CLUSTAL-OMEGA is a general purpose multiple sequence alignment program
for protein and DNA/RNA.

INTRODUCTION

Clustal-Omega is a general purpose multiple sequence alignment (MSA)
program for protein and DNA/RNA. It produces high quality MSAs and is
capable of handling data-sets of hundreds of thousands of sequences in
reasonable time.

In default mode, users give a file of sequences to be aligned and
these are clustered to produce a guide tree and this is used to guide
a "progressive alignment" of the sequences. There are also facilities
for aligning existing alignments to each other, aligning a sequence to
an alignment and for using a hidden Markov model (HMM) to help guide
an alignment of new sequences that are homologous to the sequences
used to make the HMM. This latter procedure is referred to as
"external profile alignment" or EPA.

Clustal-Omega uses HMMs for the alignment engine, based on the HHalign
package from Johannes Soeding [1]. Guide trees are made using an
enhanced version of mBed [2] which can cluster very large numbers of
sequences in O(N*log(N)) time. Multiple alignment then proceeds by
aligning larger and larger alignments using HHalign, following the
clustering given by the guide tree.

In its current form Clustal-Omega has been extensivly tested for
protein sequences, DNA/RNA support has been added since version 1.1.0.

SEQUENCE INPUT:

-i, --in, --infile={<file>,-}
        Multiple sequence input file (- for stdin)

--hmm-in=<file>
        HMM input files

--dealign
        Dealign input sequences

--profile1, --p1=<file>
        Pre-aligned multiple sequence file (aligned columns will be kept fixed)

--profile2, --p2=<file>
        Pre-aligned multiple sequence file (aligned columns will be kept fixed)

--is-profile
        disable check if profile, force profile (default no)

-t, --seqtype={Protein, RNA, DNA}
        Force a sequence type (default: auto)

--infmt={a2m=fa[sta],clu[stal],msf,phy[lip],selex,st[ockholm],vie[nna]}
        Forced sequence input file format (default: auto)

For sequence and profile input Clustal-Omega uses the Squid library
from Sean Eddy [3].

Clustal-Omega accepts 3 types of sequence input: (i) a sequence file with un-aligned or aligned sequences, (ii) profiles (a multiple alignment in a file) of aligned sequences, (iii) a HMM. Valid combinations of the above are:

(a) one file with un-aligned or aligned sequences (i); the sequences will be aligned, and the alignment will be written out. For this mode use the -i flag. If the sequences are aligned (all sequences have the same length and at least one sequence has at least one gap), then the alignment is turned into a HMM, the sequences are de-aligned and the now un-aligned sequences are aligned using the HMM as an External Profile for External Profile Alignment (EPA). If no EPA is desired use the --dealign flag.

Use the above option to make a multiple alignment from a set of sequences. A sequence file must contain more than one sequence (at least two sequences).

(b) two profiles (ii)+(ii); the columns in each profile will be kept fixed and the alignment of the two profiles will be written out. Use the --p1 and --p2 flags for this mode.

Use this option to align two alignments (profiles) together.

(c) one file with un/aligned sequences (i) and one profile (ii); the profile is converted into a HMM and the un-aligned sequences will be multiply aligned (using the HMM background information) to form a profile; this constructed profile is aligned with the input profile; the columns in each profile (the original one and the one created from the un-aligned sequences) will be kept fixed and the alignment of the two profiles will be written out. Use the -i flag in conjunction with the --p1 flag for this mode.
The un/aligned sequences file (i) must contain at least two sequences. If a single sequence has to be aligned with a profile the profile-profile option (b) has to be used.

Use the above option to add new sequences to an existing alignment.

(d) one file with un-aligned sequences (i) and one HMM (iii); the un-aligned sequences will be aligned to form a profile, using the HMM as an External Profile. So far only one HMM can be input and only HMMer2 and HMMer3 formats are allowed. The alignment will be written out; the HMM information is discarded. As, at the moment, only one HMM can be used, no HMM is produced if the sequences are already aligned. Use the -i flag in conjunction with the --hmm-in flag for this mode. Multiple HMMs can be inputted, however, in the current version all but the first HMM will be ignored.

Use this option to make a new multiple alignment of sequences from the input file and use the HMM as a guide (EPA).

Sequences that all have the same lengths but do not contain a single gap are by default not recognised as a profile. If these sequences are indeed a profile an not just a collection of unaligned sequences that happen to have the same length then specify the --is-profile flag.

Invalid combinations of the above are:

(v) an un/aligned sequence file containing just one sequence (i)

```
  (w) an un/aligned sequence file containing just one sequence and a profile
      (i)+(ii)

  (x) an un/aligned sequence file containing just one sequence and a HMM
      (i)+(iii)

  (y) two or more HMMs (iii)+(iii)+... cannot be aligned to one another.

  (z) one profile (ii) cannot be aligned with a HMM (iii)


  The following MSA file formats are allowed:

      a2m=fasta, (vienna)
      clustal,
      msf,
      phylip,
      selex,
      stockholm


  Prior to MSA, Clustal-Omega de-aligns all sequence input (i). However,
  alignment information is automatically converted into a HMM and used
  during MSA, unless the --dealign flag is specifically set.  Profiles
  (ii) are not de-aligned.

  Since version 1.1.0 the Clustal-Omega alignment engine can process
  DNA/RNA. Clustal-Omega tries to guess the sequence type (protein,
  DNA/RNA), but this can be over-ruled with the --seqtype (-t) flag.


  CLUSTERING:

    --distmat-in=<file>
          Pairwise distance matrix input file (skips distance computation)

    --distmat-out=<file>
          Pairwise distance matrix output file

    --guidetree-in=<file>
          Guide tree input file
          (skips distance computation and guide tree clustering step)

    --guidetree-out=<file>
          Guide tree output file

    --full
          Use full distance matrix for guide-tree calculation (slow; mBed is default)

    --full-iter
          Use full distance matrix for guide-tree calculation during iteration (mBed is default)

    --cluster-size=<n>
          soft maximum of sequences in sub-clusters

    --clustering-out=<file>
          Clustering output file

    --use-kimura
          use Kimura distance correction for aligned sequences (default no)

    --percent-id
          convert distances into percent identities (default no)

  In order to produce a multiple alignment Clustal-Omega requires a
```

guide tree which defines the order in which sequences/profiles are aligned. A guide tree in turn is constructed, based on a distance matrix. Conventionally, this distance matrix is comprised of all the pair-wise distances of the sequences. The distance measure Clustal-Omega uses for pair-wise distances of un-aligned sequences is the k-tuple measure [4], which was also implemented in Clustal 1.83 and ClustalW2 [5,6]. If the protein sequences inputted via -i are aligned, then Clustal-Omega uses pairwise aligned identities, these distances can be Kimura-corrected [7] by specifying --use-kimura. The distances between aligned DNA/RNA sequences are determined from the alignment, no Kimura correction can be used. The computational effort (time/memory) to calculate and store a full distance matrix grows quadratically with the number of sequences. Clustal-Omega can improve this scalability to N*log(N) by employing a fast clustering algorithm called mBed [2]; this option is automatically invoked (default). If a full distance matrix evaluation is desired, then the --full flag has to be set. The mBed mode calculates a reduced set of pair-wise distances. These distances are used in a k-means algorithm, that clusters at most 100 sequences. For each cluster a full distance matrix is calculated. No full distance matrix (of all input sequences) is calculated in mBed mode. If there are less than 100 sequences in the input, then in effect a full distance matrix is calculated in mBed mode, however, no distance matrix can be outputted (see below).

Clustal-Omega uses Muscle's [8] fast UPGMA implementation to construct its guide trees from the distance matrix. By default, the distance matrix is used internally to construct the guide tree and is then discarded. By specifying --distmat-out the internal distance matrix can be written to file. This is only possible in --full or --full-iter mode. The guide trees by default are used internally to guide the multiple alignment and are then discarded. By specifying the --guidetree-out option these internal guide trees can be written out to file. Conversely, the distance calculation and/or guide tree building stage can be skipped, by reading in a pre-calculated distance matrix and/or pre-calculated guide tree. These options are invoked by specifying the --distmat-in and/or --guidetree-in flags, respectively. However, distance matrix reading is disabled in the current version. By default, distance matrix and guide tree files are not over-written, if a file with the specified name already exists. In this case Clustal-Omega aborts during the command-line processing stage. To force over-writing of already existing files use the --force flag (see MISCELLANEOUS). In mBed mode a full distance matrix cannot be outputted, distance matrix output is only possible in --full mode. mBed or --full distance mode do not affect the ability to write out guide-trees. It is possible to perform an initial mBed (not-full) distance calculation and a subsequent full distance calculation (see section ITERATION). In this case a distance matrix can be outputted.

Guide trees can be iterated to refine the alignment (see section ITERATION). Clustal-Omega takes the alignment, that was produced initially and constructs a new distance matrix from this alignment. The distance measure used at this stage is a full alignment distance (as opposed the initial pairwise k-tuple distance); distances of protein sequences can be Kimura corrected [7], DNA/RNA distances are not. By default, Clustal-Omega constructs a reduced distance matrix at this stage using the mBed algorithm, which will then be used to create an improved (iterated) new guide tree. To turn off mBed-like clustering at this stage the --full-iter flag has to be set. While full alignment distances in general are much faster to calculate than k-tuple distances, time and memory requirements still scale quadratically with the number of sequences and --full-iter clustering should only be considered for smaller cases (<< 10,000 sequences) or if response time and resources are not an issue.

The default cluster size in mBed mode is 100. This means that sequences are grouped into clusters with a soft maximum of 100 sequences, full distance matrices are calculated for these clusters, guide-trees are calculated for the clusters and the clusters are then strung together with an over-arching guide-tree. It is possible to change the cluster-size with the --cluster-size flag. The clustering can be outputted to file. The output is comprised of the cluster index, a running index for the sequences within each cluster, the running index for the sequence within the input file, the name of the sequence and the bi-section sequence (see EXAMPLES).

Clustal-Omega uses pair-distances. Between unaligned sequences these are so called k-tuple distance, between aligned sequences they are full alignment distances, as employed by Squid. These values range between 0.0 (identical) and 1.0 (completely different). The distances are used to construct the guide-tree and are by default outputted if --distmat-out is specified (and --full and/or --full-iter are set). For full alignment distances there is a so called Kimura correction [7] which more closely reflects evolutionary distance. Kimura-corrected distances range from 0.0 (identical) to theoretically infinity (completely different). In practice there appears to be a maximum value. In Clustal-Omega these Kimura-corrected distance can be outputted for protein if the --use-kimura flag is specified. Kimura correction is not available for DNA/RNA. Up to and including version 1.1.1 Kimura-corrected distances were outputted by default (where possible). Since version 1.2.0 the default is to output uncorrected distances.

Pair-distances closely correspond to percentage pair-wise identities through i=100*(1-d), where i is the percentage pair-wise identity and d is the pair-wise distance. Percentage pair-wise identities can be outputted in Clustal-Omega instead of the distance matrix by specifying the --percent-id flag as well as --distmat-out, --full and/or --full-iter. Percentage pair-wise identities cannot be outputted if --use-kimura is specified.

ALIGNMENT OUTPUT:

```
  -o, --out, --outfile={file,-}
        Multiple sequence alignment output file (default: stdout)

  --outfmt={a2m=fa[sta],clu[stal],msf,phy[lip],selex,st[ockholm],vie[nna]}
        MSA output file format (default: fasta)

  --residuenumber, --resno
        in Clustal format print residue numbers (default no)

  --wrap=<n>
        number of residues before line-wrap in output

  --output-order={input-order,tree-order}
        MSA output order like in input/guide-tree
```

By default Clustal-Omega writes its results (alignments) to stdout. An output file can be specified with the -o flag. Output to stdout is not possible in verbose mode (-v, see MISCELLANEOUS) as verbose/debugging messages would interfere with the alignment output. By default, alignment files are not over-written, if a file with the specified name already exists. In this case Clustal-Omega aborts during the command-line processing stage. To force over-writing of already existing files use the --force flag (see MISCELLANEOUS).

Clustal-Omega can output alignments in various formats by setting the --outfmt flag:

* for Fasta format set: --outfmt=a2m or --outfmt=fa or --outfmt=fasta

* for Clustal format set: --outfmt=clu or --outfmt=clustal

* for Msf format: set --outfmt= msf

* for Phylip format set: --outfmt=phy or --outfmt=phylip

* for Selex format set: --outfmt=selex

* for Stockholm format set: --outfmt=st or --outfmt=stockholm

* for Vienna format set: --outfmt=vie or --outfmt=vienna

In ClustalW one could print the residue number of the last residue in each line in Clustal-Format. This feature can be turned on by setting the --resno or --residuenumber flag.

The line lengths in Clustal Format is usually 60 residues, in Fasta format it is usually 60 or 80 residues. This value can be set using the --wrap flag.

By default the order of sequences in the output is the same as in the input (--output-order=input-order). This can be changed to the order in which the sequences appear in the guide-tree by setting --output-order=tree-order.

ITERATION:

--iterations, --iter=<n> Number of (combined guide tree/HMM) iterations

--max-guidetree-iterations=<n> Maximum guide tree iterations

--max-hmm-iterations=<n> Maximum number of HMM iterations

By default, Clustal-Omega calculates (or reads in) a guide tree and performs a multiple alignment in the order specified by this guide tree. This alignment is then outputted. Clustal-Omega can 'iterate' its guide tree. The hope is that the full alignment distances, that can be derived from the initial alignment, will give rise to a better guide tree, and by extension, to a better alignment.

A similar rationale applies to HMM-iteration. MSAs in general are very 'vulnerable' at their early stages. Sequences that are aligned at an early stage remain fixed for the rest of the MSA. Another way of putting this is: 'once a gap, always a gap'. This behaviour can be mitigated by HMM iteration. An initial alignment is created and turned into a HMM. This HMM can help in a new round of MSA to 'anticipate' where residues should align. This is using the HMM as an External Profile and carrying out iterative EPA. In practice, individual sequences and profiles are aligned to the External HMM, derived after the initial alignment. Pseudo-count information is then transferred to the (internal) HMM, corresponding to the individual sequence/profile. The now somewhat 'softened' sequences/profiles are then in turn aligned in the order specified by the guide tree. Pseudo-count transfer is reduced with the size of the profile. Individual sequences attain the greatest pseudo-count transfer, larger profiles less so. Pseudo-count transfer to profiles larger than, say, 10 is negligible. The effect of HMM iteration is more pronounced in larger test sets (that is, with more sequences).

```
Both, HMM- and guide tree-iteration come at a cost of increasing the
run-time. One round of guide tree iteration adds on (roughly) the time
it took to construct the initial alignment. If, for example, the
initial alignment took 1min, then it will take (roughly) 2min to
iterate the guide tree once, 3min to iterate the guide tree twice, and
so on. HMM-iteration is more costly, as each round of iteration adds
three times the time required for the alignment stage. For example, if
the initial alignment took 1min, then each additional round of HMM
iteration will add on 3min; so 4 iterations will take 13min
(=1min+4*3min). The factor of 3 stems from the fact that at every
stage both intermediate profiles have to be aligned with the
background HMM, and finally the (softened) HMMs have to be aligned as
well. All times are quoted for single processors.

By default, guide tree iteration and HMM-iteration are coupled. This
means, at each iteration step both, guide tree and HMM, are
re-calculated. This is invoked by setting the --iter flag. For
example, if --iter=1, then first an initial alignment is produced
(without external HMM background information and using k-tuple
distances to calculate the guide tree). This initial alignment is then
used to re-calculate a new guide tree (using full alignment distances)
and to create a HMM. The new guide tree and the HMM are then used to
produce a new MSA.

Iteration of guide tree and HMM can be de-coupled. This means that the
number of guide tree iterations and HMM iterations can be
different. This can be done by combining the --iter flag with the
--max-guidetree-iterations and/or the --max-hmm-iterations flag.  The
number of guide tree iterations is the minimum of --iter and
--max-guidetree-iterations, while the number of HMM iterations is the
minimum of --iter and --max-hmm-iterations.  If, for example, HMM
iteration should be performed 5 times but guide tree iteration should
be performed only 3 times, then one should set --iter=5 and
--max-guidetree-iterations=3. All three flags can be specified at the
same time (however, this makes no sense). It is not sufficient just to
specify --max-guidetree-iterations and --max-hmm-iterations but not
--iter. If any iteration is desired, then --iter has to be
set. Conversely, if no alignment is desired but only distance
calculation and tree construction, then --max-hmm-iterations=-1 will
terminate the calculation before the alignment stage; --iter does not
have to be specified in this case.


LIMITS (will exit early, if exceeded):

  --maxnumseq=<n>           Maximum allowed number of sequences

  --maxseqlen=<l>           Maximum allowed sequence length

Limits can be imposed on the number of sequences in the input file
and/or the lengths of the sequences. This cap can be set with the
--maxnumseq and --maxseqlen flags, respectively. Clustal-Omega will
exit early, if these limits are exceeded.


MISCELLANEOUS:

  --auto                    Set options automatically (might overwrite some of your options)

  --threads=<n>             Number of processors to use

  -l, --log=<file>          Log all non-essential output to this file

  -h, --help                Print help and exit
```

```
  -v, --verbose             Verbose output (increases if given multiple times)

  --version                 Print version information and exit

  --long-version            Print long version information and exit

  --force                   Force file overwriting
```

Users may feel unsure which options are appropriate in certain situations even though using ClustalO without any special options should give you the desired results. The --auto flag tries to alleviate this problem and selects accuracy/speed flags according to the number of sequences. For all cases will use mBed and thereby possibly overwrite the --full option. For more than 1,000 sequences the iteration is turned off as the effect of iteration is more noticeable for 'larger' problems. Otherwise iterations are set to 1 if not already set to a higher value by the user. Expert users may want to avoid this flag and exercise more fine tuned control by selecting the appropriate options manually.

Certain parts of the MSA calculation have been parallelised. Most noticeably, the distance matrix calculation, and certain aspects of the HMM building stage. Clustal-Omega uses OpenMP. By default, Clustal-Omega will attempt to use as many threads as possible. For example, on a 4-core machine Clustal-Omega will attempt to use 4 threads. The number of threads can be limited by setting the --threads flag. This may be desirable, for example, in the case of benchmarking/timing.

Usually, non-essential (verbose) output is written to screen. This output can be written to file by specifying the --log flag.

Help is available by specifying the -h flag.

By default Clustal-Omega does not print any information to stdout (other than the final alignment, if no output file is specified). Information concerning the progress of the alignment can be obtained by specifying one verbosity flag (-v). This may be desirable, to verify what Clustal-Omega is actually doing at the moment. If two verbosity flags (-v -v) are specified, command-line flags (explicitly and implicitly set) are printed in addition to the progress report. Triple verbose level (-v -v -v) is the most verbose level. In addition to single- and double-verbose information much more information is displayed: input sequences and names, details of the tree construction and intermediate alignments. Tree construction information includes pairwise distances. The number of pairwise distances scales with the square of the number of sequences, and double verbose mode is probably only useful for a small number of sequences.

The current version number of Clustal-Omega can be displayed by setting the --version flag.

The current version number of Clustal-Omega as well as the code-name and the build date can be displayed by setting the --long-version flag.

By default, Clustal-Omega does not over-write files. These can be (i) alignment output, (ii) distance matrix and (iii) guide tree. Overwriting can be forced by setting the --force flag.

EXAMPLES:

```
./clustalo -i globin.fa

Clustal-Omega reads the sequence file globin.fa, aligns the sequences
and prints the result to screen in fasta/a2m format.


./clustalo -i globin.fa -o globin.sto --outfmt=st

If the file globin.sto does not exist, then Clustal-Omega reads the
sequence file globin.fa, aligns the sequences and prints the result to
globin.sto in Stockholm format. If the file globin.sto does exist
already, then Clustal-Omega terminates the alignment process before
reading globin.fa.


./clustalo -i globin.fa -o globin.aln --outfmt=clu --force

Clustal-Omega reads the sequence file globin.fa, aligns the sequences
and prints the result to globin.aln in Clustal format, overwriting the
file globin.aln, if it already exists.


./clustalo -i globin.fa --distmat-out=globin.mat --guidetree-out=globin.dnd --force

Clustal-Omega reads the sequence file globin.fa, aligns the sequences,
prints the result to screen in fasta/a2m format (default), the guide
tree to globin.dnd and the distance matrix to globin.mat, overwriting
those files if they already exist.


./clustalo -i globin.fa --guidetree-in=globin.dnd

Clustal-Omega reads the files globin.fa and globin.dnd, skipping
distance calculation and guide tree creation, using instead the guide
tree specified in globin.dnd.


./clustalo -i globin.fa --hmm-in=PF00042.hmm

Clustal-Omega reads the sequence file globin.fa and the HMM file
PF00042.hmm (in HMMer2 or HMMer3 format).  It then performs the
alignment, transferring pseudo-count information contained in
PF00042.hmm to the sequences/profiles during the MSA.


./clustalo -i globin.sto

Clustal-Omega reads the file globin.sto (of aligned sequences in
Stockholm format). It converts the alignment into a HMM, de-aligns the
sequences and re-aligns them, transferring pseudo-count information to
the sequences/profiles during the MSA. The guide tree is constructed
using a full distance matrix.


./clustalo -i globin.sto  --dealign

Clustal-Omega reads the file globin.sto (of aligned sequences in
Stockholm format). It de-aligns the sequences and then re-aligns
them. No HMM is produced in the process, no pseudo-count information
is transferred. Consequently, the output must be the same as for
unaligned output (like in the first example ./clustalo -i globin.fa)


./clustalo -i globin.fa --iter=2
```

Clustal-Omega reads the file globin.fa, creates a UPGMA guide tree built from k-tuple distances, and performs an initial alignment. This initial alignment is converted into a HMM and a new guide tree is built from the (preliminary) full alignment distances of the initial alignment. The un-aligned sequences are then aligned (for the second time but this time) using pseudo-count information from the HMM created after the initial alignment (and using the new guide tree). This second alignment is then again converted into a HMM and a new guide tree is constructed. The un-aligned sequences are then aligned (for a third time), again using pseudo-count information of the HMM from the previous step and the most recent guide tree. The final alignment is written to screen.

```
./clustalo -i globin.fa --iter=5 --max-guidetree-iterations=1
```

Clustal-Omega reads the file globin.fa, creates a UPGMA guide tree built from k-tuple distances, and performs an initial alignment. This initial alignment is converted into a HMM and a new guide tree is built from the (preliminary) full alignment distances of the initial alignment. The un-aligned sequences are then aligned (for the second time but this time) using pseudo-count information from the HMM created after the initial alignment (and using the new guide tree). For the last 4 iterations the guide tree is left unchanged and only HMM iteration is performed. This means that intermediate alignments are converted to HMMs, and these intermediate HMMs are used to guide the MSA during subsequent iteration stages.

```
./clustalo -i globin.fa -o globin.a2m -v
```

In case the file globin.a2m does not exist, Clustal-Omega reads the file globin.fa, prints a progress report to screen and writes the alignment in (default) Fasta format to globin.a2m. The progress report consists of the number of threads used, the number of sequences read, the current progress in the k-tuple distance calculation, completion of the guide tree computation and current progress of the MSA stage. If the file globin.a2m already exists Clustal-Omega aborts before reading the file globin.fa. Note that in verbose mode an output file has to be specified, because progress/debugging information, which is printed to screen, would interfere with the alignment being printed to screen.

```
./clustalo -i PF00042_full.fa --dealign --full --outfmt=vie -o PF00042_full.vie --force
```

Clustal-Omega reads the file PF00042_full.fa. This file contains several thousand aligned sequences. --dealign tells Clustal-Omega to erase all alignment information and re-align the sequences from scratch. As there are several thousand sequences calculating a full distance matrix may be slow. Setting the --full flag specifically selects the full distance mode over the default mBed mode. The alignment is then written out in Vienna format (fasta format all on one line, no line breaks per sequence) to file PF00042_full.vie.

```
./clustalo -i PF00042_full.fa --dealign --outfmt=vie -o PF00042_full.vie --force
```

Clustal-Omega reads the file PF00042_full.fa. This file contains several thousand aligned sequences. --dealign tells Clustal-Omega to erase all alignment information and re-align the sequences from scratch. Calculating the distance matrix will be done by mBed (default). Clustal-Omega will calculate pairwise distances to a small number of reference sequences only. This will give a significant

speed-up. The speed-up is greater for larger families (more sequences). The alignment is then written out in Vienna format (fasta format all on one line, no line breaks per sequence) to file PF00042_full.vie.

```
./clustalo --p1=globin.sto --p2=PF00042_full.vie -o globin+pf00042.fa
```

Clustal-Omega reads files globin.sto and PF00042_full.vie of aligned sequences (profiles). Both profiles are then aligned. The relative positions of residues in both profiles are not changed during this alignment, however, columns of gaps may be inserted into the profiles, respectively. The final alignment is written to file globin+pf00042.fa in fasta format.

```
./clustalo -i globin.fa --p1=PF00042_full.vie -o pf00042+globin.fa
```

Clustal-Omega reads file globin.fa of un-aligned sequences and the profile (of aligned sequences) in file PF00042_full.vie. A HMM is created from the profile. This HMM is used to guide the alignment of the un-aligned sequences in globin.fa. The profile that was generated during this alignment of un-aligned globin.fa sequences is then aligned to the input profile PF00042_full.vie. The relative positions of residues in profile PF00042_full.vie is not changed during this alignment, however, columns of gaps may be inserted into the profile. The final alignment is output to file pf00042+globin.fa in fasta format. The alignment in this example may be slightly different from the alignment in the previous example, because no HMM guidance was used generate the profile globin.sto. In this example HMM guidance was used to align the sequences in globin.fa; the hope being that this intermediate alignment will have profited from the bigger profile.

```
./clustalo -i globin.fa --clustering-out=globin.aux  --cluster-size=3
```

globin.fa contains 7 sequences. Usually a full distance matrix is created for less than 100 sequences. This is over-written by specifying --cluster-size=3. Clustal-Omega attempts to create clusters of no more than 3 sequences. This clustering is recorded in the file globin.aux which looks like

```
        Cluster 0: object 0 has index 0 (=seq P1|HBB_HUMAN )     00
        Cluster 0: object 1 has index 1 (=seq P1|HBB_HORSE )     00
        Cluster 1: object 0 has index 4 (=seq P1|MYG_PHYCA )     1
        Cluster 1: object 1 has index 5 (=seq P1|GLB5_PETMA )    1
        Cluster 1: object 2 has index 6 (=seq P1|LGB2_LUPLU )    1
        Cluster 2: object 0 has index 2 (=seq P1|HBA_HUMAN )     01
        Cluster 2: object 1 has index 3 (=seq P1|HBA_HORSE )     01
```

There are 3 clusters, named Cluster~0, Cluster~1 and Cluster~2. Cluster~0 has 2 sequences, which are sequence 0 and 1 from the input file, named P1|HBB_HUMAN and P1|HBB_HORSE. Cluster~1 has 3 sequences, sequences 4,5,6 from the input file and Cluster~2 has 2 sequences, sequences 2 and 3 from the input file. The binary string at the end of each line encode the bi-section that led to this clustering. The first digit indicated the initial split. The '0' indicates that in the first split sequences 0,1,2,3 were grouped together and the '1' that sequences 4,5,6 were grouped together. The size of Cluster~1 does not exceed --cluster-size, so it does need to be broken up. The Cluster (with the initial '0') containing sequences 0,1,2,3 is comprised of 4 sequences; this number exceed --cluster-size, so that it will have to be broken up. This second split is indicated by the second digit of the binary string. The second '0' indicates that sequences 0,1 fall into one Cluster (which

will ultimately be Cluster~0), and the second '1' indicates that sequences 2,3 fall into another cluster (ultimately Cluster~2).

LITERATURE:

[1] Johannes Soding (2005) Protein homology detection by HMM-HMM comparison. Bioinformatics 21 (7): 951â€“960.

[2] Blackshields G, Sievers F, Shi W, Wilm A, Higgins DG. Sequence embedding for fast construction of guide trees for multiple sequence alignment. Algorithms Mol Biol. 2010 May 14;5:21.

[3] http://www.genetics.wustl.edu/eddy/software/#squid

[4] Wilbur and Lipman, 1983; PMID 6572363

[5] Thompson JD, Higgins DG, Gibson TJ. (1994). CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. Nucleic Acids Res., 22, 4673-4680.

[6] Larkin MA, Blackshields G, Brown NP, Chenna R, McGettigan PA, McWilliam H, Valentin F, Wallace IM, Wilm A, Lopez R, Thompson JD, Gibson TJ, Higgins DG. (2007). Clustal W and Clustal X version 2.0. Bioinformatics, 23, 2947-2948.

[7] Kimura M (1980). "A simple method for estimating evolutionary rates of base substitutions through comparative studies of nucleotide sequences". Journal of Molecular Evolution 16: 111â€“120.

[8] Edgar, R.C. (2004) MUSCLE: multiple sequence alignment with high accuracy and high throughput.Nucleic Acids Res. 32(5):1792-1797.