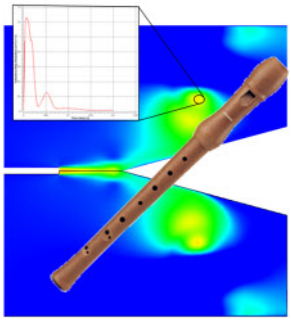
The Focus

PADT

November 9, 2007 | A Publication for ANSYS Users | Issue 60

Acoustics Modeling in CFX

By Dan Robinson



Disturbances in the flow may produce noise at unwanted levels. How can these be captured? If you've ever wondered, as many of our customers have, about the possibility of performing acoustic modeling in the ANSYS CFX software package, this article is for you.

ANSYS CFX is capable of simulating noise generated by fluid flow because it can predict the turbulence that generates noise. This is referred to as a "near field" problem. For "far field" problems where you need to simulate sound waves away from a fluid flow source, an acoustics field solver (like the one in ANSYS) is the way to go.

The following model and procedure should get users off to a good start using CFX acoustics to model near-field sound generated by flow.

Example Problem Overview

This example was developed based on some acoustic research work done in Berlin, Germany. It concerns an impinging jet that impacts a sharp edge (see Figure 1).

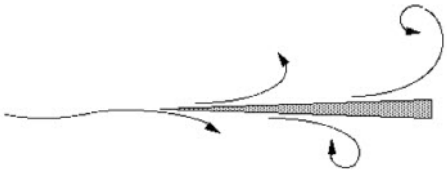


Fig. 1: Jet Impinging on a Wedge

What makes this an interesting problem is that for a jet hitting an edge:

- Instability of jet leads to oscillation on both sides of edge
- Frequency is dependent on jet to edge distance and mean exit velocity of jet
- Exit velocity profile and shape of edge drives response

This type of flow occurs in a variety of applications, and is of a nature found in many musical instruments. For instance, it is the central element of sound generation in the recorder (see Figure 2 or [WIKI]).

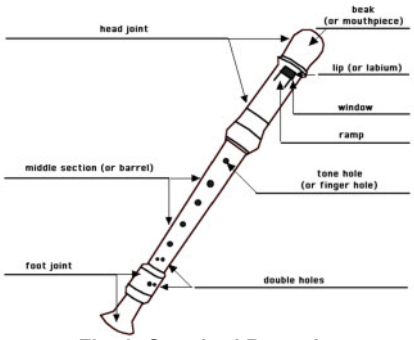


Fig. 2: Standard Recorder

(Cont. on pg. 2)

Forcing an Incore Solution

By Rod Scholl

Memory… lights the corner of my mind. Misty-water-colored memory of the way we were… When it comes to memory and ANSYS I always feel like I'm swimming with a tuna sandwich in one hand. I'm not sure what that metaphor conjures up… but I'm sticking with it.

Most memory issues are handled automatically in ANSYS. I for one part ways with the help docs – and still recommend that users usually specify –m and –db on startup (Previous Article)… But everything else memory-related, I watch with bewilderment as the memory spec's roll by after beginning a solve.

I always feel this gnawing suspicion that if I changed a single parameter or argument, everything would solve 2X faster… The latest source of suspicion was reading about the BCSOPTION and DSPOPTION argument to force an in-core solution. There's been a little talk about this lately because the default behavior on the distributed solver changed in 10.0 vs. 11.0. In version 10.0, the default method was to run incore – however, in 11.0 the default behavior for the distributed solver is to allow the memory manager to determine allocation. But let's back up a bit:

When it comes to the Sparse Solver ANSYS requires the sparse solver memory to be contiguous. This is the same as continuous but with a 'g'. Gene Poole discusses this clearly in a past article.

This is a tired subject for most windows users as we struggle to accept that a 2GB machine can usually only grab 1000 mb to 1400 mb maximum, with the typical around 1300 MB. On a 4GB machine employing the /3gb switch, one can get up to 3GB total

(Cont. on pg. 3)

Table of Contents

(Acoustics, cont...)

The basic geometry for our example problem appears in Figure 3, where a 30° wedge included angle was used with a nozzle jet width of 1 mm.

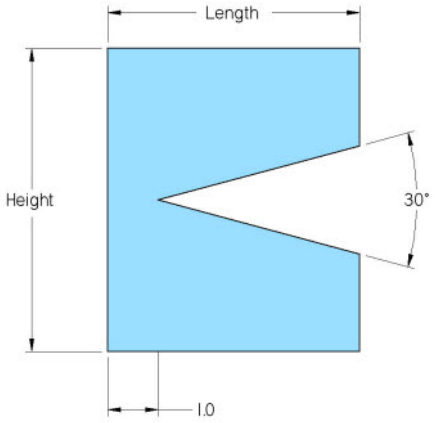


Fig. 3: Model Geometry

Figure 4 shows the model as setup in CFX. A structured mesh (approx. 230,000 nodes) was created in ICEM CFD and is shown in Figure 5. Note that CFX is a 3D code so the geometry was modeled as 1mm thick and one element through the thickness. Also note that the problem is symmetric so only the top half is modeled.

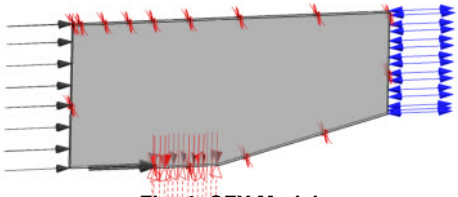


Fig. 4: CFX Model

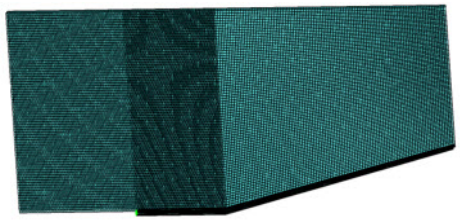


Fig. 5: Mesh

The model boundary conditions were:

- Air @ 25C, incompressible
- Jet with exit velocity U = 12.712 m/sec
- Re = 800 (*Ud /v*)
- Nozzle width: d = 1mm
- At left boundary inlet with 1% U

General Procedure – Monopole Sources

Proudman's formula is an equation for the acoustic power which can be written in terms of the turbulent quantities (Ref. 1). These quantities are available in CFX and may be extracted from the CFD simulation.

The general approach for monopole sources then is to follow this process:

1: Select monitor points in flow near regions of potential noise sources (see Figure 6).
2: Perform a transient simulation to capture unsteady velocity, pressure fluctuations.
3: Use appropriate turbulence model, usually a RANS, DES, or LES simulation.
4: Write out transient data in .csv file format. Go to File-> Export-> filename.csv. Select variables Turbulent Kinetic Energy, Turbulent Eddy Dissipation for the specified monitor points.
5: Use the Proudman's formula equations which give acoustic power, sound pressure level (SPL), and sound intensity in terms of turbulent quantities.
6: Import the data into MS EXCEL.
7: Perform Fourier analysis in EXCEL. Then plot SPL vs. frequency, intensity, acoustic power, etc.

This example was done using a RANS simulation (Reynolds Averaged Navier-Stokes), using the BSL Reynolds Stress turbulence model. Because of the unsteady nature of the problem a transient solution was performed.

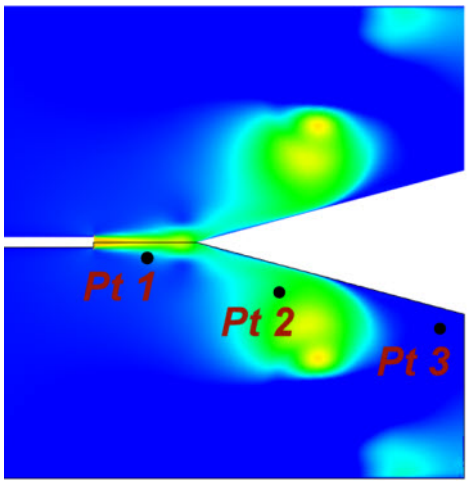


Fig. 6: Monitor Points

Governing Equations

Proudman's Formula: Acoustic power due to unit volume of isotropic disturbance

$$P_A = \alpha\ \rho_0\ (u\text{^}3/\ l)(u\text{^}5/\ a_0\text{^}5)$$

where u and l are turbulent velocity and length scales, a_0 = speed of sound.

Based on turbulent KE and Eddy Dissipation:

$$P_A = \alpha_\varepsilon\ \rho_0\ \varepsilon\ M_t\text{^}5)$$

$$\text{where } M_t = \text{sqrt}(2k)\ /\ a_0$$

Sound Pressure Level

$$SPL\ [in\ decibels] = 10\ \log\ (\ P_A\ /\ P_{ref}\ )$$

$$\text{where } P_{ref} = 10^{-12}\ W/m^3$$

Sound Intensity:

$$I = P_A\text{^}2\ /\ 2\ \rho_0\ a_0$$

Note: These equations may be entered as CEL (CFX Expression Language) expressions for output directly, or they may be extracted from the turbulence quantities within MS EXCEL later.

Model Results

The progression of the velocity oscillations within the fluid domain is shown for various timepoints in Figure 7 on the next page.

Figure 8 shows the transient results for turbulence quantities which were extracted in CFX-Post and plotted.

The final sound pressure levels plotted vs. frequency are shown in Figure 9 for point 3. The example results are near field for monopole sources. Far-field simulations must export the unsteady, transient data to third party software such as LMS SysNoise or Virtual Lab Acoustics or your own software to evaluate.

Near-field Dipole surface and Quadrapole sources may also be modeled using CEL expressions to monitor the fluid pressure fluctuations and wall shear.

Far-field solves the Ffowcs-Williams-Hawking equation:

References

1. I. Proudman, "The Generation of Noise by Isotropic Turbulence", Proc. Roy. Soc. London, A214, 119 (1952).

(Acoustics, cont...)

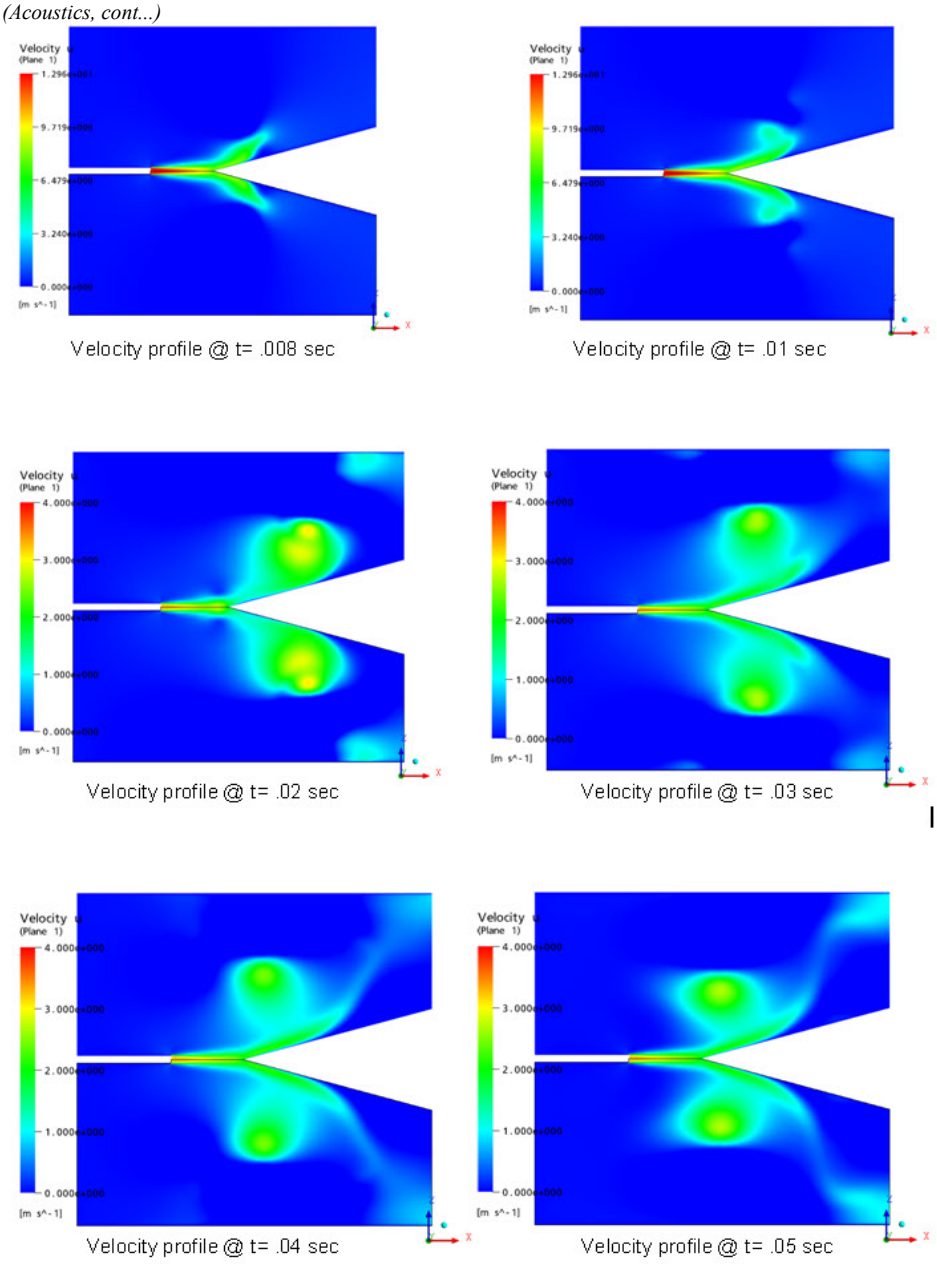


Fig. 7: Transient Velocity Profiles

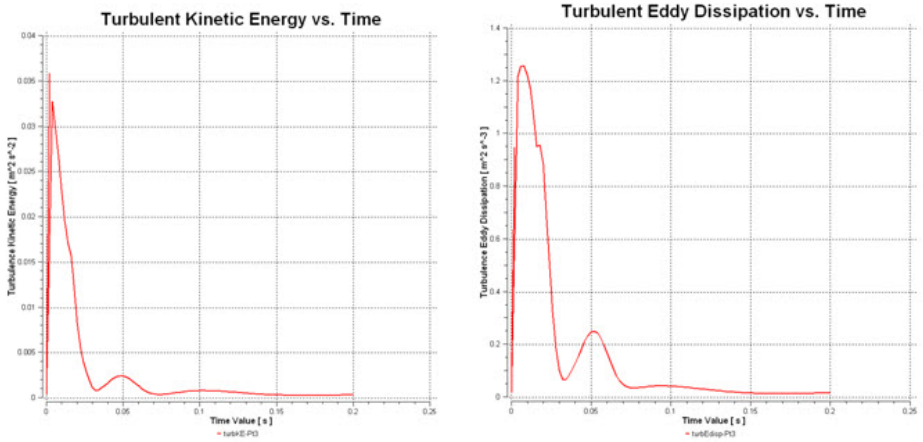


Fig. 8: Turbulence Results for Point 3

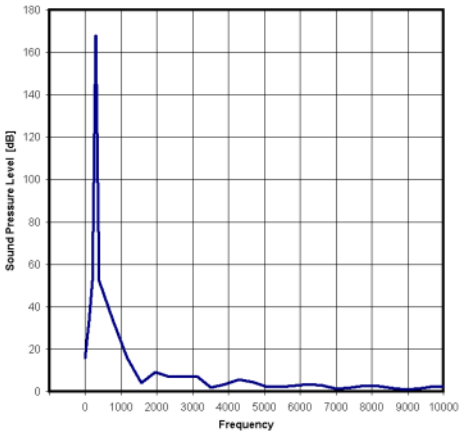


Fig. 9: Flow Induced Noise Level - Point 3

ANSYS Advantage Magazine

ANSYS Solutions magazine is now called ANSYS Advantage. For those Fluent users out there, Fluent News is also rolled up into this new name, ANSYS Advantage. The first two issues came out in 2007 and are laden with case studies. Seeing these real world applications is helpful in keeping up with the ever-expanding suite of ANSYS products across many industries and physics. Plus its online and it's free!

Virtual Memory

Let's push some of my prior confusion out of the way. In most OS's there is the ability to use virtual memory. Thus in the case of windows, a file is used on the hard disk as though it is RAM and info is passed back and forth. This is of course much slower than real RAM but an alternative to bring large chunks of data into an active workspace. For us ANSYS folk, it is best to take virtual memory off the table. Since ANSYS has its own methods for handling cases where memory isn't sufficient, such as the .page file, and some of the LNxx files. When compared head-to-head, ANSYS does better with the .page and LNxx files – than the OS's attempt at virtual memory. This makes sense because I/O is clearly the bottleneck and the OS folk didn't plan on ANSYS-style data being read and written, while ANSYS, Inc. did -

(Incore, cont...)

address space theoretically – but I've only heard anecdotally of people getting to 2400 MB, and I think typical is more like 1700 MB of space – But it is not a contiguous block since the extra GB of memory is a separate block in Win32 This is all because of the way windows slices up the memory with .dlls' and who knows what else. For a while the practice of "rebasing the .dlls" was explored but my experiences was that it was dangerous in terms of system stability, usually only got a hundred or so MB's more memory, and generally is discouraged as not worth the hassle. On Linux/Unix, one can usually grab the majority of the RAM available so feel free to laugh at us who struggle along using Win32. Anyway this is all background well covered here.

Although it was the Distributed Solver memory settings which began this investigation, we'll save that for a future more comprehensive article on all things DANSYS. Instead, let's focus on the BCSOPTION command, which parallels the distributed solvers DSPOPTION. I'd been told that if on some analyses that one could get a dramatic increase in solve time by specifying BCSOPTION,,INCORE compared to allowing the default setting of BCSOPTION,,DEFAULT. So I set out to determine what class of analyses and memory scenarios for which this is true.

Job size much smaller than available memory:

For jobs that fit easily within memory, the BCSOPTION,,DEFAULT is the clear way to go. Solve times will be dang close to using the setting of BCSOPTION,,INCORE – and so if it isn't broken, leave it alone. The reason for this improvement in I/O is that both WinXP64 and Linux systems now have an automatic system buffer cache feature that hides a lot of your I/O cost by keeping the files in memory. This works great as long as the sum of the files open plus the memory ANSYS is using are within the limits of your physical memory. In-core performance for the sparse solver will still beat out-of-core buffered I/O in these cases but not by much.

Job size much greater than available memory:

Here you want to also use the BCSOPTION,,DEFAULT. If you instead specify BCSOPTION,,INCORE the solver initially allocates a larger block than the default option would and there is likely a second subsequent memory allocation required as soon as preprocessing of the matrix determines the exact size required to run incore. There is always a certain amount of wasted memory in this approach. If the incore memory allocation fails ANSYS is smart enough to revert to optimal out-of core memory but the damage is already done in many cases, and the user can expect longer solve times than would have been had using BCSOPTION,,DEFAULT. Also, if one specified BCSOPTION,,INCORE on a job size much greater than available memory, the extra allocation of memory left unused in the ANSYS scratch space is also not available for the system buffer cache. Thus a run that may have run efficiently using the default smaller memory allocations will now use slow disk I./O sooner than if the system buffer cache had enough memory to hide the true cost of the I/O.

Job size barely fits into memory:

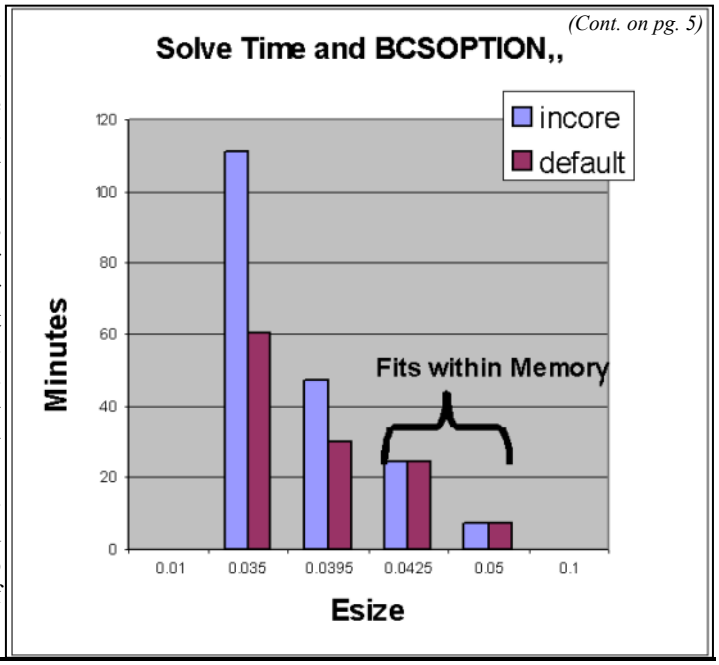
While trying to get cleared up on this, and talking with Gene Poole (one of the ANSYS experts to which we all owe the increased performance we get with every release of ANSYS) I was confused about under what conditions a BCSOPTION,,INCORE would beat the default BCSOPTION,,DEFAULT. Not having a clear picture after half an hour of conversation I asked, "Can you design for me a class of problems that is sure to have better performance by specifying INCORE". He said, "Well if I could design that problem I would have already had the default setting's memory manager fix it!" Aha…

So we're going to actually need some knowledge to understand this scenario, what I've learned from Gene is this:

There is another factor which can impede the performance of the incore option; there is a design feature of ANSYS that allows users to do multiple loads without refactoring the matrix each time. This results in much faster times for additional loads but it requires ANSYS to save the matrix factor. The sparse solver workspace is saved automatically in file.LN22 in anticipation of those users who might have additional loads (that don't change boundary conditions). As long as the matrix factor itself is in a file the work array is not huge.

But, in the case of an BCSOPT,,INCORE factorization the sparse solver work array contains the entire factored matrix. So writing the file.LN22 becomes expensive and

(Cont. on pg. 5)



(Incore, cont...)

negates some of the benefit of running incore. So, why don't the ANSYS developers fix this? Actually there is an undocumented way to avoid writing the LN22 file. The following BCSOPT command will run an incore factorization AND suppress the writing of the LN22 file:
BCSOPT,,INCORE,,-1

The -1 argument in the fourth field holds onto the sparse solver work array in memory and therefore the LN22 file is not written. Actually, a smaller LN22 file is still written but it does not contain the factored matrix file. But remember that if you do not release the sparse solver memory there is no way ANSYS can reuse that memory in subsequent operations. This can make long running jobs fail due to insufficient memory. Also note, this LN22 file behavior is only seen with static analyses. For nonlinear jobs where the matrix is always refactored the LN22 file only contains information about matrix structure and is much smaller.

Understanding I/O's Impact

To take advantage of the INCORE option, you must understand the I/O hierarchy in current high end systems. The best processors today are computing up to 5 billion operations per second per core in matrix factorization. That speed is obtainable on a desktop as well as an expensive server. A single disk drive runs at 30 Mb/sec up to perhaps 70 or 80 Mb/sec. Since every double precision word is 8 bytes this means most single disk systems can read less than 10 Mwords per second. That is a performance disparity of 500! Using the system cache approach to hide I/O the effective I/O rate for out-of-core solves jumps to 500 – 700 Mb/sec, a 10-fold increase in performance. Users can also achieve 200 – 300 Mb/sec now on desktop systems using multiple RAID0 drives. But, the system buffer cache does require bookkeeping and multiple system memory copies of data so the most impressive I/O performance is reserved for true incore solves. Starting in version 11.0 of ANSYS the effective I/O rate in the sparse solver is printed out in file.BCS or in the output file. It shows that incore performance on today's fastest desktops will usually exceed 2000 Mb/sec and rates as high as 7000 Mb/sec have been observed on some large memory systems. Obviously, if an ANSYS run was dominated by sparse solver I/O incore performance could reduce solution time by the ratio of 2000+/30 in some cases. But I./O is only part of the cost. For Block Lanczos runs I/O time will often exceed factorization times. But, for most static analyses the solve times are much less than factorization times so it is only important to avoid the huge 500X imbalance of slow disks to get very good balanced performance. Cheap but effective RAID0 arrays, large memory and incore when applicable are all ways to drive down the I./O cost.

SAMPLE BCSOPTION,,,,,,-5 output

```
End of PcgEnd
     ANSYS largest memory block available     1064518304 :  1015.20 Mbytes
                    ANSYS memory in use        35592512 :    33.94 Mbytes
                 ANSYS max memory in use       156031408 :   148.80 Mbytes
  Total Time (sec) for Sparse Assembly      1.11 cpu     8.09 wall
Incore factorization
            input matrix    11802411    90.05
           matrix factor  256446204.  1956.53
               max stack    52975368   404.17
       panel and update       720288     5.50
                   misc      1927291    14.70
 ------------------------------------------
            Total incore  271205973.  2069.14
============================
 = multifrontal statistics =
 ============================
     number of equations                     =          206754
     no. of nonzeroes in lower triangle of a =         4994481
     number of compressed nodes              =           68918
     no. of compressed nonzeroes in l. tri.  =          851280
     amount of workspace currently in use    =         1683777
     max. amt. of workspace used             =        63307754
time (cpu & wall) for structure input   =        1.421875        1.544337
     time (cpu & wall) for ordering          =        5.218750        5.490375
     time (cpu & wall) for symbolic factor   =        0.109375        0.149819
     time (cpu & wall) for value input       =        1.703125        9.779859
     time (cpu & wall) for numeric factor    =      306.156250      368.527348
```

125K DOF Limit:

If you have more than 125,000 DOF (equations) an internal switch means jobs of this size or smaller are going to perform almost identically using the INCORE or DEFAULT option on BCSOPTION, assuming it fits within core memory. For 125,000 DOF, this threshold is around 1GB of RAM. Most of us have at least that much, but if you have less, then you definitely won't have much benefit exploring the INCORE option.

Bottom Line

The sweet spot for using incore is to specify the incore option (with the secret undocumented flag from above) whenever a job can run comfortably within the available physical memory. The benefit of incore can be 25 or 30 percent for a Block Lanczos run and 15 or 20 percent for a static analysis run with multiple loads. For a single load static analysis incore versus optimal out of core is hardly noticeable. If this is not your scenario then the ANSYS default memory allocation for the sparse solver will probably deliver optimum memory configurations and performance.

Lastly, I'll repeat what Gene Poole wrote in an earlier article , become familiar with the output resulting from the command BCSOPTION,,,,,,PERFORMANCE (this is also BCSOPTION followed by 6 commas and a –5 if that's easier to remember.) This is the sort of command I put in my startup file. I won't learn much at first… but after a while of staring at the output window on a big job, I get curious and actually learn a thing or two … The comprehensive output is the sort of thing experts use to assess memory requirements. Perhaps if I ever understand any of it, we'll write a future article. There are pages of it… but I grabbed a few sections shown below to whet your appetite:

And a final statement negating the value of this article entirely: By the time you learn a class of problems / hardware configuration which makes the default settings not ideal, don't be surprised if ANSYS, Inc. has figured it out and incorporates the intelligence into the next release's memory manager, thereby making BCSOPTION,,DEFAULT once again sufficient for that case.

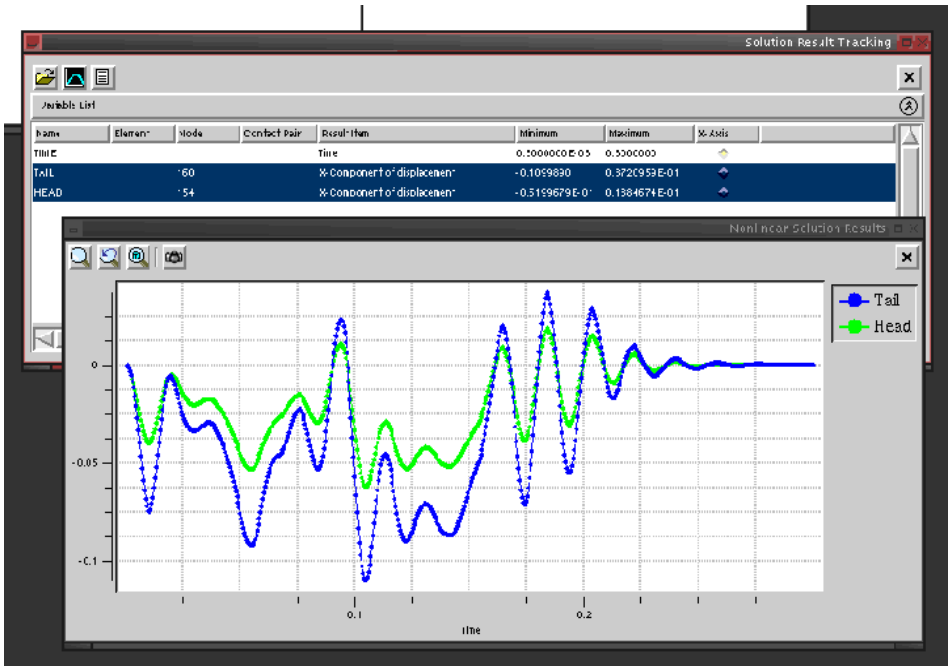
NLHIST: Remembering a Forgotten Friend

By Eric Miller

While preparing the test cases for the article on transient dynamic CMS analysis, I got a little frustrated trying to decipher how my run was doing from looking at a tail –f on the output file. I said to myself "Self, wouldn't it be great if I could plot the deflection and stress on a couple of nodes while it solved?" As is usual, my self responded with a rude insinuation about our mutual lineage and pointed out "Remember NLHIST, idiot!"

NLHIST is the key command in a suite of tools grouped as "Nonlinear Diagnostics" It basically allows the user to specify some result or status value at a node or element, and write it to a file for every substep. For interactive runs, it is displayed in the ANSYS graphics window. But we recommend that you use the nlhist110 program to display it in a more modern GUI. To view it while solving, use the nlhist110, point to the file you want to view and track away. On Linux/Unix just type nlhist110. In windows you can get to it from the ANSYS Launcher in the Tools menu under "Run Results Tracker Utility." Over the years we have found that it often tells you when your model is starting to blow up before ANSYS actually generates an error, something that can save some of time.

For the plot shown in the image to the right, I used NLHIST,NSOL, Tail, UX, 160 and NLHIST, NSOL, Head, UX, 154. This tells ANSYS to make a file called jobname.nlh. The info you asked for is stored in XML format within that file.



Once it has failed, you can use the other commands grouped with it like NLDIAG (previous article) and NLDPOST to view where the problems are in your model. These are fantastic tools that can save hours if not days of model debugging.

Learn more in the manual, section 8.12.2.1 or lookup the NLHIST command.

News - Events

- ANSYS Inc. Teams up with the Sinda/G folks integrating the abilities into ANSYS Workbench.
- ANSYS & HPC Partner Luncheon at Supercomputing '07. Reno, NV on 11/14/2007
- 2007 ANSYS Power Generation Conference in New Orleans, LA, 12/10/2007. This conference provides an opportunity to network with other ANSYS users in the power industry and to learn about how your peers are leveraging ANSYS FEA, CFD, and multiphysics software to optimize new technologies, perform safety analyses, and ensure retrofit success.

11.0 Service Pack Released

- You will definitely want to upgrade to the Service Pack 1 version of ANSYS 11.0. It includes support for Vista, error corrections and more.
- Note that you must uninstall 11.0 before reinstalling the service pack 1 version.

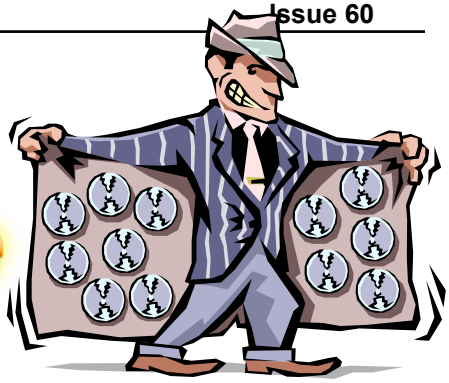
Login to your Customer Portal and download it.

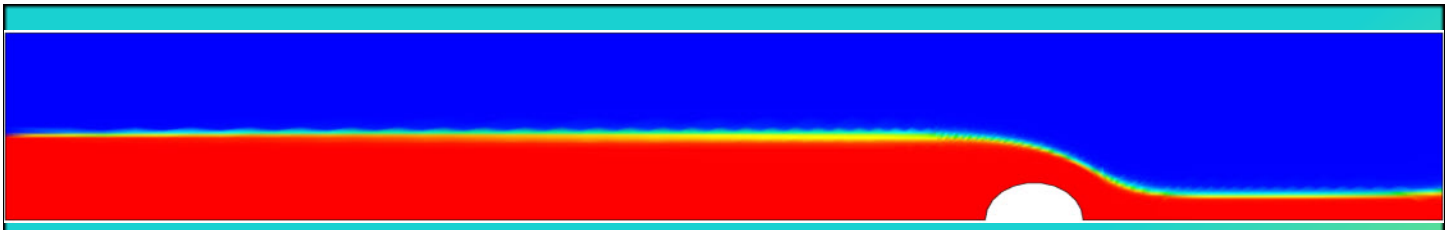
Upcoming Training Classes

Month	Start	End	#	Title	Location
Nov '07	11/12	11/13	301	Heat Transfer	Tempe, AZ
	11/15	11/16	102	Intro to ANSYS, Part II	Tempe, AZ
	11/27	11/28	204	Advanced Contact and Fasteners	Tempe, AZ
	11/29	11/30	604	Introduction to CFX	Tempe, AZ
Dec '07	12/06	12/07	302	ANSYS WB Sim. Heat Transfer	Tempe, AZ
	12/10	12/11	104	ANSYS Workbench Sim. Intro	Las Vegas
Jan '08	1/14	1/16	101	Intro to ANSYS, Part I	Tempe, AZ
	1/17	1/18	100	Engineering with FEA	Tempe, AZ

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



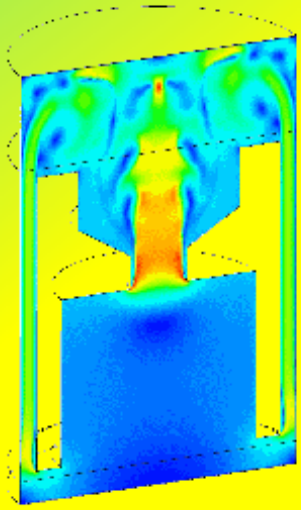


GOT CFD?

Whether you feel you need to start implementing CFD into your engineering analysis or you are looking for increased compute power, PADT, Inc. can assist you! We use two of the best and most comprehensive CFD tools available, FLUENT and CFX, which are unmatched in their breadth and depth of capability when solving the toughest or even the simplest CFD problems.

PADT, Inc. has CFD engineers with over 15 years of experience using FLUENT, CFX and a host of other CFD codes. This experience enables us to quickly assess an application, understand the challenges and provide you with timely, accurate and detailed results. Give us a call or send us an email if you:

- **Want to bring CFD into your engineering design and analysis and don't currently have the expertise?**
- **Don't have the compute power to solve larger CFD problems?**
- **Have purchased CFX or FLUENT and want some help, such as mentoring or services to get up to speed more quickly?**
- **Need a CFD job done now!?**
- **Or just need additional CFD resources...**



To speak with someone about your CFD and other engineering needs, please contact Stephen Hendry, visit us online, or call 1-800-293-PADT (7238).