which cluster grids near any number of stipulated lines. For example, clustering near the two lines $y = y_1$ and $y = y_2$ is achieved by the combination

$$y = \begin{cases} \eta_1[h - f(1 - \eta/\eta_1)], & 0 \leqslant \eta \leqslant \eta_1 \\ h\eta_1 + (\eta_0 - \eta_1) f((\eta - \eta_1)/(\eta_0 - \eta_1)), & \eta_1 \leqslant \eta \leqslant \eta_0 \\ h\eta_0 + (\eta_2 - \eta_0)[h - f((\eta_2 - \eta)/(\eta_2 - \eta_0))], & \eta_0 \leqslant \eta \leqslant \eta_2 \\ h\eta_2 + (1 - \eta_2) f((\eta - \eta_2)/(1 - \eta_2)), & \eta_2 \leqslant \eta \leqslant 1. \end{cases} \tag{4.101}$$

where $f(\eta)$ is again given by eqn (4.93), $y = y_0$ is any value intermediate to y_1 and y_2, and $\eta_j = y_j/h$, $j = 0, 1, 2$. Thus we have, explicitly,

$$y = \begin{cases} h\eta_1[e^{\alpha} - e^{\alpha(1-\eta/\eta_1)}]/(e^{\alpha} - 1), & 0 \leqslant \eta \leqslant \eta_1 \\ h\eta_1 + (\eta_0 - \eta_1)h(e^{\alpha(\eta-\eta_1)/(\eta_0-\eta_1)} - 1)/(e^{\alpha} - 1), & \eta_1 \leqslant \eta \leqslant \eta_0 \\ h\eta_2 - (\eta_2 - \eta_0)h(e^{\alpha(\eta_2-\eta)/(\eta_2-\eta_0)} - 1)/(e^{\alpha} - 1), & \eta_0 \leqslant \eta \leqslant \eta_2 \\ h\eta_2 + (1 - \eta_2)h(e^{\alpha(\eta-\eta_2)/(1-\eta_2)} - 1)/(e^{\alpha} - 1), & \eta_2 \leqslant \eta \leqslant 1. \end{cases} \tag{4.102}$$

Similar expressions can be formulated on the basis of the stretching functions defined in eqns (4.88) and (4.92), and similar monotonically increasing functions can be defined to locate grid clustering near an arbitrary number of lines $y = const.$.

4.5 Two-boundary and multisurface methods

4.5.1 Two-boundary technique

The example of the divergent nozzle in the previous section shows how a two-dimensional grid can be generated in the physical domain between two boundaries, using stretching functions to control grid-density. The techniques described here start from these basic ideas, and incorporate Hermite interpolation to produce orthogonality at the boundaries. Moreover, stretching functions are used along the curved boundaries to produce the required position of grid-nodes.

Suppose we have to generate a grid between the two curves AB ($\eta = \eta_1$), CD ($\eta = \eta_2$), shown in Fig. 4.21, consisting of curves $\xi = const.$, $\eta = const.$ The parameters will be normalized so that $\eta_1 = 0$ and $\eta_2 = 1$; the curves connecting A to D and B to C will be $\xi = 0$ and $\xi = 1$, respectively. The parameter ξ could represent a normalized arc-length along the curves, and numerical integration will generally be required to

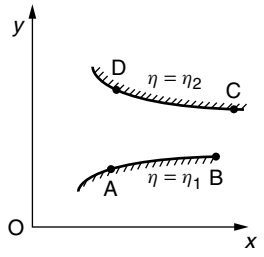

Fig. 4.21 Two-boundary grid generation.

calculate it. Stretching functions $h_{AB}(\xi)$ and $h_{DC}(\xi)$ can then be used to control the location of grid-nodes along AB and DC. Here we present the stretching function

$$h = P\xi + (1-P)\left\{1 - \frac{\tanh[Q(1-\xi)]}{\tanh Q}\right\}, \quad 0 \leqslant \xi \leqslant 1, \tag{4.103}$$

originally due to Roberts (1971) and later modified by Eiseman (1979), where P and Q are parameters to be chosen (for more details, see Section 4.6.1 below) on each boundary. In principle we can then calculate the cartesian co-ordinates of grid-nodes on the boundaries $(x_{AB}(h_{AB}), y_{AB}(h_{AB}))$ and $(x_{DC}(h_{DC}), y_{DC}(h_{DC}))$.

Stretching can also be used in the η-direction with a choice of similar stretching functions $h_{AD}(\eta)$ and $h_{BC}(\eta)$. Linear interpolation could then lead to the following expressions for cartesian co-ordinates:

$$\begin{cases} x(\xi,\eta) = \left\{1 - \tilde{h}(\xi,\eta)\right\} x_{AB}(h_{AB}(\xi)) + \tilde{h}(\xi,\eta) x_{DC}(h_{DC}(\xi)) \\ y(\xi,\eta) = \left\{1 - \tilde{h}(\xi,\eta)\right\} y_{AB}(h_{AB}(\xi)) + \tilde{h}(\xi,\eta) y_{DC}(h_{DC}(\xi)), \end{cases} \tag{4.104}$$

where $\tilde{h}(\xi,\eta) = h_{AD}(\eta) + \xi(h_{BC}(\eta) - h_{AD}(\eta))$. To achieve orthogonality at the two boundaries, however, it is possible to use the Hermite interpolation formula (4.36). Since tangent vectors at the boundaries have direction $\mathrm{d}\mathbf{r}/\mathrm{d}h$, with components $\left(\frac{\mathrm{d}x_{AB}}{\mathrm{d}h_{AB}}, \frac{\mathrm{d}y_{AB}}{\mathrm{d}h_{AB}}\right)$ and $\left(\frac{\mathrm{d}x_{DC}}{\mathrm{d}h_{DC}}, \frac{\mathrm{d}y_{DC}}{\mathrm{d}h_{DC}}\right)$, orthogonal directions will have components $\left(\frac{\mathrm{d}y_{AB}}{\mathrm{d}h_{AB}}, -\frac{\mathrm{d}x_{AB}}{\mathrm{d}h_{AB}}\right)$ and $\left(\frac{\mathrm{d}y_{DC}}{\mathrm{d}h_{DC}}, -\frac{\mathrm{d}x_{DC}}{\mathrm{d}h_{DC}}\right)$. Thus we can write the Hermite interpolation formula in component form as

$$\begin{cases} x(\xi,\eta) = \Psi_1(\eta) x_{AB}(h_{AB}) + \Psi_2(\eta) x_{DC}(h_{DC}) + T_1 \Psi_3(\eta) \dfrac{\mathrm{d}y_{AB}}{\mathrm{d}h_{AB}} + T_2 \Psi_4(\eta) \dfrac{\mathrm{d}y_{DC}}{\mathrm{d}h_{DC}} \\ y(\xi,\eta) = \Psi_1(\eta) y_{AB}(h_{AB}) + \Psi_2(\eta) y_{DC}(h_{DC}) - T_1 \Psi_3(\eta) \dfrac{\mathrm{d}x_{AB}}{\mathrm{d}h_{AB}} - T_2 \Psi_4(\eta) \dfrac{\mathrm{d}x_{DC}}{\mathrm{d}h_{DC}}, \end{cases} \tag{4.105}$$

where the two parameters T_1, T_2 that have been introduced, while not affecting orthogonality at the boundaries, can be used to control the extent to which the grid in the interior domain is orthogonal. In fact these parameters may have to be tuned to avoid folding of the grid in the interior.

The accompanying floppy disk contains a numerical code for generating a grid around an NACA-0012 airfoil using the two-boundary method. This may be found in the file *Two-boundary.f* listed in Section 4.6.4 below.

4.5.2 Multisurface transformation

The multisurface method introduced by Eiseman (1979) is another unidirectional interpolation technique for generating a grid between two given curves (or surfaces), allowing additional control over grid-node distribution by the use of intermediate curves (or surfaces). In the two-dimensional case, suppose that a given inner boundary is the curve $\mathbf{r} = \mathbf{r}_1(\xi)$, the given outer boundary is $\mathbf{r}_n(\xi)$, and we construct (n-2) non-intersecting intermediate curves $\mathbf{r}_i(\xi)$, $i = 2, 3, \ldots, (n-1)$, where ξ is a parameter for each curve. We assume that each surface is given by constant values of the independent co-ordinate

η which can be used to give an interpolation formula $\mathbf{r}(\xi, \eta)$ between inner and outer curves, with

$$\mathbf{r}(\xi, \eta_i) = \mathbf{r}_i(\xi), \quad i = 1, 2, 3, \ldots, n. \tag{4.106}$$

Piecewise-linear curves from inner curve $\mathbf{r}_1(\xi)$ to outer curve $\mathbf{r}_n(\xi)$ may be constructed by linking together points on all n curves corresponding to the same value of ξ. We assume that these piecewise-linear curves do not intersect each other. A segment of such a curve between the curves $\mathbf{r}_i(\xi)$ and $\mathbf{r}_{i+1}(\xi)$ must be the vector $\mathbf{r}_{i+1}(\xi) - \mathbf{r}_i(\xi)$. The multisurface method produces smooth co-ordinate curves $\mathbf{r}(\xi, \eta)$ with ξ fixed and η varying from η_1 to η_n by matching tangent vectors $\partial\mathbf{r}/\partial\eta$ at each surface $\mathbf{r}_i(\xi)$, $i = 1, 2, 3, \ldots, n-1$, to the directions of the line segment $\mathbf{r}_{i+1}(\xi) - \mathbf{r}_i(\xi)$. Thus we take

$$\frac{\partial \mathbf{r}}{\partial \eta} = \sum_{i=1}^{n-1} \psi_i(\eta) T_i \{\mathbf{r}_{i+1}(\xi) - \mathbf{r}_i(\xi)\}, \tag{4.107}$$

where the T_is are positive scalars associated with the surfaces $\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_{n-1}$, and the ψ_is satisfy

$$\psi_i(\eta_k) = \delta_{ik}.$$

These could be Lagrange polynomials, given by eqn (4.13).

Integration then yields

$$\mathbf{r}(\xi, \eta) = \mathbf{r}_1(\xi) + \sum_{i=1}^{n-1} \left\{ \int_{\eta_1}^{\eta} \psi_i(\eta')\, \mathrm{d}\eta' \right\} T_i \{\mathbf{r}_{i+1}(\xi) - \mathbf{r}_i(\xi)\}. \tag{4.108}$$

Note that if we put

$$T_i = \left\{ \int_{\eta_1}^{\eta_n} \psi_i(\eta')\, \mathrm{d}\eta' \right\}^{-1}, \tag{4.109}$$

we obtain an equation which is consistent at $\eta = \eta_n$, since the right-hand side of eqn (4.108) reduces to

$$\mathbf{r}_1(\xi) + \sum_{i=1}^{n-1} \{\mathbf{r}_{i+1}(\xi) - \mathbf{r}_i(\xi)\},$$

which clearly telescopes down to the left-hand side $\mathbf{r}(\xi, \eta_n) = \mathbf{r}_n(\xi)$. Equation (4.108) is, of course, also consistent at $\eta = \eta_1$.

Thus the multisurface equation is taken to be

$$\mathbf{r}(\xi, \eta) = \mathbf{r}_1(\xi) + \sum_{i=1}^{n-1} \frac{\left\{ \int_{\eta_1}^{\eta} \psi_i(\eta')\, \mathrm{d}\eta' \right\}}{\left\{ \int_{\eta_1}^{\eta_n} \psi_i(\eta')\, \mathrm{d}\eta' \right\}} \{\mathbf{r}_{i+1}(\xi) - \mathbf{r}_i(\xi)\}. \tag{4.110}$$

In the case $n = 3$, where there is one intermediate surface $\mathbf{r}_2(\xi)$, the approach we take here is to choose parameter values $\eta_1 = 0$, $\eta_2 = 1$, and η_3 initially unspecified, with value greater than one. We can then use the linear Lagrange polynomials

$$\psi_1 = 1 - \eta, \quad \psi_2 = \eta.$$

The multisurface formula becomes

$$\mathbf{r}(\xi,\eta) = \mathbf{r}_1 + \frac{\eta - \frac{1}{2}\eta^2}{\eta_3 - \frac{1}{2}\eta_3^2}(\mathbf{r}_2 - \mathbf{r}_1) + \frac{\frac{1}{2}\eta^2}{\frac{1}{2}\eta_3^2}(\mathbf{r}_3 - \mathbf{r}_2)$$
$$= \frac{(\eta_3 - \eta)(2 - \eta_3 - \eta)}{\eta_3(2 - \eta_3)}\mathbf{r}_1 + \frac{2\eta(\eta_3 - \eta)}{\eta_3^2(2 - \eta_3)}\mathbf{r}_2 + \frac{\eta^2}{\eta_3^2}\mathbf{r}_3. \tag{4.111}$$

Now if we take the value of η_3 to be very close to 1, this reduces (approximately) to the interpolation formula:

$$\mathbf{r}(\xi,\eta) = (1-\eta)^2\mathbf{r}_1(\xi) + 2\eta(1-\eta)\mathbf{r}_2(\xi) + \eta^2\mathbf{r}_3(\xi). \tag{4.112}$$

When $n = 4$ and there are two intermediate surfaces, a similar method involves taking parameter values $\eta_1 = 0$, $\eta_2 = a$, $\eta_3 = 1$, and η_4 unspecified, with $0 < a < 1$ and $\eta_4 > 1$. Then we can take second degree Lagrange polynomials

$$\psi_1 = \frac{1}{a}(a-\eta)(1-\eta), \quad \psi_2 = \frac{\eta(1-\eta)}{a(1-a)}, \quad \psi_3 = \frac{\eta(\eta-a)}{(1-a)}.$$

Equation (4.110) now yields the formula

$$\mathbf{r} = \mathbf{r}_1 + \frac{\left\{\frac{1}{3}\eta^3 - \frac{1}{2}(1+a)\eta^2 + a\eta\right\}}{\left\{\frac{1}{3}\eta_4^3 - \frac{1}{2}(1+a)\eta_4^2 + a\eta_4\right\}}(\mathbf{r}_2 - \mathbf{r}_1)$$
$$+\frac{\left(\frac{1}{2}\eta^2 - \frac{1}{3}\eta^3\right)}{\left(\frac{1}{2}\eta_4^2 - \frac{1}{3}\eta_4^3\right)}(\mathbf{r}_3 - \mathbf{r}_2) + \frac{\left(\frac{1}{3}\eta^3 - \frac{1}{2}a\eta^2\right)}{\left(\frac{1}{3}\eta_4^3 - \frac{1}{2}a\eta_4^2\right)}(\mathbf{r}_4 - \mathbf{r}_3). \tag{4.113}$$

Again, if we take η_4 to be very close to 1, this expression reduces (approximately) to the formula:

$$\mathbf{r}(\xi,\eta) = (1-\eta)^2(1-a_1\eta)\mathbf{r}_1(\xi) + (2+a_1)\eta(1-\eta)^2\mathbf{r}_2(\xi)$$
$$+\eta^2(1-\eta)(2+a_2)\mathbf{r}_3(\xi) + \eta^2(1-a_2+a_2\eta)\mathbf{r}_4(\xi), \tag{4.114}$$

where

$$a_1 = \frac{2}{3a-1} \quad \text{and } a_2 = \frac{2}{2-3a}. \tag{4.115}$$

Clearly we must avoid taking the values $\frac{1}{3}$ or $\frac{2}{3}$ for a.

Note that eqn (4.110) can also be applied in the trivial case where $n = 2$, when there are no intermediate curves. Taking ψ_1 to be a constant then leads to simple linear interpolation between inner and outer curves:

$$\mathbf{r}(\xi,\eta) = \mathbf{r}_1(\xi) + \eta\{\mathbf{r}_2(\xi) - \mathbf{r}_1(\xi)\}. \tag{4.116}$$

4.5.3 Numerical implementation

Here we describe briefly the numerical implementation of the above methods for the case of an airfoil NACA-0012. The relevant programs, *Two-boundary.f* and *Multisurface.f*, are listed at Section 4.6.4 and may be found in the directory *Book/tfi.gds* on

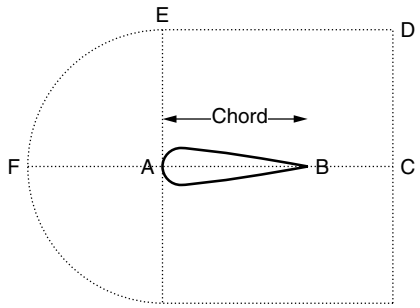

Fig. 4.22 Airfoil profile.

the companion website (www.bh.com/companions/0750650583). The geometry of the domain is shown in Fig. 4.22, with the curve AB representing the profile of an airfoil, one of a family of airfoils with the generic name NACA-00t, where t indicates the thickness as a percentage of the chord length as a two-digit number. Thus an NACA-0012 airfoil has 12% thickness.

The profiles of this family of airfoils are given (with origin at A and chord length unity) by the analytic expression

$$y = t\left(a_1 x^{\frac{1}{2}} + a_2 x + a_3 x^2 + a_4 x^3 + a_5 x^4\right), \tag{4.117}$$

where $a_1 = 1.4779155$, $a_2 = -0.624424$, $a_3 = 1.727016$, $a_4 = 1.384087$, $a_5 = -0.489769$.

The airfoil has an axis of symmetry, and it is sufficient to generate a grid for the upper half of the domain. We therefore consider the domain with inner boundary ABC, consisting of the profile AB and the straight line CD, and outer boundary FED, where FE is a quarter-circle of radius AE and ED is a straight line parallel to BC. The left-hand boundary is taken as AF and the right-hand boundary as CD. Normalized length parameters ξ can then be defined along the boundaries ABC and FED ($0 \leqslant \xi \leqslant 1$), and similarly η along AF and CD ($0 \leqslant \eta \leqslant 1$). For the profile AB this requires integration based on the length formula for plane curves and the analytical expression (4.117).

The first step of the program *Multisurface.f* is to use the univariate stretching functions given by eqn (4.103) with appropriate values of P and Q to generate grid-nodes with the required clustering along the two boundaries AF and CD. The stretching functions could be denoted by $h_{AF}(\eta)$ and $h_{CD}(\eta)$. The inner surface (curve) ABC can be denoted $\mathbf{r}_1$.

The program now has three options. The first is the 'no intermediate curve' option, in which the simple linear interpolation expression eqn (4.116) is used, subject to the stretching used in step 1. The second option is the 'one intermediate curve' option, in which eqn (4.112) is used. In this option, control of orthogonality of the grid is possible at either the inner boundary or the outer boundary.

The third option, that of 'two intermediate curves', is the one we pursue here. This allows control of orthogonality at both inner and outer boundaries. The first attempt at constructing the intermediate curves uses linear interpolation between the inner curve $\mathbf{r}_1$ and the outer one, denoted by $\mathbf{r}_4$. A subroutine 'orthogonality' is

then called to adjust the positions of grid-nodes on the intermediate curves, denoted by $\mathbf{r}_2$ and $\mathbf{r}_3$, so as to enforce orthogonality of the grid at the inner and outer boundaries.

If we consider a typical grid node R with co-ordinates (x_1, y_1) on the inner boundary $\mathbf{r}_1$, at which the gradient of the boundary is calculated to be m_1, then the straight line normal to the boundary through R has equation

$$y - y_1 = -\frac{1}{m_1}(x - x_1). \tag{4.118}$$

Suppose that the corresponding node S on the linearly interpolated curve $\mathbf{r}_2^i$ has co-ordinates (x_2^i, y_2^i) and that we can estimate the gradient of the curve $\mathbf{r}_2^i$ at this point as m_2. The tangent to the curve $\mathbf{r}_2^i$ at this node has equation

$$y - y_2 = m_2(x - x_2). \tag{4.119}$$

A better position for S should be at the intersection of these two straight lines; the subroutine calculates the new co-ordinates. A similar procedure is applied to points on the outer boundary $\mathbf{r}_4$ and the linearly interpolated curve $\mathbf{r}_3$. Having now constructed new intermediate curves $\mathbf{r}_2$ and $\mathbf{r}_3$ such that grid-lines are orthogonal at inner and outer boundaries, we can employ the multisurface transformation formula (4.114), with a chosen value of a, in the form

$$\begin{aligned}\mathbf{r}(\xi, \eta) = {} & (1-h)^2(1-a_1h)\mathbf{r}_1(\xi) + (2+a_1)h(1-h)^2\mathbf{r}_2(\xi) \\ & + h^2(1-h)(2+a_2)\mathbf{r}_3(\xi) + h^2(1-a_2+a_2h)\mathbf{r}_4(\xi),\end{aligned} \tag{4.120}$$

where we define h in terms of a linear interpolation between the two stretching functions,

$$h = h_{AF} + \xi(h_{CD} - h_{AF}). \tag{4.121}$$

The program *Two-boundary.f* is essentially a modification of *Multisurface.f*. The subroutine *orthogonality* is removed, and instead Hermite interpolation is used in a modified form of eqn (4.105),

$$\begin{cases} x(\xi, \eta) = \Psi_1(h)x_{AB}(\xi) + \Psi_2(h)x_{DC}(\xi) + T_1\Psi_3(h)\dfrac{\mathrm{d}y_{AB}}{\mathrm{d}\xi} + T_2\Psi_4(h)\dfrac{\mathrm{d}y_{DC}}{\mathrm{d}\xi} \\ y(\xi, \eta) = \Psi_1(h)y_{AB}(\xi) + \Psi_2(h)y_{DC}(\xi) - T_1\Psi_3(h)\dfrac{\mathrm{d}x_{AB}}{\mathrm{d}\xi} - T_2\Psi_4(h)\dfrac{\mathrm{d}x_{DC}}{\mathrm{d}\xi}, \end{cases} \tag{4.122}$$

with h given by (4.121).

4.6 Website programs

Here we list the programs in the directory *Book* contained on the companion website (www.bh.com/companions/0750650583) that are relevant to the material in this chapter. First the subdirectories are specified and then the names of the files with a description are given.

4.6.1 Subdirectory: Book/univariate.gds

1. File: linear.f
   This program simply generates a one-dimensional grid on a straight line based on the linear interpolation
   $$x = (1 - \xi)x_0 + \xi x_1, \quad 0 \leqslant \xi \leqslant 1.$$
2. File: Eriksson.f
   This generates a one-dimensional grid on a straight line using the Eriksson stretching function
   $$x = (e^{\beta\xi} - 1)/(e^{\beta} - 1), \quad 0 \leqslant \xi \leqslant 1.$$
   Here the constant β controls the degree of clustering near $x = 0$.
3. File: Compress1.f
   This generates another grid on a straight line using the stretching function
   $$x = (p^{kb} + c\xi)^{1/b} - p^k, \quad 0 \leqslant \xi \leqslant 1, \tag{4.123}$$
   where p, k, and b are constants, with $c = (p^k + 1)^b - p^{kb}$.
4. File: Compress2.f
   This generates a grid on a straight line with the stretching function
   $$x = P\xi + (1 - P)\left\{1 - \frac{\tanh[Q(1 - \xi)]}{\tanh Q}\right\}, \quad 0 \leqslant \xi \leqslant 1, \tag{4.124}$$
   mentioned above. For example, when the parameters P and Q take the values $P = 1.8$, $Q = 2.0$, an equally spaced set of points in the ξ computational domain maps into a set of points in the physical x domain with clustering near $x = 1$. When $P = 0.9$, $Q = 2.0$, a fairly equally spaced set of points in physical space is obtained, while the choice $P = 0.1$, $Q = 2.0$ gives clustering near $x = 0$.

4.6.2 Subdirectory: Book/Algebra

1. File: Algebraic1.f
   This generates a two-dimensional planar boundary-conforming grid using the co-ordinate transformation given by eqn (4.98), where the functions $h_1(x)$ and $h_2(x)$ are specified by the user. The program calculates the so-called *metrics* of the transformation, given by the partial derivatives $\partial\xi/\partial x, \partial\xi/\partial y, \partial\eta/\partial x, \partial\eta/\partial y$, which are required in transforming the hosted partial differential equations to be solved.
   In this example we can show that
   $$\frac{\partial\xi}{\partial x} = 1, \quad \frac{\partial\xi}{\partial y} = 0, \quad \frac{\partial\eta}{\partial x} = [h_2'(x)h_1(x) - h_1'(x)h_2(x) - yh_2'(x)] \times [h_2(x) - h_1(x)]^{-2},$$
   $$\frac{\partial\eta}{\partial y} = [h_2(x) - h_1(x)]^{-1}.$$

We also need in numerical calculations the partial derivatives

$$\frac{\partial x}{\partial \xi} = J\frac{\partial \eta}{\partial y}, \frac{\partial x}{\partial \eta} = -J\frac{\partial \xi}{\partial y}, \frac{\partial y}{\partial \xi} = -J\frac{\partial \eta}{\partial x}, \frac{\partial y}{\partial \eta} = J\frac{\partial \xi}{\partial x},$$

where the Jacobian J is given by

$$J = \frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \eta} - \frac{\partial x}{\partial \eta}\frac{\partial y}{\partial \xi} = h_2(\xi) - h_1(\xi).$$

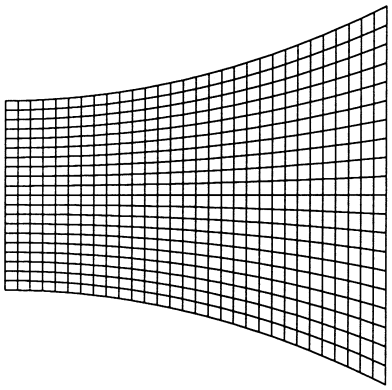

Fig. 4.23 Boundary-conforming algebraic grid.

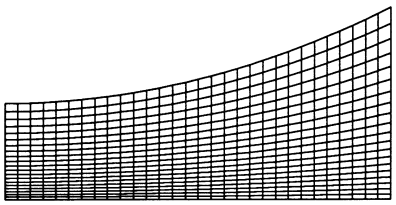

Fig. 4.24 Algebraic grid with grid-clustering near lower boundary.

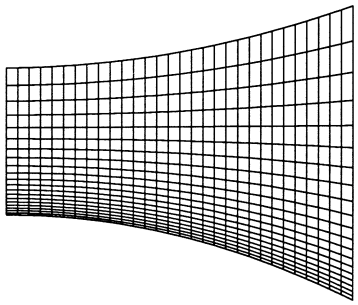

Fig. 4.25 Algebraic grid with grid-clustering near lower boundary.

2. File: Algebraic2.f
3. File: Algebraic3.f
4. File: Algebraic4.f
5. File: Algebraic5.f
   Some typical grids in divergent nozzles calculated using these programs, both without clustering and with clustering, are shown in figs 4.23–4.26.

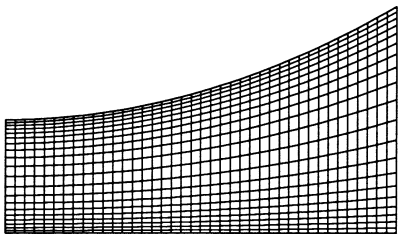

Fig. 4.26 Algebraic grid with grid-clustering near boundaries.

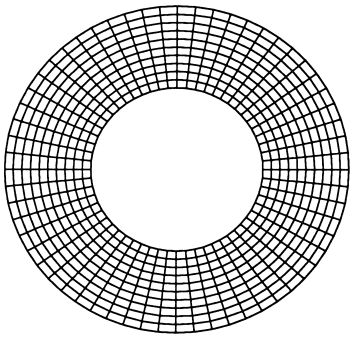

Fig. 4.27 Transfinite interpolation.

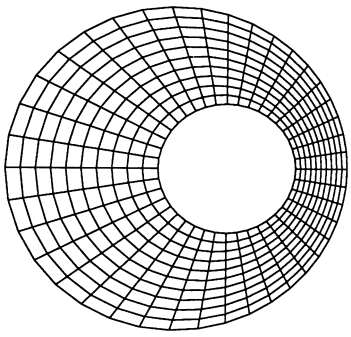

Fig. 4.28 Transfinite interpolation.

4.6.3 Subdirectory: Book/bilinear.gds

This subdirectory contains only one file.

1. File: bilinear.f
   This generates a two-dimensional planar grid in a straight-sided quadrilateral using bilinear interpolation given by

$$\mathbf{r}(\xi, \eta) = (1-\xi)(1-\eta)\mathbf{r}_{bl} + (1-\xi)\eta\mathbf{r}_{tl} + \xi(1-\eta)\mathbf{r}_{br} + \xi\eta\mathbf{r}_{tr},$$

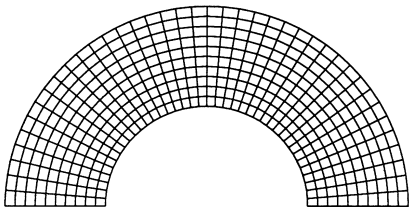

Fig. 4.29 Transfinite interpolation.

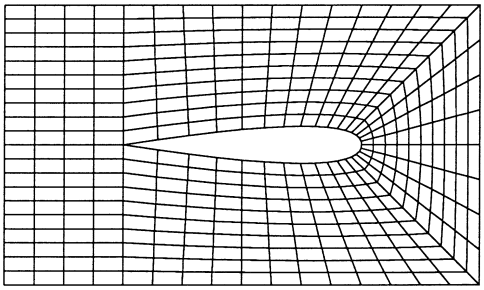

Fig. 4.30 Transfinite interpolation.

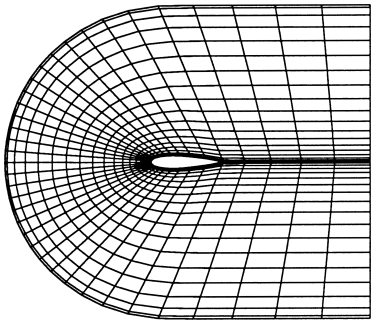

Fig. 4.31 Two-boundary technique.

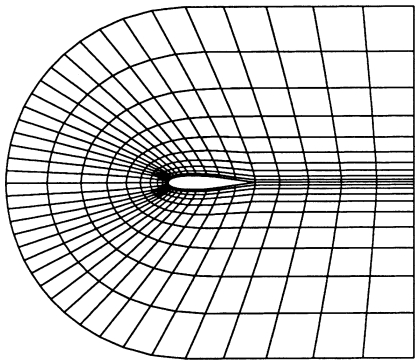

Fig. 4.32 Multisurface grid generation with one intermediate curve.

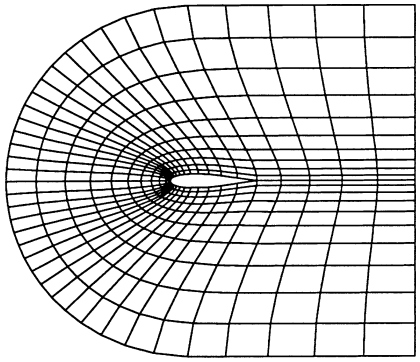

Fig. 4.33 Multisurface grid generation with two intermediate curves.

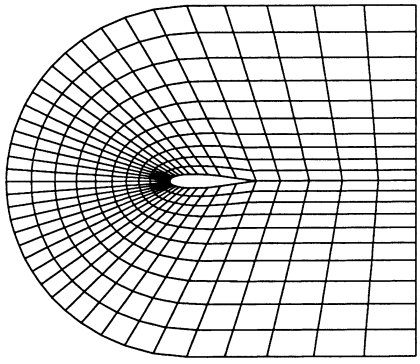

Fig. 4.34 Multisurface grid generation with no intermediate curve.

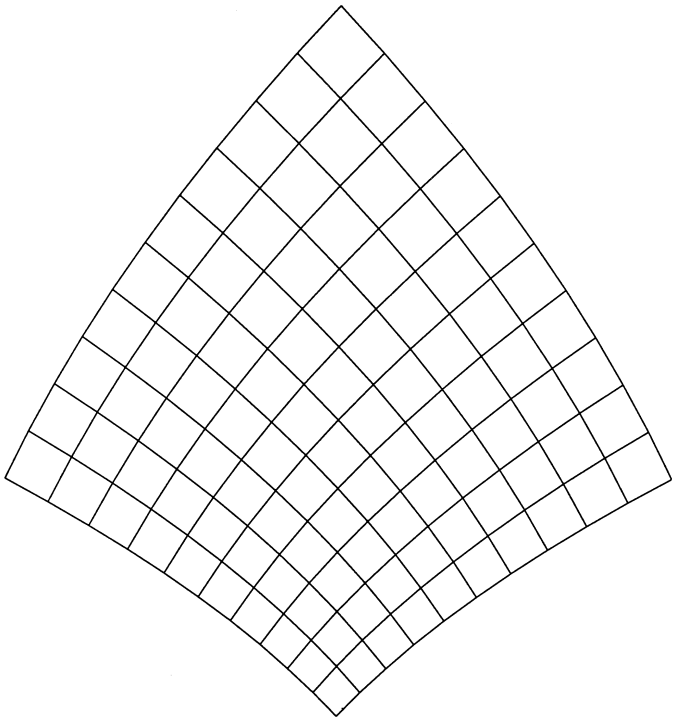

Fig. 4.35 Algebraic grid based on parabolic co-ordinates.

where $\mathbf{r}$ denotes the position vector $\binom{x}{y}$ and the subscripts bl, tl, br, tr stand for 'bottom left', 'top left', 'bottom right', and 'top right', respectively. The cartesian co-ordinates of the four corners of the quadrilateral must be specified by the user.

4.6.4 Subdirectory: Book/tfi.gds

In this subdirectory there are three files.

1. File: Transfinite.f
   This code uses transfinite interpolation to generate a planar two-dimensional grid. The boundaries of the physical domain are prescribed by specifying the cartesian co-ordinates of boundary points in a subroutine called 'boundary'. An option is included to deal with three particular geometries: (a) a square (b) an annulus (c) a trapezium.
   Some examples of grids are shown in figs 4.27–4.30.
2. File: Two-boundary.f
   A typical grid around an airfoil is shown in Fig. 4.31.
3. File: Multisurface.f
   Examples are shown in figs 4.32–4.34.

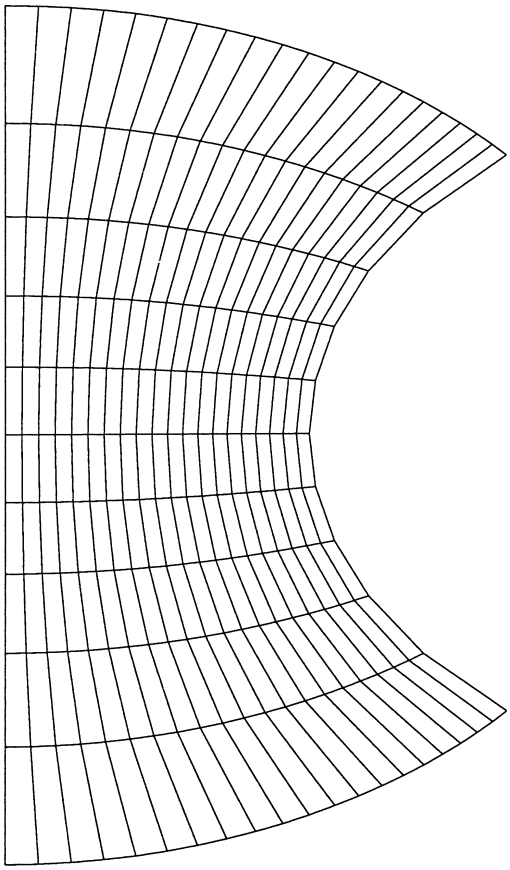

Fig. 4.36 Algebraic grid based on bipolar co-ordinates.

4.6.5 Subdirectory: Book/analytic.gds

This subdirectory contains three files. In the first two, planar two-dimensional grids are generated by discretizing classical co-ordinate transformations.

1. File: parabolic.f
   An example is shown in Fig. 4.35.
2. File: bipolar.f
   An example is shown in Fig. 4.36.