Carnegie Mellon University, Department of Mechanical Engineering
Homework 6
Genetic Algorithms and Optimization

Genetic Algorithms

1. *(30 points)*

In this programming exercise, you will solve the *map coloring problem* using a genetic algorithm. More specifically, your task is to color a map of the United States, subject to the constraints that each state must be one of four possible colors, and no two bordering states may have the same color. The main function template (`ga_mapcoloring.m`) is provided, which you need to edit to perform the genetic algorithm correctly. This file already contains a helper function called `showmap`; you do not need to edit this helper function at all, but you should call it inside the main function when you want to visualize the colored map. Also included with the assignment is a data file called `usmap.mat`, which contains the following:

`L`	A 570x786 label matrix, representing the map image. Each pixel in the image takes an integer value from 0-51, where 0 indicates lines, 1 represents the background, and 2-51 refer to the 50 states. For example, the pixels representing Pennsylvania all have a value of 41 in `L`. A simple way to view this image is with the following code: `imshow(label2rgb(L));`
`A`	The 50x50 adjacency matrix, in which `A(i,j)=1` if the *i*th and *j*th states border each other and 0 otherwise.

A suggested list of GA parameters/methods to implement is given below. You are free to modify these parameters if you are unsatisfied with your algorithm performance. In your report, make sure to specify any changes you made.

Representation	integers (1-4)
Population size (μ)	500
Elitism (e)	2
Number of offspring (λ)	498
Crossover method	uniform
Crossover rate	0.9
Mutation method	random resetting
Mutation rate	0.1
Parent selection	tournament (k = 10)
Survivor selection	generational
Initialization	random
Termination condition	solution or 500 iterations

After completing your code, you should be able to test your algorithm by typing the following line in the command window (or in a script):

```
>> [solution,score] = ga_mapcoloring();
```

The output includes the best solution, encoded as a 50x1 vector of integers (`solution`), and the corresponding scalar fitness (`score`). In your report, include one of the best solutions from running your algorithm as a colored image (see example below). In addition, complete the following tasks:

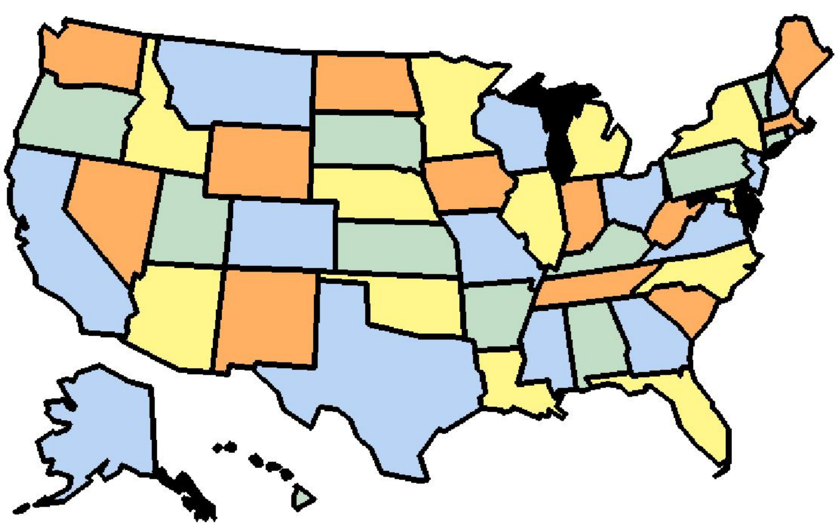
(a) Run your algorithm multiple times (at least 10). Compute the average best fitness (the mean score of the best solution at the end of a run) and the average accuracy (the percentage of runs in which a valid solution is found).

(b) In a single plot, display the best fitness value as a function of iterations for all runs. Do the results match your intuition? Explain in 1-2 sentences.

(c) Now, vary the population size (μ) from 10-10000; you should simultaneously scale the offspring size ($\mu - e$) and tournament size (e.g. k = 0.02μ). For each case, run the GA multiple times to assess performance. Discuss the effect of population size on performance metrics including mean accuracy, average number of iterations required to find the solution, average run time, etc. Include plots or a table of numeric results to convey your ideas.

(d) As you may have noticed already, there are many valid solutions to this problem. But, suppose you wanted to find a valid solution in which certain states were specific colors. For instance, how could you modify your algorithm to guarantee that the best solution at the end of a run included Florida and Pennsylvania as blue states? Implement your idea and include the final image of the colored map in your report. It may help to know that Florida and Pennsylvania are the 33rd and 40th genes in the chromosome and an allele of 1 corresponds to the color blue.

Extra credit: *(10 points max)* In the current implementation, there is no guarantee that each color will be used an equal number of times. How would you impose this constraint? Explain in a few sentences. Try to implement your idea(s) and verify that it works.



Quadratic Programming

2. *(25 points)*

Given the objective function $0.5x^2 + 3.2x + 4.5y^2 - 6y + 7$

Subject to:

$x, y \geq 0$

$x + 3y \geq 15$

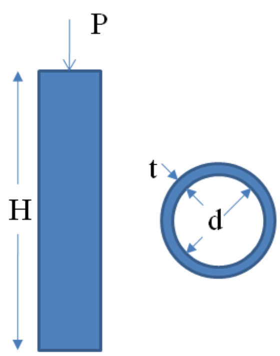
$2x + 5y \leq 100$

$3x + 4y \leq 80$

- Study Matlab's `quagprog` function. Note that this is a quadratic optimization problem with linear constraints.
- Write the above optimization problem in the standard form by identifying the matrices and vectors `H,f,A,b,Aeq,beq,lb,ub`. Be careful with the coeffficients in the standard form.
- Solve the problem using Matlab's quadprog "`'interior-point-convex'` `(default)`" option.
- Repeat the first three steps using Matlab's `fmincon` function

Mechanical Design Optimization

3. *(25 points)*



You are asked to design a column to support a compressive load P as shown in the figure. The column has a cross-section shaped as a thin-walled pipe. The design variables are the mean pipe diameter d and the wall thickness t. The cost of the pipe is given by:

$$\text{Cost} = f(t,d) = c_1 W + c_2 d$$

Where $c_1 = 4$ and $c_2 = 2$ are the cost factors and W = weight of the pipe. The column must support the load under compressive stress and not buckle.

The compressive stress is given by: $\sigma = \frac{P}{A} = \frac{P}{\pi d t}$

The buckling stress can be shown to be: $\sigma_b = \frac{\pi E I}{H^2 d t}$

where E is the modulus of elasticity and I is the second moment of the area of the cross section given by:

$$I = \frac{\pi}{8} d t (d^2 + t^2)$$

Finally, the diameters of available pipes are between d_1 and d_2, and thicknesses between t_1 and t_2.

(a) Develop the optimization problem on paper. Clearly show your mathematical model (e.g. objective function, design variables, design parameters you find useful, and necessary constraints).

(b) Using Excel Solver, solve this problem by determining the values of d and t that minimize the cost. Note that H=275cm, P=2000kg, E=900,000kg/cm^2, d_1=1cm, d_2=10cm, t_1=0.1cm, and t_2=1cm. The yield stress of the material is 550kg/cm^2. The density of the material is 0.0025 kg/cm^3. Submit both your mathematical model and your software solution file. What are the resulting thickness and diameter values?

(c) Solve the same problem using MATLAB's Active Set algorithm. Use the same initial values you used in part (b). Provide all m-files including a main script as well as the m-files you may have written for the objective functions and constraints. You may have to study the following document and related examples: http://www.mathworks.com/help/toolbox/optim/ug/brnoxzl.html

(d) Solve the same problem using MATLAB's SQP algorithm. Use the same initial values you used in part (b). Report your results similar to the way you did in part (c).

(e) Compare the solutions you obtained using Excel solver, MATLAB Active-set and MATLAB SQP methods. Do the results agree? Keeping everything else the same, choose a different set of initial values and re-run the three algorithms. Report whether the final solution is consistent among the three algorithms.

Genetic Algorithms and Minimization

4. *(20 points)*

Compute the minimum of:

$f(x, y) = \sin(x)\sin(2y) + 0.3x$ where $-6 \le x \le 3$ and $-1.5 \le y \le 0$.

a) Visualize the function as a surface in 3D.
b) Find all the stationary points in the defined domain. For each such point, determine if it is a local minimum or maximum. Compute the global minimum.
c) Use a genetic algorithm to solve the same minimization problem. You can use a variation of the GA you implemented in the first problem. In your report clearly state:
   - How you designed your genome
   - How was the initial population generated
   - What is your fitness function
   - Implement different cases: (a) crossover, no mutation (b) mutation, no crossover (c) both. Compare the performance (ability to find the global minimum, number of iterations it takes to get to the minimum etc.) across the three cases
   - Compare the performance 4% elitism vs. no elitism using the criteria you used for crossover/mutation
   - Plot the population at various stages of the algorithm (e.g. initialization, after 5, 10, 100 iterations.