Xeva Tutorial

Arvind Mer

2017-01-25

Load Xeva library and KRAS/P53 PDX data

```
library(Xeva)
data(lpdx)
```

To see all the model.id

```
lpdx.mod = modelInfo(lpdx)
head(lpdx.mod$model.id)
```

```
## [1] "PHLC1106_P5.501.A1" "PHLC1106_P5.504.A4" "PHLC1106_P5.506.B1"
## [4] "PHLC1106_P5.507.B2" "PHLC1106_P5.508.B3" "PHLC1106_P5.511.C1"
```

To get the data for one model.id

```
modId = lpdx.mod$model.id[82]
df = getExperiment(lpdx, model.id = modId)
head(df)
```

```
##             model.id         drug.join.name time    volume width length
## 1 PHLC191_P5.503.A3 Vinorelbine+ Cisplatin    0  81.20558  5.18   5.82
## 2 PHLC191_P5.503.A3 Vinorelbine+ Cisplatin    8  93.24844  5.57   5.78
## 3 PHLC191_P5.503.A3 Vinorelbine+ Cisplatin   15  90.13298  5.16   6.51
## 4 PHLC191_P5.503.A3 Vinorelbine+ Cisplatin   19 213.92906  6.99   8.42
## 5 PHLC191_P5.503.A3 Vinorelbine+ Cisplatin   22 252.04349  7.43   8.78
## 6 PHLC191_P5.503.A3 Vinorelbine+ Cisplatin   26 375.84838  8.65   9.66
##   dose body.weight       date         comment volume.change
## 1  0.0      19.762 2014-09-25            <NA>       0.00000
## 2  0.0      20.424 2014-10-03    clip removed      14.83010
## 3  0.0      21.130 2014-10-10            <NA>      10.99359
## 4 75.4      21.103 2014-10-14 Start Treatment     163.44135
## 5 74.1      20.761 2014-10-17            <NA>     210.37708
## 6 72.1      20.178 2014-10-21            <NA>     362.83569
##   average.response
## 1         0.000000
## 2         7.415048
## 3         8.607894
## 4        47.316257
## 5        79.928421
## 6       127.079632
```

In the data.fram df you will see that for first 3 time points dose is 0, which indicate no treatment is given during this time. If you want the data only during the treatment periode specify treatment.only = TRUE

```r
df = getExperiment(lpdx, modId, treatment.only = TRUE)
head(df)
```

```
##            model.id        drug.join.name time   volume width length dose
## 4 PHLC191_P5.503.A3 Vinorelbine+ Cisplatin   19 213.9291  6.99   8.42 75.4
## 5 PHLC191_P5.503.A3 Vinorelbine+ Cisplatin   22 252.0435  7.43   8.78 74.1
## 6 PHLC191_P5.503.A3 Vinorelbine+ Cisplatin   26 375.8484  8.65   9.66 72.1
## 7 PHLC191_P5.503.A3 Vinorelbine+ Cisplatin   29 526.0954  9.40  11.45 73.3
## 8 PHLC191_P5.503.A3 Vinorelbine+ Cisplatin   33 683.3432 10.43  12.08 73.3
## 9 PHLC191_P5.503.A3 Vinorelbine+ Cisplatin   36 807.8725 10.97  12.91 75.9
##   body.weight       date         comment volume.change average.response
## 4      21.103 2014-10-14 Start Treatment      163.4413         47.31626
## 5      20.761 2014-10-17            <NA>      210.3771         79.92842
## 6      20.178 2014-10-21            <NA>      362.8357        127.07963
## 7      20.528 2014-10-24            <NA>      547.8563        187.19059
## 8      20.534 2014-10-28            <NA>      741.4979        256.47900
## 9      21.257 2014-10-31            <NA>      894.8486        327.40896
```

Models which are replicates are stored togather in expDesign slot. To get the data for all the replicates pass the ‘batch.name’ in the getExperiment function.

```r
print(batchNames(lpdx))
```

```
##  [1] "PHLC1106_P5" "PHLC111_P7"  "PHLC119_P5"  "PHLC153_P6"  "PHLC181_P7"
##  [6] "PHLC189_P5"  "PHLC191_P5"  "PHLC191_P7"  "PHLC196_P5"  "PHLC215_P5"
## [11] "PHLC229_P6"  "PHLC235_P4"  "PHLC655_P7"  "PHLC82_P5"
```

```r
df = getExperiment(lpdx, batch.name = batchNames(lpdx)[1], treatment.only = TRUE)
head(df)
```

```
##              model.id        drug.join.name time   volume width length
## 8  PHLC1106_P5.502.A2 Vinorelbine+ Cisplatin   39 167.5273  6.46   7.72
## 9  PHLC1106_P5.502.A2 Vinorelbine+ Cisplatin   40 172.7149  6.48   7.91
## 10 PHLC1106_P5.502.A2 Vinorelbine+ Cisplatin   43 164.4621  6.38   7.77
## 11 PHLC1106_P5.502.A2 Vinorelbine+ Cisplatin   47 187.4881  6.76   7.89
## 12 PHLC1106_P5.502.A2 Vinorelbine+ Cisplatin   50 264.0641  7.64   8.70
## 13 PHLC1106_P5.502.A2 Vinorelbine+ Cisplatin   54 174.6014  6.66   7.57
##    dose body.weight       date         comment volume.change
## 8  79.2      22.182 2014-12-08 Start Treatment      126.5905
## 9  78.8      22.051 2014-12-09            <NA>      133.6070
## 10 78.6      21.995 2014-12-12            <NA>      122.4445
## 11 78.0      21.827 2014-12-16            <NA>      153.5886
## 12 80.2      22.467 2014-12-19            <NA>      257.1621
## 13 79.2      22.185 2014-12-23            <NA>      136.1586
##    average.response  exp.type
## 8          47.16826 treatment
## 9          56.77256 treatment
## 10         63.33976 treatment
## 11         71.54420 treatment
## 12         87.01235 treatment
## 13         90.79283 treatment
```

Here the data.fram contaions an extra column 'exp.type' . This indicates if this is treatment or control.

To calculate angle between the treatment and control samples of this batch

```
batchNames <- batchNames(lpdx)
expDesign  <- expDesign(lpdx, batchNames[1])
#ang <- calculateAngle(lpdx, expDesign, treatment.only = TRUE, plot=TRUE)
#print(ang)

#par(mfrow=c(5,3))
for(I in batchNames)
{
  expDesign  <- expDesign(lpdx, I)
  #ang <- calculateAngle(lpdx, expDesign, treatment.only = TRUE, plot=TRUE)
#  print(ang)
}
```

Summarize Response of PDXs Get slop of each model and combine summarize all model slop which belongs to same patient by "mean"

```
lpdx_slop <- summarizeResponse(lpdx, response.measure = "slop",
                               group.by="patient.id", summary.stat = "mean")
```

Get angle between treatment and control model ids. For each batch it will give one angle value

```
lpdx_angle <- summarizeResponse(lpdx, response.measure = "angle")
```

```
## Warning in .summarizePerBatchResponse(object, response.measure = "angle", : 'patient.id' mapped to mu
##   batch.name patient.id
## 1 PHLC191_P5    PHLC191
## 2 PHLC191_P7    PHLC191
```

Get mutation expression profile

```
ldxe_mut <- getMolecularProfiles(lpdx, data.type="mutation")
print(ldxe_mut)
```

```
## ExpressionSet (storageMode: lockedEnvironment)
## assayData: 16116 features, 12 samples
##   element names: exprs
## protocolData: none
## phenoData
##   sampleNames: PHLC1106 PHLC111 ... PHLC82 (12 total)
##   varLabels: PHLC.ID X.ID
##   varMetadata: labelDescription
## featureData
##   featureNames: NOC2L ISG15 ... RNF128 (16116 total)
##   fvarLabels: probe.Id
##   fvarMetadata: labelDescription
## experimentData: use 'experimentData(object)'
## Annotation: MUT
```

The sample names in expression set are called biobase.id in model slot. Sample names from the expression set canb be be mapped to individual PDX model.ids as

```r
# get sample names
library(Biobase)
```

```
## Loading required package: BiocGenerics

## Loading required package: parallel

##
## Attaching package: 'BiocGenerics'

## The following objects are masked from 'package:parallel':
##
##     clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,
##     clusterExport, clusterMap, parApply, parCapply, parLapply,
##     parLapplyLB, parRapply, parSapply, parSapplyLB

## The following objects are masked from 'package:stats':
##
##     IQR, mad, xtabs

## The following objects are masked from 'package:base':
##
##     Filter, Find, Map, Position, Reduce, anyDuplicated, append,
##     as.data.frame, cbind, colnames, do.call, duplicated, eval,
##     evalq, get, grep, grepl, intersect, is.unsorted, lapply,
##     lengths, mapply, match, mget, order, paste, pmax, pmax.int,
##     pmin, pmin.int, rank, rbind, rownames, sapply, setdiff, sort,
##     table, tapply, union, unique, unsplit, which, which.max,
##     which.min

## Welcome to Bioconductor
##
##     Vignettes contain introductory material; view with
##     'browseVignettes()'. To cite Bioconductor, see
##     'citation("Biobase")', and for packages 'citation("pkgname")'.
```

```r
sn <- Biobase::sampleNames(ldxe_mut)
smap <- mapModelSlotIds(lpdx, id=sn, id.name = "biobase.id", map.to = "model.id")
head(smap)
```

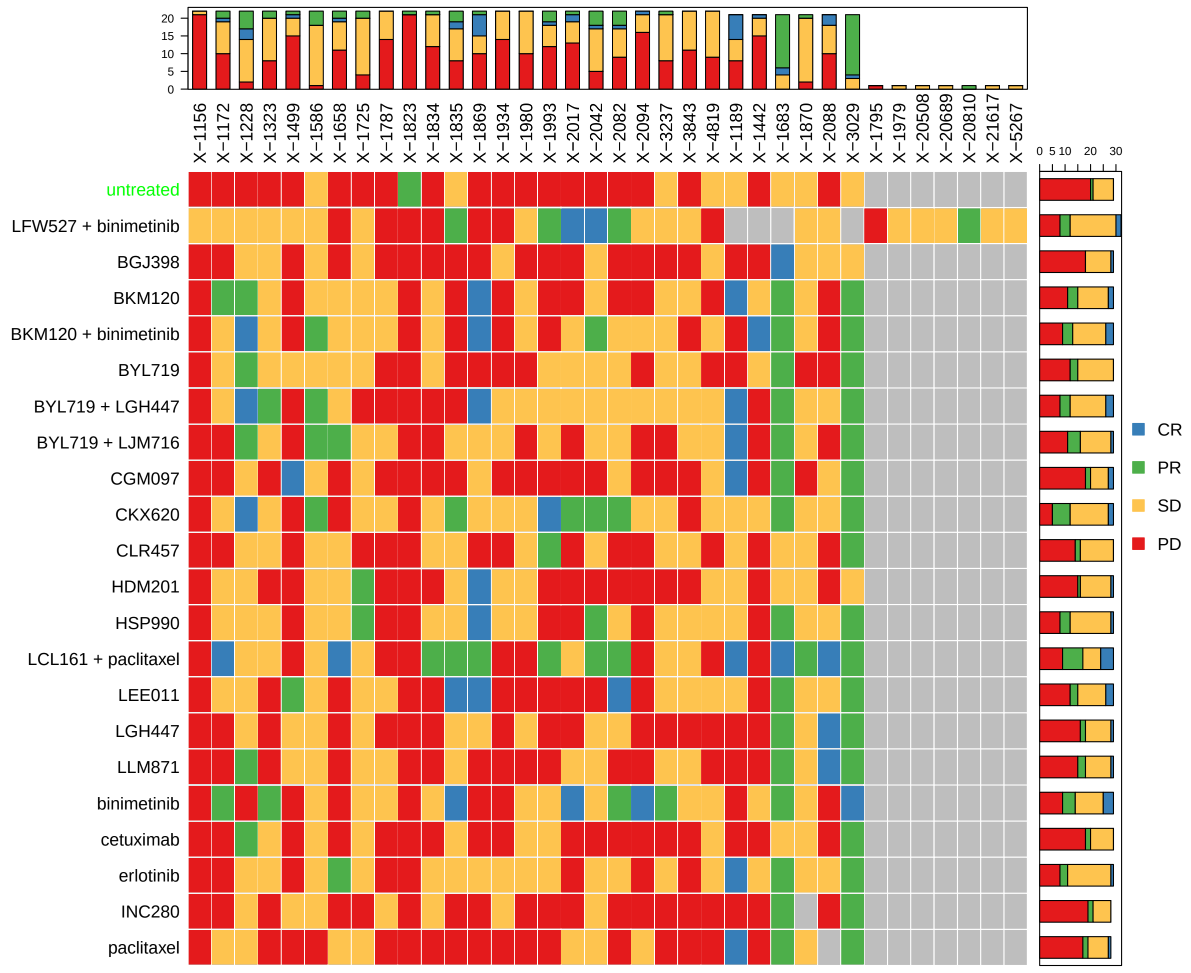
```
##                     biobase.id           model.id
## PHLC1106_P5.501.A1    PHLC1106 PHLC1106_P5.501.A1
## PHLC1106_P5.504.A4    PHLC1106 PHLC1106_P5.504.A4
## PHLC1106_P5.506.B1    PHLC1106 PHLC1106_P5.506.B1
## PHLC1106_P5.507.B2    PHLC1106 PHLC1106_P5.507.B2
## PHLC1106_P5.508.B3    PHLC1106 PHLC1106_P5.508.B3
## PHLC1106_P5.511.C1    PHLC1106 PHLC1106_P5.511.C1
```

What should we do here

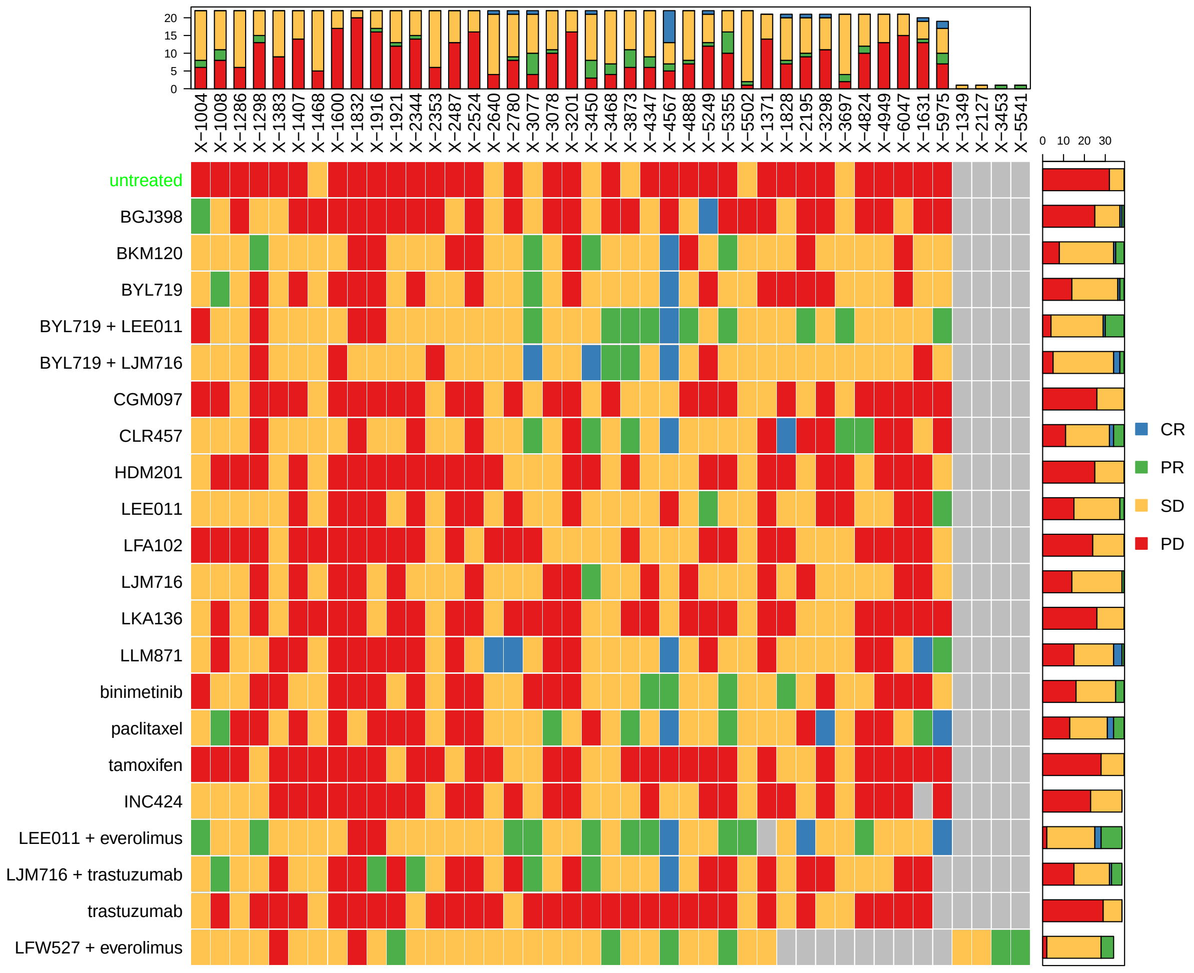
```r
df = getExperiment(lpdx, "PHLC119_P5.506.B1")
#print(df[df$time>85 & df$time<109, c("time", "width", "length", "volume", "comment", "dose")])
```

Create mRECIST plot for PDXE Lung Cancer data

```r
data(pdxe)
df <- getmRECIST(pdxe)
## add tumor.type information
dfMap <- mapModelSlotIds(object=pdxe, id=df$model.id, id.name="model.id",
                         map.to="tumor.type", unique = FALSE)
if(all(df$model.id==dfMap$model.id)) {df$tumor.type = dfMap$tumor.type}
lungDf = df[df$tumor.type=="NSCLC", ]
#pdf(file="DATA-raw/mRECIST_plot_NSCLC.pdf", width=12, height=10)
plotmRECIST(lungDf, groupBy = "biobase.id", control.name = "untreated")
```



```r
#pdf(file="DATA-raw/mRECIST_plot_BRCA.pdf", width=12, height=10)
brDF = df[df$tumor.type=="BRCA", ]
plotmRECIST(brDF, groupBy = "biobase.id", control.name = "untreated")
```

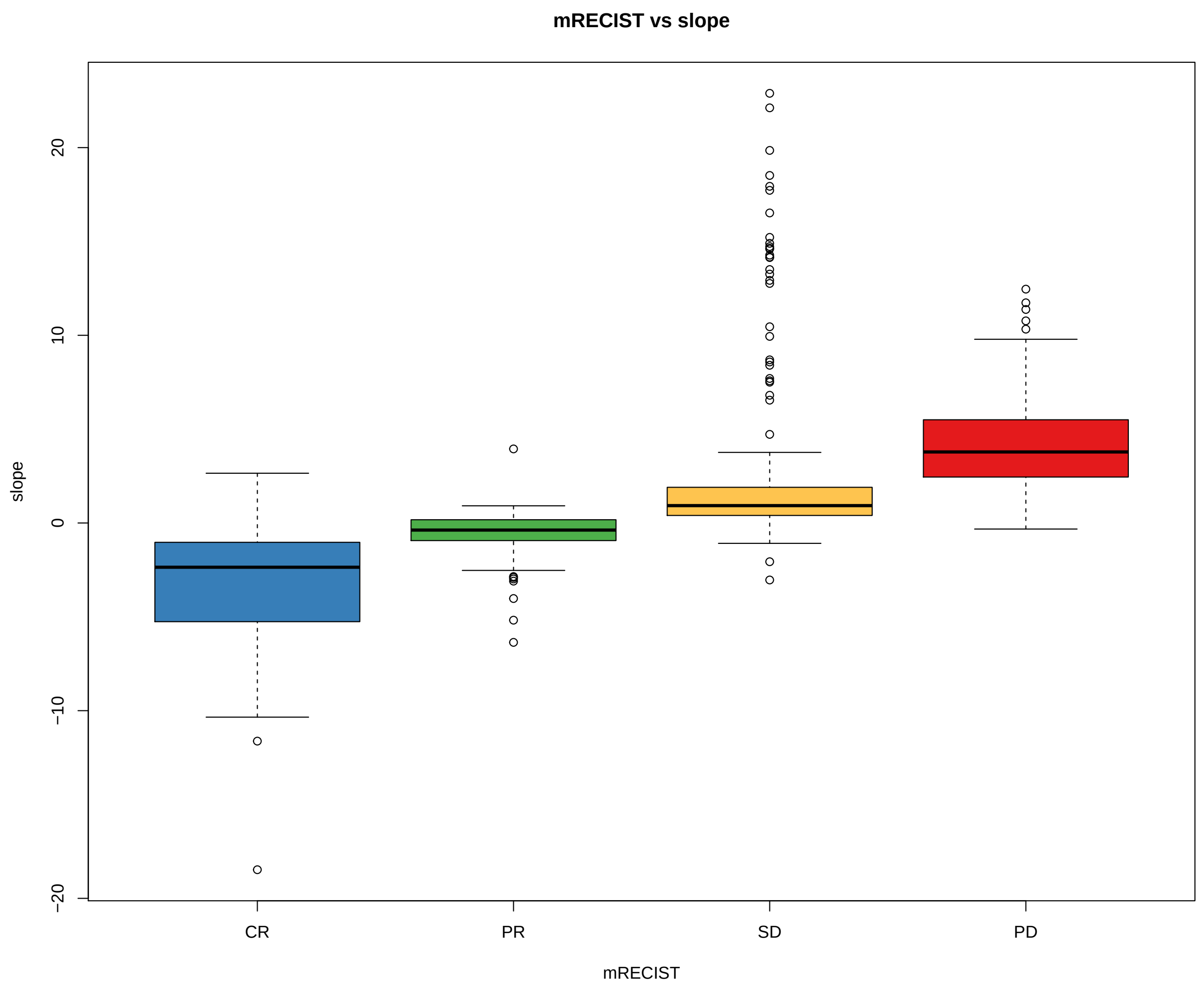


```r
#dev.off()
```

Creat mR vs slop bar-plot

```r
data(pdxe)

pm = modelInfo(pdxe)
lungPID = unique(pm[pm$tumor.type=="NSCLC", "patient.id"])

pdxe_slop <- summarizeResponse(pdxe, response.measure = "slop",
                               group.by="patient.id", summary.stat = "mean")

lung_pdxe_slope <- pdxe_slop[, lungPID]

##--------
pdxe_mR <- summarizeResponse(pdxe, response.measure = "mRECIST_recomputed",
                             group.by="patient.id")

lung_pdxe_mR = pdxe_mR[, lungPID]
```

```
slope=c(); mR=c()
for(dn in rownames(lung_pdxe_slope))
{
  for(pi in colnames(lung_pdxe_slope))
  {
    v = c(lung_pdxe_slope[dn,pi], lung_pdxe_mR[dn,pi])
    if(!is.na(v[1]) & !is.na(v[1]))
    { slope = c(slope,v[1]); mR=c(mR,v[2]) }
  }
}

df = data.frame(mR= mR, slope= as.numeric(slope), stringsAsFactors = FALSE)
df$mR= factor(df$mR, c("CR", "PR", "SD", "PD"))

colPalette = c("#377eb8", "#4daf4a", "#fec44f", "#e41a1c")
#pdf(file="DATA-raw/boxplot_lungCancer.pdf", width=12, height=10)
boxplot(slope~mR, data=df, col=colPalette,
  main="mRECIST vs slope", xlab="mRECIST", ylab="slope")
```



```
#dev.off()
```