

rMSA: Interface to Popular Multiple Sequence Alignment Tools

Michael Hahsler
Southern Methodist University

Anurag Nagar
University of Texas at Dallas

Abstract

There are many stand-alone tools available for Bioinformatics. This package aims at using R and the Biostrings package as the common interface for several important tools for multiple sequence alignment (e.g., ClustalW, Kalign, MAFFT and MUSCLE).

Keywords: bioinformatics, Bioconductor, biostrings, multiple sequence alignment.

1. Introduction

There are many tools available for multiple sequence alignment. Some tools are: T-Coffee (Notredame, Higgins, and Heringa 2000), MAFFT (Katoh, Misawa, Kuma, and Miyata 2002), MUSCLE (Edgar 2004b,a), Kalign (Lassmann and Sonnhammer 2006) and ClustalW2 and ClustalX2 (Larkin, Blackshields, Brown, Chenna, McGettigan, McWilliam, Valentin, Wallace, Wilm, Lopez, Thompson, Gibson, and Higgins 2007). Typically, for these tools can be used online via a Web site and/or the user can download and install software with a command-line interface. Often the input and output data is stored in files using various formats and the parameters that need to be supplied to the command-line interface varies greatly between different tools. All this makes using and comparing several approaches time consuming and error prone. The R-based Bioconductor project (Gentleman, Carey, Bates, and others 2004) provides a convenient infrastructure to handle and manipulate bioinformatics data. The **Biostrings** package in particular provides infrastructure for DNA, RNA and protein sequences as well as (multiple) sequence alignments. Also basic algorithms for pairwise sequence alignment are included. However, to obtain multiple sequence alignments, the user has to export the sequences into a file, then run the needed tools manually and re-import the results.

In **rMSA** we provide a simple interface to a growing set of popular tools. The tools are called directly from within R and no manual data export or import are needed. Currently we interface *ClustalW*, *Kalign*, *MAFFT* and *MUSCLE*.

2. Installing Third-Party Software

rMSA does not provide third-party software, but interfaces correctly installed software. This has the advantages that not all software has to be installed if only some of it is needed and that the user can always install the current version of the software.

Instructions on where to find the needed third-party software can be found in the manual

page for each function.

The package is loaded using:

```
R> library("rMSA")
```

To read about installing the third-party software refer to the INSTALL file.

3. Multiple Sequence Alignment

Multiple Sequence Alignment (MSA) involves comparing and aligning more than two sequences to each other. The aim is to discover regions of high similarity in a set of sequences. Although, computationally complex, MSA is quite often what biologists need to characterize a group of sequences that, e.g., are evolutionary related to each other by sharing a common ancestor. Such sequences are said to be homologous. Similarly, biologists might be interested in the similarity of genes from different organisms based on their sequences. Another area of application is to find regions which are conserved for a given species or genus. Such conserved regions can be used for identification and classification of organisms.

MSA is a NP-hard problem ?? and is computationally more complex than pairwise alignment. Some techniques that are used for pairwise alignment, such as dynamic programming, can also be used for MSA but have much greater run time requirements. To obtain results in reasonable time, various heuristics have been proposed such as progressive alignment, iterative refinement methods, and Hidden Markov Models ?. Out of these, progressive alignment is the most commonly used in many tools for MSA such as Clustal?.

3.1. ClustalW

We read an example FASTA file with DNA which comes with this package, take the first 60 nucleotides and run Clustal.

```
R> dna <- readDNASTringSet(system.file("examples/DNA_example.fasta",
+      package="rMSA"))
R> dna <- narrow(dna, start=1, end=60)
R> al <- clustal(dna)
R> al
```

DNAMultipleAlignment with 5 rows and 98 columns

	aln	names
[1]	-----...-GTGGCGGACGGGTGAGTAA	4403
[2]	-----...-GTGGCGGACGG-----	4404
[3]	-----...CGTGGCGCA-----	4399
[4]	AGAGTTTGATCCTGGCTCAGA...-----	1675
[5]	AGAGTTTGATTATGGCTCAGA...-----	4411

Using detail the alignment can be inspected.

```
R> detail(al)
```

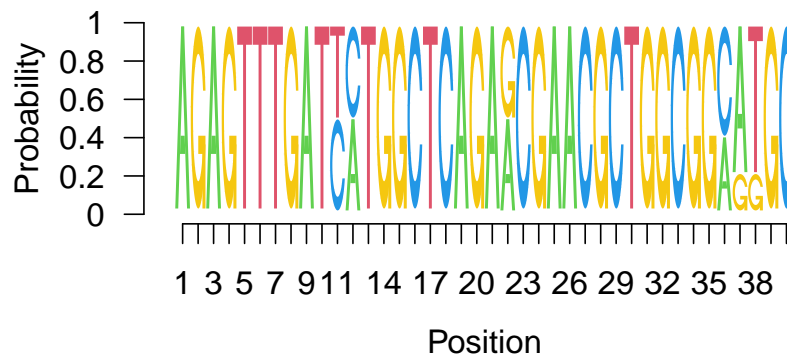


Figure 1: Sequence logo of alignment.

```

4403  ---  -----GGAATGCTNAAACACATGCAAGTCGCACGG--
4404  ---  -----GCTGGCGGAATGCTTAACACATGCAAGTCGCACGGGG
4399  ---  -----GCTGGCGGCAAGCTTAACACATGCAAGTCGAACGGGG
1675  AGA  TTTGATCCTGGCTCAGAACGAACGCTGGCGGCCTGCCTAACACATGCAAGTCGAAC---
4411  AGA  TTTGATTATGGCTCAGAGCGAACGCTGGCGGCATGCTTAACACATGCAAGTCGCAC---
consensus  gctggcGGcatGCTtAACACATGCAAGTCGcACgg

4403  ---  GCAGC--AATGTCA-GTGGCGGACGGGTGAGTAA
4404  ---  GTTTC--GGCCTTA-GTGGCGGACGG-----
4399  ---  ACCTTCGGGTCTTACGTGGCGCA-----
1675  AGA  -----
4411  AGA  -----
consensus  g      aa  t  a  gtggcg  a

```

Figure 2: Representation of a DNA multiple alignment using boxshade.

Plot produces the sequence logo shown in Figure 1.

```
R> plot(al, 1, 40)
```

Boxshade (if installed) can also be used for producing a pdf of the alignment. Figure 2 shows the result.

```
R> boxshade(al, file="alignment.pdf")
```

Clustal can also be used for RNA and protein sequences.

```
R> rna <- readRNAStringSet(system.file("examples/RNA_example.fasta",
+   package="rMSA"))
R> rna
```

RNAStringSet object of length 5:

	width	seq	names
[1]	1481	AGAGUUUGAUCCUGGCUC...AGUCGUAACAAGGUAACC	1675 AB015560.1 d...
[2]	1404	GCUGGCGGCAGGCCUAAC...UAAGGUCAGCGACUGGGG	4399 D14432.1 Rho...
[3]	1426	GGAAUGCUNAACACAUGC...GGUAGCCGUAGGGGAACC	4403 X72908.1 Ros...

```
[4] 1362 GCUGGCGGAAUGCUUAAC...UAGGUGUCUAGGCUAAC 4404 AF173825.1 A...
[5] 1458 AGAGUUUGAUUAUGGCUC...UCGUAACAAGGUAACCGU 4411 Y07647.2 Dre...
```

```
R> al <- clustal(rna)
R> al
```

RNAMultipleAlignment with 5 rows and 1500 columns

	aln	names
[1]	-----...AAGGUAGCCGUAGGGAACC	4403
[2]	-----...-----	4404
[3]	AGAGUUUGAUUAUGGCUCAGA...AAGGUAACCGU-----	4411
[4]	-----...-----	4399
[5]	AGAGUUUGAUCCUGGCUCAGA...AAGGUAACC-----	1675

```
R> aa <- readAAStringSet(system.file("examples/Protein_example.fasta",
+      package="rMSA"))
R> aa
```

AAStringSet object of length 5:

	width	seq	names
[1]	170	MKKSRRRIWIFGLLFSIW...DVYYLEAPFFQGRKCGGT	gi 340754543 ref ...
[2]	233	MYIIWKLLFFKGENVVEH...KEEEVISVDDILKKRRE	gi 340754544 ref ...
[3]	326	MKRSLSGIQPSGILHLGN...KKVQEAKEIVGLLGNIYR	gi 340754545 ref ...
[4]	317	MKYYSQVDLGGTNTKIGL...VLGNEAGILGAAALFMLS	gi 340754546 ref ...
[5]	337	MKKMGIIILGALVLAAGLV...IVLVPSIGIDKENVAEYK	gi 340754547 ref ...

```
R> al <- clustal(aa)
R> al
```

AAMultipleAlignment with 5 rows and 358 columns

	aln	names
[1]	---MKKSRRRIWIFGLLFSIW...-----	gi 340754543 ref ...
[2]	---MYIIWKLLFFKGENVVEH...-----	gi 340754544 ref ...
[3]	MKKMGIIILGALVLAAGLVGCG...DKENVAEYK-----	gi 340754547 ref ...
[4]	---MKRSLSGIQPSGILHLGN...ASKKVQEAKEIVGLLGNIYR	gi 340754545 ref ...
[5]	----MKYYSQVDLGGTNTKIGL...-----	gi 340754546 ref ...

3.2. Kalign

Another popular technique for MSA is based on the KAlign algorithm [Lassmann and Sonnhammer \(2005\)](#). It uses a progressive method for sequence alignment by first calculating pairwise distances between sequences and then constructing a guide tree from these pairwise alignments. The guide tree is used to progressively create the multiple sequence alignment profile. KAlign uses the Wu-Manber approximate string matching algorithm [Wu and Manber \(1992\)](#) for distance calculation. KAlign has been evaluated to be faster and more efficient than other methods [Lassmann and Sonnhammer \(2005\)](#) due to the use of the approximate string matching algorithm and efficient guide tree generation.

```
R> dna <- readDNASTringSet(system.file("examples/DNA_example.fasta",
+   package="rMSA"))
R> dna
```

DNASTringSet object of length 5:

	width	seq	names
[1]	1481	AGAGTTTGTATCCTGGCTC...AGTCGTAACAAGGTAACC	1675 AB015560.1 d...
[2]	1404	GCTGGCGGCAGGCCTAAC...TAAGGTCAGCGACTGGGG	4399 D14432.1 Rho...
[3]	1426	GGAATGCTNAACACATGC...GGTAGCCGTAGGGGAACC	4403 X72908.1 Ros...
[4]	1362	GCTGGCGGAATGCTTAAC...TAGGTGTCTAGGCTAACC	4404 AF173825.1 A...
[5]	1458	AGAGTTTGATTATGGCTC...TCGTAACAAGGTAACCGT	4411 Y07647.2 Dre...

```
R> ### align the sequences
R> al <- kalign(dna)
R> al
```

DNAMultipleAlignment with 5 rows and 1500 columns

	aln	names
[1]	AGAGTTTGTATCCTGGCTCAGA...-----CAAGGTAAC--C	1675
[2]	G-----...-----GCGACTGGG--G	4399
[3]	G-----...GGTAGCCGTAGGGGAAC--C	4403
[4]	G-----...-----TAGGCTAAC--C	4404
[5]	AGAGTTTGATTATGGCTCAGA...-----CAAGGTAACCGT	4411

3.3. MUSCLE

MUSCLE uses a multi-stage approach based on k -mer distance and binary guide trees to produce high-quality MSA's very quickly (Edgar 2004b,a).

```
R> dna <- readDNASTringSet(system.file("examples/DNA_example.fasta",
+   package="rMSA"))
R> dna
```

DNASTringSet object of length 5:

	width	seq	names
[1]	1481	AGAGTTTGTATCCTGGCTC...AGTCGTAACAAGGTAACC	1675 AB015560.1 d...
[2]	1404	GCTGGCGGCAGGCCTAAC...TAAGGTCAGCGACTGGGG	4399 D14432.1 Rho...
[3]	1426	GGAATGCTNAACACATGC...GGTAGCCGTAGGGGAACC	4403 X72908.1 Ros...
[4]	1362	GCTGGCGGAATGCTTAAC...TAGGTGTCTAGGCTAACC	4404 AF173825.1 A...
[5]	1458	AGAGTTTGATTATGGCTC...TCGTAACAAGGTAACCGT	4411 Y07647.2 Dre...

```
R> al <- muscle(dna)
R> al
```

DNAMultipleAlignment with 5 rows and 1502 columns

	aln	names
--	-----	-------

```
[1] AGAGTTTGATCCTGGCTCAGA...AAGGTAACC----- 1675
[2] -----...----- 4399
[3] AGAGTTTGATTATGGCTCAGA...AAGGTAACCGT----- 4411
[4] -----...AAGGTAGCCGTAGGGGAACC 4403
[5] -----...----- 4404
```

3.4. MAFFT

MAFFT (Kato *et al.* 2002) is a similarity-based MSA technique using progressive and iterative refinement methods.

```
R> dna <- readDNAStrngSet(system.file("examples/DNA_example.fasta",
+   package="rMSA"))
R> dna
```

DNAStrngSet object of length 5:

	width	seq	names
[1]	1481	AGAGTTTGATCCTGGCTC...AGTCGTAACAAGGTAACC	1675 AB015560.1 d...
[2]	1404	GCTGGCGGCAGGCCTAAC...TAAGGTCAGCGACTGGGG	4399 D14432.1 Rho...
[3]	1426	GGAATGCTNAACACATGC...GGTAGCCGTAGGGGAACC	4403 X72908.1 Ros...
[4]	1362	GCTGGCGGAATGCTTAAC...TAGGTGTCTAGGCTAACC	4404 AF173825.1 A...
[5]	1458	AGAGTTTGATTATGGCTC...TCGTAACAAGGTAACCGT	4411 Y07647.2 Dre...

```
R> al <- mafft(dna)
R> al
```

DNAMultipleAlignment with 5 rows and 1499 columns

	aln	names
[1]	AGAGTTTGATCCTGGCTCAGA...AAGGTAACC-----	1675
[2]	-----...-----	4399
[3]	-----...AAGGTAGCCGTAGGGGAACC	4403
[4]	-----...-----	4404
[5]	AGAGTTTGATTATGGCTCAGA...AAGGTAACCGT-----	4411

4. Auxiliary Function

4.1. Creating Random Sequences

Creating random sequences given letter probabilities.

```
R> seqs <- random_sequences(100, number=10, prob=c(a=.5, c=.3, g=.1, t=.1))
R> seqs
```

DNAStrngSet object of length 10:

	width	seq	names
[1]	100	ACCCGCAACCCCATAGAA...CAGAAAGATAAACAAAC	1
[2]	100	ACAAAAAAACATAATTA...ATAGCACCTAGGGGCTC	2
[3]	100	CCACCCAAATCAACCTCC...CCAAACGCATACCCACA	3
[4]	100	ATCATAATCCTCAAAAAA...AAACATTCCCCATCCAA	4
[5]	100	CACCCACACACGTAGACC...AAACCCACCTACACACC	5
[6]	100	CGGACGCGACATTACCA...AAAATTCTGACACCCCA	6
[7]	100	AAACAAGACAAGAATAAC...AGAGACAGAACAAACAC	7
[8]	100	ACCAAAACACCTTAAAAA...AACGACACACCCACGAG	8
[9]	100	AGCAACAACACATCAAAG...CCTAAAATCCAAACCTG	9
[10]	100	CATATAAACAAAAAAAT...CTAATAAACTACACATA	10

Creating random sequences using dinucleotides transition probabilities

```
R> prob <- matrix(runif(16), nrow=4, ncol=4, dimnames=list(DNA_BASES, DNA_BASES))
R> prob <- prob/rowSums(prob)
R> seqs <- random_sequences(100, number=10, prob=prob)
R> seqs
```

DNAStrngSet object of length 10:

	width	seq	names
[1]	100	AAACGCGAAACAGTGGTT...TGCGTGTGCTGGTGAGA	1
[2]	100	CGGTGGTTGCTGGTTAGT...GCGGTGACCTAAGGTTT	2
[3]	100	GGGTGAAAAAAGTCGTCA...TGGTGCCAGCTAACGTA	3
[4]	100	TTAGTTACGCGTGTGCGT...TAACAACGTGTGTACGA	4
[5]	100	AGTGACCAGGCTGCGGTG...AAAACCAACGCTAGGTG	5
[6]	100	TGTGGTGTGGAAAGGTGC...AGTGCTGAGCTTGTGTG	6
[7]	100	AAGCACACGCGTCTGGCT...GCTAGTACTGTGCTGTG	7
[8]	100	GTAGTGGCGTGTGCTT...TGCGTGTAGTCAAAAAG	8
[9]	100	TTGAAAGTTTGCGTGTCC...CGCTTGCACTTAGTGCT	9
[10]	100	AGCTCCGTTGCGGGCGTT...CGTAAGGGTTGTCAGTG	10

Creates a set of sequences which are random mutations (with base changes, insertions and deletions) for a given DNA, RNA or AA sequence.

```
R> s <- random_sequences(100, number=1)
R> s
```

DNAStrngSet object of length 1:

	width	seq	names
[1]	100	GAAATAACGATACTAGCC...ACTCCTTTTCCTGCAGGG	1

```
R> ### create 10 sequences with 1 percent base changes, insertions and deletions
R> m <- mutations(s, 10, change=0.01, insertion=0.01, deletion=0.01)
R> m
```

DNAStrngSet object of length 10:

	width	seq	names
[1]	100	GAAATAACGATACTAGCC...CTCCTTTTCCTGCAGGG	1_mutation_1
[2]	101	GAAATAACGATACTAGCC...CTCCTTTTCCTGCAGGG	1_mutation_2
[3]	99	GAAATACGATACTAGCCA...CTCCTTTTCCTGCAGGG	1_mutation_3
[4]	100	GAAATAACGATACTAGCC...CTCCTTTTCCTGCAGGG	1_mutation_4
[5]	101	AAAATAACGATACTAGCC...CCTTTTCCTCGCAGGGT	1_mutation_5
[6]	101	GAAATAACGATACTAGCC...TCCTTTTCCTGCCAGGG	1_mutation_6
[7]	100	GAAATAACGATACTAGCC...ACTCCTTTTCCTGCAGG	1_mutation_7
[8]	101	GAAATAACGATACTAGCC...CTCCTTTGTCCTGCAGG	1_mutation_8
[9]	99	GAAATAACATACTAGCCA...ACTCCTTTTCCTGCATG	1_mutation_9
[10]	100	GAAATAACGATACTAGCC...CTCCTTTTCCTGCAAGG	1_mutation_10

R> `clustal(c(s,m))`

DNAMultipleAlignment with 11 rows and 110 columns

	aln	names
[1]	GAAATAACGATACTAGCCCA...CTCCTTT-TCCTGCAGGG--	1_mutation_1
[2]	GAAATA-CGATACTAGCC-A...CTCCTTT-TCCTGCAGGG--	1_mutation_3
[3]	GAAATAACGATACTAGCC-A...CTCCTTT-TCCTGCCAGGG-	1_mutation_6
[4]	GAAATAACGATACTAGCC-A...CTCCTTT-TCCTGCAAGG--	1_mutation_10
[5]	AAAATAACGATACTAGCC-A...CTCCTTT-TCCTCGCAGGGT	1_mutation_5
[6]	GAAATAACGATACTAGCC-A...CTCCTTTGTCCTGCAGG---	1_mutation_8
[7]	GAAATAAC-ATACTAGCC-A...CTCCTTT-TCCTGCATG---	1_mutation_9
[8]	GAAATAACGATACTAGCC-A...CTCCTTT-TCCTGCAGG---	1_mutation_7
[9]	GAAATAACGATACTAGCC-A...CTCCTTT-TCCTGCAGGG--	1_mutation_4
[10]	GAAATAACGATACTAGCC-A...CTCCTTT-TCCTGCAGGG--	1
[11]	GAAATAACGATACTAGCC-A...CTCCTTT-TCCTGCAGGG--	1_mutation_2

4.2. Calculating Distances between Sequences

Sequence alignment and distance calculation between sequences (e.g., edit distance) are related. Some MSA heuristics even use similarities/distances to guide the alignment process. **rMSA** provides the following distance metrics:

- Feature frequency profile (**distFFP**): A FFP is the normalized (by the number of k -mers in the sequence) count of each possible k -mer in a sequence. The distance is defined as the Jensen-Shannon divergence (JSD) between FFPs ([Sims and Kim](#)).
- Composition Vector (**distCV**): A CV is a vector with the frequencies of each k -mer in the sequence minus the expected frequency of random background nice obtained from a Markov Model (not implemented yet!). The cosine distance is used between CVs ([Qi, Wang, and Hao 2004](#)).
- Numerical Summarization Vector (**distNSV**): An NSV is frequency distribution of all possible k -mers in a sequence. The Manhattan distance is used between NSVs ([Nagar and Hahsler 2013](#)).

- Distance between sets of k -mers (`distkMer`): Each sequence is represented as a set of k -mers. The Jaccard (binary) distance is used between sets (number of unique shared k -mers over the total number of unique k -mers in both sequences).
- Distance based on SimRank (`distSimRank`): $1 - \text{simRank}$. The function `simRank` is also available (DeSantis, Keller, Karaoz, Alekseyenko, Singh, Brodie, Pei, Andersen, and Larsen 2011).
- Edit (Levenshtein) Distance (`distEdit`): Edit distance between sequences.
- Distance based on alignment score (`distAlignment`): see `stringDist` in **Biostrings**.
- Evolutionary distances (`distApe`): see `dist.dna` in **ape**.

In the following example we create 100 mutations from a random single sequence.

```
R> s <- random_sequences(len = 100)
R> s
```

DNAStrngSet object of length 1:

	width	seq	names
[1]	100	TATCATTAGGATCTTTGC...CGGCTTGGACCGGATACT	1

```
R> ms <- mutations(s, number = 100)
R> ms
```

DNAStrngSet object of length 100:

	width	seq	names
[1]	102	TATCATTAGTATCTTTG...CTTGGGACCGGATACTT	1_mutation_1
[2]	98	TATCATCTAGGATCTTT...CGGCTTGGACCGGATACT	1_mutation_2
[3]	101	TATCATTAGGATCTTTG...GCTTGGACCGGATACCT	1_mutation_3
[4]	101	TATCACTAGGATCTTTG...GGCTTGGACCGGATACT	1_mutation_4
[5]	98	TATCATTAGGTCTTTGC...GCTTGGACCGGATACTT	1_mutation_5
...
[96]	101	TATCATTAGGATCTTTG...GCTCTGGACCGGATACT	1_mutation_96
[97]	102	TATCATGTAGGATCTTT...GGCTTGGACCGGATACT	1_mutation_97
[98]	102	TATCATTAGGATCTTTG...GGCTTGGACCGGATACT	1_mutation_98
[99]	101	ATATCATTAGGATCTTT...CGGCTGGACCGGATACT	1_mutation_99
[100]	101	TATCATTAGGATCTTTG...GCTTGGACACCGGATACT	1_mutation_100

Since the mutations also include insertions and deletions, the set of resulting sequences have different length (see width above). We calculate different distance measures on the set of mutations and then compare how much the different measures agree. We use Spearman's rank correlation coefficient to evaluate agreement and show the resulting relationship in Figure 3.

```
R> dNSV <- distNSV(ms)
R> dFFP <- distFFP(ms)
R> dCV <- distCV(ms)
```

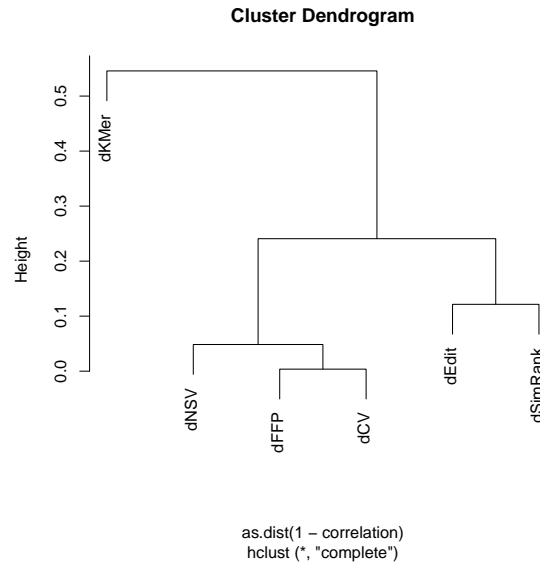


Figure 3: Correlation between different distance measures.

```
R> dKMer <- distKMer(ms)
R> dSimRank <- distSimRank(ms)
R> dEdit <- distEdit(ms)
R> correlation <- cor(cbind(dEdit, dNSV, dFFP, dCV, dKMer, dSimRank),
+   method = "spearman")
R> correlation
```

	dEdit	dNSV	dFFP	dCV	dKMer	dSimRank
dEdit	1.0000	0.7973	0.7592	0.7630	0.4577	0.8785
dNSV	0.7973	1.0000	0.9515	0.9532	0.5485	0.8171
dFFP	0.7592	0.9515	1.0000	0.9963	0.4878	0.7745
dCV	0.7630	0.9532	0.9963	1.0000	0.4756	0.7788
dKMer	0.4577	0.5485	0.4878	0.4756	1.0000	0.4543
dSimRank	0.8785	0.8171	0.7745	0.7788	0.4543	1.0000

```
R> plot(hclust(as.dist(1-correlation)))
```

SimRank produces distances most closely related to edit distance.

`distApe` calculates evolution-based distances between aligned sequences. First we align the first 20 sequence using Clustal and then calculate distances using the Kimura (1980) model which incorporates assumptions about certain base transitions and transversion. We use hierarchical clustering on the distances and display the resulting dendrogram in Figure 4.

```
R> dK80 <- distApe(clustal(ms[1:20]), model="K80")
R> par.old <- par(mar=c(5,1,2,6)+.1)
R> plot(as.dendrogram(hclust(dK80)), horiz=TRUE, type="triangle",
+   xlab="Kimura (1980) distance")
R> par(par.old)
```

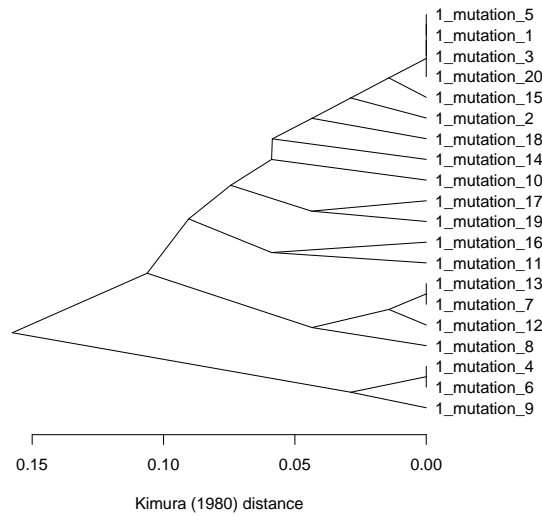


Figure 4: Evolutionaty relationship between 20 mutations based on Kimura (1980) distance.

5. Conclusion

Acknowledgments

This research was supported by research grant no. R21HG005912 from the National Human Genome Research Institute (NHGRI / NIH).

References

- DeSantis T, Keller K, Karaoz U, Alekseyenko A, Singh N, Brodie E, Pei Z, Andersen G, Larsen N (2011). “Simrank: Rapid and sensitive general-purpose k-mer search tool.” *BMC Ecology*, **11**(1). ISSN 1472-6785. doi:10.1186/1472-6785-11-11.
- Edgar R (2004a). “MUSCLE: a multiple sequence alignment method with reduced time and space complexity.” *BMC Bioinformatics*, **5**(1), 113+. ISSN 1471-2105.
- Edgar RC (2004b). “Muscle: multiple sequence alignment with high accuracy and high throughput.” *Nucleic Acids Research*, **32**, 1792–1797.
- Gentleman RC, Carey VJ, Bates DM, others (2004). “Bioconductor: Open software development for computational biology and bioinformatics.” *Genome Biology*, **5**, R80. URL <http://genomebiology.com/2004/5/10/R80>.
- Katoh K, Misawa K, Kuma K, Miyata T (2002). “MAFFT: a novel method for rapid multiple sequence alignment based on fast Fourier transform.” *Nucleic Acids Research*, **30**(14), 3059–3066.

- Larkin M, Blackshields G, Brown N, Chenna R, McGettigan P, McWilliam H, Valentin F, Wallace I, Wilm A, Lopez R, Thompson J, Gibson T, Higgins D (2007). “Clustal W and Clustal X version 2.0.” *Bioinformatics*, **23**, 2947–2948. ISSN 1367-4803.
- Lassmann T, Sonnhammer EL (2005). “Kalign—an accurate and fast multiple sequence alignment algorithm.” *BMC bioinformatics*, **6**(1), 298.
- Lassmann T, Sonnhammer EL (2006). “Kalign, Kalignvu and Mumsa: web servers for multiple sequence alignment.” *Nucleic Acids Research*, **34**. ISSN 1362-4962.
- Nagar A, Hahsler M (2013). “Fast discovery and visualization of conserved regions in DNA sequences using quasi-alignment.” *BMC Bioinformatics*, **14**(Suppl. 11).
- Notredame C, Higgins DG, Heringa J (2000). “T-Coffee: A novel method for fast and accurate multiple sequence alignment.” *Journal of Molecular Biology*, **302**(1), 205–217. ISSN 0022-2836.
- Qi J, Wang B, Hao BI (2004). “Whole proteome prokaryote phylogeny without sequence alignment: a K-string composition approach.” *Journal of Molecular Evolution*, **58**(1). doi: [10.1007/s00239-003-2493-7](https://doi.org/10.1007/s00239-003-2493-7).
- Sims G, Kim S (????). “Whole-genome phylogeny of Escherichia coli/Shigella group by feature frequency profiles (FFPs).”
- Wu S, Manber U (1992). “Fast Text Searching Allowing Errors.” *Communications of the ACM*, **35**, 83–91.

Affiliation:

Michael Hahsler
Computer Science
Lyle School of Engineering
Southern Methodist University
P.O. Box 750122
Dallas, TX 75275-0122
E-mail: mhahsler@lyle.smu.edu
URL: <http://lyle.smu.edu/~mhahsler>

Anurag Nagar
Computer Science
University of Texas at Dallas
E-mail: Anurag.Nagar@utdallas.edu