MULTISTAGE TURBOMACHINERY FLOW CALCULATION PROGRAM MULTALL_OPEN

Written for MULTALL_OPEN. February 2017

Updated for MULTALL-OPEN-17.5. August 2017.

DEVELOPMENT HISTORY

Program **MULTALL** is a 3D multistage turbomachinery flow calculation program developed by Prof John Denton at the Whittle Lab, Cambridge, UK, over many years. In fact its origins go back to when the author was working at the CEGB Marchwood Engineering Labs around 1973. The program is based on the finite volume time marching method. The original version in the early 1970's was inviscid and for a single blade row. It used the "opposed difference" numerical scheme and an overlapping grid system. It rapidly became popular because it was much faster than previous 3D methods. The extension to multiple blade rows, using the mixing plane model, took place around 1978. Around 1980 the grid was changed to use cell corner storage with no overlapping, as still used today. Around the same time multigrid was added and spatially varied time steps were used, giving a significant speed up in run times. The multigrid differs from the conventional model in that it is simpler, faster and arbitrary block sizes can be used. Several different versions were developed around this time to include splitter blades, part-span shrouds, cooling flows, bleed flows, etc.

In the mid 1980's simple models of viscous effects were added. The first of these simply added the displacement thickness of a very simply calculated boundary layer to the blade shape. Then the body force model of viscous effects was developed, with the body forces initially calculated by a thin shear layer approximation to the Navier-Stokes equations. The model used a mixing length turbulence model and wall functions for the surface shear stresses. The body force model itself is not an approximation and has the major advantage that the body forces only need to be updated every few (typically 5 or 10) time steps making it much faster than conventional N-S methods. The original simple model, subroutine LOSS, proved remarkably accurate and is still widely used today.

The solution algorithm was changed to use the "scree" scheme around 2000, this reduced the level of numerical viscosity and greatly helped with the calculation of low Mach number flows and reversed flows. Cooling and bleed flows and an allowance for surface roughness were incorporated into the basic program rather than into separate versions. An allowance for real gas flows was also included in the basic program and a separate version, MSTEAM, not described here, with steam properties was developed. Several improvements to the mixing plane model were also made around this time. Two different viscous models were added around 2010, the Spalart-Almaras model in subroutine SPAL_LOSS and an improved mixing length model in subroutine NEW_LOSS.

Around 2013 option to use an improved version of the "scree" scheme, the "SSS" scheme was introduced, allowing larger CFL numbers but being less robust than the basic "scree" scheme. In most

cases the basic “scree” scheme is still preferred. An option to perform quasi-3D blade-to-blade calculations on a stream surface using only a single cell in the spanwise direction was introduced in 2015 and a throughflow methiod, using a single cell in the pitchwise direction in 2016.

An option to run at very low Mach numbers, or even with fully incompressible flow was first added in the 1990’s and was updated in 2016.

The current version MULTALL_OPEN has been considerably tidied up from the previous version, MULTALL_15.2, but the only new feature that has been added is the ability to perform an axisymmetric throughflow calculation using only a single cell in the pitchwise direction. The input data has been considerably modified to try to make it more systematic and this makes it no longer compatible with data sets for previous versions. However, the option to use a data set closely compatible with MULTALL_15.2 set has been left in the code so that previous users can continue to use their old data sets with minimum modification. A program named CONVERT.F, to convert MULTALL_15 data sets to MULTALL_OPEN data sets with NEW_READIN format is available.

The program is for steady flow only. Unsteady flow can be calculated by the author’s other programs UNSTREST and TBLOCK which are not freely available.

OVERVIEW

MULTALL is a three dimensional flow calculation program which has been specifically developed for turbomachinery. The program is written in standard Fortran77 and should run on any computer with a Fortran compiler. The only non-standard feature is the call to the timing routine, MCLOCK, which is specific to the gfortran and g77 Linux compilers. This should be removed or replaced by an equivalent call if using other compilers.

The program can be used for axial, mixed or radial flow turbomachinery with no limitation, apart from computer storage requirements, on the number of blade rows calculated. However, it only calculates the main flow path so the annulus boundaries have to be surfaces of revolution and split flow paths, e.g. splitter blades or part span shrouds, cannot be included. The interface between blade rows is modeled using a highly developed mixing plane model.

The program is designed to be relatively simple and is also relatively fast with run times of order 15 minutes per blade row on a single processor. Tip leakage flow can be predicted using the pinched tip model and shrouded blade leakages using a source-sink model. Although the basic scheme works down to Mach numbers of order 0.15, low Mach number and incompressible flow options are provided. A source-sink model may also be used to predict cooling flows and bleed flows. Turbulence modeling is either by a thin shear layer mixing length model, a full Navier-Stokes mixing length model or the Spalart-Almaras model. Boundary layer transition may be either specified or modeled and the effects of surface roughness can be predicted. A limited redesign of the blade sections can be performed within the program.

Detailed instructions for compiling and running the program are given at the end of this document.

SOLUTION ALGORITHM

MULTALL uses the “scree” algorithm instead of the previous "Opposed Difference Scheme" which was used up to MULTIP75. The “scree” scheme is an extremely simple method that is completely second order accurate in space. The primary flow variables F (where F = ρ, ρE, ρV_x, ρV_r, or $\rho r V_t$) are updated on every timestep using

$$\Delta F = \left(2\frac{\partial F}{\partial T}\right)^{n} - \frac{\partial F}{\partial T}\Big)^{n-1}\Big)\Delta t$$

where n is the time step level. This may be thought of as an extrapolation of the rate of change to the end of the time step. This involves only a single flux evaluation per time step and so is much faster than multi-step schemes. The scheme is only first order accurate in time but this is not important for steady calculations. The “scree” scheme can be used with much lower values of artificial viscosity (smoothing) than most other algorithms and does not need any special treatment or loss of accuracy to handle reversed axial velocities. The scheme is also better than most others at very low Mach numbers and solutions can be obtained for effectively incompressible flow at Mach numbers of order 0.15.

The maximum stable timestep with the “scree” scheme is that giving a CFL number around 0.5, but as a safety factor it is more usual to set CFL = 0.4. Compared to all previous versions of this code he CPU time per point per timestep is only slightly reduced but the number of timesteps required for convergence is generally significantly reduced. Convergence is generally much more continuous than with the opposed difference scheme and, once over the initial transients, the graph of log(residual) vs time step number soon becomes a straight line.

The “super_scree” algorithm, available in TBLOCK, never worked well on real problems, although it works well on a uniform grid. It was found to be sensitive to the multigrid on non-uniform grids. It was then realised that the “super_scree” algorithm could be approximated without using the additional storage for the derivatives at the N-2 timestep by setting

$$\Delta = \Delta t/V_{ol}\ (\ F_1 * R_n\ + F_2\ R_{n-1}\ )$$
$$R_{n-1} = R_n + F_3 * R_{n-1}$$

Where Δ is the change applied, Δt is the time step, V_{ol} the cell volume and R_n is the residual at step n. F_1, F_2 and F_3 are constants.

This makes the second term into a geometric series of past residuals and to make the sum of all coefficients = 1.0, so that its time steps are comparable to the “scree” scheme, we must set

$$F_1 + F_2/(1-F_3) = 1.0\ .$$

It is found that the combination

$$F_1 = 2.0,\ F_2 = -1.65,\ F_3 = -0.65$$

is close to optimum and allows CFL numbers up to about 0.7 compared to 0.4 for the "scree" scheme. Higher values, up to 0.9, can sometimes be used. The combination $F_2/(1-F_3)$ is referred to as the effective value of F_2, F_{2eff} , and it is this value that must be input in the data file. i.e. the typical input value of F_{2eff} = -1.0 . This is called the "SSS" (Simple Super Scree) scheme.

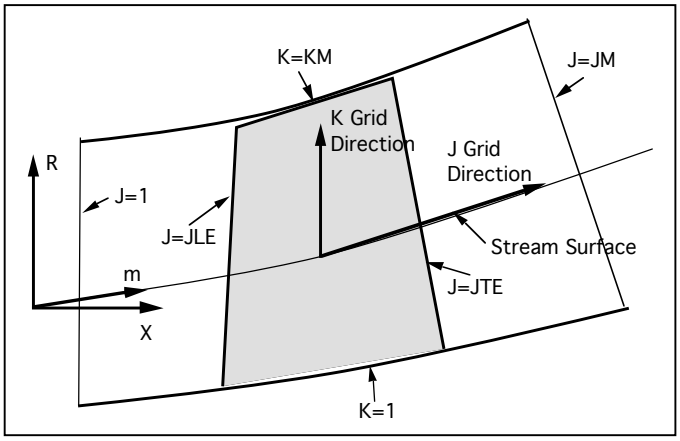
It was also found that the CFL number could be increased slightly further by the use of residual averaging, i.e. smoothing the values of R_n . A smoothing subroutine, SMOOTH_RESID, is included to perform this, although its use is optional. A single smoothing with smoothing factor 0.4 allows a small increase in CFL at the expense of about 7.5% increase in run time per step. However, a major advantage is an increase in robustness. The program no longer fails when the stable CFL number is exceeded. Instead the residuals at the unstable points oscillate at high frequency about a steady average. Although the residuals may not decrease to low levels the resulting solutions appear identical to fully converged ones. CFL values up to 0.9 can sometimes be used in such cases but it is safer to choose 0.7 as a standard value.

GRID

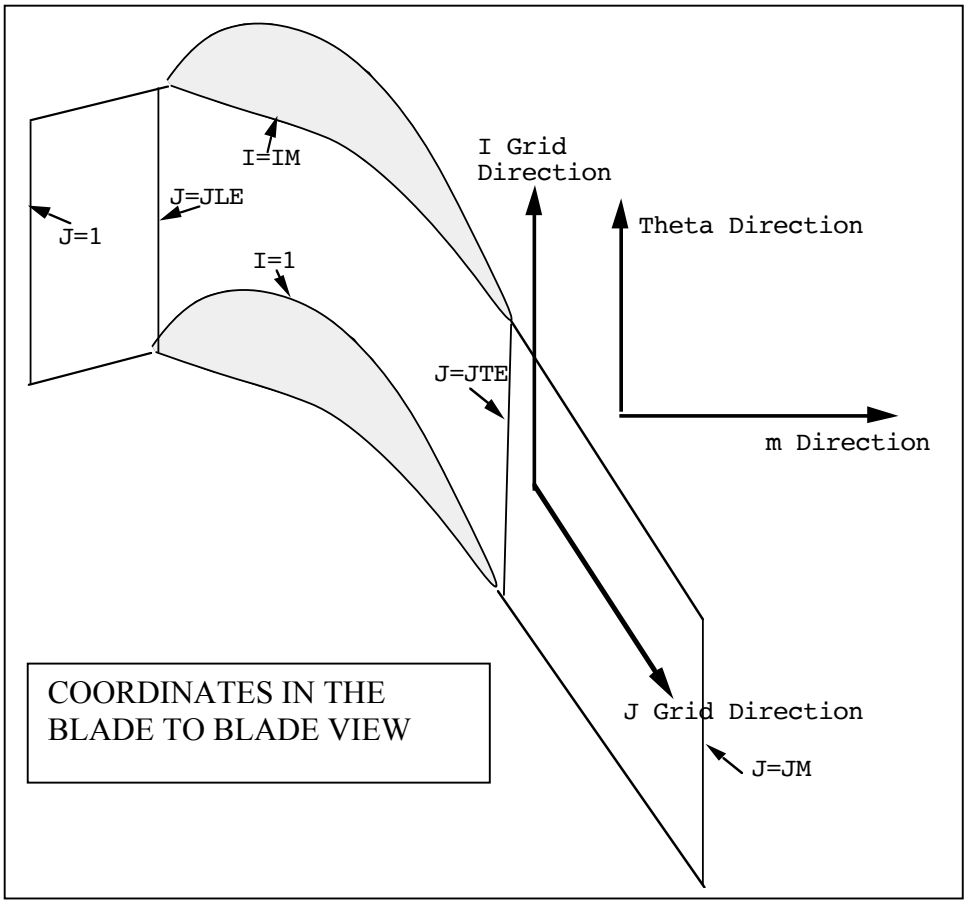
The program uses a standard "H" grid composed of pitchwise (J and K constant, I varies) grid lines, streamwise (I and K constant, J varies) grid lines and quasi-orthogonal (I and J = constant, K varies) grid lines. The simple H grid greatly simplifies grid generation, the application of the periodic boundary conditions and the modeling of the mixing planes. It inevitably leads to highly sheared cells for staggered blade rows but experience is that the numerical errors associated with this (as judged by the entropy change in inviscid flow) are negligible when "viscous" grids with more than about 30 points across the pitch and span, and with close meridional spacing around the leading edge, are used.

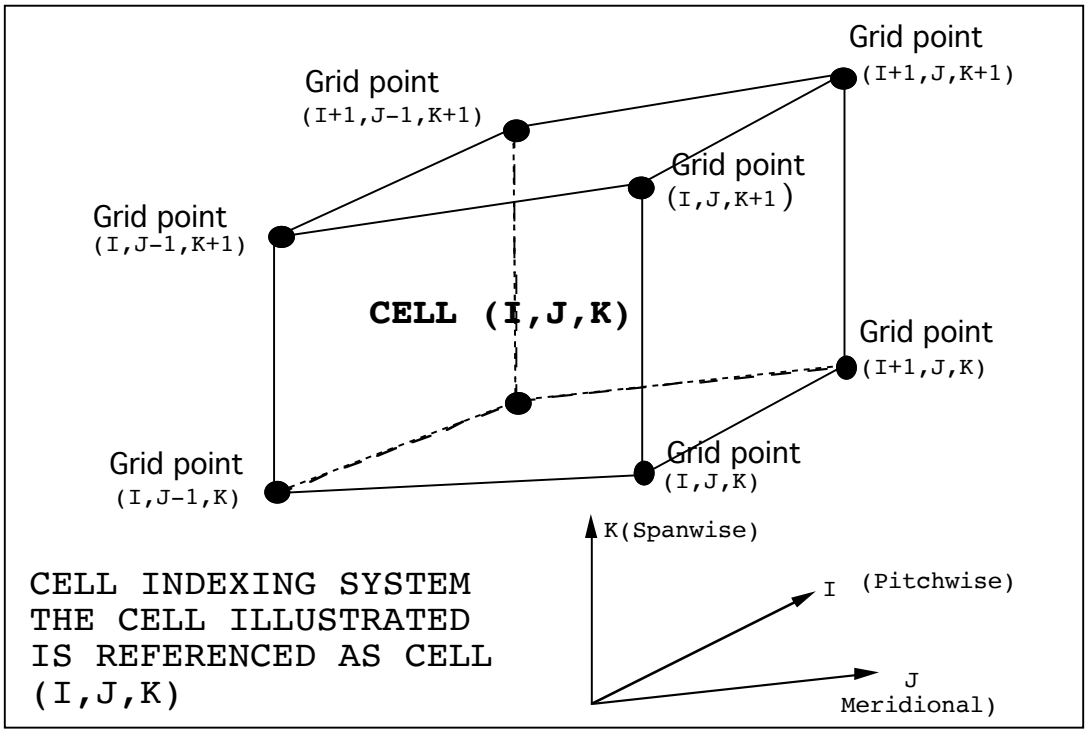
All variables are stored at cell corners, as illustrated below, which the author believes to be simpler and more accurate than cell centre storage. The cell indexing system is also illustrated in the Figure below. The cell numbered (I,J,K) has the grid point I,J,K at its corner with the largest J value, lowest I and lowest K values.

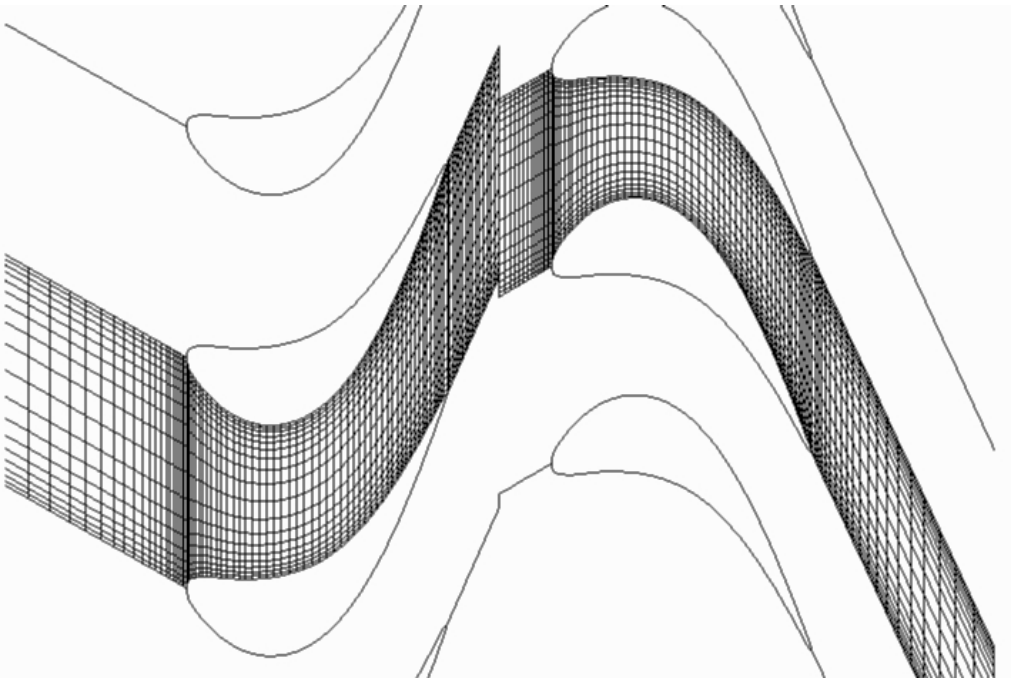
A typical number of grid points for an inviscid calculation would be IM = KM = 28, JM = 75, with fairly uniform spacing of the points in each direction, except for the J spacing being considerably reduced around the leading edge. For a viscous calculation a typical number would be IM = KM = 49, JM =150, with highly non-uniform spacing in the pitchwise and spanwise directions, a typical grid stretching factor in these directions would be 1:20.



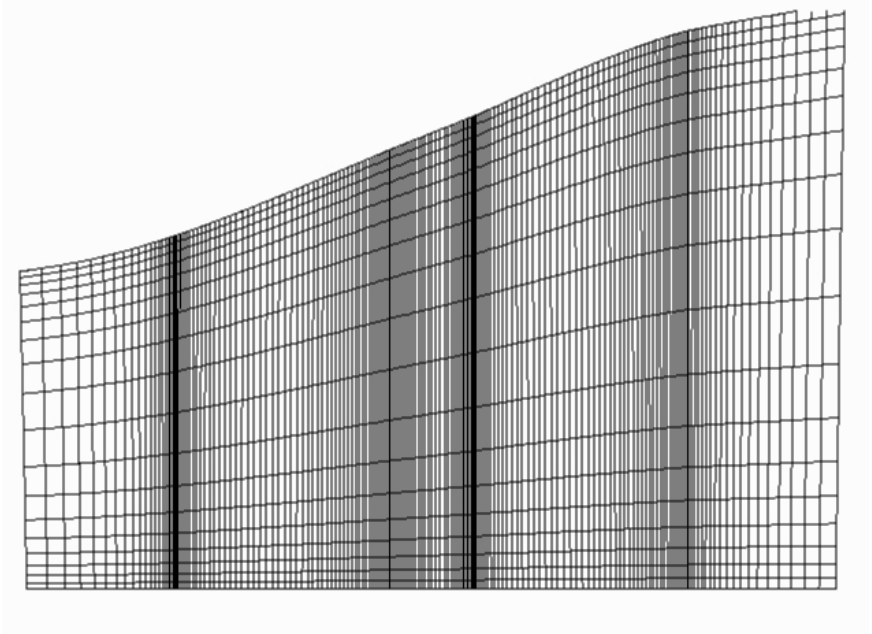
COORDINATES IN THE MERIDIONAL VIEW


I=IM
J=JLE
J=1
I=1
I Grid
Direction
Theta Direction
J=JTE
m Direction
J Grid Direction
J=JM
COORDINATES IN THE
BLADE TO BLADE VIEW

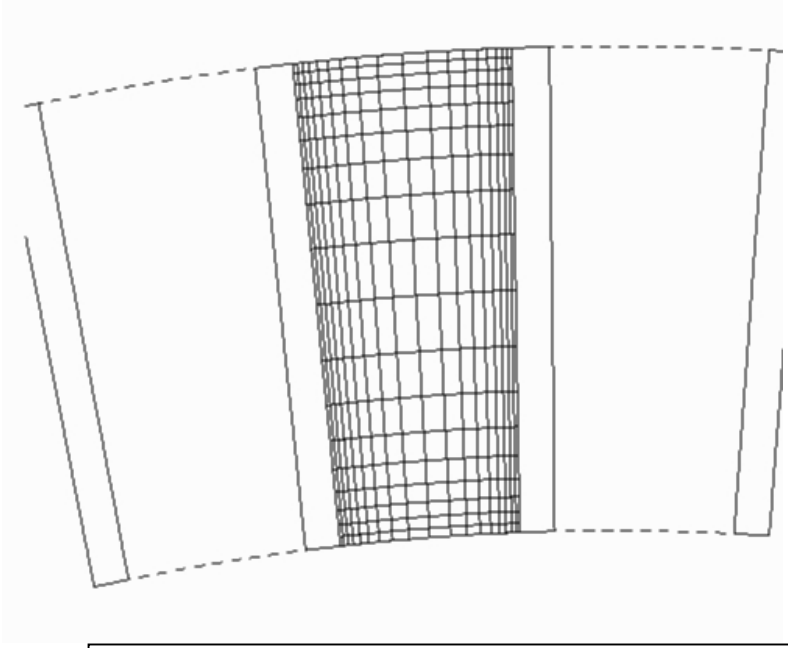

Grid point
(I+1,J,K+1)
Grid point
(I+1,J-1,K+1)
Grid point
(I,J,K+1)
Grid point
(I,J-1,K+1)
CELL (I,J,K)
Grid point
(I+1,J,K)
Grid point
(I,J-1,K)
Grid point
(I,J,K)
K(Spanwise)
I (Pitchwise)
J
Meridional)
CELL INDEXING SYSTEM
THE CELL ILLUSTRATED
IS REFERENCED AS CELL
(I,J,K)



Blade-to-blade view of the grid on a streamwise surface.
Coarse grid shown.



Meridional view of the grid.
Coarse grid shown.



Quasi-orthogonal view of the grid.
Coarse grid shown.

The grid lines should be closely spaced around the leading edge where flow properties change very rapidly, but experience shows that it is best to use a relatively coarse grid spacing, with a cusp, at the trailing edge. This is because a fine grid around the trailing edge usually gives unrealistic negative loading at the trailing edge. A cusp at the trailing edge is the standard option in this program but the latest versions allow a body force field and a fine grid to be used instead of a cusp to force the flow to separate at the trailing edge.

MULTIGRID

The most important factor in accelerating the convergence of the calculation is the use of multigrid. Three levels of multigrid are usually used and, unlike most other methods, the block sizes are not limited to simply doubling of the number of cells in each direction. In fact the size of the blocks may be chosen arbitrarily by the user but experience is that blocks of 3x3x3 cells for the lowest level and 9x9x9 cells for the middle level are often the optimum. It is best, but not essential, if there are a whole number of blocks across the span and pitch of the blade passage. The third level of multigrid is a one-dimensional calculation in which the blocks extend across the whole pitch and whole span with a fixed number of 4 blocks per blade row in the streamwise direction. The gives very rapid transfer of information from inlet to outlet of the whole flow field and so greatly speeds up convergence of multistage calculations.

The number of time steps for convergence depends greatly on the problem, particularly on the number of stages and on the uniformity of the grid, but it is typically in the range 2000 – 5000. The more the number of blade rows calculated and the more closely spaced the grid points the larger is the number of

steps required for convergence. The convergence limit set in the data is the average percentage change in meridional velocity per time step. A value of 0.005 is usual but lower values can be used if required. The calculation will not converge unless the maximum continuity error, i.e. the maximum difference between the local mass flow rate and the inlet flow rate, is less than 1%.

NEGATIVE FEEDBACK

This is a very simple but effective means of increasing the robustness of a code. It could be applied to most explicit CFD codes. The idea is to limit the rates of change of the flow properties at grid points where the calculated rates of change are greatest.

After summing the fluxes and multiplying by the local time step for all cells, the resulting rates of change, Δ_{calc}, are obtained. The absolute magnitudes of the rates of change are then averaged to find, Δ_{avg}, for the whole flow field. The rates of change of all cells are then changed by

$$\Delta_{used} = \Delta_{calc} /(1 + D) \text{ , where } D = \frac{\text{ABS}(\Delta_{calc})}{\text{DAMP}\,\Delta_{avg}}$$

DAMP is an input variable which controls the amount of negative feedback. Cells where the calculated change is much less than (DAMP x Δ_{avg}) will scarcely be changed but cells where the calculated change is comparable to or greater than (DAMP x Δ_{avg}) will have their changes reduced. This acts as a powerful stabilising influence on cells which might be unstable due to locally very high Mach numbers. Such local instabilities often occur during an initial transient when starting from a crude initial guess of the flow and would otherwise require the whole calculation to be run with a lower CFL number and/or higher smoothing. Typical values of DAMP are 10 – 25, but lower values may be used for difficult cases. If DAMP is set greater than 100 it is not used. The value of DAMP should have no effect on the final solution.

MIXING PLANE MODEL

Development of a satisfactory mixing plane model is an extremely difficult task. The objective is to allow the flow to mix out instantaneously at a plane between the blade rows, rather than gradually within the downstream blade row, whilst maintaining the mixing loss at a similar value. The mixing plane must conserve the pitchwise averaged fluxes of mass, momentum and energy but it must not impose pitchwise uniform conditions on the flow. In general the static pressure will rise and the entropy increase across the mixing plane to simulate the real mixing process. The mixing plane treatment has evolved over many years and has been changed and improved compared to the original very simple model. A detailed description of the current model is given in the note 17. The quasi-orthogonal grid lines at J = JMIX and J = JMIX +1 must be coincident in the meridional view, i.e. they must have the same x and r coordinates. This is done automatically by subroutines GRIDUP and GRIDOWN if the options ISHIFT = 2 , 3 or 4 are used. Use of this option is very strongly recommended.

VISCOUS MODELLING

As in all recent versions of the code viscous terms are included via body forces and source terms. The use of body forces to solve the Navier-Stokes equations is not an approximation and can be made as exact as any other method. The advantage of using the body force model is that the viscous terms need not be evaluated every time step, typically they are evaluated only every 5 steps leading to a very significant saving in CPU time. The code can be run as an inviscid calculation by setting ILOS = 0.

The wall shear stresses are obtained from wall functions. The usual model used assumes that the second grid point from the wall is either in the log-law region of a turbulent boundary layer, or in a laminar boundary layer. In this model the wall shear stress is obtained from a curve fit to the log law for equilibrium boundary layers. This gives a very good fit to the shear stress obtained from the standard log law over the range of Yplus from 25 to 1000. This model is used if the variable YPLUSWALL is less than 5. A different wall function is used if YPLUSWALL is greater than 5 and this uses an assumption that the first grid point is in the laminar sub layer at the specified value of YPLUS. This simplifies the calculation of skin friction but does not allow the skin friction to change with Reynolds number and so the chosen value of YPLUSWALL must be adjusted manually to allow for different Reynolds numbers. Hence the first (original) method is generally preferred.

A further option to use the wall functions suggested by Shih et al in NASA/TM-1999-209398 is available in version 17.5. These consist of two terms, a velocity based term and a pressure gradient based term. Only the velocity based term is used if YPLUSWALL is set in the range -10.0 to zero. This gives results very similar to the original wall functions. Both terms are used if YPLUSWALL is set below -10.0 , there is little experience of using this term as yet but is does not seem to make much difference for moderate pressure gradients.

Boundary layer transition can be specified at any point on any surface or can be predicted very approximately by the simple transition model suggested by Baldwin and Lomax.

The body forces were originally obtained from a thin shear layer approximation to the Navier-Stokes equations. The “thin shear layer” model is an approximation which assumes that viscous normal stresses and viscous stresses on the quasi-orthogonal faces of the elements can be neglected. Experience shows that this is not a severe limitation for turbomachinery flows. The turbulent viscosity in the original model is calculated using the mixing length approach. The mixing length is taken to vary linearly with distance from a wall up to a specified limit. This limit is input as data as a fraction of the local blade pitch and can be different on each blade surface and on each end wall, typically a limit = 0.03 of the local blade pitch is used. Lower values will give lower turbulent viscosity at the edge of the boundary layer, and hence less loss, and vice-versa. This model is still available in the code as subroutine LOSS, it is the fastest and most robust of the loss models.

Earlier versions of the program omitted shear work and internal heat flows because they may be shown to cancel in a flow with a pitchwise uniform stagnation temperature and a Prandtl number of unity. However, for cooling flows that model is not realistic and these terms are now included in all recent versions. All surfaces are, however, assumed to be adiabatic so heat transfer to or from solid to the gas cannot be calculated.

All recent version of the code include new loss routines NEW_LOSS and SPAL_LOSS. NEW_LOSS is an improved mixing length model, SPAL_LOSS is the well-known Spalart-Allmaras (SA) turbulence model. Both of these solve the full Navier-Stokes equations. They use exactly the same wall functions as in the original LOSS method.

In NEW_LOSS the local mixing length limit is calculated from the distance of the mid-span and mid-pitch point to the nearest solid surface. It is therefore roughly proportional to the local blade passage width. This length is then multiplied by a scaling factor which in input as data. The scaling factor is input for each blade row at blade row inlet (mixing plane), leading edge, trailing edge and exit (mixing plane) and is taken to vary linearly with meridional distance between these points. The value of the scaling factor should be similar to the mixing length limit used in the original LOSS routine but experience is that is needs to be slightly larger, say 0.04 instead of 0.03. The mixing length is taken as the lower of this value and the local perpendicular distance to the nearest wall. Note that, unlike in subroutine LOSS, the mixing length limit cannot be varied between the different blade surfaces or the different endwall surfaces, it can only be varied with meridional distance. The turbulent viscosity is calculated from the square of the mixing length x the local absolute vorticity x local density. Free stream turbulence may be included by inputting the ratio of the free stream turbulent viscosity to laminar viscosity in every blade row.

SPAL_LOSS is the Spalart-Almaras turbulence model. In this an additional convection equation is solved for the transformed turbulent viscosity. This uses about 20% more CPU time than the original LOSS method. However, the very complex source term used in the SA model is only calculated every 5-10 time steps and so does not use too much CPU time. In addition to the 4 source terms used in the basic SA model an additional source term has been added. This forces the turbulent viscosity towards the value obtained from NEW_LOSS model. Each of the source terms is multiplied by a scaling factor, which is input as data, and so the latter term would usually be turned off (scaled by zero) to obtain the basic SA model. By including the additional source term either hybrid mixing length-SA models or a modified mixing length model with convection of the turbulent viscosity, may be obtained. Experience is that the basic S-A model tends to underestimate the loss in turbomachinery and this may be corrected by increasing the main source term, FAC_ST0, setting this to 1.5 seems to give better results than the standard value of 1.0.

Version 17.5 includes an option to increase the main source term, FAC_ST0, in the SA model as suggested by Lee et al in ASME paper GT2017-63245. The increase consists of two terms, one increasing the source term due to streamwise vorticity (helicity) and the second increasing it due to adverse pressure gradients. These are used if the values of FAC_VORT and FAC_PGRAD in CARD 30 and set to be greater than zero. Lee et al suggest values of FAC_VORT = 0.9191 and FAC_PGRAD = 0.6565 but the values used are determined by the values input. There is little experience yet of using this option but it certainly does increase the turbulent viscosity in the boundary layers and should be useful in extending the operating range of compressors near their stall point.

With any turbulence model a question arises of how to model the transfer the turbulent viscosity across a mixing plane. This is modeled by pitchwise averaging the values of turbulent viscosity upstream of the mixing plane and passing a fraction of this average across the mixing plane as a pitchwise uniform variation to the next blade row. The fraction transferred in input as data for each blade row. It is difficult to know what is the correct fraction to transfer but a value around 0.5 seems appropriate.

The laminar viscosity may either be input directly as the dynamic viscosity or it can be calculated within the program by inputting the Reynolds number of the first blade row. The laminar viscosity is usually taken to be constant but if the input value of Reynolds number, Re, is negative then the viscosity at 288K is the (absolute value of Re) x 10^{-5} and it is then automatically scaled by a power law to allow for its variation with local temperature. e.g. for air the viscosity would be input as -1.9 and it would be automatically changed with temperature. This option is especially useful for multistage high speed machines where the temperature , and hence viscosity, varies significantly.

There is an option to allow for the increased skin friction due to surface roughness. Different levels of roughness can be specified on each surface and the wall functions are modified to allow for the roughness.

Boundary layer transition can be specified at a fixed “J” value in all 3 loss routines and the “J” value may be different on each surface. In LOSS and NEW_LOSS it is also possible to use a simple transition model, which says that the flow becomes turbulent when the maximum ratio of turbulent to laminar viscosity through the boundary layer exceeds an input limit. This is the model originally suggested by Baldwin & Lomax. This limit, FTRANS is typically 14. To obtain fully turbulent boundary layer the value of FTRANS should be very low and to obtain fully laminar ones it should be very high, say 10000 . The use of FTRANS is not possible with the SPAL_LOSS model, but this model automatically makes the turbulent viscosity low in laminar regions.

If very fine grids are used near to a solid boundary, i.e. a large number of points with a high expansion ratio, then it is possible that there may be several grid points within the laminar sub layer. The wall functions used originally only allowed the first grid point from the wall to be in the sub layer and calculated a turbulent viscosity for all other points. However, recent versions have an option to reduce the turbulent viscosity for grid points within the laminar sub layer and buffer layer, typically up to yplus = 25. This only has any effect if more than 2 points are within these layers, which is only likely to occur when very fine grids are used. This has little effect on the SA model but makes solutions with LOSS10 or NEW_LOSS more grid independent.

TIP LEAKAGE FLOWS

Tip leakage for plain tip clearances is calculated using the "pinched tip model" in which the blade is thinned towards the tip and periodicity is applied across the tip gap where the blade thickness is set to zero. This model may be used for stator hub clearances or rotor tip clearances. The model seems to give very realistic results, although, for thick blades, the actual tip gap calculated may need to be less than the physical gap to allow for the contraction of the leakage jet. A reduction in the tip gap to about 0.6 of the physical gap is recommended in such cases. The tip clearance is specified as a fraction of the local span at the blade leading and trailing edges and varies linearly between them. The number of grid cells within the gap is specified in the data and the grid is automatically adjusted to fit the gap. Typically 3 to 5 cells within the tip gap would be sufficient.

SHROUD LEAKAGE FLOWS

MULTALL also contains a shroud leakage model that models the leakage flow and loss for shrouded turbine rotor blades or compressor stator blades. The leakage mass flow is estimated from the seal clearance, the number of seals, the upstream stagnation pressure and the downstream static pressure and is bled off from the main flow. The change of angular momentum of the leakage flow due to friction on the shroud and casing is estimated using input values of the skin friction coefficient. The work done on the shroud by the leakage flow is also calculated. The leakage flow is then injected into the main flow downstream of the blade row and the conservation equations determine the mixing loss. Either hub or tip leakage can be calculated in this way and the flow may be from upstream of the blade row to downstream, as in turbine rotor blades, or from downstream to upstream as in compressor stator blades.

GENERATION OF MODIFIED GRIDS

The number of grid points in the pitchwise and spanwise directions must be the same for all blade rows and are specified by IM and KM in the input data. The grid spacing in the pitchwise (I) and spanwise (K) directions can be easily changed by specifying FP(I) and FR(K) in the input data. The spacings can etiher be input directly or generated by the program from input values of the expansion ratio and maximum grid expansion. The spacing in the streamwise (J or meridional direction) can also be changed using subroutine NEWGRID. This enables a new grid with a different number of "J" grid points to be generated from the input data. It is useful whenever more grid points are needed or the grid needs to be refined locally. Either the *relative* meridional spacing of all the new grid points can be input as data, or the *relative* spacing at just a few points can be input as a function of the meridional distance, the program then interpolates in these few points to obtain all the new grid spacings and hence the new grid points.

The grids input in the data set may overlap in the meridional direction and they need not be contiguous at the mixing planes. If ISHIFT = 2, 3 or 4 is specified then the grids are automatically made contiguous at the mixing planes with the grid spacings made into a geometric series away from the trailing edge and leading edge. If too many points are used between the trailing edge or the leading edge and the mixing plane the geometric series may produce very close spacings at the mixing plane and this may cause instability.

CUSP GENERATION

Although it is not essential to use cusps at the blade leading and trailing edges it is strongly recommended that a cusp is used at a thick trailing edge. If a cusp is used then the grid spacing should not be reduced at the trailing edge but should be made similar to that on the rear of the blade. The trailing edge cusp applies a blockage to the flow but carries no tangential load and hence does no work. It allows the flow to separate from the blade surfaces at well defined points without it starting to turn around the trailing edge, which would cause it to generate locally low pressures on the near trailing edge points. A flexible method of cusp generation at blade trailing edges is included in the program. The shape and length of the cusp can be chosen and also part of the cusp can be treated as a part of the blade so that it carries load. Cusps are not necessary at the blade leading edge as long as sufficient grid points are used to define the flow around the leading edge circle. A very fine grid is needed to achieve this, typically about 6 points on the leading edge circle, this number may be reduced by using a cusp at the leading edge but this is not usual.

The program also includes an option force the flow at the trailing edge to separate by a body force. This allows a fine grid to be used around a thick trailing edge without the solution generating unrealistic reverse loading at the trailing edge. See Note 16. However, this option is not often used.

EXIT FLOW THROTTLING

An option to model a "perforated plate" type of downstream boundary condition is included. This is useful when there is a separated flow at the downstream boundary so that the meridional velocity is negative and the standard boundary condition becomes ill conditioned. The flow is made more uniform at the downstream boundary by simulating the presence of a flow resistance, such as a gauze or a perforated plate, at the boundary. This effectively adds a pressure drop equal to PLATE_LOSS x $0.5*\rho*(Vm^2 - Vm_{mid}^2)$ at the exit. Thus it should not change the average exit static pressure but it does force the flow to become more uniform at the exit boundary. A typical value of PLATE_LOSS is about 2. If PLATE_LOSS is set to zero then the option is not used.

An alternative treatment for problems at the exit boundary is to increase the smoothing of the flow for points close to it. The flow may be given an extra smoothing, using a smoothing factor SFEXIT, which is applied over NSFEXIT grid points upstream from the downstream boundary. Typical values might be SFEXIT = 0.05, NSFEXIT = 10 . This option usually works well and is generally preferred to the simulated perforated plate model, but should only be used if the exit boundary is well downstream of the last blade row.

It is also possible to use a throttle boundary condition to vary the exit static pressure with the exit mass flow rate. This gives improved stability near to the stall point for compressors. This option is used if THROTTLE_EXIT is greater than zero. The exit static pressure is made to vary parabolically with the exit mass flow according to:

$$Pexit = THROTTLE\_PRES * (m_{exit}/THROTTLE\_MAS)^2$$

Where THROTTLE_PRES is the required exit pressure in N/m^2 and THROTTLE_MAS is the expected exit mass flow rate in Kg/sec. Changes in exit pressure are relaxed by RFTHROTL which, together with THROTTLE_PRES and THROTTLE_MAS, is input as data if THROTTLE_EXIT is non zero. Increasing the value of THROTTLE_PRES will increase the back pressure and move a compressor towards stall. However, forcing the exit flow to lie on a parabolic pressure:mass flow characteristic through the specified point allows a closer approach to the true stall point than was previously possible. Use of this option is recommended for all compressor calculations near their stall point. THROTTLE_EXIT must be set to zero to prevent the use of this option.

BLADE AND ENDWALL COOLING FLOWS

Cooling flows can be added at any point on the blade and endwall surfaces. The flow is added through a series of "patches" whose I,J,K boundaries are specified in the input data. If the "mixing plane" falls within a region where coolant is being added then two separate patches, one upstream and one

downstream of the "mixing plane" must be used. The coolant mass flow, stagnation temperature, stagnation pressure, ejection Mach number and flow directions must be specified for each patch. Note that the exit relative Mach number of the coolant flow is specified and is not calculated from the local static and stagnation pressures. If it is required to model individual cooling holes then each patch may be one grid cell in size, but this requires a great deal of input data and it is more usual to specify a single patch to cover multiple cooling flows. The overall total-to-total efficiency is calculated and printed out, allowing for the potential work of all the cooling flows. However, the polytropic efficiencies, which are also printed out, relate the mass averaged inlet and outlet flow conditions and are not meaningful when cooling flows are added.

The cooling flow temperature and pressure are input at the point where the flow is fed into the (possibly rotating) disc together with its angular momentum at that point. The pumping work done on the coolant by a rotating blade, and its consequent stagnation pressure and temperature at the point of ejection, is calculated by the program.

BLEED FLOWS

Flow can be bled from the machine (as is common in steam turbines for feed heating or in gas turbine compressors for turbine cooling) at any point on the blade surfaces or on the hub and casing. As with coolant flows the flow is bled off from a "patch" whose I,J,K boundaries must be specified by the user. However, the bled flow is always assumed to have the flow properties, e.g. enthalpy and entropy, at the point where it is bled off, and so they cannot be specified by the user. The machine power output/input is calculated allowing for the bled flows but the efficiencies which are printed out are defined by using the inlet and outlet states of the flow remaining in the machine and do not allow for the bleed flows.

LOW SPEED FLOWS

The basic program works remarkably well at low Mach numbers and will usually converge if the average Mach number is greater than about 0.15, which gives effectively incompressible flow. However, convergence becomes slower at low Mach numbers and so the program has been extended to use the artificial compressibility method to work with very low speed or incompressible flows.

The method works be inputting an artificial speed of sound, V_{snd} and calculating pressure changes directly from changes in an artificial density, Ro_{sub}, using

$$\Delta P = V_{snd}^{2} \times \Delta Ro_{sub}.$$

The change ΔRo_{sub} in the artificial density is obtained from the continuity equation in the usual way. The change in pressure is then calculated using

$$(P - P_{ref}) = V_{snd}^{2} (Ro_{sub} - Ro_{ref})$$

P_{ref} and Ro_{ref} are usually taken to be the inlet stagnation pressure and density.

The true density, which changes only slightly in low speed flow, is then calculated from the pressure and internal energy, with the changes in it relaxed by a factor RF_PTRU, for which a typical value is 0.01. This density is then used to find the flow velocities from calculated values of the mass flux ρV .

The velocity of sound should initially be chosen to be about twice the maximum velocity expected in the whole flow field but it is automatically updated as the program runs to a set multiple of the maximum relative velocity in the whole flow field. The rate of updating is controlled by the input variable RF_VSOUND for which a typical value is 0.002, which means that the updates take around 500 time steps to settle down. This method can be used to speed up convergence for flows with Mach number less than about 0.25 or it can be used for extremely low Mach numbers less than 0.05, where the basic program may not converge. It can also be used for completely incompressible flows when the true density is simply input as data and is not re-calculated. The low Mach number option is called if ITIMST = 5 and the completely incompressible option is used if ITIMST = 6. If either option is used than the artificial speed of sound and relaxation factors on changes in speed of sound and in true density must be input as data. For completely incompressible flow then the fixed value of fluid density must also be input.

REAL GAS PROPERTIES

Early versions of the program always used perfect gas properties. Recent versions allow for real gas properties where the gas constant is independent of temperature but the specific heat capacity, Cp, varies with temperature. The value of Cp is taken to vary quadratically with temperature according to

$$Cp = Cp_1 + Cp_2(T - T_{ref}) + Cp_3(T - T_{ref})^2$$

The advantage of this formulation is that the enthalpy and entropy can still be calculated analytically.

The values of Cp_1, Cp_2, Cp_3 and T_{ref} are input as data. For combustion products the values of Cp_1, Cp_2 and Cp_3 are approximately 1272.5, 0.2125 and 0.000015625 J/kg K at Tref = 1400 K . The gas constant is held constant at 287.5 .

This option is used if the value of Cp input in the standard data set is negative, in which case values of Cp_1, CP_2, Cp_3 , T_{ref} and RGAS are input in the next line of data.

Steam properties are not available in MULTALL but an earlier version called MSTEAM contains a steam property routine. However, equilibrium steam can be reasonably well approximated as a perfect gas over the typical pressure range of a few stages. For low pressure wet steam values of Cp = 7300 J/kgK, gamma = 1.07 , are suggested. For dry steam at low pressures, typical values are Cp = 1970 J/kgK, gamma = 1.3. If steam changes from dry to wet within the region being calculated then it cannot be accurately modeled as a perfect gas.

BLADE REDESIGN OPTION

Using this feature it is possible to perform a limited redesign of the blade sections by generating new camber lines and thickness distributions. The stream surface coordinates are also changed. The method of specifying the new blade section is very similar to that in the author's blade design system, STAGEN .

A related option is to restagger the input blade, in which case blade is simply rotated about a specified axis so that its leading and trailing edge coordinates will change. The blade can also be leaned in the tangential direction, by a specified tangential distance relative to the hub section.

REPEATING STAGE OPTION

In many multistage machines the flow repeats from stage to stage with the velocity profiles and stagnation pressure profiles remaining almost constant and only the stagnation pressure level changing. However, the shape of the stagnation temperature profile cannot remain constant since to do so would imply the same entropy rise for all streamlines. In reality the stagnation temperature will increase in the high loss regions near the end walls relative to that near mid span. When designing a single repeating stage it is desirable to try to satisfy this repeating stage condition and this can be done by automatically feeding back the exit stagnation pressure profile and yaw angle profile to the inlet boundary conditions. The stagnation temperature profile and pitch angle profile are not fed back but remain at the values input in the original data set. When run in this way the program naturally takes longer to converge but it saves multiple runs with manual adjustment of the boundary conditions.

This option should only be used when a single stage is being calculated.

QUAS1 3D FLOW

Recent versions allow quasi-3D (Q3D) flow on a blade to blade stream surface to be calculated with only a single cell being used in the spanwise direction. This is chosen by setting KM = 2, it enables Q3D solutions for a single blade row to be obtained in the order of 15 seconds. Viscous effects are included in the usual way on the blade surfaces but not on the stream surfaces. The stream surface is defined by the single surface on which the blade coordinates are input, together with a separate input of the relative stream surface thickness as a function of meridional distance. It is important to realise that the stream surface thickness has a very large influence on the blade loading and so it should be chosen realistically. The flow is forced to follow the stream surface by using a pressure difference between the two stream surfaces. This effectively produces a body force acting perpendicular to the surface so that it does not influence the flow on the stream surface. The value of this body force is controlled by the input value of Q3DFORCE but the value does not seem to be critical and values between 1.0 and 2.0 are usually acceptable, with higher values sometimes possible

To use this option set KM = 2 and input the value of Q3DFORCE in CARD 32.

THROUGHFLOW MODE

The program can be run as an axisymmetric throughflow calculation. This is done if IM is set = 2 so there is only a single cell in the pitchwise direction. The blade geometry is input as usual, although an accurate blade profile is not essential, the exit blade centre line angle should be close to the required

exit flow angle. Multiple stages can be calculated in this way. The number of grid points cam also be far less than for a 3D calculation and around 30 points streamwise and 15 points spanwise should be sufficient on each blade row. This gives fast run times of order 10 seconds per blade row.

The flow is forced to closely follow a surface which is coincident with the centre line of the blade The blade loading is applied by calculating a blade surface pressure distribution which will force the flow to follow this centre line. This blade surface pressure distribution is not an accurate prediction of the actual distribution but is reasonably realistic. In particular the overall integrated (tangential force x radius) will balance the change in angular momentum of the flow. The surface pressures apply axial, radial and tangential forces to the flow and so the effects of three-dimensional blade geometry are included. The flow will depart slightly from the centre line because of smoothing of the blade loading, which is necessary to prevent discontinuities at the leading and trailing edges, this smoothing is useful in that it reduces the leading and trailing edge loading and so provides some deviation at the trailing edge. Extra deviation between the blade centre line and the exit flow direction can be input as data as is done in most throughflow calculations.

The loss may be calculated by the same loss routines as used for 3D flow but these will not be accurate with only 2 grid lines. It is suggested that the option using YPLUSWALL is used, this will apply a constant skin friction factor on the blade surfaces equal to $2/(\text{YPLUSWALL})^2$ and so a value of YPLUSWALL = 20 will give a realistic skin friction coefficient of 0.005 . The optimum value of YPLUSWALL can be chosen to agree with experience. Tip leakage can be included in the usual way but it is only driven by the difference between the inlet flow direction and the blade surface, not by a pressure difference between the pressure and suction surfaces, and so the leakage will be less than in reality.

In the throughflow mode the flow along each streamline is effectively one dimensional with a flow area equal to the local passage width measured perpendicular to the flow. This means that, with supersonic flow, the flow will behave as in a one-dimensional nozzle. It will choke at a point of minimum area and the downstream expansion will be terminated by a normal shockwave. The method cannot predict the oblique shock waves that are common in turbomachines and the calculated blade loading in such cases will not be realistic. This is common to all time-marching throughflow methods unless they specify the local tangential velocity rather than the flow direction. In the later case only shock waves with an upstream axial Mach number greater than one can be predicted. However, the present method does predict the supersonic deviation found on turbine blades with supersonic exit flow.

COMPILING THE PROGRAM

The program is written in very standard FORTRAN77 and should compile and run on any computer with a FORTRAN compiler. The only non-standard routine is the timing call, which evaluates the CPU time per time step. This will vary for different compilers and operating systems. At present the "MCLOCK" routines for the LINUX g77 and gfortran compilers are coded but these may need changing. The timing calls are not essential for the running of the program and can be removed or commented out if not required, or if the program will not compile with them present.

All the variables are declared and dimensioned within a single common block, e.g. "**commall_open**" which must be present in the same directory as the program when is compiled. The sizes of the arrays are declared by PARAMETER statements in "**commall_open**". Typical maximum dimensions of the grid would be 65 points in the pitchwise and spanwise directions and 1000 points in the streamwise (J) direction. With these dimensions the program uses about 750 MBytes of memory. The large number of streamwise grid points is only needed for multistage calculations with many blade rows, typically about 120 points per blade row would be sufficient. There is no limitation on the size of the arrays apart from those due to computer memory limits. If the program will not compile due to insufficient computer memory then, under LINUX, the available memory may be extended by adding " -mcmodel=medium " to the end of the compilation line. e.g. gfortran –O –o test.x test.f –mcmodel=medium . Similar extensions are probably available for other compilers and can be searched for on the internet.

The speed of execution is generally increased by using the highest level of compiler optimization available. For a throughflow calculation, with IM = 2 , the speed is increased if the dimension ID is set = 2 in the parameter statement of "**commall-open"**.

RUNNING THE PROGRAM AND INPUT AND OUTPUT FILES

Before trying to start the program the file "**intype**" must be created in the same directory. This contains the single character "O" or "N" deciding whether the "**old**" or "**new**" input format will be used.

The main input data is read from FORTRAN unit 5. The file name is not specified within the program and so the program is most easily run using a command such as

MULTALL .X < DATA.IN

Where MULTALL.X is the name of the compiled executable code and DATA.IN is the name of the data file. The program will run either until the average residual reaches the specified tolerance, CONLIM, or for the maximum number of time steps specified, NMAX.

Several different output files are produced. The standard output is to FORTRAN unit 6, which defaults to the screen, this gives a summary of the convergence history every 5 time steps and a more detailed average of the flow properties every 200 time steps. If it is required to send this output to a file then use a command such as

MULTALL.X < DATA.IN > RESULTS.OUT

Where RESULTS.OUT is the name of the output file.

Other output files are set within the program and are produced automatically. These are:

"**stage.log** " A formatted file containing the convergence history with values of the rms error, continuity error and inlet mass flow is written to FORTRAN unit 4 output every 5 time steps. This file may be used to plot out the convergence history using program **histage**.

"**flow_out**" An unformatted file containing the all the flow properties is written to FORTRAN unit 7. This file may be output after specified numbers of time steps. It is also automatically output on convergence or on reaching the specified maximum number of time steps. This file, together with the file "**grid_out**" may be used to plot out the results. The same file is also used as a restart file if a restart is requested.

"**grid_out**" An unformatted file containing the grid coordinates is written to FORTRAN unit 21. It is used, together with "**flow_out**", to plot out the results.

'**global.plt**" An unformatted file containing the one-dimensional mass averaged flow data is written to FORTRAN unit 11. This may be used by program **globplot** to plot out the one-dimensional variation of mass flow, stagnation pressure, stagnation temperature entropy or lost efficiency along the flow path.

"**results.out**" A formatted file containing selected flow properties is written to FORTRAN unit 3. The properties may be selected as described in Note 7. This file may be very large and so the output

requested should be chosen carefully. Usually no output should be requested except for possible debugging.

"**loss-co.plt**" A formatted file which contains the loss of isentropic efficiency at every "J' station is written to FORTRAN unit 23 at the end of any run and may be used to plot lost efficiency against meridional distance.

"**mixbconds**" A formatted file written to FORTRAN unit 12. This contains the mixed out values of the flow properties at each mixing plane at every spanwise (K) grid point. It may be used to provide the inlet boundary conditions to a subsequent calculation on an individual blade row or smaller group of blade rows.

The plotting programs that use some of these files are based on the HGRAPH plotting package and so are not publicly available.

There is an option to stop the program and write out the results and a plot file at any time. Every 10 time steps the program opens and reads a file named "**stopit**" which may be opened and edited by the user whilst the program is running. This file contains a single number, if the number is zero the execution will continue, if it is 1 then the execution ends and the results files are written. To stop execution, pause the program, edit "**stopit**" and type 1 in place of the 0 , then restart execution of the program .

J D DENTON
Updated for MULTALL_OPEN February 2017