The Journal of Open Source Education

IndeterminateBeam: A Python package for solving 1D indeterminate beams

Jesse Bonanno[1]

1 No affiliation

DOI: 10.21105/jose.00111

Software

- Review
- Repository
- Archive

Submitted: 18 January 2021
Published: 30 May 2021

Summary

`IndeterminateBeam` is a Python package aiming to serve as a foundation for civil and structural engineering projects in Python. The package can also serve as a standalone program and is useful for determining:

- reaction forces for indeterminate beams
- internal forces for indeterminate beams (shear, bending, axial)
- deflection of beams due to resulting forces
- axial force, shear force, bending moment and deflection diagrams

The module is based primarily on engineering concepts of statics as described in (Hibbeler, 2013), and Python packages Sympy (Meurer et al., 2017) and Matplotlib (Hunter, 2007). The package documentation provides a brief overview of the theory behind the solutions used to calculate the forces on the indeterminate beam.

The package can be used by:

- teachers who want to generate problems
- students who want to verify solutions
- students who want to observe the effect minor geometry changes can have on forces
- students, teachers or engineers who want to create higher order engineering projects using this project as a starting point

The `IndeterminateBeam` package repository can be found on Github and is ready for installation using `pip`. A text-based example of the package can be found on this Jupyter Notebook and a web-based graphical user interface (GUI) is available at https://indeterminate-beam.herokuapp.com/.

Statement of Need

Statics is fundamental to many fields of engineering such as civil, structural and mechanical engineering. This package aims to help student understanding in two ways:

1. Explain the background theory used to solve the indeterminate beam briefly in the package documentation
2. Provide a software solution that allows students to recieve immediate visual feedback on changes a beam system can have on internal and external forces

This Python package was heavily inspired by beambending (Carella, 2019), an educational module created by Alfredo Carella of the Oslo Metropolitan University. The beambending module, although well documented, can only solve for simply supported beams consisting of a pin and roller support. The package documentation for this project includes a more rigorous overview of the theory behind the basics for solving determinate structures. A feature comparison in Table 1 below has been taken from Carella (2019) and modified to include more packages and features.

	Arbitrary distributed load functions	Full GUI (no code required)	Arbitrary number of loads	Free	Open Source	Featured theory module	Detailed solution procedure	Programmable interface	Spring Supports	Any DOF combination for Supports	Any number of supports
IndeterminateBeam	✓	✓	✓	✓	✓	✓	✗	✓	✓	✓	✓
BeamBending	✓	✗	✓	✓	✓	✓	✗	✓	✗	✗	✗
SymBeam	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗
SymPy	✓	✗	✓	✓	✓	✗	✗	✓	✗	✗	✓
Beam Calculator Online	✗	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗
Beam Guru	✗	✓	✓	✗	✗	✗	✓	✗	✗	✗	✓
MechaniCalc	✗	✓	✓	✗	✗	✓	✗	✗	✗	✓	✓
SkyCiv Beam	✗	✓	✓	✗	✗	✗	✓	✗	✗	✗	✗
Steel Beam Calculator	✗	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗
Structural Beam Calculator	✗	✓	✗	✓	✗	✗	✗	✗	✗	✗	✗
WebStructural	✗	✓	✓	✗	✗	✗	✗	✗	✗	✗	✓
ClearCalcs	✗	✓	✓	✗	✗	✗	✗	✗	✓	✗	✓

Table 1: Summary of feature comparison with existing packages

There are six main strengths for the `IndeterminateBeam` package:

- Arbitrary distributed load functions are accepted
- The package features a full GUI web-application
- It is free and open source
- Spring supports can be modelled
- Any degree of freedom combination can be constructed for supports
- Indeterminate Beams can be solved (Any number of supports)

Although most observed tools share some of these features, no other tool shares more than three of the listed strengths with `IndeterminateBeam`.

Functionality and Usage

A typical use case of the `IndeterminateBeam` package involves the following steps:

1. Create a `Beam` object
2. Create `Support` objects and assign to `Beam`
3. Create `Load` objects and assign to `Beam`
4. Solve for forces on `Beam` object
5. Plot results

You can follow along with the example below in this web-based Jupyter Notebook.

The units used throughout the Python package are the base SI units (newtons and metres), but can be updated using the `update_units` method. Units and load direction conventions are described in the package documentation.

Creating a Beam

The creation of a `Beam` instance involves the input of the beam length (m) and optionally the input of the Young's Modulus (E), second moment of area (I), and cross-sectional area (A). E, I and A are optional and by default are the properties of a steel 150UB18.0. For a beam with constant properties, these parameters will only affect the deflections calculated and not the distribution of forces, unless spring supports are specified.

```
from indeterminatebeam import Beam
# Create 7 m beam with E, I, A as defaults
beam = Beam(7)
# Create 9 m beam with E, I, and A assigned by user
beam_2 = Beam(9, E=2000, I =10**6, A = 3000)
```

Defining Supports

`Support` objects are created separately from the `Beam` object, and are defined by an x-coordinate (m) and the beams translational and rotational degrees of freedom.

Degrees of freedom are represented by a tuple of 3 booleans, representing the x , y , and m directions respectively. A `1` indicates the support is fixed in a direction and a `0` indicates it is free. Optionally, stiffness can be specified in either of the translational directions, which overrides the boolean specified.

```
from indeterminatebeam import Support
# Defines a pin support at location x = 5 m
a = Support(5, (1,1,0))
# Defines a roller support at location x = 0 m
b = Support(0, (0,1,0))
# Defines a fixed support at location x = 7 m
c = Support(7, (1,1,1))
# Assign the support objects to a beam object created earlier
beam.add_supports(a,b,c)
```

Defining loads

`Load` objects are created separately from the `Beam` object, and are generally defined by a force value and then a coordinate value, however this varies slightly for different types of loading classes.

```
from indeterminatebeam import PointLoadV, PointTorque, DistributedLoadV
# Create a 1000 N point load at x = 2 m
load_1 = PointLoadV(1000, 2)
# Create a 2000 N/m UDL from x = 1 m to x = 4 m
load_2 = DistributedLoadV(2000, (1, 4))
# Defines a 2 kN.m point torque at x = 3.5 m
load_3 = PointTorque(2*10**3, 3.5)
# Assign the load objects to the beam object
beam.add_loads(load_1,load_2,load_3)
```

Solving for Forces

Once the `Beam` object has been assigned with `Load` and `Support` objects it can then be solved. To solve for reactions and internal forces we call the analyse function.

```
beam.analyse()
```

Plotting results

After the beam has been analysed we can plot the results.

```
fig_1 = beam.plot_beam_external()
fig_1.show()

fig_2 = beam.plot_beam_internal()
fig_2.show()
```

The `plot_beam_external` and `plot_beam_internal` methods collate otherwise seperate plots.

The script above produces the following figures:

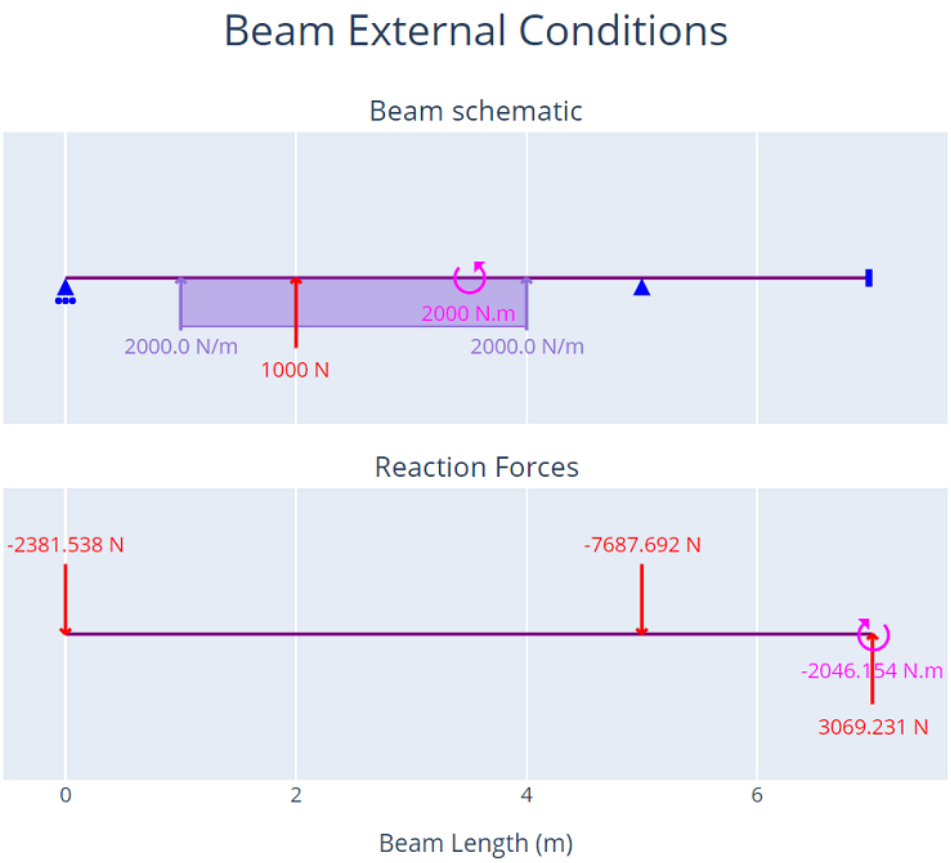

Figure 1: Beam Schematic and Reactions

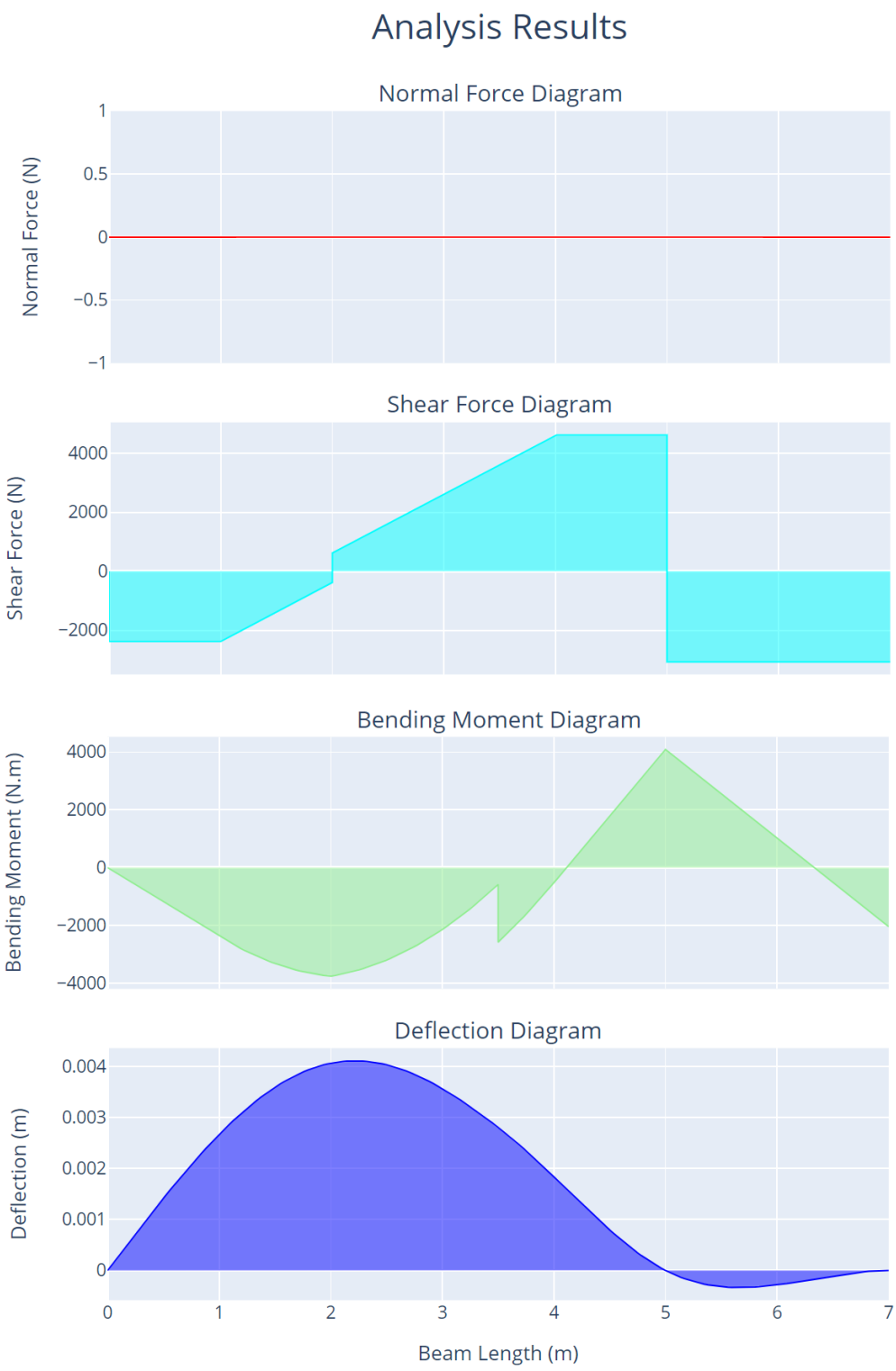

Figure 2: Analysis Results

References

Beam Calculator Online. (2020). http://rascheta.net/beamuk/; Accessed on 2020-12-21.

Beam Guru. (2020). http://beamguru.com; Accessed on 2020-12-21.

Carella, A. (2019). BeamBending: A teaching aid for 1-d shear force and bending moment diagrams. *Journal of Open Source Education*, *2*(16), 65. https://doi.org/10.21105/jose.00065

Carneiro, A. (2021). *SymBeam*. https://pypi.org/project/symbeam/; Accessed on 2021-04-07.

Hibbeler, R. (2013). *Mechanics of materials*. P.Ed Australia. ISBN: 9810694369

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, *9*(3), 90–95. https://doi.org/10.1109/MCSE.2007.55

MechaniCalc. (2019). https://mechanicalc.com/calculators/beam-analysis/; Accessed on 2020-12-21.

Meurer, A., Smith, C. P., Paprocki, M., Čertík, O., Kirpichev, S. B., Rocklin, M., Kumar, A., Ivanov, S., Moore, J. K., Singh, S., Rathnayake, T., Vig, S., Granger, B. E., Muller, R. P., Bonazzi, F., Gupta, H., Vats, S., Johansson, F., Pedregosa, F., … Scopatz, A. (2017). SymPy: symbolic computing in Python. *PeerJ Computer Science*, *3*, e103. https://doi.org/10.7717/peerj-cs.103

SkyCiv Beam. (2017). https://skyciv.com/structural-software/beam-analysis-software/; Accessed on 2020-12-21.

Steel Beam Calculator. (2020). https://www.steelbeamcalculator.com/; Accessed on 2020-12-21.

Structural Beam Deflection and Stress Calculators. (2020). https://www.amesweb.info/StructuralBeamDeflection/BeamDeflectionCalculators.aspx; Accessed on 2020-12-21.

WebStructural. (2020). https://webstructural.com/shear-and-moment-diagram.html; Accessed on 2020-12-21.