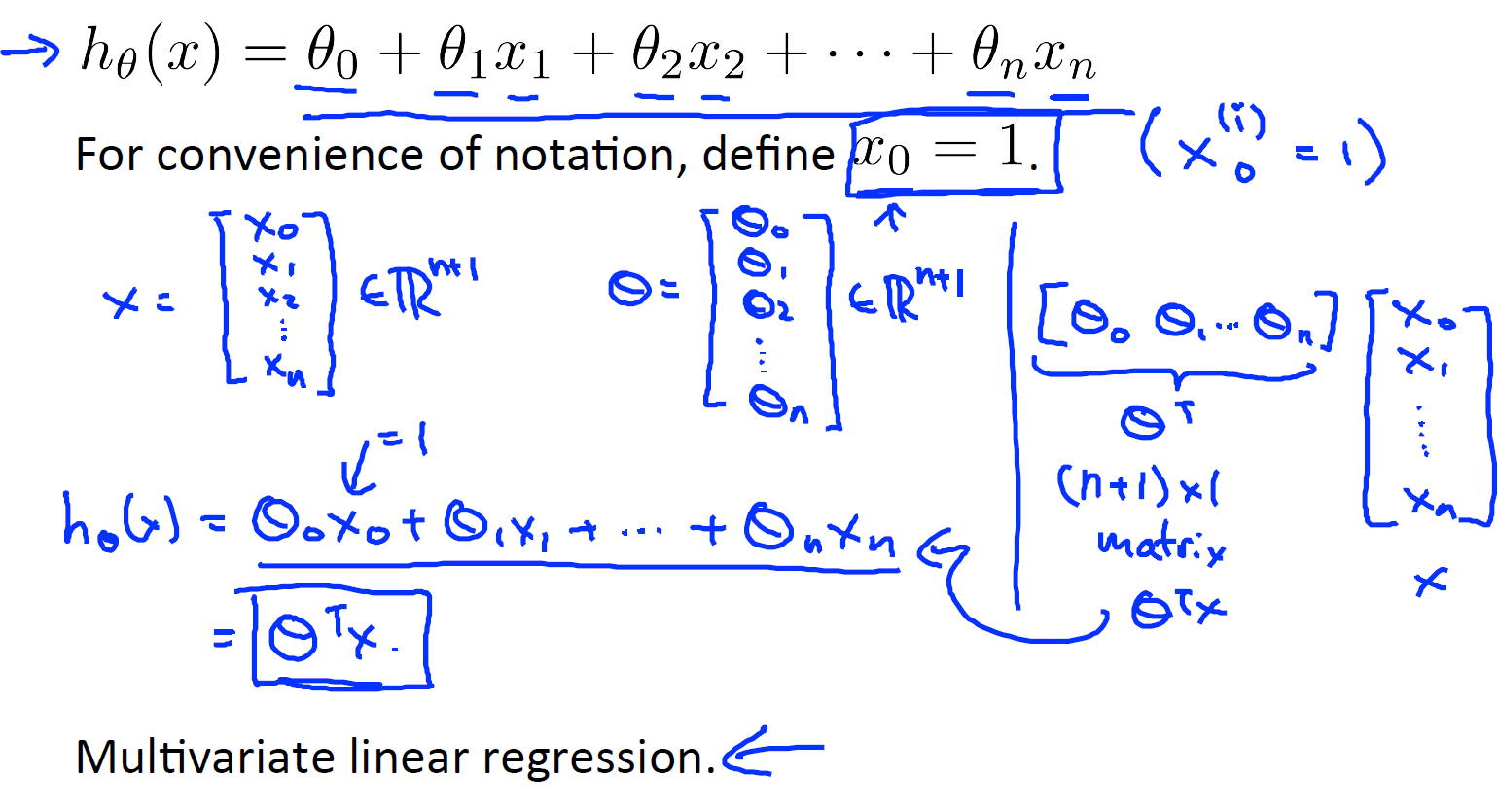
* I should use polynomial regression to express multiple features
* when we express the hypothesis with one variable (it will be a straight light) so its not fit the data, we should create multi variables from one variable.
* When we create multi variables from one variable, it is very important to use feature scaling
* Learning rate should not be so small because the cost function will converge very slow and it also shouldn't be big.
* if we use multi features with different range, we should do feature scaling, so that we can find the converge faster.
* Thay vì dung for loop, thì dung vectorization sẽ tiết kiệm sự tính toán cho máy tính rất nhiều
* 

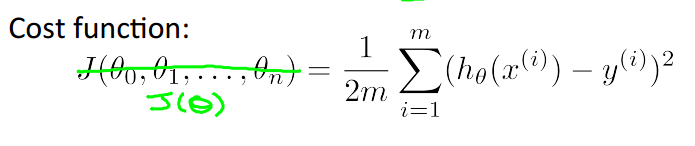
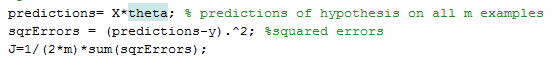
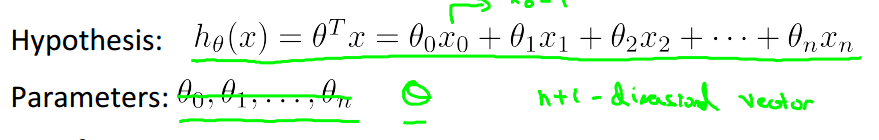
# Giải thích bài về nhà 1

N: số lượng features . ở đây là 2, 1 feature về dân số. và 1 cái là them số 1

M: số lượng samples, nhiều

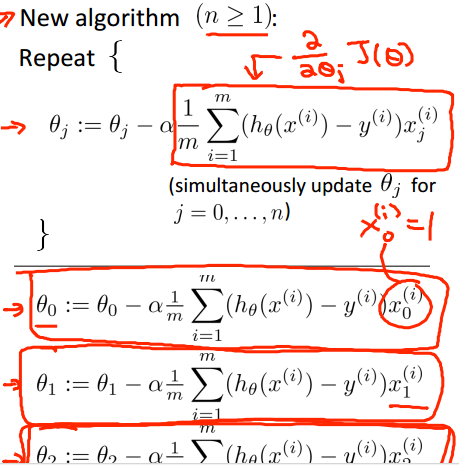
I: thứ tự samples

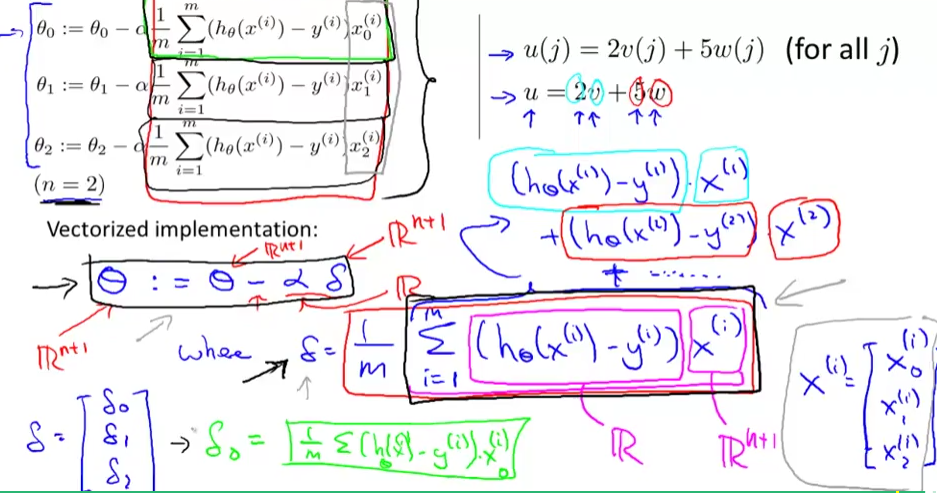
X: mxn

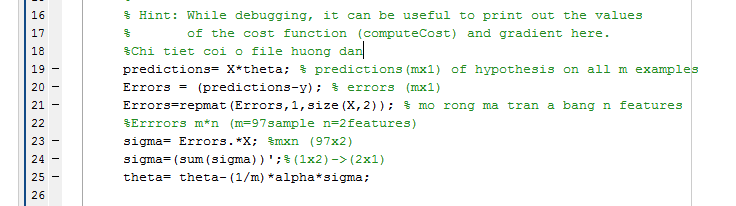
* Hàm compute cost:  
   
  + Hàm predictions
* 
  + - 

Theta ở đây là có theta0 va theta 1, X la vector gom x0 và x1 (n=2).  
Nhưng mà ta có m sample => predictions sẽ có m hang, m\*2. Từ đó tính J

## Gradient descent







Sum: sẽ là một chiều của ma trận (de dien tả cho sigma). Còn chiều con lại của ma trận là diễn tả cho việc update 1 lúc nhiều theta.

Ở đây tổng sigma của m, nghĩ là m sample mình sẽ xếp theo 1 chiều (mình dung cột của ma trân)=> X sẽ là mxsố hang (số hang này chính là bằng chiều của vector theta)

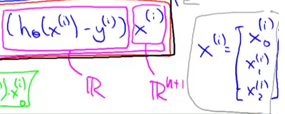
Dòng 19: tính X=(m\*n) với m= 97, n=2, theta = 2x1



Dòng 20



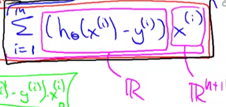
Dòng 21+23:



Vì R là một hằng số nhưng có m\*1 cột , ứng với m sample. VỚi mỗi sample thì có n=2 features

* + Để nhân từng phân tử X (m\*n) thì ma trân R cũng phải (m\*n). Đơn giản ta chỉ lập lại nó n lần nó theo cột

DÒng 24:



Dòng 25:

