Grupos de genes

Guillermo Ayala Gallego

3/16/23

Table of contents

GSEABase::GeneSet

- La opción más simple sería definirnos nuestro propio conjunto de genes.

```
data(gse20986,package="tamidata")
```

- Supongamos que queremos construir un conjunto de genes formado por aquellos que ocupan las filas de la 345 a la 405.

```
pacman::p_load(GSEABase)
(egs = GeneSet(gse20986[345:405, ], setName = "Burjasot"))
```

...

- Los identificadores son

```
head(geneIds(egs))
```

```
[1] "1552739_s_at" "1552740_at"   "1552742_at"   "1552743_at"   "1552745_at"
[6] "1552747_a_at"
```

- Podemos tener más información con

```
details(egs)
```

```
setName: Burjasot
geneIds: 1552739_s_at, 1552740_at, ..., 1552822_at (total: 61)
geneIdType: Annotation (hgu133plus2)
collectionType: ExpressionSet
setIdentifier: debian:38393:Thu Mar 16 14:02:49 2023:1
description:
organism: Homo sapiens
pubMedIds:
urls:
contributor:
setVersion: 0.0.1
creationDate:
```

- ¿Y la correspondencia con otros identificadores?

```
mapIdentifiers(egs, EntrezIdentifier(), verbose = TRUE)
```

GSEABase::GeneSetCollection()

- Definimos una colección de grupos definida utilizando GO.

```
gsc = GeneSetCollection(gse20986, setType = GOCollection())
```

- El conjunto con identificador GO “GO:0000122” es

```
gsc[["GO:0000122"]]
```

```
setName: GO:0000122
geneIds: 1316_at, 1552338_at, ..., AFFX-HUMISGF3A/M97935_MB_at (total: 2280)
geneIdType: Annotation (hgu133plus2)
collectionType: GO
  ids: GO:0000122 (1 total)
  evidenceCode: EXP IDA IPI IMP IGI IEP HTP HDA HMP HGI HEP ISS ISO ISA ISM IGC IBA IBD IKR
  ontology: CC MF BP
details: use 'details(object)'
```

...

- Podemos convertir estos mismos conjuntos a identificadores ENTREZ.

```
gsc = mapIdentifiers(gsc, EntrezIdentifier())
```

- Por ejemplo

```
gsc[["GO:0000122"]]
```

```
setName: GO:0000122
geneIds: 7067, 145258, ..., 148979 (total: 859)
geneIdType: EntrezId (hgu133plus2)
collectionType: GO
  ids: GO:0000122 (1 total)
  evidenceCode: EXP IDA IPI IMP IGI IEP HTP HDA HMP HGI HEP ISS ISO ISA ISM IGC IBA IBD IKR
  ontology: CC MF BP
details: use 'details(object)'
```

- Los identificadores son

```
head(geneIds(gsc[["GO:0000122"]]))
```

```
[1] "7067"   "145258" "2117"   "80712"  "201163" "11060"
```

...

- ¿Cuántos elementos tiene cada uno de los conjuntos que hemos construido?

```
head(sapply(geneIds(gsc), length))
```

```
GO:0000002 GO:0000003 GO:0000012 GO:0000017 GO:0000018 GO:0000019
        11          3         10          2          6          2
```

- Podemos quedarnos con los dos grupos que tengan un cardinal mínimo, por ejemplo, por encima de 10.

```
gsc.filt = gsc[sapply(geneIds(gsc), length) > 10]
```

EnrichmentBrowser::get.kegg.genesets()

- Podemos bajarnos todos las rutas de **KEGG** para un organismo dado.

```
pacman::p_load(EnrichmentBrowser)
```

- Para humanos.

```
hsaKEGGgsc=get.kegg.genesets("hsa")
```

¿Qué tenemos?

```
class(hsaKEGGgsc)
```

```
[1] "list"
```

Podemos ver sus nombres.

```
head(names(hsaKEGGgsc))
```

```
[1] "hsa01100_Metabolic_pathways"
[2] "hsa01200_Carbon_metabolism"
[3] "hsa01210_2-Oxocarboxylic_acid_metabolism"
[4] "hsa01212_Fatty_acid_metabolism"
[5] "hsa01230_Biosynthesis_of_amino_acids"
[6] "hsa01232_Nucleotide_metabolism"
```

Y los genes que componen uno determinado con su código ENTREZ.

```
hsaKEGGgsc[[3]]
```

```
character(0)
```

o bien con

```
hsaKEGGgsc$"hsa00030_Pentose_phosphate_pathway"
```

```
character(0)
```

```
hsaKEGGgsc[["hsa00030_Pentose_phosphate_pathway"]]
```

```
character(0)
```

EnrichmentBrowser::get.go.genesets

Vamos a trabajar con la Drosophila Melanogaster y con los grupos definidos por procesos biológicos en Gene Ontology.

```
dmeGOBPgsc = get.go.genesets(org="dme", onto="BP", mode="GO.db")
```

```
dmeGOBPgsc[[1]]
```

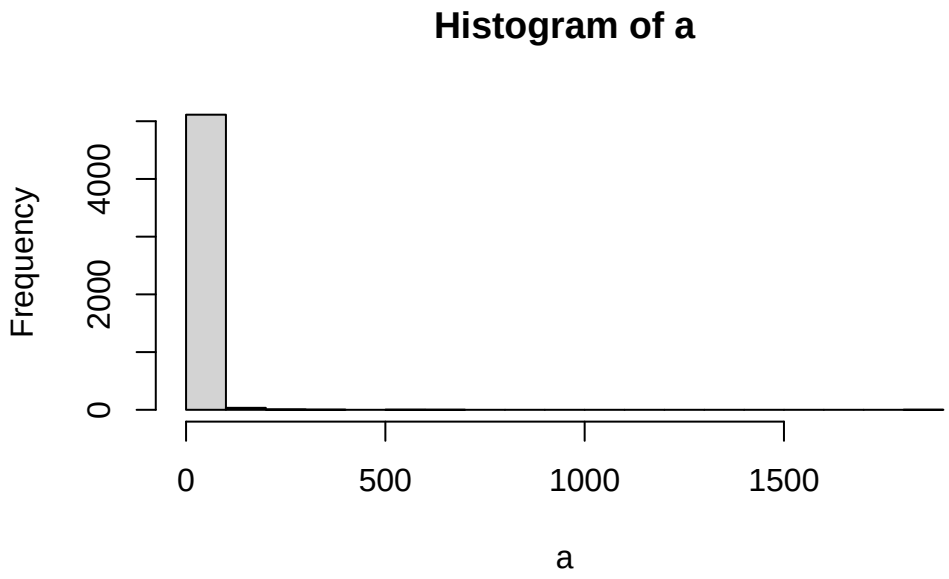
```
[1] "31607" "34792" "36309" "53567"
```

¿Qué tamaños tienen los grupos?

```
a = sapply(dmeGOBPgsc,length)
summary(a)
```

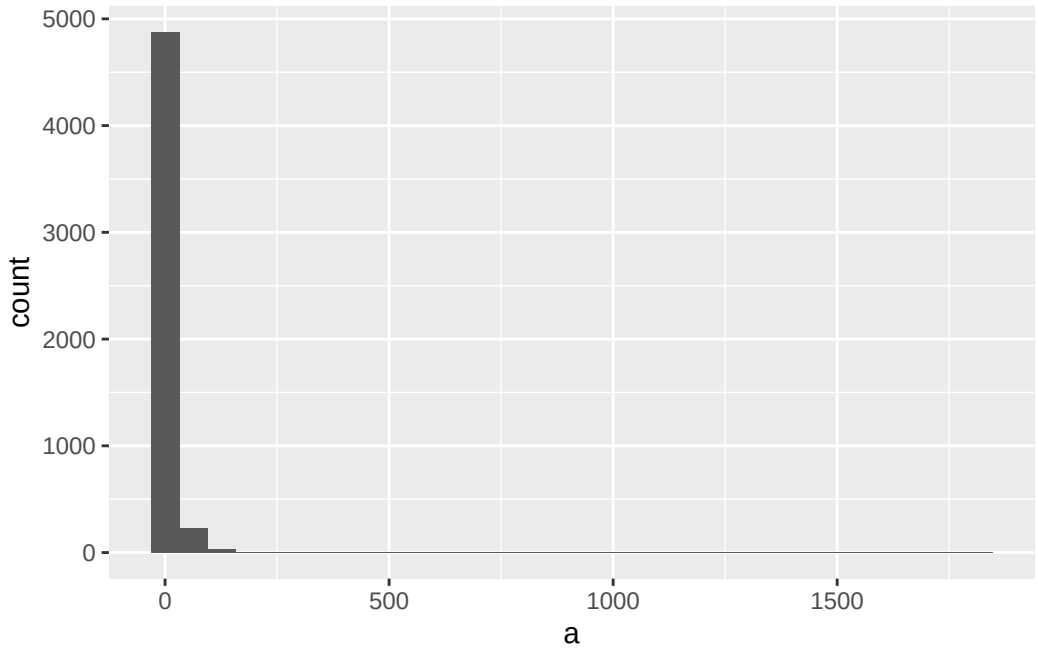
```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  1.000   1.000   3.000   9.456   8.000 1816.000 
```

```
hist(a)
```



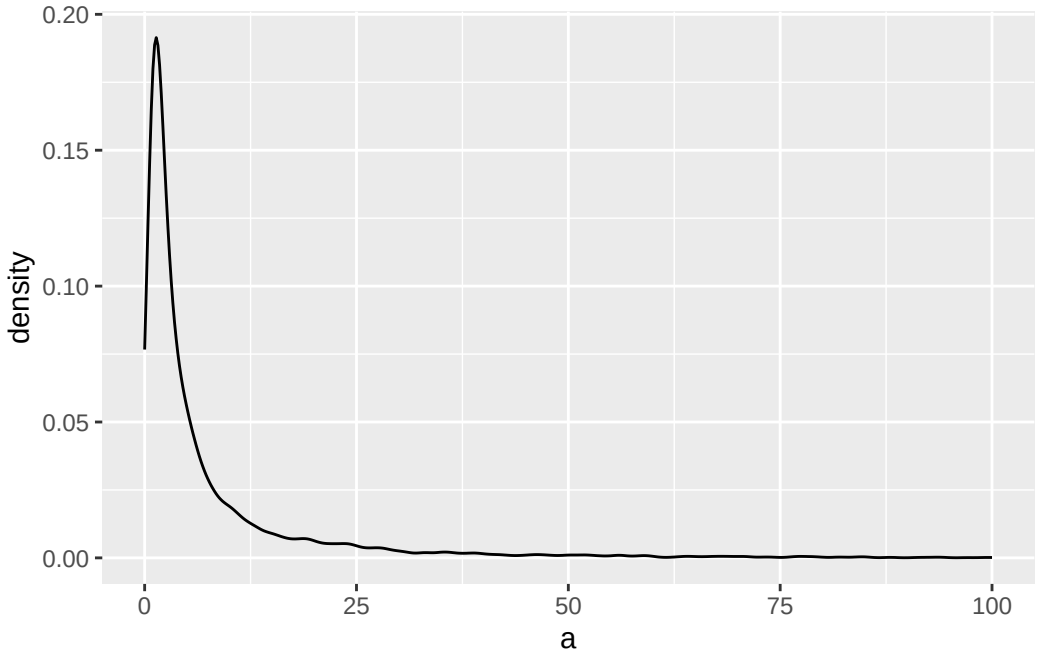

```
pacman::p_load(ggplot2)
df = data.frame(a)
ggplot(df,aes(x=a)) + geom_histogram()
```

```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
p1 = ggplot(df,aes(x=a)) + geom_density()
p1 + xlim(0,100)
```

```
Warning: Removed 48 rows containing non-finite values (`stat_density()`).
```


0.20
0.15
0.10
0.05
0.00
density
0
25
50
75
100
a