TRACTION:
Fast non-parametric improvement of estimated gene trees

S. Christensen, E. Molloy, P. Vachaspati, T. Warnow
Gene Tree Correction

Short sequences give inaccurate gene trees!
- 25% average bootstrap support on genes in avian phylogenomics project

Can we make them better?
Not without more information.

Solution: use information from other genes

TRACTION: Use estimated species tree to correct gene trees

(Note: we aren’t talking about multi-copy genes or duplication/loss models here)
Correction workflow

- Estimated Gene Tree
- Collapse low-support branches
- Reference tree
- Corrected and Completed Gene Tree
RF distances

The Robinson-Foulds (RF) distance between two trees is equal to the number of bipartitions that occur in one tree, but not in the other.
Restricting trees to a taxon subset

A tree $T$ on taxon set $S$ can be restricted to taxon set $R \subseteq S$, represented by $T|_R$. 
Refining trees

Polytomy
Compatible bipartitions

A C D E (compatible)
A B C D (compatible)
A C B E (incompatible)
Robinson-Foulds Optimal Tree Refinement & Completion (RF-O SRC)

Inputs: Binary unrooted tree $T$ with taxon set $S$
Unrooted tree $G$ with taxon set $R \subseteq S$

TRACTION completes and refines $G$ to minimize the RF distance to $T$

Output: Binary tree $G^*$ such that:
1. $G^*$ contains all the taxa in $S$
2. $G^{|R}$ is a refinement of $G$
3. $G^*$ minimizes the RF distance to $T$
Two phases for TRACTION

PHASE 1: RF-Optimal Tree Refinement
- New algorithm presented here

PHASE 2: RF-Optimal Tree Completion
- OCTAL algorithm
  (Christensen et al., WABI 2017)
  \(O(|S|^2)\)
- Bansal’s algorithm
  (Bansal, RECOMB-CG 2018):
  \(O(|S|^{1.5\log(|S|)})\)
Two steps for refinement

**INPUT:** Gene tree $G$ on taxon set $R$,  
Collapsed tree $G_{\text{collapsed}}$  
Species tree $T$ restricted to taxon set $R$  

**OUTPUT:** Fully resolved tree $G_{\text{refined}}$ minimizing RF distance to $T$

**Step 1:** Add compatible bipartitions from $T$ to $G_{\text{collapsed}}$  

**Step 2:** Refine remaining polytomies
Refinement example

Input trees

Gene tree $G$

Reference tree $T$
Step 1: Add compatible bipartitions from T

Shared bipartitions:
ABGH | CDEF
ABCFGH | DE, etc.

Compatible bipartitions in T:
ABGHC | DEF

Incompatible bipartitions in T:
AB | CDEFGH
GH | ABCDEF
Step 1: Add compatible bipartitions from $T$

Shared bipartitions:
ABGH | CDEF
ABCFGH | DE, etc.

Compatible bipartitions in $T$:
ABGHC | DEF

Incompatible bipartitions in $T$:
AB | CDEFGH
GH | ABCDEF
Step 2: Refine arbitrarily

Shared bipartitions:
ABGH | CDEF
ABCFGH | DE, etc.

Compatible bipartitions in $T$:
ABGHC | DEF

Incompatible bipartitions in $T$:
AB | CDEFGH
GH | ABCDEF
Completion - only if $G$ is on taxon subset

**INPUT:** Fully resolved gene tree $G_{\text{refined}}$ on taxon set $R \subseteq S$
Species tree $T$ on taxon set $S$

**OUTPUT:** Fully resolved gene tree $G^*$ on taxon set $S$ minimizing RF distance to $T$

Solved by OCTAL (Christensen et al., WABI 2017),
Bansal’s algorithm (Bansal et al., RECOMB-CG 2018)
Robinson-Foulds Optimal Tree Refinement & Completion (RF-OTRC)

Inputs: Binary unrooted tree $T$ with taxon set $S$
Unrooted tree $G$ with taxon set $R \subseteq S$

$\text{TRACTION}$ completes and refines $G$ to minimize the RF distance to $T$

Output: Binary tree $G^*$ such that:

1. $G^*$ contains all the taxa in $S$
2. $G^*|_R$ is a refinement of $G$
3. $G^*$ minimizes the RF distance to $T$
Sketch of correctness proof

Theorem: Traction solves RF-OTRC(G, T) exactly in O(n^{1.5} \log n) time

1. The intermediate Traction tree \( G_{\text{refined}} \) solves RF-OTR(G, T|R)
2. Traction returns the completed OCTAL tree, which solves RF-OTC(G_{\text{refined}}, T)
3. RF-OTC(G_{\text{refined}}, T) = RF-OTRC(G, T)
Asymptotic running time $O(n^{1.5} \log(n))$

After preprocessing step, check bipartition-tree compatibility in $O(n^{0.5} \log(n))$ time*

Determine compatible bipartitions between $G$ and $T$ in $O(n^{1.5} \log(n))$ time

OCTAL takes $O(n^2)$ time; Bansal’s algorithm takes $O(n^{1.5} \log(n))$ time

Total asymptotic running time is
  - $O(n^2)$ when using OCTAL
  - $O(n^{1.5} \log(n))$ when using Bansal’s algorithm

* Gawrychowski et al., 2017
Comparison methods

NOTUNG (Chen et al., 2000)

ProfileNJ (Noutahi et al., 2016)

TreeFix (Wu et al., 2012) - for ILS dataset

TreeFix-DTL (Bansal et al., 2015) - for ILS+HGT dataset

ddleTERA (Jacox et al., 2017)

Most of these methods designed for gene duplication and loss - not being tested here

Evaluation criterion: RF distance between corrected gene tree and true gene tree
Experimental evaluation (on complete gene trees)

- **ILS-only**
  - 26 species
  - 2 levels of ILS
  - 8000 genes total (20 replicates per model condition with 200 genes each)

- **ILS+HGT**
  - 51 species
  - 2 levels of HGT, 1 level of ILS
  - 3 gene sequence lengths
  - 60,000 genes total (50 replicates per model condition; 200 genes each)

Gene trees estimated with RAxML; reference species trees with ASTRID
ILS+HGT dataset: very accurate gene trees
ILS+HGT dataset: moderately accurate gene trees
ILS+HGT dataset: highly inaccurate gene trees

![Box plot showing RF Error for different methods and species per tree categories.](image-url)
Empirical running time results

Total time (in seconds) for each method to correct 50 gene trees with 51 species on one replicate of the HGT+ILS dataset with moderate HGT

<table>
<thead>
<tr>
<th>Method</th>
<th>Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>EcceTERA</td>
<td>1470</td>
</tr>
<tr>
<td>NOTUNG</td>
<td>43</td>
</tr>
<tr>
<td>TRACTION</td>
<td>30</td>
</tr>
<tr>
<td>ProfileNJ</td>
<td>87</td>
</tr>
<tr>
<td>TreeFix-DTL</td>
<td>188</td>
</tr>
</tbody>
</table>
Summary of experimental results

ILS-only: TRACTION, TreeFix, NOTUNG best performing methods

ILS+HGT: TRACTION gives improvement only when GTEE is high

TRACTION performs as well or better than competing methods

TRACTION is faster than competing methods

NOTUNG and TRACTION are generally the best performing methods

Some methods (particularly ecceTERA and ProfileNJ) fail to complete on some inputs
Acknowledgements

Co-authors: Sarah Christensen, Erin Molloy, Tandy Warnow

Funding: Ira & Debra Cohen Fellowship (SC, EM); NSF Graduate Research Fellowship Grant Number DGE-1144245 (EM, PV), NSF CCF-1535977 (TW)

This study was performed on the Illinois Campus Cluster and Blue Waters, a computing resource that is operated and financially supported by UIUC in conjunction with the National Center for Supercomputing Applications.
Refinement Step 1: Add edges compatible with $T$
Refinement Step 2:
Add edges compatible with $G$
Refinement Step 3: Randomly resolve everything else

In this case, we don’t have anything left after refining edges based on compatibility with T and G

Then, use OCTAL or Bansal’s algorithm to complete tree
TRACTION produces an RF-optimal refinement

Let $T$ be a binary tree on $R$, and let $G$ be a tree on $R$.

**Theorem:** $RF(T, G_{\text{refined}})$ is minimized iff $G_{\text{refined}}$ includes all compatible bipartitions from $T$

$$RF(G_{k'}, T) = RF(G, T) - |X| + |Y|,$$

$|X| = \# \text{ compatible bipartitions added}$

$|Y| = \# \text{ incompatible bipartitions added}$

This is minimized iff every compatible bipartition is added to $G$
If $G$ and $T$ are on the same taxon set, we are done!

Theorem: $RF(T, G_{\text{refined}})$ is minimized iff $G_{\text{refined}}$ includes all compatible bipartitions from $T$

TRACTION adds every compatible bipartition from $T$ to $G$, therefore $RF(T, G_{\text{refined}})$ is minimized
Optimal completion

OCTAL completes trees optimally

An optimal completion increases the RF distance by $2m$, where $m$ is the number of type-2 superleaves in $T$
TRACTION minimizes the number of type 2 superleaves

When we refine:
- Type 1 superleaves stay type 1
- A type 2 superleaf becomes type 1 if we add its edge to $G$

Every compatible bipartition in $T$ is added to $G$, so every type 2 superleaf that can be converted to a type 1 superleaf is converted
TRACTION solves RF-OTRC(G, T) exactly

1. The intermediate TRACTION tree $G_{\text{refined}}$ solves RF-OTR(G, T|_R)

2. TRACTION returns the completed OCTAL tree, which solves RF-OTC($G_{\text{refined'}}$, T)

3. RF-OTC($G_{\text{refined'}}$, T) = RF-OTRC(G, T)
   - $G_{\text{refined}}$ minimizes the number of Type II superleaves in T