Supplement to "Adaptive RAxML-NG: Accelerating Phylogenetic Inference under Maximum Likelihood using dataset difficulty"

Anastasis Togkousidis [*,1], Alexey M. Kozlov [1], Julia Haag [1], Dimitri Höhler [1], Alexandros Stamatakis [3,1,2]

[1]Computational Molecular Evolution Group, Heidelberg Institute for Theoretical Studies, Germany

[2]Institute of Theoretical Informatics, Karlsruhe Institute of Technology, Karlsruhe, Germany

[3]Biodiversity Computing Group, Institute of Computer Science, Foundation for Research and Technology - Hellas

***Corresponding author:** E-mail: Anastasis.Togkousidis@h-its.org

1. Heuristic approach

The pseudocode of our adaptive RAxML-NG heuristic approach is summarized in Algorithm 1. The functions are the following by order of appearance:

- GetRandStartingTrees(): Function that takes as input the difficulty score of the MSA to be analyzed and returns the number of random starting trees to be used by adaptive RAxML-NG, based on Figure 2a available in the main text.
- GetParsStartingTrees(): Function that takes as input the difficulty score of the MSA to be analyzed and returns the number of MP starting trees to be used by adaptive RAxML-NG (see Figure 2a, main text).
- Optimize(): Core function of the algorithm, which takes as input a random/MP starting tree, the difficulty score of the corresponding MSA, and a numerical parameter ϵ. This function conducts a full ML tree search based on the adaptive heuristic described in Section 3 of the main text and returns the ML tree and its log-likelihood score. The numerical parameter ϵ is used as a log-likelihood improvement threshold, that is, the tree search is terminated when the attained log-likelihood improvement does not exceed this ϵ value. This default numerical convergence parameter is taken directly from RAxML-NG (Kozlov *et al.*, 2019) and was also used in standard RAxML (Stamatakis, 2014). The default parameter value in RAxML-NG v1.1 and standard RAxML is 0.1, while in the adaptive version and in v1.2 is 10. This modification on the default parameter value is based on the results of a recent comparative study (Haag *et al.*, 2022a), where the authors investigate the effect of different ϵ value settings on the ML tree inference accuracy. Users can set the ϵ value via the command line.
- BLO(): Branch-length optimization routine, that takes as input a binary tree and optimizes its branches.

Algorithm 1 Adaptive RAxML-NG heuristic

```
Input: MSA, difficulty, $\epsilon$
Output: MLtree, MLnLL
MLnL ← $-\infty$, MLtree ← NONE                          // Initialization

// Initialize random/MP trees and execute the heuristic
randTrees ← GetRandStartingTrees(difficulty)
parsTrees ← GetParsStartingTrees(difficulty)
for randTree in randTrees do   // Execute for each random tree
    ( tmpMLtree, LnL ) ← Optimize(randTree, difficulty,$\epsilon$)
    if LnL > MLnL then
        MLnL ← LnL, MLtree ← tmpMLtree
    end if
end for

for parsTree in parsTrees do          // Execute for each MP tree
    ( tmpMLtree, LnL ) ← Optimize(parsTree, difficulty, $\epsilon$)
    if LnL > MLnL then
        MLnL ← LnL, MLtree ← tmpMLtree
    end if
end for

function Optimize(tree, difficulty, $\epsilon$)
    LnL ← $-\infty$, impr ← TRUE                         // Initialization
    // Initial BLO, MPO
    ( tree, LnL ) ← Blo (tree)
    ( tree, LnL ) ← Mpo (tree)
    // Easy and difficult datasets start with an NNI Round
    if difficulty < 0.3 OR difficulty > 0.7 then
        ( tree, LnL ) ← NNI (tree)
        ( tree, LnL ) ← Mpo (tree)
        if Converged(LnL) then go to SECOND STAGE
        end if
    end if

    // First stage, Fast-SPR + NNI
    sprRad ← 5, step ← 5, rf ← $\infty$, maxRad ← 25
    while NOT Converged(LnL) AND rf != 0 AND impr do
        ( newTree, newLnL ) ← Fast-SPR (tree, sprRad)
        ( newTree, newLnL ) ← NNI (tree)
        impr ← ( newLnL − LnL > $\epsilon$ )                      // Boolean
        rf ← RFdist ( tree, newTree)
        tree ← newTree, LnL ← newLnL
        if sprRad < maxRad then
            sprRad ← sprRad + step
        end if
    end while

SECOND STAGE:                                    // SLOW-SPR + NNI
    ( tree, LnL ) ← Mpo (tree)                   // Intermediate MPO
    impr ← TRUE
    sprRad ← GetSlowSPRradius(difficulty)
    while impr do
        ( tree, newLnL ) ← Slow-SPR (tree, sprRad)
        ( tree, newLnL ) ← NNI (tree)
        impr ← ( newLnL − LnL > $\epsilon$ ), LnL ← newLnL
    end while
    ( tree, LnL ) ← Mpo (tree)                           // Final MPO
    return ( tree, LnL )                           // Return statement
end function

function Converged(LnL)
    if MLtree != NONE AND (MLnL−LnL)/|MLnL| < 0.01
then
        return TRUE
    end if
    return FALSE
end function

function GetSlowSPRradius(difficulty)
    if difficulty < 0.5 then
        return $\lfloor 50\cdot$ difficulty $+5 \rfloor$
    else
        return $\lfloor -50\cdot$ difficulty $+55 \rfloor$
    end if
end function

function GetRandStartingTrees(difficulty)
    numTrees ← $\lfloor 5.5\cdot$NormalDist(difficulty,0.5,0.2)$\rfloor$
    return min(numTrees, 10)        // MIN(): minimum function
end function

function GetParsStartingTrees(difficulty)
    numTrees ← $\lfloor 7.0\cdot$NormalDist(difficulty,0.5,0.25)$\rfloor$
    return min(numTrees, 10)        // MIN(): minimum function
end function

function NormalDist(x,m,s)
    inv_sqrt_2pi ← 0.39894
    a ← (x - m) / s
// EXP(): exponential function
    return inv_sqrt_2pi / (s · EXP($-0.5\cdot$a$\cdot$a))
end function
```

- MPO(): Model parameter optimization routine, that takes as input a binary tree and optimizes its model parameters.
- NNI(): NNI round routine, which takes as input a binary tree and applies a sequence of NNI moves, based on a greedy, hill-climbing heuristic. This function returns the NNI-optimal tree.
- Converged(): This function takes as input the current log-likelihood score of the best-scoring tree found so far during the tree search and returns TRUE if the current score is less than 1% worse than the reference score. This function will return FALSE otherwise. As a reference score, we use the log-likelihood score of the best ML tree found so far during all (preceding) finished tree searches.
- Fast-SPR(): Fast version of the SPR round, which takes as input a binary tree, applies a sequence of SPR moves based on a hill-climbing heuristic, and returns the optimized tree.
- Slow-SPR(): Slow version of the SPR round, which takes as input a binary tree, applies a

sequence of SPR moves based on a hill-climbing heuristic, and returns the optimized tree.

- RFDIST(): RF distance (Robinson and Foulds, 1981) function, which takes as input two binary trees and returns the RF distance between them.
- GETSLOWSPRRADIUS(): This function takes as input the difficulty score of the MSA being analyzed and returns the Slow-SPR radius parameter calculated by adaptive RAxML-NG (see Figure 2b, main text).

2. Difficulty score and Datasets

Figure 3 (available in the main text) shows the distribution of empirical and simulated datasets over 10 difficulty intervals. We can observe that the proportion of datasets with a difficulty score exceeding 0.9 is somewhat low compared to the rest of the datasets. In the main text, we briefly mention that this phenomenon is associated with the definition of the difficulty score *per se*. Further, in Section 2.2 in the main text, we provide a short description of the difficulty prediction paper (Haag *et al.*, 2022b). In this study members of our lab initially defined and calculated the difficulty score of empirical MSAs by conducting 100 ML tree searches on each dataset using RAxML-NG. They provided the following definition of the difficulty score:

$$\text{difficulty} = \frac{1}{5} \cdot (RF_{all} + RF_{pl} + \frac{N^*_{all}}{N_{all}} + \frac{N^*_{pl}}{N_{pl}} + (1 - \frac{N_{pl}}{N_{all}})) \quad (1)$$

The five terms included within the parentheses in Eq. S1 are the following in order of appearance:

- RF_{all}: The average relative RF distance between all pairs of trees in the 100 output ML trees.
- RF_{pl}: The average relative RF distance between all pairs of trees in the plausible tree set.
- $\frac{N^*_{all}}{N_{all}}$: The number of unique tree topologies in the output ML tree set (N^*_{all}) divided by the total number of trees in the same set (e.g., given that Haag *et al.* carried out 100 ML tree searches, this parameter would be $N_{all} = 100$).
- $\frac{N^*_{pl}}{N_{pl}}$: The number of unique tree topologies in the plausible tree set (N^*_{pl}) divided by the total number trees in the same set (N_{pl}).
- $\frac{N_{pl}}{N_{all}}$: The number of trees in the plausible tree set (N_{pl}) divided by the total number of trees in the output ML tree set ($N_{all} = 100$). We subtract this ratio from 1, leading to the full expression of the last term $(1 - \frac{N_{pl}}{N_{all}})$

Each term results in a value between 0.0 and 1.0. Since all terms are divided by 5, each of them can individually contribute up to 0.2 units to the overall difficulty score. The reason why only a small proportion of MSAs have a difficulty score exceeding 0.9 is mainly associated with the last term in the parentheses $(1 - \frac{N_{pl}}{N_{all}})$. On easy datasets, almost all independent ML tree searches yield trees with similar topologies and likelihood scores. Therefore, the vast majority of the final ML trees is included in the plausible tree set.

In this case, the ratio $\frac{N_{pl}}{N_{all}}$ is close to 1 and the term $(1-\frac{N_{pl}}{N_{all}})$ is close to 0. However, the majority of ML trees inferred on difficult MSAs are also plausible. While the topologies are highly incongruent, their likelihood scores are almost equal, and therefore the term $(1-\frac{N_{pl}}{N_{all}})$ is, again, close to 0. Hence, on difficult MSAs the last term contributes less than 0.1 units (out of 0.2), yielding an upper "bound" of 0.9 for the overall score which is rarely exceeded. The four remaining terms, however, contribute 0.2 units each, and thus, the problem is somewhat alleviated for difficult MSAs. This happens because, in such cases, both the RF distance terms and the unique-topology ratios are close to 1, implying a high number of distinct and incongruent topologies in the final ML tree set.

Overall, the difficulty score for "hopeless-to-analyze" MSAs is usually underestimated by the current definition (Eq. S1). Intuitively, the difficulty score of such datasets should be close to 1.0, but since the contribution of the last term in the parenthesis is low (for the reasons outlined above), the calculated value usually lies within the range of [0.8,0.9]. Indeed, only a small proportion of empirical MSAs in TreeBASE (Piel *et al.*, 2009) exhibit a difficulty score exceeding 0.9. On the other hand, due to the high contribution of the remaining four terms, the difficulty scores of "hopeless-to-analyze" datasets are rarely below 0.8. We believe that the definition of the difficulty score should be reformulated such that difficult MSAs are distributed more uniformly within the respective difficulty score range [0.7,1].

Regarding the datasets used in our experiments, in Section 4 of the main text we describe in detail the filtering process we followed to generate 9,515 empirical and 5,000 simulated MSAs in total. Out of the 9,515 empirical MSAs, 8,052 are unpartitioned alignments with DNA sequences (UP-DNA), 638 are partitioned alignments with DNA sequences (P-DNA), 817 are unpartitioned alignments with amino-acid sequences (UP-AA), and 8 are partitioned alignments with amino-acid sequences (P-AA). Further, from the 5,000 simulated MSAs, 4,482 are UP-DNA, 18 are P-DNA, 487 are UP-AA, and 13 are P-AA datasets. We subsampled the simulated DNA datasets from the datasets used in study by Höhler *et al.* (2022). We simulated the AA MSAs based on a sample of RAxML Grove (Höhler *et al.*, 2021) datasets. RAxML Grove datasets contain files with inferred trees, their respective estimated substitution model parameters, and statistical information about the analyzed MSA. In order to avoid simulating very large MSAs for the consecutive analyses, we selected datasets with an MSA number of unique sites (i.e., number of patterns) and number of taxa below the 95th percentile respectively. Then, we used the trees and the substitution models that were selected by the users of the RAxML web servers to estimate the model parameters we used to simulate MSAs with AliSim (Ly-Trong *et al.*, 2022). Overall, we used the following substitution models for AA MSA simulations: JTT+Γ (Jones *et al.*, 1992)(29.79%), LG+Γ (Le and Gascuel, 2008)(28.64%), Dayhoff+Γ (Dayhoff, 1972)(26.70%), WAG+Γ (Whelan and Goldman, 2001)(7.37%), Blosum62+Γ (Henikoff and Henikoff, 1992)(4%), MtArt+Γ (Abascal *et al.*, 2007)(2.71%), VT+Γ (Müller and Vingron, 2000)(2.34%), GTR+Γ(1.57%), MtREV+Γ (Adachi and Hasegawa, 1996)(1.22%), MtMAM+Γ (Yang *et al.*, 1998)(1.0%), rtREV+Γ (Dimmic *et al.*, 2002)(0.85%), MtZOA+Γ (Rota-Stabelli *et al.*, 2009)(0.65%), cpREV+Γ (Adachi *et al.*, 2000)(0.47%), PMB+Γ (Veerassamy *et al.*, 2003)(0.22%), HIVw+Γ (Nickle *et al.*, 2007)(0.1%), FLU+Γ (Dang *et al.*, 2010)(0.07%), DCMut+Γ (Kosiol and Goldman, 2005)(0.07%), HIVb+Γ (Nickle *et al.*, 2007)(0.07%).

For the analyses of all DNA MSAs, we used the GTR+Γ (Tavaré, 1986) model. In order to reduce the computational complexity and the CO_2 footprint when analyzing protein sequences, we used the LG model for the analysis of AA MSAs and did not account for rate heterogeneity with the Γ model (i.e., LG+Γ).

3. Commands

Users can invoke the adaptive RAxML-NG version using the `--adaptive` option when running the standard version of RAxML-NG. Our results can be reproduced by executing the following commands:

Standard version:

```
./raxml-ng --threads 1 --msa {msa}
--model {model} --seed 0
--extra compat-v11
```

Adaptive version:

```
./raxml-ng --adaptive --threads 1
--msa {msa} --model {model} --seed 0
--lh-epsilon 0.1 --lh-epsilon-triplet 0.1
```

We wish to emphasize that the adaptive version of RAxML-NG was designed and implemented based on RAxML-NG v1.1. Since then, a new version of RAxML-NG (i.e., v1.2) has been released. RAxML-NG v1.2 introduces modifications in the default values of log-likelihood improvement thresholds, for example the numerical convergence parameter ϵ discussed in Section S1 or the convergence threshold when optimizing the branch-lengths of the three adjacent branches around the insertion point during a Slow SPR move (see Section 2.1 in the main text)[1]. These changes in the default values have been incorporated into the adaptive version as well.

In our experiments we compared RAxML-NG v1.1 with adaptive RAxML-NG always using the default values from RAxML-NG v1.1 for the convergence parameters. RAxML-NG v1.1 can be invoked by adding the `--extra compat-v11` argument to the standard

[1] More information regarding RAxML-NG v1.2 can be found at: `https://github.com/amkozlov/raxml-ng/releases/tag/1.2.0`

RAxML-NG command line. Similarly, the adaptive version that uses the default values from RAxML-NG v1.1 for the convergence parameters can be invoked by adding the `--lh-epsilon 0.1 --lh-epsilon-triplet 0.1` arguments to the adaptive RAxML-NG command line. In case these arguments are omitted, the user will invoke standard/adaptive RAxML-NG with the updated default values for the convergence parameters.

4. IQ-TREE 2 significance tests

IQ-TREE 2 (Minh *et al.*, 2020) supports several statistical tests (Naser-Khdour *et al.*, 2019) using the RELL approximation (Kishino *et al.*, 1990) to determine if some trees in an input tree set are statistically significantly better than the remaining trees. We use these statistical tests to asses the ML trees inferred by the standard and adaptive versions of RAxML-NG. The implemented tests are: the bootstrap proportion[2], the Kishino-Hasegawa test (Kishino and Hasegawa, 1989) and the Shimodaira-Hasegawa test (Shimodaira and Hasegawa, 1999), both in their weighted and un-weighted variants, the Approximately Unbiased test (Shimodaira, 2002) , as well as the Expected Likelihood Weight test (Strimmer and Rambaut, 2002). We use the default IQ-TREE settings regarding the number of resampled estimated log-likelihood (RELL) replicates (10,000) as well as for the significance level ($\alpha = 0.05$).

5. Absolute log-likelihood differences

Figures S1 and S2 summarize the absolute log-likelihood differences (LD) for all standard-adaptive tree pairs, measured in log-likelihood units (LHU). We divide datasets into ten difficulty intervals. The heights of the bars correspond to the proportion of datasets, within the specified difficulty interval, where the LH difference lies within a specified LHU range. For example, in Figure S1 and on difficulty interval [0,0.1), there are 1,196 datasets (purple bar) where the absolute LH difference of the standard-adaptive pair is between 0 and 2 LHU. This range corresponds to approximately 72% of the empirical datasets that have a difficulty score within [0,0.1).

ACKNOWLEDGMENT

This study was financially supported by the Klaus Tschira Foundation and by the European Union (EU) under Grant Agreement No 101087081 (Comp-Biodiv-GR).

Funded by
the European Union

References

Abascal, F., Posada, D., and Zardoya, R. 2007. Mtart: a new model of amino acid replacement for arthropoda. *Molecular biology and evolution*, 24(1): 1–5.

[2] This method is also described in Kishino *et al.* (1990).

Adachi, J. and Hasegawa, M. 1996. Model of amino acid substitution in proteins encoded by mitochondrial dna. *Journal of molecular evolution*, 42: 459–468.

Adachi, J., Waddell, P. J., Martin, W., and Hasegawa, M. 2000. Plastid genome phylogeny and a model of amino acid substitution for proteins encoded by chloroplast dna. *Journal of molecular evolution*, 50: 348–358.

Dang, C. C., Le, Q. S., Gascuel, O., and Le, V. S. 2010. Flu, an amino acid substitution model for influenza proteins. *BMC evolutionary biology*, 10(1): 1–11.

Dayhoff, M. O. 1972. A model of evolutionary change in proteins. *Atlas of protein sequence and structure*, 5: 89–99.

Dimmic, M. W., Rest, J. S., Mindell, D. P., and Goldstein, R. A. 2002. rtrev: an amino acid substitution matrix for inference of retrovirus and reverse transcriptase phylogeny. *Journal of molecular evolution*, 55(1): 65.

Haag, J., Hübner, L., Kozlov, A. M., and Stamatakis, A. 2022a. The free lunch is not over yet – systematic exploration of numerical thresholds in phylogenetic inference. *bioRxiv*.

Haag, J., Höhler, D., Bettisworth, B., and Stamatakis, A. 2022b. From Easy to Hopeless—Predicting the Difficulty of Phylogenetic Analyses. *Molecular Biology and Evolution*, 39(12). msac254.

Henikoff, S. and Henikoff, J. G. 1992. Amino acid substitution matrices from protein blocks. *Proceedings of the National Academy of Sciences*, 89(22): 10915–10919.

Höhler, D., Haag, J., Kozlov, A. M., and Stamatakis, A. 2022. A representative performance assessment of maximum likelihood based phylogenetic inference tools. *bioRxiv*.

Höhler, D., Pfeiffer, W., Ioannidis, V., Stockinger, H., and Stamatakis, A. 2021. RAxML Grove: an empirical phylogenetic tree database. *Bioinformatics*, 38(6): 1741–1742.

Jones, D. T., Taylor, W. R., and Thornton, J. M. 1992. The rapid generation of mutation data matrices from protein sequences. *Bioinformatics*, 8(3): 275–282.

Kishino, H. and Hasegawa, M. 1989. Evaluation of the maximum likelihood estimate of the evolutionary tree topologies from dna sequence data, and the branching order in hominoidea. *Journal of molecular evolution*, 29: 170–179.

Kishino, H., Miyata, T., and Hasegawa, M. 1990. Maximum likelihood inference of protein phylogeny and the origin of chloroplasts. *Journal of Molecular Evolution*, 31: 151–160.

Kosiol, C. and Goldman, N. 2005. Different versions of the dayhoff rate matrix. *Molecular biology and evolution*, 22(2): 193–199.

Kozlov, A. M., Darriba, D., Flouri, T., Morel, B., and Stamatakis, A. 2019. RAxML-NG: a fast, scalable and user-friendly tool for maximum likelihood phylogenetic inference. *Bioinformatics*, 35(21): 4453–4455.

Le, S. Q. and Gascuel, O. 2008. An improved general amino acid replacement matrix. *Molecular biology and evolution*, 25(7): 1307–1320.

Ly-Trong, N., Naser-Khdour, S., Lanfear, R., and Minh, B. Q. 2022. AliSim: A Fast and Versatile Phylogenetic Sequence Simulator for the Genomic Era. *Molecular Biology and Evolution*, 39(5). msac092.

Minh, B. Q., Schmidt, H. A., Chernomor, O., Schrempf, D., Woodhams, M. D., von Haeseler, A., and Lanfear, R. 2020. IQ-TREE 2: New Models and Efficient Methods for Phylogenetic Inference in the Genomic Era. *Molecular Biology and Evolution*, 37(5): 1530–1534.

Müller, T. and Vingron, M. 2000. Modeling amino acid replacement. *Journal of Computational Biology*, 7(6): 761–776.

Naser-Khdour, S., Minh, B. Q., Zhang, W., Stone, E. A., and Lanfear, R. 2019. The Prevalence and Impact of Model Violations in Phylogenetic Analysis. *Genome Biology and Evolution*, 11(12): 3341–3352.

Nickle, D. C., Heath, L., Jensen, M. A., Gilbert, P. B., Mullins, J. I., and Kosakovsky Pond, S. L. 2007. Hiv-specific probabilistic models of protein evolution. *PloS* *one*, 2(6): e503.

Piel, W. H., Chan, L., Dominus, M. J., Ruan, J., Vos, R. A., and Tannen, V. 2009. TreeBASE v. 2: A Database of Phylogenetic Knowledge. *e-BioSphere 2009*.

Robinson, D. and Foulds, L. 1981. Comparison of phylogenetic trees. *Mathematical Biosciences*, 53(1): 131–147.

Rota-Stabelli, O., Yang, Z., and Telford, M. J. 2009. Mtzoa: a general mitochondrial amino acid substitutions model for animal evolutionary studies. *Molecular phylogenetics and evolution*, 52(1): 268–272.

Shimodaira, H. 2002. An approximately unbiased test of phylogenetic tree selection. *Systematic biology*, 51(3): 492–508.

Shimodaira, H. and Hasegawa, M. 1999. Multiple comparisons of log-likelihoods with applications to phylogenetic inference. *Molecular biology and evolution*, 16(8): 1114.

Stamatakis, A. 2014. RAxML version 8: a tool for phylogenetic analysis and post-analysis of large phylogenies. *Bioinformatics*, 30(9): 1312–1313.

Strimmer, K. and Rambaut, A. 2002. Inferring confidence sets of possibly misspecified gene trees. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 269(1487): 137–142.

Tavaré, S. 1986. Some probabilistic and statistical problems in the analysis of dna sequences. *Lect Math Life Sci (Am Math Soc)*, 17: 57–86.

Veerassamy, S., Smith, A., and Tillier, E. R. 2003. A transition probability model for amino acid substitutions from blocks. *Journal of Computational Biology*, 10(6): 997–1010.

Whelan, S. and Goldman, N. 2001. A general empirical model of protein evolution derived from multiple protein families using a maximum-likelihood approach. *Molecular biology and evolution*, 18(5): 691–699.

Yang, Z., Nielsen, R., and Hasegawa, M. 1998. Models of amino acid substitution and applications to mitochondrial protein evolution. *Molecular biology and evolution*, 15(12): 1600–1611.

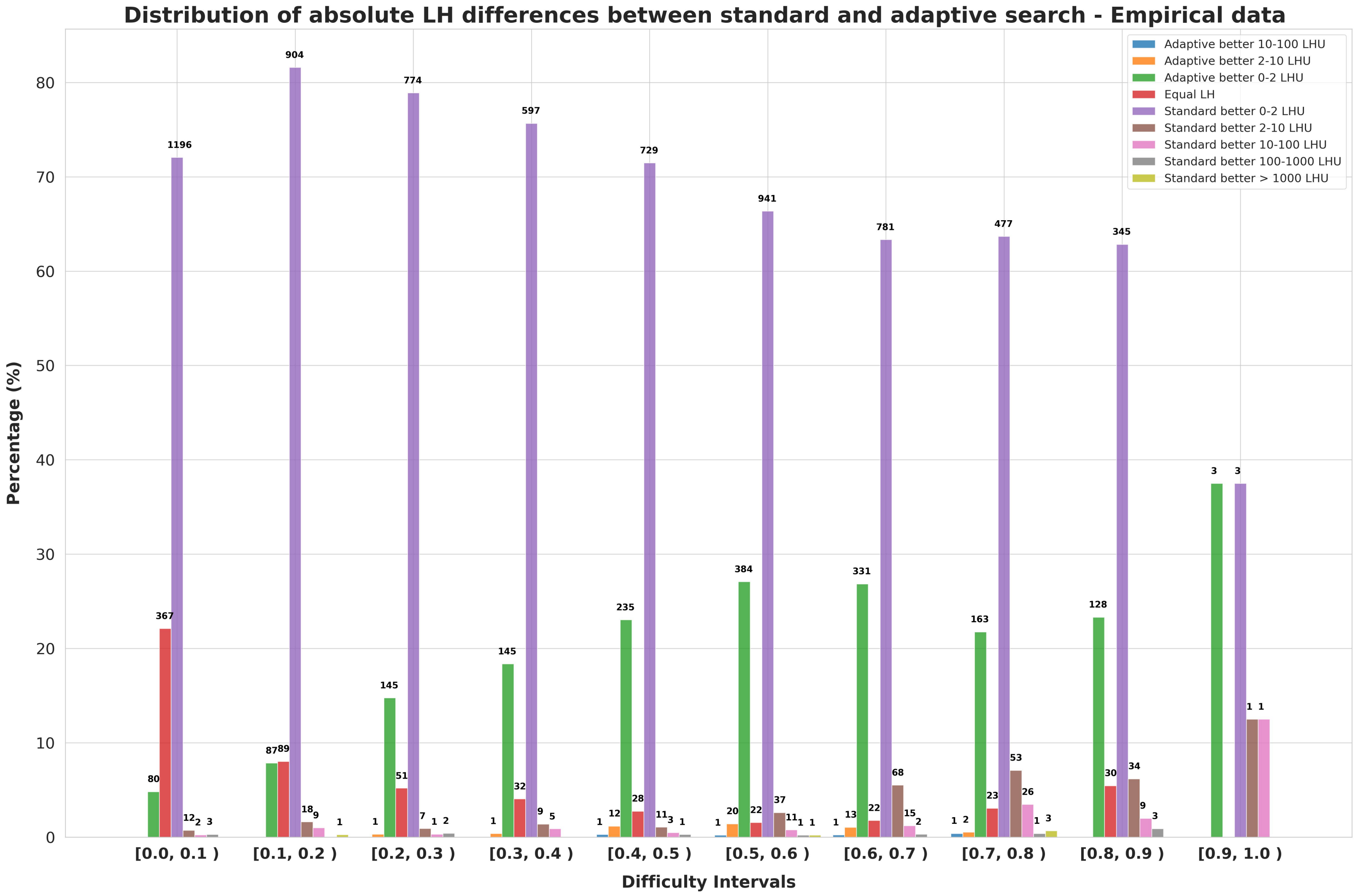

FIG. 1. Distributions of absolute log-likelihood differences in all standard-adaptive tree pairs, on empirical data. The abolute LH differences are measured in log-likelihood units (LHU). We divide the MSAs into ten difficulty intervals. The height of the bars corresponds to the proportion of datasets, within the specified difficulty interval, in which the score of standard and adaptive trees have an absolute difference within a range of LHU. The numbers at the top of the bars correspond to the number of datasets in this LHU range.

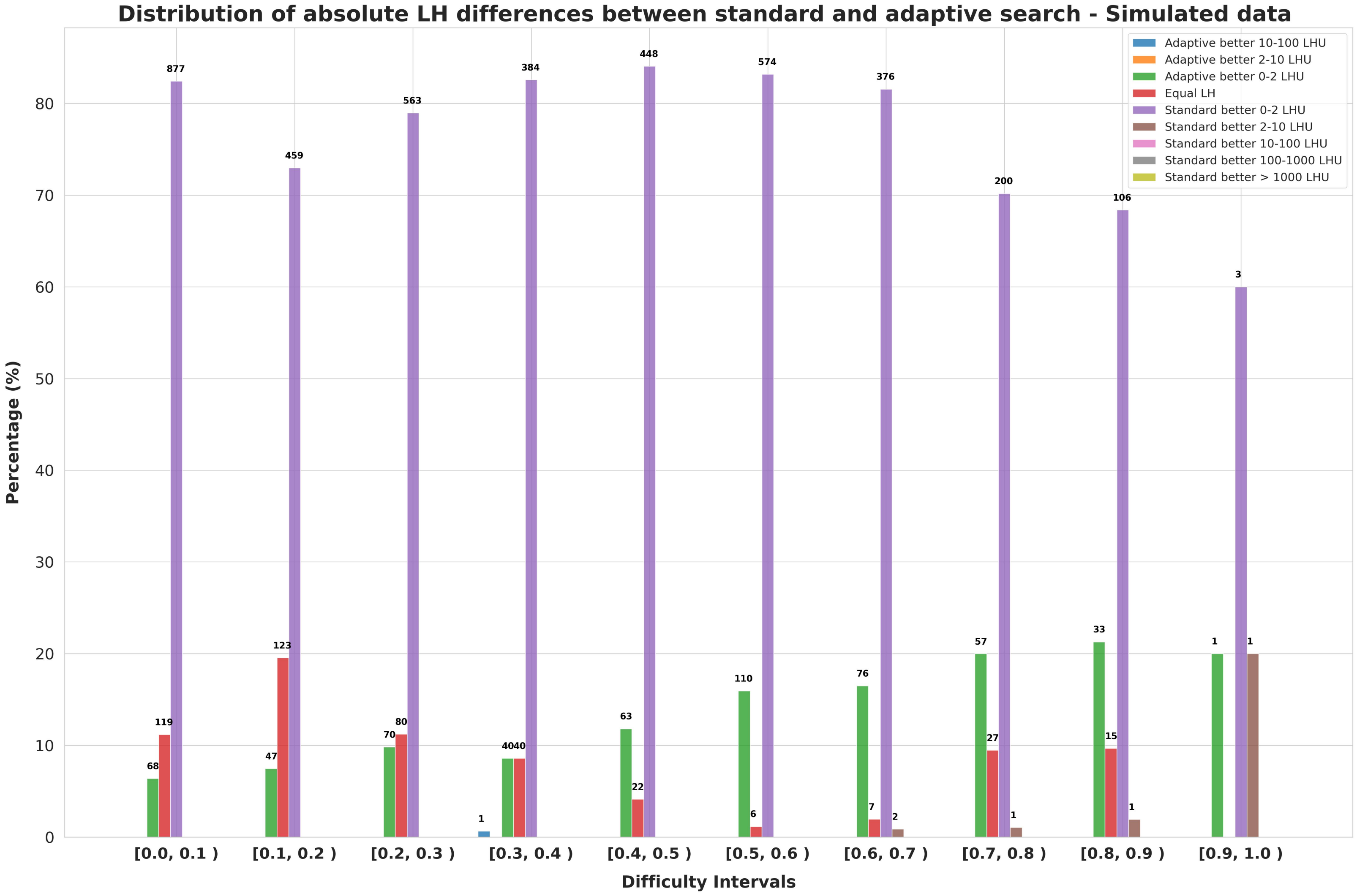

FIG. 2. Distributions of absolute log-likelihood differences in all standard-adaptive tree pairs, on simulated data. The abolute LH differences are measured in log-likelihood units (LHU). We divide the MSAs into ten difficulty intervals. The height of the bars corresponds to the proportion of datasets, within the specified difficulty interval, in which the score of standard and adaptive trees have an absolute difference within a range of LHU. The numbers at the top of the bars correspond to the number of datasets in this LHU range.