# Αριθμητική Ανάλυση I και Εργαστήριο

# Τελική Εργασία 2022

Ονοματεπώνυμο: Ελένη Στυλιανού

Email: elenistylianou03@live.com

Αριθμός Μητρώου: ge21708

Εξάμηνο: 3ο

Εργαστήριο: Ομάδα Α-Δευτέρα 15:00-16:30

## Άσκηση 1

Στην άσκηση 1 χρησιμοποιήθηκε το πρόγραμμα MATLAB.
Με την βοήθεια του editor δημιουργήθηκε ένα αρχείο «askisi1.m» στο οποίο βρίσκεται η εκτέλεση της άσκησης.

<u>Κώδικας:</u>

```
clear
clc

k=[4,6,20,200];
x1=1;
x2=2;
X0=[x1;x2];
for j=1:4
    xtel=NR(X0,k(j));
end

x1=2;
x2=4;
X0=[x1;x2];
for j=1:4
    xtel=NR(X0,k(j));
end

X0=rand(2,1);
for j=1:4
    xtel=NR(X0,k(j));
end
```

Επίσης, για την εκτέλεση του κώδικα της συγκεκριμένης άσκησης δημιουργήσαμε τρεις συναρτήσεις.
Για την κατασκευή της συνάρτησης F, η οποία δίνει τα κρίσιμα σημεία της E όταν F(x)=0, δημιουργήσαμε την συνάρτηση «F.m» .

<u>Κώδικας:</u>

```
function [f]=F(X)
x1=X(1,1);
x2=X(2,1);
f(1,1)=-(((x1-10/9)^4+(x2-10/9)^4-cos((x1-10/9)^4+(x2-10/9)^4)+2)^(-2))*4*(x1-10/9)^3*(1+sin((x1-10/9)^4+(x2-10/9)^4));
f(2,1)=-(((x1-10/9)^4+(x2-10/9)^4-cos((x1-10/9)^4+(x2-10/9)^4)+2)^(-2))*4*(x2-10/9)^3*(1+sin((x1-10/9)^4+(x2-10/9)^4));
end
```

Στην συνάρτηση αυτή παραγωγίσαμε ως προς x1 την E και ορίσαμε το αποτέλεσμα ως f(1,1) και παραγωγίσαμε ως προς x2 την E και ορίσαμε το αποτέλεσμα ως f(2,1)

Για την υλοποίηση του Ιακωβιανού πίνακα της F σε κάθε σημείο x του χώρου δημιουργήσαμε τη συνάρτηση «JF.m».

<u>Κώδικας:</u>

```
function [J]=JF(X)
x1=X(1,1);
x2=X(2,1);
J(1,1)=32*(((x1-10/9)^4+(x2-10/9)^4-cos((x1-10/9)^4+(x2-10/9)^4)+2)^(-3))*(x1-10/9)^6*(1+sin((x1-10/9).^4+(x2-10/9)^4)).^2
          -(((x1-10/9)^4+(x2-10/9)^4-cos((x1-10/9)^4+(x2-10/9)^4)+2)^(-2))*(12*(x1-10/9)^2*(1+sin((x1-10/9)^4+(x2-10/9)^4))+16*(x1-10/9)^6*cos((x1-10/9)^4+(x2-10/9)^4));
J(1,2)=32*(((x1-10/9)^4+(x2-10/9)^4-cos((x1-10/9)^4+(x2-10/9)^4)+2)^(-3))*(x2-10/9)^6*(1+sin((x1-10/9).^4+(x2-10/9)^4)).^2
          -(((x1-10/9)^4+(x2-10/9)^4-cos((x1-10/9)^4+(x2-10/9)^4)+2)^(-2))*(12*(x2-10/9)^2*(1+sin((x1-10/9)^4+(x2-10/9)^4))+16*(x2-10/9)^6*cos((x1-10/9)^4+(x2-10/9)^4));
J(2,1)=32*(((x1-10/9)^4+(x2-10/9)^4-cos((x1-10/9)^4+(x2-10/9)^4)+2)^(-3)*(x1-10/9)^3*(x2-10/9)^3*(1+sin((x1-10/9)^4+(x2-10/9)^4))^2)
          -(((x1-10/9)^4+(x2-10/9)^4-cos((x1-10/9)^4+(x2-10/9)^4)+2)^(-2))*16*(x1-10/9)^3*(x2-10/9)^3*cos((x1-10/9)^4+(x2-10/9)^4);
J(2,2)=32*(((x1-10/9)^4+(x2-10/9)^4-cos((x1-10/9)^4+(x2-10/9)^4)+2)^(-3)*(x1-10/9)^3*(x2-10/9)^3*(1+sin((x1-10/9)^4+(x2-10/9)^4))^2)
          -(((x1-10/9)^4+(x2-10/9)^4-cos((x1-10/9)^4+(x2-10/9)^4)+2)^(-2))*16*(x1-10/9)^3*(x2-10/9)^3*cos((x1-10/9)^4+(x2-10/9)^4);
end
```

Στην συνάρτηση αυτή παραγωγίσαμε ως προς x1 και x2 την f(1,1) και ορίσαμε τα αποτελέσματα ως J(1,1) και J(2,1) αντίστοιχα και παραγωγίσαμε ως προς x1 και x2 την f(2,1) και ορίσαμε τα αποτελέσματα ως J(2,2) και J(1,2) αντίστοιχα.

Για την υλοποίηση της μεθόδου του Νεύτωνα (Newton-Raphson), με την οποία επιτυγχάνουμε την προσέγγιση των κρίσιμων σημείων, κατασκευάσαμε τη συνάρτηση «NR.m».

Κώδικας:

```
function [xtel]=NR(X0,k)
for i=1:k
    xtel=JF(X0)\(JF(X0)*X0-F(X0));
    X0=xtel;
end
xtel
end
```

Στη συγκεκριμένη συνάρτηση χρησιμοποιήθηκε η ανάποδη διαίρεση, ώστε να αποφύγουμε την χρήση της εντολής inv().

xtel=X0-JF(X0)^(-1)*F(X0)=>JF(X0)*xtel=JF(X0)*X0-F(X0)=>xtel=JF(X0)\(JF(X0)*X0- F(X0))

Με την εκτέλεση του κώδικα παίρνουμε τα εξής αποτελέσματα στο command window:

- Για X0=(1,2):
  - k=4
    ```
    xtel =

      -94.6079
      -49.8465
    ```
  - k=6
    ```
    xtel =

      -94.6079
      -49.8465
    ```
  - k=20
    ```
    xtel =

      -94.6079
      -49.8465
    ```
  - k=200
    ```
    xtel =

      -94.6079
      -49.8465
    ```
- Για X0=(2,4):
  - k=4
    ```
    xtel =

      -0.1997
       4.0140
    ```

- k=6

  xtel =

  -0.1880
  4.0140
- k=20

  xtel =

  -0.1840
  4.0140
- k=200

  xtel =

  -0.1840
  4.0140

• Για X0 τυχαίο διάνυσμα, το οποίο δίνεται από την εντολή rand(2,1). Για τα συγκεκριμένα αποτελέσματα έχουμε X0=(0.8061,0.8248):

- k=4

  xtel =

  67.6458
  -74.6952
- k=6

  xtel =

  67.6458
  -74.6952
- k=20

  xtel =

  67.6458
  -74.6952

  k=200

  xtel =

  67.6458
  -74.6952

Από τα παραπάνω αποτελέσματα παρατηρούμε πως έχουμε αρκετά καλές προσεγγίσεις στα κρίσιμα σημεία και πως όσο περισσότερες επαναλήψεις εκτελεστούν τόσο πιο ακριβές είναι το αποτέλεσμα μας, γεγονός που φαίνεται κυρίως στη δεύτερη περίπτωση για X0=(1,2).

Στη συνέχεια κατασκευάσαμε κώδικα για την υλοποίηση της τροποποιημένης μεθόδου του Νεύτωνα(Newton-Raphson) για X0=(2,4).

<u>Κώδικας:</u>

```
clear
clc

k=[4,6,20,200];
x1=1;
x2=2;
X0=[x1;x2];
xprev=X0;
for j=1:4
    for i=1:k(j)
        xtel=JF(X0)\(JF(X0)*xprev-F(xprev));
        xprev=xtel;
    end
    xtel
end
```

Και σε αυτή την περίπτωση χρησιμοποιήθηκε η ανάποδη διαίρεση, ώστε να αποφύγουμε την χρήση της εντολής inv().

Με την εκτέλεση του κώδικα παίρνουμε τα εξής αποτελέσματα στο command window:

- k=4
  xtel =

    -94.6079
    -49.8465
- k=6
  xtel =

    -94.6079
    -49.8465
- k=20
  xtel =

    -94.6079
    -49.8465
- k=200
  xtel =

    -94.6079
    -49.8465

Η τροποποιημένη μέθοδος του Νεύτωνα χρησιμοποιεί τον Ιακωβιανό πίνακα υπολογισμένο στο X0 σε κάθε επανάληψη, επομένως με την χρήση αυτής της μεθόδου το πρόγραμμα είναι οικονομικότερο όσον αφορά τις πράξεις που εκτελούνται.

## Άσκηση 2

Στην άσκηση 2 χρησιμοποιήθηκε το πρόγραμμα MATLAB.
Με την βοήθεια του editor δημιουργήθηκε ένα αρχείο «askisi2.m» στο οποίο βρίσκεται η εκτέλεση της άσκησης.

<u>Κώδικας:</u>

```
clear
clc

k=[4,6,20,200];
n=[4,6,20,200];
for i=1:4
    x=ones(n(i),1).*(10/9);
    x(1,1)=x(1,1)/2;
    X0=x;
    for j=1:4
        if n(i)>6
           xtel=norm(NRn(X0,k(j),n(i)));
        else xtel=NRn(X0,k(j),n(i));
        end
        xtel
    end
end

for i=1:4
    X0=randn(n(i),1)
    for j=1:4
        if n(i)>6
           xtel=norm(NRn(X0,k(j),n(i)));
        else xtel=NRn(X0,k(j),n(i));
        end
        xtel
    end
end
```

Επίσης, για την εκτέλεση του κώδικα της συγκεκριμένης άσκησης δημιουργήσαμε τρεις συναρτήσεις.

Για την κατασκευή της συνάρτησης Fn, η οποία δίνει τα κρίσιμα σημεία της E όταν F(x)=0, δημιουργήσαμε την συνάρτηση «Fn.m» .

<u>Κώδικας:</u>

```
function [f] = Fn(X,n)
a=10/9*ones(n,1);
sum=0;
for i=1:n
    sum=sum+(X(i,1)-a(i,1))^4;
end
for i=1:n
    f(i,1)=-(sum-cos(sum)+n)^(-2)*4*abs(X(i)-a(i))^3*(1+sin(sum));
end
end
```

Στην συνάρτηση αυτή παραγωγίσαμε ως προς X(i) την E και ορίσαμε το αποτέλεσμα ως f(i,1), i=1,…,n. Επίσης, χρησιμοποιήθηκε η εντολή ones(n,1), η οποία μας δίνει πίνακα n×1 με όλα τα στοιχεία του ίσα με 1.

Για την υλοποίηση του Ιακωβιανού πίνακα της f σε κάθε σημείο x του χώρου δημιουργήσαμε τη συνάρτηση «JFn.m».

<u>Κώδικας:</u>

```
function [J] = JFn(X,n)
a=10/9*ones(n,1);
sum=0;
for i=1:n
    sum=sum+(X(i,1)-a(i,1))^4;
end
for i=1:n
    f(i,1)=-(sum-cos(sum)+n)^(-2)*4*abs(X(i)-a(i))^3*(1+sin(sum));
end
for i=1:n
    for j=1:n
        if i==j
            J(i,j)=32*(sum-cos(sum)+n)^(-3)*(X(i)-a(i))^6*(1+sin(sum))^2-4*(sum-cos(sum)+n)^(-2)*(X(i)-a(i))^2*(3*(1+sin(sum))+4*(X(i)-a(i))^4*cos(sum));
        else
            J(i,j)=16*(sum-cos(sum)+n)^(-3)*(X(i)-a(i))^3*(X(j)-a(j))^3*(2*(1+sin(sum))^2-cos(sum)*(sum-cos(sum)+n));
        end
    end
end
end
```

Στην συνάρτηση αυτή παραγωγίσαμε ως προς X(i) για i=j και X(j) για i≠j την f(i,1) και ορίσαμε τα αποτελέσματα ως J(i,j), i=1,…,n και j=1,…,n.

Για την υλοποίηση της μεθόδου του Νεύτωνα (Newton-Raphson), με την οποία επιτυγχάνουμε την προσέγγιση των κρίσιμων σημείων, κατασκευάσαμε τη συνάρτηση «NRn.m».

<u>Κώδικας:</u>

```
function [xtel]=NRn(X0,k,n)
for i=1:k
    xtel=JFn(X0,n)\(JFn(X0,n)*X0-Fn(X0,n));
    X0=xtel;
end
end
```

Χρησιμοποιήθηκε και σε αυτή τη συνάρτηση η ανάποδη διαίρεση, ώστε να αποφύγουμε την χρήση της εντολής inv().

Με την εκτέλεση του κώδικα παίρνουμε τα εξής αποτελέσματα στο command window:

- Για $X0=10/9*(0.5, 1, 1, \dots, 1)^T \in R^n$

  Παίρνουμε για όλα τα n και k

  xtel =

     NaN

  Παρατηρούμε πως παρουσιάζεται σφάλμα εξαιτίας του πίνακα JFn, ο οποίος δεν είναι αντιστρέψιμος αλλά ούτε καλά ορισμένος για το συγκεκριμένο αρχικό διάνυσμα, γεγονός που μας δείχνει πως δεν μπορούν να υπολογιστούν τα κρίσιμα σημεία στη συγκεκριμένη περίπτωση.

- Για X0 τυχαίο διάνυσμα, το οποίο δίνεται από την εντολή randn(n(i),1).
  - n=4

    Για τα συγκεκριμένα αποτελέσματα έχουμε:

    X0 =

       -0.0890
       -1.0334
        0.1007

1.6681

- k=4

  xtel =

  -0.1287
  -1.1044
  0.0672
  1.1187

- k=6

  xtel =

  -0.1488
  -1.1403
  0.0503
  1.1120

- k=20

  xtel =

  -0.3440
  -1.4891
  -0.1140
  1.1111

- k=200

  xtel =

  NaN
  NaN
  NaN
  NaN

➢ n=6

Για τα συγκεκριμένα αποτελέσματα έχουμε:

X0 =

-0.3991
0.6275
-1.1213
-0.9217
-1.2390
-1.4347

- k=4

  xtel =

  -0.3169

0.6539
-0.9997
-0.8109
-1.1110
-1.2960

- k=6

xtel =

-0.3963
0.6284
-1.1171
-0.9178
-1.2346
-1.4299

- k=20

xtel =

-0.3398
0.6465
-1.0337
-0.8418
-1.1467
-1.3347

- k=200

xtel =

-0.8081
0.4966
-1.7259
-1.4721
-1.8754
-2.1241

- n=20

Για τα συγκεκριμένα αποτελέσματα έχουμε:

X0 =

-0.9425
-0.8996
-0.4421
-1.6435
-0.5069
-0.7749
-0.7937

-0.3168
1.1718
-0.0048
0.4210
1.5855
0.6846
0.1294
1.0860
2.2299
-0.5170
0.8484
-0.8140
-1.2485

- k=4
  xtel =

  3.8335
- k=6
  xtel =

  3.8262
- k=20
  xtel =

  5.0396
- k=200
  xtel =

  NaN

- n=200
  Για τα συγκεκριμένα αποτελέσματα έχουμε:
  X0 =

  -0.6774
  -0.0737
  -1.9617
  -1.5986
  -0.0571
  0.6006
  0.7855
  0.4373
  -0.1798
  0.5300

2.4929
1.9768
-0.0594
1.4144
-0.6206
0.3508
0.1444
0.7633
-1.2149
0.6243
0.1686
0.3288
-1.0286
-0.3936
-1.5441
-0.7339
-0.9699
-0.3496
0.1692
0.8480
-1.0691
0.2936
1.0865
0.9613
-1.5944
0.7320
-0.1688
1.3814
0.9293
-0.3488
0.2694
0.5109
-2.4612
-0.6140
-0.3954
-1.1448
-2.2324
0.8400
-0.6138
1.1722
-0.9582
0.9531
0.7034

0.5599
-1.0380
-0.1783
-0.4699
-0.9599
-0.4450
-1.6212
-0.1574
-0.1975
-0.8314
0.6373
-1.0486
-0.2482
0.5249
0.2132
-2.3974
-0.7724
-0.4061
-0.5533
-1.3530
-0.0431
-0.1676
1.2796
0.1122
0.7055
-0.1304
-1.8255
-1.4733
-0.7514
-0.4337
0.5764
-0.8345
-0.6288
0.2559
-0.5620
-1.2284
-0.5540
-0.6803
0.6912
0.2219
0.5459
-0.4597
2.1167

0.1091
-0.9198
-1.3236
0.0987
-0.4066
-1.3519
-0.4698
-1.6280
0.6824
1.9437
0.7402
-0.3232
1.3596
0.2910
-2.2190
-0.1446
0.0787
0.5622
-0.1643
1.1737
-1.3284
2.4592
1.3522
-0.2468
0.2880
-0.8727
-0.9308
0.0762
0.9857
0.9258
2.1315
-0.3258
-0.2230
-2.7343
-0.3085
0.1461
0.0377
-0.6476
-1.2874
-0.4110
2.2005
-0.9290
0.0027

-1.1576
1.1645
0.7864
0.1603
-0.0475
1.4789
0.5778
-0.0035
0.1367
1.5390
0.4948
0.2689
0.5686
-0.0986
-0.9872
0.3676
-1.6223
-0.6941
1.0579
-0.6946
0.2701
-0.2153
0.2775
0.4236
-0.5009
-2.1246
1.0212
-0.5579
0.2848
0.9696
-1.0824
1.2562
0.7412
0.1224
-0.3588
1.2052
0.4458
0.5323
2.6929
0.5546
1.1765
-0.0507
1.8023

-0.0951
-0.9950
-0.0166
-1.7160
1.2576
-1.1842
0.2816
0.9143
0.4827
0.3736
-0.8645
-0.0551
0.9304
-0.7636
-1.2357
-0.6987
0.8544
-0.2746

- k=4

  xtel =

  12.7200
- k=6

  xtel =

  12.7034
- k=20

  xtel =

  12.7643
- k=200

  xtel =

  NaN

Όταν n=20 και n=200 παίρνουμε ως αποτέλεσμα την Ευκλείδεια νόρμα του διανύσματος xtel.

Στη συνέχεια κατασκευάσαμε κώδικα για την υλοποίηση της τροποποιημένης μεθόδου του Νεύτωνα(Newton-Raphson).

<u>Κώδικας:</u>

```
clear
clc

k=[4,6,20,200];
n=[4,6,20,200];
for i=1:4
    x=ones(n(i),1).*(10/9);
    x(1,1)=x(1,1)/2;
    X0=x;
    xprev=X0;
    for j=1:4
        for w=1:k(j)
            if n(i)>6
               xtel=norm(JFn(X0,n(i))\(JFn(X0,n(i))*xprev-Fn(xprev,n(i))));
            else xtel=JFn(X0,n(i))\(JFn(X0,n(i))*xprev-Fn(xprev,n(i)));
            xprev=xtel;
            end
            xtel
        end
    end
end

for i=1:4
    X0=randn(n(i),1);
    xprev=X0;
    for j=1:4
        for w=1:k(j)
            if n(i)>6
               xtel=norm(JFn(X0,n(i))\(JFn(X0,n(i))*xprev-Fn(xprev,n(i))));
            else xtel=JFn(X0,n(i))\(JFn(X0,n(i))*xprev-Fn(xprev,n(i)));
            xprev=xtel;
            end
        end
        xtel
    end
end
```

Και σε αυτή την περίπτωση χρησιμοποιήθηκε η ανάποδη διαίρεση, ώστε να αποφύγουμε την χρήση της εντολής inv().

Με την εκτέλεση του κώδικα παίρνουμε τα εξής αποτελέσματα στο command window:

- Για X0=10/9*(0.5, 1, 1, …, 1)$^T$ ∈ $R^n$

  Παίρνουμε για όλα τα n και k

  xtel =

  NaN

  Παρατηρούμε πως και στην τροποποιημένη μέθοδο παρουσιάζεται σφάλμα εξαιτίας του πίνακα JFn, ο οποίος δεν είναι αντιστρέψιμος αλλά ούτε καλά ορισμένος για το συγκεκριμένο αρχικό διάνυσμα, γεγονός που μας δείχνει πως δεν μπορούν να υπολογιστούν τα κρίσιμα σημεία στη συγκεκριμένη περίπτωση.

- Για X0 τυχαίο διάνυσμα, το οποίο δίνεται από την εντολή randn(n(i),1).
  - n=4

    Για τα συγκεκριμένα αποτελέσματα έχουμε:

    X0 =

    -0.4267
     0.4685
     0.2034
    -1.6007
    - k=4

xtel =

-0.3882
0.4846
0.2261
-1.5329

- k=6

xtel =

-0.3717
0.4915
0.2359
-1.5037

- k=20

xtel =

-0.3457
0.5024
0.2512
-1.4579

- k=200

xtel =

-0.3454
0.5025
0.2514
-1.4573

➢ n=6

Για τα συγκεκριμένα αποτελέσματα έχουμε:

X0 =

0.7659
-0.2563
-0.6622
-0.1532
0.3933
1.1050

- k=4

xtel =

0.7954
-0.1394
-0.5105

-0.0451
0.4547
1.1055

- k=6

  xtel =

  0.7968
  -0.1336
  -0.5031
  -0.0398
  0.4577
  1.1055

- k=20

  xtel =

  0.7975
  -0.1311
  -0.4998
  -0.0374
  0.4591
  1.1055

- k=200

  xtel =

  0.7978
  -0.1299
  -0.4982
  -0.0363
  0.4597
  1.1055

➢ n=20

Για τα συγκεκριμένα αποτελέσματα έχουμε:

X0 =

0.1021
-1.7247
0.8343
-0.5465
1.8452
0.6100
0.6541
-0.3410
-0.8433

-0.7299
-1.2966
-0.8237
1.1819
-0.5983
-0.0596
1.5574
-0.4987
-0.2014
0.6913
0.1994

- k=4
  xtel =

  3.7907
- k=6
  xtel =

  3.7907
- k=20
  xtel =

  3.7907
- k=200
  xtel =

  3.7907

➢ n=200
Για τα συγκεκριμένα αποτελέσματα έχουμε:
X0 =

0.4626
-0.3784
-0.1769
1.6199
2.0215
-0.0344
-0.7741
-0.0908
-0.5774
-0.7186
-1.9744
-1.5501

-0.0294
-0.6471
1.3223
-1.8512
-0.9287
0.5028
0.4690
-0.4402
-1.0034
-2.3712
-0.4758
0.4695
-0.1210
-1.3692
0.9558
0.7370
-0.1904
0.0822
0.0736
0.1784
0.7718
-0.3603
1.5890
-1.1447
-0.3220
-0.8009
-0.6657
-1.6519
1.1657
0.0763
-0.1269
0.0962
-0.7108
-0.3700
0.8974
-0.0666
0.5110
0.6554
1.0282
-0.5350
-0.7453
0.3794
-0.9428

1.1756
-1.0733
1.0916
1.6264
-1.2885
1.1821
0.5190
0.1157
-0.2380
-0.1857
0.8829
-0.6467
-0.6098
3.0903
1.5915
1.0873
0.9580
-0.0254
-1.7396
-0.1910
0.7614
0.6716
1.6044
1.2222
0.4538
-0.5517
-0.2827
0.5106
-0.4710
-0.4448
1.4544
-0.1338
-1.5812
-2.1928
0.2452
-0.4791
-0.3974
-0.1396
0.0372
-1.0191
-0.0253
-2.1957
-0.9339

-0.3942
0.2774
0.2778
1.2668
1.6823
0.4450
-0.8266
0.5333
1.2527
1.1283
0.1025
-1.4556
-1.2887
-0.3661
0.5206
-0.5894
-1.5688
1.3210
-0.4526
1.0004
0.4371
-1.3053
0.0551
0.6739
-0.6857
-0.6943
0.3203
-1.0378
-0.6817
1.3258
-0.3438
0.4547
-0.4888
-1.5437
-0.4784
-0.5615
0.2011
-0.5030
1.0104
-2.0558
0.3890
-0.5322
0.9646

0.0682
1.6319
-0.4085
-0.3175
0.4753
1.0535
-1.2079
-2.6500
-0.1904
0.2489
1.5232
-0.2607
0.0918
0.6705
-0.6185
-1.0140
-0.2299
-1.7766
0.5055
-1.0364
-1.5556
0.0585
0.4784
-0.2345
0.5309
0.7542
0.6819
-0.2079
0.2653
-0.5523
1.5097
-0.7200
-1.1526
-0.6262
-0.6943
-1.2785
-1.2769
1.7293
0.0671
-0.9238
1.0671
-0.9584
1.5342

-0.2916
-0.7687
1.3203
-0.5577
0.0089
0.7272
0.2904
-0.6390
-2.1063
-0.7821
0.6825
1.0764
0.2456
0.9271
0.3228
1.4523

- k=4
  xtel =

  13.0499
- k=6
  xtel =

  13.0499
- k=20
  xtel =

  13.0499
- k=200
  xtel =

  13.0499

Όπως και πριν, η τροποποιημένη μέθοδος του Νεύτωνα χρησιμοποιεί τον Ιακωβιανό πίνακα υπολογισμένο στο X0 σε κάθε επανάληψη, επομένως με την χρήση αυτής της μεθόδου το πρόγραμμα είναι οικονομικότερο όσον αφορά τις πράξεις που εκτελούνται.

## Άσκηση 3

Στην άσκηση 3 χρησιμοποιήθηκε το πρόγραμμα MATLAB.
Για την βελτίωση του προγράμματος της άσκησης 2, ώστε να εκτελούνται όσο το δυνατό λιγότερες πράξεις κινητής υποδιαστολής, δηλαδή να γίνει το πρόγραμμα οικονομικότερο, τροποποιήσαμε της συναρτήσεις «Fn.m» και «JFn.m» και τις αποθηκεύσαμε ως «mFn.m» και «mJFn.m» αντίστοιχα.

Κώδικες:

```
function [f] = mFn(X,n)
a=10/9*ones(n,1);
sum=0;
for i=1:n
    sum=sum+(X(i,1)-a(i,1))^4;
end
A=sum-cos(sum);
d=1+sin(sum);
for i=1:n
    f(i,1)=-(A+n)^(-2)*4*abs(X(i)-a(i))^3*d;
end
end
```

```
function [J] = mJFn(X,n)
a=10/9*ones(n,1);
sum=0;
for i=1:n
    sum=sum+(X(i,1)-a(i,1))^4;
end
A=sum-cos(sum);
d=1+sin(sum);
for i=1:n
    f(i,1)=-(A+n)^(-2)*4*abs(X(i)-a(i))^3*d;
end
for i=1:n
    D=X(i)-a(i);
    for j=1:n
        DD=X(j)-a(j);
        if i==j
            J(i,j)=32*(A+n)^(-3)*D^6*d^2-4*(A+n)^(-2)*D^2*(3*d+4*D^4*cos(sum));
        else
            J(i,j)=16*(A+n)^(-3)*D^3*(DD)^3*(2*d)^2-cos(sum)*(A+n);
        end
    end
end
end
```

Δεν έγινε τροποποίηση της συνάρτησης «NRn.m» ούτε του κώδικα της «NR2n.m», διότι η εκτέλεση της μεθόδου του Νεύτωνα έγινε με ανάποδη διαίρεση, χωρίς της χρήση της εντολής inv().