A *Mathematica* Package for the Static Analysis of Straight Euler–Bernoulli Beams

LEVENT KITIŞ
Department of Mechanical, Aerospace, and Nuclear Engineering
University of Virginia
Charlottesville, Virginia 22903

INTRODUCTION

This document describes a *Mathematica* package for calculating shear force, bending moment, slope, and deflection functions for a straight Euler–Bernoulli beam in static equilibrium. The program is flexible enough to solve most of the symbolic, numerical, and graphical exercises concerning beam deflections in the typical syllabus of a first course in engineering solid mechanics. The solution of problems using the package requires some *Mathematica* programming, an understanding of singularity functions, and a knowledge of the underlying physical principles. *Mathematica* is documented in Wolfram (1991) and the essential *Mathematica* programming constructs are explained in Maeder (1991). Popov (1990) covers the necessary background material in solid mechanics.

METHOD OF SOLUTION

The beam coordinate system is shown in Figure 1. The deflection of the beam is denoted by w. The distance x is measured along the axis of the beam from the left end at point O and $0 \leq x \leq L$. Downward deflections are negative and the slope of the elastic curve at a point $x = a$ is positive if the derivative of w with respect to x is positive at $x = a$. Figure 2 shows the sign convention for the bending moment M and the shear force V. Although external loads acting downward on the beam are negative according to the beam coordinate system in Figure 1, the program input and output of downward loads are positive. Applied concentrated couples in the counterclockwise direction are positive.

The relation between the shear force V and the load p given as a function of x over the open interval $0 < x < L$ is

$$V(x) = -\int p(x)\,dx + C_1 \tag{1}$$

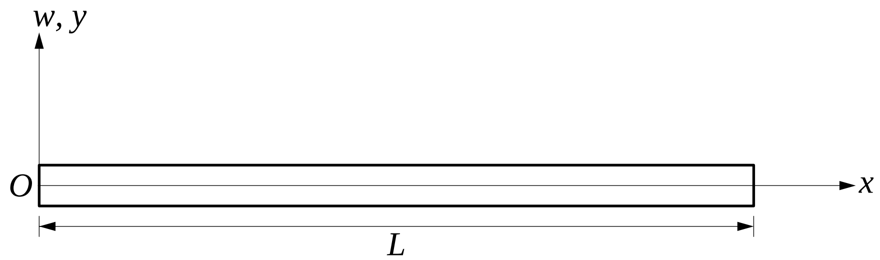

Figure 1 Beam Coordinate System

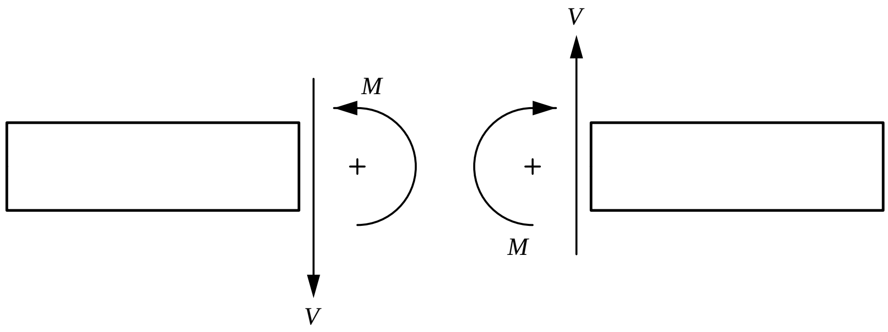

Figure 2 Positive Shear and Positive Bending Moment

where C_1 is a constant of integration. The bending moment M is found by integrating the shear force V

$$M(x) = \int V(x)\, dx + C_2 \tag{2}$$

The slope and the deflection are obtained by two more integrations

$$w'(x) = \int \frac{M(x)}{EI(x)}\, dx + C_3 \tag{3}$$

$$w(x) = \int w'(x)\, dx + C_4 \tag{4}$$

where $EI(x)$ is the flexural rigidity of the beam.

The applied load $p(x)$ is conveniently expressed in terms of singularity functions. The unit step function is defined by

$$\langle\, x - a \,\rangle^0 = \begin{cases} 0 & x < a \\ 1 & x \geq a \end{cases} \tag{5}$$

and the Dirac delta function is denoted by $\langle\, x - a \,\rangle^{-1}$. The contribution of a concentrated force of magnitude P acting at $x = a$ to the loading function $p(x)$ is

$$p_1(x) = P \,\langle\, x - a \,\rangle^{-1} \tag{6}$$

The contribution of a linearly varying distributed load of intensity q_a at $x = a$ and q_b at $x = b$, with $b > a$, to $p(x)$ is

$$p_2(x) = \frac{q_a\,(b - x) - q_b\,(a - x)}{b - a} \;\left(\langle\, x - a \,\rangle^0 - \langle\, x - b \,\rangle^0\right) \tag{7}$$

A load whose intensity is an arbitrary function $f(x)$ over the part of the span from $x = a$ to $x = b$ can be expressed as

$$p_3(x) = f(x) \;\left(\langle\, x - a \,\rangle^0 - \langle\, x - b \,\rangle^0\right) \tag{8}$$

The contribution of concentrated couple of magnitude M_a acting at $x = a$ is written as

$$p_4(x) = M_a \,\langle\, x - a \,\rangle^{-2} \tag{9}$$

where the singularity function is the derivative of the Dirac delta function with respect to x. The presence of a hinge, or a shear connection, at $x = a$ can be accounted for by adding

$$p_5(x) = \Delta\theta \,\langle\, x - a \,\rangle^{-3} \tag{10}$$

to $p(x)$, and a moment connection at $x = a$ contributes

$$p_6(x) = \Delta V \langle x - a \rangle^{-4} \tag{11}$$

The total loading function $p(x)$ is obtained by adding all contributions $p_k(x)$, $1 \leq k \leq 6$.

The program is designed to accept step changes in the flexural rigidity of the beam. If the flexural rigidity has the value EI_1 between $x = a_1 = 0$ and $x = a_2$, EI_2 between $x = a_2$ and $x = a_3$, and so on until the final value EI_n between $x = a_n$ and $x = L$ is reached, then the function $1/EI(x)$ in Eq. (3) can be written as

$$\frac{1}{EI(x)} = \frac{1}{EI_1} + (\frac{1}{EI_2} - \frac{1}{EI_1}) \langle x - a_2 \rangle^0 + \ldots + (\frac{1}{EI_n} - \frac{1}{EI_{n-1}}) \langle x - a_n \rangle^0 \tag{12}$$

In this expression, the step change locations a_k are ordered such that $a_1 = 0$ and $a_k > a_{k-1}$ for $2 \leq k \leq n$. When the reciprocal of flexural rigidity given in Eq. (12) is substituted into Eq. (3), the integrals appearing in Eqs. (3) and (4) contain products of singularity functions multiplied by ordinary functions of x. If there are several step changes in flexural rigidity and $M(x)$ is not a simple function, the calculation of these integrals can become long and tedious. However, the *Mathematica* package `DiracDelta` in the `Calculus` directory contains the rules needed to compute such integrals; this makes the use of singularity functions feasible for nonprismatic beams.

THE BEAM STATICS PACKAGE

The *Mathematica* package `BeamStatics` has three separate functions

(i) Given the loading $p(x)$, the flexural rigidity $EI(x)$, and the types of support at $x = 0$ and $x = L$, determine the shear force $V(x)$, the bending moment $M(x)$, the rotation, or slope, $w'(x)$, and the elastic curve, or deflection, $w(x)$.

(ii) Plot $V(x)$, $M(x)$, $w'(x)$, and $w(x)$ versus x, if these functions contain no symbolic parameters.

(iii) Draw a diagram of the beam and the loads acting on it.

The first of these three functions is carried out by the *Mathematica* program `Beam`, the second by `plots` and `plotm`, and the third by `DrawBeam`.

USER'S GUIDE TO **Beam**

The call to `Beam` has the form

```
Beam[x, L, EI, bnd, options]
```

The argument `x`, which has the head `Symbol`, defines the symbol to be used for the distance measured from the left end of the beam. The argument `L` is the total length of the beam and `EI` specifies the flexural rigidity. For a beam with constant flexural rigidity, `EI` is a single symbol or number. For a beam with step changes in flexural rigidity, `EI` is a list. For instance, for a beam of total length L, the definition

```
EI = {e, 0, 2 e, L/3, 3 e, 2 L/3}
```

specifies that the flexural rigidity is e between $x = 0$ and $x = L/3$, $2\,e$ between $x = L/3$ and $x = 2\,L/3$, and $3\,e$ between $x = 2\,L/3$ and $x = L$. The lengths in `EI` are given in ascending order starting with `0`.

The argument `bnd` is a list of two strings defining the types of support at the left end, at $x = 0$, and the right end, at $x = L$, of the beam. The elements of the list `bnd` can be `"Free"`, `"Guided"` `"Pinned"`, or `"Fixed"`. Thus, the definition

```
bnd = {"Fixed", "Pinned"}
```

specifies that the support at $x = 0$ is a fixed or clamped support and the one at $x = L$ is a roller or pinned support.

The loading functions are defined by the argument `options`, which is a sequence of zero or more entries, each with head `Rule`, chosen from the following list of options for `Beam`:

`PointLoad` The rule `PointLoad -> {P, a, Q, b,...}` places concentrated forces of magnitude P at $x = a$ and Q at $x = b$, etc. Upward forces have negative magnitude. A force applied at a free end is included here. A force acting at a free end contributes nothing to the loading function $p(x)$ but its presence creates a nonhomogeneous boundary condition.

`LineLoad` The rule `LineLoad -> {p, a, q, b, r, c, s, d,...}` places linearly distributed loads of intensity — force per unit length — p at $x = a$, q at $x = b$ and r at $x = c$, s at $x = d$, etc. Here $b > a$ and $d > c$. A frequent particular case is the uniformly distributed line load, which is defined by specifying identical force intensities. For example, `LineLoad -> {p, L/2, p, L}` defines a uniform line load of intensity p over the right half span of a beam of length L.

`InSpan` The rule `InSpan -> a` indicates that there is a rigid in-span support at $x = a$. `InSpan -> {a, b,...}` indicates that there are rigid in-span supports at $x = a$, $x = b$, etc.

`PointCouple` The rule `PointCouple -> {K, a, L, b,...}` places concentrated moments of magnitude K at $x = a$ and L at $x = b$, etc. Clockwise moments have negative magnitude. A couple applied at a free end or a pinned support is included here. A couple at a free end or a pinned support contributes nothing to the loading function $p(x)$ but its presence creates a nonhomogeneous boundary condition.

`LoadFunction` The rules `LoadFunction -> f[x]` or `LoadFunction -> {f[x], g[x], ...}` add loading functions $f(x)$, $g(x)$, etc. to the loading function $p(x)$. These functions can represent any loading other than those that are definable by the `PointLoad`, `LineLoad`, and `PointCouple` options. The main use of this option is to specify nonlinear distributed loads. The functions f, g etc. must be defined in a form that can be evaluated at any point over the span. For example, each one of the rules

```
 LoadFunction -> q Sin[Pi x/L](UnitStep[x - L/2] - UnitStep[x - 2 L/3])
LoadFunction -> q Sin[Pi x/L]{UnitStep[x - L/2], - UnitStep[x - 2 L/3]}
```

specifies the same sinusoidally varying distributed load. This function is nonzero only between $x = L/2$ and $x = 2\,L/3$ and can be evaluated by the program at any point x, $0 \leq x \leq L$.

ShearConnection The rule `ShearConnection -> a` indicates that there is a shear connection, or hinge, at $x = a$. `ShearConnection -> {a, b, ...}` indicates that there are hinges at $x = a$, $x = b$, etc. A hinge transmits shear and cannot resist bending moment. The slope of the elastic curve has a jump discontinuity at a hinge. The introduction of a hinge into a statically determinate beam causes the beam to become unstable.

MomentConnection The rule `MomentConnection -> a` indicates that there is a moment connection at $x = a$. `MomentConnection -> {a, b, ...}` indicates that there are moment connections at $x = a$, $x = b$, etc. A moment connection transmits moment but not shear. This introduces a jump discontinuity into the elastic deflection curve.

The rules defined by `urule` are used within `Beam` to simplify expressions containing unit step functions. These rules may also be used outside the package; `urule[x, a]` replaces `UnitStep[x]` with `1` and `UnitStep[k a]` with `UnitStep[k]` if `k` is a number. The first of these rules always holds because the problem is formulated for $x \geq 0$. The second rule holds only if $a > 0$. It is used in `Beam` to evaluate `UnitStep[k L]` where k is a number and L is the length of the beam.

If the beam has no in-span supports, the list returned by `Beam` is `{V, M, S, W}` where `V`, `M`, `S`, and `W` are the expressions for shear force, bending moment, slope and deflection, respectively. If there are in-span supports, the output list has a fifth entry, which is a list of lists with two members, giving the in-span reaction forces and their points of application on the beam. For example, if this fifth entry is `{{P, a}, {Q, b}}`, the in-span support reaction is P at $x = a$ and Q at $x = b$. A reaction force is returned with a negative sign if it is directed upward. All occurrences of Dirac delta functions and their derivatives are removed from `V`, `M`, `S` and `W` at the end of the calculations. Such terms are used only as tracers during the integrations (Popov, 1990).

THE PLOTTING ROUTINES

The functions `plots` and `plotm`, which are identical except for default settings, plot two curves in two separate frames on one page. The defaults in `plots` are set for deflection and slope curves. The defaults in `plotm` are for bending moment and shear force curves. A call to `plots` or `plotm` has the form

```
plots[f1, f2, x, x0, xf, options]
plotm[f1, f2, x, x0, xf, options]
```

where `f1` is the expression to be plotted in the upper frame and `f2` is the expression to be plotted in the lower frame. The argument `x`, which has the head `Symbol`, is the symbol used for the distance measured from the left end of the beam. Plots of `f1` and `f2` as functions of `x` from `x0` to `xf` are generated by `plots` or `plotm`. The argument `options` is a sequence of zero or more entries, each with head `Rule`, chosen from the following list of options for `plots` and from the options of `Plot`:

XTitle This option assigns the x–axis title string for both plots. The default is `XTitle -> "x"`.

Y1Title This option assigns the y–axis title string for the curve plotted in the upper frame. The default is `Y1Title -> "DEFLECTION"` for `plots` and `Y1Title -> "BENDING MOMENT"` for `plotm`.

Y2Title This option assigns the y–axis title string for the curve plotted in the lower frame. The default is `Y2Title -> "SLOPE"` for `plots` and `Y2Title -> "SHEAR FORCE"` for `plotm`.

SideTitle This option assigns the label for the right edge of the two plots. The default is the empty string.

TopTitle1 This option assigns the label for the top edge of the upper frame. The default is the empty string.

TopTitle2 This option assigns the label for the top edge of the lower frame. The default is the empty string.

Units `Units -> {"m", "mm", "mm/m"}` specifies that x is measured in meters, the deflection in millimeters, and the slope in millimeters per meter. These are the default values for `plots`, which are obtained if lengths x, L are in m, forces in kN, couples in kNm, load intensities in $\mathrm{kN/m}$, and flexural rigidities in MNm^2 in the input data to `Beam`. The default for `plotm` is `Units -> {"m", "kNm", "kN"}}`. The setting `Units -> None` suppresses the printing of units.

Xmax This option assigns a maximum width to the display area in fictitious length units. The default is `Xmax -> 100`.

Ymax This option assigns a maximum height to the display area in fictitious length units. The default is `Ymax -> 130`.

Spacing The default setting `Spacing -> 0.1` leaves a vertical space of height equal to 10% of the maximum height of the display area between the slope and deflection plots.

PS With the default setting `PS -> None`, plots are displayed on the screen only. `PS -> "slopedef.ps"` saves the plots in PostScript code in the file *slopedef.ps*. The utility *psfix* is used to process *Mathematica*'s PostScript code.

The plotting programs `plotm` and `plots` pass the argument `options` to `Plot`. By including `Plot` options in the argument `options`, users can change the default settings of the package and redesign the appearance of the plots.

USER'S GUIDE TO **DrawBeam**

A diagram of the beam and the loads acting on it is drawn by the call

```
DrawBeam[x, L, EI, bnd, options]
```

The first four arguments `x`, `L`, `EI`, and `bnd` are the same as in `Beam`. For symbolic problems, `L` and `EI` must be given fictitious numerical values so that the drawing can be scaled. `DrawBeam` takes the display area as a rectangle with lower left corner coordinates $(0,\ 0)$ and upper right corner coordinates $(X_m,\ Y_m)$. The default values are $X_m = 100$ and $Y_m = 130$. The default settings of all the sizing options of `DrawBeam` have been set assuming that these X_m and Y_m values are in effect.

The following are the options of `DrawBeam`:

PointLoad The rule `PointLoad -> {P, a, Q, b,...}` places concentrated forces of magnitude P at $x = a$ and Q at $x = b$, etc. `P` and `Q` are used as labels and may be given as numbers or symbols. Upward forces have negative magnitude. A symbol with a negative sign is interpreted as an upward force. The locations `a` and `b` must be given numerically so that the drawing can be scaled, but the option `ScaleTags` can be used to label lengths with symbols. In numerical problems, the values of `a` and `b` are used for labeling lengths. The `LengthUnit` option can be used to append units to these numerical labels.

ScaleTags `DrawBeam` places a length scaling set of double-tipped horizontal arrows above or below the drawing of the beam. These arrows mark the location of flexural rigidity changes, in-span supports, concentrated forces and moments, the start and end points of distributed loads, shear and moment connections, and the two ends of the beam. In a numerical problem, the values of the lengths are used to label the arrows. The `LengthUnit` option can be used to append units to these numerical labels. The rule `ScaleTags -> {"s1", "s2", ...}` labels the first arrow `s1`, the second arrow `s2` etc., starting at the left end and proceeding to the right along the beam axis. The number of strings in the label list must be the same as the number of double-tipped arrows in the scaling line.

LineLoad The rule `LineLoad -> {p, a, q, b, r, c, s, d,...}` places linearly distributed loads of intensity — force per unit length — p at $x = a$, q at $x = b$ and r at $x = c$, s at $x = d$, etc. Here $b > a$ and $d > c$, and all elements are given numerically. If the line loads are to have symbolic labels, the option `LoadTags` can be used to define the label strings. In numerical problems, the values of `p`, `q`, `r`, `s` are used as labels of load intensities, and the values of `a`,`b`, `c`, `d` as labels of lengths. The `LengthUnit` and `ForceUnit` options can be used to append units to the labels. Load intensity units are set to `ForceUnit/LengthUnit`.

LoadTags This is a list of strings specifying labels for the loads defined by `LineLoad`. If `LoadTags` is not given as an option, the load intensities specified by `LineLoad` are used as labels. For constant loads, only one label is given, which is centered horizontally at the midpoint of the load range. For nonconstant linear loads, two labels are given, to be placed at the start and at the end of the loading curve. If, however, the load function is zero at one end, no label can be specified for that end.

InSpan `InSpan -> a` places a rigid in-span support at $x = a$. `InSpan -> {a, b,...}` indicates that there are rigid in-span supports at $x = a$, $x = b$, etc. Pinned supports are drawn at these locations. For symbolic problems, the support locations are given given numerically so that the drawing can be scaled, but the option `ScaleTags` can be used to label lengths with symbols.

PointCouple `PointCouple -> {K, a, L, b,...}` places concentrated moments of magnitude K at $x = a$ and L at $x = b$, etc. Clockwise moments have negative magnitude. The labeling of couples follows the conventions described for point loads in the notes for the `PointLoad` option.

LoadFunction `LoadFunction -> f[x]` or `LoadFunction -> {f[x], g[x], ...}` adds loading functions $f(x)$, $g(x)$, etc. to the loading function $p(x)$. The notes for this option of `Beam` define the syntax of `f[x], g[x]`, etc., but in `DrawBeam` each function should define a loading that extends continuously over a part of the span. The ranges over which the functions are zero must be specified by the `LoadFunctionRange` option. Label strings can be defined by the `LoadFunctionTags` option.

LoadFunctionRange This defines the ranges over which the the list of functions given by the `LoadFunction` option are nonzero. For example, `LoadFunction -> {f[x], g[x]}` requires `LoadFunctionRange -> {{a, b}, {c, d}}` to specify that $f(x)$ is to be drawn between $x = a$ and $x = b$ over the span and $g(x)$ between $x = c$ and $x = d$.

LoadFunctionTags This is a list of strings defining the labels of the loads specified by `LoadFunction`. The labels are centered horizontally at approximately the midpoints of the function ranges.

LoadFunctionTagLoc This defines the vertical distance between the load function curve and its label for functions specified by the `LoadFunction` option. The default is 2.5.

ShearConnection The rule `ShearConnection -> a` indicates that there is a shear connection, or hinge, at $x = a$. `ShearConnection -> {a, b,...}` indicates that there are hinges at $x = a$, $x = b$, etc. `DrawBeam` draws a small filled disk at a hinge and marks its location on the length scaling line drawn below or above the beam. The locations must be given numerically so that the drawing can be scaled, but symbolic labels can be specified by the `ScaleTags` option. For numerical problems, `LengthUnit` can be used to append length units to the hinge location labels.

MomentConnection The rule `MomentConnection -> a` indicates that there is a moment connection at $x = a$. `MomentConnection -> {a, b,...}` indicates that there are moment connections at $x = a$, $x = b$, etc. The symbol drawn at a moment connection is a small rectangle. The labeling is done as described in the notes for `ShearConnection`.

Xmax and Ymax The display area is taken to be `Xmax` units wide, `Ymax` units high. The coordinates of the display area lie in the rectangle with lower left corner at `(0, 0)` and upper right corner at `(Xmax, Ymax)`. The default settings are `Xmax -> 100` and `Ymax -> 130`.

LeftOrigin The coordinates of the left end of the beam axis. The default is `{10, 50}`.

LengthScale `LengthScale -> s` scales all horizontal lengths x to $x\, s/L$ units where L is the actual length of the beam. The default value of `s` is 80.

HeightScale If EI is constant, a rectangle of width `LengthScale` and height `HeightScale` is drawn to represent the beam. If EI is not constant, the portion of the beam with the maximum EI has height `HeightScale`. Heights elsewhere are proportional to the flexural rigidity EI. The default is `HeightScale -> 4`.

BeamLine The `AbsoluteThickness` of the lines in the beam outline. The default is 0.75.

ShearRadius The radius of the filled disk in the symbol for a shear connection. The default is 0.5.

MomentBase The width of the rectangle drawn to represent a moment connection. The default is 1.25.

GrooveRatio This sets the length of the base of the triangular notch that is drawn at moment and shear connections equal to `2 HeightScale/GrooveRatio`. The default setting is `GrooveRatio -> 4`.

ForceHeight The height of the vertical arrows representing concentrated forces. The default is 16.

ForceTagLoc The vertical distance between the origin, for downward forces, or the tip, for upward forces, of the concentrated force vector and its label. The default is 1.

ForceTipLength The height of the filled triangles, or the tips, of the arrows representing concentrated forces. The default is 1.6.

ForceTipHeight `ForceTipHeight -> h` sets the base of the filled triangle, or the tip, of the concentrated force arrows equal to $2\,h$. The default is 0.5.

ForceThickness The `AbsoluteThickness` of the line between the origin and the terminus of concentrated force arrows. The default is 0.75.

LoadHeight The height of the longest arrow among the vertical arrows representing distributed forces. The default is 12.

LoadTipLength The height of the filled triangles, or the tips, of the arrows representing distributed forces. The default is 1.6.

LoadTipHeight `LoadTipHeight -> h` sets the base of the filled triangle, or the tip, of the distributed force arrows equal to $2\,h$. The default is 0.5.

LoadThickness The `AbsoluteThickness` of the line between the origin and the terminus of distributed force arrows. The default is 0.75.

LoadSubdivisions The number of subdivisions of the maximum horizontal range of the loads defined by `LineLoad`. The other ranges are subdivided such that the length of the subdivisions approximates that of the load with the maximum range. This determines the horizontal spacing of the vertical arrows drawn to represent distributed loads. The default setting is 5.

LoadFunctionSubdivisions This serves the same purpose as `LoadSubdivisions` for the loads defined by `LoadFunction`. The default setting is 5.

LengthUnit The string to be used as a length unit in labeling distributed loads, concentrated couples, and locations marked by the scaling line. The default is `LengthUnit -> "m"`. Distributed loads are labeled `ForceUnit/LengthUnit`, concentrated couples `ForceUnit LengthUnit`.

ForceUnit The string to be used as a force unit in labeling concentrated loads, distributed loads, and concentrated couples. The default is `ForceUnit -> "kN"`.

ScaleNear Vertical lines passing through the tips of the double-tipped arrows of the scaling line — see `ScaleTags` — are drawn to mark length dimensions. The distance between the lower beam line and the upper end of these vertical lines is defined by `ScaleNear`. If `ScaleLevel` is negative, which indicates that the scaling line is to be drawn above the beam, then `ScaleNear` specifies the distance between the upper beam line and the lower end of the vertical lines. The default setting is `ScaleNear -> 1`.

ScaleFar This gives the distance between the lower or upper beam line and the far end of the vertical lines that mark the scaling line (see `ScaleNear`). The default setting is `ScaleFar -> 15`.

ScaleLevel If `s` is positive, `ScaleLevel -> s`, places the scaling line at a vertical distance `s` below the lower beam line. If `s` is negative, the scaling line is at a vertical distance `s` above the upper beam line. The default setting is `ScaleLevel -> 14`.

ScaleTagLoc If s is positive, `ScaleTagLoc -> s`, places the scaling line labels at a vertical distance s above the double-tipped arrows. If s is negative, the labels are placed below the arrows. The default setting is `ScaleTagLoc -> 1.25`.

ArrowThickness This sets the `AbsoluteThickness` of the double-tipped arrows in the scaling line. The default is 0.25.

TipLength This sets the height of the filled triangles, or the tips, of the double-tipped arrows in the scaling line. The default is 1.6.

TipHeight `TipHeight -> h` sets the base of the filled triangle, or the tip, of the scaling line arrows equal to $2\,h$. The default is 0.3.

ArcRadius The radius of the circular arrow drawn to represent a concentrated moment. The default is 2.

ArcTipHeight `ArcTipHeight -> h` sets the base of the filled triangle, or the tip, of the concentrated moment arrows equal to $2\,h$. The default is 0.5.

ArcTipLength The height of the filled triangles, or the tips, of the circular arrows representing concentrated moments. The default is 1.6.

ArcThickness The `AbsoluteThickness` of the circular arc for concentrated couples. The default is 0.75.

ArcTagLoc `ArcTagLoc -> {a, b}` centers the label for the concentrated moments at the coordinates $(a,\, b)$ measured from the center of the circular arc. The default is `ArcTagLoc -> {0, -3}`.

ArcAngles `ArcAngles -> {a, b}` sets the beginning and ending angles, in degrees, of the circular arc drawn to represent a concentrated moment. The default setting is `{100, 270}`.

RectThickness The `AbsoluteThickness` of the lines for drawing the rectangles that represent moment connections. The default is 0.75.

RectTheta This specifies, in degrees, the angle by which the rectangles that represent moment connections are to be rotated counterclockwise from the positive x–axis. The default is 0.

PinHeight The height of the triangle drawn to represent a pinned support. The default is 2.5.

PinWidth The half base length of the triangle drawn to represent a pinned support. The default is 1.

BaseLength The length of the fixed horizontal wall line drawn at a pinned support. The default is 3.

PinLine The `AbsoluteThickness` of the lines for the triangle drawn to represent a pinned support. The default is 0.75.

BaseDivisions The number of subdivisions of the line whose length is specified by the `BaseLength` option. Short lines drawn at the subdivisions indicate a fixed wall. The default is 6.

RollerRadius The radius of the circles drawn as part of a guided support. The default is 0.5.

GuideWall The number of subdivisions of the line indicating a fixed vertical wall in a guided support. This determines the number of short lines drawn at the wall. The default is 6.

RollerLine The `AbsoluteThickness` of the circles in a guided support. The default is 0.75.

WallDivisions The number of subdivisions of the line indicating a fixed vertical wall in a fixed support. This determines the number of short lines drawn at the wall. The default is 8.

WallThickness The `AbsoluteThickness` of the wall line in fixed, guided, and pinned supports. The default is 1.5.

TickThickness `TickThickness -> t` sets the `AbsoluteThickness` of the short lines in a fixed wall symbol equal to `WallThickness/t`. The setting affects the walls for fixed, guided, and pinned supports. The default is 3.

AxisLine `AxisLine -> True` draws a dashed line along the beam axis. The default is `AxisLine -> False`.

EXAMPLES

This section contains some sample problems solved using the *Mathematica* beam statics programs described above.

EXAMPLE 1

The maximum deflection of the prismatic beam of constant flexural rigidity EI shown in Figure 3 is to be determined. The slope of the deflection curve for this beam is a piecewise continuous function, with a jump discontinuity at $x = L$, and the deflection is continuous over the span. The maximum deflection occurs either at a point where the slope is zero in the open interval $L < x < 4\,L$ or at $x = L$. The program used to solve this problem is listed in Table 1.

In this program, `uregion[x, L, expr]` removes all expressions containing `UnitStep` in `expr` and breaks `expr` into parts that define its value over continuous subintervals of $[0,\ L]$. For example, if `expr` is `P UnitStep[x -L/2]`, then `uregion[x, L, expr]` gives

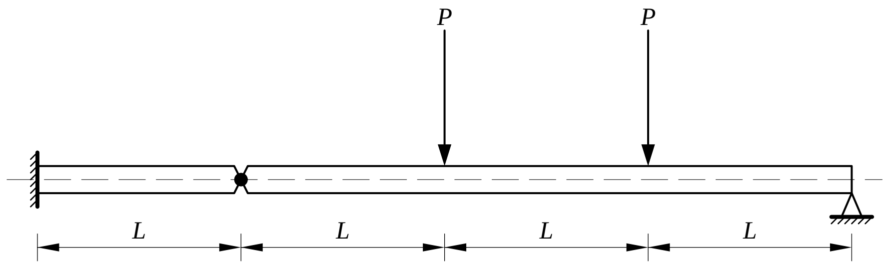

Figure 3 The Beam of Example 1

Table 1 *Mathematica Program for Example 1*

```
{s[x_], u[x_]} = Take[
    Beam[x, 4 L, EI, {"Fixed", "Pinned"},
          PointLoad -> {P, 2 L, P, 3 L}, ShearConnection -> L],
                   -2]

{slope, regions} = uregion[x, 4 L, s[x]]

soln = x /. Solve[#, x] & /@ Thread[slope == 0]

check[list_, interval_] := If[ IntervalMemberQ[Interval[interval], #], #, {}] & /@ list

locations = Flatten[Apply[check, Transpose[{soln, regions}/L], 1]] L

ans := {locations, u /@ locations /. urule[x, L]}

draw[fname_:None] := DrawBeam[x, 80, 1,
         {"Fixed", "Pinned"}, PointLoad -> {P, 40, P, 60}, ShearConnection -> 20,
         LengthUnit -> "", ForceUnit -> "", ScaleTags -> {L, L, L, L},
         ScaleNear -> 3, HeightScale -> 2,
         AxisLine -> True, LengthScale -> 60, ScaleFar -> 5,
         ScaleLevel -> 4, PS -> fname,
         ForceHeight -> 10, PinHeight -> 1.75, PinWidth -> 0.75]
```

`{{0, P}, {{0, L/2}, {L/2, L}}}`. The applicability of `uregion` is restricted to expressions in which the arguments of `UnitStep` terms appear in the same form. For instance, if `UnitStep[x - 0.5 L]` and `UnitStep[x - L/2]` are encountered in the same expression, the subdivisions found by `uregion` will be incorrect. In addition, `uregion` is only able to produce correct results if *Mathematica* can sort the arguments of `UnitStep` when $x = 0$. Thus, these arguments should be numbers or numerical multiples of a single symbol when $x = 0$.

After finding all the zeros of the slope curve, the program eliminates the x values that do not lie in the correct subintervals and the answer is returned by `ans` as

```
{{0, (43 L)/18}, {0, (-733 L^3 P)/(648 EI)}}
```

meaning that the maximum deflection is

$$w_{\max} = \frac{733\,P\,L^3}{648\,E\,I} \qquad \text{at} \qquad x = \frac{43\,L}{18}$$

Figure 3 is drawn by `draw` which calls `DrawBeam` with the arguments and options shown in Table 1.

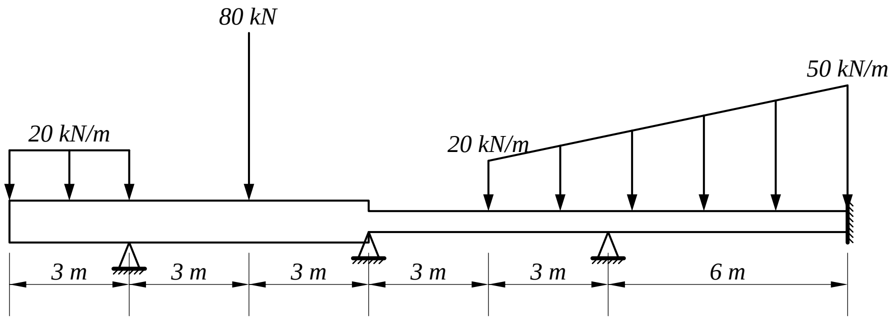

Figure 4 The Beam of Example 2

EXAMPLE 2

In this example, all the loads and material properties are given numerically. The flexural rigidity is $80\,\mathrm{MNm}^2$ between $x = 0$ and $x = 9\,\mathrm{m}$ and $40\,\mathrm{MNm}^2$ over the rest of the span. There are three rigid in-span supports, one concentrated force and two linear loads.

The system is shown in Figure 4 and the *Mathematica* program used to draw this figure and plot the deflection, slope, bending moment and shear diagrams is listed in Table 2. These plots are shown in Figures 5 – 8.

Table 2 *Mathematica Program for Example 2*

```
{V, M, S, U, inspan} = Beam[x, 21, {80, 0, 40, 9}, {"Free", "Fixed"},
    PointLoad -> {80, 6},
    LineLoad -> {20, 0, 20, 3, 20, 12, 50, 21},
    InSpan -> {3, 9, 15}]


draw[fname_:None] := DrawBeam[x, 21, {80, 0, 40, 9}, {"Free", "Fixed"},
    PointLoad -> {80, 6},
    LineLoad -> {20, 0, 20, 3, 20, 12, 50, 21},
    InSpan -> {3, 9, 15},
    ScaleFar -> 7,
    ScaleLevel -> 4,
    ForceTagLoc -> 1.6,
    PS -> fname]

mkplot := (plots[U, S, x, 0, 21, TopTitle1 -> "Example 2", TopTitle2 -> "Example 2", PS -> "exbeam2a.ps"];
           plotm[M, V, x, 0, 21, TopTitle1 -> "Example 2", TopTitle2 -> "Example 2", PS -> "exbeam2b.ps"])
```

EXAMPLE 3

The maximum bending moment is to be determined for the prismatic beam of constant flexural rigidity EI shown in Figure 9. This maximum occurs either at a point b in the open interval $(0,\ 3\,L)$ or at one of the end points $x = 0$, $x = 3\,L$. In the first case, the shear force is zero at $x = b$. The program used to solve the problem is listed in Table 3. The result, returned by `ans` is

```
{{11 L/8, 0, 3 L}, {25 L^2 p/128, -3 L^2 p/4, -9 L^2 p/8}}
```

meaning that the maximum bending moment is

$$M_{\max} = \frac{9\,p\,L^2}{8} \qquad \text{at} \qquad x = 3\,L$$

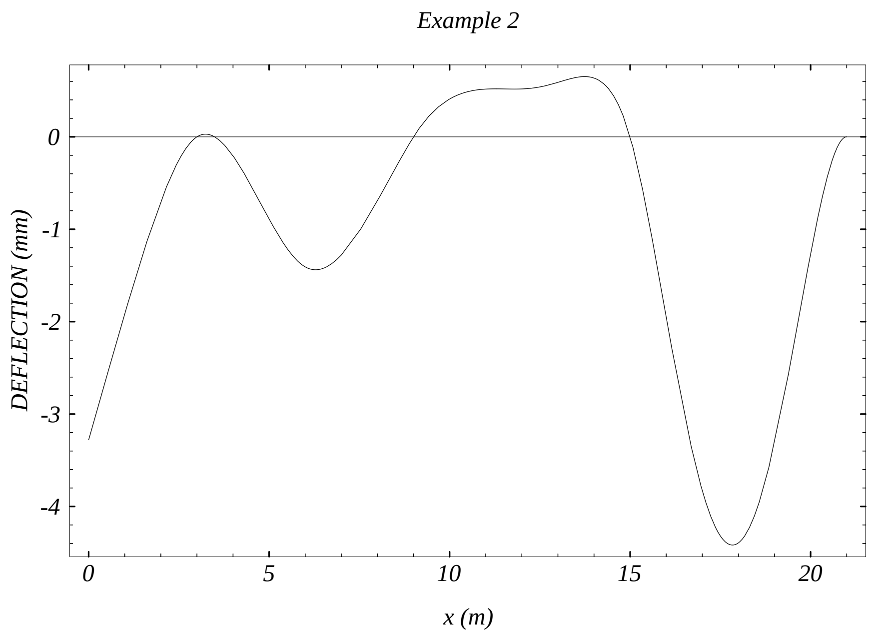

Figure 5 Deflection Curve for Example 2

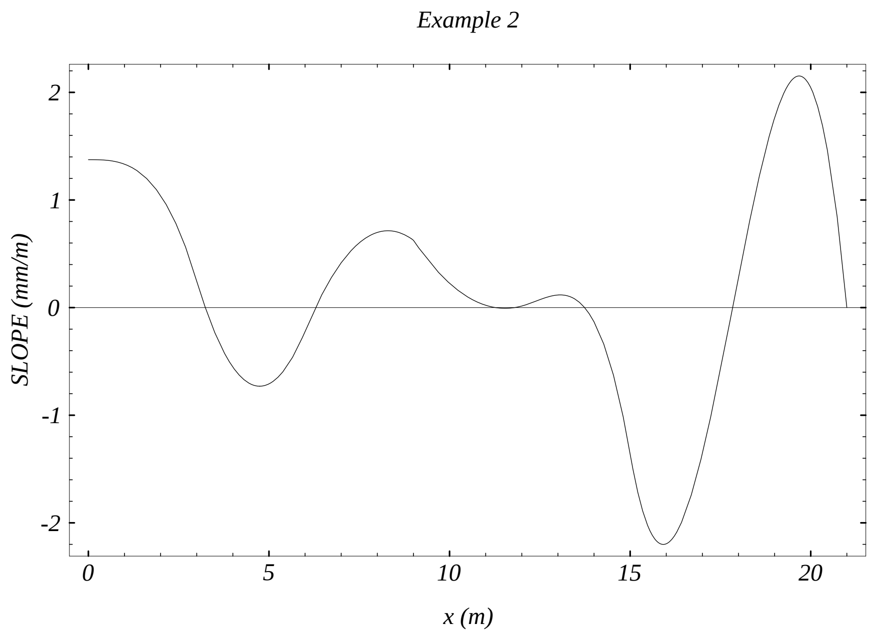

Figure 6 Slope Curve for Example 2

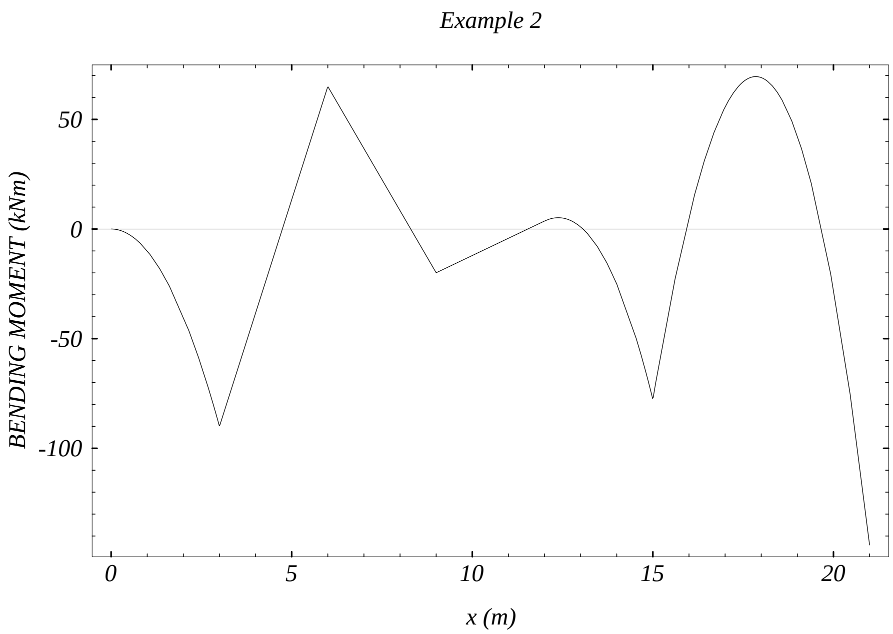

Figure 7 Bending Moment for Example 2

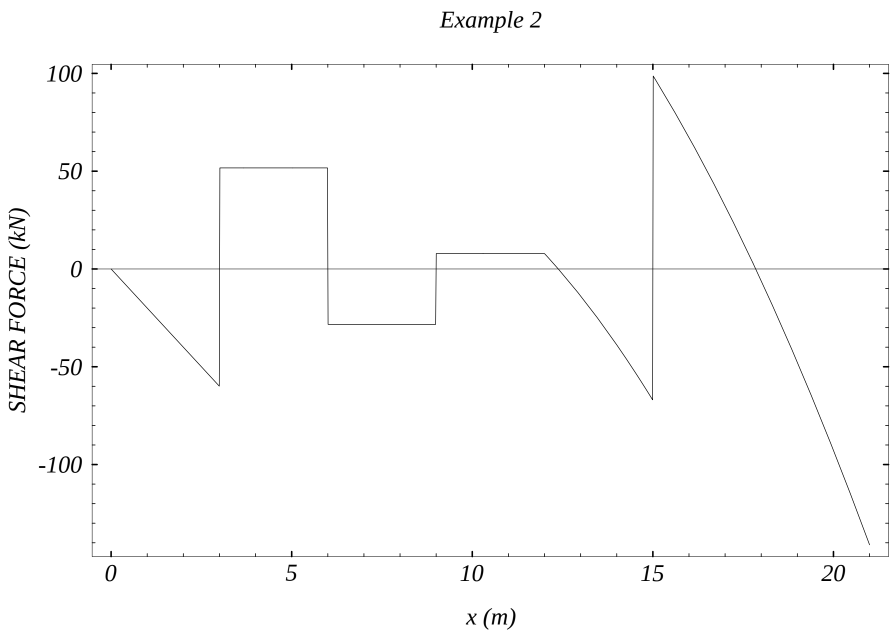

Figure 8 Shear Force for Example 2

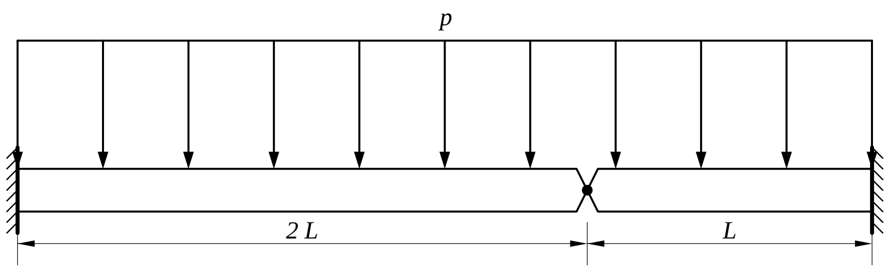

Figure 9 The Beam of Example 3

Table 3 *Mathematica Program for Example 3*

```
{v[x_], m[x_]} = Take[ Beam[x, 3 L, EI,  {"Fixed", "Fixed"},
                            LineLoad -> {p, 0, p, 3 L}, ShearConnection -> 2 L], 2]

{shear, regions} = uregion[x, 3 L, v[x]];

soln = x /. (Solve[#, x] &) /@ Thread[shear == 0];

check[list_, interval_] := If[ IntervalMemberQ[Interval[interval], #], #, {}, {}] & /@ list

locations = Join[ Flatten[Apply[check, Transpose[{soln, regions}/L], 1]] L, {0, 3 L}]

ans := {locations, m /@ locations /. urule[x, L]}

draw[fname_:None] := DrawBeam[x, 60, 1, {"Fixed", "Fixed"},
                             ScaleTags -> {"2 L", "L"},
                             ScaleFar -> 5,
                             ScaleLevel -> 3,
                             ShearConnection -> 40,
                             LoadSubdivisions -> 10,
                             ForceTagLoc -> 1.75,
                             LineLoad -> {1, 0, 1, 60},
                             LoadTags -> {"p"},
                             LengthUnit -> "",
                             ForceUnit -> "",
                             PS -> fname]
```

EXAMPLE 4

The beam of this example, shown in Figure 10, is subjected to the nonlinear distributed load

$$p(x) = \frac{4\,p_0\,x}{L}\left(1 - \frac{x}{L}\right)$$

over the entire span. The program shown in Table 4 determines the slope and the deflection at the left end (`ans`) to be

$$\begin{aligned} w'(0) &= -\frac{p_0\,L^3}{480\,EI} \\ w(0) &= \frac{203\,p_0\,L^4}{368640\,EI} \end{aligned}$$

The bending moment and the deflection at midspan are (`ans2`)

$$\begin{aligned} M(\frac{L}{2}) &= \frac{p_0\,L^2}{48} \\ w(\frac{L}{2}) &= -\frac{181\,p_0\,L^4}{368640\,EI} \end{aligned}$$

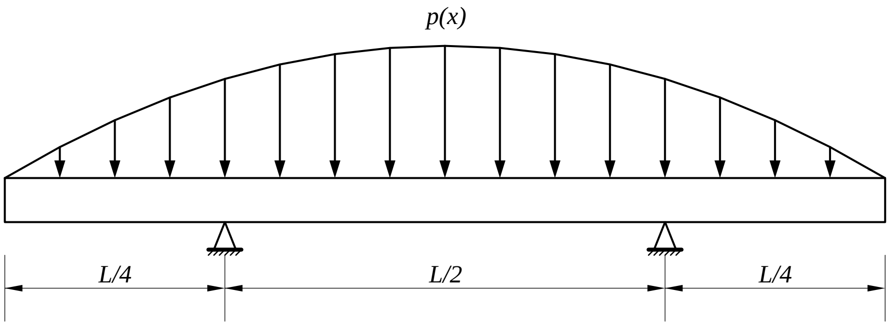

Figure 10 The Beam of Example 4

Table 4 *Mathematica Program for Example 4*

```
{v[x_], m[x_], s[x_], u[x_], inspan} = Beam[x, L, EI,  {"Free", "Free"},
                              LoadFunction -> (4 p x/L)(1 - x/L), InSpan -> {L/4, 3L/4}]

draw[fname_:None] := DrawBeam[x, 80, 1, {"Free", "Free"},
                               InSpan -> {20, 60},
                               ScaleTags -> {"L/4", "L/2", "L/4"},
                               ScaleFar -> 9,
                               ScaleNear -> 3,
                               ScaleLevel -> 6,
                               LoadFunctionSubdivisions -> 16,
                               ForceTagLoc -> 1.75,
                               LoadFunction -> x(1 - x/80),
                               LoadFunctionRange -> {{0, 80}},
                               LoadFunctionTags -> {"p(x)"},
                               LengthUnit -> "",
                               ForceUnit -> "",
                               PS -> fname]


ans := {s[0], u[0]} /. urule[x, L]

ans2 := Map[#[L/2]&, {v, m, s, u}] /. urule[x, L]
```

The two in-span support reactions are each (`inspan`)

$$R \;=\; -\frac{p_0\,L}{3}$$

where the negative sign indicates an upward force.

EXAMPLE 5

The first question for the beam loaded as shown in 11 is to determine the reaction forces at the supports. The program for this example is shown in Table 5. The shear forces at the two ends returned by `ans` give these forces

$$\begin{aligned} V(0) &= -\frac{p\,L}{6} \\ V(\frac{3\,L}{2}) &= -\frac{5\,p\,L}{12} \end{aligned}$$

According to the sign convention defined in Figure 2, a negative shear force on a left face is directed downward. Therefore, the reaction force at $x = 0$ is downward. A negative shear force on a right face

Table 5 *Mathematica Program for Example 5*

```
{v[x_], m[x_], s[x_], u[x_], inspan} = Beam[x, 3L/2, EI,  {"Pinned", "Pinned"},
                              LineLoad -> {p, L/2, p, 3L/2}, InSpan -> L/2]

draw[fname_:None] := DrawBeam[x, 60, 1, {"Pinned", "Pinned"},
                               InSpan -> 20,
                               ScaleTags -> {"L/2", "L"},
                               ScaleFar -> 9,
                               ScaleNear -> 3,
                               ScaleLevel -> 6,
                               ForceTagLoc -> 1.25,
                               LineLoad -> {1, 20, 1, 60},
                               LoadTags -> {"p"},
                               LengthUnit -> "",
                               ForceUnit -> "",
                               PS -> fname]


ans := {v[0], v[3L/2]} /. urule[x, L]

ndvar = p L^4/EI;

ndvarrule := UnitStep[c_] :> UnitStep[Cancel[c/L]]

S[z_] = SimplifyUnitStep[Cancel[s[x]/(ndvar/L)] /. x -> z L /. ndvarrule]

U[z_] = SimplifyUnitStep[Cancel[u[x]/ndvar] /. x -> z L /. ndvarrule]

mkplot := plots[U[z], S[z], z, 0, 3/2,
          TopTitle1 -> "Example 5",
          TopTitle2 -> "Example 5",
          XTitle -> "z",
          PS -> "exbeam5a.ps",
          Units -> None];
```

is directed upward. Therefore, the reaction force at $x = 3\,L/2$ is upward. The in-span reaction force at $x = L/2$ is (`inspan`)

$$R \;=\; -\frac{3\,p\,L}{4}$$

The negative sign here indicates an upward force according to the sign conventions defined earlier for external concentrated forces applied to the beam.

The second question in this example is to sketch slope and deflection curves. This can be done by defining a normalized deflection

$$\bar{w}(x) = \frac{w(x)}{\frac{p\,L^4}{EI}}$$

and a normalized slope

$$\bar{w}'(x) = \frac{w'(x)}{\frac{p\,L^3}{EI}}$$

The resulting variables `U`, for $\bar{w}(x)$, and `S`, for $\bar{w}'(x)$, are written as functions of $z = x/L$ and plotted from $z = 0$ to $z = 3\,L/2$. The results are shown in Figures 12 and 13. The analytical expression returned by `U[z]` is

$$U(z) = \frac{z}{144} - \frac{z^3}{36} + \left(-\frac{7}{384} + \frac{11\,z}{96} - \frac{z^2}{4} + \frac{5\,z^3}{24} - \frac{z^4}{24}\right)\,\langle z - \frac{1}{2}\rangle^0$$

and the expression `S[z]` is

$$S(z) = \frac{1}{144} - \frac{z^2}{12} + \left(\frac{11}{96} - \frac{z}{2} + \frac{5\,z^2}{8} - \frac{z^3}{6}\right)\,\langle z - \frac{1}{2}\rangle^0$$

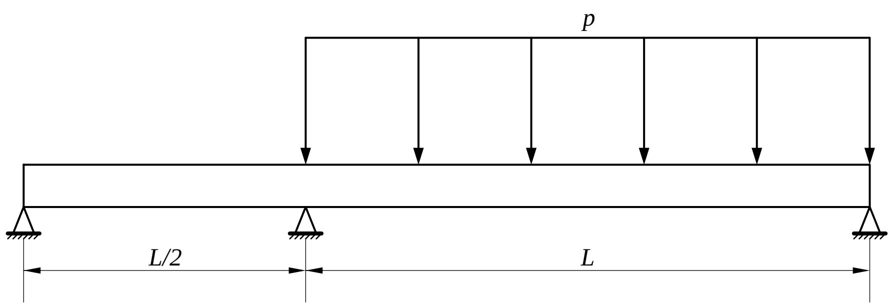

Figure 11 The Beam of Example 5

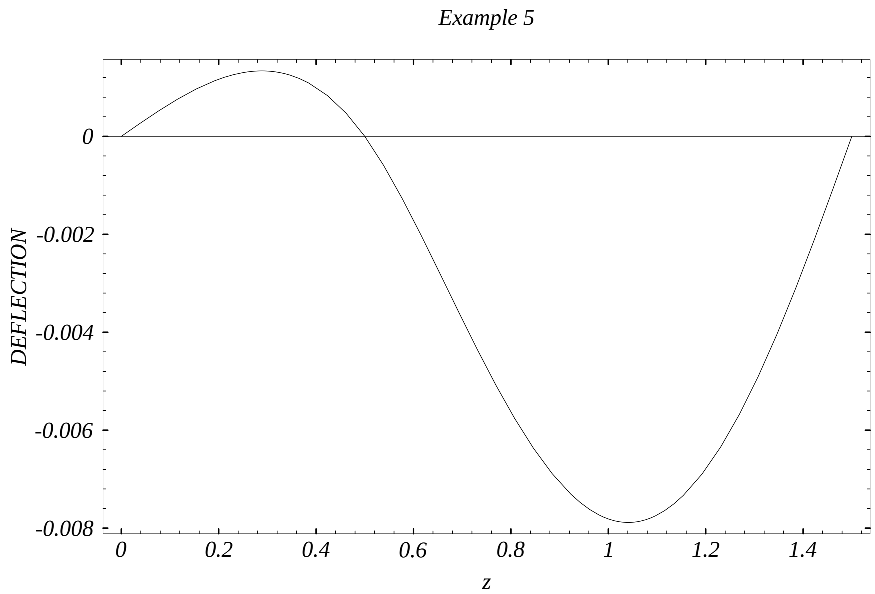

Figure 12 Deflection Curve for Example 5

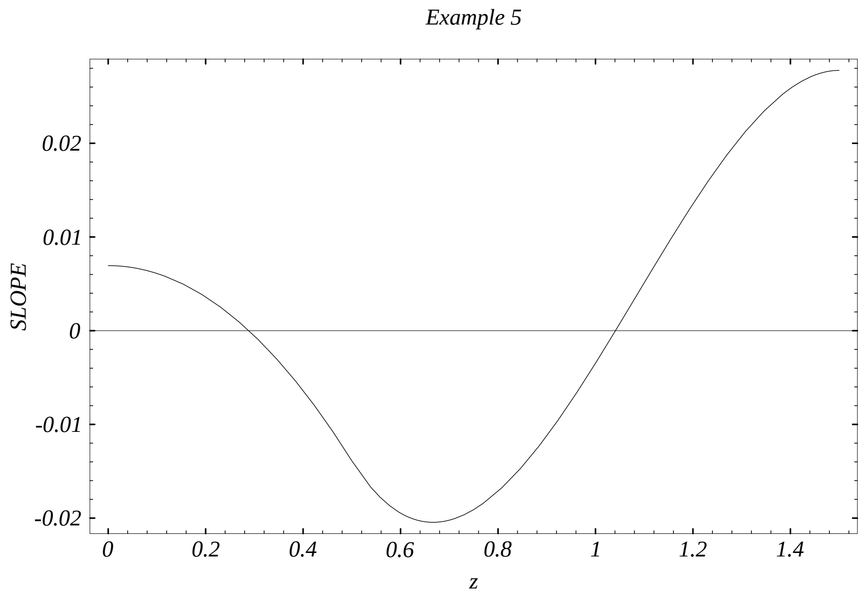

Figure 13 Slope Curve for Example 5

CONCLUSION

A *Mathematica* program intended for students taking a first course in engineering solid mechanics has been described. The five example problems solved above illustrate some possible uses of the program for homework assignments. Instructors can design these assignments to require *Mathematica* programming at various levels of complexity. For example, if the maximum normal stress in a beam of rectangular cross section is to be determined for a range of values of cross-sectional dimensions, the maximum bending moment can be calculated, as in Example 3, using the `BeamStatics` package, and students can write their own *Mathematica* programs to calculate and plot a normalized maximum stress versus the ratio of cross-sectional base to height.

The package was written for Mathematica Version 2.2. All bug reports should be e-mailed to the author at *lk3a@kelvin.seas.virginia.edu*.

REFERENCES

Wolfram, S., *Mathematica: A System for Doing Mathematics by Computer*, Second Edition, Addison-Wesley Publishing Company, Redwood City, California, 1991.

Maeder, R., *Programming in Mathematica*, Second Edition, Addison-Wesley Publishing Company, Redwood City, California, 1991.

Popov, E. P., *Engineering Mechanics of Solids*, Prentice-Hall, Englewood Cliffs, New Jersey, 1990.