The Focus

PADT

December 13, 2006 | A Publication for ANSYS Users | Issue 54

Radiation Redux: Radiation is Even Easier

By: Rod Scholl

After last months article was published on the ease of implementing radiation in a thermal model (link), someone at ANSYS, Inc. was nice enough to point out that some of my information was quite outdated.

I wrote:

For particularly hairy radiation problems you might not be able to chose a uniform starting temperature that yields convergence. This has been the rare case for me, but when encountered, I make intelligent guess for a few different reasons. In an extreme case, this too does not converge, and I first run a steady state conduction solution on my initial condition guesses (without radiation present) then add radiation and resolve. Something about the smooth temperature gradients eases convergence.

It seems ANSYS, inc. added considerable convergence stability in a subsequent version of ANSYS. My information was from a script implemented in 5.7 (wow, that makes me feel old – I know many users go back to 4.4 and I've have had to run that occasionally – so I guess at least I'm not as old as THEM!) Anyway, when testing out an old script, what was troublesome in 5.7 converged easily in 6.1.

"Smart" Initial Conditions not necessary:

I mentioned that some rare cases required testing various TUNIF's, or even complete conduction solutions implemented before adding radiation effects. This is no longer required. I tried my old script and sure enough it converged to the same answer regardless of what I started at.

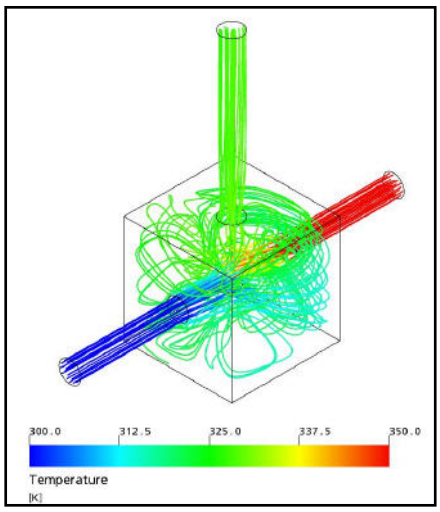
False Transient:

This method also was implemented for stability in 5.7 (when only a static solution was required). So this, and the QSOPT method which is similar, is also unnecessary. Just hit solve, and if you want specify some substeps (NSUBST) which might help, or let ANSYS figure it out on its own using solution control, auto-time-stepping, and a couple bisections in the rare case they are needed.

Test Script:

This test script (link) is similar to the others (link) but uses the static solution, rather than a false transient. The results are quite similar.

(Cont. on Pg. 2.)



By: J. Luis Rosales

ANSYS CFX: Your First CFX Model – As Easy as 1-2-3

If you haven't had the chance to try CFX, this article will show you how easy it is to setup and run a model. Of course, "how easy" depends on the complexity, physics, etc. of your model, but in general most problems will be easy to setup. This problem simulates the mixing of two different temperature water streams entering from horizontal tubes into a plain box geometry and leaving through a third protruding tube. The geometry of the model is shown in Figure 1.

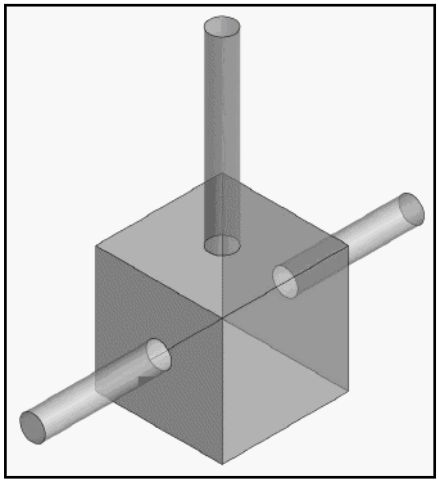


Figure 1: Simple mixing box model for two different temperature water streams.

Step 1: Importing the Geometry

The first step is to read in the grid. The grid for this example was generated in ICEM-CFD but it doesn't matter where you build it – ANSYS, CFX-Mesh, etc. If you don't already have CFX-Pre open then you have to open the ANSYS CFX Launcher and browse to set the working directory. Once you have this done then simply click on CFX-Pre in the Launcher to start it up.

Once CFX-Pre is open you will be staring at a blank gray screen and just

(Cont. on Pg. 2.)

Contents

(Radiation, Cont.)

What was implemented:

We'll start with the familiar radiation equation:

Which is now implemented in ANSYS using an effective heat transfer coefficient. Note how the temperature difference is no

$$q_k^{net} = \varepsilon_k \sigma \left(\mathrm{T_k^4} - \mathrm{T_\infty^4} \right) \quad (1)$$

longer of the 4th order temps:

So of course the other T^3 terms have to be accounted for, and are in the definition of h:

$$q_k^{net} = h\left(\mathrm{T}_k - \mathrm{T}_\infty \right) \quad (2)$$

If you do a little algebra you can see that equation (1) and (2) are equivalent when the

$$h = \varepsilon_k \sigma \left(\mathrm{T_k^2} + \mathrm{T_\infty^2} \right)\left(\mathrm{T}_k + \mathrm{T}_\infty \right) \quad (3)$$

$$\mathrm{T}_\infty = \left(\frac{1}{\sigma} \sum_{j=1}^{N} \mathrm{F}_{kj} q_j^o \right)^{1/4} \quad (4)$$

substitution for h is made. Thus for a given iteration and temperatures, h is calculated and implemented into the K matrix. The resulting linear nature of equation s(2) vs (1) makes the solution robust enough that I couldn't get it to diverge, with only minor results differences from the old method to the new (probably related to time step size).

Want to learn more about advanced Heat Transfer, Take PADT Course 301: Heat Transfer. Visit www.padtinc.com/support/training/course.asp?c=301 for more information.

(CFX, cont.)

about all the icons are grayed out. This will change as soon as you read in your grid. Note: I may use grid or mesh interchangeably through out this and other articles but I am writing about the same thing. You can either click on **File** and select **New Simulation…** under the menu or click on the first non-grayed icon that looks like a paper with a flashing bulb,  . Enter any name you want in the **New Simulation File** menu inside the **File Name:** window. I used the name *MixBox*. Just leave everything else as default – that means that the Simulation Type will be set to General. After you click on the **Save** button, the **Mesh** tab will automatically be selected for you. Click on the **Import Mesh** button, which will be the only icon not grayed out from the vertically oriented icon set to the right. Down below you will now see the **Definition** tab appear. Since this grid was built in ICEM-CFD, I will select ICEM CFD as the Mesh Format and search and select the file inside the **File** window. For a single domain, there is no need to enter an Assembly Prefix unless you want to enter one. I always try to work in SI units so the Mesh Units are set to **m** for meters. Click OK and your CFX-Pre window should appear as shown in Fig. 2.

Importing the grid into CFX-Pre is not really a step to setting up a model in CFX so it can really be discounted as a step. Now that we have a grid loaded, we can look at the real steps of setting up the model in CFX-Pre.

Ok, now you are staring at a lot of non-grayed out icons laid out horizontally just beneath the pull-down menu items. The big question is where do you start. Let's jump all the way to the icon that looks like a clock,

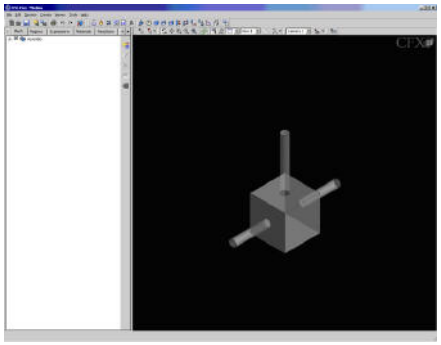


Figure 2: CFX-Pre viewer after loading the mixing box model

 . It is located a little more than half ways along the horizontal row of icons. If you are wondering about the icons we skipped, most of them are pretty straightforward like printing and saving files. Others are there to help you solve models with more complicated physics. I will be addressing all the icons briefly in future articles and some in much detail in specific articles that relate to their particular capability.

Step 2: Defining the Simulation Type

This icon is used to define whether the model is to be solved transiently or steady state (static). If the model is to be solved using a steady-state approach then don't even bother clicking on this button since steady state is the default setting. The current example will be a steady-state solution so we can skip this icon.

Step 3: Defining the Domain

The next icon,  , is used to define the domain. This is usually the very first step you will take unless you are solving a transient model. Click on the Create a Domain icon and you will get the Create Domain window. Enter any name you want. I will use WATER since that is the fluid I will be using. The window Edit Domain: Water appears as shown in Fig. 3. You should now be staring at several window settings but don't be alarmed since you will only change one or two. Let's quickly walk through them.

Under Basic Settings, the first setting is Location. Since you only imported one assembly, it is already selected and you can leave this alone. The second setting is Domain Type. We will be using water throughout the domain, so the Fluid Domain setting is already set. The next setting is the first one you

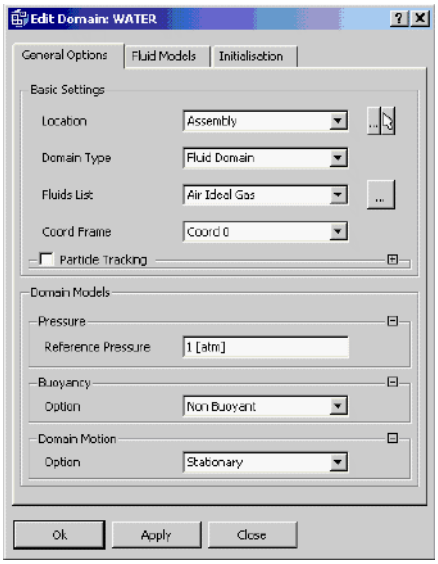


Figure 3 Edit Domain: Water Menu

need to change. Air Ideal Gas is shown in the Fluids List. Change this setting to Water. Below that you can leave Coord 0 alone for the Coord Frame setting. We will not be solving for Particle Tracking so leave the box unchecked.

Under Domain Models, leave the Reference Pressure set to 1 [atm]. There will be no buoyancy effects so keep

(Cont. on Pg. 3.)

(CFX, Cont.)

the option set to Non Buoyant for the Buoyancy Option. The last option, Domain Motion is left alone and set to Stationary.

Note, the only setting that was changed was the Fluids List – very easy. Click on the Fluid Models tab.

Change the Option for Heat Transfer Model to Thermal Energy and keep the Option for Turbulence Model set to K-Epsilon. Ignore the rest of the models, as they are not needed for this example. You're done! You have set up the domain parameters. You have probably noticed the Initialization tab but we will initialize the domain from the global domain initialization icon. You may be wondering how the other settings work. Well, some of them are easily understood by their names and the others can be looked up in the CFX help. I don't want to complicate the article by giving too many details but future articles will give you the information needed to use all the settings.

The next two icons,  , will be skipped since they are not needed for this example. They are the sub-domain icon and the point source icon. I will provide information for these icons in future articles dealing with those types of problems.

Step 4: Defining Boundary Conditions

The next icon,  , is used to specify the boundary conditions for the model. Again, I will show you how easy it is to do this in CFX-Pre. The boundary names on the grid for this mixing model were defined in ICEM CFD. Click on this icon to open up the Create Boundary menu. Enter the name INLET HOT for the boundary name and WATER is the only domain that you can pick so click on the OK button. The Edit Boundary: INLET HOT in domain: WATER window will open up. Under the Basic Setting tab, ensure that INLET is chosen for the Boundary Type and INLET1 is selected for the Location. Under the Boundary Details tab leave the Subsonic setting for the Flow Regime Option. Enter a value of 1 m/s for the Normal Speed under Mass and Momentum. Note: just enter the number as the units are selected in the pull-down menu to the right. Ensure that Medium (Intensity = 5%) is set for the Turbulence Option and enter a value of 350 K for the Static Temperature under the Heat Transfer Option. Click Ok and the first inlet boundary is done.

Click on the boundary condition icon again and enter the name INLET COLD for the boundary name. Again, WATER is the only Domain setting. Click OK to open up the Edit Boundary: INLET COLD in domain: WATER window. Under the Basic Settings tab, Inlet should be the Boundary Type and select INLET2 for Location. Under the Boundary Details, enter 1 m/s for Normal Speed and 300 K for the Static Temperature. Click OK and the second inlet boundary is done.

There is one more boundary to define. Click on the boundary condition icon once more and enter the name OUTLET for Name. Change the Boundary Type to OUTLET and select OUTLET for the Location under the Basic Setting tab. Change to the Boundary Details tab. Enter a value of 0 for the Relative Pressure using the Average Static Pressure Option under Mass and Momentum. Click OK and the last boundary condition is done.

One thing you probably noticed is that the boundary you selected under Locations was highlighted for ease of recognition. Also, as soon as you click on OK, arrows pointing inwards appear for the INLET boundary condition type while arrows pointing outward appear for the OULET boundary condition type. These are nice features that can help greatly for setting up more difficult models. You may be asking about the other remaining surfaces of the model. Do they need to be defined? It depends on the model. In this example, the remaining surfaces are all walls and by default any surface that is not defined as we did above becomes a wall and is also adiabatic. This is what we want here so we do not need to define any further boundaries. Of course, if the walls had a temperature and were different for each tube, then we would need to define a few more boundaries. To make setting up the model easier, try to give your grid boundaries names that are meaningful when building your mesh, i.e. INLET1, INLET HOT instead of BOUNDARY1.

The next icon,  , is used to create domain interfaces. We have but a single domain so we can skip this icon.

Step 5: Global Initialization.

The following icon,  , is used to define global initial values. Well let me give you some information about initialization. You can probably remember that we skipped the Initialization tab when we first defined our domain. Ok, where should we specify the initial conditions? Anywhere you want as long as you know what is happening. The initial conditions specified in the domain menu supersede those from the global specification for the domain that is being defined. In our example, we have only one domain so if we had entered values, they would supersede those specified here. Also, if we had entered any values, there is no reason to open up this global initialization icon. The global initialization icon works great when you have many domains and they have the same initial conditions. Instead of entering the same values over and over again for all the domains, you just do it once at the global level. Again, any local initial values will supersede the global value just for that domain. Click on the Global Initialization icon and you should see a window as shown below in Fig. 4. For a steady-state model, you can keep the setting set to automatic and let CFX take a stab at them. For complex models, that is basically what you would use since it would be difficult to give a good initial value for velocities. When the K-Epsilon model is selected make sure that the Turbulence Eddy Dissipation check box is set. You can keep the Option set to Automatic also.

The next icon,  , is used to adapt the mesh. That won't be used for this example so we can skip it.

Step 6: Solver Control Setting

We are almost done. The next icon of interest,  , opens up the Solver Control window. Unlike other programs that have a ton of possible settings for solver controls, CFX is quite simple. The solver itself is quite robust so you don't have to worry about the multitude of options presented in other codes.

Click on the Solver Control icon to open up the window shown below in Fig. 5. Let's begin with the Advection Scheme. It has three possible settings: High Resolution, Upwind and Specified Blend Factor. The upwind scheme is a simple and robust convective scheme but can be very diffusive. It can be used to start up difficult problems but in general should be avoided. The Specified Blend Factor allows the user to enter a value between 0 (upwind) and 1 (2nd Order), this will fix the order of the scheme. The High Resolution option will attempt to set the

(Cont. on Pg. 4.)

(Radiation, Cont.)

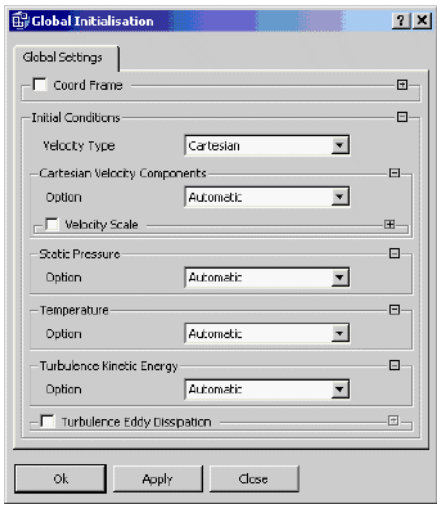


Figure 4: Global Initialization Menu

order of the scheme as high as possible automatically while keeping the solution bounded everywhere.

The High Resolution scheme is what I would recommend for users. I will go into more details about the advection schemes in future articles but for the most part, stick to the High Resolution scheme. Under Convergence Control, use the Auto Timescale for Timescale Control and also leave the value of 100 for Max Iterations. Keep the Length Scale Option set to Conservative. Change the Residual Type from RMS to MAX and leave the Residual Target at 1.E-04. A residual value of 1.E-04 for the RMS setting is good for obtaining a faster solution and to get an idea of how the flow is developing.

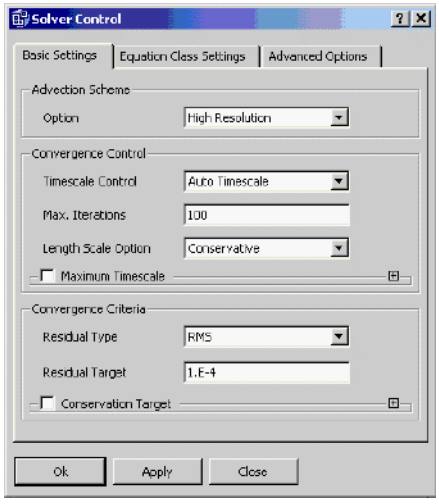


Figure 5 Solver Control menu

Setting the value of 1.E-04 for the MAX value gives a tighter convergence and is pretty good for extracting quantitative results for most problems. Click Ok to accept the settings.

The next icon,  , is useful for saving information during the solution of a model. You can save as much or as little information as you want and you also have several options to control when you save the information. We will skip this icon for this example.

Step 7: Saving the Simulation

The final icon on the horizontal row of icons,  , is for writing out the solver file. The files that are written are MixBox.def, MixBox.cfx and the MixBox.gtm.

The MixBox.cfx file contains the information entered during the CFX-Pre session and the MixBox.gtm contains the grid information. Both of these are combined to form the MixBox.def file. The *.def file is used in CFX-Solver. If you are trying to save space on your computer, you can delete the MixBox.cfx and MixBox.gtm files and just keep the MixBox.def file. Click on the icon to get the Write Solver File window as shown in Fig. 6. The name of the model is automatically used in the File name: input window. You can change the name if you

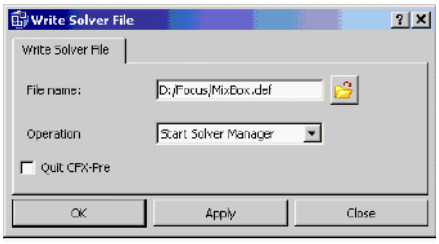


Figure 6 Write Solver File Menu

want. Under Operation, the setting is such that the solver manager will automatically start up and use the new MixBox.def file. Generally, I click on the Quit CFX-Pre check box to close the preprocessor but you can keep it open if you wish. You are now done! Click on the OK button to write out the solver file and shut down the preprocessor (if you selected the quit option for CFX-Pre). You may get a series of menus so just click on the Save & Quit button.

The CFX-Solver Manager window will open up as shown in Fig. 7. All you have to do is to click on the Start Run button and you will be on your way to solving the model. You can rest assured that if you need more options, such as distributed processing and memory resource control those are easily at your disposal. I will go into the details of setting some of the advanced options in the solver manager in future articles so if have an interest in using this option then you won't want to miss some of the future articles.

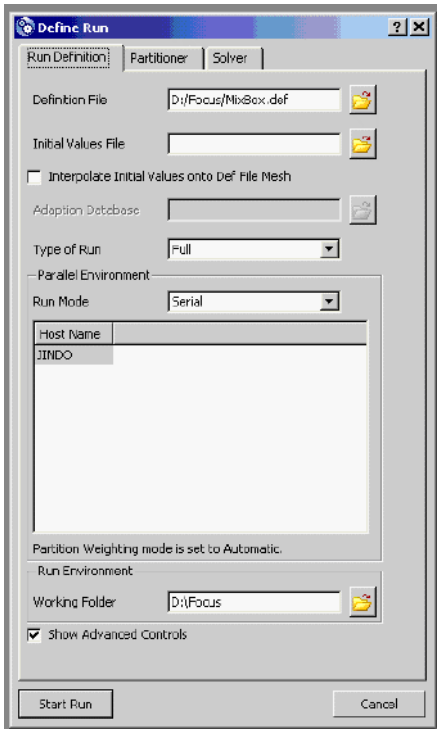


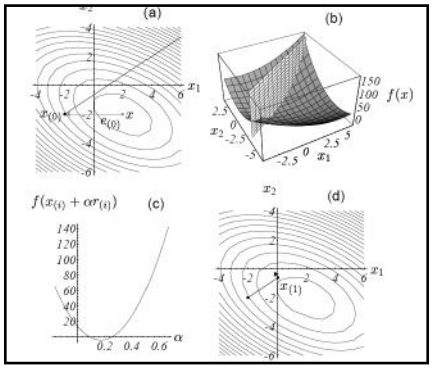
Figure 7: CFX-Solver Manager Window

A previous article discussed some of the features of the solver manager so interested users should go back and read this article. Post processing the results in CFX-Post is relatively simple. CFX has a very good help section on post processing so this won't be discussed here. Some specific post-processing features will be presented in future articles, such as writing script files for quick results extraction.

Summary

I hope I was able to show you how simple it is to setup a model in CFX-Pre. If you have a comparable model as that shown here, you can more or less follow the same steps and easily obtain the results you need. ANSYS CFX has done a very nice job of providing a simple user interface but remember that you have a lot of sophisticated physics at your disposal should you ever need them. Keep looking for future FOCUS articles where I will start to discuss the setup of more complicated and interesting flow models.

Shedding Some Light on the PCG Solver



By: Eric Miller

During a recent event with ANSYS, Inc. I was asked to discuss the effect of some new hardware from Intel and HP on the CAE industry. During one of the presentations we were waxing poetic about how much better the Lancosz PCG solver in 11.0 is and a technical writer in the audience came up afterwards and asked an awkward question: "What is a PCG solver?" I gave her the standard response of "solve a system of equations by guessing the solution then using the "shape" of the equations to iterate towards a converged solution." After she turned and walked away, it dawned on me that I did not give a good explanation and that fundamentally, I do not understand this critical tool in the ANSYS solver library fully, which leads us to this article, perhaps the driest exposition to ever grace the pages of *The Focus.*

My quest for understanding started in the ANSYS manual and moved to the Internet I became more baffled. I should have paid more attention in linear algebra lectures.

Then I found a paper by a Johnathan Shewchuk of Carnegie Mellon with the title: "An Introduction to Conjugate Gradient Method Without the Agonizing Pain." Check out the abstract, this guy has a great sense of humor. Even thought this is a nice summary, there is still no way to boil the math down into two or three paragraphs. So if you are interested in really understanding the equations and such, please read the article… several times. An algorithm for the actual PCG solution process is given in the ANSYS Theory manual as well as this CMU paper.

Why Iterate?

When solving in ANSYS we are generally solving for f = [K]u where f is the load vector, K is the stiffness matrix and u is the displacement vector. (Substitute other values for other physics). Math folks generalize this as Ax=b. For most problems you use some method to factorize A into a triangle then calculate unknowns in x. ANSYS does this for the SPARSE and FRONTAL solvers. But if A is big or dense, this can require a lot of disk space and memory and can take a long time. In fact, a good rule of thumb for the ANSYS sparse direct solver is 10 GB of disk and 1 GB of RAM per 1 million DOF. In an iterative solver, you do not need as much disk space and therefore you use a lot less disk I/O, so things run faster. And you generally need just the 1GB of RAM per 1 million DOF.

If you are still reading, you are probably asking "So, ANSYS guesses at the solution?" Well, not exactly. Turns out that if you take a system of equations like this, and if it is symmetric and positive-definite, you can rearrange it into a quadratic formulation and the solution to the equation is at the minimum of that quadratic surface.

So you don't just randomly choose solutions, you start with a value of x=0, find where you are in the quadratic function surface, take the derivative of the system to get gradients, and then use the gradients to pick the next solution, using some very hairy math. (The study of which actually put me to sleep twice, very embarrassing since everyone walks in front of my office.) Turns out that the math shows that there are two vectors that are conjugate, making it even easier to iterate towards a solution by moving along the steepest gradient – which is why they call it a Conjugant Gradient solver.

What is Preconditioning?

The P in PCG stands for Preconditioning. This is the part of the method that makes the solver so fast because the condition of the system, how close its quadratic form is to the ideal (spherical), determines how many iterations it takes to get to an acceptable solution. (By the way, you can calculate the condition of a matrix by finding the ratio of its largest and smallest eigenvalues, something we do not do because it requires a full solve.)

So, the smart mathematician types sat down and figured out that if you can come up with a matrix (M) that approximates your stiffness matrix, invert it and add it to both sides of your equation (M-1Ax = M-1b) you can change the shape of your quadratic form of the equations. By picking the proper preconditioning matrix you can make the quadratic form more spherical and therefore better conditioned resulting in fewer steps to convergence. The down side is that you need to pick the proper preconditioning matrix, store it in memory (it is only slightly smaller than the stiffness matrix) and add an additional matrix-vector multiply to every iteration step. So you need to make sure that the added cost results in a faster solution (see PCGOPT command discussion below).

PCG Usage in ANSYS

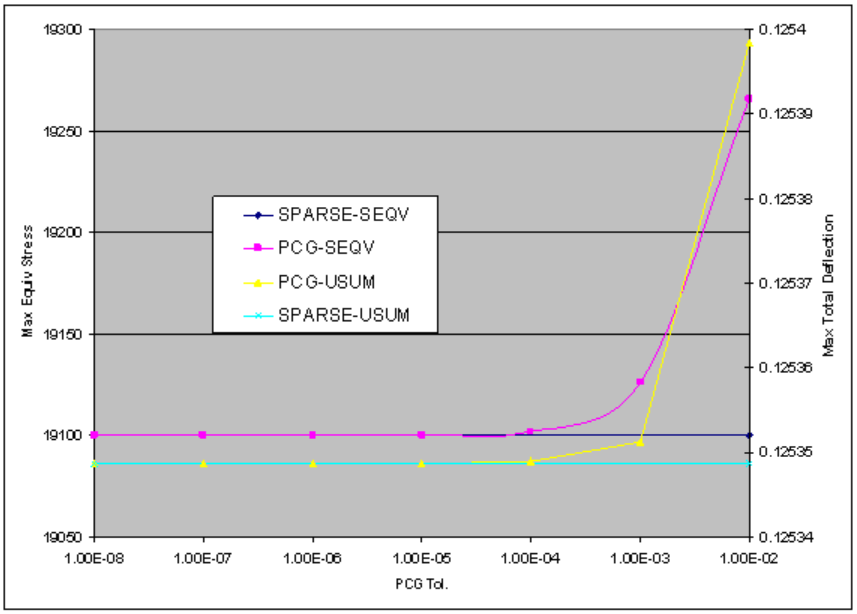
If you read the paper and understand most of it you will find that it discusses several conjugate gradient solvers, many of which are in ANSYS. For the PCG solver, ANSYS, Inc. uses a proprietary preconditioner that does a very nice job. Unfortunately, because it is proprietary, we don't know much about how it works other than the fact that it is aware of things like material properties and various special element formulations.

Over the years ANSYS has invested a large amount of effort into their PCG and SPARSE technology and every release sees improvements in performance. The solver now works with most analysis types, including an accurate modal solution with the LANPCG solver in 11.0. A lot of work has also gone into making the PCG solver shared and distributed memory parallel. However there are some things, like Full Lagrange Contact, that don't work with the PCG solver, so check the manual before doing anything out of the ordinary. Some good information can be found in the discussion on solvers in the Basic Analysis

$$
\begin{aligned}
\|e_{(i+1)}\|_A^2 &= e_{(i+1)}^T A e_{(i+1)} \\
&= (e_{(i)}^T + \alpha_{(i)} r_{(i)}^T) A (e_{(i)} + \alpha_{(i)} r_{(i)}) \qquad \text{(by Equation 12)} \\
&= e_{(i)}^T A e_{(i)} + 2\alpha_{(i)} r_{(i)}^T A e_{(i)} + \alpha_{(i)}^2 r_{(i)}^T A r_{(i)} \qquad \text{(by symmetry of } A\text{)} \\
&= \|e_{(i)}\|_A^2 + 2\frac{r_{(i)}^T r_{(i)}}{r_{(i)}^T A r_{(i)}}\left(-r_{(i)}^T r_{(i)}\right) + \left(\frac{r_{(i)}^T r_{(i)}}{r_{(i)}^T A r_{(i)}}\right)^2 r_{(i)}^T A r_{(i)} \\
&= \|e_{(i)}\|_A^2 - \frac{(r_{(i)}^T r_{(i)})^2}{r_{(i)}^T A r_{(i)}} \\
&= \|e_{(i)}\|_A^2 \left(1 - \frac{(r_{(i)}^T r_{(i)})^2}{(r_{(i)}^T A r_{(i)})(e_{(i)}^T A e_{(i)})}\right) \\
&= \|e_{(i)}\|_A^2 \left(1 - \frac{(\sum_j \xi_j^2 \lambda_j^2)^2}{(\sum_j \xi_j^2 \lambda_j^3)(\sum_j \xi_j^2 \lambda_j)}\right) \qquad \text{(by Identities 21, 22, 23)} \\
&= \|e_{(i)}\|_A^2 \omega^2, \qquad \omega^2 = 1 - \frac{(\sum_j \xi_j^2 \lambda_j^2)^2}{(\sum_j \xi_j^2 \lambda_j^3)(\sum_j \xi_j^2 \lambda_j)} \qquad (25)
\end{aligned}
$$

(Cont. on Pg. 6.)

(PCG, cont.)



Guide and the Theory Manual. Both give some good insight to all of the solvers from a practical perspective.

Options for the PCG Solver

The most important PCG solver option to know about is the TOLER option on the EQSLV command. You choose the PCG solver with EQSLV, PCG then give a tolerance. If you remember, we are trying to find the minimum of the quadratic form of our system of equations. But getting to zero takes time so we stop when we are close. The theory manual gives some insight to what this number is, but for most cases, setting the tolerance to 1e-5 is good. The default is 1e-6 or 1e-8 depending on various factors and is probably too tight, but you should always be safe and err to precision. The primary reason for caution is that the error is an estimate of the DOF solution variation. But values like stress are derived from that solution and the derivation exaggerates the error. Running a simple cantilever beam, you can see the effect of tolerance choices on stress and deflection in Figure 1.

If you are trying to solve a very large model that does not fit in your RAM then use the MSAVE,ON to tell ANSYS to solve one equation at a time rather then loading everything in memory. This slows things down but is often the difference between solving and not solving, and if you are swapping a lot, it may be faster.

More Preconditioning

When dealing with ill-conditioned matrices, you can invest more time and memory into the preconditioner by raising the level of difficulty. By default, ANSYS looks at your problem and picks a level of difficulty. But if your problem is having a hard time iterating towards a converged solution, look at raising the level with PCGOPT. You can also use PCGOPT to reduce the amount of I/O used, perform a Sturm Check on a LANPCG run, write out the full stiffness matrix, or force where a Lev_Diff=5 run is solved.

See Rod's article on this for some examples.

The PCGOPT command has some other nice features. The most important to be aware of (study the documentation to learn about the others) is the out of core (OOC) memory option when Preconditioner difficulty gets to 5. The problem with this high Preconditioning level is that the M matrix is now bigger then your original K matrix. So if you are pushing memory to begin with, you may crash on very large, ill-conditioned problems. Using this option slows down your solve but fits it into your available memory.

PCG in Modal Analysis

Most of us tried the PCG modal solver a while ago, PowerDynamics, and found that it was fast but could miss modes. Not a good thing! In 11.0 you will be able to use the PCG solver instead of the sparse solver for the Block-Lancosz solver with EQSLV, LANP. This has the same advantage as for static runs – You do not do nearly as much disk I/O and files are much smaller. As an example, I extracted 25 modes from an 18 Million DOF model on a 32GB machine in about 20 hours. Something I could not even attempt with LANC. So if you have the RAM and 11.0 Beta, try LANPCG, even on smaller models.

In a simpler example, I ran a 1 million DOF cantilever beam modal analysis, extracting 10 modes, on my 4GB Core2Duo, NPROC=2, running Windows XP x64 in ANSYS 11.0 Beta. The LANB solve took 4107 seconds and the LANP took 511 seconds, or four times faster. Big difference!

PCG and You

For the near future, PCG looks like a very efficient way to solve most problems but everyone needs to play with options for their typical problems to make sure, You really should only use the SPARSE solver on highly non-linear or ill conditioned problems. But, PCG needs memory to deliver on performance, so you should pile on as much RAM as you can. Everyone should be moving to 64bit operating systems and getting at least 4GB of RAM. At 4GB you have about 3GB of useful memory which lets the PCG solver do 3 million DOF in core. If you can get more, do so.

$i \Leftarrow 0$
$r \Leftarrow b - Ax$
$d \Leftarrow M^{-1}r$
$\delta_{new} \Leftarrow r^T d$
$\delta_0 \Leftarrow \delta_{new}$
While $i < i_{max}$ and $\delta_{new} > \varepsilon^2 \delta_0$ do
  $q \Leftarrow Ad$
  $\alpha \Leftarrow \frac{\delta_{new}}{d^T q}$
  $x \Leftarrow x + \alpha d$
  If i is divisible by 50
    $r \Leftarrow b - Ax$
  else
    $r \Leftarrow r - \alpha q$
  $s \Leftarrow M^{-1}r$
  $\delta_{old} \Leftarrow \delta_{new}$
  $\delta_{new} \Leftarrow r^T s$
  $\beta \Leftarrow \frac{\delta_{new}}{\delta_{old}}$
  $d \Leftarrow s + \beta d$
  $i \Leftarrow i + 1$

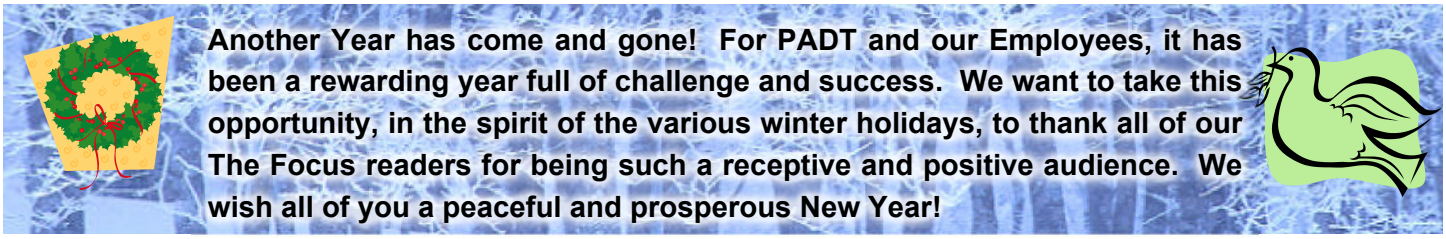
Awesome APDL: PNODIA, Harmonic Index Table

We were doing some modal runs on an impeller and the project leader asked if we could give him a table of what Nodal Diameters belonged to which Harmonic Index families. (Nodal Diameter, Harmonic Index? Read Rod's "Grok'ing the Harmonic Index" to learn more.

We cranked out this little macro to run in ANSYS to make a table based ARG1 = Number of sectors in the model and ARG2 is the number of Nodal Diameters to show for each Harmonic Index.

It shows some nice logic (use mod() to see if a number is odd, remove duplicates from an array) and more importantly how to use *VWRITE to write more than one thing on a line using a $ sign in a format. Lastly it shows how you can use *UILIST to show a text file in an interactive section.

```
nsct = arg1  !Get NumSectors
nndia = arg2  !Get Num NodalDia's to
              !  Write For each HI
ndias=        ! Clear Array

aa = mod(nsct,2)  !Use mod to calc
*if,aa,ne,0,then  ! If nsct is odd
  nhi = (nsct-1)/2 ! Or even
*else             ! To calc Number
  nhi = nsct/2    ! Of harmonic indices
*endif            ! To calc

*dim,ndias,,nhi+1,nndia*7 !Make array
*do,i,0,nhi  ! For each harmonic index
ii = i + 1
  ndias(ii,1) = i !First Nodal Dia
  ncnt = 2        ! Is HI
  *do,j,1,nndia*3 !Now do pairs
     ndias(ii,ncnt  ) = j*nsct - i
     ndias(ii,ncnt+1) = j*nsct + i
     ncnt = ncnt + 2
  *enddo
*enddo

*cfopen,nodia,txt  ! Open File
*vwrite            ! Write Header
('N O D A L   D I A M E T E R   L I S
T')
*vwrite
(' ')
*vwrite,nsct
Number of Sectors: %g %/
*vwrite
('       Unique Nodal Dia.')
*vwrite,
('| HI||',$)
*do,i,1,nndia ! Write HI Numbers
  *vwrite,i
    (F5.0,'|',$)
*enddo

*vwrite
(' ')
*vwrite       ! Write table line
('|===||',$)
*do,i,1,nndia
  *vwrite,
  ('=====|',$)
*enddo

*vwrite
(' ')
*do,i,0,nhi   ! Loop on Harm Index
  *vwrite,i
  ('|', F3.0,'||',$)
  jj = 0
  ndlst = 0
  *do,j,1,nndia*7  ! Loop on ND's
    ! If not a repeat, write
    *if,ndias(i+1,j),ne,ndlst,then
      *vlen,1
      *vwrite,ndias(i+1,j)
      (F5.0,'|',$)
      ndlst = ndias(i+1,j)
      jj = jj + 1
    *endif
    *if,jj,ge,nndia,*exit
  *enddo
  *vwrite
  (' ')
*enddo
*cfclose

! Show the File
*uilist,nodia,txt
```



Upcoming Training Classes

Month	Start	End	#	Title	Location
Dec '06	6-Dec	8-Dec	101	Introduction to ANSYS, Part 1	Irvine, CA
	11-Dec	13-Dec	104	ANSYS Workbench, Intro	Tempe, AZ
	14-Dec	14-Dec	105	ANSYS Workbench, Struc NL	Tempe AZ
	18-Dec	18-Dec	106	ANSYS WB DesignXplorer	Tempe, AZ
Jan '07	10-Jan	12-Jan	101	Introduction to ANSYS, Part 1	Tempe, AZ
	18-Jan	19-Jan	100	Engineering with FE Analysis	Tempe, AZ
	25-Jan	26-Jan	801	ANSYS Cust. With APDL	Tempe, AZ
	29-Jan	31-Jan	104	ANSYS WB Simulation, Intro	Tempe, AZ
Feb '07	1-Feb	12-Feb	301	Heat Transfer	Albq, NM
	5-Feb	5-Feb	107	ANSYS WB DesignModeler	Tempe, AZ
	7-Feb	7-Feb	411	Workbench Electromagnetics	Tempe, AZ
	12-Feb	14-Feb	101	Intro to ANSYS, Part 1	Tempe, AZ
	15-Feb	16-Feb	102	Intro to ANSYS, Part 2	Tempe, AZ
	20-Feb	21-Feb	202	Advanced Structural NL	Tempe, AZ

Links

Need Images from ANSYS, Inc. products for a presentation. Visit the press release graphics page on their News portal: www.ansys.com/special/news-images/index.htm

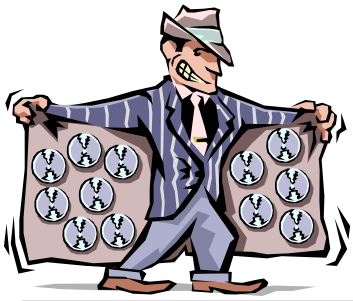
CAEA, the Channel Partner in Conn. has a great news page that features downloads from many of their seminars: www.caeai.com/index.html#Update

News

- ANSYS, Inc. Continues release of FLUENT products including FLUENT link and ICEPAK link

- ANSYS, Inc. Has Another Good Quarter and Gives Guidance on Next Year link

The Focus is a periodic publication of Phoenix Analysis & Design Technologies (PADT). Its goal is to educate and entertain the worldwide ANSYS user community. More information on this publication can be found at: http://www.padtinc.com/epubs/focus/about



The Shameless Advertising Page