Mathematics of Big Data, I

Lecture 2: Effective Optimization and Computation, Logistic Regression, and Generalized Linear Models

Weiqing Gu
Professor of Mathematics
Director of the Mathematics Clinic

Harvey Mudd College
Summer 2017

Recall last time we covered following

- **Frist: Big data introduction (answer first two questions)**

  Big Data Introduction
  - ***Where does big data come from?***
  - ***Different ways to describe big data***

- **Second: Use linear regression as an example to give an overview of big data analytics**

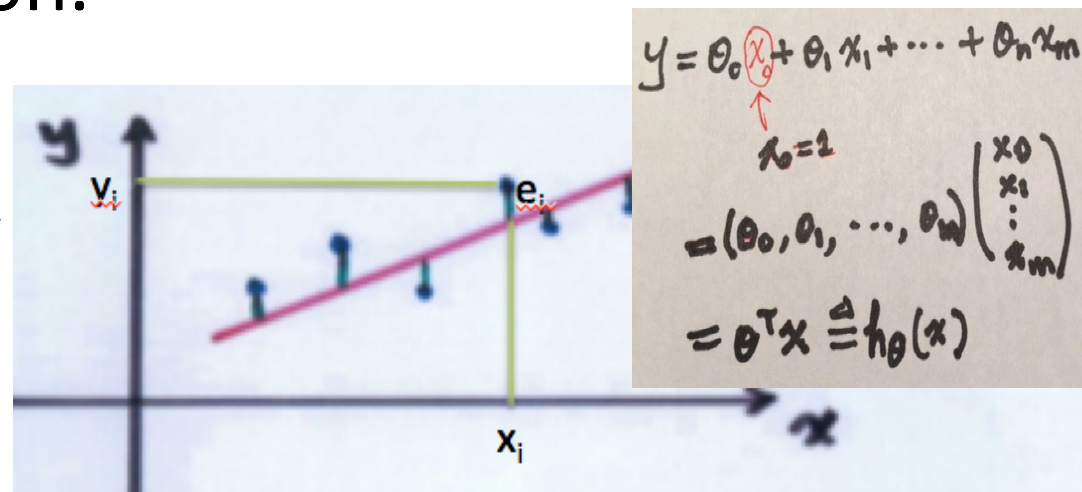
  Modeling Approaches:
  - *Statistical calculus*
  - *Geometric analytic*
  - *Probabilistic*

  Each has its own merit

Let's Recap

We had shown the following all three approaches give the same solution.

- *Statistical calculus*
- *Geometric analytic*
- *Probabilistic*



$$\boxed{\theta = (X^T X)^{-1} X^T \vec{y}}.$$

Let A = $X^T X$

Q1: What if A is not invertible? Want to achieve some perturbing of A. This is equivalent (your hw) to minimize: $\|A\mathbf{x} - \mathbf{b}\|_2^2 + \|\Gamma \mathbf{x}\|_2^2.$ (**This is Called ridge regression.**)

Q2: For big data, is it really effective to compute $(X^T X)^{-1}$? NO!

When we deal with big data, we must study

Effective optimization Techniques and Fast Computation

- For this course, we will focus on
  - Gradient Descent
    - **Batch gradient descent**
    - **Stochastic gradient descent**
  - Newton's method
  - Various matrix decompositions, for examples
    - LU decomposition
    - Cholesky decomposition

For example: We can use LU or Cholesky decomposition to solve the normal eqn:

$$X^T X\theta = X^T\vec{y}$$

Recall: For linear regression, we want want to choose θ to minimize J(θ).

$$J(\theta) = \frac{1}{2}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 = \frac{1}{2}(X\theta - \vec{y})^T(X\theta - \vec{y})$$

Key: J is quadratic on θ ; Exists Unique Minimum!

$$h_\theta(x^{(i)}) = (x^{(i)})^T\theta$$

Note: h is linear on θ !

$$\vec{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{bmatrix}$$

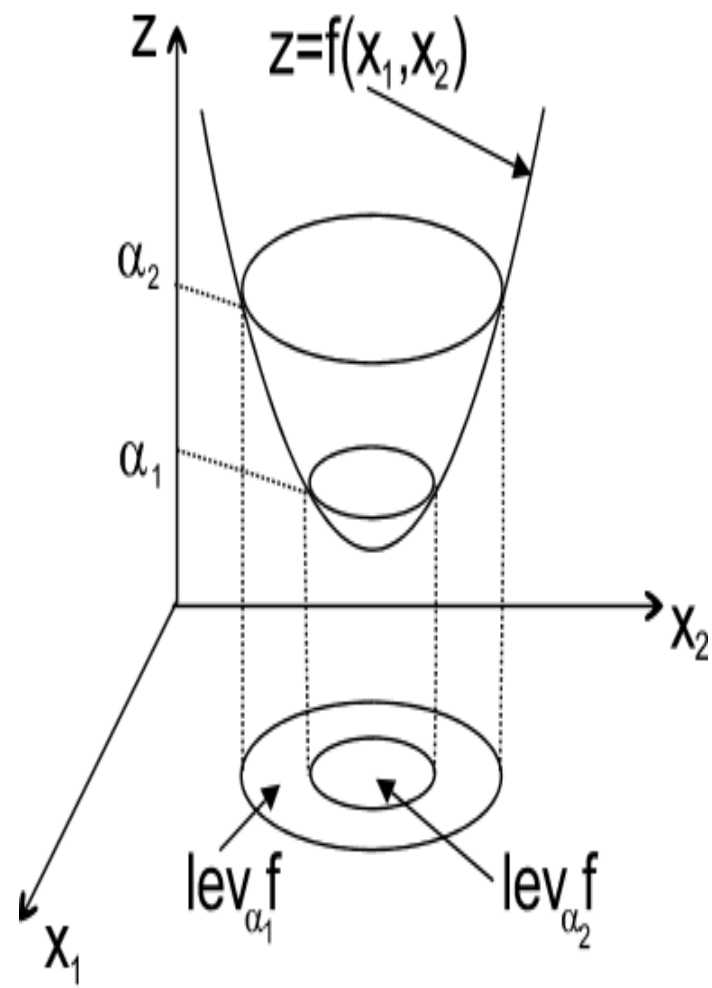
$$X\theta - \vec{y} = \begin{bmatrix} (x^{(1)})^T\theta \\ \vdots \\ (x^{(m)})^T\theta \end{bmatrix} - \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(m)} \end{bmatrix} = \begin{bmatrix} h_\theta(x^{(1)}) - y^{(1)} \\ \vdots \\ h_\theta(x^{(m)}) - y^{(m)} \end{bmatrix}.$$

$$X = \begin{bmatrix} — & (x^{(1)})^T & — \\ — & (x^{(2)})^T & — \\ & \vdots & \\ — & (x^{(m)})^T & — \end{bmatrix}$$

Why does $J(\theta)$ have a unique minimum?

(Exercise: Use two different ways to prove it—Hints below.)

- Since X^TX is positive definite when X^TX is invertible.
- (Hint for proof: $v^T(X^TX)v = (Xv)^T(Xv) = ||Xv||^2 \geq 0$ and the equality hold, figure out why v has to be 0 using the rank of X.)
- In one variable, $f(x) = ax^2 + bx + c$, if a >0, how does the graph of f look like?
- Another way: use geometric approach to get the normal equation and write down the unique solution.



(Least Mean Square) **LMS Algorithm**

Q: Given a training set, how do we pick/learn, the parameters θ?
A: Find the gradient of J(θ).

$$J(\theta) = \frac{1}{2}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2.$$

$$\theta_j := \theta_j - \alpha\frac{\partial}{\partial\theta_j}J(\theta).$$

Note: Here it real should be the transpose of it times itself. But when you take derivative, you think it is a square.

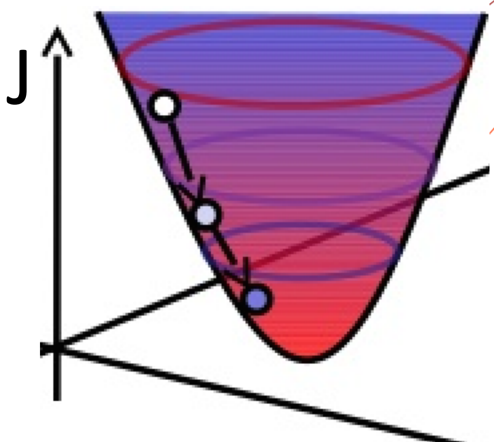
$$\begin{aligned}
\frac{\partial}{\partial\theta_j}J(\theta) &= \frac{\partial}{\partial\theta_j}\frac{1}{2}\left(h_\theta(x) - y\right)^2 \\
&= 2\cdot\frac{1}{2}\left(h_\theta(x) - y\right)\cdot\frac{\partial}{\partial\theta_j}\left(h_\theta(x) - y\right) \\
&= \left(h_\theta(x) - y\right)\cdot\frac{\partial}{\partial\theta_j}\left(\sum_{i=0}^{n}\theta_i x_i - y\right) \\
&= \left(h_\theta(x) - y\right)x_j
\end{aligned}$$

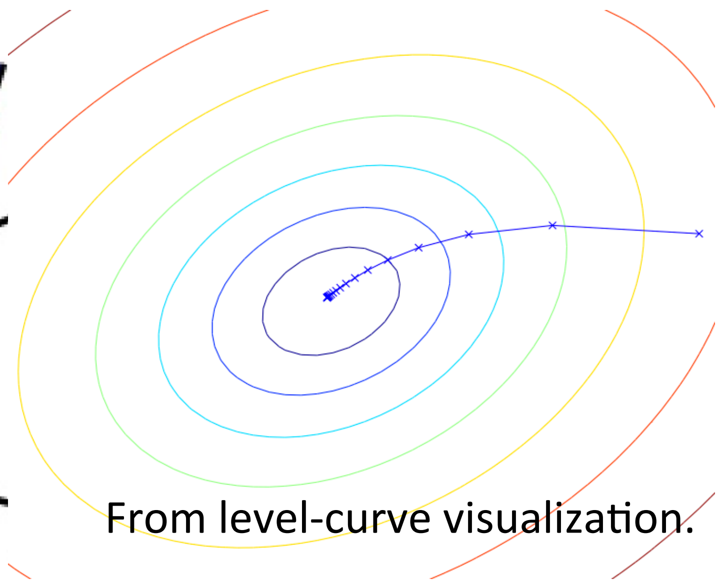
$$\theta_j := \theta_j + \alpha\left(y^{(i)} - h_\theta(x^{(i)})\right)x_j^{(i)}.$$

This rule is called the LMS update rule (or Widrow-Hoff learning rule).

Use the gradient descent algorithm

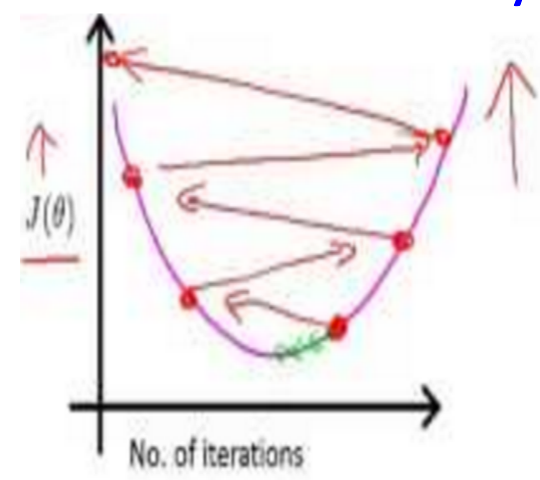
- Which starts with some initial θ, and repeatedly performs the update.
- Here α is called the learning rate.
- Geometrically, it repeatedly takes a step in the direction of steepest decrease of J.





From level-curve visualization.

Make α smaller if necessary.



Batch Gradient Descent (BGD)

Repeat until convergence {

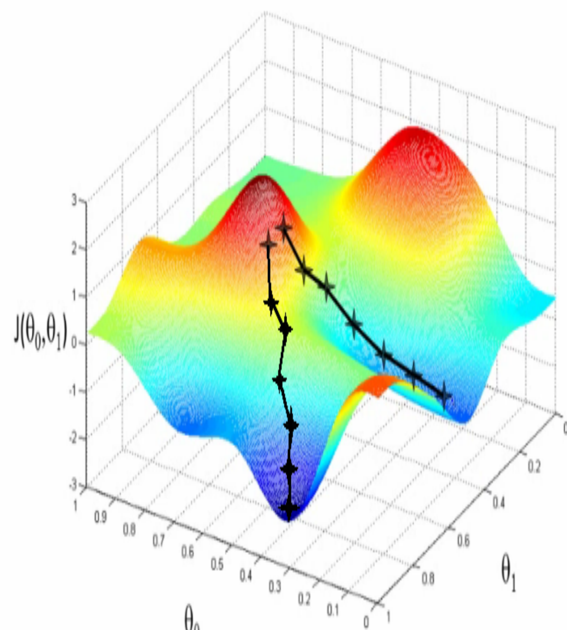
$$\theta_j := \theta_j + \alpha \underbrace{\sum_{i=1}^{m} \left(y^{(i)} - h_\theta(x^{(i)})\right) x_j^{(i)}}_{-\,\partial J(\theta)/\partial \theta_j} \qquad \text{(for every } j\text{).}$$

}

This is simply gradient descent on the original cost function J.

Remarks:

1) **This method looks at every example in the entire training set on every step**, and is called BGD.
2) It is well know that gradient descent can be susceptible to local minima in general (see the figure on right), **the optimization problem we have** posed here for linear regression **has only one global**, and no other local, **optima**; thus gradient descent always converges (assuming the learning rate α is not too large) to the global minimum.
3) **The key is that our J is a convex quadratic function.**



Stochastic Gradient Descent (SGD)

Loop {

for i=1 to m, {

$$\theta_j := \theta_j + \alpha \left(y^{(i)} - h_\theta(x^{(i)}) \right) x_j^{(i)} \qquad \text{(for every } j\text{).}$$

}

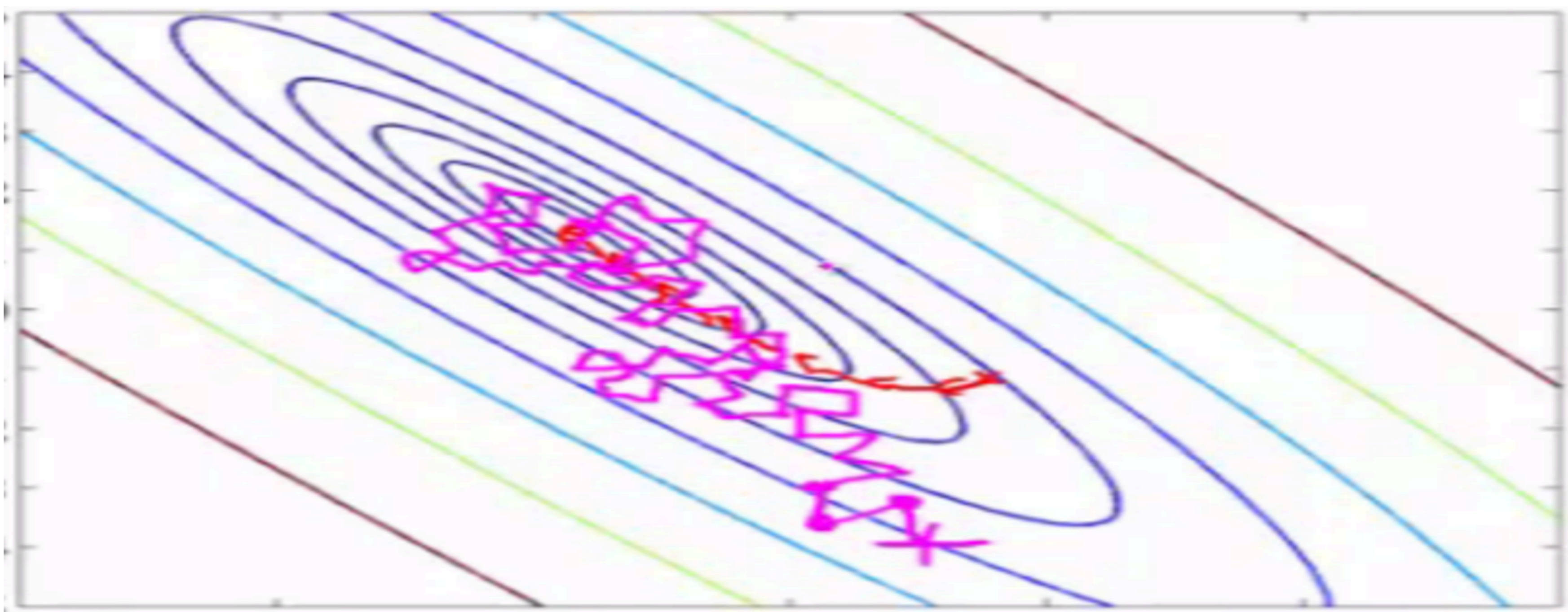
}

Remarks:

1) SGD **repeatedly run through the training set, and each time it encounters a training example, it updates the parameters** according to the gradient of the error with respect to that single training example only.
2) SGD **may never "converge" to the unique minimum**, and the parameters θ will keep oscillating around the minimum of J(θ); but **in practice** most of the values near the minimum will be **reasonably good approximations** to the true minimum.

Comparing Batch gradient descent with Stochastic gradient descent

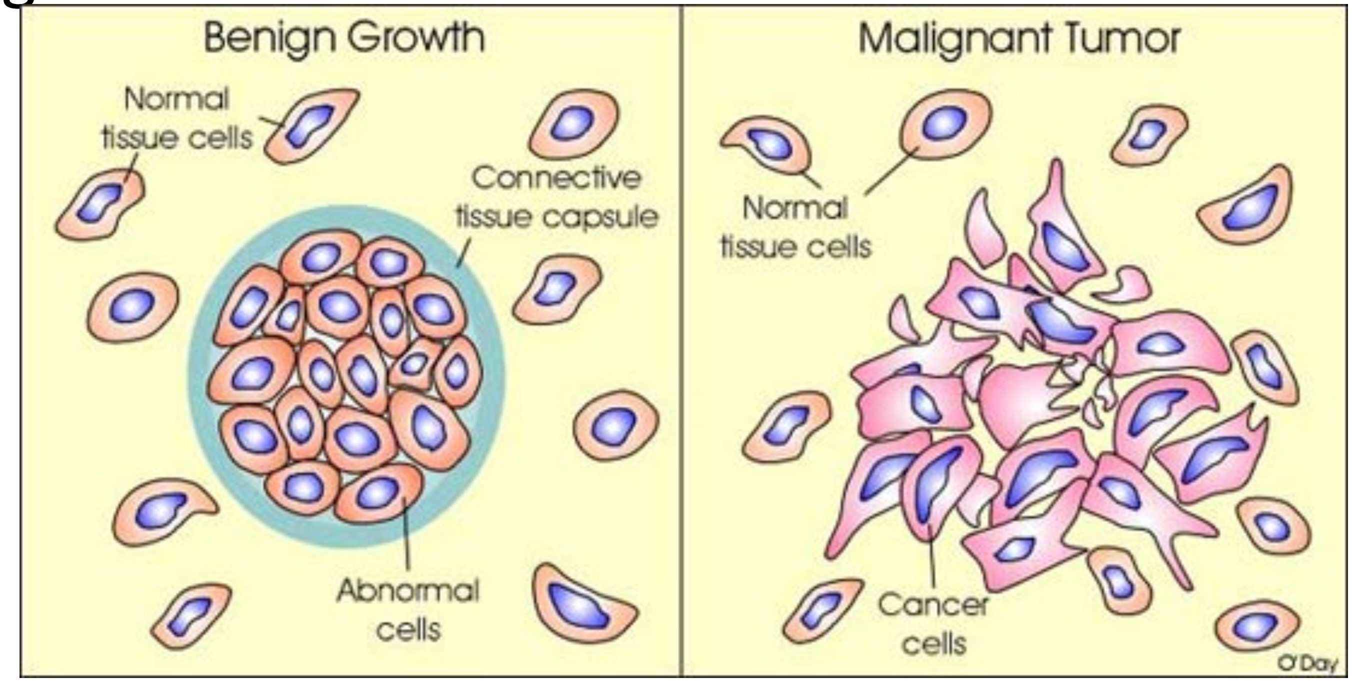
- ***For big data***, often the training set is large***, people prefer use stochastic gradient descent*** instead of batch gradient descent.
- Since *BGD has to scan thru the entire training set before taking a single step*—a costly operation if m is large—*SGD can start making progress right away*, & continues to make progress with each example it looks at.
- *SGD can run on dynamical data sets*. As data coming, it updates the parameters.
- Often, **SGD gets θ "close" to the minimum much faster than BGD**.
- But SGD gets only approximation solution of θ. This is a **trade off** when dealing with big data.



Now we switch gear:

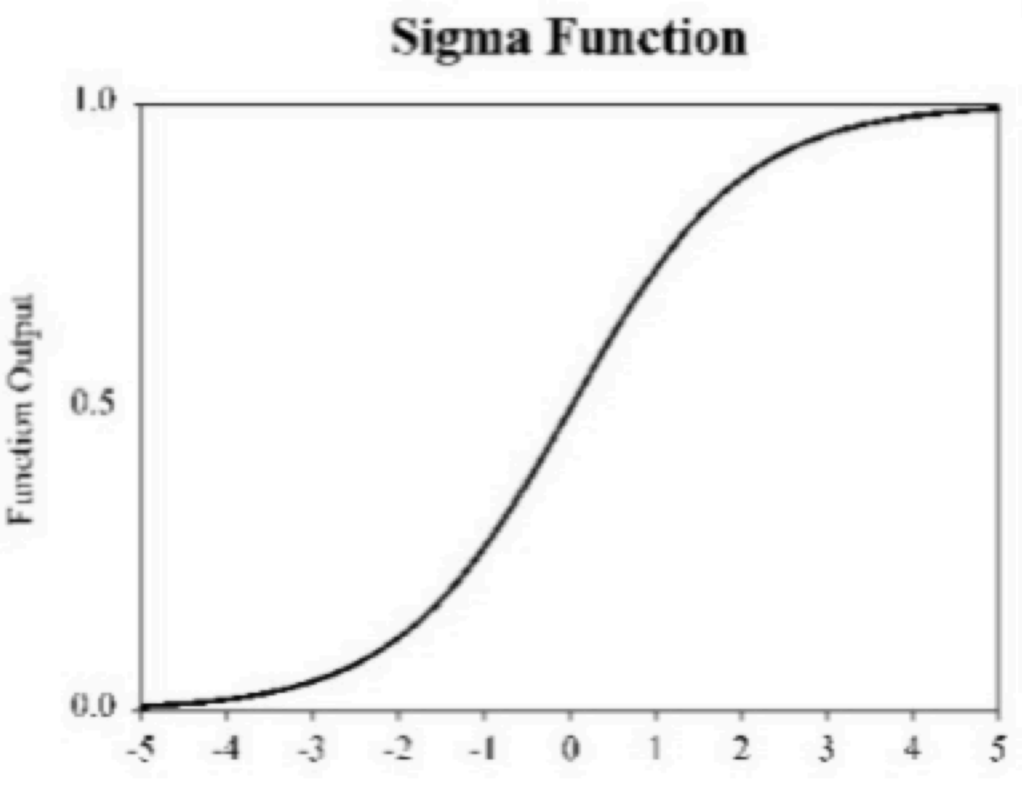
Logistic Regression and Newton's Method

- Key: Logistic Regression is for Classification Problems.
- For example: distinguish between benign tumors and malignant tumors.



Key idea: *try to utilize the linear regression techniques by transform a discrete problem to a smooth problem passing thru a sigma so that we can take gradient for optimization.*

Logistic Regression maps the fitting straight line/ hyperplane in linear regression to a monotone increasing curve, often a ***sigmoid function***.



Work out the details of Logistic regression with the students on the board.

Your homework: Problem 7

(a) Let $\sigma(x) = \frac{1}{1+e^{-x}}$ be the sigmoid function. Show that

$$\sigma'(x) = \sigma(x)\left[1 - \sigma(x)\right].$$

It is easier to maximize the log likelihood:

$$
\begin{aligned}
\ell(\theta) &= \log L(\theta) \\
&= \sum_{i=1}^{m} y^{(i)} \log h(x^{(i)}) + (1 - y^{(i)}) \log(1 - h(x^{(i)}))
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial}{\partial \theta_j} \ell(\theta) &= \left(y \frac{1}{g(\theta^T x)} - (1 - y) \frac{1}{1 - g(\theta^T x)} \right) \frac{\partial}{\partial \theta_j} g(\theta^T x) \\
&= \left(y \frac{1}{g(\theta^T x)} - (1 - y) \frac{1}{1 - g(\theta^T x)} \right) g(\theta^T x)(1 - g(\theta^T x) \frac{\partial}{\partial \theta_j} \theta^T x \\
&= \left(y(1 - g(\theta^T x)) - (1 - y) g(\theta^T x) \right) x_j \\
&= (y - h_\theta(x))\, x_j
\end{aligned}
$$

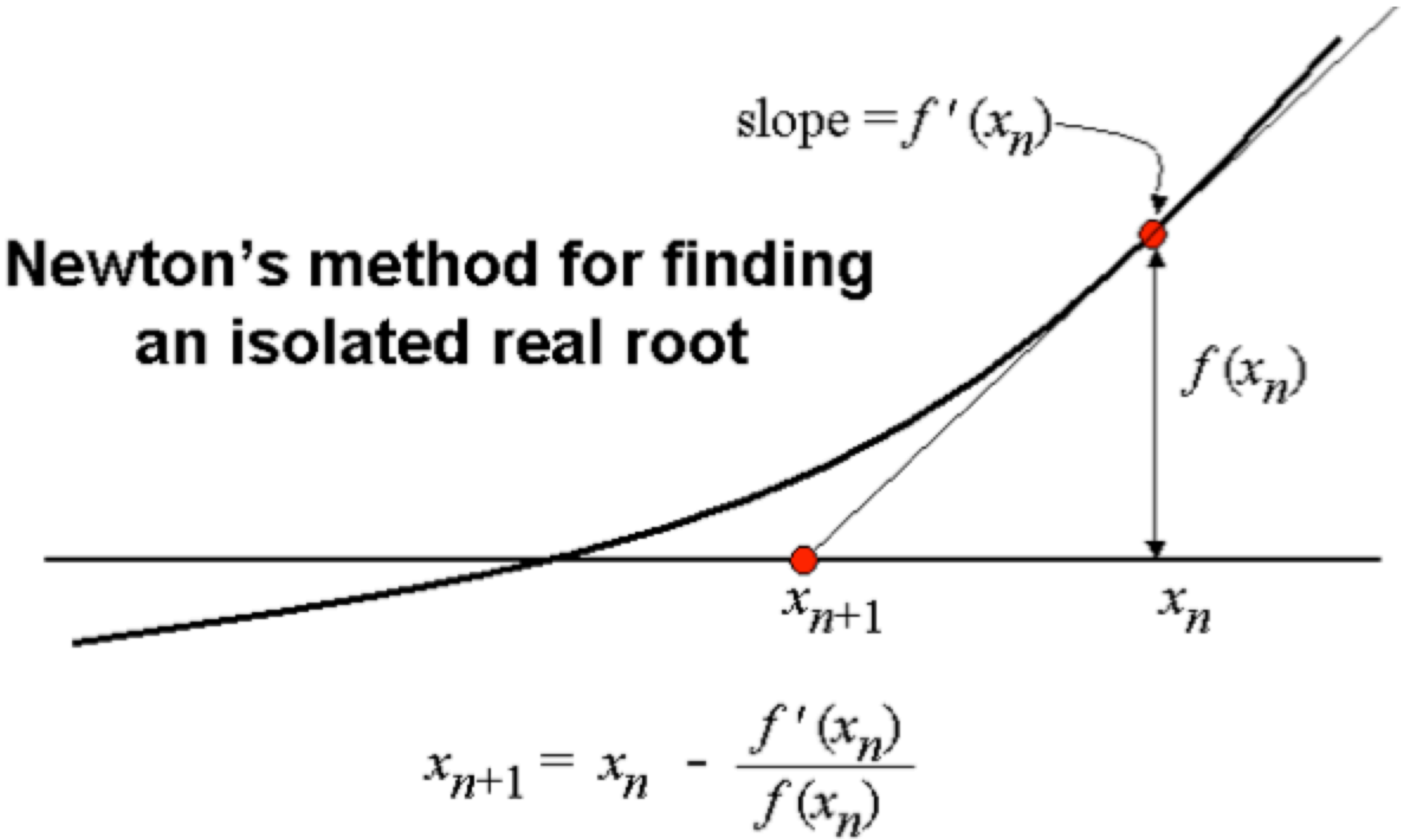
This gives us the stochastic gradient ascent rule:

$$
\theta_j := \theta_j + \alpha \left(y^{(i)} - h_\theta(x^{(i)}) \right) x_j^{(i)}
$$

Note : There is another method running even fast than this one, called Newton's method.

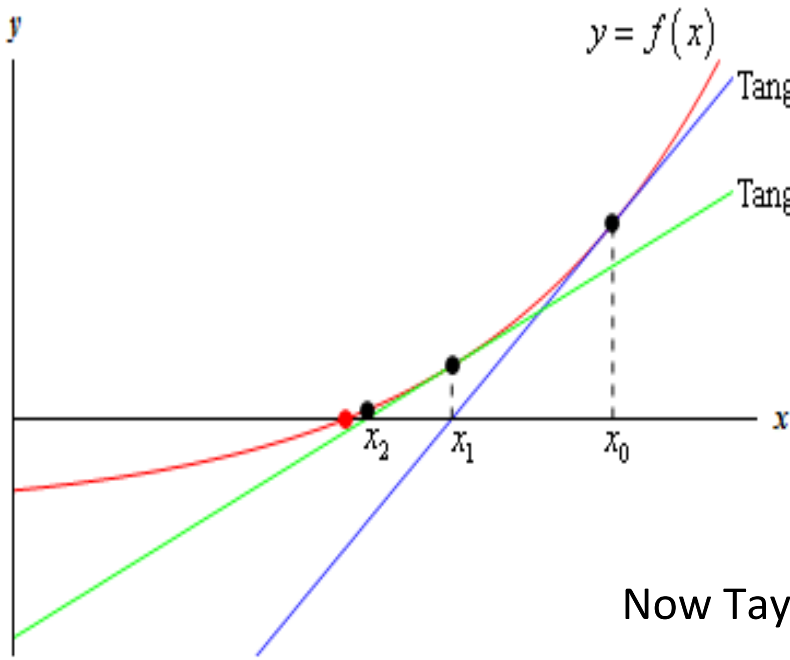
Newton's method for fast computation

In the case of line, we just use the definition of the slope of f.



Newton's method for fast computation

- <u>Case 1</u> Let f : **R** →**R** (here, we just use the definition of the slope of f.)
- Newton's method for finding an isolated real root
- *Key*: In general using Taylor expansion at $x = x_0$
- Take the linear best approximation & plug in $x = x_1$.



$$0 = f(x_0) + f'(x_0)(x_1 - x_0)$$

$$x_1 - x_0 = -\frac{f(x_0)}{f'(x_0)}$$

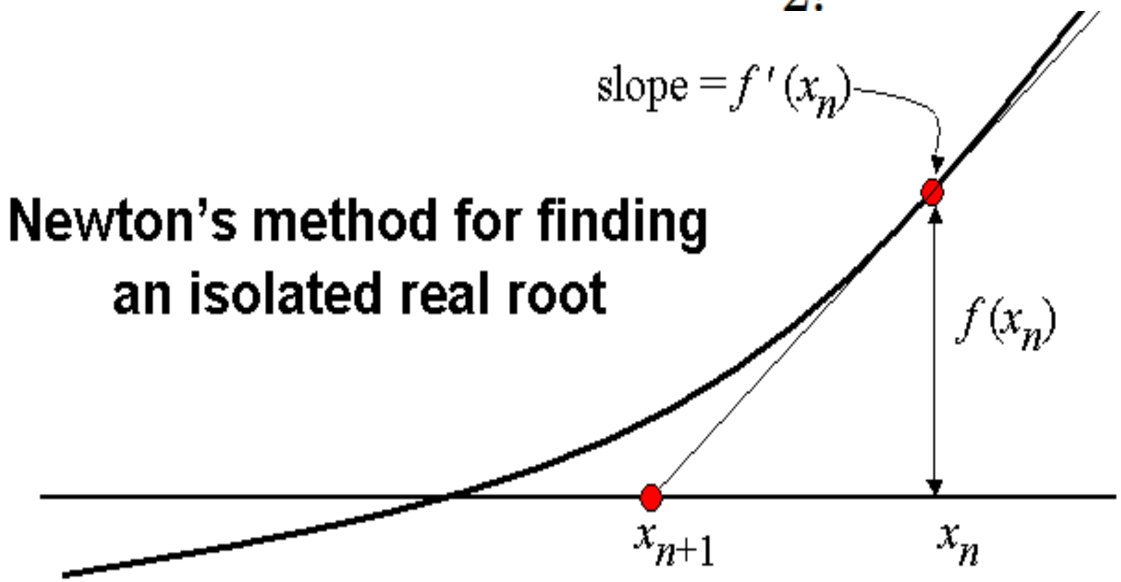
$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$$

Now Taylor expanding of f at $x = x_1$ and similarly finding x_2.

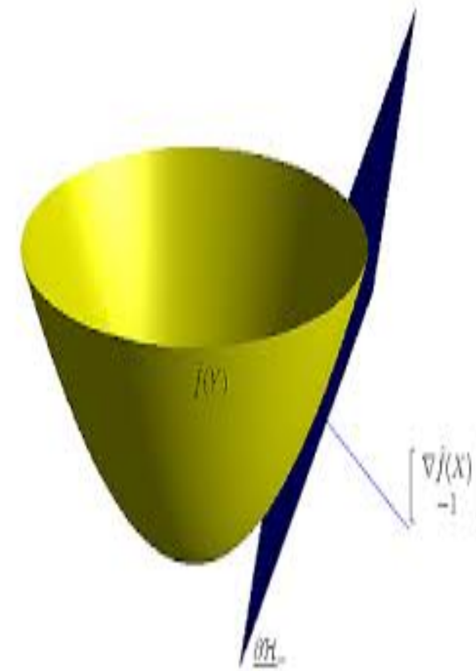
Iteratively: Taylor expanding of f at $x = x_n$, plugging in $x = x_{n+1}$, and solving x_{n+1}.

Case 2 Multivariable:

$$f(\vec{x}) = f(\vec{a}) + (\vec{x}-\vec{a})^T \nabla f(\vec{a}) + \frac{1}{2!}(\vec{x}-\vec{a})^T H_f(\vec{a})(\vec{x}-\vec{a}) + \cdots$$



$$x_{n+1} = x_n - \frac{f'(x_n)}{f(x_n)}$$



Newton's method is for finding a root of a function.
Keys: Taylor expansion, plug into linear part, solve, then iterate.

Now we switch gear again:

Generalized Linear Models (GLMs)

- This topic includes: ***exponential family*** & ***Softmax Regression***.
- *What is an exponential family?* A class of distributions is in the exponential family if

$$\boxed{p(y;\eta) = b(y)\exp(\eta^T T(y) - a(\eta))}$$

- η = the natural parameter (or the canonical parameter) of the distribution
- $T(y)$ = the sufficient statistic (often $T(y) = y$)
- $a(\eta)$ is the log partition function.

The quantity $e^{-a(\eta)}$ essentially plays the role of a normalization constant, that makes sure the distribution $p(y; \eta)$ sums/integrates over y to 1.

Let T, a and b fixed and let the parameter η vary, then it defines a family of distribution.
i.e. We get different distributions within this family.

Let's first show

Bernoulli distributions are exponential family distribution.

- Work out details with the students on the board.

Let's first show

Gaussian distributions are exponential family distribution.

$$
\begin{aligned}
p(y;\mu) &= \frac{1}{\sqrt{2\pi}}\exp\left(-\frac{1}{2}(y-\mu)^2\right) \\
&= \frac{1}{\sqrt{2\pi}}\exp\left(-\frac{1}{2}y^2\right)\cdot\exp\left(\mu y-\frac{1}{2}\mu^2\right)
\end{aligned}
$$

Compare:

$$\boxed{p(y;\eta) = b(y)\exp(\eta^T T(y) - a(\eta))}$$

We get:

$$
\begin{aligned}
\eta &= \mu \\
T(y) &= y \\
a(\eta) &= \mu^2/2 \\
&= \eta^2/2 \\
b(y) &= (1/\sqrt{2\pi})\exp(-y^2/2).
\end{aligned}
$$

Constructing GLMs

Note: you need to know which distribution models what kind of problems

(Reading assignment)

- Suppose you want to build a model to estimate the number (y) of customers arriving in your store in any given hour, based on certain features x such as store promotions, recent advertising, weather, day-of-week, etc.
- We know that the Poisson distribution usually gives a good model for numbers of visitors.
- Knowing this, how can we come up with a model for this problem?
- Fortunately, the Poisson is an exponential family distribution, so we can apply a Generalized Linear Model (GLM). *(Homework or exam problem?)*
- Lots of known distributions are exponential families.
- Here, we will describe a method for constructing GLM models for problems such as these.

Assumptions for Generalized Linear Models

- In generally, consider a classification or regression problem where we would like to predict the value of some random variable y as a function of x.
- To derive a GLM for this problem, we will make the following three assumptions about the conditional distribution of y given x and about our model:
- **1. $y \mid x; \theta \sim$ Exponential Family(η)**. I.e., given x and θ, the distribution of y follows some exponential family distribution, with parameter η.
- **2**. Given x, our goal is to predict the expected value of T(y) given x. Since often T(y) = y, so this means we would like the prediction **h(x) output by our learned hypothesis h to satisfy h(x) = E[y|x].** (Note that this assumption is satisfied in the choices for $h_\theta(x)$ for both logistic regression and linear regression. For instance, in logistic regression, we had

$h_\theta(x) = p(y = 1|x; \theta) = 0 \cdot p(y = 0|x; \theta) + 1 \cdot p(y = 1|x; \theta) = E[y|x; \theta]$.)

- **3. The natural parameter η and the inputs x are related linearly: $\eta = \theta^T x$**. (Or, if η is vector-valued, then $\eta_i = \theta_i^T x$.)

Examples: Least square and Logistic regression are GLM family of models

$$\begin{aligned} h_\theta(x) &= E[y|x;\theta] \\ &= \mu \\ &= \eta \\ &= \theta^T x. \end{aligned}$$

$$\begin{aligned} h_\theta(x) &= E[y|x;\theta] \\ &= \phi \\ &= 1/(1+e^{-\eta}) \\ &= 1/(1+e^{-\theta^T x}) \end{aligned}$$

Given that y is binary-valued, it therefore seems natural to choose the Bernoulli family of distributions to model the conditional distribution of y given x. In our formulation of the Bernoulli distribution as an exponential family distribution, we had $\phi = 1/(1 + e^{-\eta})$. Furthermore, note that if $y|x; \theta \sim \text{Bernoulli}(\phi)$, then $E[y|x; \theta] = \phi$.

Softmax Regression

- Let's look at another example of a GLM. Consider a classification problem in which the response variable y $\in \{1, 2, \ldots, k\}$.
- For example, rather than classifying email into the two classes spam or not-spam—which would have been a binary classification problem— this time we want to classify it into four classes, such as spam, family-mail, friends-mail, and work-related mail. The response variable is still discrete, but can now take on more than two values. We will thus model it as distributed according to a multinomial distribution.

Let's Derive

A GLM using Multinomial distributions as exponential family distribution.

- **What are Multinomial distributions?**
- **For example:** If a 6 sided die has
  - 3 faces painted red
  - 2 faces painted white
  - 1 faces painted blue

  And rolled 100 times.
  Find P(60 red, 30 white, and 10 blue).

  Work out details with the students on the board.

  Generally an experiment with m outcomes with respective probabilities $p_1, p_2, \ldots, p_m$ is performed n times independently.
  Let x_i = # of times outcome i appears, i=1,2,…,m
  Then $P(x_1=k_1, x_2=k_2, \ldots, x_m = k_m) = ?$

- Work out details with the students on the board.

Details of Softmax Regression

- Work out details with the students on the board.