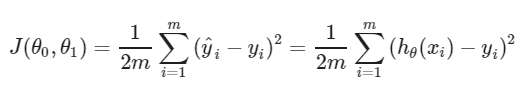
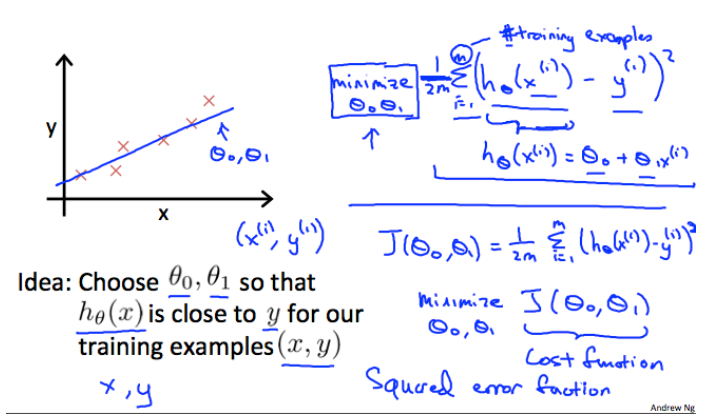
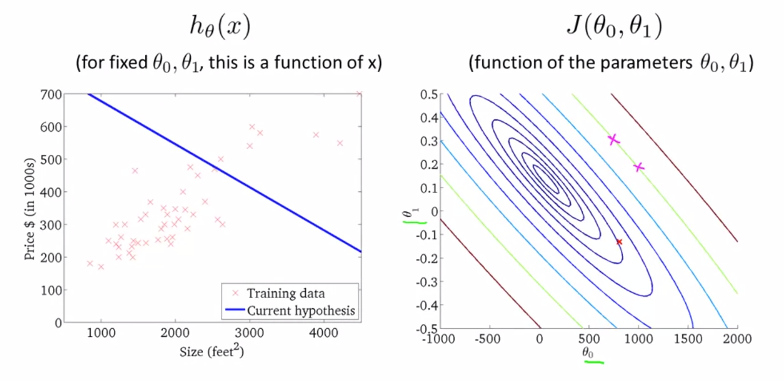
**Week 1:**

1. Algo and math not just enough, application is imp
2. ML grew out of AI
3. Examples:
   1. Click stream data used know more about users
   2. Medical record to medical knowledge
   3. Genomics
   4. Autonomous helicopter (can’t be programmed)
   5. NLP
   6. Handwriting recognition
   7. CV
   8. Recommendation systems
4. ML – ability to learn without being explicitly programmed (Checkers example)
5. Supervised Learning – labelled dataset (for every data point we had “right” answer)
   1. Classification / Regression
6. Unsupervised Learning – don’t give “right” answers
   1. Clustering
      1. Google news (cohesive news clustering)
      2. Clustering people into group from gene data
      3. Social network analysis
      4. Market segmentation
   2. Cocktail Party problem – hard to hear people in noisy party
      1. Multiple speaker recording multiple people’s audio
      2. Pass them to algo, then algo will split the audios
7. Model Representation – Regression Supervised problem (hypothesis)
8. Cost Function – Help fit best line to data (choosing parameters of model changes hypothesis function)
   1. Linear regression – we need to determine 2 parameters θ0 and θ1 so as to minimize squared error of all the training examples

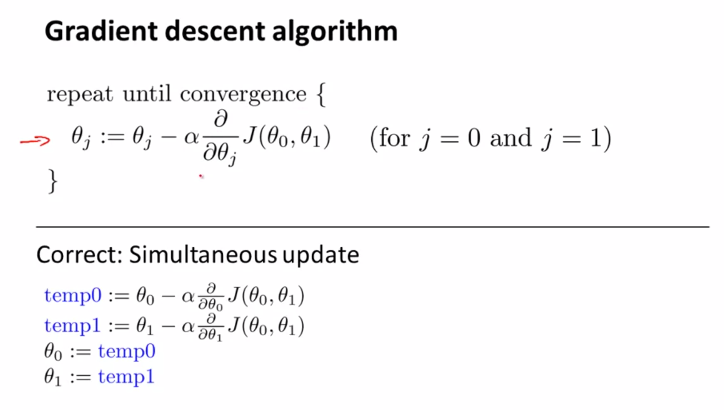




* 1. Contour plots used for better visualizing cost function minima point

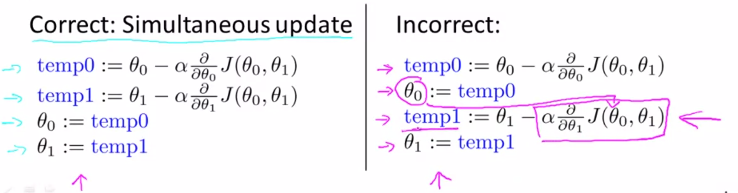


1. Gradient Descent – used to minimize cost function (in linear regression). It applied to any general cost function
   1. Vary θ0 and θ1 and go towards local minima



Alpha – learning rate, derivative of cost function (slope of tangent)

* 1. Correct way to implement gradient descent (simultaneous update)

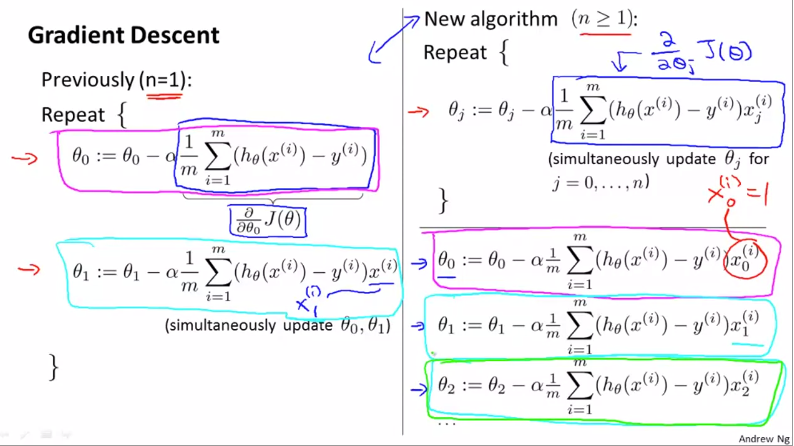


* 1. For linear regression, we will always have a convex function for cost

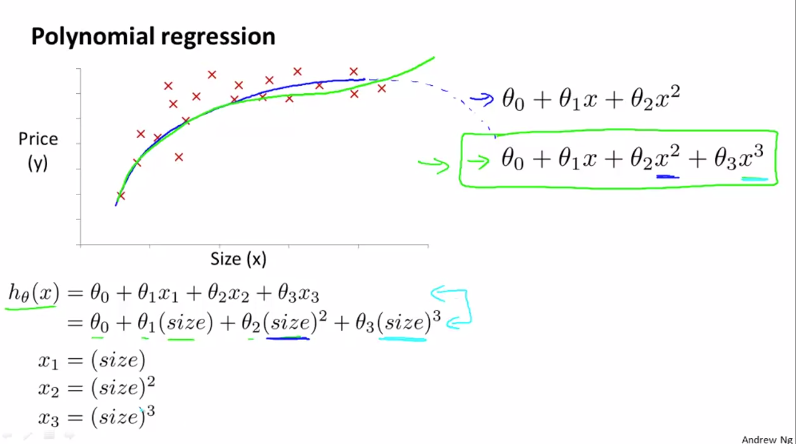
1. Linear Algebra:
   1. Matrix: dimensions
   2. Vector: n x 1 matrix, 1 indexed (1,2, 3….), 0 indexed (0,1, 2…)
   3. Matrix Algebra: addition, scalar multiplication, vector multiplication
      1. Express linear regression hypothesis function as vector multiplication
      2. Prediction = data matrix \* parameters
      3. Method to apply multiple hypothesis on dataset (linear regression)
   4. Matrix properties:
      1. A X B not equal to B X A
      2. A X (B X C) = (A X B) X C
      3. A X I = I X A = A
      4. Matrix which don’t have inverse are singular/ degenerate

**Week 2:**

1. Multiple Features (Multivariate linear regression):
   1. n = number of features, h depends on θ0 θ1 θ2 θ3
   2. To accommodate the constant term in the equation of hypothesis, theta matrix is represented in 0 indexed notation consisting of n+1 term
2. Gradient Descent for multiple variables:

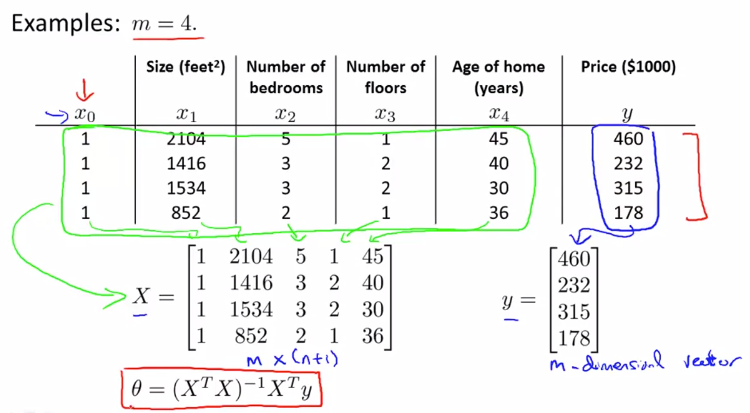


1. Feature Scaling:
   1. Convergence is fast if all features are in similar scale otherwise, we might get skewed contours
   2. It’s better to scale them in 0 to 1 (or -1 to +1) which will help in reaching global minimum faster
   3. Mean normalization: subtract all values with mean (and divide by range)
2. Learning Rate (alpha) (Practical aspect):
   1. Plot minimum cost function over iterations
   2. Ideally, the plot should decrease with each iteration
   3. When the graphs flatten, it means gradient descent has converged
   4. If the graphs do not show the usual trajectory of decrement and instead increases over iterations (or looks periodic), the gradient descent is not working properly and the alpha needs to be reduced.
   5. If alpha too small, slow convergence. If alpha is large, cost function may not converge over iterations
3. Features and Polynomial Regression:
   1. We can create new features (area), instead of using the features we already have (length, width)
   2. This can help in making better model

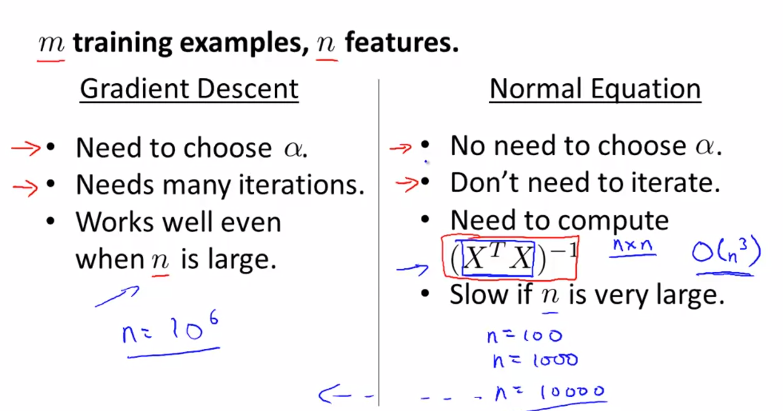


* 1. For polynomial regression, the higher order terms (square, cube) are treated as features
  2. The range will drastically change in polynomial regression, therefore scaling is important

1. Normal Equation:
   1. It is an alternative to Gradient Descent without any need of derivation
   2. Ideal way to minimize cost function is to take derivative over all thetas, equating them to zero and thus finding all thetas (features).
   3. The way to find theta that minimizes the cost function is gives as shown below



* 1. For normal equation method, it’s okay to not do feature scaling
  2. Gradient Descent v/s Normal Equation



1. Normal Equation Non-invertibility (Singular): How normal equation can be implemented for singular X transpose. X
   1. It can happen if we have redundant features (columns are linearly row)
   2. Too many features (solution – delete features, use regularization)
2. MATLAB/ Octave:
   1. Octave is used extensively for prototyping
   2. who/whos = show current variables
   3. load(‘filename.dat’) = load data from files
   4. save filename.mat variable\_name = to save variables in file (compress in binary format)
   5. save filename.txt variable\_name -ascii = human readable format
   6. A\*B = cross product
   7. A.\*B = element wise multiplication
   8. log(A), abs(A), exp(A)
   9. A’ = A transpose
   10. A < 3 = element wise comparison (generate 1,0 pattern)
   11. find (A < 3) = return row and column vectors satisfying this condition