

7. GYAKORLAT

**LIN. ALGEBRAI ALKALMAZÁSOK 2,
REGRESSZIÓ, VIZSGA MINTA****SAJÁTÉRTÉK, SAJÁTVEKTOR****Mintafeladat**

Határozzuk meg az $A = \begin{bmatrix} 5 & 1 & -2 & 1 \\ 1 & 6 & -1 & 1 \\ -2 & -1 & 4 & 2 \\ 1 & 1 & 2 & 8 \end{bmatrix}$ szimmetrikus mátrix sajátvektorait és sajátértékeit! Ellenőrizzük a sajátértékegyenlet teljesülését is!

```
>> [U L] = eig(A), ellenorzes = A*U - U*L
```

Határozzuk meg az A mátrix karakterisztikus polinomját! Helyettesítsük be a sajátértékeket a karakterisztikus polinomba, és így ellenőrizzük, hogy ezek valóban kielégítik a karakterisztikus egyenletet!

Tipp: A polyval parancs polinom kiértékelésére szolgál, adott helyen/helyeken.

```
>> p = poly(A), l = eig(A), polyval(p, l)
```

Feladat

Legyen $B = \begin{bmatrix} 6 & -1 \\ 8 & 2 \end{bmatrix}$. Valósak-e a B mátrix sajátértékei?

Tipp: Használjuk a megoldáshoz a poly és a roots párost, ill. az eig parancsot!

Az eig parancs segítségével készítsük el a mátrixhoz az U és Lambda mátrixokat úgy, hogy U oszlopvektorai az 1-re normált sajátvektorokat adják, a Lambda mátrix főátlóbeli elemei pedig a sajátértékeket. Ellenőrizzük, hogy ekkor $U \cdot \text{Lambda} \cdot \text{inv}(U)$ az eredeti mátrixot adja!

EGYSZERŰ REGRESSZIÓ

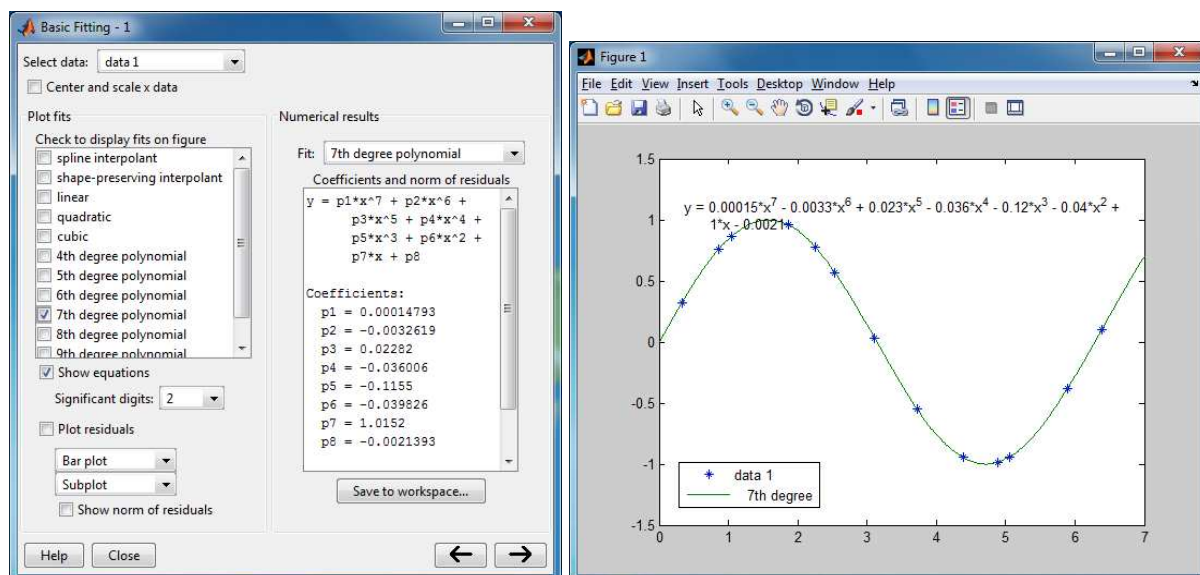
A Matlab sokféle támogatást nyújt regressziós feladatok megoldásához (lásd előadás).

A legegyszerűbb esetben adott x, y pontsorozathoz kérhető polinomos közelítés a **plot** menüjében.

Ebben a mintafeladatban egy szinusz függvényt követő pontsorozathoz készítjük el a regressziós polinomot.

```
>> x = (0:0.5:6)+0.4*rand(1,13)    % x sorozat  
>> y = sin(x)                      % y sorozat  
>> plot(x,y,'*')
```

A Figure 1 ablak Tools/Basic Fitting menüpontjában kérjük ki a 7-edfokú közelítést.



Ezután jelöljük be, hogy látszódjék az ábrán az egyenlet is, ill. a \rightarrow lenyomása után láthatjuk a numerikus értékeket is.

Megállapítható, hogy az eredmény nagyon hasonló a $\sin(x)$ Taylor-sorához.

MÉRÉSI ADATOK FELDOLGOZÁSA, ELEMZÉSE (VIZSGA MINTA)

Ebben a kidolgozott mintapéldában egy olyan komplex gyakorlati feladatot oldunk meg, amely a Matlab modul zárásaként összegzi tudásunkat. A megoldás *során néhány haladóbb ötletet is alkalmazunk és bemutatunk, ezek azonban nem tartoznak bele a tantárgy törzsanyagába és nem kerülnek számonkérésre sem.*

A feladatban egy rögzített rugalmas test nyúlásának mérési adatait kell feldolgoznunk, a hibás adatok kiszűrésével, statisztikákkal, regresszióval. A megoldás része a megfelelő modell felállítása is (ellenőrzéssel).

1. Adatoszlopok kinyerése, ábrázolás

Egy egyszerű szövegszerkesztővel nyissuk meg az adatfájlt (gyak-fel-adatok.txt), és tanulmányozzuk a struktúráját. Látható, hogy 5 adatoszlopunk van, amelyek rendre a megnyúlás (l) és az erő (F) értékeit tartalmazzák (1. és 2., ill. 3. és 4. oszlopok), továbbá egy azonosítót (5. oszlop), összesen 999 adatsorral. A fejléc számunkra most felesleges, ezért töröljük. Mentsük el a fájlt (data.txt).

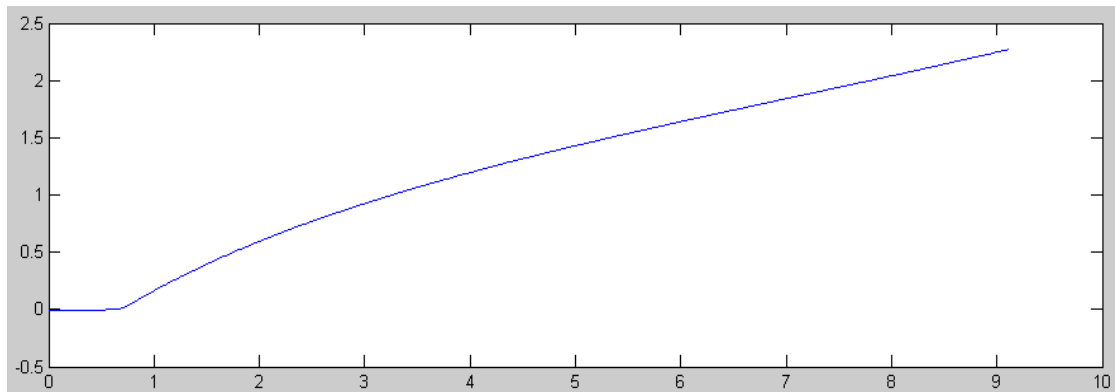
```

1 Mért adatok
2
3 max N
4
5 0 0 0 -0.0072 1676
6
7 0.00625 0.0012 0.00625 -0.0066 1675
8
9 0.0125 0.0036 0.01875 -0.0066 1674

```

Ebben a feladatban a 3. és a 4. oszlop adataival fogunk tovább dolgozni. Olvassuk be az adatokat a Matlabban egy l és egy F vektorba, majd ábrázoljuk az erőt a megnyúlás függvényében!

```
>> load data.txt, l = data(:,3); F = data(:,4);  
>> plot(l,F)
```



Az $F(l)$ függvény grafikonját vizsgálva azt találjuk, hogy három részre bontható fel.

Az első szakasz – feltehetően a befogás és a mérés technikai korlátai miatt – még nem ad reális képet a fizikai jelenségről („inicializálás”, kb. az első 70 adat). Ezután egy exponenciális/logaritmikus jellegű rész következik, majd egy lineáris szakasz figyelhető meg (a Hooke-törvény szerint).

Minket a feladat „standard” megoldása során elsősorban a lineáris rész érdekel majd, a fejlettebb (fakultatív) megoldásnál elemezzük a nem lineáris részt.

2. Osztáspontok vizsgálata

Az adatokat áttekintve láthatjuk, hogy jól érzékelhető szabályosság figyelhető meg a megnyúlási értékek sorozatában. Vizsgáljuk meg a Matlabbal, hogy vajon egyenlőközűnek vagy közel egyenlőközűnek tekinthetők-e az osztáspontok!

Elkészítjük a szomszédos l értékek differenciáit (távolságát).

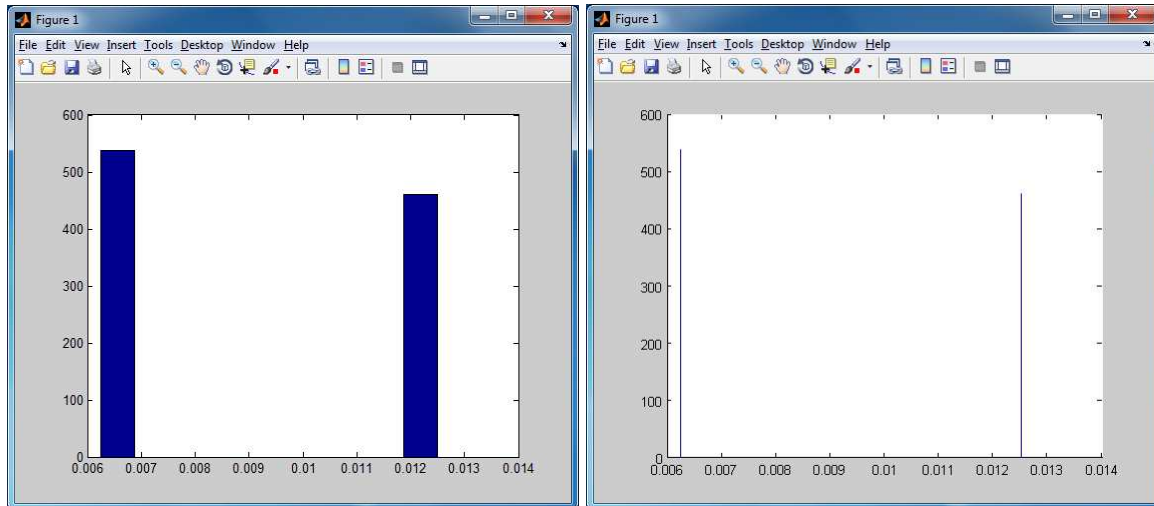
```
>> dl = diff(l); % ; nélkül is megnézhetjük, de ...
```

Ezután hisztogramot (gyakoriság oszlopdiagramot) kérünk az adatokra, hogy áttekinthessük a struktúrát.

```
>> hist(dl)
```

A grafikonon látjuk, hogy az adatsorban csak 0,0065 és 0,012 körüli értékek szerepelnek.

Módosított hisztogram (az összes eset darabszámával): `hist(dl,999)`, erről már jól látjuk, hogy valójában csak pontosan két érték fordul elő (0,00625 és 0,0125).



*Pontos ellenőrzés és számolás is elvégezhető:

```
>> db = 0; for i = 1:998 if (dl(i) > 0.0125 - 5*eps &
dl(i) < 0.0125 + 5*eps) db = db + 1; end; end; db
% 5*eps nélkül nem lenne jó
```

Ha ugyanezt megcsináljuk a másik értékre is, akkor látjuk, hogy 460, ill. 538 előfordulás van. Az ellenőrzés végrehajtható a `h = hist(dl)`, ill. `h = hist(dl, 999)` parancsokkal is.

*3. A grafikon részekre bontása

Most megkeressük az $F(l)$ diagram egyenesnek tűnő szakaszát.

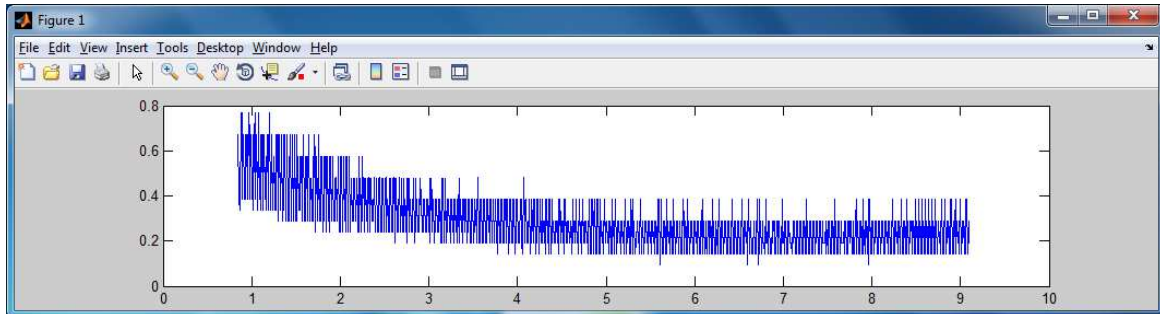
Ehhez az `axis([xkezd xvég ymin ymax])` szerkezetű paranccsal ablakot nyitunk a diagramra és ezt végigtoljuk lépésenként.

```
for bal = 1:900 ...
    jobb = bal + 99; ...
    axis([l(bal) l(jobb) min(F(bal:jobb)) max(F(bal:jobb))]), ...
    pause(0.02), ...
end
```

Az elejét még egyszer, lassabban is végignézzhetjük (írjuk át a 900-as értéket 200-ra, a pause paraméterét pedig 0,2-re).

Úgy tűnik, hogy a 200. esettől már közel lineáris a kapcsolat. Ezt azzal is ellenőrizhetjük, hogy ettől kezdve a differenciahányadossal becsült derivált közel konstans lesz.

```
>> l_orig = l; F_orig = F; l = l_orig(100:end); F = F_orig(100:end);
% lecsípjük a megfelelő részeket (100-tól)
>> dF = diff(F); dl = diff(l); deriv_F = dF./dl;
% derivált sorozat legyártása
>> deriv_F(end+1) = deriv_F(end);
% kiegészítjük a sorozatot az utolsó elem másolatával, hogy a deriv_F
és l sorozatok elemszáma azonos legyen (ábrázoláshoz)
>> plot(l,deriv_F)
```



Feltevésünkkel ellentétben az $F'(l)$ deriváltfüggvény kb. 4,2-ig csökkenő tendenciát mutat, azaz eddig a függvény biztosan nem lineáris. A deriváltra ezen a szakaszon a következő közelítő becslést adjuk (indoklás: a függvény alakja alapján látjuk, hogy negatív kitevős exponenciális típusú, megfelelő eltolással):

$$(1) \quad F'(l) = a \cdot e^{-b \cdot (l-c)} + d,$$

ahol d az egyenes rész meredeksége az $F = F(l)$ kapcsolatban. Mivel egy exponenciális függvény határozatlan integrálja szintén exponenciális függvény, ezért az $F(l)$ függvény jó közelítéssel

$$(2) \quad F(l) = A \cdot e^{-b \cdot (l-c)} + d \cdot l + e$$

alakú lehet. (Itt kihasználtuk azt a becslésnél szokásos technikát, hogy könnyebb a deriváltból következni az eredeti függvényre, mint direkt módon becsülni az eredetit.)

A továbblépésre ezután két lehetőségünk van.

a./ Csak az egyenesnek látszó részen dolgozunk, azaz az adataink második felén, kb. az 500. ponttól kezdve.

*b./ Az exponenciális rész további vizsgálatához a (2) összefüggés A, b, c, d, e paramétereit a négyzetes eltérések minimalizálásával keressük meg.

(A mért adatsor és a becsült függvényértékek eltérései négyzetének összegét minimalizáljuk az `fminsearch()` függvény segítségével.)

4. Regresszió a lineáris részre

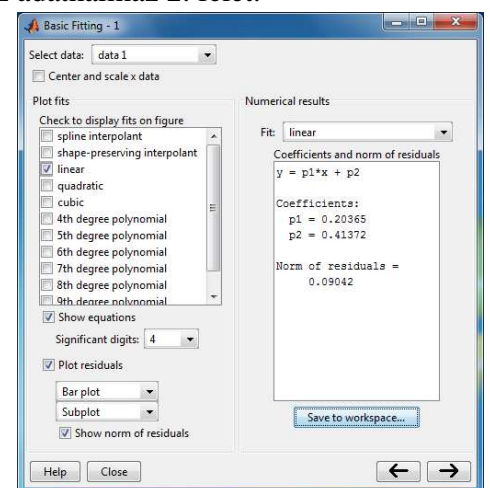
Töröljük a változóinkat, újra betöltjük az adatokat, majd vesszük az adathalmaz 2. felét.

```
>> clear all, load data.txt
>> l_orig = data(:,3); F_orig = data(:,4);
>> l = l_orig(500:end);
>> F = F_orig(500:end); plot(l,F)
```

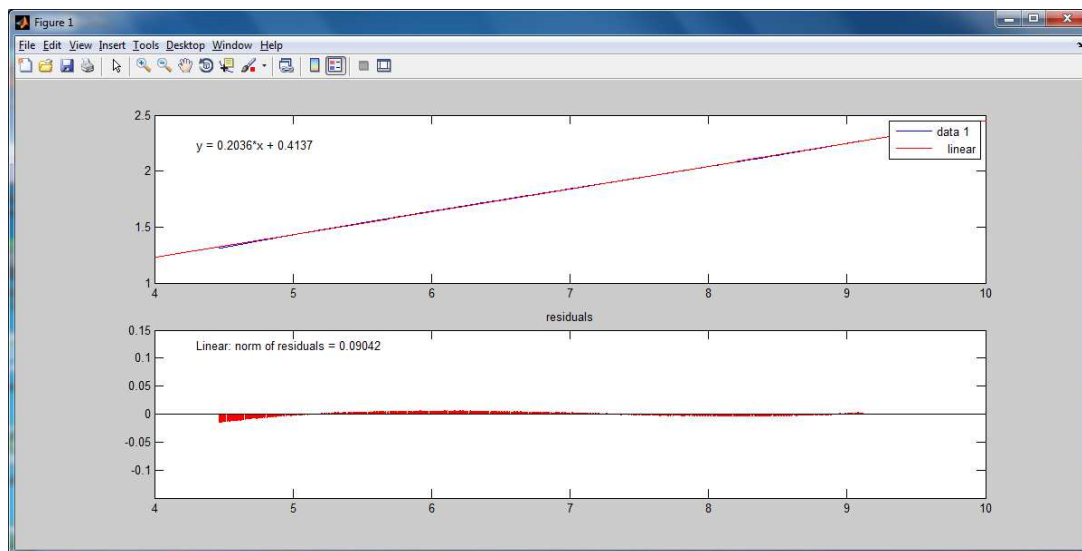
A regressziós feladat megoldását a már ismert eszközzel végezzük (Tools/Basic Fitting).

Az ablakban válasszuk ki, ill. jelöljük be a következő opciókat:

- Lineáris kapcsolat,
- Show equations,
- Plot residuals,
- Show norm of residuals.



A „Significant digits” értékét növeljük meg 4-re. A → gomb megnyomásával nyissuk ki a „Numerical results” panelt is. Itt lehetőség van az illeszkedés adatainak mentésére, ill. direkt módon is kimásolhatjuk a megfelelő változókat.

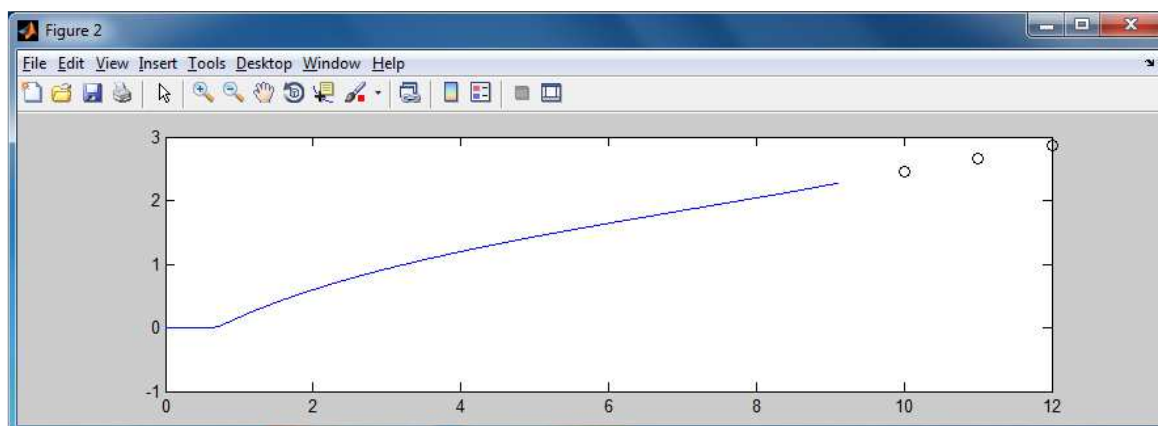


Másoljuk be a parancsablakba a megfelelő paramétereket:

```
>> p1 = 0.20365, p2 = 0.41372
```

Ezután már az $y = p1 \cdot x + p2$ képlettel további x helyeken is tudjuk az F függvény értékét becsülni.

```
>> x = [10 11 12]; y = p1*x + p2; ertektabla=[x; y]
% 3 új pontot hozunk létre
>> figure(2), plot(l_orig,F_orig), hold on, plot(x,y,'ok')
% új ábra az új pontokkal (teljes adatsor)
```

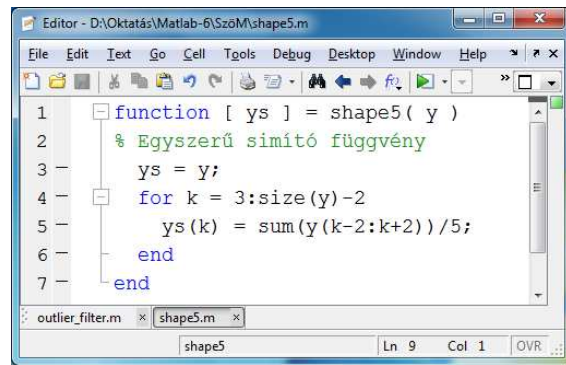


5. Adatok simítása és szűrése

A mért adatsorozat feldolgozásának fontos lépése a *simítás* (egyenetlenségek csökkentése) és ennek részeként egyes feltűnően kiugró adatok törlése (ezek feltehetően mérési hibából származnak).

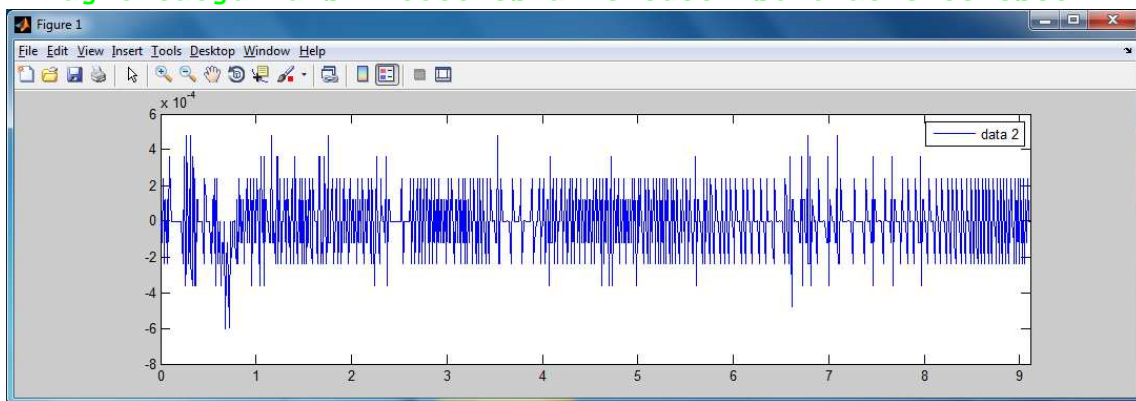
*Ha elérjük a Signal Processing Toolbox szolgáltatásait, akkor a simítási feladat megoldására igen sok eszköz áll rendelkezésünkre. (Pl. egy jól bevált módszer a Savitzky–Golay-féle simítás – lásd előadás slideok.)

Ha csak a Matlab alapszolgáltatásai használhatóak, akkor egyszerű simításként alkalmazhatjuk pl. az 5-pontos csúszóátlag módszert, amely 5 egymás melletti érték átlagával helyettesíti a középsőt. Erre készítettünk egy saját függvényt (shape5.m).



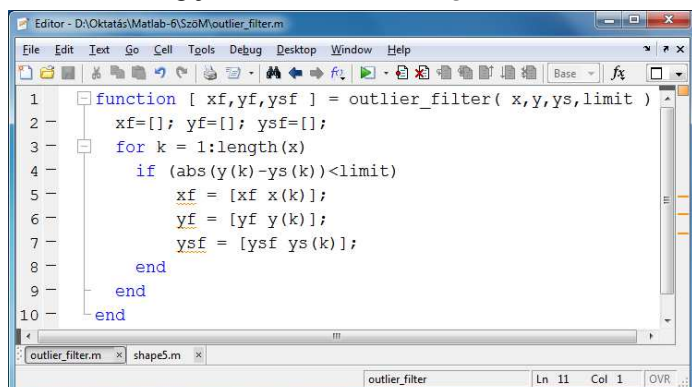
A feladatunk megoldása ezt használva:

```
>> l = l_orig; F = F_orig;  
>> Z = shape5(F); plot(l,F-Z)  
% kirajzoltatjuk a simított és az eredeti sorozat eltérését
```



Ha a mért adatsor és a simított adatsor közötti különbség jelentős, akkor a *kiugró eseteket érdemes törölni*, mint extra zajból, vagy mérési hibából származó értékeket.

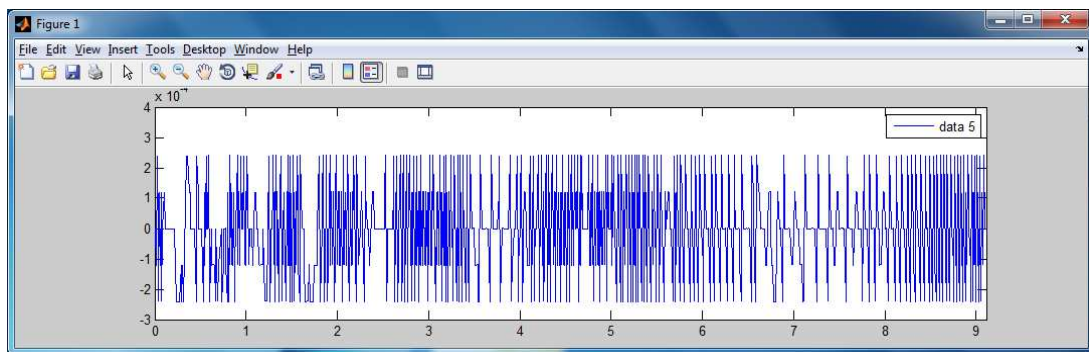
Erre a feladatra szintén egy saját függvényt készítettünk (outlier_filter), amely paraméterként megkapja az eredeti adatsor (x, y) mellett a simított értékek sorozatát is (ys), és az $x(k)$, $y(k)$, $ys(k)$ adathármasokból csak azokat hagyja meg, ahol az eltérés a szintén megadott limitnél



kisebb. Az eredmények rendre az xf , yf , ysf sorozatokba kerülnek, amelyeket üres sorozatokból kiindulva készítünk el.

A függvény segítségével most lecsípjük a $3E-4$ -nél nagyobb hibájú adatokat:

```
>> limit = 3E-4;  
>> [ls,Fs,Zs] = outlier_filter(l,F,Z,limit); plot(ls,Fs-Zs)  
% az eredeti és a simított adatsor eltérésének kirajzoltatása a  
korrekció után
```



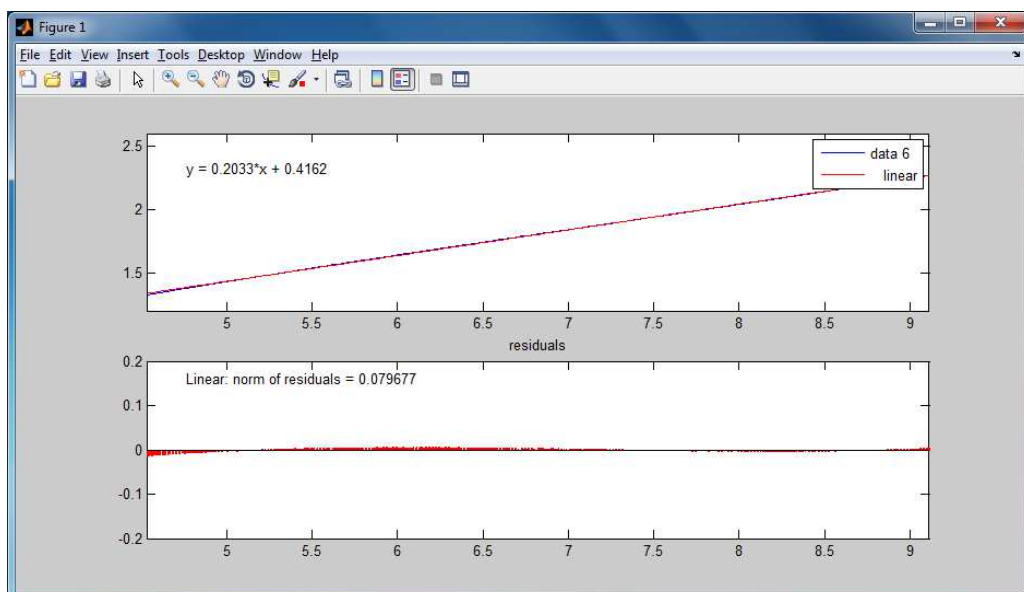
Nézzük meg, hogy az eljárás hány adatot törölt.

```
>> length(ls)
```

Látjuk, hogy 939 darab adatunk maradt.

Ezután ábrázoljuk a redukált adatsorozat 2. felét, és az ábrára a megszokott módon lineáris regressziót kérünk (Basic Fitting).

```
>> l = ls(470:end); F = Fs(470:end); plot(l,F)
```



A parancsablakba most is bemásolhatók a megfelelő paraméterek:

```
>> p1 = 0.20332, p2 = 0.4162
```

Szintén a fentihez hasonló módon előrejelzést is végezhetünk az $y = p1 \cdot x + p2$ képlettel.

```
>> x = [10 11 12]; y = p1*x + p2; ertektabla=[x; y]
>> figure(2), plot(l_orig,F_orig), hold on, plot(x,y,'ok')
```

Összefoglalva, látható, hogy a simítás és a pár kiugró pont törlése kevésbé módosította a korábbi eredményt, hiszen a sorozat eléggé lineáris volt. (A Signal Processing Toolbox parancsait használva is nagyon hasonló eredmény adódik.)

*6. Általános regresszió, paraméterbecslés a legkisebb négyzetek módszerével

Ebben a részben egy olyan általános regressziós eljárást mutatunk be, amely a mostani konkrét feladat mellett sok más probléma megoldására is használható. Csak annyi a teendőnk, hogy a mintamegoldás megfelelő sorait átírjuk (ezek most piros kommentezésűek)!

Célunk a feladatnak megfelelően az exponenciális rész regresszióval történő becslése.

Az eredeti adatsor 100. pontjánál kezdjük az illesztést.

```
>> clear all
>> load data.txt
>> l = data(:,3); F = data(:,4); xdata=l(100:end); ydata=F(100:end);
% adatelőkészítés
```

Következik az $A \cdot e^{-b(l-c)} + d \cdot l + e$ típusú illesztés végrehajtása, a fitc függvény hívja meg az fminsearch parancsot.

```
>> [estimates, model] = fitc(xdata,ydata) % a kapott paraméterek
>> est_fun = [num2str(estimates(1),4), '*exp(-', num2str(estimates(2),4), '(x-', ...
num2str(estimates(3),4), '))+ ', num2str(estimates(4),4), '*x+', num2str(estimates(5),4)]
% a képletünk karakterlánca szövegdarabkákból összeragasztva
>> [sse, FittedCurve] = model(estimates);
% FittedCurve a becsléssel kapott értéksorozat
>> sse % minimalizált négyzetes eltérés
>> R_square = min(min(corrcoef(ydata,FittedCurve).^2))
% determinációszám együttható
```

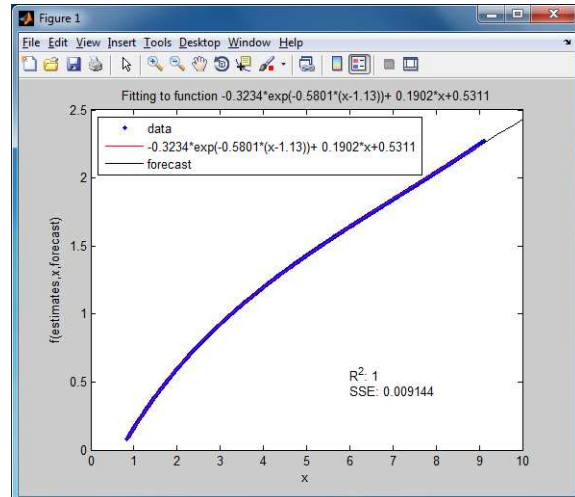
Ezután kirajzoljuk az adatokat és a regressziós függvényt, majd előrejelzést is készítünk.

```
>> plot(xdata, ydata, '.'), hold on
>> plot(xdata, FittedCurve, 'r')
>> xlabel('x'), ylabel('f(estimates,x,forecast)')
>> title(['Fitting to function ', est_fun]);
>> x = [9:0.2:10]; y = inline(est_fun), plot(x, y(x), 'k')
% előrejelzés a [9, 10] tartományban
>> legend('data', est_fun, 'forecast', 2)
>> text(6,0.5, {[ 'R^2: ', num2str(R_square,4)]}; {[ 'SSE: ', num2str(sse,4)]})
% R-négyzet pozicionálását kell átírni megfelelő helyre
>> hold off
```

Az eredményt elemezve látható, hogy az illesztésünk R-négyzet értéke (determinációs együtthatója) gyakorlatilag 1, azaz az illesztés szinte tökéletes. Így a lineáris tag most kapott együtthatója az anyag fizikai viselkedését leíró rugalmassági együttható: 0.1902.

Megjegyezzük, hogy ez pontosabb a lineáris regresszióval kapott értéknél!

Végül közöljük a `fitc()` függvény definícióját is. Itt látunk példát függvénydefiníciók egymásba ágyazására és tetszőleges darabszámú paraméter átvételére (params).



```
function [estimates, model] = fitc(xdata, ydata)
% Call fminsearch with starting point.
start_point = [-1, 0.5, 1, 0.2, 0.4];
% saját alkalmas kezdőpontok A,b,c,d,e-re
model = @expfunc; % itt tesszük hozzáférhetővé a model függvényt
estimates = fminsearch(model, start_point);
% expfunc accepts curve parameters as inputs, and outputs sse,
% the sum of squares error for
% A*exp(-b*(xdata-c)) + d*xdata + e
% and the FittedCurve. FMINSEARCH only needs sse, but we want
% to plot the FittedCurve at the end.
function [sse, FittedCurve] = expfunc(params)
    A = params(1); % 1. paraméter
    b = params(2); % 2. paraméter
    c = params(3); % 3. paraméter
    d = params(4); % 4. paraméter
    e = params(5); % 5. paraméter
    FittedCurve = A.*exp(-(b * xdata-c))+d*xdata+e;
    % a függvény definíciója
    ErrorVector = FittedCurve - ydata;
    sse = sum(ErrorVector.^ 2);
end
end
```

OTTHONI MUNKA

Feladat

Határozzuk meg a szokásos módon az $A = \begin{bmatrix} -4, & -3, & -7; & 2, & 3, & 2; & 4, & 2, & 7 \end{bmatrix}$ mátrix sajátvektorait és sajátértékeit! (S és L mátrixok.)

- a./ Igaz-e, hogy az $S^{-1}AS$ mátrix diagonálmátrix és a főátlójában éppen a sajátértékeket tartalmazza?
- b./ A sajátértékeknek a karakterisztikus polinomba való behelyettesítésével ellenőrizzük, hogy a sajátértékek kielégítik a karakterisztikus egyenletet!

Feladat

Valós szimmetrikus mátrix minden sajátértéke valós és a sajátvektorainak mátrixa ortogonális mátrix, azaz bármely két különböző sajátvektor skaláris szorzata nulla. Ellenőrizzük ezt az $A = \begin{bmatrix} -3 & 1 & -1 \\ 1 & -5 & 1 \\ -1 & 1 & -3 \end{bmatrix}$ mátrixra!

Feladat

Próbáljunk ki tanult alapstatisztikákat a mérési adatsorra!

