A User's Guide to MTFLOW 2.01

Multi-passage ThroughFLOW
Design/Analysis Program

Mark Drela
MIT Aerospace Computational Design Laboratory
June 2010

The MTFLOW system is a collection of programs for the viscous/inviscid analysis and design of axisymmetric bodies and axisymmetric flow passages, including effects of swirl, heat addition, loss generation, and area blockage.

MTFLOW can also be used for the design of three-dimensional turbomachinery blading, which can be optionally refined using the MISES system of blade-to-blade programs.

Contents

1 Overview

The MTFLOW programs and their summary descriptions are listed below.

MTSET Reads the geometry file `walls.xxx`, generates the initial streamsurface grid, and writes out an initial *state file* named `tdat.xxx`.

MTSOL Reads the `tdat.xxx` state file, accepts flow specification commands from the user, computes the solution, and writes out a modified state file containing the converged solution. Can be executed repeatedly with a variety of input conditions. Allows interactive plotting of results.

MTFLO This program serves a number of related functions:

- Takes field distributions of prescribed swirl, heat addition, etc., from a user-generated `tflow.xxx` file or a stack of blade-to-blade MISES solutions, and includes these into the `tdat.xxx` state file for subsequent reconvergence with **MTSOL** .
- Interrogates an already-converged solution to generate streamsurface properties for MISES solutions.
- Generates initial blade-to-blade streamsurface blade shapes for MISES solutions corresponding to a prescribed MTFLOW loading field.
- Generates three-dimensional blade shapes corresponding to the current stack of blade-to-blade solutions.

The block diagram for these programs is given at the end of this manual. The "`xxx`" extension suffix is used to designate the case being run, and can be chosen arbitrarily.

2 Reference Quantities

MTFLOW defines a reference inlet density $\rho_{\rm inl}$ and a reference inlet speed of sound $a_{\rm inl}$, which are related to the corresponding reference pressure $p_{\rm inl}$ and enthalpy $h_{\rm inl}$ by the state equation.

$$\gamma\, p_{\rm inl} \;=\; \rho_{\rm inl}\, a_{\rm inl}^2 \;=\; (\gamma-1)\,\rho_{\rm inl}\, h_{\rm inl}$$

The specified flow parameters are the reference Mach number $M_{\rm inl}$ and the reference Reynolds number $Re_{\rm inl}$. These implicitly define the reference velocity and reference viscosity,

$$V_{\rm inl} \;=\; M_{\rm inl}\, a_{\rm inl} \qquad\qquad \mu_{\rm inl} \;=\; \frac{\rho_{\rm inl}\, V_{\rm inl}\, L_{\rm ref}}{Re_{\rm inl}}$$

with $L_{\rm ref}$ being the length unit used to define the geometry coordinates. The speed $q = \sqrt{V_x^2 + V_r^2}$ is the projection of the total velocity on the meridional x,r plane and so does not include the swirl velocity V_θ.

The inlet reference quantities are used to compute dimensionless flow quantities such as the pressure coefficient, skin friction, etc.

$$C_p \;=\; \frac{p - p_{\rm inl}}{\frac{1}{2}\,\rho_{\rm inl}\,V_{\rm inl}^2} \qquad\qquad C_f \;=\; \frac{\tau_w}{\frac{1}{2}\,\rho_{\rm inl}\,V_{\rm inl}^2}$$

Other output is given in terms of ratios, like $q/V_{\rm inl}$, $p/p_{\rm inl}$, etc.

Internally, MTFLOW actually uses the inlet *stagnation* density $\rho_{o\,\rm inl}$ and speed of sound $a_{o\,\rm inl}$ as reference variables. This is mentioned in case changes to the source code are attempted. The Fortran names and assigned values of the internal reference quantities are

$$\begin{aligned}\rho_{o\,\rm inl} &= \texttt{RSTRO} = 1\\ h_{o\,\rm inl} &= \texttt{HSTRO} = 1/(\gamma\!-\!1)\end{aligned}$$

with $a_{o\,\rm inl} = 1$ and $p_{o\,\rm inl} = 1/\gamma$ being implied. All internal calculations also use the corresponding stagnation-condition Reynolds number

$$\begin{aligned}\frac{\rho_{o\,\rm inl}\,a_{o\,\rm inl}L_{\rm ref}}{\mu_{o\,\rm inl}} \equiv Re_{o\,\rm inl} \;&=\; \texttt{REYN} = Re_{\rm inl}\,\frac{\rho_{o\,\rm inl}}{\rho_{\rm inl}}\,\frac{a_{o\,\rm inl}}{V_{\rm inl}}\,\frac{\mu_{\rm inl}}{\mu_{o\,\rm inl}}\\ \text{where}\qquad \frac{\rho_{o\,\rm inl}}{\rho_{\rm inl}} \;&=\; \left(1 + \tfrac{\gamma-1}{2} M_{\rm inl}^2\right)^{1/(\gamma-1)}\\ \frac{a_{o\,\rm inl}}{V_{\rm inl}} \;&=\; \left(1 + \tfrac{\gamma-1}{2} M_{\rm inl}^2\right)^{1/2}\,\frac{1}{M_{\rm inl}}\\ \frac{\mu_{\rm inl}}{\mu_{o\,\rm inl}} \;&=\; \left(1 + \tfrac{\gamma-1}{2} M_{\rm inl}^2\right)^{-3/2}\,\frac{1\;+\;h_S/h_{o\,\rm inl}}{\left(1 + \tfrac{\gamma-1}{2} M_{\rm inl}^2\right)^{-1} + h_S/h_{o\,\rm inl}}\end{aligned}$$

which uses Sutherland's viscosity law, with $h_S = c_p\,T_S$ being the enthalpy corresponding to the Sutherland-constant temperature T_S, which is $T_S = 110°$K for air. All MTFLOW arithmetic is dimensionally consistent, with no assumptions being made about the values of `RSTRO` and `HSTRO` anywhere in the code except for their initialization statements. Hence, they could be set to their actual standard-unit values for any given application. For instance, the geometry coordinates could be input in meters, `RSTRO` in kg/m^3, and `HSTRO` in m^2/s^2, so that all internal quantities will be in these SI units as well. This may be convenient if the source code is modified to produce custom dimensional output. Currently, all MTFLOW graphical output is in dimensionless ratios like M, C_p, $q/V_{\rm inl}$, etc., which are unaffected by choice of units.

3 Inviscid Flow Quantities and Governing Equations

MTFLOW uses the axisymmetric Euler equations to describe the inviscid part of the flow. These are discretized on an intrinsic streamline grid, much like the older streamline-curvature methods. A major distinction is that MTFLOW has a conservative finite-volume discretization, and can therefore capture shocks properly.

This section will summarize the flow quantities and relations used by MTFLOW. This is necessary to clearly define the field parameters which can be specified to serve as field source terms for the flow solver.

3.1 Inviscid flowfield variables

In the inviscid part of the flowfield, **MTSOL** solves for the following primary flowfield variables.

x, r streamline-grid node locations

ρ density

h_o total enthalpy

$\bar{\Gamma}$ swirl, or circulation per radian: $\bar{\Gamma} = \Gamma/2\pi = rV_\theta$

m streamtube mass flow

The primary variables above are used to define the following secondary variables.

$$\begin{aligned}
A &= f(x,r) && \text{streamtube area} \\
q &= \frac{m}{\rho\, A\,(2\pi r - BT_\theta)} && \text{meridional speed } (= \sqrt{V_x^2 + V_r^2}) \\
V_\theta &= \frac{\bar{\Gamma}}{r} && \text{tangential speed} \\
h &= h_o - \frac{1}{2}q^2 - \frac{1}{2}V_\theta^2 && \text{static enthalpy} \\
p &= \frac{\gamma - 1}{\gamma}\,\rho\, h && \text{static pressure} \\
\tilde{s} &= \ln\left[\frac{(h/h_{\rm inl})^{\gamma/(\gamma-1)}}{p/p_{\rm inl}}\right] && \text{entropy}
\end{aligned}$$

In the q definition, T_θ is the circumferential blade thickness and B is the number of blades. The blockage term BT_θ therefore restricts the circumferential perimeter $2\pi r$. Outside a blade-row volume we have $T_\theta = 0$.

3.2 Field parameters

MTFLOW accepts the following prescribed *field parameters* which enter as forcing terms in the flowfield equations.

B Number of blades in a blade row.

Ω Rotation rate of a blade row, with same sign direction as $\bar{\Gamma}$ and V_θ.

T_θ Circumferential blade thickness.

$\mathcal{S}_{\rm rel}$ Blade slope ($= \tan\beta$) in the m'–θ plane. Alternative to ΔG.

ΔG Swirl change due to blade row loading. Alternative to $\mathcal{S}_{\rm rel}$.

ΔH Total enthalpy change due to heat release or removal.

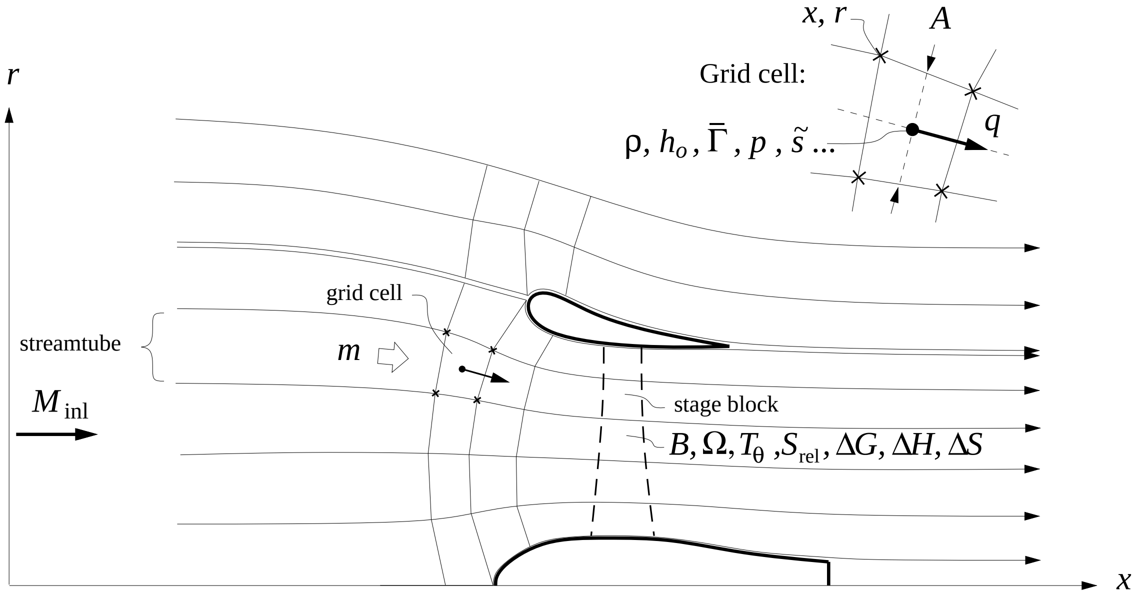

x, r
A
r
Grid cell:
ρ, h_o, $\bar{\Gamma}$, p, $\tilde{s}$ ...
q
grid cell
streamtube
m
M_{inl}
stage block
B, Ω, T_θ, S_{rel}, ΔG, ΔH, ΔS
x

Figure 1: Axisymmetric flowfield with streamline grid. Only three quasi-normal grid lines are shown. Flow variables $\rho, h_o \ldots$ are located at center of each x, r streamtube grid cell, which also defines the streamtube area and flow direction. Field parameters $B \ldots \Delta S$ are defined inside some number of stage blocks.

ΔS Entropy change due to some adiabatic throttling or frictional mechanism.

It is important to note the specific physical mechanisms associated with the parameters. Any given heating change ΔH will change the total enthalpy, and also will change the entropy in addition to ΔS (the latter is nonzero only for adiabatic changes, for which $\Delta H = 0$). Any given swirl change ΔG due to a blade row will change the swirl, and if the blade row is rotating it will also change the total enthalpy in addition to ΔH. The field parameter changes associated with idealized turbomachinery components are listed below.

Compressor:	$\Delta G > 0\ ,\ \ \Delta H = 0\ ,\ \ \Delta S = 0.$
Combustor:	$\Delta G = 0\ ,\ \ \Delta H > 0\ ,\ \ \Delta S = 0.$
Adiabatic Turbine:	$\Delta G < 0\ ,\ \ \Delta H = 0\ ,\ \ \Delta S = 0.$
Cooled Turbine:	$\Delta G < 0\ ,\ \ \Delta H < 0\ ,\ \ \Delta S = 0.$
Throttle:	$\Delta G = 0\ ,\ \ \Delta H = 0\ ,\ \ \Delta S > 0.$

In the table above, the rotating components are assumed to have a positive Ω, otherwise the sign of ΔG and its inequalities above would be reversed. If stage viscous losses are to be included for a compressor or turbine, then the nonzero overall entropy rise across the stage would be

$$\text{Nonideal Compressor or Turbine:} \qquad \Delta S \;=\; \tfrac{\gamma}{\gamma - 1} \ln \tau - \ln \pi$$

where τ is the total-temperature ratio and π is the total pressure ratio across the stage.

3.2.1 Incompressible limit

MTFLOW is a compressible-flow solver, and cannot be run at exactly zero Mach number. In practice, specifying an inflow or freestream Mach number $M_{\rm inl}$ low enough to produce $M < 0.15$ locally everywhere will give effectively incompressible results. For such flows, throttling losses are most conveniently specified as a loss in total pressure p_o. This must be converted to the equivalent ΔS for input into MTFLOW, which can be computed directly from the entropy definition, evaluated at the stagnation condition.

$$\Delta S \;=\; \ln\left[\frac{(h_o/h_{o\,\rm inl})^{\gamma/(\gamma-1)}}{p_o/p_{o\,\rm inl}}\right] \tag{1}$$

If there is no heat release, then the total enthalpy ratio term can be more conveniently defined in terms of an "ideal" or loss-free total pressure ratio.

$$\frac{p_{o\,\rm ideal}}{p_{o\,\rm inl}} \;\equiv\; \left(\frac{h_o}{h_{o\,\rm inl}}\right)^{\gamma/(\gamma-1)} \qquad \text{(adiabatic)} \tag{2}$$

The entropy change then takes on the following simpler form.

$$\Delta S \;=\; \ln\left[\frac{p_{o\,\rm ideal}}{p_o}\right] \qquad \text{(adiabatic)} \tag{3}$$

Note that these are absolute pressures, so that ΔS will in general be $\mathcal{O}(M^2)$, and must be computed for the specific $M_{\rm inl}$ which is chosen to approximate the incompressible case.

Frequently it may be more convenient to work in terms of a total-pressure loss coefficient C_{po}, which is well-defined in the incompressible limit. Rather than use the more common freestream total pressure as the reference, here it's most convenient to choose the local ideal total pressure,

$$C_{po} \;\equiv\; \frac{p_o - p_{o\,\rm ideal}}{\frac{1}{2}\rho_{\rm inl}V_{\rm inl}^2} \;=\; \frac{1}{\frac{1}{2}\gamma\, M_{\rm inl}^2}\,\frac{p_{o\,\rm inl}}{p_{\rm inl}}\,\frac{p_{o\,\rm ideal}}{p_{o\,\rm inl}}\left(\frac{p_o}{p_{o\,\rm ideal}} - 1\right) \tag{4}$$

which makes C_{po} unaffected by any p_o changes due to upstream isentropic work. The corresponding entropy rise is now

$$\Delta S \;=\; -\ln\left[1 \,+\, \frac{\gamma}{2}M_{\rm inl}^2\left(1+\frac{\gamma-1}{2}M_{\rm inl}^2\right)^{-\gamma/(\gamma-1)} C_{po}\,\frac{p_{o\,\rm inl}}{p_{o\,\rm ideal}}\right] \tag{5}$$

$$\Delta S \;\simeq\; -\frac{\gamma}{2}M_{\rm inl}^2\left(1+\frac{\gamma-1}{2}M_{\rm inl}^2\right)^{-\gamma/(\gamma-1)} C_{po}\,\frac{p_{o\,\rm inl}}{p_{o\,\rm ideal}} \tag{6}$$

$$\Delta S \;\simeq\; -\frac{\gamma}{2}M_{\rm inl}^2\, C_{po}\,\frac{p_{o\,\rm inl}}{p_{o\,\rm ideal}} \tag{7}$$

The first approximate form (6) is valid for $M_{\rm inl}^2 C_{po} \ll 1$, and the quite simple second approximate form (7) is valid for $M_{\rm inl}^2 \ll 1$. These requirements can usually be met simply by specifying a sufficiently low $M_{\rm inl}$ when computing effectively-incompressible cases. Note that if any upstream work which is present results in a total pressure rise which is $\Delta p_o = \mathcal{O}(\rho_{\rm inl}V_{\rm inl}^2)$, then we have

$$\frac{p_{o\,\rm inl}}{p_{o\,\rm ideal}} \;=\; 1 \,+\, \mathcal{O}(M_{\rm inl}^2) \tag{8}$$

and thus in such cases this somewhat inconvenient factor can be ignored in the above ΔS expressions.

3.3 Continuity equation

The MTFLOW formulation does not explicitly solve the differential continuity equation, but instead explicitly enforces a constant mass flow m along each streamtube. With the q definition above, the continuity equation

$$m \;=\; \rho q A (2\pi r - B T_\theta)$$

is then automatically satisfied.

3.4 Meridional streamwise momentum equation

3.4.1 Momentum-conserving form.

The streamwise momentum equation has the form

$$dp \;+\; \rho q \, dq \;+\; \rho V_\theta \, dV_\theta \;+\; p \, d(\Delta S) \;-\; \rho \Omega \, d\bar{\Gamma} \;=\; 0 \tag{9}$$

where differentials $d(\,)$ are taken along a streamtube. This equation is actually differenced in a finite-volume conservative form, giving correct shock capturing.

3.4.2 Entropy-conserving form.

An alternative to solving equation (9) is to solve the convective entropy equation.

$$-p \, d\tilde{s} \;+\; p \, d(\Delta S) \;+\; \rho \, d(\Delta H) \;=\; 0 \tag{10}$$

With the definition of the entropy $\tilde{s}$ given earlier, and with the total enthalpy equation to be presented shortly, equations (9) and (10) are formally equivalent in differential form. However, the two equations have different truncation errors on finite grids and also have different shock-capturing properties. Which equation is actually used is controlled by the `Smom` flag to be described later.

3.5 Meridional quasi-normal momentum equation

The complement to the streamwise momentum equation (9) is the quasi-normal momentum equation. This has the same form as (9), except that the changes $d(\,)$ are taken along a quasi-normal grid line. This equation is also differenced in a conservative manner, to allow proper shock capturing.

3.6 Energy equation

The following form of the energy equation used here.

$$dh_o \;-\; d(\Delta H) \;-\; \Omega \, d\bar{\Gamma} \;=\; 0 \tag{11}$$

For cases where the heat addition ΔH and swirl work $\Omega\, d\bar{\Gamma}$ are zero, this equation reduces to an algebraic statement of constant total enthalpy,

$$h_o \;=\; h_{o\,\mathrm{inl}}$$

which allows the energy equation to be removed from the system. This gives a considerable solution speedup for such cases.

3.7 Tangential momentum equation

One of several forms of the tangential momentum equation can be solved at any given point, depending on the specified field parameters at that point.

3.7.1 Swirl convection equation with forcing

The tangential momentum equation is

$$d\bar{\Gamma} \;-\; d(\Delta G) \;=\; 0 \tag{12}$$

where $d(\,)$ is a change along a streamtube. For intervals where no ΔG forcing is prescribed, this simply preserves the swirl along the streamtube.

3.7.2 Blade row flow-tangency

For modeling blade rows with a known geometry rotating at a known rotation speed Ω, the most realistic constraint on $\bar{\Gamma}$ is to impose a blade row flow-tangency condition, via the specified blade camberline slope $\mathcal{S}_{\mathrm{rel}}$. The equation actually solved is a rate equation for the swirl,

$$\frac{d\bar{\Gamma}}{ds} \;=\; k\,\left(\bar{\Gamma}_o - \bar{\Gamma}\right) \tag{13}$$

$$\bar{\Gamma}_o \;=\; \left(q\mathcal{S}_{\mathrm{rel}} + \Omega r\right) r \tag{14}$$

$$k \;=\; \frac{B}{2r} \tag{15}$$

where s is the distance along the meridional streamline, $\bar{\Gamma}_o$ is a high-solidity limiting swirl, and k is a rate constant to give the correct blade loading in the low-solidity limit. These values for $\bar{\Gamma}_o$ and k are derived in Appendix A. The behavior of $\bar{\Gamma}$ as governed by equation (13) is shown in Figure 2.

3.8 Response of flow variables to field parameters

Because the field parameters are defined only inside some number of *stage blocks* in the flow domain (shown in Figure 1 and described later), some switching logic is necessary for the

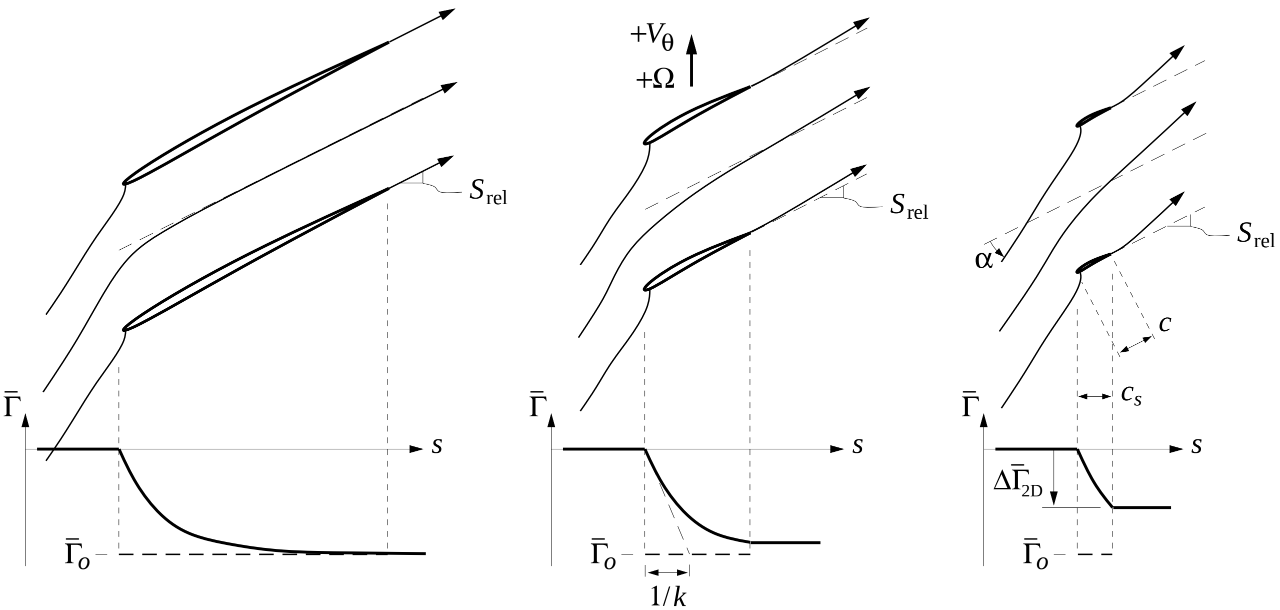


Figure 2: Swirl development along meridional streamline for high, medium, and low solidity cascades. The swirl flowfield variable $\bar\Gamma$ chases the high-solidity limiting swirl $\bar\Gamma_o$ with a rate constant k. Dashed lines indicate the geometric blade slope $\mathcal{S}_{\rm rel}$, which is shown constant here, but will vary along the chord of a cambered blade. The $\Delta\bar\Gamma_{\rm 2D}$ change in the low solidity limit corresponds to a blade-section lift coefficient $C_\ell = 2\pi\alpha$.

discrete equations. To describe this, the simple swirl-convection equation (12) will be used as an example.

The discrete field parameters are defined at each grid cell-center location $\bar x_i, \bar r_i$ (the black dot in Figure 1) as follows:

$$\Delta G_i \;=\; \begin{cases} \Delta G(\bar x_i, \bar r_i) & , \quad \bar x_i, \bar r_i \text{ is inside a stage block} \\ 0 & , \quad \bar x_i, \bar r_i \text{ is not inside a stage block} \end{cases} \tag{16}$$

The swirl-convection equation is then differenced as

$$\bar\Gamma_i - \bar\Gamma_{i-1} \;-\; \Delta G_i + \Delta G_{i-1} \;=\; 0 \quad , \qquad \text{node } i \text{ is inside a stage block} \tag{17}$$

$$\bar\Gamma_i - \bar\Gamma_{i-1} \;=\; 0 \quad , \qquad \text{node } i \text{ is not inside a stage block} \tag{18}$$

where $i\!-\!1$ and i are two successive grid stations along the streamtube. The key feature of this logic is that any field parameter term always appears as a difference along the streamtube, and this difference never straddles the outflow edge of a stage block. The effect is to accumulate the changes across successive stage blocks, as illustrated in Figure 3.

It's important to note that with a finite grid resolution, the overall change in the discrete $\bar\Gamma_i$ solution across a stage block isn't exactly the same as the change in the continuous $\Delta G(x,r)$ function. This error is minimized by using a sufficiently fine grid, and by making the $\Delta G(x,r)$ distribution have a small gradient at the outflow edge. A small outflow gradient is in fact physically realistic, since the blade Kutta condition demands that the loading and hence $d\bar\Gamma/ds$ is zero at the stage outflow.

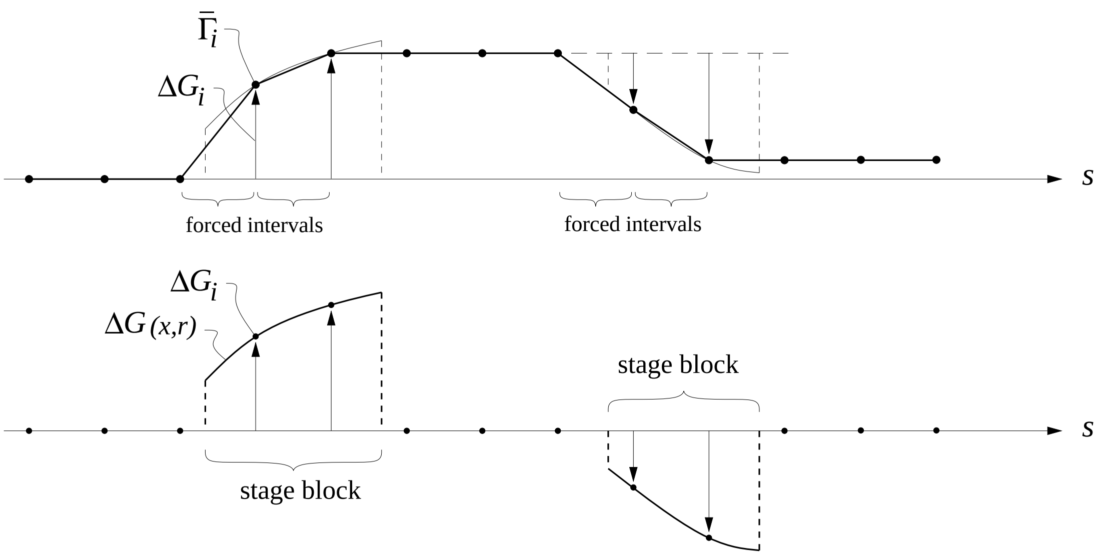


Figure 3: Discrete $\bar{\Gamma}_i$ development along a streamtube, forced by discrete ΔG_i values which are set from the $\Delta G(x,r)$ field parameter distributions defined on stage blocks. An interval is forced only if its downstream node falls inside a stage block.

4 Viscous Flow Quantities and Governing Equations

4.1 Viscous flow modeling

MTFLOW can optionally calculate boundary layers growing on the meridional surfaces together with the inviscid part of the flow. Bodies which end inside the domain will also have viscous wakes trailing from them. The viscous and inviscid flows are strongly coupled via the displacement thickness.

MTFLOW treats only the meridional component of the boundary layer velocities. The assumed axisymmetry of the flow implies that the tangential velocity component decouples, and is ignored inside the boundary layer.

4.2 Viscous flow variables

MTSOL defines the following quantities at the edge of the boundary layer, which enter directly or indirectly into the boundary layer equations. They are directly related to the inviscid flow quantities in the Euler grid streamtube adjacent to the layer.

u_e	edge velocity
h_{o_e}	edge total enthalpy
ρ_e	edge density
μ_e	edge viscosity

Along the boundary layers and wakes, **MTSOL** solves for the following primary viscous variables.

θ	momentum thickness
δ^*	displacement thickness
C_τ	shear stress coefficient (Reynolds stress scale)

The primary variables above also define the following secondary variables.

$$\begin{array}{lcll}
M_e^2 & = & \dfrac{u_e^2}{(\gamma-1)(h_{o_e} - \frac{1}{2}u_e^2)} & \text{edge Mach number} \\
Re_\theta & = & \dfrac{\rho_e u_e \theta}{\mu_e} & \text{momentum-thickness Reynolds number} \\
H & = & \dfrac{\delta^*}{\theta} & \text{shape parameter} \\
H^* & = & \dfrac{\theta^*}{\theta} \;=\; f(H, M_e) & \text{kinetic-energy shape parameter} \\
H^{**} & = & \dfrac{\delta^{**}}{\theta} \;=\; f(H, M_e) & \text{density-flux shape parameter} \\
C_f & = & \dfrac{\tau_w}{\frac{1}{2}\rho_e u_e^2} \;=\; f(H, M_e, Re_\theta) & \text{skin friction coefficient} \\
C_{\mathcal{D}} & = & \dfrac{\mathcal{D}}{\rho_e u_e^3} \;=\; f(H, M_e, Re_\theta, C_\tau) & \text{dissipation coefficient}
\end{array}$$

The local effective circumference is

$$b \;=\; 2\pi r \;-\; BT_\theta \;+\; \pi\delta^* \, \hat{n}\cdot\hat{r} \tag{19}$$

which accounts for the radial offset due to the local displacement thickness via the last term. This offset is typically significant only in the axis-body wake:

$$b \;=\; \pi\delta^* \qquad\qquad \text{(in wake on axis)} \tag{20}$$

4.3 Viscous flow equations

The primary viscous variables are governed by integral momentum and kinetic energy equations, augmented by a shear-stress coefficient evolution equation.

$$\frac{d\theta}{\theta} \;=\; \frac{C_f}{2}\,\frac{ds}{\theta} \;-\; (H+2)\,\frac{du_e}{u_e} \;-\; \frac{d\rho_e}{\rho_e} \;-\; \frac{db}{b} \tag{21}$$

$$\frac{dH^*}{H^*} \;=\; \left(\frac{2C_{\mathcal{D}}}{H^*} - \frac{C_f}{2}\right)\frac{ds}{\theta} \;+\; \left(H - 1 - \frac{2H^{**}}{H^*}\right)\frac{du_e}{u_e} \tag{22}$$

$$\frac{dC_\tau}{C_\tau} \;=\; K_c\left(C_{\tau_{EQ}}^{1/2} - C_\tau^{1/2}\right)\frac{ds}{\delta} \;+\; 2\left(\frac{\delta}{u_e}\frac{du_e}{ds}\right)_{EQ}\frac{ds}{\delta} \;-\; 2\frac{du_e}{u_e} \tag{23}$$

All the logarithmic differentials $d()/()$ in (21) are equivalent to $d(\rho_e u_e^2 \theta b)/(\rho_e u_e^2 \theta b)$, so that the total momentum defect area $\rho_e u_e^2 \theta b$ is the conserved variable. Similarly, equations (21) and (22) together conserve the kinetic energy defect area $\rho_e u_e^3 \theta^* b$. The three boundary layer equations are discretized and solved simultaneously with the inviscid equations.

5 Input Files

The principal input files needed to set up an MTFLOW solution case are listed below.

5.1 Geometry coordinate file `walls.xxx`

The geometry consists of a number of axisymmetric bodies, airfoil "rings", and/or walls defined in the meridional x, r plane. These elements are numbered sequentially outward, starting from the body on the axis (if present). The domain can extend to an outermost "pipe" body, or to some sufficiently large radius simulating an infinite flowfield. The inlet and outlet planes are perpendicular to the axis, and are defined by two x locations.

Figure 4 shows the geometry layout, as specified in the `walls.xxx` file. It contains the domain layout parameters and the discrete x, r geometry coordinates, and is used by the initialization program **MTSET** to define the domain with the initial streamline grid. This file has the following structure, with all quantities defined in units of some arbitrary length L_{ref}.

```
Name
XINL XOUT YBOT YTOP
X(1,1) Y(1,1)          ⟵ Element 1
X(2,1) Y(2,1)
X(3,1) Y(3,1)
.     .
.     .
X(I,1) Y(I,1)
999.  999.         ⟵ Start new element definition
X(1,2) Y(1,2)          ⟵ Element 2
X(2,2) Y(2,2)
X(3,2) Y(3,2)
.     .
.     .
X(I,2) Y(I,2)
```

The file contents are described below.

`Name` is the name of the case, not more than 32 characters.

`XINL` is the inlet-plane x/L_{ref} location.

`XOUT` is the outlet-plane x/L_{ref} location.

`YBOT` is the inner radius r/L_{ref} of the domain. Must be `0.0` if the first element is located on the axis.

`YTOP` is the radius r/L_{ref} of the outer boundary for infinite-flowfield cases. Must be `0.0` if the outer boundary is specified by the last element.

The geometry coordinates `X(1,1),Y(1,1)` through to `X(I,N),Y(I,N)` are the x/L_{ref}, r/L_{ref} coordinates of the element surfaces, going counterclockwise for each element, as shown in Figure 4 for internal and external flow situations. Currently, the inlet and outlet planes are assumed to be at constant x. The next MTFLOW version will likely allow general $x(t), r(t)$ inlet and outlet plane shapes for radial-flow situations.

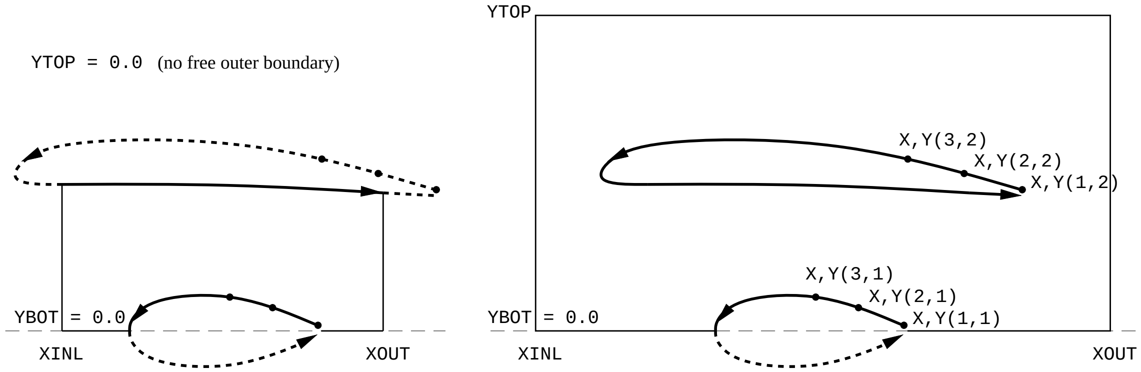


Figure 4: Coordinate directions for internal flow and external flow cases. Dotted portions of shapes can be specified, but are not used.

If the axis body (element 1) has a spinner-type leading edge, it is best to specify the entire symmetric shape for both positive and negative radii with the counterclockwise ordering, so that a smooth spline is used around the leading edge. If the axis body extends from the inlet to the outlet, then the negative radii must be omitted, but the ordering must still be "counterclockwise" from the outlet to the inlet as though the negative side existed.

If the outer boundary is a solid-wall pipe, it must extend all the way from the inlet to the outlet. The outside surface of this outer body can be included in the coordinates, but only the inner surface will be used. If only the inner surface is specified, the ordering is "counterclockwise" from inlet to outlet as though the outside surface existed.

The interior elements, if any, must be small enough to fit entirely inside the domain between the innermost and outermost radii, and between the inlet and outlet planes. Note that an outer element like a nacelle body can be used either as an outer wall for internal-flow domains by setting `YTOP = 0.0`, or as a regular interior body for external-flow domains by setting `YTOP` sufficiently large. It the outer boundary is a free-jet surface at constant pressure, then `YTOP` must be the actual radius of this outer boundary. If the actual flow domain is infinite, then `YTOP` must be sufficiently large to make the subsonic source+doublet farfield approximation accurate. Transonic flows require larger domains, since the flow disturbance decays radially more slowly, and any local supersonic zones must be fully enclosed.

A blunt trailing edge on an interior element is specified by leaving the element "open", so that the first and last coordinate points do not coincide. If the actual element shape has a semi-circular trailing edge, it must be "cut off" near the tangency points. The Kutta condition is applied between these two points. MTFLOW incorporates a blunt trailing edge model which accounts for the additional losses associated with a blunt trailing edge.

Inside all the programs, the element shapes are defined analytically as cubic splines using the arc length in the x, r plane as the spline parameter. Constant-curvature end conditions (zero third derivative) are used. A slope break can be specified anywhere on an element by making the successive coordinate pairs `X(i,n),Y(i,n)` and `X(i+1,n),Y(i+1,n)` identical. A separate spline will then be computed on each side of this doubly-specified point.

5.2 Imposed field parameter file `tflow.xxx`

5.2.1 Overview

This is an optional formatted file which specifies *field parameters* which enter as forcing terms in the flowfield equations. The field parameters, described below, are input as functions of the meridional coordinates x, r, defined on *stage blocks*. Internally, all the parameters, including x and r, are stored as bicubic splines in s, t on each stage block. A suitable s, t grid for each stage block is computed internally, and does not need to be specified.

B Number of blades in a blade row. This is simply used to multiply T_θ to get total blockage area. It also determines the cascade pitch for optional blade-to-blade definition file output.

Ω Rotation rate of a blade row, specified in units of $V_{\rm inl}/L_{\rm ref}$.

$T_\theta(s,t)$ Circumferential blade thickness, in units of $L_{\rm ref}$. Total circumferential thickness is BT_θ, which must be less than $2\pi r$ which is the limit of complete blockage.

$\mathcal{S}_{\rm rel}(s,t)$ This is the geometric blade slope $d\theta/dm'$ in the blade-to-blade m'–θ plane, which can be specified in lieu of the added-swirl field ΔG. It can be used to model a cascade row with known geometry, which forces the tangential/meridional flow slope to its passage direction. This will produce some swirl change $d\bar{\Gamma}$, and a corresponding total enthalpy change of $\Omega\, d\bar{\Gamma}$ along a streamline.

$\Delta G(s,t)$ Swirl ($\bar{\Gamma}$) change, specified in units of $V_{\rm inl}L_{\rm ref}$. Alternative to $\mathcal{S}_{\rm rel}$. This will also result in a local total enthalpy change of $\Omega\, d(\Delta G)$ along a streamline.

$\Delta H(s,t)$ Total enthalpy change due to heat release or removal, specified in units of $a_{\rm inl}^2$. Also results in an entropy change $(\rho/p)\, d(\Delta H)$ along a streamline. Must not be cumulatively more negative than $-h_{o\,\rm inl}$, since this will result in the stagnation temperature being less than absolute zero.

$\Delta S(s,t)$ Added entropy due to some adiabatic throttling or frictional mechanism, specified dimensionless. This entropy change adds to the entropy change resulting from ΔH. Physically, ΔS must monotonically increase downstream.

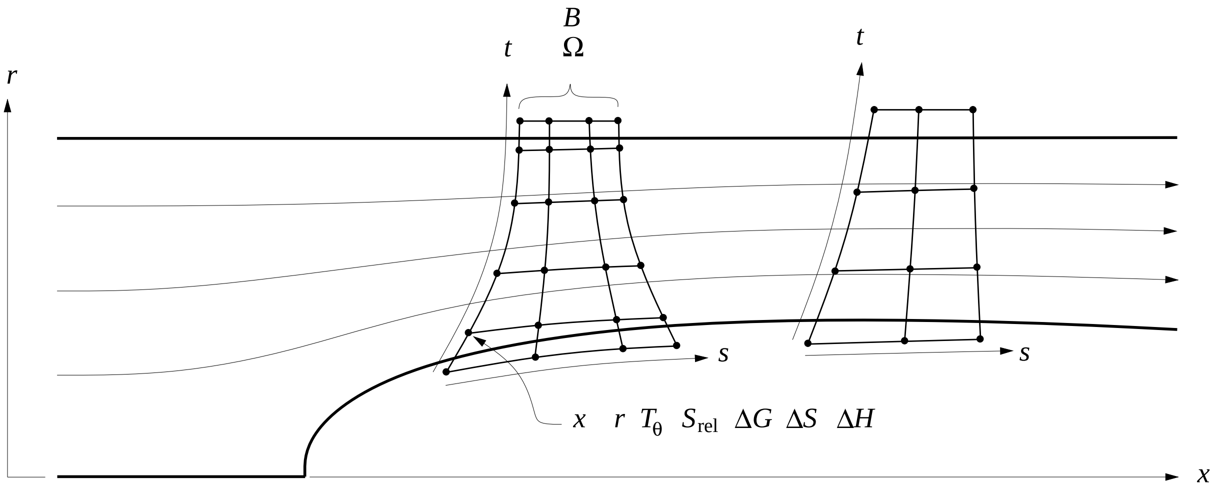


Figure 5: Field parameter s, t grids for two stage blocks. Any flowfield point inside such a grid receives the grid's local specified field parameter values.

5.2.2 File format

The prescribed distributions of ΔH, ΔS, etc., are specified in the file `tflow.xxx` as an arbitrary number of data tables, one table for each stage block. Two such data tables are shown in Figure 5. The file is keyword-driven, and has the format shown and described below. Any parameters which are absent default to zero.

```
NAME                    ! keyword
  Configuration name    ! configuration name string
END                     !

STAGE                   ! keyword: begin stage 1 block data

SNAME                   ! keyword
  Front fan             ! stage name string
END                     !

SINDEX                  ! keyword
  3                     ! stage index value
END                     !

NBLADE                  ! keyword
  25                    ! number of blades
END                     !

OMEGA                   ! keyword
  3.500                 ! rotation speed
END                     !
```

```
DATYPE                  ! keyword
   x  r  T  Sr          ! SECTION data to be given for this stage
* 1.0  1.0  10.0  1.      ! multiplier line  (optional, begins with *)
+ 0.2  0.   0.    0.      ! adder line       (optional, begins with +)
END

SECTION                                      ! keyword
  X(1,1,1) Y(1,1,1)  THK(1,1,1) SREL(1,1,1)
  X(2,1,1) Y(2,1,1)  THK(2,1,1) SREL(2,1,1)
     .        .           .           .
  X(I,1,1) Y(I,1,1)  THK(I,1,1) SREL(I,1,1)
END

SECTION                                      ! keyword
  X(1,2,1) Y(1,2,1)  THK(1,2,1) SREL(1,2,1)
  X(2,2,1) Y(2,2,1)  THK(2,2,1) SREL(2,2,1)
     .        .           .           .
  X(I,2,1) Y(I,2,1)  THK(I,2,1) SREL(I,2,1)
END
  .
  .
  .

SECTION                                      ! keyword
  X(1,J,1) Y(1,J,1)  THK(1,J,1) SREL(1,J,1)
  X(2,J,1) Y(2,J,1)  THK(2,J,1) SREL(2,J,1)
     .        .           .           .
  X(I,J,1) Y(I,J,1)  THK(I,J,1) SREL(I,J,1)
END

END                                          ! end stage 1 data


STAGE                                        ! keyword: begin new stage 2 block data

SNAME                   ! keyword
  Fan stator            ! stage name string
END                     !

SINDEX                  ! keyword
  4                     ! stage index value
END                     !
```

```
NBLADE                  ! keyword
  17                    ! number of blades
END                     !

OMEGA                   ! keyword
  0.                    ! rotation speed
END                     !

DATYPE                  ! keyword
   x  r  T  Sr          ! SECTION data to be given for this stage
END                     !

# x/L      r/L       T/L        S_rel        ! comment line begins with  #  or  !

* 1.0      1.0       10.0       1.           ! multiplier line
+ 0.5      0.        0.         0.           ! adder line

SECTION                                      ! keyword
  X(1,1,2) Y(1,1,2)  THK(1,1,2) SREL(1,1,2)
  X(2,1,2) Y(2,1,2)  THK(2,1,2) SREL(2,1,2)
     .        .           .           .
  X(I,1,2) Y(I,1,2)  THK(I,1,2) SREL(I,1,2)
END

SECTION                                      ! keyword
  X(1,2,2) Y(1,2,2)  THK(1,2,2) SREL(1,2,2)
  X(2,2,2) Y(2,2,2)  THK(2,2,2) SREL(2,2,2)
     .        .           .           .
  X(I,2,2) Y(I,2,2)  THK(I,2,2) SREL(I,2,2)
END
  .
  .
  .

SECTION                                      ! keyword
  X(1,J,2) Y(1,J,2)  THK(1,J,2) SREL(1,J,2)
  X(2,J,2) Y(2,J,2)  THK(2,J,2) SREL(2,J,2)
     .        .           .           .
  X(I,J,2) Y(I,J,2)  THK(I,J,2) SREL(I,J,2)
END

END                                          ! end stage 2 data
```

The contents of this file are described below. Most of the quantities were defined earlier. Note that the kinematic variables Ω and ΔG are taken to be normalized with $V_{\rm inl}$, while the thermodynamic variable ΔH is normalized with $a_{\rm inl}$. The ΔS variable is always dimensionless.

Blank lines in the file are ignored, except where they denote the end of data for one s station . Within each stage data table, the number `J` of radial t points must be the same for each of the `I` streamwise s stations, i.e. the s-t grid must be logically rectangular. But different stages can have different `I`×`J` dimensions.

The parameter-keyword line, which is the second non-comment data line after the `STAGE` keyword, indicate the columns in which field parameters are being specified in the subsequent data table. The recognized keywords must be one of the following strings,

```
 x  r  T  Sr  DG  DS  DH
```

and are implemented in the `TFKEY.INC` include file. The `x` and `r` keywords and data are required for each stage. Any or all of the others are optional. The order of the keywords and corresponding data columns is arbitrary.

`NAME` Name-block keyword

`Label` Arbitrary label text string.

`END` Terminator for all blocks

`...  !` All data after the "`!`" is ignored.

`# ...` Same as "`!`", but only if it appears in the first column.

`STAGE` Stage-block keyword

`Kstage(k)` Number of the rotor or stator stage which is described by this profile block. Used only to enable generation of streamsurface-definition files for MISES blade-to-blade calculations. Ignored if such files will not be generated.

`Nblade(k)` is the blade number B used to multiply T_θ to obtain the total area blockage.

`Omega(k)` is the rotation rate $\Omega\, L_{\rm ref}/V_{\rm inl}$.

`x r T Sr` Parameter keywords, as described above.

`DG DS DH` Additional parameter keywords, as described above.

`* 1.0 ...` Multiplier line. Sets multipliers which are applied to all subsequent stage data. More than one multiplier line can be used, and can appear anywhere within the stage block after the parameter-keyword line. Any such multiplier line overrides the previous multiplier line. All multipliers initially default to unity.

`+ 0.2 ...` Adder line. Sets adders which are applied to all subsequent data. More than one adder line can be used anywhere within the file after the label line. All adders initially default to zero.

The multipliers are applied before the adders:

$$\text{Data\_result} = \text{Data} * \text{multiplier} + \text{adder}$$

(blank) Blank lines within the s-t data delimit s-stations

`X(i,j,k)` is the x/L_{ref} coordinate.

`Y(i,j,k)` is the r/L_{ref} coordinate.

`THK(i,j,k)` is the blade thickness T_θ/L_{ref}.

`SREL(i,j,k)` is the geometric blade slope $\mathcal{S}_{\text{rel}}$

`DG(i,j,k)` is the added swirl $\Delta G/V_{\text{inl}}L_{\text{ref}}$.

`DS(i,j,k)` is the added entropy ΔS from some adiabatic loss process.

`DH(i,j,k)` is the added enthalpy $\Delta H/a_{\text{inl}}^2$ from heat release.

The `i,j,k` indices above denote s stations, t stations, and stages, respectively.

As mentioned earlier, each variable is interpolated via bicubic splines, in the dummy parameters s, t which are computed internally. The link between the arbitrary s, t spline parameters and physical space is provided by the geometric $x(s,t)$ and $r(s,t)$ spline functions, which are numerically inverted on demand and thus define the inverse functions $s(x,r), t(x,r)$. These then allow any flow quantity to be evaluated as a function of the physical coordinates x, r, e.g. $\Delta H(x, r) = \Delta H(s(x,r), t(x,r))$.

For reliable spline evaluations, the grid for each block must be reasonably spaced to avoid spline overshoots. Highly irregular grid point spacings should be avoided. Program **MTFLO** can be used to plot these functions in their splined form to check for overshoots.

Locating a physical grid point in the s, t spline grid requires an interval-halving search through the x, r spline fields. Hence, for evaluation speed it is best if the grids are not excessively dense. An 5 × 10 or smaller grid is frequently sufficient for defining a typical turbomachinery stage.

To give more flexibility in splining, a slope break or discontinuity in s can be specified via two identical s-value lines in each block. However, the flow solver may not tolerate excessive s-discontinuities in some of the variables, such as the blockage T_θ or an actuator-disk jump ΔG. If convergence trouble appears, it is advisable to replace any such s-discontinuity with a short high-gradient region, which is often more physically realistic anyway.

6 Program Descriptions

The descriptions for running each program are given below. Starting from scratch, the usual program execution sequence is

```
% mtset xxx
% mtflo xxx
% mtsol xxx
```

with the necessary input and output files for each step shown on the MTFLOW Roadmap data flow diagram. If there is no `tflow.xxx` file to be imposed, the **MTFLO** run can be skipped. If this is a turbomachinery case, MISES blade-to-blade executions might be interspersed between repeated **MTFLO** and **MTSOL** runs.

6.1 MTSET

6.1.1 Basic Initialization

MTSET is the program which initializes the grid, the densities and a variety of other variables. It requires the `walls.xxx` geometry description file. **MTSET** is menu-driven to allow the user to iteratively generate a good initial grid by tweaking a small number of gridding parameters.

Upon startup, **MTSET** reads the `walls.xxx` file, and begins the grid generation procedure. If there are elements with blunt leading edges present, each element is displayed with a proposed surface grid node distribution, and a brief menu allowing this distribution to be modified.

```
   D sLE/dsAvg, dsTE/dsAvg  spacing ratios
   C urvature exponent
   U pper side spacing refinement           |   B lowup
   L ower side spacing refinement           |   R eset plot

 Change what? (<Return> if spacing OK):
```

The local arc-length increment Δs between two surface grid nodes is determined from

$$\Delta s \sim \frac{1}{1 + a|\kappa|^b}$$

where κ is the local surface curvature. In regions of high curvature, the spacing is therefore smaller, depending on the curvature exponent b and the coefficient a. The exponent is specified directly from the menu described below. A large exponent ($b = 2$, say), makes the spacing small in high-curvature regions. A small exponent ($b = 0.05$, say), makes the spacing more nearly uniform everywhere. The curvature coefficient a is indirectly controlled by specifying Δs spacing at the leading edge (or stagnation point, to be more precise) of the blade. Note that if κ is rather small at the stagnation point, the effect of a is largely shut off in the expression above. If necessary, a fudged additional curvature is added locally very near the stagnation point to allow the spacing requirement to be met. A message is printed when this action is taken. Fudged curvatures are also introduced near the trailing edge, and optionally at selected local-refinement zones on the upper and lower surfaces. The aim is also to control the local spacing.

For each change request, the current values are displayed after the prompt. Selecting "`U`", for example, might produce

```
 Enter new upper s/smax limits, local/avg density ratio  0.1000  0.2500   0.800
```

and just hitting <Return> will take the current values as the default input. One can change only some of the required three inputs by using commas. Entering

```
0.15
```

will only change the first value from 0.1 to 0.15, while entering

```
,,0.5
```

will only change the third value from 0.8 to 0.5. The first two "`s/smax`" values specify the fractional arc length from stagnation point to the trailing edge where the local refinement is to be placed. Tick marks inside the airfoil element contour indicate this fractional arc length in increments of 0.10. The local/avg density ratio specifies the increase in local grid density over the average density which would occur with all points spaced equally.

After the spacing parameters are altered, the new distribution is generated and displayed. The actual LE, TE, max, min, spacing ratios are also printed out. It must be mentioned that only the stagnation point spacing ratio "`dsLE/dsAvg`" can be controlled precisely with the input parameters. The other spacing ratios are approximate and may need to be iterated.

Once a good node distribution on each element is obtained, **MTSET** proceeds to modify all the spacings to resolve any conflicts between facing airfoil or wall surfaces. Basically, all the distributions are automatically fudged to make spacings on element surfaces facing each other match as closely as possible. This prevents massive grid shearing which would otherwise occur. Once all the node distributions are finalized, intermediate streamline nodes are generated in the flowfield interior by simple linear interpolation from the stagnation and farfield streamlines. An elliptic SLOR grid smoother is then invoked to "clean up" the linearly interpolated grid. This eliminates all kinks, overlaps, and also makes all the grid streamlines correspond to streamlines of incompressible inviscid flow. This is then an excellent initial guess for the **MTSOL** solver.

6.1.2 Initial solution file output

After the grid is smoothed, the top-level option `W` can be issued to write out the initial solution file `tdat.xxx` which is then ready for the **MTFLO** field parameter specification program and/or the **MTSOL** solver. The description of this file can be found from the comments in `STATE.INC`, which declares all the variables in COMMON blocks. `STATE.INC` also contains all the primary array dimension limits which can be changed if necessary.

6.1.3 Grid parameters

If the grid is not satisfactory, the top-level option `M` puts up the grid parameter modification menu

```
   Current grid parameters:
 N      101   number of streamwise grid points            E
 E    0.800   exponent for airfoil side points: n = N*chord
 X    0.900   x-spacing parameter
 J        1   manual j-spacing flag...
S1       12    number of streamlines between airfoils 1,2
S2       10    number of streamlines between airfoils 2,3
T1     0.300   airfoil 1 surface streamtube thickness factor
T2     0.050   airfoil 2 surface streamtube thickness factor
T3     1.500   airfoil 3 surface streamtube thickness factor

Change what (<return> if done)?:
```

which allows each gridding parameter to be changed by issuing the parameter's menu character and new value as the argument. For example, typing

```
 N 121
```

will change the number of streamwise grid points to 121. A prompt will be produced if the numerical argument is omitted.

The "X-spacing" parameter controls the repelling-force between the quasi-normal grid lines during the SLOR smoothing phase.

The `Sn` parameter sets the number of streamlines between airfoil elements `n` and `n+1`, while the `Tn` parameter controls the thickness of the streamtubes adjacent to element `n`. Both `Sn` and `Tn` therefore control the streamline bunching distribution. However, a j-spacing flag of `0` overrides everything and forces a uniform streamline spacing.

6.1.4 Grid parameter saving, recall

Once a good set of gridding parameters is obtained, including the surface spacing parameters, they can all be saved to `mtgpar.xxx` by specifying Option S at any time. If this file already exists, it is overwritten. `mtgpar.xxx` will then be automatically read when **MTSET** is executed again for that same xxx case, which causes all the gridding parameters to take on their saved values. This allows rapid generation of grids for cases which differ only slightly, since the same gridding parameters can then be used.

If the trailing edge of the element is not closed, a constant-thickness wake gap is left extending from the element base. In inviscid calculations, this gap remains constant in width, but is free to move up and down so that it sustains no pressure jump and hence no lift. For viscous calculations, the gap will collapse down to the local wake displacement thickness, and is still free to move up or down. A special treatment is used to correct for the dead air region immediately behind the blunt base whose length and shape is set in **MTSET** to match experimental correlations. This special treatment results in an increase in momentum thickness downstream and accurately accounts for "base drag".

6.2 MTSOL

MTSOL is the program that solves the Euler equations. The solution is read in from the required input file `tdat.xxx`. After being interactively converged by the user, the solution can be written back to `tdat.xxx`. Thus the input file `tdat.xxx` can either be a restart file from an old calculation, or a new file created by **MTSET** .

The **MTSOL** top-level menu is:

```
  M odify solution parameters     W rite solution state file
 eX ecute calculation             R e-read solution state file

  O ptions                        Z oom
  P lotted-variable select        U nzoom
  H ardcopy current plot          A nnotate plot

  F orce summary                  C oordinate file output
  Q uit                           E lliptical grid smoother

 MTSOL   c>
```

6.2.1 Solution parameter modification

Before a solution is attempted, it may be necessary to set or change some solution parameters from their default values. This is done via the `M` command at top level, which produces the following sub-menu:

```
 1   mass flow              -1   set inlet Mach
 2 * mass differences       -2 * set TE Kutta
 3 * LE movement            -3 * set LE Kutta
 4   geom. mode amplitudes   -4   set modes fixed
 5   airf. move amplitudes   -5   set moves fixed
 6   Pspec mode amplitudes   -6   fix segment endpoints
 7   grid-exit pressure      -7   set outlet P/Po

 M  0.1000   inlet Mach       R  0.000e6 Reynolds No.
 P  0.9930   outlet P/Po      N  7.000   Ncrit
 C  0.980    Mcrit            K -1.000   MUcon
 W  0.250    PcWt             X  0.048   Xcff
 S  4        Smom type        F  2       Farfield type

 T3 *     1.0000    element 2 upper trip x/c
 T4 *     1.0000    element 2 lower trip x/c
 T1 *     1.0000    element 1 upper trip x/c

 V1,2..   Viscous side toggles

Change what?   c>
```

The topmost items labeled with positive integers are the available global variables, each of which can be toggled active/inactive by typing its integer. The items labeled with negative integers are global constraints, each of which is likewise are toggled on/off by typing its negative integer. Currently-active variables and constraints are indicated by an asterisk "*".

In general, only active variables will be changed during the calculation, and inactive variables will be held fixed. Each active variable requires one active constraint to implicitly determine its value. In the example above, the mass flow (1) is implicitly determined by the outlet pressure constraint (-7). Alternatively, the mass flow could have been determined via the specified inlet Mach number constraint (-1).

The mass flow variable (1) is treated specially. If it is "inactive", it will be set *explicitly* from the inlet Mach number value shown on the menu. Hence, it is not necessary to set the (1) and (-1) flags to change the Mach number. This is done both for convenience and for necessity — unbounded-flow cases must have the (1),(-1) flags inactive to give a well-posed problem.

The presence of internal airfoil elements requires that variables (2) and (3), and constraints (-2) and (-3) be active, and this will be the startup state generated in **MTSET** . If the TE Kutta condition is not imposed, a severe grid distortion will occur at the trailing edge as the flow tries to curl around and move the dividing streamline from the trailing edge point up onto one side of the airfoil. The lack of an LE Kutta condition is less problematic, but it will cause the leading edge stagnation point to end up at the wrong location.

The letter commands can be issued with one or more arguments to change the associated value. For example, the command

```
 M 0.2
```

will change the inlet Mach number to 0.2, and if the value argument is omitted, a prompt will be issued:

```
 M
Enter specified inlet Mach number   r>
```

6.2.2 Parameter descriptions

Mach number — `M` command

The inlet Mach number $M_{\rm inl}$ is used to define all the other inlet reference quantities from the assumed internal reference stagnation values.

$$
\begin{aligned}
\rho_{\rm inl} &= \rho_{o\,\rm inl}\left(1+\tfrac{\gamma-1}{2}M_{\rm inl}^2\right)^{-1/\gamma-1} \\
p_{\rm inl} &= p_{o\,\rm inl}\left(1+\tfrac{\gamma-1}{2}M_{\rm inl}^2\right)^{-\gamma/\gamma-1} \\
h_{\rm inl} &= h_{o\,\rm inl}\left(1+\tfrac{\gamma-1}{2}M_{\rm inl}^2\right)^{-1} \\
V_{\rm inl} &= a_{o\,\rm inl}\left(1+\tfrac{\gamma-1}{2}M_{\rm inl}^2\right)^{-1/2} M_{\rm inl}
\end{aligned}
$$

Given the fixed internal reference stagnation quantities, the total mass flow m is then related to $M_{\rm inl}$ as follows.

$$
m(M_{\rm inl}) = \rho_{\rm inl}\, V_{\rm inl}\, A_{\rm inl}
$$

This relation is always satisfied, so that $M_{\rm inl}$ is best thought of as controlling the mass flow rather than the actual Mach number at the inlet plane. In most cases it is true that $M(x_{\rm inl}) = M_{\rm inl}$, but if significant loss ΔS and/or heat release ΔH are specified at the inlet, then $M(z_{\rm inl})$ may be quite different from $M_{\rm inl}$.

In the mass flow relation above, $A_{\rm inl}$ is the inlet-plane flow area.

$$
A_{\rm inl} = \pi\left(\texttt{YTOP}^2 - \texttt{YBOT}^2\right)
$$

If `YTOP` and/or `YBOT` are specified as `0.0` (i.e. unspecified) in `walls.xxx`, then these value are obtained from the splined input geometry shapes at $x = \texttt{XINL}$.

Outlet pressure — `P` command

The normalized outlet pressure is defined as a mass average over the outlet plane.

$$
p_{\rm out}/p_{o\,\rm inl} = \frac{1}{m}\int p/p_{o\,\rm inl}\, dm
$$

This can be imposed in lieu of the inlet Mach to determine the mass flow. If the flow chokes the outlet pressure *must* be imposed, since the inlet Mach is then fixed. This switch in boundary conditions will be performed automatically if choking is detected, so that the appropriate $p_{\rm out}/p_{o\,\rm inl}$ value should be set if there is any possibility of choking occurring.

Reynolds number — `R` command

A viscous case is selected by a nonzero inlet Reynolds number, defined as

$$Re_{\rm inl} \;\; = \;\; \rho_{\rm inl}\, V_{\rm inl}\, L_{\rm ref} \,/\, \mu_{\rm inl}$$

where $\mu_{\rm inl}$ is the viscosity at the same flow conditions as the other inlet quantities, computed using Sutherland's Law as described earlier. If the Reynolds number of a previously-converged viscous solution is reset to zero, the viscous displacement thickness distributions will be held "frozen" at their current values.

Critical amplification factor — `N` command

The amplification factor $N_{\rm crit}$ is a parameter in the envelope e^n transition prediction method used in **MTSOL**. It in effect controls the ambient disturbance level, with small $N_{\rm crit}$ values implying a large disturbance level and vice versa. A value of $N_{\rm crit} = 9$ gives the "standard" e^9 method.

Trip positions — `T1,T2...` commands

A trip will force the boundary layer to undergo transition regardless of the e^n method result. The trip location is specified in terms of fractional chord x/c, and one trip per boundary layer side is available. A trip location of $x/c \geq 1.0$ implies that there is no trip, and only free transition can occur. The e^n method is always active, and free transition can still occur ahead of the trip.

Viscous side toggles — `V1,V2...` commands

Any airfoil side can be either inviscid or can have a boundary layer. This is toggled for each side with the `V1,V2...` commands. A viscous side is denoted by an asterisk on the trip-location line. All viscous/inviscid side toggles are ignored if the Reynolds number is zero, which forces everything to be inviscid.

Artificial dissipation — `C,K` commands

The artificial dissipation in MTFLOW is a speed-upwinding formulation analogous to bulk viscosity. Instead of the actual speed q, the momentum and/or entropy equations are defined using an upwinded speed $\tilde{q}$ defined by

$$\tilde{q}_i \;=\; q_i \;-\; \mu_i^{(1)} \left(q_i - q_{i-1} \right) \;+\; \mu_i^{(2)} \left(q_{i-1} - q_{i-2} \right)$$

where i is the grid node index along a streamtube, and $\mu_i^{(1)}$, $\mu_i^{(2)}$ are the first- and second-order dissipation coefficients. To maintain numerical stability and allow shock capturing, the following formulas for the dissipation coefficients used, as suggested by a stability analysis.

$$\mu_i^{(1)} \;\;=\;\; \max \left[0 \;,\; \frac{C_\mu}{\gamma} \left(1 - M_{\rm crit}^2 / M^2 \right) \right] \quad , \quad {\rm with}\; M^2 = \tfrac{1}{2}(M_i^2 + M_{i-1}^2)$$

$$\mu_i^{(2)} \;\;=\;\; \begin{cases} \mu_i^{(1)} & ; \;\; \text{2nd-order dissipation} \\ 0 & ; \;\; \text{1st-order dissipation} \end{cases}$$

The stability analysis indicates the following requirements for stability.

$$\begin{aligned}\text{all cases:} &\qquad M_{\text{crit}} \le 1 \\ \text{1st-order dissipation:} &\qquad C_\mu \ge 1/2 \\ \text{2nd-order dissipation:} &\qquad C_\mu \ge 1/4\end{aligned}$$

Violation of these thresholds will produce numerical instability and a nearly-singular Newton matrix. To give some margin of safety, the following values are recommended:

$$\begin{aligned}M_{\text{crit}} &= 0.95 \ldots 0.99 \\ C_\mu &\simeq \pm 1.0\end{aligned}$$

These values are changed with the `C` and `K` commands. A negative input value of C_μ indicates that 1st-order disspation is to be used. Larger values of C_μ can be effective in stabilizing difficult cases with strong shocks.

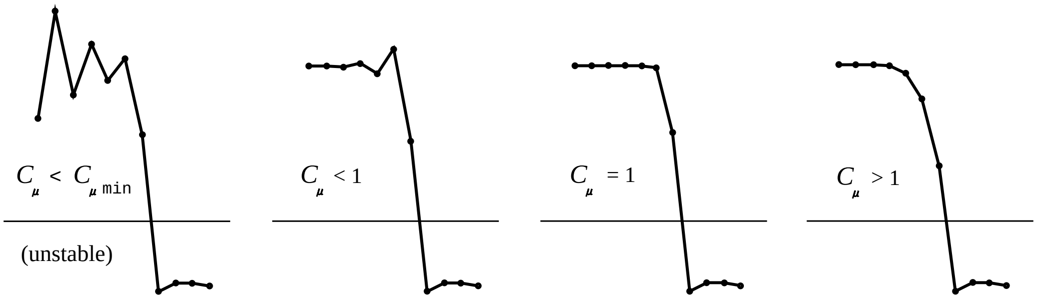


Figure 6: Effect of the dissipation weight C_μ on the numerical structure of a captured shock. Assumes $M_{\text{crit}} \simeq 1$.

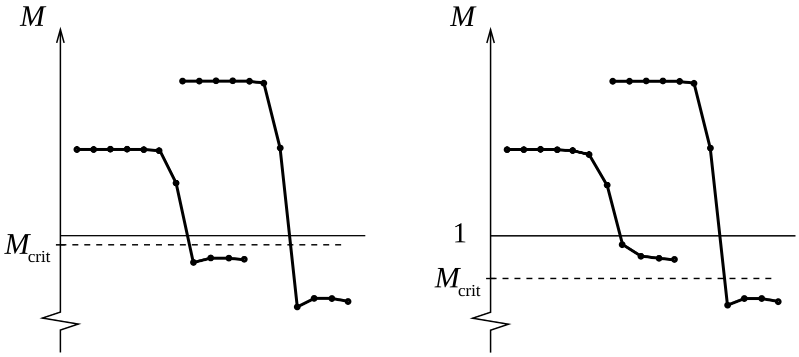


Figure 7: Effect of the dissipation threshold M_{crit} on the numerical structure of weak and strong captured shocks. Assumes $C_\mu \simeq 1$.

Streamwise momentum/entropy conservation — `S` command

The usual streamwise momentum equation can be selectively replaced by the entropy convection equation. This is controlled by the `Smom` flag, which can take on the following values.

`Smom` $= 1 \rightarrow$ use S-momentum equation (9) everywhere

2 → use entropy equation (10) everywhere
3 → use S-momentum equation, replaced by entropy equation
only near the leading edge to minimize truncation errors there.
4 → use entropy equation, replaced S-momentum equation only at shocks
where dissipation is active

Using the entropy equation has the advantage of preventing the generation of spurious losses, although it will also prevent the “correct” loss generation at shocks. Using `Smom = 4` is a good general option which will use the entropy equation where possible, but will automatically revert to the correct S-momentum equation at shocks. This is equivalent to `Smom = 2` if no shocks are present.

The `Smom = 3` option conserves entropy only in fixed regions near the leading edge, where spurious losses would typically occur otherwise. The extent of these regions is hard-wired in SUBROUTINE SETUP (in `setup.f`). This option is a good alternative if there are problems converging with `Smom = 4`, which can be flaky in some situations, especially with choked cases which are extremely sensitive to shock-induced entropy generation.

Farfield type flag — `F` command

Three types of outer boundary conditions are implemented, and can be selected as necessary:

1 Solid wall
2 Unbounded flow (subsonic source+doublet, supersonic wave)
3 Constant pressure (jet boundary)

The source + doublet farfield sets the pressure on the outermost boundary to that of a point source and point doublet on the axis. The strengths of these singularities are determined automatically by the flow solver so that the numerical streamline pattern matches the analytically-derived pattern as closely as possible. For supersonic flows, the local Prandtl-Meyer relation $\theta + \nu(M) = 0$ is imposed, which absorbs the outgoing waves without significant reflection.

The constant-pressure far field sets the pressure on the outermost streamline to the inlet pressure p_{inl} determined from the input value of M_{inl}. The mass flow must *not* be a global variable with this option, otherwise the outer streamline position and the inlet area will be free to “float” since there will be nothing to fix the value of the mass flow.

Farfield singularity location — `X` command

The source and doublet singularities are placed on the axis at the x location specified by this command. The location is initially set in **MTSET** at the volume centroid, and is plotted in **MTSOL** as a red asterisk on the axis. This singularity location is not used if the solid-wall or constant-pressure farfield boundary conditions are selected.

Inverse-side selection toggle — `I` command

Any airfoil surface can be selected to be an *inverse side* on which the pressure is imposed as a boundary condition. This replaces the usual default solid-wall boundary condition, so that the airfoil surface shape will change in response to the specified pressure. The pressure is imposed only over an *inverse segment* delimited by its endpoint indices i_1 and i_2. The inverse segment can cover all or part of the airfoil side, although it is not advisable to put an endpoint at or near a blunt leading edge, as the geometry there is extremely sensitive to the specified pressure. The actual boundary condition which is imposed at surface node i is

$$p_i \;=\; p_{\rm inl} \;+\; \frac{1}{2}\,\rho_{\rm inl}\,V_{\rm inl}^2 \left(C_{p\,{\rm spec}_i} \;+\; A_1 \frac{i_2 - i}{i_2 - i_1} \;+\; A_2 \frac{i - i_1}{i_2 - i_1} \right)$$

where A_1 and A_2 are "Pspec mode amplitude" global variables (6) which must be activated before the inverse calculation is executed. Their corresponding global constraints are the fixed-endpoint conditions (-6), which enforce the geometry to remain continuous at the segment endpoints i_1 and i_2. The A_1 and A_2 coefficients are "errors" added by the solver to the prescribed pressure $C_{p\,{\rm spec}}$ to make it physically admissible.

Prescribed-pressure editing — `E` command

The `E` command invokes the C_p editor sub-submenu:

```
 I nitialize CPspec to CP
 M odify CPspec
 D emark inverse segment
 S lope-matching at segment endpoints (toggle)

 P lot refresh
 Z oom
 U nzoom, reset plot scales

 A nnotation menu
 H ardcopy current plot
 N ew target side

Select edit option:
```

Two C_p distributions are plotted — $C_{p\,{\rm spec}(s)}$ as a solid line, and the actual current wall pressure grid node values C_{p_i} as symbols. If this is the first time this C_p editor is invoked, it is necessary to issue the `I` command to initialize $C_{p\,{\rm spec}}$ to something reasonable. This $C_{p\,{\rm spec}}$ can the be repeatedly modified with the `M` command, which accepts mouse clicks which are then splined, thus generating the modification. The inverse segment endpoints i_1 and i_2 are specified via the `D` command. The `S` toggle controls whether the new modified piece of $C_{p\,{\rm spec}(s)}$ is grafted onto the old distribution with or without slope matching at the endpoints.

6.2.3 Solver execution

The Newton solution convergence procedure is entered with the `X` command, which requests the number of Newton iterations to be performed. Entering "n" causes the following action:

$n = 0$ return back to top level menu
$n \neq 0$ perform n iterations and repeat the question

The X command can be given "n" as an argument, in which case the iteration-number prompt is skipped and the return to the top level menu is automatic.

The requested n is actually an iteration limit, and the Newton cycle may stop earlier if the convergence tolerance is met. After each Newton iteration, the value and i, j location of each maximum flowfield residual is printed out, and the flow solution is plotted. If a viscous case is being run, the transition location on each viscous surface is also displayed.

6.2.4 Solution saving, reading

The current solution state can be written back to the `tdat.xxx` state file at any time with the `W` command. An existing state file can be read in with the `R` command. The latter command has the same effect as stopping and re-starting **MTSOL** , but is slightly more convenient.

6.3 Solution Output

The output of **MTSOL** is the converged or partially-converged flowfield, and is defined entirely by the contents of the state file. Specific quantities of interest are defined below.

6.3.1 Forces

The axial thrust force coefficient C_T is calculated via the x-momentum change integrated over all the streamtubes,

$$C_T = \frac{1}{\frac{1}{2}\,\rho_{\rm inl}\,V_{\rm inl}^2\,L_{\rm ref}^2} \int \Big(u_{+\infty} - u_{-\infty} \Big)\, dm = C_{T_{\rm i}} + C_{T_{\rm v}}$$

$$\text{where} \qquad C_{T_{\rm i}} = \frac{1}{\frac{1}{2}\,\rho_{\rm inl}\,V_{\rm inl}^2\,L_{\rm ref}^2} \sum_j \Big(u_{+\infty_j} - u_{-\infty_j} \Big)\, m_j$$

$$C_{T_{\rm v}} = \frac{1}{\frac{1}{2}\,\rho_{\rm inl}\,V_{\rm inl}^2\,L_{\rm ref}^2} \sum_{BL} - \Big(\rho_e u_e^2\, 2\pi r\, \theta \Big)_{+\infty}$$

where m_j is the streamtube mass flow, and $u_{\pm\infty}$ is the downstream/upstream streamtube velocity which would occur if the streamtube flow were isentropically taken to the reference pressure $p_{\rm inl}$.

$$u_{\pm\infty} = \left\{ 2h_o \left[1 - \left(\frac{p_{\rm inl}}{p_o} \right)^{\frac{\gamma-1}{\gamma}} \right] \right\}^{\frac{1}{2}}$$

The stagnation quantities p_o, h_o are taken from the grid inlet to give $u_{-\infty}$, and from the grid outlet to give $u_{+\infty}$. In effect, an inviscid thrust or drag force $C_{T_{\rm i}}$ can result only from total pressure and/or total enthalpy changes from the inlet to the outlet.

The summation in the discrete expression for C_{T_v} above is the contribution of the body wakes to the total momentum defect, and is calculated by applying the Squire-Young relation to the grid-outlet viscous wake point, which effectively extrapolates the wake momentum defect to far downstream at $+\infty$ where p_inl is assumed to prevail.

$$\Bigl(\rho_e u_e^2 \, 2\pi r \, \theta\Bigr)_{+\infty} \;=\; \Bigl(\rho_e u_e^2 \, 2\pi r \, \theta\Bigr)_\mathrm{out} \times \left(\frac{p_\mathrm{inl}}{p_{e\,\mathrm{out}}}\right)^{(H_\mathrm{out}+H_{+\infty})/(2\gamma M_\mathrm{inl}^2)}$$

H_out is the shape parameter at the grid-outlet plane, and $H_{+\infty}$ is the shape parameter in the far-downstream wake. This is assumed to be

$$H_{+\infty} \;=\; 1 \,+\, (\gamma{-}1) M_\infty^2$$

which is valid for adiabatic walls at near-unity Prandtl numbers.

The total power coefficient C_P is similarly defined via the total enthalpy change integrated over all streamtubes.

$$C_P \;=\; \frac{1}{\frac{1}{2}\,\rho_\mathrm{inl}\, V_\mathrm{inl}^3 \, L_\mathrm{ref}^2} \int \Bigl(h_{o_{+\infty}} \;-\; h_{o_{-\infty}}\Bigr)\, dm \;=\; C_{P_\mathrm{i}} \;+\; C_{P_\mathrm{v}}$$

$$\text{where} \qquad C_{P_\mathrm{i}} \;=\; \frac{1}{\frac{1}{2}\,\rho_\mathrm{inl}\, V_\mathrm{inl}^3 \, L_\mathrm{ref}^2} \sum_j \left(h_{o_{+\infty}\,j} \;-\; h_{o_{-\infty}\,j}\right) m_j$$

$$C_{P_\mathrm{v}} \;=\; \frac{-1}{\frac{1}{2}\,\rho_\mathrm{inl}\, V_\mathrm{inl}^3 \, L_\mathrm{ref}^2} \sum_{BL} - \Bigl(\rho_e u_e^3 \, 2\pi r \, \delta_H\Bigr)_{+\infty} \;\simeq\; 0$$

With adiabatic walls, the enthalpy defect thickness δ_H of a viscous layer is zero everywhere, and so does not contribute.

If the configuration is a propulsor, a propulsive efficiency can be defined as $\eta_P = C_T/C_P$. If the enthalpy addition responsible for C_P is via $\Omega\,\Delta\bar{\Gamma}$ (mechanical work, as in a propeller), then η_p will be the usual propeller efficiency. If the enthalpy addition is via ΔH (heat addition, as in a combustor), the efficiency will be in effect 1/SFC suitably nondimensionalized with the fuel's heating value.

In addition to the overall force and power coefficients defined above, it is useful to calculate the axial force coefficients for the individual airfoil elements via surface force integration.

$$\begin{aligned}
C_{T_\mathrm{element}} \;&=\; \frac{1}{\frac{1}{2}\,\rho_\mathrm{inl}\, V_\mathrm{inl}^2 \, L_\mathrm{ref}^2} \oint 2\pi r \Bigl(p\,dr \;-\; \tau\,dx\Bigr) \\
&=\; \frac{1}{L_\mathrm{ref}^2} \left[\oint C_p \, 2\pi r\, dr \;-\; \oint C_f \, 2\pi r\, dx\right] \\
&=\; C_{T_{p\,\mathrm{element}}} + C_{T_{f\,\mathrm{element}}}
\end{aligned}$$

The contour integral is taken counterclockwise about each element.

For supersonic-freestream external flows, the correct drag can only be obtained by adding the individual element surface-force values.

$$C_T \;=\; \sum_\mathrm{elements} C_{T_{p\,\mathrm{element}}} \;+\; C_{T_{f\,\mathrm{element}}}$$

The wake-integral form $C_T = C_{T_\mathrm{i}} + C_{T_\mathrm{v}}$ defined earlier is incorrect and must not be used, since the momentum defect integral for C_{T_i} does not account for the momentum in the outgoing waves. These waves are dissipated only at extreme distances from the body, and their momentum flux cannot be deduced from the near-field entropy distribution.

6.4 Graphics

The plot library used by all the MTFLOW programs is Xplot11, (`libPlt.a`), which is aimed at driving X-terminals. A PostScript file can be generated at any time from the plot visible on the screen. The file `plotlib/Doc` contains much more information on this graphics package.

6.5 General Hints

6.5.1 Viscous solutions

The initial streamline pattern generated by **MTSET** corresponds to incompressible inviscid flow, which may be quite different from the actual flow if significant swirl, heat release, or losses are being imposed. The initial boundary layer guess for the Newton solver may therefore be very far off, possibly giving convergence difficulties. It may be advantageous to first perform several inviscid iterations (by leaving $Re_\mathrm{inl} = 0$) before the viscous solution is attempted.

6.5.2 Inverse solutions

Care must be used when running the Inverse mode with a viscous case. It is essential that there is no separation or near-separation anywhere within the inverse segment. Since the surface pressure imposed in Inverse mode is also imposed on the boundary layer, a physically ill-posed problem results if the boundary layer is separated. In practice, wild changes in the geometry will result under the separated region. Usually, the airfoil shape will fold up and the calculation will collapse. A fairly simple fix to this problem is to temporarily “freeze” the boundary layers by specifying $Re_\mathrm{inl} = 0$ when the inverse case is being converged. The case then be “unfrozen” and the case reconverged in the usual analysis mode. The resulting viscous C_p will change slightly from the specified $C_{p\,\mathrm{spec}}$ in the inverse calculation, but this is usually minor, and can be iterated if desired.

6.5.3 Grid resolution

Compared to most Euler solvers, **MTSOL** is usually quite insensitive to grid density. It behaves more like a potential solver in this regard, especially if `Smom`=3 or `Smom`=4 are used. In any case, it is a good idea to check that the leading edges are reasonably well resolved, And that no spurious losses (or gains!) are being generated there. The streamtube entropy $\tilde{s}$ distributions indicate this quite well. Aside from any prescribed variation due to ΔS, the computed $\tilde{s}$ distribution should be piecewise-constant, with sudden monotonic jumps through the shocks.

Any separation bubble present in the flow must be well-resolved. The default grid is usually adequate for most cases, but maybe not if the bubble is close to the leading edge and very small in streamwise extent. Moderate Reynolds numbers (1-3 million, say) require the finest grid, since the bubbles are then still important, but very small. Fortunately, streamwise grid spacing is "cheap", increasing the solution time only linearly, so it may be simplest to increase the grid point number parameter `N` in **MTSET** to 200 or more.

Supersonic freestream solutions typically require relatively dense grids to capture oblique shocks without excessive smearing. Second-order dissipation is particularly effective in reducing such smearing to a minimum. Supersonic flows with oblique shocks can also cause problems on grids with extreme cell aspect ratios, which typically occur in the outermost streamtubes on tall grids. Hence, for supersonic flows it is often advantageous to bring the outermost streamline relatively close to the body (via the `YTOP` parameter in the `walls.xxx` file). The supersonic-wave boundary conditions will minimize the influence of this nearby boundary on the flow at the body. In contrast, the subsonic source+doublet farfield model is more approximate and requires somewhat more distant outer boundaries.

6.5.4 Execution sequence strategy

While the Newton solution method used by **MTSOL** is very efficient for converging small parameter tweaks, it intensely dislikes large changes. Trying to reconverge a solution after a drastic parameter change, such as going from $M_{\text{inl}} = 0.05$ to $M_{\text{inl}} = 0.7$, is definitely not a good idea. A more gradual parameter change or a complete restart may be more effective. Clearly, runs should be sequenced so that such large changes are avoided.

If a solution contains shocks which are trying to migrate to their final destination, it may be effective to temporarily use first-order dissipation and also increase the dissipation coefficient to $C_\mu = -1.5$ with the `K` command in the parameter-modification submenu. Reducing the threshold Mach number to $M_{\text{crit}} = 0.8$ or so with the `C` command in addition is also effective. The added dissipation smears the shock and allows it to move considerably farther per iteration. Once the smeared shock reaches its destination, the C_μ and M_{crit} values can be restored and the case reconverged to sharpen the shock.

6.6 MTFLO

MTFLO is primarily a "data management" program, used to extract information from the `tdat.xxx` state file, as well as to place new information into it. On startup, **MTFLO** immediately tries to read the `tdat.xxx` file, and proceeds to display one of the default field parameters defined in the earlier section describing the `tflow.xxx` file. If a `tdat.xxx` file is not available, it tries to read the `walls.xxx` and `tflow.xxx` files and attempts to define the flowfield as much as possible. It then produces its top level menu and prompt:

```
  P arameter options (edit,read,write)
  G enerate MTFLOW field parameters from MISES solutions

  R ead  MTFLOW state file
  W rite MTFLOW state file

  I ises  file output for MISES solutions
  S tream file output for MISES solutions

  A nnotate plot
  H ardcopy current plot
  Q uit

 MTFLO  c>
```

The major commands are described in the following sections.

6.6.1 Parameter edit options

Command `P` displays one of the field parameters as a contour plot generated from the bicubic spline representation using the s, t variables, and produces the following field parameter submenu.

```
    # parameter index to view (0 for list)

    R ead  parameter text file
    W rite parameter text file

    E dit  parameter
    C lear parameter
    F unction-type select for parameters
    L ine-plot type toggle

    A nnotate plot
    H ardcopy current plot
    Z oom
    U nzoom

   MTFLOW field parameter option   c>
```

Any one of the field parameters can be selected for display/editing by typing its number from the following list (generated by typing `0`):

```
    1  x
    2  r
    3  T_theta
    4  S_rel
    5  Delta(G)
    6  Delta(S)
    7  Delta(H)

 Enter parameter index  i>
```

In addition to the contour plot, there is also a line plot showing profiles of the current parameter versus s or t (the plot type is toggled with command `L`).

If the field parameter data is not yet present, it can be fetched from a `tflow.xxx` file with the `R` command. Command `E` can then be used to edit the data via the cursor, in a manner similar to that used to edit the $C_{p\,\mathrm{spec}}$ distributions in **MTSOL** . The edited field can be written out to a new `tflow.xxx` file at any time with command `W`.

6.6.2 Field parameter generation options

An alternative to reading the field parameter information is to generate it from a "stack" of blade-to-blade MISES solutions. Command `G` reads and processes the sequence of MISES state files `idat.yyy_01`, `idat.yyy_02` ... `idat.yyy_nn` until no more are available. The flowfield in each state file is appropriately averaged in θ to obtain the $T_\theta(s)$, $\mathcal{S}_{\mathrm{rel}}$, $\Delta G(s)$, $\Delta S(s)$, and $\Delta H(s)$ parameter distributions along that streamsurface. The t value for each such surface is obtained from file `stack.xxx` which simply gives the "`nn`" MISES state file integer suffix, its $\bar{t}$ value which determines its radial location, and its stage index.

```
01   T(1)   Kstage(1)
02   T(2)   Kstage(2)
.
.
nn   T(n)   Kstage(n)
```

The $\mathtt{T} = \bar{t}$ parameter is defined in terms of the streamfunction (fractional mass flow) as before. Together with the θ-averaged distributions along s, the entire parameter distributions in s, t are thus determined.

6.6.3 Field parameter implementation

Once a suitable field parameter distribution is generated, it is written out in `tdat.xxx` with the `W` command at top level. This version of the field parameter distribution inside `tdat.xxx` now becomes the "official" version, and the `tflow.xxx` or `idat.xxx_nn` files are no longer needed.

The new field parameter distributions are incorporated into the MTFLOW solution by simply reconverging the flowfield in **MTSOL** as usual. **MTFLO** can then be used to edit the field parameters, or enter new field parameter distributions, and the process is repeated.

6.6.4 Iteration with MISES

The `I` and `S` top level commands in **MTFLO** produce a series of files which serve as input for the stack of MISES solutions. These solutions would then be re-converged individually, and the MTFLOW/MISES cycle can be repeated.

7 Appendix A — Blade loading model

The local $\bar{\Gamma}_o$ which drives the swirl-evolution equation (13) is defined from the local relative-frame flow tangency to the blade slope $\mathcal{S}_{\rm rel}$.

$$\frac{W_\theta}{W_s} = \frac{\bar{\Gamma}_o/r - \Omega r}{q} = \mathcal{S}_{\rm rel} \tag{24}$$

$$\bar{\Gamma}_o = (q\mathcal{S}_{\rm rel} + \Omega r)\, r \tag{25}$$

In Figure 2 the local $\bar{\Gamma}_o$ is shown constant, although in general it may vary for non-constant q, r, or $\mathcal{S}_{\rm rel}$ over the chord. The rate constant k in (13) is obtained by requiring the correct blade lift in the low-solidity limit, as in a lightly-loaded propeller. Referring to the rightmost sketch in Figure 2, the swirl change over the blade row in this case is

$$\Delta\bar{\Gamma}_{\rm 2D} \simeq \left(\frac{d\bar{\Gamma}}{ds}\right)_{\rm avg} c_s = k\, \left(\bar{\Gamma}_o - \bar{\Gamma}\right)_{\rm avg} r\, c_s \tag{26}$$

where c_s is the blade chord c projected along the s direction.

$$c_s = \frac{c}{\sqrt{1+\mathcal{S}_{\rm rel}^2}} \tag{27}$$

Substituting for $\bar{\Gamma}_o$ from (25) we have

$$\Delta\bar{\Gamma}_{\rm 2D} = k\, \left(q\mathcal{S}_{\rm rel} + \Omega r - \bar{\Gamma}/r\right)_{\rm avg} r\, c_s \tag{28}$$

In the low-solidity limit, this $\Delta\bar{\Gamma}_{\rm 2D}$ must also correspond to the total circulation of the B blades acting as 2D airfoils at the local flow incidence α, with some 2D-section lift-curve slope C_{ℓ_α}.

$$\Delta\bar{\Gamma}_{\rm 2D} = -\frac{B}{2\pi}\left(\frac{1}{2}\, c\, W\, C_{\ell_\alpha}\, \alpha\right) \tag{29}$$

$$W^2 = q^2 + (\Omega r - \bar{\Gamma}/r)^2 \tag{30}$$

The negative sign is needed to give consistency with the sign conventions shown in Figure 2. The local blade-relative speed W includes the swirl velocity $\bar{\Gamma}/r$ to allow for any upstream swirl generation. From the velocity triangle geometry, the local incidence angle α is

$$\alpha = -\frac{q\mathcal{S}_{\rm rel} + \Omega r - \bar{\Gamma}/r}{W}\,\frac{q}{W} \tag{31}$$

so equating the two $\Delta\bar{\Gamma}_{\rm 2D}$ expressions (28) and (29) above and simplifying the results we have

$$k\, \left(q\mathcal{S}_{\rm rel} + \Omega r - \bar{\Gamma}/r\right)_{\rm avg} r\, c_s = \frac{B}{2\pi}\frac{1}{2}\, c\, W\, C_{\ell_\alpha}\, \frac{q\mathcal{S}_{\rm rel} + \Omega r - \bar{\Gamma}/r}{W}\,\frac{q}{W} \tag{32}$$

$$k = \frac{B}{4\pi r}\, C_{\ell_\alpha}\, \frac{q\sqrt{1+\mathcal{S}_{\rm rel}^2}}{\sqrt{q^2 + (\Omega r - \bar{\Gamma}/r)^2}} \tag{33}$$

Although this expression for k could be used in the rate equation (13), the level of approximations in this flow model warrants some simplifications.

If we assume modest incidence and swirl angles, the last factor in (33) is nearly unity.

$$\frac{q\sqrt{1+\mathcal{S}_{\mathrm{rel}}^2}}{\sqrt{q^2 \,+\, (\Omega r - \bar{\Gamma}/r)^2}} \;\simeq\; 1$$

Large incidence will invalidate the assumed linear C_ℓ–α relation anyway, so this approximation is reasonable. Furthermore, it's reasonable to simple assume the incompressible low-solidity value $C_{\ell_\alpha} = 2\pi$. This could be made a function of blade-relative Mach number, although this is deemed to be an unnecessary complication. The rate constant then reduces to definition (15).

$$k \;=\; \frac{B}{2r} \tag{34}$$

It's useful to estimate the swirl change predicted by equation (13). We first more clearly write k in terms of the blade pitch $P = 2\pi r/B$,

$$k \;=\; \frac{\pi}{P}$$

and then for constant k and $\bar{\Gamma}_o$, and zero initial swirl, equation (13) can be integrated to give the following circulation, and swirl mismatch at the blade exit (or deviation) at $s = c_s$.

$$\bar{\Gamma}(s) \;=\; \bar{\Gamma}_o\left[1 - \exp\left(-\pi\frac{s}{P}\right)\right] \tag{35}$$

$$1 - \frac{\bar{\Gamma}}{\bar{\Gamma}_o} \;=\; \exp\left(-\pi\frac{c_s}{P}\right) \tag{36}$$

This gives the realistic numbers as shown in the table.

Table 1: Predicted exit swirl mismatch versus solidity

c_s/P	$1 - \bar{\Gamma}/\bar{\Gamma}_o$
0.1	0.730
0.2	0.533
0.5	0.210
1.0	0.043
1.5	0.009

8 Appendix B — Blade shape generation

If the swirl forcing field ΔG is being prescribed rather than the blade slope $\mathcal{S}_{\rm rel}$, then the blade flow tangency equation (13) is not used in the flow solution. Instead, it can be treated as a definition of $\mathcal{S}_{\rm rel}$, which is also the blade camberline slope in the m'–θ plane. By recasting equation (13) together with the definitions (14) and (15), we get

$$\mathcal{S}_{\rm rel} \;=\; \frac{d\theta}{dm'} \;=\; r\,\frac{d\theta}{ds} \;=\; \frac{1}{q}\left(\frac{2}{B}\frac{d\bar{\Gamma}}{ds} \;+\; \frac{\bar{\Gamma}}{r} \;-\; \Omega r\right) \tag{37}$$

Using s as the convenient independent parameter, the blade camberline coordinates can be integrated as follows.

$$m'_c(s) \;=\; \int^s \frac{1}{r}\,ds \tag{38}$$

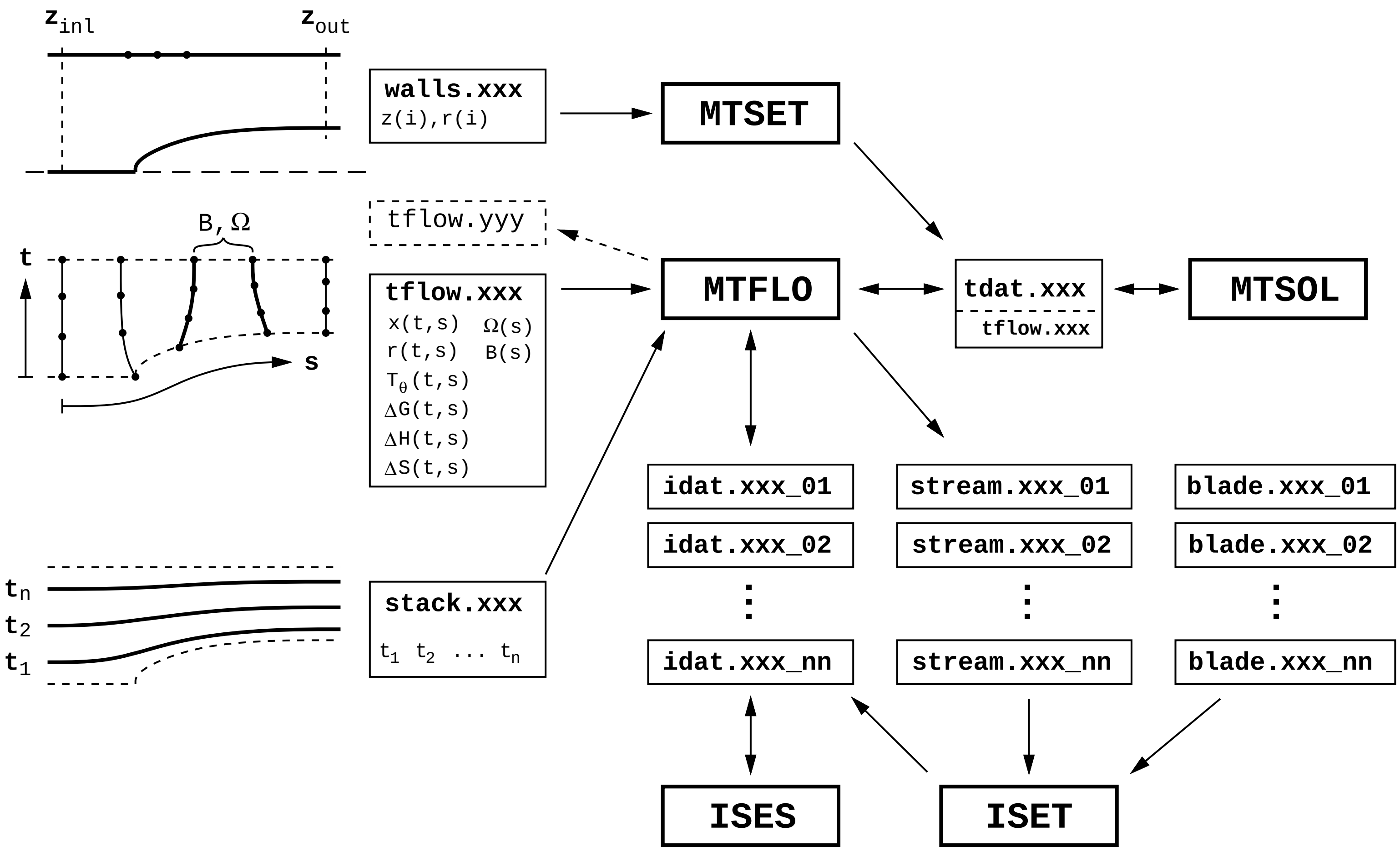
$$\theta_c(s) \;=\; \int^s \mathcal{S}_{\rm rel}\,\frac{1}{r}\,ds \tag{39}$$

A blade-perpendicular thickness distribution $T(s)$ can then be added to generate the upper and lower blade surface coordinates.

$$m'(s) \;=\; m'_c \mp \frac{\mathcal{S}_{\rm rel}}{\sqrt{1+\mathcal{S}_{\rm rel}^2}}\,\frac{T}{2r} \tag{40}$$

$$\theta(s) \;=\; \theta_c \pm \frac{1}{\sqrt{1+\mathcal{S}_{\rm rel}^2}}\,\frac{T}{2r} \tag{41}$$

MTFLOW/MISES Data Flow



z_{inl}
z_{out}
walls.xxx
z(i),r(i)
MTSET
B,Ω
t
s
tflow.yyy
tflow.xxx
x(t,s)
Ω(s)
r(t,s)
B(s)
T_θ(t,s)
ΔG(t,s)
ΔH(t,s)
ΔS(t,s)
MTFLO
tdat.xxx
tflow.xxx
MTSOL
t_n
t_2
t_1
stack.xxx
t_1 t_2 ... t_n
idat.xxx_01
idat.xxx_02
idat.xxx_nn
stream.xxx_01
stream.xxx_02
stream.xxx_nn
blade.xxx_01
blade.xxx_02
blade.xxx_nn
ISES
ISET