bioinformatics.ca

Canadian Bioinformatics Workshops

www.bioinformatics.ca

Creative Commons

This page is available in the following languages:
Afrikaans български Català Dansk Deutsch Ελληνικά English English (CA) English (GB) English (US) Esperanto Castellano Castellano (AR) Español (CL) Castellano (CO) Español (Ecuador) Castellano (MX) Castellano (PE) Euskara Suomeksi français français (CA) Galego עברית hrvatski Magyar Italiano 日本語 한국어 Macedonian Melayu Nederlands Norsk Sesotho sa Leboa polski Português română slovenski jezik српски srpski (latinica) Sotho svenska 中文 漢語 (台灣) isiZulu

creative commons

Attribution-Share Alike 2.5 Canada

You are free:

to Share — to copy, distribute and transmit the work

to Remix — to adapt the work

Under the following conditions:

Attribution. You must attribute the work in the manner specified by the author or licensor (but not in any way that suggests that they endorse you or your use of the work).

Share Alike. If you alter, transform, or build upon this work, you may distribute the resulting work only under the same or similar licence to this one.

- For any reuse or distribution, you must make clear to others the licence terms of this work.
- Any of the above conditions can be waived if you get permission from the copyright holder.
- The author's moral rights are retained in this licence.

Disclaimer

Your fair dealing and other rights are in no way affected by the above.
This is a human-readable summary of the Legal Code (the full licence) available in the following languages:

Module 2 -

Exploring FCM data in R

Radina Droumeva

Bioinformatics Specialist, BC Cancer Agency
Vancouver, British Columbia, Canada

CBW: Flow Cytometry Data Analysis using R
June 17, 2013

Module 2: Exploring FCM data in R

Module Overview:

- Read in FCS files
- Explore data annotation and structure
- Visualize FCM data in R

R Refresher: a vector and a matrix

Note: some output is not shown to save space. Run the commands to see the output.

```r
# Variable Assignment
x ← 5
y ← 10
x + y

# Vectors (one dimensional)
x ← c(1, 4, 5)
y ← seq(from = 1, to = 3, by = 1)
y
x + y

# Get help on how to construct a matrix (two dimensional)
?matrix
A ← matrix(c(1, 2, 3,  11, 12, 13), nrow = 2, ncol = 3,
    byrow = TRUE)
A
rbind(x, y)
A + rbind(x, y) # see ?rbind for explanation
A[1, 3] # Access entry in first row, third column
A[2, ] # Access all entries in second row
A[, 1] # All entries in first column
```

R Refresher: A list with named entries

```
# Lists and names
mylist ← list(‘first‘ = x, ‘second‘ = y)

# Extract the first entry in the list by index (1) or name ("first"):
mylist[[1]]
[1] 1 4 5
mylist[["first"]]
[1] 1 4 5

# Checking the size of objects
# Vectors and lists have ’length’s, matrices have ’dim’ensions
length(mylist)
[1] 2
length(x)
[1] 3
dim(A)
[1] 2 3
length(A)
[1] 6
```

R Refresher: which, intersect, union

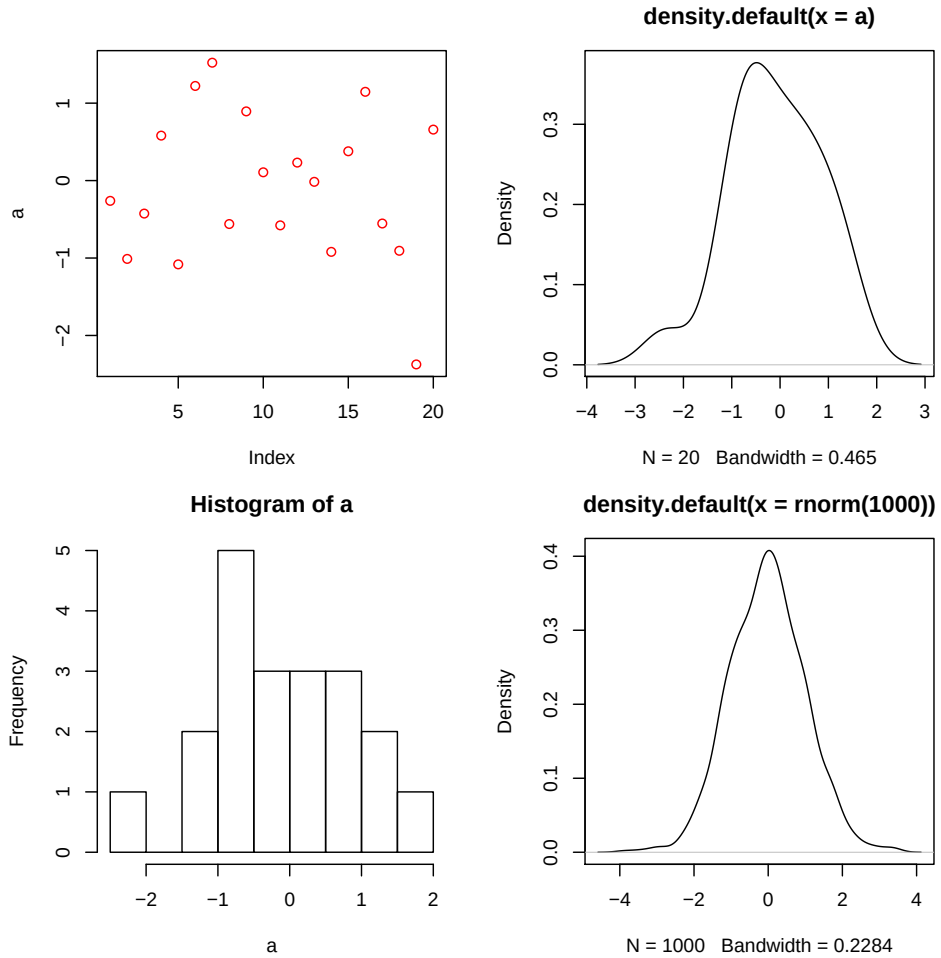
```
# Generate 20 random numbers from a normal distribution
?rnorm
a ← rnorm(20)
a
```

```
[1]  0.16215786  0.42892603 -0.40585986 -0.41403764  1.08375740 -0.16635696 -0.11281127
-1.97296631 -1.43725793 -0.61669020 0.92111531 -1.76906195 -0.23469280 -0.28725082
-1.68470666 0.86727154  1.75077450  0.08697872 -1.47718350 -0.03710011
```

```
# Identify the indices of 'a' > 0, then values < -1
which(a > 0)
[1]  1  2  5 11 16 17 18
which(a < -1)
[1]  8  9 12 15 19

# Check that their intersection is empty, take union
intersect(which(a > 0), which(a < -1))
[1] integer(0) # empty intersection!
combined ← union(which(a > 0), which(a < -1))
combined
[1]  1  2  5 11 16 17 18  8  9 12 15 19
```

R Refresher: dot, density, histogram plot




```
# Simple plotting functionality
plot(a, col = "red")
plot(density(a))
hist(a)
plot(density(rnorm(1000)))
```

FCM data: reading FCS files in R

```
# Load the package for reading FCM data
library(flowCore)
# Make sure R knows which directory our data is in
setwd('/home/rguru/Documents/Workshop/data')
f ← read.FCS('fullFCS/100715.fcs')
f
```

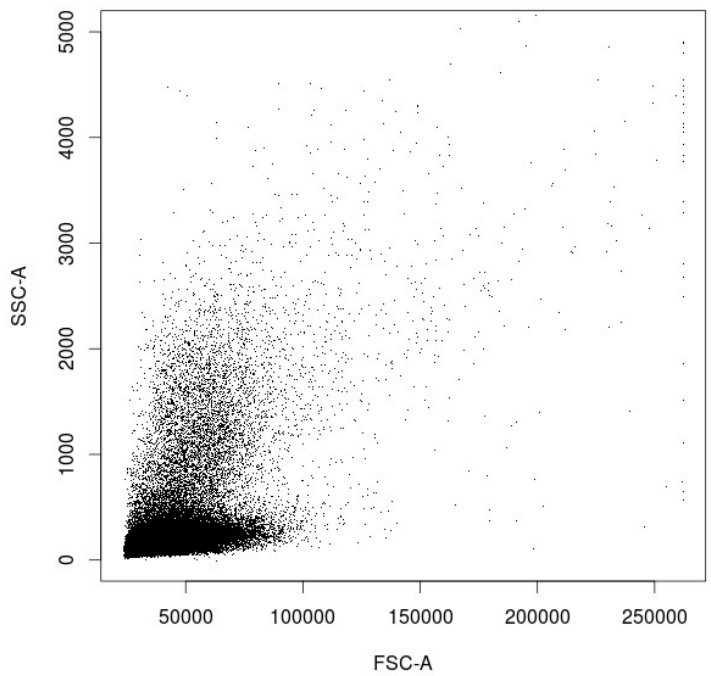
```
flowFrame object '0d8e743a-05fe-4e8b-9ec4-25993c124ee2'
with 65016 cells and 16 observables:
       name          desc  range     minRange maxRange
$P1   FSC-A              262207 23406.000000   262206
$P2   FSC-H              262207 27008.500000   262206
$P3   SSC-A              261588    -8.014621   261587
$P4  B515-A         KI67 261588   -67.282539   261587
$P5  R780-A          CD3 261588   -67.119034   261587
$P6  R710-A         CD28 261588   -44.558552   261587
$P7  R660-A       CD45RO 261588   -79.819801   261587
$P8  V800-A          CD8 261588  -110.409325   261587
$P9  V655-A          CD4 261588   -66.276711   261587
$P10 V585-A         CD57 261588  -110.472748   261587
$P11 V450-A VIVID / CD14 261588   -28.876564   261587
$P12 G780-A         CCR5 261588  -110.527817   261587
$P13 G710-A         CD19 261588   -89.503387   261587
$P14 G660-A         CD27 261588   -51.957710   261587
$P15 G610-A         CCR7 261588   -61.939350   261587
$P16 G560-A        CD127 261588   -33.256630   261587
253 keywords are stored in the 'description' slot
```

FCM data: exploring FCS meta data

```
# 'f' is a flowFrame object which comes with meta data.
# The channel labels:
colnames(f)
# The expression values measured for each cell:
E ← exprs(f)
dim(E)
# Look at first 10 rows (cells) of E
E[1:10, ]

# Study all the 'keyword' information stored in the file
f@description
# Access each 'keyword' one at a time
f@description$`TUBE NAME`
# Access parameter information such as range of expression
    values of FSC-A:
f@parameters@data
f@parameters@data[1, c("minRange", "maxRange")]
    minRange maxRange
$P1    23406   262206
```

Plotting FCM data in R




```
# First load the library which helps plot FCM data
library(flowViz)
plot(f, c("FSC-A", "SSC-A"), ylim = c(0,5000), smooth=FALSE)
# Note SSC-A is the third parameter (P3) and the meta data
    tells us it is to be viewed on a LOG scale:
colnames(f)[3]
[1] "SSC-A"
f@description$‘P3DISPLAY‘
[1] "LOG"
```

Reading multiple FCS files

```
# 'f' above is a flowFrame object.
# A set of flowFrame objects is a flowSet object:
fs ← read.flowSet(path = 'fullFCS', pattern = ".fcs")
fs
A flowSet with 3 experiments.
  column names:
  FSC-A FSC-H SSC-A B515-A R780-A R710-A R660-A V800-A V655-
      A V585-A V450-A G780-A  G710-A G660-A G610-A G560-A

# Explore the meta data for the flowSet:
sampleNames(fs)
[1] "100715.fcs" "109567.fcs" "113548.fcs"
length(fs)
[1] 3

# A flowSet object is closely related to a list
# You can access the first frame by index or name
fs[["100715.fcs"]]
fs[[1]]
```

Using 'fsApply'

```
# Use fsApply to get cell counts for all samples
nrow(fs[[1]])
[1] 65016
fsApply(fs, nrow)
               [,1]
100715.fcs  65016
109567.fcs 160074
113548.fcs 177102

# Use 'fsApply' to extract the TUBE NAME keyword
# This is useful in identifying control and stained, or
    stimulated and unstimulated samples in large data sets
fsApply(fs, function(f) f@description$`TUBE NAME`)
             [,1]
100715.fcs "Tube_025"
109567.fcs "Tube_017"
113548.fcs "Tube_003"
```

LUNCH 1hr