CS4744 FINAL EXAMINATION, Fall 2005

Due December 21, 2004, before 3:00 pm

1. This is a take-home final examination.
2. You are not allowed to discuss with any one except myself concerning any aspect of any of the problems.
3. Submit a zip file containing a copy of your programming files.

Problem 1 [100 pts]

This problem is from chapter 8 in Moler's book *Numerical Computing with Matlab* which is available at the Mathworks.

Analyze the chirp sound sample from the Matlab demos directory. By ignoring a short portion at the end, it is possible to segment the signal into eight pieces of equal length, each containing one chirp. Plot the magnitude of the FFT of each segment. Use `subplot(4,2,k)` for `k = 1:8` and the same axis scaling for all subplots. Frequencies in the range from roughly 400 Hz to 800 Hz are appropriate. You should notice that one or two of the chirps have distinctive plots. If you listen carefully, you should be able to hear the different sounds.

Problem 2 [100 pts]

A recent paper by H.-Y. Fan and J. Lampinen [”A Trigonometric Mutation Operator to Differential Evolution”, J. Global Optimization, Vol. 27, p. 105-129 (2003)] introduced a new operator to be embedded into the conventional differential evolution algorithm. The operator, described below, provides an alternate way of generating potential offsprings for the next generation.

For every individual i in the current population, three other individuals, i_1, i_2, and i_3, are randomly chosen to be the parents of a potential offspring, $\mathbf{v}^{(i)}$. Indices i, i_1, i_2, and i_3 are to be distinct from each other. According to the above article, the offspring is created by

$$\begin{aligned}\mathbf{v}^{(i)} \quad = \quad & \frac{1}{3}(\mathbf{x}^{(i_1)} + \mathbf{x}^{(i_2)} + \mathbf{x}^{(i_3)}) + (p^{(i_2)} - p^{(i_1)})(\mathbf{x}^{(i_1)} - \mathbf{x}^{(i_2)}) \\ & + (p^{(i_3)} - p^{(i_2)})(\mathbf{x}^{(i_2)} - \mathbf{x}^{(i_3)}) + (p^{(i_1)} - p^{(i_3)})(\mathbf{x}^{(i_3)} - \mathbf{x}^{(i_1)}),\end{aligned}$$

where

$$p^{(i_k)} = \frac{|f(\mathbf{x}^{(i_k)})|}{\sum_{m=1}^{3} |f(\mathbf{x}^{(i_m)})|},$$

for $k = 1, 2, 3$.

The differences in the $p's$ basically are replacing the scale factor s used in the conventional DE/rand/1/bin scheme. These new scales are related to the relative fitnesses of the three parents. The potential offspring is closer to the best parent. This operator is therefore rather greedy and must be used sparingly.

This operator is embedded in the conventional DE as follows:

(1) Initialize the population.

(2) For each individual in the population, generate a potential offspring

   (a) using the above trigonometric operator with probability M_t, or

   (b) using the conventional DE/rand/1/bin operator with probability $1 - M_t$.

(3) Perform a crossover operation.

(4) Select between the original individual, i, and its potential offspring.

(5) Repeat steps 2-4.

This modified DE algorithm now involves a total of 4 DE parameters: the population size n_p, the scale factor for the difference vectors s in the conventional DE/rand/1/bin scheme, the crossover parameter c, and the probability of the trigonometric operator M_t. Since the trigonometric operator is a rather greedy one, it is clear that M_t cannot be too large, otherwise we run into the problem of premature convergence to local minimas.

Your task is to embed the above trigonometric operator into the conventional DE/rand/1/bin scheme by modifying the Matlab program that is posted on the course website for DE. You need to find the best set of DE parameters that enables the modified DE to converge consistently to the correct solution for the 10-parameter Griewangk function, and at the same time requires the fewest number of function evaluations.

For the conventional n-dimensional Griewangk function, the global minimum is located at the origin, and the global minimum value is 0. It is not uncommon that mistakes in the program logic, or mistakes in the program itself, can cause the solution to converge to the zero vector, and to have a function value of zero there. To prevent that from happening modify the conventional Griewangk function so that the global minimum is now located at $[1, 2, ..., n]^T$, and the global minimum values is n.