# Objective

The objective of this assignment is to write the code for the simulated annealing algorithm that allows for a one-pass penalty and quadratic penalty. Then, using this simulated annealing code find an appropriate c and epsilon value given the bounds using the bump function. Then, use this value for running the SA algorithm on a 10-bar truss for optimizing the weight of the trusses.

# Code Structure

Please see the docstrings on the Python files.

# Bump Function Results

The primary procedure for getting a good value for the cooling schedule parameter, c, is by trial and error after fixing a value for the rest of the parameters. As suggested by the assignment, the following parameter values are used that were fixed, so that only one parameter will be changed at a time:

t\_start (starting temperature) = 1000

max\_iter (maximum number of iterations) = 5000

epsilon = 2

The epsilon value was chosen as per the suggestion on the lecture slide which says it is typically between 0.1-0.3 – and I simply chose the middle value which is 0.2. However, since I did not normalize move.py, epsilon had to be multiplied by the range of the bounds i.e. upper bound – lower bound. The upper bound is 10, therefore, 0.2\*10 = 2.

The initial design variable value, x0 is [1, 1] (since n=2, per the assignment). Also, per the assignment, lower bound is [0, 0] and upper bound is [10, 10] and are all inputted into move.py as floats.

Discuss the procedure you adopted to obtain a good value of the exponential cooling schedule parameter c (include averaged convergence trends for typical values of c to justify your conclusions)

Report the best solution (x & f)

The best solution for x and f, outputted as xopt and fopt in my Assignment3.py file over multiple times of running the same file and averaging the values are the following:

xopt [ 1.59415299 0.47298035]

fopt -0.362774184804

The best value for the parameter c is 0.996

A more rigourous approach is to create a linspace from 0.98 to 0.999 split into 20 segments and each segment is ran 5 times and these runs are averaged. I also added a random seed to reduce the randomness, though this did not have a noticeable effect.

{ Results for Bump function (2D) [3pts]

# 10-bar truss optimality

{ Optimal design of the 10 Bar truss structure [4pts]

## One-Pass Penalty vs Quadratic Penalty

Compare averaged convergence trends for the one-pass penalty and

quadratic penalty function approach (3 typical values of the penalty

parameter)

**Choose at least 3 penalty parameter by changing alpha and fixing everything else.**

Discuss how you went about selecting appropriate values of the penalty

parameter for the study

# Comparison

Report the best solution obtained using both approaches

Are your optimal designs feasible?

Because there is no buckling it is an incredibly thin and light weight truss structure. It is not feasible because we have to account for buckling in truss designs.