rMSA: Interface to Popular Multiple Sequence Alignment Tools

Michael Hahsler
Southern Methodist University

Anurag Nagar
University of Houston, Clear Lake

Abstract

Three 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 ne 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 optain 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 use:

```
R> ? rMSA_INSTALL
```

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.

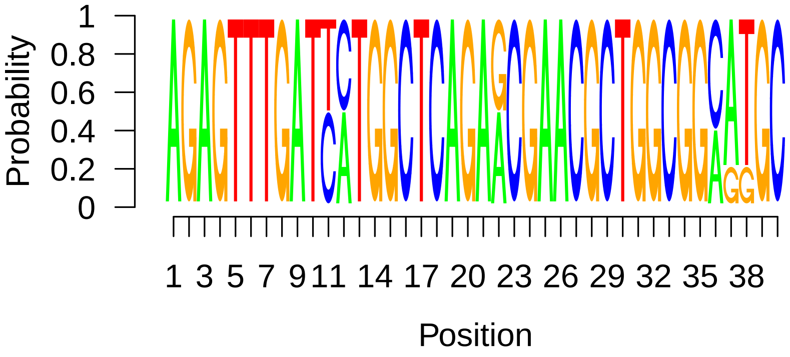
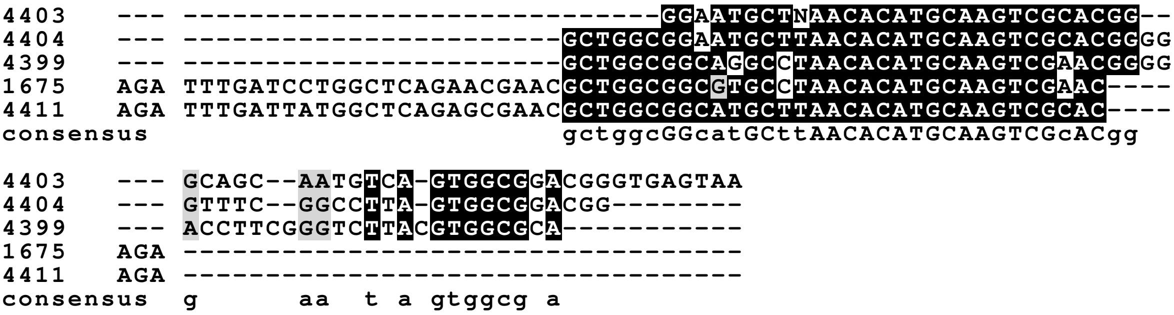


Figure 1: Sequence logo of alignment.



```
4403      --- -----------------------------GGAATGCTNAACACATGCAAGTCGCACGG--
4404      --- -----------------------GCTGGCGGAATGCTTAACACATGCAAGTCGCACGGGG
4399      --- -----------------------GCTGGCGGCAGGCCTAACACATGCAAGTCGAACGGGG
1675      AGA TTTGATCCTGGCTCAGAACGAACGCTGGCGGCGTGCCTAACACATGCAAGTCGAAC----
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.

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

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
```

```
  A RNAStringSet instance 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...UAGGUGUCUAGGCUAACC 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] ---------------------...AAGGUAGCCGUAGGGGAACC 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

  A AAStringSet instance of length 5
    width seq                                     names
[1]   170 MKKSWRRIWIFGLLFSIW...DVYYLEAPFFQGRKCGGT gi|340754543|ref|...
[2]   233 MYIIWKLLFFKGENVVEH...KEEEVISVVDDILKKRRE gi|340754544|ref|...
[3]   326 MKRSLSGIQPSGILHLGN...KKVQEAKEIVGLLGNIYR gi|340754545|ref|...
[4]   317 MKYYSGVDLGGTNTKIGL...VLGNEAGILGAAALFMLS gi|340754546|ref|...
[5]   337 MKKMGIILGALVLAAGLV...IVLVPSIGIDKENVAEYK gi|340754547|ref|...

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

AAMultipleAlignment with 5 rows and 358 columns
     aln                                        names
[1] ---MKKSWRRIWIFGLLFSIW...-------------------- gi|340754543|ref|...
[2] ---MYIIWKLLFFKGENVVEH...-------------------- gi|340754544|ref|...
[3] MKKMGIILGALVLAAGLVGCG...DKENVAEYK----------- gi|340754547|ref|...
[4] ---MKRSLSGIQPSGILHLGN...ASKKVQEAKEIVGLLGNIYR gi|340754545|ref|...
[5] ----MKYYSGVDLGGTNTKIG...-------------------- 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

  A DNAStringSet instance 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> ### align the sequences
R> al <- kalign(dna)
R> al

DNAMultipleAlignment with 5 rows and 1502 columns
     aln                                          names
[1] AGAGTTTGATCCTGGCTCAGA...--------CAAGGTAAC--C 1675 AB015560.1 d...
[2] G--------------------...--------TGGG-------G 4399 D14432.1 Rho...
[3] G--------------------...GGTAGCCGTAGGGGAAC--C 4403 X72908.1 Ros...
[4] G--------------------...--------TAGGCTAAC--C 4404 AF173825.1 A...
[5] AGAGTTTGATTATGGCTCAGA...--------CAAGGTAACCGT 4411 Y07647.2 Dre...
```

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

  A DNAStringSet instance 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 <- 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 (Katoh *et al.* 2002) is a similarity-based MSA technique using progressive and iterative refinement methods.

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

  A DNAStringSet instance 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
```

```
  A DNAStringSet instance of length 10
     width seq                                 names
 [1]   100 CCCGCAACCCCATAGAAA...AGAAAGATAAACAAACA 1
 [2]   100 CAAAAAAAACATAATTAA...TAGCACCTAGGGGCTCC 2
 [3]   100 CACCCAAATCAACCTCCA...CAAACGCATACCCACAA 3
 [4]   100 TCATAATCCTCAAAAAAA...AACATTCCCCATCCAAC 4
 [5]   100 ACCCACACACGTAGACCA...AACCCACCTACACACCC 5
 [6]   100 GGACGCGACATTCACCAC...AAATTCTGACACCCCAA 6
 [7]   100 AACAAGACAAGAATAACC...GAGACAGAACAAACACA 7
 [8]   100 CCAAAACACCTTAAAAAT...ACGACACACCCACGAGA 8
 [9]   100 GCAACAACACATCAAAGA...CTAAAATCCAAACCTGC 9
[10]   100 ATATAAACAAAAAAAATT...TAATAAACTACACATAG 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
```

```
  A DNAStringSet instance of length 10
     width seq                                 names
 [1]   100 CCGGGGCCTTAGGGTCGA...GGGGGGGGATTCTGGTT 1
 [2]   100 TTCTTCGGGAGTCGAGGA...AGAGGCGTAATCGGTTC 2
 [3]   100 CGGGCCCCCCTCAGGCGA...GTCTACCTATATTCTAT 3
 [4]   100 AAGGTAAGGGGGGGAGGG...ATTTAAGGGGGGAAGCG 4
 [5]   100 GGGCGTAGATAGAGTCTA...ATAACGACTATAGAGGG 5
 [6]   100 TCTTCCTCGCTAGTCCCT...TAGATCAGGAAGGGGGA 6
 [7]   100 TCCGAACTAGGCCCGGGG...GGAGGACCTCTATCTAG 7
 [8]   100 GAGGGATCTCCCCGATTA...GAGGGGAGGCGAATAGG 8
 [9]   100 AGCCCTCGTCCTCTCACT...GATTCCGTACGGGGGAT 9
[10]   100 GGACAGGTCCTTTAGGTA...GGATCGAGTCTTTCTCT 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
```

```
  A DNAStringSet instance of length 1
    width seq                                  names
[1]   100 GGCTTTAATCCGAGGCCA...CCTGTGGGGTGGGCACTG 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
```

```
  A DNAStringSet instance of length 10
     width seq                                    names
 [1]   100 GGCTTTAATCCGAGGCCA...TGTGGGGTGCGGCACTG 1_mutation_1
 [2]   101 GGCTTTAATCCGAGGCCA...TGTGGGGTGCGGCACTG 1_mutation_2
 [3]   100 GGCTTTATCCGAGGCCAC...CTGTGGGGTGGGCACTG 1_mutation_3
 [4]   101 GGCTTTAATCCGAGGCCA...CTGTGGGGTGGGCACTG 1_mutation_4
 [5]   102 GGCTTTAATCCGAGGCCA...CTGTGGGGTGGGCACTG 1_mutation_5
 [6]   100 GGCTTTAATCCGAGGCCA...CTGTGGGGTGGGCACTG 1_mutation_6
 [7]   101 GGCTTTAATCCGAGGCCA...CCTGTGGGGTGGGCATG 1_mutation_7
 [8]   100 GGCTTTAATCCGAGGCCA...CCTGTGGGGTGGCACTG 1_mutation_8
 [9]    99 GGCTTTAACCGAGGCCAC...CCTGTGGGGTGGGCAAT 1_mutation_9
[10]   100 GGCTTTAATCCGAGGCCA...CTGTGGGGTGGGCACTT 1_mutation_10
```

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

DNAMultipleAlignment with 11 rows and 109 columns
      aln                                        names
 [1] GGCTTTAATCCGAGGCCACC...ACCTGTGGGGTGCGGCACTG 1_mutation_1
 [2] GGCTTTAATCCGAGGCCACC...ACCTGTGGGGTGCGGCACTG 1_mutation_2
 [3] GGCTTTA-TCCGAGGCCACC...ACCTGTGGGGTG-GGCACTG 1_mutation_3
 [4] GGCTTTAATCCGAGGCCACC...ACCTGTGGGGTG-GGCACTG 1_mutation_5
 [5] GGCTTTAATCCGAGGCCACC...ACCTGTGGGGTG-GGCACTG 1
 [6] GGCTTTAATCCGAGGCCACC...ACCTGTGGGGTG-GGCACTG 1_mutation_4
 [7] GGCTTTAATCCGAGGCCACC...ACCTGTGGGGTG-G-CACTG 1_mutation_8
 [8] GGCTTTAATCCGAGGCCACC...ACCTGTGGGGTG-GGCACTG 1_mutation_6
 [9] GGCTTTAA-CCGAGGCCACC...ACCTGTGGGGTG-GGCAAT- 1_mutation_9
[10] GGCTTTAATCCGAGGCCACC...ACCTGTGGGGTG-GGCATG- 1_mutation_7
[11] GGCTTTAATCCGAGGCCACC...ACCTGTGGGGTG-GGCACTT 1_mutation_10
```

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−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

  A DNAStringSet instance of length 1
    width seq                                     names
[1]   100 GCTGTAGGTCGCCAAGTA...AGGACTACATTTTAGTGG 1

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

  A DNAStringSet instance of length 100
      width seq                                   names
  [1]   103 GCTGTAGTGTCGCCGAG...GGACTACATTTTAGTGG 1_mutation_1
  [2]    99 GCTGTAGGTCGCCAAGT...AGGACTACATTTTGTGG 1_mutation_2
  [3]   101 GCTGTAGGTCGCACAAG...GGACTACATTTTAGTGG 1_mutation_3
  [4]   102 GCTGTATGTCGCCAAGT...GGACTACATTTTAGTGG 1_mutation_4
  [5]    99 GCTGTAGGTGCACAAGT...GGACTACATTTTAGTGG 1_mutation_5
  ...   ... ...
 [96]   102 GCTGTGAGGTCGCCAAG...GACTACATTTTAGTTGG 1_mutation_96
 [97]   101 GCTGTAGGTCGCCAAGT...GGACTACATTTTAGTGG 1_mutation_97
 [98]   101 GCTGTGGTCGCCAAGTA...GGACTACATTTTAGTGG 1_mutation_98
 [99]   101 GCTGTAGGTCGCCAAGT...GGACTACATGTTAGTGG 1_mutation_99
[100]   100 GCATGTAGGTCGCCAGT...GGACTACATTTTAGTGG 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 aggree. We use Spearman's rank correlation coefficient to evaluate aggreement and show the resulting relationship in Figure 3.

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

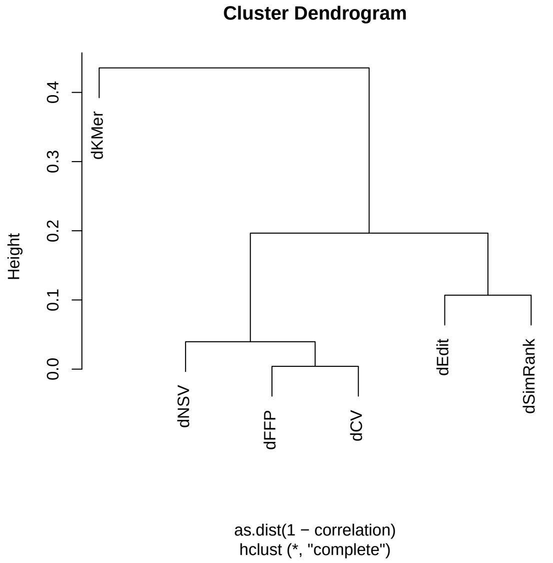


Figure 3: Correlation between different distance measures.

```
R> dCV <- distCV(ms)
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 0.8309 0.8034 0.8058 0.5762   0.8931
dNSV     0.8309 1.0000 0.9604 0.9612 0.6735   0.8434
dFFP     0.8034 0.9604 1.0000 0.9959 0.6127   0.8128
dCV      0.8058 0.9612 0.9959 1.0000 0.6059   0.8153
dKMer    0.5762 0.6735 0.6127 0.6059 1.0000   0.5646
dSimRank 0.8931 0.8434 0.8128 0.8153 0.5646   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 assumtions 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",
```

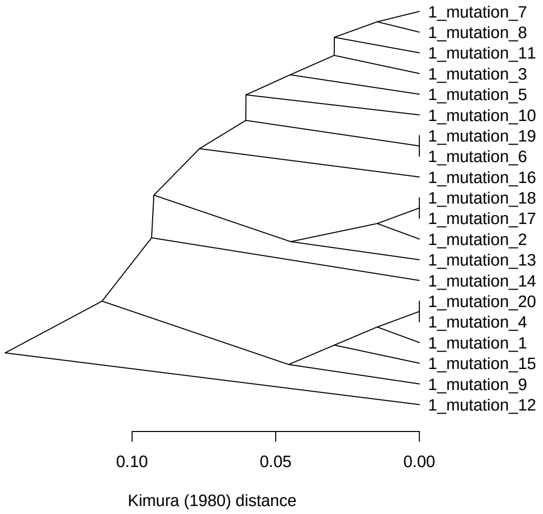


Figure 4: Evolutionaty relationship between 20 mutations based on Kimura (1980) distance.

```
+   xlab="Kimura (1980) distance")
R> par(par.old)
```

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.

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
Engineering Management, Information, and Systems
Lyle School of Engineering
Southern Methodist University
P.O. Box 750123
Dallas, TX 75275-0123
E-mail: mhahsler@lyle.smu.edu
URL: http://lyle.smu.edu/~mhahsler

Anurag Nagar
Computer Science and Engineering
Lyle School of Engineering
Southern Methodist University
P.O. Box 750122
Dallas, TX 75275-0122
E-mail: anagar@smu.edu