**The ncOrtho v1.0 Manual**

by Mirko Brüggemann,

Goethe University Institute for Applied Bioinformatics

March, 2017

**Structure of this manual**

1. About ncOrtho
2. Downloading ncOrtho
3. Installing ncOrtho
4. Input File Formats
5. Precomputing your input files
6. Performing the ortholog search
7. A simple ncOrtho analysis
8. How to create a meaningful core set for your research question
9. **About ncOrtho**

NcOrtho is a tool to predict orthologous micro-RNAs (miRNAs) for a provided seed miRNA precursor sequence. For such a seed sequence an initial core set of known orthologous miRNAs is constructed based on gene order. This core set serves as input to predict novel orthologous miRNAs within a genome of interest, using covariance models. Thus the ncOrtho tool is excellent in 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.*

1. **Downloading ncOrtho**

Github repository?

Code version x.y.z

1. **Installing ncOrtho**

Once you have downloaded the ncOrtho source code, you will see a file called:

ncOrtho-1.0.0.tar.gz

uncompress it by typing:

gunzip ncOrtho-1.0.0.tar.gz

tar xf ncOrtho-1.0.0.tar

then change into the directory and list all source files:

cd ncOrtho-1.0.0/

ls

Now you should see a list of three files:

ncOrtho-1.0.0\_main.pl

ncOrtho-1.0.0\_pre.pl

README.txt

The tool requires you to have the Infernal RNA package (INFERence of RNA ALignment) already installed on your system. You also need a running version of the T-COFFEE multiple sequence aligner, as well as BLAST (Basic Local Alignment Search Tool).

Now open the ncOrtho-1.0.0\_main.pl script with a standard text editor. In the first lines, indicated as “pathvariables” you now have to enter the path or command for each of the required tools. Make sure to only change the contents of the path variables and not the variable name itself. Please also do not change anything besides the path variables.

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

$cmbuild = /path/to/cmbuild

$cmsearch = /path/to/cmsearch

$cmcalibrate = /path/to/cmcalibrate

In order to have BLAST running, you need to specify the path to the blastall and formatdb routines.

$blastn = /path/to/blastall

$formatdb = /path/to/formatdb

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

$tcoffee = /path/to/t\_coffee

Next you have to set up the path variables in the ncOrtho-1.0.0\_pre.pl script. Therefore open it as well in a text editor. The only additional software needed for this script is BLAST, so only one path needs to be edited.

$formatdb = /path/to/formatdb

1. **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 at the moment). Also you need to have related gene annotations in .GTF format (attention, 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. Also the genome files and gene annotation files need to be in .FASTA, .GTF file format again.

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 single .FASTA format into the folder.

interest/my\_interest\_species.fa

**Micro RNA information**

Your miRNA sequences from the root species that you want to use as a seed for the orthology search needed to be placed in a separated 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 wise orthology prediction from your root to each of the core species is needed. Therefore only OMA orthologs are supported at the moment. 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 assigne 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

1. **Precomputing your Input files**

**Options**

Once you have set up your directory structure and input data as it is described in the above section, you are good to go and run the ncOrtho-1.0.0\_pre.pl script. It will compute the BLAST databased for your root genome and it will also hash all genome and gen 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.

-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 it’s 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

1. **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 a previously created 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 has to 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 construction 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.

-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 90% of the seed miRNA sequence must match within the candidate region of the core species via the BLAST search. To alter this criterion you want to edit:

-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 you interest genome is searched for orthologs.

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:

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 and together with the multiple sequence alignment of the core set stored in a STOCKHOLM formatted file. All files prior to the covariance modelling step should look like:

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 formatted 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 build 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 order list of hits:

cmsearch.out

From this list all candidates with an e-value greater 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. Therefore the sequence of each putative orthologs 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 final results file. If this file contains multiple hits not overlapping each other you have identified additional co-orthologs.

results.out

1. **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 have to 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

./core/genome/core\_genome.hash

./core/gtf/core\_gtf.hash

./oma/oma\_ortho.hash

./formatdb.log

Next, the ncOrtho-1.0.0\_main.pl routine can be called in the following way. In addition to the mandatory 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 two. 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.toplevel.fa\_chr13 -out /home/homer/ncOrtho/example/search/hsa-let-7a-1/

According to your system a run may take 2-5 minutes. The terminal form which you started to 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\_list

./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

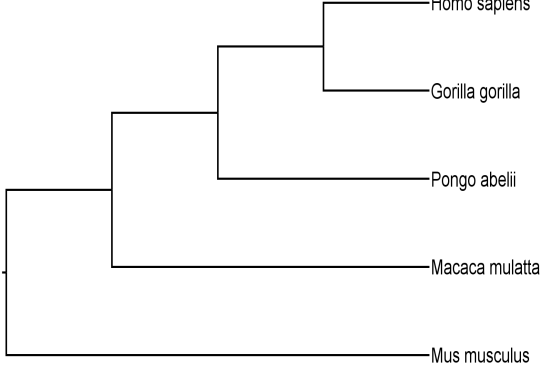
./results.out

In our case, the results.out files 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 score of the covariance search is showon (108.3).

1. **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 allow ensure that more orthologous sequence can be identified via shared synteny, which in turn will increase the sequence diversity in covariance model.



*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.*

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

**Potential pitfalls**

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 exactly the same taxonomical distance to the seed species s. This therefore means that core specie 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.

1. 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 don’t really help in bridging this gap. Consider an example, where human is the seed and the

Latimeria is the interest species, but all core species are of the hominidae.

1. 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 has to consider the number of core species used for a certain analysis. If too many core species are chosen, computational time may increase rapidly, due to the calibration of the covariance model. However when choosing not enough core species, one may end up not prediction the desired ortholog, because the model contains not enough information. This is especially true if one tries to cover larger taxonomical distances.