The ncOrtho v1.0 Manual

by Mirko Brüggemann,
Goethe University Institute for Applied Bioinformatics
March 2017
Edited by Andreas Blaumeiser in November 2018.

Structure of this manual

1. About ncOrtho

NcOrtho is a tool to predict orthologous microRNAs (miRNAs) for a provided seed miRNA precursor sequence. For such a seed sequence an initial core set of orthologous miRNAs is constructed based on conserved gene order. This core set serves as input for the prediction of novel orthologous miRNAs within a genome of interest, using covariance models. Thus, the ncOrtho tool is excellent at predicting miRNA orthologs in different species spanning also larger phylogenetic distances. The ncOrtho tool consists of two algorithms, the construction of the core set and the model-based prediction.

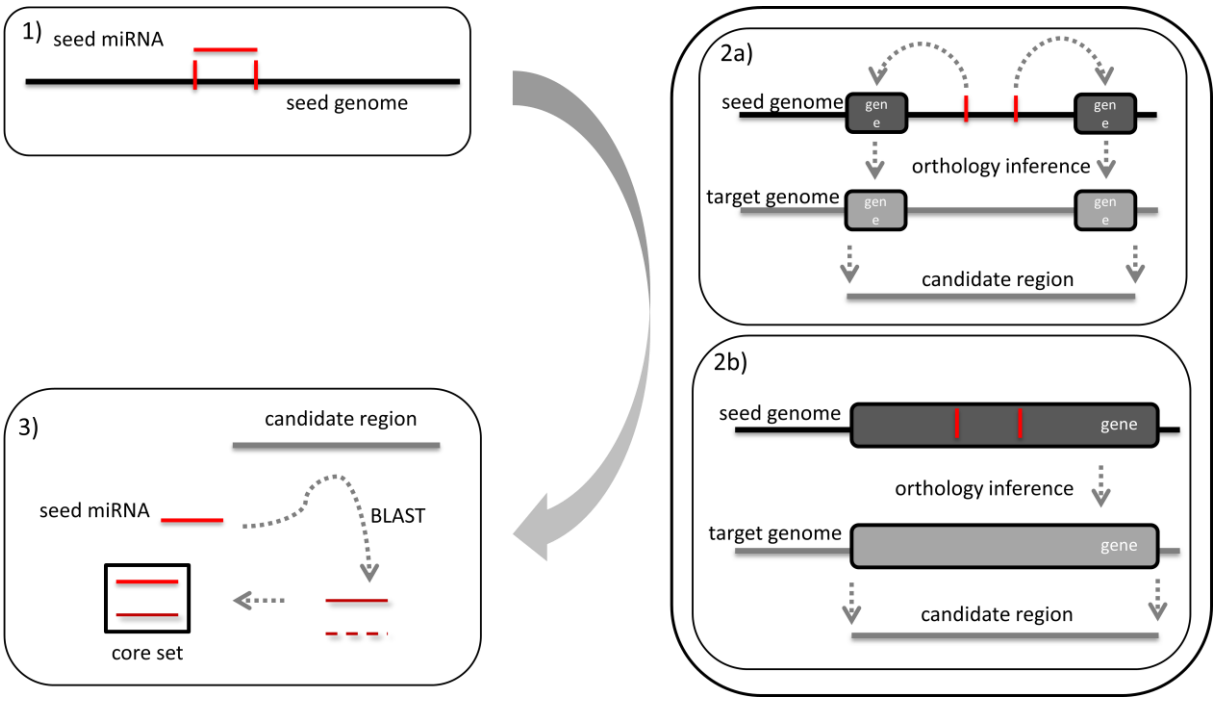


Figure 1: Workflow image of the ncOrtho routine for the core set construction. First, the position of the seed miRNA within the seed genome is located. This position can either lead to an intron or an intergenic region. In the intergenic case, orthologs in the target genome are identified for the protein-coding genes flanking the

query miRNA. In the intronic case, simply the ortholog in the target genome, to the respective protein-coding gene in the seed genome is identified. In both cases a candidate region of maximum length is extracted from the target genome. This region is used for a final BLAST search, queried by the initial seed miRNA.

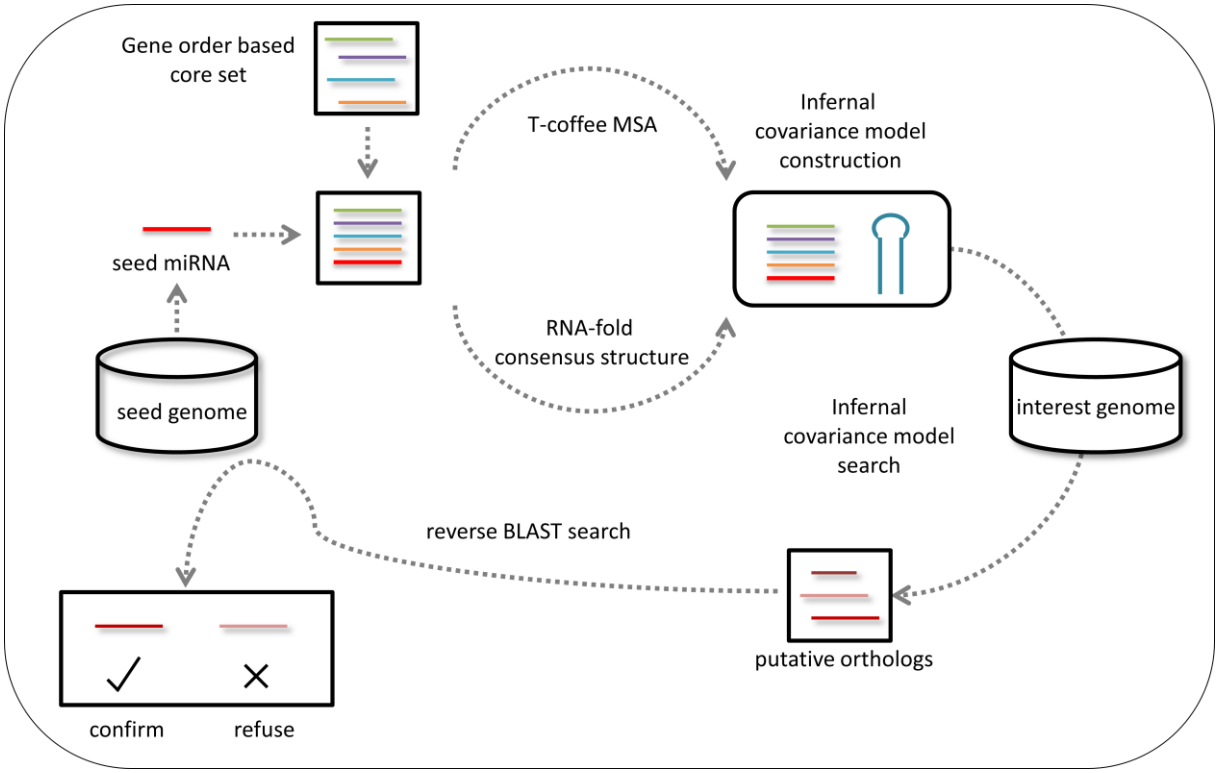


Figure 2: Workflow image of the ncOrtho routine to predict orthologs based on a core set model. Core set sequences are aligned and together with the consensus secondary structure used to construct the covariant model. The model is used to search the target genome and resulting putative orthologs are confirmed or refused based on an overlapping criterion accessed via a reverse BLAST search.

2. Downloading ncOrtho

The latest version of ncOrtho is available on GitHub:

https://github.com/acblaumeiser/ncOrtho

3. Installing ncOrtho

After downloading from GitHub, you end up with a zip archive file called:

```
ncOrtho-master.zip
```

Unzip the archive by typing:

```
unzip ncOrtho-master.zip
```

This should result in a list of four files:

```
LICENSE
```

```
ncOrtho-1.0.0_main.pl
ncOrtho-1.0.0_pre.pl
README.md
```

The tool requires you to have the following external tools installed:

BLAST+ (Camacho et al. 2009)
Infernal (Nawrocki and Eddy 2013)
T-Coffee (Notredame et al. 2000)
Vienna RNA (Lorenz et al. 2011)

In case the tools are not directly accessible via your PATH variable or you intend to use a specific version of a tool, make sure to enter the full path to the respective tool at the beginning of the ncOrtho-1.0.0_main.pl file.

To use the Infernal package, you are required to set the path variables for its three main scripts; cmbuild, cmcalibrate and cmsearch.

```
$cmbuild = /path/to/cmbuild
$cmcalibrate = /path/to/cmcalibrate
$cmsearch = /path/to/cmsearch
```

In order to have BLAST running, you need to specify the path to the blastn and makeblastdb routines.

```
$blastn = /path/to/blastn
$formatdb = /path/to/makeblastdb
```

Lastly, you need to specify the path to T-COFFEE.

```
$tcoffee = /path/to/t_coffee
```

Next you must set up the path variable in the ncOrtho-1.0.0_pre.pl file. The only additional software needed for this script is makeblastdb, so only one path needs to be edited.

```
$formatdb = /path/to/makeblastdb
```

4. Input File Formats

Genome sequences and gene annotations

In order to start the precomputation of your input data, you first need to create a folder that contains your root species data. Your root genome needs to be present in FASTA format (attention, only single line FASTA format is supported for the moment). Also, you need to have related gene annotations in GTF format (NOTE: the similar GFF format will not work). Create your directory structure in the following way:

```
mkdir root
mkdir root/genome
mkdir root/gtf
```

You now can copy your genome and gene annotation files in the respective folders and should see something like:

```
root/genome/my_species.fa
root/gtf/my_species.gtf
```

For the core species the same directory structure needs to be applied. Again, the genome files and gene annotation files need to be in FASTA and GTF file format, respectively.

```
core/genome/core_species01.fa
core/genome/core_species02.fa
core/genome/core_species03.fa
…
core/gtf/core_species01.gtf
core/gtf/core_species02.gtf
core/gtf/core_species03.gtf
…
```

For your species of interest, no genome annotations are needed, so simply create a directory and place your genome sequence in FASTA format into the folder.

```
interest/my_interest_species.fa
```

MicroRNA information

Your miRNA sequences from the root species that you want to use as a seed for the orthology search need to be placed in a separate folder, again in FASTA format.

```
miRNAs/mir-1.fa
miRNAs/mir-2.fa
miRNAs/mir-3.fa
…
```

Protein-coding gene information

In order to construct the initial core set from your list of core species, a complete species-wide orthology prediction from your root to each of the core species is needed. Only OMA orthologs are supported now. They can be downloaded directly from the OMA browser (omabrowser.org). Click on "Explore" and then select "Orthology between two genomes" in the drop-down menu. Enter the name of your root species in the first and the name of your first core species in the second field. Select "Ensembl Gene IDs" as preferred IDs and download the generated tab separated file. Move the file to your oma orthologs directory and assign it the name of the core species in a way that you abbreviate the genus by a single capital letter followed by a dot and the species name.

```
oma/C.species01
oma/C.species02
oma/C.species03
…
```

5. Precomputing your input files

Options

Once you have set up your directory structure and input data as described in the above section, you are good to go and run the ncOrtho-1.0.0_pre.pl script. It will construct the BLAST database for your

root genome and it will also hash all genome and gene annotation information, in order to speed up the main algorithm. Since nothing special is going on here, only the path (attention, the FULL PATH) to each of the input data directories is needed as input parameter.

```
-root_genome /path/to/root/genome/my_species.fa
-root_gtf /path/to/root/gtf/my_species.gtf
-core_genome_folder /path/to/core/genome/
-core_gtf_folder /path/to/core/gtf/
-oma_ortho_folder /path/to/oma/
```

Output

Once the precomputing script has done its job you will find that a .hash file has appeared in the core species genome and gtf folders, as well as in the oma folder and the root species gtf folder. In the root species genome folder, you should see the indexed files from the BLAST library.

```
root/genome/my_species.fa.nhr
root/genome/my_species.fa.nin
root/genome/my_species.fa.nsq
root/gtf/my_spcecies.gtf.hash
core/genome/core_genome.hash
core/gtf/core_gtf.hash
oma/oma_ortho.hash
```

6. Performing the orthology search

Options

After setting up your folder and precomputing your input files, you are ready for the actual search which is performed by the ncOrtho-1.0.0_main.pl script. To start the search, you have to specify again where the previously computed .hash files and respective genomes are located. Please again provide the FULL PATH to the algorithm.

```
-root_genome /path/to/root/genome/my_species.fa
-root_gtf_hash_file /path/to/root/gtf/my_species.gtf.hash
-core_genome_hash_file /path/to/core/genome/core_genome.hash
-core_gtf_hash_file /path/to/core/gtf/core_gtf.hash
-oma_hash_file /path/to/oma/oma_ortho.hash
-nc_rna /path/to/miRNA/mir-1.fa
-interest_genome /path/to/interest/my_interest_species.fa
```

Additionally, you have to specify a path to the output directory. All results and intermediate result files will be saved in this folder.

```
-outpath /path/to/mir-1_output
```

Besides the mandatory paths one must set above, a couple of additional options are available. First, if one already knows the exact position of the seed miRNA in the root species then one can input this information. If the position is not given by the user, then a simple BLAST search will be used to identify the position.

```
-rna_start start_position_of_miRNA
```

```
-rna_stop stop_position_of_miRNA
-rna_chr chromosome_of_miRNA
```

An important aspect when constructing the shared syntenic regions from your core species, is the number of allowed protein insertions within that region. As a default, no insertions are allowed within a shared syntenic region. However, you might want to relax this criterion by allowing a certain number of insertion events within a shared syntenic region. For this purpose, you can adjust the mip parameter.

```
-mip max_number_of_protein_insertions
```

Additionally, for the core set construction, one can specify the percentage of the seed miRNA sequence that must be found via BLAST within the candidate region. This percentage serves as a threshold to decide whether an ortholog to the seed miRNA is indeed present in a core species. The default value is 0.9, which means that the length of the candidate hit obtained from the BLAST search must be at least 90% of that of the seed miRNA sequence. To alter this criterion, you want to adjust the msl parameter:

```
-msl minimum_sequence_length_threshold
```

Output

Once you provided all input options to the ncOrtho-1.0.0_main.pl you are good to run the orthology prediction. First, the core set is computed according to your core species set and computation specifications. Second, the covariance model is computed, and your query genome is searched for orthologs with this model.

The core set computation will produce several intermediate files. For each core species the extracted candidate region is stored as a *.interseq* flat file. Also, you will find this file transferred into a BLAST library, giving you the additional *.nhr .nin .nsq* files, as well as a *.blastout* file for the output of the BLAST search. Your files might look like this:

```
C.species01.interseq
C.species01.interseq.nsq
C.species01.interseq.nin
C.species01.interseq.nhr
C.species01.interseq.blastout
```

All orthologous core miRNA sequences will be added to the core set, which is stored as a multi-FASTA file:

```
core_orthos.fa
```

From this file a multiple sequence alignment is computed with T-Coffee and stored in clustalw format. The consensus secondary structure is calculated with rnafold from the Vienna RNA package and together with the multiple sequence alignment of the core set stored in a STOCKHOLM format file. All files prior to the covariance modelling step should look like the following:

```
C.species01.rfold
seq.aln
rna_aln.sto
```

Additional intermediate files of the alignment and secondary structure calculation step are:

```
core_orthos.clustalw_aln
core_orthos_rfold1.template_list
core_orthos.dnd
alirna.ps
```

Next, the STOCKHOLM format file, which contains the multiple sequence alignment as well as the secondary structure information, serves as input for the covariance modelling step. After the model is built, it needs to be calibrated, which is the most time intensive step of the whole procedure and results in a calibrated covariance model:

```
rna.cm
```

The model is now used to search the genome sequence of your species of interest and putative orthologs are stored as an ordered list of hits:

```
cmsearch.out
```

From this list all candidates with an E-value greater or equal than 0.01 are chosen for the last reverse matching step. Here, the algorithm checks if a putative ortholog overlaps with the position of the initial seed miRNA. To this end, the sequence of each putative ortholog is extracted from the respective genome and stored in a FASTA file. Next, such a sequence serves as input for a BLAST search to determine the overlap with the initial seed.

```
ukn_rna1.fa
ukn_rna2.fa
ukn_rna3.fa
ukn_rna4.fa
reciproc_blast1.out
reciproc_blast2.out
reciproc_blast3.out
reciproc_blast4.out
```

If a candidate survives this final overlap criterion, the algorithm calls it as a confirmed ortholog and the exact position (start, stop, chr, strand) of the miRNA ortholog in your interest genome together with the covariance search score is stored in a results file. If this file contains multiple hits not overlapping each other, you have identified additional co-orthologs.

```
results.out
```

7. A simple ncOrtho analysis

A real-life example on how the tool works and which outputs are produced will be given below. Please note that all relevant files of this example, as well as directory structure and intermediate results are given in the example archive you downloaded. In this example we are interested in predicting an ortholog to the human miRNA hsa-let-7a-1 in mouse. This means our root species is fixed as human and our seed miRNA will be hsa-let-7a-1 as we are interested in finding its ortholog. The interest species is obviously mouse.
Next, we must decide which core species should be used. In this case, we will use the three primate species Gorilla, Pongo and Macaca, as they are close relatives to human. For each of the core species we downloaded the respective OMA orthologs to human from the OMA Browser. Please note that in order to safe space we do not use the full genomic sequence of each species, but only the chromosomes relevant for this example. The initial setup in the data folder should look like this:

```
./data
./data/root
./data/root/genome
./data/root/genome/Homo_sapiens.GRCh38.dna.toplevel.fa.reduced_chr9
./data/root/gtf
./data/root/gtf/Homo_sapiens.GRCh38.76.gtf
./data/core
./data/core/genome
./data/core/genome/Gorilla_gorilla.gorGor3.1.dna.toplevel.fa_chr9
./data/core/genome/Pongo_abelii.PPYG2.dna.toplevel.fa_chr9
./data/core/genome/Macaca_mulatta.MMUL_1.dna.toplevel.fa_chr15
./data/core/gtf
./data/core/gtf/Gorilla_gorilla.gorGor3.1.76.gtf
./data/core/gtf/Pongo_abelii.PPYG2.76.gtf
./data/core/gtf/Macaca_mulatta.MMUL_1.76.gtf
./data/interest
./data/interest/Mus_musculus.GRCm38.dna.toplevel.fa_chr13
./data/miRNAs
./data/miRNAs/hsa-let-7a-1.fa
./data/oma
./data/oma/G.gorilla
./data/oma/M.mulatta
./data/oma/P.abelii
```

With this setup ready, we are good to call the ncOrtho-1.0.0_pre.pl routine to prepare our input files:

```
/home/homer/ncOrtho/ncOrtho-1.0.0_pre.pl -root_genome
/home/homer/ncOrtho/example/data/root/genome/Homo_sapiens.GRCh38.dna
.toplevel.fa.reduced_chr9 -root_gtf
/home/homer/ncOrtho/example/data/root/gtf/Homo_sapiens.GRCh38.76.gtf
-core_genome_folder /home/homer/ncOrtho/example/data/core/genome/ -
core_gtf_folder /home/homer/ncOrtho/example/data/core/gtf/ -
oma_ortho_folder /home/homer/ncOrtho/example/data/oma/
```

If everything worked well, you should see the respective .hash files and BLAST libraries appearing in your data directories:

```
./root/genome/Homo_sapiens.GRCh38.dna.toplevel.fa.reduced_chr9.nhr
./root/genome/Homo_sapiens.GRCh38.dna.toplevel.fa.reduced_chr9.nsq
./root/genome/Homo_sapiens.GRCh38.dna.toplevel.fa.reduced_chr9.nin
./root/gtf/Homo_sapiens.GRCh38.76.gtf.hash
./root/genome/formatdb.log
./core/genome/core_genome.hash
./core/gtf/core_gtf.hash
./oma/oma_ortho.hash
```

Next, the ncOrtho-1.0.0_main.pl routine can be called in the following way. In addition to the mandatory parameters, we also set the –mip flag in this example. With the call of –mip 2, we set the number of possibly inserted protein coding genes within the shared syntenic region to 2. This allows for a more relaxed computation of the shared syntenic region.

```
/home/homer/ncOrtho/ncOrtho-1.0.0_main.pl -root_genome
/home/homer/ncOrtho/example/data/root/genome/Homo_sapiens.GRCh38.dna
.toplevel.fa.reduced_chr9 -root_gtf_hash_file
/home/homer/ncOrtho/example/data/root/gtf/Homo_sapiens.GRCh38.76.gtf
.hash -core_genome_hash_file
```

```
/home/homer/ncOrtho/example/data/core/genome/core_genome.hash -
core_gtf_hash_file
/home/homer/ncOrtho/example/data/core/gtf/core_gtf.hash -
oma_hash_file /home/homer/ncOrtho/example/data/oma/oma_ortho.hash -
nc_rna /home/homer/ncOrtho/example/data/miRNAs/hsa-let-7a-1.fa -
interest_genome
/home/homer/ncOrtho/example/data/interest/Mus_musculus.GRCm38.dna.to
plevel.fa_chr13 -out /home/homer/ncOrtho/example/search/hsa-let-7a-
1/
```

Depending on your system, a run may take between two and five minutes. The terminal from which you started the run will be used to display status information about the current run. Once the computation is finished you will find all result files stored in the specified output directory.

```
./ncRNA_rootGenome.blast.out
./M.mulatta.interseq
./formatdb.log
./M.mulatta.interseq.nhr
./M.mulatta.interseq.nsq
./M.mulatta.interseq.nin
./M.mulatta.interseq.blastout
./P.abelii.interseq
./P.abelii.interseq.nhr
./P.abelii.interseq.nsq
./P.abelii.interseq.nin
./P.abelii.interseq.blastout
./G.gorilla.interseq
./G.gorilla.interseq.nhr
./G.gorilla.interseq.nsq
./G.gorilla.interseq.nin
./G.gorilla.interseq.blastout
./core_orthos.fa
./seq.aln
./M.mulatta.rfold
./P.abelii.rfold
./query.rfold
./G.gorilla.rfold
./core_orthos_rfold1.template_l
ist
./core_orthos.dnd
./core_orthos.clustalw_aln
./rna_aln.sto
./alirna.ps
./rna.cm
./cmsearch.out
./ukn_rna1.fa
./reciproc_blast1.out
./ukn_rna2.fa
./reciproc_blast2.out
./ukn_rna3.fa
./reciproc_blast3.out
./results.out
```

In our case, the results.out file tells us that an ortholog to the human miRNA hsa-let-7a-1 has been found in mouse and that it is positioned on chromosome 13 stretching from position 48538186 to 48538265 on the – strand. Additionally, the bit score of the covariance model search is shown (108.3).

8. How to create a meaningful core set

A meaningful core set

Creating a meaningful core set can be challenging and is probably the most error-prone step in identifying orthologs with the ncOrtho approach. It is important to remember that the entire covariance modelling and final search step relies on sequences identified via shared synteny from this core set. One should always choose core species that are taxonomically closer related to the seed species than to the interest species. Additionally, all core species should be on a different taxonomical level when compared to the seed species. This will ensure that more orthologous sequences can be identified via shared synteny, which in turn will increase the sequence diversity in the covariance model.

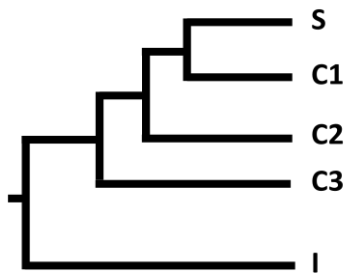


Figure 3: Intended use of the ncOrtho prediction tool, showing the three core species c1, c2 and c3 in increasing taxonomical order.

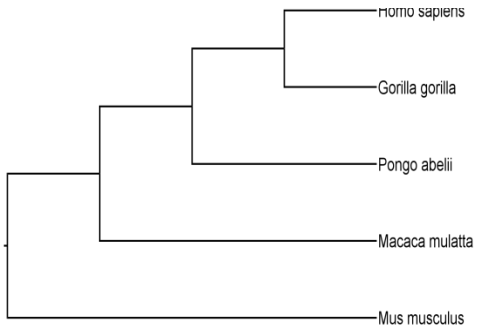
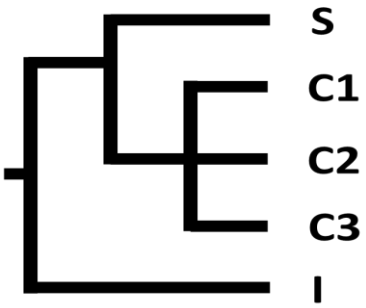


Figure 4: Species tree as it is used in the above example of a simple ncOrtho analysis. Core species are choosen with increasing taxonomical distance to the seed species, while being all more closely related to the see species, than to the search species.

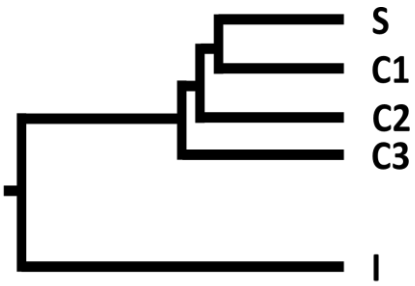
When deciding on the set of core species one wants to use for his orthology prediction analysis, a couple of mistakes can be made. In this section we try to account for the most common of them.

1. Core species are on the same taxonomical level



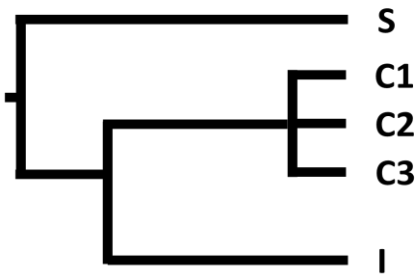
In this scenario all core species c1, c2 and c3 are on the same taxonomical level. They all exhibit the same taxonomical distance to the seed species s. This therefore means that core species c2 and c3 do not add additional information to the core set, but rather add redundant information which does not help to increase diversity in the core set.

2. Over specific core set



This scenario shows a core set that is too specific. The three core species differ in their taxonomical distance from the seed, but only slightly. The interest species however differs largely from the seed, which means the core species do not really help in bridging this gap. Consider an example, where human is the seed and *Latimeria* is the interest species, but all core species are of the Hominidae.

3. Wrong position of core set species



In this scenario the core species are chosen in a way that they are taxonomically more closely related to the interest species than to the seed species. A layout like this contradicts the idea of identifying an orthologous counterpart in the interest species with a known query from the seed species. Additionally, all three core species are on the same taxonomical level (see 1.).

In addition to the positional pitfalls, as described above, one also must consider the number of core species used for a certain analysis. If too many core species are chosen, computation time may increase rapidly, due to the calibration of the covariance model. However, when choosing not enough core species, one may end up not predicting the desired ortholog, because the model contains not enough information. This is especially true if one tries to cover larger taxonomical distances.

References

Camacho, Christiam; Coulouris, George; Avagyan, Vahram; Ma, Ning; Papadopoulos, Jason; Bealer, Kevin; Madden, Thomas L. (2009): BLAST+: architecture and applications. In *BMC bioinformatics* 10, p. 421. DOI: 10.1186/1471-2105-10-421.

Lorenz, Ronny; Bernhart, Stephan H.; Höner Zu Siederdissen, Christian; Tafer, Hakim; Flamm, Christoph; Stadler, Peter F.; Hofacker, Ivo L. (2011): ViennaRNA Package 2.0. In *Algorithms for molecular biology: AMB* 6, p. 26. DOI: 10.1186/1748-7188-6-26.

Nawrocki, Eric P.; Eddy, Sean R. (2013): Infernal 1.1: 100-fold faster RNA homology searches. In *Bioinformatics* 29 (22), pp. 2933–2935. DOI: 10.1093/bioinformatics/btt509.

Notredame, C.; Higgins, D. G.; Heringa, J. (2000): T-Coffee: A novel method for fast and accurate multiple sequence alignment. In *Journal of molecular biology* 302 (1), pp. 205–217. DOI: 10.1006/jmbi.2000.4042.