

Statistics for genomics

Mayo-Illinois Computational Genomics Course

June 11, 2019

Dave Zhao
Department of Statistics
University of Illinois at Urbana-Champaign

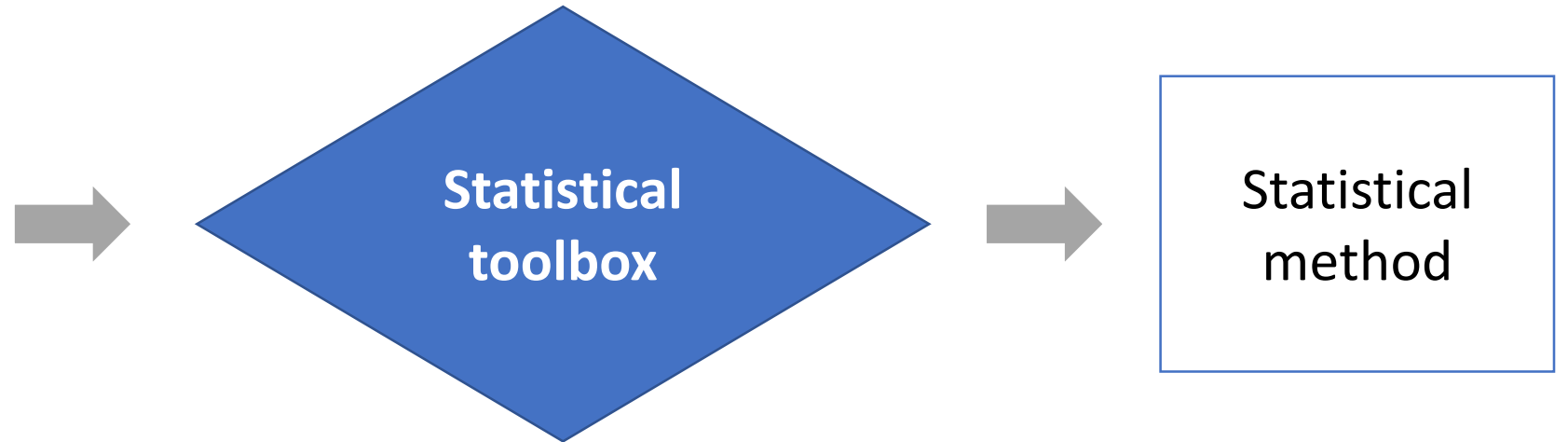
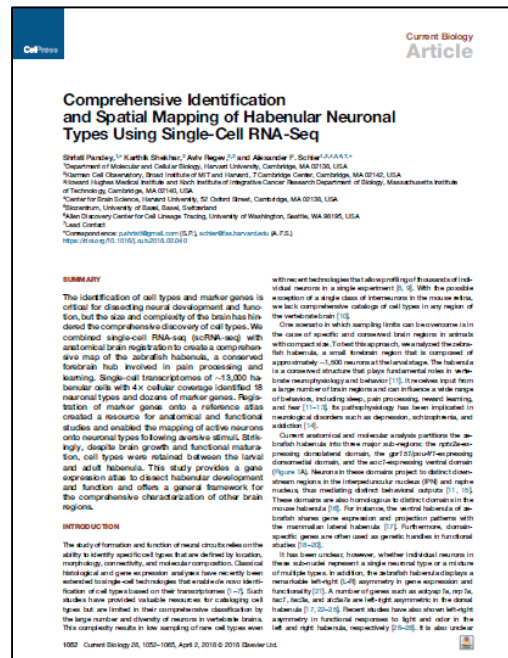


Preparation

- `install.packages(c("Seurat", "glmnet", "ranger", "caret"))`
- Download sample GSM2818521 of GSE109158 from <https://urlzs.com/7UNr6>

Objective

We will illustrate how to identify the appropriate statistical method for a genomics analysis



Classifying statistical tools

Data structure

Statistical task

	No dependent variables	Continuous outcome	Censored outcomes	Etc.
Visualize				
Identify latent factors				
Cluster observations				
Select features				
Etc.				

**APPROPRIATE
STATISTICAL
METHODS**

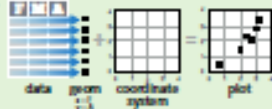
Examples from basic statistics

Data Visualization with ggplot2

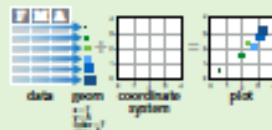


Basics

ggplot2 is based on the **grammar of graphics**, the idea that you can build every graph from the same few components: a **data** set, a set of **geoms**—visual marks that represent data points, and a **coordinate system**.



To display data values, map variables in the data set to aesthetic properties of the geom like **size**, **color**, and **x** and **y** locations.



Build a graph with `ggplot()` or `qplot()`

`ggplot(data = mpg, aes(x = cty, y = hwy))`

Begins a plot that you finish by adding layers to. No defaults, but provides more control than `qplot()`.

`ggplot(mpg, aes(hwy, cty)) +
 geom_point(aes(color = cyl)) +
 geom_smooth(method = "lm") +
 coord_cartesian() +
 scale_color_gradient() +
 theme_bw()`

add layers, elements with +
layer = geom + default stat + layer-specific mappings
additional elements

Add a new layer to a plot with a `geom_*()` or `stat_*()` function. Each provides a geom, a set of aesthetic mappings, and a default stat and position adjustment.

`qplot(x = cty, y = hwy, color = cyl, data = mpg, geom = "point")`
Creates a complete plot with given data, geom, and mappings. Supplies many useful defaults.

`last_plot()`
Returns the last plot

`ggsave("plot.png", width = 5, height = 5)`
Saves last plot as 5" x 5" file named "plot.png" in working directory. Matches file type to file extension.

Geoms - Use a geom to represent data points, use the geom's aesthetic properties to represent variables. Each function returns a layer.

One Variable

Continuous

`a <- ggplot(mpg, aes(hwy))`

`a + geom_area(stat = "bin")`
x, y, alpha, color, fill, linetype, size
b + `geom_area(aes(y = ..density..), stat = "bin")`

`a + geom_density(kernel = "gaussian")`
x, y, alpha, color, fill, linetype, size, weight
b + `geom_density(aes(y = ..county..))`

`a + geom_dotplot()`
x, y, alpha, color, fill

`a + geom_freqpoly()`
x, y, alpha, color, linetype, size
b + `geom_freqpoly(aes(y = ..density..))`

`a + geom_histogram(binwidth = 5)`
x, y, alpha, color, fill, linetype, size, weight
b + `geom_histogram(aes(y = ..density..))`

Discrete

`b <- ggplot(mpg, aes(fill))`

`b + geom_bar()`
x, alpha, color, fill, linetype, size, weight

Graphical Primitives

`map <- map_data("state")`

`c <- ggplot(map, aes(long, lat))`

`c + geom_polygon(aes(group = group))`
x, y, alpha, color, fill, linetype, size

`d <- ggplot(economics, aes(date, unemployment))`

`d + geom_path(lineend = "butt",
 linejoin = "round", linemitre = 1)`
x, y, alpha, color, linetype, size

`d + geom_ribbon(aes(ymin = unemployment - 900,
 ymax = unemployment + 900))`
x, ymax, ymin, alpha, color, fill, linetype, size

`e <- ggplot(seals, aes(x = long, y = lat))`

`e + geom_segment(aes(xend = long + delta_long,
 yend = lat + delta_lat))`
x, xend, y, yend, alpha, color, linetype, size

`e + geom_rect(aes(xmin = long, ymin = lat,
 xmax = long + delta_long,
 ymax = lat + delta_lat))`
xmax, xmin, ymax, ymin, alpha, color, fill, linetype, size

Two Variables

Continuous X, Continuous Y

`f <- ggplot(mpg, aes(cty, hwy))`

`f + geom_blank()`
(Useful for expanding limits)

`f + geom_jitter()`
x, y, alpha, color, fill, shape, size

`f + geom_point()`
x, y, alpha, color, fill, shape, size

`f + geom_quantile()`
x, y, alpha, color, linetype, size, weight

`f + geom_rug(sides = "bl")`
alpha, color, linetype, size

`f + geom_smooth(method = "lm")`
x, y, alpha, color, fill, linetype, size, weight

`f + geom_text(aes(label = cty))`
x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

Discrete X, Continuous Y

`g <- ggplot(mpg, aes(class, hwy))`

`g + geom_bar(stat = "identity")`
x, y, alpha, color, fill, linetype, size, weight

`g + geom_boxplot()`
lower, middle, upper, x, ymax, ymin, alpha, color, fill, linetype, shape, size, weight

`g + geom_dotplot(binaxis = "y",
 stackdir = "center")`
x, y, alpha, color, fill

`g + geom_violin(scale = "area")`
x, y, alpha, color, fill, linetype, size, weight

Discrete X, Discrete Y

`h <- ggplot(diamonds, aes(cut, color))`

`h + geom_jitter()`
x, y, alpha, color, fill, shape, size

Continuous Bivariate Distribution

`i <- ggplot(movies, aes(year, rating))`

`i + geom_bin2d(binwidth = c(5, 0.5))`
xmax, xmin, ymax, ymin, alpha, color, fill, linetype, size, weight

`i + geom_density2d()`
x, y, alpha, colour, linetype, size

`i + geom_hex()`
x, y, alpha, colour, fill size

Continuous Function

`j <- ggplot(economics, aes(date, unemployment))`

`j + geom_area()`
x, y, alpha, color, fill, linetype, size

`j + geom_line()`
x, y, alpha, color, linetype, size

`j + geom_step(direction = "hv")`
x, y, alpha, color, linetype, size

Visualizing error

`df <- data.frame(grp = c("A", "B"), fit = 4.5, se = 1:2)`
`k <- ggplot(df, aes(grp, fit, ymin = fit-se, ymax = fit+se))`

`k + geom_crossbar(fatten = 2)`
x, y, ymax, ymin, alpha, color, fill, linetype, size

`k + geom_errorbar()`
x, ymax, ymin, alpha, color, linetype, size, width (also `geom_errorbarh()`)

`k + geom_linerange()`
x, ymin, ymax, alpha, color, linetype, size

`k + geom_pointrange()`
x, y, ymin, ymax, alpha, color, fill, linetype, shape, size

Maps

`data <- data.frame(murder = USArrests$Murder,
 state = tolower(row.names(USArrests)))`

`map <- map_data("state")`

`l <- ggplot(data, aes(fill = murder))`

`l + geom_map(aes(map_id = state), map = map) +
 expand_limits(x = map$long, y = map$lat)`
map_id, alpha, color, fill, linetype, size

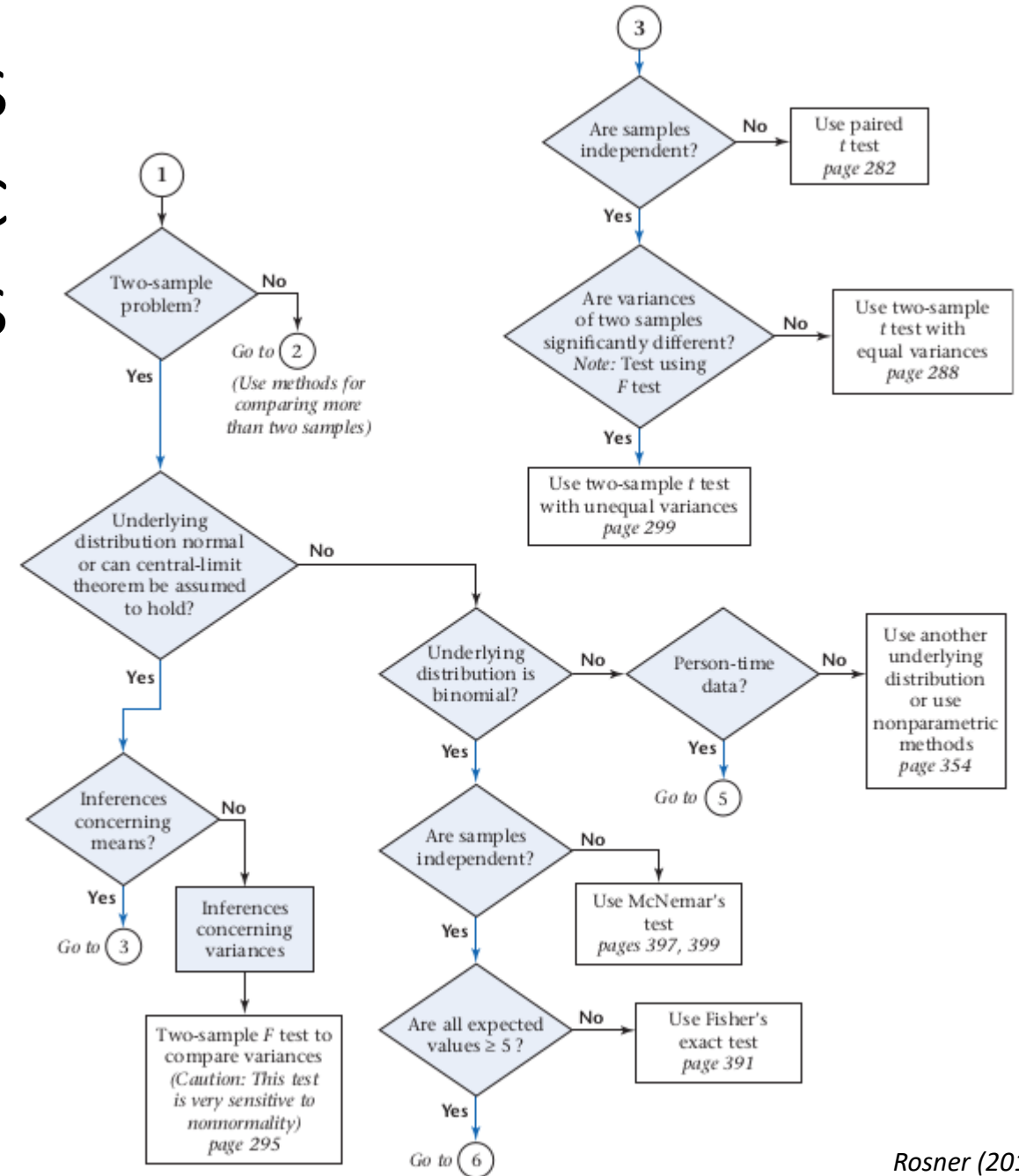
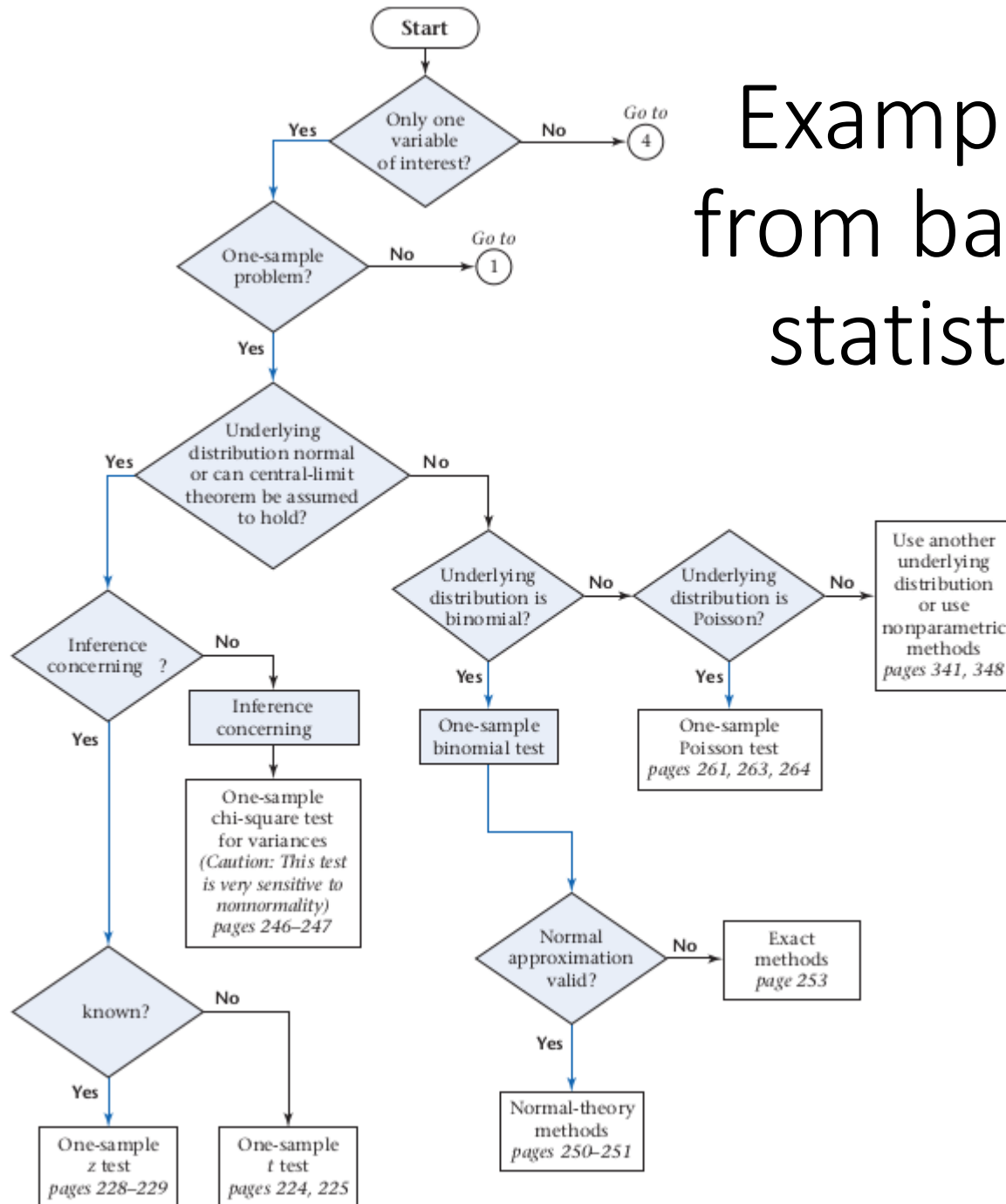
Three Variables

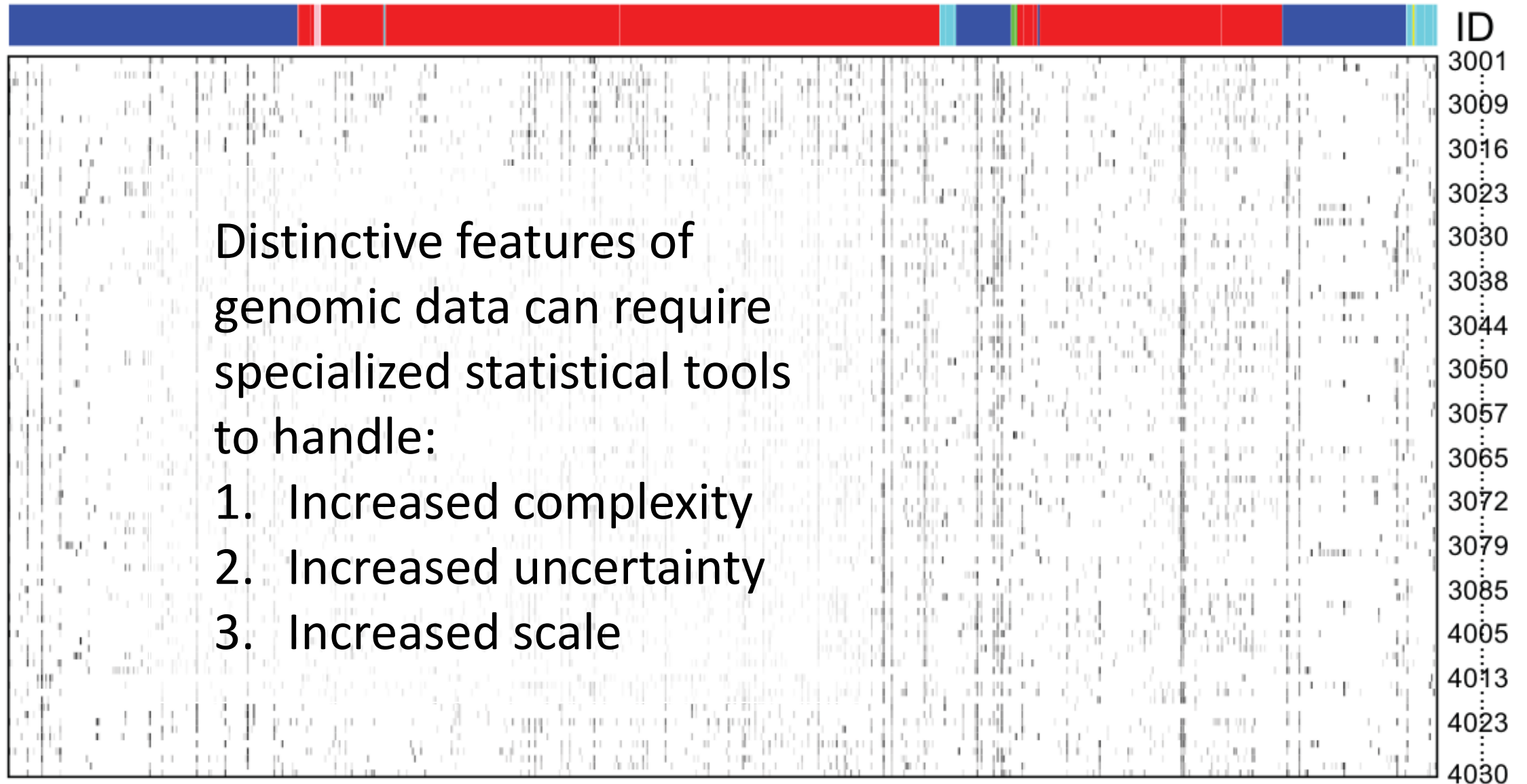
`seals$z <- with(seals, sqrt(delta_long^2 + delta_lat^2))`
`m <- ggplot(seals, aes(long, lat))`

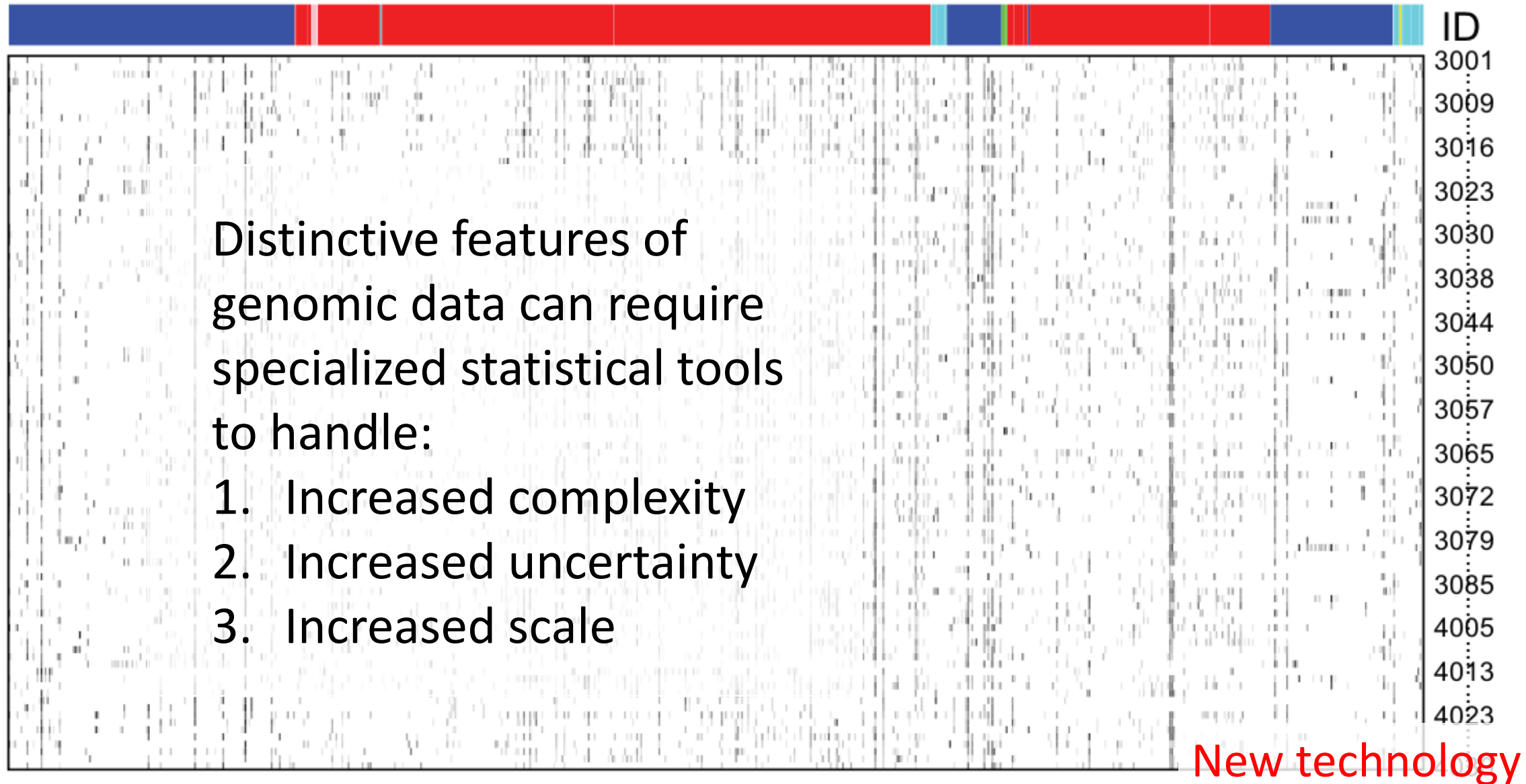
`m + geom_raster(aes(fill = z), hjust = 0.5,
 vjust = 0.5, interpolate = FALSE)`
x, y, alpha, fill (fast)

`m + geom_tile(aes(fill = z))`
x, y, alpha, color, fill, linetype, size (slow)

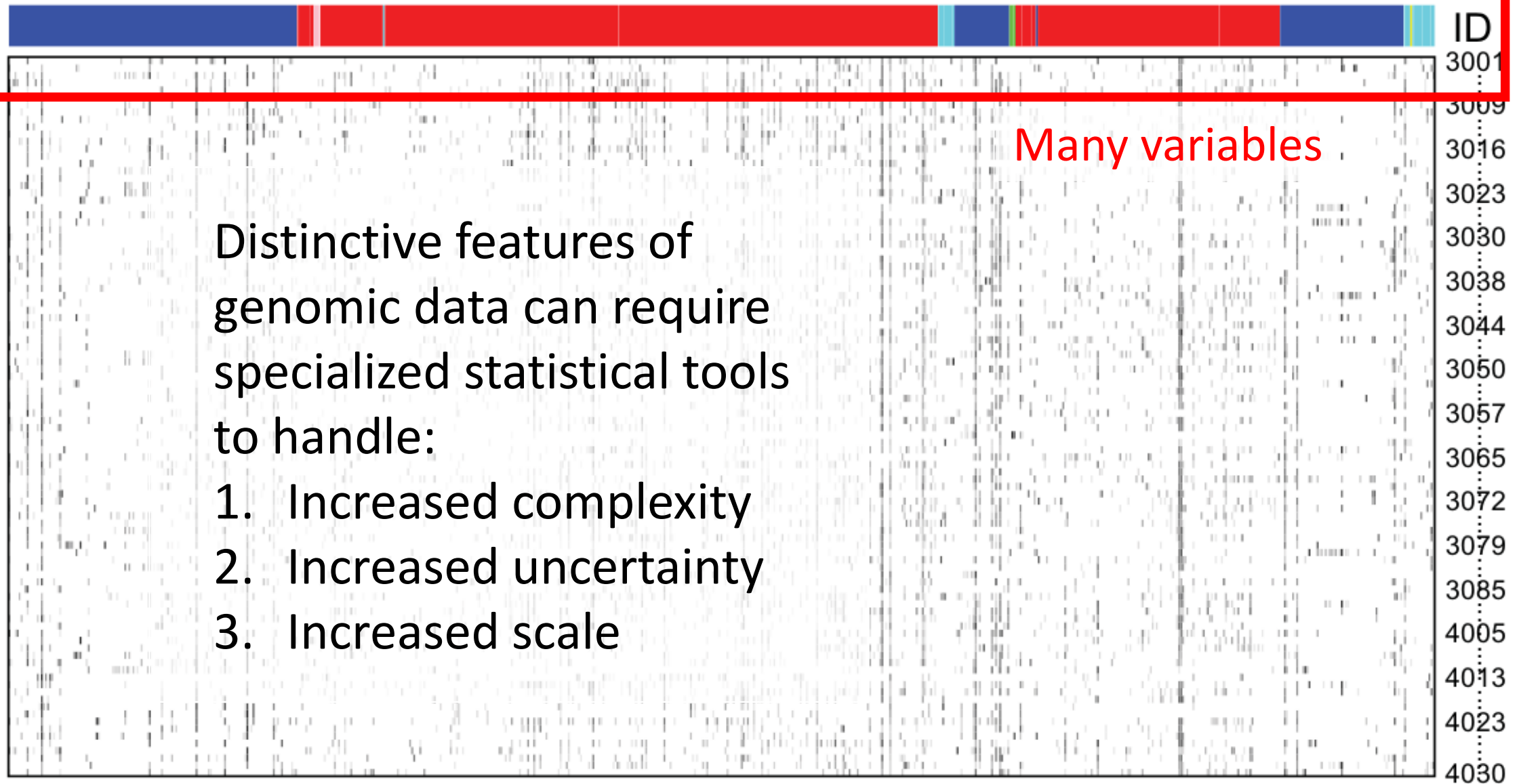
Examples from basic statistics



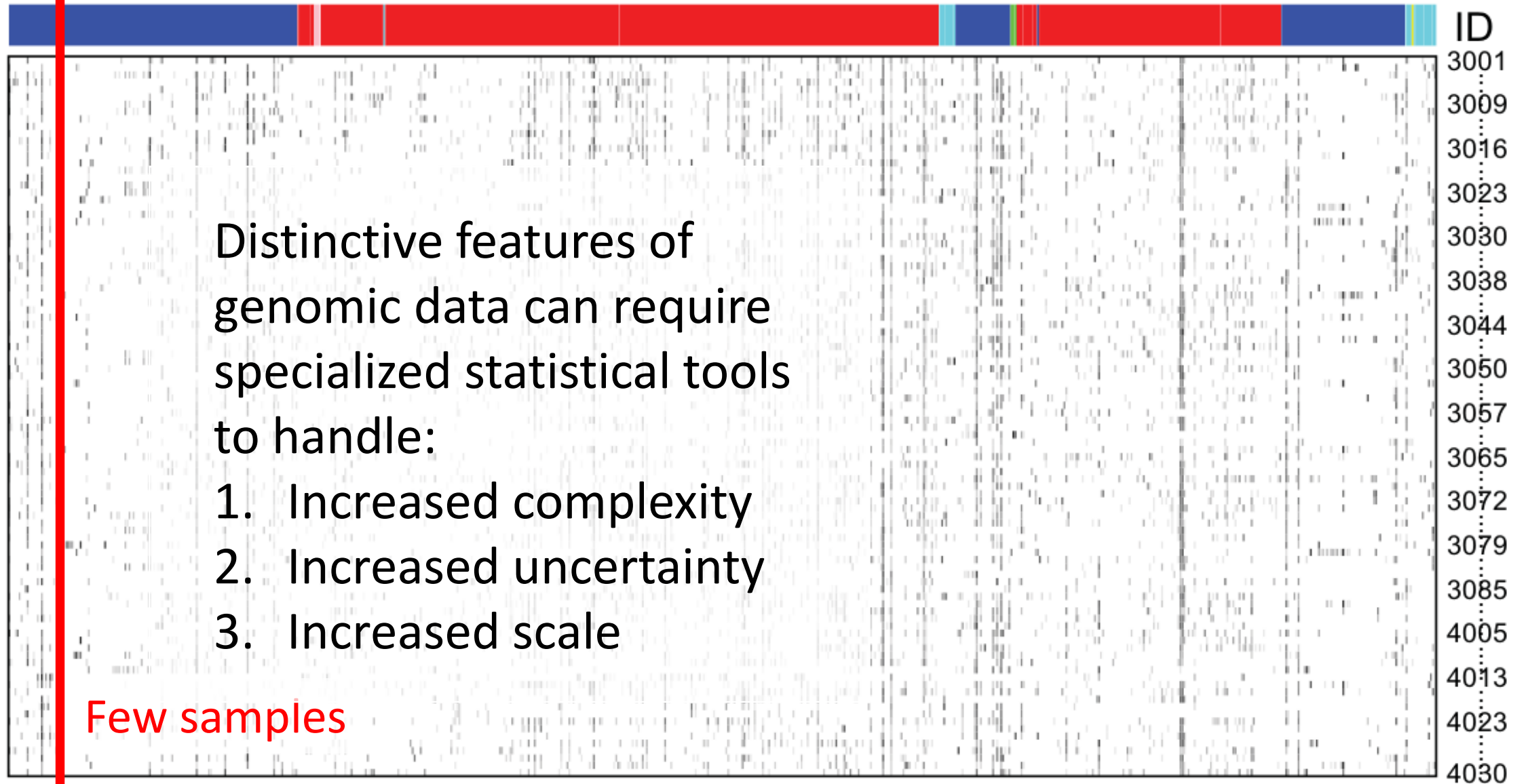




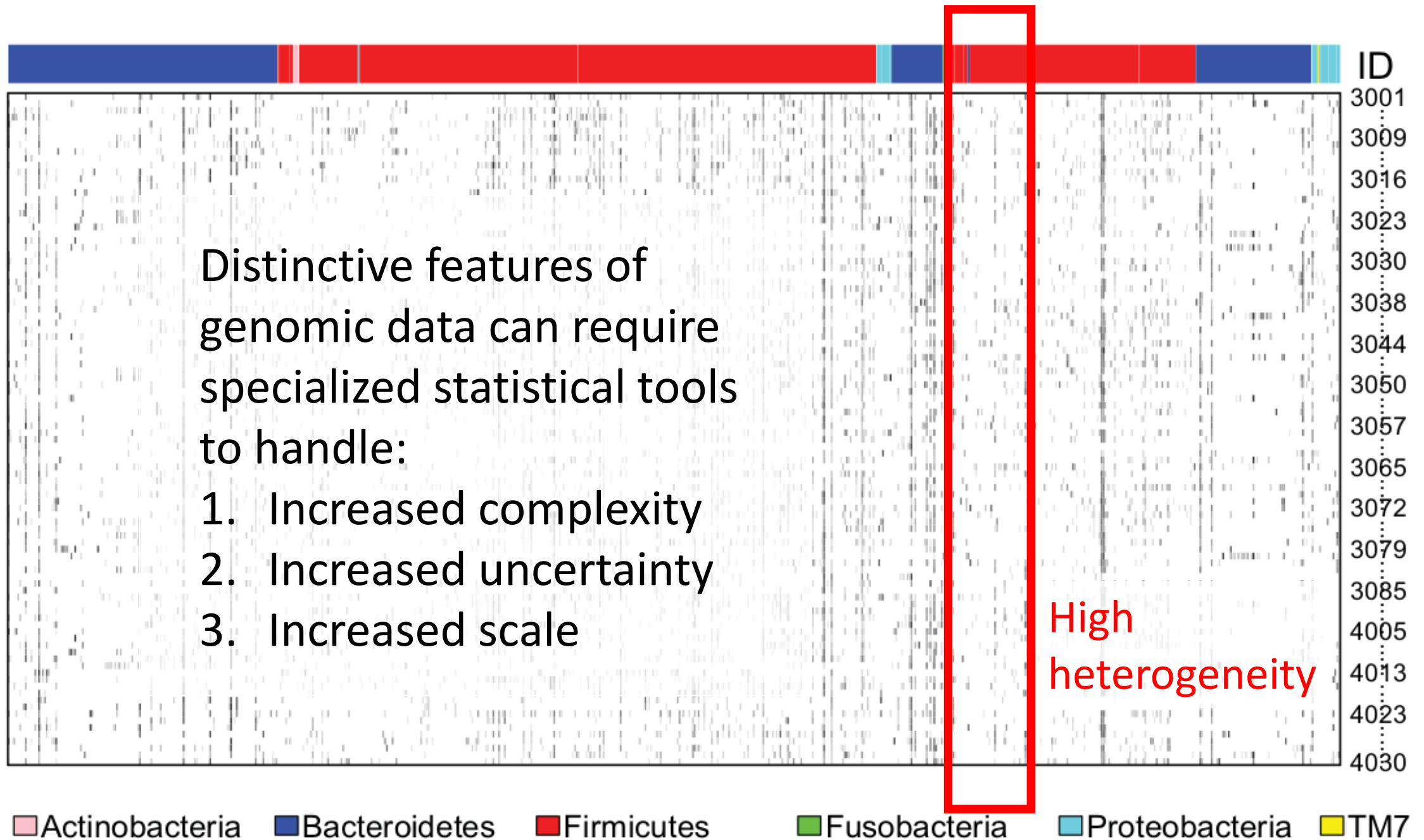
Actinobacteria Bacteroidetes Firmicutes Fusobacteria Proteobacteria TM7



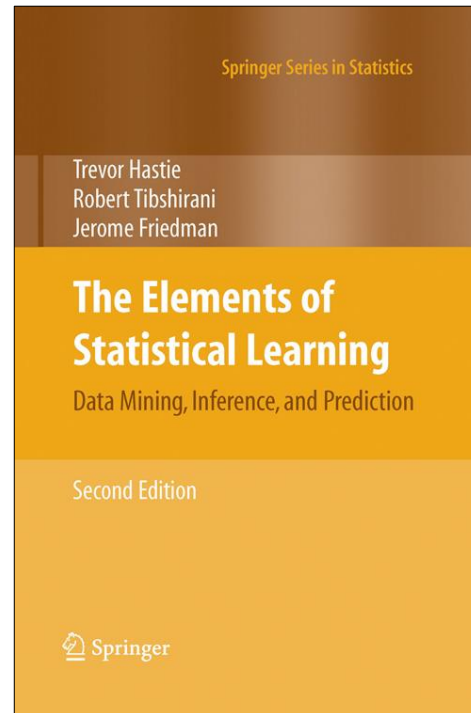
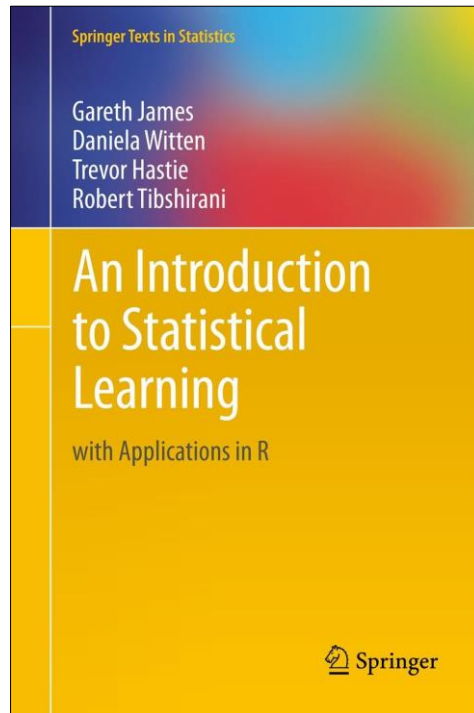
Actinobacteria Bacteroidetes Firmicutes Fusobacteria Proteobacteria TM7



Actinobacteria Bacteroidetes Firmicutes Fusobacteria Proteobacteria TM7



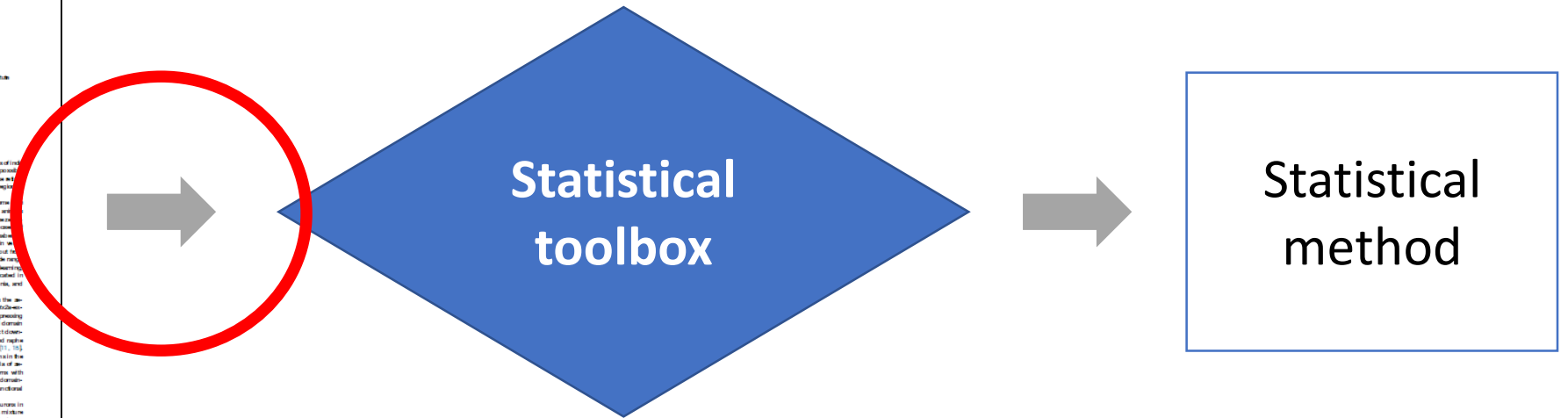
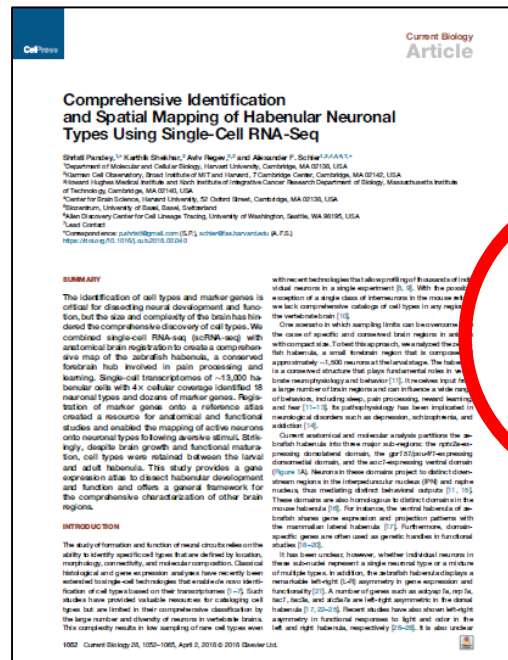
We will discuss some statistical tools that are frequently used in genomics



We will also discuss a general framework for organizing new tools.



Choosing the right tool for a given biological question requires creativity and experience



There are other frameworks that are organized by biological question rather than statistical tool

The image displays three browser windows side-by-side, showing course materials for STAT 555: Statistical Genomics.

Left Window: The address bar shows www.biostat.jhsph.edu/~iruczins/teach. The page lists a series of topics, each with a PDF icon:

- Introduction to statistical genomics
- Summarizing and presenting genomic data
- Statistical modeling I : means and two-group comparison
- Multiple hypothesis testing
- Differential expression
- Pathway and gene set analyses
- Experimental design
- Dimension reduction
- Batch effects
- Statistical modeling II : linear models in genomics
- Statistical modeling III : pre-processing genomic data

Middle Window: The address bar shows https://www.jmp.com/en_us/academic/jmpg-com. The page is titled "Step-by-Step Guides" and lists several guides:

- Step-by-Step Guide for Genetics QK Modeling #1: K
- Step-by-Step Guide for Genetics QK Modeling #2: F
- Step-by-Step Guide for Genetics QK Modeling #3: A
- Step-by-Step Guide for Genetics QK Modeling #4: U
- Step-by-Step Guide for Genetics QK Modeling #5: K
- Step-by-Step Guide for Expression #1: Importing and
- Step-by-Step Guide for Expression #2: Predictive M
- Model Comparison for Model Selection
- Step-by-Step Guide for Expression #3: Learning Cu
- Samples Needed
- Step-by-Step Guide for Expression #4: Subset Data
- Step-by-Step Guide for Expression #5: Final Model
- Step-by-Step Guide for Importing Expression Data i
- Step-by-Step Guide for Importing Genetics Data into
- Additional Step-by-Step Guides

Right Window: The address bar shows <https://onlinecourses.science.psu.edu/statprogram/stat555>. The page is titled "Course Topics" and lists the topics that will be covered in the course:

The topics that will be covered in this course will likely include:

1. Introduction to R and RStudio
2. Introduction to cell biology
3. Introduction to measurement technologies: microarrays, sequencing, SNPs and ChIP
4. Basic statistics
5. Gene Expression Microarrays: experimental designs, preprocessing and normalization, differential expression.
6. RNA-seq: experimental designs, preprocessing and normalization, differential expression, splice variants
7. SNPs
8. ChIPs
9. Replication and pooling
10. Gene Set enrichment analysis
11. Clustering samples and genes
12. Classifying samples using statistical machine learning

Example analysis using R

Use R packages to perform data analysis

CRAN - Package ranger

<https://cran.r-project.org/web/packages/ranger/index.html>

ranger: A Fast Implementation of Random Forests

A fast implementation of Random Forests, particularly suited for high dimensional data. Ensembles of classification, regression, survival and probability prediction trees are supported. Data from genome-wide association studies can be analyzed efficiently. In addition to data frames, datasets of class 'gwa.data' (R package 'GenABEL') and 'dgCMatrx' (R package 'Matrix') can be directly analyzed.

Version: 0.11.2
Depends: R (≥ 3.1)
Imports: Rcpp (≥ 0.11.2), Matrix
LinkingTo: Rcpp, RcppEigen
Suggests: survival, testthat
Published: 2019-03-07
Author: Marvin N. Wright [aut, cre], Stefan Wager [ctb], Philipp Probst [ctb]
Maintainer: Marvin N. Wright <cran at wrig.c...>
BugReports: <https://github.com/imbs-hl/ranger>
License: GPL-3
URL: <https://github.com/imbs-hl/ranger>

CRAN - Package caret

<https://cran.r-project.org/web/packages/caret/>

caret: Classification and Regression Training

Misc functions for training and plotting classification and regression models.

Version: 6.0-84
Depends: R (≥ 3.2.0), lattice (≥ 0.20), ggplot2
Imports: foreach, methods, plyr, ModelMetrics (≥ 1.1.0), nlme, reshape2, stats, stats4, utils, grDevices, recipes (≥ 0.1.4), withr (≥ 2.0.0)
Suggests: BradleyTerry2, e1071, earth (≥ 2.2-3), fastICA, gam (≥ 1.15), ipred, kernlab, knitr, klaR, MASS, ellipse, mda, mgcv, mlbench, MLmetrics, nnet, party (≥ 0.9-99992), pls, pROC, proxy, randomForest, RANN, spls, subselect, pamr, superpc, Cubist, testthat (≥ 0.9.1), rpart, dplyr
Published: 2019-04-27
Author: Max Kuhn. Contributions from Jed Wing, Steve Weston, Andre Williams, Chris Keefe, Allan Engelhardt, Tony Cooper, Zachary Mayer, Brenton Kenkel, the R Core Team, Michael Benesty, Reynald Lescarbeau, Andrew Ziem, Luca Scrucca, Yuan Tang, Can Candan, and Tyler Hunt.
Maintainer: Max Kuhn <mxkuhn at gmail.com>

CRAN - Package glmnet

<https://cran.r-project.org/web/packages/glmnet/index.html>

glmnet: Lasso and Elastic-Net Regularized Generalized Linear Models

Extremely efficient procedures for fitting the entire lasso or elastic-net regularization path for linear regression, logistic and multinomial regression models, Poisson regression and the Cox model. Two recent additions are the multiple-response Gaussian, and the grouped multinomial regression. The algorithm uses cyclical coordinate descent in a path-wise fashion, as described in the paper linked to via the URL below.

Version: 2.0-18
Depends: Matrix (≥ 1.0-6), utils, foreach
Imports: methods
Suggests: survival, knitr, lars
Published: 2019-05-20
Author: Jerome Friedman [aut, cre], Trevor Hastie [aut, cre], Balasubramanian Narasimha Murthy [ctb]
Maintainer: Trevor Hastie <trevor.hastie at stanford.edu>
License: GPL-2
URL: <http://www.jstatsoft.org/v33/i01/>
NeedsCompilation: yes

CRAN - Package Seurat

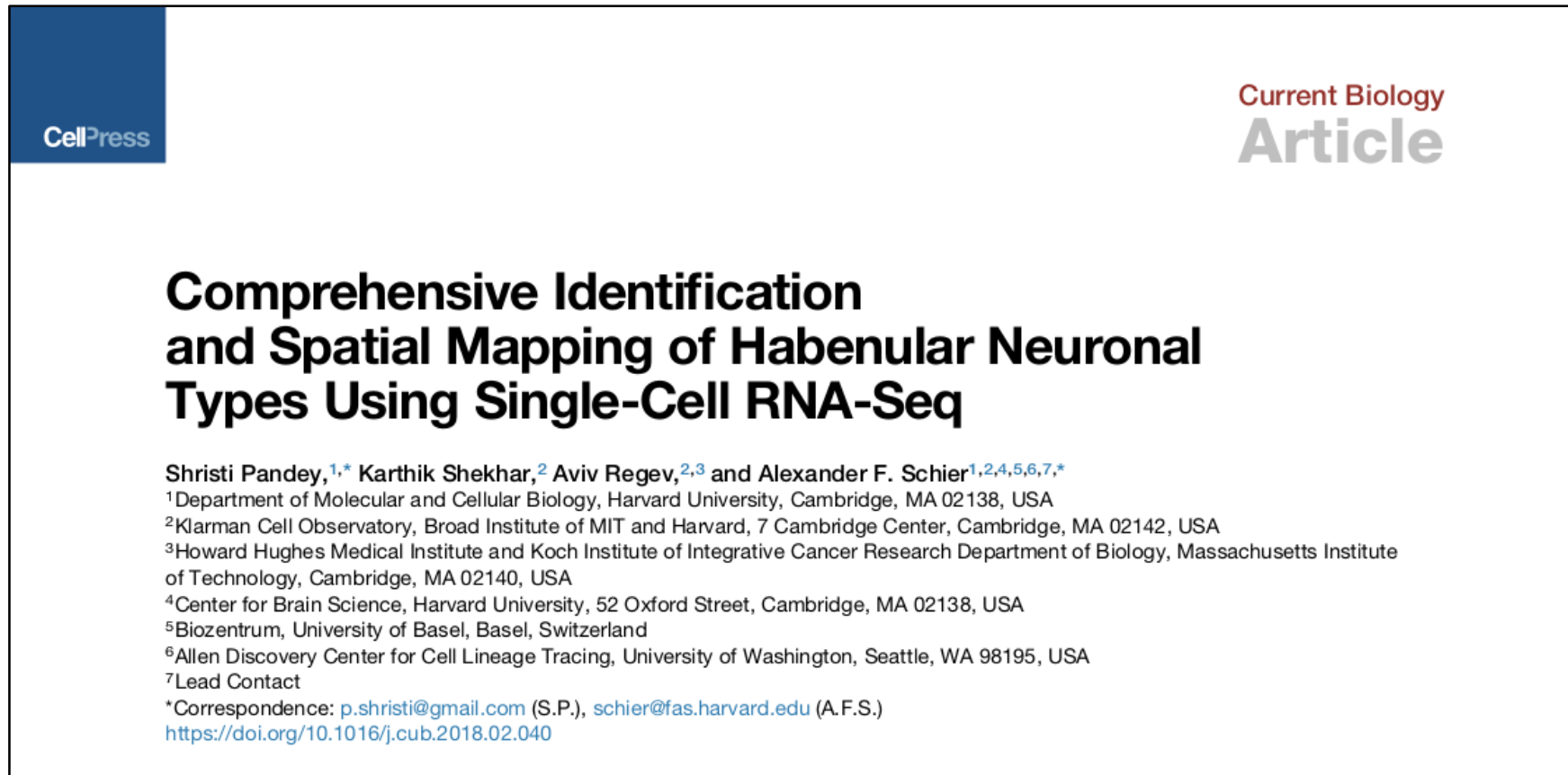
<https://cran.r-project.org/web/packages/Seurat/index.html>

Seurat: Tools for Single Cell Genomics

A toolkit for quality control, analysis, and exploration of single cell RNA sequencing data. 'Seurat' aims to enable users to identify and interpret sources of heterogeneity from single cell transcriptomic measurements, and to integrate diverse types of single cell data. See Satija R, Farrell J, Gennert D, et al (2015) <[doi:10.1038/nbt.3192](https://doi.org/10.1038/nbt.3192)>, Macosko E, Basu A, Satija R, et al (2015) <[doi:10.1016/j.cell.2015.05.002](https://doi.org/10.1016/j.cell.2015.05.002)>, and Butler A and Satija R (2017) <[doi:10.1101/164889](https://doi.org/10.1101/164889)> for more details.

Version: 3.0.1
Depends: R (≥ 3.4.0), methods
Imports: ape, cluster, cowplot, fitdistrplus, future, future.apply, ggplot2 (≥ 3.0.0), ggrepel, ggridges, graphics, grDevices, grid, ica, igraph, iriba, KernSmooth, lme4, MASS, Matrix (≥ 1.2.14), metap, pbapply, plotly, png, RANN, RColorBrewer, Rcpp, reticulate, rlang, ROCR, rsvd, Rtsne, scales, sctransform (≥ 0.2.0), SDMTTools, stats, tools, tsnr, utils
LinkingTo: Rcpp (≥ 0.11.0), RcppEigen, RcppProgress
Suggests: loomR, testthat, hdf5r, S4Vectors, SummarizedExperiment, SingleCellExperiment, Matrix, Rcpp, RcppEigen, RcppProgress

To illustrate these tools, we will analyze single-cell RNA-seq data in R



Anatomy of a basic R command

```
pandey =  
read.table("GSM2818521_larva_counts_matrix.txt")
```

- Case-sensitive
- **Function()**: performs pre-programmed calculations given **inputs and options**
- **Variable**: stores values and outputs of function, name cannot contain whitespace and cannot start with a special character

Basic preprocessing of single-cell RNA-seq data using Seurat

```
library(Seurat)

s_obj = CreateSeuratObject(counts = pandey,
min.cells = 3, min.features = 200)

s_obj = NormalizeData(s_obj)

s_obj = FindVariableFeatures(s_obj)

s_obj = ScaleData(s_obj)
```

Statistical methods for genomics

Where statistics appears in a standard genomic analysis workflow

1. Experimental design ← Not covered today
 2. Quality control
 3. Preprocessing
 4. Normalization and batch correction
 5. Analysis ← The focus of today's discussion
 6. Biological interpretation
- Uses statistics but is highly dependent on technology

Classifying statistical tools

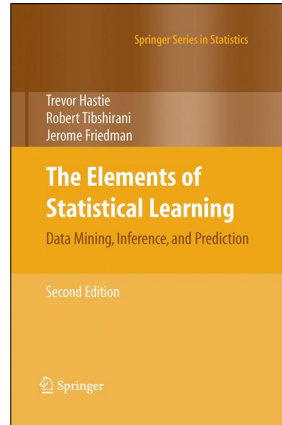
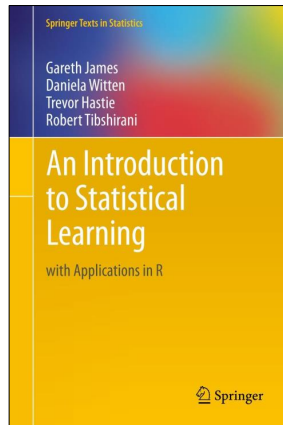
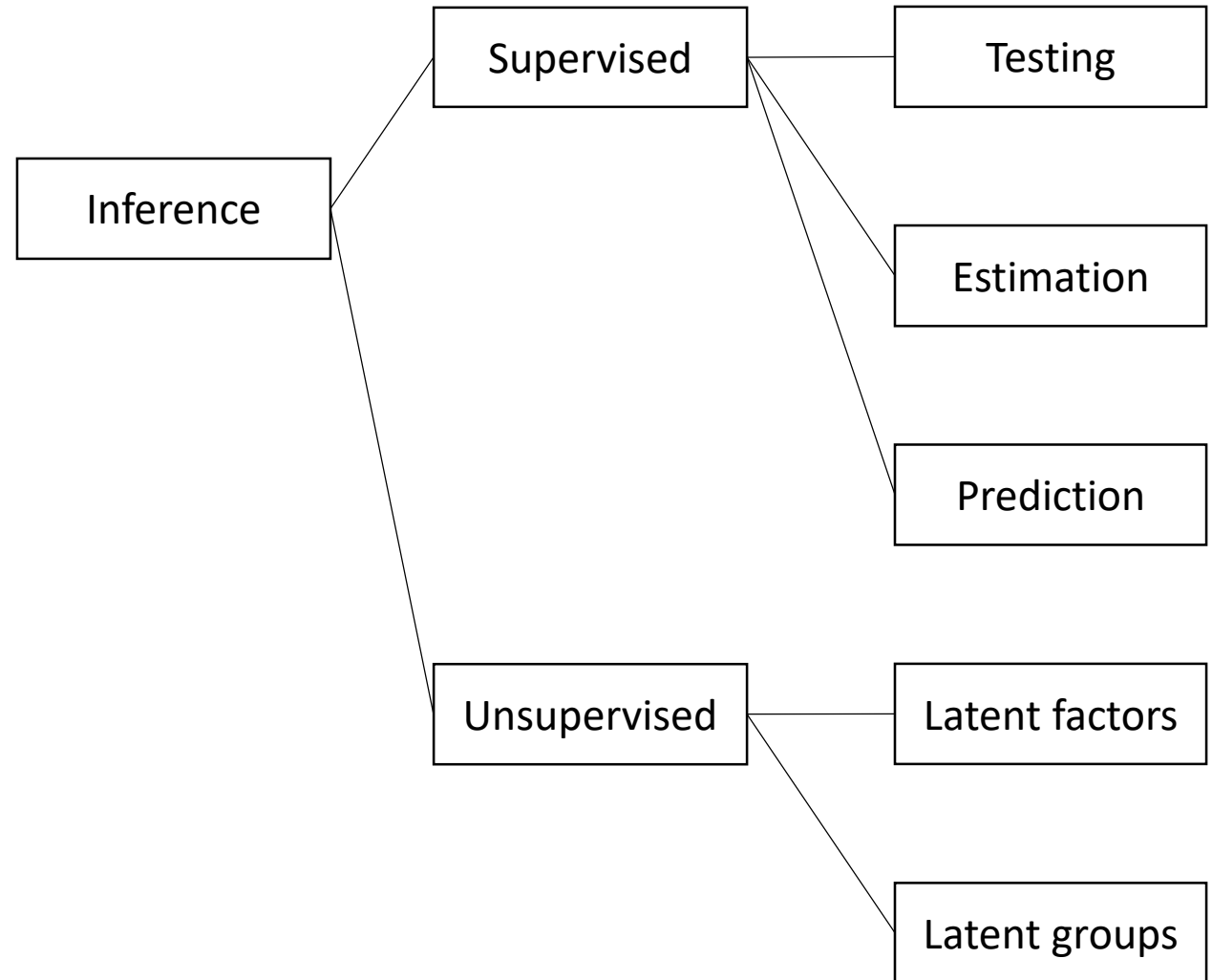
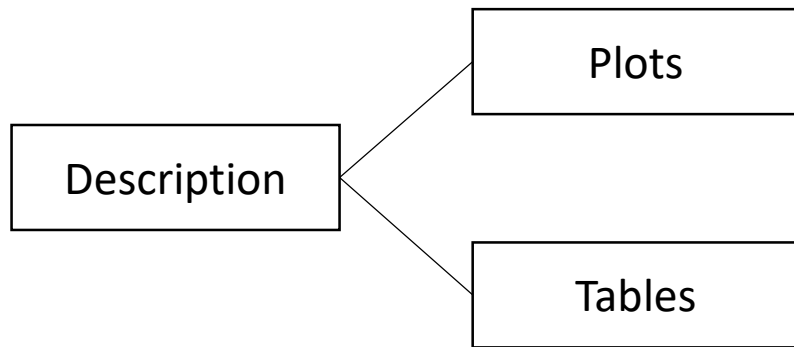
Data structure

Statistical task

	No dependent variables	Continuous outcome	Censored outcomes	Etc.
Visualize				
Identify latent factors				
Cluster observations				
Select features				
Etc.				

**APPROPRIATE
STATISTICAL
METHODS**

Classifying statistical tasks



Classifying data structures

- Can vary widely, and classification is difficult
- Important factors in genomics:
 1. Data type
 2. Number of samples relative to number of variables

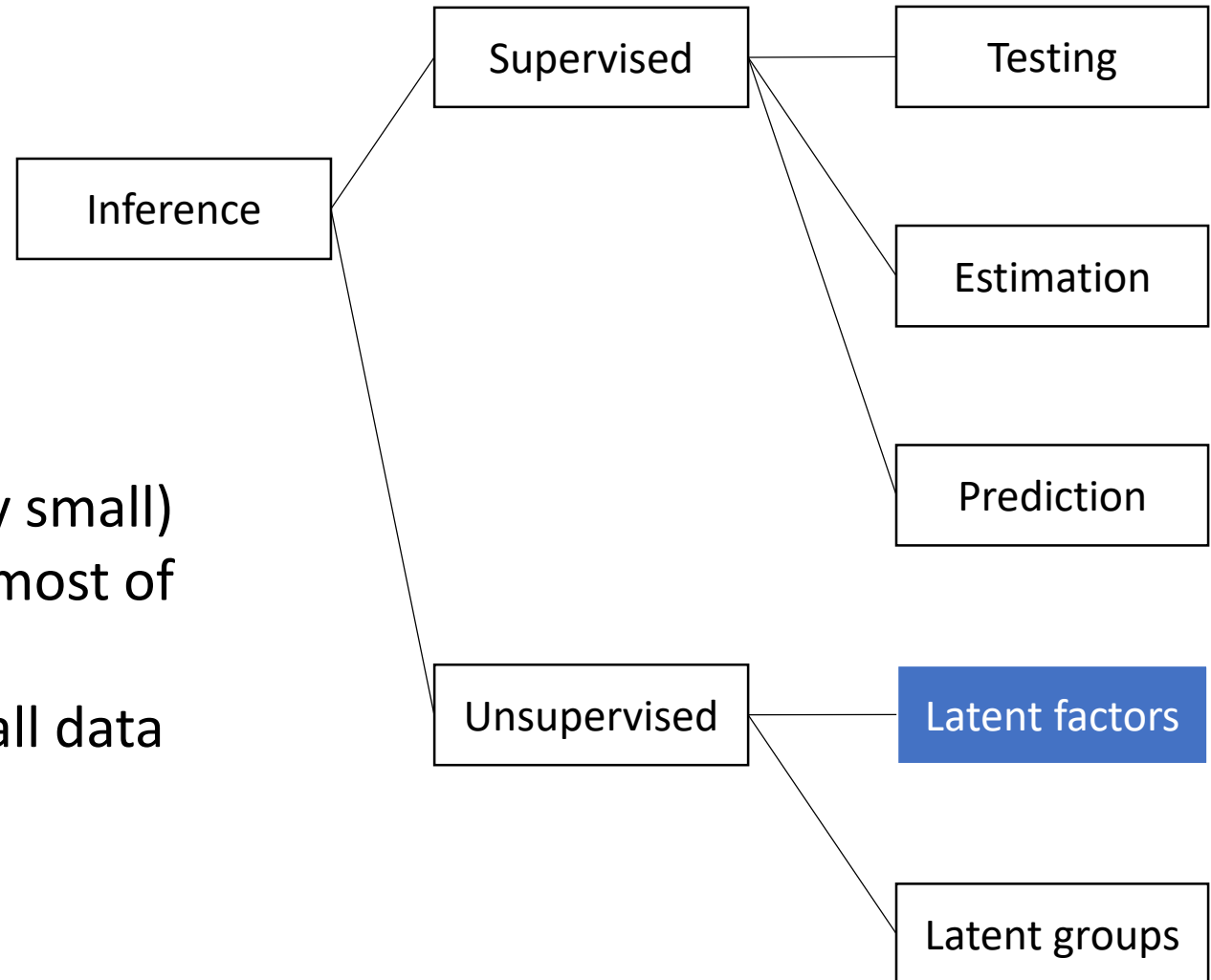
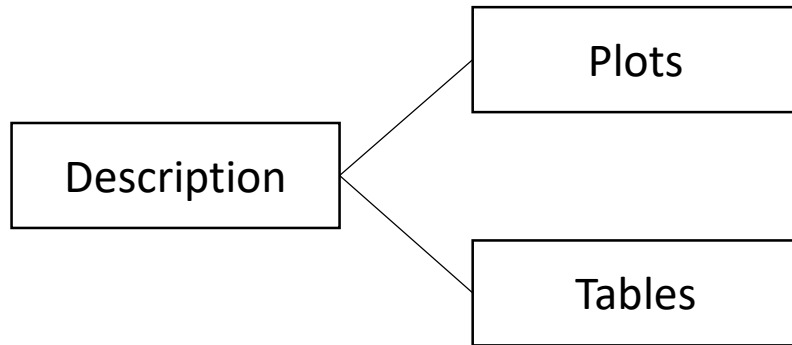
PCA

```
> dim(pandey)
[1] 24105  4365
> pandey[1:3, 1:2]
      larvalR2_AAACCTGAGACAGAGA.1 larvalR2_AAACCTGAGACTTTCG.1
SYN3                             0                             0
PTPRO                             1                             0
EPS8                             0                             0
```

Research question:

Can the gene expression information be summarized in fewer features?

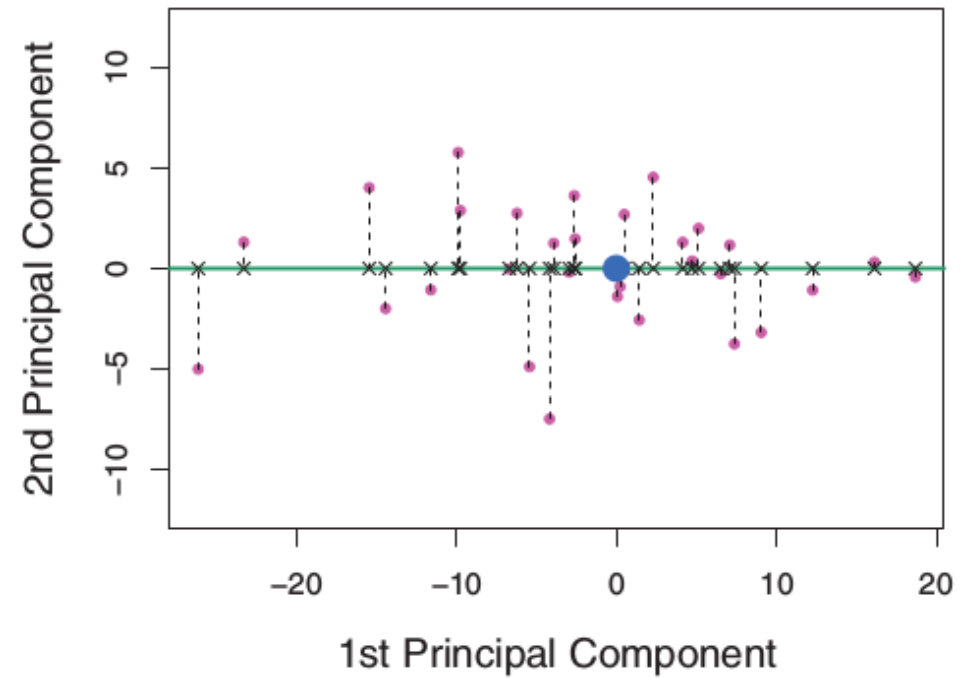
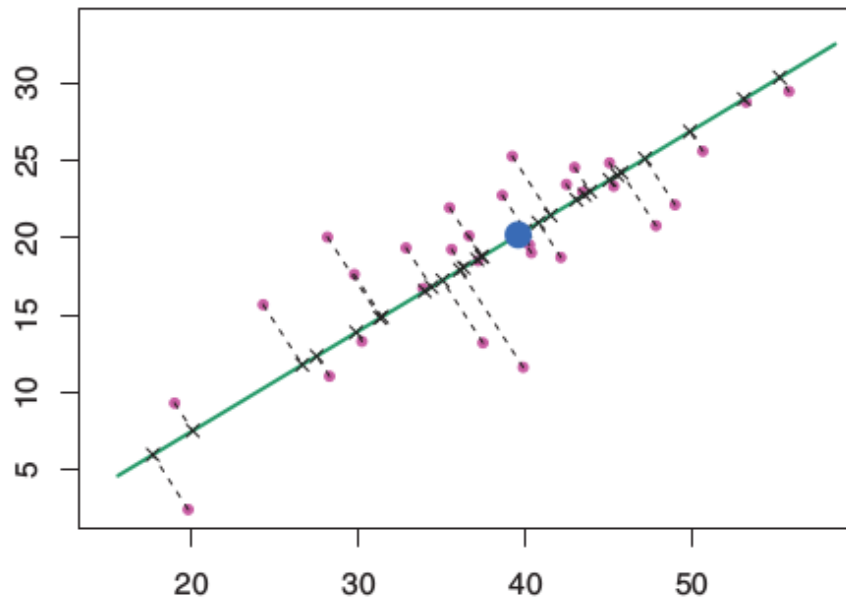
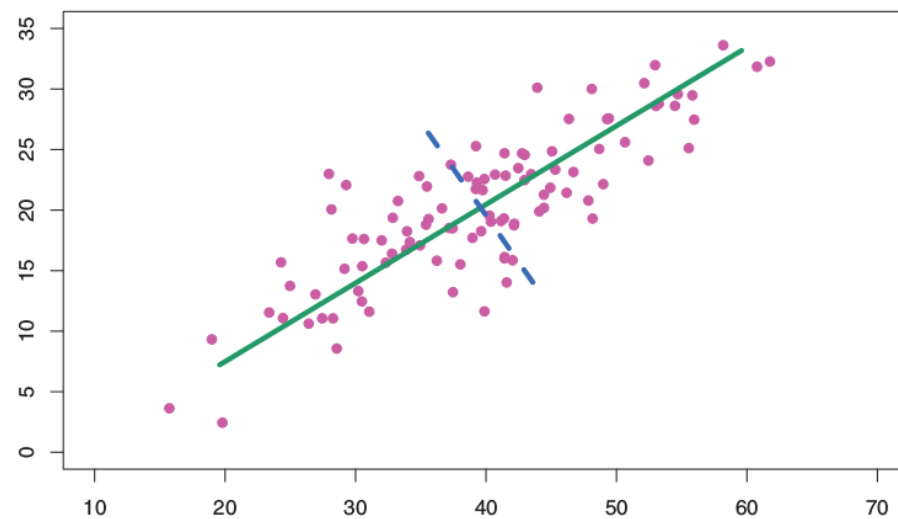
PCA



Statistical task: calculate a (usually small) set of latent factors that captures most of the information in the dataset

Data structure: can be applied to all data structures

PCA



PCA using Seurat

```
s_obj = RunPCA(s_obj)
```


Graph clustering




Research question:

How many cell types exist in the larval zebrafish habenula?

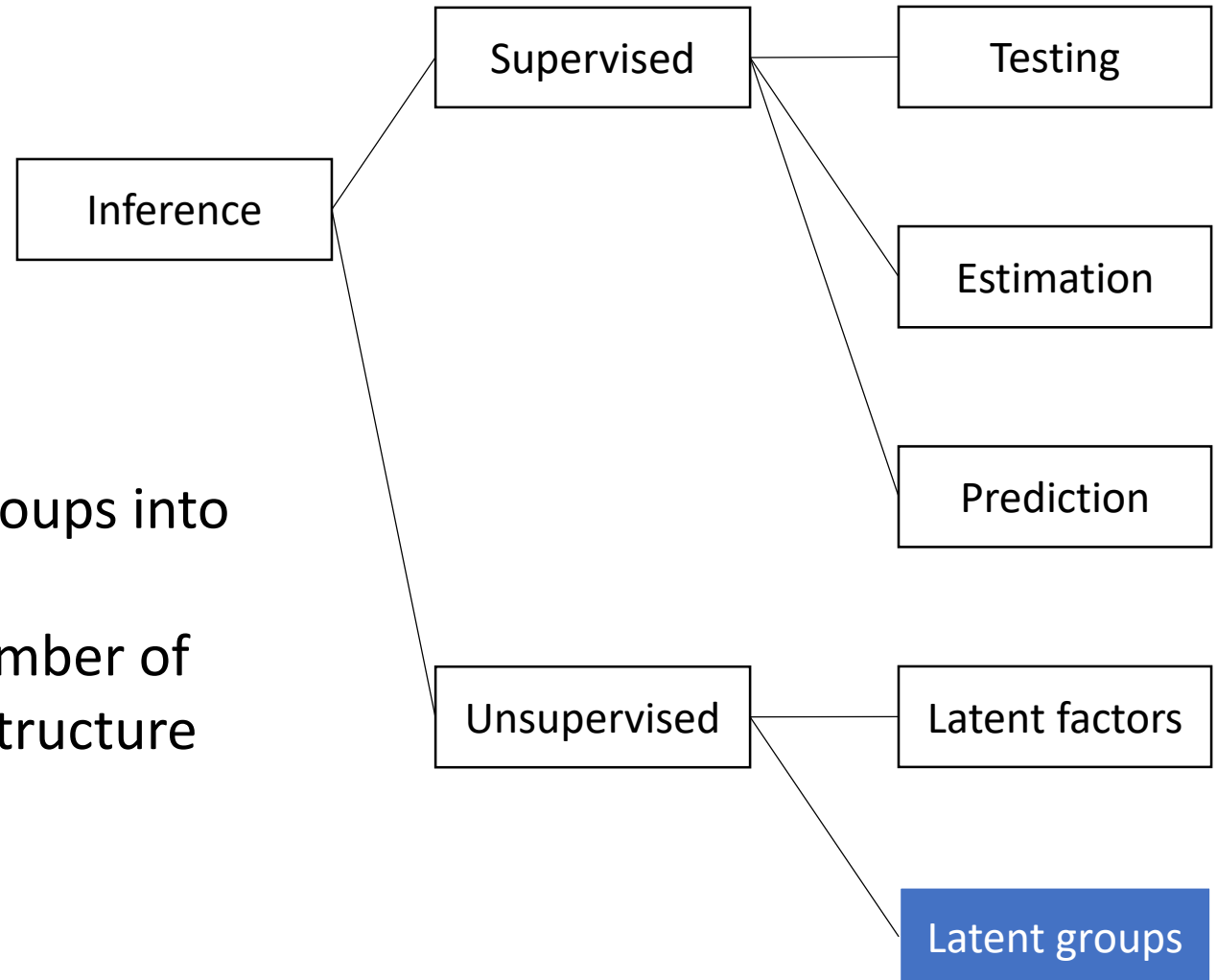
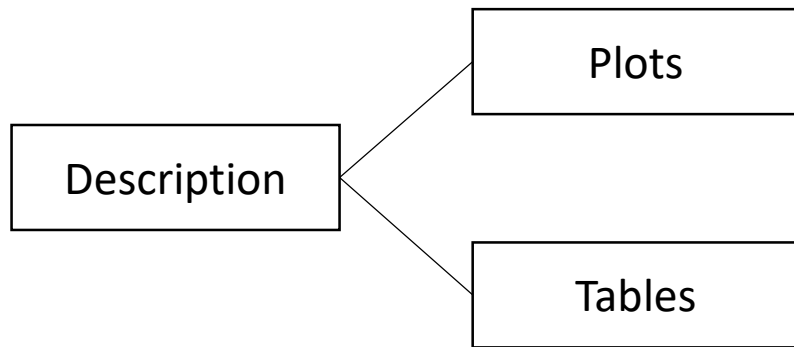
Cell Systems
Voices

CellPress

What Is Your Conceptual Definition of “Cell Type” in the Context of a Mature Organism?

What Is an Adult Cell Type, Really?	Defining Cell Type Space	Cellular Demographies, Recorded
		
<p>Hans Clevers Hubrecht Institute</p> <p>The human body is home to hundreds of cell types. Some are rather unobtrusive; others,</p>	<p>Susanne Rafelski Allen Institute for Cell Science</p> <p>Canonical cell types, e.g., muscle and nerve, were originally defined by the functions of</p>	<p>Michael Elowitz Caltech</p> <p>It seems to me that we are at the beginning of a paradigm shift on the issue of cell type.</p>

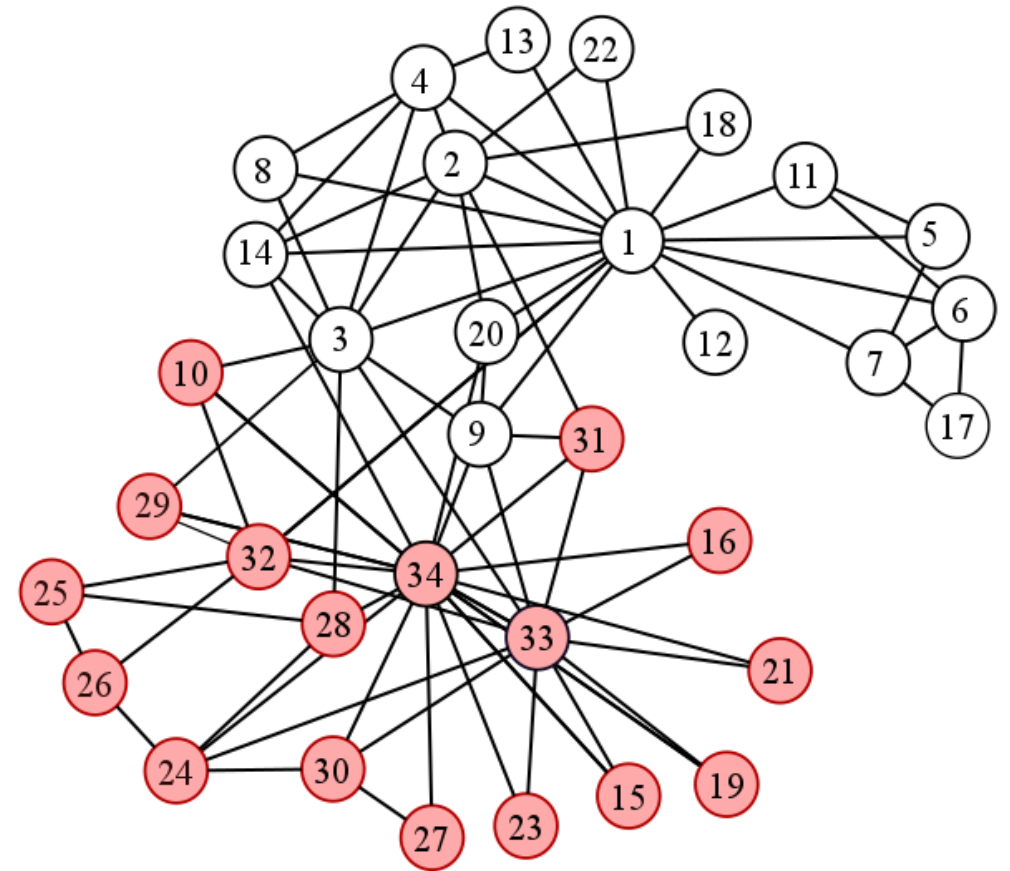
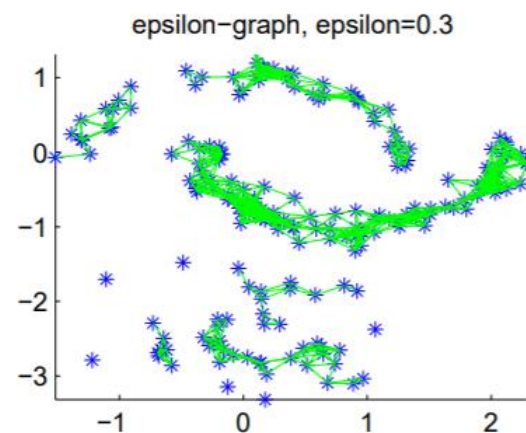
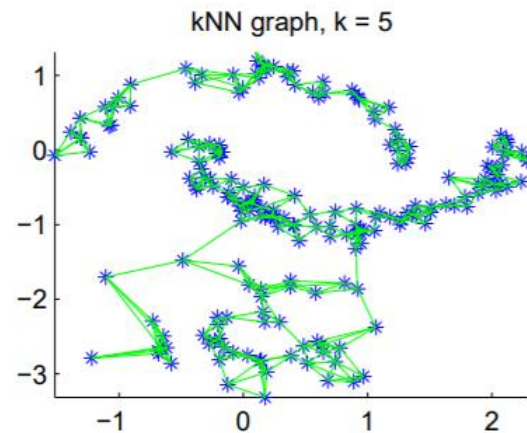
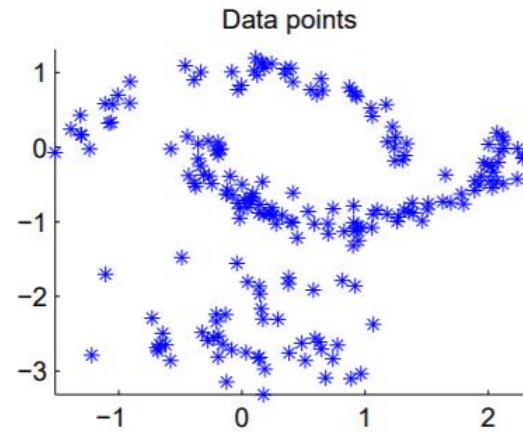
Graph clustering



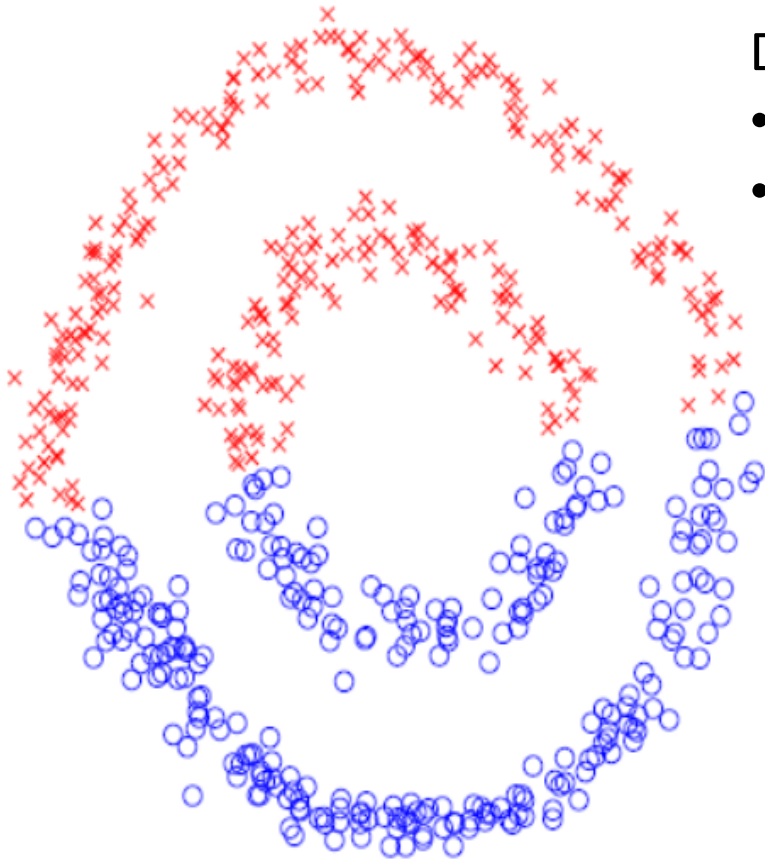
Statistical task: construct latent groups into which the observations fall

Data structure: relatively small number of features and complicated cluster structure

Graph clustering



Comparison to other clustering methods



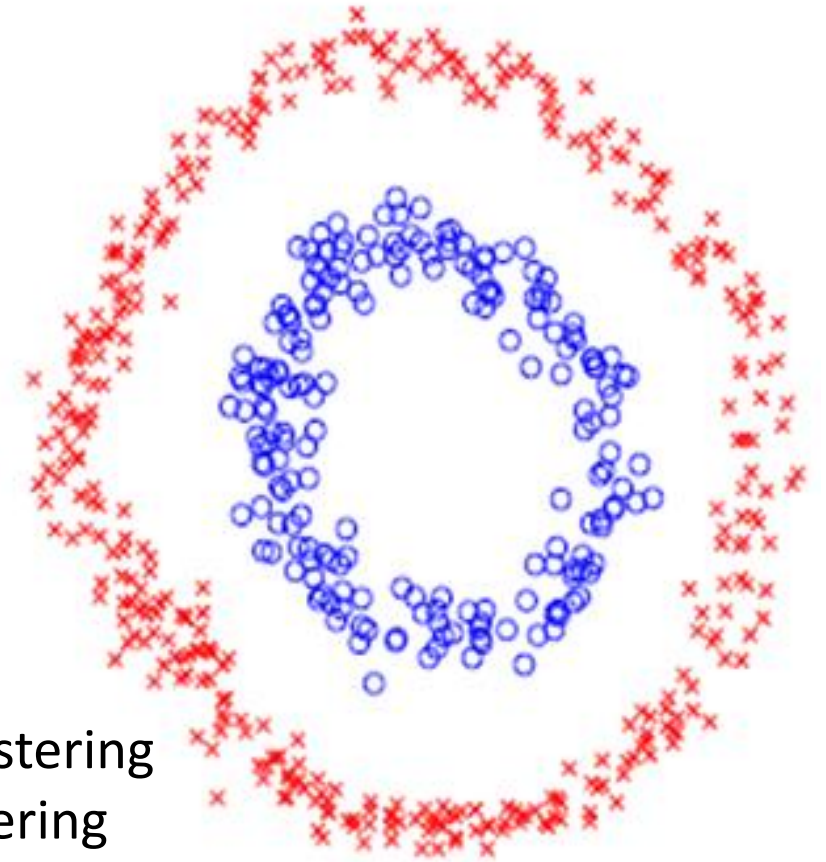
Distance

- Hierarchical clustering
- K-means clustering

vs.

Relationship

- Spectral clustering
- Graph clustering



Graph clustering using Seurat

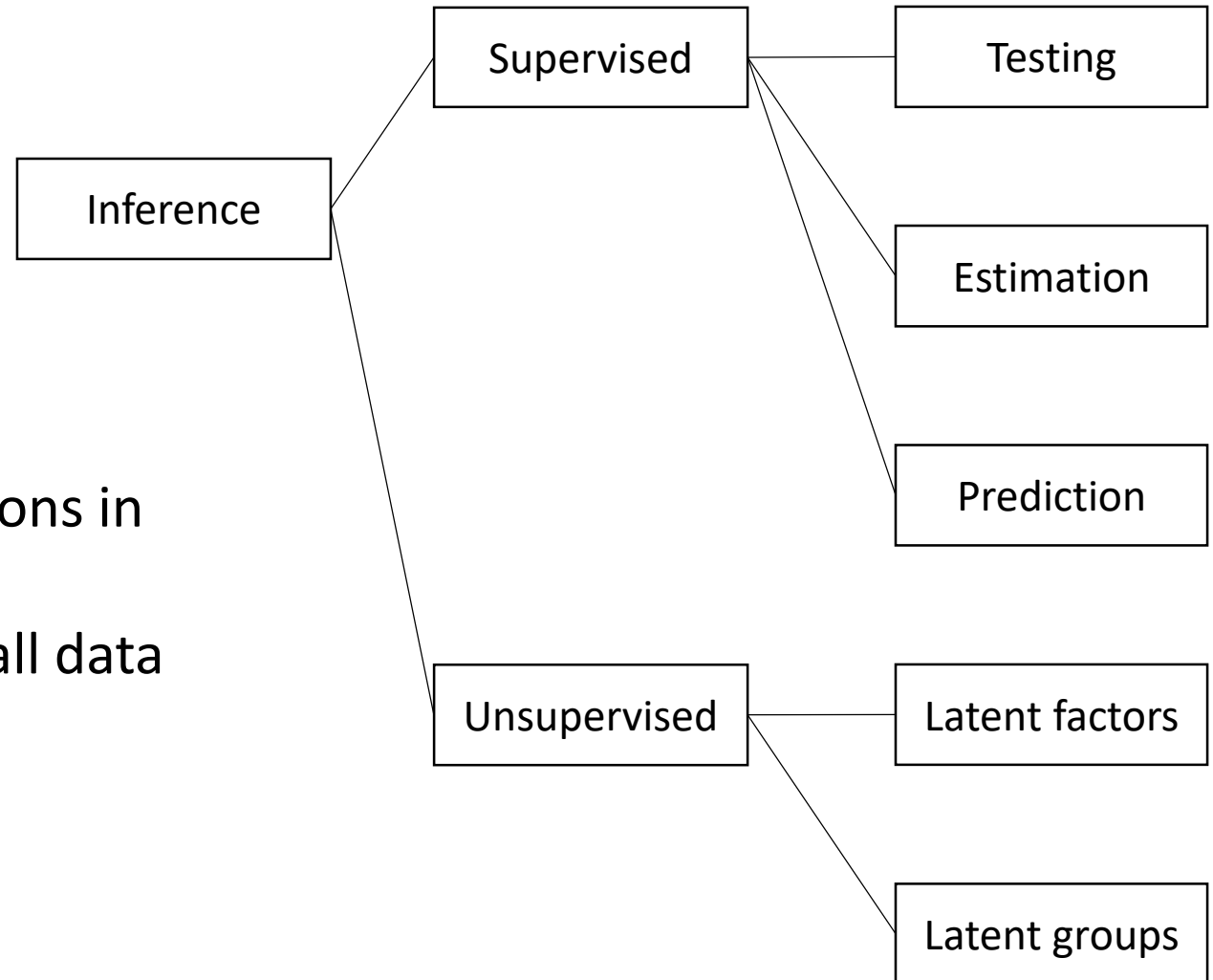
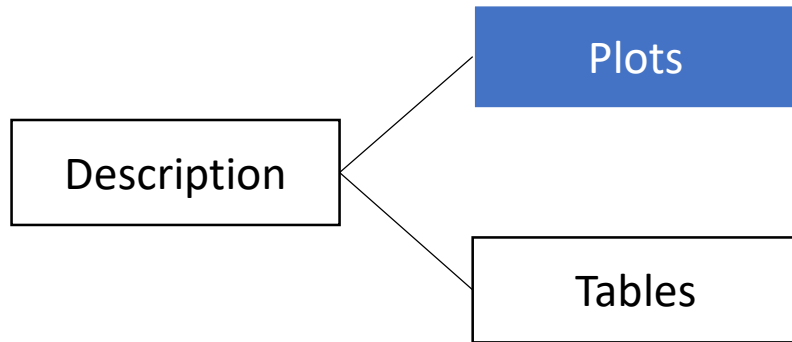
```
s_obj = FindNeighbors(s_obj, dims = 1:10)  
s_obj = FindClusters(s_obj, dims = 1:10,  
resolution = 0.1)
```

t-SNE plot

Research question:

How to visualize the different cell types?

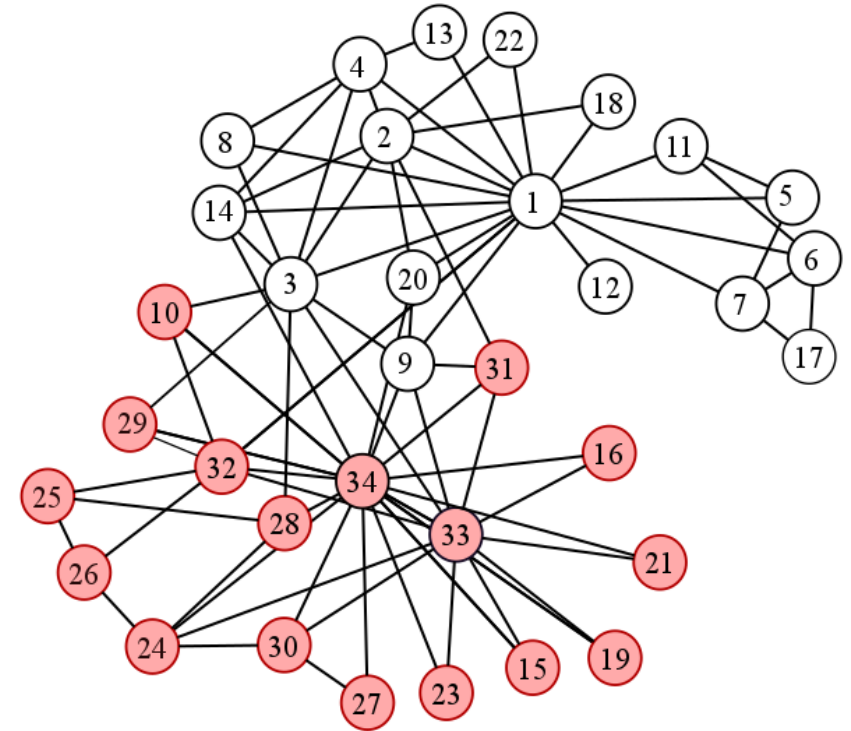
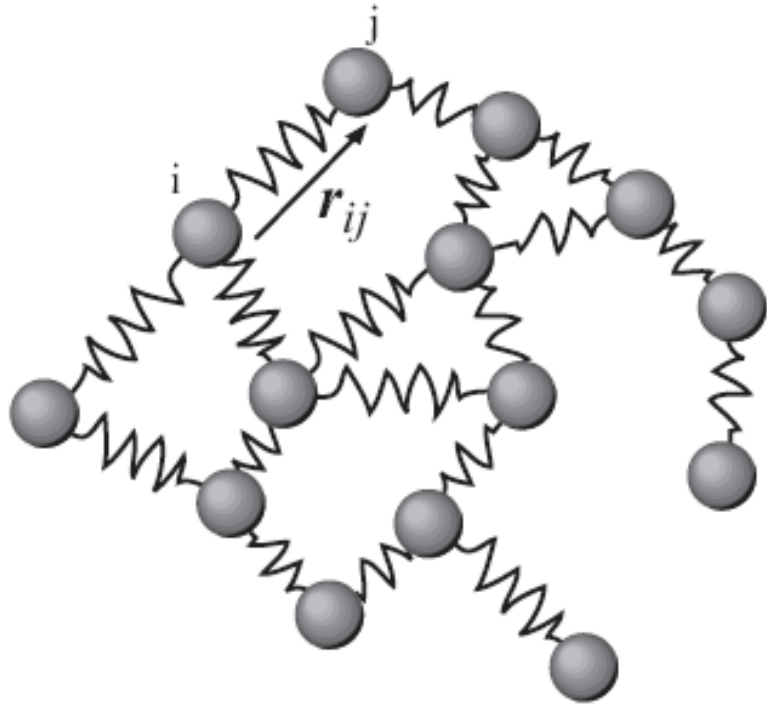
t-SNE plot



Statistical task: visualize observations in low dimensions

Data structure: can be applied to all data structures

t-SNE plot



t-SNE plot

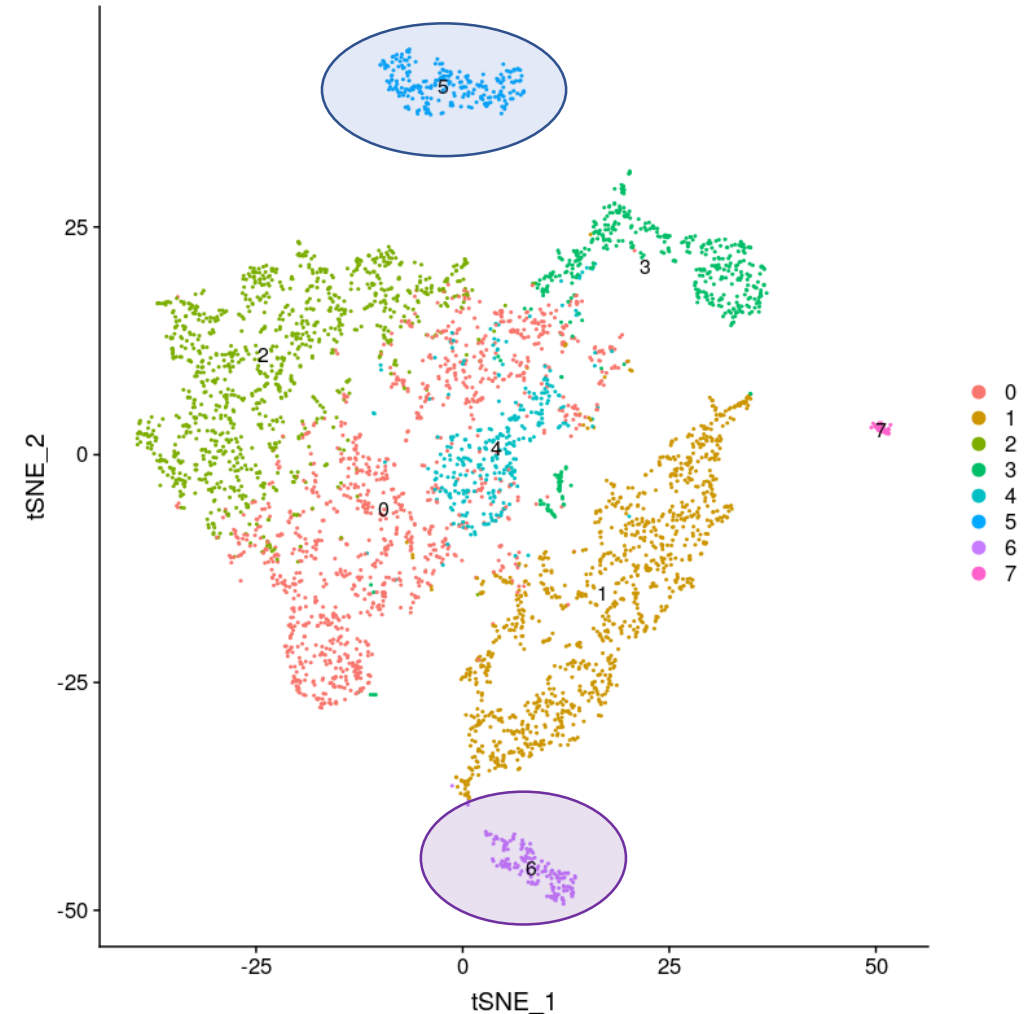
```
s_obj = RunTSNE(s_obj)
```

```
DimPlot(s_obj, reduction = "tsne", label = TRUE)
```

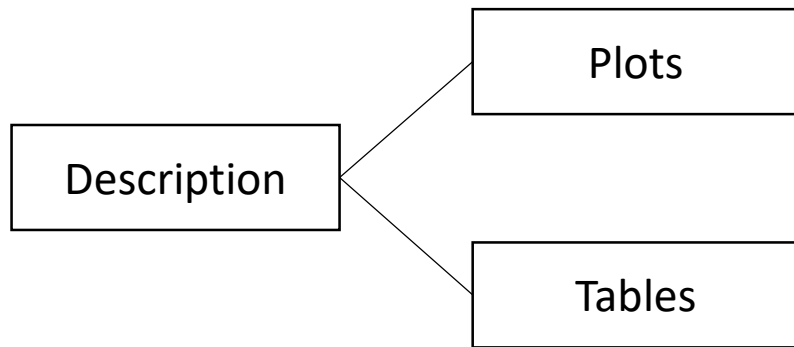
Wilcoxon test

Research question:

Does the expression of the gene PDYN differ between clusters 5 and 6?

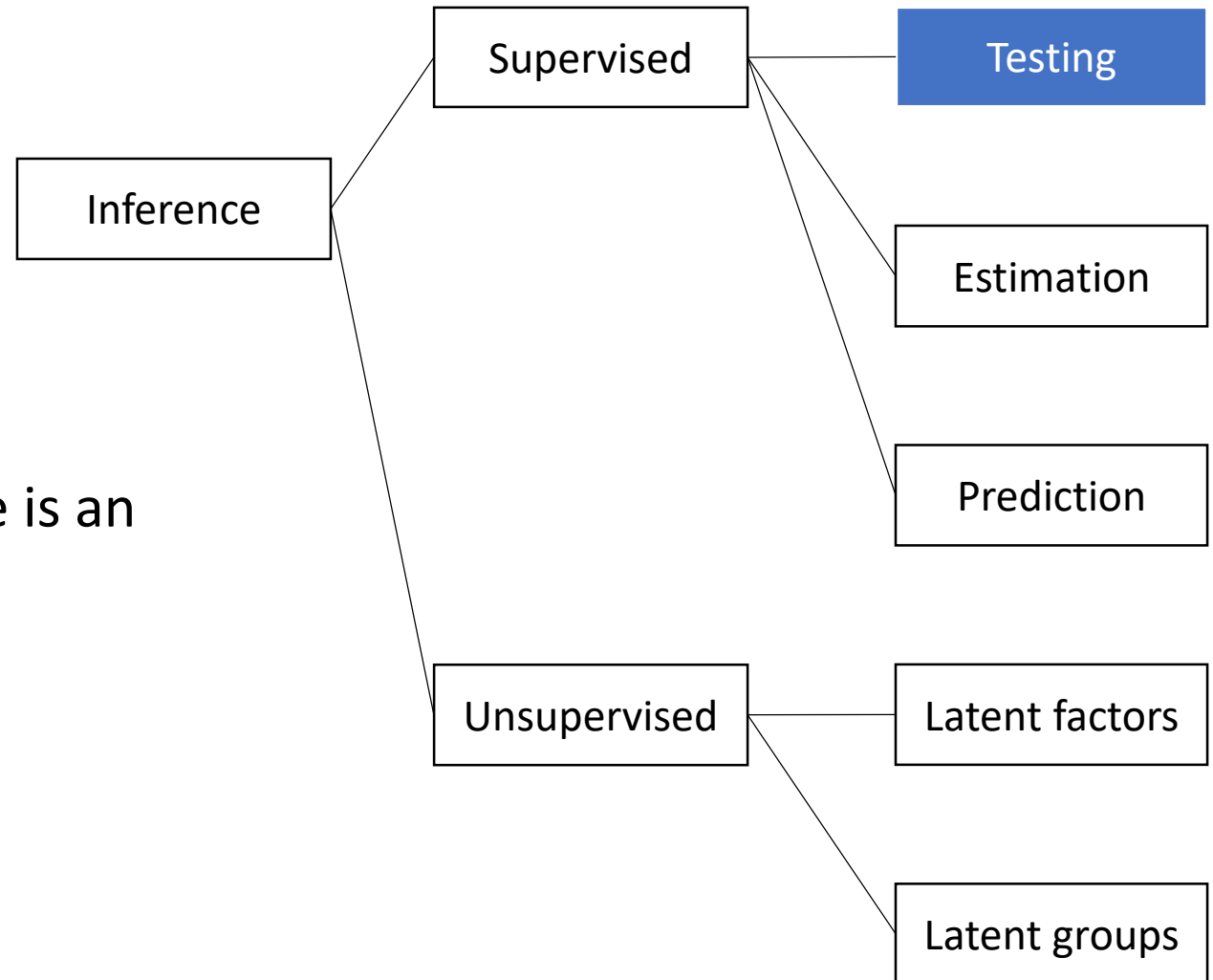


Wilcoxon test



Statistical task: test whether there is an association between two variables

Data structure: one variable is dichotomous



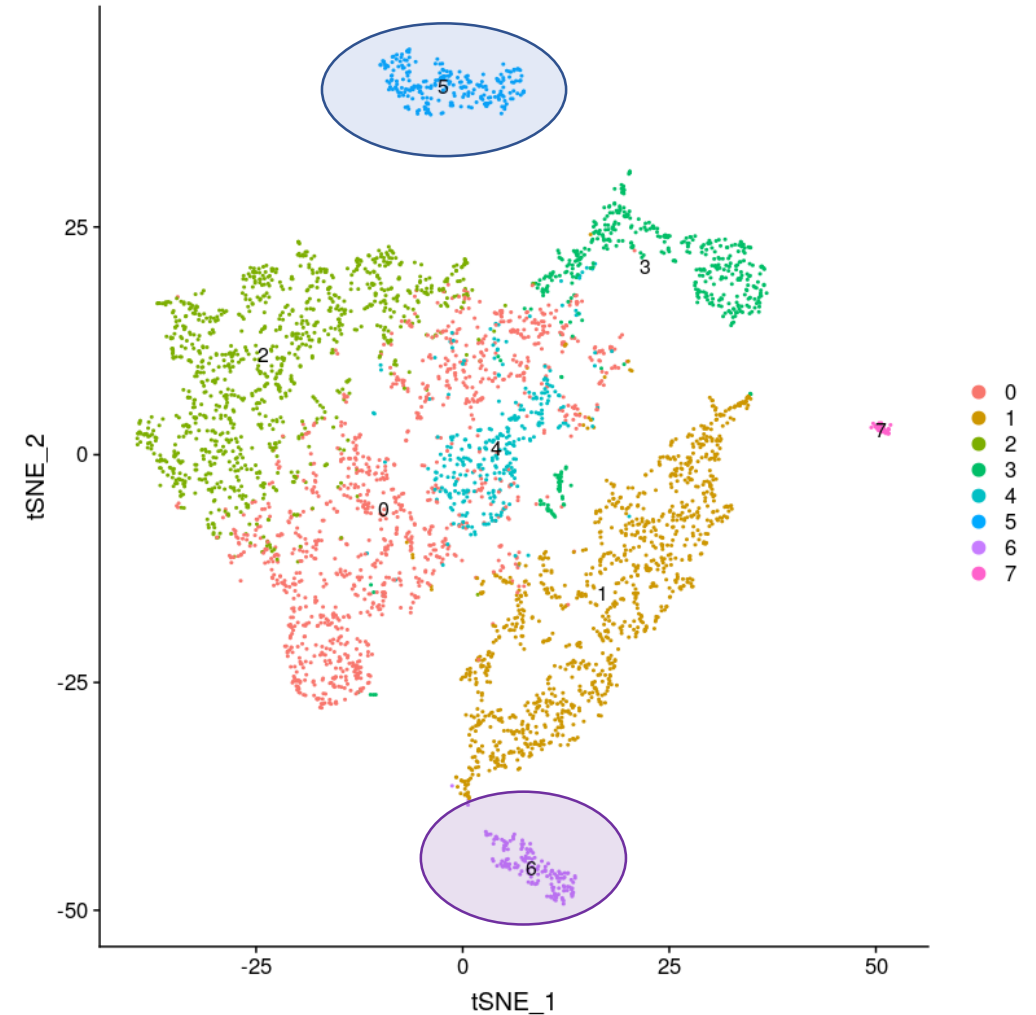
Wilcoxon test using Seurat

```
markers = FindMarkers(s_obj, ident.1 = 5,  
ident.2 = 6)  
markers["PDYN",]
```

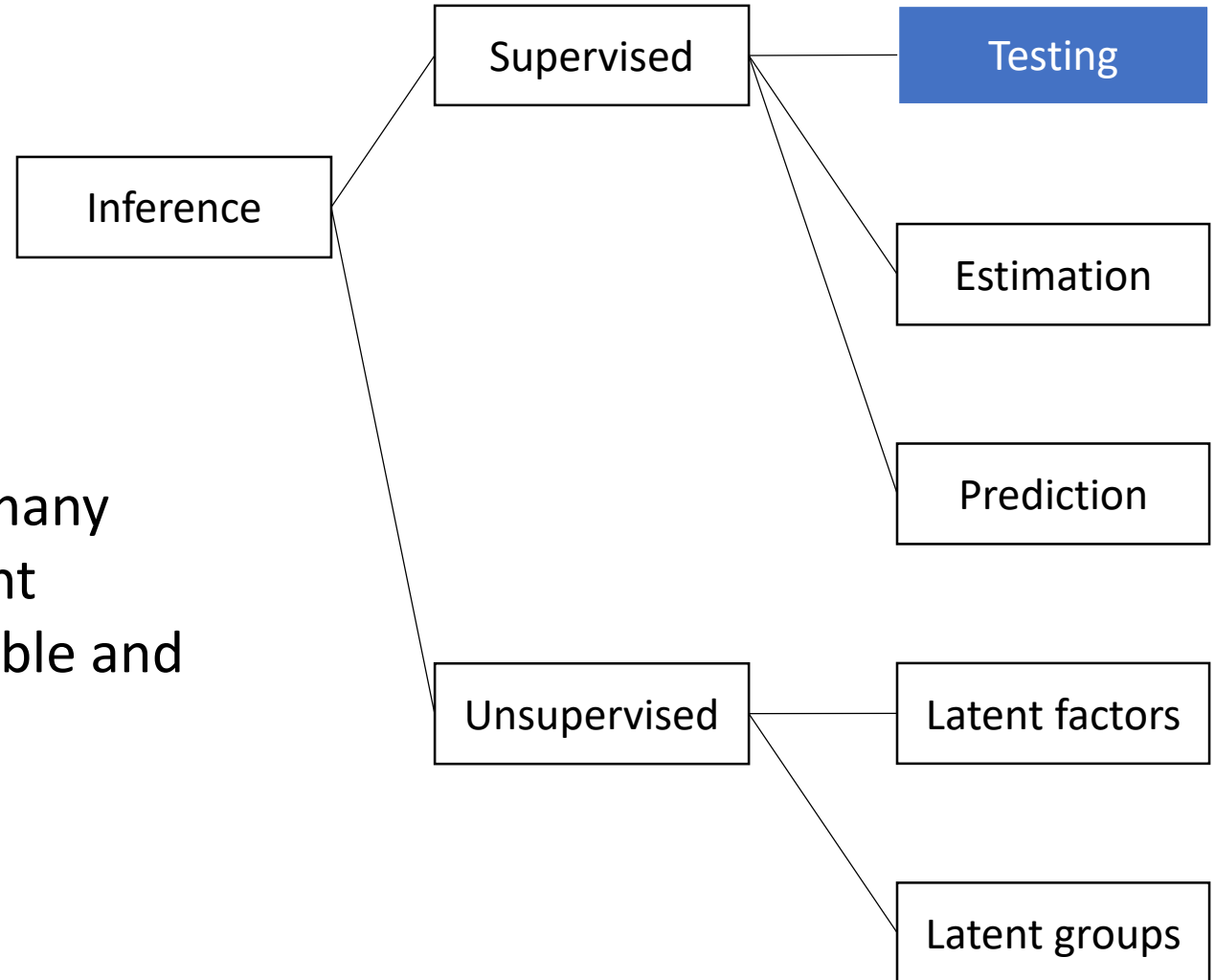
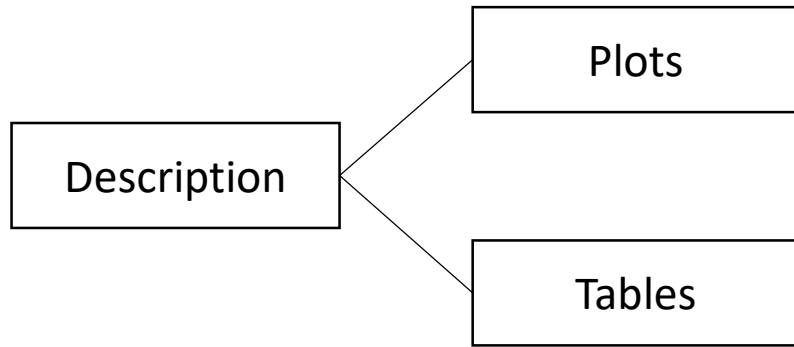
FDR control

Research question:

Which genes differ between clusters 5 and 6?



FDR control



Statistical task: identify which of many hypothesis tests are truly significant

Data structure: p-values are available and statistically independent

FDR control

- FDR = False Discovery Rate = expected value of

$$\frac{\text{\# false discoveries}}{\text{total \# of discoveries}}$$

- Statistical methods reject the largest number of hypothesis tests while maintaining $\text{FDR} \leq \alpha$, for some preset α

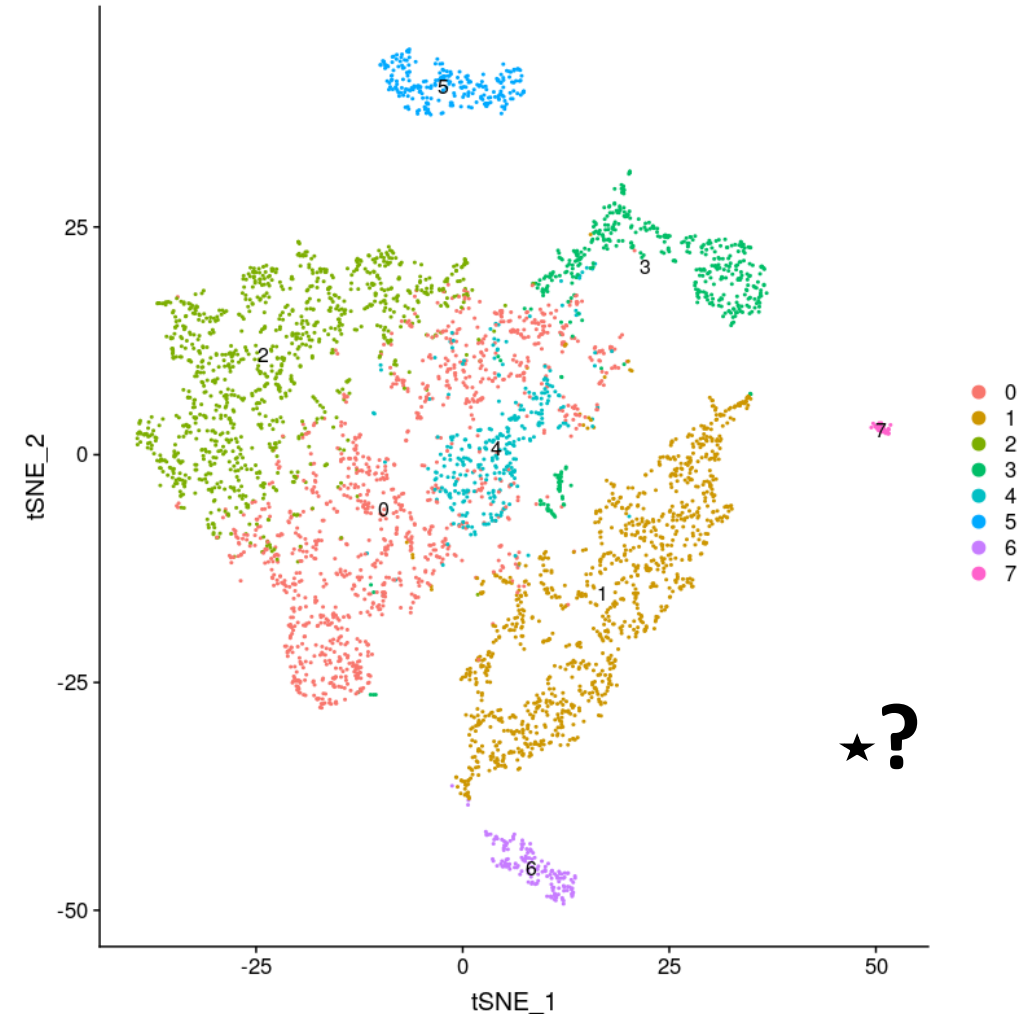
FDR control using Seurat

```
markers = FindMarkers(s_obj, ident.1 = 5,  
  ident.2 = 6)  
head(markers)  
sum(markers$p_val_adj <= 0.05)
```

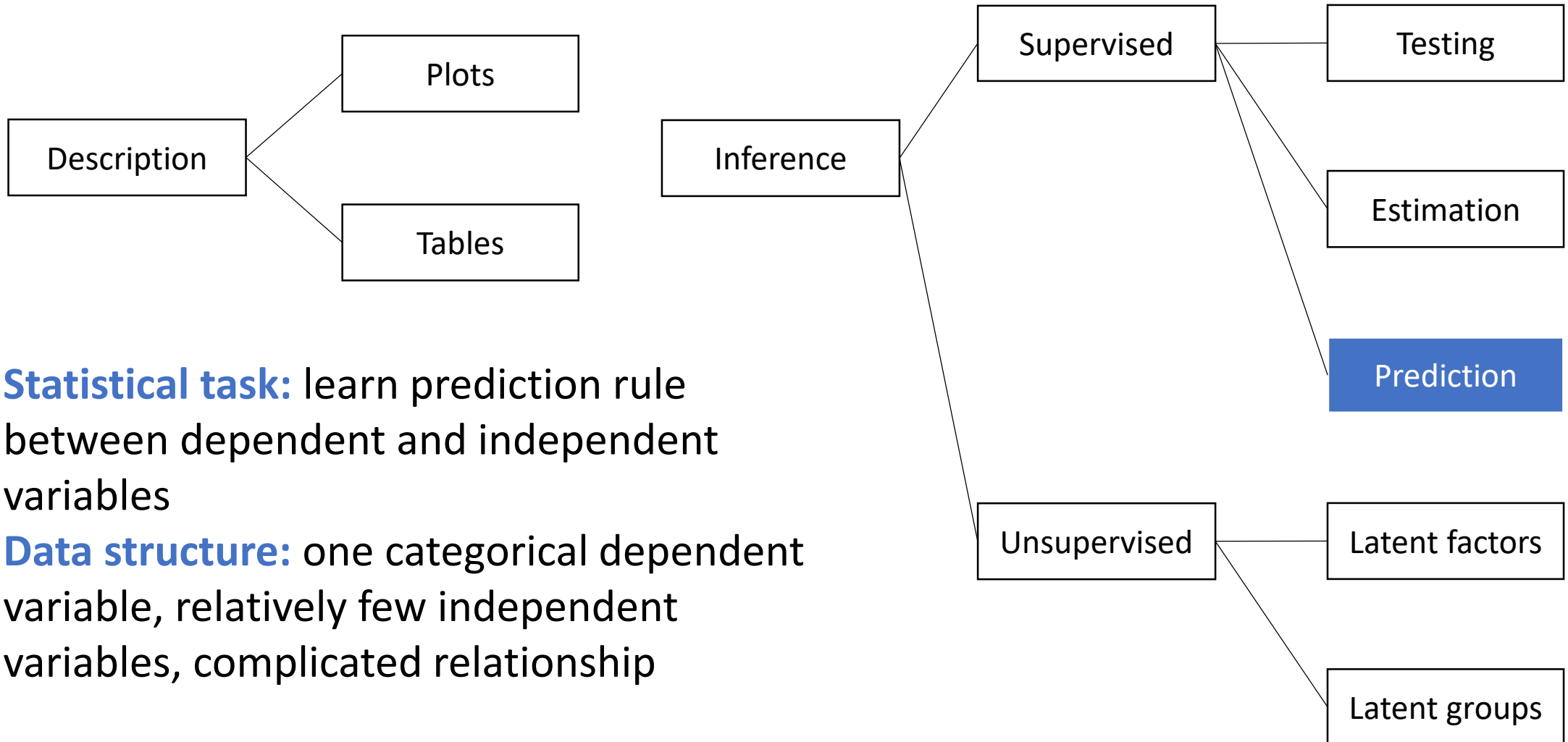
Random forest classification

Research question:

Given the principal components of the RNA-seq expression values of all genes from a new cell, how can we determine the cell's type?

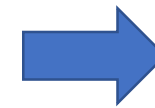
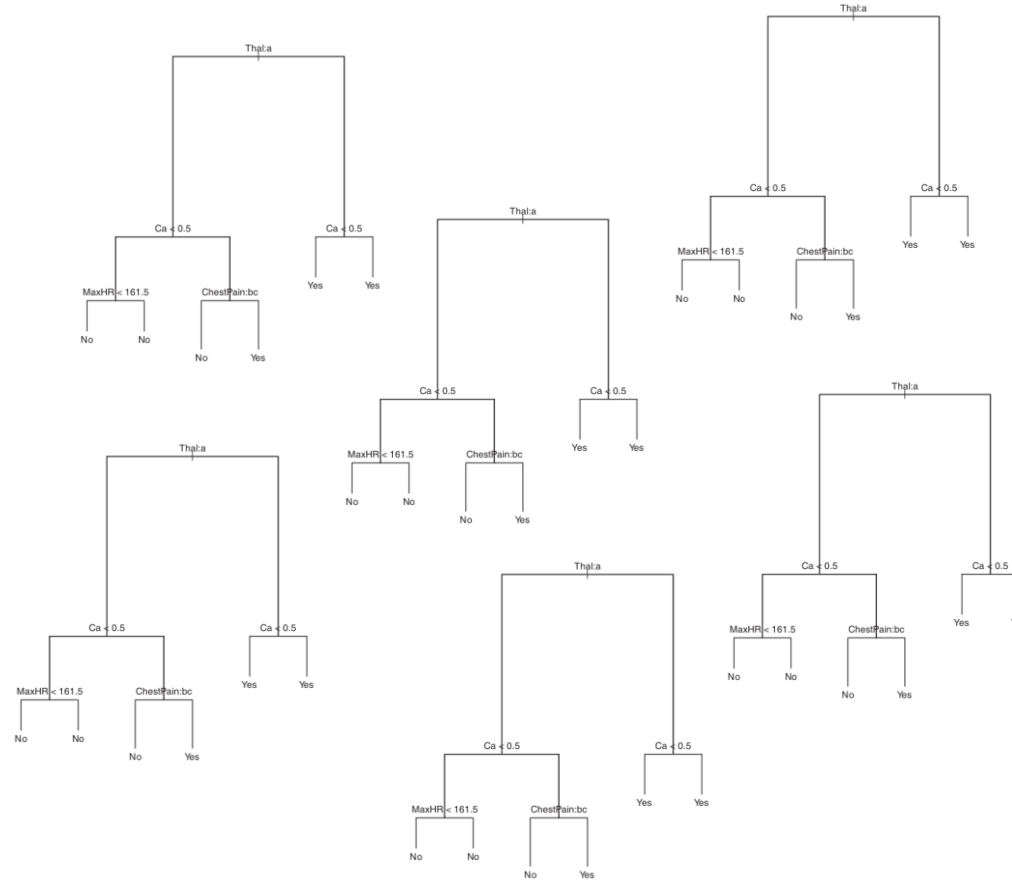
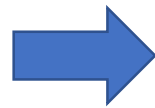


Random forest classification



Random forest classification

Independent
variables



Prediction

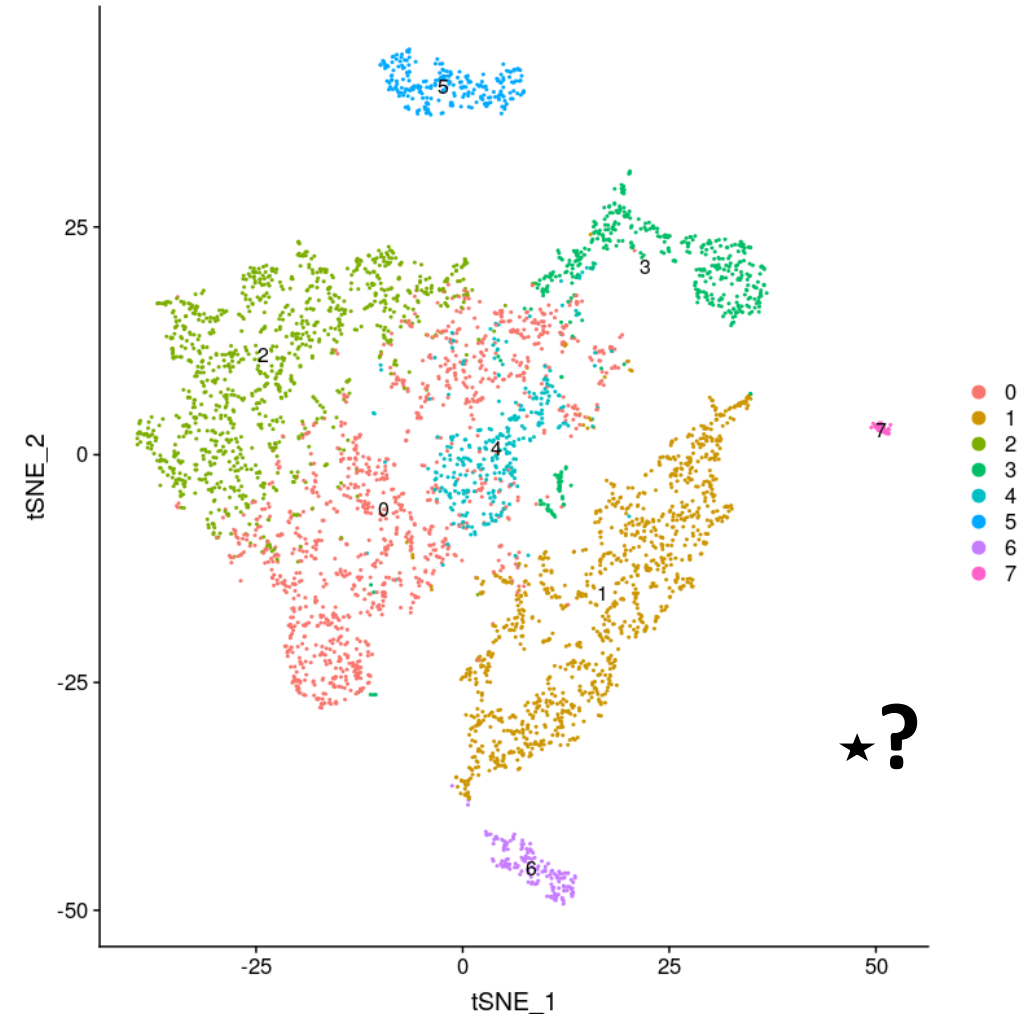
Random forest classification using caret and ranger

```
library(caret)
pcs = Embeddings(s_obj, reduction = "pca")[-(1:2), 1:10]
class = as.factor(Ids(s_obj))[-(1:2)]
dataset = data.frame(class, pcs)
rf_fit = train(class ~ .,
               data = dataset,
               method = "ranger",
               trControl = trainControl(method = "cv", number
= 3))
new_pcs = Embeddings(s_obj, reduction = "pca")[1:2, 1:10]
predict(rf_fit, new_pcs)
Ids(s_obj)[1:2]
```

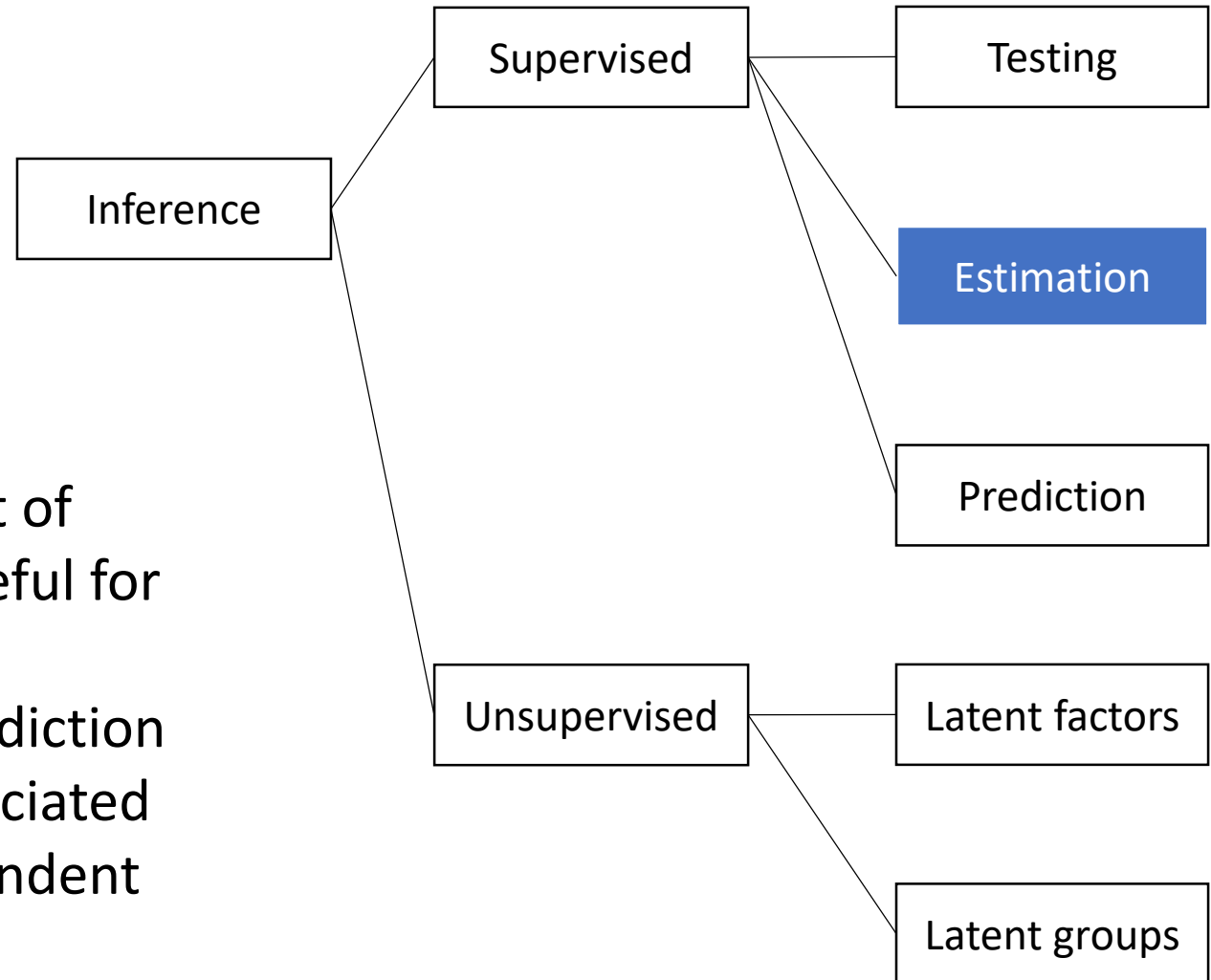
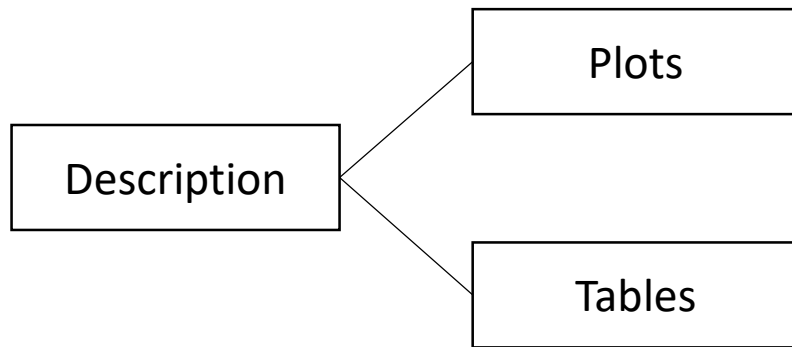
Lasso

Research question:

If we can only measure the expression of 10 genes in a new cell, which should we measure in order to most accurately predict the cell's type?



Lasso



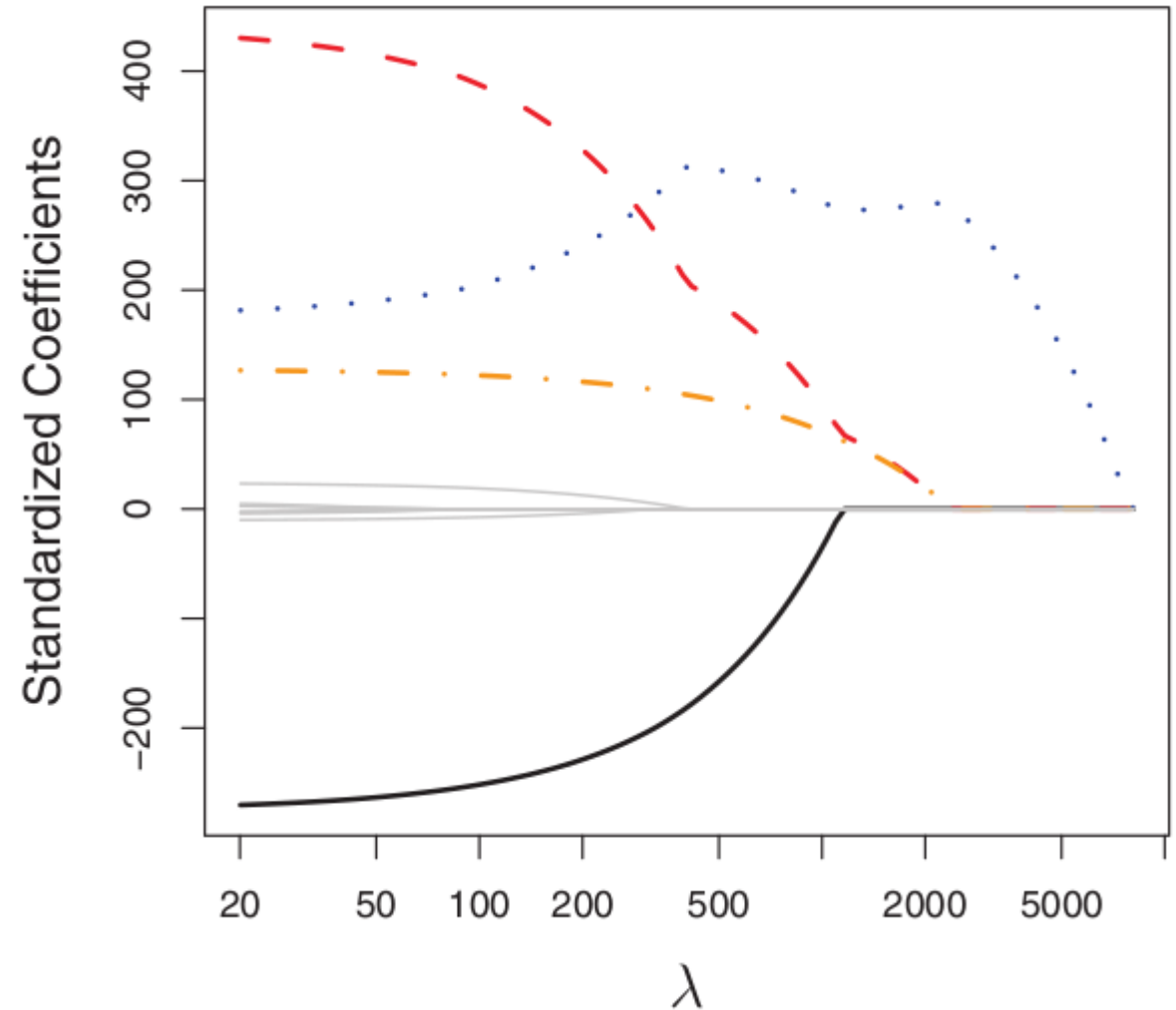
Statistical task: identify a small set of independent variables that are useful for predicting dependent variables

Data structure: simple (linear) prediction rule, dependent variables are associated with only a few (unknown) independent variables

Lasso

Estimates coefficients in the regression model

$$Y = \beta_0 + X_1\beta_1 + \cdots + X_p\beta_p$$

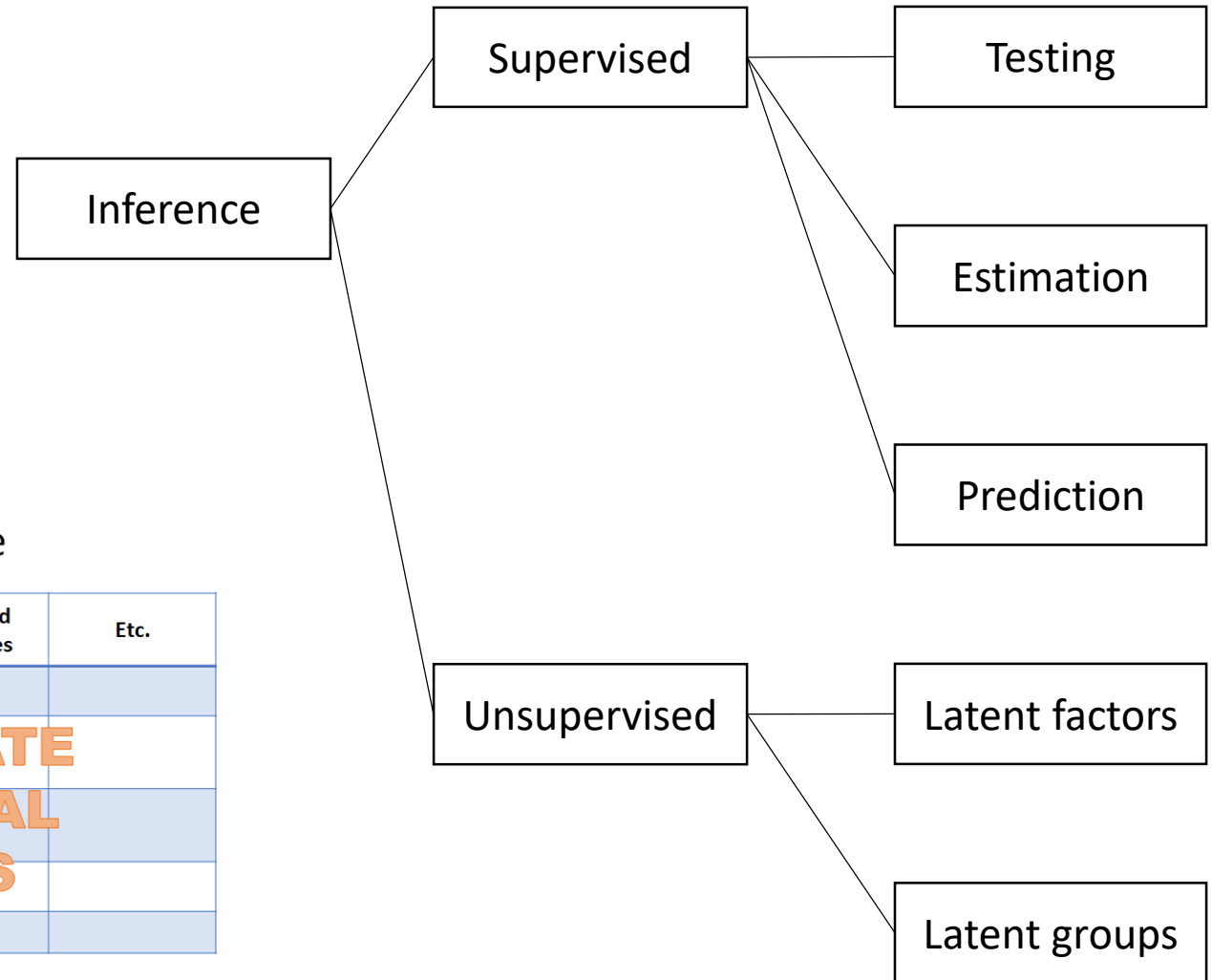
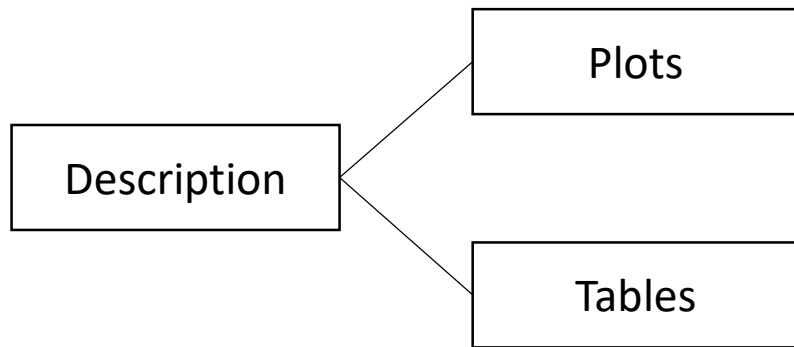


Lasso using glmnet

```
library(glmnet)
counts = t(GetAssayData(s_obj, slot = "counts"))[-(1:2),]
pcs = Embeddings(s_obj, reduction = "pca")[-(1:2), 1:10]
lasso = cv.glmnet(counts, pcs, family = "mgaussian", nfolds =
3)
lambda = min(lasso$lambda[lasso$nzero <= 10])
coefs = coef(lasso, s = lambda)
rownames(coefs$PC_1)[which(coefs$PC_1 != 0)]
new_counts = t(GetAssayData(s_obj, slot = "counts"))[1:2,]
new_pcs = predict(lasso, newx = new_counts, s = lambda)[,, 1]
predict(rf_fit, new_pcs)
Idents(s_obj)[1:2]
```

Conclusions

Statistical toolbox



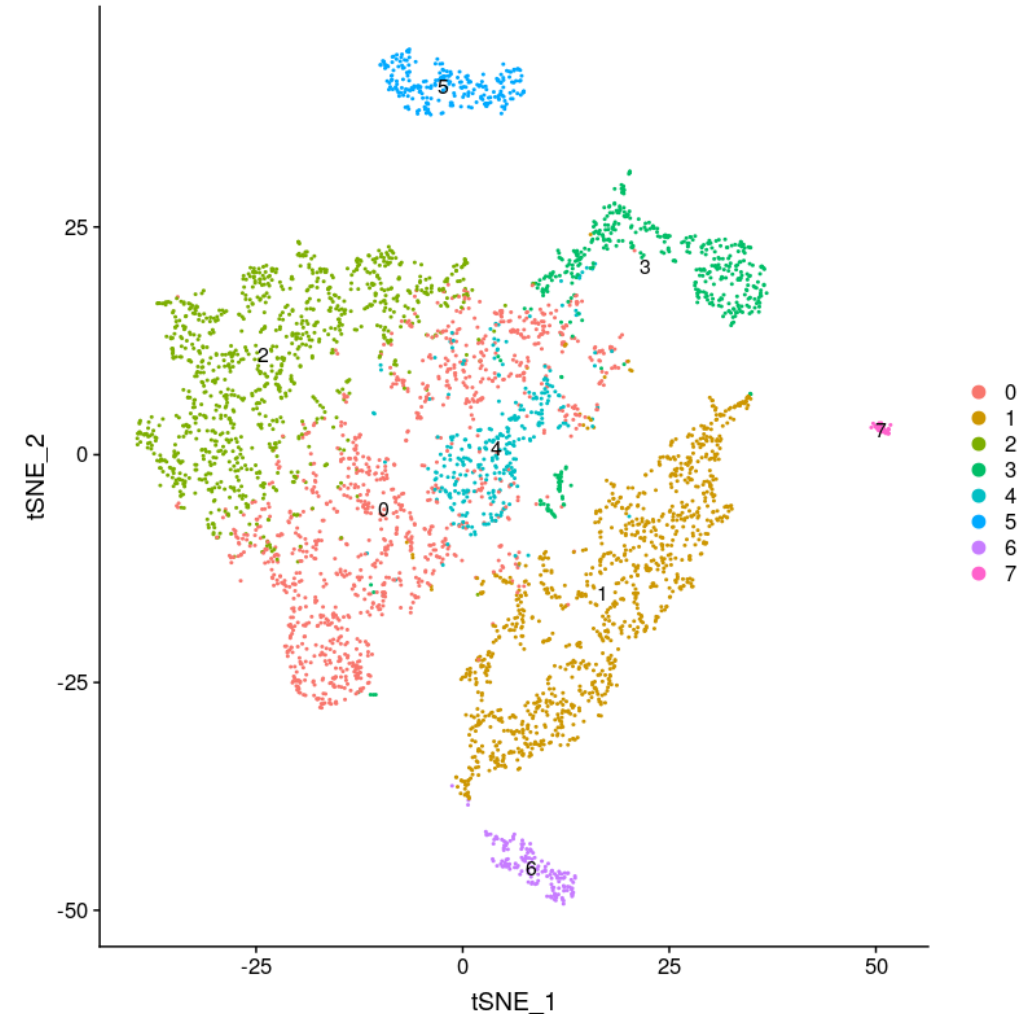
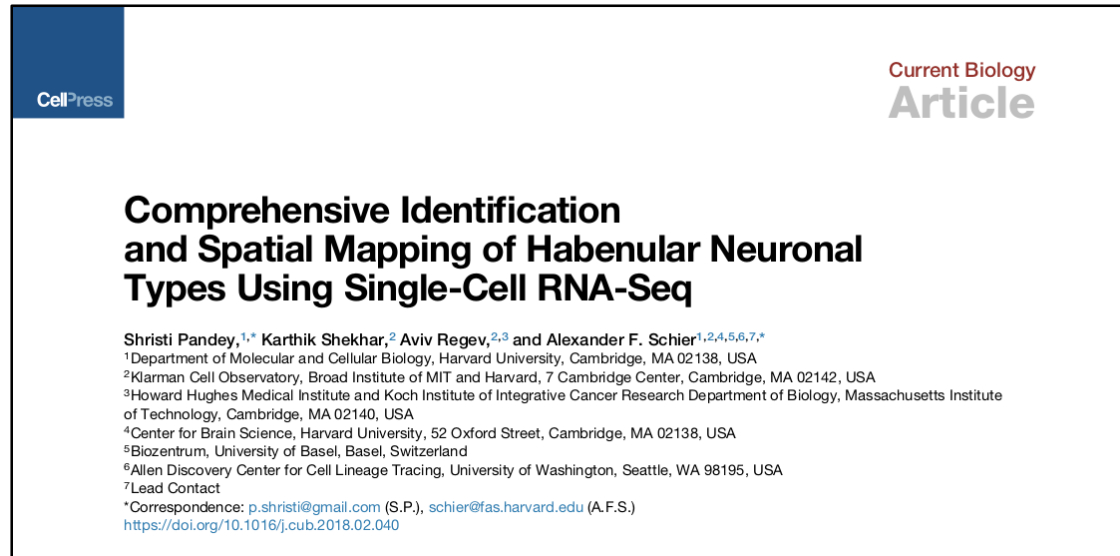
Statistical task

	Data structure			
	No dependent variables	Continuous outcome	Censored outcomes	Etc.
Visualize				
Identify latent factors	APPROPRIATE STATISTICAL METHODS			
Cluster observations				
Select features				
Etc.				

Statistical tools

1. PCA
2. Graph clustering
3. t-SNE plot
4. Wilcoxon test
5. FDR control
6. Random forest classification
7. Lasso

Can be applied to single-cell RNA-seq and beyond



To learn more

- Take systematic courses in basic statistics, statistical learning, and R/python
- Study recently published papers in your field of interest that use your technology of interest
- Consult tutorials, workshops, lab mates, and Google

Thank you

