Sequence Formats

Contents

Sequences

Before reading the rest of this document, please note:
Microsoft WORD format is not a sequence format.

Sequences can be read and written in a variety of formats. These can be very confusing for users, but EMBOSS aims to make life easier by automatically recognising the sequence format on input.

That means that if you are converting from using another sequencing package to EMBOSS and you have your existing sequences in a format that is specific for that package, for example GCG format, you will have no problem reading them in.

If you don't hold your sequence in a recognised standard format, you will not be able to analyse your sequence easily.

What a sequence format is NOT

When we talk about 'sequence format' we are NOT talking about any sort of program-specific format like a word processor format or text formatting language , so we are not talking about things like: 'NOTEPAD', 'WORD', 'WORDPAD', 'PostScript', 'PDF', 'RTF', 'TeX', 'HTML'

If you have somehow managed to type a sequence into a word-processor (!) you should:

- Save the sequence to a file as ASCII text (try selecting: File, SaveAs, Text)
- Stop using word-processors to write sequences.
- Investigate a sequence editor, such as **mse**
- Investigate using simple text editors, such as **pico**, **nedit** or, at a pinch, **wordpad**

Now, repeat after me:
Microsoft WORD format is not a sequence format

EMBOSS programs will not read in anything which is held in Microsoft WORD files.

What a sequence format IS

Sequence formats are ASCII TEXT.

They are the required arrangement of characters, symbols and keywords that specify what things such as the sequence, ID name, comments, etc. look like in the sequence entry and where in the entry the program should look to find them.

There are generally no hidden, unprintable 'control' characters in any sequence format (there are none in those that EMBOSS supports). All standard sequence formats can be printed out or viewed simply by displaying their file.

Why so many formats?

There are at least a couple of dozen sequence formats in existence at the moment. Some are much more common than others.

Formats were designed so as to be able to hold the sequence data and other information about the sequence.

Nearly every sequence analysis package written since programs were first used to read and write sequences has invented its own format. **Except for EMBOSS.**

Nearly every collection of sequences that dares call itself a database has stored its data in its own format.

Identification

A sequence does not require any sort of identification, but it certainly helps!

Most sequence formats include at least one form of ID name, usually placed somewhere at the top of the sequence format.

The simple format **fasta** has the ID name as the first word on its title line. For example the ID name 'xyz':

```
>xyz some other comment
ttcctctttctcgactccatcttcgcggtagctgggaccgccgttcagtcgccaatatgc
agctctttgtccgcgcccaggagctacacaccttcgaggtgaccggccaggaaacggtcg
cccagatcaaggctcatgtagcctcactggagggcatt
```

IDs and Accessions

An entry in a database must have some way of being uniquely identified in that database. Most sequence databases have two such identifiers for each sequence - an ID name and an Accession number.

Why are there two such identifiers? The ID name was originally intended to be a human-readable name that had some indication of the function of its sequence. In EMBL and GenBank the first two (or three) letters indicated the species and the rest indicated the function, for example 'hsfau' is the 'Homo Sapiens FAU pseudogene'. This naming scheme started to be a problem when the number of entries added each day was so vast that people could not make up the ID names fast enough. Instead, the Accession numbers were used as the ID name. Therefore you will now find ID names like 'AF061303', the same as the Accession number for that sequence in EMBL.

ID names are not guaranteed to remain the same between different versions of a database (although in practice they usually do).

Accession numbers are unique alphanumeric identifiers that are guaranteed to remain with that sequence through the rest of the life of the database. If two sequences are merged into one, then the new sequence will get a new Accession number and the Accession numbers of the merged sequences will be retained as 'secondary' Accession numbers.

EMBL, GenBank and SwissProt share an Accession numbering scheme - an Accession number uniquely identifies a sequence within these three databases.

Annotation and Features

Most formats allow you to hold other description, annotation and comments, for example **fasta** format holds comments in the title line:

```
>xyz some other comment
ttcctctttctcgactccatcttcgcggtagctgggaccgccgttcagtcgccaatatgc
agctctttgtccgcgcccaggagctacacaccttcgaggtgaccggccaggaaacggtcg
cccagatcaaggctcatgtagcctcactggagggcatt
```

Other formats have specific fields for holding information such as references, keywords, associated entries in other databases and feature tables

---

The Sequence

Nucleotide (DNA or RNA) sequences are usually stored in the IUBMB standard codes.

Similarly, protein sequences are usually stored in the IUPAC standard one-letter codes.

For example, **fasta** format holds the sequence as anything after the '>' line until the next entry starts:

```
>xyz some other comment
ttcctctttctcgactccatcttcgcggtagctgggaccgccgttcagtcgccaatatgc
agctctttgtccgcgcccaggagctacacaccttcgaggtgaccggccaggaaacggtcg
cccagatcaaggctcatgtagcctcactggagggcatt
```

There are exceptions to this code, for example, **staden** format uses non-standard ambiguity codes.

Sequence Database Formats

Some of the most widespread sequence formats apart from **fasta** are those used by the major sequence databases.

- EMBL
- GenBank
- SwissProt
- PIR

Sequence Files

Files can hold sequences in standard recognised formats.

Files can also hold sequences in non-standard unrecognisable ways. Do not expect EMBOSS to be able to read your sequences held in a word-processor format file. EMBOSS is not a word-processor!

Multiple sequences

Some sequence formats can hold multiple sequences in one file. The details of how many sequences are held in one file differs between formats, but they either allow many sequences to be concatenated one after the other, or they hold the sequences together in some sort of aligned set of sequences.

Other formats, such as **gcg**, **plain** and **staden** formats can only hold one sequence per file. An attempt to concatenate several sequences in one file leaves the results as a mess that makes it impossible to decide where the sequences start and end or what is annotation and what is sequence.

These **single** formats therefore cause problems when there are multiple sequences to write out because a single file containing multiple sequences in that format is invalid. When these formats are specified for output, an EMBOSS program will allow you to write many sequences to one file, but EMBOSS programs will not be able to reliably read in the resulting mess.

It you really wish to write multiple sequences out in formats that can not cope with multiple sequences, you are advised to add the global qualifier **-ossingle** on the command line. This will force the EMBOSS program to ignore the given output file name and will generate its own file names. One sequence will be written to each such file. These file names are made from the sequence ID name, with the name of the format as the extension (e.g. **hsfau.gcg**).

This is not ideal. Preferably, you should stay away from formats that can't cope with multiple sequences in a file.

Input Sequence Formats

To date, the following sequence formats are accepted as input.

By default, (i.e. if no format is explicitly specified) EMBOSS tries each format in turn until one succeeds.

Input Format	Auto	Nuc	Pro	Feat	Gap	Multi	Description
gcg gcg8	Yes	Yes	Yes	No	Yes	No	GCG 9.x and 10.x format with the format and sequence type identified on the first line of the file. GCG 8.x format where anything up to the first line containing ".." is considered as heading, and the remainder is sequence data.
embl em	Yes	Yes	No	Yes	Yes	No	EMBL entry format, including all the fields in the latest release format. The Staden package and others use EMBL or similar formats for sequence data.
swiss sw swissprot	Yes	No	Yes	Yes	Yes	No	SWISSPROT entry format, including all the fields in the latest release format.
nbrf pir	Yes	Yes	Yes	Yes	Yes	No	NBRF (PIR) format, as used in the PIR database sequence files. This format was used for some years as an interchange format with the reference data followed by the sequence data. This unofficial PIR format is what EMBOSS supports. If there is enough interest, we can also use NBRF database format with separate files for sequence (the main EMBOSS input/output) and for features. Documentation of this format is hard to find, but we do have a copy from PIR. The sequence files include the ID and description but no citation or feature information.
pdb	Yes	No	Yes	No	No	No	PDB protein databank format ATOM lines
pdbseq	Yes	No	Yes	No	No	No	PDB protein databank format SEQRES lines
pdbnuc	No	Yes	No	No	No	No	PDB protein databank format nucleotide ATOM lines
pdbnucseq	No	Yes	No	No	No	No	PDB protein databank format nucleotide SEQRES lines
fasta ncbi	Yes	Yes	Yes	No	Yes	No	FASTA format with optional accession number and database name in NCBI style included as part of the sequence identifier. eg **>database|accession|id description** or **>name description** or **>name accession description**
gifasta	No	Yes	Yes	No	Yes	No	FASTA format including NCBI-style GIs (alias)
pearson	Yes	Yes	Yes	No	Yes	No	FASTA format with no further processing of the "ID" eg:

							>name description Used where fasta or ncbi format interprets the ID in an unwanted way, this format skips the further ID parsing stage of reading these files.
fastq	Yes	Yes	No	No	No	No	FASTQ short read format ignoring quality scores
fastq-sanger	No	Yes	No	No	No	No	FASTQ short read format with phred quality
fastq-illumina	No	Yes	No	No	No	No	FASTQ Illumina 1.3 short read format
fastq-solexa	No	Yes	No	No	No	No	FASTQ Solexa/Illumina 1.0 short read format
genbank gb ddbj	Yes	Yes	No	Yes	Yes	No	GENBANK entry format, including the feature table..
refseqp	No	No	Yes	Yes	Yes	No	Refseq protein entry format
genpept	No	No	Yes	Yes	Yes	No	Refseq protein entry format (alias)
codata	Yes	Yes	Yes	Yes	Yes	No	Codata entry format
strider	Yes	Yes	Yes	No	Yes	No	DNA strider output format
clustal aln	Yes	Yes	Yes	No	Yes	No	ClustalW ALN (multiple alignment) format.
phylip	Yes	Yes	Yes	No	Yes	Yes	Phylip interleaved and non-interleaved formats
phylipnon	No	Yes	Yes	No	Yes	Yes	Phylip non-interleaved format
	Yes	Yes	Yes	No	Yes	No	ACEDB sequence format
dbid	No	Yes	Yes	No	Yes	No	FASTA format variant with Database name first, then ID name then an optional accession number eg: **>database name description** or **>database name accession description**
msf	Yes	Yes	Yes	No	Yes	No	Wisconsin Package GCG MSF (mutiple sequence file) file format
hennig86	Yes	Yes	Yes	No	Yes	No	Hennig86 output format
jackknifer	Yes	Yes	Yes	No	Yes	No	Jackknifer interleaved and non-interleaved formats
nexus paup	Yes	Yes	Yes	No	Yes	No	Nexus/paup interleaved format
treecon	Yes	Yes	Yes	No	Yes	No	Treecon output format
mega	Yes	Yes	Yes	No	Yes	No	Mega interleaved and non-interleaved formats
igstrict	Yes	Yes	Yes	No	Yes	No	Intelligenetics sequence format strict parser
ig	No	Yes	Yes	No	Yes	No	Intelligenetics sequence format
staden	No	Yes	Yes	No	Yes	No	This format is actually obsolete, the latest version of the Staden package does not support it anymore (see "experiment" format for the new Staden package format). Staden format was a just the sequence in simple text with, optionally, comments at any position in the sequence. When EMBOSS reads in "staden" format, it recognizes a comment at the top of the sequence as the sequence dientifier and removes any comments inside

							the sequence. Some alternative nucleotide ambiguity codes are used and should be converted.
text plain	No	Yes	Yes	No	Yes	No	Plain text. This is the format with no format. The whole of the file is read in as a sequence. No attempt is made to parse the file contents in any way. Anything is acceptable in this format. This means that any character will be included in the sequence, even digits and punctuation. Use this format only when you are sure that the input sequence file is correct and contains only what you want to be considered as your 'sequence'.
gff2	Yes	Yes	Yes	Yes	Yes	No	GFF feature file with sequence in the header Normally used as a pure feature format, but can hold the sequence as part of the structured header.
gff3 gff	Yes	Yes	Yes	Yes	Yes	No	GFF3 feature file with sequence
stockholm pfam	Yes	Yes	Yes	No	Yes	No	Stockholm (pfam) format
selex	No	Yes	Yes	No	Yes	No	SELEX format is used by Sean Eddy's HMMER package. It can store RNA secondary structure as part of the sequence annotation.
fitch	Yes	Yes	Yes	No	Yes	No	Fitch program format
mase	No	Yes	Yes	No	Yes	No	Mase program format
raw	Yes	Yes	Yes	No	No	No	Like text/plain format except that it removes any whitespace or digits, accepts only alphabetic characters and rejects anything else. This means that it is safer to use this format than **plain** format. If you have digits and spaces or TAB characters, these are removed and ignored. If you have other non-alphabetic characters (for example, punctuation characters), then the sequence will be rejected as erroneous. Gap characters, '-', and translated STOP codon characters '*' are legal.
experiment	Yes	Yes	Yes	No	Yes	No	The Staden package stores single sequencing experiment reads in a format derived from EMBL. All EMBL tags are allowed, plus many extras. Unusually, the extra tags are allowed to continue beyond the '//' line which only marks the end of the sequence. The "EX" experiment line is used to create a sequence description. Accuracy values are stored, or at least the largest value for each sequence position. To date no EMBOSS program is using these values.
abi	Yes	Yes	Yes	No	Yes	No	ABI trace file format. This is the format of file produced by ABI sequencing machines. It contains the 'trace data' i.e. the probabilities of the 4 bases along the sequencing run, together with the sequence, as deduced from that data. The sequence information is what is normally read in and used by EMBOSS programs, although the trace data is available and may be utilised by some specialised EMBOSS programs.

							The code for this is heavily based on David Mathog's fortran library with a description of ABI trace file format (abi.txt): ftp://saf.bio.caltech.edu/pub/software/molbio/abitools.zip

Special Format	Description
asis	This is not so much a sequence format as a quick way of entering a sequence on the command line, but it is included here for completeness. Where a filename would normally be given, in **asis** format there is the sequence itself. An example would be: asis::atacgcagttatctgaccat In 'asis' format the name is the sequence so no file needs to be opened. This is a special case. This syntax can be very useful for generating command lines.

Output Sequence Formats

To date, the following sequence formats are available as output.

Some sequence formats can hold multiple sequences in one file, these are marked as **multiple** in the following table.

Other formats, such as GCG, plain and staden formats can only hold one sequence per file, these are marked as **single**.

Output Format	Single	Save	Nuc	Pro	Feat	Gap	Multi	Description
gcg gcg8	No	No	Yes	Yes	No	Yes	No	Wisconsin Package GCG 9.x and 10.x format with the sequence type on the first line of the file. GCG 8.x format where anything up to the first line containing ".." is considered as heading, and the remainder is sequence data.
embl em emblnew	No	No	Yes	No	Yes	Yes	No	EMBL entry format with available fields filled in and others with no infomation omitted. The EMBOSS command line allows missing data such as accession numbers to be provided if they are not obtainable from the input sequence.
swiss sw swissprot swissnew swnew swissprotnew	No	No	No	Yes	Yes	Yes	No	Swissprot entry format with available fields filled in and others with no infomation omitted. The EMBOSS command line allows missing data such as accession numbers to be provided if they are not obtainable from the input sequence.
fasta pearson	No	No	Yes	Yes	No	Yes	No	Standard Pearson FASTA format, but with the accession number included after the identifier if available.
ncbi	No	No	Yes	Yes	No	Yes	No	NCBI style FASTA format with the database name, entry name and accession number separated by pipe ("|") characters.
gifasta	No	No	Yes	Yes	No	Yes	No	NCBI fasta format with NCBI-style IDs

								using GI number
nbrf pir	No	No	Yes	Yes	Yes	Yes	No	NBRF/PIR entry format, as used in the PIR database sequence files.
genbank gb ddbj refseq	No	No	Yes	No	No	Yes	No	GENBANK entry format with available fields filled in and others with no infomation omitted. The EMBOSS command line allows missing data such as accession numbers to be provided if they are not obtainable from the input sequence.
gff2	No	No	Yes	Yes	Yes	Yes	No	GFF format. Normally used as a pure feature format, but can hold the sequence as part of the structured header.
gff3 gff	No	No	Yes	Yes	Yes	Yes	No	GFF3 feature file with sequence in FASTA format after
ig	No	No	Yes	Yes	No	Yes	No	Intelligenetics sequence format, as used by the Intelligenetics package
codata	No	No	Yes	Yes	No	Yes	No	Codata entry format
strider	No	No	Yes	Yes	No	Yes	No	DNA strider output format
acedb	No	No	Yes	Yes	No	Yes	No	ACEDB sequence format
experiment	No	No	Yes	Yes	No	Yes	No	Staden experiment file
staden	No	No	Yes	Yes	No	Yes	No	Old staden package sequence format. This format is actually obsolete, the latest version of the Staden package does not support it anymore. Staden format is a just the sequence in simple text with, optionally, comments at any position in the sequence. When EMBOSS reads in "staden" format, it recognizes only a comment at the top of the sequence but considers comments inside the sequence as part of the sequence. Some alternative nucleotide ambiguity codes are used and must be converted.
text plain raw	No	No	Yes	Yes	No	Yes	No	Plain sequence, no annotation or heading.
fitch	No	No	Yes	Yes	No	Yes	No	Fitch program format
msf	No	Yes	Yes	Yes	No	Yes	No	Wisconsin Package GCG MSF (mutiple sequence file) file format
clustal aln	No	Yes	Yes	Yes	No	Yes	No	Clustalw multiple alignment format
selex	No	Yes	Yes	Yes	No	Yes	No	Selex format
phylip	No	Yes	Yes	Yes	No	Yes	Yes	Phylip interleaved format
phylipnon phylip3	No	Yes	Yes	Yes	No	Yes	No	PHYLIP non-interleaved format that was used in Phylip version 3.2. Also called phylip3 for back compatibility with earlier EMBOSS versions.
asn1	No	No	Yes	Yes	No	Yes	No	A subset of NCBI ASN.1 containing entry name, accession number, description and

								sequence, similar to the current ASN.1 output of readseq
hennig86	No	Yes	Yes	Yes	No	Yes	No	Hennig86 output format
mega	No	Yes	Yes	Yes	No	Yes	No	Mega interleaved output format
meganon	No	Yes	Yes	Yes	No	Yes	No	Mega non-interleaved output format
nexus paup	No	Yes	Yes	Yes	No	Yes	No	Nexus/paup interleaved format
nexusnon paupnon	No	Yes	Yes	Yes	No	Yes	No	Nexus/paup non-interleaved format
jackknifer	No	Yes	Yes	Yes	No	Yes	No	Jackknifer output interleaved format
jackknifernon	No	Yes	Yes	Yes	No	Yes	No	Jackknifer output non-interleaved format
treecon	No	Yes	Yes	Yes	No	Yes	No	Treecon output format
mase	No	No	Yes	Yes	No	Yes	No	Mase program format
dasdna	No	No	Yes	No	No	Yes	No	DASDNA DAS nucleotide-only sequence
das	No	No	Yes	Yes	No	Yes	No	DASSEQUENCE DAS any sequence
fastq-sanger fastq	No	No	Yes	No	No	No	No	FASTQ short read format with phred quality
fastq-illumina	No	No	Yes	No	No	No	No	FASTQ Illumina 1.3 short read format
fastq-solexa	No	No	Yes	No	No	No	No	FASTQ Solexa/Illumina 1.0 short read format
debug	No	No	Yes	Yes	No	Yes	No	EMBOSS sequence object report for debugging showing all available fields. Not all fields will contain data - this depends very much on the input format used.

Creating a sequence

When typing a sequence in to a sequence editor, such as **mse**, the sequence editor should save the sequence to a file in a recognised format.

If you are creating a sequence by typing it into a text editor, then the best format is probably **fasta** format. Simply start the entry with a title line. This title line starts with a > character followed by the ID name of the sequence then any other comments. Subsequent lines contain the sequence. Many sequence entries can follow each other in a single file.

If you are truely masochistic, you will have typed your sequence into a word-processor. Don't do it again! If you click on the 'File' button and then on 'Save As..' you should be able to save your sequence as 'Text'. If you are lucky, you now have a sequence in 'plain' format.

Changing the format

To convert the sequences in the file 'myfile.seq' into the format 'embl' in the new file 'myfile2.seq', run either:

seqret myfile.seq embl::myfile2.seq
or

seqret myfile.seq myfile2.seq -osf embl
('-osf' is an abbreviation for '-osformat')

These two commands are exactly equivalent.

Input sequence command-line qualifiers

There are other command-line qualifiers that change the behaviour of the sequence input.

```
-sbegin       integer         first base used
-send         integer         last base used, default=seq length
-sreverse     boolean         reverse (if DNA)
-sask         boolean         ask for begin/end/reverse
-snucleotide  boolean         sequence is nucleotide
-sprotein     boolean         sequence is protein
-slower       boolean         make lower case
-supper       boolean         make upper case
-sformat      string          input sequence format
-sopenfile    string          input filename
-sdbname      string          database name
-sid          string          entryname
-ufo          string          UFO features
-fformat      string          features format
-fopenfile    string          features file name
```

Output sequence command-line qualifiers

There are other command-line qualifiers that change the behaviour of the sequence output.

```
-osformat           string     output sequence file format
-osextension        string     file name extension
-osname             string     base file name
-osdirectory        bool       output sequence file directory
-osdbname           string     database name to add
-ossingle           bool       create a separate output file for each entry
-oufo               string     feature file to create
-offormat           string     features format
-ofname             string     features file name
-ofdirectory        string     features output directory
```

Future directions

More formats, both for input and for output, can be easily added, so suggestions are always welcome.
