

BeamBending: a teaching aid for 1-D shear force and bending moment diagrams

Alfredo R. Carella¹

¹ OsloMet - Oslo Metropolitan University

DOI: [10.21105/jose.00065](https://doi.org/10.21105/jose.00065)

Software

- [Review](#) ↗
- [Repository](#) ↗
- [Archive](#) ↗

Submitted: 10 July 2019

Published: 02 September 2019

License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC-BY](#)).

Summary

BeamBending is *both* an educational module *and* a Python package, based mainly on Hunter (2007), Oliphant (2006) and Meurer et al. (2017). It is intended to serve as a teaching aid during a first course in Statics. This module aims to enhance clarity and provide visual hands-on examples while introducing the concepts of:

- stresses on slender *one-dimensional* solids (i.e. beams)
- normal force, shear force, and bending moment diagrams

The [package documentation](#) includes a simple (but still rigorous enough) explanation of the background theory, inspired in Beer, Russell Johnston, Dewolf, & Mazurek (2017) and Bell (2015). It is assumed that the students understand static equilibrium of flat rigid bodies, but a short recap is provided. Code snippets that reproduce the theory examples are presented next to each result.

The package can be used by

- teachers who want to automatically create problem sets with their solutions (easily scriptable, *random-problem-generator friendly*);
- students who want to verify their solutions to introductory problem sets;
- students who like to play with example problems and receive immediate visual feedback (i.e. about how simple modifications to imposed loads affect the resulting reaction forces and internal stresses).

The `beambending` package is ready for installation using `pip` or can be tested online using the provided [Jupyter notebook](#).

Statement of Need

Statics courses in undergraduate engineering programs are sometimes taught before the knowledge of the relevant mathematical tools (i.e. simple calculus and linear vector algebra) is fully mature. Introducing a topic that resembles the mindset of calculus and employs an unintuitive standard sign convention, on top of a wobbly mathematical foundation, makes it fairly common for students to get lost in the calculations.

This package/module aims to bridge this gap and simplify students' first contact with this challenging new topic by working on two fronts simultaneously:

- Explain the [background theory](#) from a simple example with focus on connecting the mathematical description with the physical beam model (beambending code snippets are interleaved to illustrate how the package works).
- Provide a temporary scaffolding that helps to establish an immediate visual association between beam load states and internal stresses.

Several online tools with similar functionality are currently available, such as Beam Guru (2019), Structural Beam Deflection and Stress Calculators (2019), SkyCiv Beam (2017), MechaniCalc (2019), Engineers Edge (2019), WebStructural (2019), Beam Calculator Online (2019), and Steel Beam Calculator (2019). Most of them expose only a graphical user interface to the user, eliminating the need to write any code; roughly half of the surveyed tools are free while others charge monthly subscriptions, and only a few include a theoretical module or present to the user a detailed solution procedure for the problems. This feature comparison is presented in Table 1.

	Arbitrary distributed load functions	Full GUI (no code required)	Arbitrary number of loads	Free (as in free beer)	Open Source	Featured theory module	Detailed solution procedure	Programmable interface
BeamBending	✓	✗	✓	✓	✓	✓	✗	✓
Beam Calculator Online	✗	✓	✓	✓	✗	✗	✗	✗
Beam Guru	✗	✓	✓	✓	✗	✗	✓	✗
Engineers Edge	✗	✓	✗	✗	✗	✗	✗	✗
MechaniCalc	✗	✓	✓	✗	✗	✓	✗	✗
SkyCiv Beam	✗	✓	✓	✗	✗	✗	✓	✗
Steel Beam Calculator	✗	✓	✓	✗	✗	✗	✗	✗
Structural Beam Calculator	✗	✓	✗	✓	✗	✗	✗	✗
WebStructural	✗	✓	✓	✗	✗	✗	✗	✗

Table 1: Summary of feature comparison with existing packages

There are three main differences between the `beambending` package and the rest of the reviewed tools:

- Arbitrary distributed load functions are accepted (as long as sympy can parse them), i.e. they are not restricted to constants or linear functions.
- The package (and parts of it) can be called from regular Python code, which makes it easy to automatically generate problem sets with solutions.
- It is not only free but also completely open-source.

Functionality and Usage

A typical use case of the `beambending` package always involves creating an instance of the `Beam` class. The class constructor takes an optional `length` argument, which defaults to 10 in case no argument is provided.

```
from beambending import Beam
beam = Beam(9) # Initialize a Beam object of length 9m
```

After a Beam object is created, the properties corresponding to the x-coordinates of the pinned and rolling supports must be defined.

```
beam.pinned_support = 2 # x-coordinate of the pinned support
beam.rolling_support = 7 # x-coordinate of the rolling support
```

Note that the Beam class currently supports only statically determined beams with (exactly) one pinned and one roller support.

Each load applied to the beam requires an instance of one of the load classes `DistributedLoadH`, `DistributedLoadV`, `PointLoadH`, or `PointLoadV`. The load classes are simply *namedtuples*, and make the resulting scripts easier to read by making the user's intention explicit. The symbolic variable `x`, also defined by the module, is used for defining variable distributed loads.

```
from beambending import DistributedLoadV, PointLoadH, PointLoadV, x
```

The loads can be applied to the Beam by passing an iterable (list or tuple) to the method `add_loads`.

```
beam.add_loads((
    # 10kN pointing right, at x=3m
    PointLoadH(10, 3),
    # 20kN downwards, at x=3m
    PointLoadV(-20, 3),
    # 10 kN/m, downward, for 3m <= x <= 9m
    DistributedLoadV(-10, (3, 9)),
    # Variable load, for 0m <= x <= 2m
    DistributedLoadV(-20 + x**2, (0, 2)),
))
```

After the problem is fully defined (beam length + placement of supports + loads), the `plot` method can be invoked to plot a sketch of the loaded beam together with its corresponding load diagrams (normal force, shear force and bending moment).

```
fig = beam.plot()
```

The `plot` method is actually a wrapper that combines these four methods: `plot_beam_diagram`, `plot_normal_force`, `plot_shear_force` and `plot_bending_moment` into a single A4-sized printer-friendly plot.

Example

The following example, provided within the [package documentation](#), summarizes the explanation above. The output is shown in Figure 1.

```
from beambending import Beam, DistributedLoadV, PointLoadH, PointLoadV, x
beam = Beam(9)
beam.pinned_support = 2
```

```

beam.rolling_support = 7
beam.add_loads((
    PointLoadH(10, 3),
    PointLoadV(-20, 3),
    DistributedLoadV(-10, (3, 9)),
    DistributedLoadV(-20 + x**2, (0, 2)),
))
fig = beam.plot()

```

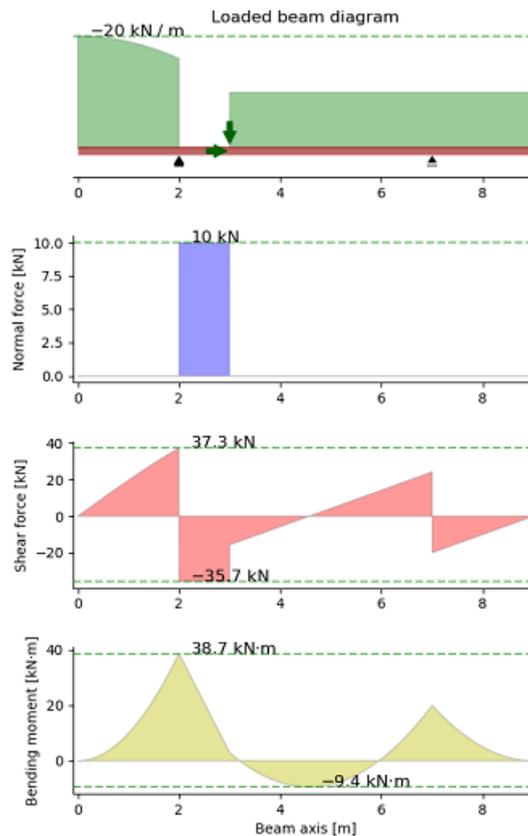


Figure 1: Output corresponding to the example code

Recent Uses

The `beambending` package was developed as a teaching aid for the *Statics* course in the Autumn 2019 semester at OsloMet - Oslo Metropolitan University. Beta versions were tried and commented by students during the Autumn 2018 semester, but the effectiveness of the tool has not been tested with large groups of students yet.

References

- Beam Calculator Online. (2019). <http://rascheta.net/beamuk/>; Accessed on 2019-08-22.
- Beam Guru. (2019). <http://beamguru.com>; Accessed on 2019-08-22.

- Beer, F., Russell Johnston, E., Dewolf, J. T., & Mazurek, D. (2017). *Statics and Mechanics of Materials* (Second Edition.). McGraw-Hill Education.
- Bell, K. (2015). *Konstruksjonsmekanikk - Del I: Likevektslære*. Fagbokforlaget.
- Engineers Edge. (2019). <https://www.engineersedge.com/>; Accessed on 2019-08-22.
- Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, 9(3), 90–95. doi:10.1109/MCSE.2007.55
- MechaniCalc. (2019). <https://mechanicalc.com/calculators/beam-analysis/>; Accessed on 2019-08-22.
- Meurer, A., Smith, C. P., Paprocki, M., Čertík, O., Kirpichev, S. B., Rocklin, M., Kumar, A., et al. (2017). SymPy: symbolic computing in Python. *PeerJ Computer Science*, 3, e103. doi:10.7717/peerj-cs.103
- Oliphant, T. E. (2006). *A guide to NumPy* (Vol. 1). Trelgol Publishing USA.
- SkyCiv Beam. (2017). <https://skyciv.com/structural-software/beam-analysis-software/>; Accessed on 2019-08-22.
- Steel Beam Calculator. (2019). <https://www.steelbeamcalculator.com/>; Accessed on 2019-08-22.
- Structural Beam Deflection and Stress Calculators. (2019). <https://www.amesweb.info/StructuralBeamDeflection/BeamDeflectionCalculators.aspx>; Accessed on 2019-08-22.
- WebStructural. (2019). <https://webstructural.com/shear-and-moment-diagram.html>; Accessed on 2019-08-22.