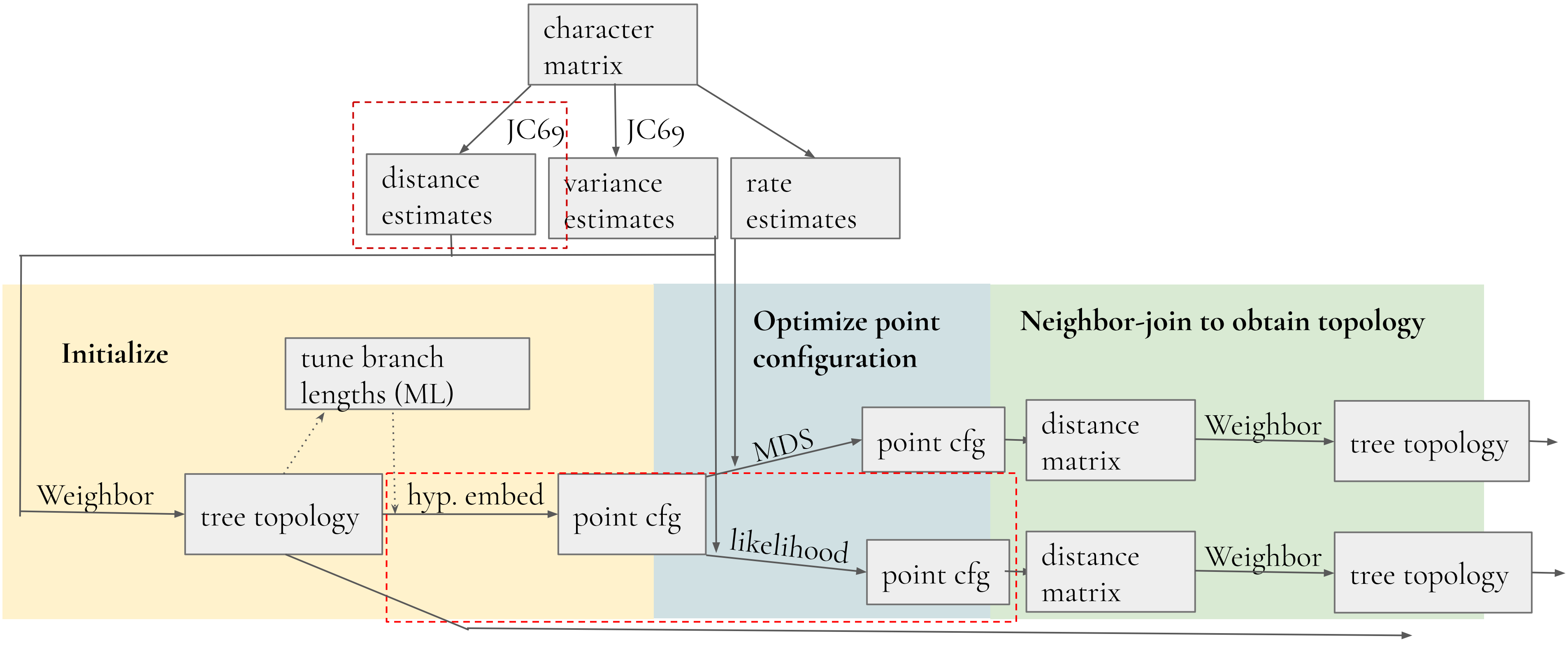
Hyperbolic point configurations for CRISPR lineage tracing (pt. 2)

Anthony Ozerov & Sitara Persad

Technique overview [Wil21]



Likelihood method of optimizing point configuration [Wil21]

Jukes-Cantor model

$$P_{ab}(t) = \begin{cases} \frac{1}{4} + \frac{3}{4}e^{-4t/3} =: P_{\text{diag}}, & \text{if } a = b \\ \frac{1}{4} - \frac{1}{4}e^{-4t/3} =: P_{\neg\text{diag}}, & \text{otherwise,} \end{cases}$$

= conditional probability of observing b at the site t time after observing a

t, the time, is effectively evolutionary distance

Smooth! No reference to tree topology (like in maximum-likelihood tree-search) or discrete topology moves needed. Wil21 further shows that maximizing **l(x)** roughly maximizes the log-likelihood of the tree

Likelihood function of distance between 2 points

$$\mathcal{L}(t) = \prod_{\text{sites } \sigma} \pi_{\sigma_i} P_{\sigma_i \sigma_j}(t),$$

Log-Likelihood function of distance between 2 points

$$\log \mathcal{L}(t) = \sum_{\text{sites } \sigma} \log P_{\sigma_i \sigma_j}(t) + C,$$

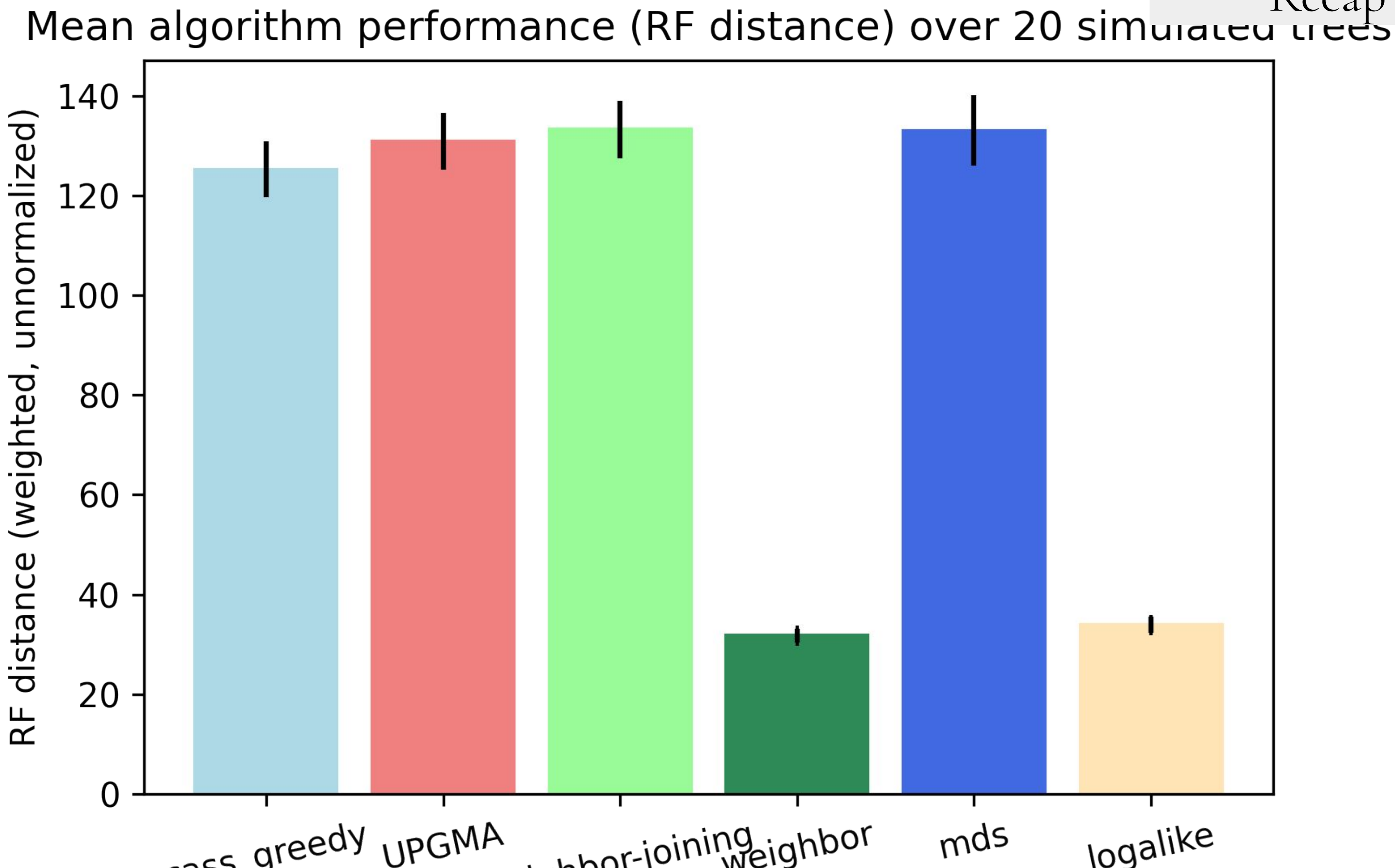
Log-Likelihood function of point configuration (with constant terms removed)

$$\boxed{\mathbf{l}(\mathbf{x}) = \frac{1}{L} \sum_{i \neq j} \sum_{\text{sites } \sigma} \log P_{\sigma_i \sigma_j}\left(\boxed{\mathrm{d}(x^i, x^j)}\right).}$$

Maximize this

What hyperbolic space changes

Recap



Mean algorithm performance (RF distance) over 20 simulated trees
RF distance (weighted, unnormalized)
0
20
40
60
80
100
120
140
UPGMA
weighbor
mds
logalike

We can do better...

JC69 model: Equal mutation rates

$$Q = \begin{bmatrix} -3\mu/4 & \mu/4 & \mu/4 & \mu/4 \\ \mu/4 & -3\mu/4 & \mu/4 & \mu/4 \\ \mu/4 & \mu/4 & -3\mu/4 & \mu/4 \\ \mu/4 & \mu/4 & \mu/4 & -3\mu/4 \end{bmatrix}$$

CRISPR Lineage tracing

$$Q = \begin{bmatrix} -\mu_1 & 0 & \lambda_{1,3} & \lambda_{1,4} & d_1 \\ 0 & -\mu_2 & \lambda_{2,3} & \lambda_{2,4} & d_2 \\ 0 & 0 & -d_3 & 0 & d_3 \\ 0 & 0 & 0 & -d_4 & d_4 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{array}{l} \text{Targets} \\ \\ \text{Non-targets} \\ \\ \text{Deletions} \end{array}$$

So both of the following in the method discussed are currently invalid for CRISPR lineage tracing:

- distance matrix used for neighbor-joining
- distance gradient used for point tuning

Maximum-likelihood distance estimates

Distances ←→ evolutionary times.
Obtain MLE of evolutionary time by finding *t* that maximizes likelihood of transitioning from an ancestor to *i* and *j* within time *t*.

$$Q = \begin{bmatrix} -\mu_1 & 0 & \lambda_{1,3} & \lambda_{1,4} & d_1 \\ 0 & -\mu_2 & \lambda_{2,3} & \lambda_{2,4} & d_2 \\ 0 & 0 & -d_3 & 0 & d_3 \\ 0 & 0 & 0 & -d_4 & d_4 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Targets

Non-targets

Deletions

Trivial but extremely inefficient:

$$P(t) = e^{tQ}$$

time complexity:
[gigantic constant]×[#states]³

$$\mathscr{L}_{i,j}(t) = \prod_{\text{sites } \sigma} \sum_{a \in A(\sigma_i,\sigma_j)} \pi_a P_{a,\sigma_i}(t/2) P_{a,\sigma_j}(t/2)$$

Prior on *a* being the ancestral state

Numerically optimize to find MLE of *t*.
(is there a better way?)

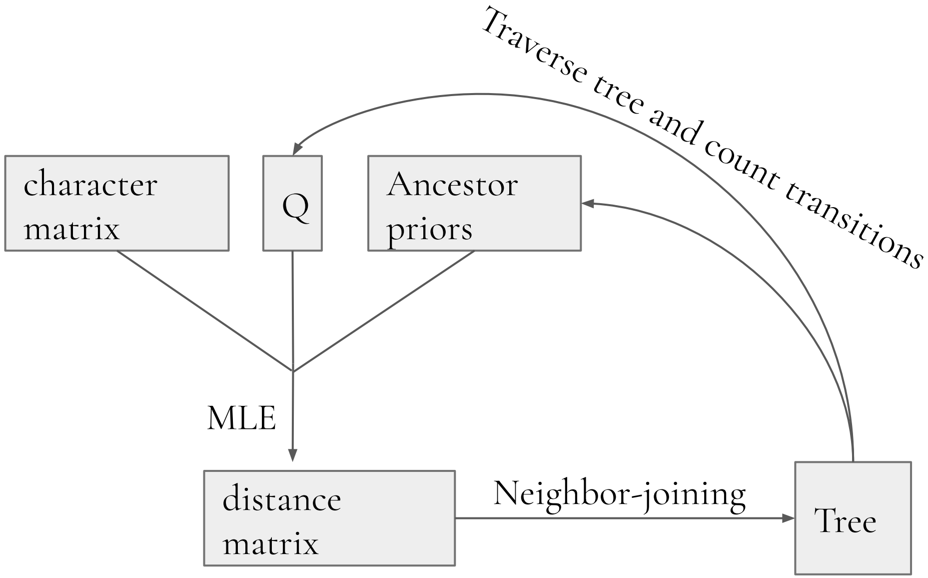
Faster, taking into account special properties of Q (assuming no deletions):

$$p(a, \sigma_i, t) = \begin{cases} e^{-\mu_a t} & a = \sigma_i \\ (1 - e^{-\mu_{a,i} t})\lambda_{a,i} / \sum_j \lambda_{a,j} & a \neq \sigma_i \end{cases}$$

$$\mathscr{L}_{i,j}(t) = \prod_{\text{sites } \sigma} \sum_{a \in A(\sigma_i,\sigma_j)} \pi_a p(a, \sigma_i, t) p(a, \sigma_j, t)$$

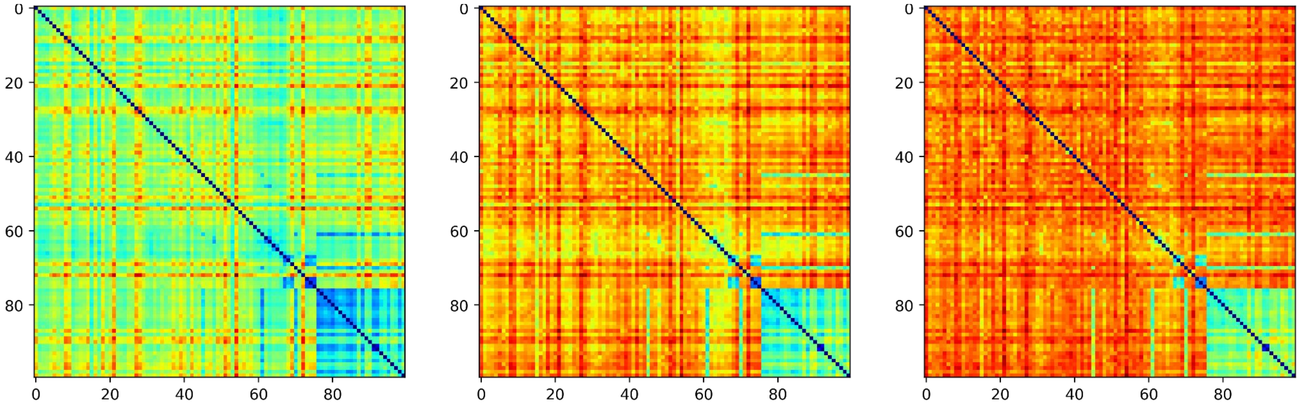
(Need to think more about incorporating deletions...)

EM Algorithm



Distance matrix comparison

EM Algorithm | CRISPR Hamming Distance | Hamming Distance



Double-count sites where cells x and y differ
and are both not the initial state

Neighbor-joining results (benchmark data from DREAM challenge)

Mean normalized Robinson-Foulds distances for 11 data sets

algo	EM	hamm-NTR	hamm-TR
weighbor	0.756030	0.870130	0.839518
nj	0.757885	0.745826	0.769944
upgma	0.853432	0.817254	0.811688
nj-lca	0.770872	0.906308	0.911874

Solid baseline performance

Cool algorithm relying on known aspects of lineage tracing data. Doesn't really improve things...

Standard deviation of means

algo	EM	hamm-NTR	hamm-TR
weighbor	0.010450	0.006701	0.007660
nj	0.010983	0.009769	0.007463
upgma	0.008530	0.009857	0.006204
nj-lca	0.008836	0.005218	0.003995

No difference between NJ performance on EM and hamm-NTR distance matrices. Likely due to only one guide.

Weighbor does much better on EM distance matrix—makes sense as its maximum-likelihood aspects rely on a maximum-likelihood distance matrix.

Yosef Lab's neighbor-joining method with clever missing-value handling:

0.694805

TODO:

- Finish incorporating ancestor priors
- Improve handling of missing values
- Test on third-party benchmarking data with multiple guides
- Change the point tuning in hyperbolic space to use a gradient based on the better distance estimates (Cython stuff...)