Programming assignment, R: Simulating Pathways, Mutual Exclusivity, et al. in Cancer Progression Models

Raquel Blanco Martínez-Illescas*, Daniel Peñas Utrilla*, Henry Secaira Morocho*

2021-01-18

Contents

1 Outline and methods

CPMs inferred by three different probabilistic models (1) are mapped to actual evolutionary models using OncoSimulR (4). In these works, CPMs are depicted as oncogenic tree (OT) models or Conjunctive Bayesian Networks (CBN) to show the order in which mutations accumulate to cancer progression. Regardless of the type of depiction, a CPM is not an evolutionary model, since ancestral relationships between genotypes are not represented, just restriction patterns among genes obtained from cross-sectional tumor samples of different individuals. However, these constraints between genes can be easily used to generate a fitness landscape and show all possible paths of tumor progression, mapping genotypes to fitness (5).

The flow work consist of defining a DAG of restrictions from CPMs inferred by probabilistic models. Then, generate a fitness landscape from the DAG and simulate tumor progression from the fitness effects specified in the DAG. Moreover, simplified versions of the models are implemented to better visualize fitness landscape as well as different effects in (!!of!!) fitness.

Complete fitness specification of each model is obtained from the function`allFitnessEffects`. The fitness-genotype mapping is specified by a data frame. Each generative models is codded setting this data frame where dependencies between genes inferred in each model are indicated as parent-child relationships. Parent gene mutations are mandatory to child gene mutations to occur (monotonicity). Parents and children gene mutations are introduced in the model by two different columns, "parent" and "child", respectively. Mutations don't requiring a previous mutation derive from a "Root" node (Wild type genotype). Moreover, two additional columns are set to specify fitness effects associated to each genotype (evolutionary model, not just a generative model). Those columns are "s" and "sh". In the column "s" is specified the fitness

*Universidad Autónoma de Madrid, Bionformatics and Computational Biology Master

effect when the restrictions defined in the CPMs model are satisfied. On the other hand, in column "sh" is set the fitness effect when restrictions are not satisfied. Mutations that are against inferred constraints can be set to a 0 value in fitness effect. However, we want to allow deviations from the monotonicity, setting a penalization when this situations occurs (negative value in "sh").

Additionally, type of dependency between mutations can also be specified in the data frame. There are three different possible dependencies in OncoSimulR: *monotone relationship*, where relationship between specific genes is fully respected; *semiminotone relationship*, where two or more genes are connected to the same gene, but if just one of the parent mutations occurs is enough to child gene mutation; and *XOR relationship*, similar to the previous one, but child gene mutation will occur only if one parent gene is already mutated. Dependency is set in a new column called "typeDep" in the data frame. Nomenclature for the three possible genes relationship is "MN", "SM" and "XMPN" for monotone, semimonotone and XOR relationship, respectively. (INCLUDE COLOR OF RELATIONSHIP???)

Restrictions' data frame just defined is used as argument of the function `allFitnessEffects`. In addition, restrictions considered in the data frame can apply not to one gene, but to a set of genes (module). This idea is considered by authors of the three different probabilistic models to map (1). This situation is established by the argument "geneToModule". Furthermore, OncoSimulR allows to specify driver or passenger genes in the models. In the model mapped, all genes/modules implemented are driver genes/modules, but they are explicitly indicated with this argument. A fitnessEffects object is returned from the function `allFitnessEffects`. This object can be used as input for the function `plot` and the DAG is visualized. It is possible because OncoSimulR package implement the method `plot.fitnessEffects` for the fitnessEffects objects. When modules are used, they can be expanded with the argument `expandModules = TRUE`. In addition, Wild type fitness is shown setting `addwt = TRUE`. To obtain a table with all the fitness effects of all genotypes, the previous fitnessEffects object is used as input in the function `evalAllGenotypes`. Finally, fitness landscape of each evolutionary model is visualize with both `plot` or `PlotFitnessLanscape` function.

Furthermore, fitness effects associated to each genotype can be used to simulate tumor progression with OncoSimulR. For that aim, the function `oncoSimulIndiv` is used. This function simulates a single evolutionary trajectory. McFarland model (continuous-time, logistic-like, and death rate depends on population size) is used for simulation of cancer progression, since it leads to a better performance (5). Initial population size, the mutation rate and final time of simulation is also set. The argument `sampleEvery` informs about how often the whole population is sampled. As we are using McFarland model, a very small value is set. The argument `keepEvery` is set larger than the argument`sampleEvery` to obtain nice plots. Furthermore, `keepPhylog = TRUE` is set to plot the parent-child relationships occurring in the simulation as well as its frequency, using `plotClonePhylog` function. The argument `onlyCancer` is set to TRUE when simulation is returned when cancer is reached. Otherwise, it is set to FALSE. Simulations are plotted using the function `plot`. Plot styles is set to "stacked" or "line" with the argument `type`.

From the canonical models, simplified derived models are constructed. Those models maintain dependencies between genes, but they just focus on the important genes in cancer progression (6). These simplified versions are used to properly show and discuss different utilities available in OncoSimulR, such as order effects, epistasis, synthetic viability or synthetic lethality. All this utilities are introduced inside the function `allFitnessEffects` as an specific argument.

Order effects between genes is introduced in the argument `orderEffects` and is defined with ">" symbols; for instance, $A > B$ means that order effect is satisfied only when gene A is mutated before gen B. This relationship is established between two or three genes. On the other hand, epistasis and the phenomena of synthetic viability or lethality are introduce with the argument `epistasis`. In this case, dependent genes are separated by ":", and the absent of a gene is indicated by a "-" before the gene.

Frequency-dependent fitness allows to make fitness depend on the frequency of other genotypes. This dependency is defined in the data frame setting a new column called `Fitness`. It includes the birth rate of each genotype and is defined as a formula containing $f\_$ to denote the relative frequency. In addition, genotypes are also specified setting another column called `Genotype`.

2 pathTiMEx, a generative probabilistic graphical model of cancer progression

In (1), the authors introduce a generative probabilistic graphical model of cancer progression called *pathTiMEx*. It is both, a waiting time model for independent mutually exclusive pathways, and a waiting time model for cancer progression among single genes. This generative model allows to generate a cancer progression model including mutual exclusivity between groups and progression among pathways. In this approach, authors think in both, genes and modules effects (set of genes). The colorectal cancer model depicted in Figure 3.A (1) is used as an example of model to map. The colorectal cancer dataset used to built that model is obtained from (7). The poset restrictions proposed can be coded using the **OncoSimulR package** (4), concretely, the function `allFitnessEffects`. It creates mutations effects given specification of restrictions, epistasis or order effects. In this case, restrictions are used to construct the graph.

Some parameters are mandatory when the function `allFitnessEffects` is used. It is the case of the restriction table. Authors don't specify `s` nor `sh` value since they are not interested in fitness. To justify the values given, we will use the waiting time rate parameter λ defined in the model. Early events in cancer progression will have greater λ values while late events will have a lower one (values for all genes or modules are showed in Table 1). Thus, genes/modules with higher λ will receive a higher fitness value (`s`). On the other hand, `sh` is given a constant value for all possible situations.

Table 1: Waiting time rate parameter (λ) for each gene/module

Gene/module	Waiting time rate parameter (λ)
APC	9.5
KRAS	2.89
TP53, EVC2	1.92
PIK3CA, EPHA3	0.17
FBXW7, TCF7L2	0.08

Dependency between genes is set as monotonic (`MN`). Model will be represented as an **Diaciclic Direct Graph (DAG)** where arrows connecting genes or modules indicate direct dependencies or constraints between them (8).

```
## First, it is necessary to load OncoSimulR and igraph package
library(OncoSimulR)


## Restriction table (extended version of the poset)
colcancer <- data.frame(
                  parent = c(rep("Root",3), "A", "B", "C"), # Parent nodes
                  child = c("A", "B", "D", "C", "E", "E"), ## Child nodes
                  s = c(0.5, 0.2, 0.1, rep(0.05, 3)),

                  sh = -0.3,

                  typeDep = "MN" ## Type of dependency
                   )
```

```r
## Fitness specification of the poset
colcancer_efec <- allFitnessEffects(
                  colcancer, # Poset

                  geneToModule = c( ## Specification of the modules
                                "Root" = "Root",
                                "A" = "APC",
                                "B" = "TP53, EVC2",
                                "C" = "KRAS",
                                "D" = "PI3KCA, EPHA",
                                "E" = "FBXW7, TCF7L2"),

                  drvNames = c( ## Specification of drivers
                                "APC", "TP53", "EVC2", "KRAS",
                                "PI3KCA", "EPHA", "FBXW7", "TCF7L2")
                  )



## DAG representation
plot(colcancer_efec, expandModules = TRUE, autofit = TRUE, lwdf = 2)
```

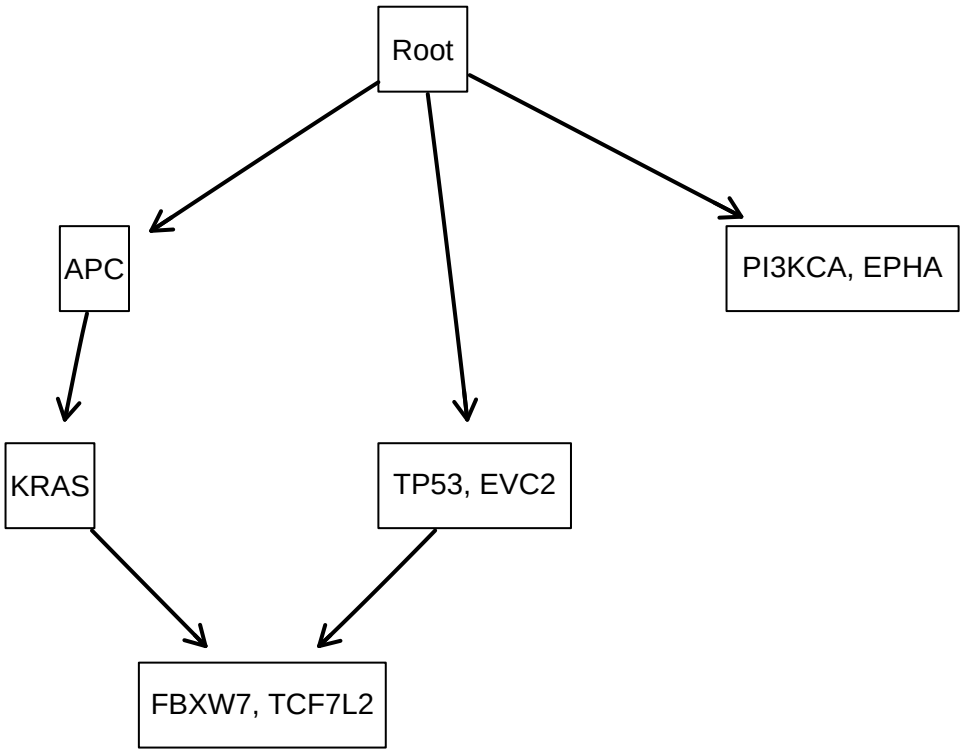

Figure 1: DAG from colorecta cancer

Figure 1 shows DAG derived from the generative model proposed by (1). From a wild type genotype

(depicted as "Root" in Figure 1), it is possible to follow three different paths. The wild type genotype can suffer a mutation in the `APC` gene or in the `TP53`, `EVC2` or `PI3KCA, EPHA` modules. These mutations occur independently, without the need for previous mutations to occur (these genes/modules derive from a "Root" node).

Mutation in `PI3KCA, EPHA` module is not mandatory for other mutations to occur, since this module is not connected to any gene or module. On the other hand, `APC` gene needs to be already altered for a mutation to occur in the `KRAS` gene. The `APC` gene is the parent node of the `KRAS` gene (**monotonic dependency**). Also, a mutation in the `KRAS` gene would be necessary for the mutation to appear in the `FBXW7`, `TCFL2` module. Alteration in the module `FBXW7, TCF7L2` requires both, the gene `KRAS` and the module `TP53, EVC2` to be already mutated.

```
colcancer_efec_FL <- evalAllGenotypes(colcancer_efec, max = 110000)
## Output is not shown due to size of the table.

## Plot of fitness landscape
plotFitnessLandscape(colcancer_efec_FL)
```

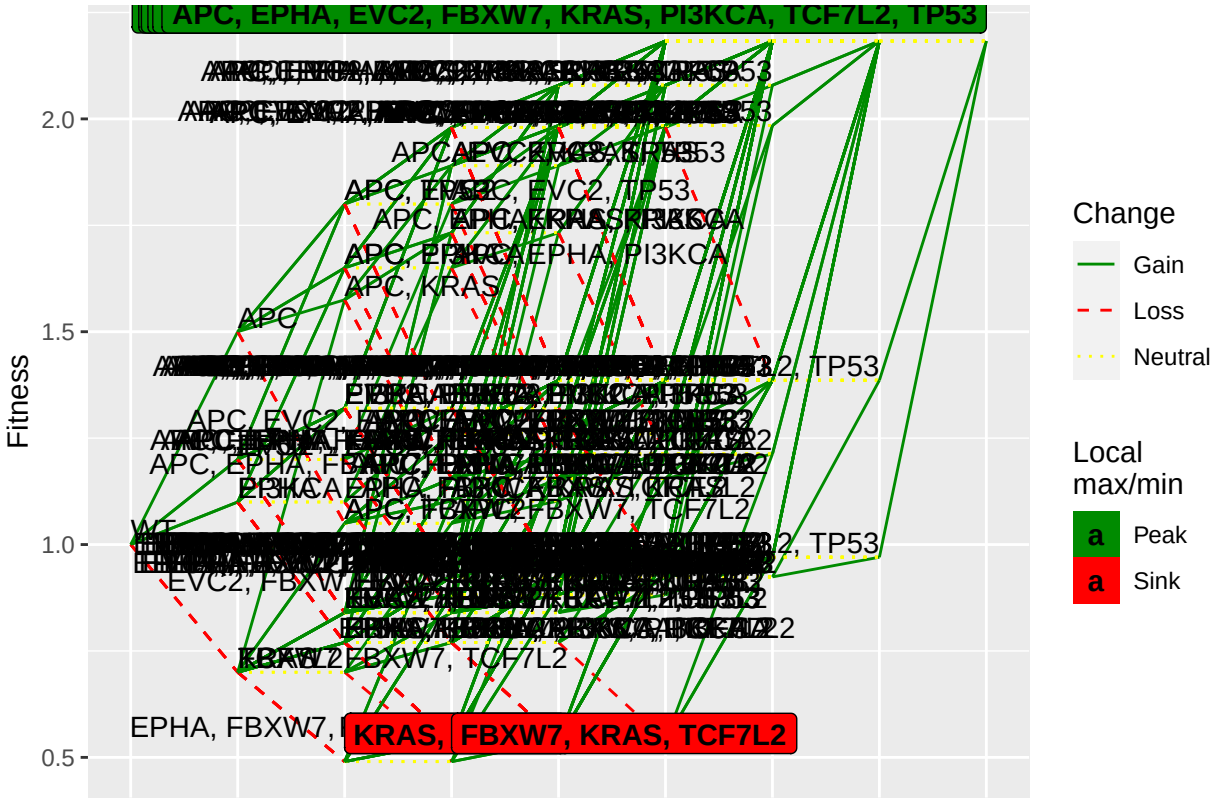

Figure 2: Fitness landscape from colorectal cancer

The function `evalAllGenotypes` is used to map genotypes to fitness values. Figure 2 shows the fitness landscape derived from the restrictions specified in Figure 1. It shows a busy fitness landscape due to the huge amount of possible genotypes combinations, each one with a different fitness value depending on whether or not the restrictions are satisfied.

2.1 Simplified cancer progression model

In order to properly visualize a fitness landscape, a simplified version of the model coded in section 2 is built. This model doesn't use modules, just individual genes. This approach will lead to clear fitness landscape and to properly identification of processes that may occur.

Authors (1) claim that there is a phenomena of mutual exclusivity between certain genes of specific pathways. Mutations of a certain genes may not be present in the genotype if another gene of the same pathway is already mutated. Moreover, mutation of one of the genes mutually excluded of a specific pathway is enough to provide all fitness contribution to the tumor cell. Therefore, mutation of just one gene of each module would lead to the same final fitness.

```
## Fitness specification of the simplified poset
Scolcancer <- allFitnessEffects(colcancer)
```

```
plot(Scolcancer)
```

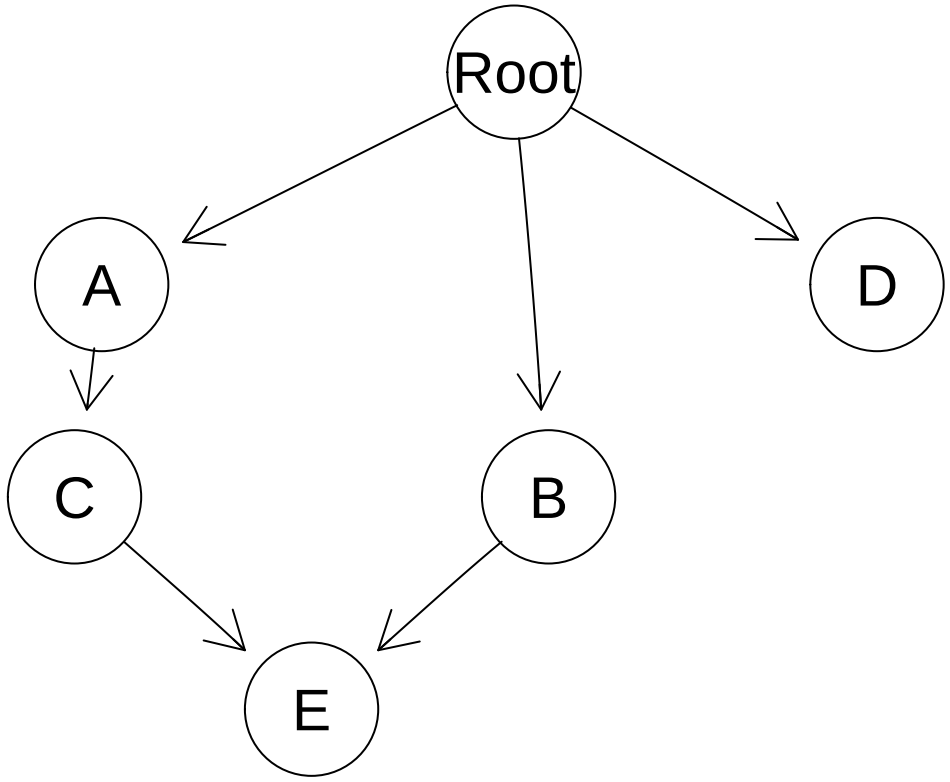

Figure 3: DAG from a simplified model of colorecta cancer

```
## Obtain all genotypes from the fitnessEffect
(Scolcancer_ge <- evalAllGenotypes(Scolcancer))
```

```
##         Genotype Fitness
## 1              A 1.50000
```

```
## 2              B 1.20000
## 3              C 0.70000
## 4              D 1.10000
## 5              E 0.70000
## 6           A, B 1.80000
## 7           A, C 1.57500
## 8           A, D 1.65000
## 9           A, E 1.05000
## 10          B, C 0.84000
## 11          B, D 1.32000
## 12          B, E 0.84000
## 13          C, D 0.77000
## 14          C, E 0.49000
## 15          D, E 0.77000
## 16       A, B, C 1.89000
## 17       A, B, D 1.98000
## 18       A, B, E 1.26000
## 19       A, C, D 1.73250
## 20       A, C, E 1.10250
## 21       A, D, E 1.15500
## 22       B, C, D 0.92400
## 23       B, C, E 0.88200
## 24       B, D, E 0.92400
## 25       C, D, E 0.53900
## 26    A, B, C, D 2.07900
## 27    A, B, C, E 1.98450
## 28    A, B, D, E 1.38600
## 29    A, C, D, E 1.21275
## 30    B, C, D, E 0.97020
## 31 A, B, C, D, E 2.18295
```

```
## Plot the fitness landscape.
plotFitnessLandscape(Scolcancer_ge,
                     use_ggrepel = TRUE)
```

DAG graph and fitness landscape of this simplified model are depicted in Figure 3 and Figure 4, respectively. DAG showed in Figure 3 is the same as the DAG depicted in Figure 1, but without expanding modules. In this case, there is not an improvement in legibility or clarity. However, if we compare the simplified fitness landscape (see Figure 4) with the previous fitness landscape (see Figure 2), there are a difference in clarity. In this new fitness landscape, it is possible to visualize the fitness given to each genotype. Fitness value associated to each genotype fits to the restrictions established in the DAG. For instance, genotype carrying all mutated genes constitutes the global maximum while genotypes carrying only mutations in "C" o "E" genes represent deviations from the monotonicity and constitute local minimums. Between maximum and minimum fitness values, there are different peaks and valleys showing genotypes with intermediate fitness values. Fitness landscape provide an evolutionary sense to cancer progression.

2.2 Simulating data from a simplified model

Restrictions set in DAG were used as a guide line to built the fitness landscape (see Figure 4). This fitness landscape shows each possible genotype as well as its fitness. This landscape can be used to simulate fitness evolution in cancer progression. The function `OncoSimulIndiv` is used to simulate colorectal tumor progression. This function simulates a single evolutionary path. It is necessary to include the poset with the order restrictions defined for the simplified model (see subsection 2.1). McFarland model is used for

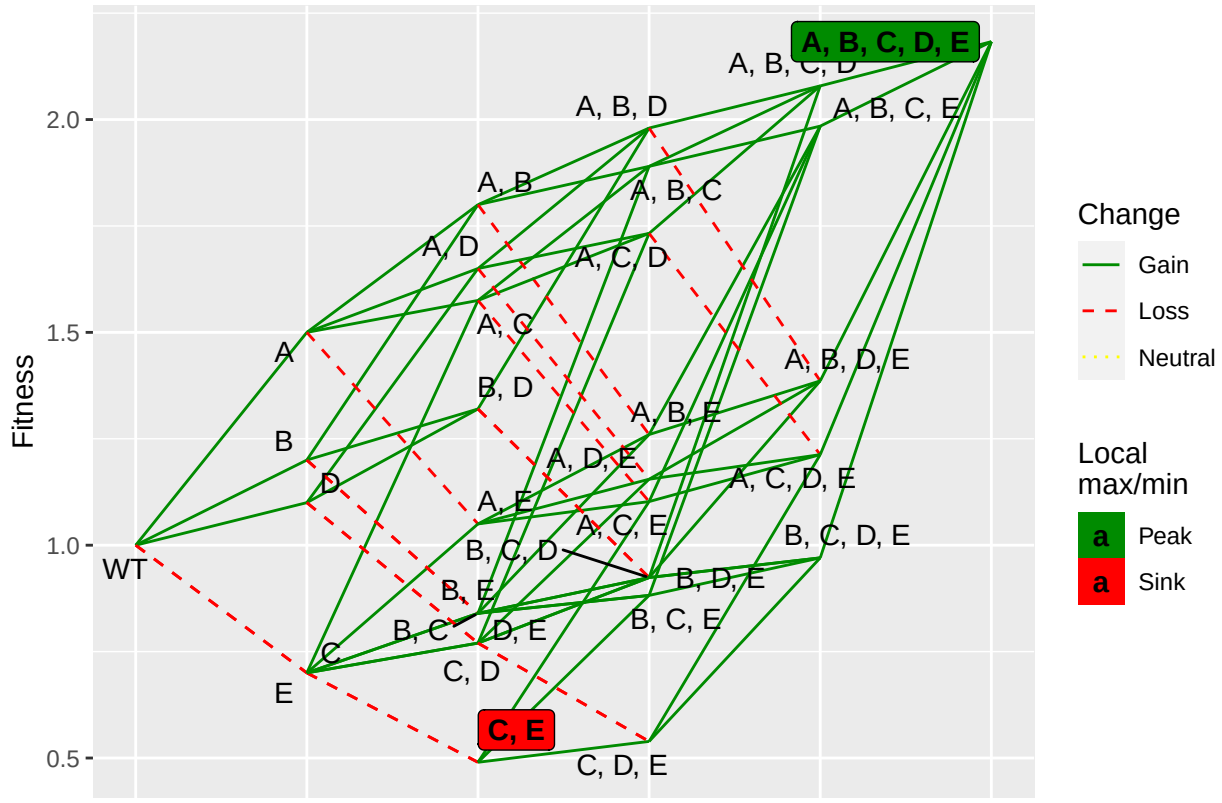

Figure 4: Fitness landscape from a simplified model of colorectal cancer

simulation of cancer progression. Initial population size is set at 400. Only one mutation rate is used, 1e-4. Final time is set to 500 to visualize clones' evolution. Furthermore, keepPhylog parameter is set true to plot the parent-child relationships occurring in the simulation as well as its frequency (`plotClonePhylog` function).

```
set.seed(35) ## Fix the seed for reproducibility

Simul <- oncoSimulIndiv(Scolcancer, ## A fitnessEffects object
                        model = "McFL", ## Model used
                        mu = 1e-4, ## Mutation rate
                        sampleEvery = 0.02, ## How often the whole population is sampled
                        keepEvery = 1,
                        initSize = 400, ## Initial population size
                        finalTime = 600,
                        keepPhylog = TRUE, ## Allow to see parent-child relationships
                        onlyCancer = FALSE
)


## Plot of simulation
plot(Simul, ## OncoSimulIndv model
     show = "genotypes",
     type = "stacked"

)
```

```
## Plot of simulation
plot(Simul, ## OncoSimulIndv model
     show = "genotypes",
     type = "line"
)
```

```
## Parent-child relationship derived from simulation
plotClonePhylog(Simul,
                N = 0, ## Specify clones that exist
                keepEvents = TRUE ## Arrows showing how many times each clones appeared


)
```

A stacked and line plot of the simulation is depicted in Figure 5 and Figure 6, respectively. Both plots show the genotype acquisition by time and the number of clones carrying that genotype in the simulation. Different cell populations coexist in different time moments, each carrying a different genotype and therefore, a different fitness.

In Figure 5, wild type genotype ("WT") progressively despairs while clones carrying a mutation in gene "A" arrive to the simulation. However, they are substituted by a new clone that also carries a mutation in gene "B". Then, this clone suffers different mutations resulting in the coexistence of different genotypes in the simulation, each one with a different survival rate. Finally, genotype carrying all mutations is stabilized in the tumor, since its fitness is the greatest among all the fitness of all genotypes.

On the other hand, Figure 6 shows the same information but it is possible to observe all genotypes generated in the simulation, even those that survive for little time. In addition to the genotypes seen in Figure 5,

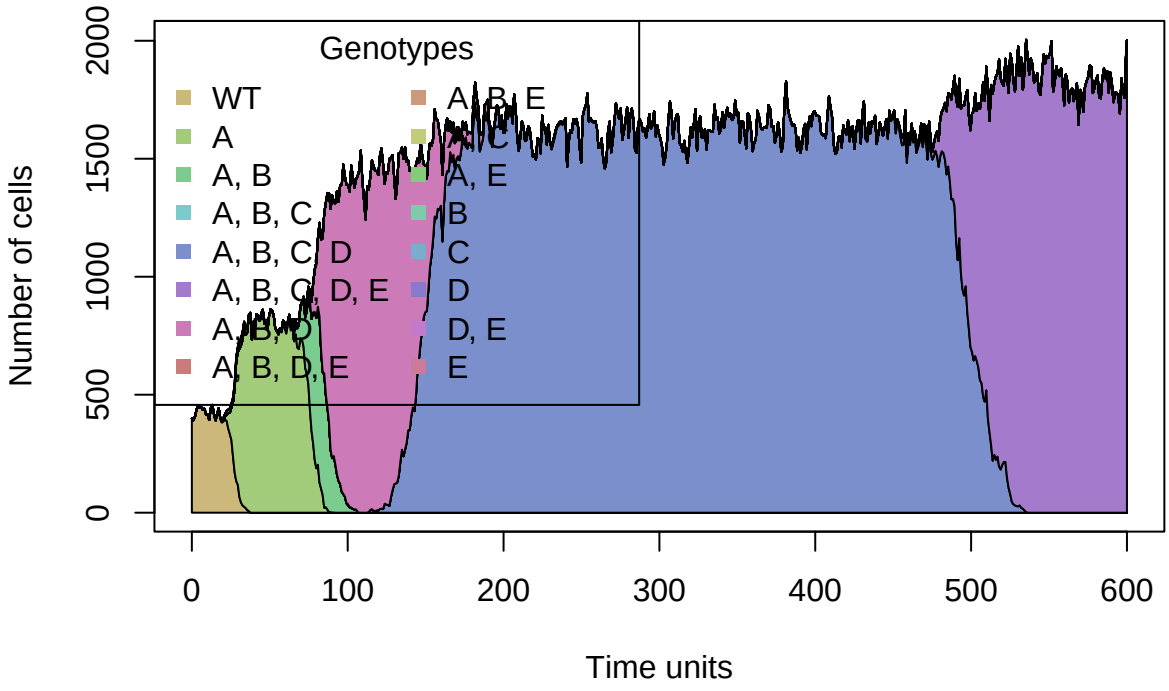

Figure 5: Simulation of cancer progression using the fitness landscape of the simplied model (stacked plot)

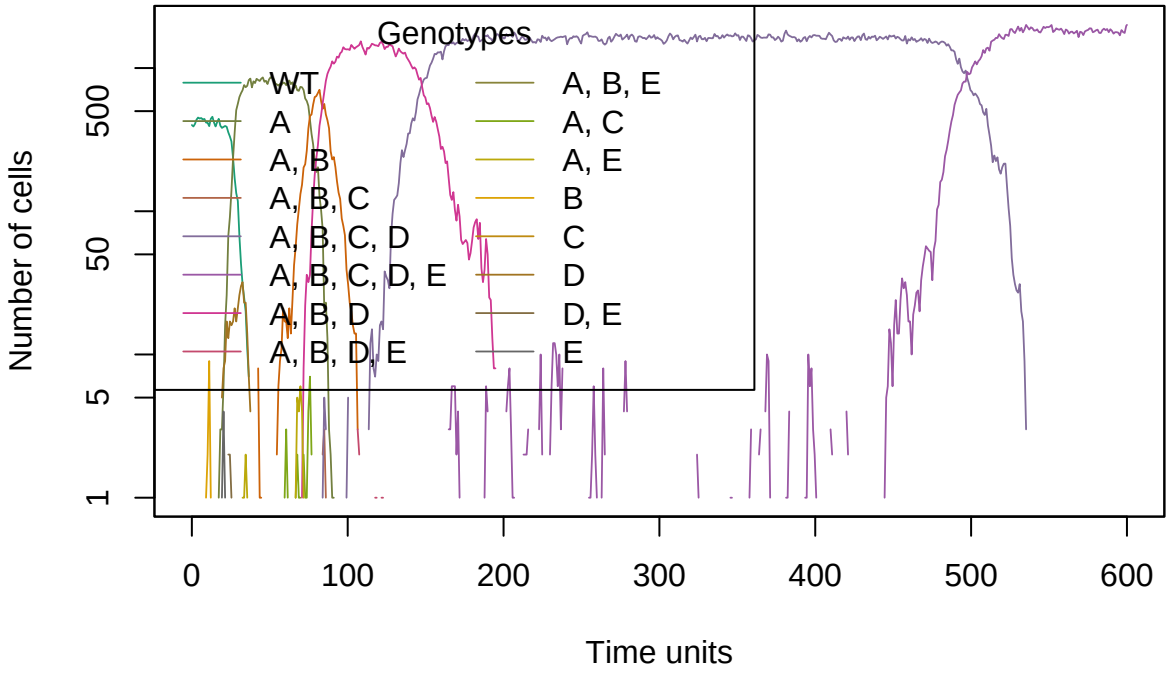

Figure 6: Simulation of cancer progression using the fitness landscape of the simplied model (line plot)

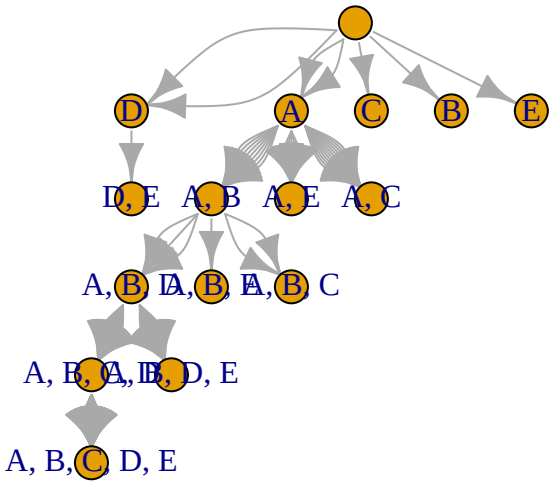

Figure 7: Parent-child relationship derived from simulation

some other genotypes appear in the cell culture, but due to selective pressure, they are not able to survive. Figure 7 shows the genealogical evolution of genotypes in the simulation. Arrows' width represent frequency of clone apparition. Widther arrows indicate a higher frequency of change from the parent genotype to the child genotype. Although Wild Type genotype mutates in every possible gene, only genotypes which first mutate in gene "A" survive. Finally, only the genotype carrying all the mutations together stabilized in the simulation.

2.3 Order effects

To explore order effects in cancer progression, a simple model derived from the restriction model inferred by (1) is created.

This simplified model just contains 3 genes: `APC`, `TP53` and `KRAS`, genes considered as `superdrivers` (6), meaning that are the main responsible for cancer progression since they provide a higher fitness gain than the other genes in the model. This conclusion is reached for the same colorectal cancer dataset as (1). Thus, it can be extrapolated to our case.

The relationships between those genes was previously depicted in section 2. In this case, we will set `APC` as the parent of `KRAS`. Both, `APC` and `TP53` have as parent `Root`. Based on the waiting time rate parameter λ, the fitness values of each possible order is given (see Table 1). λ is higher for `APC`, which means that it seems to appear before in the cancer progression. λ for `KRAS` is the lower between the three, meaning that it mutates the last. `TP53` mutation occurs between `APC` and `KRAS`. Order effects benefits this order, so clones suffering mutations in the previous order are favored with a higher fitness. Other possible paths of cancer progression are slightly less naturally selected (assumption based on (1)). Order effect is visualize using `evalAllGenotypes` function.

```
cc <- data.frame(parent = c(rep("Root", 2), "A"),
                 child = c("A", "C", "B"),
                 typeDep = "MN")

cc_order <- allFitnessEffects(
  orderEffects = c("A > B > C" = 0.5, "B > A > C" = 0.2,
                   "B > C > A" = 0.1,
                   "B > C" = 0.2,
                   "C > B" = 0.1,
                   "B > A" = 0.1,
                   "A > B" = 0.3),

  geneToModule =
    c("A" = "APC",
      "B" = "KRAS",
      "C" = "TP53") )

(cc_order_geno <- evalAllGenotypes(cc_order, order = TRUE))
```

```
##             Genotype Fitness
## 1                APC   1.000
## 2               KRAS   1.000
## 3               TP53   1.000
## 4         APC > KRAS   1.300
## 5         APC > TP53   1.000
## 6         KRAS > APC   1.100
## 7        KRAS > TP53   1.200
## 8         TP53 > APC   1.000
```

```
## 9        TP53 > KRAS   1.100
## 10 APC > KRAS > TP53   2.340
## 11 APC > TP53 > KRAS   1.430
## 12 KRAS > APC > TP53   1.584
## 13 KRAS > TP53 > APC   1.452
## 14 TP53 > APC > KRAS   1.430
## 15 TP53 > KRAS > APC   1.210
```

We obtain a table with the different possible genotypes as well as the order of appearance. However, this approach doesn't allow to generate neither a DAG nor a fitness landscape. Thus, is not possible to visualize the evolution of the genotypes with time. This is one limitation of `OncoSimulR package`, it doesn't allow to visualize those scenarios (yet).

```
#DAG
plot(cc_order)
```

```
## Error in `*tmp*`[[i]]: subíndice fuera de  los límites
```

```
# Fitness landscape
plotFitnessLandscape(cc_order_geno)
```

```
## Error in to_Fitness_Matrix(x, max_num_genotypes = max_num_genotypes): We cannot deal with order effe
```

Assuming a model where there is not an order effect, the final fitness value is the same for all the clones carrying all the mutations, regardless the path of mutation followed. A mutation in gene "B" followed by a mutation in gene "A" will reach the same fitness as if the mutation in gene "A" occurs first. However, in the model just generated, the order of the mutation affects the final fitness value reached by the tumoral clones. Previous alteration of some genes can lead to an evolutionary advantage.

3 References

1. Cristea S, Kuipers J, Beerenwinkel N. PathTiMEx: Joint Inference of Mutually Exclusive Cancer Pathways and Their Progression Dynamics. Journal of Computational Biology. 2017;24(6):603–15. doi: 10.1089/cmb.2016.0171

2. Raphael BJ, Vandin F. Simultaneous Inference of Cancer Pathways and Tumor Progression from Cross-Sectional Mutation Data. Journal of Computational Biology. 2015;22(6):510–27. doi: 10.1089/cmb.2014.0161

3. Neyshabouri MM, Jun SH, Lagergren J. Inferring tumor progression in large datasets. PLoS Computational Biology. 2020;16(10):1–16. Available from: http://dx.doi.org/10.1371/journal.pcbi.1008183

4. Diaz-Uriarte R. OncoSimulR: Genetic simulation with arbitrary epistasis and mutator genes in asexual populations. Bioinformatics. 2017;33(12):1898–9. doi: 10.1093/bioinformatics/btx077

5. Diaz-Uriarte R. Identifying restrictions in the order of accumulation of mutations during tumor progression: Effects of passengers, evolutionary models, and sampling. BMC Bioinformatics. 2015;16(1):1–26. doi: 10.1186/s12859-015-0466-7

6. Gerstung M, Eriksson N, Lin J, Vogelstein B, Beerenwinkel N. The temporal order of genetic and pathway alterations in tumorigenesis. PLoS ONE. 2011;6(10). doi: 10.1371/journal.pone.0027136

7. Wood LD, Parsons DW, Jones S, Lin J, Sjöblom T, Leary RJ, et al. The genomic landscapes of human breast and colorectal cancers. Science. 2007; doi: 10.1126/science.1145720

8. Diaz-Uriarte R. Cancer progression models and fitness landscapes: A many-to-many relationship. Bioinformatics. 2018;34(5):836–44. doi: 10.1093/bioinformatics/btx663