The Journal of Open Source Software

sectionproperties: A Python package for the analysis of arbitrary cross-sections using the finite element method

Robbie van Leeuwen[1] **and Connor Ferster**[2]

1 Independent Researcher, Australia **2** Independent Researcher, Canada

DOI: 10.21105/joss.06105

Software

- Review
- Repository
- Archive

Editor: Prashant K Jha

Reviewers:

- @dbrizard
- @pprachas

Submitted: 06 November 2023
Published: 19 April 2024

Summary

Properties of plane cross-sections are often required in engineering research, analysis, and design. For example, cross-sectional properties are used to determine the displacements, natural frequencies, and stresses within beams under complex loading. `sectionproperties` is a Python package for the analysis of *arbitrary* cross-sections using the finite element method. `sectionproperties` can be used to determine geometric and warping properties, as well as visualising cross-sectional stresses resulting from combinations of applied loads. `sectionproperties` aims to provide a pre-processor, analysis engine, and post-processor, in a single open source and accessible package, that can be used by researchers, practising engineers, and students.

Statement of Need

Obtaining the geometric properties of simple shapes is a classical engineering problem with well-defined analytical solutions. However, obtaining warping properties, e.g. for torsion and shear analyses, involves solving partial differential equations (Pilkey, 2002). While some analytical solutions exist for a small subset of geometries, the method for obtaining these results is not able to be generalised to shapes commonly used in engineering pratice. Further, the analysis of arbitrary composite geometries, in which a cross-section could consist of any shape with any number of internal holes, and be made from any number of materials, complicates both geometric and warping computations.

To the best of our knowledge, there is no open source software available for the computation of both geometric and warping propreties for composite, arbitary cross-sections. While there are several commercial solutions available, e.g. `RSECTION 1`, `ShapeDesigner SaaS`, or `CADRE Profiler`, none of these are open source or provide an application programming interface (API) that would enable these programs to be used for research. As a result, `sectionproperties` supports both engineering practice and research, by implementing an open source solution to the complex modelling problem that is arbitrary composite geometric and warping analyses.

Implementation

`sectionproperties` harnesses the power of Shapely (Gillies et al., 2023) to streamline geometry generation, and triangle (Rufat, 2023) (a python port of Triangle (Shewchuk, 2002)) to produce a triangular mesh of six-noded quadratic elements. The finite element method is used to solve for the geometric and warping properties, the latter involving the solution of partial differential equations and boundary value problems (Pilkey, 2002). For example, the Saint-Venant torsion constant (J) is obtained by solving for the warping function, ω (Pilkey, 2002):

$$\nabla^2 \omega = 0$$

subject to the boundary condition:

$$\frac{\partial \omega}{\partial x} n_x + \frac{\partial \omega}{\partial y} n_y = y n_x - x n_y$$

Using the finite element method, this problem is reduced to a set of linear equations of the form:

$$\mathbf{K} \omega = \mathbf{F}$$

where the stiffness matrix and load vector at the element level are defined as:

$$\mathbf{k}^e = \sum_{i=1}^{6} w_i \mathbf{B}_i^{\mathrm{T}} \mathbf{B}_i J_i$$

$$\mathbf{f}^e = \sum_{i=1}^{6} w_i \mathbf{B}_i^{\mathrm{T}} \begin{bmatrix} \mathbf{N}_i \mathbf{y}_e \\ -\mathbf{N}_i \mathbf{x}_e \end{bmatrix} J_i$$

In the above, $\mathbf{N}$ and $\mathbf{B}$ are the shape functions and their derivatives, and w_i and J_i are the weights and Jacobians of the current integration point. The boundary conditions neccesitate the inversion of a nearly singular global stiffness matrix. As such, the Lagrangian multiplier method is used to solve the set of linear equations of the form $\mathbf{K}\mathbf{u} = \mathbf{F}$ by introducing an extra constraint on the solution vector, whereby the mean value is equal to zero (Larson & Bengzon, 2013).

$$\begin{bmatrix} \mathbf{K} & \mathbf{C}^{\mathrm{T}} \\ \mathbf{C} & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \lambda \end{bmatrix} = \begin{bmatrix} \mathbf{F} \\ 0 \end{bmatrix}$$

where $\mathbf{C}$ is the assembly of $\sum_i w_i \mathbf{N}_i J_i$, and λ may be though of as a relatively small force acting to enforce the constraints. Once the warping function has been evaluated, the Saint-Venant torsion constant can be calculated as follows:

$$J = I_{xx} + I_{yy} - \omega^{\mathrm{T}} \mathbf{K} \omega$$

The calculation of plastic properties is meshless, and is conducted using an iterative method to enforce plastic equilibrium, yielding the plastic centroids. A full description of the theoretical background underpinning `sectionproperties` can be found in the documentation.

An example of some of the visualisation generated by `sectionproperties` can be seen in Figure 1 below.

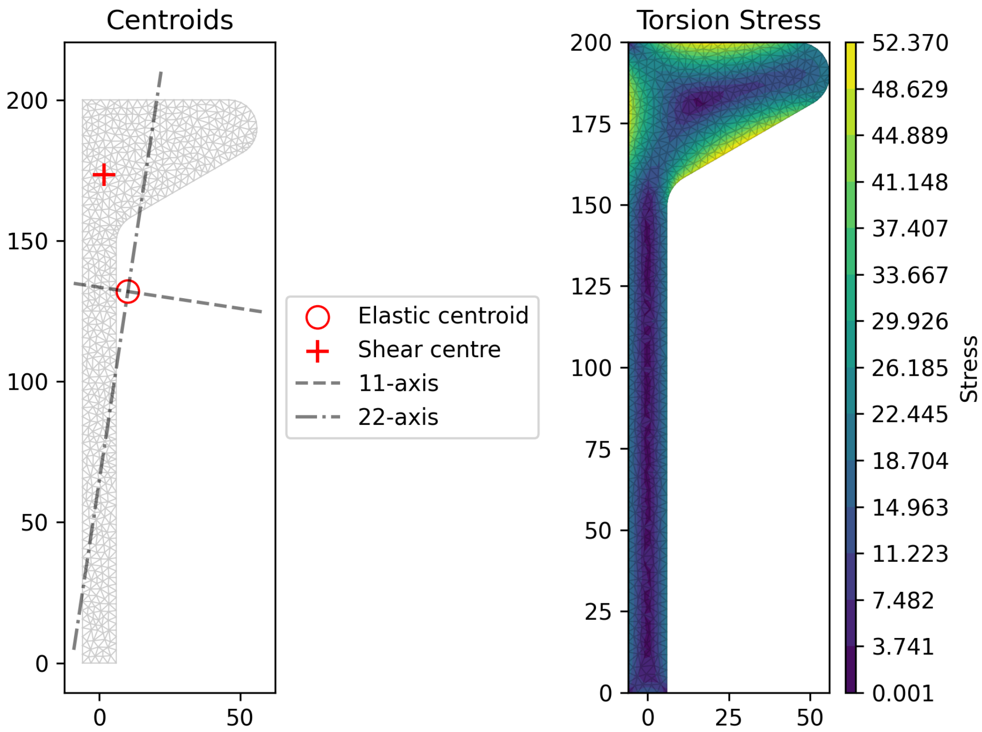

Figure 1: Plot of the centroids and torsion stress distribution for a bulb-section modelled in `sectionproperties`.

Software Development

The `sectionproperties` package is available on GitHub, where the source code, issue tracker, CI workflow, and discussion board can be found. Pre-commit hooks are used to ensure code quality and style is consistent across all contributions. There is an extensive testing and validation suite used to ensure that the output produced by `sectionproperties` is verified and repeatable, including a set of benchmarking tests. `sectionproperties` has an actively maintained and complete documentation, including installation instructions, a detailed user guide, a list of examples, and an API reference.

Conclusion

In this paper we have described `sectionproperties`, a Python package that calculates the section properties of arbitrary sections. It is our hope that this project is used by researchers and practising engineers to improve their experimental and analysis workflows.

Acknowledgements

We acknowledge the contributions from all the contributors to `sectionproperties`.

References

Gillies, S., Wel, C. van der, Van den Bossche, J., Taves, M. W., Arnott, J., Ward, B. C., & others. (2023). *Shapely* (Version 2.0.2). https://doi.org/10.5281/zenodo.5597138

Larson, M. G., & Bengzon, F. (2013). *The finite element method: Theory, implementation, and applications* (1st ed., Vol. 10). Springer Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-33287-6

Pilkey, W. D. (2002). *Analysis and design of elastic beams: Computational methods*. Wiley. https://doi.org/10.1002/9780470172667

Rufat, D. (2023). Triangle. In *GitHub repository*. GitHub. https://github.com/drufat/triangle

Shewchuk, J. R. (2002). Delaunay refinement algorithms for triangular mesh generation. *Computational Geometry*, *22*(1), 21–74. https://doi.org/10.1016/S0925-7721(01)00047-5