##### WEEK 1

###### Algorithm:

1.Supervised: Give a set of data and right answer

1).Regression : continuous valued output

2).Classfication: predict a discrete value output

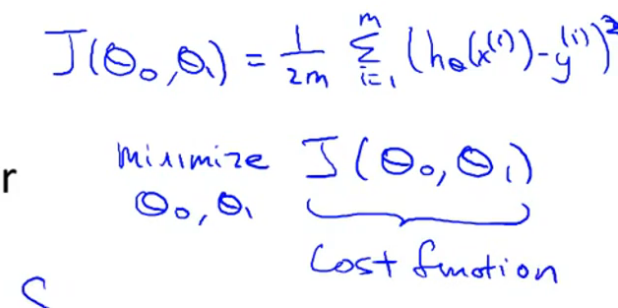
2.UnSupervised:

approach problems with little or no idea what our results should look like.

We can derive structure from data where we don't necessarily know the effect of the variables

###### Cost Function:

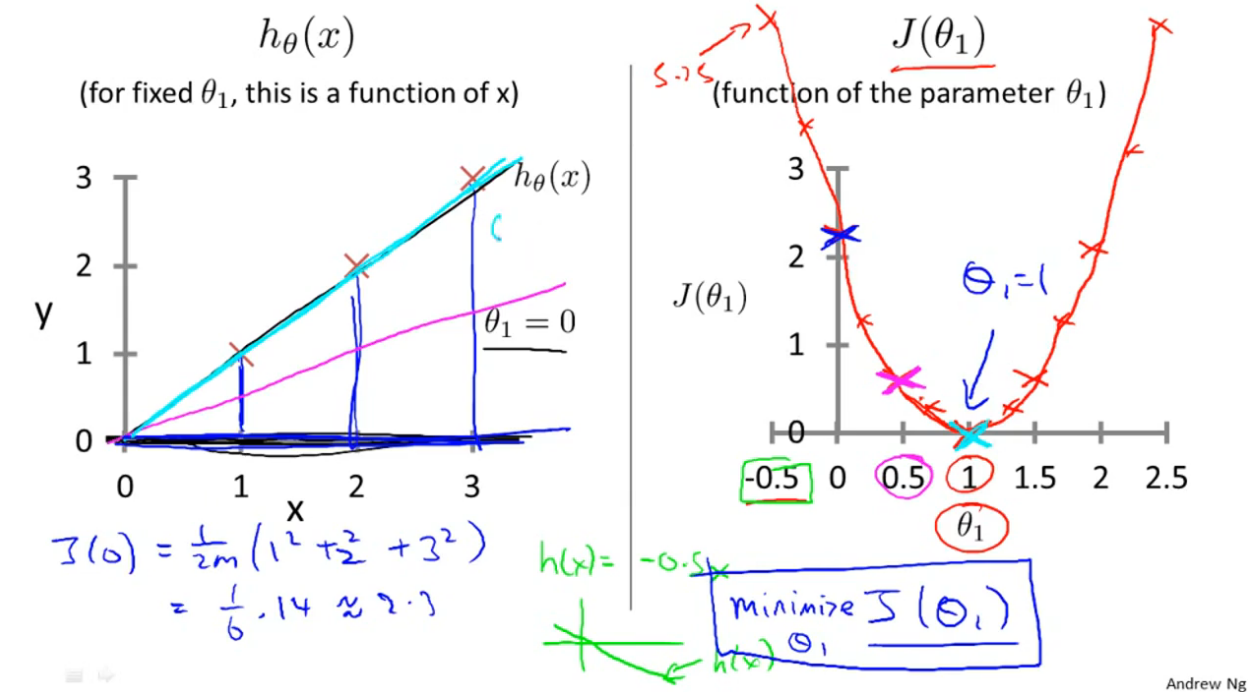
1.Mean Squired error cost function: 最常见的在linear regression问题中：



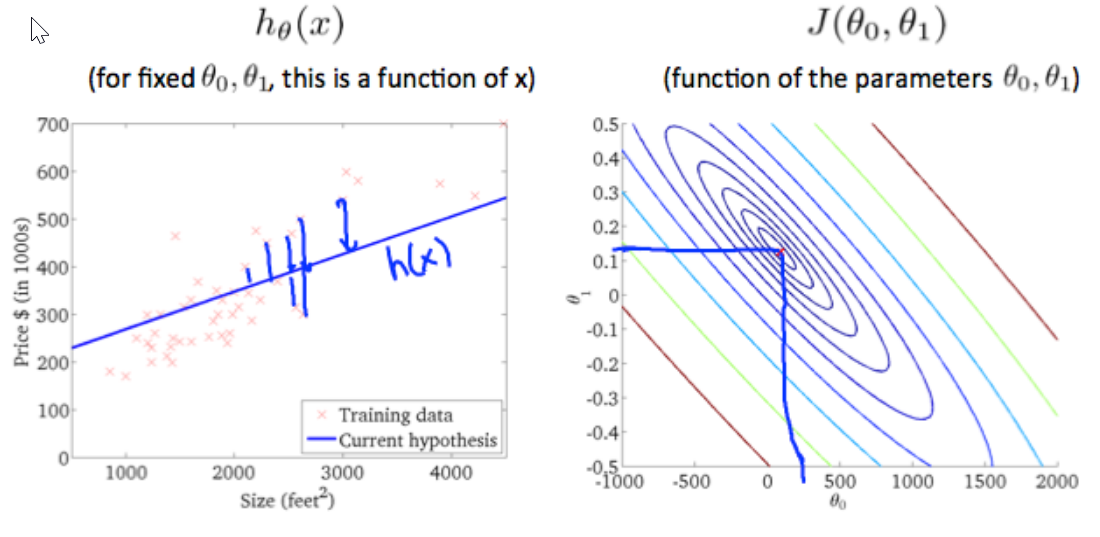
M : # of training set

相当于是差的平方和 平均数/2

除以二是因为有 gradient descent 问题， 有的地方不需要除以二

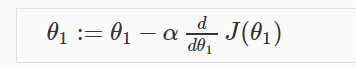


J() 就是cost function 可以看见在参数 = 1 的时候 J()最小。 所以这个参数就是1



Y = a + bx 这样的cost function就是一个碗状图

###### Gradient Descent algorithm



相当于每次参数减去alpha\*斜率 所以参数会越来越接近最低点

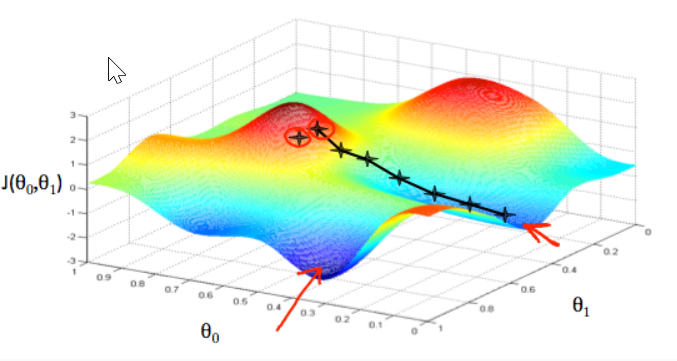
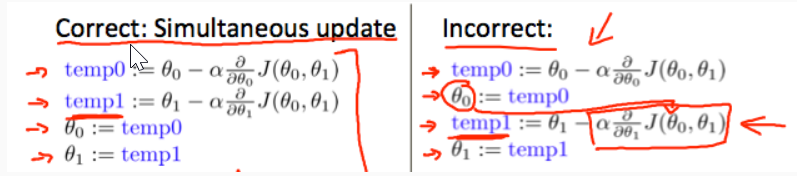
但是前提是alpha不能 too large！！！！！

a := b 赋值 a = b （判断a==b）这里和java不一样

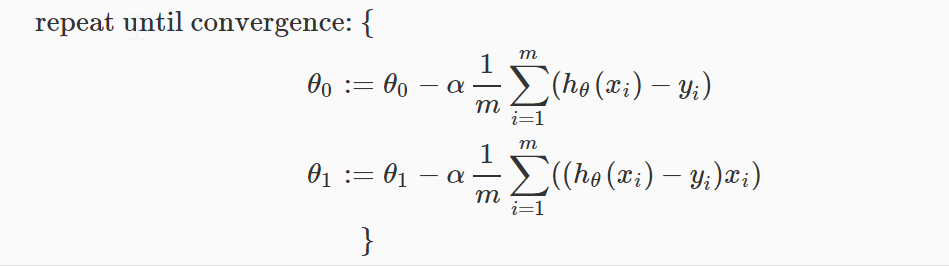
梯度递减就是一步一步往下走，直到j(a,b)的值最小

It is Simultaneous update

同步更新参数

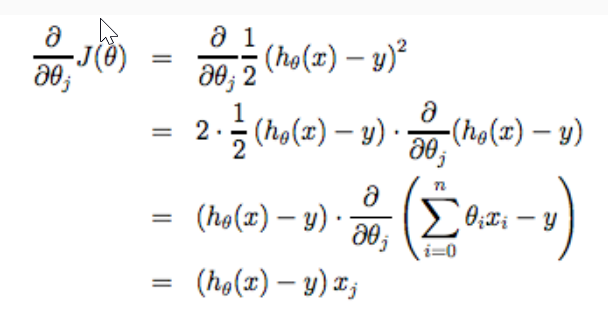


###### Gradient Descent for Linear Regression



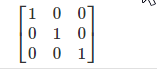
通过分别求导得出来的结果。（牛逼啊！！）

Derivative process：



###### Matrix Matrices are not commutative: A∗B≠B∗A

If B is Identity matrix , AB = BA

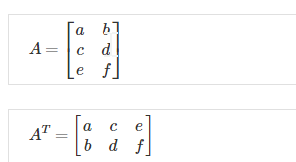


Matrix Inverse:

A \* A-1  = identity matrix

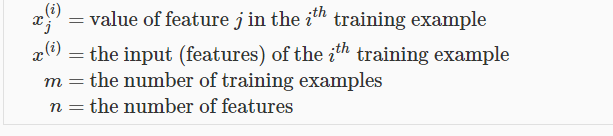
*pinv*(*A*) inv(A)都可以，有细微差别

Matrix transposition

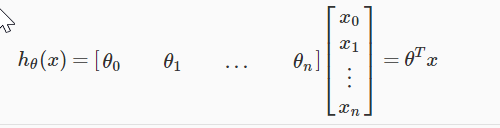


##### WEEK2

###### Multiple Features



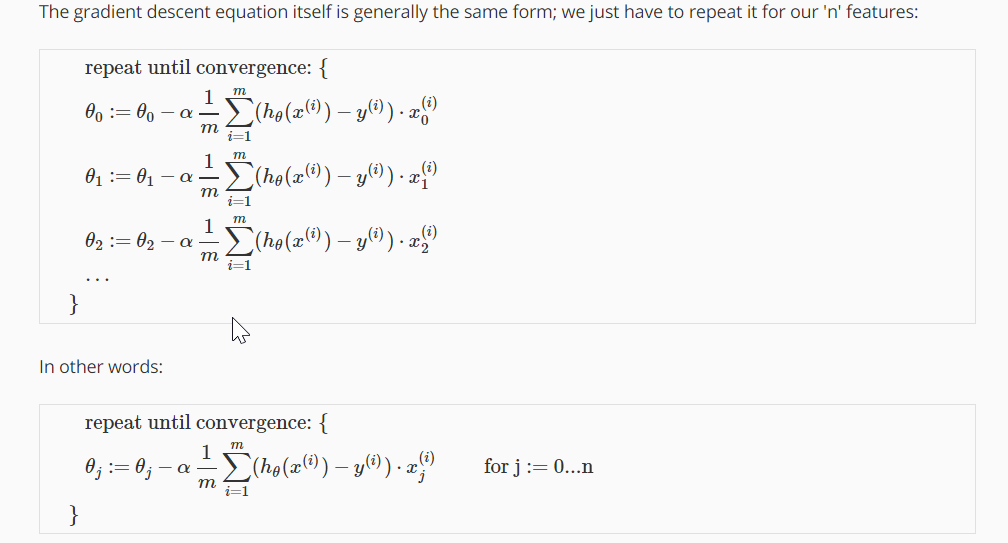
所以



变为h(x) = θ0 \* X0 + θ1 \* X1 + …..

让X0 = 1 即可

[θ0，θ1，θ2] [1, X1, X2]



###### Feature Scaling

Approximately -1<= x <= 1 range

X1 = X1 / s

###### Mean Normalization

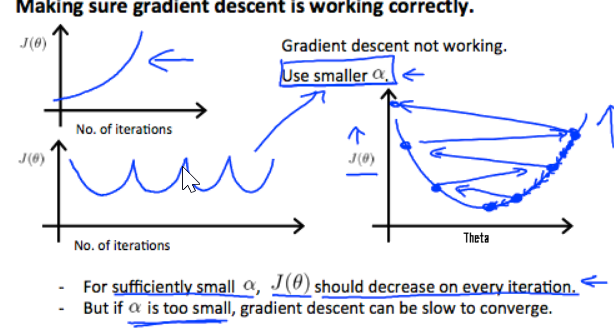
x1 = (x1 – ave) /s

S = max – min (The range)

###### Learning Rate

If *α* is too small: slow convergence.

If *α* is too large: ￼may not decrease on every iteration and thus may not converge.



Alpha 太大，cost Function就会出现左边的情况，可能不是每次都减小

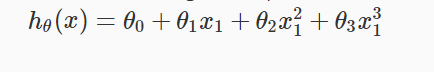
所以我要找到最大的可能值。 可以这样找： 1， 0.3 ，0.1， 0.03， 0.01， 0.003， 0.001这样。

###### Polynomial Regression

We can **change the behavior or curve** of our hypothesis function by making it a quadratic, cubic or square root function (or any other form).

多项式回归

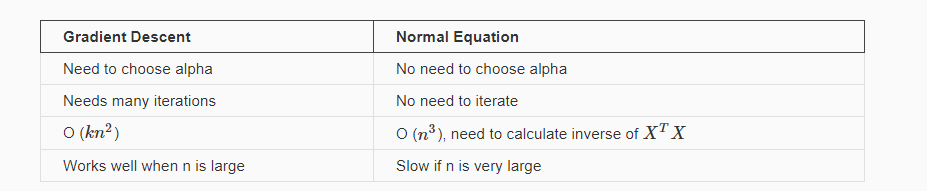
自己创建新的x 比如用X2也当做一个X



**但是这样做需要注意，需要做feature scaling. 因为数值差距太大**

###### **Two way to decreate J(**θ**)**

1. gradient descent
2. Normal Equation (Only can used in linear regression)

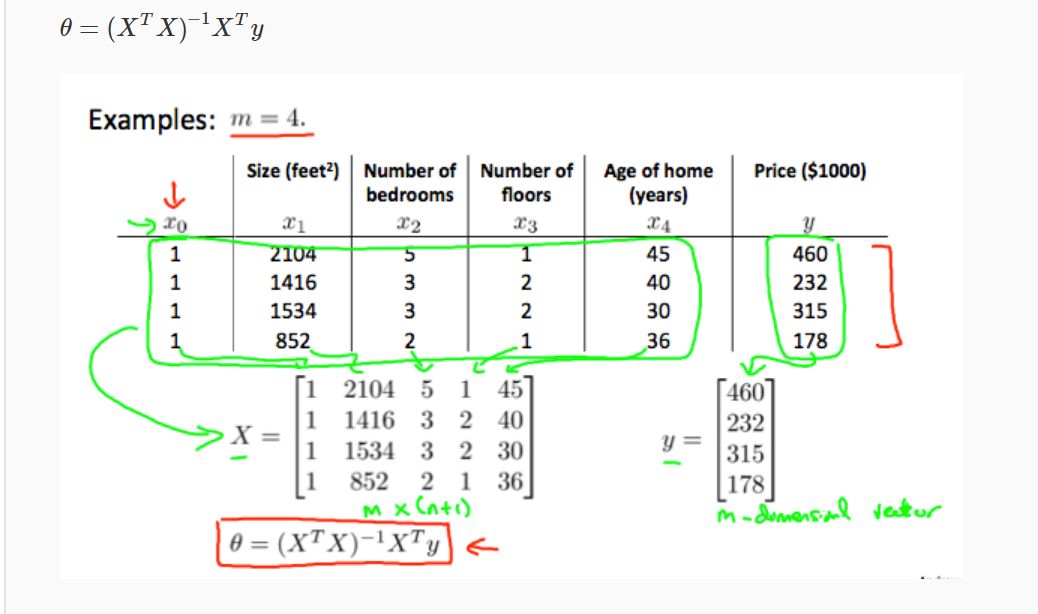


So, if n < 10000, use Normal Equation

Else use Gradient descent

Plus：Use Normal Equation , feature scaling is not necessary!!!

For Normal Equation: This table n = 5



这样计算就得出了所有参数的值。暂时我并不知道公式的来源，先这么记着

###### Normal Equation Noninvertibility

1. Redundant features, where two features are very closely related (i.e. they are linearly dependent)

比如两个属性 x1(m2) x2(feet2) 都是面积，实际上是一样的意思，他们是linearly dependent

X1 = 3.28\*3.28\*X2

所以删去一个属性就好

2.Too many features (e.g. m ≤ n). In this case, delete some features or use "regularization"

###### Octave

结尾加；就不会打印

1 == 2

1 ~= 2 不等于

a = 3.1416

disp(sprint(‘2 decimals : %0.2f’,a)) 小数点后面两位

format long

format short

v = 1:0.1:2 [1,1.1,1.2…2]

ones(2,3) 就是一个全是1的2\*3数组

rand(1,3) 随机数组

randn(1,3)有负数

hist（w,30） 画图统计个数

eye(4) identity matrix

help eye

A=[1,2;3,4;5,6]

size(A) = 3 , 2

length(A) = 3 因为最大的维度是3

load filename.dat

用who查看当前所有variable

Whos 查勘details

clear A 删除

v=princY(1:10) 取前十个

save hello.mat v; 保存文件了且在当前路径

save hello.txt v –ascii (用ascii的方式保存)

A(3,2) 下标

A = [A,[100;200;300]]; 给A加一列

A(:) put all into single vector

C = [A B] A is in left B is in right

C= [A ; B] A is in top B is in bottom

Matrix multiple

A \* B =

A .\* B = 这种是每一项相乘，和加法类似

A .^ 2 = 每一项取平方

1. ./ A 变为分母

-A = -1 \* A

A’ is transpose of A

[val,ind] = max(A)

[r,c] = find(A >= 7) row column 对应的

find(A > 2)

sum(A) 综和

prod（A） 总乘

floor(A) 向下取整

ceil(A)向上取整

max(A,[],1) 每一列取最大

max(A,[],2)每一行取最大

max(A) 默认相当于每一列最大

###### plot:

flipud(eye(3))

pinv(A) 取 A-1

plot(x,y)

hold on; 保持住当前

plot(x,z,’r’) 颜色red

xlabel(‘time’)

ylabel(‘value’) 取名

legend(‘sin’,’cos’) 标每条线名字

title(‘my plot’)

cd ‘D:/as/’; print –dpng ‘my plot.png’

figure(1);plot()

figure(2);plot() 两个不同的窗口

subplot(1,2,1) 把画面变两个小块，画在第一个小块里

axis([0.5 1 -1 1]) x变为0.5 ~ 1 y 变为 -1 ~ 1

Control flow

for I = 1:10,

V(i) = 2^I;

end;

while I <= 5,

V(i) = 100;

end;

function: 在一个文件里面 aa.m

1.cd 到function目录

2.或者把function的目录加载进来

addpath(‘d:/asd/’)

这样就可以用了

aa（32）

function可以返回多个值 aa.m

function[y1,y1] = aa(x)

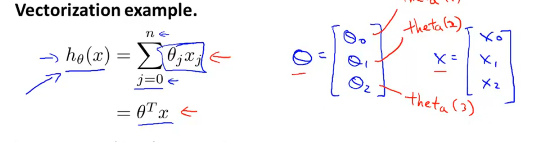
y1 = x^2

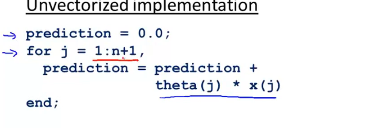
y2 = x^3

Return 2 values

Then we can use [a,b] = aa(3)

Vectorization:





用非向量化方法写的话很麻烦，需要自己写function：

然而我们可以吧问题变为变为θT \* x.