MATH 3340 - Scientific Computing Assignment 4

Libao Jin

March 10, 2021

Please note that the deadline will be enforced as per the previous homework. Remember that you are allowed to work in teams of two on this assignment. You are encouraged to prepare your work in LaTeX; a template will be provided to help you put it all together. If you choose to submit a hard copy, you may submit only one copy for a team, indicating the names of both contributors. Online submission is encouraged, however, in that case both members of a team should submit the PDF file containing their work and showing both their names.
All plots generated in this homework should have a title, legend, and labeled x and y-axes.

Instruction

1. Go to `https://www.overleaf.com` and sign in (required).
2. Click *Menu* (up left corner), then *Copy Project*.
3. Go to `LaTeX/meta.tex` (the file `meta.tex` under the folder `LaTeX`) to change the section and your name, e.g.,
   - change title to `\title{MATH 3340-01 Scientific Computing Homework 4}`
   - change author to `\author{Albert Einstein \& Carl F. Gauss}`
4. For Problem 1, you can either type the solution in LaTeX or write it on the printout.
5. For Problem 2, 3, 4, you need to write function/script files, store results to output files, and save graphs to figure files. Here are suggested names for function files, script files, output files, and figure files:

Problem	Function File	Script File	Output File	Figure File
2	`newtonNonlinear.m`	`hw4_p2.m`	`hw4_p2.txt`	
3		`hw4_p3.m`	`hw4_p3.txt`	`hw4_p3.pdf`
4		`hw4_p4.m`	`hw4_p4.txt`	`hw4_p4.pdf`

   Once finished, you need to upload these files to the folder `src` on Overleaf. If you have different filenames, please update the filenames in `\lstinputlisting{../src/your_script_name.m}` accordingly. You can code in the provided files in hw4.zip, and use the MATLAB script `save_results.m` to generate the output files and store the graphs to `.pdf` files automatically (the script filenames should be exactly same as listed above).
6. Recompile, download, and print out the generated PDF.
7. You may find LaTeX.Mathematical.Symbols.pdf and the second part of Lab 01 Slides and Lab 02 Slides helpful.

1 Problem 1

Do by hand, on paper, one iteration of Newton's method for the nonlinear system:

$$\begin{cases} x^2 + y^3 - 1 = 0, \\ x^3 - y^2 + 0.25 = 0. \end{cases}$$

Start with the initial guess $\mathbf{x}^0 = [x^0, y^0]^T = [0.5, 0.5]^T$ and compute the next iterate $\mathbf{x}^1$. Also compare the norm of the residual at the new iterate with the residual norm computed for the initial guess.

Solution.

- Script file `hw4_p1.m`

```
% MATH 3340, Spring 2020
% Homework 4, Problem 1
% Author: first_name last_name
% Date: 03/05/2020

clc; clear;

% f = @(x) [x(1)^2 + x(2)^3 - 1; x(1)^3 + x(2)^2 + 0.25];
% df = @(x) [2 * x(1), 3 * x(2)^2; 3 * x(1)^2, 2 * x(2)];

f = @(x) [x(1)^2 + x(2)^3 - 1; x(1)^3 - x(2)^2 + 0.25];
df = @(x) [2 * x(1), 3 * x(2)^2; 3 * x(1)^2, -2 * x(2)];
x0 = [0.5; 0.5];
maxIter = 100;
tol = 1e-6;
[x, iters, res] = newtonNonlinear(f, df, x0, maxIter, tol);
f(x0)
df(x0)
fprintf('%12s %10s %10s %10s\n', 'iterations', 'x', 'y', 'res');
for i = 1:length(iters)
    fprintf('%12d %10.4f %10.4f %10.2e\n', iters(i), x(1, i), x(2, i), res(i));
end

clear;
f = @(x) [x(1)^2 + x(2)^3 - 1; x(1)^3 + x(2)^2 + 0.25];
df = @(x) [2 * x(1), 3 * x(2)^2; 3 * x(1)^2, 2 * x(2)];
x0 = [0.5; 0.5];
maxIter = 100;
tol = 1e-6;
[x, iters, res] = newtonNonlinear(f, df, x0, maxIter, tol);
f(x0)
df(x0)
fprintf('%12s %10s %10s %10s\n', 'iterations', 'x', 'y', 'res');
for i = 1:length(iters)
    fprintf('%12d %10.4f %10.4f %10.2e\n', iters(i), x(1, i), x(2, i), res(i));
end
```

- Output file `hw4_p1.txt`

```
hw4_p1

ans =

  -0.6250
   0.1250


ans =

   1.0000    0.7500
   0.7500   -1.0000

  iterations          x          y        res
           0     0.5000     0.5000   6.37e-01
           1     0.8400     0.8800   3.93e-01
           2     0.7333     0.7905   3.73e-02
           3     0.7171     0.7863   6.36e-04
           4     0.7168     0.7863   2.09e-07

ans =

  -0.6250
   0.6250


ans =

   1.0000    0.7500
   0.7500    1.0000

  iterations          x          y        res
           0     0.5000     0.5000   8.84e-01
           1     3.0000    -2.0000   3.12e+01
           2     1.9224    -1.4612   9.50e+00
           3     1.1985    -0.9604   2.93e+00
           4     0.7662    -0.4236   1.01e+00
           5     0.7399     0.5597   1.01e+00
           6    11.5294   -16.1375   4.45e+03
           7     7.4640   -10.8072   1.32e+03
           8     4.7364    -7.2447   3.92e+02
           9     2.8885    -4.8548   1.17e+02
          10     1.6023    -3.2353   3.55e+01
          11     0.6247    -2.1070   1.11e+01
          12    -0.5125    -1.2522   3.18e+00
          13    -1.5262    -0.8989   2.57e+00
          14    -1.0923    -0.6013   6.92e-01
          15    -0.4715     0.6715   7.62e-01
          16    -1.1375     0.5583   1.02e+00
          17    -0.9149     0.5995   1.65e-01
          18    -0.8676     0.6310   6.45e-03
          19    -0.8654     0.6309   1.44e-05
          20    -0.8654     0.6309   5.97e-11
diary off
```

□

2 Problem 2

Solve the system

$$\begin{cases} 10 - x + \sin(x+y) - 1 = 0 \\ 8y - \cos^2(z-y) - 1 = 0 \\ 12z + \sin(z) - 1 = 0 \end{cases}$$

using a residual tolerance of 10^{-6} and the initial guess, $\mathbf{x}^0 = [0.1, 0.25, 0.08]^T$. Print out the values for x, y, and z for each iteration in a table similar to the one you created for the problem of the previous homework. You should submit your code (which can again be organized as a function and the script calling this function) together with your output.

Solution.

- Function file `newtonNonlinear.m`

```
function [x, iters, res] = newtonNonlinear(f, df, x0, maxIter, tol)
%NEWTONNONLINEAR: Newton Method for Nonlinear System
% Syntax: [x, iters] = newtonNonlinear(f, df, x0, maxIter, tol)
% Inputs:
%   f       = function (funciton handle)
%   df      = Jacobian matrix (function handle)
%   x0      = solution to the linear system (column vector)
%   maxIter = maximum of number of iterations (scalar)
%   tol     = tolerance (scalar)
% Outputs:
%   x     = solution to the linear system (matrix, i-th column is the solution in the i-th iteration)
%   iters = number of iterations performed (vector)
%   res   = norm of residuals (vector)
%
% Author: first_name last_name
% Date: 03/05/2020

x = x0;
res = norm(f(x0));
iters = 0;
while iters(end) < maxIter && res(end) > tol
    x0 = x0 - df(x0) \ f(x0);
    x = [x x0];
    res = [res; norm(f(x0))];
    iters = [iters; iters(end) + 1];
end

end
```

- Script file `hw4_p2.m`

```
% MATH 3340, Spring 2020
% Homework 4, Problem 2
% Author: first_name last_name
% Date: 03/05/2020

clc; clear;
```

```

f = @(x) [10 - x(1) + sin(x(1) + x(2)) - 1; 8 * x(2) - cos(x(3) - x(2)) ^ 2 - 1; 12 * x(3) + sin(x(3)) - 1];
df = @(x) [
      -1 + cos(x(1) + x(2)), cos(x(1) + x(2)), 0;
      0, 8 - 2 * cos(x(3) - x(2)) * sin(x(3) - x(2)), 2 * cos(x(3) - x(2)) * sin(x(3) - x(2));
      0, 0, 12 + cos(x(3))
    ];
x0 = [0.1; 0.25; 0.08];
maxIter = 100;
tol = 1e-6;
[x, iters, res] = newtonNonlinear(f, df, x0, maxIter, tol);

fprintf('%12s %10s %10s %10s %10s\n', 'iterations', 'x', 'y', 'z', 'res');
for i = 1:length(iters)
    fprintf('%12d %10.4f %10.4f %10.4f %10.2e\n', iters(i), x(1, i), x(2, i), x(3, i), res(i));
end
```

- Output file hw4_p2.txt

```
  iterations          x          y          z        res
           0     0.1000     0.2500     0.0800   9.24e+00
           1   152.4993     0.2464     0.0769   1.43e+02
           2    48.3973     0.2464     0.0769   4.04e+01
           3     9.9585     0.2464     0.0769   1.66e+00
           4     8.9871     0.2464     0.0769   2.03e-01
           5     9.0895     0.2464     0.0769   8.20e-04
           6     9.0891     0.2464     0.0769   7.50e-09
```

□

3 Problem 3

In a script file, find an exponential fit of the form $f(x) = Ce^{Ax}$ to the data (Table 1). Print your

Table 1: Problem 3 Data Points

x	0.0	0.1	0.2	0.3	0.4	0.5
y	1.388	1.647	1.951	2.633	3.321	3.977

values for A and C, then plot $f(x)$ vs. x where `x = 0:0.01:0.5` along with the original data points. Use a visible marker to mark the data points. Include the values of A and C in your report file along with the generated plot and script file contained your code.

Solution.

- Script file hw4_p3.m

```
% MATH 3340, Spring 2020
% Homework 4, Problem 3
% Author: first_name last_name
```

```
% Date: 03/05/2020

clc; clear;
% change default text interpreter to LaTeX
set(groot,'defaultTextInterpreter','latex');
set(groot, 'defaultAxesTickLabelInterpreter','latex');
set(groot, 'defaultLegendInterpreter','latex')
figure(3); hold on;

xx = [0.0 0.1 0.2 0.3 0.4 0.5]';
yy = [1.388 1.647 1.951 2.633 3.321 3.977]';
% xx = [-0.5 -0.25 0 0.25 0.5]';
% yy = [0.4 1.5 5.5 17 62]';

% C e^(Ax) => ln(y) = Ax + ln(C)
A = [xx ones(size(xx))];
b = log(yy);
c = (A' * A) \ (A' * b);

fprintf('%8s %8s\n', 'A', 'C');
fprintf('%8.4f %8.4f\n', c(1), exp(c(2)));

x = min(xx):0.01:max(xx);
y = exp(c(1) * x) * exp(c(2));

plot(xx, yy, 'o');
plot(x, y, '-');
xlabel('$x$');
ylabel('$y$');
title('Exponential fit');
grid minor;
legend(...
    {'data points',sprintf('$f(x) = %.4f e^{%.4f x}$', exp(c(2)), c(1))},...
    'Location', 'best');
```

- Output file `hw4_p3.txt`

```
       A        C
  2.1906   1.3405
```

□

4 Problem 4

Write a script file to find a quadratic fit of the form $f(x) = Ax^2 + Bx + C$ to the data (Table 2). Print your values for A, B, and C, then plot $f(x)$ vs x where `x = 0:0.1:4` along with the original

Table 2: Problem 4 Data Points

x	0	1	2	3	4
y	0.695	-1.475	-1.275	0.882	4.765

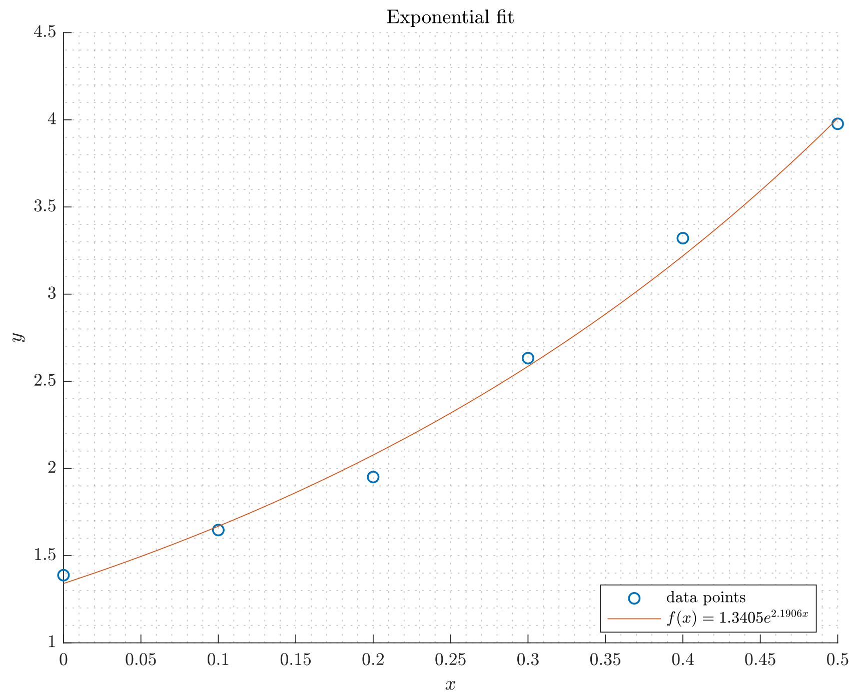

Figure 1

data points. Use a visible marker for the data points. Include the values of A, B, and C in your report file along with the generated plot and script file containing your code.

Solution.

- Script file `hw4_p4.m`

```
% MATH 3340, Spring 2020
% Homework 4, Problem 4
% Author: first_name last_name
% Date: 03/05/2020

clc; clear;
% change default text interpreter to LaTeX
set(groot,'defaultTextInterpreter','latex');
set(groot, 'defaultAxesTickLabelInterpreter','latex');
set(groot, 'defaultLegendInterpreter','latex')
figure(4); hold on;

xx = [0 1 2 3 4]';
yy = [0.695 -1.475 -1.275 0.882 4.765]';

% xx = [-2 -1 0 1 2]';
% yy = [4.3 1.2 0.02 0.9 3.9]';

A = [xx.^2 xx ones(size(xx))];
c = (A' * A) \ (A' * yy);

```

```
fprintf('%8s %8s %8s\n', 'A', 'B', 'C');
fprintf('%8.4f %8.4f %8.4f\n', c(1), c(2), c(3));

% x = 0:0.1:4;
x = min(xx):0.1:max(xx);
y = c(1) * x.^2 + c(2) * x + c(3);

plot(xx, yy, 'o');
plot(x, y, '-');
xlabel('$x$');
ylabel('$y$');
title('Quadratic fit');
grid minor;
legend(...
    {'data points',sprintf('$f(x) = %.4f x^{2} + %.4f x + %.4f$', c(1), c(2), c(3))},...
    'Location', 'best');
```

- Output file `hw4_p4.txt`

```
       A        B        C
  1.0045  -2.9683   0.6280
```

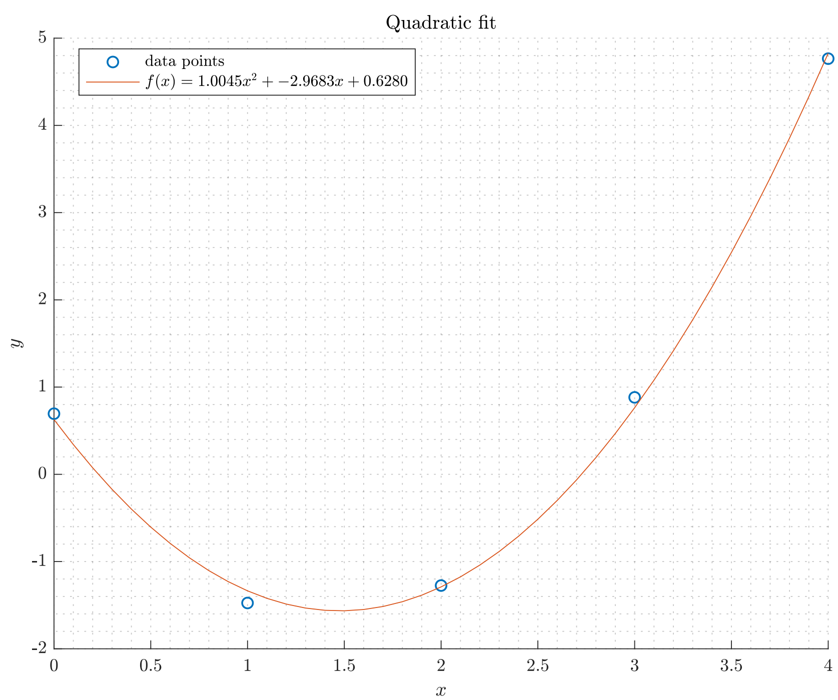

Figure 2

□