# Model fitting

By Anqi Wu

neuromatch academy

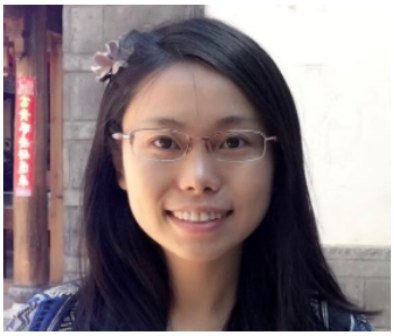
# Who is Anqi Wu

- Postdoc at Columbia with Liam Paninski and John Cunningham

- PhD at Princeton with Jonathan Pillow

- Research interest: neural sensory encoding, latent variable models for large-scale neural recordings, fMRI decoding, behavior analysis

- Modeling: probabilistic graphical model, Gaussian process, latent variable/dynamic model, Bayesian deep learning, Bayesian optimization and active learning



# Who is Pierre-Etienne

- PhD candidate in Eero Simoncelli's lab at NYU

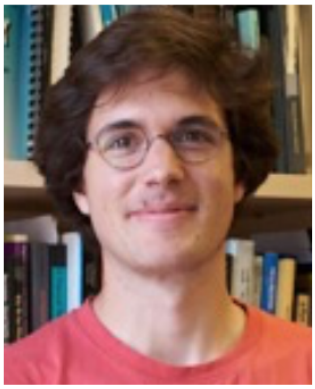
- MSc with Sophie Denève and Christian Machens at ENS Paris

- Research interests:

    Principles of biological information processing (gasp!)

    Visual motion perception and temporal prediction

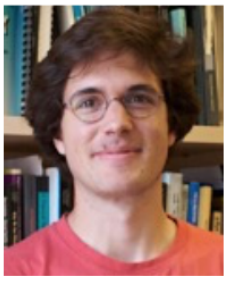
- Tools: signal processing, statistics and machine learning



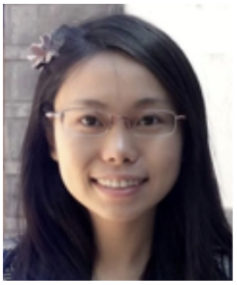
# D3 Team

Alex Hyafil

Jan Drugowitsch



Pierre Etienne Fiquet



Anqi Wu

Kunlin Wei

# Roadmap of Week 1 Day 3

Tutorial part 1

- Linear regression
  - Mean squared error
  - Maximum likelihood estimation
- Bootstrapping
- Multiple linear regression
  - Polynomial regression

Tutorial part 2

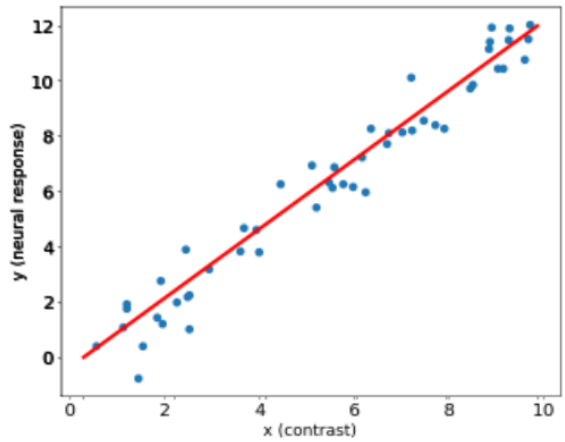
- Bias-variance tradeoff
  - Underfitting and overfitting
- Cross validation
  - K-fold
  - Leave-one-out

# W1D3 Tutorial 1: Linear Regression & Mean Squared Error

**Linear regression** makes predictions about the linear relationship between the input variable $x$ (contrast) and the output variable $y$ (neural response).

$$y = \theta_1 \times x + \theta_0$$

neural response — linear weight — contrast — Intercept




We are not considering the intercept for simplicity, resulting in a one-parameter model.

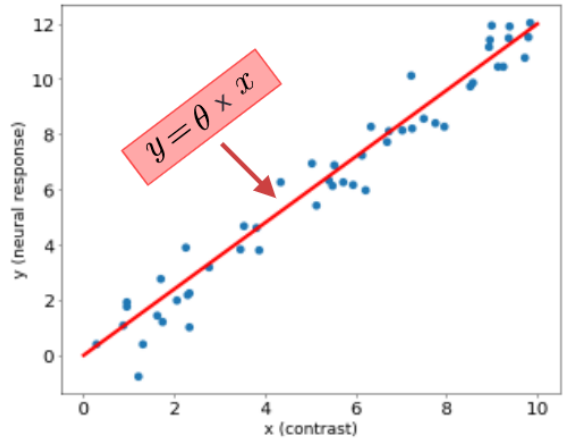
**Linear regression** makes predictions about the linear relationship between the input variable $x$ (contrast) and the output variable $y$ (neural response).

$$y = \theta \times x$$

neural response — linear weight — contrast

?

We are not considering the intercept for simplicity, resulting in a one-parameter model.



# Mean Squared Error (MSE)

MSE computes the average error between the model prediction $\hat{y}$ and the true $y$.

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 = \frac{1}{N}\sum_{i=1}^{N} e_i^2$$

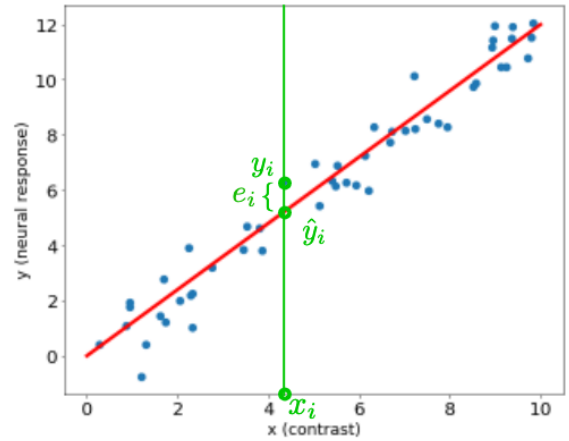
model prediction

residual

total number of data points

true neural response

index of data points i=1,...,N



# Mean Squared Error (MSE)

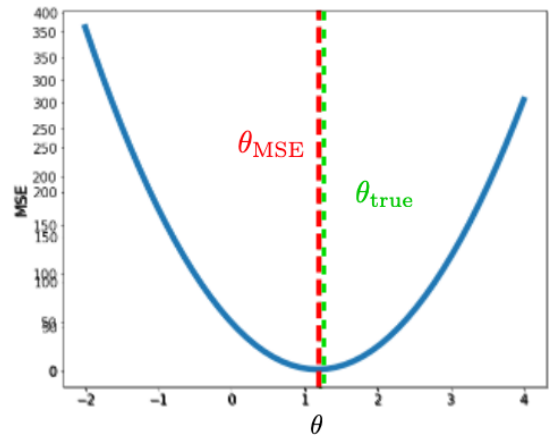
$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 = \frac{1}{N}\sum_{i=1}^{N}(y_i - \theta x_i)^2$$

Optimal $\theta^* = \underset{\theta}{\text{argmin}}\ \text{MSE} = \underset{\theta}{\text{argmin}}\ \frac{1}{N}\sum_{i=1}^{N}(y_i - \theta x_i)^2$

To minimize MSE, we solve for where its gradient is 0:

$$\frac{\partial \text{MSE}}{\partial \theta} = -\frac{2}{N}\sum_{i=1}^{N}(y_i - \theta x_i)x_i = 0 \Rightarrow \theta\sum_{i=1}^{N}x_i^2 - \sum_{i=1}^{N}x_i y_i = 0$$

$$\Rightarrow \theta_{\text{MSE}} = \frac{\sum_{i=1}^{N} x_i y_i}{\sum_{i=1}^{N} x_i^2}$$



# Notebook

In the following notebook, you will

1. Define a linear model and generate some simulated data from this underlying linear model.
2. Fit linear regression to the simulated data by minimizing MSE, and recover the linear weight.

Enjoy!!

# W1D3 Tutorial 2: Maximum Likelihood Estimation

# Maximum Likelihood Estimation (MLE)

**Gaussian/Normal distribution**

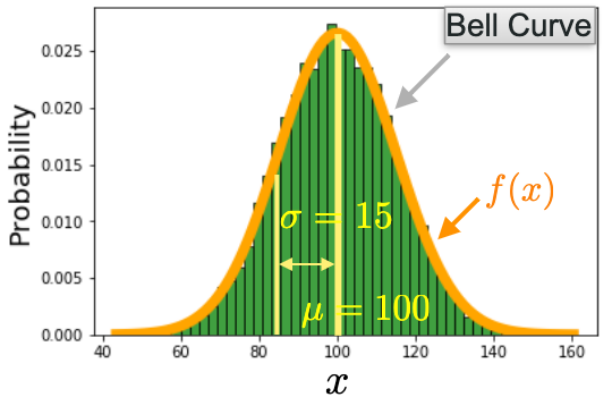
Data can be "distributed" (spread out) in different ways. But there are many cases where the data tends to be around a central value with no bias left or right, and it gets close to a "Gaussian Distribution" like this:

A Gaussian distribution function is defined by

mean parameter $\mu$ standard deviation $\sigma > 0$

variance $\sigma^2$

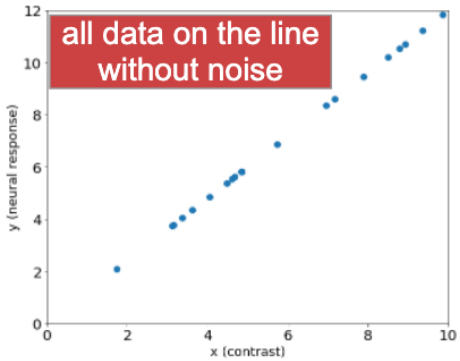
probability density function $f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2\sigma^2}(x-\mu)^2}$



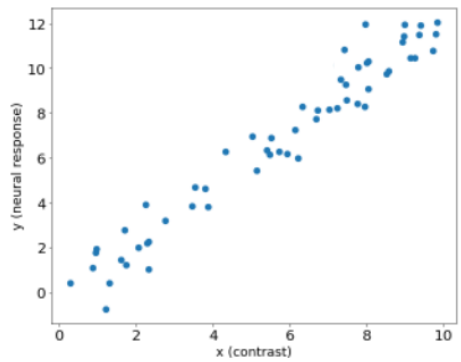
# Maximum Likelihood Estimation (MLE)

If we generate y from x using

$$y = \theta \times x$$




However this is the data we observe, with noise.



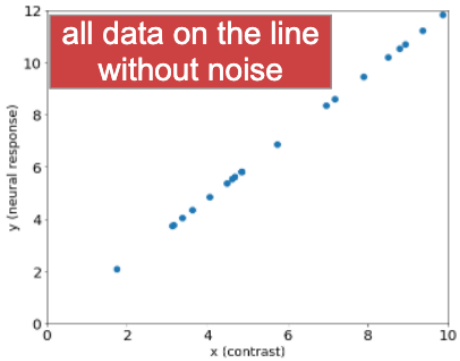

what we don't care about
(e.g., deviation from mean firing rate)

what we can't control
(e.g., measurement noise)

# Maximum Likelihood Estimation (MLE)

~~If we generate y from x using~~

$$y = \theta \times x$$



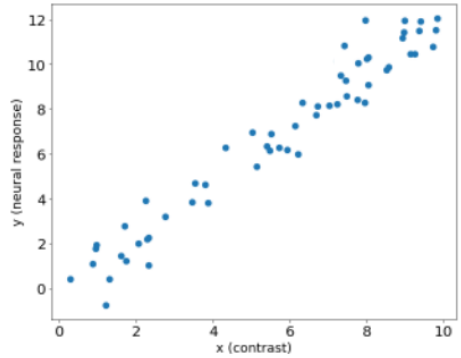

Model data with noise.

$$y = \theta \times x + \eta$$

Noise

what we don't care about
(e.g., deviation from mean firing rate)

what we can't control
(e.g., measurement noise)




$$\eta \sim \mathcal{N}(0, \sigma^2)$$

$$\downarrow$$

$$y \sim \mathcal{N}(\theta x, \sigma^2)$$

# Maximum Likelihood Estimation (MLE)

We consider the linear regression model with a Gaussian distributed noise $\eta$.

$$y = \theta \times x + \eta$$

neural response — linear weight — contrast — Gaussian noise $\eta \sim \mathcal{N}(0, \sigma^2)$

the larger $\sigma^2$ is, the noisier the data is

encoding model

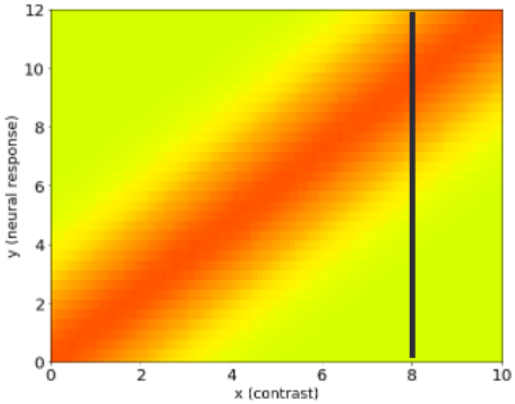
$$p(y|x,\theta) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2\sigma^2}(y-\theta x)^2}$$

$$\text{mean}(y) = \theta x$$

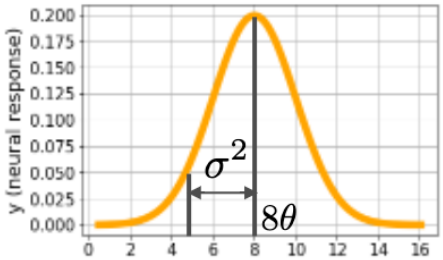
$$\text{var}(y) = \sigma^2$$

# Maximum Likelihood Estimation (MLE)

$p(y|x,\theta)$




$p(y|x=8,\theta)$




encoding model

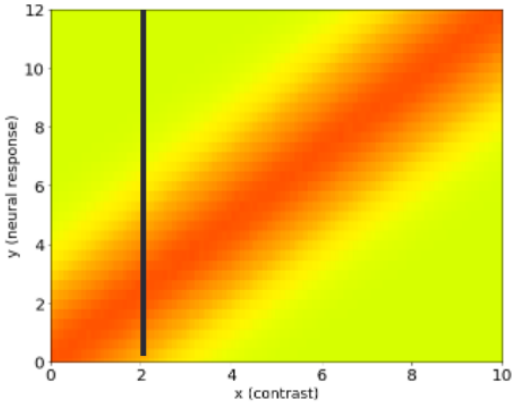
$$p(y|x=8,\theta) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2\sigma^2}(y-8\theta)^2}$$

$$\text{mean}(y) = 8\theta$$

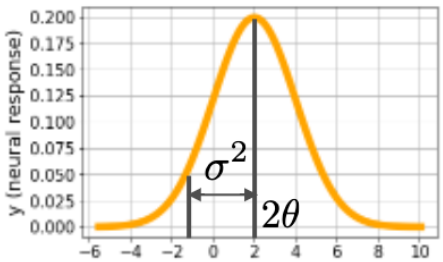
$$\text{var}(y) = \sigma^2$$

# Maximum Likelihood Estimation (MLE)

$p(y|x,\theta)$




$p(y|x=2,\theta)$




encoding model

$$p(y|x=2,\theta) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2\sigma^2}(y-2\theta)^2}$$

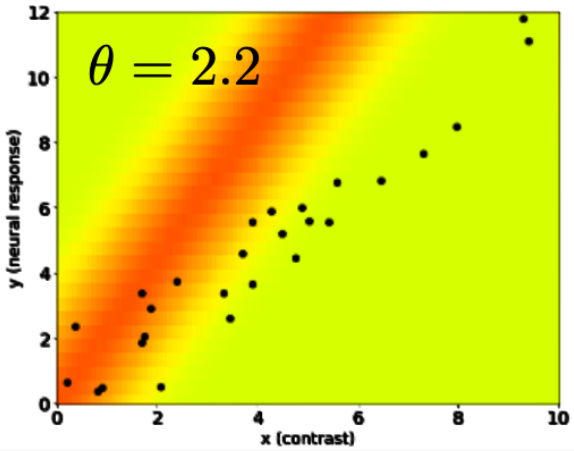
$$\text{mean}(y) = 2\theta$$

$$\text{var}(y) = \sigma^2$$

# Maximum Likelihood Estimation (MLE)

Say it here

observed data {x, y}




encoding model

$$p(y|x,\theta) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2\sigma^2}(y-\theta x)^2}$$

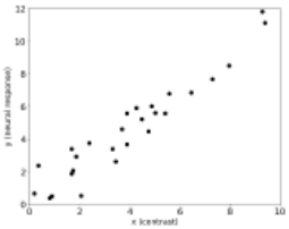
$$\text{mean}(y) = \theta x$$

$$\text{var}(y) = \sigma^2$$

$\theta$ is unknown

# Maximum Likelihood Estimation (MLE)

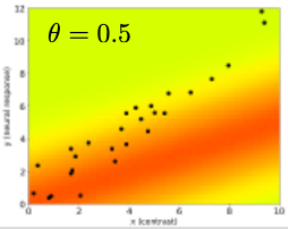
observed data {x, y}



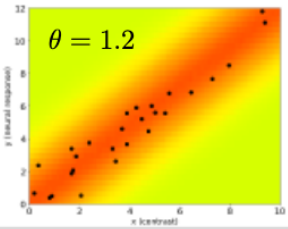
**Likelihood:** which $\theta$ leads to the Gaussian distribution that most likely matches the data points $\{x, y\}$?

$$\mathcal{L}(\theta|x,y) = p(y|x,\theta) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2\sigma^2}(y-\theta x)^2}$$

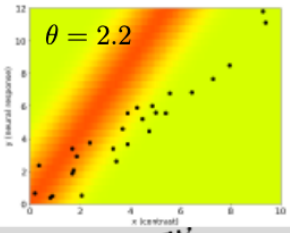
$p(y|x, \theta = 0.5)$




$p(y|x, \theta = 1.2)$




$p(y|x, \theta = 2.2)$




$$\mathcal{L}(\theta = 1.2) > \mathcal{L}(\theta = 0.5) > \mathcal{L}(\theta = 2.2)$$

Therefore, we could calculate the optimal $\theta$ by maximizing the likelihood.

# Maximum Likelihood Estimation (MLE)

Empirically, we maximize the log likelihood

$$
\begin{aligned}
\log \mathcal{L}(\theta|x,y) &= \log \prod_{i=1}^{N} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2\sigma^2}(y_i-\theta x_i)^2} \\
&= \sum_{i=1}^{N} \log \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2\sigma^2}(y_i-\theta x_i)^2} \\
&= \sum_{i=1}^{N} \log \frac{1}{\sigma\sqrt{2\pi}} + \sum_{i=1}^{N} \log\, e^{-\frac{1}{2\sigma^2}(y_i-\theta x_i)^2} \\
&= -\frac{N}{2} \log\, 2\pi\sigma^2 - \frac{1}{2\sigma^2} \sum_{i=1}^{N} (y_i - \theta x_i)^2
\end{aligned}
$$

# Maximum Likelihood Estimation (MLE)

Empirically, we maximize the log likelihood

Nx $\mathrm{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$

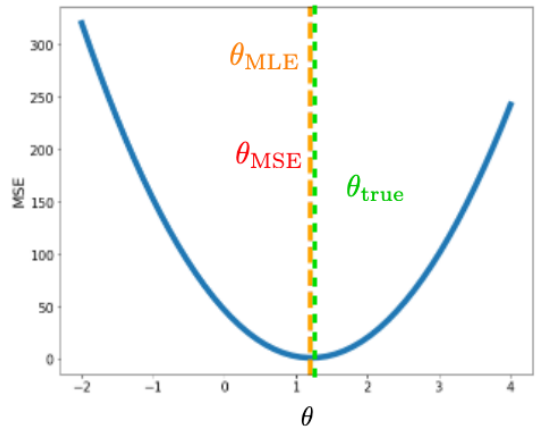
$$\log \mathcal{L}(\theta|x,y) = -\frac{N}{2}\log 2\pi\sigma^2 - \frac{1}{2\sigma^2}\sum_{i=1}^{N}(y_i - \theta x_i)^2$$

Differentiate and set to zero

$$\frac{\partial \log \mathcal{L}(\theta|x,y)}{\partial\theta} = \frac{1}{\sigma^2}\sum_{i=1}^{N}(y_i - \theta x_i)x_i = 0$$

$$\Rightarrow \theta_{\mathrm{MLE}} = \frac{\sum_{i=1}^{N} x_i y_i}{\sum_{i=1}^{N} x_i^2}$$

(the same as $\theta_{\mathrm{MSE}}$ when $p(y|x,\theta)$ is Gaussian)



# Notebook

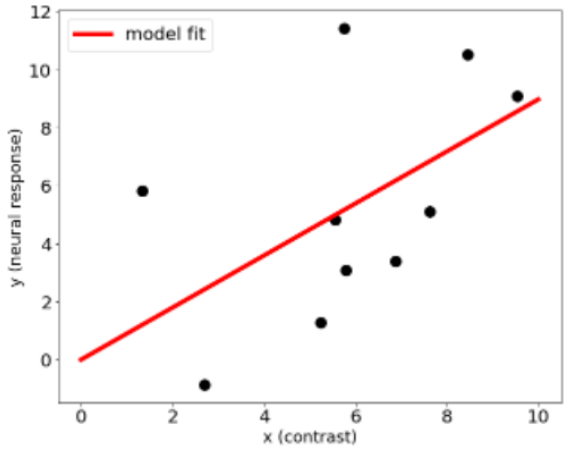
In the following notebook, you will

1. Code for likelihood function for Gaussian distribution.
2. Fit linear regression with maximum likelihood estimation.

Enjoy!!

# W1D3 Tutorial 3: Bootstrapping

# Uncertainty of Parameters



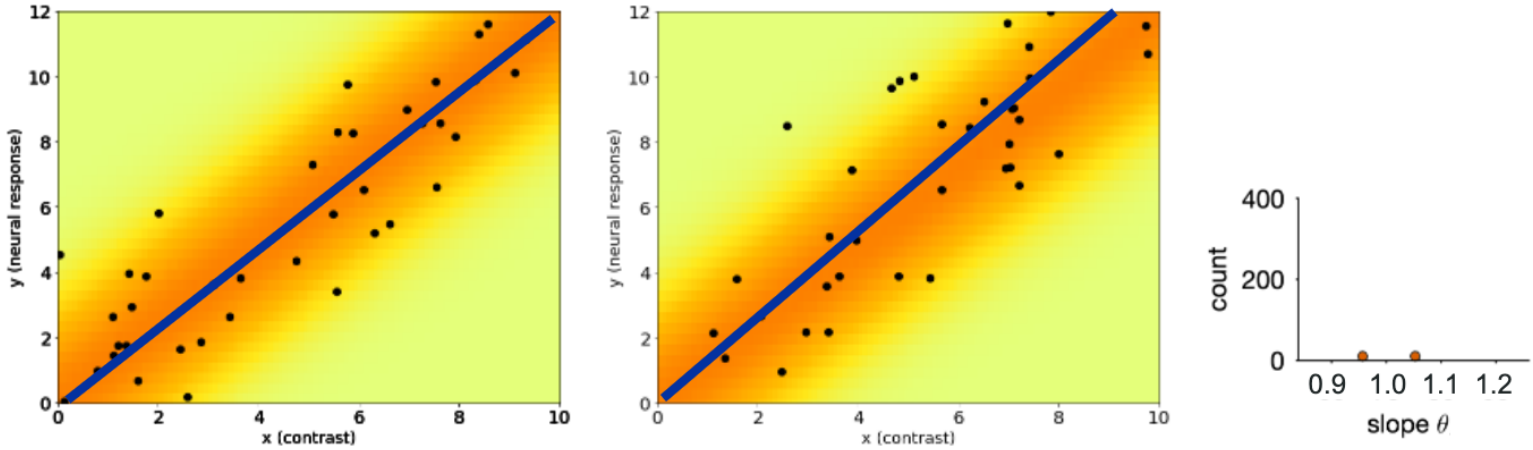

MSE = 0.38

Question: How confident we are about the model fit?

Use resampling to estimate uncertainty thus confidence

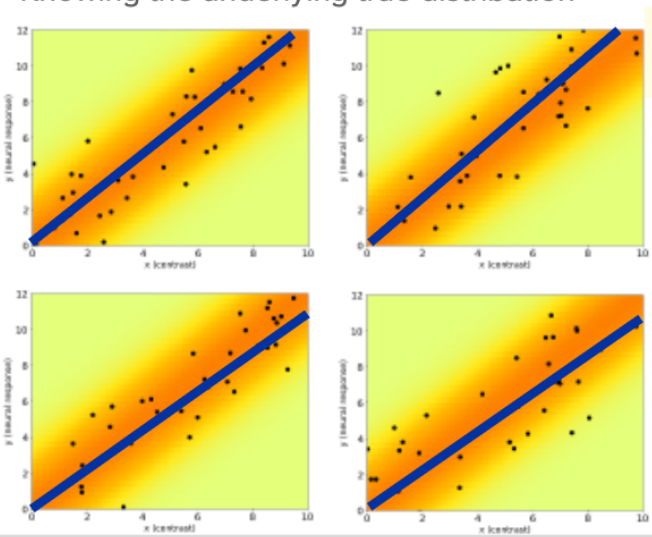
# Resampling to Estimate Uncertainty

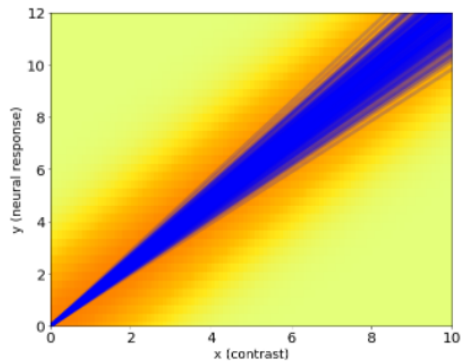
Knowing the true distribution



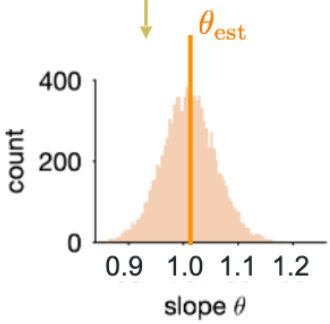
# Resampling to Estimate Uncertainty

Knowing the underlying true distribution



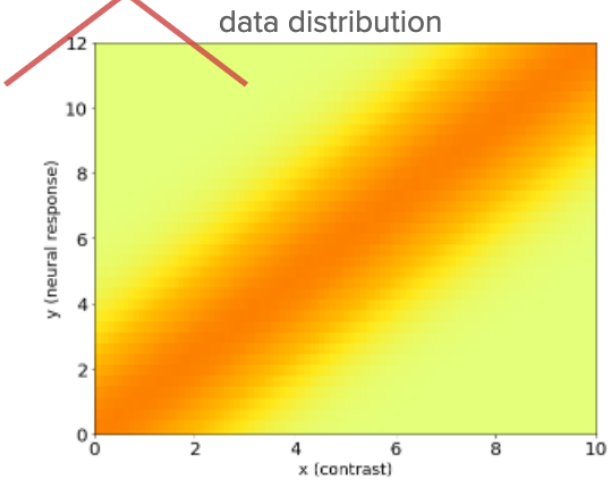


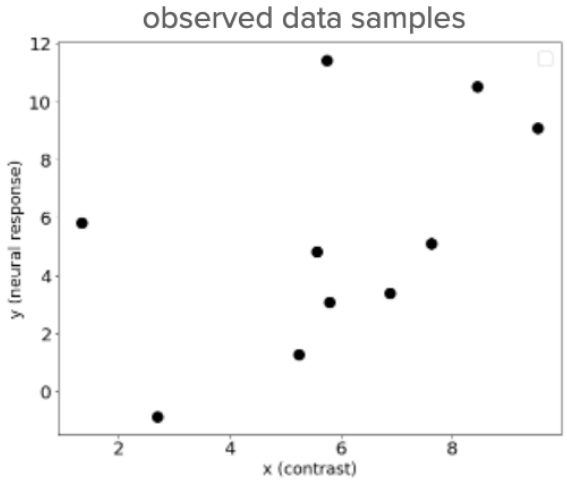

95% of bootstrap samples → confidence intervals



# Assessing Uncertainty by Bootstrap

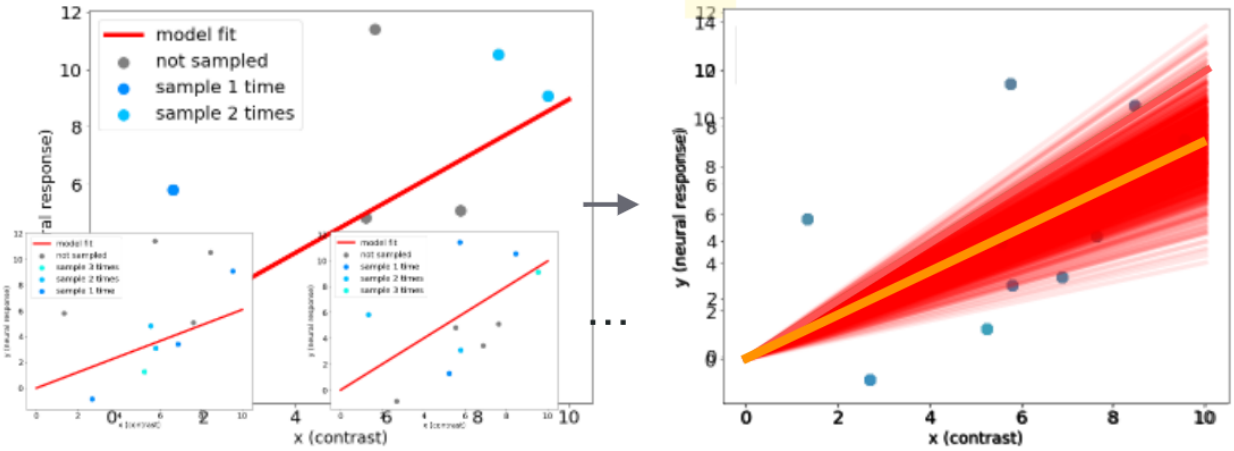
Of course we never know the true population; we only have data samples.



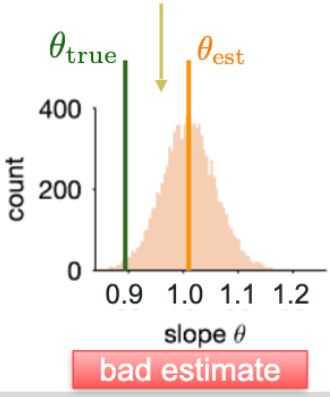



# Assessing Uncertainty by Bootstrap

1. Resampling from the observed dataset with replacement; the number of the resampled data points is the same as the number of observed data points.
2. Collect all estimates into a distribution, and analyze the confidence intervals.

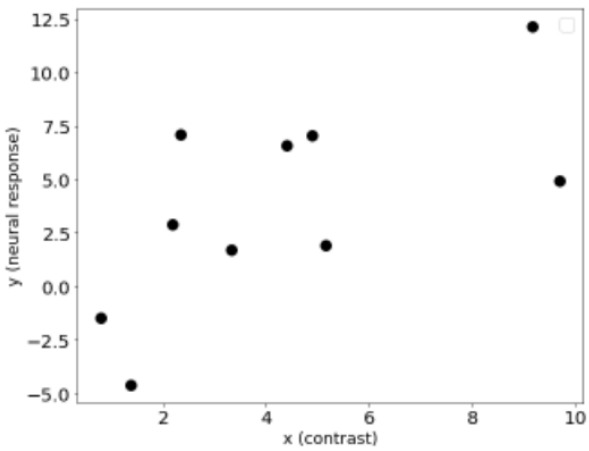


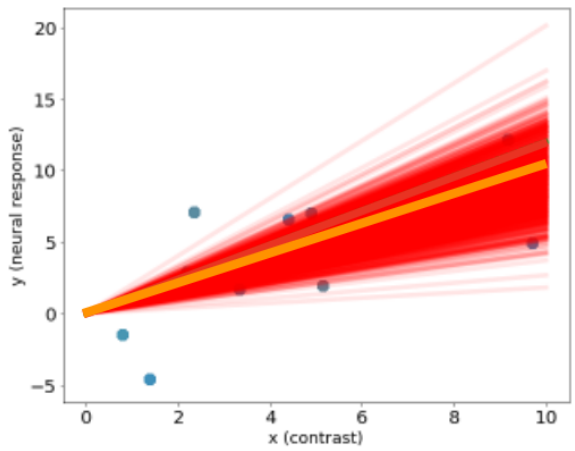

95% of bootstrap samples → confidence intervals



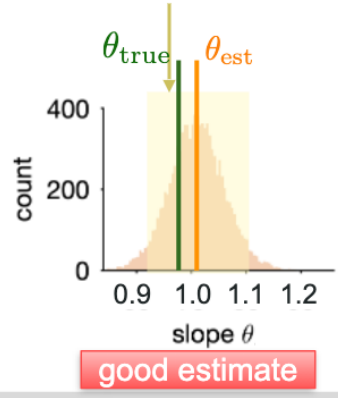
# Assessing Uncertainty by Bootstrap

1. Resampling from the observed dataset.
2. Collect all estimates into a distribution, and analyze the confidence intervals.

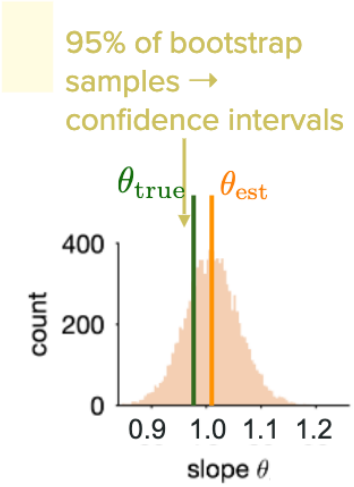



95% of bootstrap samples → confidence intervals



# Assessing Uncertainty by Bootstrap




- In most of real-world applications, we don't know the ground truth.
- But it's still beneficial to have a distribution rather than a point estimate.
- With distribution and uncertainty, you have more information to make decision.

# Notebook

In the following notebook, you will

1. Generate simulated dataset from an underlying linear model.
2. Implement bootstrap to get the confidence interval of the model fit.
3. Compare the true parameter with the estimated distribution to see how confident the model is.

Enjoy!!

# W1D3 Tutorial 4: Multiple Linear Regression

Linear model $y = \theta \times x$

neural response linear weight contrast

[Simple]
Linear model

$$y = \theta_1 \times x + \theta_0$$

neural response — linear weight — contrast — intercept

Multiple
linear model

$$y = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_d x_d$$

neural response

Assume multiple stimulus features
(e.g., orientation, contrast, etc.)

More generally,
in vector form,
for one data point i

$$y_i = \boldsymbol{\theta}^\top \boldsymbol{x}_i \qquad \forall i = 1, \ldots, N$$

$\boldsymbol{\theta} = [\theta_0, \theta_1, \theta_2, \ldots]^\top$ $\quad \boldsymbol{x}_i = [1, x_{i,1}, x_{i,2}, \ldots]^\top$

Build up to
matrix version

$$\boldsymbol{y} = X\boldsymbol{\theta}$$

$$\text{Index i} \downarrow \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \end{bmatrix} = \underbrace{\begin{bmatrix} \text{---} & \boldsymbol{x}_0 & \text{---} \\ \text{---} & \boldsymbol{x}_1 & \text{---} \\ \text{---} & \boldsymbol{x}_2 & \text{---} \\ & \vdots & \end{bmatrix}}_{\text{design matrix}} \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \vdots \end{bmatrix}$$

# Geometric Interpretation of MSE

MSE computes the average error between the model prediction $\hat{y}$ and the true $y$.

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 = \frac{1}{N}\sum_{i=1}^{N} e_i^2$$

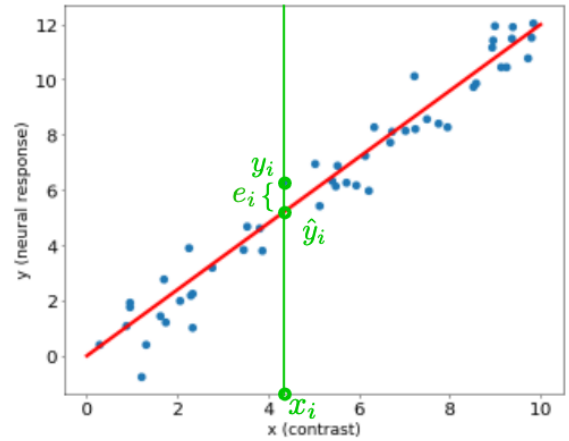
model prediction

residual

total number of data points

true neural response

index of data points i=1,...,N



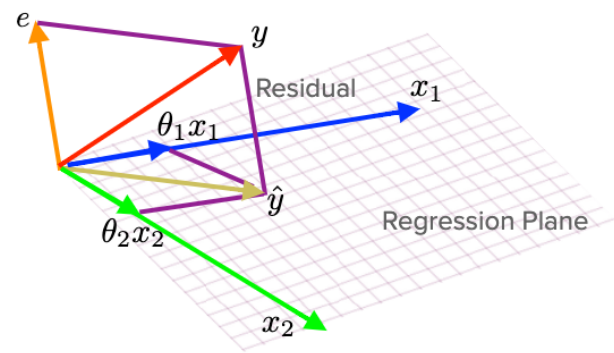
# Geometric Interpretation of MSE

2D case without intercept

model predict $\hat{y} = \theta_1 x_1 + \theta_2 x_2$

residual $e = y - \hat{y} = y - (\theta_1 x_1 + \theta_2 x_2)$

minimize $\mathrm{MSE} = (\theta_1 x_1 + \theta_2 x_2 - y)^2 = e^2$




source: https://www.datasciencecentral.com/profiles/blogs/linear-regression-geometry

MSE solution:

the number of data points

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \, ||X\boldsymbol{\theta} - \boldsymbol{y}||_2^2 = \sum_{i=1}^{N} (y_i - \boldsymbol{\theta}^\top \boldsymbol{x}_i)^2$$

Differentiate and set to zero

$$\frac{\partial \, ||X\boldsymbol{\theta} - \boldsymbol{y}||_2^2}{\partial \, \boldsymbol{\theta}} = 2X^\top (X\boldsymbol{\theta} - \boldsymbol{y}) = 0$$

$$\Rightarrow \; \boldsymbol{\theta}_{\text{MSE}} = (X^\top X)^{-1} X^\top \boldsymbol{y}$$

$$\boldsymbol{y} = X\boldsymbol{\theta}$$

$$\text{Index i} \downarrow \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \end{bmatrix} = \underbrace{\begin{bmatrix} \text{—} & \boldsymbol{x}_0 & \text{—} \\ \text{—} & \boldsymbol{x}_1 & \text{—} \\ \text{—} & \boldsymbol{x}_2 & \text{—} \\ & \vdots & \end{bmatrix}}_{\text{design matrix}} \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \vdots \end{bmatrix}$$

MLE solution:

$$\log \mathcal{L}(\boldsymbol{\theta}|X,\boldsymbol{y}) = \log p(\boldsymbol{y}|X,\boldsymbol{\theta}) = \log \prod_{i=1}^{N} p(y_i|\boldsymbol{x}_i,\boldsymbol{\theta}) = \sum_{i=1}^{N} \log p(y_i|\boldsymbol{x}_i,\boldsymbol{\theta})$$

$$= \sum_{i=1}^{N} \log \frac{1}{(\sigma\sqrt{2\pi})^d} e^{-\frac{1}{2\sigma^2}(y_i - \boldsymbol{\theta}^\top \boldsymbol{x}_i)^2}$$

the number of input features

$$= \sum_{i=1}^{N} \log \frac{1}{(\sigma\sqrt{2\pi})^d} + \sum_{i=1}^{N} \log e^{-\frac{1}{2\sigma^2}(y_i - \boldsymbol{\theta}^\top \boldsymbol{x}_i)^2}$$

$$= -\frac{Nd}{2}\log 2\pi\sigma^2 - \frac{1}{2\sigma^2}\sum_{i=1}^{N}(y_i - \boldsymbol{\theta}^\top \boldsymbol{x}_i)^2$$

$$= -\frac{Nd}{2}\log 2\pi\sigma^2 - \frac{1}{2\sigma^2}(\boldsymbol{y} - X\boldsymbol{\theta})^\top(\boldsymbol{y} - X\boldsymbol{\theta})$$

$$\boldsymbol{y} = X\boldsymbol{\theta} + \boldsymbol{\eta}$$

Index i

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \end{bmatrix} = \underbrace{\begin{bmatrix} \text{—} & \boldsymbol{x}_0 & \text{—} \\ \text{—} & \boldsymbol{x}_1 & \text{—} \\ \text{—} & \boldsymbol{x}_2 & \text{—} \\ & \vdots & \end{bmatrix}}_{\text{design matrix}} \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \vdots \end{bmatrix} + \begin{bmatrix} \eta_0 \\ \eta_1 \\ \eta_2 \\ \vdots \end{bmatrix}$$

$\eta \sim \mathcal{N}(0, \sigma^2)$

## MLE solution:

$$\log \mathcal{L}(\boldsymbol{\theta}|X,\boldsymbol{y}) = \log p(\boldsymbol{y}|X,\boldsymbol{\theta}) \; = -\frac{Nd}{2}\log 2\pi\sigma^2 - \frac{1}{2\sigma^2}(\boldsymbol{y}-X\boldsymbol{\theta})^\top(\boldsymbol{y}-X\boldsymbol{\theta})$$

Differentiate and set to zero

$$\frac{\partial \log \mathcal{L}(\boldsymbol{\theta}|X,\boldsymbol{y})}{\partial\, \boldsymbol{\theta}} = -\frac{1}{\sigma^2}X^\top(X\boldsymbol{\theta}-\boldsymbol{y}) = 0$$

$$\Rightarrow \boldsymbol{\theta}_{\mathrm{MLE}} = (X^\top X)^{-1}X^\top \boldsymbol{y}$$

(same as MSE when the noise is Gaussian)

$$\boldsymbol{y} = X\boldsymbol{\theta} + \boldsymbol{\eta}$$

$$\text{Index i} \downarrow \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \end{bmatrix} = \underbrace{\begin{bmatrix} \text{—} & \boldsymbol{x}_0 & \text{—} \\ \text{—} & \boldsymbol{x}_1 & \text{—} \\ \text{—} & \boldsymbol{x}_2 & \text{—} \\ & \vdots & \end{bmatrix}}_{\text{design matrix}} \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \vdots \end{bmatrix} + \begin{bmatrix} \eta_0 \\ \eta_1 \\ \eta_2 \\ \vdots \end{bmatrix}$$

$$\eta \sim \mathcal{N}(0,\sigma^2)$$

# Polynomial Regression

$$y = \theta_0 + \theta_1 x + \theta_2 x^2 + \ldots + \theta_P x^P = \sum_{p=0}^{P} \theta_p x^p$$

More generally,
in vector form,
for one data point i

$$y_i = \boldsymbol{\theta}^\top \boldsymbol{x}_i \qquad \forall i = 1, \ldots, N$$

$\boldsymbol{\theta} = [\theta_0, \theta_1, \theta_2, \ldots]^\top$

The only difference

$\boldsymbol{x}_i = [1, x_i, x_i^2, \ldots]^\top$

$\boldsymbol{x}_i = [1, x_{i,1}, x_{i,2}, \ldots]^\top$ linear

Build up to
matrix version

$$\boldsymbol{y} = X\boldsymbol{\theta}$$

$$\text{Index i} \downarrow \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \end{bmatrix} \underbrace{\begin{bmatrix} \text{---} & \boldsymbol{x}_0 & \text{---} \\ \text{---} & \boldsymbol{x}_1 & \text{---} \\ \text{---} & \boldsymbol{x}_2 & \text{---} \\ & \vdots & \end{bmatrix}}_{\text{design matrix}} \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \vdots \end{bmatrix}$$

# Notebook

In the following notebook, you will

1. Build design matrix for multiple linear regression.
2. Calculate MSE/MLE solutions with matrix forms.

Optional exercise:

1. Generate simulated dataset from a polynomial model.
2. Build design matrix with increasing powers.
3. Calculate MSE/MLE solutions with matrix forms.

Enjoy!!

# W1D3 Tutorial 5: Bias-variance Tradeoff

First order polynomial (Linear regression)

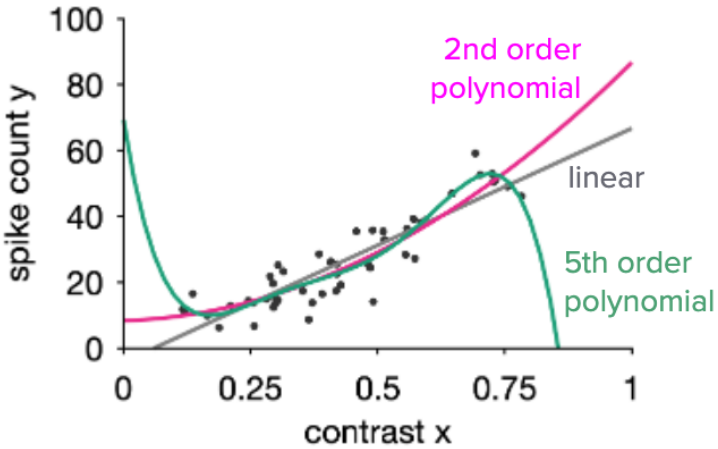
$$y = \theta_1 x + \theta_0$$

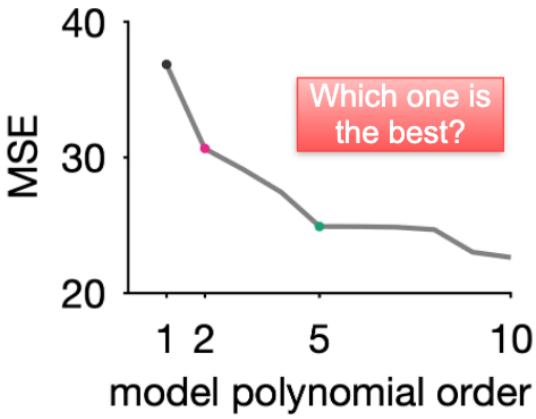
Second order polynomial

$$y = \theta_2 x^2 + \theta_1 x + \theta_0$$

Higher order polynomial

$$y = \sum_{p=0}^{5} \theta_p x^p$$

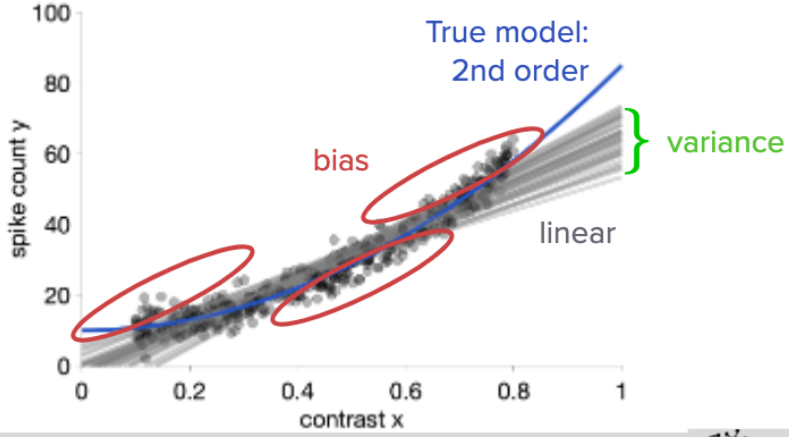


# Bias and Variance

**Bias:** systematic deviation from structure underlying data, low model complexity.

**Variance:** capturing variability in predictions from models trained on different datasets, high model complexity.




**Low** model complexity

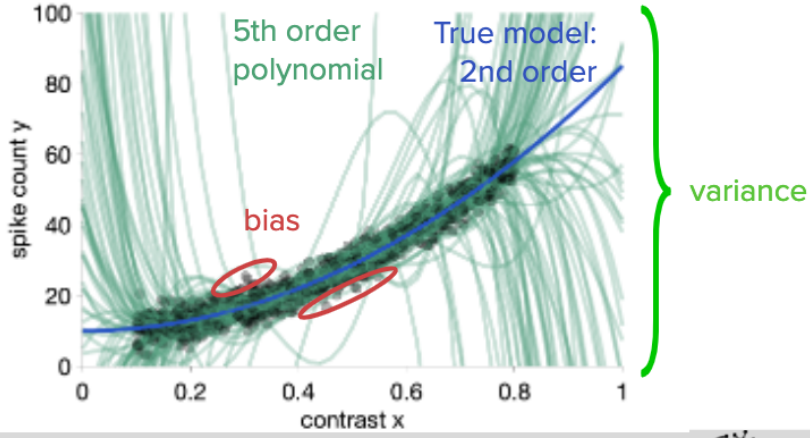
**Bias is large:** the linear slope always misses many data points

**Variance is small:** linear slopes are close to each other across samples

# Bias and Variance

**Bias:** systematic deviation from structure underlying data, low model complexity.

**Variance:** capturing variability in predictions from models trained on different datasets, high model complexity.




**High** model complexity

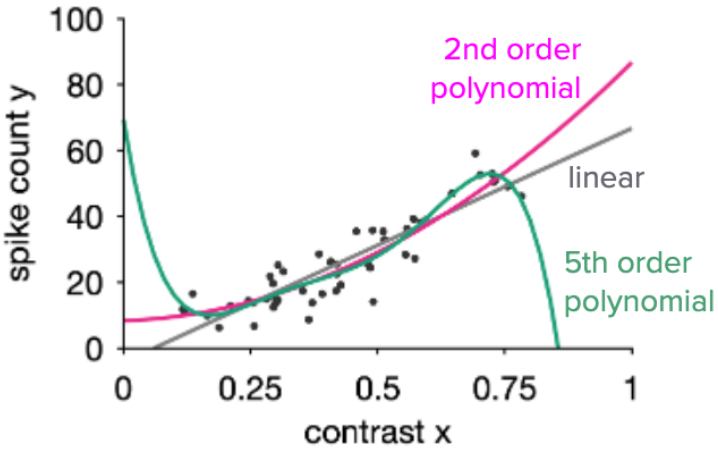
**Bias is small:** the 5th order polynomials go through almost all data points

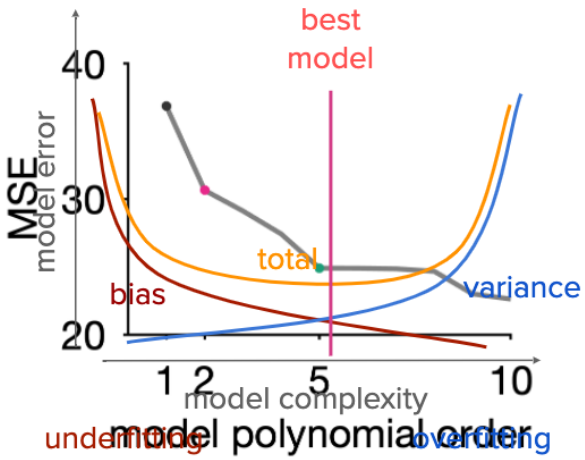
**Variance is large:** the 5th order polynomials are much different from each other

# Bias-variance Trade-off

**Total error** = **bias** + **variance**

**Best model: balances bias / variance**



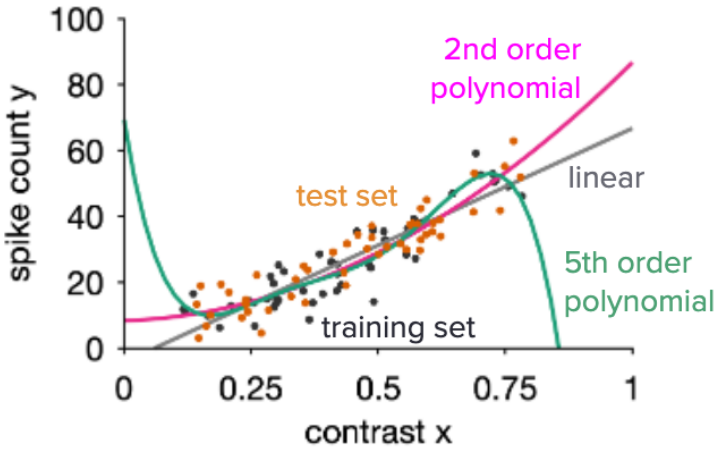



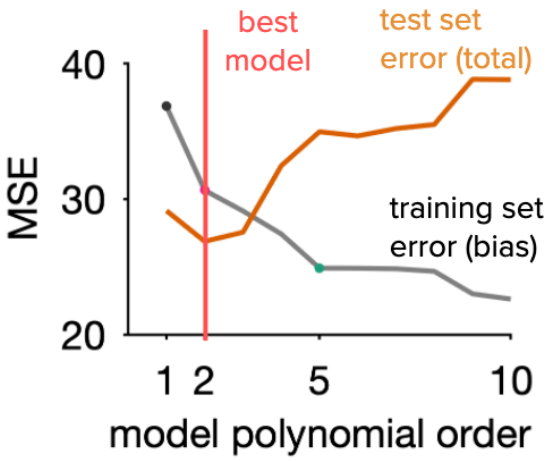
# Bias-variance Trade-off

**Model evaluation** aims at estimating the *generalization error* of the selected model, i.e., how well the selected model performs on unseen data.

**Training set**: data used for fitting

**Test set**: unseen data (not used for fitting)



# Notebook

In the following notebook, you will

1. Continue the polynomial regression exercise.
2. Split the data into training and test sets.
3. Fit different polynomial models using the training set and test on the test set.
4. Inspect the training error and the test error.
5. Explore overfitting and underfitting.

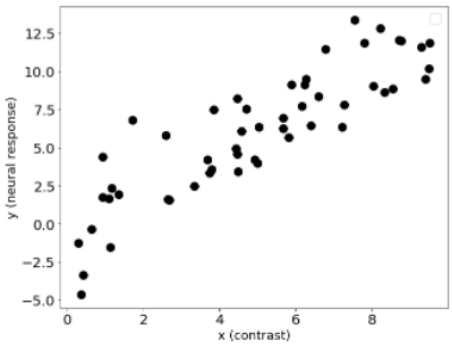
Enjoy!!

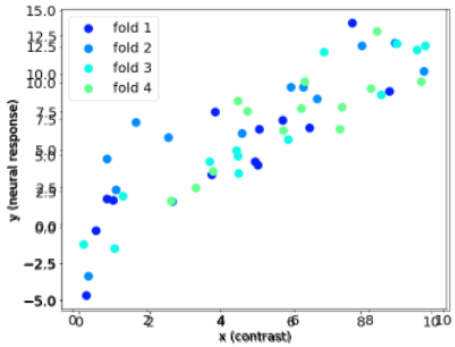
# W1D3 Tutorial 6: Cross Validation

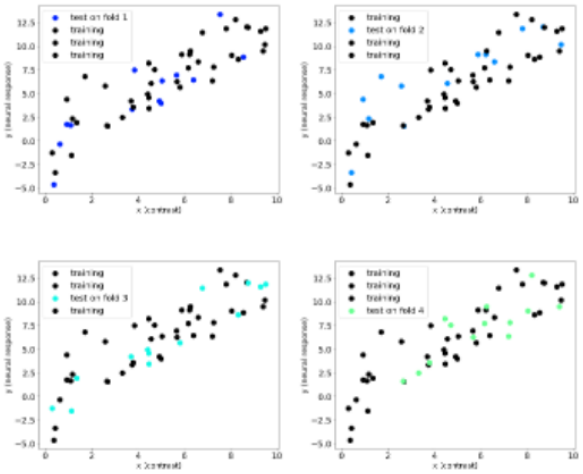
# K-fold Cross Validation

**Motivation:** one random split could result in a biased training-test distribution

**Solution:** multiple random splits with k folds



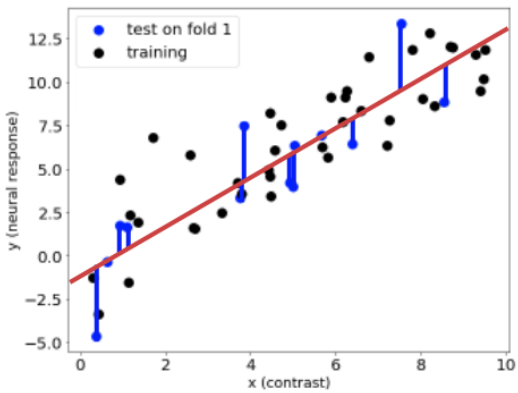



# K-fold Cross Validation

**Motivation:** one random split could result in a biased training-test distribution

**Solution:** multiple random splits with k folds.




**Steps:**

0. Define a model to evaluate, e.g. $y = \sum_{p=0}^{P} \theta_p x^p$ when P=1

1. Take one fold as the test set, the rest as the training set.
2. Fit the model with the training set, and get the optimal $\theta_p$.
3. Estimate MSE on the test set using the above $\theta_p$ ,

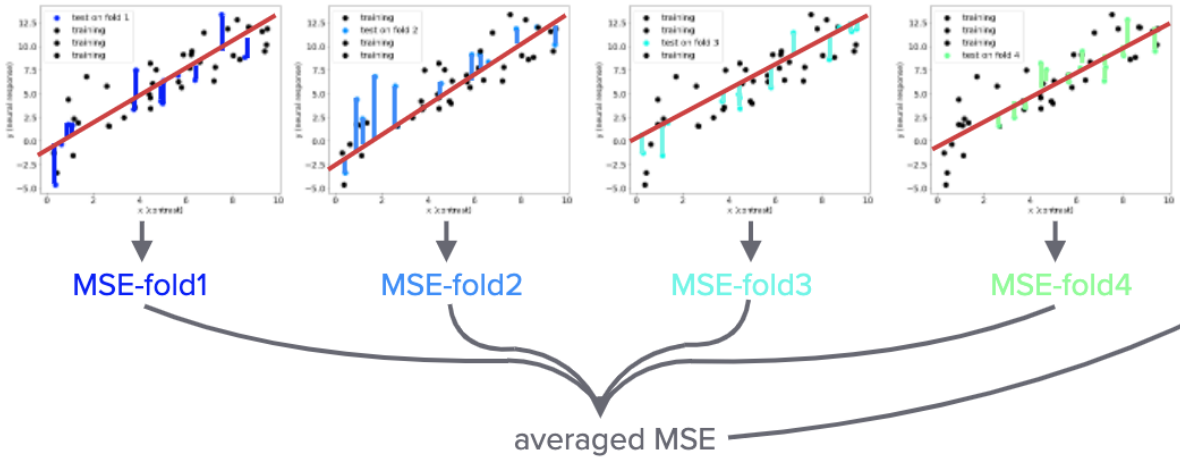
$$\mathrm{MSE}_{\mathrm{test}} = \frac{1}{N_{\mathrm{test}}} \sum_{i=1}^{N_{\mathrm{test}}} \left(\sum_{p=0}^{P} \theta_p x_{\mathrm{test}}^p - y_{\mathrm{test}}\right)^2$$

where $\{x_{\mathrm{test}}, y_{\mathrm{test}}\}_{i=1}^{N_{\mathrm{test}}}$ denotes the test set.

# K-fold Cross Validation

**Motivation:** one random split could result in a biased training-test distribution

**Solution:** multiple random splits with k folds.




model

$$y = \sum_{p=0}^{P} \theta_p x^p$$

when P=1

# K-fold Cross Validation

**Motivation:** one random split could result in a biased training-test distribution

**Solution:** multiple random splits with k folds.

model

$$y = \sum_{p=0}^{P} \theta_p x^p$$

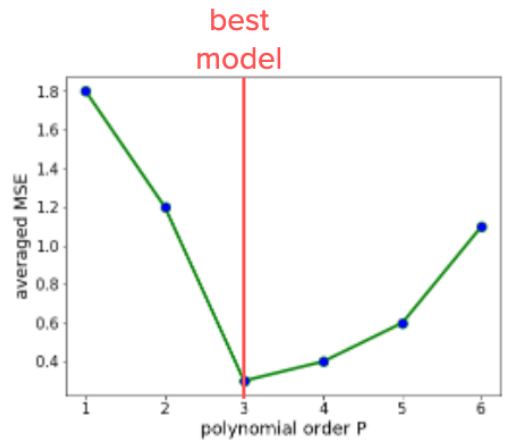
when P=1

averaged MSE

→

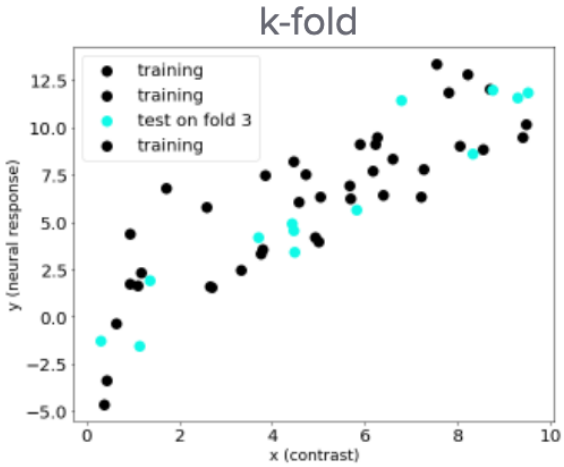
Repeat the same 4-fold cross validation on other Ps, i.e. P = {2, 3, 4…}, get averaged test MSE

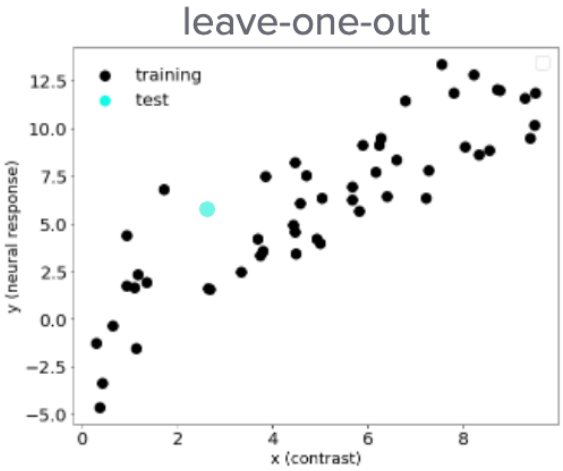
→



# Leave-one-out Cross Validation

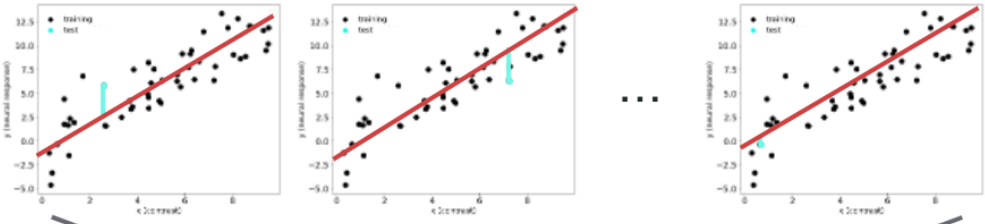
Instead of leaving a fold out for test, leaving one data point out.



# Leave-one-out Cross Validation

Instead of leaving a fold out for test, leaving one data point out.

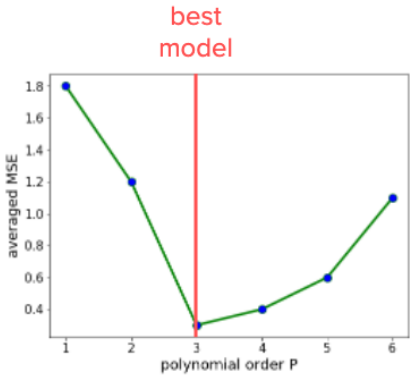



N times

averaged MSE →

Repeat the same leave-one-out cross validation on all Ps, i.e. P = {1, 2, 3, 4…}, get averaged test MSE

→



# Notebook

In the following notebook, you will

1. Implement K-fold cross validation for model selection of polynomial regression.

Enjoy!!