**Current status:**

1. **INDELible-** Simulated using guide trees and same params from PIP.
   1. Simulation without inner node labels and root labels. (last time I used labels)
   2. This is to see the Ancestral sequences, which is provided by the simulator at the output.
   3. Corrected the branch length decimal point error by rounding its values up to 4 decimal points. (**strDouble error in ProPIP)**

Max’s approach discussed in the last meeting.

**Steps proposed by Max**

***“ I.*** *Prepare a table with probability 1 for inserting only 1 character and 0 elsewhere ("emulates PIP")*

***II.****Fix the length of the ancestral sequence (root) L.*

*For us The Sequence at root is the sequence length that we provide, L+ number of INDELs (from Zipfian distribution), the root contains only \* (INSERTIONS). Below we have 300 characters (We provide 200 sites as L (since we used E= 200 in PIP) + 100 sites from Zipfian distribution)*

***ANCESTRAL Sequence INDELible***A screenshot of a video game

Description automatically generated

***TRUE MSA INDELible***

The **leaf 7** is empty in the above sequence. Like this we have 13 sequences out of 100 sequences.

1. *Simulates sufficient MSAs to obtain a clear distribution of lengths at the root.*

*These according to your previous observations should be L or L+1.* I didn’t understand.

***2.***  *Simulate sufficient MSAs with*

1. *lambda = mu = {...0.8,0.9,1.1,1.2,...} and try to understand what influence these values have on the* ***average length of the sequences at the leaves***
2. *lamda<mu What values should I use?*
3. *Do you get geometrically distributed lengths?*

***3.*** *Compare the results obtained with 1) and 2) from. above. It is possible that in 1) with a tree long enough the sequences vanish.* ***“***

1. **ProPIP –**

I generated values of lambda and mu which we use when aligning with ProPIP.

There are 2 methods:

1. Using the method implemented by Max (levenberg-marquardt algorithm, non-linear least square data fitting method). This provide a pair of lambda and mu for each sequence and Tree.

2. By Theoretical equation using E and I. For INDELible and real data which E and I should be used? As for now for all I used E=200 and I = 0.5 as per PIP settings from Hossein.

For this, I tried for one sequence and tree by manually editing the params file with the output of the theoretical equation (2) and received the same error given below.

[Setting up substitution model]

FixedFrequenciesSet::setFrequencies. Frequencies sum must equal 1 (sum= 1.04762)

1. **PRANK-** 
   1. 13/100 sequences contain empty leaves.
   2. This we run without guide tree. Instead I used the PRANK generated guide tree at the end.
   3. **Keeping as it is at the moment (as per last meeting)**
2. **MAFFT-**

1. Converted Newick tree to mafft tree using ruby.
2. Cross checked the ruby file that was used by Max with the file that I have. Both are same.
3. Tried for all sources (INDELible, PIP, real data) of sequences and tree. All received same error (TRY 1 error).
4. Aligned without Guide Tree. (Which was not recommended in last meeting)

**TRY 1:**

mafft --auto --leavegappyregion --treein ./mafftTree\_P\_1.mafft ./sim-0\_MSA.fasta > ./MAFFT\_Aligned\_P\_1.fasta

outputhat23=16

loadtree.

treein = 108

compacttree = 0

stacksize: 8192 kb

rescale = 1

All-to-all alignment.

tbfast-pair (aa) Version 7.453

alg=L, model=BLOSUM62, 2.00, -0.10, +0.10, noshift, amax=0.0

0 thread(s)

outputhat23=16

Loading 'hat3.seed' ...

done.

Writing hat3 for iterative refinement

rescale = 1

Gap Penalty = -1.53, +0.00, +0.00

Loading a tree

0 / 8

done.

/usr/local/bin/mafft: line 2719: 71365 Segmentation fault: 11 "$prefix/tbfast" \_ -u $unalignlevel $localparam -C $numthreads $seqtype $model -g $lexp -f $lgop -Q $spfactor -h $laof -L $usenaivepairscore $focusarg \_ -+ $iterate -W $minimumweight -V "-"$gopdist -s $unalignlevel $legacygapopt $mergearg $termgapopt $outnum $addarg $add2ndhalfarg -C $numthreadstb $rnaopt $weightopt $treeinopt $treeoutopt $distoutopt $seqtype $model -f "-"$gop -Q $spfactor -h $aof $param\_fft $localparam $algopt $treealg $scoreoutarg $focusarg < infile > /dev/null 2>> "$progressfile"

**TRY 2:**

mafft --phylipout --leavegappyregion --treein ./mafftTree\_P\_1.mafft ./sim-0\_MSA.fasta > ./MAFFT\_Aligned\_P\_1.fasta

**error**

nthread = 0

nthreadpair = 0

nthreadtb = 0

ppenalty\_ex = 0

stacksize: 8192 kb

rescale = 1

Gap Penalty = -1.53, +0.00, +0.00

loadtree.

Loading a tree

0 / 8disttbfast(91477,0x7fff99ed2380) malloc: \*\*\* mach\_vm\_map(size=18446744069442375680) failed (error code=3)

\*\*\* error: can't allocate region

\*\*\* set a breakpoint in malloc\_error\_break to debug

/usr/local/bin/mafft: line 2719: 91477 Segmentation fault: 11 "$prefix/disttbfast" -q $npickup -E $cycledisttbfast -V "-"$gopdist -s $unalignlevel $legacygapopt $mergearg -W $tuplesize $termgapopt $outnum $addarg $add2ndhalfarg -C $numthreads-$numthreadstb $memopt $weightopt $treeinopt $treeoutopt $distoutopt $seqtype $model -g $gexp -f "-"$gop -Q $spfactor -h $aof $param\_fft $algopt $treealg $scoreoutarg $anchoropt < infile > pre 2>> "$progressfile"

**Pending:**

# Overleaf Update

# INDELible vs PIP

To DO

1. Convert sequence file labels to corresponding integer representation for mafft.
2. INDELible simulation with different values of lambda and mu. 2 methods proposed by max.
3. Write a letter to author INDELible. Make draft and send to Manual and Maria, Max.

Manuel GilIdeally the sequence length L at the root should be sampled from the distribution of sequence lengths implied by the model of insertions and deletions (Thorne et al. 1991). However, sampling from this distribution is complicated because of the arbitrary nature of the indel-size distribution accepted by INDELible. Instead, we require L to be specified by the user. The sequence at the root is then generated by sampling L characters (nucleotides, amino acids, or codons) at random from the equilibrium distribution under the substitution model at the root.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Dear all,

@max The above plot refers to number of astericks per Iteration per replicate.

I would to point some points from the paper as I was investigating what is method 1 and method 2 he is referring and what evidence he uses for method 1 and method2.

At the beginning of the paper says, “*The models we have implemented assume that the insertion and deletion rates are constant among sites in the sequence. As a result, the substitution process is independent of insertions and deletions, and substitutions can be simulated separately from insertions and deletions. Thus, an alternative procedure is to use the* ***Gillespie algorithm to simulate indels only****, with substitutions simulated afterward by sampling from the transition probability matrix for the branch (Yang 2006, p. 303). This is the method used by Cartwright (DAWG) (2005), and will be referred to in this paper as method 1. The method described above, of simulating waiting times for substitutions as well as insertions and deletions, is referred to as method 2.*

Later in the paper I see this, *“Note that simulation of the evolutionary process by Gillespie’s algorithm (INDELible method 2 but not method 1 or DAWG) is possible as long as one can generate the sequence at the root of the tree and calculate the instantaneous rates; there is no need for matrix-exponential solutions to the transition probabilities, contra Varadarajan et al. (2008).*

Finally, in the speed comparison section he compares method 1 and INDELible (not mentioned which method and I guess this is method 2 as per above evidence).

“Speed differences between INDELible and DAWG are largely a matter of programing design. Both programs are written in C++, and both programs store sequence .

INDELible (I guess he is referring to INDELible method 2) implements insertions via **a modified lookup table** whose execution time is mostly independent of the complexity of the simulation but can be slow in very simple simulations.

*DAWG* (method 1) *implements insertions via the C++ function vector::insert, the speed of which is proportional to the number of elements inserted (copying) plus the number of elements between the insertion position and the end of the vector (moving).*

The paper at the beginning also refers to this *“DAWG (Cartwright 2005) cannot simulate amino acid or codon sequences, whereas SIMPROT (Pang et al. 2005) and indel-Seq-Gen (Strope et al. 2007) cannot simulate nucleotide or codon sequences.”* If my understanding from the paper is correct, then we used method 1 that is DAWG method as he says, and we generated amino acid sequences. This contradicts what he is referring to.

*Our extensive comparison with DAWG revealed a few problems with DAWG version 1.1.2 and earlier. For exam- ple, two biological mechanisms can generate columns with all gaps in the true alignment: 1) deleted insertions, that is, deletion of part of an earlier insertion on the same branch, and 2) parallel deletions, that is, deletion of the same nucleo- tides along different lineages. DAWG keeps track of 2) but not of 1). Furthermore, the true alignment produced by DAWG may be incorrect with nucleotides from parallel in- sertions misaligned. Those bugs will be fixed in a new release of the program (Cartwright R, personal communication).*

Comparing method 1 and method 2 through the data that I generated.

1. Both methods generated **200 amino acid characters + number of astericks (insertions/\*)** at the ROOT.

Tests at ROOT

26.03.2020

The following conditions were used for the tests:

1. **Test1: Balanced Tree, br. L=0.1, L=200, Lambda=Mu= {0.01,0.02,0.03,0.04,0.05} with Samples= {5000,5000,4155,954,317} for each pair respectively.**

After observing all the probability distribution plots from this test with a reference Power law distribution with exponent 1.7, only the first pair (lambda=mu=0.01, samples= 5000) follows a power law.

1. **Test2: Balanced Tree, br. L=0.1, L=200, Lambda= {0.01,0.02,0.03,0.04} < Mu= {0.05} with Samples= {1246,105,5000,1629} for each pair of Lambda and Mu respectively.**

After observing all the probability distribution plots from this test with a reference Power law distribution with exponent 1.7, only the first pair (lambda=0.01, mu=0.05, samples= 1246) follows a power law.

1. **Test3: Balanced Tree, br. L=0.1, L=300, Lambda=Mu= {0.01} with Samples= {956}.**

This test is carried out to see if the user defined L is included in the distribution or not. From this test it is clear that the initial L is also included in the Zipfian distribution, as we can see the shift in the distribution.

Our requirement was 5000 samples, however the tool crashed at the above given sample sizes.

**@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@**

Hi all,

the insertions under Indelible doesn't really satisfy me.

In my opinion the generation of random numbers given a distribution is not very robust.

Another problem I observe is the very very long queue that the power law distribution has.

Here you can see random numbers generated (in matlab) under the zeta distribution (discrete power law) with lambda=1.7;

A close up of a white background

Description automatically generated

As you can see the tail is long (10^7 characters).

Here a magnification of the first 100 values.

A picture containing white, sitting, kitchen, light

Description automatically generated

The problem with the very long queue is that some simulations kill the job or generate unrealistic data.

In the continuum the power law would be something like this (Y~log(pareto\_distr(1.7))

A close up of a white wall

Description automatically generated

which as you can see quickly becomes evanescent (4-5 chars).

I would suggest at this point to use in Indelible a custom function.

I implemented the zero-truncated poisson distribution (zero-truncated because it makes no sense to enter 0 characters).

Here the distribution generated with lambda=1.7.

A picture containing white, large

Description automatically generated

What do you think about it?

**@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@**

Hi all,

these results surprise me and I don't understand them.

First of all 2 considerations:

#1

When constructing the differential equations of TKF91 model the probability that a single link generates 2 or more links is O(dt) and is therefore negligible. So we can assume that each link generates at most only 1 descending link in a very small intervall of time dt.

#2

The Doob-Gillespie algorithm extracts 3N+1 (for TKF91) or 2N+1 (for PIP) exponential distributed random numbers (N is the number of links) and the smaller of all decides the time when the event occurs and the type of event (replacement, insertion, deletion). At the instant t=0, in theory, the algorithm should only extract the ancestral sequence with imposed length L or following a certain distribution (geometric or poisson).

Now, while accepting that the algorithm at the instant t=0 as well as to initialize the ancestral sequence makes it already evolve, it does not explain these results.

In fact, according to my knowledge, all the possible events (let's say 3N+1) compete but only one of them wins and is executed. But if so, we should find a distribution of lengths with only 3 values: 199-200-201. Any other value sounds strange to me. Here it seems that several insertions of a single character (as imposed by the distribution) occur at the same time, and also only insertions occur (without any deletion, also very anomalous).

I am definitely confused.

@Eldhose

Using the discrete dirac distribution ([1 0 0 0 0 ...]) do you notice a difference on the expected length of the final sequences with

1) lambda=mu and

2) lambda<mu ?

**@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@**

Dear all,

I plotted the average sequence length on each MSA using Histogram as well as Boxplot (due to higher number of replicates, I choose the initial 20 for all).[@Maiolo Massimo (maio)](mailto:maio@zhaw.ch) The sequence length variation on both test are clear, however the variation is not between 199-201. Most of them in my view follow a Gaussian distribution.

In test4, with discrete Dirac distribution, when the lambda and mu are equal to 0.01 the histogram contains gaps.  At lambda= mu= 0.02 the average sequence length become very large. The same issues are clear also from Boxplot too.

In test5, with Zero Truncated Poisson distribution, all the conditions outputs gaussian distribution in my perspective.

Please see the attached plots.

**Text for Ziheng Yang**

My name is Eldhose Poulose. I am a master’s student at Zurich University of Applied Sciences (ZHAW). I am doing my master thesis in Maria Anisimova's group, as part of my thesis I am using INDELible to simulate data. From the article describing INDELible, the user can specify a number of characters at the root, denoted as L, also used as the current sequence length. However, in addition to characters, we observe that insertions are also generated at the root (i.e., L+ insertions(\*). We carefully read the paper and cannot find the description of how insertions are generated at the root.

Could you please provide more detail?

We are also interested to know the distribution of the full sequence length at the root.

Can we set an expected sequence length?

To understand more we have conducted several tests with Zipfian distribution (1.7) by varying Insertionrate and Deletionrate(test1 and test2). Also, the same with discrete Dirac distribution (test4). The variation in the number of \* at the root and also average sequence length per MSA from theses distributions confuses me to make a decision. I would like to know if this is something exactly what INDELible will be providing or is there any error in these results. The corresponding plots you can see in the attachment.

If you need more information to answer these questions, feel free to write me back.

Best regards

Eldhose

**MEETING 31.03.20**

Manuel:

Maria:

Max: