Numerical Methods for Engineers

Homework No. 1

Shlomi Shitrit
Borak Mansour

January 30, 2024

Solution:

1. a. the requested function:

```
function [R] = MyDist_a(P, P0)
%MYDIST_A returns the distance between ...
    points vector P to the point P0
%using 1 line of code.
%    P0 - [1 N] dimension while N it is ...
    the point dimension
%    P  - [M N] dimension while M it is ...
    the number of points

R=sqrt(sum((P-P0).^2,2));
end
```

b. the requested function:

```
function [R] = MyDist_b(P, P0)
%MYDIST_A returns the distance between ...
    points vector P to the point P0
%using for loop
%    P0 - [1 N] dimension while N it is ...
    the point dimension
%    P  - [M N] dimension while M it is ...
    the number of points

num_of_points = length(P);
R = zeros(num_of_points, 1);
for i = 1:num_of_points
    R(i) = sqrt(sum((P(i,:)-P0).^2,2));
end

end
```

c. while comparing the 2 function's runtime with tic-toc we most of the time see that one-line function is faster. one running result comparsion: Elapsed time is 0.000518 seconds. Elapsed time is 0.000772 seconds.

$$\texttt{MyDist\_a} \longrightarrow \text{Elapsed time is 0.000518 seconds.}$$

$$\texttt{MyDist\_b} \longrightarrow \text{Elapsed time is 0.000772 seconds.}$$

which is almost doubled time and it is only 10 points. We have to say that we get a quiet random result and from time-to-time the for-loop solution is faster. For that, we took a side-try of 10000 points and the results are clearer while the for-loop is slower.

d. The requested function:

```
function  MyPlot(P, P0)
%MYPLOT plots the points P and the point ...
    P0, connect line between them
% and texting the distance and each point
%   P0 - [1 N] dimension while N it is ...
    the point dimension
%   P  - [M N] dimension while M it is ...
    the number of points

num_of_points = length(P);
R = MyDist_a(P,P0);
f=figure(1);
f.Name = 'Points Graph';

text(P0(1),P0(2),'P_0');
hold on
for i = 1:num_of_points
    x_line = [P0(1), P(i,1)];
    y_line = [P0(2), P(i,2)];
    plot(x_line,y_line,'Color','g',...
    'Marker','o','MarkerEdgeColor','b',...
    'MarkerFaceColor','b','MarkerSize',3);
    text(P(i,1),P(i,2),['P_{' num2str(i) ...
        '}']); % point number
    x_coor = (P0(1)+P(i,1))/2;
    y_coor = (P0(2)+P(i,2))/2;
    text(x_coor,y_coor,num2str(R(i),2))
end
hold off
end
```

And we got the following plot:

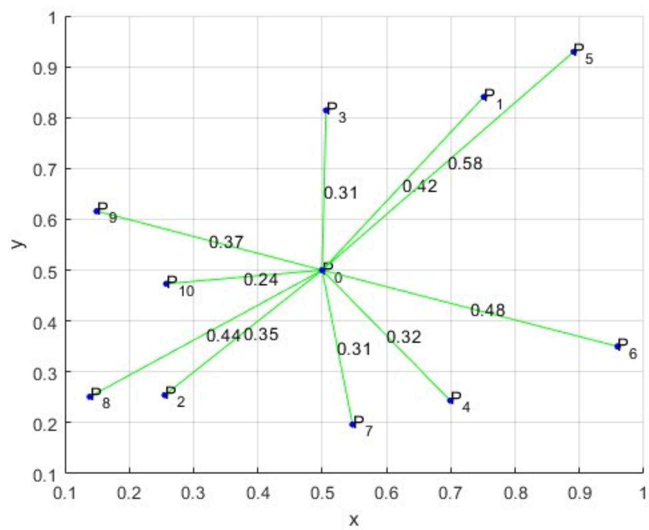

Figure 1: 10 random points and their distance from P_0

2. Given:

$$f(x, y) = \sin(x)\ln(xy)$$

a. Rewrite the Taylor series of this function, using MATLAB. we will symbolicly use the vectors and matrices form[1]:

$$T(\mathbf{x}) = f(\mathbf{a}) + (\mathbf{x}-\mathbf{a})^T \mathcal{D} f(\mathbf{a}) + \frac{1}{2!}(\mathbf{x}-\mathbf{a})^T \mathcal{D}^2 f(\mathbf{a})(\mathbf{x}-\mathbf{a}) + \ldots$$

– First order polynomial:
$T_1(x, y) = \sin(1)\ x + \sin(1)\ y - 2\ \sin(1)$

Second order polynomial:
$T_2(x.y) = \left(\cos(1) - \frac{\sin(1)}{2}\right) x^2 + \cos(1)\ x\,y + (2\ \sin(1) - 3\ \cos(1))\, x + \left(-\frac{\sin(1)}{2}\right) y^2 + (2\ \sin(1) - \cos(1))\, y + 2\ \cos(1) - 3\ \sin(1)$

[1] Taylor Series in Wikipedia

Third order polynomial:
$T_3(x,y) = \left(-\frac{\cos(1)}{2} - \frac{\sin(1)}{6}\right) x^3 + \left(-\frac{\sin(1)}{2}\right) x^2\, y + \left(\frac{5\cos(1)}{2} + \frac{\sin(1)}{2}\right) x^2 +$
$+ \left(-\frac{\cos(1)}{2}\right) x\, y^2 + \left(2\cos(1) + \sin(1)\right) x\, y + \left(\frac{\sin(1)}{2} - 5\cos(1)\right) x +$
$+ \frac{\sin(1)}{3}\, y^3 + \left(\frac{\cos(1)}{2} - \frac{3\sin(1)}{2}\right) y^2 + \left(\frac{5\sin(1)}{2} - 2\cos(1)\right) y +$
$+ 3\cos(1) - \frac{8\sin(1)}{3}$

b. We got plot of the surface of the 3 functions compared to the real surface:

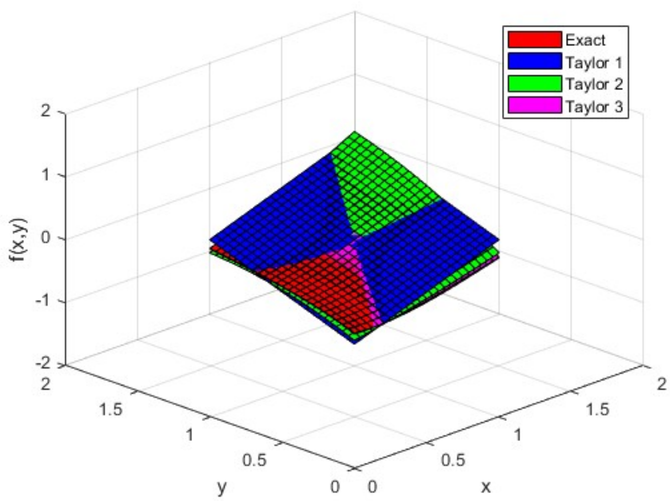

Figure 2: $f(x,y)$ and its 3 first polynomial Taylor series

all the code for this question can be seen separately on the MATLAB files.

3. The fixed code (also submitted):

```
clc;
x=[0; 1];
x_prev=[0;0]; % to enter while loop
e=1e-5;
IterN=0;

while norm(x-x_prev)>e
    x_prev=x;
    F=[x(1)^2+x(2)^2-4 ; exp(x(1))+x(2)-1];
    J=[2*x(1)      2*x(2);
       exp(x(1))       1];
    dx=-J\F;
    x=x+dx;
    IterN=IterN+1
end
```

4. The given diagram can be rewritten as the following pseudo code:

if $x < 10$ **then**
  if $x < 5$ **then**
    $x = 5$
  else
    Print x
  end if
else
  while $x \geq 50$ **do**
    $x = x - 5$
  end while
end if

Discussion and Conclusion

In this homework we backed to basis - experienced with some MATLAB and pseudo-code writing and re-freshen up with Taylor series and Newton's numerical method for non-linear set of equations solution. More detailed - in question 1 we did some comparison between a for loop and a built-in MATLAB function to taste the effects of numerical calculations. In question 2 we see, graphically, the error's order effect. In question 3 we exposed to Newton algorithm code. Finally, in question 4 we practiced on pseudo-code writing.