

# Manual of GEBT\*

Wenbin Yu<sup>†</sup>

April 15, 2011

## 1 Introduction

GEBT (*Geometrically Exact Beam Theory*) is a code implementing the mixed variational formulation of the geometrically exact intrinsic beam theory developed by Prof. Hodges of Georgia Institute of Technology<sup>1,2,3,4,5</sup> which captures all the geometrical nonlinearities obtainable by a beam model. It is a companion code of VABS, a general-purpose cross-sectional analysis tool, to enable efficient yet high-fidelity analysis of slender structures, whether they are made of composite materials or not.

For effective design space explorations, we need to simplify the original nonlinear three-dimensional (3D) analysis of slender structures into a two-dimensional (2D) cross-sectional analysis and a one-dimensional (1D) nonlinear beam analysis. This has been achieved using VABS along with GEBT.<sup>6</sup> VABS takes a finite element mesh of the cross-section including all the details of geometry and material as inputs to calculate the sectional properties including structural and inertial properties. These properties are needed for GEBT to predict the global behavior of the slender structure. The 3D pointwise displacement/strain/stress distribution within the structure can also be recovered by providing outputs from GEBT into VABS.

Since most of the theoretical details are presented in pertinent papers and collected in the book by Prof. Hodges,<sup>5</sup> this manual will only serve to help readers get started using GEBT to solve their own beam problems. This manual addresses the history of the code, its features, functionalities, conventions, inputs, outputs, maintenance, and tech support.

## 2 GEBT History

GEBT was initiated by Prof. Wenbin Yu at Wright-Patterson Air Force Base (AFRL/RB) when he was supported as a Faculty Fellow for the 2008 Air Force Summer Faculty Fellowship Program. Dr. Maxwell Blair of AFRL/RBSD has been a very strong advocate of this code due to its potential

---

\*GEBT is copyrighted by Utah State University. All rights reserved.

<sup>†</sup>Associate Professor, Department of Mechanical and Aerospace, Engineering, Utah State University, Logan, Utah 84322-4130.

application in USAF applications with highly flexible structural components. The code is written in Fortran 90/95 and designed for straightforward integration with other codes. The code is a free-ware for anybody who requests. The Alpha version was released in July 2008. The Beta version was released in June 2009. GEBT 2.0 was released in November 2009. GEBT 3.0 was released in February 2010. GEBT 4.0 was released in December 2010. The most recent version is GEBT 4.2 which was released in March 2011.

### **3 GEBT Features**

GEBT adopts the mixed variational formulation of the geometrically exact beam theory of Prof. Hodges. It uses the lowest possible shape functions and the element matrices are calculated exactly without numerical integration. Since it is a mixed variational formulation, the complete set of variables can be directly calculated. For example, for static analysis, we can obtain three displacements, three rotations, three forces, and three moments. GEBT can treat an arbitrary assembly of beams made of arbitrary material and oriented arbitrarily in the 3D space. GEBT uses dynamic link libraries (DLLs) to encapsulate the analysis capability so that it has true plug-n-play capability which is convenient for integration into other environments. Now GEBT can be used both as a standalone application and a callable library. A GEBT manual for developers can be requested separately.

### **4 GEBT Functionalities**

The most recent version of GEBT, GEBT 4.0, has the following functionalities:

1. Static, both linear and nonlinear, analysis of beam assemblies with straight and/or initially curved/twisted members under prescribed concentrated or distributed forces/moments or displacements/rotations.
2. Static, both linear and nonlinear, analysis of beam assemblies with straight members and/or initially curved/twisted with linearly varying sectional properties.
3. Static, nonlinear, analysis of beam assemblies with straight and/or initially curved/twisted members under prescribed distributed or concentrated follower forces/moments.
4. Steady state of the dynamic response of beam assemblies with straight and/or initially curved/twisted members under prescribed distributed or concentrated dead or follower forces/moments.
5. Transient dynamic response of beam assemblies with straight and/or initially curved/twisted members under prescribed distributed or concentrated dead or follower forces/moments.
6. Sensitivities for the global beam responses for all the aforementioned analysis capabilities.

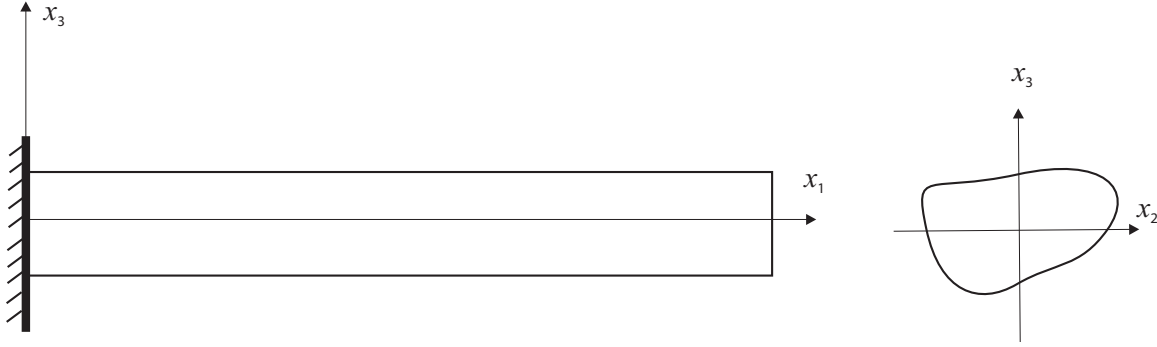


Figure 1: VABS beam coordinate system

## 5 GEBT Conventions

To understand the inputs and interpret outputs of the program correctly, we need to explain some conventions used by GEBT.

Firstly, GEBT uses a right-hand coordinate system, the beam coordinate system denoted as  $x_1, x_2$  and  $x_3$ , where  $x_1$  is along the beam axis and  $x_2$  and  $x_3$  are the local Cartesian coordinates of the cross section, see Figure 1 for a beam with an arbitrary cross section. Unit vectors  $\mathbf{b}_i$  are the corresponding base vectors. The cross-sectional coordinates ( $x_2$  and  $x_3$ ) are only needed for the purpose to define its orientation. It is noted that the beam coordinate system remains the same as the undeformed beam coordinate system defined in Ref. [5] for straight beam members. For curved member, the undeformed beam coordinate system is changing along the beam axis and the user needs to provide the undeformed beam coordinate system at the starting point of the beam member and the changing undeformed beam coordinate system along with the beam axis will be calculated automatically by the code. Usually, for a beam assembly, we define a global coordinate system, say with the triad  $\mathbf{a}_i$ , then the orientation of a beam member can relate to the global coordinate system using a direction cosine matrix

$$\begin{Bmatrix} \mathbf{a}_1 \\ \mathbf{a}_2 \\ \mathbf{a}_3 \end{Bmatrix} = \begin{bmatrix} C_{11} & C_{12} & C_{13} \\ C_{21} & C_{22} & C_{23} \\ C_{31} & C_{32} & C_{33} \end{bmatrix} \begin{Bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \mathbf{b}_3 \end{Bmatrix} = C^{ab} \begin{Bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \mathbf{b}_3 \end{Bmatrix} \quad (1)$$

Secondly, GEBT uses two points to specify each member with  $x_1$  of the member pointing to the second point. For dynamic analysis, there are 18 values for each element within a member arranged as

$$[u_1 \ u_2 \ u_3 \ \theta_1 \ \theta_2 \ \theta_3 \ F_1 \ F_2 \ F_3 \ M_1 \ M_2 \ M_3 \ P_1 \ P_2 \ P_3 \ H_1 \ H_2 \ H_3]^T \quad (2)$$

Thirdly, GEBT assumes that the cross-sectional properties have already been calculated by VABS and given in the form of a flexibility matrices and mass matrix. The flexibility matrix is

defined according to the following equation

$$\begin{pmatrix} \gamma_{11} \\ 2\gamma_{12} \\ 2\gamma_{13} \\ \kappa_1 \\ \kappa_2 \\ \kappa_3 \end{pmatrix} = \begin{bmatrix} S_{11} & S_{12} & S_{13} & S_{14} & S_{15} & S_{16} \\ S_{12} & S_{22} & S_{23} & S_{24} & S_{25} & S_{26} \\ S_{13} & S_{23} & S_{33} & S_{34} & S_{35} & S_{36} \\ S_{14} & S_{24} & S_{34} & S_{44} & S_{45} & S_{46} \\ S_{15} & S_{25} & S_{35} & S_{45} & S_{55} & S_{56} \\ S_{16} & S_{26} & S_{36} & S_{46} & S_{56} & S_{66} \end{bmatrix} \begin{pmatrix} F_1 \\ F_2 \\ F_3 \\ M_1 \\ M_2 \\ M_3 \end{pmatrix} \quad (3)$$

where  $\gamma_{11}$  is the beam axial stretching strain measure,  $2\gamma_{12}$  and  $2\gamma_{13}$  are the engineering transverse shear strains along  $x_2$  and  $x_3$  respectively,  $\kappa_1$  is the twist measure, and  $\kappa_2$  and  $\kappa_3$  are the curvature measures around  $x_2$  and  $x_3$  respectively. Note that any of the values in the flexibility matrix ( $S_{ij}$ ) can be zero to eliminate any of the deformation mechanism which the user believes to be negligible. For example, to analyze a pure bending around  $x_2$ , one can set  $S_{55}$  to be the bending flexibility and other terms to be zeroes.

The elements of the mass matrix are arranged as

$$\begin{bmatrix} \mu & 0 & 0 & 0 & \mu x_{m3} & -\mu x_{m2} \\ 0 & \mu & 0 & -\mu x_{m3} & 0 & 0 \\ 0 & 0 & \mu & \mu x_{m2} & 0 & 0 \\ 0 & -\mu x_{m3} & \mu x_{m2} & i_{22} + i_{33} & 0 & 0 \\ \mu x_{m3} & 0 & 0 & 0 & i_{22} & -i_{23} \\ -\mu x_{m2} & 0 & 0 & 0 & -i_{23} & i_{33} \end{bmatrix} \quad (4)$$

where  $\mu$  is mass per unit length,  $(x_{m2}, x_{m3})$  is the location of mass center,  $i_{22}$  is the mass moment of inertia about  $x_2$  axis,  $i_{33}$  is the mass moment of inertia about  $x_3$  axis,  $i_{23}$  is the product of inertia.

GEBT allows users to use various kinds of units. However, it is necessary to be absolutely consistent in the choice of units to avoid errors. Particularly, users must *never* use the pound as a unit of mass to avoid confusion. When pounds are used for force and feet for length, the unit of mass must be the slug = lb-sec<sup>2</sup>/ft; if inches are used for length along with pounds for force, then the unit of mass must be lb-sec<sup>2</sup>/in.

## 6 GEBT Inputs

GEBT has a build-in 1D finite element mesh generator, hence it is very easy to prepare the input file for a beam assembly. For users to understand the meaning of the input data, we explain each data entry line by line here.

The first line lists three analysis control parameters arranged as:

*analysis\_flag niter nstep*

where *analysis\_flag* can be 0, 1, 2, 3. If *analysis\_flag* is 0, GEBT will carry out a static analysis. If *analysis\_flag* is 1, GEBT will carry out a steady state analysis (*i.e.*, neglecting the time derivatives in the formulation). If *analysis\_flag* is 2, GEBT will carry out a time marching to obtain the transient time history of the dynamic response. If *analysis\_flag* is 3, GEBT will carry out an eigenvalue analysis to obtain the frequencies and mode shapes. *niter* is the maximum number of iterations for the nonlinear analysis. If *niter* is equal to 1, GEBT will compute results corresponding to a linear theory. *nstep* is the number of steps. For static analysis, it can be used to increase the load gradually to help the convergence. For dynamic analysis, it is the number of time steps.

If *analysis\_flag* is not equal to 0, the next two lines will be used to provide the angular velocity of the global frame *a* and the linear velocity of the starting pointing of the first member, and their corresponding time functions. They are arranged as:

$$\begin{array}{l} \omega_{a_1}, \omega_{a_2}, \omega_{a_3} \\ tf_{\omega_1}, tf_{\omega_2}, tf_{\omega_3} \\ v_{a_1}, v_{a_2}, v_{a_3} \\ tf_{v_1}, tf_{v_2}, tf_{v_3} \end{array}$$

where  $\omega_{a_1}, \omega_{a_2}, \omega_{a_3}, v_{a_1}, v_{a_2}, v_{a_3}$  are real numbers denoting the magnitude, while  $tf_{\omega_1}, tf_{\omega_2}, tf_{\omega_3}, tf_{v_1}, tf_{v_2}, tf_{v_3}$  are integer numbers denoting the number of the corresponding time functions. The values of these velocities are equal to the magnitude multiplied by values evaluated by the corresponding time function. The unit of angular velocity will be rad/second. If inch is chosen as the unit of length, the unit of linear velocity will be inch/second.

If *analysis\_flag* is equal to 3, the next line will be used to provide an integer *nev* for the total number of frequencies and corresponding mode shapes to be extracted from the eigenvalue analysis.

The next line lists nine integers arranged as:

$$nkp \ nmemb \ ncond\_pt \ nmate \ nframe \ ncond\_mb \ ndistr \ ntimefun \ ncurv$$

where *nkp* is the total number of key points, *nmemb* the total number of members, *ncond\_pt* the total number of points having prescribed conditions (including boundary conditions), *nmate* the total number of cross-sections, *nframe* the total number of frames, *ncond\_mb* the total number of members having prescribed, distributed loadings, *ndistr* the total number of functions used to approximate the distributed loads, and *ntimefun* the total number of time functions, *ncurv* total number of initial curvature/twist sets including  $k_1, k_2, k_3$

The next *nkp* lines are the coordinates for each key point arranged as:

$$kp\_no \ x_1 \ x_2 \ x_3$$

where *kp\_no* is an integer representing the unique number assigned to each key point and  $x_1, x_2, x_3$  are three real numbers describing the location  $(x_1, x_2, x_3)$  of the point. **Although the arrange-**

ment of  $kp\_no$  is not necessary to be consecutive, every point starting from 1 to  $nkp$  should be present.

The next  $nmemb$  lines list eight integers for each member. They are arranged as:

$memb\_no$   $kp\_1$   $kp\_2$   $mate\_no1$   $mate\_no2$   $frame\_no$   $ndiv$   $curv\_no$

where  $memb\_no$  is the number of member and  $kp\_1$  is the starting point and  $kp\_2$  is the ending point of the member.  $mate\_no1$  is the cross-section number of the starting point of the member,  $mate\_no2$  is the cross-section number of the ending point of the member. If the member has a uniform cross-section, these two numbers will be the same. Two different cross-sections can be assigned to one member, which implicitly assumes that the sectional properties are linearly varying along the length.  $frame\_no$  is the frame number of the starting point of the member. If it is set to be 0, then the frame is the same as the global frame, *i.e.*,  $\mathbf{b}_i = \mathbf{a}_i$ .  $ndiv$  is the total number of elements you want to divide this member (the element is uniformly divided into  $ndiv$  segments),  $curv\_no$  is the initial curvature/twist set number (0 means a straight member) **Although the arrangement of  $memb\_no$  is not necessary to be consecutive, every member starting from 1 to  $nmemb$  should be present.**

The next  $ncond\_pt$  blocks define the point conditions of prescribed displacements, rotations, forces, or moments. They are arranged as:

$kp\_no$   
 $dof\_1$   $dof\_2$   $dof\_3$   $dof\_4$   $dof\_5$   $dof\_6$   
 $val\_1$   $val\_2$   $val\_3$   $val\_4$   $val\_5$   $val\_6$   
 $tf\_1$   $tf\_2$   $tf\_3$   $tf\_4$   $tf\_5$   $tf\_6$   
 $ff\_1$   $ff\_2$   $ff\_3$   $ff\_4$   $ff\_5$   $ff\_6$

where  $kp\_no$  is the key point where the prescribed condition is applied,  $dof\_i$  ( $i = 1, 2, \dots, 6$ ) are the prescribed degrees of freedom and they are six integers with values ranging from 1 to 12.  $val\_i$  ( $i = 1, 2, \dots, 6$ ) are six real numbers for the prescribed values for the corresponding degree of freedom.  $tf\_i$  ( $i = 1, 2, \dots, 6$ ) are six integer numbers for the corresponding number of the time function.  $ff\_i$  ( $i = 1, 2, \dots, 6$ ) are six integer numbers (could be either 0 or 1) indicating whether the corresponding prescribed quantity is a follower quantity or not. If it is a follower, then the flag is set to be 1. Otherwise, it is 0. We assume that only forces/moments can be follower quantities and if there is a follower component at the point, GEBT assumes that the first three prescribed degrees of freedom can only be either 7, 8, 9 or 10, 11, 12. If the assumption is too restrictive, please let the author know. The prescribed value is calculated as  $val\_i$  multiplying the corresponding time function value. For example if at point 2, we applied follower forces and dead moments with forces are constant with respect to time and moments are prescribed by the # 1 time function (defined later), then we have the following

2  
7 8 9 10 11 12

```

F1 F2 F3 M1 M2 M3
0 0 0 1 1 1
1 1 1 0 0 0

```

where  $F_1, F_2, F_3, M_1, M_2, M_3$  are corresponding prescribed values, which could be zero, implying no external forces applied at this point. In a beam assembly, the key points can be further classified as boundary points or connection points. If a point is connected to only one member, it is a boundary point and six boundary conditions must be applied to this point. If a point is connected to more than one member, it is a connection point. The displacements/rotations of this point can be prescribed. Concentrated forces/moments can also be applied to the connection point.

The next *nmate* blocks provide the structural and inertial properties for each cross-section. First for the flexibility matrix, described using 36 real numbers arranged as:

```

mate_no
S11 S12 S13 S14 S15 S16
S12 S22 S23 S24 S25 S26
S13 S23 S33 S34 S35 S36
S14 S24 S34 S44 S45 S46
S15 S25 S35 S45 S55 S56
S16 S26 S36 S46 S56 S66

```

These values are defined in Eq. (3) and can be directly copied from VABS output file. If *analysis\_flag* is not equal to 0, we also need to provide inertial properties represented by the mass matrix which is arranged as:

```

m11 m12 m13 m14 m15 m16
m12 m22 m23 m24 m25 m26
m13 m23 m33 m34 m35 m36
m14 m24 m34 m44 m45 m46
m15 m25 m35 m45 m55 m56
m16 m26 m36 m46 m56 m66

```

These values are defined in Eq. (4) and can be directly copied from VABS output file.

If *nframe* > 0, the next *nframe* blocks provide the direction cosine matrix for each member, which are described using nine real numbers arranged as:

```

frame_no
C11 C12 C13
C21 C22 C23
C31 C32 C33

```

These values are defined in Eq. (1).

If  $ncond\_mb > 0$ , the next  $ncond\_mb$  blocks define members having prescribed loads. They are arranged as:

```

memb_no
dn_1 dn_2 dn_3 dn_4 dn_5 dn_6
val_1 val_2 val_3 val_4 val_5 val_6
tf_1 tf_2 tf_3 tf_4 tf_5 tf_6
ff_1 ff_2 ff_3 ff_4 ff_5 ff_6

```

where  $memb\_no$  is the member where the prescribed condition is applied,  $dn\_i$  ( $i = 1, 2, \dots, 6$ ) are the corresponding number of distributed functions.  $val\_i$  ( $i = 1, 2, \dots, 6$ ) are six real numbers for the prescribed values arranged corresponding to  $f_1, f_2, f_3, m_1, m_2, m_3$  with  $f_1, f_2, f_3$  as three components of the distribution force and  $m_1, m_2, m_3$  as three moments of the distribution moment.  $tf\_i$  ( $i = 1, 2, \dots, 6$ ) are six integer numbers for the corresponding number of the time function.  $ff\_i$  ( $i = 1, 2, \dots, 6$ ) are six integer numbers (could be either 0 or 1) indicating whether the corresponding prescribed quantity is a follower quantity or not. If it is a follower, then the flag is set to be 1. Otherwise, it is 0. The prescribed value is calculated as  $val\_i$  multiplying the corresponding time function value and multiplying the corresponding distribution function value.

If  $ndistr > 0$ , the next  $ndistr$  blocks provide the distributed load functions. They are arranged as:

```

fun_no
c0 c1 c2 c3 c4 c5

```

where  $c_i$  ( $i = 0, \dots, 5$ ) corresponds to the first five coefficients of the Chebychev polynomials used to approximate the given load function. The function is calculated as  $f(s) = \sum_{i=0}^5 c_i T_i(s)$ , where  $s$  is the length along the member and  $T_0 = 1, T_1 = s, T_2 = 2s^2 - 1, T_3 = 4s^3 - 3s, T_4 = 8s^4 - 8s^2 + 1, T_5 = 16s^5 - 20s^3 + 5s$ . For example, for a linearly distributed load  $f(s) = 1000 + 500s$ , we need to provide the input as 1000 500 0 0 0 0. Here GEBT assumes all given functions can be approximated by the Chebychev polynomials up to the fifth order.

If  $ncurv > 0$ , the next  $ncurv$  blocks provide the initial curvatures and twist. They are arranged as:

```

curv_no
k1 k2 k3

```

where  $k_1$  is the initial twist,  $k_2$  is the initial curvature around  $x_2$  direction, and  $k_3$  is the initial curvature around  $x_3$  direction.

If  $ntimefun > 0$  or  $analysis\_flag=2$ , the next line lists two numbers for the simulation range arranged as:

```

starting_time ending_time

```

and the next *ntimefun* blocks define the time functions which are arranged as follows:

```

fun_no
fun_type
t_s t_e
fun_block

```

where *fun\_no* is the time function number, *fun\_type* is the time function type, and *t\_s, t\_e* are the starting and ending time of function definition specifically. Currently, user can define two types of time functions. If *fun\_type* is 0, it is a user defined as piecewise linear function based on functional values provided for different time instances. The *fun\_block* is arranged as follows:

```

n
t_1 f_1
t_2 f_2
...
t_n f_n

```

where *n* is the number of entries needed for this time function. The time entries *t<sub>i</sub>* is arranged in an increasing fashion. That is, we have  $t_2 > t_1$ ,  $t_3 > t_2$ , and etc. The time function is assumed to be piecewise linear defined by *t<sub>i</sub>* and *f<sub>i</sub>*. If  $t < t_1$ , the function will remain the same as *f<sub>1</sub>*. If  $t > t_n$ , the function will remain the same as *f<sub>n</sub>*. Note this type of time functions, simulation time, and load steps can be used to control the load increment in static analysis.

If *fun\_type* is 1, the time function is defined as a summation of a series of harmonics as follows

$$f(t) = \sum_{i=1}^N h_i(t) = \sum_{i=1}^N a_i \sin 2\pi(t/T_i + \phi_i) \quad (5)$$

The *fun\_block* is arranged as follows:

```

n
a_1 T_1 φ_1
a_2 T_2 φ_2
...
a_n T_n φ_n

```

where *a<sub>i</sub>* is the magnitude, *T<sub>i</sub>* is the period, and *φ<sub>i</sub>* is the phase.

The input file should be ended with a blank line to avoid any possible incompatibility of different computer systems. The input file can be given any name as long as the total number of the characters of the name including extension is not more than 60. You are suggested to use a unique extension say *get* for you to identify such files with GEBT. For the convenience of the user to identify mistakes in the input file, all the inputs are echoed in a file named *input\_file\_name.ech*.

Error messages are also written at the end of *input\_file\_name.ech*.

If *analysis\_flag=2*, a file named *input\_file\_name.ini* should also exist in the same directory. If there are a total of *nelem* elements in the structure, this file contains  $2 \times nelem$  lines with the first *nelem* lines providing the initial positions and rotations ( $u_1 \ u_2 \ u_3 \ \theta_1 \ \theta_2 \ \theta_3$ ) for each element and the next *nelem* lines providing the corresponding derivatives ( $\dot{u}_1 \ \dot{u}_2 \ \dot{u}_3 \ \dot{\theta}_1 \ \dot{\theta}_2 \ \dot{\theta}_3$ ) for each element.

## 7 GEBT Outputs

GEBT outputs 1D beam displacements, rotations, sectional forces and sectional moments for each point defined in the input file and each element in a file named *input\_file\_name.out*. The element is identified using the coordinates of its mid-point.

First, we output the coordinates in the global coordinate system arranged as  $x_1, x_2, x_3$ . Immediately following, we output the results for displacement/rotation variables. For each line, there are 6 values arranged as  $u_1 \ u_2 \ u_3 \ \theta_1 \ \theta_2 \ \theta_3$ .

Then we output the results for force/moment variables. For each line, there are 6 values arranged as  $F_1 \ F_2 \ F_3 \ M_1 \ M_2 \ M_3$ .

For dynamic analysis, we also output the linear and angular momenta for each element and there are 6 values values arranged as  $P_1 \ P_2 \ P_3 \ H_1 \ H_2 \ H_3$ .

For eigenvalue analysis, GEBT outputs the steady state solution first. Then it outputs the eigenvalues (in Hz) and eigenvectors corresponding to this state. The eigenvalues are listed from smallest to the largest in magnitude.

It is noted that the displacements and rotations are expressed in the global coordinate system ( $\mathbf{a}_i$ ) while the forces/moments and momenta are expressed in the deformed beam coordinate system ( $\mathbf{B}_i$ ) for each element. The forces/moments are expressed in the global coordinate system ( $\mathbf{a}_i$ ) for each boundary point. As the forces/moments of each connection point could be different for different elements it is associated with, which can be obtained from the elements connected by this point, such results are not reported but the places are replaced with zeroes. Note the calculated forces and moments at a boundary point are the internal forces and moments evaluated at this point. If the point is the ending point of the member, the internal forces/moments will be equal to applied forces/moments. If the point is the starting point of the member, the magnitude of internal forces/moments will be equal to applied forces/moments but with a different sign. For boundary points with applied forces/moments, the prescribed values are directly copied in the output file. The output file is in pure text format and can be opened by any text editor.

## 8 GEBT Installation

GEBT is distributed in the form of GEBT#.#ReleasePCMM-DD-YEAR.zip for Windows operating systems. Sometimes to circumvent some email systems, the extension “zip” is changed to be “pass”. You just need to change it back to be a zip file once you received it. #.# is the version number. If “PC” is replaced with “LIN”, it will be a release for Linux system. If you also want to run GEBT in other platforms, please let the author know. Extracting the file to a folder you choose is all you need to do for installation. There are at least three ways to run GEBT in a Windows operating systems:

- The safest way to execute GEBT is to run *GEBT inputfile* as a DOS command in Windows. This can be achieved by Click Start, choose Run, and type in “cmd” and click OK. Then use “cd” to enter the right folder where both the GEBT executable and the input file is in.
- Drag your input file into the GEBT executable. A window appears to run GEBT along with the input file. After it is done, the window will disappear and you have new files generated in the folder of the input file. You can check your results in the output files as described previously.
- Assign a unique extension, say *get*, to your input file, and configure files with this extension to be opened by GEBT and to be edited by a text editor. From a folder’s window, click Tools->Folder Options. Once the Folder Options dialog is open, click tab File Types. Click the New icon to Create New Extension dialog, type *get* and click OK. With the new file type *get* highlighted in the Registered file types list, click Advanced to open the Edit File Type dialog. Click Change icon to choose the icon you like. Next, click on New, type *open* under Action, then click Browse and point to the GEBT executable and click OK. Click on New, type edit under Action, then click Browse to find the executable for text editor you like to use for editing input files and click OK. Click OK and Close to exit the Folder Options dialog. Then you only need to double click the input file, it will launch GEBT for you. You can also rightclick an input file and open it with the text editor.

Also the linear system is solved using HSL Mathematical Software Library MA28 (<http://www.hsl.rl.ac.uk/>). It belongs to HSL Archive and free for anybody who requests. However, I cannot freely include it in the distribution of GEBT. You need to download a double precision of MA28 (save it as ma28.f) and HSL dependencies (save as ddep.f). You also need to download a double precision of MC19 (save it as mc19.f). You also need to change all the single precision real number statement in mc19.f into double precision. Make sure in your computer you have gfortran compiler available. MakeMA28 is the makefile for you to compile a dynamic link library named as MA28.dll. The eigenvalue problem is solved using ARPACK along with some subroutines/functions from BLAS and LAPACK, which are free and already included in the distribution. The source codes are also included in the release and you are free to modify the code to make it more suitable for your own problem. Make files for compiling the code using gfortran are also included in the release.

## 9 GEBT Maintenance and Tech Support

Prof. Yu is committed to maintain and provide tech support for GEBT. A Google group is specifically set up for information exchange related with GEBT. *Users are highly encouraged to sign up through <http://groups.google.com/group/hifi-comp> to receive most recent news of GEBT, ask questions, and share with others. **A technical question must be posted in the Google group before it will be answered.** A page of GEBT FAQ will be constantly updated in the group. Before you ask questions, please do the following:*

1. Read the GEBT manual carefully, if you have not done so;
2. Check the error message at the end of *input\_file\_name.ech*;
3. Make sure that you have provided the right input data through *input\_file\_name.ech*, which is GEBT's understanding of your input file;
4. Check the GEBT FAQ page on the Google group;
5. Post your question in the discussion section of the group.

A designated web site ([hifi-comp.com](http://hifi-comp.com)) has also been created to provide more exposure of the codes Prof. Yu has developed including GEBT, VABS/PreVABS, VAPAS, VAMUCH, and DNAD through Internet. Please note the code is copyrighted by Utah State University and you are not allowed to redistribute it. Please direct those who are also interested in GEBT to Prof. Yu for permission.

## 10 Epilogue

Although GEBT is still in its infancy, it will emerge to be a useful, general-purpose analysis code for composite beams along with VABS.

## Acknowledgements

The development of GEBT was initially supported by the Chief Scientist Innovative Research Fund at AFRL/RB WPAFB and later supported by Advanced Dynamics Inc. through an Army SBIR project. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsement, either expressed or implied, of the funding agencies.

## References

- [1] D. H. Hodges. A mixed variational formulation based on exact intrinsic equations for dynamics of moving beams. *International Journal of Solids and Structures*, 26(11):1253 – 1273, 1990.
- [2] M. V. Fulton and D. H. Hodges. Aeroelastic stability of composite hingeless rotor blades in hover – part I: Theory. *Mathematical and Computer Modelling*, 18(3/4):1 – 18, 1993.

- [3] M. V. Fulton and D. H. Hodges. Aeroelastic stability of composite hingeless rotor blades in hover – part II: Results. *Mathematical and Computer Modelling*, 18(3/4):19 – 36, Aug. 1993.
- [4] D. H. Hodges, X. Shang, and C. E. S. Cesnik. Finite element solution of nonlinear intrinsic equations for curved composite beams. *Journal of the American Helicopter Society*, 41(4):313 – 321, Oct. 1996.
- [5] D. H. Hodges. *Nonlinear Composite Beam Theory for Engineers*. AIAA, Washington DC, 2006.
- [6] V. L. Berdichevsky. Variational-asymptotic method of constructing a theory of shells. *PMM*, 43(4):664 – 687, 1979.