CHAPTER 8

Time-Accurate Computations

Although most computations being performed today are for steady-state cases, it appears as though CFD will be used more and more for unsteady, time-accurate cases in the future. Therefore, since this realm is still somewhat in its infancy, this chapter has been written in order to explore in detail what is currently known about the subject, relative to CFL3D's capabilities.

CFL3D has been used extensively for time-accurate computations (**dt** > 0). See, for example, reference 33. Two types of sub-iterations, called "t-TS" and "τ-TS" are currently implemented in the code and are also described in this reference. This chapter describes the effects of the different types of sub-iterations, as well as the strategy for pursuing time-accurate computations in general.

When performing steady-state computations, the primary numerical accuracy issue about which the user needs to be concerned is that of spatial accuracy. Generally, the user runs a problem on a series of successively finer grids. As long as the solutions are fully converged on each grid, the user can get a clear picture of the accuracy of a solution on a given grid. In fact, the user can determine the numerical global order of accuracy by using a series of at least three grids *from the same family* and plotting some global quantity of interest as a function of a measure of the average grid spacing, such as $1/(\sqrt{\text{number of grid points}})$ for 2-d or $1/(\sqrt[3]{\text{number of grid points}})$ for 3-d, on a log-log plot. The slope of the plotted line represents the spatial order of accuracy of the scheme. When the standard $\kappa = -1/3$ scheme is employed, CFL3D has been demonstrated in the past to be globally approximately second-order accurate for most grids. However, this is problem-dependent and the accuracy can degrade somewhat on grids that are too coarse or on grids with extremely severe stretching.

For time-accurate problems, temporal accuracy becomes an additional numerical accuracy issue of concern for the user. Now, not only does the effect of the grid need to be assessed for each problem, but the effect of the time step as well. Additionally, because CFL3D is an implicit code and employs approximate factorization, linearization and factorization errors are introduced during each time step, which can degrade the accuracy of the time-accurate simulation. (In fact, if no sub-iterations are employed, the best that can possibly be hoped for is first-order temporal accuracy.) This is why sub-iterations are generally recommended for time-accurate computations. Sub-iterations "iterate away" the linearization and factorization errors. The more sub-iterations performed, the more accurate the simulation. But how many sub-iterations are enough? And which type of sub-iteration scheme works the best? Hopefully, this chapter will help to answer these questions.

8.1 General Effects of Numerical Parameters

A graphic representing the general effects of sub-iterations, time step, and grid is shown in Figure 8-1. Here, some "quantity of interest" is shown as a function of time step. For example, the quantity could represent Strouhal number for an unsteady circular-cylinder case. Say that the user ran a series of computations on a given grid, using three sub-iterations. Each successive computation used a smaller and smaller time step and each was completely converged to periodic quasi-steady-state. If the user plotted a global quantity of interest as a function of time step, a curve like the lower-most curve in the figure might be obtained. (Note that the figure shows the quantity of interest increasing with decreasing time step, but the trend could also be in the opposite direction, depending on the case and the quantity of interest chosen.)

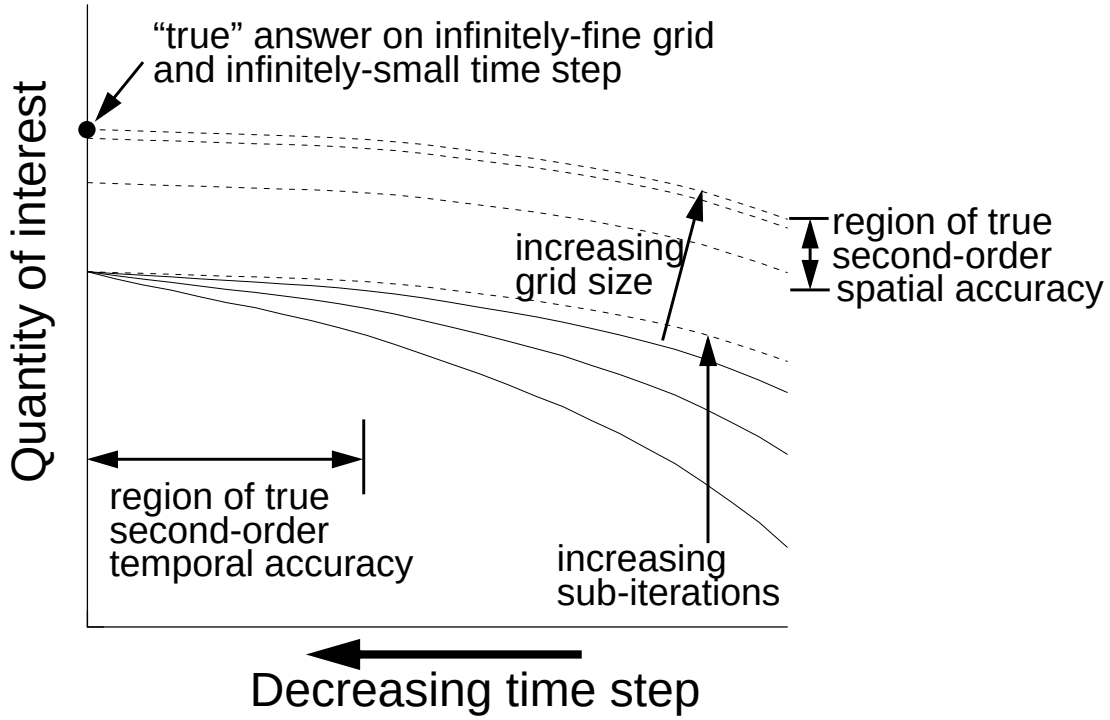

Figure 8-1. Time accuracy trends.

Now say that the user repeated the entire series of computations, except this time using six sub-iterations instead of three. A curve similar to the second curve from the bottom of the figure might be obtained. Finally, with an infinite number of sub-iterations per time step, the user would obtain the dashed curve, which represents the best possible solution *on that grid*. Note that, using a given time step, an increasing number of sub-iterations yields an increasingly better answer, but even the best answer (with infinite sub-iterations) is still in error from the answer using an infinitely-small time step.

The dashed line should behave either first or second order accurate in time (depending on the accuracy input by the user in the input file), but only for sufficiently small time

steps. This is what is graphically represented by the left horizontal arrow in the figure. If the time step is too large, the solution may not exhibit the expected temporal accuracy.

It is evident from this figure that, if an extremely small time step is taken, then sub-iterations are not as beneficial as when larger time steps are taken. The solution is already pretty good. Hence the user needs to make a trade-off between accuracy and efficiency. An extremely small time step can be taken, but at a greater cost, or a larger time step can be taken with some degradation in accuracy. And, sub-iterations may or may not contribute much toward improving accuracy, depending on the time step. Also, at one time step a certain number of sub-iterations may be enough, but at a different time step that same number may be either insufficient or overkill.

The effect of the grid size is also represented in Figure 8-1. On finer and finer grids, the location of the dashed curve will change, approaching the “true” answer on an infinitely-fine grid. (Note that the figure shows the quantity of interest increasing with finer grids, but it could also go the opposite direction, depending on the case and the quantity of interest chosen.) The standard $\kappa = -1/3$ CFL3D scheme should generally behave spatially second-order accurate on sufficiently fine grids. The filled-in circle in the figure represents the “true” answer on an infinitely fine grid with an infinitely small time step. Obviously, the user would generally like to get as close to this answer as possible, but with a reasonable expenditure of resources.

8.2 Effect of Sub-iterations With Time Step and Grid Size

The effect of sub-iterations with time step and grid size is explored in this section for a sample test case of laminar flow over a circular cylinder at Reynolds number 1200 and $M_\infty = 0.2$. The input file, for a relatively fine O-grid with t-TS multigrid sub-iterations, is given here:

```
cylnew.bin
plot3dg.bin
plot3dq.bin
cfl3d.out
cfl3d.res
cfl3d.turres
cfl3d.blomax
cfl3d.out15
cfl3d.prout
cfl3d.out20
ovrlp.bin
patch.bin
restart.bin
    circ cylinder
     XMACH     ALPHA      BETA  REUE,MIL   TINF,DR     IALPH     IHIST
    0.2000    00.000       0.0    0.0012     460.0         0         0
      SREF      CREF      BREF       XMC       YMC       ZMC
   1.00000   1.00000    1.0000   0.00000      0.00      0.00
        DT     IREST   IFLAGTS      FMAX     IUNST    CFLTAU
   +0.1000         0       000   05.0000         0        5.
     NGRID   NPLOT3D    NPRINT    NWREST      ICHK       I2D    NTSTEP       ITA
         1         1         1      6100         0         1      0100        +2
       NCG       IEM  IADVANCE    IFORCE  IVISC(I)  IVISC(J)  IVISC(K)
```

```
         2         0         0         1         0         1         1
      IDIM      JDIM      KDIM
         2       193        97
    ILAMLO    ILAMHI    JLAMLO    JLAMHI    KLAMLO    KLAMHI
         0         0         0         0         0         0
     INEWG    IGRIDC        IS        JS        KS        IE        JE        KE
         0         0         0         0         0         0         0         0
  IDIAG(I)  IDIAG(J)  IDIAG(K)  IFLIM(I)  IFLIM(J)  IFLIM(K)
         1         1         1         0         0         0
   IFDS(I)   IFDS(J)   IFDS(K)  RKAP0(I)  RKAP0(J)  RKAP0(K)
         1         1         1    0.3333    0.3333    0.3333
      GRID     NBCI0   NBCIDIM     NBCJ0   NBCJDIM     NBCK0   NBCKDIM    IOVRLP
         1         1         1         1         1         1         1         0
I0:   GRID   SEGMENT    BCTYPE      JSTA      JEND      KSTA      KEND     NDATA
         1         1      1001         0         0         0         0         0
IDIM: GRID   SEGMENT    BCTYPE      JSTA      JEND      KSTA      KEND     NDATA
         1         1      1002         0         0         0         0         0
J0:   GRID   SEGMENT    BCTYPE      ISTA      IEND      KSTA      KEND     NDATA
         1         1         0         0         0         0         0         0
JDIM: GRID   SEGMENT    BCTYPE      ISTA      IEND      KSTA      KEND     NDATA
         1         1         0         0         0         0         0         0
K0:   GRID   SEGMENT    BCTYPE      ISTA      IEND      JSTA      JEND     NDATA
         1         1      2004         0         0         0         0         2
               TWTYPE        CQ
                  0.        0.
KDIM: GRID   SEGMENT    BCTYPE      ISTA      IEND      JSTA      JEND     NDATA
         1         1      1003         0         0         0         0         0
      MSEQ    MGFLAG    ICONSF       MTT      NGAM
         1         1         0         0        01
      ISSC EPSSSC(1) EPSSSC(2) EPSSSC(3)      ISSR EPSSSR(1) EPSSSR(2) EPSSSR(3)
         0       0.3       0.3       0.3         0       0.3       0.3       0.3
      NCYC    MGLEVG     NEMGL     NITFO
        10        03        00       000
      MIT1      MIT2      MIT3      MIT4      MIT5      MIT6      MIT7     MIT8
        01        01        01        01        01         1         1        1
   1-1 BLOCKING DATA:
      NBLI
         1
 NUMBER   GRID     :    ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
      1      1            1      1      1      2      1     97      1      3
 NUMBER   GRID     :    ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
      1      1            1    193      1      2    193     97      1      3
  PATCH SURFACE DATA:
    NINTER
         0
  PLOT3D OUTPUT:
  BLOCK IPTYPE ISTART   IEND   IINC JSTART   JEND   JINC KSTART   KEND   KINC
      1      0      1     01      1     01    999      1      1    999      1
  MOVIE
      0
  PRINT OUT:
  BLOCK IPTYPE ISTART   IEND   IINC JSTART   JEND   JINC KSTART   KEND   KINC
      1      0      1     01      1     01    999      1      1    999      1
  CONTROL SURFACE:
  NCS
    0
   GRID ISTART   IEND   JSTART   JEND   KSTART   KEND  IWALL  INORM
```

The initial study is to investigate the effect of number and type of sub-iterations on the sub-iteration convergence of residual and drag. These levels are printed out automatically to the file `cfl3d.subit_res` (unit 23). (See Section 5.2.2.)

For this study, the following runs were performed (the coarser grid consists of every other point from the fine grid):

Grid	Δt	Sub-iteration Type
97×49	0.02	t-TS, multigrid
97×49	0.02	t-TS, no multigrid
97×49	0.02	τ-TS, multigrid
97×49	0.02	τ-TS, no multigrid
97×49	0.10	t-TS, multigrid
97×49	0.10	t-TS, no multigrid
97×49	0.10	τ-TS, multigrid
97×49	0.10	τ-TS, no multigrid
97×49	0.50	t-TS, multigrid
97×49	0.50	t-TS, no multigrid
97×49	0.50	τ-TS, multigrid
97×49	0.50	τ-TS, no multigrid
193×97	0.10	t-TS, multigrid
193×97	0.10	t-TS, no multigrid
193×97	0.10	τ-TS, multigrid
193×97	0.10	τ-TS, no multigrid

When multigrid was employed, a 3-level V-cycle was used.

Figure 8-2 shows the residual for density and the drag coefficient as a function of **ncyc** (number of sub-iterations +1) at a time step of 0.02. This is a fairly fine time step, yielding over 1000 steps per period. At this time step, both the t-TS and τ-TS with multigrid converge the drag with **ncyc** = 4 (3 sub-iterations) and t-TS and τ-TS without multigrid require about **ncyc** = 6 (5 sub-iterations). The residual also converges quicker with multigrid, as expected. Note that the t-TS with multigrid has a slightly better residual convergence rate than τ-TS with multigrid at this time step.

At a higher time step of **dt** = 0.10, the trends in Figure 8-3 are similar: both t-TS and τ-TS with multigrid converge the quickest, requiring about **ncyc** = 5, while the non-multigrid methods require about **ncyc** = 14-16. At this time step, the residual for τ-TS with multigrid converges slightly better than t-TS with multigrid. This time step corresponds to a little over 200 steps per period.

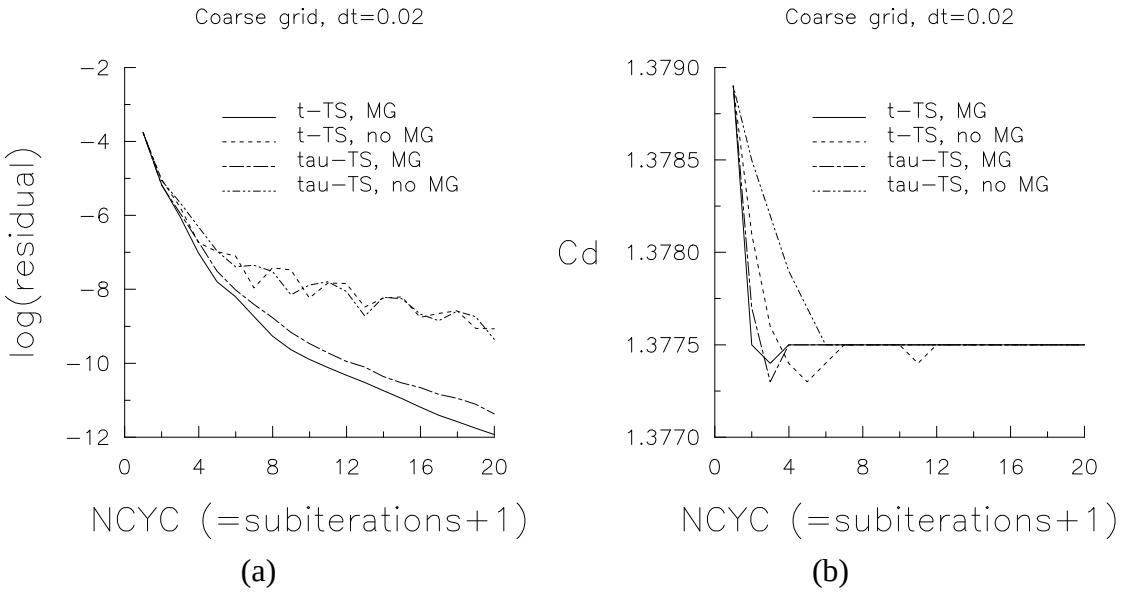

Figure 8-2. Coarse grid residual and drag coefficient histories for a single time step of $\Delta t = 0.02$.

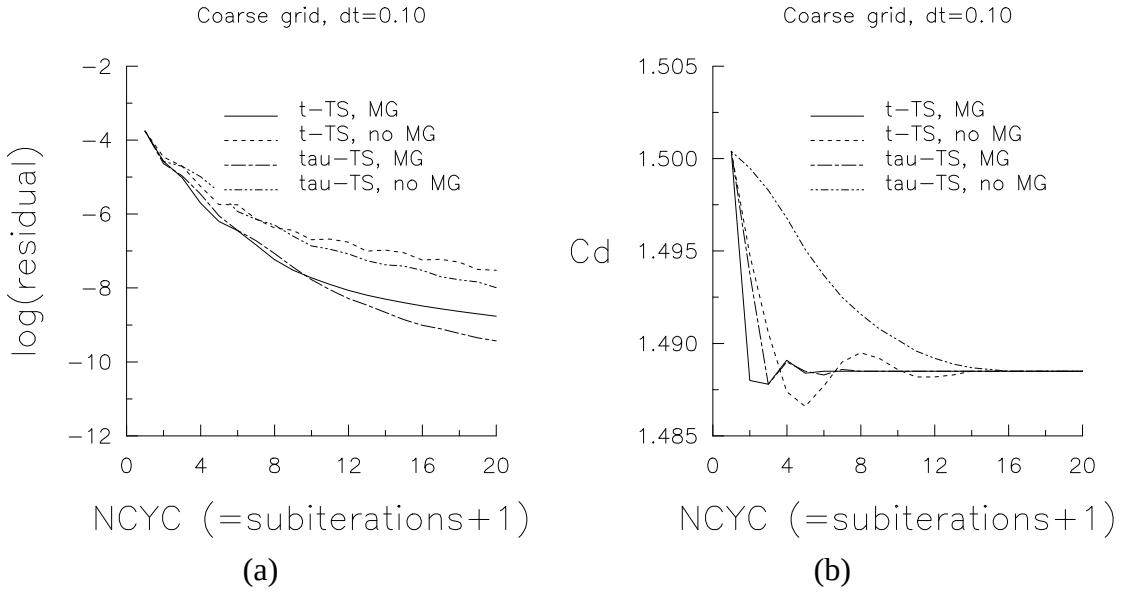

Figure 8-3. Coarse grid residual and drag coefficient histories for a single time step of $\Delta t = 0.10$.

At the highest time step of **dt** = 0.5, Figure 8-4, the multigrid sub-iterations require about **ncyc** = 12, while the non-multigrid sub-iterations are not fully converged even after **ncyc** = 40. Also note from Figure 8-4(a), at this time step the residual for t-TS with multigrid method now does not converge as well as either τ-TS method. This time step corresponds with less than 50 steps per period.

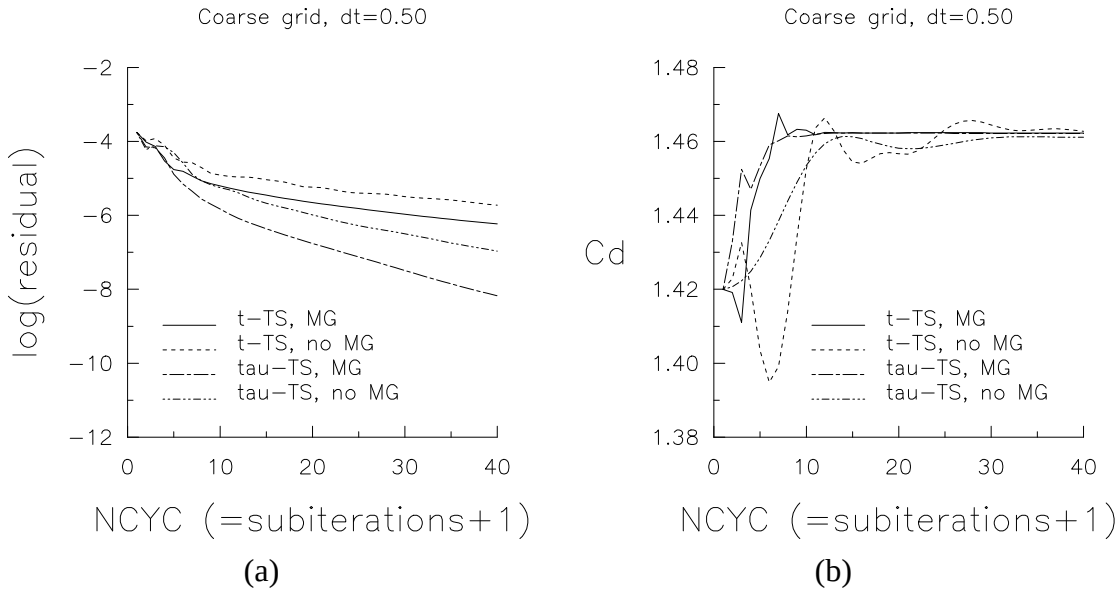

Figure 8-4. Coarse grid residual and drag coefficient histories for a single time step of $\Delta t = 0.50$.

Results for the fine 193×97 grid with **dt** = 0.1 are shown in Figure 8-5. Results are qualitatively similar to those in Figure 8-3. However, on this finer grid, more sub-iterations are required to converge the drag: 6-8 iterations for multigrid and well over 20 iterations for non-multigrid.

With a 3-level V-cycle, the multigrid method costs roughly 1.5 times as much as the non-multigrid method. (The τ-TS is only marginally more expensive than t-TS, so they may be considered essentially equivalent.) Hence, at the smallest time step of **dt** = 0.02, the user is roughly at a break-even point in terms of whether it is more efficient to use multigrid or no multigrid. However, at the larger time steps, using multigrid is clearly beneficial: for example, at **dt** = 0.1 (around 200 steps per cycle), multigrid converges the drag in approximately 0.36 the number of sub-iterations on the coarse grid at 1.4 times the cost; this means a savings of almost 50%! The savings is even greater on the fine grid. Due to this substantial savings, the remaining results will include only cases utilizing multigrid.

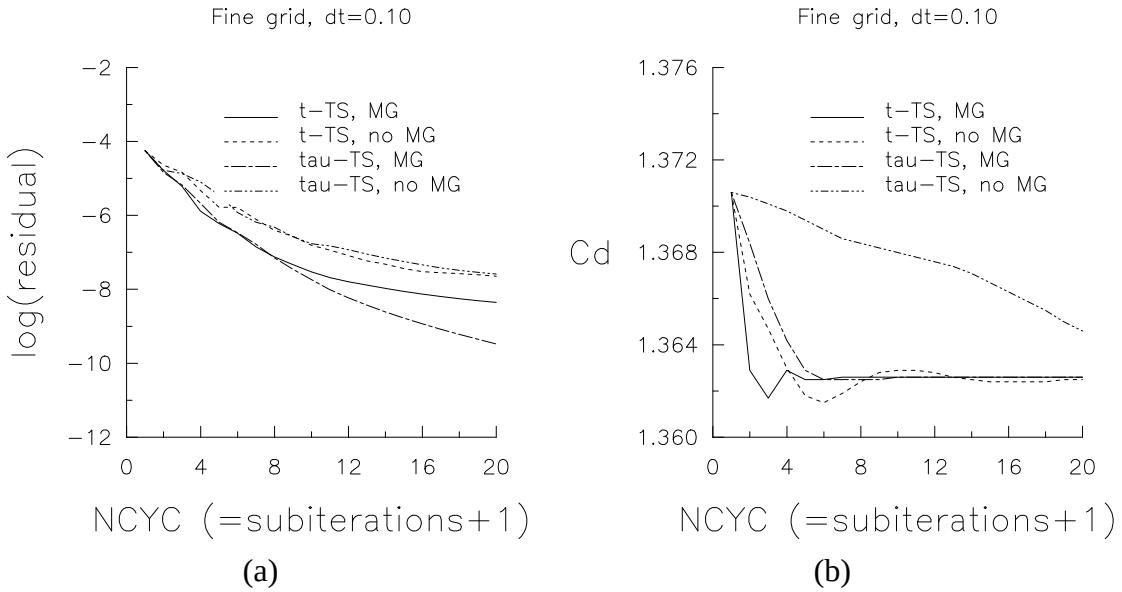

Figure 8-5. Fine grid residual and drag coefficient histories for a single time step of $\Delta t = 0.10$.

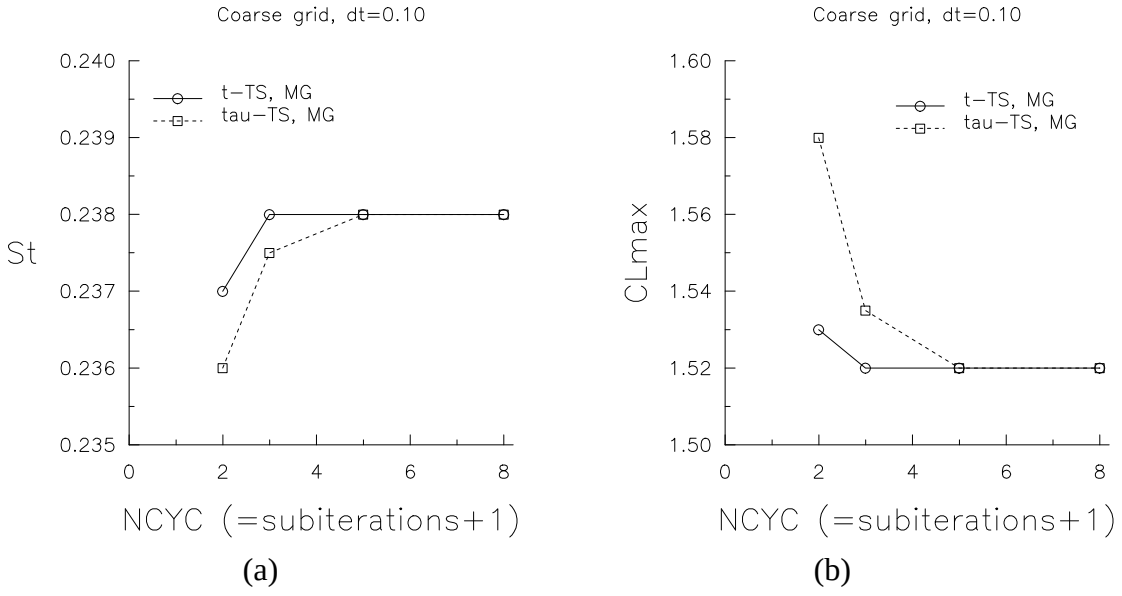

Figure 8-6. Coarse grid Strouhal number and lift coefficient histories for fully periodic solutions using $\Delta t = 0.10$.

Figure 8-2 through Figure 8-5 give a feel for the effect of sub-iterations on the convergence to the next physical time step *for one iteration*, but what is the effect of the number of sub-iterations on *global* quantities, over a long period of time? Figure 8-6 shows Strouhal number and maximum lift coefficient (in absolute value) as a function of **ncyc** for **dt** = 0.10 on the coarse grid, where the solution is obtained using the given value of **ncyc** over a long time (until periodic quasi-steady-state is reached). If at least 4 sub-iterations are run for this case (**ncyc** = 5), both t-TS and τ-TS with multigrid converge to the same result. This is consistent with the results for maximum lift coefficient shown in Figure 8-3(b). However, if less than this number of sub-iterations is run, then the t-TS method appears to give the better result.

For the higher time step of **dt** = 0.5, results are shown in Figure 8-7. If **ncyc** is less than about 7 for this time step, both t-TS and τ-TS sub-iterations with multigrid yield non-physical solutions (not shown in the figures). For example, t-TS yields a highly non-regular lift cycle, while τ-TS yields a regular lift cycle with non-zero mean. At least **ncyc** = 15-20 is required to converge the sub-iterative schemes sufficiently at this time step. This is roughly consistent with the results in Figure 8-4(b). If less than this number of sub-iterations is used, then the τ-TS method gives the better result. This is the opposite result from that given above for **dt** = 0.1, but it is consistent with the trend seen in the residual plots of Figure 8-2(a), Figure 8-3(a), and Figure 8-4(a). In other words, it appears that the t-TS method may require less sub-iterations at low time steps, while τ-TS requires less sub-iterations at higher time steps.

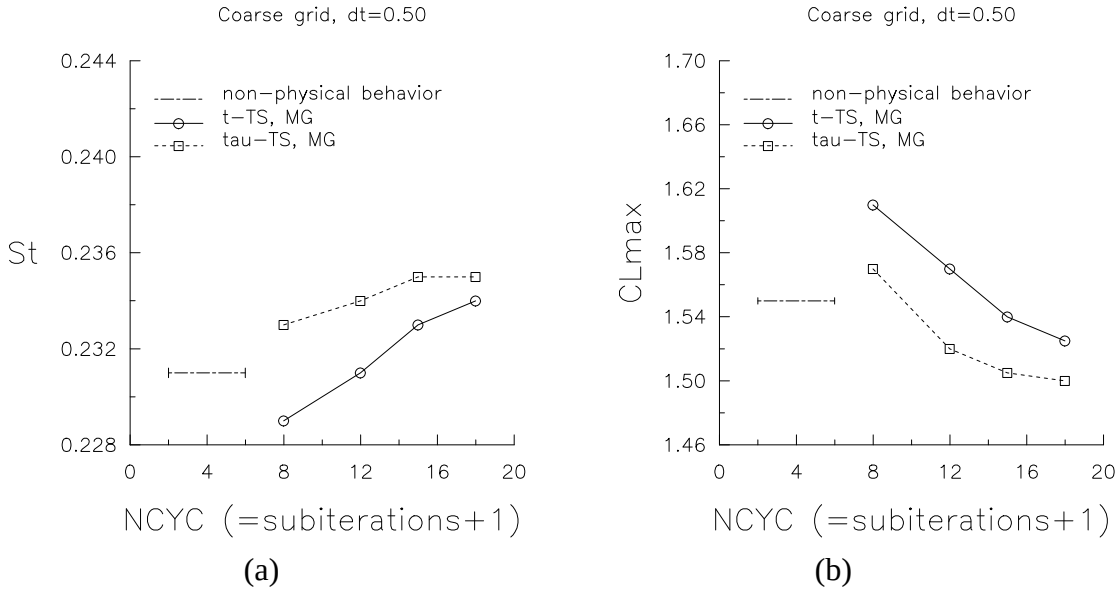

Figure 8-7. Coarse grid Strouhal number and lift coefficient histories for fully periodic solutions using $\Delta t = 0.50$.

Other time-accurate circular cylinder cases with turbulence model(s) employed (not shown), have revealed similar trends to the study shown here. One point worth mentioning in connection with these other cases is that, in some instances, the t-TS method with multigrid has been seen to either diverge or else give nonphysical answers, *regardless of the number of sub-iterations taken*, when the time step is too large. Unfortunately, determining what time step is too large remains elusive at this point. Trial and error seems to be the only way to determine it for such cases. Therefore, since the τ-TS method has not exhibited such errant behavior, it appears that, as a general rule, the safest bet is to go with the τ-TS method rather than t-TS.

From this study, combined with experience running the CFL3D code for time-accurate cases, the following conclusions are made:

1. In general, it is recommended that the user employ multigrid when using sub-iterations.
2. The larger the time step (the less steps per period), the more sub-iterations are required to converge the sub-iterative scheme.
3. The larger the grid, the more sub-iterations are required to converge the sub-iterative scheme.
4. t-TS and τ-TS are roughly equivalent in their ability to converge a quantity like "drag". However, for lowering residual, t-TS is slightly more efficient than τ-TS at small time steps, while the reverse is true at higher time steps.
5. t-TS appears to require slightly less sub-iterations at low time steps, while τ-TS appears to require slightly less sub-iterations at higher time steps.
6. Since it is not possible to know in advance what time step is "low" and what is "high" (in connection with conclusions 4. and 5.) and since t-TS has been known to *not converge* regardless of the number of sub-iterations for some cases when the time step is too high, it is recommended that τ-TS (with multigrid) be used in practice as a general rule.

Also, the following recommendation is made. When performing time-accurate computations, *always* monitor the `cfl3d.subit_res` file (unit 23) in order to insure that the sub-iterative scheme is converging sufficiently. Additionally, it is recommended that the user perform a time step study (vary **dt**), along with the usual grid density study, to determine the solution's sensitivity to time step.

8.3 Convergence Criterion for Sub-iterations

The sub-iteration equation (Equation (B-7), with $\phi' = 0$) is

$$\left[\left(\frac{1}{J\Delta\tau}+\frac{1+\phi}{J\Delta t}\right)I+\delta_\xi\mathbf{A}+\delta_\eta\mathbf{B}+\delta_\zeta\mathbf{C}\right]\Delta\mathbf{Q}^m = \frac{\phi\Delta\mathbf{Q}^{n-1}}{J\Delta t}-\frac{(1+\phi)(\mathbf{Q}^m-\mathbf{Q}^n)}{J\Delta t}+R(\mathbf{Q}^m) \tag{8-1}$$

for $m\to\infty$, $\mathbf{Q}^m\to\mathbf{Q}^{n+1}$, and the left-hand side, which is the sub-iteration residual, $R_{subit}(\mathbf{Q}^m)$, approaches 0. Write the sub-iteration residual as

$$R_{subit}(\mathbf{Q}^m) = \frac{\phi\Delta\mathbf{Q}^{n-1}}{J\Delta t}-\frac{(1+\phi)(\mathbf{Q}^m-\mathbf{Q}^n)}{J\Delta t}+R(\mathbf{Q}^m) \tag{8-2}$$

For $m\to\infty$,

$$R_{subit}(\mathbf{Q}^m)\sim\frac{\phi\Delta\mathbf{Q}^{n-1}}{J\Delta t}-\frac{(1+\phi)(\mathbf{Q}^{n+1}-\mathbf{Q}^n)}{J\Delta t}+R(\mathbf{Q}^m) \tag{8-3}$$

where $\frac{\phi\Delta\mathbf{Q}^{n-1}}{J\Delta t}-\frac{(1+\phi)(\mathbf{Q}^{n+1}-\mathbf{Q}^n)}{J\Delta t}\Rightarrow \text{discrete}-\frac{\partial\mathbf{Q}}{\partial t}$.

Then

$$\frac{\partial\mathbf{Q}^n}{\partial t} = R(\mathbf{Q}^m)-R_{subit}(\mathbf{Q}^m) \tag{8-4}$$

where $\frac{\partial\mathbf{Q}^n}{\partial t}=R(\mathbf{Q}^m)\Rightarrow$ discrete Navier-Stokes equations. Therefore, to insure that the discrete Navier-Stokes equations are solved accurately, $R_{subit}(\mathbf{Q}^m)\ll R(\mathbf{Q}^m)$ are needed. These values are calculated in the code. $R_{subit}(\mathbf{Q}^m)$ (for $\mathbf{Q}$ = density) is output to `cfl3d_subit.res`, while $R(\mathbf{Q}^m)$ (for $\mathbf{Q}$ = density) is output to the user-specified file on unit 12 (typically called `cfl3d.res` in the sample input files).