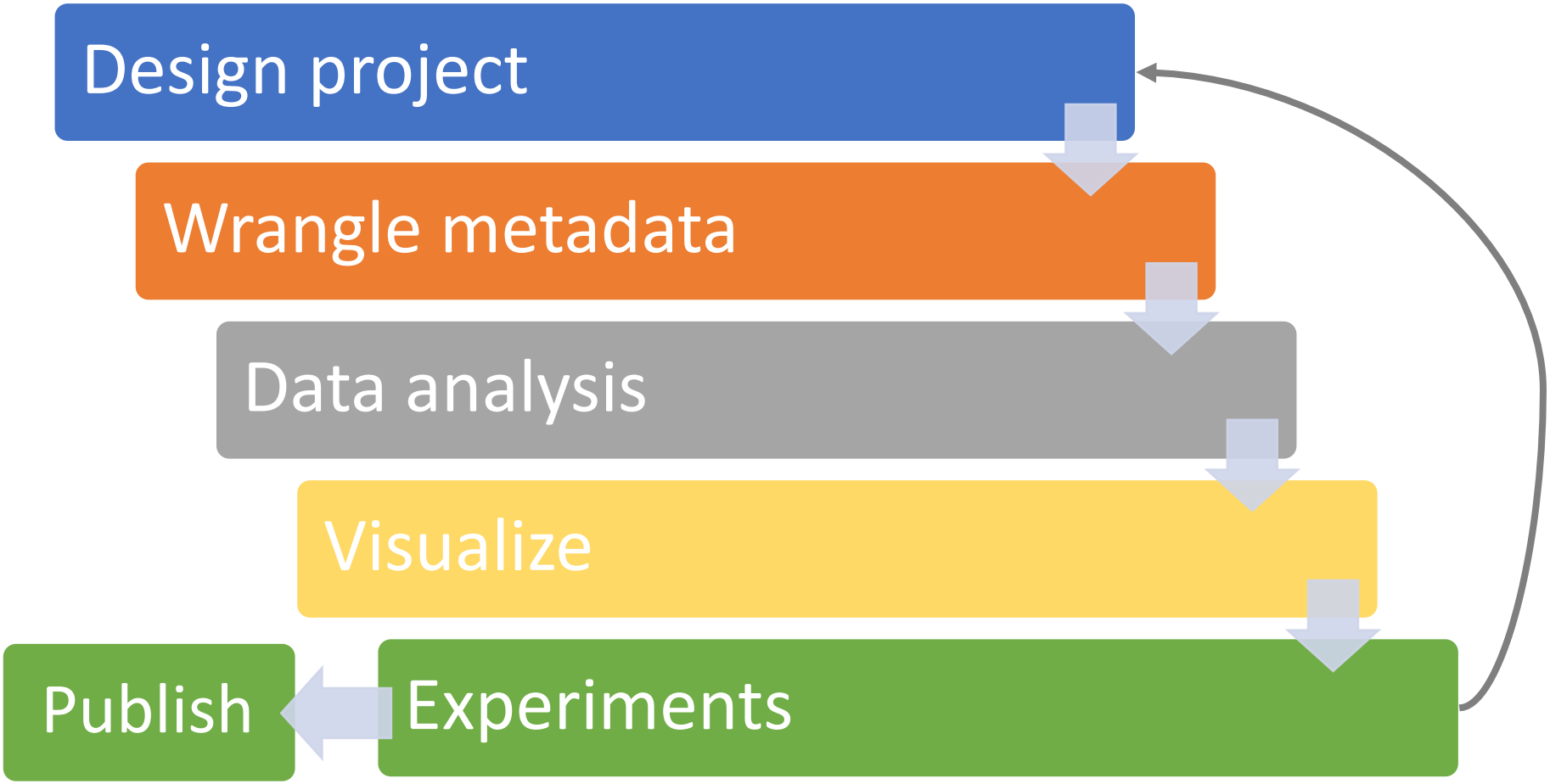
DNA microarray analysis lab

BIOTRAC Lecture
Sean, Soonweng, Cho, PhD
Kennedy Krieger Institute
and
Johns Hopkins University School of Medicine

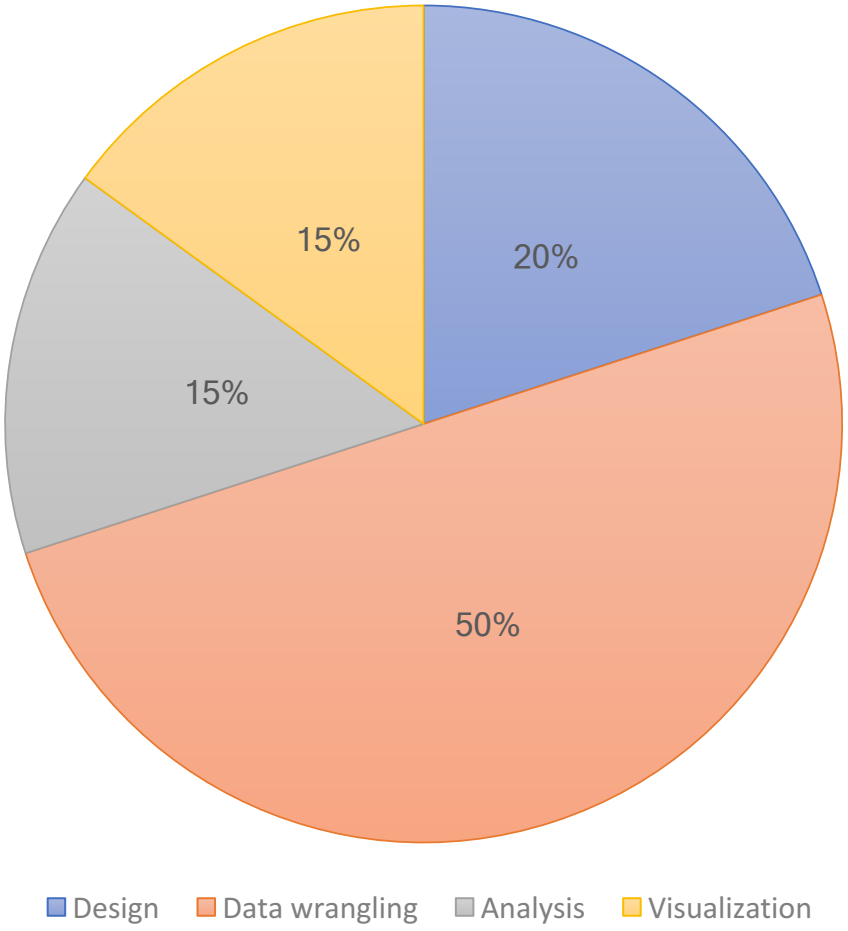

sean.cho@jhmi.edu


github.com/sean-cho

Anatomy of a bioinformatics project



Approximate time spent on each task

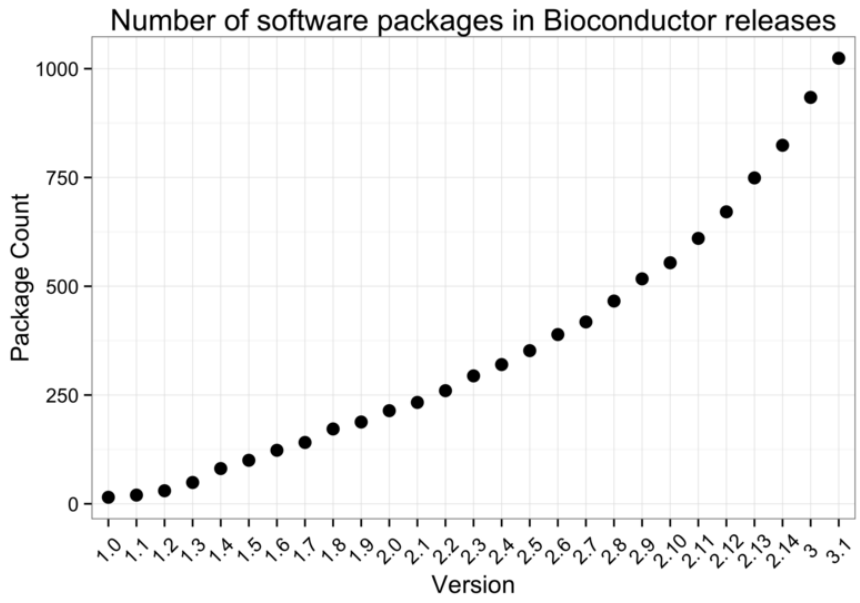


Goals

- Introduction to R and basic data wrangling
- Analysis 1: Squamous cell carcinoma
  - Download data from GEO
  - Read raw data
  - Preprocess data
  - Identifying differential methylation
  - Visualize analysis
- Analysis 2: Breast cancer
  - Download data from GEO
  - Preprocess non idat data
  - Identify differentially methylated probes (DMPs)
  - Clustering analysis

Why R?

- Free
- Powerful
- Flexible
- Plenty of online resources for learning
- Large community of active educators
- Many open source packages and tutorials for biology
- Active scientist and developers on Github and Bioconductor to offer support for these packages



Introduction to R

The R programming language

- R is a functional programming language that supports object-oriented programming
- Functional programming language
  - Computation as evaluation of mathematical functions

```
function(x) <- do something to x
```

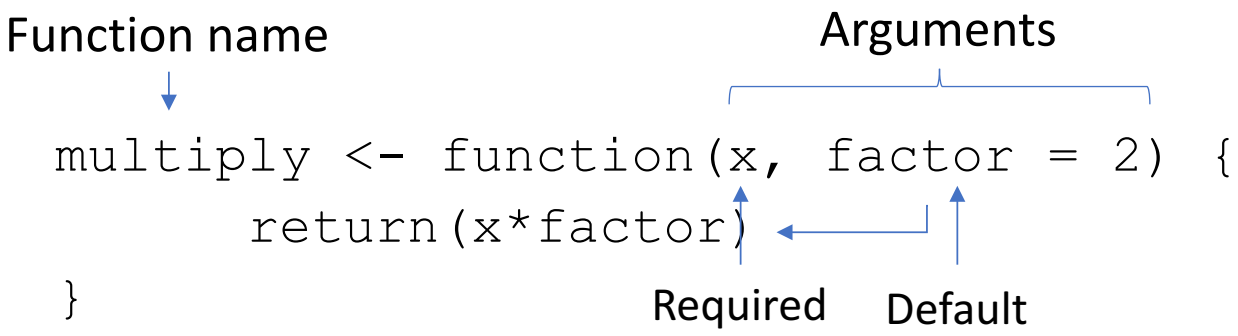
- Object-oriented programming
  - Data or functions are stored as objects
  - Manipulate objects using functions

```
data <- c(1,2,3)
sum(data)
## 6
```

- Getting help

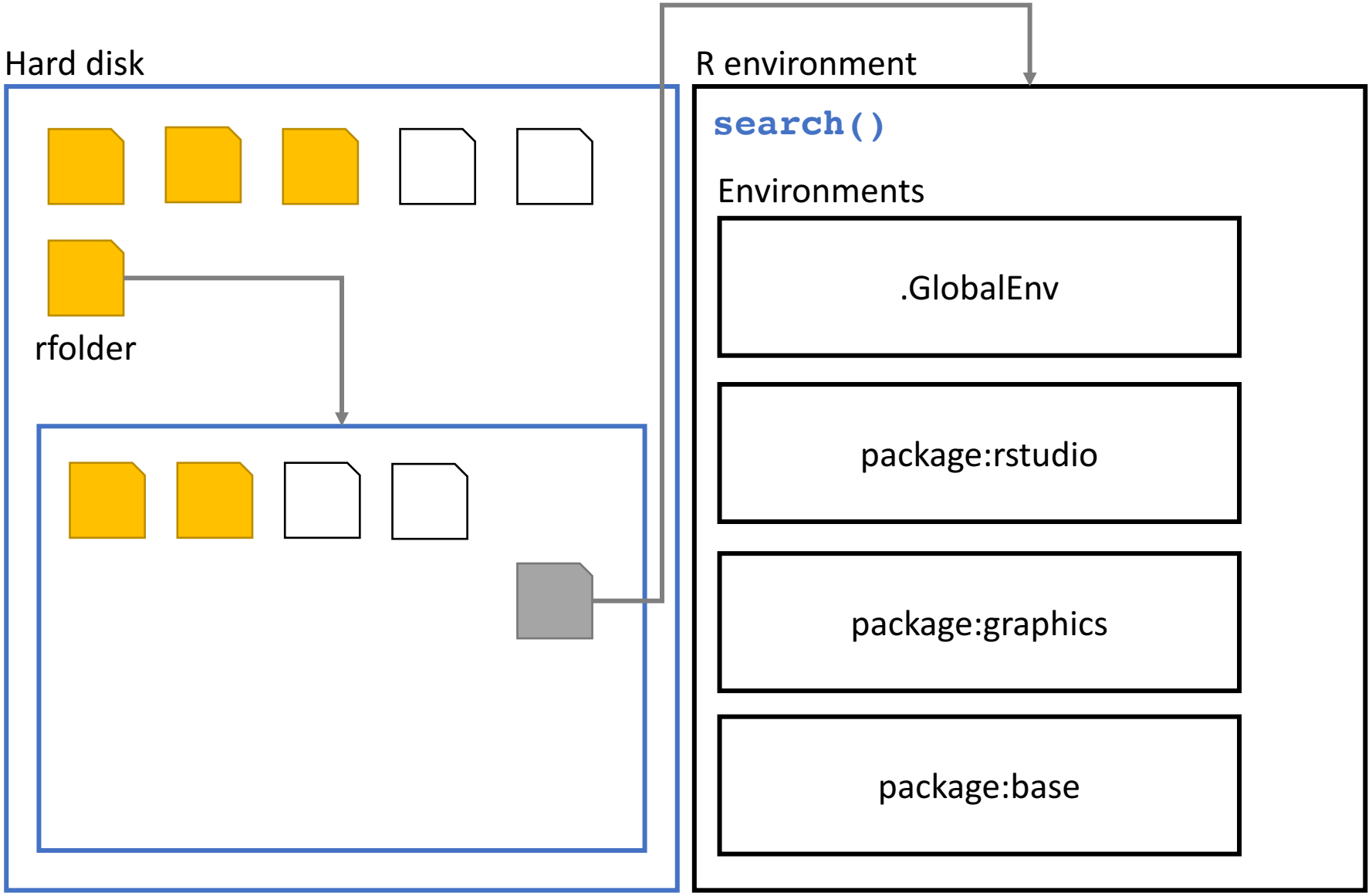
```
?t.test
help('t.test')
?mtcars
```

The anatomy of a function

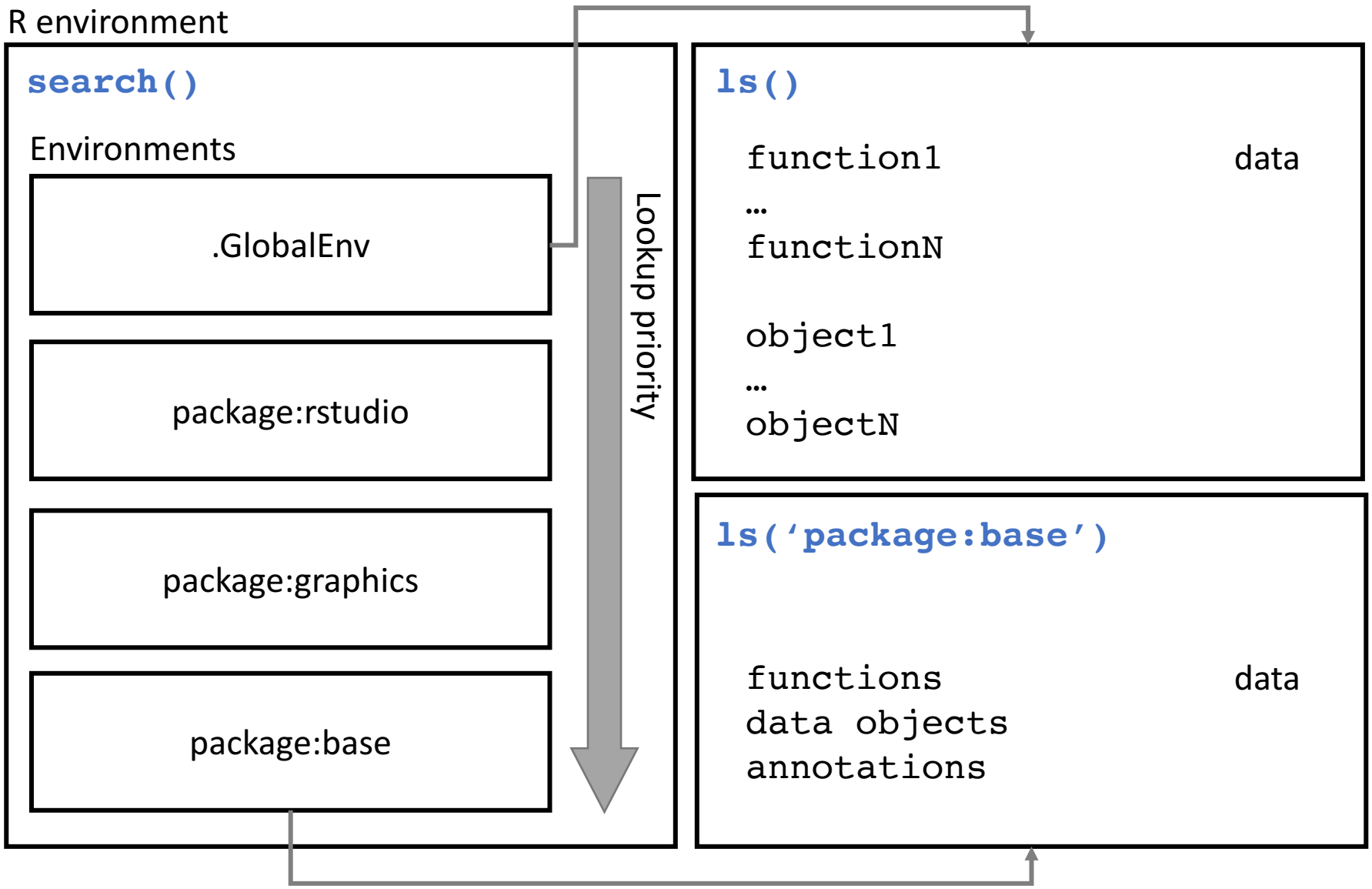



```
multiple <- function(x, factor=2){
  return(x*factor)
}
multiple(c(2,3,4))
#> [1] 4 6 8
multiple(c(2,3,4), factor = 3)
#> [1]  6  9 12
```

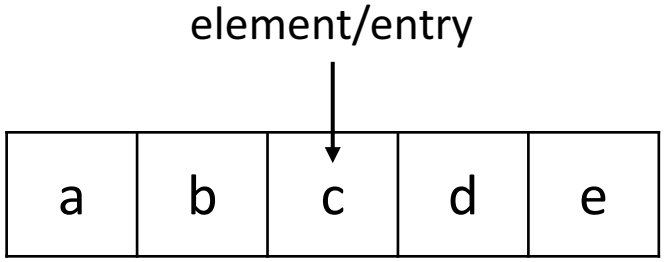
The R statistical environment



The R statistical environment



Data objects: Vectors



- One dimensional object
- Different `classes` (or types)
  - Integer

    ```
    box <- c(1,2,3)
    ```
  - Numeric

    ```
    box <- c(1.1,1.2,1.3)
    ```
  - Character

    ```
    box <- c('ctrl','tx', 'tx')
    ```
  - Factor

    ```
    box <- factor('ctrl','tx','tx')
    ```

Data objects: Matrices

- All same `class`

character	character	character	character
Sample ID	**Status**	**Treatment**	**Methylation**
Patient01	Disease	Treated	0.93
Patient02	Disease	Untreated	0.23
Control01	Control	Untreated	0.12

Data objects: Data.Frame

- Columns can be different `classes`

`character` `factor` `factor` `numeric`

Sample ID	Status	Treatment	Methylation
Patient01	Disease	Treated	0.93
Patient02	Disease	Untreated	0.23
Control01	Control	Untreated	0.12

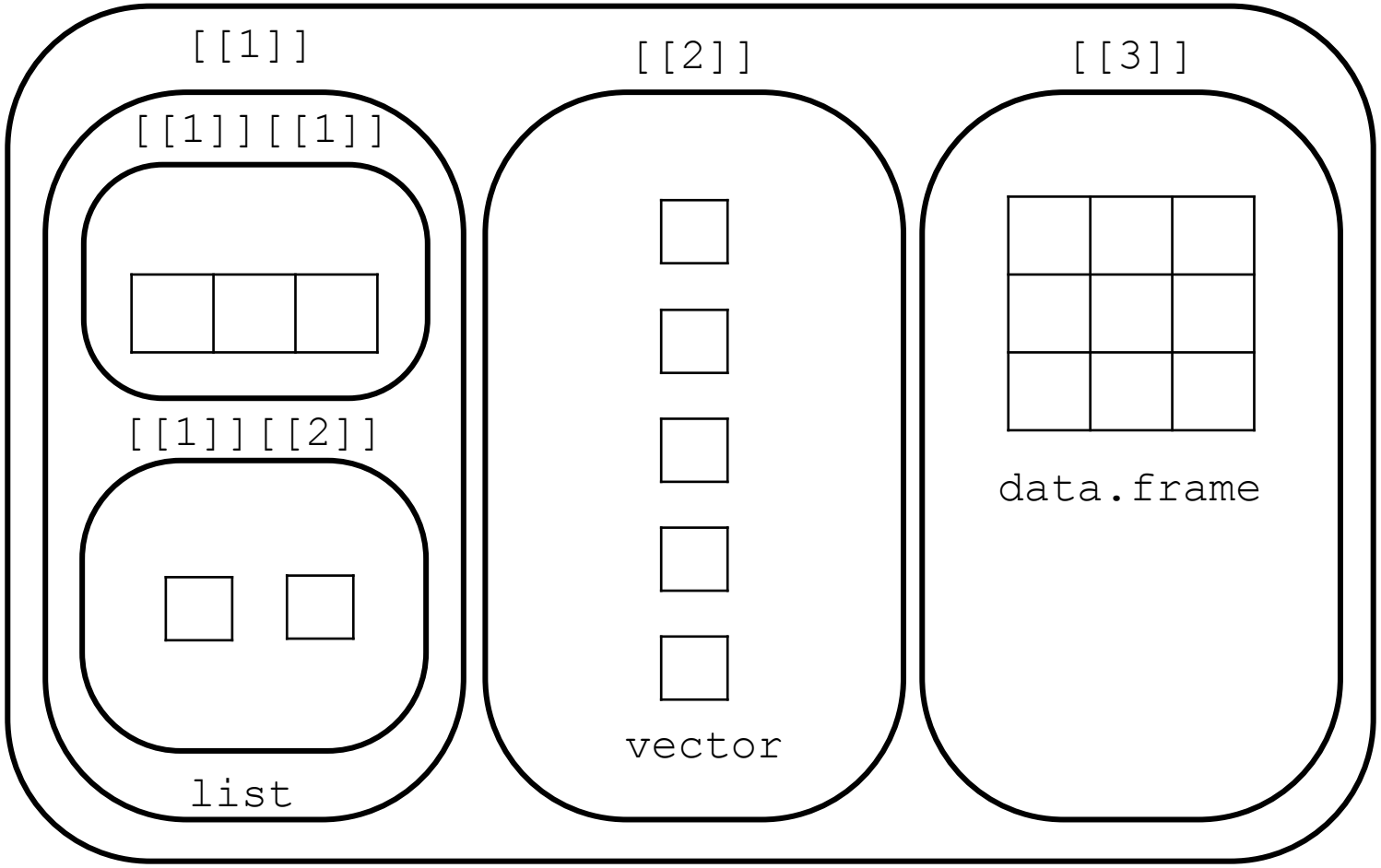
```r
dfbox <- data.frame(
  id = c('p1','p2','p3','p4','p5'),
  c1 = 1:5, c2= 5:1)
mbox <- as.matrix(dfbox)
sum(dfbox[,2])
#> [1] 15
sum(mbox[,2])
#> Error in sum(mbox[, 2]): invalid 'type' (character) of argument
```

Data objects: Lists

- A hierarchical object which can contain elements of different classes
- Can even contain lists
- A container of many boxes

```
box1 <- 1:3
box2 <- c('a','b','c')
box3 <- data.frame(a=1:5,b=6:10)
container <- list(box1,box2,box3)
bigger_container <- list(container, box1)
```

Data objects: Lists



```
box1 <- 1:3
box2 <- c('a','b','c')
box3 <- data.frame(a=1:5,b=6:10)
container <- list(box1,box2,box3)
container
#> [[1]]
#> [1] 1 2 3
#>
#> [[2]]
#> [1] "a" "b" "c"
#>
#> [[3]]
#>   a  b
#> 1 1  6
#> 2 2  7
#> 3 3  8
#> 4 4  9
#> 5 5 10
bigger_container <- list(container, box1)
bigger_container
#> [[1]]
#> [[1]][[1]]
#> [1] 1 2 3
#>
#> [[1]][[2]]
#> [1] "a" "b" "c"
#>
#> [[1]][[3]]
#>   a  b
#> 1 1  6
#> 2 2  7
#> 3 3  8
#> 4 4  9
#> 5 5 10
#>
#>
#> [[2]]
#> [1] 1 2 3
```

Creating data objects

- Assign values to objects
  - `<-` : Left assign
  - `->` : Right assign
  - `=` : Assign

- Make an object

```
box <- 1
box = 1
1 -> box
```

- Show the contents of an object

```
box
```

```
box <- 1
box
#> [1] 1
box = 1
box
#> [1] 1
1 -> box
box
#> [1] 1
```

Basic functions

- `c(i,j,…,n)` : concatenate or combine

```
box <- c(1,2,3)
```

- `:` : to

```
box <- 1:10
```

- `+` : sum
- – : minus
- `*` : multiply
- `/` : divide
- `matrix()` : create a matrix

```
box <- matrix(1:10, 5, 2) ## data, row, column
```

- `data.frame()` : create a data.frame

```
box <- data.frame(a=1:5,b=6:10)
## two columns, a and b
```

Basic functions

- `matrix()`: create a matrix

```
box <- matrix(1:10, 5, 2)
## data, row, column
```

- `data.frame()`: create a data.frame

```
box <- data.frame(a=1:5,b=6:10)
## two columns, a and b
```

- `rbind()`: row bind

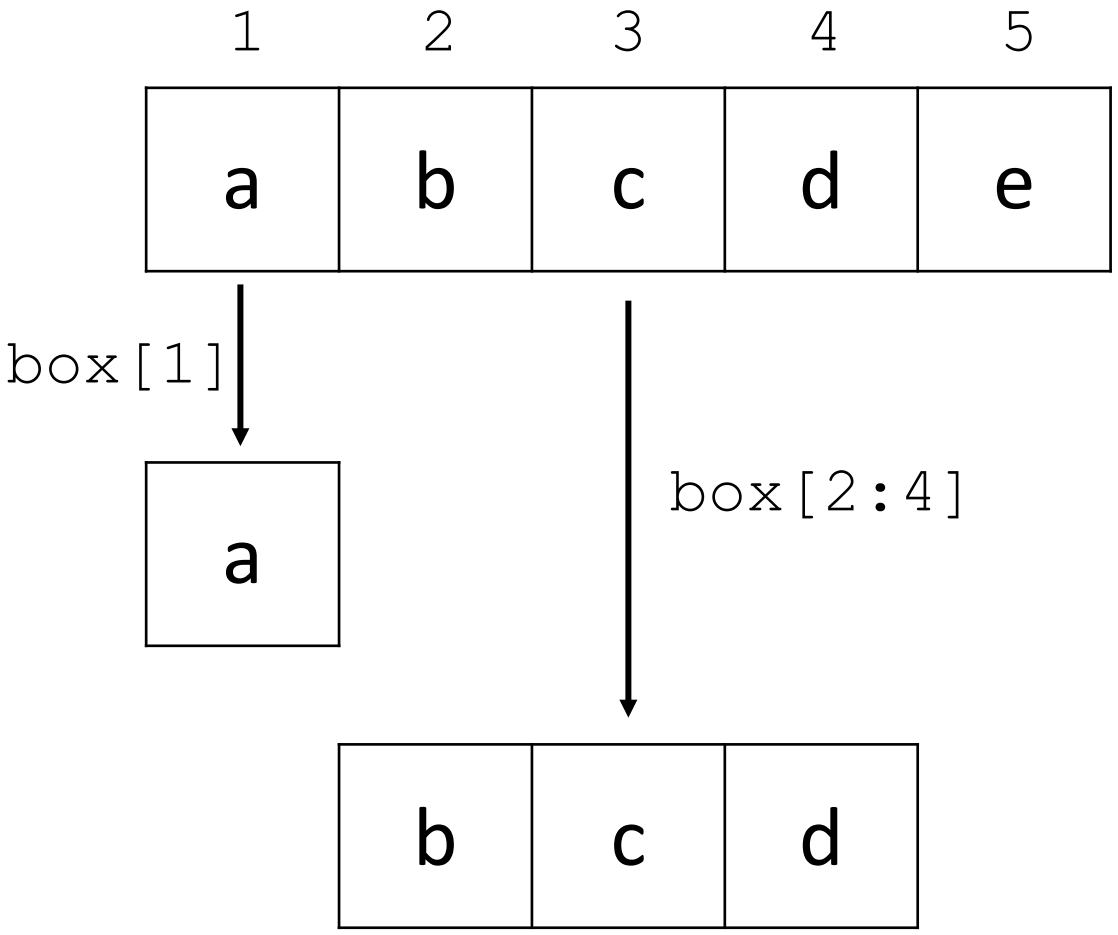
```
box <- rbind(1:3,4:6)
```

- `cbind()`: column bind

```
box <- cbind(1:3,4:6)
```

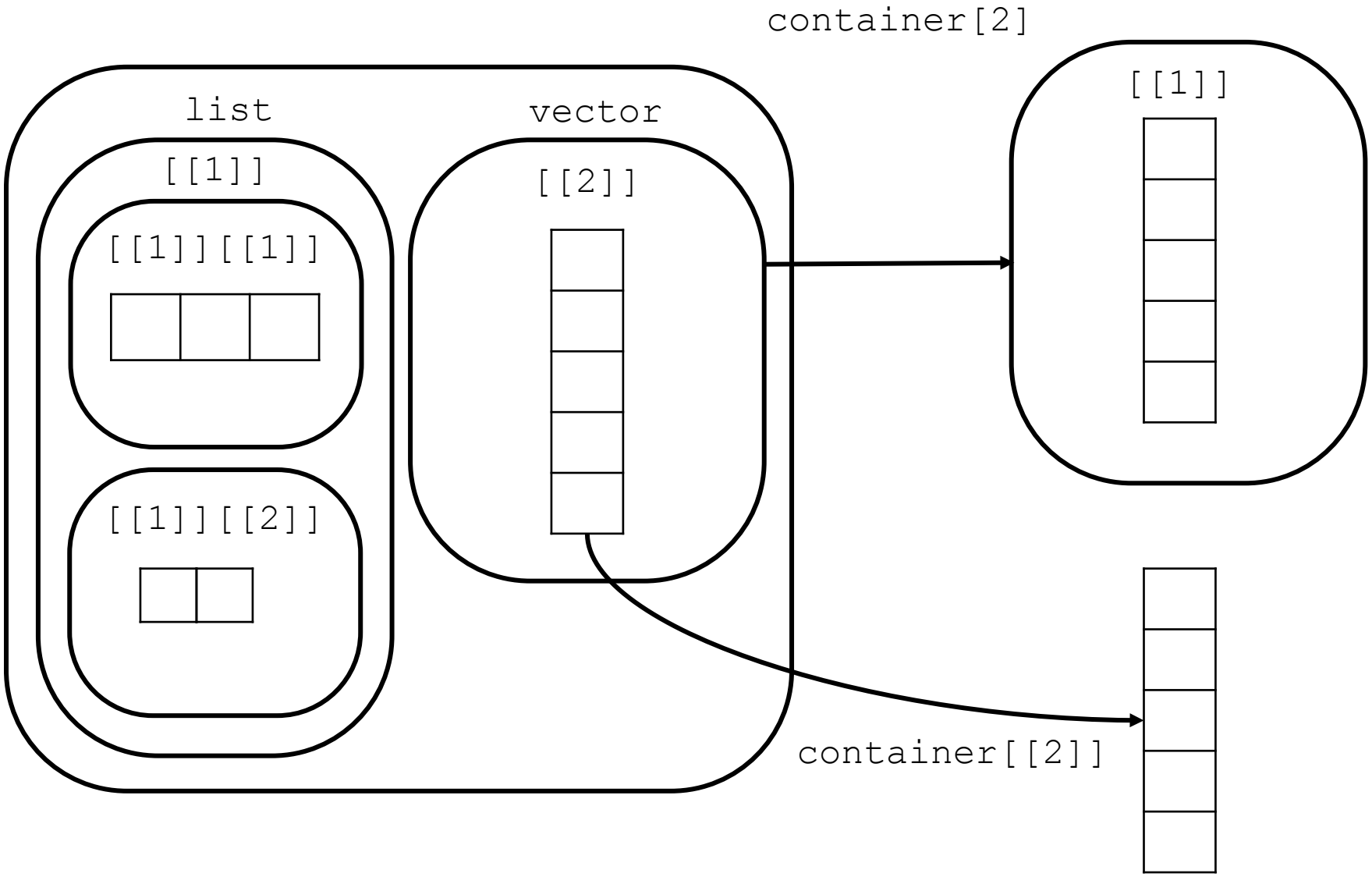
```
box <- c(1,2,3)
box
#> [1] 1 2 3
box <- 1:10
box
#>  [1]  1  2  3  4  5  6  7  8  9 10
box <- matrix(1:10, 5, 2) ## data, row, column
box
#>      [,1] [,2]
#> [1,]    1    6
#> [2,]    2    7
#> [3,]    3    8
#> [4,]    4    9
#> [5,]    5   10
box <- data.frame(a=1:5,b=6:10) ## two columns, a and b
box
#>   a  b
#> 1 1  6
#> 2 2  7
#> 3 3  8
#> 4 4  9
#> 5 5 10
box <- rbind(1:3,4:6)
box
#>      [,1] [,2] [,3]
#> [1,]    1    2    3
#> [2,]    4    5    6
box <- cbind(1:3,4:6)
box
#>      [,1] [,2]
#> [1,]    1    4
#> [2,]    2    5
#> [3,]    3    6
```

Subsetting: vectors



```
box <- c('a','b','c','d','e')
box[1]
#> [1] "a"
box[2:4]
#> [1] "b" "c" "d"
```

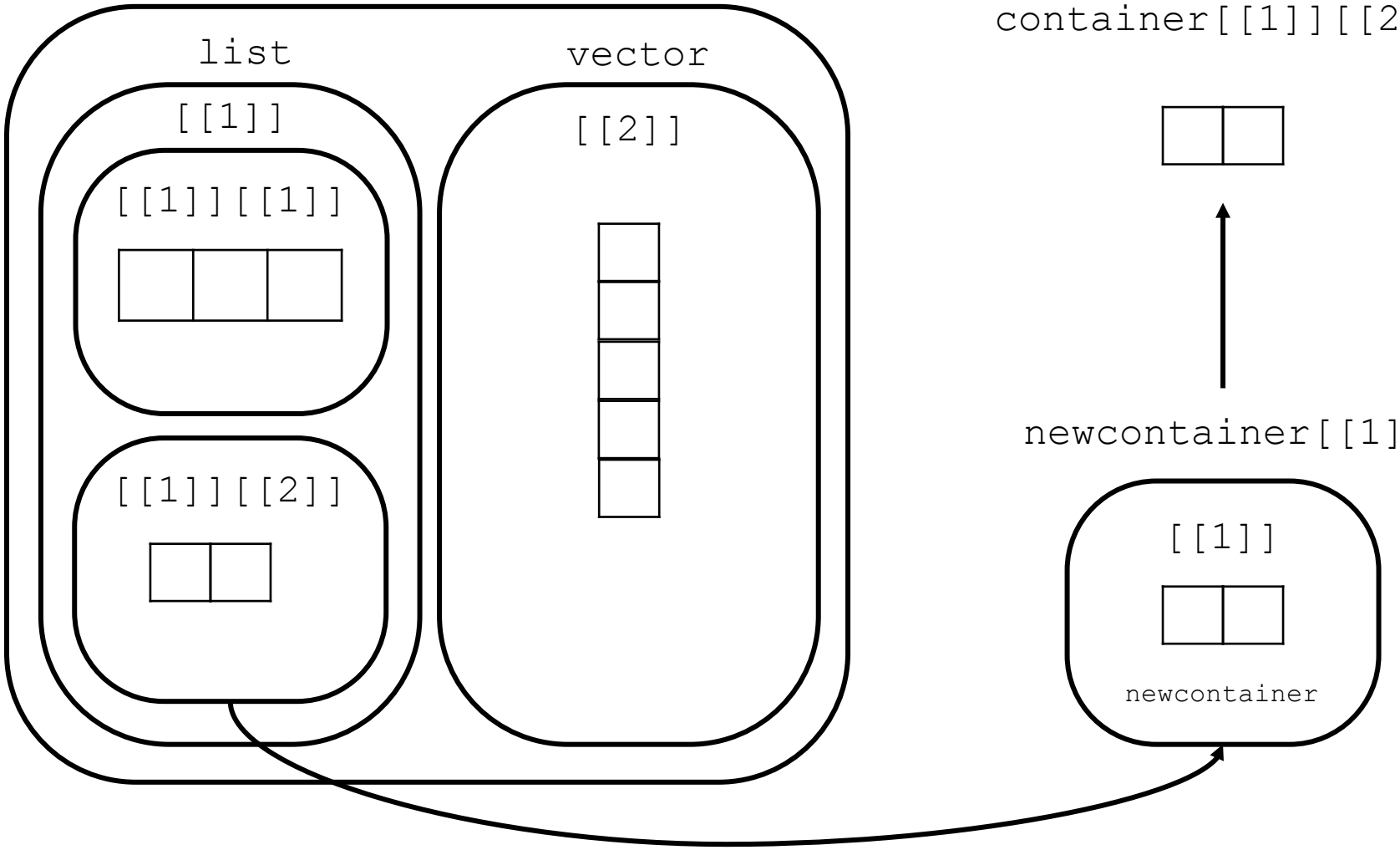
Subsetting: lists



Subsetting: lists

What if we want to extact this vector from container directly?

`container[[1]][[2]]`

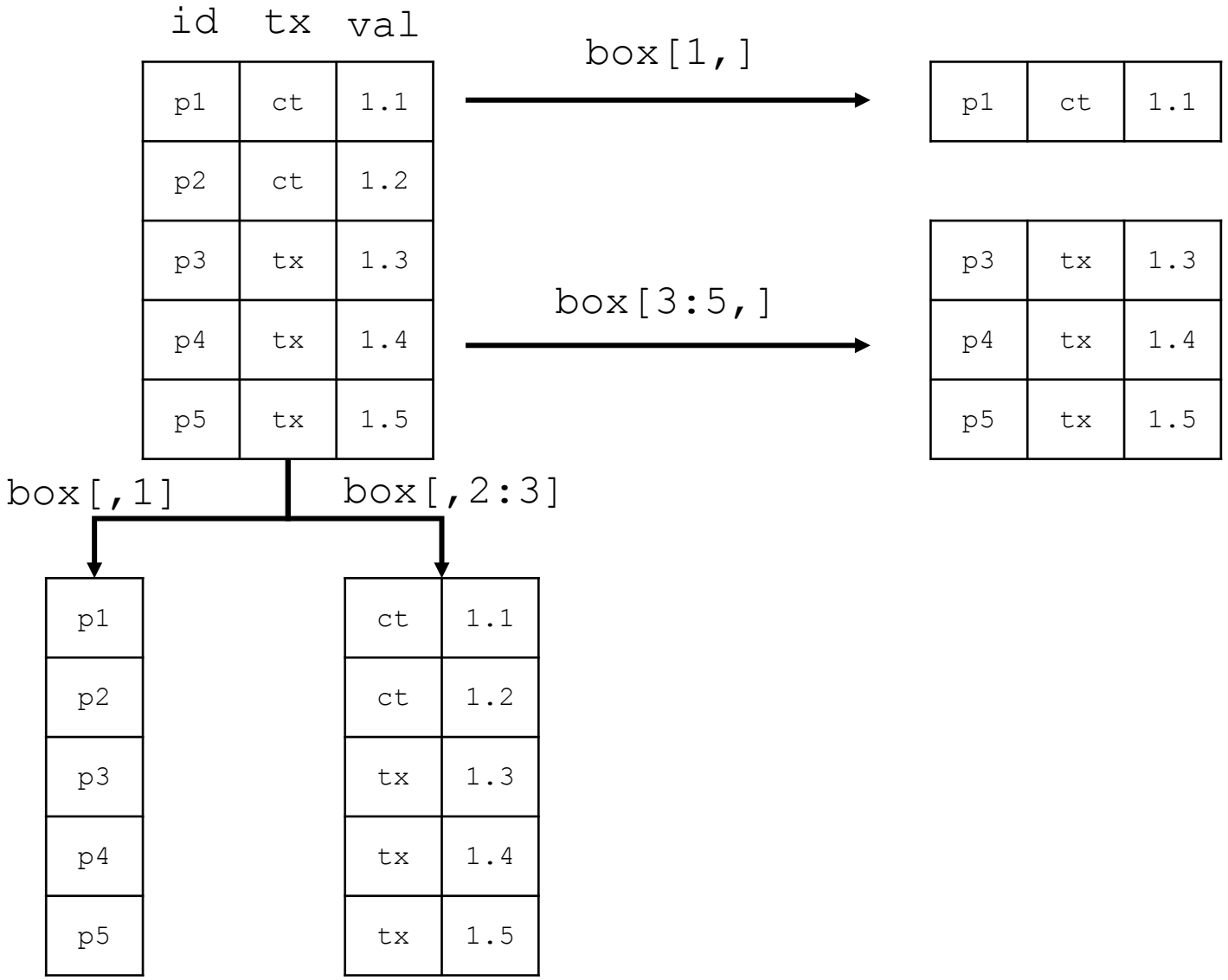


```
small_container <- list(1:3,1:2)
container <- list(small_container, 1:5)
container
#> [[1]]
#> [[1]][[1]]
#> [1] 1 2 3
#>
#> [[1]][[2]]
#> [1] 1 2
#>
#>
#> [[2]]
#> [1] 1 2 3 4 5
## subset 1
container[2]
#> [[1]]
#> [1] 1 2 3 4 5
container[[2]]
#> [1] 1 2 3 4 5
## subset 2
newcontainer <- container[[1]][2]
newcontainer
#> [[1]]
#> [1] 1 2
newcontainer[[1]]
#> [1] 1 2
## immediately get previous vector
container[[1]][[2]]
#> [1] 1 2
```

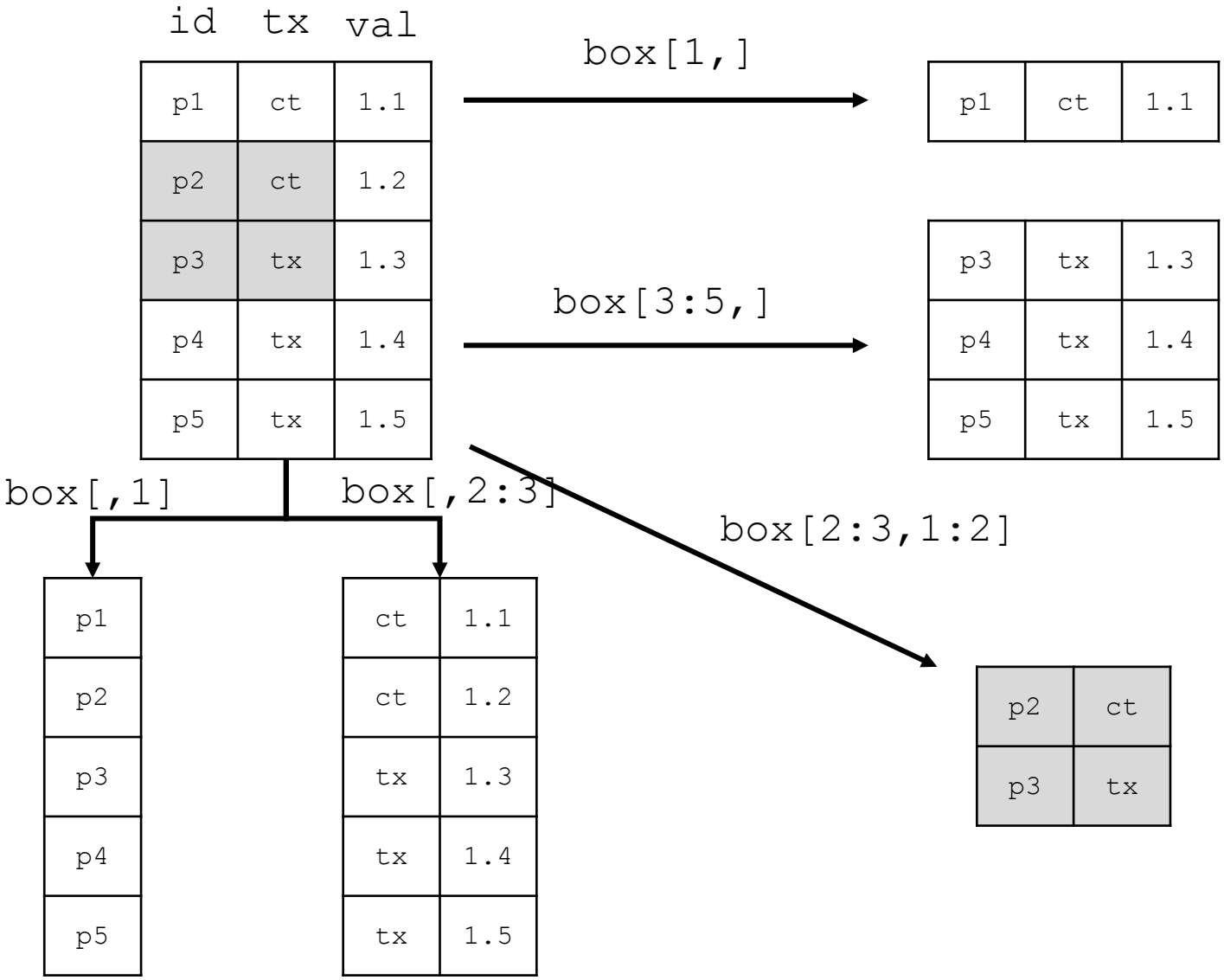
Subsetting: data.frames

	id	tx	value
r1	p1	ctrl	1.1
r2	p2	ctrl	1.2
r3	p3	tx	1.3
r4	p4	tx	1.4
r5	p5	tx	1.5

Subsetting: data.frames `df[row,column]`



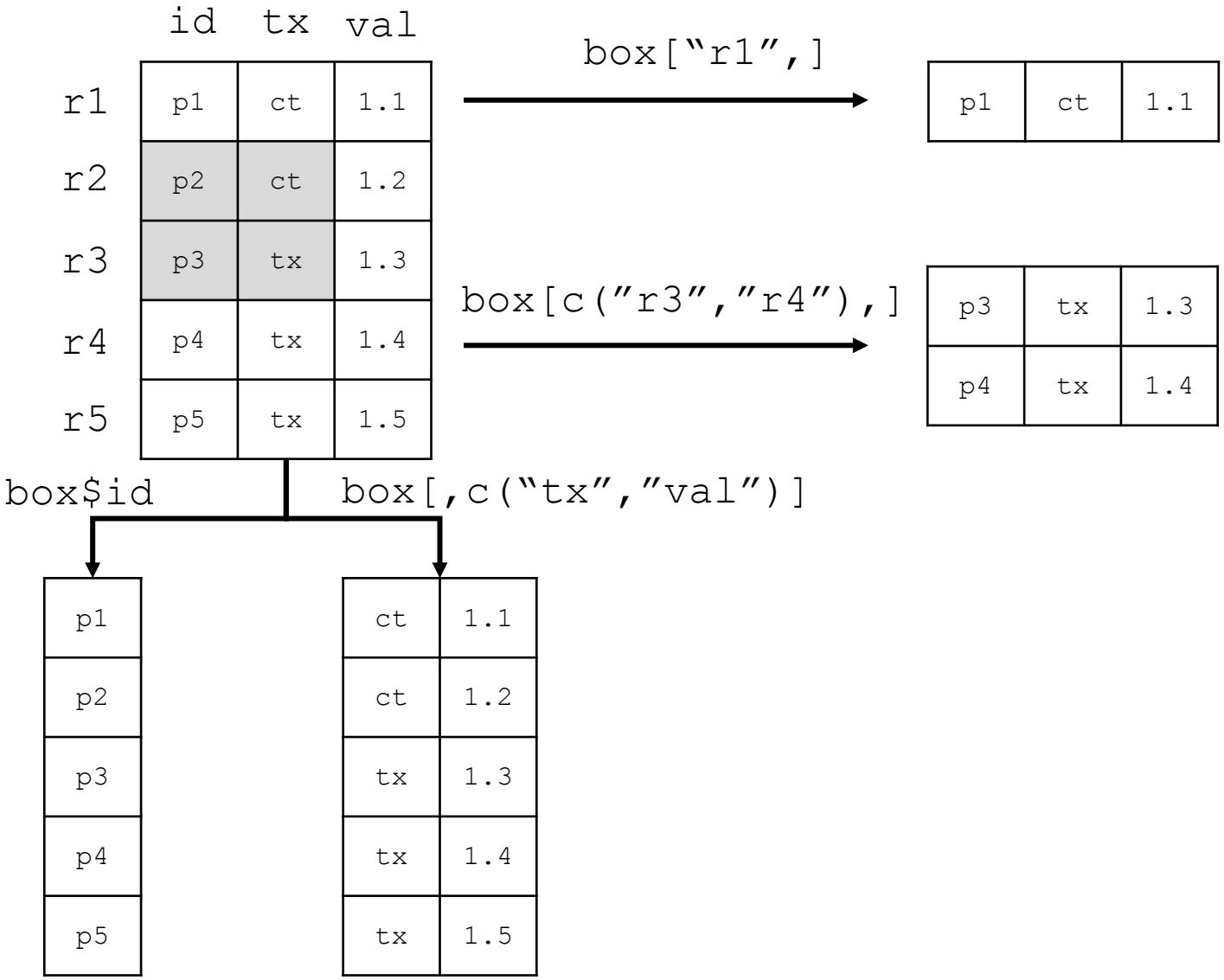
Subsetting: data.frames `df[row,column]`



```
box <- data.frame(
  id = c('p1','p2','p3','p4','p5'),
  tx = factor('ct','ct','tx','tx'),
  val = c(1.1,1.2,1.3,1.4,1.5),
  stringsAsFactors = FALSE)
box
#>   id tx val
#> 1 p1 tx 1.1
#> 2 p2 tx 1.2
#> 3 p3 tx 1.3
#> 4 p4 tx 1.4
#> 5 p5 tx 1.5
box[1,]
#>   id tx val
#> 1 p1 tx 1.1
box[3:5,]
#>   id tx val
#> 3 p3 tx 1.3
#> 4 p4 tx 1.4
#> 5 p5 tx 1.5
box[,1]
#> [1] "p1" "p2" "p3" "p4" "p5"
box[,2:3]
#>   tx val
#> 1 tx 1.1
#> 2 tx 1.2
#> 3 tx 1.3
#> 4 tx 1.4
#> 5 tx 1.5
box[2:3,1:2]
#>   id tx
#> 2 p2 tx
#> 3 p3 tx
```

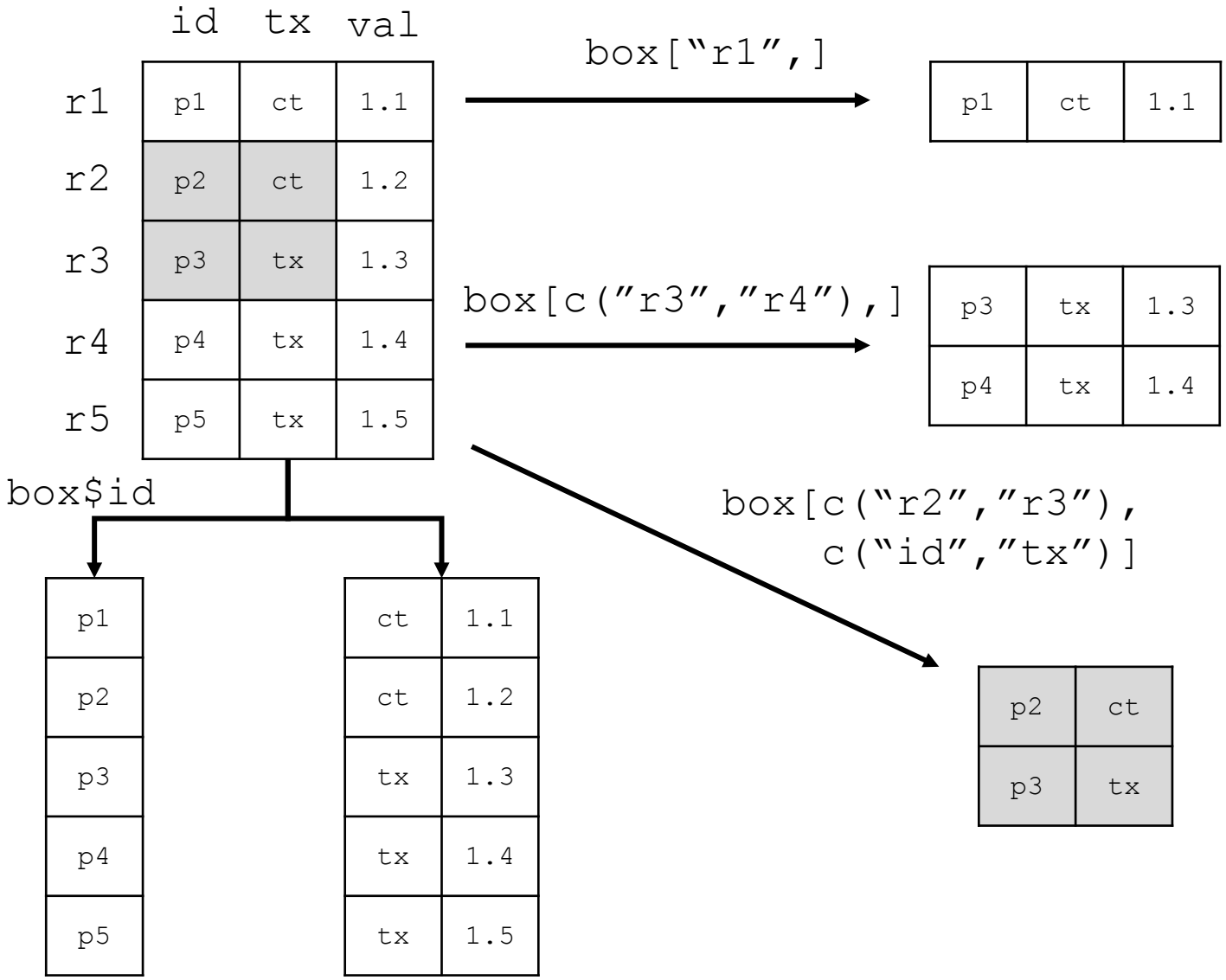
Subsetting: data.frames

```
df$colname
df['rowname',
  'colname']
```



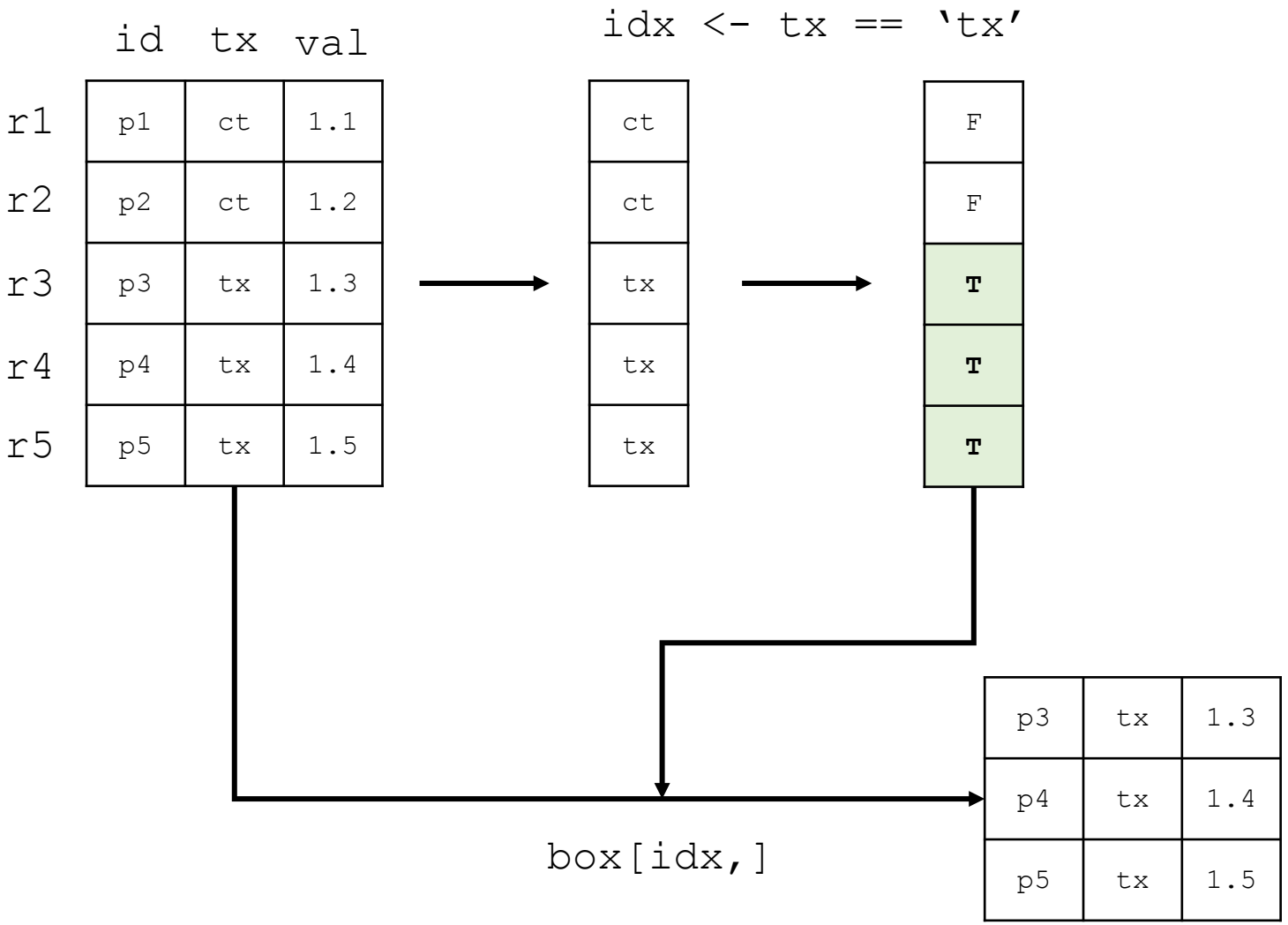
Subsetting: data.frames

```
df$colname
df['rowname',
  'colname']
```



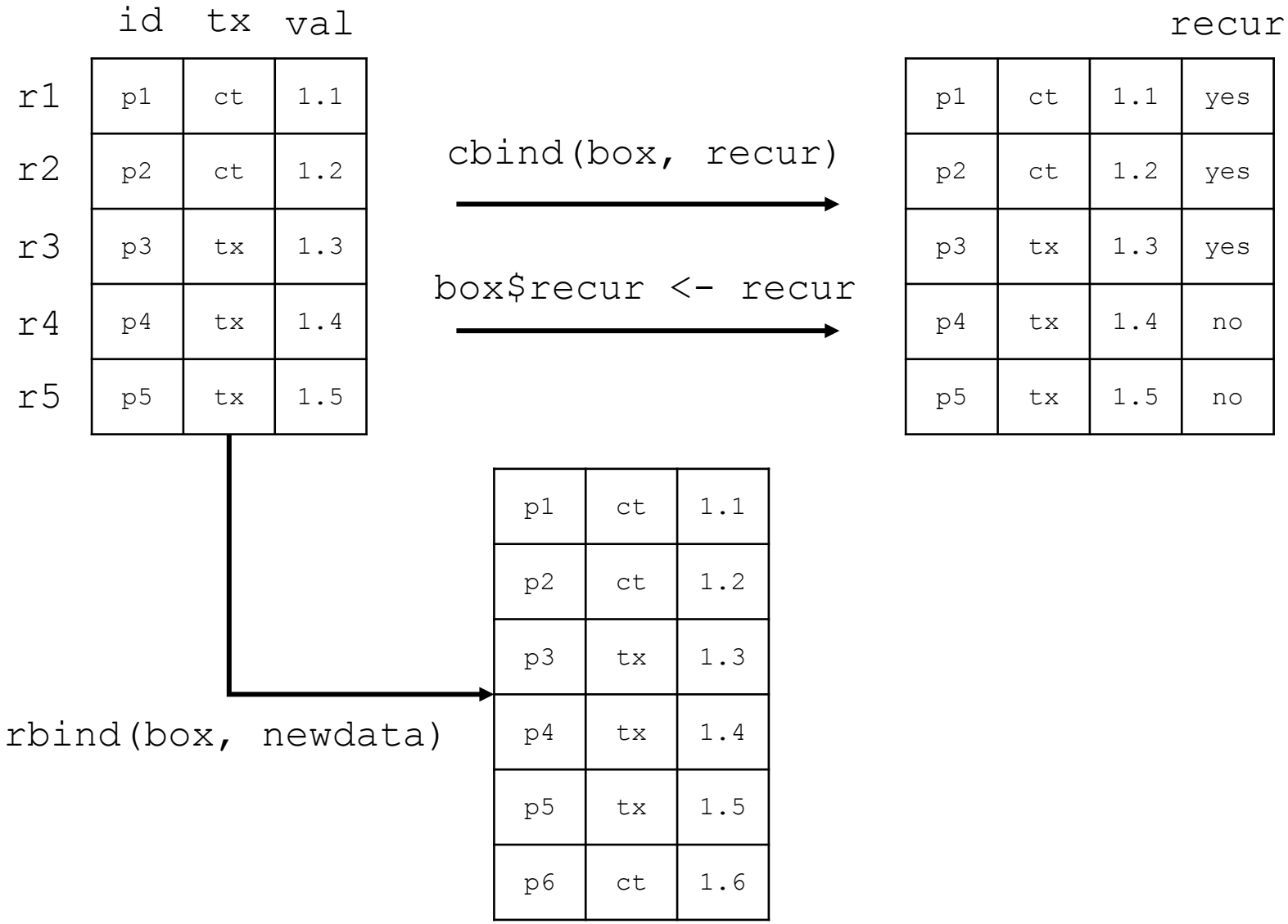
```
box <- data.frame(
  id = c('p1','p2','p3','p4','p5'),
  tx = factor(c('ct','ct','tx','tx','tx')),
  val = c(1.1,1.2,1.3,1.4,1.5),
  stringsAsFactors = FALSE)
row.names(box) <- c('r1','r2','r3','r4','r5')
box['r1',]
#>    id tx val
#> r1 p1 ct 1.1
box[c('r3','r4'),]
#>    id tx val
#> r3 p3 tx 1.3
#> r4 p4 tx 1.4
box$id
#> [1] "p1" "p2" "p3" "p4" "p5"
box[,c('tx','val')]
#>    tx val
#> r1 ct 1.1
#> r2 ct 1.2
#> r3 tx 1.3
#> r4 tx 1.4
#> r5 tx 1.5
box[c('r2','r3'),c('id','tx')]
#>    id tx
#> r2 p2 ct
#> r3 p3 tx
```

Subsetting: using logical vectors



```
box <- data.frame(
  id = c('p1','p2','p3','p4','p5'),
  tx = c('ct','ct','tx','tx','tx'),
  val = c(1.1,1.2,1.3,1.4,1.5),
  stringsAsFactors = FALSE)
row.names(box) <- c('r1','r2','r3','r4','r5')
box
#>    id tx val
#> r1 p1 ct 1.1
#> r2 p2 ct 1.2
#> r3 p3 tx 1.3
#> r4 p4 tx 1.4
#> r5 p5 tx 1.5
idx <- box$tx == 'tx'
idx
#> [1] FALSE FALSE  TRUE  TRUE  TRUE
box[idx,]
#>    id tx val
#> r3 p3 tx 1.3
#> r4 p4 tx 1.4
#> r5 p5 tx 1.5
```

Adding columns



```
box <- data.frame(
  id = c('p1','p2','p3','p4','p5'),
  tx = c('ct','ct','tx','tx','tx'),
  val = c(1.1,1.2,1.3,1.4,1.5),
  stringsAsFactors = FALSE)
row.names(box) <- c('r1','r2','r3','r4','r5')
newbox <- box
recur <- c('yes','yes','yes','no','no')
cbind(box,recur)
#>    id tx val recur
#> r1 p1 ct 1.1   yes
#> r2 p2 ct 1.2   yes
#> r3 p3 tx 1.3   yes
#> r4 p4 tx 1.4    no
#> r5 p5 tx 1.5    no
newbox$recur <- recur
newbox
#>    id tx val recur
#> r1 p1 ct 1.1   yes
#> r2 p2 ct 1.2   yes
#> r3 p3 tx 1.3   yes
#> r4 p4 tx 1.4    no
#> r5 p5 tx 1.5    no
rbind(box, c('p6','ct','1.6'))
#>    id tx val
#> r1 p1 ct 1.1
#> r2 p2 ct 1.2
#> r3 p3 tx 1.3
#> r4 p4 tx 1.4
#> r5 p5 tx 1.5
#> 6  p6 ct 1.6
```

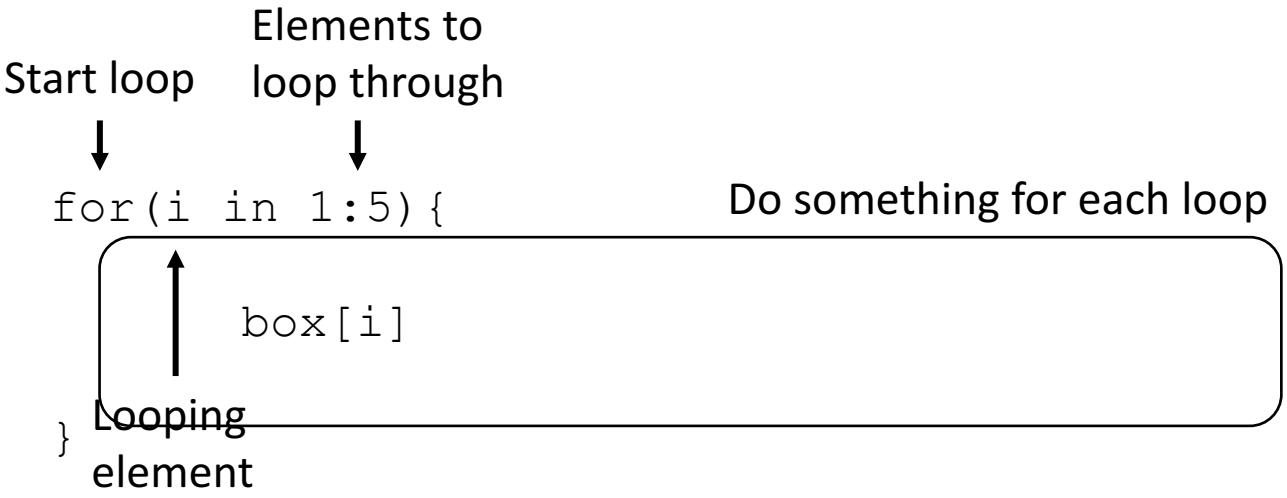
Reading data and showing data

- `read.table` : read a file
- `read.delim` : read a delimited file
- `str` : structure of a data frame
- `summary` : summarise a data frame
- `head` : to show the first few rows of a data frame

```
box <- read.delim('https://raw.githubusercontent.com/sean-cho/datasets/master/datasets/mtcars.txt')
str(box)
#> 'data.frame':    32 obs. of  11 variables:
#>  $ mpg : num  21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
#>  $ cyl : int  6 6 4 6 8 6 8 4 4 6 ...
#>  $ disp: num  160 160 108 258 360 ...
#>  $ hp  : int  110 110 93 110 175 105 245 62 95 123 ...
#>  $ drat: num  3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
#>  $ wt  : num  2.62 2.88 2.32 3.21 3.44 ...
#>  $ qsec: num  16.5 17 18.6 19.4 17 ...
#>  $ vs  : int  0 0 1 1 0 1 0 1 1 1 ...
#>  $ am  : int  1 1 1 0 0 0 0 0 0 0 ...
#>  $ gear: int  4 4 4 3 3 3 3 4 4 4 ...
#>  $ carb: int  4 4 1 1 2 1 4 2 2 4 ...
summary(box)
#>       mpg             cyl             disp             hp       
#>  Min.   :10.40   Min.   :4.000   Min.   : 71.1   Min.   : 52.0  
#>  1st Qu.:15.43   1st Qu.:4.000   1st Qu.:120.8   1st Qu.: 96.5  
#>  Median :19.20   Median :6.000   Median :196.3   Median :123.0  
#>  Mean   :20.09   Mean   :6.188   Mean   :230.7   Mean   :146.7  
#>  3rd Qu.:22.80   3rd Qu.:8.000   3rd Qu.:326.0   3rd Qu.:180.0  
#>  Max.   :33.90   Max.   :8.000   Max.   :472.0   Max.   :335.0  
#>       drat             wt             qsec             vs        
#>  Min.   :2.760   Min.   :1.513   Min.   :14.50   Min.   :0.0000  
#>  1st Qu.:3.080   1st Qu.:2.581   1st Qu.:16.89   1st Qu.:0.0000  
#>  Median :3.695   Median :3.325   Median :17.71   Median :0.0000  
#>  Mean   :3.597   Mean   :3.217   Mean   :17.85   Mean   :0.4375  
#>  3rd Qu.:3.920   3rd Qu.:3.610   3rd Qu.:18.90   3rd Qu.:1.0000  
#>  Max.   :4.930   Max.   :5.424   Max.   :22.90   Max.   :1.0000  
#>        am              gear            carb      
#>  Min.   :0.0000   Min.   :3.000   Min.   :1.000  
#>  1st Qu.:0.0000   1st Qu.:3.000   1st Qu.:2.000  
#>  Median :0.0000   Median :4.000   Median :2.000  
#>  Mean   :0.4062   Mean   :3.688   Mean   :2.812  
#>  3rd Qu.:1.0000   3rd Qu.:4.000   3rd Qu.:4.000  
#>  Max.   :1.0000   Max.   :5.000   Max.   :8.000  
head(box)
#>    mpg cyl disp  hp drat    wt  qsec vs am gear carb
#> 1 21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
#> 2 21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
#> 3 22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
#> 4 21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
#> 5 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
#> 6 18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

For loops

- Algorithms and the same scripts are often reused or applied to a set of elements within an object.
- Loops can be used to accomplish this
- “for” loops is commonly used to analyze a series of elements (from 1:n, or for every element in object)



For loops: Vector

```
## For loops
box <- c('a','b','c','d','e')
for(i in 1:5){
  print(i)
  box[i]
}
```

```
[1] 1
[1] 2
[1] 3
[1] 4
[1] 5
```

```
## For loops
box <- data.frame(matrix(1:15,5,3))
for(i in 1:ncol(box)){
  print(sum(box[,i]))
}
#> [1] 15
#> [1] 40
#> [1] 65
for(i in 1:nrow(box)){
  print(sum(box[i,]))
}
#> [1] 18
#> [1] 21
#> [1] 24
#> [1] 27
#> [1] 30
```

Apply: Vectors

- Apply accomplishes the same purpose as “for” loops
- Faster, different syntax
- sapply : simplify apply (simplifies output to simplest possible form)

```
sapply(object, function)
```

- lapply : list apply (applies function to list and returns list output)

```
lapply(object, function)
```

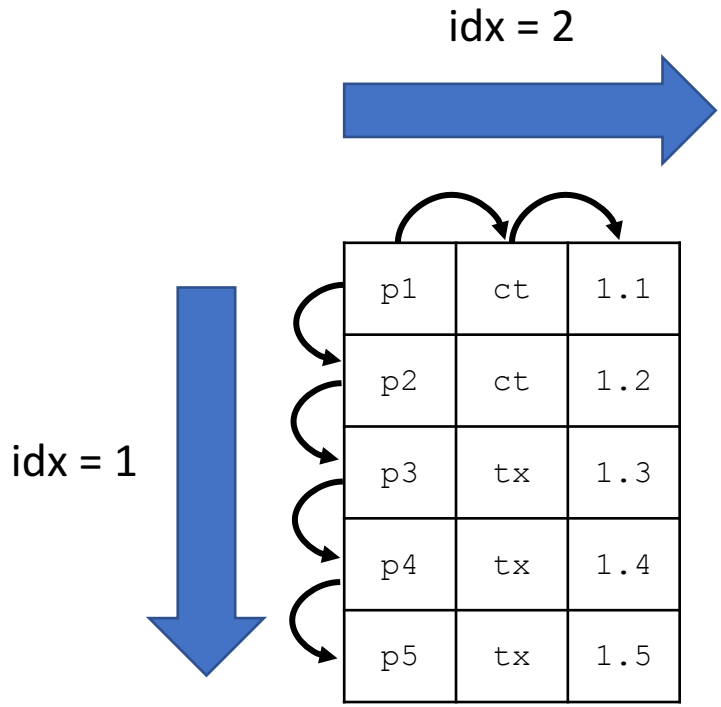
Apply: Vector

```
## For loops
box <- 1:5
sapply(box, function(x) x^x)
```

```
[1]    1    4   27  256 3125
```

Apply: Matrix/Data.frame

```
apply(object, index, function)
```



```r
## For loops
box <- data.frame(matrix(1:15,5,3))
apply(box, 1, sum)
#> [1] 18 21 24 27 30
apply(box, 2, sum)
#> X1 X2 X3
#> 15 40 65
```

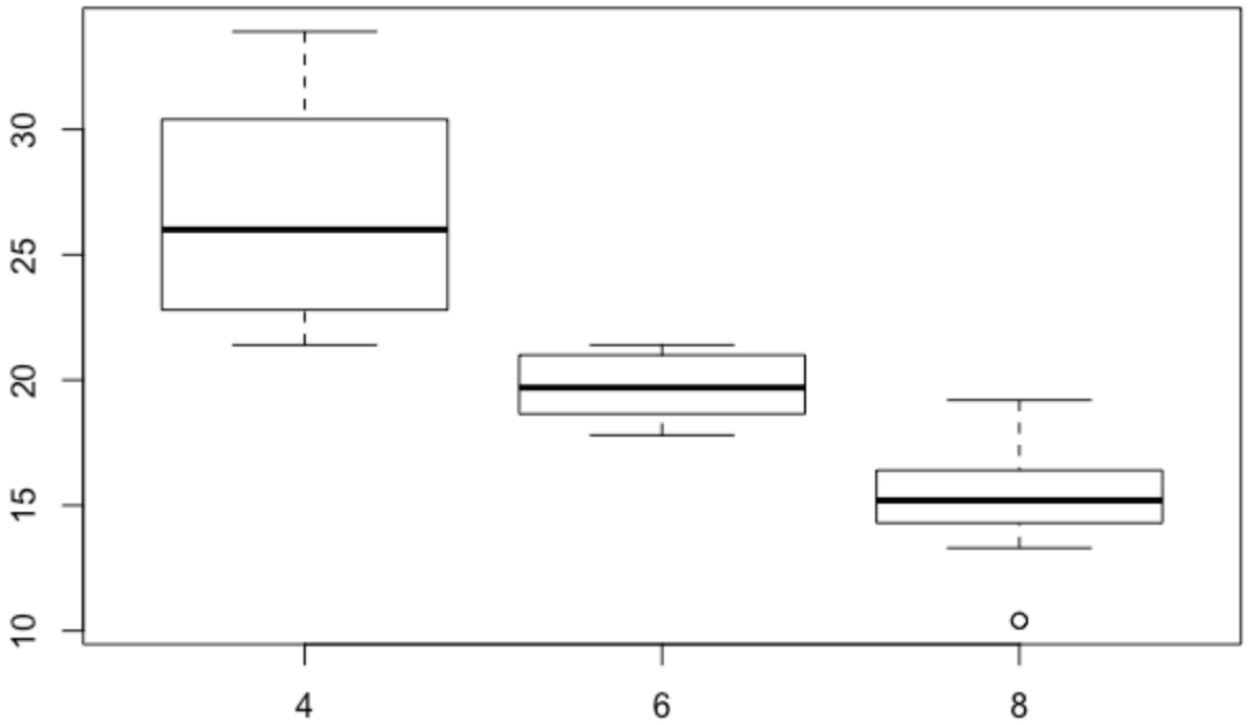
Working with a toy dataset: mtcars

- Look at the structure of a data.frame
- Look at the first few lines of the data.frame
- Make some basic plots
- Add information to plots
- Fit a simple linear model
- Add fitted line to plot

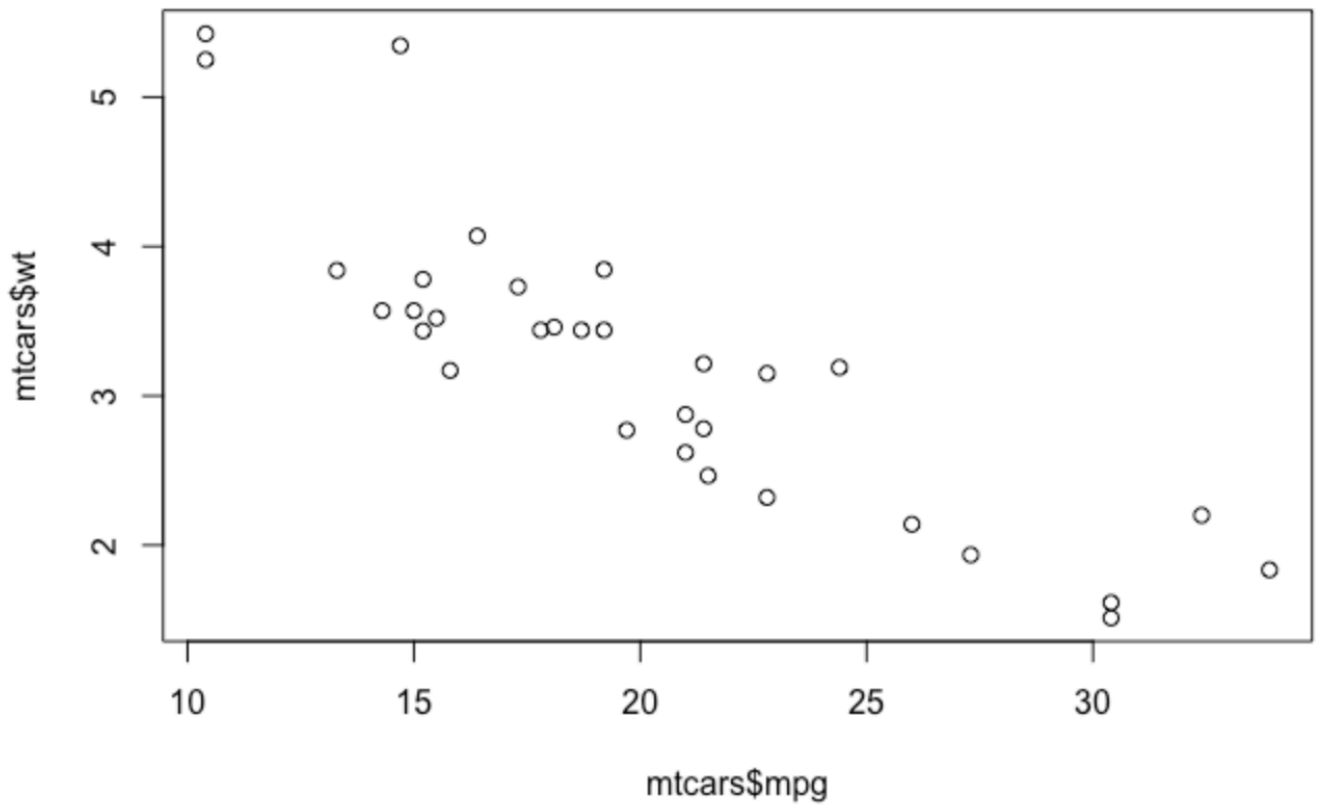
https://github.com/sean-cho/biotrac2018_epigenetics/blob/master/markdown/01_introduction_mtcars.pdf

```
data(mtcars)
str(mtcars)
#> 'data.frame':    32 obs. of  11 variables:
#>  $ mpg : num  21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
#>  $ cyl : num  6 6 4 6 8 6 8 4 4 6 ...
#>  $ disp: num  160 160 108 258 360 ...
#>  $ hp  : num  110 110 93 110 175 105 245 62 95 123 ...
#>  $ drat: num  3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
#>  $ wt  : num  2.62 2.88 2.32 3.21 3.44 ...
#>  $ qsec: num  16.5 17 18.6 19.4 17 ...
#>  $ vs  : num  0 0 1 1 0 1 0 1 1 1 ...
#>  $ am  : num  1 1 1 0 0 0 0 0 0 0 ...
#>  $ gear: num  4 4 4 3 3 3 3 4 4 4 ...
#>  $ carb: num  4 4 1 1 2 1 4 2 2 4 ...
head(mtcars)
#>                   mpg cyl disp  hp drat    wt  qsec vs am gear carb
#> Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
#> Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
#> Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
#> Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
#> Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
#> Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

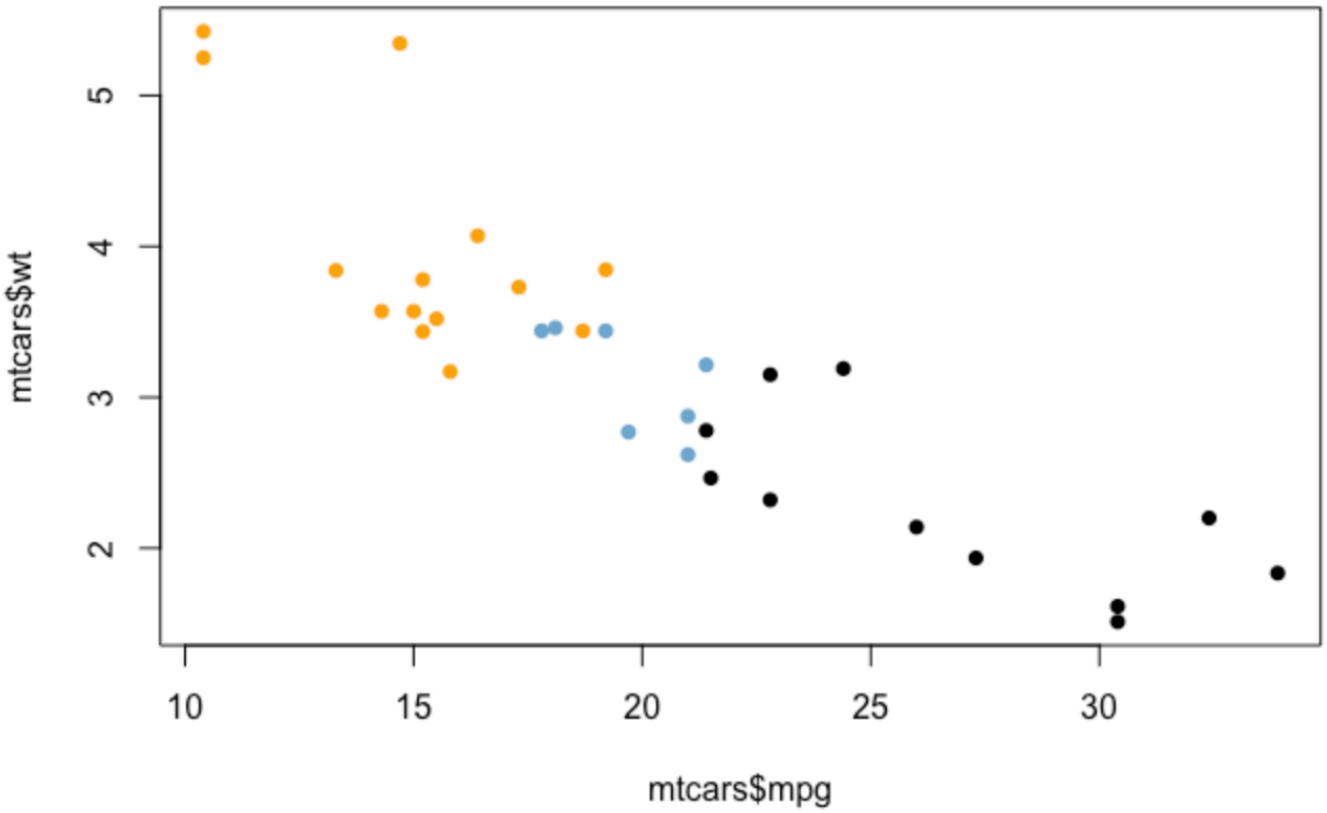
```
boxplot(mtcars$mpg ~ mtcars$cyl)
```



```
plot(mtcars$mpg, mtcars$wt)
```



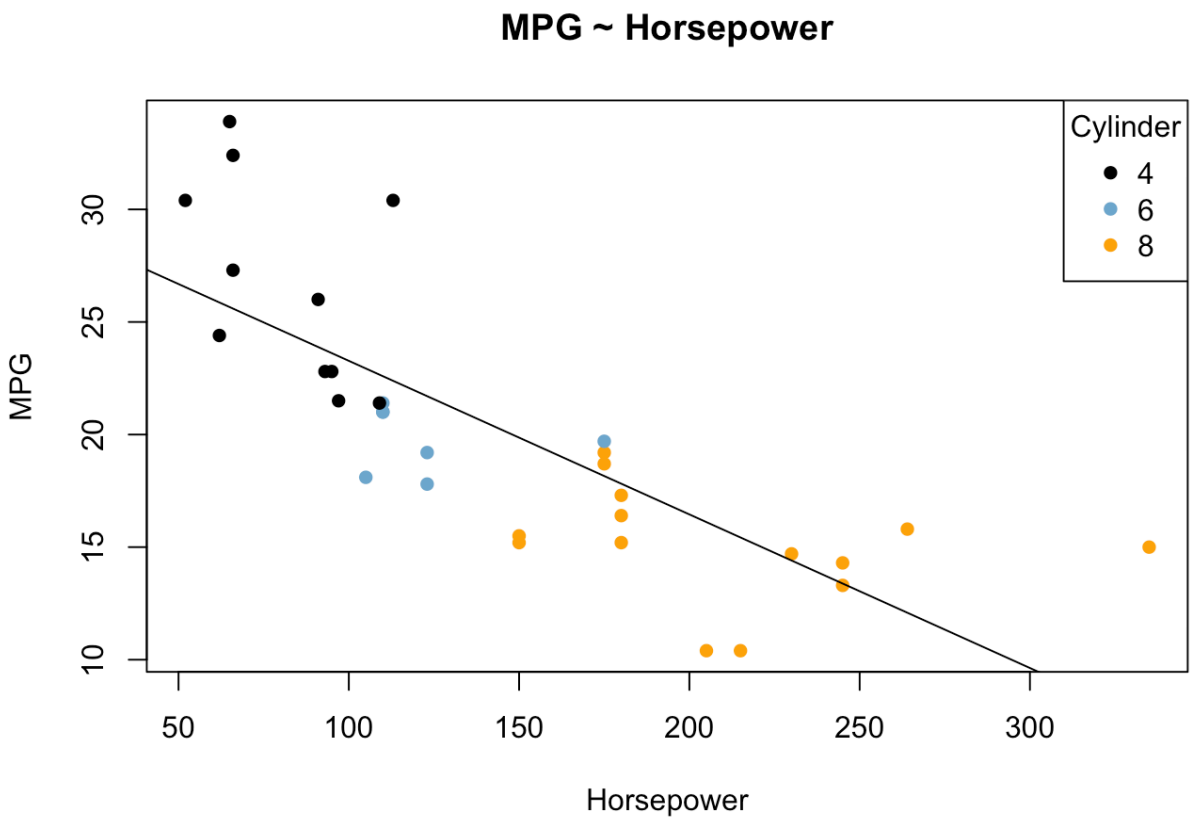
```
mapcolors <- c('4'='black','6'='skyblue3','8'='orange')
mtcolors <- mapcolors[as.character(mtcars$cyl)]

plot(mtcars$mpg, mtcars$wt, col = mtcolors, pch = 16)
```



```
## lm( y ~ x1 + x2 + ... + xn , data = dataset)
summary(lm(mpg ~ hp, data = mtcars))
```

```
##
## Call:
## lm(formula = mpg ~ hp, data = mtcars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.7121 -2.1122 -0.8854  1.5819  8.2360
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 30.09886    1.63392  18.421  < 2e-16 ***
## hp          -0.06823    0.01012  -6.742 1.79e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.863 on 30 degrees of freedom
## Multiple R-squared:  0.6024, Adjusted R-squared:  0.5892
## F-statistic: 45.46 on 1 and 30 DF,  p-value: 1.788e-07
```

```r
## create colors
mapcolors <- c('4'='black','6'='skyblue3','8'='orange')
mtcolors <- mapcolors[as.character(mtcars$cyl)]

## model
mtmodel <- lm(mpg ~ hp, data = mtcars)

## make basic plot
plot(mpg ~ hp, data=mtcars, col = mtcolors, pch = 16, xlab = 'Horsepower', ylab = 'MPG',
     main = 'MPG ~ Horsepower')
## add fitted line
abline(mtmodel)
## add legend
legend('topright', legend = c(4,6,8), col = c('black','skyblue3','orange'), pch = 16, title = 'Cylinder')
```

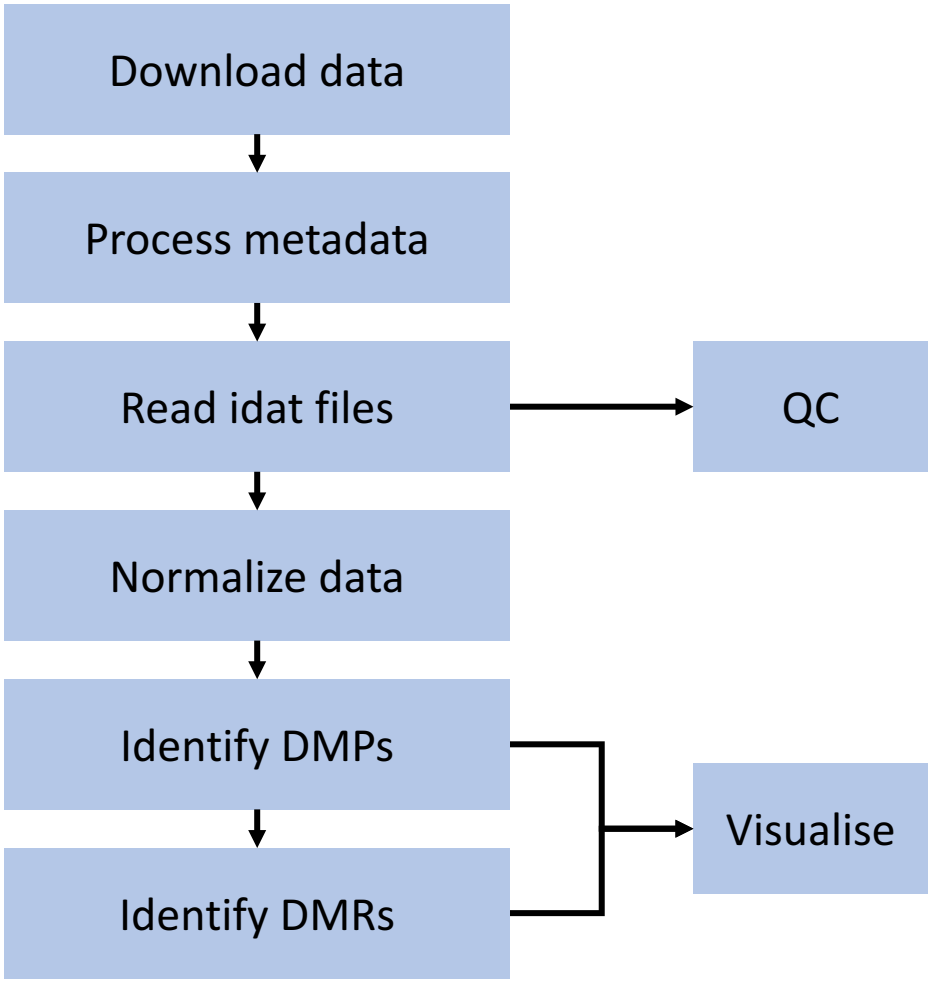


```
summary(lm(mpg ~ vs ,data = mtcars))
#>
#> Call:
#> lm(formula = mpg ~ vs, data = mtcars)
#>
#> Residuals:
#>    Min     1Q Median     3Q    Max
#> -6.757 -3.082 -1.267  2.828  9.383
#>
#> Coefficients:
#>             Estimate Std. Error t value Pr(>|t|)
#> (Intercept)   16.617      1.080  15.390 8.85e-16 ***
#> vs             7.940      1.632   4.864 3.42e-05 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 4.581 on 30 degrees of freedom
#> Multiple R-squared:  0.4409, Adjusted R-squared:  0.4223
#> F-statistic: 23.66 on 1 and 30 DF,  p-value: 3.416e-05
```

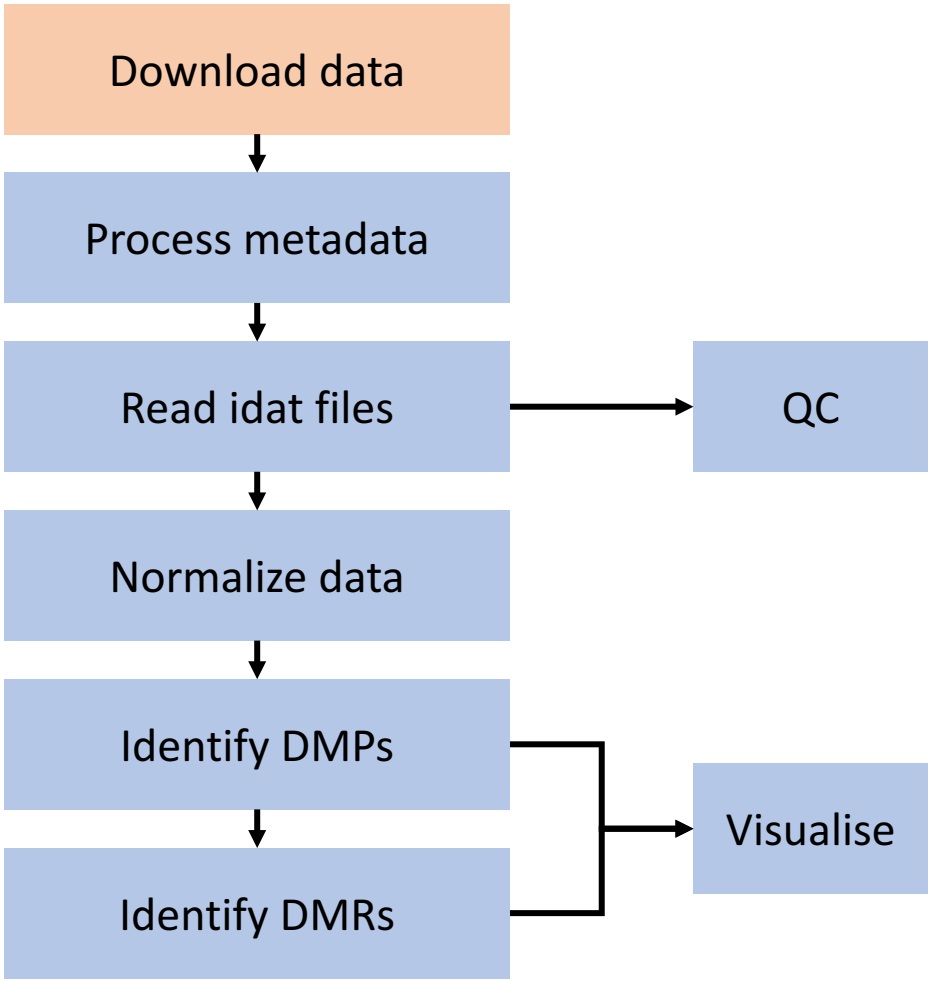
```
summary(lm(mpg ~ vs ,data = mtcars))
#>
#> Call:
#> lm(formula = mpg ~ vs, data = mtcars)
#>
#> Residuals:
#>    Min     1Q Median     3Q    Max
#> -6.757 -3.082 -1.267  2.828  9.383
#>
#> Coefficients:
#>             Estimate Std. Error t value Pr(>|t|)
#> (Intercept)   16.617      1.080  15.390 8.85e-16 ***
#> vs             7.940      1.632   4.864 3.42e-05 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 4.581 on 30 degrees of freedom
#> Multiple R-squared:  0.4409, Adjusted R-squared:  0.4223
#> F-statistic: 23.66 on 1 and 30 DF,  p-value: 3.416e-05
summary(lm(mpg ~ vs + hp + wt ,data = mtcars))
#>
#> Call:
#> lm(formula = mpg ~ vs + hp + wt, data = mtcars)
#>
#> Residuals:
#>     Min      1Q  Median      3Q     Max
#> -3.4667 -1.4857 -0.4296  1.0341  5.7384
#>
#> Coefficients:
#>             Estimate Std. Error t value Pr(>|t|)
#> (Intercept) 35.38267    2.42564  14.587 1.31e-14 ***
#> vs           1.36771    1.35296   1.011   0.3207
#> hp          -0.02542    0.01100  -2.312   0.0284 *
#> wt          -3.78003    0.63985  -5.908 2.35e-06 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 2.592 on 28 degrees of freedom
#> Multiple R-squared:  0.8329, Adjusted R-squared:  0.815
#> F-statistic: 46.52 on 3 and 28 DF,  p-value: 5.276e-11
```

Analysis 1: Squamous cell carcinoma

Squamous cell carcinoma

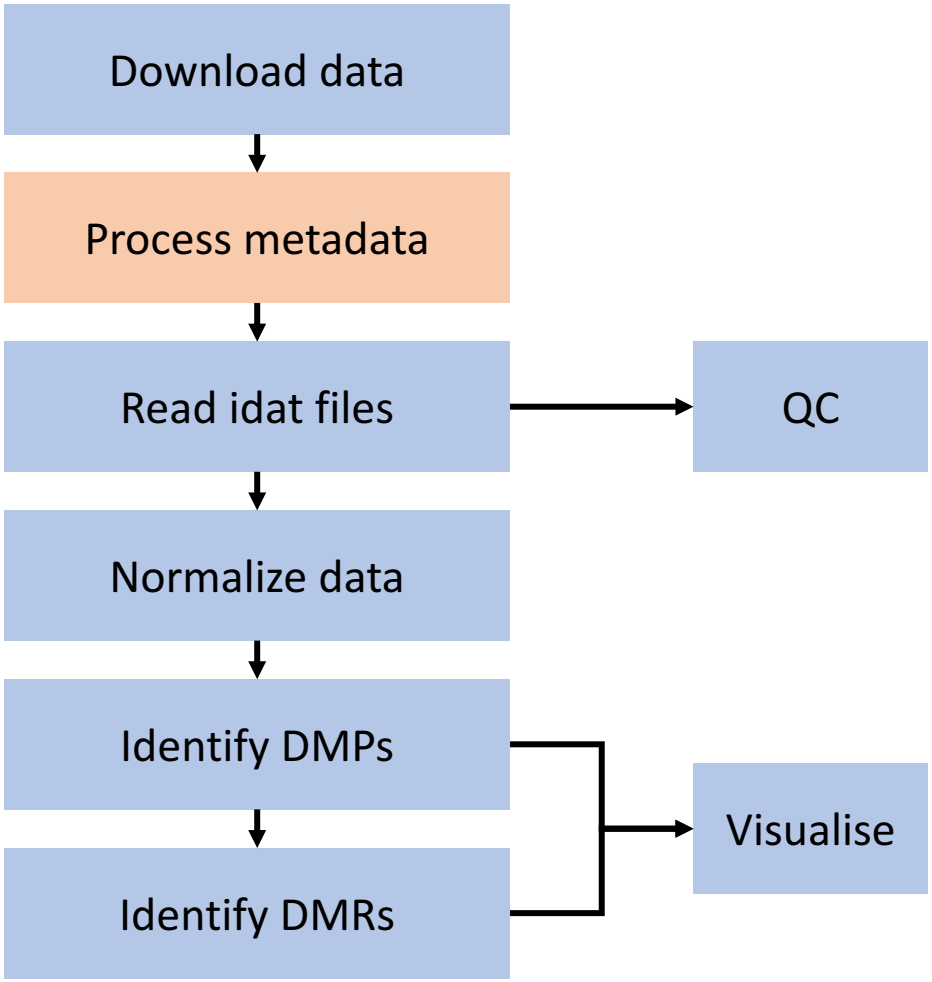


Squamous cell carcinoma



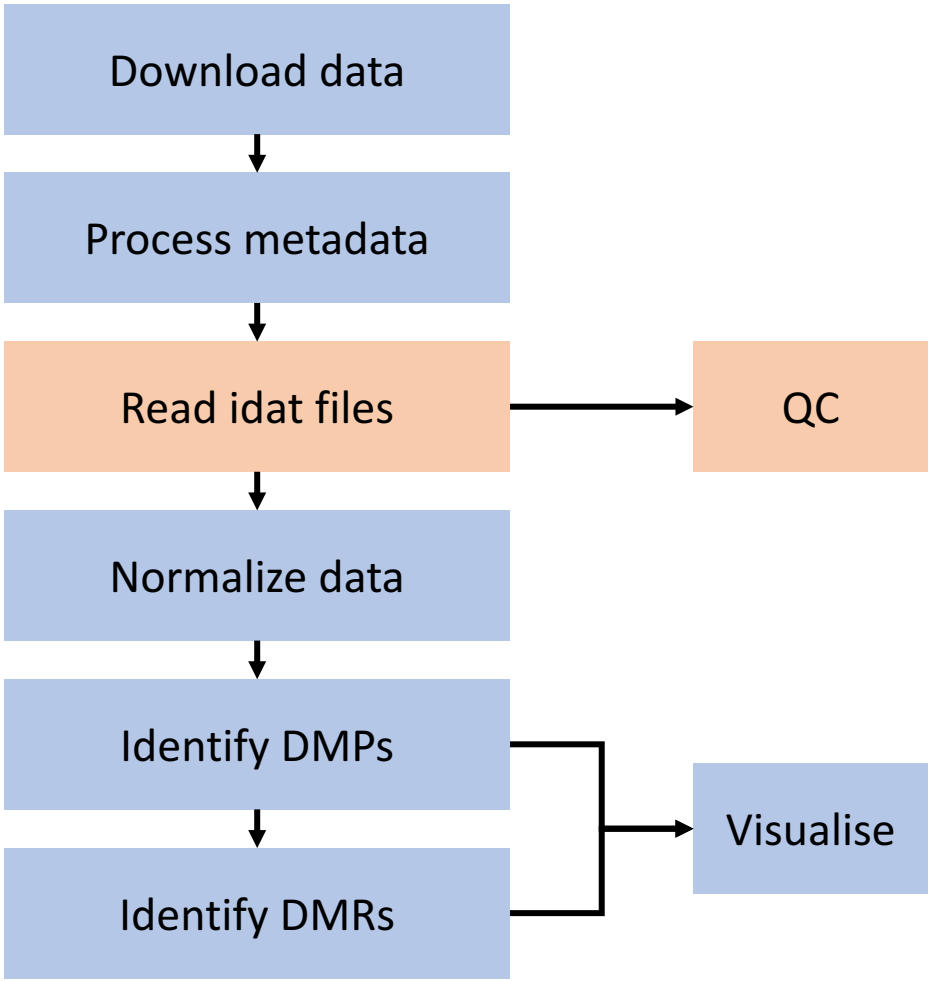
```r
#### set up
library(GEOquery)
library(minfi)
library(limma)
library(gplots)
library(scales)

## functions
movefiles <- function(from){
  to <- gsub('data/scc/GSE67097','data/scc/GSE67097/idat', from)
  file.rename(from, to)
}

## create directories
sapply(c('data','rda','output'), dir.create)

#### get data
## download raw data and move data
dir.create('data/scc', recursive = TRUE) ## create data directory
getGEOSuppFiles('GSE67097', baseDir = 'data/scc') ## download IDAT files
untar(tarfile = 'data/scc/GSE67097/GSE67097_RAW.tar', exdir = 'data/scc/GSE67097') ## untar file
dir.create('data/scc/GSE67097/idat') ## create idat directory to store idat file

idat_files <- list.files('data/scc/GSE67097/', pattern = 'idat.gz', full.names = TRUE)
sapply(idat_files, movefiles) ## move files into idat folder
idat_files <- list.files('data/scc/GSE67097/idat', pattern = 'idat.gz', full.names = TRUE)
sapply(idat_files, gunzip, overwrite = TRUE) ## unzip files

idat_files <- list.files('data/scc/GSE67097/idat', pattern = 'idat$', full.names = TRUE)
```

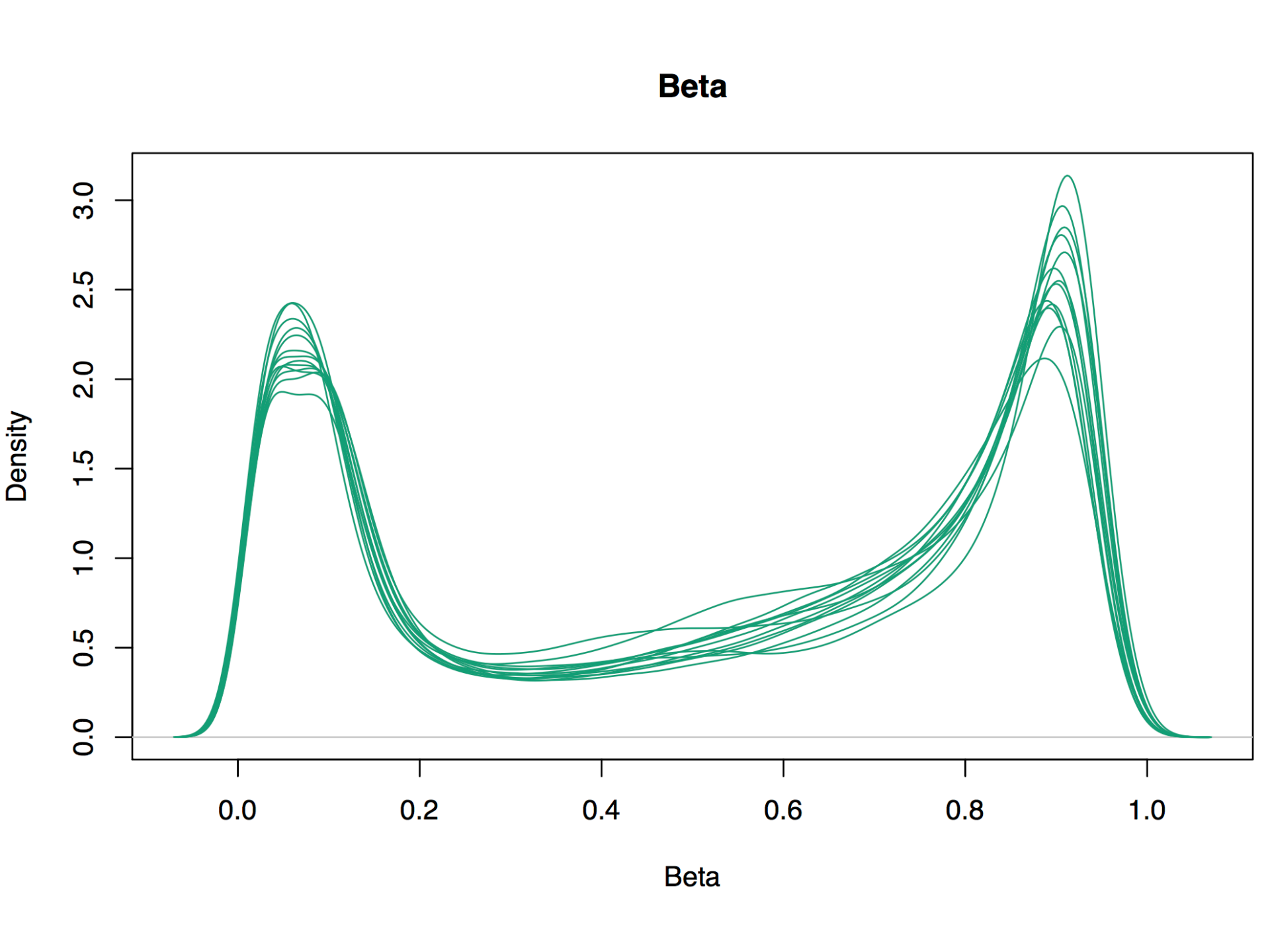
Squamous cell carcinoma

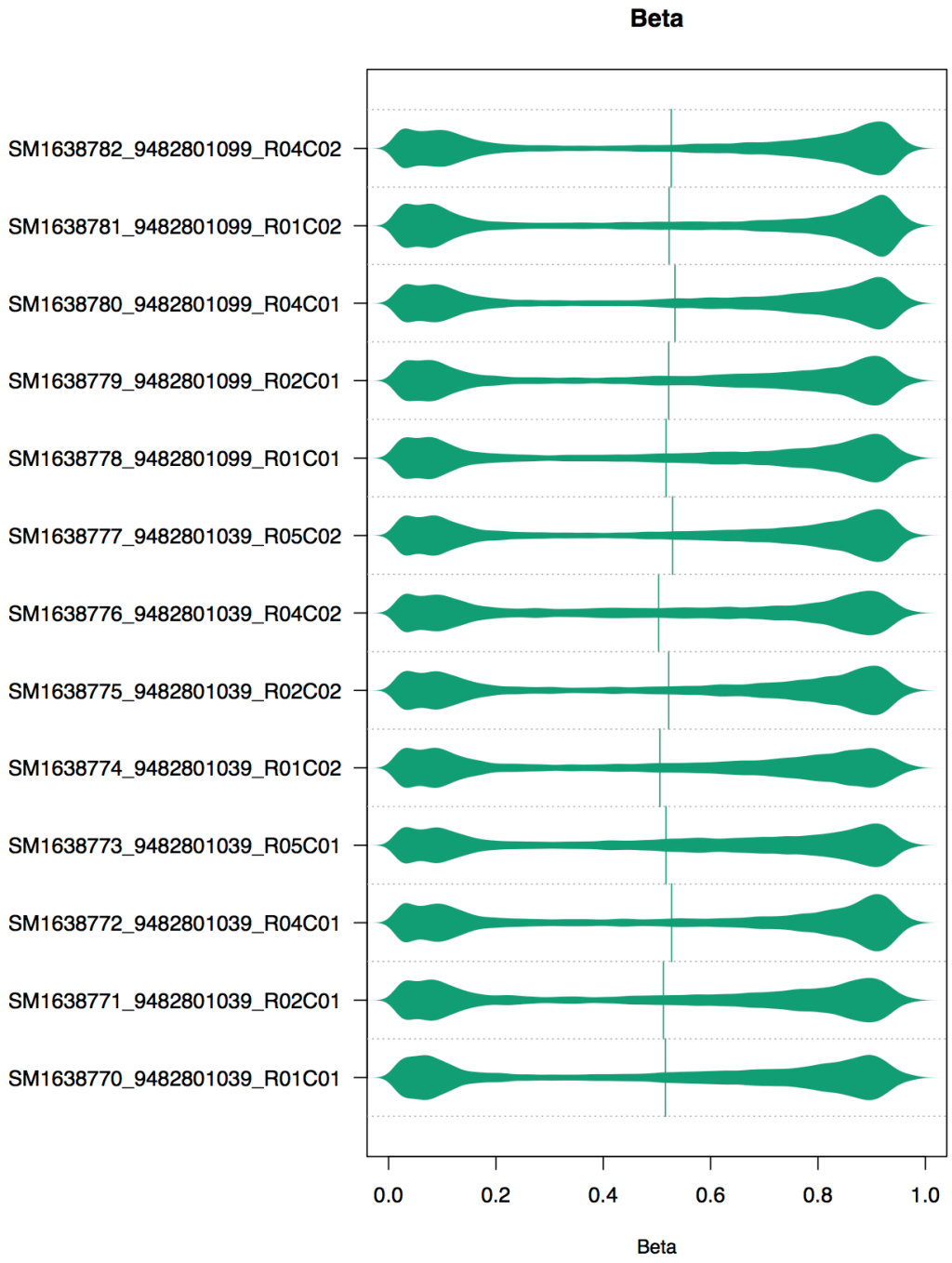


```r
## get phenodata
gse <- getGEO('GSE67097')
rawpheno <- pData(gse[[1]])

## create metadata for reading data using minfi
dl_metadata <- data.frame(do.call(rbind, strsplit(basename(idat_files), split = '_')),
                          stringsAsFactors = FALSE)
colnames(dl_metadata) <- c('Sample_Name', 'Slide', 'Array', 'suffix')
dl_metadata$Basename <- gsub('_[GrnRed]+\\.idat','', idat_files)
dl_metadata$suffix <- NULL
dl_metadata <- unique(dl_metadata)

## annotate metadata
sun <- c('Lip','Neck','Scalp','Arm')
nosun <- c('Calf','Leg')

dl_metadata$OTissue <- rawpheno$`tissue:ch1`
dl_metadata$OGender <- rawpheno$`gender:ch1`
dl_metadata$OBody <- rawpheno$`body site:ch1`
dl_metadata$Tumor <- ifelse(grepl('squamous', rawpheno$`tissue:ch1`), 'SCC', 'normal')
dl_metadata$Sun <- ifelse(dl_metadata$OBody %in% sun, 'sun', 'nosun')
```

Squamous cell carcinoma

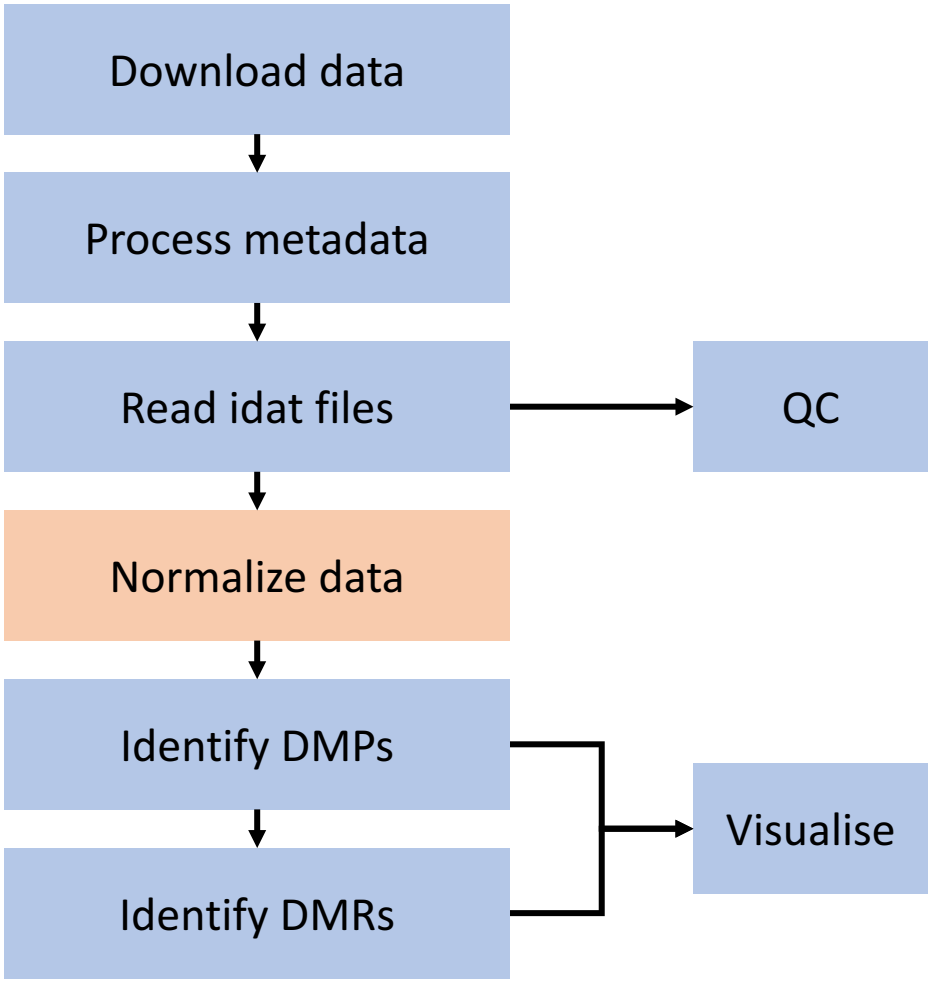



Beta
Density
0.0
0.5
1.0
1.5
2.0
2.5
3.0
0.0
0.2
0.4
0.6
0.8
1.0
Beta

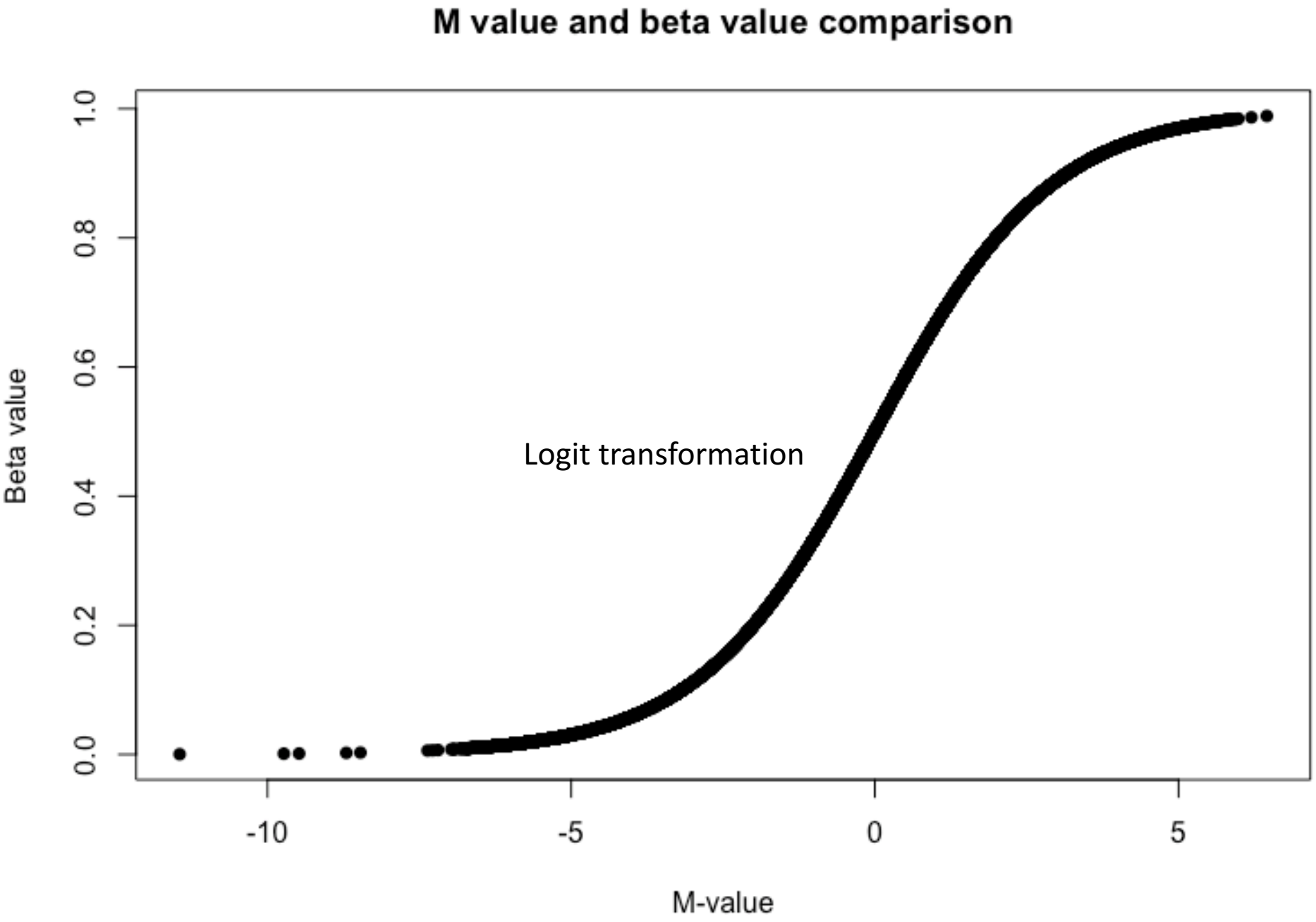

Beta
SM1638782_9482801099_R04C02
SM1638781_9482801099_R01C02
SM1638780_9482801099_R04C01
SM1638779_9482801099_R02C01
SM1638778_9482801099_R01C01
SM1638777_9482801039_R05C02
SM1638776_9482801039_R04C02
SM1638775_9482801039_R02C02
SM1638774_9482801039_R01C02
SM1638773_9482801039_R05C01
SM1638772_9482801039_R04C01
SM1638771_9482801039_R02C01
SM1638770_9482801039_R01C01
0.0
0.2
0.4
0.6
0.8
1.0
Beta

```
#### Load data and process

## read data
rawdata <- read.metharray.exp(targets = dl_metadata)
pheno <- pData(rawdata)
qcReport(rawdata, pdf = 'output/ssc_qc.pdf')
```

Squamous cell carcinoma



```r
## normalization
fndata <- preprocessFunnorm(rawdata)

## beta values
Bdat <- getBeta(fndata)
colnames(Bdat) <- sapply(colnames(Bdat), function(x) strsplit(x, '_')[[1]][1])
bdat <- rmSNPandCH(Bdat, dist = 10)

## M values
Mdat <- getM(fndata)
colnames(Mdat) <- sapply(colnames(Mdat), function(x) strsplit(x, '_')[[1]][1])
Mdat[Mdat==Inf] <- max(Mdat[Mdat!=Inf])
Mdat[Mdat==-Inf] <- min(Mdat[Mdat!=-Inf])

#### Exploratory analysis

## Multidimensional scaling, MDS (essentially PCA)
mdsPlot(mdat, sampGroups = pheno$OGender, pch = 16, numPositions = 5000)
mdsPlot(mdat, sampGroups = pheno$Tumor, pch = 16, numPositions = 5000)
```


M value and beta value comparison
Beta value
0.0
0.2
0.4
0.6
0.8
1.0
Logit transformation
-10
-5
0
5
M-value

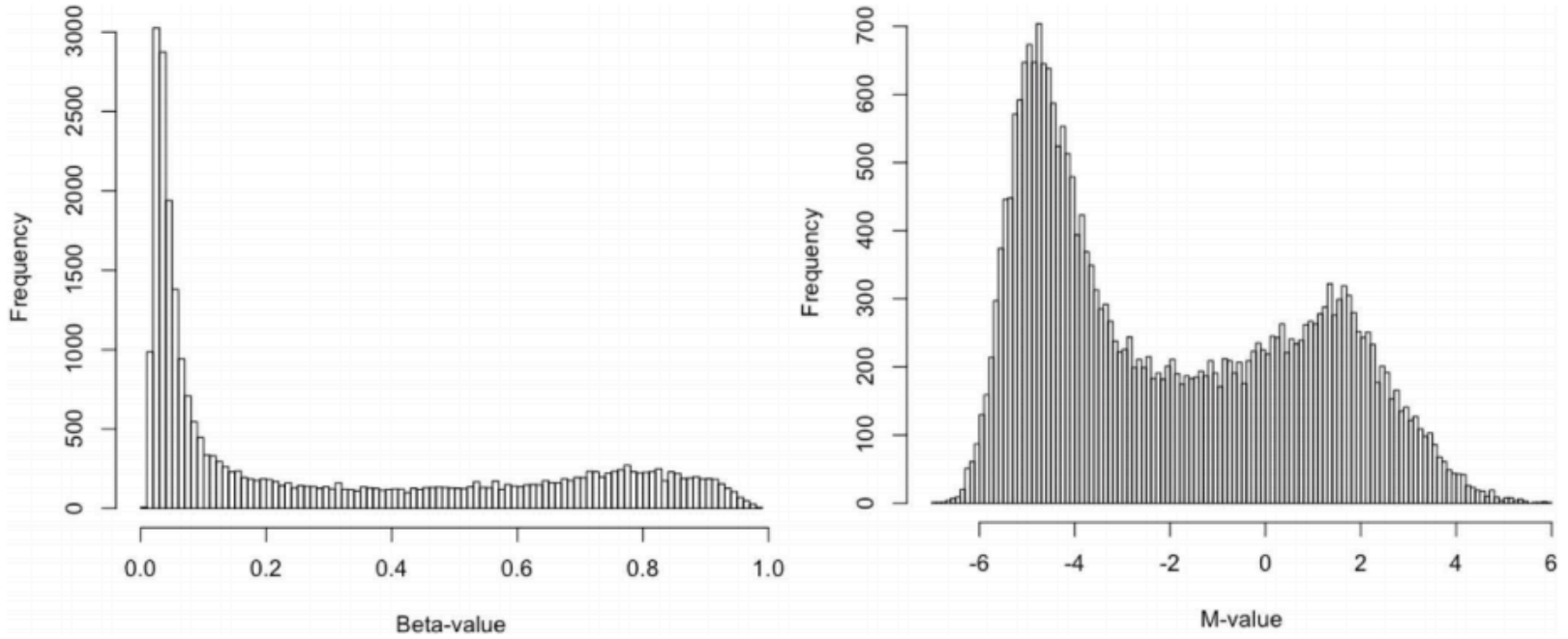


Figure 2 The histograms of Beta-value (left) and M-value (right) (27578 interrogated CpG sites in total).

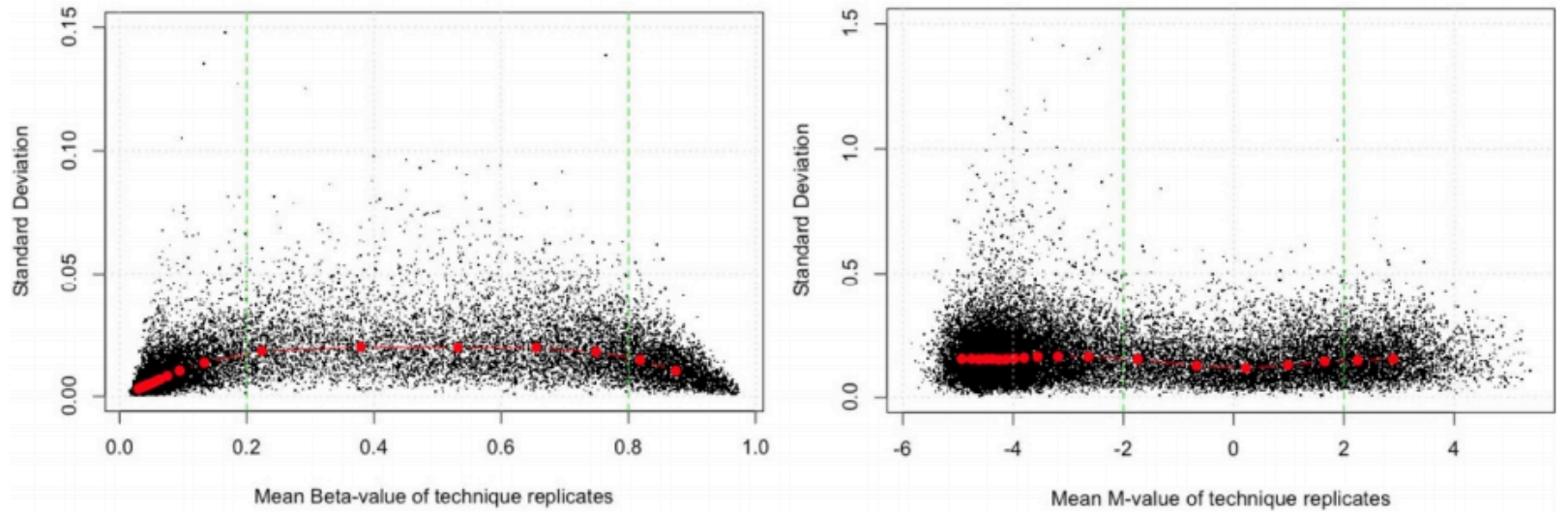
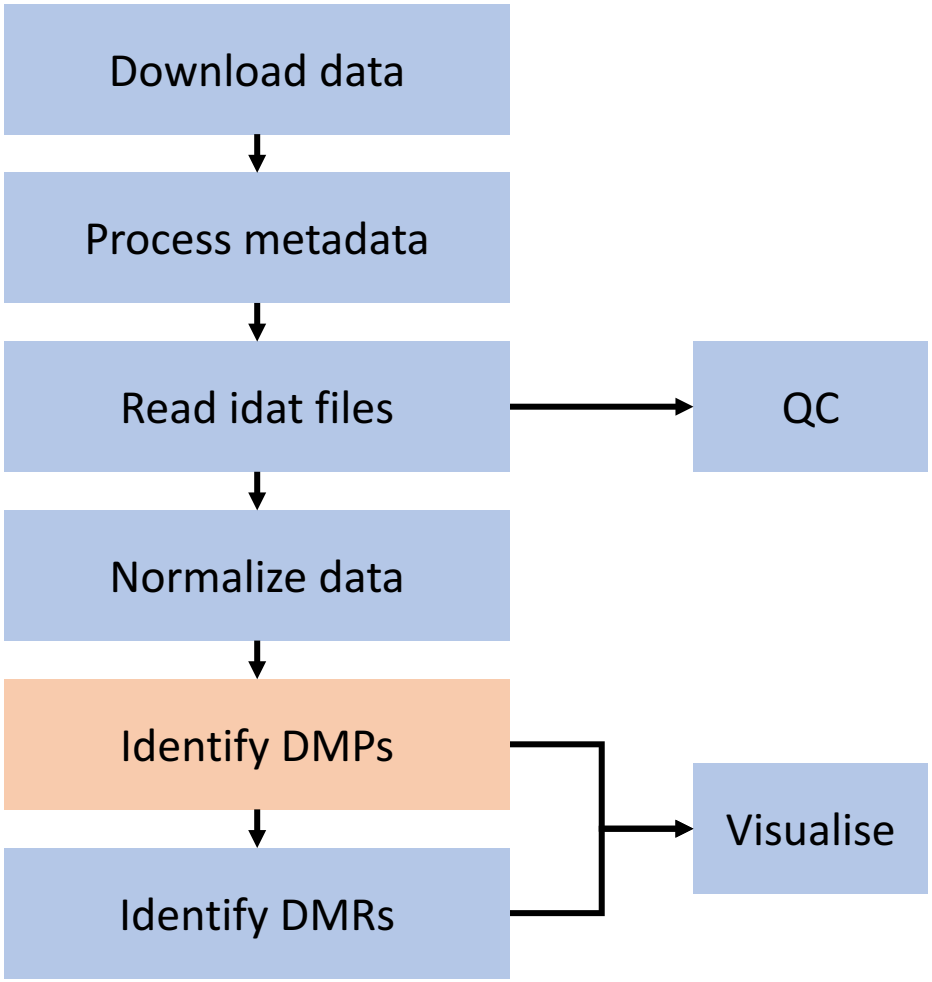


Figure 3 The mean and standard deviation relations of technical replicates. Beta-value (left) and M-value (right).

Squamous cell carcinoma



Goal: Identify differentially methylated probe/site

Linear model

Observed signal for probe p

Coefficients for probe p

$$E(Y_p) = X\beta_p$$

Design matrix

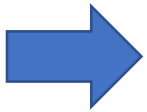
$$\sim \beta_D + \beta_1 + \cdots + \beta_n + \varepsilon$$

Identifier	Disease	Gender	…	HRT
Case01	1	1	…	1
Case02	1	0	…	0
Ctrl01	0	1	…	0
Ctrl02	0	0	…	1

Create model matrix

```
sccmodel <- model.matrix(~ 0 + pheno$Tumor + pheno$OGender)
colnames(sccmodel) <- c('normal','scc','gender')
```

```
> pheno[,c('Tumor','OGender')]
DataFrame with 13 rows and 2 columns
                                    Tumor     OGender
                                <character> <character>
GSM1638770_9482801039_R01C01          SCC      female
GSM1638771_9482801039_R02C01       normal      female
GSM1638772_9482801039_R04C01          SCC      female
GSM1638773_9482801039_R05C01          SCC      female
GSM1638774_9482801039_R01C02          SCC        male
...                                   ...         ...
GSM1638778_9482801099_R01C01          SCC      female
GSM1638779_9482801099_R02C01       normal      female
GSM1638780_9482801099_R04C01       normal      female
GSM1638781_9482801099_R01C02          SCC        male
GSM1638782_9482801099_R04C02       normal        male
```



```
> sccmodel
   normal scc gender
1       0   1      0
2       1   0      0
3       0   1      0
4       0   1      0
5       0   1      1
6       1   0      0
7       0   1      0
8       1   0      0
9       0   1      0
10      1   0      0
11      1   0      0
12      0   1      1
13      1   0      1
attr(,"assign")
[1] 1 1 2
attr(,"contrasts")
attr(,"contrasts")$`pheno$Tumor`
[1] "contr.treatment"

attr(,"contrasts")$`pheno$OGender`
[1] "contr.treatment"
```

Create contrasts

```
scc_contrasts <- makeContrasts(scc - normal, levels = sccmodel)
```

```
> sccmodel
   normal scc gender
1       0   1      0
2       1   0      0
3       0   1      0
4       0   1      0
5       0   1      1
6       1   0      0
7       0   1      0
8       1   0      0
9       0   1      0
10      1   0      0
11      1   0      0
12      0   1      1
13      1   0      1
attr(,"assign")
[1] 1 1 2
attr(,"contrasts")
attr(,"contrasts")$`pheno$Tumor`
[1] "contr.treatment"

attr(,"contrasts")$`pheno$OGender`
[1] "contr.treatment"
```

```
> scc_contrasts
        Contrasts
Levels   scc - normal
  normal           -1
  scc               1
  gender            0
```

Identify DMPs

```
sccfit1 <- lmFit(mdat, design = sccmodel)
sccfit2 <- contrasts.fit(sccfit1, scc_contrasts)
sccfiteb <- eBayes(sccfit2)
scc_res <- topTable(sccfiteb, coef = 1, number = nrow(mdat))
```

Log fold-change

t-statistic

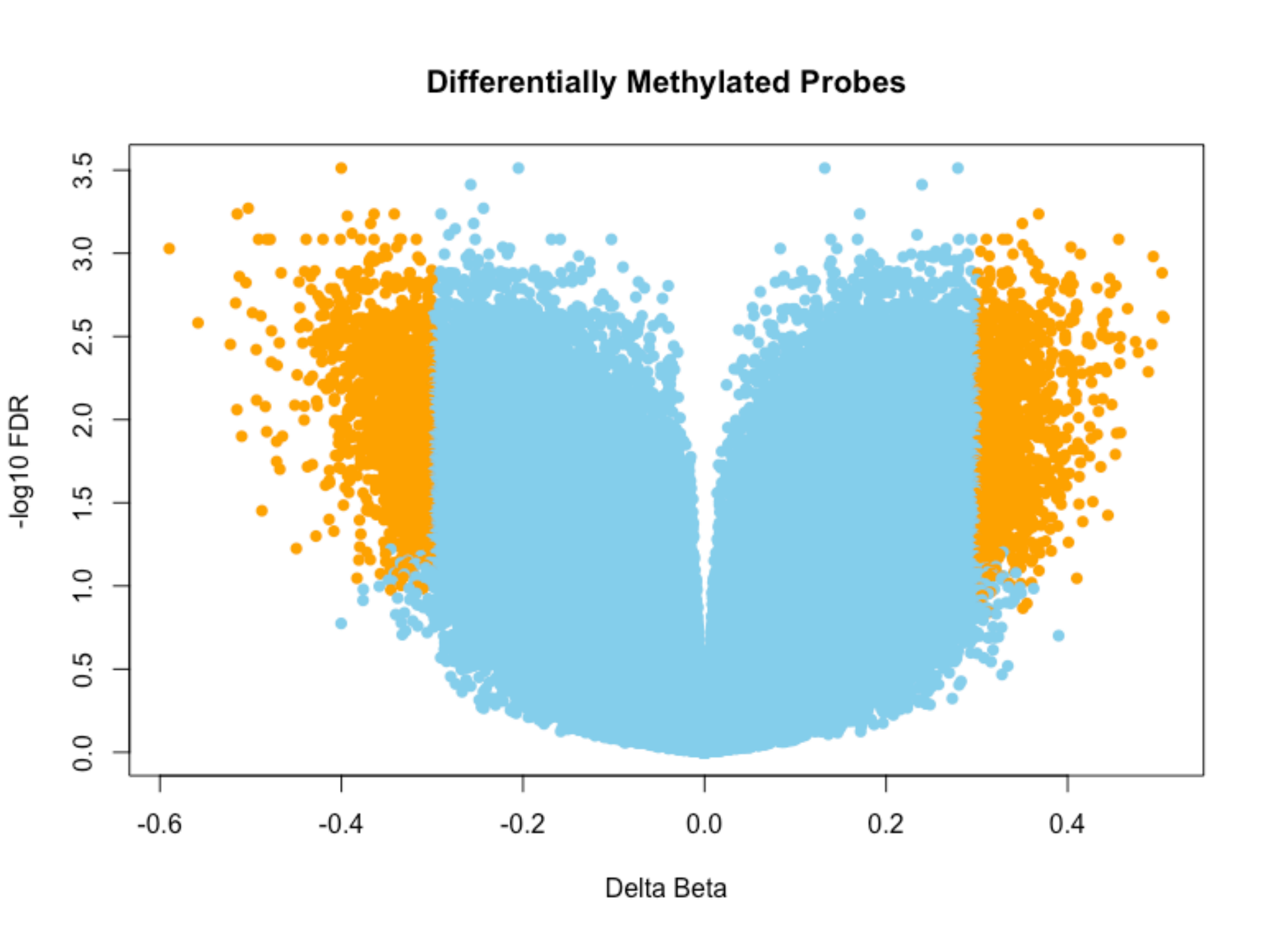
p-value

FDR

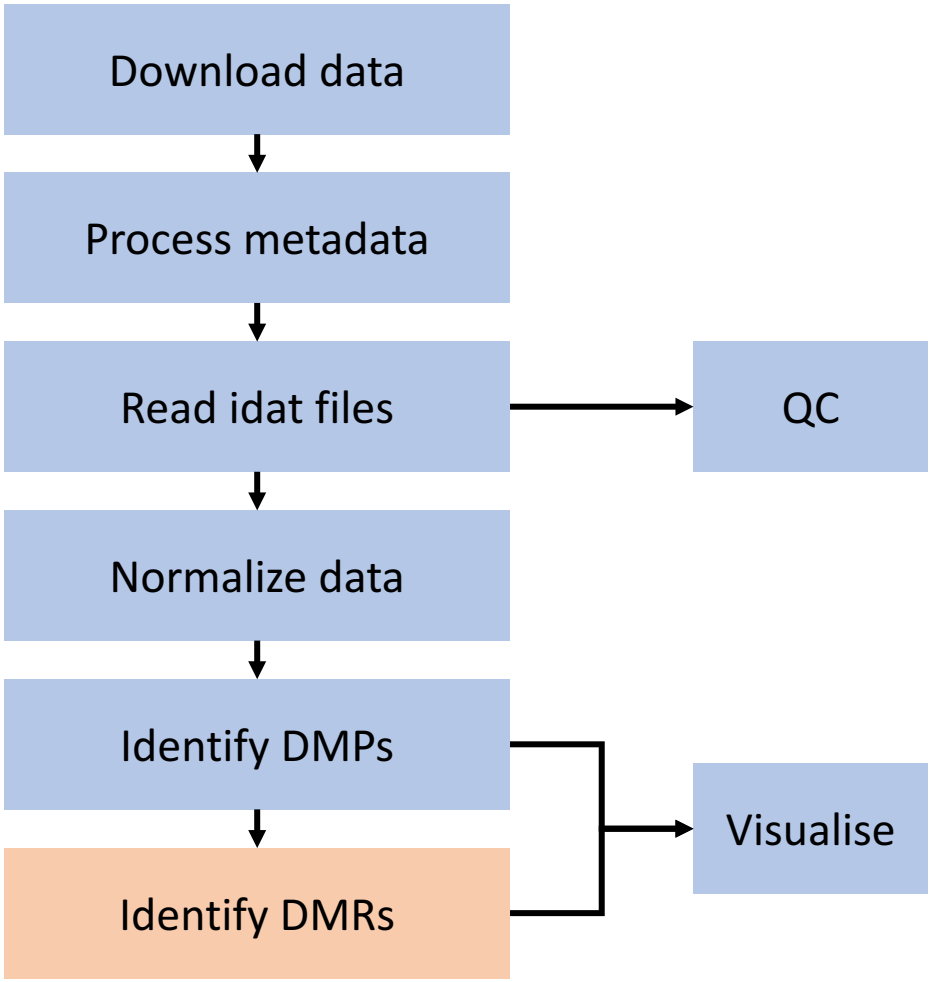
```
> head(scc_res)
               logFC   AveExpr         t      P.Value    adj.P.Val        B
cg21076680 -2.855638 -1.627467 -16.53975 6.041474e-10 0.0002015830 12.04357
cg17094249 -3.216921 -1.737327 -16.11129 8.303937e-10 0.0002015830 11.81889
cg13427361 -3.504238 -1.651158 -14.67288 2.563541e-09 0.0004148766 10.99260
cg02017450 -2.910634 -2.908526 -14.03728 4.355809e-09 0.0004497198 10.58844
cg03699566  2.057571  1.474032  13.59037 6.405636e-09 0.0004497198 10.28844
cg06033579  3.367532  2.869987  13.49918 6.939602e-09 0.0004497198 10.22555
```

Average value across all samples

B-statistic

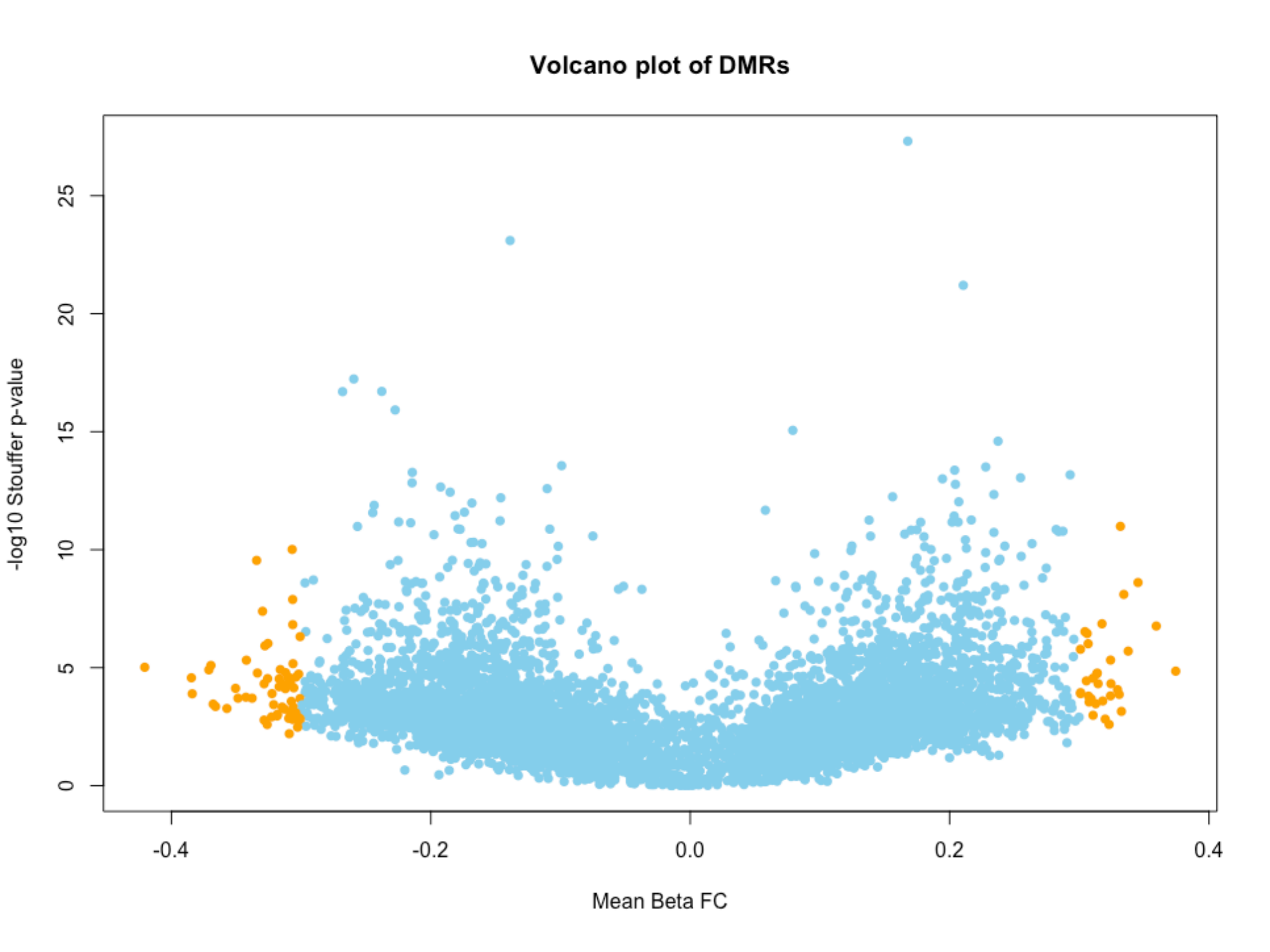

Differentially Methylated Probes
-log10 FDR
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
-0.6
-0.4
-0.2
0.0
0.2
0.4
Delta Beta

Squamous cell carcinoma

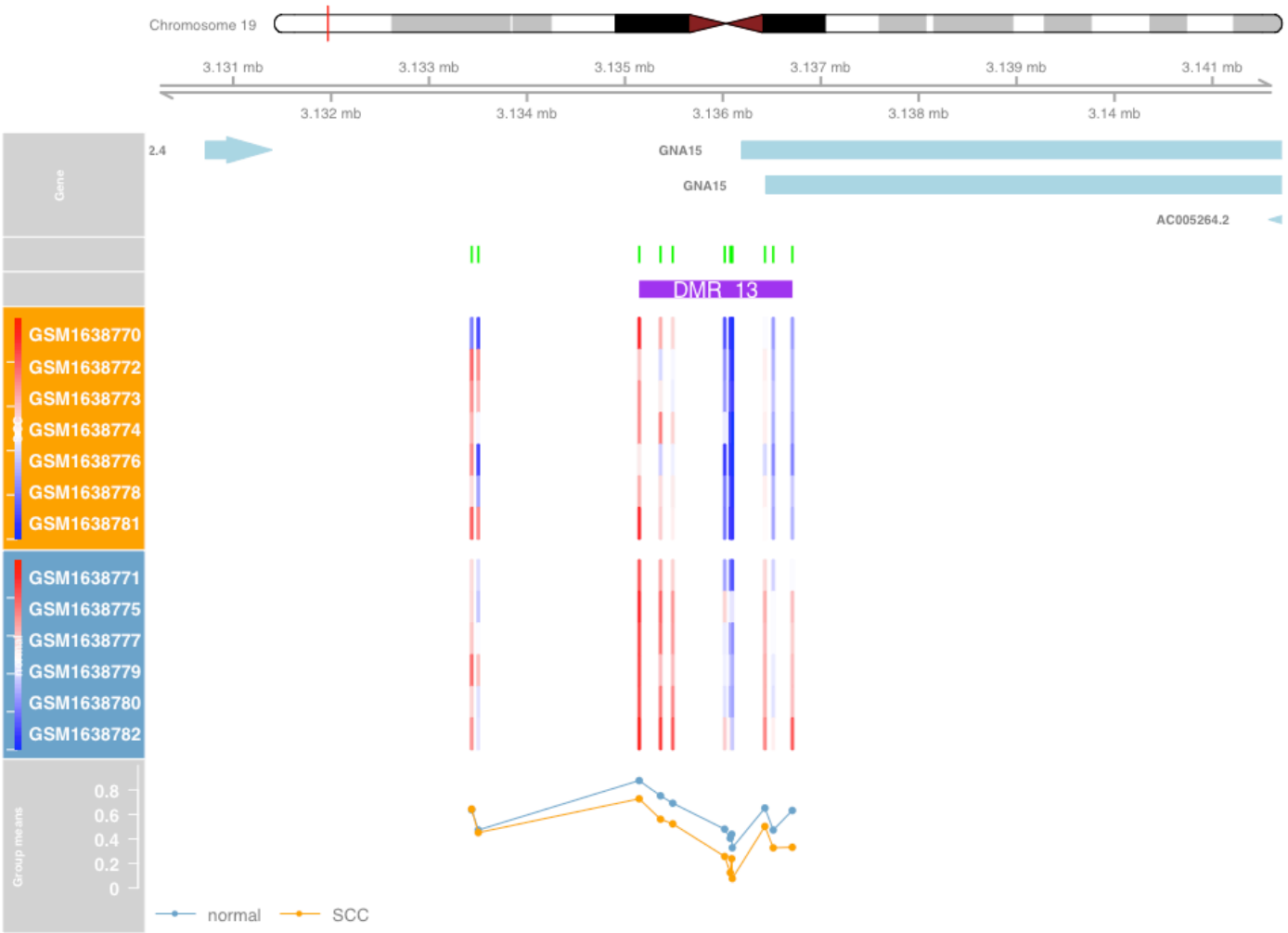


Identify DMRs

```
#### DMR analysis
## annotate
scc_dmrdat <- cpg.annotate(object = mdat, datatype = 'array',
                           what = 'M', design = sccmodel,
                           arraytype = '450K', contrasts = TRUE,
                           cont.matrix = scc_contrasts,
                           coef = 'scc - normal')
# contrasts
scc_dmr <- dmrcate(scc_dmrdat)
scc_dmranges <- extractRanges(scc_dmr, genome = 'hg19')
plot(x = scc_dmranges$meanbetafc, y = -log10(scc_dmranges$Stouffer),
     col = ifelse(abs(scc_dmranges$meanbetafc) >= 0.3 &
                    scc_dmranges$Stouffer < 0.01,'orange','skyblue'),
     pch = 16)
```

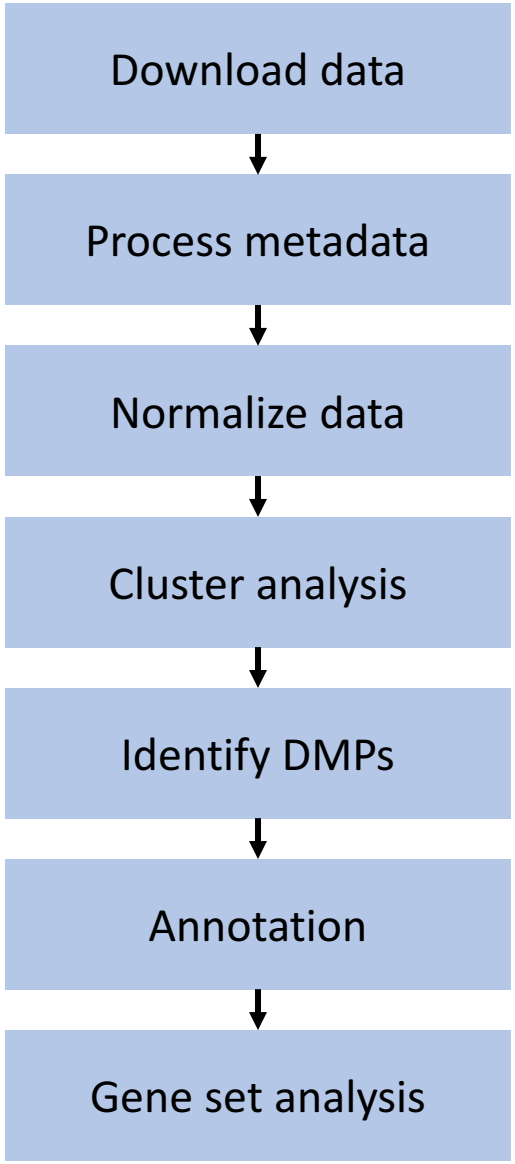

Volcano plot of DMRs
-log10 Stouffer p-value
0
5
10
15
20
25
-0.4
-0.2
0.0
0.2
0.4
Mean Beta FC

```r
## create colors
colgrp <- c('SCC' = 'orange', 'normal' = 'skyblue3')[pheno$Tumor]

DMR.plot(ranges = br, dmr = 13, CpGs=bdat,
         phen.col =  colgrp,
         what = 'Beta', arraytype = '450K', pch = 16, plotmedians = TRUE, genome='hg19',
         samps = 1:ncol(bdat), toscale = TRUE)
```



Analysis 2: Breast cancer

Breast cancer



Breast cancer dataset: GSE20713

EMBO
Molecular Medicine

Research Article

Epigenetic portraits of human breast cancers

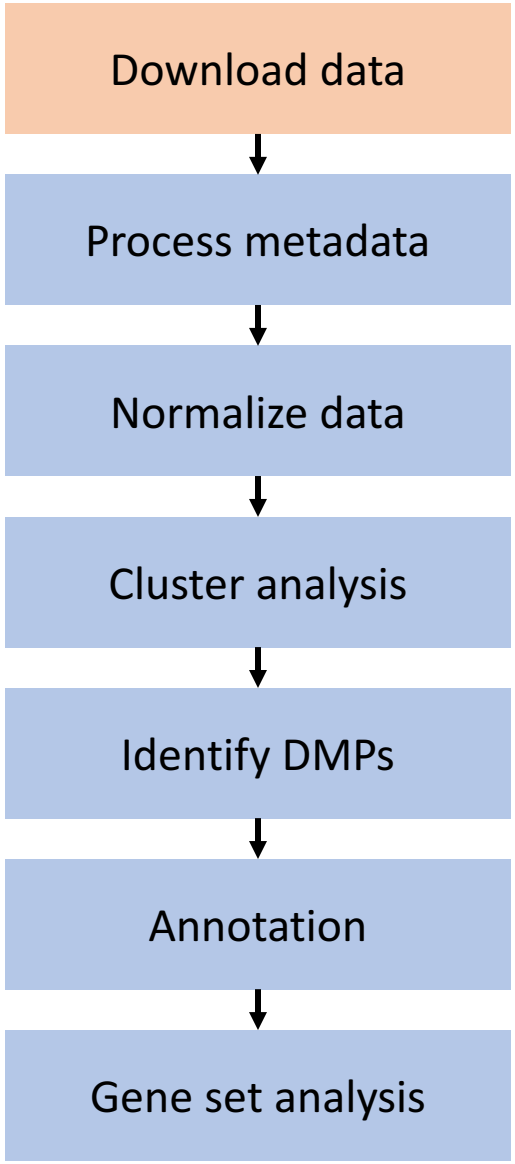
DNA methylation profiling reveals a predominant immune component in breast cancers

*Sarah Dedeurwaerder[1†], Christine Desmedt[2†], Emilie Calonne[1], Sandeep K. Singhal[2], Benjamin Haibe-Kains[2,3], Matthieu Defrance[1], Stefan Michiels[2], Michael Volkmar[1], Rachel Deplus[1], Judith Luciani[1], Françoise Lallemand[2], Denis Larsimont[4], Jérôme Toussaint[2], Sandy Haussy[2], Françoise Rothé[2], Ghizlane Rouas[2], Otto Metzger[2], Samira Majjaj[2], Kamal Saini[2], Pascale Putmans[1], Gérald Hames[5], Nicolas van Baren[6], Pierre G. Coulie[5], Martine Piccart[7], Christos Sotiriou[2**,†], François Fuks[1*,†]*

Data set

- Two datasets:
  - Methylation microarray
  - Expression microarray
- Methylation microarray
  - 27K data
  - 273 samples
    - 236 tumors
    - 12 normals
    - 25 cells lines
- Expression microarray
  - 108 samples, overlap with methylation data

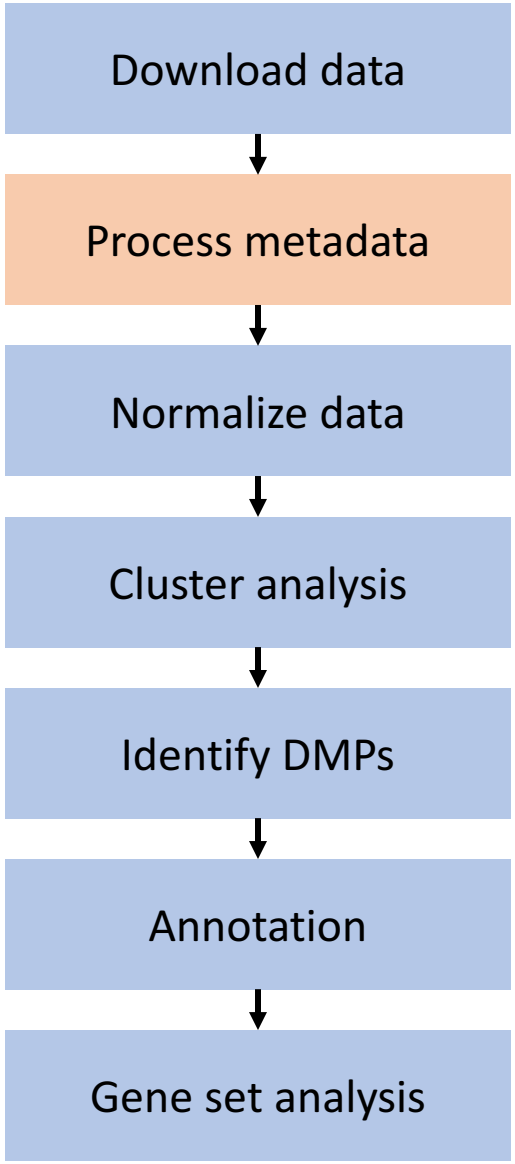
Breast cancer



Data set

- Two datasets:
  - Methylation microarray
  - Expression microarray
- Methylation microarray
  - 27K data
  - 273 samples
    - 236 tumors
    - 12 normals
    - 25 cells lines
- Expression microarray
  - 108 samples, overlap with methylation data

Breast cancer



What is in pData?

```
str(pData(breast_meth), vec.len = 2)
```

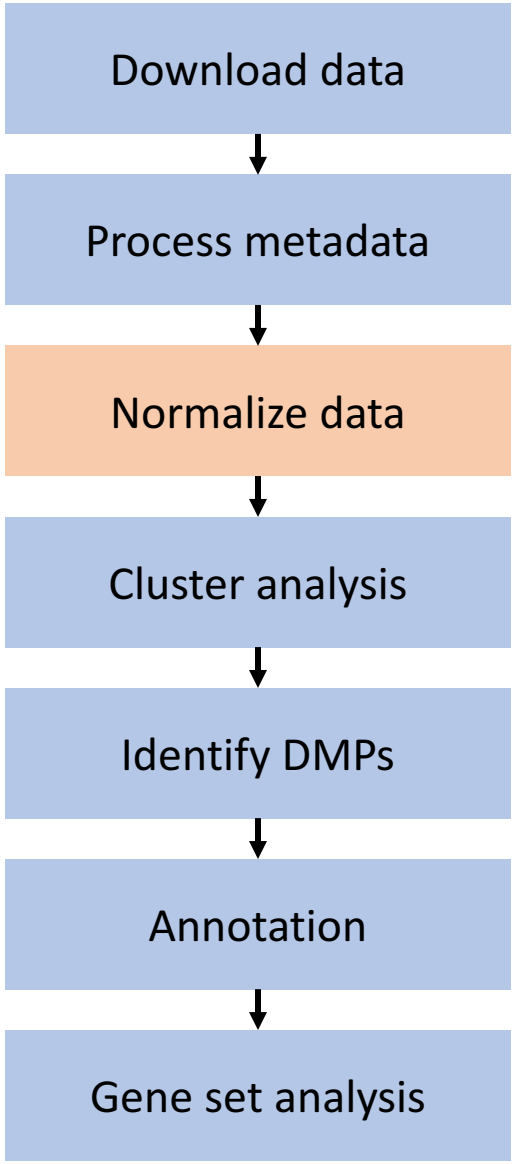
```
## 'data.frame':    273 obs. of  77 variables:
##  $ title                 : Factor w/ 273 levels "Breast cancer cell line BT20 5 AZA (methylation data)",..: 1
6 17 18 19 20 ...
##  $ geo_accession         : chr  "GSM519817" "GSM519818" ...
##  $ status                : Factor w/ 1 level "Public on Oct 19 2011": 1 1 1 1 1 ...
##  $ submission_date       : Factor w/ 4 levels "Jun 09 2010",..: 3 3 3 3 3 ...
##  $ last_update_date      : Factor w/ 1 level "Oct 19 2011": 1 1 1 1 1 ...
##  $ type                  : Factor w/ 1 level "genomic": 1 1 1 1 1 ...
##  $ channel_count         : Factor w/ 1 level "1": 1 1 1 1 1 ...
##  $ source_name_ch1       : Factor w/ 9 levels "Breast cancer cell line",..: 2 2 2 2 2 ...
```

- Rows are samples and columns are phenotype data
- 4 batches as identified by the submission data
- Repeated measures where GEO has conveniently parsed the data for us
  - characteristics_ch1.6 and agebin:ch1
- Same feature in separate fields
  - agebin:ch1 and age_bin:ch1
- **Wrangle**

Wrangle

```
#### Wrangle phenodata
bmpheno <- data.frame(lapply(pData(breast_meth), factor2char), stringsAsFactors = FALSE)

bmpheno$Samplename <- gsub('.*(?:patient|line) (.*)(?: \\(meth.*|$)', '\\1',
                           sapply(bmpheno$title, function(x) strsplit(x, ' \\(')[[1]][1]))
bmpheno$Batch <- ifelse(grepl('^P_', bmpheno$Samplename), 'P1',
       ifelse(grepl('^P2_', bmpheno$Samplename), 'P2',
              ifelse(grepl('^HCT', bmpheno$Samplename), 'C1', 'C2')))
bmpheno$Tissue <- ifelse(grepl('^C', bmpheno$Batch),'cell_line',
                         ifelse(grepl('_N', bmpheno$Samplename),
                                'normal', 'tumor'))
bmpheno$IHC <- ifelse(is.na(bmpheno$subtype.ihc.ch1),
                      bmpheno$subtypeihc.ch1, bmpheno$subtype.ihc.ch1)
bmpheno$IHC <- ifelse(is.na(bmpheno$IHC), 'cell_line', bmpheno$IHC)
bmpheno$Age <- ifelse(is.na(bmpheno$age.bin.ch1), bmpheno$agebin.ch1, bmpheno$age.bin.ch1)
rownames(bmpheno) <- bmpheno$Samplename

bmpheno <- bmpheno[,c(1:2,53:ncol(bmpheno))]
str(bmpheno)
```

Breast cancer



Get beta-values

```
## Get beta value
bm_beta_raw <- exprs(breast_meth)
head(bm_beta_raw[,1:5])
```

```
##            GSM519817 GSM519818 GSM519819 GSM519820 GSM519821
## cg00000292      0.67      0.80      0.56      0.85      0.52
## cg00002426      0.42      0.37      0.20      0.41      0.14
## cg00003994      0.05      0.08      0.25      0.16      0.25
## cg00005847      0.46      0.67      0.57      0.50      0.49
## cg00006414      0.02      0.04      0.05      0.04      0.03
## cg00007981      0.03      0.04      0.07      0.07      0.05
```

```
## all checks a logical vector and returns TRUE if
## all entries in the vector is true
all(colnames(bm_beta_raw) == bmpheno$geo_accession)
```

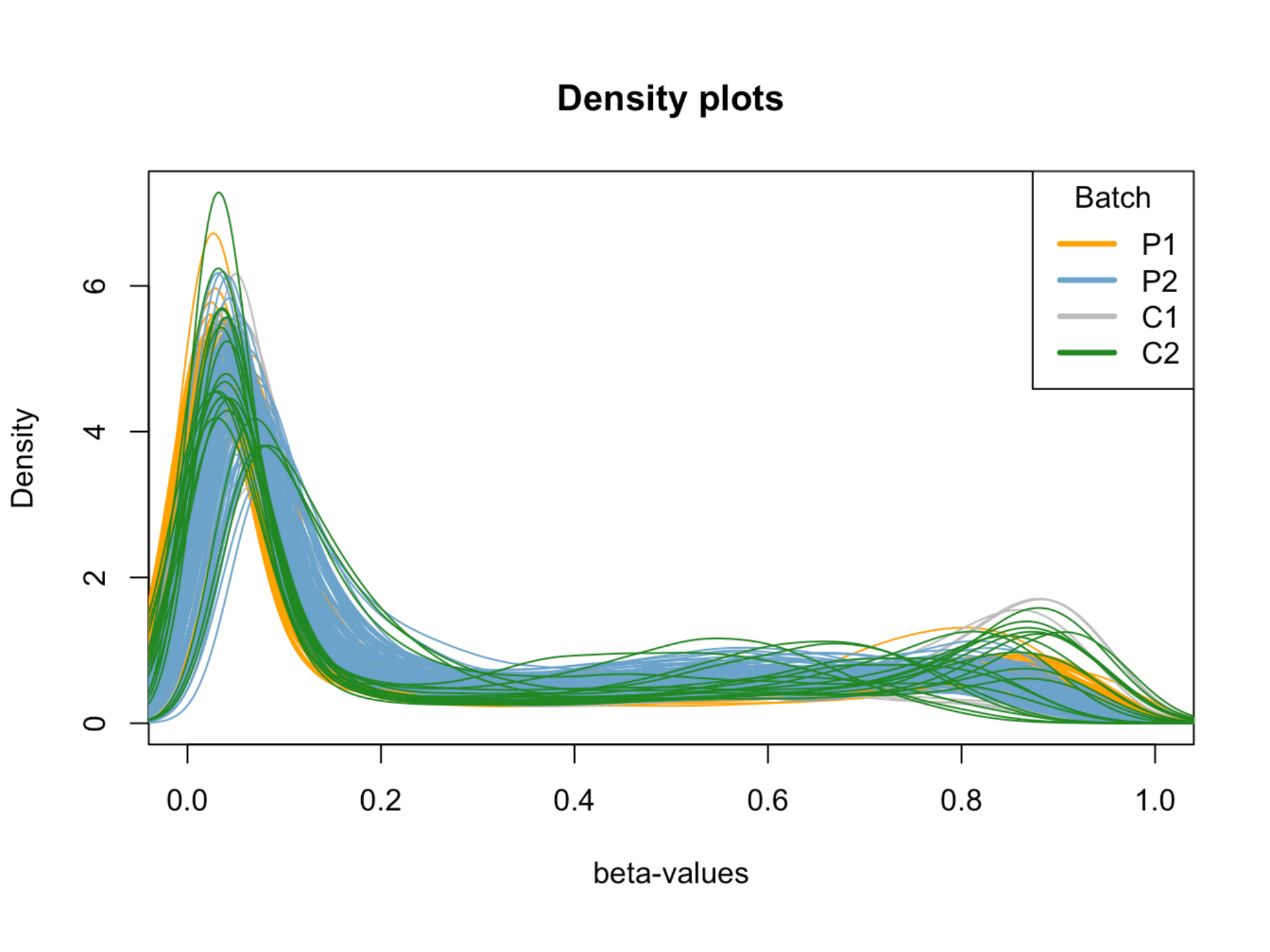
```
## [1] TRUE
```

QC

```
## Get density information
bm_density <- apply(bm_beta_raw, 2, density, na.rm = TRUE)
ylimits <- range(unlist(lapply(bm_density, '[[', 'y')))
```

Now that we have all the densities calculated and the y-limits, let's make our plot.

```
## Make plot
# Batch
plot(0, type = 'n', ylim = ylimits, xlim = c(0,1),
     main = 'Density plots',  xlab = 'beta-values',
     ylab = 'Density')
for(i in 1:length(bm_density)){
  lines(bm_density[[i]], col = colors_batch[i])
}
legend('topright', legend = names(mapper_batch),
       col = mapper_batch, lwd = 3,
       title = 'Batch')
```


Density plots
Batch
P1
P2
C1
C2
Density
0
2
4
6
0.0
0.2
0.4
0.6
0.8
1.0
beta-values

Subset for tissue only

```
## logical indexing
idx <- bmpheno$Tissue != 'cell_line'
tm_beta_raw <- bm_beta_raw[,idx]
tmpheno <- bmpheno[idx,]
```

```
> dim(tmpheno)
[1] 248  32
> dim(tm_beta_raw)
[1] 27578   248
```

Make colors

```
students <- c('dorms','offcampus','dorms','dorms','offcampus')
mapper <- c('dorms' = 'skyblue3', 'offcampus' = 'orange')
mapper['dorms']
```

```
> mapper
     dorms  offcampus
"skyblue3"   "orange"
```

```
##      dorms
## "skyblue3"
```

```
mapper[c('dorms','dorms')]
```

```
##      dorms      dorms
## "skyblue3" "skyblue3"
```

```
mapper[students]
```

```
##      dorms  offcampus      dorms      dorms  offcampus
## "skyblue3"   "orange" "skyblue3" "skyblue3"   "orange"
```

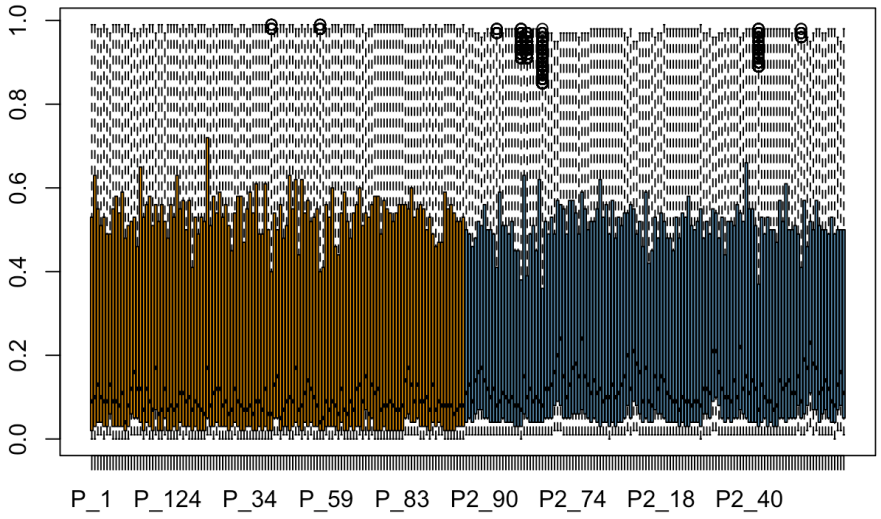
Normalization

```
#### Methylation data
tm_beta_raw <- tm_beta_raw[complete.cases(tm_beta_raw),]
tm_beta <- normalizeQuantiles(tm_beta_raw)
```

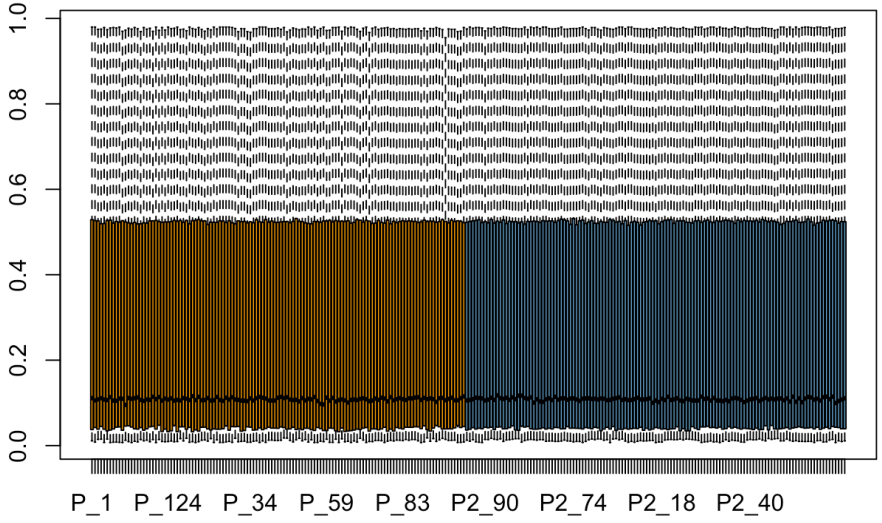
```
boxplot(tm_beta_raw, col = colors_batch)
```

```
boxplot(tm_beta, col = colors_batch)
```

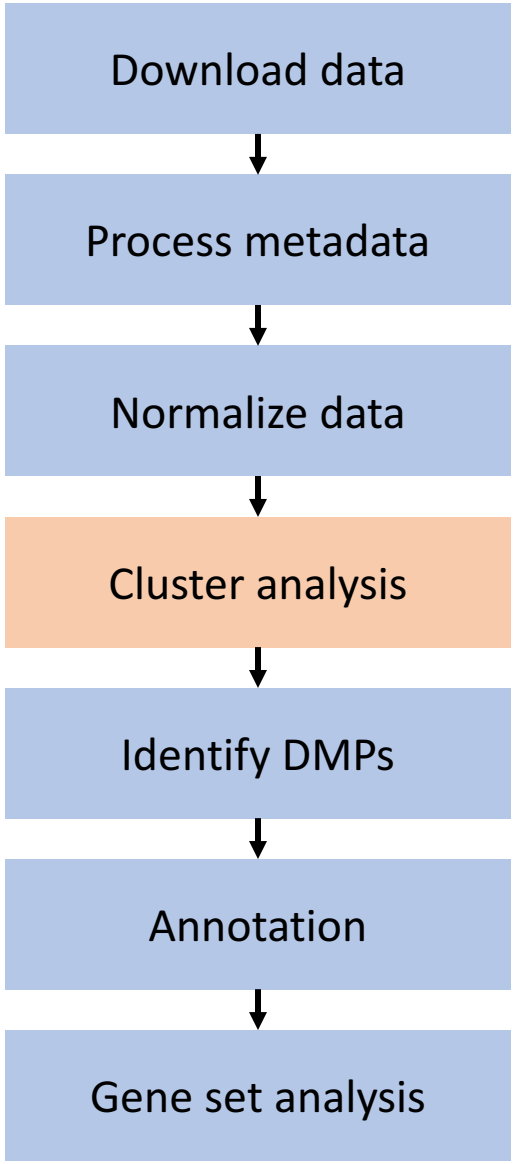
Raw



Quantile



Breast cancer



Curse of dimensionality

- Too much data can allow overfitting
  - E.g. machine learning or classification problems
  - One can train a classifier to identify the training set perfectly
  - But classifier fails in external data set
  - Reason: Classifier learned noise or all variation to be associated with classification
- Difficult to visualize

Dimensionality reduction

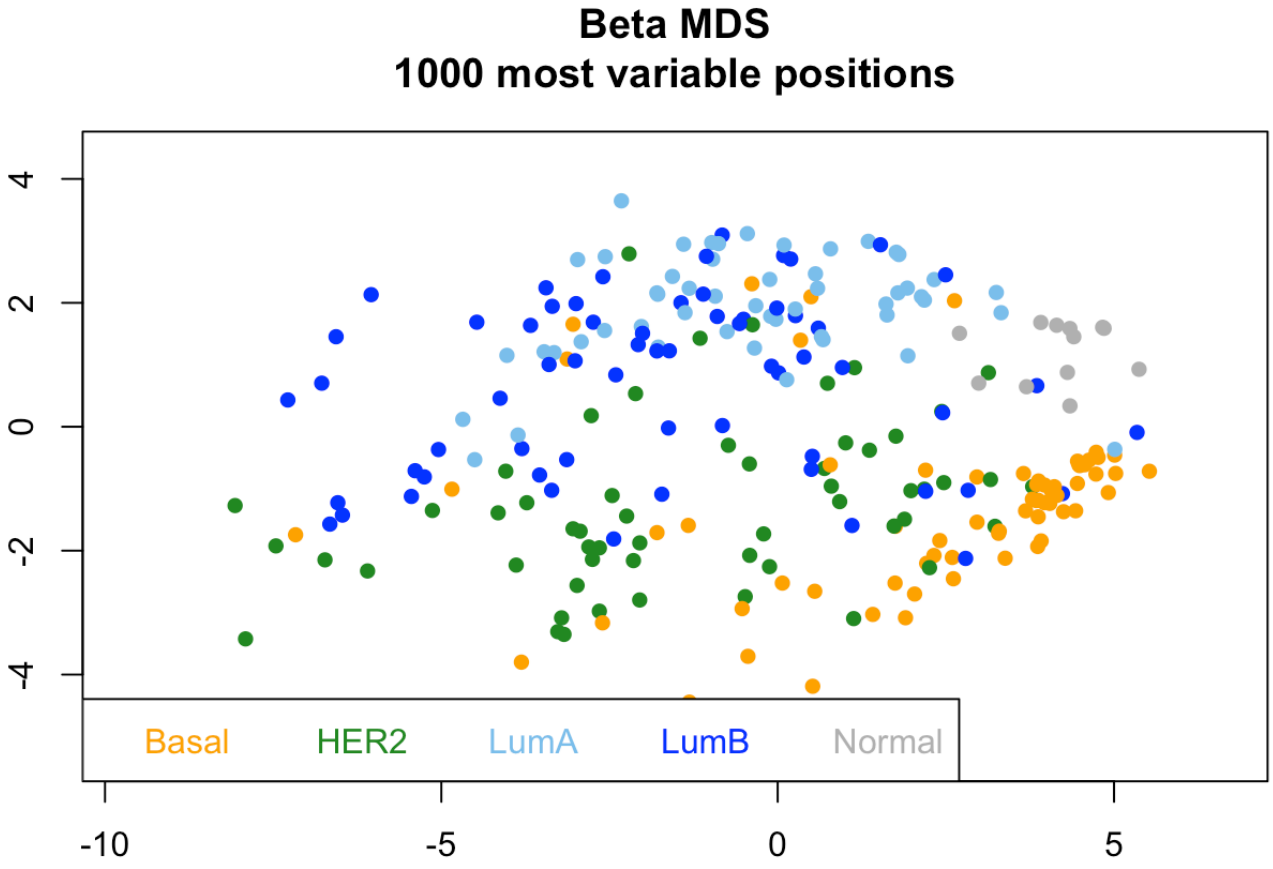
- Feature projection
  - Multidimensional scaling (MDS)
  - Principal component analysis (PCA)
- Feature selection
  - Most variable probes

Multidimensional scaling (MDS)

- Classical MDS
- While calculating projections, preserve pairwise distances between all elements in a complex data space
- Imagine flattening a 3D object to a 2D object across a single plane and choosing the angle to flatten that preserves distances
- Shadow

mdsPlot

```
mdsPlot(tm_beta, sampGroups = tmpheno$IHC, pch = 16, pal = mapper_ihc, numPositions = 1000)
```



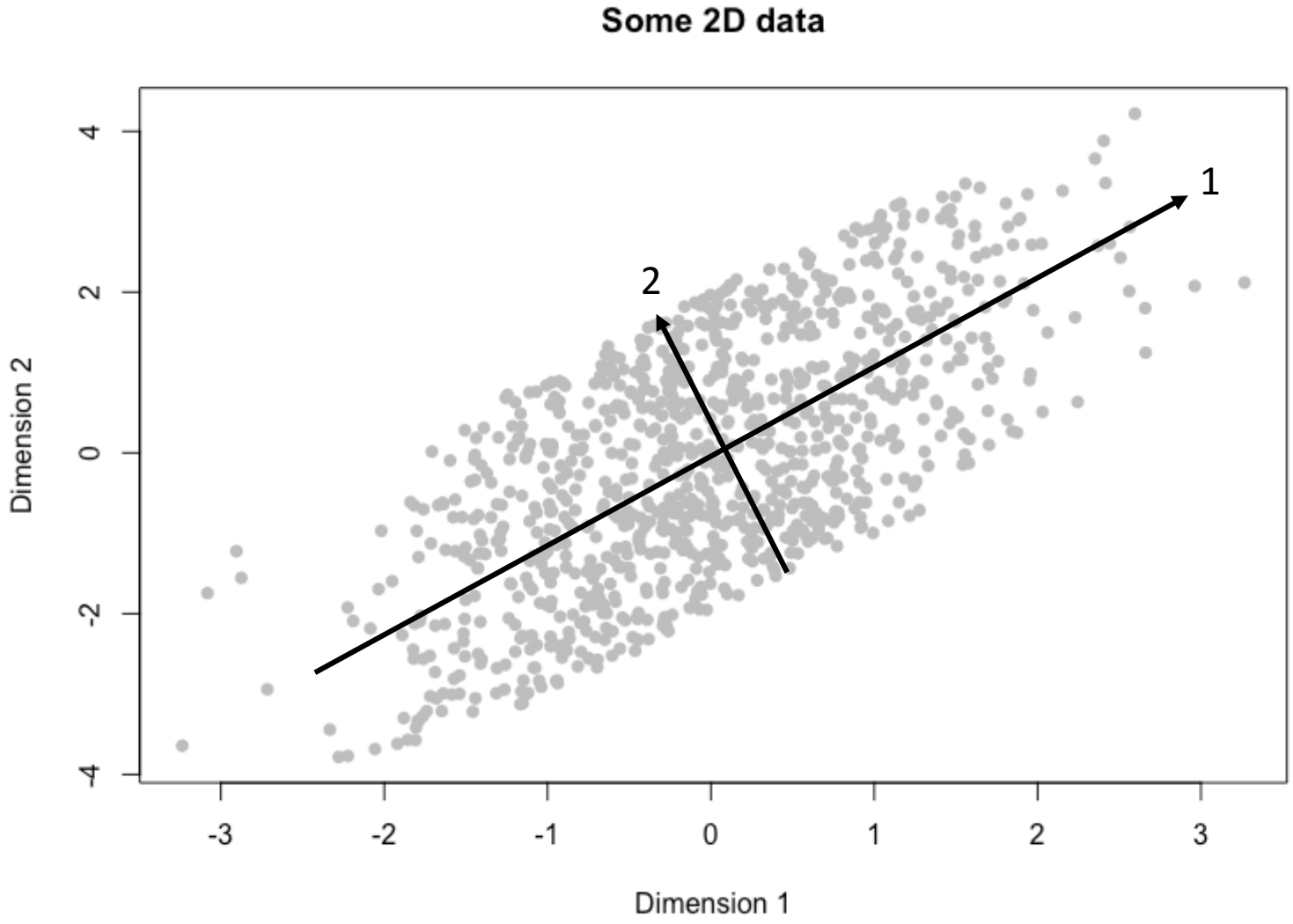
Most variable probes

- Assumption: a small number of probes capture a lot of the inter-sample variation and therefore the structure of the data
- Identify the most-variable probes by standard deviation to simplify exploratory analysis

```
tm_sd <- rowSds(tm_beta)
top1000q <- (nrow(tm_beta) - 1000)/nrow(tm_beta)
mvp_idx <- tm_sd >= quantile(tm_sd, top1000q)
tm_mvp <- tm_beta[mvp_idx,]
mvp_names <- rownames(tm_mvp)
```

`rowSds:` calculate standard deviation by row.

Principal component analysis (PCA)

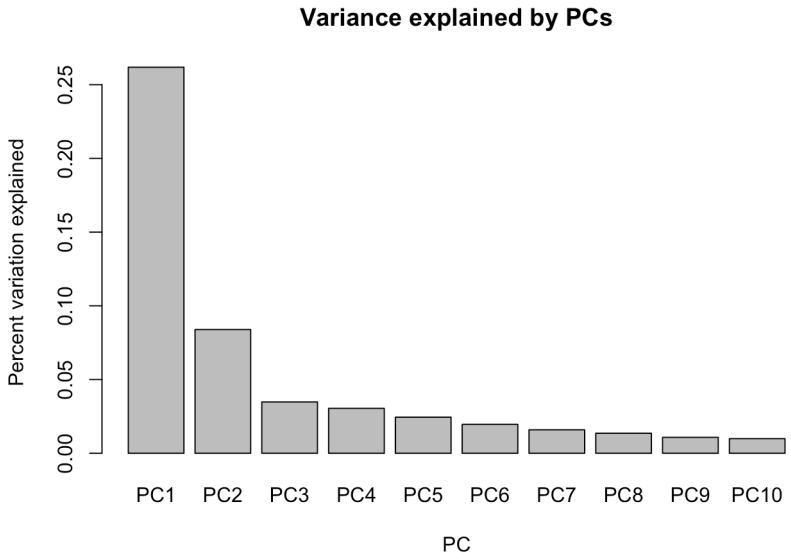


PCA

```
tmpca <- prcomp(t(tm_mvp))
summary(tmpca)$importance[,1:10]
```

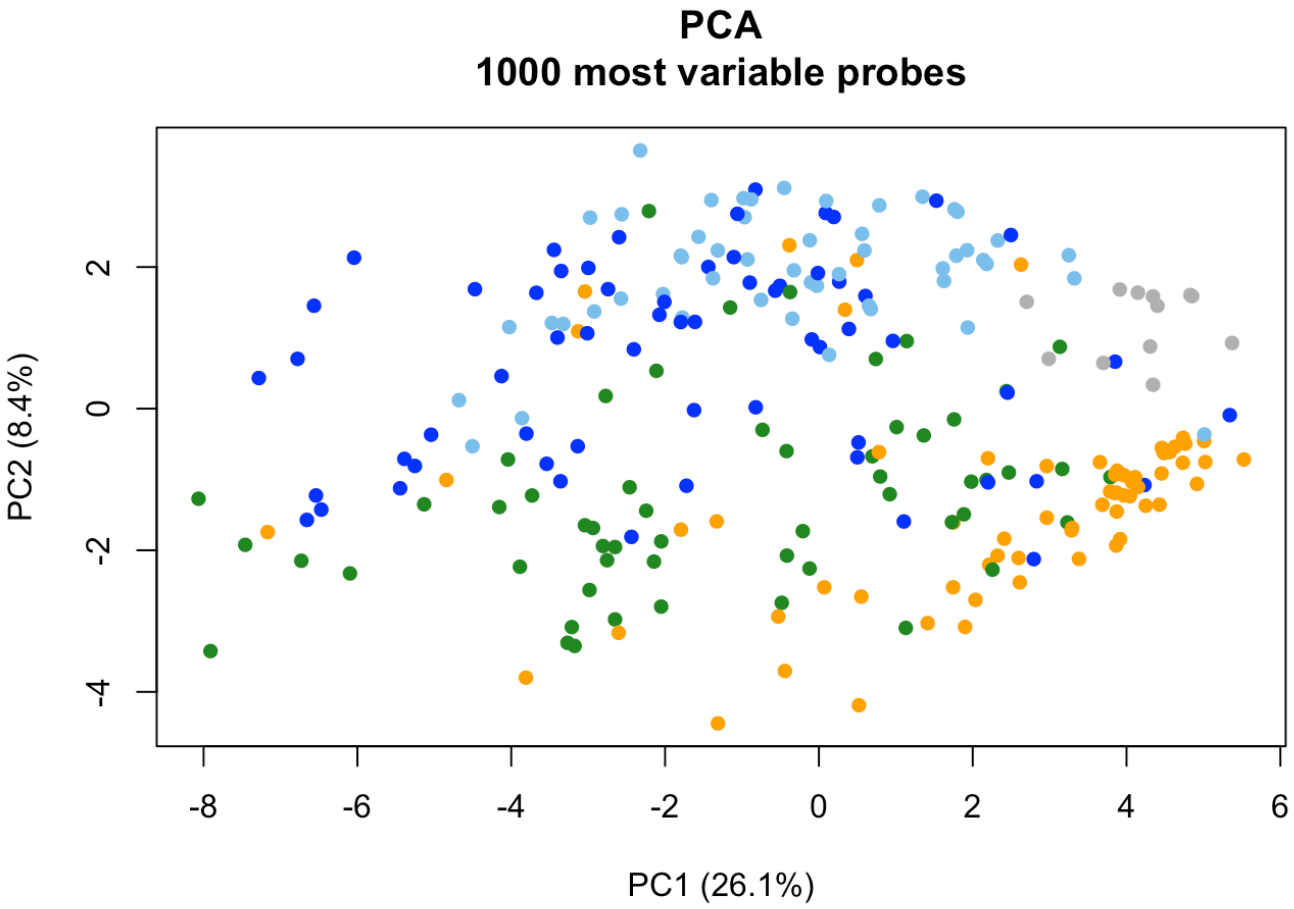
```
##                             PC1      PC2      PC3      PC4       PC5
## Standard deviation     3.223347 1.825062 1.175249 1.099454 0.9853985
## Proportion of Variance 0.261870 0.083950 0.034810 0.030470 0.0244700
## Cumulative Proportion  0.261870 0.345820 0.380630 0.411090 0.4355700
##                             PC6      PC7       PC8       PC9      PC10
## Standard deviation     0.8817906 0.794307 0.7340653 0.6551676 0.6268633
## Proportion of Variance 0.0196000 0.015900 0.0135800 0.0108200 0.0099000
## Cumulative Proportion  0.4551600 0.471070 0.4846500 0.4954700 0.5053700
```

```
barplot(summary(tmpca)$importance[2,1:10], ylab = 'Percent variation explained', xlab = 'PC', main = 'Variance explained by PCs')
```



PCA

```
plot(tmpca$x[,1], tmpca$x[,2], pch = 16, col = colors_ihc,
     main = 'PCA\n1000 most variable probes',
     xlab = 'PC1 (26.1%)', ylab = 'PC2 (8.4%)')
```

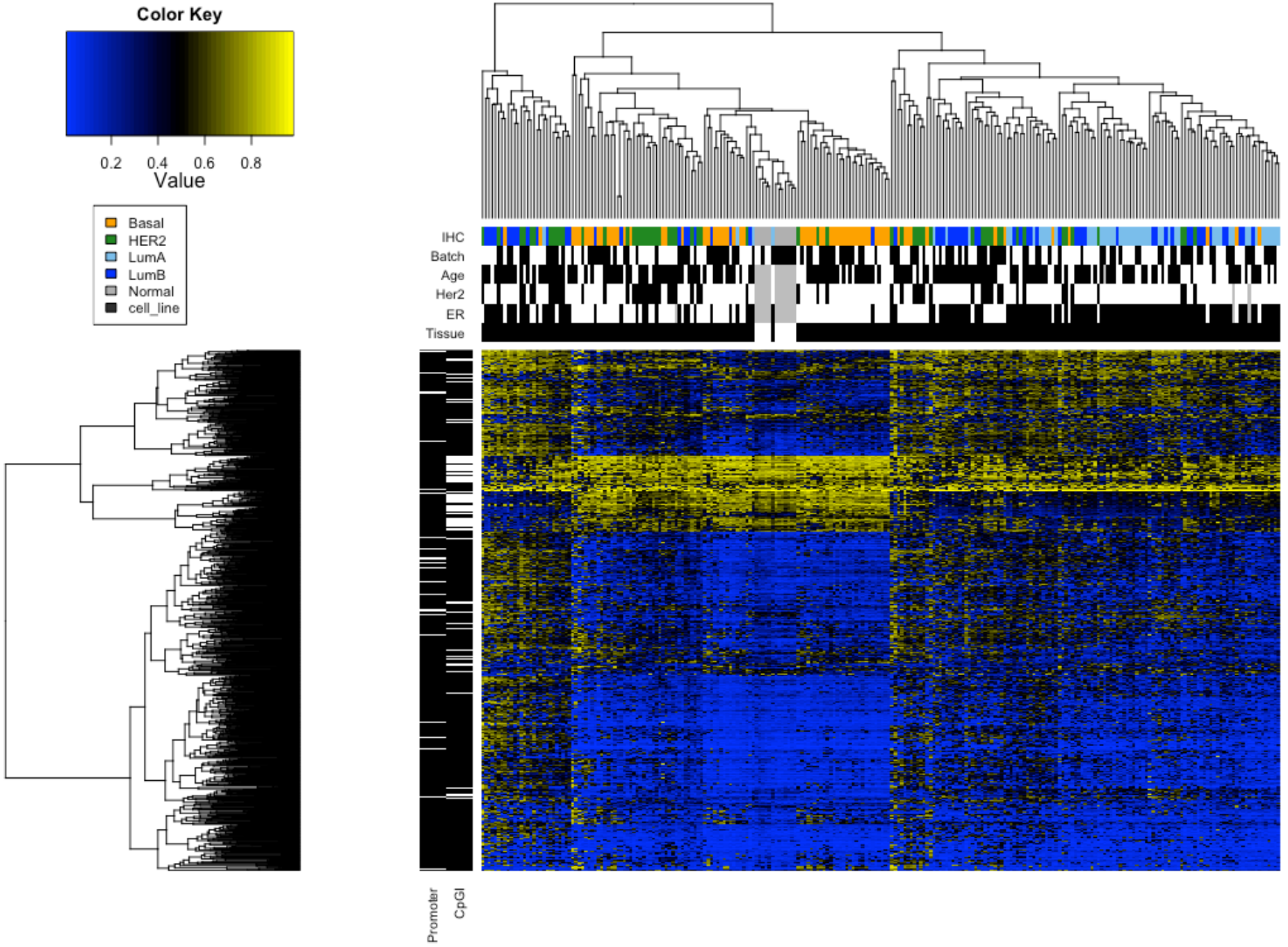


Hierachical clustering and heatmaps

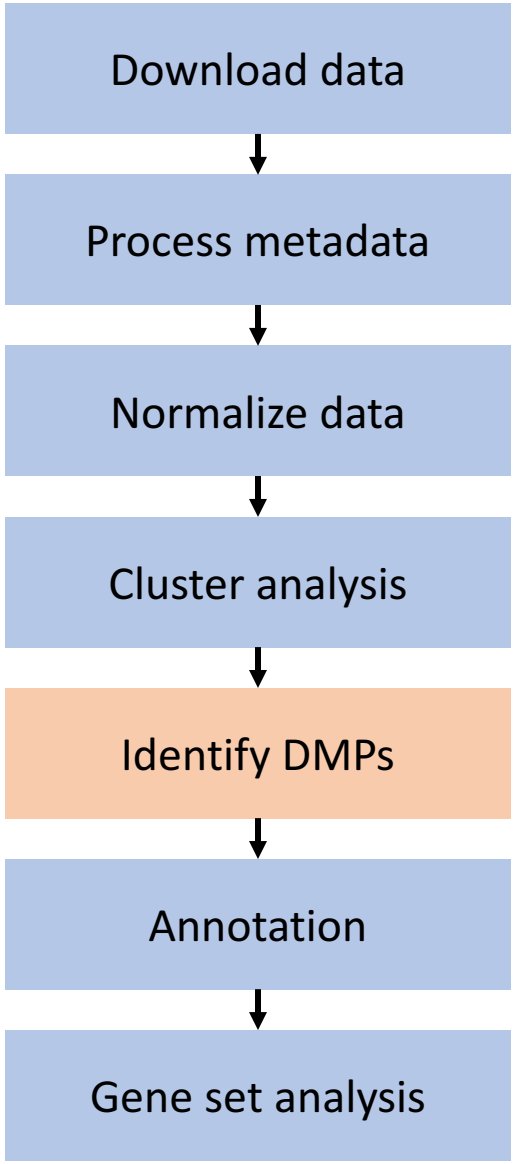
- Visualize the data using heatmaps
- Hierachical clustering of samples and features (probes)
- Identification of stable clusters using `pvclust`
  - Permutation analysis
  - Not done for the workshop

Hierachical clustering

```
## make column side colors
# tissue type
hmcolor_tissue <- c('normal' = 'white', 'tumor' = 'black', 'NA' = 'grey')[tmpheno$Tissue]
# er status
hmcolor_er <- c('0' = 'white', '1' = 'black', 'NA' = 'grey')[tmpheno$er.ch1]
# her2 status
hmcolor_her2 <- c('0' = 'white', '1' = 'black', 'NA' = 'grey')[tmpheno$her2.ch1]
# age
hmcolor_age <- c('0' = 'white', '1' = 'black', 'NA' = 'grey')[tmpheno$Age]
# batch
hmcolor_batch <- c('P1' = 'white', 'P2' = 'black', 'NA' = 'grey')[tmpheno$Batch]
# ihc
hmcolor_ihc <- colors_ihc
# make the matrix
column_colors <- cbind(Tissue = hmcolor_tissue, ER = hmcolor_er, Her2 = hmcolor_her2,
                       Age = hmcolor_age, Batch = hmcolor_batch, IHC = hmcolor_ihc)
## make row side colors
# promoter
hmrow_promoter <- ifelse(annots[mvp_names,]$Promoter == 'yes', 'black', 'white')
# cpg island
hmrow_cpgi <- ifelse(annots[mvp_names,]$CPG_ISLAND, 'black', 'white')
# make the matrix
row_colors <- rbind(NA, NA, NA, NA,
                    Promoter = hmrow_promoter,
                    CpGI = hmrow_cpgi)
## heatmap
myheatmap3(tm_mvp, ColSideCol = column_colors,
           RowSideCol = row_colors,
           side.height.fraction=0.8, labRow = NA, labCol = NA,
           margins = c(6,3))
legend(0, 0.8, legend = names(mapper_ihc), fill = mapper_ihc, cex = 0.7)
```


Color Key
0.2
0.4
0.6
0.8
Value
Basal
HER2
LumA
LumB
Normal
cell_line
IHC
Batch
Age
Her2
ER
Tissue
Promoter
CpGI

Breast cancer



M-values, design, and contrasts

```
## get M-values
tm_m <- logit2(tm_beta)
## make model matrix and contrasts
tissue_model <- model.matrix(~ 0 + tmpheno$Tissue)
colnames(tissue_model) <- c('tumor','normal')
tissue_contrasts <- makeContrasts(tumor - normal, levels = tissue_model)
```

Fit it

```
## fit model
tissue_fit1 <- lmFit(tm_m, design = tissue_model)
tissue_fit2 <- contrasts.fit(tissue_fit1, tissue_contrasts)
tissue_eb <- eBayes(tissue_fit2)
tissue_res <- topTable(tissue_eb, coef = 1, number = nrow(tm_m))

## do the same for beta-values
tissue_fit1b <- lmFit(tm_beta, design = tissue_model)
tissue_fit2b <- contrasts.fit(tissue_fit1b, tissue_contrasts)
tissue_ebb <- eBayes(tissue_fit2b)
tissue_resb <- topTable(tissue_ebb, coef = 1, number = nrow(tissue_ebb))

## flag differentially methylated probes
dmps <- rownames(tissue_res)[tissue_res$adj.P.Val <= 0.05] ## FDR < 0.05 in M-value
dmps <- dmps[abs(tissue_resb[dmps,]$logFC) >= 0.3] ## delta beta >= 0.3
tissue_resb$mfdr <- tissue_res[rownames(tissue_resb),]$adj.P.Val
tissue_resb$DMP <- rownames(tissue_resb) %in% dmps

head(tissue_resb)
```

DMPs

```
sum(tissue_resb$DMP)
```

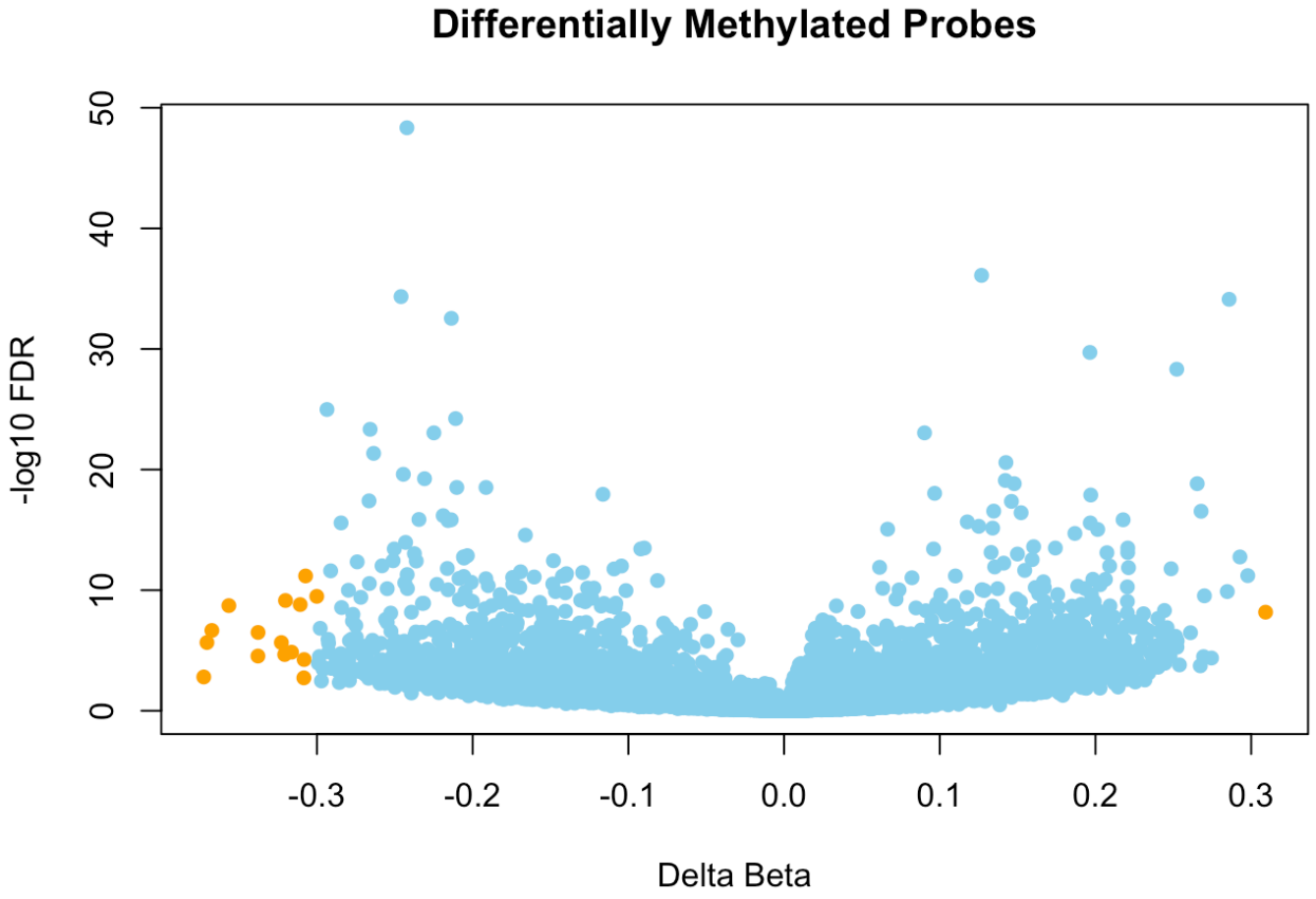
```
## [1] 16
```

```
head(subset(tissue_resb, DMP))
```

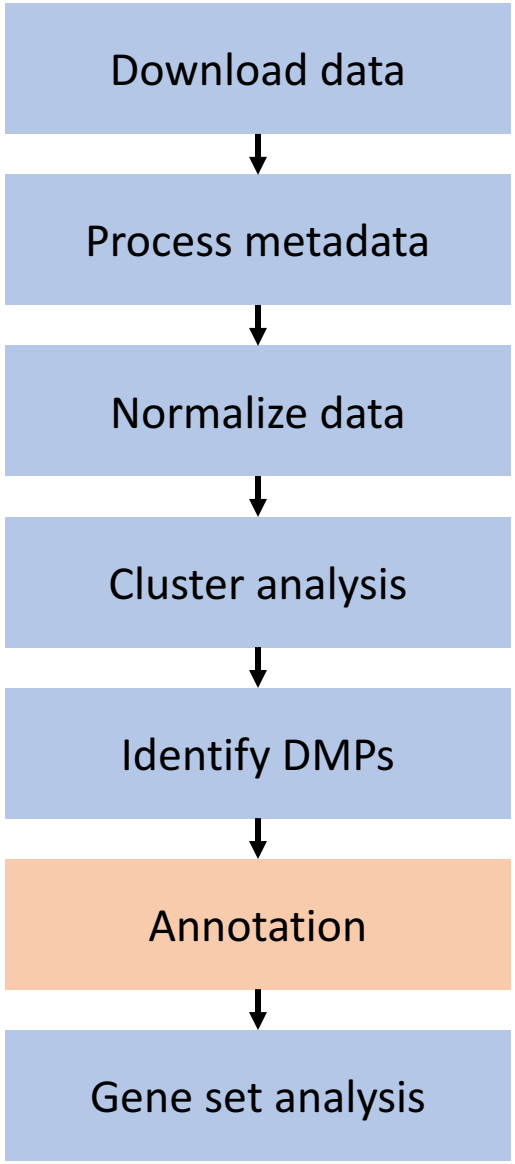
```
##                 logFC   AveExpr         t      P.Value    adj.P.Val
## cg25902889 -0.3073020 0.6960034 -8.132821 2.041019e-14 6.622026e-12
## cg04456238 -0.3001156 0.5323670 -7.448008 1.580950e-12 3.253690e-10
## cg01009664 -0.3202005 0.6292375 -7.309264 3.712436e-12 7.209969e-10
## cg13288195 -0.3107772 0.6294138 -7.162597 9.055828e-12 1.570702e-09
## cg02164046 -0.3565839 0.4826387 -7.125866 1.130194e-11 1.912178e-09
## cg02989940  0.3092370 0.4457428  6.886337 4.709808e-11 6.695210e-09
##                   B         mfdr  DMP
## cg25902889 22.05692 6.798652e-10 TRUE
## cg04456238 17.78850 2.163530e-10 TRUE
## cg01009664 16.95211 2.115743e-08 TRUE
## cg13288195 16.07897 1.114382e-08 TRUE
## cg02164046 15.86212 7.926072e-11 TRUE
## cg02989940 14.46612 1.650380e-08 TRUE
```

DMPs

```
plot(x = tissue_resb$logFC, y = -log10(tissue_resb$adj.P.Val),
     col = ifelse(tissue_resb$DMP, 'orange', 'skyblue'), pch = 16,
     ylab = '-log10 FDR', xlab = 'Delta Beta', main = 'Differentially Methylated Probes')
```



Breast cancer



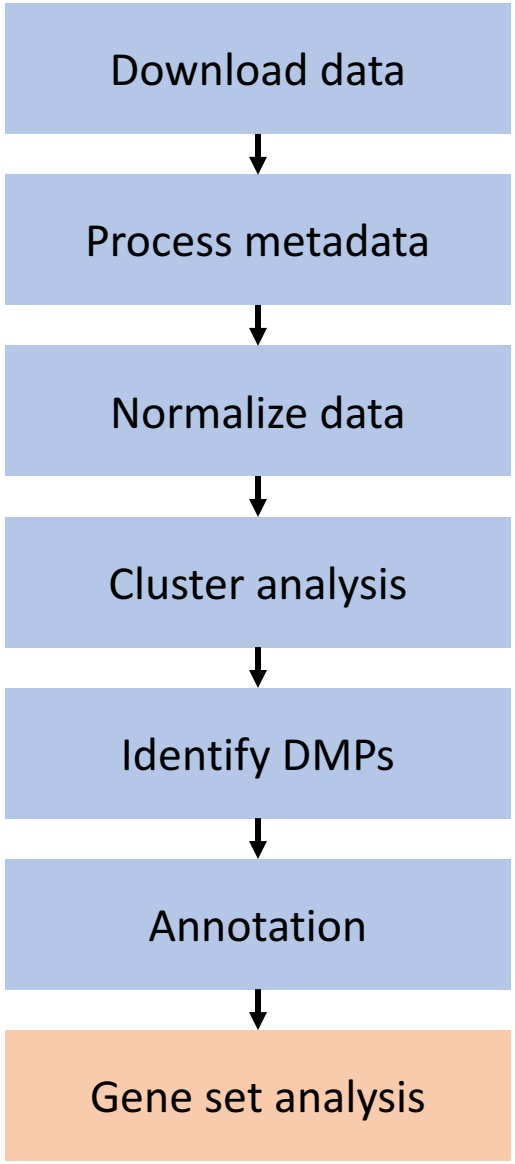
Annotation

```
toannotate <- annots[,c('Symbol', 'CPG_ISLAND', 'Distance_to_TSS', 'Promoter')]
## merge table 1 with table 2 by column. If by = 0, by row names
tissue_annotated <- merge(tissue_resb, toannotate, by = 0)
rownames(tissue_annotated) <- tissue_annotated$Row.names
tissue_annotated$Row.names <- NULL
## rearrange by pvalue and logfc
tissue_annotated <- tissue_annotated[order(
  -tissue_annotated$DMP, ## first by inverse DMP status
  tissue_annotated$mfdr, ## then by p-values (low to high)
  -1*abs(tissue_annotated$logFC)),] ## then by inverse delta-beta
tissue_annotated$dmp0.3 <- tissue_annotated$mfdr <= 0.05 & abs(tissue_annotated$logFC) >= 0.3
tissue_annotated$dmp0.2 <- tissue_annotated$mfdr <= 0.05 & abs(tissue_annotated$logFC) >= 0.2
head(tissue_annotated)
```

```
##                logFC   AveExpr         t      P.Value    adj.P.Val
## cg02164046 -0.3565839 0.4826387 -7.125866 1.130194e-11 1.912178e-09
## cg04456238 -0.3001156 0.5323670 -7.448008 1.580950e-12 3.253690e-10
## cg25902889 -0.3073020 0.6960034 -8.132821 2.041019e-14 6.622026e-12
## cg13288195 -0.3107772 0.6294138 -7.162597 9.055828e-12 1.570702e-09
## cg02989940  0.3092370 0.4457428  6.886337 4.709808e-11 6.695210e-09
## cg01009664 -0.3202005 0.6292375 -7.309264 3.712436e-12 7.209969e-10
##                   B         mfdr   DMP Symbol CPG_ISLAND Distance_to_TSS
## cg02164046 15.86212 7.926072e-11 TRUE    SST       TRUE              53
## cg04456238 17.78850 2.163530e-10 TRUE    WT1       TRUE              NA
## cg25902889 22.05692 6.798652e-10 TRUE   FSD1      FALSE             399
## cg13288195 16.07897 1.114382e-08 TRUE FBXL22      FALSE             115
## cg02989940 14.46612 1.650380e-08 TRUE   ERAF      FALSE              31
## cg01009664 16.95211 2.115743e-08 TRUE    TRH       TRUE              50
##            Promoter dmp0.3 dmp0.2
## cg02164046      yes   TRUE   TRUE
## cg04456238       no   TRUE   TRUE
## cg25902889      yes   TRUE   TRUE
## cg13288195      yes   TRUE   TRUE
## cg02989940      yes   TRUE   TRUE
## cg01009664      yes   TRUE   TRUE
```

https://www.genecards.org/cgi-bin/carddisp.pl?gene=WT1

Breast cancer



Gene set analysis

- Identify associations between differentially methylated genes and biology
- Gene sets: Any biologically relevant set of genes
  - Pathways
  - Cellular compartments
  - Cell type
  - Disease
  - Genes upregulated during drug treatment
  - Essential genes
  - DNA elements (bound by the same epigenetic effectors)

Gene set collections

- Molecular Signatures Database
- reactome
- NCI PID
- Pathway Commons
- Gene ontology
- Ingenuity Knowledge Base*
- etc…

Hallmark Gene Set

- Summarise specific “hallmark” biological processes and states
- Derived from founder sets and experimental data of genes that display coherent expression
- Expertly curated
- Reduced variation and redundancy*

```
## get gene sets
hallmarks <- readgmt('https://raw.githubusercontent.com/sean-cho/datasets/master/datasets/h.all.v6.2.symbols.gmt'
)$genesets
names(hallmarks) <- gsub('HALLMARK_', '', names(hallmarks))
names(hallmarks)
```

Types of gene set analysis

1. Over-representation analysis (ORA)
   - How likely are genes within a gene set over-represented in DMGs?
   - Requires thresholding.
2. Functional class scoring (FCS)
   - Given a set of statistics, does a gene set have a score higher than permuted null?
3. Pathway topology based (PTB)
   - Given a pathway topology or network, calculate scores based on some gene statistic from the array and weights of relationships between genes.

Wrangle our data (again)

- Prepare two sets of DMPs for ORA
  - abs(delta beta) >= 0.3
  - abs(delta beta) >= 0.2
- Prepare statistics for FCS

```
## 0.3
genes0.3 <- names(which(tapply(tissue_annotated$dmp0.3,
                               tissue_annotated$Symbol, max) == 1))
length(genes0.3)
```

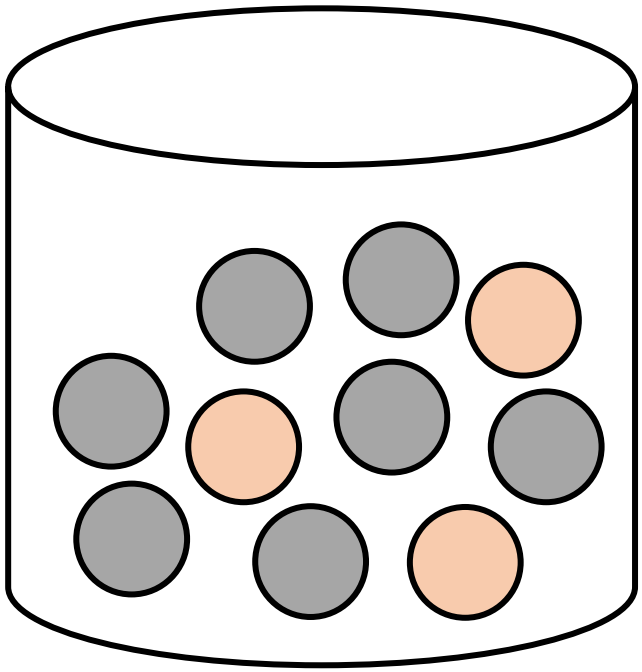
```
## [1] 15
```

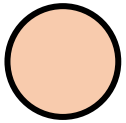
```
## 0.2
genes0.2 <- names(which(tapply(tissue_annotated$dmp0.2,
                               tissue_annotated$Symbol, max) == 1))
length(genes0.2)
```

```
## [1] 390
```

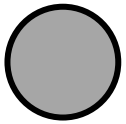
```
## get entry with the highest difference
getbest <- function(x) x[which.max(abs(x))]

## moderated t-statistic
genesall <- unique(tissue_annotated$Symbol)
gene_statistics <- tapply(tissue_annotated$t, tissue_annotated$Symbol, getbest)
```

ORA: Hypergeometric test





= in gene set



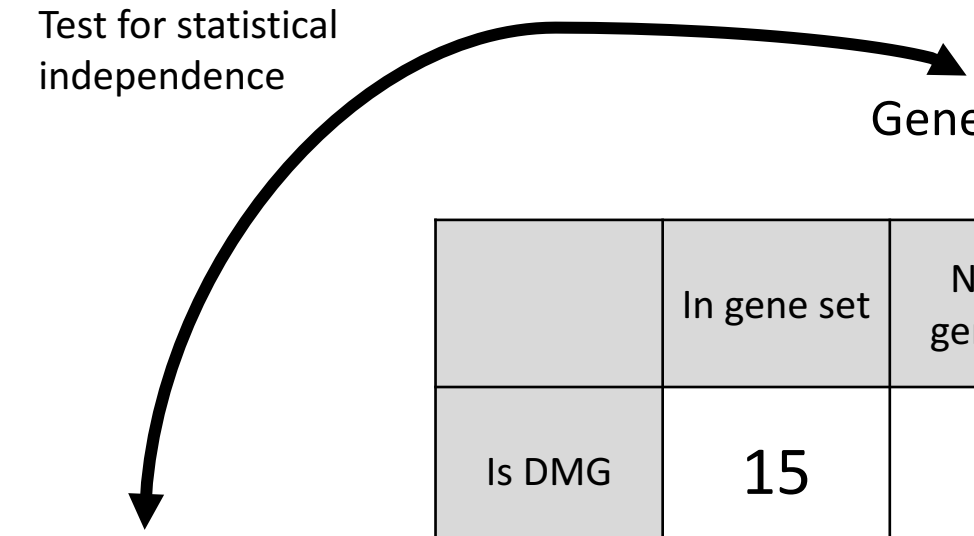
= other genes

- Given an urn with `m` colored balls and a total of `n` non-colored balls, draw `k` balls out without replacement.
- What is the probability of seeing `j` or more colored balls?

Hypergeometric test

```
hg0.3_pval <- c()
hg0.2_pval <- c()
hg0.3_n <- c()
hg0.2_n <- c()
for(i in 1:length(hallmarks)){
  geneset <- hallmarks[[i]]
  univ <- union(genesall,geneset)
  hg0.2_n[i] <- sum(genes0.2 %in% geneset)
  hg0.3_n[i] <- sum(genes0.3 %in% geneset)
  hg0.2_pval[i] <- sig_overlap(genes0.2,geneset,univ)
  hg0.3_pval[i] <- sig_overlap(genes0.3,geneset,univ)
}
hg_res <- data.frame(
  genesets = names(hallmarks),
  hg0.2p = hg0.2_pval,
  hg0.2n = hg0.2_n,
  hg0.3p = hg0.3_pval,
  hg0.3n = hg0.3_n
  )
hg_res
```

ORA: Fisher's exact test



	In gene set	Not in gene set	
Is DMG	15	5	20
Not DMG	5	75	80
	20	80	100

Count if there are ways to arrange this contingency table in more extreme combinations by keeping the total counts in each row and column the same

Fisher's Exact Test

```
geneset1 <- hallmarks[[1]]
ctab <- table(genesall %in% genes0.3, genesall %in% geneset1, dnn = c('DMP','geneset'))
ctab
```

```
##        geneset
## DMP     FALSE  TRUE
##   FALSE 14283   179
##   TRUE     15     0
```

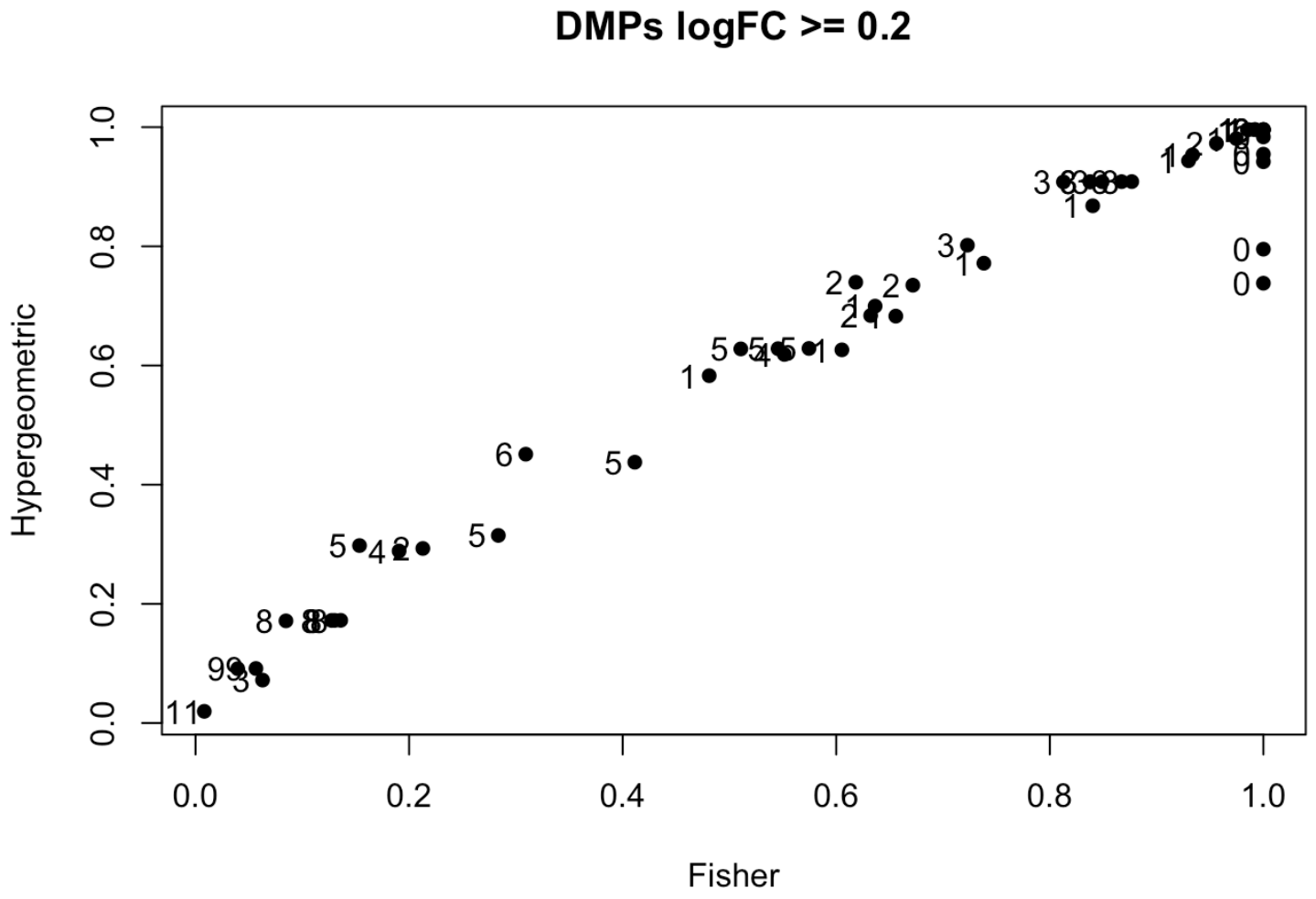
```
fisher.test(ctab)
```

```
##
##  Fisher's Exact Test for Count Data
##
## data:  ctab
## p-value = 1
## alternative hypothesis: true odds ratio is not equal to 1
## 95 percent confidence interval:
##   0.00000 22.44428
## sample estimates:
## odds ratio
##          0
```

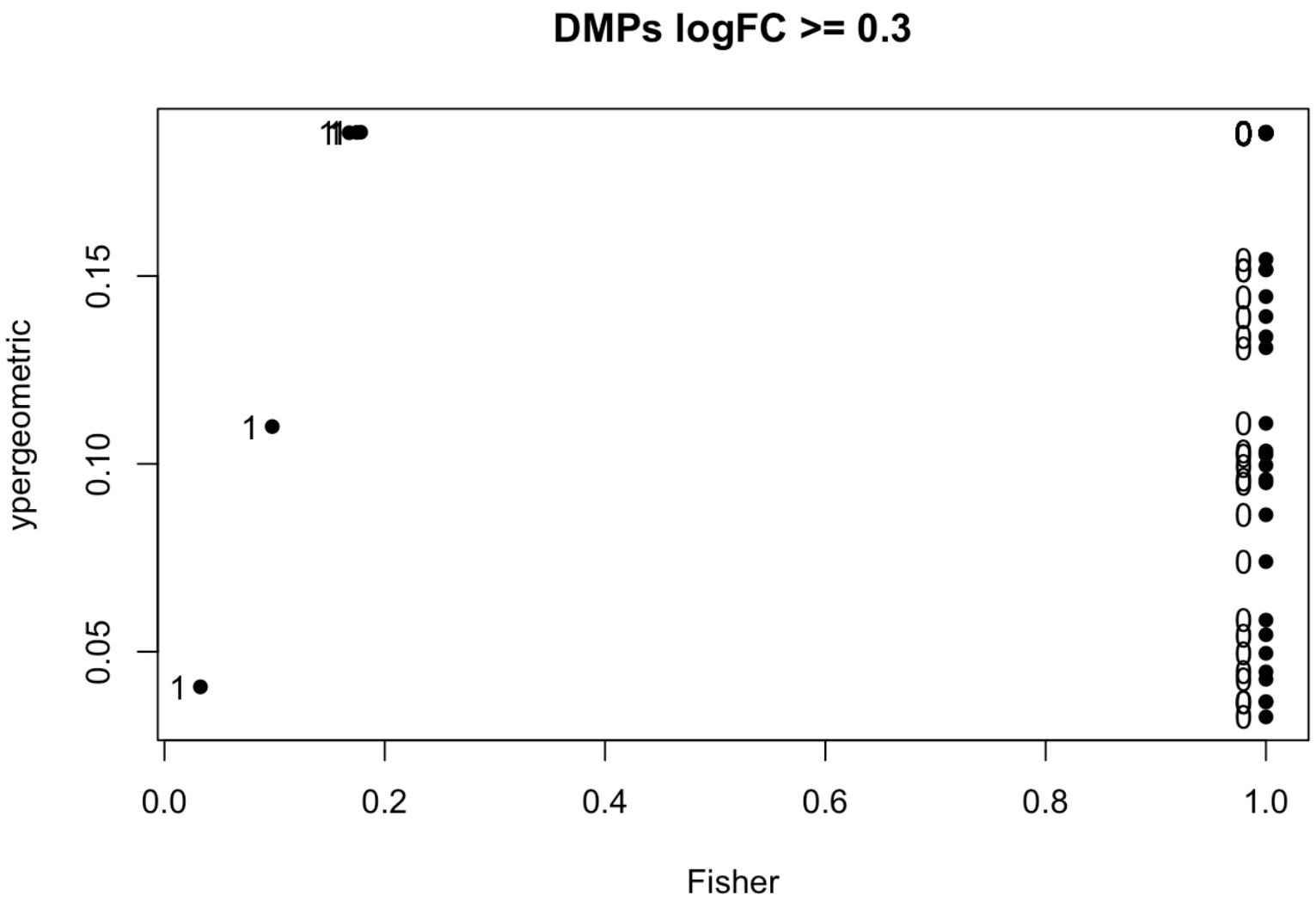
Fisher's Exact Test

```
fe0.2_pval <- c()
fe0.2_n <- c()
fe0.3_pval <- c()
fe0.3_n <- c()
for(i in 1:length(hallmarks)){
  geneset <- intersect(hallmarks[[i]], genesall)
  ctab0.2 <- table(genesall %in% genes0.2,genesall %in% geneset)
  ctab0.3 <- table(genesall %in% genes0.3,genesall %in% geneset)
    table(genesall %in% genes0.2,genesall %in% geneset)
  fe0.2_pval[i] <- fisher.test(ctab0.2,
                               alternative = 'greater')$p.value
  fe0.2_n[i] <- ctab0.2[2,2]
  fe0.3_pval[i] <- fisher.test(ctab0.3,
                               alternative = 'greater')$p.value
  fe0.3_n[i] <- ctab0.3[2,2]
}
# names(fe0.2_pval) <- names(hallmarks)
# names(fe0.3_pval) <- names(hallmarks)
fisher_res <- data.frame(
  genesets = names(hallmarks),
  fe0.2p = fe0.2_pval,
  fe0.2n = fe0.2_n,
  fe0.3p = fe0.3_pval,
  fe0.3n = fe0.3_n
  )
fisher_res
```

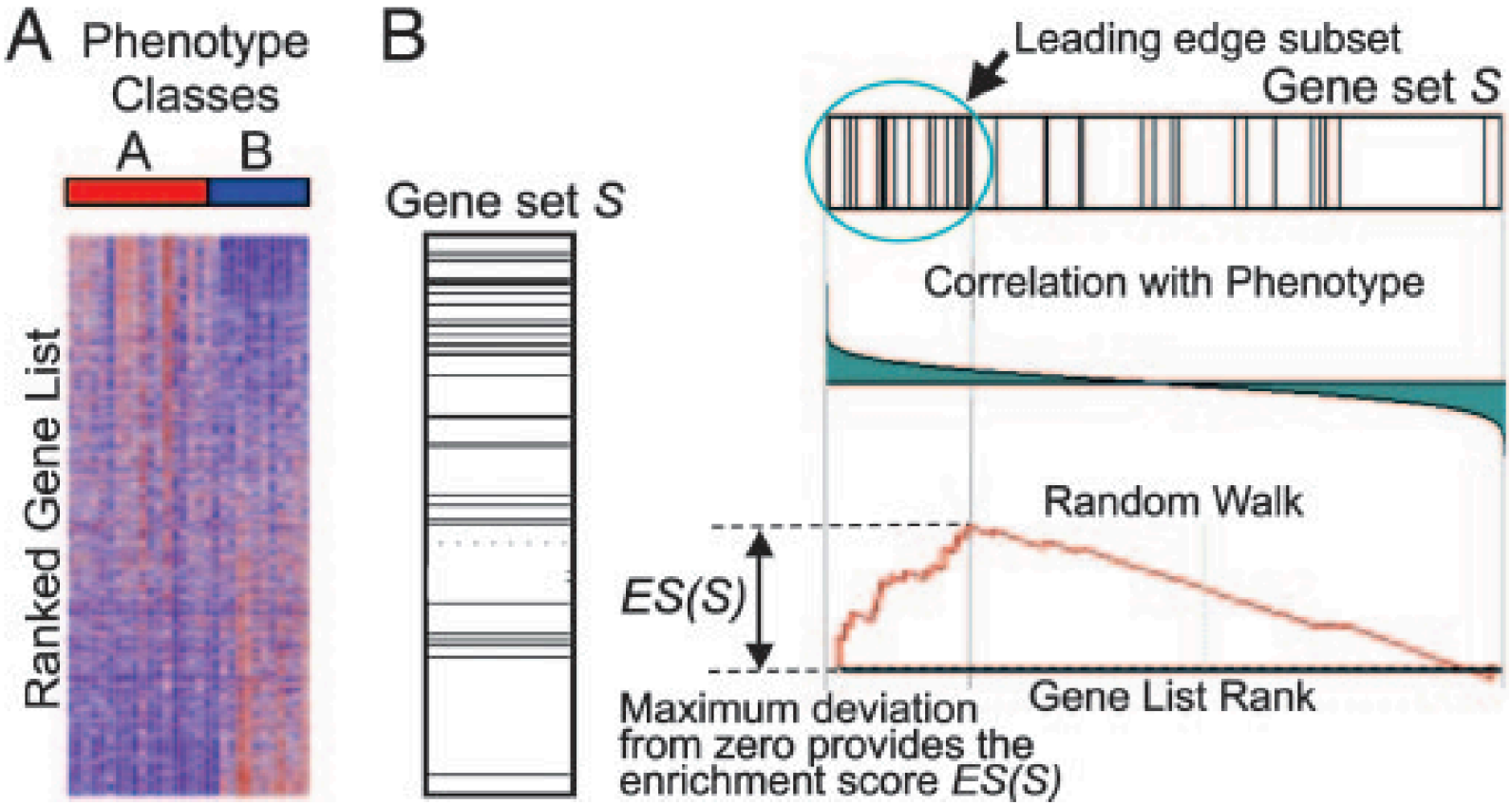
Hypergeometric vs. Fisher



Hypergeometric vs. Fisher



FCS: Gene set enrichment analysis



Subramanian, A, et al. "Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles." *Proceedings of the National Academy of Sciences* 102.43 (2005): 15545-15550

Mean-rank gene set enrichment (MR-GSE)

- Use moderated t-statistic as statistics
- Permutes null by randomizing gene labels

```
## convenience functions: gene set test wrapper and plotting
gstestwrapper <- function(gs, alternative = 'mixed'){
  cat('|')
  gs_idx <- names(gene_statistics) %in% gs
  n <- sum(gs_idx)
  pval <- wilcoxGST(gs_idx, gene_statistics, alternative = alternative)
  return(c(N = n, pval = pval))
}

plotgse <- function(gs, ...){
  gs_idx <- names(gene_statistics) %in% gs
  barcodeplot(statistics = as.numeric(gene_statistics),
              index = gs_idx, ...)
}

tissue_gse <- data.frame(t(sapply(hallmarks, gstestwrapper, alternative = 'less')),
                         stringsAsFactors = FALSE)
```

```
tissue_gse <- tissue_gse[order(tissue_gse$pval),]
tissue_gse$qval <- p.adjust(tissue_gse$pval, method = 'fdr')
```

Significant gene sets

```
subset(tissue_gse, qval <= 0.05)
```

```
##                                    N         pval         qval
## EPITHELIAL_MESENCHYMAL_TRANSITION 188 7.729898e-09 3.864949e-07
## SPERMATOGENESIS                   105 1.316100e-03 3.290250e-02
## PANCREAS_BETA_CELLS                32 3.853707e-03 4.967218e-02
## MYOGENESIS                        188 4.112013e-03 4.967218e-02
## ADIPOGENESIS                      169 4.967218e-03 4.967218e-02
```

Visualize

```
plotgse(hallmarks$EPITHELIAL_MESENCHYMAL_TRANSITION, main = 'EMT')
```