BiostringsTools: Interface to Tools for Biostrings (alignment, classification, database)

Michael Hahsler
Southern Methodist University

Anurag Nagar
Southern Methodist University

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 (clustalw, kalign), classification (RDP), sequence retrieval (BLAST) as well as database driven sequence management for 16S rRNA.

Keywords: bioinformatics, Bioconductor, biostrings, sequence alignment, sequence classification, sequence management.

---

1. Introduction

There are many tools available for sequence alignment and classification. Some tools are: BAlibase (Smith and Waterman 1981), BLAST (Altschul, Gish, Miller, Myers, and Lipman 1990), 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, these tools have a command-line interface and the input and output data is stored in files using various formats. Also the parameters supplied to the command-line interface are different. 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 important infrastructure to handle and manipulate bioinformatics data. The **Biostrings** package in particular provides infrastructure for DNA, RNA and protein sequences as well as (multiple) alignments. Also algorithms for sequence alignment are included. However, for multiple sequence alignment using BLAST the user needs to export the data into a file and then run the needed tool manually and re-import the results. Also, **Biostrings** stores sets of sequences in memory and does not directly support storing and querying classification information.

In **BiostringsTools** 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 is needed. Currently we interface *clustalw, kalign, RDP* and *clustalw*. **BiostringsTools** also provides database backed sequence management where large amounts of sequences and classification information can be stored and used for selective and efficient sequence retrieval.

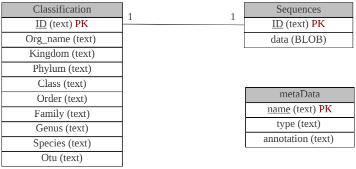


Figure 1: Entity Relationship diagram of GenDB

2. Installing Third-Party Software

BiostringsTools is designed to make installation of third-party software (RDP, clustal, kalign, MAFFT, BLAST and boxshade) easy by providing `BiostringsTools_Software_Wizard()`. With this wizard the needed software can be installed individually. This is shown in the example section.

Additional software is stored in a subdirectory of the home directory called `BiostringsTools`.

3. GenDB: Sequence storage an management

BiostringsTools provides a databases (GenDB) which can be used for efficient storage and retrieval of genetic sequences. By default the light-weight SQLite database is used, but any other compatible database such as mySQL or Oracle can also be used. Figure 1 shows the basic table layout of a GenDB instance with a table containing classification information, a table containing the sequence information and a meta data table. Each sequence we will have an entry in the classification table and an corresponding entry in the sequence table. The tables are connected by a unique sequence ID as the primary key.

GenDB is easy to use. First, we load the library into the R environment.

```
R> library(BiostringsTools)
```

To start we need to create an empty GenDB to store and organize sequences.

```
R> db<-createGenDB("example.sqlite")
R> db

Object of class GenDB with 0 sequences
DB File: example.sqlite
Tables:
```

The above command creates an empty database with a table structure similar to Figure 1 and stores it in the file example.sqlite. If a GenDB already exists, then it can be opened using `openGenDB()`.

The next step is to import sequences into the database by reading FASTA files. This is accomplished by function `addSequences()`. This function automatically extracts the classification information from the FASTA file's description lines. The default is to expect classification in the format used by the Greengenes project, however other meta data readers can be implemented (see manual page for addSequences).

The command below uses a FASTA file provided by the package, hence we use `system.file()` instead of just a string with the file name.

```
R> addSequences(db,
+    system.file("examples/Firmicutes.fasta", package="BiostringsTools"))

Read 100 sequences. Added 100 sequences.
```

After inserting the sequences, various querying and limiting functions can be used to check the data and obtain a subset of the sequences. To get a count of the number of sequences in the database, the function `nSequences()` can be used.

```
R> nSequences(db)

[1] 100
```

The function `getSequences()` returns the sequences as a vector. In the following example we get all sequences in the database and then show the first 50 bases of the first sequence.

```
R> s <- getSequences(db)
R> s

  A DNAStringSet instance of length 100
      width seq                                    names
  [1]  1521 TTTGATCCTGGCTCAGG...CGGCTGGATCACCTCCT 1250
  [2]  1392 ACGGGTGAGTAACGCGT...TTGGGGTGAAGTCGTAA 13651
  [3]  1384 TAGTGGCGGACGGGTGA...TCGAATTTGGGTCAAGT 13652
  [4]  1672 GGCGTGCCTAACACATG...TGTAAACACGACTTCAT 13654
  [5]  1386 ATCTCACCTCTCAATAG...CGAAGGTGGGGTTGGTG 13655
  ...   ... ...
 [96]  1446 ATGCAAGTCGAACGGGG...GGGGCCGATGATTGGGG 13857
 [97]  1511 ATCCTGGCTCAGGACGA...AGTCGTAACAAGGTAGC 13858
 [98]  1544 ATCCTGGCTCAGGACGA...GGTGGATCACCTCCTTC 13860
 [99]  1482 GGACGAACGCTGGCGGC...GCCGATGATTGGGGTGA 13861
[100]  1485 GACGAACGCTGGCGGCG...GAAGTCGTAACAAGGTA 13862

R> length(s)

[1] 100

R> s[[1]]
```

```
  1521-letter "DNAString" instance
seq: TTTGATCCTGGCTCAGGACGAACGCTGGCGG...TGTACCGGAAGGTGCGGCTGGATCACCTCCT
```

```
R> substr(s[[1]], 1, 50)
```

```
  50-letter "DNAString" instance
seq: TTTGATCCTGGCTCAGGACGAACGCTGGCGGCGTGCCTAATGCATGCAAG
```

Sequences in the database can also be filtered using classification information. For example, we can get all sequences of the genus name "Desulfosporomusa" by specifying rank and name.

```
R> s <- getSequences(db, rank="Genus", name="Desulfosporomusa")
R> s
```

```
  A DNAStringSet instance of length 0
```

To obtain a single sequence, getSequences can be used with rank equal to "id" and supplying the sequence's greengenes ID as the name.

```
R> s <- getSequences(db, rank="id", name="1250")
R> s
```

```
  A DNAStringSet instance of length 1
    width seq                                     names
[1]  1521 TTTGATCCTGGCTCAGGA...GCGGCTGGATCACCTCCT 1250
```

The database also stores a classification hierarchy. We can obtain the classification hierarchy used in the database with `getTaxonomyNames()`.

```
R> getTaxonomyNames(db)
```

```
 [1] "Kingdom"  "Phylum"   "Class"    "Order"    "Family"   "Genus"
 [7] "Species"  "Otu"      "Org_name" "Id"
```

To obtain all unique names stored in the database for a given rank we can use `getRank()`.

```
R> getRank(db, rank="Order")
```

```
[1] "Thermoanaerobacterales" "Clostridiales"
```

The 100 sequences in our example data base contain organisms from different orders. We can obtain the rank name for each sequence individually by using `all=TRUE` or count how many sequences we have for each genus using `count=TRUE`.

```
R> getRank(db, rank="Genus", all=TRUE)
```

```
 [1] Coprothermobacter
 [2] Desulfotomaculum; Unclassified
 [3] Desulfotomaculum
 [4] Desulfotomaculum; Unclassified
 [5] Desulfotomaculum
 [6] Desulfotomaculum; Unclassified
 [7] Desulfotomaculum; Unclassified
 [8] Desulfotomaculum; Unclassified
 [9] Desulfotomaculum
[10] Pelotomaculum; Unclassified
[11] Desulfotomaculum; Unclassified
[12] Desulfotomaculum; Unclassified
[13] Pelotomaculum; Unclassified
[14] Desulfotomaculum; Unclassified
[15] Desulfotomaculum; Unclassified
[16] Desulfotomaculum; Unclassified
[17] Desulfotomaculum
[18] Pelotomaculum; Unclassified
[19] Desulfotomaculum
[20] Desulfotomaculum
[21] Desulfotomaculum
[22] Desulfotomaculum
[23] Desulfotomaculum; Unclassified
[24] Desulfotomaculum
[25] Pelotomaculum; Unclassified
[26] Syntrophomonas; Unclassified
[27] Syntrophomonas; Unclassified
[28] Syntrophomonas
[29] Syntrophomonas; Unclassified
[30] Syntrophomonas; Unclassified
[31] unknown
[32] Syntrophomonas; Unclassified
[33] Moorella
[34] Moorella
[35] Moorella
[36] Moorella
[37] Thermacetogenium
[38] Thermaerobacter; Unclassified
[39] Carboxydothermus; Unclassified
[40] Carboxydothermus; Unclassified
[41] Thermoanaerobacterium
[42] Thermoanaerobacterium
[43] Thermoanaerobacterium; Unclassified
[44] Thermoanaerobacterium; Unclassified
[45] Thermoanaerobacterium
[46] Thermoanaerobacterium
[47] Thermoanaerobacterium
```

```
[48] Thermoanaerobacterium
[49] Thermoanaerobacter; Unclassified
[50] Thermoanaerobacter; Unclassified
[51] Thermoanaerobacter; Unclassified
[52] Thermoanaerobacter; Unclassified
[53] Thermoanaerobacter; Unclassified
[54] Thermoanaerobacter
[55] Thermoanaerobacter; Unclassified
[56] Thermoanaerobacter; Unclassified
[57] Thermoanaerobacter; Unclassified
[58] Thermoanaerobacter; Unclassified
[59] Selenomonas
[60] Selenomonas
[61] Selenomonas
[62] Selenomonas; Unclassified
[63] Selenomonas; Unclassified
[64] Mitsuokella
[65] Selenomonas
[66] Selenomonas
[67] Selenomonas
[68] unknown
[69] Selenomonas
[70] Veillonella
[71] Veillonella
[72] Veillonella; Unclassified
[73] Veillonella
[74] Veillonella; Unclassified
[75] Dialister
[76] Dialister
[77] Dialister
[78] Desulfosporomusa; Unclassified
[79] Desulfosporomusa; Unclassified
[80] unknown
[81] unknown
[82] Desulfosporomusa; Unclassified
[83] Thermosinus; Unclassified
[84] Thermosinus; Unclassified
[85] unknown
[86] Desulfosporomusa; Unclassified
[87] Desulfosporomusa; Unclassified
[88] Desulfosporomusa; Unclassified
[89] Desulfosporomusa; Unclassified
[90] unknown
[91] unknown
[92] Acidaminococcus
[93] Acidaminococcus
[94] unknown
```

```
 [95] unknown
 [96] unknown
 [97] Phascolarctobacterium
 [98] Phascolarctobacterium
 [99] unknown
[100] unknown
25 Levels: Acidaminococcus ... Veillonella; Unclassified
```

```
R> getRank(db, rank="Genus", count=TRUE)
```

```
                           unknown
                                12
     Desulfotomaculum; Unclassified
                                11
                  Desulfotomaculum
                                 9
   Thermoanaerobacter; Unclassified
                                 9
     Desulfosporomusa; Unclassified
                                 7
                       Selenomonas
                                 7
             Thermoanaerobacterium
                                 6
       Syntrophomonas; Unclassified
                                 5
                          Moorella
                                 4
        Pelotomaculum; Unclassified
                                 4
                         Dialister
                                 3
                       Veillonella
                                 3
                   Acidaminococcus
                                 2
     Carboxydothermus; Unclassified
                                 2
             Phascolarctobacterium
                                 2
         Selenomonas; Unclassified
                                 2
Thermoanaerobacterium; Unclassified
                                 2
         Thermosinus; Unclassified
                                 2
         Veillonella; Unclassified
```

```
                              2
              Coprothermobacter
                              1
                    Mitsuokella
                              1
                 Syntrophomonas
                              1
               Thermacetogenium
                              1
   Thermaerobacter; Unclassified
                              1
             Thermoanaerobacter
                              1
```

This information can be easily turned into a barplot showing the abundance of different orders in the data database (see Figure 3).

```
R> oldpar <- par(mar=c(12,5,5,5)) ### make space for labels
R> barplot(sort(
+     getRank(db, rank="Genus", count=TRUE, removeUnknown=TRUE),
+     decreasing=TRUE), las=2)
R> par(oldpar)
```

Filtering also works for `getRank()`. For example, we can find the genera within the order "Thermoanaerobacterales".

```
R> getRank(db, rank="Gen",
+     whereRank="Ord", whereName="Thermo%")
```

```
[1] "Coprothermobacter"
[2] "Moorella"
[3] "Thermacetogenium"
[4] "Carboxydothermus; Unclassified"
[5] "Thermoanaerobacter; Unclassified"
[6] "Thermoanaerobacter"
```

Note that partial matching is performed for the ranks (i.e., from "Gen" to Genus and "Ord" to Order) and also for the name from "Thermo%" to Thermoanaerobacterales. Partial matching is available for ranks and names in most operations in **BiostringsTools**.

We can also get the complete classification hierarchy for different ranks down to individual sequences. In the following we get the classification hierarchy for genus Thermaerobacter, then all orders matching Therm and then for a list of names.

```
R> getHierarchy(db, rank="Genus", name="Thermaerobacter")
```

```
 [1] Kingdom  Phylum   Class    Order    Family   Genus    Species
 [8] Otu      Org_name Id
<0 rows> (or 0-length row.names)
```

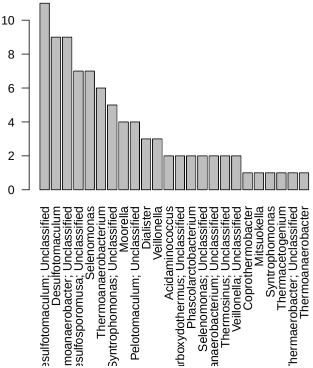


Figure 2: Abundance of different orders in the database.

```
R> getHierarchy(db, rank="Genus", name="Therm%")

    Kingdom     Phylum      Class                  Order
1  Bacteria Firmicutes Clostridia Thermoanaerobacterales
2  Bacteria Firmicutes Clostridia          Clostridiales
3  Bacteria Firmicutes Clostridia          Clostridiales
4  Bacteria Firmicutes Clostridia          Clostridiales
5  Bacteria Firmicutes Clostridia          Clostridiales
6  Bacteria Firmicutes Clostridia          Clostridiales
7  Bacteria Firmicutes Clostridia          Clostridiales
8  Bacteria Firmicutes Clostridia          Clostridiales
9  Bacteria Firmicutes Clostridia          Clostridiales
10 Bacteria Firmicutes Clostridia          Clostridiales
11 Bacteria Firmicutes Clostridia Thermoanaerobacterales
12 Bacteria Firmicutes Clostridia Thermoanaerobacterales
13 Bacteria Firmicutes Clostridia Thermoanaerobacterales
14 Bacteria Firmicutes Clostridia Thermoanaerobacterales
15 Bacteria Firmicutes Clostridia Thermoanaerobacterales
16 Bacteria Firmicutes Clostridia Thermoanaerobacterales
17 Bacteria Firmicutes Clostridia Thermoanaerobacterales
18 Bacteria Firmicutes Clostridia Thermoanaerobacterales
19 Bacteria Firmicutes Clostridia Thermoanaerobacterales
20 Bacteria Firmicutes Clostridia Thermoanaerobacterales
21 Bacteria Firmicutes Clostridia          Clostridiales
22 Bacteria Firmicutes Clostridia          Clostridiales
                                               Family
1                           Thermoanaerobacteraceae
2                                  Sulfobacillaceae
3  Thermoanaerobacterales Family III. Incertae Sedis
4  Thermoanaerobacterales Family III. Incertae Sedis
5  Thermoanaerobacterales Family III. Incertae Sedis
6  Thermoanaerobacterales Family III. Incertae Sedis
7  Thermoanaerobacterales Family III. Incertae Sedis
8  Thermoanaerobacterales Family III. Incertae Sedis
9  Thermoanaerobacterales Family III. Incertae Sedis
10 Thermoanaerobacterales Family III. Incertae Sedis
11                          Thermoanaerobacteraceae
12                          Thermoanaerobacteraceae
13                          Thermoanaerobacteraceae
14                          Thermoanaerobacteraceae
15                          Thermoanaerobacteraceae
16                          Thermoanaerobacteraceae
17                          Thermoanaerobacteraceae
18                          Thermoanaerobacteraceae
19                          Thermoanaerobacteraceae
20                          Thermoanaerobacteraceae
21                                  Veillonellaceae
```

```
22                                 Veillonellaceae
                             Genus
1                 Thermacetogenium
2       Thermaerobacter; Unclassified
3              Thermoanaerobacterium
4              Thermoanaerobacterium
5  Thermoanaerobacterium; Unclassified
6  Thermoanaerobacterium; Unclassified
7              Thermoanaerobacterium
8              Thermoanaerobacterium
9              Thermoanaerobacterium
10             Thermoanaerobacterium
11   Thermoanaerobacter; Unclassified
12   Thermoanaerobacter; Unclassified
13   Thermoanaerobacter; Unclassified
14   Thermoanaerobacter; Unclassified
15   Thermoanaerobacter; Unclassified
16                Thermoanaerobacter
17   Thermoanaerobacter; Unclassified
18   Thermoanaerobacter; Unclassified
19   Thermoanaerobacter; Unclassified
20   Thermoanaerobacter; Unclassified
21          Thermosinus; Unclassified
22          Thermosinus; Unclassified
                             Species  Otu
1                            unknown 2273
2                            unknown 2189
3  Thermoanaerobacterium saccharolyticum 2200
4  Thermoanaerobacterium saccharolyticum 2200
5                            unknown 2199
6                            unknown 2199
7  Thermoanaerobacterium saccharolyticum 2200
8  Thermoanaerobacterium saccharolyticum 2200
9  Thermoanaerobacterium saccharolyticum 2200
10 Thermoanaerobacterium saccharolyticum 2200
11                           unknown 2274
12                           unknown 2274
13                           unknown 2274
14                           unknown 2274
15                           unknown 2274
16         Thermoanaerobacter mathranii 2276
17                           unknown 2274
18                           unknown 2274
19                           unknown 2274
20                           unknown 2274
21                           unknown 2228
22                           unknown 2228
```

```
                                                                  Org_name
1                               AB020336.1_Thermacetogenium_phaeum_str._PB
2                                   AB011495.1_Thermaerobacter_marianensis
3                  L09172.1_Thermoanaerobacterium_xylanolyticum_str._LXIIT
4               L09171.1_Thermoanaerobacterium_thermosulfurigenes_str._4BT
5                            U75993.1_Thermoanaerobacterium_zeae_str._mel2
6                       U40229.1_Thermoanaerobacterium_polysaccharolyticum
7        L09170.1_Thermoanerobacter_brockii_subsp._lactiethylicus_str._ZE-1
8               L09169.1_Thermoanaerobacterium_saccharolyticum_str._B6A-RI
9                     X76743.1_Clostridium_thermoamylolyticum_str._DSM2335
10            X93359.1_Thermoanaerobacterium_aotearoense_str._JW/SL-NZ613T
11                                       L09160.1_Thermoanaerobacter_kivui
12                        X92513.1_Thermoanaerobacter_wiegelii_str._Rt8.B1
13                            U51198.1_Thermoanaerobacter_sp._str._AB11_Ad
14                     Y16940.1_Thermoanaerobacter_sulfurophilus_str._L-64
15                    L09167.1_Thermoanaerobacter_thermocopriae_str._JT-3T
16                           Y11279.1_Thermoanaerobacter_mathranii_str._A3
17             L09164.1_Thermoanaerobacter_ethanolicus_ATCC_33223_str._39E
18                                     L09165.1_Thermoanaerobacter_brockii
19                                       L09166.1_Thermoanaerobacter_finii
20 U14330.1_Thermoanerobacter_brockii_subsp._lactiethylicus_str._SEBR_5268
21           AJ009459.1_anaerobic_TCB-transforming_consortium_clone_SJA-29
22                    AJ009486.1_TCB-transforming_consortium_clone_SJA-112
      Id
1  13717
2  13721
3  13755
4  13757
5  13758
6  13759
7  13760
8  13761
9  13762
10 13763
11 13765
12 13766
13 13767
14 13768
15 13780
16 13781
17 13789
18 13790
19 13792
20 13793
21 13840
22 13842
```

```
R> getHierarchy(db, rank="Genus", name=c("Acid%", "Thermo%"))

    Kingdom     Phylum      Class                  Order
1  Bacteria Firmicutes Clostridia          Clostridiales
2  Bacteria Firmicutes Clostridia          Clostridiales
3  Bacteria Firmicutes Clostridia          Clostridiales
4  Bacteria Firmicutes Clostridia          Clostridiales
5  Bacteria Firmicutes Clostridia          Clostridiales
6  Bacteria Firmicutes Clostridia          Clostridiales
7  Bacteria Firmicutes Clostridia          Clostridiales
8  Bacteria Firmicutes Clostridia          Clostridiales
9  Bacteria Firmicutes Clostridia Thermoanaerobacterales
10 Bacteria Firmicutes Clostridia Thermoanaerobacterales
11 Bacteria Firmicutes Clostridia Thermoanaerobacterales
12 Bacteria Firmicutes Clostridia Thermoanaerobacterales
13 Bacteria Firmicutes Clostridia Thermoanaerobacterales
14 Bacteria Firmicutes Clostridia Thermoanaerobacterales
15 Bacteria Firmicutes Clostridia Thermoanaerobacterales
16 Bacteria Firmicutes Clostridia Thermoanaerobacterales
17 Bacteria Firmicutes Clostridia Thermoanaerobacterales
18 Bacteria Firmicutes Clostridia Thermoanaerobacterales
19 Bacteria Firmicutes Clostridia          Clostridiales
20 Bacteria Firmicutes Clostridia          Clostridiales
21 Bacteria Firmicutes Clostridia          Clostridiales
22 Bacteria Firmicutes Clostridia          Clostridiales
                                                 Family
1  Thermoanaerobacterales Family III. Incertae Sedis
2  Thermoanaerobacterales Family III. Incertae Sedis
3  Thermoanaerobacterales Family III. Incertae Sedis
4  Thermoanaerobacterales Family III. Incertae Sedis
5  Thermoanaerobacterales Family III. Incertae Sedis
6  Thermoanaerobacterales Family III. Incertae Sedis
7  Thermoanaerobacterales Family III. Incertae Sedis
8  Thermoanaerobacterales Family III. Incertae Sedis
9                           Thermoanaerobacteraceae
10                          Thermoanaerobacteraceae
11                          Thermoanaerobacteraceae
12                          Thermoanaerobacteraceae
13                          Thermoanaerobacteraceae
14                          Thermoanaerobacteraceae
15                          Thermoanaerobacteraceae
16                          Thermoanaerobacteraceae
17                          Thermoanaerobacteraceae
18                          Thermoanaerobacteraceae
19                                  Veillonellaceae
20                                  Veillonellaceae
21                                  Veillonellaceae
```

```
22                                   Veillonellaceae
                                 Genus
1                Thermoanaerobacterium
2                Thermoanaerobacterium
3  Thermoanaerobacterium; Unclassified
4  Thermoanaerobacterium; Unclassified
5                Thermoanaerobacterium
6                Thermoanaerobacterium
7                Thermoanaerobacterium
8                Thermoanaerobacterium
9     Thermoanaerobacter; Unclassified
10    Thermoanaerobacter; Unclassified
11    Thermoanaerobacter; Unclassified
12    Thermoanaerobacter; Unclassified
13    Thermoanaerobacter; Unclassified
14                  Thermoanaerobacter
15    Thermoanaerobacter; Unclassified
16    Thermoanaerobacter; Unclassified
17    Thermoanaerobacter; Unclassified
18    Thermoanaerobacter; Unclassified
19           Thermosinus; Unclassified
20           Thermosinus; Unclassified
21                     Acidaminococcus
22                     Acidaminococcus
                                 Species  Otu
1  Thermoanaerobacterium saccharolyticum 2200
2  Thermoanaerobacterium saccharolyticum 2200
3                                unknown 2199
4                                unknown 2199
5  Thermoanaerobacterium saccharolyticum 2200
6  Thermoanaerobacterium saccharolyticum 2200
7  Thermoanaerobacterium saccharolyticum 2200
8  Thermoanaerobacterium saccharolyticum 2200
9                                unknown 2274
10                               unknown 2274
11                               unknown 2274
12                               unknown 2274
13                               unknown 2274
14          Thermoanaerobacter mathranii 2276
15                               unknown 2274
16                               unknown 2274
17                               unknown 2274
18                               unknown 2274
19                               unknown 2228
20                               unknown 2228
21            Acidaminococcus fermentans 2203
22            Acidaminococcus fermentans 2203
```

```
                                                                  Org_name
1                  L09172.1_Thermoanaerobacterium_xylanolyticum_str._LXIIT
2               L09171.1_Thermoanaerobacterium_thermosulfurigenes_str._4BT
3                            U75993.1_Thermoanaerobacterium_zeae_str._mel2
4                       U40229.1_Thermoanaerobacterium_polysaccharolyticum
5        L09170.1_Thermoanerobacter_brockii_subsp._lactiethylicus_str._ZE-1
6               L09169.1_Thermoanaerobacterium_saccharolyticum_str._B6A-RI
7                     X76743.1_Clostridium_thermoamylolyticum_str._DSM2335
8             X93359.1_Thermoanaerobacterium_aotearoense_str._JW/SL-NZ613T
9                                        L09160.1_Thermoanaerobacter_kivui
10                        X92513.1_Thermoanaerobacter_wiegelii_str._Rt8.B1
11                            U51198.1_Thermoanaerobacter_sp._str._AB11_Ad
12                     Y16940.1_Thermoanaerobacter_sulfurophilus_str._L-64
13                    L09167.1_Thermoanaerobacter_thermocopriae_str._JT-3T
14                           Y11279.1_Thermoanaerobacter_mathranii_str._A3
15             L09164.1_Thermoanaerobacter_ethanolicus_ATCC_33223_str._39E
16                                     L09165.1_Thermoanaerobacter_brockii
17                                       L09166.1_Thermoanaerobacter_finii
18 U14330.1_Thermoanerobacter_brockii_subsp._lactiethylicus_str._SEBR_5268
19           AJ009459.1_anaerobic_TCB-transforming_consortium_clone_SJA-29
20                    AJ009486.1_TCB-transforming_consortium_clone_SJA-112
21                      X78017.1_Acidaminococcus_fermentans_str._DSM_20731
22                             X77951.1_Acidaminococcus_fermentans_str._AO
      Id
1  13755
2  13757
3  13758
4  13759
5  13760
6  13761
7  13762
8  13763
9  13765
10 13766
11 13767
12 13768
13 13780
14 13781
15 13789
16 13790
17 13792
18 13793
19 13840
20 13842
21 13852
22 13853
```

To get individual sequences we can use again the unique sequence id.

```
R> getHierarchy(db, rank="id", name="1250")

   Kingdom     Phylum      Class                  Order
1 Bacteria Firmicutes Clostridia Thermoanaerobacterales
               Family             Genus Species  Otu
1 Thermodesulfobiaceae Coprothermobacter unknown 2281
                                                Org_name   Id
1 X69335.1_Coprothermobacter_proteolyticus_str._ATCC_35245 1250
```

Finally, we can close a GenDB after we are done working with it. The database can later be reopened using `openGenDB()`.

```
R> closeGenDB(db)
```

To permanently remove the database we need to delete the file (for SQLite databases) or remove the database using the administrative tool for the database management system.

```
R> unlink("example.sqlite")
```

4. Multiple Sequence Alignment

Multiple Sequence Alignment (MSA) involves comparing and aligning more than two sequences to each other and also possibly to many others in a sequence database. The aim is to discover regions of high similarity for all the sequences taken together. The sequences are generally related such as those from the same species or same phylum.

Although, computationally complex, MSA is quite often what biologists need to identify and characterize sequences from a given group. Sequences might also share an evolutionary relationship, such as having a common ancestor. Such sequences are said to be homologous. Similarly, biologists might be interested in the similarity of genes from different organisms and want to compare 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. Various algorithms 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**?**.

Current methods for Clustal are through an online interface through the The European Bioinformatics Institute website at http://www.ebi.ac.uk/Tools/msa/clustalw2/ or through a webservice also at the same website. There is no current tool that can be run through the command line for a batch of sequences. Our package addresses this need by providing an interface that can be used for DNA Sequences.

The **BiostringsTools** provides a rich set of functionality for MSA operations including visualization options. The commands below will illustrate that in detail.

4.1. clustalw

Install the clustal software. This has to be done only once.

```
R> BiostringsTools_Software_Wizard(clustal = TRUE)

BiostringsTools Software Installation Wizard for LINUX

clustalw ... installed.
```

We read an example FASTA file with DNA, take the first 60 nucleotides and run clustal.

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

Plot produces the sequence logo shown in Figure 3.

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

Boxshade can also be used for producing a pdf of the alignment. Figure 4 shows the result.

```
R> BiostringsTools_Software_Wizard(boxshade = TRUE)

BiostringsTools Software Installation Wizard for LINUX

boxshade ... installed.

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

Clustal can also be used for RNA and protein sequences.

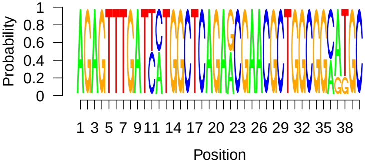
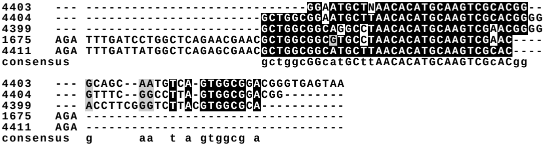


Figure 3: Sequence logo of alignment.



```
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 4: Representation of a DNA multiple alignment using boxshade.

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

4.2. 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> BiostringsTools_Software_Wizard(kalign = TRUE)


BiostringsTools Software Installation Wizard for LINUX

kalign ... installed.

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

4.3. MUSCLE

```
R> BiostringsTools_Software_Wizard(muscle = TRUE)
```

```
BiostringsTools Software Installation Wizard for LINUX

MUSCLE ... installed.

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

4.4. MAFFT

```
R> BiostringsTools_Software_Wizard(mafft = TRUE)

BiostringsTools Software Installation Wizard for LINUX

mafft ... installed.

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

5. Classification with RDP

The Ribosomal Database Project (RDP) provides various tools and services to the scientific community for data related to 16S rRNA sequences. Among other tools, it provides a hierarchical browser and a classifier that can be used to assign sequences to taxonomies. The classifier uses a Naive Bayesian approach to quickly and accurately classify sequences. The classifier uses an alignment-free approach and compares the word frequency distribution with word size of 8Wang, Garrity, Tiedje, and Cole (2007).

The RDP classifier needs to be trained first before it can be used. The default classifier comes trained with sequences from the microbial 16S rRNA gene.

First, we install RDP.

```
R> BiostringsTools_Software_Wizard(rdp = TRUE)

BiostringsTools Software Installation Wizard for LINUX

RDP ... installed.
```

5.1. Using the default RDP classifier

For this example we load some test sequences. we also shorten the names to only the sequence ID.

```
R> seq <- readRNAStringSet(system.file("examples/RNA_example.fasta",
+         package="BiostringsTools"))
R> names(seq) <-  sapply(strsplit(names(seq), " "), "[", 1)
R> seq

  A RNAStringSet instance of length 5
    width seq                                          names
[1]  1481 AGAGUUUGAUCCUGGCUC...AGUCGUAACAAGGUAACC 1675
[2]  1404 GCUGGCGGCAGGCCUAAC...UAAGGUCAGCGACUGGGG 4399
[3]  1426 GGAAUGCUNAACACAUGC...GGUAGCCGUAGGGGAACC 4403
[4]  1362 GCUGGCGGAAUGCUUAAC...UAGGUGUCUAGGCUAACC 4404
[5]  1458 AGAGUUUGAUUAUGGCUC...UCGUAACAAGGUAACCGU 4411
```

Next, we apply RDP with the default training set.

```
R> predict(rdp(), seq)

     rootrank   domain         phylum               class
1675      Root Bacteria Proteobacteria Deltaproteobacteria
4399      Root Bacteria Proteobacteria Alphaproteobacteria
4403      Root Bacteria Proteobacteria Alphaproteobacteria
4404      Root Bacteria Proteobacteria Alphaproteobacteria
4411      Root Bacteria Proteobacteria Alphaproteobacteria
                order              family       genus
1675             <NA>                <NA>        <NA>
4399 Rhodospirillales Rhodospirillaceae Rhodovibrio
4403 Rhodospirillales  Acetobacteraceae Roseococcus
4404 Rhodospirillales  Acetobacteraceae Roseococcus
4411 Rhodospirillales  Acetobacteraceae        <NA>
```

5.2. Training a custom RDP classifier

RDP can be trained using `trainRDP()`.

```
R> trainingSequences <- readDNAStringSet(
+     system.file("examples/trainingSequences.fasta", package="BiostringsTools"))
R> customRDP <- trainRDP(trainingSequences, dir = "myRDP")
R> customRDP

RDPClassifier
Location: /home/hahsler/BiostringsTools/myRDP

R> testSequences <- readDNAStringSet(
+     system.file("examples/testSequences.fasta", package="BiostringsTools"))
R> predict(customRDP, testSequences)

      rootrank  Kingdom     Phylum      Class         Order
13811     Root Bacteria Firmicutes Clostridia Clostridiales
13813     Root Bacteria Firmicutes Clostridia Clostridiales
13678     Root Bacteria Firmicutes Clostridia Clostridiales
13755     Root Bacteria Firmicutes Clostridia Clostridiales
13661     Root Bacteria Firmicutes Clostridia Clostridiales
                                              Family
13811                                Veillonellaceae
13813                                Veillonellaceae
13678                                 Peptococcaceae
13755 Thermoanaerobacterales Family III. Incertae Sedis
13661                                 Peptococcaceae
                    Genus
```

```
13811           Selenomonas
13813           Selenomonas
13678       Desulfotomaculum
13755 Thermoanaerobacterium
13661       Desulfotomaculum
```

The clustom classifier is stored on disc and can be recalled anytime using `rdp()`.

```
R> customRDP <- rdp(dir = "myRDP")
```

To permanently remove the classifier use `removeRDP()`.

```
R> removeRDP(customRDP)
```

6. Sequence Retrieval with BLAST

First we install BLAST.

```
R> BiostringsTools_Software_Wizard(blast = TRUE)

BiostringsTools Software Installation Wizard for LINUX

BLAST ... installed.
```

Next, we need a BLAST database. The installation wizard can install the default 16S rRNA database into the BiostringsTools folder. Now, we can initialize BLAST with the database.

```
R> BiostringsTools_Software_Wizard(blast16S = TRUE)

BiostringsTools Software Installation Wizard for LINUX

16SMicrobialDB ... installed.

R> bl <- blast(db="~/BiostringsTools/16SMicrobialDB/16SMicrobial")
R> bl

BLAST Database
Location: /home/hahsler/BiostringsTools/16SMicrobialDB/16SMicrobial
Database: 16S Microbial Sequences
        14,868 sequences; 21,718,706 total bases

Date: Jun 2, 2014  12:00 AM        Longest sequence: 2,211 bases

Volumes:
        /home/hahsler/BiostringsTools/16SMicrobialDB/16SMicrobial
```

We load again a few sequences.

```
R> seq <- readRNAStringSet(system.file("examples/RNA_example.fasta",
+          package="BiostringsTools"))
R> ## shorten names
R> names(seq) <-  sapply(strsplit(names(seq), " "), "[", 1)
R> seq

  A RNAStringSet instance of length 5
    width seq                                     names
[1]  1481 AGAGUUUGAUCCUGGCUC...AGUCGUAACAAGGUAACC 1675
[2]  1404 GCUGGCGGCAGGCCUAAC...UAAGGUCAGCGACUGGGG 4399
[3]  1426 GGAAUGCUNAACACAUGC...GGUAGCCGUAGGGGAACC 4403
[4]  1362 GCUGGCGGAAUGCUUAAC...UAGGUGUCUAGGCUAACC 4404
[5]  1458 AGAGUUUGAUUAUGGCUC...UCGUAACAAGGUAACCGU 4411
```

Using, predict we can BLAST the sequences.

```
R> cl <- predict(bl, seq[1,])
R> cl[1:5,]

  QueryID                     SubjectID Perc.Ident Alignment.Length
1    1675 gi|559795231|ref|NR_104821.1|      90.82             1459
2    1675 gi|444304125|ref|NR_074549.1|      85.99             1249
3    1675 gi|444304125|ref|NR_074549.1|      94.20               69
4    1675 gi|265678428|ref|NR_028730.1|      82.53             1494
5    1675 gi|343201138|ref|NR_041853.1|      82.30             1531
  Mismatches Gap.Openings Q.start Q.end S.start S.end     E Bits
1        124            9      16  1468       5  1459 0e+00 1943
2        158           15     235  1478     247  1483 0e+00 1321
3          4            0       1    69       1    69 3e-22  106
4        206           34      31  1475       1  1488 0e+00 1271
5        210           40       3  1481       1  1522 0e+00 1269
```

7. 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
```

8. Calculating Distances between Sequences

Calculating distances between sequences is important for many bioinformatics applications. The following distance metrics are available in **BiostringsTools**:

- 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, 2011).
- 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 et al, 2007).
- 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 (see simRank).
- 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.

```
R> s <- mutations(random_sequences(100), 100)
R> s

  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

R> ### calculate NSV distance
R> dNSV <- distNSV(s)
R> ### relationship with edit distance
R> dEdit <- distEdit(s)
R> df <- data.frame(dNSV=as.vector(dNSV), dEdit=as.vector(dEdit))
R> plot(sapply(df, jitter), cex=.1)
R> ### add lower bound (2*k, for Manhattan distance)
R> abline(0,1/(2*3), col="red", lwd=2)
R> ### add regression line
R> abline(lm(dEdit~dNSV, data=df), col="blue", lwd=2)
R> ### check correlation
R> cor(dNSV,dEdit)

[1] 0.8336
```

9. Conclusion

Acknowledgments

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

References

Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ (1990). "Basic local alignment search tool." *Journal of Molecular Biology*, **215**(3), 403–410. ISSN 0022-2836.

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.

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.

Smith TF, Waterman MS (1981). "Identification of common molecular subsequences." *Journal of Molecular Biology*, **147**(1), 195–197. ISSN 0022-2836.

Wang Q, Garrity GM, Tiedje JM, Cole JR (2007). "Naive Bayesian classifier for rapid assignment of rRNA sequences into the new bacterial taxonomy." *Applied and environmental microbiology*, **73**(16), 5261–5267.

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`