Regional Summit 2008 NAFEMS

2020 Vision of Engineering Analysis and Simulation

October 29 - 31, 2008 | Hampton, Virginia

CASE STUDY: HOW TO MAKE AN ANALYSIS INTERFACE THAT BOTH THE NOVICE AND THE EXPERT WILL USE

Tim Cunningham (Micro Motion, Inc.)

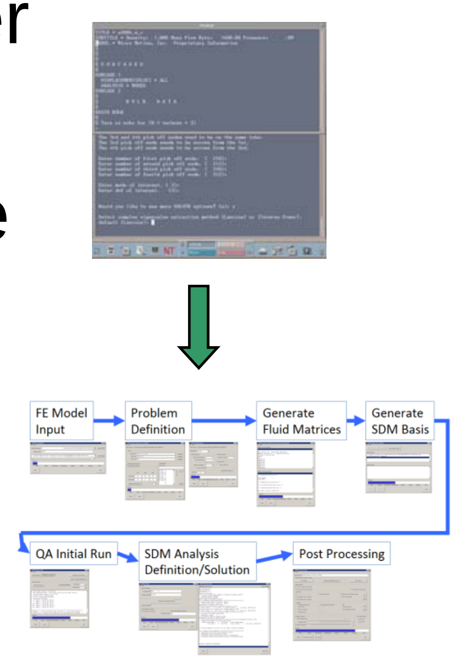
Dan Hensley (ATA Engineering, Inc.)

Overview

- Axiom: Users will not switch to a new tool unless it is SIGNIFICANTLY better than their current tool
- FE tools historically have targeted one niche

  Powerful OR Easy-to-use
- Expectation is now that FE is both

  Powerful AND Easy-to-use
- To be significantly better it must also offer significantly more features



ATA Engineering Has More Than 25 Years Experience Providing High-Value Engineering Services

ATA ENGINEERING, INC.

Micro Motion is the leading Coriolis flowmeter manufacturer

- MMI started in 1977 in Boulder, CO and received a patent for the first modern Coriolis flowmeter
- Now a $500 million company; part of Emerson
- MMI has been collaborating with ATA since 1989 on flowmeter analysis and testing and tool development

Micro Motion

EMERSON
Process Management

What is a Coriolis mass flowmeter?

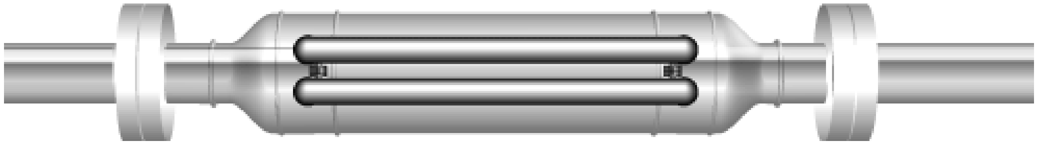
- Coriolis flowmeters give a direct reading of mass flowrate for any fluid, liquids and gases
  - – Not volumetric flow or inferred mass flow
- High accuracy mass flow measurement is valued by many industries
  - – Chemical
  - – Oil & gas
  - – Food & beverage
  - – Life Sciences

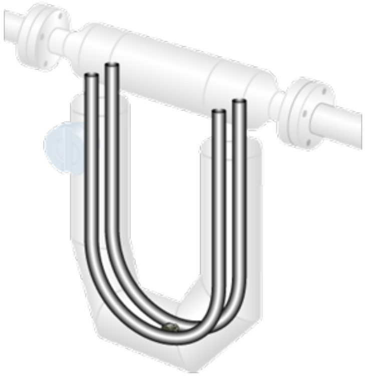
Pharmaceutical

Natural Gas Cavern Storage

How does a Coriolis mass flowmeter work?

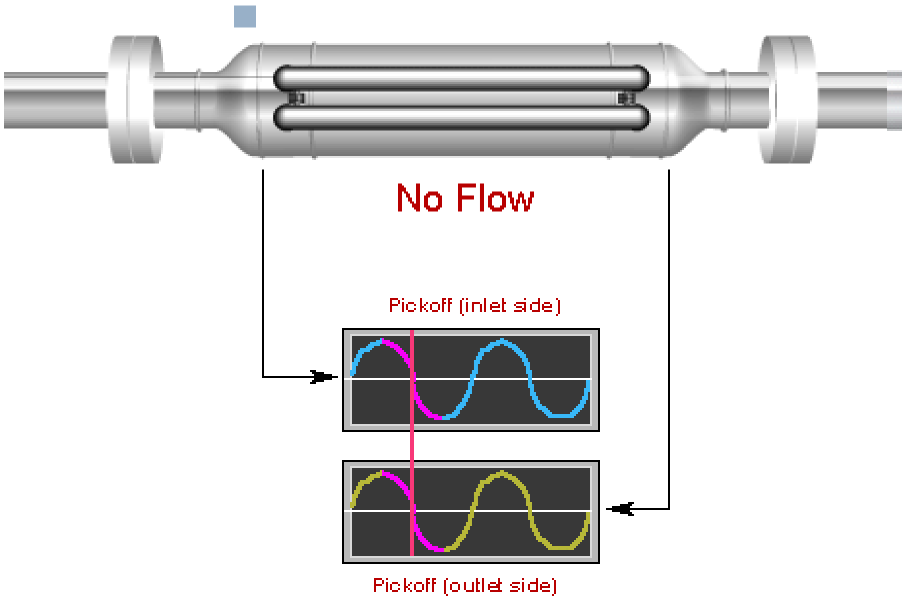
- Process fluid enters the sensor and flow is split with half the flow through each tube
- Tubes are oscillated at the first out-of-phase resonance

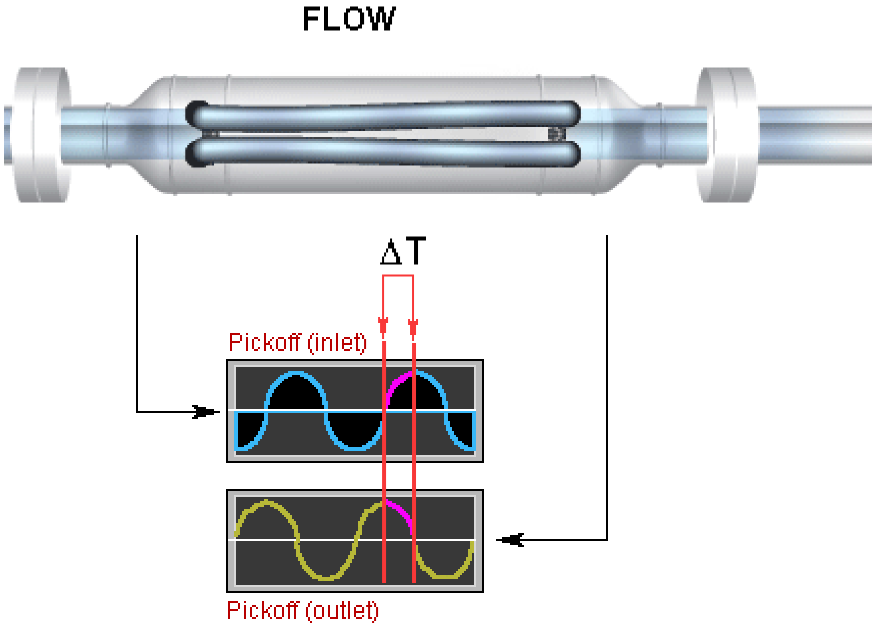




How does it measure mass flow?

- During a no flow condition, there is no Coriolis effect and the pickoff signals are in phase with each other.
- When fluid is moving through the sensor's tubes, Coriolis forces are induced causing the flow tubes to twist in opposition to each other. This Coriolis motion is superimposed on normal resonance and results in a phase time difference between the pickoff signals. This time delay, δt, is directly proportional to the mass flow rate.





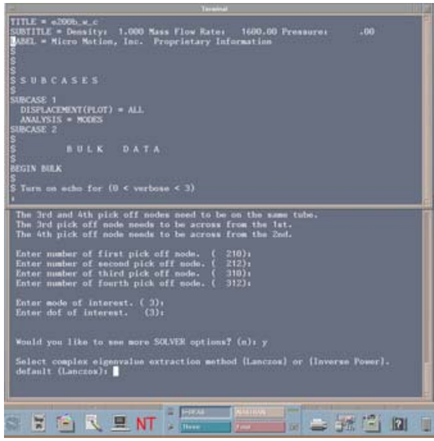
Dynamic FEA is used to predict flowmeter performance

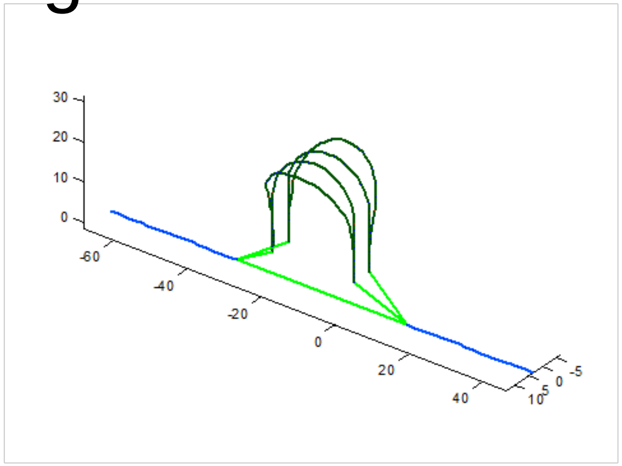
$$M\ddot{x} + C\dot{x} + Kx = f$$

- *C*, the Coriolis matrix, is the coefficient for the velocity term $\dot{x}$
- Proprietary FORTRAN code generates this matrix
- Analyzing flowmeter performance requires a whole suite of dynamic analyses
  - normal modes
  - complex modes
  - frequency domain forced response

Initially the FEA was fairly simple

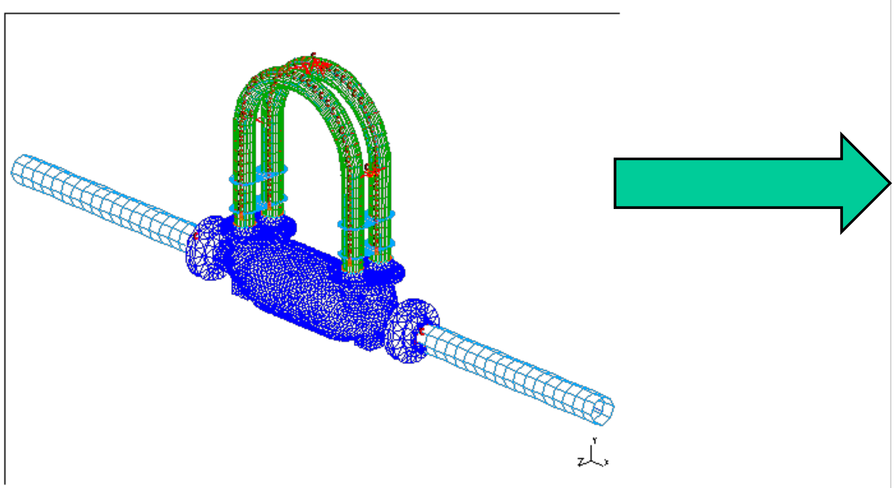
- Original utility was Unix command-line FORTRAN-based tool called MMIFEM
  - Proprietary fluid matrix generation
  - Limited analysis capabilities
  - Text file output to Excel
- As models became larger their solution time became too long for interactive use

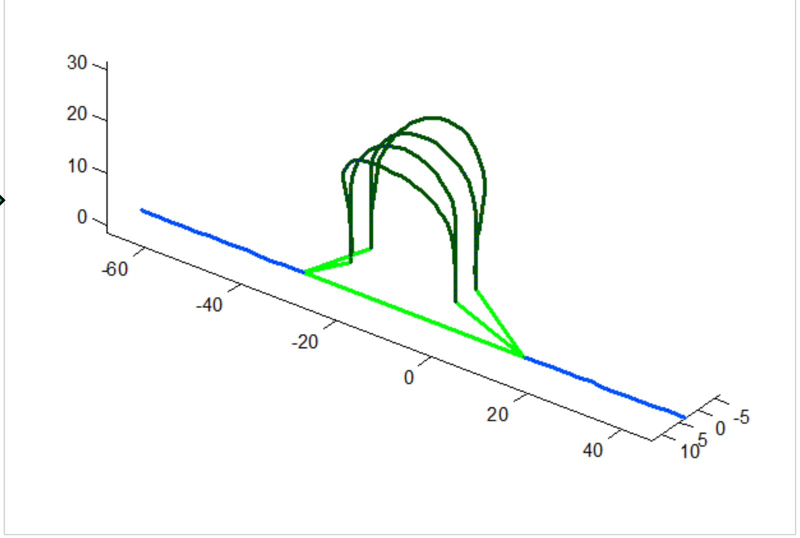




Structural Dynamic Modification improved our capabilities

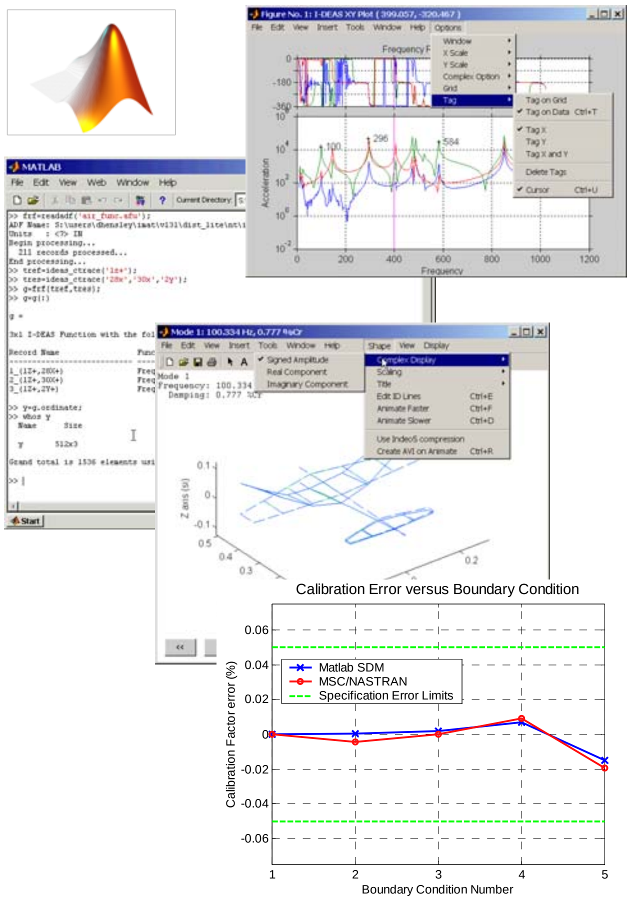
- Modal reduction dramatically reduces model size
- SDM greatly reduces solution time without significantly sacrificing accuracy
- Needed a platform that could do the math as well as integrate with FE software





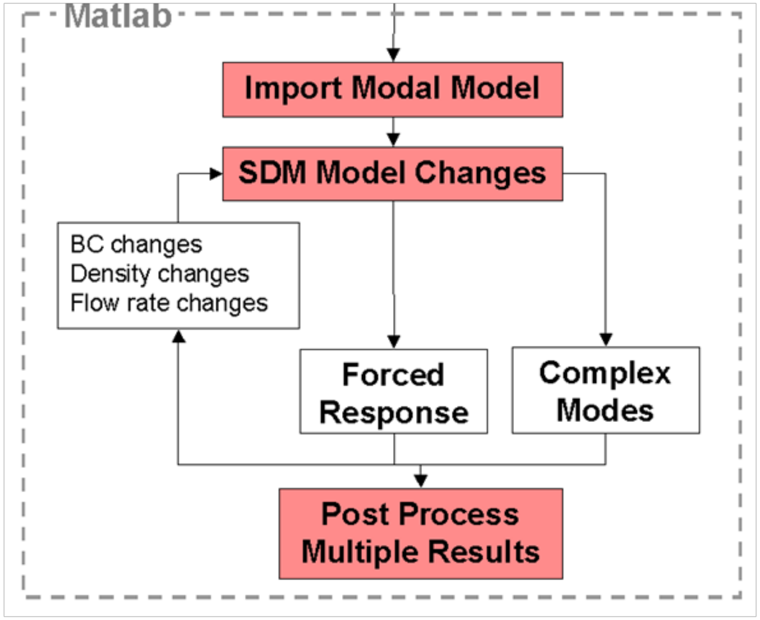
MMI & ATA first implemented SDM tools in MATLAB

- MMI and ATA were collaborating on FE analyses
- MMI was using MATLAB for SDM proof of concept
- ATA had a MATLAB toolbox for modal & FE analysis
- MATLAB was a natural choice
  - Accurate and efficient
  - Robust graphics
  - GUI development
  - Ability to use legacy code ported from UNIX to Windows



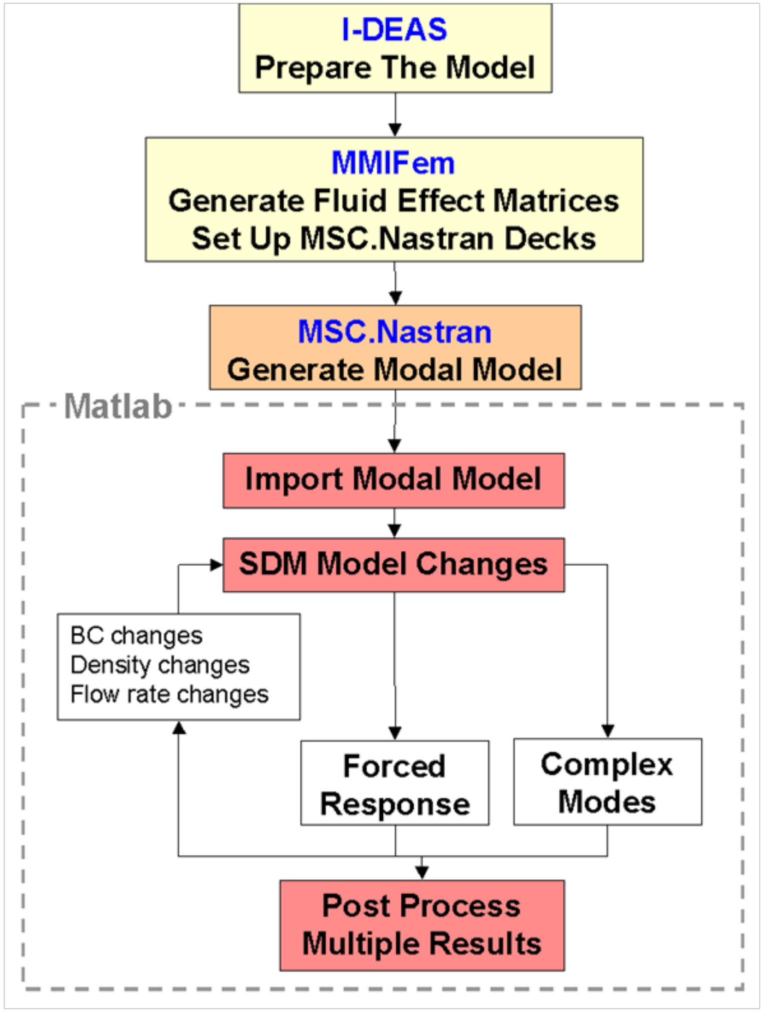
First SDM implementation was very promising

- Developed a set of command line utilities and simple GUIs to perform the various analysis steps
- Users could now interactively analyze many iterations quickly even from large models
- SDM lived up to its promise of speed and accuracy



SDM implementation was powerful but complex

- Expertise with lots of different software required
- Users need to understand intricacies and limitations of SDM
- Many tedious and exacting steps
- Lots of opportunity for error!



Initial attempt to provide SDM analyses to everyone was not successful

- SDM was much more powerful than legacy tools but was confusing for novice users
- It was not easy enough for novices AND
  Not enough more powerful for the experts to want to switch
- ***How do we satisfy all of the users?***

We had to make the tools SIGNIFICANTLY better in MANY ways

- Experts and novices both appreciated a new easy-to-use GUI that completely integrated the analysis process
- Retained all of the power of SDM, and still provided the advanced capabilities for experts
- Added many new features that were unavailable in the legacy tools

How we developed the new tools

- Leveraged software development techniques
  - Structure your data
  - Build the software in layers
  - Automate code validation with a comprehensive test suite
  - Use a version control system
- Solicited lots of user feedback on:
  - What they perceived as easy-to-use
  - What new features they wanted
  - What advanced features the power users needed

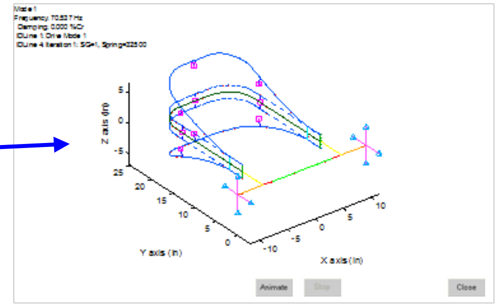
Good data structures are foundational

- Define overarching data structures that contain all of the desired analysis inputs and results for a given task
- Group related functionality and identify common data
- Select the appropriate data storage type(s)
  - IMAT provided robust and functional data types for functions, shapes, and FEM entities
- Use this data structure in the underlying functions

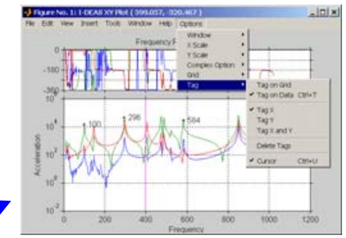
Good data structure example

```
mmi_model =

         fem: [1x1 struct]
         phi: [100x1 ideas_shp]
        phic: [1x1 ideas_shp]
          fr: [1x1 struct]
      ctrace: {[960x1 ideas_ctrace]}
      femchg: [1x1 struct]
          mr: [100x100 double]
          kr: [100x100 double]
        nsmr: [100x100 double]
       zetar: [100x100 double]
        corr: [100x100 double]
          br: [100x100 double]
         dmr: [100x100 double]
         dkr: [100x100 double]
       centr: [100x100 double]
       presr: [100x100 double]
         dof: [960x1 ideas_ctrace]
           m: []
           k: []
         nsm: []
         cor: []
           b: []
          dm: []
          dk: [1x1 struct]
        cent: []
        pres: []
         dpo: [1x1 struct]
     numattr: [10x1 double]
     strattr: {1x10 cell}
    userdata: []
        desc: {'cmf025 (cmf025)'}
     version: 6
```



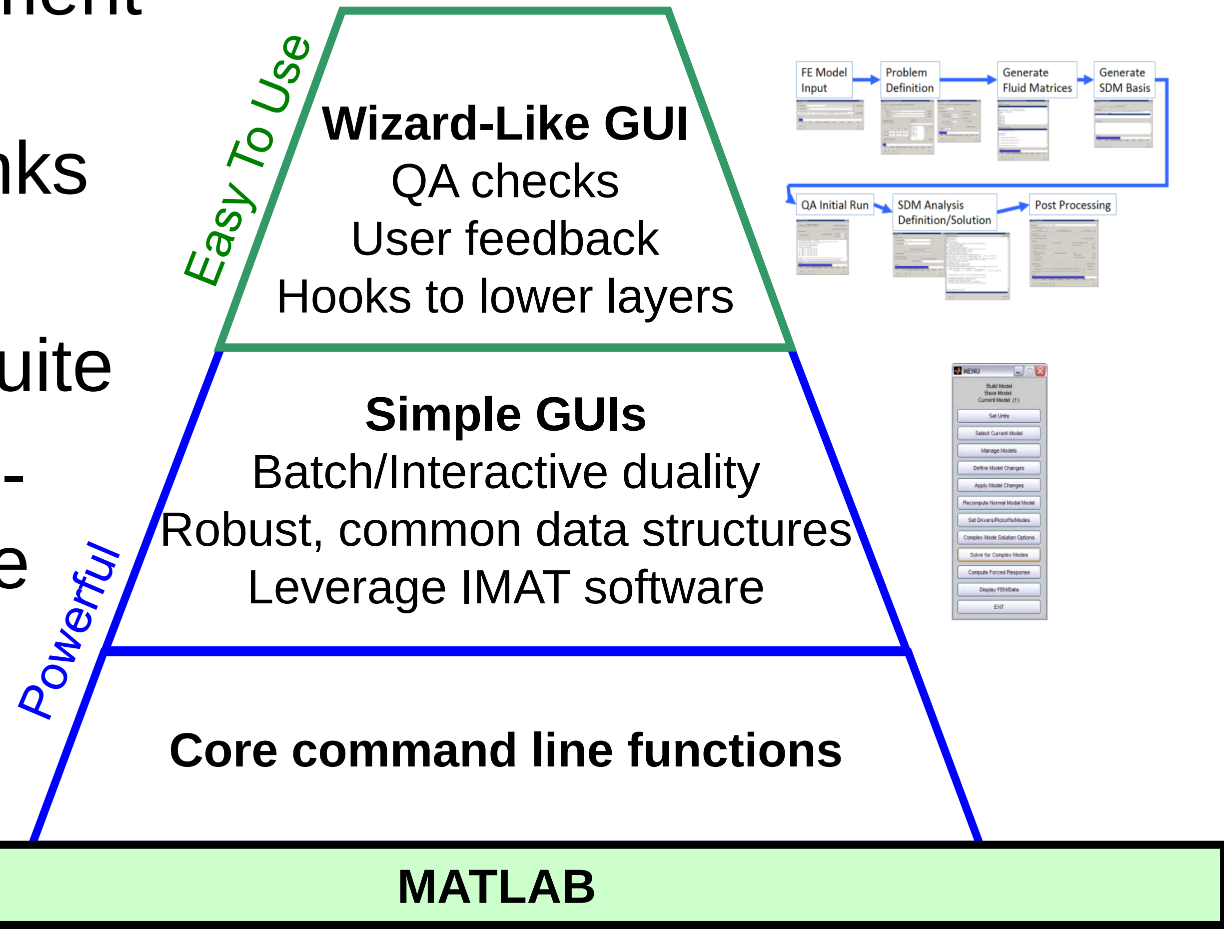
```
    freqvec: -1
 driveshape: [5x3 ideas_shp]
        frf: [5x1 ideas_fn]
```



- `mmi_model` structure is our foundation
  - Contains all of the analysis data, FEM geometry, results, user input
  - Store only baseline matrices and deltas
  - Use IMAT datatypes for FEM and mode shapes and forced response output

Building Software In Layers provides maximum flexibility

- Breaks development effort up into reasonable chunks
- Easy to build automated testsuite
- Lower level func-tionality available from higher layers



The first layer is the core functionality

- Command-line functions that perform specific tasks
- Input and output parameters are subsets of the core data structures
- Make functions as reusable as possible

```
function [phinormed] = massnorm(phiin,mass);
 pass this function modal matrix and mass matrix
 and it returns mass normalized modal matrix
```

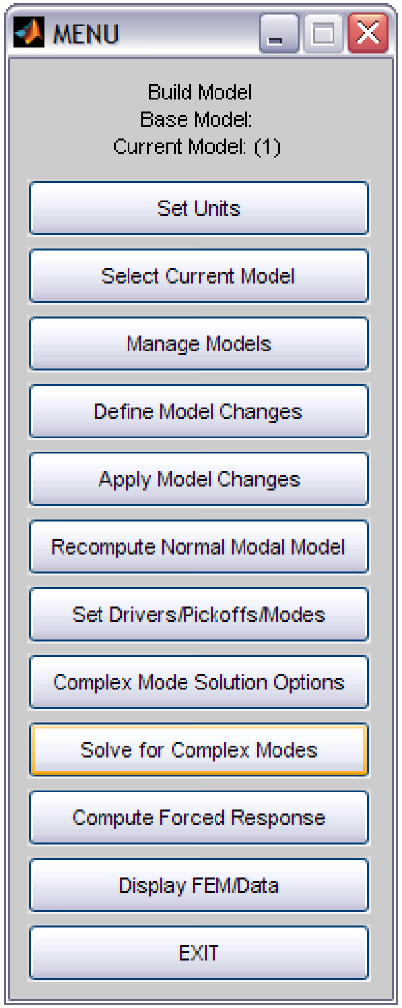
```
dt=CALC_DT(compmode,ponode1[,ponode2])
   CALC_DT calculates DT vectors, frequency vectors, phase vectors, and
   drive frequency vectors.  COMPMODE is an MxN ideas_shp, which contains
   the drive modes for the given mmi_model iteration M and drive mode N.
   PONODE1 is an ideas_ctrace containing pickoff DOF set 1, and PONODE2 is
   an ideas_ctrace containing pickoff DOF set 2.  For a U-tube meter, set 1
   is the pickoff nodes on one tube, and set 2 is the corresponding pickoff
   nodes on the other tube.  For a straight tube meter, set 1 and 2
   correspond to tube and case, respectively.  If PONODE2 is not supplied,
   CALC_DT will assume zero displacements in the calculation.

   DT is a structure with the fields
     .dtvec     = dT vectors
     .phvec     = Relative phase
     .drivefreq = Frequencies for each shape from which dT was calculated

   See also mmi_model, buildmodel, postmodel, parse_pickoff_shapes
```

The middle layer provides the bridge

- Wrapped simple GUIs around the related command-line functions
- Interactive/batch duality
  - The glue that ties the layers together
  - Critical for developing automated test suites

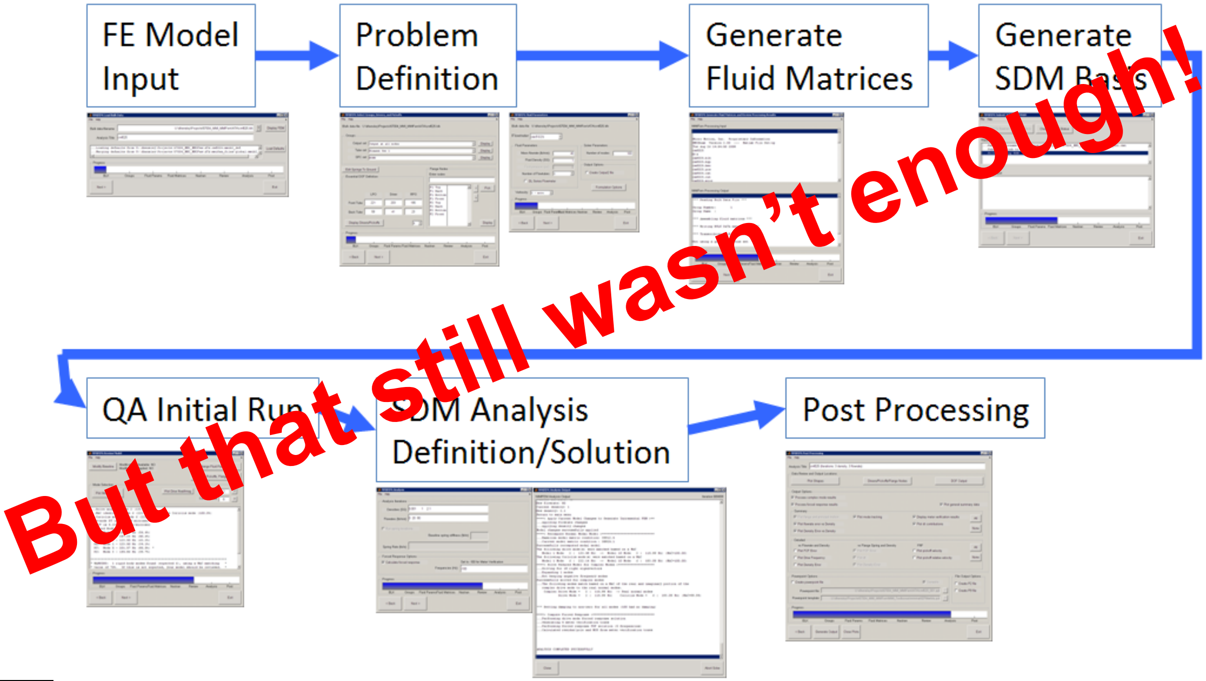



```
bcmd = { ...
    'Complex',  {'Drive',model(1).dpo.drivemode.user,false,'0',false}; ...
    'Select',   {index}; ...
    'Define',   {}; ...
        'Change Fluid Density', {inputs.denscalfact(m)}; ...
        'RETURN',               {}; ...
    'Apply',    {}; ...
    'Recompute',{}; ...
    'Solve',    {}; ...
    'Exit',     {}      };

modelr = buildmodel(modelr,1,bcmd);'));
```

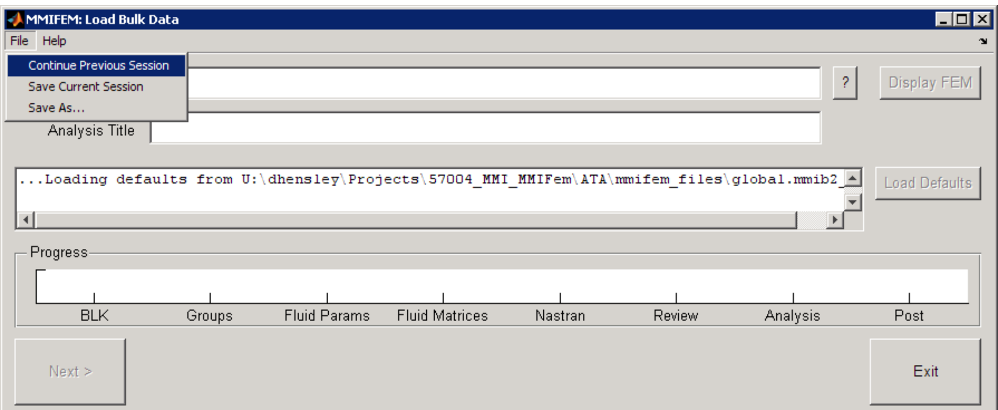
A high level GUI was necessary to satisfy everyone

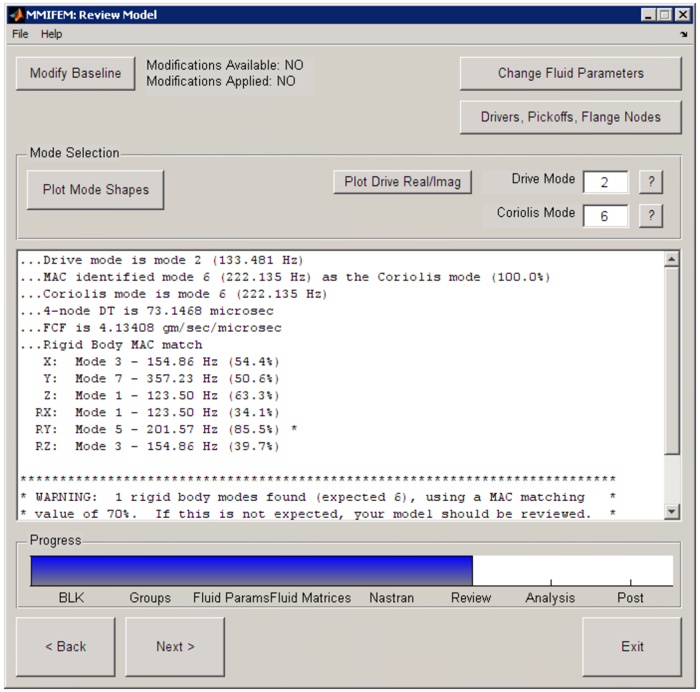
- Get feedback on GUI design early and often from users
- Step users through the entire analysis process with logically designed forms
- Software was now powerful AND easy-to-use



And we added even more capabilities

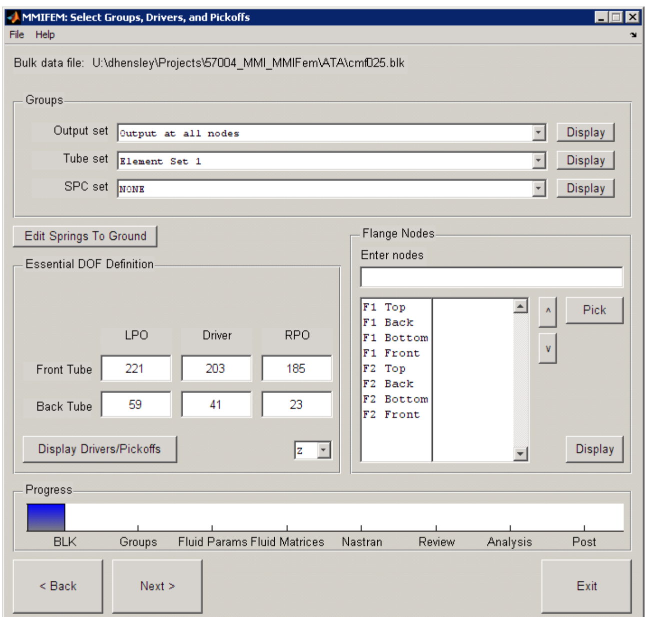
- Automatic restarts
- Lots of quality assurance for common errors
- Standardize post-processing output
- Allow advanced functionality but hide it when not needed





Automated Quality Assurance checks make everyone happy

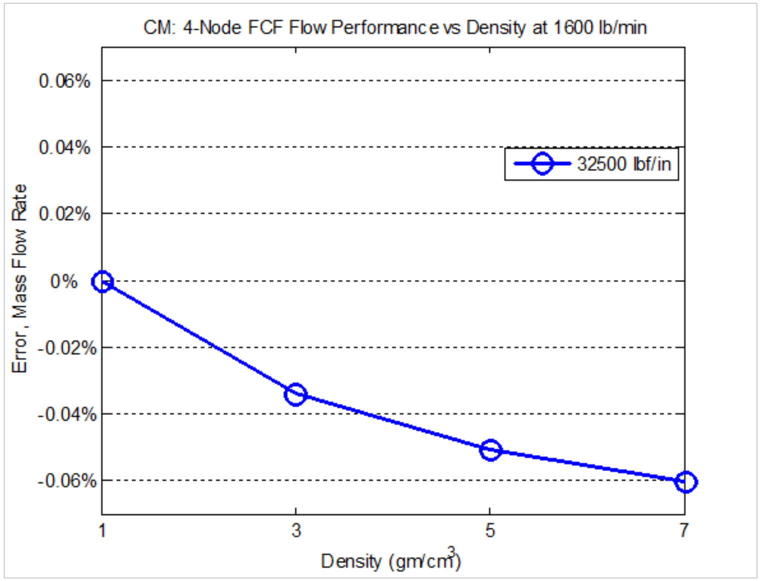
- Learn from pitfalls the experts ran into over the years
- Build checks into every step of the process

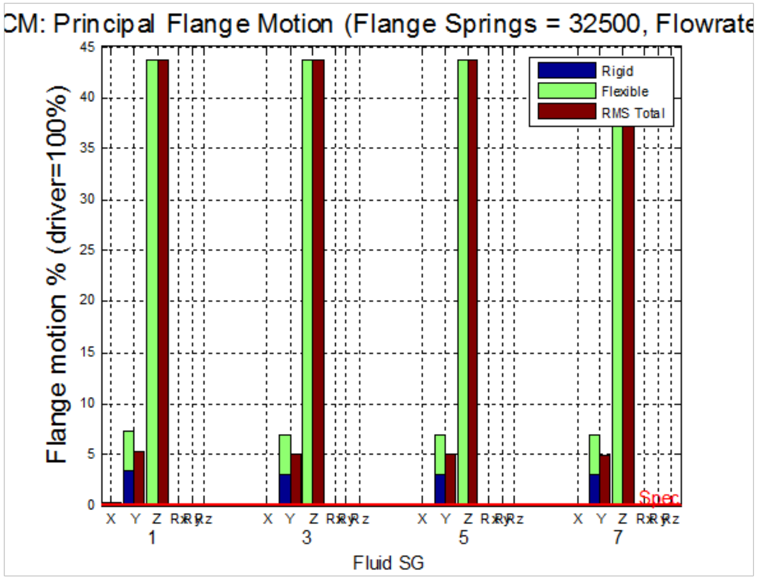


- Are necessary definition nodes in the output group?
- Are the tube beam definitions and properties valid?
- Is the number of flange nodes valid?
- Do nodes have the correct coordinate system?
- Are all of the necessary parameters defined?

Standardized Output Makes Analysis Comparison Possible

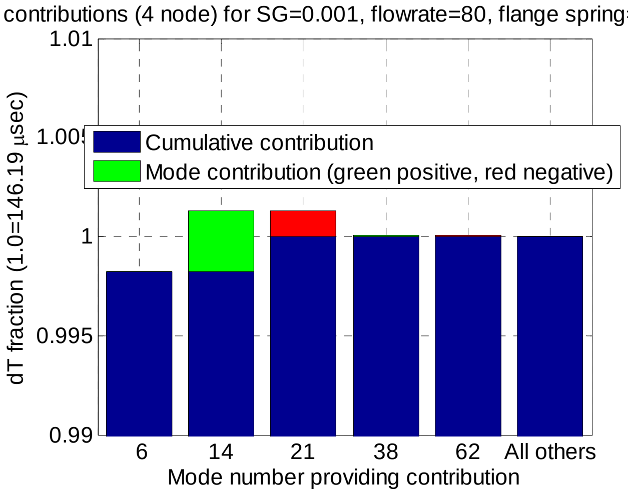
- Got consensus from users on results format
- More easily compare different analyses
- Optionally auto-create Powerpoint file to completely document analysis iterations





We finally achieved adoption of the new tools

- A senior engineer had problems with a new flowmeter design
  - The old tools were not helping him
  - We developed new (now standard) output results that provided additional insight that quantified his intuition
- He went from a skeptic to a champion of the new tools
- The biggest skeptic can provide some of the most useful feedback!



Conclusion

- Software needs to be both powerful AND easy-to-use
- Get user consensus on THEIR perceptions of ease of use and power
- Use software development best practices
- Additional features make the effort to switch to the new tools worthwhile
- Users finally adopted our tools when we made the interface SIGNIFICANTLY better

References

- “Using IMAT and MATLAB for Coriolis Flowmeter Design”, T. Cunningham & D. Hensley, IMAC XIX Proceedings, February 2001, Paper #74
- “Understanding Structural Dynamic Modification and the Effects of Truncation 1”, P. Avitabile & J. O’Callahan, International Journal of Analytical and Experimental Modal Analysis v6 n4 p215-235 Oct 1991
- MODAL SPACE Back to Basics, P. Avitabile, EXPERIMENTAL TECHNIQUES, March/April 2004 pp13-14
- “A Finite Element for the Vibration Analysis of a Fluid-conveying Timoshenko Beam”, C. Stack, R. Garnett, G. Pawlas, 34th SDM conference proceedings, 1993, AIAA