Improving Δ-affinity metrics in “Using B cell receptor lineage structures to predict affinity”

Duncan K. Ralph[1,*], Frederick A. Matsen IV[1]

1 Fred Hutchinson Cancer Research Center, Seattle, Washington, USA

[*] **dralph@fredhutch.org**

Summary

We’ve recently realized that there was an issue with the performance metric that we used to evaluate Δ-affinity metrics (e.g. nuc-lbr) in [1]. In short, we were inadvertently ignoring performance on a (somewhat rare) class of nodes. Using a different metric to evaluate Δ-affinity performance, we find that optimal τ for nuc-lbr is $1/\ell_{\text{seq}}$ (rather than a much larger value), and also that aa-lbr substantially outperforms nuc-lbr (as opposed to similar performance). While the net effect of these changes isn’t necessarily large (plain nuc-lbr still works ok, and isn’t terrible with any τ value), it means we’re changing default partis behavior so wanted to write down the justification.

Metrics for evaluating Δ-affinity

To measure Δ-affinity performance in [1] (see the “Evaluation framework” section of Methods), for each node we counted the number of steps necessary to move upwards on the tree until reaching a branch with an affinity-increasing mutation. So, a value of zero steps means the node is immediately below such a branch, and if e.g. nuc-lbr picks such a node it is performing perfectly. We listed several reasons for only looking upwards, and while we still think these reasons are more or less true, they sidestep a more important issue: only looking upwards requires ignoring nodes that are above every affinity-increasing mutation in the tree. In many (perhaps most) trees, such nodes are rare; however when using a very large τ value (as indicated by the optimization in [1]), parents get too much credit for their grandchildren, so such nodes commonly have quite large nuc-lbr values.

Looking downwards introduces some complications: it requires iteratively searching the offspring of each offspring, as well as deciding how to compare N steps upwards from the node (positive steps, by convention) to N steps downwards (negative). In order to ameliorate these complications, we give up on keeping track of the number of steps separating a node from an affinity-increasing branch, and simply count nodes immediately below such a branch (zero steps) as correct, and all others as incorrect (nodes immediately above are given -1 steps). Some example distributions of nuc-lbr and aa-lbr for this definition of correct and incorrect are shown in S1 Fig and Fig 1. Note that a node is only counted as correct if it is immediately below (but not above) the branch. This asymmetry reflects the asymmetry introduced in the definition of nuc-lbr between numerator and denominator. Calculation of nuc-lbr on a node separates the tree into two parts: that with (numerator) and without (denominator) a potential affinity-increasing mutation. We can visualize this if we split the non-zero distribution into its component parts (S2 Fig), where we see that for N steps of -1 (nodes immediately above an affinity-increasing branch), nuc-lbr is distributed almost the same as for $|n| > 1$.

As in [1], we want to summarize each such plot as a single number in order to allow averaging over many samples. Imagining a typical use case where we might choose the top few nodes by nuc-lbr as likely to be immediately below affinity-increasing branches, we summarize by choosing the top N nodes from each family, and calculate the fraction that are correct. We performed similar summaries also by fraction of nodes (e.g. choose the top 3% of each family) and with a hard cutoff in nuc-lbr (e.g. choose all nodes with nuc-lbr > 7); however these add little improvement, and have several downsides (results not shown).

We used this top N nodes summary for $N = 3$ to perform a new τ optimization (Fig 2), finding as expected that with the new, more sensible metric that optimal τ is $1/\ell_{\mathrm{seq}}$ (the same as nuc-lbi). Comparing nuc-lbr and aa-lbr more directly, we see that aa-lbr now performs significantly better (Fig 3). This agrees with the significant improvement that we found for aa-lbi compared to nuc-lbi.

Conclusion

Together these results leave us in, we think, a much simpler and more intuitive place: results for lb indices and ratios are now essentially analogous. The two changes that result are that in partis, nuc-lbr and aa-lbr are now calculated with τ of $1/\ell_{\mathrm{seq}}$ (rather than 20 times this, which was the previous default); and that we recommend using aa-lbr over nuc-lbr. See the manual for details on calculating these metrics `https://git.io/Jfesw`).

Acknowledgements

Many thanks to Tatsuya Araki and Gabriel Victora for pointing out some trees where this issue was particularly evident.

References

1. Ralph DK, Matsen FA 4th. Using B cell receptor lineage structures to predict affinity. PLoS Comput Biol. 2020 Nov;16(11):e1008391.

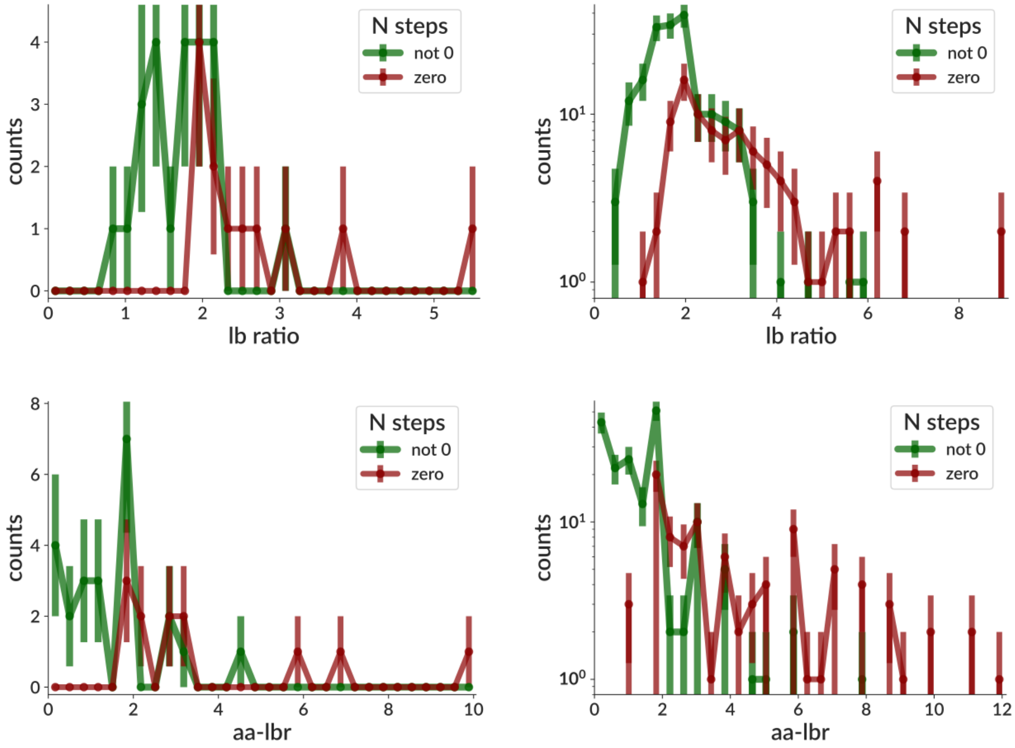


Fig 1. **Simulation performance of nuc-lbr (top) and aa-lbr (bottom)** on a single representative family (left) and combined for 10 representative families (right). Shown as distributions of metric values for "correct" nodes in red (zero steps away from, i.e. immediately below, a branch with an affinity-increasing mutation) and "incorrect" nodes in green (not immediately below such a branch). These families were simulated to a time of 300 generations with a carrying capacity of 1000, and 10 sequences were sampled at generations 100, 150, 200, 250, and 300. Both metrics were calculated with τ of 0.002. Similar plots sampling 50, rather than 10, sequences per generation are in S1 Fig. Trees corresponding to the left hand plots are in S3 Fig.

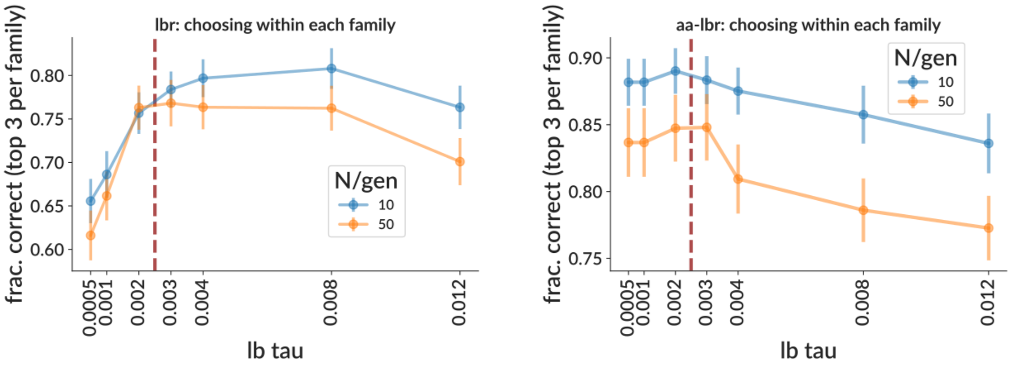


Fig 2. **Simulation performance of nuc-lbr (left) and aa-lbr (right) as a function of** τ averaged over many samples. Shown as the "fraction correct" (i.e. fraction that are immediately below an affinity-increasing branch) if choosing the top three sequences from each family with the indicated metric. This fraction is the ratio of red to red plus green in Fig 1 and S1 Fig, summed over bins starting from the right until the sum equals three (and uses the same parameters as those figures). Each point is the average performance of ten replicates (± standard error), where each replicate consists of the average performance over ten families.

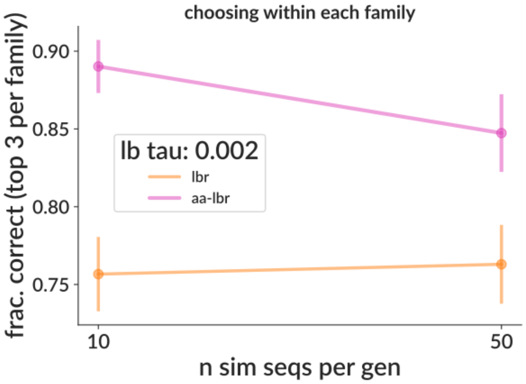
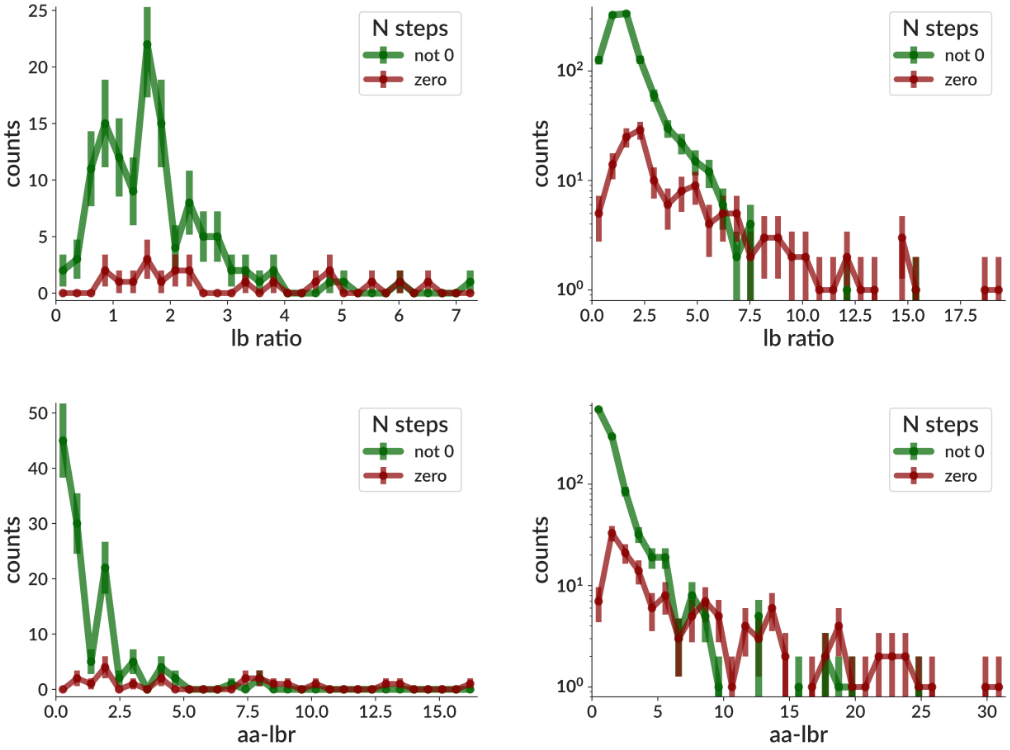
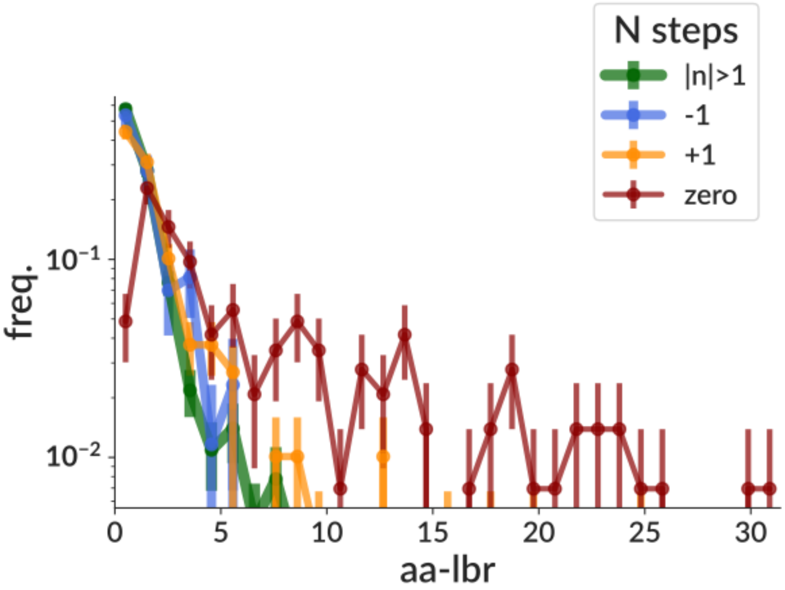


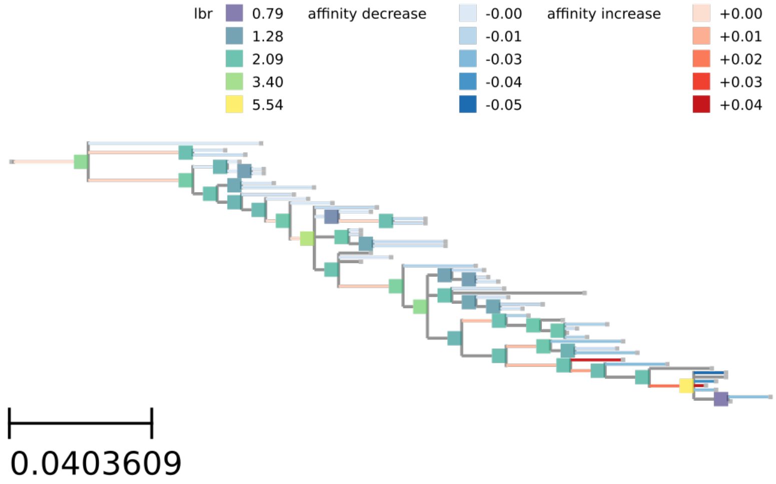
Fig 3. **Comparing Simulation performance of nuc-lbr and aa-lbr at near-optimal** τ**.** These are simply a subset of the points in Fig 2.

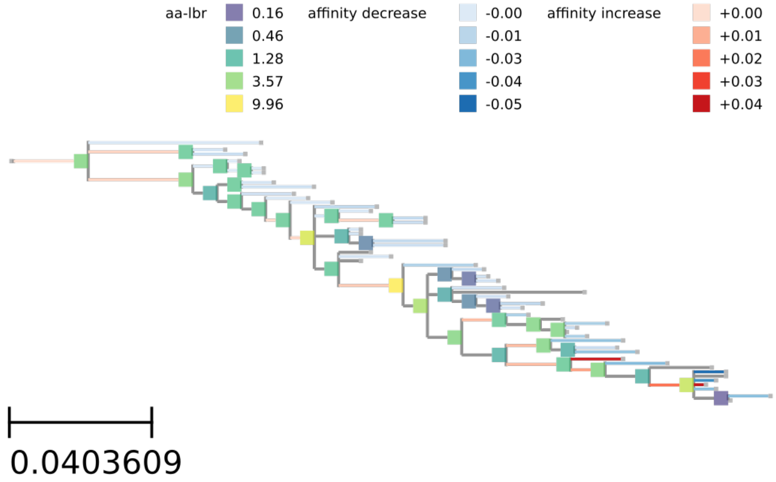


S1 Fig. Same as Fig 1, except sampling 50 sequences at each timepoint rather than 10. Trees corresponding to the left hand plots are in S4 Fig and S5 Fig.

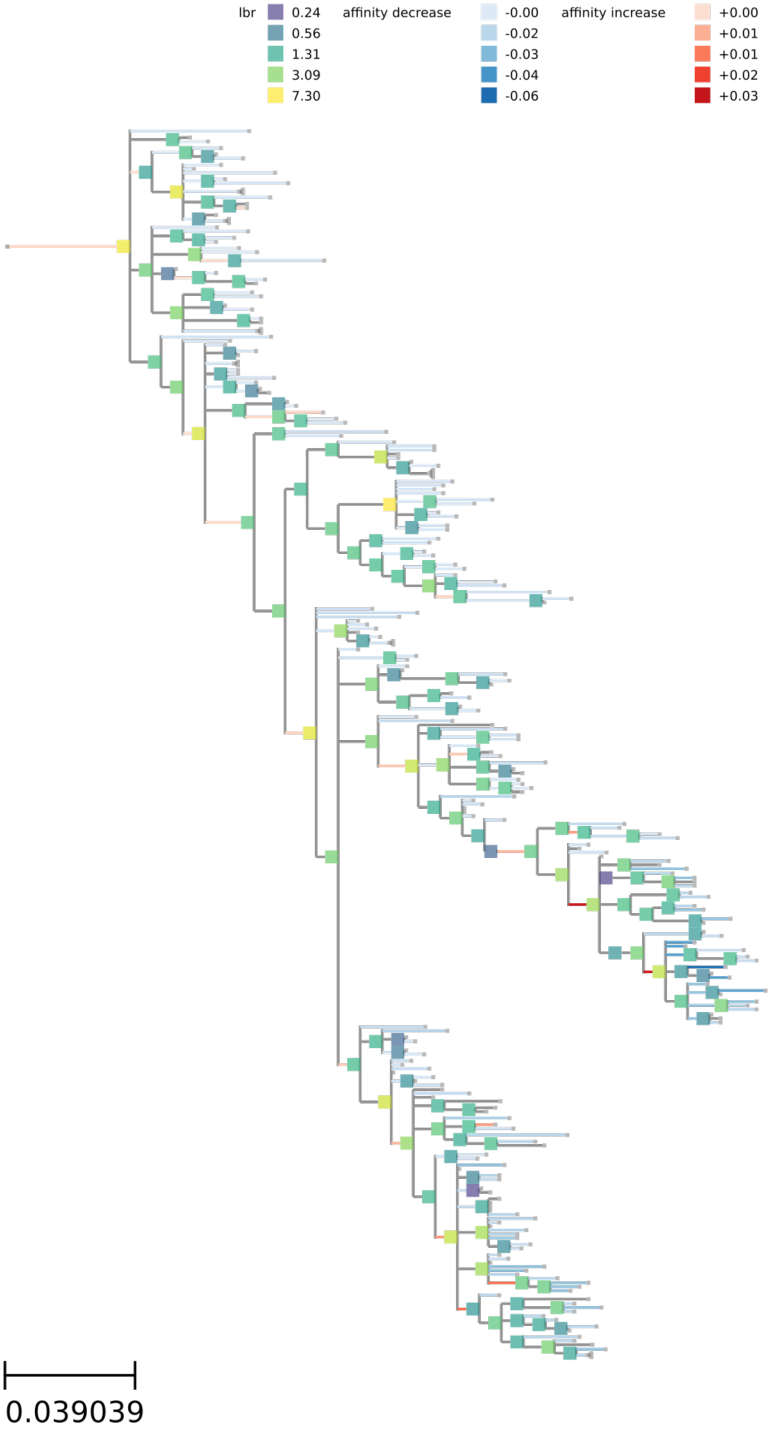


S2 Fig. **Distributions of aa-lbr for different number of steps.** Similar to bottom right of S1 Fig, except that the "not 0" category has been split in three: -1, $+1$, and $|n| > 1$. Note that to facilitate shape comparisons, the histograms have also each been normalized to one.

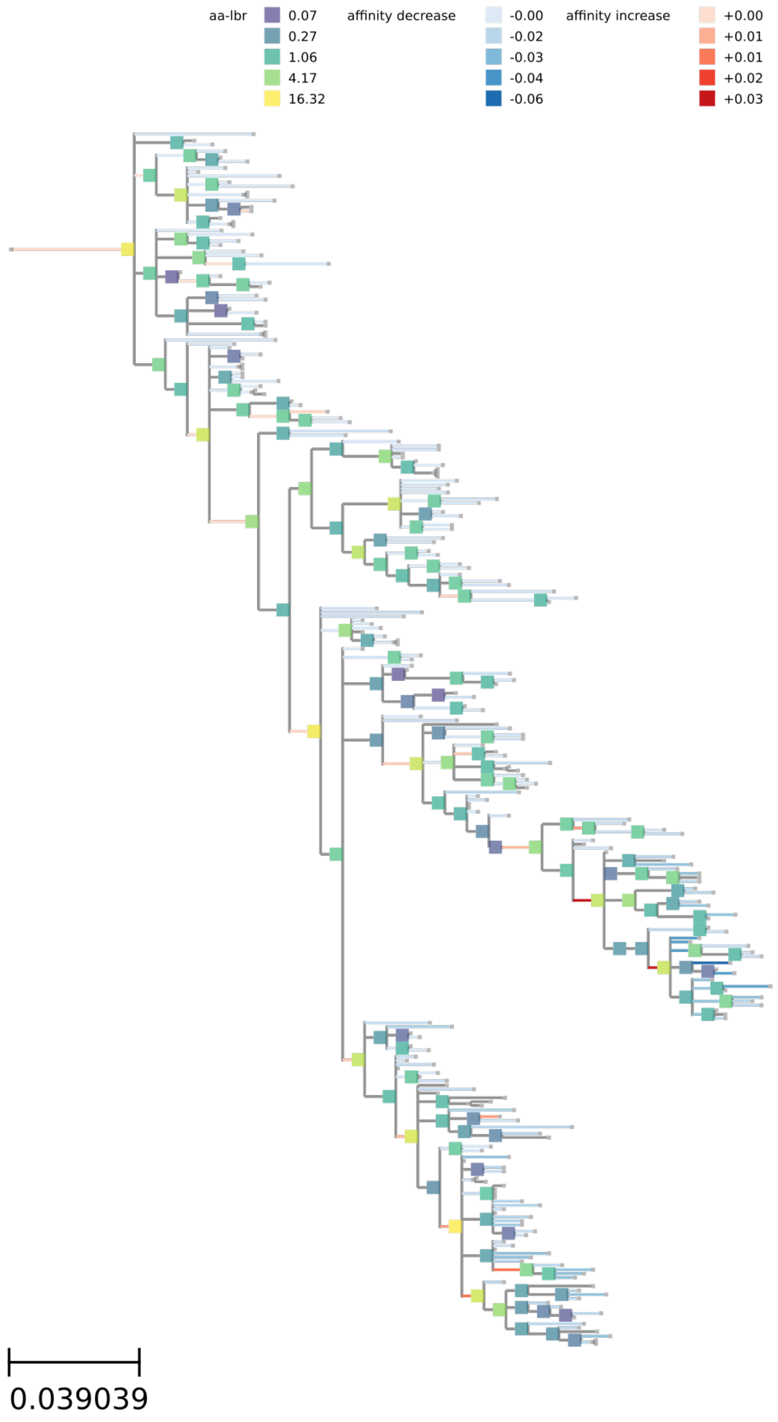




S3 Fig. **Example trees for nuc-lbr (top) and aa-lbr (bottom)** corresponding to the left hand plots in Fig 1.



S4 Fig. **Example tree for nuc-lbr** corresponding to the upper left plot in S1 Fig.



S5 Fig. **Example tree for aa-lbr** corresponding to the lower left plot in S1 Fig.