02-512/02-712 Computational Methods for Biological Modeling and Simulation
Lecturers: Russell Schwartz and Oana Carja
Tuesdays and Thursdays 3:05pm-4:25pm HH B131

Problem Set 2

Assigned 9/22/22, Due 10/6/22

Problem 1 (20 points): In this question, we will examine some variations on a hypothetical biological problem and consider how we might model them. We will suppose here that we are motivated by a problem in biomechanics of trying to characterize how cells respond to applied forces on a tissue. We imagine that we have a complex tissue containing many cells of a few different types and we apply various forces to the tissue and observe how the cells in the tissue deform. We will suppose that we want to measure the stiffness of cells, which we characterize by measuring cell length as a function of various applied forces. We conduct a set of experiments applying different forces to the tissue and use microscopy to measure lengths of various cells. Assume we have t cell types and n examples of each cell type. We have c types of force $f_1, \ldots, f_c$ (e.g., compressive, shear, torsional, etc.) and we assume the length of each cell or cell type is a function of these applied forces. Then we conduct a series of m experiments, where each experiment j assumes we apply a set of forces $f_{j1}, \ldots, f_{jc}$ and measure lengths of all the cells L_{jik} and where we will assume the applied forces are linearly independent between experiments.

For each proposed model, you should formalize the model by providing a list of output variables to be determined, a list of constraints (if any), and an objective function (if any). Then say whether the problem is an unconstrained optimization, a linear system, a linear program, or a non-linear program. If the description is ambiguous, add any details you need to fully define the problem. Finally, suggest an appropriate algorithm for solving it.

a. Assume we want a linear regression model representing how a cell length changes as a function of each type of applied force, where all cells of a given type have the same regression model. We will conduct a set of m experiments varying the applied forces where $m >> ct$.

b. Now assume we want to learn cell-specific linear regression models, where each individual cell, not each cell type, has its own regression fit. We conduct a series of m experiments where $m = c+1$.

c. Assume again we want to learn cell-specific linear regression models and conduct $m = c + 1$ experiments but we further have some prior knowledge giving a maximum and minimum possible value for each coefficient of each regression model. We will assume that we want to minimize the sum of square errors of the regression model for each cell subject to the known maxima and minima.

d. Now suppose we want to learn a quadratic regression model for each individual cell, where $m = c + 1$. Assume we no longer have maximum and minimum constraints on the regression coefficients.

Problem 2 (15 points): In this problem, we will examine how various zero-finding methods perform on a single system. In class, we saw an example of a molecular tether, where we imagine that we have a disordered protein domain that acts like a Hooke's law spring but it has a charged end that is pulled towards a charged surface on another protein. Here, we will look at a simpler one-dimensional version of that system. We will assume that the model has potential energy described

by the following function:

$$E(x) = \frac{1}{2}kx^2 - \kappa(1-x)^{1.5}$$

which would imply the force is given by the derivative

$$\frac{dE}{dx} = f(x) = kx - 1.5\kappa\sqrt{(1-x)}$$

For the purposes of this problem, we will make $k = 1$ and $\kappa = 2$.

a. Perform three steps of bisection search to solve for x from the starting interval [0,1], showing your work (i.e., the translation x and the net force at the endpoints and midpoint of each step). Each time you derive a new midpoint counts as one step. Provide your final estimate (the midpoint of the last interval) and the forward and maximum possible backward errors at that point.

b. Again assuming the compression must be between 0 and 1, perform three steps of secant method search, showing your work as in part a. Each time you derive a new midpoint counts as one step. Provide your final estimate (the midpoint of the last interval) and the forward and maximum possible backward errors at that point.

c. Perform three steps of Newton-Raphson search to solve for the point at which the forces balance, showing your work. We will assume an initial guess of $x = 0.75$. Each time you derive a new estimated x counts as one iteration. Provide your final estimate, its forward error, and an estimate of the backward error at that point.

Problem 3 (20 points): In this problem, we will use linear programming to try to optimize a treatment plan in a hypothetical medical setting. Let us suppose that we are trying to treat a cancer patient and we have two drugs, D_1 and D_2, that can potentially kill their tumor. Each drug carries risk of toxicity to various organs, though, so we need to limit the use of either to avoid killing the patient. We will suppose that we have some units to measure toxicity and the effectiveness at killing the tumor and that the two drugs have the following profiles:

drug	liver toxicity	kidney toxicity	brain toxicity	effectiveness
D_1	2	2	5	1
D_2	8	4	2	3

We will assume that the patient can tolerate 10 units of toxicity to any organ before the drugs kill them. We would like to choose the combination of drugs that maximizes effectiveness without killing the patient.

a. Set up a linear program for the problem of finding the combination of drugs D_1 and D_2 that maximizes effectiveness without killing the patient.

b. Convert your linear program to standard form.

c. Solve the linear program by the simplex method to determine the optimal amounts of D_1 and D_2. Show your work.

d. For the value at your final result, what is the total effectiveness and the total toxicity to each organ?

e. Suppose we had available to us a supplemental drug that would protect the liver, allowing the patient to tolerate a higher level of liver toxicity. Would we expect that using that drug would allow us to adjust the doses of D_1 and D_2 to increase the total effectiveness? What if it was a drug that protects the liver? What if it was a drug that protects the brain?

Problem 4 (25 points): In this programming problem, we will apply continuous optimization for a problem in non-linear regression. Let us suppose we are studying the activity of a particular kind of neuron when it fires. We have a simplified model of the neuron in which we believe its behavior is described by a peak potential A when it first fires followed by an exponential decay at a rate λ, giving an overall potential curve of the form:

$$V(t) = Ae^{-kt}$$

We will suppose that our goal here is to learn the parameters A and k from many examples of the neuron at different times after firing, yielding a set of data points of the form: (t_1, V_1), (t_2, V_2), ..., (t_n, V_n). We would expect the data to be noisy so we cannot directly measure the coefficients but will instead seek to find them so as to minimize a least-squares objective function:

$$\min_{A,k} \sum_{i=1}^{n} \left(Ae^{-kt_i} - V_i\right)^2$$

a. Find the gradient of the objective function.

b. For a Newton-Raphson solver, we will also need the Hessian. Let us assume the Hessian is too complicated to derive analytically. Instead, derive numerical approximations for the terms of the Hessian using the analytical first partial derivatives you already determined. That is, express the elements of the Hessian as functions of $\frac{\partial F(A,k)}{\partial A}$ and $\frac{\partial F(A,k)}{\partial k}$ with perturbations ΔA and Δk.

c. Provide pseudocode for a Newton-Raphson solver for this problem. Assume you are given as input $(t_1, f_1), (t_1, f_1), \ldots, (t_n, f_n)$; initial guesses A_0 and k_0; and perturbations ΔA and Δk. You can assume the method will run for a fixed number of rounds, r. You can also assume that you have a subroutine for solving 2X2 linear systems, so you do not need to explain in detail how that is done in your pseudocode.

d. Provide code implementing your solver. Your code should read as input a file containing initial guesses A_0 and k_0; perturbations ΔA and Δk; number of optimization rounds r; number of data points n; and the data points $(t_1, V_1), (t_2, V_2), \ldots, (t_n, V_n)$;. Your code should return the inferred values A and k. If you are using Matlab, you can assume the input will be given as a series of variables A_0, k_0, ΔA, Δk, r, $\vec{t}$, and $\vec{V}$. Otherwise, assume these are given in a file of the form:

A_0
k_0
ΔA
Δk
r
n
t_1 V_1
$\vdots$
t_n V_n

(Hint: You can solve a 2x2 linear system $\begin{bmatrix} a & b \\ c & d \end{bmatrix} \vec{x} = \vec{b}$ by the formula $\vec{x} = \frac{1}{ad-bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix} \vec{b}$.)

e. A sample data file is provided on Blackboard. Turn in the output of your code for that file.

***Problem 5 (20 points):** In this problem, we will consider the issue of how we should select step sizes for finite difference approximations. We will specifically look at one practical concern in estimating a derivative by a finite difference measure: the trade-off between round-off errors, which arise from using finite precision arithmetic, and truncation errors, which arise from using numerical approximations in our methods. As a general rule, making the step size too large will lead to large truncation errors, while making it too small will lead to large round-off errors. The question then becomes how to find a reasonable balance between these sources of error.

We will consider this problem in the context of the second order centered difference approximation

$$f'(x) \approx \frac{f(x+\Delta x) - f(x-\Delta x)}{2\Delta x}$$

Taylor series tell us that the error in this approximation is $O(\Delta x^2)$. That is an example of a truncation error, which would occur even if our computer had infinitely precise arithmetic. We would like Δx^2 to be small to minimize that error. However, floating point numbers have a finite precision in any real computer, meaning that any real number x will actually be stored as $x(1+\varepsilon)$, where ε is some unknown error bounded by the precision of the computer. For example, the standard for a double-precision floating point number would have $|\varepsilon| < 2^{-52} \approx 2 \times 10^{-16}$. If Δx is too small then $f(x+\Delta x)$ will be too close to $f(x-\Delta x)$, so the roundoff error will dominate the computation.

a. Suppose we know an upper-bound C_3 on the absolute value of the second derivative of $f(x)$ (i.e., $|f'''(x)| \leq C_3$). Calculate the largest possible truncation error for the forward difference approximation as a function of Δx and C_3.

b. Now assume we have an upper bound C_0 on the absolute value of $f(x)$ (i.e., $|f(x)| \leq C_0$). Find an approximate upper bound on the possible roundoff error for the centered difference approximation as a function of Δx, ε, and C_0. Assume you can evaluate $f(x+\Delta x)$ and $f(x-\Delta x)$ exactly but can only represent them with the precision of the machine you are using. You can assume Δx and ε are small enough that only the lowest-order term in each is relevant in any expansion.

c. We can get reasonably close to an optimal choice of Δx by finding a Δx that makes truncation error and roundoff error approximately equal. Assuming your bounds derived in parts a and b were actually exact values of the two errors, find the best choice of Δx.

d. In practice, we are not likely to have good bounds on the derivatives for a function we are trying to evaluate. To get a rough estimate of the magnitude of the optimal Δx, we might assume C_0 and C_3 are on the order of 1 if we do not have any better estimates. If we assume $C_0 = C_3 = 1$ and $\varepsilon = 2^{-52}$ then what would be a reasonable choice of Δx to use by your approximation?

e. Repeat your analysis from parts a and b for centered difference second derivative approximation

$$\frac{f(x+\Delta x) + f(x-\Delta x) - 2f(x)}{\Delta x^2} \approx f''(x)$$

to derive some guidelines for setting Δx for that formula. You may define whatever bounds you

need on derivatives and assume Δx and ε are small enough that only the lowest-order terms in each are relevant in any expansion. Will your optimal Δx generally be larger or smaller for the second derivative versus the first derivative approximation?

*Recall that the starred problems are only for the 02-712 students.