Dataset of simulations of cell evolution for colorectal cancer

Iurii Nagornov, Minoru Nishino and Mamoru Kato

Requirements for simulations:

R version **3.3** or later

libraries: **stringr, abc**

Table of Contents

1. Common part of models

Please, kindly see the manuscript [Iu.S. Nagornov, M. Kato, Bioinformatics, Vol. 36, N11, June 2020, pp. 3597–3599; https://doi.org/10.1093/bioinformatics/btaa182]. All parties of model are common, excepting a few conditions for hallmarks and the initial clones (see below).

---

2. Models

Models are divided by two criteria:

- compaction factor c - with or without compaction factor. The $c \in [0.1; 1]$ is compaction factor in the dependencies $H_x = c_x \cdot \sum_{k=1}^{N_{genes}} g_k \cdot w_k$, where $g_k = 1$ when function of gene k is destroyed, and $g_k = 0$ for normal state, w_k is a weight for related gene. The index x is related to the hallmarks: $H_x = \{H_a, H_b, H_i, H_d, H_{im}\}$ and $c_x = \{c_a, c_b, c_i, c_d, c_{im}\}$.
- invasion/metastasis transformation condition: $im' = 1$ (strong condition) or $im' > 0$ (weak condition). Please, pay attention that for CF_STRONG model the condition of transformation is changed for $im' = c_{im}$ that relates to transformation only if all genes are destroyed.

So, Table 1 shows 4 models: with and without compaction factors, and with strong or weak condition of invasion/metastasis transformation. For all models the simulations will be same, excluding source command to read **tugHall_clone_functions.R** with functions **trial()** and **update_Hallmarks()**, in which the probabilities of invasion/metastasis transformation and update of all probabilities with hallmarks are calculated. The names of folders are related to names of models (Table 1).

Table 1. Names of models and related folders.
CF is Compaction Factor

	CF = TRUE	CF = FALSE
STRONG condition	CF_STRONG	STRONG
WEAK condition	CF_WEAK	WEAK

3. Range of parameters

3.1 Range of initial number of primary cells

The detection threshold for number of tumor cells is $10^9 \div 10^{12}$ for observation in practice (in human body $\approx 37.2 \cdot 10^{12}$ cells) [Friberg and Mattson, Journal of Surgical Oncology, Vol.65, N4, pp.284–297 (1997)]. That is why the criteria to stop of a simulation should be a moment when the number of cells equals $N_{to\_stop} = 10^9$ or time is more than 200 steps. In practice we use $N_{to\_stop} = 10^6$ to reduce the computational cost and this value is nearby practical one.

Now we have three cases for initial clones:

CASE I: "Mutated_cell" With few exceptions, the tumor cell population(s) in a human, including metastatic one, are originated from only one cell and share not a few mutations (clonal mutations). And the clones usually have the single common ancestor. What is why we set one possibility to start from just one primary cell. If we start from 1 cell in simulation, however, the cell population become extinct in the most cases. In the case of extinction, we have to automatically "restart" the simulation (by default we set 100 as the number of restarting). The restarting function is implemented as additional part for this case. To accelerate the simulation the initial primary cell should have a driver mutation at one gene.

CASE II: "Thousand_cells" Another case is to start from 1000 primary cells in order to increase probability of mutation in one simulation and decrease computational cost. However in this case there are possibly several tumors originated from different normal cells and the tumors do not share any mutations.

CASE III: "Mutated_cell_in_Thousand_cells" And third case is a combination between two previous cases. We start with 1000 primary cells, where one cell has a driver mutation.

Disadvantage of approaches **II and III** is that the clones can affect to each other. Advantage is that in one simulation many clones has a chance to evolve into cancer cell. Also in the simulation we can, in principle, see many clones, but a few will reach number of cells around 10^6. Other clones will have much less number of cells and in practice are not detectable.

In principle one driver mutation in one cell can play role of trigger for cancer evolution. We believe that ABC allows to check this idea. That's why we have to start same simulations (with same input parameters and weights, but different initial mutated genes) for all genes in the **I and III** cases. To compare these different approaches we have to keep same number of simulations, therefore the **case II** should be utilized same times as others cases.

3.2 Balance condition to keep the primary cell's number at constant

The probability for environmental death (k') and initial division coefficient (d_0) should be in balance to keep the number of primary cells at some constant. So we can write balance condition:

$N_{cell,\tau+1} = N_{cell,\tau} \times (1 - k') \times (1 + d_0 - E_0 \times N_{cell,\tau}) = N_{cell,\tau} = N_{balance}$

After reduction in the equation we have balance condition:

$1 = (1 - k') \times (1 + d_0 - E_0 \times N_{balance}),$

where $N_{balance}$ is the number of primary cells, which keeps a balance and E_0 is friction coefficient. Therefore,

we have two additional conditions:

$E_0 \times N_{balance} < d_0 \leqslant 1$ and at least $E_0 < \frac{1}{N_{balance}}$

and probability k' can be derived as:

$k' = 1 - [1 + d_0 - E_0 \times N_{balance}]^{-1}$

In simulations we fixed the $N_{balance} = 10^3$ for cases II and III for certainty ($N_{balance} = 1$ for case I).

Also we have fixed values of parameters, which are not related to genes - hallmarks relations:

- $E_0 = 10^{-4}$ for cases II and III, and $E_0 = 10^{-1}$ for cases I;
- $F_0 = 10$;
- $d_0 = 0.5$ (in real time one time step equals around 105 days, 200 time steps corresponded to 60 years)
- $k' = 0.2857143$;
- parameter in the sigmoid function $s = 10$;
- $u_o = u_s = 0.5$.

3.3 Fraction point for weights

We take fraction points for weights:

the weights are defined as coefficients between hallmarks and genes, that's why we apply normalization for weights:

$$w_i = w_i / \sum_{i=1}^{N} w_i$$

This normalization allows us to define last weight as a rest:

$$w_N = 1 - \sum_{i=1}^{N-1} w_i$$

Case 1: **Choosing weight's values as a discrete step function**, it's possible to keep all weights in the range of defined set of values like $number\_set = \{0\ \ 0.1\ \ 0.2\ \ 0.3\ \ 0.4\ \ 0.5\ \ 0.6\ \ 0.7\ \ 0.8\ \ 0.9\ \ 1.0\}$. We determined each weight as $w_1 = r_1$, $w_2 = (r_2 - r_1)$, ..., and $w_{n-1} = r_{n-1} - r_{n-2}$, where all r_i are ascending ordered uniform random numbers in the set $number\_set$. To keep equal probabilities for all genes, we added the "shuffle" procedure in the end, in the other words, the weights are randomly chosen for genes.

Case 2: **Choosing weight's values as a continuous values** in the range $[0; 1]$. We determined each weight as $w_1 = r_1$, $w_2 = (r_2 - r_1)$, ..., and $w_{n-1} = r_{n-1} - r_{n-2}$, where all r_i are ascending ordered uniform random numbers in the range $[0; 1]$.

For example, we have to randomly define 3 weights:

1-st is $r_1 = 0.6$, 2-nd is $r_2 = 0.9$, then $w_1 = r_1 = 0.3$, $w_2 = r_2 - r_1 = 0.3$ and $w_3 = 1 - r_2 = 0.1$.

3.4 Range of mutation rate

Range of mutation rate is defined by computational cost of calculations. Fig.1 shows a dependence of the computational cost on a mutation rate for different initial number of cells, using just 1 processor.

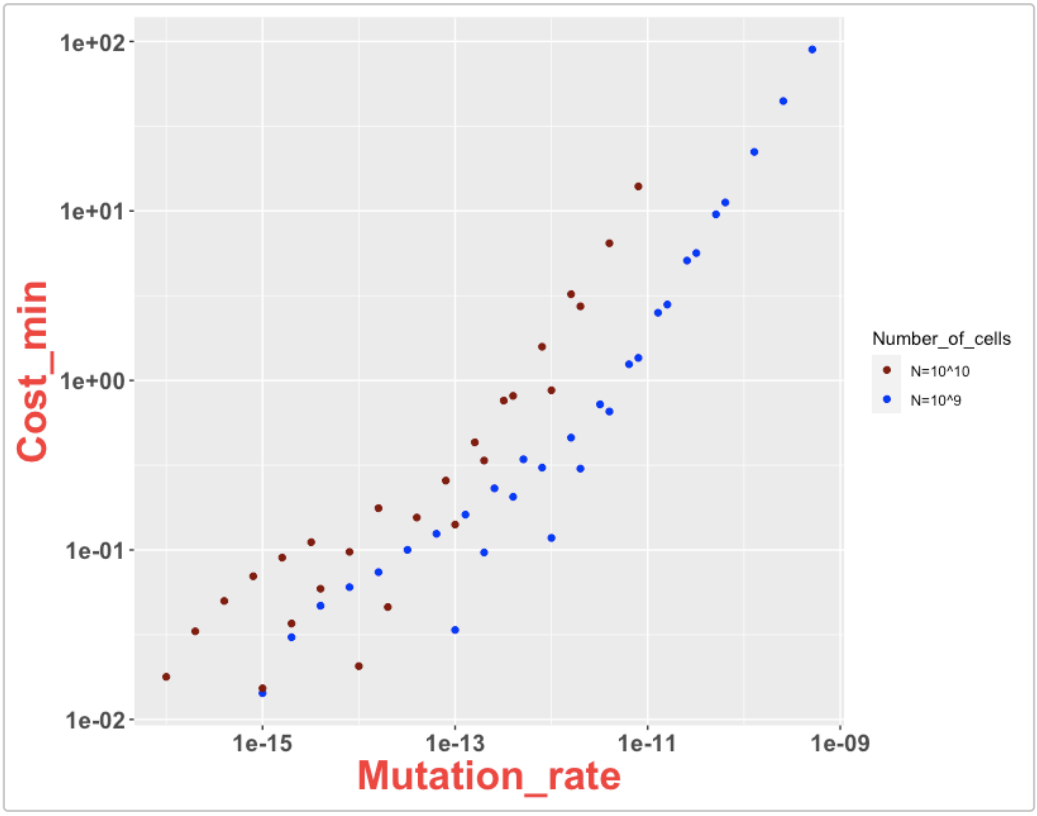


Fig.1. Computational cost for one processor in minutes: with 10^9 and 10^{10} initial cells.

There is two quasi linear dependencies in log scale, one is for number of initial cells equal $N = 10^9$ and another for $N = 10^{10}$. One can make a criteria to stop simulation as a time not more than 1 minute. Then the mutation rate will be not more $m_0 = 10^{-12}$ for $N = 10^{10}$ and $m_0 = 10^{-11}$ for $N = 10^9$. Approximation gives that for one cell the mutation rate limit is $m_0 = 10^{-2}$ and for 1000 cells is $m_0 = 10^{-5}$.

Another problem is when the metastasis cells appear, number of cells is increased, number of new clones also is increased dramatically with exponential growth. In order to avoid this problem in calculation we stopped the simulation at $N_{cells} > 10^6$. In the repeated simulations we fixed mutation rate at $3 \cdot 10^{-9}$. Related to paper [M.Gerstung, C.Jolly, and PCAWG Consortium, Nature, Vol. 578, pp. 122–128 (2020)] mutation rate is in the range $4.8 \cdot 10^{-10} - 3 \cdot 10^{-9}$ in the case of trisomy. So we use the highest value of estimation range.

---

4. Observation data

For observation data we take:

- mutation density or tumor mutation burden (in mutations per kb or per Mb: $1kb = 10^3 bp$ and $1Mb = 10^6 bp$, bp is base pair). The genes range in size from 1.148 to 37.7 kb for human (average length of gene = 8446 bp). About 20000 genes in human cell, and ≈ 30 mutations per Mb in 1 cell in average for cancer disease. So in the model $m_0 = 30/10^6/Max\_of\_time = 30/10^6/100 = 3 \cdot 10^{-7}$ in the metastatic cell.
- variant allele frequencies (VAFs) detected by next-generation sequencing (NGS) for tumor-related genes. To use all information (VAFs from n observed mutations), we use the first highest VAF pair, the second highest VAF pair, the third highest VAF pair etc., and finally the tenth VAF pair for each gene. For example, if there are 4 genes in simulation, we have 40 descending values VAFs.

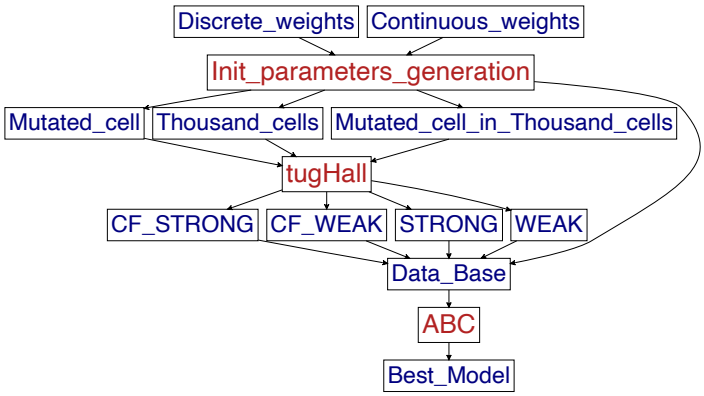
- The highest observed and simulated VAF pair per gene should be used with sorting in Approximate Bayesian Computation (**ABC**).

---

5. Procedure of simulation

5.1 Flowchart of simulations

Flow chart shows the procedures for simulations, using 4 models and 3 initial condition:



5.2 Performance of simulations

The supercomputing resource was provided by Human Genome Center, the University of Tokyo [https://supcom.hgc.jp/shirokane.html]. In R code we use *parallel* library, which allows to make parallel simulation in one node/computer with many cores. But it has trouble to simulate in parallel using several nodes. That's why we use many jobs for each node with parallel simulations at each node (using 24 cores as default).

The code of bash script (file *qsub.sh*) to submit multijobs for supercomputer is below. Here is example for 40 jobs with 1000 simulations for each job and for all types of models:

```
for ((i=1; i < 41; i++))
do

    ### make SIM_XXX.R file for R simultion
    sed "s/node <- 1/node <- $i/g" SIM.R > SIM_$i.R

    ### make the submission file script_$i.sh for HPC system
    sed "s/SIM.R/SIM_$i.R/g"  script.sh   >  script_$i.sh

    ### submit the job for specific $i
    qsub    -v LANG=en_US.UTF-8   -l s_vmem=1G,mem_req=1G   -pe def_slot 24    script_$i.sh

done
```

Files *script_XX.sh* have just one line:

```
/usr/local/package/r/3.6.0/bin/R   CMD   BATCH        /home/nagornov/TESTS/SIM_XX.R
```

where *XX* is number of job. The files *SIM_XX.R* are same as *SIM.R* with just one difference:

```
node <- 1
N_sim <- 1000
id_simulations <- ( N_sim*(node - 1) + 1):( N_sim*(node - 1) + N_sim )
```

the number of node allows to change id's of simulations. For example for $node = 1$ $id\_simulations = 1..1000$ and for $node = 2$ $id\_simulations = 1001..2000$ etc.

5.3 Computational cost

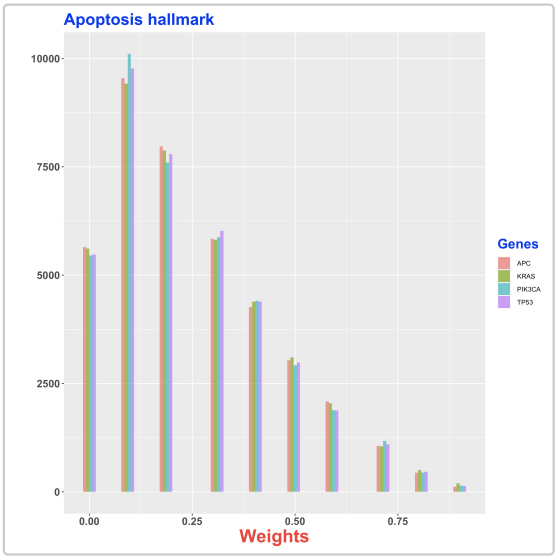
Computational cost was around an hour per job (node with 24 cores) with 1000 simulations for all 24 types of simulations thus in total it took an hour at $24 \cdot 40 = 960$ processors. To be honest, the simulation time was ranged from 3660 sec to 4050 sec that is between 61 and 68 minuets.

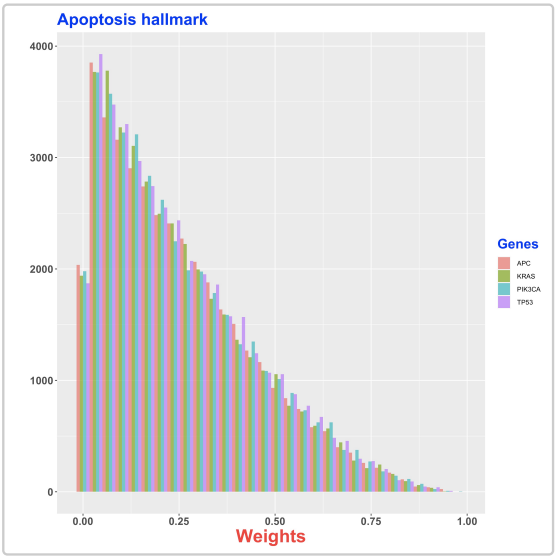
It is easy to estimate the computational cost for 1 million simulations (24 millions for all types of simulations) because it is 25 times more than 40 thousand simulations. Therefore it will take $24000\ cores \cdot hours$ or $1000\ nodes \cdot hours$.

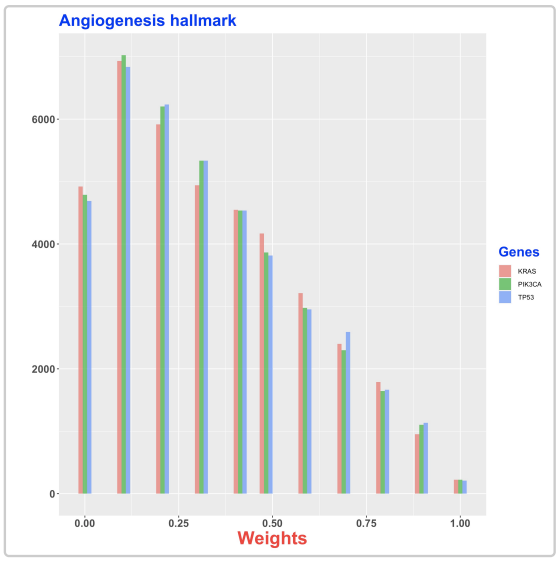
---

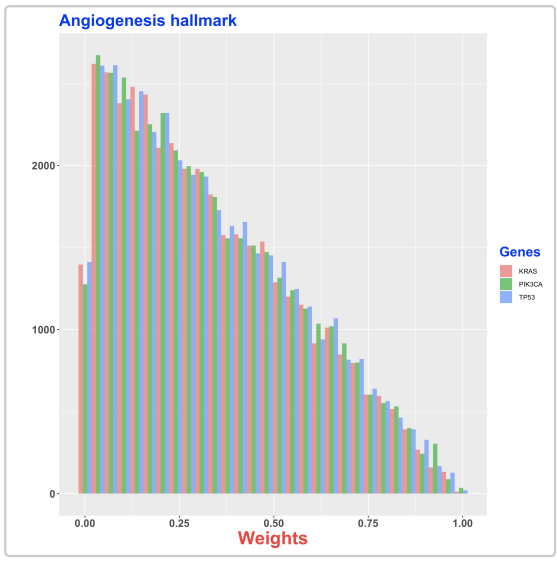
6. Distribution of priors

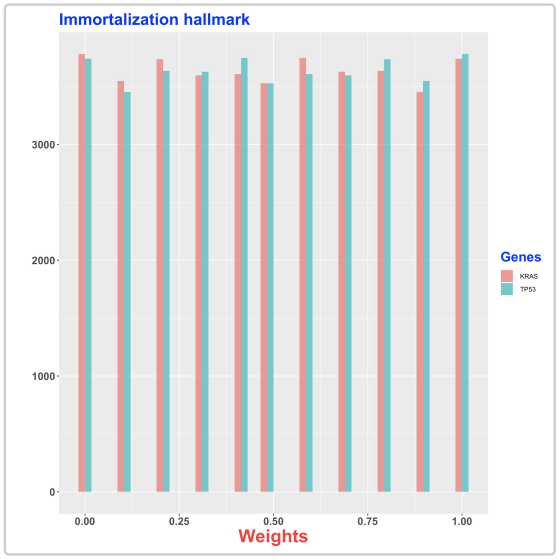
We have prepared initial parameters for all simulations and we have got prior distribution for all weights (Fig.2). The prior is uniformative Dirichlet distribution. After getting weights we added also shuffle procedure in order to keep equal probability for different genes.

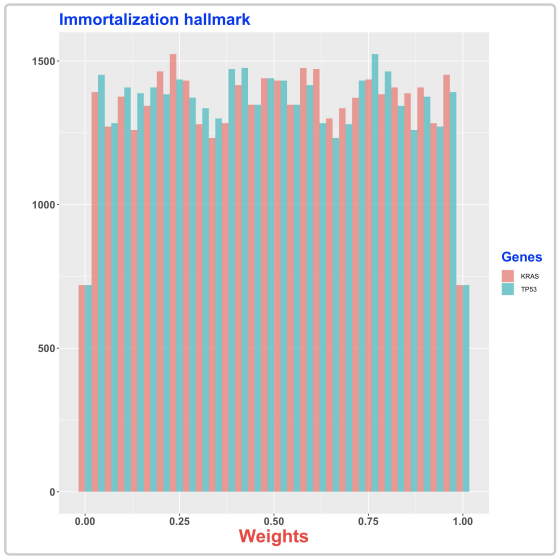


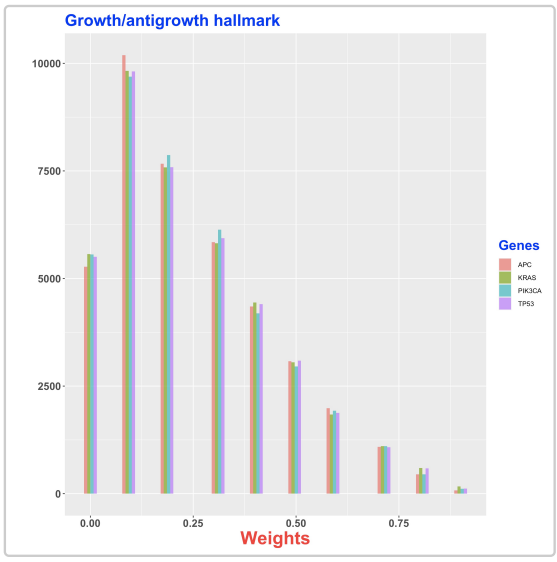


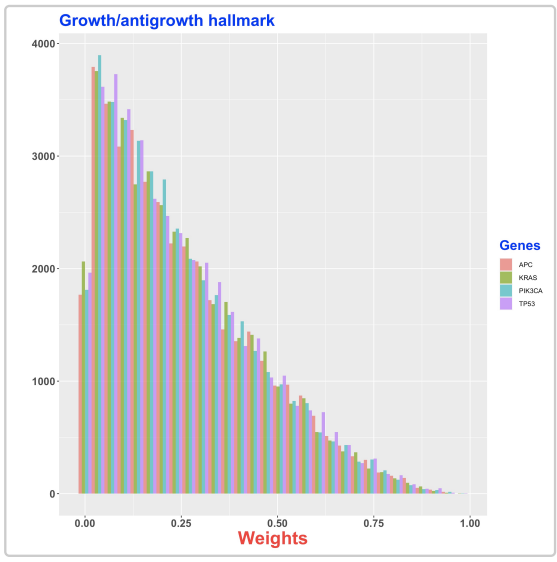

Angiogenesis hallmark
6000
4000
2000
0
0.00
0.25
0.50
0.75
1.00
Weights
Genes
KRAS
PIK3CA
TP53


Angiogenesis hallmark
2000
1000
0
0.00
0.25
0.50
0.75
1.00
Weights
Genes
KRAS
PIK3CA
TP53


Immortalization hallmark
3000
2000
1000
0
0.00
0.25
0.50
0.75
1.00
Weights
Genes
KRAS
TP53


Immortalization hallmark
1500
1000
500
0
0.00
0.25
0.50
0.75
1.00
Weights
Genes
KRAS
TP53


Growth/antigrowth hallmark
10000
7500
5000
2500
0
0.00
0.25
0.50
0.75
Weights
Genes
APC
KRAS
PIK3CA
TP53


Growth/antigrowth hallmark
4000
3000
2000
1000
0
0.00
0.25
0.50
0.75
1.00
Weights
Genes
APC
KRAS
PIK3CA
TP53

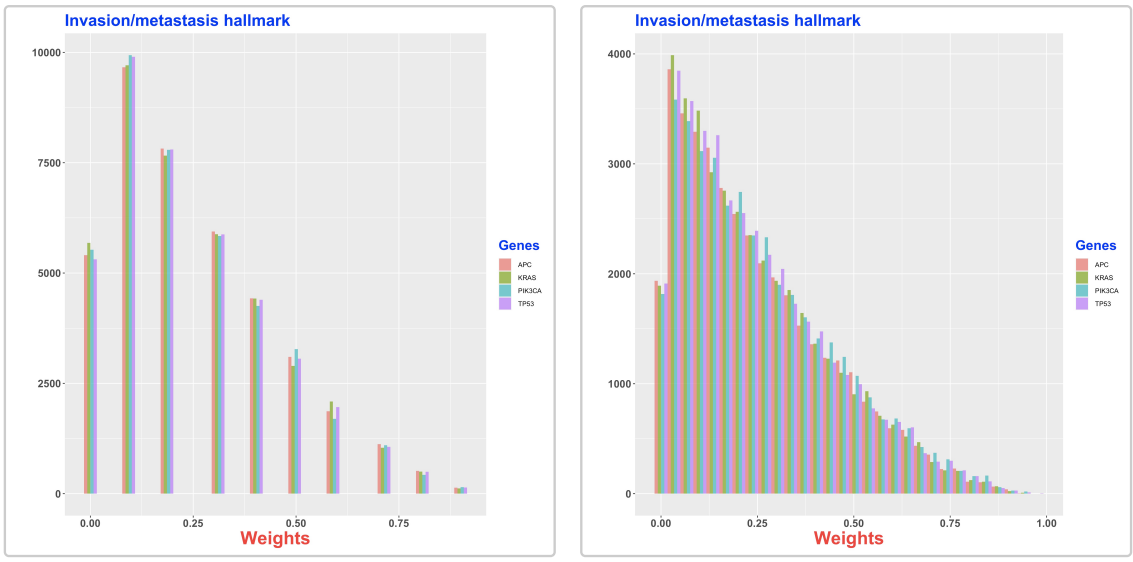


Fig.2. Prior distribution for generated weights: Each plot for different hallmarks $H_a, H_b, H_i, H_d, H_{im}$. **Left side** is for discrete weights, **Right side** is for continuous values of weighs. Number of simulations is $N_{simulations} = 10000$ for each mutated gene (in total 40000 simulations).

---

7 Statistics of simulations

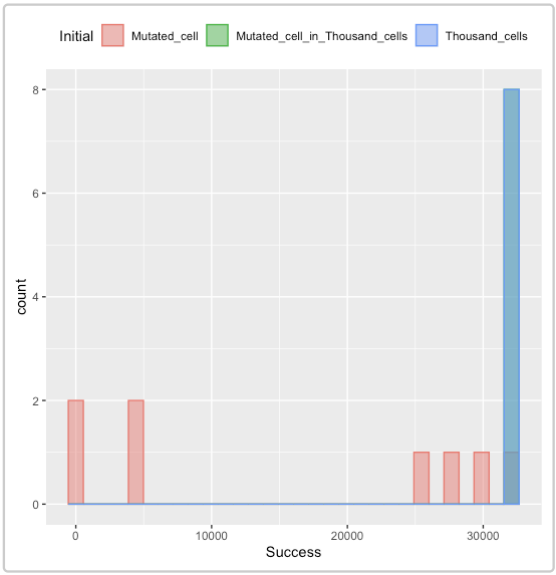
Not all simulations have output and results. Simulations, where all cell were died, have no output at all. But if simulation has output data, it is possibility to get zero VAF as a result. It occurs in the case of passenger mutations only or without mutations or if there is no driver mutations.

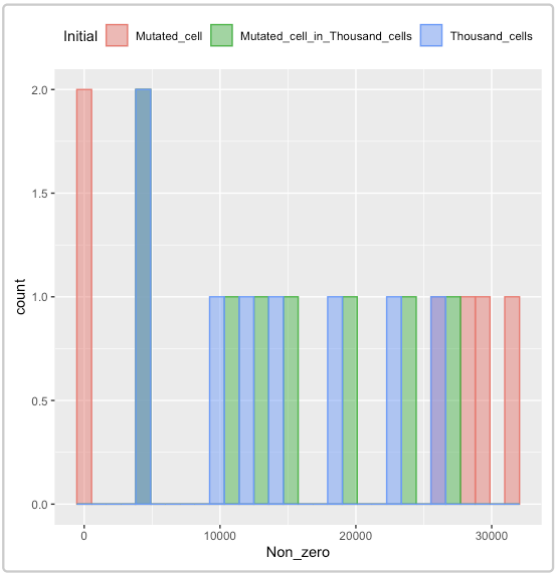
Kindly look at Table 1 with Number of successful and non-zero simulations. In the case of Mutated_cell the simulation repeated 100 times to increase probability of transformation of 1 cell to cancer tumor. Even using repetition of calculation the number of successful simulation is very low but number of non-zero simulation exactly 100% for this case (Fig.1).

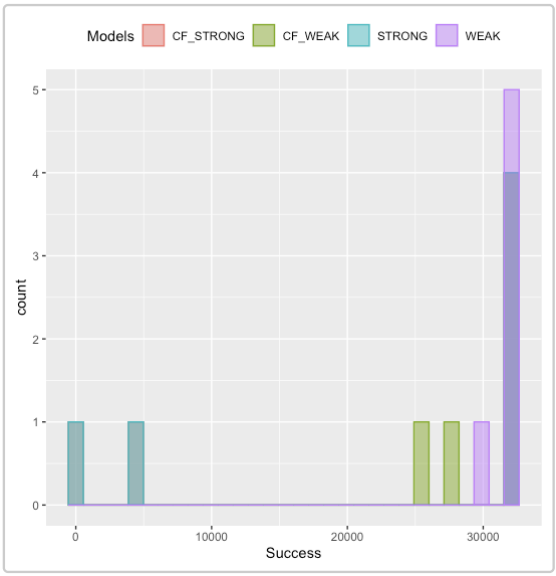
Table 1. Number of successful and non-zero simulations from 40000 simulations

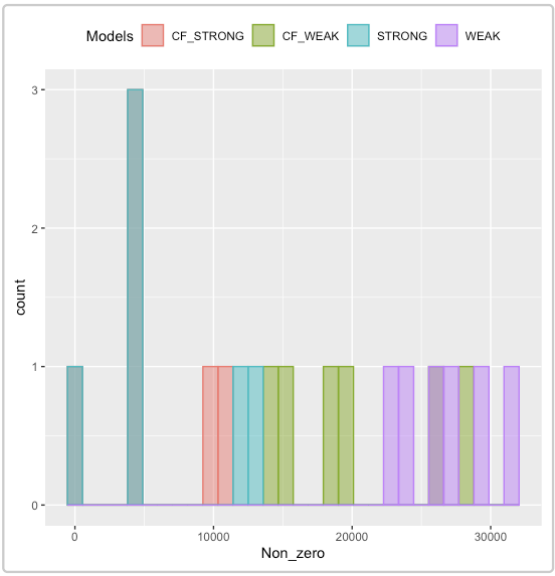
name_weights	name_model	name_init	success	non_Zero
Discrete	STRONG	Mutated_cell	4487	4487
Discrete	STRONG	Thousand_cells	32148	11595
Discrete	STRONG	Mutated_cell_in_Thousand_cells	32138	12630
Discrete	WEAK	Mutated_cell	31560	31560
Discrete	WEAK	Thousand_cells	32149	25781
Discrete	WEAK	Mutated_cell_in_Thousand_cells	32149	26957
Discrete	CF_STRONG	Mutated_cell	4354	4354
Discrete	CF_STRONG	Thousand_cells	32149	10125
Discrete	CF_STRONG	Mutated_cell_in_Thousand_cells	32142	11046
Discrete	CF_WEAK	Mutated_cell	28082	28082
Discrete	CF_WEAK	Thousand_cells	32149	18133
Discrete	CF_WEAK	Mutated_cell_in_Thousand_cells	32149	19534

name_weights	name_model	name_init	success	non_Zero
Continuous	STRONG	Mutated_cell	172	172
Continuous	STRONG	Thousand_cells	32127	4248
Continuous	STRONG	Mutated_cell_in_Thousand_cells	31867	4620
Continuous	WEAK	Mutated_cell	29754	29754
Continuous	WEAK	Thousand_cells	31896	22557
Continuous	WEAK	Mutated_cell_in_Thousand_cells	31896	24265
Continuous	CF_STRONG	Mutated_cell	55	55
Continuous	CF_STRONG	Thousand_cells	31896	4136
Continuous	CF_STRONG	Mutated_cell_in_Thousand_cells	31887	4165
Continuous	CF_WEAK	Mutated_cell	25652	25652
Continuous	CF_WEAK	Thousand_cells	31894	14020
Continuous	CF_WEAK	Mutated_cell_in_Thousand_cells	31894	15576









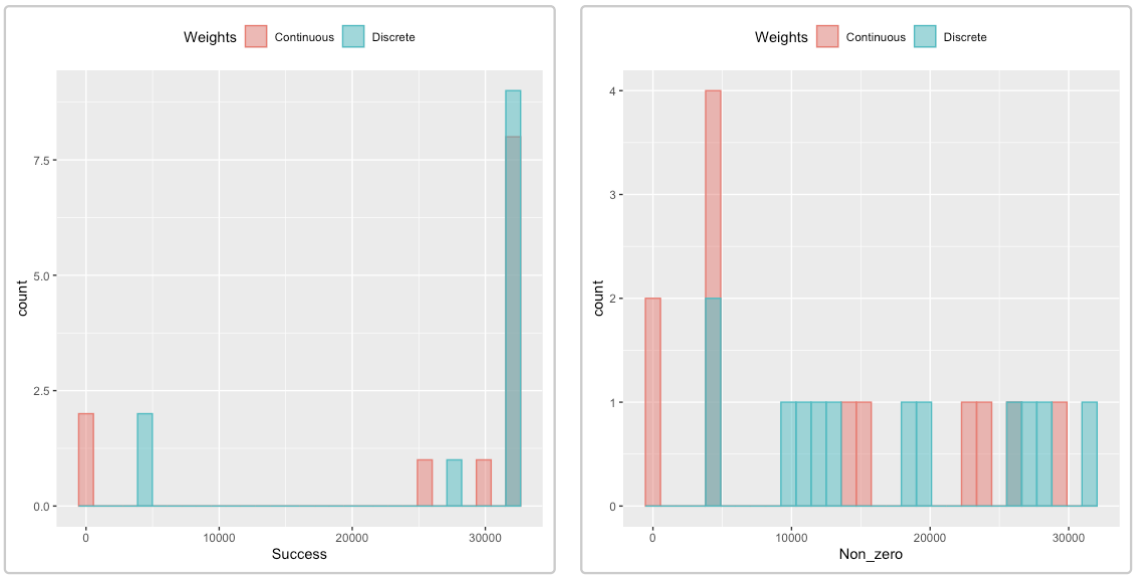


Fig.1. Histograms of statistics of simulations: upper figures are related to different initial cells, middle figures are related to different models, lower figures are related to different types of values of input parameters. The left figures are related to successful simulations, the right figures are related to non-zero simulations.

8 Data base of simulations

Table 2 shows the first 10 rows of data base of simulations. It has information about names of models, initial cells, format of input parameters and id as well as results of simulations (sorting VAF for driver mutation for each gene). Data base has 636646 records from 960000 simulations. Data base has the 10 largest values of VAF for each gene (APC, TP53, APC and PIK3CA) but in the Table 2 we have shown only first values.

Table 2. Part of data base of simulations

name_weights	name_model	name_init	i	APC_max_1	KRAS_max_1	TP53_max_1	PIK3CA_max_1
Discrete	STRONG	Mutated_cell	6	0.0000000	5.00e-01	0.0e+00	0.0000000
Discrete	STRONG	Mutated_cell	16	0.0000000	4.16e-05	0.0e+00	0.5000000
Discrete	STRONG	Mutated_cell	21	0.5000000	0.00e+00	0.0e+00	0.0001250
Discrete	STRONG	Mutated_cell	28	0.0003092	0.00e+00	0.0e+00	0.5000000
Discrete	STRONG	Mutated_cell	36	0.0000000	0.00e+00	0.0e+00	0.5000000
Discrete	STRONG	Mutated_cell	37	0.5000000	0.00e+00	0.0e+00	0.0000000
Discrete	STRONG	Mutated_cell	67	0.5000000	5.00e-01	5.0e-01	0.0975494
Discrete	STRONG	Mutated_cell	70	0.0009695	5.00e-01	0.0e+00	0.0000000
Discrete	STRONG	Mutated_cell	74	0.0000000	5.00e-01	3.6e-05	0.0000360
Discrete	STRONG	Mutated_cell	89	0.5000000	0.00e+00	0.0e+00	0.0000000