# FastME 2.0: a comprehensive, accurate, and fast distance-based phylogeny inference program

Vincent Lefort, Richard Desper and Olivier Gascuel*
Institut de Biologie Computationnelle
LIRMM, UMR 5506: CNRS & Université Montpellier 2, FRANCE

* Corresponding author: gascuel@lirmm.fr

---

## Supplementary Material

---



---

## SPR tree searching

Subtree Pruning and Regrafting (SPR) tree searching in FastME 2.0 uses formulae and algorithms that were presented in (Desper and Gascuel 2002; Hordijk and Gascuel 2005). They are described here again for purposes of clarity. Further details, explanations, and proof can be found in these papers. Let us begin with notation and definitions.

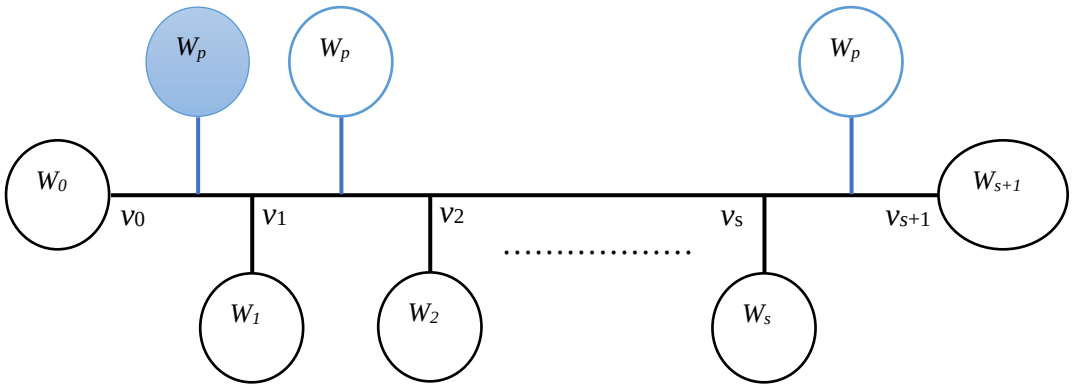
We consider a tree topology *T*, typically an initial topology to be improved using SPRs. The subtrees of *T* are denoted by capital letters: *A*, *B*, *W* … etc. Taxa and tree nodes are denoted with small letters: *a*, *b*, *v* … etc.

We also consider pairwise evolutionary distances among taxa, and average distances between subtrees. $\Delta_{ab}$ is the distance between taxa *a* and *b*; $\Delta_{AB}$ is the average distance between (non-intersecting) subtrees *A* and *B*. SPR tree searching in FastME is based on the balanced minimum evolution (BME) principle and Pauplin's (2001) tree length formula. In this framework, the average distance between subtrees is defined recursively. Assuming that *B* is composed of two subtrees *B'* and *B"*, then:

$$\Delta_{AB} = \frac{1}{2}\left(\Delta_{AB'} + \Delta_{AB''}\right).$$

If *A* and *B* each contains a single taxon *a* and *b*, respectively, then $\Delta_{AB} = \Delta_{ab}$. Given *T* and the $\Delta$ matrix of pairwise distances among taxa, the average distances between all pairs of non-intersecting subtrees in *T* is computed in $O(n^2)$ time (*n* is the number of taxa), using a tree traversal algorithm described in (Desper and Gascuel 2002). Computing all these average distances is the first step in SPR tree searching.

Moreover, these average distances can be updated (at least those being required to apply tree length formulae) all along the search for the best improving SPR. Consider the figure below, where $W_p$ is the pruned subtree, and we search for the best insertion branch for $W_p$ along the path from nodes $v_1$ to $v_{s+1}$.



Inserting $W_p$ between $v_1$ and $v_2$ corresponds to a Nearest Neighbor Interchange (NNI). The algorithm performs successive NNIs to compute recursively, with increasing distance *s*, the tree length changes when $W_p$ is inserted on every tree branch. Throughout this procedure, we compute the following average distances recursively:

$$\Delta^*_{v_s W_p} = \frac{1}{2}\left(\Delta^*_{v_{s-1} W_p} + \Delta_{W_s W_p}\right),$$

where $\Delta^*_{v_s W_p}$ is the average distance between $W_p$ (when inserted between $v_s$ and $v_{s+1}$) and the non-intersecting subtree rooted at $v_s$ (e.g. $W_0 \cup W_1$ when $W_p$ is inserted between $v_1$ and $v_2$); $\Delta^*_{v_0 W_p} = \Delta_{W_0 W_p}$ initializes the recursion. Assuming again that $W_p$ is inserted between $v_s$ and $v_{s+1}$, we also need:

$$\Delta^*_{v_s W_{s+1}} = \Delta_{v_s W_{s+1}} - \left(\frac{1}{2}\right)^{s+1} \Delta_{W_p W_{s+1}} + \left(\frac{1}{2}\right)^{s+1} \Delta_{W_0 W_{s+1}},$$

where $\Delta^*_{v_s W_{s+1}}$ is the updated average distance between $W_{s+1}$ and the non-intersecting subtree rooted at $v_s$, after $W_p$ has been pruned from its original position.

Using these updated average distances, we are able to compute the tree length change recursively when moving (by NNI) $W_p$ from branch $(v_{s-1}, v_s)$ to $(v_s, v_{s+1})$, that is:

$$dL_s = dL_{s-1} + \frac{1}{4}\left[\left(\Delta^*_{v_{s-1} W_p} + \Delta_{W_{s+1} W_s}\right) - \left(\Delta^*_{v_{s-1} W_s} + \Delta_{W_{s+1} W_p}\right)\right],$$

where $dL_s$ is the tree length difference between the new topology with $W_p$ inserted on $(v_s, v_{s+1})$, and the initial topology with $W_p$ between $v_0$ and $v_1$. Having $dL_0 = 0$ for every pruned subtree, we are able to compute the best SPR corresponding to the smallest $dL_s$ value among all subtrees and insertion positions. As we have $O(n^2)$ possibilities, and all above equations are computed in constant time, finding the best SPR is achieved in $O(n^2)$. When this best SPR improves the current tree (i.e. $dL_s < 0$), the procedure is iterated: we achieve this SPR, re-compute the average distance between all subtree pairs, and search for the best SPR, etc. Note that the branch lengths are never estimated in this algorithm. This is done for the final tree only, when no more SPR improvement is found, using an $O(n^2)$ algorithm

described in (Desper and Gascuel 2002). Altogether the time complexity of SPR tree searching is thus $O(kn^2)$, where $k$ is the number of iterations. In our experiments (see below) we always observed $k < n$ (typically between a few units and $n/3$).

## Algorithm comparison with real data

Here we compare standard algorithms and FastME using real data. Several studies with simulated data have shown the advantage of using NNIs in combination with BME (e.g. Desper and Gascuel 2002, 2004; Vinh et al. 2005). With simulated data, the true tree is known. We are thus able to compare the topological accuracy of algorithms, and FastME was shown to be substantially more accurate than NJ (among others). However, simulated data are often considered "too easy" and many authors recommend using real data. Then, the true topology is unknown, and we must rely on other criteria and approaches. We use here minimum evolution (BME version) and maximum likelihood criteria. Minimum evolution measures the fit of the inferred tree using its total length (i.e. the sum of the lengths of its branches). This criterion is minimized (the shorter the inferred tree, the better), and shares with maximum parsimony the general principle that simple explanations are preferable to complex ones. Minimum evolution forms the basis of a large number of distance based algorithms: NJ and those implemented in FastME and MEGA (Tamura et al. 2011), but also FastTree1 (Price et al. 2009), for example. We also use the likelihood of the tree topology inferred from the input alignment, as in many studies, typically comparing ML-based algorithms (e.g. Guindon et al. 2010). The higher the likelihood, the better the phylogenetic tree and inference algorithm. In the following, we first describe the features of our data sets, then the algorithms being compared, and lastly their results regarding minimum evolution and maximum likelihood criteria.

### Data sets

Large, public data sets were extracted from:

- ***Flu*** (Bao et al. 2008), http://www.ncbi.nlm.nih.gov/genomes/FLU/FLU.html

We used type B alignments, which are both large and well aligned, with “Full-length only” and “Collapse identical sequences” options. The 3 largest protein and DNA data sets were selected.

- ***Rbcl*** (Stamatakis et al. 2010), http://www.exelixis-lab.org/resource/download/rbcl/

  We used the 4 DNA data sets available.

- ***PhyML 3.0 benchmark*** (Guindon et al. 2010),

  http://www.atgc-montpellier.fr/phyml/benchmarks/index.php?ben=lg

  We used the largest DNA and protein data sets available in this benchmark (extracted from TreeBase, Sanderson et al. 1994), to leverage a total of 10 DNA and 10 protein data sets.

| Identifier | Protein vs. DNA | Origin | Number of taxa | Number of sites |
|---|---|---|---|---|
| *B-NA-684-472* | Protein | *flu* | 684 | 463 |
| *B-HA-573-585* | Protein | *flu* | 573 | 584 |
| *B-NS1-284-344* | Protein | *flu* | 284 | 281 |
| *proteic_M2624_139x348_2006* | Protein | *PhyML* | 104 | 337 |
| proteic_M3497_105x899_2007 | Protein | *PhyML* | 105 | 899 |
| *proteic_M2883_91x7386_2007* | Protein | *PhyML* | 80 | 7299 |
| proteic_M3756_77x11234_2008 | Protein | *PhyML* | 77 | 11234 |
| *proteic_M3755_77x9918_2008* | Protein | *PhyML* | 24 | 9588 |
| proteic_M3068_50x1000_2006 | Protein | *PhyML* | 50 | 1000 |
| proteic_M2577_40x12260_2005 | Protein | *PhyML* | 40 | 12260 |
| *eudicots* | DNA | *rbcL* | 3748 | 1370 |
| rosids | DNA | *rbcL* | 2445 | 1371 |
| euros1 | DNA | *rbcL* | 1718 | 1371 |
| *B-HA-986-1908* | DNA | *flu* | 986 | 1755 |
| *B-NA-633-1751* | DNA | *flu* | 633 | 1425 |
| *B-NS-629-1111* | DNA | *flu* | 629 | 1032 |
| euros2 | DNA | *rbcL* | 479 | 1371 |
| nucleic_M2839_470x829_2006 | DNA | *PhyML* | 470 | 829 |
| nucleic_M3862_362x1207_2008 | DNA | *PhyML* | 362 | 1207 |
| nucleic_M2573_346x897_2006 | DNA | *PhyML* | 346 | 897 |

Some of these alignments contain a large number of gaps. We used BMGE (Criscuolo et al. 2010) with default options to remove the gappy sites from *flu* alignments. BMGE was also used to remove the sequences having more than 50% of gaps in some of the *rbcl* and *PhyML* alignments. The main features of these data sets are summarized in the table above, where identifiers in italics correspond to alignments cleaned using BMGE. All these data are available from FastME web site: http://www.atgc-montpellier.fr/fastme/.

**Algorithms being compared**

Using these data sets, we compare: **NJ** (Saitou and Nei 1987); **BioNJ** (Gascuel 1997); **NJ+NNI**, where the initial NJ tree is improved with FastME NNIs; **NJ+SPR**, where the initial NJ tree is improved with FastME SPRs; **NJ+OLSME**, where the initial NJ tree is improved with MEGA by close neighbor interchanges (CNIs), optimizing the ordinary least-squares version of minimum evolution (OLSME, Rzhetsky and Nei 1993; MEGA options: "construct minimum evolution tree" and "ME Search Level = 2"); **FastTree1** (Price et al. 2009), which searches for minimum evolution trees, but uses profiles instead of a distance matrix. In that respect, FastTree1 is in-between distance and character methods, as it reconstructs the sequence profiles of ancestral nodes. Moreover, FastTree does not compute all pairwise distances, which constitutes the computational bottleneck of standard distance methods. We also ran **NINJA** (Wheeler 2009) and **STC** (Vinh et al. 2005). NINJA is a very fast implementation of NJ with simplified distance estimation (compared to FastME, DNADIST and PROTDIST from PHYLIP). As expected, we observed that NINJA obtained results very close to NJ's, but much faster. However, current NINJA implementation does not allow for alignments with >10,000 sites (proteic-M3756-77x1's1234-2008 and proteic-M2577-40x12260-2005 in our benchmark) and its results are not shown below. STC was very fast too and its performance was similar to NJ's, except with the 4 *rbcl* data sets where inferred trees were just inconsistent; again its results are not shown.

NJ, BioNJ, NJ+NNI and NJ+SPR are run with FastME, including distance estimation using TN93 for DNA and JTT for proteins, with a continuous gamma distribution of rates

across sites of parameter 1.0. NJ+OLSME is run with MEGA, including distance estimation, using the same models as FastME. FastTree1 uses simple evolutionary models (JC69 for DNA, and log-corrected with BLOSUM45 for proteins) and does not allow for a gamma distribution of rates across sites. As the advantage of using rates across sites with distance-based approaches is questionable (Guindon and Gascuel 2002), we also check the performance of **NJ+SPR-Γ**, where the distance matrix is estimated without gamma distribution, and tree building is achieved by NJ+SPR.

Computing times in seconds, with two Intel(R) Xeon(R) CPUs X5650 2.67GHz, are displayed in the table below. For comparison purpose, we also provide the computing times of DNADIST and PROTDIST from PHYLIP to estimate the distances matrices using F84, JTT and a continuous gamma distribution of rates across sites. We see that distance calculation by FastME is much faster than PHYLIP's, but MEGA is even faster with DNA. This is a part of the FastME code which could be improved, for example using vectorization as in FastDist (Elias and Lagergren 2007). Our implementation of tree building algorithms could likely be improved too, but all FastME algorithms are still fast, including Dist+NJ+SPR, which is the only method to achieve SPRs. FastTree1 is remarkably fast with the larger data sets.

| | | DNA | | | Protein | |
|---|---|---|---|---|---|---|
| | | *3 Flu data sets* | *3 PhyML data sets* | *4 rbcL data sets* | *3 Flu data sets* | *7 PhyML data sets* |
| PHYLYP Distance estimation | | 1,819 | 537 | 29,805 | 15,265 | 2,520 |
| FastME Distance estimation | | 75 | 9 | 831 | 345 | 42 |
| FastTree1 | | 78 | 30 | 380 | 45 | 154 |
| MEGA NJ+OLSME | | 16 | 4 | 527 | 519 | 67 |
| FastME | Dist+NJ | 91 | 11 | 1,881 | 350 | 43 |
| | Dist+NJ+NNI | 93 | 11 | 1,935 | 352 | 43 |
| | Dist+NJ+SPR | 259 | 24 | 6,549 | 400 | 43 |

**Length of inferred tree topologies**

To estimate the length of the tree topologies inferred by any of the methods being compared, we used FastME with BME option, fixed topology, branch length optimization, and TN93/JTT models with continuous gamma distribution of rates across sites of parameter 1.0. Results are reported in the following table. All methods are compared to NJ and NJ+SPR by counting the number of data sets where one method is better than the other, and calculating the average relative difference in tree length.

| | | **Ref. NJ** | | | **Ref. NJ+SPR** | | |
|---|---|---|---|---|---|---|---|
| | | **+** | **-** | **‰ tree length** | **+** | **-** | **‰ tree length** |
| **NJ** | DNA | | | | **0** | **10** | -13 |
| | Protein | | | | **0** | **9** | -9 |
| **NJ+OLSME** | DNA | **0** | **10** | -18 | **0** | **10** | -31 |
| | Protein | 2 | 3 | +0 | **0** | **9** | -9 |
| **BioNJ** | DNA | 7 | 3 | +6 | **0** | **10** | -7 |
| | Protein | 2 | 8 | -3 | **0** | **10** | -12 |
| **NJ+NNI** | DNA | **10** | **0** | +11 | **0** | **10** | -2 |
| | Protein | **9** | **0** | +7 | **0** | **7** | -2 |
| **NJ+SPR** | DNA | **10** | **0** | +13 | | | |
| | Protein | **9** | **0** | +9 | | | |
| **NJ+SPR-Γ** | DNA | 8 | 2 | -2 | **0** | **10** | -15 |
| | Protein | 7 | 3 | +8 | 2 | 8 | -1 |
| **FastTree1** | DNA | **9** | **1** | +0 | **0** | **10** | -13 |
| | Protein | 5 | 5 | +2 | **1** | **9** | -7 |

Note: In this table, we report pairwise comparisons between methods, with two references: NJ and NJ+SPR. All methods are compared to both references. For example: BioNJ is compared to NJ, and then NJ+SPR; "+" is the number of times where BioNJ is better (the BioNJ tree length is shorter than the reference), "-" is the number of times where BioNJ is worse, these numbers are bold, underlined when their difference is significant (p-value<5% using a sign test); "‰ tree length" is the average per mille relative difference in tree length of reference minus BioNJ; BioNJ is better than NJ with 7 DNA alignments, worse with 3, and its tree length is 6 ‰ shorter than NJ's; when both algorithms share $x$ topologies in common, "+" and "-" cells sum to $10 - x$ (e.g. $x$=3 with NJ+NNI, NJ+SPR and protein data).

We see from this table that NJ performs well compared to NJ+OLSME, BioNJ and FastTree1, especially with protein data. This is an expected outcome as NJ optimizes the BME version of minimum evolution (Gascuel and Steel 2006), which we use here as comparison criterion. However, as expected again, NJ and all methods (including NJ+SPR-Γ) are clearly beaten by NJ+NNI and NJ+SPR, which further improve the NJ tree using topological moves, with tree length gains of ~10 ‰ or more. Moreover, NJ+SPR finds better tree topologies than NJ+NNI, but the gains in tree length are low (~2 ‰). These results show that our SPR and NNI algorithms achieve their goal and find trees that are substantially shorter than those found by other methods. In the next section, we measure whether this gain in tree length is associated with a gain in tree likelihood.

**Likelihood of inferred tree topologies**

To compute the likelihood of the inferred tree topologies by any of the methods being compared, we use PhyML with GTR+Γ4 for DNA sequences, and JTT+Γ4 for proteins. The model parameters and branch lengths are optimized by PhyML, but the tree topology is fixed. Results are displayed in the table below. All methods are compared to NJ and NJ+SPR using the same criteria as with tree length criterion.

We see from this table that NJ+OLSME does not significantly improve NJ. BioNJ does better than NJ with DNA sequences, but similarly with proteins. NJ is significantly bested by NJ+NNI with DNA sequences, and by NJ+SPR with both DNA sequences and proteins; moreover, the likelihood gains are substantial, especially with DNA (~10 ‰). FastTree1 does slightly better than NJ+SPR with DNA sequences and slightly worse with proteins, while NJ+SPR-Γ (same as NJ+SPR but not using any rates across sites model, just as FastTree1) is best in these experiments (in accordance with Guindon and Gascuel 2001), both with DNA and protein sequences, but with a small margin compared to NJ+SPR.

Globally, we see from these experiments that tree length and likelihood criteria are congruent, with the slight exception of FastTree1 which performs very well regarding

likelihood with DNA, but not so with tree length. This could be explained by the fact that FastTree1 is partly a character method and thus closer to the likelihood approach than the other pure distance-based methods.

| | | Ref. NJ | | | Ref. NJ+SPR | | |
|---|---|---|---|---|---|---|---|
| | | + | - | ‰ log-lik. | + | - | ‰ log-lik. |
| **NJ** | DNA | | | | **0** | **10** | -11 |
| | Protein | | | | **1** | **8** | -5 |
| **NJ+OLSME** | DNA | 6 | 4 | +2 | **0** | **10** | -9 |
| | Protein | 3 | 2 | +0 | **1** | **8** | -5 |
| **BioNJ** | DNA | **9** | **1** | +7 | **1** | **9** | -4 |
| | Protein | 6 | 4 | +1 | 3 | 7 | -4 |
| **NJ+NNI** | DNA | **9** | **1** | +9 | **0** | **10** | -1 |
| | Protein | 6 | 3 | +4 | 2 | 5 | -1 |
| **NJ+SPR** | DNA | **10** | **0** | +11 | | | |
| | Protein | **8** | **1** | +5 | | | |
| **NJ+SPR-Γ** | DNA | **10** | **0** | +13 | 7 | 3 | +2 |
| | Protein | **10** | **0** | +5 | 6 | 4 | +0 |
| **FastTree1** | DNA | **10** | **0** | +12 | 8 | 2 | +1 |
| | Protein | 7 | 3 | +1 | 4 | 6 | -4 |

Note: In this table, we report pairwise comparisons between methods, with two references: NJ and NJ+SPR. All methods are compared to both references. For example: BioNJ is first compared to NJ, and then NJ+SPR; "+" is the number of times where BioNJ is better, "-" is the number of times where BioNJ is worse, these numbers are bold, underlined when their difference is significant (p-value<5% using a sign test); "‰ log-lk dif." is the average, relative difference in per mille of the log-likelihood of BioNJ minus the reference; BioNJ is better than NJ with 9 DNA alignments, worse once, and the average relative gain in log-likelihood is 9 ‰; when both algorithms share $x$ topologies in common, "+" and "-" cells sum to $10 - x$ (e.g. $x$=3 with NJ+NNI, NJ+SPR and protein data).

**Conclusion:**

These experiments with real data indicate that the SPR tree searching algorithm implemented in FastME brings a substantial improvement compared to NJ, its variants and

our original NNI algorithm, both in terms of tree length and likelihood. Moreover, the computing time is still quite low. For example, with the largest *rbcl* DNA data set (3,748 taxa and 1,370 sites), it requires about one hour and half on a desktop computer to compute the distance matrix (10 minutes), build an initial NJ tree (15 minutes), and improve this tree with SPRs (60 minutes).

## References:


Bao Y., Bolotov P., Dernovoy D., Kiryutin B., Zaslavsky L., Tatusova T., Ostell J. and Lipman D. (2008). The Influenza Virus Resource at the National Center for Biotechnology Information. *J. Virol.* 82(2):596-601.

Criscuolo A., Gribaldo S. 2010. BMGE (Block Mapping and Gathering with Entropy): a new software for selection of phylogenetic informative regions from multiple sequence alignments. *BMC Evol. Biol.* 10:210.

Desper, R. and Gascuel, O. (2002). Fast and accurate phylogeny reconstruction algorithms based on the minimum-evolution principle. *J. Comp. Biol.* 9:687–705.

Desper, R. and Gascuel, O. (2004). Theoretical foundations of the balanced minimum evolution method of phylogenetic inference and its relationship to weighted least-squares tree fitting. *Mol. Biol. Evol.* 21:587-598.

Elias, I. and J. Lagergren (2007). Fast Computation of Distance Estimators. *BMC Bioinformatics* 8:89.

Gascuel, O. and Steel, M. (2006). Neighbor-Joining Revealed. *Mol. Biol. Evol.* 23:1997–2000.

Guindon, S. and O. Gascuel (2002). Efficient biased estimation of evolutionary distances when substitution rates vary across sites. *Mol. Biol. Evol.* 19(4):534-543.

Guindon S., Dufayard J.F., Lefort V., Anisimova M., Hordijk W., Gascuel O. (2010). New Algorithms and Methods to Estimate Maximum-Likelihood Phylogenies: Assessing the Performance of PhyML 3.0. *Syst. Biol.* 59(3):307-21.

Hordijk, W., and Gascuel, O. (2005). Improving the efficiency of SPR moves in phylogenetic tree search methods based on maximum likelihood. *Bioinformatics* 21:4338-4347.

Pauplin, Y. (2000). Direct calculation of a tree length using a distance matrix. *J. Mol. Evol.* 51:41–47.

Price, M. N., Dehal, P. S., and A. P. Arkin (2009) FastTree: Computing Large Minimum Evolution Trees with Profiles instead of a Distance Matrix. *Mol. Biol. Evol.* 26: 1641-1650.

Rzhetsky, A., and M. Nei. (1993). Theoretical foundation of the minimum-evolution method of phylogenetic inference. *Mol. Biol. Evol.* 10:1073-1095.

Saitou, N. and Nei, M. (1987) The neighbor-joining method: A new method for reconstruction of phylogenetic trees. *Mol. Biol. Evol.* 4:406-425.

Sanderson M.J., Donoghue M.J., Piel W. and Eriksson T. (1994). TreeBASE: a prototype database of phylogenetic analyses and an interactive tool for browsing the phylogeny of life. *Amer. Jour. Bot.* 81:183.

Stamatakis A., Göker M., and G.W. Grimm (2010). Maximum Likelihood Analyses of 3,490 rbcL Sequences: Scalability of Comprehensive Inference versus Group-Specific Taxon Sampling. *Evol. Bioinf. Online* 6:73-90.

Tamura, K., Peterson, D., Peterson, N., Stecher, G., Nei, M. and S. Kumar (2011). MEGA5: Molecular Evolutionary Genetics Analysis Using Maximum Likelihood, Evolutionary Distance, and Maximum Parsimony Methods *Mol. Biol. Evol.* 2731-2739.

Vinh, L. S., and A. von Haeseler (2005). Shortest triplet clustering: reconstructing large phylogenies using representative sets. *BMC Bioinf.* 6:92.

Wheeler, T.J. (2009). Large-scale neighbor-joining with NINJA. In S.L. Salzberg and T. Warnow (Eds.), Proceedings of the 9th Workshop on Algorithms in Bioinformatics (WABI), Lecture Notes in Computer Science (Springer, Berlin) 5724: 375-389.