ISTA 421 + INFO 521
Introduction to Machine Learning

Lecture 4:
Geometry of LLMS,
Nonlinear response (basis fns)

Clay Morrison
claytonm@email.arizona.edu
Harvill 437A
Phone 621-6609

30 August 2017

1

Next Topics

- Moving to higher dimensions
  - Linear Algebra: matrix operators
  - Some Geometry of Linear Algebra
  - Least Mean Squares in Matrix formulation
  - The Geometry of LMS solution
- Nonlinear Response: Basis Functions
- Model Selection
  - Generalization and Overfitting
  - Method 1: Cross Validation
- Regularized Least Squares

2

Simple Linear Model in Matrix Notation

- First, express our original 1-variable, 2-param model in matrix notation:

$$\mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} \qquad \mathbf{x}_n = \begin{bmatrix} 1 \\ x_n \end{bmatrix}$$

$$f(x_n; w_0, w_1) = \mathbf{w}^\top \mathbf{x}_n = w_0 + w_1 x_n$$

$$\mathcal{L} = \frac{1}{N} \sum_{n=1}^{N} (t_n - \mathbf{w}^\top \mathbf{x}_n)^2$$

3

Simple Linear Model in Matrix Notation

$$\mathcal{L} = \frac{1}{N} \sum_{n=1}^{N} (t_n - \mathbf{w}^\top \mathbf{x}_n)^2$$

- Next, express the operations involving all of the data (the inputs $\mathbf{x}_n$ and the targets $\mathbf{t}_n$):

$$\mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \end{bmatrix}, \quad \mathbf{x}_n = \begin{bmatrix} 1 \\ x_n \end{bmatrix}$$

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1^\top \\ \mathbf{x}_2^\top \\ \vdots \\ \mathbf{x}_N^\top \end{bmatrix} = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_N \end{bmatrix}, \quad \mathbf{t} = \begin{bmatrix} t_1 \\ t_2 \\ \vdots \\ t_N \end{bmatrix}$$

Design Matrix

$$\mathbf{X}\mathbf{w} = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_N \end{bmatrix} \times \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} = \begin{bmatrix} w_0 + w_1 x_1 \\ w_0 + w_1 x_2 \\ \vdots \\ w_0 + w_1 x_N \end{bmatrix}$$

$$\mathbf{t} - \mathbf{X}\mathbf{w} = \begin{bmatrix} t_1 - w_0 - w_1 x_1 \\ t_2 - w_0 - w_1 x_2 \\ \vdots \\ t_N - w_0 - w_1 x_N \end{bmatrix}$$

$$\begin{aligned} (\mathbf{t} - \mathbf{X}\mathbf{w})^\top (\mathbf{t} - \mathbf{X}\mathbf{w}) &= (t_1 - (w_0 + w_1 x_1))^2 + (t_2 - (w_0 + w_1 x_2))^2 + \ldots \\ &\quad + (t_N - (w_0 + w_1 x_N))^2 \\ &= \sum_{n=1}^{N} (t_n - (w_0 + w_1 x_n))^2 \\ &= \sum_{n=1}^{N} (t_n - f(x_n; w_0, w_1))^2 \end{aligned}$$

$$\mathcal{L} = \frac{1}{N} (\mathbf{t} - \mathbf{X}\mathbf{w})^\top (\mathbf{t} - \mathbf{X}\mathbf{w}) = \frac{1}{N} \sum_{n=1}^{N} (t_n - \mathbf{w}^\top \mathbf{x}_n)^2 = \frac{1}{N} \sum_{n=1}^{N} (t_n - (w_0 + w_1 x_n))^2$$

Much nicer! The $\mathbf{x}^\top \mathbf{y}$ operation allows us to drop the sums!

4

Simple Linear Model in Matrix Notation

- Now that we have the matrix version of the loss function, "just" take derivative…

$$\begin{aligned}\mathcal{L} &= \frac{1}{N}(\mathbf{Xw}-\mathbf{t})^\top(\mathbf{Xw}-\mathbf{t}) \\ &= \frac{1}{N}((\mathbf{Xw})^\top-\mathbf{t}^\top)(\mathbf{Xw}-\mathbf{t}) \\ &= \frac{1}{N}(\mathbf{Xw})^\top\mathbf{Xw}\boxed{-\frac{1}{N}\mathbf{t}^\top\mathbf{Xw}-\frac{1}{N}(\mathbf{Xw})^\top\mathbf{t}}+\frac{1}{N}\mathbf{t}^\top\mathbf{t} \\ &= \frac{1}{N}\mathbf{w}^\top\mathbf{X}^\top\mathbf{Xw}-\frac{2}{N}\mathbf{w}^\top\mathbf{X}^\top\mathbf{t}+\frac{1}{N}\mathbf{t}^\top\mathbf{t}\end{aligned}$$

note: book accidentally drops the 1/N here

$\mathbf{t}^\top\mathbf{Xw}$ and $\mathbf{w}^\top\mathbf{X}^\top\mathbf{t}$ are the transpose of one another
… and both products come out to be **scalars** (i.e., "1x1 matrices", where transpose of one is the same as the other), so the products are the same and can be combined.

$$\frac{\partial\mathcal{L}}{\partial\mathbf{w}} = \begin{bmatrix}\frac{\partial\mathcal{L}}{\partial w_0} \\ \frac{\partial\mathcal{L}}{\partial w_1}\end{bmatrix}$$

$f(\mathbf{w})$	$\frac{\partial f}{\partial \mathbf{w}}$
$\mathbf{w}^\top\mathbf{x}$	$\mathbf{x}$
$\mathbf{x}^\top\mathbf{w}$	$\mathbf{x}$
$\mathbf{w}^\top\mathbf{w}$	$2\mathbf{w}$
$\mathbf{w}^\top\mathbf{C}\mathbf{w}$	$2\mathbf{C}\mathbf{w}$

Some useful identities

$$\begin{aligned}\frac{\partial\mathcal{L}}{\partial\mathbf{w}} &= \frac{2}{N}\mathbf{X}^\top\mathbf{Xw}-\frac{2}{N}\mathbf{X}^\top\mathbf{t}=0 \\ \mathbf{X}^\top\mathbf{Xw} &= \mathbf{X}^\top\mathbf{t}.\end{aligned}$$

$$\mathbf{Iw} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{t}$$

The **matrix normal equation**!
(guaranteed unique solution when the *n* column vectors of **X** are *linearly independent*)

But what does this *mean*??

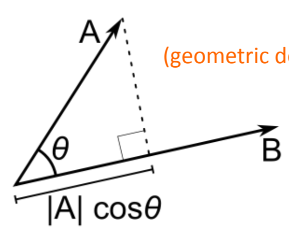
5

Relation of $a^\mathsf{T}b$ to Geometry

- $\mathbf{a}^\mathsf{T}\mathbf{b}$ is special (also $\mathbf{a}\cdot\mathbf{b}$), called the *dot product*
(aka *scalar product*; the *inner product* for the Euclidean space)

$$\mathbf{a}\cdot\mathbf{b} = \sum_{i=1}^{n} a_i b_i = a_1b_1 + a_2b_2 + \cdots + a_nb_n$$ (algebraic def.)

- Plays a role in defining
  - Euclidean distance (norm)
  - Angles



(geometric def.)

$$\mathbf{a}\cdot\mathbf{a} = \|\mathbf{a}\|^2$$
$$\mathbf{a}\cdot\mathbf{b} = \|\mathbf{a}\|\,\|\mathbf{b}\|\cos\theta$$
$$\theta = \arccos\left(\frac{\mathbf{a}\cdot\mathbf{b}}{\|\mathbf{a}\|\,\|\mathbf{b}\|}\right).$$

The dot product of vectors that are 90° (or more generally, orthogonal) is = 0

6

Geometry of Linear Systems and their Solution

A linear equation expresses a constraint between variables

A system of linear equations – more constraints!

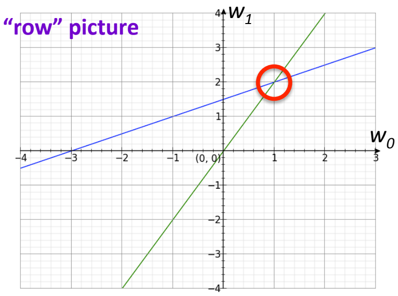
$$2w_0 - w_1 = 0$$
$$-w_0 + 2w_1 = 3$$

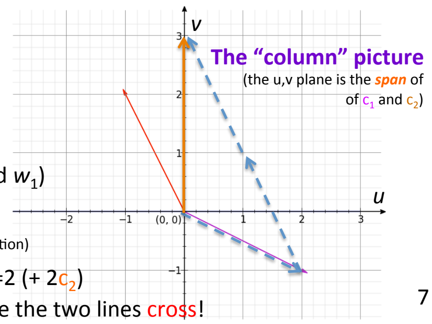
$$\begin{array}{c c} u \\ v \\ \end{array} \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} = \begin{bmatrix} 0 \\ 3 \end{bmatrix}$$

c_1 c_2

$\mathbf{X\,w = t}$ $\mathbf{w = X^{-1}\,t}$

The "row" picture





"Solving" the linear system involves finding the linear combinations (i.e., the amounts w_0 and w_1) of c_1 and c_2 that equal the column vector $(0,3)^T$.

Solve using your favorite method (e.g., Gaussian Elimination)

Here, the solution happens to be w_0=1 (1 c_1), w_1=2 (+ 2c_2)

The w_0's and w_1's corresponds to the point where the two lines cross!

Geometry of Linear Systems and their Solution

But what happens if we're **over-constrained**?

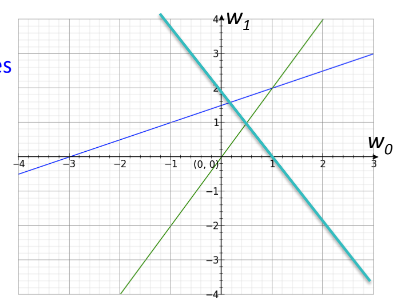
GOAL: Find a solution that is ***closest*** to (minimizes the distance between) the crossing points!

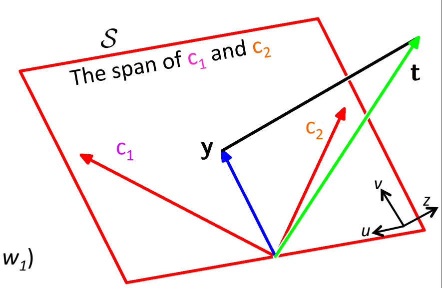
$$2w_0 - w_1 = 0$$
$$-w_0 + 2w_1 = 3$$
$$2w_0 + w_1 = 2$$

$$\begin{array}{c} u \\ v \\ z \end{array} \begin{bmatrix} 2 & -1 \\ -1 & 2 \\ 2 & 1 \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} = \begin{bmatrix} 0 \\ 3 \\ 2 \end{bmatrix}$$

c_1 c_2 t

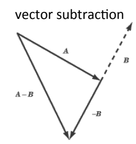
$\mathbf{X\,w = t}$ $\mathbf{X\,\hat{w} = y}$





"Solving" the linear system involves finding the linear combinations (i.e., the amounts w_0 and w_1) of c_1 and c_2 that equal the column vector $(0,3,2)^T$.

Finding the projection

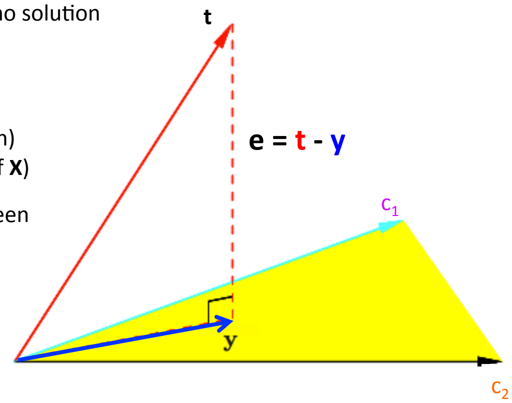


X w = **t** …what we started with, but no solution

X ŵ = **y** …something we can solve

In particular, we want **y** to be the ***closest*** to **t** but in the column space (the span) of $\mathbf{c}_1$ and $\mathbf{c}_2$ (which are the columns of **X**)

But we know the shortest distance between the end of **t** and the span is a vector **e** that is ***orthogonal*** (right angles!) to $\mathbf{c}_1$ and $\mathbf{c}_2$ (i.e., orthogonal to **X**)

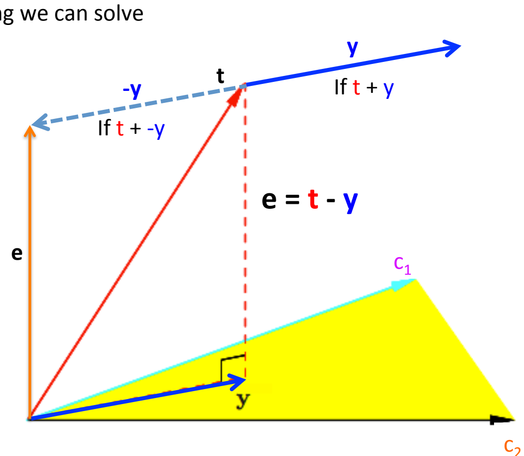


9

In case you need to remember vector subtraction…

X w = **t** …what we started with, but no solution

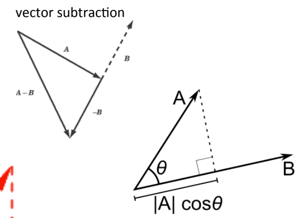
X ŵ = **y** …something we can solve



Now we want to enforce: **e**'s <u>length</u> to be *minimal* and <u>direction</u> *orthogonal* to **X**

10

Finding the projection



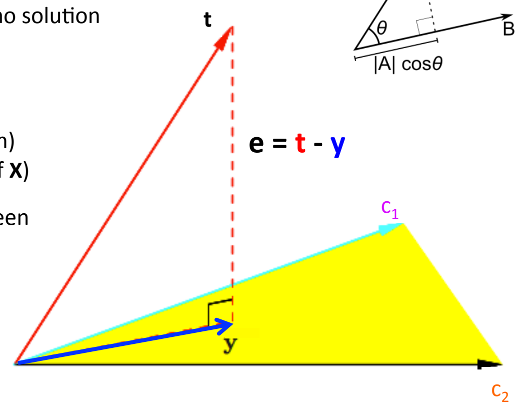
$\mathbf{X}\ \mathbf{w} = \mathbf{t}$ ...what we started with, but no solution

$\mathbf{X}\ \hat{\mathbf{w}} = \mathbf{y}$ ...something we can solve

In particular, we want **y** to be the ***closest*** to **t** but in the column space (the span) of $\mathbf{c}_1$ and $\mathbf{c}_2$ (which are the columns of **X**)

But we know the shortest distance between the end of **t** and the span is a vector **e** that is ***orthogonal*** (right angles!) to $\mathbf{c}_1$ and $\mathbf{c}_2$ (i.e., orthogonal to **X**)

That only happens when $\mathbf{e}^T\mathbf{X} = \mathbf{X}^T\mathbf{e} = 0$



Let's solve for when $\mathbf{X}^T\mathbf{e} = 0$

$\mathbf{X}^T(\mathbf{t} - \mathbf{y}) = \mathbf{0}$	plug in for **e**
$\mathbf{X}^T(\mathbf{t} - \mathbf{X}\hat{\mathbf{w}}) = \mathbf{0}$	substitute original $\mathbf{X}\hat{\mathbf{w}}=\mathbf{y}$ (since we don't know **y**)
$\mathbf{X}^T\mathbf{t} - \mathbf{X}^T\mathbf{X}\hat{\mathbf{w}} = \mathbf{0}$	multiply through...
$\mathbf{X}^T\mathbf{X}\hat{\mathbf{w}} = \mathbf{X}^T\mathbf{t}$	
$\hat{\mathbf{w}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{t}$	... ah hah! $\mathbf{I}\mathbf{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{t}$

11

Finding the projection

Parameters of hyperplane

$$2w_0 - w_1 = 0$$
$$-w_0 + 2w_1 = 3$$
$$2w_0 + w_1 = 2$$

inputs   targets

$\mathbf{X}\ \mathbf{w} = \mathbf{t}$ ...what we started with, but no solution

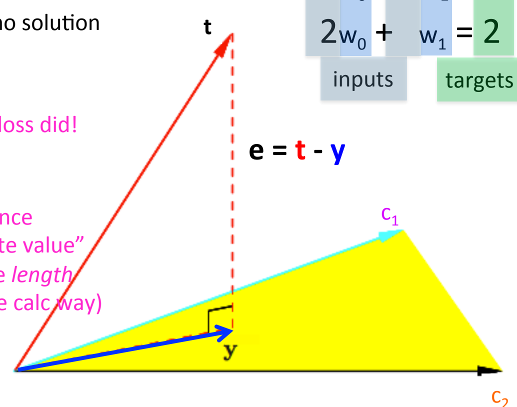
$\mathbf{X}\ \hat{\mathbf{w}} = \mathbf{y}$ ...something we can solve

This is just what minimizing the squared loss did!

$$\mathcal{L} = \frac{1}{N}(\mathbf{t} - \mathbf{X}\mathbf{w})^T(\mathbf{t} - \mathbf{X}\mathbf{w})$$

This "error" vector **e** is the shortest distance
(Note: we can't differentiate the "absolute value" so the squaring of the difference – of the *length* of **e** – was a better choice for doing it the calc way)

That only happens when $\mathbf{e}^T\mathbf{X} = \mathbf{X}^T\mathbf{e} = 0$



Let's solve for when $\mathbf{X}^T\mathbf{e} = 0$

$\mathbf{X}^T(\mathbf{t} - \mathbf{y}) = \mathbf{0}$	plug in for **e**
$\mathbf{X}^T(\mathbf{t} - \mathbf{X}\hat{\mathbf{w}}) = \mathbf{0}$	substitute original $\mathbf{X}\hat{\mathbf{w}}=\mathbf{y}$ (since we don't know **y**)
$\mathbf{X}^T\mathbf{t} - \mathbf{X}^T\mathbf{X}\hat{\mathbf{w}} = \mathbf{0}$	multiply through...
$\mathbf{X}^T\mathbf{X}\hat{\mathbf{w}} = \mathbf{X}^T\mathbf{t}$	
$\hat{\mathbf{w}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{t}$	... ah hah! $\mathbf{I}\mathbf{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{t}$

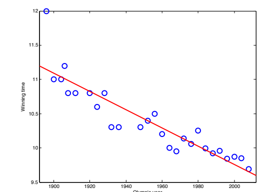
12

The Normal Equations

For model: $t = f(x_1, ..., x_k; w_0, ..., w_k) = \sum_{i=0}^{k} x_i w_i$

$$w_0 = \bar{t} - w_1 x$$

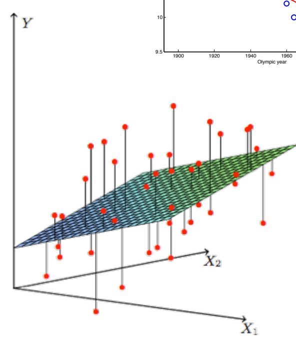
$$w_1 = \frac{\overline{xt} - \overline{x}\overline{t}}{\overline{x^2} - (\overline{x})^2}$$



$$\hat{\mathbf{w}} = \left(\mathbf{X}^\top \mathbf{X}\right)^{-1} \mathbf{X}^\top \mathbf{t}$$

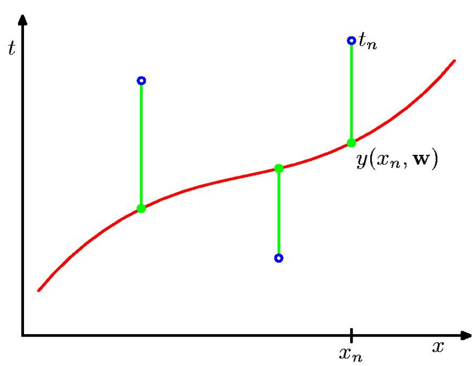
$$\mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \end{bmatrix}, \quad \mathbf{x}_n = \begin{bmatrix} 1 \\ x_n \end{bmatrix}$$

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1^\top \\ \mathbf{x}_2^\top \\ \vdots \\ \mathbf{x}_N^\top \end{bmatrix} = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_N \end{bmatrix}, \quad \mathbf{t} = \begin{bmatrix} t_1 \\ t_2 \\ \vdots \\ t_N \end{bmatrix}$$



13

Sum-of-Squares Loss (Error) Function



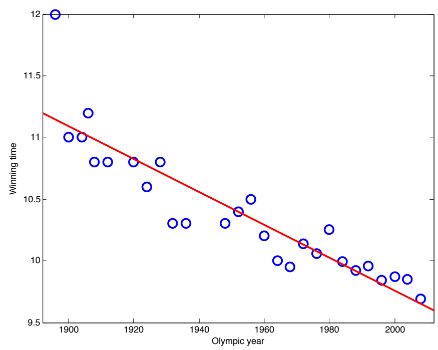
$$\mathcal{L} = \frac{1}{N}(\mathbf{t} - \mathbf{X}\mathbf{w})^\top(\mathbf{t} - \mathbf{X}\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N}(t_n - \mathbf{w}^\top \mathbf{x}_n)^2 = \frac{1}{N}\sum_{n=1}^{N}(t_n - (w_0 + w_1 x_n))^2$$

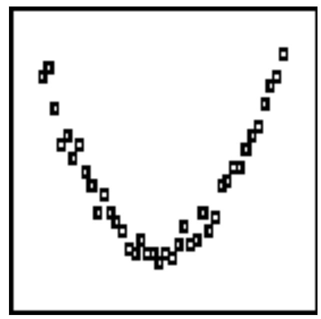
$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2$$

Another formulation, from Bishop (2006)

14

Linear (in variables) has its limit!



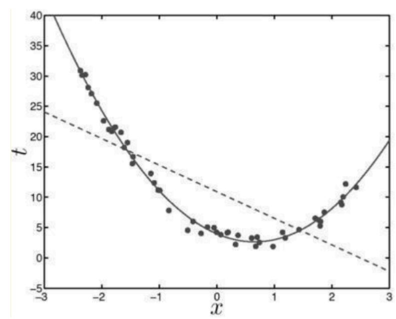


Nonlinear Response

- We can extend the power of linear LMS best fit to models that have a non-linear **response**.

$$f(x; \mathbf{w}) = \mathbf{w}^T \mathbf{x} = w_0 + w_1 x + w_2 x^2$$

$$\mathbf{x}_n = \begin{bmatrix} 1 \\ x_n \\ x_n^2 \end{bmatrix}, \quad \mathbf{X} = \begin{bmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ \vdots & \vdots & \vdots \\ 1 & x_N & x_N^2 \end{bmatrix}$$

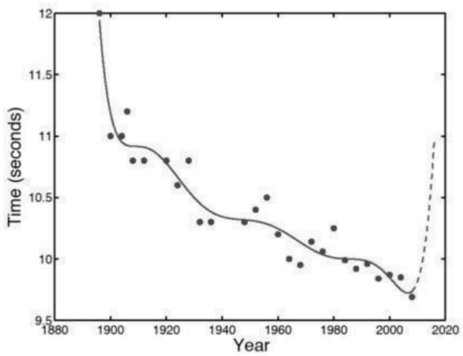


Fitting the parameters **w** still works the same! The only difference is that we square the *x* values *at the input phase* (for each of the elements of the third column vector)

Generalize to Models of k^th-order Polynomials

$$f(x; \mathbf{w}) = \sum_{k=0}^{K} w_k x^k$$

$$\mathbf{X} = \begin{bmatrix} x_1^0 & x_1^1 & x_1^2 & \cdots & x_1^K \\ x_2^0 & x_2^1 & x_2^2 & \cdots & x_2^K \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ x_N^0 & x_N^1 & x_N^2 & \cdots & x_N^K \end{bmatrix}$$



Note: this is **not** creating more ***independent*** sources of information about individuals, but it ***IS*** giving the model the capacity to consider **non-linear *components*** of what original inputs there are.

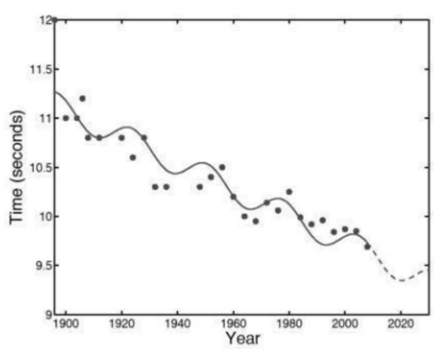
And we're still just learning **LINEAR COMBINATIONS** of those **components**

17

Linear Combination of *Basis Functions* (not just polynomials)

$$\mathbf{X} = \begin{bmatrix} h_1(x_1) & h_2(x_1) & \cdots & h_K(x_1) \\ h_1(x_2) & h_2(x_2) & \cdots & h_K(x_2) \\ \vdots & \vdots & \cdots & \vdots \\ h_1(x_N) & h_2(x_N) & \cdots & h_K(x_N) \end{bmatrix}$$

$$h_1(x) = 1$$
$$h_2(x) = x$$
$$h_3(x) = \sin\left(\frac{x-a}{b}\right)$$
$$f(x; \mathbf{w}) = w_0 + w_1 x + w_2 \sin\left(\frac{x-a}{b}\right).$$

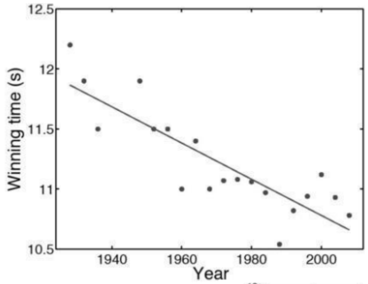


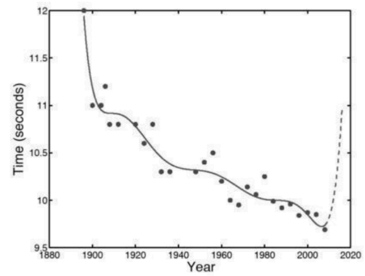
Careful !!
a and *b* must be **constants**

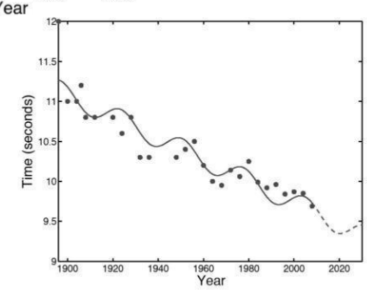
All parameters (as variables) must be ***linearly*** combined

18

Which Model is better: 1st order, 8th order ?



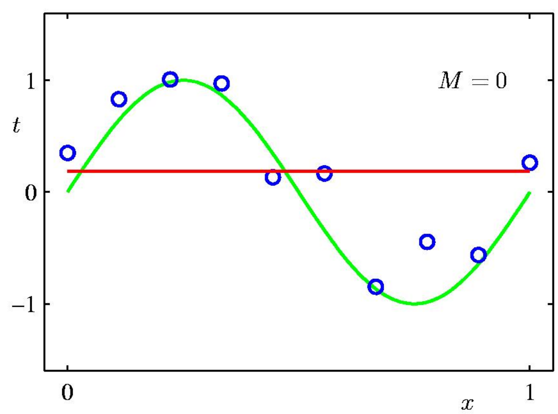




… periodic?

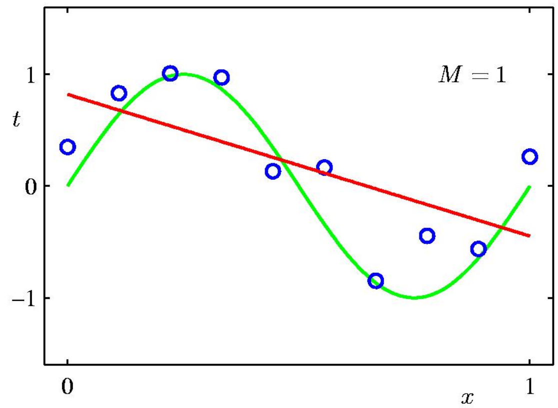
19

0th Order Polynomial



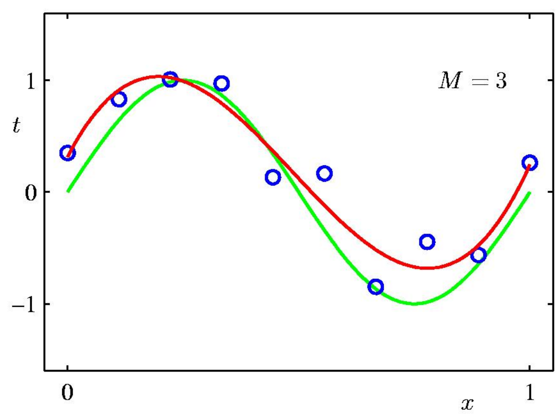
20

1st Order Polynomial



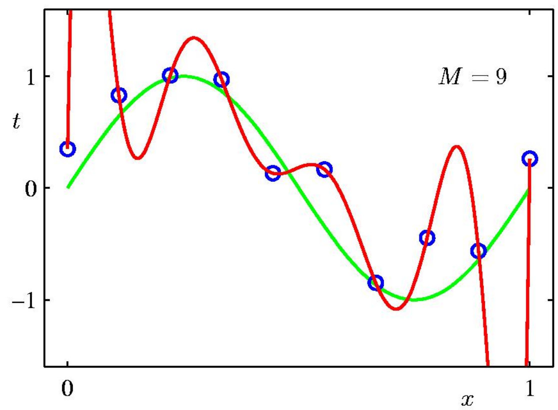
21

3rd Order Polynomial



22

9^{th} Order Polynomial



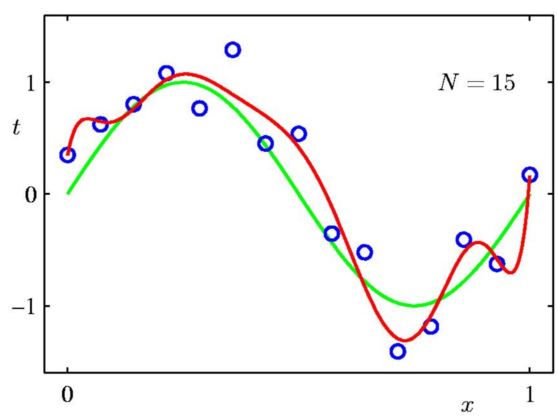
Overfitting

23

Data Set Size:

$N = 15$

9^{th} Order Polynomial

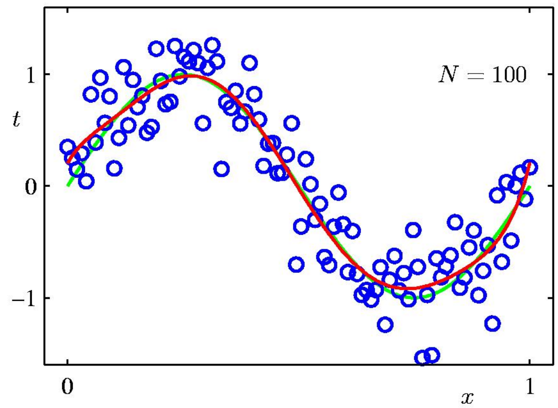


24

Data Set Size:

$N = 100$

9th Order Polynomial



25