	L_∞		L_1		ME		R_∞		R_1	
	NJ	SP-DP	NJ	SP-DP	NJ	SP-DP	NJ	SP-DP	NJ	SP-DP
Same tree	35%	35%	35%	32.5%	60%	57.5%	35%	32.5%	37.5%	35%
Closer to the matrix	7.5%	15%	40%	42.5%	20%	40%	20%	30%	45%	50%
Farther from the matrix	**2.5%**	**0%**	**15%**	**15%**	**12.5%**	**2.5%**	**5%**	**2.5%**	**17.5%**	**15%**
Same distance	55%	50%	10%	10%	7.5%	0%	40%	35%	0%	0%

Table 2: Comparison between the solutions found by NJ and SP-DP and the trees from which the random data has been generated. The values are a percentage of instances.

	Biological Data					Random Data				
	L_∞	L_1	L_∞^+	L_1^+	ME	L_∞	L_1	L_∞^+	L_1^+	ME
SP-DP was better	17%	9%	6%	8%	1%	3%	2%	4%	2%	–
NJ was better	–	1%	–	1%	1%	–	1%	–	–	1%

Table 3: Average percentage of improvement that each method achieved.

	L_∞	L_∞^+	L_1	L_1^+	ME	R_∞	R_∞^+	R_1	R_1^+
Biological Data	5%	5%	30%	35%	25%	20%	25%	25%	25%
Random Data	2.5%	7.5%	30%	25%	20%	0%	5%	22.5%	17.5%

Table 4: Comparison between SP and DP. Values are the percentage of the instances for which DP was better than SP alone.

	Worst	*Best*	*Mean*
Biological Data	2.12	1.42	1.61
Random Data	2.45	1.19	1.66

Table 5: Lower bounds obtained by SP. The theoretical worst case is three.

	L_∞	L_∞^+	L_1	L_1^+	ME
SP-DP	0%	0%	18%	17%	7%
NJ	8.3%	8.3%	8.3%	10%	2%

Table 1: Percentage of instances for which NNI improved the best solutions.

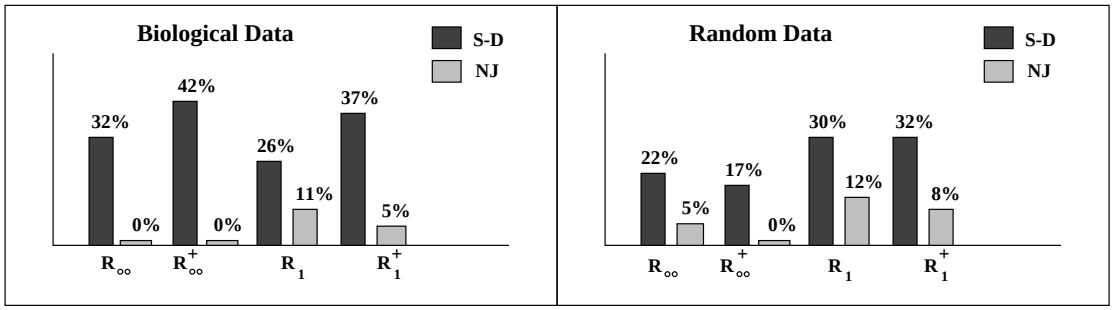

Figure 4.5: Comparative results under the relative norm distances.

edly, NNI does not change the relative performance of the heuristics.

References

[1] R. Agarwala, V. Bafna, M. Farach, B. Narayanan, M. Paterson, and M. Thorup. On the approximability of numerical taxonomy: Fitting distances by tree metrics. *Proc. of the 7th Annual ACM-SIAM Symposium on Discrete Algorithms*, 1996.

[2] J-P. Barthélemy and A. Guénoche. *Trees and Proximity Representations*. Wiley, New York, 1991.

[3] A. Caccone, G. D. Amato, and J. R. Powell. Rates and Patterns of scnDNA and divergence within the Drosophila melanogaster subgroup. *Genetics*, 118:671–683, 1988.

[4] A. Caccone and J. R. Powell. Molecular evolutionary divergence among North American cave crickets. II DNA-DNA hybridization. *Evolution*, 41:1215–1238, 1987.

[5] L. Cavalli-Sforza and A. Edwards. Phylogenetic analysis models and estimation procedures. *Amer. J. Human Genetics*, 19:233–257, 1967.

[6] H. Chang, D. Wang, and F. J. Ayala. Mitochondrial DNA variation in the Drosophila nasuta subgroup of species. *J. Mol. Evol.*, 28:337–348, 1989.

[7] W.H.E. Day. Computational complexity of inferring phylogenies from dissimilarity matrices. *Bulletin of Mathematical Biology*, 49(4):461–467, 1987.

[8] M. Farach and S. Ferenci. An evaluation of the local structure of neighbor joining. Manuscript, 1996.

[9] M. Farach, S. Kannan, and T. Warnow. A robust model for finding optimal evolutionary trees. *Algorithmica*, 13:155–179, 1993.

[10] D. Goldman, P. R. Giri, and S. J. O'Brien. Molecular genetic distance estimates among the Ursidae as indicated by one- and two- dimensional protein electrophoresis. *Evolution*, 43:282–295, 1989.

[11] R. Highton and A. Larson. The genetic relationships of the salamanders of the genus Plethodon. *Syst. Zool.*, 28:579–599, 1979.

[12] C. Krajewski. Relative rates of single-copy DNA evolution in cranes. *Mol. Biol. Evol.*, 7:65–73, 1990.

[13] M. George Jr. and O. A. Ryder. Mitochondrial DNA variation in the genus Equus. *Mol. Biol. Evol.*, 3:535–546, 1986.

[14] N. Saitou and M. Nei. The neighbor-joining method: a new method for reconstructing phylogentic trees. *Mol. Biol. Evol.*, 4:406–424, 1987.

[15] J. B. Slowinski and C. Gruyer. Testing the stochasticity of patterns of organismal diversity: an improved null model. *Am. Nat.*, 134:907–921, 1989.

[16] P. H. A. Sneath and R. R. Sokal. *Numerical Taxonomy*. W. H. Freeman, San Francisco, California, 1973.

[17] D. L. Swofford and G. J. Olsen. Phylogeny reconstruction. In D. M. Hillis and C. Moritz, editors, *Molecular Systematics*, pages 411–501. Sinauer Associates Inc., Sunderland, MA., 1990.

[18] K. J. Sytsma and L. D. Gottlieb. Chloroplast DNA evolution and the phylogenetic relationships in Clarikia sect. peripetasma. *Evolution*, 40:1248–1262, 1986.

[19] H. T. Wareham. On the computational complexity of inferring evolutionary trees. Master's thesis, Department of Computer Science, Memorial University of Newfoundland, May 1993. Technical Report no. 9301.

[20] M.S. Waterman, T.F. Smith, M. Singh, and W.A. Beyer. Additive evolutionary trees. *J. Theor. Biol.*, 64:199–213, 1977.

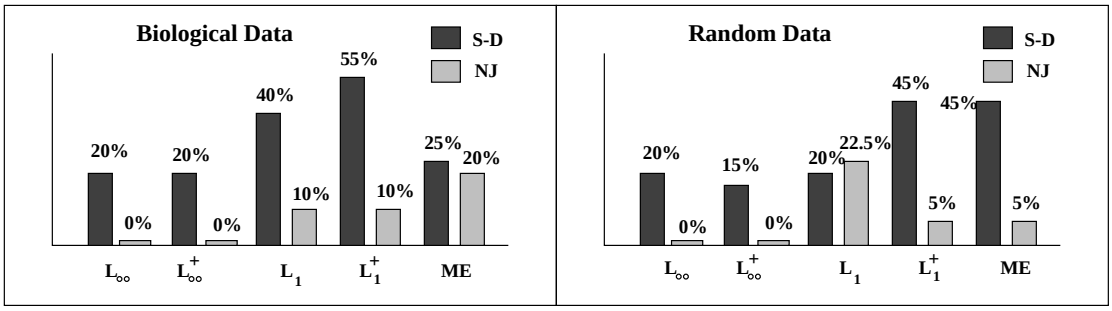

Figure 4.3: Comparison between SP/DP with LP and NJ. The values are the percentage of instances in which each algorithm found a better solution.

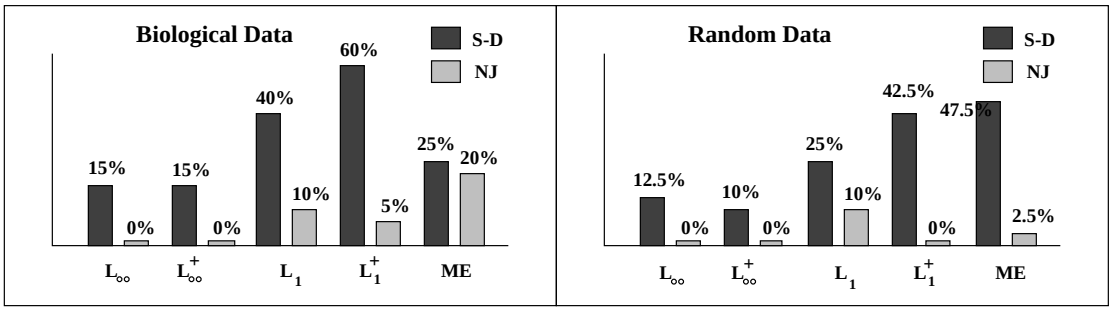

Figure 4.4: Results after the NNI heuristic has been applied.

Lower bounds Another practical aspect of the SP algorithm that we studied was the lower bounds for the optimal solutions under L_∞. As explained before, the value of a solution given by SP divided by three is a lower bound for the optimal solution. By comparing the best lower bound with the best solution, we can define an upper bound on how far the solution is from the optimal. Table 10 summarizes these results. The values represent the best solution divided by one third of the worst solution.

Running times Data collection is the bottleneck of the application. However, to complete the comparison between SP-DP and NJ, we give below a summary of the running times of our experiments.

The running times were measured with the programs running on a Sun4-Sparc workstation and correspond to CPU time. Without LP, the NJ heuristic ran in less than one second while the SP/DP without LP ran in less than 3 seconds for all instances.

When the LP optimization was applied the running times varied for each criterion being used. For L_∞, NJ still ran in less than one second while the SP/DP heuristic had its running time increased to at most 7 seconds for instances of size 10 and at most 65 seconds for instances of size 15.

For L_1, the running times increased, so that NJ ran in less than 3 seconds and SP-DP ran in less than 38 seconds and 4 minutes and 20 seconds for instances of sizes 10 and 15 respectively.

Comparing with Optimal Solutions Except for very small instances, there is little hope that provably optimal solutions can be found without huge computational expenditures. The reason is not just that the problems are NP-hard but also that exhaustive search methods have to solve expensive LP problems many times during the search. We ran a branch and bound algorithm on our set of matrices for the L_∞ norm. Optimal solutions were found for some of the matrices of size ten or less. In all such cases, SP/DP found solutions of optimal value. For the other matrices and the other criteria we do not have knowledge of the optimal solutions.

5 Conclusions

We have presented an extension of the SP heuristic for Numerical Taxonomy. As expected from theoretical analysis, the SP/DP combination performs very well on L_∞ norms [1]. Somewhat more surprisingly, SP/DP beats NJ under other measures as well. The use of LP is crucial to obtaining optimal performance from SP/DP, and while this makes SP/DP somewhat slow, data gathering will clearly dominate computational time for any reasonable Numerical Taxonomy problem. Unexpect-

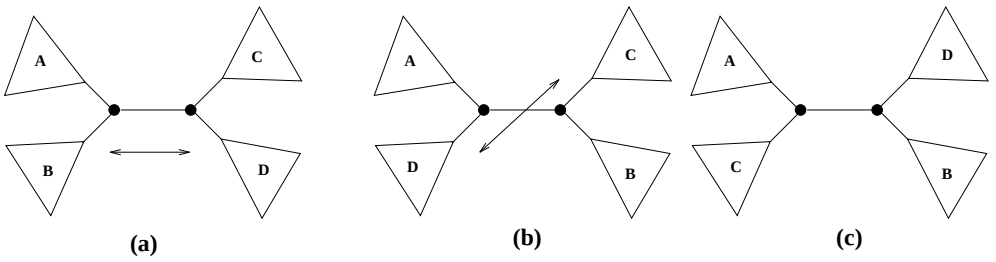

Figure 3.1: Illustration for the Nearest Neighbor Interchange (NNI). (a) The original tree. (b) and (c) The trees obtained by the NNI operation.

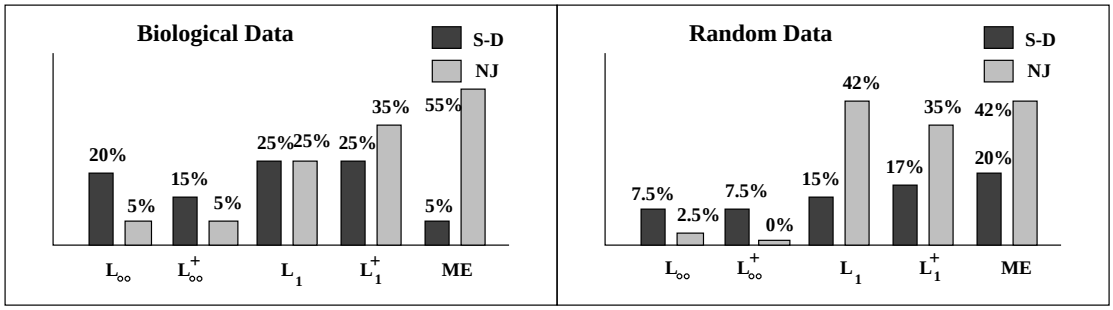

Figure 4.2: Comparison between NJ and SP/DP without LP. The values are the percentage of instances in which each algorithm found a better solution.

indicated that after a threshold of a standard deviation of fifteen, the higher its value the better NJ performs compared to SD-DP, still under L_1. It should be noted that these standard deviations are very high.

As discussed in Section 3, any tree produced by SP/DP is close to the optimal, under L_∞, by a factor of three even before the LP. This fact suggested that SP/DP could be used without the LP procedure. The results presented here revealed that in practice LP may be an important tool for selecting a good pivot.

Heuristics with NNI We applied the NNI local search technique to the solutions given by both methods. The results are shown in figure A.9. In general, the results are not very different from the original ones except for the L_1 on random data, in which the SP/DP solutions were improved for 22.5% of the instances, and the relative performance against NJ changed considerably. Table 6 shows how often NNI improved the final solution of each method. Note that for L_∞ and L_∞^+, NNI did not improve any solution given by SP/DP. In this experiments, LP was used for the choice of the best pivot.

Relative Norms For the relative norm distances the performance of the SP/DP heuristic was even better. The results without NNI and SP/DP running with LP can be seen in figure A.10. After NNI was applied, there was no significant changes in the results.

Retrieving the original tree A natural question to ask when working with randomly generated data is how often the heuristics retrieved the original tree. The solutions found can be grouped in four cases:

1. The tree is the same
2. The tree is closer to the matrix than the original tree
3. The tree is farther from the matrix
4. The tree is different but is at the same distance.

Cases 2 and 4 imply that in the search space induced by the particular criterion, the original tree is not optimal or uniquely optimal, respectively.

Case 3 constitutes a true error on the part of the heuristic. SP/DP did quite well in this regard for L_∞, R_∞ and *ME*. The results for five criteria are shown in table 7.

Quality of the solutions Table 8 shows the average percentage of improvement that each heuristic achieved when its solution was better than the solution found by the other. This results are based on the SP-DP with LP and NNI versus NJ with LP and NNI.

SP vs. DP We compared the performance of SP vs. DP. Table 9 shows how often DP found a better solution than SP alone. These results are based on the experiments with LP and without NNI.

we want to find the tree $T = (V, E, W)$ which minimizes some criterion $\Phi_M(T)$. For any particular topology $S = (V, E)$, let $W = \phi_M(S)$ be the edge weighing function that minimizes $\Phi_M((V, E, W))$. So, for example, if $\Phi_M(T) = L_1(T, M)$, then for any topology S, $\phi_M(S)$ gives the L_1 best fit of S to M.

Now, suppose we want to find the T minimizing $\Phi_M(T)$. Further suppose that we have some candidate S. Consider $W = \phi_M(S)$. If W associates a negative length to an edge e, then it almost certainly the case that the topology around e is incorrect [8].

We can modify S by one of two possible *Nearest Neighbor Interchanges (NNI)*, to yield S_1 or S_2 (See Figure A.6). We then check to see if S_1 or S_2 yields a better answer. This NNI based local optimization is part of standard practice (See Phylip manual). However, the use of negative edges under ϕ_M optimization was only recently introduced [8]. We will therefore only consider tree optimizations Φ_M for which solving ϕ_M is polynomially computable, in particular by linear programming. These include $L_\infty, L_1, L_\infty^+, L_1^+, R_\infty, R_1$ and Minimum Evolution.

4 Experimental Results

4.1 The data We consider the performance of NJ and SP/DP on both biological and randomly generated data. The biological data consist of twenty distance matrices ranging in size from six to twenty taxa. These matrices can be found in [3, 4, 6, 10, 11, 12, 13, 15, 18].

Two types of random data were generated. In both, we start with a random tree and produce its pairwise distance matrix. Then a noise factor is inserted into every entry. In the first type, we insert both positive and negative noise and in the second, only negative. This latter case is meaningful because in some cases, measured biological distances will be an underestimate of true distance. The noise was generated by a normal distribution with mean zero and different standard deviations for each of four different groups of matrices. These are five, ten, fifteen and twenty. When such a number is generated for an entry, this is taken to be the percent change of the value in that entry. Matrices having ten and fifteen taxa were generated. A total of forty matrices of each type were used, five for each combination of standard deviation and size. The matrices of the first type were used in the experiments for the L_∞, L_1, R_∞ and R_1 criteria and the ones of the second type for L_∞^+, L_1^+, R_∞^+ and R_1^+ and *ME*.

4.2 Experiments

How the trees were compared Since it is the topology itself which is usually the most meaningful part of the output in a Numerical Taxonomy problem, trees should be compared on the basis of their optimal edge lengths. The trees produced by the heuristics, both ours and NJ, do not necessarily have optimal lengths for any criterion. Therefore these lengths must be computed by solving the correspondent LP. However, these LP instances have $n \cdot (n-1)/2$ constraints and from $2 \cdot n - 3$ up to $2 \cdot n - 3 + n \cdot (n-1)/2$ variables. Therefore the time needed to solve these problems cannot be neglected. We will, however, always run LP on both SP/DP trees and NJ trees to compare the topologies of the trees. For each criterion, a different LP was used, so that trees were always tuned for the particular comparison at hand.

NJ always produces a single tree, and therefore there is no ambiguity as to how and when to apply LP. LP can be used in another way with SP/DP. Notice that SP/DP can produce $\binom{n}{2}$ trees, if all possible pivots are chosen, thus raising the issue of how to select a pivot pair. This choice can be made in two ways: by computing the optimal edge lengths for every tree or by using the original edge lengths produced by the algorithm. The first method is likely to produce a better solution while the second is much faster. We ran SP/DP using both alternatives and results are shown below.

Heuristics with no local optimization We first compared the tree produced by NJ and SP/DP on the data. No local optimization (NNI) was applied. In these experiments SP/DP tried all choices of pivots, but the best tree was selected without running LP for each tree.

For each criteria analyzed, the comparative performance between NJ and SP-DP had significant variations. Figure A.7 show the results for all absolute distance criteria. The values shown are the percentage of instances for which each algorithm found a better solution (the complement of their sum being the percentage of instances for which solutions with same values were found).

From our current theoretical knowledge, it was expected that SP/DP would perform well under L_∞ and this was confirmed by the experiments. Also under L_∞^+ the SP-DP found better solutions. For the other criteria NJ had the better performance.

Heuristics with LP As before, we report the relative performance of NJ vs. SP/DP, this time SP/DP chose the best tree by first computing the optimal edge lengths of the trees produced by each choice of pivot. As shown in the Figure A.8, the SP/DP method sometimes found better solutions than NJ, and the opposite never occured. The L_∞^+ criterion gave very similar results than the ones obtained for L_∞.

For the L_1 case, the SP/DP outperformed NJ for the biological data but they have similar performance for the random data. However, about 55% of the cases in which NJ found better solutions than SP-DP occured for the matrices having standard deviation of twenty, the farthest from being additive. Further experiments

tree T into an ultrametric on the leaves of T, that is, we want to root T at some point and lengthen the edges incident on the leaves, which we will henceforth refer to as *stems*, until they are equidistant from the root. Notice more generally that we can take any internal node from the set S to be a leaf connected to T by an edge of length 0, and so we can include internal nodes of T in our ultrametric. Thus this procedure will convert an arbitrary tree metric into an ultrametric. We need only specify how we pick the root of the ultrametric.

In [1], an arbitrary $a \in S$ was chosen as the root. There they showed that the following mapping converts T into an ultrametric rooted at a. Let $m_a = \max_i\{D[a,i]\}$. Let C^a be the centroid metric with $l_i = m_a - D[a,i]$, i.e., $C^a[i,j] = l_i + l_j = 2m_a - D[a,i] - D[a,j]$. In this case, we say that a is the *pivot*. Notice that C^a is a matrix defined only in terms of T and a, and so it is well defined whether T is a tree metric or not. The following lemma relates the structure of T and $T + C^a$.

LEMMA 3.1. ([2, TH.3.2]) *For all a, D is quasi-additive if and only if $D + C^a$ is ultrametric.*

Thus, if D is not a tree metric, $D + C^a$ is not an ultramtric. The insight in [1] is that $D + C^a$ can be mapped to its closest ultrametric via the algorithm in [9]. Let $U(D + C^a)$ be the ultrametric so derived. Then we would like to set $T = U(D + C^a) - C^a$. Unfortunately, T need not be additive, as the following lemma shows.

LEMMA 3.2. ([2, COR.3.3]) *Given an additive metric A and a centroid quasi-metric Q, $A + Q$ is additive if and only if $A + Q$ satisfies the triangle inequality.*

In particular, we are subtracting C^a from $U(D + C^a)$, and when we do so, some of the stems leading to leaves might become negative. In [1], it was shown how to fix this problem, and we refer readers to their solution. Their modification, which changes the function $U()$ so as to guarantee that $T = U(D + C^a) - C^a$ satisfies the triangle inequality, yields a three approximation for the L_∞ criterion, no matter which a is chosen. Each choice of pivot costs $O(n^2)$ to evaluate, so the SP method costs up to $O(n^3)$ time.

3.2 The Double Pivot (DP) Heuristic The SP method above suggests that other methods for picking the root of the ultrametric will yield a different heuristic. In this section, we use this idea to derive a new heuristic. In particular, suppose we pick two arbitrary points $a, b \in S$, which we call a *pivot pair*. Then we can root T at their midpoint and find the ultrametic so defined. In particular, we must derive the correct centroid C^{ab} which will yield the desired ultrametric, which we do as follows.

Let $m_{ab} = \max_i\{D[a,i], D[b,i]\}$. Let C^{ab} be the centroid metric with $l_i = m_{ab} - \max\{D[a,i], D[b,i]\}$. Then, if D is an additive metric, $D + C^{ab}$ will be an ultrametric rooted at the midpoint between a and b. As above, we show that

LEMMA 3.3. *For all a, b, D is quasi-additive if and only if $D + C^{ab}$ is ultrametric.*

So the DP heuristic is as follows. Pick a pair $a, b \in S$ (or try all pairs to see which one give the best answer). Output $T = U(D + C^{ab}) - C^{ab}$. As before, we need to make sure that T is a tree, and we refer the reader to [1] for the details of how this is done. We conclude with the following.

LEMMA 3.4. *The DP heuristic yields a 3-approximation for the L_∞ Numerical Taxonomy problem.*

Proof: Similar to [1]. ∎

As before, each choice of pivot costs $O(n^2)$ time, so DP takes up to $O(n^4)$ time.

3.3 The Neighbor Joining (NJ) Heuristic The NJ method works by repeatedly finding siblings s_1 and s_2, and collapsing them into a single node s. We then set distances from s to all remaining nodes of the tree as follows. Let a_1 and a_2 be the average distances from s_1 and s_2, respectively, to all other nodes. Then we can estimate the distance $D[s, s_1]$ to be $(D[s_1, s_2] + a_1 - a_2)/2$, and similarly for $D[s, s_2]$. Now, for any other node n, $D[s,n] = D[s_1,n] - D[s,s_1]$. Delete s_1 and s_2 and iterate.

The question is then how to find a sibling pair. Let $a_i = \sum_{j=1}^{n} D[i,j]/(n-1)$ be the average distance from i to the other points of the metric space. Then it can be shown that if D is an additive metric, the pair i, j minimizing $D[i,j] - a_i - a_j$ must be a sibling pair. The NJ method selects the pair which minimizes the quantity at each stage to be the sibling pair. NJ can be implemented to run in $O(n^2)$ time.

3.4 Local optimizationA generic sort of heuristic for optimization problems is to make a local change in the structure of a solution and see if the answer gets better. The form this heuristic takes depends in two ways on the problem being studied. First, what kind of local change in the structure is appropriate, and second, how does one choose from amongst the possible changes for a possible improving change?

In tree construction, there is often a strong signal of where the local structure of a tree is faulty. Suppose

In [1], the first positive result for numerical taxonomy was presented. They showed that if ε is the distance to the closest tree metric under the L_∞ norm, i.e., $\varepsilon = \min_T \{L_\infty(T - D)\}$, then it is possible to construct a tree T such that $L_\infty(T - D) \leq 3\varepsilon$, that is, they gave a 3-approximation algorithm for this problem.

Their result is achieved by transforming the general tree metric problem to that of ultrametrics with a loss of a factor of 3 on the approximation ratio. An ultrametric is a tree metric in which the tree can be rooted so that the root is equidistant from all points of the metric. In [9], it was shown that under the L_∞ norm an optimal ultrametric is polynomially computable, in fact in linear time. For reasons described below, we will refer to the heuristic from [1] as the *Single Pivot (SP)* heuristic.

Our Results While the SP heuristic has good worst-case behaviour, it was not clear from the analysis of [1] how it would perform on real data. Furthermore, the analysis of SP showed that its worst-case behaviour is good for L_∞, but not for other norms, for relative norms, or for the minimum evolution criterion. Indeed, while the trivial 2-approximation via MST holds for this type of steiner tree problem, the ultrametric transform is an unlikely approach for the minimum evolution criterion, since in [9] it was shown that the minimum evolution criterion for ultrametrics is as hard to approximate as graph coloring. Thus the performance of SP for biological data was unclear.

In this paper, we introduce an extension of SP, which we call the *Double Pivot (DP)* method. We present experimental results for the performance of SP/DP versus NJ. We show that SP/DP outperforms NJ on both biological and random data, for almost all standard evaluation criteria.

The paper is organized as follows. After some preliminary definitions in Section 2, we describe the single and double pivot methods, as well as Neighbor Joining in Section 3. In Section 4, we outline the design of our experiments and give results. We conclude in Section 5.

2 Preliminaries

2.1 Metrics We present some basic definition.

DEFINITION 2.1. *A* metric *on a set $S = \{1, \ldots, n\}$ is a function $D : S^2 \to \Re_{\geq 0}$ such that*

- $D[x, y] = 0 \iff x = y$,
- $D[x, y] = D[y, x]$,
- $D[x, y] \leq D[x, z] + D[z, y]$ (the triangle inequality).

Likewise, $D : S^2 \to \Re_{\geq 0}$ is a *quasi-metric* if it satisfies the first two conditions. For (quasi-)metrics A and B, $A + B$ is the usual matrix addition, i.e., $(A + B)[i, j] = A[i, j] + B[i, j]$.

DEFINITION 2.2. *A (quasi-)metric D is* (quasi-)additive *if there exists a weighted tree T spanning the points of the metric such that $D[i, j]$ is the path length in T from i to j.*

DEFINITION 2.3. *An additive metric D is an* ultrametric *if its tree T can be rooted so that all points of the metric are equidistant from the root.*

DEFINITION 2.4. *A quasi-metric D is a* centroid *quasi-metric if $\exists l_1, \ldots, l_n$ such that $\forall i \neq j$, $D[i, j] = l_i + l_j$.*

A centroid quasi-metric D is a *centroid metric* if $l_i \geq 0$ for all i. A centroid metric is a type of tree metric since it can be realized by a weighted tree with a star topology and edge weights l_i.

2.2 Optimization Criteria The k-norms are formally defined as follows.

DEFINITION 2.5. *For $n \times n$ real-valued matrices M and $k \geq 1$, define the k-*norm *by*

$$L_k(M) = (\sum_{i<j} \mid M[i, j] \mid^k)^{\frac{1}{k}},$$

$$L_\infty(M) = \max_{i<j} \{\mid M[i, j] \mid\}.$$

DEFINITION 2.6. *We define the k-*norm distance *as $L_k(A, B) = L_k(A - B)$. We define the k-*norm increment distance *as*

$$L_k^+(A, B) = \begin{cases} L_k(A, B) & \text{if } \forall i, j \ A[i, j] \geq B[i, j]; \\ \infty & \text{otherwise.} \end{cases}$$

$$L_\infty^+(A, B) = \begin{cases} L_\infty(A, B) & \text{if } \forall i, j \ A[i, j] \geq B[i, j]; \\ \infty & \text{otherwise.} \end{cases}$$

We define the relative k-norms distance *as*

$$R_k(A, B) = \left(\sum_{i<j} \left| \frac{A[i, j] - B[i, j]}{B[i, j]} \right|^k \right)^{\frac{1}{k}},$$

$$R_\infty(A, B) = \max_{i<j} \left\{ \left| \frac{A[i, j] - B[i, j]}{B[i, j]} \right| \right\}.$$

3 Heuristics

3.1 The single pivot (SP) Heuristic Suppose T is a tree which defines a tree metric over some subset S of its vertices. Suppose we want to convert a weighted

Numerical Taxonomy on Data: Experimental Results

Jaime Cohen* Martin Farach†

Abstract

We consider the problem of fitting an $n \times n$ distance matrix D by a tree metric T. This problem is NP-hard for most reasonable distance functions between D and T. Recently, an approximation algorithm was presented [1] which achieves a factor of 3 approximation to the L_∞ best fitting tree. We call this method the *Single Pivot (SP)* heuristic

Within the biology community, the so-called *Neighbor-Joining (NJ)* heuristic [14] has wide acceptance. In this paper, we introduced a new *Double Pivot (DP)* heuristic, which is an extension of the SP heuristic, and show that DP outperforms NJ on biological and random data.

1 Introduction

One of the most common methods for clustering numeric data involves fitting the data to a *tree metric*, which is defined by a weighted tree spanning the points of the metric, the distance between two points being the sum of the weights of the edges on the path between them. Not surprisingly, this problem, the so-called *Numerical Taxonomy* problem, has received a great deal of attention (see [2, 16, 17] for extensive surveys). Fitting distances by trees is an important problem in many areas. For example, in statistics, the problem of clustering data into hierarchies is exactly the tree fitting problem. In "historical sciences" such as paleontology, historical linguistics, and evolutionary biology, tree metrics represent the branching processes which have led to some observed distribution of data. Thus, the numerical taxonomy problem has been, and continues to be, the subject of intense research.

In particular, consider the case of evolutionary biology. By comparing the DNA sequences of pairs of species, biologists get an estimate of the evolutionary time which has elapsed since the species separated by a speciation event. A table of pairwise distances is thus constructed. The problem is then to reconstruct the underlying evolutionary tree. Many heuristics for this problem appear in the literature every year (see, e.g., [17]), although the *Neighbor Joining (NJ)* heuristic of Saitou and Nei [14] remains the method of choice within the biology community.

The numerical taxonomy problem is usually cast in the following terms. Let S be the set of species under consideration.

The Numerical Taxonomy Problem

Input: $D : S^2 \to \Re_{\geq 0}$, a distance matrix.

Output: A *tree metric* T which spans S such that the distance from T to D is minimized.

This definition does not specify a distance function. Since T is simply a matrix of distances[1], we can take any standard distance function between matrices, such as L_1, L_2, or L_∞. That is, for some $k \in \{1, 2, \ldots, \infty\}$, we want to find a tree metric T minimizing $L_k(T - D)$. We can similarly seek to minimize the *relative error*, which is defined in terms of percent change, rather than absolute change. Another type optimization criterion used in the biology literature is the so-called *Minimum Evolution Criterion (ME)*. Here we seek the tree T, such that $T[i, j] \geq D[i, j]$, for all i and j, and such that the sum of the weight of the edges in T is minimized. This is the Steiner Tree problem for tree metrics.

History The numerical taxonomy problem for additive metric fitting under L_k norms was explicitly stated in its current form in 1967 [5]. Since then it has collected an extensive literature. In 1977 [20], it was shown that if there is a tree T coinciding exactly with D, it is unique and constructible in linear, i.e., $O(|S|^2)$, time. Unfortunately there is typically no tree T coinciding exactly with D, and in 1987 [7], it was shown that for L_1 and L_2, the numerical taxonomy problem is $\mathcal{NP}$-hard. In [1], it was shown that the numerical taxonomy problem is $\mathcal{NP}$-hard for L_∞. In fact, it was shown that finding a tree within 9/8ths of optimal is $\mathcal{NP}$-hard for this criterion. Additional complexity results appear in [19].

*Department of Computer Science, Rutgers University, Piscataway, NJ 08855, USA. (*jaimecoh@paul.rutgers.edu, http://paul.rutgers.edu/~jaimecoh*) Supported by a CAPES-Brazil Fellowship.

†Department of Computer Science, Rutgers University, Piscataway, NJ 08855, USA. (*farach@cs.rutgers.edu, http://www.cs.rutgers.edu/~farach*) Supported by NSF Career Development Award CCR-9501942, an Alfred P. Sloan Research Fellowship and NATO Grant 960215.

[1] If a matrix exactly fits a tree metric, then the tree is unique, so a tree metric is equivalently represented by the matrix or the tree