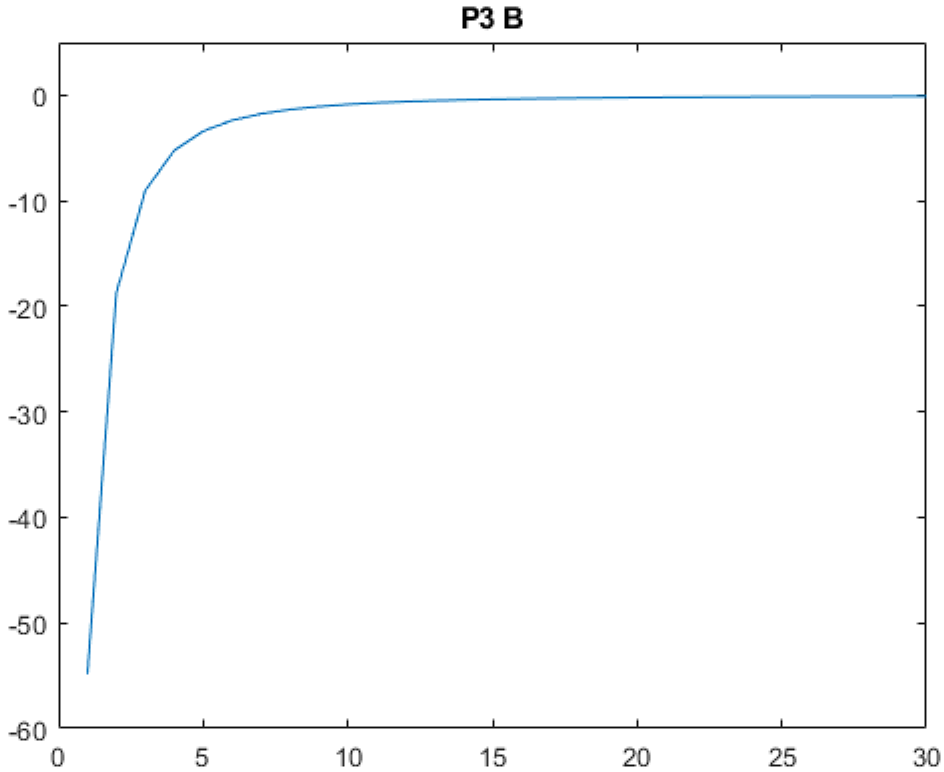
```matlab
%This example shows how local effects (e.g., point loads and boundary
%conditions distribute to more uniformly distributed stresses. This
%phenomenon applies in many different scenarios, but this example is a
%point load.

%This example is an upper 1/2 domain model that exploits symmetry.

%This example is clamped on the left, free elsewhere

%This code is clumsy. Tying BCs to n-info is not really great. it
 would be
%to mesh independent of BCs, then have a separate operation to apply
 BCs.

%-----------------------------------------------------------------------------
%Stage 1 Preprocessing/Input Section
%Commercial software can create the n_info, e_info, l_info from a CAD
 file
%instead. The e_info & n_info creation is often called "Meshing"
clc
clear all
close all
%Material Properties
E=29000;
nu=0.3;
PL=0.1;     %Point load at on top right of cantilever

%Define P-stress or P-strain
type=0;   %0 = Plane Stress,  1 = Plane Strain

h=1;       %depth
B=[1:1:30];       %Full Span
pE=[];
b_over_h=[];

for j = 1:length(B);
b=B(j);
numh=10;    %Number of elements through depth. Will will automaticall
 find
            %number through span to achieve near unit aspect ratio

%****************************************************************************
%CONSTRUCT N_INFO
%Find dx and dy which are the horizontal and vertical lengths of the
 triangles
dy=h/numh;
%Find dx that creates near unit aspect raio triangles
numb=floor(b/dy);
dx=b/numb;

%Construct left wall, which is fixed (the stuff below here is a bit
 ugly)
```

```
n_info=zeros(numh+1,1);           %x-coord of left wall
y_coords=linspace(0,h,numh+1)';   %y-coord column of a vertical strip
 of nodes
n_info=[n_info y_coords];     %append the y-coords
n_info=[n_info ones(numh+1,1) ones(numh+1,1)];   %append fixed in x
 and y columns
n_info=[n_info zeros(numh+1,2)];  %append the support disps, which are
 all zero

%Construct the rest of the vertical strips of nodes, all free
for i=1:numb
  xpos=dx*i;
  n_info_temp=[xpos*ones(numh+1,1) y_coords zeros(numh+1,4)];
  n_info=[n_info;n_info_temp];  %add new row to overall n_info matrix
end

%***************************************************************************
%CONSTRUCT E_INFO
e_info=[0 0 0];  %Initialize, will delete this row at the end
%Walk upward one strip at a time
for i=1:numb;
  bl=(i-1)*(numh+1)+1; %Bottom left node of strip
  br=i*(numh+1)+1;
  %Make lower triangles first
  e_info_temp=[linspace(bl,bl+numh-1,numh)' linspace(br,br
+numh-1,numh)' linspace(br,br+numh-1,numh)'+1];
  e_info=[e_info;e_info_temp];
  %Then upper triangle
  e_info_temp=[linspace(bl,bl+numh-1,numh)' linspace(bl,bl
+numh-1,numh)'+1 linspace(br,br+numh-1,numh)'+1];
  e_info=[e_info;e_info_temp];
end
e_info(1,:)=[];


%***************************************************************************
%CONSTRUCT L_INFO
%Only one load here. On the top-right most node.
Load_Node=size(n_info,1);  %This is the top-right node
l_info=[Load_Node 0 -PL];  %Apply downward point load at free end top
%End Input Section
%+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++



%Stage 2 Processing/Analysis/Solving, get nodal  & Reactions
solver;

%Stage 3 Postprocessing, plot def, stresses, etc
% plotter
% pause

%Additionally plot the half span horizontal displacement to check if
 PSRP?
```

```
%Select vertical strip of nodes at or near midspan
bot_node=floor(numb/2)*(numh+1)+1;
top_node=bot_node+numh;
ux=u((bot_node-1)*2+1:2:(top_node-1)*2+1);
xcoord=ux;
uy=u((bot_node-1)*2+2:2:(top_node-1)*2+2);
ycoord=y_coords+uy;
hold off
close all

%Difference between v' & theta (only work with numh=Even #) at midspan
%Find a cruciform shape of nodes for finite difference
mid_node=bot_node+numh/2;
u_node=mid_node+1;
d_node=mid_node-1;
r_node=mid_node+numh+1;
l_node=mid_node-numh-1;
theta=-(u(2*(u_node-1)+1)-u(2*(d_node-1)+1))/(2*dy);
vp=(u(2*(r_node-1)+2)-u(2*(l_node-1)+2))/(2*dx);
b_over_h(end+1)=[b/h];
pE(end+1)=(theta-vp)/vp*100;
end

figure
plot(b_over_h,pE)
xlim([0 30])
ylim([-60 5])
title('P3 B')
%+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++
```



Published with MATLAB® R2020b