Chapter 7

Closure

We bring this thesis to a close by presenting some conclusions that we have drawn from our work, together with a few suggestions as to how our work could be usefully extended.

7.1 Conclusions

A computational method has been developed which requires only modest computing resources in order to produce extremely detailed simulations of complex two-dimensional shock hydrodynamic flows. This computational efficiency has been achieved by combining a high resolution upwind scheme with an adaptive mesh strategy which includes temporal as well as spatial refinement. The key element to our Adaptive Mesh Refinement algorithm, AMR for short, is its computational grid system.

The computational grid is formed by recursively embedding fine mesh patches inside coarser mesh patches, each patch being a quasi-rectangular mesh. A process has been developed whereby the computational grid automatically adapts to the numerical flow solution thereby improving the resolution of specific features such as shock waves and contact discontinuities. This form of adaptive mesh is not as geometrically flexible as an unstructured one, however it does offer a number of advantages for the time accurate simulation of unsteady flows. Not least of which is the fact that it allows a temporal refinement strategy to be used to integrate the flow solution. More, but smaller time steps are taken on fine meshes than on coarse meshes, with the various time steps being sequenced such that the calculation remains time accurate. Thus the presence of a few very fine mesh cells in one part of the flow domain does not have an adverse affect on the rate at which the rest of the flow domain can be integrated. Another important advantage is the comparative ease with which the grid may be adapted. An important consideration for unsteady flows where it may be necessary to adapt the grid several hundred times during the course of a simulation. Our calculations have shown that the AMR algorithm

typically spends less than 1% of its time adapting the computational grid.

Although the majority of our calculations have used Roe's flux difference splitting scheme to integrate the Euler equations, in principle, any second-order cell-centred scheme developed for a single quasi-rectangular mesh may be incorporated into the AMR algorithm. For example, it would be straightforward to use Godunov's scheme to integrate the shallow water equations. Roe's scheme was chosen because of its inherent robustness and for the balance it strikes between resolution and computational expense. In common with many schemes, Roe's method is known to admit certain types of spurious solution. Incidental to our study we have found a hitherto unpublicised failing of the scheme which we have attributed to an incorrect treatment of shear waves. For the calculation in which this failing first manifested itself a simple cure has been devised and full details are presented, but the general applicability of this cure has yet to be explored.

Throughout this thesis we have taken great care to substantiate the claims made for the AMR algorithm. This process culminates with the presentation of computational results which are comparable in resolution to Schlieren photographs, and yet these results were produced within reasonable time limits using a small desktop workstation. And so we conclude, that contrary to popular belief, having access to a supercomputer is not a pre-requisite for for being able to perform so-called state-of-the-art shock hydrodynamic simulations. However, it remains to be seen to what extent our algorithm could take advantage of such computing power.

Now, compared to non-adaptive schemes the AMR algorithm is undeniably complicated for it contains many processes that require careful co-ordination. But given the quality of our computational results there can be no doubt that this complexity is justified. However, one of the reasons why our project has been successful lies with the fact that we have been able to cut across the boundaries between several areas of different study; albeit with varying levels of expertise. In particular, we have had to expend much effort in developing an interactive graphical package so as to be able to examine our computational results.

Finally, we present a list of reasons for the impressive performance of the AMR algorithm.

Principal reasons why the AMR algorithm proves to be robust

- The algorithm is built up from numerous naïve, foolproof procedures.
- The algorithm tries to circumvent rather than cure numerical difficulties.
- The algorithm is able to assume the stability characteristics associated with a class of robust *shock-capturing* schemes which have been developed for use with quasi-rectangular meshes.
- The grid adaption process preserves the representation of flow discontinuities, exactly

Principal reasons why the AMR algorithm proves to be efficient

- The algorithm refines in time as well as space.
- Most overheads associated with the grid data structure are incurred as a one off *per* mesh, therefore the overhead associated with an individual mesh cell is negligible.
- The grid adaption operators process far fewer cells than do the grid integration operators, so the cost of adapting the grid is negligible.

7.2 Future Work

The AMR algorithm has proved to be an efficient and robust means of producing very detailed simulations for complex shock on shock interactions, as such it forms a rounded piece of work. Nevertheless, there are a number of areas in which our work could be either improved or built upon, and a few suggestions are now given.

There is no inherent reason why the AMR algorithm cannot be extended to three-dimensions. All the various elements of the algorithm generalise in a straightforward manner. For example, the process that integrates a mesh would only require an extra sweep operator, while the area sub-divide process used to create new meshes would simply become a volume sub-divide process. However, such a scheme would be limited by the poor volumetric packing capabilities of cartesian mesh blocks. But this is only a problem if body-fitted meshes are used. In principle, one could simply allow the computational grid to intersect solid boundaries and then apply special integration procedures for mesh cells on or near the boundary. Several workers have shown that this approach can be made to work in two-dimensions, but further work is required if their ideas are to be generalized to three-dimensions.

The longevity of the AMR algorithm is likely to depend on how well it can exploit parallel computer architectures. While it is not difficult to identify coarse, medium and fine grain parallelism within the algorithm it remains to be seen whether this parallelism can be exploited or not. Indeed, any proposed method of exploitation would depend on the architecture of the target machine, for example, SIMD or MIMD and shared or distributed memory. Some architectures are bound to be more suitable than others. Consequently, a feasibility study is required in order to determine the extent to which the AMR algorithm can exploit these modern developments in computer hardware.

Much of the AMR algorithm has been developed using a combination of physical argument and extrapolation from theory developed for the linear advection equation. Consequently, there are a number of grey areas which from a theoretician's point of view warrant a second more rigorous look. For example, the correct manner in which to enforce conservation at *fine-coarse* mesh boundaries without compromising accuracy is not clear.

Finally, to emphasize the rounded nature of our work, the AMR algorithm has reached a stage where it could be used as a genuine research tool by Fluid Dynamicists in their endeavour to understand basic shock wave phenomena. For example, it would be possible to simulate the transition from regular reflection to Mach reflection using several different fluid models. Thereby gaining a better understanding of the subtleties that viscous and real gas effects play in the transition process.

References

[1] C.M.Albone, *An upwind, multigrid, shock-fitting scheme for the Euler equations*, RAE Technical Report 85004, (1985).

[2] R.Arieli and J.D.Murphy, *Pseudo-direct solution to the boundary layer equations for separated flow*, AIAA Journal, Vol. **18**(1980), No. 8, pp. 883–891.

[3] D.C.Arney, *An adaptive method with mesh moving and mesh refinement for solving the Euler equations*, AIAA Paper 88-3567-CP (1988).

[4] J.B.Bell, P.Collela, J.A.Trangenstein and M.Welcome, *Adaptive mesh refinement on moving quadrilateral grids*, AIAA 9^{th} Computational Fluid Dynamics conference, Buffalo, New York, (1989).

[5] G.Ben-Dor, K.Takayama and T.Kawauchi, *The transition from regular to Mach reflexion and from Mach to regular reflexion in truly non-stationary flows*, J. Fluid Mech., Vol. **100**(1980), pp. 147–160.

[6] G.Ben-Dor and K.Takayama, *Analytical prediction of the transition from Mach reflection to regular reflection over cylindrical concave wedges*, J. Fluid Mech., Vol. **158**(1985), pp. 365–380.

[7] M.J.Berger, *Adaptive mesh refinement for hyperbolic partial differential equations*, Ph.D. thesis, Computer Science Dept., Stanford University (1982).

[8] M.J.Berger and P.Collela, *Local adaptive mesh refinement for shock hydrodynamics*, J. Comput. Phys., **82**(1989), pp. 67–84.

[9] M.J.Berger and Oliger, *Adaptive mesh refinement for hyperbolic partial differential equations*, J. Comput. Phys., **53**(1984), pp. 484–512.

[10] R.Bornat, *Understanding and writing compilers*, Macmillan (1984).

[11] J.P.Boris and D.L.Book, *Flux-corrected transport,* I. *SHASTA, a fluid transport algorithm that works*, J. Comput. Phys., **11**(1973), pp. 38–69.

[12] A.Bowyer, *Computing Dirichlet Tessellations*, The Computer Journal, Vol. **24**(1981), pp. 162–166.

[13] J.U.Brackbill and J.S.Saltzmann, *Adaptive zoning for singular problems in two dimensions*, J. Comput. Phys., **46**(1982).

[14] A.E.Bryson and R.W.F.Gross, *Diffraction of strong shocks, by cones, cylinders, and spheres*, J. Fluid Mech., Vol. **10**(1961), pp. 1–16.

[15] J.Casper, Private communication (1990).

[16] P.Collela and P.R.Woodward, *The piecewise parabolic method (ppm) for gas-dynamical simulations*, J. Comput. Phys., **54**(1984), pp. 174–201.

[17] J.F.Clarke, S.Karni, J.J.Quirk, P.L.Roe, L.G.Simmonds and E.F.Toro, *Numerical computation of two-dimensional unsteady detonation waves in high energy solids*, Cranfield Inst. of Tech., College of Aeronautics, CoA Rep. 9013, (1990).

[18] S.F.Davis, *A rotationally biased upwind difference scheme for the Euler equations*, J. Comput. Phys., **56**(1984), pp. 65–92.

[19] W.N.Dawes, *Efficient implicit algorithm for the equations of 2-D viscous compressible flow: application to shock-boundary layer interaction*, Int. J. Heat & Fluid Flow, Vol. **4** No. 1 (1983), pp. 17–26.

[20] H.Deconinck, C.Hirsch and J.Peutman, Lecture Notes in Physics, **264**(1986), pp. 216–221.

[21] B.Einfeldt, *On Godunov-type methods for gas dynamics*, SIAM J. Numer. Anal., Vol. **25**(1988), No.2, pp. 294–318.

[22] B.Einfeldt, C.D.Munz, P.L.Roe and B.Sjögreen, *On Godunov-type methods near low densities*, To appear.

[23] G.Emanuel, *Gas dynamics: theory and applications*, AIAA education series (1986).

[24] Ernst Mach Institute Frieburg, Germany (Private communication).

[25] W.Fickett and W.Davis, *Detonation*, University of California Press, Berkeley, (1979).

[26] P.Glaister, *Flux-Difference splitting techniques for the Euler equations in non-cartesian geometry*, Univ. Reading, Dep. Math., Num. Anal. Rep. 8/85, (1985).

[27] I.I.Glass, *Shock Waves & Man*, University of Toronto Press (1974).

[28] H.M.Glaz, P.Colella, I.I.Glass and R.L.Deschambault, *A numerical study of oblique shock-wave reflections with experimental comparisons*, Proc. R. Soc. Lond., A **398**(1985), pp. 117–140.

[29] S.K.Godunov, *A difference scheme for numerical computation of discontinuous solutions of equations of fluid dynamics*, Mat. Sb.,**47**(1959), pp. 271–290.

[30] R.J.Hakkinen, I.Greber, L.Trilling and S.S.Arbarbanel, *The interaction of an oblique shock wave with a laminar boundary layer*, NASA Memo-2-18-59W (1959).

[31] D.G.Holmes and S.H.Lamson, *Adaptive triangular meshes for compressible flow simulations*, Numerical grid generation in CFD, Ed. J.Hauser and C.Taylor, Pineridge Press, Swansea, (1986).

[32] M.Holt, *Numerical methods in fluid dynamics*, Berlin: Springer, 273 pp. 2nd ed., (1984).

[33] A.Jameson, W.Schmidt and E.Turkel, *Numerical solutions of the Euler equations by finite-volume methods using Runge-Kutta time-stepping*, AIAA Paper 81-1259 (1981).

[34] D.E.Knuth, *The art of computer programming, Vol.1*, Addison-Wesley (1973).

[35] R.J.Le Veque, *Cartesian grid methods for flow in irregular regions*, Numerical methods for fluid dynamics III, Edited by K.W.Morton and M.J.Baines, pp. 375–382, Clarendon Press, Oxford (1988).

[36] R.Löhner, K.Morgan, J.Praire and M.Vahdati, *Finite element flux-corrected transport* (FEM-FCT) *for the Euler and Navier-Stokes equations*, Finite Elements in Fluids – Vol 7, Edited by R.H.Gallagher, R.Glowinski, P.M.Gresho, J.T.Oden and O.C.Zienkiewicz, pp. 105–121, John Wiley & Sons, (1987).

[37] R.Löhner, *Adaptive H-refinement on 3-D unstructured grids for transient problems*, AIAA Paper 89-0653 (1989).

[38] R.W.MacCormack and B.S.Baldwin, *A numerical method for solving the Navier-Stokes equations with application to shock boundary layer interactions*, AIAA Paper 75-1 (1975).

[39] G.Moretti, *Three dimensional, supersonic, steady flows with any number of embedded shocks.*, AIAA Paper 74-10 (1974).

[40] ———,*Computation of flows with shocks*, Ann. Rev. Fluid Mech., **19**(1987), pp. 313–337.

[41] W.A.Mulder, *Multigrid relaxation for the Euler equations*, J. Comput. Phys., **60**(1985), pp. 235–252.

[42] L.E.Murr, *Shock waves for industrial applications*, Noyes: New Jersey (1988).

[43] L.E.Murr, A.W.Hare and N.G.Eror, *Introducing: the metal-matrix high-temperature superconductor*, Advanced Mater. & Processes, **132**(1987), No. 4, pp. 36–44.

[44] S.Osher, *Riemann solvers, the entropy condition, and difference approximations*, SIAM J. Numer. Anal., Vol. **21**(1984), pp. 289–315.

[45] M.Pandolfi, *A contribution to numerical prediction of unsteady flows*, AIAA J., **22**(1984), pp. 602–610.

[46] J.Pike, *Grid adaptive algorithms for the solution of the Euler equations on irregular grids*, J. Comput. Phys., **71**(1987), pp. 194–223.

[47] A.Priestley, *Roe's scheme, Euler equations, cartesian grids, non-cartesian geometries, rigid walls and all that*, Univ. Reading, Dep. Math., Num. Anal. Rep. 14/87, (1987).

[48] J.J.Quirk, *Application of Roe's upwind scheme to the calculation of axisymmetric solutions to the Euler equations*, Cranfield Inst. of Tech., College of Aeronautics, CoA Rep. NFP 90/03, (1990).

[49] M.M.Rai, *A conservative treatment of zonal boundaries for Euler equation calculations*, J. Comput. Phys., **62**(1986), pp. 472–503.

[50] J.Rinehart and J.Pearson, *Explosive working of metals*, Macmillan (1963).

[51] A.Rizzi and L.E.Eriksson, *Computation of flow around wings based on the Euler equations*, J. Fluid Mech., Vol. **148**(1984), pp. 45–71.

[52] P.L.Roe, *Approximate Riemann solvers, parameter vectors and difference schemes*, J. Comput. Phys., **43**(1981), pp. 357–372.

[53] ———,*Some contributions to the modelling of discontinuous flows*, Lectures in Applied Mathematics, Vol. **22**(1985), Am. Math. Soc.

[54] ———,*Characteristic-Based schemes for the Euler equations*, Ann. Rev. Fluid Mech., **18**(1986), pp. 337–365.

[55] ———,*Discrete models for the numerical analysis of time-dependent multidimensional gas dynamics*, J. Comput. Phys., **63**(1986), pp. 458–476.

[56] ———,*Finite-Volume methods for the compressible Navier-Stokes equations*, Numerical Methods in Laminar and Turbulent Flow (part 2), Editors. C.Taylor, W.G.Habashi, M.M.Hafez. Pineridge Press (1987).

[57] ———,*Sonic flux formulae*, To appear.

[58] P.L.Roe and J.Pike, *Efficient construction and utilisation of approximate Riemann solvers*, Computing methods in applied sciences and engineering, ed. R.Glowinski, J.L.Lions, **6**(1984): pp 499-518, Amsterdam: North Holland.

[59] D.Russell, *Shock dynamics of noninvasive fracturing of kidney stones*, Proceedings of the 15^{th} International Symposium on Shock Waves and Shock Tubes, Berkeley, CA, (1985), pp. 57–64.

[60] E.M.Schmidt and S.Duffy, *Noise from shock tube facilities*, AIAA Paper 85-0049 (1985).

[61] F.Selier, *Pseudo-stationary Mach reflection of shock waves*, Proceedings of the 15^{th} International Symposium on Shock Waves and Shock Tubes, Berkeley, CA, (1985), pp. 129–138.

[62] K.Stewartson, *The theory of laminar boundary layers in compressible fluids*, Oxford University Press (1964).

[63] J.L.Thomas and R.W.Walters, *Upwind relaxation for the Navier-Stokes equations*, AIAA Journal, Vol. **25**(1987), No. 4, pp. 527–534.

[64] E.F.Toro, *A weighted average flux method for hyperbolic conservation laws*, Proc. R. Soc. Lond., A **423**(1989), pp. 401–418.

[65] ———,*A fast Riemann solver with constant covolume applied to the random choice method*, Int. J. for Numer. Methods in Fluids, Vol. **9**(1989), pp 1145–1164.

[66] B. VAN LEER, *Towards the ultimate conservative difference scheme,*II. *Monotonicity and conservation combined in a second-order scheme*, J. Comput. Phys., **14**(1974), pp. 361–376.

[67] ———,*On the relation between the upwind-differencing schemes of Godunov, Engquist&Osher, and Roe*, SIAM J. SCI. STAT. COMPUT., Vol. **25**(1988), No.2, pp. 294–318.

[68] M.VINOKUR, *An analysis of finite-difference and finite-volume formulations of conservation laws*, J. Comput. Phys., **81**(1989), pp. 1–52.

[69] J. VON NEUMANN, *Proposal and analysis of a numerical method for the treatment of hydrodynamical problems*, Nat. Def. and Res. Com. Report AM-551, March (1944).

[70] J. VON NEUMANN AND R.D.RICHTMEYER, *A method for the numerical calculation of hydrodynamic shocks*, J. Appl. Phys., **21**(1950), pp. 232–257.

[71] J.C.T.WANG AND G.F.WIDHOPF, *Numerical simulation of blast flowfields using a high resolution TVD finite volume scheme*, Computers & Fluids, Vol. **18**(1990), No. 1, pp. 103–137.

[72] P.R.WOODWARD AND P.COLLELA, *The numerical simulation of two-dimensional fluid flow with strong shocks*, J. Comput. Phys., **54**(1984), pp. 115–173.

[73] J.Y.YANG, Y.LIU AND H.LOMAX, *Computation of shock wave reflection by circular cylinders*, AIAA Journal, Vol. **25**(1987), No. 5, pp. 683–689.

[74] H.C.YEE, R.F.WARMING AND A.HARTEN, *Implicit total variation diminishing (TVD) schemes for steady-state calculations*, J. Comput. Phys., **57**(1985), pp. 327–360.

Appendix A

Moving Shock Relationships

Consider a plane shock moving through a quiescent fluid with speed u_s, and let the two fluid states be referenced by the subscripts $_1$ and $_2$, see figure A.1. By convention a shock Mach number, M_s, is defined in terms of the shock speed and the speed of sound in the quiescent fluid,

$$M_s = \frac{u_s}{a_1}. \tag{A.1}$$

For a perfect gas,

$$a = \sqrt{\frac{\gamma p}{\rho}},$$

the following relationships may be derived[23] which connect the quiescent and disturbed fluid states,

$$\frac{p_2}{p_1} = \frac{2\gamma M_s^2 - (\gamma - 1)}{(\gamma + 1)} \tag{A.2}$$

$$\frac{\rho_2}{\rho_1} = \frac{\left[\left(\frac{\gamma+1}{\gamma-1}\right)\frac{p_2}{p_1} + 1\right]}{\left[\left(\frac{\gamma+1}{\gamma-1}\right) + \frac{p_2}{p_1}\right]} \tag{A.3}$$

$$\frac{u_2}{a_1} = M_s\left[1 - \frac{(\gamma - 1)M_s^2 + 2}{(\gamma + 1)M_s^2}\right]. \tag{A.4}$$

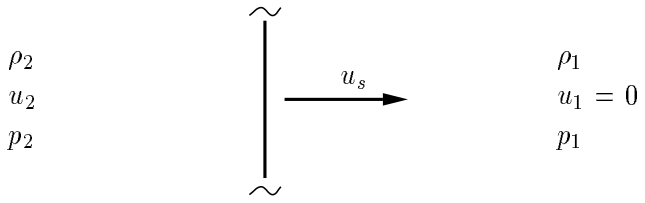

Figure A.1: Disturbed and quiescent fluid states.

Appendix B

Navier-Stokes Solver

The AMR algorithm has been used in conjunction with a two-dimensional Navier-Stokes solver to simulate a shock wave impinging on a laminar boundary layer, here we give a brief description for this solver. Strictly, the term solver is a misnomer for the AMR algorithm updates the flow solution. The solver merely provides the interface fluxes for the process described in §3.3.1.

B.1 Governing Equations

Using cartesian co-ordinates, the compressible Navier-Stokes equations for unsteady two-dimensional flows may be written as,

$$\frac{\partial}{\partial t}\begin{pmatrix}\rho\\ \rho u\\ \rho v\\ E_t\end{pmatrix}+\frac{\partial}{\partial x}\begin{pmatrix}\rho u\\ \rho u^2-\tau_{xx}\\ \rho uv-\tau_{xy}\\ (E_t+P)u-u\tau_{xx}-v\tau_{xy}+q_x\end{pmatrix}+\frac{\partial}{\partial y}\begin{pmatrix}\rho v\\ \rho vu-\tau_{yx}\\ \rho v^2-\tau_{yy}\\ (E_t+P)v-u\tau_{yx}-v\tau_{yy}+q_y\end{pmatrix}=0. \tag{B.1}$$

Where the components of the shear-stress tensor[1] and the heat-flux vector are given by,

$$\begin{aligned}\tau_{xx} &= \frac{2\mu}{3}\left[2\frac{\partial u}{\partial x}-\frac{\partial v}{\partial y}\right] & \tau_{yy} &= \frac{2\mu}{3}\left[2\frac{\partial v}{\partial y}-\frac{\partial u}{\partial x}\right]\\ \tau_{xy} &= \mu\left[\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\right] & \tau_{xy} &= \mu\left[\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\right]\\ q_x &= -\lambda\frac{\partial T}{\partial x} & q_y &= -\lambda\frac{\partial T}{\partial y}.\end{aligned} \tag{B.2}$$

[1] Assumes Stokes' hypothesis for bulk viscosity.

To complete the equations governing the flow we give the equation of state and the constitutive relations for μ and λ. A perfect gas is assumed hence,

$$P = (\gamma - 1)(E_t - \tfrac{1}{2}\rho[u^2 + v^2]). \tag{B.3}$$

Viscosity is assumed to vary with temperature in accordance with Sutherland's law,

$$\mu = \mu_\infty \left(\frac{T}{T_\infty}\right)^{3/2} \left(\frac{T_\infty + 110}{T + 110}\right), \tag{B.4}$$

and the Prandtl number is assumed constant so,

$$\lambda = \frac{\mu c_p}{Pr}. \tag{B.5}$$

B.2 Non-Dimensionalization

The governing equations are non-dimensionalized in the following manner,

$$\begin{array}{lllllll} \bar{x} & = & \dfrac{x}{L} & \bar{y} & = & \dfrac{y}{L} & \bar{t} & = & \dfrac{t}{L/U_\infty} \\ \bar{u} & = & \dfrac{u}{U_\infty} & \bar{v} & = & \dfrac{v}{U_\infty} & \bar{\rho} & = & \dfrac{\rho}{\rho_\infty} \\ \bar{E}_t & = & \dfrac{E_t}{\rho_\infty U_\infty^2} & \bar{P} & = & \dfrac{P}{\rho_\infty U_\infty^2} & \bar{T} & = & \dfrac{T}{T_\infty}. \end{array} \tag{B.6}$$

Where the non-dimensional variables are denoted by an over bar, free stream conditions are denoted by the subscript $_\infty$, and L is the reference length used in the Reynolds number,

$$Re = \frac{\rho_\infty U_\infty L}{\mu_\infty}. \tag{B.7}$$

Splitting the flux vectors into their viscous and inviscid components, the non-dimensionalized Navier-Stokes equations may be written in the form,

$$\frac{\partial \mathbf{W}}{\partial \bar{t}} + \frac{\partial}{\partial \bar{x}}\left[\mathbf{F}^I + \mathbf{F}^V\right] + \frac{\partial}{\partial \bar{y}}\left[\mathbf{G}^I + \mathbf{G}^V\right] = 0. \tag{B.8}$$

Where the vectors $\mathbf{W}$, $\mathbf{F}^I$, $\mathbf{G}^I$, $\mathbf{F}^V$ and $\mathbf{G}^V$ are given by,

$$\mathbf{W} = \begin{pmatrix} \bar{\rho} \\ \bar{\rho}\bar{u} \\ \bar{\rho}\bar{v} \\ \bar{E}_t \end{pmatrix}$$

$$\mathbf{F}^I = \begin{pmatrix} \bar{\rho}\bar{u} \\ \bar{\rho}\bar{u}^2 \\ \bar{\rho}\bar{u}\bar{v} \\ (\bar{E}_t + \bar{P})\bar{u} \end{pmatrix} \qquad \mathbf{G}^I = \begin{pmatrix} \bar{\rho}\bar{v} \\ \bar{\rho}\bar{v}\bar{u} \\ \bar{\rho}\bar{v}^2 \\ (\bar{E}_t + \bar{P})\bar{v} \end{pmatrix} \tag{B.9}$$

$$\mathbf{F}^V = \begin{pmatrix} 0 \\ -\bar{\tau}_{xx} \\ -\bar{\tau}_{xy} \\ -\bar{u}\bar{\tau}_{xx} - \bar{v}\bar{\tau}_{xy} + \bar{q}_x \end{pmatrix} \qquad \mathbf{G}^V = \begin{pmatrix} 0 \\ -\bar{\tau}_{yx} \\ -\bar{\tau}_{yy} \\ -\bar{u}\bar{\tau}_{yx} - \bar{v}\bar{\tau}_{yy} + \bar{q}_y \end{pmatrix}.$$

The components of the shear-stress tensor and the heat-flux vector are now given by,

$$\begin{aligned} \bar{\tau}_{xx} &= \frac{2\bar{\mu}}{3Re}\left[2\frac{\partial \bar{u}}{\partial \bar{x}} - \frac{\partial \bar{v}}{\partial \bar{y}}\right] & \bar{\tau}_{yy} &= \frac{2\bar{\mu}}{3Re}\left[2\frac{\partial \bar{v}}{\partial \bar{y}} - \frac{\partial \bar{u}}{\partial \bar{x}}\right] \\ \bar{\tau}_{xy} &= \frac{\bar{\mu}}{Re}\left[\frac{\partial \bar{u}}{\partial \bar{y}} + \frac{\partial \bar{v}}{\partial \bar{x}}\right] & \bar{\tau}_{yx} &= \frac{\bar{\mu}}{Re}\left[\frac{\partial \bar{u}}{\partial \bar{y}} + \frac{\partial \bar{v}}{\partial \bar{x}}\right] \\ \bar{q}_x &= \frac{-\bar{\mu}}{(\gamma-1)M_\infty^2 RePr}\frac{\partial \bar{T}}{\partial \bar{x}} & \bar{q}_y &= \frac{-\bar{\mu}}{(\gamma-1)M_\infty^2 RePr}\frac{\partial \bar{T}}{\partial \bar{y}}. \end{aligned} \tag{B.10}$$

Where M_∞ is the free stream Mach number,

$$M_\infty = \frac{U_\infty}{\sqrt{\frac{\gamma P_\infty}{\rho_\infty}}}. \tag{B.11}$$

The equation of state has become,

$$\bar{P} = \frac{\bar{\rho}\bar{T}}{\gamma M_\infty^2}, \tag{B.12}$$

and Sutherland's law,

$$\bar{\mu} = \bar{T}^{3/2}\left(\frac{1 + 110/T_\infty}{\bar{T} + 110/T_\infty}\right). \tag{B.13}$$

B.3 Method of Discretization

The numerical flux across an isolated interface is calculated in two parts. First, Roe's scheme is used to find the inviscid flux-component. Second, central differencing is used to find the viscous cum heat conduction components. A comprehensive description of

how Roe's scheme may be used to find the inviscid flux component is given in chapter 5. So, here we only consider the method of calculation for the viscous and heat conduction terms.

First, consider the mapping from (x, y) co-ordinates to (ξ, η) co-ordinates. The following metric identities may be found,

$$\begin{array}{rclrcl} \dfrac{\partial \xi}{\partial x} &=& \dfrac{1}{J}\dfrac{\partial y}{\partial \eta} & \dfrac{\partial \xi}{\partial y} &=& -\dfrac{1}{J}\dfrac{\partial x}{\partial \eta} \\ \dfrac{\partial \eta}{\partial x} &=& -\dfrac{1}{J}\dfrac{\partial y}{\partial \xi} & \dfrac{\partial \eta}{\partial y} &=& \dfrac{1}{J}\dfrac{\partial x}{\partial \xi} \end{array} . \tag{B.14}$$

Where J is the Jacobian of the transformation and is given by,

$$J = \left(\frac{\partial x}{\partial \xi}\right)\left(\frac{\partial y}{\partial \eta}\right) - \left(\frac{\partial x}{\partial \eta}\right)\left(\frac{\partial y}{\partial \xi}\right). \tag{B.15}$$

Using these identities in conjunction with the chain rule, the gradients $\frac{\partial()}{\partial x}$ and $\frac{\partial()}{\partial y}$ may be written as,

$$\begin{aligned} \frac{\partial()}{\partial x} &= +\frac{1}{J}\left(\frac{\partial y}{\partial \eta}\right)\frac{\partial()}{\partial \xi} - \frac{1}{J}\left(\frac{\partial y}{\partial \xi}\right)\frac{\partial()}{\partial \eta} \\ \frac{\partial()}{\partial y} &= -\frac{1}{J}\left(\frac{\partial x}{\partial \eta}\right)\frac{\partial()}{\partial \xi} + \frac{1}{J}\left(\frac{\partial x}{\partial \xi}\right)\frac{\partial()}{\partial \eta}. \end{aligned} \tag{B.16}$$

Now consider the interface separating the cell (i, j) from the cell $(i+1, j)$. The viscous and heat conduction terms for the midpoint of this interface, the point $(i+\frac{1}{2}, j)$, are found by discretizing equations B.14–B.16 using central differencing; ξ being associated with i and η being associated with j.

For convenience we introduce the *cell-vertex* difference operators $\Delta^{(i)}$ and $\Delta^{(j)}$, together with the *cell-centre* difference operators $\delta^{(i)}$ and $\delta^{(j)}$,

$$\begin{aligned} \Delta^{(i)}()_{i+\frac{1}{2},j} &= \frac{1}{4}\left[()_{i+\frac{3}{2},j+\frac{1}{2}} + ()_{i+\frac{3}{2},j-\frac{1}{2}} - ()_{i-\frac{1}{2},j+\frac{1}{2}} - ()_{i-\frac{1}{2},j-\frac{1}{2}}\right] \\ \Delta^{(j)}()_{i+\frac{1}{2},j} &= ()_{i+\frac{1}{2},j+\frac{1}{2}} - ()_{i+\frac{1}{2},j-\frac{1}{2}} \\ \delta^{(i)}()_{i+\frac{1}{2},j} &= ()_{i+1,j} - ()_{i,j} \\ \delta^{(j)}()_{i+\frac{1}{2},j} &= \frac{1}{4}\left[()_{i,j+1} + ()_{i+1,j+1} - ()_{i,j-1} - ()_{i+1,j-1}\right]. \end{aligned} \tag{B.17}$$

Using these difference operators the gradients $\frac{\partial()}{\partial x}$ and $\frac{\partial()}{\partial y}$ for the point $(i+\frac{1}{2}, j)$ may be written,

$$\begin{aligned} \left.\frac{\partial()}{\partial x}\right|_{i+\frac{1}{2},j} &= \Delta^{(x)}()_{i+\frac{1}{2},j} = +\frac{\Delta^{(j)}y_{i+\frac{1}{2},j}\delta^{(i)}()_{i+\frac{1}{2},j} - \Delta^{(i)}y_{i+\frac{1}{2},j}\delta^{(j)}()_{i+\frac{1}{2},j}}{\Delta^{(i)}x_{i+\frac{1}{2},j}\Delta^{(j)}y_{i+\frac{1}{2},j} - \Delta^{(j)}x_{i+\frac{1}{2},j}\Delta^{(i)}y_{i+\frac{1}{2},j}} \\ \left.\frac{\partial()}{\partial y}\right|_{i+\frac{1}{2},j} &= \Delta^{(y)}()_{i+\frac{1}{2},j} = -\frac{\Delta^{(j)}x_{i+\frac{1}{2},j}\delta^{(i)}()_{i+\frac{1}{2},j} - \Delta^{(i)}x_{i+\frac{1}{2},j}\delta^{(j)}()_{i+\frac{1}{2},j}}{\Delta^{(i)}x_{i+\frac{1}{2},j}\Delta^{(j)}y_{i+\frac{1}{2},j} - \Delta^{(j)}x_{i+\frac{1}{2},j}\Delta^{(i)}y_{i+\frac{1}{2},j}}. \end{aligned} \tag{B.18}$$

Thus we can write the shear stress and heat conduction components for the point $(i+\frac{1}{2}, j)$ as,

$$\begin{aligned}
\bar{\tau}_{xy}|_{i+\frac{1}{2},j} &= \bar{\mu}_{i+\frac{1}{2},j}\left(\Delta^{(y)}\bar{u}_{i+\frac{1}{2},j} + \Delta^{(x)}\bar{v}_{i+\frac{1}{2},j}\right) \\
\bar{\tau}_{xx}|_{i+\frac{1}{2},j} &= \frac{2\bar{\mu}_{i+\frac{1}{2},j}}{3Re}\left(2\Delta^{(x)}\bar{u}_{i+\frac{1}{2},j} - \Delta^{(y)}\bar{v}_{i+\frac{1}{2},j}\right) \\
\bar{\tau}_{yy}|_{i+\frac{1}{2},j} &= \frac{2\bar{\mu}_{i+\frac{1}{2},j}}{3Re}\left(2\Delta^{(y)}\bar{v}_{i+\frac{1}{2},j} - \Delta^{(x)}\bar{u}_{i+\frac{1}{2},j}\right) \\
\bar{q}_{x}|_{i+\frac{1}{2},j} &= \frac{-\bar{\mu}_{i+\frac{1}{2},j}}{(\gamma-1)M_\infty^2 RePr}\Delta^{(x)}\bar{T}_{i+\frac{1}{2},j} \\
\bar{q}_{y}|_{i+\frac{1}{2},j} &= \frac{-\bar{\mu}_{i+\frac{1}{2},j}}{(\gamma-1)M_\infty^2 RePr}\Delta^{(y)}\bar{T}_{i+\frac{1}{2},j}
\end{aligned} \tag{B.19}$$

Where the coefficient of viscosity is calculated using the average of the cell centre temperatures,

$$\bar{\mu}_{i+\frac{1}{2},j} = \bar{\mu}\left(\frac{\bar{T}_{i,j} + \bar{T}_{i+1,j}}{2}\right). \tag{B.20}$$

Similarly, values for $\bar{u}_{i+\frac{1}{2},j}$ and $\bar{v}_{i+\frac{1}{2},j}$ are found by averaging the cell centre values,

$$\begin{aligned}
\bar{u}_{i+\frac{1}{2},j} &= \frac{1}{2}(\bar{u}_{i,j} + \bar{u}_{i+1,j}) \\
\bar{v}_{i+\frac{1}{2},j} &= \frac{1}{2}(\bar{v}_{i,j} + \bar{v}_{i+1,j})
\end{aligned} \tag{B.21}$$

Given equations B.20–B.22 the evaluation of the interface fluxes $\mathbf{F}^V_{i+\frac{1}{2},j}$ and $\mathbf{G}^V_{i+\frac{1}{2},j}$ follows straightforwardly. A similar procedure is used to evaluate the fluxes $\mathbf{F}^V_{i,j+\frac{1}{2}}$ and $\mathbf{G}^V_{i,j+\frac{1}{2}}$, but for brevity no details are given.

Note, the discretization of the viscous terms has the second-order accuracy associated with central differencing. However, Roe[56] has argued that if the combination of the inviscid and viscous flux components is to yield a second-order accurate integration scheme then the inviscid flux must be evaluated using equation 5.26 rather than equation 5.21.

B.4 Validation

A popular test problem with which to validate a Navier-Stokes code is the laminar boundary layer over a flat plate, taking M_∞ as 0.5 and Re as 10,000; the results being compared with the Blasius solution. We have computed this problem using the mesh shown in figure B.1. Note, this mesh is not drawn to scale. Although many workers claim good

agreement between their numerical results and the Blasius solution they have in fact either ignored or missed certain subtleties of the validation process. These subtleties arise because the computed flow is not incompressible. Indeed for $M_\infty = 0.5$, $\rho_\infty/\rho_{o_\infty} \approx 0.89$! A comprehensive account of how the compressible boundary layer equations may be manipulated so as to yield the Blasius solution is given by Stewartson[62]. Here we simply note that for this validation test to make sense then the calculation should be performed with a viscosity law $\bar{\mu} = \bar{T}$ and a Prandtl number of unity. Furthermore, before the computed velocity profiles are compared with the Blasius solution they should be transformed using the Howarth-Dorodnytsin stretching transformation. For the last five cells along the plate, figure B.2 shows a comparison between the computed velocity profiles (ARROWS) and profiles computed from the Blasius solution (SOLID LINES). Pleasingly, there are no discernible differences between the two sets of profiles. The amount of compression due to the Howarth-Dorodnytsin transformation, and hence the deviation from incompressible flow, can be gauged by the fact that before applying the transformation the velocity vectors emanated from cell centres. Whilst this deviation is small it is nevertheless appreciable and so cannot be ignored by any credible validation process.

This test problem cannot be viewed as a comprehensive check of a Navier-Stokes solver, but in the context of our study it is sufficient to add credence to the results presented in §6.3.

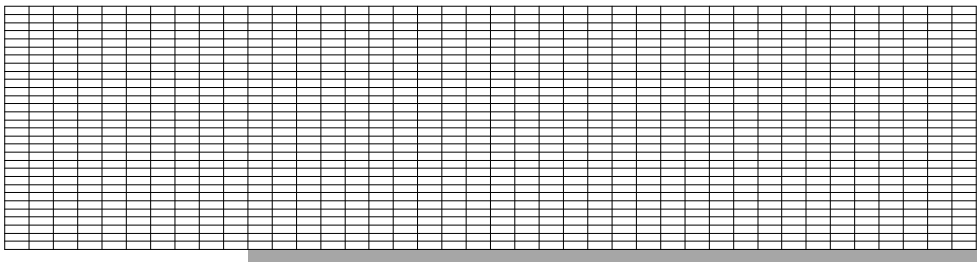

Figure B.1: Computational mesh for the Blasius test problem; Y co-ordinates magnified 5 times.

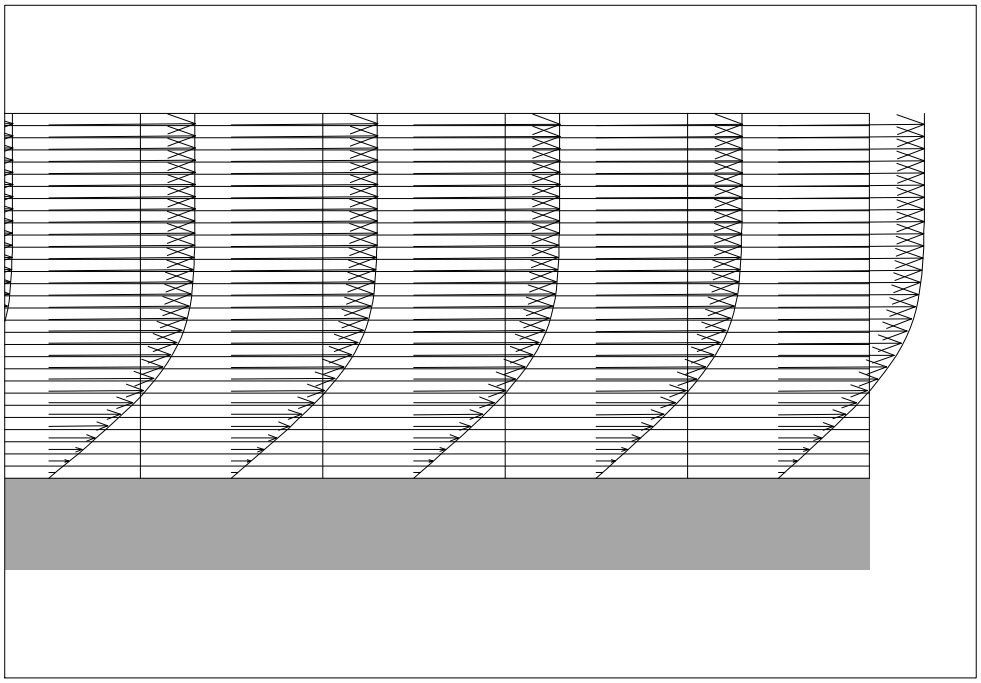

Figure B.2: Comparison of the computed velocity profiles (ARROWS) with the profiles calculated from the Blasius solution (SOLID LINES).

Appendix C

Implementation of Recursion using FORTRAN

A procedure is recursive if it calls itself either directly or indirectly. The following *pseudo-code* demonstrates direct recursion.

```
Procedure Factorial(n)
  if n = 0 {
    return 1
  }
  else {
    return n*Factorial(n − 1)
  }
End Procedure
```

Figure C.1: Recursive *pseudo-code* to calculate $n!$.

Programming languages that support recursion are stack based, that is procedure parameters, local variables and procedure return points are stored on a LIFO[1] stack rather than in static storage locations. Every time a procedure is called: the return point is placed on the stack; the procedure parameters are placed on the stack; space is set aside on the stack for use by local variables within the procedure. While every time the procedure terminates: the local variable stack frame is deleted; the return point is fetched from the stack; if required, a result is placed on the stack for the calling procedure. For example, the various states taken by the stack during the execution of **Factorial**(3) are shown in figure C.2. Note, #100 is a label pointing to the statement which follows the procedural

[1] Last In First Out

call **Factorial**(3), and #200 is a label pointing to the End Procedure statement contained by the procedure **Factorial**.

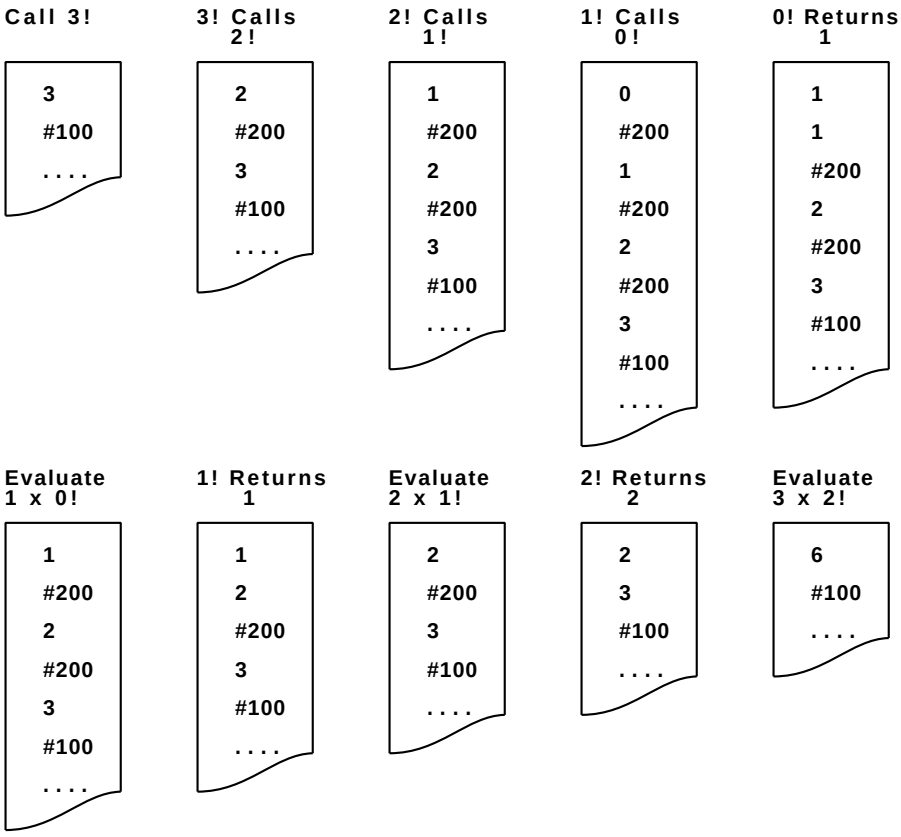

Figure C.2: Stack states during the evaluation of 3!

Note, it is always possible to unroll a recursive procedure into a non-recursive one which can then be routinely coded in FORTRAN. For trivial cases such as calculating $n!$ this approach is ideal. But, for more complicated procedures the resultant code will lose much of the clarity of the original and an alternative approach is to be preferred. The recursive procedure is embedded within a FORTRAN subroutine and recursive calls to this procedure are simulated by explicitly manipulating a stack. Figure C.3 shows how the recursive procedure **Factorial** may be coded in this manner. A thorough description of how recursive programming languages implicitly manipulate a stack is given by Bornat[10].

```
      INTEGER N,SPTR,STACK(30),RESULT

      SPTR = 0                          initialise stack pointer
      LRETURN = 1                       set return label
      READ(5,*) N                       set parameter for function call
      GO TO 500                         call factorial function

100   CONTINUE                          return point from above call
      RESULT = STACK(SPTR)              result at top of stack


      WRITE(6,*) RESULT                 print result
      STOP



500   CONTINUE                          entry point for factorial function
      STACK(SPTR+1) = LRETURN           save return label on stack
      STACK(SPTR+2) = N                 save parameter on stack
      SPTR = SPTR+2                     bump pointer to top of stack
      IF(N.EQ.0) THEN
          SPTR = SPTR–1                 return 0!
          LRETURN = STACK(SPTR)         get return label
          STACK(SPTR) = 1               place 0! on stack
          GO TO (100,200),LRETURN       return from call
      ELSE
          N = STACK(SPTR)–1             get parameter for function call
          LRETURN = 2                   set return label
          GO TO 500                     call factorial function
200       CONTINUE                      return point from above call
          SPTR = SPTR–2                 adjust stack pointer
          LRETURN = STACK(SPTR)         get return label and evaluate n*(n-1)!
          STACK(SPTR) = STACK(SPTR+1)*STACK(SPTR+2)
          GO TO (100,200),LRETURN       return from call
      ENDIF
      END
```

Figure C.3: FORTRAN code simulating recursion.

Appendix D

Contour Plotting

In order to view the results produced by the AMR algorithm it has been necessary to write a customized contouring package. The contouring procedure and some consequences of it are explained below. This information should help the reader interpret the contour plots presented earlier in this thesis.

The AMR algorithm stores the conserved variable vector **W** for a hierarchical set of nested meshes. The contouring procedure takes advantage of this hierarchical structure by contouring each grid level independently. This is possible because underlying every fine mesh solution there is also a coarse mesh solution. The grid levels are contoured in order, starting at level 0. Prior to contouring each grid the area it covers in the computational domain is blanked out. This blanking procedure coupled with the order in which grid levels are contoured ensures that, wherever possible, fine mesh contours replace the less accurate coarse mesh contours. Moreover, once the dummy cells for each mesh have been primed, the meshes on a grid level may be contoured independently. The order in which individual meshes, on each grid level, are contoured is not important.

Every mesh within the grid structure forms a logically rectangular unit, and has cell averaged values of the conserved variable vector **W** stored for each of its cells. These cell averaged values may also be viewed as the point value taken by the solution at the cell centre. Here we take the cell centre $(\bar{x}, \bar{y})$ to be the arithmetic average of the four vertices forming the cell. The process of contouring a function ϕ amounts to fitting a surface $\phi(x, y)$ through the known values $\phi(\bar{x}, \bar{y})$ at these cell centres, see figure D.1.

Given the values of ϕ at the vertices A,B,C and D then the values of ϕ at the midpoints of the sides must, for a piecewise planar distribution of ϕ, be the average of the values of ϕ at the two end points. It is easily shown that the surface abcd is also planar. Note abcd forms a parallelogram, see figure D.2, so if abcd is planar then $\phi_O = \frac{1}{2}(\phi_a + \phi_c) = \frac{1}{2}(\phi_b + \phi_d)$. But, $\phi_a + \phi_c = \phi_A + \phi_B + \phi_C + \phi_D = \phi_b + \phi_d$, therefore abcd must be planar.

This simple procedure is consistent with the methodology behind the spatial discretization of the AMR algorithm. A more complicated procedure that attempted to fit contours over a global region rather than a local region would inevitably result in some additional smearing of flow discontinuities. Contour plots drawn by the above procedure exhibit the following characteristics:

- Contours of discontinuities oblique to a mesh carry a high frequency ripple.
- Contours across fine-fine mesh interfaces are continuous.
- Contours across fine-coarse mesh interfaces may be discontinuous.
- Contours on a coarse mesh of a projected fine mesh discontinuity can lie outside the fine mesh.

All these characteristics can be seen in figure D.4, which shows the density contours for an oblique shock wave formed by the steady supersonic flow over a wedge. The last characteristic may seem surprising until one considers the contours for the one-dimensional discontinuity shown in figure D.3. The confusing contour overspill shown in figure D.4 is easily explained when the contours for the level 0 grid are plotted separately, see figure D.5. In order to avoid confusion by contour overspills, some of the contour plots in this thesis do not draw contours for the half coarse cell bordering a fine mesh.

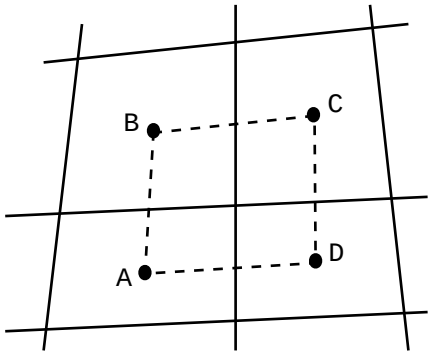

Figure D.1: Quadrilateral patch formed by neighbouring cell centres.

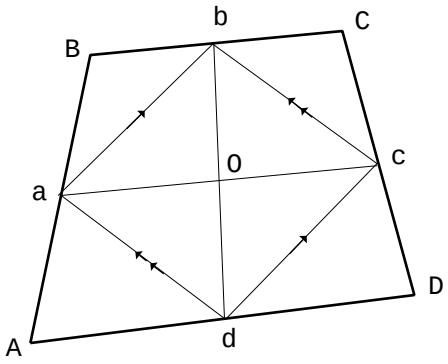

Figure D.2: Piecewise planar surface over a quadrilateral patch.

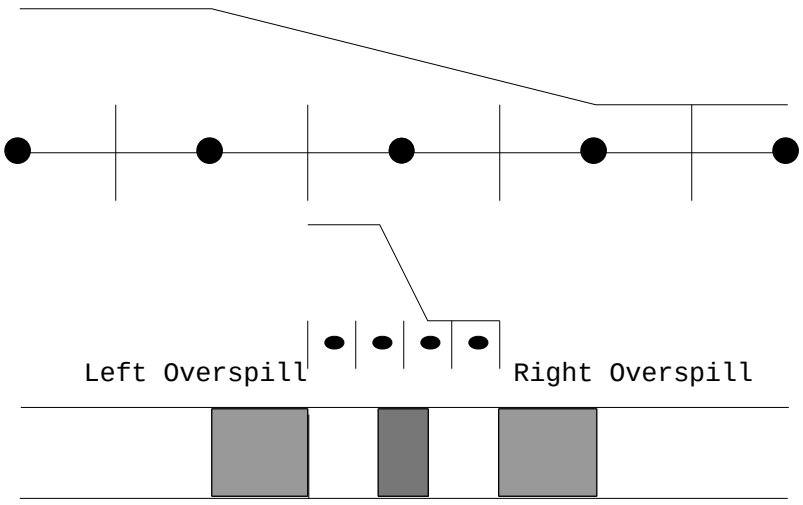

Figure D.3: Contour overspill near a refined discontinuity.

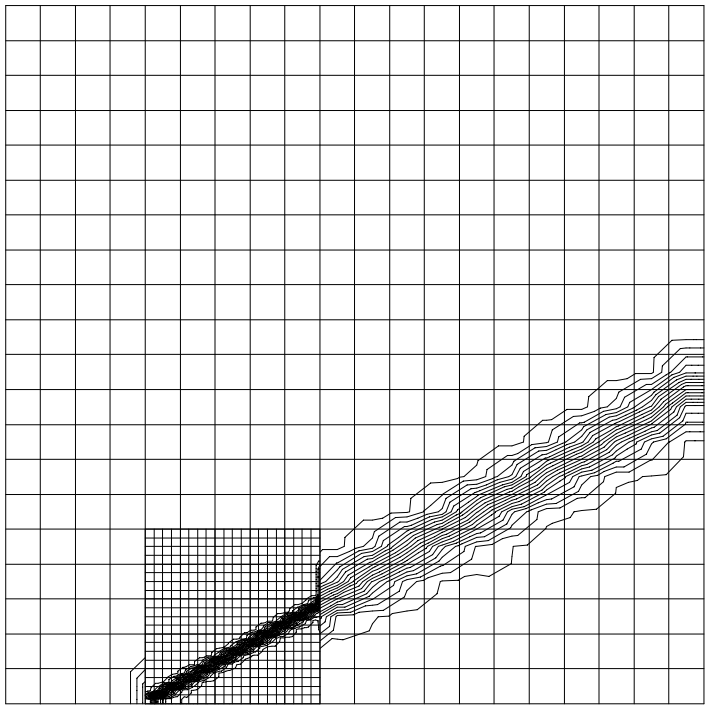

Figure D.4: Density contours for steady supersonic flow over a wedge.

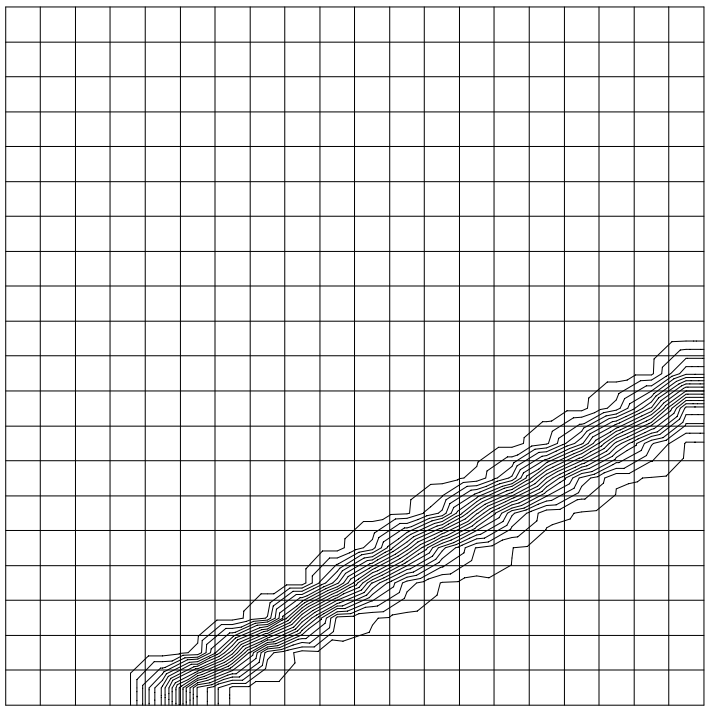

Figure D.5: Density contours for level 0.