Numerical parameters

Introduction

The default values of the parameters of *Code_Saturne* that appear in the Graphical User Interface (GUI) or in the user manual are the best compromise between robustness and precision, on the basis of the experience collected up to now with the code.

Usual numerical parameters

Gradients calculation: the default method (“finite volume gradient” with an iterative treatment of the non-orthogonality terms, IMRGRA=0) is usually satisfying enough, but it is sometimes useful to adopt a solution that often turns out to be better for distorted meshes (least square method with an extended stencil in risky regions, IMRGRA=3). This solution is also less CPU time demanding, but it is not the default option, because the experience that has been collected is not yet considered to be entirely sufficient (moreover when this method is applied to the pressure gradient, it does not allow to guarantee the conservativity for the momentum equation). For non-structured tetrahedral grids IMRGRA=2 (least square method with a full extended stencil throughout the domain) is the most robust approach.

It is not desirable to increase the threshold value of the precision parameter EPSRGR (by default 10^{-5}) that is used when an iterative method is selected to compute the gradients. Indeed, the risk of accumulating numerical errors is not compensated by the potential reduction of the CPU time.

Convective scheme: the first-order convective scheme (upwind) is usually more stable and less accurate than the second-order schemes (centred, SOLU).

- Indeed, the upwind scheme introduces an artificial numerical diffusion that stabilizes the solution but increases the error. The corresponding physical effect is similar to that of a diffusion term (second-order derivative in space) proportional to the local velocity and cell size.
- Moreover, the upwind scheme has the advantage to guarantee the “maximum principle”: without source term, the values of a scalar (concentration, temperature) advected by this scheme remain in the admissible interval, defined by its minimal and maximal initial values – keeping in mind that this property may be lost because of the discrete treatment of the diffusion term on arbitrary meshes.

These pure first- and second-order schemes may be combined:

- To increase the stability of the second-order schemes, a “slope test” is activated by default: it switches locally and without any explicit warning to the upwind scheme wherever oscillations of the solution are detected. This may reduce the order in space. This slope test may be switched off (parameter ISSTPC, 0 by default).
- It is possible to select a blended scheme that uses an interpolation between the values that the pure first and second-order schemes would have produced (formally, BLENCV*second-order+(1-BLENCV)*first-order, with BLENCV=1.0 by default).

For accurate RANS computations[1], the following advice may be followed, without claiming to universality:

- For the velocity: a second-order scheme with the default slope test (no maximum principle is required and it is desirable to give the priority to accuracy)
- For the turbulence: a first-order scheme, usually adequate enough
- For the scalars (concentration, temperature): a second-order scheme with the default slope test (this choice shall minimize numerical diffusion but do not guarantee the “maximum

[1] For LES, it is advised to keep the specific default values, but the same principles apply.

principle": the solution should be accurate, with little diffusion, but it may exhibit values outside its physical limits)

If the advected variables exhibit oscillations (in regions with high gradient values, stratification, distorted mesh…), it may be useful to use one of the following solutions, listed from the more accurate to the more robust:

- Use the SOLU convection scheme: it is still second-order, but sometimes more stable than the centred scheme.
- Use a blended convection scheme with a user-defined percentage of upwind (to begin with, one may prescribe 20% upwind, i.e. BLENCV=0.8).
- Use an upwind scheme for all the advected variables (the result obtained this way will have to be considered as a first result, still to be improved).

Flux reconstruction: the finite volume method relies on the calculation of convective and diffusive fluxes at the cell faces. These quantities are calculated from values of the variables evaluated at the orthogonal projections of the cell centres on the straight line normal to the face and containing its centre. By default, the method takes into account the fact that the mesh may not be orthogonal: the "flux reconstruction" is activated (IRCFLU=1). If this option is not used, the simulation is usually more stable but the discretization is not consistent any more (the solution converges as the mesh is refined, but it "sticks" to the mesh and is potentially wrong: for a diffusion problem, for example, the isolines are parallel to the mesh lines, independently of the physical phenomenon). Hence, it is generally compulsory to retain the flux reconstruction option. However, for RANS calculations and especially for the k-epsilon, it has been observed that switching off the flux reconstruction (IRCFLU=0) for k and epsilon could contribute to stabilizing the calculations when difficulties were encountered on low quality meshes, and this apparently without deteriorating the quality of the solution for the mean quantities. *A priori*, this also applies to the k-omega and all the other RANS variants

Pressure relaxation: for time marching schemes[2], it is possible to under-relax the pressure increments calculated within the pressure correction step (by default, no relaxation is used: RELAXV(IPR)=1.0). This is useful for distorted meshes and, *a priori*, it does not affect the quality of the solution. From previous experience, 30% is a reasonable value for the under-relaxation parameter (i.e. RELAXV(IPR)=0.7). For example, this option may be activated if the calculation fails after several time steps for which the number of iterations of the pressure iterative solver has been unusually large or if one suspects that the quality of the mesh is low in a region where large velocity values develop over a few time steps.

Convergence threshold for the iterative solvers: it is not desirable to increase *a priori* and carelessly the threshold EPSILO (by default 10^{-8}, or 10^{-5} when the number of iterations for the right-hand-side reconstruction is large enough). However, 10^{-8} is a small value and it is possible to consider using EPSILO=10^{-5} to save CPU time. For example, this choice may be made to produce a first approximation of the solution. One may also consider this choice if a large number of similar computations must be carried out: the first computation may be run with EPSILO=10^{-5} and with EPSILO=10^{-8}; if the results are close enough, the rest of the simulations may then be carried out with EPSILO=10^{-5}.

Pressure interpolation: the pressure gradient is required at the cell centre for the momentum equation. Hence, because finite volume techniques are used, pressure face-values are necessary. Those are usually interpolated (centred interpolation) from values at the neighbouring cell-centres. This standard approach is appropriate as long as the pressure gradient is reasonably continuous. When the local variations of the pressure gradient are large, the approach is not valid any more (a simple centred interpolation cannot account for the fact that the pressure gradient is significantly

[2] "Time marching scheme" is used here as opposed to "steady-state algorithm".

different on both sides of the face under consideration). If this standard approach is used carelessly, the balance of the discrete terms in the momentum equation is not satisfied anymore and spurious velocities appear (in particular, this phenomenon may be encountered in the vicinity of a stratification or close to the borders of a head loss region).

Activating the option IPHYDR (see the user manual) solves the problem through the calculation of an interface pressure taking into account the local variation of the pressure gradient. Moreover, if the user has defined a head loss region adjacent to the domain outlet (i.e. in the cells that have an outlet boundary face), the option ICALHY=0 will also have to be activated to avoid noticeable perturbations of the pressure boundary condition.

Multigrid solver: to compute the pressure, it is necessary to solve a Poisson equation for which the matrix is sometimes poorly conditioned and that may require the largest portion of the total CPU time. It is advised to use the multigrid algorithm (IMGR(IPR)=1). This option is the default for all the Poisson equations with a scalar unknown (pressure, mesh velocity for ALE, hydrostatic pressure…). It may reduce the total CPU time by a factor of 2 to 4 and has no influence on the quality of the solution.

Time-marching scheme

Time-marching algorithm and time step value

The “historic algorithm” of *Code_Saturne* is a time-marching algorithm, where a time step Dt is specified from the flow characteristics (velocity U and kinematic viscosity ν or conductivity λ, or their turbulent equivalents) and from the size of the cells Dx.

- **To begin with,** an upper limit Dtmax for the time step Dt shall be evaluated as the maximal time-scale value. Dtmax is at least as large as:
    - The time that is required for a fluid particle to pass through the whole domain (or the time necessary to cross the domain for a signal transmitted by convection or diffusion if these phenomena are dominant)
    - The characteristic time associated with potential gravity effects $(H/(g\Delta\rho/\rho))^{1/2}$ with H a characteristic height, $\Delta\rho$ the characteristic variation of the density ρ and g the gravity
- **Ideally**, the time step value Dt (DTREF) is then set on the basis of target Courant (U Dt/Dx) and Fourier (ν Dt/Dx^2) numbers. One can use target values of 1 and 10 respectively, in coherence with the default parameters of the code (COUMAX and FOUMAX). Quite small default values have been selected voluntarily.
- **In practice,** for most of the cases, the value of the time step will be chosen so that the Courant number be around 1 to 5 in the main part of the domain of interest. Quite large values are acceptable and will not deteriorate the stability of the computation (up to 10 and even 50, as long as they are reached only locally). Small values will not endanger the stability of the computation either, but the Courant number should not be too low in the important regions (for example, not lower than 0.01). Finally, the user should check *a posteriori* that the Fourier number computed by the code remains lower than values ranging from 10 to 1000.

Transient flows

- The time-marching algorithm with a time step constant in time and uniform in space is the standard choice in *Code_Saturne* (IDTVAR=0).
- If the time step value is limited because of a specific part of the transient (for example because of the occurrence of a large velocity during a short period of the transient), it is possible to let the time step value change in time (IDTVAR=1). COUMAX and FOUMAX will be modified to set the desired targets.

Steady flows

- The time-marching algorithm with a time step uniform in space (IDTVAR=0 or 1) makes it possible to compute a steady solution as the limit of a transient one that has been started from an estimated initial state. The convergence may be slow, in particular if the ratio UDt/Dx varies significantly over the computational domain; in that case, the algorithm with a time step which is variable in space and in time may be used (IDTVAR=2). COUMAX and FOUMAX will be modified to set the desired targets.

- The steady-state algorithm of *Code_Saturne* (IDTVAR=-1) is not so widely used so it benefits from less feedback, but its properties are similar to that of IDTVAR=2, with perhaps a little less stability. This algorithm does not require any time step value, but rather a relaxation parameter for each variable.

Warnings

- **Convergence**
    - o With the time-marching algorithm, the calculation must be carried out over approximately 5 to 10 times the time required for a fluid particle to cross the whole domain (of course, if thermal conduction prevails, the thermal time-scale must also be considered).
    - o With the steady-state algorithm (IDTVAR=-1), convergence is generally reached within a few thousands iterations for “standard” cases; if this algorithm fails, one may select the time-marching algorithm instead with a time step variable in space and in time (IDTVAR=2).
    - o One may also add to the computation the advection of a fictitious passive scalar, initially set to 0 over the whole domain and set to 1 at the inlet. The moment when this scalar reaches 1 in the whole domain can be used as another convergence indicator (not always required, not always sufficient, but useful all the same).
    - o In all cases, the variation of the quantities provided by the code will be examined (i.e. the “derive” quantity of the log file (“listing”) associated with all the variables but the pressure[3]). One can also use “monitoring points” to follow the evolution of the case-dependent important quantities at user-selected locations and at each time step (so that no oscillation may go unnoticed).
- **Calculations with heat transfer / scalars**
    - o It is not advised to use the steady-state algorithm when the flow is influenced by heat transfer (through the density variation) or by the distribution of advected scalars representing a concentration.
    - o On the other hand, if the flow is not seriously affected by the distribution of advected scalars, the dynamic features and the scalar transport may be decoupled and it may be possible to:
        - ▪ Multiply the time step by a factor so as to accelerate the scalar/temperature convergence if necessary.
        - ▪ Compute the dynamic variables first and deal with the temperature/scalars in a second stage on a frozen velocity/pressure field.
- **Unsteady RANS**
    - o The RANS turbulence models are based on statistical decompositions that incorporate the turbulent fluctuations into specific variables (k, epsilon, omega, Rij…). The use of RANS models for unsteady flows (U-RANS) raises a problem: is there a cut-off frequency (and what is its value) that would separate low-frequency fluctuations (responsible for variations in time of the mean quantities) from high-

[3] For a steady flow, the “derive” is the square of a L^2-norm residual. More precisely, with the subscript i standing for the cell number, the superscript n referring to the time step, with Dt_i the time step value and Vol_i the cell volume, both associated with the cell i, and with Vol_{tot} the total volume of the computational domain, for a variable V other than the pressure, the “derive” computed by *Code_Saturne* is $D(V) = \Sigma_i \left((V_i^n - V_i^{n+1})^2 \, Vol_i/Dt_i \right) / Vol_{tot}$.

frequency fluctuations (incorporated into the turbulent variables but potentially unnoticeable on the variations in time of the computed quantities)?

  - o This is an **open question.** Today, however, unsteady RANS are used without any particular precaution. LES are carried out when it is practicable (i.e. when the Reynolds number is low enough so that the available CPU resources make it possible to use a mesh sufficiently fine to resolve enough turbulent structures).

- **In case of failure of the steady-state algorithm (IDTVAR = -1)**
  - o Shall the steady-state algorithm fail, the following set of parameters for the relaxation coefficients may be used: 0.3 for pressure and 0.7 for all the other variables (the default choice, which is not a fully standard one, is 0.1 for pressure and 0.9 for all the other variables).

Boundary conditions

Although the standard boundary conditions generally cover the usual needs, some complements are provided hereafter.

Outlet boundary: for an incompressible flow, the outlet is a theoretical problem *per se*, since it is theoretically necessary to know (some characteristics of) the flow downstream of the outlet to be able to implement the boundary conditions. The standard method (ISOLIB) relies on the hypothesis that the pressure profile does not change in the direction normal to the outlet. To use this approach in the best possible conditions, it is advised to:

- Select, as much as possible, plane outlets orthogonal to the mean flow so that the hypothesis of invariance of the pressure profile in the direction normal to the outlet is as realistic as possible.
- Place the outlets sufficiently far away from the regions of interest so as to minimize the influence of the boundary conditions on the results of the calculation.
- Place the outlets sufficiently far away from the geometrical perturbations that may produce vortices. They may grow arbitrarily when crossing the outlet (at the outlet, if a vortex is strong enough, half of it may pump momentum into the domain, while the other half pumps momentum out of the domain: even if the flow remains divergence-free, the vortex may behave unphysically, sticking to the outlet and growing arbitrarily).
- If it is necessary, displace the outlets downstream: the mesh may be extended artificially with a few layers of hexahedra for example.
- Set an external Dirichlet value for all the advected scalars (temperature, concentrations): it will be used in case the flow re-enters at the outlet (otherwise, a zero-flux condition would be used, and make the calculation less stable).
- For large outlets in particular, it may be necessary to set the pressure profile using Dirichlet conditions to ensure that the calculation remains stable (for example, a uniform pressure may be used: this choice is the responsibility of the user, and depends on how well the pressure is known at the outlet).
- For multiple outlets, it is advised to set the pressure at all the outlets but one. This approach has the advantage of stabilizing of the calculation, but above all, it is physically sound: indeed, it is necessary to provide data characterizing the head loss of the circuits that are located downstream of the outlets so as to allow for a physical distribution of the flow upstream.

Wall boundary conditions: if there is no roughness to specify, the standard wall boundary conditions (IPAROI) do not require any specific attention, except for stratified flow (with variable density and gravity forces) close to a non-vertical wall. In that case, it is advisable to extrapolate the pressure boundary condition (EXTRAG(IPR)=1), otherwise, the balance of the discrete terms in the momentum equation may not be satisfied anymore, and spurious velocities may appear close to the wall. This option has no effect if IPHYDR is activated (=1).

Inlet conditions: the standard inlet conditions (IENTRE) do not usually require any specific attention. However, the following advice should be followed:

- Select, if possible, plane inlets orthogonal to the mean flow.
- Place the inlets sufficiently far away from the regions of interest so as to minimize the influence of the boundary conditions on the results. It is seldom possible to place the inlet sufficiently far away for a fully developed flow to establish (the length that is necessary for turbulent a pipe flow to be fully developed is approximately 100 times the hydraulic diameter), but it is fortunately seldom useful since the real conditions that should be reproduced in the calculation are generally not that of a fully developed flow. However, it is suggested to place the inlet approximately at least 10 hydraulic diameters upstream of the regions of interest so as to allow for some coherence to develop between the variables that are advected from the inlet.
- If it is necessary, displace the inlets upstream: the mesh may be extended artificially with a few layers of hexahedra for example.
- For the turbulent variables, provide inlet values that are coherent between each other (the default values automatically set by the code are coherent). For example, it would be a particularly bad choice to provide a very accurate profile for the turbulent kinetic energy k, and a value a hundred times too large for the turbulent dissipation epsilon (in such a case, the data for k would immediately be destroyed by the oversized dissipation).
- For LES, it is advised to use the default vortex method or the SEM (Synthetic Eddy Method). In practice, for that kind of computation, it is advisable to contact the development team.

Setting-up and checking the computation

It is a good idea to check through the topics in this section to avoid common errors that may lead to erroneous interpretations of the results (the calculation may not fail but instead produce erroneous results or the user may misinterpret the results).

It is advisable to carry out the verifications suggested here as early as possible (for example as part of a preliminary computation over the first few steps).

Data input

- Ensure that the mesh has the right units (most often in metres): otherwise, the calculation may fail or may seem to progress very slowly – towards wrong results – (for example if a mass flow rate designed for an inlet of 1 m^2 is imposed on an inlet of 1 mm^2 or of 1 km^2). If a rescaling type modification is required, one should keep in mind that *Code_Saturne* can resize the mesh (multiply all dimensions by 1000 for example).
- For inlet conditions, it should be checked right from the start of the calculation that the velocity is imposed in the right direction (an error due to the orientation of the axes should not be ruled out) and more generally that the variables have the right values and units. This may be done through a preliminary post-processing stage, immediately at the beginning of the calculation. It should be kept in mind that standard post-processing operates on cell-centre variables; as a consequence, the exact values set as inlet conditions at the boundary faces cannot be observed (however, the difference after a few time steps will usually not exceed 1%).
- If outlets are supposed to let the fluid *out* (and not *in*), it is important to ensure right from the start of the calculation that the flow goes in the right direction: should it go the wrong way, a flaw may be suspected (for example: a faulty joined mesh, an incorrect data input, a time step value too small …)
- When a heat flux is prescribed at the walls, one should double check that the sign of the flux is the right one.
- If variable physical properties have been prescribed (density, viscosity…), the user should plot

them to check their order of magnitude, their upper and lower limits and the sign of their rate of change. Amongst the most common errors, one may encounter the following:

- The use of a polynomial law outside of its domain of validity: (the interval where it represents the physical property considered) may lead to large errors, in particular with high degree polynomial laws[4]. If P(T) represents the density as a function of the temperature in the interval [a ; b], it is desirable to compute the numerical values of the density as P(T') with T' = min(max(T,a),b). Ideally, this modification should be a mere safety check. Indeed, if T' is not exactly equal to T, then the user faces a problem. On the one hand, if there is no physical reason why the temperature should remain in the interval [a ;b], then the choice of the law P(T) should be questioned. On the other hand, if T should physically remain in the interval, then it is the numerical scheme that is producing non-physical values; in that case, it is compulsory to use T' to compute the density, but this modification does not address the substance of the problem (which may be associated with the choice of the convective scheme or with the quality of the mesh).
- The coding errors in the user-defined programmes are a common cause of difficulty. To avoid these, before starting the computation, one may extract the piece of user source code (for example the lines computing rho(T)), include it in a separate *ad hoc* programme, let the input data vary artificially (here, the variable T would be varied by arbitrary steps between its lower and upper limits) and observe the output (here, print rho(T) and plot it).

Convergence

- Convergence is usually assessed from the evolution of the computational variables at some well-chosen locations ("monitoring points"). For each monitoring point, the code selects the cell whose centre is closest to the point defined by the user from its coordinates. Hence, the following advice should be followed:
    - Check the position of the cells that have been selected by the code. This can be done using the data printed out in the history files (it must not be done using the input data, since the latter does not account for the fact that the centres of the selected cells are close but not necessarily identical to the user-defined locations). A visual verification can be done by positioning each point with Ensight/Paraview on the geometry or by plotting with a 2D-plotter the couples (x,y), (y,z) and (x,z) for all the selected cells. For example, this preliminary check is particularly important if the quantities of interest of the computation is precisely the transient behaviour of some variables at some monitoring points.
    - Record all the variables at all the time steps so that no oscillation may go unnoticed.
- Check the log file ("listing") for the variations of the Courant number: it should remain stable (see the values provided previously). It is also advisable to visualize the Courant number and its 3D distribution as much as possible.
- Check the log file ("listing") for the information related to the mass conservation (and to the energy conservation, if it is available): the mass conservation should be precisely satisfied, with a minimal accuracy of approximately 10^{-6}. An inaccurate mass conservation generally indicates the imminent failure of the calculation (except for some specific capabilities: especially rotor-stator, and periodic boundary conditions – the problem has been solved for the periodic boundary conditions in *Code_Saturne* 2.1).
- Check the log file ("listing") for the information related to the energy balance (it is advisable to code the case-dependent energy balance as a user-defined programme using the example provided in the user routine usproj). When the density is constant, the energy conservation should be satisfied with a minimal accuracy of approximately 10^{-6}. When thermal coupling

[4] One should do one's best to avoid using high degree polynomials. It is often possible to replace them by a series of polynomials of lower degree.

with SYRTHES is used, the energy balance of *Code_Saturne* will still be valid with the same accuracy, but the global energy balance (SYRTHES+*Code_Saturne*) may not be perfect, the meshes and the numerical methods of both codes being different: it is a good idea to check the results with the support..

- Check the log file ("listing") for the information related to the mean, minimal and maximal values of the variables. It is usually a good way to identify the problems of convergence or even the input errors.
- Check the log file ("listing") for the information related to "clipping": indeed, for the turbulent variables (and for the temperature or the scalars, if they are used), an automatic limitation is implemented. When the calculation is converged, the number of cells where a variable has been "clipped" should be small (less than 0.1% of the number of cells). If more cells have been "clipped" this may indicate a flaw in the mesh or in the modelling or an unsatisfying level of convergence. However, it is possible that the number of cells where a variable has been limited be somehow larger (a few %) if a second-order convective scheme is used (especially the centred scheme, but also SOLU).
- Check the log file ("listing") for the "derives" quantity of the variables: indeed, for a steady-state flow, they should diminish by several orders of magnitude between the beginning of the calculation and the converged state (by approximately 2 orders of magnitude for the pressure and 3 or 4 for the velocity).
- Check the log file ("listing") for the numbers of iterations of the linear solvers: for a steady-state flow, they should diminish quite rapidly and should, for standard computations[5], tend typically towards values of approximately 10 for the velocity and 500 for the pressure (and respectively up to 100 and 5000 on large or distorted meshes). These order of magnitude values are of course not sufficient to conclude to convergence, but the can help identifying quite early a risk of failure of the computation.

[5] For the pressure, it is assumed that the multigrid option is used (default option); if the multigrid is switched off, the number of iterations increases with the number of cells and it may be necessary to raise the maximal value (the default is 10 000 iterations). Moreover, for the pressure, the number of iterations printed out if the total number of iterations cumulated over all the reconstruction sweeps, whereas for all other variables, the number of iterations that is printed out is that of the last sweep.