Ryan Reedy	AMDS **Homework 4**	**16 Apr 2019**

Note: Parts 1 and 2 were submitted on 15 April 2019 as Matlab live script PDFs. Since I used an entirely different codebase for Part 3 (it shares code with my semester project), I am submitting this separately.

Part 3: Cantilever Beam Design

Beams are built up from point masses on a 9x6 grid represented genetically by 54 bit chromosomes, where a 1 indicates the presence of a point, which can be easily reshaped into a matrix. Adjacent points in the matrix (including diagonals) are then connected by spring members with spring constant k = 200.

This program attempts to optimize both deflection and mass. Deflection due to a distributed +0.25 force in the y-direction applied to the right end is computed via the direct stiffness method [1,2]. For comparison, a dynamic simulation that takes 1.4 seconds to calculate the equilibrium position takes 0.04 seconds with the `directStiffness()` function. Damping is ignored because this is a static analysis. Mass is calculated assuming each member has a mass of 1 per unit length.

The NSGA-II algorithm is implemented in Matlab, modified from [3]. In the `genetic_operator()` function, crossover occurs with a probability of 90%. Simulated binary crossover in the original code is replaced with simple binary crossover (Figure 1). Mutation occurs with a probability of 10%, where each gene bit has a 10% chance of being flipped (Figure 2). “Perverse” solutions, including those having disjoint points or no points at the right end, are not automatically discarded, but are heavily penalized.

```
prob = 0.5;
mask = rand(1, V) < prob;
child_1(mask) = parent_1(mask);
child_1(~mask) = parent_2(~mask);

child_2(mask) = parent_2(mask);
child_2(~mask) = parent_1(~mask);
```

Figure 1: Binary crossover

```
mut_prob = 0.1;
mask = rand(1, V) < mut_prob;
child_3(mask) = ~child_3(mask);
```

Figure 2: Binary mutation

Running the algorithm with a population size of 100 over 20 generations takes about 10 minutes. See Figures 3 and 4 for results.

Running the algorithm with a population size of 100 over 20 generations takes about 10 minutes. See Figures 3 and 4 for results, as processed by `plotPopulation();`

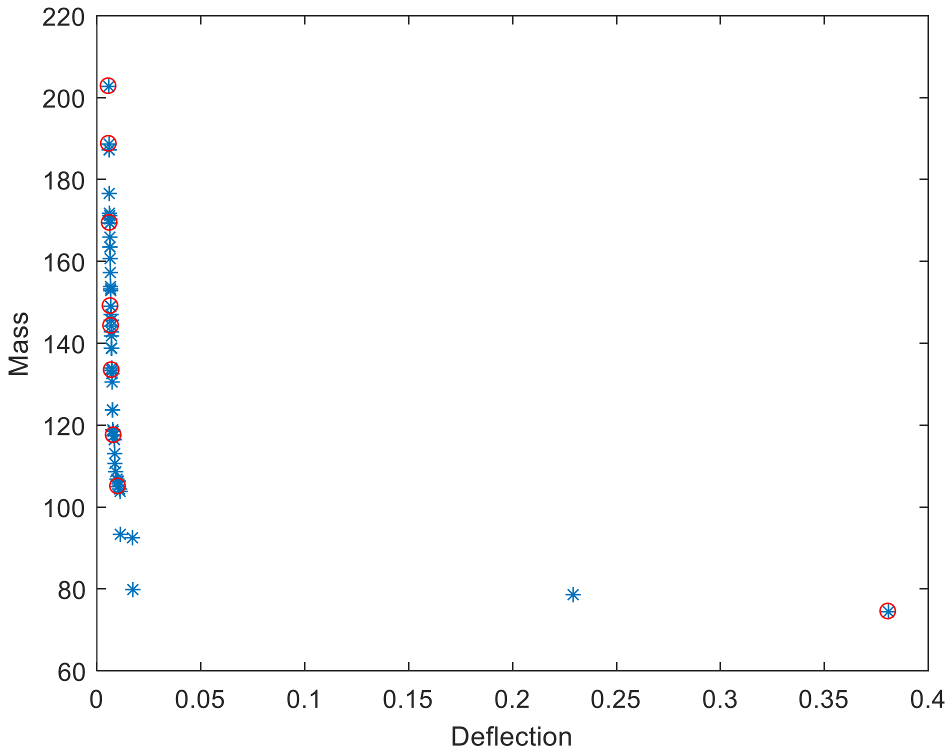

Figure 3: Pareto front

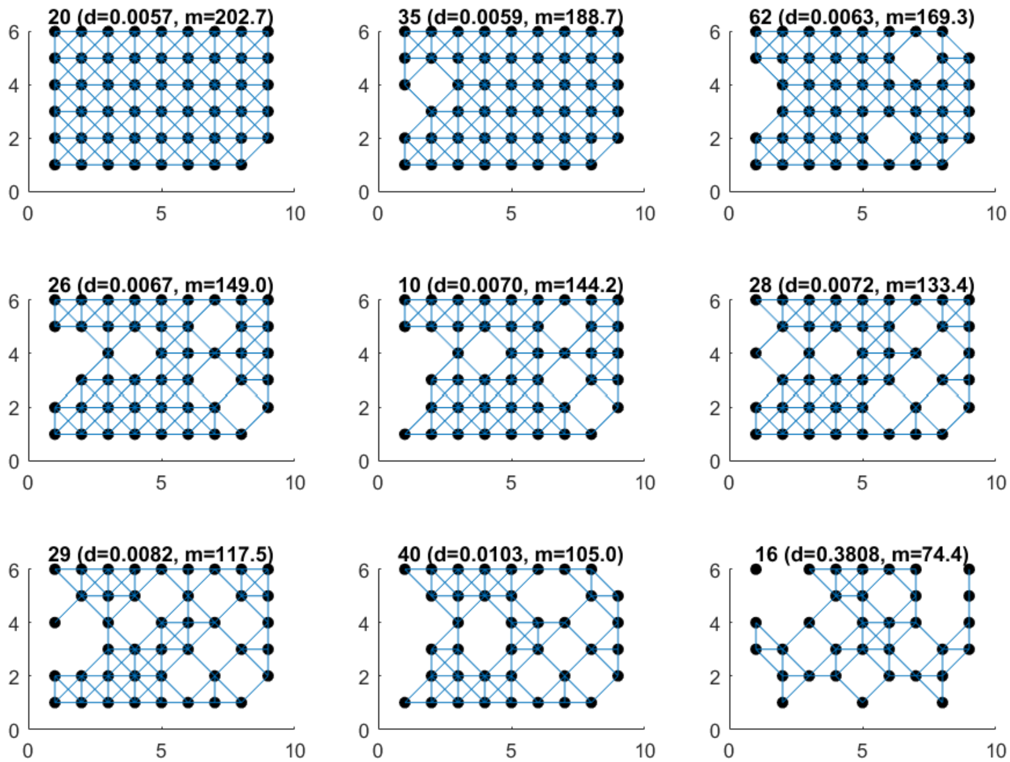

Figure 4: Sample of beam solutions taken from the Pareto front, sorted by ascending deflection

References

[1] https://engineering.purdue.edu/~aprakas/CE474/CE474-Ch5-StiffnessMethod.pdf

[2] http://people.duke.edu/~hpgavin/cee421/truss-method.pdf

[3] https://www.mathworks.com/matlabcentral/fileexchange/10429-nsga-ii-a-multi-objective-optimization-algorithm