# Using OncoSimulR: a package for simulating cancer progression data, including drivers and passengers, and allowing for order restrictions.

Ramon Diaz-Uriarte
Dept. Biochemistry, Universidad Autónoma de Madrid
Instituto de Investigaciones Biomédicas "Alberto Sols" (UAM-CSIC)
Madrid, Spain*
http://ligarto.org/rdiaz

2014-07-14 (Rev: 8c8c022)

## Contents



## 1 Introduction

This vignette presents the OncoSimulR package. OncoSimulR allows you to simulate tumor progression using several models of tumor progression. In these simulations you can restrict the order in which mutations can accumulate. For instance, you can restrict the allowed order as specified, for instance, in Oncogenetic Tree (OT; [1, 2]) or Conjunctive Bayesian Network (CBN; [3, 4, 5]) models. Moreover, you can add passenger mutations to the simulations. The models so far implemented are all continuous time models, which are simulated using the BNB algorithm of Mather et al. [6]. This is a summary of some of the key features:

- You can pass arbitrary restrictions as specified by OTs or CBNs.
- You can add passenger mutations.
- You can allow for deviations from the OT and CBN models, specifying a penalty for such deviations (the $s_h$ parameter).
- Right now, three different models are available, two that lead to exponential growth, one of them loosely based on Bozic et al. [7], and another that leads to logistic-like growth, based on McFarland et al. [8].
- Simulations are generally very fast as I use the BNB algorithm implemented in C++.

*ramon.diaz@iib.uam.es, rdiaz02@gmail.com

Further details about the motivation for wanting to simulate data this way can be found in [9], where additional comments about model parameters and caveats are discussed. The Java program by [10] offers somewhat similar functionality, but they are restricted to at most four drivers, you cannot use arbitrary CBNs or OTs to specify order restrictions, there is no allowance for passengers, and a single type of model (a discrete time Galton-Watson process) is implemented.

Using this package will often involve the following steps:

1. Specify the restrictions in the order of mutations: section 2.
2. Simulate cancer progression: section 3. You can simulate for a single subject or for a set of subjects. You will need to
   - Decide on a model (e.g., Bozic or McFarland).
   - Specify the parameters of the model.

   Of course, at least for initial playing around, you can use the defaults.
3. Sample from the simulated data: section 4, and do something with those simulated data (e.g., fit an OT model to them). What you do with the data, however, is outside the scope of this package.

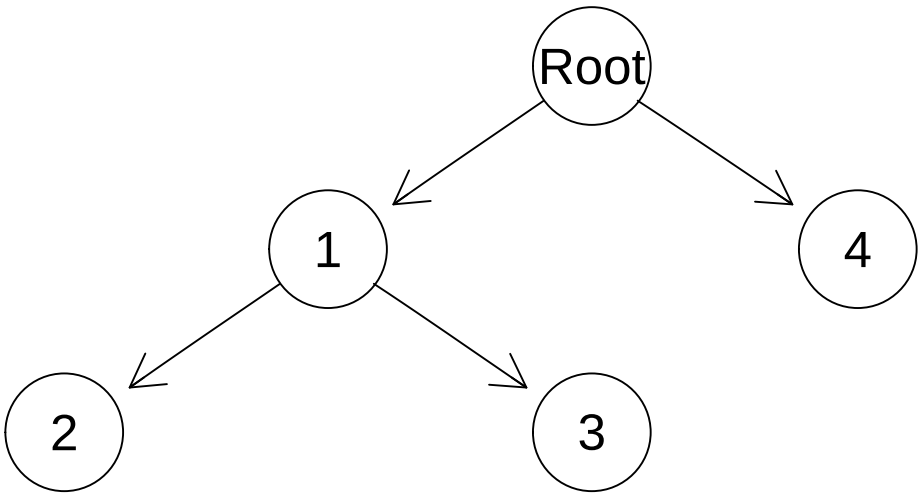
Before anything else, let us load the package. We also explicitly load *graph* for the vignette to work (you do not need that for your usual interactive work).

```
library(OncoSimulR)
library(graph)
```

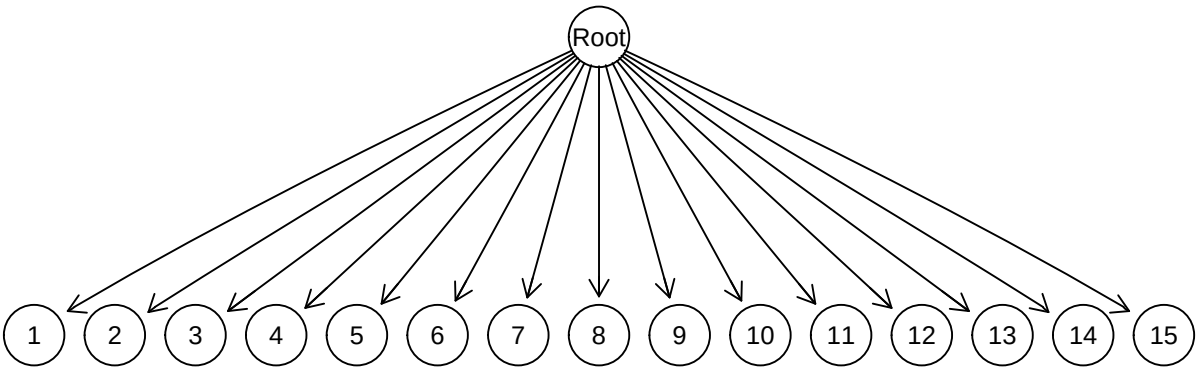
## 2 Specifying restrictions: posets

How to specify the restrictions is shown in the help for `poset`. It is often useful, to make sure you did not make any mistakes, to plot the poset. This is from the examples:

```
## Node 2 and 3 depend on 1, and 4 depends on no one
p1 <- cbind(c(1, 1, 0), c(2, 3, 4))
plotPoset(p1, addroot = TRUE)
```

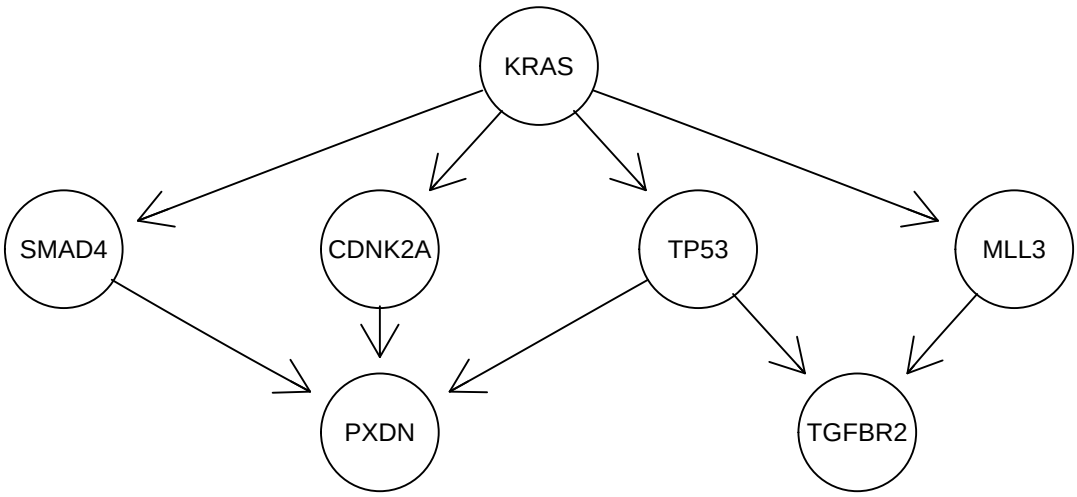



```
## A simple way to create a poset where no gene (in a set of 15) depends
## on any other.
p4 <- cbind(0, 15)
plotPoset(p4, addroot = TRUE)
```




Specifying posets is actually straightforward. For instance, we can specify the pancreatic cancer poset in Gerstung et al. [5] (their figure 2B, left). We specify the poset using numbers, but for nicer plotting we will use names (KRAS is 1, SMAD4 is 2, etc). This example is also in the help for `poset`:

```
pancreaticCancerPoset <- cbind(c(1, 1, 1, 1, 2, 3, 4, 4, 5),
                               c(2, 3, 4, 5, 6, 6, 6, 7, 7))
plotPoset(pancreaticCancerPoset,
          names = c("KRAS", "SMAD4", "CDNK2A", "TP53",
                    "MLL3","PXDN", "TGFBR2"))
```




## 3 Simulating cancer progression

We can simulate the progression in a single subject. Using an example very similar to the one in the help:

```
## use poset p1101
data(examplePosets)
p1101 <- examplePosets[["p1101"]]

## Bozic Model
```

```
b1 <- oncoSimulIndiv(p1101, keepEvery = 15)
summary(b1)
```

```
##   NumClones TotalPopSize LargestClone MaxNumDrivers
## 1       547    104328494     43472411             4
##   MaxDriversLast NumDriversLargestPop TotalPresentDrivers
## 1              4                    1                   6
##   FinalTime NumIter HittedWallTime errorMF OccurringDrivers
## 1       542   28375          FALSE      NA 1, 2, 3, 7, 8, 9
```

The first thing we do is make it simpler (for future examples) to use a set of restrictions. In this case, those encoded in poset p1101. Then, we run the simulations and look at a simple summary and a plot.

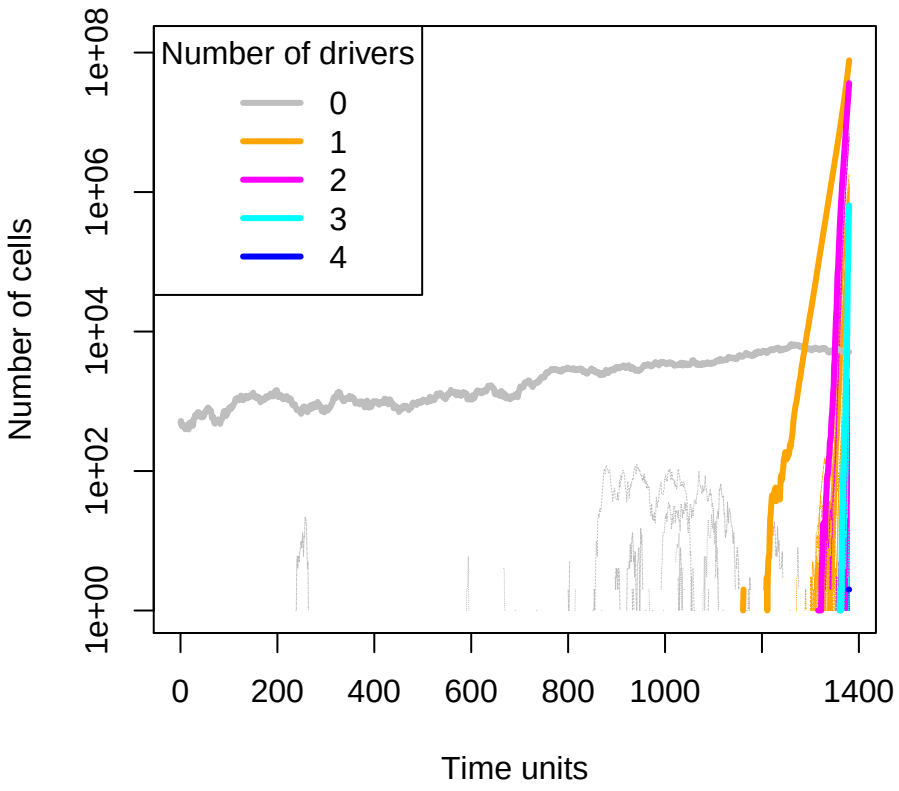
If you want to plot the trajectories, it is better to keep more frequent samples, so you can see when clones appear:

```
b2 <- oncoSimulIndiv(p1101, keepEvery = 1)

summary(b2)
```

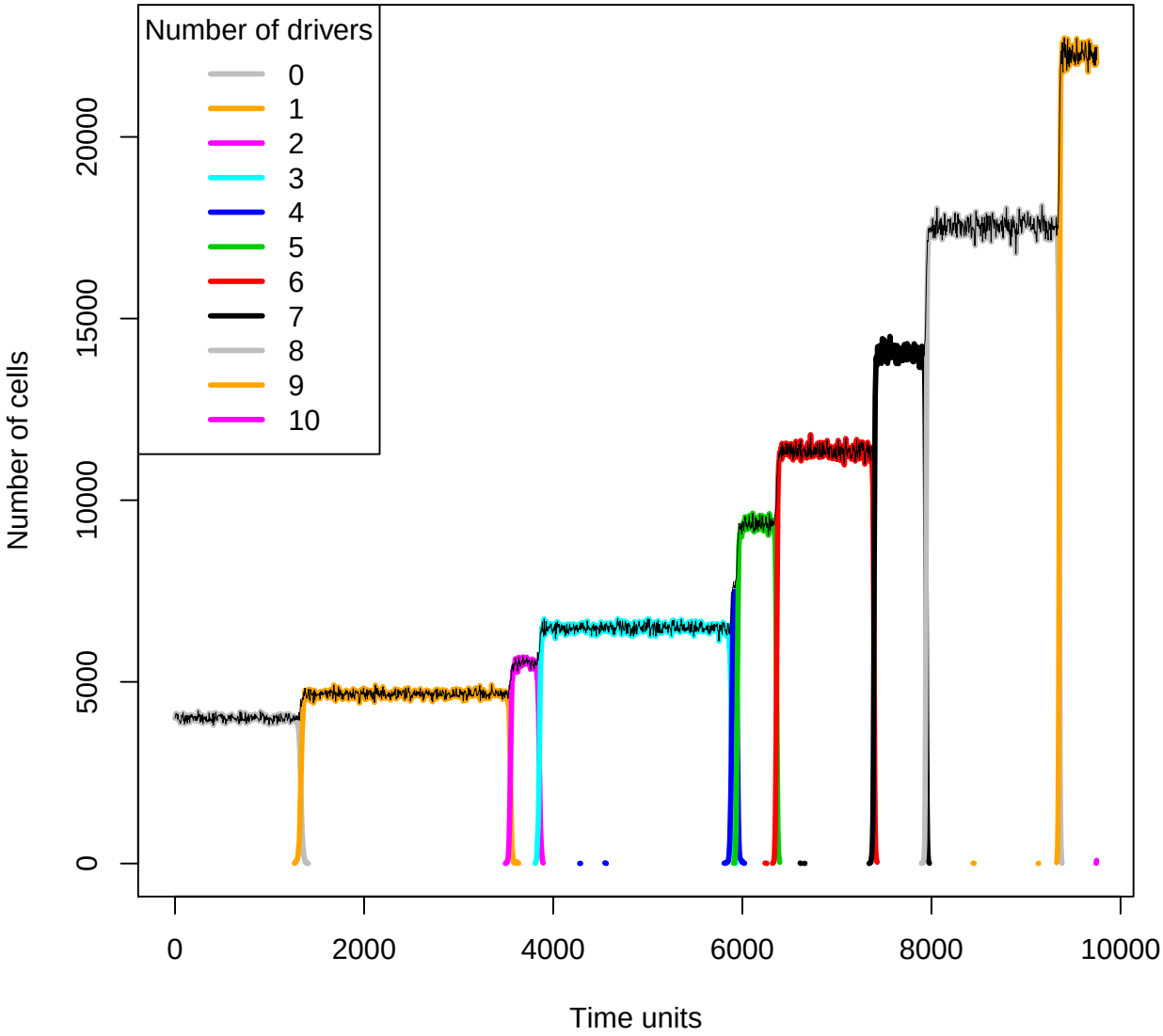
```
##   NumClones TotalPopSize LargestClone MaxNumDrivers
## 1       638    114379946     53370786             4
##   MaxDriversLast NumDriversLargestPop TotalPresentDrivers
## 1              4                    1                   9
##   FinalTime NumIter HittedWallTime errorMF
## 1      1380   32924          FALSE      NA
##            OccurringDrivers
## 1 1, 2, 3, 4, 5, 6, 7, 8, 9
```

```
plot(b2)
```




The following is an example where we do not care about passengers, but we want to use a different graph, and we want a few more drivers before considering cancer has been reached. And we allow it to run for longer. Note that in the McF model `detectionSize` really plays no role. Note also how we pass the poset: it is the same as before, but now we directly access the poset in the list of posets.

```
m2 <- oncoSimulIndiv(examplePosets[["p1101"]], model = "McFL",
                     numPassengers = 0, detectionDrivers = 10,
                     mu = 5e-7, initSize = 4000,
                     sampleEvery = 0.025,
                     finalTime = 25000, keepEvery = 5,
                     detectionSize = 1e6)
plot(m2, addtot = TRUE, log = "")
```




The default is to simulate progression until a simulation reaches cancer (i.e., only simulations that satisfy the detectionDrivers or the detectionSize will be returned). If you use the McF model with large enough `initSize` this will often be the case but not if you use very small `initSize`. Likewise, most of the Bozic runs do not reach cancer. Lets try a few:

```
b3 <- oncoSimulIndiv(p1101, onlyCancer = FALSE)
summary(b3)
```

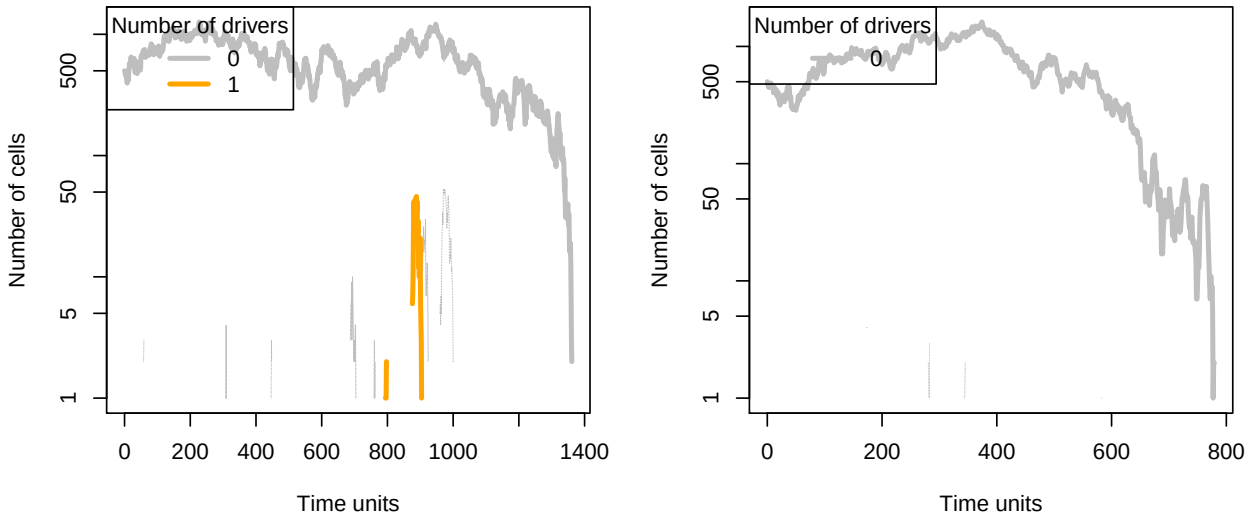
```
##   NumClones TotalPopSize LargestClone MaxNumDrivers
## 1        14            0            0             1
##   MaxDriversLast NumDriversLargestPop TotalPresentDrivers
## 1              0                    0                   2
##   FinalTime NumIter HittedWallTime errorMF OccurringDrivers
## 1      1362    1391          FALSE      NA             1, 7
```

```
b4 <- oncoSimulIndiv(p1101, onlyCancer = FALSE)
summary(b4)
```

```
##   NumClones TotalPopSize LargestClone MaxNumDrivers
## 1        10            0            0             0
##   MaxDriversLast NumDriversLargestPop TotalPresentDrivers
## 1              0                    0                   0
##   FinalTime NumIter HittedWallTime errorMF OccurringDrivers
## 1       780     798          FALSE      NA               NA
```

Plot those runs:

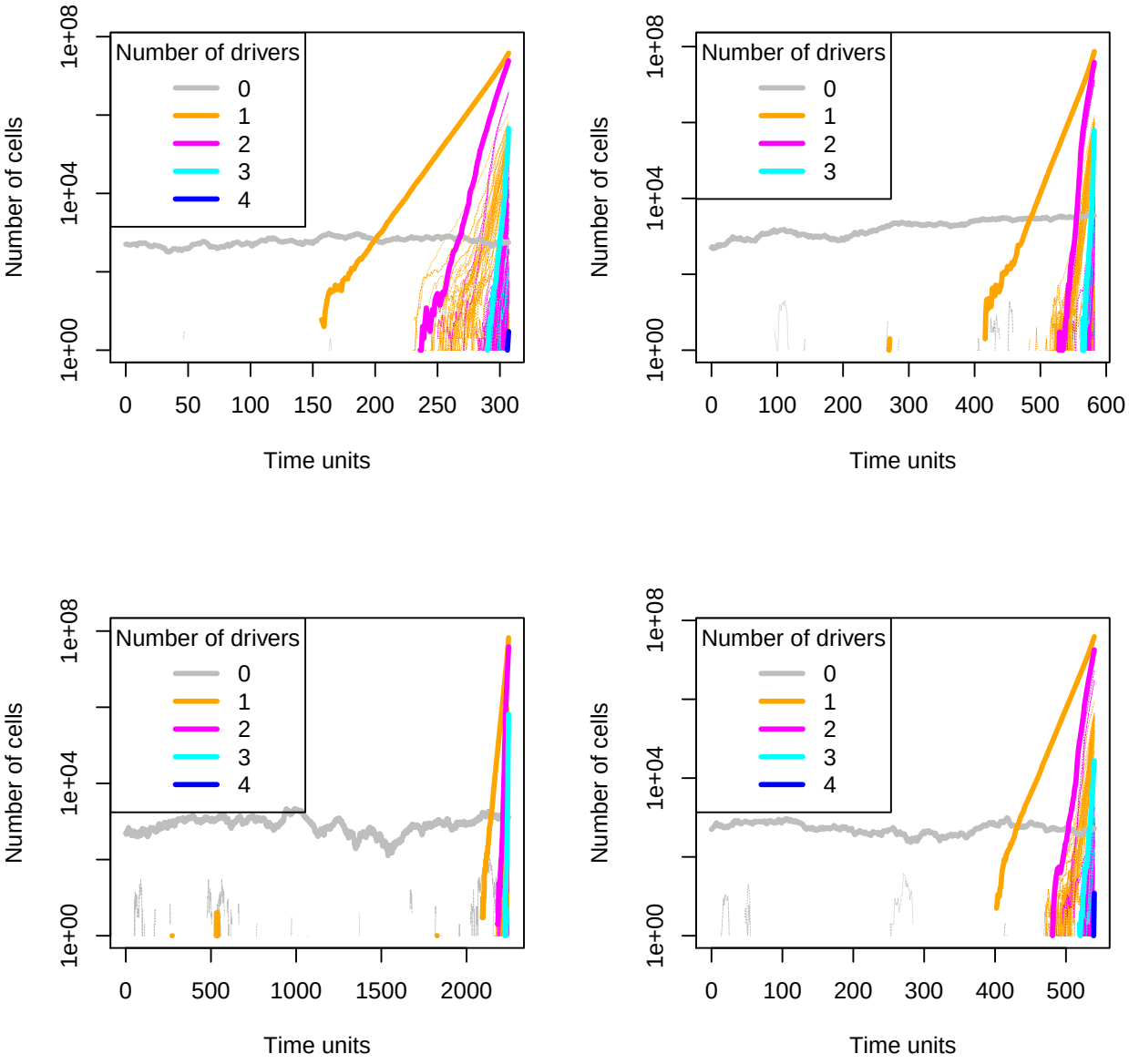
```
par(mfrow = c(1, 2))
par(cex = 0.8) ## smaller font
plot(b3)
plot(b4)
```




## 3.1 Simulating progression in several subjects

To simulate the progression in a bunch of subjects (we will use only four, so as not to fill the vignette with plots) we can do, with the same settings as above:

```
p1 <- oncoSimulPop(4, p1101)
par(mfrow = c(2, 2))
plot(p1)
```




## 4 Sampling from a set of simulated subjects

You will often want to do something with the simulated data. For instance, sample the simulated data. Here we will obtain the trajectories for 100 subjects in a scenario without passengers. Then we will sample with the default options and store that as a vector of genotypes (or a matrix of subjects by genes):

```
m1 <- oncoSimulPop(100, examplePosets[["p1101"]],
                   numPassengers = 0)
```

The function samplePop samples that object, and also gives you some information about the output:

```
genotypes <- samplePop(m1)

##
##  Subjects by Genes matrix of  100  subjects and  11  genes:
```

What can you do with it? That is up to you. As an example, let us try to infer an oncogenetic tree (and plot it) using the *Oncotree* package [11] after getting a quick look at the marginal frequencies of events:

```
colSums(genotypes)/nrow(genotypes)
```

```
##  G.1  G.2  G.3  G.4  G.5  G.6  G.7  G.8  G.9 G.10 G.11
## 0.52 0.03 0.03 0.00 0.00 0.00 0.55 0.03 0.01 0.00 0.00
```

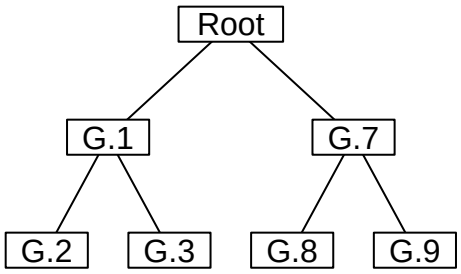
```
require(Oncotree)
```

```
## Loading required package: Oncotree
## Loading required package: boot
```

```
ot1 <- oncotree.fit(genotypes)
```

```
## The following events had no observed occurances,so they will not be included in the construc
## G.4 G.5 G.6 G.10 G.11
```

```
plot(ot1)
```




Your run will likely differ from mine, but with the defaults (detection size of $10^8$) it is likely that events down the tree will never appear. You can set `detectionSize = 1e9` and you will see that events down the tree are now found in the cross-sectional sample.

Alternatively, you can use single cell sampling and that, sometimes, recovers one or a couple more events.

```
genotypesSC <- samplePop(m1, typeSample = "single")
```

```
##
##  Subjects by Genes matrix of  100  subjects and  11  genes:
```

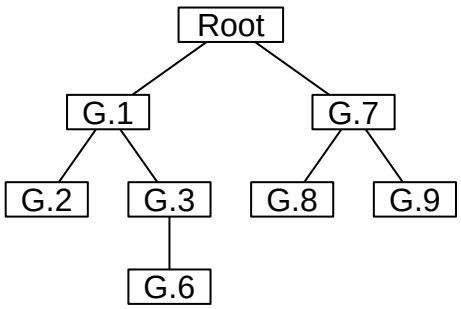
```
colSums(genotypesSC)/nrow(genotypesSC)
```

```
##  G.1  G.2  G.3  G.4  G.5  G.6  G.7  G.8  G.9 G.10 G.11
## 0.66 0.09 0.09 0.00 0.00 0.01 0.61 0.11 0.15 0.00 0.00
```

```
ot2 <- oncotree.fit(genotypesSC)

## The following events had no observed occurances,so they will not be included in the constru
## G.4 G.5 G.10 G.11

plot(ot2)
```




You can of course rename the columns of the output matrix to something else if you want so the names of the nodes will reflect those potentially more meaningful names.

## 5 Session info and packages used

This is the information about the version of R and packages used:

```
sessionInfo()

## R version 3.2.0 RC (2015-04-08 r68161)
## Platform: x86_64-apple-darwin13.4.0 (64-bit)
## Running under: OS X 10.9.5 (Mavericks)
##
## locale:
## [1] C/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets
## [6] methods   base
##
## other attached packages:
## [1] Oncotree_0.3.3   boot_1.3-16      graph_1.46.0
## [4] OncoSimulR_1.2.0 knitr_1.9
```

```
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_0.11.5          chron_2.3-45
##  [3] grid_3.2.0           plyr_1.8.1
##  [5] stats4_3.2.0         formatR_1.1
##  [7] evaluate_0.6         highr_0.4.1
##  [9] reshape2_1.4.1       data.table_1.9.4
## [11] Rgraphviz_2.12.0     BiocStyle_1.6.0
## [13] tools_3.2.0          stringr_0.6.2
## [15] parallel_3.2.0       BiocGenerics_0.14.0
```

## References


[1] R Desper, F Jiang, O P Kallioniemi, H Moch, C H Papadimitriou, and A A Schäffer. Inferring tree models for oncogenesis from comparative genome hybridization data. *J Comput Biol*, 6(1):37–51, 1999. URL: http://view.ncbi.nlm.nih.gov/pubmed/10223663.

[2] A Szabo and Kenneth M Boucher. Oncogenetic trees. In W-Y Tan and L Hanin, editors, *Handbook of cancer models with applications*, chapter 1, pages 1–24. World Scientific, 2008. URL: http://www.worldscibooks.com/lifesci/6677.html.

[3] Niko Beerenwinkel, Nicholas Eriksson, and Bernd Sturmfels. Conjunctive Bayesian networks. *Bernoulli*, 13(4):893–909, November 2007. URL: http://projecteuclid.org/euclid.bj/1194625594, doi:10.3150/07-BEJ6133.

[4] Moritz Gerstung, Michael Baudis, Holger Moch, and Niko Beerenwinkel. Quantifying cancer progression with conjunctive Bayesian networks. *Bioinformatics (Oxford, England)*, 25(21):2809–2815, November 2009. URL: http://dx.doi.org/10.1093/bioinformatics/btp505, doi:10.1093/bioinformatics/btp505.

[5] Moritz Gerstung, Nicholas Eriksson, Jimmy Lin, Bert Vogelstein, and Niko Beerenwinkel. The Temporal Order of Genetic and Pathway Alterations in Tumorigenesis. *PLoS ONE*, 6(11):e27136, November 2011. URL: http://dx.plos.org/10.1371/journal.pone.0027136, doi:10.1371/journal.pone.0027136.

[6] William H Mather, Jeff Hasty, and Lev S Tsimring. Fast stochastic algorithm for simulating evolutionary population dynamics. *Bioinformatics (Oxford, England)*, 28(9):1230–1238, March 2012. URL: http://www.ncbi.nlm.nih.gov/pubmed/22437850, doi:10.1093/bioinformatics/bts130.

[7] Ivana Bozic, Tibor Antal, Hisashi Ohtsuki, Hannah Carter, Dewey Kim, Sining Chen, Rachel Karchin, Kenneth W Kinzler, Bert Vogelstein, and Martin A Nowak. Accumulation of driver and passenger mutations during tumor progression. *Proceedings of the National Academy of Sciences of the United States of America*, 107:18545–18550, September 2010. URL: http://www.ncbi.nlm.nih.gov/pubmed/20876136, doi:10.1073/pnas.1010978107.

[8] Christopher D McFarland, Kirill S Korolev, Gregory V Kryukov, Shamil R Sunyaev, and Leonid a Mirny. Impact of deleterious passenger mutations on cancer progression. *Proceedings of the National Academy of Sciences of the United States of America*, 110(8):2910–5, February 2013. URL: http://www.ncbi.nlm.nih.gov/pubmed/23388632, doi:10.1073/pnas.1213968110.

[9] R Diaz-Uriarte. Inferring restrictions in the temporal order of mutations during tumor progression: effects of passengers, evolutionary models, and sampling. *bioRxiv*, pages 0–36, 2014. URL: http:

//biorxiv.org/content/early/2014/06/22/005587, doi:doi:http://dx.doi.org/10.1101/005587.

[10] JG Reiter, Ivana Bozic, Krishnendu Chatterjee, and MA Nowak. TTP: tool for tumor progression. In Natasha Sharygina and Helmut Veith, editors, *Computer Aided Verification, Lecture Notes in Computer Science*, pages 101–106. Springer-Verlag, Berlin, Heidelberg, 2013. URL: http://link.springer.com/chapter/10.1007/978-3-642-39799-8_6.

[11] Aniko Szabo and Lisa Pappas. Oncotree: Estimating oncogenetic trees, 2013. http://cran.r-project.org/package=Oncotree.