# 第14章　非圧縮性流れの解法—MAC型解法 (続き)

## 14.1　直角座標系から一般曲線座標系への変換

直角座標 (cartesian coordinates) を $\boldsymbol{x}$，一般曲線座標 (general curvilinear coordinates) を $\boldsymbol{\xi}$ で表す[1]．これらの座標間の変換の測度 (metric) $x_\xi, \ldots$ と $\xi_x, \ldots$ の間には次の関係が成立する[2]．

$$\begin{bmatrix} x_\xi & x_\eta & x_\zeta \\ y_\xi & y_\eta & y_\zeta \\ z_\xi & z_\eta & z_\zeta \end{bmatrix} \begin{bmatrix} \xi_x & \xi_y & \xi_z \\ \eta_x & \eta_y & \eta_z \\ \zeta_x & \zeta_y & \zeta_z \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{14.1}$$

また変換のヤコビアン $J$ は

$$J = \begin{vmatrix} x_\xi & x_\eta & x_\zeta \\ y_\xi & y_\eta & y_\zeta \\ z_\xi & z_\eta & z_\zeta \end{vmatrix} \tag{14.2}$$

となる．これらは２次元の場合には次のようになる．

$$x_\xi = J\eta_y, \quad x_\eta = -J\xi_y, \quad y_\xi = -J\eta_x, \quad y_\eta = J\xi_x, \qquad J = x_\xi y_\eta - x_\eta y_\xi \tag{14.3}$$

$\boldsymbol{x}$ に関する微分を $\boldsymbol{\xi}$ に関する微分に書き換えれば次のようになる[3]．

$$\frac{\partial}{\partial x_i} = \frac{\partial \xi_j}{\partial x_i}\frac{\partial}{\partial \xi_j}, \qquad \nabla = (\nabla \xi_j)\frac{\partial}{\partial \xi_j} \tag{14.4}$$

具体的には，例えば

$$\frac{\partial}{\partial x} = \frac{1}{J}\begin{vmatrix} y_\xi & y_\eta & y_\zeta \\ z_\xi & z_\eta & z_\zeta \\ \partial/\partial\xi & \partial/\partial\eta & \partial/\partial\zeta \end{vmatrix}, \quad \text{２次元では} \quad \frac{\partial}{\partial x} = \frac{1}{J}\left(y_\eta \frac{\partial}{\partial \xi} - y_\xi \frac{\partial}{\partial \eta}\right)$$

また一般に

$$\frac{\partial}{\partial \xi_j} J \frac{\partial \xi_j}{\partial x_i} = 0, \qquad \frac{\partial}{\partial \xi_j} J \nabla \xi_j = 0$$

[1] ここでの一般は直交に対し直交+非直交の意である．以前には非直交であることを強調し非直交曲線座標格子 (non-orthogonal curvilinear coordinate grid) などと言われた．

[2] 例えば $x = x(\xi, \eta, \zeta) = x(\xi(x, y, z), \eta(x, y, z), \zeta(x, y, z))$ であるから，この式を $y$ で微分すれば $0 = \xi_y x_\xi + \eta_y x_\eta + \zeta_y x_\zeta$ なる関係が得られ，式 (14.1) はこれらの関係を纏めたものである．

[3] ここでは Einstein 総和規約 (Einstein cnovention) が用いられる．すなわち同じ添字が繰り返されるときには $a_i b_i = a_1 b_1 + a_2 b_2 + a_3 b_3$ のように解釈する．

であるから次の関係が導かれる．

$$J\frac{\partial\phi}{\partial x_i} = J\frac{\partial\xi_j}{\partial x_i}\frac{\partial\phi}{\partial\xi_j} = \frac{\partial}{\partial\xi_j}\Big(J\frac{\partial\xi_j}{\partial x_i}\phi\Big) \tag{14.5a}$$

$$J\nabla\phi = \frac{\partial}{\partial\xi_j}J(\nabla\xi_j)\phi \tag{14.5b}$$

$$J\nabla^2 = \frac{\partial}{\partial\xi_j}\Big(Jg^{jk}\frac{\partial}{\partial\xi_k}\Big) \tag{14.5c}$$

ただし $(g^{ij})$ は測度テンソルで，その成分は次のように定義される．

$$g^{ij} = \nabla\xi_i\cdot\nabla\xi_j = \frac{\partial\xi_i}{\partial x_k}\frac{\partial\xi_j}{\partial x_k} \tag{14.6}$$

流線に沿う微分は次のように書き換えられる．

$$\boldsymbol{u}\cdot\nabla = u_i\frac{\partial}{\partial x_i} = u_i\frac{\partial\xi_j}{\partial x_i}\frac{\partial}{\partial\xi_j} = U_j\frac{\partial}{\partial\xi_j} = \boldsymbol{U}\cdot\tilde{\nabla} \tag{14.7}$$

ただし $U_j$ は反変速度 (contravariant velocity) の $\xi_j$ 方向成分である．反変速度 $\boldsymbol{U}$ は $\boldsymbol{\xi}$ 空間の流速で[4]，物理空間の流速 $\boldsymbol{u}$ との間には次の関係がある．

$$U_j = \frac{\partial\xi_j}{\partial x_i}u_i, \qquad u_i = \frac{\partial x_i}{\partial\xi_j}U_j \tag{14.8}$$

なおこのような関係を知らずに，基礎方程式の $x_i$ の微分を，式 (14.4) によって $\xi_j$ の微分に書き換えるとたいへん長い式になり収拾がつかなくなる．

流れの渦度 $\boldsymbol{\zeta} = \nabla\times\boldsymbol{u}$ の $x_i$ 方向成分は

$$\zeta_i = \epsilon_{ijk}\frac{\partial u_k}{\partial x_j} \tag{14.9}$$

ただし $(\epsilon_{ijk})$ は Eddington の $\boldsymbol{\epsilon}$ で

$$\epsilon_{ijk} = \begin{cases} 1 & ((ijk) = (123),\ (231),\ \text{または } (312)) \\ -1 & ((ijk) = (321),\ (213),\ \text{または } (132)) \\ 0 & (\text{その他の場合}) \end{cases}$$

例えば，$\zeta_1 = \partial u_3/\partial x_2 - \partial u_2/\partial x_3$ である．任意のベクトル $\boldsymbol{a}$ に対し次の関係が成立する．

$$\nabla\xi_l\cdot\nabla\times\boldsymbol{a} = \nabla\xi_l\cdot\nabla\xi_m\times\frac{\partial\boldsymbol{a}}{\partial\xi_m} = \frac{1}{J}\epsilon_{lmn}\frac{\partial x_i}{\partial\xi_n}\frac{\partial a_i}{\partial\xi_m} = \frac{1}{J}\epsilon_{lmn}\frac{\partial}{\partial\xi_m}\Big(\frac{\partial\boldsymbol{x}}{\partial\xi_n}\cdot\boldsymbol{a}\Big) \tag{14.10}$$

なおこの式の導出には次式が用いられた．

$$x_\xi = J\begin{vmatrix}\eta_y & \eta_z \\ \zeta_z & \zeta_z\end{vmatrix}, \ \cdots, \qquad \epsilon_{lmn}\frac{\partial^2\boldsymbol{x}}{\partial\xi_m\partial\xi_n} = 0 \tag{14.11}$$

式 (14.10) で，特に $\boldsymbol{a} = \boldsymbol{u}$ ならば．次の反変渦度と反変速度の関係が得られる．

$$Z_l = \frac{1}{J}\epsilon_{lmn}\frac{\partial}{\partial\xi_m}(g_{nj}U_j) \tag{14.12}$$

[4] 反変速度が写像空間の流速であることは次のように説明できる．ある時間に点 $\boldsymbol{x}$ にある流体粒子は単位時間後に点 $\boldsymbol{x}+\boldsymbol{u}$ に移動する．点 $\boldsymbol{x}$ の曲線座標を $\boldsymbol{\xi}$ とすれば点 $\boldsymbol{x}+\boldsymbol{u}$ の曲線座標は $\boldsymbol{\xi}+\boldsymbol{u}\cdot\nabla\boldsymbol{\xi}$ となる．したがって反変速度 $U \equiv \boldsymbol{u}\cdot\nabla\boldsymbol{\xi}$ は写像空間で流体粒子が単位時間に移動する距離すなわち流速であることが分かる．

ただし $(g_{ij})$ は測度テンソルで，$(g_{ij})^{-1} = (g^{ij})$，その成分は次式で定義される．

$$g_{ij} = \frac{\partial \boldsymbol{x}}{\partial \xi_i} \cdot \frac{\partial \boldsymbol{x}}{\partial \xi_j} = \frac{\partial x_k}{\partial \xi_i} \frac{\partial x_k}{\partial \xi_j} \tag{14.13}$$

また $Z_l$ は反変渦度の $\xi_l$ 方向成分で，反変渦度 $\boldsymbol{Z}$ と物理空間の渦度 $\boldsymbol{\zeta}$ の間には次の関係がある．

$$Z_j = \frac{\partial \xi_j}{\partial x_i} \zeta_i, \qquad \zeta_i = \frac{\partial x_i}{\partial \xi_j} Z_j \tag{14.14}$$

## 14.2 曲線座標格子の SMAC Δ形陰解法

物理空間の曲線座標格子を写像空間の立方体格子に写像する．立方体格子の稜の長さ $\Delta\xi, \Delta\eta, \Delta\zeta$ は通常 1 に取られるが，以下の説明では当面 省略せずに残しておくことにする．直角座標格子の MAC 型解法を曲線座標格子の解法に拡張する際には，その特徴を損なわないようにしなければならない．そのために，各セル面の中心に妥当な流速成分として写像空間の流速である反変速度 $\boldsymbol{U}$ の成分またはこれに相当のものが定義される．既存の解法では，反変速度成分，体積流束，反変物理速度成分，共変物理速度成分が用いられているが，ここでは体積流束 (volume flux) $\tilde{\boldsymbol{U}} \equiv J\boldsymbol{U}$ を用いることにする．

連続方程式は式 (14.5) を用いれば次のようになる．

$$J\frac{\partial u_i}{\partial x_i} = \frac{\partial}{\partial \xi_j}\left(J\frac{\partial \xi_j}{\partial x_i} u_i\right) = \frac{\partial}{\partial \xi_j}\tilde{U}_j = 0 \tag{14.15}$$

図 14.1 は $\boldsymbol{\xi}$ 空間内の 1 つの立方体格子セルを示したものである．式 (14.15) は，この立方体格子セルにわたって積分すれば

$$(\tilde{U}_{100} - \tilde{U}_{000})\Delta\eta\Delta\zeta + (\tilde{V}_{010} - \tilde{V}_{000})\Delta\zeta\Delta\xi + (\tilde{W}_{001} - \tilde{W}_{000})\Delta\xi\Delta\eta = 0 \tag{14.16}$$

となる．式 (14.16) の中で，

$$\tilde{U}_{000}\Delta\eta\Delta\zeta = \left(J\frac{\partial \xi}{\partial x_i} u_i\right)_{000U} \Delta\eta\Delta\zeta = \Delta\eta\Delta\zeta \begin{vmatrix} u & x_\eta & x_\zeta \\ v & y_\eta & y_\zeta \\ w & z_\eta & z_\zeta \end{vmatrix}_{000U}$$

$$\approx \boldsymbol{u}_{000U} \cdot (\boldsymbol{x}_{010} - \boldsymbol{x}_{000}) \times (\boldsymbol{x}_{001} - \boldsymbol{x}_{000})$$

ここに $(\boldsymbol{x}_{010} - \boldsymbol{x}_{000}) \times (\boldsymbol{x}_{001} - \boldsymbol{x}_{000})$ は $\boldsymbol{x}$ 空間内の対応する六面体格子セルの，$\xi = \xi_0$ 面の面積ベクトルであるから，$\tilde{U}_{000}\Delta\eta\Delta\zeta$ はこの面を通る流量である．したがって式 (14.16) はこの六面体セルの 6 つの面から出入りする流量の和が 0 になるという意味の式である．写像空間内の流れに垂直に取られた単位面積当たりの物理空間の体積流量は体積流束 (volume flux) と呼ばれ，$\tilde{U}, \tilde{V}, \tilde{W}$ はこの体積流束の $\xi, \eta, \zeta$ 方向成分で，また $\Delta\xi = \Delta\eta = \Delta\zeta = 1$ ならば対応する六面体セルの面を通る流量になる．なお上式の導出には式 (14.1) から得られる次の関係が用いられた．

$$\xi_x = \frac{1}{J}\begin{vmatrix} y_\eta & y_\zeta \\ z_\eta & z_\zeta \end{vmatrix}, \ \cdots$$

次に体積流束の運動方程式について述べる．体積流束成分 $\tilde{U}_l$ は流速 $\boldsymbol{u}$ の左から $J\nabla\xi_l\cdot$ を演算したものであるから，$\tilde{U}_l$ の運動方程式は運動方程式 $d\boldsymbol{u}/dt \equiv \partial\boldsymbol{u}/\partial t+\nabla\cdot\boldsymbol{uu} = -\nabla p+\nu\nabla^2\boldsymbol{u}+\boldsymbol{g}$ の左から $J\nabla\xi_l\cdot$ を演算することによって導くことができる．その対流項は次のようになる[5]．

$$\begin{aligned} C_l &= J\nabla\xi_l\cdot(\nabla\cdot\boldsymbol{uu}) = \nabla\xi_l\cdot\frac{\partial}{\partial\xi_i}\tilde{U}_i\boldsymbol{u} = \frac{\partial}{\partial\xi_i}\tilde{U}_iU_l - \boldsymbol{u}\cdot\tilde{U}_i\frac{\partial}{\partial\xi_i}\nabla\xi_l \\ &= \frac{\partial}{\partial\xi_i}\tilde{U}_iU_l + \left\{{}_i^l{}_j\right\}\tilde{U}_iU_j \end{aligned} \tag{14.17}$$

ただし $\left\{{}_i^l{}_j\right\}$ は Christoffel の記号である．

$$\frac{\partial^2\boldsymbol{x}}{\partial\xi_i\partial\xi_j} = \left\{{}_i^l{}_j\right\}\frac{\partial\boldsymbol{x}}{\partial\xi_l}, \qquad \left\{{}_i^l{}_j\right\} = \frac{1}{2}g^{lk}\Big(\frac{\partial g_{jk}}{\partial\xi_i}+\frac{\partial g_{ik}}{\partial\xi_j}-\frac{\partial g_{ij}}{\partial\xi_k}\Big) \tag{14.18}$$

なお式 (14.17) の導出には次の関係が用いられた[6]．

$$\frac{\partial\boldsymbol{x}}{\partial\xi_j}\cdot\frac{\partial}{\partial\xi_i}\nabla\xi_l = -\nabla\xi_l\cdot\frac{\partial^2\boldsymbol{x}}{\partial\xi_i\partial\xi_j} = -\nabla\xi_l\cdot\left\{{}_i^k{}_j\right\}\frac{\partial\boldsymbol{x}}{\partial\xi_k} = -\left\{{}_i^l{}_j\right\}$$

粘性拡散項については **14.4** 節に詳しく述べる．結局，体積流束成分 $\tilde{\boldsymbol{U}}_l$ の運動方程式は次のようになる．

$$\frac{\partial}{\partial t}\tilde{U}_l = -\frac{\partial}{\partial\xi_i}\tilde{U}_iU_l - \left\{{}_i^l{}_j\right\}\tilde{U}_iU_j - \tilde{g}^{li}\frac{\partial p}{\partial\xi_i} - \nu\epsilon_{lij}\frac{\partial}{\partial\xi_i}(\tilde{g}_{jk}\tilde{Z}_k) + J\frac{\partial\xi_l}{\partial x_k}g_k \tag{14.19}$$

ただし $\tilde{U}_i = JU_i$, $\tilde{g}^{ij} = Jg^{ij}$, $\tilde{g}_{ij} = g_{ij}/J$, $\tilde{Z} = JZ$ である．この式の右辺第 2 項は，対流項を保存形で表したために出てきた付加項で，その値は滑らかな格子では通常小さく等間隔の直交格子では 0 になる．

**13.2** 節の SMAC 法の基礎方程式は一般曲線座標系の SMAC 法の式に書き換えれば次のようになる．

$$\tilde{U}_l^* = \tilde{U}_l^n - \Delta t\Big(C_l+\tilde{g}^{li}\frac{\partial p}{\partial\xi_i}-D_l\Big)^n \qquad (l=1,\,2,\,3) \tag{14.20a}$$

$$\tilde{U}_l^{n+1} = \tilde{U}_l^* - \Delta t\,\tilde{g}^{li}\frac{\partial\phi}{\partial\xi_i} \tag{14.20b}$$

$$\frac{\partial}{\partial\xi_l}\Big(\tilde{g}^{li}\frac{\partial\phi}{\partial\xi_i}\Big) = \frac{1}{\Delta t}\frac{\partial}{\partial\xi_l}\tilde{U}_l^*, \qquad p^{n+1} = p^n+\phi \tag{14.20c}$$

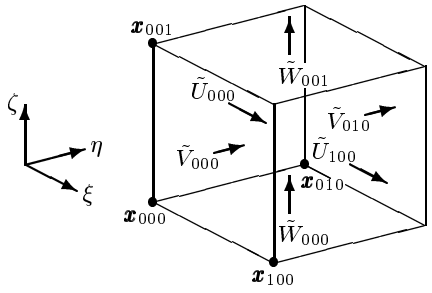


図 14.1: $\boldsymbol{\xi}$ 空間内の立方体格子セル

[5] 第 2 式から第 3 式へは式 (14.5) と (14.8)，第 3 式へは $\tilde{U}_iU_l = \nabla\xi_l\cdot(\tilde{U}_i\boldsymbol{u})$ の微分が用いられた．

[6] 第 1 式から第 2 式へは $\dfrac{\partial}{\partial\xi_i}\Big(\dfrac{\partial\boldsymbol{x}}{\partial\xi_j}\cdot\nabla\xi_l\Big) = \dfrac{\partial}{\partial\xi_i}\delta_{jl} = 0$ なる関係，第 3 式へは上の Christoffel 記号の式，それから第 4 式へは $\nabla\xi_l\cdot(\partial\boldsymbol{x}/\partial\xi_k) = \delta_{kl}$ を用いている．

SMAC 法の陰解法化は，前にも述べたように，NS 方程式の前段の式 (14.20a) だけを陰解法化すれば達成でき，それによって時間間隔 $\Delta t$ を大きく取ることが可能になる．一般曲線座標系の SMAC Δ形陰解法の式は次のようになる．

$$\Big\{J+\Delta t\,\theta\Big(\tilde{U}_i\frac{\partial}{\partial\xi_i}-\mathfrak{D}_l\Big)\Big\}\Delta U_l^* = rhs_l^n, \qquad (l=1,\,2,\,3) \tag{14.21a}$$

$$rhs_l = -\Delta t\Big(C_l+\tilde{g}^{li}\frac{\partial p}{\partial\xi_i}-D_l\Big), \qquad \tilde{U}_l^* = \tilde{U}_l^n + J\Delta U_l^*$$

$$\tilde{U}_l^{n+1} = \tilde{U}_l^* - \Delta t\,\tilde{g}^{li}\frac{\partial\phi}{\partial\xi_i} \tag{14.21b}$$

$$\frac{\partial}{\partial\xi_l}\Big(\tilde{g}^{li}\frac{\partial\phi}{\partial\xi_i}\Big) = \frac{1}{\Delta t}\frac{\partial}{\partial\xi_l}\tilde{U}_l^*, \qquad p^{n+1} = p^n+\phi \tag{14.21c}$$

定常流れの計算では，解が収束すれば $\tilde{U}_l^* = \tilde{U}_l^n$ であるから，式 (14.21a) の右辺 $rhs$ は残差，$\Delta U_l^*$ は修正値を表している．解が収束すればこの残差はゼロに近づき，修正値もゼロに近づく．したがってこの式の右辺は，解の精度に直接関係するので精度良く離散化しなければならないが，左辺の演算子は大胆に近似することができ，1 次上流差分が用いられ，拡散項は主要部のみ考慮され，また近似因子法が適用される．$\mathfrak{D}_l$ は拡散項の近似演算子で無視することもできるが，岐点の近傍の流れではこの項の働きが大きいのでここではその主要部のみ残すことにすれば，一般に $g_{ii} > g_{ij}\ (i\neq j)$ であるから次のようになる．

$$\mathfrak{D}_1\cdot = \nu_{\text{eff}}\Big\{\frac{\partial}{\partial\xi}\Big(\tilde{g}_{22}\,\tilde{g}_{33}\frac{\partial}{\partial\xi}J\,\cdot\Big)+\frac{\partial}{\partial\eta}\Big(\tilde{g}_{33}\frac{\partial}{\partial\eta}g_{11}\cdot\Big)+\frac{\partial}{\partial\zeta}\Big(\tilde{g}_{22}\frac{\partial}{\partial\zeta}g_{11}\cdot\Big)\Big\} \tag{14.22}$$

また 2 次元の場合には

$$\mathfrak{D}_1\cdot = \nu_{\text{eff}}\Big\{\frac{\partial}{\partial\xi}\Big(\tilde{g}_{22}\frac{\partial}{\partial\xi}\,\cdot\Big)+\frac{\partial^2}{\partial\eta^2}(\tilde{g}_{11}\cdot)\Big\}$$

非定常流れの計算では，$\theta = 1/2$ の Crank-Nicholson 法を用い，各時間ステップごとに Newton 反復法で反復計算を行うことにする．この非定常流れの SMAC Δ形陰解法の式は次のようになる．

$$\Big\{J+\Delta t\theta\Big(\tilde{U}_i\frac{\partial}{\partial\xi_i}-\mathfrak{D}_l\Big)\Big\}\Delta U_l^{*(m)} = -\big(\tilde{U}_l^{(m-1)}-\tilde{U}_l^n\big)+\frac{1}{2}\big(rhs_l^n+rhs_l^{(m-1)}\big), \quad (l=1,\,2,\,3) \tag{14.23a}$$

$$\tilde{U}_l^{*(m)} = \tilde{U}_l^{(m-1)} + J\Delta U_l^{*(m)}$$

$$\tilde{U}_l^{(m)} = \tilde{U}_l^{*(m)} - \frac{1}{2}\Delta t\,\tilde{g}^{li}\frac{\partial\phi^{(m)}}{\partial\xi_i} \tag{14.23b}$$

$$\frac{\partial}{\partial\xi_l}\Big(\tilde{g}^{li}\frac{\partial\phi^{(m)}}{\partial\xi_i}\Big) = \frac{2}{\Delta t}\frac{\partial}{\partial\xi_l}\tilde{U}_l^{*(m)}, \qquad p^{(m)} = p^{(m-1)}+\phi^{(m)} \tag{14.23c}$$

上添字 $^{(m)}$ は時間ステップ $(n+1)$ における第 $m$ 近似値を意味する．$\tilde{U}_l^{(0)} = \tilde{U}_l^n$, $p^{(0)} = p^n$ と置いて計算を始め，解が収束すれば $\tilde{U}_l^{(m)} = \tilde{U}_l^{n+1}$, $p^{(m)} = p^{n+1}$ となる．一般に定常流れが収束するまでには数千回の反復が必要であるが，非定常流れの各時間ステップの解は 3～5 回の反復で十分である．定常流れの収束までに多くの反復を要するのは拡散効果によるもので，一方 1 時間ステップに拡散効果の実質及ぶ範囲は限られるので 1 時間ステップ当たりの反復回数は少しで済むことになる．

## 14.3 バックステップ流路流れのプログラム

前章と同じバックステップ流路を通る流れを，長方形格子の変わりに曲線座標格子を用い SMAC Δ形陰解法で解く１つのプログラムを示す．ただしレイノルズ数がある程度高くなっても剥離域が下流境界に達しないように下流側流路はかなり延長されている．このプログラムは，長方形格子の SMAC Δ形陰解法のプログラムとだいたい同じであるがかなり長大になるので，始めに図 14.2 のフローチャートを使って計算手順の概要を説明する．このプログラムではメインプログラムで `Mode` を指定することにより，定常または非定常流れ，低レイノルズ数または高レイノルズ数流れの都合４つの場合を選択できる．定常流れの初期値はこのプログラムの中で与えられるが，非定常流れの初期値は別途計算したものを読込ませなければならない．低レイノルズ数流れは対流項を 2 次中心差分で計算するもの，高レイノルズ数流れは 3 次の Chakravarthy-Osher TVD スキームで計算するものである．

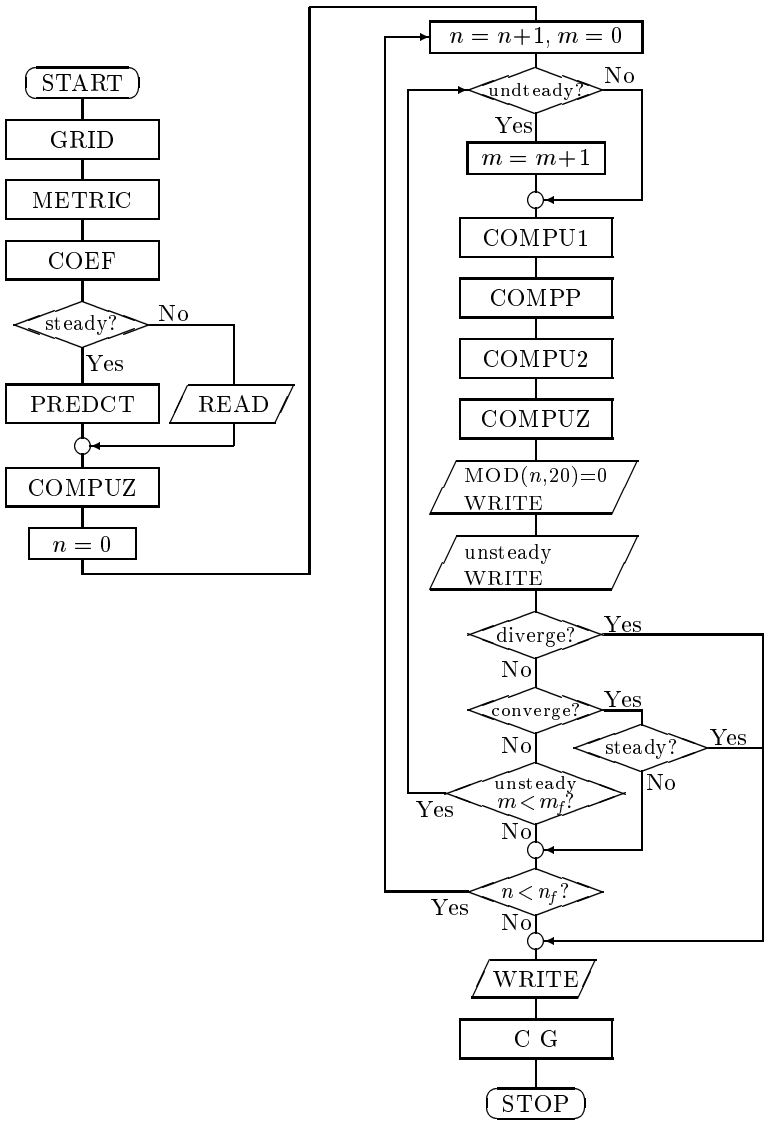


図 14.2: SMAC Δ形陰解法のフローチャート

このフローチャートによれば，始めに GRID で格子形成，METRIC で後の計算に必要な測度，ヤコビアン，測度テンソル，また COEF で圧力差分方程式の係数の設定が行われ，それから定常流れの予測値の計算または非定常流れの初期値の読込みが行われる．それから左側の反復計算に移るが，$n$ は時間ステップ，$m$ は非定常流れの各時間ステップにおける反復数である．各反復計算では式 (14.21) または (14.23) により COMPU1 で $\tilde{\boldsymbol{U}}^*$，COMPP で $\phi$ と $p$，COMPU2 で $\tilde{\boldsymbol{U}}$，また COMPUZ で $\zeta$ の計算が行われる．$n$20 回ごとに NS 差分方程式の最大残差値と圧力差分方程式の最大残差値を打出し，また非定常流れでは CG に必要なデータをファイルに書込む．この反復計算は解が発散したときには直ちに中止し，解が収束または規定の反復数に達した時に次のステップまたは計算を終了する．終了直前には計算の最終結果をファイルに書込み CG で流れの可視化を行う．以下にバックステップ流路を通る定常流れのプログラムを示す．

```
PROGRAM MAIN
! ********************************************************************************
!  Flow Problem: 2D Incompressible Flow through Backward Facing Stepped Duct
!  Numerical Method: General Curvilinear Coordinate Grid, SMAC Delta-Form Implicit Method,
!      Chakravarthy-Osher TVD Scheme
! ********************************************************************************
PARAMETER (if=140,jf=25)
DIMENSION x(2,0:if,0:jf),UJ(2,0:if,0:jf),u(2,0:if,0:jf),p(0:if,0:jf),phi(0:if,0:jf),     &
    z(0:if,0:jf),UJX(2,0:if,0:jf),f(0:if,0:jf),co(0:if,0:jf,-1:1,-1:1),locf(2,8),fmax(8)
COMMON //n,Re,dt,i0,j0,lowRe,steady,unsteady
COMMON /COMPUZ1/UJX
LOGICAL lowRe,steady,unsteady
CHARACTER*4 z1,z2;  CHARACTER*10 z3(8)
DATA nf,mf,Re,dt,Mode/5000,5,1000.,1.,2/                !nf=max n, mf=max(m), Re=Reynolds number
i0=if-1; j0=jf-1
! Mode=1: for low Re steady flow
! Mode=2: for high Re steady flow
! Mode=3: for low Re unsteady flow
! Mode=4: for high Re unsteady flow
steady   = (Mode==1 .OR. Mode==2)
unsteady = (Mode==3 .OR. Mode==4)
lowRe    = (Mode==1 .OR. Mode==3)
CALL GRID(x,if,jf)                                          !grid generation
CALL METRIC(x)                                              !metrics, metric tensors
CALL COEF(co)                                               !coefficients of press-fde
IF(steady)CALL PREDCT(x,UJ,u,p)                             !predict JU and p
IF(unsteady)READ(10)x,UJ,u,p                                !undteady: initial data
CALL COMPUZ(UJ,u,z,locf,fmax,CFLmax)                      !velocities vorticity & divergence
OPEN(20,FILE='OUTPUT.dat')
WRITE(20,60); IF(unsteady)WRITE(20,61)
n=0; 100 n=n+1                                              !n: step
m=0; 101 IF(unsteady)m=m+1                                  !m: approx
CALL COMPU1(m,UJ,u,p,z,locf,fmax)                           !compute JU^*
CALL COMPP (UJ,p,phi,co,resP,locf,fmax)                     !compute phi and p
CALL COMPU2(UJ,phi)                                         !compute JU
CALL COMPUZ(UJ,u,z,locf,fmax,CFLmax)                      !velocities vorticity & divergence
! Decide convergence and output computational results
IF(MOD(n,20)==0 .AND. (steady.OR.m==1))THEN
  i=if; resNS=0.
```

```
  resNS=AMAX1(fmax(1),-fmax(2),fmax(3),-fmax(4))
  flow=3./8.*(3.*UJ(1,i,0)+UJ(1,i,1)+UJ(1,i,23)+3.*UJ(1,i,24))        !outlet flow rate
  DO j=1,23,2; flow=flow+(UJ(1,i,j)+4.*UJ(1,i,j+1)+UJ(1,i,j+2))/3.; ENDDO
  CALL CPU_TIME(sec)
  WRITE(20,62)n,resNS,resP,CFLmax,flow,sec                            !print residuals
ENDIF
60 FORMAT(/1H  3X 'n', 6X 'resNS', 6X 'resP', 7X 'CFLmax', 4X 'outflow', 5X 'CPU-time'/)
61 FORMAT(1H  5X 'm', 6X 'resNS', 6X 'resP'/)
62 FORMAT(1H  I5, 2F11.5, 2X F9.3, F11.5, 2X F9.2)
IF(resNS>2. .OR. resP>2.)                          GOTO 110          !diverge
IF(resNS<.0001 .AND. resP<.0001)THEN                                 !converge
  IF(steady)                                       GOTO 110
  IF(unsteady)                                     GOTO 111
ENDIF
IF(unsteady.AND.m<mf)                              GOTO 101          !unsteady:(m)-approx
111 IF(n<nf)                                       GOTO 100          !n-step
110 CONTINUE
DO k=1,6
  SELECT CASE(k)
    CASE(1); FORALL(i=0:if,j=0:jf)f(i,j)=x(1,i,j) ; z1="   x"; z2="    "; mz= 0
    CASE(2); FORALL(i=0:if,j=0:jf)f(i,j)=x(2,i,j) ; z1="   y"; z2="    "; mz= 0
    CASE(3); FORALL(i=0:if,j=0:jf)f(i,j)=u(1,i,j) ; z1="   u"; z2="    "; mz= 0
    CASE(4); FORALL(i=0:if,j=0:jf)f(i,j)=u(2,i,j) ; z1="   v"; z2="    "; mz= 0
    CASE(5); FORALL(i=0:if,j=0:jf)f(i,j)=p(i,j)   ; z1="   p"; z2="*10 "; mz= 1
    CASE(6); FORALL(i=0:if,j=0:jf)f(i,j)=z(i,j)   ; z1="zeta"; z2="/10 "; mz=-1
  ENDSELECT                                                          !mz:scale factor
  CALL WRTF(f,z1,z2,mz,n)                                  !print out computational results
ENDDO
CALL REATTACH(x,UJ,UJX,if,jf,xrap,k)                       !compute reattachment point xrap
WRITE(20,'(5X A7,F5.1,3X I4)')'xrap = ',xrap,k             !print out xrap
! Output locations and values of maximum and minimum of residuals and divergence
DATA z3/'maxUq =','minUq =','maxVq =','minVq =','maxphi =','minphi =','maxdiv =','mindiv ='/
FORALL(k=1:2,i=1:8)locf(k,i)=locf(k,i)-1
DO i=1,8; WRITE(20,'(5X A10,2I4,3X E10.3)')z3(i),(locf(k,i),k=1,2),fmax(i); ENDDO
CALL StepDF_CG(x,u,p,z,if,jf)                              !computer graphics for steady
CLOSE(20)
END PROGRAM MAIN

! ********** Generate curvilinear coordinate grid by analytical method
SUBROUTINE GRID(x,if,jf)                                             !if=140,jf=25
! Generate by sloving boundary value problem of elliptic pde
DIMENSION x(2,0:if,0:jf),x0(2,0:if,0:jf),xxi(4,0:if,0:jf),aJ(0:if,0:jf),g(3,0:if,0:jf), &
          P(0:if,0:jf),Q(0:if,0:jf),f(0:if,0:jf)
CHARACTER*4 z1
DATA dxi,det,al1/3*1./
nf=200; i1=35; i2=40; i0=if-1; j0=jf-1
OPEN(20,FILE='OUTPUT.dat')
! ***** Give values of grid coordinates on boundaries by geometrical series
! a+ar+ar^2+...+ar^(n-1)=a(r^n-1)/(r-1)
x(1,i1,0)=0.; x(2,i1,0)=0.
DO i=i1-1,0,-1
  x(1,i,0)=x(1,i+1,0)-.09975*1.055**(i1-i-1); x(2,i,0)=0.            !bottom wall
```

```
ENDDO
DO i=i1+1,i2
  x(1,i,0)=0.; x(2,i,0)=x(2,i-1,0)-.10045*1.35**(i-i1-1)              !step wall
ENDDO
DO i=i2+1,70
  x(1,i,0)=x(1,i-1,0)+.2*1.05**(i-i2-1)                              !bottom wall
ENDDO
FORALL(i=71:if)x(1,i,0)=x(1,70,0)+.8232*FLOAT(i-70)                  !bottom wall
FORALL(i=i2:if)x(2,i,0)=-1.
FORALL(i=0:if)x(2,i,jf)=3.                                           !top wall
! ***** Give predict values of x on top bound
FORALL(i=0:i1)   x(1,i,jf)=x(1,i,0)
FORALL(i=i1+1:i2)x(1,i,jf)=x(1,i1,jf)-x(2,i,0)
FORALL(i=i2+1:if)x(1,i,jf)=x(1,i2,jf)+(x(1,if,0)-1.)/x(1,if,0)*x(1,i,0)
! ***** Determine starting values of x,y by interporation
DO i=0,if; DO j=1,j0
  et=j/FLOAT(jf); alp=(-et*et+1.6*et+.4)*et
  ! The cubic polynomial is formed to satisfy the conds of f(0)=0, f(1)=1, f'(0)=.4, f'(1)=.6
  ! for finer grid near walls. But it is impossible sufficiently to control grid spaces by it.
  x(1,i,j)=(1.-alp)*x(1,i,0)+alp*x(1,i,jf)
  x(2,i,j)=(1.-alp)*x(2,i,0)+alp*x(2,i,jf)
ENDDO; ENDDO
n=0; 100 n=n+1; IF(n==10)al1=1.4
! ***** Compute x_xi y_xi x_eta y_eta
DO l=1,2
  FORALL(i=1:i0,j=0:jf)xxi(l  ,i,j)=(x(l,i+1,j)-x(l,i-1,j))/2./dxi    !x_xi, y_xi
  FORALL(i=0:if,j=1:j0)xxi(l+2,i,j)=(x(l,i,j+1)-x(l,i,j-1))/2./det    !x_eta, y_eta
  FORALL(j=0:jf)xxi(l  , 0,j)=-(3.*x(l, 0,j)-4.*x(l, 1,j)+x(l,   2,j))/2./dxi
  FORALL(j=0:jf)xxi(l  ,if,j)= (3.*x(l,if,j)-4.*x(l,i0,j)+x(l,i0-1,j))/2./dxi
  FORALL(i=0:if)xxi(l+2,i, 0)=-(3.*x(l,i, 0)-4.*x(l,i, 1)+x(l,i,   2))/2./det
  FORALL(i=0:if)xxi(l+2,i,jf)= (3.*x(l,i,jf)-4.*x(l,i,j0)+x(l,i,j0-1))/2./det
ENDDO
! ***** Compute J J^2; xi_x eta_x xi_y eta_y; g_11 g_12 g_22
FORALL(i=0:if,j=0:jf)
  aJ(i,j)=xxi(1,i,j)*xxi(4,i,j)-xxi(3,i,j)*xxi(2,i,j)                !J
  g(1,i,j)=(xxi(1,i,j)*xxi(1,i,j)+xxi(2,i,j)*xxi(2,i,j))/aJ(i,j)     !g_11/J
  g(2,i,j)=(xxi(1,i,j)*xxi(3,i,j)+xxi(2,i,j)*xxi(4,i,j))/aJ(i,j)     !g_12/J
  g(3,i,j)=(xxi(3,i,j)*xxi(3,i,j)+xxi(4,i,j)*xxi(4,i,j))/aJ(i,j)     !g_22/J
ENDFORALL
! ***** Compute control functions P and Q
! P: control i1-line and i2-line, Q: control grid spacing of j-direction
FORALL(i=0:if,j=0:jf)P(i,j)=0.
FORALL(i=0:if,j=0:jf)Q(i,j)=0.
FORALL(i=1:i1-1,j=1:jf)P(i,j)=2./(x(1,i+1,0)-x(1,i-1,0)) &           !xi_xx
    *(1./(x(1,i+1,0)-x(1,i,0))-1./(x(1,i,0)-x(1,i-1,0)))
FORALL(i=i1+1:i2-1,j=1:15)P(i,j)=(15-j)/15.*2./(x(2,i+1,0)-x(2,i-1,0)) & !c*xi_yy
    *(1./(x(2,i+1,0)-x(2,i,0))-1./(x(2,i,0)-x(2,i-1,0)))
FORALL(i=i2+1:i0,j=1:10)  P(i,j)=(10-j)/10.*2./(x(1,i+1,0)-x(1,i-1,0)) & !c*xi_xx
    *(1./(x(1,i+1,0)-x(1,i,0))-1./(x(1,i,0)-x(1,i-1,0)))
FORALL(j=1:jf)P(i1,j)=(P(i1-1,j)+P(i1+1,j))/2.
FORALL(j=1:jf)P(i2,j)=(P(i2-1,j)+P(i2+1,j))/2.
DO i=0,if; DO j=1,jf
```

```
  coef=25./(x(2,i,jf)-x(2,i,0))/(x(2,i,jf)-x(2,i,0))            !coef=(d eta/d te)*(d ty/dy)^2
  ty=(x(2,i,j)-x(2,i,0))/(x(2,i,jf)-x(2,i,0))                   !ty=(y-y_0)/(y_f-y_0)
!  Q(i,j)=coef*(6.*ty-4.)      !te(ty)=ty^3-2ty^2+2ty         te'(0,.5,1)=(12/6 4.5/6  6/6)
  Q(i,j)=coef*(8.*ty-5.)       !te(ty)=(8ty^3-15ty^2+13ty)/6  ty'        =(13/6   4/6  7/6)
!  Q(i,j)=coef*(10.*ty-6.)     !te(ty)=(5ty^3-9ty^2+7ty)/3    te'        =(14/6 3.5/6  8/6)
!  Q(i,j)=coef*(12.*ty-7.)     !te(ty)=(4ty^3-7ty^2+5ty)/2    te'        =(15/6   3/6  9/6)
!  Q(i,j)=coef*(18.*ty-10.)    !te(ty)=3ty^3-5ty^2+3ty        te'        =(18/6 1.5/6 12/6)
! for all te(ty), te(0, .5, 1.0)=(0, .625, 1.0),
! Q=d^2eta/dy^2=(d ty/dy)^2*te''(ty)*(d eta/d te)=coef*te''(ty)
! Lower Q's correspond to grids whose spaces are larger difference.
  IF(i>i1.AND.i<i1+10.AND.j<10) &                          !modify grid spaces near corner
  Q(i,j)=Q(i,j)-coef*12.*(1.-FLOAT(j)/10.)*(1.-ABS(i1+5-i)/5.)
ENDDO; ENDDO
! ***** Solve difference eqs of L(x)=-J(x_xiP+x_etaQ), L(y)=-J(y_xiP+y_etaQ) by SOR
FORALL(l=1:2,i=0:if,j=0:jf)x0(l,i,j)=x(l,i,j)
nn=0; 110 nn=nn+1
DO 10 l=1,2; DO 10 i=1,i0; DO 10 j=1,j0
  w=(g(3,i,j)*(x(l,i-1,j)+x(l,i+1,j))                                  &
    -g(2,i,j)*(x(l,i-1,j-1)-x(l,i-1,j+1)-x(l,i+1,j-1)+x(l,i+1,j+1))/2. &
    +g(1,i,j)*(x(l,i,j-1)+x(l,i,j+1))                                  &
    +aJ(i,j)*(xxi(l,i,j)*P(i,j)+xxi(l+2,i,j)*Q(i,j)))/2./(g(1,i,j)+g(3,i,j))
10 x(l,i,j)=x(l,i,j)+al1*(w-x(l,i,j))
j=jf; DO 11 i=1,i0                                                     !top wall
  w=(g(3,i,j)*(x(1,i-1,j)+x(1,i+1,j))+g(1,i,j)*2.*x(1,i,j-1) &
    +aJ(i,j)*(xxi(1,i,j)*P(i,j)+xxi(3,i,j)*Q(i,j)))/2./(g(1,i,j)+g(3,i,j))
11 x(1,i,j)=x(1,i,j)+al1*(w-x(1,i,j))
i=0; DO 12 j=1,j0                                                      !inlet bound
  w=(g(3,i,j)*2.*x(2,1,j)+g(1,i,j)*(x(2,i,j-1)+x(2,i,j+1)) &
    +aJ(i,j)*(xxi(2,i,j)*P(i,j)+xxi(4,i,j)*Q(i,j)))/2./(g(1,i,j)+g(3,i,j))
12 x(2,i,j)=x(2,i,j)+al1*(w-x(2,i,j))
i=if; DO 13 j=1,j0                                                     !outlet bound
  w=(g(3,i,j)*2.*x(2,i-1,j)+g(1,i,j)*(x(2,i,j-1)+x(2,i,j+1)) &
    +aJ(i,j)*(xxi(2,i,j)*P(i,j)+xxi(4,i,j)*Q(i,j)))/2./(g(1,i,j)+g(3,i,j))
13 x(2,i,j)=x(2,i,j)+al1*(w-x(2,i,j))
IF(nn<10)                                        GOTO 110
! ***** Decide convergence of x,y
adx=0.
DO i=0,if; DO j=1,j0
  adx=AMAX1(adx,ABS(x(1,i,j)-x0(1,i,j))+ABS(x(2,i,j)-x0(2,i,j)))       !|dx|+|dy|
ENDDO; ENDDO
IF(n==1)WRITE(20,60)
IF(MOD(n,10)==0)WRITE(20,61) n,adx
60 FORMAT(/1H  3X 'n', 7X 'adx'/)
61 FORMAT(1H  I5, F11.5)
IF(n<nf.AND.adx>1.E-4.AND.adx<5.)                GOTO 100
ENDSUBROUTINE GRID

! ********** Computation of metrics and metric tensors
SUBROUTINE METRIC(x)
PARAMETER(if=140,jf=25,if1=280,jf1=50)
DIMENSION x(2,0:if,0:jf),x1(2,0:if1,0:jf1),xxi(4,0:if1,0:jf1),aJ(0:if1,0:jf1), &
       gU(3,0:if,0:jf),gV(3,0:if,0:jf),xix(4,0:if1,0:jf1),dxix(0:if,0:jf,2,4), &
```

```
      w1(0:if),w2(0:jf),w11(0:if1),w12(0:if1),w21(0:jf1),w22(0:jf1)
COMMON //na,Re,dt,i0,j0,lowRe,steady,unsteady
COMMON /METRIC1/x1 /METRIC2/aJ /METRIC3/gU,gV /METRIC4/xix /METRIC5/dxix
! ***** Determine x1
DO l=1,2; DO j=0,jf
  FORALL(i=0:if)w1(i)=x(l,i,j); CALL INTP(w1,w11,if)
  FORALL(i=0:if1)x1(l,i,2*j)=w11(i)
ENDDO; ENDDO
x1(1,69,0)=(-x(1,33,0)+6.*x(1,34,0)+3.*x(1,35,0))/8.; x1(2,69,0)=0.
x1(1,71,0)=0.; x1(2,71,0)=(3.*x(2,35,0)+6.*x(2,36,0)-x(2,37,0))/8.
x1(1,79,0)=0.; x1(2,79,0)=(-x(2,38,0)+6.*x(2,39,0)+3.*x(2,40,0))/8.
x1(1,81,0)=(3.*x(1,40,0)+6.*x(1,41,0)-x(1,42,0))/8.; x1(2,81,0)=-1.
DO l=1,2; DO i=0,if1
  FORALL(j=0:jf)w2(j)=x1(l,i,2*j); CALL INTP(w2,w21,jf)
  FORALL(j=0:jf1)x1(l,i,j)=w21(j)
ENDDO; ENDDO
! ***** Determine xxi, J
DO l=1,2; DO j=0,jf1
  FORALL(i=0:if1)w11(i)=x1(l,i,j); CALL DIFX(w11,w12,if1,.5)
  FORALL(i=0:if1)xxi(l,i,j)=w12(i)                                  !x_xi y_xi at point X
ENDDO; ENDDO
DO l=1,2; DO i=0,if1
  FORALL(j=0:jf1)w21(j)=x1(l,i,j); CALL DIFX(w21,w22,jf1,.5)
  FORALL(j=0:jf1)xxi(l+2,i,j)=w22(j)                                !x_eta y_eta at X
ENDDO; ENDDO
FORALL(i=0:if1,j=0:jf1)aJ(i,j)=xxi(1,i,j)*xxi(4,i,j)-xxi(3,i,j)*xxi(2,i,j)  !Jacobian J
! ***** Determine g_11/J g_12/J g_22/J at U and V
DO i=0,if; ii=2*i; DO j=0,j0; jj=2*j+1
  gU(1,i,j)=(xxi(1,ii,jj)*xxi(1,ii,jj)+xxi(2,ii,jj)*xxi(2,ii,jj))/aJ(ii,jj) !g_11/J at U
  gU(2,i,j)=(xxi(1,ii,jj)*xxi(3,ii,jj)+xxi(2,ii,jj)*xxi(4,ii,jj))/aJ(ii,jj) !g_12/J at U
  gU(3,i,j)=(xxi(3,ii,jj)*xxi(3,ii,jj)+xxi(4,ii,jj)*xxi(4,ii,jj))/aJ(ii,jj) !g_22/J at U
ENDDO; ENDDO
DO i=0,i0; ii=2*i+1; DO j=0,jf; jj=2*j
  gV(1,i,j)=(xxi(1,ii,jj)*xxi(1,ii,jj)+xxi(2,ii,jj)*xxi(2,ii,jj))/aJ(ii,jj) !g_11/J at V
  gV(2,i,j)=(xxi(1,ii,jj)*xxi(3,ii,jj)+xxi(2,ii,jj)*xxi(4,ii,jj))/aJ(ii,jj) !g_12/J at V
  gV(3,i,j)=(xxi(3,ii,jj)*xxi(3,ii,jj)+xxi(4,ii,jj)*xxi(4,ii,jj))/aJ(ii,jj) !g_22/J at V
ENDDO; ENDDO
! ***** Determine xix
DO i=0,if1; DO j=0,jf1
  xix(1,i,j)= xxi(4,i,j)/aJ(i,j); xix(2,i,j)=-xxi(3,i,j)/aJ(i,j)      !xi_x, xi_y at X
  xix(3,i,j)=-xxi(2,i,j)/aJ(i,j); xix(4,i,j)= xxi(1,i,j)/aJ(i,j)     !eta_x, eta_y at X
ENDDO; ENDDO
! ***** Determine (xi_x)_xi
DO j=0,j0; jj=2*j+1
  DO i=1,i0; ii=2*i
    dxix(i,j,1,1)=xix(1,ii+1,jj)-xix(1,ii-1,jj)                     !(xi_x)_xi  at U
    dxix(i,j,1,3)=xix(2,ii+1,jj)-xix(2,ii-1,jj)                     !(xi_y)_xi  at U
  ENDDO
  dxix(if,j,1,1)=3.*xix(1,if1,jj)-4.*xix(1,if1-1,jj)+xix(1,if1-2,jj)
  dxix(if,j,1,3)=3.*xix(2,if1,jj)-4.*xix(2,if1-1,jj)+xix(2,if1-2,jj)
  DO i=1,if; ii=2*i
    dxix(i,j,1,2)=xix(1,ii,jj+1)-xix(1,ii,jj-1)                     !(xi_x)_eta at U
```

```
    dxix(i,j,1,4)=xix(2,ii,jj+1)-xix(2,ii,jj-1)                          !(xi_y)_eta at U
ENDDO; ENDDO
DO i=0,i0; ii=2*i+1; DO j=1,j0; jj=2*j
  dxix(i,j,2,1)=xix(3,ii+1,jj)-xix(3,ii-1,jj)                            !(eta_x)_xi  at V
  dxix(i,j,2,3)=xix(4,ii+1,jj)-xix(4,ii-1,jj)                            !(eta_y)_xi  at V
  dxix(i,j,2,2)=xix(3,ii,jj+1)-xix(3,ii,jj-1)                            !(eat_x)_eta at V
  dxix(i,j,2,4)=xix(4,ii,jj+1)-xix(4,ii,jj-1)                            !(eta_y)_eta at V
ENDDO; ENDDO
ENDSUBROUTINE METRIC

! ********** Prediction of volume flux and pressure
SUBROUTINE PREDCT(x,UJ,u,p)
PARAMETER(if=140,jf=25,if1=280,jf1=50)
DIMENSION x(2,0:if,0:jf),UJ(2,0:if,0:jf),u(2,0:if,0:jf),p(0:if,0:jf),x1(2,0:if1,0:jf1), &
          aJ(0:if1,0:jf1),xix(4,0:if1,0:jf1)
COMMON //na,Re,dt,i0,j0,lowRe,steady,unsteady
COMMON /METRIC1/x1 /METRIC2/aJ /METRIC4/xix
! ***** Set up starting values of u, U and p
DO j=0,jf; ty=x1(2,0,2*j)/x1(2,0,jf1)
  u(1,0,j)=6.*ty*(1.-ty)                                                 !mean velocity at inlet=1
ENDDO
DO j=0,j0; ty=x1(2,0,2*j+1)/x1(2,0,jf1)
  UJ(1,0,j)=aJ(0,2*j+1)*xix(1,0,2*j+1)*6.*ty*(1.-ty)
  FORALL(i=1:if)UJ(1,i,j)=UJ(1,0,j)
ENDDO
Dp=1./2.-.75*.75/2.                                          !recovery pressure in step duct
p0=(4./3.*10.+9./16.*70.9)/Re-Dp                                         !inlet pressure
FORALL(i=0:35,j=0:j0)p(i,j)=p0+4./3.*(x(1, 0,j)-x1(1,2*i+1,2*j+1))/Re
FORALL(i=100:i0,j=0:j0)p(i,j)=9./16.*(x(1,if,j)-x1(1,2*i+1,2*j+1))/Re
FORALL(i=36:99,j=0:j0)p(i,j)=(p(100,j)*(x1(1,2*i+1,2*j+1)-x1(1,71,2*j+1)) &
  +p(35,j)*(x1(1,201,2*j+1)-x1(1,2*i+1,2*j+1)))/(x1(1,201,2*j+1)-x1(1,71,2*j+1))
ENDSUBROUTINE PREDCT

! ********** Compute JU^* and JV^* by delta-form implicit method using Chakravarthy-Osher
!            type TVD scheme
SUBROUTINE COMPU1(ma,UJ,u,p,z,locf,fmax)
PARAMETER(if=140,jf=25,if1=280,jf1=50)
DIMENSION UJ(2,0:if,0:jf),Uq(2,0:if,0:jf),u(2,0:if,0:jf),p(0:if,0:jf),z(0:if,0:jf),         &
    aJ(0:if1,0:jf1),gU(3,0:if,0:jf),gV(3,0:if,0:jf),xix(4,0:if1,0:jf1),dxix(0:if,0:jf,2,4), &
    UJX(2,0:if,0:jf),hf(0:if),rhs(2,0:if,0:jf),rhs0(2,0:if,0:jf),UJ0(2,0:if,0:jf),          &
    a(140,140,3),b(140,140),c(0:4,0:if,0:jf),f(0:if,0:jf),w(0:if),w1(0:if),w11(0:if1),      &
    w12(0:if1),w13(0:if1),w14(0:if),w2(0:jf),w21(0:jf1),w22(0:jf1),w23(0:jf1),w24(0:jf),    &
    lo(2),locf(2,8),fmax(8)
COMMON //na,Re,dt,i0,j0,lowRe,steady,unsteady
COMMON /METRIC2/aJ /METRIC3/gU,gV /METRIC4/xix /METRIC5/dxix /COMPUZ1/UJX
LOGICAL lowRe,steady,unsteady
DATA theta,beta/1.,.5/                                    !theta:trapezoidal, beta:damping
FORALL(l=1:2,i=0:if,j=0:jf)rhs(l,i,j)=0.
IF(steady.AND.lowRe)THEN                                                 !for training
! ***** Compute convection term using central-differences
DO j=0,j0
  FORALL(i=0:if)w1(i)=UJ(1,i,j); CALL INTP(w1,w11,if)                    !w11=JU_P(JU at poin P)
```

```
  FORALL(i=0:i0)hf(i)=w11(2*i+1)*w11(2*i+1)/aJ(2*i+1,2*j+1)          !hf=JUU_P
  FORALL(i=1:i0)Uq(1,i,j)=hf(i)-hf(i-1)                             !JUU_xi
ENDDO
DO i=1,i0
  FORALL(j=0:jf)hf(j)=UJX(2,i,j)*UJX(1,i,j)/aJ(2*i,2*j)             !hf=JVU_X
  FORALL(j=0:j0)Uq(1,i,j)=Uq(1,i,j)+hf(j+1)-hf(j)                   ! +JVU_eta
ENDDO
DO j=1,j0
  FORALL(i=1:i0)hf(i)=UJX(1,i,j)*UJX(2,i,j)/aJ(2*i,2*j)             !hf=JUV_X
  FORALL(i=1:i0-1)Uq(2,i,j)=hf(i+1)-hf(i)                           !JUV_xi
ENDDO
DO i=1,i0-1
  FORALL(j=0:jf)w2(j)=UJ(2,i,j); CALL INTP(w2,w21,jf)               !w21=JV_P
  FORALL(j=0:j0)hf(j)=w21(2*j+1)*w21(2*j+1)/aJ(2*i+1,2*j+1)         !hf=JVV_P
  FORALL(j=1:j0)Uq(2,i,j)=Uq(2,i,j)+hf(j)-hf(j-1)                   ! +JVV_eta
ENDDO
ELSE
! ***** Compute convection term using Chakravarthy-Osher TVD scheme
DO j=0,j0
  FORALL(i=0:if)w(i)=UJ(1,i,j)/aJ(2*i,2*j+1)                        !w  =U
  FORALL(i=0:if)w1(i)=UJ(1,i,j); CALL INTP(w1,w11,if)               !w11=JU_P
  i=0   ; hf(i)=w11(2*i+1)*(w(i)+w(i+1))/2.                         !near inlet
  i=if-1; hf(i)=w11(2*i+1)*(w(i)+w(i+1))/2.                         !near outlet
  cycle_1: DO i=1,i0; UJP=w11(2*i+1)
    m=1; IF(UJP>0.)m=-1; ip=i+m
    IF(i==i0.AND.m==1)CYCLE cycle_1
    UP=(w(i+1)+w(i))/2.; dU=w(i+1)-w(i); dU1=w(ip+1)-w(ip)
    hf(i)=UJP*UP+ABS(UJP)*(-dU/2.+AMINMOD(dU1,4.*dU)/6.+AMINMOD(dU,4.*dU1)/3.) !hf=JUU_P
  ENDDO cycle_1
  FORALL(i=1:i0)Uq(1,i,j)=hf(i)-hf(i-1)                             !JUU_xi
ENDDO
! ***** Correct convection term just behind corner
FORALL(i=35:37)w(i)=UJ(1,i,0)*UJ(1,i,0)/aJ(2*i,1)
dum=w(36)-w(35); du=w(37)-w(36)
IF(UJ(1,36,0)<=0.)THEN; Uq(1,36,0)=AMINMOD((dum/du+1.)/2.,2.)*du
                  ELSE; Uq(1,36,0)=AMINMOD((du/dum+1.)/2.,2.)*dum; ENDIF
DO i=1,i0
  FORALL(j=0:j0)w(j)=UJ(1,i,j)/aJ(2*i,2*j+1)                        !w=U
  j=0 ; hf(j)=0.                                                    !bottom
  j=1 ; hf(j)=UJX(2,i,j)*(3.*w(j-1)+2.*w(j)-w(j+1)/5.)/4.           !neighbor of bottom wall
  j=j0; hf(j)=UJX(2,i,j)*(3.*w(j)+2.*w(j-1)-w(j-2)/5.)/4.           !neighbor top
  j=jf; hf(j)=0.                                                    !top
  cycle_2: DO j=1,j0; VJX=UJX(2,i,j)
    m=1; IF(VJX>0.)m=-1; jp=j+m
    IF(j==1.AND.m==-1 .OR. j==j0.AND.m==1)CYCLE cycle_2
    UX=(w(j-1)+w(j))/2.; dU=w(j)-w(j-1); dU1=w(jp)-w(jp-1)
    hf(j)=VJX*UX+ABS(VJX)*(-dU/2.+AMINMOD(dU1,4.*dU)/6.+AMINMOD(dU,4.*dU1)/3.) !hf=JVU_X
  ENDDO cycle_2
  FORALL(j=0:j0)Uq(1,i,j)=Uq(1,i,j)+hf(j+1)-hf(j)                   ! +JVU_eta
ENDDO
DO j=1,j0
  FORALL(i=0:i0)w(i)=UJ(2,i,j)/aJ(2*i+1,2*j)                        !w=V
```

```
  i=0 ; hf(i)=0.
  i=1 ; hf(i)=UJX(1,1,j)*(w(i-1)+w(i))/2.                              !neighbor inlet
  i=i0; hf(i)=UJX(1,i,j)*(-w(i-2)+6.*w(i-1)+3.*w(i))/8.                !neighbor outlet
  cycle_3: DO i=2,i0; UJX0=UJX(1,i,j)
    m=1; IF(UJX0>0.)m=-1; ip=i+m
    IF(i==i0.AND.m==1)CYCLE cycle_3
    VX=(w(i-1)+w(i))/2.; dV=w(i)-w(i-1); dV1=w(ip)-w(ip-1)
    hf(i)=UJX0*VX+ABS(UJX0)*(-dV/2.+AMINMOD(dV1,4.*dV)/6.+AMINMOD(dV,4.*dV1)/3.) !hf=JUV_X
  ENDDO cycle_3
  FORALL(i=0:i0-1)Uq(2,i,j)=hf(i+1)-hf(i)                              !JUV_xi
ENDDO
DO i=0,i0-1
  FORALL(j=0:jf)w(j)=UJ(2,i,j)/aJ(2*i+1,2*j)                           !w  =V
  FORALL(j=0:jf)w2(j)=UJ(2,i,j); CALL INTVP(w2,w21,jf)                 !w21=JV_P
  j=0 ; hf(j)=w21(2*j+1)*(12.*w(1)-w(2))/32.                           !near bottom
  j=j0; hf(j)=w21(2*j+1)*(12.*w(j0)-w(j0-1))/32.                       !near top
  cycle_4: DO j=0,j0; VJP=w21(2*j+1)
    m=1; IF(VJP>0.)m=-1; jp=j+m
    IF(j==0.AND.m==-1 .OR. j==j0.AND.m==1)CYCLE cycle_4
    VP=(w(j)+w(j+1))/2.; dV=w(j+1)-w(j); dV1=w(jp+1)-w(ip)
    hf(j)=VJP*VP+ABS(VJP)*(-dV/2.+AMINMOD(dV1,4.*dV)/6.+AMINMOD(dV,4.*dV1)/3.) !hf=JVV_P
  ENDDO cycle_4
  FORALL(j=1:j0)Uq(2,i,j)=Uq(2,i,j)+hf(j)-hf(j-1)                      ! +JVV_eta
ENDDO
ENDIF
! ***** Compute convection term at outlet using upwind-difference scheme
DO j=0,j0; i=if; Uq(1,i,j)=0.
  DO k=1,2
    Uq(1,i,j) = Uq(1,i,j)+xix(k,2*i,2*j+1)                             &
      *(UJ(1,i,j)*(u(k,i,j)+u(k,i,j+1)-u(k,i-1,j)-u(k,i-1,j+1))/2. &
      +(UJX(2,i,j)+UJX(2,i,j+1))/2.*(u(k,i,j+1)-u(k,i,j)))
ENDDO; ENDDO
DO j=1,j0; i=i0; Uq(2,i,j)=0.
  DO k=1,2
    Uq(2,i,j) = Uq(2,i,j)+xix(k+2,2*i+1,2*j)                           &
      *((UJX(1,i,j)+UJX(1,i+1,j))/2.*(u(k,i+1,j)-u(k,i-1,j))/2.    &
      +UJ(2,i,j)*(u(k,i,j+1)+u(k,i+1,j+1)-u(k,i,j-1)-u(k,i+1,j-1))/4.)
ENDDO; ENDDO
! ***** Compute additional, pressure and diffusion terms
DO i=1,if
  FORALL(j=0:jf)w2(j)=u(1,i,j);   CALL INTP(w2,w21,jf)                 !w21=u_U
  FORALL(j=0:jf)w2(j)=u(2,i,j);   CALL INTP(w2,w22,jf)                 !w22=v_U
  FORALL(j=0:jf)w2(j)=UJX(2,i,j); CALL INTP(w2,w23,jf)                 !w23=JV_U
  FORALL(j=0:j0)w2(j)=(p(i-1,j)+p(i,j))/2.; CALL DIFP(w2,w24,jf,1.)    !w24=(p_eta)_U
  DO j=0,j0; jj=2*j+1
    ad = w21(jj)*(UJ(1,i,j)*dxix(i,j,1,1)+w23(jj)*dxix(i,j,1,2)) &     !additional term
        +w22(jj)*(UJ(1,i,j)*dxix(i,j,1,3)+w23(jj)*dxix(i,j,1,4))
    IF(i==if)ad = 0.
    pr = gU(3,i,j)*(p(i,j)-p(i-1,j))-gU(2,i,j)*w24(j)                  !pressure term
    Uq(1,i,j)  = Uq(1,i,j)-ad+pr+(z(i,j+1)-z(i,j))/Re                  !residuals of NS eqn
    rhs(1,i,j) = -dt*Uq(1,i,j)                                         !rhs of linear eqns
ENDDO; ENDDO
```

```
DO j=1,j0
  FORALL(i=0:if)w1(i)=u(1,i,j);   CALL INTP(w1,w11,if)                !w11=u_V
  FORALL(i=0:if)w1(i)=u(2,i,j);   CALL INTP(w1,w12,if)                !w12=v_V
  FORALL(i=0:if)w1(i)=UJX(1,i,j); CALL INTP(w1,w13,if)                !w13=JU_V
  FORALL(i=0:if)w1(i)=(p(i,j-1)+p(i,j))/2.; CALL DIFP(w1,w14,if,1.)   !w14=(p_xi)_V
  i=i0; w14(i)=(w1(i+1)-w1(i-1))/2.                                   !outlet
  DO i=0,i0; ii=2*i+1
    ad = w11(ii)*(w13(ii)*dxix(i,j,2,1)+UJ(2,i,j)*dxix(i,j,2,2)) &    !additional term
        +w12(ii)*(w13(ii)*dxix(i,j,2,3)+UJ(2,i,j)*dxix(i,j,2,4))
    IF(i==i0)ad = 0.
    pr = -gV(2,i,j)*w14(i)+gV(1,i,j)*(p(i,j)-p(i,j-1))                !pressure term
    Uq(2,i,j)  = Uq(2,i,j)-ad+pr-(z(i+1,j)-z(i,j))/Re                 !residuals of NS eqn
    rhs(2,i,j) = -dt*Uq(2,i,j)                                        !rhs of linear eqns
ENDDO; ENDDO
IF(unsteady)THEN                                                      !only unsteady
  IF(ma==1)FORALL(l=1:2,i=0:if,j=0:jf) UJ0(l,i,j)= UJ(l,i,j)
  IF(ma==1)FORALL(l=1:2,i=0:if,j=0:jf)rhs0(l,i,j)=rhs(l,i,j)
  FORALL(l=1:2,i=0:if,j=0:jf)rhs(l,i,j)=-(UJ(l,i,j)-UJ0(l,i,j))+(rhs0(l,i,j)+rhs(l,i,j))/2.
ENDIF
! Compute UJ^* by implicit SMAC-scheme
tt=dt*theta
DO i=1,if; DO j=0,j0
  UJp=(UJ(1,i,j)+ABS(UJ(1,i,j)))/2.; UJm=(UJ(1,i,j)-ABS(UJ(1,i,j)))/2.
  VJU=(UJX(2,i,j)+UJX(2,i,j+1))/2.
  VJp=(VJU+ABS(VJU))/2.; VJm=(VJU-ABS(VJU))/2.
  c(1,i,j)=-tt*(UJp+gU(3,i,j)/Re)                                     !coefs of linear eqs
  c(2,i,j)= tt*(UJm-gU(3,i,j)/Re); IF(i==if)c(2,i,j)=0.
  c(3,i,j)=-tt*(VJp+gU(1,i,j)/Re)
  c(4,i,j)= tt*(VJm-gU(1,i,j)/Re)
  c(0,i,j)=aJ(2*i,2*j+1)-c(1,i,j)-c(2,i,j)-c(3,i,j)-c(4,i,j)
ENDDO; ENDDO
!Compute dU^* by modifies AF-method
DO j=0,j0; l=j+1; DO i=1,if
  a(l,i,1)=c(1,i,j)
  a(l,i,2)=c(0,i,j)
  a(l,i,3)=c(2,i,j); b(l,i)=rhs(1,i,j)
ENDDO; ENDDO
CALL GAUSSZ(a,b,140,jf,if)
FORALL(j=0:j0,i=1:if)rhs(1,i,j)=b(j+1,i)                              !dU^**
DO j=0,j0; l=j+1; DO i=1,if
  a(i,l,1)=c(3,i,j)
  a(i,l,2)=c(0,i,j)
  a(i,l,3)=c(4,i,j); b(i,l)=c(0,i,j)*rhs(1,i,j)
ENDDO; ENDDO
CALL GAUSSZ(a,b,140,if,jf)
FORALL(j=0:j0,i=1:if)rhs(1,i,j)=b(i,j+1)                              !dU^*
! Compute VJ^* by implicit SMAC-scheme
DO i=0,i0; DO j=1,j0
  UJV=(UJX(1,i,j)+UJX(1,i+1,j))/2.
  UJp=(UJV      +ABS(UJV)      )/2.; UJm=(UJV      -ABS(UJV)      )/2.
  VJp=(UJ(2,i,j)+ABS(UJ(2,i,j)))/2.; VJm=(UJ(2,i,j)-ABS(UJ(2,i,j)))/2.
  c(1,i,j)=-tt*(UJp+gV(3,i,j)/Re)                                     !coefs of linear eqs
```

```
  c(2,i,j)= tt*(UJm-gV(3,i,j)/Re)
  c(3,i,j)=-tt*(VJp+gV(1,i,j)/Re)
  c(4,i,j)= tt*(VJm-gV(1,i,j)/Re)
  c(0,i,j)=aJ(2*i+1,2*j)-c(1,i,j)-c(2,i,j)-c(3,i,j)-c(4,i,j)
ENDDO; ENDDO
!Compute dV^* by modifies AF-method
DO j=1,j0; DO i=0,i0; k=i+1
  a(j,k,1)=c(1,i,j)
  a(j,k,2)=c(0,i,j)
  a(j,k,3)=c(2,i,j); b(j,k)=rhs(2,i,j)
ENDDO; ENDDO
CALL GAUSSZ(a,b,140,j0,if)
FORALL(j=1:j0,i=0:i0)rhs(2,i,j)=b(j,i+1)                                !dV^**
DO j=1,j0; DO i=0,i0; k=i+1
  a(k,j,1)=c(3,i,j)
  a(k,j,2)=c(0,i,j)
  a(k,j,3)=c(4,i,j); b(k,j)=c(0,i,j)*rhs(2,i,j)
ENDDO; ENDDO
CALL GAUSSZ(a,b,140,if,j0)
FORALL(j=1:j0,i=0:i0)rhs(2,i,j)=b(i+1,j)                                !dV^*
FORALL(j=0:j0,i=1:if)UJ(1,i,j)=UJ(1,i,j)+beta*aJ(2*i,2*j+1)*rhs(1,i,j) !JU^*
FORALL(j=1:j0,i=0:i0)UJ(2,i,j)=UJ(2,i,j)+beta*aJ(2*i+1,2*j)*rhs(2,i,j) !JV^*
FORALL(i=0:if,j=0:jf)f(i,j)=Uq(1,i,j)
lo=MAXLOC(f); FORALL(k=1:2)locf(k,1)=lo(k); fmax(1)=MAXVAL(f)
lo=MINLOC(f); FORALL(k=1:2)locf(k,2)=lo(k); fmax(2)=MINVAL(f)
FORALL(i=0:if,j=0:jf)f(i,j)=Uq(2,i,j)
lo=MAXLOC(f); FORALL(k=1:2)locf(k,3)=lo(k); fmax(3)=MAXVAL(f)
lo=MINLOC(f); FORALL(k=1:2)locf(k,4)=lo(k); fmax(4)=MINVAL(f)
ENDSUBROUTINE COMPU1

! ********** Set up coefficients of difference equations for phi
SUBROUTINE COEF(co)
PARAMETER(if=140,jf=25,if1=280,jf1=50)
DIMENSION co(0:if,0:jf,-1:1,-1:1),aJ(0:if1,0:jf1),gU(3,0:if,0:jf),gV(3,0:if,0:jf), &
    UJX(2,0:if,0:jf),w1(0:if,0:jf),gw(4,0:if,0:jf)
COMMON //na,Re,dt,i0,j0,lowRe,steady,unsteady
COMMON /METRIC2/aJ /METRIC3/gU,gV /COMPUZ1/UJX
! ***** Set up coefficients of difference-equations for phi
DO i=0,if; DO j=0,jf
  gw(1,i,j)= dt*gU(3,i,j)
  gw(2,i,j)=-dt*gU(2,i,j)/4.
  gw(3,i,j)=-dt*gV(2,i,j)/4.
  gw(4,i,j)= dt*gV(1,i,j)
ENDDO; ENDDO
FORALL(l=-1:1,m=-1:1,i=0:if,j=0:jf)co(i,j,l,m)=0.
DO i=0,i0; DO j=0,j0                                                    !1st step
  co(i,j,1, 1)= gw(2,i+1,j)
  co(i,j,0, 1)= gw(2,i+1,j)
  co(i,j,1, 0)= gw(1,i+1,j)
  co(i,j,0, 0)=-gw(1,i+1,j)
  co(i,j,1,-1)=-gw(2,i+1,j)
  co(i,j,0,-1)=-gw(2,i+1,j)
```

```
  ENDDO
  co(i, 0,1, 1)=co(i, 0,1, 1)+3.*gw(2,i+1, 0)                    !bottom
  co(i, 0,0, 1)=co(i, 0,0, 1)+3.*gw(2,i+1, 0)
  co(i, 0,1, 0)=co(i, 0,1, 0)-3.*gw(2,i+1, 0)
  co(i, 0,0, 0)=co(i, 0,0, 0)-3.*gw(2,i+1, 0)
  co(i,j0,1, 0)=co(i,j0,1, 0)+3.*gw(2,i+1,j0)                    !top
  co(i,j0,0, 0)=co(i,j0,0, 0)+3.*gw(2,i+1,j0)
  co(i,j0,1,-1)=co(i,j0,1,-1)-3.*gw(2,i+1,j0)
  co(i,j0,0,-1)=co(i,j0,0,-1)-3.*gw(2,i+1,j0)
ENDDO
DO i=1,i0; DO j=0,j0                                             !2nd step
  co(i,j, 0, 1)=co(i,j, 0, 1)-gw(2,i,j)
  co(i,j,-1, 1)=co(i,j,-1, 1)-gw(2,i,j)
  co(i,j, 0, 0)=co(i,j, 0, 0)-gw(1,i,j)
  co(i,j,-1, 0)=co(i,j,-1, 0)+gw(1,i,j)
  co(i,j, 0,-1)=co(i,j, 0,-1)+gw(2,i,j)
  co(i,j,-1,-1)=co(i,j,-1,-1)+gw(2,i,j)
  ENDDO
  co(i, 0, 0, 1)=co(i, 0, 0, 1)-3.*gw(2,i, 0)                    !bottom
  co(i, 0,-1, 1)=co(i, 0,-1, 1)-3.*gw(2,i, 0)
  co(i, 0, 0, 0)=co(i, 0, 0, 0)+3.*gw(2,i, 0)
  co(i, 0,-1, 0)=co(i, 0,-1, 0)+3.*gw(2,i, 0)
  co(i,j0, 0, 0)=co(i,j0, 0, 0)-3.*gw(2,i,j0)                    !top
  co(i,j0,-1, 0)=co(i,j0,-1, 0)-3.*gw(2,i,j0)
  co(i,j0, 0,-1)=co(i,j0, 0,-1)+3.*gw(2,i,j0)
  co(i,j0,-1,-1)=co(i,j0,-1,-1)+3.*gw(2,i,j0)
ENDDO
DO j=0,j0-1; DO i=0,i0                                           !3rd step
  co(i,j, 1,1)=co(i,j, 1,1)+gw(3,i,j+1)
  co(i,j, 1,0)=co(i,j, 1,0)+gw(3,i,j+1)
  co(i,j, 0,1)=co(i,j, 0,1)+gw(4,i,j+1)
  co(i,j, 0,0)=co(i,j, 0,0)-gw(4,i,j+1)
  co(i,j,-1,1)=co(i,j,-1,1)-gw(3,i,j+1)
  co(i,j,-1,0)=co(i,j,-1,0)-gw(3,i,j+1)
  ENDDO
  co(0,j,1,1)=co(0,j,1,1)+3.*gw(3,0,j+1)                         !inlet
  co(0,j,1,0)=co(0,j,1,0)+3.*gw(3,0,j+1)
  co(0,j,0,1)=co(0,j,0,1)-3.*gw(3,0,j+1)
  co(0,j,0,0)=co(0,j,0,0)-3.*gw(3,0,j+1)
ENDDO
DO j=1,j0; DO i=0,i0                                             !4th step
  co(i,j, 1, 0)=co(i,j, 1, 0)-gw(3,i,j)
  co(i,j, 1,-1)=co(i,j, 1,-1)-gw(3,i,j)
  co(i,j, 0, 0)=co(i,j, 0, 0)-gw(4,i,j)
  co(i,j, 0,-1)=co(i,j, 0,-1)+gw(4,i,j)
  co(i,j,-1, 0)=co(i,j,-1, 0)+gw(3,i,j)
  co(i,j,-1,-1)=co(i,j,-1,-1)+gw(3,i,j)
  ENDDO
  co(0,j,1, 0)=co(0,j,1, 0)-3.*gw(3,0,j)                         !inlet
  co(0,j,1,-1)=co(0,j,1,-1)-3.*gw(3,0,j)
  co(0,j,0, 0)=co(0,j,0, 0)+3.*gw(3,0,j)
  co(0,j,0,-1)=co(0,j,0,-1)+3.*gw(3,0,j)
```

```
ENDDO
ENDSUBROUTINE COEF

! ********** Compute static pressure p
SUBROUTINE COMPP(UJ,p,phi,co,resP,locf,fmax)
PARAMETER(if=140,jf=25,ife=70,jfe=13)                              !ife=(if+1)/2,jfe=(jf+1)/2
DIMENSION UJ(2,0:if,0:jf),p(0:if,0:jf),phi(0:if,0:jf),co(0:if,0:jf,-1:1,-1:1), &
    rhs(0:if,0:jf),a(ife,0:i0,3),b(ife,0:i0),r(ife),lo(2),locf(2,8),fmax(8)
COMMON //na,Re,dt,i0,j0,lowRe,steady,unsteady
REAL(8) a,b
LOGICAL lowRe,steady,unsteady
! Determine rhs of pressure difference eqn
FORALL(i=0:if,j=0:jf)phi(i,j)=0.
FORALL(i=0:i0,j=0:j0)rhs(i,j)=(UJ(1,i+1,j)-UJ(1,i,j)+UJ(2,i,j+1)-UJ(2,i,j)) !UJ=JU^*
!IF(steady.AND.lowRe)THEN
IF(steady)THEN
! ***** Solve pressure difference eqn by SOR method
alp=1.5; bet=0.5                                          !alp:over-relaxation, bet:damping
nf=4
n=0; 100 n=n+1; resP=0.
ib=0; ie=i0; id=1; jb=0; je=j0; jd=1                          !odd sweep, throughout region
IF(MOD(n,2)==0)THEN
  ib=i0-1; ie=1; id=-1; jb=j0-1; je=1; jd=-1                    !even sweep, inside region
ENDIF
DO i=ib,ie,id; DO j=jb,je,jd
  res=-rhs(i,j)
  DO ip=-1,1; ic=0; IF(i+ip==-1)ic=3
  DO jp=-1,1; jc=0; IF(j+jp==-1)jc=3; IF(j+jp==if)jc=-3
    res=res+co(i,j,ip,jp)*phi(i+ip+ic,j+jp+jc)                  !residuals
  ENDDO; ENDDO
  phi(i,j)=phi(i,j)-alp*res/co(i,j,0,0)                         !over-relaxation of phi
  resP=AMAX1(resP,ABS(res))                                     !max residual
ENDDO; ENDDO
IF(resP>1.E-5 .AND. n<nf)                  GOTO 100           !dicide convergence
ELSE
20 CONTINUE
! ***** Solve pressure difference eqn by Tschebyscheff SLOR method
alp=1.25; bet=0.75
ifo=if-ife; jfo=jf-jfe; nf=4
n=0; 110 n=n+1; resP=0.
! ***** Compute even i-columns
DO j=0,j0
  DO ii=1,ife; i=2*ii-2
    a(ii,j,1)=co(i,j,0,-1); a(ii,j,2)=co(i,j,0, 0)*alp; a(ii,j,3)=co(i,j,0, 1)
    b(ii,j)  =rhs(i,j)-(1.-alp)*co(i,j,0,0)*phi(i,j)
    r(ii)    =b(ii,j)
  ENDDO
  DO jp=-1,1; jc=0; IF(j+jp==-1)jc=3; IF(j+jp==jf)jc=-3
  DO ii=1,ife; i=2*ii-2
    im1=i-1; IF(im1==-1)im1=2
    cop=co(i,j,-1,jp)*phi(im1,j+jp+jc)+co(i,j,1,jp)*phi(i+1,j+jp+jc)
    b(ii,j)=b(ii,j)-cop
```

```
    r(ii)=r(ii)-cop-a(ii,j,jp+2)*phi(i,j+jp+jc)                        !residuals
  ENDDO; ENDDO
  DO ii=1,ife
    resP=AMAX1(resP,ABS(r(ii)))                                       !max residual
  ENDDO
ENDDO
CALL GAUSSP(a,b,ife,i0,j0,ife)
FORALL(j=0:j0,ii=1:ife)phi(2*ii-2,j)=b(ii,j)                          !even column phi
! ***** Compute odd i-columns
DO j=0,j0
  DO ii=1,ifo; i=2*ii-1
    a(ii,j,1)=co(i,j,0,-1); a(ii,j,2)=co(i,j,0, 0)*alp; a(ii,j,3)=co(i,j,0, 1)
    b(ii,j)  =rhs(i,j)-(1.-alp)*co(i,j,0,0)*phi(i,j)
    r(ii)    =b(ii,j)
  ENDDO
  DO jp=-1,1; jc=0; IF(j+jp==-1)jc=3; IF(j+jp==jf)jc=-3
  DO ii=1,ifo; i=2*ii-1
    cop=co(i,j,-1,jp)*phi(i-1,j+jp+jc)+co(i,j,1,jp)*phi(i+1,j+jp+jc)
    b(ii,j)=b(ii,j)-cop
    r(ii)=r(ii)-cop-a(ii,j,jp+2)*phi(i,j+jp+jc)                       !residuals
  ENDDO; ENDDO
  DO ii=1,ifo
    resP=AMAX1(resP,ABS(r(ii)))                                       !max residual
  ENDDO
ENDDO
CALL GAUSSP(a,b,ife,i0,j0,ifo)
FORALL(ii=1:ifo,j=0:j0)phi(2*ii-1,j)=b(ii,j)                          !odd column phi
! Compute even j-rows
n=n+1; resP=0.
DO i=0,i0
  DO jj=1,jfe; j=2*jj-2
    a(jj,i,1)=co(i,j,-1,0); a(jj,i,2)=co(i,j, 0,0)*alp; a(jj,i,3)=co(i,j, 1,0)
    b(jj,i)  =rhs(i,j)-(1.-alp)*co(i,j,0,0)*phi(i,j)
    r(jj)    =b(jj,i)
  ENDDO
  DO ip=-1,1; ic=0; IF(i+ip==-1)ic=3
  DO jj=1,jfe; j=2*jj-2
    jm1=j-1; IF(jm1==-1)jm1=2
    jp1=j+1; IF(jp1==jf)jp1=j0-2
    cop=co(i,j,ip,-1)*phi(i+ip+ic,jm1)+co(i,j,ip,1)*phi(i+ip+ic,jp1)
    b(jj,i)=b(jj,i)-cop
    r(jj)=r(jj)-cop-a(jj,i,ip+2)*phi(i+ip+ic,j)                       !residuals
  ENDDO; ENDDO
  DO jj=1,jfe
    resP=AMAX1(resP,ABS(r(jj)))                                       !max residual
  ENDDO
ENDDO
CALL GAUSSP(a,b,ife,i0,i0,jfe)
FORALL(i=0:i0,jj=1:jfe)phi(i,2*jj-2)=b(jj,i)                          !even row phi
! ***** Compute odd j-rows
DO i=0,i0
  DO jj=1,jfo; j=2*jj-1
```

```
    a(jj,i,1)=co(i,j,-1,0); a(jj,i,2)=co(i,j, 0,0)*alp; a(jj,i,3)=co(i,j, 1,0)
    b(jj,i)  =rhs(i,j)-(1.-alp)*co(i,j,0,0)*phi(i,j)
    r(jj)    =b(jj,i)
  ENDDO
  DO ip=-1,1; ic=0; IF(i+ip==-1)ic=3
  DO jj=1,jfo; j=2*jj-1
    jm1=j-1; IF(jm1==-1)jm1=2
    jp1=j+1; IF(jp1==jf)jp1=j0-2
    cop=co(i,j,ip,-1)*phi(i+ip+ic,jm1)+co(i,j,ip,1)*phi(i+ip+ic,jp1)
    b(jj,i)=b(jj,i)-cop
    r(jj)=r(jj)-cop-a(jj,i,ip+2)*phi(i+ip+ic,j)                        !residuals
  ENDDO; ENDDO
  DO jj=1,jfo
    resP=AMAX1(resP,ABS(r(jj)))                                        !max residual
  ENDDO
ENDDO
CALL GAUSSP(a,b,ife,i0,i0,jfo)
FORALL(jj=1:jfe,i=0:i0)phi(i,2*jj-1)=b(jj,i)                           !odd row phi
IF(resP>1.E-5 .AND. n<nf)                    GOTO 110
ENDIF
FORALL(i=0:i0,j=0:j0)p(i,j)=p(i,j)+bet*phi(i,j)                        !p=p+bet*phi
lo=MAXLOC(phi); FORALL(k=1:2)locf(k,5)=lo(k); fmax(5)=MAXVAL(phi)
lo=MINLOC(phi); FORALL(k=1:2)locf(k,6)=lo(k); fmax(6)=MINVAL(phi)
ENDSUBROUTINE COMPP

! ********** Compute volume flux JU and JV
SUBROUTINE COMPU2(UJ,phi)
PARAMETER(if=140,jf=25)
DIMENSION UJ(2,0:if,0:jf),phi(0:if,0:jf),gU(3,0:if,0:jf),gV(3,0:if,0:jf), &
          pV(0:if),dpV(0:if),pU(0:jf),dpU(0:jf)
COMMON //na,Re,dt,i0,j0,lowRe,steady,unsteady
COMMON /METRIC3/gU,gV
FORALL(j=0:j0)phi(if,j)=-phi(i0,j)
DO i=1,if                                                              !JU=JU^*-dt(g22phi_xi-g12phi_eta)
  FORALL(j=0:j0)pU(j)=(phi(i-1,j)+phi(i,j))/2.
  CALL DIFP(pU,dpU,jf,1.)
  FORALL(j=0:j0) &
    UJ(1,i,j)=UJ(1,i,j)-dt*(gU(3,i,j)*(phi(i,j)-phi(i-1,j))-gU(2,i,j)*dpU(j))
ENDDO
DO j=1,j0                                                              !JV=JV^*-dt(-g12phi_xi+g11phi_eta)
  FORALL(i=0:if)pV(i)=(phi(i,j-1)+phi(i,j))/2.
  CALL DIFP(pV,dpV,if,1.)
  i=i0; dpV(i)=(phi(i+1,j)-phi(i-1,j))/2.                              !outlet
  FORALL(i=0:i0) &
    UJ(2,i,j)=UJ(2,i,j)-dt*(-gV(2,i,j)*dpV(i)+gV(1,i,j)*(phi(i,j)-phi(i,j-1)))
ENDDO
ENDSUBROUTINE COMPU2

! ********** Compute velocities u and v and vorticity zeta
SUBROUTINE COMPUZ(uJ,u,z,locf,fmax,CFLmax)
PARAMETER(if=140,jf=25,if1=280,jf1=50)
DIMENSION UJ(2,0:if,0:jf),u(2,0:if,0:jf),z(0:if,0:jf),div(0:if,0:jf),UJX(2,0:if,0:jf), &
```

```
          aJ(0:if1,0:jf1),gU(3,0:if,0:jf),gV(3,0:if,0:jf),xix(4,0:if1,0:jf1),w1(0:if), &
          w2(0:jf),w11(0:if1),w12(0:jf1),lo(2),locf(2,8),fmax(8)
COMMON //na,Re,dt,i0,j0,lowRe,steady,unsteady
COMMON /COMPUZ1/UJX /METRIC2/aJ /METRIC3/gU,gV /METRIC4/xix
! ***** Compute velocities at X                                    !u= xix4*JUX-xix2*JVX
DO i=0,if                                                          !v=-xix3*JUX+xix1*JVX
  FORALL(j=0:jf)w2(j)=UJ(1,i,j)
  CALL INTUX(w2,w12,jf)
  FORALL(j=0:jf)UJX(1,i,j)=w12(2*j)
ENDDO
DO j=0,jf
  FORALL(i=0:if)w1(i)=UJ(2,i,j)
  CALL INTVX(w1,w11,if)
  FORALL(i=0:if)UJX(2,i,j)=w11(2*i)
ENDDO
FORALL(i=1:if,j=1:j0)u(1,i,j)= xix(4,2*i,2*j)*UJX(1,i,j)-xix(2,2*i,2*j)*UJX(2,i,j)
FORALL(i=1:if,j=1:j0)u(2,i,j)=-xix(3,2*i,2*j)*UJX(1,i,j)+xix(1,2*i,2*j)*UJX(2,i,j)
! ***** Compute vorticity zeta at X
FORALL(i=0:if,j=0:jf)z(i,j)=0.
DO j=1,j0
  FORALL(i=0:if)w1(i)=UJX(1,i,j)
  CALL INTP(w1,w11,if)                                             !w11=JU_V
  FORALL(i=0:i0)w1(i)=gV(2,i,j)*w11(2*i+1)+gV(3,i,j)*UJ(2,i,j)
  FORALL(i=1:i0)z(i,j)=w1(i)-w1(i-1)                               !*=(g21JU+g22JV)_,xi
  i=0 ; z(i,j)=-w1(i+2)+3.*w1(i+1)-2.*w1(i)                        !inlet bound
  i=if; z(i,j)= w1(i-3)-3.*w1(i-2)+2.*w1(i-1)                      !outlet bound
ENDDO
DO i=0,if
  FORALL(j=0:jf)w2(j)=UJX(2,i,j)
  CALL INTVP(w2,w12,jf)                                            !w12=JV_U
  FORALL(j=0:j0)w2(j)=gU(1,i,j)*UJ(1,i,j)+gU(2,i,j)*w12(2*j+1)
  FORALL(j=1:j0)z(i,j)=z(i,j)-(w2(j)-w2(j-1))                      !*=*-(g11JU+g12JV)_,eta
  j=0 ; z(i,j)=-(-w2(j+1)/3.+3.*w2(j)); z(40,0)=0.                 !bottom wall
  j=jf; z(i,j)=-( w2(j-2)/3.-3.*w2(j-1))                           !top wall
  FORALL(j=0:jf)z(i,j)=z(i,j)/aJ(2*i,2*j)                          !zeta=*/J
ENDDO
! ***** Divergence and max CFL number
CFLmax=0.
FORALL(i=0:i0,j=0:j0)div(i,j)=(UJ(1,i+1,j)-UJ(1,i,j) &
  +UJ(2,i,j+1)-UJ(2,i,j))/aJ(2*i+1,2*j+1)
lo=MAXLOC(div); FORALL(k=1:2)locf(k,7)=lo(k); fmax(7)=MAXVAL(div)
lo=MINLOC(div); FORALL(k=1:2)locf(k,8)=lo(k); fmax(8)=MINVAL(div)
DO i=0,i0; DO j=0,j0
  CFLmax=MAX(CFLmax,ABS(UJ(1,i,j)/aJ(2*i,2*j+1)),ABS(UJ(2,i,j)/aJ(2*i+1,2*j)))
ENDDO; ENDDO
CFLmax=CFLmax*dt                                                   !max|CFL|
ENDSUBROUTINE COMPUZ

! ********** Compute location of reattachment point
SUBROUTINE REATTACH(x,UJ,UJX,if,jf,xrap,k)
DIMENSION x(2,0:if,0:jf),UJ(2,0:if,0:jf),UJX(2,0:if,0:jf),al(-3:8)
k=100; irap=45; 10 irap=irap+1; IF(irap>105)RETURN
```

```
IF(UJ(1,irap,0)<0.)                                GOTO 10   !irap: grid point near rap
ii=-4; 12 ii=ii+1; i=irap+ii
du=UJ(1,i,0); d2u=UJX(1,i,1)-2.*du                           !du=DeltaJU, d2u=Delta^2JU
d3u=UJ(1,i,1)-3.*UJX(1,i,1)+3.*du                            !d3u=Delta^3JU
a=2.; 13 fa = du+(a-1.)/2.*(d2u+(a-2.)/3.*d3u)               !fa=f(alpha) by cubic interpolation
IF(ABS(fa)<.00001)                                 GOTO 14
dfa = d2u/2.+(2.*a-3.)/6.*d3u                                !dfa=f'(alpha)
a = a-fa/dfa;                                      GOTO 13   !by Newton method
14 al(ii)=a; IF(a>0.14)                            GOTO 12   !al(ii): alpha of u=0
IF(al(ii)<0.01)ii=ii-1                                       !irap+ii: adjacent grid point to rap
  ! using quadratic interpolation formula f(beta)=a0+da*beta+d2a*beta*beta/2.
  ! get beta from f(beta)=0 by Newton method
a2=al(ii-2); a1=al(ii-1); a0=al(ii)
da=(3.*a0-4.*a1+a2)/2.; d2a=a0-2.*a1+a2
k=0; b=0.; 22 k=k+1; fb = a0+da*b+d2a*b*b/2.                 !fb=f(beta)
IF(k>10)                                           GOTO 20
IF(ABS(fb)<0.01)                                   GOTO 21
dfb = da+d2a*b                                               !dfb=f'(beta)
b = b-fb/dfb;                                      GOTO 22   !by Newton method
21 xii=x(1,irap+ii,0); xi=x(1,irap+ii+1,0)
20 xrap = xii+b*(xi-xii)                                     !x of reattachment point
ENDSUBROUTINE REATTACH

! ********** minmod limiter
FUNCTION AMINMOD(a,b)
  s=SIGN(1.,a); AMINMOD=s*MAX(0.,MIN(ABS(a),s*b))
ENDFUNCTION

! ********** Compute x- or y-differences at point P (pressure)
SUBROUTINE DIFP(u,du,n,h)
DIMENSION u(0:n),du(0:n)
FORALL(i=1:n-2)du(i)=(u(i+1)-u(i-1))/(2.*h)
du(0)=(-3.*u(0)+4.*u(1)-u(2))/(2.*h)
du(n-1)=(u(n-3)-4.*u(n-2)+3.*u(n-1))/(2.*h)
ENDSUBROUTINE DIFP

! ********** Compute x- or y-differences at point X (metrics)
SUBROUTINE DIFX(u,du,n,h)
DIMENSION u(0:n),du(0:n)
FORALL(i=1:n-1)du(i)=(u(i+1)-u(i-1))/(2.*h)
du(0)=(-3.*u(0)+4.*u(1)-u(2))/(2.*h)
du(n)=(u(n-2)-4.*u(n-1)+3.*u(n))/(2.*h)
ENDSUBROUTINE DIFX

! ********** Solve in parallel, systems of linear eqns with modified tri-diagonal matrix
! by Gaussian eliminaton
SUBROUTINE GAUSSP(a,b,ife,i0,n,if)
DIMENSION a(ife,0:i0,3),b(ife,0:i0)
REAL(8)     a,b
FORALL(i=1:if)a(i,0,1)=a(i,0,1)/a(i,0,2)
FORALL(i=1:if)a(i,1,3)=a(i,1,3)-a(i,1,1)*a(i,0,1)
DO k=0,n-1; DO i=1,if
```

```
  b(i,k)=b(i,k)/a(i,k,2); a(i,k,3)=a(i,k,3)/a(i,k,2)
  b(i,k+1)=b(i,k+1)-a(i,k+1,1)*b(i,k)
  a(i,k+1,2)=a(i,k+1,2)-a(i,k+1,1)*a(i,k,3)
ENDDO; ENDDO
FORALL(i=1:if)a(i,n,1)=        -a(i,n,3)*a(i,n-2,3)
FORALL(i=1:if)a(i,n,2)=a(i,n,2)-a(i,n,1)*a(i,n-1,3)
FORALL(i=1:if)b(i,n)=(b(i,n)-a(i,n,3)*b(i,n-2)-a(i,n,1)*b(i,n-1))/a(i,n,2)
DO k=n-1,0,-1; DO i=1,if
  b(i,k)=b(i,k)-a(i,k,3)*b(i,k+1)
ENDDO; ENDDO
DO i=1,if; b(i,0)=b(i,0)-a(i,0,1)*b(i,2); ENDDO
ENDSUBROUTINE GAUSSP

! ********** Solve in paralell, systems of linear eqns with tri-diagonal matrix
! by Gaussian elimination
SUBROUTINE GAUSSZ(a,b,m1,m,n)
DIMENSION a(m1,m1,3),b(m1,m1)
DO k=1,n-1; DO l=1,m
  b(l,k)  =b(l,k)  /a(l,k,2)
  a(l,k,3)=a(l,k,3)/a(l,k,2)
  b(l,k+1)  =b(l,k+1)  -a(l,k+1,1)*b(l,k)
  a(l,k+1,2)=a(l,k+1,2)-a(l,k+1,1)*a(l,k,3)
ENDDO; ENDDO
FORALL(l=1:m)b(l,n)=b(l,n)/a(l,n,2)
DO k=n-1,1,-1
  FORALL(l=1:m)b(l,k)=b(l,k)-a(l,k,3)*b(l,k+1)
ENDDO
ENDSUBROUTINE GAUSSZ

! ********** Interpolate u1 from u (U_P from U)
SUBROUTINE INTP(u,u1,n)
DIMENSION u(0:n),u1(0:2*n)
FORALL(i=0:n)u1(2*i)=u(i)
FORALL(i=1:n-2)u1(2*i+1)=(-u(i-1)+9.*(u(i)+u(i+1))-u(i+2))/16.
i=0  ; u1(2*i+1)=(3.*u(i)+6.*u(i+1)-u(i+2))/8.
i=n-1; u1(2*i+1)=(3.*u(i+1)+6.*u(i)-u(i-1))/8.
ENDSUBROUTINE INTP

! ********** Interpolate U_X from U
SUBROUTINE INTUX(u,u1,jf)
DIMENSION u(0:jf),u1(0:2*jf)
FORALL(j=0:jf-1)u1(2*j+1)=u(j)
FORALL(j=2:jf-2)u1(2*j)=(-u(j-2)+9.*(u(j-1)+u(j))-u(j+1))/16.
j=0   ; u1(2*j)=0.
j=1   ; u1(2*j)=(3.*u(j-1)+2.*u(j)-u(j+1)/5.)/4.
j=jf-1; u1(2*j)=(3.*u(j)+2.*u(j-1)-u(j-2)/5.)/4.
j=jf  ; u1(2*j)=0.
ENDSUBROUTINE INTUX

! ********** Interpolate V_P from V
SUBROUTINE INTVP(u,u1,n)
DIMENSION u(0:n),u1(0:2*n)
```

```
FORALL(j=0:n)u1(2*j)=u(j)
FORALL(j=1:n-2)u1(2*j+1)=(-u(j-1)+9.*(u(j)+u(j+1))-u(j+2))/16.
j=0  ; u1(2*j+1)=(12.*u(j+1)-u(j+2))/32.
j=n-1; u1(2*j+1)=(12.*u(j  )-u(j-1))/32.
ENDSUBROUTINE INTVP

! ********** Interpolate V_X from V
SUBROUTINE INTVX(u,u1,if)
DIMENSION u(0:if),u1(0:2*if)
FORALL(i=0:if-1)u1(2*i+1)=u(i)
FORALL(i=2:if-2)u1(2*i)=(-u(i-2)+9.*(u(i-1)+u(i))-u(i+1))/16.
i=0   ; u1(2*i)=0.
i=1   ; u1(2*i)=(3.*u(i-1)+2.*u(i)-u(i+1)/5.)/4.
i=if-1; u1(2*i)=(3.*u(i)+6.*u(i-1)-u(i-2))/8.
i=if  ; u1(2*i)=(3.*u(i-1)-u(i-2))/2.
ENDSUBROUTINE INTVX

!**********  Print out computational results in flow field
SUBROUTINE WRTF(f,z1,z2,mz,na)
PARAMETER(if=140,jf=25)
DIMENSION f(0:if,0:jf)
CHARACTER*4 z1,z2,form(7),area(7)
DATA form/"(1H ","2X I","2,2X"," 36(","    ","F8.4","))  "/
DATA area/" -3P"," -2P"," -1P","    ","  1P","  2P","  3P"/
form(5)=area(mz+4)
OPEN(20,FILE='OUTPUT.dat')
WRITE(20,60)z1,z2,na
iif=(if-1)/35+1
DO ii=1,iif; is=35*(ii-1); ie=MIN(35*ii,if)
  DO j=jf,0,-1; WRITE(20,form)j,(f(i,j),i=is,ie); ENDDO
  WRITE(20,61)(i,i=is,ie)
ENDDO
60 FORMAT(/1H  20X "* * * ", 2A4, " * * *", 10X "na =", I5, /)
61 FORMAT(1H  4X 36I8//)
ENDSUBROUTINE WRTF

! ********** Drawing of Grid and Computational Results
SUBROUTINE StepDF_CG(x,u,p,z,if,jf)
USE DFLIB
DIMENSION x(2,0:if,0:jf),u(2,0:if,0:jf),p(0:if,0:jf),z(0:if,0:jf),pp(7:105,0:jf)
LOGICAL            statusmode
TYPE(windowconfig) myscreen
TYPE(wxycoord)     wxy
INTEGER(2)  status,xwidth,yheight,fontnum,numfonts
INTEGER(4)  oldcolor,colors(13)
REAL(4)     p0(0:if,0:jf),p1(0:if),p11(0:if),psi(0:if,0:jf),psik(13), &
            x1(0:70,0:25),y1(0:70,0:25),f1(0:70,0:25),                &
            x2(0:30,0:25),y2(0:30,0:25),f2(0:30,0:25),ak(13),Dps(13)
REAL(8)     xs,ys,xt,yt
CHARACTER*5 z1(7)
myscreen.numxpixels  = -1
myscreen.numypixels  = -1
```

```
myscreen.numtextcols = -1
myscreen.numtextrows = -1
myscreen.numcolors   = -1
myscreen.fontsize    = -1
myscreen.title       = " "C
statusmode = SETWINDOWCONFIG(myscreen)
IF(.NOT. statusmode) statusmode = SETWINDOWCONFIG(myscreen)
statusmode =GETWINDOWCONFIG(myscreen)
xwidth  = myscreen.numxpixels
yheight = myscreen.numypixels
oldcolor=SETBKCOLORRGB(#FFFFFF)
CALL CLEARSCREEN($GCLEARSCREEN)

! Set first window coordinates
CALL SETVIEWPORT(yheight/5.,INT2(0), yheight,yheight/5.)
status = SETWINDOW(.TRUE., -5.,-2., 11.,2.)
! Draw partial computational grid
oldcolor=SETCOLORRGB(#606060)                              !gray
DO i=0,if; xs=x(1,i,0); ys=x(2,i,0); CALL MOVETO_W(xs,ys,wxy)
  DO j=1,jf; xt=x(1,i,j); yt=x(2,i,j); status = LINETO_W(xt,yt); ENDDO
ENDDO
DO j=0,jf; xs=x(1,0,j); ys=x(2,0,j); CALL MOVETO_W(xs,ys,wxy)
  DO i=1,if; xt=x(1,i,j); yt=x(2,i,j); status = LINETO_W(xt,yt); ENDDO
ENDDO
oldcolor=SETCOLORRGB(#000000)
CALL CHARAC(1.0,-1.8,'COMPUTATIONAL GRID')

! Set second window coordinates
OPEN(20,FILE='OUTPUT.dat')
CALL SETVIEWPORT(yheight/5.,yheight/5., yheight,3.*yheight/5.)
status = SETWINDOW(.TRUE., -6.,-8., 18.,4.)
! Set colors
colors(1) =#E070A0; colors(2) =#FF7070                     !blue
colors(3) =#D0A070; colors(4) =#A0D070; colors(5) =#70FF70  !green
colors(6) =#60FFA0; colors(7) =#50FFD0; colors(8) =#40FFFF  !yellow
colors(9) =#50E0FF; colors(10)=#60C0FF; colors(11)=#7090FF
colors(12)=#7070FF                                         !red
colors(13)=#A070E0
status = REMAPALLPALETTERGB(colors)
! Draw pressure distribution
DO j=0,jf-1                                                        !interpolate p_X
  FORALL(i=0:if-1)p1(i)=p(i,j)
  CALL INTPX(p1,p11,if,if)
  FORALL(i=0:if)p0(i,j)=p11(i)
ENDDO
DO i=0,if
  FORALL(j=0:jf-1)p1(j)=p0(i,j)
  CALL INTPX(p1,p11,if,jf)
  FORALL(j=0:jf)p0(i,j)=p11(j)
ENDDO
DATA Dps/.005, .00667, .0075, .010, .0125, .015, .01667, .02, .025, &
         .03, .03333, .04, .05/                                !standard pressure increments
```

```
FORALL(i=7:105,j=0:jf)pp(i,j)=p0(i,j)
pmax=MAXVAL(pp); pmin=MINVAL(pp)
Dp=(pmax-pmin)/13.
IF(Dp<.004 .OR. Dp>.06)                          RETURN              !no suitable Dps
k=0; 10 k=k+1; IF(Dps(k)<Dp .AND. k<13)          GOTO 10             !determine Dps
kkk=k; Dpk=Dps(k); imin=INT(pmin/Dpk); IF(pmin>0.)imin=imin+1         !No of min pressure
ak(12)=Dpk*(FLOAT(imin)+11.5)
11 IF(ak(12)>=pmax)THEN; imin=imin-1; ak(12)=ak(12)-Dpk; GOTO 11     !determine imin
ENDIF
FORALL(k=1:12)ak(k)=Dpk*(FLOAT(imin+k)-.5)
FORALL(i=0:70,j=0:jf)
  x1(i,j)=x(1,i+6,j); y1(i,j)=x(2,i+6,j); f1(i,j)=p0(i+6,j); ENDFORALL
CALL DISTRIB(x1,y1,f1,70,jf,ak,colors,1,13)
WRITE(20,'(5X A6,F8.4,5X A6,F8.4)')'pmax =',pmax,'pmin =',pmin
! Draw velocity vectors
oldcolor=SETCOLORRGB(#000000)                                        !black
amag=.6
DO i=7,75,4; DO j=1,jf-1
  CALL LINE(x(1,i,j),x(2,i,j),x(1,i,j)+amag*u(1,i,j),x(2,i,j)+amag*u(2,i,j))
ENDDO; ENDDO
status = SETWINDOW(.TRUE., 18.,-3., 42.,9.)
! Draw pressure distribution
FORALL(i=0:30,j=0:jf)
  x2(i,j)=x(1,i+75,j); y2(i,j)=x(2,i+75,j); f2(i,j)=p0(i+75,j); ENDFORALL
CALL DISTRIB(x2,y2,f2,30,jf,ak,colors,1,13)
! Draw velocity vectors
oldcolor=SETCOLORRGB(#000000)
DO i=75,99,4; DO j=1,jf-1
  CALL LINE(x(1,i,j),x(2,i,j),x(1,i,j)+amag*u(1,i,j),x(2,i,j)+amag*u(2,i,j))
ENDDO; ENDDO
! Color-pressure relation
CALL COLORVAL(39.0,-0.95,39.3,2.95,colors,13,kkk,Dpk,imin)
oldcolor=SETCOLORRGB(#404040)
CALL CHARAC(23.0,-2.3,'PRESSURE DISTRIBUTION AND VELOCITY VECTORS')

! Set third window coordinates
CALL SETVIEWPORT(yheight/5.,3.*yheight/5., yheight,yheight)
status = SETWINDOW(.TRUE., -6.,-8., 18.,4.)
! Draw vorticity distribution
FORALL(i=0:70,j=0:jf)
  x1(i,j)=x(1,i+6,j); y1(i,j)=x(2,i+6,j); f1(i,j)=z(i+6,j); ENDFORALL
FORALL(k=1:13)ak(k)=-2.6+.4*FLOAT(k)          !paint colors(k) between ak(k-1) and ak(k)
CALL DISTRIB(x1,y1,f1,70,jf,ak,colors,1,13)
! Draw streamlines
DO i=0,if
  psi(i,0)=0.
  DO j=1,jf
    psi(i,j)=psi(i,j-1)+(u(1,i,j-1)+u(1,i,j))/2.*(x(2,i,j)-x(2,i,j-1)) &
                       -(u(2,i,j-1)+u(2,i,j))/2.*(x(1,i,j)-x(1,i,j-1))
  ENDDO
  dpsi=3.-psi(i,jf)
  FORALL(j=1:jf)psi(i,j)=psi(i,j)+dpsi*FLOAT(j)/jf
```

```
ENDDO
FORALL(i=0:70,j=0:jf)f1(i,j)=psi(i+6,j)
oldcolor=SETCOLORRGB(#606060)                                  !gray
FORALL(k=1:13)psik(k)=.25*FLOAT(k-1)
CALL CONTOUR(x1,y1,f1,70,jf,psik,13)
oldcolor=SETCOLORRGB(#808080)                                  !light gray
FORALL(k=1:13)psik(k)=-.01*FLOAT(k)
CALL CONTOUR(x1,y1,f1,70,jf,psik,13)
oldcolor=SETCOLORRGB(#606060)                                  !gray
CALL LINE(-6.0,0.0,0.0,0.0)  ; CALL LINE(0.0,0.0,0.0,-1.0)
CALL LINE(0.0,-1.0,18.0,-1.0); CALL LINE(-6.0,3.0,18.0,3.0)
status = SETWINDOW(.TRUE., 18.,-3.1, 42.,9)
! Draw vorticity distribution
FORALL(i=0:30,j=0:jf)
  x2(i,j)=x(1,i+75,j); y2(i,j)=x(2,i+75,j); f2(i,j)=z(i+75,j); ENDFORALL
CALL DISTRIB(x2,y2,f2,30,jf,ak,colors,1,13)
! Draw streamlines
FORALL(i=0:30,j=0:jf)f2(i,j)=psi(i+75,j)
oldcolor=SETCOLORRGB(#606060)                                  !gray
FORALL(k=1:13)psik(k)=.25*FLOAT(k-1)
CALL CONTOUR(x2,y2,f2,30,jf,psik,13)
oldcolor=SETCOLORRGB(#808080)                                  !light gray
FORALL(k=1:13)psik(k)=-.01*FLOAT(k)
CALL CONTOUR(x2,y2,f2,30,jf,psik,13)
oldcolor=SETCOLORRGB(#606060)                                  !gray
CALL LINE(18.0,-1.0,42.0,-1.0); CALL LINE(18.0,3.0,42.0,3.0)
! Color-vorticity relation
xs=39.0; xt=39.3
DO k=1,13
  oldcolor=SETCOLORRGB(colors(k))
  ys=-0.95+0.3*FLOAT(k); yt=-0.95+0.3*FLOAT(k-1)
  status=RECTANGLE_W($GFILLINTERIOR,xs,ys,xt,yt)
ENDDO
oldcolor=SETCOLORRGB(#000000)
ys=2.95; yt=-.95; status = RECTANGLE_W($GBORDER,xs,ys,xt,yt)
numfonts=INITIALIZEFONTS()
fontnum =SETFONT('t''Arial''h14w6')
DATA z1/'-2.4 ','-1.6 ','-0.8 ',' 0.0 ',' 0.8 ',' 1.6 ',' 2.4 '/
DO k=1,7; xt=39.4; yt=-0.55+0.6*FLOAT(k-1)
  CALL MOVETO_W(xt,yt,wxy); CALL OUTGTEXT(z1(k))
ENDDO
CALL CHARAC(24.0,-2.3,'VORTICITY DISTRIBUTION AND STREAMLINES')
ENDSUBROUTINE StepDF_CG

! ********** character
SUBROUTINE CHARAC(x,y,A)
USE DFLIB
TYPE (windowconfig) myscreen
TYPE (wxycoord)     wxy
INTEGER(2)          numfonts,fontnum
REAL(8)             xt,yt
CHARACTER*100       A
```

```
numfonts=INITIALIZEFONTS()
fontnum=SETFONT("t'Arial''h16w6'")
xt=x; yt=y+0.33; CALL MOVETO_W(xt,yt,wxy); CALL OUTGTEXT(A)
ENDSUBROUTINE CHARAC

! ********** Values of colors
SUBROUTINE COLORVAL(x1,y1,x2,y2,colors,kf,kkk,Dpk,imin)
USE DFLIB
TYPE(wxycoord) wxy
INTEGER*2   status,numfonts,fontnum,intv(13)
INTEGER*4   oldcolor,colors(kf)
REAL*8      xs,ys,xt,yt
CHARACTER*5 z1(-20:50)
dy=(y2-y1)/FLOAT(kf)
xs=x1; xt=x2
DO k=1,kf
  oldcolor=SETCOLORRGB(colors(k))
  ys=y1+dy*FLOAT(k); yt=y1+dy*FLOAT(k-1)
  status = RECTANGLE_W($GFILLINTERIOR,xs,ys,xt,yt)
ENDDO
oldcolor=SETCOLORRGB(#000000)
ys=y2; yt=y1; status = RECTANGLE_W($GBORDER,xs,ys,xt,yt)
numfonts=INITIALIZEFONTS()
fontnum =SETFONT('t''Arial''h14w6')
DATA z1/'-0.20','-0.19','-0.18','-0.17','-0.16','-0.15','-0.14','-0.13','-0.12','-0.11', &
        '-0.10','-0.09','-0.08','-0.07','-0.06','-0.05','-0.04','-0.03','-0.02','-0.01', &
' 0.00',' 0.01',' 0.02',' 0.03',' 0.04',' 0.05',' 0.06',' 0.07',' 0.08',' 0.09', &
' 0.10',' 0.11',' 0.12',' 0.13',' 0.14',' 0.15',' 0.16',' 0.17',' 0.18',' 0.19', &
' 0.20',' 0.21',' 0.22',' 0.23',' 0.24',' 0.25',' 0.26',' 0.27',' 0.28',' 0.29', &
' 0.30',' 0.31',' 0.32',' 0.33',' 0.34',' 0.35',' 0.36',' 0.37',' 0.38',' 0.39', &
' 0.40',' 0.41',' 0.42',' 0.43',' 0.44',' 0.45',' 0.46',' 0.47',' 0.48',' 0.49', &
' 0.50'/                                        !pressure value of each color band
DATA intv/2, 3, 4, 2, 4, 2, 3, 3*2, 3, 2*2/                          !text intervals
intvk=intv(kkk)
IF(imin>=0)THEN; itextmin=((imin+intvk-1)/intvk)*intvk
ELSE           ; itextmin=( imin         /intvk)*intvk; ENDIF         !No of bottom text
l=0; 12 l=l+1
xt=x2+0.1; yt=y1+dy*(itextmin-imin+intvk*(l-1))+0.4                   !coordinates of character
itext=INT((itextmin+intvk*(l-1))*(Dpk+.0001)*100.)                   !No of press character
CALL MOVETO_W(xt,yt,wxy); CALL OUTGTEXT(z1(itext))
IF(itextmin-imin+intvk*l<=12)                    GOTO 12
ENDSUBROUTINE COLORVAL

! ********** Draw contours of f with oldcolor
SUBROUTINE CONTOUR(x,y,f0,if,jf,fk,kf)
USE DFLIB
DIMENSION x(0:if,0:jf),y(0:if,0:jf),f0(0:if,0:jf),fk(kf)
TYPE(wxycoord) wxy
INTEGER(2)     status
REAL(4)        f(0:if,0:jf)
REAL(8)        xs,ys
e=.00001
```

```
DO k=1,kf
  DO i=0,if; DO j=0,jf
    f(i,j)=f0(i,j)-fk(k)
    IF(ABS(f(i,j))<e) f(i,j)=SIGN(e,f(i,j))
  ENDDO; ENDDO
  DO 10 i=0,if-1; DO 10 j=0,jf-1; imoveto=0              !for all elements
    i1=i;   j1=j;   x1=x(i1,j1); y1=y(i1,j1); f1=f(i1,j1)
    i2=i+1; j2=j;   x2=x(i2,j2); y2=y(i2,j2); f2=f(i2,j2)
    i3=i;   j3=j+1; x3=x(i3,j3); y3=y(i3,j3); f3=f(i3,j3)
    i4=i+1; j4=j+1; x4=x(i4,j4); y4=y(i4,j4); f4=f(i4,j4)
    IF(f1*f2<0)THEN
      xs=(f1*x2-f2*x1)/(f1-f2); ys=(f1*y2-f2*y1)/(f1-f2)    !starting point
      CALL MOVETO_W(xs,ys,wxy); imoveto=1
    ENDIF
    IF(f3*f4<0)THEN
      xs=(f3*x4-f4*x3)/(f3-f4); ys=(f3*y4-f4*y3)/(f3-f4)
      IF(imoveto==1)THEN
        status = LINETO_W(xs,ys);     imoveto=0
      ELSE; CALL MOVETO_W(xs,ys,wxy); imoveto=1
      ENDIF
    ENDIF
    IF(f1*f3<0)THEN
      xs=(f1*x3-f3*x1)/(f1-f3); ys=(f1*y3-f3*y1)/(f1-f3)
      IF(imoveto==1)THEN
        status = LINETO_W(xs,ys);     imoveto=0;  GOTO 10
      ELSE; CALL MOVETO_W(xs,ys,wxy); imoveto=1
      ENDIF
    ENDIF
    IF(imoveto==1)THEN
      xs=(f2*x4-f4*x2)/(f2-f4); ys=(f2*y4-f4*y2)/(f2-f4)
      status = LINETO_W(xs,ys)
    ENDIF
  10 CONTINUE
ENDDO
ENDSUBROUTINE CONTOUR

! ********** Draw distribution of f with colors
SUBROUTINE DISTRIB(x,y,f,if,jf,ak,colors,kb,kf)
USE DFLIB
DIMENSION  x(0:if,0:jf),y(0:if,0:jf),f(0:if,0:jf),ak(kb:kf),colors(kb:kf)
TYPE(wxycoord)     wxy, poly(6)
INTEGER(2) status,m(4)
INTEGER(4) oldcolor,colors
REAL(4)    fe(4)
REAL(8)    xs,ys,xt,yt,xs0,ys0,xt0,yt0,xs1,ys1,xt1,yt1,wx,wy
jp=jf+1; ak(kf)=1.E10
DO i=0,if-1; DO j=0,jf-1; m1=i*jp+j+1                    !for all quadrilateral elements
  m(1)=m1;      fe(1)=f(i,j)
  m(2)=m1+jp;   fe(2)=f(i+1,j)
  m(3)=m1+1;    fe(3)=f(i,j+1)
  m(4)=m1+jp+1; fe(4)=f(i+1,j+1)
! m() and fe() are four node numbers and function values of each element, respectively.
```

```
! l1 l2 l3 or l4 is one of four element node numbers 1 2 3 4, and the fe-values increase in
! order. First locate max and min fe-value nodes l1 and l4, then determine middle fe-value
! nodes l2 and l3.
  l1=MINLOC(fe,DIM=1); l4=MAXLOC(fe,DIM=1)
  sum=FLOAT(10-l1-l4); prd=FLOAT(24/l1/l4); sqr=SQRT(sum*sum-4.*prd)
  l2=INT((sum-sqr)/2.+.1); l3=INT((sum+sqr)/2.+.1)
  IF(fe(l2)>fe(l3))THEN; l=l2; l2=l3; l3=l; ENDIF
! a1<=a2<=a3<=a4 are nodal values of function f.
  i1=(m(l1)-1)/jp; j1=MOD(m(l1)-1,jp); x1=x(i1,j1); y1=y(i1,j1); a1=f(i1,j1)
  i2=(m(l2)-1)/jp; j2=MOD(m(l2)-1,jp); x2=x(i2,j2); y2=y(i2,j2); a2=f(i2,j2)
  i3=(m(l3)-1)/jp; j3=MOD(m(l3)-1,jp); x3=x(i3,j3); y3=y(i3,j3); a3=f(i3,j3)
  i4=(m(l4)-1)/jp; j4=MOD(m(l4)-1,jp); x4=x(i4,j4); y4=y(i4,j4); a4=f(i4,j4)
  k1=kb-1; 11 k1=k1+1; IF(ak(k1)<a1)                GOTO 11
  k2=kb-1; 12 k2=k2+1; IF(ak(k2)<a2)                GOTO 12
  k3=kb-1; 13 k3=k3+1; IF(ak(k3)<a3)                GOTO 13
  k4=kb-1; 14 k4=k4+1; IF(ak(k4)<a4)                GOTO 14
! ak(k1)<=ak(k2)<=ak(k3)<=ak(k4)  Small range ak(k-1)-ak(k) is painted by colors(k).
! i.e. -infty-ak(kb) by colors(kb), ak(kf-1)-ak(kf):infty by colors(kf).
! k1<=k2<=k3<=k4. ntype shows arrangement of k1 k2 k3 k4,
!   ntype=1: k1-k2-k4-k3-k1
!   ntype=2: k1-k2-k3-k4-k1,
!   ntype=3: k1-k3-k2-k4-k1(saddle point).
  ntype=1
  LA = i3==i1.OR.j3==j1; IF(LA==.FALSE.)ntype=2
  LB = i2==i1.OR.j2==j1; IF(LB==.FALSE.)ntype=3
! xs ys and xt yt are coordinates of points on edge of quadrilateral element, whose function
! value is a. The coordinates are determined by linear interpolation.
  DO 20 k=k1,k4; a=ak(k)
    IF(ntype==3)THEN; nvertx=4                              !temporary
      poly(1)%wx=x1; poly(1)%wy=y1
      poly(2)%wx=x3; poly(2)%wy=y3
      poly(3)%wx=x2; poly(3)%wy=y2
      poly(4)%wx=x4; poly(4)%wy=y4;             GOTO 21
    ENDIF
    IF(k==k4)                                   GOTO 22
    IF(ntype==1)THEN                                        !ntype=1
      IF(k<k3)THEN; xs=(a1-a)/(a1-a3)*x3+(a-a3)/(a1-a3)*x1  !k1<=k<k3
                    ys=(a1-a)/(a1-a3)*y3+(a-a3)/(a1-a3)*y1
      ELSE;         xs=(a3-a)/(a3-a4)*x4+(a-a4)/(a3-a4)*x3  !k3<=k<k4
                    ys=(a3-a)/(a3-a4)*y4+(a-a4)/(a3-a4)*y3
      ENDIF
      IF(k<k2)THEN; xt=(a1-a)/(a1-a2)*x2+(a-a2)/(a1-a2)*x1  !k1<=k<k2
                    yt=(a1-a)/(a1-a2)*y2+(a-a2)/(a1-a2)*y1
      ELSE;         xt=(a2-a)/(a2-a4)*x4+(a-a4)/(a2-a4)*x2  !k2<=k<k4
                    yt=(a2-a)/(a2-a4)*y4+(a-a4)/(a2-a4)*y2
      ENDIF
    ELSEIF(ntype==2)THEN                                    !ntype=2
                    xs=(a1-a)/(a1-a4)*x4+(a-a4)/(a1-a4)*x1  !k1<=k<k4
                    ys=(a1-a)/(a1-a4)*y4+(a-a4)/(a1-a4)*y1
      IF(k<k2)THEN; xt=(a1-a)/(a1-a2)*x2+(a-a2)/(a1-a2)*x1  !k1<=k<k2
                    yt=(a1-a)/(a1-a2)*y2+(a-a2)/(a1-a2)*y1
      ELSEIF(k<k3)THEN
```

```
                  xt=(a2-a)/(a2-a3)*x3+(a-a3)/(a2-a3)*x2  !k2<=k<k3
                  yt=(a2-a)/(a2-a3)*y3+(a-a3)/(a2-a3)*y2
      ELSE;       xt=(a3-a)/(a3-a4)*x4+(a-a4)/(a3-a4)*x3  !k3<=k<k4
                  yt=(a3-a)/(a3-a4)*y4+(a-a4)/(a3-a4)*y3
      ENDIF
    ELSE;         xs=(a1-a)/(a1-a4)*x4+(a-a4)/(a1-a4)*x1  !ntype=3
                  ys=(a1-a)/(a1-a4)*y4+(a-a4)/(a1-a4)*y1
      IF(k2<k3.AND.k2<=k.AND.k<=k3)THEN
                  xs1=(a2-a)/(a2-a3)*x3+(a-a3)/(a2-a3)*x2
                  ys1=(a2-a)/(a2-a3)*y3+(a-a3)/(a2-a3)*y2
      ENDIF
      IF(k<k3)THEN; xt=(a1-a)/(a1-a3)*x3+(a-a3)/(a1-a3)*x1
                  yt=(a1-a)/(a1-a3)*y3+(a-a3)/(a1-a3)*y1
      ENDIF
      IF(k2<=k)THEN; xt1=(a2-a)/(a2-a4)*x4+(a-a4)/(a2-a4)*x2
                   yt1=(a2-a)/(a2-a4)*y4+(a-a4)/(a2-a4)*y2
      ENDIF
    ENDIF
    22 CONTINUE
! Divide into some polygons quadrilateral element in accordance with stepped function a(k).
! poly() are coordinates of vertices of polygon, and nvertx is number of the vertices.
! Coloring is performed in order k1...k4 in each element, but the following program written
! in order k1 k4 k1<k<k4.
    IF(k==k1)THEN                                          !k==k1
      poly(1)%wx=x1; poly(1)%wy=y1
      poly(2)%wx=xs; poly(2)%wy=ys
      poly(3)%wx=xt; poly(3)%wy=yt; nvertx=3               !k==k1<k2
      IF(k==k2)THEN
        poly(4)%wx=x2; poly(4)%wy=y2; nvertx=4             !k==k1==k2<k3
      ENDIF
      IF(k==k4)THEN                                        !k==k1==k2==k3==k4
        IF(ntype==1)THEN
          poly(2)%wx=x3; poly(2)%wy=y3
          poly(3)%wx=x4; poly(3)%wy=y4
        ENDIF
        IF(ntype==2)THEN
          poly(2)%wx=x4; poly(2)%wy=y4
          poly(3)%wx=x3; poly(3)%wy=y3
        ENDIF
      ELSEIF(k==k3)THEN                                    !k==k1==k2==k3<k4
        poly(5)%wx=x2; poly(5)%wy=y2; nvertx=5
        IF(ntype==1)THEN
          poly(2)%wx=x3; poly(2)%wy=y3
          poly(3)%wx=xs; poly(3)%wy=ys
          poly(4)%wx=xt; poly(4)%wy=yt
        ENDIF
        IF(ntype==2)THEN
          poly(4)%wx=x3; poly(4)%wy=y3
      ENDIF; ENDIF
    ELSEIF(k==k4)THEN                                      !k1<k==k4
      poly(1)%wx=xt0; poly(1)%wy=yt0
      poly(2)%wx=xs0; poly(2)%wy=ys0
```

```
      poly(3)%wx=x4;  poly(3)%wy=y4; nvertx=3              !k3<k==k4
      IF(k==k3)THEN;                 nvertx=4              !k2<k==k3==k4
        IF(ntype==1)THEN
          poly(3)%wx=x3; poly(3)%wy=y3
          poly(4)%wx=x4; poly(4)%wy=y4
        ENDIF
        IF(ntype==2)THEN
          poly(4)%wx=x3; poly(4)%wy=y3
      ENDIF; ENDIF
      IF(k==k2)THEN                                        !k1<k==k2==k3==k4
        poly(5)%wx=x2; poly(5)%wy=y2; nvertx=5
      ENDIF
    ELSE                                                   !k1<k<k4
      poly(1)%wx=xt0; poly(1)%wy=yt0
      poly(2)%wx=xs0; poly(2)%wy=ys0
      poly(3)%wx=xs;  poly(3)%wy=ys
      poly(4)%wx=xt;  poly(4)%wy=yt; nvertx=4              !k/=k2,k3
      IF(k==k2.AND.k<k3)THEN
        poly(5)%wx=x2; poly(5)%wy=y2; nvertx=5             !k1<k==k2<k3
      ENDIF
      IF(k==k3)THEN;                 nvertx=5              !k2<k==k3<k4
        IF(ntype==1)THEN
          poly(3)%wx=x3; poly(3)%wy=y3
          poly(4)%wx=xs; poly(4)%wy=ys
          poly(5)%wx=xt; poly(5)%wy=yt
        ENDIF
        IF(ntype==2)THEN
          poly(5)%wx=x3; poly(5)%wy=y3
        ENDIF
        IF(k==k2)THEN                                      !k1<k==k2==k3<k4
          poly(6)%wx=x2; poly(6)%wy=y2; nvertx=6
      ENDIF; ENDIF
    ENDIF
    IF(k<k4)THEN
      xs0=xs; ys0=ys; xt0=xt; yt0=yt
    ENDIF
    21 oldcolor=SETCOLORRGB(colors(k))
    status=POLYGON_W($GFILLINTERIOR,poly,nvertx)
  20 CONTINUE
ENDDO; ENDDO
ENDSUBROUTINE DISTRIB

! ********** Interpolate values at U from values at P
SUBROUTINE INTPX(u,u1,if,n)
DIMENSION u(0:if),u1(0:if)
FORALL(i=2:n-2)u1(i)=(-u(i-2)+9.*(u(i-1)+u(i))-u(i+1))/16.
i=0  ; u1(i)=(3.*u(i)-u(i+1))/2.
i=1  ; u1(i)=(3.*u(i-1)+6.*u(i)-u(i+1))/8.
i=n-1; u1(i)=(3.*u(i)+6.*u(i-1)-u(i-2))/8.
i=n  ; u1(i)=(3.*u(i-1)-u(i-2))/2.
ENDSUBROUTINE INTPX
```

```
! ********** Draw line
SUBROUTINE LINE(x1,y1,x2,y2)
USE DFLIB
TYPE (wxycoord) wxy
INTEGER(2)      status
REAL(8)         xs,ys,xt,yt
xs=x1; ys=y1; xt=x2; yt=y2; CALL MOVETO_W(xs,ys,wxy); status = LINETO_W(xt,yt)
ENDSUBROUTINE LINE
```

次にこのプログラムのおもなサブルーチンを説明する．サブルーチン GRID は，楕円型微分方程式の境界値問題を解いて曲線座標格子を形成する解析的格子形成法 (analytical grid generation) のプログラムである．計算はすべて写像面上の単位正方形格子に対して行われる．バックステップ流路は写像面上の長方形領域 $0 \le \xi \le 140,\ 0 \le \eta \le 25$ に写像される．格子間隔 $\Delta\xi = \Delta\eta = 1$．上流境界 $(x = -10)$ の格子点番号は $i = 0$，下流境界 $(x = 70)$ は $i_f = 140$，ステップ $(x = 0)$ の角は $i_1 = 35$，隅は $i_2 = 40$ である．また下方壁面 $(y = 0\,(x \le 0),\ x = 0\,(-1 < y < 0),\ y = -1\,(x \ge 0))$ の格子点番号は $j = 0$，上方壁面 $(y = 3)$ は $j_f = 25$ である．

曲線座標格子は基本的に次の Poisson 方程式の境界値問題を解くことによって形成される．

$$\xi_{xx} + \xi_{yy} = P, \qquad \eta_{xx} + \eta_{yy} = Q \tag{14.24}$$

ただし $P, Q$ は制御関数と呼ばれ主に格子間隔の制御に使われる．しかしながらこの方程式は，独立変数が $xy$ で $\xi\eta$ 面上での計算に適しない．格子形成の計算には上式を書換えた次式が用いられる．

$$L(x) = -J(x_\xi P + x_\eta Q) \tag{14.25a}$$

$$L(y) = -J(y_\xi P + y_\eta Q) \tag{14.25b}$$

ただし $L$ は微分演算子で

$$L = \tilde{g}_{22}\frac{\partial^2}{\partial\xi^2} - 2\tilde{g}_{12}\frac{\partial^2}{\partial\xi\partial\eta} + \tilde{g}_{11}\frac{\partial^2}{\partial\eta^2}$$

$\tilde{g}_{ij} = g_{ij}/J$，ヤコビアン $J$ と測度テンソル $g_{ij}$ は前節で定義されたものである[7]．

式 (14.25) の境界値問題の境界条件は次のように与えられる．下方境界には境界点の座標が等比級数によって，また上方境界点の $y = 3$，上流境界点の $x = -10$，下流境界点の $x = 70$ が与えられる．上方境界点の $x$ の値は適当に，また内点の $xy$ の値はとりあえず $\boldsymbol{x}_{ij} = (1-\alpha_j)\boldsymbol{x}_{i0} + \alpha_j\boldsymbol{x}_{ij_f}$ によって補間される．ただし $\alpha(\tilde{\eta}) = -\tilde{\eta}^3 + 1.6\,\tilde{\eta}^2 + 0.4\,\tilde{\eta},\ \tilde{\eta} = \eta/\eta_f$ である．なおこの 3 次式は条件 $\alpha(0) = 0,\ \alpha(1) = 1,\ \alpha'(0) = 0.4,\ \alpha'(1) = 0.6$ を満足するように作られたものである．

[7] 自明の関係 $x = x(\xi(x,\,y),\,\eta(x,\,y))$ を $x$ で 2 階微分した式と $y$ で 2 階微分した式

$$\xi_x^2 x_{\xi\xi} + 2\xi_x\eta_x x_{\xi\eta} + \eta_x^2 x_{\eta\eta} + \xi_{xx}x_\xi + \eta_{xx}x_\eta = 0,$$
$$\xi_y^2 x_{\xi\xi} + 2\xi_y\eta_y x_{\xi\eta} + \eta_y^2 x_{\eta\eta} + \xi_{yy}x_\xi + \eta_{yy}x_\eta = 0$$

を加え，式 (14.6) と (14.24) を用いれば次式が得られる．

$$g^{11}x_{\xi\xi} + 2g^{12}x_{\xi\eta} + g^{22}x_{\eta\eta} + x_\xi P + x_\eta Q = 0$$

この式にヤコビアン $J$ をかけ，$Jg^{11} = \tilde{g}_{22},\ Jg^{12} = -\tilde{g}_{12},\ Jg^{22} = \tilde{g}_{11}$ なる関係を用いれば，式 (14.25a) を導出できる．同様に式 $y = y(\xi(x,\,y),\,\eta(x,\,y))$ から始めれば式 (14.25b) を導出できる．

制御関数 $P$ は，ここでは下方境界上に与えられた格子間隔が領域内部まで適当に浸透するように次式によって与えられる．

$$P_{ij} = P_{i0} \quad \text{or} \quad = \frac{\eta_t - \eta}{\eta_t} P_{i0}$$
$$P_{i0} = (\xi_{xx})_{i0} = \frac{2}{x_{i+1,0} - x_{i-1,0}} \Big( \frac{1}{x_{i+1,0} - x_{i0}} - \frac{1}{x_{i0} - x_{i-1,0}} \Big)$$

また制御関数 $Q$ は，$j$ 方向の格子間隔が流路中央で粗く境界とりわけ下方境界で細かくなるように次式によって与えられる．

$$Q_{ij} = (\eta_{yy})_{ij} = \eta''(y)_{ij} = \Big( \frac{d\tilde{y}}{dy} \Big)^2_{ij} \tilde{\eta}''(\tilde{y})_j \frac{d\eta}{d\tilde{\eta}}$$
$$\Big( \frac{d\tilde{y}}{dy} \Big)^2_{ij} \frac{d\eta}{d\tilde{\eta}} = \frac{25}{(y_{ij_f} - y_{i0})^2}, \qquad \tilde{\eta}''(\tilde{y}) = 8\tilde{y} - 5$$

ただし下の行の第 1 式は $\tilde{\eta} = \eta/25,\ \tilde{y} = (y - y_0)/(y_{j_f} - y_0)$ なる関係から求められる倍率，また第 2 式は 3 次式 $\tilde{\eta}(\tilde{y}) = (8\tilde{y}^3 - 15\tilde{y}^2 + 13\tilde{y})/6,\ (0 \le \tilde{y} \le 1)$ の 2 階微分である．この 3 次式は条件 $\tilde{\eta}(0) = 0,\ \tilde{\eta}(1) = 1,\ \tilde{\eta}'(0) = 13/6,\ \tilde{\eta}'(1) = 7/6$ を満足するもので，$\eta$ 方向の格子間隔の分布はこの式によって規定される．なおプログラムには，格子間隔の差の小さいものから大きいものまでいくつかの式が用意されており，その中から好みのものを選ぶことができる．

式 (14.25) の差分方程式は $xy$ の格子点の値 $x_{ij}$ または $y_{ij}$ の連立 1 次方程式で，SOR 法で容易に解くことができる．ただしこれらの差分方程式の係数と右辺には解 $xy$ が含まれるので，これらの係数と右辺の値を更新し SOR 法の計算を続行することになる．

図 14.3: 一般曲線座標格子

`METRIC` は，測度，ヤコビアン，測度テンソルなどの値を計算するものである．始めに点 $X$, $P$, $U$, $V$ すなわち $(2i_f+1)\times(2j_f+1)$ 個の点における座標 $x$, $y$，測度 $x_\xi$, $y_\xi$, $x_\eta$, $y_\eta$，ヤコビアン $J = x_\xi y_\eta - x_\eta y_\xi$，測度 $\xi_x = y_\eta/J,\ \xi_y = -x_\eta/J,\ \eta_x = y_\xi/J,\ \eta_y = x_\xi/J$ の値が求められる．また点 $U$ または $V$ における測度テンソル成分 $\tilde{g}_{11} = (x_\xi^2 + y_\xi^2)/J,\ \tilde{g}_{12} = (x_\xi x_\eta + y_\xi y_\eta)/J,\ \tilde{g}_{22} = (x_\eta^2 + y_\eta^2)/J$ が計算され，それぞれ配列 `gU` または `gV` に格納される．$\tilde{g}^{11} = \tilde{g}_{22},\ \tilde{g}^{12} = -\tilde{g}_{12},\ \tilde{g}^{22} = \tilde{g}_{11}$. 最後に点 $U$ の測度の微分 $\partial\xi_x/\partial\xi,\ \partial\xi_x/\partial\eta,\ \partial\xi_y/\partial\xi,\ \partial\xi_y/\partial\eta$ と点 $V$ の $\partial\eta_x/\partial\xi,\ \partial\eta_x/\partial\eta,\ \partial\eta_y/\partial\xi,\ \partial\eta_y/\partial\eta$ が計算され，配列 `dxix` に格納される．

`PREDCT` は，体積流束 $\tilde{U}, \tilde{V}$ と静圧 $p$ の予測値を与えるものである．まず上流境界の流速が平均流速 1 の放物線分布 $u_{0,j} = 6y_{\tilde{0},j}(1 - y_{\tilde{0},j}),\ y_{\tilde{0}j} = y_{0j}/y_{0j_f},\ v_{0j} = 0$ によって与えられる．このとき体積流速は $\tilde{U}_{0j} = (J\xi_x u)_{0jU},\ \tilde{V}_{0j} = 0$ となる．反変速度が $\xi\eta$ 空間の流速であることを考えれば，予測値は $\tilde{U}_{ij} = \tilde{U}_{0j},\ \tilde{V}_{ij} = \tilde{V}_{0j}$

のように置くことができる．平行平板間流路の層流では，粘性による圧力降下は $\Delta p = \nu(\partial^2 u/\partial y^2)\Delta x$ から求められる．ステップの上流側では $\partial^2 u/\partial y^2 = -4/3$，下流側では $\partial^2 u/\partial y^2 = -9/16$ である．また拡大部の (回復) 圧力上昇は $\Delta p = 1/2 - .75^2/2 = 7/32$ となる．したがって下流境界の圧力を 0 とすれば，上流境界の圧力の予測値は $p_0 = \{-(4/3)x_0 + (9/16)x_{i_f}\}/Re - 7/32$ となる．プログラムでは，$p$ の予測値は $j$ 方向に一様で，上流側 $(0 \le i \le 35)$ では $p_{ij} = p_0 + (4/3)(x_{0j} - x_{ij})/Re$，下流側の $(35 \le i \le i_0)$ では $p_{ij} = (9/16)(x_{ijf} - x_{ij})/Re$ とし，またステップ下流 $(35 < i < 100)$ では $p_{35,j}$ と $p_{100,j}$ から線形補間している．なお拡大部の圧力上昇は，平均流速の $\bar{u}^2/2$ からではなく $(1/2Y)\int_0^Y [u(y)]^2 dy$ から求めれば，$\Delta p = .6 - .6 \times .75^2 = .2625$ となる．

COMPUZ は点 $X$ の流速 $u, v$ と渦度 $\zeta$ を計算するサブルーチンである．これらの量の計算式は式 (14.8) または (14.12) から次のようになる．

$$u = \eta_y \tilde{U} - \xi_y \tilde{V}, \qquad v = -\eta_x \tilde{U} + \xi_x \tilde{V} \tag{14.26}$$

$$\zeta = \frac{1}{J}\Big\{ \frac{\partial}{\partial \xi}(\tilde{g}_{21}\tilde{U} + \tilde{g}_{22}\tilde{V}) - \frac{\partial}{\partial \eta}(\tilde{g}_{11}\tilde{U} + \tilde{g}_{12}\tilde{V}) \Big\} \tag{14.27}$$

なお INTP は格子点 $X$ の値が既知のときに点 $U$ または $V$ の値を補間するサブルーチンで，もちろん点 $U$ または $V$ の値から点 $P$ の値の補間にも使用できる．INTVP は点 $V$ の $\tilde{V}$ の値から点 $P$ の $\tilde{V}_P$ の値を境界条件 $\tilde{V}_{i0} = (\partial \tilde{V}/\partial \eta)_{i0} = 0$ を考慮して補間するものである．同様に INTVX は点 $V$ の $\tilde{V}$ の値から点 $X$ の $\tilde{V}_X$ の値を，INTUX は点 $U$ の $\tilde{U}$ の値から点 $X$ の $\tilde{U}_X$ の値を境界条件を考慮して補間するものである．等間隔計算点 $x_i$ の関数値 $u_i = u(x_i)$ が与えられるときに中間点 $x_{i+1/2}$ の微分値は周知のように $u'_{i+1/2} = (u_{i+1} - u_i)/\Delta x$ である．また境界 $x_{-1/2}$ のところでは内点の関数値を用い $u'_{-1/2} = (-2u_0 + 3u_1 - u_2)/\Delta x$ となるが，特に境界値 $u_{-1/2} = 0$ の場合には $u'_{-1/2} = (3u_0 - u_1/3)/\Delta x$ となる．

$$\nabla \cdot \boldsymbol{u} = \frac{1}{J}\Big( \frac{\partial}{\partial \xi}\tilde{U} + \frac{\partial}{\partial \eta}\tilde{V} \Big)$$

COMPU1 と COMPU2 は，Navier-Stokes 方程式を 2 段階に分けて解く SMAC 法において，それぞれ前段の $\tilde{\boldsymbol{U}}^* = J\boldsymbol{U}^*$，後段の $\tilde{\boldsymbol{U}} = J\boldsymbol{U}$ を計算するサブルーチンである．前段の COMPU1 から説明する．$\Delta$形陰解法の式 (14.21a) は，2 次元の場合には次のようになる．

$$\Big\{ J + \Delta t\,\theta\Big( \tilde{U}\frac{\partial}{\partial \xi} + \tilde{V}\frac{\partial}{\partial \eta} \Big) - \mathfrak{D}_1 \Big) \Big\} \Delta U_1^* = rhs_1^n,$$

$$\Big\{ J + \Delta t\,\theta\Big( \tilde{U}\frac{\partial}{\partial \xi} + \tilde{V}\frac{\partial}{\partial \eta} \Big) - \mathfrak{D}_2 \Big) \Big\} \Delta U_1^* = rhs_2^n$$

ただし $\mathfrak{D}_i$ は粘性項の主要部で，また右辺 $rhs$ は次のようになる．

$$rhs_1 = -\Delta t\Big\{ \frac{\partial}{\partial \xi}\tilde{U}U + \frac{\partial}{\partial \eta}\tilde{V}U - u\Big( \tilde{U}\frac{\partial \xi_x}{\partial \xi} + \tilde{V}\frac{\partial \xi_x}{\partial \eta} \Big) - v\Big( \tilde{U}\frac{\partial \xi_y}{\partial \xi} + \tilde{V}\frac{\partial \xi_y}{\partial \eta} \Big) + \tilde{g}^{11}p_\xi + \tilde{g}^{12}p_\eta + \nu\zeta_\eta \Big\},$$

$$rhs_2 = -\Delta t\Big\{ \frac{\partial}{\partial \xi}\tilde{U}V + \frac{\partial}{\partial \eta}\tilde{V}V - u\Big( \tilde{U}\frac{\partial \eta_x}{\partial \xi} + \tilde{V}\frac{\partial \eta_x}{\partial \eta} \Big) - v\Big( \tilde{U}\frac{\partial \eta_y}{\partial \xi} + \tilde{V}\frac{\partial \eta_y}{\partial \eta} \Big) + \tilde{g}^{21}p_\xi + \tilde{g}^{22}p_\eta - \nu\zeta_\xi \Big\}$$

これらの式は左辺の演算子に 1 次上流差分，右辺の対流項に Chakravarthy-Osher TVD スキームを用い差

分方程式に書き換えれば次のようになる．

$$J\Delta U^* + \Delta t\,\theta\Big\{\Big(\tilde U^+ + \frac{\tilde g_{22}}{Re}\Big)(\Delta U^*_{ij} - \Delta U^*_{i-1,j}) + \Big(\tilde U^- - \frac{\tilde g_{22}}{Re}\Big)(\Delta U^*_{i+1,j} - \Delta U^*_{ij}) + \Big(\tilde V^+ + \frac{\tilde g_{11}}{Re}\Big)(\Delta U^*_{ij} - \Delta U^*_{i,j-1}) + \Big(\tilde V^- - \frac{\tilde g_{11}}{Re}\Big)(\Delta U^*_{i,j+1} - \Delta U^*_{ij})\Big\} = rhs_1 \tag{14.28a}$$

$$J\Delta V^* + \Delta t\,\theta\Big\{\Big(\tilde U^+ + \frac{\tilde g_{22}}{Re}\Big)(\Delta V^*_{ij} - \Delta V^*_{i-1,j}) + \Big(\tilde U^- - \frac{\tilde g_{22}}{Re}\Big)(\Delta V^*_{i+1,j} - \Delta V^*_{ij}) + \Big(\tilde V^+ + \frac{\tilde g_{11}}{Re}\Big)(\Delta V^*_{ij} - \Delta V^*_{i,j-1}) + \Big(\tilde V^- - \frac{\tilde g_{11}}{Re}\Big)(\Delta V^*_{i,j+1} - \Delta V^*_{ij})\Big\} = rhs_2 \tag{14.28b}$$

$$rhs_1 = -\Delta t\Big\{\widehat{\tilde U U}_{ijP} - \widehat{\tilde U U}_{i-1,jP} + \widehat{\tilde V U}_{i,j+1X} - \widehat{\tilde V U}_{ijX} - u_{ijU}\Big(\tilde U\frac{\partial \xi_x}{\partial \xi} + \tilde V\frac{\partial \xi_x}{\partial \eta}\Big)_{ijU} - v_{ijU}\Big(\tilde U\frac{\partial \xi_y}{\partial \xi} + \tilde V\frac{\partial \xi_y}{\partial \eta}\Big)_{ijU} + (\tilde g_{22})_{ijU}(p_{ij} - p_{i-1,j}) - (\tilde g_{12})_{ijU}\frac{1}{2}(p_{i,j+1U} - p_{i,j-1U}) + \frac{1}{Re}(\zeta_{i,j+1} - \zeta_{ij})\Big\} \tag{14.28c}$$

$$rhs_2 = -\Delta t\Big\{\widehat{\tilde U V}_{i+1,jX} - \widehat{\tilde U V}_{i,jX} + \widehat{\tilde V V}_{ijP} - \widehat{\tilde V V}_{i,j-1P} - u_{ijV}\Big(\tilde U\frac{\partial \eta_x}{\partial \xi} + \tilde V\frac{\partial \eta_x}{\partial \eta}\Big)_{ijV} - v_{ijV}\Big(\tilde U\frac{\partial \eta_y}{\partial \xi} + \tilde V\frac{\partial \eta_y}{\partial \eta}\Big)_{ijV} - (\tilde g_{12})_{ijV}\frac{1}{2}(p_{i+1,jV} - p_{i-1,jV}) - (\tilde g_{11})_{ijV}(p_{i,j} - p_{i,j-1}) - \frac{1}{Re}(\zeta_{i+1,j} - \zeta_{ij})\Big\} \tag{14.28d}$$

なおここでは添字 ${}_{ij}{}^n$ を一部省略している．$\widehat{\ }$ の付いている量は数値流束で，このサブルーチンには中心差分と Chakravarthy-Osher TVD スキームの 2 つが組み込まれている．後者の数値流束は

$$\widehat{\tilde U U}_{ijP} = \tilde U U_{ijP} + |\tilde U_{ijP}|\Big\{-\frac{\Delta U_{i+1/2,j}}{2} + \frac{1}{6}\mathrm{minmod}(\Delta U_{i\pm1+1/2,j},\, 4\Delta U_{i+1/2,j}) + \frac{1}{3}\mathrm{minmod}(\Delta U_{i+1/2,j},\, 4\Delta U_{i\pm1+1/2,j})\Big\} \tag{14.29}$$

$$\Delta U_{i+1/2,j} = U_{i+1,j} - U_{i,j}, \qquad \Delta U_{i\pm1+1/2,j} = \begin{cases} \Delta U_{i-1/2,j} & (\tilde U_{ijP} \ge 0) \\ \Delta U_{i+3/2,j} & (\tilde U_{ijP} < 0) \end{cases}$$

のようになる．

境界条件は次のように考慮される．流路の入口と出口に逆流はないものとする．境界とその近傍点の数値流束は次のようになる．

$$\widehat{\tilde U U}_{0jP} = \tilde U_{0jP}\frac{U_{0j} + U_{1j}}{2}, \tag{14.30a}$$

$$\widehat{\tilde V U}_{i0X} = \widehat{\tilde V U}_{ij_fX} = 0, \qquad \widehat{\tilde V U}_{i1X} = \tilde V_{i1X}\frac{1}{4}\Big(3U_{i0} + 2U_{i1} - \frac{U_{i2}}{5}\Big), \qquad (\tilde V_{i1X} > 0) \tag{14.30b}$$

$$\widehat{\tilde U V}_{1jX} = \tilde U_{1jX}\frac{V_{0j} + V_{1j}}{2}, \qquad \widehat{\tilde U V}_{i_0jX} = \tilde U_{i_0jX}\frac{1}{8}(-V_{i_0-2,j} + 6V_{i_0-1,j} + 3V_{i_0}), \tag{14.30c}$$

$$\widehat{\tilde V V}_{i0P} = \tilde V_{i0P}\frac{1}{32}(12V_{i1} - V_{i2}) \qquad (\tilde V_{i0P} > 0) \tag{14.30d}$$

なお最後の式は壁面上の反射の条件 $\partial V/\partial \eta = 0$ を用いて導かれたものである[8]

[8] $y = 0$ 壁面上で流速成分 $u, v$ はそれぞれ $u \sim y,\ v \sim y^2$ のようにふるまうので，曲線座標格子が壁面上で直交ないしはそれに近いところでこのような取り扱いは妥当と言える．

下流境界点の $\tilde{U}^*$ とその隣接点の $\tilde{V}^*$ の式の右辺の対流項は，非保存形の式を 1 次上流差分で近似した安定かつ境界から反射のない次式によって求められる．

$$C_l = J\nabla\xi_l\cdot(\boldsymbol{u}\cdot\nabla\boldsymbol{u}) = (\xi_l)_x\Big(\tilde{U}\frac{\partial u}{\partial\xi}+\tilde{V}\frac{\partial u}{\partial\eta}\Big) + (\xi_l)_y\Big(\tilde{U}\frac{\partial v}{\partial\xi}+\tilde{V}\frac{\partial v}{\partial\eta}\Big) \qquad (l=1,\,2) \tag{14.31a}$$

$l=1$ の式の $(\partial u/\partial\xi)_{i_f j}$ と $l=2$ の式の $(\partial u/\partial\xi)_{i_0 j}$ は次のように上流差分で近似される．$(\partial v/\partial\xi)_{i_f j}$, $(\partial v/\partial\xi)_{i_0 j}$ についても同様である．

$$\Big(\frac{\partial u}{\partial\xi}\Big)_{i_f j} = \frac{1}{2}(u_{i_f j}-u_{i_0 j}+u_{i_f,j+1}-u_{i_0,j+1}), \qquad \Big(\frac{\partial u}{\partial\xi}\Big)_{i_0 j} = \frac{1}{2}(u_{i_f j}-u_{i_0-1,j}) \tag{14.31b}$$

付加項，圧力項，粘性項の計算については説明を省略する．この計算には `METRIC` で計算した `dxix` $=\partial\xi_x/\partial\xi,\ldots$ と `gU,gV` $=\tilde{g}_{11},\cdots$ が用いられる．

SMAC $\Delta$形陰解法の式 (14.28) は **13.5** 節に述べた改良型近似因子法を用いて効率的に解かれる．**13.5** 節の因子化する前の差分方程式

$$c_0 u_{00}-c_1^+ u_{-1,0}+c_1^- u_{10}-c_2^+ u_{0,-1}+c_2^- u_{01} = rhs$$

と上記の式 (14.28a) を比較すれば，この式の係数 $c_1^+$, $c_1^-$, $c_2^+$, $c_2^-$, $c_0$ は式 (14.28a) の係数と

$$c_1^+ = -\Delta t\,\theta(\tilde{U}^+ +\tilde{g}_{22}/Re), \quad c_1^- = \Delta t\,\theta(\tilde{U}^- -\tilde{g}_{22}/Re),$$
$$c_2^+ = -\Delta t\,\theta(\tilde{V}^+ +\tilde{g}_{11}/Re), \quad c_2^- = \Delta t\,\theta(\tilde{V}^- -\tilde{g}_{11}/Re), \quad c_0 = J-c_1^+-c_1^--c_2^+-c_2^-$$

のように対応していることが分かる．これらの係数は配列 `c` に入れられ，このようにして得られた連立 1 次方程式はサブルーチン `GAUSSZ` で解かれる．`GAUSSZ` については `13.5` 節の説明参照．

後段の `COMPU2` は式 (14.21b) から $\tilde{\boldsymbol{U}}^{n+1}$ を計算する短いサブルーチンである．式 (14.21b) の差分方程式は次のようになる．

$$\tilde{U}_{ij}^{n+1} = \tilde{U}_{ij}^{*} - \Delta t\Big\{(\tilde{g}_{22})_{ijU}(\phi_{ij}-\phi_{i-1,j}) - (\tilde{g}_{12})_{ijU}\frac{\phi_{i,j+1U}-\phi_{i,j-1U}}{2}\Big\} \tag{14.32a}$$

$$\tilde{V}_{ij}^{n+1} = \tilde{V}_{ij}^{*} - \Delta t\Big\{-(\tilde{g}_{12})_{ijV}\frac{\phi_{i+1,jV}-\phi_{i-1,jV}}{2} + (\tilde{g}_{11})_{ijV}(\phi_{ij}-\phi_{i,j-1})\Big\} \tag{14.32b}$$

なお非定常流れの場合にも，式 (14.23a) から $\tilde{\boldsymbol{U}}^{*(m)}$，式 (14.23b) から $\tilde{\boldsymbol{U}}^{(m)}$ を全く同じに要領で計算することができる．

`COEF`，`COMPP` は $\phi$ の差分方程式の係数を設定し，その連立 1 次方程式を SOR 法または Chebyshev SLOR 法で解いて $\phi$ の値を求め，静圧 $p$ の値を更新するサブルーチンである．解くべき $\phi$ の微分方程式 (14.21c) または (14.23c) は

$$\frac{\partial}{\partial\xi}\Big(\Delta t\,\tilde{g}_{22}\frac{\partial\phi}{\partial\xi}-\Delta t\,\tilde{g}_{12}\frac{\partial\phi}{\partial\eta}\Big) + \frac{\partial}{\partial\eta}\Big(-\Delta t\,\tilde{g}_{12}\frac{\partial\phi}{\partial\xi}+\Delta t\,\tilde{g}_{11}\frac{\partial\phi}{\partial\eta}\Big) = \frac{\partial\tilde{U}^*}{\partial\xi}+\frac{\partial\tilde{V}^*}{\partial\eta} \tag{14.33}$$

のようなもので，その差分方程式は形式的に次のように表すことができる．

$$\sum_{i'=-1}^{1}\sum_{j'=-1}^{1} c_{ij\,i'j'}\phi_{i+i',j+j'} = rhs_{ij} \tag{14.34}$$

より具体的に書けば次のようになる．

$$
\begin{aligned}
&(g_1)_{i+1,jU}(\phi_{i+1,j}-\phi_{ij}) + (g_2)_{i+1,jU}(\phi_{i+1,j+1}+\phi_{i,j+1}-\phi_{i+1,j-1}-\phi_{i,j-1})\\
-\,&(g_1)_{ijU}(\phi_{ij}-\phi_{i-1,j}) \quad -(g_2)_{ijU}(\phi_{i,j+1}+\phi_{i-1,j+1}-\phi_{i,j-1}-\phi_{i-1,j-1})\\
+\,&(g_4)_{i,j+1V}(\phi_{i,j+1}-\phi_{ij}) + (g_3)_{i,j+1V}(\phi_{i+1,j+1}+\phi_{i+1,j}-\phi_{i-1,j+1}-\phi_{i-1,j})\\
-\,&(g_4)_{ijV}(\phi_{ij}-\phi_{i,j-1}) \quad -(g_3)_{ijV}(\phi_{i+1,j}+\phi_{i+1,j-1}-\phi_{i-1,j}-\phi_{i-1,j-1})\\
&= \tilde{U}^*_{i+1,j}-\tilde{U}^*_{ij}+\tilde{V}^*_{i,j+1}-\tilde{V}^*_{ij}
\end{aligned}
$$

ただし $(g_1)_{ijU}=(\Delta t\,\tilde{g}_{22})_{ijU}$, $(g_2)_{ijU}=-(\Delta t\,\tilde{g}_{12})_{ijU}/4$, $(g_3)_{ijV}=-(\Delta t\,\tilde{g}_{12})_{ijV}/4$, $(g_4)_{ijV}=(\Delta t\,\tilde{g}_{11})_{ijV}$ で，これらの値は配列 gw に格納される．なお局所時間ステップ法では $\Delta t$ は局所時間を取る．

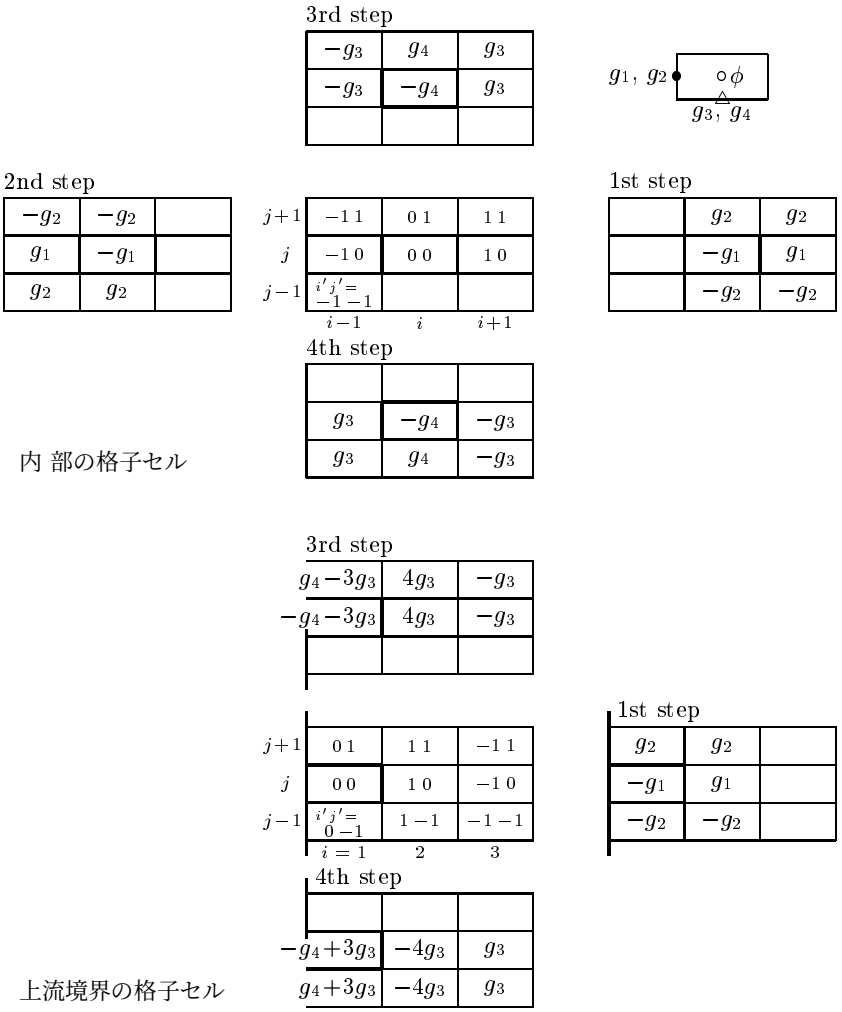


図 14.4: 圧力差分方程式の係数の設定

$\phi$ の係数 $c_{ij\,i'j'}$ の値は，図14.4に示すように４段階に分けて配列 co に入力される．この図には２つの場合，領域内部の格子セル，上流境界に隣接する格子セルの場合が取り上げられている．$\phi$ の１階微分は，一般には中心差分 $(\phi_\xi)_{ij}=(\phi_{i+1,j}-\phi_{i-1,j})/2$ が用いられるが，上流境界では片側差分 $(\phi_\xi)_{0j}=(-3\phi_{0j}+4\phi_{1j}-\phi_{2j})/2$

が用いられる．係数 $c_{0j\,4j'}$ の値は空いている co(0,j,1,jp) に入れられる．上方と下方の流路壁に隣接する格子セルの場合にも，上流境界に隣接する格子セルの場合と同様の取り扱いが必要である．なお下流境界には $p_{i_f\,j}=0$ を与えているので，下流境界に隣接する格子セルは領域内部のものと同じに取扱える．

$\phi$ の連立 1 次方程式は，ここでは趣向をかえ SOR 法または Chebyshev SLOR 法で解くことにする．SOR 法では，奇数回の掃引は点 $ij=00$ から $i_0j_0$ まで，また偶数回の掃引は点 $i_0-1, j_0-1$ から 11 まで逆向きに行われる．境界点の点緩和は，このように過緩和の効果が損なわれないように行われるべきである．$\phi$ の差分方程式 (14.34) の残差値は

$$Res_{ij}=\sum_{i'=-1}^{1}\sum_{j'=-1}^{1}c_{ij\,i'j'}\phi_{i+i',j+j'}-rhs_{ij}$$

から求められ，点過緩和の修正計算は次式によって行われる．

$$\phi_{ij}^{(n+1)}=\phi_{ij}^{(n)}-\alpha Res_{ij}/c_{ij00}$$

$\alpha=1.5$ は過緩和係数である．

非定常流れの場合には，各時間ステップごとに，$\phi$ の差分方程式を十分に満足させる必要があり，ここではベクトルコンピュータの性能を十分に引き出せるよう，Chebyshev SLOR 法を用意した．この方法では緩和計算は奇数列 (i=1,3,5, . . . ) と偶数列 (i=0,2,4, . . . ) の 2 段階に分け，各段階の計算では 3 重対角行列の連立 1 次方程式の系が Gauss 消去法で解かれる． 1 つの列の連立 1 次方程式は

$$\begin{aligned}
&\alpha c_{i0\,00}\phi_{i0}^{(n+1)}+c_{i0\,01}\phi_{i1}^{(n+1)}+c_{i0\,02}\phi_{i2}^{(n+1)}\\
&=r_{i0}-\sum_{j'=0}^{2}(c_{i0\,-1j'}\phi_{i-1,j'}^{(n)}+c_{i0\,1j'}\phi_{i+1,j'}^{(n)})-(1-\alpha)c_{i0\,00}\phi_{i0}^{(n)}\\
&c_{ij\,0\,-1}\phi_{i,j-1}^{(n+1)}+\alpha c_{ij\,00}\phi_{ij}^{(n+1)}+c_{ij\,01}\phi_{i,j+1}^{(n+1)}\\
&=r_{ij}-\sum_{j'=-1}^{1}(c_{ij\,-1j'}\phi_{i-1,j+j'}^{(n)}+c_{ij\,1j'}\phi_{i+1,j+j'}^{(n)})-(1-\alpha)c_{ij\,00}\phi_{ij}^{(n)}\qquad(j=1,2,\ldots,j_0)\\
&c_{ij_0\,0\,-2}\phi_{i,j_0-2}^{(n+1)}+c_{ij_0\;0\,-1}\phi_{i,j_0-1}^{(n+1)}+\alpha c_{ij_0\,00}\phi_{i,j_0}^{(n+1)}\\
&=r_{ij_0}-\sum_{j'=-2}^{0}(c_{ij_0\,-1j'}\phi_{i-1,j_0+j'}^{(n)}+c_{ij_0\,1j'}\phi_{i+1,j_0+j'}^{(n)})-(1-\alpha)c_{ij_0\,00}\phi_{ij_0}^{(n)}
\end{aligned}$$

のようになる．ただし $\alpha$ は加速のパラメータである．なお上流境界の連立 1 次方程式は境界条件が入り多少異なるものになる．係数 $c_{ij\,i'j'}$ の値は一般に配列要素 co(i,j,ip,jp) に入れられるが，$c_{i0\,i'2}$ は co(i,0,ip,-1) に，$c_{ij_0\,i'\,-2}$ は co(i,j0,ip,1) に入れられる．

これらの連立 1 次方程式はサブルーチン GAUSSP を引用して解かれる．GAUSSP では奇数列または偶数列の連立 1 次方程式が一まとめに同時に解かれる．また各連立 1 次方程式は，次のように係数行列が変形 3

重対角行列になっている．

$$\begin{pmatrix} a_{i02} & a_{i03} & a_{i01} & & & 0 \\ a_{i11} & a_{i12} & a_{i13} & & & \\ & a_{i21} & a_{i22} & a_{i23} & & \\ & & \ddots & \ddots & \ddots & \\ & & & a_{i\,n-1\,1} & a_{i\,n-1\,2} & a_{i\,n-1\,3} \\ 0 & & & a_{in3} & a_{in1} & a_{in2} \end{pmatrix} \begin{pmatrix} x_{i0} \\ x_{i1} \\ x_{i2} \\ \vdots \\ x_{i\,n-1} \\ x_{in} \end{pmatrix} = \begin{pmatrix} b_{i0} \\ b_{i1} \\ b_{i2} \\ \vdots \\ b_{i\,n-1} \\ b_{in} \end{pmatrix}$$

ただし添字 $i$ は $i$ 番目の連立 1 次方程式を意味する．この係数行列は 3 重対角要素に要素 $a_{i01}$ と $a_{in3}$ が加わったもので，その Gauss 消去法の計算では，通常の 3 重対角要素の計算に，その開始時，折返し点，終了時にこれらの要素の計算が追加される．このプログラムは，定常低レイノルズ数の場合には SOR 法，その他の場合には Chebyshev SLOR 法で計算するように組まれている．これらの反復計算の掃引数は残差値 `resNS` と `resP` の大きさが同等になるように決められたものである．なお Chebyshev SLOR 法はベクトル計算機を利用するときに威力を発揮する．

次に計算結果について述べる．ここにはレイノルズ数 $100, 500, 1000$ の結果を示すが，これらはいずれも $\Delta t = 1.0$，`mode` $= 2$ と置いて計算したものである．サブルーチン `COMPU1` では Chakravarthy-Osher 型の TVD スキームを用い $\theta = 1.0$，$\beta = 0.5$ と置き，またサブルーチン `COMPP` では SOR 法を用いる場合には $\alpha = 1.5$，$\beta = 0.5$，チェビシェフ SLOR 法の場合にはレイノルズ数 100 では $\alpha = \beta = 1.0$，レイノルズ数 500 と 1000 では $\alpha = 1.25$, $\beta = 0.75$ としている．図 14.5，14.6，14.7 は SOR 法の場合の結果である．これらの図には，上からステップ近くの曲線座標格子，圧力分布と速度ベクトルプロフィル，渦度分布と流線が示されている．計算は $x = -10 \sim 70.91$ の範囲で行われたが，図には全領域ではなく $x = -6 \sim 18$ と $x = 18 \sim 42$ の範囲が示されている．圧力分布は例えば $Re = 100$ の場合には，一つの色の範囲は 0.015 で，$p = 0.21$ の色は $p = 0.2025 \sim 0.2175$ の領域に塗られている．流線は下壁面で流れ関数 $\psi = 0$ 上壁面で 3 とし，$\Delta\psi = 0.25$ おきのものが描かれている．また特に剥離域内では $\Delta\psi = 0.01$ おきに描かれている．

収束判定は前章同様 $res_{NS} < 0.0001$ かつ $res_P < 0.0001$ で行っている．$Re = 1000$ の場合の解の収束状況を下表に示す．

表 14.1: バックステップ流路流れの解の収束状況 ($Re = 1000$)

$\times 10^{-5}$

| | SOR | | | Chebyshev SLOR | | |
|---|---|---|---|---|---|---|
| n | $res_{NS}$ | $res_P$ | $\Delta flow$ | $res_{NS}$ | $res_P$ | $\Delta flow$ |
| 100 | 736 | 436 | 25693 | 481 | 189 | 29678 |
| 1000 | 41 | 50 | 5095 | 157 | 79 | −6564 |
| 2000 | 12 | 16 | 3435 | 31 | 25 | 4669 |
| 2340 | 9 | 10 | 2608 | | | |
| 2460 | | | | 9 | 10 | 4402 |

収束解を得るまでの反復数 $n$ は，$\Delta$ 形陰解法の適用によって大幅に減少しているが，時間間隔 $\Delta t$ やプログラム中の係数 $\alpha$ などにもより，それらの値も適切に設定しないと効率よく収束解を得ることはできない．次表は SOR 法を用いた場合の時間間隔と収束に要する反復数の関係を示しものである．

表 14.2: 時間間隔 $\Delta t$ と収束反復数 $n$

| $Re$ | $\Delta t$ | $n$ | $x_{rap}$ | $p_{\max}$ | $p_{\min}$ |
|---|---|---|---|---|---|
| 100 | 0.1 | 2580 | 7.2 | 0.245 | 0.064 |
| | 0.25 | 1180 | 7.2 | 0.240 | 0.058 |
| | 0.63 | 900 | 7.1 | 0.242 | 0.064 |
| | 1.0 | 1366 | 7.2 | 0.242 | 0.062 |
| | 2.5 | 3562 | 7.2 | 0.242 | 0.062 |
| | 4.0 | * | 7.2 | 0.243 | 0.062 |
| 500 | 0.25 | ** | | | |
| | 0.4 | 3100 | 21.8 | −0.017 | −0.159 |
| | 0.63 | 1580 | 21.7 | −0.018 | −0.159 |
| | 1.0 | 1744 | 23.5 | −0.016 | −0.155 |
| | 1.6 | 2713 | 23.5 | −0.016 | −0.155 |
| | 2.5 | 4156 | 23.3 | −0.016 | −0.155 |
| | 4.0 | * | 23.7 | −0.015 | −0.151 |
| 1000 | 0.63 | ** | | | |
| | 1.0 | 2348 | 30.7 | −0.038 | −0.130 |
| | 1.6 | 2917 | 38.3 | −0.033 | −0.129 |
| | 2.5 | 3951 | 38.8 | −0.033 | −0.130 |
| | 4.0 | * | 41.1 | −0.034 | −0.125 |

表中の*印は $n = 5000$ 回まで計算した未収束の結果，また**印は残差値の状況から収束解が得られないと判断されたことを示す．この表にはまた再付着距離，図示領域の最大最小圧力値も示してある．これらは本来時間間隔 $\Delta t$ には無関係であるべきものであるが，表に示すように必ずしも無関係とはいえない．次表はこれらの値を更に，$\Delta t = 1.0$ に対し，本章の Chebyshev SLOR 法と前章の長方形格子 (RG) の計算結果と比較したものである．

表 14.3: バックステップ流路流れの再付着距離，最大最小圧力の比較

| | RG | SOR | | | Chebyshev SLOR | | |
|---|---|---|---|---|---|---|---|
| $Re$ | $x_{rap}$ | $x_{rap}$ | $p_{\max}$ | $p_{\min}$ | $x_{rap}$ | $p_{\max}$ | $p_{\min}$ |
| 100 | 7.3 | 7.2 | 0.242 | 0.062 | 7.1 | 0.242 | 0.064 |
| 500 | 20.8 | 23.5 | −0.016 | −0.155 | 22.5 | −0.013 | −0.143 |
| 1000 | | 30.7 | −0.038 | −0.130 | 31.4 | −0.033 | −0.124 |

このことについて読者の多くは収束判定条件があまいのではと思われるかもしれない．しかし結論を先に言えば判定条件は問題ないが，流路の角近傍の精度が不十分ということである．このプログラムでは比較的粗い格子に高次スキームを用い精度を確保しようとしているが，高次スキームも角近傍の極端に歪んだ格子に対しては精度を発揮できない．この曲線座標格子では，上記の判定条件で計算が収束した後に更に計算を継続すると，ある時点で残差が突然増大しその後再付着距離のかなり長い解に収束するかまたは発散するという不都合が生じる．その原因はおそらく $\tilde{U}_{36,0}$ の符号が正から負へ切替わることによる．通常点では **13.4** 節に詳しく述べたようにこの符号の切替わりにより対流項の値が急変することはないが，角の直後のこの点ではするということである．上記のプログラムでは $\tilde{U}_{36,0}$ の対流項を 2 次の TVD 補間式を微分した式を用いて計算するようにし，とりあえずこの問題に対処している．しかし正攻法はより細かい格子を用いることである．

図 14.5: バックステップ流路の流れ ($Re = 100$)

図 14.6: バックステップ流路の流れ ($Re = 500$)

図 14.7: バックステップ流路の流れ ($Re = 1000$)

## 14.4 反変速度を用いた拡散項の簡潔な表示

初めに層流の計算や乱流の直接シミュレーションに用いられる粘性項 $\nu\nabla^2\boldsymbol{u}$ を考える．この粘性項に左から $J\nabla\xi_l\cdot$ を演算したものは，式 (14.10)–(14.14) を用いれば次のようになる[9]．

$$D_l = J\nabla\xi_l\cdot\nu\nabla^2\boldsymbol{u} = -\nu J\nabla\xi_l\cdot\nabla\times\boldsymbol{\zeta} = -\nu\epsilon_{lij}\frac{\partial}{\partial\xi_i}\Big(\frac{\partial\boldsymbol{x}}{\partial\xi_j}\cdot\boldsymbol{\zeta}\Big) = -\nu\epsilon_{lij}\frac{\partial}{\partial\xi_i}(\tilde{g}_{jk}\tilde{Z}_k) \tag{14.35}$$

これは 6 項からなる簡潔な式である．渦度 $\tilde{Z}_k$ は式 (14.12) から求められる．すなわち

$$\tilde{Z}_k = \epsilon_{kmn}\frac{\partial}{\partial\xi_m}(\tilde{g}_{ni}\tilde{U}_i)$$

なお反変渦度 $Z_k$ は立方体セルの $\xi_k$ 軸に平行な稜の中心点に定義される．また 2 次元の場合には渦度は紙面に垂直な成分 $\zeta$ のみとなり，粘性項は次のようになる．

$$\begin{aligned} &D_1 = -\nu\frac{\partial\zeta}{\partial\eta}, \qquad D_2 = \nu\frac{\partial\zeta}{\partial\xi} \\ &\zeta = \frac{1}{J}\Big\{\frac{\partial}{\partial\xi}(\tilde{g}_{21}\tilde{U}+\tilde{g}_{22}\tilde{V})-\frac{\partial}{\partial\eta}(\tilde{g}_{11}\tilde{U}+\tilde{g}_{12}\tilde{V})\Big\} \end{aligned} \tag{14.36}$$

次に乱流の計算に用いられるより一般的な粘性拡散項を考える．この粘性項は粘性応力テンソル $\boldsymbol{\Pi}$ を用い一般に次のように表わされる[10]．

$$\begin{aligned} D_l &= J\nabla\xi_l\cdot(\nabla\cdot\boldsymbol{\Pi}) = J\frac{\partial\xi_l}{\partial x_n}\frac{\partial\tau_{mn}}{\partial x_m} = \frac{\partial}{\partial\xi_i}\Big(J\frac{\partial\xi_i}{\partial x_m}\frac{\partial\xi_l}{\partial x_n}\tau_{mn}\Big) - J\frac{\partial\xi_i}{\partial x_m}\tau_{mn}\frac{\partial}{\partial\xi_i}\frac{\partial\xi_l}{\partial x_n} \\ &= \frac{\partial}{\partial\xi_i}(JT^{il}) + \Big\{{}^{l}_{i\,j}\Big\}JT^{ij} \end{aligned} \tag{14.37}$$

ただし $\tau_{mn}$ は $\boldsymbol{\Pi}$ の成分，$T^{ij}$ はその反変成分で次のように表される．

$$T^{ij} = \frac{\partial\xi_i}{\partial x_m}\tau_{mn}\frac{\partial\xi_j}{\partial x_n}$$

乱流を渦粘性近似を置いて計算する場合には，応力テンソル成分は次式で与えられる．

$$\tau_{mn} = \nu_{\rm eff}\Big(\frac{\partial u_n}{\partial x_m}+\frac{\partial u_m}{\partial x_n}\Big)-\frac{2}{3}\delta_{mn}k \tag{14.38}$$

ただし $\nu_{\rm eff}=\nu+\nu_t$，$\nu_t$ は渦粘性係数，$k$ は乱れの運動エネルギーである．このとき応力テンソルの反変成分 $JT^{ij}$ は次のようになる．

$$\begin{aligned} JT^{ij} &= J\Bigg[\nu_{\rm eff}\frac{\partial\xi_i}{\partial x_m}\Big\{\frac{\partial}{\partial x_m}\Big(u_n\frac{\partial\xi_j}{\partial x_n}\Big)-u_n\frac{\partial^2\xi_j}{\partial x_n\partial x_m}\Big\} \\ &\qquad +\nu_{\rm eff}\frac{\partial\xi_j}{\partial x_n}\Big\{\frac{\partial}{\partial x_n}\Big(u_m\frac{\partial\xi_i}{\partial x_m}\Big)-u_m\frac{\partial^2\xi_i}{\partial x_m\partial x_n}\Big\}-\frac{2}{3}g^{ij}k\Bigg] \\ &= \nu_{\rm eff}J\frac{\partial\xi_i}{\partial x_m}\Big(\frac{\partial U_j}{\partial x_m}-U_k\frac{\partial}{\partial\xi_k}\frac{\partial\xi_j}{\partial x_m}\Big)+\nu_{\rm eff}J\frac{\partial\xi_j}{\partial x_n}\Big(\frac{\partial U_i}{\partial x_n}-U_k\frac{\partial}{\partial\xi_k}\frac{\partial\xi_i}{\partial x_n}\Big)-\frac{2}{3}\tilde{g}^{ij}k \\ &\equiv \nu_{\rm eff}(A^{ij}+A^{ji})-\frac{2}{3}\tilde{g}^{ij}k \qquad\qquad (i,\,j=1,2,3) \end{aligned} \tag{14.39}$$

[9] 第 2 式から第 3 式へは $\nabla\times(\nabla\times\boldsymbol{a})=\nabla(\nabla\cdot\boldsymbol{a})-\nabla^2\boldsymbol{a}$ なる関係，第 3 式から第 4 式へは式 (14.5) と (14.11)，また第 4 式から第 5 式へは式 (14.13) と (14.14) が用いられた．

[10] 式 (14.4)，(14.5)，(14.18) を用いる．

ここに

$$
\begin{aligned}
A^{ij} &= J\frac{\partial \xi_i}{\partial x_n}\Big(\frac{\partial U_j}{\partial x_n} - U_k\frac{\partial}{\partial \xi_k}\frac{\partial \xi_j}{\partial x_n}\Big) \\
&= J\frac{\partial \xi_i}{\partial x_n}\Big(\frac{\partial \xi_m}{\partial x_n}\frac{\partial U_j}{\partial \xi_m} + U_k\frac{\partial \xi_m}{\partial x_n}\Big\{{}^{j}_{m\,k}\Big\}\Big) \equiv \frac{\partial \xi_i}{\partial x_n}b_n^j \\
&= \tilde{g}^{im}\Big(\frac{\partial U_j}{\partial \xi_m} + \Big\{{}^{j}_{m\,k}\Big\}U_k\Big) \qquad\qquad \equiv \tilde{g}^{im}a_m^j
\end{aligned}
$$

拡散項は，渦粘性近似のもとでは，式 (14.37) と式 (14.39) から次のようになる[11]．

$$
\begin{aligned}
D_l &= \frac{\partial}{\partial \xi_i}\Big\{\nu_{\rm eff}(A^{il}+A^{li}) - \frac{2}{3}\tilde{g}^{il}k\Big\} + \Big\{{}^{l}_{i\,j}\Big\}\Big\{\nu_{\rm eff}(A^{ij}+A^{ji}) - \frac{2}{3}\tilde{g}^{ij}k\Big\} \\
&= \nu_{\rm eff}\frac{\partial}{\partial \xi_i}(A^{il}-A^{li}) + \frac{\partial \nu_{\rm eff}}{\partial \xi_i}(A^{il}+A^{li}) - \frac{2}{3}\tilde{g}^{li}\frac{\partial k}{\partial \xi_i} \qquad (l=1,\,2,\,3)
\end{aligned} \tag{14.40}
$$

ただし

$$
A^{ij} = \tilde{g}^{im}a_m^j = \frac{\partial \xi_i}{\partial x_n}b_n^j \qquad (i,\,j=1,\,2,\,3) \tag{14.41}
$$

$$
a_m^j = \frac{\partial U_j}{\partial \xi_m} + \Big\{{}^{j}_{m\,k}\Big\}U_k\,, \qquad b_n^j = \frac{\partial}{\partial \xi_k}\Big(\frac{\partial \xi_k}{\partial x_n}\tilde{U}_j - \frac{\partial \xi_j}{\partial x_n}\tilde{U}_k\Big)
$$

である[12]．なお $A^{11}$, $a_m^1$, $b_n^1$ は $\tilde{U}_1$ と同じ立方体セルの面中心に，$A^{12}$, $A^{21}$ は $Z_3$ と同じ陵の中心に定義される．

[11] 次のように計算する．

$$
\Big\{{}^{l}_{i\,j}\Big\}A^{ij} = \frac{\partial \xi_i}{\partial x_n}\Big\{{}^{l}_{i\,j}\Big\}b_n^j = -\Big(\frac{\partial}{\partial \xi_j}\frac{\partial \xi_l}{\partial x_n}\Big)b_n^j = -\frac{\partial}{\partial \xi_j}A^{lj}, \quad \Big\{{}^{l}_{i\,j}\Big\}\tilde{g}^{ij} = -J\frac{\partial \xi_i}{\partial x_n}\frac{\partial}{\partial \xi_i}\frac{\partial \xi_l}{\partial x_n} = -\frac{\partial}{\partial \xi_i}\tilde{g}^{il}
$$

[12] $\partial b_n^j/\partial \xi_j = 0$，したがって $b_n^j$ は $\xi_j = \text{const.}$ 線に沿って一定になる量である．