- Least Square estimator

(This is one of the another topic considered in Machine Learning)

1. Pproblem : Consider a car is moving on the road. A policeman measure the position of the car as

$$(t_1, x_1), (t_2, x_2), \ldots, (t_n, x_n)$$

here t_k, x_k are the sampled time and the measured position. If the movement is considerd to be constant velocity,

$$x = x_0 + \mathrm{vt}$$

since the measurement is corruped by a noise, what is the best estimator of x_0, v?

2. Model:

In matrix form, it will be

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} 1 & t_1 \\ 1 & t_2 \\ \vdots & \vdots \\ 1 & t_n \end{bmatrix} \begin{bmatrix} x_0 \\ v \end{bmatrix}$$

which is

$$X = \mathrm{AE}$$

so that given X, A find *E*. It may be unique *E*, so it is estimated in some sense as

$$\underset{\left(\tilde{x0}, \tilde{v}\right)}{\arg\min} \sum_{i=1}^{n} \left(x_i - \left(\tilde{x_0} + \tilde{v} t_i \right) \right)^2$$

The solution is

$$(\hat{x_0}, \hat{v}) = (A^T A)^{-1} A^T X$$

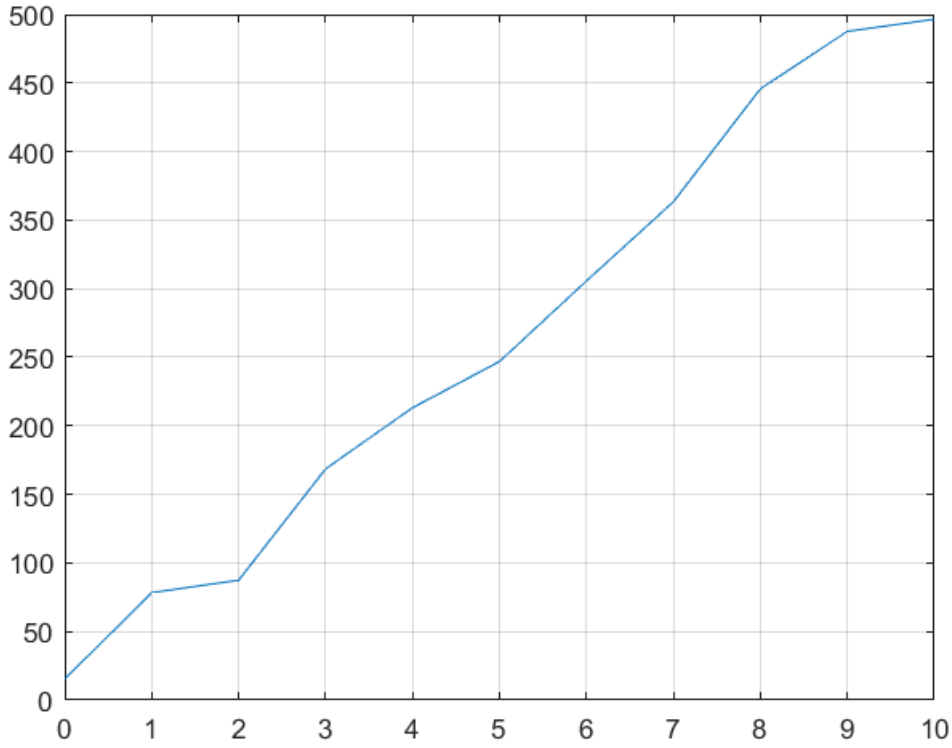
which is called as the least square estimator (LS estimator).

```
clear all;clf
rng('default')
n = 11;    % sample number
t = linspace(0,10,n)';
x0 = 10;
v0= 50;
x = x0 + v0.*t + 10*randn(n,1)
```

```
x = 11×1
   15.3767
   78.3389
   87.4115
  168.6217
```

```
  213.1877
  246.9231
  305.6641
  363.4262
  445.7840
  487.6944
     ⋮
```

```
plot(t,x); grid on
```




```
% generate the A
for i = 1:n
    A(i,1)=1;
    A(i,2)=t(i);
end
% the LS estimator of x0 , v0
est=inv(A'*A)*A'*x
```

```
est = 2×1
    9.8914
   50.9113
```

which is similar to the real values. One of the problems is the modeling. Consider the car is moving with a constant acceleration as

$$x = x_0 + v_0 t + \frac{1}{2} \mathrm{at}^2$$

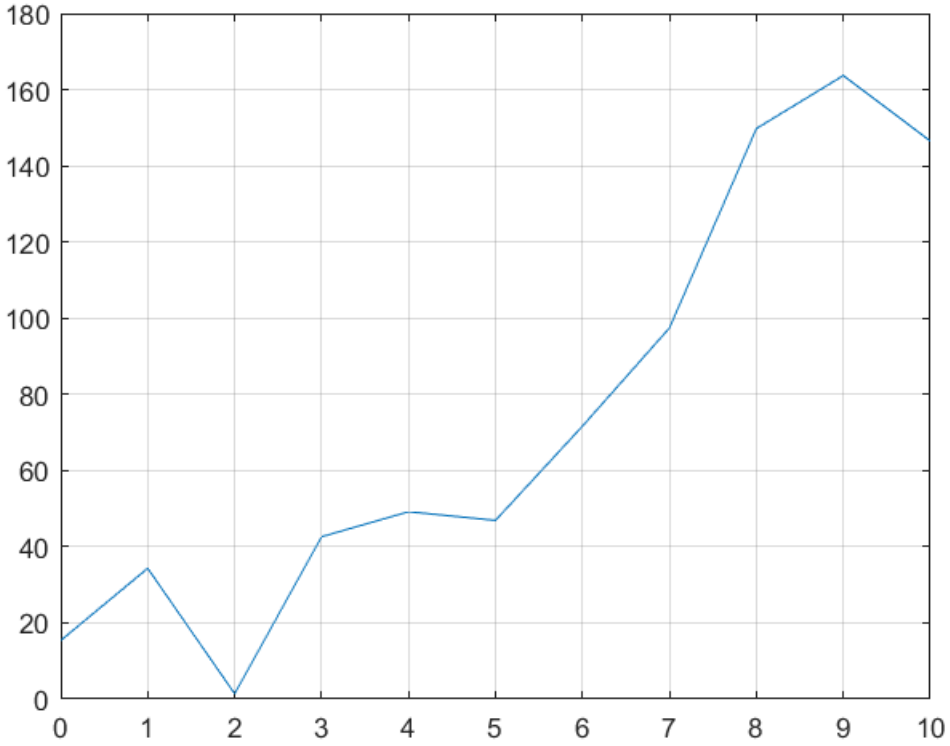
Now assum we may consider the car is moving with a constant acceleration as,

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} 1 & t_1 & \frac{1}{2}t_1^2 \\ 1 & t_2 & \frac{1}{2}t_2^2 \\ \vdots & \vdots & \vdots \\ 1 & t_n & \frac{1}{2}t_n^2 \end{bmatrix} \begin{bmatrix} x_0 \\ v \\ a \end{bmatrix}$$

```
clear all;clf
rng('default')
n = 11;
t = linspace(0,10,n)';
x0 = 10;
v0 = 5;         % initial velocity
a = 2;
v = v0 + a*t;
x = x0 + v0*t+ 1/2*a*t.^2 +10*randn(n,1)
```

```
x = 11×1
   15.3767
   34.3389
    1.4115
   42.6217
   49.1877
   46.9231
   71.6641
   97.4262
  149.7840
  163.6944
      ⋮
```

```
plot(t,x); grid on
```




```
% generate the A
for i = 1:n
    A(i,1)=1;
    A(i,2)=t(i);
    A(i,3)=1/2*t(i)^2;
end
% the LS estimator of x0, v0 and a
est=inv(A'*A)*A'*x
```

```
est = 3×1
   13.5625
    3.4639
    2.4895
```

Here if you model the car's movement with a constant velocity, then the total errors are bigger than the with a constant acceleration. So how to choose? There are no loyal road to select it. If you model more varaibles, it may be overfitted.It is upto you.

- Non-linear least square estimator.

1.Problem

$$f(a,b,x) = a + ab + bx$$

This is a linear w.r.t the variable *t*, but non-linear w.r.t the estimators (a,b).Now given data $(x_1,f_1),(x_2,f_2),\ldots,(x_n,f_n)$, estimate (a,b) in some sense. In this case it is not possible to followe the linear estimator problem.

2. Taylor series

If $f(a)$ is differentiable w.r.t *a*, then it mya be expanded into power series as the Taylor series

$$f(a) = f(\hat{a}) + \frac{\mathrm{df}}{\mathrm{da}}|_{a=\hat{a}}(a-\hat{a}) + \frac{1}{2}\frac{d^2 f}{\mathrm{da}^2}|_{a=\hat{a}}(a-\hat{a})^2 + ,\ldots.\forall\ \hat{a}$$

In multivariable case, if $f(a,b)$ is differentiable w.r.t (a,b), then

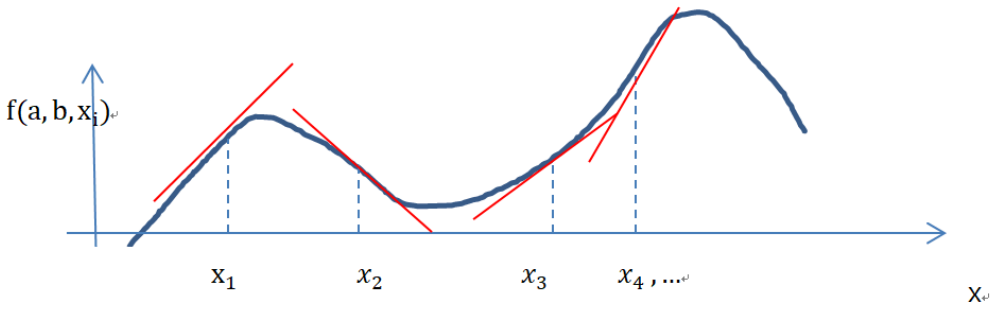
$$f(a,b,x) = f(\hat{a},\hat{b}) + \begin{bmatrix}\frac{\partial f}{\partial a} & \frac{\partial f}{\partial b}\end{bmatrix}|_{(a,b)=(\hat{a},\hat{b})}\begin{bmatrix}(a-\hat{a})\\(b-\hat{b})\end{bmatrix} + O^2(a,b),\ldots\forall\ \hat{a},\hat{b} \qquad (1)$$

3 Non-linear least square estimator

Now consider non-linear function with un-known parameters $f(a,b,x)$ with measurements

$$(x_1, f(a,b,x_1)), (x_2, f(a,b,x_2)), \ldots., f(a,b,x_n)$$

A conceptual graph is drawn as



in contrast to linear case, the slopes may be different at each sample points, which is more complicate to linear case.

However from the equation (1), if the O^2 is neglected, then

$$f(a,b,x) \sim f(\hat{a},\hat{b},x) + \begin{bmatrix}\frac{\partial f}{\partial a} & \frac{\partial f}{\partial b}\end{bmatrix}|_{(a,b)=(\hat{a},\hat{b})}\begin{bmatrix}(a-\hat{a})\\(b-\hat{b})\end{bmatrix}$$

which in turn

$$f(a,b,x) - f(\hat{a},\hat{b},x) = \begin{bmatrix}\frac{\partial f}{\partial a} & \frac{\partial f}{\partial b}\end{bmatrix}|_{(a,b)=(\hat{a},\hat{b})}\begin{bmatrix}(a-\hat{a})\\(b-\hat{b})\end{bmatrix}$$

or given the measurement $(x_1,f_1),(x_2,f_2),\ldots,(x_n,f_n)$

$$f(a,b,x_1) - f(\hat{a},\hat{b},x_1) = \left[\frac{\partial f(a,b,x_1)}{\partial a} \quad \frac{\partial f(a,b,x_1)}{\partial b}\right]|_{(a,b)=(\hat{a},\hat{b})}\begin{bmatrix}(a-\hat{a})\\(b-\hat{b})\end{bmatrix}$$

$$f(a,b,x_2) - f(\hat{a},\hat{b},x_2) = \left[\frac{\partial f(a,b,x_2)}{\partial a} \quad \frac{\partial f(a,b,x_2)}{\partial b}\right]|_{(a,b)=(\hat{a},\hat{b})}\begin{bmatrix}(a-\hat{a})\\(b-\hat{b})\end{bmatrix}$$

…

$$f(a,b,x_n) - f(\hat{a},\hat{b},x_n) = \left[\frac{\partial f(a,b,x_n)}{\partial a} \quad \frac{\partial f(a,b,x_n)}{\partial b}\right]|_{(a,b)=(\hat{a},\hat{b})}\begin{bmatrix}(a-\hat{a})\\(b-\hat{b})\end{bmatrix}$$

Let us define the error between the real and the estimator is z_i, and $a-\hat{a}=\delta a$ and so on then the above equation is

$$z_1 = \left[\frac{\partial f(a,b,x_1)}{\partial a} \quad \frac{\partial f(a,b,x_1)}{\partial b}\right]|_{(a,b)=(\hat{a},\hat{b})}\begin{bmatrix}\delta a\\ \delta b\end{bmatrix} \text{ at } x=x_1$$

$$z_2 = \left[\frac{\partial f(a,b,x_2)}{\partial a} \quad \frac{\partial f(a,b,x_2)}{\partial b}\right]|_{(a,b)=(\hat{a},\hat{b})}\begin{bmatrix}\delta a\\ \delta b\end{bmatrix} \text{ at } x=x_2$$

…

$$z_n = \left[\frac{\partial f(a,b,x_n)}{\partial a} \quad \frac{\partial f(a,b,x_n)}{\partial b}\right]|_{(a,b)=(\hat{a},\hat{b})}\begin{bmatrix}\delta a\\ \delta b\end{bmatrix} \text{at } x=x_n$$

So that

$$Z=\begin{bmatrix}z_1\\z_2\\\vdots\\z_n\end{bmatrix}=\begin{bmatrix}\frac{\partial f(a,b,x_1)}{\partial a} & \frac{\partial f(a,b,x_1)}{\partial b}\\ \frac{\partial f(a,b,x_2)}{\partial a} & \frac{\partial f(a,b,x_2)}{\partial b}\\ & \cdots \\ \frac{\partial f(a,b,x_n)}{\partial a} & \frac{\partial f(a,b,x_n)}{\partial b}\end{bmatrix}\begin{bmatrix}\delta a\\ \delta b\end{bmatrix}\equiv H\begin{bmatrix}\delta a\\ \delta b\end{bmatrix}$$

This is a standard linear least square problem, the matrix *H* is called Jacobian so that the least square estimator are

$$\begin{bmatrix}\delta a\\ \delta b\end{bmatrix}=(H^TH)^{-1}H^TZ$$

i.e. ,

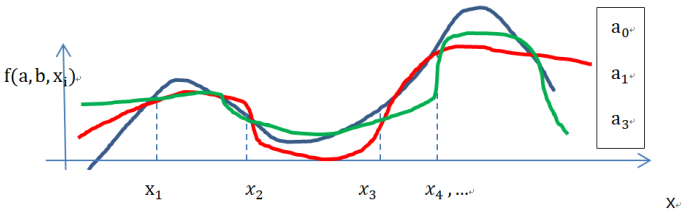
$$\begin{bmatrix}\hat{a}\\ \hat{b}\end{bmatrix}=\begin{bmatrix}a\\ b\end{bmatrix}+\begin{bmatrix}\delta a\\ \delta b\end{bmatrix}=\begin{bmatrix}a\\ b\end{bmatrix}+(H^TH)^{-1}H^TZ$$

If $f(a,b,x_i) - f(\hat{a},\hat{b},x_i)$ is small enough, then it will be finished. If not, select the next estimator as

$$\begin{bmatrix}\hat{a}\\ \hat{b}\end{bmatrix} \longleftarrow \begin{bmatrix}\hat{a}\\ \hat{b}\end{bmatrix} + \begin{bmatrix}\delta a\\ \delta b\end{bmatrix} = \begin{bmatrix}\hat{a}\\ \hat{b}\end{bmatrix} + (H^T H)^{-1} H^T Z$$

And proceed until the error is small enough.

The conceptual recursive is



4. Procedure

1) select the initial estimator as $\hat{a}_0, \hat{b}_0$. This is the first estimators

2) Construct the Jacobian *H*at each $x_1, x_2, \ldots, x_n$ with $\hat{a}_0, \hat{b}_0$

3) Define $\begin{bmatrix}\hat{a}_1\\ \hat{b}_1\end{bmatrix} = \begin{bmatrix}\hat{a}_0\\ \hat{b}_0\end{bmatrix} + (H^T H)^{-1} H^T Z$

This is the second estimators.

3) calculate the errors $z_1 = \sum_{i=1}^{n} \left(f(\hat{a}_1, \hat{b}_1, x_i) - f(\hat{a}_0, \hat{b}_0, x_i)\right)^2$,

4) if z_1 is small than terminate but it is larger than the prescribed value, the next estimator are defined as

$$\begin{bmatrix}\hat{a}_2\\ \hat{b}_2\end{bmatrix} = \begin{bmatrix}\hat{a}_1\\ \hat{b}_1\end{bmatrix} + (H^T H)^{-1} H^T Z$$

Here *H* is calcuted at $x_1, x_2, \ldots, x_n$ with $\hat{a}_1, \hat{b}_1$

5) Check the error, $z_1 = \sum_{i=1}^{n} \left(f(\hat{a}_1, \hat{b}_1, x_i) - f(\hat{a}_0, \hat{b}_0, x_i)\right)^2$, it is not satisfied,

6) go to 4) ...

1.2 Generate Data

For study Newton- gauss non-linear LME, we need measurements in real situation, but here we generate a data

```
clear all;clc
syms a b S
f= a+a*b + b*S;
Data =[ ];
```

```
for i = 1:10
    Data =[Data ; [i, subs(f, [a,b,S], [1,2,i])]];
end
Data
```

```
Data =
```

$$\begin{pmatrix} 1 & 5 \\ 2 & 7 \\ 3 & 9 \\ 4 & 11 \\ 5 & 13 \\ 6 & 15 \\ 7 & 17 \\ 8 & 19 \\ 9 & 21 \\ 10 & 23 \end{pmatrix}$$

Here the real values to be estimate are

$$(a, b) = (1, 2)$$

1.3 Generation the pseudo Inverse of the Jacobian

```
H = [diff(f,a)  diff(f,b)];  % gradient w.r.t the estimators
estX =[4;2];                 % initial guess of estimators
Z =[ ];
JA = [ ];
for k = 1:5
    for i = 1:10
        Z=[Z; Data(i,2) - subs(f,[a,b,S],[estX(1), estX(2), Data(i,1)])];
        JA =[JA;subs(H,[a,b,S],[estX(1), estX(2),Data(i,1)])];
    end

  PseJA = inv(JA'*JA)*JA';   % pseudo Inverse
  estX =estX +PseJA*Z;       % next estimator
  k
  estX = double(estX)      % conver to double
  Z=[ ];
  JA =[ ];
 end
```

```
k = 1
estX = 2×1
     1
     2
k = 2
estX = 2×1
     1
     2
k = 3
estX = 2×1
     1
     2
```

```
k = 4
estX = 2×1
     1
     2
k = 5
estX = 2×1
     1
     2
```

```
estX
```

```
estX = 2×1
     1
     2
```

here we get the real value at the frist step. If you change the initial guess

1.1 Problem

$$f = a/(b+t)$$

This is a linear w.r.t the variable *t*, but non-linear w.r.t the estimators

1.2 Generate Data

For study Newton- gauss non-linear LME, we need measurements in real situation, but here we generate a data

```
clear all;clc
syms a b t
f= a/(b+t);
Data =[ ];
for i = 1:10
    Data =[Data ; [i, subs(f, [a,b,t], [1,2,i])] ];
end
Data
```

```
Data =
```

$$\begin{pmatrix} 1 & \frac{1}{3} \\ 2 & \frac{1}{4} \\ 3 & \frac{1}{5} \\ 4 & \frac{1}{6} \\ 5 & \frac{1}{7} \\ 6 & \frac{1}{8} \\ 7 & \frac{1}{9} \\ 8 & \frac{1}{10} \\ 9 & \frac{1}{11} \\ 10 & \frac{1}{12} \end{pmatrix}$$

Here the real values to be estimate are

$$(a, b) = (1, 2)$$

1.3 Generation the pseudo Inverse of the Jacobian

```
H = [diff(f,a)  diff(f,b)];  % gradient w.r.t the estimators
estX =[1;1];                 % initial guess of estimators
Z =[ ];
JA = [ ];
for k = 1:10
    for i = 1:10
        Z=[Z;Data(i,2) - subs(f,[a,b,t],[estX(1), estX(2), Data(i,1)])];
        JA =[JA;subs(H,[a,b,t],[estX(1), estX(2),Data(i,1)])];
    end

  PseJA = inv(JA'*JA)*JA';   % pseudo Inverse
  estX =estX +PseJA*Z;        % next estimator
  estX = double(estX)       % conver to double
  Z=[ ];
  JA =[ ];
 end
```

```
estX = 2×1
    0.9506
    1.5740
estX = 2×1
    0.9916
    1.9376
estX = 2×1
    0.9998
```

```
    1.9988
estX = 2×1
    1.0000
    2.0000
estX = 2×1
    1.0000
    2.0000
estX = 2×1
     1
     2
estX = 2×1
     1
     2
estX = 2×1
     1
     2
estX = 2×1
     1
     2
estX = 2×1
     1
     2
```

```
 estX
```

```
estX = 2×1
     1
     2
```

here we get the real value at the frist step. If you change the initial guess

2. Example

https://en.wikipedia.org/wiki/Gauss%E2%80%93Newton_algorithm

1. problem

In biology an example, the measurement is the rate *R*, function of the concentration, *S*

$$R = \frac{aS}{b+S},$$

to estiamte $a,\ b$ with the measurement as

i	1	2	3	4	5	6	7
$[S]$	0.038	0.194	0.425	0.626	1.253	2.500	3.740
Rate	0.050	0.127	0.094	0.2122	0.2729	0.2665	0.3317

Here is the non-linear w.r.t the variable *S* and the estimators

2. Procesure

2.1 Data generation

```
clear all; clc
% Data
DataS =[0.038;0.194;0.425;0.626;1.253;2.500;...
        3.740];
```

```
DataR = [0.050;0.127;0.094;0.2122;0.2729;0.2665;...
        0.3317];
Data=[DataS DataR]
```

```
Data = 7×2
    0.0380    0.0500
    0.1940    0.1270
    0.4250    0.0940
    0.6260    0.2122
    1.2530    0.2729
    2.5000    0.2665
    3.7400    0.3317
```

```
plot(Data(:,1), Data(:,2),'ro');grid on;hold on

syms a b S
R = a*S/(b + S);
% Gradient
H = [diff(R,a) diff(R,b)];
% calculate the error with the first estimator
estX =[0.9;0.2];
Z =[ ];
JA = [ ];
for k = 1:10
    for i = 1:7
        Z=[Z; Data(i,2) - subs(R,[a,b,S],[estX(1), estX(2), Data(i,1)])];
        JA =[JA;subs(H,[a,b,S],[estX(1), estX(2),Data(i,1)])];
    end
  PseJA = inv(JA'*JA)*JA';
  estX =estX +PseJA*Z;
  estX = double(estX)
  Z = [ ];
  JA =[ ];
 end
```

```
estX = 2×1
    0.3327
    0.2602
estX = 2×1
    0.3428
    0.4261
estX = 2×1
    0.3578
    0.5295
estX = 2×1
    0.3614
    0.5537
estX = 2×1
    0.3618
    0.5561
estX = 2×1
    0.3618
    0.5563
estX = 2×1
    0.3618
    0.5563
estX = 2×1
    0.3618
    0.5563
estX = 2×1
```

```
    0.3618
    0.5563
estX = 2×1
    0.3618
    0.5563
```

```
for i =1:7
    temp(i) =estX(1)*DataS(i)/(estX(2)+DataS(i))
end
```

```
temp = 0.0231
temp = 1×2
    0.0231    0.0936
temp = 1×3
    0.0231    0.0936    0.1567
temp = 1×4
    0.0231    0.0936    0.1567    0.1916
temp = 1×5
    0.0231    0.0936    0.1567    0.1916    0.2506
temp = 1×6
    0.0231    0.0936    0.1567    0.1916    0.2506    0.2960
temp = 1×7
    0.0231    0.0936    0.1567    0.1916    0.2506    0.2960    0.3150
```

```
plot(DataS,temp,'linewidth',2)
```