Non-linear, Multivariate & Bayesian Statistics

Lecture 9

LSM3257

AY22/23; Sem 2 | Ian Z.W. Chan

Summary (Learning Objectives)

Non-linear Modelling

- GAM

Multivariate Statistics

- Theory: Response variables, purposes, dissimilarity matrices
- Understanding structure
    - Clustering: AHC, PAM, K-means
    - Unconstrained ordination: PCA, PCoA, NMDS, CA
- Interpreting/Making predictions
    - Constrained ordination: RDA, CAP, CCA
    - “Modelling”: MANOVA, PERMANOVA, MANCOVA & Multivariate GLM

Bayesian Statistics

- Bayes’ Rule and a stan_glm() example

Non-linear Modelling

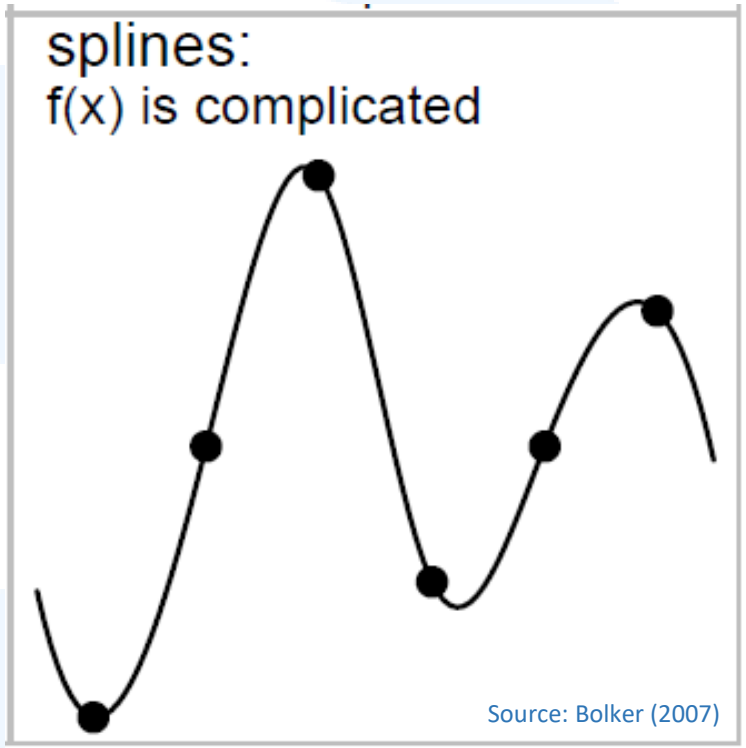
GAM

Generalised Additive Model

What is a GAM?

Used to create (very) complex non-linear models

- Uses splines to fit a curve to the data: splines are curves with constantly changing radius that are made to pass through a series of fixed points.



The curvature that is introduced should improve the predictive performance (Maximum Likelihood) of the model

But the model will be penalised for the curvature (because this increases its complexity).

Fitting a GAM

Load package and dataset:

```
require(mgcv)
data(columb)
str(columb)
```

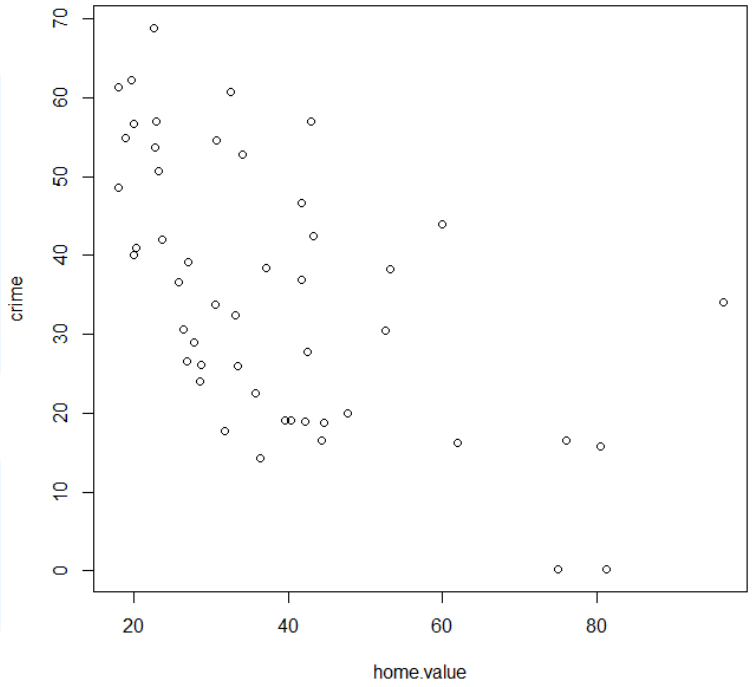
Dataset on <crime> rates as a function of <area> of the district, average <home.value>, average <income> and <open.space> in the area.

```
> str(columb)
'data.frame':   49 obs. of  8 variables:
 $ area      : num  0.3094 0.2593 0.1925 0.0838 0.4889 ...
 $ home.value: num  80.5 44.6 26.4 33.2 23.2 ...
 $ income    : num  19.53 21.23 15.96 4.48 11.25 ...
 $ crime     : num  15.7 18.8 30.6 32.4 50.7 ...
 $ open.space: num  2.851 5.297 4.535 0.394 0.406 ...
 $ district  : Factor w/ 49 levels "0","1","2","3",..: 1 2
 $ x         : num  8.83 8.33 9.01 8.46 9.01 ...
 $ y         : num  14.4 14 13.8 13.7 13.3 ...
```

Let's use <home.value> to explain <crime>

Visualise data:

```
plot(crime~home.value,data=columb)
```



Is this linear or non-linear?

Fit a linear model:

This is equivalent to a linear model

```
mod3.lm=gam(crime~home.value,data=columb)
```

Function to fit a GAM

Fitting a GAM

Fit the GAM non-linear model:

```
mod3.g=gam(crime~s(home.value),data=columb)
```

This s(…) tells R to use splines on the variable. R decides how many splines to use on its own (you can specify this number "s(home.value, k=20)" and some other things, check ?gam). This can only take continuous variables. You can add categorical variables as "linear variables", i.e. without the s(…).

Note: you would still have to specify the correct error distribution using "family=": poisson, binomial, quasipoisson, quasibinomial, or nb (negative binomial).

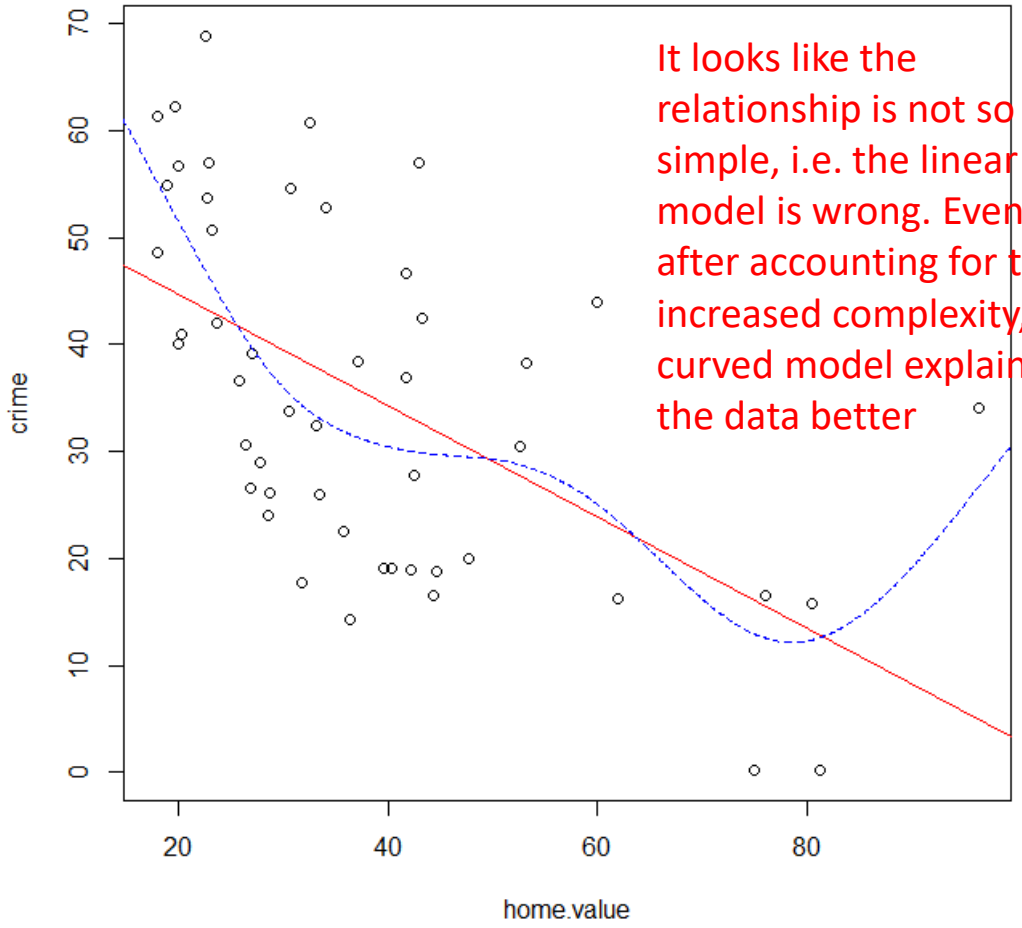
Compare the two:

```
AIC(mod3.lm,mod3.g)
#The GAM is better
```

```
> AIC(mod3.lm,mod3.g)
             df      AIC
mod3.lm 3.000000 400.5179
mod3.g  6.137319 394.3689
```

Visualise the 2 models:

```
plot(crime~home.value,data=columb) #points
abline(mod3.lm,col="red") #linear model
xv=seq(0,100,0.1)
yv_3.g=predict(mod3.g,list(home.value=xv))
lines(xv,yv_3.g,col="blue",lty=2) #GAM
```



It looks like the relationship is not so simple, i.e. the linear model is wrong. Even after accounting for the increased complexity, the curved model explains the data better

Interpreting results

Check the model:

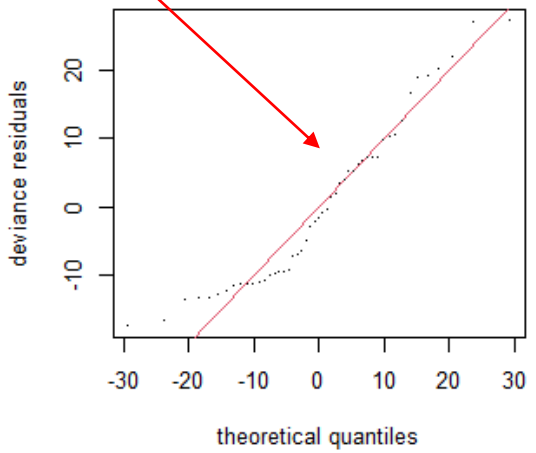
```
par(mfrow=c(2,2))
gam.check(mod3.g)
```

Tells you whether the model converged. If not, try to simplify your model.

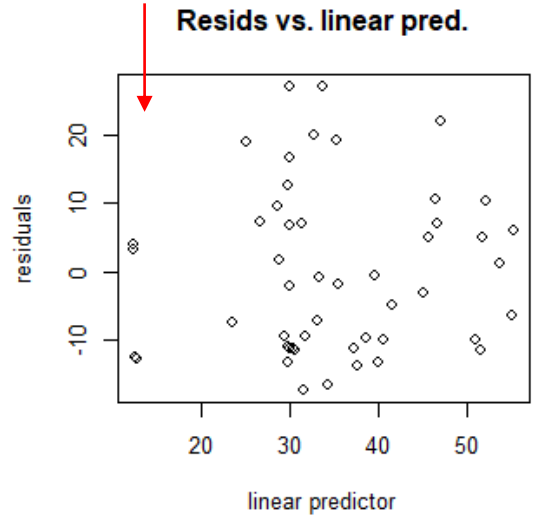
```
> gam.check(mod3.g)

Method: GCV   Optimizer: magic
Smoothing parameter selection converged after 4 iterations.
The RMS GCV score gradient at convergence was 0.001427689 .
The Hessian was positive definite.
Model rank =  10 / 10

Basis dimension (k) checking results. Low p-value (k-index<1) may
indicate that k is too low, especially if edf is close to k'.

                 k'  edf k-index p-value
s(home.value) 9.00 4.14    1.05    0.52
```

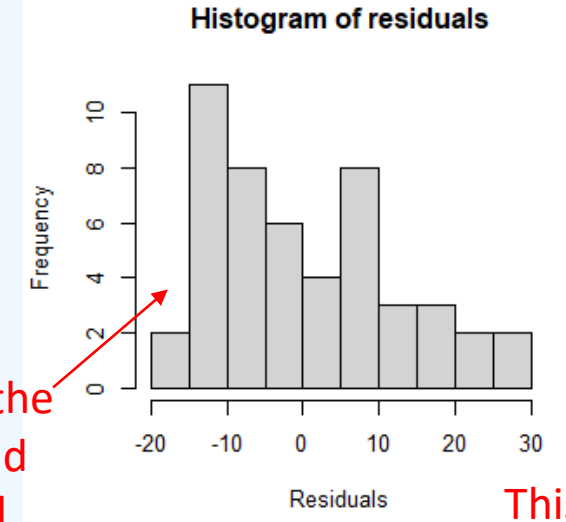
p-value < 0.05 would be bad (indicates that residuals are not randomly distributed) – try increasing the "k" specified for the variable to more than the value here. Here it is OK.

This looks at whether the residuals are normally distributed. They should follow the red line. A little marginal here.

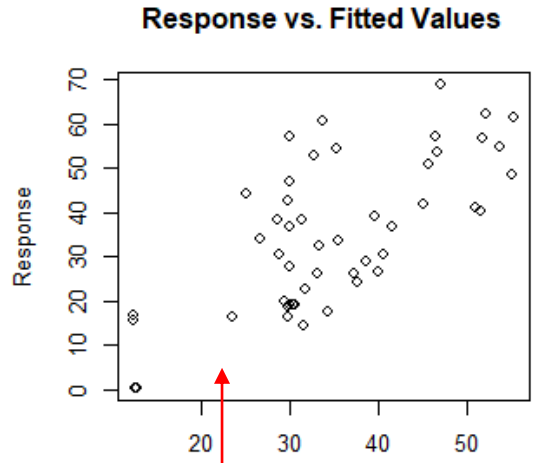


This should be randomly distributed. Looks OK in general (cannot expect too much from a GAM)





This also looks at the normality of the residuals. It should look like a normal distribution.



This looks at how well your model fits the data. It should be as close as possible to the y = x diagonal line.

Interpreting results

View results:

`summary(mod3.g)`

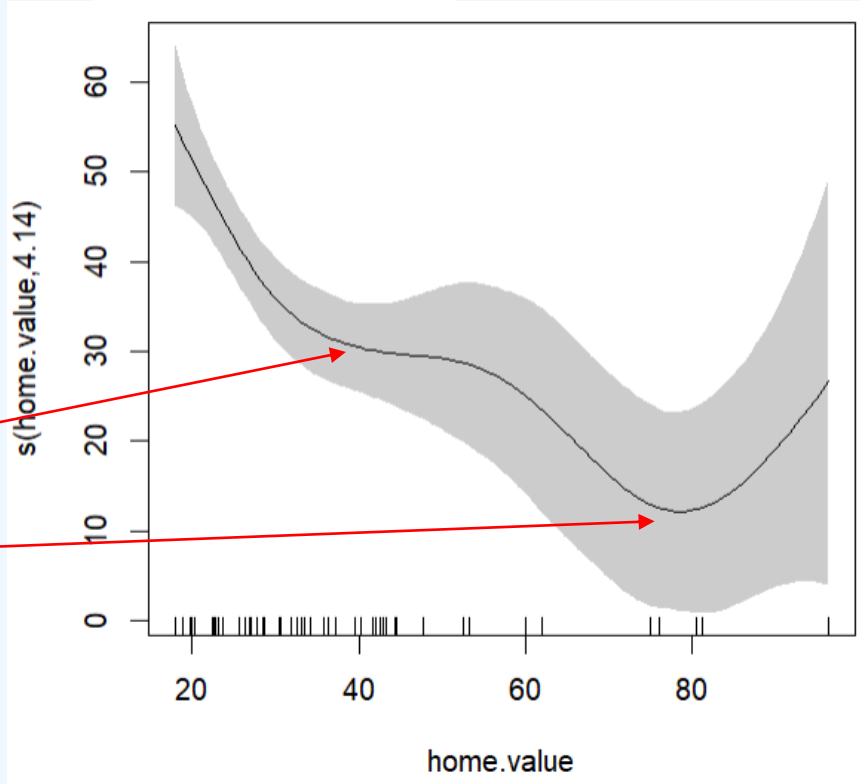
```
> summary(mod3.g)

Family: gaussian
Link function: identity

Formula:
crime ~ s(home.value)

Parametric coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   35.129      1.803   19.48   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Approximate significance of smooth terms:
                edf Ref.df     F  p-value
s(home.value) 4.137  5.068 7.507 3.44e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Linear terms are reported here (there are none in this model)

The non-linear term for <home.value> is significant and uses 4.137 effective degrees of freedom

Graphically:

`plot(mod3.g,shift=coef(mod3.g)[1],shade=T)`



All else being equal, there are about 30 crimes for properties worth about $40k, and about 12 crimes for those worth about $80k

Fitting more complicated GAMs

Note: You can also fit random effects using gamm(). There are 2 ways to do it. See: ?gamm and http://r.qcbs.ca/workshop08/book-en/quick-intro-to-generalized-additive-mixed-models-gamms.html.

Fit a GAM with multiple terms and an interaction:

```
mod3.g2=gam(crime~s(home.value)+s(area)+te(income,open.space),data=columb)
```

This te() tells R to allow <income> and <open.space> to interact.

Simplify:

```
summary(mod3.g2)
```

```
Approximate significance of smooth terms:
                        edf Ref.df     F p-value
s(home.value)         4.508  5.435 3.263  0.0131 *
s(area)               1.594  1.973 1.486  0.2229
te(income,open.space) 4.586  5.219 2.952  0.0233 *
```

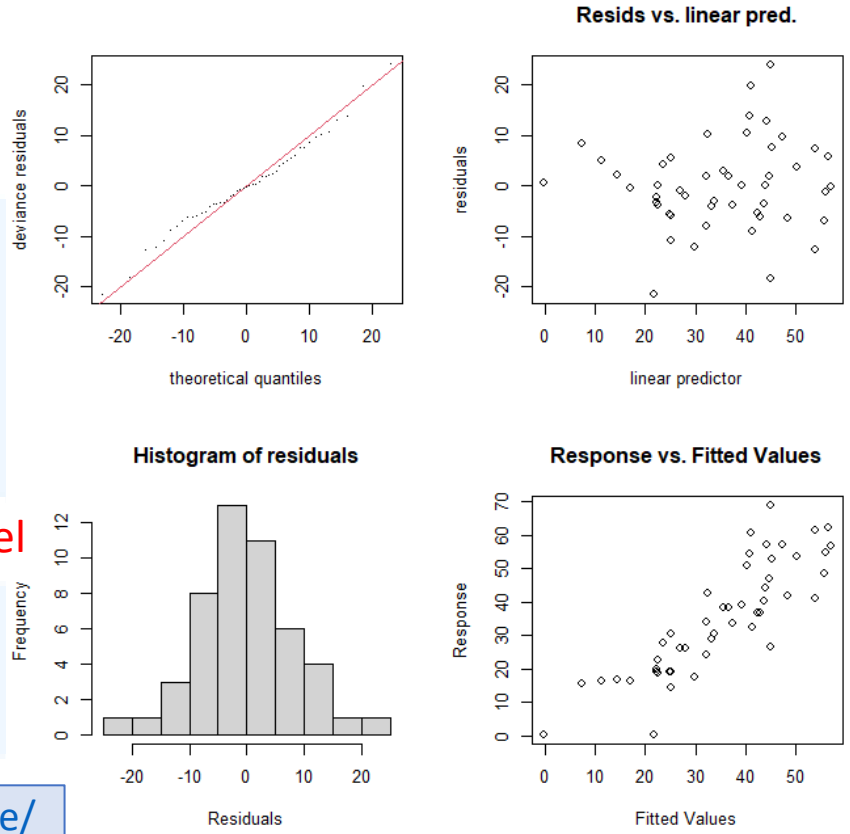
```
mod3.g3=update(mod3.g2,~.-s(area))
summary(mod3.g3)
AIC(mod3.g,mod3.g2,mod3.g3)
```

```
> AIC(mod3.g,mod3.g2,mod3.g3)
                df      AIC
mod3.g   6.137319 394.3689
mod3.g2 12.687751 375.4379
mod3.g3 10.236688 377.6154
```

```
#I decide to follow the AIC: mod3.g2 is the best(my personal interpretation)
```

Check:

```
gam.check(mod3.g2)
```



The residuals of this new model look much more normally distributed than in mod3.g

There is A LOT more… a great starter resource: https://noamross.github.io/gams-in-r-course/

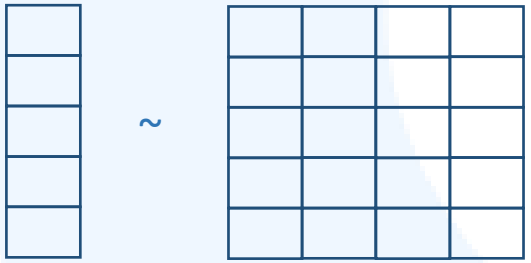
Multivariate Statistics

What are Multivariate Analyses

Used to analyse 2 or more response variables at the same time.

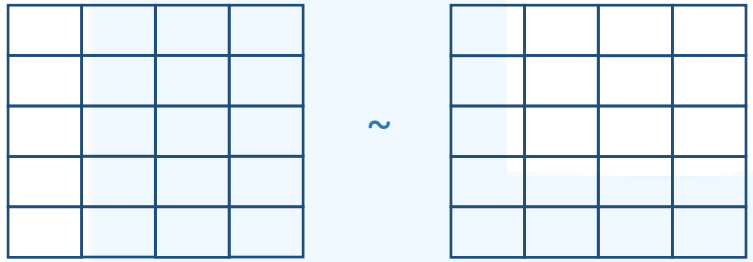
Univariate analysis
(everything so far)

1 Response variable ~ ≥1 Explanatory variable(s)



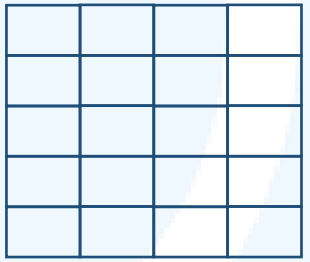
Multivariate analysis

≥2 Response variables ~ ≥1 Explanatory variable(s)



or

≥2 Response variables



Many (many many) different analyses: I will only **introduce the most common**

- Read up details on how they work, their strengths and weaknesses, etc.
- Google for alternative analyses

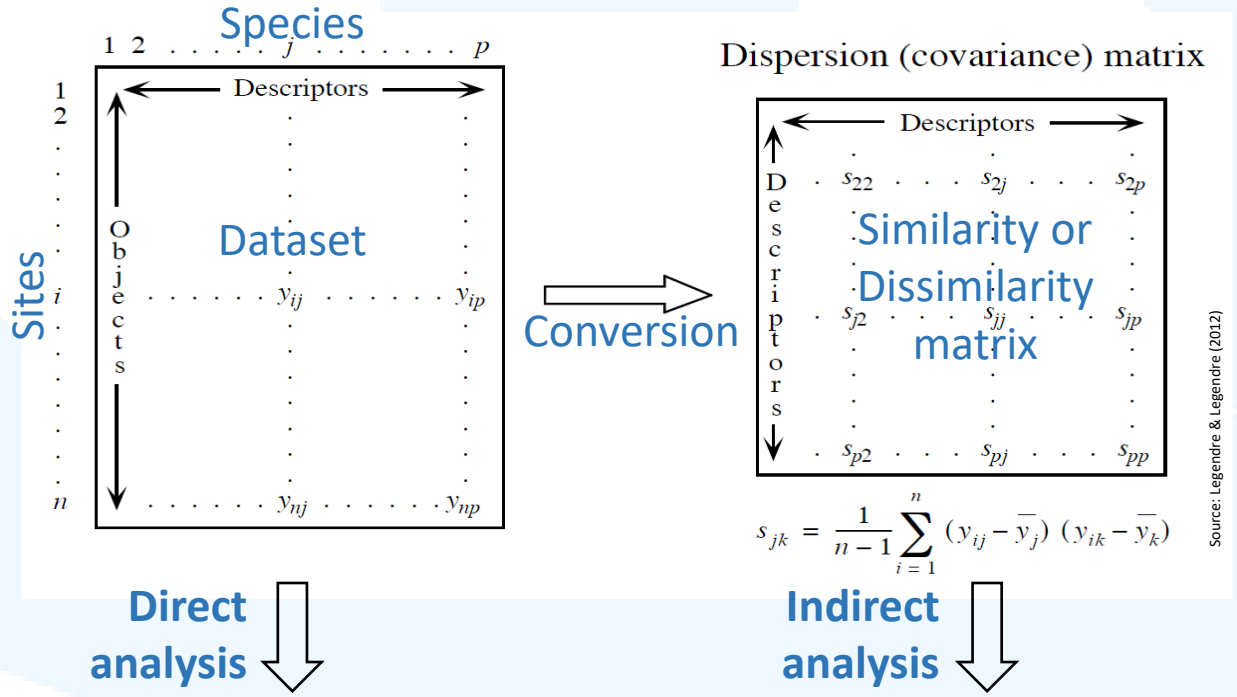
Purposes, Datasets and (Dis)similarity matrices

In general, multivariate analyses have 2 purposes:

1) Understanding structure: visually looking for groupings within your matrix of response variables (i.e. no explanatory variables and no p-values).

2) Interpreting/Predicting: using a matrix of explanatory variables to explain a matrix of response variables.

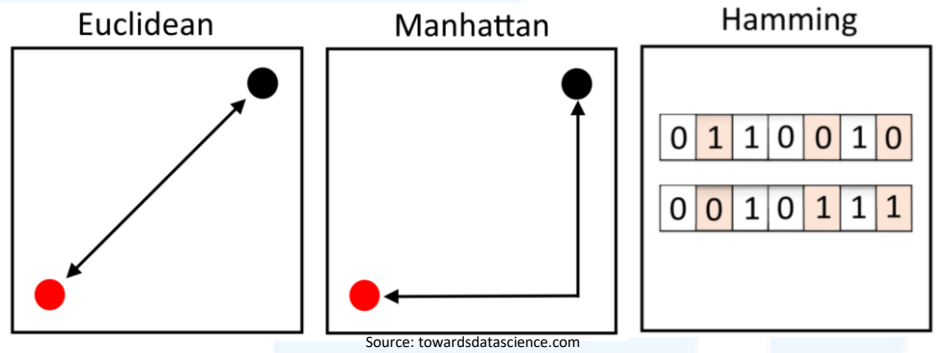
General concept:



Interpretation/Prediction (using a dataset of explanatory variables)
Structure (without using explanatory variables)

(Dis)similarity/Distance matrices

A matrix of dissimilarities/distances between the values in your original dataset.
Note: Distance ≠ Dissimilarity, read more here.



Many different distance measures in statistics.

- Commonly-used in ecology:

Euclidean: direct distance between two points for continuous data.

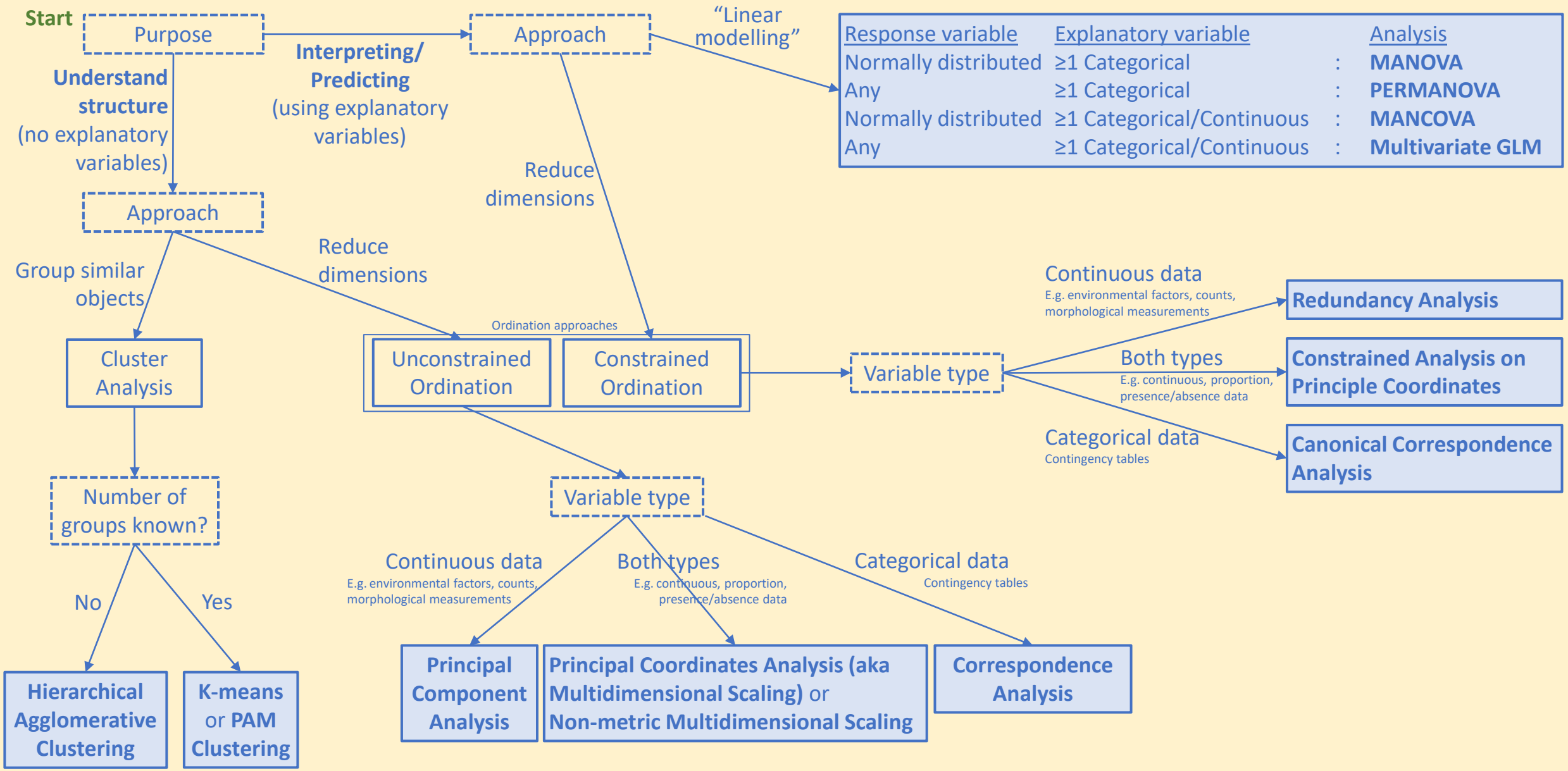
Manhattan: x-distance + y-distance between two points. Preferred in datasets with many variables (i.e. high dimensionality).

Hamming: counts the number of variables that are different. For categorical variables.

Bray-Curtis: Based on counts of similar species at different sites. Preferred for species matrices.

Fuller guide to choosing the right distance measure.

Multivariate analyses – Analysis decision tree



Cluster Analysis

HAC, PAM, K-means

What is clustering?

Clustering tries to split your data into groups based on how similar they are.

Two most common types…

1) **Connectivity-based** (Hierarchical): when number of groups is not known.

- Agglomerative: starts with individual datapoints (singletons) then groups the closest together.
- Divisive: starts with all datapoint in one cluster and then splits them into groups.

2) **Centroids-based** (Partitioning): Repeatedly reassigns points to a pre-specified number of groups and minimises the distances of the points to their centroids.

Before you cluster:

- Remove all NAs.
- Consider scaling your variables using scale() (changes all your variables to have a mean of 0 and s.d. of 1): places the same importance on all variables.

Hierarchical Agglomerative Clustering

If you don't know how many clusters (groups) you want.

Load dataset, remove NAs and scale data:

```
require(car)
d1=Freedman #dataset on characteristics of cities in the US
d1=na.omit(d1)
d1.1=scale(d1)
```

	population	nonwhite	density	crime
Akron	675	7.3	746	2602
Albany	713	2.6	322	1388
Allentown	534	0.8	491	1182
Anaheim	1261	1.4	1612	3341
Atlanta	1330	22.8	770	2805
Bakersfield	331	7.0	41	3306

Calculate dissimilarity matrix with Euclidean distances (AHC needs this as input):

```
d1_dist=dist(d1.1) #Note: can specify other distances with "method="
```

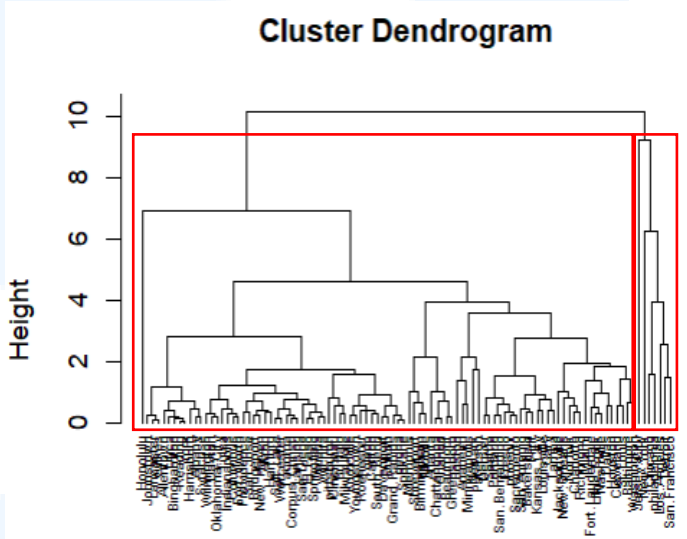
Perform clustering:

```
ahcmod1=hclust(d1_dist)
```

Hierarchical Agglomerative Clustering

View dendrogram to decide on number of clusters you want:

```
plot(ahcmod1,cex=0.5,hang=-1)
#I decide that I want 2 clusters
```



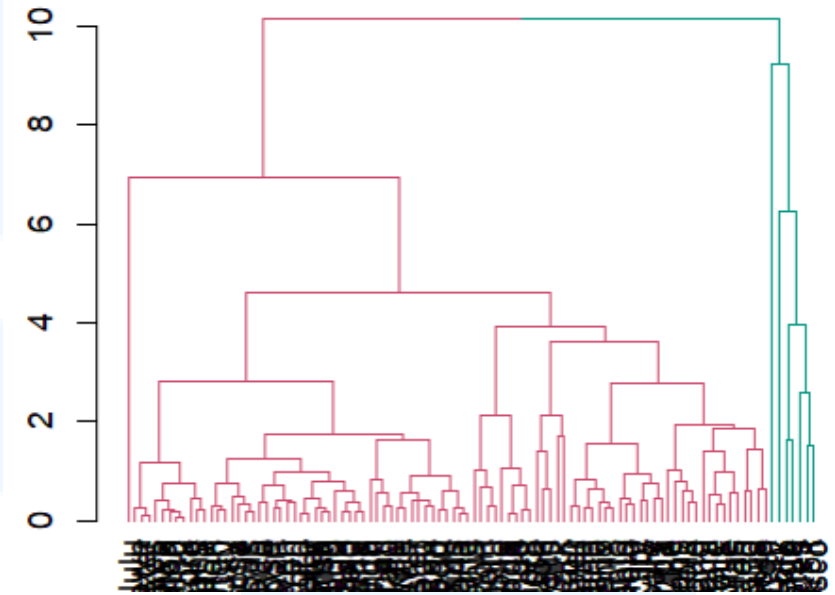
Cut into 2 clusters to save results:

```
ahcmod2=cutree(ahcmod1,k=2)
d1$cluster=as.integer(ahcmod2)
```

These are your results: which cluster each row is assigned to

Plot results:

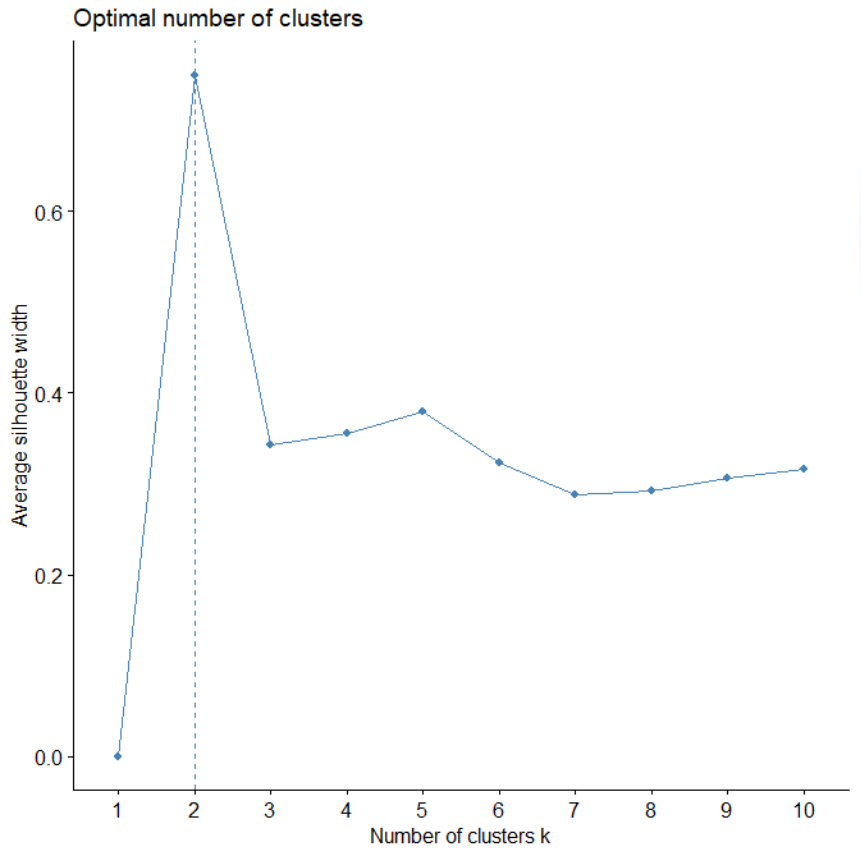
```
require(dendextend)
ahcdend1=as.dendrogram(ahcmod1)
ahcdend2=color_branches(ahcdend1,k=2)
plot(ahcdend2)
```



Centroid-based: K-means and PAM Clustering

If you know how many clusters (groups) you want:

- K-means uses centroids (imaginary points).
- PAM uses medoids (actual points, more robust to outliers).



Read in dataset:

```
d2=Freedman
d2=na.omit(d2)
```

Determining optimal number of clusters:

```
require(factoextra)
fviz_nbclust(d2,FUNcluster=kmeans,method="silhouette") #2
```

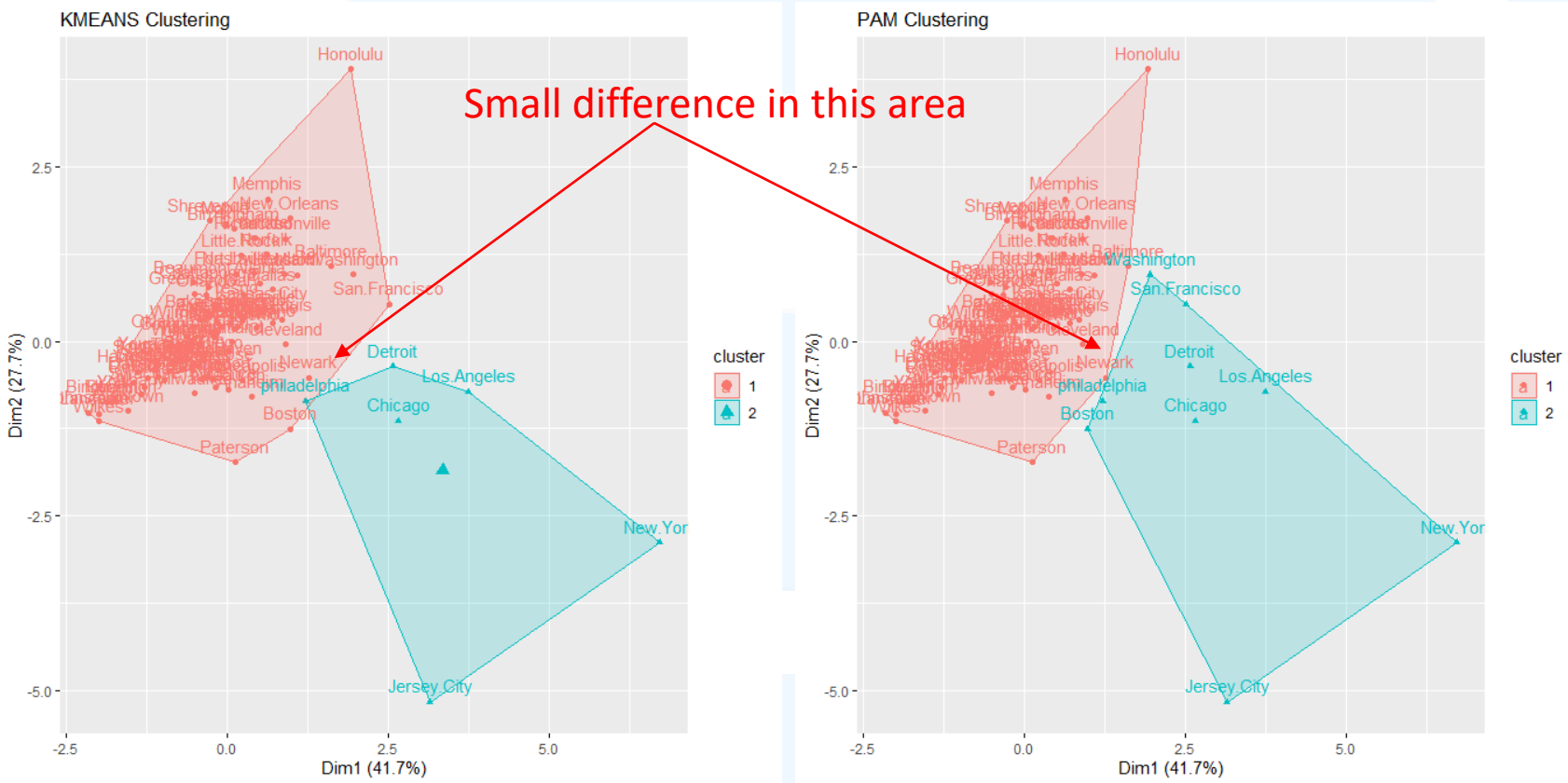
Change to “pam” for PAM clustering

Centroid-based: K-means and PAM Clustering

Do the clustering:

```
kmmod=eclust(d2,FUNcluster="kmeans",k=2)
pammod=eclust(d2,FUNcluster="pam",k=2)
```

Change to “pam” for PAM clustering



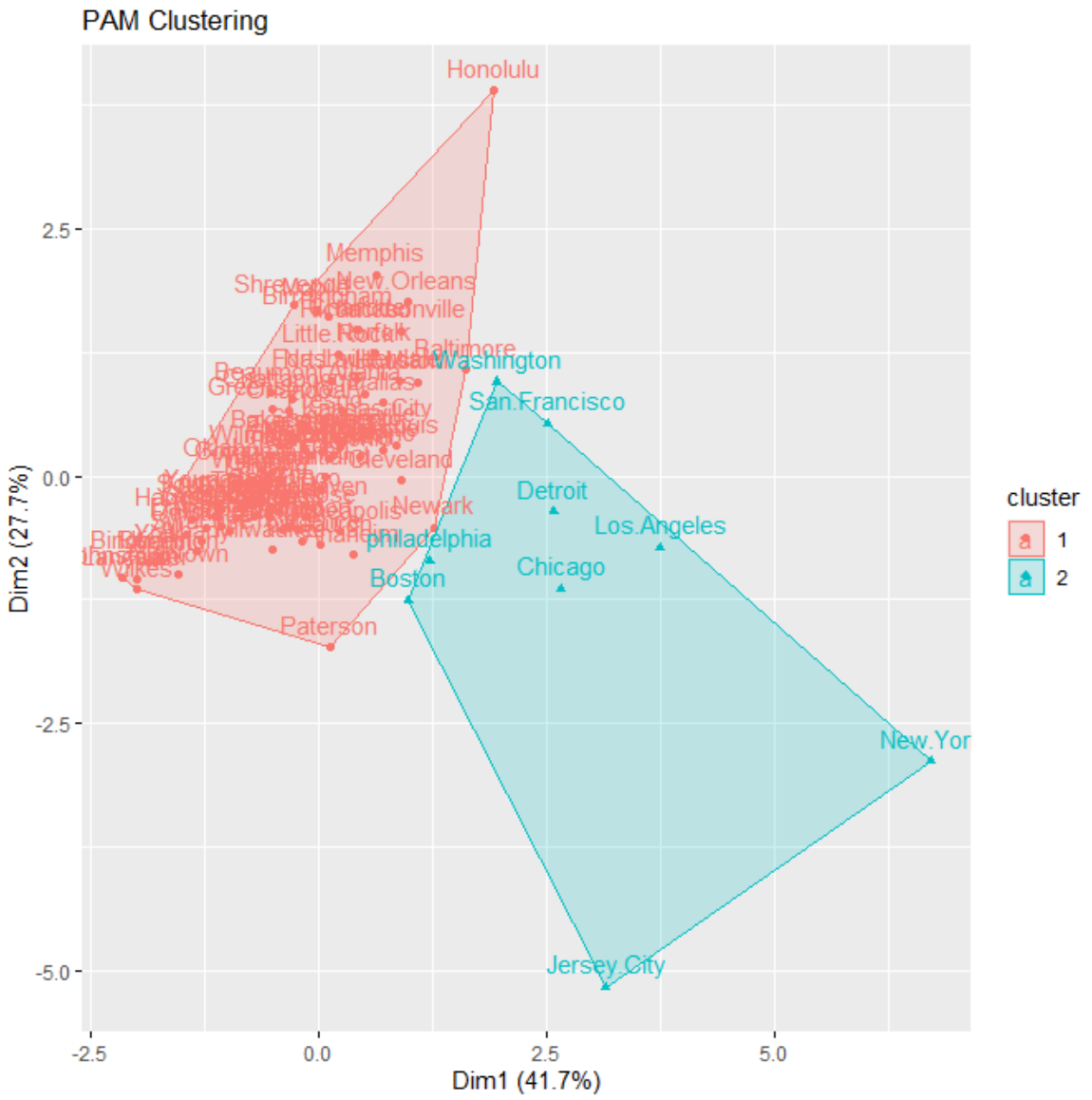
Saving results:

```
d2$cluster=as.integer(kmmod$cluster)
```

Interpreting results

Clustering just **tells you visually which datapoints are more similar to one another** (similar to Unconstrained Ordination). There are no p-values.

It is then up to you to **interpret these results based on biological intuition**.



Example (from PAM results):

The cities in Cluster 2 are all large, metropolitan cities, suggesting that this has an effect on the variables in the dataset.

Unconstrained Ordination

PCA, PCoA, NMDS, CA

What is Unconstrained Ordination?

When you have many variables, it's difficult to see relationships in the dataset:

- E.g. if you have 10 variables, to compare all of them, you would need to plot 55 graphs at least; even more to investigate interactions and combinations of variables.

Unconstrained Ordination helps to make it easier to visualise what variables are important:

1) We first plot all the datapoints in multivariate space (e.g. 10-D space for 10 variables).

2) We then rotate this scatterplot so that we are looking at it from the direction where the points are most spread out, i.e. so that the x- and y- axes are the axes with the most variance. **The axes we see are hence combinations of the original variables.**

3) We then flatten (aka project) this scatterplot onto 2 dimensions for easy viewing and interpreting.

There are no p-values.

What is Unconstrained Ordination?

Different types of ordination for different types of data.

Method	Distance preserved	Variables
Principal component analysis (PCA)	Euclidean distance	Quantitative data, linear relationships (beware of double-zeros)
Correspondence analysis (CA)	χ^2 distance	Non-negative, dimensionally homogeneous quantitative or binary data; species frequencies or presence/absence data
Principal coordinate analysis (PCoA), metric (multidimensional) scaling, classical scaling	Any distance measure	Quantitative, semiquantitative, qualitative, or mixed
Nonmetric multidimensional scaling (nMDS)	Any distance measure	Quantitative, semiquantitative, qualitative, or mixed

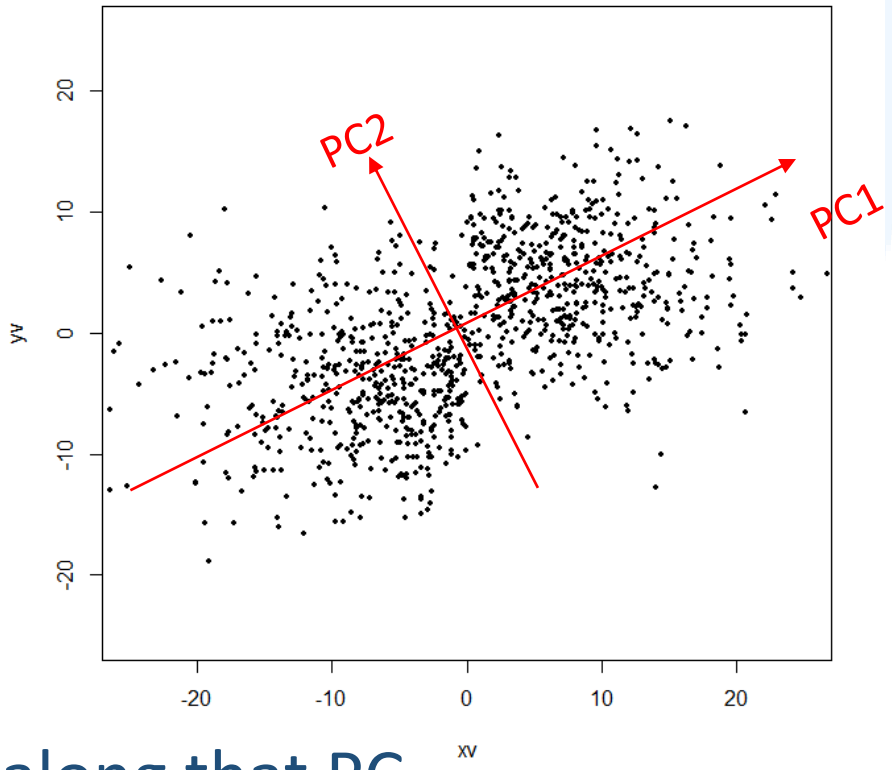
Source: Legendre & Legendre (2012)

Principal Component Analysis (PCA)

Note: It is NOT compulsory that the data are normally distributed, but PCA tends to work better when they are (read this).

Direct, unconstrained ordination for continuous data. Assumes a linear relationship. Uses Euclidean distance.

Visualises the data so that the x-axis of the PCA plot (aka PC1) is the axis with the most variation, and the y-axis (aka PC2) is the axis with the most variation while being orthogonal (i.e. at right angles) to PC1.



Each PC is a linear combination of the original variables.

- The first few PCs explain most of the variation.
- Looking at what variables are in PC1 and PC2 allows us to identify important variables.

Note: eigenvector = PC; eigenvalue = the variance along that PC.

Principal Component Analysis (PCA)

Loading a dataset of 9 variables:

```
d3=mtcars[,c(1:8,11)]
```

Running the PCA and viewing the results:

```
pca1=prcomp(d3,center=T,scale.=T)
summary(pca1)
biplot(pca1) #easy to plot but not very pretty
```

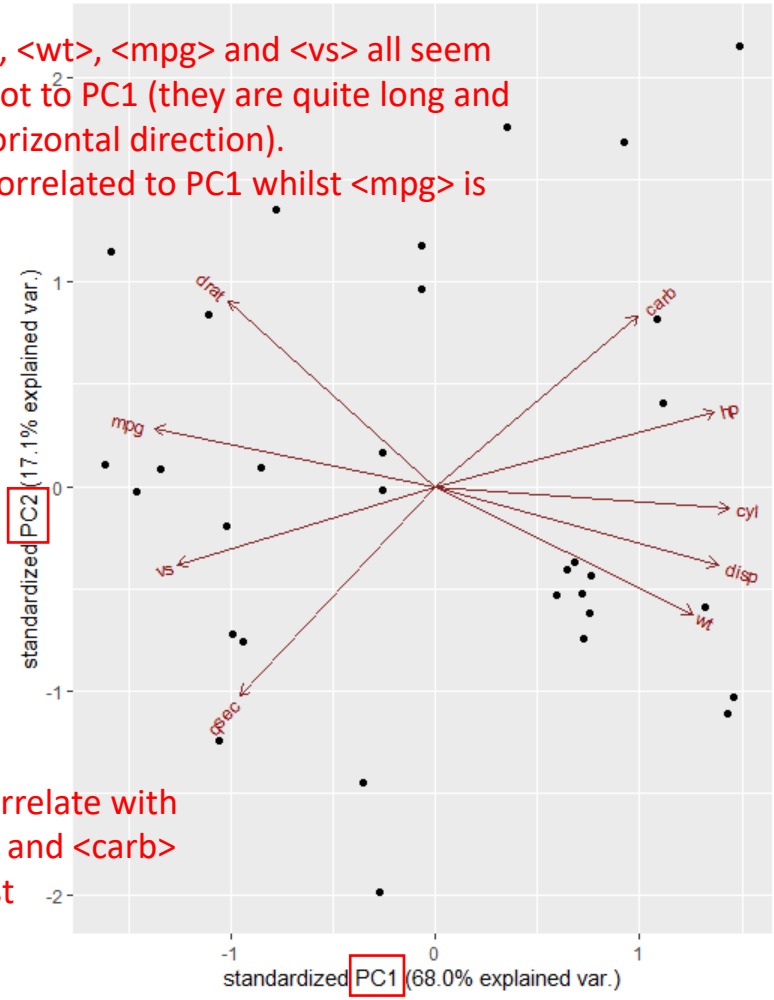
Visualising (a little annoying to install the packages):

```
install.packages("devtools")
library(devtools)
install_github("vqv/ggbiplot")
require(ggbiplot)
ggbiplot(pca1) #nicer
```

```
> summary(pca1)
Importance of components:
                          PC1    PC2     PC3     PC4     PC5
Standard deviation     2.4732 1.2390 0.75052 0.50607 0.47234
Proportion of Variance 0.6796 0.1706 0.06259 0.02846 0.02479
Cumulative Proportion  0.6796 0.8502 0.91277 0.94123 0.96601
```

Notice how the first 2 PCs already explain >85% of the variation in the dataset

1) <hp>, <cyl>, <disp>, <wt>, <mpg> and <vs> all seem to contribute quite a lot to PC1 (they are quite long and generally point in a horizontal direction).
2) <cyl> is positively correlated to PC1 whilst <mpg> is negatively correlated



No variable seems to correlate with PC2, but <drat>, <qsec> and <carb> may contribute the most

For more plotting options, see: https://www.datacamp.com/community/tutorials/pca-analysis-r

Principal Component Analysis (PCA)

Calculating loadings (how much each variable contributes to the PCs) to see which variables are important:

```
func1=function(rotation,sdev){rotation*sdev}
func2=function(varcos,compcos){varcos*100/compcos}
varcos=(t(apply(pca1$rotation,1,func1,pca1$sdev)))^2
compcos=apply(varcos,2,sum)
loadings=t(apply(varcos,1,func2,compcos))
loadings
```

<mpg> contributes to 13.6% of PC1

```
> loadings
             PC1        PC2
mpg   13.627724  2.3488690
cyl   14.957621  0.3283814
disp  13.846794  4.2457884
hp    13.437417  3.7812641
drat   7.379182 23.7509079
wt    11.507712 11.1725419
qsec   6.614576 30.1349935
vs    11.551932  4.1977220
carb   7.077041 20.0395319
```

Just change these to your PCA model object and run the whole chunk of code

Summary:
1) a lot of variables are important in PC1
2) <drat>, <qsec> and <carb> are important in PC2.

Interpretation: the various cars differ mostly due to a combination of fuel efficiency <mpg>, number of cylinders <cyl>, size <disp> and power <hp>.

Principal Component Analysis (PCA)

If you have pre-defined groups based on some variable, you can see how well these groups spread out from one another.

Assigning grouping by country manually:

```
d3groups=c(rep("Japan", 3),
rep("US",4), rep("Europe",
7),rep("US",3), "Europe", rep("Japan",
3), rep("US",4), rep("Europe", 3),
"US", rep("Europe", 3))
```

Plotting different countries by colour:

```
ggbiplot(pca1,labels=rownames(d3),
groups=d3groups,ellipse=T,var.axes=F)
```

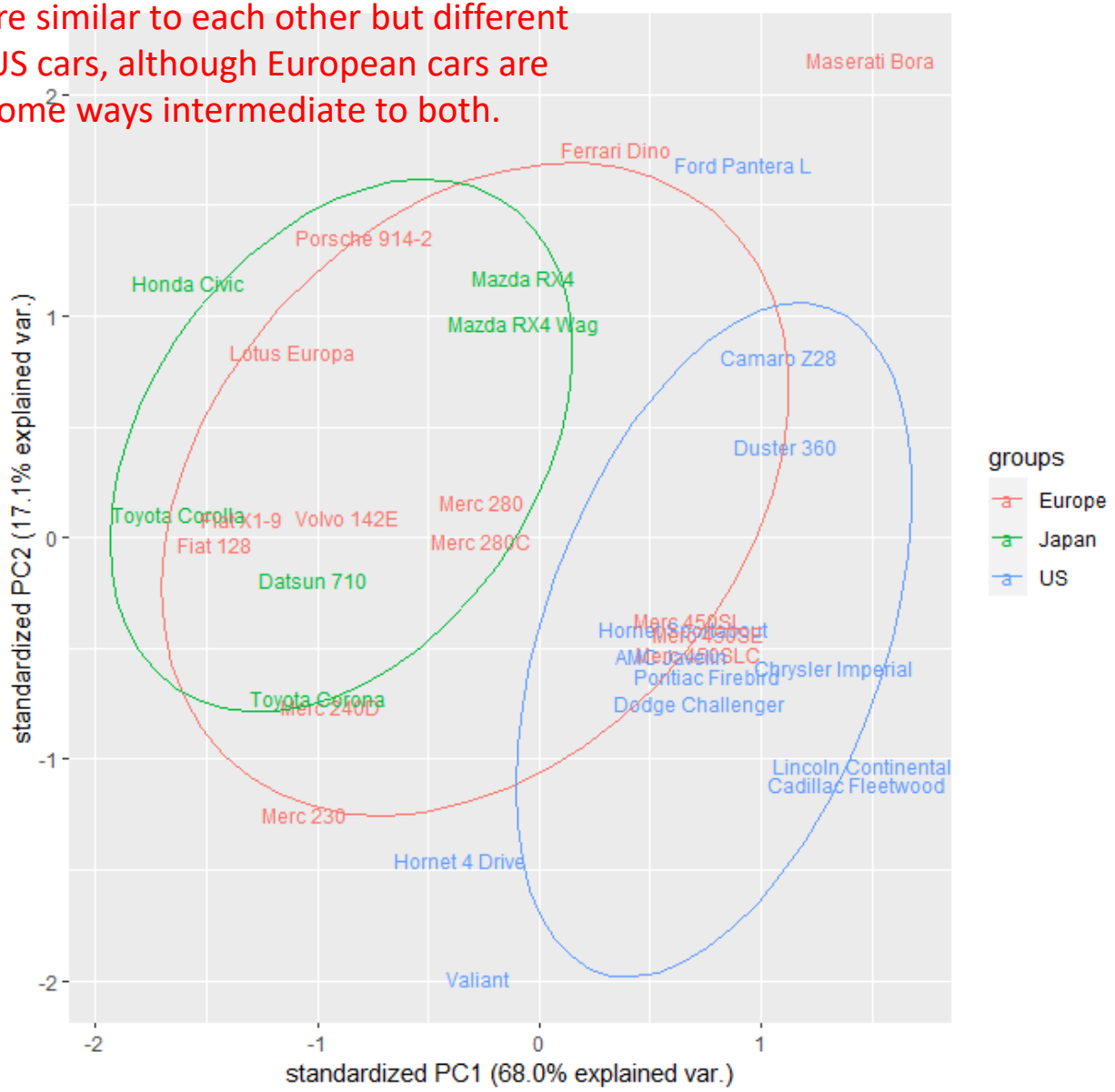
Replaces the points with the names of the cars

Assign coloured groups

Draws ellipses based on 68% confidence, change to 95% by adding “ellipse.prob=0.95”

Gets rid of the arrows (we are now interested in groupings)

European and Japanese cars tend to be more similar to each other but different to US cars, although European cars are in some ways intermediate to both.



Principle Coordinates Analysis (PCoA)

Also known as Multidimensional Scaling (MDS). Indirect, unconstrained ordination for many types of data. Assumes a linear relationship.

Note: distances available are "bray" (the default), "manhattan", "euclidean", "canberra", "kulczynski", "jaccard", "gower", "altGower", "morisita", "horn", "mountford", "raup" , "binomial" or "chao". If you choose “Euclidean”, it’s the same as a PCA. Use “bray” for abundance data. Use “jaccard” for presence/absence data.

Dataset:

```
d3=mtcars[,c(1:8,11)]
```

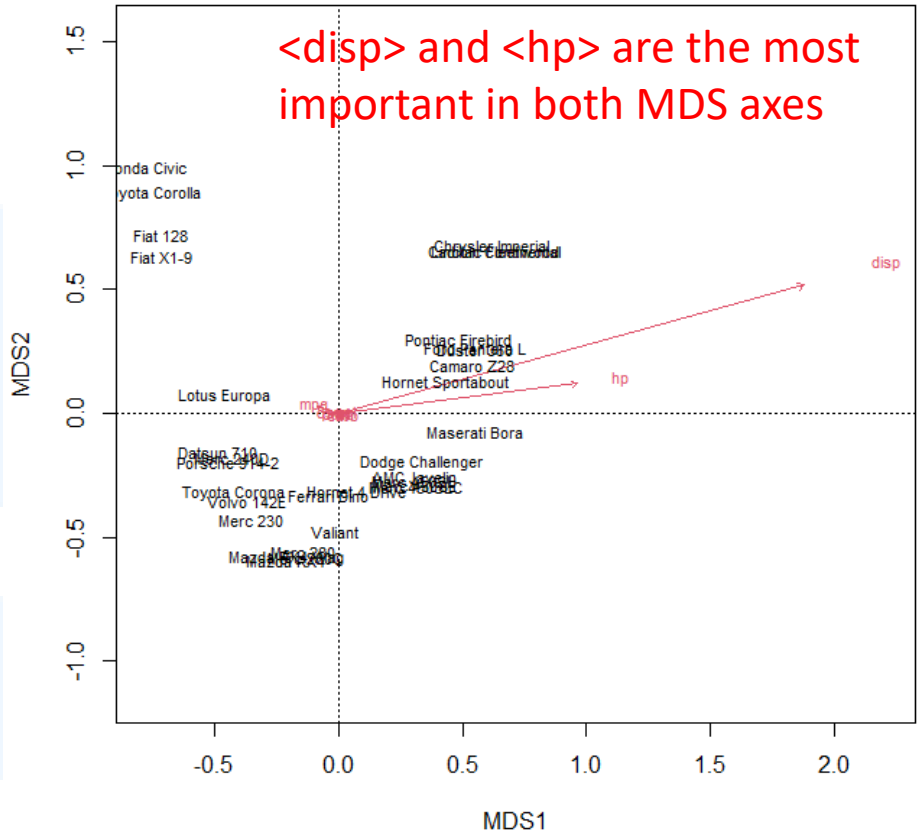
Fit the PCoA:

```
require(vegan)
pcoa1=capscale(d3~1,distance="bray")
```

View results:

```
summary(pcoa1)
biplot(pcoa1)
```

```
> summary(pcoa1)

Importance of components:
                          MDS1    MDS2
Eigenvalue              1.3792 0.08743
Proportion Explained    0.8622 0.05466
Cumulative Proportion   0.8622 0.91687
```



Non-metric Multidimensional Scaling (NMDS)

Indirect, unconstrained ordination for many types of data. **Better for ordinal data** (e.g. Likert scales). Uses ranks rather than actual data. Fits by trial and error (so you may get different results with successive runs).

Load dataset:

```
require(vegan)
data(varespec) #from vegan
```

Fit NMDS:

```
nmds1=metaMDS(comm=varespec,k=2,
distance="bray",trymax=100)
```

Chooses number of dimensions

Also many different distance measures available

View results:

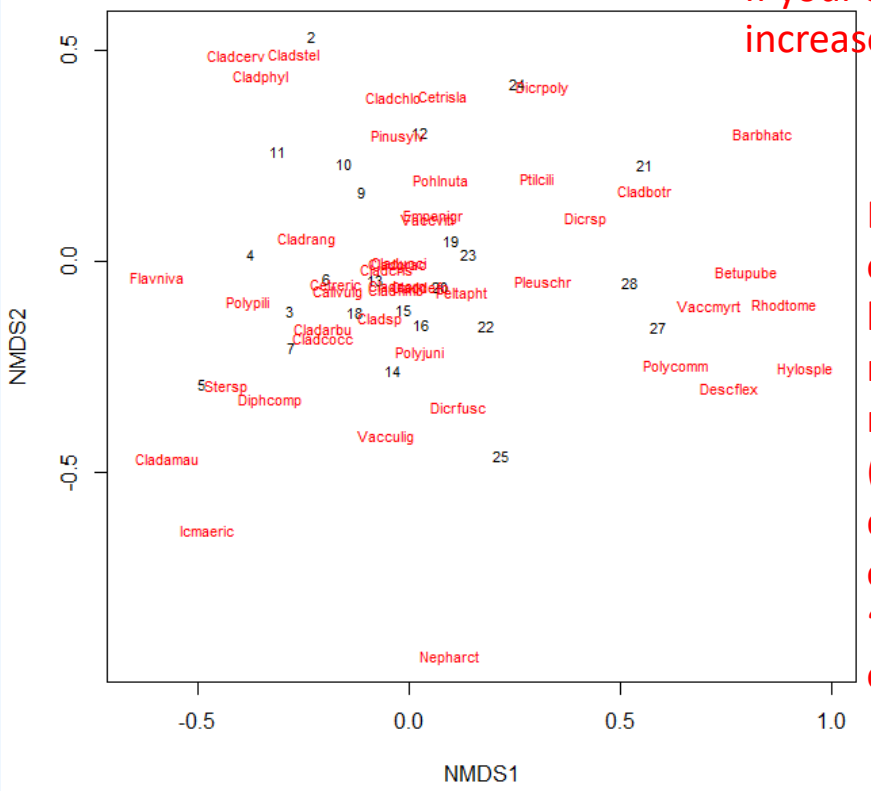
```
nmds1
ordiplot(nmds1,type="t")
```

For more plotting ideas, see: help(ordiplot), https://jonlefcheck.net/2012/10/24/nmds-tutorial-in-r/ and https://jkzorz.github.io/2019/06/06/NMDS.html.

Iteration produced a solution. If you see: “No convergent solutions”, try increasing the “trymax” argument.

```
> nmds1
Dimensions: 2
Stress:     0.1843196
```

Stress is an indication of how good the results are. Excellent: <0.05. Good: <0.1. OK: <0.2. Poor: >0.2. If your stress is too high, increase k in your NMDS.



By default, ordiplot() plots both columns in red (“species”) and rows in black (“sites”) from the dataset. To choose one, specify “display="species"” or “display="site"”.

Correspondence Analysis (CA)

Unconstrained ordination for categorical data (chi-square distance). Assumes a unimodal relationship. Data should be in the form of a contingency table.

Load dataset:

```
require(FactoMineR) #for performing the analysis
require(factoextra)
data(housetasks) #dataset from factoextra
```

Perform analysis:

```
ca1=CA(housetasks,graph=F)
```

Has to be a contingency table

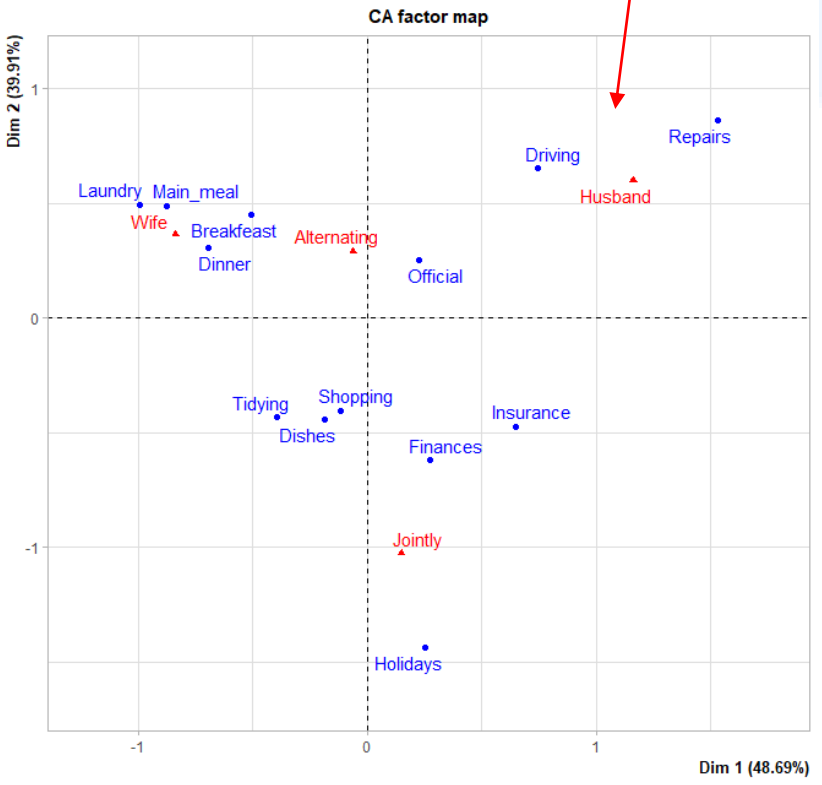
View results:

```
summary(ca1)
plot(ca1)
```

For more: http://www.sthda.com/english/articles/31-principal-component-methods-in-r-practical-guide/113-ca-correspondence-analysis-in-r-essentials/.

```
> housetasks
          Wife Alternating Husband Jointly
Laundry    156          14       2       4
Main_meal  124          20       5       4
Dinner      77          11       7      13
Breakfeast  82          36      15       7
Tidying     53          11       1      57
Dishes      32          24       4      53
Shopping    33          23       9      55
Official    12          46      23      15
Driving     10          51      75       3
Finances    13          13      21      66
Insurance    8           1      53      77
Repairs      0           3     160       2
Holidays     0           1       6     153
> summary(ca1)
Eigenvalues
                       Dim.1   Dim.2   Dim.3
Variance               0.543   0.445   0.127
% of var.             48.692  39.913  11.395
Cumulative % of var.  48.692  88.605 100.000
```

Driving and repairs tend to be more similar to each other, and they tend to be done by the husband alone.



Constrained Ordination

RDA, CAP, CCA

Redundancy Analysis (RDA)

The **constrained equivalent of PCA**. The ordination is constrained by a dataset of explanatory variables (they determine the axes). Data interpretation is visual.

Load dataset:

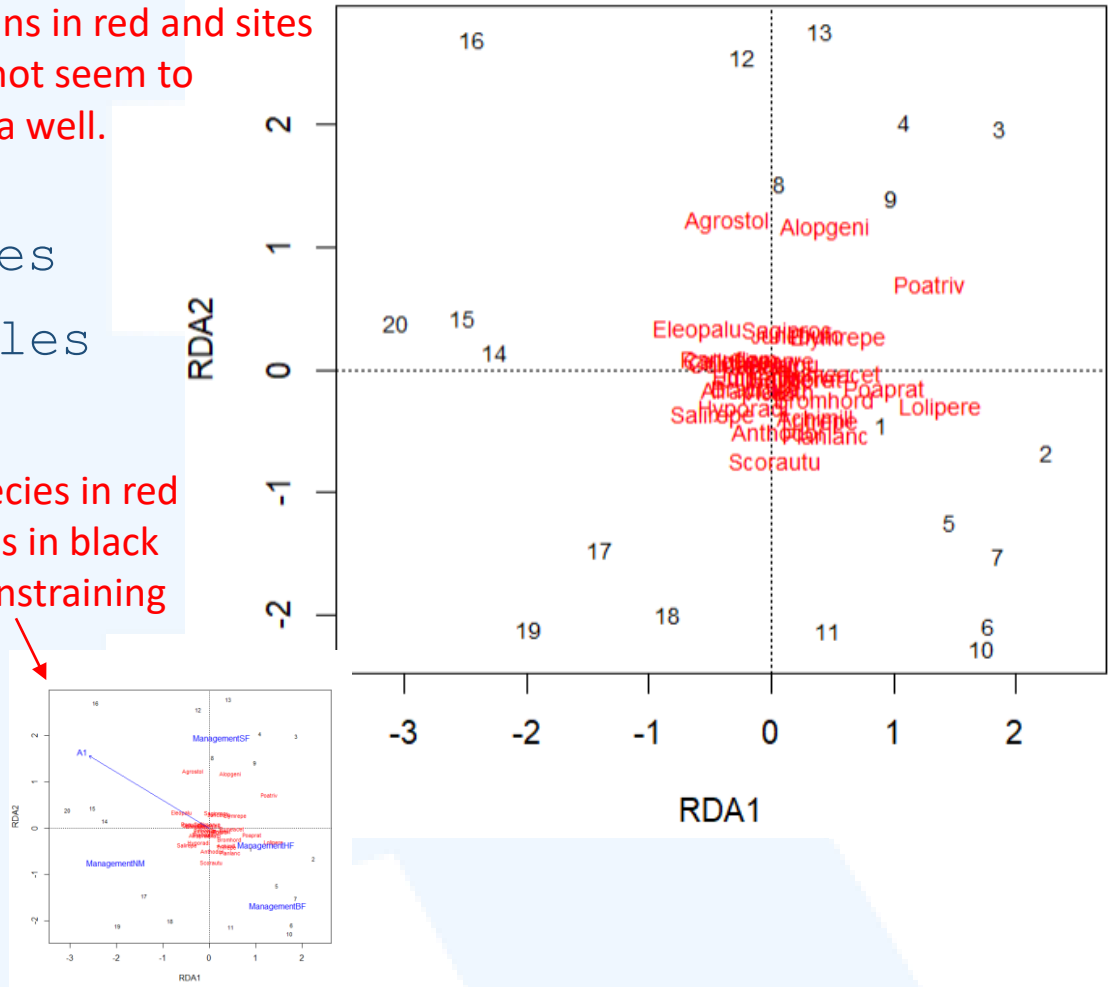
```
require(vegan)
data(dune) #the matrix of response variables
data(dune.env) #for the explanatory variables
```

Run RDA:

```
rda1=rda(dune~Management+A1,data=dune.env)
plot(rda1)
plot(rda1,display=c("wa","cn"))
```

The unconstrained part of the analysis (columns in red and sites in black) does not seem to explain the data well.

Default plot displays species in red (aka columns, “sp”), sites in black (aka rows, “wa”) and constraining variables in blue (“cn”)



Redundancy Analysis (RDA)

Display results:

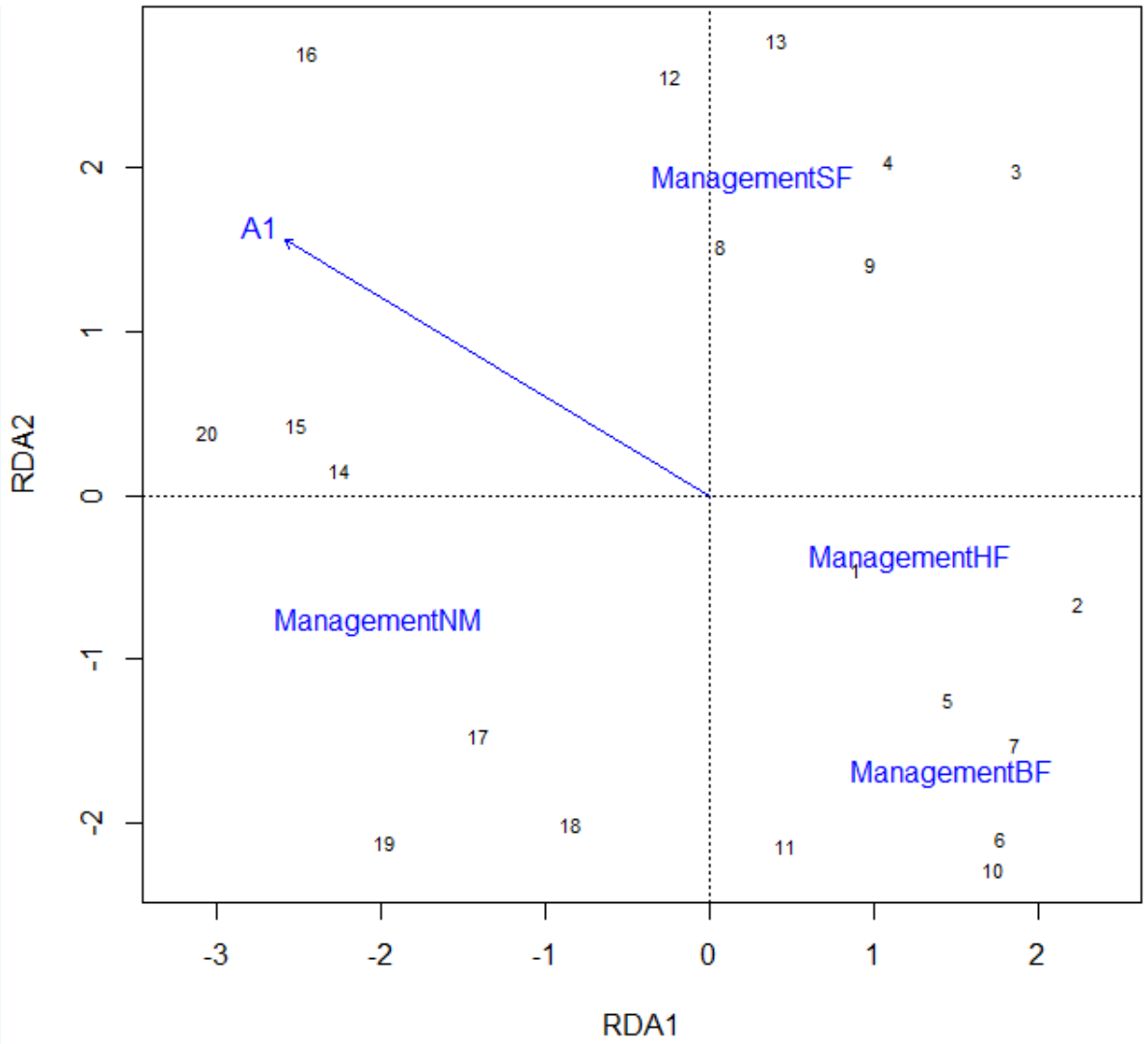
```
plot(rda1,display=c("wa","cn"))
summary(rda1)
```

```
Accumulated constrained eigenvalues
Importance of components:
                         RDA1    RDA2   RDA3    RDA4
Eigenvalue            15.1445 11.8619 4.0532 2.53821
Proportion Explained   0.4508  0.3531 0.1206 0.07555
Cumulative Proportion  0.4508  0.8038 0.9245 1.00000

Site scores (weighte

       RDA1    RDA2
1   0.89393 -0.4471
2   2.23895 -0.6637
3   1.86599  1.9795
4   1.08262  2.0321
5   1.45165 -1.2385
6   1.76693 -2.0860
7   1.85326 -1.5143
8   0.06223  1.5213
9   0.97741  1.4042
10  1.72119 -2.2735
11  0.46469 -2.1281
12 -0.24074  2.5504
13  0.40046  2.7709
14 -2.26001  0.1493
15 -2.52432  0.4329
16 -2.45355  2.6938
17 -1.40908 -1.4644
18 -0.84683 -1.9976
19 -1.98250 -2.1061
20 -3.06227  0.3849
```

How much variation is captured by each RDA axis: these two in the plot are already displaying 80% of the variation in the whole dataset

How important each site is in each RDA axis

The constraining variables (blue) do a good job of explaining the differences at the various sites (black)



Constrained Analysis on Principle Coordinates (CAP)

The **constrained equivalent of PCoA**. The ordination is constrained by a dataset of explanatory variables (they determine the axes). Data interpretation is visual.

Code is similar to previous.

- Same dataset as RDA

Results are similar to previous, but the plot looks different because Bray-Curtis distance was used (instead of Euclidean).
The unconstrained part of the analysis (red and black points in the small graph) similarly does not explain the data well. But the constraining variables (blue) do a good job of explaining the differences at the various sites (black)

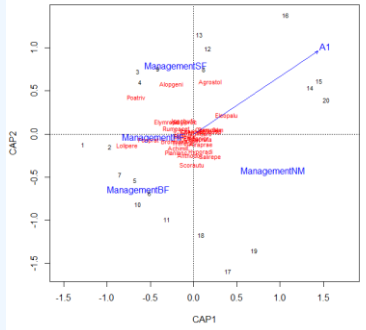
Run CAP:

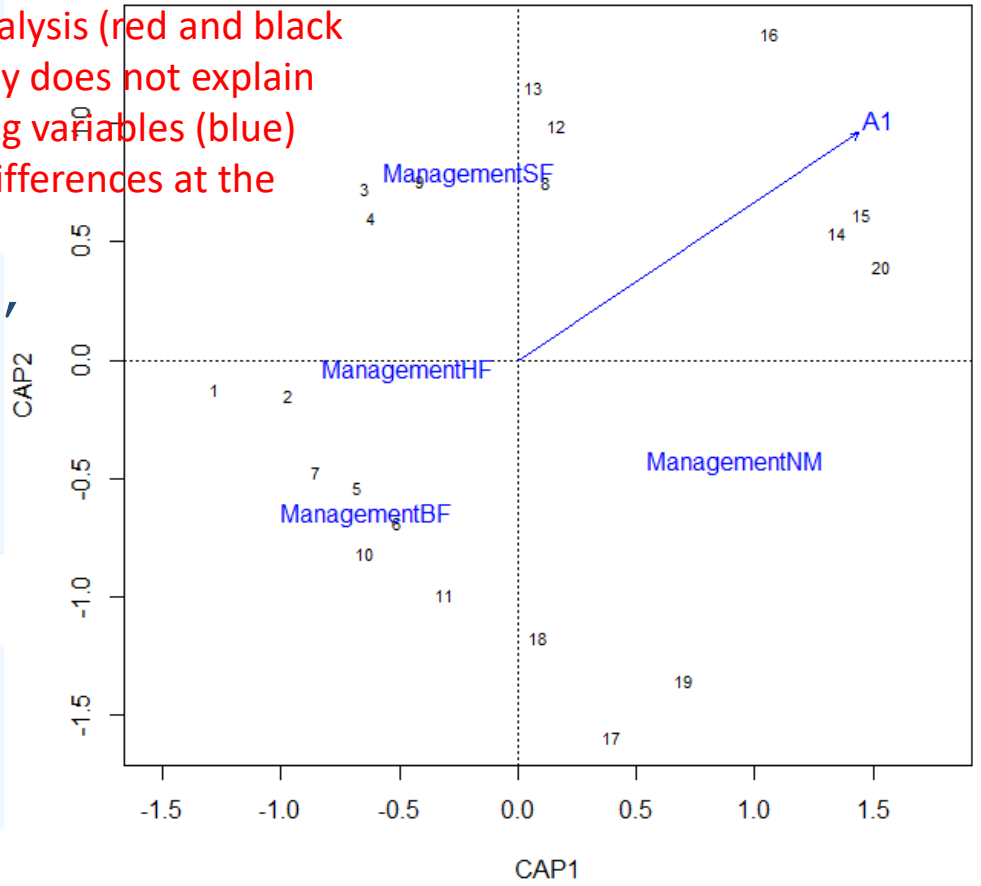
```
cap1=capscale(dune~Management+A1,data=dune.env,
distance="bray")
```

Recall in the PCoA, this part after the tilde is a "1" (reflecting the fact there are no explanatory variables in unconstrained ordination)

Display results:

```
plot(cap1)
plot(cap1,display=c("wa","cn"))
summary(cap1)
```





Canonical Correspondence Analysis (CCA)

The **constrained equivalent of CA**. The ordination is constrained by a dataset of explanatory variables (they determine the axes). Data interpretation is visual.

Same dataset as CA:

```
data(housetasks) #response contingency table
task.properties=data.frame(
  physical=c(4,3,3,3,4,4,5,3,7,3,3,9,3),
  math=c(F,F,F,F,F,F,T,T,F,T,T,F,F)
) #additional explanatory variables for each task
```

```
> housetasks
          Wife Alternating Husband Jointly
Laundry    156          14       2       4
Main_meal  124          20       5       4
Dinner      77          11       7      13
Breakfeast  82          36      15       7
Tidying     53          11       1      57
Dishes      32          24       4      53
Shopping    33          23       9      55
Official    12          46      23      15
```

Run CCA:

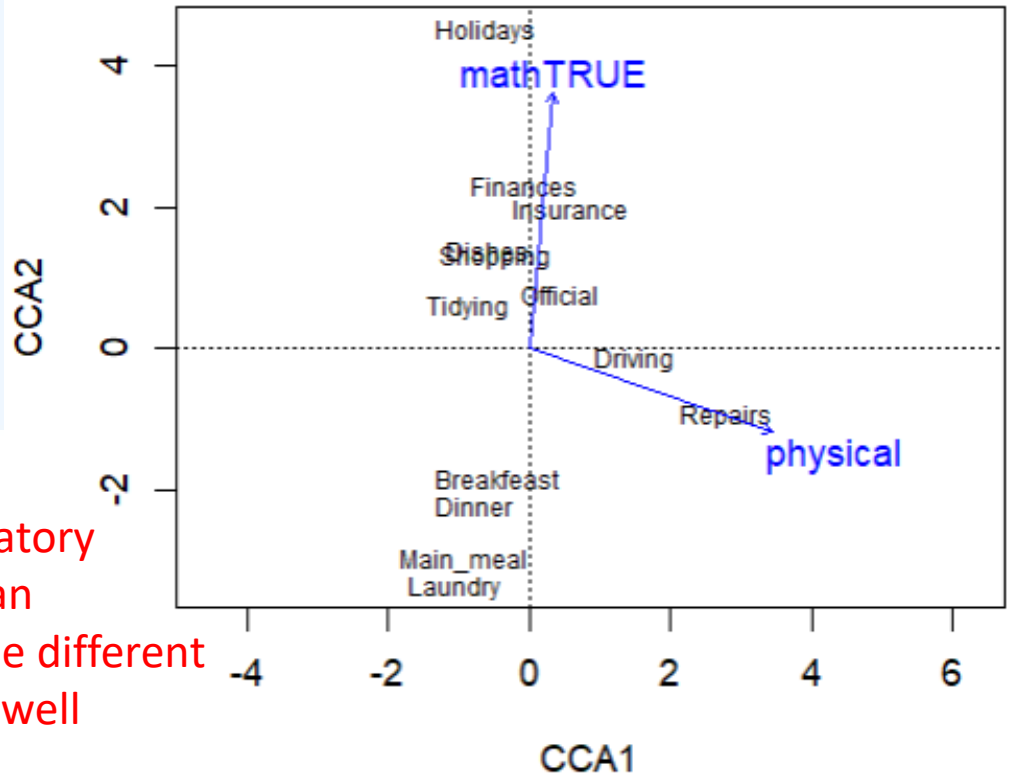
```
cca1=cca(housetasks~physical+math,data=task.properties)
```

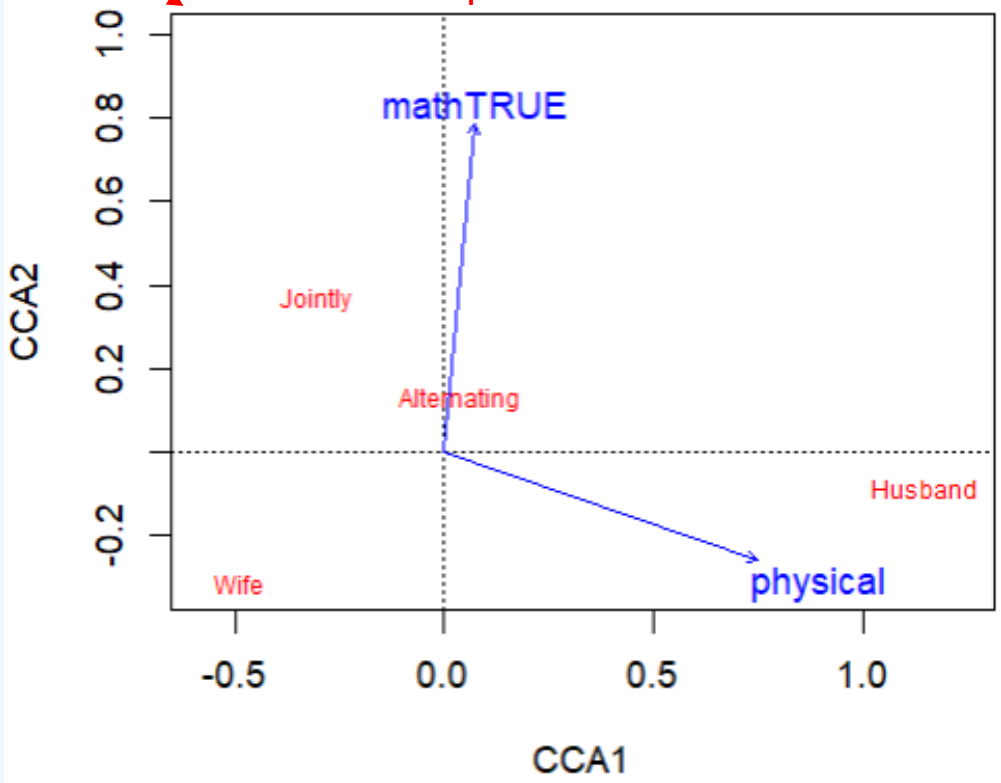
Canonical Correspondence Analysis (CCA)

Display results:

```
plot(cca1,display=c("sp","cn"))
plot(cca1,display=c("wa","cn"))
summary(cca1)
```

The constrained analysis suggest that:
- husbands do tasks that are more physical; and
- the couples jointly do tasks that require math





The explanatory variables can separate the different tasks quite well

“Linear Modelling”

MANOVA, PERMANOVA, MANCOVA, Multivariate GLM

"Linear modelling"-like approaches

Explaining more than 1 response variable (different types need different analyses) using 1 or more explanatory variables whilst accounting for the fact that the response variables may also interact.

Used to test a hypothesis, and therefore **will produce p-values**.

Analyses available are usually extensions of univariate analyses, with similar requirements and assumptions:

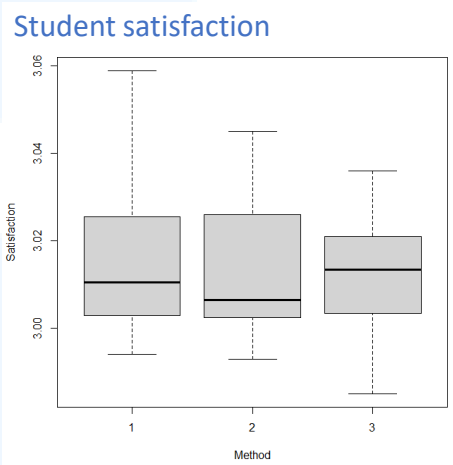
Univariate basis	Multivariate equivalent(s)	
ANOVA →	MANOVA →	PERMANOVA
ANCOVA →	MANCOVA	
GLM →	Multivariate GLM	

Multivariate ANOVA (MANOVA)

Explaining > 1 **normal continuous response** variable using ≥1 categorical variables.

Assumptions: similar to ANOVA.

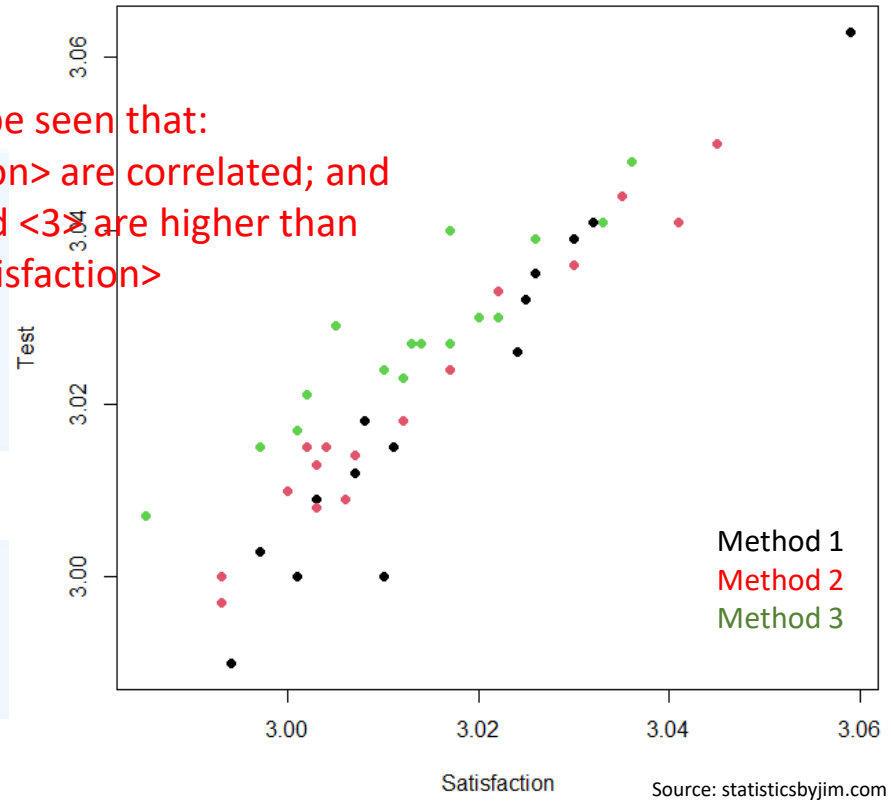
Each dependent variable does not seem to vary much amongst teaching methods. Individually, we would expect a non-significant result



Load dataset – teaching <Method> to explain student <Test> scores and <Satistfaction>:

```
d4=read.csv("testScores.csv")
```

When plotted together, it can be seen that:
i) <Test> scores and <Satisfaction> are correlated; and
ii) the <Test> scores for Method <3> are higher than Methods 1 & 2 for a given <Satisfaction>



Perform MANOVA:

```
manova1=manova(cbind(Test,Satisfaction)~
Method,data=d4)
summary(manova1)
```

```
> summary(manova1)
          Df  Pillai approx F num Df den Df    Pr(>F)
Method     1 0.45766   18.987      2     45 1.05e-06 ***
Residuals 46
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Note: cannot simplify using update().

Source: statisticsbyjim.com

Permutational MANOVA (PERMANOVA)

Any response variable type, one or more categorical explanatory variables. Tests many permutations of the data (i.e. rearranges them over and over again) and sees how many result in an increase or decrease in the measured "correlation".

Assumptions:

- Assumes objects in the datasets are exchangeable (i.e. are independent and have similar amounts of dispersion), e.g. if the values are very different, you may have to scale them first

- Does not assume any distribution; is insensitive to multicollinearity; allows for multiple variables; is insensitive to data with many zeros

Note: ANOSIM does something similar (although slightly different) but can only take 1 categorical variable, so I do not cover it. You can learn about it here: https://jkzorz.github.io/2019/06/11/ANOSIM-test.html.

Permutational MANOVA (PERMANOVA)

Load dataset:

```
require(vegan)
data(dune) #the response variables, counts of different plant species
data(dune.env) #contains explanatory variables <Management> and <Moisture>
```

Make sure to choose the correct type of distance for your response variables (Google is your friend)

Perform analysis:

```
perm1=adonis(dune~Management*Moisture,data=dune.env,method="euclidean")
perm1
```

```
> perm1

Call:
adonis(formula = dune ~ Management * Moisture, data = dune.env,      method = "euclidean")

Permutation: free
Number of permutations: 999

Terms added sequentially (first to last)

                  Df SumsOfSqs MeanSqs F.Model      R2 Pr(>F)
Management         3    555.38 185.128  3.3438 0.34747  0.001 ***
Moisture           3    326.69 108.897  1.9669 0.20439  0.007 **
Management:Moisture  5    273.36  54.672  0.9875 0.17103  0.493
Residuals          8    442.92  55.365         0.27711
Total             19   1598.35                 1.00000
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Continue to simplify manually according to what we have learnt (`update()` works!)

Multivariate ANCOVA (MANCOVA)

Explaining > 1 **normal continuous variable** using one or more categorical and/or continuous variables.

Run a MANCOVA using the mtcars dataset:

```
require(jmv)

man1=mancova(mtcars,deps=vars(qsec,hp),
factors=vars(am,vs),covs=vars(drat,mpg))

man1
```

Response ("dependent") variables

Categorical ("factor") explanatory variables

Continuous ("covariates") explanatory variables

No need to simplify, just use results

Note: PERMANCOVA is not available in R (have to use PRIMER), you can use multivariate GLM instead.

Different ways of calculating significance. Pillai's trace is usually considered the most reliable

<am> has a significant effect on both <qsec> and <hp> together

```
> man1

MANCOVA

Multivariate Tests
```

		value	F	df1	df2	p
am	Pillai's Trace	0.43453995	9.6058940	2	25	0.0008036
	Wilks' Lambda	0.5654601	9.6058940	2	25	0.0008036
	Hotelling's Trace	0.76847152	9.6058940	2	25	0.0008036
	Roy's Largest Root	0.76847152	9.6058940	2	25	0.0008036
vs	Pillai's Trace	0.72974539	33.7526794	2	25	< .0000001
	Wilks' Lambda	0.2702546	33.7526794	2	25	< .0000001
	Hotelling's Trace	2.70021435	33.7526794	2	25	< .0000001
	Roy's Largest Root	2.70021435	33.7526794	2	25	< .0000001
am:vs	Pillai's Trace	0.01422425	0.1803688	2	25	0.8360392
	Wilks' Lambda	0.9857757	0.1803688	2	25	0.8360392
	Hotelling's Trace	0.01442950	0.1803688	2	25	0.8360392
	Roy's Largest Root	0.01442950	0.1803688	2	25	0.8360392
drat	Pillai's Trace	0.03144342	0.4058026	2	25	0.6707527
	Wilks' Lambda	0.9685566	0.4058026	2	25	0.6707527
	Hotelling's Trace	0.03246421	0.4058026	2	25	0.6707527
	Roy's Largest Root	0.03246421	0.4058026	2	25	0.6707527
mpg	Pillai's Trace	0.37520672	7.5066172	2	25	0.0027971
	Wilks' Lambda	0.6247933	7.5066172	2	25	0.0027971
	Hotelling's Trace	0.60052937	7.5066172	2	25	0.0027971
	Roy's Largest Root	0.60052937	7.5066172	2	25	0.0027971

Univariate Tests

	Dependent Variable	Sum of Squares	df	Mean Square	F	p
am	qsec	5.230139474	1	5.230139474	4.924711499	0.0354097
	hp	8619.498481781	1	8619.498481781	5.451895339	0.0275391
vs	qsec	62.495550579	1	62.495550579	58.845955849	< .0000001
	hp	69789.621491764	1	69789.621491764	44.142442034	0.0000005
am:vs	qsec	0.152447177	1	0.152447177	0.143544617	0.7078587
	hp	577.091258620	1	577.091258620	0.365014409	0.5509704
drat	qsec	0.002548780	1	0.002548780	0.002399938	0.9613023
	hp	963.937223484	1	963.937223484	0.609697289	0.4419536
mpg	qsec	3.494957886	1	3.494957886	3.290860478	0.0812239
	hp	24670.478788701	1	24670.478788701	15.604256859	0.0005322
Residuals	qsec	27.612506104	26	1.062019466		
	hp	41106.247755650	26	1581.009529063		

Looking at each response variable individually: <mpg> has a significant effect on <hp> but not on <qsec> (marginally)

Multivariate GLM

To more than one response variable of various types (must be the same type within the analysis) using one or more categorical and/or continuous explanatory variables. No random effects.

Load dataset:

```
require(mvabund) #also needed for the manyglm() function
data(spider) #a List with 2 datasets in it: “abund” and “x”
```

Response variables: counts of 12 different spiders

```
Y=as.matrix(spider$abund) #must be matrix
```

Explanatory variables: 6 environmental variables

```
X=as.data.frame(spider$x) #must be dataframe to use the variables in it
```

Multivariate GLM

Matrix of response variables

Explanatory variables from the X dataframe, can take interactions and nesting

Perform the multivariate GLM:

```
mod1=manyglm(Y~soil.dry*reflection+fallen.leaves/moss,data=X,
family="poisson")
summary(mod1) #for simplification (according to what we've learnt)
```

Choose the error distribution based on the same rules in GLM. Default is negative binomial. Note: no "quasi" distributions yet.

Check for heteroscedasticity:

```
plot(mod1)
```

Used to calculate p-value

```
> summary(mod1)

Test statistics:
                      wald value Pr(>wald)
(Intercept)             14.99      0.001 ***
soil.dry                13.62      0.003 **
reflection              15.83      0.002 **
fallen.leaves           24.99      0.001 ***
soil.dry:reflection     17.25      0.001 ***
fallen.leaves:moss      15.86      0.005 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Interpretation of results is usually done visually, by plotting the different groupings with different colours (e.g. using a PCA and grouping based on the levels in the significant explanatory variables). It gets complicated!

- For more info (very well written), see: help(manyglm).

Note: To fit Gaussian errors, use manylm()

Note: although it is possible to use AIC() to compare different models (e.g. for simplification), I personally find the results to be unreliable so I currently don't suggest you use it.

Bayesian Statistics

A (very) gentle introduction from my viewpoint

https://www.youtube.com/watch?v=GEFxFVESQXc

Frequentist viewpoint

In experiments, we assume that the data follow a given distribution, then we gauge **how well our data represent the truth** by…

$$P(\text{Hypothesis}|\text{Evidence}) = P(\text{Evidence}|\text{Distribution})$$

Probability that the hypothesis is true given the evidence that we have

Probability of observing our data given that the population and/or errors follow the assumed distribution. This is related to our p-value.

The crux of the matter:

- We assume that there is a “truth” out there (e.g. Lecturers are more handsome than Pilots).
- We go out and collect the data (measure the good-looking score of Lecturers and Pilots).
- Then we say: **given the data** we have **and the distribution** we assume the population follows, it is **very likely true** (p-value < 0.001) that Lecturers are more handsome than Pilots

Bayesian viewpoint: Bayes' rule

Basis of Bayesian statistics:

$$P(\text{Hypothesis}|\text{Evidence}) = \frac{P(\text{Evidence}|\text{Hypothesis}) \cdot P(\text{Hypothesis})}{P(\text{Evidence})}$$

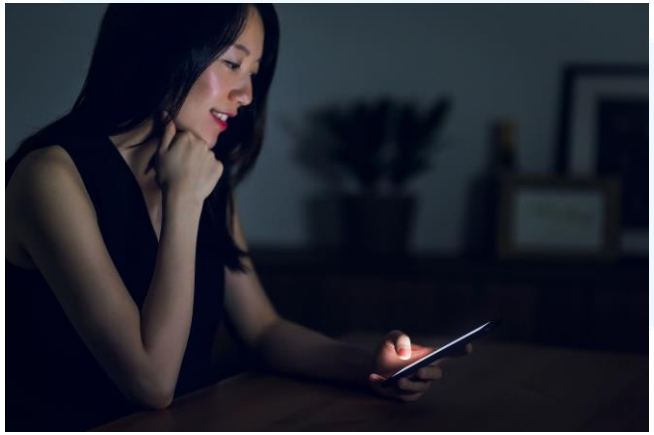
Probability that the hypothesis is true given the evidence that we have. This is known as the **POSTERIOR**.

What we previously believed about whether the hypothesis is true. This is known as the **PRIOR**.

We update our PRIOR belief based on evidence to get our new POSTERIOR belief.

Example: You're going on a blind date.

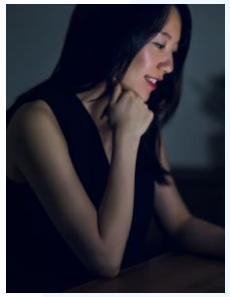
You love Star Wars! You wonder whether your date does too.



Bayesian viewpoint: Bayes' rule

Disclaimer: Don't look too closely at the numbers. The calculations in this example have been simplified and massaged to get a message across.

You think it's maybe a 50% chance that a random person likes Star Wars.



Your **POSTERIOR belief**: what you think now that you've seen your new data

P(Evidence|Hypothesis)

P(Hypothesis)
The PRIOR belief.

$$P(\text{Hypothesis}|\text{Evidence}) = \frac{0.99 \cdot 0.5}{0.6}$$

P(Evidence)

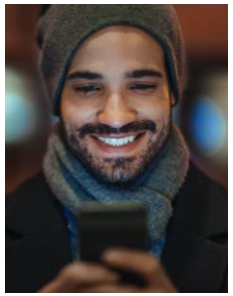
You don't want to be too obvious, so you say:

"I just watched The Rise of Skywalker? Have you seen it?"

Your date replies: *"YES, I have! It was great!"*

You get excited!! You think that the probability that someone would have seen the movie given that they're a fan is 99%! (Some may be in comas.)

But wait! What's the overall likelihood that someone went to watch the movie in the first place? It was a bit of a flop, so let's say 60%.



So you now believe the chance your date is a fan is … **82.5%**!! Love at first sight!

Bayesian viewpoint

In typical experiments, *P*(Evidence) is a constant, so we can reduce Bayes' equation to:

$$P(\text{Hypothesis}|\text{Evidence}) = P(\text{Evidence}|\text{Hypothesis}) \cdot P(\text{Hypothesis})$$

Probability that the hypothesis is true given the evidence that we have. This is known as the **POSTERIOR**.

The probability of getting our data given that the hypothesis is true. We calculate this from our experiment.

What we previously believed about whether the hypothesis is true. This is known as the **PRIOR**.

The crux of the matter:

- We are **updating a previous belief** using the new evidence we just collected.

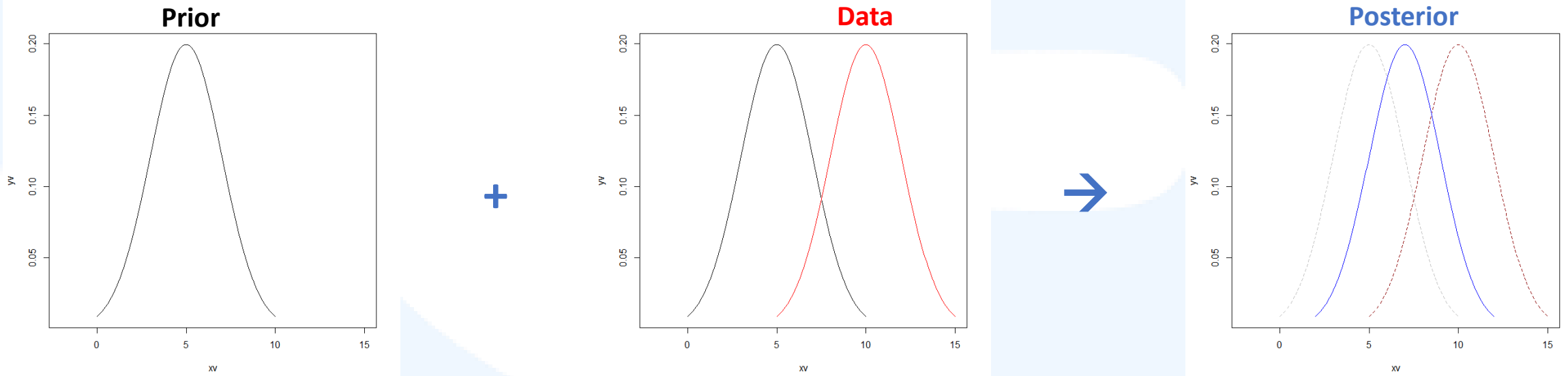
- We are **not specifically interested in how well our data represents a "truth"**, hence there's usually no p-value.

- We just interpret the new estimates/effect sizes.

Bayesian statistical methods

There are Bayesian versions for many of the advanced and multivariate analyses

Whenever we run a Bayesian analysis, we need to define a prior.

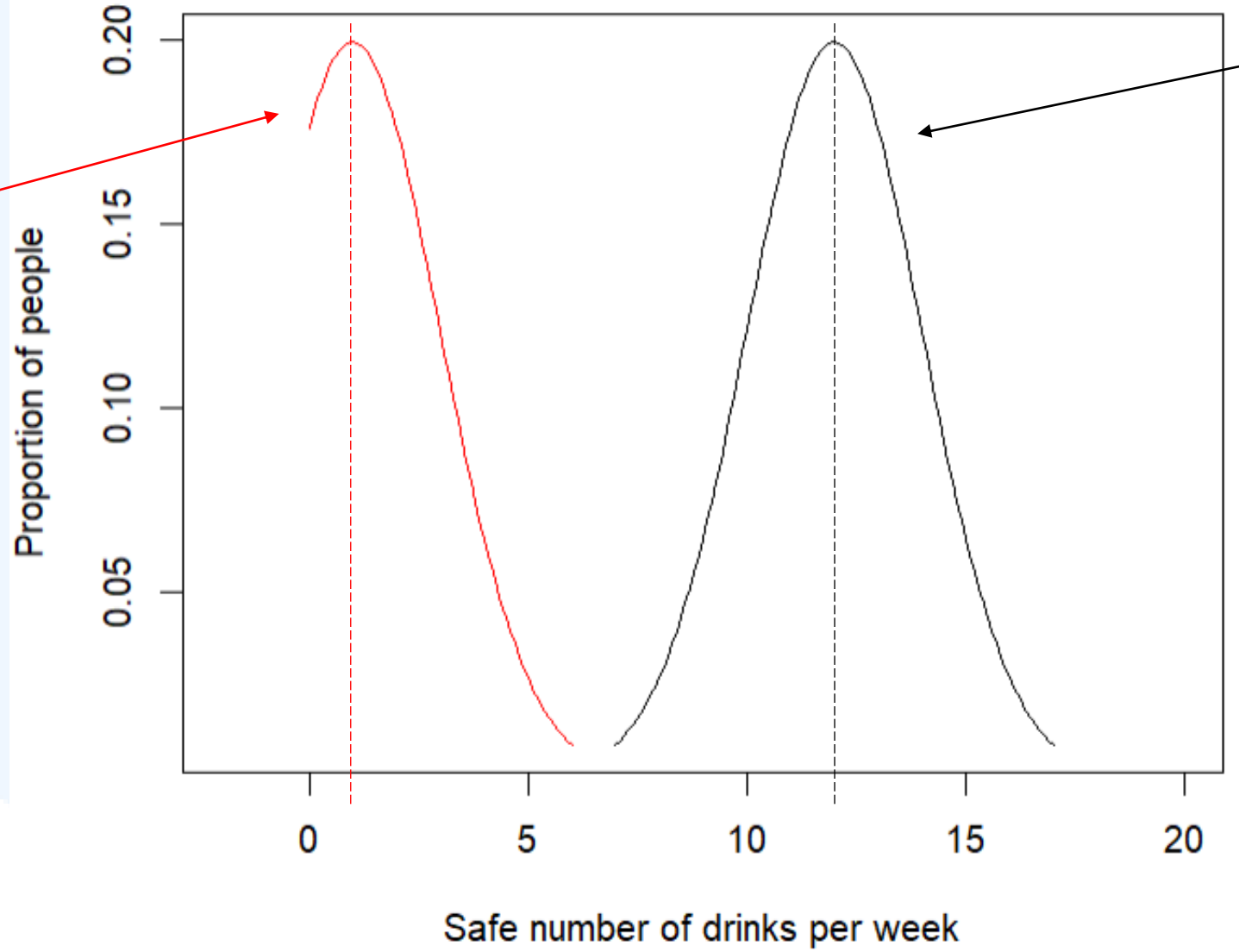


Using the data, the analysis will then give us the final posterior through repeated trial and error (e.g. Markov Chain Monte Carlo aka MCMC).

- Recall: Some R functions (e.g. stan_glmer) can supply the prior for us based on our error family, but for others (e.g. MCMCglmm), we need to learn how to do it ourselves.

Example: how many drinks per week is safe?

New data: the latest experiment shows (with statistically significant data) that the safe number is 2 on average



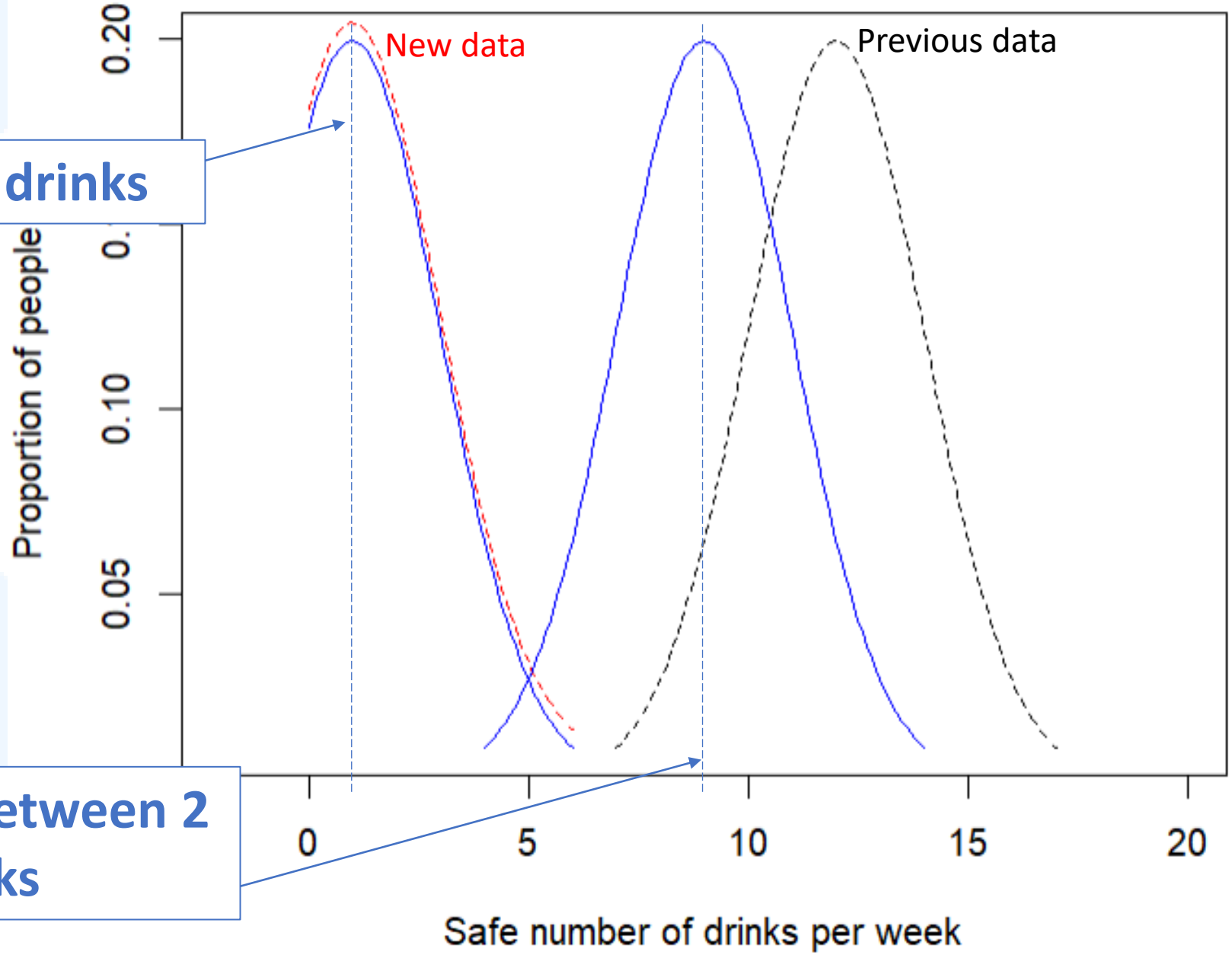
Previous data: based on existing experiments, the consensus is that the safe number is 12 on average

Example: how many drinks per week is safe?

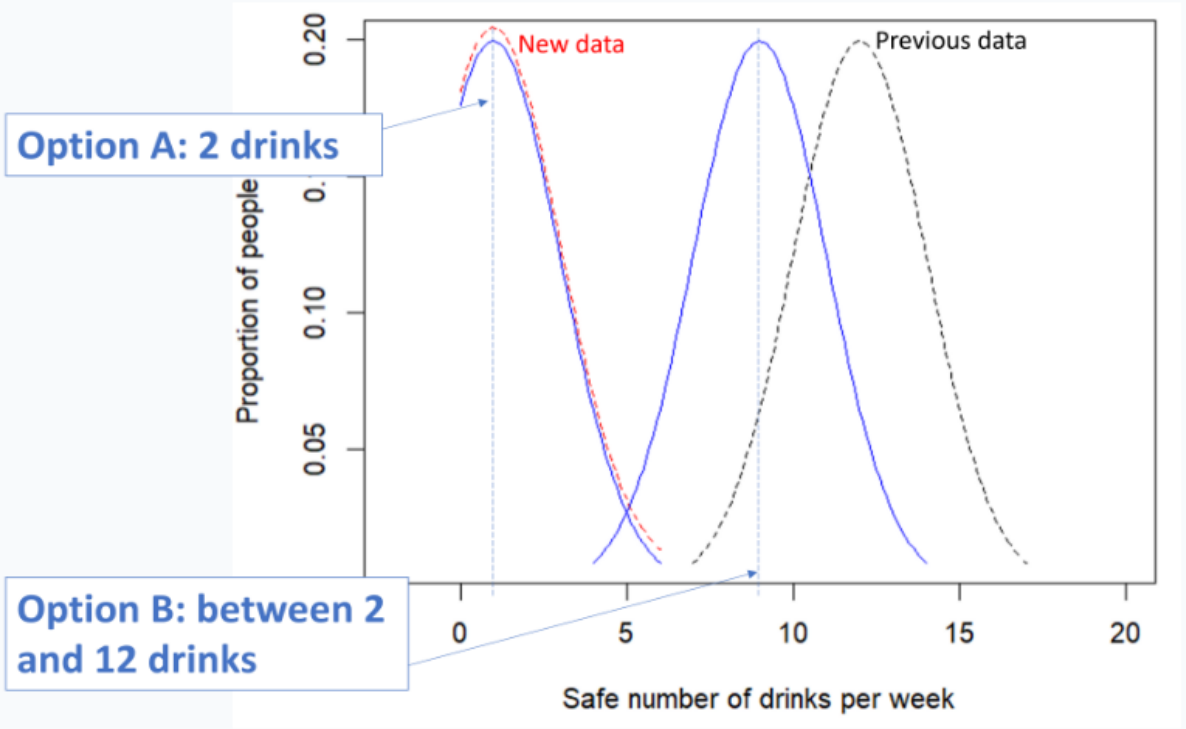
What is your verdict?

Option A: 2 drinks

Option B: between 2 and 12 drinks



What's your verdict?



Option A

Option B

Example: how many drinks per week is safe?

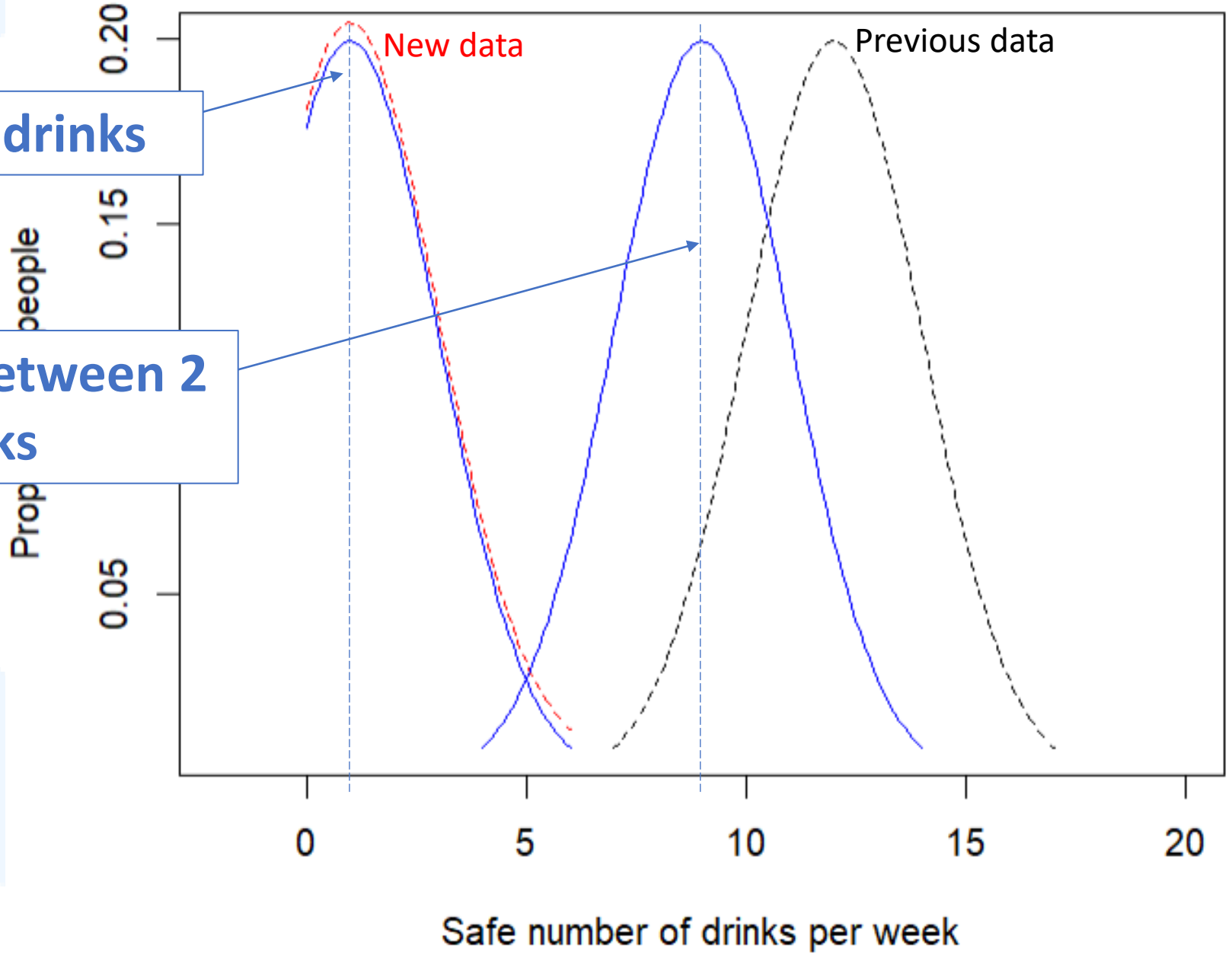
A is the Frequentist view

Option A: 2 drinks

B is the Bayesian view

Option B: between 2 and 12 drinks

You could be Frequentist in some situations and Bayesian in others: both have pros and cons



Fitting a Bayesian GLM

Example using stan_glm()

If published studies suggest that your y-variable's:

- Intercept follows a normal distribution of mean = 0 and S.D. = 1
- Slope follows a normal distribution of mean = 3 and S.D. = 0.5

Then you fit the model like this:

```
mod1=stan_glm(Y~X,data=d1,family=Gamma,prior_intercept=normal(0,1),
prior=normal(3,0.5))
```

The prior for the intercept

The prior for the slope

More information on priors in "rstanarm"

- ?normal #after loading rstanarm
- Read more here.

```
Usage:

     normal(location = 0, scale = NULL, autoscale = FALSE)

     student_t(df = 1, location = 0, scale = NULL, autoscale = FALSE)

     cauchy(location = 0, scale = NULL, autoscale = FALSE)

     hs(df = 1, global_df = 1, global_scale = 0.01, slab_df = 4, slab_scale = 2.5)

     hs_plus(
       df1 = 1,
       df2 = 1,
       global_df = 1,
       global_scale = 0.01,
       slab_df = 4,
       slab_scale = 2.5
     )

     laplace(location = 0, scale = NULL, autoscale = FALSE)
```

Summary (Learning Objectives)

Non-linear Modelling

- GAM

Multivariate Statistics

- Theory: Response variables, purposes, dissimilarity matrices
- Understanding structure
    - Clustering: AHC, PAM, K-means
    - Unconstrained ordination: PCA, PCoA, NMDS, CA
- Interpreting/Making predictions
    - Constrained ordination: RDA, CAP, CCA
    - “Modelling”: MANOVA, PERMANOVA, MANCOVA & Multivariate GLM

Bayesian Statistics

- Bayes’ Rule and a stan_glm() example

Overview

LSM3257

AY22/23; Sem 2 | Ian Z.W. Chan

BL5233 at a glance

A) If you have only one response variable: univariate statistics

- If you have simple controlled experiments:

Basic Tests

- If you have complex studies with potential confounding effects:

Advanced Analyses

(including non-linear modelling)

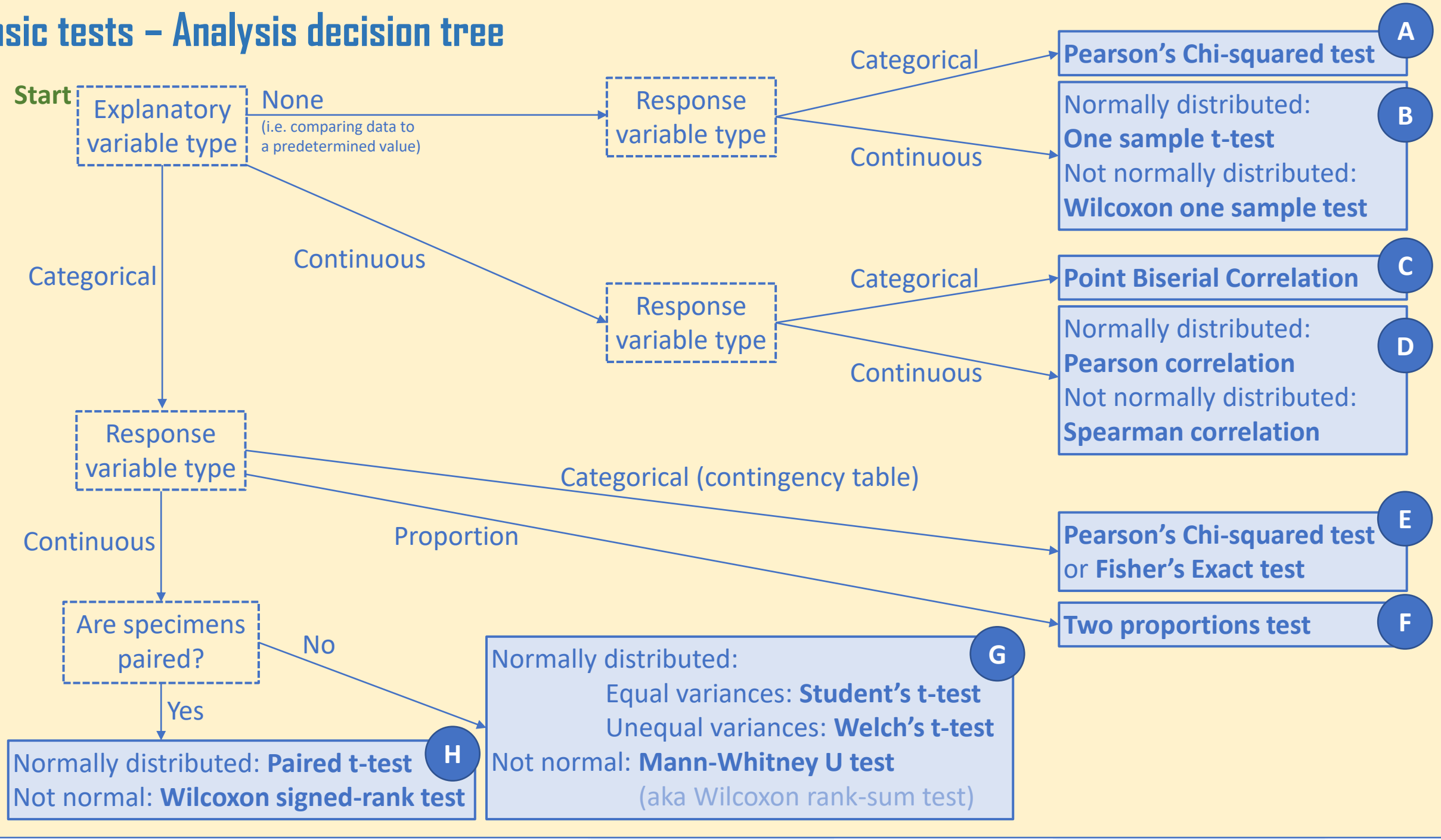
(has Bayesian alternatives)

B) If you have more than one response variable: multivariate statistics

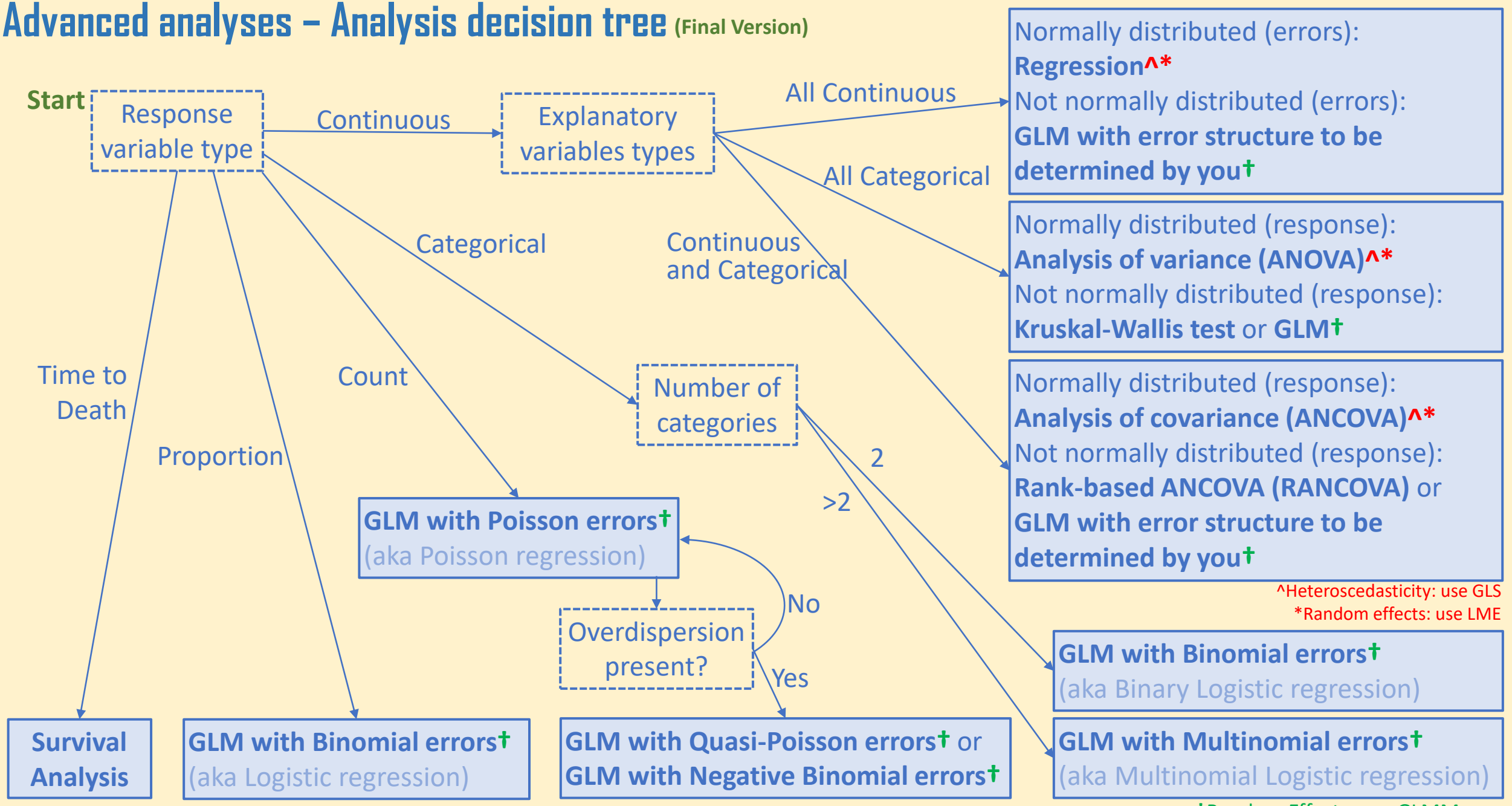
Multivariate Analyses

(has Bayesian alternatives)

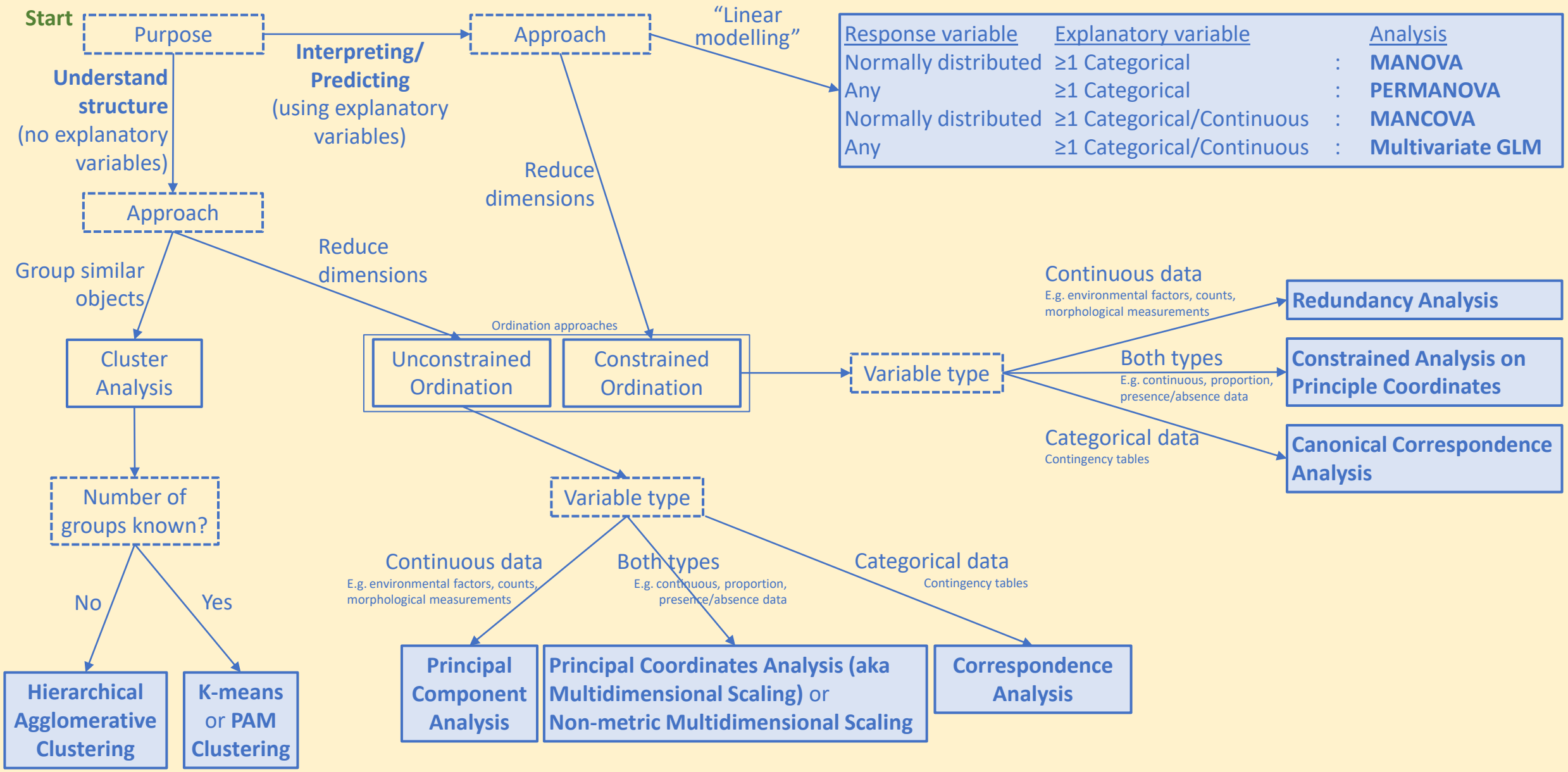
Basic tests – Analysis decision tree



Advanced analyses – Analysis decision tree (Final Version)



Multivariate analyses – Analysis decision tree

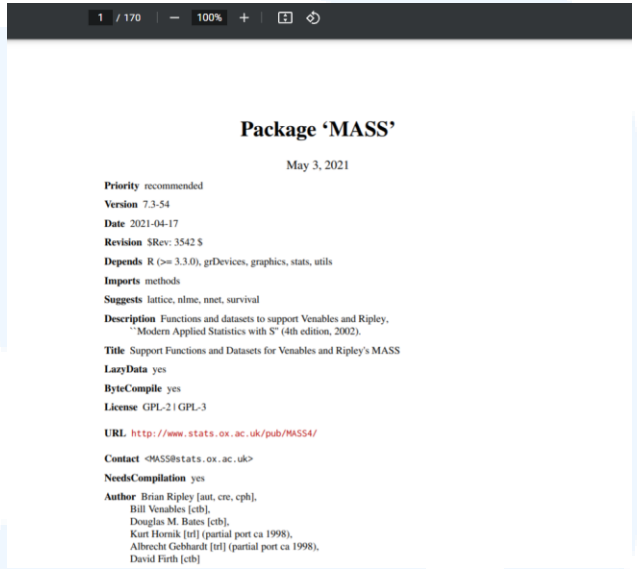


Further information: Now the learning moves to your home/office!

For specific questions

1) Built-in R help:

```
help(glmmPQL)
?glmmPQL #same thing as help()
??glmmPQL #if you haven't installed the package yet
```



2) Vignettes of packages:

The vignette contains the documentation written by the authors. It's the best bet for getting information and finding out what the function can do, e.g. glmmPQL() is from the "MASS" package, so google: "MASS vignette R".

3) Forum posts, e.g. https://stats.stackexchange.com/

For general knowledge

Special interest groups: on Facebook, in the R community.

SEARCH and READ...

then

be BRAVE and TRY!!!

AC2

Released tonight; Material from Weeks 7 to 9

Due 31 Mar, 2359h

OPEN UNIVERSE; BUT PLEASE DO NOT DISCUSS

Projects

Project Presentation (Group)

10-min presentation, followed by Q&A (about 5 mins)

- Weeks 12 and 13: you're only required to attend the week that you're presenting

Project Presentation Timings (approximate)					
Week 12					
1000	Group 2	Jian Xi	May Ching	Huile	Yin Chuan
1020	Group 16	Jerome	Jin Chi	Zhi Cheng	
1040	Group 14	Benedick	Shin Yin	Kaizeng	Victoria
	Break				
1110	Group 18	Wen Han	Yee Qi	Amanda	Benjamin
1130	Group 17	Clara	Lixuan	Kelly	Alicia
1150	Group 12	Samuel	Ophelia	Sarita	Vera
	Break				
1220	Group 8	Ho Ning	Shuna	Dana	
1240	Group 6	Cian Jin	Ryan	Gen Fong	Min Xian
1300	Group 3	Sherry	Anna	Sarah-Ann	Jia Wei
1320	Group 20	Phoebe	Salman	Han Lin	Clive

Week 13					
1000	Group 7	Judith	Regina	Wen Xin	Yan Zhi
1020	Group 4	Shannon	Iliya	Isaac	Kate-Lyn
1040	Group 5	Justin	Wee Meng	Jia Le	Jing Wei
	Break				
1110	Group 19	Ruth	Kendrick	Gen Koh	Wei Kai
1130	Group 13	Ler Shan	Maryam	Diya	
1150	Group 1	Kimberly	Sin Yu	Wan Ling	Jing Min
	Break				
1220	Group 11	Jun Ning	Raine	Michelle	
1240	Group 9	Vicki	Divina	Si Ying	Rachel
1300	Group 15	Choon	Sophia	Boon Hao	
1320	Group 10	Kai Le	Hao Yu	Danish	Whelan

- Week 11: consultations, email me to arrange

Project Report (Individual)

Deadline: 28 Apr 2023, 2359h

Scientific Communication tips

Remember you're telling a story (in BOTH presentations and reports)

Stories need to **be interesting** so each section in a Research paper/report is designed to help you tell the story…

- Tell us about the existing situation and knowledge: Background
- You run into a problem: Research Question
- What did you do to solve it?: Materials & Methods
    - How the authors (of the original dataset) collected the data (brief): context for understanding
    - How you analysed the data: what variables and analysis did you use?
- The climax of the story—how you saved the day!: Results
- The happy-ever-after: Discussion/Conclusions

Presentation tips

Design your slides to tell the story: each slide communicates one or two points and advances the story

Use as few words as possible in each slide

- Main points are written, your explanations are verbal
- My lecture slides are NOT a good example because they're designed as hybrid presentation and notes for your future use

Choose the right visuals to illustrate your points

Practise, practise, practise!

- Decide how you want to say things in advance
- Keep within the time limit: usually 1 slide = about 1 minute

Teammate grading is available upon request (as a last resort)

IMPORTANT: Asking other groups (friendly and constructive) questions is part of the grading rubric!!

Report tips

Again: tell the story!

- The Title is one sentence summarising the main result from your whole story

- The Abstract is a TL;DR form of your report (don't copy and paste from your proposal): it should have 2 sentences of Introduction/Research Question, 2 sentences of M&Ms, 2 sentences of Results and 1 sentence of Discussion/Conclusion

- The rest of the sections follow the purposes described above

Refer to published journal articles and try to follow them:

- The sections and what information is in each section

- The referencing

- The professional tone of language

From Presentation to Report
- Your M&Ms and Results can be similar to your group members (but NOT word-for-word identical)
- Your Abstract, Introduction and Discussion should be quite different (even if the main points are similar)
- Bottomline: So long as you sit down and write your report yourself, it will be fine

Proof-read to eliminate typos and grammatical errors