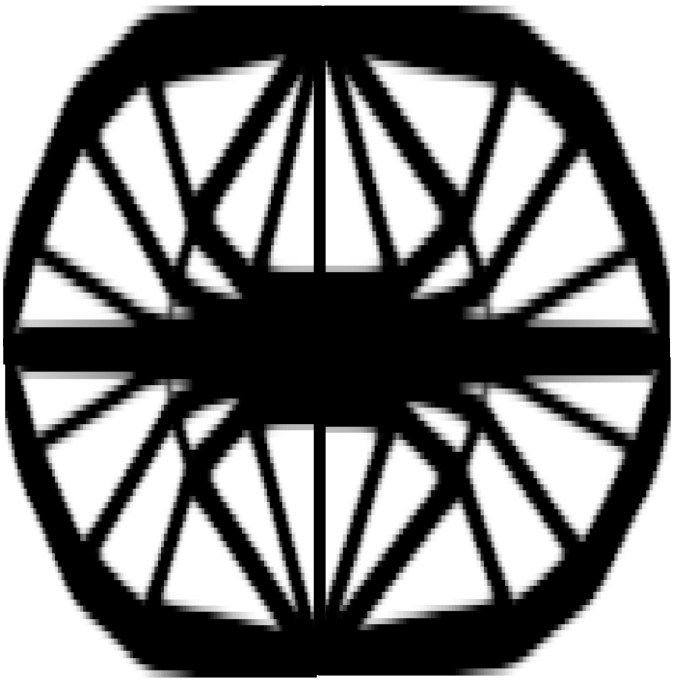


A Topology Optimization output using Finite Elements Implemented on a Structural problem

TOPOLOGY OPTIMIZATION USING FINITE ELEMENT METHODS BASED ON OPTIMALITY CRITERIA

-Manthan N. Dhisale (193109014)

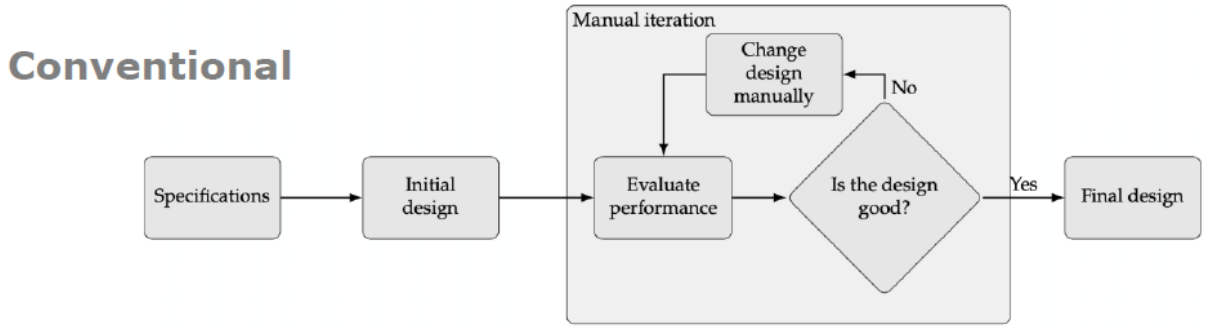
Introduction

Optimization can be referred in many ways. People optimize like investors optimize for getting maximum return on investments, manufacturers optimize to seek maximum efficiency in their production process while the engineers optimize their designs by adjusting various parameter to seek maximum performance from the system.

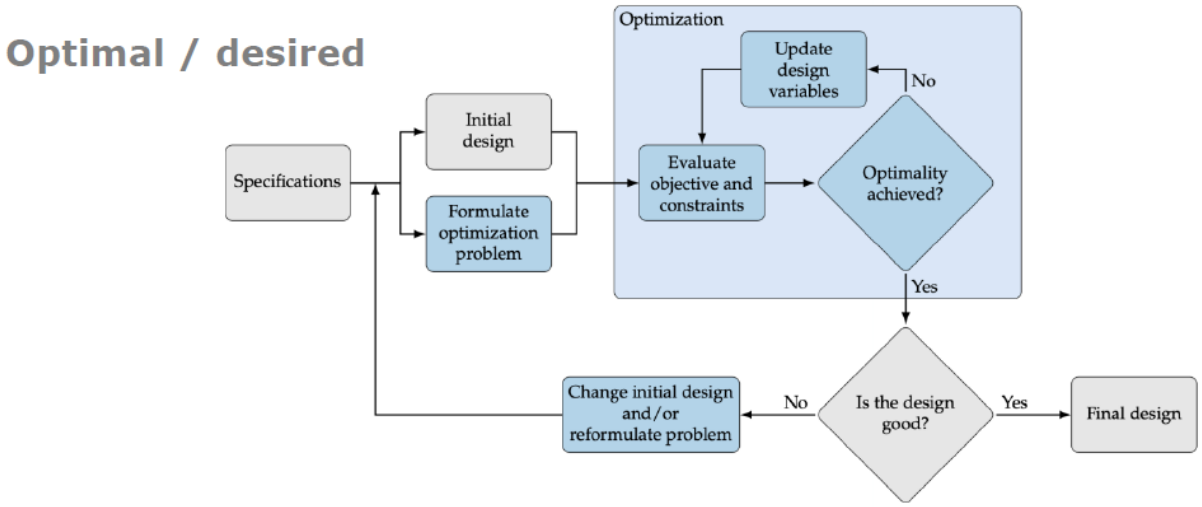
Even it's fascinating to note that Mother Nature optimizes. It optimizes various systems for minimum potential energy. Even during chemical reactions molecules react in such a way that the distance between the newly formed atomic interactions (bonds) has minimum total energy of reaction.

Optimization is an important tool for analysis of physical systems. Our goal is to find values of the variables that optimize (maximize or minimize) the objective of which the variables are either restricted or constrained by some ways.

A Conventional design process without involvement of optimization techniques can be understood as follows. The major pitfall of this process is that, the design process has to go over a number of iterations, until the desired performance or cost target is achieved.



On the contrary, with involvement of optimization, the conventional process can be simply modified as follows:



Mathematical Preliminaries

Mathematical Formulation:

Any optimization problem can be mathematically modelled in the following fashion:

$$\text{x}\epsilon R^n \; where \; x \; are \; design \; variables$$

$$\text{p}\epsilon R^m \; where \; p \; are \; parameters$$

$$\text{J}: (R^n, R^m) \rightarrow R^l \; where \; J \; are \; objective \; functions$$

$$\text{g}: (R^n, R^m, R^l) \rightarrow R^k \; where \; g \; are \; inequality \; constraints$$

$$\text{h}: (R^n, R^m, R^l) \rightarrow R^j \; where \; h \; are \; equality \; constraints$$

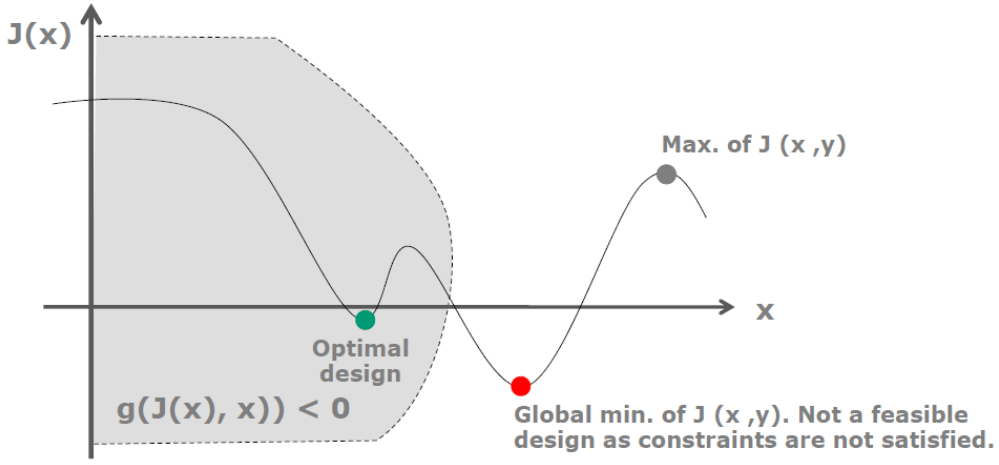
$$\text{find } x^* | \; \text{J}(x^*) \leq \text{J}(x^*)$$

$$\text{g}(x^*) \leq 0$$

$$\text{h}(x^*) = 0$$

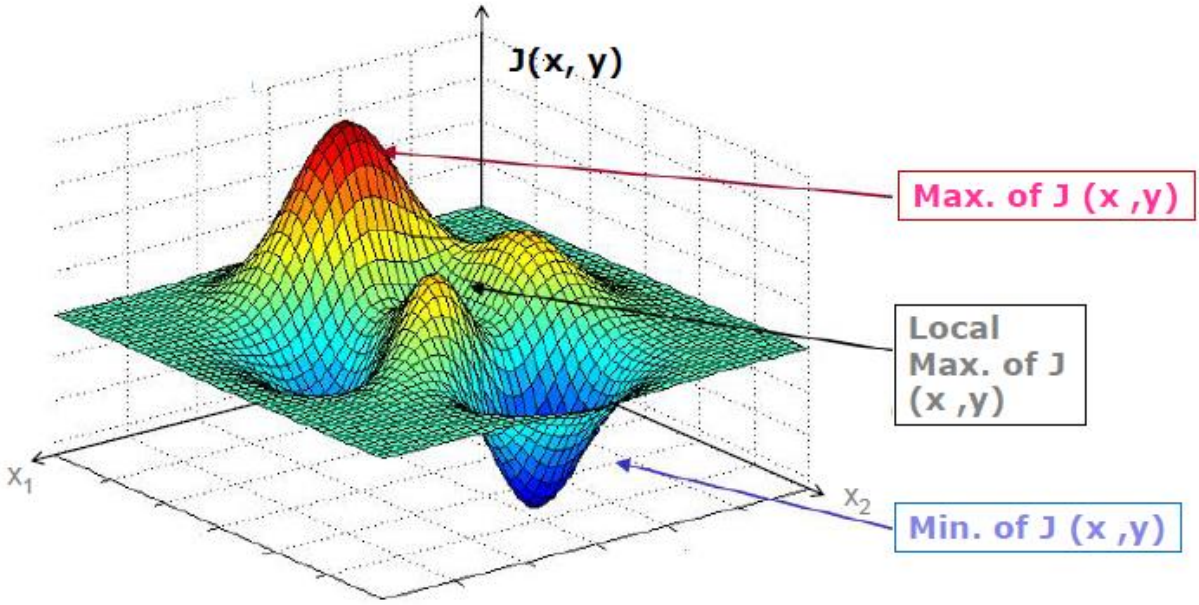
- Constraints arise because of finiteness of variables such as space, time, resources and limitations on available technology. They are the boundaries of design variables.
- Design variables cannot violate constraints while evaluating objective function
- In several situations, optimal designs lie at the intersection of several constraints
- Constraints can be of three types:
    1. Bounds $x_{iLB} \leq x_i \leq x_{iUB} \; where \; i = 1,2,3, \ldots \ldots n$
    2. Inequality $g_j(\text{x}) \leq 0 \; where \; j = 1,2,3, \ldots \ldots m1$
    3. Equality $h_k \leq 0 \; where \; k = 1,2,3, \ldots \ldots m2$

Design Optimization for 1D problems:

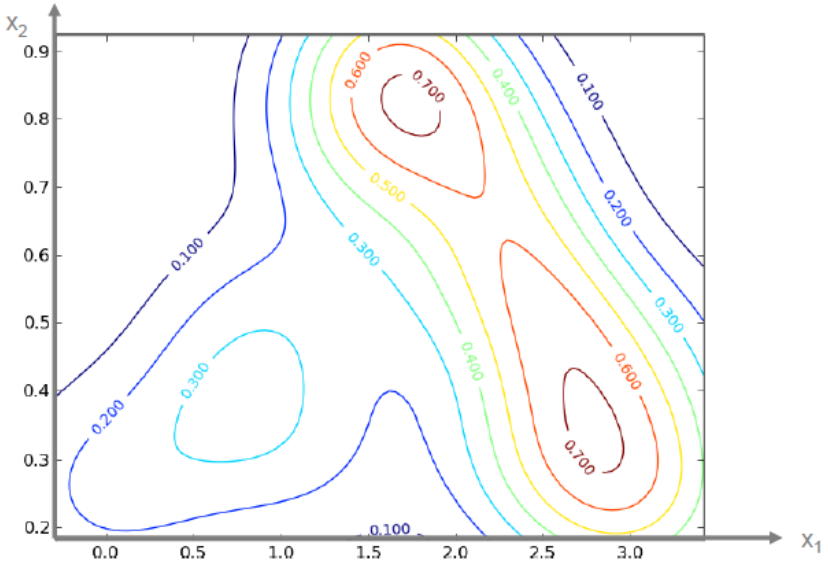


J(x) is the cost function that is to be minimized
X is the design variable
g(J(x), x) are inequality constraints
It is customary to minimize the cost function. If a quantity is to be maximized, the cost function is the negative of that quantity.

Design Optimization for 2D problems:



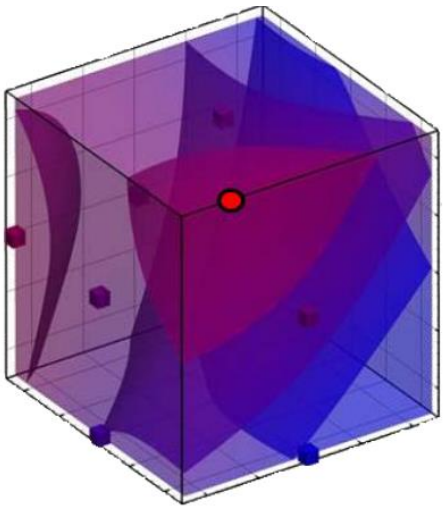
Here J(x) is the objective function to be optimised and x1, x2 are design variables



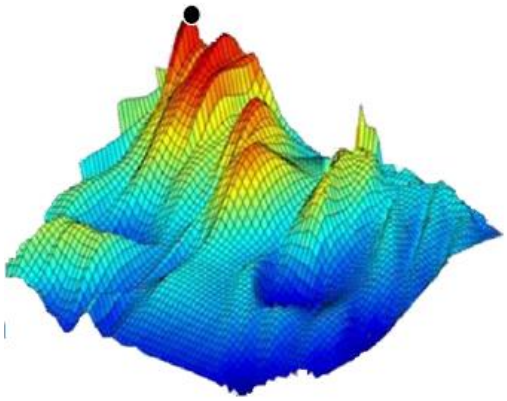
On each contour J is constant. The crowding of contours indicate rapid change in objective function value

Design Optimization for 3D and beyond problems:

Colours represent magnitude of objective function. Graphical methods can still be employed to identify and visualize maxima or minima. Graphical methods can still be used beyond objective functions with more than 3 variables, however it starts receiving a limitation beyond 4 or 5 design variables. Below plots show the objective function value in 3D and 4D space

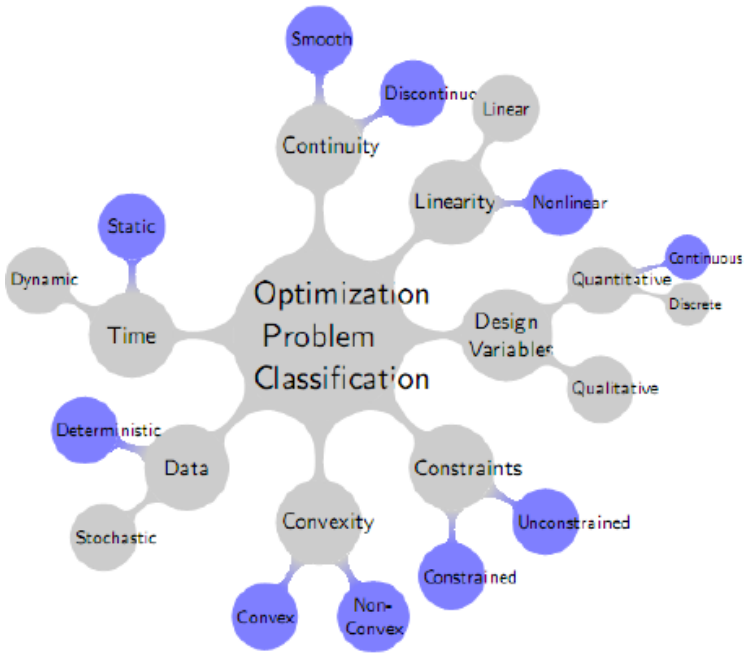


Plot for 3-D space



Plot for 4-D space

Classification of Optimization problems:



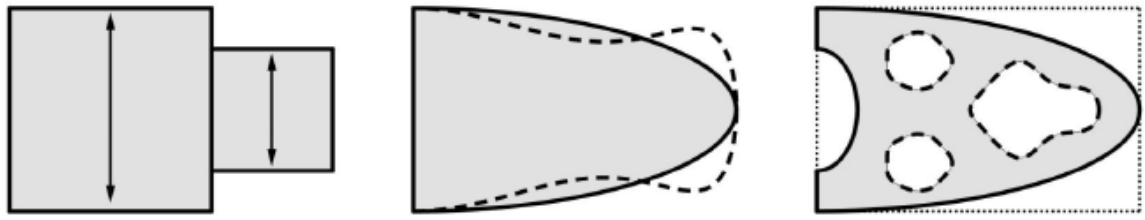
Basis for classifying the optimization problems:

- Continuity of J
- Linearity of J
- Types of design variables
- Types of constraints
- Nature of data
- Time variant / invariant

Structural Optimization

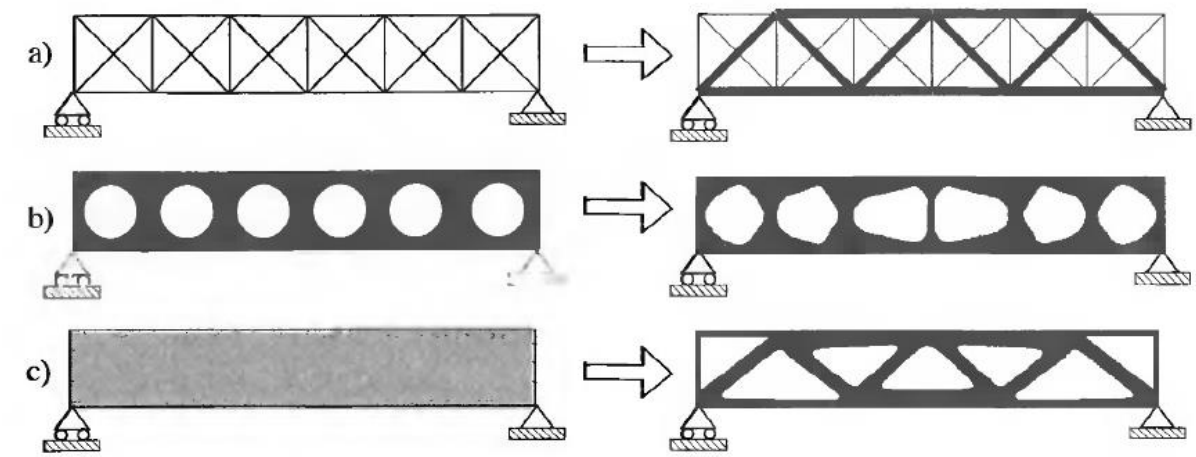
Sizing, shape and topology address different aspects of structural design problem. In a typical sizing (parametric) problem, the goal may be to find the optimal thickness distribution of a linearly elastic plate or optimal thickness distribution of a linearly elastic plate or truss structure. The optimal thickness distribution minimizes (or maximizes) a physical quantity such as mean compliance (external work), peak stress, deflection, etc. while equilibrium and other constraints on the state and design variables are satisfied. The design variable is the thickness of the plate and the state variable may be its deflection. The main feature of the sizing problem is that the domain of the design model and state variables is known apriority and is fixed throughout the optimization process. On the other hand, in shape optimization problem, the goal is to find the optimum shape of the domain, that is the shape problem is defined on a domain which is now a design variable. Topology optimization of the solid structures involves the determination of features such as the number and location and shape of the holes and connectivity of the domain.

In the aspects of a solid/object body:



1) Parametric optimization 2) Shape optimization 3) Topology optimization

In the aspects of a truss/2D frame:



a) Sizing/Parametric Optimization of 2D truss structure b) Shape optimization of 2D plate with circular holes c) Complete solid/filled body with topology optimization

Initial problem formulations are shown on the left hand side and optimized structures on right hand side.

Methodology

Parametric Shape Optimization

Consider the membrane problem. Suppose that we wish to find an optimal thickness distribution $h \in H$ at minimizes the objective function

$$J(h) = \iint_\Omega j(u(x))\, dx,$$

Where J : R → R is a specified continuously differentiable function
Note that the objective function does not contain h explicitly. Rather, u depends on h via the solution of

$$-\nabla \cdot (h(x)\nabla u(x)) = f(x),\ x \in \Omega, \quad u(x) = 0,\ x \in \partial\Omega.$$

In addition, we want to minimize J over an admissible set of thickness functions H. For instance,

$$\mathcal{H} = \left\{ h : \Omega \to \mathbb{R}_+ \;\middle|\; h_{\min} \le h(x) \le h_{\max}, \iint_\Omega h(x)\, dx = h_0|\Omega| \right\}$$

Solution with Projected Gradient Descent

If we are able to compute the functional derivative δJ(h) / δh of J with respect to h, we can use the projected gradient algorithm to find the optimal h:

$$h_{n+1} = P_{\mathcal{H}} \left(h_n - \alpha_n \frac{\delta J(h_n)}{\delta h} \right)$$

We compute the projection $P_H(h)$ of h : Ω → R as follows:

$$P_{\mathcal{H}}(h) = \max(h_{\min}, \min(h + \ell, h_{\max})),$$

where the constant $l \in R$ is chosen such that

$$\iint_\Omega P_{\mathcal{H}}(h)(x)\, dx = h_0|\Omega|.$$

In practice, simple algorithms like the bisection method are used to estimate l

Adjoint Method to compute δJ(h) / δh

Direct calculation of δJ(h) / δh is cumbersome since J depends indirectly on h via the dependence of u on h.
The adjoint method provides an efficient and elegant alternative to compute this functional derivative.
The Lagrangian for the constrained minimization of J is written using the adjoint field p : Ω → R as

$$L(h,u,p) = \iint_\Omega j(u(x))\, dx - \iint_\Omega p\left(\nabla \cdot (h(x)\nabla u(x)) + f(x)\right)\, dx$$

Note that setting δL(h; u; p)/ δp = 0 yields the governing equation for u. We will call this the primal equation.
δL(h; u; p)/ δp = 0 is called the adjoint equation. For the current problem, the adjoint equation takes the form

$$-\nabla \cdot (h(x)\nabla p(x)) = -j'(u(x)), \quad x \in \Omega,$$
$$p(x) = 0, \quad x \in \partial\Omega.$$

Denote the solutions of the primal and adjoint equations as uh and ph, respectively.
The required gradient δJ/δh : $\Omega \rightarrow$ R is now computed as

$$\frac{\delta J(h)}{\delta h}(x) = \nabla u_h(x) \cdot \nabla p_h(x).$$

This technically corresponds to an L2 projection of the functional derivative δJ/δh

Regularization of Iterates

The iterates h_n(x) often do not have the right regularity, and hence need to be regularized.
The simplest option is to apply a filter to h. For instance, the Heaviside filter acts on h to produce a more regular H_βh, for large positive constant β, as

$$H_\beta h(x) = 1 - \exp(-\beta h(x)) + h(x)\exp(-\beta).$$

One could also instead do a smoothing of h directly by solving, for small β > 0,

$$-\epsilon^2\nabla^2\tilde{h}(x) + \tilde{h}(x) = h(x), \quad x \in \Omega,$$
$$\nabla\tilde{h}(x) \cdot n(x) = 0, \quad x \in \partial\Omega.$$
$$-\epsilon^2\nabla^2 q(x) + q(x) = \nabla u_h(x) \cdot \nabla p_h(x), \quad x \in \Omega,$$
$$\nabla q(x) \cdot n(x) = 0, \quad x \in \partial\Omega.$$

Denoting by qh the solution of this equation,

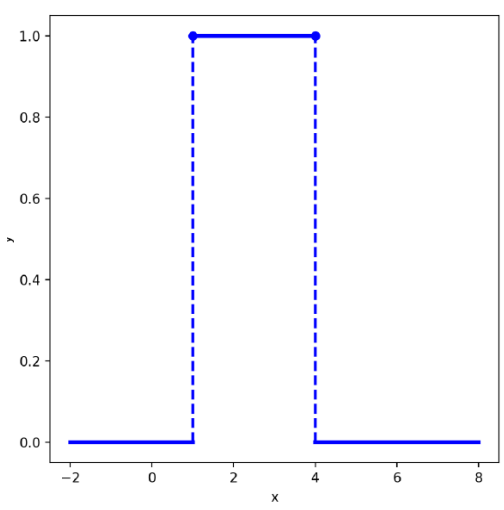
$$(\delta J(h)/\delta h)(x) = q_h(x)$$

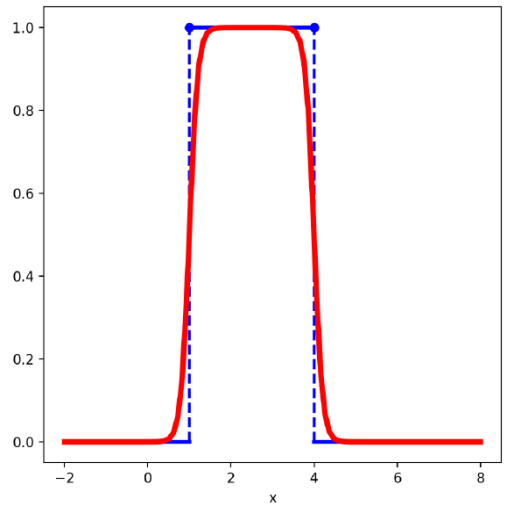
Pseudo Parametric Shape Optimization Algorithm

1. Initialize thickness h.
2. Iterate until convergence:
   - Solve the primal equation to find uh.
   - Solve the adjoint equation to find ph.
   - Compute gradient of objective function, δJ/δh, using uh and ph.
   - Update h: h = h - $\alpha * (\delta$J/δh)
   - Enforce constraints on h, if any.
   - Regularize h.

In order to extend the Shape Optimization Algorithm for Topology Optimization we follow the below approach

Density based Homogenization





Given a subdomain A ⊆ Ω of the domain Ω ⊆ R^d, d > 1, the characteristic function x_A : Ω → R defined as

$$\chi_A(\boldsymbol{x}) = \begin{cases} 1, & \boldsymbol{x} \in A, \\ 0, & \boldsymbol{x} \notin A, \end{cases}$$

We define a density ρ_A : Ω → [0; 1] as a smooth approximation of the characteristic function x_A
Given a density function ρ, the subdomain of Ω that it approximates can be extracted from it numerically using a tolerance ϵ > 0 as

$$A(\rho) = \{\boldsymbol{x} \in \Omega \,|\, \rho(\boldsymbol{x}) > \epsilon\}.$$

SIMP: Smooth Interpolation of Material Properties

Consider an initial design where the material, with material property C, occupies a region Ω ⊆ R^d. If the optimal material distribution that satisfies the external constraints only occupies a subdomain A ⊆ Ω, then this can be approximated using the ersatz material approximation as C_ϵ : Ω → R, for small ϵ > 0:

$$\boldsymbol{C}_\epsilon(\boldsymbol{x}) = \chi_A(\boldsymbol{x})\boldsymbol{C} + (1 - \chi_A(\boldsymbol{x}))\epsilon\boldsymbol{C}$$

Smoothing $x_A$with a density ρ, we can reframe the topology optimization problem in terms of _ by replacing the material property C with C_ϵ. This is called the Smooth Interpolation of Material Properties (SIMP) method.
The rest of the algorithm is the same as discussed for parametric optimization.
Finally, the optimal domain can be extracted from ρ. In practice, interpolating functions ζ: R →R are used to sharpen the density. A common choice is ζ (t) = t^3.

Topology Optimization Code and Results

Following the above mentioned algorithm and the techniques, a MATLAB code proposed by Sigmund using Finite elements is as follows:

```
function top(nelx,nely,volfrac,penal,rmin);
% INITIALIZE
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)));
  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
%%%%%%%%% MESH-INDEPENDENCY FILTER
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [dcn]=check(nelx,nely,rmin,x,dc)
```

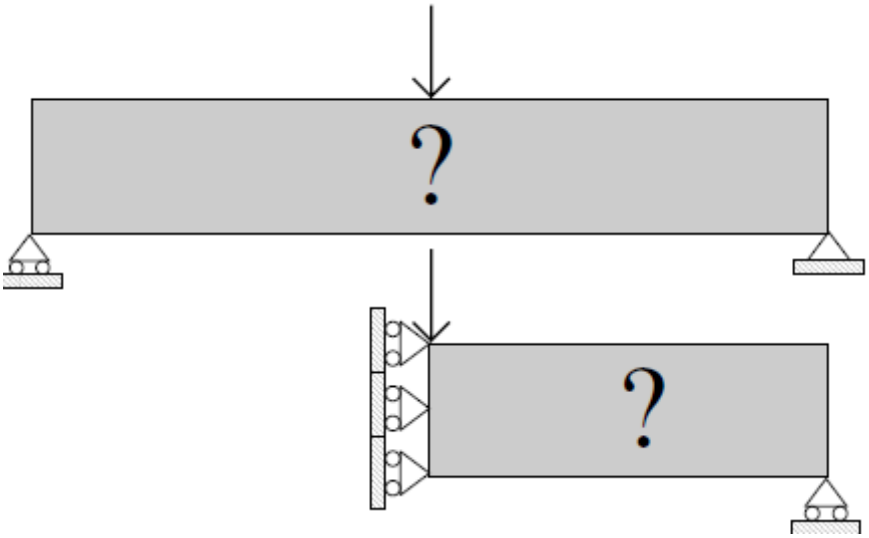
```
dcn=zeros(nely,nelx);
for i = 1:nelx
  for j = 1:nely
    sum=0.0;
    for k = max(i-floor(rmin),1):min(i+floor(rmin),nelx)
      for l = max(j-floor(rmin),1):min(j+floor(rmin),nely)
        fac = rmin-sqrt((i-k)^2+(j-l)^2);
        sum = sum+max(0,fac);
        dcn(j,i) = dcn(j,i) + max(0,fac)*x(l,k)*dc(l,k);
      end
    end
    dcn(j,i) = dcn(j,i)/(x(j,i)*sum);
  end
end
%%%%%%%%% 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 = zeros(2*(nely+1)*(nelx+1),1);
for elx = 1:nelx
  for ely = 1:nely
    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;
%%%%%%%%% 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)];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

In the code topology (nelx,nely,volfrac,penal,rmin) where nelx and nely are the number of elements in the horizontal and vertical directions, respectively, volfrac

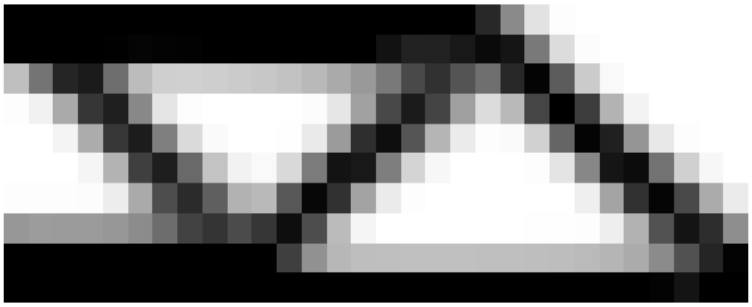
is the volume fraction, penal is the penalization power and rmin is the filter size (divided by element size).

The Sigmund code is implemented on structural mechanic problem as follows:
A Point load is applied at the centre of beam supported at two supports can be simplified utilizing its symmetry as follows:



>> topology (nelx,nely,volfrac,penal,rmin)

>> topology (30, 10, 0.5, 3.0, 1.5)



Further refinement can be done as follows:

>> topology (80, 50, 0.5, 3.0, 1.5)

The optimized topology obtained is as follows:

References

1. Sigmund, O. 1994: Design of material structures using topology optimization. Ph.D. Thesis, Department of Solid Mechanics, Technical University of Denmark
2. Sigmund, O. 1997: On the design of compliant mechanisms using topology optimization. Mech. Struct. Mach. 25, 495–526
3. Zhao, C.; Hornby, P.; Steven, G.P.; Xie, Y.M. 1998: A generalized evolutionary method for numerical topology optimization of structures under static loading conditions. Struct. Optimization 15, 251–260
4. Zhou, M.; Rozvany, G.I.N. 1991: The COC algorithm, part II: Topological, geometry and generalized shape optimization. Comp. Meth. Appl. Mech. Engrng. 89, 197–224

Updated Plan of Action

Summer of Science Topic: Topology Optimization using Finite Element Methods Based on Optimality Criteria

	Planned Activity		Completed Activity

Work Activity by weeks	May		June				July	
	3rd	4th	1st	2nd	3rd	4th	1st	2nd
Plan of Action Submission								
Literature Review								
Weekly meet with mentor for progress tracking (if mentor is available)								
Hands on with Fenics software for FEM computation								
Selection of a particular topology optimization method with selection of suitable application (structural+thermal)								
Development Pseudo-codes showcasing the algorithm implementation								
Midterm report preparation and expected submission								
Final coding and validating the results using ANSYS/ABACUS								
SoS report submission and video submission								