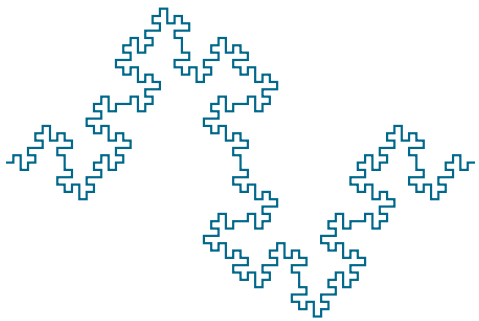
NGS Course

Week 1

Biostatistics and Bioinformatics



Summer 2015

Outline

Section 1

Introduction

RNA-Seq: A tool for measuring abundance of RNA from cells

	gene	AGTCAA	AGTTCC	ATGTCA	CCGTCC	GTCCGC	GTGAAA
20	GeneID:12930133	23	36	32	28	34	31
21	GeneID:12930134	4	2	0	0	6	0
22	GeneID:12930135	19	19	24	13	57	22
23	GeneID:12930136	18	39	47	36	35	26
24	GeneID:12930137	175	238	227	88	103	97
25	GeneID:12930138	27	49	46	47	24	37
26	GeneID:12930139	44	63	43	20	50	24
27	GeneID:12930140	17	23	18	8	23	13
28	GeneID:12930141	2	3	0	0	0	2
29	GeneID:12930142	4	3	1	10	8	9
30	GeneID:12930143	746	928	754	723	831	776

Data from dry run.

PCR/Microarray versus RNA-seq: Common objectives and challenges

- Hypothesis testing: Is the RNA level related to a phenotype, or changed in response to treatment or over time
- Effect size estimation: How to quantify the effect size and then how to estimate it from data
- Classification: Predict an outcome on the basis of baseline RNA levels from multiple genes
- Class Discovery: Discover subsets on the basis of baseline levels or changes in the levels of multiple genes
- Multiplicity: several candidate genes or genome-wide analysis

PCR/Microarray versus RNA-seq: Main Difference

- PCR/Microarray
  - Quantify the "expression" of a gene
- RNA-seq
  - The observed data are digital counts
- Two general approaches for analysis of RNA-seq
  - Two-stage method: Convert counts to "Expression" (e.g., RPKM, FPKM, TPM) and then plug these into a standard test (e.g., t-test)
  - One-stage method: Relate the counts directly to the phenotype (through statistical methods for modeling counts)

Emphasis, Focus, Approach and Topics

- Concepts rather than on mechanics (e.g., which software or method to use to fit a regression model)
- How statistical concepts are misunderstood or misinterpreted
- How and why things could go wrong
- Use simulation as a tool to illustrate these issues
- Topics:
  - Statistical Inference (testing and estimation)
  - Supervised learning (classification and regression)
  - Unsupervised learning (class discovery)
  - Multiple testing
  - Distributions and regression models for counts
- Week 1: Focus on general issues
- Week 2 and later: Focus on RNA-Seq specific issues

On Statistics, Conclusions and Solutions

"No isolated experiment, however significant in itself, can suffice for the experimental demonstration of any natural phenomenon; for the 'one chance in a million' will undoubtedly occur, with no less and no more than its appropriate frequency, however surprised we may be that it should occur to us."

Ronald Aylmer Fisher (The Design of Experiments (1935), 16)

"Doing statistics is like doing crosswords except that one cannot know for sure whether one has found the solution."

John Wilder Tukey (Annals of Statistics, 2002:30(6))

Section 2

Elements of Statistical Inference

Statistical Hypothesis Testing (Recap of Yesterday)

- Formulate a scientific hypothesis
- Formulate the corresponding statistical hypothesis
- This will consist of a *null* hypothesis (H_0) and an *alternative* hypothesis (H_1)
- Specify an experimental design
- Specify the testing procedure to be used:
  - an appropriate test statistic
  - decision rule based on the test statistic (typically under a set of assumptions)
- Execute Experiment (collect data)
- Based on the amount of evidence using the decision rule
  - either conclude there is evidence to reject the null hypothesis H_0 in favor of H_1
  - or fail to reject H_0 (inconclusive)

IMPORTANT: Failing to reject H_0 does *not* afford us to conclude that H_1 is *true*

Null versus Alternative

- The null hypothesis posits the status quo
- It is the conservative hypothesis
- In the US legal system, the defendant is assumed to be innocent
- The null hypothesis: Defendant is innocent
- Study: Investigate if gene XYZ is differentially expressed with respect to treatment
- In other words, does the distributions of the feature of the gene you are interested in change when the experimental unit is exposed to treatment?
  - H_0 gene XYZ is *not* differentially expressed with respect to treatment
  - H_1 gene XYZ is differentially expressed with respect to treatment

More on Null versus Alternative

- Suppose that your are studying the effect of a drug in a clinical study
- Safety Study:
  - H_0: Drug is toxic
  - H_1: Drug is safe
- Efficacy study:
  - H_0: Drug is not efficacious
  - H_1: Drug is efficacious

Notation: True versus False Null Hypothesis

- The truth may be stated either by the null or alternative hypothesis
- If the truth is stated by the statement of the null hypothesis, we will say that
  - The null hypothesis is true
  - or call it a true null hypothesis
- If the truth is stated by the statement of the alternative hypothesis, we will say that
  - The null hypothesis is false
  - or call it a false null hypothesis
- We will use these terms for notational convenience

Type I and II errors

- Type I Error: Erroneously decide in favor of the alternative hypothesis (reject a true null hypothesis)
- Type II Error: Erroneously decide in favor of the null hypothesis (fail to reject a false null hypothesis)
- The so called "alpha" level is the probability of a type I error
- The "power" of a test, is the complement of the probability of the type II error
- IMPORTANT: There is a trade-off between these two error rates

Type I and II error trade-off

- In our court system, a defendant is assumed innocent until proven guilty
  - Type I error: Convict an innocent defendant
  - Type II error: Free a guilty defendant
- If the prosecution gets too much leeway, the the likelihood of convicting an innocent defendant increases
- Conversely, if the prosecution is reigned in by the judge, the likelihood of letting a guilty defendant walk free increases
- Similar analogy in the case of a smoke detector
  - Dialing up the sensitivity, increases the likelihood of annoying beeps when using your toaster
  - Dialing down the sensitivity, increases the likelihood of missing a true fire

Notation: Decision

- false-positive (FP): Reject a true null hypothesis (Type I error)
- true-positive (TP): Reject a false null hypothesis
- false-negative (FN): Fail to reject a false null hypothesis (Type II error)
- true-negative (TN): Fail to reject a false null hypothesis
- We will use these terms for notational convenience

Three Decision Rules

- Following the collection of data, consider using one of the three decision rules
- Decision Rule 1: Reject H_0
- Decision Rule 2: Do not reject H_0
- Decision Rule 3: Flip a coin: Reject H_0 if tails and do not reject H_0 if heads
- What are the type I and II error rates for these decision rules?
- Which one would you choose?

Decision Rule 1

- Decision: Reject H_0
- If H_0 is true, then it will be rejected
- A false-positive decision will be made if H_0 is true
- $\alpha = 1$
- If H_0 is false, then it will be rejected
- A true-positive decision will be made if H_0 is false
- $\beta = 0$

Decision Rule 2

- Decision: Do not reject H_0
- If H_0 is true, then it will not be rejected
- A false-positive decision will not be made
- $\alpha = 0$
- If H_0 is false, then it will not be rejected
- A false-negative decision is will be made
- $\beta = 1$

Decision Rule 3

- Decision: Flip a coin: Reject H_0 if tails and do not reject H_0 if heads
- If H_0 is true, then the probability of rejecting it is one-half
- $\alpha = \frac{1}{2}$
- If H_0 is false, then probability of not rejecting it is one-half
- $\beta = \frac{1}{2}$

A Bad Rule is a Valid (but bad) Decision Rule

- Decision Rule 1: Reject H_0
  - $\alpha = 1$ and $\beta = 0$
- Decision Rule 2: Do not reject H_0
  - $\alpha = 0$ and $\beta = 1$
- Decision Rule 3: Flip a coin: Reject H_0 if tails and do not reject H_0 if heads
  - $\alpha = \frac{1}{2}$ and $\beta = \frac{1}{2}$
- Note that these decision rules effectively ignore the data
- While they are poor decision rules, they are technically valid decision rules
- A poor statistical approach will effectively reduce to one of the three
- Note that while $\alpha + \beta = 1$ in all these cases, that is generally not the case
- The type I error is generally *not* the complement of the type II error

A Simple Example: Formulation

- You suspect that a coin (H on side and T on the other) is not fair (biased)
- Let π denote the probability that the coin lands a head on any given toss
- A coin is "fair" if $\pi = \frac{1}{2}$
- or is "biased" otherwise (i.e., $\pi \neq \frac{1}{2}$)
- It is more likely to land a tail if $\pi < \frac{1}{2}$
- or more likely to land a head if $\pi > \frac{1}{2}$

A Simple Example: Statistics and Plain English

- The statistical hypotheses could be succinctly stated as:
  - Test $H_0 : \pi = \frac{1}{2}$ against $H_1 : \pi \neq \frac{1}{2}$
- In English:
  - We give benefit of the doubt to the fact that the coin is fair and then will, under this assumption, ascertain if there is enough evidence, on the basis of the data, to conclude that the coin is biased

A Simple Example: Decision Rule

- Following the formulation of the hypotheses, we have to decide on an experimental design and a decision rule
- These, along with the specification of the hypotheses, should be done before collecting data. Why?
- Our experimental design: flip the coin $n = 12$ times
- Why $n = 12$ and not say $n = 13$ (more on this later)
- A reasonable decision rule for this type of design is to use the so called Binomial Test
- We will skip the technical details on the test

A Simple Example: Collect Data

- We conduct the experiment and observe

```
##  [1] "T" "T" "T" "T" "T" "H" "T" "H" "T" "T" "T" "T"
```

- There are (per design) 12 flips of the coin
- We observe 2 heads and 10 tails
- What would you conclude?
- Would you reject if the number of heads were 3?
- how about 4?
- or 5?

A Simple Example: Binomial Test in Action

- We conduct the binomial test

```
test=binom.test(x=sum(x=='T'),n=length(x),p=1/2)
test

##
##  Exact binomial test
##
## data:  sum(x == "T") and length(x)
## number of successes = 10, number of trials = 12, p-value = 0.03857
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.5158623 0.9791375
## sample estimates:
## probability of success
##              0.8333333
```

- What should we conclude?
- At the $\alpha = 0.05$ level, there is sufficient evidence to reject the hypothesis that the coin is fair (P-value=0.039)
- Note that there is *not* sufficient evidence to reject the null if you wish to control the type I error rate at $\alpha = 0.01$

The Two-Sample Problem: Formulation

- Question: Does treatment alter the distribution of the RNA abundance of a given gene?
- μ_0 denotes the average abundance level of the untreated group
- In other words: If we take a large random sample of untreated experimental units from the untreated group, the "average" RNA abundance for the sample will be about μ_0
- μ_1 denotes the average RNA abundance of the treated group

The Two-Sample Problem: Treatment Effect

- There is a treatment effect if the means, μ_0 and μ_1, differ:
  - Null Hypothesis: There is no treatment effect ($\mu_0 = \mu_1$)
  - Alternative Hypothesis: There is a treatment effect ($\mu_0 \neq \mu_1$)
- Why is the null hypothesis not $\mu_0 \neq \mu_1$?
- and the alternative hypothesis not ($\mu_0 = \mu_1$)?

The Two-sample Problem: Assumptions

- The decision rule is typically chosen on the basis of some putative assumptions
- Distributional assumptions:
  - RNA abundance for the untreated group follows a normal distribution with mean μ_0 and variance σ^2
  - RNA abundance for the treated group follows a normal distribution with mean μ_1 and variance σ^2
- Assumptions:
  - the distributions are normal (questionable assumption for digital counts)
  - the variability of the RNA abundance is not affected by treatment (same σ^2)
  - Another implicit key assumption: The experimental units are independent
- The (two-sample) t-test is a commonly method for testing this hypothesis under the given set of assumptions

Quick Note: Conservative versus Anti-conservative; Robustness

- The properties of the decision rule will depend on these underlying assumptions
- They may be greatly sensitive to these assumptions
- The type I error of a decision procedure we hope to achieve is called the nominal level
- Example: If we claim that the nominal level of our decision is 0.05, then we are asserting that the probability of committing a false-positive is at most 0.05.
- If the *actual* type I error rate exceeds the nominal level the test is said to be anti-conservative
- If the *actual* type I error rate is less than the nominal level the test is said to be conservative
- A decision rule that is not sensitive to the underlying assumptions, with respect to type I error control, is said to be robust

Designing the Experiment

- The sample size to achieve the desired power at a given type I error rate depends on the effect size
- Given everything else fixed, a larger effect size requires a smaller size to achieve a power at a given type I error rate
- The effect size for the two-sample t-test is defined as

$$\Delta = \frac{|\mu_0 - \mu 1|}{\sigma}$$

- The numerator $|\mu_0 - \mu 1|$ is the difference (in absolute value) of the means
- The size of this difference (how large it is) is in relation to (scaled by) the standard deviation

Sample Size Formula

- The sample size formula the two-sample t-test is

$$n = 2\frac{(Z_{1-\alpha} + Z_{1-\beta})^2}{\Delta^2}$$

- Here $Z_{1-\alpha}$ denote the right α tail of a normal distribution
- Let's forget most of the technical details
- Just observe that the sample size decreases as the effect size become larger. Why?
- Many other sample size formulas look very similar

Our Example: The Unknown Truth

- The true values of the unknown parameters:
  - $\mu_0 = 0$
  - $\mu_1 = 2$
  - $\sigma = 5$
- The effect size is

$$\Delta = \frac{|0 - 2|}{5} = 0.4$$

Forgot about the Design

- What is the power if we use 3 units per group

```
##
##      Two-sample t test power calculation
##
##               n = 3
##           delta = 2
##              sd = 5
##       sig.level = 0.05
##           power = 0.05784303
##     alternative = two.sided
##
## NOTE: n is number in *each* group
```

Forgot about the Design

- What is the power if we use 6 units per group

```
##
##      Two-sample t test power calculation
##
##               n = 6
##           delta = 2
##              sd = 5
##       sig.level = 0.05
##           power = 0.09156966
##     alternative = two.sided
##
## NOTE: n is number in *each* group
```

Forgot about the Design

- What is the power if we use 12 units per group

```
##
##      Two-sample t test power calculation
##
##               n = 12
##           delta = 2
##              sd = 5
##       sig.level = 0.05
##           power = 0.1532882
##     alternative = two.sided
##
## NOTE: n is number in *each* group
```

Now Use Experimental Design

- The required sample size, per group, to detect an effect size of

$$\Delta = \frac{|0 - 2|}{5} = 0.4$$

with a power of 0.8, at the 0.05 level is

```
##
##      Two-sample t test power calculation
##
##               n = 99.08057
##           delta = 2
##              sd = 5
##       sig.level = 0.05
##           power = 0.8
##     alternative = two.sided
##
## NOTE: n is number in *each* group
```

How to check the type I Error and Power

- Simulation provide a powerful framework for understanding the properties of the decision rule
- In the case of the two-sample t-test this works as follows
  1. Draw a random sample of size n from a normal distribution with mean μ_0 and standard deviation σ
  2. Draw a random sample of size n from a normal distribution with mean μ_1 and standard deviation σ
  3. Apply the two-sample test to the two data samples and record the P-value
- Now repeat the last three steps a large number of times
- The distribution of these simulated P-values should be similar to the true distribution of the P-value

Simulation Example

- Set parameters

```
set.seed(4141)
n=6;mu0=0;mu1=2;sigma=5
```

- Simulate data

```
x0=rnorm(n,mu0,sigma)
x1=rnorm(n,mu1,sigma)
x0
```

```
## [1] -2.1071177 -0.2402046  2.6668539 -4.4884699  2.6865668  5.1362518
```

```
x1
```

```
## [1]  6.0170556 -4.3949286 -1.4848887 -3.5189476 -8.7897573 -0.4961073
```

- Carry out t-test

```
t.test(x0,x1)
```

```
##
##  Welch Two Sample t-test
##
## data:  x0 and x1
## t = 1.0984, df = 9.104, p-value = 0.3002
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -2.872410  8.312895
## sample estimates:
## mean of x mean of y
##  0.608980 -2.111262
```

Simulation: Important Notes

- Data are generated under the truth
- Parameters and distributions are set by you
- A simulated experiment is to mimic a hypothetical, but real, experiment
- The truth is not known in the context of a real experiment
- IMPORTANT: The decision rule step has to remain *blinded* to this truth
- Computing Exercise: Evaluate the type I error and power for the two-sample example using simulation and formula

Stat 101 Example: One-sided or Two-sided Test?

- Suppose that company XYZ Dairies sells milk in glass bottles
- The company claims that the net content of each bottle is 1 gallon
- Mr. Smith, owner of the ABC Supermarket, suspects he, and ultimately his customers, are being swindled by XYZ
- Let μ denote the mean net content (in gallons) of the *population* of XYZ Dairies milk bottles
- The company claims $\mu = 1$
- Mr. Smith hypothesizes that $\mu < 1$

Stat 101 Example (null vs alternative)

- Mr. Smith has to give benefit of the doubt to company XYZ's claim (i.e., $\mu = 1$)
- The purpose of the experiment is to ascertain if there is sufficient evidence to the contrary (i.e., show $\mu \neq 1$)
- The null hypothesis is formulated as $H_0 : \mu = 0$
- The alternative is formulated as $H_1 : \mu \neq 0$
- Mr. Smith has no interest in gathering evidence for showing that XYZ overfills its bottles (i.e., $\mu > 1$
- In this case, a one-sided hypothesis would be appropriate

Stat 101 Example (continued)

- Hypothesis: Test $H : \mu = 1$ versus $\bar{H} : \mu < 1$
- He collects a random sample of twenty milk bottles.
- Let $X_1, \ldots, X_{20}$ denote the observed net contents for these 20 bottles
- He decides to assume that these are normally distributed with mean μ and variance σ^2

Stat 101 Example (continued)

- The one-sample t-test is a commonly used test for this setting (normality assumption):

$$T = \sqrt{n}\frac{\bar{X}_n - \mu}{s_n}, \tag{1}$$

  where $\bar{X}_n$ and s_n are the sample mean and standard deviations
- Under H where $\mu = 1$ the *sampling* distribution of T follows a t distribution with $n - 1 = 19$ degrees of freedom

Statistical versus Clinical/Biological Significance

- Hypothesis testing is carried out to investigate *statistical* and not *biological* significance
- It is the responsibility of the investigator to pose a biologically relevant hypothesis.
- It is also the responsibility of the investigator to ensure that a statistically significant finding is biologically plausible/realistic
- Statistical significance does not necessarily imply biological significance or vice versa

Biologically but not Statistically Significant

```
set.seed(1122333)
x0=rnorm(3,1,1)
x1=rnorm(3,2,1)
x0
```

```
## [1] -0.25824011  0.02820527  2.20878939
```

```
x1
```

```
## [1] 1.5462733 0.6578732 3.1782064
```

```
t.test(x0,x1)
```

```
##
##  Welch Two Sample t-test
##
## data:  x0 and x1
## t = -1.0572, df = 3.988, p-value = 0.3502
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -4.117361  1.848295
## sample estimates:
## mean of x mean of y
## 0.6595849 1.7941176
```

Statistically but not Biologically Significant

```
x0=c(3.0001,3.0002,3.0003,3.0004,3.0005)
x1=c(3.0006,3.0007,3.0008,3.0009,3.0010)
x0
```

```
## [1] 3.0001 3.0002 3.0003 3.0004 3.0005
```

```
x1
```

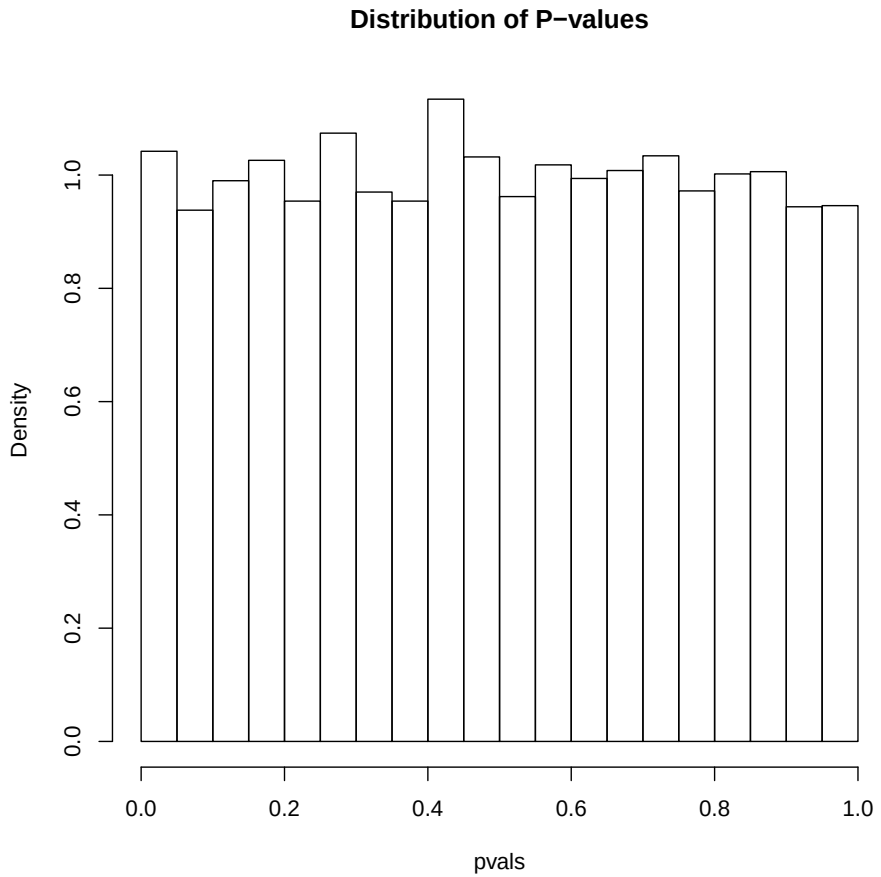
```
## [1] 3.0006 3.0007 3.0008 3.0009 3.0010
```

```
t.test(x0,x1)
```

```
##
##  Welch Two Sample t-test
##
## data:  x0 and x1
## t = -5, df = 8, p-value = 0.001053
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.0007306004 -0.0002693996
## sample estimates:
## mean of x mean of y
##    3.0003    3.0008
```

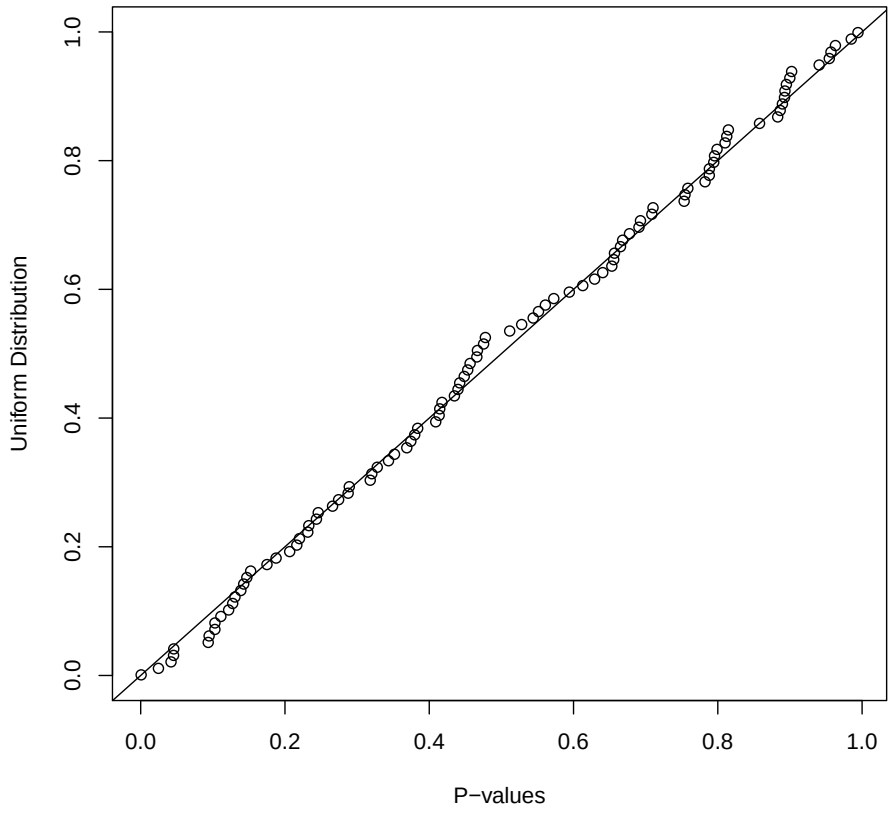
Distribution of *P*-values under H_0

- Under the null hypothesis, the distribution of the *P*-values is uniform
- If you repeat the experiment many times under the null hypothesis (e.g., no differential expression), the distribution of the *P*-values will look like this



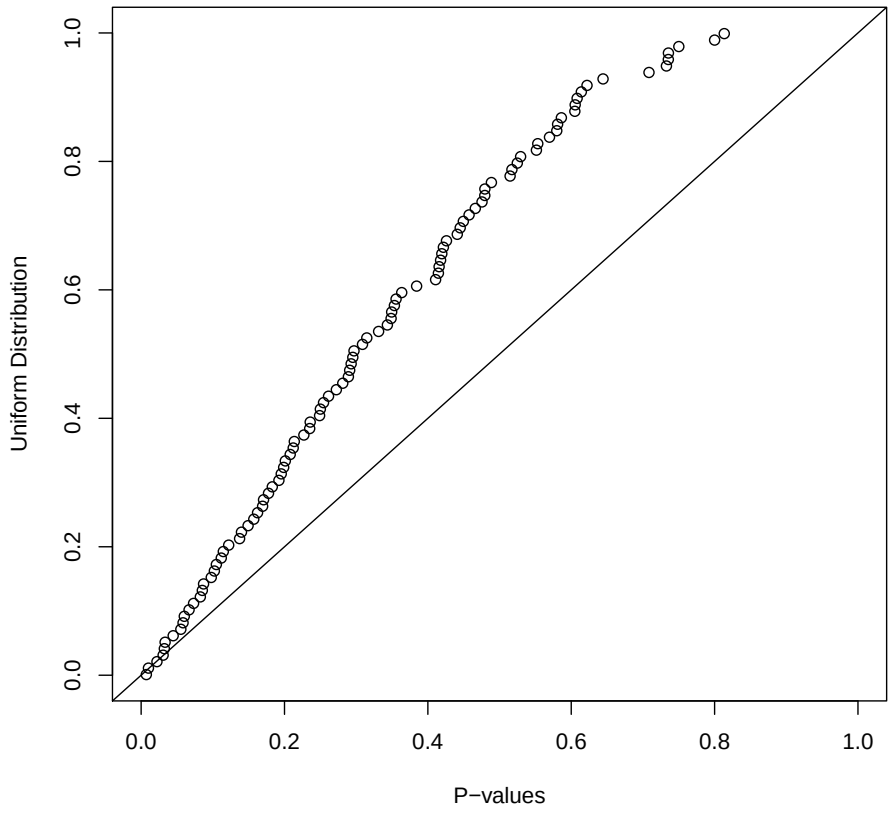
Quantile-Quantile Plot

- An important tool to assess type I error control is the Quantile-Quantile Plot (aka QQ-Plot)
- The plot should look like this under H_0



Quantile-Quantile Plot: Deviation

- A deviation in the QQ-Plot indicates that there may be evidence to reject H_0
- Or that teh decision rule is not accounting for type I error: INFLATION!!



Estimation

- So far we have considered concepts and issues related to hypothesis testing
- What is often of interested is estimate the unknown parameters
- First determine how to quantify the effect size
- Consider the two sample problem
- Examples
  - Mean level for the untreated group μ_0
  - Mean level for the treated group μ_1
  - Fold-change $\rho = \frac{\mu_1}{\mu_0}$
  - Standardized difference $\Delta = |\mu_1 - \mu_0|/\sigma$
- Next figure out how to estimate the effect size
- Two types of estimates
  - Point estimate
  - Interval estimate

Confidence Intervals

- Example: The sample mean (the average of the observations) is a point estimate of the population (true) mean
- It is either equal to the true value of the parameter or is not
- As it is a single number it does not provide any direct measure of accuracy
- An interval estimate incorporates some measure of accuracy
- Thus it is generally more appropriate to present an interval estimate
- A common example of an interval estimate is the confidence interval

Estimation Example

- Assumption: The RNA abundance follows a normal distribution with mean $\mu = 0$ and standard deviation $\sigma = 1$
- Goal: The population mean μ is to be estimated on the basis of sample of size $n = 6$
- Objectives:
  - Produce point estimate of μ
  - Produce a 95% confidence interval of μ
- We will produce these estimates on the basis of the sample mean
- The sample mean is obtained by averaging the n observations

Simulate Experiment 1

- Simulate the data

```
n
```

```
## [1] 6
```

```
mu
```

```
## [1] 0
```

```
sigma
```

```
## [1] 1
```

```
set.seed(12331)
x=rnorm(n,mu,sigma)
```

- Calculate the sample mean

```
mean(x)
```

```
## [1] -0.4014889
```

- Calculate confidence interval

```
# sample standard deviation
s=sd(x)
# Margin of error
error=qt(0.975,df=n-1)*s/sqrt(n)
# A 95% CI
c(mean(x)-error,mean(x)+error)
```

```
## [1] -1.4687200  0.6657421
```

Repeat the Experiment

exp	n	mu	sigma	avg	lcl	ucl	cover	len
1	6	0	1	0.36	0.05	0.67	0	0.62
2	6	0	1	0.67	-0.23	1.57	1	1.80
3	6	0	1	-0.23	-0.89	0.42	1	1.31
4	6	0	1	-0.88	-2.09	0.34	1	2.42
5	6	0	1	-0.88	-1.62	-0.14	0	1.49
6	6	0	1	0.57	-0.64	1.78	1	2.42
7	6	0	1	-0.03	-1.60	1.54	1	3.15
8	6	0	1	-0.62	-1.18	-0.05	0	1.13
9	6	0	1	-0.05	-1.46	1.37	1	2.82
10	6	0	1	0.21	-0.92	1.34	1	2.25

Confidence Interval: Common Misunderstanding

- A (not the) 95% CI for the mean based on the first experiment was $(0.05, 0.67)$
- A (not the) 95% CI for the mean based on the second experiment was $(-0.23, 1.57)$
- It is wrong to say that the probability that the first CI does not contain the true value $\mu = 0$ is 95%
- It is also wrong to say that the probability that the second CI contains the true value $\mu = 0$ is 95%
- We conduct one and only was experiment
- Based on the first experiment, we can say that we are 95% confident that it contains the true value
- That is of course not the case
- If we repeated the experiment a large number of times, 95% of the CIs would cover the true value
- We are 95% confident that the first experiment is among these (which it is not)

Repeat the Experiment

- Now repeat experiment with a sample size of $n = 12$.

exp	n	mu	sigma	avg	lcl	ucl	cover	len
1	12	0	1	0.07	-0.50	0.65	1	1.16
2	12	0	1	-0.07	-0.66	0.51	1	1.16
3	12	0	1	0.68	0.11	1.25	0	1.14
4	12	0	1	-0.32	-1.12	0.49	1	1.60
5	12	0	1	-0.14	-0.82	0.55	1	1.37
6	12	0	1	-0.26	-0.81	0.30	1	1.11
7	12	0	1	-0.13	-0.47	0.21	1	0.68
8	12	0	1	-0.20	-0.76	0.35	1	1.10
9	12	0	1	0.19	-0.32	0.70	1	1.02
10	12	0	1	-0.46	-1.13	0.21	1	1.34

- Compare the widths of the CIs

Quick Note: Estimate versus Estimator

- We use the terms estimate and estimators interchangeably
- There is a subtle but important distinction
- Suppose that you decide to estimate the population mean using the sample mean (once you get the data)
- The sample mean is the estimator
- Its outcome is random before you collect the data
- Once you collect the data and plug them into the estimator you get an (not the) estimate

Section 3

Model Building Illustration

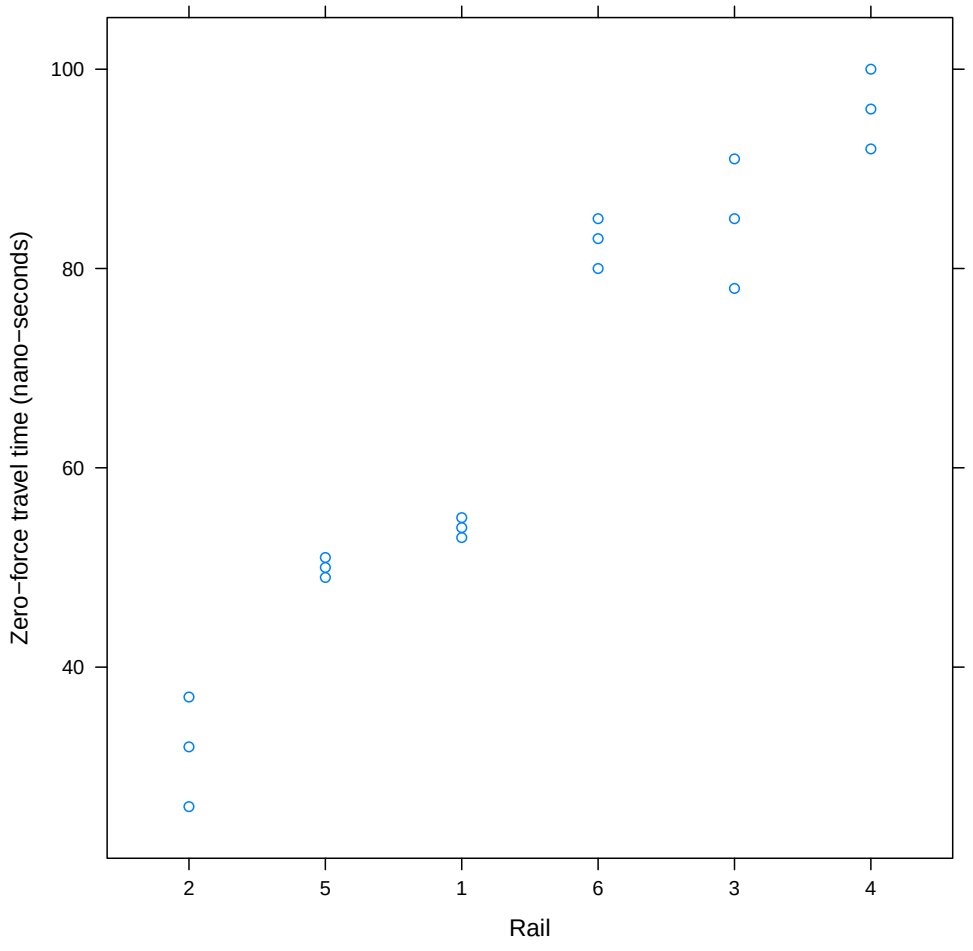
Intra- and Inter-subject Variability

- In most experiments, including RNA-Seq, the variability may not be exclusively due to measurement error
- Another source could be due to repeated measurements
- or sampling from strains or cell lines
- or due to batch effects
- We will motivate these ideas using a classical toy example
- We will illustrate the caveats of properly accounting for these two sources of variability through two simulation studies

Rails Data

- Observation adjusted travel time for ultrasonic head-waves in the rail (nanoseconds).
- Data set: 6 rails; the travel time is sampled three times per rail
- Eighteen measurements
- Six Experimental Units
- Implicit assumption: The six rails are randomly selected from a *large* pool of rails
- What is of interest is neither the batch or any of these 6 rails (specifically)
- What is of interest is the population (the huge pool)

Rail Data



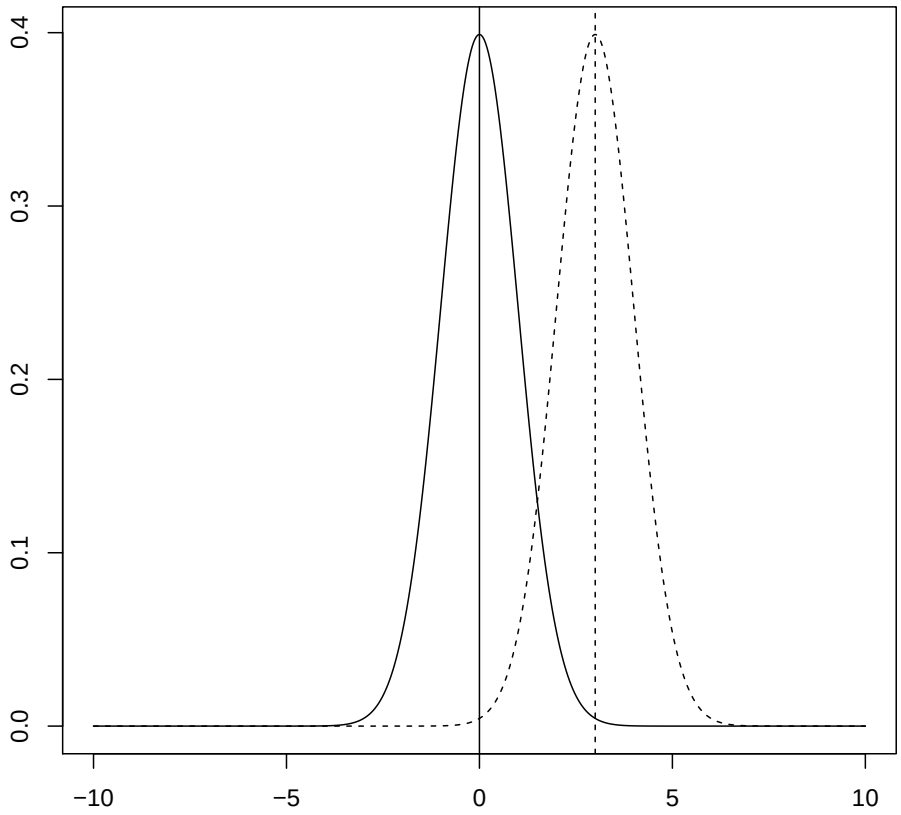
Rail Data: Model Formulation

- μ denotes the *true* travel time
- μ is an unknown fixed quantity
- Y_i denotes the *observed* travel time (for observation $i = 1, \ldots, 18$)
- In absence of noise, true value μ is observed
- In other words, $Y_i = \mu$ for $i = 1, \ldots, 18$

Important Fact about Normal Distribution

- Consider a normal distribution with mean 0 and standard deviation σ
- If the data are shifted by a constant μ, then
  1. resulting distribution remains normal
  2. The mean of the new distribution is $\mu + 0 = \mu$
  3. Its standard deviation remains unchanged
- The last two (but not first) property are true for any distribution

Shift Normal Distribution



Rail Data: Simple Model

- What is observed is a distorted version of μ

$$Y_i = \mu + \epsilon_i$$

- Notes:
  - Y_i is observable
  - ϵ_i is *not* observable
  - μ is an unknown parameter
- The variability observed here is exclusively attributed to the measurement error ϵ_i

Linear Model

```
summary(lm(travel~1,data=Rail))
```

```
##
## Call:
## lm(formula = travel ~ 1, data = Rail)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -40.50 -16.25   0.00  18.50  33.50
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   66.500      5.573   11.93  1.1e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 23.65 on 17 degrees of freedom
```

Rail Data: Account for Two Source of Variability

- What is observed is a distorted version of μ
- It is distorted by a ra
- Y_{ij}: Index the rail by $i = 1, \ldots, 6$ and the replicate by $j = 1, 2, 3$
- Y_{23}: The obeservation for the third replicate for rail 2
- Model

$$Y_{ij} = \mu + b_i + \epsilon_{ij}$$

- Notes:
  - Y_{ij} is observable
  - b_i is *not* observable
  - ϵ_{ij} is *not* observable
  - μ is an unknown parameter

Linear Mixed Effects Model

```
lme(travel~1,random=~1|Rail,data=Rail)
```

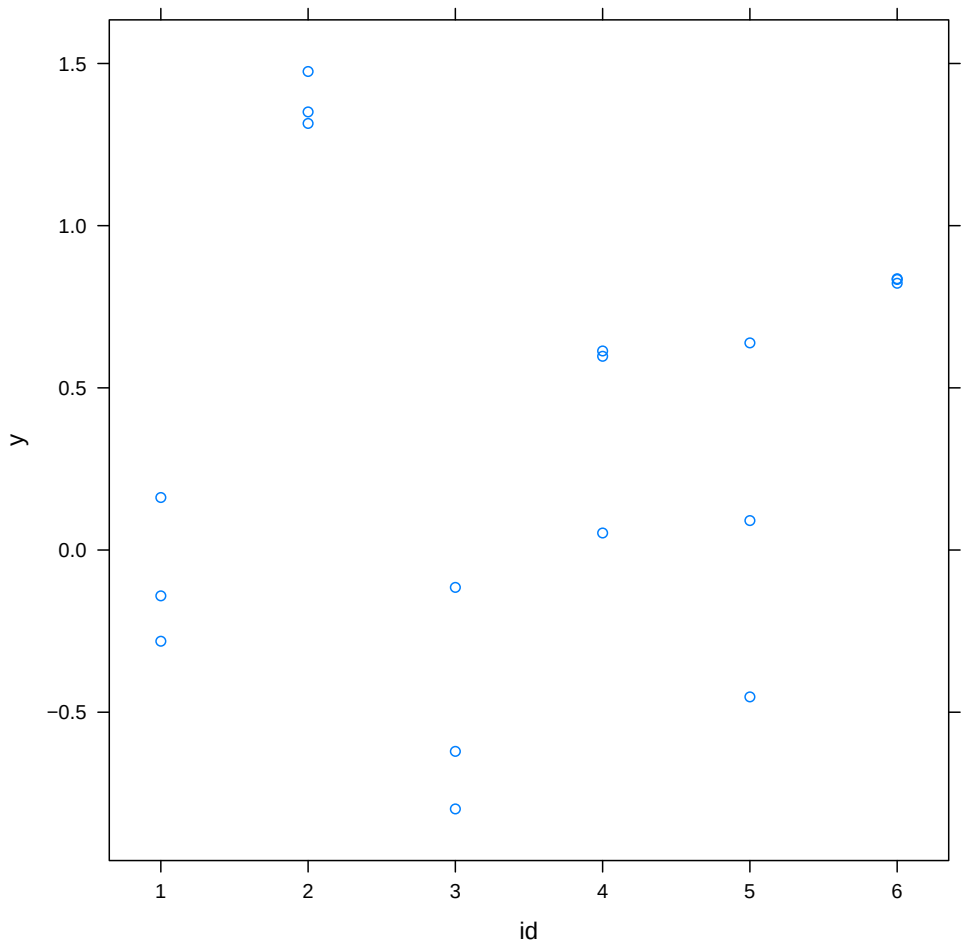
```
## Linear mixed-effects model fit by REML
##   Data: Rail
##   Log-restricted-likelihood: -61.0885
##   Fixed: travel ~ 1
## (Intercept)
##        66.5
##
## Random effects:
##  Formula: ~1 | Rail
##         (Intercept) Residual
## StdDev:    24.80547 4.020779
##
## Number of Observations: 18
## Number of Groups: 6
```

Is the Mixed Model Adequate?

- Assumptions:
  - b_i is normally distributed $N[0, \sigma_b^2]$
  - σ_b^2 does *not* depend on i (homoscedastic)
  - ϵ_{ij} is normally distributed $N[0, \sigma_e^2]$
  - σ_e^2 does *not* depend on i or j (homoscedastic)
  - Error model is additive (could be multiplicative)

Example 1: Setup

- What are the ramifications for ignoring the clustering?
- We will sample 6 experimental units each with three replicates
- $\mu = 0, \sigma_e = 0.25, \sigma_b = 0.5$



Example 1: Simulation

- Simulation outline
  1. Simulate a data set
  2. Test $H_0 : \mu = 0$ ignoring the random effect (save P-value)
  3. Test $H_0 : \mu = 0$ accounting for the random effect (save P-value)
- Repeat the three steps 999 additional times
- Given that the *true* $\mu = 0$ (by design), we would expect 50 of these P-values to be less than 0.05
- Why?

Example 1: Results

```
set.seed(210)
res=replicate(B3,sim.ranef(3,6,0.25,0.5,verbose=FALSE))
mean(res[1,]<0.05)
```

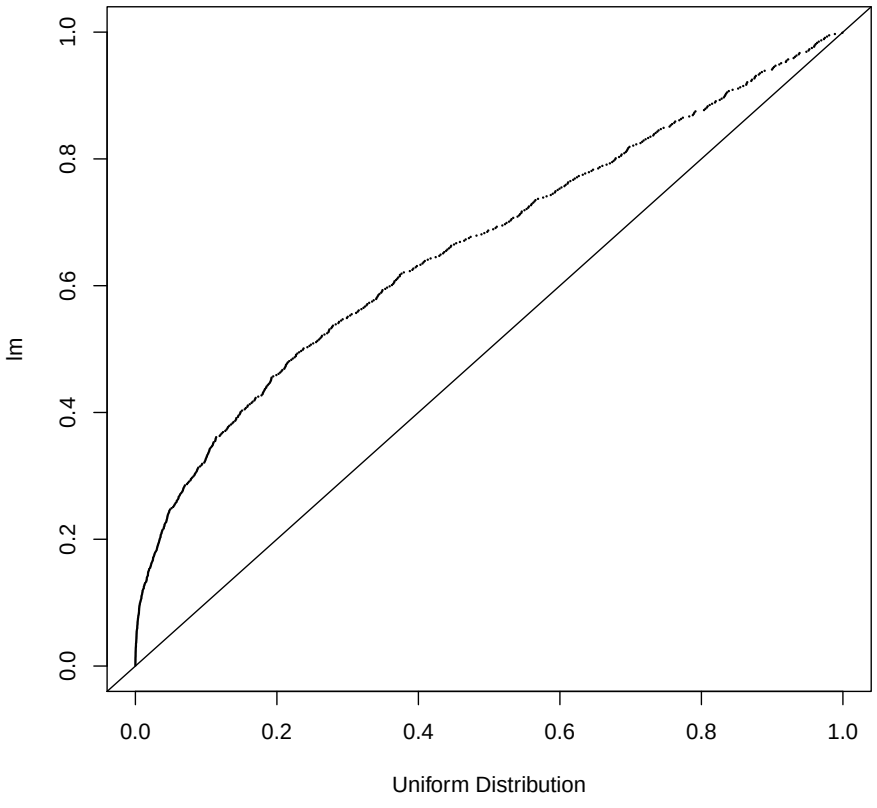
```
## [1] 0.247
```

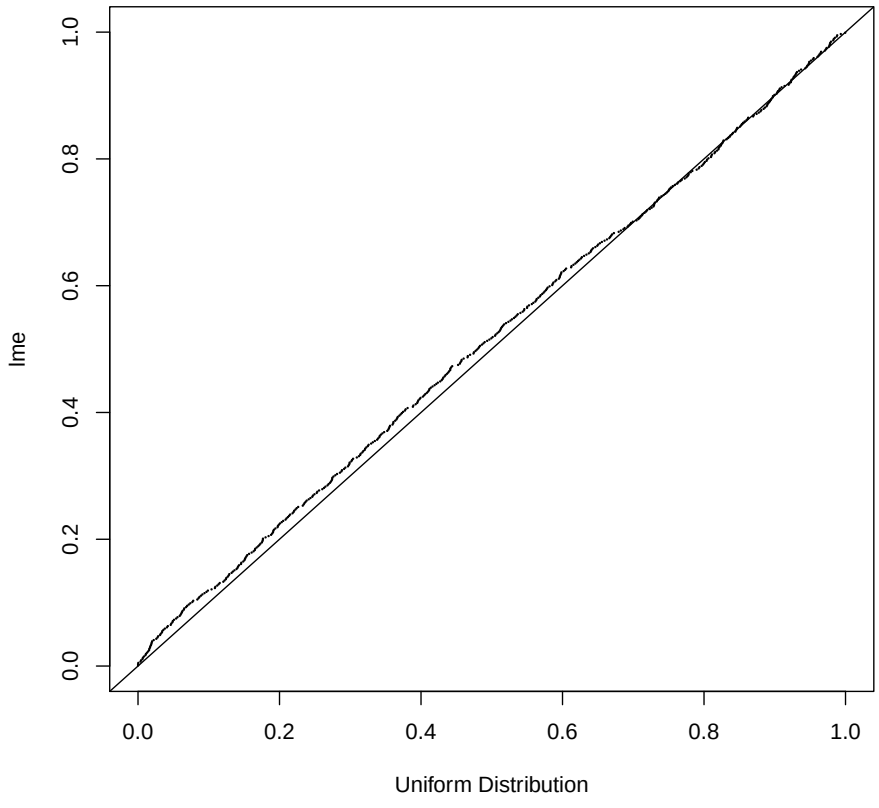
```
mean(res[2,]<0.05)
```

```
## [1] 0.072
```

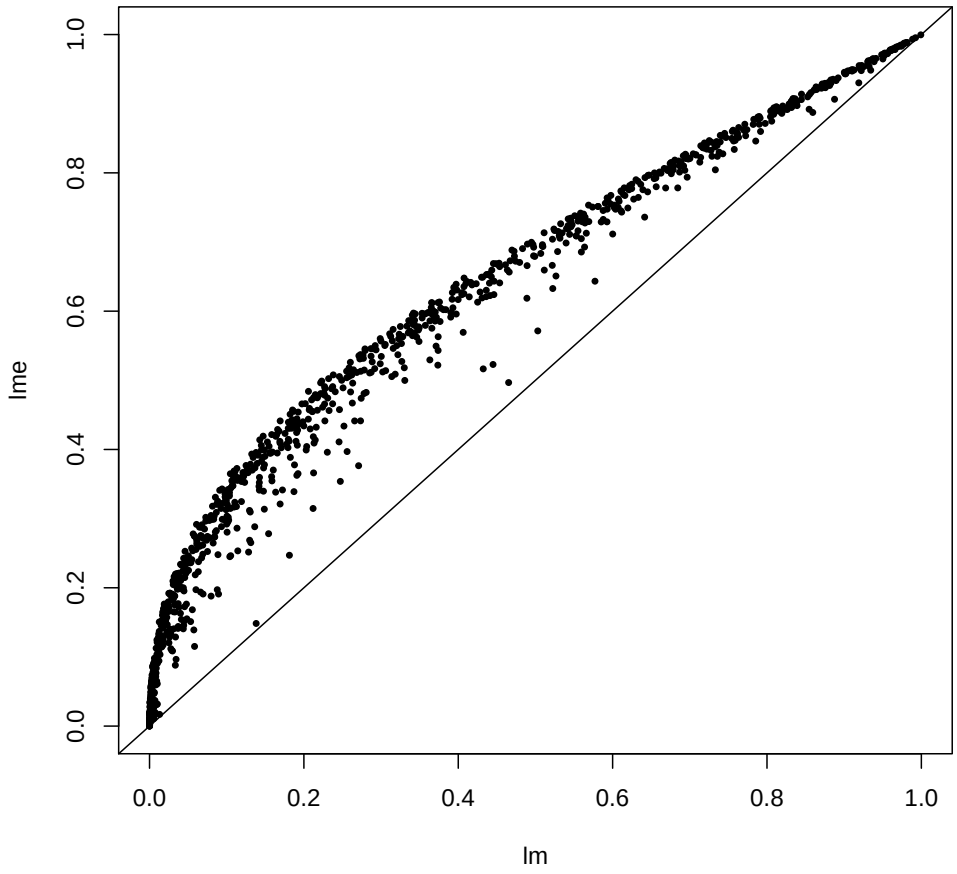
- The empirical type I error rate when not accounting for the random effect is 0.25.
- This inflated by a factor of 4.9.
- The empirical error rate when accounting for the random effect is slightly inflated
- This is due to the small sample size ($n = 6$)
- More on this later.

Example 1: Results





Example 1: Results



Example 1: Results

- Now, we repeat the simulation with a larger sample size

```
res=replicate(B3,sim.ranef(3,50,0.25,0.5,verbose=FALSE))
mean(res[1,]<0.05)
```

```
## [1] 0.215
```

```
mean(res[2,]<0.05)
```

```
## [1] 0.052
```

- The empirical type I error when not accounting for the random effect remains inflated by a factor of 4.3.
- The empirical type I error when accounting for the random effect is now right about the nominal level of 0.05

Example 2: Setup

- Now consider the two-sample problem we have previously considered with a twist
- Question: Does treatment alter the distribution of the RNA level of a given gene?
- Assumptions:
  - the RNA level for the untreated group follows a normal distribution with mean μ_0 and variance σ^2
  - The RNA level for the treated group follows a normal distribution with mean μ_1 and variance σ^2
- Sample n units from each treatments in replicates of 3
- Apply the two-sample t-test which does not account for the clustering

Example 2: Simulation

```
set.seed(2314)
# Simulate with no clustering effect (sb=0)
pval0=replicate(B3,sim.twosample.clustered(3,10,0.25,0))
# Simulate with no clustering effect (sb>0)
pval1=replicate(B3,sim.twosample.clustered(3,10,0.25,0.5))
mean(pval0<0.05)
```

```
## [1] 0.049
```

```
mean(pval1<0.05)
```

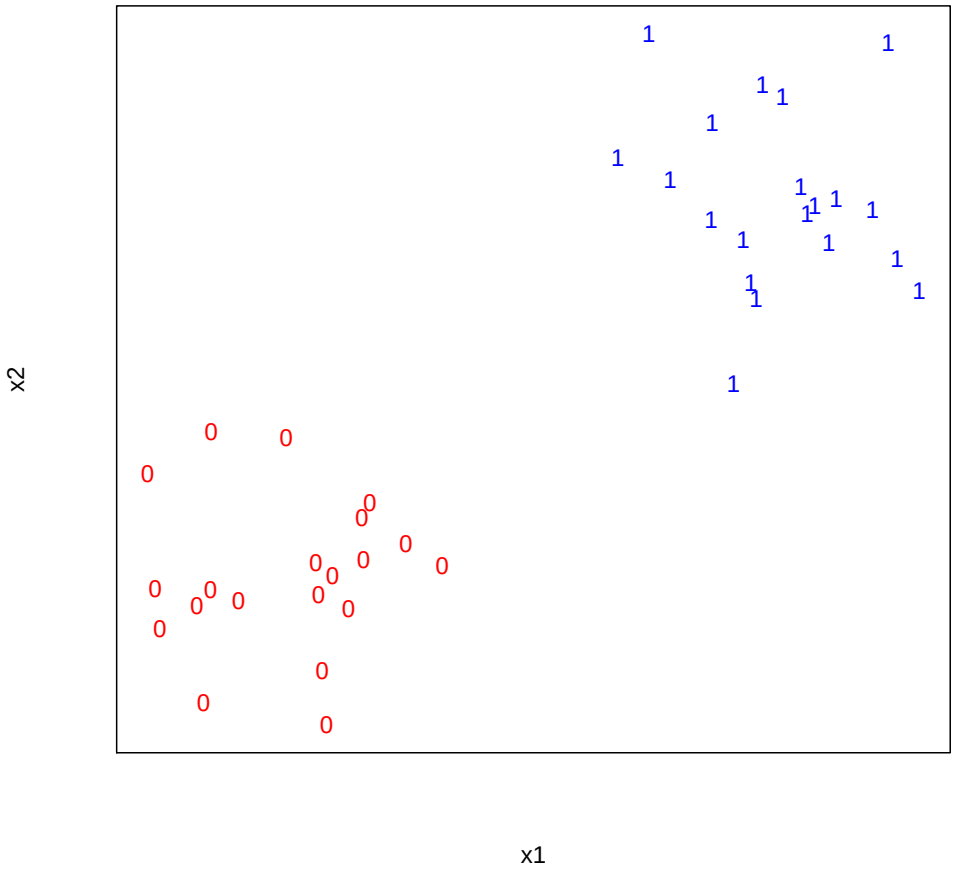
```
## [1] 0.252
```

- The empirical type I error when there is no clustering effect is 0.049
- The empirical type I error when there is a clustering effect is 0.25
- This off by a factor of 5!

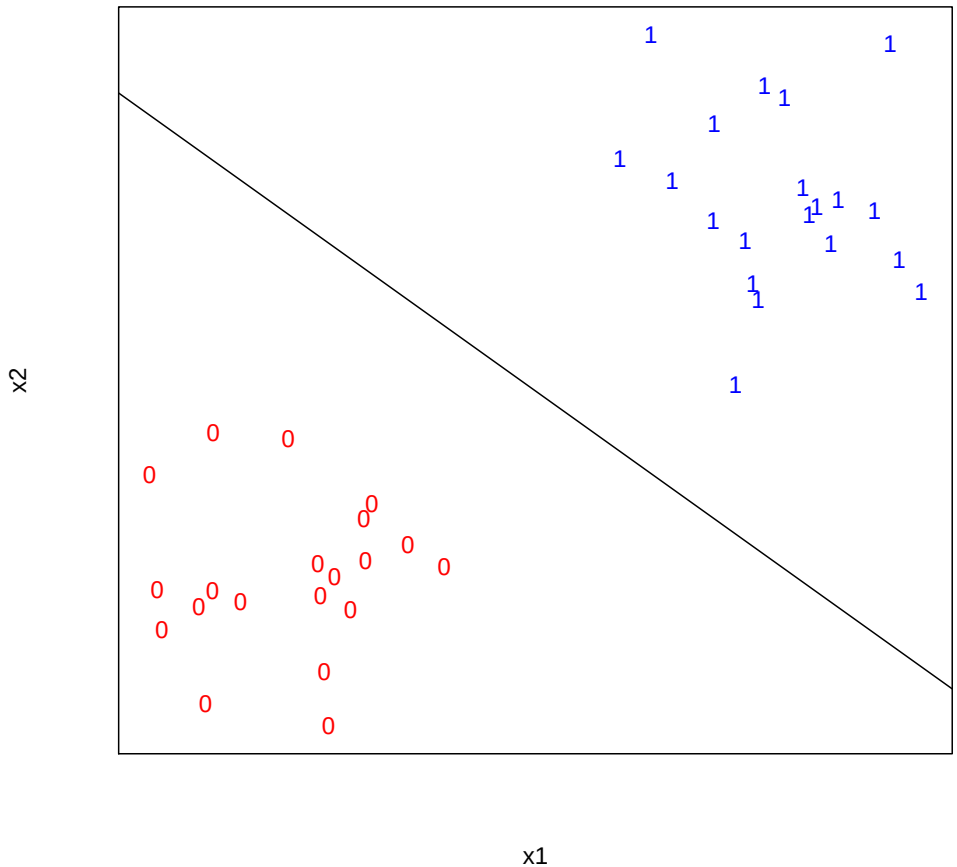
Section 4

Elements of Supervised Learning

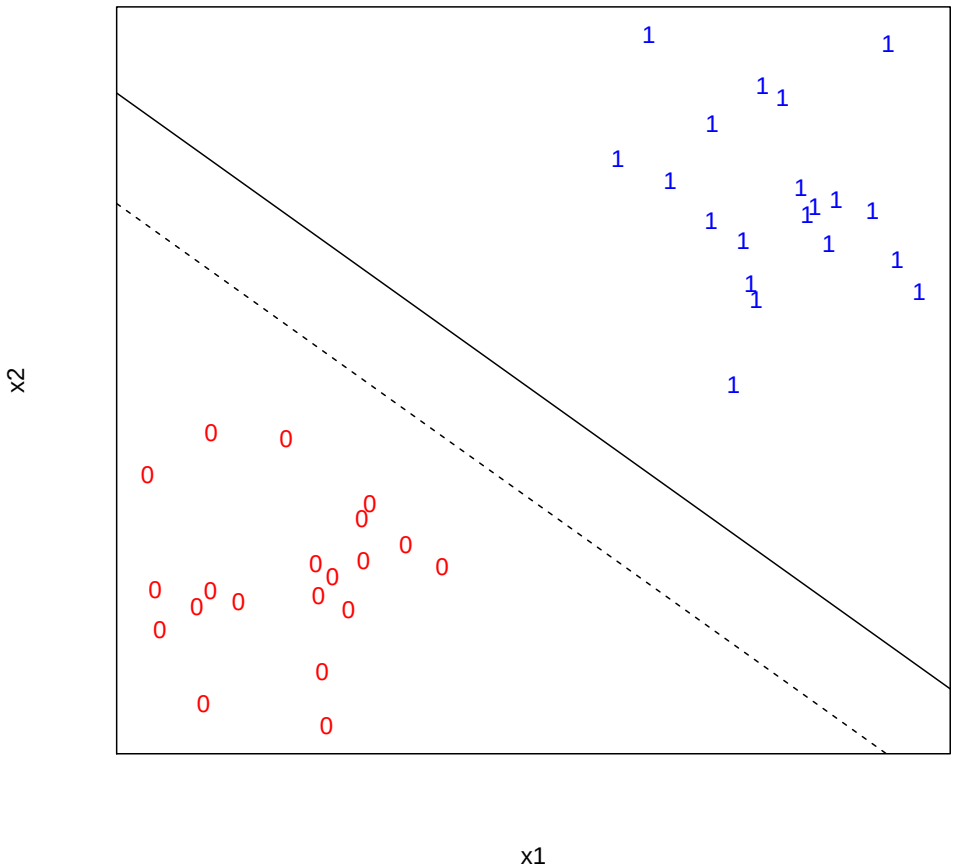
Classification Problem



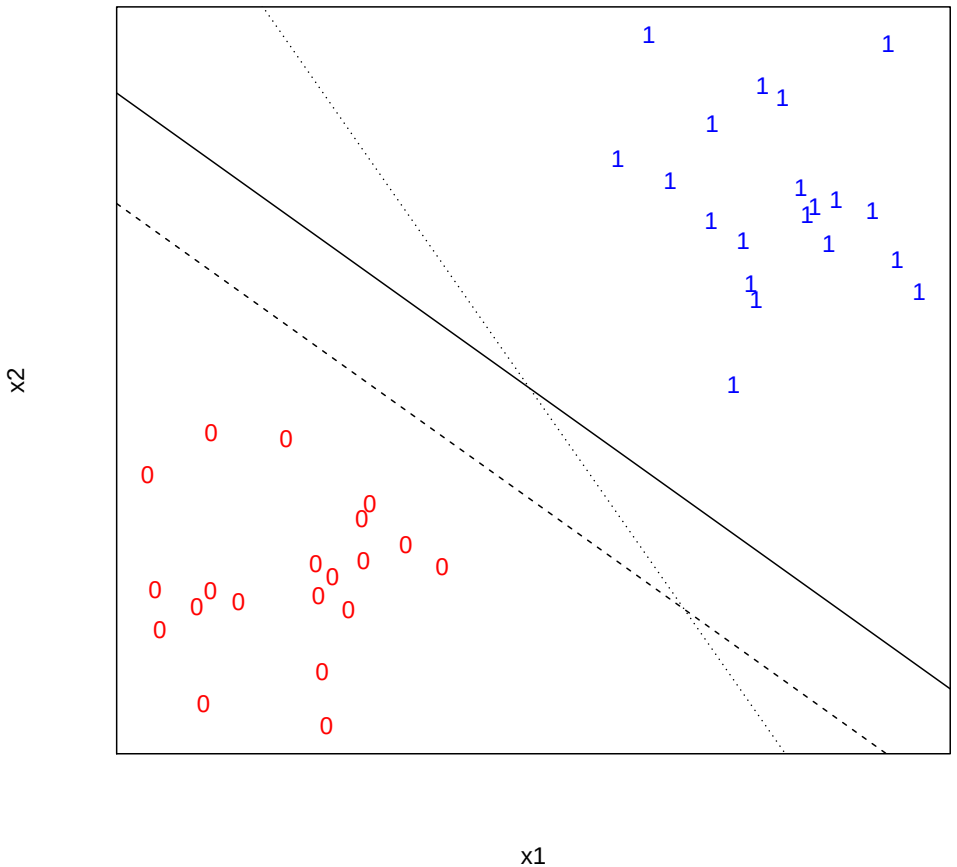
Clear-cut case



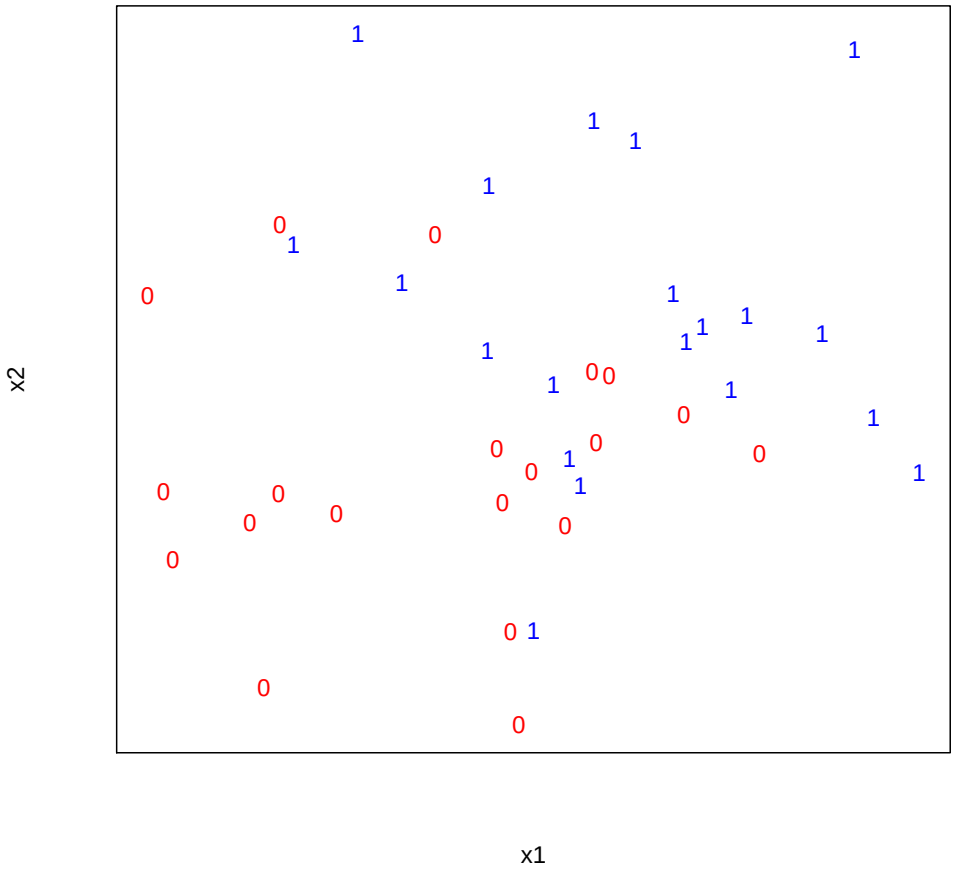
Clear-cut case?



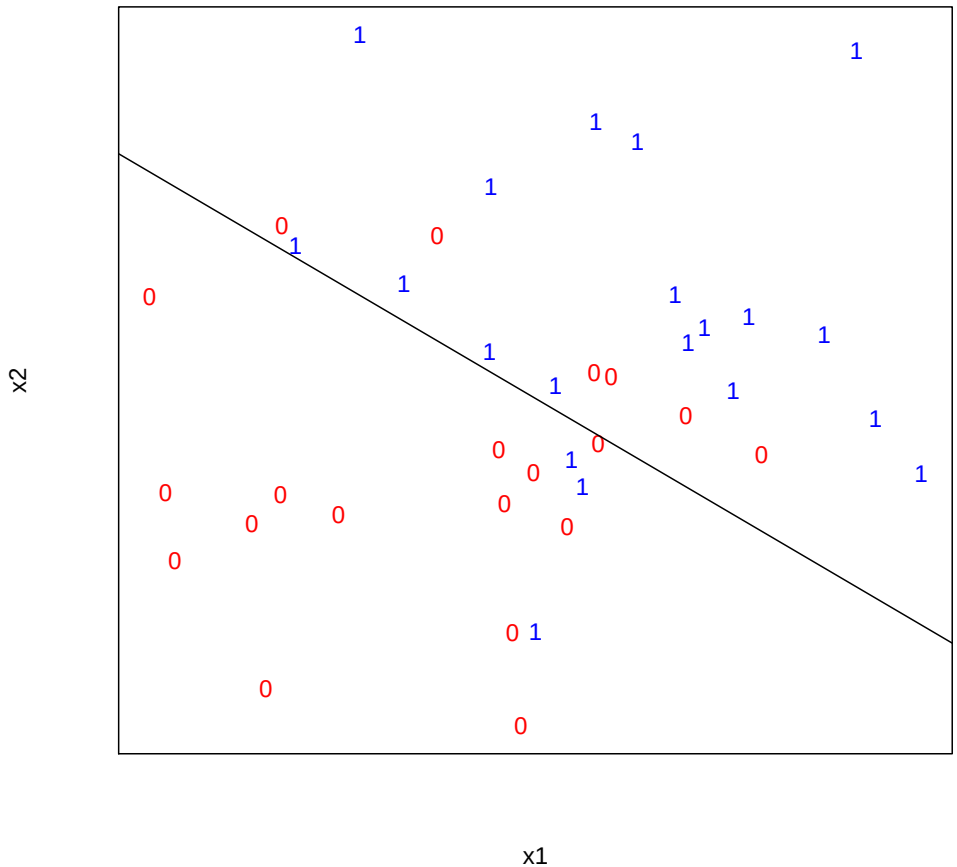
Clear-cut case??



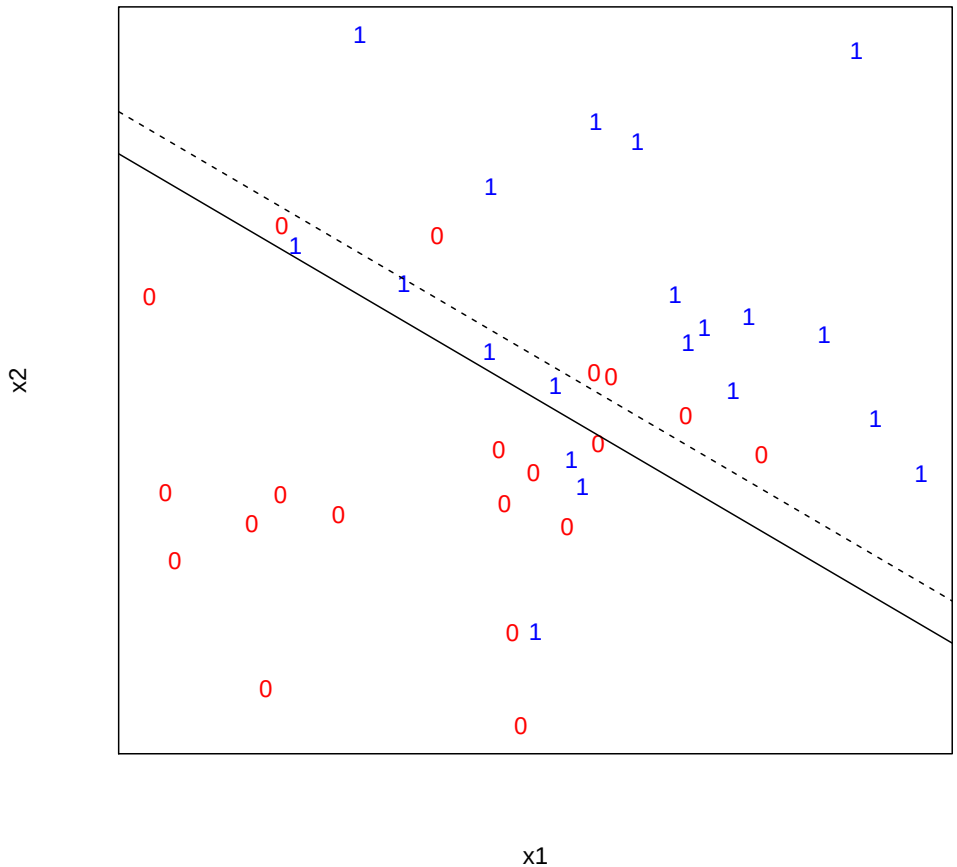
Less Clear-cut case



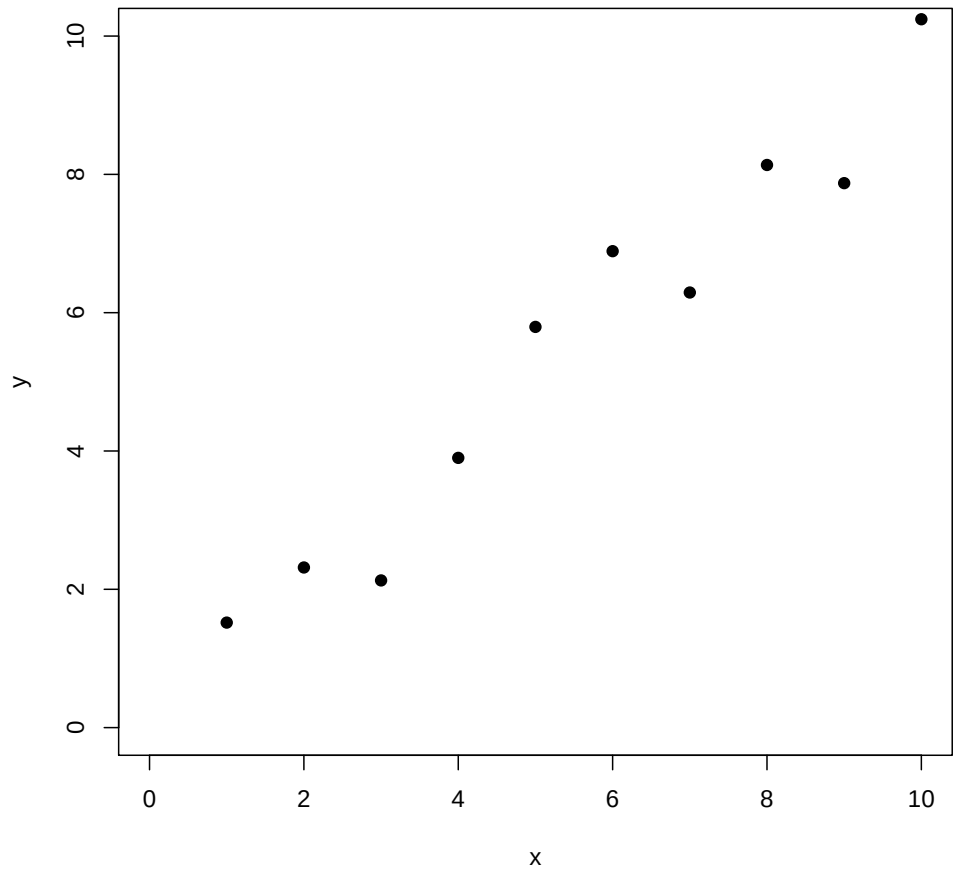
Less Clear-cut case



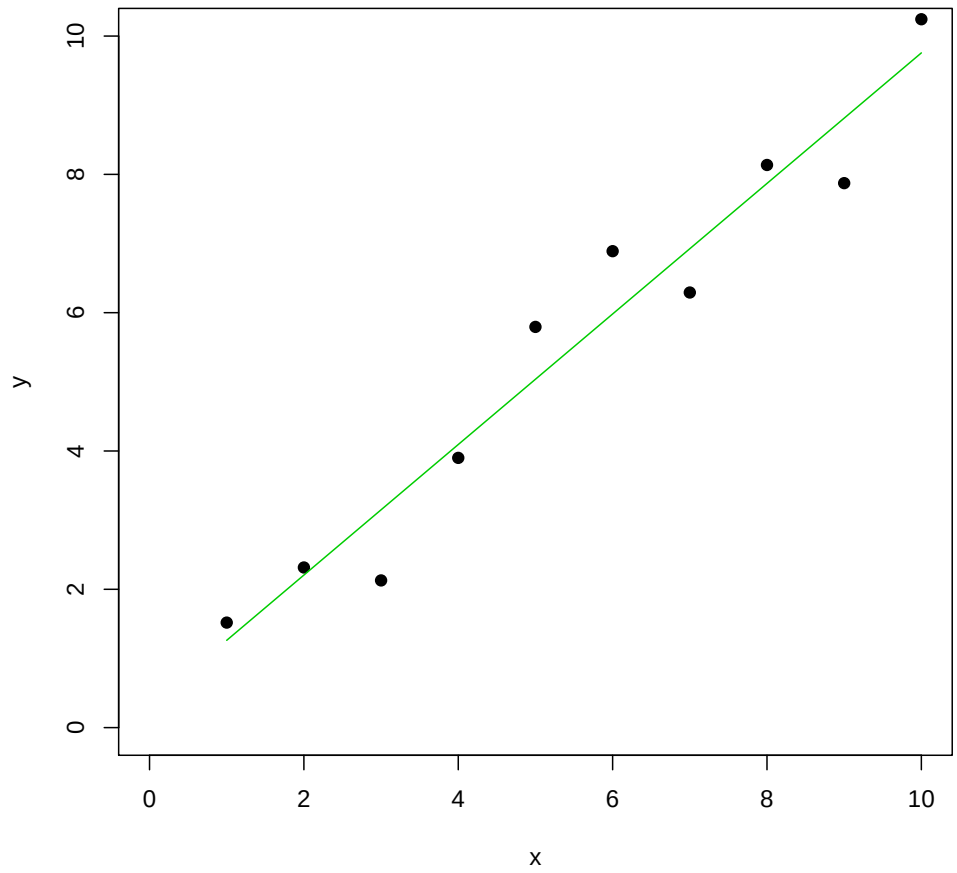
Less Clear-cut case



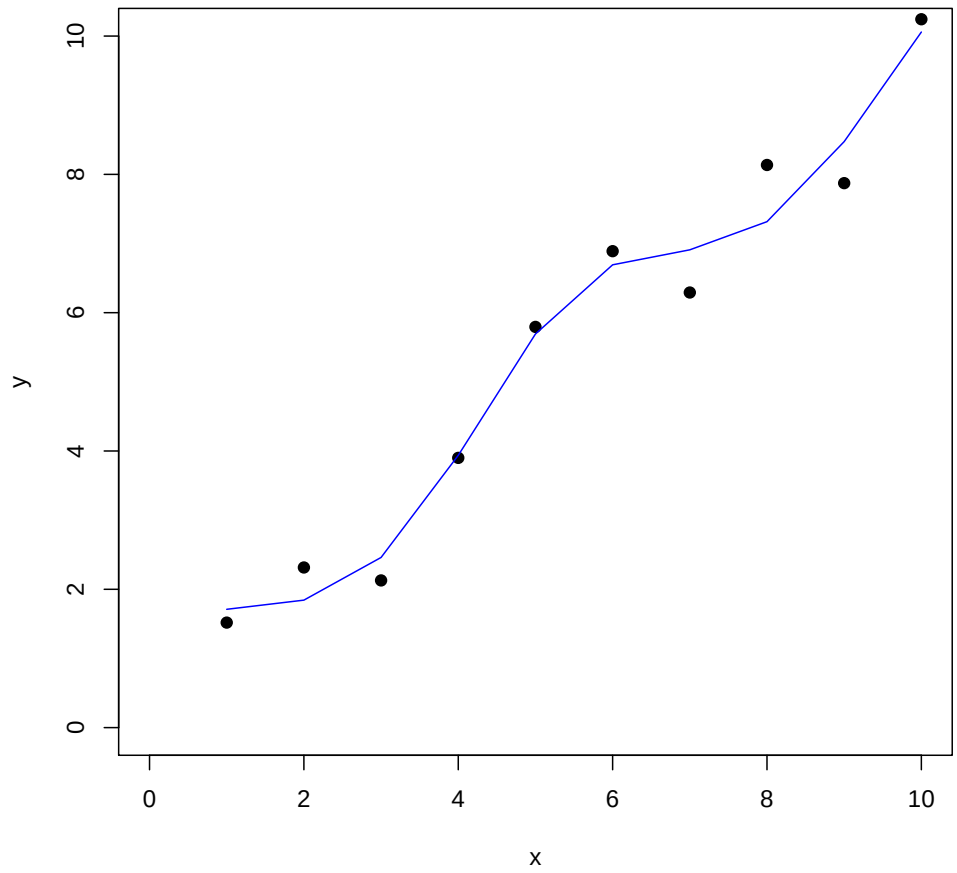
Regression Problem



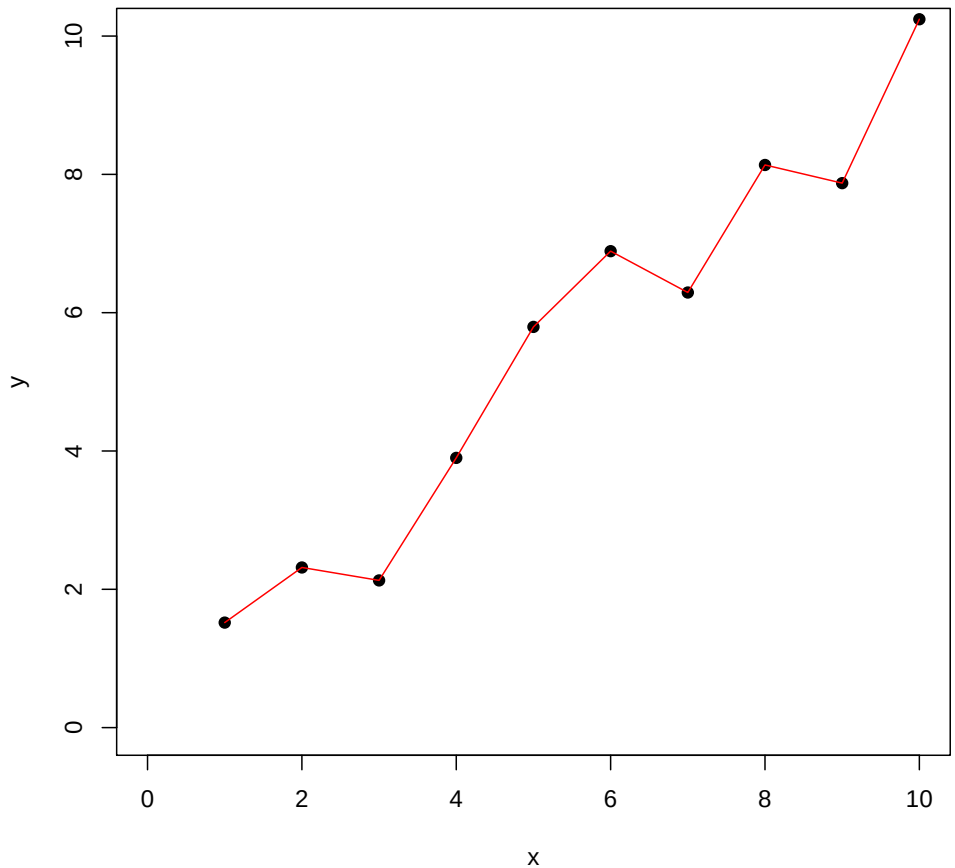
Linear Regression (lin)



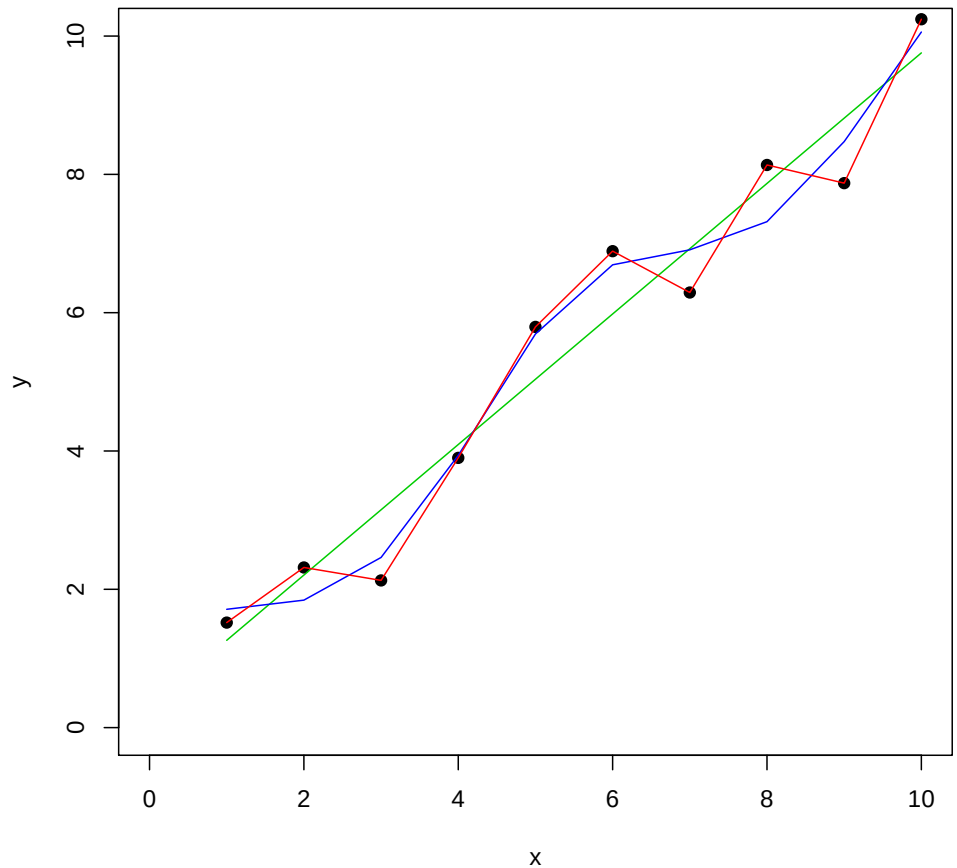
Spline Regression (spl)



Connect the dots (ctd)



Which Approach?



Supervised Learning (Classification)

- Goal: Predict a binary outcome (Y) on the basis of baseline information (X)
- Y assumes the value 0 or 1 (e.g., control vs case, or AML vs ALL)
- X could be single variable or be a vector of multiple variables
- Example: Can you predict Y on the basis of two genes say X_1 and X_2
- Note that a goal is to build a machine that will take on two values X_1 and X_2 and return a 0 or a 1
- You can denote this machine as a function $g(x_1, x_2)$

Classifier

- We will denote the predictor or classifier by $g(x)$
- $x = (x_1, x_2)$ is the vector of gene expressions for genes 1 and 2
- Based on x, the classifier g makes a prediction for the outcome
- Note that $g(x) = 0$ or $g(x) = 1$
- The prediction is *correct* if $Y = 1$ and $g(x_1, x_2) = 1$, or $Y = 0$ and $g(x_1, x_2) = 0$
- The prediction is *wrong* if $Y = 0$ and $g(x_1, x_2) = 1$, or $Y = 1$ and $g(x_1, x_2) = 0$

Prediction Assessment

	$g(x_1, x_2) = 0$	$g(x_1, x_2) = 1$
$Y = 0$	True-Negative	False-Negative
$Y = 1$	False-Negative	True-Positive

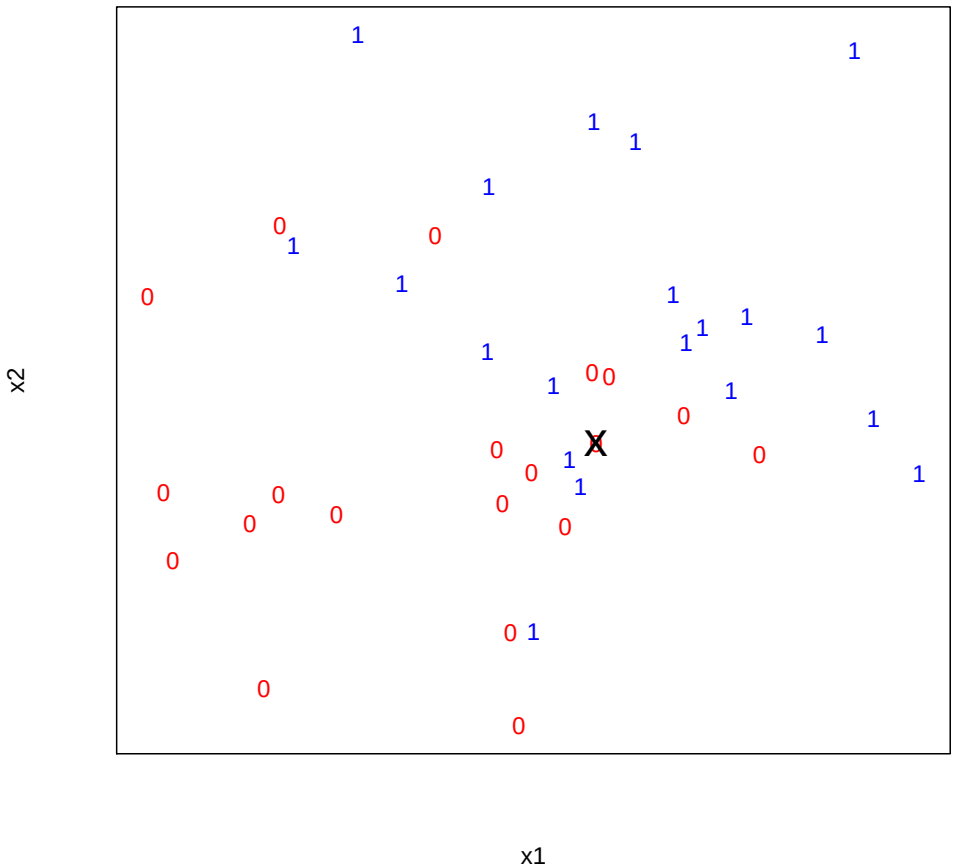
Steps to Construct a Classifier

- Collect a random data set of size n to build (train) a classifer
- This is called the training data
- On the basis of these data, construct the classifier g_n
- It is subscripted by n to emphasize that it is trained on the basis of the training data
- Note that the final performance of g_n is *not* be judged on the basis of the training data
- It is to be judged on the basis of its performance on *future* data
- Called testing data

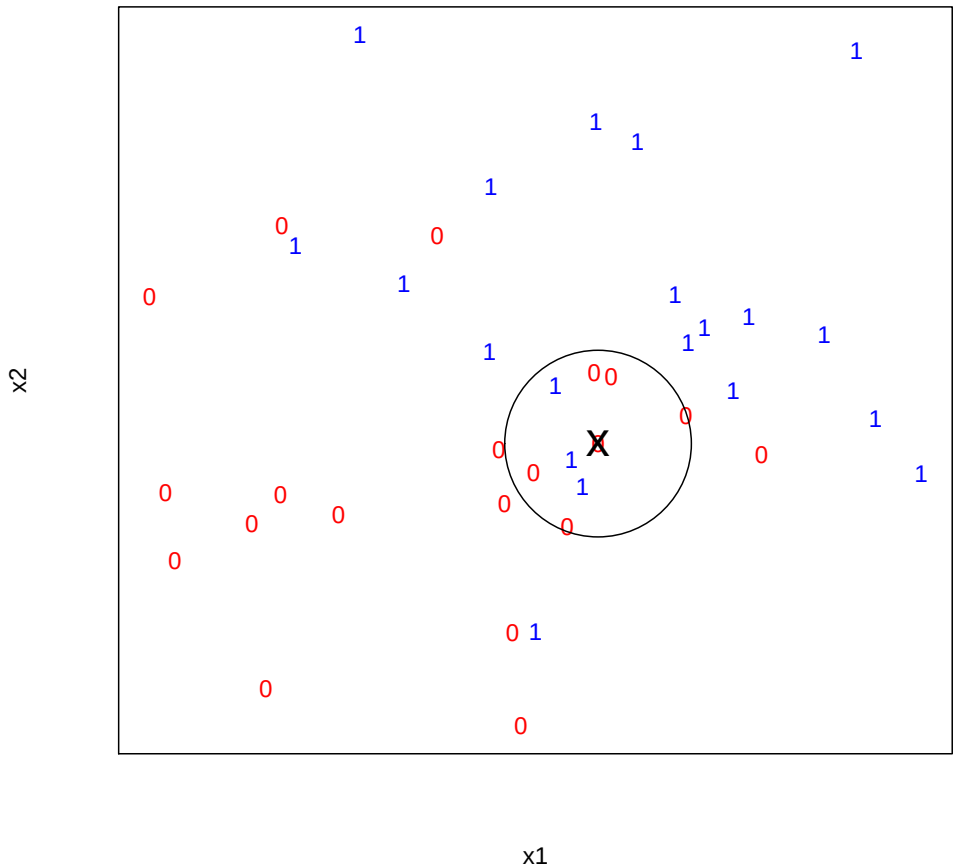
Steps in Notation

- Collect the training data $(X_1, Y_1), \ldots, (X_n, Y_n)$
- Construct a classifier g_n on the basis of the training data
- Apply g_n to a new data set $X_1^*, \ldots, X_k^*$ to get
- k predictions: $\hat{Y}_1^*, \ldots, \hat{Y}_k^*$
- Compare the predictions to the observed outcomes $Y_1^*, \ldots, Y_k^*$
- Note that at the testing stage, you are blinded to the Y_k^*

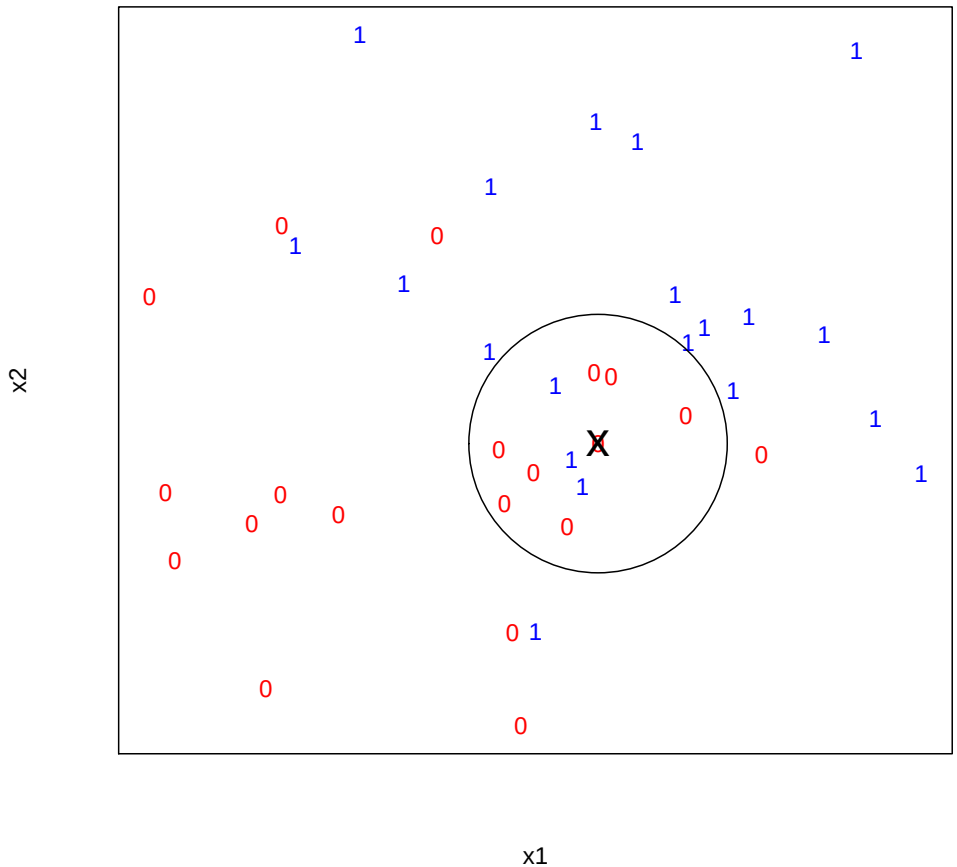
k-Nearest Neighborhood



3-Nearest Neighborhood



5-Nearest Neighborhood



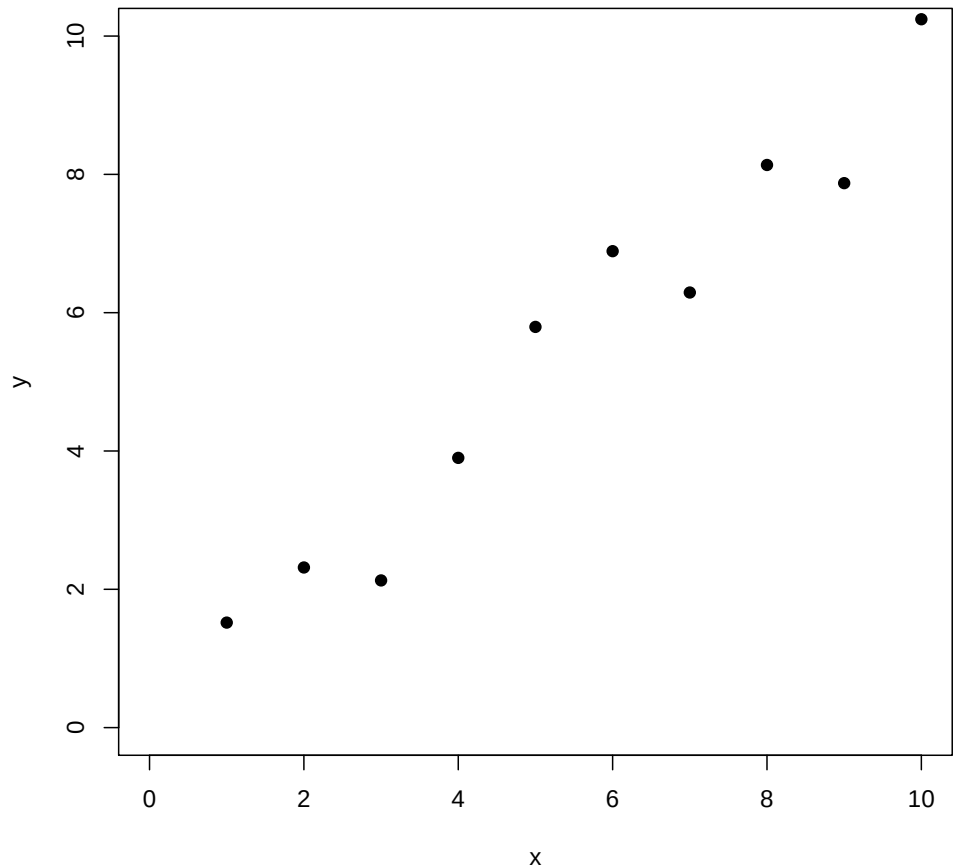
Parsimony

- The model should be parsimonious (less is more)
- Including too many noisy/unimportant features often degrades the performance of the classifier.
- Including highly dependent induces problems (e.g., multi-collinearity from simple linear regression).
- Additional complication: It is not practically/computationally feasible to include tens of thousands of features in the model.

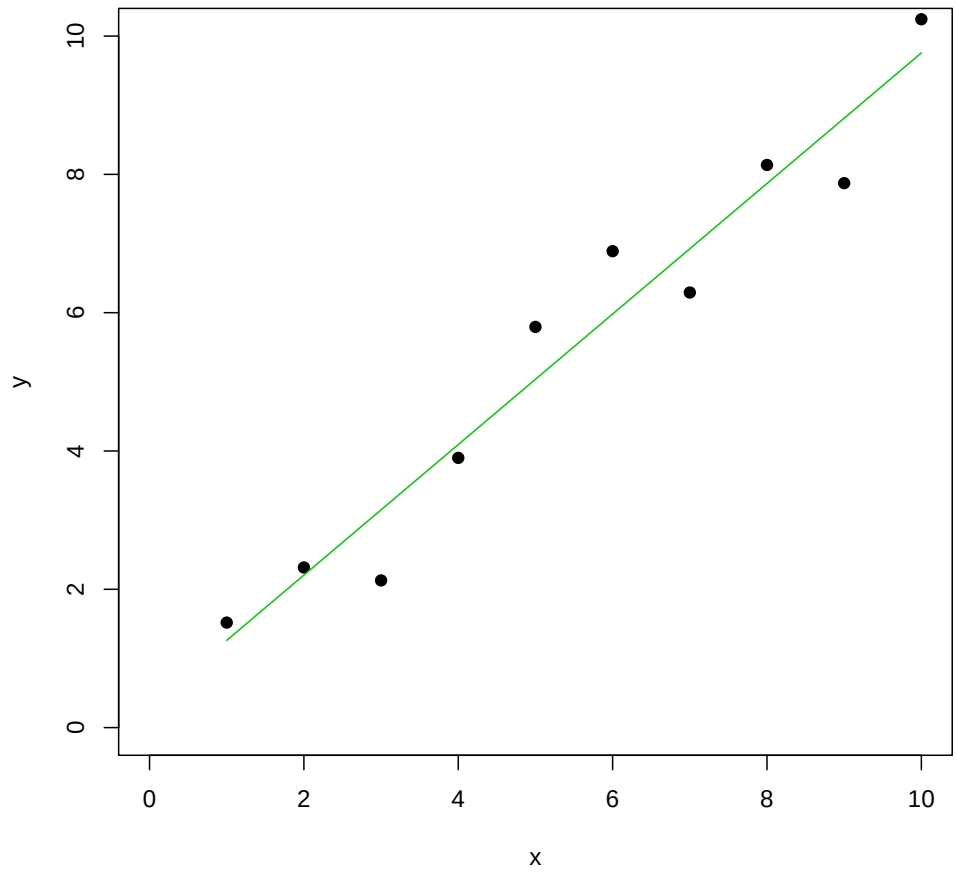
Overfitting

- Two many parameters compared to the number of data points in the training set
- A complicated model will fit the training set well
- It will however perform poorly for an independent set.

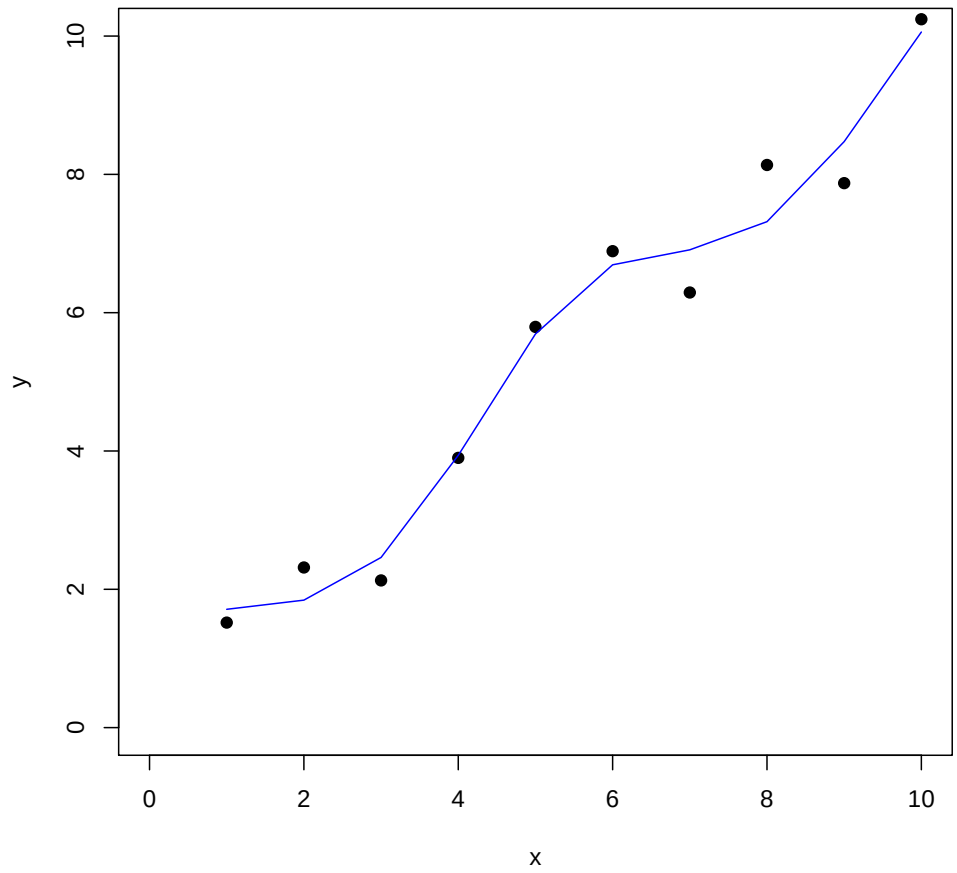
Overfitting



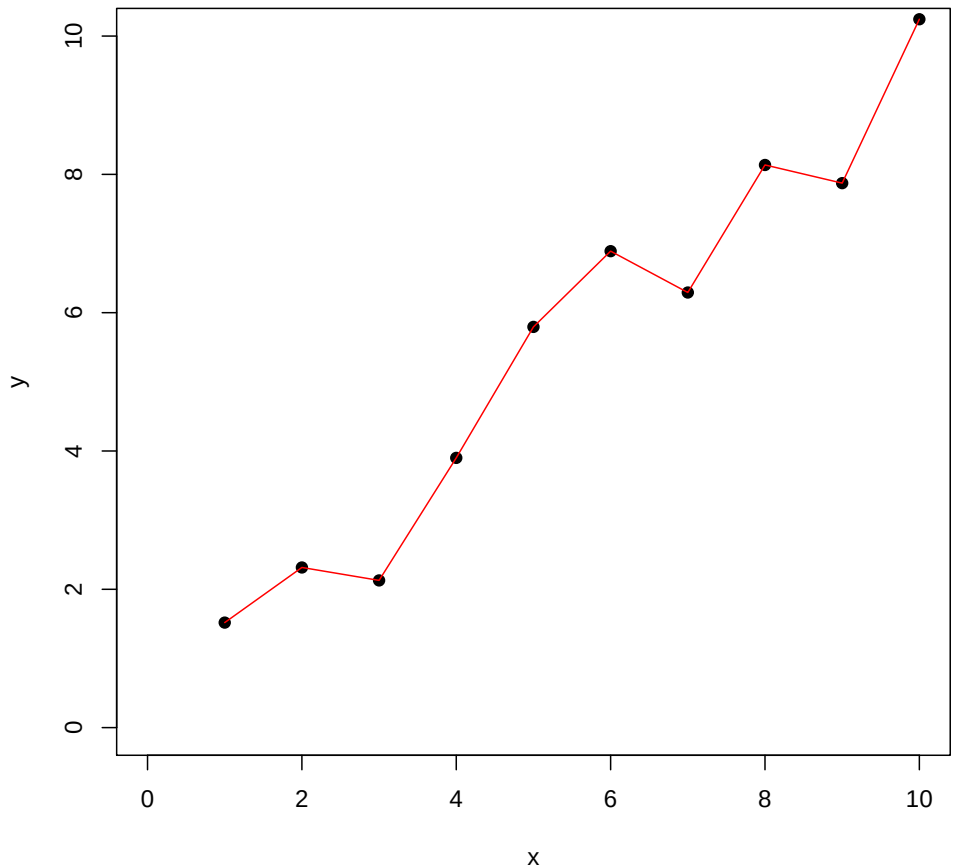
Linear Regression (lin)



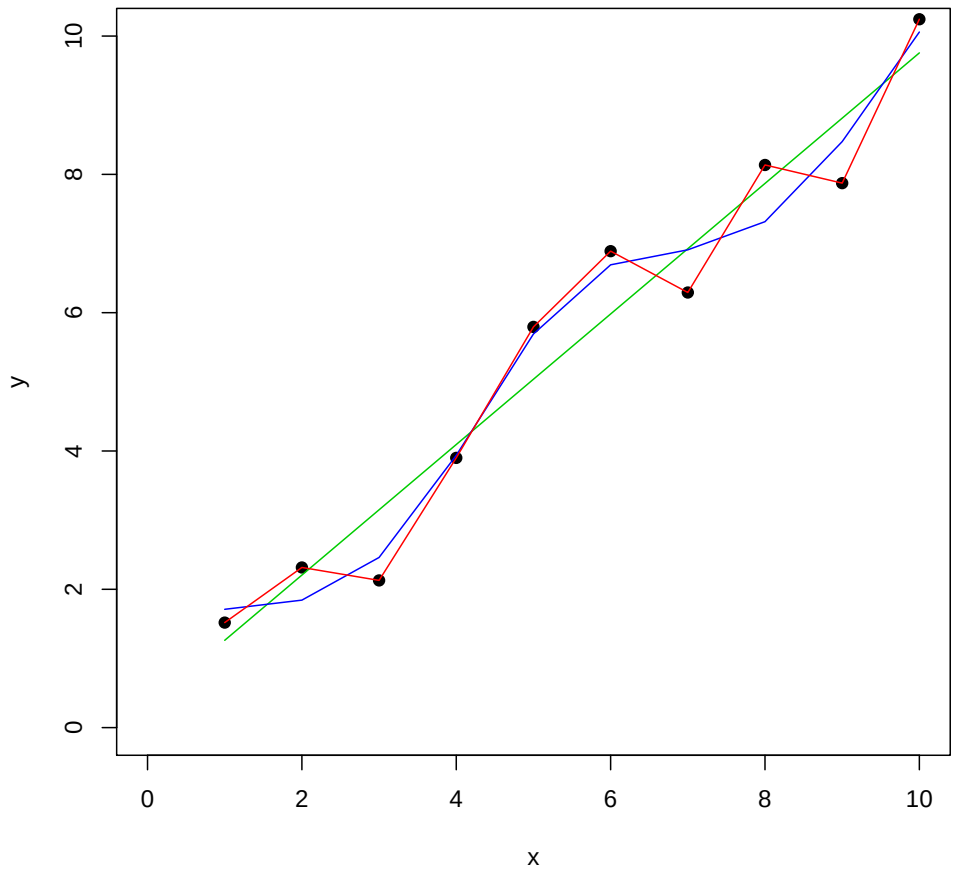
Spline Regression (spl)



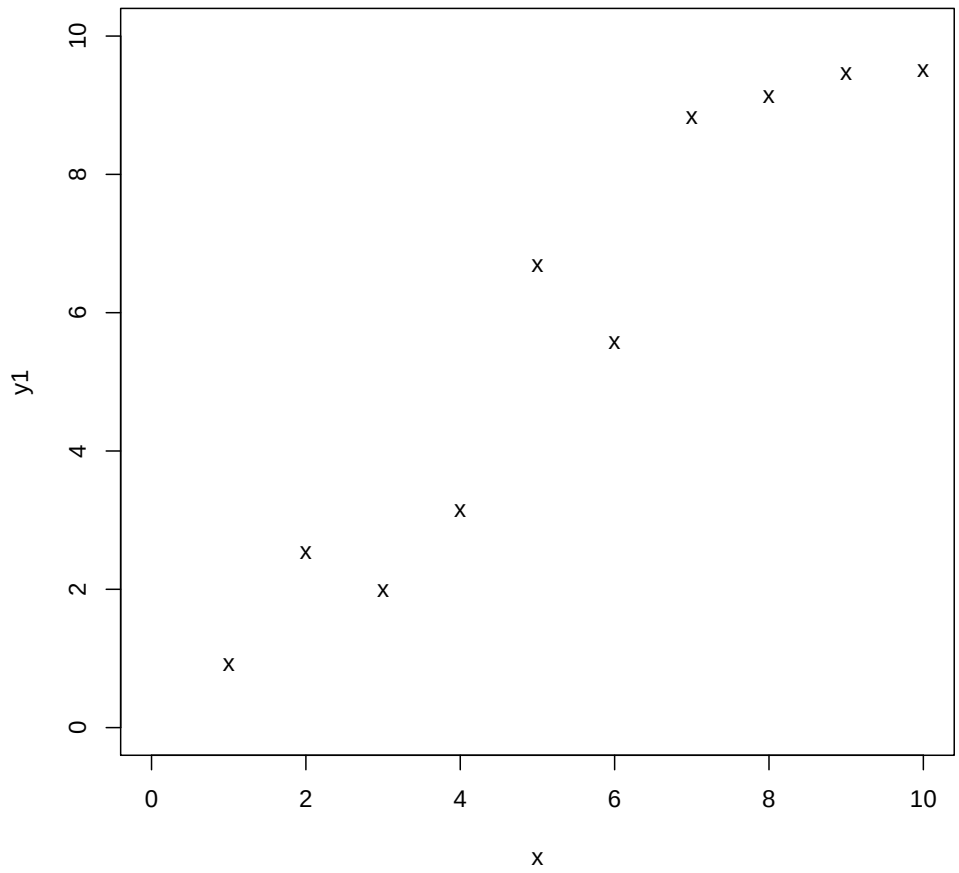
Connect the dots (ctd)



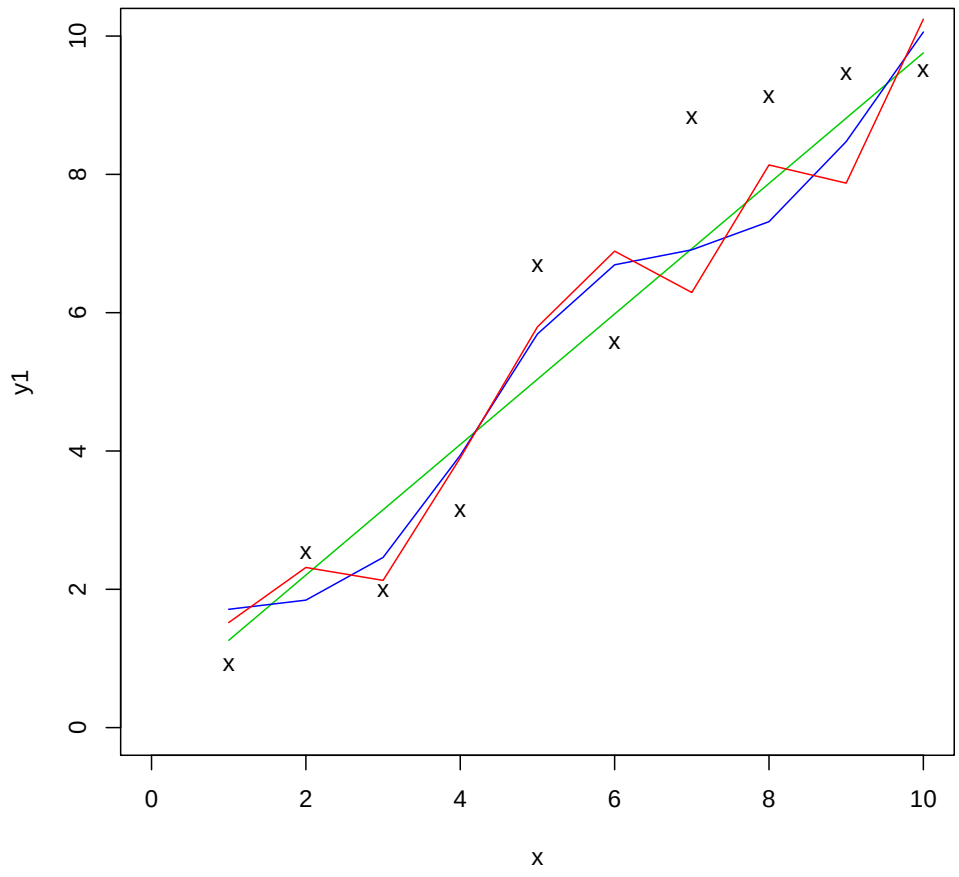
RSS: 4.1 (lin) vs 1.9 (spl) vs 0 (ctd)



New Data Set



RSS: 11 (lin) vs 12.4 (spl) vs 14 (ctd)



Two Challenges in Building a Classifier

1. Feature Selection:
   - It is neither feasible nor provident to build a classifier based on all available variables
   - A subset of the variables has to be selected to build the model
   - This is also called feature extraction
2. Tuning Parameter Selection:
   - Statistical methods may have one or more parameters that have to be set
   - For example when using k-NN, one has to decide what k should be (e.g., 1, 3 or 5 or how about 8)?
   - Choosing the defaults set by the software is inappropriate
   - The feature selection method could also have tuning parameters that have to be set (e.g., the number of features to be selected)
   - The performance of the method could be highly sensitive to the choice of these parameters

Feature Selection

- Reasonable Feature Selection is *critical* if not the most important component of model building.
- You cannot expect to build a good model if you select poor features.
- This is also called Feature Extraction
- We will talk about a few approaches that have been used in the literature.

Feature Selection (ranked based on test-statistic)

- Compute the two-sample t-test for all m features (based on the training set)
- Identify the top say 10 or 15 features (e.g, ranked based on the absolute value of the test statistic).
- Build a model on these "top" features (based on the training set)
- Alternatively, you could select all features for which the P-value is less than a certain threshold (say 0.001).
- You can also use the Wilcoxon rank sum statistic to protect against choosing features with outliers.

Feature Selection (Ordination Methods)

- A standard approach for reducing the dimension in the microarray setting is the method of Principal Components (PCs)
- The PCs are combinations of the original variables (gene expressions) that have maximum variability
- The are also constructed as to be uncorrelated with another
- This attempts to address the issue of high dimension and multi-collinearity simultaneously.
- One can use the principal components (say the first two or three) as the features
- Alternatively, one can first reduce the dimension by using the two-sample test-statistic approach and then get the PCs

Tuning

- You cannot expect to be able to build a model using default values provided by the software package.
- If you use k-NN you need to decide which k (e.g., 3 or 5 or 7) you want to use
- If you use the simple feature selection method you need to determine how many "top" features you want to use
- If you are doing PC dimension reduction, you need to determine how many PCs you want to use.
- In some books and articles, "tuning" only refers to the choice of the model parameter (e.g., k in k-NN)
- Must take a broader perspective as the choices in the FS part also affect the results.

Validation

- Split the data into a training and a mutually exclusive testing set
- Build the model (including feature selection, tuning) on the *training* set
- Evaluate the performance of the model on the *testing set*
- IMPORTANT: The model is built based on the *training* set. The *testing* set should not contribute *any* information.
- Violating this principle will invariably result in bias

Error Substitution Validation

- Error Substitution Validation: The testing set is empty.
- Test the model you just built on the *training* set
- This approach cannot be recommended under any circumstance.
- Analogy: Assess the fit of the linear model by plotting the fitted (from the data) to the observed data.
- A bona-fide testing set is required.
- Will demonstrate how this can lead to noise discovery

Hold-out Method

- Split the data into two parts
- Keep the testing set locked up
- Better yet, ask an "honest" broker to keep it from you until you are ready to test the model
- This approach is reasonable if you have a large number of cases
- It may be problematic if the outcomes are sparse

k-fold Cross-Validation

- Many microarray experiments are from smaller (e.g., pilot) studies
- It is not impossible to get reasonably size training and testing sets this cases
- A reasonable approach to get around this is k-fold cross-validation (CV)
- Randomly split cases into k (nearly) equally sized subsets (folds).
- At each step take of these k portions as the *testing* set and construct the *training* set based on the other $k - 1$ portions
- Special case is Leave-One-Out CV (LOOCV) where $k = n$
- For really small data sets, LOOCV is often the best (most practical) choice.

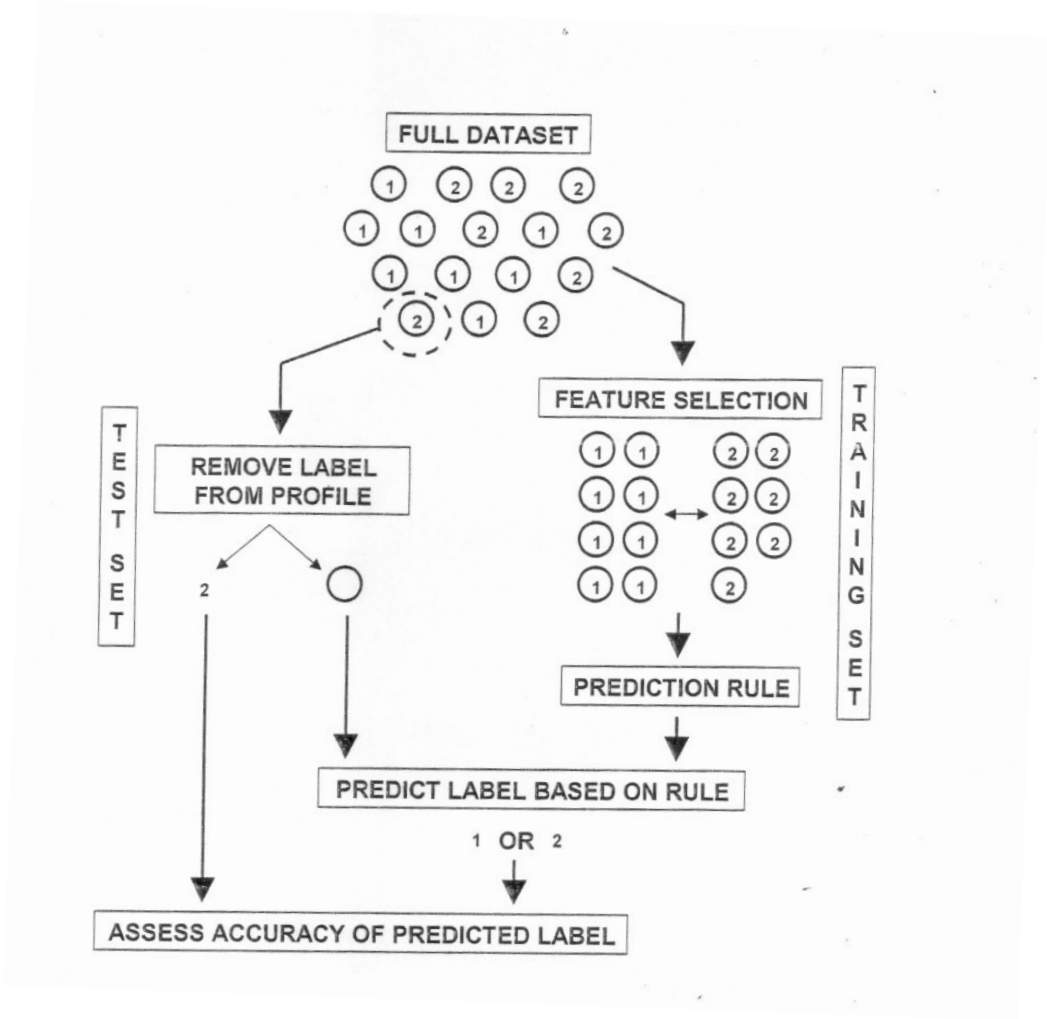
Naive Cross-Validation

- Naive Validation: Do the feature selection once based on all n cases
- In each CV step use the same set of features.
- This will invariably make the results look better than they really are
- It should be avoided unless one feels *very* certain about the features (say biologically relevant gathered *a priori*

Proper Cross-Validation

- Choose the first fold and set it aside the other $k-1$ folds
- Carry out Feature Selection on the other $k-1$ folds
- Train the model based the top features on the $k-1$ folds
- Test the model on the first fold left out
- Repeat the above for the second fold (set aside the second fold, leave in the first and the next $k-2$ folds).

Important Illustration (Fig 8.5) from Simon et al.



Simulate Data for *k*-NN Prediction

- Simulate expression from 1000 genes for 40 patients. Let the first 20 be responders and the remaining 20 be non-responders

```
set.seed(123)
n=20
m=1000
EXPRS=matrix(rnorm(2*n*m),2*n,m)
rownames(EXPRS)=paste("pt",1:(2*n),sep="")
colnames(EXPRS)=paste("g",1:m,sep="")
grp=rep(0:1,c(n,n))
```

- Pick the top 10 features based on the two-sample *t*-test

```
library(genefilter)
stats=abs(rowttests(t(EXPRS), factor(grp))$statistic)
ii=order(-stats)
```

- Filter out all genes except the top 10

```
TOPEXPRS=EXPRS[, ii[1:10]]
```

Error Resubstitution and Naive CV

- Error resubstitution (Training and Testing set are the same)

```
mod0=knn(train=TOPEXPRS,test=TOPEXPRS,cl=grp,k=3)
table(mod0,grp)

##      grp
## mod0  0  1
##    0 17  0
##    1  3 20
```

- Cross-validated predictions (the features selection is not part of the CV process)

```
mod1=knn.cv(TOPEXPRS,grp,k=3)
table(mod1,grp)

##      grp
## mod1  0  1
##    0 16  0
##    1  4 20
```

- Note that in both examples, `TOPEXPR` not `EXPR` is used.

R Function to Implement Proper CV based on k-NN

```
top.features=function(EXP,resp,test,fsnum)
  {
    top.features.i=function(i,EXP,resp,test,fsnum)
      {
        stats=abs(mt.teststat(EXP[,-i],resp[-i],test=test))
        ii=order(-stats)[1:fsnum]
        rownames(EXP)[ii]
      }
    sapply(1:ncol(EXP),top.features.i,EXP=EXP,resp=resp,test=test,fsnum=fsnum)
  }


# This function evaluates the knn

knn.loocv=function(EXP,resp,test,k,fsnum,tabulate=FALSE,permute=FALSE)
  {
    if(permute)
      resp=sample(resp)
    topfeat=top.features(EXP,resp,test,fsnum)
    pids=rownames(EXP)
    EXP=t(EXP)
    colnames(EXP)=as.character(pids)
    knn.loocv.i=function(i,EXP,resp,k,topfeat)
      {
        ii=topfeat[,i]
        mod=knn(train=EXP[-i,ii],test=EXP[i,ii],cl=resp[-i],k=k)[1]
      }
    out=sapply(1:nrow(EXP),knn.loocv.i,EXP=EXP,resp=resp,k=k,topfeat=topfeat)
    if(tabulate)
      out=ftable(pred=out,obs=resp)
    return(out)
  }
```

Proper Cross-Validation

- Finally, we conduct proper cross-validation using the previous `R` function
- At each iteration, the top 10 features are selected based on the data from the $n - 1$ samples in the training set

```
knn.loocv(t(EXPRS),as.integer(grp),"t.equalvar",3,10,TRUE)

##      obs  0  1
## pred
## 0      7  7
## 1     13 13
```

- Note that `EXPRS` not `TOPEXPR` is used.
- The classification rate is 50% (as expected)

Naive LOOCV: Quantitative trait

- Repeat the last experiment with a noisy quantitative outcome
- First simulate a data matrix of dimension $n = 50$ (patients) and m (genes)
- Next draw the outcome for $n = 50$ patients from a standard normal distribution independent of the data matrix
- There is no relationship between the expressions and the outcome (by design)
- We consider $m = 45$ and $m = 50000$
- We conduct Naive LOOCV using the top 10 features

Naive LOOCV: Quantitative trait

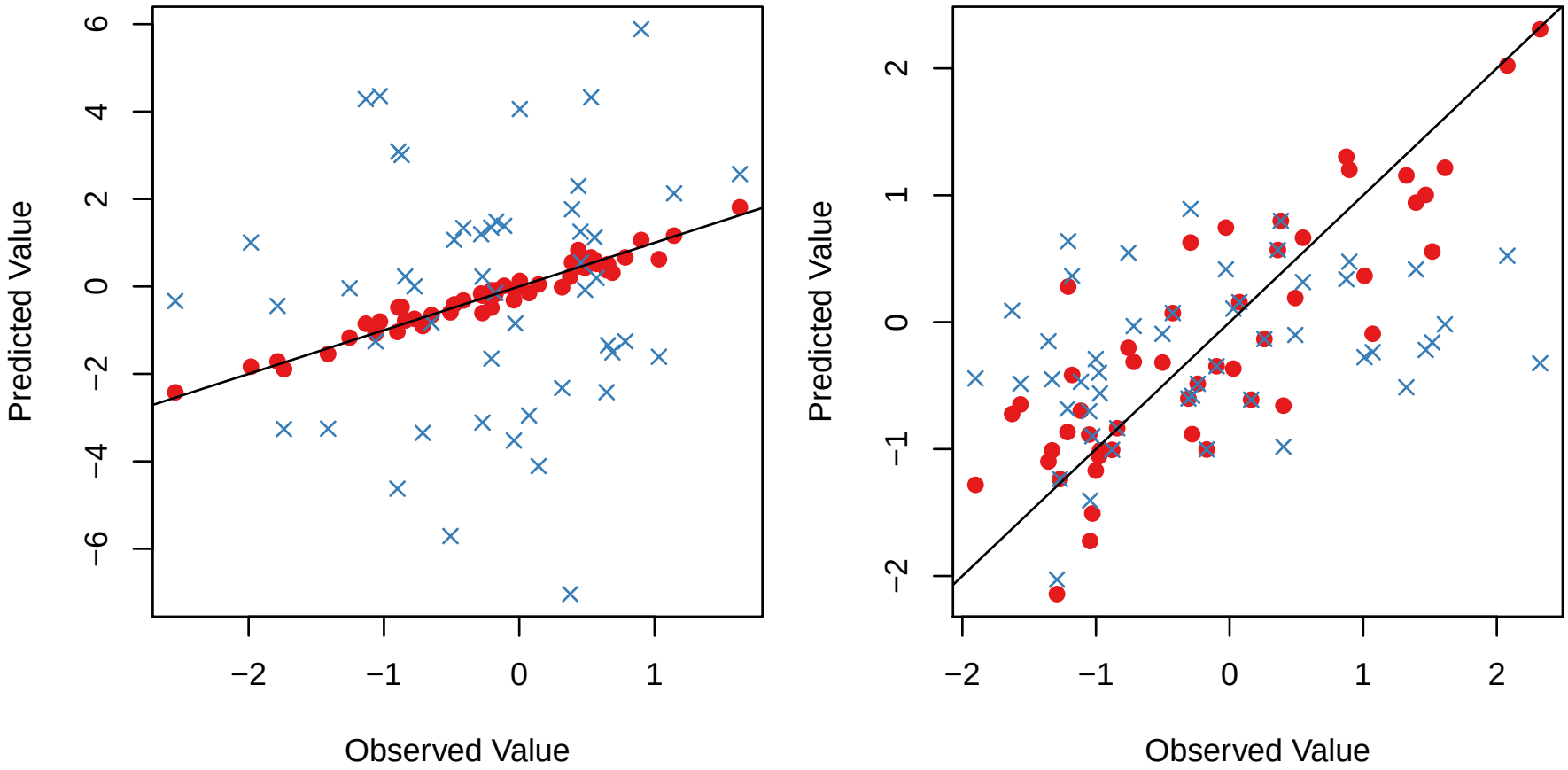


Figure taken from Owzar *et al*; *Clin Transl Sci* 2011.

Training, Validation and Testing Approach

- Before you test the model, you must freeze it
- You may want to split the Training set further into a Training and Validation set
- Use the Validation set to "tune" the model.

Final Remarks

- It is OK to try different methods (other classifiers, feature selection or tuning methods)
- Keep track of what you have done and report it (brief description in the paper and details in supplementary material)
- Be careful if you have too few responders
- You could have a model that will classify most patients as a non-responder.
- In this case a 00 ($Y = 0$ and $g(X) = 0$) may not be bona-fide true-negative
- The gold-standard for model validation, is to follow up the cross-validatiion by permutation resampling
- The `R` function provided can be used for this purpose

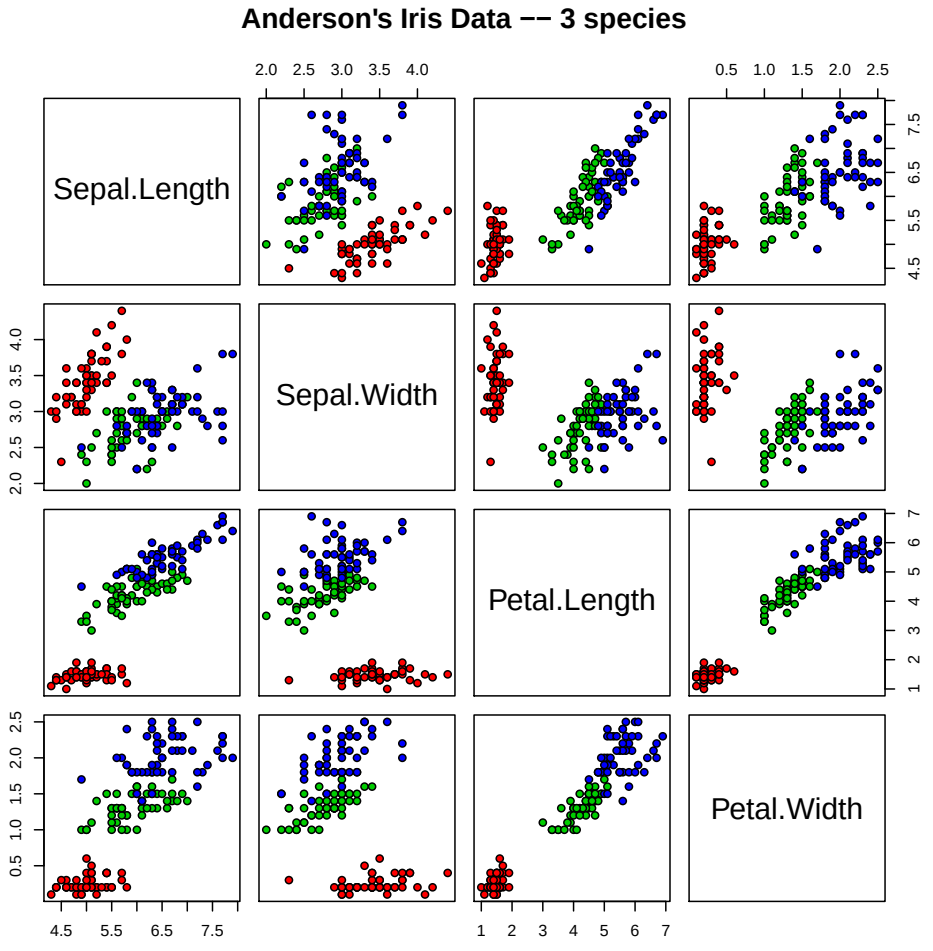
Section 5

Elements of Unsupervised Learning

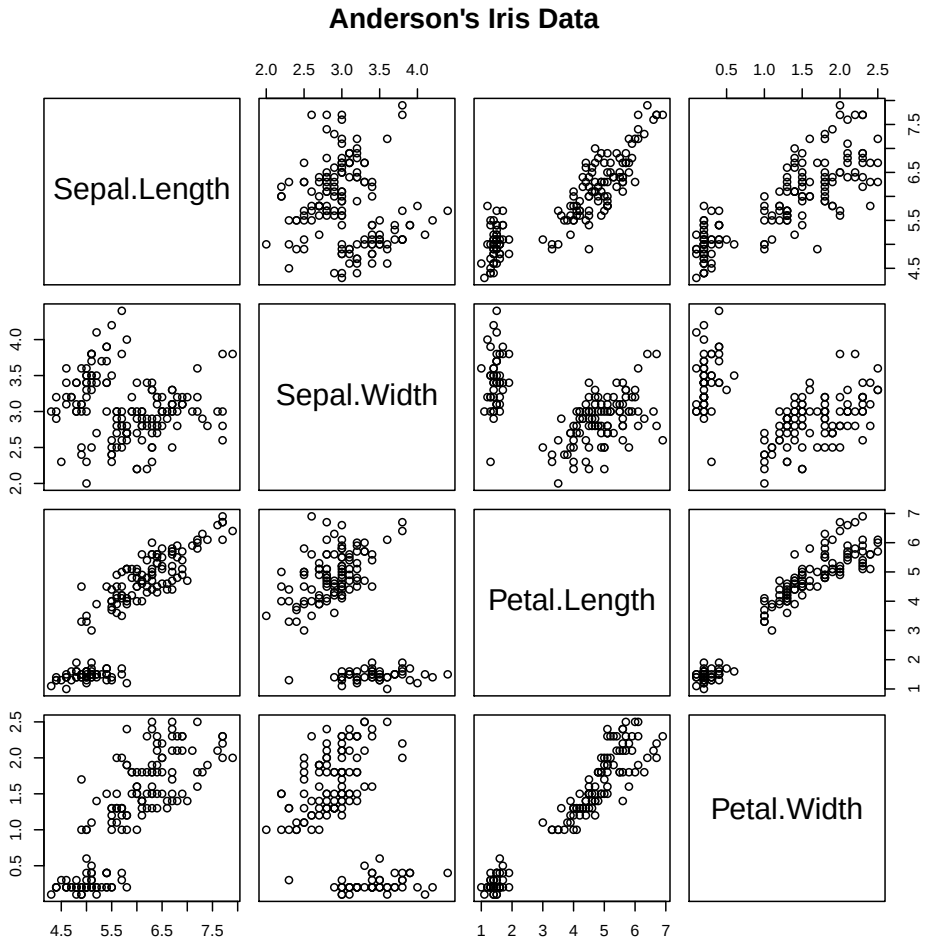
Scope

- Often we would like to discover clusters or outliers based on the gene expression profiles
- These are *unsupervised* methods in the sense that the algorithm knows nothing about the outcome
- It is only aware of the gene profiles (X) and not the outcome Y

Fisher's Iris Data



Fisher's Iris Data



A Self-fulfilling Prophecy

- Statistical method for unsupervised learning guarantee one thing
- They will return a clustering of your data
- What they do not guarantee and are invariably unable to verify, is the biological relevance or reproducibility of the clustering
- In light of this Self-fulfilling Prophecy, these methods should be used with utmost care

Golub *et al* Leukemia Data

- 47 patients with acute lymphoblastic leukemia (ALL)
- 25 patients with acute myeloid leukemia (AML)
- Platform: Affymetrix Hgu6800
- 7129 probe sets
- Golub *et al.* (1999). Molecular classification of cancer: class discovery and class prediction by gene expression monitoring, Science, Vol. 286:531-537.

Chiaretti *et al* ALL Data

- 128 patients with acute lymphoblastic leukemia (ALL)
- Platform: Affymetrix hgu95av2
- 12625 probe sets
- Chiaretti *et al.* Gene expression profile of adult T-cell acute lymphocytic leukemia identifies distinct subsets of patients with different response to therapy and survival. *Blood*, 1 April 2004, Vol. 103, No. 7.

Methods to be Discussed

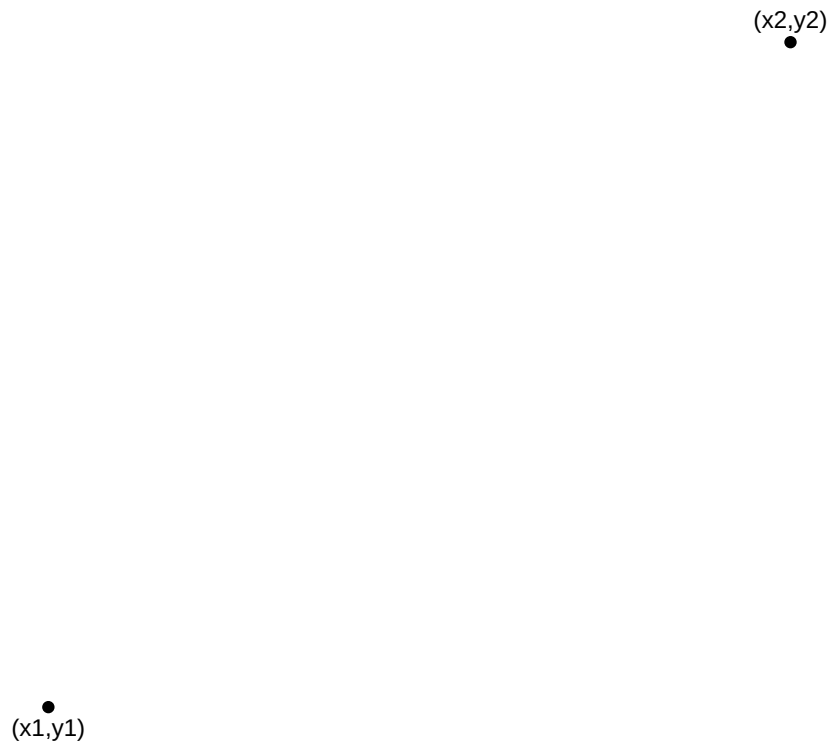
- There are many methods for unsupervised class discovery.
- We will consider three types of methods:
  - Ordination Methods (e.g., Multi-Dimensional Scaling (MDS) and Principal Components (PC))
  - Hierarchical Clustering
  - k-means Clustering
- Note that there are many variations of these methods
- Most mathematical details will be left out
- We focus on discovering classes among patients (not genes)

Distance between Two Points

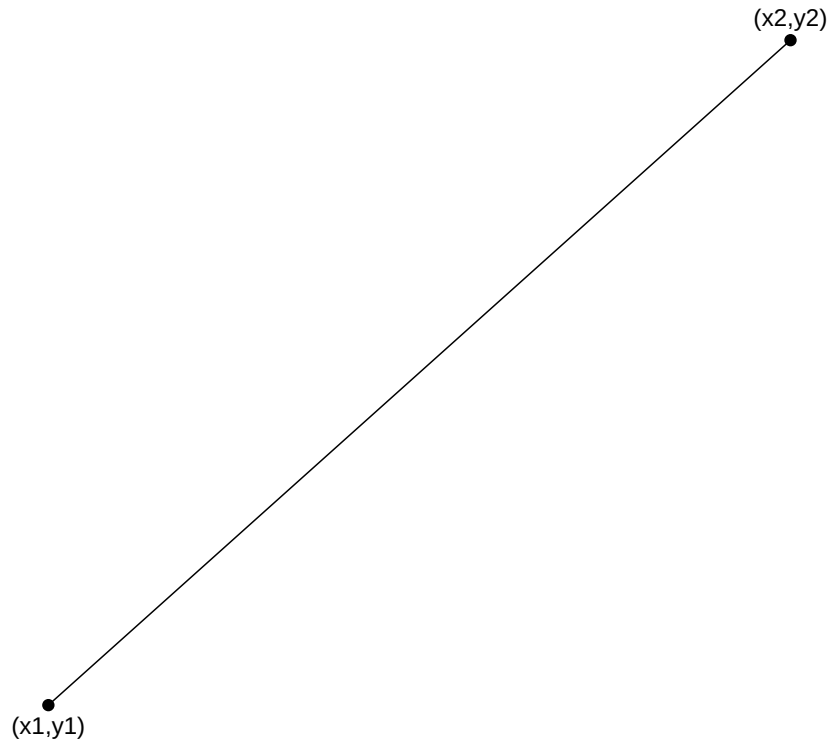
- Many class discover methods aim to quantify the similarity (or dissimilarity) among patients
- For each patient, the vector of gene expression can be thought of a "point" in a m-dimensional space
- For many class discovery methods, one has to be able to quantify the "distance" between two points (the expression profiles between two individuals)
- A common distance measure is the Euclidean distance

Distance (Two points on the plane)

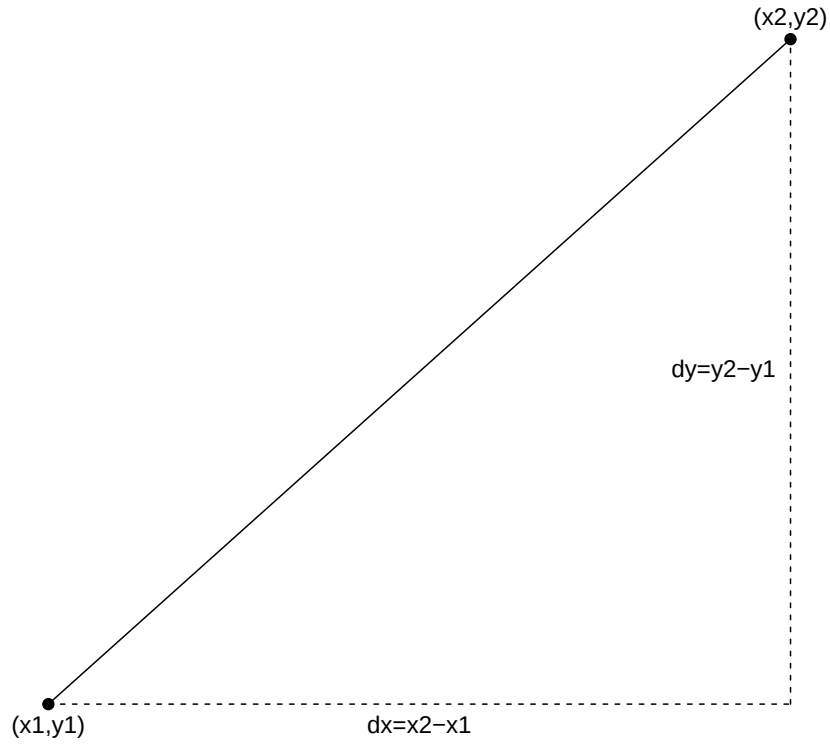
Distance (Coordinates)



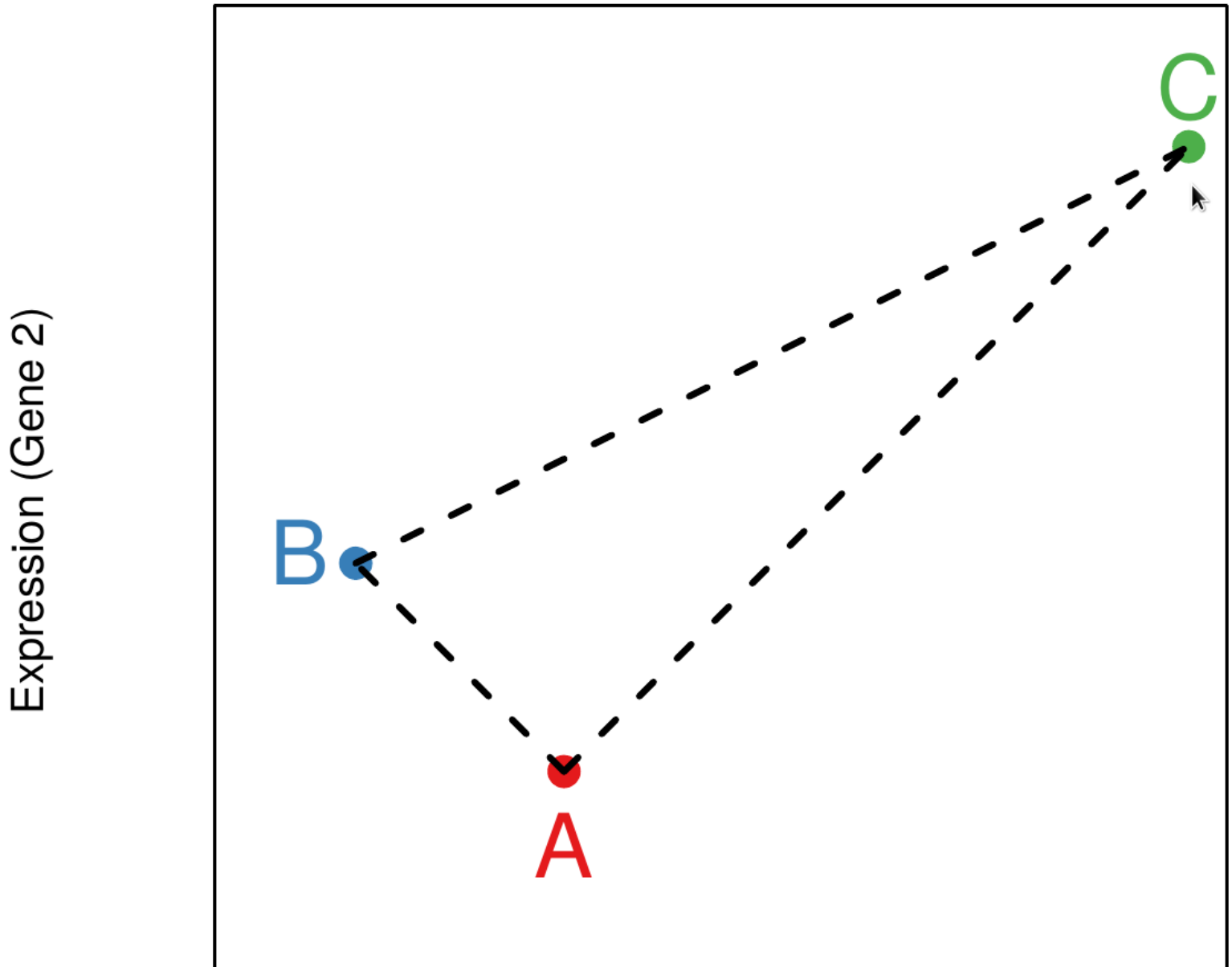
Distance



Distance (horizontal/vertical shifts)



Relative Distance (From CST 2011 Paper)



Dissimilarity matrix

- Use a distance to quantify similarity (or dissimilarity) among patients
- A matrix containing all pairwise distances
- Take the first three patients in the Golub data set (based on 7129 probe sets

```
dist(t(exprs(Golub_Merge[,1:3])))
```

```
##           39        40
## 40 101530.75
## 42  94405.04  89502.29
```

- The distance between patient 39 and 40 is 1.0153075×10^5
- Let us calculate this by hand

```
x=exprs(Golub_Merge)[,"39"]
y=exprs(Golub_Merge)[,"40"]
sqrt(sum((x-y)^2))
```

```
## [1] 101530.8
```

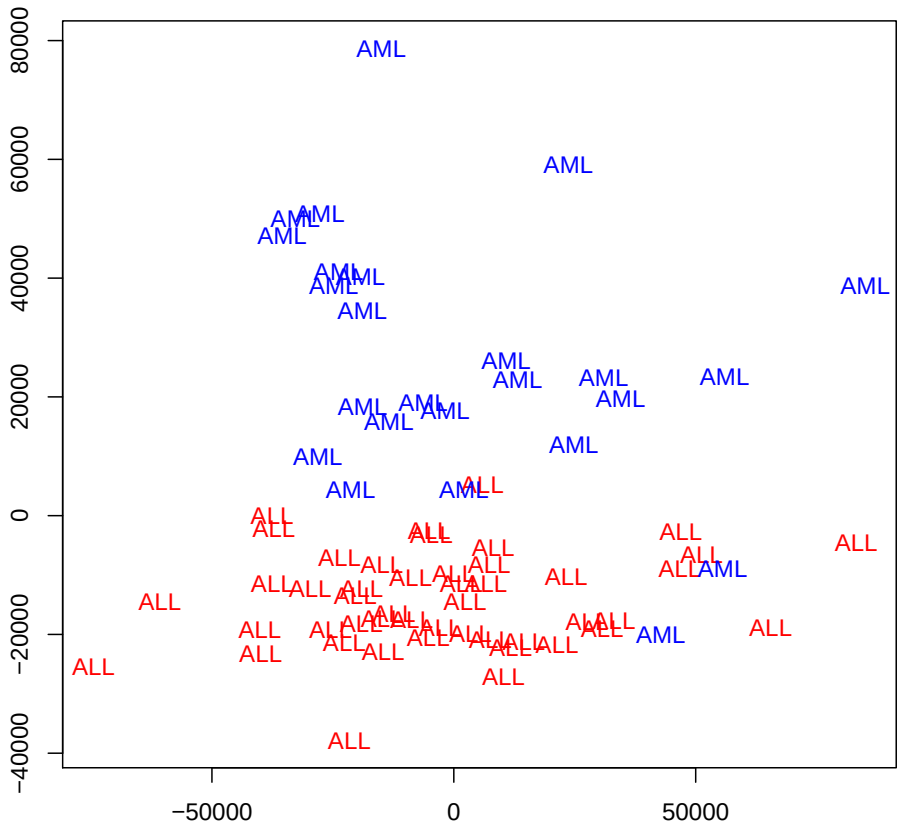
Dimension reduction

- Genome-wide profiling platforms are high-dimensional (m is large)
- Visualization beyond $m = 3$ not possible (for mortals)
- Representing the data by a lower dimensional format without losing too much information is desired.

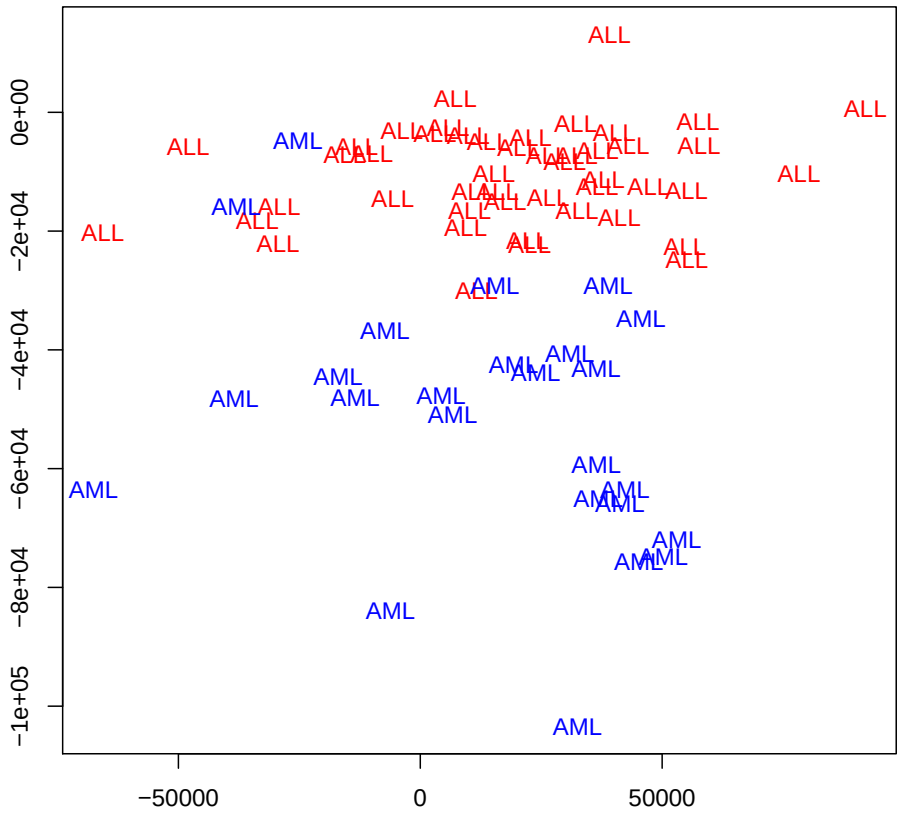
Multi-Dimensional Scaling (MDS)

- Compute the dissimilarity matrix based on a distance measure
- Project the points into a lower dimensional space (say 2D or 3D) while preserving the similarity matrix
- PCA is a related (and in a sense equivalent method to MDS)
- Project the points into a lower dimensional space where the new variables are linear combinations of the original variables
- The new variables are chosen so as to have maximum variance and to be uncorrelated.

MDS for Golub Data



PCA for Golub Data



Preserving The Distances

- Extract and standardize expression matrix for Golub data set

```
scexpdat=scale(t(exprs(Golub_Merge)))
dim(scexpdat)

## [1]   72 7129
```

- Check means for the first 4 genes

```
apply(scexpdat[,1:4],2,mean)

## AFFX-BioB-5_at AFFX-BioB-M_at AFFX-BioB-3_at AFFX-BioC-5_at
##  -7.841417e-17  -4.460287e-18   1.491832e-17  -5.051177e-17
```

- Check standard deviations for the first 4 genes

```
apply(scexpdat[,1:4],2,sd)

## AFFX-BioB-5_at AFFX-BioB-M_at AFFX-BioB-3_at AFFX-BioC-5_at
##              1              1              1              1
```

Preserving The Distances

- Check distance among the first three patients

```
dist(scexpdat[1:3,])
```

```
##          39       40
## 40 125.3402
## 42 118.1911 125.0390
```

- Calculate MDS $d = 2$

```
MDS=cmdscale(dist(scexpdat),2)
dist(MDS[1:3,])
```

```
##           39        40
## 40  4.644939
## 42 29.665656 34.287630
```

- Calculate MDS $d = 3$

```
MDS=cmdscale(dist(scexpdat),3)
dist(MDS[1:3,])
```

```
##           39        40
## 40  9.293559
## 42 45.719192 54.869668
```

Preserving The Distances

- Check distance among the first three patients

```
dist(scexpdat[1:3,])
```

```
##          39       40
## 40 125.3402
## 42 118.1911 125.0390
```

- Calculate MDS $d = 20$

```
MDS=cmdscale(dist(scexpdat),3)
dist(MDS[1:3,])
```

```
##           39        40
## 40  9.293559
## 42 45.719192 54.869668
```

- Calculate MDS $d = 45$

```
MDS=cmdscale(dist(scexpdat),45)
dist(MDS[1:3,])
```

```
##          39       40
## 40 124.9860
## 42 113.3668 121.7808
```

Distance between two clusters

- Let $c_1, c_2, \ldots, c_n$ denote the n patients
- We now know how to calculate a distance say between c_1 and c_5
- Define a cluster to be a set of "points"
  - $\{c_1\}$ is a cluster with one member: c_1
  - $\{c_1, c_3\}$ is a cluster of two members: c_1 and c_3
  - $\{c_1, c_2, c_3\}$ is a cluster of three members of c_1, c_2 and c_3

Notion of a Linkage

- The distance measure quantified the distance between two points
- In clustering, you need to think about the criterion to link (merge) the clusters
- maximum distance (aka complete linkage)
- average distance (aka average linkage)
- minimum distance (aka single linkage)

Agglomerative Hierarchical Clustering

- Agglomerate: To form clusters
- Let each of the n points be its own cluster (n clusters each with one single member)
- Find the pair of clusters that is most similar
- Now you have $n-1$ clusters (1 cluster with two members and $n-2$ clusters each with a single member)
- Compute the similarities between the $n-2$ "old" clusters with the new cluster
- Repeat the last two steps until all members have been merged into a single cluster.

Clustering Cities by Distances

	ATL	BOS	ORD	DCA
ATL	0	934	585	542
BOS	934	0	853	392
ORD	585	853	0	598
DCA	542	392	598	0

Clustering Cities by Distances (Single Linkage)

	ATL	BOS	ORD	DCA
ATL	0	934	585	542
BOS	934	0	853	392
ORD	585	853	0	598
DCA	542	392	598	0

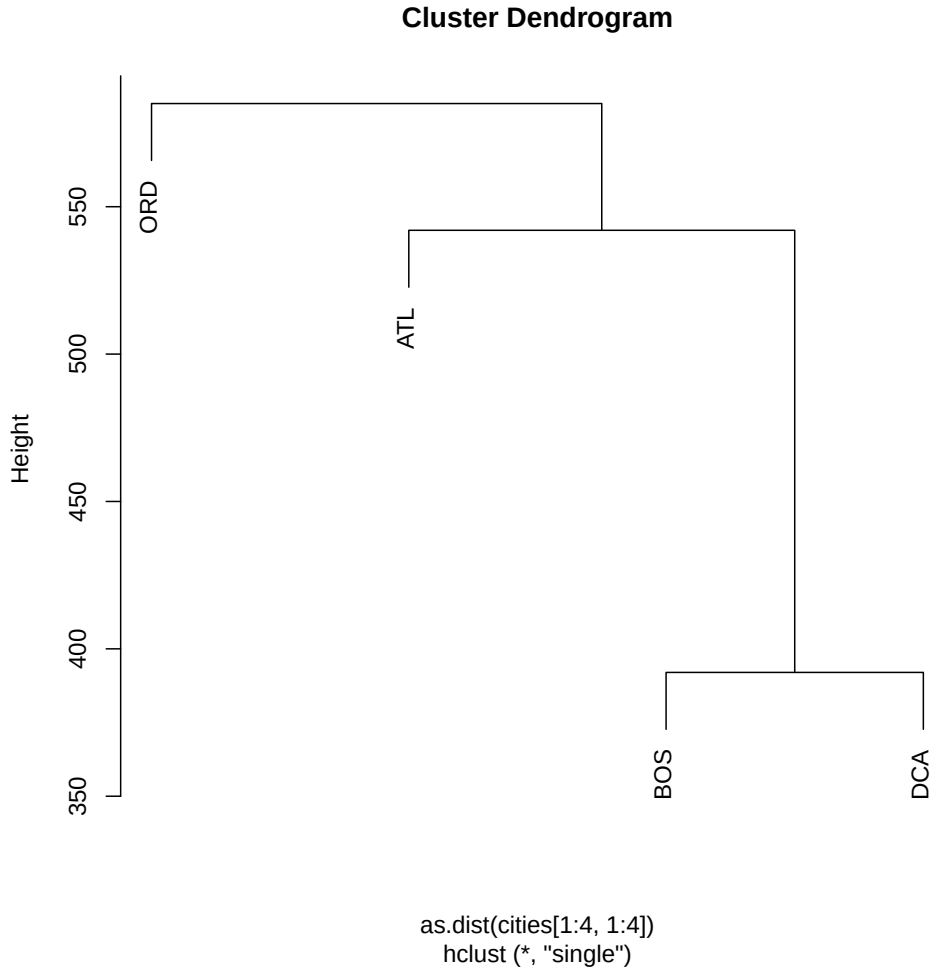
	DCA-BOS	ATL	ORD
DCA-BOS	0	542	598
ATL	542	0	585
ORD	598	585	0

Clustering Cities by Distances (Single Linkage)

	DCA-BOS	ATL	ORD
DCA-BOS	0	542	598
ATL	542	0	585
ORD	598	585	0

	DCA-BOS-ATL	ORD
DCA-BOS-ATL	0	585
ORD	585	0

Four Airports (Single linkage)



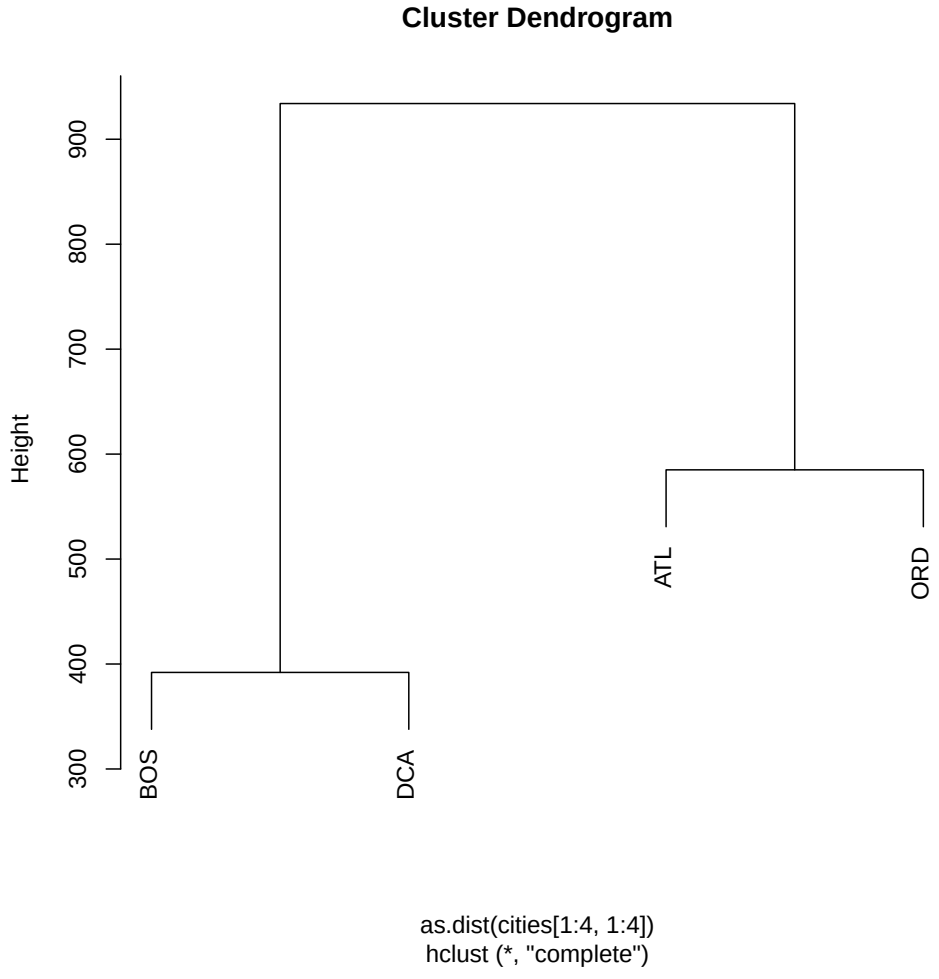
Clustering Cities by Distances (complete linkage)

	ATL	BOS	ORD	DCA
ATL	0	934	585	542
BOS	934	0	853	392
ORD	585	853	0	598
DCA	542	392	598	0

	DCA-BOS	ATL	ORD
DCA-BOS	0	934	853
ATL	934	0	585
ORD	853	585	0

	DCA-BOS	ATL-ORD
DCA-BOS	0	934
ATL-ORD	934	0

Four Airports (complete linkage)



Four Airports (side by side)

	ATL	BOS	ORD	DCA
ATL	0	934	585	542
BOS	934	0	853	392
ORD	585	853	0	598
DCA	542	392	598	0

	DCA-BOS	ATL	ORD
DCA-BOS	0	934	853
ATL	934	0	585
ORD	853	585	0

	DCA-BOS	ATL-ORD
DCA-BOS	0	934
ATL-ORD	934	0

Table : Complete Linkage

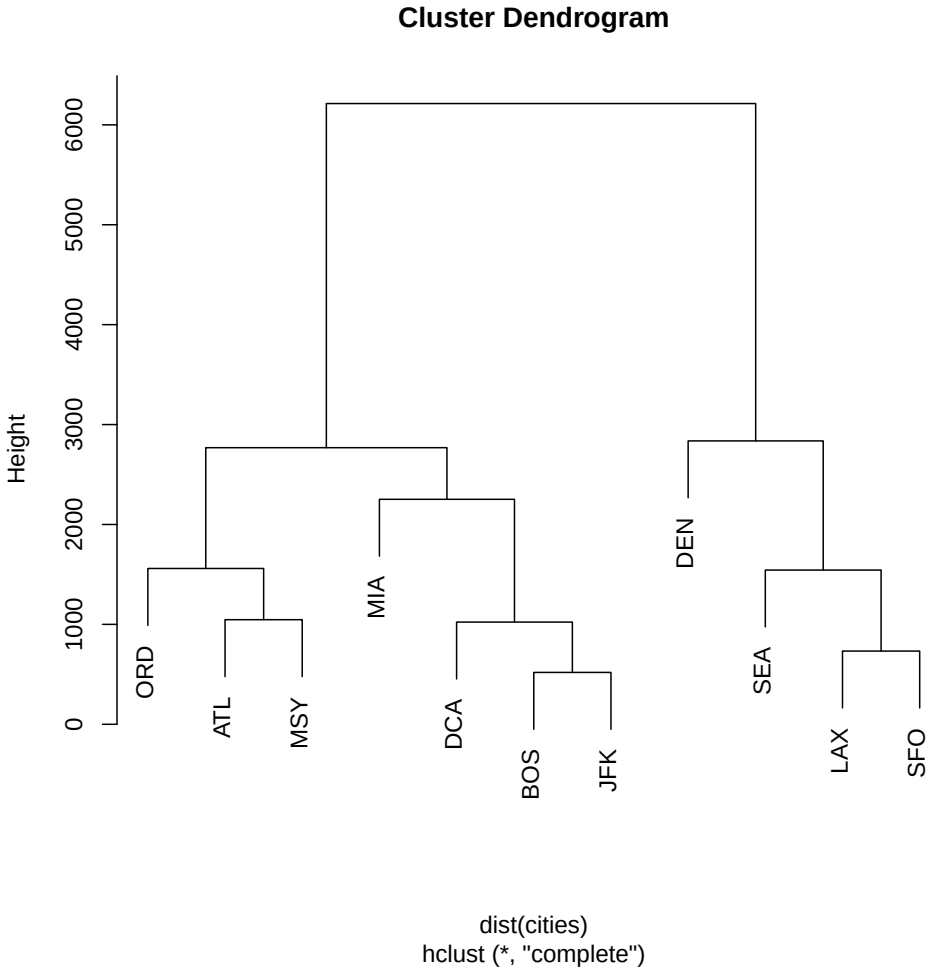
	ATL	BOS	ORD	DCA
ATL	0	934	585	542
BOS	934	0	853	392
ORD	585	853	0	598
DCA	542	392	598	0

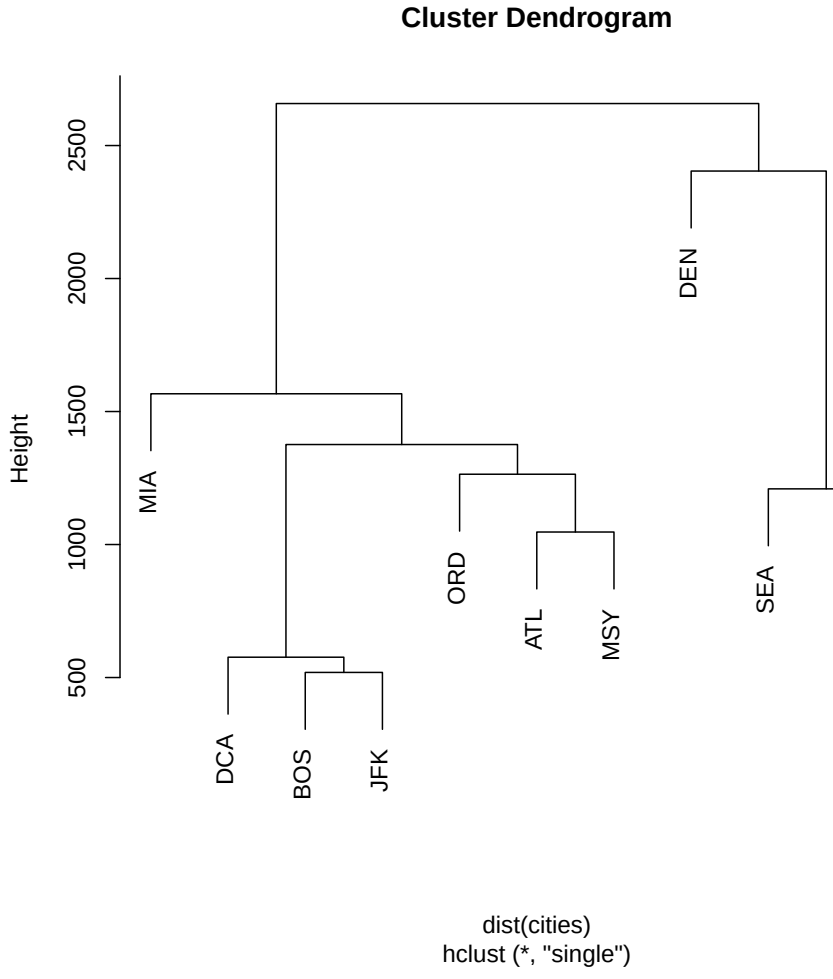
	DCA-BOS	ATL	ORD
DCA-BOS	0	542	598
ATL	542	0	585
ORD	598	585	0

	DCA-BOS-ATL	ORD
DCA-BOS-ATL	0	585
ORD	585	0

Table : Single Linkage

All Airports (comparison)

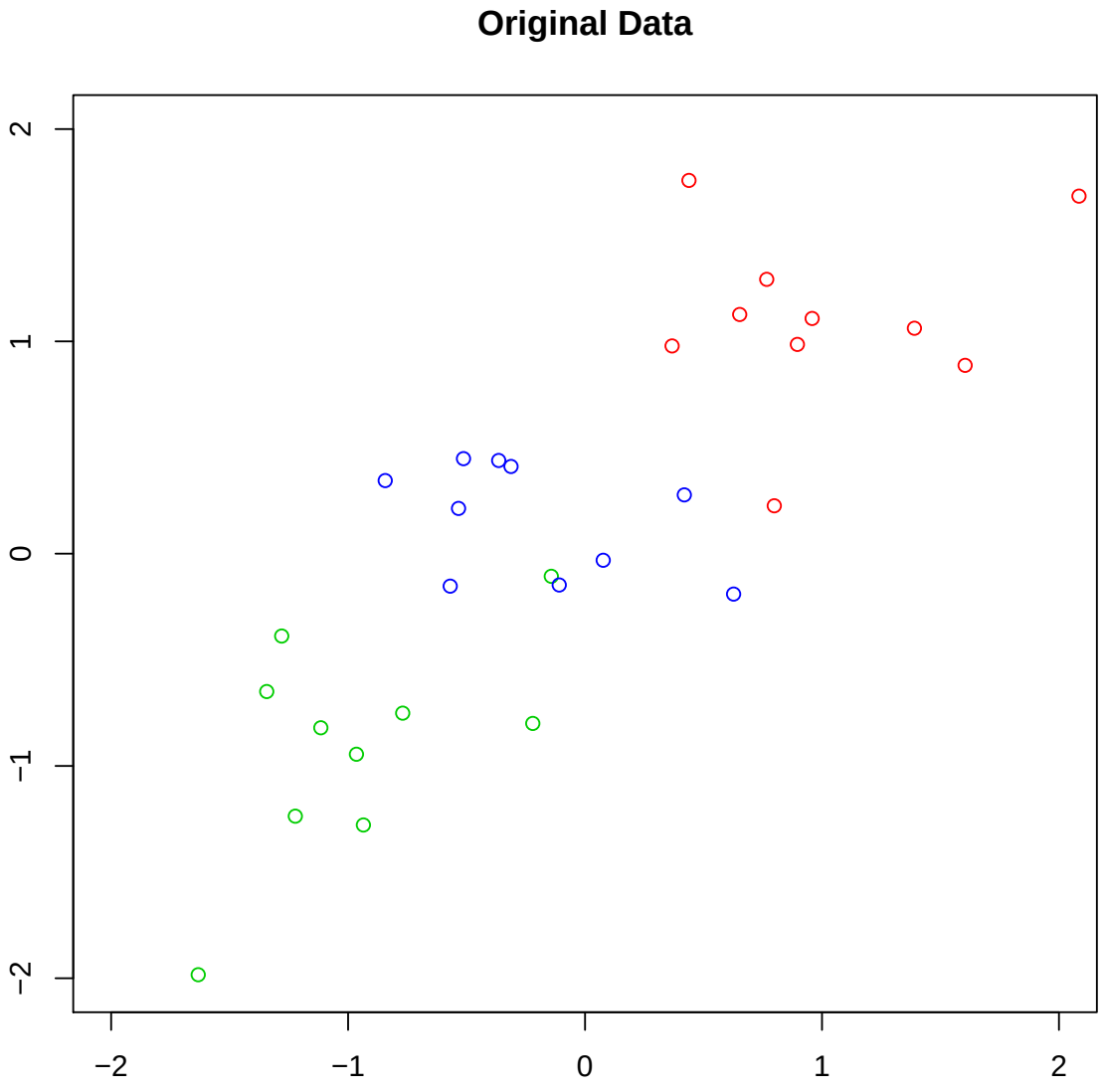




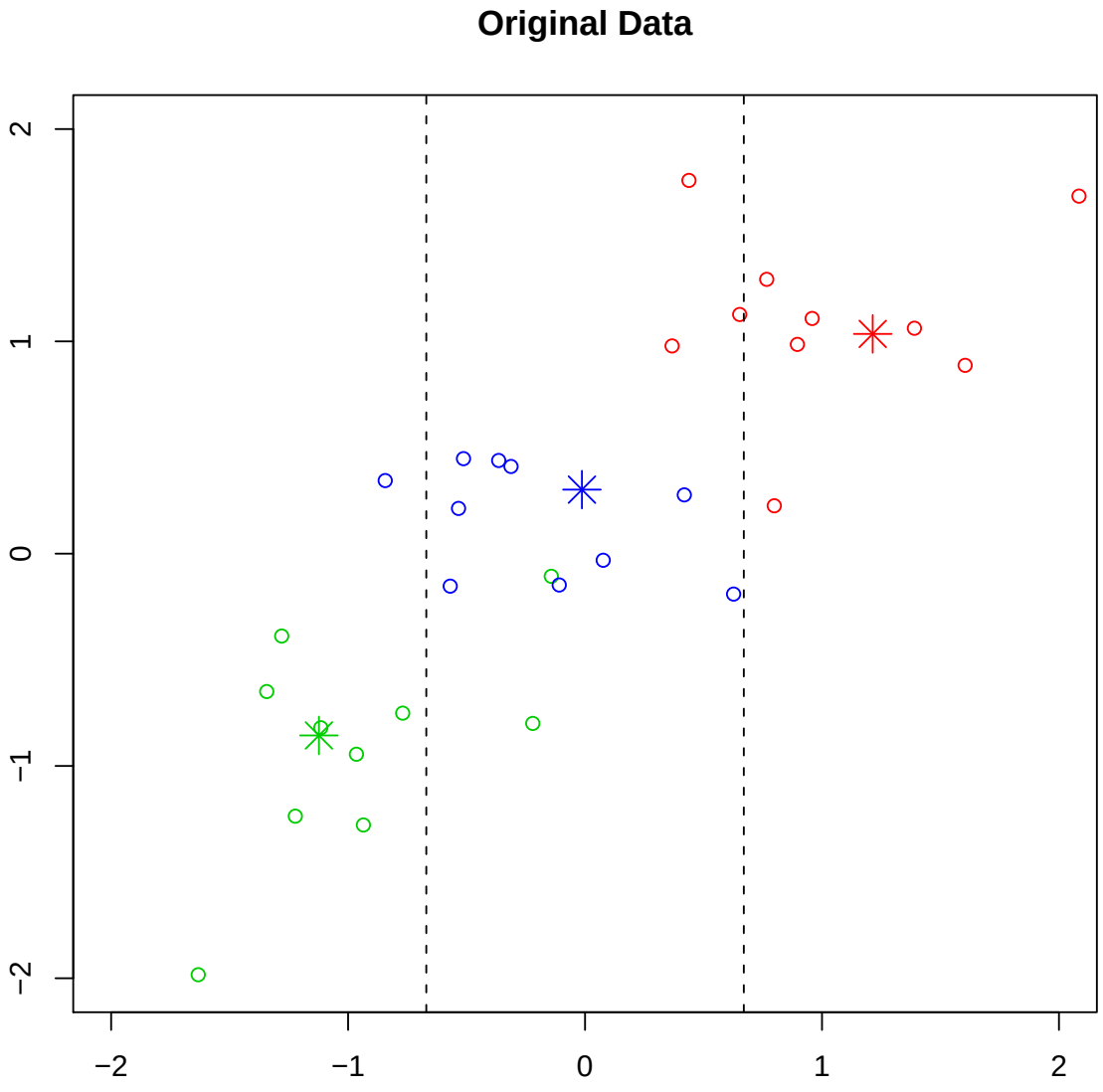
k-means Clustering

- Specify a number of potential clusters (k)
- Split of the data (either randomly or based on some previous results) into k partitions
- Compute the mean (aka centroid) for each partition
- For the first point (sample) determine the *nearest* centroid
- The closeness is typically quantified using the Euclidean distance
- Assign that point to that center
- Repeat for points 2 through n
- Assess the fit using the intra-cluster variance
- Repeat as needed.

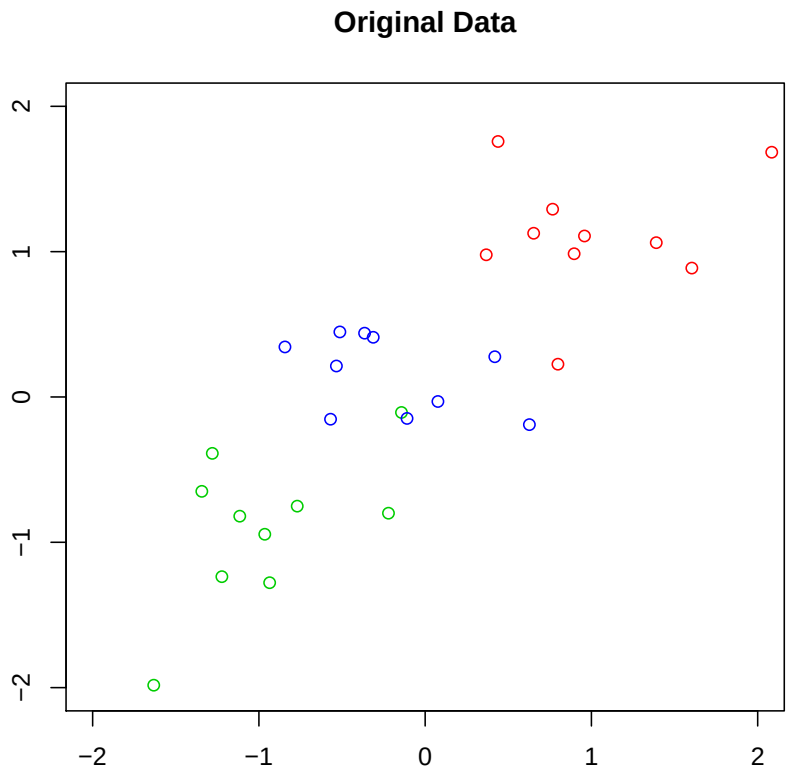
k-means Illustration

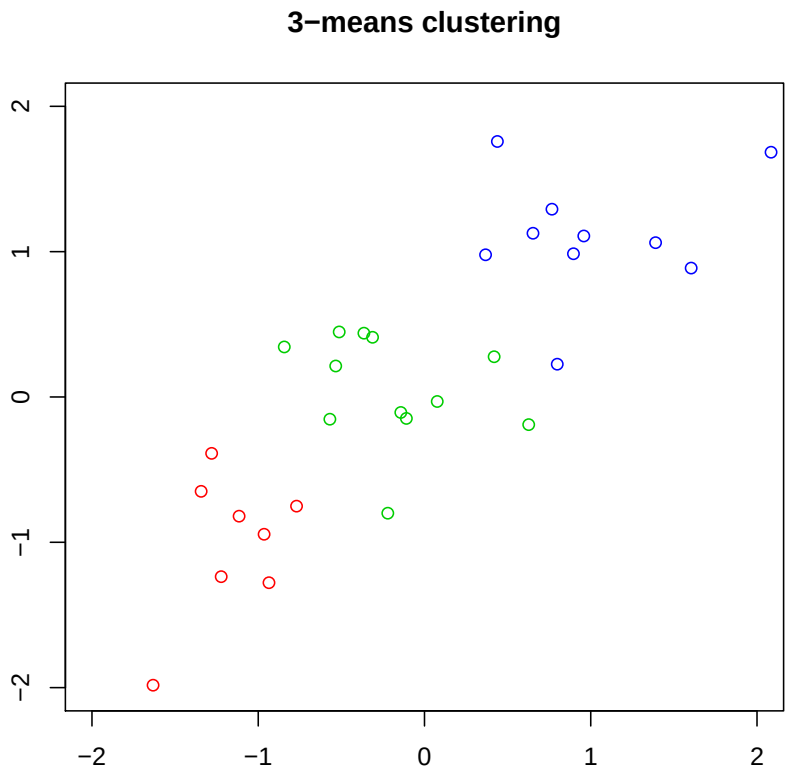


k-means Illustration



k-means Illustration





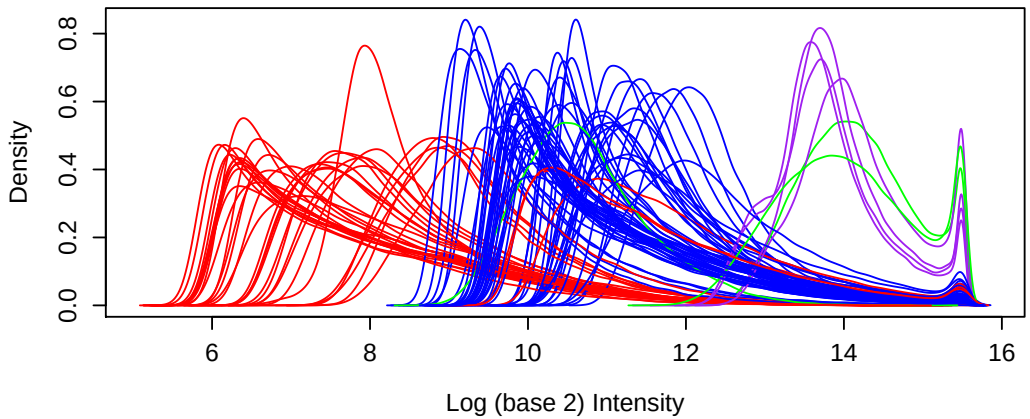
k-means

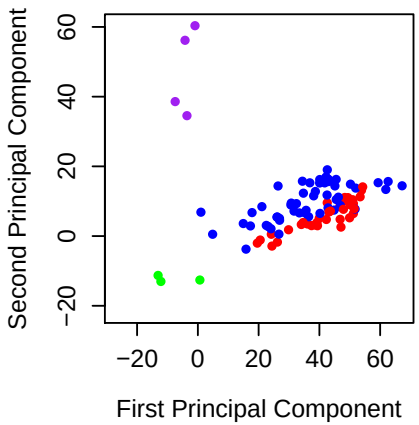
- This is an example of *non-hierarchical* clustering
- Need to specify the number of clusters up front
- Need to specify (deterministically or randomly) the centers of the clusters up front
- Results are sensitive to the choice of k and initial partitions
- There is a relationship between k-means and PCA.

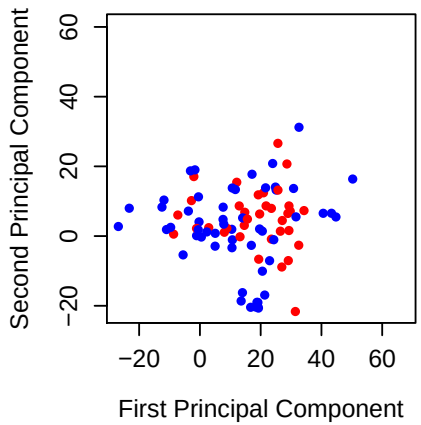
Batch Effect Discovery

- The MDS method is very useful for detecting batch effects
- Batch effects tend to be stronger that biological effects
- They also affect most probe sets (the biological effect may only be captured by a few)
- This can be an effective weapon in your QC arsenal (this is how I start any new analysis)

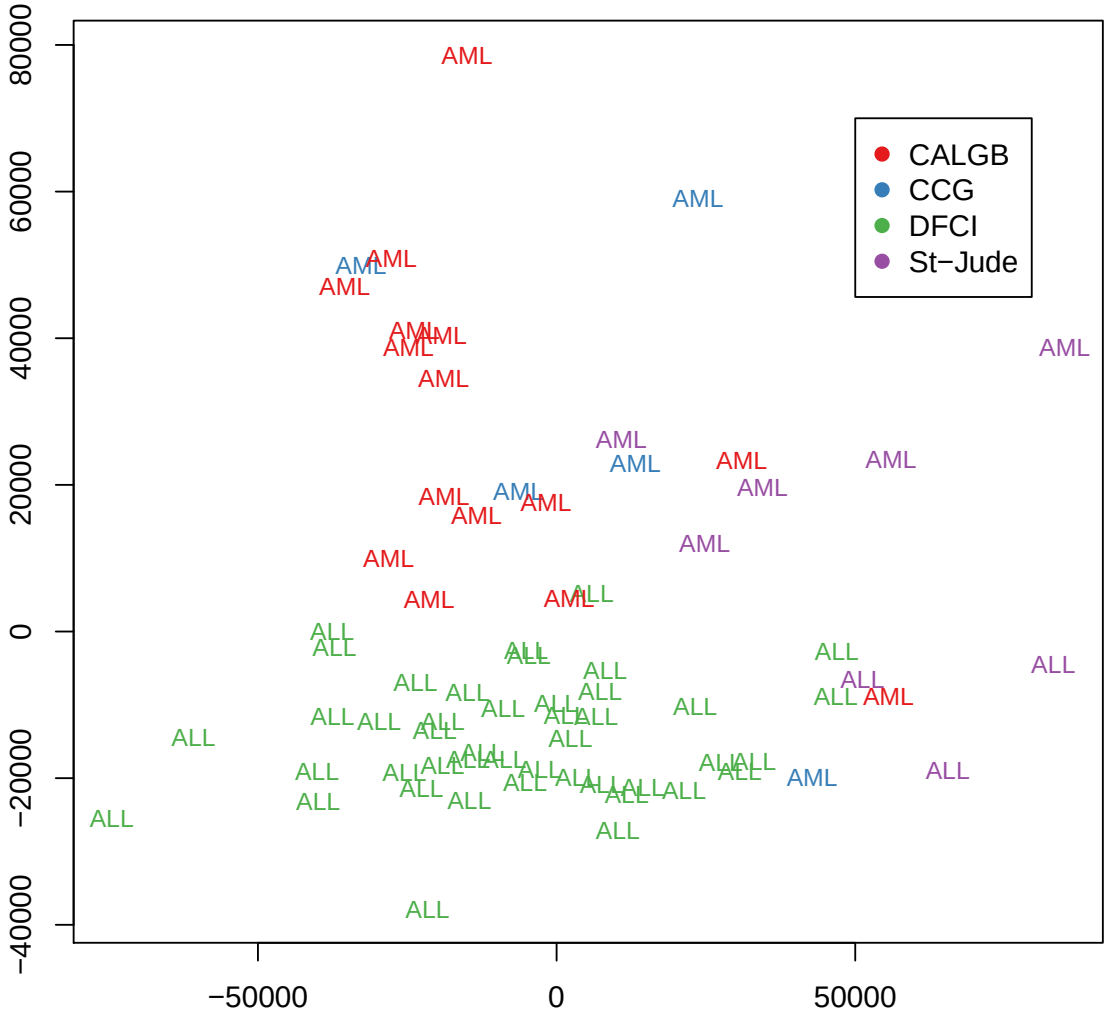
From CCR 2008 Paper







ALL/AML Data


CALGB
CCG
DFCI
St−Jude
80000
60000
40000
20000
0
−20000
−40000
−50000
0
50000

Semi-supervised Learning

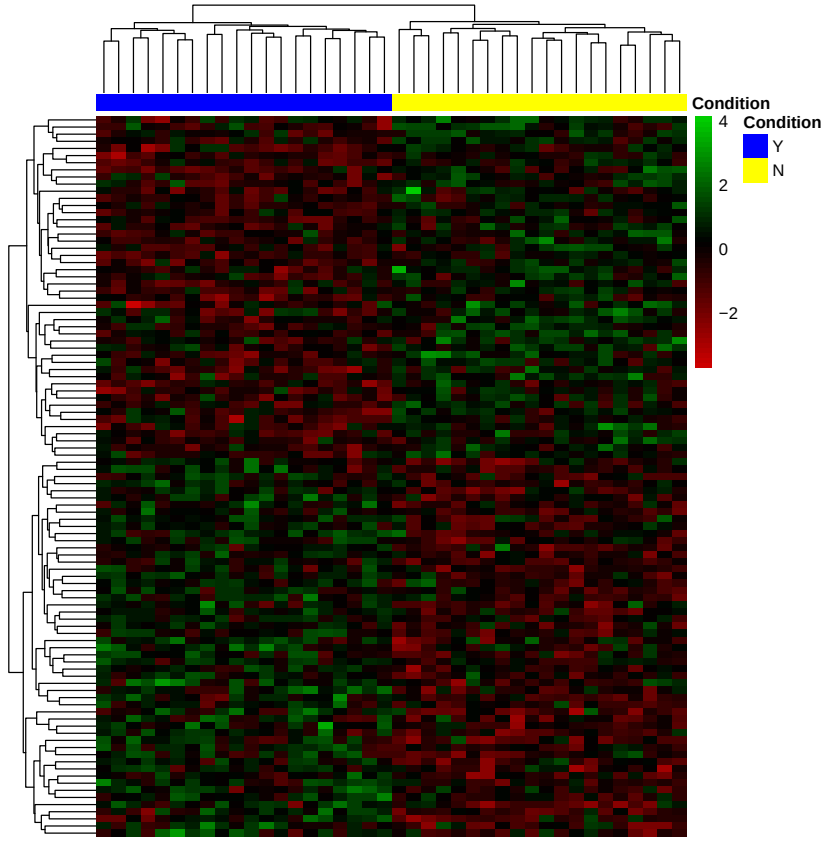
- Heatmap illustration:
  - Select a panel of probe-sets based on the two-sample t-test
  - Carry out hierarchical clustering with respect to the patients (the columns)
  - Carry out hierarchical clustering with respect to the probe sets in the panel (the rows)
  - Present the results using a heatmap
- Some consider this an *unsupervised* analysis as the hierarchical clustering algorithm is unaware of the classes
- This is not an accurate assessment: It is semi-supervised in the sense that we are picking genes based on the phenotype
- A procedure is *unsupervised* if the class info is only used for annotation

R Code to simulate Heatmap

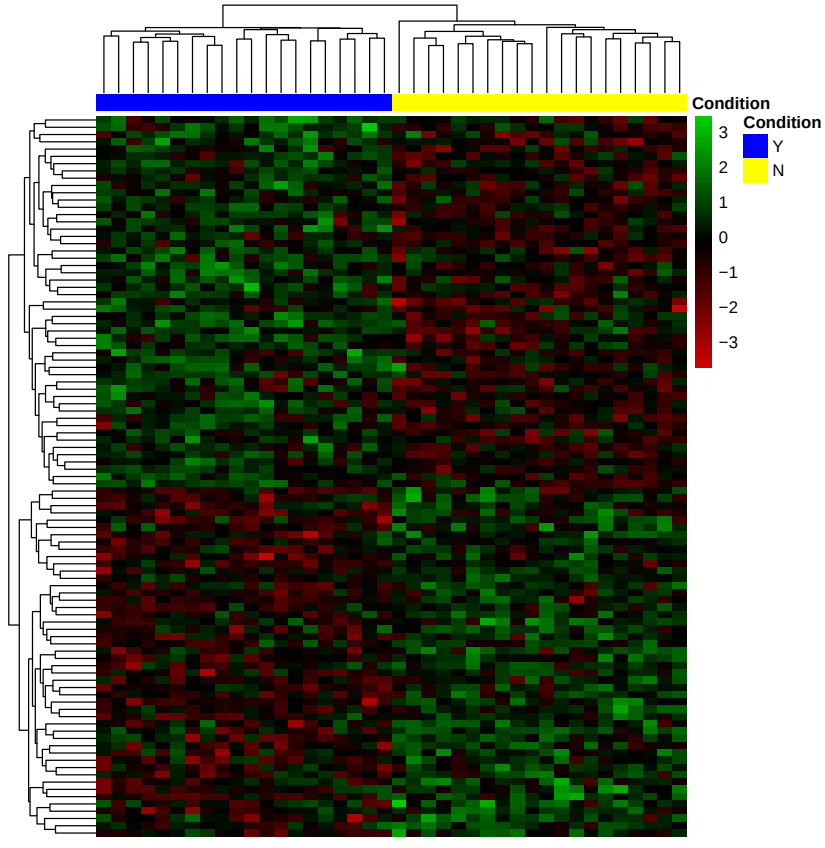
```
simulate.noise.heatmap=function(n,m,alpha)
  {
    # Simulate Expression Matrix
    EXPRS=matrix(rnorm(2*n*m),m,2*n)
    grp=factor(rep(0:1,c(n,n)))
    rownames(EXPRS)=paste("Gene",1:m,sep="")
    colnames(EXPRS)=paste("patient id",1:(2*n),sep="")

    # Get the two sample t-statistics
    pvals=rowttests(EXPRS, grp)$p.value
    topgenes=which(pvals<alpha)
    EXPRS=EXPRS[topgenes,]
    annodat=data.frame(Condition=ifelse(grp==0,"N","Y"),row.names=colnames(EXPRS))
    pheatmap(EXPRS,
             border_color =NA,
             show_rownames = FALSE,
             show_colnames=FALSE,
             annotation_col=annodat,
             color=colorRampPalette(c("red3", "black", "green3"))(50),
             annotation_colors=list(Condition=c(Y="blue",N="yellow")))
    return(length(topgenes))
  }
```

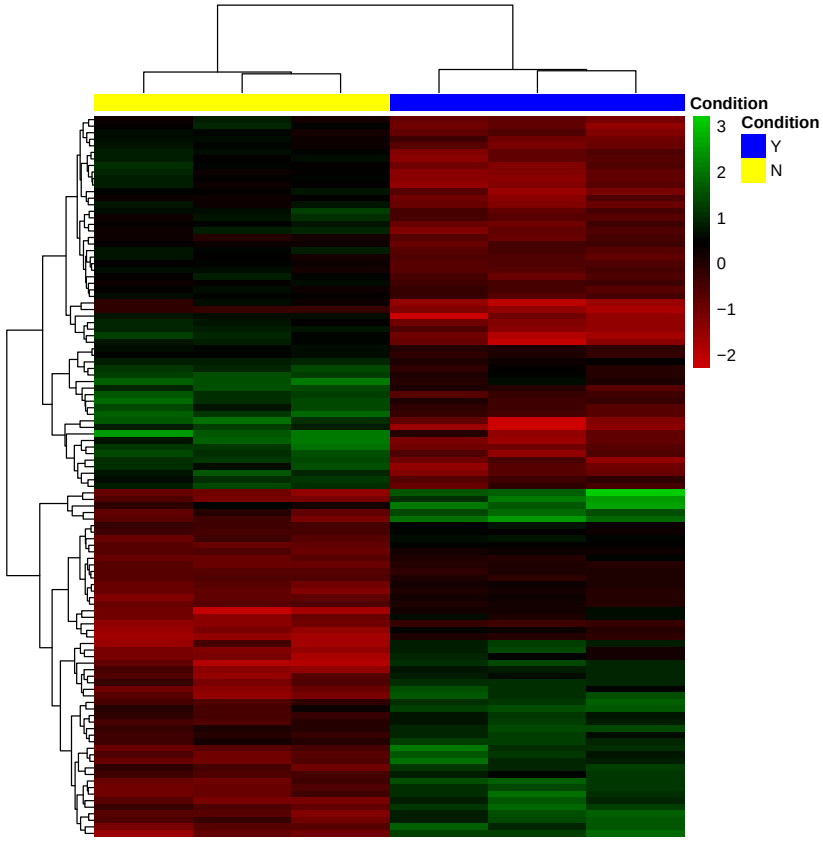
Heatmap Example: $m = 20,000$, $n = 20$, $\alpha = 0.005$



Heatmap Example: $m = 40,000$, $n = 20$, $\alpha = 0.0025$



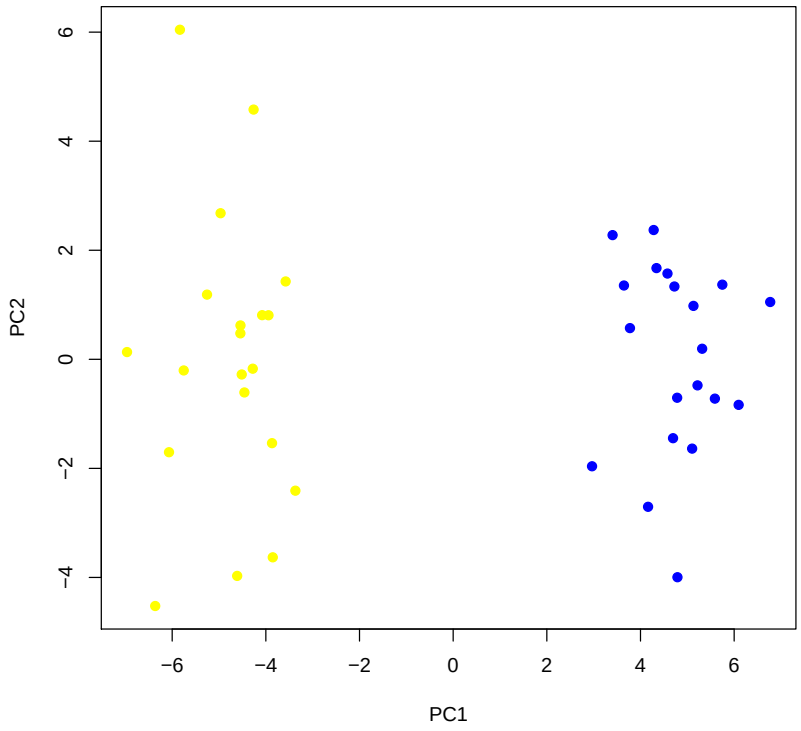
Heatmap Example: $m = 20,000$, $n = 3$, $\alpha = 0.005$



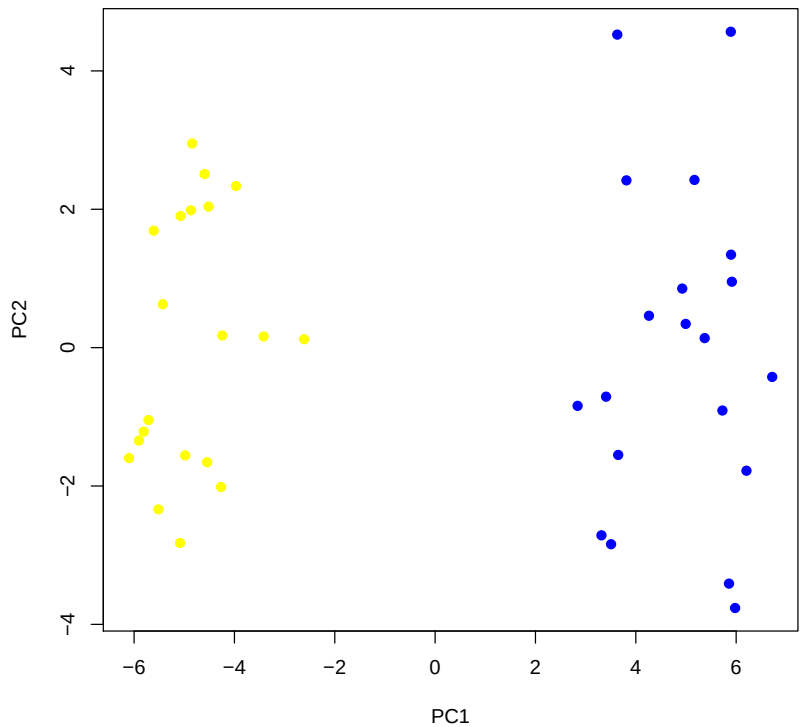
R Code to simulate PC

```
simulate.noise.PC=function(n,m,alpha)
  {
    # Simulate Expression Matrix
    EXPRS=matrix(rnorm(2*n*m),m,2*n)
    grp=factor(rep(0:1,c(n,n)))
    # Get the two sample t-statistics
    pvals=rowttests(EXPRS, grp)$p.value
    topgenes=which(pvals<alpha)
    EXPRS=EXPRS[topgenes,]
    annodat=data.frame(Condition=ifelse(grp==0,"N","Y"),row.names=colnames(EXPRS))
    PC=cmdscale(dist(t(EXPRS)))
    plot(PC,xlab="PC1",ylab="PC2",col=ifelse(grp==0,"yellow","blue"),pch=19)
    return(length(topgenes))
}
```

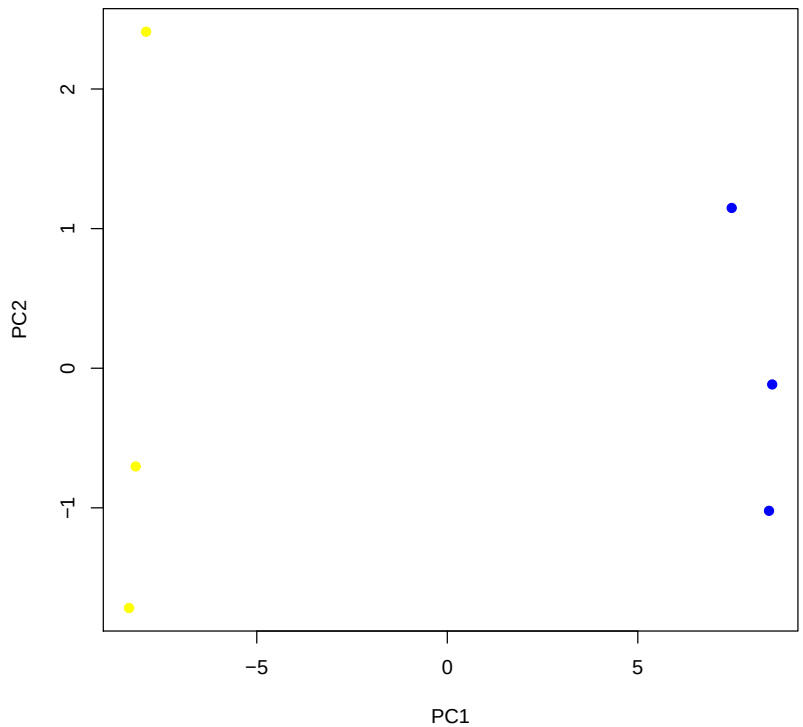
Heatmap Example: $K = 20000, n = 20, \alpha = 0.005$



Heatmap Example: $K = 40000, n = 20, \alpha = 0.0025$



Heatmap Example: $K = 20000, n = 3, \alpha = 0.005$



Reminder: A Self-fulfilling Prophecy

- Statistical method for unsupervised learning guarantee one thing
- They will return a clustering of your data
- What they do not guarantee and are invariably unable to verify, is the biological relevance or reproducibility of the clustering
- In light of this Self-fulfilling Prophecy, these methods should be used with utmost care

Section 6

Elements of Multiple Testing

Multiple Testing: Motivation

- Flip a single coin from a large batch of newly minted coins 10 times

```
##  [1] "T" "T" "T" "T" "T" "T" "T" "H" "T" "T"
```

- Is this a biased coin?

```
##
##  Exact binomial test
##
## data:  sum(x == "T") and length(x)
## number of successes = 9, number of trials = 10, p-value = 0.02148
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.5549839 0.9974714
## sample estimates:
## probability of success
##                    0.9
```

Multiple Testing: Motivation

- Flip two coins each 10 times

```
##  [1] "T" "T" "T" "T" "T" "T" "T" "H" "T" "T"
##  [1] "T" "H" "T" "H" "H" "H" "T" "T" "H" "H"
```

- Are any of the two coins biased?

```
binom.test(sum(x1=='T'), n=length(x), p = 0.5)
```

```
##
##  Exact binomial test
##
## data:  sum(x1 == "T") and length(x)
## number of successes = 9, number of trials = 10, p-value = 0.02148
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.5549839 0.9974714
## sample estimates:
## probability of success
##                    0.9
```

```
binom.test(sum(x2=='T'), n=length(x), p = 0.5)
```

```
##
##  Exact binomial test
##
## data:  sum(x2 == "T") and length(x)
## number of successes = 4, number of trials = 10, p-value = 0.7539
```

Multiple Testing

- We have previously considered testing for significance of a single gene
- The analysis of high-dimensional data, including array and sequencing data, is concerned with testing the significance of multiple loci/genes
  - Microarray : 20,000-50,000 probe sets
  - GWAS: 500,000-5,000,000 typed SNPs
  - RNA-Seq: 22,000 genes (humans), ? genes (ecoli)
- Let m denote the number of genes (or SNPs) to be tested
- Rather than testing a single hypothesis, we are concerned with testing multiple hypotheses
- The decision rule must now account for testing m hypotheses simultaneously (multiple testing)

Hypothesis Notation

- Gene j (among the m genes) is either associated with the outcome or not
- The truth is unknown to us
- The null hypothesis for gene j is denoted by H_j (gene j is
- H_j: gene j is not associated with the outcome of interest
- The alternative hypothesis is denoted by $\bar{H}_j$
- $\bar{H}_j$: gene j is associated with the outcome of interest
- Suppose that we only test a single gene, say gene j, among the m genes
- Let p_j (lower case p) denote the corresponding P-value
- p_j is called the *marginal* or *unadjusted* P-value
-

Unadjusted vs Adjusted *P*-values

- Suppose that we only test a single gene, say gene j, among the m genes
- Let p_j (lower case p) denote *P*-value corresponding to H_j
- p_j is called the *marginal* or *unadjusted* *P*-value
- If m hypotheses are tested, inference on H_j on the basis of p_j is inappropriate
- The *P*-value for H_j has to account for testing the other $m - 1$ hypotheses
- We will denote the *adjusted* *P*-value by P_j (upper case P)

Additional Notation

- Suppose that gene j is not associated with the outcome of interest (H_j is true)
- Then
  - Decision rule rejects $\rightarrow$ False-Positive (FP)
  - Decision rule fails to reject $\rightarrow$ True-Negative (TN)
- Suppose that gene j is associated with the outcome of interest (H_j is false)
  - Decision rule rejects $\rightarrow$ True-Positive (TP)
  - Decision rule fails to reject $\rightarrow$ False-Negative (FN)

Summarizing a Multiple Testing Procedure

- The results from any multiple testing procedure can be summarized as the following table

	Accept	Reject	Total
Truth Null	A_0	R_0	m_0
Alt.	A_1	R_1	m_1
	A	R	m

- Notation:
  - m: Number of tests, m_0, m_1 number of null/true genes
  - R: Number of genes rejected according to the decision rule
  - A: Number of genes accepted according to the decision rule
  - R_0/R_1 number of TN/FP
  - A_0/A_1 number of FN/TP

Example

- Results from an analysis based on $m = 10$ genes:

```
##       gene truth  pvalue
## 1    gene1     0 0.29070
## 2    gene2     1 0.61630
## 3    gene3     1 0.00320
## 4    gene4     0 0.01641
## 5    gene5     0 0.25150
## 6    gene6     0 0.58450
## 7    gene7     0 0.22890
## 8    gene8     1 0.12630
## 9    gene9     0 0.26080
## 10 gene10     0 0.04980
```

- Investigator decides to use following decision rule: Any gene with a corresponding unadjusted P-value of less than 0.05 will be rejected.
- Note:
  - $m_0 = 7$ and $m_1 = 3$
  - $R = 3$ will be rejected based on the decision rule
  - Consequently $A = m - R = 7$ will be accepted
  - $R_0 = 2, R_1 = 1, A_0 = 5$ and $A_1 = 2$

Example: Fill in the 2x2 table

	Accept	Reject	Total
Truth Null	$A_0 = 5$	$R_0 = 2$	$m_0 = 7$
Alt.	$A_1 = 2$	$R_1 = 1$	$m_1 = 3$
	$A = 7$	$R = 3$	$m = 10$

The Truth

- What know or observe is this

```
##       gene  pvalue
## 1    gene1 0.29070
## 2    gene2 0.61630
## 3    gene3 0.00320
## 4    gene4 0.01641
## 5    gene5 0.25150
## 6    gene6 0.58450
## 7    gene7 0.22890
## 8    gene8 0.12630
## 9    gene9 0.26080
## 10 gene10 0.04980
```

- and not (truth colum is not known to us):

```
dat

##       gene truth  pvalue
## 1    gene1     0 0.29070
## 2    gene2     1 0.61630
## 3    gene3     1 0.00320
## 4    gene4     0 0.01641
## 5    gene5     0 0.25150
## 6    gene6     0 0.58450
## 7    gene7     0 0.22890
## 8    gene8     1 0.12630
## 9    gene9     0 0.26080
## 10 gene10     0 0.04980
```

Example: Fill in the 2x2 table (based on what we observe)

- We can only fill in the bottom row of the table

	Accept	Reject	Total
Truth Null	A_0	R_0	m_0
Alt.	A_1	R_1	m_1
	$A = 7$	$R = 3$	$m = 10$

- The remaining quantities are fixed unknown quantities or unobservable random variables.

Framework of multiple testing

	Accept	Reject	Total
Truth Null	A_0	R_0	m_0
Alt.	A_1	R_1	m_1
	A	R	m

- m is a known constant
- m_0 and m_1 are unknown constants
- R and A are determined on the basis of applying the decision rule to the data
- They are *observable* random quantities
- The true states of the genes of the genes are unknown
- A_0, A_1, R_0 and R_1 are *unobservable* random quantities

Framework versus Method

- To account for multiple testing one has to first decide on a framework and then on a method
- Framework: The quantity that we aim to control
- Method: statistical procedure used to for estimating or controlling the error rate for a set of hypothesis tests.
- Example: Investment
  - What is the objective: capital preservation or growth
  - Approach: Index funds, individual stocks, CDs, money under mattress
- : When thinking of multiple testing, first decide what the framework is and then decide on an appropriate strategy

Family-wise Error Rate (FWER)

- What is the probability to commit at least one false-rejection (among m) given that *all* genes are null
- What is the the probability of the event $R \geq 1$ if $m = m_0$
- $\mathrm{FWER} = P(R \geq 1 | m = m0)$
- Note that when $m = 1$ (single gene), this definition is identical to the tyep I error we have previously considered

Bonferroni

- A simple method for controlling FWER is called the Bonferroni method
- To control the type I error of the experiement at the α level, test each gene at the $\frac{\alpha}{m}$ level
- The Bonferroni adjusted *P-value* is defined as

$$P_j = m \times p_j$$

- Technical note: P_j is defined above could be larger than 1 so a more technically rigorous definition is

$$P_j = \min\{m \times p_j, 1\}$$

- In other words, if $m \times p_j$ is larger than 1, then truncate P_j at 1.

False Discovery Rate (FDR)

- In the FWER framwork, the objective is to control $\mathrm{FWER} = P(R \geq 1 | m = m0)$
- This is the probability of at least one false-discovery when none of the genes are true.
- Consider the quantity $\frac{R_0}{R}$
- This is the proportion of of false discoveries among the genes rejected
- This is an *unobservable* random quantity (As R_0 is not observable)
- In the FDR framework is based on controlling the *expected* value of this ratio
- The FDR is defined as $E[\frac{R_0}{R}]$
- Note that when $m_0 = m$ (none of the genes are true), FWER=FDR

Methods for the FDR Framework

- An early method proposed to control FDR, is a method due to Benjamini and Hochberg (BH; JRSBB 1985)
- One of the assumptions for the BH method is that of independence among the genes
- That assumption may be questionable (due to co-regulation among genes)
- A more recent approach is due to Storey
- The adjusted *P*-values calculated based on Storey's method are called *Q*-values

Genome-wide Significance

- In GWAS papers, $\alpha = 5 \times 10^{-8}$ is typically considered the threshold for genome-wide significance
- It is based on a Bonferroni correction: If you consider testing $m = 1,000,000$ SNPs at the FWER level of 0.05, then each SNP should be tested at the

$$\alpha = \frac{0.05}{1,000,000} = 5 \times 10^{-8},$$

  level
- Suppose that the unadjusted P=value for a SNP is 5×10^{-7}
- Is this "reaching" genome-wide significance?
- The term "suggestive" is also used

"Reaching" Genome-wide Significance

- Suppose that the $m = 1,000,000$ SNPs are independent
- The adjusted P-value is

$$P = 5 \times 10^{-7} \times m = 5 \times 10^{-7} \times 10^{6} = 0.5,$$

- This is off by an order of magnitude $(0.5 = 0.05 \times 10)$
- It is not "reaching"

Summary of Multiple Testing

- Multiple testing *must* be accounted for when testing for associations in the context of high-dimensional data
- FWER and FDR are the two common frameworks for quantifying error
- Error rate estimates can be used to compute 'adjusted' p-values
- Resampling-based methods can increase power in controlling error when sample sizes are sufficient for their use.
- When large-scale patterns of differential expression are observed, it is important to consider if such effects are biologically reasonable, and if technical factors can be attributed to the variation.

Section 7

Distributions for Counts

Two Approaches for Analysis of RNA-Seq

- Two-stage method: Convert counts to "Expression" and then use statistical methods for microarrays (e.g., t-test) and then
- One-stage method: Relate the counts directly to the phenotype
- This is done through using statistical methods for modeling counts
- We generally promote the latter approach for data analysis

DESeq for RNA-Seq

- The goal is to provide sufficient background to understand the DESeq method
- We are not suggesting that DESeq is the best approach for analysis of RNA-Seq data
- We are considering it in this course as one, of many other methods, that adhere to the one-stage approach principle
- Added bonus: Nicely written `R` extension package (important feature for teaching)
- DESeq has many limitations (e.g., it cannot directly deal with quantitative and censored outcomes)
- Also some of the theoretical details (e.g., the effect of using plugin estimates for nuisance parameters) have seemingly not been fully fleshed out

Three Distributions for Count Data

- RNA-Seq data are counts (not continuous measurements)
- To properly model RNA-Seq data, we need to consider distributions to model counts
- We will consider three important distributions for counts:
  - Binomial
  - Poisson
  - Negative Binomial
- There are many other distributions for counts (e.g., geometric distribution) that will not be discussed

Distribution for Counts: Support

- When considering a distribution of a count variable, we first have to determine its *support*
- The support of the distribution consists of the values that could occur with positive probability
- For example, if we toss a coin once and we count the number of heads, the support is $\{0, 1\}$
- If we flip it twice, the support is $\{0, 1, 2\}$
- Why is 3 not in the support? How about -1?
- These values are not *possible* (they have zero probability)

Distribution for Counts: Probability Mass Function

- Example: we toss a fair coin once and we count the number of heads (call it K)

$$P(K = 0) = \frac{1}{2} \text{ and } P(K = 1) = \frac{1}{2}$$

and

$$P(K = k) = 0$$

if k is not 0 or 1

- The probability mass function (PMF) determines the probability that K assumes value k in the support
- Sometimes we use the terms "distribution" and "PMF" interchangeably

Distribution for Counts: Probability Mass Function

- Example: we toss a fair coin twice and we count the number of heads (call it K)

$$P(K = 0) = \frac{1}{4} \text{ and } P(K = 1) = \frac{1}{2} \text{ and } P(K = 2) = \frac{1}{4}$$

- Why?
- Note that if once adds up $P(K = k)$ over all k in the support the sum should be one

$$\sum_k P(K = k) = 1$$

Exercise: Support and PMF

- we toss a biased coin twice and we count the number of heads (call it K)
- the probability that any toss lands a head is $\pi = \frac{1}{3}$
- What is the support of the distribution
- What is the PMF
- Repeat the last steps if π is any arbitrary number (between 0 and 1 of course)

Exercise: Support and PMF

- the support is as in the previous example $\{0, 1, 2\}$
- Why is it unchanged

$$P(K = 0) = \frac{4}{9} \text{ and } P(K = 1) = \frac{4}{9} \text{ and } P(K = 2) = \frac{1}{9}$$

- More generally

$$P(K = 0) = (1-\pi)^2 \text{ and } P(K = 1) = 2\pi(1-\pi) \text{ and } P(K = 2) = \pi^2$$

Flipping the coin

- Throughout this discussing we will consider flipping a coin
- The coin lands a head with probability π (could be biased) or tail with probability $1 - \pi$
- For convenience, we will recode H as 1 and T as 0
- We will flip it n times.
- Notation:
  - n is to denote the number of *trials*
  - On any trial (or flip), if we land an H we will call it an event (or success)
  - or if we land a T we will call it a failure
- RNA-seq connection: You can think of a read mapping to a gene to be an event

Three Variants of the Coin Tossing Experiment

1. Fix the number of trials (n) upfront and then toss the coin n times
   - The number of events (among n trials) is random
2. Toss the coin a large number of times and assume that each one of these many trials has a small probability of being an event
   - Here n is large and π is small (close to 0)
3. Fix the number of desired events upfront, then toss the coin repeatedly to achieve that number
   - Here the number of trials n is random

Example: Fixed n

- We flip the coin $n = 6$ times
- Observed sequence: TTHTTH
- We recode this as 001001
- This corresponds to
  - $n = 6$ trials
  - 2 events (or successes)
  - or equivalently 4 failures

Number of possible Outcomes

- Example 1: Suppose that $n = 2$
  - 4 possible outcomes: $\{00, 10, 01, 11\}$
  - $4 = 2 \times 2 = 2^2$
- Example 2: Suppose that $n = 3$
  - Eight possible outcomes: $\{000, 100, 101, 001, 110, 011, 101, 111\}$
  - $8 = 2 \times 2 \times = 2^3$
- The number of possible outcomes based on n trials is 2^n

Permutations of the integers 1 through n

- $n = 1 : \{1\}$
- $n = 2 : \{12, 21\}$
- $n = 3 : \{123, 132, 213, 231, 312, 321\}$
- The number of permutations of the integers $1, 2, 3, \ldots, n$ is $n!$
- We say n factorial

Factorial Function

- Integers are "whole" numbers $\ldots, -2, -1, 0, 1, 2, \ldots$
- Consider a non-negative integer k $(0, 1, 2, \ldots)$
- $0! = 1$
- $1! = 1$
- $2! = 2 \times 1 = 2$
- $3! = 3 \times 2 \times 1 = 6$
- $4! = 4 \times 3 \times 2 = 24$
- $\ldots$
- $k! = k \times (k-1) \times (k-2) \times \ldots 3 \times 2 \times 1$

Number of Permutations

- Example 1: Suppose that $n = 3$ and $k = 1$
  - We had 1 event among three trials
  - The three possible permutations are $\{001, 010, 100\}$
- Example 2: Suppose that $n = 4$ and $k = 2$
  - We had 2 events among four trials
  - The three possible permutations are $\{1100, 1010, 1001, 0011, 0101, 0110\}$
- What is the number of permutations for k events amomg n trials

Number of Permutations

- The number of possible permutations on the basis of k events among n trials

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

- Example 1: Suppose that $n = 3$ and $k = 1$

$$\binom{3}{1} = \frac{3!}{1!(2-1)!} = \frac{3 \times 2 \times 1}{1 \times 2 \times 1} = 3$$

```
choose(3,1)
```

```
## [1] 3
```

- Example 2: Suppose that $n = 4$ and $k = 2$

$$\binom{4}{2} = \frac{4!}{2!(4-2)!} = \frac{4 \times 3 \times 2 \times 1}{2 \times 1 \times 2 \times 1} = \frac{24}{4} = 6$$

```
choose(4,2)
```

```
## [1] 6
```

Bernoulli Distribution

- Suppose that we toss the coin just once
- In other words $n = 1$
- We say that the number of events follows a Bernoulli distribution with parameter π
- The distribution is

$$P[K = k] = \pi^k (1 - \pi)^{1-k}, k = 0, 1$$

```
set.seed(12324)
# Simulate 10 Bernoulli random variables with
# parameter pi=0.5
rbinom(10,1,0.5)
```

```
##  [1] 1 1 1 1 1 0 0 0 0 0
```

```
# Simulate 5 Bernoulli random variables with
# parameter pi=0.23
rbinom(5,1,0.23)
```

```
## [1] 0 0 0 0 0
```

Binomial Distribution

- For the Bernoulli distribution $n = 1$
- More generally (when $n \geq 1$) the number of events K is said to follow a Binomial distribution with parameters n and π
- The distribution is

$$P[K = k] = \binom{n}{k} \pi^k (1 - \pi)^{n-k},$$

$k = 0, 1, 2, \ldots, n$

- Note that when $n = 1$ the Binomial reduces to a Bernoulli distribution

```
set.seed(12324)
# Simulate 10 Binomial random variables with
# parameter n=2 and pi=0.5
rbinom(10,2,0.5)
```

```
##  [1] 1 2 2 1 2 0 0 1 1 1
```

```
# Simulate 5 Binomial random variables with
# parameter n=2 and pi=0.23
rbinom(5,2,0.23)
```

```
## [1] 0 1 0 0 0
```

Negative Binomial Distribution

- How many times do you have to flip a coin to get $r > 0$ events
- Model the number of *random* trials needed to get r events
- This distribution is called the negative binomial distribution
- The probability distribution is

$$P[K = k] = \binom{k + r - 1}{r - 1} \pi^r (1 - \pi)^k,$$

where $k = r, r + 1, r + 2, \ldots$

```
set.seed(13224)
# Simulate the number of trials needed to get k=5 events
rnbinom(10,5,0.1)
```

```
##  [1] 63 60 56 30 64 62 36 36 44 37
```

Poisson Distribution

- The number of rare events (π is small) among this large number of trials follows a Poisson distribution
- The probability distribution is

$$P[K = k] = \frac{e^{-\lambda}\lambda^k}{k!},$$

where $k = 0, 1, 2, \ldots$

```
set.seed(13224)
# Simulate 10 Poisson variates with m
rpois(10,0.1)
```

```
##  [1] 0 1 0 0 0 0 1 0 0 0
```

Relationship between Binomial and Poisson Distribution

- Consider tossing the coin a large number of times

```
n=1000000
p=1/n
```

- Note that we have $n = 10^6$ trials with a low success probability of $p = 10^{-6}$
- The expected number of events among these 10^6 trials is $n \times p = 1$. Why?
- Now simulate 99999 numbers from this binomial distribution

```
set.seed(9988)
x=rbinom(B9,n,p)
length(x)

## [1] 99999
```

- What is the expected number of events (i.e., the expected number of events (among n trials) across $B = 99999$ simulations)?

```
mean(x)

## [1] 1.00055
```

Relationship between Binomial and Poisson Distribution

- Now compare the empirical distributions to the Poisson distributions

```
round(dpois(0:7,lambda=1),3)

## [1] 0.368 0.368 0.184 0.061 0.015 0.003 0.001 0.000

round(table(x)/B9,3)

## x
##     0     1     2     3     4     5     6     7
## 0.367 0.369 0.183 0.061 0.016 0.003 0.000 0.000
```

Mean and Variance of Negative Binomial

- A negative binomial distribution can be parameterized in terms of
  - r and p
  - or μ and σ^2
  - or μ and a dispersion parameter α (more on this later)
- The relationship between these two parametrizations is given by

$$\mu = r\frac{1-p}{p} \text{ and } \sigma^2 = r\frac{1-p}{p^2},$$

and

$$p = \frac{\mu}{\sigma^2} \text{ and } r = \frac{\mu^2}{\sigma^2 - \mu}$$

- If you provide r and p, you can calculate μ and σ^2
- Or, if you provide μ and σ^2, you can recover r and p.

Negative Binomial PMF in terms of μ and α

- The NB PMF parametrized in terms of p and r (the number of events) is

$$P[K = k] = \binom{k + r - 1}{r - 1} \pi^r (1 - \pi)^k,$$

  where $k = r, r + 1, r + 2, \ldots$
- The NB PMF parametrized in terms of the mean μ and the dispersion parameter α is

$$P[K = k] = \frac{\Gamma[k + \alpha^{-1}]}{\Gamma[\alpha^{-1}]\Gamma[k + 1]} \left(\frac{1}{1 + \mu\alpha} \right)^{\alpha^{-1}} \left(\frac{\mu}{\alpha^{-1} + \mu} \right)^k,$$

  where $k = 0, 1, \ldots$
- The variance is $\mu(1 + \alpha\mu)$
- As α shrinks to 0 (no-dispersion), the distribution becomes Poisson

Means and Variances

Distribution	Support	Mean	Variance
Bernoulli(π)	0,1	π	$\pi(1-\pi)$
Binomial(n, π)	$0, 1, \ldots, n$	$n\pi$	$n\pi(1-\pi)$
Poisson(λ)	$0, 1, 2, \ldots,$	λ	λ
NB(p, r)	$r, r+1, r+2, \ldots,$	$r\frac{1-p}{p}$	$r\frac{1-p}{p^2}$

Section 8

Logistic Regression

Linear Regression Example: Gene Expression

- Consider the simple linear regression model

$$Y = \beta_0 + \beta_1 x + \epsilon,$$

  where
  - $x = 0$ (untreated)
  - or $x = 1$ (treated)
- Y is the observed "expression" of the gene
- ϵ is the measurement noise term
- We assume that it follows a normal distribution with mean 0 and variance σ^2

Reminder: Important Fact about Normal Distribution

- Consider a normal distribution with mean 0 and standard deviation σ
- If the data are shifted by a constant μ, then
  1. resulting distribution remains normal
  2. The mean of the new distribution is $\mu + 0 = \mu$
  3. Its standard deviation remains unchanged
- The last two (but not first) property are true for any distribution
- Recall $Y = \beta_0 + \beta_1 x + \epsilon$
- Y follows a normal distribution with mean $\mu = \beta_0 + \beta_1 x$ and variance σ^2
- IMPORTANT: μ depends on x (unless of course $\beta_1 = 1$)

Linear Regression Example: Interpretation

- Model
$$Y = \beta_0 + \beta_1 x + \epsilon,$$
- The goal of (mean) regression is to estimate the expected value of Y given treatment status
- Conditional on $x = 0$ (i.e., not receiving treatment), the expected value of Y is
$$\beta_0 + \beta_1 \times 0 = \beta_0$$
- Conditional on $z = 1$ (i.e., receiving treatment), the expected value of Y is
$$\beta_0 + \beta_1 \times 1 = \beta_0 + \beta_1$$

Linear Regression Example: Interpretation

- Model
$$Y = \beta_0 + \beta_1 x + \epsilon,$$
- β_0 (the intercept) is the expected value of Y if no treatment is administered (average baseline value)
- β_1 is the treatment effect
- If treatment is administered, the expected value of expression is
  - increased by β_1 units if $\beta_1 > 0$
  - decreased by β_1 units if $\beta_1 < 0$
  - unchanged if $\beta_1 = 0$

Regression for Binary Outcomes

- Suppose that Y is a binary outcome
- It assumes values 0 or 1
- Consider the previous model

$$Y = \beta_0 + \beta_1 x + \epsilon,$$

- Is it appropriate? Why or why not?

Logistic Regression

- Relate the probability of the outcome of the event $Y = 1$ to treatment
- More specifically, relate the log-odds to the treatment
- The log-odds will be modeled as a linear function of x

$$\beta_0 + \beta_1 x + \epsilon$$

- This is an example of a generalized linear model
- The expected outcome of Y is not modeled directly as a linear function
- A transformation of the expected outcome of Y is modeled as a linear function

Expected value of a binary event

- Suppose that Y assumes 1 with probability π or 0 with probability $1 - \pi$
- $P(Y = 1) = \pi$ and $P(Y = 0) = 1 - \pi$
- IMPORTANT: $P(Y = 1) = E(Y)$
- The expected value of Y is the probability that it assumes the value 1
- Why?

Odds vs Probability

- Suppose that $\pi = P(Y = 1)$
- The odds of the event $Y = 1$ (to occur) is defined as

$$\text{Odds}[Y = 1] = \frac{\text{Probability that } Y = 1 \text{ occurs}}{\text{Probability that } Y = 1 \text{ does not occur}} = \frac{\pi}{1 - \pi}$$

Odds Ratio Versus Relative Risk

- $\pi_0 = P[Y = 1|X = 0]$: Probability that the event occurs if sample is not treated
- $\pi_1 = P[Y = 1|X = 1]$: Probability that the event occurs if $X = 1$sample is treated
- The odds-ratio is

$$\mathrm{OR} = \frac{\frac{\pi_1}{1-\pi_1}}{\frac{\pi_0}{1-\pi_0}}$$

- The relative risk is

$$\mathrm{RR} = \frac{\pi_1}{\pi_0}$$

The Logistic Model

- The log-odds of the event $Y = 1$

$$\log \frac{P(Y = 1|X = x)}{1 - P(Y = 1|X = x)} = \beta_0 + \beta_1 x$$

- or equivalently

$$\log \frac{E(Y|X = x)}{1 - E(Y|X = x)} = \beta_0 + \beta_1 x$$

- Recall that in the simple linear regression case, we assumed that

$$E[Y|X = x] = \beta_0 + \beta_1 x$$

Link Function

- For a probability π, define the "logit" transformation as

$$\log \frac{\pi}{1-\pi}$$

- This is the log-odds of an event with probability π
- Note that in the logistic model, the probability of the event is linear in the parameter through this logit transformation

$$\log \frac{\pi}{1-\pi} = \beta_0 + \beta_1 x$$

- In the GLM literature, this is called the link function

Overdispersion

- Recall that if K follows a binomial distribution with parameters n and π, then
  - mean $\mu = n\pi$
  - variance $\sigma^2 = n\pi(1-\pi)$
- Clustering in the data results in the actual variance to be different than the nominal variance ($n\pi(1-\pi)$)
  - Overdispersion: Actual variance is larger than nominal variance
  - Underdispersion: Actual variance is smaller than nominal variance
- The choice of a GLM and evaluation of its performance *should* start and end with considering/addressing the overdispersion issue
- The use of Poisson and Negative Binomial models are two common choices for GLM for overdispersed data

Generalized Linear Models (GLM)

Define $\mu_x = E(Y|X = x)$ as the expected value of the outcome given treatment status ($x = 0$ or $x = 1$)

Distribution	Support	Link	Mean
Binomial	$0, 1, \ldots, n$	$\beta_0 + \beta_1 x = \log \frac{\mu_x}{1-\mu_x}$	$\mu_x = \frac{\exp(\beta_0+\beta_1 x)}{1+\exp(\beta_0+\beta_1 x)}$
Poisson	$0, 1, 2, \ldots$	$\beta_0 + \beta_1 x = \log(\mu_x)$	$\mu_x = \exp(\beta_0 + \beta_1 x)$
Negative Binomial	$r, r+1, \ldots$	$\beta_0 + \beta_1 x = \log(\mu_x)$	$\mu_x = \exp(\beta_0 + \beta_1 x)$

Section 9

Negative Binomial GLM for RNA-Seq

General Note

- Recall the simple linear regression model for expression

$$Y = \beta_0 + \beta_1 x + \epsilon,$$

  where
  - $x = 0$ (untreated)
  - or $x = 1$ (treated)
- Y is the observed "expression" of the gene
- ϵ is the measurement noise term
- The parameter of interest is β_1 (the treatment effect)
- There are two other unknown parameters, β_0 and σ^2 the estimation procedure has to deal with in a *principled* manner
- β_0 and σ^2 are *nuisance* parameters
- They are not of primary (or any) interest. But you have to deal with them!

General Hypothesis

- Is the RNA abundance level for any of the m genes affected by treatment
- Let H_j denote the null hypothesis for gene j
- H_j: The RNA abundance level for gene j is not affected by treatment
- $\bar{H}_j$: The RNA abundance level for gene j is affected by treatment
- The global null hypothesis: H_1 and H_2 and .... and H_m are all true
- The global alternative: $\bar{H}_1$ or $\bar{H}_2$ or .... or $\bar{H}_m$ is true
- In other words, under the alternative at least one of the marginal null hypotheses is false

Observed Data

- Some notation
  - n denotes the number of samples
  - m denotes the number of genes
  - K_{ij} denotes the *observed* number of reads mapped to gene i for sample j
  - $x_j = 0$ or 1 denotes the treatment status for sample j
- What is observed for sample j is the vector

$$K_{1j}, \ldots, K_{mj}, x_j$$

- In other words m counts (one per gene) and the experimental factor
- Note that the K_{ij} form a table of counts of dimension $n \times m$ (n samples and m genes)

DESeq: Notation for Negative Binomial Distribution

- The count K is assumed to follow a negative binomial distribution with parameters $p \in (0, 1)$ and $r > 1$
- The distribution is PMF is

$$P(K = k) = \binom{k + r - 1}{r - 1} p^r (1 - p)^k,$$

for $k = r, r + 1, \ldots$

- Rather than considering the model as $\mathrm{NB}[p, r]$ we will consider it as $\mathrm{NB}[\mu, \alpha]$, where

$$P[K = k] = \frac{\Gamma[k + \alpha^{-1}]}{\Gamma[\alpha^{-1}]\Gamma[k + 1]} \left(\frac{1}{1 + \mu\alpha} \right)^{\alpha^{-1}} \left(\frac{\mu}{\alpha^{-1} + \mu} \right)^k,$$

where $k = 0, 1, \ldots$

DESeq: Notation

- K_{ij} denotes the *observed* number of reads mapped to gene i for sample j
- K_{ij} follows a negative binomial distribution with
  - Mean μ_{ij} (indexed by gene i *and* sample j)
  - Dispersion parameter α_i (indexed by the gene i)
- The mean is assumed to be $\mu_{ij} = s_j q_{ij}$ where
  - $\log q_{ij} = \beta_{i0} + \beta_{i1} x_j$
  - s_j is a gene j specific normalization constant

DESeq: Reformulate Hypotheses

- Hypotheses of interest
  - The global null hypothesis: H_1 and H_2 and .... and H_m are all true
  - The global alternative: $\bar{H}_1$ or $\bar{H}_2$ or .... or $\bar{H}_m$ is true
- Reformulation
  - The global null hypothesis: $\beta_{11} = 0$ and $\beta_{21} = 0$ and .... and $\beta_{m1} = 0$
  - In other words, all of the β_{j1} are equal to zero
  - The global alternative: $\beta_{11} \neq 0$ or $\beta_{21} = 0$ or .... or $\beta_{m1} = 0$
  - In other words, at least one of the β_{j1} is not equal to zero

DESeq: Assumption on Distribution

K_{ij} follows a negative binomial distribution with mean μ and dispersion parameter α

DESeq: Assumption on Mean of Distribution

- Conditional on the treatment status of sample j ($x_j = 0$ or 1), the expected value of K_{ij} is

$$\mu_{ij} = s_j \times q_{ij}$$

where

$$\log q_{ij} = \beta_{i0} + \beta_{i1} x_j$$

- Note that two regression parameters are indexed by i
- Why? Because these are gene i specific parameters
- Why is x_j not indexed by i?
- Final Assumption: $s_{ij} = s_j$
- In other words: Within sample j, the normalization parameter is constant across the genes
- How many assumptions so far?

DESeq: Main parameters and Nuisance Parameters

- The m main parameters of interest

$$\beta_{11}, \ldots, \beta_{m1}$$

- The unknown nuisance parameters are
  - The m gene specific intercepts

$$\beta_{10}, \ldots, \beta_{m0}$$

  - the n sample specific normalization constants

$$s_1, \ldots, s_n$$

  - The m gene specific nuisance parameters

$$\alpha_1, \ldots, \alpha_m$$

DESeq: Main parameters and Nuisance Parameters

- Assuming the model assumptions are correct, the estimation of the regression parameters β_{i0}, β_{i1} is fairly straightforward
- The DESeq authors propose to estimate the normalization constant for sample j as

$$s_j = \text{median} \frac{K_{ij}}{K_i^R},$$

where

$$K_i^R = \left(\prod_{j=1}^{m} K_{ij} \right)^{\frac{1}{m}}$$

- Here K_i^R is the geometric mean of $K_{i1}, \ldots, K_{in}$ (the n counts for gene i)
- The median is taken over all m genes for which K_i^R is positive

DESeq: Dispersion parameter

- A key issue in using the NB model is proper handling of the gene specific dispersion parameters

$$\alpha_1, \ldots, \alpha_m$$

- The estimation of the dispersion parameter is a challenging task
- DESeq2 assumes that α_i is random following a normal distribution
- The results are sensitive to the estimates
- One of the key differences between DESeq2 and DESeq is the approach taken to estimate these nuisance parameters

DESeq Software Overview

- The analysis of RNA-Seq data using the `DESeq2` package will be reviewed in detail in the upcoming weeks
- The estimation and inference for the model is done through the `DESeq` function
- It performs the following steps in the order give
  1. estimation of size factors $s_1, \ldots, s_n$
  2. estimation of dispersion parameters $\alpha_1, \ldots, alpha_m$
  3. Fit NB GLM model

DESeq: Model Exercise

- K_{ij} denotes the *observed* number of reads mapped to gene i for sample j
- $x_j = 0$ or 1 denotes the treatment status for sample j
- Say we want to account for another covariate z_j (e.g., temperature)
- What is observed for sample j is the vector

$$K_{1j}, \ldots, K_{mj}, x_j, z_j$$

- Questions
  - State the hypotheses
  - Propose a model (that incorporates the additional covariate)
  - List any assumptions that you have made

DESeq: Model Exercise

- The null hypothesis
  $H_0 : \beta_{11} = 0$ and $\beta_{21} = 0$ and $\ldots \beta_{m1} = 0$
- Conditional on x_j and z_j, the observed number of reads mapped to gene i for sample j, K_{ij}, follows a negative binomial distribution with
  - Mean μ_{ij}
  - Dispersion parameter α_i (gene specific)
- Conditional on the treatment status of sample j ($x_j = 0$ or 1) and the temperature z_j, the expected value of K_{ij} is

$$\mu_{ij} = s_j \times q_{ij}$$

  where

$$\log q_{ij} = \beta_{i0} + \beta_{i1} x_j + \beta_{i2} z_j$$

- The normalization parameters are assumed to be sample (not gene) specific ($s_{ij} = s_j$)

DESeq: Model Nuisance Parameter

- The m main parameters of interest

$$\beta_{11}, \ldots, \beta_{m1}$$

- The unknown nuisance parameters are
  - The m gene specific intercepts

$$\beta_{10}, \ldots, \beta_{m0}$$

  - The m gene specific coefficients for the new covariate

$$\beta_{12}, \ldots, \beta_{m2}$$

  - the n sample specific normalization constants

$$s_1, \ldots, s_n$$

  - The m gene specific nuisance parameters

$$\alpha_1, \ldots, \alpha_m$$

edgeR: Another NB Model for RNA-Seq Counts

- Assume that the K_{ij} follows a NB distribution with mean μ_{ij} and dispersion paramater α_i
- The mean (conditional on treatment status x) is

$$\mu_i j = M_j p_{xi}$$

where

  - M_j is the library size (total number of reads for sample j
  - p_{xi} is the relative abudance of the gene i given treatment status x
    - p_{0i} is the relative abudance of the gene i given no treatment
    - p_{1i} is the relative abudance of the gene i given treatment
- Treatment changes the abudance of RNA in gene i if $p_{0i} \neq p_{1i}$
- This is same distributional assumption as in DESeq

MLE Illustration

- In a GLM, the parameters β_{i0} and β_{i1} are estimated using the method of Maximum likelihood (MLE)
- We illustrate the method using this coin tossing example:
- We toss a coin once and record the number of heads
- Suppose that you conduct two independent replicates of this experiment
- K_1 the number of events (among $n = 1$ trial) in experiment 1
- K_2 the number of events (among $n = 1$ trial) in experiment 2
- The PMF of K_1 is

$$P(K_1 = k) = \pi^k (1 - \pi_k)^{1-k}$$

- The PMF of K_1 is

$$P(K_2 = k) = \pi^k (1 - \pi_k)^{1-k}$$

- Here $k = 0$ or 1

Joint Distribution

- Repeat the experiment B times
- The joint PMF is

$$P(K_1 = k_1, \ldots, K_B = k_B) = \pi^{k_1}(1-\pi)^{1-k_1} \times \ldots \times \pi^{k_B}(1-\pi)^{1-k_B}$$

- Note that the implicit assumption is that the experiments are mutually independent
- Under this assumption, the joint PMF is the product of the marginal PMFs
- Plugging in the *observed* counts into the joint PMF yields the likelihood function

Binomial Example: Observed data

```
set.seed(2131)
x=rbinom(5,1,0.5)
x
```

```
## [1] 1 0 0 0 1
```

- Observed data $x_1 = 1$, $x_1 = 0$, $x_3 = 0$, $x_4 = 0$ and $x_5 = 1$
- What is the likelihood?

Binomial Example: Likelihood

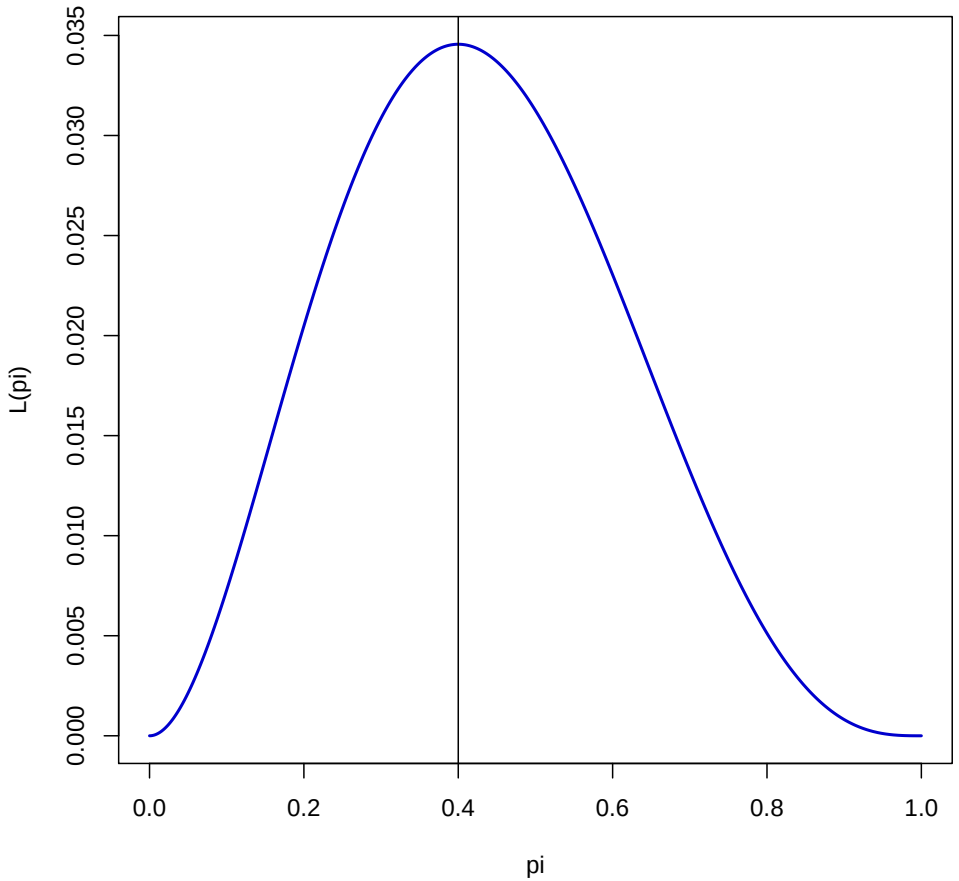
- Observed data $x_1 = 1$, $x_1 = 0$, $x_3 = 0$, $x_4 = 0$ and $x_5 = 1$
- The likelihood

$$\begin{aligned} L[\pi] &= \pi^{x_1}(1-\pi)^{x_1} \times \pi^{x_2}(1-\pi)^{x_2} \times \pi^{x_3}(1-\pi)^{x_3} \times \\ &\quad \pi^{x_4}(1-\pi)^{x_4} \times \pi^{x_5}(1-\pi)^{x_5} \times \\ &= \pi^{1}(1-\pi)^{1-1} \times \pi^{0}(1-\pi)^{1-0} \times \pi^{0}(1-\pi)^{1-0} \times \\ &\quad \pi^{0}(1-\pi)^{1-0} \times \pi^{1}(1-\pi)^{1-1} \\ &= \pi^{2}(1-\pi)^{3} \end{aligned}$$

- Given the observed data find the value of π that maximizes this probability

Binomial Example: Maximum Likelihood

The maximum value of the function $L[\pi] = \pi^2(1-\pi)^3$ occurs at $\pi = 0.4$.



Maximum Likelihood Calculation for NB

- For gene i, let $k_{11}, \ldots, k_{1n}$ the n observed counts
- For patient j plug the observed count k_{ij} into the PMF of the NB distribution $f[k_{ij}; \mu_{ij}; \alpha_i]$
- Write the likelihood function as a product of these n terms

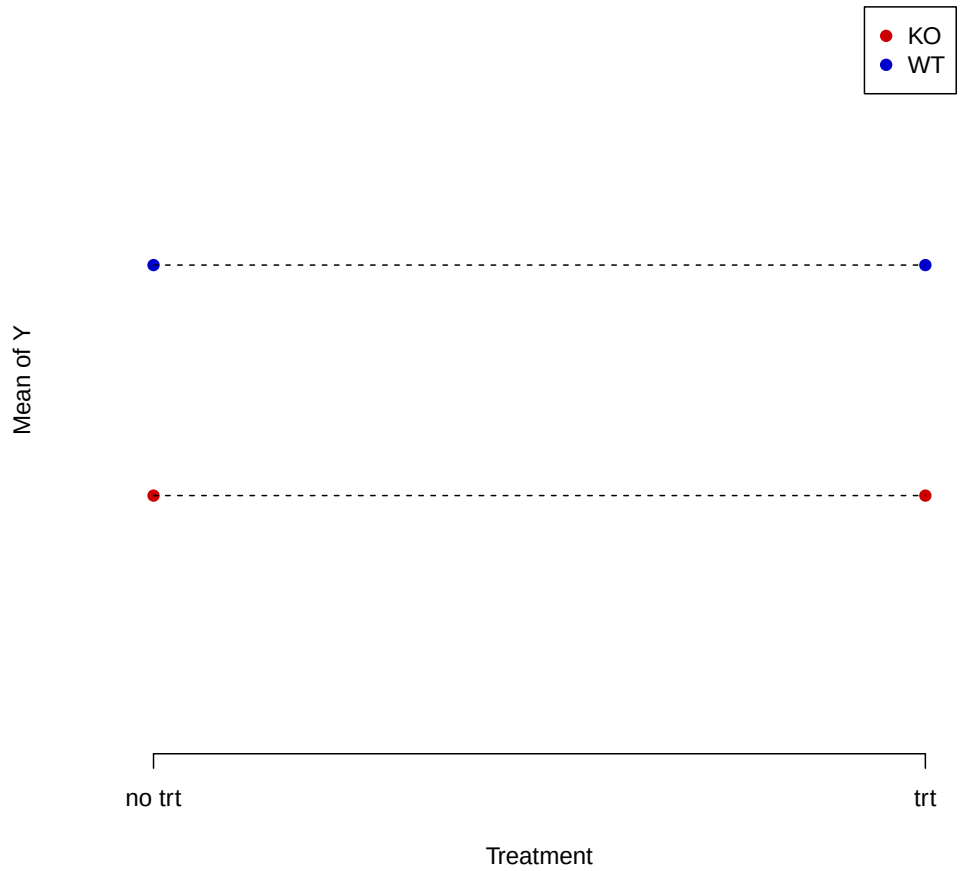
$$L = \prod_{j=1}^{n} f[k_{ij}; \mu_{ij}; \alpha_i] = f[k_{ij}; \beta_{0i}, \beta_{1i}, s_j, \alpha_i]$$

- The function depends on $\beta_{0i}, \beta_{1i}, s_j$ and α_i
- One approach: Come up with some estimates of s_j and α_i and plug them into the likelihood
- Pretend that these are the *true* values
- Now the likelihood is only a function of β_{0i} and β_{1i}

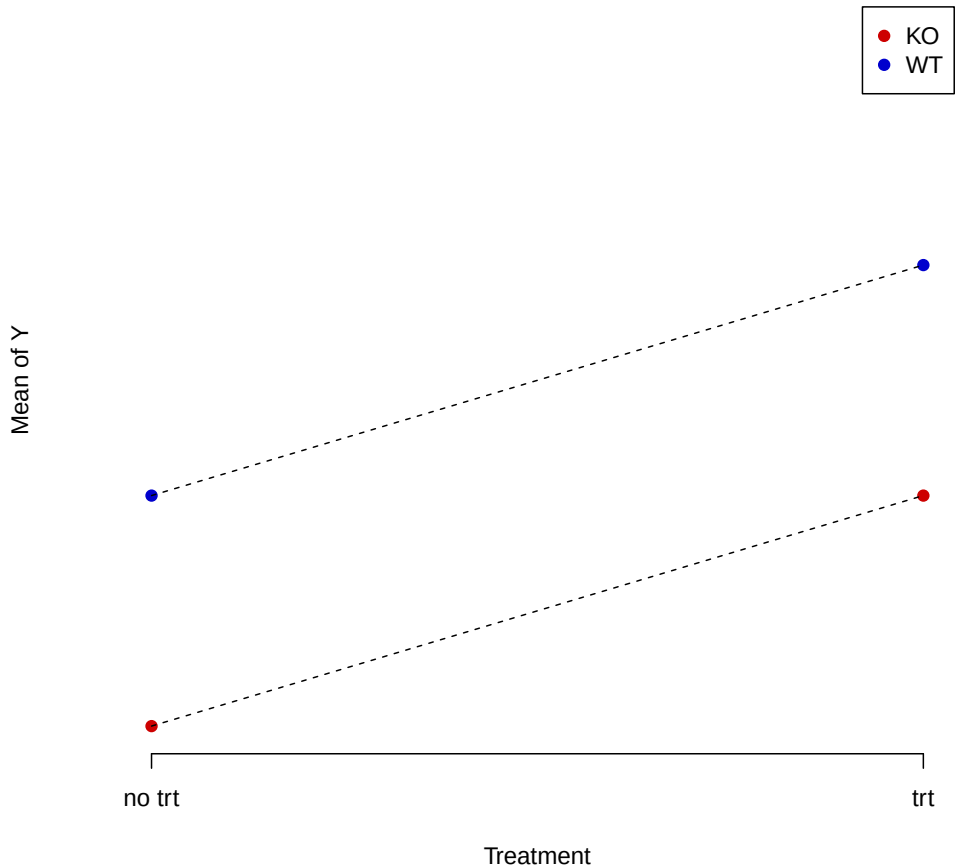
Section 10

Interaction versus Additive Effects

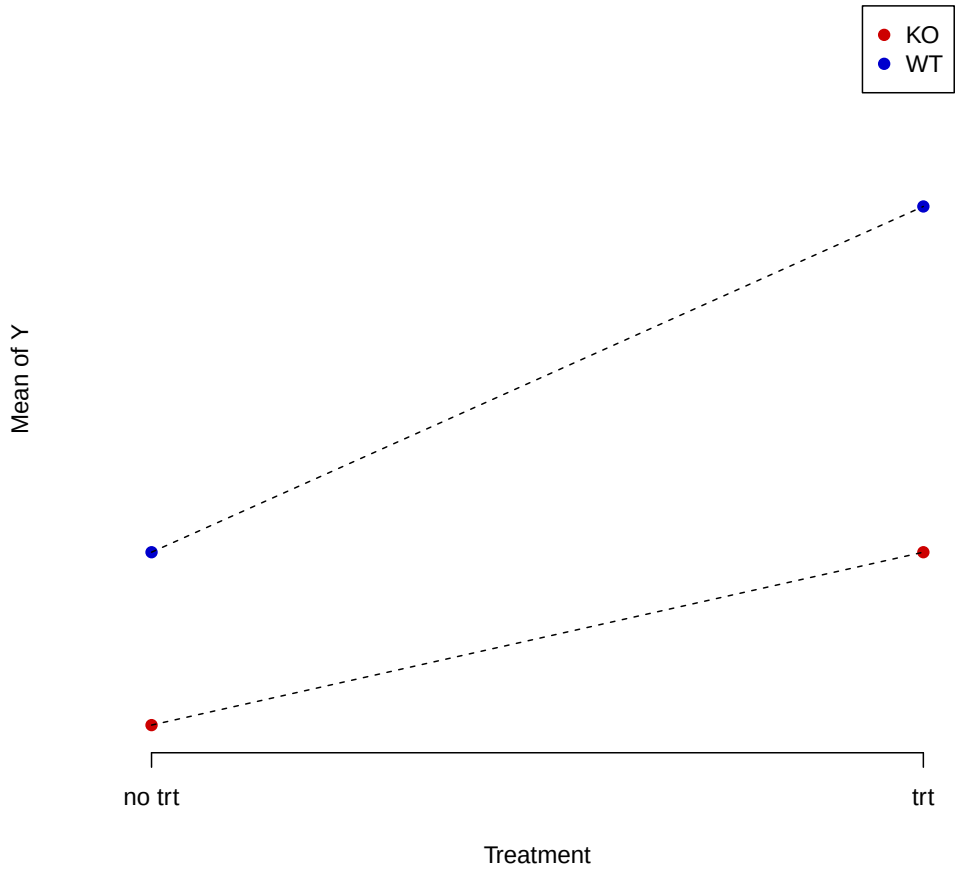
Example 1: No Interaction



Example 2: No Interaction



Example 3: Interaction



Model Interaction

- Y denotes the gene expression
- Let x denote the treatment indicator
  - $x = 0$ if not treated or 1 if treated
- Let z denote the knock-out indicator
  - $z = 0$ is WT or 1 otherwise
- The expected value of Y given treatment indicator x and knock out indicator z is denoted by

$$\mu_{x,z} = E[Y|X = x, Z = z]$$

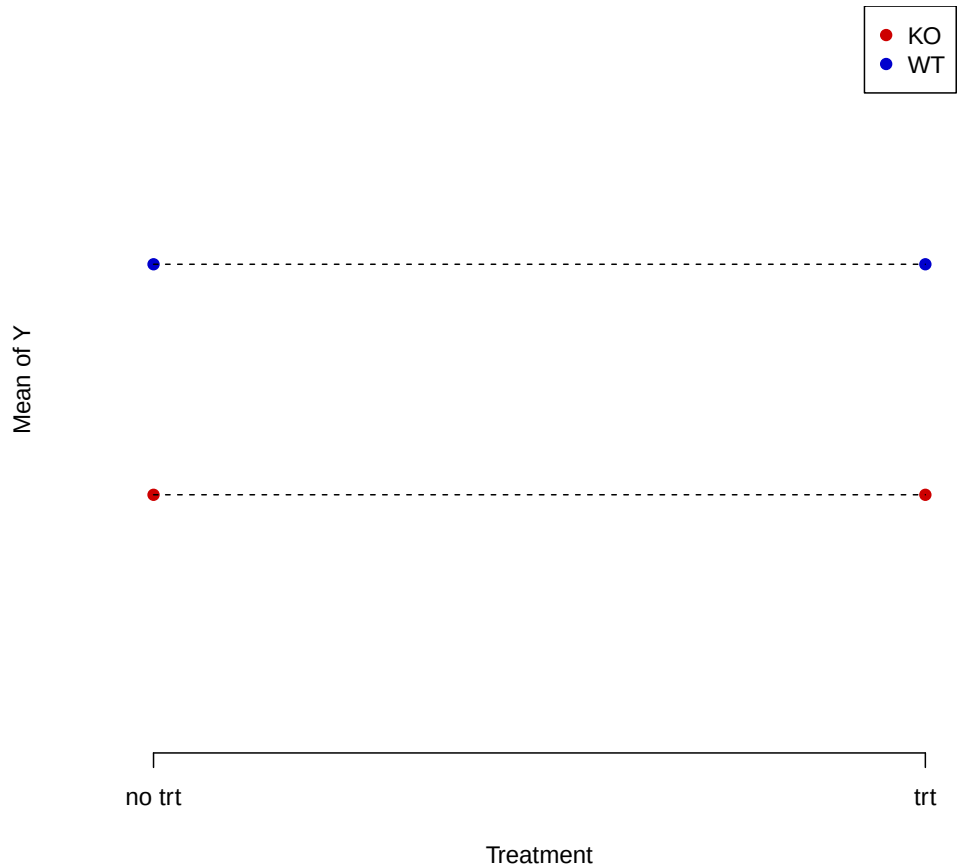
- The model will be

$$Y = \mu_{x,z} + \epsilon$$

where ϵ is a the measurement error

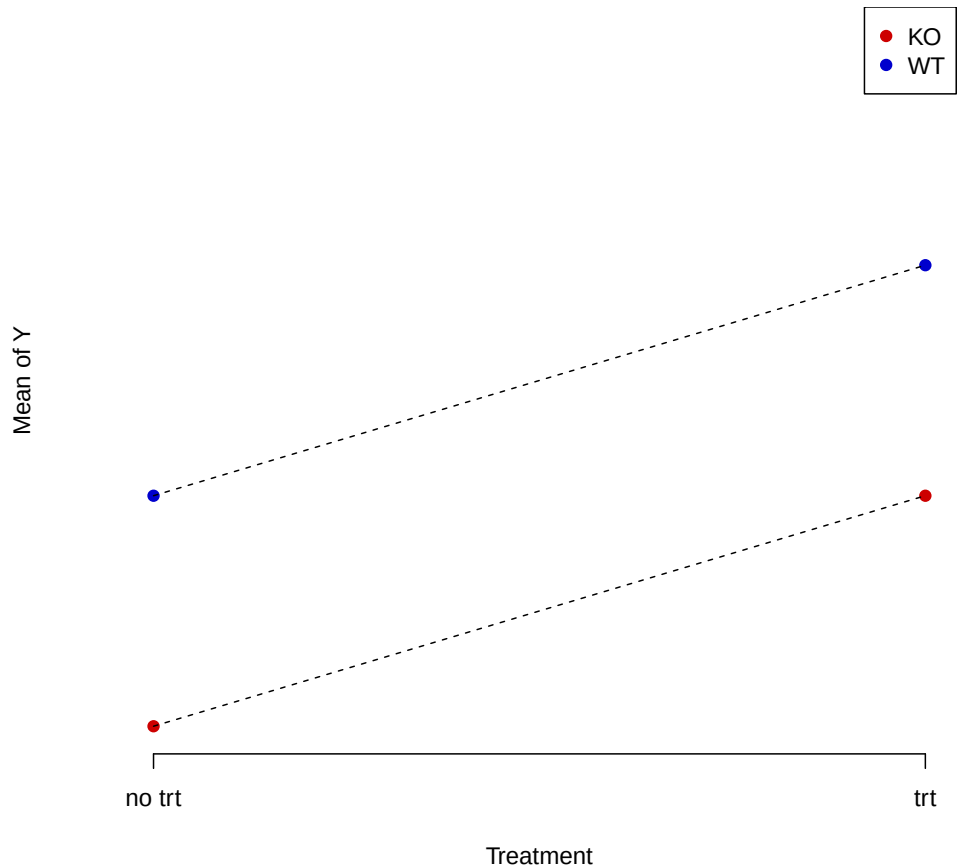
Example 1: Linear Model for No Interaction

$$Y = \beta_0 + \beta_1 z + \epsilon \ (\beta_2 = 0)$$



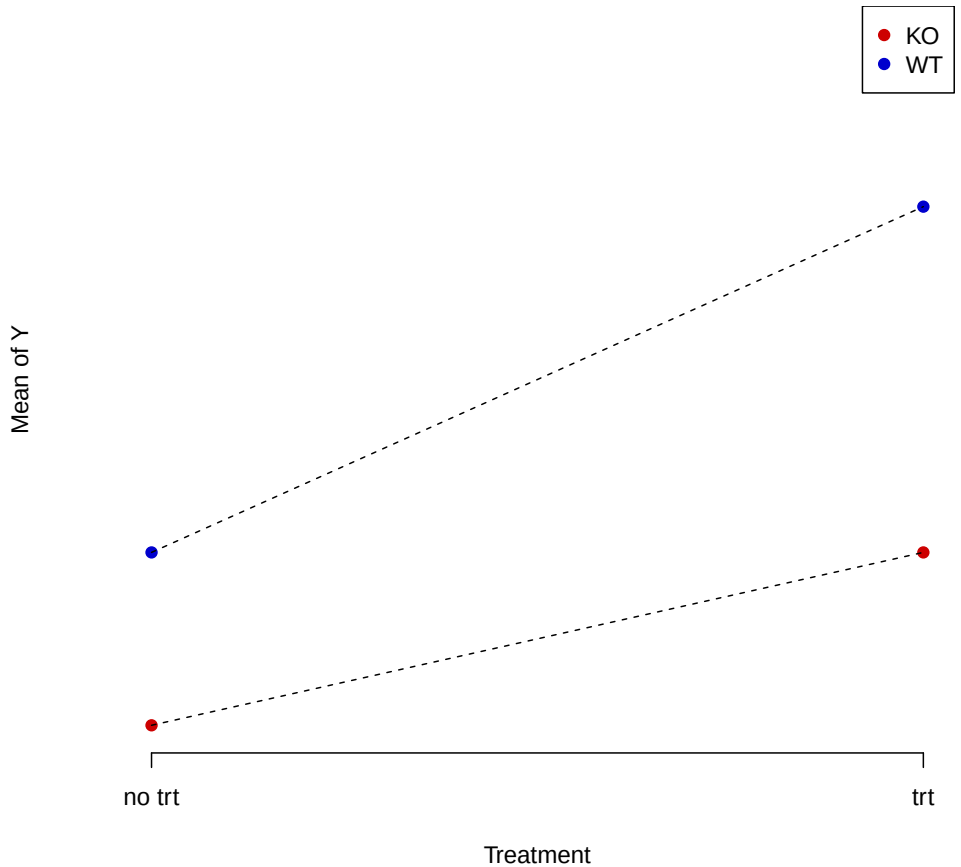
Example 2: No Interaction

$$Y = \beta_0 + \beta_1 z + \beta_2 x + \epsilon$$



Example 3: Interaction

$$Y = \beta_0 + \beta_1 z + \beta_2 x + \beta_3 xz + \epsilon$$



Interaction Examples

- Example 1: What are the signs for β_0 and β_1?
- Example 2: What are the signs for β_0, β_1 and β_2?
- Example 2: What are the signs for $\beta_0, \beta_1, \beta_2$ and β_3?

Incorporating Interactions into the NB Model

- Conditional on x_j and z_j, the observed number of reads mapped to gene i for sample j, K_{ij}, follows a negative binomial distribution with
  - Mean μ_{ij}
  - Dispersion parameter α_i (gene specific)
- Conditional on the treatment status of sample j ($x_j = 0$ or 1) and the temperature z_j, the expected value of K_{ij} is

$$\mu_{ij} = s_j \times q_{ij}$$

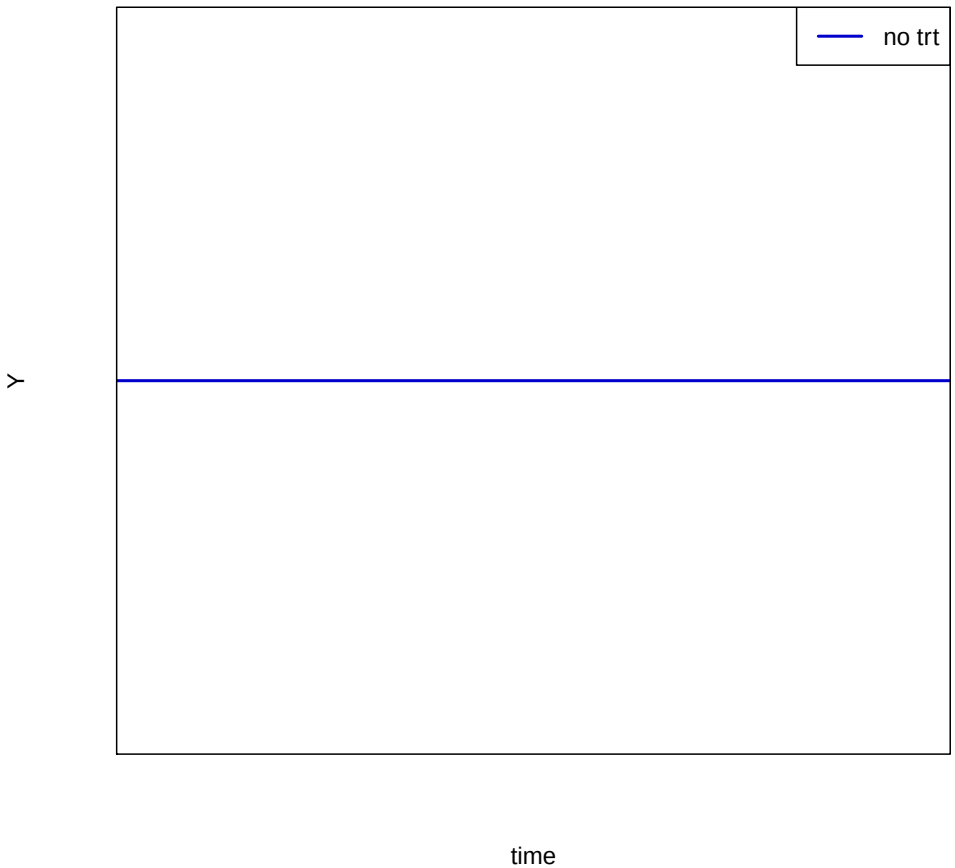
  where

$$\log q_{ij} = \beta_{i0} + \beta_{i1} x_j + \beta_{i2} z_j + \beta_{i3} x_j z_j$$

- The normalization parameters are assumed to be sample (not gene) specific ($s_{ij} = s_j$)

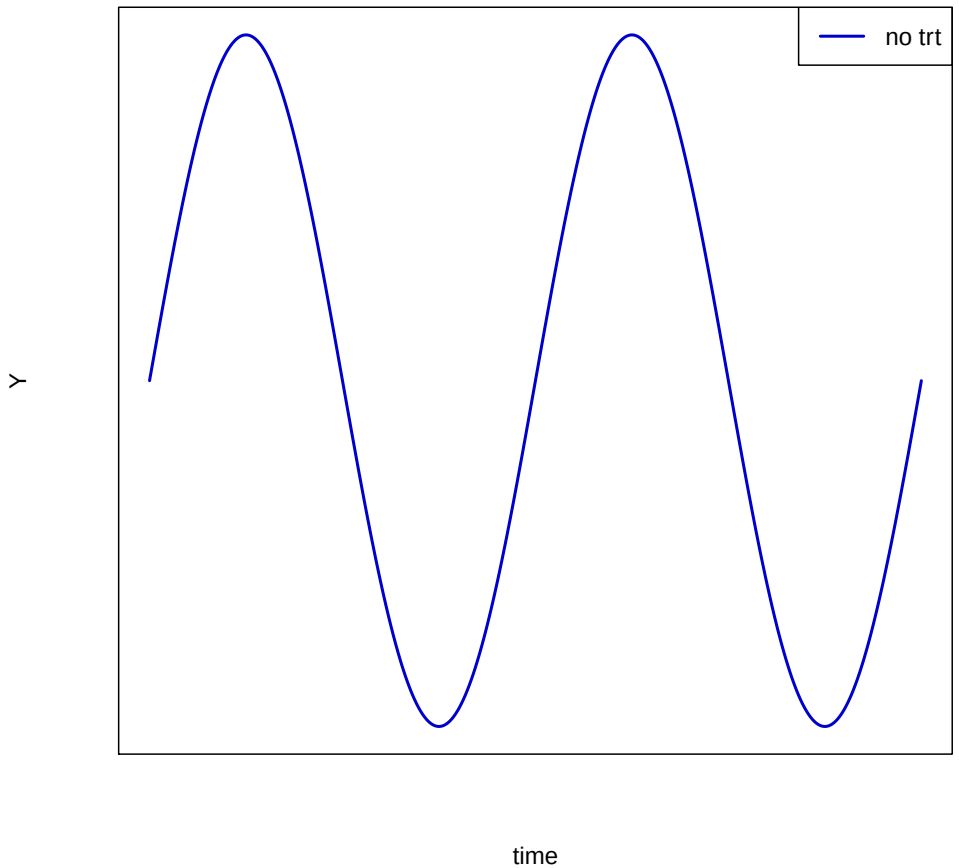
Example 1: No Time Course Effect

There is no time-course effect



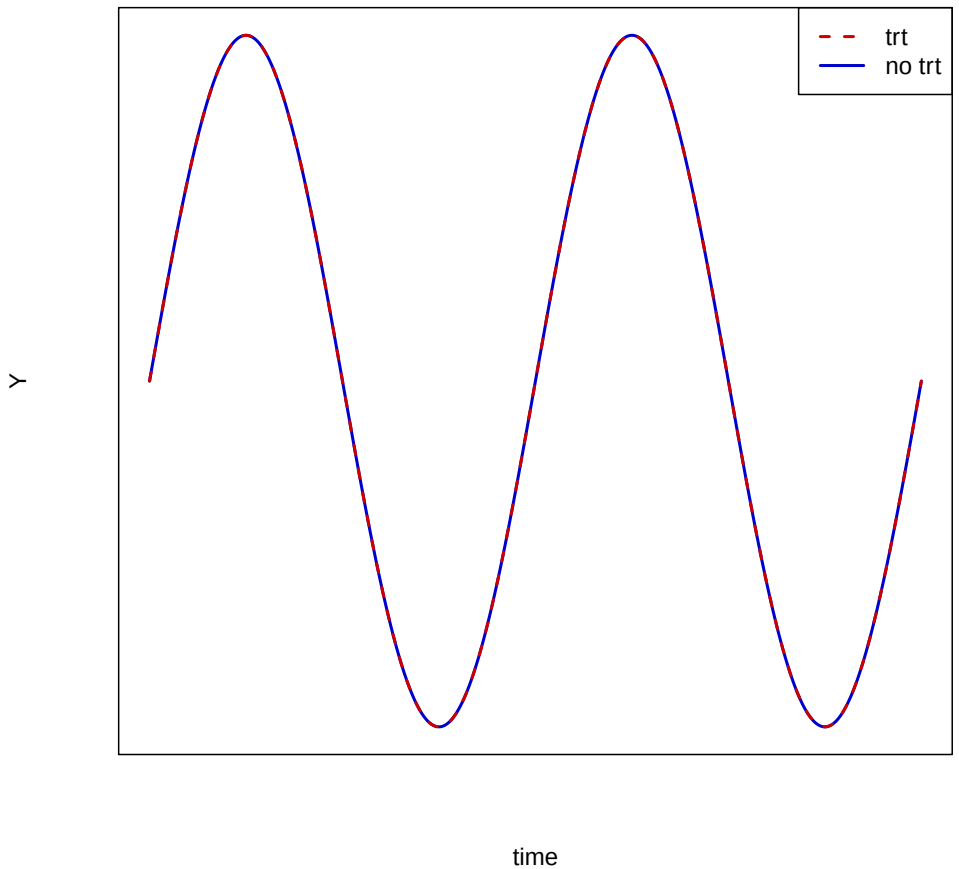
Example 2: Time Course Effect

There is a time course effect



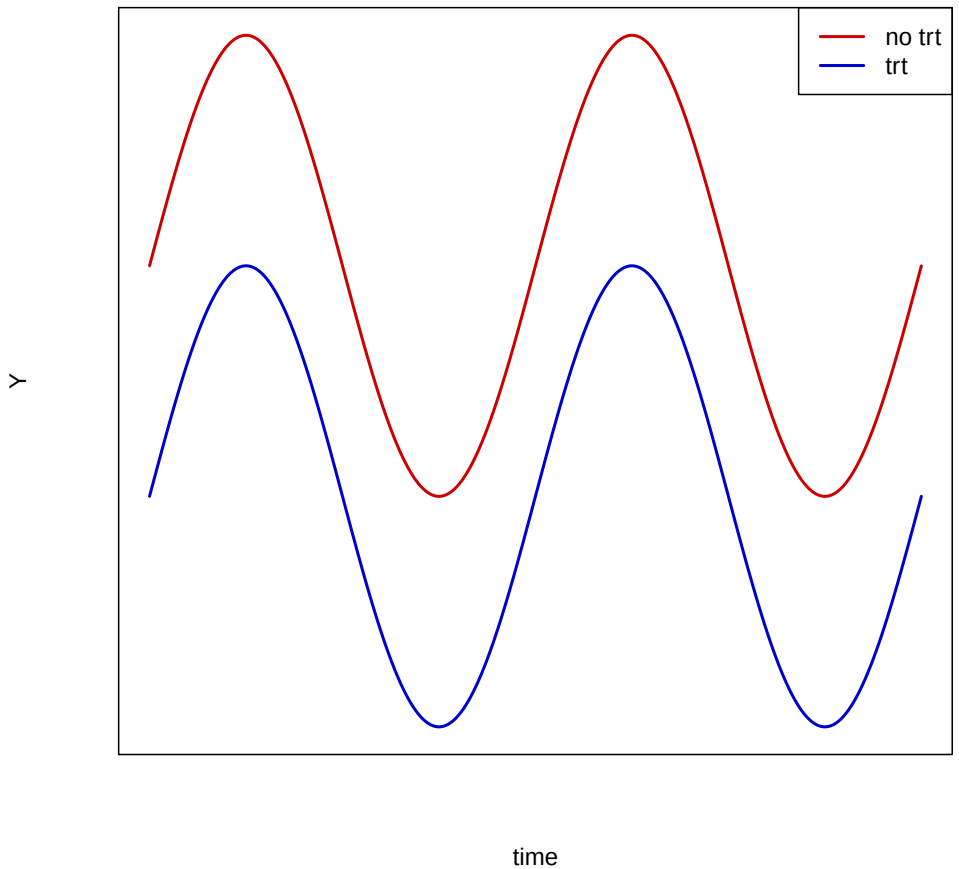
Example 3: Time Course Effect

There is a time-course effect within each condition but not time-course effect across conditions. Is this interesting?



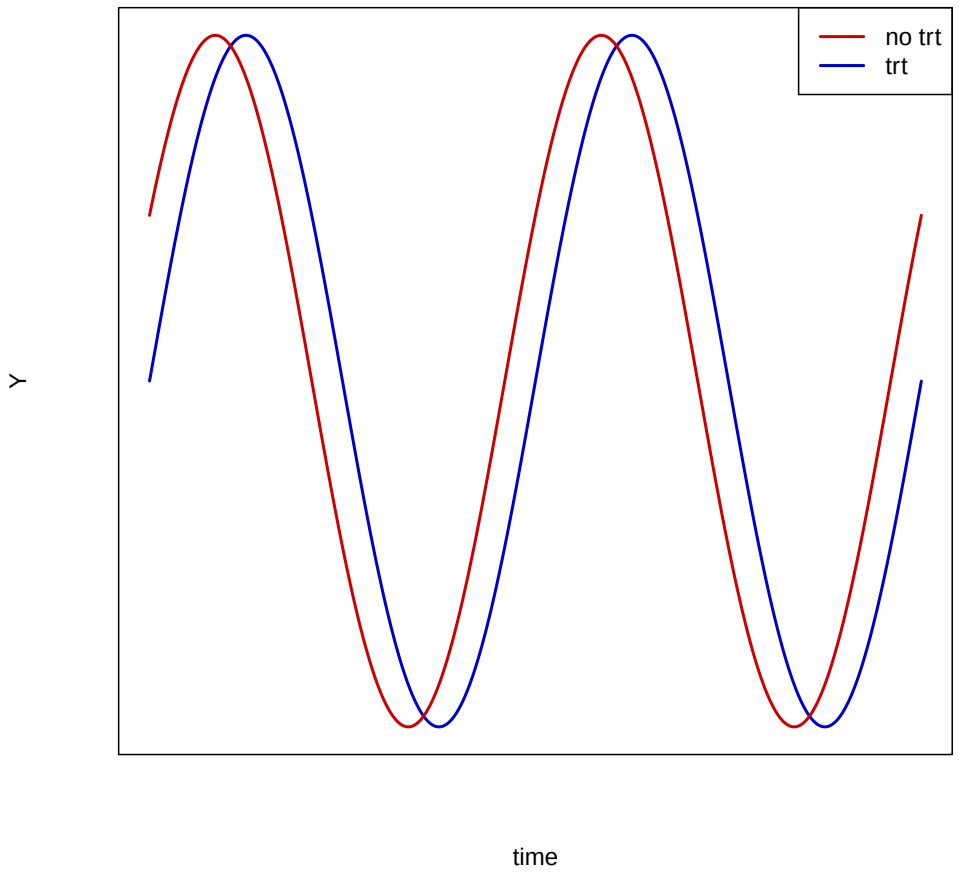
Example 4: Time Course Effect

There is a time-course effect within each condition and a vertical shift with respect to treatment. Is this interesting?

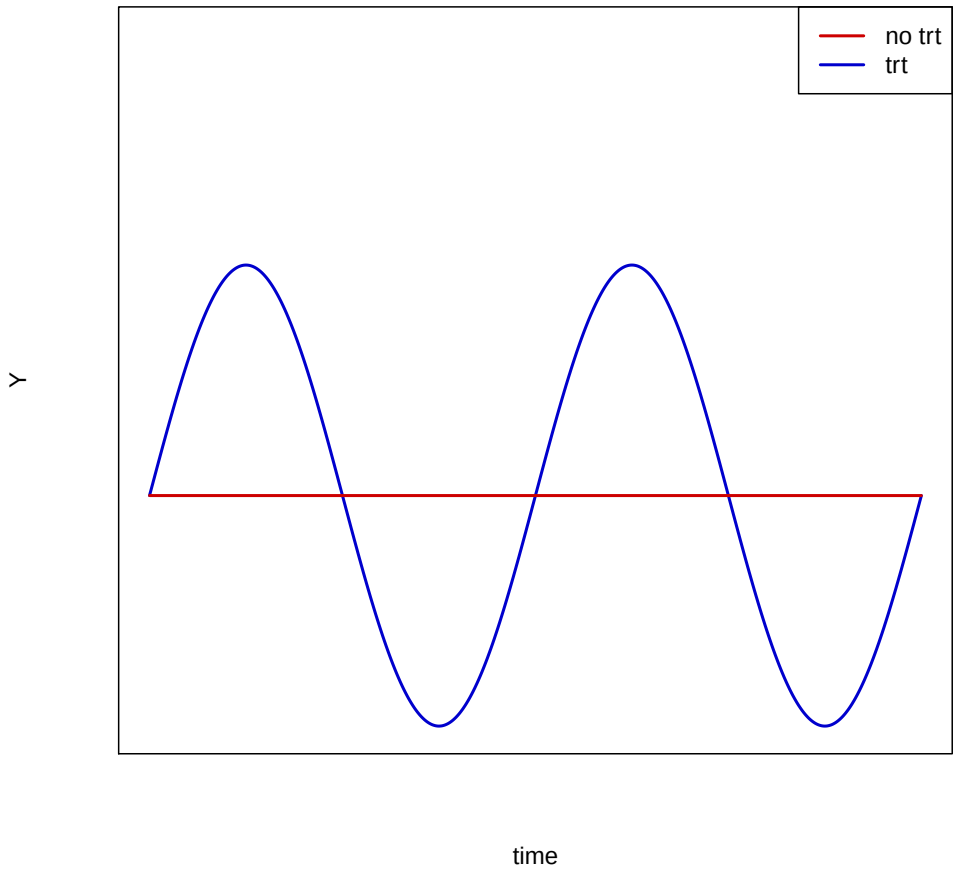


Example 5: Time Course Effect

There is a time-course effect within each condition and a phase shift with respect to treatment. Is this interesting?



Example 6: Treatment Time Course Effect



Example 6: Treatment Time Course Effect