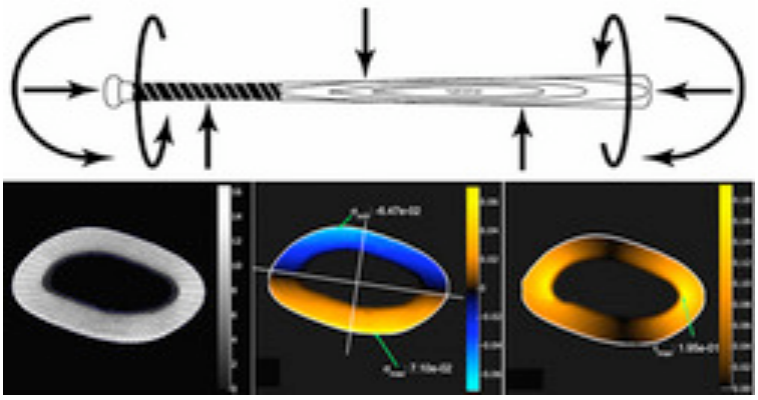
VA-BATTS V3 User's Manual



May 2008 (updated Nov 2011)

1. Background

VA-BATTS is a Matlab-based software package that calculates the out-of-plane stresses for inhomogeneous cross-sections of prismatic beams subjected to bending, axial, torsional and/or transverse shear loading. VA-BATTS is distributed free of charge for general use subject to the following conditions.

1. While every attempt has been made to eliminate programming and procedural errors, we do not guarantee the accuracy of this code. Any errors brought to our attention will be corrected in future versions of VA-BATTS.

2. We strongly encourage distribution of this software, but we can not be responsible for the integrity of source code. If you have any concerns about the software, please contact us. The latest version of the software can be obtained at https://simtk.org/home/va-batts.

3. If you modify the source code in any way, please clearly indicate (with comments in the code and in the documentation) exactly what you have changed. If you believe your addition may help other researchers, please let us know about it. If appropriate, we will include it (with proper credit) in future versions of VA-BATTS.

4. If you publish or present any results obtained with the help of VA-BATTS, we ask that you acknowledge its use. We suggest referencing the following article:

   Kourtis LC, Carter DR, Kesari H, Beaupre GS: A new software tool (VA-BATTS) to calculate bending, axial, torsional, and transverse shear stresses within bone cross sections having inhomogeneous material properties. In press: Computer Methods in Biomechanics and Biomedical Engineering, Jan 2008.

5. Finally, if you have any suggestions for improvements to the code or comments about its usefulness, please let us know.

Lampros Kourtis lakourtis@gmail.com
Gary S. Beaupre beaupre@va51.stanford.edu

RR&D Center (153)
VA Medical Center
3801 Miranda Avenue
Palo Alto, California 94304-1200

2. System Requirements

The program is written in Matlab; benchmarking was done for the R2006a and the R2007b version. Apart from the core Matlab installation, the Image Processing, Spline and Optimization toolboxes are required (optionally, the statistics toolkit is used). The program has been used on

Windows XP, Redhat and Ubuntu Linux platforms. Other platforms are also compatible, but not benchmarked. The VA-BATTS team would appreciate your feedback on functionality on other platforms as well as other suggestions, errors etc.

3. Tutorial

In this section, a couple of examples of analyzing bone stresses with VA-BATTS are presented, the first one starts with a pQCT cross sectional image (*.M01 file format) of a distal femur, and the second one deals with a DICOM image (*.dcm file format) of a midshaft of a femur.

Example 1

Open MATLAB Go to the folder where VA-BATTS is stored.

Open main.m and edit the first section where the loads are specified. In this case we are going to simulate a load applied anterolaterally on the midshaft of the femur. The cross section under investigation is one distal one, taken at 12.5% of the total length of the bone (distally). The moment arm is 150mm and the force is 1000N at 45^{o}. An axial load equal to 500N and an internal rotation of 0.01rad/mm are applied. The first section of main.m should then read:

```
%% 0. INPUT

% Flexural Loads
Mx              = 0;            %in N.mm
My              = 0;            %in N.mm
% OR (3pt bending)
Qx              = 707;          %in N
Qy              = 707;          %in N
z_pos           = 150           %in mm
% Axial Loads
P               = 500;          %in N

% Torsion displacement
theta           = 0.01;         %in rad/mm
```

The mesh density can also be defined by specifying:

```
%% Model parameters
nodes_r  = 20                   % number of nodes in the radial direction
nodes_th = 96;                  % number of nodes along the circumference
```

Also, the conversion relation (image units to density units should be reviewed. Open the image2density.m routine. Depending on the calibration, the routine should read:

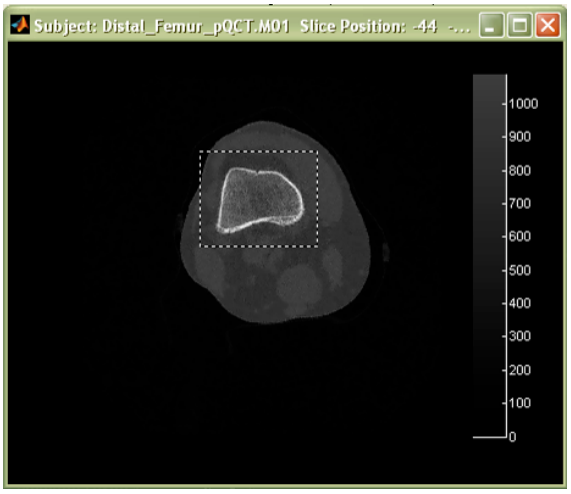
```
A=A/1000;
a=2.2092;
b=-0.5852;
%% Convert
A_d=bin_A.*abs(A*a + b*ones(size_x,size_y)); %in gr/cm3
```

These parameters have been estimated experimentally at the Palo Alto VA Hospital R&D Center. The user is advised to enter his/her own conversion relation.

-Run main.m by either typing main or by opening main.m and by clicking on the Run button; the file selection dialogue appears.

-Use one of the files included in the package, select file “Distal_Femur_pQCT.M01” and click Open. The image opens.

-Crop the image so that it shows only the region of interest (ROI) by dragging the mouse (left click) and selecting a rectangular window.



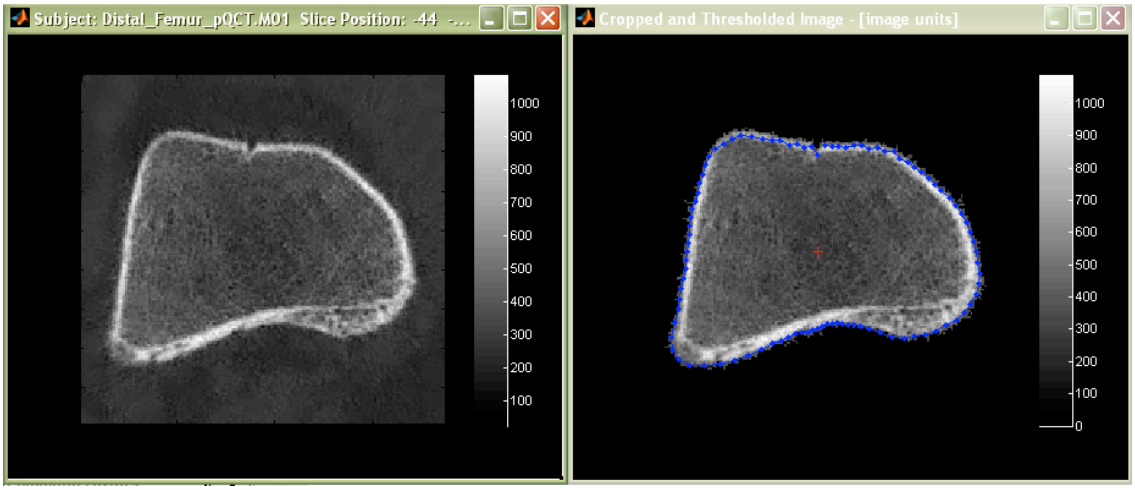
-The mesh type selection dialogue appears asking whether the mesh type should be solid or donut. Select solid as the epiphyseal cross section is non-hollow.

-The program then calculates the allowable thresholding levels.

-Click (left) on a point inside the bone (the point must be located within the medullary canal, **not** within the cortex) to indicate which bone is of interest. In cases where there are more than one bones present in the image, this will tell the program which one is to be analyzed. A second, version of the image appears on the right; the bone is contoured by a blue spline; the

control points of the spline are marked as dots and they will subsequently be used as mesh seed points.

All mouse clicks are to be done while in the **left** window unless editing of the contour control points is required; this is done in the **right** window.



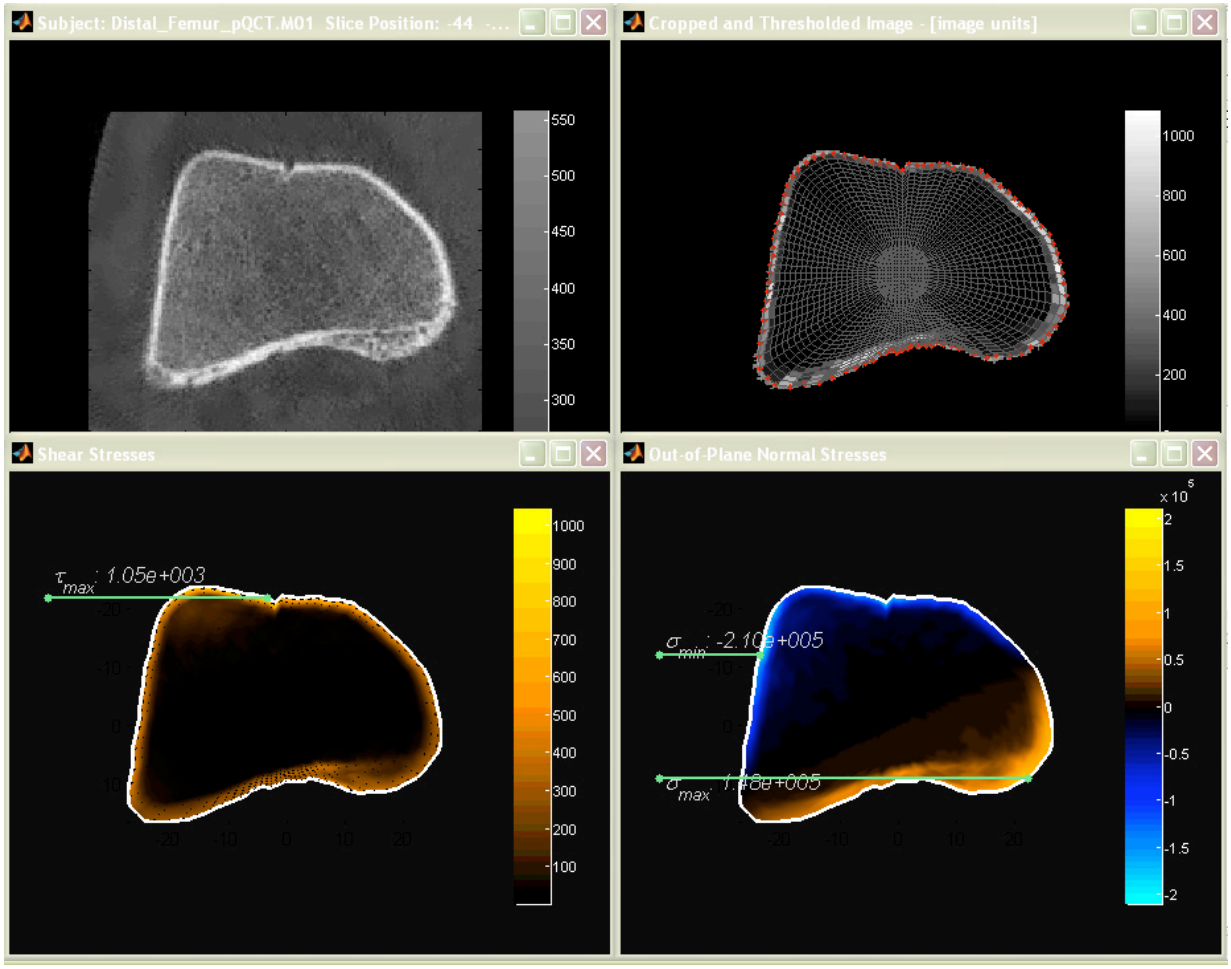
-At any time, adjust the threshold value by clicking on the colorbar on the **left** image. Higher values will discard more pixels and vice versa. A red cross indicates the weighted centroid of the selected object.

-At any time click at the center of the bone (**left** window) in case previous selections have invalidated the original selection.

-At any time, if necessary, edit the control points of the contour spline. This is done on the **right** window by left clicking on one node and dragging it to the new location. This will cause a group of nodes to move along, the number of nodes that are influenced depends on the local curvature. If the user wants to only move a single node, the CTRL key should be pressed down and then a regular click and drag of the node should follow. The user is advised to use this option when the thresholding option have been exhausted to save time and effort.

-Once the contour spline satisfactorily prescribes the bone, then **right** click in the **left** window confirms the selections and allows the program to continue to the analysis step.

-From now on there is no user interference, the program will calculate and report the cross sectional properties as well as the stresses (in MPa) given the load case. Here is what the result looks like:



The program also reports the sectional properties such as the moments of inertia etc. but also calculates and prints the Torque that the cross section is supporting. The user is advised to look into the last line of the main.m routine to find out what other metrics are calculated.

Example 2

Open MATLAB. Go to the folder where VA-BATTS is stored.

Open main.m and edit the first section where the loads are specified. In this case we are going to capture an instance of the gait cycle and the stresses it creates on the midshaft of the femur. The cross section under investigation is at 50%. The moments are 120Nm anteroposteriorly and

20Nm mediolaterally while the axial load is 1000N. An internal rotation of 2e-4rad/mm is applied (we will see later that this displacement generates about 30Nm torque). The first section of main.m should then read:

```
%% 0. INPUT

% Flexural Loads
Mx              = 120000;         %in N.mm
My              = 20000;          %in N.mm
% OR (3pt bending)
Qx              = 0;              %in N
Qy              = 0;              %in N
z_pos           = 0;              %in mm
% Axial Loads
P               = 1000;           %in N

% Torsion displacement
theta           = 2e-4;           %in rad/mm
```

The mesh density can also be defined by specifying:

```
%% Model parameters
nodes_r  = 12                     % number of nodes in the radial direction
nodes_th = 64;                    % number of nodes along the circumference
```

Also, the conversion relation (image units to density units should be reviewed. Open the image2density.m routine. Depending on the calibration (this is a 120kVp scan), the routine should read:

```
% DICOM GE CT parameters
A=A-1000;
a=0.000521
b=-0.05
%% Convert
A_d=bin_A.*abs(A*a + b*ones(size_x,size_y)); %in gr/cm3
```

A is the image variable and *A_bin* is the binary blob of the segmented image. These parameters have been estimated experimentally at the Palo Alto VA Hospital R&D Center by also taking into account NIST data. The user is advised to enter his/her own conversion relation.

-Run main.m by either typing main or by opening main.m and by clicking on the Run button; the file selection dialogue appears.

-Use one of the files included in the package, select file “Midshaft_Femur_CT.dcm” and click Open. The image opens. This is a DICOM image.

-Crop the image so that it shows only the region of interest (ROI) by dragging the mouse (left click and hold – drag - release) and selecting a rectangular window.

-The mesh type selection dialogue appears asking whether the mesh type should be solid or donut. Select donut as the epiphyseal cross section is non-hollow.

-The program then calculates the allowable thresholding levels.

-Click (left) on a point inside the bone (in the marrow channel) to indicate which bone is of interest. In cases where there are more than one bones present in the image, this will tell the program which one is to be analyzed. A second, version of the image appears on the right; the bone is contoured by a red spline whereas the intramedullary canal is contoured by a blue spline.

Again, all mouse clicks are to be done while in the **left** window unless editing of the contour control points is required; this is done in the **right** window.

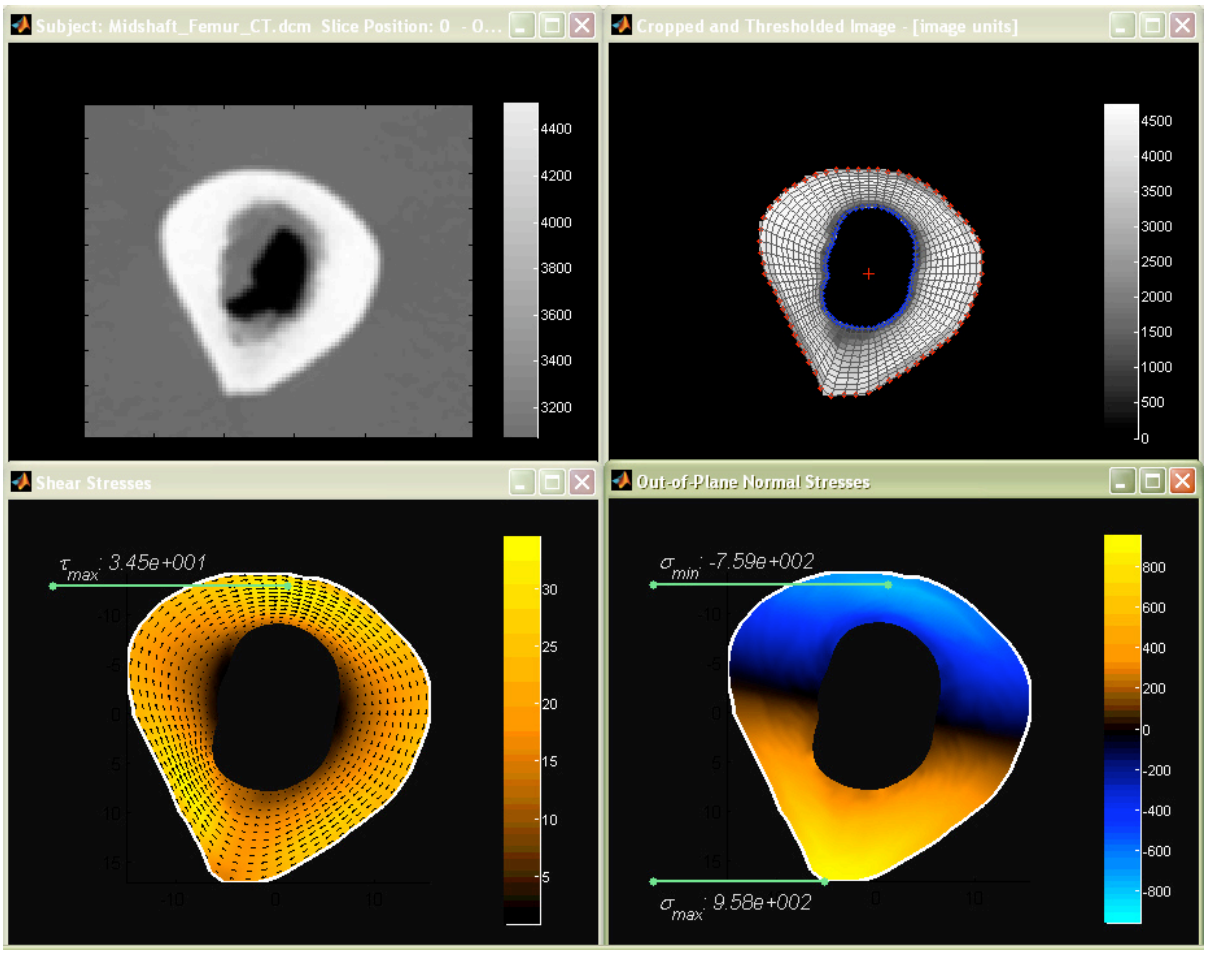
-At any time, adjust the threshold value by clicking on the colorbar on the **left** image. The range of selection is such so that the bone will always successfully be segmented. Higher values will "thin" the selection whereas lower values will thicken the selection, sometimes with additional unwanted soft tissue. A red cross points the weighted centroid of the selected object.

-At any time click at the center of the bone (**left** window), inside the hollow portion of the bone in case previous selections have invalidated the original selection.

-At any time, if necessary, edit the control points of the contour spline. This is done on the **right** window by left clicking on one node and dragging it to the new location. If the user wants to only move a single node, the CTRL key should be pressed down and then a regular click and drag of the node should follow.

-Once the contour splines are satisfactory, then **right** click in the **left** window confirms the selections and allows the program to continue to the analysis step.

-From now on there is no user interference, the program will calculate and report the cross sectional properties as well as the stresses (in MPa) given the load case (see figure)



- With stresses calculated, the user can try plotting contours that result from operations between stress components. For example, type:

```
bending_stress_tissue = bending_stress./((rho./1.92).^2);
```

to convert from continuum level stresses into tissue level stresses. And then plot the result using:

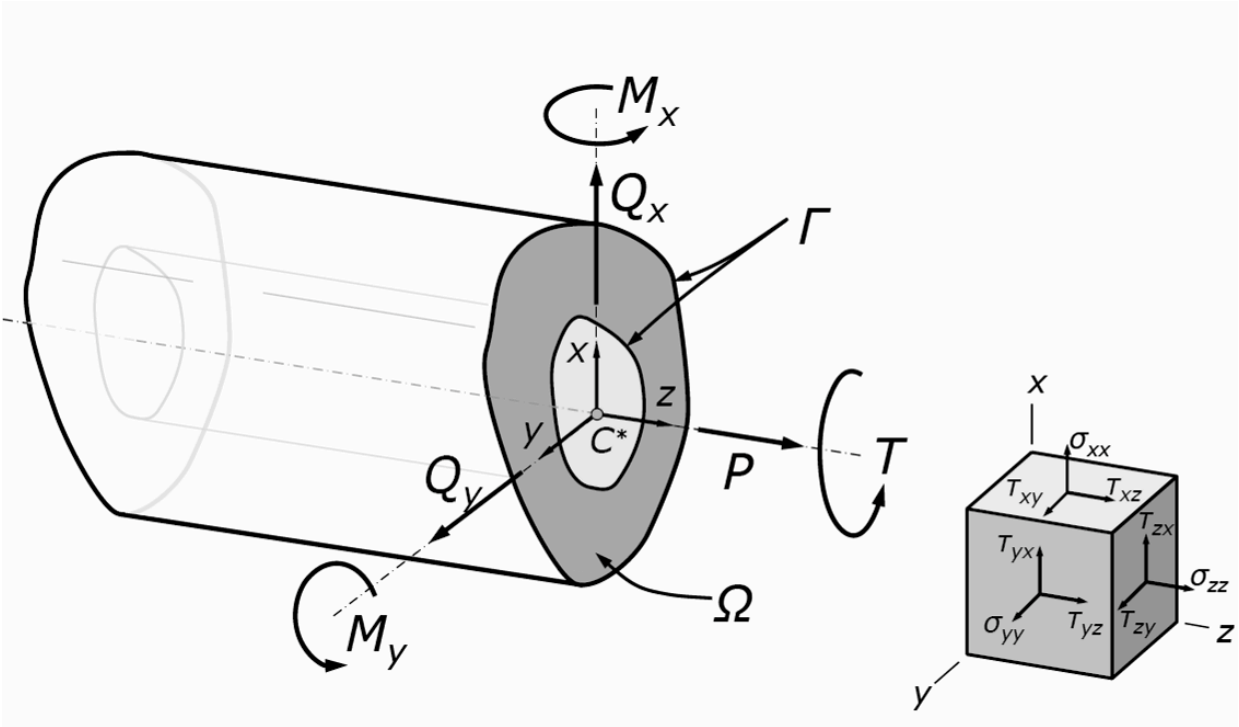
```
plot_contour(node_real, element, node_ext_cart, node_int_cart, bending_stress_tissue,
5, 'Tissue Level Normal Stresses' , [1 2], 'sigma');
```

4. Description of the routines

Please note that the **bold portions** of this manual require special attention. VA-BATTS users should first check these points before running an analysis. VA-BATTS require the involvement of the user and welcomes suggestions for improvement. Most importantly, VA-BATTS is an open program that can be adapted to a number of needs. Some of the routines here are simply specific implementations, other versions can also be implemented.

main.m

This is the main routine that makes the calls for all other subroutines. In the first part, the loads that are going to be applied to the beam are described. Below is a sketch demonstrating the



coordinate system of the bone as well as the possible applied loads.

Although other options exist, the user is advised to initially start with the following input values:

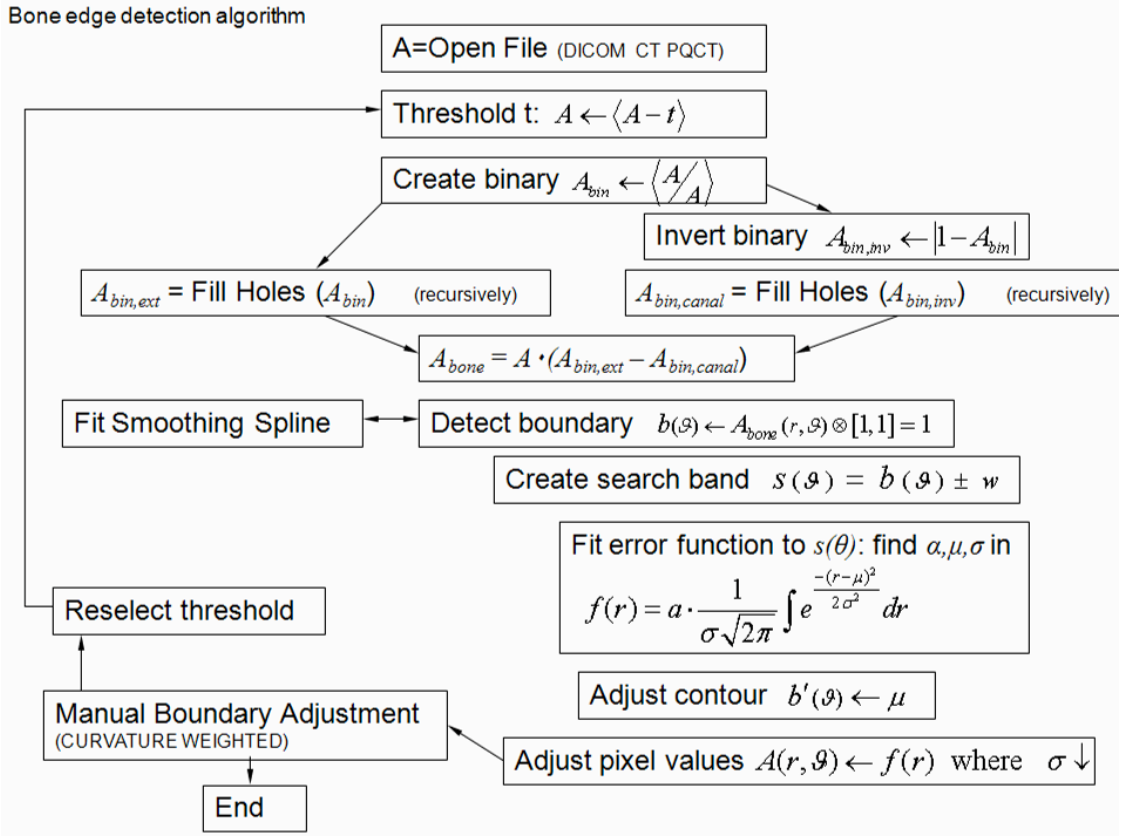
```
%% INPUT

% Flexural Loads
% Moments
Mx              = 10            %in N.mm
My              = 0;            %in N.mm
% or Transverse Loads
Qx              = 0;            %in N
Qy              = 0;            %in N
Z_pos           = 0;            %in mm

% Axial Loads
P               = 0;            %in N

% Torsional displacement
Angle           = 0;            %in degrees/mm
…
nodes_r  = 9;    % number of mesh nodes in the radial direction
nodes_th = 48;   % number of mesh nodes around the circumference
```

No further typed information/input is required for the program to run (except for the image to density conversion). Note that either the moments (M_x , M_y) or the transverse loads along with the moment arm (Q_x , Q_y , z) are supplied, the user has to set the others to zero.

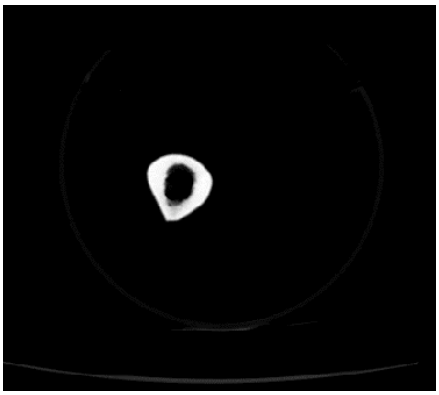
Next, step 1 mainly deals with importing the image data, step 2 deals with image processing, step 3 specifies the region of interest, and segments the bone. Step 4 creates the mesh, and steps 5 assigns material properties to the mesh that depend on the image data. Step 6 is responsible for calculating the cross sectional properties such as moment of inertia, etc., and finally steps 7 & 8 solve the problem; step 7 is a finite element approach to find shear stresses, step 8 follows beam bending theory.

Following, the subroutines called by main.m are presented in a logical order.

The first section of the program is mainly image processing. Here is a logical diagram of this section:



read_section_image.m



This routine is responsible for reading in the image data. Depending on the format of the image file (DICOM or pQCT), the appropriate lines are called. For DICOM, Matlab's native functions (dicominfo, dicomread) are used. For pQCT format (STRATEC, Germany), another routine (parse_pQCT.m) is employed that also applies a 4x4 wiener image filter to reduce noise. In both cases, the pixel size (in mm) of the image is. Optionally, the slice location is read in case of multiple section analysis. The image is then cropped by the user so that only the bone of interest is shown. Also, the pixel size is read and stored in *resolution* for later use in the analysis.

There exist conversion relations that are applied to the image to make it VA-BATTS compatible**. It is very important for the user to review this routine and decide whether the units are appropriate based on his data. Routine image2density.m is responsible for the conversion from image units to bone apparent density units (g/cm^3). The user needs to make sure that this conversion pertains to his/her application.**

determine_mesh_type.m

This routine opens a dialogue ere the user selects the type of the mesh, **donut** for hollow cross sections (midshaft for example) and **solid** for non-hollow sections (epiphysis for example).

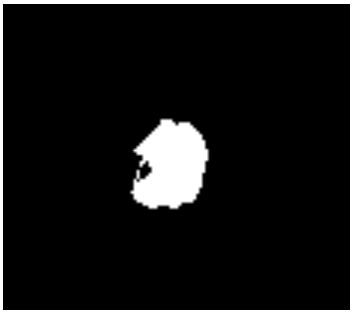
threshold_image.m

This routine simply thresholds the image *A* based on an interactive value supplied by the user by clicking on the colorbar to the left of the original scan image. The routine returns both the thresholded image and the binary map (see right image) of the segmented bone.

interactive_thresholding.m

Medical images may contain various sections that are not of interest for the VA-BATTS analysis. Therefore, the actual bone of interest needs to be centered and a new region of interest (ROI) needs to be defined.

The routine creates a blob by region growth (starting point: user clicked inside the bone). Another blob is created by inverting the image (intramedullary canal, see right image) which then is subtracted from the original blob. A binary mask of the bone (*A_bin*) is returned which is then multiplied point-by-point with the original image to yield the thresholded image. An error may occur if the threshold value selected does not produce a blob, that is if the level is to high and all bone is segmented out, there is no blob for further evaluation created.

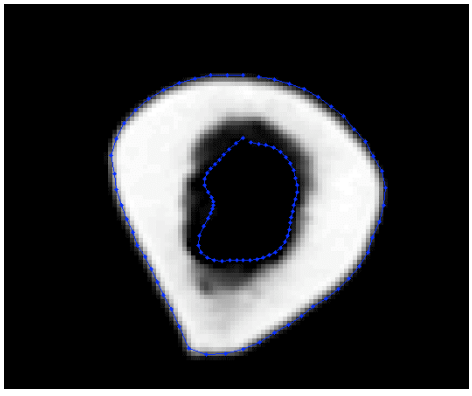




find_center.m

This routine calculates the image intensity weighted center of the bone (variable *center*). It is a first approximation of the image intensity weighted centroid, this will be recalculated again later as soon as the material properties are assigned:

$$(C_x^*, C_y^*) = \left(\frac{\int xE(x,y)dA}{\int E(x,y)dA}, \frac{\int yE(x,y)dA}{\int E(x,y)dA} \right)$$

detect_boundary.m



This routine, first creates a pixilated perimeter of the bone based on the blob (*A_bin*) supplied by interactive_thresholding.m. It then fits a spline over those boundary points. This is repeated for both the outside and the inside (in the case of a hollow bone) contours. It returns the coordinates of the spline control points of the external (ext) and internal (int) contours in a polar (*node_ext_polar, node_int_polar*) and in a cartesian (*node_ext_cart, node_int_cart*) coordinate system based on the center supplied by *center*. This is the first approximation of the bone geometry. The number of control points is defined by *nodes_th* in the beginning of main.m and can be

adjusted (typically 48 to 96 points). These points are going to be the seed nodes for the cross section mesh.

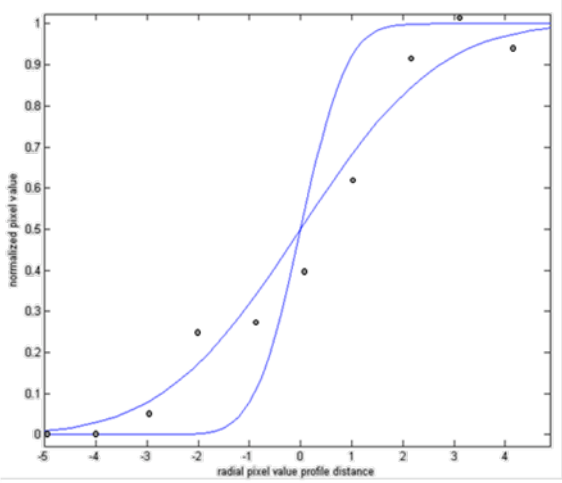
find_bounding_box.m

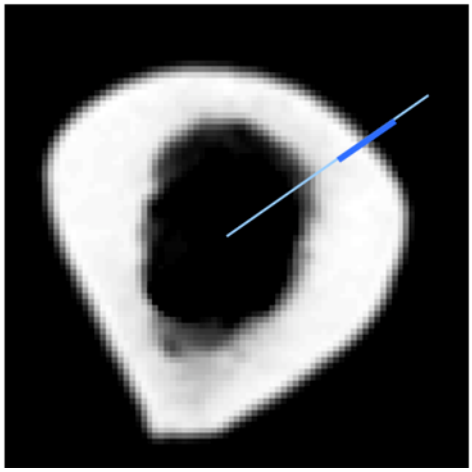
This routine simply finds a box that inscribes the bone. The size of the box is passed on *x_dimension, y_dimension.*

refine_boudary.m

This elaborate routine takes the control points given by detect_boundary.m and tries to adjust them better so that partial volume effects are diminished. The routine performs a fit of the normal cumulative function on the values of the pixel that belong to the radial line (light blue) connecting the center (*center)* with each contour spline control point (*node_ext_polar)* across the length of a search (thick blue) zone (*zone_width*):

$$f(x) = a \cdot \frac{1}{\sigma\sqrt{2\pi}} \int e^{\frac{-(x-\mu)^2}{2\sigma^2}} dx$$



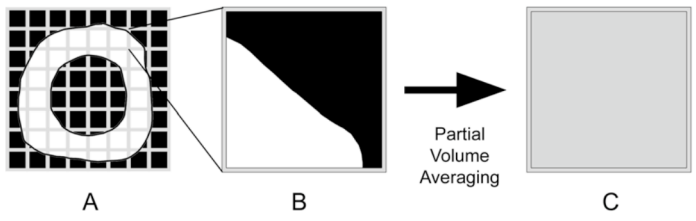


As shown in the plot, the dots represent the pixel values across the search zone. The graph represents a fit of those data points by the cumulative (error) function. This function is described in fun.m

Once the function has been fitted, the μ and σ of the distribution are found and the control point that represents the edge of the bone is adjusted appropriately. Finally an updated vector containing the adjusted bone boundary locations is returned (*node_ext_polar)* .

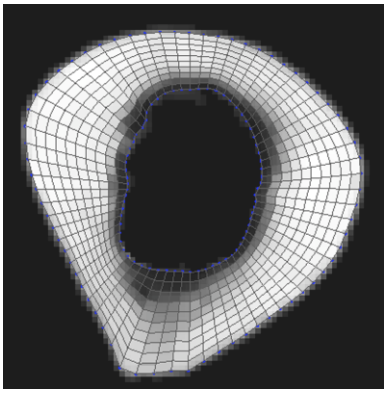
partial_volume _correction.m **(not available in this version)**

This subroutine makes sure that there is no "lost matter" during segmentation. Inevitably, tomographic imaging modalities produce blurry images, a phenomenon known as partial volume averaging (see graph below). Since a line is drawn along a gray area of the bone, there will be attenuated energy (that corresponds to bone) that will not be taken into account. This routine makes sure that whatever is left "outside" the line is returned on the "inside" of the line producing thus a sharper image. This routine however takes a long time to run therefore the user can chose not to use it by selecting **partial_vol = false** in the beginning of the main.m program.



create_mesh.m

This routine practically takes the control points of the splines (seed nodes) that perscribe the bone and uses them as seed points for the mesh. Depending on the type of the bone two different mesh generators are triggered: for a hollow bone (*canal_flag=1*) a donut shaped mesh is created (see left image) while for a solid bone, a butterfly mesh is generated (see right image).



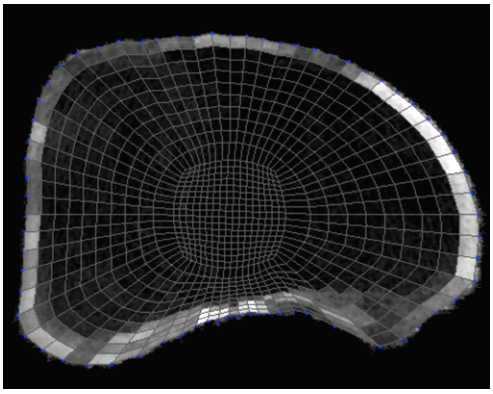
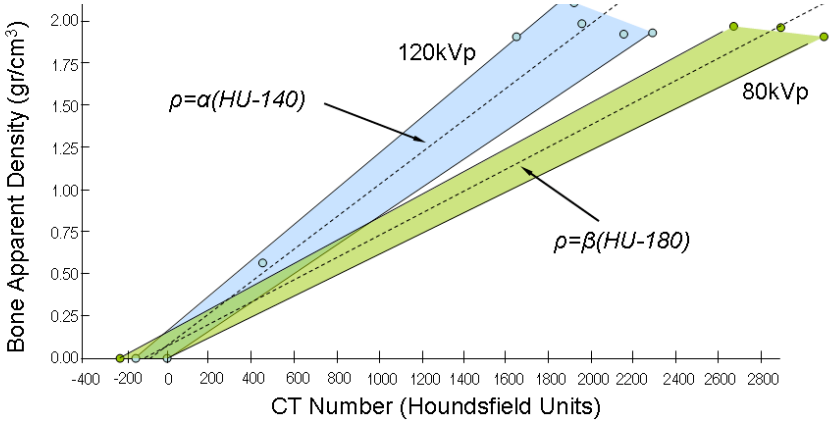


image2density.m

This is a critical routine that requires the user's inspection before running an analysis. **The image *A* that contains the HU values for each voxel, is transformed into an apparent density ρ_{app} [g/cm^3] image (*A_d*).**



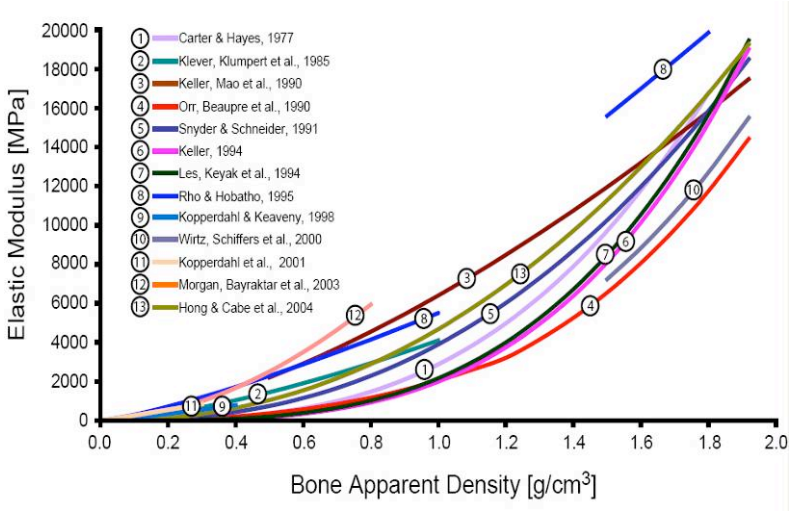
There needs to exist a relation that links the image units (for example HU at a certain energy) to ρ_{app}. VA-BATTS is using a linear relation to do this, but the users can use any other relation might come up while callibrating their scanner using for example a callibration phantom or NIST data. The plot presents such relations for different scanning energies (80kVp; 120kVp) that where used with VA-BATTS. **The user has to make sure that the relation given in this routine fits his/her parameters.**

Relations are usually of the form $\rho = \beta + \alpha \cdot \mu$ where μ is the image intensity (can be HU or linear attenuation coefficient, or hydroxyapatite-mineral content values etc.). In this routine parameters *α, β* are defined here (*A* is the original image in the modality's units):

```
a=2.2092;
b=-0.5852;
A_d=bin_A.*abs(A*a + b*ones(size_x,size_y)); %in gr/cm3
```

density2elastic_props

This is another critical routine that requires the user's inspection before running an analysis. **After the original units image (*A*) has been converted to an aparent density image (*A_d*), the elastic modulus *E* [MPa] needs to be assigned. The image is therefore converted into an elastic modulus image (*A_YM*) and a shear modulus image (*A_G*) based on a relation. There exist many such relations to convert from bone apparent density to elastic modulus in the litterature (see plot). VA-BATTS is using the Keller 1994 relation but users may want to select a different one. Once the elastic modulus image is created another relation is used to create a shear modulus [MPa] image (*A_G*). VA-BATTS is using the Reilly & Burstein 1976 relation but users may want to select a different one.**



In general these relations are of the form: $E = \alpha \cdot \rho^b$ that is a linear term and an exponential term. The following is describing the Keller 1994 relation. Users can change this according to another reference or their own calibration data. In VA_BATTS *E* is in MPa and ρ is in g/cm^3

```
exp1=3.46;
lin_term1=1990;
A_YM = lin_term1 * A_d.^exp1 ;
```

assign_material.m

This routine is responsible for assigning propery values to each element that has been created by create_mesh.m based on the underlying image values. A uniform grid is created in the natural (element) space; for each element, linear shape functions are employed to map the grid points to the physical space. These points are then used as sampling points. A typical grid is composed of 8x8 points that when mapped yields 64 sampling points on the image for an element. An intuitive method in order to determine a single set of elastic property values for

each element is to average the sampling point values over the area or volume of the continuum. However, in cases of increased partial volume averaging where the bone cross section contour is not sharp, averaging might result in underestimation of the elastic modulus for the elements at the bone boundary. In this case it useful to discard the lowest portion (e.g. lower 50%) of the sampling point values, and then average over the remaining point values. The output arguments are material property vectors, that is for example *YM()* contains the elastic modulus in MPa values of the elements, *rho()* contains the density values in g/cm^3 of the elements etc. Other properties are: area, shear modulus, Poisson's ratio, and centroid of each element.

This routine concludes the image processing and meshing parts. The next routines involve the cross sectional property calculation and the stress calculations.

section_properties.m

This routine calculates the various metrics of the cross section. More specifically and since elements have been defined and have been assigned material properties the modulus weighted centroid of the cross section:

$$\left(C_x^*, C_y^*\right) = \left(\frac{\sum_{i=1..nel} x_i A_i E_i}{\sum_{i=1..nel} A_i E_i}, \frac{\sum_{i=1..nel} y_i A_i E_i}{\sum_{i=1..nel} A_i E_i} \right),$$

where *A* is the area of each element in mm^2, *E* is the elastic modulus in MPa assigned to that element, and *x, y* are the coordinates in mm of the centroid of the element. Also the moments of inertia are determined:

$$I_{xx}^* = \sum_{i=1}^{nel} \left(\left(I_{xx}^e E \right)_i + y_i^{\,2} A_i E_i \right), \quad I_{yy}^* = \sum_{i=1}^{nel} \left(\left(I_{yy}^e E \right)_i + x_i^{\,2} A_i E_i \right), \quad I_{xy}^* = \sum_{i=1}^{nel} \left(\left(I_{xy}^e E \right)_i + x_i y_i A_i E_i \right).$$

In addition the principal axes of the section are also found as the eigenvectors of : $\begin{vmatrix} I_{xx} & I_{xy} \\ I_{xy} & I_{yy} \end{vmatrix}$

The routine argument reads:

```
function [I, J, I_principal, V,   I_E, J_E, I_E_principal, V_E,   I_G, J_G,
I_G_principal, V_G] = section_properties (YM, G, rho, centroid, area, center_mesh,
center_mesh_E)
```

The first block of output variables is the geometric moment of inertia tensor (`I`), the polar moment of inertia (`J`), the principal moment of inertia tensor (`I_principal`) and the principal moment of inertia vectors (`V`). This calculation takes into account the geometric centroid (`center_mesh`) as $\left(C_x^*, C_y^*\right)$ where E_i's are omitted in the calculation.

The second block of output variables is the elastic modulus weighted moment of inertia tensor (`I_E`), the polar moment of inertia (`J_E`), the principal moment of inertia tensor (`I_principal_E`) and the principal moment of inertia vectors (`V_E`). This calculation takes into account the modulus weighted centroid (`center_mesh_E`) as $\left(C_x^*, C_y^*\right)$ in the calculation.

Similarly the last block of output variables is the shear modulus weighted quantities presented above.

The user is advised to employ the appropriate quantities for their applications.

This is an important routine as these metrics are used at subsequent steps to calculate stresses.

calculate_bending.m

This simple routine employs beam theory to calculate the normal stresses throughout the cross section. Calculations take place only at the centroid locations and then are averaged across the element boundaries to produce continuous plots. Stresses are found using:

$$\sigma_{zz} = \left(\frac{Q_x I^*_{xx} - Q_y I^*_{xy}}{I^*_{xx} I^*_{yy} - I^{*\,2}_{xy}} E_i x_i + \frac{Q_y I^*_{yy} - Q_x I^*_{xy}}{I^*_{xx} I^*_{yy} - I^{*\,2}_{xy}} E_i y_i \right) \cdot z + \frac{E_i P}{A^*},$$

where Q_x, Q_y are the transverse loads in N, z is the moment arm, P is the axial load in N, and A^* is given by:

$$A^* = \int E(x,y)dA \quad = \sum_{elem} A_i E_i$$

A similar formula is used if moments (M_x, M_y) are directly given. The maximum and minimum stress values are found and are pointed out on the plots.

The routine is used as :

```
[bending_stress] = calculate_bending (Mx, My, Qx, Qy, z, P, I, node, area, centroid,
YM, center)
```

where `Mx` [N.mm], `My` [N.mm], `Qx` [N], `Qy` [N], `z` [mm], `P` [N] are the loads (note that either `Mx`, `My` or `Qx`, `Qy`, `z` are to be supplied). `I` is the moment of inertia matrix, `node` is the node location matrix, `area`, `centroid`, `YM` are vectors containing the elements area, centroid and elastic moduli. `I` **is the geometric moment of inertia tensor, alternatively for inhomogenous material properties, the modulus weighted moment of inertia** `I_E` **tensor needs to be employed.**

`bending_stress` is a vector that contains the normal stresses at the element centroids:

$$\begin{vmatrix} \sigma_{zz}^{\,1} \\ \sigma_{zz}^{\,2} \\ \dots \end{vmatrix}$$ where superscript signifies the element number.

Users can use function elemental2nodal.m to convert element values to nodal values.

solve_FE_problem

This is the key source routine of VA-BATTS. The user is advised to refer to Kourtis L.C. et al 2008 JoMMS to find a thorough description of the methods employed here. In short the problems of torsional shear and transverse shear are combined into one strong form:

$$(\mathrm{S}): \text{Given } \alpha,\beta,\theta \text{ and } G(x,y) \,\&\, E(x,y), \text{ find } \Xi \text{ in } C^2 \text{ such that}$$

$$\begin{aligned} (G\Xi_{,i})_{,i} &= f(x,y) \text{ on } \Omega \\ G\Xi_{,i}n_i &= Gt_in_i \text{ on } \Gamma \end{aligned} \qquad \text{where,}$$

$$f(x,y) = 2\nu xy(\beta G_{,x}+\alpha G_{,y}) - (E-2G\nu)(\alpha x+\beta y)+\theta(G_{,x}\,y - G_{,y}\,x)$$

$$\mathbf{t} = \begin{bmatrix} (\theta y + 2\beta\nu xy) \\ (-\theta x + 2\alpha\nu xy) \end{bmatrix}$$

where E, G are the elastic and shear moduli in MPa, v is the Poisson's ratio, θ is the linear angular twist (in rad/mm), and α, β are given by:

$$\alpha = \frac{Q_x I^*_{xx} - Q_y I^*_{xy}}{I^*_{xx} I^*_{yy} - I^{*\,2}_{xy}} \text{ and } \beta = \frac{Q_y I^*_{yy} - Q_x I^*_{xy}}{I^*_{xx} I^*_{yy} - I^{*\,2}_{xy}}$$

The strong form is converted to a weak form:

$$(\Omega): \text{Given } \alpha,\beta,\theta \text{ and } G(x,y) \,\&\, E(x,y), \text{ find } \Xi \text{ in } \Upsilon \text{ such that}$$

$$\int_\Omega w_{,i}\, G\Xi_{,i}\, d\Omega = \int_\Gamma w G t_i n_i \, d\Gamma - \int_\Omega w f(x,y)\, d\Omega$$

which then by Introducing the finite element function space and element shape functions becomes,

$$
\begin{aligned}
k^e_{ab} &= \int_{\Omega^e} N^e_{a,i} G^e N^e_{b,i}\, d\Omega^e \\
f^e_a &= \left(f^f_a\right) + \left(f^g_a\right) \\
\left(f^f_a\right) &= -\int_{\Omega^e} N^e_a f(x,y)\, d\Omega^e \\
\left(f^g_a\right) &= \int_{\Gamma^e} N^e_a G t_i n_i \, d\Gamma^e \\
f(x,y) &= 2\nu\, x\, y\,(\beta\, G_{,x} + \alpha\, G_{,y}) + (2G\nu - E)(\alpha\, x + \beta\, y) + \theta(G_{,x} y - G_{,y} x) \\
\mathbf{t} &= \begin{bmatrix} (\theta\, y + 2\beta\nu\, x\, y) \\ (-\theta\, x + 2\alpha\nu\, x\, y) \end{bmatrix}
\end{aligned}
$$

The gradients of the shear modulus can be calculated using the shape function derivatives:

$$G_{,x} = \int N_{a,x} G^n d\Omega^e$$

The results here are calculated for each integration point, the values are then extrapolated to the nodes, they are averaged across shared nodes and then reassigned and interpolated to the internal of the elements.

The usage of the routine is as follows:

```
[shear_stress, resultant_shear_stress] = solve_FE_problem (node,element, YM, G, v, I,
Qx, Qy, theta);
```

Where node is the node location matrix, element is the element connectivity matrix, YM, G, v are vectors containing the elastic properties of the elements, I is the moment f inertia matrix, and Qx, Qy, theta are user supplied loads (see plot in the beginning).

`shear_stress` is a vector that contains the shear stress components at the element centroids.

$\begin{vmatrix} \tau_{zx}^{\ 1} & \tau_{zy}^{\ 1} \\ \tau_{zx}^{\ 2} & \tau_{zy}^{\ 2} \\ \dots & \dots \end{vmatrix}$ where superscript signifies the element number.

`resultant_shear_stress_e` is a vector that contains the resultant shear stresses at the elements as given by : $\tau_{res} = \sqrt{\tau_{zx}^{\ 2} + \tau_{zy}^{\ 2}}$

Also, these values (elemental: `resultant_shear_stress_e`) are converted to nodal values (`resultant_shear_stress`) by using:

```
resultant_shear_stress = elemental2nodal (resultant_shear_stress_e, node, element);
```

This routine concludes the stress calculation step.

A Note on Torsion

VA-BATTS uses a torsional displacement to calculate stresses by employing the Saint Venant theorem. The boundary condition therefore is prescribed as a displacement. In order to find the torque that is taken by the cross section, the shear stresses are integrated and at the end of the analysis the torque is reported. The user can find out the torque applied by looking at the results or by typing Torque.

Post Processing and Plotting

The user can now select a stress (failure) criterion to plot results, by combining the principal and shear stresses. VA-BATTS plots the tissue level stresses that are obtained by dividing the actual continuum level stresses by the apparent density squared.

```
bending_stress = bending_stress./((rho./1.92).^2);
```

Note that 1.92 is the value that is considered as the highest bone density possible for which continuum level stresses are equal to tissue level stresses, which is a valid assumption.

However this calculation can be omitted, if continuum level stresses are sought.

VA-BATTS users are encouraged to come up with their own stress plotting criteria, failure criteria etc. that combine normal and shear stresses. When this is done the result can be viewed by using:

```
plot_contour(node, element, node_ext_cart, node_int_cart, resultant_stress,
figure_number, 'Plot Title' , [1 2], 'sigma or tau');
```

where the only thing that needs to be supplied is the stresses `[resultant_stress]` (a 1xN vector) or in general any kind of metric the user wishes to plot.

For example, users may use an elliptical criterion to plot the potential of the cross section:

```
plot_contour(node_real, element, node_ext_cart, node_int_cart,...
sqrt((bending_stress/failure_normal_stress).^2 +
(shear_stress/failure_shear_stress).^2),...
 figure_number, 'Stress Potential' , [1 2], 'sigma');
```

where failure_normal_stress and failure_shear _stress are material constants (such as given by Carter & Hayes 1976)

List of output variables:

After running the program simply type the name of the variable or a relation that contains one or more of them, or even plot using the plot_contour command syntax. To plot, use either centroid or nodal values (the routine detects automatically). However, to perform mathematical operations between different variables, they need to be either all centroidal or nodal – use the following routine to convert.

```
nodal_value = elemental2nodal (element_value)
```

Also to find maximum values use max(variable) or min(variable).

Nomenclature: N_elem is the number of elements in the model, N_node is the number of nodes in the model, nodes_th is a user input (see heading of main.m) and specifies the mesh density, exactly how many nodes are seeded along the perimeter of the bone.

Variable Name	Description	Size
A	The cropped, thresholded image [in image units, e.g. HU]	image size
A_YM	The elastic modulus scaled image [in MPa]	image size
A_G	The shear modulus scaled image [in MPa]	image size
A_d	The density scaled image [in g/cm^3]	image size
resolution	Image (user) input. Automatically determined, in [mm/pixel]	1 x 2
nodes_ext_cart nodes_int_cart	The locations of the control points of the splines that segment the bone. They are also the mesh seed points.	nodes_th x 2

element	The connectivity matrix of the mesh	N_elem x 4
node	The node definitions (x, y pixel coordinates)	N_node x 2
node_real	The node definitions (x, y real coordinates in mm)	N_node x 2
centroid	The coordinates of the element centroids	N_elemx 2
center_mesh_E	The modulus weighted centroid of the mesh	2 x 1
I	The geometric moment of inertia tensor $\begin{vmatrix} Ixx & Ixy \\ Ixy & Iyy \end{vmatrix}$ [in mm^4]	2 x 2
I_E	The modulus weighted moment of inertia tensor [in mm^4]	2 x 2
J	The polar moment of inertia [in mm^4]	1 x 1
I_principal	The principal geometric moment of inertia tensor	2 x 2
V	The eigenvectors of the geometric principal directions	2 x 2
I_E_principal	The modulus weighted principal moment of inertia tensor	2 x 2
V_E	The eigenvectors of the modulus weighted principal directions	2 x 2
area	The area of each element	N_elem x 1
rho	The assigned apparent density of each element	N_elem x 1
YM	The assigned elastic modulus of each element	N_elem x 1
G	The assigned shear modulus of each element	N_elem x 1
bending_stress	The bending stresses evaluated at the centroids σ_{zz}	N_elem x 1
shear_stress	The shear stress components τ_{zx}, τ_{zy} at the nodes	N_node x 2
resultant_stress	The magnitude of the shear stress at the nodes	N_node x 1
resultant_stress_e	The magnitude of the shear stress at the centroids	N_elem x 1

Palo Alto Veteran Affairs Hospital, Rehabilitation Research and Development Center.

Stanford Biomechanical Engineering Division. Stanford University