Assignment 4

20 November, 2023

Due date: 4 December

Instructions

- Submit to Avenue a **PDF file** containing your solutions.
  A file in another format e.g. IMG, PNG, will be ignored.
- Name your PDF file **Lastname-Firstname-studentnumber.pdf**.
- Name your MATLAB files **exactly** as specified.
- Submit your MATLAB code to Avenue and also include it in your PDF.
- Include your numerical results and plots in the PDF.
- Do not submit compressed (.zip, .rar) files.
  We will **ignore any compressed file** containing your files.

On using generative AI. Your are not permitted to use generative AI for this assignment. In alignment with McMaster academic integrity policy, it "shall be an offense knowingly to ... submit academic work for assessment that was purchased or acquired from another source". This includes work created by generative AI tools. Also state in the policy is the following, "Contract Cheating is the act of "outsourcing of student work to third parties" (Lancaster & Clarke, 2016, p. 639) with or without payment." Using Generative AI tools is a form of contract cheating. Charges of academic dishonesty will be brought forward to the Office of Academic Integrity.

Problem 1 [10 points] You are given the file `points.mat` with training data. There are three kinds of points. The goal is to train with these data and classify the points in $[0, 1] \times [0, 1]$.

Modify the file `netbpfull.m` such that it works with the three categories of points.

- Load the data with `load points.mat`. This will result in an array `x` containing points in 2D and an array `labels` containing labels.
- Modify the function `cost` such that it returns

$$\text{accuracy} = \frac{\text{number of points classified correctly}}{\text{total number of points}} \times 100$$

  and also returns the indices (in `x`) of training points that are not classified correctly.

- `netbpfull.m` should plot
  - accuracy versus number of iterations
  - cost versus number of iterations and
  - two plots like in Figure 1. Your plots may not be exactly like these two.
- The training should stop if accuracy of 95% is reached; otherwise it should continue to `Niter=1e6`. For full marks, you need to achieve 95%.

Submit `netbpfull.m` and the four plots. In the PDF, highlight the parts of `netbpfull.m` that you have modified.

Bonus [10 points] Modify `nlsrun.m` such that it runs on the points from `points.mat`. Highlight the changes in `nlsrun.m` in the PDF. Submit plots as in Figure 1 and indicate the accuracy over the training data. To get full marks, it should be above 95%. One of my plots is shown in Figure 2.

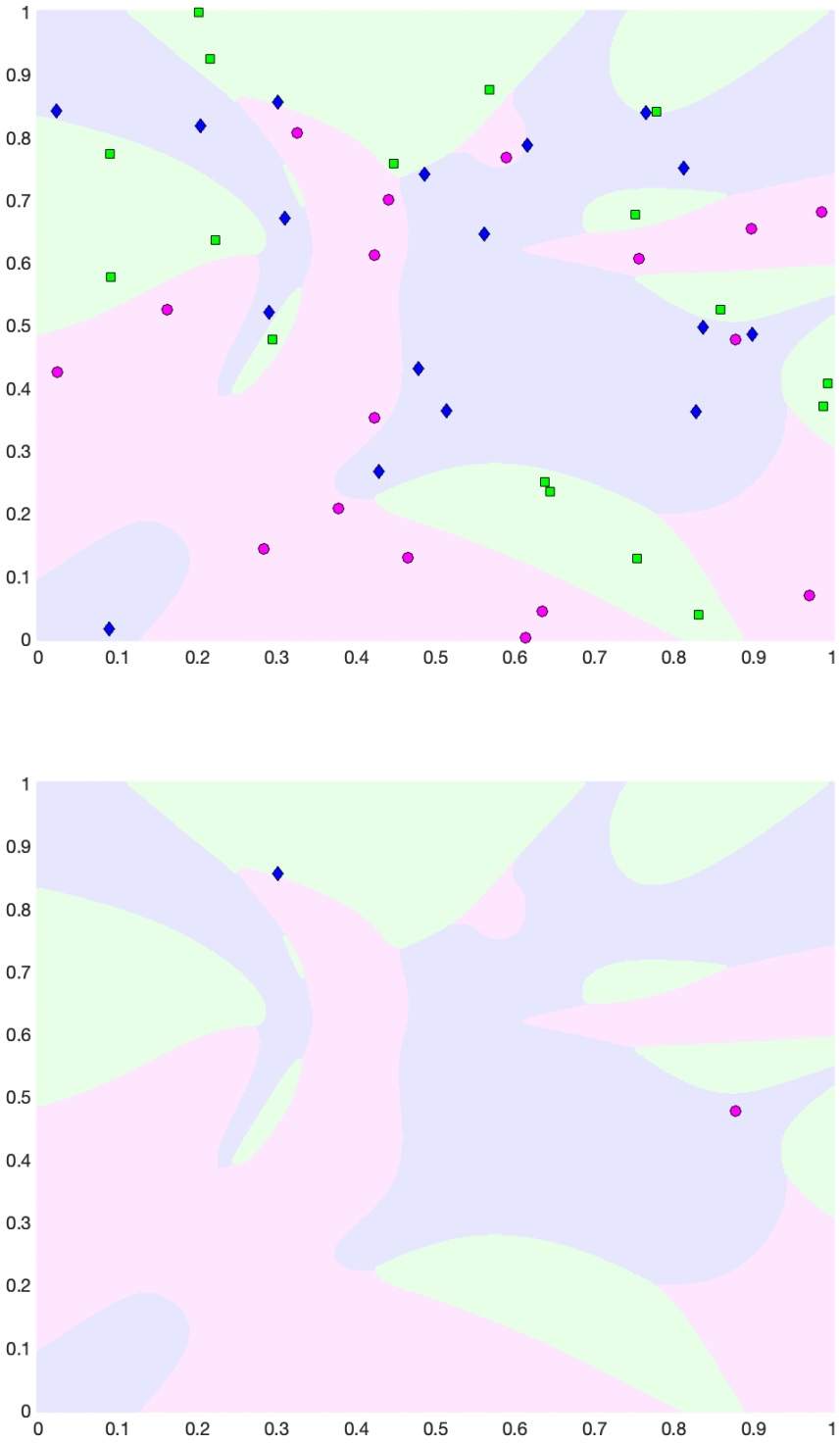

Figure 1: Top: training points and classified (x, y) points. Bottom: training points from `x` that are not classified correctly.

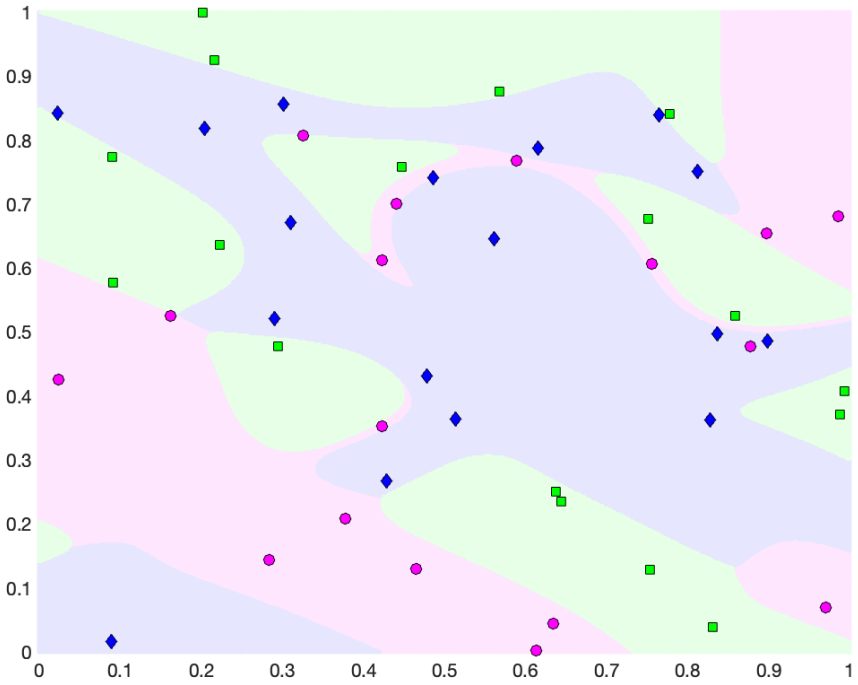

Figure 2: Plot using `nlsrun.m`

Problem 2 [8 points] Implement in Matlab the bisection and Newton's method for finding roots of scalar equations.

Use your implementation of the bisection method to find a root of

a. $1/x - \exp(2 - \sqrt{x})$ in $[0.1, 1]$

b. $x\sin(x) - 1$ in $[0, 2]$.

Use your implementation of Newton's method and Matlab's `fsolve` to find a root of

a. $1/x - \exp(2 - \sqrt{x})$ with initial guess $x_0 = 1$

b. $x\sin(x) - 1$ with initial guess $x_0 = 2$.

For the bisection method, use tol $= 10^{-10}$. For your Newton and `fsolve`, solve until $|f(x_n)| \leq 10^{-10}$.

Report in a table (in the PDF)

method	root r	$\lvert f(r)\rvert$	num. iterations
bisection			
Newton			
fsolve			

If you are obtaining different roots, explain the differences. Also, discuss the number of iterations.

Problem 3 [4 points] The annuity equation is

$$A = \frac{P}{r}(1 - (1 + r)^{-n}),$$

where A is borrowed amount, P is the amount of each payment, r is the interest rate per period, and there are n equally spaced payments.

a. Write Newton's method for finding r.

b. Implement the function

```
function r = interest(A, n, P)
```

which returns the annual interest rate. Your function must call `fsolve`. Ensure that `fsolve` uses the analytical form of the derivative.

Report the values of
`interest(100000, 20*12, 1000)`, `interest(100000, 20*12, 100)`.

Interpret the results.

Problem 4 [3 points] Consider Newton's method on

$$x^5 - x^3 - 4x = 0$$

(a) How do the computed approximations behave with $x_0 = 1$?

(b) Try your implementation with $x_0 = 1$ and $x_0 = 1 + 10^{-14}$. Explain why this method behaves differently, when started with $x_0 = 1 + 10^{-14}$, compared to when it is started with $x_0 = 1$.

(c) Solve also with `fsolve`. Comment on the results.

Problem 5 [8 points] Implement Newton's method for systems of equations.

Each of the following systems of nonlinear equations may present some difficulty in computing a solution. Use Matlab's `fsolve` and your own implementation of Newton's method to solve each of the systems from the given starting point.

In some cases, the nonlinear solver may fail to converge or may converge to a point other than a solution. When this happens, try to explain the reason for the observed behavior.

Report (in your PDF) for `fsolve` and your implementation of Newton's method and each of the systems below, the number of iterations needed to achieve accuracy of 10^{-6} (if achieved).

(a)

$$\begin{aligned} x_1 + x_2(x_2(5 - x_2) - 2) &= 13 \\ x_1 + x_2(x_2(1 + x_2) - 14) &= 29 \end{aligned}$$

starting from $x_1 = 15$, $x_2 = -2$.

(b)

$$\begin{aligned} x_1^2 + x_2^2 + x_3^2 &= 5 \\ x_1 + x_2 &= 1 \\ x_1 + x_3 &= 3 \end{aligned}$$

starting from $x_1 = (1 + \sqrt{3})/2$, $x_2 = (1 - \sqrt{3})/2$, $x_3 = \sqrt{3}$.

(c)

$$\begin{aligned} x_1 + 10x_2 &= 0 \\ \sqrt{5}(x_3 - x_4) &= 0 \\ (x_2 - x_3)^2 &= 0 \\ \sqrt{10}(x_1 - x_4)^2 &= 0 \end{aligned}$$

starting from $x_1 = 1$, $x_2 = 2$, $x_3 = 1$, $x_4 = 1$.

(d)

$$x_1 = 0$$
$$10x_1/(x_1 + 0.1) + 2x_2^2 = 0$$

starting from $x_1 = 1.8$, $x_2 = 0$.

Problem 6 [8 points] You are given the data file `data.txt`. Each row contains the 2D coordinates (x_i, y_i) of an object at time t_i. This object exhibits a periodic motion. Implement the function

```
function period = findPeriod(file_name)
```

that reads the data from a file and computes the period of the periodic motion.

The points in time where the object returns to the same position must be determined using `fsolve`.

Report the value for the computed period.

Problem 7 [3 points] Consider two bodies of masses $\mu = 0.012277471$ and $\widehat{\mu} = 1 - \mu$ (Earth and Sun) in a planar motion, and a third body of negligible mass (moon) moving in the same plane. The motion is given by

$$u_1'' = u_1 + 2u_2' - \widehat{\mu}\frac{u_1 + \mu}{((u_1 + \mu)^2 + u_2^2)^{3/2}} - \mu\frac{(u_1 - \widehat{\mu})}{\left((u_1 - \widehat{\mu})^2 + u_2^2\right)^{3/2}}$$
$$u_2'' = u_2 - 2u_1' - \widehat{\mu}\frac{u_2}{((u_1 + \mu)^2 + u_2^2)^{3/2}} - \mu\frac{u_2}{\left((u_1 - \widehat{\mu})^2 + u_2^2\right)^{3/2}}.$$

The initial values are

$$u_1(0) = 0.994, \qquad u_1'(0) = 0,$$
$$u_2(0) = 0, \qquad u_2'(0) = -2.00158510637908252242053786222 4.$$

Implement the classical Runge-Kutta method of order 4 and integrate this problem on [0,17.1] with uniform stepsize using 100, 1000, 10,000, and 20,000 steps. Plot the orbits for each case. How many uniform steps are needed before the orbit appears to be qualitatively correct?

Submit

- PDF: plots and discussion.

Problem 8 [5 points] The following system of ODEs, formulated by Lorenz, represents are crude model of atmospheric circulation:

$$\begin{aligned} y_1' &= \sigma(y_2 - y_1) \\ y_2' &= r y_1 - y_2 - y_1 y_3 \\ y_3' &= y_1 y_2 - b y_3 \end{aligned}$$

Set $\sigma = 10$, $b = 8/3$, $r = 28$, take initial values $y_1(0) = 15$, $y_2(0) = 15$, and $y_3(0) = 36$, and integrate this ODE from $t = 0$ to $t = 100$ using Matlab's `ode45`. Plot each component of the solution as a function of t. Plot also (y_1, y_2), (y_1, y_3), and (y_2, y_3) (in separate plots).

Change the initial values by a tiny amount (e.g. 10^{-10}) and integrate again. Compare the difference in the computed solutions.

Problem 9 Bonus [8 points] Let A be an $n \times n$ singular matrix. Let

$$F(X) = I - AX,$$

where I is the $n \times n$ identity matrix. When $F(X)$ is the zero $n \times n$ matrix, then $X = A^{-1}$. We can use Newton's method to find A^{-1}:

$$X_{k+1} = X_k + A^{-1}(I - AX_k).$$

We replace A^{-1} by X_k to obtain the formula

$$X_{k+1} = X_k + X_k(I - AX_k). \tag{1}$$

a. [4 points] Write a function to compute the inverse of a given matrix A using (1). You can use as an initial guess

$$X_0 = \frac{A^T}{\|A\|_1 \|A\|_\infty}$$

Test your program on a few random matrices and report numerical experiments comparing its accuracy and efficiency with Matlab's inverse function `inv`.

b. [4 points] Does (1) converge quadratically? Provide sufficient detail supporting your claim.