The Xeva User's Guide

Arvind Mer [1,2] ***and Benjamin Haibe-Kains*** [1,2,3]

[1]Princess Margaret Cancer Centre, University Health Network, Toronto, Canada
[2]Department of Medical Biophysics, University of Toronto, Toronto, Canada
[3]Department of Computer Science, University of Toronto, Toronto, Canada

November 1, 2018

Contents

1 Introduction

The Xeva package provides efficient and powerful functions for patient-drived xenograft (PDX) based pharmacogenomic data analysis.

2 Installation and Settings

Xeva requires that several packages be installed. All dependencies are available from CRAN or Bioconductor:

```
source('http://bioconductor.org/biocLite.R')
biocLite('Xeva')
```

Load Xeva into your current workspace:

```
library(Xeva)
```

Load the dataset you wish to analyze. For the sake of this tutorial, here we load the NIBR PDXE breast cancer dataset as an example:

```
data(brca)
print(brca)
```

```
## Xeva-set name: PDXE.BRCA
## Creation date: Fri Sep 14 11:41:33 2018
## Number of models: 849
## Number of drugs: 22
## Moleculer dataset: RNASeq, mutation, cnv
```

3 Definitions

Before we further dive into the analysis and visualization, it is important to underastand the terminology used in the *Xeva* package. In a **Xeva** object, the **experiment** slot stores the data for each individual PDX/mouse. With the exception of tumor growth data (time vs. tumor volume), for each individual PDX/mouse, you can access metadata such as the patient's age, sex, tissue histology, and passage information. All of this metadata is stored in the **pdxModel** class, where a unique ID called `model.id` is given to each PDX/mouse model. As for the tumor growth information, Xeva provides separate functions for retrieving visualizing time vs. tumor volume data. We will see later how to get these data for an individual *model.id*, but first, let's define some other terms that appear in the *Xeva* package.

A PDX experiment can be one of the two categories:

- **treatment** represents experiments in which the PDX receives some kind of drug (or drug combination)
- **control** represents experiments in which the PDX receives no drug

To see the effect of a drug, several replicate experiments are done for both the control and the treatment categories. In **Xeva**, a collection of PDX *model.ids* originating from the same patient is organized in **batches** (*batch*). A *batch* has two arms: *control* and *treatment*. This is illustrated in Figure 1.

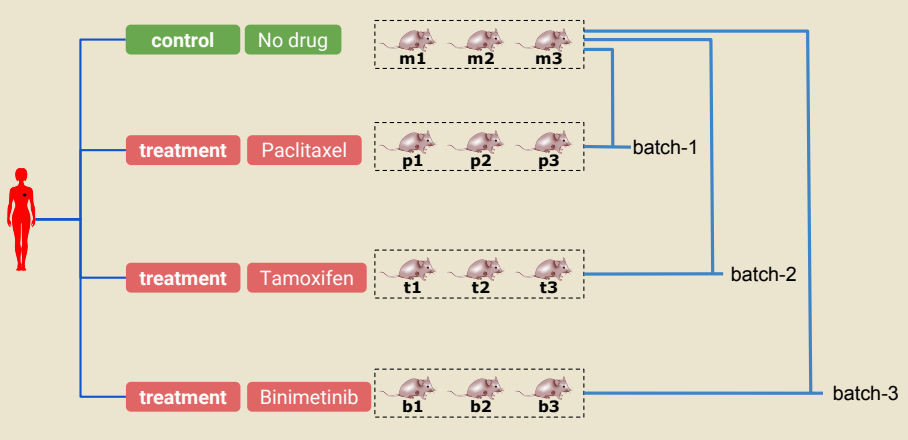

Figure 1: A PDX experiment
The text under each of the PDX/mouse (ie. m1, m2, p1, etc.) denotes the *model.id* in **Xeva**. In this example, three PDXs are delclared as control (m1, m2, and m3). Similarly, in the treatment arm, 3 PDXs are given the drug paclitaxel (p1, p2, and p3), 3 are given tamoxifen (t1, t2, and t3), and 3 are given binimetinib (b1, b2, b3). The PDXs in the control arm and one of the treatment arms together constitute a *batch*. For example, control arm models (m1, m2, and m3) and treatment arm models (t1,t2, and t3) together create a batch called batch-2.

A **Xeva** object binds together all individual experiments, batch information, and molecular data into one single class called XevaSet.

4 Data Access

As mentioned earlier, **Xeva** stores metadata for each individual PDX model. We can retrieve the meta-information about each PDX, such as number of models and tissue type, using:

```
brca.mod <- modelInfo(brca)
dim(brca.mod)
```

```
## [1] 849   5
```

```
brca.mod[1:4, ]
```

```
##                model.id tissue   tissue.name patient.id        drug
## X.1004.BG98 X.1004.BG98   BRCA Breast Cancer     X-1004      BGJ398
## X.1004.biib X.1004.biib   BRCA Breast Cancer     X-1004 binimetinib
## X.1004.BK20 X.1004.BK20   BRCA Breast Cancer     X-1004      BKM120
## X.1004.BY19 X.1004.BY19   BRCA Breast Cancer     X-1004      BYL719
```

The output shows that the *brca* dataset contains 849 PDX models. We can also see the time vs. tumor volume data for a model using:

```
model.data <- getExperiment(brca, model.id = "X.1004.BG98")
head(model.data)
```

```
##      model.id drug.join.name time volume body.weight volume.normal
## 1 X.1004.BG98         BGJ398    0  199.7        28.2     0.0000000
## 2 X.1004.BG98         BGJ398    2  181.9        28.0    -0.0891337
## 3 X.1004.BG98         BGJ398    5  172.7        28.4    -0.1352028
## 4 X.1004.BG98         BGJ398    9  129.6        27.2    -0.3510265
## 5 X.1004.BG98         BGJ398   12   91.3        26.7    -0.5428142
## 6 X.1004.BG98         BGJ398   16  117.1        26.2    -0.4136204
```

Similarly, for **batch** names, we can obtain all predefined batch names using:

```
batch.name <- batchInfo(brca)
batch.name[1:4]
```

```
## [1] "X-1004.BGJ398"      "X-1004.binimetinib" "X-1004.BKM120"
## [4] "X-1004.BYL719"
```

The information about a **batch** can be shown using:

```
batchInfo(brca, batch = "X-1004.binimetinib")
```

```
## $`X-1004.binimetinib`
## name = X-1004.binimetinib
## control = X.1004.uned
## treatment = X.1004.biib
```

Here, for the batch named *X-1004.binimetinib*, we can see that the control sample is *X.1004.uned* and the treatment sample is *X.1004.biib*.

5 Visualizing PDX Growth Curve

Xeva provides a function to plot time vs. tumor volume data for individual models as well as for individual batches. These data can be plotted by using the name of the batch:

```
plotPDX(brca, batch = "X-4567.BKM120")
```

You can choose to see different aspects of this visualization. For example, we can plot normalized volume; we can also change the colors of the lines:

```
plotPDX(brca, batch = "X-4567.BKM120", vol.normal = T, control.col = "#a6611a",
        treatment.col = "#018571", major.line.size = 1, max.time = 40)
```

Data can also be visualized at the patient level by specifying `patient.id`:

```
plotPDX(brca, patient.id="X-3078", drug="paclitaxel",control.name = "untreated")
```

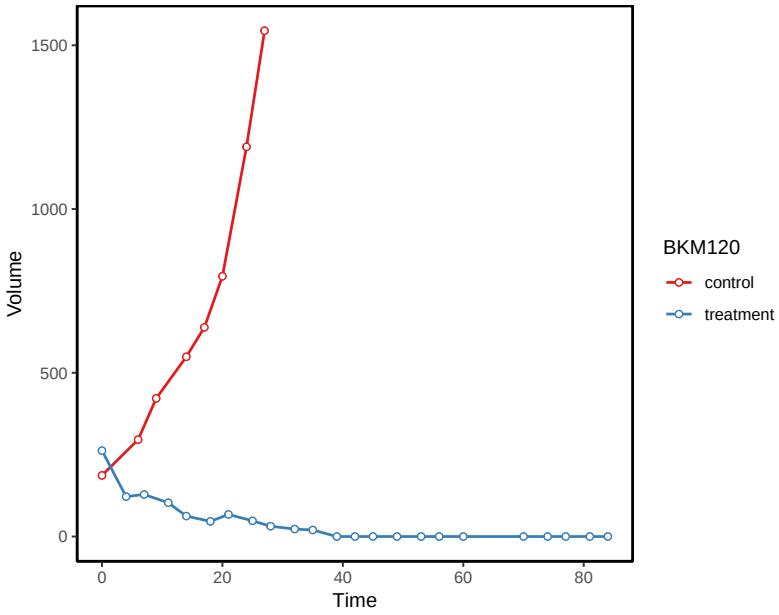

Figure 2: Tumor growth curve for a batch of control and treated PDXs

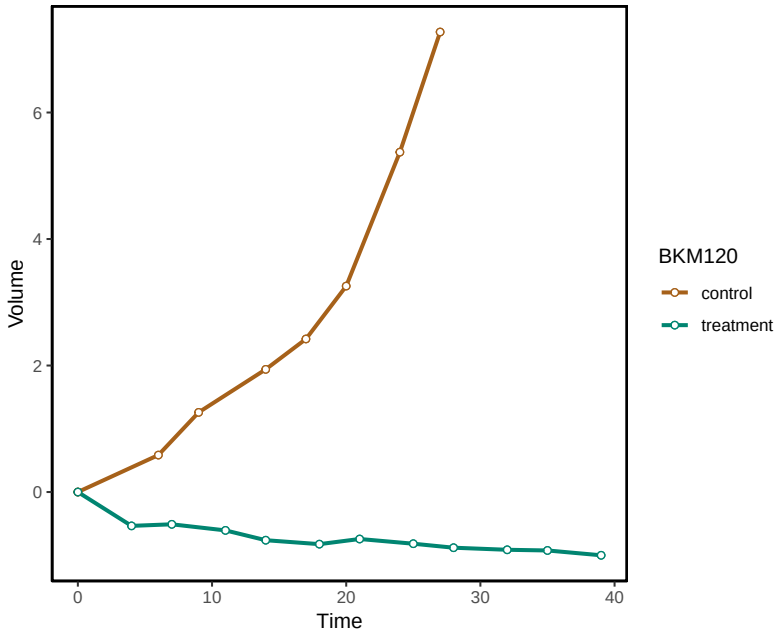

Figure 3: Tumor growth curve for a batch of control and treated PDX
Here, the volume is normalized and plots are truncated at 40 days

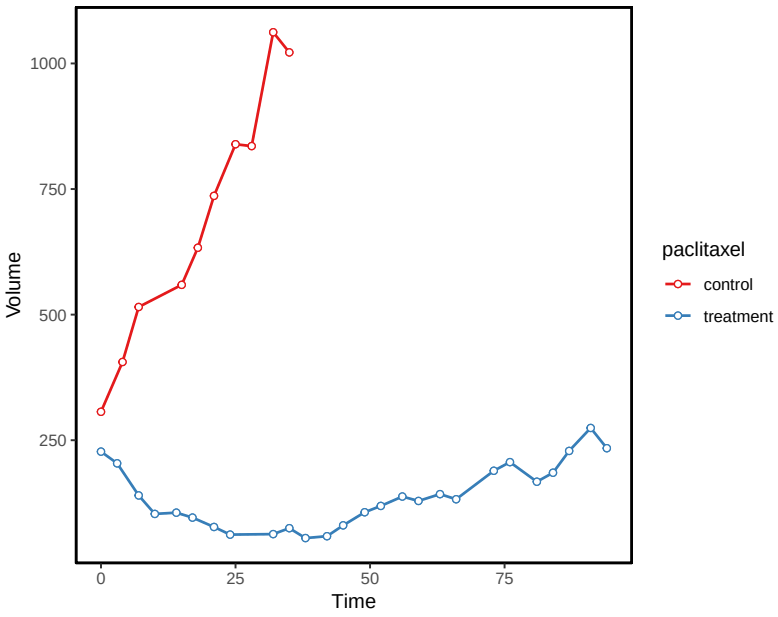

Figure 4: Tumor growth curve for a batch of control and treated PDX generated using patient ID and drug name

6 PDX Model Response

Xeva can effectively summarize PDX drug response data. Here we summarize the **mRECIST** values for the models in our dataset:

```
brca.mr <- summarizeResponse(brca, response.measure = "mRECIST")
brca.mr[1:5, 1:4]
```

```
##                X-1004 X-1008 X-1286 X-1298
## BGJ398             PR     SD     PD     SD
## binimetinib        PD     SD     SD     PD
## BKM120             SD     SD     SD     PR
## BYL719             SD     PR     SD     PD
## BYL719 + LEE011    PD     SD     SD     PD
```

These **mRECIST** values can be visualized using:

```
plotmRECIST(brca.mr, control.name="untreated", row_fontsize=13, col_fontsize=12)
```

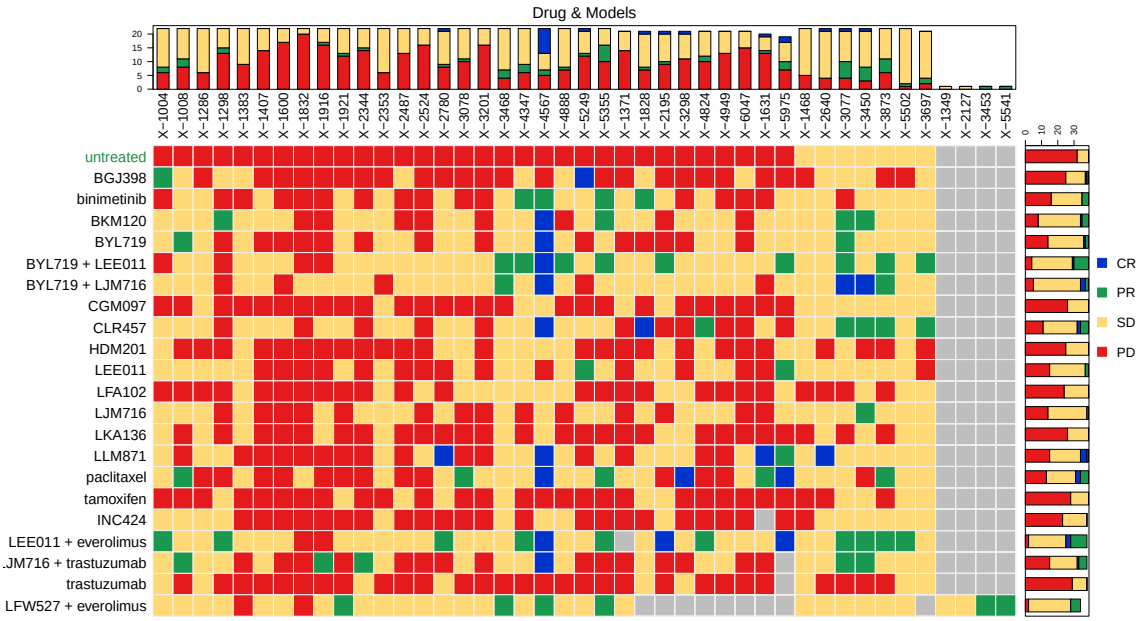

Figure 5: mRECIST plot for PDXE breast cancer data

Waterfall plots are also commonly used to visualize PDX drug response data. Xeva provides a function to visualize and color waterfall plots:

```
waterfall(brca, drug="binimetinib", res.measure="best.average.response")
```

It is useful to color the bars of your waterfall plot by genomic properties. Here we create a waterfall plot for drug BYL719 and color it based on the mutation status of the CDK13 gene. First, we extract the genomic data for the models. Then, we can plot the waterfall plots:

```
mut <- summarizeMolecularProfiles(brca,drug = "BYL719", mDataType="mutation")
```

```
## Loading required package:  Biobase
```

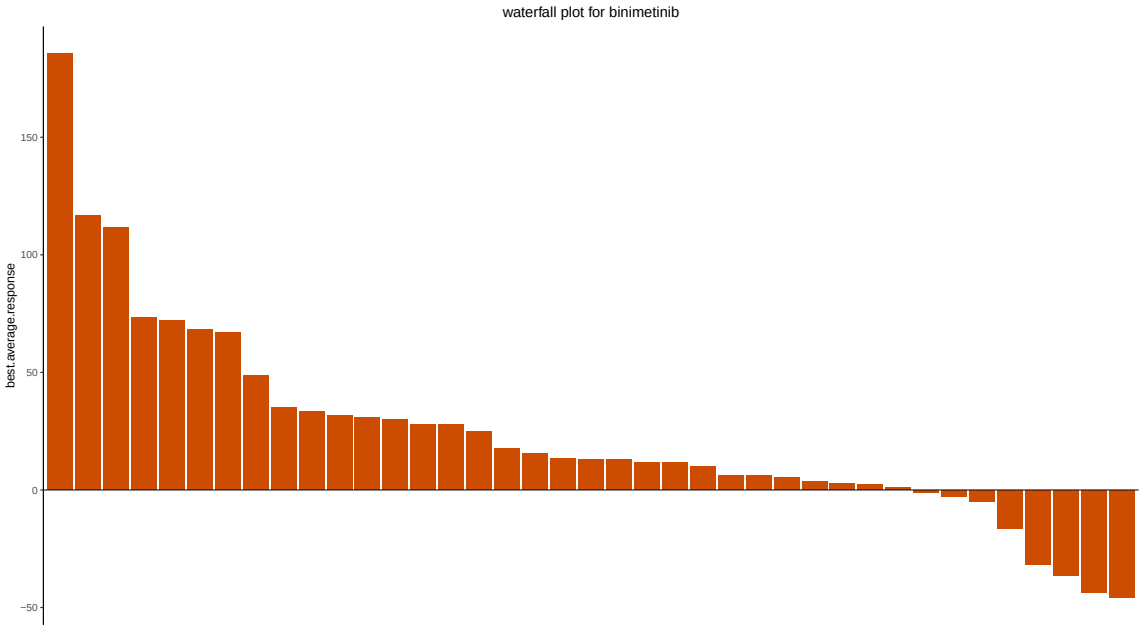

Figure 6: Waterfall plot for binimetinib drug response in PDXs

```
## 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, sd, var, xtabs

## The following objects are masked from 'package:base':
##
##    anyDuplicated, append, as.data.frame, cbind, colMeans, colnames,
##    colSums, do.call, duplicated, eval, evalq, Filter, Find, get,
##    grep, grepl, intersect, is.unsorted, lapply, lengths, Map,
##    mapply, match, mget, order, paste, pmax, pmax.int, pmin,
##    pmin.int, Position, rank, rbind, Reduce, rowMeans, rownames,
##    rowSums, 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")'.
```

```
model.type <- Biobase::exprs(mut)["CDK13", ]
model.type[grepl("Mut", model.type)] <- "mutation"
model.type[model.type!="mutation"] <- "wild type"
model.color <- list("mutation"="#fb8072", "wild type"="#80b1d3")
waterfall(brca, drug="BYL719", res.measure="best.average.response",
          model.id=names(model.type), model.type= model.type,
          type.color = model.color)
```

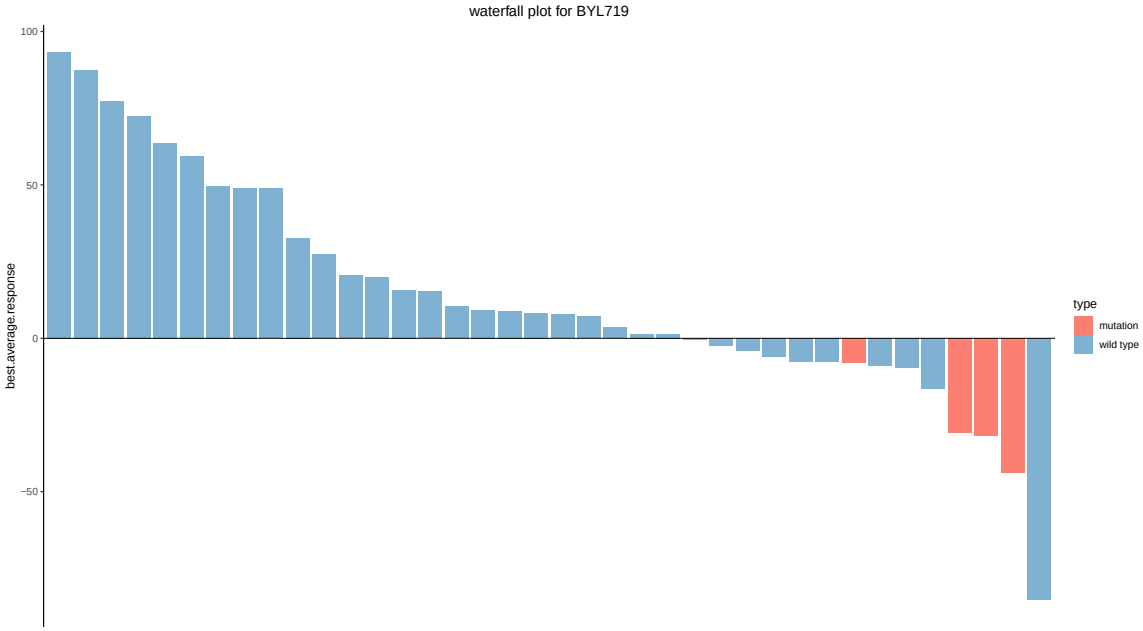

Figure 7: Waterfall plot for binimetinib drug response in PDXs