Análise de dados ómicos

Conceitos e implementação em R/ Bioconductor

Dados biológicos / **ómicos**

A **Biologia** tem-se tornado em anos recentes uma ciência com um contributo relevante para a geração de grandes quantidades de dados

Os dados biológicos cresceram dado o desenvolvimento tecnológico na sequenciação de ADN e em muitas outras técnicas de alto débito, que permitem a medição de dados em larga escala – chamados dados **ómicos**

A análise de dados para domínios biológicos é uma necessidade crescente em tarefas de investigação ou industriais, requerendo o conhecimento das interconexões de diferentes entidades e sistemas

A Biologia é atualmente uma ciência baseada em dados e informação

Dados ómicos

Genómica

Sequenciação de dados – genomas completos, examos, análise de variantese mutações, etc.

Transcritómica

Medição da expressão genética (mRNA)

Sequenciação de RNA, DNA microarrays

Epigenómica

Metilação de DNA (e.g. de sequenciação)

Proteómica

Medição da quantidade de proteínas

Espectrometria de massa (MS), protein microarrays, …

Metabolómica

Medição da quantidade de pequenos compostos envolvidos no metabolismo

Cromatografia + MS, RMN, etc

Interatómica

Interações Proteina-DNA, proteina-proteina, ...

Representação de dados ómicos

Dados ómicos seguem tipicamente a estrutura matricial que vimos anteriormente: variáveis tipicamente numéricas – medição de **entidades biológicas** (e.g. genes, proteínas, metabolitos) – linhas - em **amostras diferentes** – colunas**.**

Matrizes tipicamente de grande dimensão principalmente no que diz respeito ao número de linhas (entidades)

Embora estas matrizes encaixem na estrutura anterior, é comum considerar as amostras nas linhas (transpondo a matriz)

	Amostra 1	**Amostra 2**	**Amostra 3**	**…**
Gene 1	4.5	12.1	9.7	…
Gene 2	12.4	7.8	7.3	..
Gene 3	9.4	6.5	8.9	…
Gene 4	13.1	15.1	12.7	…
Gene 5	8.5	11.3	14.1	…
…	…	…	…	…

Representação de metadados

Na análise de dados ómicos, existem tipicamente complementares sobre as amostras (e.g. dados clínicos/ fenotípicos), que podem ser importantes do ponto de vista da análise

Normalmente chamamos a estes **metadados**, podendo ser representados numa outra matriz, seguindo a estrutura anterior

Em alguns casos, esta matriz (ou parte) é junta com a matriz de dados para efeitos de alguns tipos de análise

	Sangue	Mutações	Fumador	Sub-Tipo cancro
Amostra 1	234.1	Negativo	Sim	A
Amostra 2	117.2	A	Nao	B
Amostra 3	90.7	B	Nao	A
Amostra 4	85.9	B	Nao	B
…	…	…	…	…

Dados de expressão genética

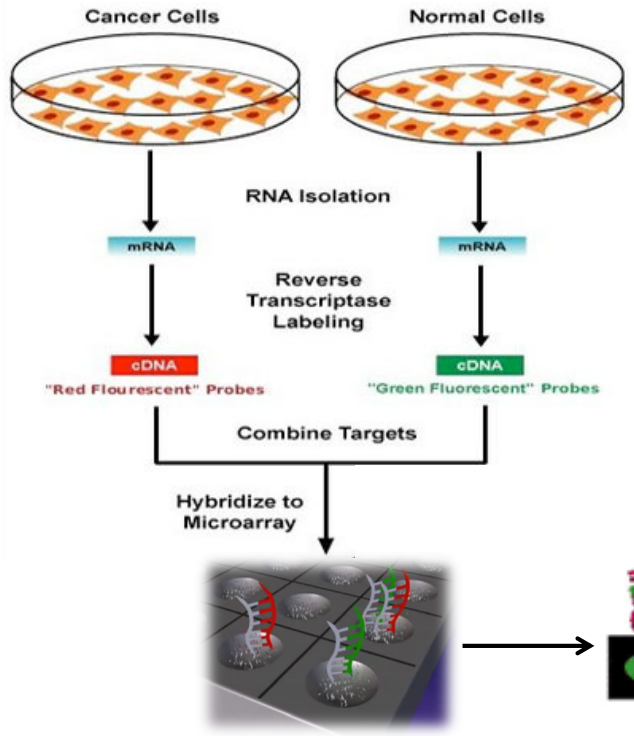
Técnicas como *DNA microarrays* e *RNA sequencing* medem o **nível de expressão de todos os genes** numa célula, em diversas condições/ instantes de tempo

Nível de expressão de um dado gene é estimado pela medição da quantidade de mRNA respectivo.

- Um gene está ativo se está a ser transcrito.
- Mais mRNA indica mais atividade do gene (embora em termos biológicos isto nem sempre seja verdade)

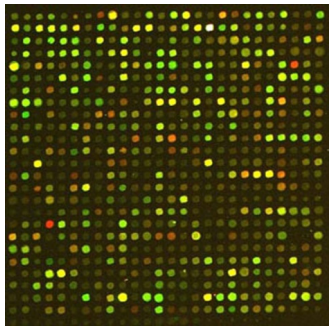
DNA Microarrays

Técnica que permite medir o nível de expressão de milhares de genes em simultâneo. Cada gene representado por uma sequência de DNA presente numa (ou em várias) posições do array



http://www.youtube.com/watch?v=VNsThMNjKhM

Microarray



RNA Sequencing

Técnicas mais recentes de ***next generation sequencing*** permitem sequenciar cDNA e assim ter medidas do mRNA associado

Estas fornecem medidas mais precisas do RNA lendo extratos de 30~400 bps; pouco ruído de background

Usadas mais frequentemente com eucariontes; permite medir diferentes alelos e isoformas (e.g. splicing alternativo)

Permitem medir transcriptomas de organismos não sequenciados

Permitem detectar diferenças numa escala maior (maior dynamic range) até valores de 8000 vezes

Tecnologias: *Illumina*, *Roche 454*, *ABS*

Preparação dos dados

Dados provenientes dos equipamentos são tipicamente extensos e com muita informação que é necessário tratar antes da análise dos dados

Grande parte do tratamento inicial é realizado por software específico para cada equipamento, sendo dependente da experiência e equipamento

Passos deste primeiro pré-processamento em microarrays incluem o tratamento da imagem para chegar a um valor de expressão da probe, eliminação de ruído de background, consolidação de valores de réplicas de probes, controlo de qualidade, etc

Os dados de arrays de 2 cores têm tipicamente mais necessidade de pré-tratamento por serem menos standardizados; os de arrays de oligonucleotidos incluem mais processamento feito por software próprio obtendo-se dados em bruto mais consistentes

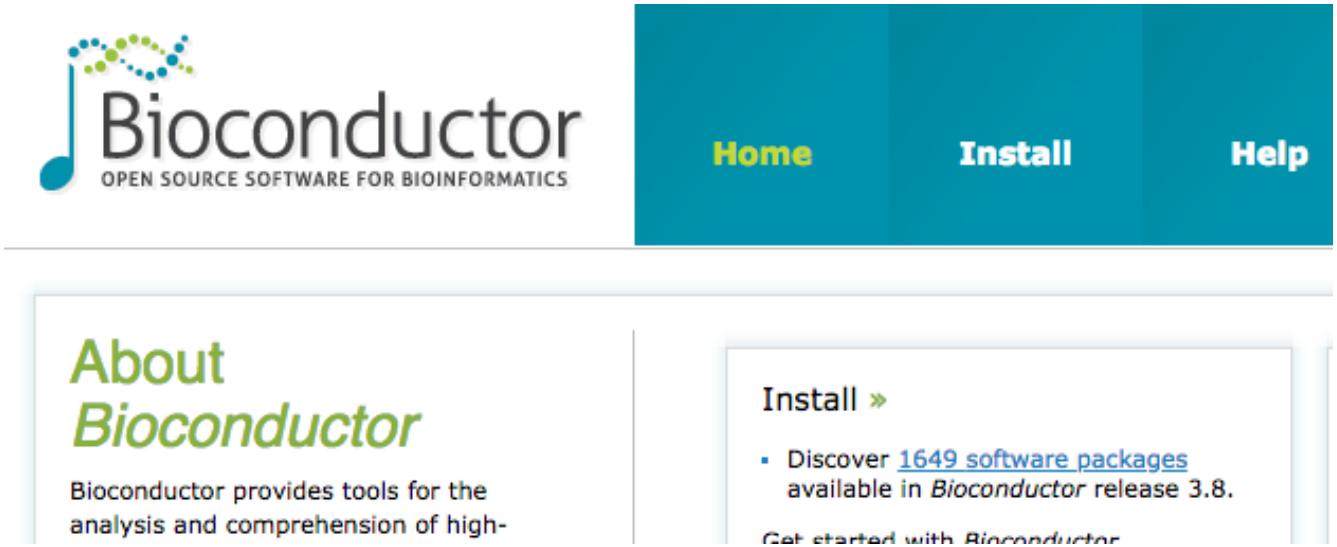
Dados de RNA seq têm algumas etapas distintas de processamento pois são contagens absolutas (que têm que ser normalizadas pelo nº de total)

Ferramenta : R/ Bioconductor

Bioconductor: conjunto de *packages* do R para análise e compreensão de dados genómicos.

Inúmeros packages para análise de expressão genética

www.bioconductor.org



Ferramenta : R/ Bioconductor: instalação

```
if (!requireNamespace("BiocManager", quietly = TRUE) )
    install.packages("BiocManager")
BiocManager::install(version = "3.12")


BiocManager::install(c("genefilter", "limma"))
BiocManager::install(c("ALL"))
BiocManager::install(c("GEOquery"))
```

Instalação dos packages de base

Instalação de packages específicos

Representação de dados no Bioconductor

Um dos objetos base para representar dados de expressão no Bioconductor é o ***ExpressionSet***, que é constituído por diversas componentes:

- **Dados de expressão**: uma matriz numérica que guarda os valores numéricos de expressão (genes nas linhas, amostras nas colunas): consultado com a função ***exprs***
- **Nomes dos genes** (ou features): vetor de strings que pode ser consultado com a função ***featureNames***
- **Nomes das amostras**: vetor de strings que pode ser consultado com a função ***sampleNames***
- Lista de **atributos** usado na caracterização das amostras (metadata): consultado com a função ***varMetadata***, que retorna um data frame com dois campos: o nome e a descrição de cada atributo
- **Valores dos atributos** de metadata para cada amostra: um objeto do tipo data frame que pode ser obtido com a função ***phenoData***
- Informação sobre a experiência, obtida com a função ***experimentData***
- Biblioteca a usar na anotação dos genes (i.e. ligação probe-gene), obtido pela função ***annotation***

Caso de estudo - ALL

Acute lymphoblastic leukemia (*ALL*) : é o tipo de leucemia mais comum em crianças.

ALL: informação recolhida de 128 pacientes com leucemia; 12625 genes

- 95 exemplos referentes a células B (linfócitos B)
- 33 exemplos referentes a células T (linfócitos T)
- Dados fenótipo dos pacientes: idade, sexo, dados analíticos, testes biologia molecular, etc

```
> library(ALL)
> data(ALL)
```

Exemplo: Estrutura do ExpressionSet

```
> library(ALL)
> data(ALL)
> ALL
ExpressionSet (storageMode: lockedEnvironment)
assayData: 12625 features, 128 samples
  element names: exprs
protocolData: none
phenoData
  sampleNames: 01005 01010 ... LAL4 (128 total)
  varLabels: cod diagnosis ... date last seen (21 total)
  varMetadata: labelDescription
featureData: none
experimentData: use 'experimentData(object)'
  pubMedIds: 14684422 16243790
Annotation: hgu95av2
> dim(ALL)
Features  Samples
   12625      128
```

Carregar o conjunto de dados e verificar o seu conteúdo e dimensão

Exemplo: Estrutura do ExpressionSet

```
> exp = exprs(ALL)
> dim(exp)
[1] 12625   128
> class(exp)
[1] "matrix”
> exp[1,1:5]
   01005    01010    03002    04006    04007
7.597323 7.479445 7.567593 7.384684 7.905312
> sampleNames(ALL)[1:5]
[1] "01005" "01010" "03002" "04006" "04007"
> featureNames(ALL)[1:5]
[1] "1000_at"   "1001_at"   "1002_f_at" "1003_s_at" "1004_at”
> varMetadata(ALL)
                            labelDescription
cod                                Patient ID
diagnosis                   Date of diagnosis
sex                       Gender of the patient
age                 Age of the patient at entry
…
> ALL$sex
  [1] M    M    F    M    M    M    F …
```

exp – matriz de dados de expressão

Nomes das amostras

Nomes das features

Atributos dos metadados das amostras

Valores da variável “sex” dos metadados para cada amostra

Exemplo: Estrutura do ExpressionSet

```
> annotation(ALL)
[1] "hgu95av2"

> experimentData(ALL)
Experiment data
  Experimenter name: Chiaretti et al.
  Laboratory …
 PMIDs: 14684422 16243790

> abstract(ALL)
[1] "Gene expression profiles were examined in 33 adult
patients with T-cell acute lymphocytic leukemia (T-ALL).
…
```

Biblioteca de anotação – serve para saber que genes estão associados a cada probe

Informação sobre a experiência

Exemplo: acesso a dados e indexação do *ExpressionSet*

```
> subALL = ALL [2:4,4:7]
> subALL
ExpressionSet (storageMode: lockedEnvironment)
assayData: 3 features, 4 samples
…
> females = ALL [, ALL$sex == "F"]
> females
ExpressionSet (storageMode: lockedEnvironment)
assayData: 12625 features, 45 samples
…
> anyB = grep("^B", ALL$BT)
> anyB
> ALL[,anyB]
ExpressionSet (storageMode: lockedEnvironment)
assayData: 12625 features, 95 samples

> exp= exprs(ALL)
> exp[2:3,4:5]
              04006    04007
1001_at   4.922627 4.844565
1002_f_at 4.206798 3.416923
```

Considerar 4ª a 7ª amostras e 2º a 4º genes; manter toda a informação de metadados para as amostras

Considerar apenas amostras cuja variável “sex” == “F”; resultado: ExpressionSet

Considerar apenas amostras cuja cujo campo BT inicia pela letra B

Valores de expressão dos genes 2 e 3 para as amostras 4 e 5; resultado é uma matriz

Pré-processamento dos dados

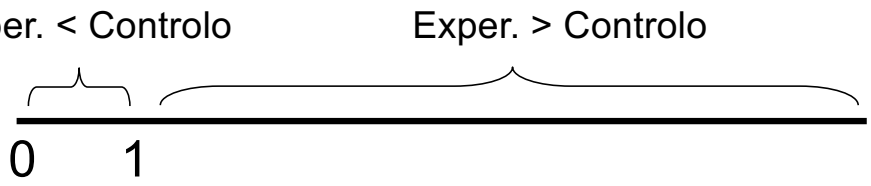
Mesmo quando já temos os dados na forma de uma matriz genes x amostras há necessidade de algumas tarefas adicionais de pré-processamento para várias tarefas de análise de dados

Passos do pré-processamento a este nível incluem tratamento de valores omissos, normalização, filtros de *flat patterns*, transformações logarítmicas, etc.

Estes passos permitem que os dados sejam de dimensões mais tratáveis e que estejam em gamas de valores adequadas para análise posterior

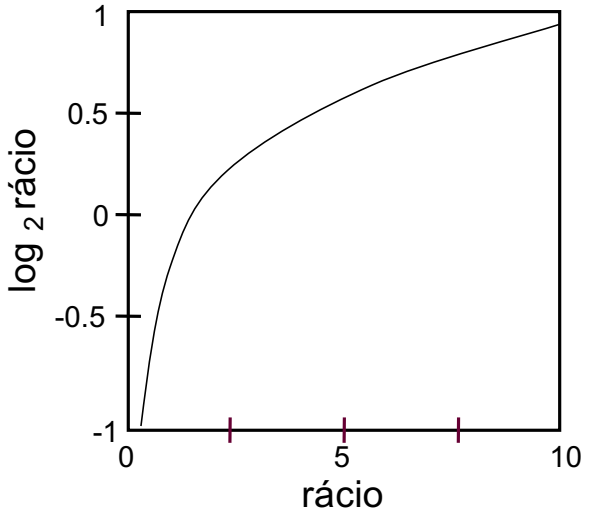
Transformação logarítmica

Rácio = Nível RNA experiência / Nível RNA controlo



Log base 2:

(…)
Rácio ½– log = -1
Rácio 1 - log = 0
Rácio 2– log = 1
Rácio 4 – log = 2
(…)



|log2(0.01)| == log2(100)

Filtros de “flat patterns”

Genes com valores muito constantes normalmente não trazem informação relevante

COMO DETECTAR ?

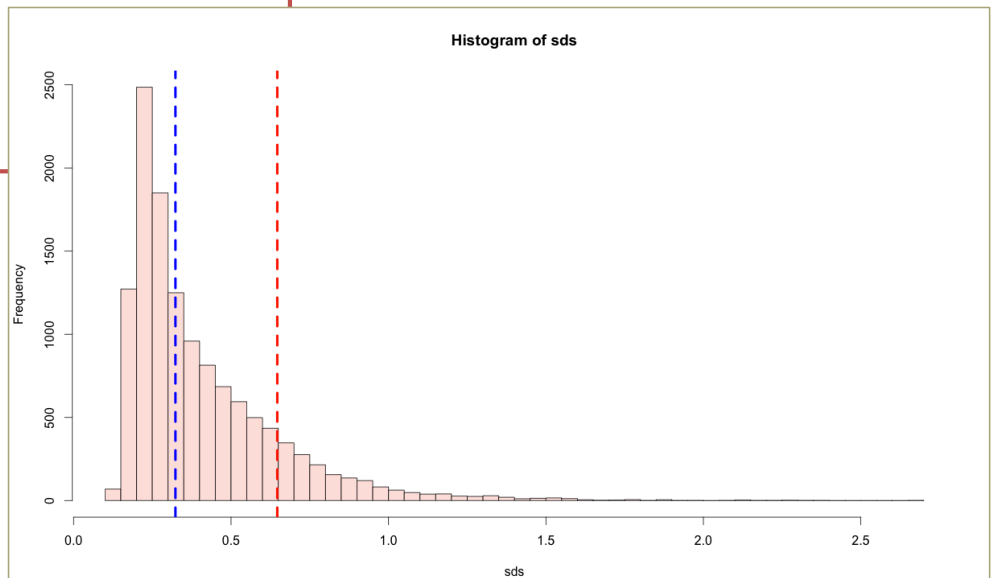
Desvio padrão/ desvio absoluto médio / IQR
Valor dos picos mínimos e máximos (e.g. max/min)

SOLUÇÃO
Remover estes genes da análise
Remover apenas os que têm valores médios próximos de 0

Filtros flat patterns - exemplos

```
> library(genefilter)
> exp = exprs(ALL)
> sds = rowSds(exp)
> m = median(sds)
> hist(sds, breaks = 50, col = "mistyrose")
> abline(v=m, col="blue",lwd = 4,lty = 2)
> abline(v=m*2, col="red",lwd = 4,lty = 2)
> ALLr = ALL[sds >= 3*median(sds), ]
> ALLr
ExpressionSet
assayData: 430 features,
 128 samples
```

Filtra genes cujo desvio padrão dos valores de expressão é maior do que duas vezes a mediana dos desvios padrão de todos os genes



Filtros flat patterns - exemplos

```
> maximos = apply(exp,1,max)
> minimos = apply(exp,1,min)
> vl = maximos/minimos > 2
> ALLm2=ALL[vl,]
> ALLm2
ExpressionSet (storageMode: lockedEnvironment)
assayData: 1023 features, 128 samples
…
```

Filtra genes cujo rácio do máximo valor sobre o mínimo valor de expressão seja superior a 2

Outros filtros: ver função **nsFilter**

Standardização – entre diferentes amostras

Tipicamente, os valores são re-escalonadas para ficarem dentro de limites normalizados (distribuição normal com média 0 e variância 1)

Como:

- Subtrair o valor de cada gene pela média (ou mediana) dos valores de expressão
- Dividir pelo desvio padrão da amostra (ou desvio absoluto médio)
- Por vezes faz-se apenas a primeira

```
> exp_m2 = exprs(ALLm2)
> exprs(ALLm2) = scale(exp_m2)
```

Exemplo: conjunto de dados do GEO

```
> library(GEOquery)
> gds = getGEO('GDS3257', destdir=".")
> data = GDS2eSet(gds)
> dim(data)
Features    Samples
22283   107
> class(data)
[1] “ExpressionSet”
> exp = exprs(data)
> write.csv(exp, "gds3257.csv")
> meta = phenoData(data)
> varLabels(meta)
> data$disease.state
> dim(meta)
> m = slot(meta, “data”)
> write.csv(m, “meta-gds3257.csv”)
```

Leitura diretamente da base de dados GEO

Exportação de dados e metadados para ficheiros CSV

Cigarette smoker & lung cancer

Análise de adenocarcinomas pulmonares em diferentes estadios; dados emparelhados normal vs tumor; dados de fumadores e não fumadores. Total de 107 amostras; 22283 genes. Metadados incluem 7 variáveis discretas.

Landi MT, Dracheva T, Rotunno M, Figueroa JD et al. Gene expression signature of cigarette smoking and its role in lung adenocarcinoma development and survival. *PLoS One* 2008 Feb 20;3(2):e1651.

Exemplo: conjunto de dados cachexia (metabolómica)

```
> download.file("http://darwin.di.uminho.pt/dataAnalysisR/cachexia.csv","cachexia.csv")
> download.file("http://darwin.di.uminho.pt/dataAnalysisR/meta_cachexia.csv","meta.csv")

> cachexia = read.table("cachexia.csv", sep = ",", header = T, row.names = 1)
> dim(cachexia)
[1] 77 63
> meta_cachexia = read.table("meta.csv", sep = ",", header = T, row.names = 1)
> dim(meta_cachexia)
[1] 77  1
> cachexia[1:5,1:5]
     X1.6.Anhydro.beta.D.glucose X1.Methylnicotinamide X2.Aminobutyrate ...
PIF_178                   40.85                 65.37            18.73     ...
> meta_cachexia
…
```

Cachexia

Concentrações de 63 metabolitos em 77 amostras de urina de doentes de cancro, medidos com RMN 1H (Eisner et al, 2011). Metadados definem os dois grupos: controlo e caquexia.

Eisner et al. , 2011. Learning to predict cancer-associated skeletal muscle wasting from 1H-NMR profiles of urinary metabolites. Metabolomics 7(1), March 2011, pp. 25-34-

Exercício: verificar estrutura dos dados - cachexia

- Verificar tipos de dados de cada variável (compostos)
- Verificar o tipo de dados da variável de metadados e alterar se necessário
- Verificar nomes dos compostos e identificadores das amostras (pacientes)
- Alterar nome do 1º composto para “One-six-Anhydro-beta-D-glucose“
- Verificar se há valores omissos nos dados

Exercício: verificar estrutura dos dados - cachexia

```
> unlist(lapply(cachexia, class))
X1.6.Anhydro.beta.D.glucose       X1.Methylnicotinamide        X2.Aminobutyrate      ...
             "numeric"                   "numeric"                   "numeric"      ...
tau.Methylhistidine …
   "numeric"
> class(meta_cachexia[,1])
[1] "character“
> meta_cachexia[,1] = as.factor(meta_cachexia[,1])
> meta_cachexia[,1]
…
> names(cachexia)
[1] "X1.6.Anhydro.beta.D.glucose" "X1.Methylnicotinamide" "X2.Aminobutyrate"
"X2.Hydroxyisobutyrate“
[5] "X2.Oxoglutarate" "X3.Aminoisobutyrate" "X3.Hydroxybutyrate"
"X3.Hydroxyisovalerate"
[9] "X3.Indoxylsulfate" "X4.Hydroxyphenylacetate"   "Acetate"    "Acetone“   ...
> row.names(cachexia)
[1] "PIF_178"      "PIF_087"      "PIF_090"      "NETL_005_V1“ ...

> names(cachexia)[1] = "One-six-Anhydro-beta-D-glucose"
```

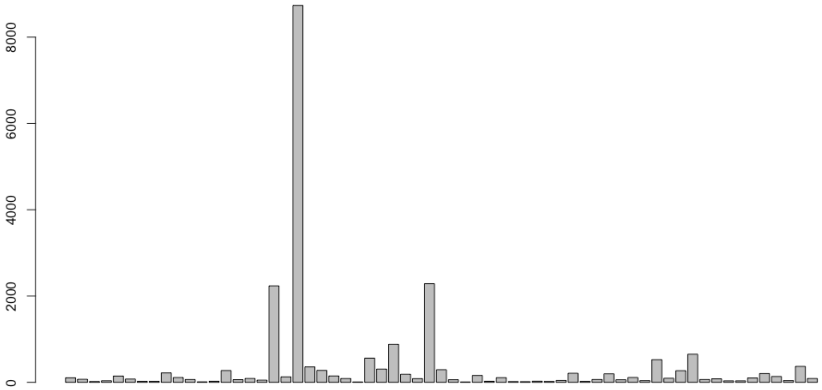
```
> sum(unlist(lapply(lapply(cachexia, is.na), sum)))
[1] 0
```

Exercício: transformação e standardização de dados

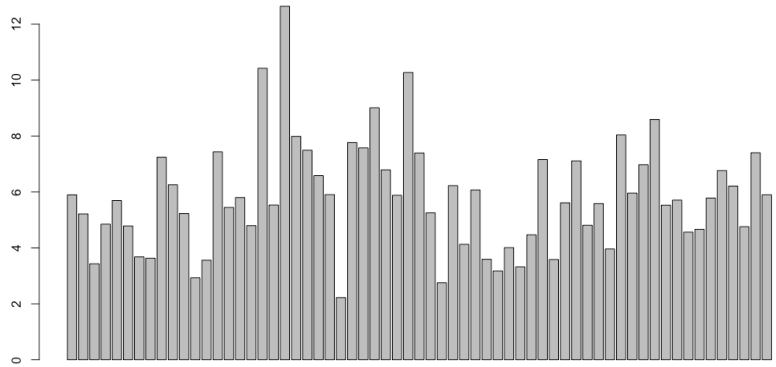
- Realizar uma transformação logarítmica sobre os dados dos compostos (base 2) e verifique o efeito da transformação sobre as médias de cada variável (incluindo de forma gráfica)
- Realizar a standardização dos dados dos compostos; verificar a média e desvio padrão para cada variável
- Comparar para o composto “Glucose” as distribuições de valores nos 3 datasets usando boxplots

Exercício: transformação e standardização de dados

```
> cachexia.log = log2(cachexia)
> colMeans(cachexia)
…
> colMeans(cachexia.log)
…
> barplot(colMeans(cachexia), names.arg = "")
> barplot(colMeans(cachexia.log), names.arg = "")
```




```
"log_norm" = function(x, min.val) {
   log2((x + sqrt(x^2 + min.val^2))/2)
}
```



Função alternativa
(permite zeros)

Exercício: transformação e standardização de dados

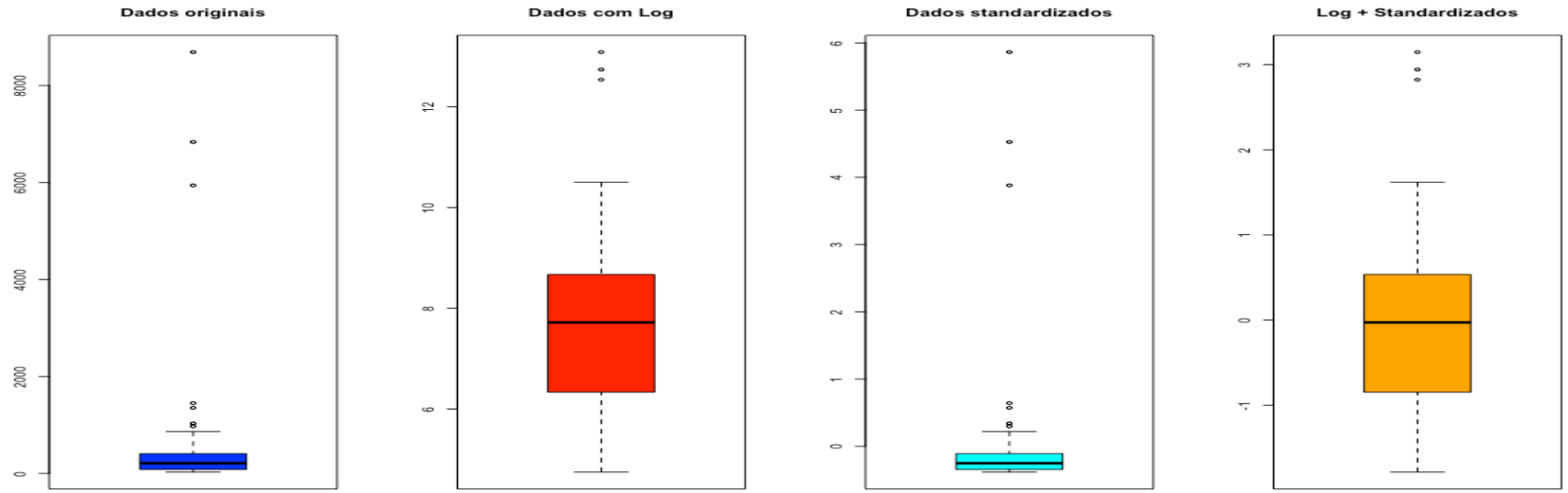
```
> cachexia_sc = as.data.frame(scale(cachexia))

> sapply(cachexia_sc, mean)
…
> sapply(cachexia_sc, sd)
…
> all (abs(sapply(cachexia_sc, mean)) < 0.0001 )
[1] TRUE
> all (abs(sapply(cachexia_sc, sd) -1) < 0.0001 )
[1] TRUE
```

Exercício: transformação e standardização de dados

```
cachexia.log.sc = as.data.frame(scale(cachexia.log))

par(mfrow = c(2,2))
boxplot(cachexia[,"Glucose"], main = "Dados originais", col = "blue")
boxplot(cachexia.log[,"Glucose"], main = "Dados com Log", col = "red")
boxplot(cachexia_sc[,"Glucose"], main = "Dados standardizados", col = "cyan")
boxplot(cachexia.log.sc[,"Glucose"], main = "Log + Standardizados", col = "orange")
par(mfrow = c(1,1))
```

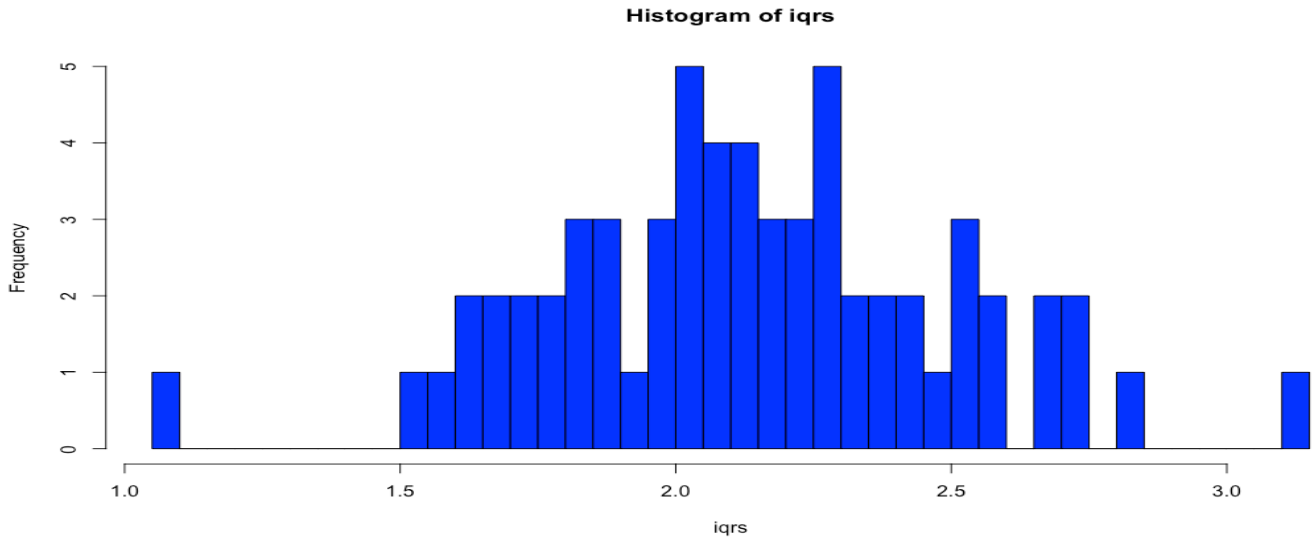


Exercício: identificar variabilidade

- Identificar qual o compostos que varia mais, tomando como métrica o IQR (use os dados após transformação logarítmica)
- Realizar um histograma com os valores do IQR
- Identificar os compostos que têm um IQR acima da sua mediana e filtrar o dataset para considerar apenas esses

Exercício: transformação e standardização de dados

```
iqrs = sapply(cachexia.log, IQR)
which.max(iqrs)
hist(iqrs, breaks = 30, col = "blue")
median(iqrs)
which(iqrs>median(iqrs))
cachexia.log.filt = cachexia.log[, which(iqrs>median(iqrs))]
```



Exercício: correlações

- Calcular as correlações entre o composto “Glucose” e todos os outros compostos
- Qual o composto mais correlacionado com a “Glucose”
- Represente na forma de um scatter plot a relação entre as duas variáveis, colorindo os pontos com a variável dos meta-dados

Exercício: correlações

```
> corr_glucose = function (x) { cor(cachexia[,"Glucose"], x) }
> corrs = sapply(cachexia, corr_glucose)
> sort(corrs, decreasing = TRUE)
> plot(cachexia[,"Glucose"], cachexia[,"Lactate"], pch = 19,
     xlab = "Glucose", ylab = "Lactate", col = meta_cachexia$Muscle.loss)
> legend(0, 3000, legend=levels(meta_cachexia$Muscle.loss), col=c("black","red"),
    pch=c(19,19))
```