

## **msa**

### **An R Package for Multiple Sequence Alignment**

**Enrico Bonatesta, Christoph Horejš-Kainrath, and Ulrich Bodenhofer**

Institute of Bioinformatics, Johannes Kepler University Linz  
Altenberger Str. 69, 4040 Linz, Austria  
*msa@bioinf.jku.at*

**Version 1.4.5, August 9, 2016**

## **Scope and Purpose of this Document**

This document provides a gentle introduction into the R package `msa`. Not all features of the R package are described in full detail. Such details can be obtained from the documentation enclosed in the R package. Further note the following: (1) this is not an introduction to multiple sequence alignment or algorithms for multiple sequence alignment; (2) this is not an introduction to R or any of the Bioconductor packages used in this document. If you lack the background for understanding this manual, you first have to read introductory literature on the subjects mentioned above.

## Contents

## 1 Introduction

Multiple sequence alignment is one of the most fundamental tasks in bioinformatics. Algorithms like ClustalW [?], ClustalOmega [?], and MUSCLE [?, ?] are well known and widely used. However, all these algorithms are implemented as stand-alone command line programs without any integration into the R/Bioconductor ecosystem. Before the `msa` package, only the `muscle` package has been available in R, but no other multiple sequence alignment algorithm, although the `Biostrings` package has provided data types for representing multiple sequence alignments for quite some time. The `msa` package aims to close that gap by providing a unified R interface to the multiple sequence alignment algorithms ClustalW, ClustalOmega, and MUSCLE. The package requires no additional software packages and runs on all major platforms. Moreover, the `msa` package provides an R interface to the powerful  $\text{\LaTeX}$  package `\text{\texttt{TeXshade}}` [?] which allows for a highly customizable plots of multiple sequence alignments. Unless some very special features of `\text{\texttt{TeXshade}}` are required, users can pretty-print multiple sequence alignments without the need to know the details of  $\text{\LaTeX}$  or `\text{\texttt{TeXshade}}`.

## 2 Installation

The `msa` R package (current version: 1.4.5) is available via Bioconductor. The simplest way to install the package is the following:

```
source("http://www.bioconductor.org/biocLite.R")
biocLite("msa")
```

To test the installation of the `msa` package, enter

```
library(msa)
```

in your R session. If this command terminates without any error message or warning, you can be sure that the `msa` package has been installed successfully. If so, the `msa` package is ready for use now and you can start performing multiple sequence alignments.

To make use of all functionalities of `msaPrettyPrint()`, a  $\text{\TeX}/\text{\LaTeX}$  system [?] must be installed. To make use of  $\text{\LaTeX}$  code created by `msaPrettyPrint()` or to use the output of `msaPrettyPrint()` in Sweave [?] or knitr [?] documents, the  $\text{\LaTeX}$  package `\text{\texttt{TeXshade}}` (file `texshade.sty`) [?] must be accessible to the  $\text{\LaTeX}$  system too. The file `texshade.sty` is shipped with the `msa` package. To determine where the file is located, enter the following command in your R session:

```
system.file("tex", "texshade.sty", package="msa")
```

Alternatively, `\text{\texttt{TeXshade}}` can be installed directly from the Comprehensive  $\text{\TeX}$  Archive Network (CTAN).<sup>1</sup>

---

<sup>1</sup><https://www.ctan.org/pkg/texshade>

### 3 msa for the Impatient

In order to illustrate the basic workflow, this section presents a simple example with default settings and without going into the details of each step. Let us first load amino acid sequences from one of the example files that are supplied with the msa package:

```
mySequenceFile <- system.file("examples", "exampleAA.fasta", package="msa")
mySequences <- readAAStringSet(mySequenceFile)
mySequences

## A AAStringSet instance of length 9
##      width seq                      names
## [1]   452 MSTAVLENPGLGRKLS...NSEIGILCSALQKIK PH4H_Homo_sapiens
## [2]   453 MAAVLENGVLSRKLS...SEVGILCNALQKIKS PH4H_Rattus_norve...
## [3]   453 MAAVLENGVLSRKLS...SEVGILCHALQKIKS PH4H_Mus_musculus
## [4]   297 MNDRADFVVPDITTRK...LNAGDRQGWADTEDV PH4H_Chromobacter...
## [5]   262 MKTTQYVARQPDDNGF...RLGLHAPLFPPKQAA PH4H_Pseudomonas...
## [6]   451 MSALVLESRALGRKLS...SSEVEILCSALQKLK PH4H_Bos_taurus
## [7]   313 MAIATPTSAAPTPAPA...LNAGTREGWADTADI PH4H_Ralstonia_so...
## [8]   294 MSGDGLSNGPPPGARP...AYATAGGRLAGAAAG PH4H_Caulobacter...
## [9]   275 MSVAEYARDCAAQGLR...VARRKDQKALDPATV PH4H_Rhizobium_loti
```

Now that we have loaded the sequences, we can run the `msa()` function which, by default, runs ClustalW with default parameters:

```
myFirstAlignment <- msa(mySequences)

## use default substitution matrix

myFirstAlignment

## CLUSTAL 2.1
##
## Call:
##      msa(mySequences)
##
## MsaAAMultipleAlignment with 9 rows and 456 columns
##      aln                      names
## [1] MAAVLENGVLSRKLSDF...SINSEVGILCNALQKIKS PH4H_Rattus_norve...
## [2] MAAVLENGVLSRKLSDF...SINSEVGILCHALQKIKS PH4H_Mus_musculus
## [3] MSTAVLENPGLGRKLSDF...SINSEIGILCSALQKIK- PH4H_Homo_sapiens
## [4] MSALVLESRALGRKLSDF...SISSEVEILCSALQKLK- PH4H_Bos_taurus
## [5] -----...GWADTEDV----- PH4H_Chromobacter...
## [6] -----...GWADTADI----- PH4H_Ralstonia_so...
## [7] -----...AYATAGGRLAGAAAG--- PH4H_Caulobacter...
```

```
## [8] -----...----- PH4H_Pseudomonas_...
## [9] -----...----- PH4H_Rhizobium_loti
## Con -----...??????IL??A??? Consensus
```

Obviously, the default printing function shortens the alignment for the sake of compact output. The `print()` function provided by the `msa` package provides some ways for customizing the output, such as, showing the entire alignment split over multiple blocks of sub-sequences:

```
print(myFirstAlignment, show="complete")

##
## MsaAAMultipleAlignment with 9 rows and 456 columns
##      aln (1..39)                                names
## [1] MAAVLENGVLSRKLSDFGQETSYIEDNSNQNGAISLIF PH4H_Rattus_norve...
## [2] MAAVLENGVLSRKLSDFGQETSYIEDNSNQNGAVSLIF PH4H_Mus_musculus
## [3] MSTAVLENPGLGRKLSDFGQETSYIEDNCNQNNGAISLIF PH4H_Homo_sapiens
## [4] MSALVLESRALGRKLSDFGQETSYIEGNSDQN-AVSLIF PH4H_Bos_taurus
## [5] ----- PH4H_Chromobacter...
## [6] ----- PH4H_Ralstonia_so...
## [7] ----- PH4H_Caulobacter...
## [8] ----- PH4H_Pseudomonas_...
## [9] ----- PH4H_Rhizobium_loti
## Con ----- Consensus
##
##      aln (40..78)                                names
## [1] SLKEEVGALAKVLRRLFEEENDINLTHIESRPSRLNKDEYE PH4H_Rattus_norve...
## [2] SLKEEVGALAKVLRRLFEEENEINLTHIESRPSRLNKDEYE PH4H_Mus_musculus
## [3] SLKEEVGALAKVLRRLFEEENDVNLTHIESRPSRLKKDEYE PH4H_Homo_sapiens
## [4] SLKEEVGALARVLRRLFEEENDINLTHIESRPSRLRKDEYE PH4H_Bos_taurus
## [5] ----- PH4H_Chromobacter...
## [6] ----- PH4H_Ralstonia_so...
## [7] ----- PH4H_Caulobacter...
## [8] ----- PH4H_Pseudomonas_...
## [9] ----- PH4H_Rhizobium_loti
## Con ----- Consensus
##
##      aln (79..117)                                names
## [1] FFTYLDKRTKPVLSGIKSLRNDIGATVHELSDRDKKNT PH4H_Rattus_norve...
## [2] FFTYLDKRSPVLSGIKSLRNDIGATVHELSDRDKKNT PH4H_Mus_musculus
## [3] FFTHLDKRSPLALTNIIKILRHDIGATVHELSDRDKKNT PH4H_Homo_sapiens
## [4] FFTNLDQRSVPALANI KILRHDIGATVHELSDRDKKNT PH4H_Bos_taurus
## [5] ----- PH4H_Chromobacter...
## [6] ----- PH4H_Ralstonia_so...
## [7] ----- PH4H_Caulobacter...
## [8] ----- PH4H_Pseudomonas_...
## [9] ----- PH4H_Rhizobium_loti
```

```

## Con ----- Consensus
##
##      aln (118..156)                                names
## [1] VPWFPRTIQELDRFANQILSYGAELDADHPGFKDPVYRA PH4H_Rattus_norve...
## [2] VPWFPRTIQELDRFANQILSYGAELDADHPGFKDPVYRA PH4H_Mus_musculus
## [3] VPWFPRTIQELDRFANQILSYGAELDADHPGFKDPVYRA PH4H_Homo_sapiens
## [4] VPWFPRTIQELDNFANQVLSYGAELDADHPGFKDPVYRA PH4H_Bos_taurus
## [5] -----MNDRADFVVPD-----ITTRKNVG PH4H_Chromobacter...
## [6] -----MAIATPTSAAPTAPAGFTGTLTDKLRQ PH4H_Ralstonia_so...
## [7] -----MSG-----DGLSNG PH4H_Caulobacter_...
## [8] -----MKTTQY PH4H_Pseudomonas_...
## [9] -----MSVAEYAR-----DCAAQG PH4H_Rhizobium_loti
## Con -----????????Y????D????D???? Consensus
##
##      aln (157..195)                                names
## [1] RRKQFADIAYNYRHGQPIPRVEYTEEEKQWTGTVFRTLK PH4H_Rattus_norve...
## [2] RRKQFADIAYNYRHGQPIPRVEYTEEEKQWTGTVFRTLK PH4H_Mus_musculus
## [3] RRKQFADIAYNYRHGQPIPRVEYMEEEKQWTGTVFRTLK PH4H_Homo_sapiens
## [4] RRKQFADIAYNYRHGQPIPRVEYTEEEKQWTGTVFRTLK PH4H_Bos_taurus
## [5] LSHDAN-----DFTLPQPLDRYSAEDHATWATLYQRQC PH4H_Chromobacter...
## [6] FAEGLDGQTLRPDFTMEQPVHRYTAADHATWRTLYDRQE PH4H_Ralstonia_so...
## [7] PPPGAR-----PDWTIDQGWETYTQAEDHVWITLYERQT PH4H_Caulobacter_...
## [8] VARQPD-----DNGFIHYPETEHQVWNTLITRQL PH4H_Pseudomonas_...
## [9] LRGDYS--VCRADFTVAQDYD--YSDEEQAVWRTLCDRQT PH4H_Rhizobium_loti
## Con ?R?Q????????P?P???YTEEE??TW?TL??RQ? Consensus
##
##      aln (196..234)                                names
## [1] ALYKTHACYEHNHIFPLLEKYCGFREDNIPQLEDVSQFL PH4H_Rattus_norve...
## [2] ALYKTHACYEHNHIFPLLEKYCGFREDNIPQLEDVSQFL PH4H_Mus_musculus
## [3] SLYKTHACYEYNHIFPLLEKYCGFHEDNIPQLEDVSQFL PH4H_Homo_sapiens
## [4] SLYKTHACYEHNHIFPLLEKYCGFREDNIPQLEEVVSQFL PH4H_Bos_taurus
## [5] KLLPGRACDEFMEGL----ERLEVDADRVPDFNKLNQKL PH4H_Chromobacter...
## [6] ALLPGRACDEFQGL----STLGMSREGVPSFDRLNETL PH4H_Ralstonia_so...
## [7] DMLHGRACDEFMRGL----DALDLHRSGIPDFARINEEL PH4H_Caulobacter_...
## [8] KVIEGRACQEYLDGI----EQLGLPHERIPQLDEINRVL PH4H_Pseudomonas_...
## [9] KLTRKLAHHSYLDGV----EKLGL-LDRIPDFEDVSTKL PH4H_Rhizobium_loti
## Con ?L????AC?E???G?----??LG???D?IPQLE?VSQ?L Consensus
##
##      aln (235..273)                                names
## [1] QTCTGFRLRPVAGLLSSRDFLGGLAFRVFHCTQYIRHGS PH4H_Rattus_norve...
## [2] QTCTGFRLRPVAGLLSSRDFLGGLAFRVFHCTQYIRHGS PH4H_Mus_musculus
## [3] QTCTGFRLRPVAGLLSSRDFLGGLAFRVFHCTQYIRHGS PH4H_Homo_sapiens
## [4] QSCTGFRLRPVAGLLSSRDFLGGLAFRVFHCTQYIRHGS PH4H_Bos_taurus
## [5] MAATGWKIVAVPGLIPDDVFFEHLANRRFPVTWWLREPH PH4H_Chromobacter...
## [6] MRATGWQIVAVPGLVPDEVFFEHLANRRFPASWWMRRPD PH4H_Ralstonia_so...
## [7] KRLTGWTVAVPGLVPDDVFFDHLANRRFPAGQFIRKPH PH4H_Caulobacter_...

```

```

## [8] QATTGWRVARVPALIPFQTFPELLASQQFPVATFIRTPE PH4H_Pseudomonas_...
## [9] RKLTGWEIIAVPGLIPAAPFFDHLANRRFPVTNWLRTQ PH4H_Rhizobium_loti
## Con Q??TGWR??VPGL?P??FF??LA?R?FP?TQ?IR??? Consensus
##
##      aln (274..312)                                names
## [1] KPMYTPEPDICHELLGHVPLFSDRSFAQFSQEIG-LASL PH4H_Rattus_norve...
## [2] KPMYTPEPDICHELLGHVPLFSDRSFAQFSQEIG-LASL PH4H_Mus_musculus
## [3] KPMYTPEPDICHELLGHVPLFSDRSFAQFSQEIG-LASL PH4H_Homo_sapiens
## [4] KPMYTPEPDICHELLGHVPLFSDRSFAQFSQEIG-LASL PH4H_Bos_taurus
## [5] QLDYLQEPDVFHDLFHGVPLLINPVFADYLEAYGKGGVK PH4H_Chromobacter...
## [6] QLDYLQEPDGFHDIFGHVPLLINPVFADYMQAYGQGGLK PH4H_Ralstonia_so...
## [7] ELDYLQEPDIFHDVFGHVPLMTDPVFADYMQAYGEGGRR PH4H_Caulobacter_...
## [8] ELDYLQEPDIFHEIFGHCPLLTNPWFAEFTHTYKGLGLK PH4H_Pseudomonas_...
## [9] ELDYIVEPDMFHDFFGHVPVLSQPVFADFMQMYGKKAGD PH4H_Rhizobium_loti
## Con ?LDY??EPDIFHELFGHVPLSDP?FA?F?Q?YG?LA?? Consensus
##
##      aln (313..351)                                names
## [1] GAPDEYIEKLATIIYWFTVEFGLCKEG-DSIKAYGAGLLS PH4H_Rattus_norve...
## [2] GAPDEYIEKLATIIYWFTVEFGLCKEG-DSIKAYGAGLLS PH4H_Mus_musculus
## [3] GAPDEYIEKLATIIYWFTVEFGLCKQG-DSIKAYGAGLLS PH4H_Homo_sapiens
## [4] GAPDEYIEKLATIIYWFTVEFGLCKQG-DSIKAYGAGLLS PH4H_Bos_taurus
## [5] AKALGALPMLARLYWYTVEFGLINTP-AGMRIYGAGILS PH4H_Chromobacter...
## [6] AARLGALDMLARLYWYTVEFGLIRTP-AGLRIYGAGIVS PH4H_Ralstonia_so...
## [7] ALGLGRLANLARLYWYTVEFGLMNTP-AGLRIYGAGIVS PH4H_Caulobacter_...
## [8] ASKE-ERVFLARLYWMTIEFGLVETD-QGKRIYGGGILS PH4H_Pseudomonas_...
## [9] IIALGGDEMITRLYWYTAEYGLVQEAGQPLKAFGAGLMS PH4H_Rhizobium_loti
## Con ?A?????E?LARLYW?TVEFGL????-???KAYGAGLLS Consensus
##
##      aln (352..390)                                names
## [1] SFGELQYCLSD-KPKLLPLELEKTACQEYSVTEFQPLYY PH4H_Rattus_norve...
## [2] SFGELQYCLSD-KPKLLPLELEKTACQEYTVTEFQPLYY PH4H_Mus_musculus
## [3] SFGELQYCLSE-KPKLLPLELEKTAIQNYTVTEFQPLYY PH4H_Homo_sapiens
## [4] SFGELQYCLSD-KPKLLPLELEKTAVQEYTITEFQPLYY PH4H_Bos_taurus
## [5] SKSESIYCLDSASPNRVGFDLMRIMNTRYRIDTFQKTYF PH4H_Chromobacter...
## [6] SKSESVYALDSASPNRIGFDVHRIMRTRYRIDTFQKTYF PH4H_Ralstonia_so...
## [7] SRTESIFALDDPSPNRIGFDLERVMRTLYRIDDFQQVYF PH4H_Caulobacter_...
## [8] SPKETVYSLSD-EPLHQA FNPLEAMRTPYRIDILQPLYF PH4H_Pseudomonas_...
## [9] SFTELQFAVEGKDAHHPFDLETVMRTGYEIDKFQRAYF PH4H_Rhizobium_loti
## Con SF?ELQYCLSD-?P???PF?LE??M?T?Y?ID?FQPLYF Consensus
##
##      aln (391..429)                                names
## [1] VAESFSDAKEKVRTFAATIPRPF SVRYDPYTQRVEVL DN PH4H_Rattus_norve...
## [2] VAESFNDAKEKVRTFAATIPRPF SVRYDPYTQRVEVL DN PH4H_Mus_musculus
## [3] VAESFNDAKEKVRNFAATIPRPF SVRYDPYTQRIEVL DN PH4H_Homo_sapiens
## [4] VAESFNDAKEKVRNFAATIPRPF SVHYDPYTQRIEVL DN PH4H_Bos_taurus
## [5] VIDSFKQLFDATA-PDFAPLYLQLADAQPWGAGDVAPDD PH4H_Chromobacter...

```



```
## [6] VIDSFEQLFDATR-PDFTPLYEALGTLPTFGAGDVVDGD PH4H_Ralstonia_so...
## [7] VIDSIQTLQEVTL-RDFGAIYERLASVSDIGVAEIVPGD PH4H_Caulobacter_...
## [8] VLPDLKRLFQLAQ-EDIMALVHEAMRLG-LHAPLFPPKQ PH4H_Pseudomonas_...
## [9] VLPSFDALRDAFQTADFEAIVARRKDQKALDPATV---- PH4H_Rhizobium_loti
## Con V??SF??L?E??R??D?T??????????P??????V?D? Consensus
##
##      aln (430..456)                names
## [1] TQQLKILADSINSEVGILCNALQKIKS PH4H_Rattus_norve...
## [2] TQQLKILADSINSEVGILCHALQKIKS PH4H_Mus_musculus
## [3] TQQLKILADSINSEIGILCSALQKIK- PH4H_Homo_sapiens
## [4] TQQLKILADSISSVEILCSALQKIK- PH4H_Bos_taurus
## [5] LVLNAGDRQGWADTEDV----- PH4H_Chromobacter...
## [6] AVLNAGTREGWADTADI----- PH4H_Ralstonia_so...
## [7] AVLTRGT-QAYATAGGRLAGAAAG--- PH4H_Caulobacter_...
## [8] AA----- PH4H_Pseudomonas_...
## [9] ----- PH4H_Rhizobium_loti
## Con ??????????????IL??A???--- Consensus
```

The `msa` package additionally offers the function `msaPrettyPrint()` which allows for pretty-printing multiple alignments using the `LATEX` package `TeXshade`. As an example, the following R code creates a PDF file `myfirstAlignment.pdf` which is shown in Figure ??:

```
msaPrettyPrint(myFirstAlignment, output="pdf", showNames="none",
               showLogo="none", askForOverwrite=FALSE, verbose=FALSE)
```

In the above call to `msaPrettyPrint()`, the printing of sequence names has been suppressed by `showNames="none"`. The settings `askForOverwrite=FALSE` and `verbose=FALSE` are necessary for building this vignette, but, in an interactive R session, they are not necessary.

Almost needless to say, the file names created by `msaPrettyPrint()` are customizable. By default, the name of the argument is taken as file name. More importantly, the actual output of `msaPrettyPrint()` is highly customizable, too. For more details, see the Section ?? and the help page of the function (`?msaPrettyPrint`).

The `msaPrettyPrint()` function is particularly useful for pretty-printing multiple sequence alignments in Sweave [?] or knitr [?] documents. More details are provided in Section ?. Here, we restrict to a teasing example:

```
msaPrettyPrint(myFirstAlignment, y=c(164, 213), output="asis",
               showNames="none", showLogo="none", askForOverwrite=FALSE)
```



Figure 1: The PDF file `myfirstAlignment.pdf` created with `msaPrettyPrint()`.

```

IAYNYRHGQPIPRVEYTEEEKQTWGTVFRTLKALYKTHACYEHNHIFPLL 213
IAYNYRHGQPIPRVEYTEEEKRTWGTVFRTLKALYKTHACYEHNHIFPLL 213
IAYNYRHGQPIPRVEYMEEEEKKTWGTVFRTLKSLYKTHACYEYNHIFPLL 213
IAYNYRHGQPIPRVEYTEEEKKTWGTVFRTLKSLYKTHACYEHNHIFPLL 212
.....DFTLPQPLDRYSAEDHATWATLYQRQCKLLPGRACDEFMEGL... 67
QTLRPDFTMEQPVHRYTAADHATWRTL YDRQEALLPGRACDEFLLQGL... 83
....PDWTIDQGWETYTQA EHDVWITLYERQTDMLHGRACDEFMRGL... 58
.....DNGFIHYPETEHQVWNTLITRQLKVIIEGRACQEYLDGI... 50
.VCRADFTVAQDYD.YSDEEQAVWRTLCDRQTKLTRKL AHHSYLDGV... 65
          *  *      !****      *!  !*      **      *      !*  *      *

```

X non conserved  
X  $\geq 50\%$  conserved

## 4 Functions for Multiple Sequence Alignment in More Detail

The example in Section ?? above simply called the function `msa()` without any additional arguments. We mentioned already that, in this case, ClustalW is called with default parameters. We can also explicitly request ClustalW or one of the two other algorithms ClustalOmega or MUSCLE:

```

myClustalWAlignment <- msa(mySequences, "ClustalW")

## use default substitution matrix

myClustalWAlignment

## CLUSTAL 2.1
##
## Call:
##   msa(mySequences, "ClustalW")
##
## MsaAAMultipleAlignment with 9 rows and 456 columns
##      aln                                     names
## [1] MAAVVLENGVL SRKLSDF...SINSEVGILCNALQKIKS PH4H_Rattus_norve...
## [2] MAAVVLENGVL SRKLSDF...SINSEVGILCHALQKIKS PH4H_Mus_musculus
## [3] MSTAVLENPGLGRKLSDF...SINSEIGILCSALQKIK- PH4H_Homo_sapiens
## [4] MSALVLESRALGRKLSDF...SISSEVEILCSALQKIK- PH4H_Bos_taurus
## [5] -----...GWADTEDV----- PH4H_Chromobacter...
## [6] -----...GWADTADI----- PH4H_Ralstonia_so...
## [7] -----...AYATAGGRLAGAAAG--- PH4H_Caulobacter_...
## [8] -----...----- PH4H_Pseudomonas_...
## [9] -----...----- PH4H_Rhizobium_loti
## Con -----...???????IL??A???--- Consensus

myClustalOmegaAlignment <- msa(mySequences, "ClustalOmega")

```

```
## using Gonnet

myClustalOmegaAlignment

## ClustalOmega 1.2.0
##
## Call:
##   msa(mySequences, "ClustalOmega")
##
## MsaAAMultipleAlignment with 9 rows and 467 columns
##      aln                                     names
## [1] MSALVLESRALGRKLSDF...SISSEVEILCSALQKLK- PH4H_Bos_taurus
## [2] MSTAVLENPGLGRKLSDF...SINSEIGILCSALQKIK- PH4H_Homo_sapiens
## [3] MAAVLENGVLSRKLSDF...SINSEVGILCNALQKIKS PH4H_Rattus_norve...
## [4] MAAVLENGVLSRKLSDF...SINSEVGILCHALQKIKS PH4H_Mus_musculus
## [5] -----...----- PH4H_Pseudomonas_...
## [6] -----...----- PH4H_Rhizobium_loti
## [7] -----...LAGAAAG----- PH4H_Caulobacter_...
## [8] -----...V----- PH4H_Chromobacter...
## [9] -----...I----- PH4H_Ralstonia_so...
## Con -----...???????----- Consensus

myMuscleAlignment <- msa(mySequences, "Muscle")
myMuscleAlignment

## MUSCLE 3.8.31
##
## Call:
##   msa(mySequences, "Muscle")
##
## MsaAAMultipleAlignment with 9 rows and 460 columns
##      aln                                     names
## [1] MAAVLENGVLSRKLSDF...SINSEVGILCNALQKIKS PH4H_Rattus_norve...
## [2] MAAVLENGVLSRKLSDF...SINSEVGILCHALQKIKS PH4H_Mus_musculus
## [3] MSTAVLENPGLGRKLSDF...SINSEIGILCSALQKIK- PH4H_Homo_sapiens
## [4] MSALVLESRALGRKLSDF...SISSEVEILCSALQKLK- PH4H_Bos_taurus
## [5] -----...----- PH4H_Pseudomonas_...
## [6] -----...----- PH4H_Rhizobium_loti
## [7] -----...AYATAGGRLAGAAAG--- PH4H_Caulobacter_...
## [8] -----MNDRADF...QGWADTEDV----- PH4H_Chromobacter...
## [9] MAIATPTSAAPTAPAGF...EGWADTADI----- PH4H_Ralstonia_so...
## Con M????????????????DF...????????L??A???--- Consensus
```

Please note that the call `msa(mySequences, "ClustalW", ...)` is just a shortcut for the call `msaClustalW(mySequences, ...)`, analogously for `msaClustalOmega()` and `msaMuscle()`.

In other words, `msa()` is nothing else but a wrapper function that provides a unified interface to the three functions `msaClustalW()`, `msaClustalOmega()`, and `msaMuscle()`.

All three functions `msaClustalW()`, `msaClustalOmega()`, and `msaMuscle()` have the same parameters: The input sequences are passed as argument `inputSeqs`, and all functions have the following arguments: `cluster`, `gapOpening`, `gapExtension`, `maxiters`, `substitutionMatrix`, `order`, `type`, and `verbose`. The ways these parameters are interpreted, are largely analogous, although there are some differences, also in terms of default values. See the subsections below and the man page of the three functions for more details. All of the three functions `msaClustalW()`, `msaClustalOmega()`, and `msaMuscle()`, however, are not restricted to the parameters mentioned above. All three have a `'...'` argument through which several other algorithm-specific parameters can be passed on to the underlying library. The following subsections provide an overview of which parameters are supported by each of the three algorithms.

#### 4.1 ClustalW-Specific Parameters

The original implementation of ClustalW offers a lot of parameters for customizing the way a multiple sequence alignment is computed. Through the `'...'` argument, `msaClustalW()` provides an interface to make use of most these parameters (see the documentation of ClustalW<sup>2</sup> for a comprehensive overview). Currently, the following restrictions and caveats apply:

- The parameters `infile`, `clustering`, `gapOpen`, `gapExt`, `numiters`, `matrix`, and `outorder` have been renamed to the standardized argument names `inputSeqs`, `cluster`, `gapOpening`, `gapExtension`, `maxiters`, `substitutionMatrix`, and `order` in order to provide a consistent interface for all three multiple sequence alignment algorithms.
- Boolean flags must be passed as logical values, e.g. `verbose=TRUE`.
- The parameter `quiet` has been replaced by `verbose` (with the exact opposite meaning).
- The following parameters are (currently) not supported: `bootstrap`, `check`, `fullhelp`, `interactive`, `maxseqlen`, `options`, and `tree`.
- For the parameter `output`, only the choice `"clustal"` is available.

#### 4.2 ClustalOmega-Specific Parameters

In the same way as ClustalW, the original implementation of ClustalOmega also offers a lot of parameters for customizing the way a multiple sequence alignment is computed. Through the `'...'` argument, `msaClustalOmega()` provides an interface to make use of most these parameters (see the documentation of ClustalOmega<sup>3</sup> for a comprehensive overview). Currently, the following restrictions and caveats apply:

- The parameters `infile`, `cluster-size`, `iterations`, and `output-order` have been renamed to the argument names `inputSeqs`, `cluster`, `maxiters`, and `order` in order to provide a consistent interface for all three multiple sequence alignment algorithms.

---

<sup>2</sup>[http://www.clustal.org/download/clustalw\\_help.txt](http://www.clustal.org/download/clustalw_help.txt)

<sup>3</sup><http://www.clustal.org/omega/README>

- ClustalOmega does not allow for setting custom gap penalties. Therefore, setting the parameters `gapOpening` and `gapExtension` currently has no effect and will lead to a warning. These arguments are only defined for future extensions and consistency with the other algorithms available in `msa`.
- ClustalOmega only allows for choosing substitution matrices from a pre-defined set of names, namely "BLOSUM30", "BLOSUM40", "BLOSUM50", "BLOSUM65", "BLOSUM80", and "Gonnet". This is a new feature — the original ClustalOmega implementation does not allow for using any custom substitution matrix. However, since these are all amino acid substitution matrices, ClustalOmega is still hardly useful for multiple alignments of nucleotide sequences.
- Boolean flags must be passed as logical values, e.g. `verbose=TRUE`.
- The following parameters are (currently) not supported: `maxSeqLength` and `help`.
- For the parameter `outFmt`, only the choice "clustal" is available.

### 4.3 MUSCLE-Specific Parameters

Finally, also MUSCLE offers a lot of parameters for customizing the way a multiple sequence alignment is computed. Through the `'...'` argument, `msaMuscle()` provides an interface to make use of most these parameters (see the documentation of MUSCLE<sup>4</sup> for a comprehensive overview). Currently, the following restrictions and caveats apply:

- The parameters `in`, `gapOpen`, `gapExtend`, `matrix`, and `seqtype` have been renamed to `inputSeqs`, `gapOpening`, `gapExtension`, `substitutionMatrix` and `type` in order to provide a consistent interface for all three multiple sequence alignment algorithms.
- Boolean flags must be passed as logical values, e.g. `verbose=TRUE`.
- The parameter `quiet` has been replaced by `verbose` (with the exact opposite meaning).
- The following parameters are currently not supported: `clw`, `clwstrict`, `fastaout`, `group`, `html`, `in1`, `in2`, `log`, `loga`, `msaout`, `msf`, `out`, `phyi`, `phyiout`, `phys`, `physout`, `refine`, `refinew`, `scorefile`, `spscore`, `stable`, `termgaps4`, `termgapsfull`, `termgapshalf`, `termgapshalflonger`, `tree1`, `tree2`, `usetree`, `weight1`, and `weight2`.

## 5 Printing Multiple Sequence Alignments

As already shown above, multiple sequence alignments can be shown in plain text format on the R console using the `print()` function (which is implicitly called if just the object name is entered on the R console). This function allows for multiple customizations, such as, enabling/disabling to display a consensus sequence, printing the entire alignment or only a subset, enabling/disabling to display sequence names, and adjusting the width allocated for sequence names. For more information, the reader is referred to the help page of the `print` function:

<sup>4</sup><http://www.drive5.com/muscle/muscle.html>

```
help("print,MsaDNAMultipleAlignment-method")
```

We only provide some examples here:

```
print(myFirstAlignment)

## CLUSTAL 2.1
##
## Call:
##   msa(mySequences)
##
## MsaAAMultipleAlignment with 9 rows and 456 columns
##      aln                      names
## [1] MAAVLENGVLSRKLSDF...SINSEVGILCNALQKIKS PH4H_Rattus_norve...
## [2] MAAVLENGVLSRKLSDF...SINSEVGILCHALQKIKS PH4H_Mus_musculus
## [3] MSTAVLENPGLGRKLSDF...SINSEIGILCSALQKIK- PH4H_Homo_sapiens
## [4] MSALVLESRALGRKLSDF...SISSEVEILCSALQKLK- PH4H_Bos_taurus
## [5] -----...GWADTEDV----- PH4H_Chromobacter...
## [6] -----...GWADTADI----- PH4H_Ralstonia_so...
## [7] -----...AYATAGGRLAGAAAG--- PH4H_Caulobacter_...
## [8] -----...----- PH4H_Pseudomonas_...
## [9] -----...----- PH4H_Rhizobium_loti
## Con -----...???????IL??A???--- Consensus

print(myFirstAlignment, show="complete")

##
## MsaAAMultipleAlignment with 9 rows and 456 columns
##      aln (1..39)                      names
## [1] MAAVLENGVLSRKLSDFGQETSYIEDNSNQNGAISLIF PH4H_Rattus_norve...
## [2] MAAVLENGVLSRKLSDFGQETSYIEDNSNQNGAVSLIF PH4H_Mus_musculus
## [3] MSTAVLENPGLGRKLSDFGQETSYIEDNCNQNGAISLIF PH4H_Homo_sapiens
## [4] MSALVLESRALGRKLSDFGQETSYIEGNSDQN-AVSLIF PH4H_Bos_taurus
## [5] ----- PH4H_Chromobacter...
## [6] ----- PH4H_Ralstonia_so...
## [7] ----- PH4H_Caulobacter_...
## [8] ----- PH4H_Pseudomonas_...
## [9] ----- PH4H_Rhizobium_loti
## Con ----- Consensus
##
##      aln (40..78)                      names
## [1] SLKEEVGALAKVLRRLFEEENDINLTHIESRPSRLNKDEYE PH4H_Rattus_norve...
## [2] SLKEEVGALAKVLRRLFEEENEINLTHIESRPSRLNKDEYE PH4H_Mus_musculus
## [3] SLKEEVGALAKVLRRLFEEENDVNLTTHIESRPSRLKKDEYE PH4H_Homo_sapiens
## [4] SLKEEVGALARVLRRLFEEENDINLTHIESRPSRLRKDEYE PH4H_Bos_taurus
```

```

## [5] ----- PH4H_Chromobacter...
## [6] ----- PH4H_Ralstonia_so...
## [7] ----- PH4H_Caulobacter_...
## [8] ----- PH4H_Pseudomonas_...
## [9] ----- PH4H_Rhizobium_loti
## Con ----- Consensus
##
##      aln (79..117)                                names
## [1] FFTYLDKRTKPVLSGSIKSLRNDIGATVHELSDKEKNT PH4H_Rattus_norve...
## [2] FFTYLDKRSKPVLSGSIKSLRNDIGATVHELSDKEKNT PH4H_Mus_musculus
## [3] FFTHLDKRSLPALTNIIKILRHDIGATVHELSDKKKDT PH4H_Homo_sapiens
## [4] FFTNLDQRSVPALANI KILRHDIGATVHELSDKKKDT PH4H_Bos_taurus
## [5] ----- PH4H_Chromobacter...
## [6] ----- PH4H_Ralstonia_so...
## [7] ----- PH4H_Caulobacter_...
## [8] ----- PH4H_Pseudomonas_...
## [9] ----- PH4H_Rhizobium_loti
## Con ----- Consensus
##
##      aln (118..156)                                names
## [1] VPWFPRTIQELDRFANQILSYGAELDADHPGFKDPVYRA PH4H_Rattus_norve...
## [2] VPWFPRTIQELDRFANQILSYGAELDADHPGFKDPVYRA PH4H_Mus_musculus
## [3] VPWFPRTIQELDRFANQILSYGAELDADHPGFKDPVYRA PH4H_Homo_sapiens
## [4] VPWFPRTIQELDNFANQVLSYGAELDADHPGFKDPVYRA PH4H_Bos_taurus
## [5] -----MNDRADFVVPD-----ITTRKNVG PH4H_Chromobacter...
## [6] -----MAIATPTSAAPTAPAGFTGTLTDKLREQ PH4H_Ralstonia_so...
## [7] -----MSG-----DGLSNG PH4H_Caulobacter_...
## [8] -----MKTQTQY PH4H_Pseudomonas_...
## [9] -----MSVAEYAR-----DCAAQG PH4H_Rhizobium_loti
## Con -----?????????Y????D?????D????? Consensus
##
##      aln (157..195)                                names
## [1] RRKQFADIAYNRYRHGQPIPRVEYTEEEKQWGTVFRTLK PH4H_Rattus_norve...
## [2] RRKQFADIAYNRYRHGQPIPRVEYTEEEKQWGTVFRTLK PH4H_Mus_musculus
## [3] RRKQFADIAYNRYRHGQPIPRVEYMEEEKQWGTVFRTLK PH4H_Homo_sapiens
## [4] RRKQFADIAYNRYRHGQPIPRVEYTEEEKQWGTVFRTLK PH4H_Bos_taurus
## [5] LSHDAN-----DFTLPQPLDRYSAEDHATWATLYQRQC PH4H_Chromobacter...
## [6] FAEGLDGQTLRPDFTMEQPVHRYTAADHATWRTLYDRQE PH4H_Ralstonia_so...
## [7] PPPGAR-----PDWTIDQGWETYTQAEDVWITLYERQT PH4H_Caulobacter_...
## [8] VARQPD-----DNGFIHYPETEHQVWNTLITRQL PH4H_Pseudomonas_...
## [9] LRGDYS--VCRADFTVAQDYD--YSDEEQAVWRTLCDRQT PH4H_Rhizobium_loti
## Con ?R?Q????????????P?P???YTEEE??TW?TL??RQ? Consensus
##
##      aln (196..234)                                names
## [1] ALYKTHACYEHNHIFPLLEKYCGFREDNIPQLEDVSQFL PH4H_Rattus_norve...
## [2] ALYKTHACYEHNHIFPLLEKYCGFREDNIPQLEDVSQFL PH4H_Mus_musculus

```



```

## [3] SLYKTHACYEYNHIFPILLEKYCGFHEDNIPQLEDVSQFL PH4H_Homo_sapiens
## [4] SLYKTHACYEHNHIFPILLEKYCGFREDNIPQLEEVSQFL PH4H_Bos_taurus
## [5] KLLPGRACDEFMEGL-----ERLEVDADRVPDFNKLNQKL PH4H_Chromobacter...
## [6] ALLPGRACDEFQGL-----STLGMSREGVPSFDRLNETL PH4H_Ralstonia_so...
## [7] DMLHGRACDEFMRGL----DALDLHRSRGIPDFARINEEL PH4H_Caulobacter_...
## [8] KVIEGRACQEYLDGI----EQLGLPHERIPQLDEINRVL PH4H_Pseudomonas_...
## [9] KLTRKLAHHSYLDGV----EKLGL-LDRIPDFEDVSTKL PH4H_Rhizobium_loti
## Con ?L????AC?E???G?-----??LG???D?IPQLE?VSQ?L Consensus
##
##      aln (235..273)                                names
## [1] QTCTGFRLRPVAGLLSSRDFLGGLAFRVFHCTQYIRHGS PH4H_Rattus_norve...
## [2] QTCTGFRLRPVAGLLSSRDFLGGLAFRVFHCTQYIRHGS PH4H_Mus_musculus
## [3] QTCTGFRLRPVAGLLSSRDFLGGLAFRVFHCTQYIRHGS PH4H_Homo_sapiens
## [4] QSCTGFRLRPVAGLLSSRDFLGGLAFRVFHCTQYIRHGS PH4H_Bos_taurus
## [5] MAATGWKIVAVPGLIPDDVFFFEHLANRRFPVTWWLREPH PH4H_Chromobacter...
## [6] MRATGWQIVAVPGLVPDEVFFFEHLANRRFPASWWMRRPD PH4H_Ralstonia_so...
## [7] KRLTGWTIVAVPGLVPDDVFFDHLANRRFPAGQFIRKPH PH4H_Caulobacter_...
## [8] QATTGWRVARVPALIPFQTFELLASQQFPVATFIRTPE PH4H_Pseudomonas_...
## [9] RKLTGWEIIAVPGLIPAAPFFDHLANRRFPVTNWLRTQ PH4H_Rhizobium_loti
## Con Q???TGWR???VPGL?P???FF??LA?R?FP?TQ?IR??? Consensus
##
##      aln (274..312)                                names
## [1] KPMYTPEPDICHELLGHVPLFSDRSFAQFSQEIG-LASL PH4H_Rattus_norve...
## [2] KPMYTPEPDICHELLGHVPLFSDRSFAQFSQEIG-LASL PH4H_Mus_musculus
## [3] KPMYTPEPDICHELLGHVPLFSDRSFAQFSQEIG-LASL PH4H_Homo_sapiens
## [4] KPMYTPEPDICHELLGHVPLFSDRSFAQFSQEIG-LASL PH4H_Bos_taurus
## [5] QLDYLQEPDVFHDLFGHVPLLINPVFADYLEAYGKGGVK PH4H_Chromobacter...
## [6] QLDYLQEPDGFHDIFGHVPLLINPVFADYMQAYGQGGLK PH4H_Ralstonia_so...
## [7] ELDYLQEPDIFHDVFGHVPLMTDPVFADYMQAYGEGGRR PH4H_Caulobacter_...
## [8] ELDYLQEPDIFHEIFGHCPLLTNPWFAEFTHTYKGLGLK PH4H_Pseudomonas_...
## [9] ELDYIVEPDMFHDFFGHVPVLSQPVFADFMQMYGKKAGD PH4H_Rhizobium_loti
## Con ?LDY??EPDIFHELFGHVPLLSDP?FA?F?Q?YG?LA?? Consensus
##
##      aln (313..351)                                names
## [1] GAPDEYIEKLATIIYWFTVEFGLCKEG-DSIKAYGAGLLS PH4H_Rattus_norve...
## [2] GAPDEYIEKLATIIYWFTVEFGLCKEG-DSIKAYGAGLLS PH4H_Mus_musculus
## [3] GAPDEYIEKLATIIYWFTVEFGLCKQG-DSIKAYGAGLLS PH4H_Homo_sapiens
## [4] GAPDEYIEKLATIIYWFTVEFGLCKQG-DSIKAYGAGLLS PH4H_Bos_taurus
## [5] AKALGALPMLARLYWYTVEFGLINTP-AGMRIYGAGILS PH4H_Chromobacter...
## [6] AARLGALDMLARLYWYTVEFGLIRTP-AGLRIYGAGIVS PH4H_Ralstonia_so...
## [7] ALGLGRLANLARLYWYTVEFGLMNTP-AGLRIYGAGIVS PH4H_Caulobacter_...
## [8] ASKE-ERVFLARLYWMTIEFGLVETD-QGKRIYGGGILS PH4H_Pseudomonas_...
## [9] IIALGGDEMITRLYWYTAEYGLVQEAGQPLKAFGAGLMS PH4H_Rhizobium_loti
## Con ?A?????E?LARLYW?TVEFGL????-???KAYGAGLLS Consensus
##
##      aln (352..390)                                names

```

```

## [1] SFGELQYCLSD-KPKLLPLELEKTACQEYSVTEFQPLY PH4H_Rattus_norve...
## [2] SFGELQYCLSD-KPKLLPLELEKTACQEYTVTEFQPLY PH4H_Mus_musculus
## [3] SFGELQYCLSE-KPKLLPLELEKTAIQNYTVTEFQPLY PH4H_Homo_sapiens
## [4] SFGELQYCLSD-KPKLLPLELEKTAVQEYTITEFQPLY PH4H_Bos_taurus
## [5] SKSESIYCLDSASPNRVGFDLMRIMNTRYRIDTFQKTYF PH4H_Chromobacter...
## [6] SKSESVYALDSASPNRIGFDVHRIMRTRYRIDTFQKTYF PH4H_Ralstonia_so...
## [7] SRTESIFALDDPSPNRIGFDLVRMRTLYRIDDFQQVYF PH4H_Caulobacter_...
## [8] SPKETVYSLSD-EPLHQAFNPLEAMRTPYRIDILQPLYF PH4H_Pseudomonas_...
## [9] SFTELQFAVEGKDAHHVPFDLETVMRTGYEIDKFQRAYF PH4H_Rhizobium_loti
## Con SF?ELQYCLSD-?P???PF?LE??M?T?Y?ID?FQPLYF Consensus
##
##      aln (391..429)                      names
## [1] VAESFSDAKEKVRTFAATIPRPFVRYDPYTQRVEVLN PH4H_Rattus_norve...
## [2] VAESFNDAKEKVRTFAATIPRPFVRYDPYTQRVEVLN PH4H_Mus_musculus
## [3] VAESFNDAKEKVRNFAATIPRPFVRYDPYTQRIEVLN PH4H_Homo_sapiens
## [4] VAESFNDAKEKVRNFAATIPRPFVHYDPYTQRIEVLN PH4H_Bos_taurus
## [5] VIDSFKQLFDATA-PDFAPLYLQLADAQPWGAGDVAPDD PH4H_Chromobacter...
## [6] VIDSFEQLFDATR-PDFTPLYEALGTLPTFGAGDVVDGD PH4H_Ralstonia_so...
## [7] VIDSIQTLQEVTL-RDFGAIYERLASVSDIGVAEIVPGD PH4H_Caulobacter_...
## [8] VLPDLKRLFQLAQ-EDIMALVHEAMRLG-LHAPLFPKQ PH4H_Pseudomonas_...
## [9] VLPSFDALRDAFQTADFEAIVARRKDQKALDPATV---- PH4H_Rhizobium_loti
## Con V??SF??L?E??R??D?T?????????P??????V?D? Consensus
##
##      aln (430..456)                      names
## [1] TQQLKILADSINSEVGILCNALQKIKS PH4H_Rattus_norve...
## [2] TQQLKILADSINSEVGILCHALQKIKS PH4H_Mus_musculus
## [3] TQQLKILADSINSEIGILCSALQKIK- PH4H_Homo_sapiens
## [4] TQQLKILADSISSVEILCSALQKIK- PH4H_Bos_taurus
## [5] LVLNAGDRQGWADEDV----- PH4H_Chromobacter...
## [6] AVLNAGTREGWADTADI----- PH4H_Ralstonia_so...
## [7] AVLTRGT-QAYATAGGRLAGAAAAG--- PH4H_Caulobacter_...
## [8] AA----- PH4H_Pseudomonas_...
## [9] ----- PH4H_Rhizobium_loti
## Con ??????????????IL??A???--- Consensus

print(myFirstAlignment, showConsensus=FALSE, halfNrow=3)

## CLUSTAL 2.1
##
## Call:
##      msa(mySequences)
##
## MsaAAMultipleAlignment with 9 rows and 456 columns
##      aln                      names
## [1] MAAVLENGVLSRKLSD...SINSEVGILCNALQKIKS PH4H_Rattus_norve...
## [2] MAAVLENGVLSRKLSD...SINSEVGILCHALQKIKS PH4H_Mus_musculus

```

```

## [3] MSTAVLENPGLGRKLSDF...SINSEIGILCSALQKIK- PH4H_Homo_sapiens
## ... ...
## [7] -----...AYATAGGRLAGAAAG--- PH4H_Caulobacter_...
## [8] -----...----- PH4H_Pseudomonas_...
## [9] -----...----- PH4H_Rhizobium_loti

print(myFirstAlignment, showNames=FALSE, show="complete")

##
## MsaAAMultipleAlignment with 9 rows and 456 columns
##     aln (1..60)
## [1] MAAVLENGVLSRKLSDFGQETSYIEDNSNQNGAISLIFSLKEEVGALAKVLRLEENDI
## [2] MAAVLENGVLSRKLSDFGQETSYIEDNSNQNGAVSLIFSLKEEVGALAKVLRLEENEI
## [3] MSTAVLENPGLGRKLSDFGQETSYIEDNCNQNGAISLIFSLKEEVGALAKVLRLEENDV
## [4] MSALVLESRALGRKLSDFGQETSYIEGNSDQN-AVSLIFSLKEEVGALARVLRLEENDI
## [5] -----
## [6] -----
## [7] -----
## [8] -----
## [9] -----
## Con -----
##
##     aln (61..120)
## [1] NLTHIESRPSRLNKDEYEFFTYLDKRTKPVLGSIKSLRNDIGATVHELSDKEKNTVPW
## [2] NLTHIESRPSRLNKDEYEFFTYLDKRSKPVLGSIKSLRNDIGATVHELSDKEKNTVPW
## [3] NLTHIESRPSRLKKDEYEFFTHLDKRSPLALTNIKILRHDIGATVHELSDKKKDTVPW
## [4] NLTHIESRPSRLRKDEYEFFTNLDQRSVPALANI KILRHDIGATVHELSDKKKDTVPW
## [5] -----
## [6] -----
## [7] -----
## [8] -----
## [9] -----
## Con -----
##
##     aln (121..180)
## [1] FPRTIQELDRFANQILSYGAELDADHPGFKDPVYRARRKQFADIAYNYRHGQPIPRVEYT
## [2] FPRTIQELDRFANQILSYGAELDADHPGFKDPVYRARRKQFADIAYNYRHGQPIPRVEYT
## [3] FPRTIQELDRFANQILSYGAELDADHPGFKDPVYRARRKQFADIAYNYRHGQPIPRVEYM
## [4] FPRTIQELDNFANQVLSYGAELDADHPGFKDPVYRARRKQFADIAYNYRHGQPIPRVEYT
## [5] -----MNDRADFVVPD-----ITTRKNVGLSHDAN-----DFTLPQPLDRYS
## [6] -----MAIATPTSAAPTAPAGFTGTLTDKLREQFAEGLDGQTLRPDFTMEQPVHRYT
## [7] -----MSG-----DGLSNGPPPGAR-----PDWTIDQGWEITYT
## [8] -----MKTQYVARQPD-----DNGFIHYP
## [9] -----MSVAEYAR-----DCAAQGLRGDYS--VCRADFTVAQDYD-YS
## Con -----????????Y????D?????D?????R?Q????????P?P???YT
##

```

```

##      aln (181..240)
## [1] EEEKQWGTVFRTLKALYKTHACYEHNHIFPILLEKYCGFREDNIPQLEDVVSQFLQTCTGF
## [2] EEERKTWGTVFRTLKALYKTHACYEHNHIFPILLEKYCGFREDNIPQLEDVVSQFLQTCTGF
## [3] EEEKKTWGTVFKTLKSLYKTHACYEYNNHIFPILLEKYCGFHEDNIPQLEDVVSQFLQTCTGF
## [4] EEEKKTWGTVFRTLKSLYKTHACYEHNHIFPILLEKYCGFREDNIPQLEEVVSQFLQSCTGF
## [5] AEDHATWATLYQRQCKLLPGRACDEFMEGL-----ERLEVDADRVPDFNKNLQKLMAATGW
## [6] AADHATWRTLYDRQEALLPGRACDEFQLGL-----STLGMSREGVPSFDRLNETLMRATGW
## [7] QAEHDVWITLYERQTDMLHGRACDEFMRGL-----DALDLHRSGIPDFARINEELKRLTGW
## [8] ETEHQVWNTLITRQLKVIEGRACQEYLDGI-----EQLGLPHERIPQLDEINRVLQATTGW
## [9] DEEQAVWRTLCDRQTKLTRKLAHHSYLDGV-----EKLGL-LDRIPDFEDVSTKLRLKTGW
## Con EEE??TW?TL??RQ??L????AC?E???G?-----??LG???D?IPQLE?VSQ?LQ??TGW
##
##      aln (241..300)
## [1] RLRPVAGLLSSRDFLGGLAFRVFHCTQYIRHGSKPMYTPEPDICHELLGHVPLFSDRSFA
## [2] RLRPVAGLLSSRDFLGGLAFRVFHCTQYIRHGSKPMYTPEPDICHELLGHVPLFSDRSFA
## [3] RLRPVAGLLSSRDFLGGLAFRVFHCTQYIRHGSKPMYTPEPDICHELLGHVPLFSDRSFA
## [4] RLRPVAGLLSSRDFLGGLAFRVFHCTQYIRHGSKPMYTPEPDICHELLGHVPLFSDRSFA
## [5] KIVAVPGLIPDDVFFEHLANRRFPVTWWLREPHQLDYLQEPDVFDLFGHVPLLINPVFA
## [6] QIVAVPGLVPDEVFFEHLANRRFPASWMMRPDQLDYLQEPDGFHDIFGHVPLLINPVFA
## [7] TVVAVPGLVPDDVFFDHLANRRFPAGQFIRKPHELDYLQEPDIFHDVFGHVPMLTDPVFA
## [8] RVARVPALIPFQTFFELLASQQFPVATFIRTPEELDYLQEPDIFHEIFGHCPLLTNPWFA
## [9] EIIAVPGLIPAAPFFDHLANRRFPVTNWLRTREQELDYIVEPDMFHDFFGHVPVLSQPVFA
## Con R???VPGL?P???FF??LA?R?FP?TQ?IR????LDY??EPDIFHELFGHVPLLSDP?FA
##
##      aln (301..360)
## [1] QFSQEIG-LASLGAPDEYIEKLATIIYWFTVEFGLCKEG-DSIKAYGAGLLSSFGEQYCL
## [2] QFSQEIG-LASLGAPDEYIEKLATIIYWFTVEFGLCKEG-DSIKAYGAGLLSSFGEQYCL
## [3] QFSQEIG-LASLGAPDEYIEKLATIIYWFTVEFGLCKQG-DSIKAYGAGLLSSFGEQYCL
## [4] QFSQEIG-LASLGAPDEYIEKLATIIYWFTVEFGLCKQG-DSIKAYGAGLLSSFGEQYCL
## [5] DYLEAYGKGGVKAKALGALPMLARLYWYTVEFGLINTP-AGMRIYGAGILSSKSESIYCL
## [6] DYMQAYGQGGKKAARLGALDMLARLYWYTVEFGLIRTP-AGLRIYGAGIVSSKSESIVYAL
## [7] DYMQAYGEGRRALGLGRLANLARLYWYTVEFGLMNT-AGLRIYGAGIVSSRTESIFAL
## [8] EFTHTYGLGLKASKE-ERVFLARLYWMTIEFGLVETD-QGKRIYGGGILSSPKETVYSL
## [9] DFMQMYGKKAGDIIALGGDEMITRLYWYTAEYGLVQEAGQPLKAFGAGLMSSFTTELQFAV
## Con ?F?Q?YG?LA???A?????E?LARLYW?TVEFGL????-???KAYGAGLLSSF?ELQYCL
##
##      aln (361..420)
## [1] SD-KPKLLPLELEKTACQEYSVTEFQPLYYVAESFSDAKEKVRTFAATIPRPFVRYDPY
## [2] SD-KPKLLPLELEKTACQEYTVTEFQPLYYVAESFNDAKEKVRTFAATIPRPFVRYDPY
## [3] SE-KPKLLPLELEKTAIQNYTVTEFQPLYYVAESFNDAKEKVRNFAATIPRPFVRYDPY
## [4] SD-KPKLLPLELEKTAVQEYTTITEFQPLYYVAESFNDAKEKVRNFAATIPRPFVHYDPY
## [5] DSASPNRVGFDLMRIMNTRYRIDTFQKTYFVIDSFKQLFDATA-PDFAPLYLQLADAQPW
## [6] DSASPNRIGFDVHRIMRTRYRIDTFQKTYFVIDSFEQLFDATR-PDFTPLYEALGTLPTF
## [7] DDPSPNRIGFDLERVMRTLYRIDDFQQVYFVIDSIQTLQEVTL-RDFGAIYERLASVSDI
## [8] SD-EPLHQAFNPLEAMRTPYRIDILQPLYFVLPDLKRLFQLAQ-EDIMALVHEAMRLG-L
## [9] EGKDAHHPFDLETVMRTGYEIDKFQRAYFVLPSPFDALRDAFQTADFEAIVARRKDQKAL

```

```
## Con SD-?P???PF?LE?M?T?Y?ID?FQPLYFV??SF??L?E??R??D?T??????????P?
##
##      aln (421..456)
## [1] TQRVEVLDNTTQQLKILADSINSEVGILCNALQKIKS
## [2] TQRVEVLDNTTQQLKILADSINSEVGILCHALQKIKS
## [3] TQRIEVLDTNTQQLKILADSINSEIGILCSALQKIK-
## [4] TQRIEVLDTNTQQLKILADSSISSEVEILCSALQKLK-
## [5] GAGDVAPDDLVLNAGDRQGWADTEDV-----
## [6] GAGDVVDGDAVLNAGTREGWADTADI-----
## [7] GVAEIVPGDAVLTRGT-QAYATAGGRLAGAAAG---
## [8] HAPLFPPKQAA-----
## [9] DPATV-----
## Con ?????V?D?????????????????IL??A???---
```

## 6 Processing Multiple Alignments

### 6.1 Methods Inherited From Biostrings

The classes defined by the `msa` package for storing multiple alignment results have been derived from the corresponding classes defined by the `Biostrings` package. Therefore, all methods for processing multiple alignments are available and work without any practical limitation. In this section, we highlight some of these.

The classes used for storing multiple alignments allow for defining masks on sequences and sequence positions via their row and column mask slots. They can be set by `rowmask()` and `colmask()` functions which serve both as setter and getter functions. To set row or column masks, an `IRanges` object must be supplied:

```
myMaskedAlignment <- myFirstAlignment
rowM <- IRanges(start=1, end=2)
rowmask(myMaskedAlignment) <- rowM
myMaskedAlignment

## CLUSTAL 2.1
##
## Call:
##      msa(mySequences)
##
## MsaAAMultipleAlignment with 9 rows and 456 columns
##      aln                                     names
## [1] #####...##### PH4H_Rattus_norve...
## [2] #####...##### PH4H_Mus_musculus
## [3] MSTAVLENPGLGRKLSDF...SINSEIGILCSALQKIK- PH4H_Homo_sapiens
## [4] MSALVLESRALGRKLSDF...SISSEVEILCSALQKLK- PH4H_Bos_taurus
## [5] -----...GWADTEDV----- PH4H_Chromobacter...
```

```
## [6] -----...GWADTADI----- PH4H_Ralstonia_so...
## [7] -----...AYATAGGRLAGAAAG--- PH4H_Caulobacter...
## [8] -----...----- PH4H_Pseudomonas...
## [9] -----...----- PH4H_Rhizobium_loti
## Con -----...???????IL??A???--- Consensus
```

The `unmasked()` allows for removing these masks, thereby casting the multiple alignment to a set of aligned Biostrings sequences (class `AAStringSet`, `DNAStringSet`, or `RNAStringSet`):

```
unmasked(myMaskedAlignment)
```

```
## A AAStringSet instance of length 9
## width seq names
## [1] 456 MAAVVLENGVLSRKLS...SEVGILCNALQKIKS PH4H_Rattus_norve...
## [2] 456 MAAVVLENGVLSRKLS...SEVGILCHALQKIKS PH4H_Mus_musculus
## [3] 456 MSTAVLENPGLGRKLS...SEIGILCSALQKIK- PH4H_Homo_sapiens
## [4] 456 MSALVLESRALGRKLS...SEVEILCSALQKLK- PH4H_Bos_taurus
## [5] 456 -----...DTEDV----- PH4H_Chromobacter...
## [6] 456 -----...DTADI----- PH4H_Ralstonia_so...
## [7] 456 -----...TAGGRLAGAAAG--- PH4H_Caulobacter...
## [8] 456 -----...----- PH4H_Pseudomonas...
## [9] 456 -----...----- PH4H_Rhizobium_loti
```

Consensus matrices can be computed conveniently as follows:

```
conMat <- consensusMatrix(myFirstAlignment)
dim(conMat)
```

```
## [1] 21 456
```

```
conMat[, 101:110]
```

```
## [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
## - 5 5 5 5 5 5 5 5 5 5
## A 0 0 0 4 0 0 0 0 0 0
## C 0 0 0 0 0 0 0 0 0 0
## D 4 0 0 0 0 0 0 0 0 0
## E 0 0 0 0 0 0 0 4 0 0
## F 0 0 0 0 0 0 0 0 0 0
## G 0 0 4 0 0 0 0 0 0 0
## H 0 0 0 0 0 0 4 0 0 0
## I 0 4 0 0 0 0 0 0 0 0
## K 0 0 0 0 0 0 0 0 0 0
## L 0 0 0 0 0 0 0 0 4 0
```

```
## M    0    0    0    0    0    0    0    0    0    0
## N    0    0    0    0    0    0    0    0    0    0
## P    0    0    0    0    0    0    0    0    0    0
## Q    0    0    0    0    0    0    0    0    0    0
## R    0    0    0    0    0    0    0    0    0    0
## S    0    0    0    0    0    0    0    0    0    4
## T    0    0    0    0    4    0    0    0    0    0
## V    0    0    0    0    0    4    0    0    0    0
## W    0    0    0    0    0    0    0    0    0    0
## Y    0    0    0    0    0    0    0    0    0    0
```

Note that `consensusMatrix()` cannot handle alignments with active masks. So, the masks in multiple alignment objects must be removed prior to the computation of the consensus matrix:

```
conMat <- consensusMatrix(unmasked(myMaskedAlignment))
```

Consensus strings can be computed from consensus matrices:

```
## auxiliary function for splitting a string into displayable portions
printSplitString <- function(x, width=getOption("width") - 1)
{
  starts <- seq(from=1, to=nchar(x), by=width)

  for (i in 1:length(starts))
    cat(substr(x, starts[i], starts[i] + width - 1), "\n")
}

printSplitString(consensusString(conMat))

## -----
## -----?
## ?????????Y????D??????D??????R?Q????????????P?P???YTEEE??TW?TL??
## RQ??L????AC?E???G?----??LG???D?IPQLE?VSQ?LQ??TGWR???VPGL?P???FF?
## ?LA?R?FP?TQ?IR????LDY??EPDIFHELFGHVPLLSDP?FA?F?Q?YG?LA???A?????E
## ?LARLYW?TVEFGL????-???KAYGAGLLSSF?ELQYCLSD-?P???PF?LE??M?T?Y?ID?
## FQPLYFV??SF??L?E??R??D?T??????????P??????V?D????????????????IL?
## ?A???---
```

Consensus sequences can also be computed directly without computing intermediate consensus matrices. However, the `consensusString()` function cannot handle the masks contained in the multiple alignment objects (no matter whether there are active masks or not). Therefore, it is necessary to remove the masks beforehand:

```
printSplitString(consensusString(unmasked(myFirstAlignment)))

## -----
## -----?
## ????????Y???D?????D?????R?Q????????????P?P???YTEEE??TW?TL??
## RQ??L????AC?E???G?----??LG???D?IPQLE?VSQ?LQ??TGWR???VPGL?P???FF?
## ?LA?R?FP?TQ?IR????LDY??EPDIFHELFGHVPLLSDP?FA?F?Q?YG?LA???A?????E
## ?LARLYW?TVEFGL????-???KAYGAGLLSSF?ELQYCLSD-?P???PF?LE??M?T?Y?ID?
## FQPLYFV??SF??L?E??R??D?T??????????P??????V?D????????????????IL?
## ?A???---
```

```
printSplitString(consensusString(unmasked(myMaskedAlignment)))

## -----
## -----?
## ????????Y???D?????D?????R?Q????????????P?P???YTEEE??TW?TL??
## RQ??L????AC?E???G?----??LG???D?IPQLE?VSQ?LQ??TGWR???VPGL?P???FF?
## ?LA?R?FP?TQ?IR????LDY??EPDIFHELFGHVPLLSDP?FA?F?Q?YG?LA???A?????E
## ?LARLYW?TVEFGL????-???KAYGAGLLSSF?ELQYCLSD-?P???PF?LE??M?T?Y?ID?
## FQPLYFV??SF??L?E??R??D?T??????????P??????V?D????????????????IL?
## ?A???---
```

Actually, the `print()` method (see Section ?? above) uses this function to compute the consensus sequence.

## 6.2 Interfacing to Other Packages

There are also other sequence analysis packages that use or make use of multiple sequence alignments. The `msa` package does not directly interface to any of these packages in order to avoid dependencies and possible incompatibilities. However, `msa` provides a function `msaConvert()` that allows for converting multiple sequence alignment objects to other types/classes. Currently, two such conversions are available, namely to objects of class `alignment` (as defined and used by the `seqinr` package [?]) and to objects of class `align` (as defined and used by the `bios2mds` package [?]). Note that the conversion is performed without loading or depending on the respective packages.

In the following example, we perform a multiple alignment of Hemoglobin alpha example sequences and convert the result for later processing with the `seqinr` package:

```
hemoSeq <- readAAStringSet(system.file("examples/HemoglobinAA.fasta",
                                         package="msa"))
hemoAln <- msa(hemoSeq)

## use default substitution matrix
```



```

hemoAln

## CLUSTAL 2.1
##
## Call:
##   msa(hemoSeq)
##
## MsaAAMultipleAlignment with 17 rows and 143 columns
##      aln                                names
## [1] -VLSPADKTNVKAAGKV...LDKFLASVSTVLTSKYR HBA1_Homo_sapiens
## [2] MVLSPADKTNVKAAGKV...LDKFLASVSTVLTSKYR HBA1_Pan_troglodytes
## [3] -VLSPADKSNVKAAGKV...LDKFLASVSTVLTSKYR HBA1_Macaca_mulatta
## [4] -VLSAADKGNVKAAGKV...LDKFLANVSTVLTSKYR HBA1_Bos_taurus
## [5] -VLSPADKTNVKGTSKI...LDKFLASVSTVLTSKYR HBA1_Tursiops_tru...
## [6] -VLSGEDKSNIAAGVKI...LDKFLASVSTVLTSKYR HBA1_Mus_musculus
## [7] MVLSADKTNIKNCWGKI...LDKFLASVSTVLTSKYR HBA1_Rattus_norve...
## [8] -VLSATDKANVKTFWGKL...LDKFLATVATVLTSKYR HBA1_Erinaceus_eu...
## [9] -VLSAADKSNVKACWGKI...LDKFFSAVSTVLTSKYR HBA1_Felis_silves...
## [10] -VLSPADKTNIKSTWDKI...LDKFFTAVSTVLTSKYR HBA1_Chrysocyon_b...
## [11] -VLSNDKTNVKATWSKV...LDKFLSNVSTVLTSKYR HBA1_Loxodonta_af...
## [12] -VLSAADKTNVKAWSKV...LDKFLALLSTVLTSKYR HBA1_Monodelphis_...
## [13] -MLTDAEKKEVTALWGKA...MDKFLSKVATVLTSKYR HBA1_Ornithorhync...
## [14] -VLSAADKNNVGIFTKI...LDKFLCAVGTVLTAKYR HBA1_Gallus_gallus
## [15] -HLTADKKHIKAIWPSV...LDKFLVSVSNVLTSKYR HBA1_Xenopus_trop...
## [16] -VLTEEDKARVRVAWVPV...VDKFLGQISKVLASRYR HBA1_Microcephalo...
## [17] -SLSDTDKAVVKAIWAKI...VDKFFNNLALALSEKYR HBA1_Danio_rerio
## Con -VLS?ADK?NVKA?WGK?...LDKFLA?VSTVLTSKYR Consensus

hemoAln2 <- msaConvert(hemoAln, type="seqinr::alignment")

```

Now we compute a distance matrix using the `dist.alignment()` function from the `seqinr` package:

```

library(seqinr)

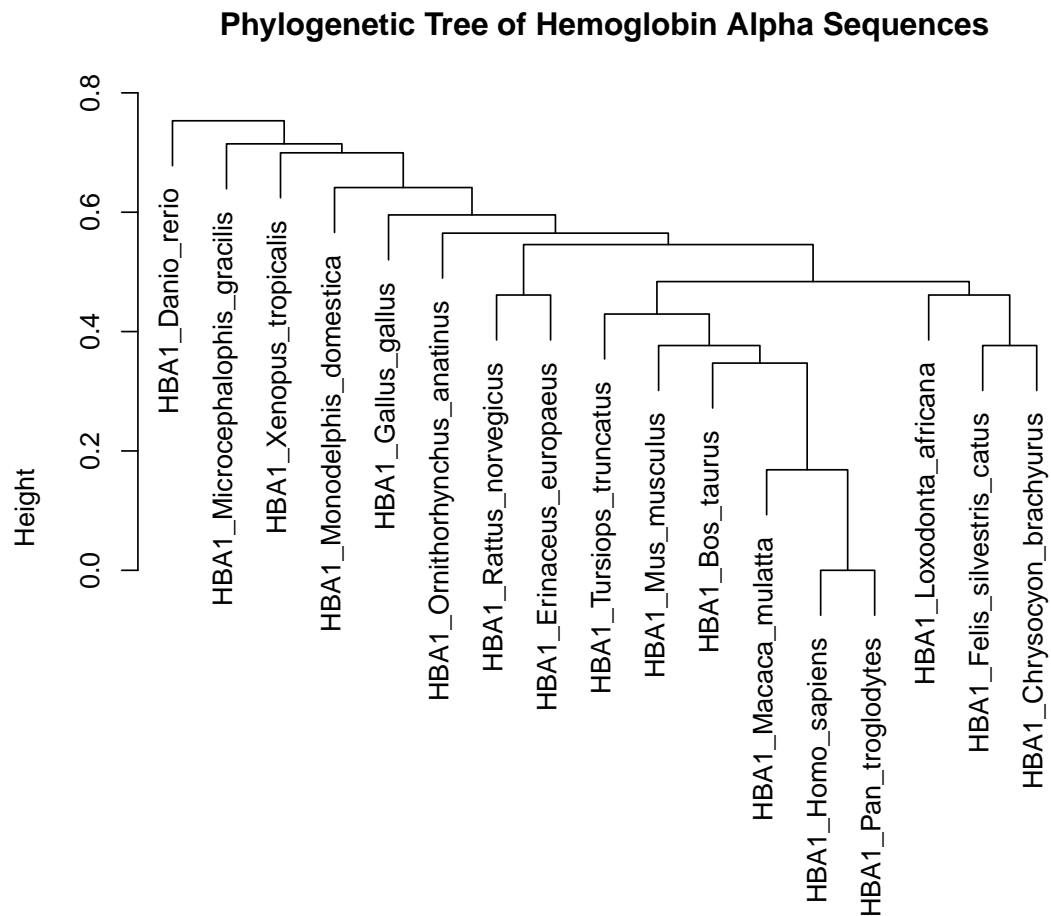
d <- dist.alignment(hemoAln2, "identity")
as.matrix(d)[3:4, 3:4]

##               HBA1_Macaca_mulatta HBA1_Bos_taurus
## HBA1_Macaca_mulatta           0.0000000      0.3368608
## HBA1_Bos_taurus              0.3368608      0.0000000

```

Now we can construct a draft phylogenetic tree using the `hclust()` function from the `stats` package:

```
hemoTree <- hclust(d)
plot(hemoTree, main="Phylogenetic Tree of Hemoglobin Alpha Sequences",
     xlab="", sub="")
```



The following example shows how to convert a multiple alignment object in an object of class `align` as defined by the `bios2mds` package:

```
hemoAln3 <- msaConvert(hemoAln, type="bios2mds::align")
str(hemoAln3)

## List of 17
## $ HBA1_Homo_sapiens      : chr [1:143] "-" "V" "L" "S" ...
## $ HBA1_Pan_troglodytes   : chr [1:143] "M" "V" "L" "S" ...
## $ HBA1_Macaca_mulatta    : chr [1:143] "-" "V" "L" "S" ...
```

```
## $ HBA1_Bos_taurus      : chr [1:143] "-" "V" "L" "S" ...
## $ HBA1_Tursiops_truncatus : chr [1:143] "-" "V" "L" "S" ...
## $ HBA1_Mus_musculus    : chr [1:143] "-" "V" "L" "S" ...
## $ HBA1_Rattus_norvegicus : chr [1:143] "M" "V" "L" "S" ...
## $ HBA1_Erinaceus_europaeus : chr [1:143] "-" "V" "L" "S" ...
## $ HBA1_Felis_silvestris_catus : chr [1:143] "-" "V" "L" "S" ...
## $ HBA1_Chrysocyon_brachyurus : chr [1:143] "-" "V" "L" "S" ...
## $ HBA1_Loxodonta_africana : chr [1:143] "-" "V" "L" "S" ...
## $ HBA1_Monodelphis_domestica : chr [1:143] "-" "V" "L" "S" ...
## $ HBA1_Ornithorhynchus_anatinus : chr [1:143] "-" "M" "L" "T" ...
## $ HBA1_Gallus_gallus    : chr [1:143] "-" "V" "L" "S" ...
## $ HBA1_Xenopus_tropicalis : chr [1:143] "-" "H" "L" "T" ...
## $ HBA1_Microcephalophis_gracilis: chr [1:143] "-" "V" "L" "T" ...
## $ HBA1_Danio_rerio      : chr [1:143] "-" "S" "L" "S" ...
## - attr(*, "class")= chr "align"
```

The conversions to the standard Biostrings classes are straightforward using standard `as()` methods and not provided by the `msaConvert()` function. The following example converts a multiple alignment object to class `BStringSet` (e.g. the `msaplot()` function from the `ggtree` package [?] accepts `BStringSet` objects):

```
hemoAln4 <- as(hemoAln, "BStringSet")
hemoAln4

## A BStringSet instance of length 17
##      width seq                      names
## [1]   143 -VLSPADKTNVKAAW...KFLASVSTVLTSKYR HBA1_Homo_sapiens
## [2]   143 MVLSPADKTNVKAAW...KFLASVSTVLTSKYR HBA1_Pan_troglodytes
## [3]   143 -VLSPADKSNVKAAW...KFLASVSTVLTSKYR HBA1_Macaca_mulatta
## [4]   143 -VLSAADKGNVKAAW...KFLANVSTVLTSKYR HBA1_Bos_taurus
## [5]   143 -VLSPADKTNVKGTW...KFLASVSTVLTSKYR HBA1_Tursiops_tru...
## ...   ...
## [13]  143 -MLTDAEKKEVTALW...KFLSKVATVLTSKYR HBA1_Ornithorhync...
## [14]  143 -VLSAADKNNVKGIF...KFLCAVGTVLTAKYR HBA1_Gallus_gallus
## [15]  143 -HLTADDKKHIKAIW...KFLVSVSNVLTASKYR HBA1_Xenopus_trop...
## [16]  143 -VLTEEDKARVRVAW...KFLGQISKVLASRYR HBA1_Microcephalo...
## [17]  143 -SLSDTDKAVVKAIW...KFFNNLALALSEKYR HBA1_Danio_rerio
```

**Note:** The `msaConvert()` function has been introduced in version 1.3.3 of the `msa` package. So, to have this function available, at least Bioconductor 3.3 is required, which requires at least R 3.3.0.

## 7 Pretty-Printing Multiple Sequence Alignments

As already mentioned above, the `msa` package offers the function `msaPrettyPrint()` which allows for pretty-printing multiple sequence alignments using the  $\text{\LaTeX}$  package `\text{\TeXshade}` [?]. Which prerequisites are necessary to take full advantage of the `msaPrettyPrint()` function is described in Section ??.

The `msaPrettyPrint()` function writes a multiple sequence alignment to an alignment (`.aln`) file and then creates  $\text{\LaTeX}$  code for pretty-printing the multiple sequence alignment on the basis of the  $\text{\LaTeX}$  package `\text{\TeXshade}`. Depending on the choice of the output argument, the function `msaPrettyPrint()` either prints a  $\text{\LaTeX}$  fragment to the R session (choice `output="asis"`) or writes a  $\text{\LaTeX}$  source file (choice `output="tex"`) that it processes to a DVI file (choice `output="dvi"`) or PDF file (choice `output="pdf"`). Note that no extra software is needed for choices `output="asis"` and `output="tex"`. For `output="dvi"` and `output="pdf"`, however, a  $\text{\TeX}/\text{\LaTeX}$  distribution must be installed in order to translate the  $\text{\LaTeX}$  source file into the desired target format (DVI or PDF).

The function `msaPrettyPrint()` allows for making the most common settings directly and conveniently via an R interface without the need to know the details of  $\text{\LaTeX}$  or `\text{\TeXshade}`. In the following, we will describe some of these customizations. For all possibilities, the user is referred to the documentation of `\text{\TeXshade}`.<sup>5</sup>

### 7.1 Consensus Sequence and Sequence Logo

The consensus sequence of the alignment is one of the most important results of a multiple sequence alignment. `msaPrettyPrint()` has a standard possibility to show this consensus sequence with the parameter `showConsensus`. The default value is `"bottom"`, which results in the following:

```
msaPrettyPrint(myFirstAlignment, output="asis", y=c(164, 213),
               subset=c(1:6), showNames="none", showLogo="none",
               consensusColor="ColdHot", showLegend=FALSE,
               askForOverwrite=FALSE)
```

```
IAYNYRHGQPIPRVEYTEEEKQTWGTVFRTLKALYKTHACYEHNHIFPLL 213
IAYNYRHGQPIPRVEYTEEEKRTWGTVFRTLKALYKTHACYEHNHIFPLL 213
IAYNYRHGQPIPRVEYMEEEEKKTWGTVFRTLKSLYKTHACYEYHNHIFPLL 213
IAYNYRHGQPIPRVEYTEEEKKTWGTVFRTLKSLYKTHACYEHNHIFPLL 212
.....DFTLPQPLDRYSAEDHATWATLYQRQCKLLPGRACDEFMEGL... 67
QTLRPDFTMEQPVHRYTAADHATWRTLYDRQEALLPGRACDEFLLQGL... 83
*****!*****!*****!*****!*****!*****!*****
```

Consensus sequences can also be displayed on top of a multiple sequence alignment or omitted completely.

<sup>5</sup><https://www.ctan.org/pkg/texshade>

In the above example, an exclamation mark ‘!’ in the consensus sequence stands for a conserved letter, i.e. a sequence positions in which all sequences agree, whereas an asterisk ‘\*’ stands for positions in which there is a majority of sequences agreeing. Positions in which the sequences disagree are left blank in the consensus sequence. For a more advanced example how to customize the consensus sequence, see the example in Subsection ?? below.

The color scheme of the consensus sequence can be configured with the `consensusColors` parameter. Possible values are "ColdHot", "HotCold", "BlueRed", "RedBlue", "GreenRed", "RedGreen", or "Gray". The above example uses the color scheme "RedGreen".

Additionally, `msaPrettyPrint()` also offers a more sophisticated visual representation of the consensus sequence — sequence logos. Sequence logos can be displayed either on top of the multiple sequence alignment (`showLogo="top"`), below the multiple sequence alignment (`showLogo="bottom"`), or omitted at all (`showLogo="none"`):

```
msaPrettyPrint(myFirstAlignment, output="asis", y=c(164, 213),
               subset=c(1:6), showNames="none", showLogo="top",
               logoColors="rasmol", shadingMode="similar",
               showLegend=FALSE, askForOverwrite=FALSE)
```



The color scheme of the sequence logo can be configured with the `logoColors` parameter. Possible values are "chemical", "rasmol", "hydropathy", "structure", "standard area", and "accessible area". The above example uses the color scheme "rasmol".

Note that a consensus sequence and a sequence logo can be displayed together, but only on opposite sides.

Finally, a caveat: for computing consensus sequences, `msaPrettyPrint()` uses the functionality provided by `TeXshade`, therefore, the results need not match to the results of the methods described in Section ?? above.

## 7.2 Color Shading Modes

`TeXshade` offers different shading schemes for displaying the multiple sequence alignment itself. The following schemes are available: "similar", "identical", and "functional". Moreover, there are five different color schemes available for shading: "blues", "reds", "greens", "grays", or "black". The following example uses the shading mode "similar" along with the color scheme "blues":

```
msaPrettyPrint(myFirstAlignment, output="asis", y=c(164, 213),
               showNames="none", shadingMode="similar",
               shadingColors="blues", showLogo="none",
               showLegend=FALSE, askForOverwrite=FALSE)
```

```

IAYNYRHHGQPIPRVEYTEEEKQTWGTVFRTLKALYKTHACYEHNHIFPLL 213
IAYNYRHHGQPIPRVEYTEEEKRTWGTVFRTLKALYKTHACYEHNHIFPLL 213
IAYNYRHHGQPIPRVEYMEEKKTWGTVFRTLKSLYKTHACYEYHNHIFPLL 213
IAYNYRHHGQPIPRVEYTEEEKKTWGTVFRTLKSLYKTHACYEHNHIFPLL 212
.....DFTLPQPLDRYS AEDHATWATLYQRQCKLLPGRACDEFMEGL... 67
QTLRPDFTMEQPVHRYTAADHATWRTLYDRQEALLPGRACDEFMQGL... 83
....PDWTIDQGWETYTQA EHDVWITLYERQTDMLHGRACDEFMRGL... 58
.....DNGFIHYPETEHQVWNTLITRQLKVI EGRACQEYLDGI... 50
.VCRADFTVAQDYD.YSDEEQAVWRTLCDRQTKLITRKL AHHSYLDGV... 65
          *** * !**** *!*** ** * * * !* **** **
```

If the shading modes "similar" or "identical" are used, the `shadingModeArg` argument allows for setting a similarity threshold (a numerical value between 0 and 100). For shading mode "functional", the following settings of the `shadingModeArg` argument are possible: "charge", "hydropathy", "structure", "hemical", "rasmol", "standard area", and "accessible area". The following example uses shading mode "functional" along with `shadingModeArg` set to "structure":

```
msaPrettyPrint(myFirstAlignment, output="asis", y=c(164, 213),
               showNames="none", shadingMode="functional",
               shadingModeArg="structure",
               askForOverwrite=FALSE)
```

```

IAYNYRHHGQPIPRVEYTEEEKQTWGTVFRTLKALYKTHACYEHNHIFPLL 213
IAYNYRHHGQPIPRVEYTEEEKRTWGTVFRTLKALYKTHACYEHNHIFPLL 213
IAYNYRHHGQPIPRVEYMEEKKTWGTVFRTLKSLYKTHACYEYHNHIFPLL 213
IAYNYRHHGQPIPRVEYTEEEKKTWGTVFRTLKSLYKTHACYEHNHIFPLL 212
.....DFTLPQPLDRYS AEDHATWATLYQRQCKLLPGRACDEFMEGL... 67
QTLRPDFTMEQPVHRYTAADHATWRTLYDRQEALLPGRACDEFMQGL... 83
....PDWTIDQGWETYTQA EHDVWITLYERQTDMLHGRACDEFMRGL... 58
.....DNGFIHYPETEHQVWNTLITRQLKVI EGRACQEYLDGI... 50
.VCRADFTVAQDYD.YSDEEQAVWRTLCDRQTKLITRKL AHHSYLDGV... 65
```

```

X external
X ambivalent
X internal
```

In the above example, a legend is shown that specifies the meaning of the color codes with which the letters are shaded. In some of the other examples above, we have suppressed this legend with the option `showLegend=FALSE`. The default, however, is that a legend is printed underneath the multiple sequence alignment like in the previous example.

### 7.3 Subsetting

In case that not the complete multiple sequence alignment should be printed, `msaPrettyPrint()` offers two ways of sub-setting. On the one hand, the `subset` argument allows for selecting only a subset of sequences. Not surprisingly, `subset` must be a numeric vector with indices of sequences to be selected. On the other hand, it is also possible to slice out certain positions of the multiple sequence alignment using the `y` argument. In the simplest case, `y` can be a numeric vector with two elements in ascending order which correspond to the left and right bounds between which the multiple sequence alignment should be displayed. However, it is also possible to slice out multiple windows. For this purpose, the argument `y` must be an `IRanges` object containing the starts and ends of the windows to be selected.

### 7.4 Additional Customizations

The `msaPrettyPrint()` function provides an interface to the most common functionality of `TeXshade` in a way that the user does not need to know the specific commands of `TeXshade`. `TeXshade`, however, provides a host of additional customizations many of which are not covered by the interface of the `msaPrettyPrint()` function. In order to allow users to make use of all functionality of `TeXshade`, `msaPrettyPrint()` offers the `furtherCode` argument through which users can add `LaTeX` code to the `texshade` environment that is created by `msaPrettyPrint()`. Moreover, the `code` argument can be used to bypass all of `msaPrettyPrint()`'s generation of `TeXshade` code.

Here is an example how to use the `furtherCode` argument in order to customize the consensus sequence and to show a ruler on top:

```
msaPrettyPrint(myFirstAlignment, output="asis", y=c(164, 213),
               subset=c(1:6), showNames="none", showLogo="none",
               consensusColor="ColdHot", showLegend=FALSE,
               shadingMode="similar", askForOverwrite=FALSE,
               furtherCode=c("\\defconsensus{.}{lower}{upper}",
                             "\\showruler{1}{top}"))
```

```

      170      180      190      200      210
IAYNYRHHGQPIPRVEYTEEEKQ TWGTVFRTLKALYKTHACYEHNHIFPLL 213
IAYNYRHHGQPIPRVEYTEEEKR TWGTVFRTLKALYKTHACYEHNHIFPLL 213
IAYNYRHHGQPIPRVEYMEEEKK TWGTVFRTLKSLYKTHACYEYNNHIFPLL 213
IAYNYRHHGQPIPRVEYTEEEKK TWGTVFRTLKSLYKTHACYEHNHIFPLL 212
.....DFTLPQPLDRYS AEDHATWATLYQRQCKLLPGRACDEFMEGL... 67
QTLRPDFTMEQPVHRYTAADHATWRTLYDRQEALLPGRACDEFLLQGL... 83
iaynyrhgqpiPrveYteeek.TWgTvfrtlk.LykthACyE.nhifpll
```

### 7.5 Sweave or knitr Integration

The function `msaPrettyPrint()` is particularly well-suited for pretty-printing multiple alignments in Sweave [?] or knitr [?] documents. The key is to set `output` to "asis" when calling

`msaPrettyPrint()` and, at the same time, to let the R code chunk produce output that is directly included in the resulting  $\text{\LaTeX}$  document as it is. This can be accomplished with the code chunk option `results="tex"` in Sweave and with the code chunk option `results="asis"` in knitr. Here is an example of a Sweave code chunk that displays a pretty-printed multiple sequence alignment inline:

```
<<AnyChunkName,results="tex">>=
msaPrettyPrint(myFirstAlignment, output="asis")
@
```

The same example in knitr:

```
<<AnyChunkName,results="asis">>=
msaPrettyPrint(myFirstAlignment, output="asis")
@
```

Note that, for processing the resulting  $\text{\LaTeX}$  source document, the  $\text{\TeX}$ **shade** package must be installed (see Section ??) and the  $\text{\TeX}$ **shade** package must be loaded in the preamble:

```
\usepackage{texshade}
```

## 7.6 Sequence Names

The `Biostrings` package does not impose any restrictions on the names of sequences. Consequently, `msa` also allows all possible ASCII strings as sequence (row) names in multiple alignments. As soon as `msaPrettyPrint()` is used for pretty-printing multiple sequence alignments, however, the sequence names are interpreted as plain  $\text{\LaTeX}$  source code. Consequently,  $\text{\LaTeX}$  errors may arise because of characters or words in the sequence names that  $\text{\LaTeX}$  does not or cannot interpret as plain text correctly. This particularly includes appearances of special characters and backslash characters in the sequence names.

The `msa` package offers a function `msaCheckNames()` which allows for finding and replacing potentially problematic characters in the sequence names of multiple alignment objects (see `?msaCheckNames`). However, the best solution is to check sequence names carefully and to avoid problematic sequence names from the beginning. Note, moreover, that too long sequence names will lead to less appealing outputs, so users are generally advised to consider sequence names carefully.

## 7.7 Pretty-Printing Wide Alignments

If the alignment to be printed with `msaPrettyPrint()` is wide (thousands of columns or wider),  $\text{\LaTeX}$  may terminate prematurely because of exceeded  $\text{\TeX}$  capacity. Unfortunately, this problem remains opaque to the user, since `texi2dvi()` and `texi2pdf()` do not convey much details about  $\text{\LaTeX}$  problems when typesetting a document. We recommend the following if a user encounters problems with running `msaPrettyPrint()`'s output with `texi2dvi()` and `texi2pdf()`:



1. Run `pdflatex` on the generated `.tex` file to see whether it is actually a problem with  $\text{\TeX}$  capacity.
2. If so, split the alignment into multiple chunks and run `msaPrettyPrint()` on each chunk separately.

The following example demonstrates this approach for a multiple alignment object ‘aln’:

```
chunkSize <- 300 ## how much fits on one page depends on the length of
                  ## names and the number of sequences;
                  ## change to what suits your needs

for (start in seq(1, ncol(aln), by=chunkSize))
{
  end <- min(start + chunkSize - 1, ncol(aln))
  alnPart <- DNAMultipleAlignment(subseq(unmasked(aln), start, end))

  msaPrettyPrint(x=alnPart, output="pdf", subset=NULL,
                 file=paste0("aln_", start, "-", end, ".pdf"))
}
```

This creates multiple PDF files all of which show one part of the alignment. Please note, however, that the numbering of columns is restarted for each chunk.

## 7.8 Further Caveats

- Note that `texi2dvi()` and `texi2pdf()` always save the resulting DVI/PDF files to the current working directory, even if the  $\text{\LaTeX}$  source file is in a different directory. That is also the reason why the temporary file is created in the current working directory in the example below.
- `\text{\TeXshade}` has a wide array of functionalities. Only the most common ones have been tested for interoperability with R. So the use of the arguments `furtherCode` and `code` is the user’s own risk!

## 8 Known Issues

### Memory Leaks

The original implementations of ClustalW, ClustalOmega, and MUSCLE are stand-alone command line programs which are only run once each time a multiple sequence alignment is performed. During the development of the `msa` package, we performed memory management checks using Valgrind [?] and discovered multiple memory leaks in ClustalW and MUSCLE. These memory leaks have no effect for the command line tools, since the program is closed each time the alignment is finished. In the implementation of the `msa` package, however, these memory leaks may have an effect if the same algorithm is run multiple times.

For MUSCLE, we managed to eliminate all memory leaks by deactivating the two parameters `weight1` and `weight2`. ClustalOmega did not show any memory leaks. ClustalW indeed has several memory leaks which are benign if the algorithm is run only a few times, but which may have more severe effects if the algorithm is run many times. ClustalOmega also has a minor memory leak, but the loss of data is so small that no major problems are to be expected except for thousands of executions of ClustalOmega.

### **ClustalOmega vs. Older GCC Versions on Linux/Unix**

We have encountered peculiar behavior of ClustalOmega if the package was built using an older GCC version: if we built the package on an `x86_64` Linux system with GCC 4.4.7, ClustalOmega built smoothly and could be executed without any errors. However, the resulting multiple sequence alignment was more than sub-optimal. We could neither determine the source of this problem nor which GCC versions show this behavior. We therefore recommend Linux/Unix users to use an up-to-date GCC version (we used 4.8.2 during package development, which worked nicely) or, in case they encounter dubious results, to update to a newer GCC version and re-install the package.

### **ClustalOmega: OpenMP Support on Mac OS**

ClustalOmega is implemented to make use of OpenMP (if available on the target platform). Due to issues on one of the Bioconductor build servers running Mac OS, we had to deactivate OpenMP generally for Mac OS platforms. If a Mac OS user wants to re-activate OpenMP, he/she should download the source package tarball, untar it, comment/uncomment the corresponding line in `msa/src/ClustalOmega/msaMakefile` (see first six lines), and build/install the package from source.

## **9 Future Extensions**

We envision the following changes/extensions in future versions of the package:

- Integration of more multiple sequence alignment algorithms, such as, T-Coffee [?] or DIALIGN [?]
- Support for retrieving guide trees from the multiple sequence alignment algorithms
- Interface to methods computing phylogenetic trees (e.g. as contained in the original implementation of ClustalW)
- Elimination of memory leaks described in Section ?? and re-activation of parameters that have been deactivated in order to avoid memory leaks
- More tolerant handling of custom substitution matrices (MUSCLE interface)

## 10 How to Cite This Package

If you use this package for research that is published later, you are kindly asked to cite it as follows:

U. Bodenhofer, E. Bonatesta, C. Horejš-Kainrath, and S. Hochreiter (2015). msa: an R package for multiple sequence alignment. *Bioinformatics* **31**(24):3997–3999. DOI: [bioinformatics/btv494](https://doi.org/10.1093/bioinformatics/btv494).

To obtain a BibTeX entries of the reference, enter the following into your R session:

```
toBibtex(citation("msa"))
```

Moreover, we insist that, any time you cite the package, you also cite the original paper in which the original algorithm has been introduced (see bibliography below).

## 11 Change Log

### Version 1.4.5:

- fixes in ClustalOmega source code to ensure Windows compatibility of GCC6 compatibility fix

### Version 1.4.4:

- bug fix in `msaClustalW()`: unsupported parameter ‘tree’ deactivated
- fixes in ClustalOmega source code to ensure GCC 6 compatibility

### Version 1.4.3:

- minor changes in package vignette

**Versions 1.4.1 and 1.4.2:** version number bumps for technical reasons related to Bioconductor build servers

**Version 1.4.0:** release as part of Bioconductor 3.3

### Version 1.3.7:

- fixes in `msaPrettyPrint()` function

### Version 1.3.6:

- `msaPrettyPrint()` now also accepts dashes in file names
- added section about pretty-printing wide alignments to package vignette

### Version 1.3.5:

- adaptation of displaying help text by `msa()` function

**Version 1.3.4:**

- added function for checking and fixing sequence names for possibly problematic characters that could lead to  $\text{\LaTeX}$  errors when using `msaPrettyPrint()`
- corresponding changes in documentation
- minor namespace fix

**Version 1.3.3:**

- added function for converting multiple sequence alignments for use with other sequence alignment packages
- corresponding changes in documentation

**Version 1.3.2:**

- further fixes in Makefiles and Makevars files to account for changes in build system
- update of citation information

**Version 1.3.1:**

- fixes in Makefiles and Makevars files to account for changes in build system

**Version 1.3.0:** new branch for Bioconductor 3.3 devel

**Version 1.2.0:** release as part of Bioconductor 3.2

**Version 1.1.3:**

- bug fix related to custom substitution matrices in the MUSCLE interface
- corrections and updates of documentation

**Version 1.1.2:**

- new `print()` function for multiple alignments that also allows for displaying alignments in their entirety (plus additional customizations)
- strongly improved handling of custom substitution matrices by `msaClustalW()`: now custom matrices can also be supplied for nucleotide sequences which can also be passed via the `substitutionMatrix` argument. The `dnamatrix` argument is still available for the sake of backwards compatibility.
- strongly improved handling of custom substitution matrices by `msaMuscle()`
- fix of improperly aligned sequence logos produced by `msaPrettyPrint()`
- updated citation information

**Version 1.1.1:**

- fix of `msa()` function

**Version 1.1.0:** new branch for Bioconductor 3.2 devel

**Version 1.0.0:** first official release as part of Bioconductor 3.1