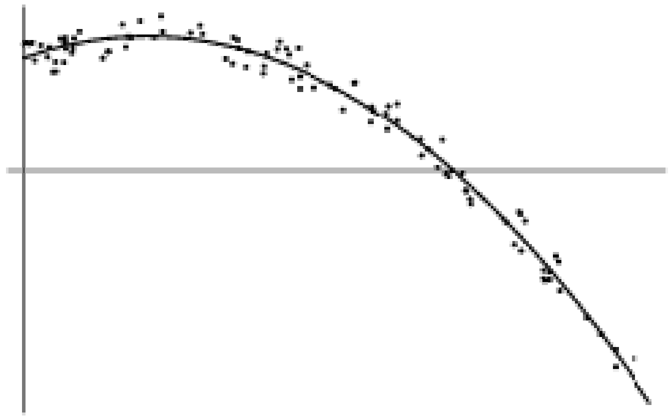
Least Squares

Nuno Vasconcelos
(Ken Kreutz-Delgado)

UCSD

(Unweighted) Least Squares

- **Assume linearity in the unknown, deterministic model parameters θ**
- **Scalar, additive noise model:**



$$\boxed{y = f(x;\theta) + \varepsilon = \gamma(x)^T\theta + \varepsilon}$$

  – E.g., for a line (***f*** affine in ***x***),

$$f(x;\theta) = \theta_1 x + \theta_0 \qquad \gamma(x) = \begin{bmatrix} 1 \\ x \end{bmatrix} \qquad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \end{bmatrix}$$

- This can be **generalized to arbitrary functions of *x***:

$$\boxed{\gamma(x) = \begin{bmatrix} \gamma_0(x) \\ \vdots \\ \gamma_k(x) \end{bmatrix} \qquad \theta = \begin{bmatrix} \theta_0 \\ \vdots \\ \theta_k \end{bmatrix}}$$

Examples

- Components of $\gamma(x)$ can be arbitrary nonlinear functions of x that are linear in θ:
  - Line Fitting (affine in *x*):

$$\boxed{f(x;\theta) = \theta_1 x + \theta_0} \quad \boxed{\gamma(x)^T = [1 \quad x]}$$

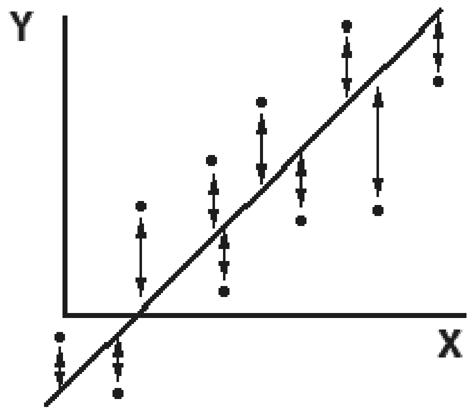
  - Polynomial Fitting (nonlinear in *x*):

$$\boxed{f(x;\theta) = \sum_{i=0}^{k} \theta_i x^i} \quad \boxed{\gamma(x)^T = \begin{bmatrix} 1 & \cdots & x^k \end{bmatrix}}$$

  - Truncated Fourier Series (nonlinear in *x*):

$$\boxed{f(x;\theta) = \sum_{i=0}^{k} \theta_i \cos(i\,x)} \quad \boxed{\gamma(x)^T = \begin{bmatrix} 1 & \cdots & \cos(k\,x) \end{bmatrix}}$$

(Unweighted)Least Squares



- Loss = Euclidean norm of model error:

$$\boxed{\mathrm{L}(\theta) = \left\| y - \Gamma(x)\theta \right\|^2}$$

where

$$y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \qquad \Gamma(x) = \begin{bmatrix} \gamma^T(x_1) \\ \vdots \\ \gamma^T(x_n) \end{bmatrix} \qquad \theta = \begin{bmatrix} \theta_0 \\ \vdots \\ \theta_k \end{bmatrix}$$

- The loss function can also be rewritten as,

$$\boxed{\mathrm{L}(\theta) = \sum_{i=1}^{n} \left(y_i - \gamma^T(x_i)\theta \right)^2 = n \left\langle \left(y - \gamma^T(x)\theta \right)^2 \right\rangle_n}$$

E.g., for the line, $\mathrm{L}(\theta) = \sum_i \left(y_i - \theta_0 - \theta_1 x_i \right)^2$

Examples

- The most important component is the *Design Matrix* $\Gamma(x)$
  - Line Fitting:

$$\boxed{f(x;\theta) = \theta_1 x + \theta_0}$$

$$\boxed{\Gamma(x) = \begin{bmatrix} 1 & x_1 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix}}$$

  - Polynomial Fitting:

$$\boxed{f(x;\theta) = \sum_{i=0}^{k} \theta_i x^i}$$

$$\boxed{\Gamma(x) = \begin{bmatrix} 1 & \cdots & x_1^k \\ \vdots & \ddots & \vdots \\ 1 & \cdots & x_n^k \end{bmatrix}}$$

  - Truncated Fourier Series:

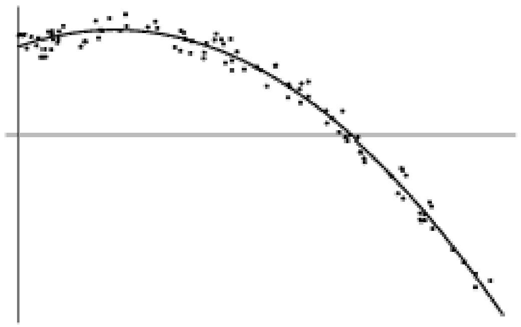
$$\boxed{f(x;\theta) = \sum_{i=0}^{k} \theta_i \cos(i\,x)}$$

$$\boxed{\Gamma(x) = \begin{bmatrix} 1 & \cdots & \cos(k\,x_1) \\ \vdots & \ddots & \vdots \\ 1 & \cdots & \cos(k\,x_n) \end{bmatrix}}$$

(Unweighted) Least Squares

- One way to minimize

$$\boxed{L(\theta) = \left\| y - \Gamma(x)\theta \right\|^2}$$



is to find a value $\hat{\theta}$ such that

$$\frac{\partial}{\partial\theta} L(\hat{\theta}) = -2\left[y - \Gamma(x)\hat{\theta} \right]^T \Gamma(x) = 0 \quad \text{and} \quad \frac{\partial^2}{\partial\theta^2} L(\hat{\theta}) = 2\Gamma(x)^T \Gamma(x) \succ 0$$

– These conditions will hold when $\Gamma(x)$ is one-to-one, yielding

$$\boxed{\hat{\theta} = \left[\Gamma^T(x)\Gamma(x) \right]^{-1} \Gamma^T(x) y = \Gamma^+(x) y}$$

Thus, the least squares solution is determined by the Pseudoinverse of the Design Matrix $\Gamma(x)$:

$$\boxed{\Gamma^+(x) = \left[\Gamma^T(x)\Gamma(x) \right]^{-1} \Gamma^T(x)}$$

(Unweighted) Least Squares

- If the design matrix $\Gamma(x)$ is one-to-one (has full column rank) the least squares solution is conceptually easy to compute. This will nearly always be the case in practice.
- Recall the example of fitting a line in the plane:

$$\Gamma^T(x)\Gamma(x) = \begin{bmatrix} 1 & \dots & 1 \\ x_1 & \dots & x_n \end{bmatrix} \begin{bmatrix} 1 & x_1 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix} = n \begin{bmatrix} 1 & \langle x \rangle \\ \langle x \rangle & \langle x^2 \rangle \end{bmatrix}$$

$$\Gamma^T(x)\,y = \begin{bmatrix} 1 & \dots & 1 \\ x_1 & \dots & x_n \end{bmatrix} \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} = n \begin{bmatrix} \langle y \rangle \\ \langle xy \rangle \end{bmatrix}$$

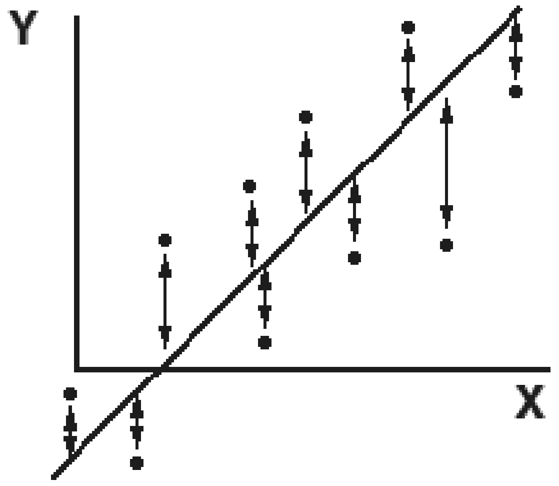
(Unweighted)Least squares

- **Pseudoinverse solution = LS solution:**

$$\boxed{\hat{\theta} = \Gamma^{+}(x)\,y = \left[\Gamma^{T}(x)\Gamma(x)\right]^{-1}\Gamma^{T}(x)\,y}$$

The LS line fit solution is

$$\boxed{\hat{\theta} = \begin{bmatrix} 1 & \langle x \rangle \\ \langle x \rangle & \langle x^2 \rangle \end{bmatrix}^{-1} \begin{bmatrix} \langle y \rangle \\ \langle xy \rangle \end{bmatrix}}$$



which (naively) requires

- a 2x2 matrix inversion
- followed by a 2x1 vector post-multiplication

(Unweighted) Least Squares

- What about fitting ***k*th order polynomial? :**

$$\boxed{f(x;\theta)=\sum_{i=0}^{k}\theta_i x^i} \quad \boxed{\Gamma(x)=\begin{bmatrix}1 & \cdots & x_1^k\\ \vdots & \ddots & \vdots\\ 1 & \cdots & x_n^k\end{bmatrix}}$$

$$\Gamma^T(x)\Gamma(x)=\begin{bmatrix}1 & \cdots & 1\\ \vdots & \ddots & \vdots\\ x_1^k & \cdots & x_n^k\end{bmatrix}\begin{bmatrix}1 & \cdots & x_1^k\\ \vdots & \ddots & \vdots\\ 1 & \cdots & x_n^k\end{bmatrix}$$

$$=n\begin{bmatrix}1 & \cdots & \langle x^k\rangle\\ \vdots & \ddots & \vdots\\ \langle x^k\rangle & \cdots & \langle x^{2k}\rangle\end{bmatrix}$$

(Unweighted) Least squares

and

$$\Gamma^T(x)y = \begin{bmatrix} 1 & \cdots & 1 \\ \vdots & \ddots & \vdots \\ x_1^k & \cdots & x_n^k \end{bmatrix} \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} = n \begin{bmatrix} \langle y \rangle \\ \vdots \\ \langle x^k y \rangle \end{bmatrix}$$

Thus, when $\Gamma(x)$ is one-to-one (has full column rank):

$$\boxed{\hat{\theta} = \Gamma^+(x)y = \left(\Gamma^T(x)\Gamma(x)\right)^{-1}\Gamma^T(x)y = \begin{bmatrix} 1 & \cdots & \langle x^K \rangle \\ \vdots & \ddots & \vdots \\ \langle x^K \rangle & \cdots & \langle x^{2K} \rangle \end{bmatrix}^{-1} \begin{bmatrix} \langle y \rangle \\ \vdots \\ \langle x^k y \rangle \end{bmatrix}}$$

- Mathematically, this is a very straightforward procedure.
- Numerically, this is generally NOT how the solution is computed. (Viz. the sophisticated algorithms in Matlab)

(Unweighted) Least Squares

- Note the computational costs
  - when (naively) fitting a line (1st order polynomial):

$$\boxed{\hat{\theta} = \begin{bmatrix} 1 & \langle x \rangle \\ \langle x \rangle & \langle x^2 \rangle \end{bmatrix}^{-1} \begin{bmatrix} \langle y \rangle \\ \langle xy \rangle \end{bmatrix}}$$

2x2 matrix inversion, followed by a 2x1 vector post-multiplication

  - when (naively) fitting a general *k*th order polynomial:

$$\boxed{\hat{\theta} = \begin{bmatrix} 1 & \cdots & \langle x^k \rangle \\ \vdots & \ddots & \vdots \\ \langle x^k \rangle & \cdots & \langle x^{2k} \rangle \end{bmatrix}^{-1} \begin{bmatrix} \langle y \rangle \\ \vdots \\ \langle x^k y \rangle \end{bmatrix}}$$

(k+1)x(k+1) matrix inversion, followed by a (k+1)x1 multiplication

    - It is evident that the computational complexity is an increasing function of the number of parameters.
    - More efficient (and numerically stable) algorithms are used in practice, but complexity scaling with number of parameters still remains true.

(Unweighted) Least Squares

- Suppose the dataset $\{x_i\}$ is constant in value over repeated, non-constant measurements of the dataset $\{y_i\}$.
  - e.g. consider some process (e.g., temperature) where the measurements $\{y_i\}$ are taken at the same locations $\{x_i\}$ every day
- Then the design matrix Γ(x) is constant in value!
  - In advance (*off-line*) compute:
  - Everyday (*on-line*) re-compute:

$$\boxed{A = \begin{bmatrix} 1 & \cdots & \langle x^k \rangle \\ \vdots & \ddots & \vdots \\ \langle x^k \rangle & \cdots & \langle x^{2k} \rangle \end{bmatrix}^{-1}}$$

$$\boxed{\hat{\theta} = A \begin{bmatrix} \langle y \rangle \\ \vdots \\ \langle x^k y \rangle \end{bmatrix}}$$

- Hence, least squares sometimes can be implemented very efficiently.
- There are also efficient recursive updates, e.g. the Kalman filter.

Geometric Solution & Interpretation

- Alternatively, we can derive the least squares solution using geometric (Hilbert Space) considerations.
- Goal: minimize the size (norm) of the model prediction error (aka residual error), $e(\theta) = y - \Gamma(x)\theta$:

$$\boxed{\mathbf{L}(\theta) = \|e(\theta)\|^2 = \|y - \Gamma(x)\theta\|^2}$$

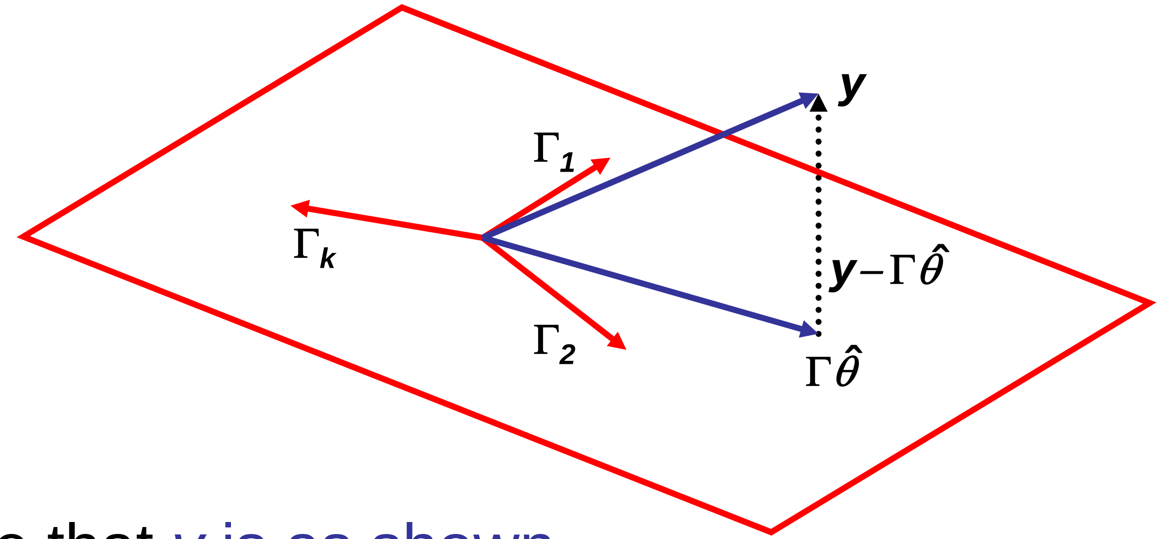
- Note that given a known design matrix $\Gamma(x)$ the vector

$$\boxed{\Gamma(x)\theta = \begin{bmatrix} | & & | \\ \Gamma_1 & \dots & \Gamma_k \\ | & & | \end{bmatrix} = \sum_{i=1}^{k} \begin{bmatrix} | \\ \Gamma_i \\ | \end{bmatrix} \theta_i}$$

  is a linear combination of the column vectors Γ_i .
- I.e. $\Gamma(x)\theta$ is in the range space (column space) of $\Gamma(x)$

(Hilbert Space) Geometric Interpretation

- The vector $\Gamma\theta$ lives in the range (column space) of $\Gamma = \Gamma(x) = [\Gamma_1, \dots, \Gamma_k]$:

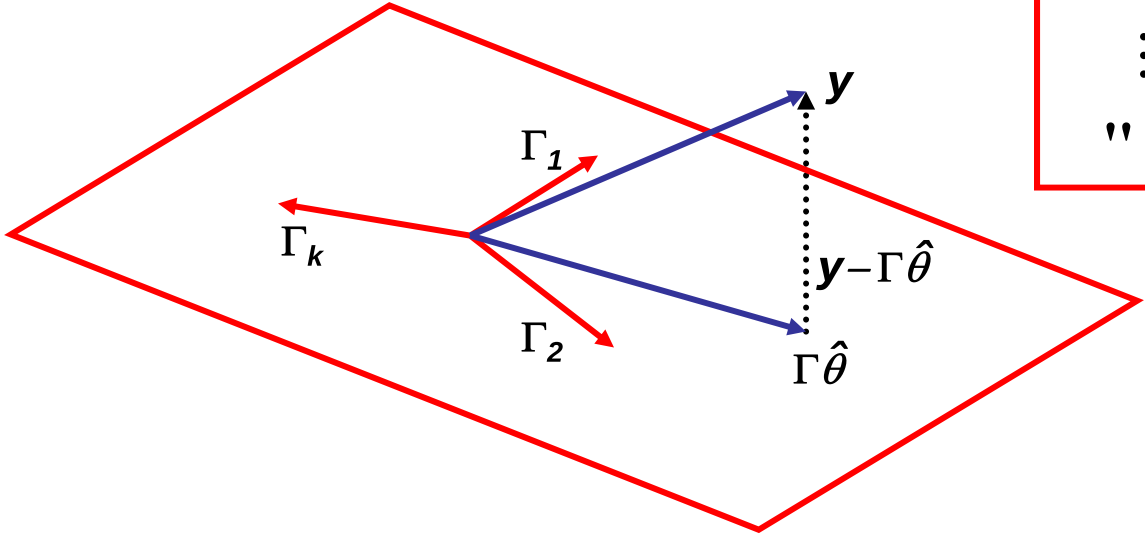


- Assume that *y* is as shown.
- Equivalent statements are:
  - $\Gamma\hat{\theta}$ is the value of $\Gamma\theta$ closest to *y* in the range of Γ.
  - $\Gamma\hat{\theta}$ is the orthogonal projection of *y* onto the range of Γ.
  - The residual e = y- $\Gamma\hat{\theta}$ is $\perp$ to the range of Γ.

Geometric Interpretation of LS

- e = y - $\Gamma\hat{\theta}$ is $\perp$ to the range of Γ iff

$$\begin{array}{cc} \left(y-\Gamma\hat{\theta}\right) & \perp\Gamma_1 \\ \text{"} \quad \text{"} & \perp\Gamma_2 \\ \vdots & \vdots \\ \text{"} \quad \text{"} & \perp\Gamma_k \end{array}$$



- Thus e = y - $\Gamma\hat{\theta}$ is in the nullspace of Γ^T:

$$\begin{cases} \Gamma_1^T\left(y-\Gamma\hat{\theta}\right)=0 \\ \vdots \\ \Gamma_1^T\left(y-\Gamma\hat{\theta}\right)=0 \end{cases} \Leftrightarrow \begin{bmatrix} - & \Gamma_1^T & - \\ & \vdots & \\ - & \Gamma_1^T & - \end{bmatrix}\left(y-\Gamma\hat{\theta}\right)=0 \Leftrightarrow \Gamma^T\left(y-\Gamma\theta\right)=0$$

Geometric interpretation of LS

- Note: Nullspace Condition $\equiv$ Normal Equation:

$$\boxed{\Gamma^T\left(y-\Gamma\hat{\theta}\right)=0 \quad \Leftrightarrow \quad \Gamma^T\Gamma\hat{\theta}=\Gamma^T y}$$

- Thus, if Γ is one-to-one (has full column rank) we again get the pseudoinverse solution:

$$\boxed{\hat{\theta}=\Gamma^{+}(x)y=\left[\Gamma^T(x)\ \Gamma(x)\right]^{-1}\Gamma^T(x)y}$$

Probabilistic Interpretation

- We have seen that estimating a parameter by minimizing the least squares loss function is a special case of MLE
- This interpretation holds for many loss functions
  - First, note that

$$\boxed{\begin{aligned}\hat{\theta} &= \arg\min_{\theta \in \Theta} \mathrm{L}\left[y, f(x;\theta)\right] \\ &= \arg\max_{\theta \in \Theta} e^{-\mathrm{L}\left[y, f(x;\theta)\right]}\end{aligned}}$$

  - Now note that, because

$$\boxed{e^{-L\left[y, f(x;\theta)\right]} \geq 0, \quad \forall y, \forall x}$$

we can make this exponential function into a ***Y|X*** pdf by an appropriate normalization.

Prob. Interpretation of Loss Minimization

- I.e. by defining

$$\boxed{P_{Y|X}(y\,|\,x;\theta) \square \left(\frac{1}{\int e^{-L[y,f(x;\theta)]}dy} \right) e^{-L[y,f(x;\theta)]} \square \alpha(x;\theta) e^{-L[y,f(x;\theta)]}}$$

- If the normalization constant $\alpha(x;\theta)$ does not depend on θ, $\alpha(x;\theta) = \alpha(x)$, then

$$\boxed{\begin{aligned} \hat{\theta} &= \arg\max_{\theta \in \Theta} e^{-L[y,f(x;\theta)]} \\ &= \arg\max_{\theta \in \Theta} P_{Y|X}\left(y\,|\,x;\theta\right) \end{aligned}}$$

which makes the problem a special case of MLE

Prob. Interpretation of Loss Minimization

- Note that for loss functions of the form,

$$\boxed{L(\theta) = g[y - f(x;\theta)]}$$

the model ***f*(*x*;*θ*)** only changes the mean of ***Y***

- – A shift in mean does not change the shape of a pdf, and therefore cannot change the value of the normalization constant
- – Hence, for loss functions of the type above, it is always true that

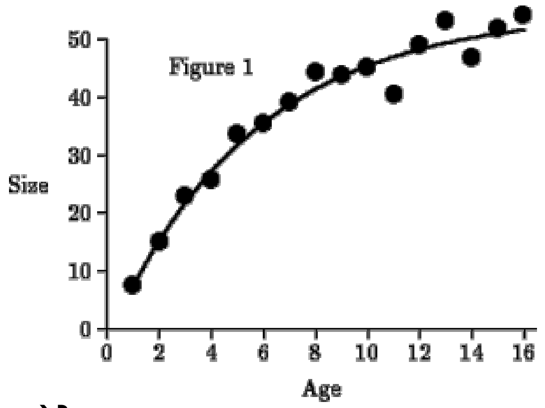
$$\boxed{\begin{aligned}\hat{\theta} &= \arg\max_{\theta \in \Theta} e^{-L[y, f(x;\theta)]} \\ &= \arg\max_{\theta \in \Theta} P_{Y|X}\left(y \mid x;\theta\right)\end{aligned}}$$

Regression

- This leads to the interpretation that we saw last time
- Which is the usual definition of a regression problem
  - two random variables X and Y
  - a dataset of examples $\mathcal{D} = \{(x_1,y_1), \ldots ,(x_n,y_n)\}$
  - a parametric model of the form

$$\boxed{y = f(x;\theta) + \varepsilon}$$

  - where θ is a parameter vector, and ε a random variable that accounts for noise
- the pdf of the noise determines the loss in the other formulation



Regression

- Error pdf:
  - Gaussian

$$\boxed{P_\varepsilon(\varepsilon) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{\varepsilon^2}{2\sigma^2}}}$$

  - Laplacian

$$\boxed{P_\varepsilon(\varepsilon) = \frac{1}{2\sigma} e^{-\frac{|\varepsilon|}{\sigma}}}$$

  - Rayleigh

$$\boxed{P_\varepsilon(\varepsilon) = \frac{\varepsilon}{\sigma^2} e^{-\frac{\varepsilon^2}{2\sigma^2}}}$$

- Equivalent Loss Function:
  - L_2 distance

$$\boxed{L(x, y) = (y - x)^2}$$

  - L_1 distance

$$\boxed{L(x, y) = |y - x|}$$

  - Rayleigh distance

$$\boxed{\begin{aligned} L(x, y) = (y - x)^2 \\ - \log(y - x) \end{aligned}}$$

Regression

- We know how to solve the problem with losses
  - Why would we want the added complexity incurred by introducing error probability models?
- The main reason is that this allows a data driven definition of the loss
- One good way to see this is the problem of weighted least squares
  - Suppose that you know that not all measurements (x_i, y_i) have the same importance
  - This can be encoded in the loss function
  - Remember that the unweighted loss is

$$\boxed{L = \| y - \Gamma(x)\theta \|^2 = \sum_i \left(y = [\Gamma(x)\theta]_i \right)^2}$$

Regression

- To weigh different points differently we could use

$$\boxed{L = \sum_i w_i \left(y_i - [\Gamma(x)\theta]_i \right)^2, w_i > 0,}$$

or even the more generic form

$$\boxed{L = \left(y - \Gamma(x)\theta \right)^T W \left(y - \Gamma(x)\theta \right), \; W = W^T > 0,}$$

- In the latter case the solution is (homework)

$$\boxed{\theta^* = \left[\Gamma(x)^T W \; \Gamma(x) \right]^{-1} \Gamma(x)^T W \; y}$$

- The question is “how do I know these weights”?
- Without a probabilistic model one has little guidance on this.

Regression

- The probabilistic equivalent

$$\boxed{\begin{aligned}\theta^* &= \arg\max_\theta e^{-L[y, f(x;\theta)]} \\ &= \arg\max_\theta \exp\left\{-\frac{1}{2}(y-\Gamma(x)\theta)^T W (y-\Gamma(x)\theta)\right\}\end{aligned}}$$

is the **MLE** for a Gaussian pdf of known covariance

$$\boxed{\Sigma = W^{-1}}$$

- In the case where the covariance W is diagonal we have

$$\boxed{L = \sum_i \frac{1}{\sigma_i^2}\left(y_i - [\Gamma(x)\theta]_i\right)^2}$$

- In this case, each point is weighted by the inverse variance.

Regression

- This makes sense
  - Under the probabilistic formulation the variance σ_i is the variance of the error associated with the ith observation

$$\boxed{y_i = f(x_i;\theta) + \varepsilon_i}$$

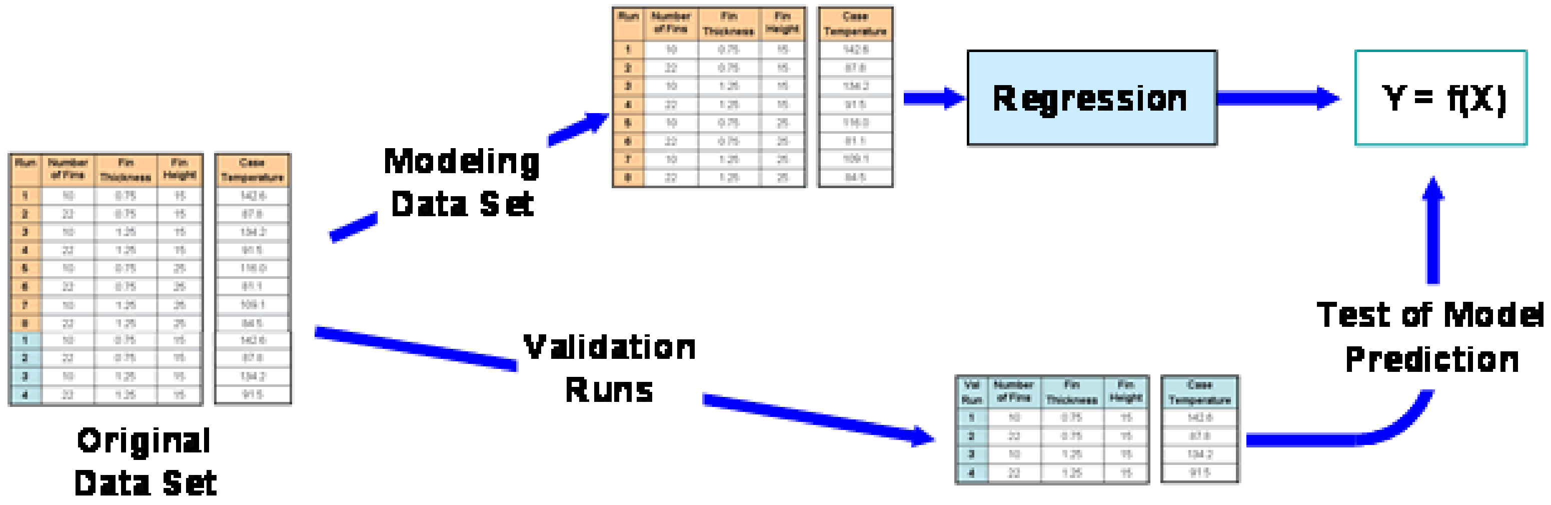
  - This means that it is a measure of the uncertainty of the observation
- When

$$\boxed{W = \Sigma^{-1}}$$

  we are weighting each point by the inverse of its uncertainty (variance)
- We can also check the goodness of this weighting matrix by plotting the histogram of the errors
- if ***W*** is chosen correctly, the errors should be Gaussian

Model Validation

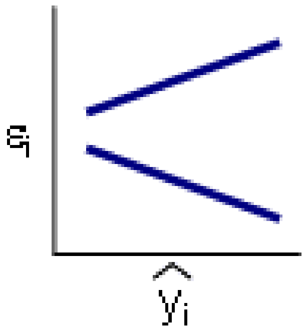
- In fact, by analyzing the errors of the fit we can say a lot
- This is called model validation



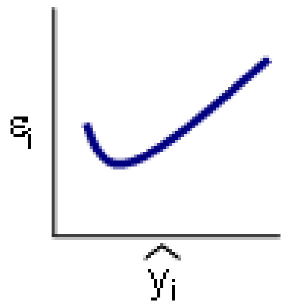
  - Leave some data on the side, and run it through the predictor
  - Analyze the errors to see if there is deviation from the assumed model (Gaussianity for least squares)

Model Validation & Improvement

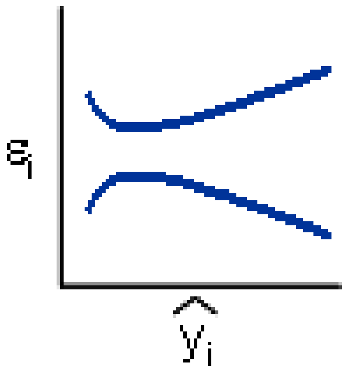
- Many times this will give you hints to alternative models that may fit the data better
- Typical problems for least squares (and solutions)



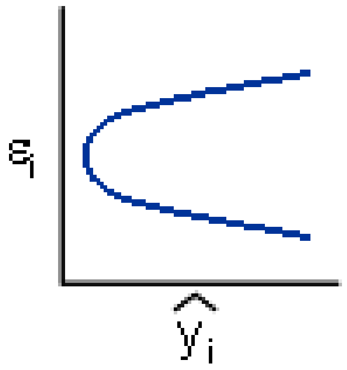
Increasing residuals.
Try ln(Y)=f(X)



A curving smile or frown.
Try ln(Y)=f(X)



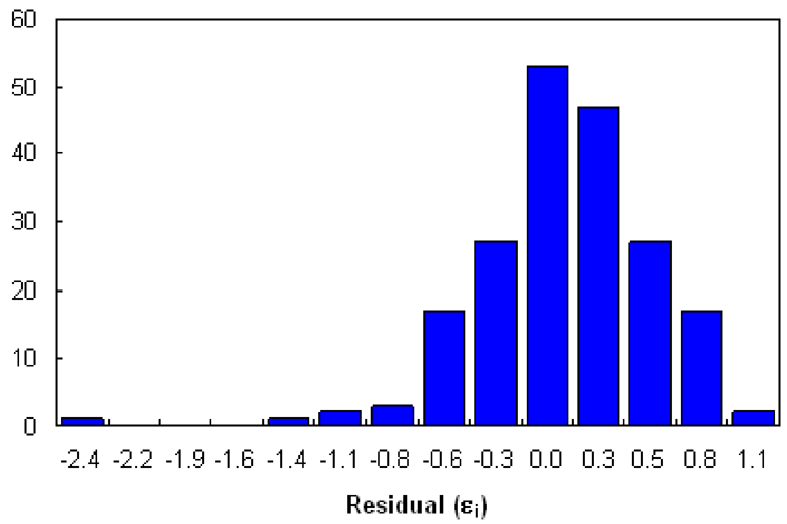
A venturi tube.
Try 1/Y=f(X)

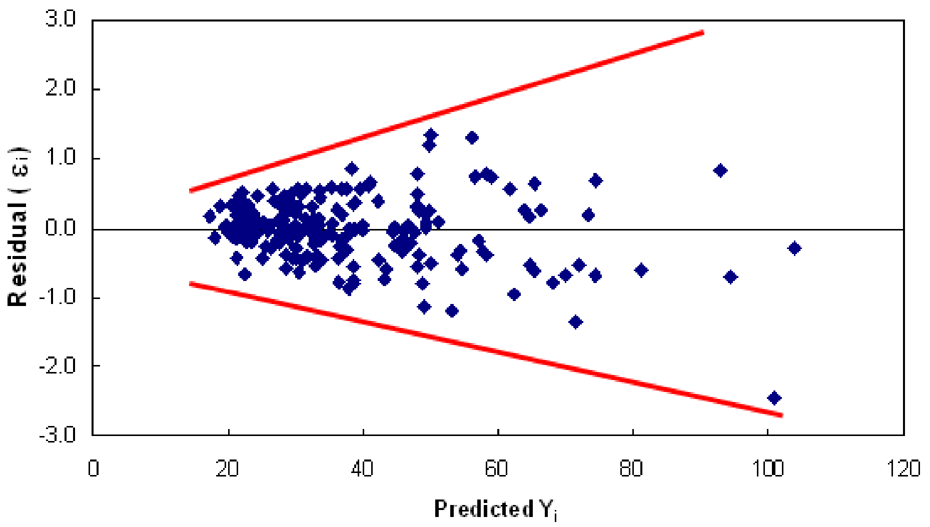


A bullet shape.
Try Sqrt(Y)=f(X)

Model Validation & Improvement

- Example 1 error histogram



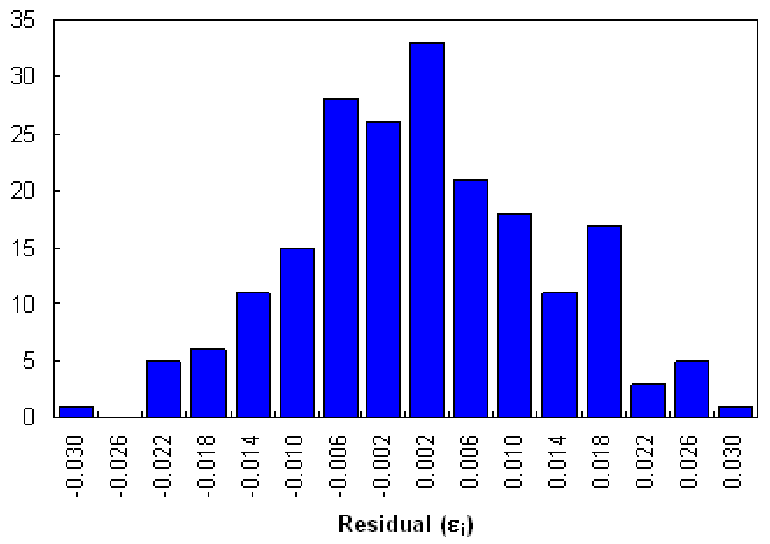


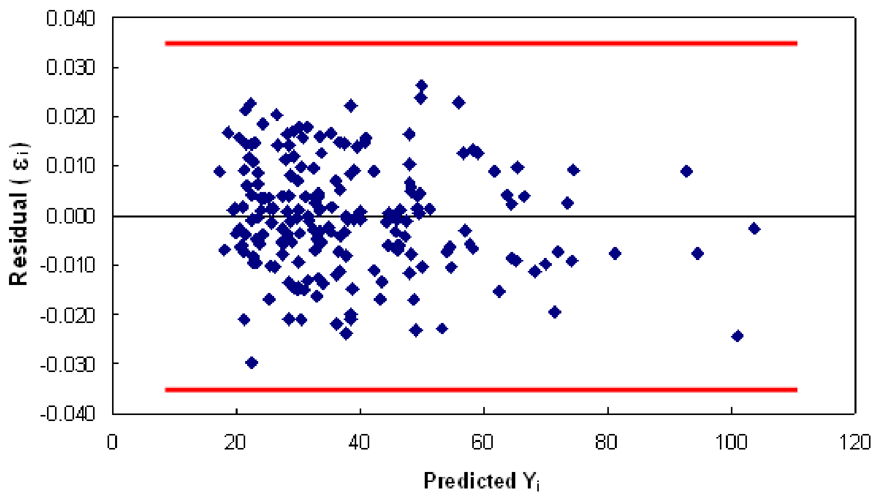
- this does not look Gaussian
- look at the scatter plot of the error (y – f(x,θ*))
  - increasing trend, maybe we should try

$$\boxed{\log(y_i) = f(x_i;\theta) + \varepsilon_i}$$

Model Validation & Improvement

- Example 1 error histogram for the new model

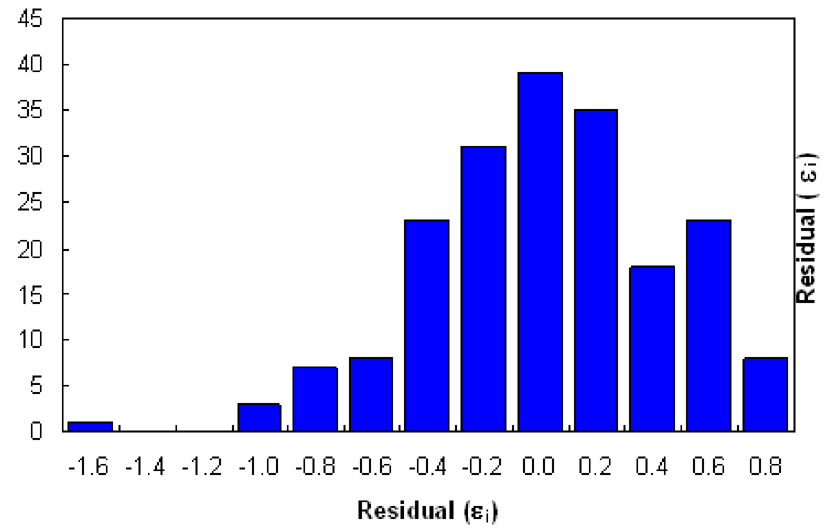


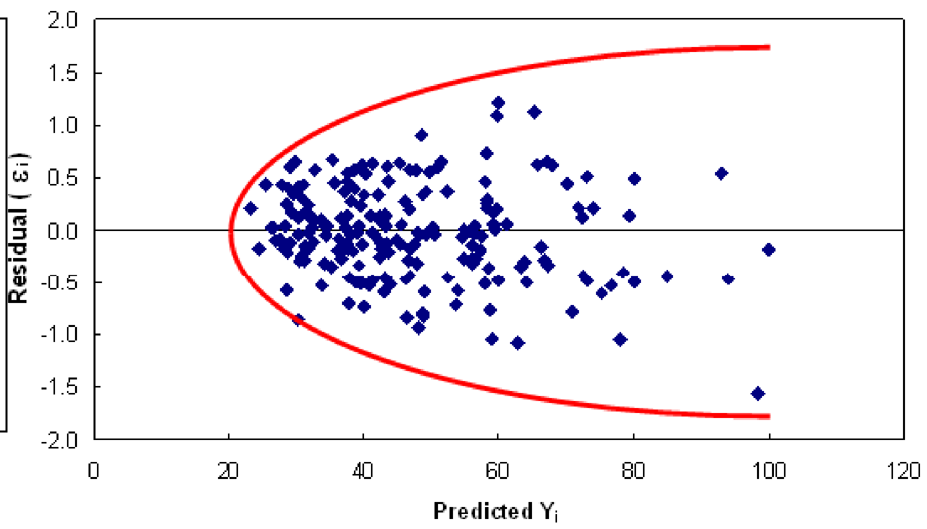


- this looks Gaussian
  - this model is probably better
  - there are statistical tests that you can use to check this objectively
  - these are covered in statistics classes

Model Validation & Improvement

- Example 2 error histogram



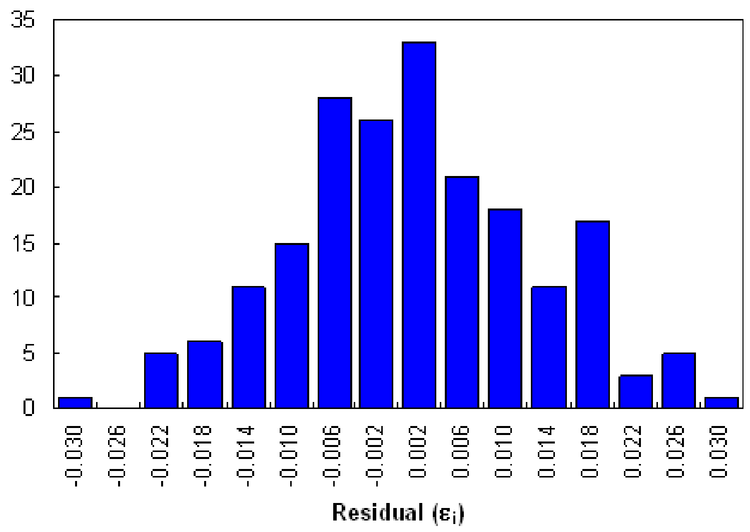


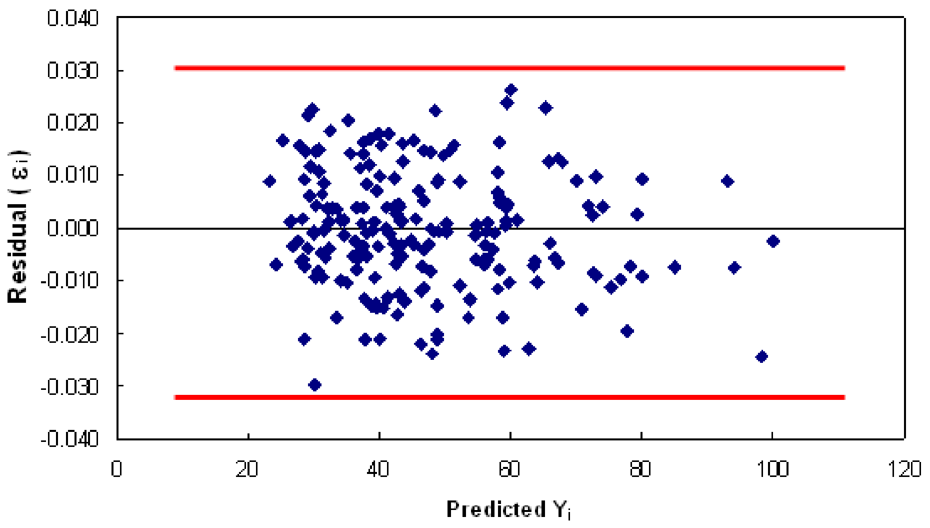
- This also does not look Gaussian
- Checking the scatter plot now seems to suggest to **try**

$$\boxed{\sqrt{y_i} = f(x_i;\theta) + \varepsilon_i}$$

Model Validation & Improvement

- Example 2 error histogram for the new model





- Once again, seems to work
- The residual behavior looks Gaussian
  - However, It is NOT always the case that such changes will work.
    - If not, maybe the problem is the assumption of Gaussianity itself
    - Move away from least squares, try MLE with other error pdfs

END