Computational Science - Assignment 2

January 16, 2025

1 Iteration And Recursion

1.1 Question 1

A) For the given function $F(x) = (x-1)\cos(x) + \frac{1}{2}$, we can use the Intermediate Value Theorem to find the domain. After examining the graph, we observe that the function is continuous everywhere, and values 0 and 1 will be suitable for the problem.

Setting this equal to zero, we can solve for x to find the critical points. These critical points, along with the endpoints of the domain, will divide the domain into intervals. We can then test each interval by choosing a test point in the interval and substituting it into the equation.

$$f(0) = (0-1)\cos(0) + \tfrac{1}{2} = \tfrac{-1}{2} < 0$$

Now evaluating by setting one in function, we get

$$f(1) = (1-1)\cos(1) + \tfrac{1}{2} = \tfrac{1}{2} > 0$$

Now finding the derivative of the function we get

$$f'(x) = \cos(x) - (x-1)\sin(x)$$

Looking at the derivative, we can say that $\sin(x)$ and $\cos(x)$ are always between -1 and 1. The root occurs when $(x-1) = 0$, which gives $x = 1$. As the cosine term does not affect the sign change, we choose the interval (0,1). Looking at the results, we can say that $f'(x)$ is always increasing on this interval. The interval $(-\infty, 1)$ contains precisely one root.

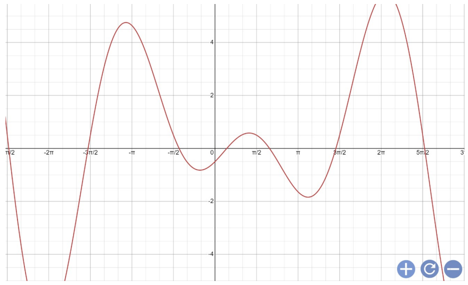

Figure 1: Caption for Figure 1

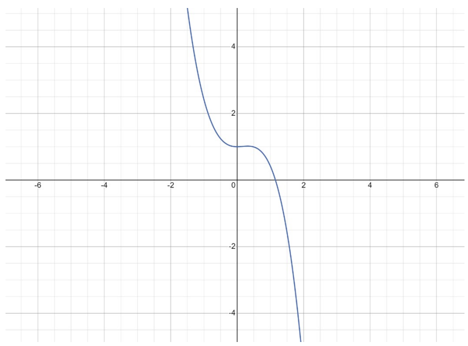

Figure 2: Caption for Figure 2

B) For the second equation $f(x) = \sqrt{x^2+1} - x^3$, we will follow the same method. Also, the function is continuous everywhere. So the derivative of the function is

$$f'(x) = \frac{x}{\sqrt{x^2+1}} - 3x^2$$

As x approaches to $(-/+\infty)$, $f(x)$ approaches to (∞). Examining the signs at $x = 0$ and $x = 1$:

$$f(0) = \sqrt{1} - 0 > 0$$
$$f(1) = \sqrt{2} - 1 > 0$$

By Intermediate Value Theorem, since $f(0) > 0$ and $f(1) > 0$, we can say that there exists one root in the interval.

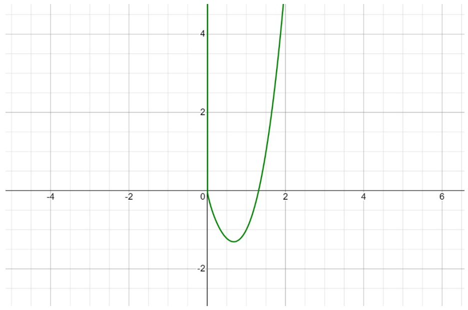

Figure 3: Caption for Figure 3

C) By looking at the third function's graph, we can say it is continuous on $(0, \infty)$. As x approaches $(+0)$ or (∞), $f(x)$ also goes to $(+\infty)$. Taking the derivative of the function we get,

$$f'(x) = 2x + \ln(x) - 1$$

Now if we examine $f(x)$ at $x = 1$ and $x = 2$ we get,

$$f(1) = 1\ln(1) - 2.1 + (1^2) + 1 = -1 < 0$$
$$f(2) = 2\ln(2) - 4 + 4 + 1 = 2\ln(2) + 1 > 0$$

Since, $f(1) < 0$ and $f(2) > 0$, there exists at least one root in the interval $(1, 2)$.

D) The function $f(x)$ is continuous everywhere except $(x = -1)$ and $(x = -2)$. Finding the derivative of the function we get,

$$f'(x) = \frac{-2((5x^3)+(18x^2)+24x+12)}{((x+1)^3)(x+2)^3}$$

As x approaches $-\infty$ or -2^-, $f(x)$ goes to $+\infty$. As x approaches -2 or -1^-, $f(x)$ goes to $-\infty$. As x approaches -1^+, $f(x)$ goes to $+\infty$.

So, if we examine it at $x = -3$ and $x = 0$; we get,

$$f(-3) = \frac{1}{4} - 0 - 1 < 0$$
$$f(0) = 1 - 1 - 1 < 0$$

Since, $f(-3) < 0$ and $f(0) < 0$, by IVT, there exists at least one root in the interval $(-3, 0)$.

E) The function is also continuous everywhere. As x approaches $-\infty$ or $+\infty$, $f(x)$ approaches $\frac{-1}{10}$. Taking the derivative of the function, we get,

$$f'(x) = e((-3x^2) + 1)$$

Examining the values at $x = 0$ and $x = 2$, we get,

$$f(0) = e((-2) - (\frac{1}{10}) < 0)$$
$$f(2) = e((-2) - (\frac{1}{10}) < 0)$$

By IVT, since $f(0) < 0$ and $f(2) < 0$, there exists at least one root in the interval $(0, 2)$.

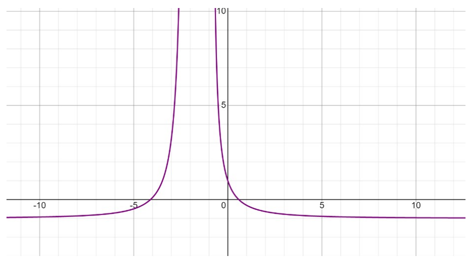

Figure 4: Caption for Figure 4

Figure 5: Caption for Figure 5

Linear System And Complexity

January 16, 2025

Question 1

To check whether it holds the inequality, we have to find certain values. First, we will find the relative error. The relative error is given by:

$$\text{Relative Error} = \frac{\|\mathbf{x} - \mathbf{x}'\|}{\|\mathbf{x}\|}$$

where $\mathbf{x}$ is the exact solution and $\mathbf{x}'$ is the approximate solution. Calculating $v = [v_1, v_2]$ as a vector norm: $\sqrt{{v_1}^2 + {v_2}^2}$.

We get,

$$\frac{\sqrt{(3 - 3.00712)^2 + (1 - 1.00102)^2}}{\sqrt{3^2 + 1^2}}$$

Solving it, we get a value equal to 0.00237.

Now we will find the condition number, which is the product of the norm of matrix A and the norm of the inverse of matrix A.

$$A = \begin{bmatrix} 3 & -21 \\ -4 & 27 \end{bmatrix}$$

Norm of matrix $A = \sqrt{a^2 + b^2 + c^2 + d^2}$.

Norm of inverse of matrix $A = \frac{1}{ad-bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$.

Putting the values,

$$\sqrt{3^2 + (-21)^2 + (-4)^2 + 27^2} \cdot \sqrt{\left(\frac{27}{-3 \cdot 27 - 4 \cdot (-21)}\right)^2 + \left(\frac{-21}{-3 \cdot 27 - 4 \cdot (-21)}\right)^2 + \left(\frac{-4}{-3 \cdot 27 - 4 \cdot (-21)}\right)$$

We get the value as

$$3^2 + (-21)^2 + (-4)^2 + 27^2 = 9 + 441 + 16 + 729 = 1195$$

$$\sqrt{1195} \times \sqrt{\frac{1195}{9}} = \sqrt{1195} \times \frac{\sqrt{739}}{3} \approx 1.0806$$

For the last step, we have to find the relative residual. Let's denote $b = [-12, 15]$ and A and x' are as defined above. The matrix-vector product of

$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$ and $v = [v1, v2]$ is $[a * v1 + b * v2, c * v1 + d * v2]$. So, the relative residual is:

$$\frac{\sqrt{(-12 - (3 * 3.00712 - 21 * 1.00102))^2 + (15 - (-4 * 3.00712 + 27 * 1.00102))^2}}{\sqrt{(-12)^2 + 15^2}}$$

After calculating we get, 0.00219. Now we check if it holds the inequality:

$$\text{Relative error of } x' \leq \text{ Condition number of } A \times \text{ Relative residual of } x'$$

$$0.00237 \leq 1.0806 \times 0.00219$$

$$0.00237 \leq 0.00237$$

So we can see that they are equal to each other. So they hold the following inequality.

Question 2

a) To compute LU decomposition of matrix A using partial pivoting, we will do row operations to transform A into an upper triangular matrix U while keeping track of the row swaps in the permutation matrix P and the multipliers in the lower triangular matrix L.

Given matrix A:

$$A = \begin{bmatrix} 1 & -2 & 5 \\ 3 & 6 & 1 \\ -2 & -2 & 2 \end{bmatrix}$$

We start with:

$$P = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad L = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad U = \begin{bmatrix} 1 & -2 & 5 \\ 3 & 6 & 1 \\ -2 & -2 & 2 \end{bmatrix}$$

Step 1: Swap rows 1 and 2 to place the largest pivot element (3) in the (1,1) position. Update P accordingly.

$$P = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad L = \begin{bmatrix} 1 & 0 & 0 \\ \frac{1}{3} & 1 & 0 \\ -\frac{2}{3} & 0 & 1 \end{bmatrix}, \quad U = \begin{bmatrix} 3 & 6 & 1 \\ 1 & -2 & 5 \\ -2 & -2 & 2 \end{bmatrix}$$

Step 2: Use $m_{21} = \frac{a_{21}}{a_{11}} = \frac{1}{3}$ to eliminate a_{21} from row 2. Update L and U accordingly.

$$P = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad L = \begin{bmatrix} 1 & 0 & 0 \\ \frac{1}{3} & 1 & 0 \\ -\frac{2}{3} & 0 & 1 \end{bmatrix}, \quad U = \begin{bmatrix} 3 & 6 & 1 \\ 0 & -4 & \frac{14}{3} \\ -2 & -2 & 2 \end{bmatrix}$$

Step 3: Use $m_{31} = \frac{a_{31}}{a_{11}} = \frac{-2}{3}$ to eliminate a_{31} from row 3. Update L and U accordingly.

$$P = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad L = \begin{bmatrix} 1 & 0 & 0 \\ \frac{1}{3} & 1 & 0 \\ -\frac{2}{3} & 0 & 1 \end{bmatrix}, \quad U = \begin{bmatrix} 3 & 6 & 1 \\ 0 & -4 & \frac{14}{3} \\ 0 & 2 & \frac{8}{3} \end{bmatrix}$$

The decomposition is complete:

$$P = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad L = \begin{bmatrix} 1 & 0 & 0 \\ \frac{1}{3} & 1 & 0 \\ -\frac{2}{3} & 0 & 1 \end{bmatrix}, \quad U = \begin{bmatrix} 3 & 6 & 1 \\ 0 & -4 & \frac{14}{3} \\ 0 & 2 & \frac{8}{3} \end{bmatrix}$$

b) And for the Ax=R: For $Ly = Pb$ for y where $pb = R$, then solve $Ux = y$ for x:

$$\begin{aligned} y_1 &= 2 \\ y_2 - \frac{1}{3}y_1 &= -3 \\ -\frac{2}{3}y_1 + y_3 &= 0 \end{aligned}$$

This gives us $y_1 = 2$, $y_2 = -\frac{5}{3}$, $y_3 = \frac{4}{3}$.

For $Ux = y$:

$$\begin{aligned} 3x_1 + 6x_2 + x_3 &= 2 \\ -4x_2 + \frac{14}{3}x_3 &= -\frac{5}{3} \\ 2x_2 + \frac{8}{3}x_3 &= \frac{4}{3} \end{aligned}$$

This gives us $x_1 = -\frac{5}{6}$, $x_2 = \frac{1}{3}$, $x_3 = \frac{1}{2}$.

Therefore, the solution to $Ax = R$ is:

$$x = \begin{bmatrix} -\frac{5}{6} \\ \frac{1}{3} \\ \frac{1}{2} \end{bmatrix}$$

Question 3

Algorithm 1 Compute Matrix-Vector Product

Input: vector $x \in \mathbb{R}^n$; vectors $a, b, c \in \mathbb{R}^n$ that define matrix A according to definition (1).
Output: vector $y \in \mathbb{R}^n$ such that $y = Ax$
Initialize the output vector y with zeros: $y = [0] * n$
Compute the first element of y: $y[0] = a[0] * x[0] + c[0] * x[1]$
for $i = 1$ to $n - 1$ **do**
    Compute the middle elements of y:
        $y[i] = b[i] * x[i-1] + a[i] * x[i] + c[i] * x[i+1]$
end for
Compute the last element of y: $y[n-1] = b[n-1]*x[n-2]+a[n-1]*x[n-1]$

a) The pseudo-code provided computes the matrix-vector product $y = Ax$ for a matrix A defined by vectors a, b, c and a vector x. The algorithm initializes the output vector y with zeros and then computes each element of y based on the formula.

b) To analyze the complexity, we count the number of floating-point operations (flops) required. We have one addition and one multiplication for the first element of y, which is two flops. We have 3 multiplications and 2 additions for each of the $n-2$ middle parts of y, which is 5 flops for each, for a total of $5\times(n-2)$ flops. There are two multiplications and one addition for the final element of y, or three flops. So, the total number of flops is $2 + 5 \times (n-2) + 3 = 5n - 7$. Since we only take into account the highest order term and disregard the constants while using "Big-Oh" notation, the algorithm's asymptotic behavior remains $O(n)$.

Compared to standard matrix-vector multiplication for a dense matrix, which requires multiplying each matrix element by each vector element, this method requires significantly fewer $O(n^2)$ flops. Because there are many zeros in the provided matrix A, the method minimizes the amount of flops by avoiding pointless multiplications and additions involving these zeros. For big n, this improves the algorithm's efficiency.

d) The graph is titled as Comparison of wall times for different matrix-vector multiplication methods. X- axis represents size of matrix, while Y axis represents time taken to perform the operation. Looking at the black line, which represents circ_prod, we can see that the wall time for this method remains relatively constant as n increases. Similarly, the built-in method (blue) also shows that the wall time remains relatively the same as n increases.

The red line, which represents n, shows that the wall time increases for this method, while n^2 shows a similar trend to n. The 'circ_prod' and 'built-in' methods appear to be more efficient as the size of the matrix/vector increases, compared to the n and n^2 methods. This could be important in scenarios where large matrix/vector multiplications are required, and computational efficiency is a concern.

Interpolation

January 16, 2025

Question 1

(a) The interpolating conditions that the polynomial interpolant $P_3(x)$ satisfies are:

$$P_3(x_k) = y_k \quad \text{for} \quad k = 0, 1, 2, 3$$

This means that the polynomial $P_3(x)$ passes through all the given data points.

(b) For interpolation data, we will construct linear system by Vandermonde matrix using x values and will solve for coefficients. The system can be shown as:

$$\begin{bmatrix} 1 & x_0 & x_0^2 & x_0^3 \\ 1 & x_1 & x_1^2 & x_1^3 \\ 1 & x_2 & x_2^2 & x_2^3 \\ 1 & x_3 & x_3^2 & x_3^3 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{bmatrix}$$

Substituting the values x_k and y_k into the matrix, we get:

$$\begin{bmatrix} 1 & -1.20 & (-1.20)^2 & (-1.20)^3 \\ 1 & 0.500 & (0.500)^2 & (0.500)^3 \\ 1 & 2.20 & (2.20)^2 & (2.20)^3 \\ 1 & 3.10 & (3.10)^2 & (3.10)^3 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} -0.400 \\ 1.20 \\ 5.50 \\ 10.2 \end{bmatrix}$$

(c) It involves solving the linear system for the interpolation coefficients. Solving the coefficients, we get:

$$a_0 = -0.4, \quad a_1 = 0.52333333, \quad a_2 = 1.56388889, \quad a_3 = 1.57083333$$

These coefficients represent the polynomial $P_3(x)$. We get this by constructing Vandermonde matrix using the x-values and then solve the linear system for the coefficients. The Vandermonde matrix is a matrix where each column is an increasing power of the input vector 'x'. Solving the system we get the polynomial coefficients.

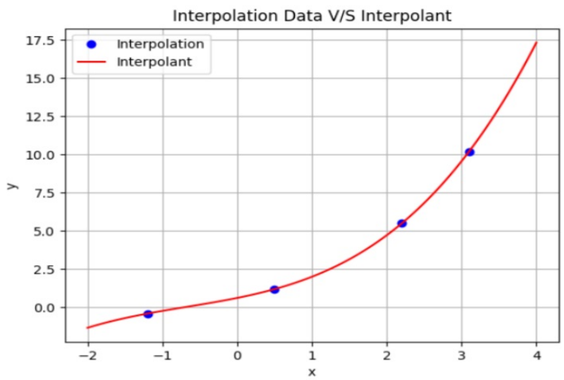

Figure 1: d part

Question 2

For the function $f(x)$, we start by finding the derivative $f'(x)$:

$$f'(x) = -\frac{1}{(x+1)^2}$$

The second derivative is $f''(x) = \frac{2}{(x+1)^3}$.

Setting $f'(x)$ equal to zero gives $x = -1$, which is not in the interval $[0, 1]$. So, we only need to consider the endpoints of the interval:

- At $x = 0$, $f(0) = -1$. - At $x = 1$, $f'(1) = -\frac{1}{4}$.

Therefore, the maximum value of the second derivative on the interval $[0, 1]$ is $M_f = 4$. Using the error bound theorem, we get:

$$|E_f(x)| \leq \frac{M_f}{4(N+1)}$$

For the function $g(x)$, the derivative is $g'(x) = -\exp(-x)$. The second derivative is $g''(x) = \exp(-x)$. The extrema occur at $x = 0$ and $x = 1$. The maximum value of the second derivative on the interval $[0, 1]$ is $M_g = 1$. Applying the error bound theorem, we get:

$$|E_g(x)| < \frac{M_g}{4(N+1)}$$

C) Based on the calculations in parts (a) and (b), we can infer that the function $f(x) = \frac{1}{x+1}$ produces interpolation data more amenable to polynomial interpolation compared to $g(x) = \exp(-x)$.

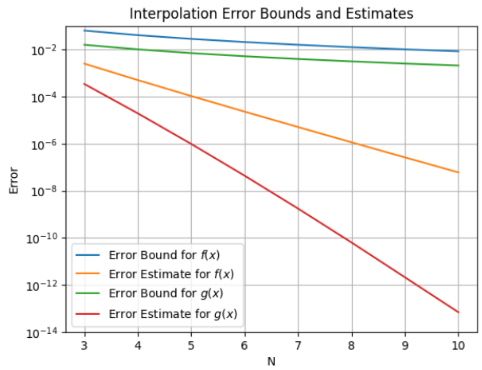

Figure 2: Enter Caption

For $f(x)$, the second derivative on the interval $[0,1]$ has a maximum value of $M_f = 4$, while for $g(x)$, the second derivative has a maximum value of $M_g = 1$. A larger value of M typically indicates that the function has a smoother behavior, making it more suitable for polynomial interpolation.

Additionally, the error bounds for $f(x)$ decrease faster with increasing N compared to $g(x)$, indicating that the interpolation errors for $f(x)$ are likely to be smaller for a given N.

Therefore, based on these observations, we can conclude that the function $f(x) = \frac{1}{x+1}$ produces interpolation data that is more amenable to polynomial interpolation.

Exponential least-squares, part I

January 16, 2025

A

The sum of squares of the errors, denoted by $E(c, \lambda)$, is a measure of how well the model $a(x) = ce^{\lambda x}$ fits the measured data points (x_j, y_j).

For each data point j, the error is calculated as the difference between the model's predicted value $a(x_j)$ and the actual measured value y_j. This is expressed as $a(x_j) - y_j$.

After squaring each of these individual mistakes, one can find the sum of squares of the errors for all data points between $j = 1$ and N (where N is the total number of data points). This is expressed mathematically as:

$$E(c, \lambda) = \sum_{j=1}^{N} (a(x_j) - y_j)^2$$

Substituting the model $a(x) = ce^{\lambda x}$ into the above equation gives:

$$E(c, \lambda) = \sum_{j=1}^{N} (ce^{\lambda x_j} - y_j)^2$$

The total difference between the data and the model's predictions is measured using this formula. Finding the values of c and λ that minimize this sum is the aim of the least-squares fit, which yields the best-fitting exponential curve to the data.

B

To find the values of c and λ that minimize or maximize the sum of squares of the errors $E(c, \lambda)$, we differentiate E with respect to c and λ and set the derivatives to zero. This gives us the system of equations that must be satisfied at the minimum or maximum of E.

The derivative of E with respect to c is:

$$\frac{\partial E}{\partial c} = 2 \sum_{j=1}^{N} (ce^{\lambda x_j} - y_j) e^{\lambda x_j}$$

Setting this derivative to zero gives:

$$2\sum_{j=1}^{N}(ce^{\lambda x_j} - y_j)e^{\lambda x_j} = 0$$

The derivative of E with respect to λ is:

$$\frac{\partial E}{\partial \lambda} = 2\sum_{j=1}^{N}(ce^{\lambda x_j} - y_j)cx_j e^{\lambda x_j}$$

Setting this derivative to zero gives:

$$2\sum_{j=1}^{N}(ce^{\lambda x_j} - y_j)cx_j e^{\lambda x_j} = 0$$

These two equations form a system of nonlinear equations that must be solved to find the values of c and λ that minimize or maximize E.

C

To compute the elements of the Jacobian matrix for the given system of equations, we need to find the second partial derivatives of the error function $E(c, \lambda)$ with respect to c and λ. Let's denote these derivatives as $\frac{\partial^2 E}{\partial c^2}$, $\frac{\partial^2 E}{\partial \lambda \partial c}$, $\frac{\partial^2 E}{\partial c \partial \lambda}$, and $\frac{\partial^2 E}{\partial \lambda^2}$, respectively.

Second partial derivative of E with respect to c:

$$\frac{\partial^2 E}{\partial c^2} = 2\sum_{j=1}^{N} e^{2\lambda x_j}$$

Mixed partial derivative of E with respect to λ and c:

$$\frac{\partial^2 E}{\partial \lambda \partial c} = 2\sum_{j=1}^{N} x_j e^{2\lambda x_j}$$

Mixed partial derivative of E with respect to c and λ (same as the previous one due to the symmetry of second derivatives):

$$\frac{\partial^2 E}{\partial c \partial \lambda} = 2\sum_{j=1}^{N} x_j e^{2\lambda x_j}$$

Second partial derivative of E with respect to λ:

$$\frac{\partial^2 E}{\partial \lambda^2} = 2\sum_{j=1}^{N} c^2 x_j^2 e^{2\lambda x_j}$$

Therefore, the Jacobian matrix is:

$$J = \begin{bmatrix} \frac{\partial^2 E}{\partial c^2} & \frac{\partial^2 E}{\partial \lambda \partial c} \\ \frac{\partial^2 E}{\partial c \partial \lambda} & \frac{\partial^2 E}{\partial \lambda^2} \end{bmatrix} = \begin{bmatrix} 2\sum_{j=1}^{N} e^{2\lambda x_j} & 2\sum_{j=1}^{N} x_j e^{2\lambda x_j} \\ 2\sum_{j=1}^{N} x_j e^{2\lambda x_j} & 2\sum_{j=1}^{N} c^2 x_j^2 e^{2\lambda x_j} \end{bmatrix}$$

This Jacobian matrix is used in the Newton-Raphson iteration method to iteratively find the values of c and λ that minimize or maximize the error function $E(c, \lambda)$.

D

To find initial values for c and λ using the first and last data points, we can set up the following equations:

$$c \cdot e^{\lambda \cdot x_1} = y_1$$

$$c \cdot e^{\lambda \cdot x_N} = y_N$$

where (x_1, y_1) and (x_N, y_N) are the first and last data points, respectively. Solving these equations simultaneously will give us initial estimates for c and λ. Let's denote these initial estimates as c_0 and λ_0, respectively. Solving them we get, First equation:

$$c \cdot e^{\lambda \cdot x_1} = y_1$$

$$c = \frac{y_1}{e^{\lambda \cdot x_1}}$$

$$c = \frac{y_1}{e^{\lambda \cdot x_1}}$$

Second equation:

$$c \cdot e^{\lambda \cdot x_N} = y_N$$

$$\frac{y_1}{e^{\lambda \cdot x_1}} \cdot e^{\lambda \cdot x_N} = y_N$$

$$y_1 \cdot e^{\lambda \cdot (x_N - x_1)} = y_N$$

$$e^{\lambda \cdot (x_N - x_1)} = \frac{y_1}{y_N}$$

$$\lambda = \frac{\ln\left(\frac{y_1}{y_N}\right)}{x_N - x_1}$$

Now, we have $c_0 = \frac{y_1}{e^{\lambda \cdot x_1}}$ and $\lambda_0 = \frac{\ln\left(\frac{y_1}{y_N}\right)}{x_N - x_1}$ as initial guesses for c and λ, respectively.

Exponential least-squares, part II

Deadline: Friday, April 5th, 5pm. For this assignment, you must submit only code. All the pen & paper work necessary for completing this assignment was part of assignment 5.

Many processes in nature, like the growth of a bacterial culture in the abundance of nutrition or the intensity of light travelling through a medium, are exponential in nature. If a is the quantity we are interested in and x is the variable it depends on (in the examples time and distance travelled, respectively), then we expect that

$$a(x) = c\exp(\lambda x)$$

for some c, $\lambda \in \mathbb{R}$. When measuring a in an experiment, at discrete values of x, we necessarily incur some error and uncertainty. Assume we are given a list of measurements (x_j, y_j) where y_j is close to, but differs from, the model value $a(x_j)$.
In this assignment, you will implement a least-squares fit function for data of this kind. Follow the pointers below:

- The input should be two arrays of shape $(n+1,)$, one with x-values and one with the corresponding y-values.
- The output should be the values of c and λ that are the result of the least-squares computation. Do not change the inputs/outputs with respect to the starter code.
- In order to solve the system of equations (here Q is the sum of squares of errors, see parts (a) and (b) of assignment 5)

  $$\frac{\partial Q}{\partial c} = 0 \qquad \frac{\partial Q}{\partial \lambda} = 0$$

  you should use Newton-Raphson iteration. Pull the Newton-Raphson function from the course_codes repository. The inputs `f` and `Df` correspond to the system of equations above and the matrix of second-order derivatives (assignment 5 part `c`), respectively.
- You may have to make some improvements to the method you proposed to find good initial values, this problem appears to be rather sensitive to this choice.
- If you use other functions from the course_codes repository do include them with your submission so that the autograder can run your code.
- Set the tolerance for the error and residual small enough to compute the least-squares values of c and λ with at least 8 significant digits.