Computations of Determinate and Low Degree Indeterminate Beams Using MATLAB

Grace Genszler

8.4.17

1 Methodology

The code can be divided into the following sections: parameter input, determination of degree of indeterminacy, reaction force calculations, construction of shear and moment diagrams, and displacement calculations at the user's specifications.

1.1 Parameter Input

Users are first asked to input the values for Young's Modulus, moment of inertia, and beam length. All calculations were coded with the intent of base units in either SI or Imperial. However, smaller or larger powers may be used at the user's discretion. Information about the supports is entered next. Moving left to right, each support's location and type data are saved to a matrix. This code accounts for roller, pinned, and fixed supports. The loading case data is also saved in a similar way. The location of the loading type, magnitude, directionality, and location of each load are saved to a matrix. Calculations involving applied moments, point loads, and discrete step distributed loads can all be performed.

1.2 Degree of Indeterminacy

Following parameter input, the support data is used to determine the degree of indeterminacy in a separate function, 'indet.' This function examines the dimensions of the supports matrix (since each row is a support), the number of reaction forces is saved in a corresponding column in each row, and performs

$$i = r - 3 * n$$

where

$$i = degreeindeterminate$$

$$r = numberofreactions$$

$$n = numberofsections$$

The numerical value of indeterminance is returned upon function completion.

1.3 Reaction Force Calculations

If the degree of indeterminacy is zero, then the supports and load data is used in the function 'statics' to calculate the support reactions which are saved to an array to be used later. Statics performs $\sum F_y = 0$ and $\sum M = 0$ calculations to solve for the reaction forces are returns them in an array. If the degree of indeterminacy is nonzero, then the supports and load data must be reformatted so the Force Method can be utilized. At this time, it should also be noted that axial deformations are not assumed and only cases of low indeterminacy, up to degree two, can be examined with this code.

First, the supports matrix is examined. If there is a fixed support, the non-fixed support is removed. Should there be two fixed supports, the right hand side support is removed. For pin/roller beams, all interior supports are removed and the exteriors are left. This new support matrix and the original load case matrix are then run through the statics function and the reactions are saved. Then, using the location of the removed support, the newly calculated reactions array, original loading matrix, and edited supports matrix, the displacement at the point of the removed support is calculated. In a following section, the displacement function will be explained more throughly.

The next step is editing the load array. For each redundancy that was removed in the first step, a variable is made and a new row is added to a new loading matrix. This new row treats the removed redundancy from earlier as either a qualitative point load or an applied moment that can be solved for at a later time. Similar to the first step, the new supports matrix and the new load matrix are run through the statics function and the reactions are saved. Again, the displacement function is called again and the location of the removed support, newly calculated reactions array, newly created loading matrix, and edited supports matrix are used.

Finally, compatibility between the two displacements can be enforced. The displacement from the first and second steps must be the same, so the reaction forces of the removed redundancies can be solved. Once this is done, these values can be added as rows to the original load matrix. In order to avoid double counting the redundant

supports as supports and loading, the edited supports matrix is renamed under the same variable as the original supports matrix. This along with the newly edited load matrix are then used to solve for the remaining reactions using the function statics.

1.4 Shear and Moment Diagrams

The function that graphs shear and moment diagrams, 'shearmoment,' requires the reactions array, loading matrix, supports matrix, beam length, and step size to iterate with. This last value is preset in the code. Two sets of loops, one set for shear and another for moment, are used to construct discontinuity function components in terms of a variable, x. These components, saved in a column of a matrix with its corresponding location value in another, are used in another set of loops that calculate the values for shear and moment at given distances by combining the necessary terms from the discontinuity function component matrix. These values are saved to arrays through an iterative process and are returned at the end of the function.

In the main script, the returned arrays are used to plot shear and moment diagrams using the subplot function. The moment diagram is drawn on compression side.

1.5 Displacement Calculations

Similar to 'shearmoment,' the function 'displacement' also begins by construction discontinuity function components in terms of x. The code becomes slightly more complex with the addition of two constants from double integration. For simply supported beams, it is only necessary to construct discontinuity function components for displacement. If there is a fixed support, then discontinuity function components for theta, rotation, must also be constructed. This allows for a system of two equations with two unknowns to be created and solved. In the case of simply supported, displacement is equal to zero at the locations of both of the supports. In the case of a cantilever beam, displacement and rotation is equal to zero at the fixed support. With these boundary conditions and the system of equations, the two constants can be solved for.

The second section of the function constructs the displacement discontinuity function for the user specified distance along the beam complete with terms from the discontinuity function component matrix and the two constants from double integration. This value is returned at the end of the function and displayed in the main function. The user then has the option to compute another displacement or terminate the main script.

2 Results

All examples use steel for a material for a Young's Modulus of 200GPa and a W1100x607 I beam.

2.1 Determinate Structure: Cantilever Beam

For this example, a 12m beam will be used.

The fixed support will be on the left side, so 0 is entered for the leftmost undefined support location and another 0 is entered for support type. The supports matrix looks like

$$supports = \begin{bmatrix} 0 & 0 \end{bmatrix}$$

A distributed load of 5 kilograms in the negative direction over the entire beam is the loading case, so the loading type is 1, the location is 0, the value is -5, and the dl length is 12. The loading matrix looks like

$$load = \begin{bmatrix} 0 & -5 & 12 & 1 \end{bmatrix}$$

The function 'indet' is then used and a value of 0 is saved since

$$r = 3$$

$$n = 1$$

The formula for calculating degree of indeterminacy can now be filled in:

$$i = r - 3 * n$$

$$0 = 3 - 3 * 1$$

Since the structure is determinate, reformatting of the supports and load matrices is not necessary. The function 'statics' can be called immediately. The reactions array looks like

$$rxns = \begin{bmatrix} 60 & 360 \end{bmatrix}$$

Sum of forces in the 'y' direction:

$$\sum F_y = 0 = A_y - w$$

$$\sum F_y = 0 = A_y - 5kN * 12m$$

$$A_y = 60kN.m$$

Sum of moments about distance 0m:

$$\sum M = 0 = M_a - \frac{wl}{2}$$

$$\sum M = 0 = M_a - 5kN * 12m * 6m$$

$$M_a = 360kN.m^2$$

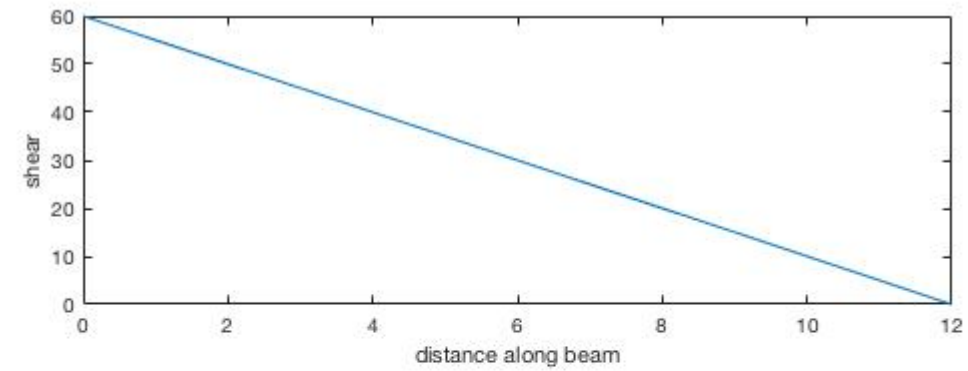
In 'shearmoment,' the discontinuity function component matrices are formed first. Column one is the location of the application or ending of a load. Column two is the discontinuity term.

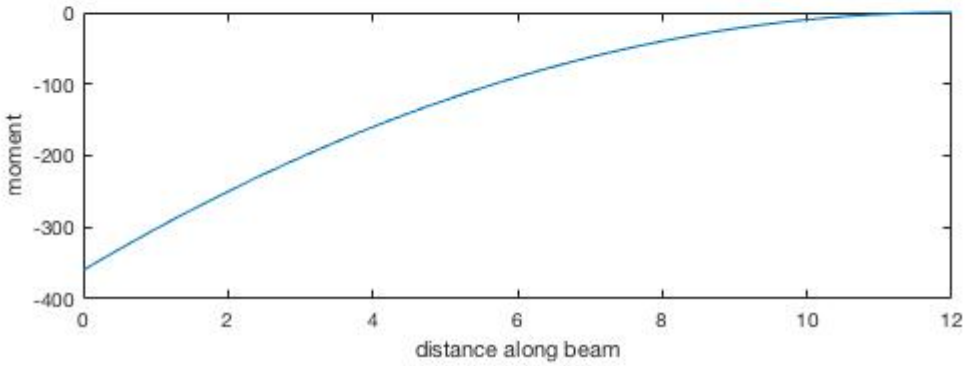
$$vcoord = \begin{bmatrix} 0 & 60 \\ 0 & -5 * x \\ 12 & 5 * x - 60 \end{bmatrix}$$

$$mcoord = \begin{bmatrix} 0 & 60 * x - 360 \\ 0 & \frac{-(5*x^2)}{2} \\ 12 & \frac{(5*(x-12)^2)}{2} \end{bmatrix}$$

In order to form the arrays v and m for plotting the shear and moment diagrams, the for loops cycle through positions along the beam. At each position in question, the distance is compared to the location given in the first column. If the distance along the beam is greater than or equal to the matrix location in question, the term is used to calculate the shear or moment. If it is not, then the if/else statement moves to the next row of the discontinuity function component matrix.

Once 'shearmoment' has completed, the shear and moment diagrams may be plotted.





In 'displacemnet' dcoord and tcoord are first formed. Column one is the location of the application or ending of a load. Column two is the discontinuity term.

$$dcoord = \begin{bmatrix} 0 & 10000 * x^3 - 180000 * x^2 \\ 0 & \frac{-(625*x^4)}{3} \\ 12 & \frac{(625*(x-12)^4)}{3} \end{bmatrix}$$

$$tcoord = \begin{bmatrix} 0 & 30000 * x^2 - 360000 * x \\ 0 & \frac{-(2500*x^3)}{3} \\ 12 & \frac{(2500*(x-12)^3)}{3} \end{bmatrix}$$

Since this is a cantilever beam, the boundary conditions at 0m are

$$\delta = 0$$

$$\theta = 0$$

The two equations in the system are

$$A = c_2$$

$$B = c_1$$

When solved, the constants from double integration are

$$c_1 = 0$$

$$c_2 = 0$$

For this example, the displacement at the end of the beam, at 12m, will be calculated. This value is given to be

$$d = \frac{-1942916495809225365}{18889465931478580854784}$$

$$d = -.1029m$$

This value matches what is given by the tabulated formula

$$d = \frac{qL^4}{8EI}$$

$$d = \frac{-5 * 12^4}{8 * 200 * 10^9 * .00063}$$

$$d = -.1029m$$

2.2 Determinate Structure: Simply Supported Beam

A 7m beam will be used here.

A pinned support will be placed at 2m and a roller support will be placed at 6m

$$supports = \begin{bmatrix} 1 & 2 \\ 2 & 6 \end{bmatrix}$$

Two point loads will be applied. Let there be a 10kN force in the negative direction at 0m and a 15kN force in the negative direction at 4m

$$load = \begin{bmatrix} 0 & -10 & 0 & 1 \\ 4 & -15 & 4 & 1 \end{bmatrix}$$

The function 'indet' is then used and a value of 0 is saved since

$$r = 3$$

$$n = 1$$

The formula for calculating degree of indeterminacy can now be filled in:

$$i = r - 3 * n$$

$$0 = 3 - 3 * 1$$

Since the structure is determinate, reformatting of the supports and load matrices is not necessary. The function 'statics' can be called immediately. The reactions array looks like

$$rxns = \begin{bmatrix} 22.5 & 2.5 \end{bmatrix}$$

Sum of moments about distance 2m:

$$\sum M = 0 = 10kN * 2m - 15kN * 2m + B_y * 4m$$

$$B_y = \frac{-10kN * 2m + 15kN * 2m}{4m}$$

$$B_y = \frac{10kN.m}{4m}$$

$$B_y = 2.5kN$$

Sum of forces in the 'y' direction:

$$\sum F_y = 0 = A_y + B_y - 10kN - 15kN$$

$$A_y = -B_y + 10kN + 15kN$$

$$A_y = -2.5kN + 10kN + 15kN$$

$$A_y = 22.5kN$$

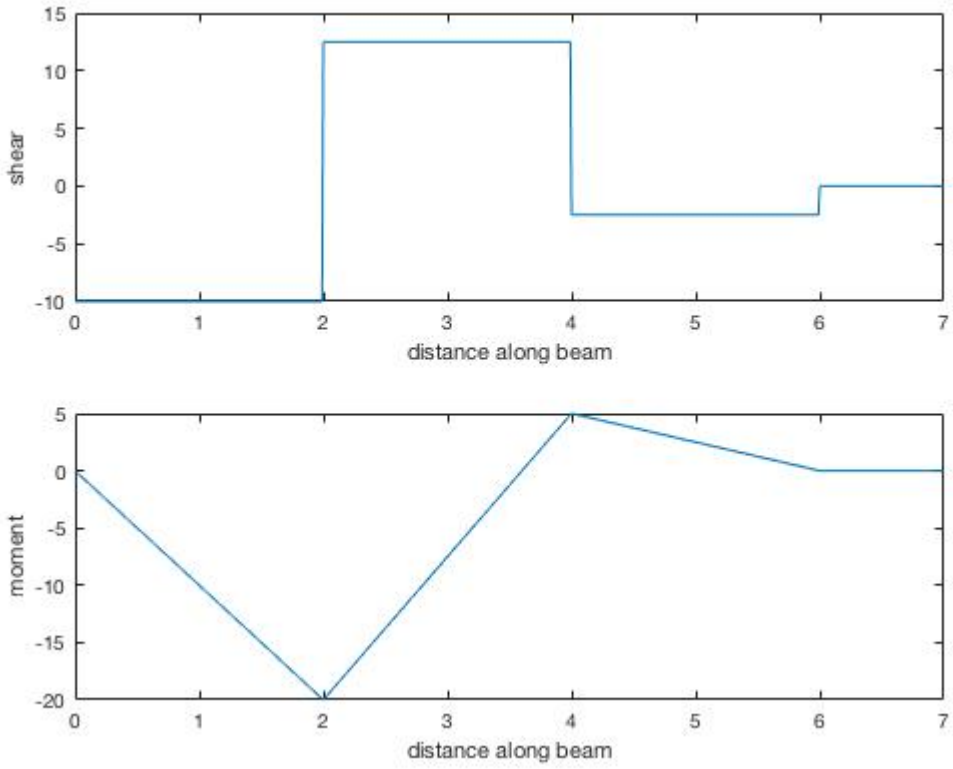
In 'shearmoment,' the discontinuity function component matrices are formed first. Column one is the location of the application or ending of a load. Column two is the discontinuity term.

$$vcoord = \begin{bmatrix} 2 & 22..5 \\ 6 & 2.5 \\ 0 & -10 \\ 4 & -15 \end{bmatrix}$$

$$mcoord = \begin{bmatrix} 2 & 22.5 * x - 45 \\ 6 & 2.5 * x - 15 \\ 0 & -10 * x \\ 4 & 60 - 15 * x \end{bmatrix}$$

In order to form the arrays v and m for plotting the shear and moment diagrams, the for loops cycle through positions along the beam. At each position in question, the distance is compared to the location given in the first column. If the distance along the beam is greater than or equal to the matrix location in question, the term is used to calculate the shear or moment. If it is not, then the if/else statement moves to the next row of the discontinuity function component matrix.

Once 'shearmoment' has completed, the shear and moment diagrams may be plotted.



In 'displacemnet' dcoord and tcoord are first formed. Column one is the location

of the application or ending of a load. Column two is the discontinuity term.

$$dcoord = \begin{bmatrix} 2 & 3750 * (x-2)^3 \\ 6 & \frac{1250*(x-2)^3}{3} \\ 0 & \frac{-5000*x^3}{3} \\ 4 & -2500 * (x-4)^3 \end{bmatrix}$$

This is a simply supported beam, so the boundary conditions are

$$\delta = 0$$

at x=2m and x=6m. The two equations in the system are

$$A = 2 * c_1 + c_2 - \frac{40000}{3}$$

$$B = 6 * c_1 + c_2 - 140000$$

When solved, the constants from double integration are

$$c_1 = \frac{95000}{3}$$

$$c_2 = -50000$$

For this example, the displacement at the end of the beam, at 3.5m, will be calculated. This value is given to be

$$d = \frac{-326817743893835748125}{1208925819614629174706176}$$

$$d = -1.612 * 10^{-5} m$$

2.3 Indeterminate Structure: Cantilever and Pinned Beam

The beam will be 4m long with fixed supports on the left and pinned at 3m.

$$supports = \begin{bmatrix} 0 & 0 \\ 1 & 3 \end{bmatrix}$$

A 6kN.m counter-clockwise moment will be applied at 2m. One distributive load of -2kN/m will happen from 2-3m. Another distributive load of -8kN/m will happen from 3-4m.

$$load = \begin{bmatrix} 2 & 6 & 2 & 0 \\ 2 & -2 & 3 & 1 \\ 3 & -8 & 4 & 1 \end{bmatrix}$$

The function 'indet' returns 2 for a degree of indeterminacy

$$r = 5$$

$$n = 1$$

The formula for calculating degree of indeterminacy can now be filled in:

$$i = r - 3 * n$$

$$2 = 5 - 3 * 1$$

Reformatting of the supports and load matrices is necessary. The pinned support's entry, the second row, is removed from the support matrix and this new matrix is saved as

$$lsupports = \begin{bmatrix} 0 & 0 \end{bmatrix}$$

Statics is then performed using lsupports and load, yielding

$$lrxns = \begin{bmatrix} 10 & 39 \end{bmatrix}$$

Then, the displacement function is used to solve for the displacement on the cantilever beam due to the original loading case at x=3m. This value comes out to be

$$dload = \frac{-961263932957465512375}{906694364710971881029632}$$

$$dload = -.0011$$

Since axial rigidity has been assumed, only one reaction needs to be solved for. The redundant load matrix is

$$rload = \begin{bmatrix} 3 & B & 3 & 1 \end{bmatrix}$$

Once again, 'statics' is used. From lsupports and rload, rrxns is saved as

$$rrxns = \begin{bmatrix} -B & -3 * B \end{bmatrix}$$

'Displacement' is now used to find the displacement on the cantilever beam due to the application of the redundant loading case at x=3m. This value is

$$dred = \frac{5396990266136737125 * B}{75557863725914323419136}$$

$$dred = 7.143 * 10^{-5} * B$$

Compatibility between dload and dred can now be enforced.

$$dload = dred$$

$$.0011 = 7.143 * 10^{-5} * B$$
$$B = \frac{1603}{108}$$
$$B = 14.843kN$$

Finally, 'statics' can be used to solve the remaining reactions.

$$rxns = \begin{bmatrix} \frac{-523}{108} & \frac{-199}{36} \end{bmatrix}$$

$$rxns = \begin{bmatrix} -4.843 & -5.528 \end{bmatrix}$$

Sum of forces in the 'y' direction:

$$\sum F_y = 0 = A_y + B_y - 2kN - 8kN$$

$$A_y = -B_y + 2kN + 8kN$$
$$A_y = -14.843kN + 2kN + 8kN$$
$$A_y = 4.843kN$$

Sum of moments about distance 2m:

$$\sum M = 0 = B_y * 3 - w_1 * l_1 - w_2 * l_2 + M_a$$

$$M_a = 2kN * 2.5m + 8kN * 3.5m - 14.843kN * 3m$$
$$M_a = -5.528kN.m$$

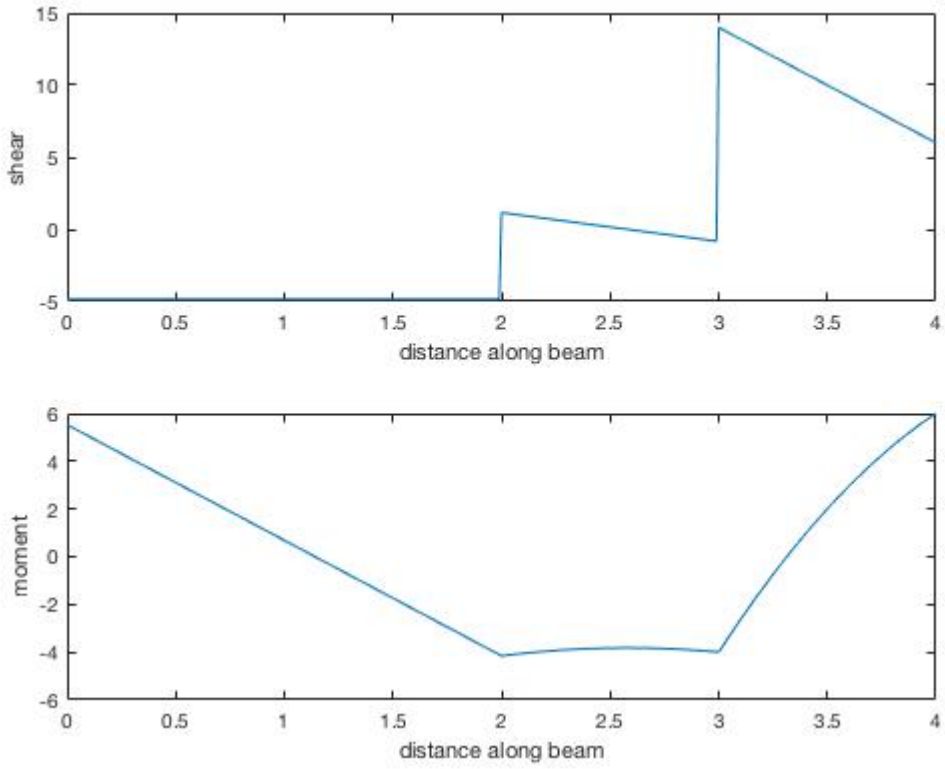
In 'shearmoment,' the discontinuity function component matrices are formed first. Column one is the location of the application or ending of a load. Column two is the discontinuity term.

$$vcoord = \begin{bmatrix} 0 & \frac{-523}{108} \\ 2 & 6 \\ 2 & 4 - 2 * x \\ 3 & 2 * x - 6 \\ 3 & 24 - 8 * x \\ 4 & 8 * x - 32 \\ 3 & \frac{1603}{108} \end{bmatrix}$$

$$mcoord = \begin{bmatrix} 0 & \frac{199}{36} - \frac{523*x}{108} \\ 2 & 6*x-12 \\ 2 & -(x-2)^2 \\ 3 & (x-3)^2 \\ 3 & -4*(x-3)^2 \\ 4 & 4*(x-4)^2 \\ 3 & \frac{1603}{108} - \frac{1603}{36} \end{bmatrix}$$

In order to form the arrays v and m for plotting the shear and moment diagrams, the for loops cycle through positions along the beam. At each position in question, the distance is compared to the location given in the first column. If the distance along the beam is greater than or equal to the matrix location in question, the term is used to calculate the shear or moment. If it is not, then the if/else statement moves to the next row of the discontinuity function component matrix.

Once 'shearmoment' has completed, the shear and moment diagrams may be plotted.



In 'displacemnet' dcoord and tcoord are first formed. Column one is the location of the application or ending of a load. Column two is the discontinuity term.

$$dcoord = \begin{bmatrix} 0 & \frac{65375*x^3}{81} + \frac{24875*x^2}{9} \\ 2 & -3000*(x-2)^2 \\ 2 & \frac{-(250*(x-2)^4}{3} \\ 3 & \frac{250*(x-3)^4}{3} \\ 3 & \frac{-1000*(x-3)^4}{3} \\ 4 & \frac{1000*(x-4)^4}{3} \end{bmatrix}$$

$$tcoord = \begin{bmatrix} 0 & \frac{49750*x}{9} - \frac{65375*x^2}{29} \\ 2 & -12000 - 6000*x \\ 2 & \frac{-(1000*(x-2)^3}{3} \\ 3 & \frac{1000*(x-3)^3}{3} \\ 3 & \frac{-4000*(x-3)^3}{3} \\ 4 & \frac{4000*(x-4)^3}{3} \end{bmatrix}$$

The boundary conditions for this case at x=0m

$$\delta = 0$$

$$\theta = 0$$

The two equations in the system are

$$A = c_2$$

$$B = c_1$$

When solved, the constants from double integration are

$$c_1 = 0$$

$$c_2 = 0$$

For this example, the displacement at the end of the beam, at 2m, will be calculated. This value is given to be

$$d = \frac{446750860919096573125}{1224037392359812039390003 2}$$

$$d = 3.65 * 10^{-5} m$$

3 Code

Table of Contents


```
% Grace Genszler
% SURE 2017
% Beam Deflection Computations

clear all
```

Parameter Input

```
% parameters
E=input('Input modulous of elasticity \n');
I=input('Input moment of inertia \n');
l=input('Input length of beam \n');
lstep=.01;
L=0:lstep:l;

% supports
n=0;
supports=[];

while n==0
    loc=input('Input location of leftmost undefined support \n');
    stype=input('Input support type \n 0 for fixed \n 1 for pinned \n
 2 for roller \n');
    supports=[supports;
              stype    loc];
    n=input('Enter another support? \n 0 for YES \n 1 for NO \n');
end

% loading
m=0;
load=[];

while m==0
    ltype=input('Input applied loading type \n 0 for moment \n 1 for
 force \n');
    a=input('Input location of leftmost undefined loading \n');
    p=input('Input value of leftmost undefined loading \n + for
 upwards and CCW \n - for downwards and CW \n');
    dl=input('Input dl length \n enter zero (0) for applied moment or
 point load \n');
    load=[load;
```

```
          a    p    a+dl      ltype];
    m=input('Enter another loading? \n 0 for YES \n 1 for NO \n');
end

Error using input
Cannot call INPUT from EVALC.

Error in beam_calcs (line 9)
E=input('Input modulous of elasticity \n');
```

Static Indeterminacy

```
deg_indet=indet(supports);
```

Solve Reaction Forces

```
if deg_indet==0
    rxns=statics(supports,load);
    %if determinate, reactions are solved
else
    [m1,n1]=size(supports);
    if supports(1,1)==0
        lsupports=[supports(1,1)    supports(1,2)];
    elseif supports(2,1)==0
        lsupports=[supports(2,1)    supports(2,2)];
    else
        lsupports=[supports(1,1)    supports(1,2);
                   supports(m1,1)   supports(m1,2)];
    end
    % eliminate redundancies, leaving only the fixed support or the
 two end
    % supports for a simply supported beam. First case is for
 cantilever
    % beams with the wall on the left. Second case is for cantilever
 beams
    % with the wall on the right. Third case is for simply supported
 beams.

    lrxns=statics(lsupports,load);
    % reactions for loaded case

    d_load=[];

     if supports(1,1)==0
         d=displacement(supports(2,2),lrxns,load,lsupports,l,E,I);
         d_load=[d_load     d];
     elseif supports(2,1)==0
         d=displacement(supports(1,2),lrxns,load,lsupports,l,E,I);
         d_load=[d_load     d];
     else
         for i=2:m1-1
            d=displacement(supports(i,2),lrxns,load,lsupports,l,E,I);
            d_load=[d_load  d];
```

```
        end
    end
   % compute deflections at the locations of the removed supports.
First
   % case is for cantilever beams with the wall on the left. Second
case
   % is for cantilever beams with the wall on the right. Third case
is for
   % simply supported beams.

   if size(d_load,2)==1
       syms B
       rload=[supports(2,2)    B       supports(2,2)       1];
   else
       syms B C
       if supports(1,1)==0 && supports(2,1)==0
           rload=[supports(2,2)    B       supports(2,2)       1;
                  supports(2,2)    C       supports(2,2)       0];
       else
           rload=[supports(2,2)    B       supports(2,2)       1;
                  supports(3,2)    C       supports(3,2)       1];
       end
       % First case is for a beam between two fixed supports. Second
case
       % is for one fixed support and one pin or roller support.
   end
   % create variables of the reaction forces generated by the removed
   % supports. First case is for one redundancy. Second case is for
two
   % redundancies.

   rrxns=statics(lsupports,rload);
   %solves the reactions for the loaded case

   d_red=[];

   if supports(1,1)==0 && supports(2,1)~=0
       d=displacement(supports(2,2),rrxns,rload,lsupports,l,E,I);
       d_red=[d_red    d];
   elseif supports(2,1)==0 && supports(1,1)~=0
       d=displacement(supports(1,2),rrxns,rload,lsupports,l,E,I);
       d_red=[d_red    d];
   else
       for i=2:m1-1
           d=displacement(supports(i,2),rrxns,rload,lsupports,l,E,I);
           d_red=[d_red    d];
       end
   end
   % compute deflections at the locations of the redundants. First
case is
   % for a cantilever beam with the fixed support on the left. Second
case
   % is for a cantilever beam with the fixed support on the right.
Third
```

```
    % case is for simply supported.

    if size(d_load,2)==1
        [By]=solve(d_red==-d_load, B);
        % solves the reaction force of the removed support
        if supports(2,1)==0
            load=[load;
                  supports(1,2)     By      supports(1,2)   1];
        else
            load=[load;
                  supports(2,2)     By      supports(2,2)   1];
        end
        % first case is for cantilever beams with the fixed support on
 the
        % right. Second case is for everything else.
    else
        [By, Cy]=solve(d_red==d_load, [B, C]);
        % solves the reaction forces of the removed support(s)
        if supports(1,1)==0 && supports(2,1)==0
            load=[load;
                  suports(2,2)      By      supports(2,2)   1;
                  supports(2,2)     Cy      supports(3,2)   0];
        else
            load=[load;
                  supports(2,2)     By      supports(2,2)   1;
                  supports(3,2)     Cy      supports(3,2)   1];
        end
        % first case is for a beam fixed at both ends. Second case is
 for
        % everything else
    end
    %reformats the array 'load,' adding a new entry that treats the
    %newly solved reaction force of the removed redundant as a point
    %load

    supports=lsupports;
    %rename the supports array since the redundant support has been
    %accounted for in the load array

    rxns=statics(supports,load);
    %solve for the remaining reactions
end
```

Shear and Moment

```
[v,m]=shearmoment(rxns,load,supports,l,lstep);

figure(1)
subplot(2,1,1)
plot(L, v)
xlabel('distance along beam')
ylabel('shear')
subplot(2,1,2)
```

```
plot(L, m)
xlabel('distance along beam')
ylabel('moment')
```

Displacement

```
r=0;

while r==0
    y=input('Input distance to calculate displacement \n');
    d=displacement(y,rxns,load,supports,l,E,I)
    r=input('Calculate another displacement? \n 0 for YES \n 1 for NO
 \n');
end
```

Published with MATLAB® R2016a

```
% function solves degree of static indeterminacy
function degindet=indet(y)
    rxns=0;
    [m,n]=size(y);
    for x=1:m
        rxns=rxns+3-y(x,1);
    end
    % for every row in the matrix y, the first column's entry is
 examined
    % and used to calculate the number of reactions
    degindet=rxns-3;
    % degree indeterminate is equal to the number of reactions minus
 thrice
    % the number of sections. However, the number of sections will
 always
    % be one for the cases this code can handle.
end
```

```
Not enough input arguments.

Error in indet (line 4)
    [m,n]=size(y);
```

Published with MATLAB® R2016a

```
% function solves the reaction forces for a simply supported beam
function rxns=statics(supports,load)
    [m,n]=size(load);
    pfy=0;
    dfy=0;
    pm=0;
    dm=0;

    for i=1:m
        if load(i,1)==load(i,3) && load(i,4)==1
            pfy=pfy-load(i,2);
            pm=pm-load(i,2)*(load(i,1)-supports(1,2));
            % calculates the amount of force due to a distributed load
 and
            % the moment caused by it
        elseif load(i,1)==load(i,3) && load(i,4)==0
            pfy=pfy;
            pm=pm+load(i,2);
            % accounts for an applied moment in the moment sum
        else
            dfy=dfy-load(i,2)*(load(i,3)-load(i,1));
            dm=dm-load(i,2)*(load(i,3)-
load(i,1))*(1/2*(load(i,1)+load(i,3))-supports(1,2));
            % calculates the amount of force due to a point load and
 the
            % moment caused by it
        end
    end

    fy=pfy+dfy;
    mm=pm+dm;
    % combines all moments and loads

    if supports(1,1)==0 || supports(2,1)==0
        rxns=[fy, mm];
        % assigns the sums as force in the y direction and reactional
        % moment
    else
        By=(mm)/(supports(2,2)-supports(1,2));
        Ay=(fy-By);
        rxns=[Ay, By];
        % performs sum of moments about the left most support and uses
 the
        % answer to solve sum of forces in the y direction
end
```

```
Not enough input arguments.

Error in statics (line 3)
    [m,n]=size(load);
```

Published with MATLAB® R2016a

```
% function graphs shear and moment diagrams using discontinuity
 functions
function [v,m]=shearmoment(rxns,load,supports,l,lstep)
    [m1,n1]=size(supports);
    [m2,n2]=size(load);

    syms x;
    vcoord=[];
    mcoord=[];

    % supports: generates correct discontinuity term from supports
    for i=1:m1
        if supports(i,1)==0
            vcoord=[vcoord;
                    supports(i,2)    rxns(1)];
            mcoord=[mcoord;
                    supports(i,2)    rxns(1)*(x-supports(i,2))-
rxns(2)];
        else
            vcoord=[vcoord;
                    supports(i,2)    rxns(i)];
            mcoord=[mcoord;
                    supports(i,2)    rxns(i)*(x-supports(i,2))];
        end
    end

    % loading: generates correct discontinuity term from loading
    for i=1:m2
        if load(i,1)==load(i,3)
            vcoord=[vcoord;
                    load(i,1)    load(i,2)+x*0];
            mcoord=[mcoord;
                    load(i,1)    load(i,2)*(x-load(i,1))];
        else
            vcoord=[vcoord;
                    load(i,1)    load(i,2)*(x-load(i,1));
                    load(i,3)    -load(i,2)*(x-load(i,3))];
            mcoord=[mcoord;
                    load(i,1)    1/2*load(i,2)*(x-load(i,1))^2;
                    load(i,3)    -1/2*load(i,2)*(x-load(i,3))^2];
        end
    end

    [m3,n3]=size(vcoord);
    [m4,n4]=size(mcoord);

    vc(x)=vcoord;
    mc(x)=mcoord;

    v=[];
    m=[];
```

```
    % computes shear and moment values and stores them in arrays
    lcount=0;
    for y=0:l/lstep
        vnew=0;
        for i=1:m3
            if lcount>=vcoord(i,1)
                vmag=vc(lcount);
                vnew=vnew+vmag(i,2);
            else
                continue
            end
        end
        v=[v vnew];
        lcount=lcount+lstep;
    end

    lcount=0;
    for y=0:l/lstep
        mnew=0;
        for i=1:m4
            if lcount>=mcoord(i,1)
                mmag=mc(lcount);
                mnew=mnew+mmag(i,2);
            else
                continue
            end
        end
        m=[m mnew];
        lcount=lcount+lstep;
    end
end
```

```
Not enough input arguments.

Error in shearmoment (line 3)
    [m1,n1]=size(supports);
```

Published with MATLAB® R2016a

```
% solves the displacement at a given distance along the beam
function d=displacement(y,rxns,load,supports,l,E,I)

[m1,n1]=size(supports);
[m2,n2]=size(load);

syms x;
dcoord=[];
tcoord=[];

for i=1:m1
    if supports(i,1)==0
        dcoord=[dcoord;
                supports(i,2)    rxns(1)/6*x^3-rxns(2)/2*x^2];
        tcoord=[tcoord;
                supports(i,2)    rxns(1)/2*x^2-rxns(2)*x];
    else
        dcoord=[dcoord;
                supports(i,2)    rxns(i)/6*(x-supports(i,2))^3];
    end
end


for i=1:m2
    if load(i,1)==load(i,3) && load(i,4)==1
        dcoord=[dcoord;
                load(i,1)    load(i,2)/6*(x-load(i,1))^3];
    elseif load(i,1)==load(i,3) && load(i,4)==0
        dcoord=[dcoord;
                load(i,1)    -load(i,2)/2*(x-load(i,1))^2];
    else
        dcoord=[dcoord;
                load(i,1)    load(i,2)/24*(x-load(i,1))^4;
                load(i,3)    -load(i,2)/24*(x-load(i,3))^4];
    end
end

if size(tcoord,1)==0
    ;
else
    for i=1:m2
        if load(i,1)==load(i,3) && load(i,4)==1
            tcoord=[tcoord;
                    load(i,1)    load(i,2)/2*(x-load(i,1))^2];
        elseif load(i,1)==load(i,3) && load(i,4)==0
            tcoord=[tcoord;
                    load(i,1)    -load(i,2)*(x-load(i,1))];
        else
            tcoord=[tcoord;
                    load(i,1)    load(i,2)/6*(x-load(i,1))^3;
                    load(i,3)    -load(i,2)/6*(x-load(i,3))^3];
        end
```

```
    end
    tc(x)=tcoord;
end
% creates discontinuity function component matrices. Each application
 or
% ending of a load is assigned a distance and discontinuity term per
 row.
% This is all done in terms of the variable x. The first for loop is
 for
% supports. The second loop is for loading on a simply supported beam.
 The
% third loop is for loading on a cantilever beam.

[m3,n3]=size(dcoord);
[m4,n4]=size(tcoord);
dc(x)=dcoord;
d=0;
t=0;
A=0;
B=0;

syms c1 c2

Amag=dc(supports(1,2));

for i=1:m3
    if supports(1,2)>=dcoord(i,1)
        A=A+Amag(i,2);
    else
        continue
    end
end
A=A+c1*supports(1,2)+c2;

if supports(1,1)==0
    tmag=tc(supports(1,2));
    for i=1:m4
        if supports(1,2)>=tcoord(i,1)
            B=B+tmag(i,2);
        else
            continue
        end
    end
    B=B+c1;
else
    Bmag=dc(supports(2,2));
    for i=1:m3
        if supports(2,2)>=dcoord(i,1)
            B=B+Bmag(i,2);
        else
            continue
        end
    end
    B=B+c1*supports(2,2)+c2;
```

```
end
% The for loops above create the necessary discontinuity function
 system in
% order to solve for the two constants from double integration.

eqs=[A==0, B==0];
vars=[c1 c2];

[sol1, sol2]=solve(eqs, vars);
% For a simply supported beam, the displacements at the locations of
 the
% supports will be zero. For a cantilever beam, displacement and
 rotation
% at the fixed support will be zero.

for i=1:m3
    if y>=dcoord(i,1)
        dmag=dc(y);
        d=d+dmag(i,2);
    else
        continue
    end
end
% With the constants now solved for, the displacement at the location
 in
% question can now be calculated.
d=10^3/(E*I)*(d+sol1*y+sol2);
end
```

```
Not enough input arguments.

Error in displacement (line 4)
[m1,n1]=size(supports);
```

Published with MATLAB® R2016a