

ThinSecBeam – User Manual

“Thin-walled Sections of Beams”
Notes to version 1.0 (October 2019)

© 2021 Juan Carlos del Caño
Ph.D. in Continuum Mechanics – University Teacher

This is a free translation of the original manual (in Spanish) released in 2019.

The author is grateful for all the support from his colleagues at the Area of Continuum Mechanics and Structures at the Escuela de Ingenierías Industriales (University of Valladolid), teachers Antonio Foces, Jesús Magdaleno, Antolín Lorenzana, Mariano Cacho, Alvaro Magdaleno, Estrella Requejo & José Pereda. Their suggestions have been inspiring and useful in the development of both versions 0.8 (previous stable) and 1.0 (current).

Foreword and License

The program is mostly self-documented. The short tips in the interface and its intuitive design should suffice. Some background in “thin-walled beams” is assumed on the user side though. This manual offers some more elaborated explanations along with guidance on additional aspects (remarkably how to install the program), but you can find your way around without reading it.

The program ThinSecBeam (“Thin-walled Sections of Beams”) analyzes a beam’s section under “thin-wall” and elastic hypothesis. The thin-walled hypothesis is broadly considered as appropriate in the study of metallic beams, as they are usually profiled with a small wall thickness compared to the outer dimensions of the section. This hypothesis has implications both in geometry and stresses and is consistently assumed in the program. It is up to the user to assess whether his problem reasonably fits the thin-walled model or not in the first place.

The program aims at being both a reliable numerical tool in general and a useful resource for a student. You certainly have not much opportunity of looking at a warped section after all !

The program calculates almost everything at hand based only on the (thin-wall idealized) section’s geometry: it calculates barycentre, principal inertias and axis, section moduli, section core, torsion stiffness, warping modulus, shear centre and shear areas. If values are provided for one or more stress resultants (shear force, bending moment, axial force, torque), some plots of stress fluxes and warping displacements will be drawn. Among them, 3D practicable plots of warping displacements are of high value in understanding the deformation of the section. The text output includes additional info (significant points of the core, shear swirl tensor etc).

An original approach to shear stresses and warping displacements calculation has been implemented. It applies equally to torsion or bending and makes it irrelevant whether closed paths in the section are present or not.

There are several aspects in the analysis of a thin-walled beam that are dependent on the third dimension of the beam (its length). Non-uniform torsion effects and beam’s instability come to mind. ThinSecBeam does not take any third-dimension-related data (such as the length and support of the beam or the evolution of the loads over the length), thus it must be clear that these effects fall outside the scope of the program. Nevertheless, the text report, if requested, offers some rough info about instability.

The present version 1.0 of the program is considered “finished” in that it implements all the features initially planned. Compared to the previous stable version 0.8, this 1.0 version implements the circular tract which was considered a critical improvement. Furthermore, its implementation is “exact” in the scope of the thin-walled model (all calculations use analytical formulae obtained by using the symbolic manipulator SymPy when required). This allows for the representation of the section using only a handful of tracts, which in turn greatly simplifies the data input. The user interface has been completely revamped being now possible to modify the problem data and to analyze as many load cases as desired.

Due to these improvements, data files of ThinSecBeam version 0.8 are no longer compatible with version 1.0. This shouldn’t be a real problem as providing the data to the user interface will be in general a short and comfortable process.

ThinSecBeam is Free Software offered under the GPL v3 license. It is in turn based on free tools having similar licenses making its redistribution possible and legal. The tools and resources used include Python3 (the language ThinSecBeam is coded in), Tk, NumPy, SciPy, SymPy & Matplotlib. The Geany editor used for coding deserves also an acknowledgement along with all software integrated into the Gnu-Linux system, under which the program has been developed.

The use of Free Software gives some rights to the user, but also comes with some obligations. Please see the option “License” in the user interface for a brief on this subject.

As for this manual, it is offered under the following license:

“Attribution-NonCommercial-ShareAlike 3.0 Unported” from Creative Commons. It is a license conceived for sharing, not that much for restricting the usage conditions. You can read the license at <http://creativecommons.org/licenses/by-nc-sa/3.0/> or you can write a letter to Creative Commons, 444 Castro Street, Suite 900, Mountain View, California, 94041, USA.



To sum up, this license gives you the right to freely:

- *Share – make copies of this work and distribute them.*
- *Reuse – adapt and make derivatives of this work.*

Under the following conditions:

- *Attribution (“BY”)- You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use. In this case you must acknowledge the author as “Juan Carlos del Caño, assistant professor at Escuela de Ing. Industriales - Universidad de Valladolid”.*
- *Non commercial (“NC”)- You may not use the material for commercial purposes.*
- *Share Alike (“SA”)- If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original. .*

Installation

First step: get the program.

The current version of ThinSecBeam is 1.0. It has been tested quite thoroughly and is considered stable. The whole program is in the file ThinSecBeam_10_en.py (English version), valid for any operating system (Linux, Windows, Mac, BSD, etc.). The author is fine with you getting the program “out there”, but he can’t be aware of possible, maybe malicious, modifications. Just to be sure, the recommended way of getting the program is from the author’s GitHub page:

<https://github.com/JC-Cheloven/ThinSecBeam>

You can also ask directly to the author at jcarlosc@eii.uva.es.

Second step: install Python3 and required dependencies.

To date, there is no such thing as “an installer” for ThinSecBeam. Perhaps there will be one in the future, even if the author is not especially happy with the idea. If that eventually happens will be because of environmental pressure and common uses. In the author’s opinion, the main disadvantage of “an installer” is that it creates a sort of distance between the user and the code. As a teacher, the author very much prefers to ease access to the code for those who want to learn from it or are just curious about the implementation.

So, to run the program you need to install Python 3 and the dependencies the program uses. This also gives you the appropriate coding environment in case you want to study or modify the program. As a design criterion, ThinSecBeam uses as few dependencies as possible. Most likely you’ll be ready to go by installing the library Matplotlib under your Python 3. Matplotlib in turn depends on NumPy, SciPy, Tk, and a handful of other libraries which (hopefully) will be automatically installed for you. If a dependency is missing, a message in the terminal will tell you what is it so that you can install it using your preferred method.

The recommended way to make the dependencies available depend on the operating system as described below. The recommended procedures aim to be light and take some 120Mb of your disk space.

GNU-LINUX ® systems

A Linux user will probably be familiar with the dependencies thing, as it is the usual way of keeping the system up and running, installing updates etc. The directions are simple here: use your package manager to install the missing libraries. Again if some further dependency is not satisfied you’ll be informed in the terminal. As an example, for Debian systems and derivatives (Ubuntu etc) the following command was enough:

```
sudo apt install python3, python3-tk, python3-matplotlib
```

Finally, run the program ThinSecBeam_10_en.py in the way you like better (from terminal with **python3 ThinSecBeam_10_en.py** or from your file manager, or make a desktop icon etc).

WINDOWS ® systems

To make Python 3 available (if it isn’t yet), you can visit <https://www.python.org/downloads> and click the download you need (will be Python 3.6 or later). You’ll obtain the right download for your operating system version. Please follow the install process, selecting the “Add Python 3 to system’s path” box when it appears. You can check also “Make Python 3 available to all users”, depending on your preference and needs.

Note: the installer deploys python in a subfolder many levels deep. As for this writing, Windows still comes with a limitation of 260 characters for the PATH variable. Thus, the Python path may or may not fit depending on how populated your PATH variable already is. If the Python installer shows an option to unlock this limitation, please check it. If all goes well the limitation will grow to 23000 characters. You can also perform this action from Windows itself.

Follow up: since the times of Python 3.6.3 this issue is no longer observed and the PATH variable is correctly assigned in all cases.

The program “pip3” will be available once Python 3 is installed. It is a sort of package manager for Python-related libraries and extensions. Please open a system symbol window and type

```
pip3 install matplotlib
```

Which will hopefully install as secondary dependencies all needed stuff for ThinSecBeam to work. After that, it should be possible to run the file ThinSecBeam_10_en.py as a python3 program, from the file manager (right button of the mouse, “open with...” and choosing python), or from the system symbol (travel first to the folder where the program is) by typing:

```
python ThinSecBeam_10_en.py
```

If something goes wrong it’s probably because of some missing dependency. Please read the output in the system symbol to find out. It may be numpy or scipy, or tk... You can install using pip3 that dependency (for example **pip3 install tk** or similar).

There are other ways to make Python 3 available in your system, typically by installing a Python distribution as “Anaconda” (see: <https://docs.continuum.io/anaconda>). There is a reduced version of Anaconda called Miniconda (<https://conda.io/miniconda.html>) which is lighter and equally competent regarding the execution of ThinSecBeam. You can install any of both as administrator and then open the particular terminal of Anaconda/Miniconda and type:

```
conda install matplotlib
```

Which again should install all needed dependencies. You are ready to run ThinSecBeam by using one of the methods mentioned before. On the first run, your operating system may ask you which application should be used to open this file. You must specify the path to the executable called “python.exe”, which is in the Anaconda / Miniconda folder in your user directory.

In the future, your operating system will remember (or you’ll make it to remember) that ThinSecBeam is to be opened with that particular python executable. Double click or any other usual procedure will be available to run ThinSecBeam. You may also create a desktop shortcut etc.

You should not care about Anaconda / Miniconda anymore, except if you want to run some other Python program that uses different dependencies. Again, if the name of that dependency is “xyzt” you will type **conda install xyzt** as you did with matplotlib.

MAC[®] systems

If you own a Mac and have ever needed something not packed from factory, you probably installed the popular program HomeBrew. It offers similar functionality to a typical package manager found in Gnu-Linux distributions. If so, you can install Python3 from terminal by typing:

```
brew install python3
```

If you don’t have HomeBrew installed but you wish to, it is available at <https://brew.sh>

The process of installing Python 3 without HomeBrew is similar to Windows': download Python 3 from <https://www.python.org/downloads> and follow the guided installation (don't forget to check the "add python3 to system path" box).

Furthermore, Anaconda / Miniconda are also available for Mac systems. You may choose that if the environment and functionality offered by Anaconda are useful for you. No further reference will be made here for the sake of conciseness (you'll be using **conda install xyz** to install libraries etc). Using HomeBrew and Anaconda/Miniconda in the same system (or any two package managers for that matter) is not recommended as they could conflict.

Whatever the process, you have now Python 3 available in your system. So you have the utility program pip3 as well. You can use that to install the dependencies ThinSecBeam needs. From terminal:

```
pip3 install matplotlib
```

After that you should be able to run ThinSecBeam from terminal (adjust the path as needed):

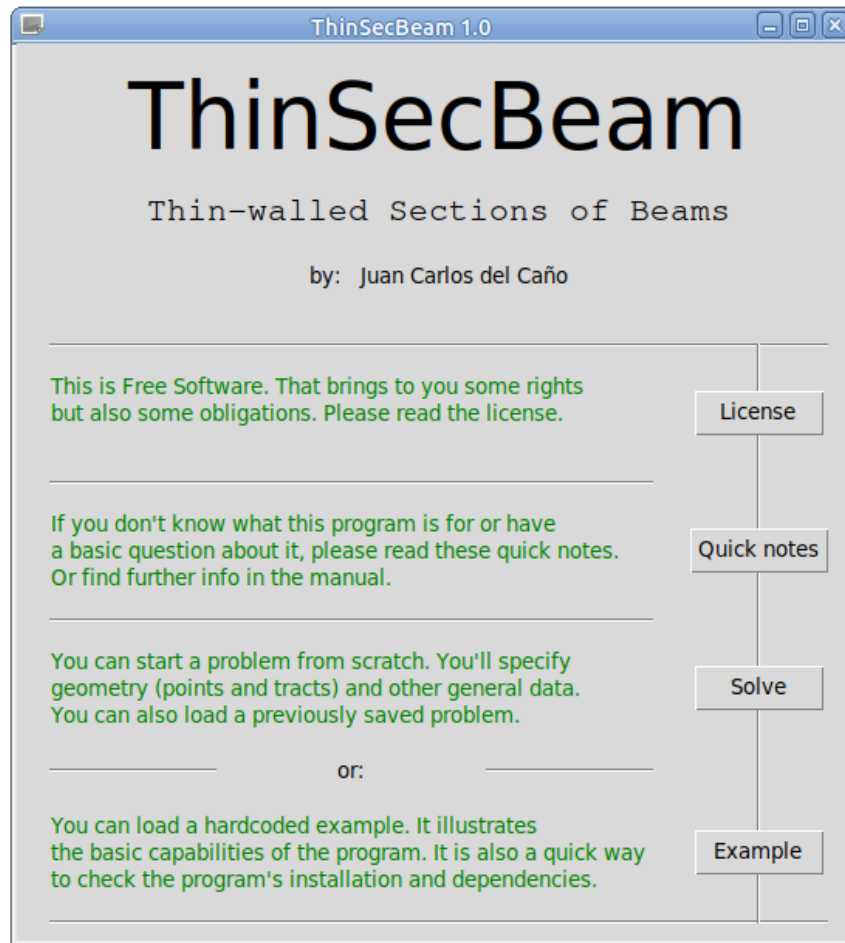
```
python3 ~/Downloads/secciones/ThinSecBeam_10.py
```

Or you can invoke ThinSecBeam by using some other desktop procedure. Again, a missing dependency will be likely the cause of any arising error. Look in the terminal output what is it and install it (for example by **pip3 install xyz**).

The above procedure has been successfully tested with Python 3.7.4. Installation of numpy & matplotlib libraries was required.

How to use ThinSecBeam

At start, the program shows a window like this:



- Figure 1 -

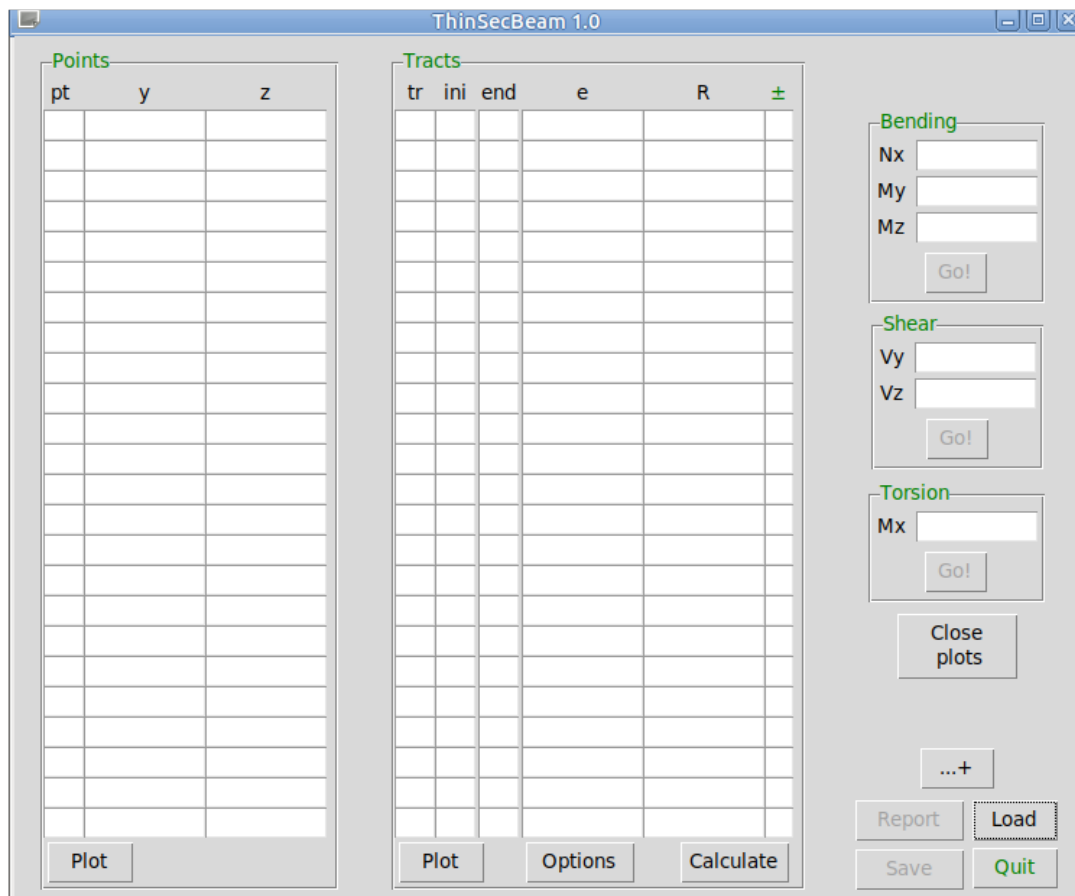
A small disclaimer: the author isn't an interface design expert. He's not particularly interested in becoming one either. My hope is that you'll find it an effective interface that doesn't distract you from your work. No more, no less.

You can interact with the previous window using the mouse (of course), but also `<Tab>` or `<Shift><Tab>` to move among buttons, and `<Space>` to push the current button.

In general: the key shortcuts: `<Tab>`, `<Shift><Tab>`, `<Space>` will work similarly on any window in the program.

Buttons "License" & "Quick notes" show respectively the license terms of the program and a short user guide. The "Example" button launches the analysis of a hardcoded example with no further preamble. It demonstrates some of the capabilities of the program. Afterwards, you can make changes to the problem, or you can load an existing problem from file, or perform any other action.

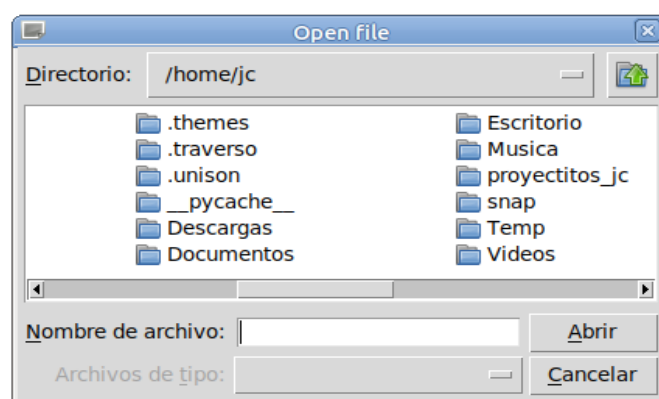
The "Solve" button takes you to an empty main interface. From here you can load from file a problem previously saved or you can start typing the data of a new problem. The empty main interface looks as in the next figure:



- Figure 2 -

LOAD an existing problem

The “Load” button is preselected on opening the main interface (so the load action will be directly launched by pressing <space>). This button will take you to a window as in figure 3, where you select the file to be loaded. The initial folder is the user’s root directory.

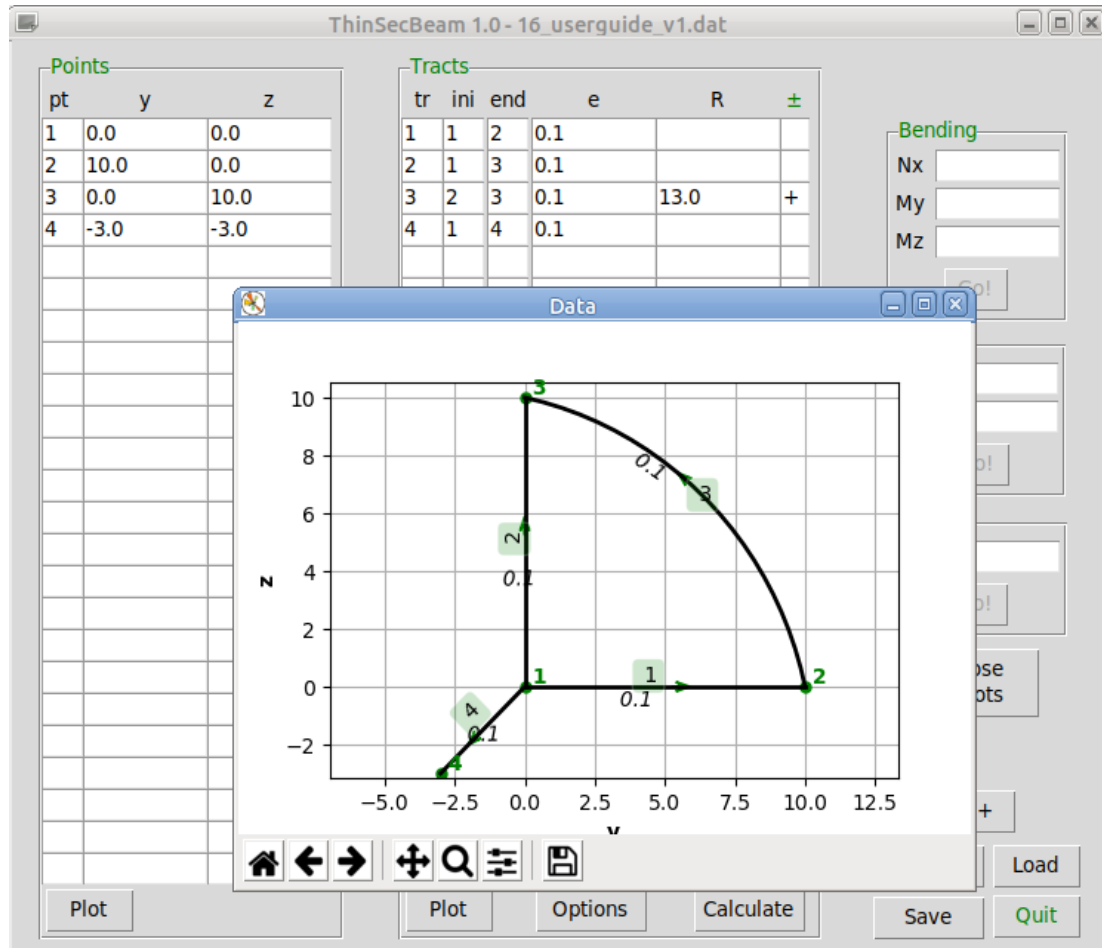


- Figure 3 -

After loading the data from a file, the main window will get the same exact state as if you have typed your problem’s data by hand. This last situation (when you type the required data by hand) is illustrated just in the next section.

Providing data for a NEW PROBLEM

You must provide the “points” and “tracts” that made up for the geometry of the section. “Points” are understood as extreme points of “tracts”, and should be introduced in the first place. Each “point” is identified by a code number (any positive integer not being used for some other point) and defined by two real numbers describing the y-z coordinates in the plane of the section. The “Plot” button in the area of “Points” will draw the just defined points in a separate window for you to ckeck for errors etc.



- Figure 4-

Tracts are also identified by a natural number (not being used for some other tract), and are defined by the code of its initial and ending points (“ini” & “end”), and the wall thickness (“e”). Nothing else is needed for a straight tract.

When the tract is circular, you must specify also the radius (“R”), and the sign of the angle from the initial point to the ending point, this one being positive counter-clockwise. You must write “+” or “-” for this last parameter. The program will assume that the tract follows the shortest path from the initial point to the ending point. This makes the data input simpler and avoids ambiguities, but implicitly forbids tracts covering angles of 180° or greater. If your section has an arc covering an angle equal or greater than 180°, simply use two tracts to describe it. As said, if the radius and sign fields are left blank, the program will generate a straight tract.

The “Plot” button in the area of “Tracts” will plot points and tracts alike in a separate window, again for you to check. This window will be labeled “Data”. Figure 4 shows such window in the context of an example for which data have just been provided. The main window interface is shown in the background.

In general it makes no sense “to discretize” a constant thickness tract in sub-tracts: the program uses exact analytical expressions to calculate displacements, stresses, Warping Modulus and Shear Centre, among other parameters. The symbolic library SymPy has been used to derive the more complicated analytical expressions.

Once data is provided and checked, we’re ready to press the “Calculate” button. This launches the calculations needed for the analysis of the section. The “Data” window is refreshed, as you may have introduced some changes without plotting afterwards. A summary of the most relevant attributes of the section will be shown in a window labeled “Datasheet”.

Anytime you press the “Plot” button in the tracts area, it is assumed that you made some changes in the data (if you didn’t there isn’t a reason to re-plot). The practical consequence of this is: you have to press again the “Calculate” button before requesting further results.

The program is “unit agnostic”. Any provided data (for example point coordinates, tract thickness, stress resultants) are assumed to be given in a coherent unit system chosen by the user. The program will not question that unit system so be careful at this point.

Once the “Calculate” button is pressed, options appear to analyze the section against loads. The data you must provide as loads are the values of each stress resultant in the section. They are often obtained from an analysis of the beam, or you may want to simply use 1-valued loads. Again, the program does not question where these data come from.

Analyzing the section against LOADS.

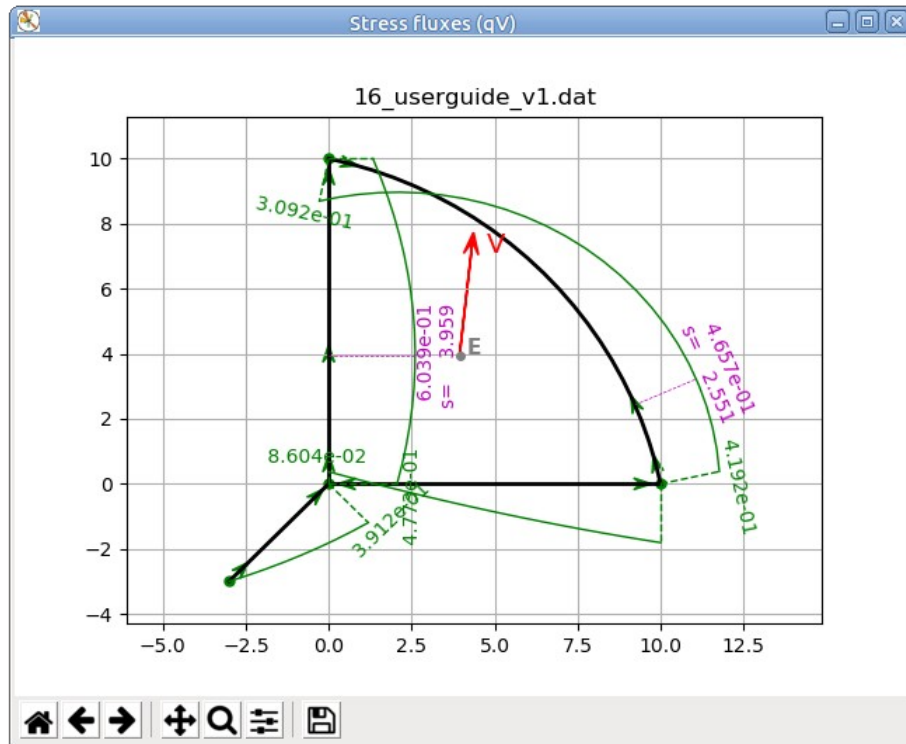
The area labelled “Bending” is where you can provide values for bending moment components M_y , M_z , and axial force N_x . Button “Go!” will plot 2D and 3D drawings of axial stresses associated with these loads. As it is well known, axial normal stresses simply follow the equation of a plane, which certainly is not very interesting. These plots are provided for completeness and for illustrative and teaching purposes anyway.

Components of stress resultants are considered positive when they point in the direction of the axis, except for M_z which is considered positive when contrary to the z axis.

The area labeled “Shear” is where you can provide values for shear force components V_y , V_z . The button “Go!” in this area will create a 2D drawing of stress fluxes and a 3D plot of warping displacements.

In the 2D representation of shear fluxes, measures are always parallel to the flux. In points connecting only two tracts, the flux measure is written only once for the sake of clarity. Extreme values (maximum or minimum) are shown in magenta along with its “s” coordinate. This coordinate is local to the tract, measured from its initial point. In the Data window, each tract has an arrow indicating the growing direction of “s” for your convenience.

Figure 5 shows this representation of stress fluxes associated with a particular shear force. One or more arrows in the tract indicate the flux direction in each portion of the tract. The shear force V is represented by a red vector in the plane of the section. When reasonable regarding aesthetics, V will be drawn from the shear centre E . In some cases (for example when E is far away outside the section) the program will decide to plot V somewhere aesthetically acceptable. In any case, a valid reduction of the shear stresses is of course the resultant V passing by E .



- Figure 5 -

Buttons below the plot in Figure 5 allow you to interact with the plot. They may appear somewhat different depending on your desktop environment and platform. Their functionality is (left to right):

- return to the initial view,
- return to the previous view,
- return to the next view,
- move (left mouse button drag) or zooming (right mouse button drag)
- zoom to a rectangle
- adjust margins and similar stuff
- save the figure to file

These buttons also appear in any other 2D results window in the program. Use them when you want to change the view or if, for example, two measures overlap (they will change their relative position when zooming).

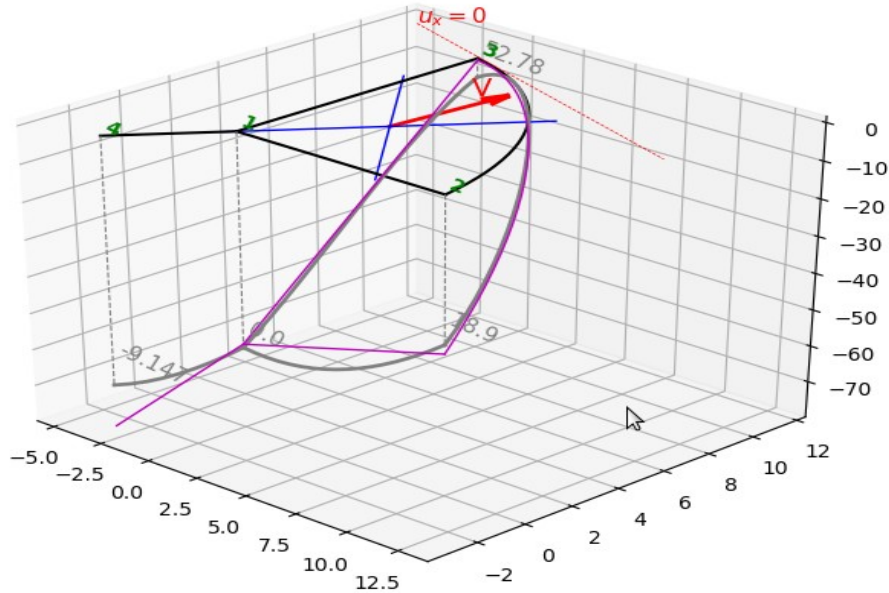
Figure 6 shows a 3D plot of warping displacements associated with shear. The warped shape of the section is plotted in grey. An interpolation plane on the previous warping displacements is shown in magenta. This plane is calculated using a geometric criterion (there are plans for implementing also an energy criterion in the future). Principal inertia lines (blue), shear force (red), and the intersection line of the interpolated plane with the plane of the section itself (dotted red) are also represented for your reference.

The angle between the interpolated plane (magenta) and the plane in which the section is drawn (black), would be the additional turning angle of the section over the angle from Navier-Bernoulli's model, according to Timoshenko's model.

All displacements calculated by the program are multiplied by the Shear Modulus G , whatever value it has (the program does not need to explicitly account for this value and in fact, it doesn't).

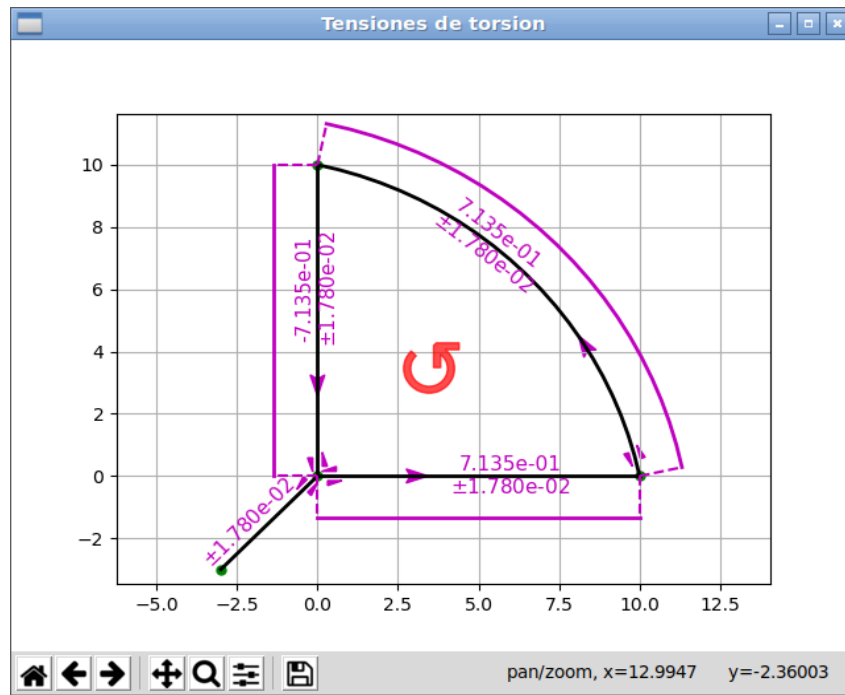
There can be a constant shift between the shown measures of warping displacement u_x and what is appreciated in the plot. "This is not an error but a feature": there is an option to displace the plot for better visualization (see the "Options" button later). The default option for this plot is to shift the

deformed shape outside the section. You can select any other available option and re-draw the figures if you want so. In any case, the numeric measures in the plot are the calculated ones. Calculated displacements are relative to a particular “base point” which is arbitrarily given a value $u_x=0$. The program chooses a base point with no particular criterion, but if you’re interested you can pick your choice in the “Options” window. You can identify this point in the plot as the one labeled with zero displacement. In this case, the base point is point 1. This figure, like any other 3D plot in the program, can be rotated using the mouse to achieve a better view etc.



- Figure 6 -

The area labeled “Torsion” is where you can provide a value for the torque moment M_x . Button “Go” in this area will generate a 2D figure with the torsion stresses (in units of stress, not flux in this case), and a 3D figure with warping displacements associated to torsion. Uniform torsion is assumed. Figure 7 shows the 2D stress plot for a particular value of M_x .



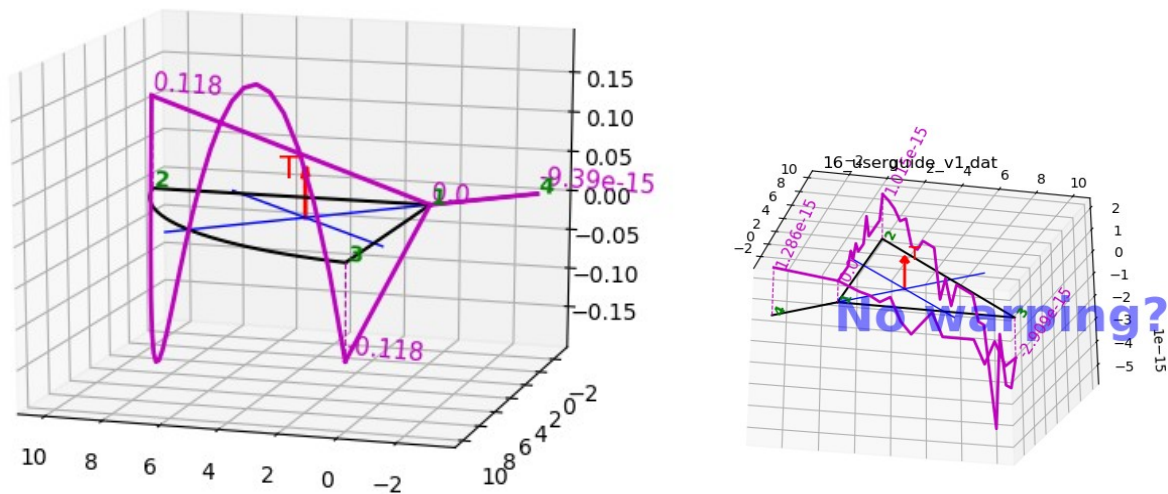
- Figure 7 -

According to the common model adopted in the program, torsion will produce flux-like stresses and also bidirectional stresses across the thickness of the wall. Both stress types should be superimposed to get the actual stress in a material point.

“Flux-like stresses” will only exist in closed paths of the section, and will be constant both in the length of the tract and through its thickness. They are denoted by an arrow on the tract. The function (constant) drawn over the tract shows the magnitude of this stress.

“Bidirectional stresses” have a linear evolution over the thickness of the wall, being zero at its central line, and remain constant in the length of the tract. These stresses do exist both in open branches and in closed paths of the section. The element of moment dT associated with an element ds of the tract’s length has always the direction of the torque T . These stresses are indicated by two semi-arrows at both sides of the thickness. Its maximum value occurs at the surfaces of the wall and is indicated prepended by a “ \pm ” sign as a reminder of its particular evolution. If there are closed paths, these “bidirectional stresses” are in general much lesser than the “flux-like stresses”, and you may want to disregard them as it is common practice in manual calculations.

Figure 8 illustrates the 3D plotting of warping displacements associated with torsion for the same value of T . Remarks made for the 3D plot of shear-related displacements apply also here.



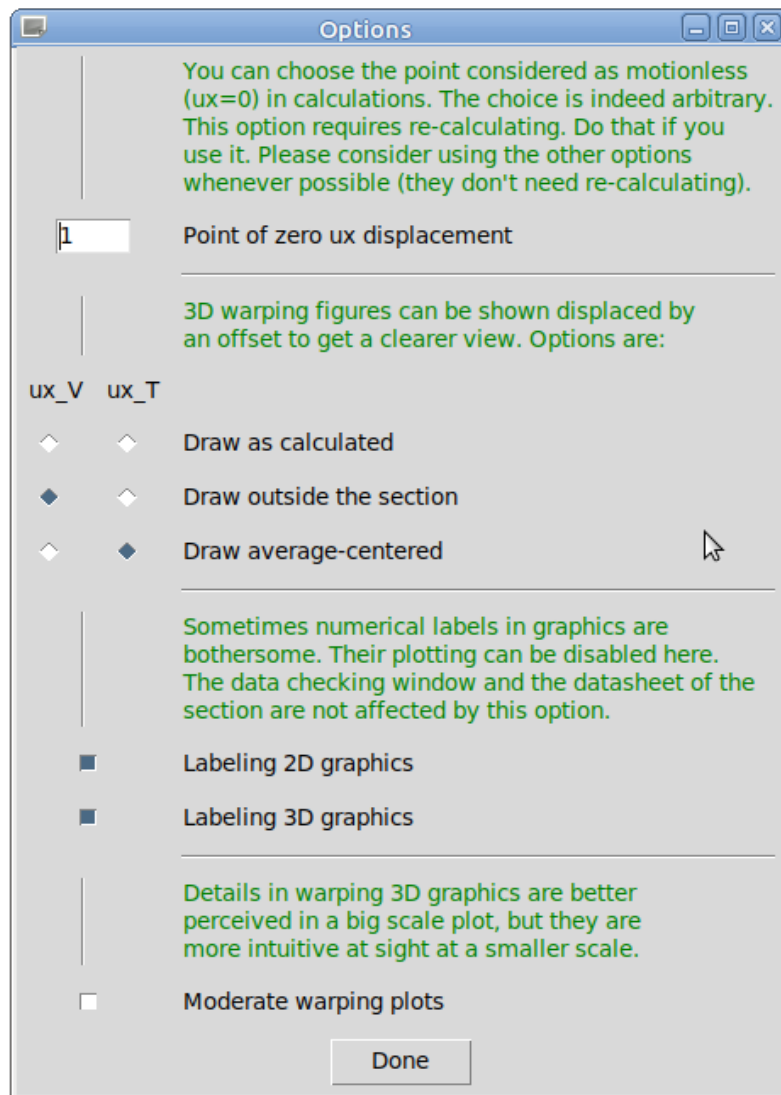
- Figure 8-

Please note that stresses being constant in the length of the tract, displacements will have a linear evolution in the tract’s length, producing a straight deformed line for straight tracts. This is a consequence of the underlying mathematical model, not any undesired plot effect or anything.

Some sections, due to their topology, does not warp at all under torsion. The program tries to detect such situations and prints a warning message (“No warping?”) over the figure if the case seems suspicious. Please see the right of figure 8 for an example. In such cases, the calculated displacements are merely numerical noise of course.

Among the rest of the buttons in the interface, the “Options” button deserves a particular overview. It opens a window as in figure 9 where you can tweak some aspects of the program’s behaviour.

In the first field, with label “Point of zero u_x displacement”, you can enter a point of your choice to be taken as the zero reference for the rest of u_x displacements. The choice is indeed arbitrary. The one appearing by default will be used unless you specify a different one. Modifying this option requires to re-calculate the section, so you may prefer using some other option if possible (for example you have other options to shift a 3D warping plot or to make more noticeable some symmetry or anti-symmetry in the solution).



- Figure 9-

The next field gives you control on the shift to be applied in the visualization of warping 3D plots, both due to shear or torsion.

There is some common use (at least in the author's perception) consisting in plotting the torsion warping draw centred where the undeformed section has been drawn. This option, labelled as "Draw average-centred", comes preselected from factory but you may want to change it.

Please note: while the overall length increment of a rod under torsion is in fact zero, that's not a valid argument to dogmatize about how u_x displacements should be plotted (there could be rigid-solid motions or net tractions superimposed on the rod etc). But the aforementioned common use probably has its roots in that fact.

On the other hand, the generated figure for shear-warping has a lot of stuff to show. In order to avoid overlapping of data, plotting the warping displacements outside the plane of the undeformed section is preferable. That's why the default option is "Draw outside the section" in this case. You can choose any option for each figure anyway, including the third one "Draw as calculated" which will draw the base point on the plane of the undeformed section.

Next checkboxes allow disabling the labelling (measures etc) of 2D and 3D graphics, for a cleaner plot if so desired. Finally, the “Moderate warping plots” checkbox will produce 3D warping plots closer to the undeformed section. As indicated in the interface, a closer plot may be found more intuitive but a larger-scale one allows for a better view of details.

Back to the main window (see figure 2), the button labelled as “...+” will load a problem from file and launch the analysis right away, as if the “Calculate” button would have been pressed. The “Data” and “Datasheet” windows will be generated. In the main interface, the areas for points and tracts data are disabled and it will only be possible to analyze the section against loads.

The foreseen use of this functionality is to analyze problems with more points and/or tracts than the user interface can allocate. To do this, the user has to generate the data file “by hand” using a plain text editor of his choice, with no limitations in the size of the data. Specifications for this file can be found in the next section “The data file”.

The author actually expects this feature to be never needed. Remember that you have at hand straight and circular tracts of exact treatment. You will not achieve better precision by “discretizing” in more tracts than necessary to define the geometry.

The “Load” button in the main interface (see figure 2) will load from file a regular problem (namely one fitting in the interface). The file was typically generated by the program itself at the touch of the “Save” button, although it can be an edited file as well. The data will simply be put in the interface. From this point on, you can proceed as if you’d just finished typing the data by hand.

Lastly, the “Report” button generates a text report, which is wider and more detailed than the “Datasheet”. The contents of this report are commented on in the next section.

The text report

A text report can be requested by pressing the “Report” button in the interface. It provides more detailed info than the “Datasheet”. Its contents are commented below for the same example previously appearing in this manual. Some familiarity and a general understanding of the mathematical model behind thin-walled beams is assumed on the user side. Please consult some book about beams and structures if you have to (teaching that falls out of scope in this user guide).

The identification and geometric data of the problem are printed first:

```
##### ThinSecBeam 1.0 generated report #####
-----
Identification:  /home/jc/l6_userguide_v1.report
-----

Points:
Base point= 1
_____given axes_____      _____axes at G_____
ip      y      z      yG      zG      inciP
1      0      0      -3.3948      -3.3948 [1, 2, 4]
2      10     0      6.6052      -3.3948 [1, 3]
3      0      10     -3.3948      6.6052 [2, 3]
4      -3     -3     -6.3948     -6.3948 [4]
-----

Tracts - basic parameters:
it  tipoT  pto_i  pto_j      e      L
1   0      1      2      0.1     10
2   0      1      3      0.1     10
3   1      2      3      0.1    14.953
4   0      1      4      0.1     4.2426

Tracts - geometric parameters:
it      alfa      alfai      R      yC      zC
1      0
2      1.5708
3      1.1502      0.21029      13      -2.7136      -2.7136
4      3.927
```

Point coordinates are given both in the initial axes and in axes by barycenter G. The array inciP contains the tracts sharing a given point. For a straight tract, alfa is the angle from the y-axis to the tract, and it is the angle covered by the tract for a circular tract. The angle alfai (circular tracts only) comes from the y-axis to the initial radius of the tract. All angles are in radians.

Some parameters internally used by the program are informed next:

```
Tracts - operation parameters:
it      Qy      Qz      iQy      iQz      Asect
1      -3.3948      1.6052      -16.974      -0.30714      39.424
2      1.6052      -3.3948      -0.30714      -16.974      -39.424
3      3.8662      3.8662      9.8487      47.962      61.268
4      -2.0767      -2.0767      -3.9553      -3.9553      5.2755e-14
-----
```

- Qy & Qz are the tract's static moments (area-wise moments).
- iQy & iQz are the integrals of static moments Qy(s) & Qz(s) over the length of the tract.
- Asect is the sectoral area of the tract measured from the shear centre E.

Other parameters calculated by the program are listed next. They are commented in the same order of appearance in the report:

Section - global parameters:

yG	zG	A	ACy	ACz	
3.3948	3.3948	3.9196	1.6276	1.6276	
Warning: shear areas seem to NOT make sense, at least in yz axis.					
Iy	Iz	Iyz	Ipsi	Ieta	thetaPR
56.244	56.244	-3.1639	53.08	59.408	-0.7854
Wpsi	Weta	spherical I	principal yz		
5.8694	8.4016	no	no		
(W_psi is conditioned by yG,zG=			-6.3948	-6.3948)	
(W_eta is conditioned by yG,zG=			6.6052	-3.3948)	
yE	zE	yE(G)	zE(G)		
3.9424	3.9424	0.54762	0.54762		
J	I_a				
56.181	1.1511				

- Baricentre coordinates (yG, zG)
- Section area (A)
- Shear areas (ACy, ACz)
- An indication about the possible usefulness of the previous shear areas in yz axes. It is based on whether the Shear Twirl Tensor is substantially diagonal or not.
- Inertia moments relative to the given axes (yz), and principal axes (psi,eta). The principal axis with greater inertia (strong axis) is named "eta" in the program.
- Angle from z-axis to the strong axis, thetaPR. Positive if counter-clockwise. This angle is also shown in the Datasheet of the section.
- Four lines follow with the bending modulus of the section (Wpsi, Weta) along with some indication on which point conditions each modulus.
- Coordinates (yE, zE) of Shear Centre (E) respect yz axes and respect parallel axes by G come next.
- Finally, values of the torsion stiffness (J) and Warping Modulus (I_a) are given in the last line. These parameters are calculated by "exact" analytical expressions (they're not obtained using numerical integration).

The next section shows miscellaneous information. Probably of not much interest for a typical normal use of the program:

```

For a torque Mx=1, and for the current base point,
exact average of torsion displacement ux is:
      G*ux_mean = 1.1068834e-15

Shear Twirl Tensor C_ij, with i,j in y,z:
[[0.61438543 0.10436452]
 [0.10436452 0.61438543]]

In inertia principal axis, C_psi,eta is:
[[ 5.10020907e-01 -6.10622664e-16]
 [-2.22044605e-16  7.18749947e-01]]

In self principal axis, C_diag is:
C_I= 0.51002 ; C_II= 0.71875
with eigenvectors:
n_I = -0.707, 0.707
n_II= 0.707, 0.707
(angle from yz= -45.000 degrees)

```

- The average displacement ux for a unit torque T (named Mx here) is presented in the first place. Again, this is an "exact" value, not based on sampling or interpolation but on analytical expressions.

- Detailed information on the components of the “Shear Twirl Tensor” is given afterwards, both in y-z and in psi-eta inertia principal axes. Components of this tensor are the rotated angles towards y&z of the interpolated planes (obtained from ux displacements) when each component of shear force is in action. The author defines this tensor according to the following relation:

$$G u_x = \begin{pmatrix} V_y & V_z \end{pmatrix} \begin{pmatrix} C_{yy} & C_{yz} \\ C_{zy} & C_{zz} \end{pmatrix} \begin{pmatrix} y \\ z \end{pmatrix}$$

where G is the Shear Modulus of the material. A constant term, which would represent an arbitrary shift of the section in the x-direction, is omitted. This tensor has some properties and uses which exceed the scope of this user guide.

We would have a hard time trying to define what “exact” means when calculating the Shear Twirl Tensor, as it refers to interpolated planes and the interpolation can be based on various criteria. The program performs a least-squares interpolation to a sampling of the warping displacement ux to define interpolated planes. The sampling is equally spaced in the tracts of the section, and the number of points is decided by using certain internal logic in the program. As a reference, the program takes 67 sample points in the example used in this user guide. It is a geometry interpolating criterion anyway. Other criteria could be used, and in fact, there are plans to implement also an energy criterion in the future.

Some information about lateral buckling is presented next. Please be aware of this phenomenon being at all times dependent on conditions in the third dimension (the length) of the beam. Therefore it falls out of the scope of this program. On the other hand, as lateral buckling is a potentially important phenomenon in thin-walled beams, some indicative words about it could be appreciated by the user:

Guidance info on lateral buckling						
--- (not rigorous, please read the manual) ---						
Bending on the strong axis (eta), and a typical steel for beams are assumed.						
As a rough guidance:						
For a beam of length = 2.2931 or higher, neglecting the effect of the warping modulus is reasonable.						
Guidance values of M _{cr} /E for some beam lengths, being E~ 2.1e5 MPa (please translate to the problem units):						
L	65	130	195	260	325	390
M _{cr} /E	1.732	0.8659	0.5772	0.4329	0.3463	0.2886

The above information is based on the recommendations of Eurocode 3, Annex F “Lateral buckling”, where the provided formulae are for a symmetric section over the weak inertia axis, the bending occurring around the strong axis (named z in the Eurocode).

The simplified expression used in this report has additional requirements, namely:

- section with equal wings
- bending moment constant in the length
- loads being applied at the shear centre
- the extreme sections are embedded (roughly, please check the Eurocode for precise info)

The expression in this situation is:

$$M_{cr} = \frac{\pi^2 E I_z}{L^2} \left[\frac{I_a}{I_z} + \frac{L^2}{\pi^2} \cdot \frac{G I_T}{E I_z} \right]^{1/2}$$

The report gives values of the bending critical moment M_{cr} under the conditions above, for some values of the length of the beam.

Next in the report comes the section core. All of the points used in its drawing are specified.

Circular tracts, if any, have been sampled at 5° intervals or less. Exact tangent lines at these sampled points are taken as neutral lines for the plotting. There is a “difficult” case: the one in which a neutral line has to pass by some point of the section outside the circular tract (and of course be tangent to the tract). For these special cases, the tangency point in the tract is approximated by the nearest sampling point. Thus, some particular points of the core may not be analytically exact (those whose neutral lines involve that particular case of tangency, if any). Again, if all the above sounds obscure to you, please consult some appropriate book. In particular, a chapter devoted to columns.

Core of the section: (y,z coordinates in G-centered axis; points are ordered anti-clockwise):

y	z
-0.83729	2.955
-2.4109	-0.38357
-2.4628	-0.49377
-2.5006	-0.60738
-2.5641	-0.84979
-2.5945	-1.1058
-2.5853	-1.3683
-2.5317	-1.6281
-2.4313	-1.8745
-2.2849	-2.0967
-2.0967	-2.2849
-1.8745	-2.4313
-1.6281	-2.5317
-1.3683	-2.5853
-1.1058	-2.5945
-0.84979	-2.5641
-0.60738	-2.5006
-0.49377	-2.4628
-0.38357	-2.4109
2.955	-0.83729
-0.83729	2.955

The report continues with the calculated value of ux displacements and fluxes for unitary torque (Mx) and shear (Vy, Vz) components. All subsequent calculations inside a tract are performed in the program using this information.

Values at the extreme points of the tracts (per unit loading):

i	pto	ux(Vy=1)	ux(Vz=1)	ux(Mx=1)
1		0	0	0
2		5.7019	1.4661	0.011802
3		1.4661	5.7019	-0.011802
4		-0.9147	-0.9147	-9.3902e-16

q(Vy=1)		q(Vz=1)		q(Mx=1)
i_tr	i	j	i	j
1	-0.054768	0.029543	0.015645	0.043294
2	0.015645	0.043294	-0.054768	0.029543
3	-0.029543	-0.043294	-0.043294	-0.029543
4	0.039123	0	0.039123	0

The above displacements are positive as the x-axis indicates: coming from the plane of the section to you. Fluxes are positive if they come towards the point, be it the initial one in the tract (i) or the ending one (j). In the case of torsion, being the flux constant in the tract, it will have the same value and opposite signs in the extreme points of the tract. That's why only the value of the flux at the ending point (j) is specified.

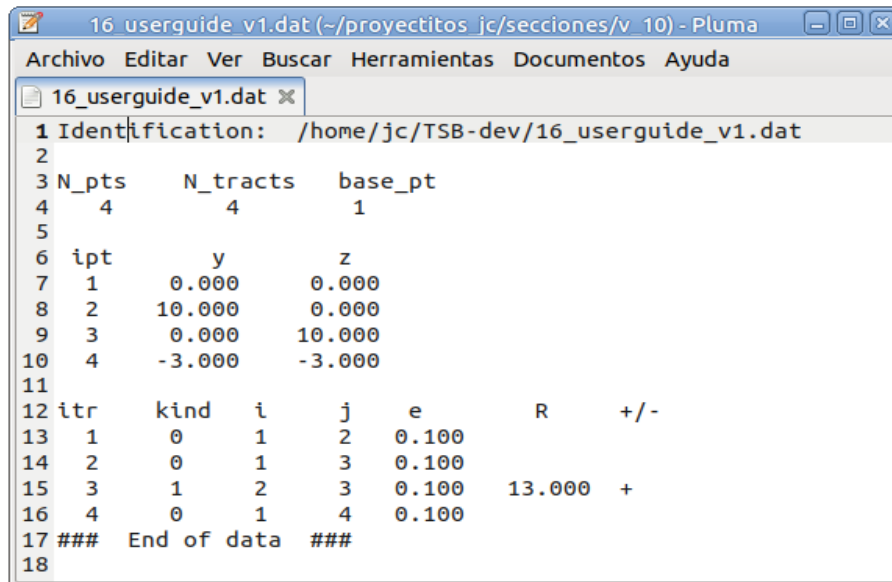
The report ends with a big list containing the sampling values used by the program to generate the plots requested by the user with “Go!” buttons. All values have been obtained by using analytical expressions, so they are “exact” as for the thin-walled mathematical model, except for the rounding error (16-digit float is used thoroughly). The listing is trimmed here for the sake of conciseness. Also, the inclusion of bending data has been deemed unnecessary as it is all about the equation of a plane.

Detailed info for unit loads:								
i_tr	y	z	s	ux(Vy)	ux(Vz)	q(Vy)	q(Vz)	ux(Mx)
1	0	0	0	0	0	0.054768	-0.015645	0
1	0.58824	0	0.58824	0.33262	-0.080997	0.058221	-0.0119	0.00069421
1	1.1765	0	1.1765	0.68375	-0.14007	0.061058	-0.0081901	0.0013884
1	1.7647	0	1.7647	1.0497	-0.17742	0.063277	-0.0045148	0.0020826
1	2.3529	0	2.3529	1.427	-0.19325	0.064879	-0.00087426	0.0027768
1	2.9412	0	2.9412	1.8118	-0.18777	0.065864	0.0027316	0.003471
1	3.5294	0	3.5294	2.2006	-0.16118	0.066231	0.0063027	0.0041652
1	4.1176	0	4.1176	2.5898	-0.11369	0.065982	0.0098391	0.0048595
1	4.7059	0	4.7059	2.9757	-0.045499	0.065115	0.013341	0.0055537
1	5.2941	0	5.2941	3.3546	0.043191	0.063631	0.016808	0.0062479
1	5.8824	0	5.8824	3.7231	0.15217	0.06153	0.02024	0.0069421
1	6.4706	0	6.4706	4.0773	0.28124	0.058812	0.023638	0.0076363
1	7.0588	0	7.0588	4.4138	0.43019	0.055477	0.027	0.0083305
1	7.6471	0	7.6471	4.7288	0.59882	0.051524	0.030328	0.0090247
1	8.2353	0	8.2353	5.0187	0.78693	0.046955	0.033622	0.0097189
1	8.8235	0	8.8235	5.28	0.99431	0.041768	0.03688	0.010413
1	9.4118	0	9.4118	5.5089	1.2207	0.035964	0.040104	0.011107
1	10	0	10	5.7019	1.4661	0.029543	0.043294	0.011802
2	0	0	0	0	0	-0.015645	0.054768	0
2	3.6019e-17	0.58824	0.58824	-0.080997	0.33262	-0.0119	0.058221	-0.00069421
2	7.2038e-17	1.1765	1.1765	-0.14007	0.68375	-0.0081901	0.061058	-0.0013884
2	1.0806e-16	1.7647	1.7647	-0.17742	1.0497	-0.0045148	0.063277	-0.0020826
. (etc)								
3	0.58186	9.8617	14.355	1.734	5.8582	-0.046212	-0.022754	-0.001394
3	0	10	14.953	1.4661	5.7019	-0.043294	-0.029543	-0.011802
4	0	0	0	0	0	-0.039123	-0.039123	0
4	-0.375	-0.375	0.53033	-0.19816	-0.19816	-0.035544	-0.035544	-1.1738e-16
4	-0.75	-0.75	1.0607	-0.37634	-0.37634	-0.03159	-0.03159	-2.3476e-16
4	-1.125	-1.125	1.591	-0.53256	-0.53256	-0.027262	-0.027262	-3.5213e-16
4	-1.5	-1.5	2.1213	-0.66483	-0.66483	-0.022559	-0.022559	-4.6951e-16
4	-1.875	-1.875	2.6517	-0.77117	-0.77117	-0.017481	-0.017481	-5.8689e-16
4	-2.25	-2.25	3.182	-0.84959	-0.84959	-0.012029	-0.012029	-7.0427e-16
4	-2.625	-2.625	3.7123	-0.89809	-0.89809	-0.0062017	-0.0062017	-8.2165e-16
4	-3	-3	4.2426	-0.9147	-0.9147	0	0	-9.3902e-16
----- end of the report -----								

The data file

The data file for a problem is a plain text file that you can read and edit. In a typical use case, the program will handle this file transparently and the user will not see or edit the contents of this file. The user could need to edit this file only in the (unlikely) situation of a problem with more tracts and/or points than the user interface can allocate. This section purposely covers that situation.

Figure 10 shows the data file for the example used before in this user guide. The author has the habit of appending “.dat” to data files but this is entirely up to you. Any name is admitted.



```
1 Identification: /home/jc/TSB-dev/16_userguide_v1.dat
2
3 N_pts    N_tracts    base_pt
4 4        4          1
5
6 ipt      y          z
7 1        0.000      0.000
8 2        10.000     0.000
9 3        0.000      10.000
10 4       -3.000     -3.000
11
12 itr     kind    i      j      e      R      +/-
13 1       0       1      2      0.100
14 2       0       1      3      0.100
15 3       1       2      3      0.100    13.000    +
16 4       0       1      4      0.100
17 ### End of data ###
18
```

- Figure 10 -

The first line in the file contains a short identification of the problem. It will be the complete path to the file if it has been generated by the program using the save button. When the file is being edited by hand, any human-readable explanation is accepted. The program does not use it in any way. The second line is blank by default, but it could be used also for human-readable identification if edited by hand. The third line serves as a readable heading to the fourth line. All these lines are included for convenience and legibility but are ignored by the program. They must exist though, even if left blank.

Line 4 is where data starts. Its content is, left to right: number of points, number of tracts, and the point to use as “base point” (point of zero displacement).

Data in a line are separated by one or more spaces. A decimal point (not a comma) is to be used when appropriate. Powers of 10 are indicated by the “e” letter. For example, a valid decimal number is 5.477e-4. Please note the absence of spaces.

Line 5 is left blank for legibility. Line 6 is a human-readable heading for line 7 onwards. These lines must exist although its content is ignored by the program.

The coordinates of each “Point” are specified from line 7 in as many lines as points were specified in the first field of line 4. Four lines in this case, each containing the code for the Point, the y coordinate and the z coordinate.

The next line (11 in this case) is left blank for legibility. The next line (12) is a human-readable header for lines 13 onwards. Again, the lines must exist although their content is irrelevant to the program.

The Tracts data begins at line 13. There will be as many lines as the number of tracts, as specified in

the second field of line 4. Each of these lines contains: tract code, initial point code, ending point code, tract thickness. No more data will be present for a straight tract. For a circular tract there will also be the radius and the sign of the arc from initial node to ending node (“+” for counterclockwise, “-” for clockwise).

The program realizes whether the tract is circular or straight by detecting (or not) the presence of these two fields. These data (radius and sign of the angle) along with the convention of shortest angular path from initial to ending points, unambiguously determine the geometry of the tract.

The codes for the extreme points of a tract don’t need to fit any particular condition (initial and ending points don’t need to be in ascendant order, for example)

The codes for Points or Tracts don’t need either to be in a particular order or follow a consecutive or compact numbering. If they often do in the examples is for user convenience, not a program requirement.

Next line, 17 in this case, is included for legibility but it is ignored by the program. It can be omitted if desired. Ditto for any following line, be it blank or not. It is possible to use such lines, in the desired quantity, for any use. For example, you can write here additional remarks, or paste the report, or the terminal output. You can also leave here alternate chunks of a data file for later use, etc.

The user interface is the recommended way to make changes to an existing problem, be it small details (as modifying some point coordinates or some thickness) or substantial changes (as adding tracts and points). The manual editing of a text file is perhaps more prone to errors although you may prefer it. As said, the only situation when manual editing is mandatory is when the user interface cannot accommodate all points and/or tracts of your problem.

*I hope you’ll find ThinSecBeam useful.
The author can be contacted at the email shown in the “Installation” section.
Suggestions and comments are welcome.*

*Juan Carlos del Caño
Ph. D. in Continuum Mechanics – University Teacher
University of Valladolid
Spain*