Budapesti Műszaki és Gazdaságtudományi Egyetem

Méréselmélet

(BMEVIMIM108)

1. Házi feladat

Készítette:

Sipos-Takáts Bence László
T35NOC

Budapest, 2011.03.29

```
%%%%%% 1. Feladat - Wiener-Hopf mátrixok kiszámítása

% Paraméterek

p = 0.62;
r = 0.98;
q = 0.16;
M = 230;
N = 14;

db = 10000;

% Véletlen jel és színusz elõállítása, majd összeadása
% (a véletlen jel ezzel a függvénnyel azonos teljesítményû, mint a sin)

v_0 = normrnd (0, 1/sqrt(2), 1, db);
sin_fv = sin (pi/2.*0:(db-1));

v_n = v_0 + sin_fv;

% Digitálisan szûrjük

u_n = filter ([(1-p^2), 0], [1, 0, -p^2], v_n);

% Szimuláljuk a rendszert

rdsz = tf ([(1-r^2), 0], [1, 0, r^2], 1);
y = lsim (rdsz, u_n);

% R és P mátrixok létrehozása

P = zeros(N,1);
R = zeros(N,N);

R0 = (sum (u_n(N:M+N).*u_n(N:M+N))) / M;

for i = 1:N-1;
    R1(i,1) = sum (u_n(N:M+N).*u_n(N-i:M+N-i)) / M;
end

for i = 1:N
    P(i,1) = sum (u_n(N-(i-1):M+N-(i-1)).*y(N:M+N)') / M;
end


P
R = toeplitz([R0; R1])

% W vektor meghatározása a Wiener-Hopf egyenlettel

W = inv(R)*P
```

```
% Az adaptálandó rendszer és a lineáris kombinátor kimenõjelének
% idõdiagramjának kiszámítása

y_kalap_n = zeros(1,db);
for i = N:db
    y_kalap_n(i) = u_n(i:-1:i-N+1)*W;
end

figure(1)
hold all
plot(y(1:200),'k')
plot(y_kalap_n(1:200),'r')
```

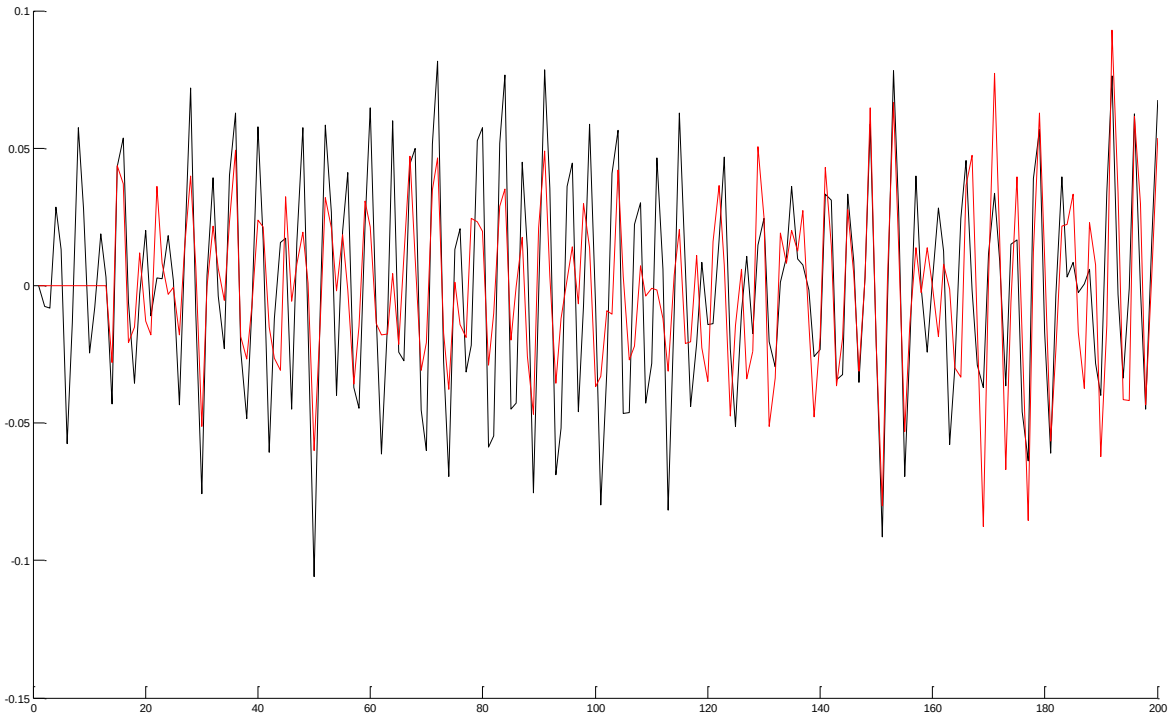
R= (első 7 oszlop)

0,3315	0,0870	0,0002	-0,1476	-0,0791	0,0128	0,1674
0,0870	0,3315	0,0870	0,0002	-0,1476	-0,0791	0,0128
0,0002	0,0870	0,3315	0,0870	0,0002	-0,1476	-0,0791
-0,1476	0,0002	0,0870	0,3315	0,0870	0,0002	-0,1476
-0,0791	-0,1476	0,0002	0,0870	0,3315	0,0870	0,0002
0,0128	-0,0791	-0,1476	0,0002	0,0870	0,3315	0,0870
0,1674	0,0128	-0,0791	-0,1476	0,0002	0,0870	0,3315
0,1190	0,1674	0,0128	-0,0791	-0,1476	0,0002	0,0870
0,0076	0,1190	0,1674	0,0128	-0,0791	-0,1476	0,0002
-0,1152	0,0076	0,1190	0,1674	0,0128	-0,0791	-0,1476
-0,1085	-0,1152	0,0076	0,1190	0,1674	0,0128	-0,0791
-0,0031	-0,1085	-0,1152	0,0076	0,1190	0,1674	0,0128
0,1318	-0,0031	-0,1085	-0,1152	0,0076	0,1190	0,1674
0,1363	0,1318	-0,0031	-0,1085	-0,1152	0,0076	0,1190

(második 7 oszlop)

0,1190	0,0076	-0,1152	-0,1085	-0,0031	0,1318	0,1363
0,1674	0,1190	0,0076	-0,1152	-0,1085	-0,0031	0,1318
0,0128	0,1674	0,1190	0,0076	-0,1152	-0,1085	-0,0031
-0,0791	0,0128	0,1674	0,1190	0,0076	-0,1152	-0,1085
-0,1476	-0,0791	0,0128	0,1674	0,1190	0,0076	-0,1152
0,0002	-0,1476	-0,0791	0,0128	0,1674	0,1190	0,0076
0,0870	0,0002	-0,1476	-0,0791	0,0128	0,1674	0,1190
0,3315	0,0870	0,0002	-0,1476	-0,0791	0,0128	0,1674
0,0870	0,3315	0,0870	0,0002	-0,1476	-0,0791	0,0128
0,0002	0,0870	0,3315	0,0870	0,0002	-0,1476	-0,0791
-0,1476	0,0002	0,0870	0,3315	0,0870	0,0002	-0,1476
-0,0791	-0,1476	0,0002	0,0870	0,3315	0,0870	0,0002
0,0128	-0,0791	-0,1476	0,0002	0,0870	0,3315	0,0870
0,1674	0,0128	-0,0791	-0,1476	0,0002	0,0870	0,3315

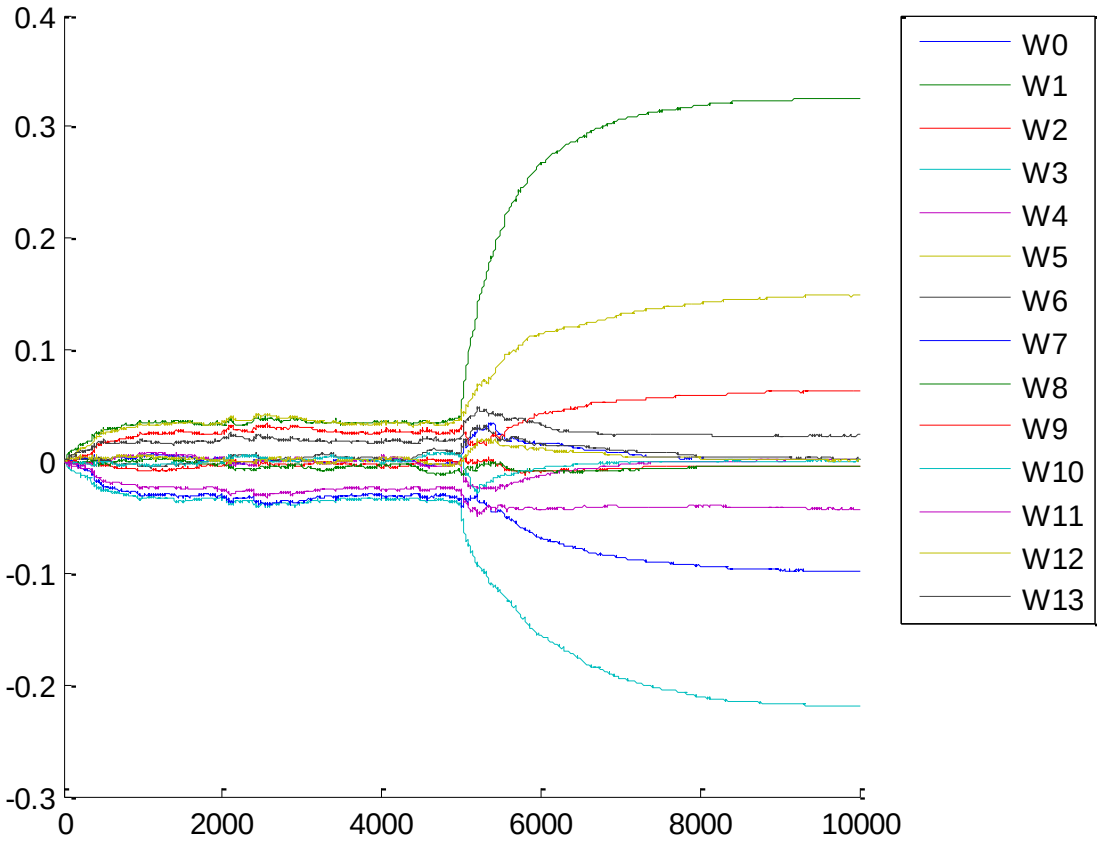
P =	W=
0,0010	0,0049
0,0397	0,0084
-0,0010	-0,0012
-0,0416	-0,0080
-0,0019	-0,0046
0,0402	0,0044
0,0033	0,0048
-0,0363	0,0024
-0,0058	0,0002
0,0297	-0,0017
0,0080	-0,0044
-0,0262	-0,0025
-0,0034	0,0042
0,0205	0,0076



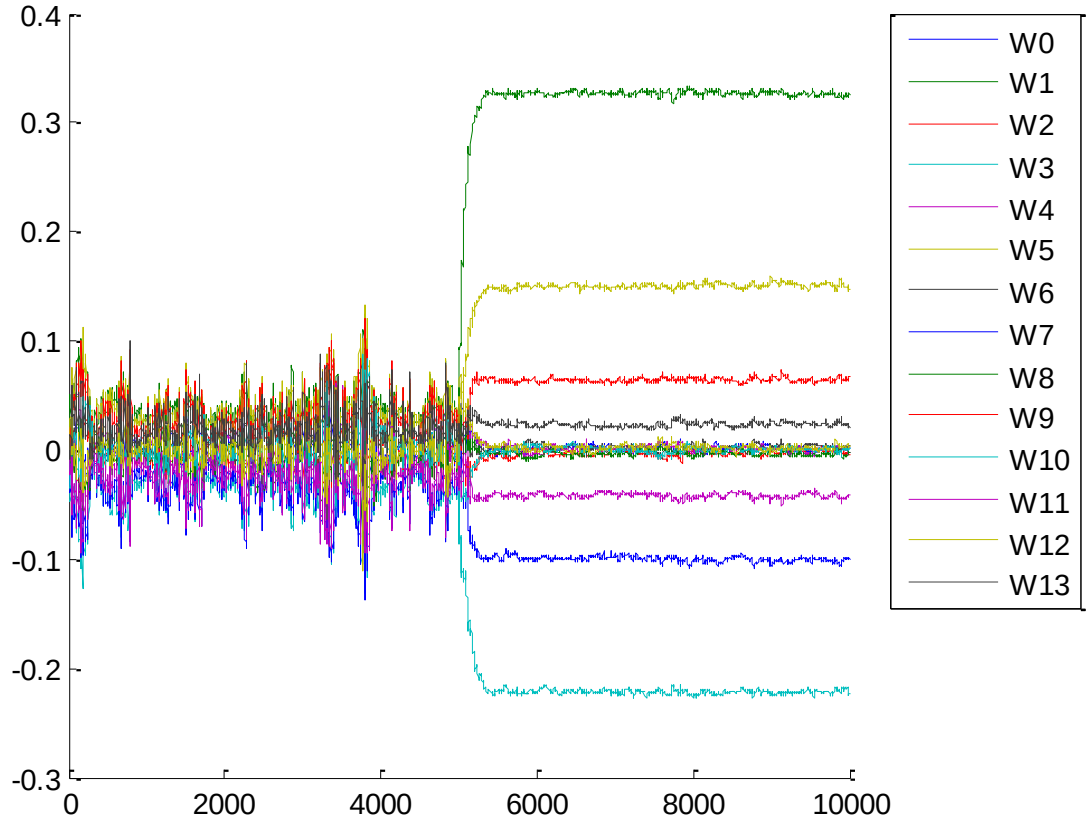
1. ábra A rendszer(fekete) és a lineáris kombinátor(piros) kimeneti jelei

```
%%%%% 2. feladat - LMS eljárás

db2 = db/2;
W_LMS = zeros(N,1);
LMS_X= zeros(1,N);

% Mû megválasztása a 0 < mû < 1/(2*ra) alapján történt, hol az ra az R
% mátrix sajátértékeinek maximuma

mu_LMS = 1/ (100 * max (eig(R)));

% X, y, e megadása, W rekurzív függvénye

for i = N:(db2)
    X = u_n(i:-1:i-N+1);
    y_LMS(i) = X * W_LMS;
    e(i) = y(i) - y_LMS(i);

    W_LMS = W_LMS + 2*mu_LMS * e(i)* X';
    W_LMS_vekt(:,i) = W_LMS;
end

% Innentõl a q-val csökkentett r értékkel futtatjuk ugyanúgy, ahogy elõbb.

rdsz2=tf([1-(r-q)^2 0],[1 0 (r-q)^2],1);
y2 = lsim(rdsz2,u_n);

for i = db2+1:db
    X = u_n(i:-1:i-N+1);
    y_LMS(i) = X * W_LMS;
    e(i) = y2(i) - y_LMS(i);

    W_LMS = W_LMS + mu_LMS * e(i)* X';
    W_LMS_vekt(:,i) = W_LMS;
end

% Konvergencia diagram kirajzolása

figure(2)
hold all
for i = 1:N
    plot (W_LMS_vekt(i,:))
end

legend
('W0','W1','W2','W3','W4','W5','W6','W7','W8','W9','W10','W11','W12','W13',
-1)
```

Ahogy említettem, a bátorsági tényező a 0 < mű < 1/(2*lambda_max) alapján választottam. Az alábbi ábrákon látszik, hogy kicsi mű esetén lassabb, de kisebb hibával történik a beállás, nagy esetén gyorsabban, de nagy hibával.



2. ábra A konvergencia diagram kis mű esetén (LMS)



3. ábra A konvergencia diagram nagy mű esetén (LMS)

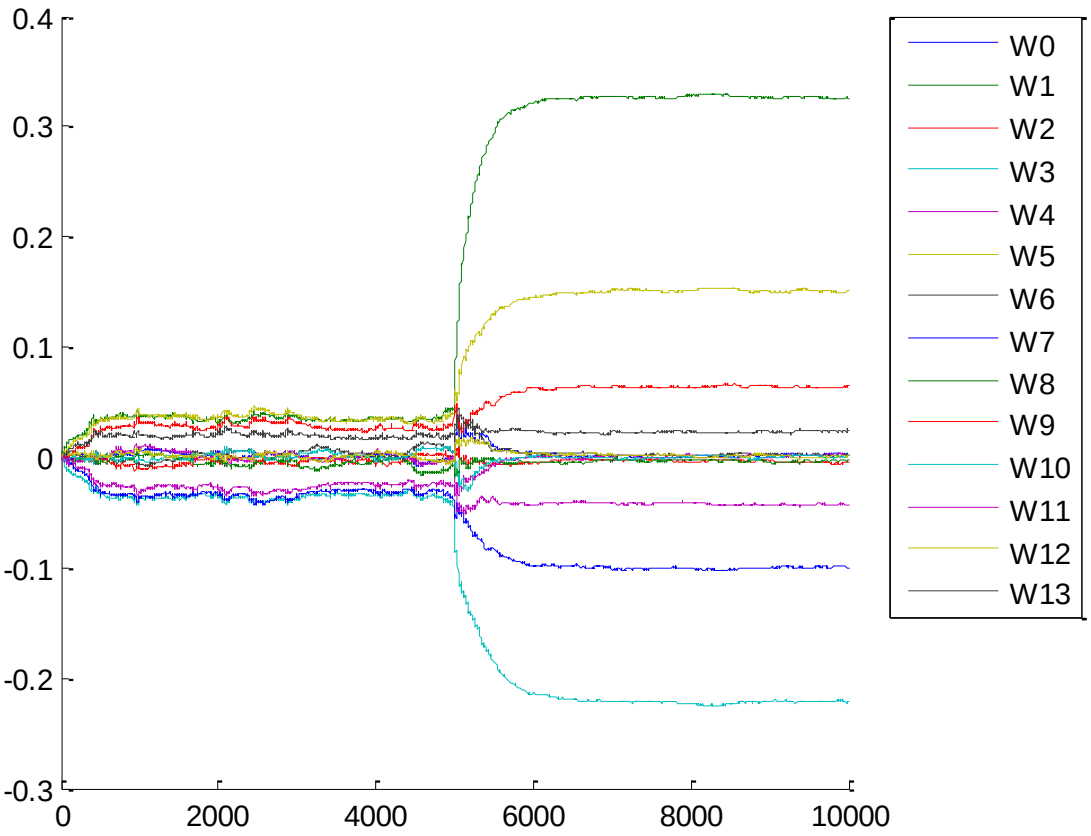
```
%%%%% 3. feladat - alfa-LMS eljárás

W_Alfa = zeros(N,1);
X = zeros(1,N);
k = 25;

% Alfa meghatározása

alfa = 1/(k*u_n*u_n'/db);

% X, y, e megadása, W rekurzív függvénye

for i = N:db2
    X = u_n(i:-1:i-N+1);
    y_Alfa(i) = X * W_Alfa;
    e(i) = y(i) - y_Alfa(i);

    W_Alfa = W_Alfa + alfa*e(i)/(X*X') * X';
    W_Alfa_vekt(:,i) = W_Alfa;
end

% Innentõl a q-val csökkentett r értékkel futtatjuk ugyanúgy, ahogy elõbb.

for i = db2+1:db;
    X = u_n(i:-1:i-N+1);
    y_Alfa(i) = X * W_Alfa;
    e(i) = y2(i) - y_Alfa(i);

    W_Alfa = W_Alfa + alfa*e(i)/(X*X') * X';
    W_Alfa_vekt(:,i) = W_Alfa;
end

% Konvergencia diagram kirajzolása

figure(3)
hold all
for i = 1:N
    plot (W_Alfa_vekt(i,:))
end

legend
('W0','W1','W2','W3','W4','W5','W6','W7','W8','W9','W10','W11','W12','W13',
-1)
```

Itt a bátorsági tényezőt a 0 < alfa < 1/(X'*X) alapján határoztam meg: itt egy *k* szorzó is belekerült, mely a gyorsaságot és pontosságot szabályozza (ugyanúgy, mint az LMS-nél).

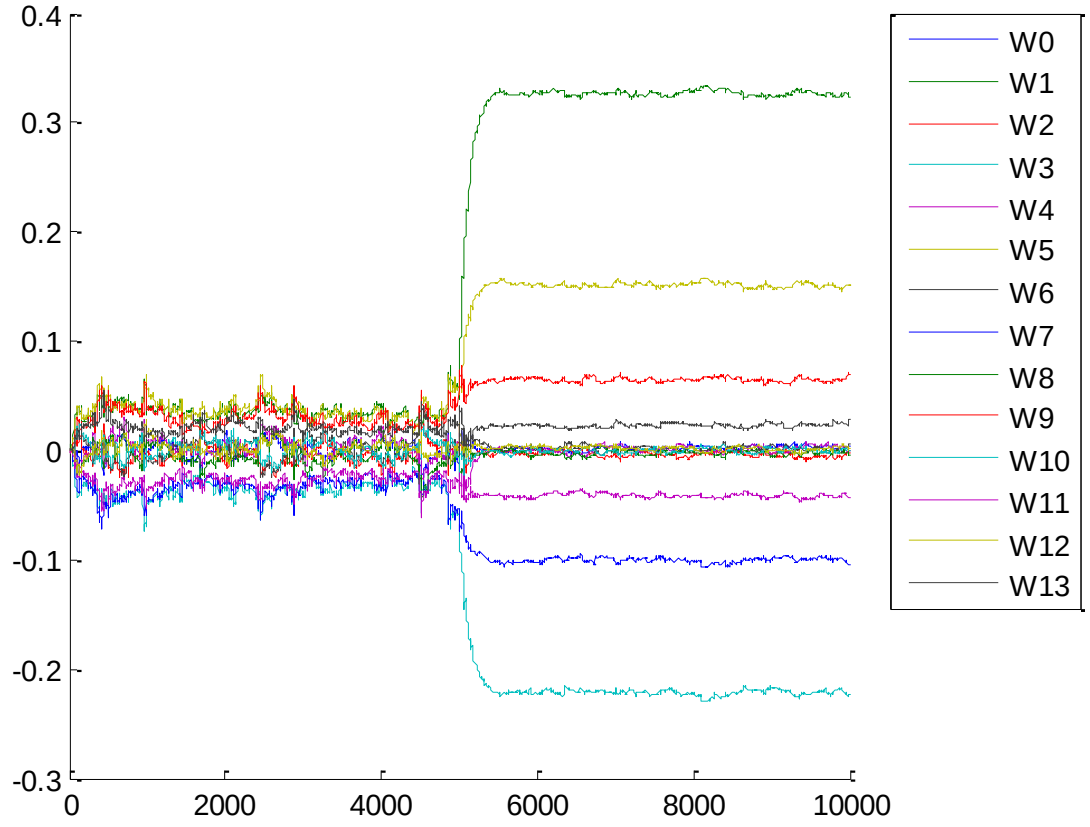
(Hely- és papírtakarékosság miatt a 3-as és 4-es ábra egy lapon található)

```
%%%%% 4. feladat: LMS-Newton eljárás

% Kezdeti értékek

lambda = 0.9;
nu = 0.1;
Rn = eye(N);

W_Newton = zeros(N, 1);

% X, Rn, y, e megadása, W rekurzív függvénye

for i = N:db2

    X = u_n(i:-1:i-N+1);
    Rn = (1/lambda)*(Rn - (Rn*X'*X*Rn / (lambda/nu + X*Rn*X')));

    % mu kiszámítása az aktuális ablakozás szerint

    mu_Newton = 1/(k*X*X');

    y_Newton(i) = X * W_Newton;
    e(i) = y(i) - y_Newton(i);
    W_Newton = W_Newton + 2*mu_Newton*Rn*e(i)*X';
    W_Newton_vekt(:,i) = W_Newton;
end

% Innentõl a q-val csökkentett r értékkel futtatjuk ugyanúgy, ahogy elõbb.

for i = db2+1:db

    X = u_n(i:-1:i-N+1);
    Rn = (1/lambda)*(Rn - (Rn*X'*X*Rn / (lambda/nu + X*Rn*X')));

    mu_Newton = 1/(k*X*X');

    y_Newton(i) = X * W_Newton;
    e(i) = y2(i) - y_Newton(i);
    W_Newton = W_Newton + 2*mu_Newton*Rn*e(i)*X';
    W_Newton_vekt(:,i) = W_Newton;
end

% Konvergencia diagram kirajzolása

figure(4)
hold all
for i = 1:N
    plot (W_Newton_vekt(i,:))
end

legend
('W0','W1','W2','W3','W4','W5','W6','W7','W8','W9','W10','W11','W12','W13',
-1)
```

Itt a bátorsági tényezőt szintén a 0 < alfa < 1/(X'*X) alapján határoztam meg, azonban itt nem az összes bemenő értékre, hanem csak az aktuális N értékre (ablakozás).



4. ábra A konvergencia diagram (α-LMS)



5. ábra A konvergencia diagram (Newton-LMS)

```
%%%%% 5. feladat: Értékelés
```

W=	W_LMS=	W_Alfa=	W_Newton=
0,0048	0,0013	0,0002	-0,0021
0,0321	0,3257	0,3277	0,3289
-0,0099	-0,0026	-0,0005	0,0037
-0,0321	-0,2197	-0,2219	-0,2240
0,0103	0,0017	0,0020	0,0006
0,0309	0,1494	0,1524	0,1567
-0,0078	0,0017	0,0015	0,0019
-0,0281	-0,0989	-0,1015	-0,1040
0,0066	-0,0025	-0,0023	-0,0029
0,0237	0,0631	0,0659	0,0686
-0,0078	-0,0002	-0,0005	-0,0002
-0,0207	-0,0421	-0,0439	-0,0458
0,0073	0,0024	0,0028	0,0060
0,0164	0,0239	0,0247	0,0261

Kapott súlytényezők összefoglalása

A házi feladat során megismerkedtem a Wiener-Hopf és a különböző LMS eljárásokkal. Megismerkedtem a bátorsági tényezők szerepével és fontosságával: kis bátorsági tényező esetén lassabb a beállás, hiszen kisebb lépésenként közelít, viszont pontosabb. Ezzel szemben a nagy tényező esetén gyorsabb, de nagyobb a hiba. Érdemes végiggondolni a megfelelő értéket, hiszen ha nagyon kicsi tényezőt választunk, a pontosság nem kompenzálja a beállás lassúságát, ugyanígy nagyon nagy bátorsági tényező esetén hiába lesz gyors, nagy a hiba.

A különböző eljárások kisebb-nagyobb hibával hasonlóképpen követték a jelet, azonban – mint már mondtam – nagyon sok múlik a bátorsági tényezőn.