SUMMARY through Week 4

This is meant to serve as a summary of the topics we've looked at in the modeling course up through Week 4. In this first section of the course, we have introduced modeling by discrete dynamical systems, and have spent quite a bit of time getting familiar with Matlab.

To further get to know Matlab, we looked at Reinforcement Learning (via the $n-$armed bandit) and genetic algorithms. Further considering the genetic algorithms as methods for optimization, we then started to look at classical optimization using Calculus.

Discrete Dynamical Systems

We began by looking at discrete dynamical systems (or difference equations):

$$x_{i+1} = f(x_i)$$

- Important vocabulary: Orbit, closed form (of the solution), fixed points, points of order k (for example, period two points).
- We should be able to give the solution to the general first order equation, $x_{n+1} = ax_n + b$.
- Give a particular solution to $x_{n+1} = ax_n + f(n)$ when $f(n)$ is a polynomial of degree k (Guess a full polynomial of degree k, like the Method of Undetermined Coefficients in Math 244).
- Be able to perform a graphical analysis of the orbit using a cobweb diagram.
- Use the information about first order equations to get a model for the discrete time compounding of interest.

The n-armed Bandit

For the $n-$armed bandit problem, we should be able to:

- Explain what the problem is (there are some competing goals).
- Explain (in words) the 4 strategies: Greedy, $\epsilon-$greedy, Softmax, Pursuit (or Win-Stay, Lose-Shift).
- Know the update rules for constructing the probabilities in the softmax and pursuit algorithms.
- For softmax, explain the role of τ and be able to compute the limits as τ goes to zero and infinity.

We should understand that the training in this problem is designed to balance out two competing goals- We need to *explore* the space of possible payoffs, and we want to *exploit* our current estimate to maximize our rewards.

Genetic Algorithms

Key idea: In the genetic algorithm, we also must *explore* the domain space, and at the same time, *exploit* our current estimates of fitness to maximize (or minimize) the fitness function. This is what ties the $n-$armed bandit problem to the genetic algorithm.

You should know generally what a genetic algorithm is (we didn't have a specific definition). Genetic algorithms are typically defined by the problem statement and algorithm specification- List those four things. (Define the population/chromosomes, Define the fitness function, Define the selection process for mating, Define crossover and mutation for the construction of the children).

- We looked at two distinct types of population: Binary strings, and strings of real numbers.
- The fitness function is typically given as part of the problem statement.
- How to choose mates with some probability?
  - Use the current normalized fitness values as probabilities (only if they are all non-negative).
  - Use the "softmax": Divide by τ, exponentiate, normalize. Discuss the role of τ and be able to compute the limits using τ.
  - Use rank-ordering: If the best chromosome is in position 1 and the worst is in position N, then choose chromosome j with probability:

$$P_j = \frac{N-(j-1)}{\sum_{i=1}^{N} i}$$

- Crossover: With binary strings, we can just pick the site at which crossover occurs and recombine the parent strings. With a set of real numbers, we have two random selections: Which coordinate should be used, and the β to use. For example, if we choose the i^{th} coordinate, then we create new points where all the coordinates are the same except for the i^{th} one, which become:

$$\begin{array}{l} \text{Child 1 } X_i = (1-\beta)(X_i)_{\text{ma}} + \beta(X_i)_{\text{pa}} \\ \text{Child 2 } X_i = (1-\beta)(X_i)_{\text{pa}} + \beta(X_i)_{\text{ma}} \end{array}$$

  Then Mutation is typically done by randomly selecting a member of the population (and a random coordinate) and changing the value to be some random element of the domain.

Nonlinear Optimization

Root finding and optimizing are very closely related. That is, if you have a root finding algorithm (to solve $g(x) = 0$ for x), then you can use it to calculate critical points (solve $f'(x) = 0$ for x).

You should be able to write down the steps (using words) for the bisection method and Newton's method in one dimension.

For higher dimensional functions, be able to compute the gradient (gradients are for real-valued functions), the Hessian (for real-valued functions), and the Jacobian (for vector-valued functions).

Be able to compute the linearization of a function of the form $y = f(x_1, \ldots, x_n)$.

Understand how the one dimensional Newton's method transforms into the multidimensional Newton's method, and be able to compute one or two steps (you'll need to compute the inverse of a 2×2 matrix, or use Cramer's Rule to solve a system).

If we get that far, also be able to compute a step or two of the method of gradient descent, and be able to explain why the algorithm works.

Exercises

1. Newton's Method is a discrete dynamical system of the form $x_{i+1} = F(x_i)$. Let $g(x)$ be the function for which we're determining the root (using Newton's Method).
   (a) What is F (in terms of g)?
   (b) Show that the fixed point for F corresponds to the root(s) of g.
2. Find all fixed points for $x_{i+1} = F(x_i)$ and classify them as attracting, repelling or neutral:
   (a) $F(x) = \sqrt{x}$
   (b) $F(x) = x(1 - x)$
   (c) $F(x) = 1/x^2$
   (d) $F(x) = (2 - x)/10$
3. Use graphical analysis to describe the fate of all orbits, if $x_{i+1} = F(x_i)$, and
   (a) $F(x) = -2x + 1$
   (b) $F(x) = \sin(x)$
4. Determine the general solution:
   (a) $x_{n+1} = \frac{3}{2}x_n + 1$
   (b) $x_{n+1} = 5x_n + n^2$

(c) $x_{n+1} = \frac{1}{2}x_n + 4n^2 + 2n + 1$

5. Let $F(x) = |x - 2|$. Use graphical analysis to display a variety of orbits under F. If we were to ask about points of period 2, what might you say?

6. (Calculator) You currently have \$5000 in a savings account that pays 6% interest per year. Interest is compounded monthly, and you add \$200 per month to your account. How much money do you have after 5 years? What is the total interest earned?

7. (Calculator) Mary receives \$5000 as a graduation gift. She deposits the money is a savings account at an annual interest rate of 3%, and interest is compounded monthly.

   Mary plans on working for three years and during this time, she will be making equal monthly deposits.

   After the three years, Mary will withdraw \$1200 per month during the fourth year, \$1300 per month for her fifth, \$1400 per month for her sixth, and finally \$1500 per month for her seventh year.

   What must the monthly payment be so that the savings account balance is zero at the end of the seventh year? What is the total interest earned over the seven year period?

8. Short Answer: Say (in words) what the following Matlab commands do:

   (a) `x=8:-2:1`

   (b) `exp([-1, 0, 1, 2])`

   (c) `probs=exp(P)./sum(exp(P))` (for some vector P that was previously defined, and could have any real numbers as entries).

   (d) `z=linspace(1,8,150);`

   (e) `2.^[-1,0,1,2]`

   (f) `Q=[3 1 2 2 1]; idx=find(Q==max(Q));` (You only need to tell me what `idx` will be)

9. In Matlab, if X is a matrix, write the Matlab command that gives you:

   (a) The entire 3rd row.

   (b) The entire 4th column.

   (c) The odd rows, with columns 1 and 2.

10. Name several differences between a *script* and a *function* in Matlab.

11. What was the $n-$armed bandit problem? In particular, what were the two competing goals, and why were they "competing"?

12. For the $n-$armed bandit problem, what is *the greedy algorithm*? How is it modified to get *the $\epsilon-$greedy algorithm*? How did the curve that you produced depend on ϵ (and was that reasonable)?

13. What is the "softmax" action selection? In particular, how did we change a set of payouts Q_i to a set of probabilities, P_i?

14. Suppose $Q = [-0.5, 0, 0.5, 1.0]$. Use the softmax selection technique with $\tau = 0.1$ to compute the probabilities.

15. If $Q_1 < Q_2 < Q_3 < Q_4$ for 4 machines, how do the probabilities change (under softmax) as $\tau \to 0$? As $\tau \to 1$?

16. What is the win-stay, lose-shift (or pursuit) strategy? What are the update rules?

17. Suppose we play with three machines, and machine 3 is chosen and gives a big payout (enough to make $Q_t(3)$ the maximum). Update the probabilities for win-stay, lose-shift, if they are: $P_1 = 0.3, P_2 = 0.5, P_3 = 0.2$ and $\beta = 0.3$.

18. Suppose we have a genetic algorithm with 4 chromosomes, and current fitness values $[-1, 1, 2, 3]$. We want to construct the probability of choosing these chromosomes for mating. Calculate the probabilities (if possible) using the current ordering, if we use:
    (a) Normalized fitness values:
    (b) Rank order, normalized:
    (c) Softmax with $\tau = 1$.

19. Suppose we have a genetic algorithm using three dimensional real coordinates. Suppose the pair of coordinates for mating are given as $(1, 2, 1)$ and $(3, -1, 1)$. Calculate the children if we choose the middle coordinate and $\beta = 1/2$.

20. Let
$$f(x_1, x_2) = 3x_1x_2 + x_1^2$$
    (a) Linearize f about the point $(1, 1)$.
    (b) Compute the Hessian of f
    (c) Show that Newton's Method, started at $(1, 1)$, converges in one step.

21. Let
$$f(x, y) = 3x^2 + xy - y^2 + 3x - 5y$$
    (a) At the point $(1, 1)$, in which direction is f increasing the fastest? What is its rate of change in the fastest direction (Hint: Directional derivative)
    (b) Compute the Hessian of f.
    (c) Use the "second derivatives" test to determine if we have a local extremum at the critical point.