Software Tool for Computing Maximum Von Mises Stress

Speaker: Eric Martin

Author(s):
Long Chen, JPL
Kurt Knutson ATA Engineering
Eric Martin, ATA Engineering

June 26-28, 2007
Software Tool for Maximum Von Mises Stress

- Motivation, Background, and Summary of Equations
- Software Approach
- Outline of Algorithm & Source
- Example Problem #1, Solid Beam of Solids
- Example Problem #2, Hollow Beam of Shells
- Example Problem #3, Detailed Part Example
- Software Tool Applications
- Limitations and Future Works
- References and Acknowledgements
- Appendix A: Equations and Notes on Matrix [M]
- Appendix B: Complete Source Scripts with Comments
Motivation:
The Acceleration Loading in Any Direction

• Design Requirement: 200g in any direction
  "What are the stress margins in a given part?"
  "What are the directions that the part is most sensitive to?"

• Example and historic previous methods for calculating stress due to acceleration loading
  – Hand calculations in one or more directions deemed critical by an individual analyst
  – Running models with several direction vectors to summarize results
  – RSS combinations of loads; which is not an accurate nor a bounding description of the Von Mises stress in the worst loading direction.

• These methods can be time consuming, prone to errors in judgments and/or execution, and do not present the maximum direction or associated Von Mises Stress.
Mass Acceleration Curve

- Concept of Mass Acceleration Curves (MAC)
  - Developed by JPL to perform preliminary structural sizing
  - Mariners, Voyager, Galileo, Pathfinder, MER, ... MSL
  - Acceleration of physical masses are bounded by a curve
  - G-levels of vibro-acoustic and transient environments
  - Convergent process before the couple loads cycle
  - Semi-empirical method to bound the loads, not a simulation of the actual response

![Preliminary Mass Acceleration Curve for Appendages of MSL Spacecraft Launched on Delta IV or Atlas V](image-url)
Background


• The idea is to calculate the magnitude and value of a critical acceleration direction Von Mises Stress, accomplished by:
  - Applying three orthogonal acceleration loadings
  - Superimposing stress results from three loadcases (stresses are weighted sums from the three directions)
  - Calculating Von Mises stress from the stress components
  - Differentiating the Von Mises stress equation with respect to each loading case weight and setting to zero to provide a system of equations to solve for direction and value of maximum Von Mises stress for each element on a part or structure.

• Details for the mathematics can be found in the 2006 reference above or the appendix to this presentation.

• This presentation focuses on a software tool / approach to determine the Maximum Von Mises stress and direction.
Mises stress is determined from a stress state as:

\[ \sigma_v = \sqrt{\frac{1}{2} \left( \left( \sigma_x - \sigma_y \right)^2 + \left( \sigma_y - \sigma_z \right)^2 + \left( \sigma_z - \sigma_x \right)^2 \right) + 3 \left( \tau_{xy}^2 + \tau_{yz}^2 + \tau_{zx}^2 \right)} \]

The six stress components follow superposition from individual direction contributions. The superscript (capital) indicates a loading case and the subscript (small) indicates the stress component. The multipliers, A, B and C represent the weighted variable from each loading case to that stress component.

\[
\sigma_x = A\sigma_x^A + B\sigma_x^B + C\sigma_x^C \\
\sigma_y = A\sigma_y^A + B\sigma_y^B + C\sigma_y^C \\
\sigma_z = A\sigma_z^A + B\sigma_z^B + C\sigma_z^C \\
\tau_{xy} = A\tau_{xy}^A + B\tau_{xy}^B + C\tau_{xy}^C \\
\tau_{yz} = A\tau_{yz}^A + B\tau_{yz}^B + C\tau_{yz}^C \\
\tau_{zx} = A\tau_{zx}^A + B\tau_{zx}^B + C\tau_{zx}^C
\]
Summary of Equations (con't)

Differentiate the Mises stress with respect to each loading case weight and set to zero.

$$\frac{\partial \sigma_y}{\partial A} = 0 \quad \frac{\partial \sigma_y}{\partial B} = 0 \quad \frac{\partial \sigma_y}{\partial C} = 0$$

The results can be written in a matrix form and be solved by the eigen transformation. The determinant is the characteristic equation for the matrix. The root $\lambda$ of the equation is an eigenvalue of the matrix and the corresponding values for $A$, $B$ and $C$ is the eigenvector corresponding to the $\lambda$ eigenvalue.

$$\begin{bmatrix} (\sigma_y^A)^2 & b & c \\ b & (\sigma_y^B)^2 & d \\ c & d & (\sigma_y^C)^2 \end{bmatrix} \begin{bmatrix} A \\ B \\ C \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \quad \begin{vmatrix} \lambda^2 -(\sigma_y^A)^2 & -b & -c \\ -b & \lambda^2 -(\sigma_y^B)^2 & -d \\ -c & -d & \lambda^2 -(\sigma_y^C)^2 \end{vmatrix} = 0$$

Spacecraft and Launch Vehicle

ATA ENGINEERING, INC

JPL

NASA
Formulation as a Matrix Eigenvalue Problem

Want to solve for the eigenvalue (max Von Mises stress) and eigenvectors (max Von Mises stress direction) for each stress point associated with:

\[
\begin{vmatrix}
\lambda^2 - (\sigma_v^A)^2 & -b & -c \\
-b & \lambda^2 - (\sigma_v^B)^2 & -d \\
-c & -d & \lambda^2 - (\sigma_v^C)^2
\end{vmatrix} = 0
\]

Where \( \sigma_v^A, \sigma_v^B, \sigma_v^C \) are expressions for the Von Mises stress for each of the individual load direction vectors and the b,c,d are expressions for specific terms off the diagonal associated with the partial derivatives. These terms can be found in the appendix A.
Simple Matrix Expression Is a 3x3 Eigenvalue Problem Per Stress Location.

Define

$$\sigma = \begin{bmatrix} \sigma_x^A & \tau_{xy}^A & \sigma_y^A & \tau_{xz}^A & \tau_{yz}^A & \sigma_z^A \\ \sigma_x^B & \tau_{xy}^B & \sigma_y^B & \tau_{xz}^B & \tau_{yz}^B & \sigma_z^B \\ \sigma_x^C & \tau_{xy}^C & \sigma_y^C & \tau_{xz}^C & \tau_{yz}^C & \sigma_z^C \end{bmatrix}$$

$$[A] = \begin{bmatrix} 1 & 0 & -0.5 & 0 & 0 & -0.5 \\ 0 & 3 & 0 & 0 & 0 & 0 \\ -0.5 & 0 & 1 & 0 & 0 & -0.5 \\ 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ -0.5 & 0 & -0.5 & 0 & 0 & 1 \end{bmatrix}$$

$[\sigma]$ is a 3x6 matrix of stress components, where each row is associated with each solved orthogonal direction and each column is a component of stress for that direction.

There is a 6x6 coefficient matrix $[A]$ such that the matrix $[M]$ can be easily written as:

$$[M]_{3x3} = \sigma_{3x6} [A]_{6x6} \sigma_{6x3}$$

The matrix $M$ is a 3x3 matrix. The maximum eigenvalue is the maximum Von Mises stress and associated eigenvector is the direction of maximum stress.

The matrix $[M]$ can be verified through multiplication and comparison to reference [1].

The form of $[A]$ can also be seen easily from reference [2].
Software Approach

• Use Commercial Off The Shelf (COTS) Software Wherever Possible:
  – FE Solver (Nastran)
  – Read stresses as objects (IMAT (Matlab toolbox))
    see http://www.ata-e.com/software/imat/
  – Apply Matrix Math and Obtain Eigenvalues (Matlab)
  – Post Processing (I-deas) for viewing stresses and stresses from maximum direction that can be superimposed

• Work with large models for parts (hundreds of thousands of elements per part)
• Easily summarize maximum direction Von Mises magnitude and direction for every stress location
• Result is a tensor that can be reused for combined stress states, (i.e. more than maximum acceleration).
• Easy automation for multiple materials, parts, etc.
Diagram of Software Approach

Nastran Stresses (3 subcases) part.op2

FE solver

IMAT Matlab Toolbox
maxdir.m
1. Sets the units of the .op2 file
2. Load stresses from Nastran to IMAT Results Object
3. Call max_von_mises
4. Write output to .unv, .csv

Table of stresses and associated vectors part.csv

Maximum Von Mises Stress Tensor part.unv

Matrix Math
max_von_mises.m
3.1 Loop over all stress points in stress object.
3.2 Calculate $[M] = \sigma [A] \sigma^T$.
3.3 Solve for eigenvalues.
3.4 Sort eigenvalues.
3.5 Return With:
   (Result Object) - maximum Von Mises Stress Tensor
   (Array Object) - a sorted array of stress values and directions for entire domain

EXCEL.EXE

I-DEAS12.exe FE Post

Nastran.exe

Spacecraft and Launch Vehicle Dynamic Environments (SLE)

ATA ENGINEERING, INC

JPL

NASA
Notes for Software Approach

[1] Solve for stresses using a commercial finite element solver program [NX Nastran], for each of 3 directions and scale factors (3 subcases for acceleration in X,Y,Z). Output can be per part or material with sets.

[2] Read these stresses direct from binary output as an object containing component values for each location and subcase into application memory [IMAT].

[3] Use a commercial application to quickly form the 3x3 Matrix [M] and compute the eigenvalues and eigenvectors and sort to get the maximum Von Mises Stress and associated load vector weighting factors (stress direction) for each stress location [MATLAB].

[4] Output the full maximum Von Mises Stress stress tensor to a format that a commercial post-processing code can read [I-deas] and output a direct listing of the sorted stress values and directions [Excel], for quick and easy summary. The stress tensor can then be added with other known loadcases as necessary such as assembly, thermal or other loadcases that may need super-position with maximum direction for part stresses and margins.


[6] Key direction vectors for a structure can easily be found and verified.
function maxdir(filename, units)
    setunits(units);
    disp(sprintf("Processing %s", filename));
    data=readnas([filename,' .op2'], 'blocks', 'OES1', 'silent', 'noprogbars');
    [von_mises, tab_data]=max_von_mises(data.result);
    writeunv(von_mises,[filename,'.unv']);
    tab_data=sortrows(tab_data,-4);
    FID = fopen([filename,'.csv'],'w');
    fprintf(FID, 'Element ID, Local Node ID, Layer #, Von Mises Stress, dir(1),
            dir(2),dir(3)\n');
    rows_out=length(tab_data);
    for k=1:rows_out
        fprintf(FID, '%d, %d, %d, %d, %d, %d, %d\n', tab_data(k,1), tab_data(k,2),
                tab_data(k,3),
                tab_data(k,4), tab_data(k,5), tab_data(k,6), tab_data(k,7));
    end
    fclose(FID);
    return
function [vm_max_stress, tabular_data] = max_von_mises(component_stress)
vm_max_stress = component_stress(1);
vm_max_stress.Name = 'Worst Case Von Mises Stress';
vm_max_stress.data = zeros(size(component_stress(1).data));
output_data = vm_max_stress.data;
tabular_data = zeros(length(component_stress.data), 4);
stress_data = component_stress.data;
A = [1 0 -.5 0 0 -.5; 0 3 0 0 0 0; -.5 0 1 0 0 -.5;
   0 0 0 3 0 0; 0 0 0 0 3 0; -.5 0 -.5 0 0 1];
for ii = 1:6:length(stress_data)
    stress_tensor = stress_data(ii:ii+5,:);
    [V, D] = eig(stress_tensor*A*stress_tensor');
    max_D = max(max(D));
    max_V = V(:, find(max_D == D(1, 1) D(2, 2) D(3, 3)));
    output_data(ii:ii+5, 1) = stress_tensor'*max_V;
    tabular_data(ii,:) = [sqrt(max_D) max_V];
end

tabular_data = [component_stress.numericcomponents(1:6:end,[1:2,4]) tabular_data(1:6:end,:)];
vm_max_stress.data = output_data;
return
Example Problem #1: Solid Bar of Brick Elements Fixed at One End

All Dimensions shown in mm.

Fixed in Y-Z Plane, Load = 1g acceleration in any direction

CHEXA Elements 25.4/3 mm per side.
Material = Aluminum,
E=6.83E10Pa, nu=0.33, Rho =2710kg/m^3
Example Problem #1: Sample NX Nastran Input Deck

$ NX NASTRAN MODEL OF SOLID BAR MODELED WITH BRICKS
$ Author: Kurt Knutson, Date: 08/01/07, Units: SI, Model Mass = 8.882E-01 kg
$ Acceleration, 1g 3 orthogonal directions
SOL 101
CEND
TITLE = ex_01_solid_bar.dat
SUBTITLE = example, solid bar, unit 1g X, Y, Z cases
ECHO = NONE
DISPLACEMENT(PLOT) = ALL
STRESS(PLOT) = ALL
SPC = 1
SUBCASE 101
  LABEL = 1g X LOAD
  LOAD = 101
SUBCASE 102
  LABEL = 1g Y LOAD
  LOAD = 102
SUBCASE 103
  LABEL = 1g Z LOAD
  LOAD = 103
BEGIN BULK
PARAM AUTOSPC YES
PARAM GRDPNT 0
$ op2 file formatted for I-deas, IMAT
PARAM POST -2
$ Unit 1g loads on structure
GRAV 101 0 1.00 9.81 0.0 0.0
GRAV 102 0 1.00 0.0 9.81 0.0
GRAV 103 0 1.00 0.0 0.0 9.81
include 'solid_bar.blk'
ENDDATA
Example Problem #1:
Von Mises Stress (Pa) for 1g X,Y,Z &
Execution to get Maximum Von Mises Direction.

Simple Script
Generates
Maximum Direction
Example Problem #1:
Maximum Von Mises Stress & Maximum Direction.

**Nodal Contour**

**Elemental Contour**
Example Problem #1: Table of maximum values and directions for all elements.

- A file called `ex_01_solid_bar.csv` summarizes the sorted maximum values and directions per location.

<table>
<thead>
<tr>
<th>Element ID</th>
<th>Local Node</th>
<th>Layer #</th>
<th>Von Mises Stress</th>
<th>dir(1)</th>
<th>dir(2)</th>
<th>dir(3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>4</td>
<td>2.40E+05</td>
<td>3.47E-02</td>
<td>8.81E-01</td>
<td>4.73E-01</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>3</td>
<td>2.40E+05</td>
<td>3.47E-02</td>
<td>8.81E-01</td>
<td>-4.73E-01</td>
</tr>
<tr>
<td>361</td>
<td>361</td>
<td>8</td>
<td>2.40E+05</td>
<td>3.47E-02</td>
<td>-8.81E-01</td>
<td>4.73E-01</td>
</tr>
<tr>
<td>366</td>
<td>366</td>
<td>7</td>
<td>2.40E+05</td>
<td>3.47E-02</td>
<td>-8.81E-01</td>
<td>-4.73E-01</td>
</tr>
</tbody>
</table>

Maximum Value is 13.2% higher than maximum unit direction.
Example Problem #1:
Verification of Maximum Direction by Re-running Maximum Vector.

Re-running the maximum Direction Vector with Nastran Directly Yields:

Output From Maximum Direction Reapplied, Gives Load consistent result on structure and verifies result of 2.40E5Pa
Example Problem #2: Hollow Bar of Shell Elements Fixed at One End

All Dimensions shown in mm.

Fixed in Y-Z Plane, Load = 1g acceleration in any direction

CQUAD4 Elements 25.4/3 mm per side.
Material = Aluminum, Thickness = 1mm
E=6.83E10Pa, nu=0.33, Rho = 2710kg/m^3
Result of Maximum Direction for Example #2

RX_02_SHELL_BAR
Worst CASE VON MISES STRESS
TITLE = RX_02_SHELL_BAR.ANT
STRESS VON Mises Unaveraged Top shell
Max: 1.438E05 Pa Max: 1.209E05 Pa
Part Coordinate System

Spacecraft
and Launch Vehicle
Dynamic Environment Model

22

ATA
ENGINEERING, INC.

JPL
NASA
Example #3, Detailed Part Example

~123,000 Nodes
~35,700 Elements
Maximum Direction Von Mises Contour
for Unit Cases for Example #3
Software Tool Areas of Interest

• The maximum Von Mises stress and stress direction are of interest for analyzing launch accelerations such as with the Mass Acceleration Curves developed by JPL.

• Maximum launch stresses can be combined with appropriate load cases at consistent locations with resulting stress tensors.

• Maximum Von Mises stress is also of interest for understanding maximum operational loading such as traverse events.
  - For example, planetary traversing simulations may prescribe bounding acceleration values during traverse for a rover such as Mars Science Lab (MSL) in (X,Y,Z) of the rover.
  - Such accelerations can be really in any directions for many parts such as a mast or head mounted components which can be in numerous configurations and orientations when traversing a planet surface.
Limitations and Future Work

• Limitations
  - Stress output from Nastran needs to be PARAM, POST, -2. For the IMAT objects to be read.
  - Corner stresses need to be used when using shell elements STRESS(PLOT,CORNER) = ALL.
  - There is not enough direct stress data output from Nastran to do elements such as Beams and Bars in a Von Mises sense for the stress recovery points.

• Future Work and Notes
  - The 3x3 matrix \([M]\) can be written directly without the 6x6 coefficient matrix to avoid matrix multiplications and additions with zero on the first matrix multiply.
  - Would like to see the same approach work with FEMAP for post. Some enhancements to the script to format out neutral file output and/or to the IMAT toolbox to read PARAM, POST, -1 may allow this in the future.
  - Interested in developing other maximum direction solutions for stress functions besides Von Mises.
  - Encourage FE tools and software vendors to include Maximum Von Mises value and direction as a standard post-processing or add-on feature.
References


Author Acknowledgements

- Maximum Von Mises Stress Methods
  - Long Chen (JPL) & Bob Glaser (JPL)
  - Matt Orzewalla (JPL)
- Software Contributions
  - Kurt Knutson (ATA)
  - Eric Martin (ATA)
  - Charles Webber (ATA)
Appendix A: Equations and Matrix M.

Most of the equations here in Appendix A are already summarized and published online, refer to:
http://www.aero.org/conferences/sclv/2006proceedings.html

Appendix A: Equations and Matrix M

Mises stress is determined from a stress state as:

\[ \sigma_M = \sqrt{\frac{1}{2} \left( (\sigma_x - \sigma_y)^2 + (\sigma_y - \sigma_z)^2 + (\sigma_z - \sigma_x)^2 \right) + 3(\tau_{xy}^2 + \tau_{yz}^2 + \tau_{zx}^2)} \]

The six stress components follow superposition from individual direction contributions. The superscript (capital) indicates a loading case and the subscript (small) indicates the stress component. The multipliers, A, B and C represent the weighted variable from each loading case to that stress component.

\[ \sigma_x = A\sigma_x^A + B\sigma_x^B + C\sigma_x^C \]
\[ \sigma_y = A\sigma_y^A + B\sigma_y^B + C\sigma_y^C \]
\[ \sigma_z = A\sigma_z^A + B\sigma_z^B + C\sigma_z^C \]

\[ \tau_{xy} = A\tau_{xy}^A + B\tau_{xy}^B + C\tau_{xy}^C \]
\[ \tau_{yz} = A\tau_{yz}^A + B\tau_{yz}^B + C\tau_{yz}^C \]
\[ \tau_{zx} = A\tau_{zx}^A + B\tau_{zx}^B + C\tau_{zx}^C \]
Appendix A: Equations and Matrix M

Substitution for the components in the Mises equation from the weighted load cases:

\[
\frac{1}{2} (\sigma_x - \sigma_y)^2 = \frac{1}{2} \left( A (\sigma_x^A - \sigma_y^A) + B (\sigma_x^B - \sigma_y^B) + C (\sigma_x^C - \sigma_y^C) \right)^2 \\
\frac{1}{2} (\sigma_y - \sigma_z)^2 = \frac{1}{2} \left( A (\sigma_y^A - \sigma_z^A) + B (\sigma_y^B - \sigma_z^B) + C (\sigma_y^C - \sigma_z^C) \right)^2 \\
\frac{1}{2} (\sigma_z - \sigma_x)^2 = \frac{1}{2} \left( A (\sigma_z^A - \sigma_x^A) + B (\sigma_z^B - \sigma_x^B) + C (\sigma_z^C - \sigma_x^C) \right)^2 \\
3 \tau_{xy}^2 = 3 \left( A \tau_{xy}^A + B \tau_{xy}^B + C \tau_{xy}^C \right)^2 \\
3 \tau_{yz}^2 = 3 \left( A \tau_{yz}^A + B \tau_{yz}^B + C \tau_{yz}^C \right)^2 \\
3 \tau_{zx}^2 = 3 \left( A \tau_{zx}^A + B \tau_{zx}^B + C \tau_{zx}^C \right)^2
\]

Differentiate the Mises stress with respect to each loading case weight and set to zero.

\[
\frac{\partial \sigma_v}{\partial A} = 0 \quad \frac{\partial \sigma_v}{\partial B} = 0 \quad \frac{\partial \sigma_v}{\partial C} = 0
\]
Appendix A: Equations and Matrix M

The derivative of the Mises equation with respect to scale factor A is below:

\[
\frac{\partial \sigma_v}{\partial A} = \frac{1}{2\sigma_v} \left( A(\sigma_x^A - \sigma_y^A) + B(\sigma_x^B - \sigma_y^B) + C(\sigma_x^C - \sigma_y^C) \right) (\sigma_y^A - \sigma_y^A) + \\
\frac{1}{2\sigma_y} \left( A(\sigma_y^A - \sigma_z^A) + B(\sigma_y^B - \sigma_z^B) + C(\sigma_y^C - \sigma_z^C) \right) (\sigma_y^A - \sigma_z^A) + \\
\frac{1}{2\sigma_z} \left( A(\sigma_z^A - \sigma_x^A) + B(\sigma_z^B - \sigma_x^B) + C(\sigma_z^C - \sigma_x^C) \right) (\sigma_z^A - \sigma_x^A) + \\
\frac{3}{\sigma_y} \left( A\tau_{xy}^A + B\tau_{xy}^B + C\tau_{xy}^C \right) \tau_{xy}^A + \\
\frac{3}{\sigma_y} \left( A\tau_{yz}^A + B\tau_{yz}^B + C\tau_{yz}^C \right) \tau_{yz}^A + \\
\frac{3}{\sigma_y} \left( A\tau_{zx}^A + B\tau_{zx}^B + C\tau_{zx}^C \right) \tau_{zx}^A = 0
\]
Appendix A: Equations and Matrix M

Taking each derivative setting them equal to zero results in an equation that is linear with A, B, and C:

For: \[ \frac{\partial \sigma_v}{\partial A} \rightarrow aA + bB + cC = 0 \]

Where:

\[
\begin{align*}
    a &= \frac{1}{2} \left( (\sigma_x^A - \sigma_y^A)^2 + (\sigma_y^A - \sigma_z^A)^2 + (\sigma_z^A - \sigma_x^A)^2 \right) + 3 \left( \tau_{xy}^A \right)^2 + 3 \left( \tau_{yz}^A \right)^2 + 3 \left( \tau_{zx}^A \right)^2 \left( \sigma_v^A \right)^2 \\
    b &= \frac{1}{2} \left( (\sigma_x^B - \sigma_y^B) (\sigma_x^A - \sigma_y^A) + (\sigma_y^B - \sigma_z^B) (\sigma_y^A - \sigma_z^A) + (\sigma_z^B - \sigma_x^B) (\sigma_z^A - \sigma_x^A) \right) + 3 \left( \tau_{xy}^B \tau_{xy}^A \right) + 3 \left( \tau_{yz}^B \tau_{yz}^A \right) + 3 \left( \tau_{zx}^B \tau_{zx}^A \right) \\
    c &= \frac{1}{2} \left( (\sigma_x^C - \sigma_y^C) (\sigma_x^A - \sigma_y^A) + (\sigma_y^C - \sigma_z^C) (\sigma_y^A - \sigma_z^A) + (\sigma_z^C - \sigma_x^C) (\sigma_z^A - \sigma_x^A) \right) + 3 \left( \tau_{xy}^C \tau_{xy}^A \right) + 3 \left( \tau_{yz}^C \tau_{yz}^A \right) + 3 \left( \tau_{zx}^C \tau_{zx}^A \right)
\end{align*}
\]
Appendix A: Equations and Matrix M

The results can be written in a matrix form by inspection for all three derivatives and be solved by the eigen transformation. The determinant is the characteristic equation for the matrix. The root \( \lambda \) of the equation is an eigenvalue of the matrix \([M]\) and the corresponding values for \( A, B \) and \( C \) is the eigenvector corresponding to the \( \lambda \) eigenvalue.

\[
\begin{bmatrix}
(\sigma_v^A)^2 & b & c \\
 b & (\sigma_v^B)^2 & d \\
 c & d & (\sigma_v^C)^2
\end{bmatrix}
\begin{bmatrix}
A \\
B \\
C
\end{bmatrix}
= 
\begin{bmatrix}
0 \\
0 \\
0
\end{bmatrix}
\]

\[
\begin{vmatrix}
\lambda^2 - (\sigma_v^A)^2 & -b & -c \\
-b & \lambda^2 - (\sigma_v^B)^2 & -d \\
-c & -d & \lambda^2 - (\sigma_v^C)^2
\end{vmatrix} = 0
\]
Appendix A: Equations and Matrix \( M \)

\[
[\sigma] = \begin{bmatrix}
\sigma_x^A & \tau_{xy}^A & \sigma_y^A & \tau_{xz}^A & \tau_{yz}^A & \sigma_z^A \\
\sigma_x^B & \tau_{xy}^B & \sigma_y^B & \tau_{xz}^B & \tau_{yz}^B & \sigma_z^B \\
\sigma_x^C & \tau_{xy}^C & \sigma_y^C & \tau_{xz}^C & \tau_{yz}^C & \sigma_z^C
\end{bmatrix}
\]

\[
[A] = \begin{bmatrix}
1 & 0 & -0.5 & 0 & 0 & -0.5 \\
0 & 3 & 0 & 0 & 0 & 0 \\
-0.5 & 0 & 1 & 0 & 0 & -0.5 \\
0 & 0 & 0 & 3 & 0 & 0 \\
0 & 0 & 0 & 0 & 3 & 0 \\
-0.5 & 0 & -0.5 & 0 & 0 & 1
\end{bmatrix}
\]

This Matrix is shown in reference [2]

Verify \( [M] = [\sigma][A][\sigma'] \)

\[
[\sigma][A] = \begin{bmatrix}
\sigma_x^A & \frac{\sigma_y^A}{2} & \frac{\sigma_z^A}{2} & 3\tau_{xy}^A & \frac{\sigma_y^A}{2} & \frac{\sigma_z^A}{2} & 3\tau_{xz}^A & 3\tau_{yz}^A & \frac{\sigma_y^A}{2} & \frac{\sigma_z^A}{2} & \frac{\sigma_z^A}{2} \\
\sigma_x^B & \frac{\sigma_y^B}{2} & \frac{\sigma_z^B}{2} & 3\tau_{xy}^B & \frac{\sigma_y^B}{2} & \frac{\sigma_z^B}{2} & 3\tau_{xz}^B & 3\tau_{yz}^B & \frac{\sigma_y^B}{2} & \frac{\sigma_z^B}{2} & \frac{\sigma_z^B}{2} \\
\sigma_x^C & \frac{\sigma_y^B}{2} & \frac{\sigma_z^B}{2} & 3\tau_{xy}^C & \frac{\sigma_y^C}{2} & \frac{\sigma_z^C}{2} & 3\tau_{xz}^C & 3\tau_{yz}^C & \frac{\sigma_y^C}{2} & \frac{\sigma_z^C}{2} & \frac{\sigma_z^C}{2}
\end{bmatrix}
\]
Appendix A: Equations and Matrix M

Now Multiply $[\sigma][A]$ by $[\sigma']$

Writing out all of the multiplications, combining terms and factoring the resulting polynomials, the following result is obtained:

$$[\sigma][A][\sigma]' = \begin{bmatrix} (\sigma_v^A)^2 & b & c \\ b & (\sigma_v^B)^2 & d \\ c & d & (\sigma_v^C)^2 \end{bmatrix} = [M]$$
function maxdir(filename, units)
% File: maxdir.m
% Author: Kurt Knutson
% Date : 03/10/2007
% Input:
% The input is the base file name of the .op2 file and the units.
% Output: (filename.unv, filename.csv)
% filename.unv => I-deas .unv file with stress tensor associated with
% filename.csv => Comma seperated file with sorted Von Mises Maximums:
% [Element ID, Local Node ID, Layer Number, Max Von Mises Stress, A, B, C]
% The contents of this file are sorted on Maximum Von Mises Stress
% Example Usage:
% maxdir(op2file, 'SI')
% Set the units of the .op2 file
% setunits(units);
disp(sprintf('Processing %s',filename));
% Read the data from the Nastran .op2 file
data=readnas([filename,' .op2'],'blocks',('OESI'),'silent','noprogbar');
% Call the max_von_mises function
[von_mises, tab_data]=max_von_mises(data.result);
% Write the maximum Von Mises Stress to a tensor for visualization
writeunv(von_mises,[filename,'.unv']);
% Sort the tabulated data
tab_data=sortrows(tab_data,-4);
% Write the tabulated data to an Excel File
FID = fopen([filename,'.csv'],'w');
fprintf(FID,'%s
', [filename,'.op2']);
fprintf(FID, 'Element ID, Local Node ID, Layer #, Von Mises Stress, dir(1), dir(2), dir(3)
');
rows_out=length(tab_data);
for k=1:rows_out
    fprintf(FID, '%d, %d, %d, %d, %d, %d, %d
', tab_data(k,1), tab_data(k,2), tab_data(k,3),
            tab_data(k,4), tab_data(k,5), tab_data(k,6), tab_data(k,7));
end
fclose(FID);
return
Appendix B: Script Source Files, max_von_mises.m

function [vm_max_stress tabular_data]=max_von_mises(component_stress)
%File: max_von_mises.m
%Author: Kurt Knutson, Eric Martin, Charles Webber
%Date: 3/10/07

%MAX_VON_MISES  Highest von Mises stress for a load of unknown direction
%
%[VON_MISES,TABULAR_DATA]=MAX_VON_MISES(COMPONENT_STRESSES)
% MAX_VON_MISES returns an IMAT result object containing stress tensors
% associated with the highest von Mises stress for a load of known
% magnitude, but unknown direction. COMPONENT_STRESS is a 3x1 vector of
% IMAT result objects containing stresses associated with the load applied
% in 3 orthogonal directions. VON_MISES contains stress tensors at the
% same locations (I.E. element centroids or corners) contained in
% COMPONENT_STRESS.
%
%[VON_MISES,TABULAR_DATA]=MAX_VON_MISES(COMPONENT_STRESS) returns an
%array containing the following:
%
%[Element ID, Local Node ID, Layer Number, Max von Mises Stress, A, B, C]
%
% where A, B, and C are scale factors on the 3 orthogonal load cases which
% results in the maximum von Mises stress.
%
%Since VON_MISES is an IMAT result object, it can be written to a
%universal file using WRITEUNV for plotting stress contours in IDEAS.
%Note that since the load directions associated with adjacent elements
%need not be the same, it may not be appropriate to turn averaging on in
%Visualizer. Also, by default, NASTRAN stores stresses on thin shell
%elements at element centroids. IDEAS does not support centroidal
%stresses in UNV files. You can ask for nodal stresses in NASTRAN with
%STRESS(PLOT, CORNER)=ALL.
%
%Sample usage:
%DATA = readnas('my_filename.op2');
%[MAX_VON_MISES TABULAR_DATA] = max_von_mises(DATA.result);
%writeunv(MAX_VON_MISES,'to_ideas.unv');
% instantiate a result object to store the worst case stresses
vm_max_stress = component_stress(l);
vm_max_stress.Name = 'Worst Case Von Mises Stress';
% initialize vm_max_stress to store zeros for all tensors
vm_max_stress.data=zeros(size(component_stress(l).data));

% Initialize some other arrays for performance improvements
output_data=vm_max_stress.data;
%tabular_data=component_stress.numericalcomponents(1:6:end,1:2);
tabular_data=zeros(length(component_stress.data),4);
stress_data = component_stress.Data;

% [stress tensor] * A * [stress tensor] = VM_stress^2
% where stress tensor = [s11 s12 s22 s13 s23 s33]
A = [1 0 -.5 0 0 -.5; 0 3 0 0 0 0; .5 0 1 0 0 -.5; 0 0 3 0 0 0; 0 0 0 3 0 0; .5 0 0 0 1];

for ii=1:6:length(stress_data)
    stress_tensor=stress_data(ii:ii+5,:)';
    [V,D] = eig(stress_tensor*A*stress_tensor');
    max_D = max(max(D));
    max_V = V(:,find(max_D == [D(1,1) D(2,2) D(3,3)]));
    output_data(ii:ii+5,1) = stress_tensor'*max_V;
    tabular_data(ii,:)=[sqrt(max_D) max_V'];
end

output_data=component_stress.numericalcomponents(1:6:end,[1:2,4]) tabular_data(1:6:end,:));
vm_max_stress.data=output_data;
return