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

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

2021-01-17

Contents

*Universidad Autónoma de Madrid, Bionformatics and Computational Biology Master

1 Introduction

1.1 Cancer Progression Models

Cancer is an heterogeneous disease caused by the continuous accumulation of different somatic mutations during lifetime of an individual (1). This somatic mutations include Single Nucleotide Variants (SNV), Copy Number Variations (CPV) or insertions and deletions (INDELS). However, not all mutations contribute to malignant initiation or cancer progression. Therefore, identifying mutation leading to cancer progression becomes key to understand cancer development and treatment (3). Somatic mutations that affect the cells are classified in two main groups: *passenger* mutations and *driver* mutations. Passenger mutations are silent mutations which doesn't lead to any pathogenic situation (these are the main alterations affecting cells). On the other hand, driver mutations are responsible for cancer onset. Those mutations provide the cells with morphological and metabolic alterations that ultimately lead to a selective growth advantage over wild type cells from which they derived (4).

Driver mutations can affect both oncogenes and tumor suppressor genes, but conversely. Oncogenes are altered proto-oncogenes leading to an uncontrolled growth rate, division and survival of cells. Thus, specific mutations promoting over-activation of these genes will yield to cancer onset. On the other hand, tumor suppressor genes prevent unrestrained cellular growth and promote DNA repair by protein recruitment. Furthermore, lower or abnormal expression of these genes threat DNA repair mechanisms and growth homeostasis (5).

Cancer progression is caused by the accumulation of various mutations not just simply one mutation as occurs in monogenic diseases. Moreover, not all possible mutations order seem to be equally responsible for cancer progression. Indeed, some mutations require some previous alteration to happen (dependencies between genes). Therefore, it is necessary to know which are the constraints and restrictions that lead to cancer development. Models explaining those dependencies are called **Cancer Progression Models (CPMs)** (6). CPMs are depicted as Directed Acyclic Graphs (DAG). In this graph, nodes represent genes and arrows dependencies between them. The path refers to the order of mutations to cancer progression (7).

Although cancer progression is a dynamic process, tumor data is usually obtained as **cross-sectional** samples. It is a combination of snapshots taken from different tumor progression at different stages (2). Assuming that these observations reflect the same stochastic process, they can be used to infer restrictions between tumor events. This order of effects can be report as a direct graph. Knowing this constraints between genes, it allows to define a therapeutic target to avoid cancer progression (7). However, temporal progression of any cancer is difficult to infer by cross-sectional data. The utopian dataset to infer cancer progression is a *longitudinal dataset* consisting of same individual tumor samples from different time points during cancer development. Nevertheless, it is almost impossible without damaging individuals (1).

1.2 Evolutionary Models

(Just and idea) Mention the evolutionary models and how OncoSimulR allows deviations from monotonicity with genotype to fitness mapping

1.3 Order of effects

(Just and idea) Describe order of effects

1.4 Epistatic interactions

Mutual exclusivity is a common phenomena in cancer progression. It is defined as the situation where two events (mutations) occur less frequently than expected by chance (2). Two genes are mutually exclude if the presence of one of them avoids the presence of the other. In nature, there are two mechanisms that can lead to this phenomena:

- Synthetic lethality, where carrying both mutations is detrimental for the viability of the cell.
- Null effect, where whichever mutation occurs first involves most of the selective advantage and decrease the selective pressure to occur for the others.

1.4.1 Synthtetic Viability

(Just and idea) Describe sign synthetic viability

1.4.2 Mutual exclusivity

(Just and idea) Here mention: Mutual exclusivity for Null effects and synthetic lethality (reciprocal sign epistasis)

1.5 Frequency dependent fitness

(Just and idea) Describe frequency dependent fitness

2 pathTiMEx, a generative probabilistic graphical model of cancer progression

In (8), 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 (8) is used as an example of model to map. The colorectal cancer dataset used to built that model is obtained from (9). The poset restrictions proposed can be coded using the **OncoSimulR package** (10), concretely, the `allFitnessEffects` function. 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 `allFitnessEffects` function is used. It is the case of the *restriction table*. It specifies the dependencies between genes or modules (genes/modules parents and genes/module children). Parameter `s` and `sh` refers to a numeric vector with the fitness effect that applies if the relationship is or is not satisfied, respectively. Authors don't specify its 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

Although authors don't specify the sort of dependency, in this poset, a semimonotonic dependency is defined between modules B and C with E (`SM`), while a monotonic dependency is defined for the others (`MN`). Model will be represented as an **Diaciclic Direct Graph (DAG)** where arrows conecting genes or modules indicate direct dependencies or constraints between them (7).

```
## 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(rep(0.5, 3), rep(0.05, 3)),

                  sh = -0.5,

                  typeDep = c(rep("MN", 4), rep("SM", 2)) ## Type of dependency
                   )

## 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)
```

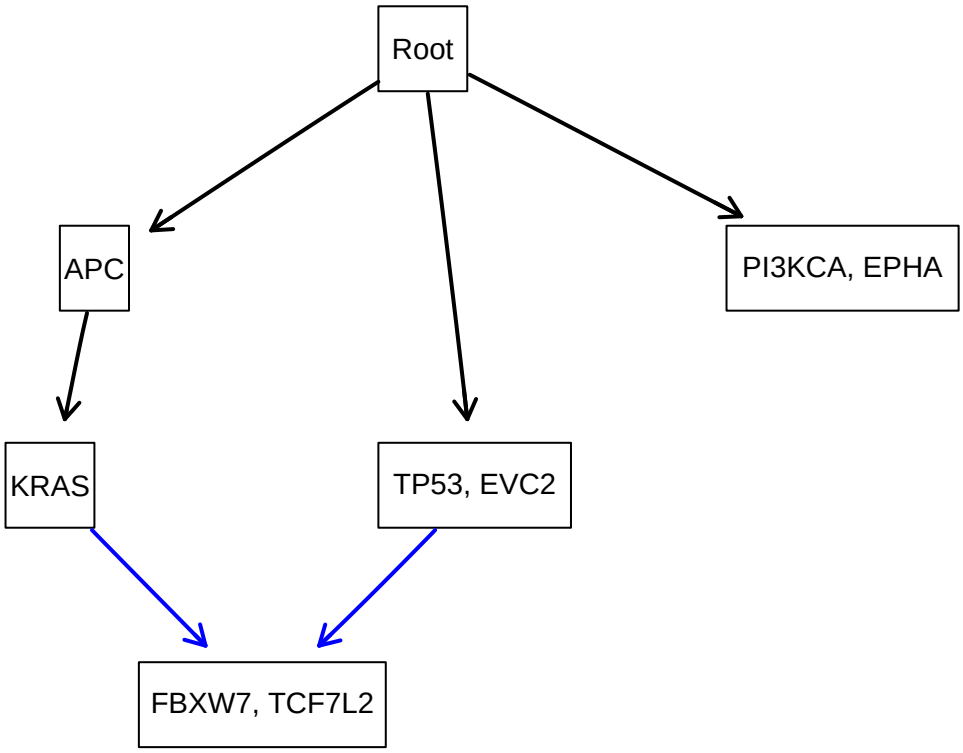

Figure 1: DAG from colorecta cancer

Model proposed by (8) indicates that, 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.

In the case of the `PI3KCA`, `EPHA` module, it is not essential for other mutated genes or modules to appear. On the other hand, the mutation in the APC gene is necessary for a mutation to occur in the `KRAS` gene. The `APC` gene is the parent node of the `KRAS` gene (**monotonicity dependency**). Also, the `KRAS` gene would be necessary for the mutation to appear in the `FBXW7`, `TCFL2` module. However, the relationship of the `KRAS`

gene to the FBXW7, TCF7L2 module has been defined as a **semitonicity dependency**. This implies that there is no need for a mutation in the TP53, EVC2 module in order for it to mutate. Similarly, a mutation in the TP53, EVC2 module would be sufficient for a mutation in the FBXW7, TCF7L2 module to occur (this semitonicity dependency is dipected in blue color in Figure 1).

In a DAG, only the genotypes that fulfill the restrictions defined by the arrows connecting the genes/modules can exist. Moreover, restrictions set in DAG do not contain any information about fitness of each individual genotypes, it is just a pathway to follow to cancer progression. On the other hand, fitness landscapes (or genotype-fitness maps) show the fitness associated to each genotype allowing to know the impact of a specific mutation. Furthermore, the restrictions reflected in the DAG just show sign epistasis between genes/modules. Epistasis is an effect where the phenotypic consequences of a certain mutation depend on the genetic background in wich it take place.

A specific type of epistasis is called *sign epistasis* and it refers to the case where a mutation yields to a increase (beneficial) or decrease (deleterious) of the fitness depending on the genotypic background of the cell. It has a different sign depending on the other genes mutated in the clone cell. Although this kind of epistasis can be depicted using a DAG, it can not show reciprocal sign epistasis, a specific situation where two individual mutations increase the fitness, although combine reduce it (synthetic lethality).

Hence, to visualize the relationships between genotypes and effects in fitness, the fitness landscape using the restrictions specified in Figure 1 is generated. For that aim, the `evalAllGenotypes` function is used. It returns a table with all the genotypes from the fitnessEffects description indicated as well as the genotype associated to it. The table obtained can be used as an object to plot a fitness landscape of the Figure 1.

```
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)
```

Fitness landscape obtained is displayed in Figure 2. It shows a quite busy fitness landscape due to the huge amount of possible genotypes combinations, each one with a different fitness value. However, it reflects genotype acquisition in terms of survival and adaptation for the cell, allowing to set what is the fitness associated to a clone that gets a certain mutations or a certain number of them.

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 constructed. This model doesn't use modules, just individual genes. This approach will lead to clear fitness landscape and to properly identify processes that may occur.

Authors (8) claim that there is a phenomena of mutual exclusivity between certain genes of specific pathways. Mutual exclusivity means that the presence of one gene in a specific pathway will be enough to fitness contribution, since mutation in the other genes of the same pathway are negative selected and therefore, the presence of the other gene in the same module can be discarded, since they will not mutate.

Therefore, the same model as in the previous case will be coded, but without specifying modules. Each module will be consider as an specific gene.

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

```
plot(Scolcancer)
```

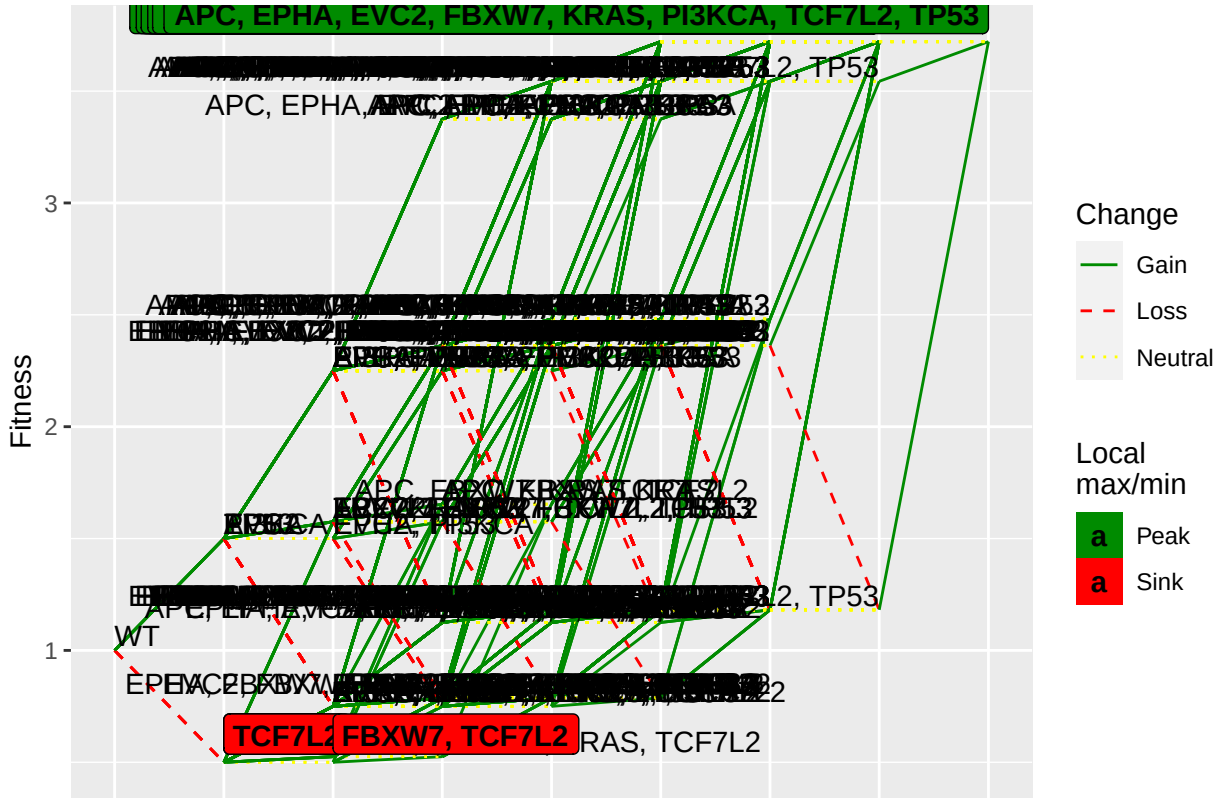

Figure 2: Fitness landscape from colorectal cancer

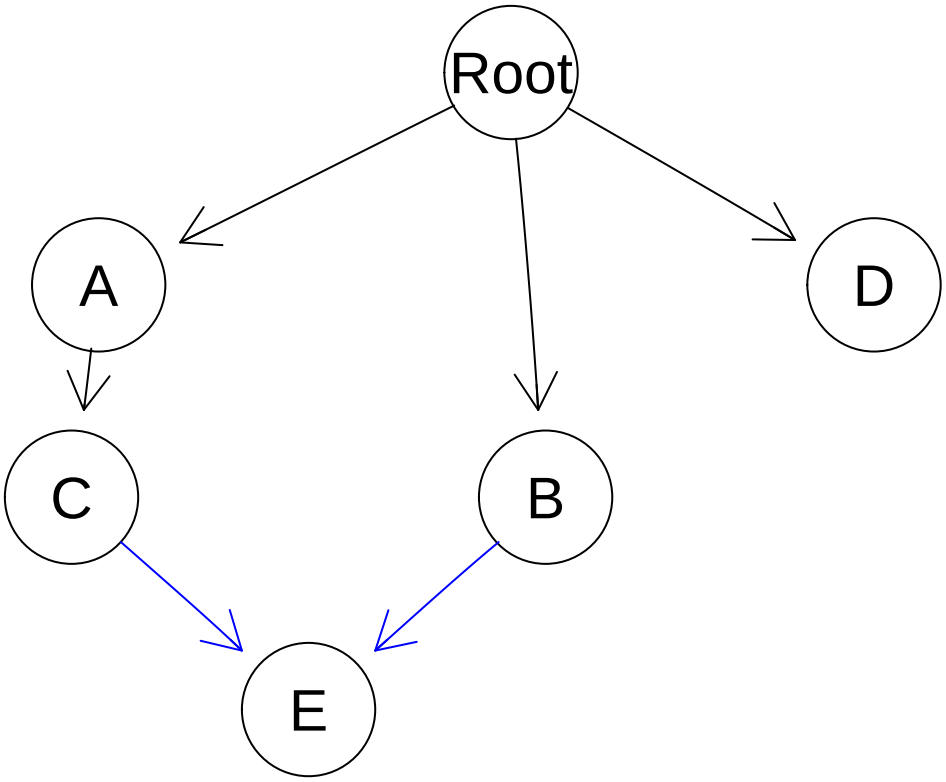

Figure 3: DAG from a simplified model of colorecta cancer

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

```
##          Genotype  Fitness
## 1               A 1.500000
## 2               B 1.500000
## 3               C 0.500000
## 4               D 1.500000
## 5               E 0.500000
## 6            A, B 2.250000
## 7            A, C 1.575000
## 8            A, D 2.250000
## 9            A, E 0.750000
## 10           B, C 0.750000
## 11           B, D 2.250000
## 12           B, E 1.575000
## 13           C, D 0.750000
## 14           C, E 0.525000
## 15           D, E 0.750000
## 16        A, B, C 2.362500
## 17        A, B, D 3.375000
## 18        A, B, E 2.362500
## 19        A, C, D 2.362500
## 20        A, C, E 1.653750
## 21        A, D, E 1.125000
## 22        B, C, D 1.125000
## 23        B, C, E 0.787500
## 24        B, D, E 2.362500
## 25        C, D, E 0.787500
## 26     A, B, C, D 3.543750
## 27     A, B, C, E 2.480625
## 28     A, B, D, E 3.543750
## 29     A, C, D, E 2.480625
## 30     B, C, D, E 1.181250
## 31 A, B, C, D, E 3.720938
```

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

DAG graph and fitness landscape of this simplified model is 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 fitness landscape obtained with the simplification (see Figure 4) with the previous fitness landscape (see Figure 2), there are a clear difference in clarity. In this new fitness landscape is possible to visualize the fitness given to each genotype and therefore, give an evolutionary sight to the model.

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. `OncoSimulIndiv` function is used to simulate colorectal tumor progression.

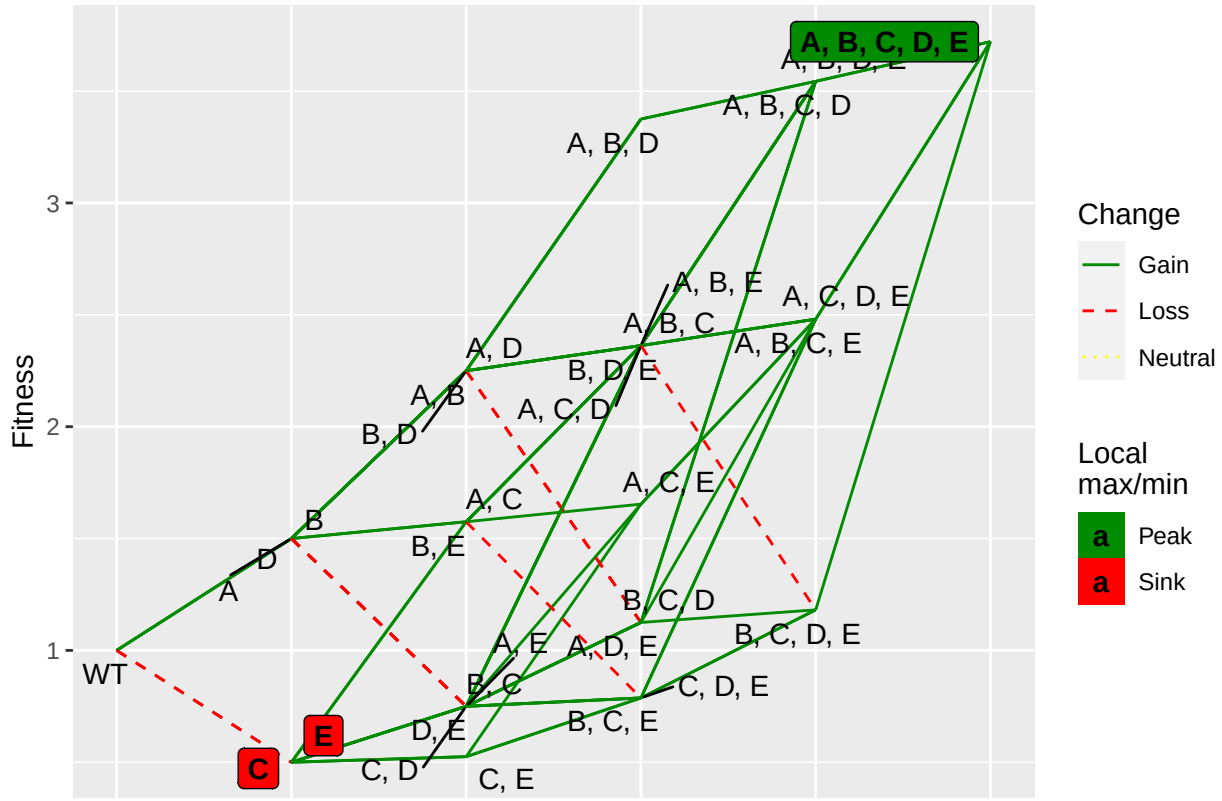

Figure 4: Fitness landscape from a simplified model of colorectal cancer

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 (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 (6). Initial population size is set at 400. Only one mutation rate is used: 1e-4. Final time is set to 220 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(257) ## 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 = 220,
                        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 cell culture. Different cell population converge in different time moments, each carrying a diferent 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 culture. However, they are substituted by a new clone that also carries a mutation in "B". Then, this clone suffers different mutations resulting in the coexistence of different genotypes in the cell culture, each one with a different fitness. Finally, genotype carrying all genotypes is stabilized in the culture. As it fitness is the greatest among all the fitness of all genotypes. It is obvious that it is more selective favored and it will overcome the concurrence with the other genotypes.

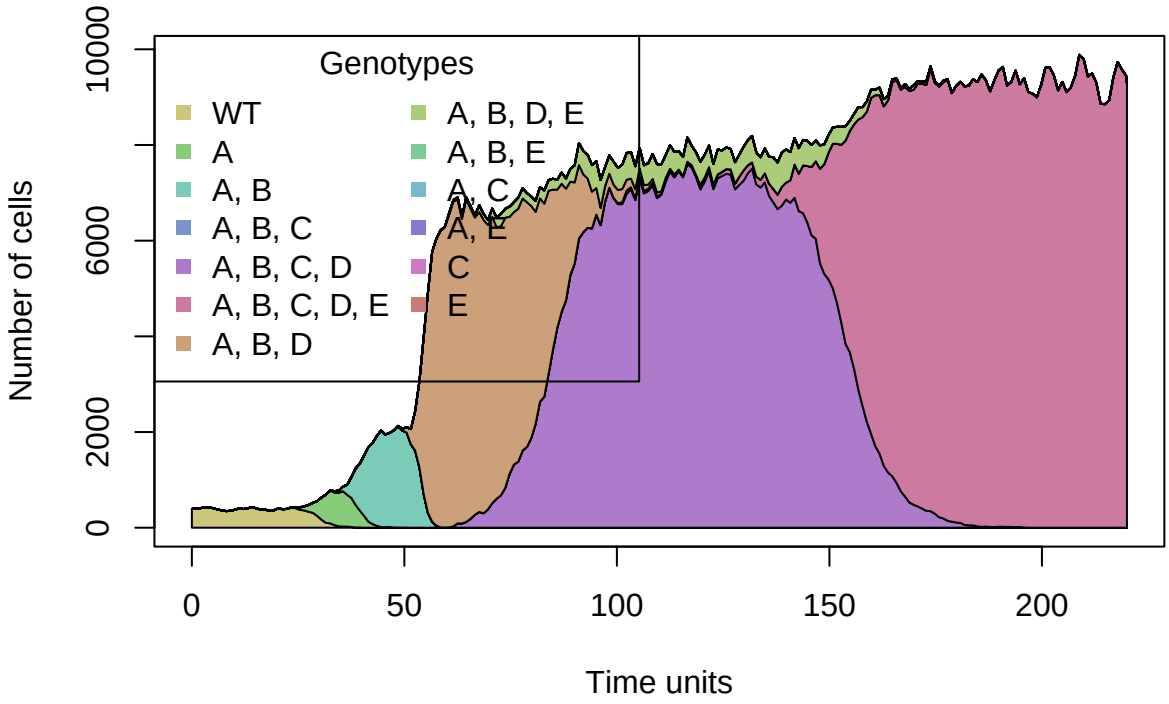

Figure 5: Simulation of cancer progression using the fitness landscape of the simplied model (stacked plot)

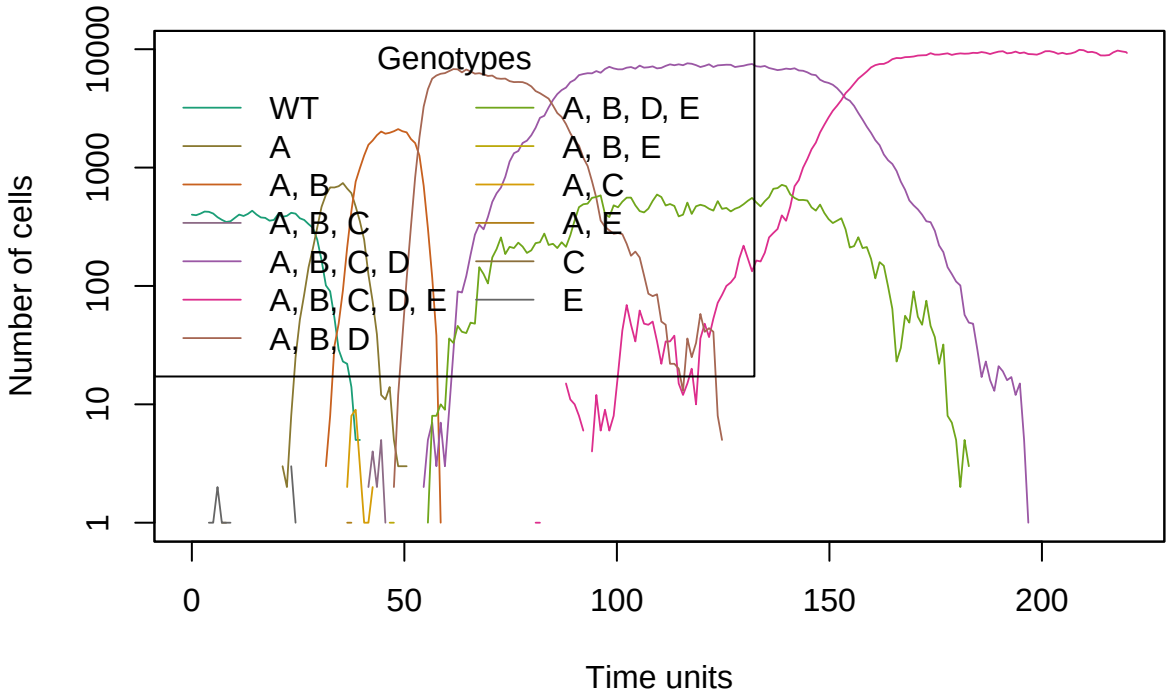

Figure 6: Simulation of cancer progression using the fitness landscape of the simplied model (line plot)

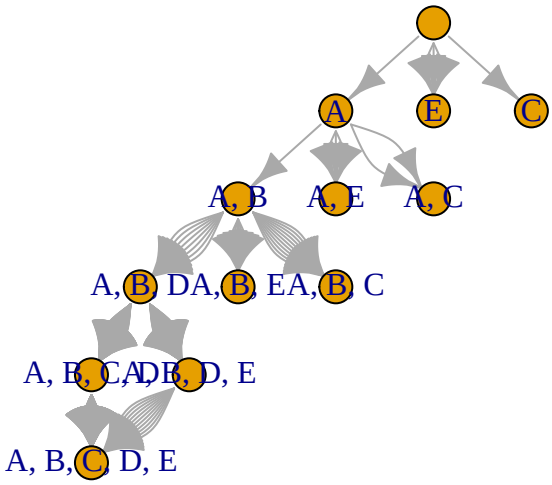

Figure 7: Parent-child relationship derived from simulation

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, some other genotypes appear in the cell culture, but due to selective pressure, they are not able to survive.

Hence, genotypes observed in the simulation not only follow the restrictions set in DAG of section 2. It is just a generative model where dependencies between genes are defined, but they may not occur in real life in that order. Moreover, this model is constructed using cross-sectional data, tumor snapshots in a specific time of cancer progression, it is not a temporal dataset from the onset to the end of the cancer progression.

Figure 7 shows the genotype evolution in the simulation. Arrows' width represent frequency of clone creation. Widther arrows indicate a higher frequency of change from the parent genotype to the child genotype.

2.3 Order effects

To explore order effects in cancer progression, a simple model derived from the restriction model inferred by (8) is created.

This simplified model just contains 3 genes: `APC`, `TP53` and `KRAS`, genes considered as `superdrivers` (11), 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 obtained from its article where they used the same colorectal cancer dataset as (8). 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 are defined following this criteria: 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 (8)). 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 neither a fitness landscape. Thus, is not possible to visualize the evolution of the genotypes with time.

```
#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, 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 impacts the final fitness reached by the tumoral clone. Since, the previous alteration of some genes before can lead to an evolutionary advantage.

In a non order effect model, the final fitness value is the same for all the clones, while this is unlikely to happen in real life. Clones carrying a certain mutation from the beginning would survive easily than those reaching the same genotype in a different order.

This is one limitation of `OncoSimulR package`, it doesn't allow to visualize those scenarios (yet).

2.4 Epistasis to simulate order effects

Epistasis assume that there is a dependence between genotypes. The effect of a mutation depends on the genetic background in which it happens (12). Now, we will cope with dependencies between genes using epistasis.

For that, we will use the same model described in subsection 2.3. As explained before, it is supposed to be a certain cancer progression restriction and therefore, the fitness values given to each different genotype is based in that criteria.

```
## Fitness object defined using epistasias
cc_epi <- allFitnessEffects(epistasis =
                              c("A: -B: -C" = 0.4,
                                "-A: B: -C" = -0.4,
                                "-A: -B: C" = 0.3,
```

```
                                  "A: B: -C" = 0.8,
                                  "A : B: C" = 1.4,
                                  "-A: B: C" = 0.1,
                                  "A : -B: C" = 0.5
                                  ),
                                geneToModule =
                                  c("A" = "APC",
                                  "B" = "KRAS",
                                  "C" = "TP53")
                                )

## DAG (epistasis)
plot(cc_epi, expandModules = TRUE, autofit = TRUE)
```

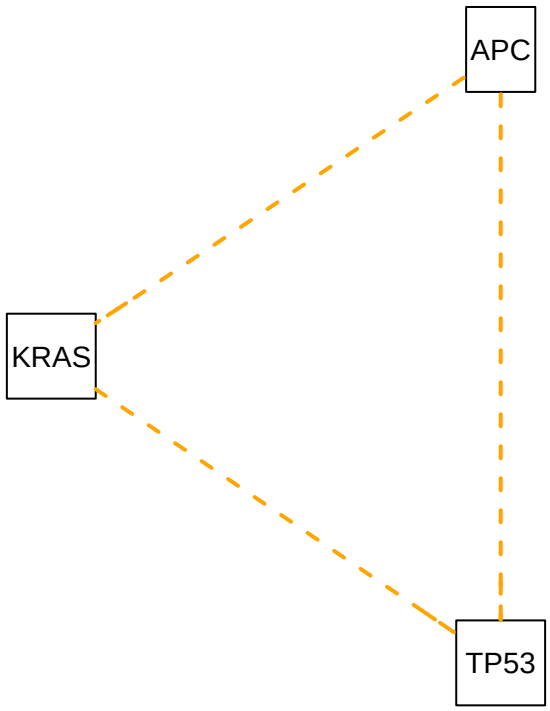

Figure 8: DAG showing epistasis between genes

```
## Genotypes derived from fitness defined with epistasia relationships
(cc_epi_geno <- evalAllGenotypes(cc_epi ))
```

```
##          Genotype Fitness
## 1             APC     1.4
## 2            KRAS     0.6
## 3            TP53     1.3
## 4       APC, KRAS     1.8
```

```
## 5       APC, TP53     1.5
## 6      KRAS, TP53     1.1
## 7 APC, KRAS, TP53     2.4
```

```
## Fitness landscape from this relationships
plotFitnessLandscape(cc_epi_geno, use_ggrepel = TRUE)
```

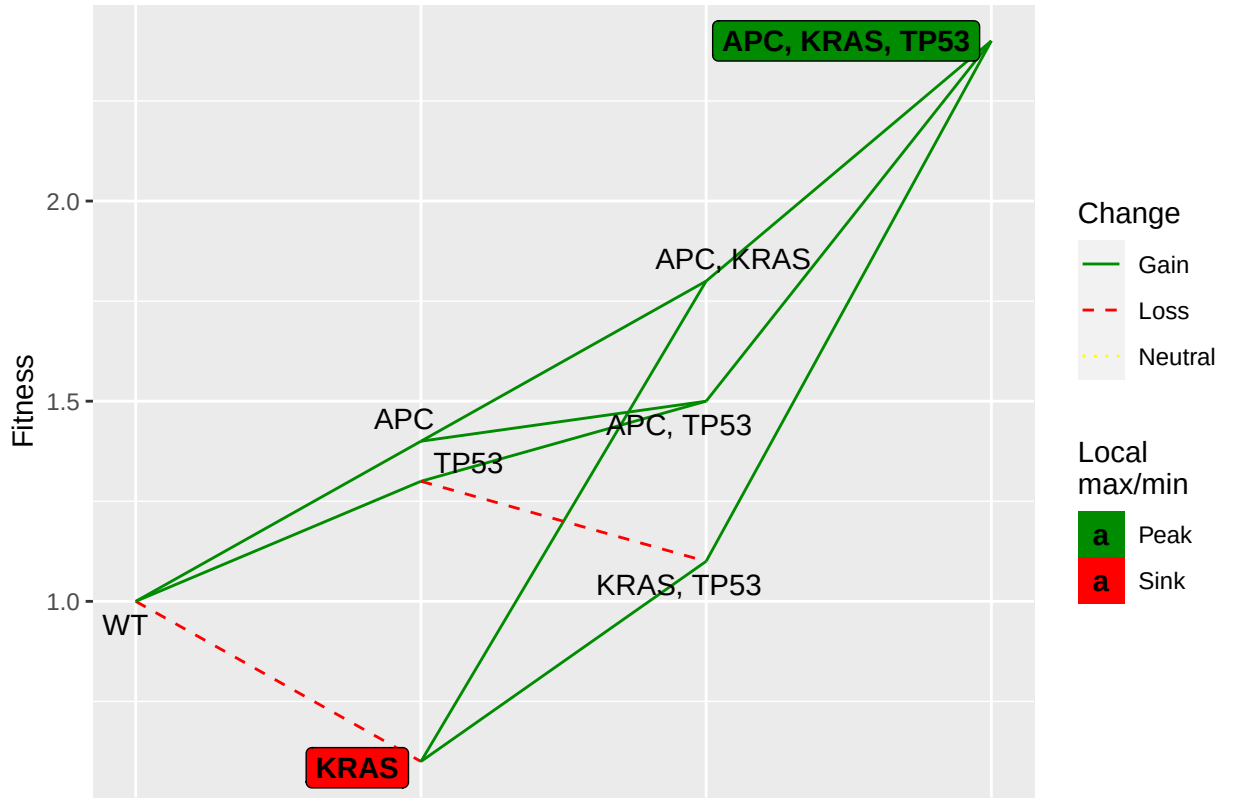

Figure 9: Fitness landscape of model defined by epistasis

Using this approach, it is possible to visualize the DAG (see Figure 8). In this case, there are discontinues yellow lines connecting each gene. This lines indicate a dependence between them. Fitness landscape is also plotted (see Figure 9).

With this model, we promote the clones of tumoral cells beginning with a first mutation in `APC`. Conversely, other clones not starting with that mutation (`KRAS` or `TP53`) have a lower fitness value. On the other hand, all genotypes end with the same fitness, but it is selective favored clones following the order defined in subsection 2.3. Hence, epistasis cannot be used to define order effects, but it can be used as a temporal solution.

3 Pathway Linear Progression Model: Raphael & Vanding, 2015

The Pathway Linear Progression Model (PLPM) described in (1) introduces the idea that driver mutations target pathways. This is an important concept since different individuals have driver mutations in different

genes that affect the same pathway (1). Therefore, the order in which mutations arise are better described at the pathway level instead of a gene level (1).

Moreover, it has been previously described that there is a tendency of mutually exclusive mutations of genes participating in the same pathway and a tendency of co-occurring (overlapping) mutations of genes that affect different pathways (13). Indeed, several cancer genes participate in multiple pathways and mutually exclusive events may allow for overlapping pathways (2). DISCUSS THIS LATER, MAYBE MODELS OF EVOLUTION IN ONCOSIMULR ALLOW FOR OVERLAPPING PATTERNS?

Here, we mapped the progression model from colorectal cancer data inferred by (1) (originally described in (9)) into an evolutionary model, allowing deviations from the restriction imposed in the DAG. For this, we used a vector `s` to indicate the fitness effects when the restrictions are satisfied and a vector `sh` for deviations.

In (1), the authors analyzed eight genes: APC, EPHA3, EVC2, FBXW7, KRAS, PIK3CA, TCF7L2, and TP53. In this model, APC mutations is an early event, followed by mutations in TP53 and PIK3CA (mutually exclusive). KRAS mutations appear after TP53/PIK3CA mutations.

We used the `allFitnessEffects` function to define the nodes and their relationships. Moreover, we used modules to represent mutually exclusive genes that affect the same pathway. Assigned fitness effects (`s`) values were higher for earlier mutations and lower for late mutation, since an earlier mutation in more prevalent in the clonal population than a later mutation, as explained in (13). A single negative value was set for deviations from restrictions (`sh`) and a monotic relationship (`MN`) was used for relationships between nodes of the DAG since nodes have only one parent.

Figure 10 shows the DAG inferred by (1) mapped to an evolutionary model that allows deviation from restrictions. Note that genes within a module are mutually exclusive and the restrictions goes top-down (i.e. from the root to the later mutation).

```
## Define poset restrictions, mapping of genes to modules, and driver genes
CRC_W <- allFitnessEffects(data.frame(parent = c("Root", "A", "B", "C"),
                                      child = c("A", "B", "C", "D"),
                                      s = c(0.6, 0.4, 0.1, 0.05),
                                      sh = -0.5,
                                      typeDep = "MN"),
                           geneToModule = c("Root" = "Root",
                                            "A" = "APC, EPHA3, TCF7L2",
                                            "B" = "EVC2, PIK3CA, TP53",
                                            "C" = "KRAS",
                                            "D" = "FBXW7"),
                           drvNames = c("APC", "EPHA3", "TCF7L2", "EVC2", "PIK3CA",
                                        "TP53", "KRAS", "FBXW7"))

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

The function `evalAllGenotypes` was used to map genotypes to fitness values. Figure 11 shows the fitness landscape inferred from the DAG of Figure 10. As mentioned before, this fitness landscape with eight genes is difficult to visualize. Nevertheless, we can give a general description of the topology of the landscape. Note that there are multiple peaks and valleys, suggesting a high degree of ruggedness. Moreover, note that KRAS constitutes a local minima. This results confirms the order of restrictions imposed by the DAG. It is important to mention that some genotypes in the local maxima are composed of genes that belong to the same module. Such result may reflect driver genes that are mutually exclusive and participate in multiple pathways, previously described as overlapping pathways (13).

```
## Map genotypes to fitness
CRC_F <- evalAllGenotypes(CRC_W, order = FALSE, addwt = TRUE)
```

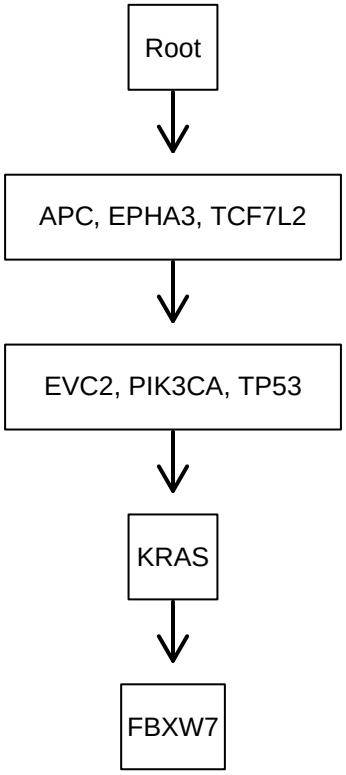

Figure 10: DAG from colorectal cancer dataset

```
## Plot of fitness landscape

plot(CRC_F)
```

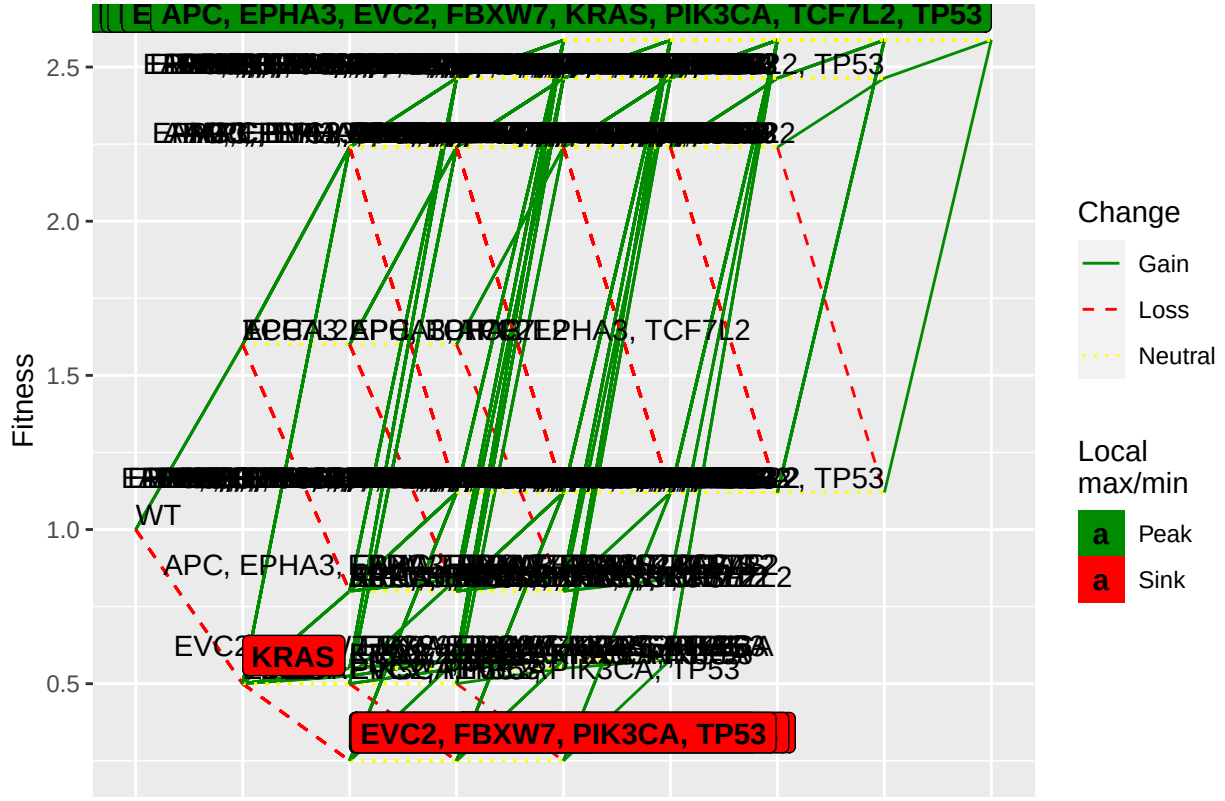

Figure 11: Fitness landscape inferred from colorectal cancer DAG

3.1 Simplified Model

Given that our initial DAG contains eight genes, then number of possible genotypes is $2^8 = 256$ which makes difficult to visualize the fitness landscape. For this, reason a smaller number of genes will be used to build a slightly different DAG to model other interesting scenarios (see Figure 12). The idea is to represent mutual exclusivity with a `XOR` relationship (red edges). Also, note that the fitness value for mutual exclusive genes (APC and TP53) is almost the same.

```
## Simplified model
## Define poset restrictions, mapping of genes to modules, and driver genes
CRC_W2 <- allFitnessEffects(data.frame(parent = c(rep("Root", 2), "A", "B", "C"),
                                       child = c("A", "B", rep("C", 2), "D"),
                                       s = c(0.2, 0.1, rep(0.05, 2), 0.01),
                                       sh = -0.5,
                                       typeDep = c(rep("XMPN", 4), "MN")),
                            geneToModule = c("Root" = "Root",
                                             "A" = "APC,",
```

```
                                          "B" = "TP53",
                                          "C" = "KRAS",
                                          "D" = "FBXW7"),
                              drvNames = c("APC", "TP53", "KRAS", "FBXW7"))

# DAG representation
plot(CRC_W2, expandModules = TRUE, autofit = TRUE, lwdf = 2)
```

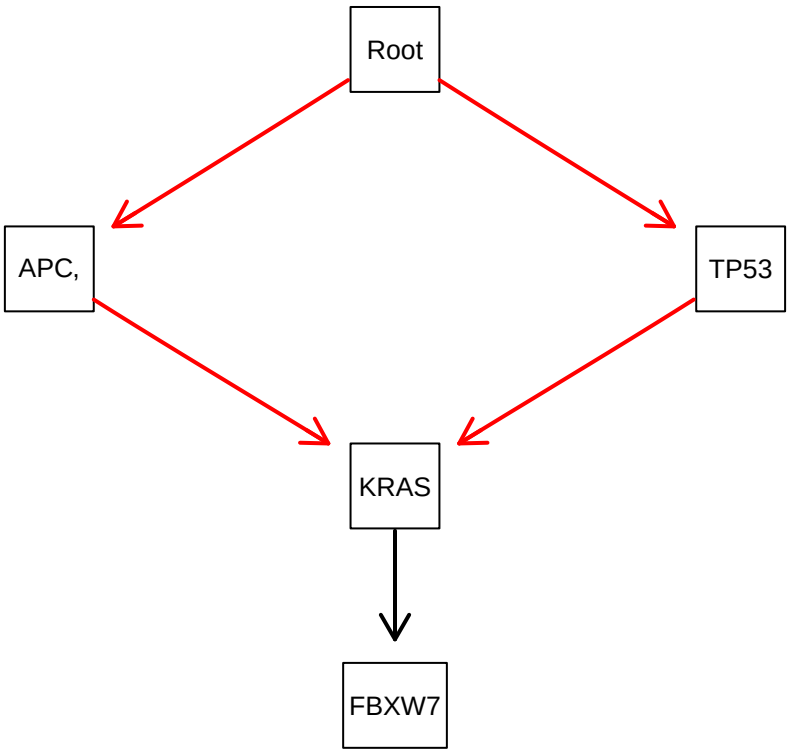

Figure 12: Simplified model from colorectal cancer DAG

Since only four genes are used in the DAG, then the possible number of genotypes is $2^4 = 16$, which are easier to interpret in a fitness landscape (see Figure 13). Now, the genotypes with the highest fitness are the ones that fulfill the order of restrictions imposed by the DAG (e.g. APC, FBXW7, KRAS - APC, FBXW7, KRAS, TP53). On the other hand, genotypes that deviates from the imposed restrictions have the lowest fitness (e.g. KRAS - FBXW7 - APC, KRAS, TP53). However, specifying mutual exclusivity with `XOR` relationships cannot capture null effect or synthetic lethality between APC and TP53. Also, if an `AND` relationship is defined from the Root to APC and TP53, then there is no change in fitness values.

```
## Simplified Model
## Map genotypes to fitness
(CRC_F2 <- evalAllGenotypes(CRC_W2, order = FALSE, addwt = TRUE))
```

```
##                  Genotype Fitness
## 1                      WT 1.00000
```

```
## 2                    APC 1.20000
## 3                  FBXW7 0.50000
## 4                   KRAS 0.50000
## 5                   TP53 1.10000
## 6              APC, FBXW7 0.60000
## 7               APC, KRAS 1.26000
## 8               APC, TP53 1.32000
## 9             FBXW7, KRAS 0.50500
## 10            FBXW7, TP53 0.55000
## 11             KRAS, TP53 1.15500
## 12       APC, FBXW7, KRAS 1.27260
## 13       APC, FBXW7, TP53 0.66000
## 14        APC, KRAS, TP53 0.66000
## 15      FBXW7, KRAS, TP53 1.16655
## 16 APC, FBXW7, KRAS, TP53 0.66660
```

```
## Plot of fitness landscap

plot(CRC_F2, use_ggrepel = TRUE)
```

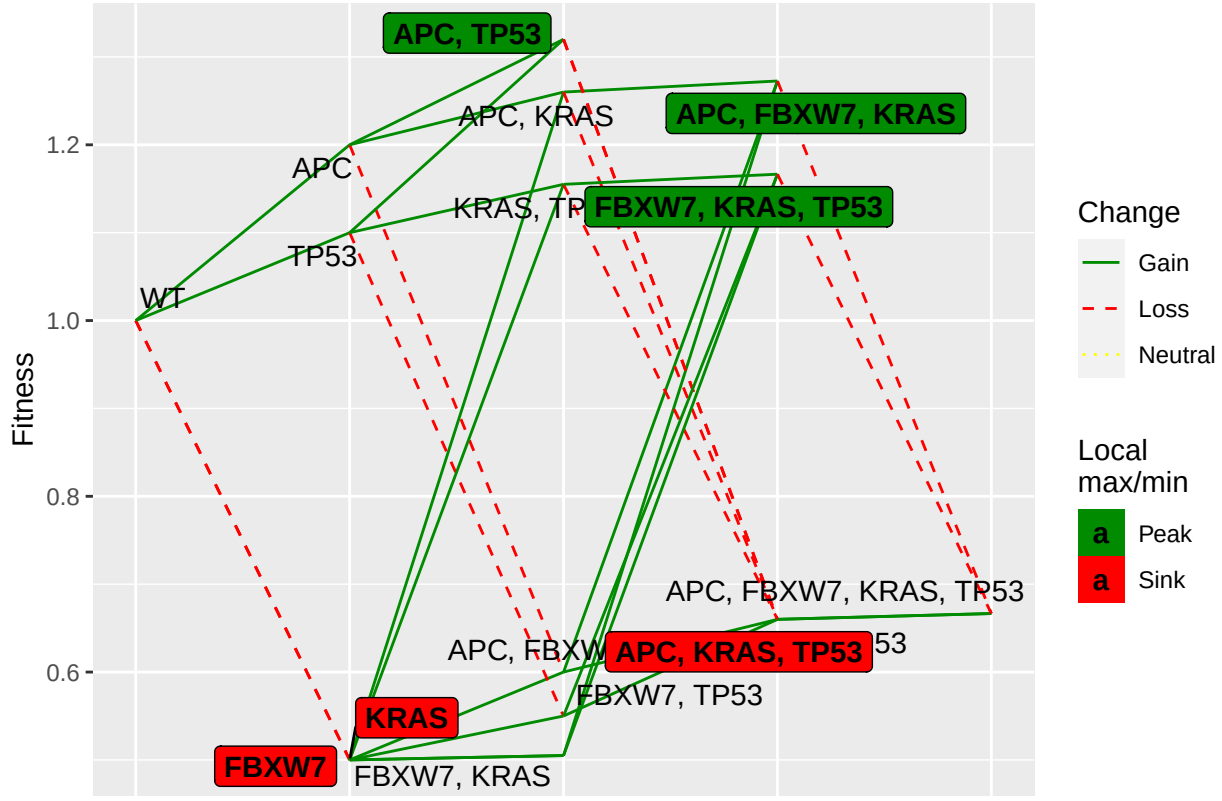

Figure 13: Fitness landscape from simplified model

3.2 SImulating Data from Simplified Model

Fitness effects and restrictions defined in the DAG from Figure 12 from previous section was used to simulate clonal evolution. The same parameters from subsection 2.2, except `initSize` and `finalTime`, were set in the `OncoSimulIndiv` function. Figure 14 shows the genotypes during that arises during clonal evolution. The genotype APC, TP53 fixates quickly in the clonal population. This result supports the fitness value for APC, FBXW7, KRAS depicted in Figure 13, since that genotype is one of the local maxima. A more detailed order of genotype appearances and extinctions is shown in Figure 15. Note that not all the 16 genotypes appear in the simulation because the best fitted genotype fixates rapidly in the population, leading to the extinction of some genotypes, whereas other cannot even appear. When simulation were executed with 'onlyCancer = TRUE, cancer is never reached, although the fixated genotype is the global maxima of the fitness landscape (see Figure 16 and Figure 17).

```
## Fix the seed for reproducibility
set.seed(87)


CRC_W2_S <- oncoSimulIndiv(CRC_W2, ## 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 = 2000, ## Initial population size
  finalTime = 2000,
  keepPhylog = TRUE, ## Allow to see parent-child relationships
  onlyCancer = FALSE)

## Plot of simulation for genotypes
plot(CRC_W2_S,
show = "genotypes",
type = "stacked")
```

```
## Plot of simulation for genotypes
plot(CRC_W2_S,
show = "genotypes",
legend.ncols = 2,
xlim = c(0, 1500),
type = "line")
```

```
## Fix the seed for reproducibility
set.seed(52)


CRC_W2_S1 <- oncoSimulIndiv(CRC_W2, ## 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 = 2000, ## Initial population size
  finalTime = 800,
  keepPhylog = TRUE, ## Allow to see parent-child relationships
  onlyCancer = TRUE,
```

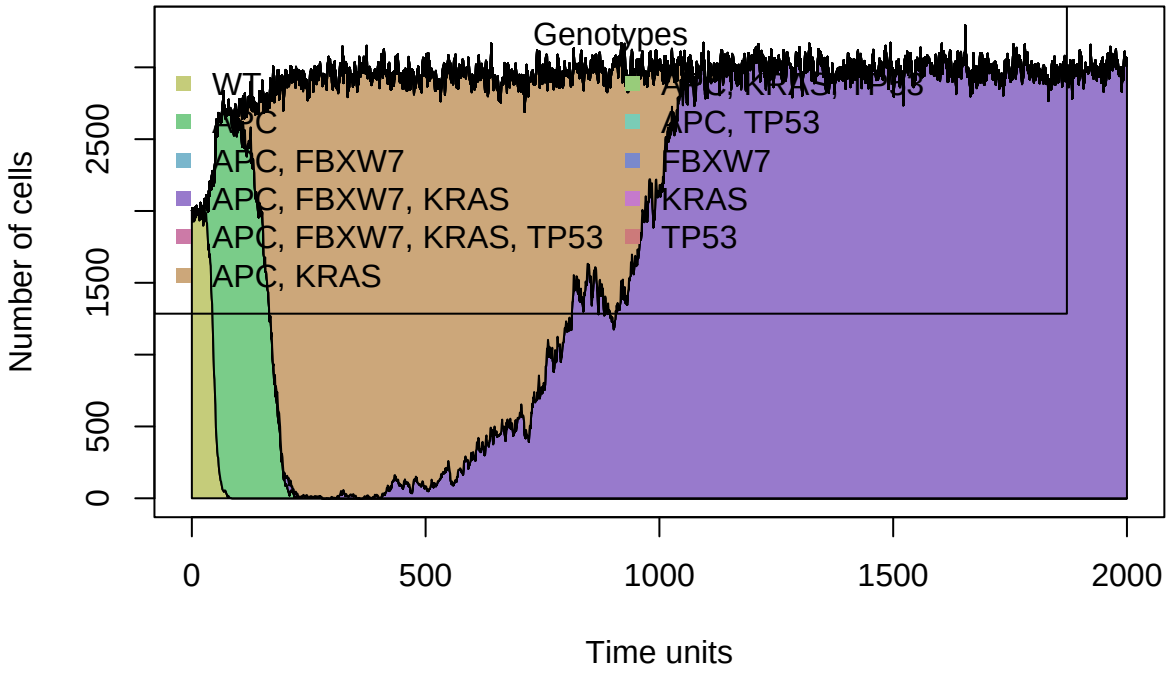

Figure 14: Simulation of cancer progression for the simplified model. Genotypes are shown stacked

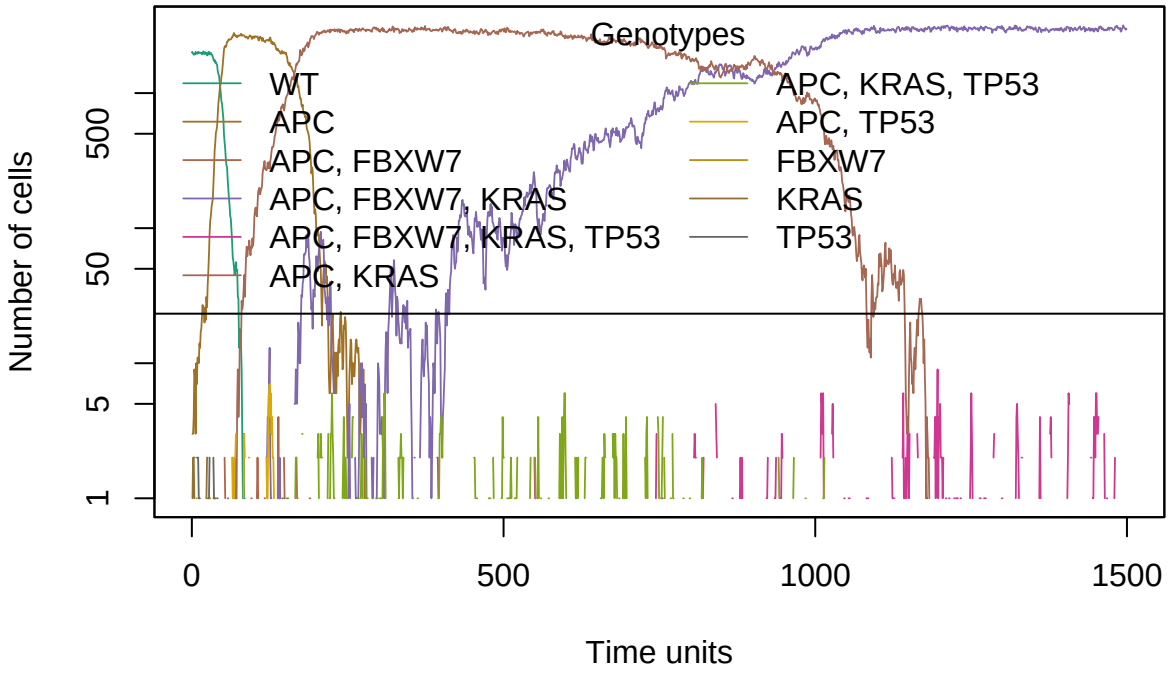

Figure 15: Simulation of cancer progression for the simplified model. Genotypes are shown as lines.

```
  errorHitWallTime = FALSE, ## See results even if stopping conditions are not met
  errorHitMaxTries = FALSE)
```

```
##
##  Hitted maxtries. Exiting.
```

```
## Time to reach cancer
(CRC_W2_S1$FinalTime)
```

```
## [1] 800
```

```
## Plot of simulation for genotypes
plot(CRC_W2_S1,
show = "genotypes",
legend.ncols = 2,
xlim = c(0, 500),
type = "stacked")
```

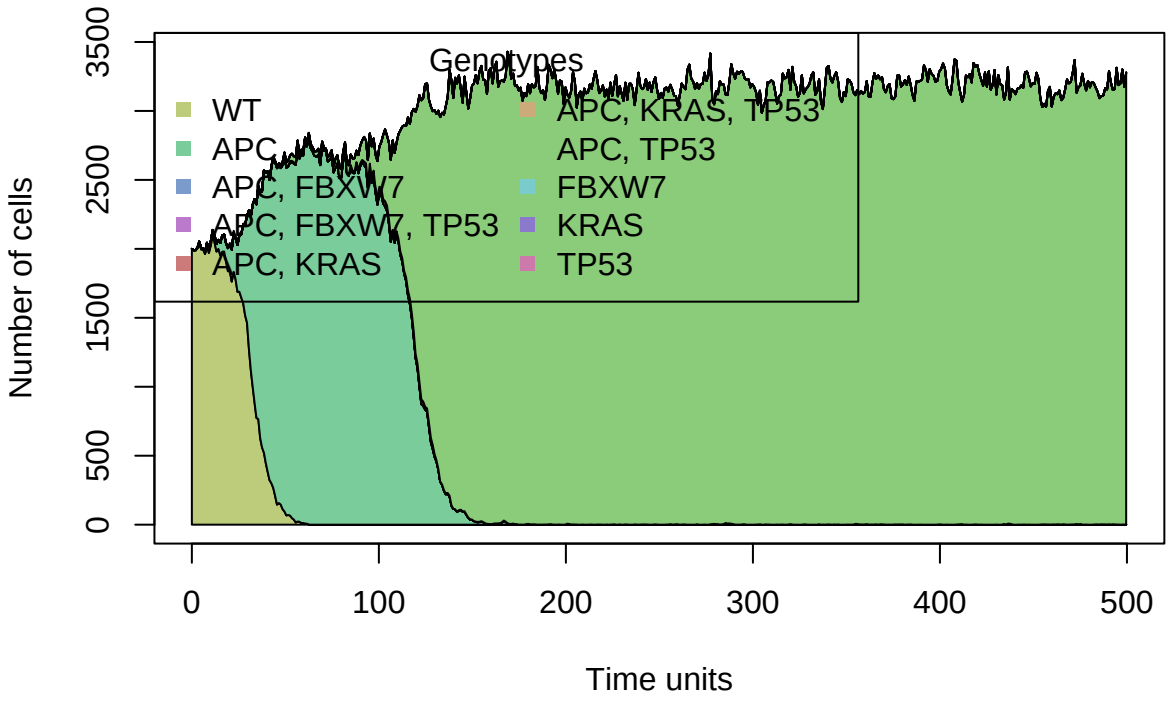

Figure 16: Simulation of cancer progression for the simplified model when onlyCancer = TRUE. Genotypes are shown stacked.

```
## Plot of simulation for genotypes
plot(CRC_W2_S1,
show = "genotypes",
legend.ncols = 1,
xlim = c(0, 300),
type = "line")
```

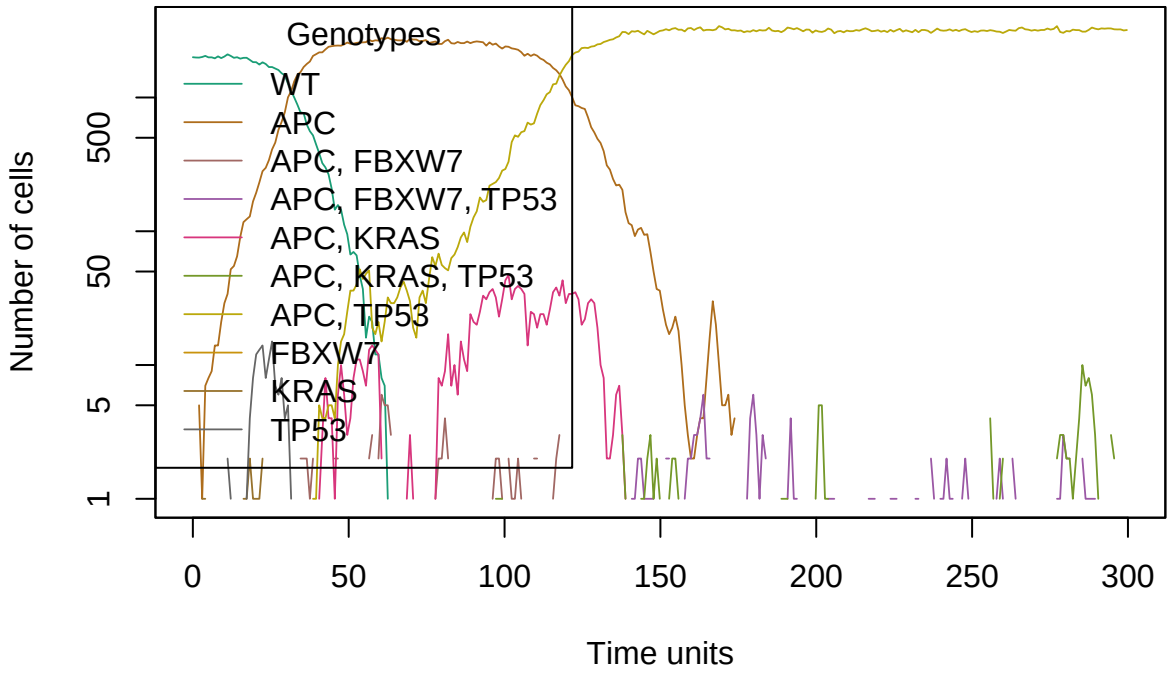

Figure 17: Simulation of cancer progression for the simplified model when onlyCancer = TRUE. Genotypes are shown as lines.

Figure 18 and Figure 19 shows the genealogical relationships of clones that appeared during the simulations. The number of the arrows represent the times that each clone appeared. When simulation are set to reach cancer the clones that have a genotype belonging to the two local optima appear(Figure 19). Whereas if simulation are executed without the cancer parameter, the most represented clone is the one that has the best fitted genotype (see Figure 18).

```
## Plot of genealogical relationships
plotClonePhylog(CRC_W2_S, N = 0, keepEvents = TRUE)
```

```
## Plot of genealogical relationships
plotClonePhylog(CRC_W2_S1, N = 0, keepEvents = TRUE)
```

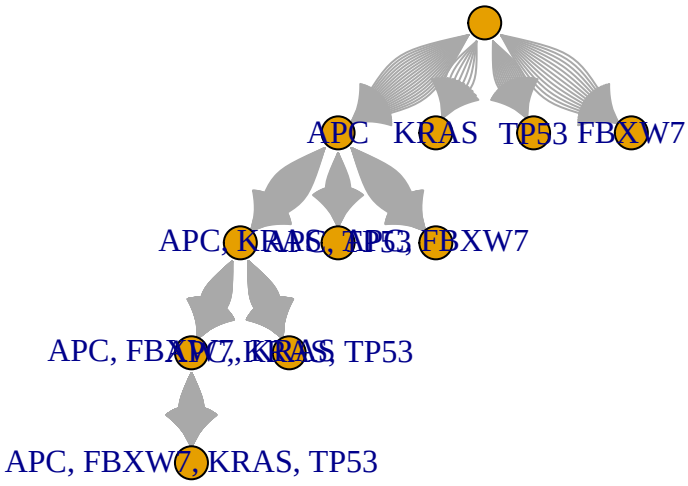

Figure 18: Genealogical relationships of clones.

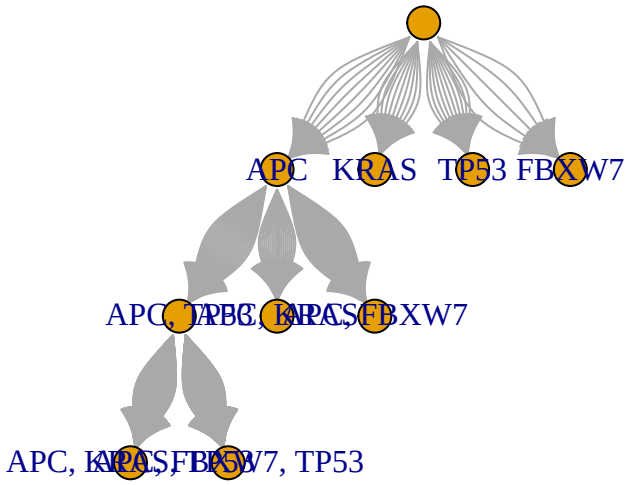

Figure 19: Genealogical relationships of clones when onlyCancer = TRUE.

3.3 Synthetic Lethality

Synthetic lethality is a special type of epistasis. Therefore, we used the epistasis module inside `allFitnessEffects` to define an epistatic interaction between TP53 and AP (see Figure 20) and restriction imposed by the DAG (i.e. `XOR` relationships). The fitness values were assigned such that an scenario where synthetic lethality via pairwise interaction occurs.

The fitness landscape shows that the genotype for which the synthetic lethality was specified has a lower fitness value as expected, although it is not a local minima. Similarly to fitness landscape in Figure 13, the local maxima is composed by the genotyopes that satisfy both epistatic interactions and restrictions imposed. Whereas, local minima is composed by genotypes that contain genes with synthetic lethality and other genes that have top-down dependencies (see Figure 21).

```
## Simplified model
## Define poset restrictions, mapping of genes to modules, and driver genes
CRC_W3 <- allFitnessEffects(data.frame(parent = c(rep("Root", 2), "A", "B", "C"),
                                       child = c("A", "B", rep("C", 2), "D"),
                                       s = c(0.2, 0.1, rep(0.05, 2), 0.01),
                                       sh = -0.5,
                                       typeDep = c(rep("XMPN", 4), "MN")),
                            epistasis = c("-A : B" = 0.1,
                                          "-B : A" = 0.2,
                                          "A:B" = -0.5),
                           geneToModule = c("Root" = "Root",
                                            "A" = "APC",
                                            "B" = "TP53",
                                            "C" = "KRAS",
                                            "D" = "FBXW7"),
                           drvNames = c("APC", "TP53", "KRAS", "FBXW7"))

# DAG representation
plot(CRC_W3, expandModules = TRUE, autofit = TRUE, lwdf = 2)
```

```
## Map genotypes to fitness
CRC_F1 <- evalAllGenotypes(CRC_W3, order = FALSE, addwt = TRUE)

(CRC_F1)
```

```
##                   Genotype  Fitness
## 1                       WT 1.000000
## 2                      APC 1.440000
## 3                    FBXW7 0.500000
## 4                     KRAS 0.500000
## 5                     TP53 1.210000
## 6              APC, FBXW7 0.720000
## 7               APC, KRAS 1.512000
## 8               APC, TP53 0.660000
## 9             FBXW7, KRAS 0.505000
## 10            FBXW7, TP53 0.605000
## 11             KRAS, TP53 1.270500
## 12       APC, FBXW7, KRAS 1.527120
## 13       APC, FBXW7, TP53 0.330000
## 14        APC, KRAS, TP53 0.330000
```

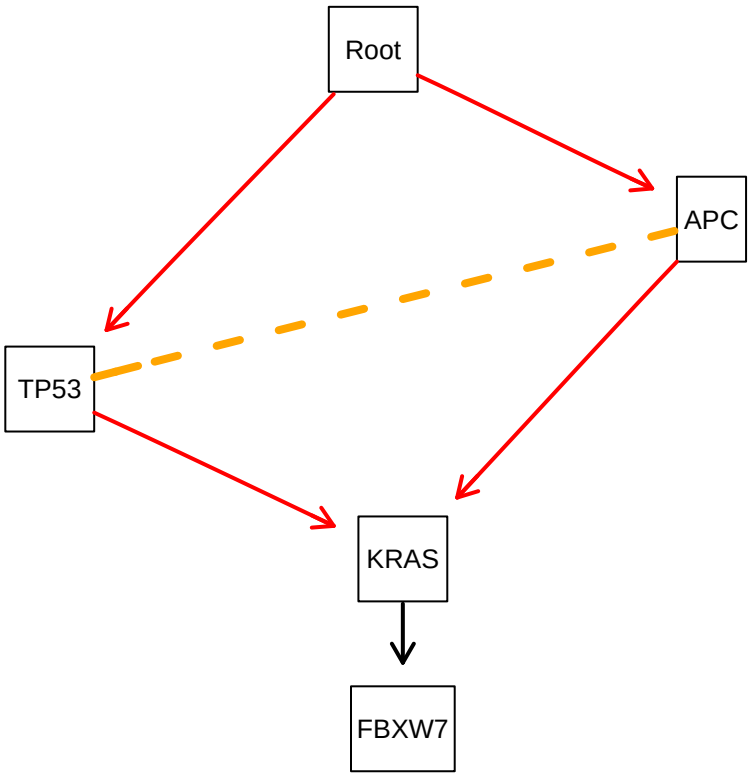

Figure 20: DAG with synthetic lethality.

```
## 15      FBXW7, KRAS, TP53 1.283205
## 16 APC, FBXW7, KRAS, TP53 0.333300
```

```
## Plot of fitness landscape

plot(CRC_F1, use_ggrepel = TRUE)
```

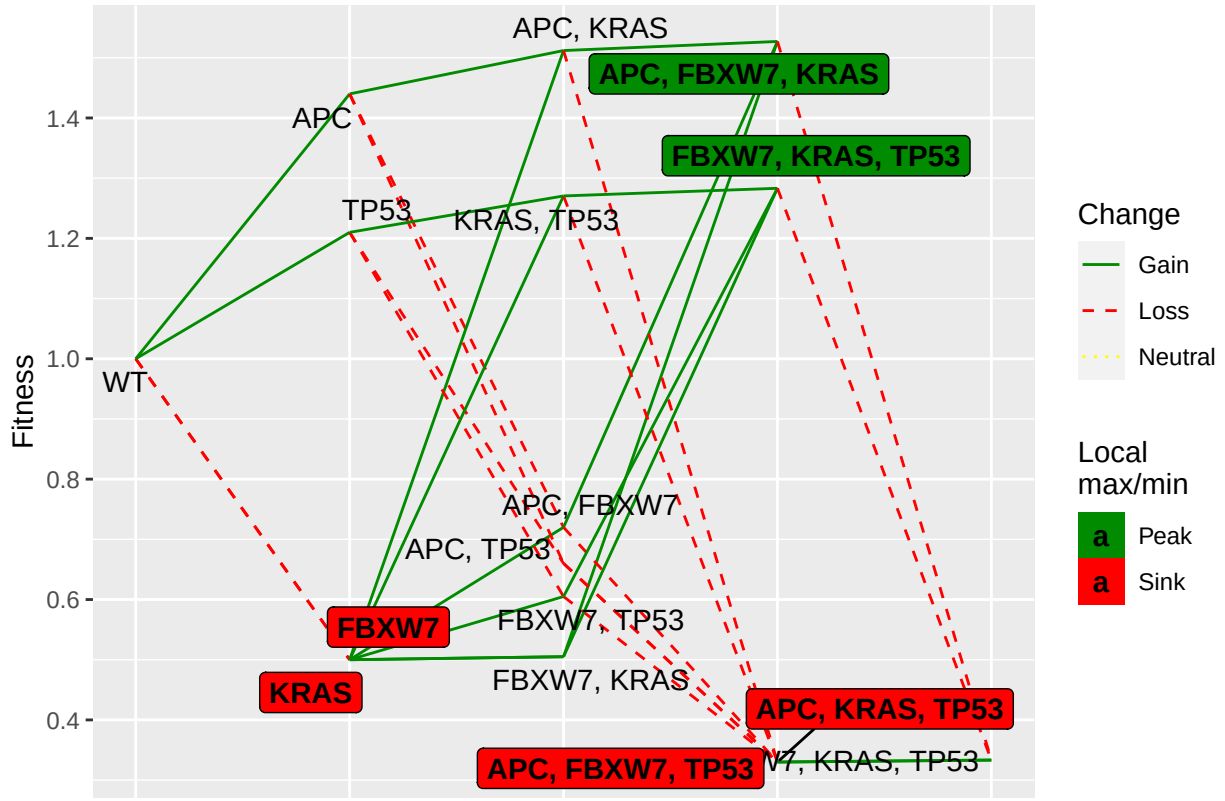

Figure 21: Fitness landscape inferred from simplified DAG with synthetic lethality.

In order to simulate synthetic lethality via three-way interaction, we set fitness values that reflect slightly deleterious effect (if two genes appear) or a highly deleterious effect (if three genes appear). Figure 22 shows the DAG derived for the three-way interaction between APC, TP53, and KRAS. The inferred fitness landscape shows that the global minima is composed by the genotype that carries the synthetic lethality. Whereas, local maxima is composed by genotypes that follow the restrictions imposed in the DAG. Also, note that the global maxima is APC. This is not surprising given that APC is an earlier mutation and has the highest fitness values compared to other genes/genotypes (see Figure 23).

```
## Simplified model
## Define poset restrictions, mapping of genes to modules, and driver genes
CRC_W4 <- allFitnessEffects(data.frame(parent = c(rep("Root", 2), "A", "B", "C"),
                                       child = c("A", "B", rep("C", 2), "D"),
                                       s = c(0.2, 0.1, rep(0.05, 2), 0.01),
                                       sh = -0.5,
                                       typeDep = c(rep("XMPN", 4), "MN")),
                            epistasis = c("A : -B : -C" = 0.2,
```

```
                                        "-A : B : -C" = 0.1,
                                        "-A : -B : C" = 0.05,
                                        "A : B : -C" = 0.01,
                                        "-A : B : C" = 0.02,
                                        "-B : A : C" = 0.02,
                                        "A : B : C" = -0.5),
                          geneToModule = c("Root" = "Root",
                                           "A" = "APC",
                                           "B" = "TP53",
                                           "C" = "KRAS",
                                           "D" = "FBXW7"),
                          drvNames = c("APC", "TP53", "KRAS", "FBXW7"))

# DAG representation
plot(CRC_W4, expandModules = TRUE, autofit = TRUE, lwdf = 2)
```

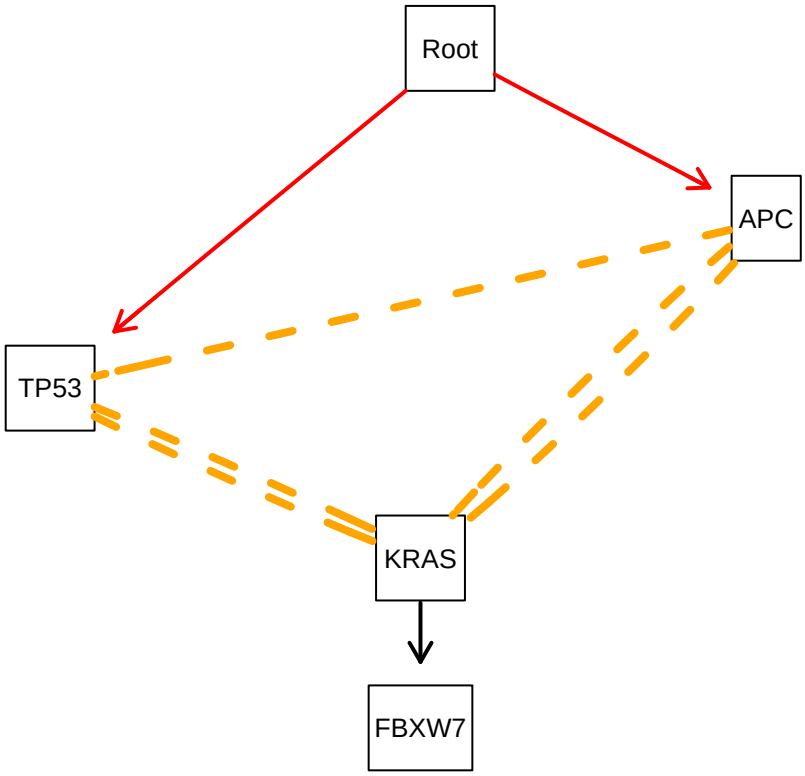

Figure 22: DAG with synthetic lethality (three-way interaction).

```
## Map genotypes to fitness
CRC_F2 <- evalAllGenotypes(CRC_W4, order = FALSE, addwt = TRUE)

(CRC_F2)
```

```
##                  Genotype  Fitness
```

```
## 1                     WT 1.000000
## 2                    APC 1.440000
## 3                 FBXW7 0.500000
## 4                   KRAS 0.525000
## 5                   TP53 1.210000
## 6             APC, FBXW7 0.720000
## 7              APC, KRAS 1.285200
## 8              APC, TP53 1.333200
## 9           FBXW7, KRAS 0.530250
## 10          FBXW7, TP53 0.605000
## 11           KRAS, TP53 1.178100
## 12      APC, FBXW7, KRAS 1.298052
## 13      APC, FBXW7, TP53 0.666600
## 14       APC, KRAS, TP53 0.330000
## 15    FBXW7, KRAS, TP53 1.189881
## 16 APC, FBXW7, KRAS, TP53 0.333300
```

```
## Plot of fitness landscape

plot(CRC_F2, use_ggrepel = TRUE)
```

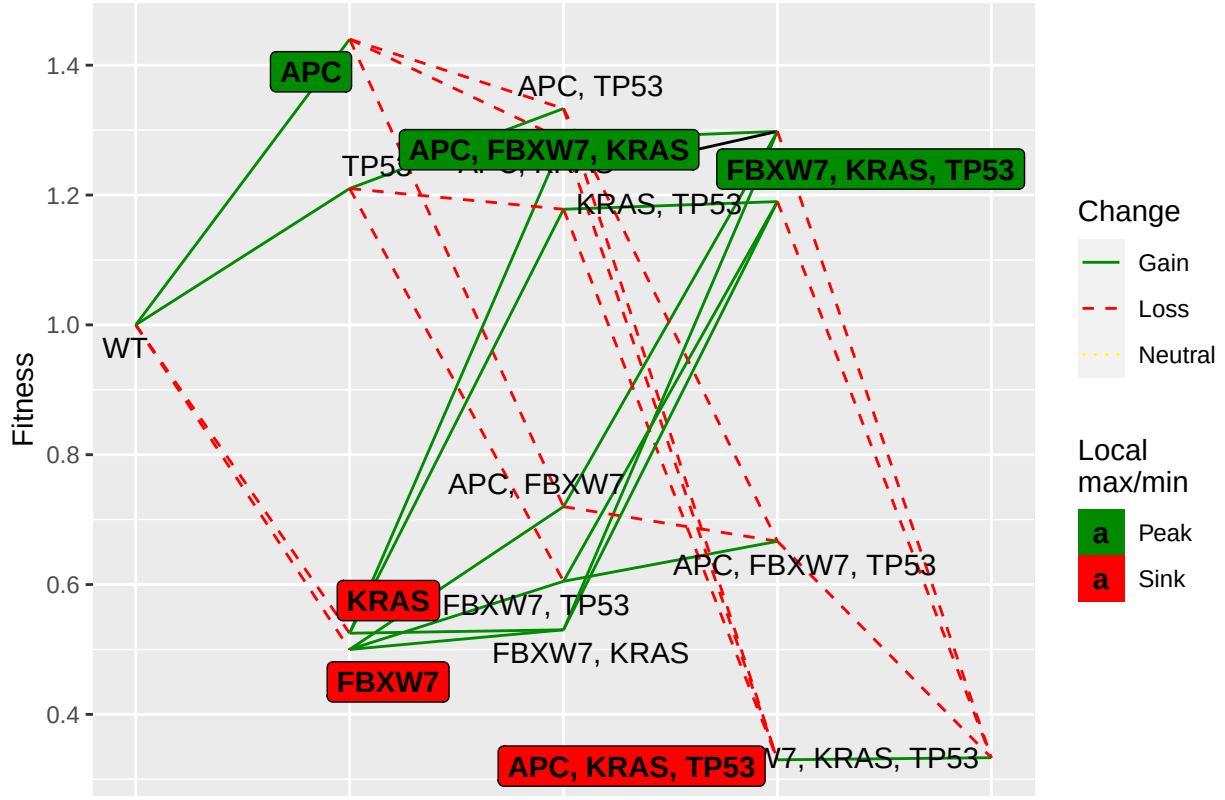

Figure 23: Fitness landscape inferred from simplified DAG with synthetic lethality (three-way interaction).

3.4 Synthetic Viability

Synthetic viability is specified for genotype APC, TP53 (see Figure 24). Here the genotypes composed only by APC or TP53 are deleterious. Figure 25 shows the fitness landscape for synthetic viability via pairwise interaction. Note that the global maxima is composed by the genotype that contains all genes. On the other hand, local minima is composed by genotypes that contains one gene that has a deleterious effect. Note that despite the lower fitness value of genotype FBXW7, KRAS, it conforms a local maxima, although the restrictions imposed in the DAG are not completely satisfied. Moreover, in this fitness landscape, the global maxima may not be reached because the mutational paths required lead to a region composed of multiplpe valleys. It important to mention that order of effects could provide a more realistic fitness landscape. For example, a possible path that leads to the global maxima requires a mutation in KRAS before a mutation in FBXW7.

```
## Simplified model
## SM because synthetic viability requires both parent nodes.
## Define poset restrictions, mapping of genes to modules, and driver genes
CRC_W5 <- allFitnessEffects(data.frame(parent = c(rep("Root", 2), "A", "B", "C"),
                                       child = c("A", "B", rep("C", 2), "D"),
                                       s = c(0.2, 0.1, rep(0.05, 2), 0.01),
                                       sh = -0.5,
                                       typeDep = c(rep("MN", 5))),
                            epistasis = c("-A : B" = -0.2,
                                          "-B : A" = -0.3,
                                          "A:B" = 0.5),
                           geneToModule = c("Root" = "Root",
                                            "A" = "APC",
                                            "B" = "TP53",
                                            "C" = "KRAS",
                                            "D" = "FBXW7"),
                           drvNames = c("APC", "TP53", "KRAS", "FBXW7"))

# DAG representation
plot(CRC_W5, expandModules = TRUE, autofit = TRUE, lwdf = 2)
```



```
## Map genotypes to fitness
CRC_F3 <- evalAllGenotypes(CRC_W5, order = FALSE, addwt = TRUE)
(CRC_F3)
```

```
##                  Genotype Fitness
## 1                      WT 1.00000
## 2                     APC 0.84000
## 3                  FBXW7 0.50000
## 4                    KRAS 0.50000
## 5                    TP53 0.88000
## 6              APC, FBXW7 0.42000
## 7               APC, KRAS 0.42000
## 8               APC, TP53 1.98000
## 9             FBXW7, KRAS 0.50500
## 10            FBXW7, TP53 0.44000
## 11             KRAS, TP53 0.44000
## 12       APC, FBXW7, KRAS 0.42420
## 13       APC, FBXW7, TP53 0.99000
## 14        APC, KRAS, TP53 2.07900
```

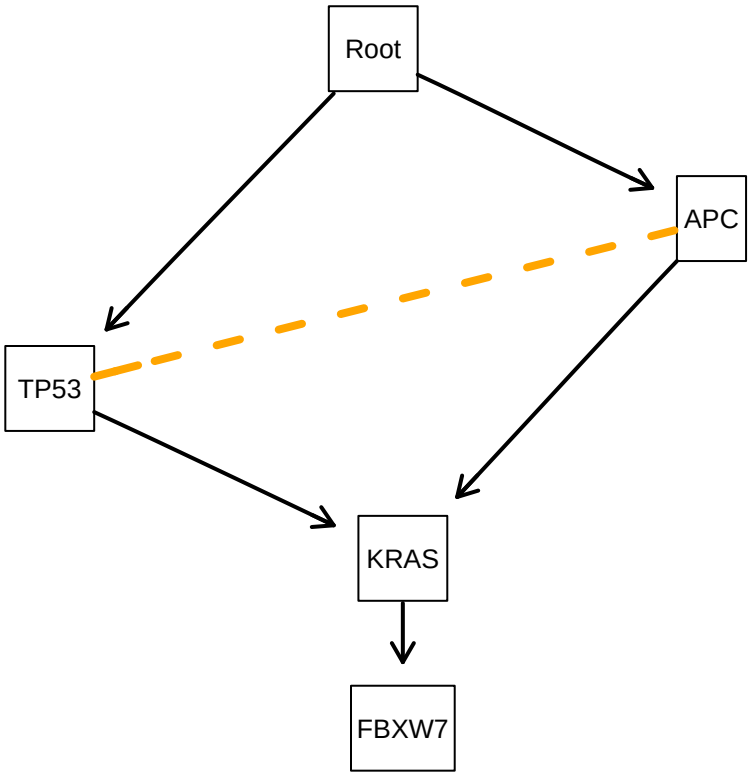

Figure 24: DAG with synthetic viability (pairwise interaction).

```
## 15      FBXW7, KRAS, TP53 0.44440
## 16 APC, FBXW7, KRAS, TP53 2.09979
```

```
## Plot of fitness landscape

plot(CRC_F3, use_ggrepel = TRUE)
```

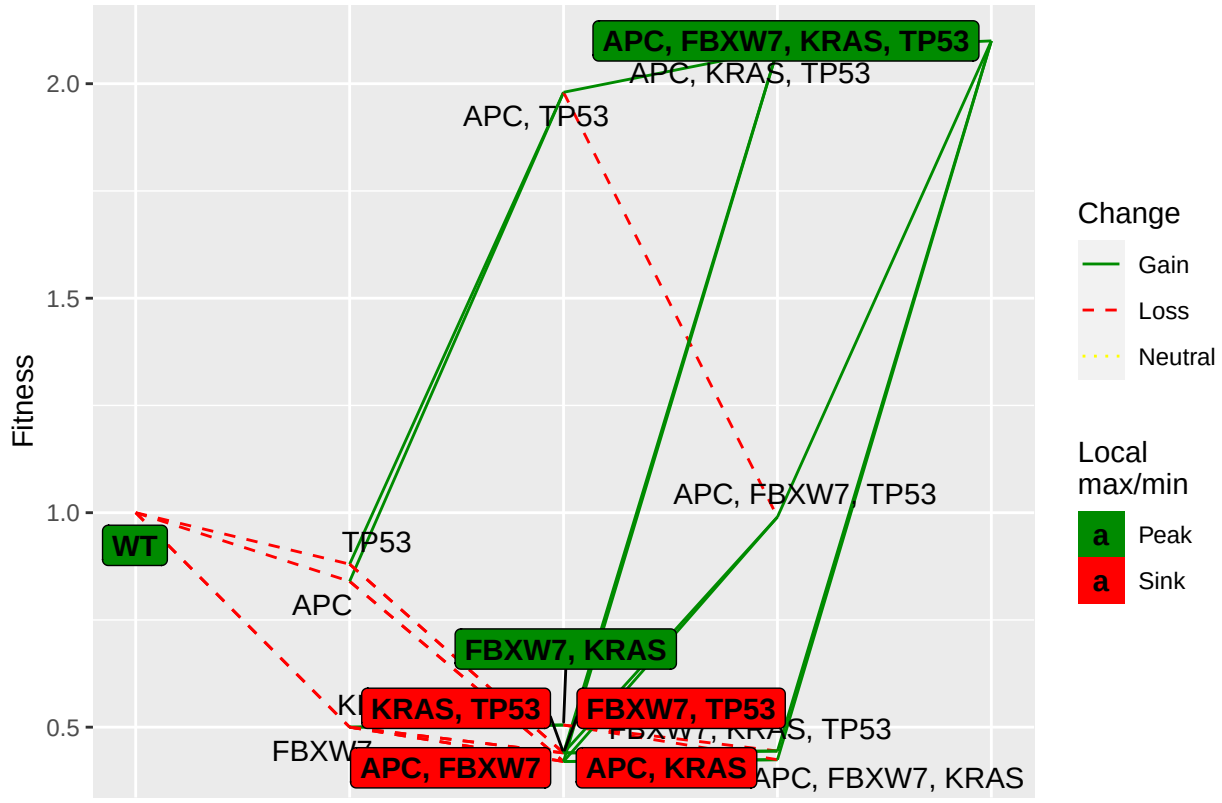

Figure 25: Fitness landscape inferred from simplified DAG with synthetic viability.

Figure 26 shows synthetic viability with a three-way interaction between APC, TP53, and KRAS. For this, we specified highly deleterious effects if APC, TP53, or KRAS appear independently. Whereas, slightly deleterious effects were set if two of those genes appear in a genotype. The fitness landscape for this scenario (see Figure 27) shows the order of restrictions and epistatic interactions set lead to the global maxima composed by the genotype APC, TP53, KRAS, FBXW7. This result support the idea that DAGs are better suited to represent sign epistasis (7). Nevertheless, as mentioned above, including order of effects can give a more realistic fitness values associated to genotypes.

In this work, we have represented synthetic lethality via pairwise and three-way interactions. However, this can be achieved if the DAG is composed by individual genes instead of modules because modules does not allow to define epistatic relationships between genes of the same module. This is important because genes of the same module can participate in the same pathway, as discussed in (2).

```
## Simplified model
## SM because synthetic viability requires both parent nodes.
## Define poset restrictions, mapping of genes to modules, and driver genes
```

```
CRC_W6 <- allFitnessEffects(data.frame(parent = c(rep("Root", 2), "A", "B", "C"),
                                       child = c("A", "B", rep("C", 2), "D"),
                                       s = c(0.2, 0.1, rep(0.05, 2), 0.01),
                                       sh = -0.5,
                                       typeDep = c(rep("MN", 5))),
                            epistasis = c("A : -B : -C" = -0.2,
                                          "-A : B : -C" = -0.2,
                                          "-A : -B : C" = -0.3,
                                          "A : B : -C" = -0.05,
                                          "-A : B : C" = -0.01,
                                          "A : -B : C" = -0.01,
                                          "A : B : C" = 0.5),
                           geneToModule = c("Root" = "Root",
                                            "A" = "APC",
                                            "B" = "TP53",
                                            "C" = "KRAS",
                                            "D" = "FBXW7"),
                           drvNames = c("APC", "TP53", "KRAS", "FBXW7"))

# DAG representation
plot(CRC_W6, expandModules = TRUE, autofit = TRUE, lwdf = 2)
```

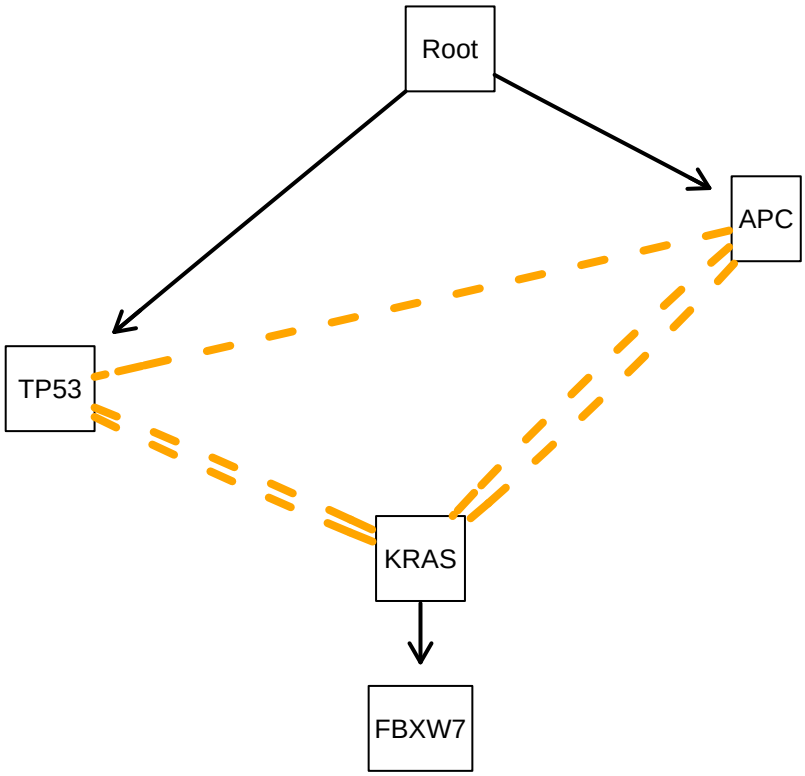

Figure 26: DAG with synthetic viability (three-way interaction).

```
## Map genotypes to fitness
CRC_F4 <- evalAllGenotypes(CRC_W6, order = FALSE, addwt = TRUE)
(CRC_F4)
```

```
##                   Genotype  Fitness
## 1                       WT 1.000000
## 2                      APC 0.960000
## 3                   FBXW7 0.500000
## 4                     KRAS 0.350000
## 5                     TP53 0.880000
## 6               APC, FBXW7 0.480000
## 7                APC, KRAS 0.594000
## 8                APC, TP53 1.254000
## 9              FBXW7, KRAS 0.353500
## 10             FBXW7, TP53 0.440000
## 11              KRAS, TP53 0.544500
## 12       APC, FBXW7, KRAS 0.599940
## 13       APC, FBXW7, TP53 0.627000
## 14        APC, KRAS, TP53 2.079000
## 15      FBXW7, KRAS, TP53 0.549945
## 16 APC, FBXW7, KRAS, TP53 2.099790
```

```
## Plot of fitness landscape

plot(CRC_F4, use_ggrepel = TRUE)
```

4 A probabilistic model of mutually exclusive linearly ordered driver pathways

Mohaghegh Neyshabouri et al. (14) propose a probabilistic model of mutually exclusive linearly ordered driver pathways and analyze two large datasets of colorectal adenocarcinoma (COADREAD) and glioblastoma (GBM) from IntOGen-mutations database. Their model assumes driver genes are over-represented among those mutated across a large tumor collection and, thus, they can be identified in terms of frequency. Also, those participating of the same pathway are mutated in a mutually exclusive manner because more than one mutation in a pathway does not give any selective advantage to the clone.

Like with previous generative models, we map the COADREAD and GMB generative models to actual evolutionary models using different **OncoSimulR** functionalities. This time, we extent what authors model using the mutator and frequency-dependent fitness specifications, to illustrate how differently fitness landscapes evolve even though they are built from exact CPMs when we consider these additional evolutionary phenomena.

4.1 Colorectal adenocarcinoma (COADREAD) dataset

4.1.1 Modelling mutual exclusivity and order restrictions

Figure 7.C from (14) shows the CPM inferred from the COADREAD dataset, consisting of seven modules with between 1 to 4 genes each. The model clearly reconstructs the well-known initiator events in colorectal cancer, including mutations in *APC*, *TP53* and *KRAS* (11). Using the DAG of restrictions as starting

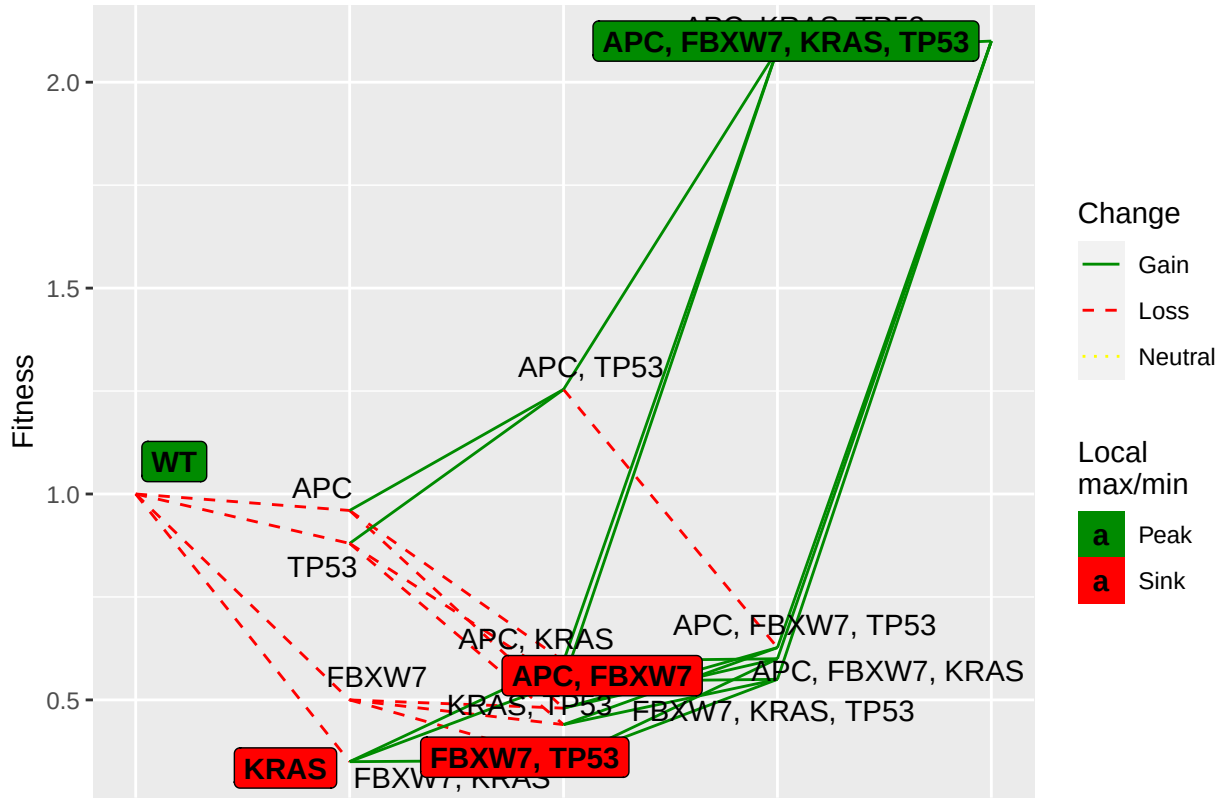

Figure 27: Fitness landscape inferred from simplified DAG with synthetic viability (three-way interaction).

point[1], the evolutionary model is created specifying same genotype fitness for all modules as authors do not state any differences in fitness for when the restrictions in the DAG are satisfied (`s`). However, based on the confidence parameter used by the authors to assess the reliability of modeled restrictions, different fitness are set when the DAG of restrictions is not satisfied (`sh`) (Table 2). Since this method reconstructs linear models (*i.e.* oncogenic trees), there is no need to specify any particular type of dependency between modules (`typeDep`), so we set it to monotonic (MN) as it is a mandatory argument for `allFitnessEffects` function. Figure 28 shows the DAG of restrictions created with `allFitnessEffects`, recapitulating the poset inferred in (14).

Table 2: confidence parameter for each module transition

Module	Confidence parameter (%)
APC	100
TP53	100
KRAS	100
PIK3CA, *NRAS*, *LRP1B*	100
FBXW7, *TCF7L2*, *FAT4*, *ARID1A*	87.7
ATM, *SMAD2*, *ERBB3*, *MTOR*, *CTNNB1*	86.9
SOX9, *SMAD4*	66.7

```
## Restriction table, including DAG of restrictions specifications and associated fitness
COADREAD_rT <- data.frame(parent = c("Root", "A", "B", "C", "D", "E", "F"), # Parent nodes
                          child = c("A", "B", "C", "D", "E", "F", "G"), # Child nodes
                          s = 0.5,
                          sh = c(rep(-1, 4), rep(-.5, 2), -.2),
                          typeDep = "MN")


## Create fitness specifications from DAG of restrictions considering modules
COADREAD_fitness <- allFitnessEffects(COADREAD_rT,
                                      geneToModule = c( "Root" = "Root",
                                                        "A" = "APC",
                                                        "B" = "TP53",
                                                        "C" = "KRAS",
                                                        "D" = "PIK3CA, NRAS, LRP1B",
                                                        "E" = "FBXW7, ARID1A",
                                                        "F" = "ATM, SMAD2, ERBB3, MTOR",
                                                        "G" = "SOX9, SMAD4")) # Modules


## DAG of restrictions representation
plot(COADREAD_fitness, expandModules = TRUE, autofit = TRUE)
```

[1]Due to memory exhaustion, the following genes from the dataset have been removed: FAT4 (module E), CTNNB1 (module F), TCF7L2 (module E)

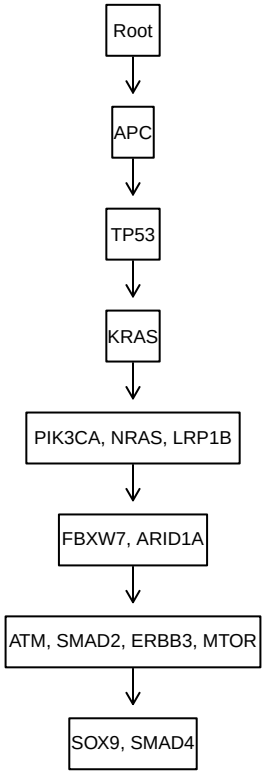

Figure 28: DAG of restrictions for the COADREAD dataset

```r
# Evaluation of all possible genotypes fitness under the previous fitness specifications
COADREAD_FL <- evalAllGenotypes(COADREAD_fitness, max = 131072)

# Fitness landscape representation
plotFitnessLandscape(COADREAD_FL)
```

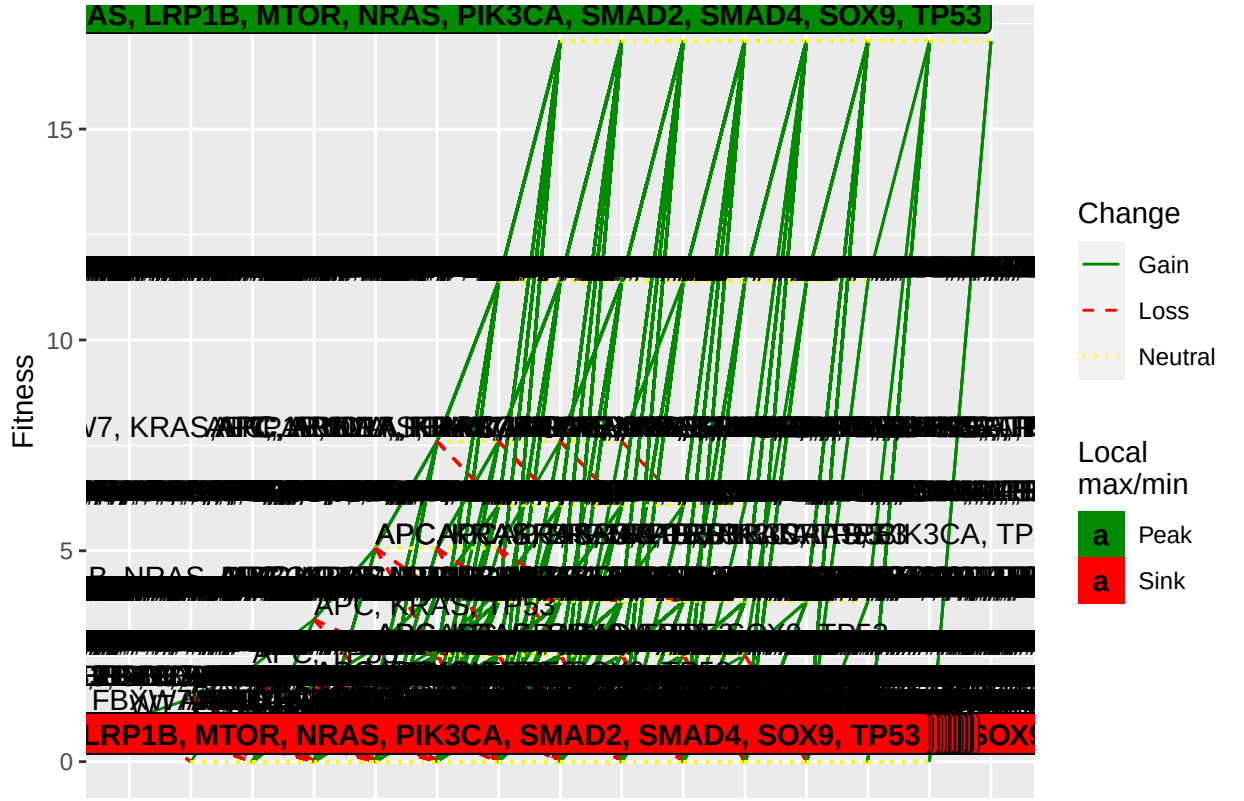

Figure 29: Fitness landscape corresponding with the DAG of restrictions for the COADREAD dataset

Figure 29 plots all possible genotypes in a very busy fitness landscape. Although it is not possible to visualize genotypes clearly, we can see an exponential tendency towards local maxima, corresponding to the clones carrying seven driver-genes genotypes. Stands out how very different genotypes behave the same due to the mutual exclusivity effect, which is probably simplifying a much more complex interaction among genes, specially the differences in fitness effect genes from a module may have.

4.1.2 Simplified cancer progression model

As we did with previous models, for illustrating purposes we designed a simplified version of the CPM by (14) to explore the relationships between modules. Considering each module represents a set of mutually exclusive genes, this simplified version of the model assumes a scenario in which theses genes never mutate at the same time, and thus those genotypes never exist. This way, modules can incorporate a single gene and the relationships between them can be more clearly visualized both in the fitness landscape and in the simulations.

In the simplified version of the CPM, we use the dataframe `COADREAD_rT` without the `geneToModule` specification in the `allFitnessEffects` function. Thus, we visualize "module genotypes" instead of genes.

```
## Create fitness specifications from somplified DAG of restrictions
COADREADsim_fitness <- allFitnessEffects(COADREAD_rT)

## Simplified DAG of restrictions representation
plot(COADREADsim_fitness)
```

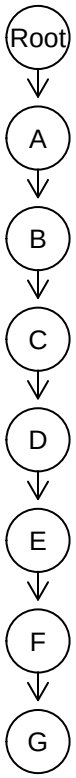

Figure 30: Simplified DAG of restrictions for the COADREAD dataset

```
# Evaluation of all possible genotypes fitness under the previous fitness specifications
COADREADsim_FL <- evalAllGenotypes(COADREADsim_fitness)

# Fitness landscape representation
plotFitnessLandscape(COADREADsim_FL)
```

The fitness landscape in Figure 31 more clearly reflects how fitness increases with the accumulation of mutations in the order specified in the DAG of restrictions exponentially, as we set fitness parameter the same for every parent-child relationship.

Next, we use the simplified fitness landscape to simulate tumor progression for one individual with the `oncoSimulIndiv` functionality.

```
COADREADsim_Simul <- oncoSimulIndiv(COADREADsim_fitness,
                            model = "McFL", ## Model used
                            mu = 1e-4, ## Mutation rate
```

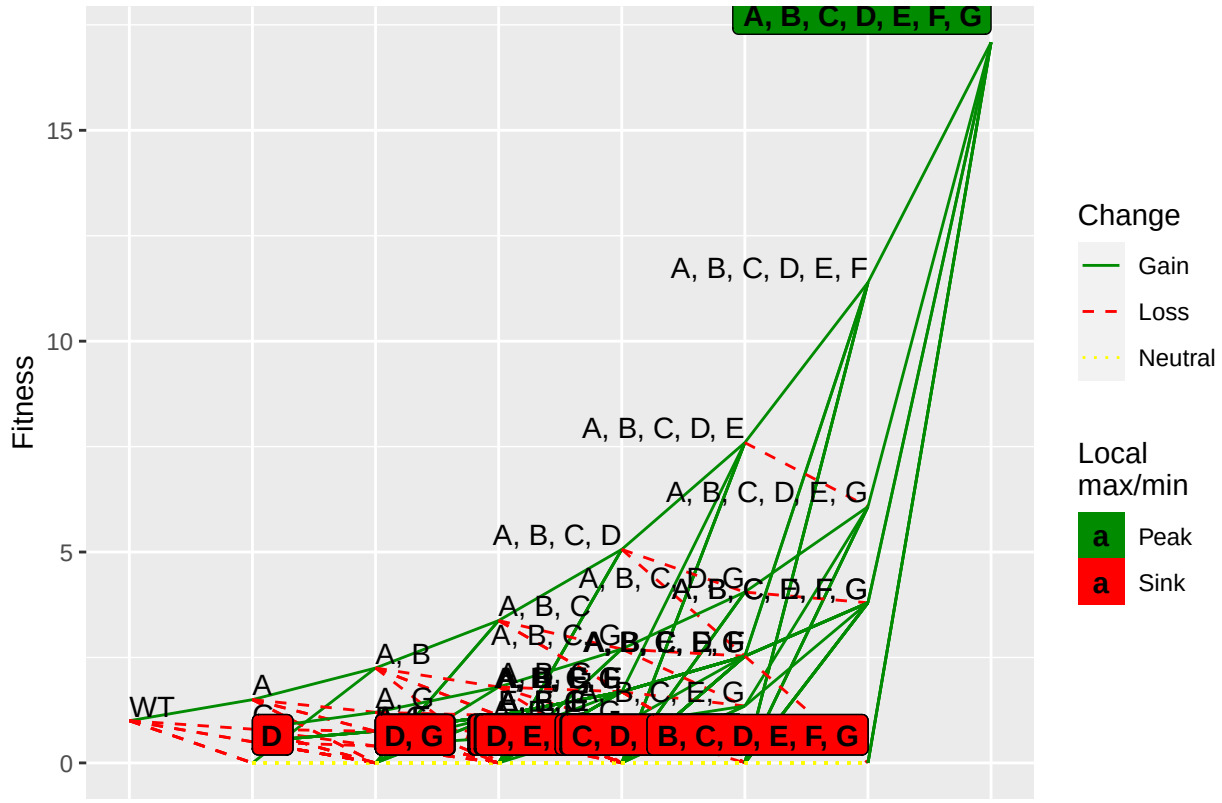

Figure 31: Fitness landscape corresponding with the simplified DAG of restrictions for the COADREAD dataset

```
                          sampleEvery = 0.02, ## How often the whole population is sampled
                          keepEvery = 1,
                          initSize = 2000, ## Initial population size
                          finalTime = 200,
                          keepPhylog = TRUE, ## Allow to see parent-child relationships
                          onlyCancer = FALSE)

## Plot of simulation for genotypes
plot(COADREADsim_Simul,
show = "genotypes",
type = "stacked")
```

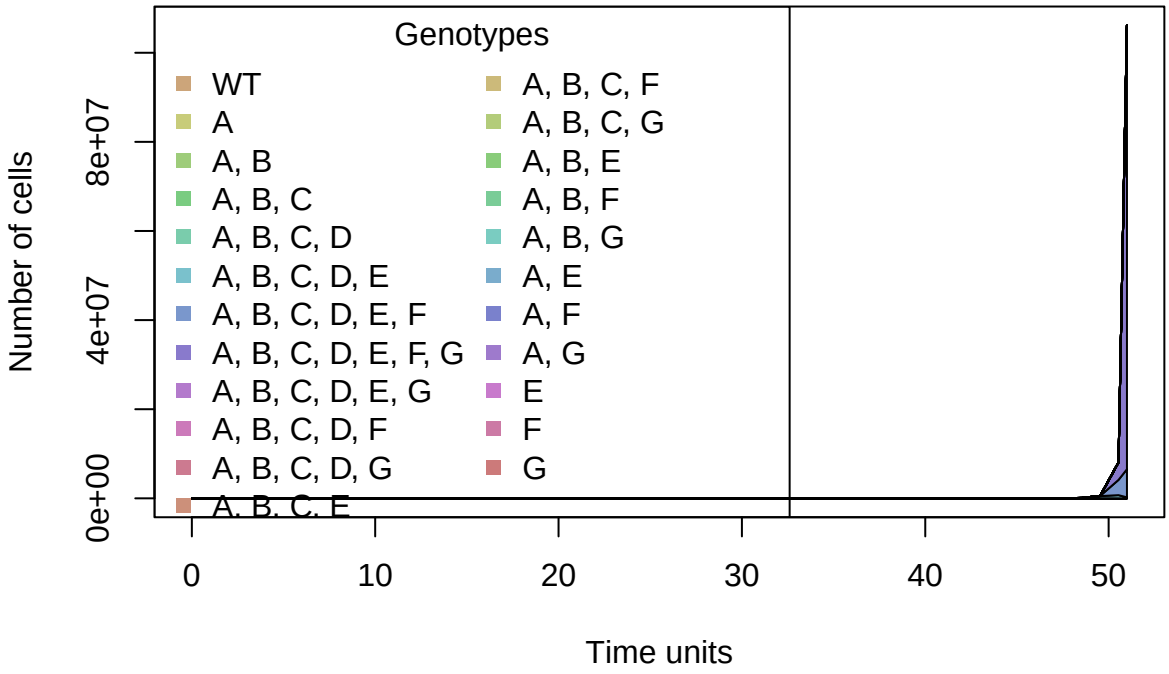

Figure 32: Simulation of cancer progression using the fitness landscape of the simplied model for the COADREAD dataset (stacked plot)

```
plot(COADREADsim_Simul,
     show = "genotypes",
     type = "line"
)
```

```
## Parent-child relationship derived from simulation
plotClonePhylog(COADREADsim_Simul,
                N = 0, ## Specify clones that exist
```

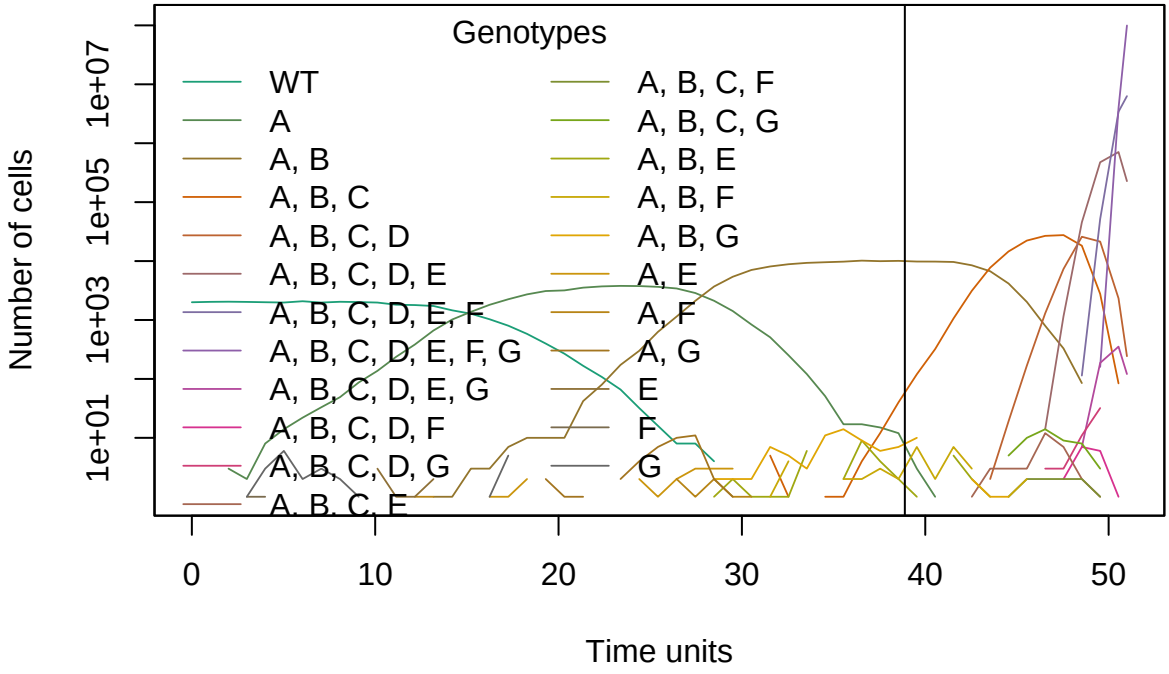

Figure 33: Simulation of cancer progression using the fitness landscape of the simplified model for the COADREAD dataset (line plot)

```
                keepEvents = TRUE ## Arrows showing how many times each clones appeared
)
```

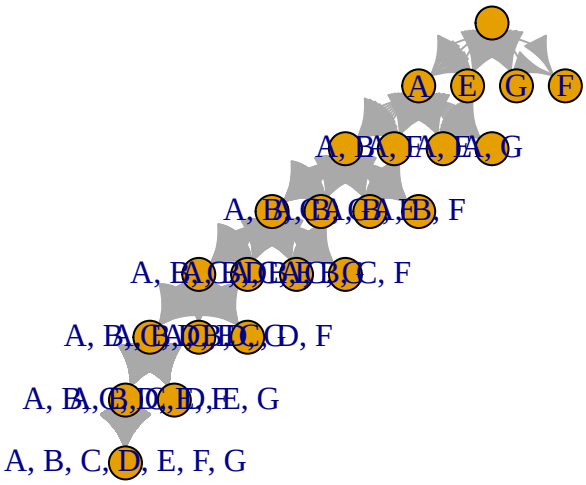

Figure 34: Parent-child relationship derived from simulation

4.1.3 Frequency-dependent fitness

Clones that coexist in a tumor can influence the fitness of each other in a frequency-dependent manner when a mutation produces a phenotype able to modulate the tumor microenvironment. OncoSimulR incorporates the `frecuencyDependentFitness` specification to allow for modelling interaction among clones during tumor progression. To further explore this option, we bring back the complete COADREAD evolutionary model and zoom into its five first nodes to model a scenario in which all the possible node-five genotypes coexist at a time and influence each other. For simplicity, we are not using modules this time.

```
# Mapping of genotypes to frequency-dependent fitness
COADREAD5_gen <- data.frame(Genotype = c("WT", "APC", "TP53", "KRAS",
                                         "APC, TP53", "APC, KRAS", "KRAS, TP53",
                                         "APC, TP53, KRAS",
                                         "APC, TP53, KRAS, PIK3CA",
                                         "APC, TP53, KRAS, NRAS",
                                         "APC, TP53, KRAS, LRP1B"),
                            Fitness = c(rep("1", 5), rep("0", 2), "1",
                                        "2 + 1.5 * f_APC_TP53_KRAS_LRP1B",
                                        "2 + 1.1 * f_APC_TP53_KRAS_LRP1B",
```

```
                                        "2 - f_APC_TP53_KRAS_NRAS"))

## Fitness specifications

COADREAD5_fitness <- allFitnessEffects(genotFitness = COADREAD5_gen,
                                                    frequencyDependentFitness = TRUE,
                                                    frequencyType = "rel")

## Evaluate all genotypes considering population sizes of the clones
COADREAD5_FL <- evalAllGenotypes(COADREAD5_fitness,
                                 spPopSizes = c("WT" = 5, "APC" = 5, "TP53" = 5, "KRAS" = 5,
                                                "APC, TP53" = 5, "APC, KRAS" = 0, "KRAS, TP53" = 0,
                                                "APC, TP53, KRAS" = 100,
                                                "APC, TP53, KRAS, PIK3CA" = 50,
                                                "APC, TP53, KRAS, NRAS" = 50,
                                                "APC, TP53, KRAS, LRP1B" = 50))


# Fitness landscape representation
plotFitnessLandscape(COADREAD5_FL)
```

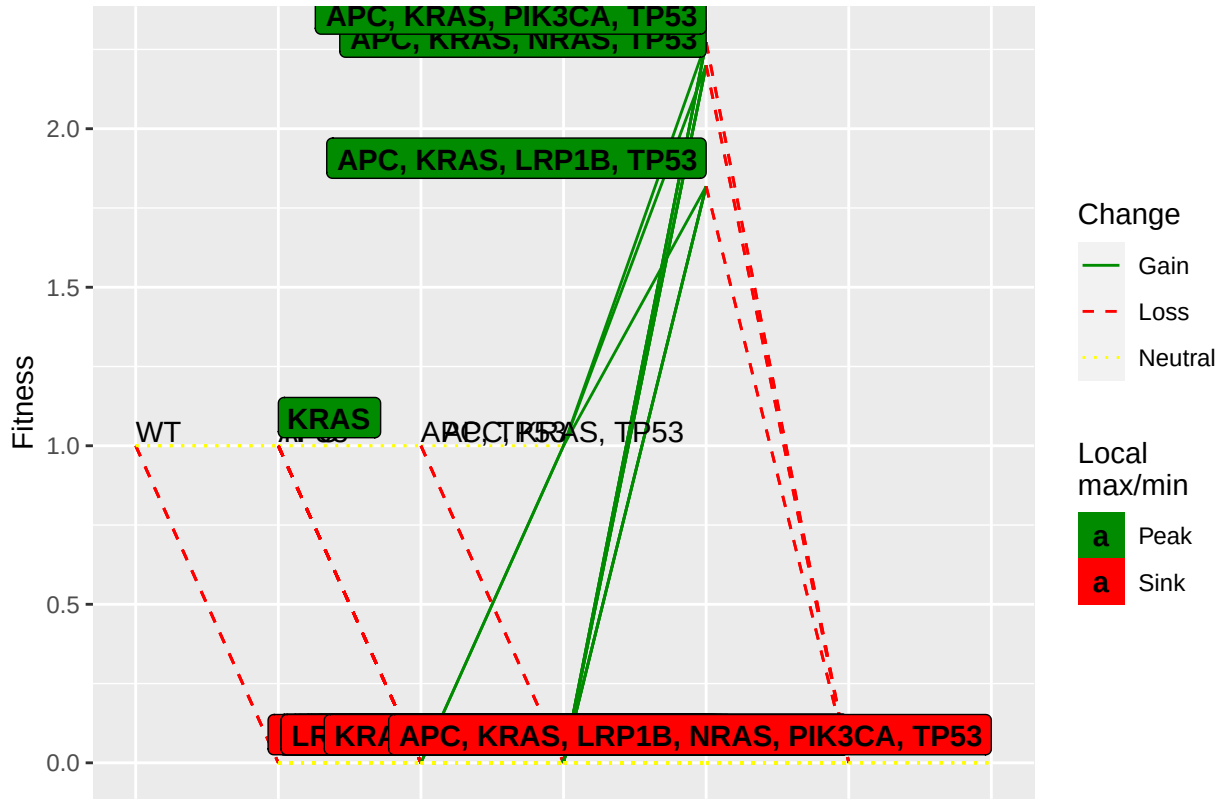

Figure 35: Fitness landscape corresponding with the first-five-nodes possible genotypes for the COADREAD dataset accounting for frequency-dependent fitness

```
COADREAD5sim_Simul <- oncoSimulIndiv(COADREAD5_fitness,
                            model = "McFL", ## Model used
                            mu = 1e-4, ## Mutation rate
                            sampleEvery = 0.02, ## How often the whole population is sampled
                            keepEvery = 1,
                            initSize = 2000, ## Initial population size
                            finalTime = 2000,
                            keepPhylog = TRUE, ## Allow to see parent-child relationships
                            onlyCancer = FALSE)

## Plot of simulation for genotypes
plot(COADREAD5sim_Simul,
show = "genotypes",
type = "stacked")
```

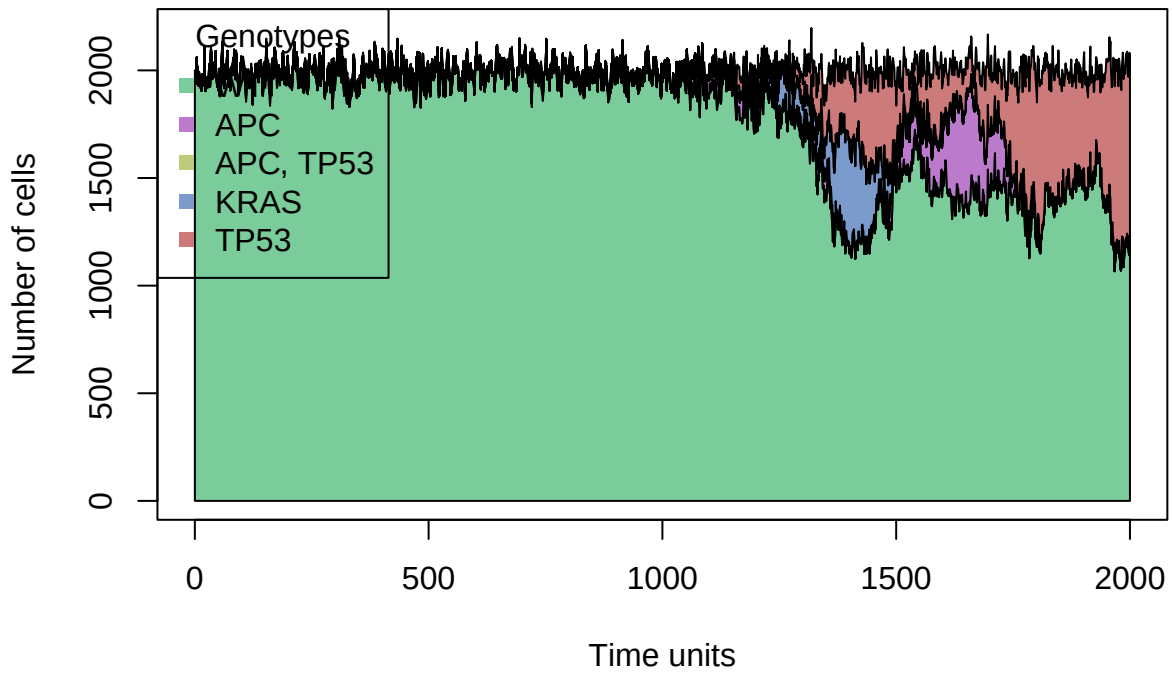

Figure 36: Simulation of cancer progression using the frequency-dependent fitness model for the COADREAD dataset (stacked plot)

```
plot(COADREAD5sim_Simul,
     show = "genotypes",
     type = "line"
)
```

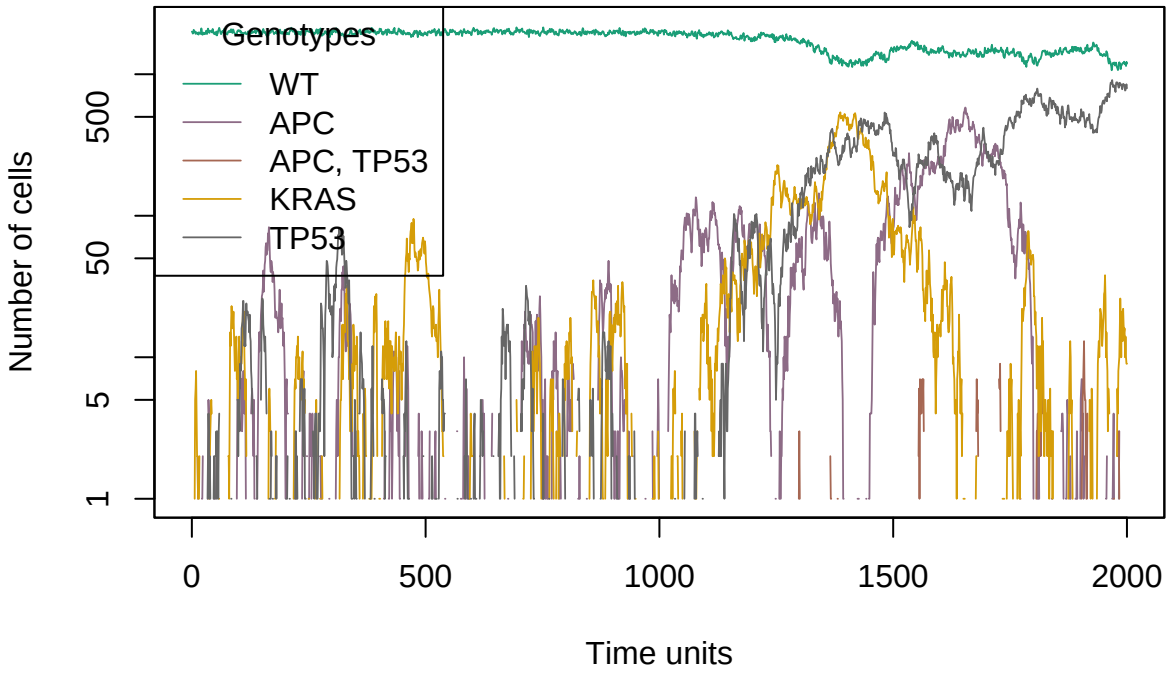

Figure 37: Simulation of cancer progression using the frequency-dependent fitness model for the COADREAD dataset (line plot)

```
## Parent-child relationship derived from simulation
plotClonePhylog(COADREAD5sim_Simul,
                N = 0, ## Specify clones that exist
                keepEvents = TRUE ## Arrows showing how many times each clones appeared
)
```

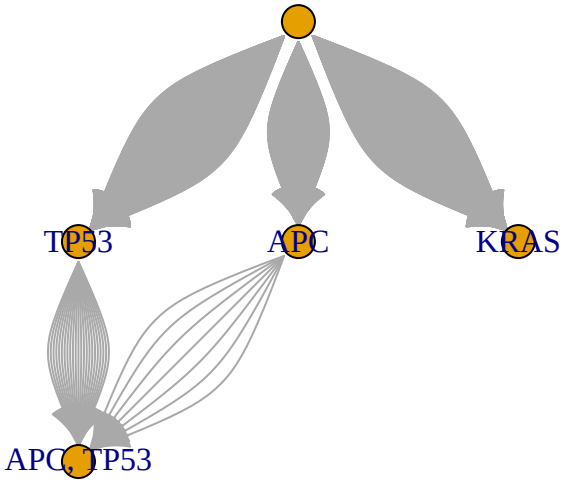

Figure 38: Parent-child relationship derived from simulation

4.2 Glioblastoma (GBM) dataset

4.2.1 Modelling mutual exclusivity and order restrictions

Figure 8.C from (14) shows the CPM inferred from the COADREAD dataset, consisting of seven modules with between 1 to 3 genes each. The knowledge on GMB progression is not as extended as for colorectal cancer and there is not a clearly defined order of first driver mutations, but known driver events such as *EGFR* and *NF1* mutations are correctly inferred in the module **???**. Using the DAG of restrictions as starting point, the evolutionary model is created specifying satisfied and not satisfied genotype fitness for all modules following the same criterion as with the COADREAD dataset. The confidence parameter for each module transition is shown in Table subsubsection 4.2.1. Again, dependency between modules (`typeDep`) is set to monotonic (MN) as it is a mandatory argument for `allFitnessEffects` function.

Table 3: confidence parameter for each module transition

Module	Confidence parameter (%)
NF1, *EGFR*	100
PTEN	100
TP53	100
PIK3CA, *PIK3R1*	97.8
RB1, *IDH1*, *STAG2*	100
ATRX, *LZTR1*	97.3
BCOR, *DCAF12L2*	49.2

```
## Restriction table, including DAG of restrictions specifications and associated fitness
GBM_rT <- data.frame(parent = c("Root", "A", "B", "C", "D", "E", "F"), # Parent nodes
                     child = c("A", "B", "C", "D", "E", "F", "G"), # Child nodes
                     s = 0.5,
                     sh = c(rep(-1, 6), -.2),
                     typeDep = "MN")

## Create fitness specifications from DAG of restrictions considering modules
GBM_fitness <- allFitnessEffects(GBM_rT,
                                 geneToModule = c( "Root" = "Root",
                                                   "A" = "NF1, EGFR",
                                                   "B" = "PTEN",
                                                   "C" = "TP53",
                                                   "D" = "PIK3R1, PIK3CA",
                                                   "E" = "RB1, IDH1, STAG2",
                                                   "F" = "ATRX, LZTR1",
                                                   "G" = "BCOR, DCAF12L2")) # Modules


## DAG of restrictions representation
plot(GBM_fitness, expandModules = TRUE, autofit = TRUE)
```

```
# Evaluation of all possible genotypes fitness under the previous fitness specifications
GBM_FL <- evalAllGenotypes(GBM_fitness, max = 8192)

# Fitness landscape representation
plotFitnessLandscape(GBM_FL)
```

4.2.2 Simplified cancer progression model

As with the COADREAD dataset, we designed a simplified version of the GBM CPM to explore the relationships between modules. We use the dataframe `GBM_rT` without the `geneToModule` specification in the `allFitnessEffects` function.

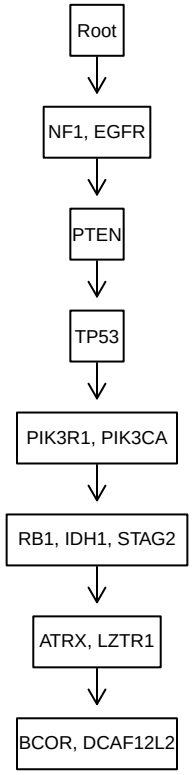

Figure 39: DAG of restrictions for the GBM dataset

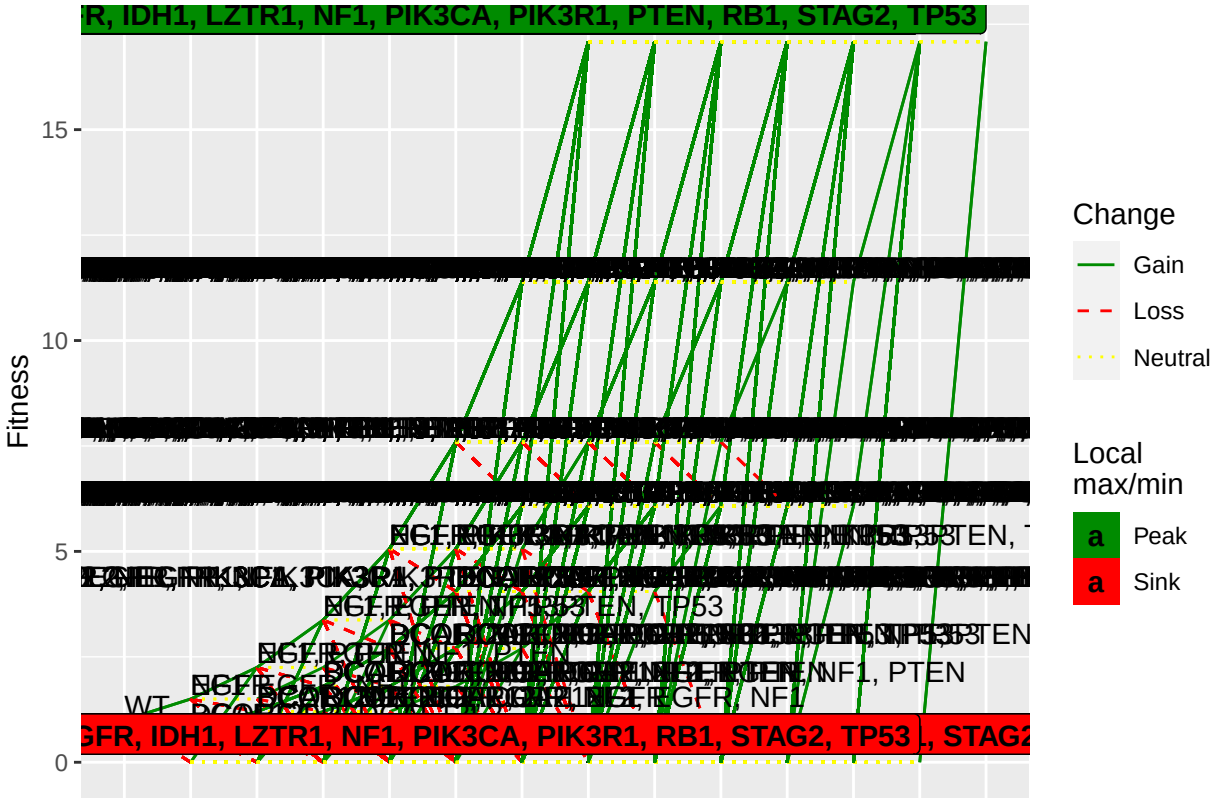

Figure 40: Fitness landscape corresponding with the DAG of restrictions for the GBM dataset

```
## Create fitness specifications from simplified DAG of restrictions
GBMsim_fitness <- allFitnessEffects(GBM_rT)

## Simplified DAG of restrictions representation
plot(GBMsim_fitness)
```

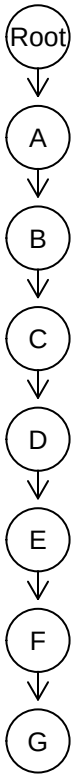

Figure 41: Simplified DAG of restrictions for the GBM dataset

```
# Evaluation of all possible genotypes fitness under the previous fitness specifications
GBMsim_FL <- evalAllGenotypes(GBMsim_fitness)

# Fitness landscape representation
plotFitnessLandscape(GBMsim_FL)
```

The fitness landscape in Figure 32 more clearly shows how fitness increases with the accumulation of mutations in the order specified in the DAG of restrictions exponentially.

Next, we use the simplified fitness landscape to simulate tumor progression for one individual with the `oncoSimulIndiv` functionality.

```
GBMsim_Simul <- oncoSimulIndiv(GBMsim_fitness,
                               model = "McFL", ## Model used
                               mu = 1e-4, ## Mutation rate
                               sampleEvery = 0.02, ## How often the whole population is sampled
```

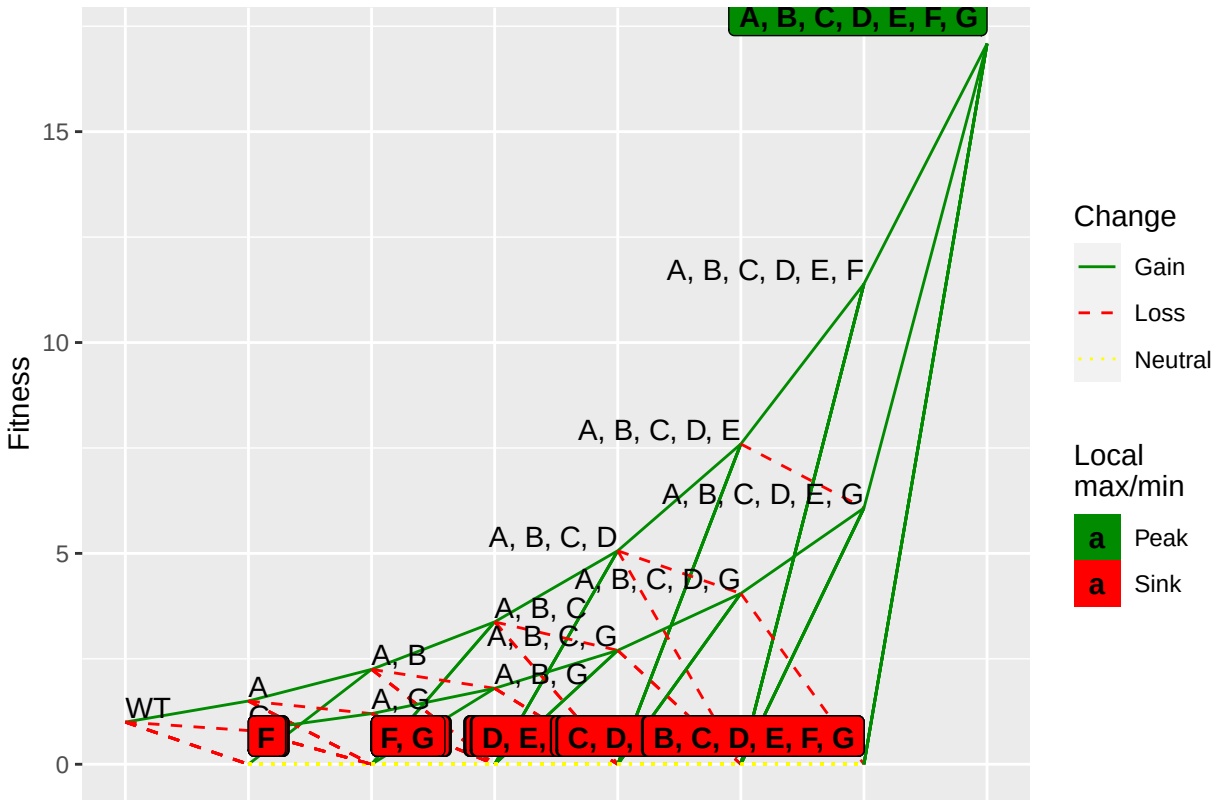

Figure 42: Fitness landscape corresponding with the simplified DAG of restrictions for the GBM dataset

```
                          keepEvery = 1,
                          initSize = 200, ## Initial population size
                          finalTime = 5000,
                          keepPhylog = TRUE, ## Allow to see parent-child relationships
                          onlyCancer = FALSE)

## Plot of simulation for genotypes
plot(GBMsim_Simul,
show = "genotypes",
type = "stacked")
```

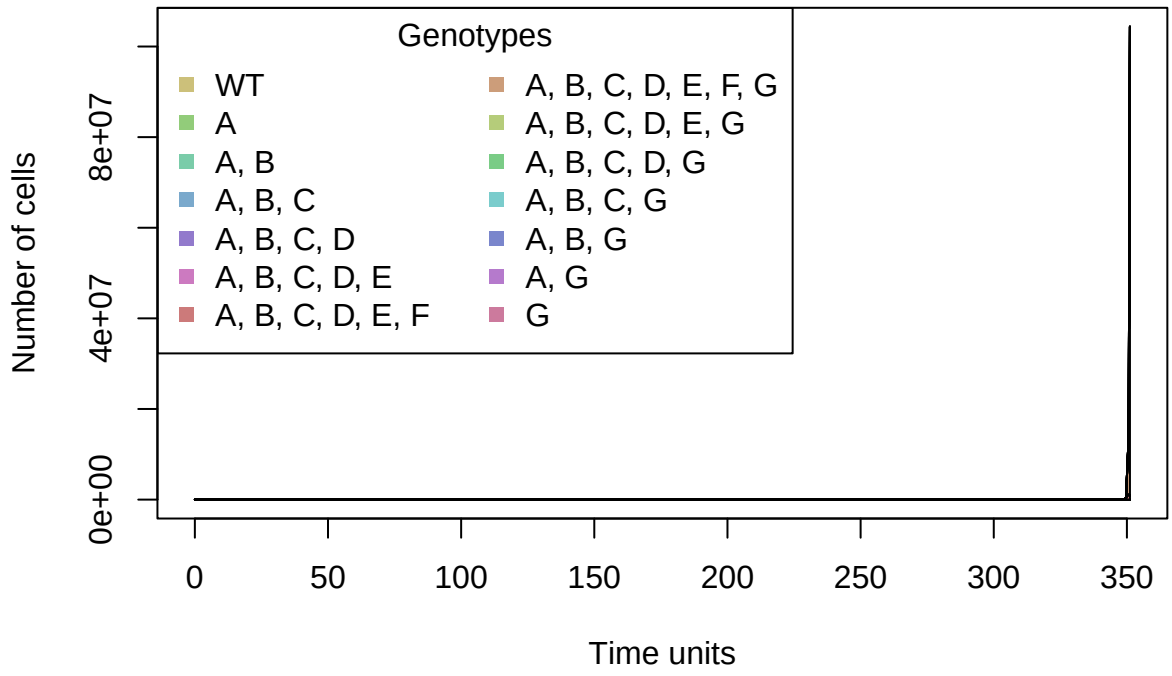

Figure 43: Simulation of cancer progression using the fitness landscape of the simplied model for the GBM dataset (stacked plot)

```
plot(GBMsim_Simul,
     show = "genotypes",
     type = "line"
)
```

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

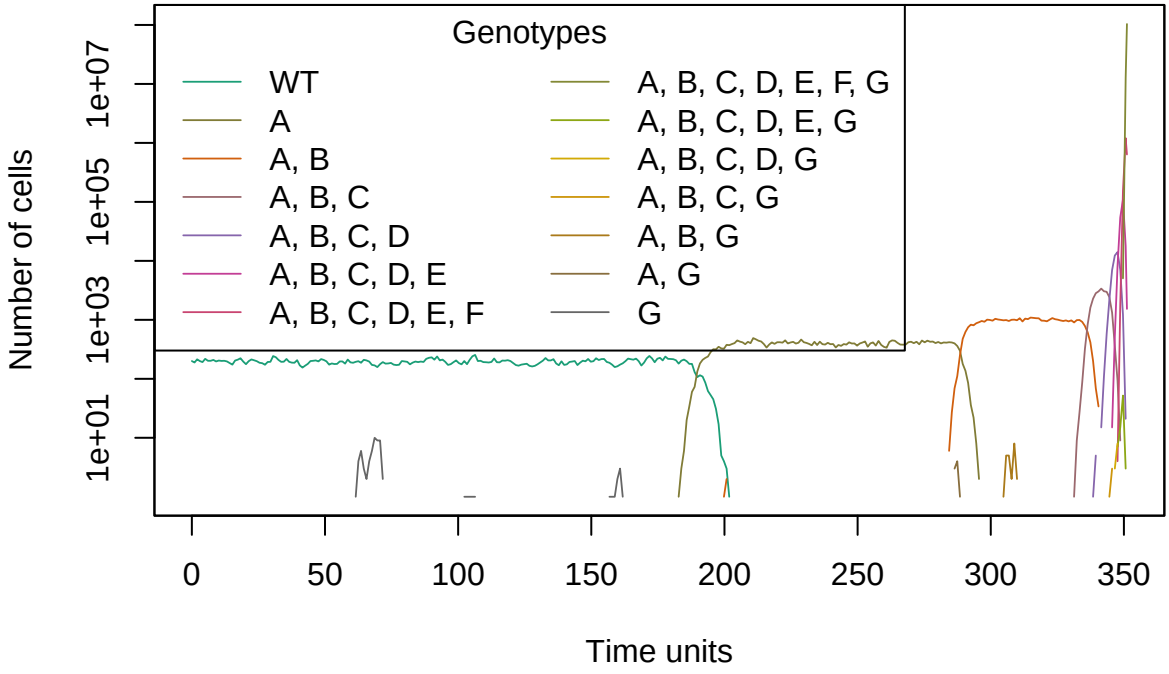

Figure 44: Simulation of cancer progression using the fitness landscape of the simplified model for the GBM dataset (line plot)

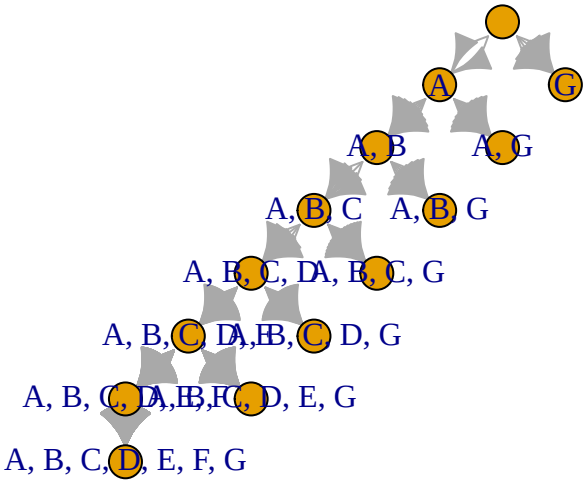

Figure 45: Parent-child relationship derived from simulation

4.2.3 Frequency-dependent fitness

To study frequency-dependent fitness with the GBM dataset, we bring back the first two nodes of the complete GBM evolutionary model and model a scenario in which all the possible genotypes coexist at a time and influence each other. For simplicity, we are not using modules this time.

```
# Mapping of genotypes to frequency-dependent fitness
GBM2_gen <- data.frame(Genotype = c("WT", "NF1", "EGFR", "NF1, PTEN", "EGFR, PTEN"),
                       Fitness = c("0.5", "1",
                                   "1",
                                   "2 + 1.5 * f_EGFR_PTEN",
                                   "2 + f_EGFR"))


## Evaluate all genotypes considering population sizes of the clones
GBM2_fitness <- allFitnessEffects(genotFitness = GBM2_gen,
                                                  frequencyDependentFitness = TRUE,
                                                  frequencyType = "rel")
GBM2_FL <- evalAllGenotypes(GBM2_fitness,
                            spPopSizes = c("WT" = 0,
                                           "NF1" = 0,
                                           "EGFR" = 20,
                                           "NF1, PTEN" = 0,
                                           "EGFR, PTEN" = 20))


# Fitness landscape representation
plotFitnessLandscape(GBM2_FL, use_ggrepel = TRUE)
```

```
set.seed(125)
GBM2sim_Simul <- oncoSimulIndiv(GBM2_fitness,
                        model = "McFL", ## Model used
                        mu = 1e-4, ## Mutation rate
                        sampleEvery = 0.02, ## How often the whole population is sampled
                        keepEvery = 1,
                        initSize = 2000, ## Initial population size
                        finalTime = 200,
                        keepPhylog = TRUE, ## Allow to see parent-child relationships
                        onlyCancer = FALSE)

## Plot of simulation for genotypes
plot(GBM2sim_Simul,
show = "genotypes",
type = "stacked")
```

```
plot(GBM2sim_Simul,
     show = "genotypes",
     type = "line"
)
```

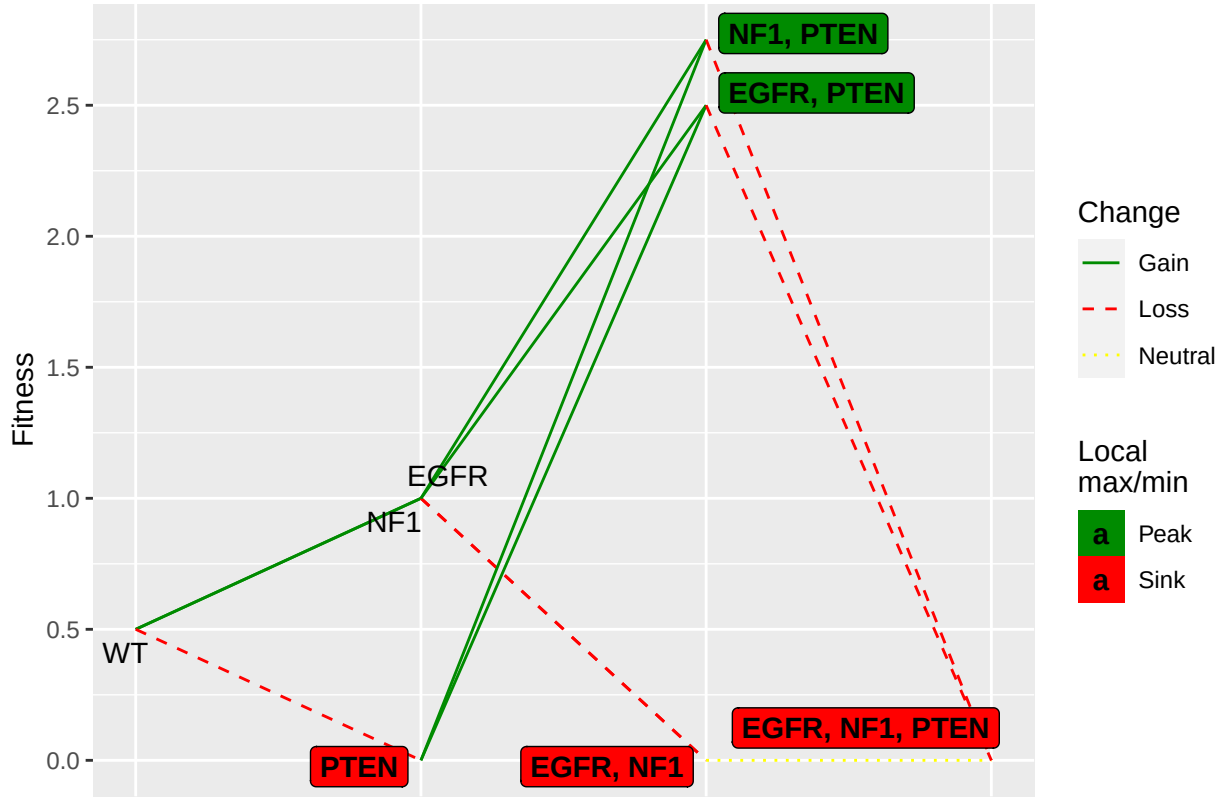

Figure 46: Fitness landscape corresponding with the first-two-nodes possible genotypes for the COADREAD dataset accounting for frequency-dependent fitness

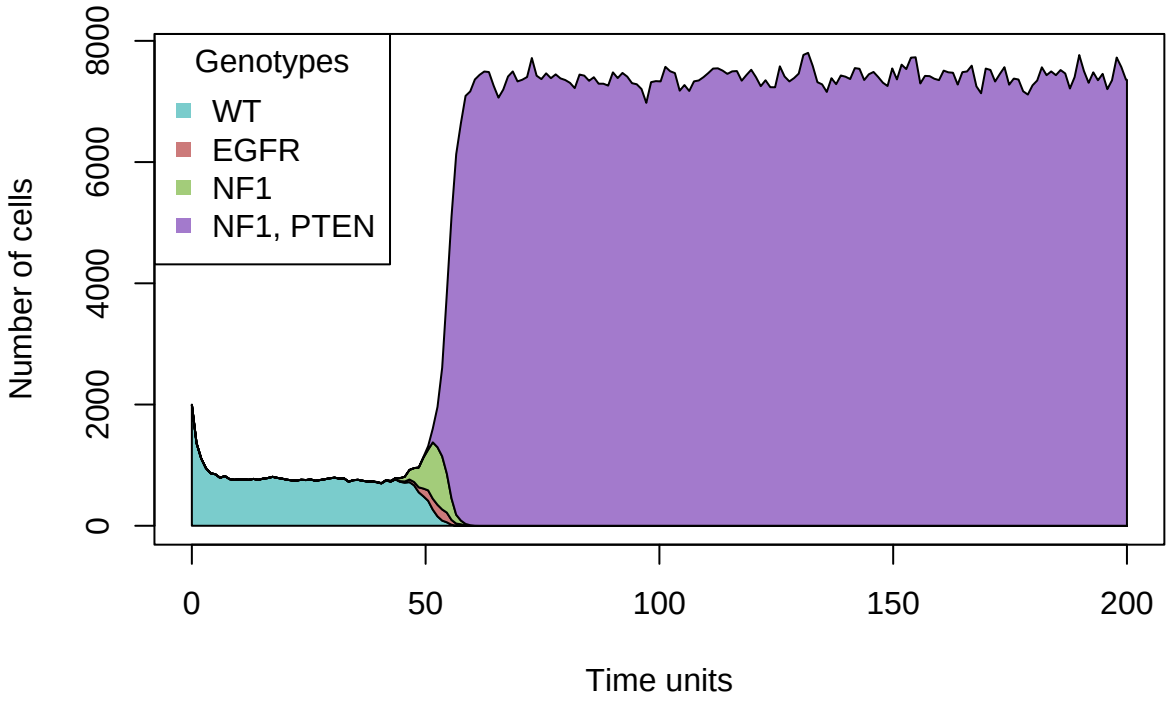

Figure 47: Simulation of cancer progression using the frequency-dependent fitness model for the GBM dataset (stacked plot)

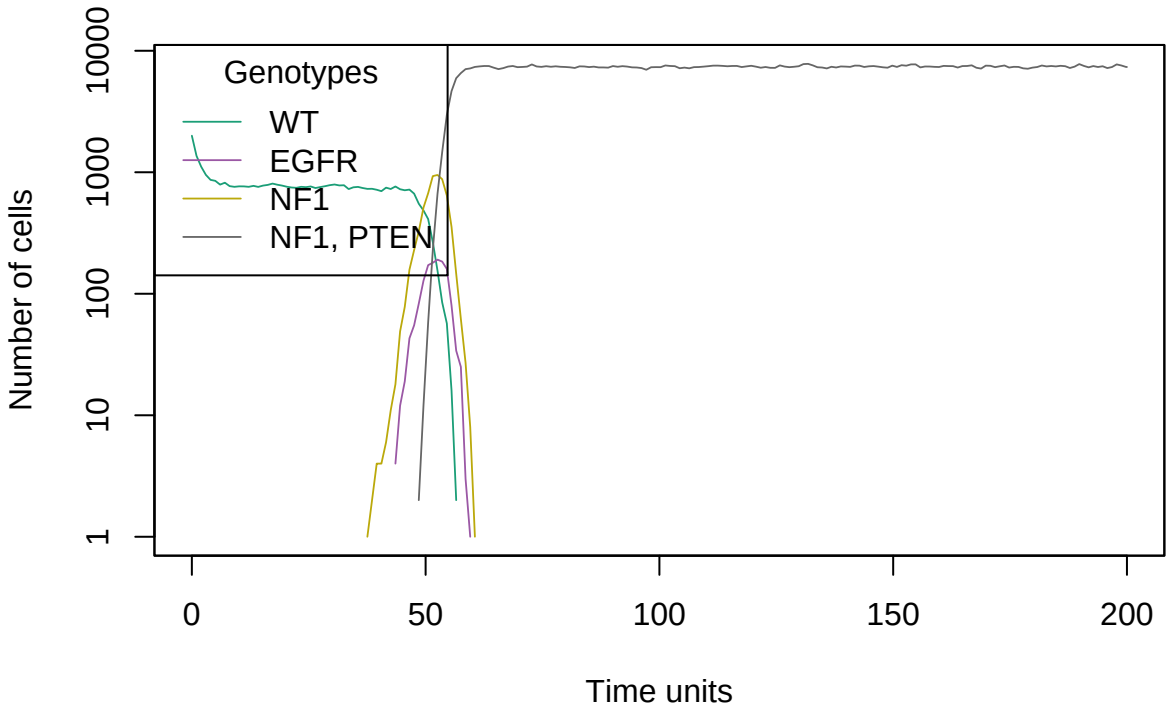

Figure 48: Simulation of cancer progression using the frequency-dependent fitness model for the GBM dataset (line plot)

```
## Parent-child relationship derived from simulation
plotClonePhylog(GBM2sim_Simul,
                N = 0, ## Specify clones that exist
                keepEvents = TRUE ## Arrows showing how many times each clones appeared
)
```

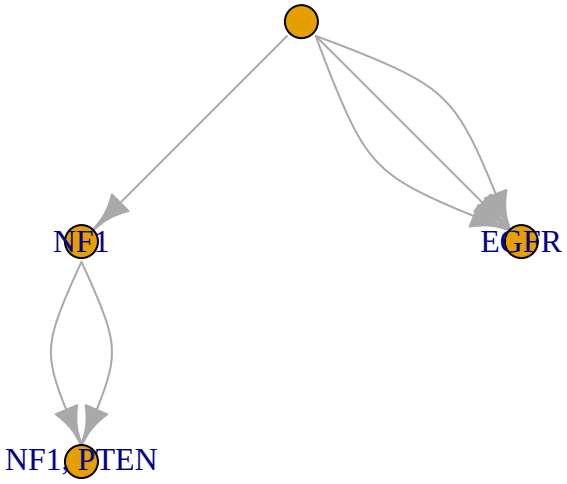

Figure 49: Parent-child relationship derived from simulation

5 References

1. 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

2. Schill R, Solbrig S, Wettig T, Spang R. Modelling cancer progression using Mutual Hazard Networks. Bioinformatics. 2020;36(1):241–9. doi: 10.1093/bioinformatics/btz513

3. Pon JR, Marra MA. Driver and passenger mutations in cancer. Annual Review of Pathology: Mechanisms of Disease. 2015; doi: 10.1146/annurev-pathol-012414-040312

4. Nowell PC. The clonal evolution of tumor cell populations. Science. 1976; doi: 10.1126/science.959840

5. Lee EYHP, Muller WJ. Oncogenes and tumor suppressor genes. 2010. doi: 10.1101/cshperspect.a003236

6. 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

7. 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

8. 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

9. 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

10. 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

11. 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

12. Poelwijk FJ, Kiviet DJ, Weinreich DM, Tans SJ. Empirical fitness landscapes reveal accessible evolutionary paths. Nature. 2007;445(7126):383–6. doi: 10.1038/nature05451

13. Yeang C-H, McCormick F, Levine A. Combinatorial patterns of somatic gene mutations in cancer. The FASEB Journal. 2008; doi: 10.1096/fj.08-108985

14. 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