MAE 598
Design Optimization
Project 3

Submitted by:

Shubham M. Chaudhary

ASU ID – 1219648911

Introduction

In this project, we do topology optimization. The structure is a fixed cantilever beam with one free end. A force of F = -10 N along y-axis is applied at the node at the free end. It is given in fig 1.

Fig 1: Cantilever beam

The pseudo code for compliance minimization

The pseudo code for compliance minimization is the following:

- Problem setup (see details below)
- Algorithm setup: $\epsilon = 0.001$ (or any small positive number), $k = 0$ (counter for the iteration), $\Delta x = 1000$ (or any number larger than ϵ)
- While $||\Delta x|| \leq \epsilon$, Do:
  - Update the stiffness matrix $\mathbf{K}$ and the displacement (state) $\mathbf{u}$ (finite element analysis)
  - Calculate element-wise compliance $\mathbf{u}_i^T \bar{\mathbf{K}}_e \mathbf{u}_i$
  - Calculate partial derivatives

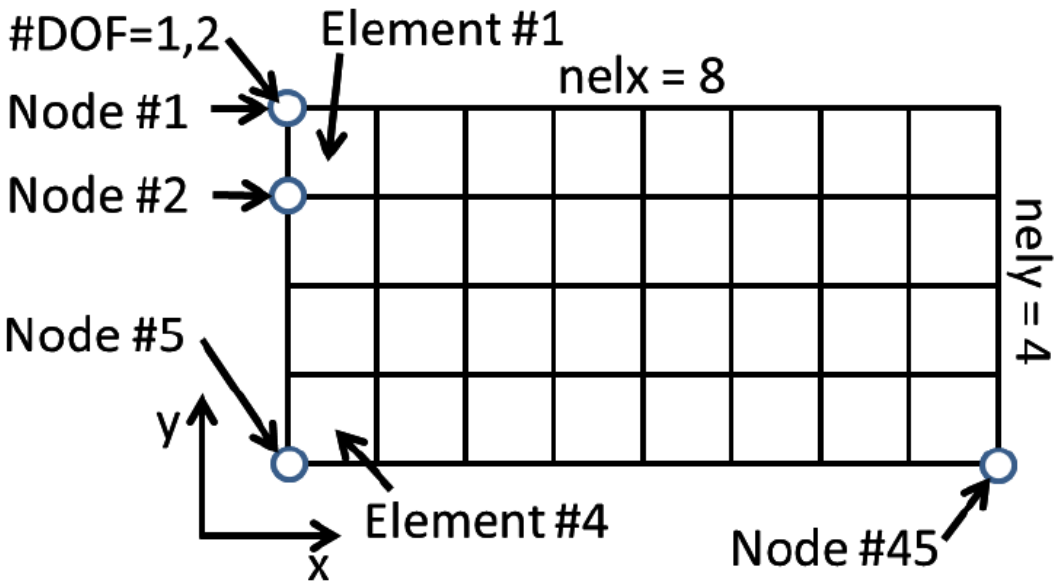
$$\frac{df}{dx_i} = -3\Delta E x_i^2 \mathbf{u}_i^T \bar{\mathbf{K}}_e \mathbf{u}_i$$

  - The gradient with respect to g is a constant vector $[1, ..., 1]^T$
  - Apply filter to $\frac{df}{d\mathbf{x}}$ (See discussion later)
  - Update $\mathbf{x}$:

$$\mathbf{x}'_{k+1} = \mathbf{x}_k - \alpha_k \left(\frac{df}{d\mathbf{x}} + \mu \mathbf{1} \right),$$

    where $\mu \geq 0$ is determined in the next step. To ensure that the gradient descent is successful, we will either set α_k to a small value, or truncate $\Delta x = -(\frac{df}{d\mathbf{x}} + \mu \mathbf{1})$ within a range (conceptually similar to the idea of trust region).
  - Move $\mathbf{x}'_{k+1}$ back to the feasible domain: If $\mathbf{1}^T \mathbf{x}_k < v$ and $-\mathbf{1}^T \frac{df}{d\mathbf{x}} < 0$, then $\mathbf{x}'_{k+1}$ satisfies g. with $\mu = 0$. If $\mathbf{x}'_{k+1}$ does not satisfy g, we will increase μ using bisection, i.e., search in $[0, \mu_{max}]$ where μ_{max} is a large positive number. Also, we will truncate $\mathbf{x}'_{k+1}$ between 0 and 1.
  - Update $||\Delta x||$, $k = k + 1$

The numbering of nodes is given in the figure below.



We take the number of elements in y direction, nely = 50

the number of elements in y direction, nelx = 200

penalty parameters of Young's Modulus, penal = 3

Filter radius, rmin = 3

We consider the material structural steel.

Young's Modulus, $E_o = 210$

Poisson's ratio, $\nu = 0.3$

All the node of left side are fixed in both x and y-axis.

Results

We run the code

top88(200,50,0.7,3.0,3.0,2);

We get the optimal design in the figure below.

The new design has a 30% less volume than the original beam and can sustain the applied force without failure.

MATLAB Code

The MATLAB code is slightly modified from the code written by E. Andreassen, A. Clausen, M. Schevenels[1].

```
%%%% Modified by Max Yi Ren (ASU) %%%%%%%%%%%%%%%%%%%%%%%
%%%% AN 88 LINE TOPOLOGY OPTIMIZATION CODE Nov, 2010 %%%%
function top88(nelx,nely,volfrac,penal,rmin,ft)
%% MATERIAL PROPERTIES
E0 = 210;
Emin = 1e-9;
nu = 0.3;
%% PREPARE FINITE ELEMENT ANALYSIS
A11 = [12 3 -6 -3; 3 12 3 0; -6 3 12 -3; -3 0 -3 12];
A12 = [-6 -3 0 3; -3 -6 -3 -6; 0 -3 -6 3; 3 -6 3 -6];
B11 = [-4 3 -2 9; 3 -4 -9 4; -2 -9 -4 -3; 9 4 -3 -4];
B12 = [ 2 -3 4 -9; -3 2 9 -2; 4 9 2 3; -9 -2 3 2];
KE = 1/(1-nu^2)/24*([A11 A12;A12' A11]+nu*[B11 B12;B12' B11]);
nodenrs = reshape(1:(1+nelx)*(1+nely),1+nely,1+nelx);
edofVec = reshape(2*nodenrs(1:end-1,1:end-1)+1,nelx*nely,1);
edofMat = repmat(edofVec,1,8)+repmat([0 1 2*nely+[2 3 0 1] -2 -
1],nelx*nely,1);
iK = reshape(kron(edofMat,ones(8,1))',64*nelx*nely,1);
jK = reshape(kron(edofMat,ones(1,8))',64*nelx*nely,1);
% DEFINE LOADS AND SUPPORTS (HALF MBB-BEAM)
F = sparse(2*nelx*(nely+1)+2,1,-10,2*(nely+1)*(nelx+1),1);
U = zeros(2*(nely+1)*(nelx+1),1);
fixeddofs = [1:2*(nely+1)];
alldofs = [1:2*(nely+1)*(nelx+1)];
freedofs = setdiff(alldofs,fixeddofs);
%% PREPARE FILTER
iH = ones(nelx*nely*(2*(ceil(rmin)-1)+1)^2,1);
jH = ones(size(iH));
sH = zeros(size(iH));
k = 0;
for i1 = 1:nelx
for j1 = 1:nely
e1 = (i1-1)*nely+j1;
for i2 = max(i1-(ceil(rmin)-1),1):min(i1+(ceil(rmin)-1),nelx)
for j2 = max(j1-(ceil(rmin)-1),1):min(j1+(ceil(rmin)-1),nely)
e2 = (i2-1)*nely+j2;
k = k+1;
iH(k) = e1;
jH(k) = e2;
sH(k) = max(0,rmin-sqrt((i1-i2)^2+(j1-j2)^2));
end
end
end
end
H = sparse(iH,jH,sH);
Hs = sum(H,2);
%% INITIALIZE ITERATION
x = repmat(volfrac,nely,nelx);
xPhys = x;
loop = 0;
change = 1;
%% START ITERATION
while change > 0.01
loop = loop + 1;
```

```
%% FE-ANALYSIS
sK = reshape(KE(:)*(Emin+xPhys(:)'.^penal*(E0-Emin)),64*nelx*nely,1);
K = sparse(iK,jK,sK);
K = (K+K')/2;
U(freedofs) = K(freedofs,freedofs)\F(freedofs);
%% OBJECTIVE FUNCTION AND SENSITIVITY ANALYSIS
ce = reshape(sum((U(edofMat)*KE).*U(edofMat),2),nely,nelx); % element-wise
strain energy
c = sum(sum((Emin+xPhys.^penal*(E0-Emin)).*ce)); % total strain energy
dc = -penal*(E0-Emin)*xPhys.^(penal-1).*ce; % design sensitivity
dv = ones(nely,nelx);
%% FILTERING/MODIFICATION OF SENSITIVITIES
if ft == 1
dc(:) = H*(x(:).*dc(:))./Hs./max(1e-3,x(:));
elseif ft == 2
dc(:) = H*(dc(:)./Hs);
dv(:) = H*(dv(:)./Hs);
end
%% OPTIMALITY CRITERIA UPDATE OF DESIGN VARIABLES AND PHYSICAL DENSITIES
l1 = 0; l2 = 1e9; move = 0.2;
while (l2-l1)/(l1+l2) > 1e-3
lmid = 0.5*(l2+l1);
    xnew = max(0,max(x-move,min(1,min(x+move,x.*sqrt(-dc./dv/lmid)))));
    if ft == 1
      xPhys = xnew;
    elseif ft == 2
      xPhys(:) = (H*xnew(:))./Hs;
    end
    if sum(xPhys(:)) > volfrac*nelx*nely, l1 = lmid;
    else l2 = lmid;
    end
end
change = max(abs(xnew(:)-x(:)));
x = xnew;
%% PRINT RESULTS
fprintf(' It.:%5i Obj.:%11.4f Vol.:%7.3f ch.:%7.3f\n',loop,c, ...
mean(xPhys(:)),change);
%% PLOT DENSITIES
colormap(gray);
imagesc(1-xPhys);
caxis([0 1]);
axis equal;
axis off;
drawnow;
end
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% This Matlab code was written by E. Andreassen, A. Clausen, M.
Schevenels,%
% B. S. Lazarov and O. Sigmund, Department of Solid Mechanics, %
% Technical University of Denmark, %
% DK-2800 Lyngby, Denmark. %
% Please sent your comments to: sigmund@fam.dtu.dk %
% %
% The code is intended for educational purposes and theoretical details %
% are discussed in the paper %
% "Efficient topology optimization in MATLAB using 88 lines of code, %
% E. Andreassen, A. Clausen, M. Schevenels, %
% B. S. Lazarov and O. Sigmund, Struct Multidisc Optim, 2010 %
% This version is based on earlier 99-line code %
```

```
% by Ole Sigmund (2001), Structural and Multidisciplinary Optimization, %
% Vol 21, pp. 120--127. %
% %
% The code as well as a postscript version of the paper can be %
% downloaded from the web-site: http://www.topopt.dtu.dk %
% %
% Disclaimer: %
% The authors reserves all rights but do not guaranty that the code is %
% free from errors. Furthermore, we shall not be liable in any event %
% caused by the use of the program. %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
```

References

1. Andreassen, E., Clausen, A., Schevenels, M., Lazarov, B.S. and Sigmund, O., 2011. Efficient topology optimization in MATLAB using 88 lines of code. *Structural and Multidisciplinary Optimization*, *43*(1), pp.1-16.