TOPOLOGY OPTIMIZATION

PROJECT 3 REPORT

YONESHWAR BABU

ASU ID:1220454365

MAE 598: Design Optimization (Fall 2021)

DECEMBER 11, 2021

Under the guidance of

Yi Ren

Assistant Professor of Aerospace and Mechanical Engineering School for Engineering of Matter, Transport and Energy Arizona State University

ACKNOWLEDGEMENT

I consider myself highly fortunate for the opportunity to do this project under the guidance of **YI (MAX) REN** who provided us a sample template code and instructions to work on.

ABSTRACT

In this project, you will learn to implement an optimization algorithm for minimizing the compliance of a cantilever beam with a point load in y direction at its equilibrium state with respect to its topology.

Introduction

I have considered similar conditions from the Sigmund's original paper to perform topology optimization for cantilever beam with a point load at the end of the beam in y direction.

I have used both 88 and 99 lines code to perform topology optimization

88 line matlab code

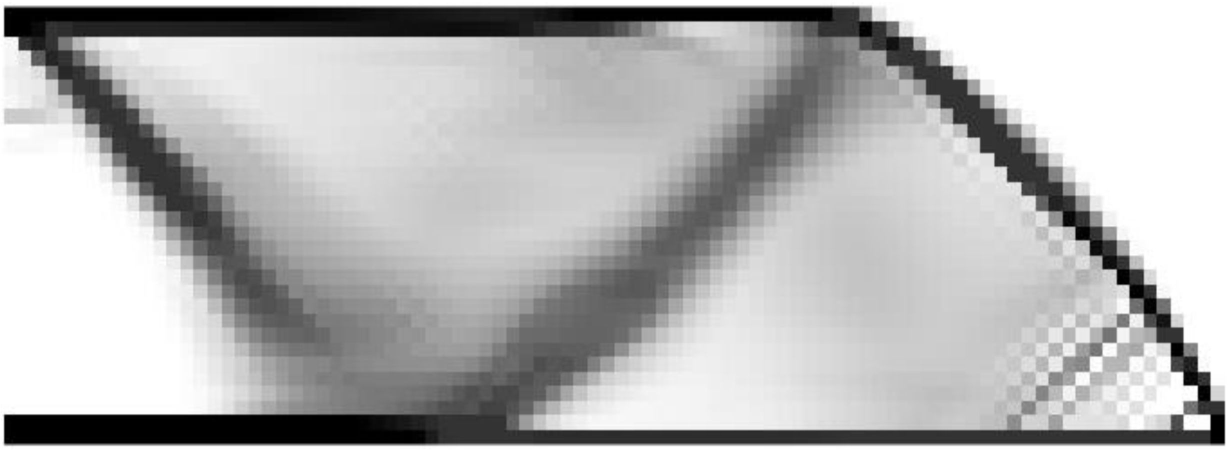
```
%%%% Modified by Max Yi Ren (ASU) %%%%%%%%%%%%%%%%%%%%%%%
%%%% AN 88 LINE TOPOLOGY OPTIMIZATION CODE Nov, 2010 %%%%
nelx=60;
nely=20;
volfrac=0.3;
penal=3;
rmin=1.5;
%function Designproject3(nelx,nely,volfrac,penal,rmin,ft)
%% MATERIAL PROPERTIES
E0 = 1;
Emin = 1e-9;
nu = 0.3;
%% PREPARE FINITE ELEMENT 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(2,1,-1,2*(nely+1)*(nelx+1),1);
U = zeros(2*(nely+1)*(nelx+1),1);
fixeddofs = union((1:2:2*(nely+1)),(2*(nelx+1)*(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 > 0.1
    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
ft=heaviside(x);
    if ft == 2
        dc(:) = H*(x(:).*dc(:))./Hs./max(1e-3,x(:));
    elseif ft == 3
        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
```

output:



99 line Matlab code:

The code was borrowed from " A 99 line topology optimization code written in Matlab" – Ole Sigmond.

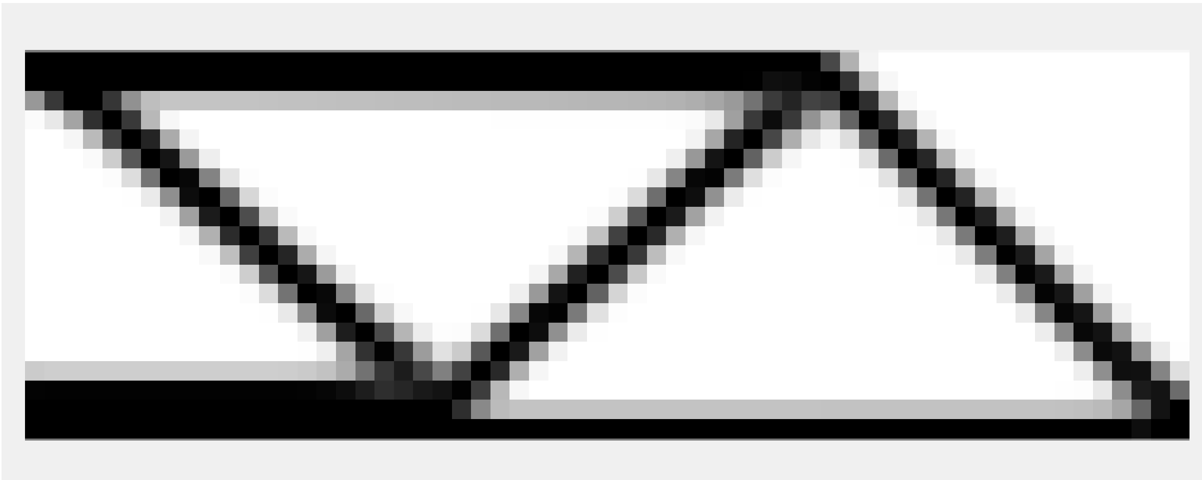
I have used this code since it produces better output.

```
%function (nelx,nely,volfrac,penal,rmin)
% INITIALIZE
nelx=60;
nely=20;
% nelz = 4;
volfrac=0.3;
penal=3;
rmin=1.5;
x(1:nely,1:nelx) = volfrac;
loop = 0;
change = 1.;
% START ITERATION
while change > 0.01
loop = loop + 1;
xold = x;
% FE-ANALYSIS
[U]=FE(nelx,nely,x,penal);
% OBJECTIVE FUNCTION AND SENSITIVITY ANALYSIS
[KE] = lk;
```

```
c = 0.;
for ely = 1:nely
    for elx = 1:nelx
        n1 = (nely+1)*(elx-1)+ely;
        n2 = (nely+1)* elx +ely;
        Ue = U([2*n1-1;2*n1; 2*n2-1;2*n2; 2*n2+1;2*n2+2; 2*n1+1;2*n1+2],1);
        c = c + x(ely,elx)^penal*Ue'*KE*Ue;
        dc(ely,elx) = -penal*x(ely,elx)^(penal-1)*Ue'*KE*Ue;
    end
end
% FILTERING OF SENSITIVITIES
[dc] = check(nelx,nely,rmin,x,dc);
% DESIGN UPDATE BY THE OPTIMALITY CRITERIA METHOD
[x] = OC(nelx,nely,x,volfrac,dc);
% PRINT RESULTS
change = max(max(abs(x-xold)));
fprintf(' It.:%5i Obj.:%11.4f Vol.:%7.3f ch.:%7.3f\n',loop,c, ...
 sum(sum(x))/(nelx*nely),change);
% disp(['It.: 'sprintf('%4i',loop)' Obj.: 'sprintf('%10.4f',c)' ...
% 'Vol.: ' sprintf('%6.3f',sum(sum(x))/(nelx*nely)) ...
% 'ch.: ' sprintf('%6.3f',change)]);
% PLOT DENSITIES
colormap(gray); imagesc(-x); axis equal; axis tight;
 axis off;pause(1e-6);
end
%%%%%%%%%% OPTIMALITY CRITERIA UPDATE %%%%%%%%%
function [xnew]=OC(nelx,nely,x,volfrac,dc)
l1 = 0; l2 = 100000; move = 0.2;
while (l2-l1 > 1e-4)
lmid = 0.5*(l2+l1);
xnew = max(0.001,max(x-move,min(1.,min(x+move,x.*sqrt(-dc./lmid)))));
if sum(sum(xnew)) - volfrac*nelx*nely > 0
 l1 = lmid;
else
 l2 = lmid;
end
end
end
%%%%%%%%%% MESH-INDEPENDENCY FILTER %%%%%%%%%%%
function [dcn]=check(nelx,nely,rmin,x,dc)
dcn=zeros(nely,nelx);
for i = 1:nelx
 for j = 1:nely
 sum=0.0;
 for k = max(i-round(rmin),1):min(i+round(rmin),nelx)
 for l = max(j-round(rmin),1):min(j+round(rmin), nely)
 fac = rmin-sqrt((i-k)^2+(j-l)^2);
 sum = sum+max(0,fac);
 dcn(j,i) = dcn(j,i) + max(0,fac)*x(l,k)*dc(l,k);
 end
 end
 dcn(j,i) = dcn(j,i)/(x(j,i)*sum);
 end
end
end
%%%%%%%%%% FE-ANALYSIS %%%%%%%%%%%%
```

```
function [U]=FE(nelx,nely,x,penal)
 [KE] = lk;
 K = sparse(2*(nelx+1)*(nely+1), 2*(nelx+1)*(nely+1));
 F = sparse(2*(nely+1)*(nelx+1),1);
 U = sparse(2*(nely+1)*(nelx+1),1);
    for ely = 1:nely
        for elx = 1:nelx
            n1 = (nely+1)*(elx-1)+ely;
            n2 = (nely+1)* elx +ely;
            edof = [2*n1-1; 2*n1; 2*n2-1; 2*n2;2*n2+1;2*n2+2;2*n1+1; 2*n1+2];
            K(edof,edof)=K(edof,edof)+x(ely,elx)^penal*KE;
        end
    end
% DEFINE LOADS AND SUPPORTS(HALF MBB-BEAM)
F(2,1) = -1;
fixeddofs = union([1:2:2*(nely+1)],[2*(nelx+1)*(nely+1)]);
alldofs = [1:2*(nely+1)*(nelx+1)];
freedofs = setdiff(alldofs,fixeddofs);
% SOLVING
U(freedofs,:) = K(freedofs,freedofs) \ F(freedofs,:);
U(fixeddofs,:)= 0;
end
%%%%%%%%%% ELEMENT STIFFNESS MATRIX %%%%%%%
function [KE]=lk
 E = 1.;
 nu = 0.3;
 k=[ 1/2-nu/6 1/8+nu/8 -1/4-nu/12 -1/8+3*nu/8 ...
 -1/4+nu/12 -1/8-nu/8 nu/6 1/8-3*nu/8];
 KE = E/(1-nu^2)*...
 [ k(1) k(2) k(3) k(4) k(5) k(6) k(7) k(8)
 k(2) k(1) k(8) k(7) k(6) k(5) k(4) k(3)
 k(3) k(8) k(1) k(6) k(7) k(4) k(5) k(2)
 k(4) k(7) k(6) k(1) k(8) k(3) k(2) k(5)
 k(5) k(6) k(7) k(8) k(1) k(2) k(3) k(4)
 k(6) k(5) k(4) k(3) k(2) k(1) k(8) k(7)
 k(7) k(4) k(5) k(2) k(3) k(8) k(1) k(6)
 k(8) k(3) k(2) k(5) k(4) k(7) k(6) k(1)];
end
```

Output:



Conclusion:

The topology optimization for the cantilever beam, where the volume is effectively reduced by determining the load acting on different places on beam .

References:

1. Efficient topology optimization in MATLAB using 88 lines of code - Erik Andreassen, Anders Clausen, Mattias Schevenels, Boyan S. Lazarov, Ole Sigmund.
2. A 99 line topology optimization code written in Matlab - Ole Sigmond.
3. Topology optimization tutorial – Yi (Max) Ren.