Scientific Programming Assignment 3

MPhil in Computational Biology

November 25, 2017

If there are errors found, I will update the assignment on the web at `http://github.com/sje30/rpc2017`

Due date: 2018-01-16 23:45

Please submit your report to Moodle as a PDF: NO OTHER FORMATS ARE ACCEPTED. Name your file `spa3_XXX.pdf`, where XXX is your three digit unique code, to come from the graduate office. (For example, I would save my file as `spa3_987.pdf` if my code were 987.) Please also write your code instead of your name on the front page.

Your report must be a maximum of ten pages, excluding the appendix. (List your code in the appendix.) This course work will consist of 30% towards your overall mark for this module.

Only R packages that come installed by default with a R installation can used for this assignment. If in doubt, check with Stephen whether you can use a particular package.

Include in your report an appendix containing your R code.

1 Travelling salesman problem [15 marks]

Implement the Elastic Net model (Durbin and Willshaw, 1986) and solve the travelling salesman problem (TSP). Demonstrate your algorithms on tours where the optimal solution is known (`http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/`). How good is the Elastic Net at solving the TSP?

Durbin R, Willshaw D (1987) An analogue approach to the travelling salesman problem using an elastic net method. Nature 326:689691. `http://dx.doi.org/10.1038/326689a0`.

2 Particle Swarm Optimization [15 marks]

Solve problems 1 to 3 (not problem 4) below. Ensure that you document your parameter settings and state how you met the constraints imposed on input values. Compare your answers to those found by R's optim family of functions. (This problem taken from `http://issc.uj.ac.za/downloads/problems/SCandPP.pdf`.)

Chapter 14

Particle Swarm Optimization

Problem 1. Particle Swarm Optimization (PSO) is based on the behavior of a colony or swarm of insects, such as ants, termites, bees, and wasps; a flock of birds; or a school of fish. The particle swarm optimization algorithm mimics the behavior of these social organisms. The word particle denotes, for example, a bee in a colony or a bird in a flock. Each individual or particle in a swarm behaves in a distributed way using its own intelligence and the collective or group intelligence of the swarm. As such, if one particle discovers a good path to food, the rest of the swarm will also be able to follow the good path instantly even if their location is far away in the swarm. Optimization methods based on swarm intelligence are called behaviorally inspired algorithms as opposed to the genetic algorithms, which are called evolution-based procedures. The PSO algorithm was originally proposed by Kennedy and Eberhart in 1995.

In the context of multivariable optimization, the swarm is assumed to be of specified or fixed size with each particle located initially at random locations in the multidimensional design space. Each particle is assumed to have two characteristics: a position and a velocity. Each particle wanders around in the design space and remembers the best position (objective function value) it has discovered. The particles communicate information or good positions to each other and adjust their individual positions and velocities based on the information received on the good positions.
The PSO is developed based on the following model:

1. When one particle locates an extremum point of the objective function, it

instantaneously transmits the information to all other particules.
2. All other particles gravitate to the extremum point of the objective function,

but not directly.
3. There is a component of each particle's own independent thinking as well as

its past memory.
Thus the model simulates a random search in the design space for the extremum

points of the objective function. As such, gradually over many iterations, the particles go to the target (the extremum point of the objective function).
The algorithm for determining the maximum of a function $f(\mathbf{x})$ (with $\mathbf{x}$ an n

dimensional vector) is as follows:

1) Initialize the number of particles N, the search intervals for each dimension (a_i, b_i), $i = 1, \ldots, n$ (n being the dimension of the search space), the search precision for each dimension ϵ_i, the maximum number of iterations i_{max}.
Initialize the positions of the particles $\mathbf{x}_j(0) = rand()$, $j = 1, \ldots, N$ randomly in the search domain.
Initialize the speeds of the particles $\mathbf{v}_j(0) = 0$, $j = 1, \ldots, N$.
Initialize the individual best positions of the particles $\mathbf{x}_{best,j}(0) = \mathbf{x}_j(0)$, $j = 1, \ldots, N$.
Initialize the iteration count $k = 0$.

2) Check the stop conditions: the diameter of the swarm in each dimension is less then the dimension's precision ϵ_i, or the maximum number of iterations i_{max} was reached. If yes then terminate else continue to step 3).

3) Calculate the values of the function $f(\mathbf{x})$ in the current positions of the particles, $f(\mathbf{x}_j(k))$, $j = 1, \ldots, N$. Update the values of the best individual points for each particle $\mathbf{x}_{best,j}(k)$, $j = 1, \ldots, N$ and the value of the global best point $\mathbf{x}_{best}(k)$. Continue to 4).

4) Update the particles' speeds by applying the formula:

$$\mathbf{v}_j(k) = \theta(k)\mathbf{v}_j(k-1) + c_1 r_1 \left[\mathbf{x}_{best,j}(k) - \mathbf{x}_j(k-1)\right] + c_2 r_2 \left[\mathbf{x}_{best}(k) - \mathbf{x}_j(k-1)\right]$$

where $j = 1, \ldots, N$ and r_1 and r_2 are random number between 0 and 1. The parameters c_1 and c_2 have usually the value 2 so that the particles would overfly the target about half of the time. $\theta(k)$ is the inertia weight dependent of the iteration count according to the formula:

$$\theta(k) = \theta_{max} - \left(\frac{\theta_{max} - \theta_{min}}{i_{max}}\right) k$$

where θ_{max} is a maximum value and θ_{min} is a minimum value, typically $\theta_{max} = 0.9$ and $\theta_{min} = 0.4$. Continue to 5).
5) Update the particles' positions by applying the formula:

$$\mathbf{x}_j(k) = \mathbf{x}_j(k-1) + \mathbf{v}_j(k), \quad j = 1, \ldots, N$$

6) Increment the iteration count k and go to 2).

Determine the maximum of the function

$$f(x) = -x^2 + 2x + 11, \;\; -2 \leq x \leq 2$$

by applying the PSO (Particle Swarm Optimization) method. The required precision is 10^{-4}.

Problem 2. Minimize

$$f(x_1, x_2) = x_1^2 + x_2^2 - 2x_1 - 4x_2$$

subject to the constraints

$$x_1 + 4x_2 - 5 \leq 0, \quad 2x_1 + 3x_2 - 6 \leq 0, \quad x_1 \geq 0, \quad x_2 \geq 0$$

by applying the Particle Swarm Optimization (PSO) method. The required precision is 10^{-3}.

Problem 3. Find the global minimum of the De Jong's function (or sphere model)

$$f(\mathbf{x}) = \sum_{i=1}^{n} \mathbf{x}_i^2, \;\; n \geq 2$$

by applying the Particle Swarm Optimization (PSO) method. The required precision is 10^{-3}.

Problem 4. Differntial evolution (DE) is a population-based optimization method that attacks the starting point problem by sampling the objective function at multiple, randomly chosen initial points. At initialization a vector population of dimension N_p is generated such that the allowed parameter region is entirely covered. Each vector is indexed with a number from 0 to $N_p - 1$ for bookkeeping because each of them has to enter a competition. Like other Evolutionary Strategy population-based methods, DE generates new points that are perturbations of existing points, but unlike other Evolutionary Strategy methods, DE perturbs population vectors with the scaled difference of two randomly