 0, 1, \dots, n$)
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $N \geq 0$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
4000	演算の途中でオーバフローが発生した.	

(6) 注意事項

- (a) このサブルーチンは精度を保つため、内部で倍長演算を用いている。

(7) 使用例

(a) 問題

$H_n(x)$ の値を $n = 3, x = 0.8$ について求める。

(b) 入力データ

$N = 3, XI = 0.8$

(c) 主プログラム

```

PROGRAM BIOPHE
! *** EXAMPLE OF DIOPHE ***
IMPLICIT REAL(8) (A-H,O-Z)
DIMENSION XO(0:3)
READ (5,*) N
READ (5,*) XI
WRITE (6,1000) N,XI
CALL DIOPHE(N,XI,XO,IERR)
WRITE (6,2000) IERR,XO(3)
1000 FORMAT(' ',/,/,5X,'*** DIOPHE ***',/,/,6X,'** INPUT **',&
/,/,8X,'N =',I3,5X,'XI =',F6.2 )
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF HN(X)',/,/,10X,'XO = ',D18.10)
END

```

(d) 出力結果

```

*** DIOPHE ***
** INPUT **
N = 3    XI = 0.80

** OUTPUT**
IERR = 0
VALUE OF HN(X)
XO = -0.5504000000D+01

```

2.13.5 DIOPCH, RIOPCH チェビシェフ多項式

(1) 機能

チェビシェフ多項式

$$T_i(x) = \frac{(-1)^i}{(2i-1)!!} \sqrt{1-x^2} \frac{d^i}{dx^i} (1-x^2)^{i-1/2} \quad (i = 0, 1, \dots, n)$$

の値を求める.

(2) 使用法

倍精度サブルーチン:

CALL DIOPCH (N, XI, XO, IERR)

単精度サブルーチン:

CALL RIOPCH (N, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER(4)} \\ 64 \text{ ビット整数版では INTEGER(8)} \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入 力	最高次数 n
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 x
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	0 : N	出 力	$T_i(x)$ の値 ($i = 0, 1, \dots, n$)
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $N \geq 0$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
4000	演算の途中でオーバーフローが発生した.	

(6) 注意事項

- (a) このサブルーチンは精度を保つため、内部で倍長演算を用いている。

(7) 使用例

(a) 問題

$T_n(x)$ の値を $n = 3, x = 0.8$ について求める。

(b) 入力データ

$N = 3, XI = 0.8$

(c) 主プログラム

```

PROGRAM BIOPCH
! *** EXAMPLE OF DIOPCH ***
IMPLICIT REAL(8) (A-H,O-Z)
DIMENSION XO(0:3)
READ (5,*) N
READ (5,*) XI
WRITE (6,1000) N,XI
CALL DIOPCH(N,XI,XO,IERR)
WRITE (6,2000) IERR,XO(3)
1000 FORMAT(' ',/,/,5X,'*** DIOPCH ***',/,/,6X,'** INPUT **',&
/,/,8X,'N = ',I3,5X,'XI = ',F6.2 )
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF TN(X)',/,/,10X,'XO = ',D18.10)
END
    
```

(d) 出力結果

```

*** DIOPCH ***
** INPUT **
   N =  3      XI =  0.80

** OUTPUT**
IERR =      0
VALUE OF TN(X)
XO = -0.3520000000D+00
    
```

2.13.6 DIOPC2, RIOPC2

第2種チェビシェフ関数

(1) 機能

第2種チェビシェフ関数

$$U_i(x) = \frac{\sqrt{1-x^2}}{i} \frac{dT_i(x)}{dx} \quad (i = 0, 1, \dots, n)$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL DIOPC2 (N, XI, XO, IERR)

単精度サブルーチン:

CALL RIOPC2 (N, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入 力	最高次数 n
2	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 x
3	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	0 : N	出 力	$U_i(x)$ の値 ($i = 0, 1, \dots, n$)
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $N \geq 0, |XI| \leq 1.0$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.

(6) 注意事項

なし

(7) 使用例

(a) 問題

$U_3(0.8)$ の値を求める.

(b) 入力データ

$N = 3, XI = 0.8$

(c) 主プログラム

```

PROGRAM BIOPC2
! *** EXAMPLE OF DIOPC2 ***
IMPLICIT REAL(8) (A-H,O-Z)
DIMENSION XO(0:3)
READ (5,*) N
READ (5,*) XI
WRITE (6,1000) N,XI
CALL DIOPC2(N,XI,XO,IERR)
WRITE (6,2000) IERR,XO(3)
1000 FORMAT(' ',/,/,5X,'*** DIOPC2 ***',/,/,6X,'** INPUT **',&
/,/,8X,'N =',I3,5X,'XI =',F6.2 )
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF UN(X)',/,/,10X,'XO = ',D18.10)
END
    
```

(d) 出力結果

```

*** DIOPC2 ***

** INPUT **

N = 3      XI = 0.80

** OUTPUT**

IERR = 0

VALUE OF UN(X)

XO = 0.9360000000D+00
    
```

2.13.7 DIOPGL, RIOPGL 一般ラゲール多項式

(1) 機能

一般ラゲール多項式

$$L_i^{(\alpha)}(x) = \frac{e^x x^{-\alpha}}{i!} \frac{d^i}{dx^i} (e^{-x} x^{i+\alpha}) \quad (i = 0, 1, \dots, n)$$

の値を求める.

(2) 使用法

倍精度サブルーチン:

CALL DIOPGL (N, ALF, XI, XO, IERR)

単精度サブルーチン:

CALL RIOPGL (N, ALF, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER(4)} \\ 64 \text{ ビット整数版では INTEGER(8)} \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入 力	最高次数 n
2	ALF	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 α
3	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 x
4	XO	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	0 : N	出 力	$L_i^{(\alpha)}(x)$ の値 ($i = 0, 1, \dots, n$)
5	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $N \geq 0$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
4000	演算の途中でオーバーフローが発生した.	

(6) 注意事項

なし

(7) 使用例

(a) 問題

$L_n^{(\alpha)}(x)$ を $n = 3$, $\alpha = 0.5$, $x = 0.8$ について求める.

(b) 入力データ

$N = 3$, $ALF = 0.5$, $XI = 0.8$

(c) 主プログラム

```

PROGRAM BIOPGL
! *** EXAMPLE OF DIOPGL ***
IMPLICIT REAL(8) (A-H,O-Z)
DIMENSION XO(0:3)
READ (5,*) N
READ (5,*) ALF
READ (5,*) XI
WRITE(6,1000) N,ALF,XI
CALL DIOPGL(N,ALF,XI,XO,IERR)
WRITE(6,2000) IERR,XO(3)
1000 FORMAT(' ',/,/,5X,'*** DIOPGL ***',/,/,6X,'** INPUT **',&
/,/,8X,'N =',I3,5X,'ALF =',F6.2,5X,'XI =',F6.2 )
2000 FORMAT(' ',/,/,6X,'** OUTPUT**',/,/,8X,'IERR = ',I5,&
/,/,8X,'VALUE OF LNA(X)',/,/,10X,'XO = ',D18.10)
END

```

(d) 出力結果

```

*** DIOPGL ***
** INPUT **
   N =   3      ALF =   0.50      XI =   0.80

** OUTPUT**
IERR =       0
VALUE OF LNA(X)
XO =  -0.2778333333D+00

```

2.14 マシユー関数

2.14.1 DIMTCE, RIMTCE

整数次マシユー関数 $ce_n(x, q)$

(1) 機能

整数次マシユー関数 $ce_n(x, q)$ の値を求める.

(2) 使用法

倍精度サブルーチン:

CALL DIMTCE (NORD, N, Q, X, CE, ISW, WORK, IERR)

単精度サブルーチン:

CALL RIMTCE (NORD, N, Q, X, CE, ISW, WORK, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NORD	I	1	入力	フーリエ展開項の個数 -1 (注意事項 (a) 参照)
2	N	I	1	入力	次数 n
3	Q	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入力	パラメータ q
4	X	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入力	変数 x
5	CE	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出力	関数値 $ce_n(x, q)$
6	ISW	I	1	入力	処理スイッチ ISW =0 : 初期設定後, 関数値計算を行う ISW =1 : 関数値計算のみ行う
				出力	初期設定が行われた場合 1 を返す
7	WORK	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	内容参照	入出力	作業領域 (係数テーブル) (注意事項 (b) 参照) 大きさ: $(2 \times \text{NORD}^2 + 6 \times \text{NORD} + 108)$
8	IERR	I	1	出力	エラーインディケータ

(4) 制限条件

(a) $2 \leq \text{NORD} \leq 50$

(b) $0 \leq N \leq 2 \times \text{NORD} + 1$

(c) ISW=0 または ISW=1 であること.

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
2000	十分な精度で計算できなくなった.	処理を打ち切る.
3000	制限条件 (a) または (b) を満足しなかった.	処理を打ち切る.

(6) 注意事項

- (a) NORD の値が小さいとき、CE は十分な精度が得られないので、NORD は大きいことが望ましい。パラメータ Q および次数 N に対する NORD の値の目安は、次の範囲である。

$$\min(50.0, 0.5 \times N + 10 + |Q|) \leq \text{NORD} \leq 50$$

- (b) 特定のパラメータ q について、複数のマシュー関数 $ce_n(x, q)$ を求める場合は、ISW=0 としてこのサブルーチンを一度呼び出した後、X と N の値のみを変え、続けてこのサブルーチンを使用すればよい。ただし、このとき NORD の値は

$$\min(50.0, 0.5 \times N_{\max} + 10 + |Q|) \leq \text{NORD} \leq 50$$

の範囲に定めなければならない。ここで、 N_{\max} は次数 N の最大値である。

このようにすれば、初期設定処理が一度だけしか行われなため、演算回数の無駄を省くことができる。

- (c) IERR=2000 で終了した場合、計算結果の精度は保証できない。
- (d) $ce_n(x, q)$ は、(余弦関数を展開項とする)Fourier 級数で表される。このサブルーチンでは、この級数を (NORD+1) 番目の展開項までの級数和として近似している。したがって、NORD の値は $ce_n(x, q)$ の計算精度と計算効率に影響する。
- (e) N と |Q| が大きい程、計算時間が増加する傾向がある。 $N \leq 90$, $|Q| \leq 70.0$ とすることが望ましい。

(7) 使用例

(a) 問題

展開項数 21 の近似条件で $ce_7(x, 5.0)$ の値を $x=1.0, 2.0, \dots, 10.0$ について求める。

(b) 主プログラム

```

PROGRAM BIMTCE
IMPLICIT REAL(8)(A-H, 0-Z)
PARAMETER( NORD = 20 )
PARAMETER( NSIZE= 2*NORD*NORD+6*NORD+108 )
REAL(8) WORK(NSIZE)
!
N7= 7
Q = 5.0D0
X = 1.0D0
ISW=0
ISWO=ISW
CALL DIMTCE(NORD, N7, Q, X, CE, ISW, WORK, IERR)
WRITE(6,4000)
WRITE(6,4500)
WRITE(6,5000) N7, Q , NORD
WRITE(6,5300)
WRITE(6,5500)
WRITE(6,6000) X, CE, IERR, ISWO
DO 2000 I= 2, 10
  X= I
  ISWO=ISW
  CALL DIMTCE(NORD, N7, Q, X, CE, ISW, WORK, IERR)
  WRITE(6,6000) X, CE, IERR, ISWO
2000 CONTINUE
STOP
4000 FORMAT(1X , '*** DIMTCE*')
4500 FORMAT(1X , '*** INPUT *')
5000 FORMAT(1X , 'MATHIEU FUNCTION :N=', I4, ' Q=', F15.6, ' NORD=', I4)
5300 FORMAT(1X, '*** OUTPUT *')
5500 FORMAT(1X, 7X, 'X', 7X, 5X, 6X, 'CEN', 5X, 6X, ' CODE', 5X, ' ISW')
6000 FORMAT(1X, F15.6, 5X, F15.6, 5X, I6, 5X, I6)
END

```

(c) 出力結果

```
*** DIMTCE*
*** INPUT *
MATHIEU FUNCTION :N= 7 Q= 5.000000 NORD= 20
*** OUTPUT *
  X          CEN          CODE          ISW
1.000000    0.902463         0           0
2.000000   -0.128610         0           1
3.000000   -0.666293         0           1
4.000000   -0.796419         0           1
5.000000   -0.769659         0           1
6.000000   -0.217824         0           1
7.000000   -0.057805         0           1
8.000000    0.858668         0           1
9.000000    0.931020         0           1
10.000000   0.869127         0           1
```

2.14.2 DIMTSE, RIMTSE

整数次マシユー関数 $se_n(x, q)$

(1) 機能

整数次マシユー関数 $se_n(x, q)$ の値を求める.

(2) 使用法

倍精度サブルーチン:

CALL DIMTSE (NORD, N, Q, X, SE, ISW, WORK, IERR)

単精度サブルーチン:

CALL RIMTSE (NORD, N, Q, X, SE, ISW, WORK, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER(4)} \\ 64 \text{ ビット整数版では INTEGER(8)} \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NORD	I	1	入 力	フーリエ展開項の個数 -1 (注意事項 (a) 参照)
2	N	I	1	入 力	次数 n
3	Q	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	パラメータ q
4	X	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数 x
5	SE	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	関数値 $se_n(x, q)$
6	ISW	I	1	入 力	処理スイッチ ISW =0 : 初期設定後, 関数値計算を行う ISW =1 : 関数値計算のみ行う
				出 力	初期設定が行われた場合 1 を返す
7	WORK	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	内容参照	入出力	作業領域 (係数テーブル) (注意事項 (b) 参照) 大きさ: $(2 \times \text{NORD}^2 + 6 \times \text{NORD} + 108)$
8	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $2 \leq \text{NORD} \leq 50$

(b) $1 \leq N \leq 2 \times \text{NORD} + 2$

(c) ISW=0 または ISW=1 であること.

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
2000	十分な精度で計算できなくなった.	処理を打ち切る.
3000	制限条件 (a) または (b) を満足しなかった.	処理を打ち切る.

(6) 注意事項

- (a) NORD の値が小さいとき, SE は十分な精度が得られないので, NORD は大きいことが望ましい. パラメータ Q および次数 N に対する NORD の値の目安は, 次の範囲である.

$$\min(50.0, 0.5 \times N + 10 + |Q|) \leq \text{NORD} \leq 50$$

- (b) 特定のパラメータ q について, 複数のマシユ関数 $se_n(x, q)$ を求める場合は, ISW=0 としてこのサブルーチンを一度呼び出した後, X と N の値のみを変え, 続けてこのサブルーチンを使用すればよい. ただし, このとき NORD の値は

$$\min(50.0, 0.5 \times N_{\max} + 10 + |Q|) \leq \text{NORD} \leq 50$$

の範囲に定めなければならない. ここで, N_{\max} は次数 N の最大値である.

このようにすれば, 初期設定処理が一度だけしか行われなため, 演算回数の無駄を省くことができる.

- (c) IERR=2000 で終了した場合, 計算結果の精度は保証できない.
- (d) $ce_n(x, q)$ は, (正弦関数を展開項とする)Fourier 級数で表される. このサブルーチンでは, この級数を (NORD+1) 番目の展開項までの級数和として近似している. したがって, NORD の値は $se_n(x, q)$ の計算精度と計算効率に影響する.
- (e) N と $|Q|$ が大きい程, 計算時間が増加する傾向がある. $N \leq 90$, $|Q| \leq 70.0$ とすることが望ましい.

(7) 使用例

(a) 問題

展開項数 21 の近似条件で $se_7(x, 5.0)$ の値を $x=1.0, 2.0, \dots, 10.0$ について求める.

(b) 主プログラム

```

PROGRAM BIMTSE
IMPLICIT REAL(8)(A-H, O-Z)
PARAMETER( NORD = 20 )
PARAMETER( NSIZE= 2*NORD*NORD+6*NORD+108 )
REAL(8) WORK(NSIZE)
!
N7= 7
Q = 5.0D0
X = 1.0D0
ISW=0
ISWO=ISW
CALL DIMTSE(NORD, N7, Q, X, SE, ISW, WORK, IERR)
WRITE(6,4000)
WRITE(6,4500)
WRITE(6,5000) N7, Q , NORD
WRITE(6,5300)
WRITE(6,5500)
WRITE(6,6000) X, SE, IERR, ISWO
DO 2000 I= 2, 10
  X= I
  ISWO=ISW
  CALL DIMTSE(NORD, N7, Q, X, SE, ISW, WORK, IERR)
  WRITE(6,6000) X, SE, IERR, ISWO
2000 CONTINUE
STOP
4000 FORMAT(1X , '*** DIMTSE*')
4500 FORMAT(1X , '*** INPUT *')
5000 FORMAT(1X , 'MATHIEU FUNCTION :N=', I4, ' Q=', F15.6, ' NORD=', I4)
5300 FORMAT(1X, '*** OUTPUT *')
5500 FORMAT(1X, 7X, 'X', 7X, 5X, 6X, 'SEN', 5X, 6X, ' CODE', 5X, ' ISW')
6000 FORMAT(1X, F15.6, 5X, F15.6, 5X, I6, 5X, I6)
END

```

(c) 出力結果

```
*** DIMTSE*
*** INPUT *
MATHIEU FUNCTION :N= 7 Q= 5.000000 NORD= 20
*** OUTPUT *
  X          SEN          CODE      ISW
1.000000    0.370085      0        0
2.000000    0.956161      0        1
3.000000    0.815456      0        1
4.000000    0.585207      0        1
5.000000   -0.568524      0        1
6.000000   -1.022084      0        1
7.000000   -1.001135      0        1
8.000000   -0.411058      0        1
9.000000    0.447748      0        1
10.000000   0.529804       0        1
```

2.15 その他の関数

2.15.1 WIXSPS, VIXSPS

ディログ関数

(1) 機能

N 個の実数 $X_i (\geq 0, i = 1, \dots, N)$ に対するディログ関数

$$Li_2(X_i) = - \int_0^{X_i} \log |t - 1| \frac{dt}{t}$$

を求める。

(2) 使用法

倍精度サブルーチン:

CALL WIXSPS (NV,XV,YV, IERR)

単精度サブルーチン:

CALL VIXSPS (NV,XV,YV, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入力	X_i の個数 N
2	XV	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	入力	$X_i (i = 1, \dots, NV)$
3	YV	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	出力	$Li_2(X_i) (i = 1, \dots, NV)$
4	IERR	I	1	出力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(b) $XV(i) \geq 0 (i = 1, \dots, NV)$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) または (b) を満足しなかった.	処理を打ち切る.

(6) 注意事項

(a) $YV(i)$ ($i = 1, \dots, NV$) には, $Li_2(XV(i))$ が代入される.

(b) $Li_2(x)$ は, $0 \leq x < 2$ で単調増加し, $x = 2$ で最大値 $\frac{\pi^2}{4}$ をとる. $x > 2$ では単調減少して, 漸近式

$$Li_2(x) = -\frac{1}{2}(\log x)^2 + \frac{\pi^2}{3} + O(x^{-1})$$

が成り立つ.

(c) $Li_2(x)$ の正零点の近似値は, 12.59517037 である.

(7) 使用例

(a) 問題

$x_i = 0.2i$ ($i = 1, 2, \dots, 10$) に対するディログ関数の値 $Li_2(x_i)$ を求める.

(b) 入力データ

NV=10, 配列 XV

(c) 主プログラム

```

PROGRAM EIXSPS
! *** EXAMPLE OF WIXSPS ***
  IMPLICIT NONE
!
  INTEGER NV
  PARAMETER( NV = 10 )
  INTEGER IERR,I
  REAL(8) XV(NV),YV(NV)
  REAL(8) FIVE
  PARAMETER( FIVE = 5.DO )
!
  DO 100 I=1,NV
    XV(I)=DBLE(I)/FIVE
  100 CONTINUE
!
  WRITE(6,6000) NV
  DO 110 I=1,NV
    WRITE(6,6010) I,XV(I)
  110 CONTINUE
!
  CALL WIXSPS( NV, XV, YV, IERR )
!
  WRITE(6,6020) IERR
  DO 120 I=1,NV
    WRITE(6,6030) I, YV(I)
  120 CONTINUE
!
  STOP
6000 FORMAT(/,&
  1X,'*** WIXSPS ***',/,/,&
  1X,' ** INPUT **',/,/,&
  1X,' NV = ',I4,/)
6010 FORMAT(1X,'
  XV(',I2,',)=',F10.6)
6020 FORMAT(/,&
  1X,' ** OUTPUT **',/,/,&
  1X,' IERR = ',I5,/)
6030 FORMAT(1X,'
  LI2( XV(',I2,', )= ',F10.6)
  END

```

(d) 出力結果

```

*** WIXSPS ***
** INPUT **
  NV = 10
  XV( 1)= 0.200000
  XV( 2)= 0.400000
  XV( 3)= 0.600000
  XV( 4)= 0.800000
  XV( 5)= 1.000000
  XV( 6)= 1.200000
  XV( 7)= 1.400000
  XV( 8)= 1.600000
  XV( 9)= 1.800000
  XV(10)= 2.000000
** OUTPUT **
  IERR = 0
  LI2( XV( 1) )= 0.211004
  LI2( XV( 2) )= 0.449283

```

LI2(XV(3))= 0.727586
LI2(XV(4))= 1.074795
LI2(XV(5))= 1.644934
LI2(XV(6))= 2.129169
LI2(XV(7))= 2.319073
LI2(XV(8))= 2.413131
LI2(XV(9))= 2.455876
LI2(XV(10))= 2.467401

2.15.2 WIDBEY, VIDBEY

デバイ関数

(1) 機能

N 個の実数 X_i ($\geq 0, i = 1, \dots, N$) に対して, デバイ関数

$$F_D(X_i) = \frac{3}{X_i^3} \int_0^{X_i} \frac{e^{t^4}}{(e^t - 1)^2} dt$$

の値を求める.

(2) 使用法

倍精度サブルーチン:

CALL WIDBEY (NV,XV,YV, IERR)

単精度サブルーチン:

CALL VIDBEY (NV,XV,YV, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER(4)} \\ 64 \text{ ビット整数版では INTEGER(8)} \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	X_i の値の個数 N
2	XV	$\begin{cases} D \\ R \end{cases}$	NV	入 力	$X_i (i = 1, \dots, NV)$
3	YV	$\begin{cases} D \\ R \end{cases}$	NV	出 力	$F_D(X_i) (i = 1, \dots, NV)$
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(b) $XV(i) \geq 0$ ($i = 1, \dots, NV$)

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) または (b) を満足しなかった.	処理を打ち切る.

(6) 注意事項

- (a) $YV(i)$ ($i = 1, \dots, NV$) には, $F_D(XV(i))$ が代入される.
- (b) デバイ関数 $F_D(y)$ は単調減少する.
- (c) $F_D(0)$ は, $\lim_{y \rightarrow +0} F_D(y)$ を意味する.

(7) 使用例

(a) 問題

$x_i = 0.2i$ ($i = 1, 2, \dots, 10$) に対するデバイ関数の値 $F_D(x_i)$ を求める.

(b) 入力データ

NV=10, 配列 XV

(c) 主プログラム

```

PROGRAM EIDBEY
! *** EXAMPLE OF WIDBEY ***
  IMPLICIT NONE
!
  INTEGER NV
  PARAMETER( NV = 10 )
  INTEGER IERR,I
  REAL(8) XV(NV),YV(NV)
  REAL(8) FIVE
  PARAMETER( FIVE = 5.DO )
!
  DO 100 I=1,NV
    XV(I)=DBLE(I)/FIVE
  100 CONTINUE
!
  WRITE(6,6000) NV
  DO 110 I=1,NV
    WRITE(6,6010) I,XV(I)
  110 CONTINUE
!
  CALL WIDBEY( NV, XV, YV, IERR )
!
  WRITE(6,6020) IERR
  DO 120 I=1,NV
    WRITE(6,6030) I, YV(I)
  120 CONTINUE
!
  STOP
6000 FORMAT(/,&
  1X,'*** WIDBEY ***',/,/,&
  1X,'** INPUT **',/,/,&
  1X,' NV = ',I4,/,&
6010 FORMAT(1X,' XV(',I2,')=' ,F10.6)
6020 FORMAT(/,&
  1X,'** OUTPUT **',/,/,&
  1X,' IERR = ',I5,/,&
6030 FORMAT(1X,' FD( XV(',I2,') )=' ,F10.6)
  END

```

(d) 出力結果

```

*** WIDBEY ***
** INPUT **
  NV = 10
  XV( 1)= 0.200000
  XV( 2)= 0.400000
  XV( 3)= 0.600000
  XV( 4)= 0.800000
  XV( 5)= 1.000000
  XV( 6)= 1.200000
  XV( 7)= 1.400000
  XV( 8)= 1.600000
  XV( 9)= 1.800000
  XV(10)= 2.000000
** OUTPUT **
  IERR = 0
  FD( XV( 1) )= 0.998003
  FD( XV( 2) )= 0.992045
  FD( XV( 3) )= 0.982229
  FD( XV( 4) )= 0.968717
  FD( XV( 5) )= 0.951732
  FD( XV( 6) )= 0.931545
  FD( XV( 7) )= 0.908467
  FD( XV( 8) )= 0.882842
  FD( XV( 9) )= 0.855031
  FD( XV(10) )= 0.825408

```

2.15.3 WINPLG, VINPLG

球面調和関数

(1) 機能

N 個の実数 X_i ($|X_i| \leq 1; i = 1, \dots, N$) に対して正規化された n 次の球面調和関数系 (位数 $m = 0, \dots, n$)

$$P_n^{*m}(X_i) = \frac{1}{4\pi\sqrt{-1}^m} A_{n,m} \int_{-\pi}^{\pi} (X_i + \sqrt{-1}\sqrt{1-X_i^2} \cos \phi)^n \cos(m\phi) d\phi$$

($m = 0, \dots, n; i = 1, \dots, N$) を求める。ただし、正規化定数 $A_{n,m}$ は

$$A_{n,0} = \sqrt{\frac{2n+1}{\pi}}; A_{n,m} = \sqrt{\frac{2(2n+1)(n-m)!(n+m)!}{\pi(n!)^2}} \quad (m = 1, \dots, n)$$

である。

(2) 使用法

倍精度サブルーチン:

CALL WINPLG (NV,XV,N,PLG,NVL,WORK, IERR)

単精度サブルーチン:

CALL VINPLG (NV,XV,N,PLG,NVL,WORK, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	X_i の値の個数 N
2	XV	$\begin{cases} D \\ R \end{cases}$	NV	入 力	$X_i (i = 1, \dots, NV)$
3	N	I	1	入 力	次数 n
4	PLG	$\begin{cases} D \\ R \end{cases}$	内容参照	出 力	球面調和関数 $P_n^{*m}(X_i)$ ($i = 1, \dots, NV; m = 0, \dots, N$) (注意事項 (a) 参照) 大きさ: (NVL, (N + 1))
5	NVL	I	1	入 力	PLG の整合寸法
6	WORK	$\begin{cases} D \\ R \end{cases}$	内容参照	ワーク	作業領域 大きさ: ($3 \times NV + N + 1$)
7	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $1 \leq NV \leq NVL$

(b) $N \geq 1$

(c) $|XV(i)| \leq 1 \quad (i = 1, \dots, NV)$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) または (b) を満足しなかった.	処理を打ち切る.
3100	制限条件 (c) を満足しなかった.	

(6) 注意事項

(a) PLG(i, m + 1) (i = 1, ..., NV; m = 0, ..., N) には, $P_n^{*m}(XV(i))$ が代入される.

(b) n を固定して $m = 0, \dots, n$ について計算する場合は 2.12.1 $\left\{ \begin{array}{l} \text{DILEG1} \\ \text{RILEG1} \end{array} \right\}$ より効率が良い.

(c) 非負整数 m について, $n_1, n_2 \geq m$ である整数 n_1, n_2 をとるとき, 以下の積分関係式が成り立つ.

$$\int_{-1}^1 P_{n_1}^{*m}(x) P_{n_2}^{*m}(x) dx = \frac{\delta_{n_1, n_2}}{(1 + \delta_{0, m})\pi}$$

球面調和関数 $P_n^{*m}(\cos \theta) \cos(m\phi)$ ($m = 0, \dots, n$) と $P_n^{*m}(\cos \theta) \sin(m\phi)$ ($m = 1, \dots, n$) を $n = 1, 2, \dots$ についてとれば, これらは単位球面上の面積分 $\int_{-\pi \leq \phi \leq \pi; 0 \leq \theta \leq \pi} \sin \theta d\theta d\phi$ について, 正規直交基底となる.

(d) 関係式

$$\frac{2n+1}{4\pi} P_n(xy + \sqrt{(1-x^2)(1-y^2)} \cos \phi) = \sum_{m=0}^n P_n^{*m}(x) P_n^{*m}(y) \cos(m\phi) \quad (-1 \leq x, y \leq 1)$$

が成り立つ.

(7) 使用例

(a) 問題

$x_1 = 0.57735026919, x_2 = -0.57735026919$ に対する, $n = 10$ 次の球面調和関数の値

$$P_n^{*m}(x_i) \quad (m = 0, \dots, n)$$

を求める.

(b) 入力データ

NV=2, NVL= 2, 配列 XV, N= 10

(c) 主プログラム

```

PROGRAM EINPLG
! *** EXAMPLE OF WINPLG ***
IMPLICIT NONE
!
INTEGER NV, N, NVL
PARAMETER( NV = 2, N = 10, NVL = 2 )
INTEGER IERR, I
REAL(8) XV(NV), PLG(NVL, N+1), WORK(3*NV+N+1)
!
XV(1)=0.57735026919D0
XV(2)=-XV(1)
!
WRITE(6,6000) NV, N, NVL
WRITE(6,6010) XV(1), XV(2)
!
CALL WINPLG( NV, XV, N, PLG, NVL, WORK, IERR )
!
WRITE(6,6020) IERR
DO 100 I=0, N
WRITE(6,6030) N, I, PLG(1, I+1), PLG(2, I+1)
100 CONTINUE
!
STOP
6000 FORMAT(/, &
1X, '*** WINPLG ***', /, /, &
1X, ' ** INPUT **', /, /, &
1X, ' NV = ', I4, ', N = ', I4, ' NVL = ', I4, /)
6010 FORMAT(1X, ' XV= ', F10.6, ', ', F10.6, /)
6020 FORMAT(/, &

```

```

1X,' ** OUTPUT **',/,/,&
1X,' IERR =',I5,/,/,&
1X,' TABLE OF SPHERICAL HARMONICS FOR XV(1)',&
        XV(2)',/)
6030 FORMAT(1X,7X,I2,' TH DEGREE HARMONIC ORDER ',&
        I2,' = ',F10.6,3X,F10.6)
END

```

(d) 出力結果

```

*** WINPLG ***
** INPUT **
NV = 2 N = 10 NVL = 2
XV= 0.577350 -0.577350

** OUTPUT **
IERR = 0
TABLE OF SPHERICAL HARMONICS FOR XV(1) XV(2)
10 TH DEGREE HARMONIC ORDER 0 = -0.345789 -0.345789
10 TH DEGREE HARMONIC ORDER 1 = 0.076718 -0.076718
10 TH DEGREE HARMONIC ORDER 2 = 0.503967 0.503967
10 TH DEGREE HARMONIC ORDER 3 = 0.061596 -0.061596
10 TH DEGREE HARMONIC ORDER 4 = -0.492768 -0.492768
10 TH DEGREE HARMONIC ORDER 5 = -0.358105 0.358105
10 TH DEGREE HARMONIC ORDER 6 = 0.239552 0.239552
10 TH DEGREE HARMONIC ORDER 7 = 0.634916 -0.634916
10 TH DEGREE HARMONIC ORDER 8 = 0.586511 0.586511
10 TH DEGREE HARMONIC ORDER 9 = 0.319568 -0.319568
10 TH DEGREE HARMONIC ORDER 10 = 0.101056 0.101056

```

2.15.4 WIXSLA, VIXSLA ランジュバン関数

(1) 機能

$x = X_i$ に対するランジュバン関数

$$L(x) = \coth(x) - \frac{1}{x}$$

の値を求める。

(2) 使用法

倍精度サブルーチン:

CALL WIXSLA (NV, XI, XO, IERR)

単精度サブルーチン:

CALL VIXSLA (NV, XI, XO, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\left\{ \begin{array}{l} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{array} \right\}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力データ数
2	XI	$\left\{ \begin{array}{l} D \\ R \end{array} \right\}$	NV	入 力	変数値 X_i
3	XO	$\left\{ \begin{array}{l} D \\ R \end{array} \right\}$	NV	出 力	$L(X_i)$ の値
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NV \geq 1$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.

(6) 注意事項

(a) $L(x)$ を定義式どおりに計算すると $x \simeq 0$ で精度が悪くなるのでこのサブルーチンを使用されたい.

(7) 使用例

(a) 問題

$L(x)$ の値を $x = 0.0, 0.1, \dots, 0.9$ について求める.

(b) 主プログラム

```

PROGRAM EIXSLA
IMPLICIT REAL(8) (A-H,O-Z)
PARAMETER (NV=10)
REAL(8) XI(NV), XO(NV)
CHARACTER*6 CNAME, CFNC
PARAMETER( CNAME='WIXSLA', CFNC=' L' )
!
DNV=NV
DO 1000 I=1,NV
  XI(I)=(I-1)/DNV
1000 CONTINUE
!
CALL WIXSLA( NV, XI, XO, IERR )
!
WRITE(6,6000) CNAME
WRITE(6,6100)
DO 2000 I=1,NV
  WRITE(6,6200) I,XI(I)
2000 CONTINUE
!
WRITE(6,6300)
WRITE(6,6400) IERR
DO 3000 I=1,NV
  WRITE(6,6500) CFNC,XI(I), XO(I)
3000 CONTINUE
STOP
!
6000 FORMAT(1X,'*** ',A6,' *')
6100 FORMAT(1X,'*** INPUT *')
6200 FORMAT(1X,'XI(',I2,')=',F10.6 )
6300 FORMAT(1X,'*** OUTPUT *')
6400 FORMAT(1X,'IERR=',I5 )
6500 FORMAT(1X,A6,'(',F10.6,')=',F10.6 )
END

```

(c) 出力結果

```

*** WIXSLA *
*** INPUT *
XI( 1)= 0.000000
XI( 2)= 0.100000
XI( 3)= 0.200000
XI( 4)= 0.300000
XI( 5)= 0.400000
XI( 6)= 0.500000
XI( 7)= 0.600000
XI( 8)= 0.700000
XI( 9)= 0.800000
XI(10)= 0.900000
*** OUTPUT *
IERR= 0
L( 0.000000)= 0.000000
L( 0.100000)= 0.033311
L( 0.200000)= 0.066490
L( 0.300000)= 0.099405
L( 0.400000)= 0.131932
L( 0.500000)= 0.163953
L( 0.600000)= 0.195359
L( 0.700000)= 0.226050
L( 0.800000)= 0.255941
L( 0.900000)= 0.284956

```

2.15.5 WIXZTA, VIXZTA フルビッツゼータ関数

(1) 機能

$a > 0, s = X_i \geq 0$ に対して、フルビッツゼータ関数から $(s-1)^{-1}$ を引いた

$$\zeta(s, a) - (s-1)^{-1} = \frac{1}{\Gamma(s)} \left(\int_1^\infty \frac{e^{-at}}{1-e^{-t}} t^{s-1} dt + \int_0^1 \left(\frac{e^{-at}}{1-e^{-t}} - \frac{1}{t} \right) t^{s-1} dt + (s-1)^{-1} \right) - (s-1)^{-1}$$

の値を求める。この右辺は

$$\sum_{n=0}^{\infty} (n+a)^{-s} - (s-1)^{-1}$$

($\Re(s) > 1$) を $\Re(s) > 0$ にまで解析接続した積分表示の例である。

(2) 使用法

倍精度サブルーチン:

CALL WIXZTA (NV, X, A, Y, IERR)

単精度サブルーチン:

CALL VIXZTA (NV, X, A, Y, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	NV	I	1	入 力	入力値の個数
2	X	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	入 力	変数値 s
3	A	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	パラメータ a
4	Y	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NV	出 力	フルビッツゼータ関数の値 $\zeta(s, a) - 1/(s-1)$
5	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

- (a) $A > 0.0$
- (b) $X(i) \geq 0.0$
- (c) $NV > 0$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a),(b) または (c) を満足しなかった.	処理を打ち切る.

(6) 注意事項

- (a) $\zeta(s, a)$ は s を複素変数とする有理型関数に解析接続され, $s = 1$ のみが極である. このサブルーチンは, フルビッツゼータ関数の値そのものを出力するのではなく, $1/(s-1)$ を引いた値を出力している. したがって, $X(i) = 1$ のとき $-Y(i)$ はディガンマ関数 $\psi(a)$ に等しい.
- (b) ポリガンマ関数は, $X(i) = 2, 3, \dots$ としたときの $Y(i)$ の定数倍で与えることもできる.

(7) 使用例

(a) 問題

$x = 0.5i$ ($i = 1, \dots, 10$), $a = 1$ について, $\zeta(x, a) - (x-1)^{-1}$ を求める.

(b) 主プログラム

```

PROGRAM EIXZTA
REAL(8) X(10) , Y(10) , A
INTEGER NV,IERR,I
!
NV=10
DO 1000 I=1, 10
X(I) = 0.5D0*I
1000 CONTINUE
A=1.D0
CALL WIXZTA(NV, X, A, Y, IERR)
WRITE(6,5900)
WRITE(6,5950)
WRITE(6,6000) X(1), A
DO 2000 I=2,NV
WRITE(6,6050) X(I)
2000 CONTINUE
WRITE(6,6060)
WRITE(6,6100) IERR
DO 3000 I=1,NV
IF(X(I).EQ.1.D0) THEN
WRITE(6,6200) X(I), X(I), Y(I)
ELSE
WRITE(6,6300) X(I), X(I), Y(I), X(I), Y(I)+1/(X(I)-1)
ENDIF
3000 CONTINUE
STOP
!
5900 FORMAT(1X,'*** WIXZTA ***',/,/)
5950 FORMAT(1X,'*** INPUT ***',/,/)
6000 FORMAT(1X,'X= ',F10.7, ' A= ',F10.7)
6050 FORMAT(1X,'X= ',F10.7)
6060 FORMAT(1X,/,/,1X,'*** OUTPUT ***',/,/)
6100 FORMAT(1X,'OUTPUT VALUES : IERR= ',I5,/,/)
6200 FORMAT(1X,'ZETA(',F10.7,',',1) -1/(',F10.7,', -1)= ',F10.7,&
'EULER CONSTANT '
6300 FORMAT(1X,'ZETA(',F10.7,',',1) -1/(',F10.7,', -1)= ',F10.7,&
'ZETA(',F10.7,',',1)= ',F10.7)
END

```

(c) 出力結果

```

*** WIXZTA ***

*** INPUT ***

X= 0.5000000 A= 1.0000000
X= 1.0000000
X= 1.5000000
X= 2.0000000
X= 2.5000000
X= 3.0000000
X= 3.5000000
X= 4.0000000
X= 4.5000000
X= 5.0000000

*** OUTPUT ***

OUTPUT VALUES : IERR= 0

ZETA( 0.5000000,1) -1/( 0.5000000 -1)= 0.5396455 ZETA( 0.5000000,1)= -1.4603545
ZETA( 1.0000000,1) -1/( 1.0000000 -1)= 0.5772157 EULER CONSTANT
ZETA( 1.5000000,1) -1/( 1.5000000 -1)= 0.6123753 ZETA( 1.5000000,1)= 2.6123753
ZETA( 2.0000000,1) -1/( 2.0000000 -1)= 0.6449341 ZETA( 2.0000000,1)= 1.6449341
ZETA( 2.5000000,1) -1/( 2.5000000 -1)= 0.6748206 ZETA( 2.5000000,1)= 1.3414873
ZETA( 3.0000000,1) -1/( 3.0000000 -1)= 0.7020569 ZETA( 3.0000000,1)= 1.2020569
ZETA( 3.5000000,1) -1/( 3.5000000 -1)= 0.7267339 ZETA( 3.5000000,1)= 1.1267339
ZETA( 4.0000000,1) -1/( 4.0000000 -1)= 0.7489899 ZETA( 4.0000000,1)= 1.0823232
ZETA( 4.5000000,1) -1/( 4.5000000 -1)= 0.7689932 ZETA( 4.5000000,1)= 1.0547075
ZETA( 5.0000000,1) -1/( 5.0000000 -1)= 0.7869278 ZETA( 5.0000000,1)= 1.0369278

```

2.15.6 DIXEPS, RIXEPS

正定値 2 次形式 $x^2 + ay^2$ のゼータ関数

(1) 機能

正定値 2 次形式 $x^2 + ay^2$ のゼータ関数から極を消すための関数を減じた

$$f(s; a) \equiv \sum_{(m,n) \in Z^2, (m,n) \neq (0,0)} (m^2 + an^2)^{-s} - \frac{\pi^s a^{-s/2}}{\Gamma(s)(s-1)} \quad (s > 1)$$

の解析接続を, $s > -1$ に対して求める.

(2) 使用法

倍精度サブルーチン:

CALL DIXEPS (S, A, Y, IERR)

単精度サブルーチン:

CALL RIXEPS (S, A, Y, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER(4)} \\ 64 \text{ ビット整数版では INTEGER(8)} \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	S	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	変数値 s
2	A	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	正定値 2 次形式の係数 a
3	Y	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	正定値 2 次形式のゼータ関数 $f(s; a)$ の値 (注意事項 (a) 参照)
4	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $S > -1$

(b) $A > 0$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) または (b) を満足しなかった.	処理を打ち切る.

(6) 注意事項

(a) 正定値 2 次形式 $x^2 + ay^2$ のゼータ関数は $s = 1$ において 1 次の唯一の極をもつ有理型関数である. このサブルーチンはゼータ関数の値そのものを出力するのではなく, 極を消すために $\frac{\pi^s a^{-s/2}}{\Gamma(s)(s-1)}$ を引いた値を出力している.

(b) 整数対 (m, n) は, $Z^2(0$ および負も含めたすべての整数対) の $(0, 0)$ 以外の全部を動く.

(7) 使用例

(a) 問題

$s = 3.0$, $a = 1.0$ に対する正定値 2 次形式 $x^2 + ay^2$ のゼータ関数の値を求めて、フルビッツのゼータ関数値を用いて別の方法で計算した値との比較をする。

(b) 入力データ

S=3.0, A=1.0

(c) 主プログラム

```

PROGRAM BIXEPS
! *** EXAMPLE OF DIXEPS ***
IMPLICIT NONE
!
REAL(8) S,Y,A,Z,PAI3,Z1,Z2,Z3,F1,F2
PARAMETER( PAI3 = 31.006276680299820175476315067101D0/4.DO )
INTEGER IERR
!
S = 3.DO
A = 1.DO
F1 = 0.25D0
F2 = 0.75D0
!
WRITE(6,6000) S, A
!
CALL DIXEPS(S, A, Y, IERR)
!
WRITE(6,6010) IERR
WRITE(6,6020) Y
Z = Y + PAI3
CALL DIXZTA(S, A, Z1, IERR)
IF(IERR.NE.0) WRITE(6,6030)
CALL DIXZTA(S, F1, Z2, IERR)
IF(IERR.NE.0) WRITE(6,6030)
CALL DIXZTA(S, F2, Z3, IERR)
IF(IERR.NE.0) WRITE(6,6030)
Z2 = (Z2 - Z3)*F1**S
Z1 = Z1*Z2/F1
WRITE(6,6040) Z, Z1
!
STOP
6000 FORMAT(/,&
1X,'*** DIXEPS ***',/,/,&
1X,' ** INPUT **',/,/,&
1X,' S= ',F10.7,' A= ',F10.7,/)
6010 FORMAT(/,&
1X,' ** OUTPUT **',/,/,&
1X,' IERR =',I5,/)
6020 FORMAT(1X,' Y = ',F10.7,/)
6030 FORMAT(1X,' ** ERROR IN DIXZTA **',/)
6040 FORMAT(1X,' ZETA FUNCTION FOR M*M+N*N = Y + POLER = ',&
F10.7,/,&
1X,' ANOTHER OBTAINING 4*ZETA(S)*L(S,KAI4) = ',&
F10.7,/)
END
    
```

(d) 出力結果

```

*** DIXEPS ***
** INPUT **
S= 3.0000000 A= 1.0000000

** OUTPUT **
IERR = 0
Y = -3.0926556

ZETA FUNCTION FOR M*M+N*N = Y + POLER = 4.6589136
ANOTHER OBTAINING 4*ZETA(S)*L(S,KAI4) = 4.6589136
    
```


第 3 章 ソート・順位付け

3.1 概要

本章では、データのソートと順位付けおよびマージを行うサブルーチンについて説明する。本ライブラリでは、以下の機能を持つサブルーチンが用意されている。

- (1) データ列のソート
- (2) ペアデータ列のソート
- (3) データ列の順位付け
- (4) 上位 N 件の抽出
- (5) ソート済みデータ列のマージ
- (6) ソート済みペアデータ列のマージ

3.1.1 使用しているアルゴリズム

3.1.1.1 ソート

昇順にソートする場合のアルゴリズムを以下に示す。降順にソートする場合のアルゴリズムは大小が異なるだけで同様である。

(1) シェル・ソート (shell sort)

- (1) 間隔 h を設定する。
- (2) データ列より、間隔 h の部分列をすべて取り出す。
- (3) 各部分列の中が小さい順に並ぶように隣同士を比較する。
逆順なら位置を交換し、交換したデータはさらにその前のデータとの順序を確かめる。
さらに逆順ならば位置の交換が前にさかのぼって行われる。
- (4) 間隔 h を小さくして (2) から (3) を繰り返し、 $h=1$ の処理を行えばソートは終了する。

(2) ヒープ・ソート (heap sort)

- (1) 与えられたデータをヒープ・ツリー (整列二分木。親は子よりも大きいか等しい値を持っている) に構成する。
- (2) ルートとツリーの一番後ろのデータを交換する。
- (3) 一番後ろのデータを除いた部分を A とする。
- (4) A を新しいツリーと考え、これを再びヒープ・ツリーに構成する。
- (5) (2) から (4) を繰り返し、データがルートだけになればソートは終了する。

(3) クイック・ソート (quick sort)

- (1) ソート区間内のデータ数を数える。
- (2) データ数により以下のことをする。
 - データ数が 1 以下のとき：
何もしない。
 - データ数が 2 のとき：
逆順なら位置を交換する。
 - データ数が 3 以上のとき：
 - ① 区間内から枢軸値を一つ選ぶ。
 - ② 区間内のデータを枢軸値より小さいものと大きいものとの二つの区間に振り分ける。
- (3) (1) から (2) を繰り返し、すべてのデータ区間のデータ数が 2 以下になればソートは終了する。

(4) マージ・ソート (merge sort)

- (1) ソート区間内のデータ数を数える。
- (2) データ数により以下のことをする。
 - データ数が 1 のとき：
何もしない。
 - データ数が 2 のとき：
逆順なら位置を交換する。
 - データ数が 3 以上のとき：
 - ① 区間のデータを半分ずつ前半と後半に分ける。
 - ② 前半のデータを再帰的にマージ・ソートする。

後半のデータを再帰的にマージ・ソートする。

③ ソートされた前半と後半のデータをマージする。

3.1.1.2 データ列の順位付け

n 個のデータが与えられた場合の各データに対応する昇順順位番号と同一順位にあるデータの数を返す。

3.1.1.3 上位 N 件の抽出

n 個のデータ $a_i (i = 1, 2, \dots, n)$ が与えられたときこれを大きい順または小さい順に並べ換えたデータ列のうち先頭から m 個のデータ $a_j (j = j_1, j_2, \dots, j_m) (m < n)$ を求める。

3.1.1.4 ソート済みデータ列のマージ

昇順に整列された 2 つのデータ列 $a_i (i = 1, 2, \dots, n)$ と $b_j (j = 1, 2, \dots, m)$ をマージして、データ列 $c_k (k = 1, 2, \dots, \ell)$ を求める。

ただし、 c_k は

$$c_1 \leq c_2 \leq \dots \leq c_\ell$$

を満たす。

3.1.1.5 ソート済みペアデータ列のマージ

a_i について昇順に整列されたデータの組 $(a_i, b_i) (i = 1, 2, \dots, n)$ と c_j について昇順に整列されたデータの組 $(c_j, d_j) (j = 1, 2, \dots, m)$ をマージして、データの組 $(e_k, f_k) (k = 1, 2, \dots, \ell)$ を求める。

ただし、 e_k は

$$e_1 \leq e_2 \leq \dots \leq e_\ell$$

を満たす。

なお、2 次ソートを指定した場合には、 $e_k = e_{k+1}$ を満たす任意の k について

$$f_k \leq f_{k+1}$$

を満たすように $k = 1, 2, \dots, \ell$ を決める。

3.1.2 参考文献

- (1) Niklaus Wirth, “ALGORITHMS + DATA STRUCTURES = PROGRAMS”, Prentice-Hall Inc. (1976).
- (2) 浪平博人, “ソート・検索”, CQ 出版.
- (3) 近藤嘉雪, “アルゴリズムとデータ構造”, SOFTBANK.

3.2 ソート

3.2.1 DSSTA1, RSSTA1

データ列のソート

(1) 機能

n 個のデータ $a_{ik} (k = 1, 2, \dots, n)$ が与えられたとき a_i を並べ換えたデータ列 $a_{jk} (k = 1, 2, \dots, n)$ を求める。
ただし、 a_j は

昇順の場合： $a_{j_1} \leq a_{j_2} \leq \dots \leq a_{j_n}$

降順の場合： $a_{j_1} \geq a_{j_2} \geq \dots \geq a_{j_n}$

を満たす。

(2) 使用法

倍精度サブルーチン:

CALL DSSTA1 (A,N,ISW,WK,IWK, IERR)

単精度サブルーチン:

CALL RSSTA1 (A,N,ISW,WK,IWK, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER(4)} \\ 64 \text{ ビット整数版では INTEGER(8)} \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	A	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	N	入力	ソートされるデータ a_i
				出力	ソートされたデータ a_j
2	N	I	1	入力	配列 A の大きさ
3	ISW	I	1	入力	ソート法の選択スイッチ (注意事項 (a) 参照)
4	WK	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	内容参照	ワーク	作業領域 大きさ： N (ISW= 4, -4 のとき) 1 (それ以外の場合)
5	IWK	I	内容参照	ワーク	作業領域 大きさ： $2 \times N$ (ISW= 3, -3 のとき) 1 (それ以外の場合)
6	IERR	I	1	出力	エラーインディケータ

(4) 制限条件

(a) $N \geq 1$

(b) ISW=1, 2, 3, 4, -1, -2, -3, -4

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
3100	制限条件 (b) を満足しなかった.	

(6) 注意事項

(a) ISW で選択するソート法は以下の通りである.

ISW	ソート法	ISW	ソート法
1	シェル・ソート (昇順)	-1	シェル・ソート (降順)
2	ヒープ・ソート (昇順)	-2	ヒープ・ソート (降順)
3	クイック・ソート (昇順)	-3	クイック・ソート (降順)
4	マージ・ソート (昇順)	-4	マージ・ソート (降順)

利用者は入力データの性質により適切なソート法を選ばよ。各ソート法の特徴を以下に示す。

・シェル・ソート

計算量は平均 $O(n^{1.5})$ 程度である。どのようなデータに対しても安定して速いソートを行う。とくに、データ系列の一部がソートされている場合は速くなる。

データに複数個同じ値が存在する場合、データの順序がソートを行う前後で保たれる保証はない。

ワーク領域が不要である。

・ヒープ・ソート

計算量は $O(n \log n)$ だが、定数項部分は大きめである。入力データの性質によりソート時間があまり変わらない。

データに複数個同じ値が存在する場合、データの順序がソートを行う前後で保たれる保証はない。

ワーク領域が不要である。

・クイック・ソート

計算量は平均 $O(n \log n)$ だが、最初から部分的にソートされているなど、ある種の規則性があるものに対しては、大変非効率的なソートになる。ランダムなデータに対してはもっとも速いソート方法である。

データに複数個同じ値が存在する場合、データの順序がソートを行う前後で保たれる保証はない。

・マージ・ソート

計算量は $O(n \log n)$ だが、定数項部分は大きめである。入力データの性質によりソート時間があまり変わらない。

同じ値のデータ間で整列前の順序関係が保たれる。

(7) 使用例

(a) 問題

A (1) = 5.0
A (2) = 4.0
A (3) = 9.0
A (4) = 6.0
A (5) = 2.0
A (6) = 5.0

をシェル・ソートで昇順にソートする.

(b) 入力データ

配列 A, N=6, ISW=1

(c) 主プログラム

```

PROGRAM BSSTA1
! *** EXAMPLE OF DSSTA1 ***
IMPLICIT NONE
!
INTEGER NA
PARAMETER( NA = 100 )
INTEGER N, ISW, IWK(2*NA), IERR, I
REAL(8) A(NA), WK(NA)
!
!
! DATA SET
DATA (A(I), I=1,6) /5.0D0,4.0D0,9.0D0,6.0D0,2.0D0,5.0D0/
N = 6
ISW = 1
!
! WRITE INPUT DATA
WRITE(6,6000) ISW,N
DO 110 I=1,N
WRITE(6,6010) I,A(I)
110 CONTINUE
!
! SORT
CALL DSSTA1(A,N,ISW,WK,IWK,IERR)
!
! WRITE OUTPUT DATA
WRITE(6,6020) IERR
IF( IERR .LT. 3000 ) THEN
DO 120 I=1,N
WRITE(6,6010) I,A(I)
120 CONTINUE
ENDIF
!
STOP
6000 FORMAT(/,&
1X,'*** DSSTA1 ***',/,/,&
1X,'** INPUT **',/,/,&
1X,' ISW =',I6,/,&
1X,' N =',I6,/,&
6010 FORMAT(1X,' A(',I2,',)=',F5.1)
6020 FORMAT(/,&
1X,'** OUTPUT **',/,/,&
1X,' IERR =',I5,/)
END

```

(d) 出力結果

```

*** DSSTA1 ***
** INPUT **
ISW = 1
N = 6
A( 1)= 5.0
A( 2)= 4.0
A( 3)= 9.0
A( 4)= 6.0
A( 5)= 2.0
A( 6)= 5.0
** OUTPUT **
IERR = 0
A( 1)= 2.0
A( 2)= 4.0
A( 3)= 5.0
A( 4)= 5.0
A( 5)= 6.0
A( 6)= 9.0

```

3.2.2 DSSTA2, RSSTA2 ペアデータ列のソート

(1) 機能

2組の n 個のデータ $a_{i_k} (k = 1, 2, \dots, n), b_{i_k} (k = 1, 2, \dots, n)$ が与えられたとき a_i を並べ換えたデータ列 $a_{j_k} (k = 1, 2, \dots, n)$ と a_j に対応するデータ列 $b_{j_k} (k = 1, 2, \dots, n)$ を求める。ただし, a_j は

昇順の場合: $a_{j_1} \leq a_{j_2} \leq \dots \leq a_{j_n}$

降順の場合: $a_{j_1} \geq a_{j_2} \geq \dots \geq a_{j_n}$

を満たす。

なお, 2次ソートを指定した場合には, $a_{j_k} = a_{j_{k+1}}$ を満たす任意の k について

昇順の場合: $b_{j_k} \leq b_{j_{k+1}}$

降順の場合: $b_{j_k} \geq b_{j_{k+1}}$

を満たすように $j = j_1, j_2, \dots, j_n$ を決める。

(2) 使用法

倍精度サブルーチン:

CALL DSSTA2 (A,N,B,ISW1,ISW2,WK,IWK, IERR)

単精度サブルーチン:

CALL RSSTA2 (A,N,B,ISW1,ISW2,WK,IWK, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	A	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	N	入 力	ソートされるデータ a_i
				出 力	ソートされたデータ a_j
2	N	I	1	入 力	配列 A の大きさ
3	B	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	N	入 力	a_i に対するデータ b_i
				出 力	ソートされた a_j に対するデータ b_j
4	ISW1	I	1	入 力	ソート法の選択スイッチ (注意事項 (a) 参照)
5	ISW2	I	1	入 力	2次ソートスイッチ ISW2=0 : 2次ソートは行わない ISW2=1 : 2次ソートを行う
6	WK	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	内容参照	ワーク	作業領域 大きさ : $2 \times N$ (ISW1= 4, -4 のとき) 1 (それ以外のとき)
7	IWK	I	内容参照	ワーク	作業領域 大きさ : $2 \times N$ (ISW1= 3, -3 のとき) 1 (それ以外のとき)
8	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

- (a) $N \geq 1$
- (b) ISW1=1, 2, 3, 4, -1, -2, -3, -4
- (c) ISW2=0 または 1

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
3100	制限条件 (b) を満足しなかった.	
3200	制限条件 (c) を満足しなかった.	

(6) 注意事項

(a) ISW1 で選択するソート法は以下の通りである.

ISW1	ソート法	ISW1	ソート法
1	シェル・ソート (昇順)	-1	シェル・ソート (降順)
2	ヒープ・ソート (昇順)	-2	ヒープ・ソート (降順)
3	クイック・ソート (昇順)	-3	クイック・ソート (降順)
4	マージ・ソート (昇順)	-4	マージ・ソート (降順)

利用者は入力データの性質により適切なソート法を選ばばよい. 各ソート法の特徴を以下に示す.

・シェル・ソート

計算量は平均 $O(n^{1.5})$ 程度である. どのようなデータに対しても安定して速いソートを行う. とくに, データ系列の一部がソートされている場合は速くなる.

第 1 の組のデータに複数個同じ値が存在する場合, それに対応する第 2 の組のデータの順序がソートを行う前後で保たれる保証はない.

ワーク領域が不要である.

・ヒープ・ソート

計算量は $O(n \log n)$ だが, 定数項部分は大きめである. 入力データの性質によりソート時間があまり変わらない.

第 1 の組のデータに複数個同じ値が存在する場合, それに対応する第 2 の組のデータの順序がソートを行う前後で保たれる保証はない.

ワーク領域が不要である.

・クイック・ソート

計算量は平均 $O(n \log n)$ だが, 最初から部分的にソートされているなど, ある種の規則性があるものに対しては, 大変非効率的なソートになる. ランダムなデータに対してはもっとも速いソート方法である.

第 1 の組のデータに複数個同じ値が存在する場合, それに対応する第 2 の組のデータの順序がソートを行う前後で保たれる保証はない.

・マージ・ソート

計算量は $O(n \log n)$ だが, 定数項部分は大きめである. 入力データの性質によりソート時間があまり変わらない.

同じ値のデータ間で整列前の順序関係が保たれる.

(7) 使用例

(a) 問題

A (1) = 5.0, B (1) = 3.0

A (2) = 4.0, B (2) = 4.0

A (3) = 9.0, B (3) = 2.0

A (4) = 6.0, B (4) = 3.0

A (5) = 2.0, B (5) = 8.0

A (6) = 5.0, B (6) = 1.0

の A をシェル・ソートで昇順にソートし, それに対応するように B を並べかえる. 2 次ソートも行う.

(b) 入力データ

配列 A, 配列 B, N=6, ISW1=1, ISW2=1

(c) 主プログラム

```
PROGRAM BSSTA2
! *** EXAMPLE OF DSSTA2 ***
IMPLICIT NONE
!
```



```

    INTEGER NA
    PARAMETER( NA = 100 )
    INTEGER N, ISW1, ISW2, IWK(2*NA), IERR, I
    REAL(8) A(NA), B(NA), WK(2*NA)
!
!   DATA SET
    DATA (A(I), I=1,6) /5.0D0,4.0D0,9.0D0,6.0D0,2.0D0,5.0D0/
    DATA (B(I), I=1,6) /3.0D0,4.0D0,2.0D0,3.0D0,8.0D0,1.0D0/
    N = 6
    ISW1 = 1
    ISW2 = 1
!
!   WRITE INPUT DATA
    WRITE(6,6000) ISW1, ISW2, N
    DO 110 I=1, N
        WRITE(6,6010) I, A(I), I, B(I)
110 CONTINUE
!
!   SORT
    CALL DSSTA2(A, N, B, ISW1, ISW2, WK, IWK, IERR)
!
!   WRITE OUTPUT DATA
    WRITE(6,6020) IERR
    IF( IERR .LT. 3000 ) THEN
        DO 120 I=1, N
            WRITE(6,6010) I, A(I), I, B(I)
120 CONTINUE
        ENDIF
!
    STOP
6000 FORMAT(/, &
           1X, '*** DSSTA2 ***', /, /, &
           1X, ' ** INPUT **', /, /, &
           1X, ' ISW1 =', I6, /, &
           1X, ' ISW2 =', I6, /, &
           1X, ' N =', I6, /)
6010 FORMAT(1X, ' A(', I2, ')=' , F5.1, 7X, ' B(', I2, ')=' , F5.1)
6020 FORMAT(/, &
           1X, ' ** OUTPUT **', /, /, &
           1X, ' IERR = ', I5, /)
    END

```

(d) 出力結果

```

*** DSSTA2 ***
** INPUT **
ISW1 = 1
ISW2 = 1
N = 6

A( 1)= 5.0      B( 1)= 3.0
A( 2)= 4.0      B( 2)= 4.0
A( 3)= 9.0      B( 3)= 2.0
A( 4)= 6.0      B( 4)= 3.0
A( 5)= 2.0      B( 5)= 8.0
A( 6)= 5.0      B( 6)= 1.0

** OUTPUT **
IERR = 0

A( 1)= 2.0      B( 1)= 8.0
A( 2)= 4.0      B( 2)= 4.0
A( 3)= 5.0      B( 3)= 1.0
A( 4)= 5.0      B( 4)= 3.0
A( 5)= 6.0      B( 5)= 3.0
A( 6)= 9.0      B( 6)= 2.0

```

3.3 順位付け

3.3.1 DSSTRA, RSSTRA

データ列の順位付け

(1) 機能

n 個のデータが与えられた場合の各データに対応する昇順順位番号と同一順位にあるデータの数を返す (注意事項 (a) 参照).

n 個のデータ $a_i (i = 1, 2, \dots, n)$ が与えられ, これを昇順に並べ換えたデータ列が $a_j (j = j_1, j_2, \dots, j_{m_1+\dots+m_k})$:

$$\begin{aligned} a_{j_1} &= a_{j_2} \cdots = a_{j_{m_1}} \leq \\ a_{j_{m_1+1}} &= a_{j_{m_1+2}} \cdots = a_{j_{m_1+m_2}} \leq \\ &\cdots \leq \\ a_{j_{m_1+\dots+m_{k-1}+1}} &= a_{j_{m_1+\dots+m_{k-1}+2}} \cdots = a_{j_{m_1+\dots+m_k}} \end{aligned}$$

($m_1+\dots+m_k = n$) で与えられるとき $r_{j_{m_1+\dots+m_{\ell-1}+1}} = r_{j_{m_1+\dots+m_{\ell-1}+2}} = \cdots = r_{j_{m_1+\dots+m_{\ell}}} = \ell$ で定義される順位データ $r_j (j = 1, 2, \dots, n)$ を求める. ここで, m_{ℓ} は ℓ 番目に小さいデータに対する同一順位の数である. なお, 同一順位の数を求める場合には $c_j (j = 1, 2, \dots, n)$ に $c_{j_{m_1+\dots+m_{\ell-1}+1}} = c_{j_{m_1+\dots+m_{\ell-1}+2}} = \cdots = c_{j_{m_1+\dots+m_{\ell}}} = m_{\ell}$ を満たす様にデータを格納する.

(2) 使用法

倍精度サブルーチン:

CALL DSSTRA (A, N, IR, IC, ISW, IW, IERR)

単精度サブルーチン:

CALL RSSTRA (A, N, IR, IC, ISW, IW, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	A	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	N	入力	順位付けられるデータ a_i
2	N	I	1	入力	配列 A の大きさ
3	IR	I	N	出力	A に対応づけられて, 与えられる順位 r_j
4	IC	I	N	出力	ISW=0 のとき 同順位のデータの個数 c_j ISW=1 のとき 使用しない (注意事項 (b) 参照)
5	ISW	I	1	入力	同順位データ数出力スイッチ ISW=0: IC に同順位のデータの個数を出力する. ISW=1: 同順位のデータの個数は出力しない.
6	IW	I	N	ワーク	作業領域
7	IERR	I	1	出力	エラーインディケータ

(4) 制限条件

- (a) $N \geq 2$
- (b) ISW=0 または ISW=1

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
3010	制限条件 (b) を満足しなかった.	

(6) 注意事項

- (a) A(i) は全体のうちで IR(i) 番目に小さいデータであり, ISW=0 のとき, IR(i) 番目に小さいデータは IC(i) 個ある.
- (b) ISW=1 のとき, IC は使用されないのので, ダミー配列を引数に設定することができる.

(7) 使用例

(a) 問題

A (1) = 1.2
A (2) = 3.2
A (3) = 4.2
A (4) = 5.2
A (5) = 7.2
A (6) = 1.2
A (7) = 9.2
A (8) = 1.2
A (9) = 1.2
A (10) = 7.2
A (11) = 6.2
A (12) = 8.2
A (13) = 7.2
A (14) = 5.2
A (15) = 0.2
A (16) = 2.2
に順位をつける.

(b) 入力データ

配列 A, N=16, ISW=0

(c) 主プログラム

```

PROGRAM BSSTRA
! *** EXAMPLE OF DSSTRA ***
IMPLICIT REAL(8) (A-H,O-Z)
PARAMETER(N=16, ISW=0)
DIMENSION A(N), IR(N), IC(N), IW(N)
!
READ(5,*) (A(I), I=1,N)
WRITE(6,6000) N, ISW
CALL DSSTRA(A, N, IR, IC, ISW, IW, IERR)
WRITE(6,6010) IERR
WRITE(6,6020)
DO 100 I=1,N

```

```

      WRITE(6,6030) I,A(I),IR(I),IC(I)
100 CONTINUE
STOP
!
6000 FORMAT(' *** DSSTRA ***',/,/, ' ** INPUT **',/,/,7X,'N = ',I4,/,7X,&
' ISW = ',I4,/)
6010 FORMAT(' ** OUTPUT **',/,/,7X,' IERR = ',I4,/)
6020 FORMAT(14X,'A',7X,'IR',6X,'IC')
6030 FORMAT(6X,I3,2X,F5.1,3X,I5,3X,I5)
!
      END

```

(d) 出力結果

```

*** DSSTRA ***

** INPUT **
      N = 16
      ISW = 0

** OUTPUT **
      IERR = 0

      A      IR      IC
1      1.2    2      4
2      3.2    7      1
3      4.2    8      1
4      5.2    9      2
5      7.2   12      3
6      1.2    2      4
7      9.2   16      1
8      1.2    2      4
9      1.2    2      4
10     7.2   12      3
11     6.2   11      1
12     8.2   15      1
13     7.2   12      3
14     5.2    9      2
15     0.2    1      1
16     2.2    6      1

```

3.3.2 DSSTPT, RSSTPT

上位 N 件の抽出

(1) 機能

n 個のデータ $a_i (i = 1, 2, \dots, n)$ が与えられたときこれを大きい順または小さい順に並べ換えたデータ列のうち先頭から m 個のデータ $a_j (j = j_1, j_2, \dots, j_m) (m < n)$ を求める。

(2) 使用法

倍精度サブルーチン:

CALL DSSTPT (A, N, M, P, ISW, IERR)

単精度サブルーチン:

CALL RSSTPT (A, N, M, P, ISW, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	A	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	N	入 力	入力データ a_i
				出 力	大きい順または小さい順に整列されたデータ a_j
2	N	I	1	入 力	配列 A の大きさ
3	M	I	1	入 力	大きい順または小さい順に整列するデータの数 (注意事項 (a) 参照)
				出 力	実際に大きい順または小さい順に整列されたデータの数
4	P	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	閾値の初期値を求めるためのパラメータ (注意事項 (b) 参照)
				出 力	閾値の更新回数 (注意事項 (b) 参照)
5	ISW	I	1	入 力	ISW=0:小さい順に整列する ISW=1:大きい順に整列する (注意事項 (a) 参照)
6	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $ISW \in \{0, 1\}$

(b) $M \leq 0, N \leq 0, N < M$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) または (b) を満足しなかった.	処理を打ち切る.

(6) 注意事項

- (a) 配列 A のはじめの M 個の要素に、配列 A の要素を大きい順 (ISW=1 の場合)、または小さい順 (ISW=0 の場合) に整列した場合の結果が求まる。なお、M には、整列したい要素数を入力し、実際に整列された要素数が出力される。ここで、(整列したい要素数) \leq (実際に整列された要素数) である。
- (b) このサブルーチンでは、ある閾値をもとに入力データを逐次この閾値よりも大きい集合と小さい集合とに分割しながら処理を行う。このようにして得られた集合の大きさが整列したい要素数に近付いたときその集合について並べ替えを行う。最初の閾値は、パラメータ P に与えたデータをもとに次のように計算する。

$$\text{閾値の初期値} = \text{MAX} \times P + \text{MIN} \times (1.0 - P)$$

ここで MAX は配列 A に含まれるデータの最大値、MIN は配列 A に含まれるデータの最小値をそれぞれ表す。したがって閾値の初期値は MAX と MIN とを $(1.0 - P) : P$ に内分する点として定義される。並べ替えたいデータの性質が分かっている場合には P すなわち閾値の初期値に適切なデータを与えることによって処理スピードをあげることができる。例えば n 個の $(0, 1)$ 一様乱数が与えられてそのうち小さい方から m 個のデータを取り出したい場合、最適な閾値の推定値は $\frac{m}{n}$ であり、したがって

$$P = \frac{m}{n}$$

と指定すればよい。なお、P の出力には実際に閾値の更新を行った回数を $\left\{ \begin{array}{l} \text{倍精度} \\ \text{単精度} \end{array} \right\}$ 実数として出力する。この値が小さい程閾値の初期値が適切であったことを示している。

(7) 使用例

(a) 問題

A (1) = 5.0 A (2) = 39.0 A (3) = 15.0 A (4) = 8.0 A (5) = 23.0
 A (6) = 45.0 A (7) = 61.0 A (8) = 25.0 A (9) = 33.0 A (10) = 45.0
 A (11) = 39.0 A (12) = 10.0 A (13) = 21.0 A (14) = 5.0 A (15) = 23.0
 A (16) = 38.0 A (17) = 41.0 A (18) = 55.0 A (19) = 61.0 A (20) = 39.0

を昇順にソートした場合の小さなものから 5 番目までを求める。

(b) 入力データ

配列 A, N=20, M=5, P=0.3, ISW=0

(c) 主プログラム

```

PROGRAM BSSTPT
! *** EXAMPLE OF DSSTPT ***
IMPLICIT REAL(8) (A-H,O-Z)
REAL(8) A(100)
INTEGER I,N,IERR,M
INTEGER ISW
ISW=0
!
READ(*,5000)N,M,P,ISW
WRITE(6,6000) N,M,ISW,P
READ(*,5010) (A(I),I=1,20)
DO 100 I=1,N
WRITE(6,6010)I,A(I)
100 CONTINUE
!
CALL DSSTPT(A,N,M,P,ISW,IERR)
!
WRITE(6,6020)
WRITE(6,6030) IERR
WRITE(6,6040) M
DO 110 I=1,M
WRITE(6,6010)I,A(I)
110 CONTINUE
!
5000 FORMAT(I2,I2,F4.1,I2)
5010 FORMAT(20F5.1)
6000 FORMAT(' ',/,5X,'*** DSSTPT ***',/,&
' ',/,6X,'** INPUT **',/,&
9X,'N=',I2,' M=',I2,' ISW=',I2,' P=',F5.1,/)
6010 FORMAT(9X,'A(',I2,')',F5.1)
6020 FORMAT(' ',/,6X,'** OUTPUT **',/)

```

```
6030 FORMAT(9X,'IERR = ',I4)
6040 FORMAT(9X,'M = ',I4,/)
      END
```

(d) 出力結果

```
*** DSSTPT ***
** INPUT **
  N=20  M= 5  ISW= 0  P= 0.3
  A( 1)  5.0
  A( 2) 39.0
  A( 3) 15.0
  A( 4)  8.0
  A( 5) 23.0
  A( 6) 45.0
  A( 7) 61.0
  A( 8) 25.0
  A( 9) 33.0
  A(10) 45.0
  A(11) 39.0
  A(12) 10.0
  A(13) 21.0
  A(14)  5.0
  A(15) 23.0
  A(16) 38.0
  A(17) 41.0
  A(18) 55.0
  A(19) 61.0
  A(20) 39.0
** OUTPUT **
  IERR =  0
  M    =  6
  A( 1)  5.0
  A( 2)  5.0
  A( 3)  8.0
  A( 4) 10.0
  A( 5) 15.0
  A( 6) 21.0
```

3.4 マージ

3.4.1 DSMGON, RSMGON

ソート済みデータ列のマージ

(1) 機能

昇順に整列された 2 つのデータ列 a_i ($i = 1, 2, \dots, n$) と b_j ($j = 1, 2, \dots, m$) をマージして、データ列 c_k ($k = 1, 2, \dots, \ell$) を求める。

ただし、 c_k は

$$c_1 \leq c_2 \leq \dots \leq c_\ell$$

を満たす。

(2) 使用法

倍精度サブルーチン:

CALL DSMGON (A, NN, B, NM, C, NL, IERR)

単精度サブルーチン:

CALL RSMGON (A, NN, B, NM, C, NL, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	A	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NN	入力	マージされるデータ a_i
2	NN	I	1	入力	配列 A の大きさ
3	B	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NM	入力	マージされるデータ b_j
4	NM	I	1	入力	配列 B の大きさ
5	C	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NL	出力	マージされたデータ c_k
6	NL	I	1	入力	配列 C の大きさ
7	IERR	I	1	出力	エラーインディケータ

(4) 制限条件

(a) $NN \geq 1$

(b) $NM \geq 1$

(c) $1 \leq NL \leq NN + NM$

(d) $A(1) \leq A(2) \leq \dots \leq A(NN)$

(e) $B(1) \leq B(2) \leq \dots \leq B(NM)$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1000	制限条件 (c) を満足しなかった.	NL = NN + NM として処理する.
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
3100	制限条件 (b) を満足しなかった.	
3200	制限条件 (d) を満足しなかった.	
3300	制限条件 (e) を満足しなかった.	

(6) 注意事項

(a) $NL < NN + NM$ のとき, 小さいものから NL 個のみマージした結果が出力される.

(7) 使用例

(a) 問題

配列 X に格納された n 個のデータからなる数列 a_i ($i = 1, 2, \dots, n$) を n_s 個ずつの部分列に分割して, 部分列間のマージを繰り返すことにより元の数列全体を昇順にソートする. なお, プログラム内の配列のかわりに外部記憶装置上のファイルを用いれば, この問題は n 個のデータ a_i を外部ソートする問題となる.

(b) 入力データ

配列 X , 数列の長さ n

(c) 主プログラム

```

PROGRAM BSMGON
! *** EXAMPLE OF DSMGON ***
IMPLICIT NONE
!
INTEGER NA
PARAMETER( NA = 100 )
INTEGER N, IERR
INTEGER ILOOP, ISIZE, ISIZE2, IA, IB, IC, ICREST
INTEGER ISTA, ISTB, ISTC, ISIZEA, ISIZEB, ISIZEC
INTEGER I, J, K
REAL(8) X(NA), A(NA), B(NA), C(NA)
!
! DATA SET
N = 17
DO 100 I=1,N
  X(I) = DBLE( INT(SIN(DBLE(I))*100) )
100 CONTINUE
!
! WRITE INPUT DATA
WRITE(6,6000) N
DO 110 I=1,N
  WRITE(6,6010) I,X(I)
110 CONTINUE
!
! EXTERNAL SORT
DO 120 I=1,N
  C(I) = X(I)
120 CONTINUE
IF( N .EQ. 1 )THEN
  GOTO 130
ENDIF
!
ILOOP = 0
DO 140 I=1,N
  ILOOP = ILOOP + 1
  IF( 2**ILOOP .GE. N ) THEN
    GOTO 150
  ENDIF
140 CONTINUE
150 CONTINUE
DO 160 I=1,ILOOP
  ISIZE = 2 ** (I-1)
  ISIZE2 = 2 ** I
!
  IA = 0
  IB = 0
  IC = 0
  DO 170 J=1,N/ISIZE2
    DO 180 K=1,ISIZE
      IA = IA + 1
      A(IA) = C(IC+K)
    
```

```

180     CONTINUE
        IC = IC + ISIZE
        DO 190 K=1,ISIZE
            IB = IB + 1
            B(IB) = C(IC+K)
190     CONTINUE
        IC = IC + ISIZE
170     CONTINUE
        ICREST = N - IC
        IF( (0 .LT. ICREST) .AND. (ICREST .LE. ISIZE) ) THEN
            DO 200 K=1,ICREST
                IA = IA + 1
                A(IA) = C(IC+K)
200     CONTINUE
        ENDIF
        IF( (ISIZE .LT. ICREST) .AND. (ICREST .LT. (ISIZE*2)) ) THEN
            DO 210 K=1,ISIZE
                IA = IA + 1
                A(IA) = C(IC+K)
210     CONTINUE
            IC = IC + ISIZE
            DO 220 K=1,(ICREST-ISIZE)
                IB = IB + 1
                B(IB) = C(IC+K)
220     CONTINUE
        ENDIF
!
        DO 230 J=1,N/ISIZE2
            ISTA = (J-1) * ISIZE + 1
            ISTB = (J-1) * ISIZE + 1
            ISTD = (J-1) * ISIZE2 + 1
            ISIZEA = ISIZE
            ISIZEB = ISIZE
            ISIZEC = ISIZEA + ISIZEB
            CALL DSMGON&
                (A(ISTA),ISIZEA,B(ISTB),ISIZEB,C(ISTD),ISIZEC,IERR)
230     CONTINUE
            IF( (0 .LT. ICREST) .AND. (ICREST .LE. ISIZE) ) THEN
                ISTA = N/ISIZE2 * ISIZE
                ISTD = N/ISIZE2 * ISIZE2
                ISIZEA = ICREST
                DO 240 K=1,ISIZEA
                    C(ISTD+K) = A(ISTA+K)
240     CONTINUE
            ENDIF
            IF( (ISIZE .LT. ICREST) .AND. (ICREST .LT. (ISIZE*2)) ) THEN
                ISTA = N/ISIZE2 * ISIZE + 1
                ISTB = N/ISIZE2 * ISIZE + 1
                ISTD = N/ISIZE2 * ISIZE2 + 1
                ISIZEA = ISIZE
                ISIZEB = ICREST - ISIZE
                ISIZEC = ISIZEA + ISIZEB
                CALL DSMGON&
                    (A(ISTA),ISIZEA,B(ISTB),ISIZEB,C(ISTD),ISIZEC,IERR)
            ENDIF
160 CONTINUE
130 CONTINUE
!
!     WRITE OUTPUT DATA
        WRITE(6,6020) IERR
        IF( IERR .LT. 3000 ) THEN
            DO 250 I=1,N
                WRITE(6,6030) I,C(I)
250     CONTINUE
        ENDIF
!
        STOP
6000 FORMAT(/,&
            1X,'*** DSMGON ***',/,&
            1X,' ** INPUT **',/,&
            1X,' N =',I6,/),&
6010 FORMAT(1X,' X(',I2,')=',F5.1)
6020 FORMAT(/,&
            1X,' ** OUTPUT **',/,&
            1X,' IERR =',I5,/),&
6030 FORMAT(1X,' C(',I2,')=',F5.1)
        END

```

(d) 出力結果

```

*** DSMGON ***
** INPUT **
N = 17
X( 1)= 84.0
X( 2)= 90.0
X( 3)= 14.0
X( 4)=-75.0
X( 5)=-95.0
X( 6)=-27.0
X( 7)= 65.0
X( 8)= 98.0
X( 9)= 41.0
X(10)=-54.0
X(11)=-99.0
X(12)=-53.0
X(13)= 42.0
X(14)= 99.0

```

X(15)= 65.0
X(16)=-28.0
X(17)=-96.0

** OUTPUT **

IERR = 0

C(1)=-99.0
C(2)=-96.0
C(3)=-95.0
C(4)=-75.0
C(5)=-54.0
C(6)=-53.0
C(7)=-28.0
C(8)=-27.0
C(9)= 14.0
C(10)= 41.0
C(11)= 42.0
C(12)= 65.0
C(13)= 65.0
C(14)= 84.0
C(15)= 90.0
C(16)= 98.0
C(17)= 99.0

3.4.2 DSMGPA, RSMGPA

ソート済みペアデータ列のマージ

(1) 機能

a_i について昇順に整列されたデータの組 (a_i, b_i) ($i = 1, 2, \dots, n$) と c_j について昇順に整列されたデータの組 (c_j, d_j) ($j = 1, 2, \dots, m$) をマージして、データの組 (e_k, f_k) ($k = 1, 2, \dots, \ell$) を求める。

ただし、 e_k は

$$e_1 \leq e_2 \leq \dots \leq e_\ell$$

を満たす。なお、2次ソートを指定した場合には、 $e_k = e_{k+1}$ を満たす任意の k について

$$f_k \leq f_{k+1}$$

を満たすように $k = 1, 2, \dots, \ell$ を決める。

(2) 使用法

倍精度サブルーチン:

CALL DSMGPA (A, NN, B, C, NM, D, E, NL, F, ISW, IERR)

単精度サブルーチン:

CALL RSMGPA (A, NN, B, C, NM, D, E, NL, F, ISW, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	A	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NN	入 力	マージされるデータ a_i
2	NN	I	1	入 力	配列 A の大きさ
3	B	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NN	入 力	a_i に対応するデータ b_i
4	C	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NM	入 力	マージされるデータ c_j
5	NM	I	1	入 力	配列 C の大きさ
6	D	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NM	入 力	c_j に対応するデータ d_j
7	E	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NL	出 力	マージされたデータ e_k
8	NL	I	1	入 力	配列 E の大きさ
9	F	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NL	出 力	e_k に対応するデータ f_k
10	ISW	I	1	入 力	2次ソートスイッチ ISW=0: 2次ソートは行わない ISW=1: 2次ソートを行う
11	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

- (a) $NN \geq 1$
- (b) $NM \geq 1$
- (c) $1 \leq NL \leq NN + NM$
- (d) $A(1) \leq A(2) \leq \dots \leq A(NN)$
- (e) $C(1) \leq C(2) \leq \dots \leq C(NM)$
- (f) $ISW=0$ または $ISW=1$
- (g) $ISW=1$ を指定した場合には $A(i) = A(i+1)$ を満たす任意の i について $B(i) \leq B(i+1)$ を満たすこと
- (h) $ISW=1$ を指定した場合には $C(j) = C(j+1)$ を満たす任意の j について $D(j) \leq D(j+1)$ を満たすこと

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1000	制限条件 (c) を満足しなかった.	$NL = NN + NM$ として処理する.
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
3100	制限条件 (b) を満足しなかった.	
3200	制限条件 (d) を満足しなかった.	
3300	制限条件 (e) を満足しなかった.	
3400	制限条件 (f) を満足しなかった.	
3500	制限条件 (g) を満足しなかった.	
3600	制限条件 (h) を満足しなかった.	

(6) 注意事項

- (a) $NL < NN + NM$ のとき、小さいものから NL 個のみマージした結果が出力される.

(7) 使用例

(a) 問題

配列 X と Y に格納された n 個のデータの組からなる列 (a_i, b_i) ($i = 1, 2, \dots, n$) を n_s 個ずつの部分列に分割して、部分列間のマージを繰り返すことにより元の n 個のデータの組全体を昇順にソートする. 2 次ソートも行う. なお、プログラム内の配列のかわりに外部記憶装置上のファイルを用いれば、この問題は n 個のデータの組 (a_i, b_i) を外部ソートする問題となる.

(b) 入力データ

配列 X, 数列の長さ n , $ISW=1$

(c) 主プログラム

```

PROGRAM BSMGPA
! *** EXAMPLE OF DSMGPA ***
IMPLICIT NONE
!
INTEGER NA
PARAMETER( NA = 100 )
INTEGER N, ISW, IERR
INTEGER ILOOP, ISIZE, ISIZE2, IA, IB, IC, ICREST
INTEGER ISTA, ISTB, ISTC, ISIZEA, ISIZEB, ISIZEC
INTEGER I, J, K
REAL(8) PI
PARAMETER( PI = 3.1415926535897932384D0 )
REAL(8) X(NA), A(NA), B(NA), C(NA)
REAL(8) Y(NA), A2(NA), B2(NA), C2(NA)
!
! DATA SET
    
```

```

N = 17
DO 100 I=1,N
  X(I) = DBLE( INT(SIN(DBLE(I))*100) )
  Y(I) = DBLE( INT(SIN(DBLE(I)+PI*0.5D0)*100) )
100 CONTINUE
ISW = 1
!
!
WRITE INPUT DATA
WRITE(6,6000) N,ISW
DO 110 I=1,N
  WRITE(6,6010) I,X(I),I,Y(I)
110 CONTINUE
!
!
EXTERNAL SORT
DO 120 I=1,N
  C(I) = X(I)
  C2(I) = Y(I)
120 CONTINUE
IF( N .EQ. 1 )THEN
  GOTO 130
ENDIF
!
ILOOP = 0
DO 140 I=1,N
  ILOOP = ILOOP + 1
  IF( 2**ILOOP .GE. N ) THEN
    GOTO 150
  ENDIF
140 CONTINUE
150 CONTINUE
DO 160 I=1,ILOOP
  ISIZE = 2 ** (I-1)
  ISIZE2 = 2 ** I
!
  IA = 0
  IB = 0
  IC = 0
  DO 170 J=1,N/ISIZE2
    DO 180 K=1,ISIZE
      IA = IA + 1
      A(IA) = C(IC+K)
      A2(IA) = C2(IC+K)
180 CONTINUE
      IC = IC + ISIZE
      DO 190 K=1,ISIZE
        IB = IB + 1
        B(IB) = C(IC+K)
        B2(IB) = C2(IC+K)
190 CONTINUE
      IC = IC + ISIZE
170 CONTINUE
      ICREST = N - IC
      IF( (0 .LT. ICREST) .AND. (ICREST .LE. ISIZE) ) THEN
        DO 200 K=1,ICREST
          IA = IA + 1
          A(IA) = C(IC+K)
          A2(IA) = C2(IC+K)
200 CONTINUE
        ENDIF
        IF( (ISIZE .LT. ICREST) .AND. (ICREST .LT. (ISIZE*2)) ) THEN
          DO 210 K=1,ISIZE
            IA = IA + 1
            A(IA) = C(IC+K)
            A2(IA) = C2(IC+K)
210 CONTINUE
            IC = IC + ISIZE
            DO 220 K=1,(ICREST-ISIZE)
              IB = IB + 1
              B(IB) = C(IC+K)
              B2(IB) = C2(IC+K)
220 CONTINUE
            ENDIF
          !
          DO 230 J=1,N/ISIZE2
            ISTA = (J-1) * ISIZE + 1
            ISTB = (J-1) * ISIZE + 1
            ISTD = (J-1) * ISIZE2 + 1
            ISIZEA = ISIZE
            ISIZEB = ISIZE
            ISIZEC = ISIZEA + ISIZEB
            CALL DSMGPA&
              (A(ISTA),ISIZEA,A2(ISTA),B(ISTB),ISIZEB,B2(ISTB),&
                C(ISTD),ISIZEC,C2(ISTD),ISW,IERR)
230 CONTINUE
            IF( (0 .LT. ICREST) .AND. (ICREST .LE. ISIZE) ) THEN
              ISTA = N/ISIZE2 * ISIZE
              ISTD = N/ISIZE2 * ISIZE2
              ISIZEA = ICREST
              DO 240 K=1,ISIZEA
                C(ISTC+K) = A(ISTA+K)
                C2(ISTC+K) = A2(ISTA+K)
240 CONTINUE
              ENDIF
              IF( (ISIZE .LT. ICREST) .AND. (ICREST .LT. (ISIZE*2)) ) THEN
                ISTA = N/ISIZE2 * ISIZE + 1
                ISTB = N/ISIZE2 * ISIZE + 1
                ISTD = N/ISIZE2 * ISIZE2 + 1
                ISIZEA = ISIZE
                ISIZEB = ICREST - ISIZE
                ISIZEC = ISIZEA + ISIZEB
                CALL DSMGPA&

```

```

        (A(ISTA), ISIZEA, A2(ISTA), B(ISTB), ISIZEB, B2(ISTB), &
        C(ISTC), ISIZEC, C2(ISTC), ISW, IERR)
    ENDIF
160 CONTINUE
130 CONTINUE
!
!   WRITE OUTPUT DATA
    WRITE(6,6020) IERR
    IF( IERR .LT. 3000 ) THEN
        DO 250 I=1,N
            WRITE(6,6030) I, C(I), I, C2(I)
250     CONTINUE
        ENDIF
!
    STOP
6000 FORMAT(/, &
           1X, '*** DSMGPA ***', /, /, &
           1X, ' ** INPUT **', /, /, &
           1X, ' N =', I6, /, &
           1X, ' ISW =', I6, /)
6010 FORMAT(1X, ' X(', I2, ')=' , F5.1, 7X, ' Y(', I2, ')=' , F5.1)
6020 FORMAT(/, &
           1X, ' ** OUTPUT **', /, /, &
           1X, ' IERR =', I5, /)
6030 FORMAT(1X, ' C(', I2, ')=' , F5.1, 7X, ' C2(', I2, ')=' , F5.1)
    END

```

(d) 出力結果

```

*** DSMGPA ***

** INPUT **

N = 17
ISW = 1

X( 1)= 84.0      Y( 1)= 54.0
X( 2)= 90.0      Y( 2)=-41.0
X( 3)= 14.0      Y( 3)=-98.0
X( 4)=-75.0      Y( 4)=-65.0
X( 5)=-95.0      Y( 5)= 28.0
X( 6)=-27.0      Y( 6)= 96.0
X( 7)= 65.0      Y( 7)= 75.0
X( 8)= 98.0      Y( 8)=-14.0
X( 9)= 41.0      Y( 9)=-91.0
X(10)=-54.0      Y(10)=-83.0
X(11)=-99.0      Y(11)= 0.0
X(12)=-53.0      Y(12)= 84.0
X(13)= 42.0      Y(13)= 90.0
X(14)= 99.0      Y(14)= 13.0
X(15)= 65.0      Y(15)=-75.0
X(16)=-28.0      Y(16)=-95.0
X(17)=-96.0      Y(17)=-27.0

** OUTPUT **

IERR = 0

C( 1)=-99.0      C2( 1)= 0.0
C( 2)=-96.0      C2( 2)=-27.0
C( 3)=-95.0      C2( 3)= 28.0
C( 4)=-75.0      C2( 4)=-65.0
C( 5)=-54.0      C2( 5)=-83.0
C( 6)=-53.0      C2( 6)= 84.0
C( 7)=-28.0      C2( 7)=-95.0
C( 8)=-27.0      C2( 8)= 96.0
C( 9)= 14.0      C2( 9)=-98.0
C(10)= 41.0      C2(10)=-91.0
C(11)= 42.0      C2(11)= 90.0
C(12)= 65.0      C2(12)=-75.0
C(13)= 65.0      C2(13)= 75.0
C(14)= 84.0      C2(14)= 54.0
C(15)= 90.0      C2(15)=-41.0
C(16)= 98.0      C2(16)=-14.0
C(17)= 99.0      C2(17)= 13.0

```


第 4 章 方程式の根

4.1 概要

本章では、代数方程式や非線形方程式、連立非線形方程式の根を求めるサブルーチンについて説明する。本ライブラリでは、代数方程式として次の 2 種類を用意している。

- (1) 実係数実数型入力で複素根実数型出力タイプ
- (2) 複素係数複素型入力で複素根複素型出力タイプ

また非線形方程式としては次の 3 種類を用意している。

- (1) 初期値を与えて 1 根を求めるもの
- (2) 区間を与えて 1 根を求めるもの
- (3) 区間内のすべての根を求めるもの

このうち初期値を与えて 1 根を求める実数型のものは、初期値が根から遠い、あるいは根と初期値の間で関数が振動する場合でも根が求まるように特別の配慮がなされている。

連立非線形方程式では、ヤコビ行列計算サブルーチンが与えられる場合と与えられない場合がある。どちらも大域的収束性があり、連立する各式がスケールリングされていなくても解けるような配慮がされている。

4.1.1 使用上の注意

- (1) 代数方程式は、実係数代数方程式については 0 に近い根から求まる傾向があるが、複素係数代数方程式についてはこの傾向はない。
- (2) 重根があると、根の多重度を n としてそこでの解の精度は、代数方程式では $\sqrt[n]{\text{(誤差判定のための単位)}}$ 程度、初期値を与えて解を得る非線形方程式では $n \times \text{(要求精度)}$ 程度になる。
- (3) 非線形方程式の収束判定は、区間内の 1 根を求めるものを除き、 e_r を要求精度、 ε を誤差判定のための単位、 Δx を x の更新量として

$$|\Delta x| < e_r \max(1, |x|) \text{ and } |f(x)| < e_r + 64\varepsilon|x|$$

または

$$f(x) = 0$$

が成立すれば収束したものと見なす。すなわち、関数値が 0 または、関数値も解の動きもともに 0 に近いときに収束したものとす。これにより条件の悪い場合でも正しく収束判定がなされ、しかも重根の場合には演算量が増えるが精度を保つことができる。

ただし、解の動きのみで判定する方法や関数値のみで判定する方法、または解の動きと関数値のいずれかで判定する方法に比べて判定がきびしくなる傾向がある。

- (4) 連立非線形方程式の収束判定は、
 $(|\Delta x_i| < e_r \max(1, |x_i|) \text{ and } \|f(x)\|_\infty < e_r + 64\varepsilon|x_i|) \text{ or } f_i(x) = 0$
 $(i = 1, \dots, \text{元数})$ がすべての i に対し成立したとき収束したものとす。
- (5) 非線形方程式および連立非線形方程式では、これらのサブルーチンを使用するプログラム内で次の注意が必要である。

① 実引数として使用する方程式定義関数名や方程式定義サブルーチン名を EXTERNAL 文を用いて宣言しなければならない。

例 非線形方程式

- 主プログラム $\left(\begin{array}{l} \text{F, DF は主プログラム, 関数副プロ} \\ \text{グラムで同じ名前とする。} \end{array} \right)$

```

}
EXTERNAL   F, DF
}
CALL      RLNRDS (F, DF , X, ER, NEV, IERR)
}
END
    
```

- 関数副プログラム

```

FUNCTION   F(x)
}
F=(f(x))
RETURN
END
FUNCTION   DF(x)
}
DF=(f'(x))
RETURN
END
    
```

例 連立非線形方程式

- 主プログラム (SUB, SUBJ は主プログラム, サブルーチン)
副プログラムで同じ名前とする.

```

}
EXTERNAL  SUB, SUBJ
}
CALL  RLSRDS (SUB, SUBJ, X, N, ER, NEV, IWK, WK, DWK, IERR)
}
END

```

- サブルーチン副プログラム

```

SUBROUTINE  SUB(X, N, F)
REAL  X(N), F(N)
F(1) =  $f_1(x_1, \dots, x_N)$ 
}
F(N) =  $f_N(x_1, \dots, x_N)$ 
RETURN
END
SUBROUTINE  SUBJ(X, N, A)
REAL  X(N), A(N, N)
A(1, 1) =  $\partial f_1 / \partial x_1$ 
}
A(N, N) =  $\partial f_N / \partial x_N$ 
RETURN
END

```

- (6) いくつかのエラーが重なって発生したとき、エラーインディケータには最も重大なエラー値が出力されるので、他のエラー情報が隠れてしまう場合がある.

4.1.2 使用しているアルゴリズム

4.1.2.1 実係数代数方程式の根

4.1.2.1.1 次数 $n = 2$ の場合

2 次方程式の根の公式を用いてつぎのように根を求める.

2 次方程式

$$x^2 + a_1x + a_0 = 0$$

において

$$r = -\frac{a_1}{2}$$

$$D = r^2 - a_0$$

とすると

(1) $|D| \leq \varepsilon$ のとき

$$x = r, r$$

(2) $D < 0$ のとき

$$x = (r, \pm\sqrt{-D})$$

(3) $D > 0$ のとき

$$x = \alpha, \frac{a_0}{\alpha}$$

ただし

$$\alpha = \begin{cases} r + \sqrt{D} & (r \geq 0) \\ r - \sqrt{D} & (r < 0) \end{cases}$$

である.

4.1.2.1.2 次数 $n = 3$ の場合

カルダノ法を用いて次のように根を求める.

3 次方程式

$$x^3 + a_2x^2 + a_1x + a_0 = 0$$

において

(1) $|a_0| \leq \varepsilon$ の場合 ($x(x^2 + a_2x + a_1) = 0$)

(4.1.2.1.1) に述べた方法により 2 次方程式

$$x^2 + a_2x + a_1 = 0$$

を解き根 α, β を求めることによって

$$x = 0, \alpha, \beta$$

と求まる.

(2) $|a_2| \leq \varepsilon$ の場合 ($x^3 + a_1x + a_0 = 0$)

(a) $|a_1| \leq \varepsilon$ のとき ($x^3 + a_0 = 0$)

$$x = r, \left(-\frac{r}{2}, \pm \frac{\sqrt{3}r}{2} \right)$$

ただし

$$r = \begin{cases} -\sqrt[3]{a_0} & (a_0 \geq 0) \\ \sqrt[3]{-a_0} & (a_0 < 0) \end{cases}$$

である.

(b) それ以外の場合

$$\begin{aligned} b_1 &= \frac{a_1}{3} \\ b_0 &= -\frac{a_0}{2} \end{aligned}$$

として,

$$x^3 + 3b_1x - 2b_0 = 0$$

を次の (3) に示す方法を用いて解く.

(3) それ以外の場合

3 次方程式

$$x^3 + a_2x^2 + a_1x + a_0 = 0$$

は

$$\begin{aligned} x &= y - YMX \\ YMX &= \frac{a_2}{3} \end{aligned}$$

なる変数変換によって

$$y^3 + 3b_1y - 2b_0 = 0$$

と変形できる.

ここで,

$$\begin{aligned} b_1 &= \frac{3a_1 - a_2^2}{9} \\ b_0 &= \frac{(9a_1 - 2a_2^2)a_2 - 27a_0}{54} \end{aligned}$$

である.

(a) $|b_0| \leq \varepsilon$ の場合 ($y(y^2 + 3b_1) = 0$)

i. $b_1 < 0$ のとき

$$\begin{aligned} y &= 0, \pm \sqrt{-3b_1} \\ x &= -YMX, \pm \sqrt{-3b_1} - YMX \end{aligned}$$

なお, 計算上桁落ちが生じる場合には根と係数の関係を用いることで桁落ちを防止する.

ii. $b_1 \geq 0$ のとき

$$y = 0, (0, \pm\sqrt{-3b_1})$$

$$x = -YMX, (-YMX, \pm\sqrt{-3b_1})$$

(b) $|b_1| \leq \varepsilon$ の場合 ($y^3 - 2b_0 = 0$)

$$y = r, \left(-\frac{r}{2}, \pm\frac{\sqrt{3}r}{2}\right)$$

$$x = r - YMX, \left(-\frac{r}{2} - YMX, \pm\frac{\sqrt{3}r}{2}\right)$$

ただし,

$$r = \begin{cases} \sqrt[3]{2b_0} & (b_0 \geq 0) \\ -\sqrt[3]{-2b_0} & (b_0 < 0) \end{cases}$$

なお, 計算上桁落ちが生じる場合には根と係数の関係を用いることで桁落ちを防止する.

(c) それ以外の場合 ($y^3 + 3b_1y - 2b_0 = 0$)

いま,

$$y = s + t$$

とおくと,

$$s^3 + t^3 - 2b_0 + 3(st + b_1)(s + t) = 0$$

が成立する.

s, t を

$$\begin{cases} st = -b_1 \\ s^3 + t^3 = 2b_0 \end{cases}$$

が成り立つように決定すれば, これから y を求めることができる. s^3, t^3 は

$$z^2 - 2b_0z - b_1^3 = 0$$

の 2 根となっている.

$$D = b_0^2 + b_1^3 = D_s^2 D_d$$

$$D_s = \begin{cases} b_0 & (|b_0| \geq |b_1|) \\ b_1 & (|b_0| < |b_1|) \end{cases}$$

$$D_d = \begin{cases} 1 + b_1\left(\frac{b_1}{b_0}\right)^2 & (|b_0| \geq |b_1|) \\ \left(\frac{b_0}{b_1}\right)^2 + b_1 & (|b_0| < |b_1|) \end{cases}$$

とおくと

i. $|D_d| \leq \varepsilon^2$ の場合 ($(z - b_0)^2 = 0$)

$$s^3 = t^3 = b_0$$

$$s, t = r, \left(-\frac{r}{2}, \pm\frac{\sqrt{3}r}{2}\right)$$

ただし,

$$r = \begin{cases} \sqrt[3]{b_0} & (b_0 \geq 0) \\ -\sqrt[3]{-b_0} & (b_0 < 0) \end{cases}$$

s, t は $st = -b_1$ を満たすことから

$$\begin{aligned} y &= 2r, -r, -r \\ x &= 2r - YMX, -r - YMX, -r - YMX \end{aligned}$$

が得られる.

なお, 計算上桁落ちが生じる場合には根と係数の関係を用いることで桁落ちを防止する.

ii. $D < 0$ の場合

$$\begin{aligned} s^3 &= (b_0, \sqrt{-D}) \\ t^3 &= (b_0, -\sqrt{-D}) \end{aligned}$$

いま,

$$|s^3|^2 = |t^3|^2 = -b_1^3$$

であるから,

$$\begin{aligned} s &= \frac{r}{2} e^{\sqrt{-1}\frac{\theta}{3}}, \frac{r}{2} e^{\sqrt{-1}(\pi+\frac{\theta-\pi}{3})}, \frac{r}{2} e^{\sqrt{-1}(\frac{\theta+\pi}{3}-\pi)} \\ t &= \frac{r}{2} e^{-\sqrt{-1}\frac{\theta}{3}}, \frac{r}{2} e^{-\sqrt{-1}(\pi+\frac{\theta-\pi}{3})}, \frac{r}{2} e^{-\sqrt{-1}(\frac{\theta+\pi}{3}-\pi)} \end{aligned}$$

したがって,

$$\begin{aligned} y &= r \cos \frac{\theta}{3}, -r \cos \frac{\pi-\theta}{3}, -r \cos \frac{\pi+\theta}{3} \\ x &= r \cos \frac{\theta}{3} - YMX, -r \cos \frac{\pi-\theta}{3} - YMX, -r \cos \frac{\pi+\theta}{3} - YMX \end{aligned}$$

ただし,

$$\begin{aligned} r &= \begin{cases} -2\sqrt{b_1} & (b_1 \geq 0) \\ 2\sqrt{-b_1} & (b_1 < 0) \end{cases} \\ \theta &= \begin{cases} \tan^{-1} D_\theta & (b_0 \geq 0) \\ \pi - \tan^{-1} D_\theta & (b_0 < 0) \end{cases} \\ D_\theta &= \frac{|D|}{|b_0|} = \begin{cases} \frac{\sqrt{|D_d|}}{|b_0|} & (|b_0| \geq |b_1|) \\ \frac{|D_s|\sqrt{|D_d|}}{|b_0|} & (|b_0| < |b_1|) \end{cases} \end{aligned}$$

なお, 計算上桁落ちが生じる場合には根と係数の関係を用いることで桁落ちを防止する.

iii. $D > 0$ の場合

$$\begin{aligned} s^3 &= \alpha \\ t^3 &= \beta \\ \alpha &= \begin{cases} b_0 + \sqrt{D} = |D_s|r & (b_0 \geq 0) \\ b_0 - \sqrt{D} = -|D_s|r & (b_0 < 0) \end{cases} \\ \beta &= -\frac{b_1^3}{\alpha} \end{aligned}$$

ただし,

$$r = \frac{|b_0| + \sqrt{D}}{|D_s|} = \begin{cases} \frac{1 + \sqrt{|D_d|}}{|D_s|} & (|b_0| \geq |b_1|) \\ \frac{|b_0|}{|D_s|} + \sqrt{|D_d|} & (|b_0| < |b_1|) \end{cases}$$

である. したがって,

$$\begin{aligned} s &= \alpha', \left(-\frac{\alpha'}{2}, \pm \frac{\sqrt{3}\alpha'}{2} \right) \\ t &= \beta', \left(-\frac{\beta'}{2}, \pm \frac{\sqrt{3}\beta'}{2} \right) \end{aligned}$$

ここで,

$$\alpha' = \begin{cases} \sqrt[3]{|\alpha|} & (b_0 \geq 0) \\ -\sqrt[3]{|\alpha|} & (b_0 < 0) \end{cases}$$

$$\beta' = \begin{cases} \sqrt[3]{|\beta|} & (b_0 b_1 < 0) \\ -\sqrt[3]{|\beta|} & (b_0 b_1 \geq 0) \end{cases}$$

s, t は $st = -b_1$ を満たすことから

$$y = u, \left(-\frac{u}{2}, \pm \frac{\sqrt{3}v}{2} \right)$$

$$x = u - YMX, \left(-\frac{u}{2} - YMX, \pm \frac{\sqrt{3}v}{2} \right)$$

ただし,

$$u = \alpha' + \beta', v = \alpha' - \beta'$$

である.

なお, 計算上桁落ちが生じる場合には根と係数の関係を用いることで桁落ちを防止する.

4.1.2.1.3 次数 $n = 4$ の場合

フェラーリ法を用いて次のように根を求める.

4 次方程式

$$x^4 + a_3x^3 + a_2x^2 + a_1x + a_0 = 0$$

において

- (1) $|a_0| \leq \varepsilon$ の場合 ($x(x^3 + a_3x^2 + a_2x + a_1) = 0$)
 (4.1.2.1.2) に述べた方法によって 3 次方程式

$$x^3 + a_3x^2 + a_2x + a_1 = 0$$

を解き, その根 α, β, γ を求めることによって

$$x = 0, \alpha, \beta, \gamma$$

と求まる.

- (2) $|a_3| < \varepsilon$ の場合 ($x^4 + a_2x^2 + a_1x + a_0 = 0$)

- (a) $|a_1| \leq \varepsilon$ のとき ($x^4 + a_2x^2 + a_0 = 0$)

$$r = -\frac{a_2}{2}$$

$$D = r^2 - a_0$$

とおくと

- i. $|D| \leq \varepsilon$ のとき ($(x^2 - r)^2 = 0$)

$$x = \begin{cases} \sqrt{r}, \sqrt{r}, -\sqrt{r}, -\sqrt{r} & (r \geq 0) \\ (0, \sqrt{-r}), (0, \sqrt{-r}), (0, -\sqrt{-r}), (0, -\sqrt{-r}) & (r < 0) \end{cases}$$

- ii. $D < 0$ のとき

$$x^2 = (r, \pm\sqrt{-D})$$

したがって,

$$x = (\alpha, \pm\beta), (-\alpha, \pm\beta)$$

ただし,

$$\begin{cases} \alpha = \sqrt{\frac{r + NRD}{2}}, & \beta = \frac{\sqrt{-D}}{2\alpha} & (r > 0) \\ \beta = \sqrt{\frac{-r + NRD}{2}}, & \alpha = \frac{\sqrt{-D}}{2\beta} & (r \leq 0) \end{cases}$$

$$NRD = \sqrt{r^2 - D} = \begin{cases} |r| \sqrt{1 + \frac{-D}{r^2}} & (|r| \geq \sqrt{-D}) \\ \sqrt{-D} \sqrt{\frac{r^2}{-D} + 1} & (|r| < \sqrt{-D}) \end{cases}$$

である.

iii. $D > 0$ のとき

$$x^2 = \alpha, \beta$$

ただし,

$$\alpha = \begin{cases} r + \sqrt{D} & (r \geq 0) \\ r - \sqrt{D} & (r < 0) \end{cases}$$

$$\beta = \frac{a_0}{\alpha}$$

したがって,

$$x = \begin{cases} \pm\sqrt{\alpha}, \pm\sqrt{\beta} & (r, \beta \geq 0) \\ \pm\sqrt{\alpha}, (0, \pm\sqrt{-\beta}) & (r \geq 0, \beta \leq 0) \\ (0 \pm \sqrt{-\alpha}), \pm\sqrt{\beta} & (r < 0, \beta \geq 0) \\ (0 \pm \sqrt{-\alpha}), (0 \pm \sqrt{-\beta}) & (r, \beta < 0) \end{cases}$$

である.

(b) それ以外のとき

$$x^4 + a_2x^2 + a_1x + a_0 = 0$$

を次の (3) に示す方法を用いて解く.

(3) それ以外の場合

4 次方程式

$$x^4 + a_3x^3 + a_2x^2 + a_1x + a_0 = 0$$

が与えられた時

$$x = y - YMX$$

$$YMX = \frac{a_3}{4}$$

なる変数変換を行うと

$$y^4 + b_2y^2 + b_1y + b_0 = 0$$

となる.

ただし,

$$b_2 = \frac{8a_2 - 3a_3^2}{8}$$

$$b_1 = \frac{8a_1 - a_3(4a_2 - a_3^2)}{8}$$

$$b_0 = \frac{256a_0 - a_3(64a_1 - a_3(16a_2 - 3a_3^2))}{256}$$

である.

(a) $b_1^2 \leq \varepsilon^2 \max(|4b_0b_2|, |b_2^3|)$ のとき ($y^4 + b_2y^2 + b_0 = 0$)

$$\begin{aligned} r &= -\frac{b_2}{2} \\ D &= r^2 - b_0 \end{aligned}$$

とおくと

i. $|D| \leq \varepsilon$ のとき

$$\begin{aligned} (y^2 - r)^2 &= 0 \\ y &= \begin{cases} \pm\sqrt{r}, \pm\sqrt{r} & (r \geq 0) \\ (0, \pm\sqrt{-r}), (0, \pm\sqrt{-r}) & (r < 0) \end{cases} \\ x &= \begin{cases} \pm\sqrt{r} - YMX, \pm\sqrt{r} - YMX & (r \geq 0) \\ (-YMX, \pm\sqrt{-r}), (-YMX, \pm\sqrt{-r}) & (r < 0) \end{cases} \end{aligned}$$

ii. $D < 0$ のとき

$$\begin{aligned} y^2 &= (r, \pm\sqrt{-D}) \\ y &= (\alpha, \pm\beta), (-\alpha, \pm\beta) \\ x &= (\alpha - YMX, \pm\beta), (-\alpha - YMX, \pm\beta) \end{aligned}$$

ただし,

$$\begin{cases} \alpha = \sqrt{\frac{r + NRD}{2}}, & \beta = \frac{\sqrt{-D}}{2\alpha} & (r > 0) \\ \beta = \sqrt{\frac{-r + NRD}{2}}, & \alpha = \frac{\sqrt{-D}}{2\beta} & (r \leq 0) \end{cases}$$

$$NRD = \sqrt{r^2 - D} = \begin{cases} |r| \sqrt{1 + \frac{-D}{r^2}} & (|r| > \sqrt{-D}) \\ \sqrt{-D} \sqrt{\frac{r^2}{-D} + 1} & (|r| < \sqrt{-D}) \end{cases}$$

である.

iii. $D > 0$ のとき

$$\begin{aligned} y^2 &= \alpha, \beta \\ \alpha &= \begin{cases} r + \sqrt{D} & (r \geq 0) \\ r - \sqrt{D} & (r < 0) \end{cases} \\ \beta &= \frac{b_0}{\alpha} \\ y &= \begin{cases} \pm\sqrt{\alpha}, \pm\sqrt{\beta} & (r, \beta \geq 0) \\ (0, \pm\sqrt{-\alpha}), \pm\sqrt{\beta} & (r < 0, \beta \geq 0) \\ \pm\sqrt{\alpha}, (0, \pm\sqrt{-\beta}) & (r \geq 0, \beta < 0) \\ (0, \pm\sqrt{-\alpha}), (0, \pm\sqrt{\beta}) & (r, \beta < 0) \end{cases} \\ x &= \begin{cases} \pm\sqrt{\alpha} - YMX, \pm\sqrt{\beta} - YMX & (r, \beta \geq 0) \\ (-YMX, \pm\sqrt{-\alpha}), \pm\sqrt{\beta} - YMX & (r < 0, \beta \geq 0) \\ \pm\sqrt{\alpha} - YMX, (-YMX, \pm\sqrt{-\beta}) & (r \geq 0, \beta < 0) \\ (-YMX, \pm\sqrt{-\alpha}), (-YMX, \pm\sqrt{-\beta}) & (r, \beta < 0) \end{cases} \end{aligned}$$

(b) $|b_0| \leq \varepsilon$ の場合 ($y(y^3 + b_2y + b_1) = 0$)

(4.1.2.1.2) に述べた方法によって 3 次方程式

$$y^3 + b_2y + b_1 = 0$$

を解き根 α, β, γ を求めることによって

$$x = -YMX, \alpha - YMX, \beta - YMX, \gamma - YMX$$

と求まる.

- (c) $b_1^2 > 10^{-4}|b_2(b_2^2 - 4b_0)|$ の場合
4 次方程式

$$y^4 + b_2y^2 + b_1y + b_0 = 0$$

の両辺に $py^2 + \frac{p^2}{4}$ を加えて変形すると

$$\left(y^2 + \frac{p}{2}\right)^2 = (p - b_2)y^2 - b_1y + \frac{p^2}{4} - b_0$$

となる. 右辺が一次式の平方の形になるためには

$$b_1^2 - (p - b_2)(p^2 - 4b_0) = 0$$

すなわち

$$p^3 - b_2p^2 - 4b_0p + (4b_2b_0 - b_1^2) = 0$$

を満足していれば良い. (4.1.2.1.2) に述べた方法によってこの方程式を解き, 得られたこの方程式の実根を p ととれば,

$$\left(y^2 + \frac{p}{2}\right)^2 = (p - b_2) \left\{y - \frac{b_1}{2(p - b_2)}\right\}^2$$

が成立するので, 2 次方程式

$$y^2 \pm \sqrt{p - b_2}y \mp \frac{b_1}{2\sqrt{p - b_2}} + \frac{p}{2} = 0 \quad (\text{複合同順})$$

を解くことによって解が次のように得られる.

$$\begin{aligned} y &= \alpha \pm \sqrt{\beta - \gamma}, -\alpha \pm \sqrt{\beta + \gamma} \\ \alpha &= \frac{\sqrt{p - b_2}}{2} \\ \beta &= -\frac{p + b_2}{4} \\ \gamma &= \frac{4b_1}{\alpha} \end{aligned}$$

ただし, $p \simeq b_2$ の場合, この方法では精度よく解を求められない. $p \simeq b_2$ となるのは

$$|p - b_2| = \left| \frac{b_1^2}{p^2 - 4b_0} \right| \simeq \left| \frac{b_1^2}{b_2^2 - 4b_0} \right| < \delta |b_2|$$

すなわち,

$$b_1^2 < \delta |b_2(b_2^2 - 4b_0)|$$

のときである. ここで, δ は十分小さな正数である.

$$b_1^2 \leq \varepsilon^2 \max(|b_2^3|, |4b_0b_2|)$$

の場合には前述のように b_1 を無視して

$$y^4 + b_2y^2 + b_0 = 0$$

を解いて解を求める. 一方,

$$10^{-4}|b_2(b_2^2 - 4b_0)| \geq b_1^2 > \varepsilon^2 \max(|b_2^3|, |4b_0b_2|)$$

の場合には次の (d) に述べる方法を用いて解を求める.

(d) それ以外の場合

4 次方程式

$$y^4 + b_2y^2 + b_1y + b_0 = 0$$

の根は 3 次方程式

$$z^3 + \frac{b_2}{2}z^2 + \frac{b_2^2 - 4b_0}{16}z - \frac{b_1^2}{64} = 0$$

の 3 根 z_1, z_2, z_3 (z_1, z_2, z_3 は一般には複素数) の 2 乗根を用いて

$$\begin{aligned} y = & \sqrt{z_1} + \sqrt{z_2} + \sqrt{z_3}, \\ & \sqrt{z_1} - (\sqrt{z_2} + \sqrt{z_3}), \\ & -\sqrt{z_1} + \sqrt{z_2} - \sqrt{z_3}, \\ & -\sqrt{z_1} - (\sqrt{z_2} - \sqrt{z_3}) \end{aligned}$$

と表せることが知られている。ただし、ここでは \sqrt{z} は z の 2 つある 2 乗根 ($\pm\sqrt{z}$) の一方を表す。2 乗根のどちらを選べば良いかは前もって分からないが、4 根の組み合わせは前述の y かまたはすべての根の符号を反転した $-y$ となるので、根と係数の関係から b_1 の符号によってどちらが根かの判定を行う。なお、3 次方程式は (4.1.2.1.2) に述べた方法で解く。

4.1.2.1.4 次数 $n > 4$ の場合

平野法を用いて根を求める。与えられた方程式を

$$P_n(x) = a_nx^n + a_{n-1}x^{n-1} + \cdots + a_1x + a_0 = 0$$

とする。平野法では、近似根の反復改良による根の探索と方程式の減次を交互に行うことによって方程式のすべての根を求める。近似根は原点を出発値とし、多項式 $P_n(x)$ をこの近似根の回りで展開したときの展開係数を用いて順次真根に近づけていく。このようにして得られた根 $x = (x_R, x_I)$ を用いて方程式の減次を行い、減次後の方程式について同様の操作を順次繰り返すことによって、方程式の全根を求める。

根 $x = (x_R, x_I)$ は次のように決定する。

(1) 根 x の近似値 $z = (z_R, z_I)$ の初期値を $z = 0$ とする。

(2) $|P_n(z + \zeta_m)| \leq \delta$ が成立するまで z を $z + \zeta_m$ と置き換え、次の計算を繰り返す。

(a) 多項式 $P_n(x)$ を z を中心として

$$P_n(\zeta + z) = c_n\zeta^n + c_{n-1}\zeta^{n-1} + \cdots + c_1\zeta + c_0 \quad (x = \zeta + z)$$

と展開し、その係数 c_k , ($k = n, n-1, \dots, 0$) を組立除法をもちいて計算する。

(b) $\mu = 1$ とおく。

(c) $\zeta_k(\mu)$ ($k = 1, \dots, n$) の内で絶対値最小のものを ζ_m とすると、 $|P_n(z + \zeta_m)| \leq (1 - \frac{\mu}{4}) |P_n(z)|$ が成立するまで順次 μ を $\frac{\mu}{2}$ と置き換えて計算を進める。

ただし、

$$\zeta_k(\mu) = (-\mu \frac{c_0}{c_k})^{\frac{1}{k}} \quad (k = 1, \dots, n) \text{ である.}$$

(3) $z + \zeta_m$ を方程式 $P_n(x) = 0$ の根とする。

ここで、収束判定値 δ は次のように決定する。

いま、係数 c_k , ($k = n, n-1, \dots, 0$) の計算に $\varepsilon |c_k|$ の誤差があり、 ζ の計算誤差が $\varepsilon |x|$ (x は真の根の値, $x = z + \zeta$) まで許されるとする。このとき $P_n(z + \zeta)$ の計算誤差 δ は、 $P_n(z + \zeta)$ が

$$b_n = c_n$$

$$b_k = b_{k+1}\zeta + c_k \quad (k = n-1, n-2, \dots, 0)$$

$$P_n(z + \zeta) = b_0$$

から計算できることから,

$$d_n = \varepsilon |c_n|$$

$$d_k = d_{k+1}(|\zeta| + \varepsilon |x|) + \varepsilon(|x| |b_{k+1}| + |c_k|) \quad (k = n-1, n-2, \dots, 0)$$

$$\delta = d_0$$

から計算できる. なお, 計算では, $\zeta = \zeta_m$ とし, x は $z + \zeta_m$ で近似する.

平野法では, 方程式を減次しながら根を順次求めて行くが, 各減次過程で誤差解析を行い, 減次の成功の判定を次のように行う.

(1) 1 実根による減次

方程式

$$P_n(x) = a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x + a_0 = 0$$

をその実根 z を用いて

$$P_n(x) = (x - z)(b_n x^{n-1} + b_{n-1} x^{n-2} + \dots + b_1) + b_0 = 0$$

と減次した場合,

$$b_n = a_n$$

$$b_k = a_k + b_{k+1}z \quad (k = n-1, n-2, \dots, 0)$$

として各係数は計算される. b_0 は本来 0.0 であるが, z に $\sqrt{\varepsilon}|z|$, a_k に $\varepsilon|a_k|$ の誤差まで許されるとすると, b_k に許される最大誤差 δ_k は

$$\delta_n = \varepsilon |a_n|$$

$$\delta_k = \varepsilon |a_k| + (\delta_{k+1} + \sqrt{\varepsilon} |b_{k+1}|) |z| \quad (k = n-1, n-2, \dots, 0)$$

と表せる. したがって, $|b_0| < \delta_0$ のとき減次が成功したものとす.

(2) 複素共役根による減次

方程式

$$P_n(x) = a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x + a_0 = 0$$

をその複素共役根 $z = (z_R, z_I), \bar{z} = (z_R, -z_I)$ を用いて

$$P_n(x) = (x - z)(x - \bar{z})(b_n x^{n-2} + b_{n-1} x^{n-3} + \dots + b_2) + b_1 x + b_0 = 0$$

と減次した場合,

$$b_n = a_n$$

$$b_{n-1} = a_{n-1} + 2z_R b_n$$

$$b_k = a_k + 2z_R b_{k+1} - (z_R^2 + z_I^2) b_{k+2} \quad (k = n-1, n-3, \dots, 0)$$

として各係数は計算される. b_0, b_1 は本来 0.0 であるが, z_R に $\sqrt{\varepsilon}|z_R|$, z_I に $\sqrt{\varepsilon}|z_I|$, a_k に $\varepsilon|a_k|$ の誤差まで許されるとすると, b_k に許される最大誤差 δ_k は

$$\delta_n = \varepsilon |a_n|$$

$$\delta_{n-1} = \varepsilon |a_{n-1}| + \delta_n |2z_R| + \sqrt{\varepsilon} |2z_R b_n|$$

$$\delta_k = \varepsilon |a_k| + \delta_{k+1} |2z_R| + \sqrt{\varepsilon} |2z_R b_{k+1}| + \delta_{k+2} (z_R^2 + z_I^2) + 2\sqrt{\varepsilon}$$

$$(|z_R| + |z_I|) |b_{k+2}| \quad (k = n-2, n-3, \dots, 0)$$

と表せる. したがって, $|b_0| < \delta_0, |b_1| < \delta_1$ のとき減次が成功したものとす.

4.1.2.2 複素係数代数方程式の根

デュラン-ケルナー (Durand-Kerner) の 3 次法で解く.

与えられた方程式を

$$P_n(Z) = a_0 Z^n + a_1 Z^{n-1} + \dots + a_{n-1} Z + a_n = 0 \quad (a_0 = 0)$$

とする.

全根を同時に求める反復公式は,

$$\begin{cases} z_i^{(\nu+1)} = z_i^{(\nu)} + \phi_i(z_1^{(\nu)}, \dots, z_n^{(\nu)}) \\ \phi_i(z_1^{(\nu)}, \dots, z_n^{(\nu)}) = \frac{P_n(z_i^{(\nu)})/P'_n(z_i^{(\nu)})}{1 - \frac{P_n(z_i^{(\nu)})}{P'_n(z_i^{(\nu)})} \sum_{\substack{j=1 \\ j \neq i}}^n \frac{1}{z_i^{(\nu)} - z_j^{(\nu)}}} \quad (i = 1, \dots, n) \end{cases}$$

によって求める. ここで $p'_n(z_i^{(\nu)})$ は, 次式により求める.

$$\begin{cases} z_i^{(\nu+1)} = z_i^{(\nu)} \Phi_i(z_1^{(\nu)}, \dots, z_n^{(\nu)}) \\ \Phi_i(z_1^{(\nu)}, \dots, z_n^{(\nu)}) = \frac{P_n(z_i^{(\nu)})/P'_n(z_i^{(\nu)})}{1 - \frac{P_n(z_i^{(\nu)})}{P'_n(z_i^{(\nu)})} \sum_{\substack{j=1 \\ j \neq i}}^n \frac{1}{z_i^{(\nu)} - z_j^{(\nu)}}} \quad \text{for } (i = 1, \dots, n) \end{cases}$$

$$\begin{cases} z_i^{(\nu+1)} = z_i^{(\nu)} \Phi_i(z_1^{(\nu)}, \dots, z_n^{(\nu)}) \\ \Phi_i(z_1^{(\nu)}, \dots, z_n^{(\nu)}) = \frac{P_n(z_i^{(\nu)})/P'_n(z_i^{(\nu)})}{1 - \frac{P_n(z_i^{(\nu)})}{P'_n(z_i^{(\nu)})} \sum_{\substack{j=1 \\ j \neq i}}^n \frac{1}{z_i^{(\nu)} - z_j^{(\nu)}}} \quad \text{for } (i = 1, \dots, n) \end{cases}$$

$$P'_n(z_i^{(\nu)}) = b_{n-1}^{(2)}$$

収束判定は, $P_n(z)$ の値がホーナー法で計算するときに含まれる誤差内になれば収束したものとみなす. すなわち, ε を誤差判定のための単位として,

$$\begin{cases} b_0^{(1)} = a_0 \\ b_0^{(2)} = a_0 \\ b_k^{(1)} = z_i^{(\nu)} b_{k-1}^{(1)} + a_k \\ b_k^{(2)} = z_i^{(\nu)} b_{k-1}^{(2)} + b_k^{(1)} \end{cases} \quad \text{for } (k = 1, \dots, n-1)$$

$$P'_n(z_i^{(\nu)}) = b_{n-1}^{(2)}$$

よって, $P_n(z_i^{(\nu)})$ の最大誤差の指定値 δ_n は,

$$\begin{cases} \delta_0 = 0.0 \\ \delta_k = \delta_{k-1} |z_i^{(\nu)}| + \varepsilon \{ |b'_k| + \max(|a_k|, |b'_k|, |b_k|) \} \quad (k = 1, \dots, n) \end{cases}$$

であり, $|P_n(z_i^{(\nu)})| \leq \delta_n$ ならば, 収束したとみなす.

反復は, 未収束の $z_i^{(\nu)}$ に対してのみ続けるようにし, すべて収束したとみなされたとき, 計算は終了する.

反復の初期値は次の 3 段階で決める.

(i) アッパーズの初期値の決定

$$\beta = \frac{\alpha_1 + \alpha_2 + \dots + \alpha_n}{n} = -\frac{a_1}{na_0}$$

を中心に, 全根を内部に含む半径 r を求める. 組立除法により,

$$P_n(Z) = P_n(\beta + \zeta) = c_0 \zeta^n + c_1 \zeta^{n-1} + \dots + c_{n-1} \zeta + c_n$$

とした係数 c_0, \dots, c_n を求める. c_1, \dots, c_n の中で 0 でないものの個数を m として,

$$r^+ = \max_{k=1, \dots, n} (m |c_k| / |c_0|)^{1/k}$$

r は r^+ を初期値として

$$Q_n(x) = |c_0| x^n - |c_1| x^{n-1} - \dots - |c_{n-1}| x - |c_n| = 0$$

となる x を求め、これを r にする。

(ii) 全根を含む円の半径を小さくする。

$$z = \beta + rw \text{ として}$$

$$\begin{aligned} P_n(\beta + rw) &= (c_0 r^n) w^n + (c_1 r^{n-1}) w^{n-1} + \dots + (c_{n-1} r) w + c_n \\ &= d_0^{(0)} w^n + d_1^{(0)} w^{n-1} + \dots + d_{n-1}^{(0)} w + d_n^{(0)} \\ &= \varphi_0(w) \end{aligned}$$

$$\varphi_\ell(w) = d_0^{(\ell)} w^{n_\ell} + d_1^{(\ell)} w^{n_\ell-1} + \dots + d_{n_\ell-1}^{(\ell)} w + d_{n_\ell}^{(\ell)} \quad (d_0^{(\ell)} \neq 0)$$

○ $|a_0^{(\ell)}| < |a_{n_\ell}^{(\ell)}|$ のとき

$$\varphi_{\ell+1}(w) = \varphi_\ell(w) - \frac{d_0^{(\ell)}}{d_{n_\ell}^{(\ell)}} \tilde{\varphi}_\ell(w)$$

○ $|a_0^{(\ell)}| \geq |a_{n_\ell}^{(\ell)}|$ のとき

$$\varphi_{\ell+1}(w) = \left\{ \varphi_\ell(w) - \frac{d_0^{(\ell)}}{d_{n_\ell}^{(\ell)}} \tilde{\varphi}_\ell(w) \right\} / w$$

ここで、

$$\tilde{\varphi}_\ell(w) = \overline{d_{n_\ell}^{(\ell)}} w^{n_\ell} + \overline{d_{n_\ell-1}^{(\ell)}} w^{n_\ell-1} + \dots + \overline{d_1^{(\ell)}} w + \overline{d_0^{(\ell)}}$$

として $\varphi_n =$ 定数まで定義できる。半径 r の円外の根の個数は、 φ_n まで求めたときの $|a_0^{(\ell)}| < |a_{n_\ell}^{(\ell)}|$ になった回数であり、円外はその残りである。全根を含む最小円の半径は、すべて $|a_0^{(\ell)}| > |a_{n_\ell}^{(\ell)}|$ になる条件でアップバースの初期値 r から 2 分法で求める。

(iii) 各根からの距離の 2 乗の総和が最小となる半径にする。

(ii) で求めた半径 r を 2 分しながら各半径の円の内外の根の個数を (ii) で示した方法で求める。 j 回の反復で幅 $r \times 2^{-j}$ の円環 D_i が求まる。円環 D_i の内径と外径の平均を r_i^* 、 D_i に含まれる根の個数を N_i とすると、求める半径 r は、

$$r = \left(\sum_{i=1}^m r_i^* N_i \right) / n$$

で得られる。初期値はこの r から次式で求める。

$$z_i^{(0)} = \beta + r \exp\left[i\left(\frac{2\pi(j-1)}{n} + \frac{3}{2n}\right)\right] \quad (j = 1, \dots, n, i = \sqrt{-1})$$

4.1.2.3 実関数の根 (初期値指定) (導関数定義必要)

与えられた非線形方程式を $f(x) = 0$ 、 $f(x)$ の導関数を $f'(x)$ として

$$x^{(\nu+1)} = x^{(\nu)} - f(x^{(\nu)}) / f'(x^{(\nu)})$$

のニュートン法を基本とする。ただし、この方法は関数が振動していたり、根の更新量が大きすぎると根が求まらないので、次のように改良する。

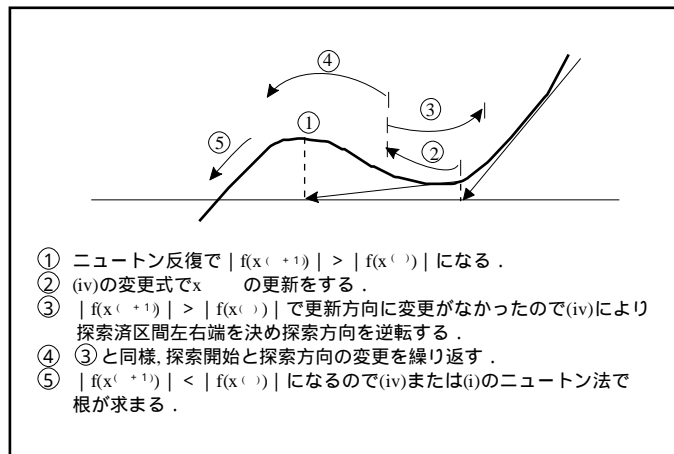
(i) ニュートン反復 $x^{(\nu+1)} = x^{(\nu)} - f(x^{(\nu)}) / f'(x^{(\nu)})$ を行う。

(ii) $f'(x^{(\nu)}) = 0$ または $|f(x^{(\nu+1)})| \geq |f(x^{(\nu)})|$ のとき (iii) に移る。それ以外は $x^{(\nu)} = x^{(\nu+1)}$ として (i) にもどる。

- (iii) $\nu = 1$ または $f'(x^{(\nu-1)}) \geq 0$ のとき $IS = -1$, それ以外は $IS = 1$ にする.
 $R = 1, P = 0, IOLD = \text{SIGN}\{1, IS \cdot f(x^{(\nu)})\}, I1 = -IOLD$ とする.
- (iv) $I2 = \text{SIGN}\{1, IS \cdot f(x^{(\nu)})\}$ とする.
 $I2 \cdot I1 > 0$ のとき $P = P + 1$ で加速, それ以外は $R = R + 1$ で減速されるようにする.
 (I1: 旧更新方向, I2: 新更新方向で, + は正, - は負方向の更新)
 次の反復で $x^{(\nu)}$ の更新をする.
 ε : 誤差判定のための単位として
 $x^{(\nu+1)} = x^{(\nu)} + IS \sinh^{-1}(f(x^{(\nu)}))2^{\{(p-3)/3-R\}} + |x^{(\nu)} + 1| I2\varepsilon$
 (IS: 大域的な関数の傾き推定で, + のとき右下がり, - のとき右上がり)
 この更新は, 少なくとも 3 回は連続して行う.
- (v) $f(x^{(\nu-1)}) \gg f(x^{(\nu)}) \gg f(x^{(\nu+1)})$ で $|f(x^{(\nu+1)})|$ が十分小さくなれば (i) のニュートン法にもどる.
- (vi) $|f(x^{(\nu+1)})| > |f(x^{(\nu)})|$ で, $f(x^{(\nu+1)})$ と $f(x^{(\nu)})$ が同符号であり $I2 = IOLD$ のとき, $IS = -IS, IOLD = IOLD, I1 = IOLD$ として関数の傾きと探索方向を修正する.
 初めてこの条件に入ったとき,
 探索済区間左端 $XP = x^{(\nu+1)}, FP = f(x^{(\nu+1)})$
 探索済区間右端 $XQ = XP, FQ = FP$
 2回目以後のとき

$I2 > 0$ のとき (更新方向が正)	$XQ = x^{(\nu+1)}, FQ = f(x^{(\nu+1)})$	で探索済区間右端更新
	$x^{(\nu)} = XP, f(x^{(\nu)}) = FP$	で探索済区間左端から始める.
$I2 \leq 0$ のとき (更新方向が負)	$XP = x^{(\nu+1)}, FP = f(x^{(\nu+1)})$	で探索済区間左端更新
	$x^{(\nu)} = XQ, f(x^{(\nu)}) = FQ$	で探索済区間右端から始める.
- P = 0 にし, R は谷から抜け出すように, 何度も連続して (vi) の条件に入るときは加速させるため $R = 2$, それ以外は $R = 0$ と設定し (iv) にもどる.
- (vii) (vi) の条件に入らなかったら, $I1 = I2, x^{(\nu)} = x^{(\nu+1)}$ として (iv) にもどる. 以上 (i) から (vii) の動作例を図示すると図 4-1 のようになる.

図 4-1



収束判定は, e_r を要求精度とし,

- $|x^{(\nu+1)} - x^{(\nu)}| < e_r \max(1, |x^{(\nu+1)}|)$ かつ $|f(x^{(\nu+1)})| < e_r + 64\varepsilon |x^{(\nu+1)}|$
- $f(x^{(\nu+1)}) = 0$

のいずれかを満たしたとき収束したものと見なすようにしている。

4.1.2.4 実関数の根 (初期値指定) (導関数定義不要)

基本的にはセカント法

$$x^{(\nu+1)} = x^{(\nu)} - f(x^{(\nu)}) \frac{x^{(\nu)} - x^{(\nu-1)}}{f(x^{(\nu)}) - f(x^{(\nu-1)})}$$

を採用し, (3) と同様の改良を行うが, $f(x) = 0$ に向かい谷になっている所は極値探索, 根の符号の反転が見つければ 2 分法を採用する. 具体的にアルゴリズムを次に示す.

(i) セカント法で解を更新する.

(ii) 関数値, 更新量ともに大きいとき, $x^{(\nu)}$ が根から遠いので安全のため,

$$\left. \begin{array}{l} XP = XQ = x^{(\nu)} \\ FP = FQ = f(x^{(\nu)}) \end{array} \right\} \text{(探索済区間左右端の設定, XP は左端, XQ は右端)}$$

$x^{(\nu+1)} = x^{(\nu)}, f^{(\nu+1)} = f^{(\nu)}$ として (iii) に移る.

$|f(x^{(\nu+1)})| \geq |f(x^{(\nu)})|$ のとき

$|(x^{(\nu)} - x^{(\nu-1)})/f(x^{(\nu-1)})| > 0.125$ のとき ($x^{(\nu)}$ に近いところに根があると予想される).

$$XP = XQ = x^{(\nu-1)}$$

$$FP = FQ = f(x^{(\nu-1)})$$

$$x^{(\nu+1)} = x^{(\nu-1)}, f(x^{(\nu+1)}) = f(x^{(\nu-1)})$$

として (iii) に行く.

$|(x^{(\nu)} - x^{(\nu-1)})/f(x^{(\nu-1)})| \leq 0.125$ のとき

正方向に更新されていたなら

$$\left. \begin{array}{l} XP = x^{(\nu-1)}, XQ = x^{(\nu+1)} \\ FP = f(x^{(\nu-1)}), FQ = f(x^{(\nu+1)}) \end{array} \right\} \text{(探索済区間の設定)}$$

として, $x^{(\nu-1)} \sim x^{(\nu+1)}$ 間で極値探索を行う.

負方向に更新されていたなら

$$\left. \begin{array}{l} XP = x^{(\nu+1)}, XQ = x^{(\nu-1)} \\ FP = f(x^{(\nu+1)}), FQ = f(x^{(\nu-1)}) \end{array} \right\} \text{(探索済区間の設定)}$$

として, $x^{(\nu+1)} \sim x^{(\nu-1)}$ 間で極値探索を行う.

極値探索で根が見つからなければ, (iii) に移る.

以上のような状況がなければ, $x^{(\nu)} = x^{(\nu+1)}$ として (i) にもどる.

(iii) $x^{(\nu-1)} \sim x^{(\nu)}$ で関数が右上りなら $IS = -1$, 左上りなら $IS = 1$ にする.

$R = 1, P = 0, IOLD = \text{SIGN}\{1, IS \cdot f(x^{(\nu+1)})\}, I1 = -IOLD$ とする.

(iv) $I2 = \text{SIGN}\{1, IS \cdot f(x^{(\nu+1)})\}$ とする.

$I2 \cdot I1 > 0$ のとき $P = P + 1$ で加速, それ以外は $R = R + 1$ で減速されるようにする.

(I1 : 旧更新方向, I2 : 新更新方向で + は正, - は負方向の更新)

$$\begin{array}{ll} x^{(\nu)} & = x^{(\nu+1)}, \quad f(x^{(\nu)}) & = f(x^{(\nu+1)}) \\ x^{(\nu-1)} & = x^{(\nu)}, \quad f(x^{(\nu-1)}) & = f(x^{(\nu)}) \end{array}$$

とする.

次の反復で $x^{(\nu)}$ の更新をする.

ε : 誤差判定のための単位として

$$x^{(\nu+1)} = x^{(\nu)} + IS \sinh^{-1}(f(x^{(\nu)}))2^{\{(P-3)/3-R\}+} |x^{(\nu)} + 1| I2_\varepsilon$$

(IS: 大域的な関数の傾きの推定で, + のとき右下がり, - のとき右上がり)

この更新は, 少なくとも 3 回は連続して行う.

(v) $f(x^{(\nu-1)}) \gg f(x^{(\nu)}) \gg f(x^{(\nu+1)})$ で $|f(x^{(\nu+1)})|$ が十分小さくなければ (i) のセカント法にもどる.

(vi) $|f(x^{(\nu-1)})| > |f(x^{(\nu)})| < |f(x^{(\nu+1)})|$ で $f(x^{(\nu-1)}) \sim f(x^{(\nu+1)})$ の符号が等しいとき,

- $I2 > 0$ (更新方向が正) のとき
 $XQ = x^{(\nu+1)}, FQ = f(x^{(\nu+1)})$ で探索済区間右端更新.
- $I2 \leq 0$ (更新方向が負) のとき
 $XP = x^{(\nu+1)}, FP = f(x^{(\nu+1)})$ で探索済区間左端更新.

そして, $x^{(\nu+1)} \sim x^{(\nu-1)}$ 間で極値探索を行う.

極値探索で根が見つからなければ, 解を更新してきた方向と反対方向の探索済区間端に $x^{(\nu+1)}, f(x^{(\nu+1)})$ をセットし,

$$IS = -IS, IOLD = -IOLD, I2 = IOLD, P = 0, R = 1$$

として反対方向に更新させるようにして (iv) にもどる.

(vii) $|f(x^{(\nu-1)})| < |f(x^{(\nu)})| < |f(x^{(\nu+1)})|$ で $f(x^{(\nu-1)}) \sim f(x^{(\nu+1)})$ の符号が等しいとき, 解を更新してきた方向と反対方向の探索区間端に $x^{(\nu+1)}, f(x^{(\nu+1)})$ をセットし,

$$IS = -IS, IOLD = -IOLD, I2 = IOLD, P = 0, R = 1$$

として反対方向に更新されるようにして (iv) にもどる.

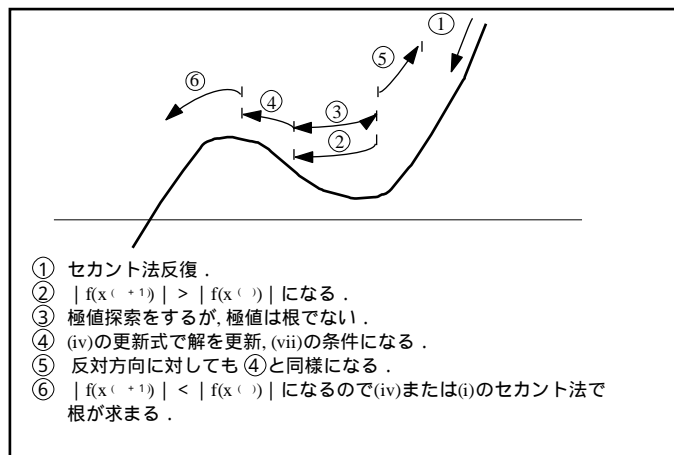
(viii) (v) から (vii) の条件に入らなかったら $I1 = I2$, として (iv) にもどる.

極値探索部は, 黄金分割探索と逐次放物線補間を組み合わせるもので, その詳細は第 5 章「極値問題」5.1.2「使用しているアルゴリズム」に述べている.

なお, 極値探索中に方程式の値の符号の逆転が見つければ, その点の初期値に近い方の根に対し 2 分法探索を行う.

以上 (i) から (viii) の動作例を図示すると, 図 4-2 のようになる.

図 4-2



収束判定は, e_r を要求精度とし,

- $(|x^{(\nu+1)} - x^{(\nu)}| < e_r \max(1, |x^{(\nu+1)}|) \text{ かつ } |f(x^{(\nu+1)})| < e_r + 64\varepsilon |x^{(\nu+1)}|$
- $f(x^{(\nu+1)}) = 0$

のいずれかを満たしたとき、収束したものとする。また極値探索や2分法では、
 (探索縮小区間) $< e_r \max(1, |x|)$ and $|f(x)| < e_r + 64\varepsilon |x|$
 のとき収束したものとする。

4.1.2.5 実関数の根 (区間指定) (導関数定義不要)

$f(x)$ を $[a, b]$ で連続かつ $f(a)$ と $f(b)$ で符号が逆転するものとする。

- (i) $c = a, d = e = b - a$ にする。
- (ii) $|f(c)| < |f(b)|$ ならば、 b と c を入れ替え、 a を入れ替え後の c にする。
 こうして常に $|f(b)| \leq |f(c)|$ になるようにしておき、 $[b, c]$ 間で根をさがす。
- (iii) $m = (c - b)/2$ とする。また、
 ε : 誤差判定のための単位, $e_r = \text{要求精度}$
 として、
 $\delta = (\varepsilon |b| + e_r)/2$
 とする。
- (iv) $|e| \geq \delta$ かつ $|f(b)| < |f(a)|$ のとき、
 $a = c$ のとき c と b の間で線形補間法を適用する。

$$\begin{cases} P = (c - b)f(b)/f(c) \\ Q = 1 - f(b)/f(c) \end{cases}$$
 $a = c$ ($|f(a)| \leq |f(c)|$) のとき a, b, c 間で逆2次補間法を適用する。

$$\begin{cases} r_1 = f(a)/f(c), r_2 = f(b)/f(c), r_3 = f(b)/f(a) \text{ として} \\ P = r_3\{(c - b)r_1(r_1 - r_2) - (b - a)(r_2 - 1)\} \\ Q = (r_1 - 1)(r_2 - 1)(r_3 - 1) \end{cases}$$
 $|P/Q| > \frac{3}{4} |c - b| - |\delta|$ または $|P/Q| > |e/2|$ のとき (v) に移る。
 これ以外のときは、
 $e = d$
 $d = -P/Q$ とする。
 $a = b$ にしておいて
 $b = b + \max(d, \text{sign}(\delta, c - b))$
- (v) (iv) の条件にならないときは、 $a = b$ にし b を $[b, c]$ の中点とする2分法を採用し、
 $d = e = m$ とする。
- (vi) $f(b)$ と $f(c)$ の符号が等しいときは、 $c = a$ にし、 $d = e = b - a$ にする。
 以上の操作の後、(ii) にもどる。
 収束判定は、 $f(b) = 0$, または $|c - b| \leq e_r + 2\varepsilon |b|$ のとき収束したもののみなす。

4.1.2.6 実関数の全根 (区間指定) (導関数定義不要)

基本は (4.1.2.1.2) 「導関数定義不要とした非線形方程式の1根」と同じアルゴリズムを利用するが、ここで (iv) で示した式が IS を使って探索方向を一定に保つことができることを利用し、常に区間の両端から内側に向かい根を求めていくようにしている。すなわち、次の点を変更している。

- (1) 解の探索除外区間を区間の両側からとっていく。
- (2) まず区間右端から根を探索し、根が見つければそこを区間右端に変更する。次に区間左端より根を探索する。根が見つければ次々にそこを区間左端にしていく。
- (3) 解の更新値が区間外に出たときは、区間内にもどして処理を続ける。
- (4) 極値探索中に方程式の値の符号が反転したときには、その左右双方の根を2分法で求める。
- (5) 区間 $\leq \delta$ になれば処理を終了する。

なお、根が見つかったとき区間縮小するが、その場合はその根より区間内側へ、次の δ だけずらすものとする。

$$\delta = \max(2e_r, (|A| + |B|)\varepsilon, \sqrt[3]{\varepsilon})$$

ここで e_r : 要求精度

ε : 誤差判定のための単位

A, B : 最初に設定した区間の左右端

4.1.2.7 複素関数の根 (初期値指定) (導関数定義不要)

マラー (Muller) 法によって解く。

出発値を、初期値 $z = 0$ のとき

$$\begin{cases} z_1 = -1.0 \\ z_2 = 1.0 \\ z_3 = 0.0 \end{cases}$$

初期値 $z = 0$ のとき

$$\begin{cases} z_1 = 0.9z \\ z_2 = 1.1z \\ z_3 = z \end{cases}$$

とする。

$f_i = f(z_i)$ として

$$q = (z_3 - z_2)/(z_2 - z_1)$$

$$q' = q + 1$$

$$a = qf_3 - qq'f_2 + q^2f_1$$

$$b = (q + q')f_3 - q'^2f_2 + q^2f_1$$

$$c = q'f_3$$

$r = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} = \frac{2c}{-b \mp \sqrt{b^2 - 4ac}}$ として絶対値の小さい方の r を採用する (r の分母が 0 のときで $f_1 = f_2 = f_3$ のときは、 $r = 1$ としておく)。

$$z = z_3 + r(z_3 - z_2)$$

$$\begin{cases} z_3 = z \\ z_2 = z_3 \\ z_1 = z_2 \end{cases}$$

として反復をくり返す。

ここで収束判定は、 e_r を要求精度、 ε を誤差判定のための単位として、

$(|z - z_3| < e_r \max(1, |z|) \text{ and } |f(z)| < e_r + 64\varepsilon |z|) \text{ or } f(z) = 0$

のとき収束したものとする。

4.1.2.8 連立非線形方程式の根 (ヤコビ行列定義任意)

マルカール (Marquardt) 法によって解く. その方法を以下に示す.

(i) $\lambda = 0.1$ とする.

(ii) $f(x)$ のヤコビ行列を A とする.

$$f(x) = (f_1, f_2, \dots, f_n)^T$$

$$x = (x_1, x_2, \dots, x_n)^T, n : \text{元数として}$$

$$A = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \dots & \frac{\partial f_1}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial f_n}{\partial x_1} & \dots & \frac{\partial f_n}{\partial x_n} \end{bmatrix} = \begin{bmatrix} a_{11} & \dots & a_{1n} \\ \vdots & & \vdots \\ a_{n1} & \dots & a_{nn} \end{bmatrix}$$

D を $A^T A$ の対角要素以外を 0 にした行列とする.

(iii) $A^T A + \lambda D \Delta x = -A^T f(x)$

の連立 1 次方程式を解き Δx を得る (x を $D^{\frac{1}{2}}$ でスケール変換したうえで付加項 λI のマルカール法を適用したのと同様の式).

(iv) $y = x + \Delta x$ として

$\|f(y)\| \geq \|f(x)\|$ ならば ($\|\dots\|$ 記号は, 2 乗ノルムとする)

$\lambda = 0$ のとき $\lambda = 0.001$,

$\lambda = 0$ のとき $\lambda = 10\lambda$ として (iii) にもどる.

$\|f(y)\| < \|f(x)\|$ ならば

前回も $\|f(y)\| < \|f(x)\|$ なら $\lambda = \lambda/10$ にする.

$x = y$ にして (ii) にもどる.

以上のマルカール法は正規方程式系を陽に形成するため, 誤差に対して弱いという欠点がある. そこで

e_r : 要求精度

ε : 誤差判定のための単位

として

$(\|\Delta x\|_\infty \leq e' \max(1, \|y\|_\infty)$ and $\|f(y)\|_\infty \leq e')$ or $\|f(y)\|_\infty = 0$

$e' = e_r^{0.2}, \|\dots\|_\infty$ は各要素の絶対値最大の値

に達したとき, または $\|f(y)\|$ が $4n$ 回反復前の $\|f(y)\|$ に対して 0.75 倍にしかならないときは, 次のスケールリング付ニュートン法に処理移行する.

スケールリング付ニュートン法

$$g_i = \max(\varepsilon, a_{i1}, a_{i2}, \dots, a_{in}) \quad (i = 1, \dots, n)$$

これより

$$(a_{i1}, a_{i2}, \dots, a_{in}) = (a_{i1}, a_{i2}, \dots, a_{in})/g_i$$

$$f_i = f_i/g_i$$

このようにスケールリングされたヤコビ行列 A および関数値 $f(x)$ を使って連立 1 次方程式 $A \Delta x = -f(x)$ を解き, Δx を修正ベクトルとして解を更新する.

収束判定は次の式が成立したとき収束するものとする.

$$(\|\Delta x\|_\infty < e_r \max(1, \|y\|_\infty)$$
 and $\|f(y)\|_\infty < e_r + 64\varepsilon \|y\|_\infty)$ or $\|f(y)\|_\infty = 0$

4.1.2.9 連立非線形方程式の根 (ヤコビ行列定義不要)

(1) 修正ベクトル Δx の計算

$f(x)$: 変数値 x での関数値 $f(x)$

A : ヤコビ行列 $\partial f / \partial x$

G : ヤコビ行列の逆行列

とする. まずガウスニュートン解を Δx_G , 最急降下解を Δx_S として

$$\Delta x_G = -Gf(x)$$

$$\Delta x_S = (\|b\|^2 / \|A b\|^2) b$$

ただし, $b = -A^T f(x)$

最初は, $\Delta x = \Delta x_S$ とし, $d = \|\Delta x_S\|$ とする.

2回目以後は, d をステップ幅とし,

(a) $d \leq \|\Delta x_S\|$ の場合

$$\Delta x = d \Delta x_S / \|\Delta x_S\|$$

(b) $\|\Delta x_S\| < d < \|\Delta x_G\|$ の場合

$$\Delta x = \alpha \Delta x_S + \beta \Delta x_G \quad (\alpha > 0, \beta > 0, \|\Delta x\| = d)$$

(c) $\|\Delta x_G\| \leq d$ の場合

$$\Delta x = \Delta x_G$$

(2) ステップサイズ d の変更

線形化した模型の関数値 2 乗和の差を Δs , 実際の関数値 2 乗和の差を ΔT として, その比 r を用いて非線形性の程度を測る.

$$\Delta s = \|f(x) - A\Delta x\|^2 - \|f(x)\|^2$$

$$\Delta T = \|f(x + \Delta x)\|^2 - \|f(x)\|^2$$

$$r = \Delta T / \Delta s$$

(a) $r < 0.1$ のとき d を半減させ, $\tau = 1.0$ とする.

(b) $r \geq 0.1$ のとき d の増加率 λ を以下のように計算する.

$$\lambda = \sqrt{1 - \frac{(r-0.1)\Delta S}{s_p + \sqrt{s_p^2 - (r-0.1)S_s\Delta S}}}$$

ただし, $\delta f = f(x + \Delta x) - \{f(x) + A\Delta x\}$ として

$$s_p = \sum_{i=1}^n \|f_i(x + \Delta x)\delta f_i\|$$

$$s_s = \|\delta f\|^2$$

実際には, d の振動を防ぐために 2 回続けて増加が要求されたときにだけ d を増加させる. また増加率は 2 以下に抑える. 実際増加率 μ は以下のように計算される.

$$\mu = \min(2, \lambda, \tau)$$

$$\tau = \lambda / \mu$$

ここで, τ は初期値を 1 とする.

また d には上限 d_{\max} と下限 d_{\min} が設けられており, その間に入るように制御されている.

(3) ヤコビ行列 A およびその逆行列 G の計算

ヤコビ行列は, 最初の 1 回だけは差分によって求め, 後は逐次更新していく. 同様に, その逆行列も最初の 1 回だけはヤコビ行列から求め, 後は逐次更新していく. 逐次更新の方法は以下の式によって計算する.

$$A = A + \delta f \Delta x^T / \|\Delta x\|^2$$

$$G = G + (\Delta x - G\Delta f)\Delta x^T G / \Delta x^T G \Delta f$$

ただし, $\Delta f = f(x + \Delta x) - f(x)$

(4) 修正ベクトルの独立性検査

ヤコビ行列の修正が効率よく行われるためには、順次取られる修正ベクトルが互いに直交に近いことが必要である。そのために、ハイブリッド法では、独自の独立性概念を定義して、修正ベクトルができるだけ独立な方向に取られるように制御している。ベクトル a が j 個のベクトル (a_1, a_2, \dots, a_j) と独立であるとは、 a がこれらのベクトルで張られた空間の任意のベクトルと 30 度以上の角度を成していることである。パウエルによって考案された独立性検査の算法を以下に示す。

過去 $2n$ 回のヤコビ行列の修正で用いられたベクトルのうち互いに独立な n 個のベクトルを直交化して、 $\Omega = (\omega_1, \omega_2, \dots, \omega_n)$ に保持しておく。大きさ n の配列 ℓ を用いて、 ω_i が何回前の修正ベクトルであったかの情報を保持しておく。すなわち、 ω_i は ℓ_i 回前に取られたベクトルであることを意味する。 Ω は単位行列で初期化し、 ℓ は $\ell_i = n - i + 1 (i = 1, \dots, n)$ で初期化する。

解の修正を行うときには、以下のようにする。

(a) $\Delta x = \Delta x_G$ の場合

独立性のいかにかわらず、 Δx を修正ベクトルとして採用する。

(b) $\Delta x = \Delta x_G$ でない場合

$j_1 < 2n$ であるか、または Δx が $(\omega_2, \omega_3, \dots, \omega_n)$ と独立、すなわち $\|\omega_1^T \Delta x\| > \|\Delta x\|/2$ なら、 Δx を修正ベクトルとして採用する。そうでないなら、解の修正を行わない。

ヤコビ行列の修正を行うときには以下のようにする。

$\|\Delta x\| < d_{\min}$ であるか、または、 $j_1 = 2n$ で Δx が $(\omega_2, \omega_3, \dots, \omega_n)$ と独立でないなら、 $\Delta x = d_{\min} \omega_1$ とする。そうでないなら、 Δx を採用する。

つぎに、 Ω と j の改訂を行う。 $\Delta x = d_{\min} \omega_1$ としたときには、

$$\omega_i = \omega_{i+1} (i = 1, \dots, n-1)$$

$$\omega_n = \omega_1$$

$$j_i = j_{i+1} + 1 (i = 1, \dots, n-1)$$

$$j_n = 1$$

とすればよい。そうでないときには、以下のようにする。

$(\omega_{k+1}, \omega_{k+2}, \dots, \omega_n, \Delta x)$ が互いに独立になる最小の k を求める。

$(\omega_1, \dots, \omega_{k-1}, \omega_{k+1}, \omega_{k+2}, \dots, \omega_n, \Delta x)$ を直交化して、それを改めて

$(\omega_1, \omega_2, \dots, \omega_n)$ とする。

$$j_i = j_i + 1 (i = 1, \dots, k-1)$$

$$j_i = j_{i+1} + 1 (i = k, \dots, n-1)$$

$$j_n = 1$$

このようにして、常に、 $j_1 \leq 2n$ となるようにする。

(5) スケーリング付ニュートン法への処理移行

与えられた問題の各方程式がスケーリングされていないと、解の変更過程に異常がおこり、本アルゴリズムでは収束しないので、下記の条件で解の変更過程の異常を検出し、スケーリング付ニュートン法へ処理を移行する。

$$(\|f(x + \Delta x)\|^2 - \|f(x)\|^2) / d > 10^{10}$$

なお、スケーリング付ニュートン法のアルゴリズムを以下に示す。

(a) ヤコビ行列の計算

ヤコビ行列は毎回差分によって求める。

(b) 修正ベクトルの計算

ヤコビ行列の (i, j) 成分を a_{ij} とする。まず次を求める。

$$g_i = \max(\text{誤差判定のための単位}, a_{i1}, a_{i2}, \dots, a_{iN})$$

$$(i = 1, 2, \dots, N)$$

これより

$$(a_{i1}, a_{i2}, \dots, a_{iN}) \leftarrow (a_{i1}, a_{i2}, \dots, a_{iN})/g_i$$

$$f_i \leftarrow f_i/g_i$$

$$(i = 1, 2, \dots, N)$$

このようにスケールされたヤコビ行列 A および関数値 $f(x)$ を使って連立1次方程式 $A\Delta x = -f(x)$ をクラウト法によって求める。この Δx が修正ベクトルとなる。

(6) 収束判定

収束判定は以下の式によって行う。

$$(\|\Delta x\|_\infty < e_r \max(1, \|x + \Delta x\|_\infty) \text{ and } \|f(x + \Delta x)\|_\infty < e_r + 64\varepsilon \|x + \Delta x\|)$$

$$\text{or } f(x + \Delta x) = 0$$

ここで, e_r は要求相対精度であり,

$$\|x\|_\infty = \max_i |x_i|$$

$$\|\Delta x\|_\infty = \max_i |\Delta x_i|$$

$$\|f(x + \Delta x)\|_\infty = \max_i |f_i| \text{ である.}$$

4.1.3 参考文献

- (1) 伊理正夫, “数値計算”, 朝倉書店, (1981).
- (2) 渡部力, 名取亮, 小国力編, “第3版数値解析とFORTRAN”, 丸善, (1983).
- (3) 森正武, “FORTRAN77 数値計算プログラミング”, 岩波コンピュータサイエンス, 岩波書店, (1986).
- (4) 中田多美, 川本則行, 名取亮, “代数方程式に対する Durand-Kerner 法の初期値の改良”, 情報処理学会論文誌, Vol. 29, No. 12, pp. 1200-1207, (1988).
- (5) 伊理正夫, 藤野和建, “数値計算の常識”, 共立出版, (1985).
- (6) Kantaris, N. , Howden, P. F. , 三井斌友訳, “方程式の数値解法”, 啓学出版, (1987).
- (7) Forsythe, G. E. , Malcolm, M. A. , and Moler, C. B. , 森正武訳, “計算機のための数値計算法”, 日本コンピュータ協会, (1978).
- (8) 一松信, 戸川隼人編, “数値計算における誤差”, 共立出版, (1975).
- (9) Ralston, A. , Rabinowitz, P. , 戸田英雄, 小野令美訳, “電子計算機のための数値解析の理論と応用 (下)”, プレイン図書 (株), (1986).
- (10) 田辺國土, “非線形方程式”, コンピュートロール, コロナ社, No12, pp. 36-39, (1985).
- (11) 中川徹, 小柳義夫, “最小二乗法による実験データ解析”, 東京大学出版会, (1982).

4.2 代数方程式

4.2.1 DLARHA, RLARHA

実係数代数方程式の根

(1) 機能

実数係数の代数方程式を解く (根は実数型配列).

(2) 使用法

倍精度サブルーチン:

CALL DLARHA (A, N, XR, XI, WK, IERR)

単精度サブルーチン:

CALL RLARHA (A, N, XR, XI, WK, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	A	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	N+1	入 力	代数方程式の係数 (高次項から A (1) , A (2) , ..., A (N+1))
2	N	I	1	入 力	代数方程式の次数
3	XR	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	N	出 力	代数方程式の根の実部
4	XI	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	N	出 力	代数方程式の根の虚部
5	WK	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	4 × (N+1)	ワー ーク	作業領域 N ≤ 4 のときは使用しない
6	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) A(i), i = 1, 2, ..., N のうち少なくとも 1 つが |A(i)| > (誤差判定の単位) を満たす.

(b) N ≥ 1

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
2000+i	i 番目までの根は正しく求められているが、それ以外は精度が悪い.	i + 1 番目以降の根の精度が悪いままで処理を終了する.
2500+j	$ A(i) \leq$ 誤差判定の単位, $i = 1, 2, \dots, j$ であった.	$A(i) = 0.0, i = 1, 2, \dots, j$ と見なして処理する.
3000	制限条件 (a) または (b) を満足しなかった.	処理を打ち切る.
4000+i	i 番目までの根は求めたが、それ以後の根は求められない.	i 番目までの根を計算して、処理を打ち切る.

(6) 注意事項

- (a) 根が 4 根以上あるときは、最後の 4 根を除き 0.0 に近い根から求まる傾向にある。
 (b) 重根のときの相対精度は根の多重度を n として、 $\sqrt[n]{}$ 誤差判定のための単位 程度になる。

(7) 使用例

(a) 問題

$$x^{10} - 55x^8 + 1023x^6 - 7645x^4 + 21076x^2 - 14400 = 0 \text{ を解く.}$$

(b) 入力データ

代数方程式の係数を格納した配列 A: A (1) =1.0, A (2) =0.0, A (3) = -55.0, A (4) =0.0, A (5) =1023.0, A (6) = 0.0, A (7) =-7645.0, A (8) =0.0, A (9) =21076.0, A (10) =0.0, A (11) =14400.0, N=10

(c) 主プログラム

```

PROGRAM BLARHA
! *** EXAMPLE OF DLARHA ***
IMPLICIT REAL(8) (A-H,O-Z)
INTEGER N,IERR,I
DIMENSION A(11),XR(10),XI(10),WK(44)
!
READ(5,*) N
READ(5,*) (A(I),I=1,N+1,1)
WRITE(6,1000)
WRITE(6,2000) N
WRITE(6,3000)
WRITE(6,4000) (A(I),I=1,N+1,1)
CALL DLARHA(A,N,XR,XI,WK,IERR)
WRITE(6,5000)
WRITE(6,6000) IERR
WRITE(6,7000)
WRITE(6,8000) (XR(I),XI(I),I=1,N,1)
STOP
!
1000 FORMAT(' ',/,5X,'*** DLARHA ***',/,&
7X,'* ALGEBRAIC EQUATION *',/,&
9X,'X**10-55*(X**8)+1023*(X**6)-7645*(X**4)+21076*(X**2)',&
'-14400=0',/,&
6X,'** INPUT **')
2000 FORMAT(9X,'DEGREE OF ALGEBRAIC EQUATION = ',I2)
3000 FORMAT(9X,'COEFFICIENTS OF ALGEBRAIC EQUATION',/)
4000 FORMAT(9X,F8.1)
5000 FORMAT(' ',/,/,6X,'** OUTPUT **')
6000 FORMAT(9X,'IERR = ',I4)
7000 FORMAT(9X,'ROOTS',/,&
9X,'REAL PART',13X,'IMAGINARY PART',/)
8000 FORMAT(9X,D17.10,5X,D17.10)
END
    
```

(d) 出力結果

```

*** DLARHA ***
* ALGEBRAIC EQUATION *
X**10-55*(X**8)+1023*(X**6)-7645*(X**4)+21076*(X**2)-14400=0
** INPUT **
DEGREE OF ALGEBRAIC EQUATION = 10
COEFFICIENTS OF ALGEBRAIC EQUATION
    
```

1.0

0.0
-55.0
0.0
1023.0
0.0
-7645.0
0.0
21076.0
0.0
-14400.0

** OUTPUT **
IERR = 0
ROOTS
REAL PART

IMAGINARY PART

0.1000000000D+01	0.0000000000D+00
-0.1000000000D+01	0.0000000000D+00
0.2000000000D+01	0.0000000000D+00
-0.2000000000D+01	0.0000000000D+00
0.3000000000D+01	0.0000000000D+00
-0.3000000000D+01	0.0000000000D+00
0.4000000000D+01	0.0000000000D+00
0.5000000000D+01	0.0000000000D+00
-0.5000000000D+01	0.0000000000D+00
-0.4000000000D+01	0.0000000000D+00

4.2.2 ZLACHA, CLACHA 複素係数代数方程式の根

(1) 機能

複素数係数の代数方程式を解く (係数, 根は複素数型配列).

(2) 使用法

倍精度サブルーチン:

CALL ZLACHA (CA, N, NEV, CX, WK, CWK, IERR)

単精度サブルーチン:

CALL CLACHA (CA, N, NEV, CX, WK, CWK, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	CA	$\begin{Bmatrix} Z \\ C \end{Bmatrix}$	N+1	入 力	代数方程式の係数 (高次項から CA (1) , CA (2) , ..., CA (N+1))
2	N	I	1	入 力	代数方程式の次数
3	NEV	I	1	入 力	最大関数評価回数 (既定値:100)
				出 力	実際の関数評価回数
4	CX	$\begin{Bmatrix} Z \\ C \end{Bmatrix}$	N	出 力	代数方程式の根
5	WK	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	N + 1	ワーク	作業領域
6	CWK	$\begin{Bmatrix} Z \\ C \end{Bmatrix}$	4 × (N+1)	ワーク	作業領域
7	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) CA(1) ≠ (0, 0)

(b) N ≥ 1

(c) NEV > 0 (既定値にするため, 0 を入力する場合は除く)

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1500	制限条件 (c) を満足しなかった.	既定値にセットして処理を続ける.
2000+i	i + 1 番目以降は最大関数評価回数で収束しなかった.	i + 1 番目以降の根が収束しないままで処理を終了する.
3000	制限条件 (a) または (b) を満足しなかった.	処理を打ち切る.

(6) 注意事項

- (a) 引数 NEV は, 0 を入力すれば既定値がセットされる.
- (b) 重根のときの相対精度は, 根の多重度を n として, $\sqrt[n]{\text{誤差判定のための単位}}$ 程度になる.

(7) 使用例

(a) 問題

$x^{10} - 100\sqrt{-1}x^9 - 17x^6 + 1700\sqrt{-1}x^5 + 16x^2 - 1600\sqrt{-1}x = 0$ を解く.

(b) 入力データ

代数方程式の係数を格納した配列 CA:

CA (1) =(1.0, 0.0), CA (2) =(0.0, -100.0), CA (3) =(0.0, 0.0), CA (4) =(0.0, 0.0), CA (5) =(-17.0, 0.0), CA (6) =(0.0, 1700.0), CA (7) =(0.0, 0.0), CA (8) =(0.0, 0.0), CA (9) =(16.0, 0.0), CA (10) =(0.0, -1600.0), CA (11) =(0.0, 0.0)

N=10, NEV=0

(c) 主プログラム

```

PROGRAM ALACHA
! *** EXAMPLE OF ZLACHA ***
IMPLICIT COMPLEX(8) (A-H,O-V,X-Z)
IMPLICIT REAL(8) (W)
INTEGER N,NEV,IERR,I
DIMENSION CA(11),CX(10),WK(11),CWK(44)
!
  READ(5,*) N
  READ(5,*) NEV
  READ(5,*) (CA(I),I=1,N+1,1)
  WRITE(6,1000)
  WRITE(6,1500) N
  WRITE(6,2000) NEV
  WRITE(6,2500)
  WRITE(6,3000) (CA(I),I=1,N+1,1)
  CALL ZLACHA(CA,N,NEV,CX,WK,CWK,IERR)
  WRITE(6,3500)
  WRITE(6,4000) IERR
  WRITE(6,4500) NEV
  WRITE(6,5000)
  WRITE(6,5500) (CX(I),I=1,N,1)
  STOP
!
1000 FORMAT(' ',/,5X,'*** ZLACHA ***',/,&
7X,'* ALGEBRAIC EQUATION *',/,&
9X,'X**10-100I*(X**9)-17*(X**6)+1700I*(X**5)+16*(X**2)',&
'-1600I*X=0',/,&
6X,'** INPUT **')
1500 FORMAT(9X,'DEGREE OF ALGEBRAIC EQUATION = ',I2)
2000 FORMAT(9X,'MAXIMUM NUMBER OF FUNCTION EVALUATIONS = ',I3)
2500 FORMAT(9X,'COEFFICIENTS OF ALGEBRAIC EQUATION',/,&
9X,'REAL PART',5X,'IMAGINARY PART',/)
3000 FORMAT(9X,F7.1,7X,F7.1)
3500 FORMAT(' ',/.,/.,6X,'** OUTPUT **')
4000 FORMAT(9X,'IERR = ',I4)
4500 FORMAT(9X,'PRACTICAL NUMBER OF FUNCTION EVALUATIONS = ',I3)
5000 FORMAT(9X,'ROOTS',/,&
9X,'REAL PART',13X,'IMAGINARY PART',/)
5500 FORMAT(9X,D17.10,5X,D17.10)
END

```

(d) 出力結果

```

*** ZLACHA ***
* ALGEBRAIC EQUATION *
X**10-100I*(X**9)-17*(X**6)+1700I*(X**5)+16*(X**2)-1600I*X=0
** INPUT **
DEGREE OF ALGEBRAIC EQUATION = 10
MAXIMUM NUMBER OF FUNCTION EVALUATIONS = 0
COEFFICIENTS OF ALGEBRAIC EQUATION
REAL PART      IMAGINARY PART
1.0             0.0
0.0             -100.0
0.0             0.0
0.0             0.0
-17.0          0.0
0.0            1700.0
0.0            0.0
0.0            0.0
16.0           0.0
0.0            -1600.0
0.0            0.0
** OUTPUT **

```

```
IERR =      0
PRACTICAL NUMBER OF FUNCTION EVALUATIONS = 17
ROOTS
REAL PART      IMAGINARY PART
0.000000000D+00  0.100000000D+03
0.000000000D+00 -0.200000000D+01
0.200000000D+01  0.648373437D-18
-0.2869859255D-41  0.200000000D+01
-0.200000000D+01  0.7826070366D-19
0.000000000D+00 -0.100000000D+01
-0.100000000D+01  0.8029504743D-18
0.000000000D+00  0.100000000D+01
0.100000000D+01  0.6074219118D-19
0.000000000D+00  0.000000000D+00
```

4.3 非線形方程式

4.3.1 DLNRDS, RLNRDS

実関数の根 (初期値指定) (導関数定義必要)

(1) 機能

導関数を定義して、非線形方程式の1根を初期値から求める。

(2) 使用法

倍精度サブルーチン:

CALL DLNRDS (F, DF, X, ER, NEV, IERR)

単精度サブルーチン:

CALL RLNRDS (F, DF, X, ER, NEV, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	F	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	—	入 力	方程式を定義する関数副プログラム F(X) の関数名
2	DF	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	—	入 力	導関数を定義する関数副プログラム DF(X) の関数名
3	X	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	根の初期値
				出 力	根
4	ER	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	要求精度 (既定値:(誤差判定のための単位) × 64)
5	NEV	I	2	入 力	NEV (1) :最大関数評価回数 (既定値:100) NEV (2) :最大導関数評価回数 (既定値:100)
				出 力	NEV (1) :実際の関数評価回数 NEV (2) :実際の導関数評価回数
6	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

- (a) $NEV(1) > 0$ (既定値にするため, 0 を入力する場合は除く)
- (b) $NEV(2) > 0$ (既定値にするため, 0 を入力する場合は除く)
- (c) $ER \geq$ 誤差判定のための単位 (既定値にするため, 0.0 を入力する場合は除く)

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1500	制限条件 (a), (b) または (c) を満足しなかつた.	既定値にセットして処理を続ける.
4000	根が求められない.	処理を打ち切る.
5000	最大関数評価回数または最大導関数評価回数に到達しても根が求められない.	その時点での X を返す.

(6) 注意事項

- (a) 引数第 1 項 F, 引数第 2 項 DF の実際の名前は使用者側のプログラムで EXTERNAL 文を用いて宣言し, F, DF のそれぞれの実際の名前の関数副プログラムを作っておかなければならない (詳細は 4.1.1 参照). 関数副プログラム (倍精度) の作り方は次のようになる.

```

REAL(8) FUNCTION F(X)
REAL(8) X
F = ~
RETURN
END
REAL(8) FUNCTION DF(X)
REAL(8) X
DF = ~
RETURN
END

```

- (b) 引数の内容の欄に既定値が記されている場合は, 整数型のときは 0, 実数型のときは 0.0 を入力すれば既定値がセットされる.
- (c) 初期値が根から離れていても大域的収束性を持ち, 問題を解くことができるようになっている.
- (d) 収束判定は $\{| \text{根の更新量} | < ER \times \max(1, | \text{根の値} |)\}$ および $\{| \text{関数値} | < ER + 64 \times (\text{誤差判定のための単位}) \times | \text{根の値} |\}$ が成立したとき収束したものとする.

(7) 使用例

- (a) 問題

$$\begin{cases} f = x^7 + 28x^4 - 483.809683 = 0 \\ f' = 7x^6 + 112x^3 \end{cases}$$

の 1 根を求める.

- (b) 入力データ

関数 $f(x)$ に対応する関数副プログラム名: FLNRDS
 導関数 $f'(x)$ に対応する関数副プログラム名: FLNRDD
 X = -1.0, ER=0.0, NEV (1) =0, NEV (2) =0

- (c) 主プログラム

```

PROGRAM BLNRDS
! *** EXAMPLE OF DLNRDS ***
IMPLICIT REAL(8) (A-H,O-Z)
IMPLICIT INTEGER (I-N)

```

```

EXTERNAL FLNRDS,FLNRD2
DIMENSION NEV(2)
!
  READ(5,*) (NEV(I),I=1,2,1)
  READ(5,*) ER
  READ(5,*) X
  WRITE(6,1000)
  WRITE(6,1500) NEV(1)
  WRITE(6,2000) NEV(2)
  WRITE(6,2500) ER
  WRITE(6,3000)
  WRITE(6,3500) X
  CALL DLNRDS(FLNRDS,FLNRD2,X,ER,NEV,IERR)
  WRITE(6,4000)
  WRITE(6,4500) IERR
  WRITE(6,5000) NEV(1)
  WRITE(6,5500) NEV(2)
  WRITE(6,6000)
  WRITE(6,6500) X
  STOP
!
1000 FORMAT(' ',/,5X,'*** DLNRDS ***',/,&
7X,'* NONLINEAR EQUATION *',/,&
9X,'X**7+28*(X**4)-483.809683=0',/,&
7X,'* DERIVATIVE *',/,&
9X,'7*(X**6)+112*(X**3)',/,&
6X,'** INPUT **')
1500 FORMAT(9X,'MAXIMUM NUMBER OF FUNCTION EVALUATIONS = ',I3)
2000 FORMAT(9X,'MAXIMUM NUMBER OF DERIVATIVE EVALUATIONS = ',I3)
2500 FORMAT(9X,'REQUIRED ACCURACY = ',D7.1)
3000 FORMAT(9X,'INITIAL VALUE OF ROOT')
3500 FORMAT(9X,F4.1)
4000 FORMAT(' ',/,/,6X,'** OUTPUT **')
4500 FORMAT(9X,'IERR = ',I4)
5000 FORMAT(9X,'PRACTICAL NUMBER OF FUNCTION EVALUATIONS = ',I3)
5500 FORMAT(9X,'PRACTICAL NUMBER OF DERIVATIVE EVALUATIONS = ',I3)
6000 FORMAT(9X,'ROOT')
6500 FORMAT(9X,D17.10)
END

REAL(8) FUNCTION FLNRDS(X)
REAL(8) X
!
  FLNRDS = X**7+(28.0D0)*(X**4)-(483.809683D0)
  RETURN
END

REAL(8) FUNCTION FLNRD2(X)
REAL(8) X
!
  FLNRD2 = 7.0D0*(X**6)+112.0D0*(X**3)
  RETURN
END

```

(d) 出力結果

```

*** DLNRDS ***
* NONLINEAR EQUATION *
X**7+28*(X**4)-483.809683=0
* DERIVATIVE *
7*(X**6)+112*(X**3)
** INPUT **
MAXIMUM NUMBER OF FUNCTION EVALUATIONS = 0
MAXIMUM NUMBER OF DERIVATIVE EVALUATIONS = 0
REQUIRED ACCURACY = 0.0D+00
INITIAL VALUE OF ROOT
-1.0

** OUTPUT **
IERR = 0
PRACTICAL NUMBER OF FUNCTION EVALUATIONS = 36
PRACTICAL NUMBER OF DERIVATIVE EVALUATIONS = 1
ROOT
0.1926185353D+01

```

4.3.2 DLNRIS, RLNRIS

実関数の根 (初期値指定) (導関数定義不要)

(1) 機能

導関数が与えられないものとして非線形方程式の1根を初期値から求める。

(2) 使用法

倍精度サブルーチン:

CALL DLNRIS (F, X, ER, NEV, IERR)

単精度サブルーチン:

CALL RLNRIS (F, X, ER, NEV, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	F	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	—	入 力	方程式を定義する関数副プログラム F(X) の関数名
2	X	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	根の初期値
				出 力	根
3	ER	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	要求精度 (既定値:(誤差判定のための単位) × 64)
4	NEV	I	1	入 力	最大関数評価回数 (既定値:100)
				出 力	実際の関数評価回数
5	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NEV > 0$ (既定値にするため, 0 を入力する場合は除く)

(b) $ER \geq$ 誤差判定のための単位 (既定値にするため, 0.0 を入力する場合は除く)

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1500	制限条件 (a) または (b) を満足しなかった.	既定値にセットして処理を続ける.
4000	根が求められない.	処理を打ち切る.
5000	最大関数評価回数に到達しても根が求められない.	その時点での X を返す.

(6) 注意事項

- (a) 引数第 1 項 F の実際の名前は使用者側のプログラムで EXTERNAL 文を用いて宣言し, F の実際の名前の関数副プログラムを作っておかなければならない(詳細は 4.1.1 参照). 関数副プログラム (倍精度) の作り方は次のようにする.

```
REAL(8) FUNCTION F(X)
REAL(8) X
F = ~
RETURN
END
```

- (b) 引数の内容の欄に既定値が記されている場合は, 整数型のときは 0, 実数型のときは 0.0 を入力すれば既定値がセットされる.
- (c) 初期値が根から離れていても大域性収束性を持ち, 問題を解くことができるようになっている.
- (d) 収束判定は $\{ | \text{根の更新量} | < \text{ER} \times \max(1, | \text{根の値} |) \}$ および $\{ | \text{関数値} | < \text{ER} + 64 \times (\text{誤差判定のための単位}) \times | \text{根の値} | \}$ が成立したとき収束したものとす.

(7) 使用例

- (a) 問題

$$f = \exp(0.01x) + 3 - (x - 231)(x - 597) = 0 \text{ の } 1 \text{ 根を求める.}$$

- (b) 入力データ

関数 $f(x)$ に対応する関数副プログラム名: FLNRIS

X=0.0, ER=0.0, NEV=0

- (c) 主プログラム

```
PROGRAM BLNRIS
! *** EXAMPLE OF DLNRIS ***
IMPLICIT REAL(8) (A-H,O-Z)
EXTERNAL FLNRIS
INTEGER NEV,IERR
!
READ(5,*) NEV
READ(5,*) ER
READ(5,*) X
WRITE(6,1000)
WRITE(6,1500) NEV
WRITE(6,2000) ER
WRITE(6,2500)
WRITE(6,3000) X
CALL DLNRIS(FLNRIS,X,ER,NEV,IERR)
WRITE(6,3500)
WRITE(6,4000) IERR
WRITE(6,4500) NEV
WRITE(6,5000)
WRITE(6,5500) X
STOP
!
1000 FORMAT(' ',/,5X,'*** DLNRIS ***',/,&
7X,'* NONLINEAR EQUATION *',/,&
9X,'EXP(0.01*X)+3-(X-231)*(X-597)=0',/,&
6X,'** INPUT **')
1500 FORMAT(9X,'MAXIMUM NUMBER OF FUNCTION EVALUATIONS = ',I3)
2000 FORMAT(9X,'REQUIRED ACCURACY = ',D7.1)
2500 FORMAT(9X,'INITIAL VALUE OF ROOT')
3000 FORMAT(9X,F3.1)
3500 FORMAT(' ',/,/,6X,'** OUTPUT **')
4000 FORMAT(9X,'IERR = ',I4)
4500 FORMAT(9X,'PRACTICAL NUMBER OF FUNCTION EVALUATIONS = ',I3)
5000 FORMAT(9X,'ROOT')
5500 FORMAT(9X,D17.10)
END

REAL(8) FUNCTION FLNRIS(X)
REAL(8) X
!
FLNRIS = EXP((0.01D0)*X)+(3.0D0)-(X-(231.0D0))*(X-(597.0D0))
RETURN
END
```

(d) 出力結果

```
*** DLNRIS ***
* NONLINEAR EQUATION *
EXP(0.01*X)+3-(X-231)*(X-597)=0
** INPUT **
MAXIMUM NUMBER OF FUNCTION EVALUATIONS = 0
REQUIRED ACCURACY = 0.0D+00
INITIAL VALUE OF ROOT
0.0

** OUTPUT **
IERR = 0
PRACTICAL NUMBER OF FUNCTION EVALUATIONS = 10
ROOT
0.2309642908D+03
```

4.3.3 DLNRSS, RLNRSS

実関数の根 (区間指定) (導関数定義不要)

(1) 機能

非線形方程式の関数値符号が変わる区間内の 1 根を求める。

(2) 使用法

倍精度サブルーチン:

CALL DLNRSS (F, AX, BX, ER, X, IERR)

単精度サブルーチン:

CALL RLNRSS (F, AX, BX, ER, X, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	F	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	—	入 力	方程式を定義する関数副プログラム F(X) の関数名
2	AX	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	根をはさむ区間の左端
3	BX	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	根をはさむ区間の右端
4	ER	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	要求精度 (既定値:(誤差判定のための単位) × 64)
5	X	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	根
6	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $ER \geq$ 誤差判定のための単位 (既定値にするため, 0.0 を入力する場合は除く)

(b) $AX \neq BX$

(c) $F(AX) \times F(BX) \leq 0.0$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1500	制限条件 (a) を満足しなかった.	既定値にセットして処理を続ける.
3000	制限条件 (b) または (c) を満足しなかった.	処理を打ち切る.

(6) 注意事項

- (a) 引数第 1 項 F の実際の名前は使用者側のプログラムで EXTERNAL 文を用いて宣言し、F の実際の名前の関数副プログラムを作っておかなければならない(詳細は 4.1.1 参照). 関数副プログラム (倍精度) の作り方は次のようにする.

```
REAL(8) FUNCTION F(X)
REAL(8) X]
F = ~
RETURN
END
```

- (b) 引数 ER は、0.0 を入力すれば既定値がセットされる。
(c) AX が区間の右端、BX が区間の左端のときでも、問題が解ける。
(d) 収束判定は、
{| 根をはさむ区間 | ≤ ER + 2 × (誤差判定のための単位) × | 根の値 | が成立したとき収束したものとする。

(7) 使用例

- (a) 問題

$f = \exp(0.01x) + 3 - (x - 231)(x - 597) = 0$ の 1 根を求める。

- (b) 入力データ

関数 $f(x)$ に対応する関数副プログラム名: FLNRSS

AX = -200.0, BX=240.0, ER=0.0

- (c) 主プログラム

```
PROGRAM BLNRSS
! *** EXAMPLE OF DLNRSS ***
IMPLICIT REAL(8) (A-H,O-Z)
EXTERNAL FLNRSS
INTEGER IERR
!
READ(5,*) ER
READ(5,*) AX
READ(5,*) BX
WRITE(6,1000)
WRITE(6,2000) ER
WRITE(6,3000)
WRITE(6,4000) AX,BX
CALL DLNRSS(FLNRSS,AX,BX,ER,X,IERR)
WRITE(6,5000)
WRITE(6,6000) IERR
WRITE(6,7000)
WRITE(6,8000) X
STOP
!
1000 FORMAT(' ',/,5X,'*** DLNRSS ***',/,&
7X,'* NONLINEAR EQUATION *',/,&
9X,'EXP(0.01*X)+3-(X-231)*(X-597)=0',/,&
6X,'** INPUT **')
2000 FORMAT(9X,'REQUIRED ACCURACY = ',D7.1)
3000 FORMAT(9X,'INTERVAL IN WHICH ROOTS EXIST')
4000 FORMAT(9X,'(',F6.1,2X,F6.1,')')
5000 FORMAT(' ',/,/,6X,'** OUTPUT **')
6000 FORMAT(9X,'IERR = ',I4)
7000 FORMAT(9X,'ROOT:')
8000 FORMAT(9X,D17.10)
END

REAL(8) FUNCTION FLNRSS(X)
REAL(8) X
!
FLNRSS = EXP((0.01D0)*X)+(3.0D0)-(X-(231.0D0))*(X-(597.0D0))
RETURN
END
```

- (d) 出力結果

```
*** DLNRSS ***
* NONLINEAR EQUATION *
EXP(0.01*X)+3-(X-231)*(X-597)=0
** INPUT **
```

```
REQUIRED ACCURACY = 0.0D+00  
INTERVAL IN WHICH ROOTS EXIST  
(-200.0 240.0)
```

```
** OUTPUT **  
IERR = 0  
ROOT  
0.2309642908D+03
```


4.3.4 DLNRSA, RLNRSA

実関数の全根 (区間指定) (導関数定義不要)

(1) 機能

非線形方程式の区間内の全根を求める。

(2) 使用法

倍精度サブルーチン:

CALL DLNRSA (F, AX, BX, ER, NEV, X, M, IERR)

単精度サブルーチン:

CALL RLNRSA (F, AX, BX, ER, NEV, X, M, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	F	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	—	入 力	方程式を定義する関数副プログラム F(X) の関数名
2	AX	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	根をはさむ区間の左端
3	BX	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	根をはさむ区間の右端
4	ER	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	要求精度 (既定値:(誤差判定のための単位) × 64)
5	NEV	I	1	入 力	最大関数評価回数 (既定値:100)
				出 力	実際の関数評価回数
6	X	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	M	出 力	根
7	M	I	1	入 力	求める根の最大数
				出 力	求められた根の数
8	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

- (a) $ER \geq$ 誤差判定のための単位 (既定値にするため, 0.0 を入力する場合は除く)
- (b) $NEV > 0$ (既定値にするため, 0 を入力する場合は除く)
- (c) $M > 0$
- (d) $AX \neq BX$
- (e) $AX < BX$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1200	制限条件 (e) を満足しなかった.	AX と BX を交換して処理を続ける.
1500	制限条件 (a) または (b) を満足しなかった.	既定値にセットして処理を続ける.
2000	関数評価が最大関数評価回数に達した.	求めた M 個の根までを出力し, 処理を打ち切る.
2500	根の数が根の最大数 M をこえる.	
3000	制限条件 (c) または (d) を満足しなかった.	処理を打ち切る.
4000	区間内に根が存在しない.	

(6) 注意事項

- (a) 引数第 1 項 F の実際の名前は使用者側のプログラムで EXTERNAL 文を用いて宣言し, F の実際の名前の関数副プログラムを作っておかなければならない (詳細は 4.1.1 参照). 関数副プログラムの作り方は次のようにする.

```
REAL(8) FUNCTION F(X)
REAL(8) X
F = ~
RETURN
END
```

- (b) 引数の内容の欄に既定値が記されている場合は, 整数型のときは 0, 実数型のときは 0.0 を入力すれば既定値がセットされる.
- (c) 収束判定は
 $\{ | \text{根の更新量} | < \text{ER} \times \max(1, | \text{根の値} |) \}$ および $\{ | \text{関数値} | < \text{ER} + 64 \times (\text{誤差判定のための単位}) \times | \text{根の値} | \}$ が成立したとき収束したものとする.
- (d) $\max(2 \times (\text{要求精度}), (|A| + |B|) \times (\text{誤差判定のための単位}), \sqrt[3]{\text{誤差判定のための単位}})$ よりも近接した 2 根は, 根の分離ができないのでどちらか一方の根が求められないことになる.

(7) 使用例

(a) 問題

$f = \exp(0.01x) + 3 - (x - 231)(x - 597) = 0$ の全根を求める.

(b) 入力データ

関数 $f(x)$ に対応する関数副プログラム名: FLNRSA

AX = -200.0, BX=800.0, ER=0.0, NEV=0, M=15

(c) 主プログラム

```
PROGRAM BLNRSA
! *** EXAMPLE OF DLNRSA ***
IMPLICIT REAL(8) (A-H,O-Z)
EXTERNAL FLNRSA
INTEGER NEV,M,IERR,I
DIMENSION X(10)
!
READ(5,*) NEV
READ(5,*) ER
READ(5,*) M
READ(5,*) AX
READ(5,*) BX
WRITE(6,1000)
```

```

WRITE(6,1500) NEV
WRITE(6,2000) ER
WRITE(6,2500) M
WRITE(6,3000)
WRITE(6,3500) AX,BX
CALL DLNRSA(FLNRSA,AX,BX,ER,NEV,X,M,IERR)
WRITE(6,4000)
WRITE(6,4500) IERR
WRITE(6,5000) NEV
WRITE(6,5500) M
WRITE(6,6000)
WRITE(6,6500) (X(I),I=1,M,1)
STOP
!
1000 FORMAT(' ',/,/,5X,'*** DLNRSA ***',/,&
7X,'* NONLINEAR EQUATION *',/,&
9X,'EXP(0.01*X)+3-(X-231)*(X-597)=0',/,&
6X,'** INPUT **')
1500 FORMAT(9X,'MAXIMUM NUMBER OF FUNCTION EVALUATIONS = ',I3)
2000 FORMAT(9X,'REQUIRED ACCURACY = ',D7.1)
2500 FORMAT(9X,'MAXIMUM NUMBER OF ROOTS = ',I2)
3000 FORMAT(9X,'INTERVAL IN WHICH ROOTS EXIST')
3500 FORMAT(9X,'(',F6.1,2X,F6.1,')')
4000 FORMAT(' ',/,/,6X,'** OUTPUT **')
4500 FORMAT(9X,'IERR = ',I4)
5000 FORMAT(9X,'PRACTICAL NUMBER OF FUNCTION EVALUATIONS = ',I3)
5500 FORMAT(9X,'NUMBER OF ROOTS = ',I2)
6000 FORMAT(9X,'ROOTS',/)
6500 FORMAT(9X,D17.10)
END

REAL(8) FUNCTION FLNRSA(X)
REAL(8) X
!
FLNRSA = EXP((0.01D0)*X)+(3.0D0)-(X-(231.0D0))*(X-(597.0D0))
RETURN
END

```

(d) 出力結果

```

*** DLNRSA ***
* NONLINEAR EQUATION *
EXP(0.01*X)+3-(X-231)*(X-597)=0
** INPUT **
MAXIMUM NUMBER OF FUNCTION EVALUATIONS = 0
REQUIRED ACCURACY = 0.0D+00
MAXIMUM NUMBER OF ROOTS = 15
INTERVAL IN WHICH ROOTS EXIST
(-200.0 800.0)

** OUTPUT **
IERR = 0
PRACTICAL NUMBER OF FUNCTION EVALUATIONS = 98
NUMBER OF ROOTS = 2
ROOTS
0.2309642908D+03
0.5980863437D+03

```

4.3.5 ZLNCIS, CLNCIS

複素関数の根 (初期値指定) (導関数定義不要)

(1) 機能

導関数が与えられないものとして複素型の非線形方程式の 1 根を初期値から求める。

(2) 使用法

倍精度サブルーチン:

CALL ZLNCIS (F, CX, ER, NEV, IERR)

単精度サブルーチン:

CALL CLNCIS (F, CX, ER, NEV, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	F	$\begin{Bmatrix} Z \\ C \end{Bmatrix}$	—	入 力	方程式を定義する複素関数副プログラム F(X) の関数名
2	CX	$\begin{Bmatrix} Z \\ C \end{Bmatrix}$	1	入 力	根の初期値
				出 力	根
3	ER	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	要求精度 (既定値:誤差判定のための単位) × 64)
4	NEV	I	1	入 力	最大関数評価回数 (既定値:100)
				出 力	実際の関数評価回数
5	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $NEV > 0$ (既定値にするため, 0 を入力する場合は除く)

(b) $ER \geq$ 誤差判定のための単位 (既定値にするため, 0.0 を入力する場合は除く)

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1500	制限条件 (a) または (b) を満足しなかった.	既定値にセットして処理を続ける.
4000	ゼロ除算エラーが発生した.	処理を打ち切る.
5000	最大関数評価回数に到達しても根が求められない.	その時点での X を返す.

(6) 注意事項

- (a) 引数第 1 項 F の実際の名前は使用者側のプログラムで EXTERNAL 文を用いて宣言し, F の実際の名前の関数副プログラムを作っておかなければならない(詳細は 4.1.1 参照). 関数副プログラム (倍精度) の作り方は次のようにする.

```

COMPLEX(8) FUNCTION F(X)
COMPLEX(8) X
F = ~
RETURN
END

```

- (b) 引数の内容の欄に既定値が記されている場合は, 整数型のときは 0, 実数型のときは 0.0 を入力すれば既定値がセットされる.

- (c) 収束判定は,

{ | 根の更新量 | < ER × max(1, | 根の値 |)} および { | 関数値 | < ER + 64 × (誤差判定のための単位) × | 根の値 |} が成立したとき収束したものとする.

(7) 使用例

- (a) 問題

$f = x^{10} - 1 = 0$ の 1 根を求める.

- (b) 入力データ

関数 $f(x)$ に対応する関数副プログラム名: FLNCIS

CX = (1.0, -1.0), ER = 1.0D - 10, NEV=100

- (c) 主プログラム

```

PROGRAM ALNCIS
! *** EXAMPLE OF ZLNCIS ***
IMPLICIT COMPLEX(8) (A-D,F-H,O-Z)
EXTERNAL FLNCIS
INTEGER NEV,IERR
REAL(8) ER
!
READ(5,*) NEV
READ(5,*) ER
READ(5,*) CX
WRITE(6,1000)
WRITE(6,1500) NEV
WRITE(6,2000) ER
WRITE(6,2500)
WRITE(6,3000) CX
CALL ZLNCIS(FLNCIS,CX,ER,NEV,IERR)
WRITE(6,3500)
WRITE(6,4000) IERR
WRITE(6,4500) NEV
WRITE(6,5000)
WRITE(6,5500) CX
STOP
!
1000 FORMAT(' ',/,/,5X,'*** ZLNCIS ***',/,&
7X,'* NONLINEAR EQUATION *',/,&
9X,'Z**10-1=0',/,&
6X,'** INPUT **')
1500 FORMAT(9X,'MAXIMUM NUMBER OF FUNCTION EVALUATIONS = ',I3)
2000 FORMAT(9X,'REQUIRED ACCURACY = ',D7.1)
2500 FORMAT(9X,'INITIAL VALUE OF ROOT',/,&
9X,'REAL PART',5X,'IMAGINARY PART')
3000 FORMAT(9X,F3.1,11X,F3.1)
3500 FORMAT(' ',/,/,9X,'** OUTPUT **')
4000 FORMAT(9X,'IERR = ',I4)
4500 FORMAT(9X,'PRACTICAL NUMBER OF FUNCTION EVALUATIONS = ',I3)
5000 FORMAT(9X,'ROOT',/,&
9X,'REAL PART',13X,'IMAGINARY PART')
5500 FORMAT(9X,D17.10,5X,D17.10)
END

COMPLEX(8) FUNCTION FLNCIS(X)
!
COMPLEX(8) X
!

```

```
FLNCIS = X**10-(1.0D0,0.0D0)
RETURN
END
```

(d) 出力結果

```
*** ZLNCIS ***
* NONLINEAR EQUATION *
Z**10-1=0
** INPUT **
MAXIMUM NUMBER OF FUNCTION EVALUATIONS = 100
REQUIRED ACCURACY = 0.1D-09
INITIAL VALUE OF ROOT
REAL PART      IMAGINARY PART
1.0            1.0

** OUTPUT **
IERR = 0
PRACTICAL NUMBER OF FUNCTION EVALUATIONS = 11
ROOT
REAL PART      IMAGINARY PART
0.8090169944D+00  0.5877852523D+00
```

4.4 連立非線形方程式

4.4.1 DLSRDS, RLSRDS

連立非線形方程式の根 (ヤコビ行列定義任意)

(1) 機能

ヤコビ行列を定義する場合および定義しない場合のどちらにも対応して連立非線形方程式の1根を求める。

(2) 使用法

倍精度サブルーチン:

CALL DLSRDS (SUB, SUBJ, X, N, ER, NEV, ISW, IWK, WK, DWK, IERR)

単精度サブルーチン:

CALL RLSRDS (SUB, SUBJ, X, N, ER, NEV, ISW, IWK, WK, DWK, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	SUB	-	-	入 力	N 個の非線形方程式を定義する サブルーチン SUB(X, N, F) の手続き名
2	SUBJ	-	-	入 力	ヤコビ行列を定義する サブルーチン SUBJ(X, N, A) の手続き名
3	X	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	N	入 力	根の初期値
				出 力	根
4	N	I	1	入 力	非線形方程式の連立数
5	ER	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	要求精度 (既定値:(誤差判定のための単位)×64)
6	NEV	I	2	入 力	NEV (1) :最大関数評価回数 (既定値: 100 × N) NEV (2) :最大ヤコビ行列評価回数 (既定値: 100 × N)
				出 力	NEV (1) :実際の関数評価回数 NEV (2) :実際のヤコビ行列評価回数
7	ISW	I	1	入 力	入力スイッチ ISW = 0:ヤコビ行列自動計算 ISW ≠ 0:使用者側の定義したヤコビ行列作成用サブルーチンを使う。(注意事項 (b) 参照)
8	IWK	I	N	ワーク	作業領域
9	WK	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	N×(N+3)	ワーク	作業領域
10	DWK	D	内容参照	ワーク	作業領域. この変数の型は, 倍精度. 大きさ: (N × (2 × N + 2))
11	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

- (a) NEV (1) > 0 (既定値にセットするため, 0 を入力する場合は除く)
- (b) NEV (2) > 0 (既定値にセットするため, 0 を入力する場合は除く)
- (c) ER ≥ 誤差判定のための単位 (既定値にセットするため, 0.0 を入力する場合は除く)
- (d) N > 0

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1500	制限条件 (a) または (b) または (c) を満足しなかった.	既定値にセットして処理を続ける.
3000	制限条件 (d) を満足しなかった.	処理を打ち切る.
4000	連立 1 次方程式を解くときにエラーが発生した.	
4500	非線形性が強すぎての解の更新ができなくなった.	
5000	最大関数評価回数または最大ヤコビ行列評価回数に到達したが, 解が得られなかった.	その時点での X を返す.

(6) 注意事項

- (a) 引数第 1 項 SUB, 引数第 2 項 SUBJ の実際の名前は使用者側のプログラムで EXTERNAL 文を用いて宣言し, SUB, SUBJ のそれぞれの実際の名前のサブルーチンを作っておかなければならない (詳細は 4.1.1 参照).

サブルーチンの作り方は次に示すとおりである.

連立非線形方程式を

$$\begin{cases} f_1(x_1, \dots, x_N) = 0 \\ \vdots \\ f_N(x_1, \dots, x_N) = 0 \end{cases}$$

とすると, サブルーチン SUB は単精度として

```
SUBROUTINE SUB(X, N, F)
```

```
REAL X(N), F(N)
```

```
F(1) = f1(x1, ..., xN)
```

```
⋮
```

```
F(N) = fN(x1, ..., xN)
```

```
RETURN
```

```
END
```

ヤコビ行列 A は

$$A = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \dots & \dots & \frac{\partial f_1}{\partial x_N} \\ \vdots & & & \vdots \\ \frac{\partial f_N}{\partial x_1} & \dots & \dots & \frac{\partial f_N}{\partial x_N} \end{bmatrix} = \begin{bmatrix} A(1,1), & \dots, & A(1,N) \\ \vdots & & \vdots \\ \vdots & & \vdots \\ A(N,1), & \dots, & A(N,N) \end{bmatrix}$$

よってサブルーチン SUBJ は, 単精度として


```

SUBROUTINE SUBJ(X, N, A)
REAL X(N), A(N, *)
A(1,1) = ∂f1/∂x1
  ⋮
A(N,1) = ∂fN/∂x1
  ⋮
A(1,N) = ∂f1/∂xN
  ⋮
A(N,N) = ∂fN/∂xN
RETURN
END

```

として記述される.

例

$$\begin{cases} x_1 + x_2 + x_3 = 0 \\ x_1x_2 + x_2x_3 + x_3x_1 = 0 \\ x_1x_2x_3 = 0 \end{cases}$$

$$A = \begin{bmatrix} 1, & 1, & 1 \\ x_2 + x_3, & x_1 + x_3, & x_1 + x_2 \\ x_2x_3, & x_1x_3, & x_1x_2 \end{bmatrix}$$

したがって各サブルーチンは次のように記述される.

```

SUBROUTINE SUB(X, N, F)
REAL X(N), F(N)
F(1) = X(1) + X(2) + X(3)
F(2) = X(1) * X(2) + X(2) * X(3) + X(3) * X(1)
F(3) = X(1) * X(2) * X(3)
RETURN
END

```

```

SUBROUTINE SUBJ(X, N, A)
REAL X(N), A(N, *)
A(1,1) = 1.0
A(1,2) = 1.0
A(1,3) = 1.0
A(2,1) = X(2) + X(3)
A(2,2) = X(1) + X(3)
A(2,3) = X(1) + X(2)
A(3,1) = X(2) * X(3)
A(3,2) = X(1) * X(3)
A(3,3) = X(1) * X(2)
RETURN
END

```

- (b) ISW=0 のときは, SUBJ という引数はダミーであり, サブルーチンを作る必要はない. また, NEV (2) も入力不要である. ISW \neq 0 のときは, SUBJ の実際の名前のサブルーチンを作る必要がある. また, NEV (2) も入力が必要である.
- (c) 引数の内容の欄に既定値が記されている場合は, 整数型のときは 0, 実数型のときは 0.0 を入力すれば既定値がセットされる.
- (d) 収束判定はすべての根またはすべての関数値に対し, $\{\| \text{根の更新量} \|_{\infty} < ER \times \max(1, \| \text{根の値} \|_{\infty})\}$ および $\{\| \text{関数値} \|_{\infty} < ER + 64 \times \varepsilon \times \| \text{根の値} \|_{\infty}\}$ が成立したとき収束したものとする.

(7) 使用例

(a) 問題

$$\begin{cases} 10(x_2 - x_1^2) = 0 \\ 1 - x_1 = 0 \end{cases} \text{ を解く}$$

(b) 入力データ

非線形方程式に対応した関数 $f(x)$ を定義するサブルーチン名: FLSRDS

X(1) = -1.2, X (2) = 1.0, N = 2, ER = 0.0, NEV (1) = 200, NEV (2) = 200, ISW = 0

(c) 主プログラム

```

PROGRAM BLSRDS
! *** EXAMPLE OF DLSRDS ***
IMPLICIT REAL(8) (A-H,O-Z)
IMPLICIT INTEGER (I-N)
EXTERNAL FLSRDS
DIMENSION NEV(2),IWK(2),X(2),WK(10),DWK(12)
!
  READ(5,*) (NEV(I),I=1,2,1)
  READ(5,*) N
  READ(5,*) ER
  READ(5,*) (X(I),I=1,N,1)
  READ(5,*) ISW
  WRITE(6,1000)
  WRITE(6,1500) NEV(1)
  WRITE(6,2000) NEV(2)
  WRITE(6,2500) N
  WRITE(6,3000) ER
  WRITE(6,3500)
  WRITE(6,4000) (X(I),I=1,N,1)
  CALL DLSRDS(FLSRDS,NONSUB,X,N,ER,NEV,ISW,IWK,WK,DWK,IERR)
  WRITE(6,4500)
  WRITE(6,5000) IERR
  WRITE(6,5500) NEV(1)
  WRITE(6,6000) NEV(2)
  WRITE(6,7000)
  WRITE(6,7500) (X(I),I=1,N,1)
  STOP
!
1000 FORMAT(' ',/,5X,'*** DLSRDS ***',/,&
7X,'* SYSTEM OF NONLINEAR EQUATIONS *',/,&
9X,'10*(X2-X1*X1)=0',/,&
9X,'1-X1 =0',/,&
6X,'** INPUT **')
1500 FORMAT(9X,'MAXIMUM NUMBER OF FUNCTIONS EVALUATIONS = ',I3)
2000 FORMAT(9X,'MAXIMUM NUMBER OF JACOBIAN EVALUATIONS = ',I3)
2500 FORMAT(9X,'NUMBER OF NONLINEAR EQUATIONS = ',I2)
3000 FORMAT(9X,'REQUIRED ACCURACY = ',D7.1)
3500 FORMAT(9X,'INITIAL VALUE OF ROOT',/)
4000 FORMAT(9X,F4.1)
4500 FORMAT(' ',/,/,6X,'** OUTPUT **')
5000 FORMAT(9X,'IERR = ',I4)
5500 FORMAT(9X,'PRACTICAL NUMBER OF FUNCTIONS EVALUATIONS = ',I3)
6000 FORMAT(9X,'PRACTICAL NUMBER OF JACOBIAN EVALUATIONS = ',I3)
7000 FORMAT(9X,'ROOT',/)
7500 FORMAT(9X,D17.10)
END

SUBROUTINE FLSRDS(X,N,F)
INTEGER N
REAL(8) X,F
DIMENSION X(N),F(N)
!
  F(1) = 10.0D0*(X(2)-X(1)*X(1))
  F(2) = 1.0D0-X(1)
  RETURN
END

```

(d) 出力結果

```

*** DLSRDS ***
* SYSTEM OF NONLINEAR EQUATIONS *

```

```
10*(X2-X1*X1)=0
1-X1      =0
** INPUT **
MAXIMUM NUMBER OF FUNCTIONS EVALUATIONS = 200
MAXIMUM NUMBER OF JACOBIAN EVALUATIONS = 200
NUMBER OF NONLINEAR EQUATIONS = 2
REQUIRED ACCURACY = 0.0D+00
INITIAL VALUE OF ROOT
-1.2
 1.0

** OUTPUT **
IERR = 0
PRACTICAL NUMBER OF FUNCTIONS EVALUATIONS = 66
PRACTICAL NUMBER OF JACOBIAN EVALUATIONS = 0
ROOT
0.1000000000D+01
0.1000000000D+01
```

4.4.2 DLSRIS, RLSRIS

連立非線形方程式の根 (ヤコビ行列定義不要)

(1) 機能

ヤコビ行列を定義しないものとして連立非線形方程式の1根を求める。

(2) 使用法

倍精度サブルーチン:

CALL DLSRIS (SUB, X, N, ER, NEV, IWK, WK, IERR)

単精度サブルーチン:

CALL RLSRIS (SUB, X, N, ER, NEV, IWK, WK, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	SUB	-	-	入 力	N 個の非線形方程式を定義するサブルーチン SUB(X, N, F) の手続き名
2	X	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	N	入 力	根の初期値
				出 力	根
3	N	I	1	入 力	連立非線形方程式の連立数
4	ER	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	要求精度 (既定値:(誤差判定のための単位) × 64)
5	NEV	I	1	入 力	最大関数評価回数 (既定値: 100 × N)
				出 力	実際の関数評価回数
6	IWK	I	2 × N	ワーク	作業領域
7	WK	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	内容参照	ワーク	作業領域 大きさ: (N × (3 × N + 7))
8	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

- (a) $NEV > 0$ (既定値にするため, 0 を入力する場合は除く)
- (b) $ER \geq$ 誤差判定のための単位 (既定値にするため, 0.0 を入力する場合は除く)
- (c) $N > 0$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1500	制限条件 (a) または (b) を満足しなかった.	既定値にセットして処理を続ける.
3000	制限条件 (c) を満足しなかった.	処理を打ち切る.
4000+i	ヤコビ行列の行列式が 0 である. すなわち, i は最初に 0 となるヤコビ行列の (i, i) 成分である.	
4500	ヤコビ行列の逆行列またはステップサイズの計算時にゼロ除算エラーが発生した.	
5000	最大関数評価回数に到達したが解が得られなかった.	その時点での X を返す.

(6) 注意事項

- (a) 引数第 1 項 SUB の実際の名前は使用者側のプログラムで EXTERNAL 文を用いて宣言し, SUB の実際の名前のサブルーチンを作っておかなければならない (詳細は 4.1.1 参照).

サブルーチンの作り方は, 次に示すとおりである.

連立非線形方程式を

$$\begin{cases} f_1(x_1, \dots, x_N) = 0 \\ \vdots \\ f_N(x_1, \dots, x_N) = 0 \end{cases}$$

とすると, サブルーチン SUB は単精度として

```
SUBROUTINE SUB(X, N, F)
```

```
REAL X(N), F(N)
```

```
F(1) = f1(x1, ..., xN)
```

```
⋮
```

```
F(N) = fN(x1, ..., xN)
```

```
RETURN
```

```
END
```

- (b) 引数の内容の欄に既定値が記されている場合は, 整数型のときは 0, 実数型のときは 0.0 を入力すれば既定値がセットされる.
- (c) 連立非線形方程式の各式をスケールリングしなくても解けるが, 効率を考えるとスケールリングした方がよい.
- (d) 収束判定はすべての根またはすべての関数値に対し, $\{\| \text{根の更新量} \|_{\infty} < ER \times \max(1, \| \text{根の値} \|_{\infty})\}$ および $\{\| \text{関数値} \|_{\infty} < ER + 64 \times \varepsilon \times \| \text{根の値} \|_{\infty}\}$ が成立したとき収束したものとする.

(7) 使用例

- (a) 問題

$$\begin{cases} 10(x_2 - x_1^2) = 0 \\ 1 - x_1 = 0 \end{cases}$$

を解く

- (b) 入力データ

非線形方程式に対応した関数 $f(x)$ を定義するサブルーチン名: FLSRIS

X(1) = -1.2, X(2) = 1.0, N=2, ER=0.0, NEV=200

(c) 主プログラム

```

PROGRAM BLSRIS
! *** EXAMPLE OF DLSRIS ***
  IMPLICIT REAL(8) (A-H,O-Z)
  IMPLICIT INTEGER (I-N)
  EXTERNAL FLSRIS
  DIMENSION IWK(4),X(2),WK(26)
!
  READ(5,*) NEV
  READ(5,*) N
  READ(5,*) ER
  READ(5,*) (X(I),I=1,N,1)
  WRITE(6,1000)
  WRITE(6,1500) NEV
  WRITE(6,2000) N
  WRITE(6,2500) ER
  WRITE(6,3000)
  WRITE(6,3500) (X(I),I=1,N,1)
  CALL DLSRIS(FLSRIS,X,N,ER,NEV,IWK,WK,IERR)
  WRITE(6,4000)
  WRITE(6,4500) IERR
  WRITE(6,5000) NEV
  WRITE(6,5500)
  WRITE(6,6000) (X(I),I=1,N,1)
  STOP
!
1000 FORMAT(' ',/,5X,'*** DLSRIS ***',/,&
7X,'* SYSTEM OF NONLINEAR EQUATIONS *',/,&
9X,'10*(X2-X1*X1)=0',/,&
9X,'1-X1 =0',/,&
6X,'** INPUT **')
1500 FORMAT(9X,'MAXIMUM NUMBER OF FUNCTIONS EVALUATIONS = ',I3)
2000 FORMAT(9X,'NUMBER OF NONLINEAR EQUATIONS = ',I2)
2500 FORMAT(9X,'REQUIRED ACCURACY = ',D7.1)
3000 FORMAT(9X,'INITIAL VALUE OF ROOT',/)
3500 FORMAT(9X,F4.1)
4000 FORMAT(' ',/.,/.,6X,'** OUTPUT **')
4500 FORMAT(9X,'IERR = ',I4)
5000 FORMAT(9X,'PRACTICAL NUMBER OF FUNCTIONS EVALUATIONS = ',I3)
5500 FORMAT(9X,'ROOT',/)
6000 FORMAT(9X,D17.10)
  END

  SUBROUTINE FLSRIS(X,N,F)
  INTEGER N
  REAL(8) X,F
  DIMENSION X(N),F(N)
!
  F(1) = 10.0D0*(X(2)-X(1)*X(1))
  F(2) = 1.0D0-X(1)
  RETURN
  END

```

(d) 出力結果

```

*** DLSRIS ***
* SYSTEM OF NONLINEAR EQUATIONS *
10*(X2-X1*X1)=0
1-X1 =0
** INPUT **
MAXIMUM NUMBER OF FUNCTIONS EVALUATIONS = 200
NUMBER OF NONLINEAR EQUATIONS = 2
REQUIRED ACCURACY = 0.0D+00
INITIAL VALUE OF ROOT

-1.2
1.0

** OUTPUT **
IERR = 0
PRACTICAL NUMBER OF FUNCTIONS EVALUATIONS = 34
ROOT
0.1000000000D+01
0.1000000000D+01

```

第 5 章 極値問題・最適化

5.1 概要

本章では、制約なし 1 変数関数の極小化、制約なし多変数関数の極小化、制約なし関数二乗和の極小化、制約付き 1 変数関数の極小化、制約付き多変数線形関数の最小化（線形計画、混合 0-1 計画、最小費用流問題、日程計画問題、輸送問題）、多変数 2 次関数の最小化（2 次計画）、制約付き多変数関数の最小化（非線形計画）、ネットワーク上の距離の最短化（最短路問題）を行う、サブルーチンについて説明する。

制約なし多変数関数の極小化では、以下のサブルーチンが用意されている。

(i) 1 変数関数の極小化

このサブルーチンは、与えられた初期点から出発して、1 変数関数の極小値探索を行う。多変数関数の極小化では、以下のサブルーチンが用意されている。

(i) 多変数関数の極小化（導関数定義不要）

(ii) 多変数関数の極小化（導関数定義必要）

これらのサブルーチンは、与えられた初期点から出発して、多変数関数の極小値探索を行う。利用者が関数の勾配ベクトルを計算するサブルーチンを用意できないときには、導関数定義不要のサブルーチンを利用することができる。関数の勾配ベクトルを計算するサブルーチンを用意できるときには、導関数定義必要のサブルーチンを利用した方がよい結果が得られる。

制約なし関数二乗和の極小化では、以下のサブルーチンが用意されている。

(i) 非線形最小二乗法（導関数定義不要）

このサブルーチンは、与えられた初期点から出発して、 m 個の多変数関数の二乗和の極小値探索を行う。利用者が関数のヤコビ行列を計算するサブルーチンを用意する必要はない。

制約付き 1 変数関数の極小化では、以下のサブルーチンが用意されている。

(i) 1 変数関数の極小化（区間指定）

このサブルーチンは、与えられた区間の中で、1 変数関数の極小値探索を行う。

制約付き多変数線形関数の最小化では、以下のサブルーチンが用意されている。

(i) 制約付き多変数線形関数の最小化（線形制約）

(ii) 制約付き多変数線形関数の最小化（実不規則スパース行列で与えられる線形制約）

(iii) 0-1 変数を含む制約付き多変数線形関数の最小化（線形制約）

(iv) ネットワーク上の流れに対する費用の最小化

(v) プロジェクトの日程計画に対する費用の最小化

(vi) 供給地から需要地への輸送費用の最小化

これらのサブルーチンは与えられた線形制約の中で、多変数線形関数の最小値探索を行う。（線形計画、混合 0-1 計画、最小費用流問題）

多変数 2 次関数の最小化では、以下のサブルーチンが用意されている。

-
- (i) 制約付き多変数凸型 2 次関数の最小化 (線形制約)
 - (ii) 制約付き多変数広義凸型 2 次関数の最小化 (線形制約)
 - (iii) 制約無し 0-1 多変数 2 次関数の最小化

これらのサブルーチンは与えられた制約の中で, 多変数 2 次関数の最小値探索を行う. (2 次計画問題)
制約付き多変数関数の最小化では, 以下のサブルーチンが用意されている.

- (i) 制約付き多変数線形関数の最小化 (非線形制約)

このサブルーチンは与えられた非線形制約の中で, 多変数関数の最小値探索を行う. (非線形計画)
ネットワーク上の距離の最小化では, 以下のサブルーチンが用意されている.

- (i) ネットワーク上のある節点から他のすべての節点までの距離の最小化
- (ii) ネットワーク上の全 2 節点間の距離の最小化
- (iii) ネットワーク上の 2 節点間の距離の最小化

これらのサブルーチンはネットワーク上の節点間の距離の最短化を行う. (最短路問題)

5.1.1 使用上の注意

- (1) 探索の出発点はできるかぎり真解に近いところにとるのが望ましい。
- (2) 要求精度は、 $\sqrt{(\text{誤差判定のための単位})}$ 程度にとるのが適当である。
- (3) 多変数関数の極小化を行う場合、各変数から関数値への寄与が同程度になるようにスケーリングするのが望ましい。たとえば、 $f(x) = 10000x_1^2 + x_2^3$ のような問題の場合、 $y_1 = 100x_1$, $y_2 = x_2$ と変数変換を行って、 $h(y) = y_1^2 + y_2^3$ としたほうがよい結果が得られる。なお、制約条件がある場合は、制約条件も変数変換によって変更する必要がある。

5.1.2 使用しているアルゴリズム

5.1.2.1 1変数関数の極小化

黄金分割探索と逐次放物線補間を組み合わせた方法を用いる。最初、黄金分割探索で探索を開始し、可能になれば逐次放物線補間に切り替える。これは、ブレント (1973) にもとづくものである。

(1) 黄金分割探索

黄金分割による区間縮小のアルゴリズムを以下に述べる。

縮小されるべき区間を $[a, b]$ とし、このなかの1点 $x(a < x < b)$ で関数値 $f(x)$ が計算されているとする。区間の中点 $c = (a + b)/2$ に対して、 $x < c$ であるとき、 $u = x + r(b - x)$ で関数値 $f(u)$ を計算し、 $f(x) < f(u)$ ならば、 $b = u$ とする。そうでなければ、 $a = x, x = u$ とする。 $x \geq c$ のときも、同様である。このようにすると、前に計算した関数値を利用できるので、1回区間縮小するのに関数計算を1回行えばよい。ここで、 $r = (3 - \sqrt{5})/2$ とし、最初に $x = a + r(b - a)$ ととっておくと、常に一定の比率 $1 - r$ で区間縮小されていく。

収束の次数は1である。

(2) 逐次放物線補間

逐次放物線補間による区間縮小のアルゴリズムを以下に述べる。

縮小されるべき区間を $[a, b]$ とする。いま、3点 v, w, x で関数値が計算されているとき、 $(v, f(v)), (w, f(w)), (x, f(x))$ の3点を通る放物線を作り、その頂点の x 座標を u とする。 $x < u$ のとき、 $f(x) < f(u)$ ならば、 $b = u$ とし、そうでなければ、 $a = x, x = u$ とする。 $x \geq u$ のときも、同様である。

極小点で $f''(x) > 0$ であり、十分よい近似から出発すれば、収束の次数は $1.324 \dots$ であることが証明される。

(3) 黄金分割探索から逐次放物線補間への切り替え

新たに関数値が計算されると、いままで計算された関数値のうち、もっとも小さな関数値の3点を与えるように v, w, x が更新される。可能ならば上記の放物線の頂点の x 座標 u が計算され、次の条件が満たされたとき、黄金分割探索から逐次放物線補間へ切り替えられる。

$$a < u < b$$

$$|x - u| < (b - a)/2$$

(4) 収束判定

要求精度を e_r とするとき、次の条件により収束判定を行う。

$$\max(b - x, x - a) \leq 2e_r \max(1, |x|)$$

(5) 囲い込み

制約なしの極小化では、任意の出発点から極小点を含む区間を求める囲い込みの操作を行っている。

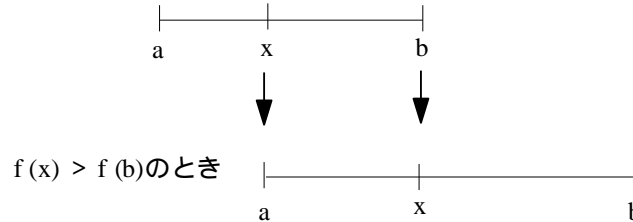
このアルゴリズムは、次のようになっている。

(i) 出発点 a の傾斜方向を求める。

(ii) 傾斜方向に $D = \max(\sqrt{(\text{誤差判定のための単位}) |x|}, 1.0, 2e_r |x|)$ の幅で囲い込み b 点とする。

(iii) b 点が a 点よりも関数値が小さい場合は極小点がないと判断し、 b 点を x 点にして b 点を a 点から $(3 - r)(x - a)$ の幅に広げ囲い込む。こうすれば、 $b - a : x - a = 1 : r$ の黄金分割比になる。 b 点が a 点よりも関数値が大きい場合は区間 $[a, b]$ に極小点があるので、この間で探索を開始する。

- (iv) b 点が x 点よりも関数値が小さい場合は, x 点を a 点に, b 点を x 点にし, 新しい b 点は a 点から $(3-r)(x-a)$ のところにとる. このようにして (iv) の作業を繰り返す. b 点が x 点よりも関数値が大きい場合は区間 $[a, b]$ で探索を開始する. このとき x 点は黄金分割探索の関数値計算点になっており, 囲い込みで得られた関数値を利用することができる.



5.1.2.2 多変数関数の極小化

問題は, n 変数の関数 $f(x)$ が与えられたとき, $f(x)$ を極小にする x を求めることである.

これは, 方程式 $\nabla f(x) = 0$ の解になっている. この方程式を解くために近似解を x_0 を出発点として逐次修正していく.

いま, x まで探索が進んで次の修正解 x' を求めようとしているとする. ニュートン法の修正ベクトル d は, 次式で与えられる.

$$d = -Hg$$

ただし, g は勾配ベクトル $g = \nabla^T f(x)$ であり, H はヘッセ行列 $B = \nabla^2 f(x)$ の逆行列である (T は転置を意味する). 実際には, d を方向ベクトルとして直線探索を行い次式により解を修正する.

$$x' = x + \alpha d$$

ここで, H を直接に計算せず, 逐次更新していく方法を準ニュートン法という. α は (b) 直線探索で決める.

(1) ヘッセ行列の逆行列 H の計算

H の更新公式は種々のものが提案されているが, 本ライブラリでは BFGS 公式を用いる. これは, Broyden, Fletcher, Goldfarb, Shanno によって提案されたものである.

H の初期値は単位行列 E であり, 以後, 以下の式で更新する.

$$H' = \left\{ E - \frac{\delta \gamma^T}{\delta \gamma^T} \right\} H \left\{ E - \frac{\gamma \delta^T}{\delta \gamma^T} \right\} + \frac{\delta \delta^T}{\delta \gamma^T}$$

ただし,

$$\begin{aligned} \delta &= x' - x \\ \gamma &= \nabla^T f(x') - \nabla^T f(x) \end{aligned}$$

である.

(2) 直線探索

正確な直線探索は効率が良くないため, Armijo の方法を用いる.

$$f(x + \beta^m \alpha_0 d) - f(x) \leq \beta^m \alpha_0 \mu \nabla f(x) d$$

を満たす最小の非負整数 m を求めて, $\alpha = \beta^m \alpha_0$ とする.

ただし, $\alpha_0 > 0, 0 < \beta < 1, 0 < \mu < 1$ である.

(3) 収束判定

収束判定は以下の式によって行い, x' を解とする.

$$\|\nabla^T f(x')\|_\infty \leq \varepsilon (\|x' - x\|_\infty \leq \varepsilon \text{ かつ 前回の修正に異常が発生した})$$

または,

$$\|x' - x\|_\infty \leq e_r \max(1, \|x'\|_\infty) \text{ かつ } \|\nabla^T f(x')\|_\infty \leq 2e_r$$

ここで, ε は誤差判定のための単位, e_r は要求精度であり, $\|x\|_\infty = \max_i |x_i|$ である.

5.1.2.3 非線形最小二乗法

問題は, m 個の n 変数関数 $f_1(x), \dots, f_m(x)$ が与えられたとき, 関数の二乗和,

$$S(x) = \sum_{i=1}^m f_i(x)^2 = \|f(x)\|^2$$

を極小とする x を求めることである. ただし, $f(x)$ は, $f_1(x), \dots, f_m(x)$ を成分とするベクトル値関数であり, $\|x\| = \sqrt{x^T x}$ である (T は転置を意味する).

それには方程式 $\partial S / \partial x = 0$ を解く必要がある. 解を求めるために, 初期値 x_0 から探索を開始し, パウエルハイブリッド法を用いて, 逐次修正していく.

ハイブリッド法では, 関数の非線形性に依じて線形化したモデルのガウスニュートン解と最急降下解とを折衷する. 修正ベクトルは常に独立性の検査が行われ, 部分空間の中に閉じ込められてしまわないように配慮されている. また, ヤコビ行列の計算を直接行わず, 関数情報を用いて修正する.

(1) 修正ベクトル Δx の計算

f を線形化したモデルは,

$$S_L(x + \Delta x) = \|f(x) + A\Delta x\|^2$$

である. ここで, A は f のヤコビ行列 $\partial f / \partial x$ である.

このモデルの最急降下解 Δx_S は, $b = -A^T f(x)$ として,

$$\Delta x_S = (\|b\|^2 / \|Ab\|^2) b$$

によって与えられる.

一方, ガウスニュートン解 Δx_G は, 正規方程式

$$A^T A \Delta x = b$$

を解いて得られる. 本ライブラリでは, これを解くために線形最小二乗法の QR 分解法を用いる.

一般に, $\|\Delta x_S\| \leq \|\Delta x_G\|$ であることが証明される.

この 2 つの解を関数の非線形性に依じて決定されるステップサイズ d によって以下のように折衷する.

(i) $d \leq \|\Delta x_S\|$ の場合

$$\Delta x = d \Delta x_S / \|\Delta x_S\|$$

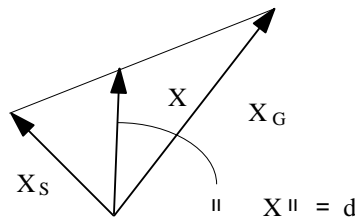
(ii) $\|\Delta x_S\| < d < \|\Delta x_G\|$ の場合 (図 5-1 参照)

$$\Delta x = \alpha \Delta x_S + \beta \Delta x_G (\alpha > 0, \beta > 0, \|\Delta x\| = d)$$

(iii) $\|\Delta x_G\| \leq d$ の場合

$$\Delta x = \Delta x_G$$

図 5-1



(2) ステップサイズ d の変更

ステップサイズは、初期値 $d = \|\Delta x_S\|$ として、以後、関数の非線形性が強いときには減少させ、線形に近いときには増加させる。

非線形性の程度を測るために線形のモデルの変化量 $\Delta S_L = \Delta S_L(x + \Delta x) - S_L(x)$ と実際の変化量 $\Delta S = S(x + \Delta x) - S(x)$ との比 $r = \Delta S / \Delta S_L$ を用いる。

- a) $r < 0.1$ のとき非線形性が強いと判断し、 d を半分にする。
- b) $r \geq 0.1$ のとき、 d の増加率 λ を以下のように計算する。

$$\lambda^2 = 1.0 - (r - 0.1)\Delta S_L / (S_P + \sqrt{(S_P^2 - S_S(r - 0.1)\Delta S_L)})$$

ただし、 $\delta f = f(x + \Delta x) - (f(x) + A\Delta x)$ として

$$S_P = \sum_{i=1}^n \|f_i(x + \Delta x)\delta f_i\|$$

$$S_S = \|\delta f\|^2$$

である。

実際には、 d の振動を防ぐために 2 回続けて増加が要求されたときにだけ d を増加させる。また、増加率は 2 以下におさえる。実際の増加率 μ は以下のように計算される。

$$\begin{aligned} \mu &= \min(2, \lambda, \tau) \\ \tau &= \lambda / \mu \end{aligned}$$

ここで、 τ は初期値が 1 であり、縮小が要求されたとき 1 にリセットされる。

また、 d には上限 d_{\max} と下限 d_{\min} が設けられており、その間にはいるように制御される。

(3) 修正ベクトルの独立性検査

ヤコビ行列の修正が効率よく行われるためには、順次取られる修正ベクトルが互いに直交に近いことが必要である。そのために、ハイブリッド法では、独自の独立性概念を定義して、修正ベクトルができるだけ独立な方向に取られるように制御している。ベクトル p が i 個のベクトル (p_1, p_2, \dots, p_i) と独立であるとは、 p がこれらのベクトルで張られた空間の任意のベクトルと 30 度以上の角度をなしていることである。パウエルによって考案された独立性検査の算法を以下に示す。

過去 $2n$ 回のヤコビ行列の修正で用いられたベクトルのうち互いに独立な n 個のベクトルを直交化して、 $\Omega = (\omega_1, \omega_2, \dots, \omega_n)$ に保持しておく。大きさ n の配列 j を用いて、 ω_i が何回前の修正ベクトルであったかという情報を保持しておく。すなわち、 ω_i は j_i 回前に取られたベクトルであることを意味する。 Ω は単位行列で初期化し、 j は $j_i = n - i + 1$ ($i = 1, \dots, n$) で初期化する。

解の修正を行うときには、以下のようにする。

(i) $\Delta x = \Delta x_G$ の場合

独立性のいかんにかかわらず, Δx を修正ベクトルとして採用する.

(ii) $\Delta x = \Delta x_G$ でない場合

$j_1 < 2n$ であるか, または Δx が $(\omega_2, \omega_3, \dots, \omega_n)$ と独立なら, Δx を修正ベクトルとして採用する. そうでないなら, 解の修正を行わない.

(4) ヤコビ行列 A の計算

ヤコビ行列は, 最初の 1 回だけは差分によって求め, 後は逐次更新していく. この方法は, プロイデンによるもので, 以下の式によって計算する.

$$A' = A + \delta f \Delta x^T / \|\Delta x\|^2$$

ただし, $\|\Delta x\| < d_{\min}$ であるか, または, $j_1 = 2n$ で Δx が $(\omega_2, \omega_3, \dots, \omega_n)$ と独立でないなら, $\Delta x = d_{\min} \omega_1$ とする.

(5) Ω と j の改訂

Ω と j の改訂は以下のように行う.

$\Delta x = d_{\min} \omega_1$ としたときには,

$$\omega_i = \omega_{i+1} \quad (i = 1, \dots, n-1)$$

$$\omega_n = \omega_1$$

$$j_i = j_{i+1} + 1 \quad (i = 1, \dots, n-1)$$

$$j_n = 1$$

とすればよい. そうでないときには, 以下のようにする.

$(\omega_{k+1}, \omega_{k+2}, \dots, \omega_n, \Delta x)$ が互いに独立になる最小の k を求める.

$(\omega_1, \dots, \omega_{k-1}, \omega_{k+1}, \omega_{k+2}, \dots, \omega_n, \Delta x)$ を直交化して, それを改めて $(\omega_1, \omega_2, \dots, \omega_n)$ とする.

$$j_i = j_{i+1} \quad (i = 1, \dots, k-1)$$

$$j_i = j_{i+1} + 1 \quad (i = k, \dots, n-1)$$

$$j_n = 1$$

このようにして, 常に, $j_1 \leq 2n$ となるようにする.

(6) 収束判定

収束判定は以下の式によって行い, $x + \Delta x$ を解とする.

$$\|\Delta x\|_{\infty} \leq e_r \max(1, \|x + \Delta x\|_{\infty})$$

ここで, e_r は要求精度であり, $\|x\|_{\infty} = \max_i |x_i|$ である.

5.1.2.4 制約付き多変数線形関数の最小化 (線形制約)

ここでは, 次のように, n 次ベクトル x に関する多変数線形関数 $f(x)$ と, m 個の線形制約条件, 並びにベクトル x の定義域が与えられたとき, 線形制約条件を満たし, かつ $f(x)$ を最小にする n 次のベクトル x を求める問題 (線形計画問題) を扱う.

$$f(x) = c^T x \rightarrow \min$$

$$\begin{aligned} \text{線形制約条件: } \mathbf{a}_i^T \mathbf{x} &= b_i \quad (i = 1, 2, \dots, m_e) \\ \mathbf{a}_i^T \mathbf{x} &\leq b_i \quad (i = m_e + 1, m_e + 2, \dots, m_{ne}) \\ \mathbf{a}_i^T \mathbf{x} &\geq b_i \quad (i = m_{ne} + 1, m_{ne} + 2, \dots, m) \quad 0 \leq m_e \leq m_{ne} \leq m \end{aligned}$$

$$\mathbf{x} \text{ の定義域: } \mathbf{d} \leq \mathbf{x} \leq \mathbf{u}$$

ここで, $\mathbf{c}, \mathbf{a}_i, \mathbf{d}, \mathbf{u}$ はそれぞれ n 次の定数ベクトルであり, (b_i) は m 次の定数ベクトルである. これらは問題によって定まる.

なお, $\mathbf{d} \leq \mathbf{x}$ というように, ベクトルに対して不等式を用いているが, これらは, ベクトルの各要素について, この不等式が成立するという意味で用いている.

なお, 変数 X_i を新たに追加することによって, 不等式制約条件 $\mathbf{a}_i^T \mathbf{x} \leq b_i$ は $\mathbf{a}_i^T \mathbf{x} + X_i = b_i$ に $\mathbf{a}_i^T \mathbf{x} \geq b_i$ は $\mathbf{a}_i^T \mathbf{x} - X_i = b_i$ にそれぞれ変換できる. この変数 X_i をスラック変数と呼ぶ. スラック変数の導入により, 制約条件は次のように全て等式で表現できるようになる.

$$\text{線形制約条件: } \mathbf{A}\mathbf{x}' = \mathbf{b}$$

$$\mathbf{x}' \text{ の定義域: } \mathbf{d}' \leq \mathbf{x}' \leq \mathbf{u}'$$

ここで, \mathbf{A} は n 行 $n + m - m_e$ 列の行列, $\mathbf{x}', \mathbf{b}, \mathbf{d}', \mathbf{u}'$ はそれぞれ $n + m - m_e$ 次のベクトルである. 従って, 以降では等式制約条件のみを考える. なお, 表記を簡単にするため, $\mathbf{x}', \mathbf{d}', \mathbf{u}'$ をそれぞれ $\mathbf{x}, \mathbf{d}, \mathbf{u}$ と表すことにする. 本ライブラリでは, 線形計画問題を解くために, 制約条件を与える行列 $\mathbf{A} = (a_{ij})$ が密行列である場合を扱うための改訂シンプレックス法を用いたサブルーチンと, \mathbf{A} がスパース行列である場合を扱うための内点法を用いたサブルーチンを用意している.

まず, 改訂シンプレックス法について説明する.

(1) 実行可能基底解

最適解は, 実行可能基底解の中に存在する. シンプレックス法は, 有限集合である実行可能基底解の中から最適解を求めるものである. 以下, 実行可能基底解について説明する.

行列 \mathbf{A} の列から m 個の 1 次独立な列を選び, このように選んだ列を要素とする m 行 m 列の正則 \mathbf{B} を考える. 今 $n + m - m_e$ 次の変換行列 \mathbf{P} を

$$\begin{aligned} \mathbf{AP} &= [\mathbf{B} | \mathbf{L} | \mathbf{U}] \\ (\mathbf{P}^{-1}\mathbf{x})^T &= [\mathbf{x}_B^T | \mathbf{x}_L^T | \mathbf{x}_U^T], \\ (\mathbf{P}^{-1}\mathbf{d})^T &= [\mathbf{d}_B^T | \mathbf{d}_L^T | \mathbf{d}_U^T], \\ (\mathbf{P}^{-1}\mathbf{u})^T &= [\mathbf{u}_B^T | \mathbf{u}_L^T | \mathbf{u}_U^T] \end{aligned}$$

を満たすように定めると, 制約条件は次の形に書き表せる.

$$\begin{aligned} \mathbf{B}\mathbf{x}_B + \mathbf{L}\mathbf{x}_L + \mathbf{U}\mathbf{x}_U &= \mathbf{b} \\ \mathbf{d}_B &\leq \mathbf{x}_B \leq \mathbf{u}_B \\ \mathbf{d}_L &\leq \mathbf{x}_L \leq \mathbf{u}_L \\ \mathbf{d}_U &\leq \mathbf{x}_U \leq \mathbf{u}_U \end{aligned}$$

ここでは, $\mathbf{x}_B, \mathbf{d}_B, \mathbf{u}_B$ は m 次のベクトル, \mathbf{L} は m 行 ℓ 列の行列, $\mathbf{x}_L, \mathbf{d}_L, \mathbf{u}_L$ は ℓ 次のベクトル, \mathbf{U} は m 行 $n - m_e - 1$ 列の行列, $\mathbf{x}_U, \mathbf{d}_U, \mathbf{u}_U$ は $n - m_e - 1$ 次のベクトルをあらわす. ただし, ℓ は $0 \leq \ell \leq n - m_e$ を満たす整数である.

今, x_B, x_L, x_U が次の条件を満たしている場合, $x = P \begin{bmatrix} x_B \\ x_L \\ x_U \end{bmatrix}$ を実行可能解と呼ぶ.

$$\begin{aligned} d_B &\leq x_B \leq u_B \\ x_L &= d_L \\ x_U &= u_U \end{aligned}$$

なお, B を基底行列, x_B を基底変数, そして L, U を非基底行列, x_L, x_U を非基底変数と呼ぶ.

(2) シンプレックス法

シンプレックス法の手順を示す.

- (a) 初期実行可能基底解を求める.
- (b) 他の非基底変数を固定した状態で, ある1つの非基底変数 X_{IN} (x_L または x_U の要素) を, ある変数 $x(x_B$ の要素または $X_{IN})$ が上限値または下限値になるまで, 変化させる.
- (c) x_B が上限値または下限値になったときには, 基底と非基底を入れ替える.
- (d) この操作を, 関数の値を小さくできる限り, いろいろな非基底変数 x を X_{IN} として選び, 繰り返し行う.

以下に具体的に説明する.

まず, 初期実行可能基底解を求める (参考文献 (7) 参照).

次に, 変化させる非基底変数 X_{IN} を求める. 基底変数, 非基底変数を用い, 制約条件を含めた形で関数値 $f(x)$ を表すと次式になる.

$$\begin{aligned} f(x) &= c_B^T B^{-1} b + g_L^T x_L + g_U^T x_U \\ g_L^T &= c_L^T - c_B^T B^{-1} L \\ g_U^T &= c_U^T - c_B^T B^{-1} U \end{aligned}$$

ただし, $(Pc)^T = (c_B^T \ c_L^T \ c_U^T)$ で c_B, c_L, c_U はそれぞれ m 次, ℓ 次, $n - m_e - 1$ 次のベクトルである.

$x_L = d_L$ および $x_U = u_U$ であるので, x_L の要素は + 方向, x_U の要素は - 方向にしか変化させることができない. そのため, ある g_L の要素が負, またはある g_U の要素が正のときのみ関数値 $f(x)$ を減少させることができる. このような要素がないときは現在の値 x が最適解である. この条件を満たす g_U または g_L の要素で絶対値が最大のものが, 変化させる要素 IN となる.

このとき, 上限値または下限値に変化する $x(x_B$ の要素または $X_{IN})$ は次のものになる. 非基底変数 X_{IN} を変化させたとき, X_{IN} が x_L の要素と仮定すると, 制約条件を満たしている間は次式が成立する. ここで, a_{IN} は A の第 IN 列 (前述の a とは異なる), Δ は X_{IN} の変化量 ($\Delta \geq 0$) である.

$$\begin{aligned} d_B &\leq x_B - B^{-1} a_{IN} \Delta \leq u_B \\ d_{IN} &\leq X_{IN} + \Delta \leq u_{IN} \end{aligned}$$

Δ を変化させたとき, 最初に上式を満たさなくなる $x(x_B$ の要素または $X_{IN})$ が上限値または下限値に変化する変数になる. なお, x_{IN} が x_U の要素のときには, 上式ではなく次式になる.

$$\begin{aligned} d_B &\leq x_B + B^{-1} a_{IN} \Delta \leq u_B \\ d_{IN} &\leq X_{IN} - \Delta \leq u_{IN} \end{aligned}$$

いま $x^{(k)}$ が実行可能解であるとき、 $x^{(k+1)} = x^{(k)} + td^{(k)}$ が実行可能かつ $c^T x^{(k+1)} < c^T x^{(k)}$ を満たすベクトル $d^{(k)}$ のうちで、目的関数の減少率がもっとも大きくなる方向ベクトル $d^{(k)}$ の方向を探索する。ここで t をステップサイズと呼ぶ。もし、制約条件を考慮しなければ、目的関数の値 $c \cdot x$ をもっとも減少させる方向は $-c$ の方向である。実際は、制約条件 $Ax^{(k+1)} = A(x^{(k)} + td^{(k)}) = b$ を満たす必要があるから、 $d^{(k)}$ の方向としては $-c$ を S の部分空間 $\{x | Ax = 0\}$ に射影したものを取れば良い。

(3) ビッグ M 法

内点法により最適解を探索するためには、探索の出発点として実行可能領域の内点すなわち、実行可能領域の内部にあって、境界上にない初期解 $x^{(0)}$ が必要である。しかしながら、このような初期解を求めるのは自明なことではない。そこで、本ライブラリでは、ビッグ M 法と呼ばれる方法を用いて実行可能解の計算と最適解の計算を同時に行うようにしている。まず人為変数

$$x' = (x_{n+1}, x_{n+2}, \dots, x_{n+m})^T$$

を導入して、以下のような $n + m$ 変数の線形計画問題を定式化する。

$$\begin{aligned} \bar{f}(x, x') &= c^T x + Mx' \rightarrow \min \\ \text{線形制約条件: } & Ax + \bar{A}x' = b \\ x \text{ の定義域: } & 0 \leq x \leq u \\ x' \text{ の定義域: } & 0 \leq x' \end{aligned} \tag{5.2}$$

任意の $x^{(0)} = (x_1^{(0)}, x_2^{(0)}, \dots, x_n^{(0)})^T$ に対して $m \times m$ 行列 $\bar{A} = (\bar{a}_{ij}) (1 \leq i, j \leq m)$ を

$$\bar{a}_{ij} = \begin{cases} \sum_{k=1}^n a_{ik} x_k^{(0)} & (i = j \text{ のとき}) \\ 0 & (i \neq j \text{ の場合}) \end{cases}$$

で定義しておけば、

$$\begin{bmatrix} x \\ x' \end{bmatrix} = (x_1^{(0)}, x_2^{(0)}, \dots, x_n^{(0)}, 1, 1, \dots, 1)^T$$

は、あきらかに (5.2) の実行可能解である。パラメータ M を十分大きな正数に取っておけば、人為変数 $x' = (x'_1, x'_2, \dots, x'_m)^T$ (5.2) の目的関数 $\bar{f}(x, x')$ を内点法の手続きによって減少させて行くとき、まず、人為変数の項 Mx' が速やかに 0 に近づく。このことは、各人為変数が急速に 0 に近づくことを意味する。各人為変数が十分に 0 に近ければ、(5.2) の実行可能解の人為変数を除いた部分は (5.1) の実行可能解と見なせる。さらに目的関数 $\bar{f}(x, x')$ を減少させて行くと、今度は $c^T x$ の項が減少して行くので、最終的に (5.2) の最適解の人為変数を除いた部分は、(5.1) の実行可能な最適解になる。なお、(5.2) の最適解において人為変数が十分に 0 に近づいていないとき、すなわち与えられた小さな正数 ϵ_A に対して

$$\max_{1 \leq j \leq m} \{x'_{n+j}\} > \epsilon_A$$

であるとき問題 (5.1) は実行不可能であったと判定する。

以下の説明では (5.1) にはすでに人為変数を組み入れてあるものとする。

(4) アフィン変換による探索方向の決定

上記の最適解の探索方向の決定方法 $x^{(k)}$ が S の中央付近にある場合はステップサイズ t を十分大きく取れるため、目的関数の値を大きく減少させることができるが、 $x^{(k)}$ が S の境界付近にある場合、ステップサイズを大きく取ることができないため、目的関数を十分に減少させられない。そこで変数 x に対する変換を行って現在の

解 $x^{(k)}$ が変換先の空間における実行可能領域 S' の境界から十分離れるようにしてから、目的関数を大きく減少させるように解を更新し、そうして得られた変換先の空間での新しい解に対して逆変換を行って $x^{(k+1)}$ とする。この目的のために、現在の解 $x^{(k)}$ に対して定義される対角行列

$$D^{(k)} = \begin{bmatrix} x_1^{(k)} & & & 0 \\ & x_2^{(k)} & & \\ & & \ddots & \\ 0 & & & x_n^{(k)} \end{bmatrix}$$

によりアフィン変換

$$y = (D^{(k)})^{-1}x$$

を行う。変換先の空間において $x^{(k)}$ がつねに $(1, 1, \dots, 1)^T$ に移されることは明らかである。 y の空間においてもとの線形計画問題は次のように表される。

$$\begin{aligned} f(y) &= \hat{c}^{(k)T}y \rightarrow \min \\ \text{線形制約条件: } & \hat{A}^{(k)}y = b \\ y \text{ の定義域: } & 0 \leq y \leq \hat{u}^{(k)} \end{aligned}$$

ここで

$$\hat{A}^{(k)} = AD^{(k)}$$

$$\hat{c}^{(k)} = D^{(k)}c$$

である。この問題における目的関数の減少率を最大にする方向 $\hat{d}^{(k)}$ は $-\hat{c}^{(k)}$ を部分空間 $\{y | \hat{A}^{(k)}y = 0\}$ に射影したものである。この射影は射影行列

$$\hat{P}^{(k)} = I - (\hat{A}^{(k)})^T(\hat{A}^{(k)}(\hat{A}^{(k)})^T)^{-1}\hat{A}^{(k)}$$

で与えられるので、探索方向は

$$\hat{d}^{(k)} = -\hat{P}^{(k)}\hat{c}^{(k)}$$

となるが、これはもとの変数での探索方向 $d^{(k)}$ とは、

$$d^{(k)} = D^{(k)}\hat{d}^{(k)}$$

という関係を持っているので、結局、もとの空間における探索方向は

$$d^{(k)} = -D^{(k)}\hat{P}^{(k)}\hat{c}^{(k)} = -(D^{(k)})^2(I - A^T(A(D^{(k)})^2A^T)^{-1}A(D^{(k)})^2)c$$

で与えられる。この式には逆行列の計算が含まれているが、実際の計算では、まず連立1次方程式

$$(A(D^{(k)})^2A^T)w^{(k)} = A(D^{(k)})^2c$$

を解いて、 $w^{(k)}$ を求め

$$d^{(k)} = -(D^{(k)})^2(c - A^T w^{(k)})$$

として探索方向 $d^{(k)}$ を定めることができる。

(5) ステップサイズの決定

上記のように定められた探索方向 $d^{(k)}$ により与えられる直線 $x^{(k)} + td^{(k)}$ ($t \geq 0$) 上の点は制約条件

$$Ax = b$$

を満たし、かつ目的関数はステップサイズ t の増加に対して単調に減少する。しかし、変数の定義域 $0 \leq x \leq u$ の外に出ることは許されないので、ステップサイズ t の取りうる最大値は

$$t_{\max} = \min \left\{ \min \left\{ \frac{x_j^{(k)}}{-d_j^{(k)}} \mid d_j^{(k)} < 0 \right\}, \min \left\{ \frac{u_j - x_j^{(k)}}{d_j^{(k)}} \mid d_j^{(k)} > 0 \right\} \right\}$$

となる。実際には、実行可能領域の境界に達してしまうと、これ以上探索を続けることができないので、 $0 < \alpha < 1$ を満たすパラメータ α を用いて、

$$t^{(k)} = \alpha t_{\max}$$

が $x^{(k)}$ に対するステップサイズとなり、新しい解は、

$$x^{(k+1)} = x^{(k)} + t^{(k)} d^{(k)}$$

で与えられる。

(6) 変数の上限と下限の入れ替え

最適解の探索の途中で、いずれかの変数が下限に近づいた場合に前述のアフィン変換は有効であるが、変数が上限に近づいた場合は同じ方法は用いることができない。そこで、本サブルーチンでは変数 x_j がその上限 u_j に近づくと以下のような変換を行って変数の上限と下限を入れ替える。

$$\begin{aligned} x_j &\leftarrow u_j - x_j \\ u_j &\leftarrow u_j \\ c_j &\leftarrow -c_j \\ b_i &\leftarrow b_i - a_{ij}u_j \quad (i = 1, 2, \dots, m) \\ a_{ij} &\leftarrow -a_{ij} \quad (i = 1, 2, \dots, m) \end{aligned}$$

(7) 制約条件に対する残差のチェック

最適解の探索のための反復計算を繰り返すうちに、計算誤差のため制約条件に対する残差 $\|Ax - b\|$ が大きくなってしまふことがある。いったん残差が大きくなってしまふとそれ以降の計算は意味をなさなくなる。これを防ぐため、反復回数の ℓ 回ごとに与えられた正数 ϵ_r に対して

$$\|Ax^{(k)} - b\| \leq \epsilon_r$$

が満たされているかどうかをチェックし、満たされていないならば

$$\begin{bmatrix} x \\ x' \end{bmatrix} = (x_1^{(k-\ell)}, x_2^{(k-\ell)}, \dots, x_n^{(k-\ell)}, 1, 1, \dots, 1)^T$$

を初期解として問題 (5.2) の行列 \bar{A} を再計算して最適解の探索を始めからやり直す。

(8) 収束の判定

$$\frac{c^T x^{(k-1)} - c^T x^{(k)}}{1 + \|c^T x^{(k)}\|} \leq \epsilon_C$$

を満たした時、 $x^{(k)}$ は最適解に収束したものとみなす。 ϵ_C は収束判定のためのパラメータである。

5.1.2.5 0-1 変数を含む線形制約付き多変数線形関数の最小化

ここでは線形計画問題に、変数のうちのいくつかが 0-1 変数すなわち 0 または 1 のみを値として取るという条件を付加した問題 (混合 0-1 計画問題) を扱う。

$$\begin{aligned} \text{目的関数} & : f(\mathbf{x}) = \mathbf{c}^T \mathbf{x} \rightarrow \min \\ \text{線形制約条件} & : a_{ij}x_j = b_i \quad (i = 1, \dots, m; j = 1, \dots, n) \\ \mathbf{x} \text{の定義域} & : d_j \leq x_j \leq u_j \quad (j = n_{01} + 1, \dots, n) \\ \text{0-1 変数条件} & : x_j = 0 \text{ or } 1 \quad (j = 1, \dots, n_{01}) \\ & \quad (1 \leq n_{01} \leq n) \end{aligned}$$

ただしここであらかじめ必要な個数のスラック変数を導入するなどして、制約条件としては等式制約条件だけを含むようにしてあるものとして説明を進める。

本ライブラリでは、混合 0-1 計画問題を解くために、分枝限定法を用いる。

分枝限定法とは元の問題をいくつかの部分問題に分解し、それらを全て解くことにより元の問題の解を得る方法である。混合 0-1 計画問題での部分問題とは、0-1 変数のいくつかの値を 0 または 1 に固定したものであって、以下のように表される。

$$\begin{aligned} \text{目的関数} & : f(\mathbf{x}) = \mathbf{c}^T \mathbf{x} \rightarrow \min \\ \text{線形制約条件} & : a_{ij}x_j = b_i \quad (i = 1, \dots, m; j = 1, \dots, n) \\ \mathbf{x} \text{の定義域} & : d_j \leq x_j \leq u_j \quad (j \notin S^0 \cup S^1 \cup F) \\ \text{0-1 変数条件} & : x_j = 0 \quad (j \in S^0) \\ & \quad x_j = 1 \quad (j \in S^1) \\ & \quad x_j = 0 \text{ or } 1 \quad (j \in F) \end{aligned}$$

ここで S^0, S^1, F はそれぞれ部分問題 P_k において、0 に固定された 0-1 変数の添字の集合、1 に固定された 0-1 変数の添字の集合、ならびに値が固定されていない 0-1 変数の添字の集合である。なお、0 または 1 に固定された 0-1 変数を固定変数、それ以外の 0-1 変数を自由変数と呼ぶ。部分問題は、それ自体混合 0-1 計画問題である。とくに $S^0 \cup S^1 = \phi$ すなわち固定変数を持たない部分問題は元の混合 0-1 計画問題 (以後、 P_1 と呼ぶ。) そのものである。また、部分問題に対して次のように表される線形計画問題を元の部分問題に対する緩和問題と呼ぶ。

$$\begin{aligned} \text{目的関数} & : f(\mathbf{x}) = \mathbf{c}^T \mathbf{x} \rightarrow \min \\ \text{線形制約条件} & : a_{ij}x_j = b_i \quad (i = 1, \dots, m; j = 1, \dots, n) \\ \mathbf{x} \text{の定義域} & : d_j \leq x_j \leq u_j \quad (j \notin S^0 \cup S^1 \cup F) \\ \text{0-1 変数条件} & : x_j = 0 \quad (j \in S^0) \\ & \quad x_j = 1 \quad (j \in S^1) \\ & \quad 0 \leq x_j \leq 1 \quad (j \in F) \end{aligned}$$

以下に分枝限定法により P_1 の解を求める方法について説明する。

(1) 初期設定

(a) 部分問題のリスト

分枝限定法では部分問題のリスト PLIST を使用する。初期状態では PLIST のメンバは P_1 だけで、PLIST のメンバ数 L は 1 である。 L は分枝限定法の計算が進むにしたがって、増減する。

(b) 暫定解

初期状態では P_1 の実行可能解は 1 つも見つかっていないので、暫定解 \mathbf{x}^* は未定で暫定解 \mathbf{x}^* に対する目的関数値 z^* としては十分大きな正の値を設定しておく。

(c) 擬コスト

PLIST の更新に使用する擬コスト $h_{up}(j), h_{down}(j)$ ($j \in F$) の初期値を以下のように設定する. まず P_1 に対する緩和問題を改訂シンプレックス法を用いて解く. このとき求められた最適解を \bar{x} とし, そのうち基底変数の添字の集合を B , 定義域の上限値を取る非基底変数の添字の集合を N_{up} , 定義域の下限値を取る非基底変数の添字の集合を N_{down} とすると各基底変数 \bar{x}_{B_i} と目的関数値 $f(\bar{x})$ は非基底変数を用いて以下のように表わされる.

$$f(\bar{x}) = \bar{z}_0 + \sum_{j \in N_{down}} \bar{c}_j \bar{x}_j - \sum_{j \in N_{up}} \bar{c}_j \bar{x}_j$$

$$\bar{x}_{B_i} = \bar{b}_i - \sum_{j \in N_{down}} y_{ij} \bar{x}_j - \sum_{j \in N_{up}} y_{ij} \bar{x}_j \quad (B_i \in B)$$

$\bar{z}_0, \bar{b}_i, \bar{c}_j$ および y_{ij} は P_1 の緩和問題の最適解を求める際に改訂シンプレックス法の計算の過程で求められる量であって,

$$\bar{c}_j \geq 0 \quad (j \in N_{up} \cup N_{down})$$

を満たす. これらの量を用いて $h_{up}(p), h_{down}(p)$ の初期値を以下のように設定する.

$p \in N_{up}$ の場合:

$$h_{up}(p) = 0$$

$$h_{down}(p) = -\bar{c}_p$$

$p \in N_{down}$ の場合:

$$h_{up}(p) = \bar{c}_p$$

$$h_{down}(p) = 0$$

$p = B_i \in B$ の場合:

$$h_{up}(p) = \min\{-\bar{c}_j/y_{ij} | y_{ij} < 0, j \in N_{down} - F \text{ または } y_{ij} > 0, j \in N_{up} - F\}$$

$$h_{down}(p) = \min\{\bar{c}_j/y_{ij} | y_{ij} > 0, j \in N_{down} - F \text{ または } y_{ij} < 0, j \in N_{up} - F\}$$

ただし $p = B_i \in B$ の場合で $h_{up}(p)$ または $h_{down}(p)$ の一方について右辺の条件を満たす j が存在しない場合には, 値として十分大きな正の数を設定する. もしも $h_{up}(p)$ と $h_{down}(p)$ の両方で右辺の条件を満たす j が存在しない場合には

$$h_{up}(p) = \frac{\sum_{j \in N_{up}} -\bar{c}_j/y_{ij}}{|M_{up}|}$$

$$h_{down}(p) = \frac{\sum_{j \in N_{up}} \bar{c}_j/y_{ij}}{|M_{down}|}$$

ここで

$$M_{up}(p) = \{j | y_{ij} < 0, j \in N_{down} \cap F \text{ または } y_{ij} > 0, j \in N_{up} \cap F\}$$

$$M_{down}(p) = \{j | y_{ij} > 0, j \in N_{down} \cap F \text{ または } y_{ij} < 0, j \in N_{up} \cap F\}$$

(2) 緩和問題

PLIST の含まれる部分問題について対応する緩和問題が解かれていないものがあれば, それを改訂シンプレックス法を用いてを解く. 部分問題 P_i に対応する緩和問題の解に対する最適値 $g(P_i)$ について $g(P_i) > z^*$ であった場合には, P_i を PLIST から除き, $L \leftarrow L - 1$ とする. また最適解を \bar{x} とすると. すべての $j \in F$ について \bar{x}_j の値が 0 または 1 であった場合には, 緩和問題の解が元の部分問題の解になり, 部分問題の最適値は緩和問題の最適値 $g(P_i)$ に等しい. このとき $f(P_i) < z^*$ ならば x^* と z^* を \bar{x} と $f(P_i)$ で置き換える.

なお、緩和問題を新たに解く必要があるのは、上記の初期設定のときと後述する分枝操作によって L が増加したときである。後者の場合、PLIST の最後の 2 つの部分問題は分枝変数 x_{j_0} を 0 に固定した部分問題 P_{i_0} と 1 に固定した部分問題 P_{i_1} の組になっている。このとき上で求めた緩和問題の最適値 $g(P_{i_0}), g(P_{i_1})$ および分枝操作前の部分問題 P_i の最適値 $g(P_i)$ と最適解の値 \bar{x}_{j_0} を用いて、擬コストを次のように更新する。

$$\begin{aligned} h_{\text{up}}(j_0) &= (g(P_{i_1}) - g(P_i)) / (1 - \bar{x}_{j_0}) \\ h_{\text{down}}(j_0) &= (g(P_{i_0}) - g(P_i)) / \bar{x}_{j_0} \end{aligned}$$

(3) 最適値の推定

もし後述する分枝操作の後であれば $K' = \min\{K + 1, L\}$, そうでない場合には $K' = \min\{K, L\}$ とする。ここで、 K は分枝限定法における探索の深さと呼ばれ、前もって与えておく正の整数のパラメータである。PLIST の最後の K' 個の部分問題 P_i について、緩和問題の最適値 $g(P_i)$, 最適解 \bar{x} および擬コスト $h_{\text{up}}(j), h_{\text{down}}(j)$ を用いて、部分問題の最適値 $f(P_k)$ の推定値

$$h(P_i) = g(P_i) + \sum_{j \in F} \min\{h_{\text{up}}(j)(1 - \bar{x}_j), h_{\text{down}}(j)\bar{x}_j\}$$

を計算を計算し、 $h(P_i)$ の降順に PLIST の最後の K' 個を並べ代える。

(4) 部分問題のテスト

PLIST の最後の部分問題 P_k についてその緩和問題の最適解の基底変数 \bar{x}_{B_i} ($B_i \in B$) を非基底変数 \bar{x}_j ($j \in N_{\text{up}} \cup N_{\text{down}}$) で表現すると、先に擬コストの初期値を計算する際に現われた $\bar{z}_0, y_{ij}, \bar{c}_j, \bar{b}_i$ といった量に相当するものが現われる。これらの量を用いて、 $Z_{\text{max}}(i), Z_{\text{min}}(i)$ を次のように定義する。

$$\begin{aligned} Z_{\text{max}}(i) &= \bar{b}_i - \sum_{j \in N_{\text{down}} - S^0} \min\{0, y_{ij}\}(u_j - d_j) + \sum_{j \in N_{\text{up}} - S^1} \max\{0, y_{ij}\}(u_j - d_j) \\ Z_{\text{min}}(i) &= \bar{b}_i - \sum_{j \in N_{\text{down}} - S^0} \max\{0, y_{ij}\}(u_j - d_j) + \sum_{j \in N_{\text{up}} - S^1} \min\{0, y_{ij}\}(u_j - d_j) \end{aligned}$$

そして以下の 8 つの規則により自由変数 $x_j (j \in F)$ のうちのいくつかを 0 または 1 に固定することができる。

- $Z_{\text{max}}(i) < 1$ かつ $B_i \in F$ ならば $x_{B_i} = 0$
- $Z_{\text{min}}(i) > 0$ かつ $B_i \in F$ ならば $x_{B_i} = 1$
- $Z_{\text{max}}(i) - \max\{0, y_{ij}\} < d_{B_i}$ かつ $j \in N_{\text{down}} \cap F$ ならば, $x_j = 0$
- $Z_{\text{max}}(i) + \min\{0, y_{ij}\} < d_{B_i}$ かつ $j \in N_{\text{up}} \cap F$ ならば, $x_j = 1$
- $Z_{\text{min}}(i) - \min\{0, y_{ij}\} > u_{B_i}$ かつ $j \in N_{\text{down}} \cap F$ ならば, $x_j = 0$
- $Z_{\text{min}}(i) + \max\{0, y_{ij}\} > u_{B_i}$ かつ $j \in N_{\text{up}} \cap F$ ならば, $x_j = 1$
- $|\bar{c}_j| \geq z^* - g(P_k)$ かつ $j \in N_{\text{down}} \cap F$ ならば, $x_j = 0$
- $|\bar{c}_j| \geq z^* - g(P_k)$ かつ $j \in N_{\text{up}} \cap F$ ならば, $x_j = 1$

なお、変数が固定されるごとに F, S^0, S^1 などは更新していく。計算の結果、すべての自由変数が固定された場合には、部分問題 P_k が解けたことになり、 P_k を PLIST から除き、 $L \leftarrow L - 1$ とする。このとき $f(P_k) < z^*$ ならば、 P_k の解と $f(P_k)$ で x^* と z^* を置き換える。それから更新された PLIST により、処理 (6) に移る。もし、固定されない自由変数が残っている場合には、次の分枝操作に移る。

(5) 分枝操作

PLIST から P_k を除く. そして上記の部分問題のテストで固定されなかった自由変数から分枝変数 x_{j_0} を一つ選び, j_0 を S^0 に追加した部分問題 P_{k_0} と j_0 を S^1 に追加した部分問題 P_{k_1} を PLIST の最後に追加して $L \leftarrow L+1$ とする. ここで分枝変数 x_{j_0} は以下のようにして決定する.

暫定解 x^* が求まっていない場合:

$|h_{up}(j)(1 - \bar{x}_j) - h_{down}(j)\bar{x}_j|$ を最大にする $j \in F \cap B$ を j_0 とする. ここで \bar{x} は部分問題 P_k の緩和問題の最適解である.

暫定解 x^* が求まっている場合:

$\min\{h_{up}(j)(1 - \bar{x}_j), h_{down}(j)\bar{x}_j\}$ を最大にする $j \in F \cap B$ を j_0 とする.

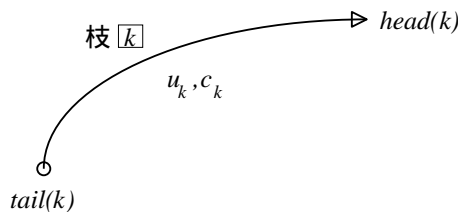
(6) 終了条件

もし, $L = 0$ になっていれば, 分枝限定法の処理は終了し, そのときの暫定解 x^* が P_1 すなわち元の混合 0-1 計画問題の解になる. そうでない場合には, ふたたび (2) 以降の処理を繰り返す.

5.1.2.6 ネットワーク上の流れに対する費用の最小化

n 個の節点と m 本の枝をもつネットワークを考える. ただしループ (始点と終点と同じ節点である枝) は存在しないと仮定する. 有向枝 k はその始点 $tail(k)$ と終点 $head(k)$ で表され, 非負の容量 u_k と単位流量あたりの費用係数 c_k をもつ. 最小費用流問題はこのようなネットワーク上で, 節点 i の流入出量 b_i および枝 k の容量 u_k の制約を満たし,

図 5-2 枝 k とその容量 u_k および費用係数 c_k

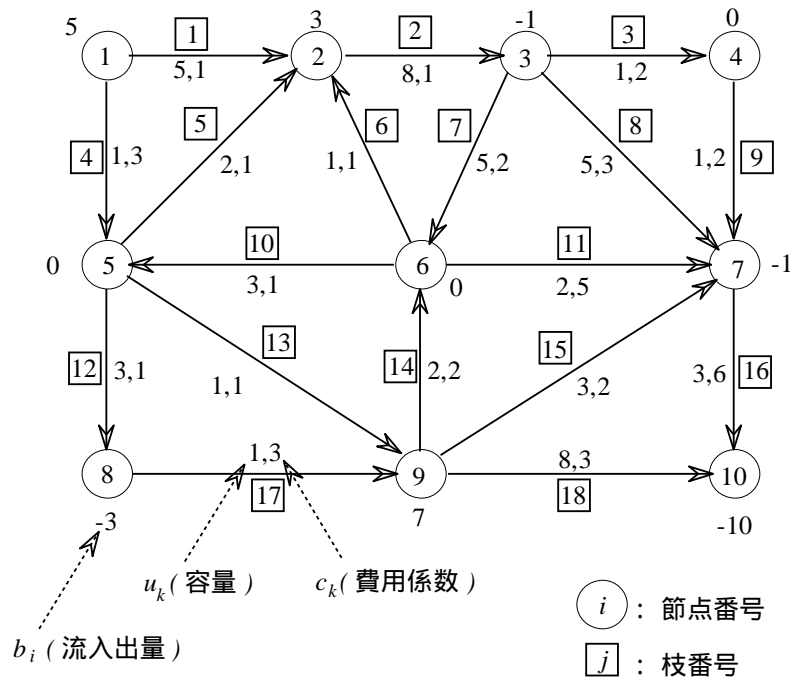


かつ全枝の費用の和を最小にする非負の流れ x_k ($k = 1, \dots, m$) を求める問題である.

$$\begin{aligned} \text{目的関数} &: \sum_{k=1}^m c_k x_k \longrightarrow \text{最小} \\ \text{制約条件} &: \sum_{tail(k)=i} x_k - \sum_{head(k)=i} x_k = b_i, \quad (i = 1, \dots, n) \\ &: 0 \leq x_k \leq u_k, \quad (k = 1, \dots, m) \\ &: \sum_{i=1}^n b_i = 0 \end{aligned}$$

これは線形計画問題でありシンプレックス法を適用できるが, 制約条件の特殊性を生かしてタブローをグラフ的に表現するデータ構造が可能でありピボット演算にともなう計算を効率よく実行できる.

図 5-3 ネットワークの例



(1) 初期実行可能解

ネットワークに節点 $n + 1$ を追加し, $b_i \geq 0$ なる節点 i からは枝 $k = (i, n + 1)$ を, $b_i < 0$ なる節点 i へは枝 $k = (n + 1, i)$ を作る. ただし, 追加枝の費用係数は十分大きな正の値 L , 容量は ∞ (具体的には十分大きな正の値 U), $b_{n+1} = 0$ と定める.

そのとき修正後の最小費用流問題は以下のようなになる.

$$\begin{aligned} \text{目的関数} &: \sum_{k=1}^{m'} c_k x_k \longrightarrow \text{最小} \\ \text{制約条件} &: \sum_{\text{tail}(k)=i} x_k - \sum_{\text{head}(k)=i} x_k + \text{sign}(b_i) \cdot x_{m+i} = b_i, \quad (i = 1, \dots, n) \\ &: 0 \leq x_k \leq u_k, \quad (k = 1, \dots, m' (= m + n)) \\ &: \sum_{i=1}^n b_i = 0 \end{aligned}$$

ただし,

$$\begin{aligned} c_k &= L, \quad k = m + 1, \dots, m' \\ u_k &= U \\ \text{sign}(b_i) &= \begin{cases} 1, & b_i \geq 0 \\ -1, & b_i < 0 \end{cases} \end{aligned}$$

これより, 初期実行可能解は

$$x_k = \begin{cases} 0 & (k = 1, \dots, m) \\ \text{sign}(b_{k-m}) \cdot b_{k-m} & (k = m + 1, \dots, m') \end{cases}$$

とすればよい.

(2) ピボット列の選択とピボット演算

A_k は制約条件の係数行列の第 k 列, B は係数行列から適当な $n \times n$ 正則部分行列を選んで定まった基底行列とすると, 非基底変数 x_k に対して

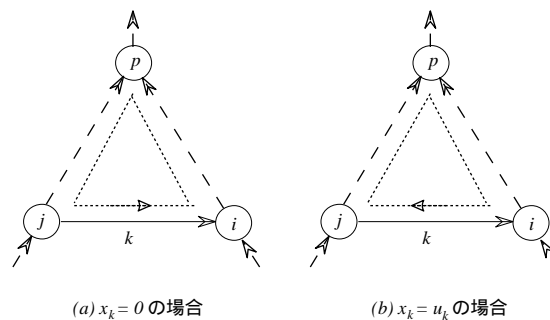
$$\bar{c}_k = c_k - c_B^T B^{-1} A_k$$

を求め,

$$\begin{aligned} \bar{c}_k < 0, \quad x_k = 0 \text{ の場合} \\ \bar{c}_k > 0, \quad x_k = u_k \text{ の場合} \end{aligned} \tag{5.3}$$

を満たすものを一つピボット列に選択する. ピボット列 k が定まったとき, x_k の変化の上限 Δ は, 枝 k の端点 i と j の共通の祖先のうち深さ最大のもの p を通る閉路 $j \rightarrow i \rightarrow p \rightarrow j$ に沿って環流させるとき, 実現可能な流量の最大値に等しい.

図 5-4 流量 Δ の環流



この閉路中の各枝 l について次のように定まる Δ_l のうちの最小値を Δ とすればよい.

- 枝 k では $\Delta_k = u_k$
- i から p への路上の枝 l の方向が下から上ならば $\Delta_l = u_l - x_l$, 上から下ならば $\Delta_l = x_l$
- j から p への路上の枝 l の方向が下から上ならば $\Delta_l = x_l$, 上から下ならば $\Delta_l = u_l - x_l$

次にこの閉路に沿って流量 Δ を環流させる. すなわち閉路中の各枝 l について

$$x_l := \begin{cases} x_l + \Delta, & \text{枝 } l \text{ と閉路の向きが一致} \\ x_l - \Delta, & \text{枝 } l \text{ と閉路の向きが逆} \end{cases}$$

とし閉路以外の枝の流量は変化しない. これがピボット演算である. また, 閉路中 $\Delta_l = \Delta$ を満たしていた枝 l の新しい流量 $x_l = 0$ あるいは u_l を新しい非基底変数 x_r とする (複数個存在すればその一つを任意に選ぶ). 以上の処理を繰り返し, 式 (5.3) を満たすものがなくなったとき処理を終了する.

5.1.2.7 プロジェクトの日程計画に対する費用の最小化

複数の作業からなるプロジェクトを考える. 各作業はその先行作業をいくつかもち, それらがすべて完了していなければ開始できない. プロジェクトネットワークは, 各作業を有向枝で表し, 節点を介することで先行関係を示す.

各作業 k は通常作業時間 $t_N(k)$ と特急作業時間 $t_C(k)$ をもち

$$t_N(k) \geq t_C(k)$$

の関係がある. また, 作業 k の遂行にかかる通常作業費用 $c_N(k)$ と特急作業費用 $c_C(k)$ には

$$c_N(k) \leq c_C(k)$$

の関係がある. $t_N(k) \geq t_C(k)$ のとき, 作業 k を中間の作業時間 $t(k)$ で遂行する場合の作業費用 $c(k)$ は,

$$c(k) = a(k) - b(k) \times t(k)$$

ただし,

$$a(k) = \frac{c_C(k) \times t_N(k) - c_N(k) \times t_C(k)}{t_N(k) - t_C(k)}, \quad b(k) = \frac{c_C(k) - c_N(k)}{t_N(k) - t_C(k)}$$

で与えられると考える.

ここで, 各作業 k を作業時間 $t(k)$ で行った場合, 各節点 i に対し, i に入る作業が全て完了し, i から出る枝の作業にいつでもかかれるという時刻 (最早節点時刻) $E(i)$ は,

$$E(1) = 0$$

$$E(i) = \max \{E(\text{tail}(k)) + t(k) \mid \text{head}(k) = i\}, \quad i = 2, 3, \dots, n$$

で計算できる. ここで, n は節点数, $\text{tail}(k)$ は作業 k の始点の節点, $\text{head}(k)$ は作業 k の終点の節点を表す. またプロジェクトの完了時刻 T は

$$T = E(n)$$

で与えられる.

また, このプロジェクトを時刻 T に完了するために, 各節点 i に入る作業を遅くとも完了しなければならない時刻 (最遅節点時刻) $L(i)$ は,

$$L(n) = T$$

$$L(i) = \min \{L(\text{head}(k)) - t(k) \mid \text{tail}(k) = i\}, \quad i = n-1, n-2, \dots, 1$$

で計算できる.

具体的な日程計画を立案するには, まず予定完了時刻 T_S を設定する. 通常

$$T_C \leq T_S \leq T_N$$

の範囲におく. 予定完了時刻 T_S までにプロジェクトを最小費用で完了する問題は, 作業 k の所要時間 $t(k)$ と節点 i の節点時刻 $\tau(i)$ を変数として, 次の線形計画問題に定式化できる.

$$\text{目的関数} : \sum_{k=1}^n (a(k) - b(k) \times t(k)) \rightarrow \text{最小}$$

$$\text{制約条件} : t(k) + \tau(\text{tail}(k)) - \tau(\text{head}(k)) \leq 0, \quad k = 1, 2, \dots, n$$

$$\tau(1) = 0$$

$$\tau(n) \leq T_S$$

$$t(k) \leq t_N(k), \quad k = 1, 2, \dots, n$$

$$-t(k) \leq -t_C(k), \quad k = 1, 2, \dots, n$$

節点時刻 $\tau(i)$ が求まると各作業 k の所要時間 $t(k)$ を求めることができる.

$$t(k) = \min \{t_N(k), \tau(\text{head}(k)) - \tau(\text{tail}(k))\}$$

とすればよい. さらに, 各節点 i の最早節点時刻 $E(i)$ と最遅節点時刻 $L(i)$ を求めることができる.

このとき, 作業 k にとりかかれる最も早い時刻 (作業 k の最早開始時刻) $ES(k)$ およびプロジェクトを T_S までに完了するために遅くともとりかからなければならない時刻 (作業 k の最遅開始時刻) $LS(k)$ はそれぞれ

$$ES(k) = E(\text{tail}(k))$$

$$LS(k) = L(\text{head}(k)) - t(k)$$

で求められる。また、作業 k の開始を遅らせても残りの作業の日程を調整すればプロジェクトを T_S 以内に完了できる範囲 (総余裕時間) $TF(k)$ および総余裕時間のうち作業 k の開始を遅らせてもその後の作業計画に影響を与えない範囲 (自由余裕時間) $FF(k)$ はそれぞれ

$$TF(k) = LS(k) - ES(k)$$

$$FF(k) = E(head(k)) - E(tail(k)) - t(k)$$

で求められる。

5.1.2.8 供給地から需要地への輸送費用の最小化

輸送問題は線形計画問題の特殊な問題である。これは、物の移動に伴う問題である品物を、いくつかの供給地よりいくつかの需要地に最小の輸送費用で運搬するには、どんなルートがあるのか、またその時の費用はどのくらいかを見いだす問題である。この輸送問題を解くには、第1次近似解で最初の計画を定め、それを改善して最終計画 (最適解) に行き着く。本ライブラリでは、第1次近似解を求める方法として北西隅の規則、ハウサッカー法、改善の方法には、改訂シンプレックス法 (使用しているアルゴリズム (5.1.2.4) 参照) を用いる。ここでは、供給地 i ($i = 1, \dots, m$) での供給量を a_i 、需要地 j ($j = 1, \dots, n$) での需要量を b_j とし、供給地 i から需要地 j への輸送量を x_{ij} とすると、制約条件

$$\sum_{j=1}^n x_{ij} = a_i \quad (i = 1, \dots, m)$$

$$\sum_{i=1}^m x_{ij} = b_j \quad (j = 1, \dots, n)$$

$$x_{ij} \geq 0 \quad (\text{すべての } i, j \text{ に対して})$$

に従って目的関数である総輸送費用は、次の関数を最小にする x_{ij} を求めることで得られる。

$$Z = \sum_{j=1}^n \sum_{i=1}^m c_{ij} x_{ij}$$

(1) 第1次近似解法 (北西隅の規則, ハウサッカー法)

北西隅の規則は与えられた単位量の輸送費と供給量および需要量の行列の左上から、需要量と供給量の資源を比較し、その量の最小をつぎつぎと配分していく方法である。ハウサッカー法は、各供給地から各需要地への単位輸送費用の最小に資源を多量に配分して、つぎつぎと配分していく方法である。

(2) 不釣り合いな輸送問題

ある輸送問題において、総供給量 $\sum a_i$ が総需要量 $\sum b_j$ より少ない場合、全体の需要量を満足できないにしても総出荷費用を最小にする方法で、需要地へ供給地よりある量を配分することができる。この場合は、 $\sum b_j - \sum a_i$ の総量を取り扱う架空の供給地を仮定する。この架空の供給地と需要地との間の1単位出荷する費用は零とする。もし、原形の問題が $NS \times ND$ の問題ならば、 $(NS+1) \times ND$ の問題を取りあげる輸送問題と同様にこの問題を解くことになる。もし問題が、総供給量が総需要量より多い場合、これと同様に架空の問題を組み立てる。この場合は $\sum a_i - \sum b_j$ を満たす架空の需要地を仮定し、この架空の需要地の出荷費を零とする。もし、原形の問題が $NS \times ND$ の問題ならば、 $NS \times (ND+1)$ の問題を取りあげる輸送問題と同様にこの問題を解くことになる。

5.1.2.9 制約付き多変数凸型2次関数の最小化 (線形制約)

ここでは n 変数の目的関数

$$M(x) = c^T x + \frac{1}{2} x^T G x$$

を制約条件

$$\begin{aligned} \mathbf{a}_i^T \mathbf{x} &= b_i & i = 1, 2, \dots, m_e \\ \mathbf{a}_i^T \mathbf{x} &\geq b_i & i = m_e + 1, \dots, m \end{aligned}$$

の下で最小にする問題 (2 次計画問題) を扱う。ここで \mathbf{a}_i と c は n 次元列ベクトル, b_i は実数, G は $n \times n$ 正定値対称行列である。この問題を解くために本ライブラリでは GI 法を用いる。これは D. Goldfarb と A. Idnani によって提案されたものである。

制約条件の添字の内のいくつかからなる集合を J とし, J の要素に対応するすべての制約条件の下で目的関数 $M(\mathbf{x})$ を最小にする \mathbf{x} を J -最適解と呼び, 元の問題の解を単に最適解と呼ぶことにする。また, J の要素の個数を $|J|$, \mathbf{a}_i ($i \in J$) を列に持つ $n \times |J|$ 行列を A_J と表記する。さらに K をすべての不等号制約条件の添字の集合, E をすべての等号制約条件の添字の集合とする。したがって, $J = E \cup K$ のとき J -最適解は最適解そのものである。最適解を求める手順としてまず最初に $J = E$ とし, このときの J -最適解を初期解として, J に満たすべき不等号制約条件を 1 つずつ付け加えていき, 付け加えた制約条件を満たすように J -最適解を補正していく。最終的に $J = E \cup K$ となるまで補正を繰り返す。

(1) 初期解の計算

$J = E$ のときの J -最適解を求める。目的関数が凸型であるとき, 最適解を求める問題は Kuhn-Tucker 条件

$$\begin{aligned} G\mathbf{x} - A_J \mathbf{v}_J &= -\mathbf{c} \\ A_J^T \mathbf{x} &= \mathbf{b}_J \\ \mathbf{v}_i &\geq 0 \quad (i \in J) \end{aligned}$$

を満たす \mathbf{x} と \mathbf{v}_J を求める事と等価である。ここで \mathbf{v}_J と \mathbf{b}_J はそれぞれ v_i ($i \in J$) と b_i ($i \in J$) を成分とする $|J|$ 次元のベクトルである。いま

$$\begin{aligned} A_J^\dagger &= (A_J^T G^{-1} A_J)^{-1} A_J^T G^{-1} \\ H_J &= G^{-1} (I - A_J A_J^\dagger) \end{aligned}$$

と置くと, \mathbf{x} と \mathbf{v}_J は

$$\begin{aligned} \mathbf{x} &= (A_J^\dagger)^T \mathbf{b}_J - H_J \mathbf{c} \\ \mathbf{v}_J &= (A_J^T G^{-1} A_J)^{-1} \mathbf{b}_J + A_J^\dagger \mathbf{c} \end{aligned}$$

として, 計算される。

(2) J の更新

ある $J \subset K$ について J -最適解が求められており, しかもそれが最適解でないとき, 次のようにして J に付け加える制約条件を決定する。まず

$$c_i(\mathbf{x}) = \mathbf{a}_i^T \mathbf{x} - b_i$$

とすると, $c_s(\mathbf{x}) < 0$ を満たす $s \notin J$ が少なくとも 1 つ存在する。そこで

$$c_s(\mathbf{x}) = \min \{c_i(\mathbf{x}) | i \notin J\} < 0$$

により s を決定する。そして, \mathbf{a}_s が A_J の列を成す \mathbf{a}_i ($i \in J$) と独立ならば $J \cup \{s\}$ を新たな J とする。また, そうでないときは J から, 要素を 1 つ取り除いて J -最適解の計算をやり直す。

(3) J -最適解の更新

J に (b) で決定した不等号制約条件

$$\mathbf{a}_s^T \mathbf{x} - b_s \geq 0$$

を付け加えて J を更新した後の J -最適解を次のようにして求める. \mathbf{x} が J を更新する前の J -最適解であり, かつ $c_i = 0$ ($i \in J$) であるとする. このとき $\bar{\mathbf{x}} = \mathbf{x} + tH_J\mathbf{a}_s$ と置くと (b) で定義した c_i ($i \in J$) と c_s および v_i ($i \in J$) について

$$c_i(\bar{\mathbf{x}}) = c_i(\mathbf{x}) = 0 \quad (i \in J)$$

$$c_s(\bar{\mathbf{x}}) = c_s(\mathbf{x}) + t\mathbf{a}_s^T H_J \mathbf{a}_s$$

$$v_i(\bar{\mathbf{x}}) = v_i(\mathbf{x}) - tr_i$$

が成り立つ. ただし, r_i ($i \in J$) は $|J|$ 次元ベクトル $\mathbf{r} = A_J^T \mathbf{a}_s$ の第 i 成分である. いま

$$t_1 = \min \left\{ \frac{v_i(\mathbf{x})}{r_i} \mid r_i > 0, i \in J \cap K \right\}$$

$$t_2 = -\frac{c_s(\mathbf{x})}{\mathbf{a}_s^T H_J \mathbf{a}_s}$$

とすると, $0 \leq t \leq t_1$ の範囲では

$$v_i(\bar{\mathbf{x}}) \geq 0 \quad (i \in \bar{J} \cup K)$$

であり, また $t = t_2$ のとき

$$c_s(\bar{\mathbf{x}}) = 0$$

である. そこで, もし $t_1 \geq t_2$ ならば $t = t_2$ のときの $\bar{\mathbf{x}}$ は $J \cup \{s\}$ に対応する制約条件について (a) で使用した Kuhn-Tucker 条件を満たしており, これが J 更新後の J -最適解となる. 一方, $t_1 < t_2$ ならば J から要素を 1 つ取り除いて J -最適解の計算をやり直す.

(4) 終了条件

J -最適解 \mathbf{x} がすべての $i \in \bar{J} \cap K$ について

$$c_i(\mathbf{x}) \geq 0$$

を満たしているとき, 最適解は \mathbf{x} であるものとして, 計算を終了する.

5.1.2.10 多変数広義凸型 2 次関数の最小化 (線形制約)

ここでは, 制約条件

$$\sum_{j=1}^n a_{ij}x_j = b_i \quad (i = 1, \dots, m_e)$$

$$\sum_{j=1}^n a_{ij}x_j \geq b_i \quad (i = m_e + 1, \dots, m)$$

$$x_i \geq 0 \quad (i = 1, \dots, n)$$

の下で目的関数である n 変数の広義凸型 2 次関数

$$f(x) = \frac{1}{2}x^T Gx + c^T x \quad (G : \text{半正定値対称行列})$$

を最小にする x^* とそのときの関数値 $f(x^*)$ を求める問題を扱う。この問題を解くために本ライブラリでは与えられた問題をそれと等価な線形相補性問題に変換し、それをレムケ法を用いて解く。

以下に具体的な手順を示す。

(1) 線形相補性問題の作成

まず、等式制約条件を用いて m_e 個の変数を消去し、元の 2 次計画問題を残りの変数についての 2 次計画問題に変換する。そして新しい 2 次計画問題に対して、等価な線形相補性問題を作成する。 $x = [x_1, x_2, \dots, x_n]$ から等式制約条件を用いて、 x_1, x_2, \dots, x_{m_e} を x_{m_e+1}, \dots, x_n で

$$x_i = \sum_{j=m_e+1}^n p_{ij}x_j + r_i \quad (i = 1, \dots, m_e)$$

のように表わす。この式を用いて、与えられた 2 次計画問題を $n - m_e$ 個の変数 $x' = [x_{m_e+1}, \dots, x_n] = [x'_1, \dots, x'_{n-m_e}]$ についての 2 次計画問題に変換する。

$$\begin{aligned} \text{制約条件: } \sum_{j=1}^{n-m_e} a'_{ij}x'_j &\geq b'_i \quad (i = 1, \dots, n - m_e) \\ x'_i &\geq 0 \quad (i = 1, \dots, n - m_e) \\ \text{目的関数: } f(x') &= \frac{1}{2}(x')^T G'x' + c'^T x' \end{aligned}$$

ここで

$$\begin{aligned} (G')_{i-m_e, j-m_e} &= (G)_{i,j} + \sum_{k=1}^{m_e} G_{i,k}p_{k,j} + \sum_{k=1}^{m_e} G_{k,j}p_{k,i} + \sum_{s=1}^{m_e} \sum_{t=1}^{m_e} p_{s,i}G_{s,t}p_{t,j} \\ c'_{i-m_e} &= c_i + \sum_{j=1}^{m_e} c_j p_{j,i} + \sum_{k=1}^{m_e} G_{i,k}r_k + \sum_{s=1}^{m_e} \sum_{t=1}^{m_e} G_{s,t}p_{s,i}r_t \\ a'_{i,j-ME} &= p_{i,j} \quad (i = 1, \dots, m_e; j = m_e + 1, \dots, n) \\ a'_{i,j-m_e} &= a_{i,j} + \sum_{k=1}^{m_e} a_{i,k}p_{k,j} \quad (i = m_e + 1, \dots, m; j = m_e + 1, \dots, n) \\ b'_i &= -r_i \quad (i = 1, \dots, m_e) \\ b'_i &= b_i - \sum_{k=1}^{m_e} a_{ik}r_k \quad (i = m_e + 1, \dots, m) \end{aligned}$$

とし、 $A' = (a'_{i,j})$, $G' = (G'_{i,j})$, $(c')^T = [c'_1, \dots, c'_{n-m_e}]$, $(b')^T = [b'_1, \dots, b'_{n-m_e}]$ と置くことで以下の線形相補性問題を作成する。ここで v は不等式制約条件に対するラグランジュ乗数ベクトルである。

$$\begin{aligned} \begin{bmatrix} y \\ w \end{bmatrix} &= \begin{bmatrix} G' & -A'^T \\ A' & 0 \end{bmatrix} \begin{bmatrix} x \\ v \end{bmatrix} + \begin{bmatrix} c' \\ -b' \end{bmatrix} \\ y_i &\geq 0 \quad \text{for } (i = 1, \dots, n - m_e) \\ w_i &\geq 0 \quad \text{for } (i = 1, \dots, m) \\ x_i &\geq 0 \quad \text{for } (i = 1, \dots, n - m_e) \\ v_i &\geq 0 \quad \text{for } (i = 1, \dots, m) \\ [y^T \ w^T] \begin{bmatrix} x \\ v \end{bmatrix} &= 0 \end{aligned}$$

(2) 線形相補性問題

線形相補性問題を解いて、等式制約条件を消去して得られた新しい2次計画問題の解を求める。

与えられた $n \times n$ 行列

$$T = \begin{bmatrix} G' & -A^T \\ A & 0 \end{bmatrix}$$

に対して、

$$\mathbf{y} = T\mathbf{x} + \mathbf{q} \tag{5.4}$$

$$x_i \geq 0 \quad (i = 1, \dots, n) \tag{5.5}$$

$$y_i \geq 0 \quad (i = 1, \dots, n) \tag{5.6}$$

を満たす、 n 次元ベクトル \mathbf{x} , \mathbf{y} を求める。まず線形相補性問題 (5.4), (5.5), (5.6) に対し、 $2n + 1$ 個の変数 $(\mathbf{y}, \mathbf{x}, \xi)$ についての方程式

$$\mathbf{y} - T\mathbf{x} - d\xi = \mathbf{q}$$

を定義する。ここで、 d は全成分が正である n 次元定数ベクトルである。 $(\mathbf{y}, \mathbf{x}, \xi)$ のうち、0でない成分を基底変数、それ以外を非基底変数と呼ぶ。

次に (5.4) の方程式に対して初期解を $(\mathbf{y}, \mathbf{x}, \xi) = (\mathbf{q} + d\xi_0, \mathbf{0}, \xi_0)$ とする。ここで

$$\xi_0 = \max\{-q_i/d_i | i = 1, 2, \dots, n\}$$

$$\eta = \max\{-q_i/d_i | i = 1, 2, \dots, n\}$$

とする。式 (5.5), (5.6) を満たしたまま η の値を無限に大きくできるならば計算を終了する。ある基底変数の値が0になった時、この時の基底変数が y_s ならば、 y_s を基底から出して、 η を基底に入れ、 $\eta = x_s$ とする。基底変数が x_s ならば、 x_s を基底から出して、 η を基底に入れ、 $\eta = y_s$ とする。この処理を繰り返し、その基底変数が ξ になったところで計算を終了する。また、式 (5.5), (5.6) を満たしたまま η の値を無限に大きくできる場合には解が存在しないものとして計算を終了する。

(3) 解の変換

求めた解を等式制約条件を消去する前の元の2次計画問題の解に変換する。

5.1.2.11 制約無し0-1多変数2次関数の最小化

すべての変数が0-1変数すなわち0または1のみを値として取るという条件の下で2次関数を最小化する問題(0-1無制約2次計画問題)を扱う。

$$\text{目的関数} \quad : \quad f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T G\mathbf{x} + \mathbf{c}^T \mathbf{x} \rightarrow \min$$

$$\text{0-1変数条件} \quad : \quad x_j = 0 \text{ or } 1 \quad (j = 1, \dots, n)$$

本ライブラリでは、0-1無制約2次計画問題を解くために、分枝限定法を用いる。

0-1無制約2次計画問題では以下のような部分問題を用いる。

$$\text{目的関数} \quad : \quad f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T G\mathbf{x} + \mathbf{c}^T \mathbf{x} \rightarrow \min$$

$$\text{0-1変数条件} \quad : \quad x_j = 0 \quad (j \in S^0)$$

$$x_j = 1 \quad (j \in S^1)$$

$$x_j = 0 \text{ or } 1 \quad (j \in F)$$

ここで S^0, S^1, F はそれぞれ部分問題 P_k において, 0 に固定された 0-1 変数の添字の集合, 1 に固定された 0-1 変数の添字の集合, ならびに自由変数の添字の集合である. 部分問題は, それ自体 0-1 無制約 2 次計画問題である. とくに $S^0 \cup S^1 = \phi$ すなわち固定変数を持たない部分問題は元の 0-1 無制約 2 次計画問題 (以後, P_1 と呼ぶ.) そのものである.

また, 部分問題に対する緩和問題を以下のように定義する.

$$\begin{aligned} \text{目的関数} & : f(x) = \frac{1}{2}x^T Gx + c^T x \rightarrow \min \\ \text{0-1 変数条件} & : x_j = 0 \quad (j \in S^0) \\ & \quad x_j = 1 \quad (j \in S^1) \\ & \quad 0 \leq x_j \leq 1 \quad (j \in F) \end{aligned}$$

以下に分枝限定法により P_1 の解を求める方法について説明する.

(1) 初期設定

(a) 係数行列の前処理

後述する緩和問題を解くためには, 目的関数の係数行列が正定値行列である必要がある. 0-1 無制約 2 次計画問題では 0-1 変数が

$$x_i^2 = x_i \quad (i = 1, \dots, n)$$

という性質を持つことを利用して, 目的関数を対称行列を係数に持つ以下のような 2 次関数に書き換えることが出来る.

$$f(x) = \frac{1}{2}x^T G'x$$

ただし,

$$G' = \frac{1}{2}(G + G^T) + 2\text{diag}(c_1, \dots, c_n)$$

ここで, G' が半正定値であれば, P_1 は自明な最適解

$$x^* = (0, 0, \dots, 0)^T$$

を持つ. G' が半正定値でない, すなわち負の固有値を持つ場合には λ_{\min} を G' の最小の固有値, $\beta (> 0)$ を正数のパラメータとして

$$G'' = G' + (|\lambda_{\min}| + \beta)\text{diag}(1, 1, \dots, 1)$$

$$c'' = -\frac{1}{2}(|\lambda_{\min}| + \beta)(1, 1, \dots, 1)^T$$

と定義することにより, 目的関数を正値対称行列 G'' を係数に持つ 2 次関数

$$f(x) = \frac{1}{2}x^T G''x + (c'')^T x$$

に書き換えることが出来る. ここで β は行列 G'' の最小の固有値となっている. なお, 以降の説明では係数行列 G はあらかじめ, このような前処理を施して, 正値対称行列に変換されているものとする.

(b) 部分問題のリスト

分枝限定法では部分問題のリスト PLIST を使用する. 初期状態では PLIST のメンバは P_1 だけで, PLIST のメンバ数 L は 1 である. L は分枝限定法の計算が進むにしたがって, 増減する.

(c) 暫定解

混合 0-1 計画問題の場合と異なり, 実行可能解の存在は自明であるが, なるべく目的関数の値の小さな解を暫定解として用いる方が分枝限定法による最適解の探索の効率が良い. 本ライブラリでは模擬焼なまし法およびタブー探索という 2 つの発見的探索の方法により得た解を初期状態の暫定解として用いる.

(d) 模擬焼なまし法

- ① 初期解 $x = x_0$ をランダムに選ぶ.
- ② 初期温度を $T = T_0$ に設定する.
- ③ $z \leftarrow f(x_0)$ とする.
- ④ $y \leftarrow f(x)$ とする.
- ⑤ 温度パラメータを $T \leftarrow \alpha \times T$ とする. ($0 < \alpha \leq 1$)
- ⑥ 添字 $i (1 \leq i \leq n)$ をランダムに一つ選ぶ.
- ⑦ $x_i \leftarrow 1 - x_i$ とする.
- ⑧ $y < f(x)$ ならば, 実数 $r \in [0, 1]$ をランダムに選び $r > \exp(-1/T)$ の場合には $x_i \leftarrow 1 - x_i$ とする.
- ⑨ $z \leftarrow \min\{y, z\}$ とする.
- ⑩ 所定の回数だけ, 処理④から⑨を繰り返し, 最終的に得られた z を分枝限定法における初期暫定解の目的関数値とする.

(e) タブー探索

- ① 初期解 $x = x_0$ をランダムに選ぶ.
- ② タブーリストを $K = \phi$ に設定する.
- ③ $z \leftarrow f(x_0)$ とする.
- ④ $U(x) = \{u^{(1)}, u^{(2)}, \dots, u^{(n)}\}$ $u^{(i)} = (x_1, \dots, x_{i-1}, 1 - x_i, x_{i+1}, \dots, x_n)^T$ とする.
- ⑤ $x \leftarrow \min_{U \setminus K} u_i$ とする.
- ⑥ $K \leftarrow K \cup \{x\}$ とする. K の要素数が所定の数を超えた場合は最も以前に K に加えられた要素を K から取り除く.
- ⑦ $z \leftarrow \min\{f(x), z\}$ とする.
- ⑧ 所定の回数だけ, 処理④から⑦を繰り返し, 最終的に得られた z を分枝限定法における初期暫定解の目的関数値とする.

タブー探索は計算時間がかかるが良好な解が得られ, 模擬焼なまし法は計算時間が短い. これらの探索方法に加えて, 本ライブラリでは模擬焼なまし法により第 1 次の解を計算したあとで, それをタブー探索により改良する方法もサポートしている. なお, 変数の数 n が大きい場合, 後述する分枝限定法で厳密な最適解を実用的な計算時間で求めるのは困難である. しかし, 多くの場合, 発見的探索で得られた解は十分な近似精度を持っている. そこで, 発見的探索で得られた解を近似解としてそのまま利用するため分枝限定法の計算を省略する方法も本ライブラリではサポートしている.

(2) 下界値の計算

各部分問題について対応する緩和問題を用いて最適値の下界値を以下のように計算する.

- (a) 係数行列 G の固有値を $\lambda_1, \dots, \lambda_n$ 対応する正規直交化された固有ベクトルを $u^{(1)}, \dots, u^{(n)}$ とする.
- (b) 緩和問題である凸型 2 次計画問題を G-I 法を用いて解き, その最適解を \bar{x} とする.
- (c) 自由変数 $x_i (i \in F)$ が

$$f(\bar{x}) + \frac{1}{2} \left| \frac{\bar{x}_i^2}{\sum_{k=1}^n \frac{(u_i^{(k)})^2}{\lambda_k}} \right| \leq z^* < f(\bar{x}) + \frac{1}{2} \left| \frac{(1 - \bar{x}_i)^2}{\sum_{k=1}^n \frac{(u_i^{(k)})^2}{\lambda_k}} \right|$$

を満たせば, x_i は 0 に固定される. このとき, $S_0 \leftarrow S_0 \cup \{i\}, F \leftarrow F - \{i\}$ とする.

また, 自由変数 $x_i (i \in F)$ が

$$f(\bar{x}) + \frac{1}{2} \left| \frac{(1 - \bar{x}_i)^2}{\sum_{k=1}^n \frac{(u_i^{(k)})^2}{\lambda_k}} \right| \leq z^* < f(\bar{x}) + \frac{1}{2} \left| \frac{\bar{x}_i^2}{\sum_{k=1}^n \frac{(u_i^{(k)})^2}{\lambda_k}} \right|$$

を満たせば, x_i は 1 に固定される. このとき, $S_1 \leftarrow S_1 \cup \{i\}, F \leftarrow F - \{i\}$ とする.

このような操作により, 自由変数のいくつかが固定された場合には, 緩和問題を解き直し, それを新たに固定される自由変数が無くなるまで繰り返す.

(d) 座標変換

$$x_i - \bar{x}_i \leftarrow \sum_{j=1}^n \frac{u_i^{(j)}}{\sqrt{\lambda_j}} \hat{x}_j$$

で定義される変数 \hat{x} の空間における点の集合 U_i^0 および U_i^1 を以下のように定義する.

$$U_i^0 = \left\{ \hat{x} \mid \sum_{j=1}^n \frac{u_i^{(j)}}{\sqrt{\lambda_j}} \hat{x}_j = 0 \right\}$$

$$U_i^1 = \left\{ \hat{x} \mid \sum_{j=1}^n \frac{u_i^{(j)}}{\sqrt{\lambda_j}} \hat{x}_j = 1 \right\}$$

任意の自由変数の添字の集合 $\{i_1, i_2, \dots, i_r\}$ について

$$D = \min \{ \|\hat{x}\| \mid \mathbf{x} \in \cap_{k=1}^r (U_{i_k}^0 \cup U_{i_k}^1) \}$$

とすると

$$g(\bar{\mathbf{x}}) = f(\bar{\mathbf{x}}) + \frac{D^2}{2}$$

は部分問題の下界値を与える. 本ライブラリでは, 自由変数のうちで

$$\frac{\min\{\bar{x}_i, 1 - \bar{x}_i\}^2}{\sum_{k=1}^n \frac{(u_i^{(k)})^2}{\lambda_k}}$$

の大きい方から r 個を D の計算に用いている.

部分問題 P_i に対応する最適値の下界値 $g(P_i)$ について $g(P_i) > z^*$ であった場合には, P_i を PLIST から除き, $L \leftarrow L - 1$ とする. また最適解を \bar{x} とすると, すべての $j \in F$ について \bar{x}_j の値が 0 または 1 であった場合には, 緩和問題の解が元の部分問題の解になり, 部分問題の最適値は緩和問題の最適値 $g(P_i)$ に等しい. このとき $f(P_i) < z^*$ ならば x^* と z^* を \bar{x} と $f(P_i)$ で置き換える.

なお, 緩和問題を新たに解く必要があるのは, 上記の初期設定のときと後述する分枝操作によって L が増加したときである. 後者の場合, PLIST の最後の 2 つの部分問題は分枝変数 x_{j_0} を 0 に固定した部分問題 P_{i_0} と 1 に固定した部分問題 P_{i_1} の組になっている.

(3) 分枝操作

PLIST から最後尾の部分問題 P_k を除く. そして自由変数から分枝変数 x_{j_0} を 1 つ選び, j_0 を S^0 に追加した部分問題 P_{k0} と j_0 を S^1 に追加した部分問題 P_{k1} を PLIST の最後に追加して $L \leftarrow L + 1$ とする. ここで分枝変数 x_{j_0} は以下のようにして決定する.

$$j_0 = \operatorname{argmin}_{j \in F} \left| \bar{x}_j - \frac{1}{2} \right|$$

分枝操作の後, 部分問題 P_{k0} および P_{k1} のそれぞれについて, 最適値の下界値を計算する.

(4) PLIST のソート

$K' = \min\{K, L\}$ とする. ここで, K は分枝限定法における探索の深さと呼ばれ, 前もって与えておく正の整数のパラメータである. PLIST の最後の K' 個の部分問題 P_i について, 下界値 $g(P_i)$ の降順に PLIST の最後の K' 個を並べ代える.

(5) 部分問題のテスト

PLIST の最後の部分問題 P_k についてそのすべての自由変数が固定された場合には、部分問題 P_k が解けたことになり、 P_k を PLIST から除き、 $L \leftarrow L - 1$ とする。このとき $f(P_k) < z^*$ ならば、 P_k の解と $f(P_k)$ で x^* と z^* を置き換える。それから更新された PLIST により、処理 (6) に移る。もし、固定されない自由変数が残っている場合には、次の分枝操作に移る。

(6) 終了条件

もし、 $L = 0$ になっていれば、分枝限定法の処理は終了し、そのときの暫定解 x^* が P_1 すなわち元の 0-1 無制約 2 次計画問題の解になる。そうでない場合には、ふたたび (2) 以降の処理を繰り返す。

5.1.2.12 制約付き多変数関数の最小化

ここでは n 変数の関数 $f(x)$ が与えられた時、これを与えられた $m + \ell$ 個の制約条件

$$\begin{cases} g_i(x) \leq 0 & (i = 1, \dots, m) \\ h_i(x) = 0 & (i = 1, \dots, \ell) \end{cases}$$

のもとで最小化する点 x^* (最適解) を求める問題を扱う。大域的な最小化は一般に困難であり、本ライブラリでは局所的な最小化のみを扱う。したがって、以下の説明において、最小化という用語は局所的な意味でのみ用いる。同様に局所的な最適解 x^* を単に最適解と称する。いま、ペナルティ関数 $\theta_\delta(x)$ を

$$\theta_\delta(x) = f(x) + \delta \max(0, g_1(x), \dots, g_m(x), \|h_1(x)\|, \dots, \|h_\ell(x)\|) \quad (0 < \delta)$$

で定義すると、

$$\theta_\delta(x) \geq f(x)$$

が任意の x について成り立ち、特に x が上記の制約条件を満たす点である時には

$$\theta_\delta(x) = f(x)$$

が成り立つ。したがって制約条件 (1), (2) の下で $f(x)$ を最小化する問題は、制約なしで $\theta_\delta(x)$ を最小化する問題に帰着される。 $\theta_\delta(x)$ は厳密なペナルティ関数であり、これに準ニュートン法を適用すれば、原理的には最適解を求めることができる。しかし、 $\theta_\delta(x)$ は数値的に不安定な性質を持っているため、本ライブラリでは逐次 2 次計画法と呼ばれるアルゴリズムを採用し、Lagrange 関数

$$L(x, \lambda, \mu) = f(x) + \sum_{i=1}^{m-m_e} \lambda_i g_i(x) + \sum_{i=1}^{m_e} \mu_i h_i(x)$$

に対して準ニュートン法に類似した解の反復改良を行う。 $\theta_\delta(x)$ は、直線探索の評価関数として使用する。計算の手順は以下の通りである。

(1) 部分 2 次計画問題

最適解 x^* の第 k 近似 x_k が求められている時、目的関数と制約条件を x_k のまわりで展開して得られる 2 次計画問題

目的関数 : $\frac{1}{2} d^T B_k d + \nabla f(x_k)^T d$

不等式制約条件 : $g_i(x_k) + \nabla g_i(x_k)^T d \leq 0 \quad (i = 1, \dots, m)$

等式制約条件 : $h_i(x_k) + \nabla h_i(x_k)^T d \leq 0 \quad (i = 1, \dots, \ell)$

を解き、その解を d_k とする。ただし、ヘッセ行列 $\nabla^2 f(x_k)$ そのものを計算するのは数値的に困難であるため、準ニュートン法の場合と同じく、その近似行列である B_k で置き換えている。この2次計画問題を解くためのアルゴリズムとしては、Goldfarb-Idnani 法を用いる。もし $d_k = 0$ であれば、 $x^* = x_k$ として計算を終了する。なおここで求められた Lagrange 乗数ベクトルを λ_{k+1}, μ_{k+1} とする。

(2) 直線探索

d_k を探索方向として x_{k+1} を探索する。 $\alpha_k = 1$ から出発して、条件

$$\theta_\delta(x_k + \alpha_k d_k) \leq \theta_\delta(x_k) - \omega \alpha_k d_k^T B_k d_k$$

を満たすならば (c) の処理に移り、もし満たさなければ $\alpha_k \rightarrow \tau \alpha_k$ なる置き換えを、条件が満たされるまで繰り返す。

(3) x_k の更新

$$x_{k+1} = x_k + \alpha_k d_k$$

(4) B_k の更新

オリジナルの BFGS 公式では B_k の正定値性が維持されないので、以下に示す修正 BFGS 公式により B_k を更新する。

$$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{\eta_k \eta_k^T}{s_k^T \eta_k}$$

ここで

$$s_k = x_{k+1} - x_k$$

$$y_k = \nabla_x L(x_{k+1}, \lambda_{k+1}, \mu_{k+1}) - \nabla_x L(x_k, \lambda_{k+1}, \mu_{k+1})$$

$$\phi = \begin{cases} 1 & : s_k^T y_k \geq 0.2 s_k^T \\ \frac{0.8 s_k^T B_k s_k}{s_k^T (B_k s_k - y_k)} & : \text{その他の場合} \end{cases}$$

$$\eta_k = \phi y_k + (1 - \phi) B_k s_k$$

(5) 更新の終了

x_k の更新は Karush-Kuhn-Tucker 条件

$$\begin{aligned} \nabla f(x) + \sum_{i=1}^m \lambda_i \nabla g(x) + \sum_{j=1}^{\ell} \mu_j \nabla h_j(x) &= \mathbf{0} \\ g_i(x) &= 0 \quad (i = 1, \dots, m) \\ h_j(x) &= 0 \quad (j = 1, \dots, \ell) \\ \sum_{i=1}^m \lambda_i g_i(x) &= 0 \\ \lambda_i &\geq 0 \quad (i = 1, \dots, m) \end{aligned}$$

が満たされるまで繰り返される。

5.1.2.13 ネットワーク上の2節点間の距離の最小化

(1) ある節点から他のすべての節点への最短経路の計算

Dijkstra の方法を用いてグラフ $G = (V, E)$ におけるある指定された節点 $init$ から他のすべての節点までの最短経路とその距離を求める。ただし、すべての枝の重みは非負であると仮定する。

- (i) 各節点 $i \in V$ に対して $Distance(i) = \infty$, $Path(i) = init$ および $P = \phi$ とする. 出発点 $init$ に対して $Distance(init) = 0$ とする. また, $next = init$ とする.
- (ii) $P = P \cup \{next\}$ とする. 節点 $next$ に接続している各節点 j に対して, もし $Distance(j) > Distance(next) + Weight(next, j)$ ならば, $Distance(j) = Distance(next) + Weight(next, j)$ および $Path(j) = next$ と更新する.
- (iii) 各節点 $i \in P$ に対して $Distance(i)$ が最小である節点 v を選択する. それを $next = v$ とする.
- (iv) $P = V$ となるまで, (ii), (iii) を繰り返す.

(2) 全 2 節点間の最短経路の計算

Floyd の方法を用いてグラフ $G = (V, E)$ におけるすべての 2 節点間の最短経路とその距離を求める.

まず, 無向グラフの場合について説明する. 負の重みの枝を含まないと仮定する.

- (i) 各節点 $i, j \in V$ に対して, $Distance(i, j) = \infty$ および $Path(i, j) = 0$ とする.
- (ii) 各節点 $i, j \in V$ に対して, もし $Distance(i, j) > Distance(i, k) + Distance(k, j)$ ならば, $Distance(i, j) = Distance(i, k) + Distance(k, j)$ および $Path(i, j) = k$ と更新する.
- (iii) $k = 1, \dots, n$ まで, (ii) を繰り返す.

次に, 有向グラフの場合について説明する. 負の重みの枝を含むが, 負の長さのサイクルを含まないと仮定する.

- (i) 各節点 $i, j \in V$ に対して $Distance(i, j) = \infty$ および $Path(i, j) = i$ とする.
- (ii) 各節点 $i, j \in V$ に対して $Distance(i, j) > Distance(i, k) + Distance(k, j)$ ならば, $Distance(i, j) = Distance(i, k) + Distance(k, j)$ および $Path(i, j) = k$ と更新する.
- (iii) もし $Distance(i, i)$ が負であるならば解が存在しないので, 処理を打ち切る. そうでなければ, $k = 1, \dots, n$ まで (ii) を繰り返す.

(3) 2 節点間の最短経路の計算

前述した Dijkstra の方法を応用してグラフ $G = (V, E)$ における 2 節点間の最短経路とその距離を求める. ただし, すべての枝の重みは非負であると仮定する.

- (i) 各節点 $i \in V$ に対して $Distance(i) = \infty$, $Path(i) = i$ および $P = \phi$ とする. 出発点 $init$ に対し, $Distance(init) = 0$ とする. また, $next = init$ とする.
- (ii) $P = P \cup \{next\}$ とする. 節点 $next$ に接続している各節点 j に対して, もし $Distance(j) > D(next) + Weight(next, j)$ ならば, $Distance(j) = D(next) + Weight(next, j)$ および $Path(j) = next$ と更新する.
- (iii) 各節点 $i \in P$ に対して $Distance(i)$ が最小である節点 v を選択する. それを $next = v$ とする.
- (iv) $next = iend$ となるまで, (ii),(iii) を繰り返す.

5.1.3 参考文献

- (1) G. E. Forsythe, M. A. Malcolm, C. B. Moler, 森正武訳, “計算機のための数値計算法”, 日本コンピュータ協会 (1978).
- (2) R. P. Brent, “Algorithms for minimization without derivatives”, Englewood Cliffs, N. J., Prent-Hall (1973).
- (3) 今野浩, 山下浩, “非線形計画法”, 日科技連 (1978).
- (4) 中川徹, 小柳義夫, “最小二乗法による実験データ解析”, 東京大学出版会 (1982).
- (5) M. J. D. Powell, “A Hybrid Method for Nonlinear Equations”, Numerical Methods for Nonlinear Algebraic Equations, P. Rabinowits, ed. , Gordon and Breach, pp. 87–161 (1970).
- (6) 茨木俊秀, 福島雅夫, “最適化プログラミング”, 岩波書店 (1991).
- (7) V. フバータル, 阪田省二郎, 藤野和建訳, “線形計画法 (上)”, 啓学出版 (1986).
- (8) 玄光男, 井田憲一, “BASIC による線形計画”, 共立出版 (1988).
- (9) ANSOP 研究会編, “パソコン FORTRAN 版非線形最適化プログラミング”, 日刊工業新聞社 (1991).
- (10) 伊理正夫, 白川功, 梶谷洋司, 篠田庄司, “演習グラフ理論”, コロナ社,(1983).
- (11) 茨木俊秀, “アルゴリズムとデータ構造”, 照光堂,(1989).
- (12) 茨木俊秀, “離散最適化とアルゴリズム”, 岩波書店,(1993).
- (13) 茨木俊秀, 福島雅夫, “最適化の手法”, 共立出版,(1993).
- (14) 棒沢芳雄, “オペレーションズ・リサーチ”, コロナ社,(1997).
- (15) 今野浩, “線形計画法”, 日科技連,(1997).

5.2 制約なし1変数関数の極小化

5.2.1 DMUUSN, RMUUSN

1変数関数の極小化

(1) 機能

1変数関数 $f(x)$ の極小値探索を行う。

(2) 使用法

倍精度サブルーチン:

CALL DMUUSN (F, X, ER, NEV, Y, IERR)

単精度サブルーチン:

CALL RMUUSN (F, X, ER, NEV, Y, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	F	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	—	入 力	関数 $f(x)$ を定義する関数副プログラム名 (注意事項 (a) 参照)
2	X	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	探索の出発 x_0
				出 力	最終到達点 x^*
3	ER	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	要求精度 (既定値: $2 \times \sqrt{\text{(誤差判定のための単位)}}$)
4	NEV	I	1	入 力	関数 $f(x)$ の最大評価回数 (既定値:100)
				出 力	実際の関数評価回数
5	Y	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	最終到達点 x^* での関数値 $y = f(x^*)$
6	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $ER \geq \text{誤差判定のための単位}$ (既定値にするため, 0.0 を入力する場合は除く)

(b) $NEV > 3$ (既定値にするため, 0 を入力する場合は除く)

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1500	制限条件 (a) または (b) を満足しなかった.	既定値にセットして処理する.
4000	困り込みに失敗した.	処理を打ち切る.
5000	与えられた最大評価回数に達しても収束しなかった.	その時点の X, Y の値を出力して, 処理を打ち切る.

(6) 注意事項

- (a) 引数第 1 項 F の実際の名前は、使用者側のプログラムで EXTERNAL 文を用いて宣言し、実際の名前の関数副プログラムを作っておかなければならない。

この関数副プログラムの作り方は、次に示すとおりである。

関数副プログラムの作り方

```
REAL(8) FUNCTION F(X)
REAL(8) X
      F=f(x)
RETURN
END
```

- (b) 区間縮小により探索区間が $[a, b]$ となったとき、次の条件により収束判定を行う。

$$\max(b - x, x - a) \leq 2 \times ER \times \max(1, |x|)$$

ER としては、既定値程度にとるのが望ましい。

- (c) 引数の内容の欄に既定値が記されている場合は、整数型のときは 0、実数型のときは 0.0 を入力すれば既定値がセットされる。
- (d) 極小値が複数あるときは、いずれの極小値に収束するか保証されない。
- (e) 関数は 1 階連続微分可能である必要がある。
- (f) 極大値探索を行いたい場合は、 $f(x)$ の関数値が正負逆になるように設定すればよい。このとき、引数第 5 項 Y の値は、正負逆に出力される。

(7) 使用例

- (a) 問題

$$f(x) = x(x^2 - 2) - 5$$

の極小値探索を行う。

- (b) 入力データ

関数 $f(x)$ に対応する関数副プログラム名: FMUUSN

X = 1.0, ER=0.0, NEV=0

- (c) 主プログラム

```
PROGRAM BMUUSN
! *** EXAMPLE OF DMUUSN ***
IMPLICIT REAL(8) (A-H,O-Z)
EXTERNAL FMUUSN
!
WRITE(6,1000)
READ(5,*) NEV
READ(5,*) ER
READ(5,*) X
WRITE(6,1100) NEV,ER,X
CALL DMUUSN(FMUUSN,X,ER,NEV,Y,IERR)
WRITE(6,1200) IERR,NEV,X,Y
STOP
!
1000 FORMAT(' ',/,', ' *** DMUUSN ***')
1100 FORMAT(' ** INPUT **',/,&
5X,'NEV =',I5,/,&
5X,'ER =',D18.10,/,&
5X,'(( INITIAL VALUE ))',/,&
5X,' X =',F5.1)
1200 FORMAT(' ** OUTPUT **',/,&
5X,'IERR =',I5,/,&
5X,'NEV =',I5,/,&
5X,'(( SOLUTION ))',/,&
5X,' X =',D18.10,/,&
5X,'(( FUNCTION VALUE ))',/,&
5X,' Y =',D18.10)
END

REAL(8) FUNCTION FMUUSN(X)
REAL(8) X
```

```
!
  FMUUSN = X*(X*X-2.0D0)-5.0D0
  RETURN
  END
```

(d) 出力結果

```
*** DMUUSN ***
** INPUT **
  NEV = 0
  ER = 0.000000000D+00
  (( INITIAL VALUE ))
  X = 1.0
** OUTPUT **
  IERR = 0
  NEV = 23
  (( SOLUTION ))
  X = 0.8090169944D+00
  (( FUNCTION VALUE ))
  Y = -0.6088525492D+01
```

5.3 制約なし多変数関数の極小化

5.3.1 DMUMQN, RMUMQN

多変数関数の極小化 (導関数定義不要)

(1) 機能

n 変数関数 $f(x)$ の極小値探索を行う。

(2) 使用法

倍精度サブルーチン:

CALL DMUMQN (F, X, N, ER, NEV, Y, WK, IERR)

単精度サブルーチン:

CALL RMUMQN (F, X, N, ER, NEV, Y, WK, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	F	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	—	入 力	関数 $f(x)$ を定義する関数副プログラム F(X) の関数副プログラム名 (注意事項 (a) 参照)
2	X	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	N	入 力	探索点の初期値 x_0
				出 力	探索点の最終到達点 x^*
3	N	I	1	入 力	独立変数 x の成分の数 n
4	ER	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	要求精度 (既定値: $2 \times \sqrt{\text{誤差判定のための単位}}$)
5	NEV	I	1	入 力	関数 $f(x)$ の最大評価回数 (既定値: $400 \times N$)
				出 力	実際の関数評価回数
6	Y	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	最終到達点 x^* での関数値 $y = f(x^*)$
7	WK	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	内容参照	ワーク	作業領域 大きさ: $(N \times (3 \times N + 7))$
8	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $N > 0$

(b) $ER \geq \text{誤差判定のための単位}$ (既定値にするため, 0.0 を入力する場合は除く)

(c) $NEV > 0$ (既定値にするため, 0 を入力する場合は除く)

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1500	制限条件 (b) または (c) を満足しなかった.	既定値にセットして処理する.
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
5000	与えられた最大評価回数に達しても収束しなかった.	その時点の X, Y の値を出力して, 処理を打ち切る.

(6) 注意事項

- (a) 引数第 1 項 F の実際の名前は, 使用者側のプログラムで EXTERNAL 文を用いて宣言し, 実際の名前の関数副プログラムを作っておかなければならない.

この関数副プログラムの作り方は, 次に示すとおりである.

関数副プログラムの作り方

```

REAL(8) FUNCTION F (X)
REAL(8) X
DIMENSION X(*)
      F=f(x)
RETURN
END

```

- (b) 収束判定は次式によって行い, $x+\Delta x$ を解とする.

$$\|\Delta x\| \leq ER \times \max(1, \|x + \Delta x\|) \text{ かつ } \|\nabla f(x)\| \leq 2 \times ER$$

または $\|\nabla f(x)\| \leq$ 誤差判定のための単位

ここで, Δx は x に対する修正ベクトルであり, $\|x\| = \max_i |x_i|$ である.

また $\nabla f(x)$ は $f(x)$ の勾配ベクトルで $\partial f(x)/\partial x_i$ を成分とする.

ER としては, 既定値程度にとるのが望ましい.

- (c) 引数の内容の欄に既定値が記されている場合は, 整数型のときは 0, 実数型のときは 0.0 を入力すれば既定値がセットされる.

- (d) 勾配ベクトルが解析的に計算できるなら, 5.3.2 $\left\{ \begin{array}{l} \text{DMUMQG} \\ \text{RMUMQG} \end{array} \right\}$ を利用したほうが効率がよい.

- (e) 各変数から関数値への寄与が同程度になるようにスケールするのが望ましい(詳細は 5.1.1 参照).

- (f) 初期点で勾配ベクトルが 0 のときは, その点が解として出力される.

- (g) 極小値がないときは, 最大関数評価回数まで計算し, IERR=5000 となる.

- (h) 極小値が複数あるときは, いずれの極小値に収束するか保証されない.

- (i) 関数は 2 階連続微分可能である必要がある.

- (j) 極大値探索を行いたい場合は, $f(x)$ の関数値が正負逆になるように設定すればよい. このとき, 引数第 6 項 Y の値は, 正負逆に出力される.

(7) 使用例

(a) 問題

$$f(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$$

を初期値 $x = [-1.2, 1.0]^T$ として、極小値探索する。

(b) 入力データ

関数 $f(x)$ に対応する関数副プログラム名: FMUMQN

$X(1) = -1.2, X(2) = 1.0, ER = 0.0, NEV = 0$

(c) 主プログラム

```

PROGRAM BMUMQN
! *** EXAMPLE OF DMUMQN ***
IMPLICIT REAL(8) (A-H,O-Z)
PARAMETER (N = 2)
DIMENSION X(N),WK(N*(3*N+7))
EXTERNAL FMUMQN
!
WRITE(6,1000)
READ(5,*) NEV
READ(5,*) ER
READ(5,*) X
WRITE(6,1100) N,NEV,ER,(I,X(I),I=1,N)
CALL DMUMQN(FMUMQN,X,N,ER,NEV,Y,WK,IERR)
WRITE(6,1200) IERR,NEV
WRITE(6,1300) (I,X(I),I=1,N)
WRITE(6,1400) Y
STOP
!
1000 FORMAT(' ',/, ' *** DMUMQN ***')
1100 FORMAT(' ** INPUT **',/, &
5X,'N =',I5,/, &
5X,'NEV =',I5,/, &
5X,'ER =',D18.10,/, &
5X,'(( INITIAL VALUE ))',/, &
(5X,' X(',I2,') =',F5.1))
1200 FORMAT(' ** OUTPUT **',/, &
5X,'IERR =',I5,/, &
5X,'NEV =',I5)
1300 FORMAT(5X,'(( SOLUTION ))',/, &
(5X,' X(',I2,') =',D18.10))
1400 FORMAT(5X,'(( FUNCTION VALUE ))',/, &
5X,' Y =',D18.10)
END

REAL(8) FUNCTION FMUMQN(X)
REAL(8) W,X,Y
DIMENSION X(*)
!
Y(1) = 10.000*(X(2)-X(1)*X(1))
Y(2) = 1.000-X(1)
W = 0.000
DO 100 I=1,2
W = W+Y(I)*Y(I)
100 CONTINUE
FMUMQN = W
RETURN
END

```

(d) 出力結果

```

*** DMUMQN ***
** INPUT **
N = 2
NEV = 0
ER = 0.0000000000D+00
(( INITIAL VALUE ))
X( 1) = -1.2
X( 2) = 1.0
** OUTPUT **
IERR = 0
NEV = 140
(( SOLUTION ))
X( 1) = 0.1000000000D+01
X( 2) = 0.1000000000D+01
(( FUNCTION VALUE ))
Y = 0.5453357277D-23

```

5.3.2 DMUMQG, RMUMQG

多変数関数の極小化 (導関数定義必要)

(1) 機能

n 変数関数 $f(x)$ の極小値探索を行う。

(2) 使用法

倍精度サブルーチン:

CALL DMUMQG (F, SUBG, X, N, ER, NEV, Y, WK, IERR)

単精度サブルーチン:

CALL RMUMQG (F, SUBG, X, N, ER, NEV, Y, WK, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	F	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	—	入 力	関数 $f(x)$ を定義する関数副プログラム F(X) の関数副プログラム名 (注意事項 (a) 参照)
2	SUBG	—	—	入 力	勾配ベクトル $\nabla f(x)$ を計算するサブルーチン SUBG(X, G) のサブルーチン名 (注意事項 (b) 参照)
3	X	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	N	入 力	探索点の初期値 x_0
				出 力	探索点の最終到達点 x^*
4	N	I	1	入 力	独立変数 x の成分の数 n
5	ER	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	要求精度 (既定値: $2 \times \sqrt{(\text{誤差判定のための単位})}$)
6	NEV	I	2	入 力	NEV (1) は, 関数 F の最大評価回数 (既定値: $100 \times N$) NEV (2) は, サブルーチン SUBG の最大評価回数 (既定値: $100 \times N$)
				出 力	実際の関数評価回数
7	Y	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	最終到達点 x^* での関数値 $y = f(x^*)$
8	WK	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	内容参照	ワーク	作業領域 大きさ: $(N \times (3 \times N + 7))$
9	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $N > 0$

(b) $ER \geq \text{誤差判定のための単位}$ (既定値にするため, 0.0 を入力する場合は除く)

(c) $NEV(i) > 0$ ($i = 1, 2$) (既定値にするため, 0 を入力する場合は除く)

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1500	制限条件 (b) または (c) を満足しなかった.	既定値にセットして処理する.
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
5000	与えられた最大評価回数に達しても収束しなかった.	その時点の X, Y の値を出力して, 処理を打ち切る.

(6) 注意事項

- (a) 引数第 1 項 F の実際の名前は, 使用者側のプログラムで EXTERNAL 文を用いて宣言し, 実際の名前の関数副プログラムを作っておかなければならない.

この関数副プログラムの作り方は次に示すとおりである.

関数副プログラムの作り方

```
REAL(8) FUNCTION F (X)
REAL(8) X
DIMENSION X(*)
    F = f(x)
RETURN
END
```

- (b) 引数第 2 項 SUBG の実際の名前は, 使用者側のプログラムで EXTERNAL 文を用いて宣言し, 実際の名前のサブルーチンを作っておかなければならない.

このサブルーチンの作り方は次に示すとおりである.

サブルーチンの作り方

```
SUBROUTINE SUBG(X, G)
REAL(8) X, G
DIMENSION X(*), G(*)
    G(1) = ∇ f(x) の第 1 成分 = ∂f(x)/∂x1
    ⋮
    G(N) = ∇ f(x) の第 n 成分 = ∂f(x)/∂xn
RETURN
END
```

- (c) 収束判定は次式によって行い, $x + \Delta x$ を解とする.

$$\|\Delta x\| \leq ER \times \max(1, \|x + \Delta x\|) \text{ かつ } \|\nabla f(x)\| \leq 2 \times ER$$

または $\|\nabla f(x)\| \leq$ 誤差判定のための単位

ここで, Δx は x に対する修正ベクトルであり, $\|x\| = \max_i |x_i|$ である.

ER としては, 既定値程度にとるのが望ましい.

- (d) 引数の内容の欄に既定値が記されている場合は, 整数型のときは 0, 実数型のときは 0.0 を入力すれば既定値がセットされる.

- (e) 各変数から関数値への寄与が同程度になるようにスケーリングするのが望ましい (詳細は 5.1.1 参照).

- (f) 初期点で勾配ベクトルが 0 のときは, その点が解として出力される.

- (g) 極小値がないときは, 最大関数評価回数まで計算し, IERR=5000 となる.

- (h) 極小値が複数あるときは, いずれの極小値に収束するか保証されない.

- (i) 関数は 2 階連続微分可能である必要がある。
- (j) 極大値探索を行いたい場合は、関数値 $f(x)$ と勾配ベクトル $\nabla f(x)$ が正負逆になるように設定すればよい。このとき、引数第 7 項 Y の値は、正負逆に出力される。

(7) 使用例

(a) 問題

$$f(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$$

を初期値 $x = [-1.2, 1.0]^T$ として、極小値探索する。

(b) 入力データ

関数 $f(x)$ に対応する関数副プログラム名: FMUMQG

勾配ベクトル $\nabla f(x)$ を計算するサブルーチン名: GMUMQG

X(1) = -1.2, X(2) = 1.0, N=2, ER=0.0, NEV(1) = 0, NEV(2) = 0

(c) 主プログラム

```

PROGRAM BMUMQG
! *** EXAMPLE OF DMUMQG ***
IMPLICIT REAL(8) (A-H,O-Z)
PARAMETER (N = 2)
DIMENSION NEV(2),X(N),WK(N*(3*N+7))
EXTERNAL FMUMQG,FMUMQ2
!
WRITE(6,1000)
READ(5,*) (NEV(I),I=1,2)
READ(5,*) ER
READ(5,*) X
WRITE(6,1100) N,NEV(1),NEV(2),ER,(I,X(I),I=1,N)
CALL DMUMQG(FMUMQG,FMUMQ2,X,N,ER,NEV,Y,WK,IERR)
WRITE(6,1200) IERR,NEV(1),NEV(2)
WRITE(6,1300) (I,X(I),I=1,N)
WRITE(6,1400) Y
STOP
!
1000 FORMAT(' ',/,', ' *** DMUMQG ***')
1100 FORMAT(' ** INPUT **',/,&
5X,' N =',I5,/,&
5X,' NEV(1)=',I5,/,&
5X,' NEV(2)=',I5,/,&
5X,' ER =',D18.10,/,&
5X,' (( INITIAL VALUE ))',/,&
(5X,' X(',I2,',) =',F5.1))
1200 FORMAT(' ** OUTPUT **',/,&
5X,' IERR =',I5,/,&
5X,' NEV(1)=',I5,/,&
5X,' NEV(2)=',I5)
1300 FORMAT(5X,' (( SOLUTION ))',/,&
(5X,' X(',I2,',) =',D18.10))
1400 FORMAT(5X,' (( FUNCTION VALUE ))',/,&
5X,' Y =',D18.10)
END

REAL(8) FUNCTION FMUMQG(X)
REAL(8) W,X,Y
DIMENSION X(*),Y(2)
!
Y(1) = 10.000*(X(2)-X(1)*X(1))
Y(2) = 1.000-X(1)
W = 0.000
DO 100 I=1,2
W = W+Y(I)*Y(I)
100 CONTINUE
FMUMQG = W
RETURN
END

SUBROUTINE FMUMQ2(X,G)
REAL(8) G,X,Y
DIMENSION X(*),G(*),Y(2)
!
Y(1) = 10.000*(X(2)-X(1)**2)
Y(2) = 1.000-X(1)
G(1) = -2.000*(20.000*X(1)*Y(1)+Y(2))
G(2) = 20.000*Y(1)
RETURN
END

```

(d) 出力結果

```

*** DMUMQG ***
** INPUT **
N = 2
NEV(1)= 0

```



```
NEV(2)= 0
ER = 0.0000000000D+00
(( INITIAL VALUE ))
X( 1) = -1.2
X( 2) = 1.0
** OUTPUT **
IERR = 0
NEV(1)= 55
NEV(2)= 36
(( SOLUTION ))
X( 1) = 0.1000000000D+01
X( 2) = 0.1000000000D+01
(( FUNCTION VALUE ))
Y = 0.7664560722D-24
```

5.4 制約なし関数二乗和の極小化

5.4.1 DMUSSN, RMUSSN

非線形最小二乗法 (導関数定義不要)

(1) 機能

n 変数関数の二乗和 $s = \sum_{i=1}^m f_i(x)^2$ の極小値探索を行う。

ここで、 $f_i(x)$ は、 n 変数ベクトル値関数 $f(x)$ の第 i 成分である ($i = 1, \dots, m$)。

(2) 使用法

倍精度サブルーチン:

CALL DMUSSN (SUB, X, N, ER, NEV, Y, M, S, IWK, WK, IERR)

単精度サブルーチン:

CALL RMUSSN (SUB, X, N, ER, NEV, Y, M, S, IWK, WK, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	SUB	—	—	入力	関数値 $y = f(x)$ を計算するサブルーチン SUB(X, Y) の手続き名 (注意事項 (a) 参照)
2	X	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	N	入力	探索点の初期値 x_0
				出力	探索点の最終到達点 x^*
3	N	I	1	入力	独立変数 x の成分の数 n
4	ER	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入力	要求精度 (既定値: $2 \times \sqrt{\text{(誤差判定のための単位)}}$)
5	NEV	I	1	入力	関数 $f(x)$ の最大評価回数 (既定値: $100 \times N$)
				出力	実際の関数評価回数
6	Y	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	M	出力	最終到達点 x^* での関数の値 $y = f(x^*)$
7	M	I	1	入力	従属変数 y の成分の数 m
8	S	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出力	最終到達点 x^* での関数二乗和の値 $s = \sum_{i=1}^m f_i(x^*)^2$
9	IWK	I	$4 \times N$	ワーク	作業領域
10	WK	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	内容参照	ワーク	作業領域 大きさ: $M \times (2 \times N + 1) + N \times (N + 4)$
11	IERR	I	1	出力	エラーインディケータ

(4) 制限条件

- (a) $0 < N \leq M$
- (b) $ER \geq$ 誤差判定のための単位 (既定値にするため, 0.0 を入力する場合は除く)
- (c) $NEV > 0$ (既定値にするため, 0 を入力する場合は除く)

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1500	制限条件 (b) または (c) を満足しなかった.	既定値にセットして処理する.
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
4000	線形最小二乗法が解けなかった.	その時点の X, Y, S の値を出力して処理を打ち切る.
4100	最急降下を計算できなかった.	
4200	2N 回連続して解の修正ができなかった.	
5000	与えられた最大評価回数に達しても収束しなかった.	

(6) 注意事項

- (a) 引数第 1 項 SUB の実際の名前は, 使用者側のプログラムで EXTERNAL 文を用いて宣言し, 実際の名前のサブルーチンを作っておかなければならない.

このサブルーチンの作り方は次に示すとおりである.

サブルーチンの作り方

```

SUBROUTINE SUB(X, Y)
  REAL(8) X, Y
  DIMENSION X(*), Y(*)
  Y(1) = f1(x)
  ⋮
  Y(M) = fm(x)
RETURN
END

```

- (b) 収束判定は次式によって行い, $x + \Delta x$ を解とする.

$$\|\Delta x\| \leq ER \times \max(1, \|x + \Delta x\|)$$
 ここで, Δx は x に対する修正ベクトルであり, $\|x\| = \max_i |x_i|$ である.
 ER としては, 既定値程度にとるのが望ましい.
- (c) 引数の内容の欄に既定値が記されている場合は, 整数型のときは 0, 実数型のときは 0.0 を入力すれば既定値がセットされる.
- (d) 各変数から関数値への寄与が同程度になるようにスケールするのが望ましい (詳細は 5.1.1 参照).
- (e) 極小値が複数あるときは, いずれの極小値に収束するか保証されない.
- (f) 関数は 1 階連続微分可能である必要がある.

(7) 使用例

(a) 問題

$$f(x) = \begin{bmatrix} f_1(x) \\ f_2(x) \end{bmatrix} = \begin{bmatrix} 10(x_2 - x_1^2) \\ 1 - x_2 \end{bmatrix}$$

のとき, 初期値 $x = [-1.2, 1.0]^T$ として,

$s = f_1(x)^2 + f_2(x)^2$ を極小にする.

(b) 入力データ

関数値 $f(x)$ を計算するサブルーチン名: FMUSSN

$X(1) = -1.2, X(2) = 1.0, N=2, ER=0.0, NEV=0, M=2$

(c) 主プログラム

```

PROGRAM BMUSSN
! *** EXAMPLE OF DMUSSN ***
IMPLICIT REAL(8) (A-H,O-Z)
PARAMETER (N = 2, M = 2)
DIMENSION IWK(3*N)
DIMENSION X(N), Y(M), WK(M*(2*N+1)+N*(N+4))
EXTERNAL FMUSSN
!
WRITE(6,1000)
READ(5,*) NEV
READ(5,*) ER
READ(5,*) X
WRITE(6,1100) N,M,NEV,ER
WRITE(6,1200) (I,X(I),I=1,N)
CALL DMUSSN(FMUSSN,X,N,ER,NEV,Y,M,SY,IWK,WK,IERR)
WRITE(6,1300) IERR,NEV
WRITE(6,1400) (I,X(I),I=1,N)
WRITE(6,1500) SY
WRITE(6,1600) (I,Y(I),I=1,M)
STOP
!
1000 FORMAT(' ',/, ' *** DMUSSN ***')
1100 FORMAT(' ** INPUT **',/, &
5X,'N =',I5,/, &
5X,'M =',I5,/, &
5X,'NEV =',I5,/, &
5X,'ER =',D18.10)
1200 FORMAT(5X,'(( INITIAL VALUE ))',/, &
(5X,' X(',I2,') =',F5.1))
1300 FORMAT(' ** OUTPUT **',/, &
5X,'IERR =',I5,/, &
5X,'NEV =',I5)
1400 FORMAT(5X,'(( SOLUTION ))',/, &
(5X,' X(',I2,') =',D18.10))
1500 FORMAT(5X,'(( LEAST SQUARES ))',/, &
5X,' S =',D18.10)
1600 FORMAT(5X,'(( FUNCTION VALUE ))',/, &
(5X,' Y(',I2,') =',D18.10))
END

SUBROUTINE FMUSSN(X,Y)
REAL(8) X,Y
DIMENSION X(*),Y(*)
!
Y(1) = 10.0D0*(X(2)-X(1)*X(1))
Y(2) = 1.0D0-X(1)
RETURN
END

```

(d) 出力結果

```

*** DMUSSN ***
** INPUT **
N = 2
M = 2
NEV = 0
ER = 0.0000000000D+00
(( INITIAL VALUE ))
X( 1) = -1.2
X( 2) = 1.0
** OUTPUT **
IERR = 0
NEV = 30
(( SOLUTION ))
X( 1) = 0.1000000000D+01
X( 2) = 0.1000000000D+01
(( LEAST SQUARES ))
S = 0.0000000000D+00
(( FUNCTION VALUE ))
Y( 1) = 0.0000000000D+00
Y( 2) = 0.0000000000D+00

```

5.5 制約付き 1 変数関数の極小化

5.5.1 DMCUSN, RMCUSN

1 変数関数の極小化 (区間指定)

(1) 機能

区間 $[a, b]$ の中で, 1 変数関数 $f(x)$ の極小値探索を行う.

(2) 使用法

倍精度サブルーチン:

CALL DMCUSN (F, AX, BX, ER, NEV, X, Y, IERR)

単精度サブルーチン:

CALL RMCUSN (F, AX, BX, ER, NEV, X, Y, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	F	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	—	入 力	関数 $f(x)$ を定義する関数副プログラム名 (注意事項 (a) 参照)
2	AX	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	探索区間の左端の初期値 a
3	BX	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	探索区間の右端の初期値 b
4	ER	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	要求精度 (既定値: $2 \times \sqrt{\text{(誤差判定のための単位)}}$)
5	NEV	I	1	入 力	関数 $f(x)$ の最大評価回数 (既定値:100)
				出 力	実際の関数評価回数
6	X	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	最終到達点 x^*
7	Y	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	最終到達点 x^* での関数値 $y = f(x^*)$
8	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $AX < BX$

(b) $AX \neq BX$

(c) $ER \geq \text{誤差判定のための単位}$ (既定値にするため, 0.0 を入力する場合は除く)

(d) $NEV > 0$ (既定値にするため, 0 を入力する場合は除く)

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1200	制限条件 (a) を満足しなかった.	AX と BX を入れ換えて処理する.
1500	制限条件 (c) または (d) を満足しなかった.	既定値にセットして処理する.
3000	制限条件 (b) を満足しなかった.	処理を打ち切る.
5000	与えられた最大評価回数に達しても収束しなかった.	その時点の X, Y の値を出力して, 処理を打ち切る.

(6) 注意事項

- (a) 引数第 1 項 F の実際の名前は, 使用者側のプログラムで EXTERNAL 文を用いて宣言し, 実際の名前の関数副プログラムを作っておかなければならない.

この関数副プログラムの作り方は次に示すとおりである.

関数副プログラムの作り方

```
REAL(8) FUNCTION F (X)
REAL(8) X
      F = f(x)
RETURN
END
```

- (b) 区間縮小により探索区間が (a, b) となったとき, 次の条件により収束判定を行う.

$$\max(b - x, x - a) \leq 2 \times ER \times \max(1, |x|)$$

ER としては, 既定値程度にとるのが望ましい.

- (c) 引数の内容の欄に既定値が記されている場合は, 整数型のときは 0, 実数型のときは 0.0 を入力すれば既定値がセットされる.
- (d) 極小値がない場合, 端点を解とし, IERR は 0 になる.
- (e) 区間内に極小値が複数あるときは, いずれの極小値に収束するか保証されない.
- (f) 関数は 1 階連続微分可能である必要がある.
- (g) 極大値探索を行いたい場合は, $f(x)$ の関数値が正負逆になるように設定すればよい. このとき, 引数第 7 項 Y の値は, 正負逆に出力される.

(7) 使用例

- (a) 問題

区間 $[0, 1]$ において,

$$f(x) = x(x^2 - 2) - 5$$

の極小値探索を行う.

- (b) 入力データ

関数 $f(x)$ に対応する関数副プログラム名: FMCUSN

AX=0.0, BX=1.0, ER=0.0, NEV=0

- (c) 主プログラム

```
PROGRAM BMCUSN
! *** EXAMPLE OF DMCUSN ***
IMPLICIT REAL(8) (A-H,O-Z)
EXTERNAL FMCUSN
!
```

```

WRITE(6,1000)
READ(5,*) NEV
READ(5,*) ER
READ(5,*) AX,BX
WRITE(6,1100) NEV,ER,AX,BX
CALL DMCUSN(FMCUSN,AX,BX,ER,NEV,X,Y,IERR)
WRITE(6,1200) IERR,NEV,X,Y
STOP
!
1000 FORMAT(' ',/, ' *** DMCUSN ***')
1100 FORMAT(' ** INPUT **',/, &
5X, 'NEV =', I5,/, &
5X, 'ER =', D18.10,/, &
5X, '(( SEARCH SECTION ))',/, &
5X, ' AX =', F5.1,/, &
5X, ' BX =', F5.1)
1200 FORMAT(' ** OUTPUT **',/, &
5X, 'IERR =', I5,/, &
5X, 'NEV =', I5,/, &
5X, '(( SOLUTION ))',/, &
5X, ' X =', D18.10,/, &
5X, '(( FUNCTION VALUE ))',/, &
5X, ' Y =', D18.10)
END

REAL(8) FUNCTION FMCUSN(X)
REAL(8) X
!
FMCUSN = X*(X*X-2.0D0)-5.0D0
RETURN
END

```

(d) 出力結果

```

*** DMCUSN ***
** INPUT **
NEV = 0
ER = 0.000000000D+00
(( SEARCH SECTION ))
AX = 0.0
BX = 1.0
** OUTPUT **
IERR = 0
NEV = 11
(( SOLUTION ))
X = 0.8164965811D+00
(( FUNCTION VALUE ))
Y = -0.6088662108D+01

```

5.6 制約付き多変数線形関数の最小化 (線形計画)

5.6.1 DMCLSN, RMCLSN

多変数線形関数の最小化 (線形制約)

(1) 機能

線形制約の中で, n 次のベクトル $x = [x_1, \dots, x_n]^T$ の多変数線形関数 $f(x)$ を最小にする x を求める.

$$f(x) = c^T x$$

制約条件は m 個で, $i = 1, 2, \dots, m$ について次のどれかである.

- $a_i^T x = b_i$
- $a_i^T x \leq b_i$
- $a_i^T x \geq b_i$

また, x の定義域は

$$d_j \leq x_j \leq u_j \quad (j = 1, 2, \dots, n)$$

ここで, $c^T = [c_1, c_2, \dots, c_n]$, $a_i^T = [a_{i,1}, a_{i,2}, \dots, a_{i,n}]$ は, n 次ベクトル, b_i, d_j, u_j は定数 ($i = 1, 2, \dots, m; j = 1, 2, \dots, n$) とする.

(2) 使用法

倍精度サブルーチン:

CALL DMCLSN (A, LMA, NM, B, M, XUP, XLOW, C, ITYPE, ER, NEV, X, Y, ISW, IWK, WK, IERR)

単精度サブルーチン:

CALL RMCLSN (A, LMA, NM, B, M, XUP, XLOW, C, ITYPE, ER, NEV, X, Y, ISW, IWK, WK, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER(4)} \\ 64 \text{ ビット整数版では INTEGER(8)} \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	A	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	LMA, NM	入 力	ISW=0 のとき制約条件の係数に対応する行列 $A = (a_{i,j})$ ($i = 1, 2, \dots, m; j = 1, 2, \dots, n$) (注意事項 (a) (f) 参照)
				出 力	スラック変数を用いて変更した制約条件の係数に対応する行列
2	LMA	I	1	入 力	配列 A の整合寸法
3	NM	I	1	入 力	ISW=0 のとき $n + 2 \times m$ の値. ここで n は変数の数, m は制約条件の数 (注意事項 (f) 参照)
				出 力	作業変数

項番	引数名	型	大きさ	入出力	内 容
4	B	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	M	入 力	ISW=0 のとき制約条件の右辺 $\mathbf{b} = (b_i)$ ($i = 1, 2, \dots, m$) (注意事項 (f) 参照)
				出 力	スラック変数を用いて変更した制約条件の右辺
5	M	I	1	入 力	制約条件の数 m
6	XUP	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NM	入 力	ISW=0 のとき変数の上限値 $\mathbf{u} = (u_j)$ ($j = 1, 2, \dots, n$) (注意事項 (c)(f) 参照)
				出 力	スラック変数を用いて変更した変数の上限値
7	XLOW	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NM	入 力	ISW=0 のとき変数の下限値 $\mathbf{d} = (d_j)$ ($j = 1, 2, \dots, n$) (注意事項 (c)(f) 参照)
				出 力	スラック変数を用いて変更した変数の下限値
8	C	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NM	入 力	ISW=0 のとき関数の係数 $\mathbf{c} = (c_j)$ ($j = 1, 2, \dots, n$) (注意事項 (f) 参照)
				出 力	スラック変数を用いて変更した関数の係数
9	ITYPE	I	M	入 力	制約条件の等号, 不等号の区別 0: $\mathbf{a}_i^T \mathbf{x} = b_i$ 1: $\mathbf{a}_i^T \mathbf{x} \leq b_i$ -1: $\mathbf{a}_i^T \mathbf{x} \geq b_i$
10	ER	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	要求精度 (既定値: $2 \times \sqrt{\text{(誤差判定のための単位)}}$)
11	NEV	I	1	入 力	関数 $f(\mathbf{x})$ の最大評価回数 (既定値: $10 \times M$)
				出 力	実際の関数評価回数
12	X	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NM	出 力	最終到達点 \mathbf{x} (注意事項 (b) 参照)
13	Y	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	最終到達点 \mathbf{x} での関数値 $f(\mathbf{x})$
14	ISW	I	1	入 力	処理スイッチ (注意事項 (i) 参照) 0: 最初の処理 1: 継続の処理
15	IWK	I	NM + M	ワーク	作業領域
16	WK	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	内容参照	ワーク	作業領域 大きさ: $2 \times M^2 + NM \times (7 + M)$
17	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

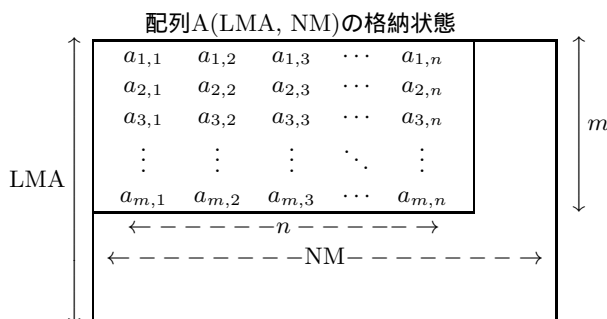
- (a) $0 < NM, 0 < M \leq LMA$
- (b) ISW = 0 または ISW = 1
- (c) $ER \geq \text{誤差判定のための単位}$ (既定値にするため, 0.0 を入力する場合は除く)
- (d) $NEV > 0$ (既定値にするため, 0 を入力する場合は除く)

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1500	制限条件 (c) または (d) を満足しなかった.	既定値にセットして処理する.
3000	制限条件 (a) または (b) を満足しなかった.	処理を打ち切る.
4000	実行可能基底解を求められなかった.	処理を打ち切る.
5000	与えられた最大評価回数に達しても収束しなかった.	その時点での X, Y の値を出力して, 処理を打ち切る. (注意事項 (h)(i) 参照)

(6) 注意事項

- (a) ISW=0 の時, 制約条件に対応する係数 $a_{i,j}$ ($i = 1, 2, \dots, m; j = 1, 2, \dots, n$) を以下の様に配列 A に格納する. ここで, m は制約条件の数, n は変数の数をそれぞれ表す.



備 考

- a. $NM = n + 2 \times m$ を満たさなければならない.

- (b) 最終到達点 x の値は $X(1) \sim X(n)$ に設定される. 配列 X の残りはスラック変数等の値である.
- (c) 変数の上限値または下限値が特に定められていないときには, 適当に絶対値の大きな正数または負数をそれぞれ上限値または下限値とする. なお, 最終到達点での変数値がここで定めた上限値または下限値と一致するときには, 最適解が求められていない可能性があるため, さらに絶対値の大きな数を上限値または下限値に設定して再計算を実行する必要がある.
- (d) 引数の内容の欄に既定値が記されている場合には, 整数型のときは 0, 実数型のときは 0.0 を入力すれば既定値がセットされる.
- (e) 各変数から関数値への寄与が同程度になるようにスケーリングするのが望ましい. 例えば, $f(x) = 100x_1 + x_2$ 制約条件: $200x_1 + 5x_2 = 3$ のような場合, $y_1 = 100x_1, y_2 = x_2$ と変数変換を行って, $h(y) = y_1 + y_2$ 制約条件: $2y_1 + 5y_2 = 3$ としたほうがよい結果が得られる.
- (f) A, B, XUP, XLOW, C, NM の出力には制約条件等を変換した後の値が入る.
- (g) 最大値探索を行いたい場合は, $-f(x)$ の最小値探索を行えば良い. このとき, Y の値を正負逆にしたものが最大値である.
- (h) IERR = 5000 のときの x の値は, 最適解ではないが, 制約条件を満たしている.
- (i) 指定した収束回数が少なく IERR=5000 が返されたとき, 途中まで計算した情報を用いて, 引き続き計算することができる. この処理を行うときには, ISW の値を 1, NEV の値を十分な値とし, それ以外の入力値には前回の出力値をそのまま用いる. また, 前回の作業領域の情報も用いる ((7) 使用例参照).

(7) 使用例

(a) 問題

$$\begin{aligned} x_1 + 2x_2 + 3x_3 - 3x_4 - 2x_5 - x_6 &\leq 20 \\ 2x_1 - 3x_2 - x_3 + 2x_4 + 4x_5 + x_6 &\geq 28 \\ -3x_1 + 2x_2 + 2x_3 + x_4 + 5x_5 + 2x_6 &= 40 \\ 5x_1 - x_2 + 3x_3 + 3x_4 - 2x_5 + 4x_6 &= 50 \\ 0.0 \leq x_i \leq 1000.0 \quad (i = 1, 2, 3, 4, 5, 6) \end{aligned}$$

のもとで

$$f(x) = -x_1 - 3x_2 + 2x_3 + 3x_4 - 4x_5 - 2x_6$$

を最小にする。

なお、この例では、注意事項 (i) で述べた継続処理を行うために、最大評価回数 NEV=6 と小さく設定し、IERR=5000 が出力されるようにしている。

(b) 入力データ

(1 回目): NM=14, M=4, ER=0.0, NEV=6, ISW=0, LMA=11, 配列 A, B, XUP, XLOW, C, ITYPE

(2 回目以降): NEV=6, ISW=1

(他は、前回の計算後の値を、そのまま入力値とする)

(c) 主プログラム

```

PROGRAM BMCLSN
! *** EXAMPLE OF DMCLSN ***
IMPLICIT REAL(8) (A-H,O-Z)
PARAMETER ( NMO = 14 )
PARAMETER ( MO = 4 )
PARAMETER ( LMA = 11 )
PARAMETER ( NWK = 2*MO*MO+NMO*(7+MO) )
PARAMETER ( NIWK = NMO+MO )
DIMENSION A(LMA,NMO),B(MO),XUP(NMO),XLOW(NMO),C(NMO),X(NMO)
DIMENSION WK(NWK),ITYPE(MO),IWK(NIWK)
!
WRITE(6,1000)
READ(5,*) NM,M,ER,NEV,ISW
WRITE(6,1100) NM , M
NN = NM-2*M
WRITE(6,1300)
DO 10 I = 1,M
READ(5,*) ( A(I,J),J=1,NN )
WRITE(6,1200) ( A(I,J),J=1,NN )
10 CONTINUE
WRITE(6,1400)
READ(5,*) ( B(I),I=1,M )
WRITE(6,1200) ( B(I),I=1,M )
WRITE(6,1500)
READ(5,*) ( XUP(I),I=1,NN )
WRITE(6,1200) ( XUP(I),I=1,NN )
WRITE(6,1600)
READ(5,*) ( XLOW(I),I=1,NN )
WRITE(6,1200) ( XLOW(I),I=1,NN )
WRITE(6,1700)
READ(5,*) ( C(I),I=1,NN )
WRITE(6,1200) ( C(I),I=1,NN )
WRITE(6,1800)
READ(5,*) ( ITYPE(I),I=1,M )
WRITE(6,1250) ( ITYPE(I),I=1,M )
CALL DMCLSN&
(A,LMA,NM,B,M,XUP,XLOW,C,ITYPE,ER,NEV,X,Y,ISW,IWK,WK,IERR)
WRITE(6,1900)
WRITE(6,2000) IERR
WRITE(6,2100)
WRITE(6,2400) ( I,X(I),I=1,NN )
WRITE(6,2200) Y
!
20 READ(5,*) NEV,ISW
CALL DMCLSN&
(A,LMA,NM,B,M,XUP,XLOW,C,ITYPE,ER,NEV,X,Y,ISW,IWK,WK,IERR)
WRITE(6,2300)
WRITE(6,2000) IERR
WRITE(6,2100)
WRITE(6,2400) ( I,X(I),I=1,NN )
WRITE(6,2200) Y
IF (IERR.EQ.5000) GOTO 20
STOP
!
1000 FORMAT(' ',/,6X,'*** DMCLSN ***',/,&

```

```

      7X, '** INPUT **')
1100 FORMAT(9X, 'NM =', I3, 8X, 'M =', I3)
1200 FORMAT(8X, 6(F7.1))
1250 FORMAT(8X, 4I4)
1300 FORMAT(8X, '** MATRIX A **')
1400 FORMAT(8X, '** VECTOR B **')
1500 FORMAT(8X, '** VECTOR XUP **')
1600 FORMAT(8X, '** VECTOR XLOW **')
1700 FORMAT(8X, '** VECTOR C **')
1800 FORMAT(8X, '** VECTOR ITYPE **')
1900 FORMAT(7X, '** OUTPUT **', /, &
      8X, '** FIRST RESULT (ISW.EQ.0) **')
2000 FORMAT(9X, 'IERR =', I5)
2100 FORMAT(8X, '** VECTOR X **')
2200 FORMAT(9X, 'Y =', D18.10)
2300 FORMAT(7X, '** OUTPUT **', /, &
      8X, '** IMPROVED RESULT (ISW.NE.0) **')
2400 FORMAT(8X, 'X(', I2, ') =', D18.10)
      END

```

(d) 出力結果

```

*** DMCLSN ***
** INPUT **
  NM = 14      M = 4
** MATRIX A **
  1.0  2.0  3.0 -3.0 -2.0 -1.0
  2.0 -3.0 -1.0  2.0  4.0  1.0
 -3.0  2.0  2.0  1.0  5.0  2.0
  5.0 -1.0  3.0  3.0 -2.0  4.0
** VECTOR B **
 20.0 28.0 40.0 50.0
** VECTOR XUP **
1000.0 1000.0 1000.0 1000.0 1000.0 1000.0
** VECTOR XLOW **
  0.0  0.0  0.0  0.0  0.0  0.0
** VECTOR C **
 -1.0 -3.0  2.0  3.0 -4.0 -2.0
** VECTOR ITYPE **
  1 -1  0  0
** OUTPUT **
** FIRST RESULT (ISW.EQ.0) **
  IERR = 5000
** VECTOR X **
X( 1) = 0.5162689805D+01
X( 2) = 0.000000000D+00
X( 3) = 0.9843817787D+01
X( 4) = 0.000000000D+00
X( 5) = 0.6412147505D+01
X( 6) = 0.1869848156D+01
  Y = -0.1486334056D+02
** OUTPUT **
** IMPROVED RESULT (ISW.NE.0) **
  IERR = 0
** VECTOR X **
X( 1) = 0.1811320755D+02
X( 2) = 0.1415094340D+02
X( 3) = 0.000000000D+00
X( 4) = 0.000000000D+00
X( 5) = 0.1320754717D+02
X( 6) = 0.000000000D+00
  Y = -0.1133962264D+03

```

5.6.2 DMCLAF, RMCLAF

多変数線形関数の最小化 (実不規則スパース行列で与えられる線形制約)

(1) 機能

線形制約の中で, n 次のベクトル $x = [x_1, \dots, x_n]^T$ の多変数線形関数 $f(x)$ を最小にする x を求める.

$$f(x) = c^T x$$

制約条件は m 個で, $i = 1, 2, \dots, m$ について次のどれかである.

- $a_i^T x = b_i$
- $a_i^T x \leq b_i$
- $a_i^T x \geq b_i$

また, x の定義域は

$$d_j \leq x_j \leq u_j \quad (j = 1, 2, \dots, n)$$

ここで, $c^T = [c_1, c_2, \dots, c_n]$, $a_i^T = [a_{i,1}, a_{i,2}, \dots, a_{i,n}]$ は, n 次ベクトル, b_i, d_j, u_j は定数 ($i = 1, 2, \dots, m; j = 1, 2, \dots, n$) とする.

(2) 使用法

倍精度サブルーチン:

CALL DMCLAF (AVAL, NA, JCN, IA, NM, B, M, XUP, XLOW, C, ITYPE, ETB, AP, BM,
NCK, NEV, X, Y, ISW1, ISW2, IW, W, NW, IERR)

単精度サブルーチン:

CALL RMCLAF (AVAL, NA, JCN, IA, NM, B, M, XUP, XLOW, C, ITYPE, ETB, AP, BM,
NCK, NEV, X, Y, ISW1, ISW2, IW, W, NW, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	AVAL	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NA+2×M	入 力	ISW1=0 のとき制約条件の係数に対応する行列 $A = (a_{i,j})$ ($i = 1, 2, \dots, m; j = 1, 2, \dots, n$) (注意事項 (e) 参照)
				出 力	スラック変数を用いて変更した制約条件の係数に対応する行列
2	NA	I	1	入 力	行列 A の非零要素の個数
3	JCN	I	NA+2×M	入 力	行列 A の非零要素の列番号 (注意事項 (e) (f) 参照)
4	IA	I	M+1	入 力	行列 A の非零要素の情報 (注意事項 (e) (f) 参照)
5	NM	I	1	入 力	ISW2=0 のとき $n + 2 \times m$ の値. ここで n は変数の数, m は制約条件の数 (注意事項 (f) 参照)
				出 力	作業変数
6	B	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	M	入 力	ISW2=0 のとき制約条件の右辺 $b = (b_i)$ ($i = 1, 2, \dots, m$) (注意事項 (f) 参照)
				出 力	スラック変数を用いて変更した制約条件の右辺
7	M	I	1	入 力	制約条件の数 m
8	XUP	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NM	入 力	ISW2=0 のとき変数の上限値 $u = (u_j)$ ($j = 1, 2, \dots, n$) (注意事項 (b)(f) 参照)
				出 力	スラック変数を用いて変更した変数の上限値
9	XLOW	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NM	入 力	ISW2=0 のとき変数の下限値 $d = (d_j)$ ($j = 1, 2, \dots, n$) (注意事項 (b)(f) 参照)
				出 力	スラック変数を用いて変更した変数の下限値
10	C	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NM	入 力	ISW2=0 のとき関数の係数 $c = (c_j)$ ($j = 1, 2, \dots, n$) (注意事項 (f) 参照)
				出 力	スラック変数を用いて変更した関数の係数
11	ITYPE	I	M	入 力	制約条件の等号, 不等号の区別 0: $a_i^T x = b_i$ 1: $a_i^T x \leq b_i$ -1: $a_i^T x \geq b_i$
12	ETB	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	3	入 力	ETB(1): 収束判定の単位 ϵ_C (既定値: $10 \times \sqrt{\text{(誤差判定のための単位)}}$) ETB(2): 残差の検定の単位 ϵ_r (既定値: $\sqrt{\text{(誤差判定のための単位)}}$) ETB(3): 実行可能性の検定の単位 ϵ_A (既定値: $\sqrt{\text{(誤差判定のための単位)}}$) (注意事項 (i) 参照)

項番	引数名	型	大きさ	入出力	内 容
13	AP	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	ステップサイズのパラメータ α (既定値: 0.99)(注意事項 (h) 参照)
14	BM	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	ビッグ M 法のパラメータ M (既定値: $10 \times \max_i c_i $)
15	NCK	I	1	入 力	制約条件に対する残差のチェックを行う間隔 (既定値: 10) (注意事項 (i) 参照)
16	NEV	I	1	入 力	関数 $f(x)$ の最大評価回数 (既定値: $10 \times M$)
				出 力	実際の関数評価回数
17	X	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NM	出 力	最終到達点 x (注意事項 (a) 参照)
18	Y	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	最終到達点 x での関数値 $f(x)$
19	ISW1	I	1	入 力	処理スイッチ (既定値:0) (注意事項 (j) 参照) 0: 自動的に選択 1: 連立 1 次方程式の係数行列を密行列として扱う. 2: 連立 1 次方程式の係数行列を疎行列として扱う.
				出 力	ISW1=0 のとき選択された処理スイッチ
20	ISW2	I	1	入 力	処理スイッチ (既定値:0) (注意事項 (k) 参照) 0: 最初の処理 1: 継続の処理
21	IW	I	内容参照	ワーク	作業領域 大きさ: $NW + NA + 14 \times M + 4 \times NM + 30$
22	W	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NW	ワーク	作業領域
23	NW	I	1	入 力	配列 W の大きさ (注意事項 (l) 参照)
24	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $XLOW(J) \leq XUP(J) \quad J = 1, 2, \dots, NM - 2 \times M$

(b) $ISW1 = 0$ または $ISW1 = 1$

(c) $ISW2 = 0$ または $ISW2 = 1$

(d) $ETB(I) \geq$ 誤差判定のための単位 ($I = 1, 2, 3$) (既定値にするため, 0.0 を入力する場合は除く)(e) $NEV > 0$ (既定値にするため, 0 を入力する場合は除く)(f) $NCK > 0$ (既定値にするため, 0 を入力する場合は除く)(g) $AP > 0$ (既定値にするため, 0.0 を入力する場合は除く)(h) $BM > 0$ (既定値にするため, 0.0 を入力する場合は除く)

(i) $0 < NM, 0 < M$

(j) $IA(1) = 1$

$0 < IA(i+1) - IA(i) \leq N \quad (i = 1, 2, \dots, N-1)$

$0 < NA - IA(N) + 1 \leq N$

(k) $M \leq NA \leq M \times N$

(l) $0 <$ 行列 A の非零要素の列番号 $\leq N$ 配列 JCN に格納されている行列 A の非零要素の列番号は, 各行ごとに昇順でなければならない。また, 各列に非零要素が少なくとも 1 個存在しなければならない。(m) $NW \geq NA + 17 \times M + 13 + (A(D^{(k)})^2 A^T$ を LU 分解するのに必要な領域の大きさ)。なお, ここで $N = NM - 2 \times M$ である。また, $D^{(k)}$ は, k 回目の反復の時点での解 $x^{(k)} = (x_1^{(k)}, x_2^{(k)}, \dots, x_n^{(k)})^T$ に対して $x_i^{(k)}$ を (i, i) 要素に持つ対角行列である。

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1000	ある j ($j = 1, 2, \dots, NM - 2 \times M$) について, $XLOW(j)$ と $XUP(j)$ が, 制限条件 (a) を満足しなかった.	$XLOW(j)$ と $XUP(j)$ の値を入れ替えて処理する.
1500	制限条件 (b)-(h) のいずれかを満足しなかった.	既定値にセットして処理する.
3000	制限条件 (i) を満足しなかった.	処理を打ち切る.
3010	制限条件 (j)-(l) のいずれかを満足しなかった.	
3100	制限条件 (m) を満足しなかった.	
4000	実行可能基底解を求められなかった.	
4100	制約条件を与える行列 A が $rank(A) < m$ であった. (m は制約条件の個数)	
4200	制約条件に対する残差が要求精度を満たさなくなった. (注意事項 (i) 参照)	
5000	与えられた最大評価回数に達しても収束しなかった.	その時点での X, Y の値を出力して, 処理を打ち切る. (注意事項 (h)(i) 参照)

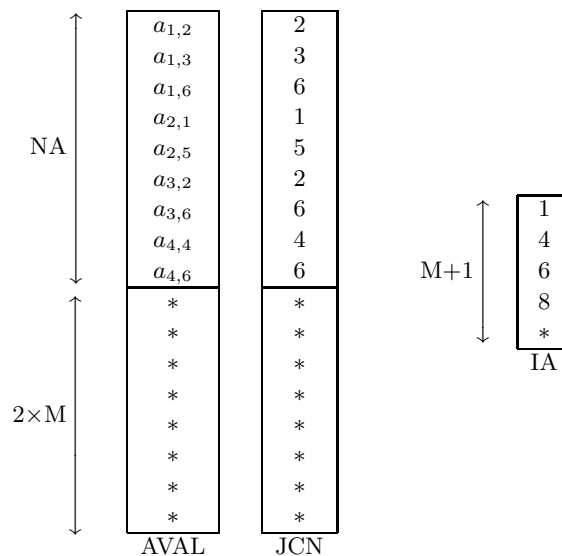
(6) 注意事項

- (a) 最終到達点 x の値は $X(1) \sim X(n)$ に設定される。配列 X の残りはスラック変数等の値である。
- (b) 変数の上限値または下限値が特に定められていないときには、適当に絶対値の大きな正数または負数をそれぞれ上限値または下限値とする。なお、最終到達点での変数値がここで定めた上限値または下限値と一致するときには、最適解が求められていない可能性があるため、さらに絶対値の大きな数を上限値または下限値に設定して再計算を実行する必要がある。
- (c) 引数の内容の欄に既定値が記されている場合には、整数型のときは 0、実数型のときは 0.0 を入力すれば既定値がセットされる。
- (d) 各変数から関数値への寄与が同程度になるようにスケーリングするのが望ましい。例えば、 $f(x) = 100x_1 + x_2$ 制約条件: $200x_1 + 5x_2 = 3$ のような場合、 $y_1 = 100x_1, y_2 = x_2$ と変数変換を行って、 $h(y) = y_1 + y_2$ 制約条件: $2y_1 + 5y_2 = 3$ としたほうがよい結果が得られる。
- (e) ISW=0 の時、制約条件に対応する係数 $a_{i,j}$ ($i = 1, 2, \dots, m; j = 1, 2, \dots, n$) のうち 0 でないものだけを配列 AVAL に格納する。ここで、 m は制約条件の数、 n は変数の数をそれぞれ表す。例えば、制約条件を表す行列 $A = (a_{ij})$ が

$$A = \begin{bmatrix} 0.0 & a_{12} & a_{13} & 0.0 & 0.0 & a_{16} \\ a_{21} & 0.0 & 0.0 & 0.0 & a_{25} & 0.0 \\ 0.0 & a_{32} & 0.0 & 0.0 & 0.0 & a_{36} \\ 0.0 & 0.0 & 0.0 & a_{44} & 0.0 & a_{46} \end{bmatrix}$$

で与えられるときに配列 AVAL, JCN および IA の格納状態は次のようになる。

配列 AVAL, JA, IA の格納状態



備考 * の部分は入力値として設定されていなくてもよい。

- (f) A, B, XUP, XLOW, C, NM の出力には制約条件等を変換した後の値が入る。
- (g) 最大値探索を行いたい場合は、 $-f(x)$ の最小値探索を行えば良い。このとき、 Y の値を正負逆にしたものが最大値である。
- (h) ステップサイズのパラメータ α が 1 に近いほど、理論上は最適解への収束に要する反復回数は少なくなるが、 α が 1 に近すぎると、数値計算上の誤差が大きくなる場合がある。
- (i) このサブルーチンでは、計算誤差により制約条件が十分な精度で満たされなくなることを防止するため、反復回数 k が

- i. k が 1 のとき
- ii. k が引数 NCK の値で割り切れるとき
- iii. k が引数 NEV の値に等しい時

のいずれかの条件を満たす時には、その時点での解 $x^{(k)}$ の制約条件に対する残差が条件

$$\|Ax^{(k)} - b\| \leq \epsilon_r$$

を満たしているかどうかをチェックする。ここで ϵ_r は引数 ETB(2) に設定されている値である。もしこの条件が満たされないときには、それ以前にこの条件を満たした解をもとに初期解を再計算して反復を繰り返す。初期解の再計算から NCK 回後の反復で再び上記の条件を満たさなかったときは、 $NCK \leftarrow \max(NCK/2, 1)$ としてさらに反復を繰り返す。NCK=1 になっても上記の条件を満たさなかった場合は IERR=4200 を出力して処理を打ち切る。

- (j) 本サブルーチンでは、最適解の探索方向を定めるために、連立 1 次方程式を解くが、その方法として連立 1 次方程式の係数行列を密行列として扱うか、疎行列として扱うかを引数 ISW1 の値により、選択することができる。係数行列を密行列として扱う場合には、< 基本機能第 2 分冊 > 2.2.2 $\left\{ \begin{array}{l} \text{DBGMSL} \\ \text{RBGMSL} \end{array} \right\}$ が使用され、疎行列として扱う場合には < 基本機能第 2 分冊 > 2.21.1 $\left\{ \begin{array}{l} \text{DBMFSL} \\ \text{RBMFSL} \end{array} \right\}$ が使用される。制約条件の係数行列 A が疎行列で、なおかつ AA^T もまた疎行列である場合には ISW1=2 に設定して、連立 1 次方程式の係数行列を疎行列として扱った方が計算時間は少なく済む。それ以外の場合は ISW1=0 に設定した方がよい。
- (k) 指定した収束回数が少なく IERR=5000 が返されたとき、途中まで計算した情報を用いて、引き続き計算することができる。この処理を行うときには、ISW2 の値を 1、NEV の値を十分な値とし、それ以外の入力値には前回の出力値をそのまま用いる。また、前回の作業領域の情報も用いる (例題参照)。
- (l) 反復計算が終了した時点で、人為変数の絶対値の最大値が ETB(3) に設定された値より大きい時には、与えられた問題が実行不可能であると判定し、IERR=4000 が出力される。
- (m) 制約条件を与える行列 A について $\text{rank}(A) < m$ の場合、本サブルーチンでは最適解を計算することができず、IERR=4100 が出力される。このような場合には 5.6.1 $\left\{ \begin{array}{l} \text{DMCLSN} \\ \text{RMCLSN} \end{array} \right\}$ を利用されたい。
- (n) 配列 W の大きさ NW は前もって見積もる必要がある。NW が十分大きくないために IERR=3100 が出力されることを防ぐには、 $NW = NA + 11 \times NM - 4 \times M + 2 \times M \times M + 10$ に設定すれば良い。

(7) 使用例

(a) 問題

$$x_1 + 2x_4 + 3x_5 \geq -7$$

$$2x_2 - x_3 = 0$$

$$x_1 + 2x_5 \leq 8$$

$$0 \leq x_1 \leq 1, 0 \leq x_2 \leq 1, 1 \leq x_3 \leq 2, -3 \leq x_4 \leq 0, 2 \leq x_5 \leq 5$$

のもとで

$$f(x) = 2x_1 + x_2 + x_3 - 3x_4 + x_5$$

を最小にする。

なお、この例では、注意事項 (k) で述べた継続処理を行うために、最大評価回数 NEV=5 と小さく設定し、IERR=5000 が出力されるようにしている。

(b) 入力データ

(1 回目): NM=14, M=4, AP=0.0, BM=0.0, NEV=5, ISW1=0, ISW2=0, LMA=11, 配列 A, JCN, IA, B, XUP, XLOW, C, ETB, ITYPE

(2 回目以降): NEV=20, ISW2=1

(他は, 前回の計算後の値を, そのまま入力値とする)

(c) 主プログラム

```

PROGRAM BMCLAF
! *** EXAMPLE OF DMCLAF ***
IMPLICIT NONE
INTEGER N,NA,M,NM,NCK,NW,LNW,LNA,LM,LNM,LA
INTEGER NEV,ISW1,ISW2
PARAMETER (LNA=50,LM=5,LNM=20,LA=5,LNW=5000)
INTEGER I
INTEGER IA(LA),ITYPE(LM),IERR
INTEGER JCN(LNA),IW(LNW+LNA+14*LM+4*LNM+30)
REAL(8) B(LA),AVAL(LNA),ER(3),W(LNW)
REAL(8) X(LNM),XUP(LNM),XLOW(LNM),C(LNM)
REAL(8) AP,BM,Y
!
READ (5,*) NA
READ (5,*) NM
READ (5,*) M
READ (5,*) ER(1),ER(2),ER(3)
READ (5,*) AP
READ (5,*) BM
READ (5,*) NCK
READ (5,*) NEV
READ (5,*) ISW1
READ (5,*) ISW2
READ (5,*) NW
N = NM - 2 * M
WRITE (6,1000) NA,NM,M,ER(1),ER(2),ER(3),AP,BM,&
NCK,NEV,ISW1,ISW2,NW
!
READ (5,*) (AVAL(I),I=1,NA)
WRITE (6,1100) (AVAL(I),I=1,NA)
!
READ (5,*) (JCN(I),I=1,NA)
WRITE (6,1130) (JCN(I),I=1,NA)
!
READ (5,*) (IA(I),I=1,M)
WRITE (6,1150) (IA(I),I=1,M)
!
READ (5,*) (B(I),I=1,M)
WRITE (6,1200) (B(I),I=1,M)
!
READ (5,*) (XUP(I),I=1,N)
WRITE (6,1210) (XUP(I),I=1,N)
!
READ (5,*) (XLOW(I),I=1,N)
WRITE (6,1220) (XLOW(I),I=1,N)
!
READ (5,*) (C(I),I=1,N)
WRITE (6,1230) (C(I),I=1,N)
!
READ (5,*) (ITYPE(I),I=1,M)
WRITE (6,1160) (ITYPE(I),I=1,M)
!
WRITE (6,1300)
!
CALL DMCLAF(AVAL,NA,JCN,IA,NM,B,M,XUP,XLOW,C,ITYPE,&
ER,AP,BM,NCK,NEV,X,Y,ISW1,ISW2,IW,W,NW,IERR)
!
WRITE (6,1900)
WRITE (6,1400) 'DMCLAF',IERR
WRITE (6,1450) NEV
WRITE (6,1470) ISW1
IF ((IERR .GE. 3000) .AND. (IERR .NE. 5000)) STOP
WRITE (6,1500) (I,X(I),I=1,N)
WRITE (6,1600) Y
!
IF (IERR .EQ. 5000) THEN
ISW2 = 1
NEV = 100
CALL DMCLAF(AVAL,NA,JCN,IA,NM,B,M,XUP,XLOW,C,ITYPE,&
ER,AP,BM,NCK,NEV,X,Y,ISW1,ISW2,IW,W,NW,IERR)
!
WRITE (6,2300)
WRITE (6,1400) 'DMCLAF',IERR
WRITE (6,1450) NEV
WRITE (6,1470) ISW1
IF (IERR .GE. 3000) STOP
WRITE (6,1500) (I,X(I),I=1,N)
WRITE (6,1600) Y
ENDIF
STOP
!
1000 FORMAT(' ',/,/,&
' *** DMCLAF ***',/,&
2X,' *** INPUT ***',/,&
6X,'NA =',I3,/,&
6X,'NM =',I3,/,&
6X,'M =',I3,/,&

```

```

        6X,'ER(1)=' ,F9.3,/,&
        6X,'ER(2)=' ,F9.3,/,&
        6X,'ER(3)=' ,F9.3,/,&
        6X,'AP  =' ,F9.3,/,&
        6X,'BM  =' ,F9.3,/,&
        6X,'NCK =' ,I3,/,&
        6X,'NEV =' ,I3,/,&
        6X,'ISW1 =' ,I3,/,&
        6X,'ISW2 =' ,I3,/,&
        6X,'NW  =' ,I3,/,&
        6X,'COEFFICIENT MATRIX AVAL')
1100 FORMAT(' ',7(F9.3))
1130 FORMAT(' ',/,/,&
        6X,'JCN (COLUMN NUMBER CORRESPONDING TO AVAL(K))',/,&
        7X,11(I4))
1150 FORMAT(' ',/,/,&
        6X,'IA (STARTING POSITION OF THE I-TH ROW IN ',/,&
        6X,' ARRAYS AVAL,JCN)',/,&
        7X,10(I4))
1160 FORMAT(' ',/,/,&
        6X,'ITYPE (TYPE OF EACH CONSTRAINT)',/,&
        7X,10(I4))
1200 FORMAT(' ',/,/,&
        6X,'CONSTANT VECTOR',/, (7X,F10.4))
1210 FORMAT(' ',/,/,&
        6X,'UPPER BOUND OF EACH VARIABLE',/, (7X,F10.4))
1220 FORMAT(' ',/,/,&
        6X,'LOWER BOUND OF EACH VARIABLE',/, (7X,F10.4))
1230 FORMAT(' ',/,/,&
        6X,'COEFFICIENTS OF OBJECTIVE FUNCTION',/, (7X,F10.4))
1300 FORMAT(' ',/,/,&
        2X,'** OUTPUT **',/,/)
1400 FORMAT(6X,'IERR (' ,A6,') =' ,I5)
1450 FORMAT(6X,'ITERATION NUMBER =' ,I5)
1470 FORMAT(6X,'SELECTED ISW1 =' ,I5)
1500 FORMAT(' ',/,&
        6X,'SOLUTION',/, (8X,'X(' ,I2,') =' ,D18.10))
1600 FORMAT(' ',/,&
        6X,'FUNCTION VALUE ',/, (8X,D18.10))
1900 FORMAT(' ',/,/,&
        7X,'** OUTPUT **',/,&
        8X,'** FIRST RESULT (ISW2.EQ.0) **',/)
2300 FORMAT(' ',/,/,&
        7X,'** OUTPUT **',/,&
        8X,'** IMPROVED RESULT (ISW2.NE.0) **',/)
END

```

(d) 出力結果

```

*** DMCLAF ***
** INPUT **
NA = 7
NM = 11
M = 3
ER(1)= 0.000
ER(2)= 0.000
ER(3)= 0.000
AP = 0.000
BM = 0.000
NCK = 0
NEV = 5
ISW1 = 0
ISW2 = 0
NW =200
COEFFICIENT MATRIX AVAL
1.000 2.000 3.000 2.000 -1.000 1.000 2.000

JCN (COLUMN NUMBER CORRESPONDING TO AVAL(K))
1 4 5 2 3 1 5

IA (STARTING POSITION OF THE I-TH ROW IN
ARRAYS AVAL,JCN)
1 4 6

CONSTANT VECTOR
-7.0000
0.0000
8.0000

UPPER BOUND OF EACH VARIABLE
1.0000
1.0000
2.0000
0.0000
5.0000

LOWER BOUND OF EACH VARIABLE
0.0000
0.0000
1.0000
-3.0000
2.0000

COEFFICIENTS OF OBJECTIVE FUNCTION

```

```

2.0000
1.0000
1.0000
-3.0000
1.0000

```

```

ITYPE (TYPE OF EACH CONSTRAINT)
-1  0  1

```

```

** OUTPUT **

```

```

** OUTPUT **
** FIRST RESULT (ISW2.EQ.0) **

```

```

IERR (DMCLAF) = 5000
ITERATION NUMBER = 5
SELECTED ISW1 = 1

```

```

SOLUTION
X( 1) = 0.3804293265D+00
X( 2) = 0.7015472780D+00
X( 3) = 0.1403143425D+01
X( 4) = -0.2995978790D+01
X( 5) = 0.2002974615D+01

```

```

FUNCTION VALUE
0.1385646034D+02

```

```

** OUTPUT **
** IMPROVED RESULT (ISW2.NE.0) **

```

```

IERR (DMCLAF) = 1500
ITERATION NUMBER = 20
SELECTED ISW1 = 1

```

```

SOLUTION
X( 1) = 0.2194790042D-08
X( 2) = 0.5000000006D+00
X( 3) = 0.1000000001D+01
X( 4) = -0.1322835175D-09
X( 5) = 0.2000000000D+01

```

```

FUNCTION VALUE
0.3500000007D+01

```

5.6.3 DMCLMZ, RMCLMZ

0-1 変数を含む線形制約付き多変数線形関数の最小化 (混合 0-1 計画)

(1) 機能

制約条件

$$\begin{aligned} \sum_{j=1}^n a_{i,j}x_j &= b_i && (i = 1, \dots, m_e; j = 1, \dots, n) \\ \sum_{j=1}^n a_{i,j}x_j &\leq b_i && (i = m_e + 1, \dots, m; j = 1, \dots, n) \\ d_j &\leq x_j \leq u_j && (j = 1, \dots, n) \\ x_j &= 0, 1 && (j \in N_{01}) \end{aligned}$$

のもとで目的関数

$$f(x) = \sum_{j=1}^n c_j x_j$$

を最小にする $x = (x_1, \dots, x_n)$ と、それに対する目的関数 $f(x)$ の値を求める。 N_{01} は 0-1 変数の添え字の集合である。

(2) 使用法

倍精度サブルーチン:

CALL DMCLMZ (A, MA, N, B, M, ME, XUP, XLOW, LZ, LN, C, MP, NP, ER, NEV, X, Y,
ISW, IWK, WK, IERR)

単精度サブルーチン:

CALL RMCLMZ (A, MA, N, B, M, ME, XUP, XLOW, LZ, LN, C, MP, NP, ER, NEV, X, Y,
ISW, IWK, WK, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER(4)} \\ 64 \text{ ビット整数版では INTEGER(8)} \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	A	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	MA,N	入 力	制約条件の左辺の係数 $a_{i,j}$ (注意事項 (a) 参照)
2	MA	I	1	入 力	配列 A の整合寸数
3	N	I	1	入 力	変数の数 n
4	B	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	M	入 力	制約条件の右辺の定数 b_i を成分に持つ m 次元ベクトル
5	M	I	1	入 力	制約条件の数 m
6	ME	I	1	入 力	等式制約条件の数 m_e
7	XUP	I	N	入 力	変数 x_j の上限値 u_j (注意事項 (b) 参照)
8	XLOW	I	N	入 力	変数 x_j の下限値 d_j (注意事項 (b) 参照)
9	LZ	I	LN	入 力	0-1 変数の添え字の集合 N_{01} (注意事項 (i) 参照)
10	LN	I	1	入 力	0-1 変数の個数 (注意事項 (h) 参照)
11	C	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	N	入 力	目的関数の係数ベクトル c_j

項番	引数名	型	大きさ	入出力	内 容
12	MP	I	1	入 力	分枝限定法における探索の深さ p $p = 1$ のとき深さ優先探索 p が大きくなるにしたがって発見的探索に近づく (注意事項 (f) 参照)
13	NP	I	1	入 力	分枝限定法における部分問題のリストの長さの最大値
14	ER	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	要求精度 (注意事項 (c) 参照)
15	NEV	I	1	入 力	緩和問題を解くときの最大反復回数 (注意事項 (d) 参照)
16	X	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	N	出 力	最適解 x
17	Y	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	最適解 x に対する目的関数の値 $f(x)$
18	ISW	I	1	入 力	処理スイッチ (注意事項 (g) および (h) 参照) ISW=0:最初の処理 ISW=1:継続処理
19	IWK	I	内容参照	ワーク	作業領域 大きさ: $MP + (MP + 3) \times (N + M - ME) + M + (LN + 2) \times NP + 7$
20	WK	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	内容参照	ワーク	作業領域 大きさ: $M \times 4 + (M + 4) \times (N + 2 \times M - ME) + (MP + 1) \times (M + 1) \times (N - ME + 1) + (M + 1) \times (N + M - ME + 1) + 2 \times LN + (N + M - ME + 2) \times (NP + 1) + 2$
21	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

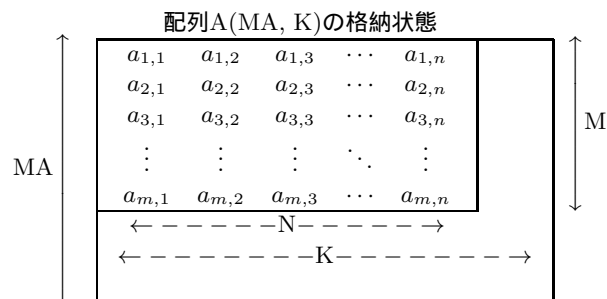
- (a) $M, LN, N, MP > 0$
- (b) $MA \geq M$
- (c) $0 \leq ME \leq N, M$
- (d) $N \geq LN$
- (e) $NP \geq 2$
- (f) $1 \leq LZ(1) < LZ(2) < \dots < LZ(LN) \leq N$
- (g) $ER > 0.0$ (既定値を使用するため 0.0 または負値を入力する場合を除く)
- (h) $NEV > 0$ (既定値を使用するため 0 または負値を入力する場合を除く)
- (i) ISW=0 または ISW=1
- (j) $XUP(i) \geq XLOW(i)$ ($i = 1, 2, \dots, N$)
- (k) IERR=5600 が出力された後に ISW=1 を設定して継続処理を行うときには, NP の値は前回の処理よりも大きく設定されていなければならない。

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1000	制限条件 (g) または (h) を満たさなかった.	既定値を使用して処理を続ける.
1100	制限条件 (i) を満たさなかった.	ISW=0 と見なして処理を続ける.
1200	制限条件 (j) を満たさなかった.	上限値と下限値を入れ換えて処理を続ける.
3000	制限条件 (a) を満たさなかった.	処理を打ち切る.
3010	制限条件 (b) を満たさなかった.	
3020	制限条件 (c) を満たさなかった.	
3030	制限条件 (d) を満たさなかった.	
3040	制限条件 (e) を満たさなかった.	
3100	制限条件 (f) を満たさなかった.	
3500	制限条件 (k) を満たさなかった.	
4000	実行可能解が存在しない.	
5000	与えられた反復回数内で緩和問題の解が求められなかった. 暫定解は求められている.	その時点での暫定解を出力して処理を打ち切る.
5100	与えられた反復回数内で緩和問題の解が求められなかった. 暫定解は求められていない.	処理を打ち切る.
5500	分枝限定法の計算において, 終端されずにメモリ上に保持されている部分問題の個数が, NP に達した. 暫定解は求められている.	その時点での暫定解を出力して処理を打ち切る.
5600	分枝限定法の計算において, 終端されずにメモリ上に保持されている部分問題の個数が, NP に達した. 暫定解は求められていない.	処理を打ち切る.

(6) 注意事項

(a) 制約条件の左辺の係数 $a_{i,j}$ は配列 A に以下のように格納されている.



備 考

a. $MA \geq M, K \geq N$ を満たさなければならない.

(b) x_j が 0-1 変数である場合については, XUP (j) と XLOW (j) の値は設定しておかなくてもよい.

(c) 引数 ER に正でない値が入力された場合, 既定値の $2 \times \sqrt{\text{誤差判定のための単位}}$ が要求精度として用いられる.

- (d) 混合 0-1 問題の 0-1 変数の一部を 0 または 1 に固定した問題を部分問題と呼ぶ。部分問題の固定されていない 0-1 変数についての条件を弱めてそれらが 0 以上 1 以下の実数値を取り得るようにしてできる線形計画問題をその部分問題の緩和問題と呼ぶ。混合 0-1 計画問題の解を探索する過程で、生成される部分問題の緩和問題をシンプレックス法で解く際に、引数 NEV の値が反復回数の上限として用いられる。引数 NEV に正でない値が入力された場合、既定値の $10 \times M$ が反復回数の上限として用いられる。
- (e) 分枝限定法による最適解の探索の途中で、その時点までに得られている制約条件を満たす解の中で最も目的関数の値を小さくする解をその時点での暫定解と呼ぶ。IERR=5000 または 5500 が返されたときには、処理が打ち切られた時点での暫定解が X に出力される。このときには、得られた解は制約条件は満たすが、真の最適解とは必ずしも一致しない。
- (f) 探索の深さ MP は分枝限定法における解の探索方法を設定するために利用者があたえるパラメータである。MP を大きくするほど、早い時点で良い暫定解が得られる傾向が強い。しかし MP が大きくなるほど、終端されずにメモリ上に保持される部分問題の個数は急激に増える傾向にあるため、NP の値を十分大きく設定しておく必要がある。MP=1 のとき、計算に必要な領域は最小となる。NP の大きさが十分でないために IERR=5500 または 5600 が返されることを防ぐためには、MP の値を 1 に、NP の値を LN+1 に設定すれば良い。
- (g) IERR=5000 または 5100 を出力して処理が打ち切られた場合に、ISW=1 を設定し、NEV をより大きい値に設定し直して、前回の計算結果を利用して計算を継続することができる。この場合、NEV 以外の引数については前回の処理で使用した引数の内容を保存しておく必要がある。
- (h) IERR=5500 または 5600 を出力して処理が打ち切られた場合に、ISW=1 を設定し、NP をより大きい値に設定し直して、前回の計算結果を利用して計算を継続することができる。この場合、NP、IWK および WK 以外の引数については前回の処理で使用した引数の内容を保存しておく必要がある。さらに作業用の配列である IWK と WK については、NP の値に合わせて、適切な大きさの領域を確保し、その先頭の領域に前回の処理の終了時の IWK と WK の内容を格納しておく必要がある。
- (i) 配列 LZ には変数 x_1, x_2, \dots, x_n のうち 0-1 変数である変数の添字を昇順に格納し、変数 LN に 0-1 変数の個数を格納する。例えば、変数が x_1, x_2, \dots, x_8 の 8 個あり、そのうち x_2, x_5, x_8 が 0-1 変数である混合 0-1 計画問題を扱う場合には以下のように設定する。

$$LN = 3$$

$$LZ(1) = 2$$

$$LZ(2) = 5$$

$$LZ(3) = 8$$

(7) 使用例

(a) 問題

制約条件

$$\begin{aligned} x_2 + x_3 + x_{10} &= 1 \\ 6x_1 + 28x_2 + 6x_3 + 6x_4 + 3x_5 + 6x_6 \\ + 21x_7 + 8x_8 - 18x_9 + 12x_{10} + 20x_{11} + 23x_{12} &\leq 60 \\ 7x_1 + 7x_2 + 5x_3 + 2x_4 + 2x_5 + 5x_6 \\ + 4x_7 + 2x_8 - 3x_9 + 3x_{10} + 8x_{11} + 3x_{12} &\leq 20 \\ -x_4 - x_{11} &\leq -1 \\ -x_6 - x_7 - x_{12} &\leq -1 \\ 0 \leq x_1 \leq 2, 0 \leq x_5 \leq 3, 0 \leq x_8 \leq 1, -2 \leq x_9 \leq 0 \\ x_j &= 0, 1 \quad (j = 2, 3, 4, 6, 7, 10, 11, 12) \end{aligned}$$

の下で目的関数

$$f(x) = -13x_1 - 18x_2 - 10x_3 - 8x_4 - 8x_5 - 12x_6 \\ -10x_7 - 8x_8 + 11x_9 - 9x_{10} - 30x_{11} - 10x_{12}$$

を最小にする.

(b) 入力データ

MA= 6, N= 12, M= 5, ME= 1, MP= 4, NP= 50, ER= 0.0 (既定値にセット), NEV= 0 (既定値にセット), ISW= 0, 制約条件の係数を格納する配列 A と B, 目的関数の係数を格納する配列 C, 0-1 変数の添え字を格納する配列 LZ

(c) 主プログラム

```

PROGRAM BMCLMZ
IMPLICIT NONE
!
INTEGER MAO,NO,MO,LNO,MPO,NPO,MEO
INTEGER NIWK,NWK
PARAMETER ( MAO = 6 )
PARAMETER ( NO = 12 )
PARAMETER ( MO = 5 )
PARAMETER ( LNO = 8 )
PARAMETER ( NPO = 10 )
PARAMETER ( MPO = 4 )
PARAMETER ( MEO = 1 )
PARAMETER ( NIWK=MPO+(MPO+3)*(NO+MO-MEO)+MO+(LNO+2)*NPO+7 )
PARAMETER ( NWK=4*MO+(MO+4)*(NO+2*MO-MEO)&
+ (MPO+1)*(MO+1)*(NO-MEO+1)&
+ (MO+1)*(NO+MO-MEO+1)&
+2*LNO+(NO+MO-MEO+2)*(NPO+1)+2)
INTEGER MA,ME,N,M,LN,MP,NP,NEV,IERR
REAL(8) A(MAO,NO),B(MO),XUP(NO),XLOW(NO),C(NO),X(NO)
REAL(8) ER,Y,WK(NWK)
INTEGER LZ(LNO),IWK(NIWK)
INTEGER I,J,ISW
!
READ(5,*) MA,N,M,ME,LN,MP,NP,NEV,ISW,ER
WRITE(6,6000) MA,N,M,ME,LN,MP,NP,NEV,ISW,ER
WRITE(6,6010)
DO 100 I = 1,M
READ(5,*) (A(I,J),J=1,N),B(I)
100 CONTINUE
DO 110 I = 1,M
WRITE(6,6020) (A(I,J),J=1,N),B(I)
110 CONTINUE
WRITE(6,6030)
READ(5,*) (C(J),J=1,N)
DO 120 J = 1,N
WRITE(6,6040) J,C(J)
120 CONTINUE
WRITE(6,6070)
READ(5,*) (LZ(I),I=1,LN)
WRITE(6,6080) (LZ(I),I=1,LN)
WRITE(6,6050)
DO 130 J = 1,N
DO 132 I = 1,LN
IF(LZ(I).EQ.J) THEN
GOTO 135
ENDIF
132 CONTINUE
READ(5,*) XUP(J),XLOW(J)
WRITE(6,6060) J,XUP(J),J,XLOW(J)
135 CONTINUE
130 CONTINUE
CALL DMCLMZ&
(A,MA,N,B,M,ME,XUP,XLOW,LZ,LN,C,MP,NP,ER,NEV,X,Y,ISW,IWK,WK,&
IERR)
WRITE(6,6090)
WRITE(6,6100) IERR
DO 150 J = 1,N
WRITE(6,6110) J,X(J)
150 CONTINUE
WRITE(6,6120) Y
6000 FORMAT(/,/,5X,'** INPUT **',/,/,/,&
11X,'MA = ',I5,8X,',N = ',I5,/,&
11X,'M = ',I5,8X,',ME = ',I5,/,&
11X,'LN = ',I5,8X,',MP = ',I5,/,&
11X,'NP = ',I5,8X,',NEV = ',I5,/,&
11X,'ISW = ',I5,8X,',ER = ',D7.1)
6010 FORMAT(/,5X,'CONSTRAINTS',/,/,&
32X,' A ',31X,'B',/)
6020 FORMAT(12(F5.1),3X,F5.1)
6030 FORMAT(/,5X,'COEFFICIENTS OF THE OBJECTIVE FUNCTION',/)
6040 FORMAT(8X,'C(',I2,') = ',D15.5)
6050 FORMAT(/,5X,'UPPER AND LOWER BOUNDS OF X',/)
6060 FORMAT(8X,'XUP(',I2,') = ',D15.5,2X,'XLOW(',I2,') = ',D15.5)
6070 FORMAT(/,5X,'INDICES OF 0-1 VARIABLES',/)
6080 FORMAT(8X,12(1X,I2),/)
6090 FORMAT(/,5X,'** OUTPUT **',/,/)
6100 FORMAT(8X,'IERR = ',I5)
6110 FORMAT(8X,'X(',I2,') = ',D15.5)

```

```
6120 FORMAT(8X,'Y = ',D15.5)
STOP
END
```

(d) 出力結果

```
** INPUT **
```

```
MA = 6 ,N = 12
M = 5 ,ME = 1
LN = 8 ,MP = 4
NP = 10 ,NEV = 20
ISW = 0 ,ER = 0.1D-11
```

```
CONSTRAINTS
```

												A				B
0.0	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0				1.0	
6.0	28.0	6.0	6.0	3.0	6.0	21.0	8.0	-18.0	12.0	20.0	23.0				60.0	
7.0	7.0	5.0	2.0	2.0	5.0	4.0	2.0	-3.0	3.0	8.0	3.0				20.0	
0.0	0.0	0.0	-1.0	0.0	0.0	0.0	0.0	0.0	0.0	-1.0	0.0				-1.0	
0.0	0.0	0.0	0.0	0.0	-1.0	-1.0	0.0	0.0	0.0	0.0	-1.0				-1.0	

```
COEFFICIENTS OF THE OBJECTIVE FUNCTION
```

```
C( 1) = -0.13000D+02
C( 2) = -0.18000D+02
C( 3) = -0.10000D+02
C( 4) = -0.80000D+01
C( 5) = -0.80000D+01
C( 6) = -0.12000D+02
C( 7) = -0.10000D+02
C( 8) = -0.80000D+01
C( 9) = 0.11000D+02
C(10) = -0.90000D+01
C(11) = -0.30000D+02
C(12) = -0.10000D+02
```

```
INDICES OF 0-1 VARIABLES
```

```
2 3 4 6 7 10 11 12
```

```
UPPER AND LOWER BOUNDS OF X
```

```
XUP( 1) = 0.20000D+01 XLOW( 1) = 0.00000D+00
XUP( 5) = 0.30000D+01 XLOW( 5) = 0.00000D+00
XUP( 8) = 0.10000D+01 XLOW( 8) = 0.00000D+00
XUP( 9) = 0.00000D+00 XLOW( 9) = -0.20000D+01
```

```
** OUTPUT **
```

```
IERR = 0
X( 1) = 0.00000D+00
X( 2) = 0.00000D+00
X( 3) = 0.00000D+00
X( 4) = 0.10000D+01
X( 5) = 0.30000D+01
X( 6) = 0.10000D+01
X( 7) = 0.00000D+00
X( 8) = 0.10000D+01
X( 9) = -0.66667D+00
X(10) = 0.10000D+01
X(11) = 0.00000D+00
X(12) = 0.00000D+00
Y = -0.68333D+02
```

5.6.4 DMCLMC, RMCLMC

ネットワーク上の流れに対する費用の最小化 (最小費用流問題)

(1) 機能

n 個の節点と m 本の枝をもつネットワーク上で、節点 i の流入出量 b_i および有向枝 k の非負の容量 u_k の制約を満たし、かつ全枝の費用の和を最小にする非負の流れ x_k ($k = 1, 2, \dots, m$) とそのときの最小値 $\sum_{k=1}^m c_k x_k$ を求める。

$$\text{目的関数} : \sum_{k=1}^m c_k x_k \rightarrow \text{最小}$$

$$\text{制約条件} : \sum_{tail(k)=i} x_k - \sum_{head(k)=i} x_k = b_i, \quad (i = 1, \dots, n)$$

$$0 \leq x_k \leq u_k, \quad (k = 1, \dots, m)$$

$$\sum_{i=1}^n b_i = 0$$

(2) 使用法

倍精度サブルーチン:

CALL DMCLMC (N, M, B, ITL, IHD, CAP, COST, NEV, X, Y, ISW, IWK, WK, IERR)

単精度サブルーチン:

CALL RMCLMC (N, M, B, ITL, IHD, CAP, COST, NEV, X, Y, ISW, IWK, WK, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入力	節点数 n
2	M	I	1	入力	枝の数 m
3	B	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	N	入力	節点 i の流入出力 b_i
4	ITL	I	M+N	入力	枝 k の始点の節点番号 $tail(k)$ (注意事項 (a) 参照)
				出力	グラフ変形後の始点の節点番号
5	IHD	I	M+N	入力	枝 k の終点の節点番号 $head(k)$ (注意事項 (a) 参照)
				出力	グラフ変形後の終点の節点番号
6	CAP	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	M+N	入力	枝 k の容量 u_k (注意事項 (a), (b) 参照)
				出力	グラフ変形後の容量
7	COST	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	M+N	入力	枝 k の単位流量あたりの費用係数 c_k (注意事項 (a) 参照)
				出力	グラフ変形後の費用係数
8	NEV	I	1	入力	基底木の最大更新回数 (既定値: $N \times 10$) (注意事項 (c) 参照)
				出力	実際の基底木の更新回数
9	X	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	M	出力	全枝の費用の和を最小にする流れ x_k
10	Y	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出力	最終到達点 X での費用の和 $\sum_{k=1}^m c_k x_k$
11	ISW	I	1	入力	処理スイッチ (注意事項 (d) 参照) ISW=0:最初の処理 ISW=1:継続の処理
12	IWK	I	内容参照	ワーク	作業領域 大きさ: $N^2 + 11 \times N + M + 3$
13	WK	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	内容参照	ワーク	作業領域 大きさ: $(2 \times N + M + 1)$
14	IERR	I	1	出力	エラーインディケータ

(4) 制限条件

- (a) $N \geq 2$
- (b) $M \geq 1$
- (c) $B(1) + B(2) + \dots + B(N) = 0.0$
- (d) $1 \leq ITL(k) \leq N \quad (k = 1, \dots, M)$
- (e) $1 \leq IHD(k) \leq N \quad (k = 1, \dots, M)$
- (f) $ITL(k) \neq IHD(k) \quad (k = 1, \dots, M)$ (ループ (始点と終点と同じ枝) は存在しない)
- (g) $CAP(k) \geq 0.0 \quad (k = 1, \dots, M)$
- (h) $NEV > 0$ (既定値にするため 0 または負値を入力する場合を除く)
- (i) $ISW = 0$ または $ISW = 1$

(5) エラーインディケータ

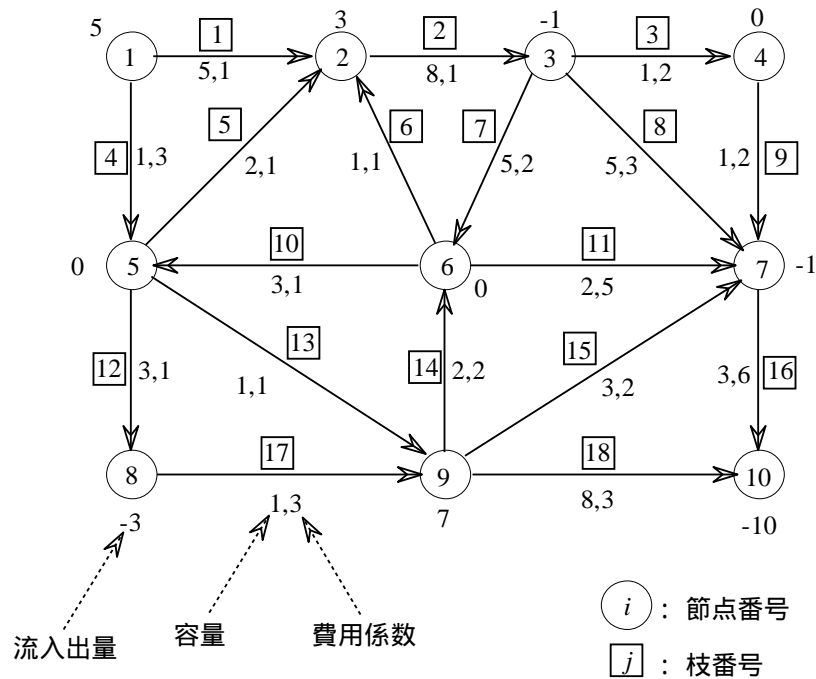
IERR の値	意 味	処 理 内 容
0	正常終了.	
1000	制限条件 (h) を満足しなかった.	既定値にセットして処理する.
1100	制限条件 (i) を満足しなかった.	ISW=0 にセットして処理する.
3000	制限条件 (a) または (b) を満足しなかった.	処理を打ち切る.
3100	制限条件 (c) を満足しなかった.	
3200	制限条件 (d) または (e) を満足しなかった.	
3300	制限条件 (f) を満足しなかった.	
3400	制限条件 (g) を満足しなかった.	
4000	実行可能基底解を求められなかった.	
5000	与えられた基底木の最大更新回数に達しても問題が解けなかった	

(6) 注意事項

- (a) ITL, IHD, CAP および COST には最初の M 個のみ値を入力すればよい. 後の N 個は作業領域として使用される.
- (b) 容量が特に定められていない問題を解く場合は, 適当に大きな正数を容量として入力すればよい. なお, この時, 最終到達点での X の値が定めた容量と一致する場合は, 最適解が求められていない可能性がある. さらに大きな値を容量として入力したほうがよい.
- (c) NEV に 0 または負の値を入力すれば既定値がセットされる.
- (d) 指定した基底木の最大更新回数が少なく IERR=5000 が返されたとき, 途中まで計算した情報を用いて引き続き計算することができる. この処理を行うときには ISW の値を 1, NEV の値を十分に大きな値とし, それ以外の入力値には前回の値をそのまま用いる. また, 前回のワークの情報も用いる.

(7) 使用例

- (a) 問題
以下のようなネットワーク上で各節点の流入出量および各枝の容量の制約を満たしかつ全枝の費用の和を最小にする流れを求める.



(b) 入力データ

N = 10, M = 18, 流入出量を格納する配列 B, 容量を格納する配列 CAP, 枝の節点番号を格納する配列 ITL と IHD, 費用係数を格納する配列 COST, NEV = 0 (既定値セット), ISW = 0

(c) 主プログラム

```

PROGRAM BMCLMC
!
!   IMPLICIT REAL(8) (A-H,0-Z)
!
!   PARAMETER( NMAX = 10 )
!   PARAMETER( MMAX = 18 )
!   DIMENSION B(NMAX), ITAIL(MMAX+NMAX), IHEAD(MMAX+NMAX)
!   DIMENSION CAP(MMAX+NMAX), COST(MMAX+NMAX), X(MMAX)
!   DIMENSION IWK(NMAX**2+11*NMAX+MMAX+3), WK(2*NMAX+MMAX+1)
!
!   READ(5,*) N,M
!   DO 100 I=1,N
!     READ(5,*) B(I)
100 CONTINUE
!   DO 110 K=1,M
!     READ(5,*) ITAIL(K), IHEAD(K), CAP(K), COST(K)
110 CONTINUE
!
!   NEV = 0
!   ISW = 0
!   WRITE(6,6000) N,M,NEV,ISW
!   WRITE(6,6010)
!   DO 120 I=1,N
!     WRITE(6,6020) B(I)
120 CONTINUE
!   WRITE(6,6030)
!   DO 130 K=1,M
!     WRITE(6,6040) ITAIL(K), IHEAD(K), CAP(K), COST(K)
130 CONTINUE
!
!   CALL DMCLMC(N,M,B,ITAIL,IHEAD,CAP,COST,NEV,X,Y,ISW,IWK,WK,IERR)
!
!   WRITE(6,6050) IERR,NEV
!   DO 140 K=1,M
!     WRITE(6,6060) K,X(K)
140 CONTINUE
!   WRITE(6,6070) Y
!
6000 FORMAT(/,&
1X,' ** INPUT **',/,/,&
1X,' N = ',I5,/,&
1X,' M = ',I5,/,&
1X,' NEV = ',I5,/,&
1X,' ISW = ',I5)
6010 FORMAT(/,&
1X,' VALUES OF B')
6020 FORMAT(1X,' ',F7.2)
6030 FORMAT(/,&
1X,' ITAIL IHEAD CAP COST')
6040 FORMAT(1X,' ',I7,' ',I7,' ',F7.2,' ',F7.2)
6050 FORMAT(/,&
1X,' ** OUTPUT **',/,/,&
1X,' IERR = ',I5,/,/,&

```

```

1X, '      NEV = ', I5, /)
6060 FORMAT(1X, '      X(', I2, ') =', F10.2)
6070 FORMAT(/, &
1X, '      Y      =', F10.2)
STOP
END

```

(d) 出力結果

** INPUT **

```

N   =   10
M   =   18
NEV =    0
ISW =    0

```

VALUES OF B

```

5.00
3.00
-1.00
0.00
0.00
0.00
-1.00
-3.00
7.00
-10.00

```

ITAIL	IHEAD	CAP	COST
1	2	5.00	1.00
2	3	8.00	1.00
3	4	1.00	2.00
1	5	1.00	3.00
5	2	2.00	1.00
6	2	1.00	1.00
3	6	5.00	2.00
3	7	5.00	3.00
4	7	1.00	2.00
6	5	3.00	1.00
6	7	2.00	5.00
5	8	3.00	1.00
5	9	1.00	1.00
9	6	2.00	2.00
9	7	3.00	2.00
7	10	3.00	6.00
8	9	1.00	3.00
9	10	8.00	3.00

** OUTPUT **

IERR = 1000

NEV = 13

```

X( 1) =   4.00
X( 2) =   7.00
X( 3) =   0.00
X( 4) =   1.00
X( 5) =   0.00
X( 6) =   0.00
X( 7) =   3.00
X( 8) =   3.00
X( 9) =   0.00
X(10) =   3.00
X(11) =   0.00
X(12) =   3.00
X(13) =   1.00
X(14) =   0.00
X(15) =   0.00
X(16) =   2.00
X(17) =   0.00
X(18) =   8.00

```

Y = 72.00

5.6.5 DMCLCP, RMCLCP

プロジェクトの日程計画に対する費用の最小化 (日程計画問題)

(1) 機能

予定完了時刻までにプロジェクトを最小費用で完了する問題を解く。

(2) 使用法

倍精度サブルーチン:

CALL DMCLCP (N, M, ITL, IHD, TN, TC, CN, CC, TS, NEV, TMAX, TMIN, TIME, TTIME, ES, LS, TF, FF, CMAX, CMIN, COST, TCOST, IWK, WK, IERR)

単精度サブルーチン:

CALL RMCLCP (N, M, ITL, IHD, TN, TC, CN, CC, TS, NEV, TMAX, TMIN, TIME, TTIME, ES, LS, TF, FF, CMAX, CMIN, COST, TCOST, IWK, WK, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入 力	節点の数
2	M	I	1	入 力	作業の数
3	ITL	I	M	入 力	作業 k の始点の節点番号
4	IHD	I	M	入 力	作業 k の終点の節点番号
5	TN	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	M	入 力	作業 k の通常作業時間
6	TC	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	M	入 力	作業 k の特急作業時間
7	CN	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	M	入 力	作業 k の通常作業費用
8	CC	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	M	入 力	作業 k の特急作業費用
9	TS	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	予定完了時刻
10	NEV	I	1	入 力	最小費用流問題の基底木の更新回数 (既定値: $N \times 10$)
				出 力	実際の基底木の更新回数
11	TMAX	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	通常作業時間によるプロジェクトの完了時刻
12	TMIN	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	特急作業時間によるプロジェクトの完了時刻

項番	引数名	型	大きさ	入出力	内 容
13	TIME	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	M	出力	作業 k の作業時間
14	TTIME	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出力	総作業時間
15	ES	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	M	出力	作業 k の最早開始時刻
16	LS	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	M	出力	作業 k の最遅開始時刻
17	TF	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	M	出力	作業 k の総余裕時間
18	FF	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	M	出力	作業 k の自由余裕時間
19	CMAX	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出力	通常作業時間によるプロジェクトの総作業費用
20	CMIN	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出力	特急作業時間によるプロジェクトの総作業費用
21	COST	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	M	出力	作業 k の作業費用
22	TCOST	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出力	総作業費用
23	IWK	I	内容参照	ワーク	作業領域 大きさ: $N^2 + 15 \times N + 6 \times M + 23$
24	WK	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	内容参照	ワーク	作業領域 大きさ: $7 \times N + 12 \times M + 14$
25	IERR	I	1	出力	エラーインディケータ

(4) 制限条件

- (a) $N > 1$
- (b) $M \geq N - 1$
- (c) $1 \leq \text{ITL}(k) < N \quad (k = 1, \dots, M)$
- (d) $1 < \text{IHD}(k) \leq N \quad (k = 1, \dots, M)$
- (e) $\text{ITL}(k) < \text{IHD}(k) \quad (k = 1, \dots, M)$
- (f) $\text{TN}(k) \geq 0.0 \quad (k = 1, \dots, M)$
- (g) $\text{TC}(k) \geq 0.0 \quad (k = 1, \dots, M)$
- (h) $\text{TN}(k) \geq \text{TC}(k) \quad (k = 1, \dots, M)$
- (i) $\text{CN}(k) \leq \text{CC}(k) \quad (k = 1, \dots, M)$
- (j) $\text{NEV} > 0$ (既定値にするため 0 を入力する場合を除く)

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1000	NEV=0 であった.	既定値にセットして処理する.
1100	$TS \geq TMAX$ であった.	$TS=TMAX$ として処理を続ける.
1200	$TS \leq TMIN$ であった.	$TS=TMIN$ として処理を続ける.
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
3100	制限条件 (b) を満足しなかった.	
3200	制限条件 (c) を満足しなかった.	
3300	制限条件 (d) を満足しなかった.	
3400	制限条件 (e) を満足しなかった.	
3500	制限条件 (f) を満足しなかった.	
3600	制限条件 (g) を満足しなかった.	
3700	制限条件 (h) を満足しなかった.	
3800	制限条件 (i) を満足しなかった.	
3900	制限条件 (j) を満足しなかった.	
4000	最小費用流問題において実行可能基底解を求められなかった.	
5000	最小費用流問題において与えられた基底木の最大更新回数に達しても問題が解けなかった.	

(6) 注意事項

- (a) 基底木の更新回数 NEV に 0 の値を入力すれば既定値がセットされる.
- (b) 予定完了時刻 TS に通常作業時間による完了時刻 TMAX より大きい値を入力した場合, 通常作業時間による完了時刻を予定完了時刻として扱う. また, 予定完了時刻 TS に特急作業時間による完了時刻 TMIN より小さい値を入力した場合, 特急作業時間による完了時刻を予定完了時刻として扱う.

(7) 使用例

(a) 問題

以下のような作業リストで表されるプロジェクトの通常作業時間および特急作業時間による完了時刻, 最適日程の各作業の時間, 費用, 最早開始時刻, 最遅開始時刻, 総余裕時間および自由余裕時間, 総作業時間および総作業費用を求める.

作業	始点の 節点	終点の 節点	通常作業 時間	特急作業 時間	通常作業 費用	特急作業 費用
1	1	2	5	3	3	6
2	1	3	8	4	2	3
3	2	4	5	2	1	5
4	2	5	9	4	1	3
5	4	5	0	0	0	0
6	2	6	3	3	4	4
7	3	6	7	3	2	3
8	5	7	5	1	1	5
9	6	7	0	0	0	0
10	6	8	12	6	7	11
11	7	8	7	4	2	7
12	5	9	15	8	5	10
13	7	9	8	2	5	11
14	8	9	2	1	3	4
15	8	10	8	5	2	5
16	9	10	3	2	1	3
17	8	11	13	10	10	12
18	10	11	8	4	6	10

(b) 入力データ

$N=11, M=18$, 始点の節点 ITL, 終点の節点 IHD, 通常作業時間 TN, 特急作業時間 TC, 通常作業費用 CN, 特急作業費用 CC, $TS=36, NEV=N \times 10$.

(c) 主プログラム

```

PROGRAM BMCLCP
! *** EXAMPLE OF DMCLCP ***
IMPLICIT REAL(8) (A-H,O-Z)
!
INTEGER NMAX,MMAX
PARAMETER( NMAX = 11, MMAX = 18 )
INTEGER N,M,ITAIL(MMAX),IHEAD(MMAX),NEV,IERR
INTEGER IWK(NMAX*NMAX+15*NMAX+6*MMAX+23)
REAL(8) TN(MMAX),TC(MMAX),CN(MMAX),CC(MMAX),TS
REAL(8) TMIN,TMAX,TIME(MMAX),TTIME
REAL(8) ES(MMAX),LS(MMAX),TF(MMAX),FF(MMAX)
REAL(8) CMIN,CMA,CCOST(MMAX),TCOST
REAL(8) WK(7*NMAX+12*MMAX+14)
!
DATA ITAIL /1,1,2,2,4,2,3,5,6,6,7,5,7,8, 8, 9, 8,10/
DATA IHEAD /2,3,4,5,5,6,6,7,7,8,8,9,9,9,10,10,11,11/
DATA TN&
/5.0D0,8.0D0,5.0D0,9.0D0,0.0D0,&
3.0D0,7.0D0,5.0D0,0.0D0,12.0D0,&
7.0D0,15.0D0,8.0D0,2.0D0,8.0D0,&
3.0D0,13.0D0,8.0D0/
DATA TC&
/3.0D0,4.0D0,2.0D0,4.0D0,0.0D0,&
3.0D0,3.0D0,1.0D0,0.0D0,6.0D0,&
4.0D0,8.0D0,2.0D0,1.0D0,5.0D0,&
2.0D0,10.0D0,4.0D0/
DATA CN&

```


17	8	11	13.000	10.000	10.000	12.000
18	10	11	8.000	4.000	6.000	10.000

REQUEST TASK TIME = 36.000

** OUTPUT **

IERR = 0

NORMAL COMPLETION TIME = 43.000

CRASH COMPLETION TIME = 23.000

TASK	TASK TIME	TASK COST	EARLIEST START TIME	LATEST START TIME	TOTAL FLOAT	FREE FLOAT
1	5.000	3.000	0.000	0.000	0.000	0.000
2	7.000	2.250	0.000	0.000	0.000	0.000
3	5.000	1.000	5.000	5.000	0.000	0.000
4	5.000	2.600	5.000	5.000	0.000	0.000
5	0.000	0.000	10.000	10.000	0.000	0.000
6	3.000	4.000	5.000	7.000	2.000	2.000
7	3.000	3.000	7.000	7.000	0.000	0.000
8	5.000	1.000	10.000	10.000	0.000	0.000
9	0.000	0.000	10.000	15.000	5.000	5.000
10	12.000	7.000	10.000	10.000	0.000	0.000
11	7.000	2.000	15.000	15.000	0.000	0.000
12	15.000	5.000	10.000	10.000	0.000	0.000
13	8.000	5.000	15.000	17.000	2.000	2.000
14	2.000	3.000	22.000	23.000	1.000	1.000
15	6.000	4.000	22.000	22.000	0.000	0.000
16	3.000	1.000	25.000	25.000	0.000	0.000
17	13.000	10.000	22.000	23.000	1.000	1.000
18	8.000	6.000	28.000	28.000	0.000	0.000

ITERATION COUNT = 19

TOTAL TASK TIME = 36.000

TOTAL TASK COST = 59.850

5.6.6 DMCLTP, RMCLTP

供給地から需要地への輸送費用の最小化 (輸送問題)

(1) 機能

供給地 i ($i = 1, \dots, m$) での供給量を a_i , 需要地 j ($j = 1, \dots, n$) での需要量を b_j , 供給地 i から需要地 j への輸送量を x_{ij} とすると, 制約条件

$$\begin{aligned} \sum_{j=1}^n x_{ij} &= a_i \quad (i = 1, \dots, m) \\ \sum_{i=1}^m x_{ij} &= b_j \quad (j = 1, \dots, n) \\ x_{ij} &\geq 0 \quad (\text{すべての } i, j \text{ に対して}) \end{aligned}$$

に従って総輸送費用

$$Z = \sum_{j=1}^n \sum_{i=1}^m c_{ij} x_{ij}$$

を最小にする x_{ij} を求める.

(2) 使用法

倍精度サブルーチン:

CALL DMCLTP (C, LMC, NS, ND, S, D, NEV, X, Z, ISW1, ISW2, IWK, WK, IERR)

単精度サブルーチン:

CALL RMCLTP (C, LMC, NS, ND, S, D, NEV, X, Z, ISW1, ISW2, IWK, WK, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	C	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	内容参照	入 力	輸送費用 $c_{ij}(i = 1, \dots, m : j = 1, \dots, n)$ (注意事項 (a) 参照) 大きさ: (LMC, (ND + 1))
2	LMC	I	1	入 力	配列 C の整合寸法
3	NS	I	1	入 力	架空の供給地を含めた供給地の数 m (注意事項 (b) 参照)
4	ND	I	1	入 力	架空の需要地を含めた需要地の数 n (注意事項 (b) 参照)
5	S	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NS+1	入 力	各供給地の供給量 $a_i(i = 1, \dots, m)$
6	D	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	ND+1	入 力	各需要地の需要量 $b_j(j = 1, \dots, n)$
7	NEV	I	1	入 力	輸送計画 x_{ij} の最大評価回数
				出 力	実際の評価回数
8	X	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	内容参照	出 力	改善された輸送計画 $x_{ij}(i = 1, \dots, m : j = 1, \dots, n)$ (注意事項 (c),(d) 参照) 大きさ: (LMC, (ND + 1))
9	Z	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	改善された輸送計画の総輸送費用 Z
10	ISW1	I	1	入 力	処理スイッチ (注意事項 (e) 参照) 0: 最初の処理 1: 継続の処理
11	ISW2	I	1	入 力	第一次近似解法選択スイッチ (注意事項 (f) 参照) 0: 北西隅の規則 1: ハウサッカー法
12	IWK	I	内容参照	ワーク	ワーク配列 大きさ: $(12 \times (NS + ND + 1) + 5)$
13	WK	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	内容参照	ワーク	ワーク配列 大きさ: $(3 \times (NS + ND + 2))$
14	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

- (a) $C(i, j) \geq 0$ ($i = 1, \dots, NS; j = 1, \dots, ND$)
- (b) $S(i) > 0$ ($i = 1, \dots, NS$)
- (c) $D(i) > 0$ ($i = 1, \dots, ND$)
- (d) $ISW1, ISW2 = 0$ または 1
- (e) $NS > 2, ND > 2$
- (f) $LMC \geq NS + 1$
- (g) $NEV \geq 0$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1000	総需要量が総供給量よりも多い.	余分を扱う架空の供給地を設定し処理を続ける.
1100	総需要量が総供給量よりも少ない.	余分を扱う架空の需要地を設定し処理を続ける.
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
3100	制限条件 (b) を満足しなかった.	
3200	制限条件 (c) を満足しなかった.	
3300	制限条件 (d) を満足しなかった.	
3400	制限条件 (e) を満足しなかった.	
3500	制限条件 (f) を満足しなかった.	
3600	制限条件 (g) を満足しなかった.	
5000	与えられた最大評価回数に達しても収束しなかった.	その時点の X, Z の値を出力して、処理を打ち切る.

(6) 注意事項

- (a) 輸送費用は、需要地と供給地に対応するよう以下の表の様に配列 C に格納する。以下の表において、例えば $c_{2,1}$ とは供給地 2 から 需要地 1 への単位輸送費用である。

	需要地 1	需要地 2	...	需要地 n	供給量
供給地 1	c_{11}	c_{12}	...	$c_{1,n}$	a_1
供給地 2	c_{21}	c_{22}	...	$c_{2,n}$	a_2
⋮	⋮	⋮	⋮	⋮	⋮
供給地 m	$c_{m,1}$	$c_{m,2}$...	$c_{m,n}$	a_m
需要量	b_1	b_2	...	b_n	

- (b) 配列 C, X は $(ND + 1) \times (NS + 1)$ 個以上、配列 S, D はそれぞれ、 $(NS + 1)$ 個、 $(ND + 1)$ 個以上確保しなければならない。但し、実際に入力する値は、配列 C は $ND \times NS$ 個、配列 S, D はそれぞれ NS 個、 ND 個でよい。
- (c) 改善された輸送計画は、需要地と供給地に対応するよう以下の表のように配列 X に格納される。以下の表において、例えば $x_{1,2}$ とは供給地 1 から 需要地 2 への輸送量である。

	需要地 1	需要地 2	...	需要地 n	(架空の需要地 $n+1$)
供給地 1	$x_{1,1}$	$x_{1,2}$...	$x_{1,n}$	*
供給地 2	$x_{2,1}$	$x_{2,2}$...	$x_{2,n}$	*
...	*
供給地 m	$x_{m,1}$	$x_{m,2}$...	$x_{m,n}$	*
(架空の供給地 $m+1$)	*	*	*	*	

- (d) $IERR = 5000$ のときの x の値は、最適解ではないが、制約条件を満たしている。
- (e) 指定した反復回数が少なく $IERR=5000$ が返されたとき、途中まで計算した情報を用いて、引き続き計算することができる。この処理を行うときには、ISW の値を 1、NEV の値を十分な値とし、それ以外の入力値には前回の出力値をそのまま用いる。また、前回の作業領域の情報も用いる (例題参照)。
- (f) このサブルーチンでは、第一次近似解法として北西隅の規則とハウサッカー法を採用している。ハウサッカー法は北西隅の規則より最適解に近い第一次近似解の値を求めることができ、その後の解の改善の時間を短縮することが可能である。しかし北西隅の規則に比べ、近似解の計算にはより時間がかかる。

(7) 使用例

(a) 問題

輸送費用

$$C = \begin{bmatrix} 6.000 & 2.000 & 6.000 & 1.000 & 3.000 & 3.000 & 7.000 \\ 7.000 & 5.000 & 3.000 & 8.000 & 5.000 & 8.000 & 5.000 \\ 4.000 & 8.000 & 9.000 & 7.000 & 5.000 & 7.000 & 2.000 \\ 4.000 & 7.000 & 3.000 & 6.000 & 7.000 & 3.000 & 4.000 \\ 2.000 & 3.000 & 6.000 & 4.000 & 7.000 & 9.000 & 9.000 \\ 2.000 & 3.000 & 9.000 & 4.000 & 3.000 & 2.000 & 3.000 \end{bmatrix}$$

各需要地での需要量

$$D = [40.000 \quad 73.000 \quad 32.000 \quad 24.000 \quad 32.000 \quad 45.000 \quad 35.000]$$

各供給地での供給量

$$S = [73.000 \quad 29.000 \quad 64.000 \quad 23.000 \quad 76.000 \quad 34.000]$$

のもとで、総輸送費用 Z を最小にする輸送計画 $X(I, J)$ を、ハウサッカー法を用いて求める。なおこの例題では、注意事項 (e) で述べた継続処理を行うために、最大評価回数を $NEV=2$ 回と小さく設定し、 $IERR=5000$ が出力されるようにしている。

(b) 入力データ

(1 回目):配列 C, NS=7, LMC=7, ND=8, 配列 D, 配列 S, NEV=2, ISW1=0, ISW2=1

(2 回目):NEV=2, ISW1=1(他は前回の計算後の値をそのまま入力値とする)

(c) 主プログラム

```

PROGRAM BMCLTP
! *** EXAMPLE OF DMCLTP ***
INTEGER I, J, LMC, MD, MWK
INTEGER ND, NS, NEV, ISW1, ISW2, IERR
PARAMETER( LMC=7, MD=8, MWK=12*(LMC+MD+1)+5, MDWK=3*(LMC+MD+2) )
REAL(8) C(LMC, MD), X(LMC, MD), Z
REAL(8) S(LMC), D(MD), WK(MDWK)
INTEGER IWK(MWK)
DATA ZERO/0.0D0/
!
READ(5,*) ISW1

```

```

      READ(5,*) ISW2
      READ(5,*) NEV
      READ(5,*) NS
      READ(5,*) ND
!
      DO 100 I=1,NS
      DO 110 J=1,ND
        C(I,J)=ZERO
110 CONTINUE
100 CONTINUE
      DO 120 I=1,NS
        S(I) = ZERO
120 CONTINUE
      DO 130 J=1,ND
        D(J) = ZERO
130 CONTINUE
!
      DO 140 I=1,NS
        READ(5,*) ( C(I,J), J=1,ND )
140 CONTINUE
      READ(5,*) ( S(I), I=1, NS )
      READ(5,*) ( D(J), J=1, ND )
!
      WRITE(6,6000)
      WRITE(6,6010) LMC, NS, ND, NEV, ISW1, ISW2
      WRITE(6,6020)
      DO 150 I=1,NS
        WRITE(6,6030) I, ( C(I,J), J=1,ND )
150 CONTINUE
      WRITE(6,6040) ( I,I=1,ND)
      WRITE(6,6050)
      WRITE(6,6060)
      WRITE(6,6070) ( D(J), J=1,ND )
      WRITE(6,6080)
      WRITE(6,6070) ( S(I), I=1,NS )
!
      CALL DMCLTP&
        ( C, LMC, NS, ND, S, D, NEV,&
          X, Z, ISW1, ISW2, IWK, WK, IERR )
!
      WRITE(6,6090)
      WRITE(6,6100)
      WRITE(6,6110) IERR
      WRITE(6,6120) NEV
      WRITE(6,6130) Z
      WRITE(6,6140)
      DO 160 I=1,NS+1
        WRITE(6,6030) I, ( X(I,J), J=1,ND+1 )
160 CONTINUE
      WRITE(6,6040) ( I,I=1,ND+1)
      WRITE(6,6050)
      IF( IERR .EQ. 5000 ) THEN
        NEV = 1000
        ISW1 = 1
!
        CALL DMCLTP&
          ( C, LMC, NS, ND, S, D, NEV,&
            X, Z, ISW1, ISW2, IWK, WK, IERR )
!
        WRITE(6,6150)
        WRITE(6,6110) IERR
        WRITE(6,6120) NEV
        WRITE(6,6130) Z
        WRITE(6,6140)
        DO 170 I=1,NS+1
          WRITE(6,6030) I, ( X(I,J), J=1,ND+1 )
170 CONTINUE
        WRITE(6,6040) ( I,I=1,ND+1)
        WRITE(6,6050)
      ENDIF
!
6000 FORMAT( 1X,/,&
1X, ' *** DMCLTP *** ' )
6010 FORMAT( 1X,/,&
1X, ' ** INPUT **',/,/,&
1X, ' LMC = ',I5,/,&
1X, ' NS = ',I5,/,&
1X, ' ND = ',I5,/,&
1X, ' NEV = ',I5,/,&
1X, ' ISW1 = ',I5,/,&
1X, ' ISW2 = ',I5 )
6020 FORMAT( 1X,/,&
1X, ' TRANSPORTATION COST = ' )
6030 FORMAT( 1X, ' ',I2, ' ',8(2X,F5.2) )
6040 FORMAT( 1X, ' SUPPLY-SITE ',8(I3,4X) )
6050 FORMAT( 1X, ' DEMAND-SITE' )
6060 FORMAT( 1X,/,&
1X, ' DEMAND = ' )
6070 FORMAT( 1X, ' ',8(2X,F5.2) )
6080 FORMAT( 1X,/,&
1X, ' SUPPLY = ' )
6090 FORMAT( 1X,/,/,&
1X, ' ** OUTPUT **' )
6100 FORMAT( 1X,/,&
1X, ' ** FIRST RESULT (ISW1.EQ.0) **' )
6110 FORMAT( 1X, ' IERR = ',I5 )
6120 FORMAT( 1X, ' NEV = ',I5 )
6130 FORMAT( 1X,/,&
1X, ' TOTAL COST = ',F10.4,/)
6140 FORMAT( 1X, ' TRANSPORTATION PLAN = ' )
6150 FORMAT( 1X,/,&

```

```

!          1X,' ** IMPROVED RESULT (ISW1.EQ.1) **',/)
          STOP
          END

```

(d) 出力結果

```

*** DMCLTP ***

```

```

** INPUT **

```

```

LMC   =    7
NS    =    6
ND    =    7
NEV   =    2
ISW1  =    0
ISW2  =    1

```

```

TRANSPORTATION COST =
  1    6.00  2.00  6.00  1.00  3.00  3.00  7.00
  2    7.00  5.00  3.00  8.00  5.00  8.00  5.00
  3    4.00  8.00  9.00  7.00  5.00  7.00  2.00
  4    4.00  7.00  3.00  6.00  7.00  3.00  4.00
  5    2.00  3.00  6.00  4.00  7.00  9.00  9.00
  6    2.00  3.00  9.00  4.00  3.00  2.00  3.00
SUPPLY-SITE  1    2    3    4    5    6    7
DEMAND-SITE

```

```

DEMAND =
  40.00  73.00  32.00  24.00  32.00  45.00  35.00

```

```

SUPPLY =
  73.00  29.00  64.00  23.00  76.00  34.00

```

```

** OUTPUT **

```

```

** FIRST RESULT (ISW1.EQ.0) **

```

```

IERR  = 5000
NEV   = 2

```

```

TOTAL COST = 707.0000

```

```

TRANSPORTATION PLAN =
  1    0.00  37.00  0.00  24.00  3.00  0.00  0.00  9.00
  2    0.00  0.00  20.00  0.00  0.00  0.00  0.00  9.00
  3    0.00  0.00  0.00  0.00  29.00  0.00  35.00  0.00
  4    0.00  0.00  12.00  0.00  0.00  11.00  0.00  0.00
  5   40.00  36.00  0.00  0.00  0.00  0.00  0.00  0.00
  6    0.00  0.00  0.00  0.00  0.00  34.00  0.00  0.00
  7    0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
SUPPLY-SITE  1    2    3    4    5    6    7    8
DEMAND-SITE

```

```

** IMPROVED RESULT (ISW1.EQ.1) **

```

```

IERR  = 1100
NEV   = 3

```

```

TOTAL COST = 680.0000

```

```

TRANSPORTATION PLAN =
  1    0.00  37.00  0.00  24.00  12.00  0.00  0.00  0.00
  2    0.00  0.00  29.00  0.00  0.00  0.00  0.00  0.00
  3    0.00  0.00  0.00  0.00  11.00  0.00  35.00  18.00
  4    0.00  0.00  3.00  0.00  0.00  20.00  0.00  0.00
  5   40.00  36.00  0.00  0.00  0.00  0.00  0.00  0.00
  6    0.00  0.00  0.00  0.00  9.00  25.00  0.00  0.00
  7    0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
SUPPLY-SITE  1    2    3    4    5    6    7    8
DEMAND-SITE

```

5.7 多変数 2 次関数の最小化 (2 次計画)

5.7.1 DMCQSN, RMCQSN

多変数凸型 2 次関数の最小化 (線形制約)

(1) 機能

線形制約の中で, n 次のベクトル $\mathbf{x} = [x_1, \dots, x_n]^T$ の多変数凸型 2 次関数 $f(\mathbf{x})$ を最小にする \mathbf{x} を求める.

$$f(\mathbf{x}) = \mathbf{c}^T \mathbf{x} + (1/2)\mathbf{x}^T G \mathbf{x}$$

制約条件 :

$$\mathbf{a}_i^T \mathbf{x} = b_i \quad (i = 1, 2, \dots, m_e)$$

$$\mathbf{a}_i^T \mathbf{x} \geq b_i \quad (i = m_e + 1, m_e + 2, \dots, m; \quad 0 \leq m_e \leq m)$$

ここで, G は $n \times n$ 正定値行列, $\mathbf{c}^T = [c_1, c_2, \dots, c_n]$ と $\mathbf{a}_i^T = [a_{i,1}, a_{i,2}, \dots, a_{i,n}]$ は n 次ベクトル b_i は定数 ($i = 1, 2, \dots, m$ とする).

(2) 使用法

倍精度サブルーチン:

CALL DMCQSN (A, LNA, M, B, G, C, N, ME, ER, NEV, X, Y, ISW, IWK, WK, IERR)

単精度サブルーチン:

CALL RMCQSN (A, LNA, M, B, G, C, N, ME, ER, NEV, X, Y, ISW, IWK, WK, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	A	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	LNA, M	入 力	ISW=0 のとき制約条件の係数に対応する行列の転置行列 $A^T = (a_{j,i})$ ($i = 1, 2, \dots, m; j = 1, 2, \dots, n$) (注意事項 (a) (d) 参照)
				出 力	変換後の制約条件の係数に対応する行列の転置行列
2	LNA	I	1	入 力	配列 A の整合寸法
3	M	I	1	入 力	制約条件の数 m
4	B	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	M	入 力	ISW=0 のとき制約条件の右辺 $b = (b_i)$ ($i = 1, 2, \dots, m$) (注意事項 (d) 参照)
				出 力	変換後の制約条件の右辺
5	G	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	LNA, N	入 力	目的関数の 2 次の係数行列 G
6	C	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	N	入 力	目的関数の 1 次の係数ベクトル c
7	N	I	1	入 力	変数の数 n
8	ME	I	1	入 力	等式の制約条件の数 m_e
9	ER	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	要求精度 (既定値: $2 \times \sqrt{\text{(誤差判定のための単位)}}$)
10	NEV	I	1	入 力	関数 $f(x)$ の最大評価回数 (既定値: $10 \times N + M$)
				出 力	実際の関数評価回数
11	X	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	N	出 力	最終到達点 x
12	Y	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	最終到達点 x での関数値 $f(x)$
13	ISW	I	1	入 力	処理スイッチ (注意事項 (f) 参照) 0: 最初の処理 1: 継続の処理
14	IWK	I	3	ワーク	作業領域
15	WK	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	内容参照	ワーク	作業領域 大きさ: $2 \times N^2 + 17 \times N + 16 \times M + 16$
16	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

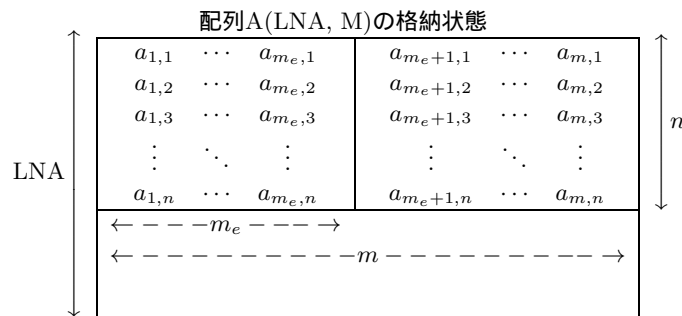
- (a) $0 < N \leq LNA, 0 < M, 0 \leq ME \leq M$
- (b) $ISW = 0$ または $ISW = 1$
- (c) $ER \geq$ 誤差判定のための単位 (既定値にするため, 0.0 を入力する場合は除く)
- (d) $NEV > 0$ (既定値にするため, 0 を入力する場合は除く)

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1000	$n=1$ であった.	処理を続ける.
1500	制限条件 (c) または (d) を満足しなかった.	既定値にセットして処理する.
3000	制限条件 (a) または (b) を満足しなかった.	処理を打ち切る.
3500	等式制約条件が互いに独立でなかった.	処理を打ち切る.
4000	問題が実行不可能であった.	処理を打ち切る.
4000+i	QR 分解の i 列目の処理において対角要素が 0.0 となった. または LL^T 分解の i 列目の処理において, 対角要素が 0.0 以下となった.	処理を打ち切る.
5000	与えられた最大評価回数に達しても収束しなかった.	その時点での X, Y の値を出力して, 処理を打ち切る. (注意事項 (e)(f) 参照)

(6) 注意事項

- (a) $ISW=0$ の時, 等式制約条件に対応する係数 $a_{i,j}$ ($i = 1, 2, \dots, m_e; j = 1, 2, \dots, n$) と不等式制約条件に対応する係数 $a_{i,j}$ ($i = m_e + 1, m_e + 2, \dots, m; j = 1, 2, \dots, n$) は以下の様に配列 A に格納する. ここで, m は制約条件の数, n は変数の数をそれぞれ表す.



備 考
a. $LNA \geq n$ を満たさなければならない.

- (b) 目的関数の 2 次の係数行列 G が正定値でないものは, 本サブルーチンでは最適解を求められない.
- (c) 引数の内容の欄に既定値が記されている場合には, 整数型のときは 0, 実数型のときは 0.0 を入力すれば既定値がセットされる.
- (d) A, B の出力には変換した値が入る.
- (e) 通常 $IERR = 5000$ が返されたとき x の値は制約条件も満たさない.
- (f) 指定した収束回数が少なく $IERR = 5000$ が返されたとき, 途中まで計算した情報を用いて, 引き続き計算することができる. この処理を行うときには, ISW の値を 1, NEV の値を十分な値とし, それ以外の入力値には前回の出力値をそのまま用いる. また, 前回のワークの情報も用いる. (例題参照)

(7) 使用例

(a) 問題

$$\begin{aligned}
 -x_1 - 2x_2 &\geq -10 \\
 -3x_1 - x_2 &\geq -15 \\
 -2x_1 - 3x_2 &\geq -30 \\
 15x_1 - 13x_2 &\geq 0 \\
 x_1 &\geq 0 \\
 x_2 &\geq 0
 \end{aligned}$$

のもとで

$$f(x) = 1/2(5x_1^2 + 6x_1x_2 + 5x_2^2) - 95x_1 - 105x_2$$

を最小にする.

なお, この例では, 注意事項 (f) で述べた継続処理を行うために, 最大評価回数 NEV=3 と小さく設定し, IERR=5000 が出力されるようにしている.

(b) 入力データ

(1 回目): N=2, M=6, ME=0, LNA=11, NEV=3, ISW=0, ER=0.0

配列 A, B, G, C

(2 回目以降): NEV=3, ISW=1

(他は, 前回の計算後の値を, そのまま入力値とする)

(c) 主プログラム

```

PROGRAM BMCQSN
! *** EXAMPLE OF DMCQSN ***
IMPLICIT REAL(8) (A-H,O-Z)
PARAMETER ( MO = 6 )
PARAMETER ( NO = 2 )
PARAMETER ( LNA = 11 )
PARAMETER ( NWK = (2*NO+17)*NO+16*(MO+1) )
DIMENSION A(LNA,MO),B(MO),G(LNA,NO),C(NO),X(NO)
DIMENSION WK(NWK),IWK(3)
!
WRITE(6,1000)
READ(5,*) N,M,ME,ER,NEV,ISW
WRITE(6,1100) N,M,ME
WRITE(6,1300)
DO 10 I = 1,N
  READ(5,*) ( A(I,J),J=1,M )
  WRITE(6,1200) ( A(I,J),J=1,M )
10 CONTINUE
WRITE(6,1400)
READ(5,*) ( B(I),I=1,M )
WRITE(6,1200) ( B(I),I=1,M )
WRITE(6,1500)
DO 20 I = 1,N
  READ(5,*) ( G(I,J),J=1,N )
  WRITE(6,1200) ( G(I,J),J=1,N )
20 CONTINUE
WRITE(6,1600)
READ(5,*) ( C(I),I=1,N )
WRITE(6,1200) ( C(I),I=1,N )
CALL DMCQSN&
  (A,LNA,M,B,G,C,N,ME,ER,NEV,X,Y,ISW,IWK,WK,IERR)
WRITE(6,1700)
WRITE(6,1800) IERR
WRITE(6,1900)
WRITE(6,2200) ( I,X(I),I=1,N )
!
30 READ(5,*) NEV,ISW
CALL DMCQSN&
  (A,LNA,M,B,G,C,N,ME,ER,NEV,X,Y,ISW,IWK,WK,IERR)
WRITE(6,2100)
WRITE(6,1800) IERR
WRITE(6,1900)
WRITE(6,2200) ( I,X(I),I=1,N )
WRITE(6,2000) Y
IF (IERR.EQ.5000) GOTO 30
STOP
!
1000 FORMAT(' ',/,6X,'*** DMCQSN ***',/,&
7X,'** INPUT **')
1100 FORMAT(9X,'N =',I3,8X,'M =',I3,8X,'ME =',I3)

```



```

1200 FORMAT(8X,6(F7.1))
1300 FORMAT(8X,'** MATRIX A **')
1400 FORMAT(8X,'** VECTOR B **')
1500 FORMAT(8X,'** MATRIX G **')
1600 FORMAT(8X,'** VECTOR C **')
1700 FORMAT(7X,'** OUTPUT **',/,&
           8X,'** FIRST RESULT (ISW.EQ.0) **')
1800 FORMAT(9X,'IERR =',I5)
1900 FORMAT(8X,'** VECTOR X **')
2000 FORMAT(9X,'Y =',D18.10)
2100 FORMAT(7X,'** OUTPUT **',/,&
           8X,'** IMPROVED RESULT (ISW.NE.0) **')
2200 FORMAT(8X,'X(',I2,') =',D18.10)
      END

```

(d) 出力結果

```

*** DMCQSN ***
** INPUT **
  N = 2           M = 6           ME = 0
** MATRIX A **
  -1.0  -3.0  -2.0  15.0   1.0   0.0
  -2.0  -1.0  -3.0 -13.0   0.0   1.0
** VECTOR B **
  -10.0 -15.0 -30.0   0.0   0.0   0.0
** MATRIX G **
   5.0   3.0
   3.0   5.0
** VECTOR C **
  -95.0 -105.0
** OUTPUT **
** FIRST RESULT (ISW.EQ.0) **
  IERR = 5000
** VECTOR X **
X( 1) = 0.2142857143D+01
X( 2) = 0.8571428571D+01
** OUTPUT **
** IMPROVED RESULT (ISW.NE.0) **
  IERR = 0
** VECTOR X **
X( 1) = 0.4000000000D+01
X( 2) = 0.3000000000D+01
  Y = -0.5965000000D+03

```

5.7.2 DMCQLM, RMCQLM

多変数広義凸型 2 次関数の最小化 (線形制約)

(1) 機能

制約条件

$$\sum_{j=1}^n a_{i,j}x_j = b_i \quad (i = 1, \dots, m_e)$$

$$\sum_{j=1}^n a_{i,j}x_j \geq b_i \quad (i = m_e + 1, \dots, m)$$

$$x_i \geq 0 \quad (i = 1, \dots, n)$$

の下で目的関数

$$f(x) = \frac{1}{2}x^T Gx + c^T x$$

を最小にする x とその時の目的関数の値 $f(x)$ を求める. ここで, G は $n \times n$ 半正定値行列, $c^T = [c_1, c_2, \dots, c_n]$, $a_i^T = [a_{i,1}, a_{i,2}, \dots, a_{i,n}]$ は n 次のベクトル ($i = 1, \dots, m$).

(2) 使用法

倍精度サブルーチン:

CALL DMCQLM (A, NA, N, B, M, ME, G, NG, C, NEV, X, Y, ISW, IW, W, IERR)

単精度サブルーチン:

CALL RMCQLM (A, NA, N, B, M, ME, G, NG, C, NEV, X, Y, ISW, IW, W, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	A	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NA,N	入 力	ISW=0 のとき制約条件の係数に対応する行列 $A = (a_{i,j}) (i = 1, 2, \dots, m; j = 1, 2, \dots, n)$
2	NA	I	1	入 力	配列 A の整合寸法
3	N	I	1	入 力	変数の数 n
4	B	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	M	入 力	制約条件の右辺の定数 b_i を成分に持つ m 次元ベクトル
5	M	I	1	入 力	制約条件の数 m
6	ME	I	1	入 力	等式制約条件の数 m_e
7	G	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NG,N	入 力	目的関数の 2 次の係数行列 G
8	NG	I	1	入 力	配列 G の整合寸法
9	C	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	N	入 力	目的関数の 1 次の係数ベクトル c
10	NEV	I	1	入 力	最大反復回数 (注意事項 (c) 参照)
				出 力	計算終了に要した反復回数
11	X	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	N	出 力	最適解 x^*
12	Y	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	最適解 x に対する目的関数の値 $f(x)$
13	ISW	I	1	入 力	処理スイッチ (注意事項 (d) 参照) 0:最初の処理 1:継続処理
14	IW	I	内容参照	ワーク	作業領域 大きさ: $2 \times N - ME + M + 2$
15	W	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	内容参照	ワーク	作業領域 大きさ: $(N + M - ME + 2) \times (N + M - ME) \times 2 + (M + ME) \times (N + 1) + N$
16	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

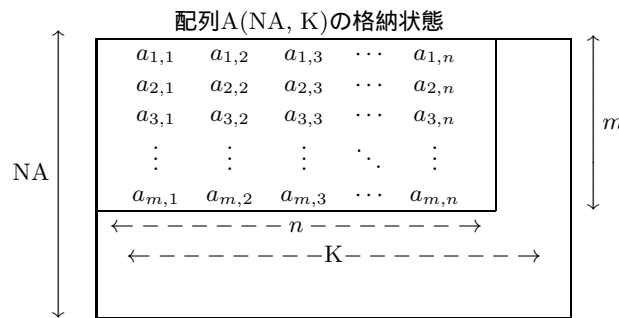
- (a) $NA \geq M, NG \geq N$
- (b) $NG, NA, N > 0$
- (c) $M, ME \geq 0$
- (d) $M \geq ME$
- (e) $N \geq ME$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
1000	N=1 であった.	処理を続ける.
1500	M=0 であった.	
3000	制限条件 (a)~(e) のいずれかを満足しなかつた.	処理を打ち切る.
3500	等式制約条件を満たす解が存在しない.	
4000	最適解が存在しないことがわかった.	
5000	与えられた反復回数内で解が求められなかつた.	

(6) 注意事項

- (a) ISW=0 の時, 制約条件に対応する係数 $a_{i,j}$ ($i = 1, 2, \dots, m; j = 1, 2, \dots, n$) は以下の様に配列 A に格納する. ここで, m は制約条件の数, n は変数の数をそれぞれ表す.



備 考

- a. $K \geq n$ を満たさなければならない.

- (b) 目的関数の 2 次の係数行列 G が半正定値でないものは, 本サブルーチンでは最適解を求められない.
- (c) 引数 NEV に正でない値が入力された場合, 既定値として $NEV = N - M - ME$ にセットし直す.
- (d) 指定した反復回数が少なく IERR=5000 が返されたとき, 途中まで計算した情報を用いて計算する. この処理を行うときは ISW=1 として, 前回の出力値を入力値として用いる. このとき作業配列 W, IW の内容を保持しておかなければならない. なお, 反復回数がすでに $N + M - ME$ に達している場合には, さらに反復を繰り返しても解は求められない.

(7) 使用例

(a) 問題

制約条件

$$\begin{aligned} -x_1 - 2x_2 &\geq -10 \\ -3x_1 - x_2 &\geq -15 \\ -2x_1 - 3x_2 &\geq -30 \\ 15x_1 - 13x_2 &\geq 0 \\ x_1 &\geq 0 \\ x_2 &\geq 0 \end{aligned}$$

のもとで

$$f(x) = \frac{1}{2}(5x_1^2 + 6x_1x_2 + 5x_2^2) - 95x_1 - 105x_2$$

を最小にする. なお, この例では, 注意事項 (d) で述べた継続処理を行うために, 最大評価回数 NEV=2 と小さく設定し, IERR=5000 が出力されるようにしている.

(b) 入力データ

(1 回目): N=2, M=4, ME=0, NA=11, NEV=0

配列 A, B, G, C

(2 回目以降): NEV=0, ISW=1

(他は前回の計算後の値を, そのまま入力値とする.)

(c) 主プログラム

```

PROGRAM BMCQLM
IMPLICIT REAL(8) (A-H,O-Z)
PARAMETER ( NO = 2)
PARAMETER ( MO = 4)
PARAMETER ( MEO = 0)
PARAMETER ( NIWK = 2*NO-MEO+MO+2)
PARAMETER ( NWK = 2*(NO+MO-MEO+2)*(NO+MO-MEO)+(MO+MEO)*(NO+1)+NO)
PARAMETER ( NA = 11, NG = 8)
DIMENSION A(NA,NO),B(MO),G(NG,NO),C(NO),X(NO)
DIMENSION W(NWK),IW(NIWK)
!
WRITE(6,1000)
READ(5,*) N,M,ME,NEV,ISW
WRITE(6,1100) N,M,ME,NEV,ISW
WRITE(6,1300)
DO 10 I = 1,M
  READ(5,*) ( A(I,J),J=1,N )
  WRITE(6,1200) ( A(I,J),J=1,N )
10 CONTINUE
WRITE(6,1400)
READ(5,*) ( B(I),I=1,M )
WRITE(6,1200) ( B(I),I=1,M )
WRITE(6,1500)
DO 20 I = 1,N
  READ(5,*) ( G(I,J),J=1,NG )
  WRITE(6,1200) ( G(I,J),J=1,NG )
20 CONTINUE
WRITE(6,1600)
READ(5,*) ( C(I),I=1,N )
WRITE(6,1200) ( C(I),I=1,N )
CALL DMCQLM&
  (A,NA,N,B,M,ME,G,NG,C,NEV,X,Y,ISW,IW,W,IERR)
WRITE(6,1700)
WRITE(6,1800) IERR
READ(5,*) NEV,ISW
CALL DMCQLM&
  (A,NA,N,B,M,ME,G,NG,C,NEV,X,Y,ISW,IW,W,IERR)
WRITE(6,1900)
WRITE(6,1800) IERR
WRITE(6,2000)
WRITE(6,2200) ( I,X(I),I=1,N )
WRITE(6,2100) Y
STOP
!
1000 FORMAT(' ',/ ,6X,'*** DMCQLM ***',/,&
  7X,'** INPUT **')
1100 FORMAT(6X,'N =',I3,5X,'M =',I3,5X,'ME =',I3&
  ,5X,'NEV =',I3,5X,'ISW =',I3)
1200 FORMAT(8X,6(F8.2))
1300 FORMAT(8X,'** MATRIX A **')

```

```

1400 FORMAT(8X,'** VECTOR B **')
1500 FORMAT(8X,'** MATRIX G **')
1600 FORMAT(8X,'** VECTOR C **')
1700 FORMAT(7X,'** OUTPUT **',/,&
      8X,'** FIRST RESULT (ISW .EQ. 0) **')
1800 FORMAT(9X,'IERR =',I5)
1900 FORMAT(7X,'** OUTPUT **',/,&
      8X,'** IMPROVED RESULT (ISW .EQ. 1) **')
2000 FORMAT(8X,'** VECTOR X **')
2100 FORMAT(9X,'Y =',D18.10)
2200 FORMAT(8X,'X(',I2,') =',D18.10)
      END

```

(d) 出力結果

```

*** DMCQLM ***
** INPUT **
N = 2      M = 4      ME = 0      NEV = 2      ISW = 0
** MATRIX A **
  -1.00   -2.00
  -3.00   -1.00
  -2.00   -3.00
  15.00  -13.00
** VECTOR B **
 -10.00  -15.00  -30.00   0.00
** MATRIX G **
   5.00   3.00
   3.00   5.00
** VECTOR C **
 -95.00 -105.00
** OUTPUT **
** FIRST RESULT (ISW .EQ. 0) **
  IERR = 5000
** OUTPUT **
** IMPROVED RESULT (ISW .EQ. 1) **
  IERR = 0
** VECTOR X **
X( 1) = 0.400000000D+01
X( 2) = 0.300000000D+01
  Y = -0.596500000D+03

```

5.7.3 DMCQAZ, RMCQAZ

制約無し 0-1 多変数 2 次関数の最小化 (0-1 無制約 2 次計画問題)

(1) 機能

n 次の 0-1 ベクトル $x = [x_1, \dots, x_n]^T$ ($x_i \in \{0, 1\}$) の多変数 2 次関数 $f(x)$ を最小にする x を求める.

$$f(x) = c^T x + (1/2)x^T G x$$

ここで, G は $n \times n$ 行列, $c^T = [c_1, c_2, \dots, c_n]$ は n 次ベクトルとする.

(2) 使用法

倍精度サブルーチン:

CALL DMCQAZ (G, NA, N, C, IR1, IR2, IPM, RPM, IX, Y, IWK, WK, IERR)

単精度サブルーチン:

CALL RMCQAZ (G, NA, N, C, IR1, IR2, IPM, RPM, IX, Y, IWK, WK, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	G	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	NA,N	入 力	目的関数の 2 次の係数行列 G
2	NA	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	配列 G の整合寸法
3	N	I	1	入 力	変数の数 n
4	C	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	N	入 力	目的関数の 1 次の係数ベクトル c
5	IR1	I	1	入 力	乱数の初期値 (注意事項 (a), (b) 参照)
				出 力	次の乱数の初期値
6	IR2	I	1	入 力	乱数の初期値 (注意事項 (a), (b) 参照)
				出 力	次の乱数の初期値
7	IPM	I	12	入 力	整数パラメータ
				出 力	IPM (2-11) : 実際に使用された整数パラメータの値 IPM (12) : 分枝限定法における解の評価回数
8	RPM	I	5	入 力	整数パラメータ
				出 力	RPM (2-5) : 実際に使用された整数パラメータの値
9	IX	I	N	入 力	初期解 $x^{(0)}$ (IPM (2) $\neq 1$ のときのみ)
				出 力	最終到達点 x
10	Y	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	最終到達点 x での関数値 $f(x)$
11	IWK	I	内容参照	ワーク	作業領域 大きさ: $N \times (\text{MAX}(\text{IPM}(7), \text{IPM}(10)) + 3) + \text{IPM}(10) + 13$
12	WK	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	内容参照	ワーク	作業領域 大きさ: $3 \times N^2 + 12 \times N + \text{MAX}(\text{IPM}(10), N) + (N + 1) \times \text{IPM}(10) + 9$
13	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $0 < N \leq NA, 0 < N$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) を満足しなかった.	処理を打ち切る.
4000	計算誤差により分枝限定法による最適解の探索が, 続行できなくなった	その時点での IX, Y の値を出力して, 処理を打ち切る.
5000	緩和問題の解が与えられた最大評価回数に達しても得られなかった.	
5500	分枝限定法で終端されない部分問題が与えられた個数に達した,	
5600	与えられた評価回数で分枝限定法の計算が終了しなかった.	

(6) 注意事項

- (a) IR1, IR2 は奇数が望ましい.
- (b) IR1, IR2 に前回の出力値を用いることによって, 前回発生した乱数列に続く乱数列を得ることができる.
- (c) 最大値探索を行いたい場合は, $-f(x)$ の最小値探索を行えば良い. このとき, Y の値を正負逆にしたものが最大値である.
- (d) IERR = 4000, 5000, 5500 または 5600 が返されて, 計算をやり直す場合には, 最初の計算で得られた解 x を次の計算での初期解として用いれば効率的な探索が出来る. この処理を行うときには, IPM (1) と IPM (2) をともに 0 以外の値に設定し, それ以外の入力値には前回の出力値をそのまま用いる.
- (e) n が大きい場合には分枝限定法による解の探索には多大な計算時間を要する. このような場合にはパラメータ IPM (6) を 0 以外の値に設定することにより, 最適解の発見的探索のみを行い, 得られた解を近似的な最適解として利用できる. ただし, 発見的探索で得られた解が真の最適解のどの程度良い近似となっているかは保証されない.
- (f) パラメータ IPM (8) を大きく設定すると, 分枝限定法における下界値テストの精度が良くなり, 探索回数を減らせる場合がある. しかし, 1 回の下界値テストに要する計算時間がこのパラメータの値を増やすと指数的に増大するため, 通常はこのパラメータの値は 1 に設定しておけば良い.
- (g) パラメータ RPM (2) を大きく設定すると, 計算時間は増大するが, 数値的には安定し, IERR = 4000 や 5000 が出力されにくくなる.

表 5-1 パラメータ表 (0-1 無制約 2 次計画問題)

引数名	既定値	内 容
IPM (1)	-	既定値の設定 0:整数パラメータ IPM (2-12) を既定値に設定する. 0 以外:整数パラメータ IPM (2-12) の値を利用者が設定する.
IPM (2)	1	初期解設定スイッチ 0:初期解 $x^{(0)}$ が自動的に設定される. 0 以外:初期解 $x^{(0)}$ を利用者が設定する.
IPM (3)	0	発見的探索方法選択スイッチ 0:模擬焼なまし法で初期解を求め, それをタブー探索で改良する. 1:模擬焼なまし法を用いる. 2:タブー探索を用いる. それ以外:発見的探索を行わない.
IPM (4)	10000	模擬焼なまし法での探索回数. (0 以下に設定すると既定値を用いる.)
IPM (5)	100	タブー探索での探索回数. (0 以下に設定すると既定値を用いる.)
IPM (6)	0	分枝限定法による解の探索を行うかどうかの選択スイッチ. 0:分枝限定法による解の探索を行う. 0 以外:分枝限定法による解の探索を行わない.
IPM (7)	IPM (5)	タブーリストの長さ. (0 以下に設定すると既定値を用いる.)
IPM (8)	1	分枝限定法における部分問題の下界値計算に用いる自由変数の個数. (0~10 以外に設定すると既定値を用いる.)
IPM (9)	1	緩和 2 次計画問題の計算における最大反復回数. (0 以下に設定すると既定値を用いる.)
IPM (10)	内容参照	分枝限定法における活性部分問題の個数の最大値. 既定値:MAX(N,IPM (7)) (0 以下に設定すると既定値を用いる.)
IPM (11)	1	分枝限定法における探索の深さ. (0 以下に設定すると既定値を用いる.)
IPM (12)	100×N	分枝限定法における最大評価回数. (0 以下に設定すると既定値を用いる.)

表 5-1 パラメータ表 (0-1 無制約 2 次計画問題) (続)

引数名	既定値	内 容
RPM (1)	-	既定値の設定 0.0:実数パラメータ RPM (2-5) を既定値に設定する. 0.0 以外:実数パラメータ RPM (2-5) の値を利用者が設定する.
RPM (2)	内容参照	緩和 2 次計画問題の計算における要求精度. 既定値: $\sqrt{\text{誤算判定の単位}}$ (0.0 以下に設定すると既定値を用いる.)
RPM (3)	1.0	係数行列 G を正値対称行列に変換した後の最小固有値 β (0.0 以下に設定すると既定値を用いる.)
RPM (4)	50000.0	模擬焼なまし法における初期温度 T_0 (0.0 未満に設定すると既定値を用いる.)
RPM (5)	0.999	模擬焼なまし法における温度低下率 α (0.0 以下または 1.0 より大きい値に設定すると既定値を用いる.)

(7) 使用例

(a) 問題

行列 G とベクトル c がそれぞれ,

$$G = \begin{pmatrix} 1 & -2 & -1 & 3 & 2 & 0 \\ -1 & 4 & 2 & 1 & 5 & -4 \\ 0 & 1 & 3 & 0 & -3 & 2 \\ 2 & 5 & 0 & -3 & 1 & 3 \\ 1 & 1 & 5 & 3 & -3 & 2 \\ 4 & 5 & -1 & -1 & 3 & 1 \end{pmatrix}$$

$$c = (1, 1, 1, -1, -1, -1)^T$$

で与えられるとき,

$$f(x) = c^T x + \frac{1}{2} x^T G x$$

を最小にする 0-1 ベクトル x^* とそのときの関数値 $f(x^*)$ を求める.

(b) 入力データ

NA=7, N=6, IR1=1, IR2=1, IPM (1) =0, RPM (1) =0.0, 係数行列 G , 係数ベクトル c

(c) 主プログラム

```

PROGRAM BMCLMZ
IMPLICIT NONE
!
INTEGER NA
PARAMETER ( NA = 7 )
REAL(8) G(NA,NA),C(NA),RPM(5),Y,WK(989)
INTEGER N,IR1,IR2,IPM(12),IX(NA),IWK(731),IERR
INTEGER I,J
!
READ(5,*) N
READ(5,*) IR1,IR2
IPM(1) = 0
RPM(1) = 0.0D0
WRITE(6,6000) NA,N,IR1,IR2
WRITE(6,6010)
DO 100 I=1,N
  READ(5,*) (G(I,J),J=1,N)
100 CONTINUE
DO 110 I = 1,N
  WRITE(6,6020) (G(I,J),J=1,N)
110 CONTINUE

```

```

WRITE(6,6030)
READ(5,*) (C(J),J=1,N)
WRITE(6,6020) (C(J),J=1,N)
CALL DMCQAZ(G,NA,N,C,IR1,IR2,IPM,RPM,IX,Y,IWK,WK,IERR)
WRITE(6,6090)
WRITE(6,6100) IERR
DO 150 J = 1,N
  WRITE(6,6110) J,IX(J)
150 CONTINUE
  WRITE(6,6120) Y
!
6000 FORMAT(3X,'** DMCQAZ **',/,/,&
  5X,'** INPUT **',/,/,&
  7X,'NA = ',I5,8X,' N = ',I5,/,&
  7X,'IR1 = ',I5,8X,' IR2 = ',I5)
6010 FORMAT(/,7X,'MATRIX G')
6020 FORMAT(/,9X,6(F5.1))
6030 FORMAT(/,7X,'VECTOR C')
6090 FORMAT(/,5X,'** OUTPUT **',/)
6100 FORMAT(8X,'IERR = ',I5,/)
6110 FORMAT(8X,'IX(',I2,') = ',I5)
6120 FORMAT(/,8X,'Y = ',D15.5)
!
  STOP
  END

```

(d) 出力結果

```

** DMCQAZ **

** INPUT **

NA =      7          N =      6
IR1 =     1          IR2 =     1

MATRIX G
  1.0 -2.0 -1.0  3.0  2.0  0.0
 -1.0  4.0  2.0  1.0  5.0 -4.0
  0.0  1.0  3.0  0.0 -3.0  2.0
  2.0  5.0  0.0 -3.0  1.0  3.0
  1.0  1.0  5.0  3.0 -3.0  2.0
  4.0  5.0 -1.0 -1.0  3.0  1.0

VECTOR C
  1.0  1.0  1.0 -1.0 -1.0 -1.0

** OUTPUT **

IERR =      0

IX( 1) =     0
IX( 2) =     0
IX( 3) =     0
IX( 4) =     1
IX( 5) =     1
IX( 6) =     0

Y =      -0.30000D+01

```

5.8 制約付き多変数関数の最小化 (非線形計画)

5.8.1 DMSQPM, RMSQPM

制約付き多変数関数の最小化 (非線形制約)

(1) 機能

m 個の不等式制約条件

$$g_i(\mathbf{x}) \leq 0, \quad i = 1, \dots, m$$

と, ℓ 個の等式制約条件

$$h_j(\mathbf{x}) = 0, \quad j = 1, \dots, \ell$$

のもとで, 多変数関数 $f(\mathbf{x})$ を局所的に最小化する \mathbf{x}^* とそのときの関数値 $f(\mathbf{x}^*)$ を求める.

(2) 使用法

倍精度サブルーチン:

CALL DMSQPM (F, GF, HF, M, ML, X, N, ER, NEV, Y, IWK, WK, IERR)

単精度サブルーチン:

CALL RMSQPM (F, GF, HF, M, ML, X, N, ER, NEV, Y, IWK, WK, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	F	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	—	入 力	関数 $f(x)$ を定義する関数 F(X) の関数副プログラム名
2	GF	—	—	入 力	不等式制約条件を与えるサブルーチン GF(X, N, G, MM) の手続き名 (MM=M-ML)
3	HF	—	—	入 力	等式制約条件を与えるサブルーチン HF(X, N, H, ML) の手続き名
4	M	I	1	入 力	制約条件の数 $m + \ell$
5	ML	I	1	入 力	等式制約条件の数 ℓ
6	X	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	N	入 力	探索点の初期値 x_0
				出 力	最適解 x^*
7	N	I	1	入 力	x の成分の数
8	ER	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	0 判定のための単位 (既定値: $2, \sqrt{\text{(誤差判定のための単位)}}$)
9	NEV	I	1	入 力	x の最大更新回数 (既定値: 100)
				出 力	実際の x の更新回数
10	Y	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	最終到達点 x^* での関数値 $f(x^*)$
11	IWK	I	M+3	ワーク	作業領域
12	WK	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	内容参照	ワーク	作業領域 大きさ: $4 \times N \times N + 2 \times M \times N + 21 \times N + 108 \times M$
13	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

- (a) $M > 0$ (制約条件のない問題を解くため, 0 を入力する場合を除く)
 (b) $ML \geq 0$
 (c) $N > 0$
 (d) $N \geq ML$
 (e) $M \geq ML$
 (f) $ER > 0.0$ (既定値にするため, 0.0 を入力する場合を除く)
 (g) $NEV \geq 0$ (既定値にするため, 負値を入力する場合を除く)

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容	
0	正常終了.		
1000	制限条件 (f), (g) を満足しなかった.	既定値にセットして処理する.	
1500	M=0 であった.	処理を続ける.	
3000	制限条件 (a) から (e) を満足しなかった.	処理を打ち切る.	
3500	部分 2 次計画問題の等号制約条件が互いに独立でなかった.	その時点での X, Y の値を出力して処理を打ち切る. 得られた X は制約条件を満たす.	
3600	部分 2 次計画問題が実行不可能であった.		
3700	部分 2 次計画問題のための QR 分解の計算において分解される行列の対角要素が 0.0 となった (この行列はランク落ちしている). または部分 2 次計画問題のための LL^T 分解の計算において分解される行列の対角要素が 0.0 以下となった (この行列は特異に近い).		
3800	部分 2 次計画問題の解が収束しなかった.		
3900	直線探索で刻み幅が 0.0 以下になった.		
4000	部分 2 次計画問題の等号制約条件が互いに独立でなかった.		
4100	部分 2 次計画問題が実行不可能であった.		
4200	部分 2 次計画問題のための QR 分解の計算において分解される行列の対角要素が 0.0 となった (この行列はランク落ちしている). または部分 2 次計画問題のための LL^T 分解の計算において分解される行列の対角要素が 0.0 以下となった (この行列は特異に近い).		その時点での X, Y の値を出力して処理を打ち切る. 得られた X は制約条件を満たさない.
4300	部分 2 次計画問題の解が収束しなかった.		
4400	直線探索で刻み幅が 0.0 以下になった.		
4500	与えられた X の最大更新回数に達しても問題が解けなかった		
5000	与えられた X の最大更新回数に達しても問題が解けなかった		

(6) 注意事項

- (a) 引数第 1 項 F の実際の名前は, 使用者側のプログラムで EXTERNAL 文を用いて宣言し, F の実際の名前の関数副プログラムを作っておかなければならない.

関数副プログラム (倍精度) の作り方は以下に示す通りである.

```
REAL(8) FUNCTION F(X)
REAL(8) X
DIMENSION X(N)
F = f(x)
RETURN
END
```

- (b) 引数第 2, 3 項 GF, HF の実際の名前は, 使用者側のプログラムで EXTERNAL 文を用いて宣言し, GF, HF の実際の名前のサブルーチン副プログラムを作っておかなければならない.

サブルーチン副プログラム (倍精度) の作り方は以下に示す通りである.

```
SUBROUTINE GF(X, N, G, MM)
INTEGER N, MM
REAL(8) X, G
DIMENSION X(N), G(MM)
G(1) = g1(x)
G(2) = g2(x)
  ⋮
G(MM) = gMM(x)
RETURN
END

SUBROUTINE HF(X, N, H, ML)
INTEGER N, ML
REAL(8) X, H
DIMENSION X(N), H(ML)
H(1) = h1(x)
H(2) = h2(x)
  ⋮
H(ML) = hML(x)
RETURN
END
```

なお, ここで引数 MM の値は不等式制約条件の数である.

- (c) 指定した x の最大更新回数が少なく IERR=5000 が返されたとき, 途中まで計算した情報を用いて計算する. この処理を行うときは前回の X の出力値を入力値として用いる.

(7) 使用例

(a) 問題

制約条件

$$\begin{aligned} -x_1 + x_2 &\leq 0 \\ x_1^2 + x_2^2 + x_3^2 - 1.0 &= 0 \end{aligned}$$

の下で

$$f(\mathbf{x}) = -x_3$$

を最小にする.

(b) 入力データ

目的関数の値を計算する関数副プログラム名: F

不等式制約条件を与えるサブルーチン名: GF

等式制約条件を与えるサブルーチン名: HF

N=3, M=2, ML=1, ER = 1.0D - 14, NEV=100, X(1) = -0.1, X(2) = 1.0, X(3) = 0.1

(c) 主プログラム

```

PROGRAM BMSQPM
! *** EXAMPLE OF DMSQPM ***
IMPLICIT NONE
INTEGER MO,NO,NWK
PARAMETER(MO=2,NO=3)
PARAMETER(NWK=(4*NO+2*MO+21)*NO+108*MO)
INTEGER M,ML,N,NEV,IWK(MO+3),IERR,I
REAL(8) F,X(NO),ER,Y,WK(NWK)
EXTERNAL F,GF,HF
!
READ(5,*) M,ML,N
READ(5,*) ER
READ(5,*) NEV
DO 10 I=1,N
  READ(5,*) X(I)
10 CONTINUE
CALL DMSQPM(F,GF,HF,M,ML,X,N,ER,NEV,Y,IWK,WK,IERR)
WRITE(6,601)
WRITE(6,602) M,ML,N
WRITE(6,603) IERR
IF( IERR .EQ. 3000 .OR. IERR .EQ. 4000 ) GOTO 9999
WRITE(6,604) NEV
DO 20 I=1,N
  WRITE(6,605) I,X(I)
20 CONTINUE
WRITE(6,606) Y
9999 CONTINUE
STOP
!
601 FORMAT(' ',/,/,',', ' *** DMSQPM ***',/,2X,'** INPUT **')
602 FORMAT(6X,'M = ',I4,/,6X,'ML = ',I4,/,6X,'N = ',I4)
603 FORMAT(2X,'** OUTPUT **',/,6X,'IERR = ',I5)
604 FORMAT(6X,'NEV = ',I5,/)
605 FORMAT(8X,'X(',I2,') = ',1PD18.10)
606 FORMAT(8X,'Y = ',1PD18.10)
END

REAL(8) FUNCTION F(X)
REAL(8) X(3)
F = -1.0D0 * X(3)
RETURN
END

SUBROUTINE GF(X,N,G,MM)
INTEGER N,MM
REAL(8) X(N),G(MM)
G(1) = -1.0D0 * X(1) + 1.0D0 * X(2)
RETURN
END

SUBROUTINE HF(X,N,H,ML)
INTEGER N,ML
REAL(8) X(N),H(ML)
H(1) = X(1)**2 + X(2)**2 + X(3)**2 - 1.0D0
RETURN
END

```

(d) 出力結果

```

*** DMSQPM ***
** INPUT **
M = 2
ML = 1
N = 3
** OUTPUT **
IERR = 0
NEV = 15

X( 1) = -3.6562662107D-17
X( 2) = -3.7019577088D-16
X( 3) = 1.0000000000D+00
Y = -1.0000000000D+00

```

5.9 ネットワーク上の距離の最短化 (最短路問題)

5.9.1 DMSP1M, RMSP1M

ネットワーク上のある節点から他のすべての節点までの距離の最小化

(1) 機能

n 個の節点と m 本の枝をもち、全枝が非負の重みであるグラフ上で、ある節点 v_1 から他の節点 v_p への枝の重み $w(k_j)\{(v_j, v_{j+1})\}$ の和 $W(P)$ を最小にする経路 $P = (v_1, v_2, \dots, v_p)$ とそのときの $W(P)$ の値 (最短距離) を求める。

$$\text{目的関数} : W(P) = \sum_{j=1}^{p-1} w(k_j) \rightarrow \text{最小}$$

(2) 使用法

倍精度サブルーチン:

CALL DMSP1M (N, M, ITL, IHD, WGHT, INIT, D, IP, ISW, IWK, WK, IERR)

単精度サブルーチン:

CALL RMSP1M (N, M, ITL, IHD, WGHT, INIT, D, IP, ISW, IWK, WK, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\left\{ \begin{array}{l} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{array} \right\}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入 力	節点の数 n
2	M	I	1	入 力	枝の数 m
3	ITL	I	M	入 力	枝 k の始点の節点番号 $tail(k)$
4	IHD	I	M	入 力	枝 k の終点の節点番号 $head(k)$
5	WGHT	$\left\{ \begin{array}{l} D \\ R \end{array} \right\}$	M	入 力	枝 k の重み $weight(k)$
6	INIT	$\left\{ \begin{array}{l} D \\ R \end{array} \right\}$	1	入 力	出発点の節点番号 v_1
7	D	$\left\{ \begin{array}{l} D \\ R \end{array} \right\}$	N	出 力	出発点 v_1 から節点 v_p までの最短距離 $D(v_1, v_p)$ (注意事項 (a) 参照)
8	IP	I	N	出 力	出発点 v_1 から節点 v_p までの最短経路 (注意事項 (b) 参照)
9	ISW	I	1	入 力	処理スイッチ (注意事項 (c) 参照) ISW=0:有向グラフ ISW=1:無向グラフ
10	IWK	I	内容参照	ワーク	作業領域 大きさ: $(4 \times N + 2 \times M)$
11	WK	$\left\{ \begin{array}{l} D \\ R \end{array} \right\}$	$2 \times M$	ワーク	作業領域
12	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

- (a) $N \geq 2$
- (b) $M \geq 1$
- (c) $WGHT(k) > 0.0, (k = 1, \dots, M)$
- (d) $1 \leq ITL(k) \leq N, (k = 1, \dots, M)$
- (e) $1 \leq IHD(k) \leq N, (k = 1, \dots, M)$
- (f) $1 \leq INIT \leq N$
- (g) $ISW=0$ または $ISW=1$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) または (b) を満足しなかった.	処理を打ち切る.
3100	制限条件 (c) を満足しなかった.	
3200	制限条件 (d) または (e) を満足しなかった.	
3300	制限条件 (f) を満足しなかった.	
3400	制限条件 (g) を満足しなかった.	

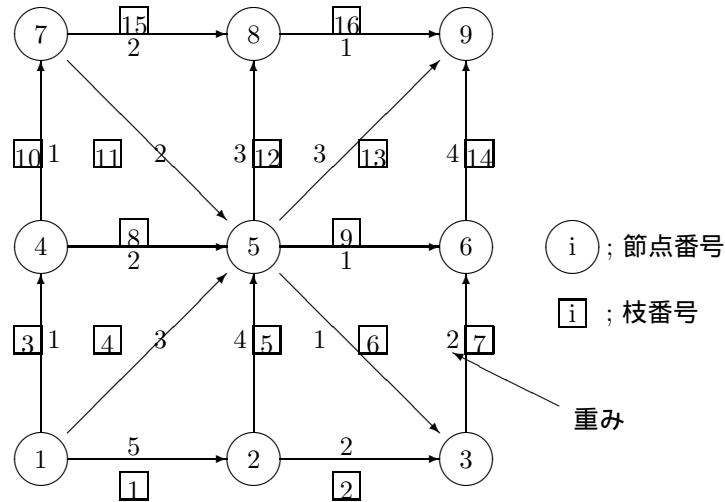
(6) 注意事項

- (a) 最短距離 $D(v_p)$ の値が負であるならば, 出発点 v_1 から節点 v_p への経路が存在しない.
- (b) 出発点 v_1 から節点 v_p への最短路上の節点の系列が $(v_1, \delta, \dots, \beta, \alpha, v_p)$ であるとき, これらの節点は終点から始点へと順次 $IP(v_p) = \alpha, IP(\alpha) = \beta, \dots, IP(\delta) = v_1$ のように求められる.
- (c) 無向グラフの各枝はサブルーチン内で自動的に2本の有向枝に置き換えられるので, 入力データを2重化しておく必要はない.

(7) 使用例

(a) 問題

以下のようなグラフ上で出発点 v_1 から各節点 v_p までの枝の重みの和を最小にする経路を求める。ただし、全枝の重みは非負とする。



(b) 入力データ

$N=9$, $M=16$, 枝の節点番号を格納する配列 ITL と IHD, 枝の重みを格納する配列 WGHT, 出発点 INIT =1, ISW =0.

(c) 主プログラム

```

PROGRAM BMSP1M
!
!   IMPLICIT REAL(8) (A-H,O-Z)
!
!   PARAMETER ( NMAX=10 )
!   PARAMETER ( MMAX=40 )
!   DIMENSION ITL(MMAX),IHD(MMAX),IP(NMAX)
!   DIMENSION WGHT(MMAX),D(NMAX)
!   DIMENSION IWK(4*NMAX+2*MMAX),WK(2*MMAX)
!
!   READ(5,*) N,M
!   DO 100 I=1,M
!     READ(5,*)ITL(I),IHD(I),WGHT(I)
100  CONTINUE
!
!   INIT=1
!   ISW=0
!   WRITE(6,6000)N,M,INIT,ISW
!   DO 110 I=1,M
!     WRITE(6,6010)ITL(I),IHD(I),WGHT(I)
110  CONTINUE
!   CALL DMSP1M(N,M,ITL,IHD,WGHT,INIT,D,IP,ISW,IWK,WK,IERR)
!
!   WRITE(6,6020)IERR
!   DO 120 I=1,N
!     WRITE(6,6030)I,D(I),I,IP(I)
120  CONTINUE
!   STOP
6000 FORMAT(/,5X,'*** DMSP1M ***',/,&
/,5X,' ** INPUT **',/,/,&
8X,'N   = ',I5,/,&
8X,'M   = ',I5,/,&
8X,'INIT = ',I5,/,&
8X,'ISW = ',I5,/,&
8X,'   ITL  IHD  WGHT')
6010 FORMAT(8X,I5,I5,F10.2)
6020 FORMAT(/,5X,'*** OUTPUT **',/,/,&
8X,'IERR = ',I5,/)
6030 FORMAT(8X,' D(',I2,',) = ',F10.2,8X,' IP(',I2,',) = ',I10)
END

```

(d) 出力結果

```

*** DMSP1M ***
** INPUT **
N   =    9
M   =   16

```

```
INIT =      1
ISW  =      0
ITL  IHD    WGHT
  1    2    5.00
  2    3    2.00
  1    4    1.00
  1    5    3.00
  2    5    4.00
  5    3    1.00
  3    6    2.00
  4    5    2.00
  5    6    1.00
  4    7    1.00
  7    5    2.00
  5    8    3.00
  5    9    3.00
  6    9    4.00
  7    8    2.00
  8    9    1.00

**  OUTPUT  **
IERR =      0
D( 1) =      0.00      IP( 1) =      1
D( 2) =      5.00      IP( 2) =      1
D( 3) =      4.00      IP( 3) =      5
D( 4) =      1.00      IP( 4) =      1
D( 5) =      3.00      IP( 5) =      1
D( 6) =      4.00      IP( 6) =      5
D( 7) =      2.00      IP( 7) =      4
D( 8) =      4.00      IP( 8) =      7
D( 9) =      5.00      IP( 9) =      8
```

5.9.2 DMSPMM, RMSPMM

ネットワーク上の全 2 節点間の距離の最小化

(1) 機能

n 個の節点と m 本の枝をもち、無向グラフの場合は負の重みの枝を含まないという条件を満たし、有向グラフの場合は負の長さのサイクルを含まないという条件を満たすグラフ上で、2 節点 $(v_1, v_p) (i = 1, 2, \dots, n)$ 間の枝の重みの和 $W(P)$ を最小にする経路 $P = (v_1, v_2, \dots, v_p)$ とそのとき $W(P)$ の値 (最短距離) を求める。

(2) 使用法

倍精度サブルーチン:

CALL DMSPMM (N, M, ITL, IHD, WGHT, D, IP, ISW, IWK, IERR)

単精度サブルーチン:

CALL RMSPMM (N, M, ITL, IHD, WGHT, D, IP, ISW, IWK, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
 R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入 力	節点の数 n
2	M	I	1	入 力	枝の数 m
3	ITL	I	M	入 力	枝 k の始点の節点番号 $tail(k)$
4	IHD	I	M	入 力	枝 k の終点の節点番号 $head(k)$
5	WGHT	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	M	入 力	枝 k の重み $weight(k)$
6	D	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	N, N	出 力	節点 v_i から節点 v_p までの最短距離 $D(v_i, v_p)$ (注意事項 (a) 参照)
7	IP	I	N, N	出 力	節点 v_i から節点 v_p までの最短経路 (注意事項 (b) 参照)
8	ISW	I	1	入 力	処理スイッチ (注意事項 (c) 参照) ISW=0:有向グラフ ISW=1:無向グラフ
9	IWK	I	N, N	ワ ーク	作業領域
10	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

(a) $N \geq 2$

(b) $M \geq 1$

(c) $WGHT(k) > 0.0, (k = 1, \dots, M)$ (ISW=1 のとき)

(d) $1 \leq ITL(k) \leq N, (k = 1, \dots, M)$

(e) $1 \leq IHD(k) \leq N, (k = 1, \dots, M)$

(f) ISW=0 または ISW=1

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) または (b) を満足しなかった.	処理を打ち切る.
3100	制限条件 (c) を満足しなかった.	
3200	制限条件 (d) または (e) を満足しなかった.	
3300	制限条件 (f) を満足しなかった.	
4000	グラフに負のサイクルが含まれていた.	

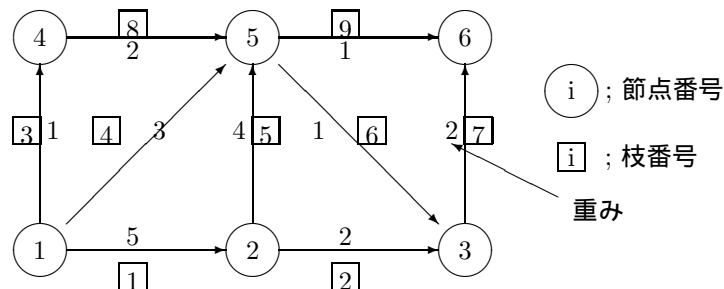
(6) 注意事項

- (a) 最短距離 $D(v_1, v_p)$ の値が負の値であるならば, 節点 v_1 から節点 v_p への経路は存在しない.
- (b) 節点 v_1 から節点 v_p への最短路上の節点の系列が $(v_1, \delta, \dots, \beta, \alpha, v_p)$ であるとき, これらの節点は終点から始点へと順次 $IP(v_1, v_p) = \alpha, IP(v_1, \alpha) = \beta, \dots, IP(v_1, \delta) = v_1$ のように求められる.
- (c) 無向グラフの各枝はサブルーチン内で自動的に 2 本の有向枝に置き換えられるので, 入力データを 2 重化しておく必要はない.

(7) 使用例

(a) 問題

以下のようなグラフ上で節点 v_1 から節点 v_p への枝の重みの和を最小にする経路を求める.



(b) 入力データ

$N=6, M=9$, 枝の節点番号を格納する配列 ITL と IHD, 枝の重みを格納する配列 WGHT, ISW = 0.

(c) 主プログラム

```

PROGRAM BMSPMM
!
! IMPLICIT REAL(8) (A-H,O-Z)
!
PARAMETER ( NMAX=6 )
PARAMETER ( MMAX=18 )
DIMENSION ITL(MMAX), IHD(MMAX), IP(NMAX, NMAX)
DIMENSION WGHT(MMAX), D(NMAX, NMAX)
DIMENSION IWK(NMAX*NMAX)
!
READ(5,*) N,M
DO 100 I=1,M
  READ(5,*) ITL(I), IHD(I), WGHT(I)
100 CONTINUE
!
ISW=0
WRITE(6,6000) N,M, ISW
DO 110 I=1,M
  WRITE(6,6010) ITL(I), IHD(I), WGHT(I)
110 CONTINUE
CALL DMSPMM(N,M, ITL, IHD, WGHT, D, IP, ISW, IWK, IERR)
!
WRITE(6,6020) IERR

```

```

      DO 120 I=1,N
      DO 130 J=1,N
        IF(D(I,J).NE.0) WRITE(6,6030)I,J,D(I,J),I,J,IP(I,J)
130    CONTINUE
120    CONTINUE
      STOP
6000  FORMAT(/,5X,'*** DMSPMM ***',/,&
          /,5X,' ** INPUT **',/,&
          8X,'N = ',I5,/,&
          8X,'M = ',I5,/,&
          8X,'ISW = ',I5,/,&
          8X,' ITL  IHD  WGHT')
6010  FORMAT(8X,I5,I5,F10.2)
6020  FORMAT(/,5X,' ** OUTPUT **',/,&
          8X,' IERR = ',I5,/)
6030  FORMAT(8X,' D(',I2,',',I2,',') = ',F10.2,&
          ', IP(',I2,',',I2,',') = ',I5)
      END

```

(d) 出力結果

```

*** DMSPMM ***
** INPUT **
N      =      6
M      =      9
ISW    =      0
  ITL  IHD      WGHT
    1   2      5.00
    2   3      2.00
    1   4      1.00
    1   5      3.00
    2   5      4.00
    5   3      1.00
    3   6      2.00
    4   5      2.00
    5   6      1.00
** OUTPUT **
IERR =      0
D( 1, 2) =      5.00   IP( 1, 2) =      1
D( 1, 3) =      4.00   IP( 1, 3) =      5
D( 1, 4) =      1.00   IP( 1, 4) =      1
D( 1, 5) =      3.00   IP( 1, 5) =      1
D( 1, 6) =      4.00   IP( 1, 6) =      5
D( 2, 3) =      2.00   IP( 2, 3) =      2
D( 2, 5) =      4.00   IP( 2, 5) =      2
D( 2, 6) =      4.00   IP( 2, 6) =      3
D( 3, 6) =      2.00   IP( 3, 6) =      3
D( 4, 3) =      3.00   IP( 4, 3) =      5
D( 4, 5) =      2.00   IP( 4, 5) =      4
D( 4, 6) =      3.00   IP( 4, 6) =      5
D( 5, 3) =      1.00   IP( 5, 3) =      5
D( 5, 6) =      1.00   IP( 5, 6) =      5

```


5.9.3 DMSP11, RMSP11 ネットワーク上の2節点間の距離の最小化

(1) 機能

n 個の節点と m 本の枝をもち、全枝が非負の重みであるグラフ上で、2 節点間 (v_1, v_p) の枝の重み $w(k_j)\{(v_j, v_{j+1})\}$ の和 $W(P)$ を最小にする経路 $P = (v_1, v_2, \dots, v_p)$ とそのときの $W(P)$ の値 (最短距離) を求める。

$$\text{目的関数} : W(P) = \sum_{j=1}^{p-1} w(k_j) \rightarrow \text{最小}$$

(2) 使用法

倍精度サブルーチン:

CALL DMSP11 (N, M, ITL, IHD, WGHT, INIT, IEND, D, IP, ISW, IWK, WK, IERR)

単精度サブルーチン:

CALL RMSP11 (N, M, ITL, IHD, WGHT, INIT, IEND, D, IP, ISW, IWK, WK, IERR)

(3) 引数

D:倍精度実数型 Z:倍精度複素数型 I: $\begin{cases} 32 \text{ ビット整数版では INTEGER}(4) \\ 64 \text{ ビット整数版では INTEGER}(8) \end{cases}$
R:単精度実数型 C:単精度複素数型

項番	引数名	型	大きさ	入出力	内 容
1	N	I	1	入 力	節点の数 n
2	M	I	1	入 力	枝の数 m
3	ITL	I	M	入 力	枝 k の始点の節点番号 $tail(k)$
4	IHD	I	M	入 力	枝 k の終点の節点番号 $head(k)$
5	WGHT	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	M	入 力	枝 k の重み $weight(k)$
6	INIT	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	出発点 v_1
7	IEND	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	入 力	終点 v_p
8	D	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	1	出 力	出発点 v_1 から終点 v_p までの最短距離 (注意事項 (a) 参照)
9	IP	I	N	出 力	出発点 v_1 から終点 v_p までの最短経路 (注意事項 (b) 参照)
10	ISW	I	1	入 力	処理スイッチ (注意事項 (c) 参照) ISW=0:有向グラフ ISW=1:無向グラフ
11	IWK	I	内容参照	ワーク	作業領域 大きさ: $(4 \times N + 2 \times M)$
12	WK	$\begin{Bmatrix} D \\ R \end{Bmatrix}$	$N + 2 \times M$	ワーク	作業領域
13	IERR	I	1	出 力	エラーインディケータ

(4) 制限条件

- (a) $N \geq 2$
- (b) $M \geq 1$
- (c) $WGHT(k) > 0.0, (k = 1, \dots, M)$
- (d) $1 \leq ITL(k) \leq N, (k = 1, \dots, M)$
- (e) $1 \leq IHD(k) \leq N, (k = 1, \dots, M)$
- (f) $1 \leq INIT \leq N$
- (g) $1 \leq IEND \leq N$
- (h) $ISW=0$ または $ISW=1$

(5) エラーインディケータ

IERR の値	意 味	処 理 内 容
0	正常終了.	
3000	制限条件 (a) または (b) を満足しなかった.	処理を打ち切る.
3100	制限条件 (c) を満足しなかった.	
3200	制限条件 (d) または (e) を満足しなかった.	
3300	制限条件 (f) または (g) を満足しなかった.	
3400	制限条件 (h) を満足しなかった.	

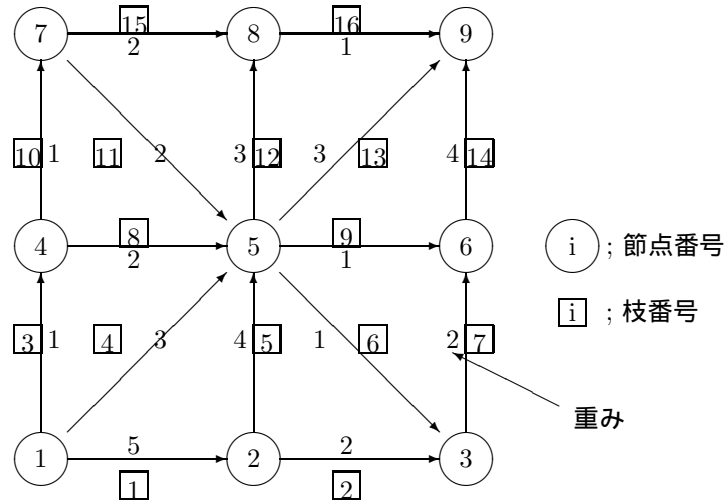
(6) 注意事項

- (a) 最短距離 D の値が負であるとき, 出発点 v_1 から終点 v_p への経路が存在しないことを表す.
- (b) 出発点 v_1 から節点 v_p への最短路上の節点の系列が $(v_1, \delta, \dots, \beta, \alpha, v_p)$ であるとき, これらの節点は終点から始点へと順次 $IP(v_p) = \alpha, IP(\alpha) = \beta, \dots, IP(\delta) = v_1$ のように求められる.
- (c) 無向グラフの各枝はサブルーチン内で自動的に2本の有向枝に置き換えられるので, 入力データを2重化しておく必要はない.

(7) 使用例

(a) 問題

以下のようなネットワーク上で出発点 v_1 から終点 v_p までの枝の重みの和を最小にする経路を求める。ただし、全枝の重みは非負とする。



(b) 入力データ

N= 9, M= 16, 枝の節点番号を格納する配列 ITL と IHD,
枝の重みを格納する配列 WGHT,
出発点 INIT =1, 終点 IEND=8 ,ISW= 0

(c) 主プログラム

```

PROGRAM BMSP11
!
! IMPLICIT REAL(8) (A-H,O-Z)
!
PARAMETER ( NMAX=10 )
PARAMETER ( MMAX=40 )
DIMENSION ITL(MMAX),IHD(MMAX),IP(NMAX)
DIMENSION WGHT(MMAX)
DIMENSION IWK(4*NMAX+2*MMAX),WK(2*MMAX+NMAX)
!
READ(5,*) N,M
DO 100 I=1,M
  READ(5,*) ITL(I),IHD(I),WGHT(I)
100 CONTINUE
!
INIT=1
IEND=8
ISW=0
WRITE(6,6000) N,M,INIT,IEND,ISW
DO 110 I=1,M
  WRITE(6,6010) ITL(I),IHD(I),WGHT(I)
110 CONTINUE
CALL DMSP11(N,M,ITL,IHD,WGHT,INIT,IEND,D,IP,ISW,IWK,WK,IERR)
!
WRITE(6,6020) IERR
WRITE(6,6030) D
DO 120 I=1,N
  WRITE(6,6040) I,IP(I)
120 CONTINUE
STOP
6000 FORMAT(/,5X,'*** DMSP11 ***',/,&
/,5X,' ** INPUT **',/,/,&
8X,'N = ',I5,/,&
8X,'M = ',I5,/,&
8X,'INIT = ',I5,/,&
8X,'IEND = ',I5,/,&
8X,'ISW = ',I5,/,&
8X,' ITL IHD WGHT')
6010 FORMAT(8X,I5,I5,F10.2)
6020 FORMAT(/,5X,' ** OUTPUT **',/,/,&
8X,'IERR = ',I5,/)
6030 FORMAT(8X,'D = ',F10.2,/)
6040 FORMAT(8X,'IP(',I2,',') = ',I5)
END
  
```

(d) 出力結果

```
*** DMSP11 ***
** INPUT **
N   =    9
M   =   16
INIT =    1
IEND =    8
ISW  =    0
ITL  IHD  WGHT
  1   2   5.00
  2   3   2.00
  1   4   1.00
  1   5   3.00
  2   5   4.00
  5   3   1.00
  3   6   2.00
  4   5   2.00
  5   6   1.00
  4   7   1.00
  7   5   2.00
  5   8   3.00
  5   9   3.00
  6   9   4.00
  7   8   2.00
  8   9   1.00
** OUTPUT **
IERR =    0
D    =   4.00
IP( 1) =    0
IP( 2) =    1
IP( 3) =    5
IP( 4) =    1
IP( 5) =    1
IP( 6) =    5
IP( 7) =    4
IP( 8) =    7
IP( 9) =    5
```

付録 A 用語説明

(1) グラフ

有限個の点の集合 $V = \{v_1, v_2, \dots, v_n\}$ と V に属する点の対の集合 $V \times V = \{(v_i, v_j) | v_i \in V, v_j \in V\}$ に対し、 $E \subseteq V \times V$ と V の組をグラフといい、 $G = (V, E)$ と表す。

(2) 節点と枝

与えられたグラフ $G = (V, E)$ に対して V の要素のことをグラフ G の節点、 E の要素のことをグラフ G の枝と呼ぶ。

(3) 有向枝

グラフ $G = (V, E)$ において E の要素 (v_i, v_j) ($v_i \in V, v_j \in V$) と (v_j, v_i) を区別して扱う場合、 E の各要素を有向枝と呼ぶ。なお有向枝 $e = (v_i, v_j)$ に対して $\text{tail}(e) = v_i$, $\text{head}(e) = v_j$ と表す。

(4) 閉路

グラフ $G = (V, E)$ に対して、 V の要素の組 $p = (v_{j_1}, v_{j_2}, \dots, v_{j_m})$ が任意の $1 \leq k \leq m-1$ について $(v_{j_k}, v_{j_{k+1}}) \in E$ を満たすとき、 p をグラフ G 上の路と呼ぶ。とくに $v_{j_1} = v_{j_m}$ であるような路を閉路と呼ぶ。

(5) 木 (tree)

閉路を含まないグラフを木 (tree) と呼ぶ。

(6) ネットワーク

グラフ $G = (V, E)$ の各枝 $e \in E$ に対して、費用係数 $c(e)$ と容量 $u(e)$ という 2 種類の実数と、始点 s と終点 t という V の 2 つの要素が与えられているとき、 G と c, u, s, t の組をネットワークと呼び、 $N = (G, c, u, s, t)$ のように表す。なお、 c, u, s, t の一部だけを扱う場合もやはり、ネットワークと呼ばれる。例えば始点 s と終点 t を特に定めない問題を扱う場合には、 $N = (G, c, u)$ のように表す。

(7) 流れ

ネットワークの各枝 $e \in E$ に対して、 $0 \leq x(e) \leq u(e)$ を満たすベクトル $x(e)$ をネットワーク N 上の流れ (flow) と呼ぶ。

(8) 最小費用流問題

与えられたネットワーク N の各節点 $v \in V$ に対して実数 $b(v)$ が与えられ、

$$\sum_{\text{tail}(e)=v} x(e) - \sum_{\text{head}(e)=v} x(e) = b(v)$$

$$\sum_{v \in V} b(v) = 0$$

が成立しているときに、全枝の費用の和

$$\sum_{e \in E} c(e)x(e)$$

を最小にする流れ $x(e)$ を求める問題を最小費用流問題と呼ぶ。

付録 B ASL で使用している計算機依存定数

B.1 誤差判定のための単位

ASL では、浮動小数点演算における誤差判定のための単位として次の値を設定している。誤差判定のための単位は、浮動小数点データの内部表現によって決まる数値であり、ASL ではこの単位を収束判定、零判定などに用いることがある。

表 B-1 誤差判定のための単位

単精度演算	倍精度演算
$2^{-23} (\simeq 1.19 \times 10^{-7})$	$2^{-52} (\simeq 2.22 \times 10^{-16})$

備考 誤差判定の単位 ϵ はマシン ϵ と呼ばれることもあり、通常、対応する浮動小数点形式で $1 + \epsilon$ の計算結果が 1 と異なるような最小の正の定数として定義される。したがって、誤差判定の単位を見れば、その浮動小数点形式での (仮数部の) 演算の最大有効桁数がわかる。

B.2 浮動小数点データの値の最大値・最小値

ASL の内部で定義している浮動小数点データの値の最大値、最小値を以下に示す。

なお、以下の最大値、最小値はハードウェアが実際に採用している浮動小数点形式のそれとは異なる場合があるので注意されたい。

表 B-2 浮動小数点データの値の最大値・最小値

	単精度演算	倍精度演算
最大値	$2^{127}(2 - 2^{-23}) (\simeq 3.40 \times 10^{38})$	$2^{1023}(2 - 2^{-52}) (\simeq 1.80 \times 10^{308})$
正の最小値	$2^{-126} (\simeq 1.17 \times 10^{-38})$	$2^{-1022} (\simeq 2.23 \times 10^{-308})$
負の最大値	$-2^{-126} (\simeq -1.17 \times 10^{-38})$	$-2^{-1022} (\simeq -2.23 \times 10^{-308})$
最小値	$-2^{127}(2 - 2^{-23}) (\simeq -3.40 \times 10^{38})$	$-2^{1023}(2 - 2^{-52}) (\simeq -1.80 \times 10^{308})$

索引

- CAM1HH : 第 1 分册, 83
 CAM1HM : 第 1 分册, 80
 CAM1MH : 第 1 分册, 77
 CAM1MM : 第 1 分册, 74
 CAN1HH : 第 1 分册, 95
 CAN1HM : 第 1 分册, 92
 CAN1MH : 第 1 分册, 89
 CAN1MM : 第 1 分册, 86
 CANVJ1 : 第 1 分册, 123
 CARGJM : 第 1 分册, 36
 CARSJD : 第 1 分册, 31
 CBGMDI : 第 2 分册, 71
 CBGMLC : 第 2 分册, 64
 CBGMLS : 第 2 分册, 66
 CBGMLU : 第 2 分册, 62
 CBGMLX : 第 2 分册, 73
 CBGMMS : 第 2 分册, 68
 CBGMSL : 第 2 分册, 58
 CBGMSM : 第 2 分册, 54
 CBGNDI : 第 2 分册, 90
 CBGNLC : 第 2 分册, 83
 CBGNLS : 第 2 分册, 85
 CBGNLU : 第 2 分册, 81
 CBGNLX : 第 2 分册, 92
 CBGNMS : 第 2 分册, 87
 CBGNSL : 第 2 分册, 78
 CBGNSM : 第 2 分册, 75
 CBHEDI : 第 2 分册, 208
 CBHELX : 第 2 分册, 203
 CBHELS : 第 2 分册, 210
 CBHEMS : 第 2 分册, 205
 CBHESL : 第 2 分册, 196
 CBHEUC : 第 2 分册, 201
 CBHEUD : 第 2 分册, 199
 CBHFDI : 第 2 分册, 192
 CBHFSL : 第 2 分册, 187
 CBHFLX : 第 2 分册, 194
 CBHFMS : 第 2 分册, 189
 CBHFSL : 第 2 分册, 179
 CBHFUC : 第 2 分册, 185
 CBHFUD : 第 2 分册, 183
 CBHPDI : 第 2 分册, 158
 CBHPLS : 第 2 分册, 153
 CBHPLX : 第 2 分册, 160
 CBHPMS : 第 2 分册, 155
 CBHPSL : 第 2 分册, 145
 CBHPUC : 第 2 分册, 151
 CBHPUD : 第 2 分册, 149
 CBHRDI : 第 2 分册, 175
 CBHRLS : 第 2 分册, 170
 CBHRLX : 第 2 分册, 177
 CBHRMS : 第 2 分册, 172
 CBHRSL : 第 2 分册, 162
 CBHRUC : 第 2 分册, 168
 CBHRUD : 第 2 分册, 166
 CCGEAA : 第 1 分册, 155
 CCGEAN : 第 1 分册, 158
 CCGHAA : 第 1 分册, 306
 CCGHAN : 第 1 分册, 310
 CCGJAA : 第 1 分册, 312
 CCGJAN : 第 1 分册, 316
 CCGKAA : 第 1 分册, 318
 CCGKAN : 第 1 分册, 322
 CCGNAA : 第 1 分册, 160
 CCGNAN : 第 1 分册, 163
 CCGRAA : 第 1 分册, 300
 CCGRAN : 第 1 分册, 304
 CCHEAA : 第 1 分册, 197
 CCHEAN : 第 1 分册, 200
 CCHEEE : 第 1 分册, 208
 CCHEEN : 第 1 分册, 212
 CCHESN : 第 1 分册, 206
 CCHESL : 第 1 分册, 202
 CCHJSS : 第 1 分册, 258
 CCHRAA : 第 1 分册, 179
 CCHRAN : 第 1 分册, 182
 CCHREE : 第 1 分册, 190
 CCHREN : 第 1 分册, 195

- CCHRSN : 第 1 分册, 188
CCHRSS : 第 1 分册, 184
CFC1BF : 第 3 分册, 53
CFC1FB : 第 3 分册, 50
CFC2BF : 第 3 分册, 103
CFC2FB : 第 3 分册, 100
CFC3BF : 第 3 分册, 128
CFC3FB : 第 3 分册, 125
CFCMBF : 第 3 分册, 79
CFCMFB : 第 3 分册, 76
CIBH1N : 第 5 分册, 131
CIBH2N : 第 5 分册, 133
CIBINZ : 第 5 分册, 118
CIBJNZ : 第 5 分册, 85
CIBKNZ : 第 5 分册, 120
CIBYNZ : 第 5 分册, 87
CIGAMZ : 第 5 分册, 168
CIGLGZ : 第 5 分册, 170
CLACHA : 第 5 分册, 327
CLNCIS : 第 5 分册, 342
- D1CDBN : 第 6 分册, 71
D1CDBT : 第 6 分册, 111
D1CDCC : 第 6 分册, 142
D1CDCH : 第 6 分册, 75
D1CDEX : 第 6 分册, 128
D1CDFB : 第 6 分册, 99
D1CDGM : 第 6 分册, 105
D1CDGU : 第 6 分册, 131
D1CDIB : 第 6 分册, 114
D1CDIC : 第 6 分册, 78
D1CDIF : 第 6 分册, 102
D1CDIG : 第 6 分册, 108
D1CDIN : 第 6 分册, 68
D1CDIS : 第 6 分册, 96
D1CDIT : 第 6 分册, 90
D1CDIX : 第 6 分册, 84
D1CDLD : 第 6 分册, 133
D1CDLG : 第 6 分册, 139
D1CDLN : 第 6 分册, 136
D1CDNC : 第 6 分册, 81
D1CDNO : 第 6 分册, 65
D1CDNT : 第 6 分册, 93
D1CDPA : 第 6 分册, 122
D1CDTB : 第 6 分册, 87
- D1CDTR : 第 6 分册, 119
D1CDUF : 第 6 分册, 117
D1CDWE : 第 6 分册, 125
D1DDBP : 第 6 分册, 145
D1DDGO : 第 6 分册, 149
D1DDHG : 第 6 分册, 153
D1DDHN : 第 6 分册, 156
D1DDPO : 第 6 分册, 151
D2BA1T : 第 6 分册, 166
D2BA2S : 第 6 分册, 171
D2BAGM : 第 6 分册, 182
D2BAHM : 第 6 分册, 190
D2BAMO : 第 6 分册, 186
D2BAMS : 第 6 分册, 178
D2BASM : 第 6 分册, 193
D2CCMA : 第 6 分册, 213
D2CCMT : 第 6 分册, 208
D2CCPR : 第 6 分册, 218
D2VCGR : 第 6 分册, 201
D2VCMT : 第 6 分册, 196
D3IECD : 第 6 分册, 291
D3IEME : 第 6 分册, 278
D3IERA : 第 6 分册, 275
D3IESR : 第 6 分册, 295
D3IESU : 第 6 分册, 281
D3IETC : 第 6 分册, 288
D3IEVA : 第 6 分册, 285
D3TSCD : 第 6 分册, 329
D3TSME : 第 6 分册, 309
D3TSRA : 第 6 分册, 300
D3TSRD : 第 6 分册, 304
D3TSSR : 第 6 分册, 332
D3TSSU : 第 6 分册, 314
D3TSTC : 第 6 分册, 324
D3TSVA : 第 6 分册, 320
D41WR1 : 第 6 分册, 345
D42WR1 : 第 6 分册, 365
D42WRM : 第 6 分册, 357
D42WRN : 第 6 分册, 351
D4BI01 : 第 6 分册, 420
D4GL01 : 第 6 分册, 416
D4MU01 : 第 6 分册, 398
D4MWRF : 第 6 分册, 373
D4MWRM : 第 6 分册, 385
D4RBO1 : 第 6 分册, 412

- D5CHEF : 第 6 分册, 428
D5CHMD : 第 6 分册, 437
D5CHMN : 第 6 分册, 434
D5CHTT : 第 6 分册, 431
D5TEMH : 第 6 分册, 447
D5TESG : 第 6 分册, 440
D5TESP : 第 6 分册, 451
D5TEWL : 第 6 分册, 443
D6CLAN : 第 6 分册, 495
D6CLDA : 第 6 分册, 499
D6CLDS : 第 6 分册, 491
D6CPCC : 第 6 分册, 463
D6CPSC : 第 6 分册, 465
D6CVAN : 第 6 分册, 475
D6CVSC : 第 6 分册, 478
D6DAFN : 第 6 分册, 482
D6DASC : 第 6 分册, 485
D6FALD : 第 6 分册, 469
D6FAVR : 第 6 分册, 471
DABMCS : 第 1 分册, 13
DABMEL : 第 1 分册, 15
DAM1AD : 第 1 分册, 46
DAM1MM : 第 1 分册, 62
DAM1MS : 第 1 分册, 55
DAM1MT : 第 1 分册, 65
DAM1MU : 第 1 分册, 52
DAM1SB : 第 1 分册, 49
DAM1TM : 第 1 分册, 68
DAM1TP : 第 1 分册, 107
DAM1TT : 第 1 分册, 71
DAM1VM : 第 1 分册, 98
DAM3TP : 第 1 分册, 109
DAM3VM : 第 1 分册, 101
DAM4VM : 第 1 分册, 104
DAMT1M : 第 1 分册, 58
DAMVJ1 : 第 1 分册, 112
DAMVJ3 : 第 1 分册, 115
DAMVJ4 : 第 1 分册, 119
DARGJM : 第 1 分册, 26
DARSJD : 第 1 分册, 21
DASBCS : 第 1 分册, 17
DASBEL : 第 1 分册, 19
DATM1M : 第 1 分册, 60
DBBDDI : 第 2 分册, 221
DBBDLC : 第 2 分册, 217
DBBDLS : 第 2 分册, 219
DBBDLU : 第 2 分册, 215
DBBDLX : 第 2 分册, 223
DBBDSL : 第 2 分册, 212
DBBPDI : 第 2 分册, 234
DBBPLS : 第 2 分册, 232
DBBPLX : 第 2 分册, 236
DBBPSL : 第 2 分册, 226
DBBPUC : 第 2 分册, 230
DBBPUU : 第 2 分册, 229
DBGMDI : 第 2 分册, 49
DBGMLC : 第 2 分册, 42
DBGMLS : 第 2 分册, 44
DBGMLU : 第 2 分册, 40
DBGMLX : 第 2 分册, 51
DBGMMS : 第 2 分册, 46
DBGMSL : 第 2 分册, 36
DBGMSM : 第 2 分册, 32
DBPDDI : 第 2 分册, 102
DBPDLS : 第 2 分册, 100
DBPDLX : 第 2 分册, 104
DBPDSL : 第 2 分册, 94
DBPDUC : 第 2 分册, 98
DBPDUU : 第 2 分册, 97
DBSMDI : 第 2 分册, 134
DBSMLS : 第 2 分册, 129
DBSMLX : 第 2 分册, 136
DBSMMS : 第 2 分册, 131
DBSMSL : 第 2 分册, 122
DBSMUC : 第 2 分册, 127
DBSMUD : 第 2 分册, 125
DBSNLS : 第 2 分册, 143
DBSNSL : 第 2 分册, 138
DBSNUD : 第 2 分册, 141
DBSPDI : 第 2 分册, 118
DBSPLS : 第 2 分册, 113
DBSPLX : 第 2 分册, 120
DBSPMS : 第 2 分册, 115
DBSPSL : 第 2 分册, 106
DBSPUC : 第 2 分册, 111
DBSPUD : 第 2 分册, 109
DBTDSL : 第 2 分册, 238
DBTLCO : 第 2 分册, 275
DBTLDI : 第 2 分册, 277
DBTLSL : 第 2 分册, 273

- DBTOSL : 第 2 分册, 256
DBTPSL : 第 2 分册, 240
DBTSSL : 第 2 分册, 260
DBTUCO : 第 2 分册, 269
DBTUDI : 第 2 分册, 271
DBTUSL : 第 2 分册, 267
DBVMSL : 第 2 分册, 263
DCGBFF : 第 1 分册, 324
DCGEAA : 第 1 分册, 144
DCGEAN : 第 1 分册, 148
DCGGAA : 第 1 分册, 264
DCGGAN : 第 1 分册, 269
DCGJAA : 第 1 分册, 288
DCGJAN : 第 1 分册, 292
DCGKAA : 第 1 分册, 294
DCGKAN : 第 1 分册, 298
DCGNAA : 第 1 分册, 150
DCGNAN : 第 1 分册, 153
DCGSAA : 第 1 分册, 271
DCGSAN : 第 1 分册, 274
DCGSEE : 第 1 分册, 282
DCGSEN : 第 1 分册, 286
DCGSSN : 第 1 分册, 280
DCGSSS : 第 1 分册, 276
DCSBAA : 第 1 分册, 214
DCSBAN : 第 1 分册, 217
DCSBFF : 第 1 分册, 225
DCSBSN : 第 1 分册, 223
DCSBSS : 第 1 分册, 219
DCSJSS : 第 1 分册, 251
DCSMAA : 第 1 分册, 164
DCSMAN : 第 1 分册, 167
DCSMEE : 第 1 分册, 173
DCSMEN : 第 1 分册, 177
DCSMSN : 第 1 分册, 171
DCSMSS : 第 1 分册, 168
DCSRSS : 第 1 分册, 245
DCSTAA : 第 1 分册, 229
DCSTAN : 第 1 分册, 232
DCSTEE : 第 1 分册, 239
DCSTEN : 第 1 分册, 243
DCSTSN : 第 1 分册, 237
DCSTSS : 第 1 分册, 233
DFASMA : 第 6 分册, 242
DFC1BF : 第 3 分册, 46
DFC1FB : 第 3 分册, 43
DFC2BF : 第 3 分册, 96
DFC2FB : 第 3 分册, 93
DFC3BF : 第 3 分册, 120
DFC3FB : 第 3 分册, 116
DFCMBF : 第 3 分册, 70
DFCMFB : 第 3 分册, 66
DFCN1D : 第 3 分册, 143
DFCN2D : 第 3 分册, 152
DFCN3D : 第 3 分册, 159
DFCR1D : 第 3 分册, 169
DFCR2D : 第 3 分册, 177
DFCR3D : 第 3 分册, 184
DFCRCS : 第 6 分册, 240
DFCRCZ : 第 6 分册, 238
DFCRSC : 第 6 分册, 236
DFCVCS : 第 6 分册, 232
DFCVSC : 第 6 分册, 229
DFDPED : 第 6 分册, 248
DFDPES : 第 6 分册, 246
DFDPET : 第 6 分册, 251
DFLAGE : 第 3 分册, 225
DFLARA : 第 3 分册, 220
DFPS1D : 第 3 分册, 194
DFPS2D : 第 3 分册, 201
DFPS3D : 第 3 分册, 208
DFR1BF : 第 3 分册, 61
DFR1FB : 第 3 分册, 57
DFR2BF : 第 3 分册, 111
DFR2FB : 第 3 分册, 107
DFR3BF : 第 3 分册, 137
DFR3FB : 第 3 分册, 133
DFRMBF : 第 3 分册, 88
DFRMFB : 第 3 分册, 84
DFWTFF : 第 3 分册, 250
DFWTFT : 第 3 分册, 252
DFWTH1 : 第 3 分册, 228
DFWTH2 : 第 3 分册, 236
DFWTHI : 第 3 分册, 242
DFWTHR : 第 3 分册, 230
DFWTHS : 第 3 分册, 233
DFWHTH : 第 3 分册, 239
DFWTMF : 第 3 分册, 246
DFWTMT : 第 3 分册, 248
DGICBP : 第 4 分册, 410

- DGICBS : 第 4 分册, 430
DGICCM : 第 4 分册, 388
DGICCN : 第 4 分册, 391
DGICCO : 第 4 分册, 384
DGICCP : 第 4 分册, 377
DGICCCQ : 第 4 分册, 378
DGICCR : 第 4 分册, 380
DGICCS : 第 4 分册, 382
DGICCT : 第 4 分册, 386
DGIDBY : 第 4 分册, 414
DGIDCY : 第 4 分册, 396
DGIDMC : 第 4 分册, 360
DGIDPC : 第 4 分册, 352
DGIDSC : 第 4 分册, 355
DGIDYB : 第 4 分册, 403
DGIIBZ : 第 4 分册, 416
DGIICZ : 第 4 分册, 398
DGIIMC : 第 4 分册, 372
DGIIPC : 第 4 分册, 365
DGIISC : 第 4 分册, 368
DGIIZB : 第 4 分册, 407
DGISBX : 第 4 分册, 412
DGISCX : 第 4 分册, 394
DGISI1 : 第 4 分册, 433
DGISI2 : 第 4 分册, 437
DGISI3 : 第 4 分册, 444
DGISMC : 第 4 分册, 347
DGISPC : 第 4 分册, 339
DGISPO : 第 4 分册, 418
DGISPR : 第 4 分册, 421
DGISS1 : 第 4 分册, 450
DGISS2 : 第 4 分册, 454
DGISS3 : 第 4 分册, 462
DGISSC : 第 4 分册, 342
DGISSO : 第 4 分册, 424
DGISSR : 第 4 分册, 427
DGISXB : 第 4 分册, 400
DH2INT : 第 4 分册, 245
DHBDFS : 第 4 分册, 217
DHBSFC : 第 4 分册, 220
DHEMNH : 第 4 分册, 223
DHEMNI : 第 4 分册, 236
DHEMNL : 第 4 分册, 187
DHNANL : 第 4 分册, 214
DHNEFL : 第 4 分册, 196
DHNENH : 第 4 分册, 229
DHNENL : 第 4 分册, 206
DHNFML : 第 4 分册, 257
DHNFMN : 第 4 分册, 251
DHNIFL : 第 4 分册, 200
DHNINH : 第 4 分册, 232
DHNINI : 第 4 分册, 242
DHNINL : 第 4 分册, 210
DHNOFH : 第 4 分册, 226
DHNOFI : 第 4 分册, 239
DHN OFL : 第 4 分册, 193
DHNPNL : 第 4 分册, 203
DHN RML : 第 4 分册, 254
DHN RNM : 第 4 分册, 248
DHNSNL : 第 4 分册, 190
DIB AID : 第 5 分册, 155
DIB AIX : 第 5 分册, 151
DIB BEI : 第 5 分册, 137
DIB BER : 第 5 分册, 135
DIB BID : 第 5 分册, 157
DIB BIX : 第 5 分册, 153
DIB IMX : 第 5 分册, 112
DIB INX : 第 5 分册, 108
DIB JMX : 第 5 分册, 79
DIB JNX : 第 5 分册, 75
DIB KEI : 第 5 分册, 141
DIB KER : 第 5 分册, 139
DIB KMX : 第 5 分册, 115
DIB KNX : 第 5 分册, 110
DIB SIN : 第 5 分册, 127
DIB SJN : 第 5 分册, 123
DIB SKN : 第 5 分册, 129
DIB SYN : 第 5 分册, 125
DIB YMX : 第 5 分册, 82
DIB YNX : 第 5 分册, 77
DIE II1 : 第 5 分册, 180
DIE II2 : 第 5 分册, 182
DIE II3 : 第 5 分册, 184
DIE II4 : 第 5 分册, 186
DIGIG1 : 第 5 分册, 164
DIGIG2 : 第 5 分册, 166
DIICOS : 第 5 分册, 212
DI IERF : 第 5 分册, 228
DI ISIN : 第 5 分册, 210
DILEG1 : 第 5 分册, 232

- DILEG2 : 第 5 分册, 235
DIMTCE : 第 5 分册, 252
DIMTSE : 第 5 分册, 255
DIOPC2 : 第 5 分册, 248
DIOPTH : 第 5 分册, 246
DIOPLG : 第 5 分册, 250
DIOPLH : 第 5 分册, 244
DIOPLA : 第 5 分册, 242
DIOPLB : 第 5 分册, 237
DIXEPS : 第 5 分册, 270
DIZBS0 : 第 5 分册, 90
DIZBS1 : 第 5 分册, 92
DIZBSL : 第 5 分册, 98
DIZBSN : 第 5 分册, 94
DIZBYN : 第 5 分册, 96
DIZGLW : 第 5 分册, 239
DJTECC : 第 6 分册, 32
DJTEEX : 第 6 分册, 29
DJTEGM : 第 6 分册, 41
DJTEGU : 第 6 分册, 35
DJTELG : 第 6 分册, 44
DJTENO : 第 6 分册, 26
DJTEUN : 第 6 分册, 21
DJTEWE : 第 6 分册, 38
DKFNCS : 第 4 分册, 66
DKHNCS : 第 4 分册, 70
DKINCT : 第 4 分册, 51
DKMNCN : 第 4 分册, 74
DKSNCA : 第 4 分册, 45
DKSNCS : 第 4 分册, 39
DKSSCA : 第 4 分册, 60
DLARHA : 第 5 分册, 324
DLNRDS : 第 5 分册, 330
DLNRIS : 第 5 分册, 333
DLNRSA : 第 5 分册, 339
DLNRSS : 第 5 分册, 336
DLSRDS : 第 5 分册, 345
DLSRIS : 第 5 分册, 350
DMCLAF : 第 5 分册, 407
DMCLCP : 第 5 分册, 427
DMCLMC : 第 5 分册, 422
DMCLMZ : 第 5 分册, 416
DMCLSN : 第 5 分册, 402
DMCLTP : 第 5 分册, 433
DMCQAZ : 第 5 分册, 449
DMCQLM : 第 5 分册, 444
DMCQSN : 第 5 分册, 439
DMCUSN : 第 5 分册, 399
DMSP11 : 第 5 分册, 467
DMSP1M : 第 5 分册, 460
DMSPMM : 第 5 分册, 464
DMSQPM : 第 5 分册, 455
DMUMQG : 第 5 分册, 392
DMUMQN : 第 5 分册, 389
DMUSSN : 第 5 分册, 396
DMUUSN : 第 5 分册, 386
DNCBPO : 第 4 分册, 316
DNDAAO : 第 4 分册, 296
DNDANL : 第 4 分册, 302
DNDAP0 : 第 4 分册, 299
DNGAPL : 第 4 分册, 312
DNLNMA : 第 6 分册, 525
DNLNRG : 第 6 分册, 513
DNLNRR : 第 6 分册, 518
DNNLGF : 第 6 分册, 535
DNNLPO : 第 6 分册, 530
DNRAPL : 第 4 分册, 307
DOFNNF : 第 4 分册, 98
DOFNNV : 第 4 分册, 92
DOHNLV : 第 4 分册, 117
DOHNNF : 第 4 分册, 111
DOHNNV : 第 4 分册, 105
DOIEF2 : 第 4 分册, 127
DOIEV1 : 第 4 分册, 130
DOLNLV : 第 4 分册, 123
DOPDH2 : 第 4 分册, 133
DOPDH3 : 第 4 分册, 139
DOSNNF : 第 4 分册, 85
DOSNNV : 第 4 分册, 79
DPDAPN : 第 4 分册, 284
DPDOPL : 第 4 分册, 281
DPGOPL : 第 4 分册, 293
DPLOPL : 第 4 分册, 288
DQFODX : 第 4 分册, 154
DQMOGX : 第 4 分册, 157
DQMOHX : 第 4 分册, 160
DQMOJX : 第 4 分册, 163
DSMGON : 第 5 分册, 290
DSMGPA : 第 5 分册, 294
DSSTA1 : 第 5 分册, 277

- DSSTA2 : 第 5 分冊, 280
DSSTPT : 第 5 分冊, 287
DSSTRA : 第 5 分冊, 284
DXA005 : 第 1 分冊, 39
- GAM1HH : 共有メモリ並列機能編, 41
GAM1HM : 共有メモリ並列機能編, 37
GAM1MH : 共有メモリ並列機能編, 33
GAM1MM : 共有メモリ並列機能編, 29
GAN1HH : 共有メモリ並列機能編, 54
GAN1HM : 共有メモリ並列機能編, 51
GAN1MH : 共有メモリ並列機能編, 48
GAN1MM : 共有メモリ並列機能編, 45
GBHESL : 共有メモリ並列機能編, 126
GBHEUD : 共有メモリ並列機能編, 130
GBHFSL : 共有メモリ並列機能編, 120
GBHFUD : 共有メモリ並列機能編, 124
GBHPSL : 共有メモリ並列機能編, 108
GBHPUD : 共有メモリ並列機能編, 112
GBHRSL : 共有メモリ並列機能編, 114
GBHRUD : 共有メモリ並列機能編, 118
GCGJAA : 共有メモリ並列機能編, 244
GCGJAN : 共有メモリ並列機能編, 248
GCGKAA : 共有メモリ並列機能編, 250
GCGKAN : 共有メモリ並列機能編, 254
GCGRAA : 共有メモリ並列機能編, 238
GCGRAN : 共有メモリ並列機能編, 242
GCHEAA : 共有メモリ並列機能編, 202
GCHEAN : 共有メモリ並列機能編, 206
GCHESN : 共有メモリ並列機能編, 212
GCHESS : 共有メモリ並列機能編, 208
GCHRAA : 共有メモリ並列機能編, 189
GCHRAN : 共有メモリ並列機能編, 193
GCHRSN : 共有メモリ並列機能編, 200
GCHRSS : 共有メモリ並列機能編, 195
GFC2BF : 共有メモリ並列機能編, 301
GFC2FB : 共有メモリ並列機能編, 298
GFC3BF : 共有メモリ並列機能編, 325
GFC3FB : 共有メモリ並列機能編, 322
GFCMBF : 共有メモリ並列機能編, 276
GFCMFB : 共有メモリ並列機能編, 272
- HAM1HH : 共有メモリ並列機能編, 41
HAM1HM : 共有メモリ並列機能編, 37
HAM1MH : 共有メモリ並列機能編, 33
HAM1MM : 共有メモリ並列機能編, 29
- HAN1HH : 共有メモリ並列機能編, 54
HAN1HM : 共有メモリ並列機能編, 51
HAN1MH : 共有メモリ並列機能編, 48
HAN1MM : 共有メモリ並列機能編, 45
HBGMLC : 共有メモリ並列機能編, 86
HBGMLU : 共有メモリ並列機能編, 84
HBGMSL : 共有メモリ並列機能編, 80
HBGMSM : 共有メモリ並列機能編, 76
HBGNLC : 共有メモリ並列機能編, 96
HBGNLU : 共有メモリ並列機能編, 94
HBGNSL : 共有メモリ並列機能編, 91
HBGNSM : 共有メモリ並列機能編, 88
HBHESL : 共有メモリ並列機能編, 126
HBHEUD : 共有メモリ並列機能編, 130
HBHFSL : 共有メモリ並列機能編, 120
HBHFUD : 共有メモリ並列機能編, 124
HBHPSL : 共有メモリ並列機能編, 108
HBHPUD : 共有メモリ並列機能編, 112
HBHRSL : 共有メモリ並列機能編, 114
HBHRUD : 共有メモリ並列機能編, 118
HCGJAA : 共有メモリ並列機能編, 244
HCGJAN : 共有メモリ並列機能編, 248
HCGKAA : 共有メモリ並列機能編, 250
HCGKAN : 共有メモリ並列機能編, 254
HCGRAA : 共有メモリ並列機能編, 238
HCGRAN : 共有メモリ並列機能編, 242
HCHEAA : 共有メモリ並列機能編, 202
HCHEAN : 共有メモリ並列機能編, 206
HCHESN : 共有メモリ並列機能編, 212
HCHESS : 共有メモリ並列機能編, 208
HCHRAA : 共有メモリ並列機能編, 189
HCHRAN : 共有メモリ並列機能編, 193
HCHRSN : 共有メモリ並列機能編, 200
HCHRSS : 共有メモリ並列機能編, 195
HFC2BF : 共有メモリ並列機能編, 301
HFC2FB : 共有メモリ並列機能編, 298
HFC3BF : 共有メモリ並列機能編, 325
HFC3FB : 共有メモリ並列機能編, 322
HFCMBF : 共有メモリ並列機能編, 276
HFCMFB : 共有メモリ並列機能編, 272
- IIIERF : 第 5 分冊, 230
JIIERF : 第 5 分冊, 230
PAM1MM : 共有メモリ並列機能編, 17

- PAM1MT : 共有メモリ並列機能編, 20
 PAM1MU : 共有メモリ並列機能編, 14
 PAM1TM : 共有メモリ並列機能編, 23
 PAM1TT : 共有メモリ並列機能編, 26
 PBSNSL : 共有メモリ並列機能編, 103
 PBSNUD : 共有メモリ並列機能編, 106
 PBSPSL : 共有メモリ並列機能編, 98
 PBSPUD : 共有メモリ並列機能編, 101
 PCGJAA : 共有メモリ並列機能編, 226
 PCGJAN : 共有メモリ並列機能編, 230
 PCGKAA : 共有メモリ並列機能編, 232
 PCGKAN : 共有メモリ並列機能編, 236
 PCGSAA : 共有メモリ並列機能編, 214
 PCGSAN : 共有メモリ並列機能編, 217
 PCGSSN : 共有メモリ並列機能編, 224
 PCGSSS : 共有メモリ並列機能編, 219
 PCSMAA : 共有メモリ並列機能編, 179
 PCSMAN : 共有メモリ並列機能編, 182
 PCSMSN : 共有メモリ並列機能編, 187
 PCSMSS : 共有メモリ並列機能編, 184
 PFC2BF : 共有メモリ並列機能編, 294
 PFC2FB : 共有メモリ並列機能編, 291
 PFC3BF : 共有メモリ並列機能編, 317
 PFC3FB : 共有メモリ並列機能編, 314
 PFCMBF : 共有メモリ並列機能編, 266
 PFCMFB : 共有メモリ並列機能編, 262
 PFCN2D : 共有メモリ並列機能編, 339
 PFCN3D : 共有メモリ並列機能編, 346
 PFCR2D : 共有メモリ並列機能編, 354
 PFCR3D : 共有メモリ並列機能編, 361
 PFPS2D : 共有メモリ並列機能編, 370
 PFPS3D : 共有メモリ並列機能編, 377
 PFR2BF : 共有メモリ並列機能編, 309
 PFR2FB : 共有メモリ並列機能編, 305
 PFR3BF : 共有メモリ並列機能編, 334
 PFR3FB : 共有メモリ並列機能編, 330
 PFRMBF : 共有メモリ並列機能編, 285
 PFRMFB : 共有メモリ並列機能編, 281
 PSSTA1 : 共有メモリ並列機能編, 393
 PSSTA2 : 共有メモリ並列機能編, 396
 PXE010 : 共有メモリ並列機能編, 143
 PXE020 : 共有メモリ並列機能編, 150
 PXE030 : 共有メモリ並列機能編, 157
 PXE040 : 共有メモリ並列機能編, 164
 QAM1MM : 共有メモリ並列機能編, 17
 QAM1MT : 共有メモリ並列機能編, 20
 QAM1MU : 共有メモリ並列機能編, 14
 QAM1TM : 共有メモリ並列機能編, 23
 QAM1TT : 共有メモリ並列機能編, 26
 QBGMLC : 共有メモリ並列機能編, 74
 QBGMLU : 共有メモリ並列機能編, 72
 QBGMSL : 共有メモリ並列機能編, 68
 QBGMSM : 共有メモリ並列機能編, 65
 QBSNSL : 共有メモリ並列機能編, 103
 QBSNUD : 共有メモリ並列機能編, 106
 QBSPSL : 共有メモリ並列機能編, 98
 QBSPUD : 共有メモリ並列機能編, 101
 QCGJAA : 共有メモリ並列機能編, 226
 QCGJAN : 共有メモリ並列機能編, 230
 QCGKAA : 共有メモリ並列機能編, 232
 QCGKAN : 共有メモリ並列機能編, 236
 QCGSAA : 共有メモリ並列機能編, 214
 QCGSAN : 共有メモリ並列機能編, 217
 QCGSSN : 共有メモリ並列機能編, 224
 QCGSSS : 共有メモリ並列機能編, 219
 QCSMAA : 共有メモリ並列機能編, 179
 QCSMAN : 共有メモリ並列機能編, 182
 QCSMSN : 共有メモリ並列機能編, 187
 QCSMSS : 共有メモリ並列機能編, 184
 QFC2BF : 共有メモリ並列機能編, 294
 QFC2FB : 共有メモリ並列機能編, 291
 QFC3BF : 共有メモリ並列機能編, 317
 QFC3FB : 共有メモリ並列機能編, 314
 QFCMBF : 共有メモリ並列機能編, 266
 QFCMFB : 共有メモリ並列機能編, 262
 QFCN2D : 共有メモリ並列機能編, 339
 QFCN3D : 共有メモリ並列機能編, 346
 QFCR2D : 共有メモリ並列機能編, 354
 QFCR3D : 共有メモリ並列機能編, 361
 QFPS2D : 共有メモリ並列機能編, 370
 QFPS3D : 共有メモリ並列機能編, 377
 QFR2BF : 共有メモリ並列機能編, 309
 QFR2FB : 共有メモリ並列機能編, 305
 QFR3BF : 共有メモリ並列機能編, 334
 QFR3FB : 共有メモリ並列機能編, 330
 QFRMBF : 共有メモリ並列機能編, 285
 QFRMFB : 共有メモリ並列機能編, 281
 QSSTA1 : 共有メモリ並列機能編, 393
 QSSTA2 : 共有メモリ並列機能編, 396
 QXE010 : 共有メモリ並列機能編, 143

- QXE020 : 共有メモリ並列機能編, 150
QXE030 : 共有メモリ並列機能編, 157
QXE040 : 共有メモリ並列機能編, 164
- R1CDBN : 第 6 分冊, 71
R1CDBT : 第 6 分冊, 111
R1CDCC : 第 6 分冊, 142
R1CDCH : 第 6 分冊, 75
R1CDEX : 第 6 分冊, 128
R1CDFB : 第 6 分冊, 99
R1CDGM : 第 6 分冊, 105
R1CDGU : 第 6 分冊, 131
R1CDIB : 第 6 分冊, 114
R1CDIC : 第 6 分冊, 78
R1CDIF : 第 6 分冊, 102
R1CDIG : 第 6 分冊, 108
R1CDIN : 第 6 分冊, 68
R1CDIS : 第 6 分冊, 96
R1CDIT : 第 6 分冊, 90
R1CDIX : 第 6 分冊, 84
R1CDLD : 第 6 分冊, 133
R1CDLG : 第 6 分冊, 139
R1CDLN : 第 6 分冊, 136
R1CDNC : 第 6 分冊, 81
R1CDNO : 第 6 分冊, 65
R1CDNT : 第 6 分冊, 93
R1CDPA : 第 6 分冊, 122
R1CDTB : 第 6 分冊, 87
R1CDTR : 第 6 分冊, 119
R1CDUF : 第 6 分冊, 117
R1CDWE : 第 6 分冊, 125
R1DDBP : 第 6 分冊, 145
R1DDGO : 第 6 分冊, 149
R1DDHG : 第 6 分冊, 153
R1DDHN : 第 6 分冊, 156
R1DDPO : 第 6 分冊, 151
R2BA1T : 第 6 分冊, 166
R2BA2S : 第 6 分冊, 171
R2BAGM : 第 6 分冊, 182
R2BAHM : 第 6 分冊, 190
R2BAMO : 第 6 分冊, 186
R2BAMS : 第 6 分冊, 178
R2BASM : 第 6 分冊, 193
R2CCMA : 第 6 分冊, 213
R2CCMT : 第 6 分冊, 208
- R2CCPR : 第 6 分冊, 218
R2VCGR : 第 6 分冊, 201
R2VCMT : 第 6 分冊, 196
R3IECD : 第 6 分冊, 291
R3IEME : 第 6 分冊, 278
R3IERA : 第 6 分冊, 275
R3IESR : 第 6 分冊, 295
R3IESU : 第 6 分冊, 281
R3IETC : 第 6 分冊, 288
R3IEVA : 第 6 分冊, 285
R3TSCD : 第 6 分冊, 329
R3TSME : 第 6 分冊, 309
R3TSRA : 第 6 分冊, 300
R3TSRD : 第 6 分冊, 304
R3TSSR : 第 6 分冊, 332
R3TSSU : 第 6 分冊, 314
R3TSTC : 第 6 分冊, 324
R3TSVA : 第 6 分冊, 320
R41WR1 : 第 6 分冊, 345
R42WR1 : 第 6 分冊, 365
R42WRM : 第 6 分冊, 357
R42WRN : 第 6 分冊, 351
R4BI01 : 第 6 分冊, 420
R4GL01 : 第 6 分冊, 416
R4MU01 : 第 6 分冊, 398
R4MWRF : 第 6 分冊, 373
R4MWRM : 第 6 分冊, 385
R4RB01 : 第 6 分冊, 412
R5CHEF : 第 6 分冊, 428
R5CHMD : 第 6 分冊, 437
R5CHMN : 第 6 分冊, 434
R5CHTT : 第 6 分冊, 431
R5TEMH : 第 6 分冊, 447
R5TESG : 第 6 分冊, 440
R5TESP : 第 6 分冊, 451
R5TEWL : 第 6 分冊, 443
R6CLAN : 第 6 分冊, 495
R6CLDA : 第 6 分冊, 499
R6CLDS : 第 6 分冊, 491
R6CPCC : 第 6 分冊, 463
R6CPSC : 第 6 分冊, 465
R6CVAN : 第 6 分冊, 475
R6CVSC : 第 6 分冊, 478
R6DAFN : 第 6 分冊, 482
R6DASC : 第 6 分冊, 485

- R6FALD : 第 6 分册, 469
R6FAVR : 第 6 分册, 471
RABMCS : 第 1 分册, 13
RABMEL : 第 1 分册, 15
RAM1AD : 第 1 分册, 46
RAM1MM : 第 1 分册, 62
RAM1MS : 第 1 分册, 55
RAM1MT : 第 1 分册, 65
RAM1MU : 第 1 分册, 52
RAM1SB : 第 1 分册, 49
RAM1TM : 第 1 分册, 68
RAM1TP : 第 1 分册, 107
RAM1TT : 第 1 分册, 71
RAM1VM : 第 1 分册, 98
RAM3TP : 第 1 分册, 109
RAM3VM : 第 1 分册, 101
RAM4VM : 第 1 分册, 104
RAMT1M : 第 1 分册, 58
RAMVJ1 : 第 1 分册, 112
RAMVJ3 : 第 1 分册, 115
RAMVJ4 : 第 1 分册, 119
RARGJM : 第 1 分册, 26
RARSJD : 第 1 分册, 21
RASBCS : 第 1 分册, 17
RASBEL : 第 1 分册, 19
RATM1M : 第 1 分册, 60
RBBDDI : 第 2 分册, 221
RBBDL C : 第 2 分册, 217
RBBDL S : 第 2 分册, 219
RBBDL U : 第 2 分册, 215
RBBDL X : 第 2 分册, 223
RBBDSL : 第 2 分册, 212
RBBPDI : 第 2 分册, 234
RBBPL S : 第 2 分册, 232
RBBPL X : 第 2 分册, 236
RBBPSL : 第 2 分册, 226
RBBPUC : 第 2 分册, 230
RBBPUU : 第 2 分册, 229
RBGMDI : 第 2 分册, 49
RBGMLC : 第 2 分册, 42
RBGMLS : 第 2 分册, 44
RBGMLU : 第 2 分册, 40
RBGMLX : 第 2 分册, 51
RBGMMS : 第 2 分册, 46
RBGMSL : 第 2 分册, 36
RBGMSM : 第 2 分册, 32
RBPDDI : 第 2 分册, 102
RBPDL S : 第 2 分册, 100
RBPDL X : 第 2 分册, 104
RBPDSL : 第 2 分册, 94
RBPDUC : 第 2 分册, 98
RBPDUU : 第 2 分册, 97
RBSMDI : 第 2 分册, 134
RBSMLS : 第 2 分册, 129
RBSMLX : 第 2 分册, 136
RBSMMS : 第 2 分册, 131
RBSMSL : 第 2 分册, 122
RBSMUC : 第 2 分册, 127
RBSMUD : 第 2 分册, 125
RBSNLS : 第 2 分册, 143
RBSNSL : 第 2 分册, 138
RBSNUD : 第 2 分册, 141
RBSPDI : 第 2 分册, 118
RBSPL S : 第 2 分册, 113
RBSPL X : 第 2 分册, 120
RBSPMS : 第 2 分册, 115
RBSPSL : 第 2 分册, 106
RBSPUC : 第 2 分册, 111
RBSPUU : 第 2 分册, 109
RBTDSL : 第 2 分册, 238
RBTLCO : 第 2 分册, 275
RBTLDI : 第 2 分册, 277
RBTLSL : 第 2 分册, 273
RBTOSL : 第 2 分册, 256
RBTPSL : 第 2 分册, 240
RBTSSL : 第 2 分册, 260
RBTUCO : 第 2 分册, 269
RBTUDI : 第 2 分册, 271
RBTUSL : 第 2 分册, 267
RBVMSL : 第 2 分册, 263
RCGBFF : 第 1 分册, 324
RCGEAA : 第 1 分册, 144
RCGEAN : 第 1 分册, 148
RCGGAA : 第 1 分册, 264
RCGGAN : 第 1 分册, 269
RCGJAA : 第 1 分册, 288
RCGJAN : 第 1 分册, 292
RCGKAA : 第 1 分册, 294
RCGKAN : 第 1 分册, 298
RCGNAA : 第 1 分册, 150

- RCGNAN : 第 1 分册, 153
RCGSAA : 第 1 分册, 271
RCGSAN : 第 1 分册, 274
RCGSEE : 第 1 分册, 282
RCGSEN : 第 1 分册, 286
RCGSSN : 第 1 分册, 280
RCGSSS : 第 1 分册, 276
RCSBAA : 第 1 分册, 214
RCSBAN : 第 1 分册, 217
RCSBFF : 第 1 分册, 225
RCSBSN : 第 1 分册, 223
RCSBSS : 第 1 分册, 219
RCSJSS : 第 1 分册, 251
RCSMAA : 第 1 分册, 164
RCSMAN : 第 1 分册, 167
RCSMEE : 第 1 分册, 173
RCSMEN : 第 1 分册, 177
RCSMSN : 第 1 分册, 171
RCSMSS : 第 1 分册, 168
RCSRSS : 第 1 分册, 245
RCSTAA : 第 1 分册, 229
RCSTAN : 第 1 分册, 232
RCSTEE : 第 1 分册, 239
RCSTEN : 第 1 分册, 243
RCSTSN : 第 1 分册, 237
RCSTSS : 第 1 分册, 233
RFASMA : 第 6 分册, 242
RFC1BF : 第 3 分册, 46
RFC1FB : 第 3 分册, 43
RFC2BF : 第 3 分册, 96
RFC2FB : 第 3 分册, 93
RFC3BF : 第 3 分册, 120
RFC3FB : 第 3 分册, 116
RFCMBF : 第 3 分册, 70
RFCMFB : 第 3 分册, 66
RFCN1D : 第 3 分册, 143
RFCN2D : 第 3 分册, 152
RFCN3D : 第 3 分册, 159
RFCR1D : 第 3 分册, 169
RFCR2D : 第 3 分册, 177
RFCR3D : 第 3 分册, 184
RFCRCS : 第 6 分册, 240
RFCRCZ : 第 6 分册, 238
RFCRSC : 第 6 分册, 236
RFCVCS : 第 6 分册, 232
RFCVSC : 第 6 分册, 229
RFDPED : 第 6 分册, 248
RFDPEB : 第 6 分册, 246
RFDPET : 第 6 分册, 251
RFLAGE : 第 3 分册, 225
RFLARA : 第 3 分册, 220
RFPS1D : 第 3 分册, 194
RFPS2D : 第 3 分册, 201
RFPS3D : 第 3 分册, 208
RFR1BF : 第 3 分册, 61
RFR1FB : 第 3 分册, 57
RFR2BF : 第 3 分册, 111
RFR2FB : 第 3 分册, 107
RFR3BF : 第 3 分册, 137
RFR3FB : 第 3 分册, 133
RFRMBF : 第 3 分册, 88
RFRMFB : 第 3 分册, 84
RFWTFF : 第 3 分册, 250
RFWTFT : 第 3 分册, 252
RFWTH1 : 第 3 分册, 228
RFWTH2 : 第 3 分册, 236
RFWTHI : 第 3 分册, 242
RFWTHR : 第 3 分册, 230
RFWTHS : 第 3 分册, 233
RFWTHT : 第 3 分册, 239
RFWTMF : 第 3 分册, 246
RFWTMT : 第 3 分册, 248
RGICBP : 第 4 分册, 410
RGICBS : 第 4 分册, 430
RGICCM : 第 4 分册, 388
RGICCN : 第 4 分册, 391
RGICCO : 第 4 分册, 384
RGICCP : 第 4 分册, 377
RGICCQ : 第 4 分册, 378
RGICCR : 第 4 分册, 380
RGICCS : 第 4 分册, 382
RGICCT : 第 4 分册, 386
RGIDBY : 第 4 分册, 414
RGIDCY : 第 4 分册, 396
RGIDMC : 第 4 分册, 360
RGIDPC : 第 4 分册, 352
RGIDSC : 第 4 分册, 355
RGIDYB : 第 4 分册, 403
RGIIBZ : 第 4 分册, 416
RGIICZ : 第 4 分册, 398

- RGIIMC : 第 4 分册, 372
RGIIPC : 第 4 分册, 365
RGIISC : 第 4 分册, 368
RGIIZB : 第 4 分册, 407
RGISBX : 第 4 分册, 412
RGISCX : 第 4 分册, 394
RGISI1 : 第 4 分册, 433
RGISI2 : 第 4 分册, 437
RGISI3 : 第 4 分册, 444
RGISMC : 第 4 分册, 347
RGISPC : 第 4 分册, 339
RGISPO : 第 4 分册, 418
RGISPR : 第 4 分册, 421
RGISS1 : 第 4 分册, 450
RGISS2 : 第 4 分册, 454
RGISS3 : 第 4 分册, 462
RGISSC : 第 4 分册, 342
RGISSO : 第 4 分册, 424
RGISSR : 第 4 分册, 427
RGISXB : 第 4 分册, 400
RH2INT : 第 4 分册, 245
RHBDFS : 第 4 分册, 217
RHBSFC : 第 4 分册, 220
RHEMNH : 第 4 分册, 223
RHEMNI : 第 4 分册, 236
RHEMNL : 第 4 分册, 187
RHNANL : 第 4 分册, 214
RHNEFL : 第 4 分册, 196
RHNENH : 第 4 分册, 229
RHNENL : 第 4 分册, 206
RHNFML : 第 4 分册, 257
RHNFMN : 第 4 分册, 251
RHNIFL : 第 4 分册, 200
RHNINH : 第 4 分册, 232
RHNINI : 第 4 分册, 242
RHNINL : 第 4 分册, 210
RHNOFH : 第 4 分册, 226
RHNOFI : 第 4 分册, 239
RHNOFL : 第 4 分册, 193
RHNPNL : 第 4 分册, 203
RHNRMN : 第 4 分册, 254
RHNRLM : 第 4 分册, 248
RHNSNL : 第 4 分册, 190
RIBAID : 第 5 分册, 155
RIBAIX : 第 5 分册, 151
RIBBEI : 第 5 分册, 137
RIBBER : 第 5 分册, 135
RIBBID : 第 5 分册, 157
RIBBIX : 第 5 分册, 153
RIBIMX : 第 5 分册, 112
RIBINX : 第 5 分册, 108
RIBJMX : 第 5 分册, 79
RIBJNX : 第 5 分册, 75
RIBKEI : 第 5 分册, 141
RIBKER : 第 5 分册, 139
RIBKMX : 第 5 分册, 115
RIBKNX : 第 5 分册, 110
RIBSIN : 第 5 分册, 127
RIBSIN : 第 5 分册, 123
RIBSKN : 第 5 分册, 129
RIBSYN : 第 5 分册, 125
RIBYMX : 第 5 分册, 82
RIBYNX : 第 5 分册, 77
RIEII1 : 第 5 分册, 180
RIEII2 : 第 5 分册, 182
RIEII3 : 第 5 分册, 184
RIEII4 : 第 5 分册, 186
RIGIG1 : 第 5 分册, 164
RIGIG2 : 第 5 分册, 166
RIICOS : 第 5 分册, 212
RIIERF : 第 5 分册, 228
RIISIN : 第 5 分册, 210
RILEG1 : 第 5 分册, 232
RILEG2 : 第 5 分册, 235
RIMTCE : 第 5 分册, 252
RIMTSE : 第 5 分册, 255
RIOPC2 : 第 5 分册, 248
RIOPCN : 第 5 分册, 246
RIOPLA : 第 5 分册, 250
RIOPLA : 第 5 分册, 244
RIOPLA : 第 5 分册, 242
RIOPLA : 第 5 分册, 237
RIXEPS : 第 5 分册, 270
RIZBS0 : 第 5 分册, 90
RIZBS1 : 第 5 分册, 92
RIZBSL : 第 5 分册, 98
RIZBSN : 第 5 分册, 94
RIZBYN : 第 5 分册, 96
RIZGLW : 第 5 分册, 239
RJTEBI : 第 6 分册, 47

- RJTECC : 第 6 分册, 32
RJTEEX : 第 6 分册, 29
RJTEGM : 第 6 分册, 41
RJTEGU : 第 6 分册, 35
RJTELG : 第 6 分册, 44
RJTENG : 第 6 分册, 50
RJTEN0 : 第 6 分册, 26
RJTEPO : 第 6 分册, 53
RJTEUN : 第 6 分册, 21
RJTEWE : 第 6 分册, 38
RKFNCS : 第 4 分册, 66
RKHNCs : 第 4 分册, 70
RKINCT : 第 4 分册, 51
RKMNCN : 第 4 分册, 74
RKSNCa : 第 4 分册, 45
RKSNCs : 第 4 分册, 39
RKSSCA : 第 4 分册, 60
RLARHA : 第 5 分册, 324
RLNRDS : 第 5 分册, 330
RLNRIS : 第 5 分册, 333
RLNRSA : 第 5 分册, 339
RLNRSS : 第 5 分册, 336
RLSRDS : 第 5 分册, 345
RLSRIS : 第 5 分册, 350
RMCLAF : 第 5 分册, 407
RMCLCP : 第 5 分册, 427
RMCLMC : 第 5 分册, 422
RMCLMZ : 第 5 分册, 416
RMCLSN : 第 5 分册, 402
RMCLTP : 第 5 分册, 433
RMCQAZ : 第 5 分册, 449
RMCQLM : 第 5 分册, 444
RMCQSN : 第 5 分册, 439
RMCUSN : 第 5 分册, 399
RMSP11 : 第 5 分册, 467
RMSP1M : 第 5 分册, 460
RMSPMM : 第 5 分册, 464
RMSQPM : 第 5 分册, 455
RMUMQG : 第 5 分册, 392
RMUMQN : 第 5 分册, 389
RMUSSN : 第 5 分册, 396
RMUUSN : 第 5 分册, 386
RNCBPO : 第 4 分册, 316
RNDAAO : 第 4 分册, 296
RNDANL : 第 4 分册, 302
RNDAP0 : 第 4 分册, 299
RNGAPL : 第 4 分册, 312
RNLNMA : 第 6 分册, 525
RNLNRG : 第 6 分册, 513
RNLNRR : 第 6 分册, 518
RNNLGF : 第 6 分册, 535
RNRAPL : 第 4 分册, 307
ROFNNF : 第 4 分册, 98
ROFNNV : 第 4 分册, 92
ROHNLV : 第 4 分册, 117
ROHNNF : 第 4 分册, 111
ROHNNV : 第 4 分册, 105
ROIEF2 : 第 4 分册, 127
ROIEV1 : 第 4 分册, 130
ROLNLV : 第 4 分册, 123
ROPDH2 : 第 4 分册, 133
ROPDH3 : 第 4 分册, 139
ROSNNF : 第 4 分册, 85
ROSNNV : 第 4 分册, 79
RPDAPN : 第 4 分册, 284
RPDOPL : 第 4 分册, 281
RPGOPL : 第 4 分册, 293
RPLOPL : 第 4 分册, 288
RQFODX : 第 4 分册, 154
RQMOGX : 第 4 分册, 157
RQMOHX : 第 4 分册, 160
RQMOJX : 第 4 分册, 163
RSMGON : 第 5 分册, 290
RSMGPA : 第 5 分册, 294
RSSTA1 : 第 5 分册, 277
RSSTA2 : 第 5 分册, 280
RSSTPT : 第 5 分册, 287
RSSTRA : 第 5 分册, 284
RXA005 : 第 1 分册, 39
VIBHOX : 第 5 分册, 143
VIBH1X : 第 5 分册, 145
VIBHY0 : 第 5 分册, 147
VIBHY1 : 第 5 分册, 149
VIBIOX : 第 5 分册, 100
VIBI1X : 第 5 分册, 104
VIBJOX : 第 5 分册, 67
VIBJ1X : 第 5 分册, 71
VIBK0X : 第 5 分册, 102
VIBK1X : 第 5 分册, 106

- VIBY0X : 第 5 分册, 69
 VIBY1X : 第 5 分册, 73
 VIDBEY : 第 5 分册, 261
 VIECI1 : 第 5 分册, 176
 VIECI2 : 第 5 分册, 178
 VIEJAC : 第 5 分册, 188
 VIEJEP : 第 5 分册, 198
 VIEJTE : 第 5 分册, 200
 VIEJZT : 第 5 分册, 196
 VIENMQ : 第 5 分册, 190
 VIEPAI : 第 5 分册, 202
 VIERFC : 第 5 分册, 226
 VIERRF : 第 5 分册, 224
 VIETHE : 第 5 分册, 193
 VIGAMX : 第 5 分册, 159
 VIGBET : 第 5 分册, 174
 VIGDIG : 第 5 分册, 172
 VIGLGX : 第 5 分册, 162
 VIICNC : 第 5 分册, 222
 VIICND : 第 5 分册, 220
 VIIDAW : 第 5 分册, 218
 VIIEXP : 第 5 分册, 205
 WIIFCO : 第 5 分册, 216
 WIIFSI : 第 5 分册, 214
 WIIOLOG : 第 5 分册, 208
 VINPLG : 第 5 分册, 263
 VIXSLA : 第 5 分册, 266
 VIXSPS : 第 5 分册, 258
 VIXZTA : 第 5 分册, 268

 WBTCLS : 第 2 分册, 252
 WBTCSL : 第 2 分册, 249
 WBTDLs : 第 2 分册, 246
 WBTDSL : 第 2 分册, 243
 WIBHOX : 第 5 分册, 143
 WIBH1X : 第 5 分册, 145
 WIBHYO : 第 5 分册, 147
 WIBHY1 : 第 5 分册, 149
 WIBIOX : 第 5 分册, 100
 WIBI1X : 第 5 分册, 104
 WIBJOX : 第 5 分册, 67
 WIBJ1X : 第 5 分册, 71
 WIBKOX : 第 5 分册, 102
 WIBK1X : 第 5 分册, 106
 WIBY0X : 第 5 分册, 69

 WIBY1X : 第 5 分册, 73
 WIDBEY : 第 5 分册, 261
 WIECI1 : 第 5 分册, 176
 WIECI2 : 第 5 分册, 178
 WIEJAC : 第 5 分册, 188
 WIEJEP : 第 5 分册, 198
 WIEJTE : 第 5 分册, 200
 WIEJZT : 第 5 分册, 196
 WIENMQ : 第 5 分册, 190
 WIEPAI : 第 5 分册, 202
 WIERFC : 第 5 分册, 226
 WIERRF : 第 5 分册, 224
 WIETHE : 第 5 分册, 193
 WIGAMX : 第 5 分册, 159
 WIGBET : 第 5 分册, 174
 WIGDIG : 第 5 分册, 172
 WIGLGX : 第 5 分册, 162
 WIICNC : 第 5 分册, 222
 WIICND : 第 5 分册, 220
 WIIDAW : 第 5 分册, 218
 WIIEXP : 第 5 分册, 205
 WIIFCO : 第 5 分册, 216
 WIIFSI : 第 5 分册, 214
 WIIOLOG : 第 5 分册, 208
 WINPLG : 第 5 分册, 263
 WIXSLA : 第 5 分册, 266
 WIXSPS : 第 5 分册, 258
 WIXZTA : 第 5 分册, 268

 ZAM1HH : 第 1 分册, 83
 ZAM1HM : 第 1 分册, 80
 ZAM1MH : 第 1 分册, 77
 ZAM1MM : 第 1 分册, 74
 ZAN1HH : 第 1 分册, 95
 ZAN1HM : 第 1 分册, 92
 ZAN1MH : 第 1 分册, 89
 ZAN1MM : 第 1 分册, 86
 ZANVJ1 : 第 1 分册, 123
 ZARGJM : 第 1 分册, 36
 ZARSJD : 第 1 分册, 31
 ZBGMDI : 第 2 分册, 71
 ZBGMLC : 第 2 分册, 64
 ZBGMLS : 第 2 分册, 66
 ZBGMLU : 第 2 分册, 62
 ZBGMLX : 第 2 分册, 73

- ZBGMS : 第 2 分册, 68
ZBGMSL : 第 2 分册, 58
ZBGMSM : 第 2 分册, 54
ZBGNDI : 第 2 分册, 90
ZBGNLC : 第 2 分册, 83
ZBGNLS : 第 2 分册, 85
ZBGNLU : 第 2 分册, 81
ZBGNLX : 第 2 分册, 92
ZBGNMS : 第 2 分册, 87
ZBGNSL : 第 2 分册, 78
ZBGNSM : 第 2 分册, 75
ZBHEDI : 第 2 分册, 208
ZBHELS : 第 2 分册, 203
ZBHELX : 第 2 分册, 210
ZBHEMS : 第 2 分册, 205
ZBHESL : 第 2 分册, 196
ZBHEUC : 第 2 分册, 201
ZBHEUD : 第 2 分册, 199
ZBHFDI : 第 2 分册, 192
ZBHFLS : 第 2 分册, 187
ZBHFLX : 第 2 分册, 194
ZBHFMS : 第 2 分册, 189
ZBHFSL : 第 2 分册, 179
ZBHFUC : 第 2 分册, 185
ZBHFUD : 第 2 分册, 183
ZBHPTI : 第 2 分册, 158
ZBHPLS : 第 2 分册, 153
ZBHPLX : 第 2 分册, 160
ZBHPMS : 第 2 分册, 155
ZBHPSL : 第 2 分册, 145
ZBHPUC : 第 2 分册, 151
ZBHPUD : 第 2 分册, 149
ZBHRDI : 第 2 分册, 175
ZBHRLS : 第 2 分册, 170
ZBHRLX : 第 2 分册, 177
ZBHRMS : 第 2 分册, 172
ZBHRSL : 第 2 分册, 162
ZBHRUC : 第 2 分册, 168
ZBHRUD : 第 2 分册, 166
ZCGEAA : 第 1 分册, 155
ZCGEAN : 第 1 分册, 158
ZCGHAA : 第 1 分册, 306
ZCGHAN : 第 1 分册, 310
ZCGJAA : 第 1 分册, 312
ZCGJAN : 第 1 分册, 316
ZCGKAA : 第 1 分册, 318
ZCGKAN : 第 1 分册, 322
ZCGNAA : 第 1 分册, 160
ZCGNAN : 第 1 分册, 163
ZCGRAA : 第 1 分册, 300
ZCGRAN : 第 1 分册, 304
ZCHEAA : 第 1 分册, 197
ZCHEAN : 第 1 分册, 200
ZCHEEE : 第 1 分册, 208
ZCHEEN : 第 1 分册, 212
ZCHESN : 第 1 分册, 206
ZCHESS : 第 1 分册, 202
ZCHJSS : 第 1 分册, 258
ZCHRAA : 第 1 分册, 179
ZCHRAN : 第 1 分册, 182
ZCHREE : 第 1 分册, 190
ZCHREN : 第 1 分册, 195
ZCHRSN : 第 1 分册, 188
ZCHRSS : 第 1 分册, 184
ZFC1BF : 第 3 分册, 53
ZFC1FB : 第 3 分册, 50
ZFC2BF : 第 3 分册, 103
ZFC2FB : 第 3 分册, 100
ZFC3BF : 第 3 分册, 128
ZFC3FB : 第 3 分册, 125
ZFCMBF : 第 3 分册, 79
ZFCMFB : 第 3 分册, 76
ZIBH1N : 第 5 分册, 131
ZIBH2N : 第 5 分册, 133
ZIBINZ : 第 5 分册, 118
ZIBJNZ : 第 5 分册, 85
ZIBKNZ : 第 5 分册, 120
ZIBYNZ : 第 5 分册, 87
ZIGAMZ : 第 5 分册, 168
ZIGLGZ : 第 5 分册, 170
ZLACHA : 第 5 分册, 327
ZLNCIS : 第 5 分册, 342

アプリケーションシステム
科学技術計算ライブラリ
ASL ユーザーズガイド

〈 基本機能編 第 5 分冊 〉

2023 年 3 月 ASL (1.1)

付属説明書 3.0.0-230301

日本電気株式会社

© NEC Corporation 2023

日本電気株式会社の許可なく複製・改変などを行うことはできません。

本書の内容に関しては将来予告なしに変更することがあります。