Using \textit{muscle} to produce multiple sequence alignments in \textit{Bioconductor}

Alex T. Kalinka

May 1, 2024
alex.t.kalinka@gmail.com

Institute for Population Genetics, Vetmeduni Vienna, Veterinärplatz 1, 1210 Vienna, Austria.

Abstract

Producing high-quality multiple sequence alignments of DNA, RNA, or amino acid sequences is often an essential component of any comparative sequence-based study. The MUSCLE algorithm employs a progressive alignment approach to optimise pairwise alignment scores, and achieves both high accuracy and reduced computational time even when handling thousands of sequences (Edgar, 2004,a). The \textit{R} package \texttt{muscle} integrates the MUSCLE algorithm into the \textit{Bioconductor} project by utilizing existing \textit{Biostrings} classes for representing sequence objects and multiple alignments.

Contents

1 Introduction 1
2 Example session 2
3 Arguments for the \texttt{muscle} function 3
4 \textit{R} Session Information 4

1 Introduction

Performing multiple sequence alignments of biological sequences is often an essential aspect of studies that utilize sequence data. For example, multiple sequence alignments are at the core of several studies, such as phylogenetic tree estimation based on sequence data, testing for signatures of selection in coding or non-coding sequences, comparative genomics, secondary structure prediction, or critical residue identification. Hence, the multiple sequences may be homologous sequences belonging to several different species, paralogous sequences belonging to a single species, orthologous sequences belonging to multiple individuals of a single species, or any other variant thereof.

The MUSCLE algorithm is a progressive alignment method that works with DNA, RNA, and amino acid sequences producing high-accuracy alignments with very fast computational times (Edgar, 2004,a). The algorithm is iterative, with later iterations refining the earlier alignments. In each iteration, pairwise alignment scores are computed for all sequence pairs (based on $k$-mer counting or global pairwise alignments) and the values are entered into a triangular distance matrix. This matrix is then used to build a binary tree of all the sequences (using one of various different hierarchical clustering algorithms, such as UPGMA or neighbour-joining). A progressive alignment is then built from this matrix by following the tree from the tips (individual sequences) to the root (all sequences aligned) adding in gaps as appropriate.
2 Example session

First, we must load the muscle package into our current R session:

```r
> library(muscle)
```

To illustrate the package, we will perform a multiple sequence alignment of the MAX gene (Wagner et al., 1992) across 31 mammalian species. These sequences are available in the umax object that is part of the muscle package, and is an object of class DNAStringSet:

```r
> umax
```

DNAStringSet object of length 31:

width seq names

```
[1] 483 ATGAGCGGATAAAGTACATGACATCGAG...GCTCCGGATGGAGGCCAGCTAA Ailuropoda_melan...  
[2] 489 ATGAGCGGATAAAGTACATGACATCGAG...GCTCCGGATGGAGGCCAGCTAA Bos_taurus   
[3] 483 ATGAGCGGATAAAGTACATGACATCGAG...GCTCCGGATGGAGGCCAGCTAA Callithrix_jacchus 
[4] 483 ATGAGCGGATAAAGTACATGACATCGAG...GCTCCGGATGGAGGCCAGCTAA Canis_familiaris 
[5] 483 ATGAGCGGATAAAGTACATGACATCGAG...GCTCCGGATGGAGGCCAGCTAA Cavia_porcellus  
...
[27] 483 ATGAGCGGATAAAGTACATGACATCGAG...GCTCCGGATGGAGGCCAGCTAA Rattus_norvegicus
[28] 483 ATGAGCGGATAAAGTACATGACATCGAG...GCTCCGGATGGAGGCCAGCTAA Sorex_araneus    
[29] 447 GAAGAGGATCAGGGTATCAATCAAGG...GCTCCGGATGGAGGCCAGCTAA Tarsius_syrich... 
[30] 444 GAAGAGCAGACGGAGGTTTCAATC...GCTCCGGATGGAGGCCAGCTAA Tupaia_belangeri 
[31] 447 GAAGAGGCAACCGAGGTTTCAATC...GCTCCGGATGGAGGCCAGCTAA Tursiops_trunc... 
```

All input to the muscle function should be objects of class XStringSet, which can be one of DNAStringSet, RNAStringSet, or AAStringSet (see package Biostrings (Pages et al., 2015)). An alignment is generated as follows (muscle automatically detects whether the input is DNA, RNA, or amino acid):

```r
> aln <- muscle(umax)
```

The output is an object of class MultipleAlignment (see package Biostrings):

```r
> aln
```

DNAMultipleAlignment with 31 rows and 492 columns:

```
[1] ATGAGCGGATAAAGTACATGACATCGAGG...GCTCCGGATGGAGGCCAGCTAA Ailuropoda_melan...  
[2] ATGAGCGGATAAAGTACATGACATCGAGG...GCTCCGGATGGAGGCCAGCTAA Bos_taurus   
[3] ATGAGCGGATAAAGTACATGACATCGAGG...GCTCCGGATGGAGGCCAGCTAA Callithrix_jacchus 
[4] ATGAGCGGATAAAGTACATGACATCGAGG...GCTCCGGATGGAGGCCAGCTAA Canis_familiaris 
[5] ATGAGCGGATAAAGTACATGACATCGAGG...GCTCCGGATGGAGGCCAGCTAA Cavia_porcellus  
[6] ATGAGCGGATAAAGTACATGACATCGAGG...GCTCCGGATGGAGGCCAGCTAA Choloepus_hoffm... 
[7] ATGAGCGGATAAAGTACATGACATCGAGG...GCTCCGGATGGAGGCCAGCTAA Dipodops... 
[8] ATGAGCGGATAAAGTACATGACATCGAGG...GCTCCGGATGGAGGCCAGCTAA Echinops_telfa... 
[9] -------------------------...GAACTCCGAGTGGAGGCCAGCTAA Erinaceus_europae... 
...
[23] ATGAGCGGATAAAGTACATGACATCGAGG...GAACTCCCGATGGAGGCCAGCTAA Sus_scrofa    
[24] ATGAGCGGATAAAGTACATGACATCGAGG...GAACTCCCGATGGAGGCCAGCTAA Pongo_abell... 
[25] -------------------------...GAACTCCCGATGGAGGCCAGCTAA Procavia_capens... 
[26] ATGAGCGGATAAAGTACATGACATCGAGG...GAACTCCCGATGGAGGCCAGCTAA Oryctolagus_cun... 
[27] ATGAGCGGATAAAGTACATGACATCGAGG...GAACTCCCGATGGAGGCCAGCTAA Rattus_norveg... 
[28] ATGAGCGGATAAAGTACATGACATCGAGG...GAACTCCCGATGGAGGCCAGCTAA Sorex_araneus 
[29] -------------------------...GAACTCCCGATGGAGGCCAGCTAA Tarsius_syrich... 
[30] -------------------------...GAACTCCCGATGGAGGCCAGCTAA Tupaia_belangeri 
[31] -------------------------...GAACTCCCGATGGAGGCCAGCTAA Tursiops_trunc... 
```
If the desired input is initially present in an external file, such as a fasta file, then these sequences can be read into an XStringSet object using one of the XstringSet input-output functions (readDNAStringSet, readRNAStringSet, or readAAStringSet). For example, to read in one of the example fasta files in the external data contained in the Biostrings package:

```r
> file.path <- system.file("extdata", "someORF.fa", package = "Biostrings")
> orf <- readDNAStringSet(file.path, format = "fasta")
```

This will read in a DNAStringSet object containing 7 unaligned sequences:

```r
> orf
```

**DNAStringSet object of length 7:**

<table>
<thead>
<tr>
<th>width</th>
<th>seq</th>
<th>names</th>
</tr>
</thead>
<tbody>
<tr>
<td>5573</td>
<td>ACTTGTAAATATATTTTATTT...CTTATGACGTTAGGATAT</td>
<td>YAL001C TPC3 SGD1...</td>
</tr>
<tr>
<td>5825</td>
<td>TTCCAGCGGATGAGACGACT...AGTAAATTCTCTCTCT</td>
<td>YAL002W VPS8 SGD1...</td>
</tr>
<tr>
<td>2987</td>
<td>CTTCATGTCAGGCTGCACTTCTG...TGGTACTCATATGCCTCAT</td>
<td>YAL003W EFB1 SGD1...</td>
</tr>
<tr>
<td>3929</td>
<td>CACTCATATCGGGGGTCTTACTT...TGTCCCGAAACCGAAAAAGTAC</td>
<td>YAL005C SSA1 SGD1...</td>
</tr>
<tr>
<td>2648</td>
<td>AYAGAAAGATCGGTCCTGTTCATAG...CTATAAATTAGTGTAGAATAG</td>
<td>YAL007C ERP2 SGD1...</td>
</tr>
<tr>
<td>2597</td>
<td>GTGTCCGGGCCTCGCAGGCGTTC...AAGTTTTGAGCAGAATGTACTTTT</td>
<td>YAL008W FUN14 SGD1...</td>
</tr>
<tr>
<td>2780</td>
<td>CAAGATATGCTAAAGTCTCC...GCTAAGGAAGAAAAATACAC</td>
<td>YAL009W SP07 SGD1...</td>
</tr>
</tbody>
</table>

## 3 Arguments for the muscle function

Many different arguments can be passed to the muscle function, and these are described in detail in the online documentation. These arguments are either options (taking various values) or flags (either TRUE or FALSE). Here, I describe some of the more commonly-used arguments.

**Enhanced speed.** To enhance the speed of the algorithm, the `diags = TRUE` flag will optimize the speed with a potential loss of accuracy:

```r
> aln <- muscle(umax, diags = TRUE)
```

**Gap penalties.** Default gap penalties can be modified to produce altered alignments. The gap penalty must be negative, with larger negative values indicating more stringent penalties:

```r
> aln <- muscle(umax, gapopen = -30)
```

**Remove progress indicators.** When running the algorithm repeatedly (for a batch of sequences, for example), it may be preferred to stop output of the algorithm's progress to the screen (e.g. if there is a global progress indicator running):

```r
> aln <- muscle(umax, quiet = TRUE)
```

**Maximum number of hours.** If an alignment is expected to take a long time, a maximum total number of hours can be specified, which, if reached, will lead to the algorithm stopping at this point and returning the current alignment:

```r
> aln <- muscle(umax, maxhours = 24.0)
```

**Log file.** To find out what default settings are being used for all the arguments, a log file can be written to disk using the `log` argument in conjunction with the `verbose` argument, e.g. `log = "log.txt", verbose = TRUE`. This will write out the default values to the file `log.txt` in the current working directory of R.
4 R Session Information

The examples in this vignette were run under the following conditions:

```r
> sessionInfo()
```

R version 4.4.0 beta (2024-04-15 r86425)
Platform: x86_64-pc-linux-gnu
Running under: Ubuntu 22.04.4 LTS

Matrix products: default
BLAS: /home/biocbuild/bbs-3.19-bioc/R/lib/libRblas.so
LAPACK: /usr/lib/x86_64-linux-gnu/lapack/liblapack.so.3.10.0

locale:
[1] LC_CTYPE=en_US.UTF-8    LC_NUMERIC=C
[2] LC_TIME=en_US.UTF-8     LC_COLLATE=en_US.UTF-8
[3] LC_MONETARY=en_US.UTF-8 LC_MESSAGES=en_US.UTF-8
[4] LC_PAPER=en_US.UTF-8    LC_NAME=C
[5] LC_ADDRESS=C            LC_TELEPHONE=C
[6] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C

time zone: America/New_York
tzcode source: system (glibc)

attached base packages:
[1] stats4  stats graphics grDevices utils datasets methods
[2] base

other attached packages:
[1] muscle_3.46.0 Biostrings_2.72.0 GenomeInfoDb_1.40.0
[2] XVector_0.44.0 IRanges_2.38.0 S4Vectors_0.42.0
[3] BiocGenerics_0.50.0

loaded via a namespace (and not attached):
[1] httr_1.4.7    zlibbioc_1.50.0    compiler_4.4.0
[2] R6_2.5.1      tools_4.4.0       GenomeInfoDbData_1.2.12
[3] crayon_1.5.2    UCSC.utils_1.0.0    jsonlite_1.8.8

References


