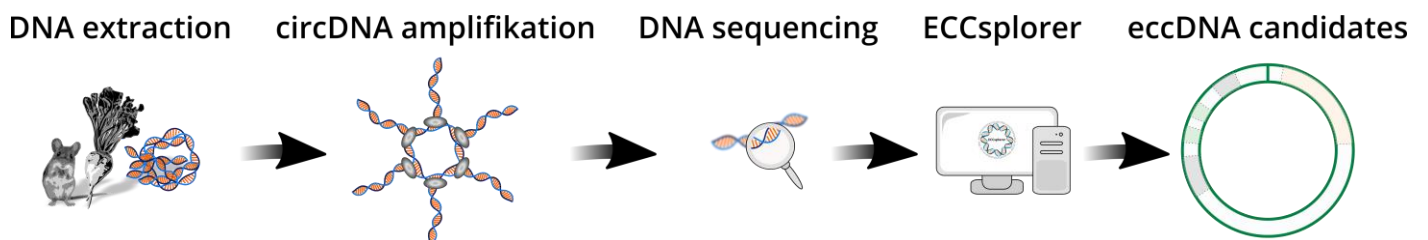
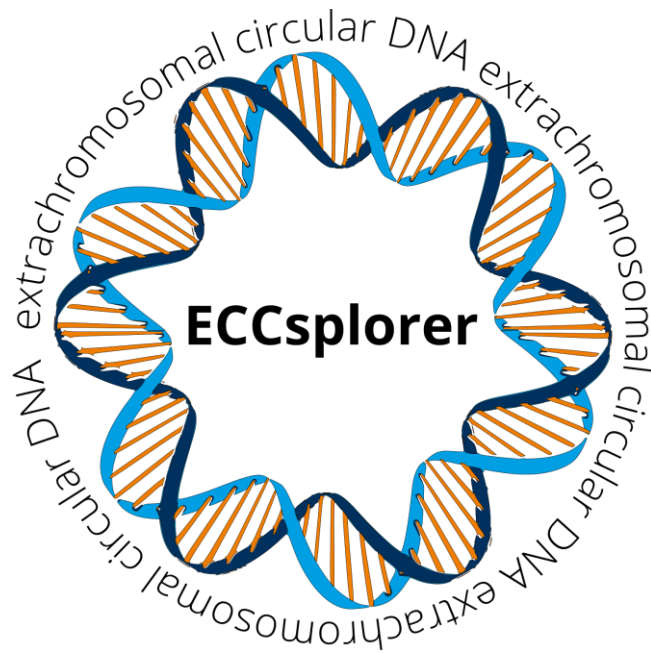


ECCsplorer Quickstart Manual



What is the ECCsplorer pipeline?

The ECCsplorer is a bioinformatics pipeline for the automated detection of extrachromosomal circular DNA from paired-end read data of amplified circular DNA.

Extrachromosomal circular DNAs (eccDNAs) are ring-like DNA structures physically separated from the chromosomes. They have been reported in a broad range of eukaryotes ranging from around 100 bp to several mega base pairs in size. Most of their functions still remain unknown, however some are associated with stress, cancer or aging whilst certain carry gene copies or harbor repetitive DNAs. Here, we present the concept for a bioinformatic pipeline to detect eccDNAs in any kind of organism or tissue using Next-generation sequencing (NGS). Amplified circular DNA serves as input for an easy and automated discovery of eccDNA candidates. The approach is based on two major procedures: Firstly, read mapping to the reference genome is followed by detection of exceptional read distributions including high coverage, discordant mapping and split reads. Secondly, we compare reference-free read clusters of the amplified eccDNA with control sample data determining specific enrichment. Both parts can be run separately or combined depending on the individual aim or data availability. Observed eccDNA candidates can be further used in a wide field of investigations – from tracking of active transposable elements to therapy approaches in terms of cancer-related eccDNAs.

Index

What is the ECCsplorer pipeline?	1
Index	2
Quick start guide	3
What do I need to use the ECCsplorer?	4
Minimum data requirements	4
Recommended data	4
Software requirements	4
Installation.....	4
What does the ECCsplorer do in detail?	6
Command line input files	7
Command line optional arguments	8
Output files.....	10
Output files per directory	10
Special output formats and image explanation	14
Interlaced and concatenated reads format:	14
Modified BED format:	15
Windowed coverage plot (Manhattan plot).....	15
Coverage plot	15
Abundance scatter plot	16
Coverage plot with annotation	16
config.py: pipeline configuration.....	16

Quick start guide

ECCsplorer is started over the command line interface. For needed software and installation see software requirements (page 4) and installation (page 4).

```
$python ECCsplorer.py <read_files> [optional arguments]
```

Short read paired-end sequencing data might be in FASTA or (compressed) FASTQ format in separated files. For usage specify either second/control read data or a reference (genome) sequence (for details see command line input files page 7).

```
$python ECCsplorer.py readsA1.fa/q readsA2.fa/q readsB1.fa/q readsB2.fa/q
```

or

```
$python ECCsplorer.py readsR1.fa/q readsR2.fa/q -ref sequence.fa
```

The first command will run the clustering module the second command will run the mapping module. Best results are achieved by inclusion of the comparative module by providing control read data and reference sequence (see Command line input files page 7), using the following command.

```
$python ECCsplorer.py readsA1.fa/q readsA2.fa/q readsB1.fa/q readsB2.fa/q -ref  
sequence.fa [optional arguments]
```

```
Use $python ECCsplorer.py -h/--help for an overview on all optional arguments.
```

Important hints:

- Make sure to use the trimming option `(-trm/--trim_reads)` or use trimmed reads.
- Use the count option `(-cnt/--read_count)` to ensure a maximal genome coverage of 0.1x in the clustering analysis or else you might experience very long module run times. If using a reference genome sequence you can set the count option to auto to automatically use a 0.1x genome coverage (based on the provided reference genome sequence or length, see Command line optional arguments).
- If you are expecting repetitive DNA (such as satellites or transposons) as eccDNA fraction, make sure to use a control data set (B) or else false positive rate might be higher.

What do I need to use the ECCsplorer?

In the following a short overview on data and software requirements is provided. For detailed information on input data see page 7. For detailed information on installation and 3rd party tools see page 4.

Minimum data requirements

1. Paired-end read data (FASTA/Q) from amplified (phi29 polymerase) circular DNA from the organism of interest and a reference (genome) sequence (FASTA).
- or
2. Paired-end read data (FASTA/Q) from amplified (phi29 polymerase) circular DNA from the organism of interest and paired-end read data (FASTA/Q) of a control (either non-amplified, or amplified from different treatment or organism).

Recommended data

3. Paired-end read data (FASTA/Q) from amplified (phi29 polymerase) circular DNA from the organism of interest, paired-end read data (FASTA/Q) of a control (either non-amplified, or amplified from different treatment or organism) and a reference (genome) sequence (FASTA).

Software requirements

ECCsplorer requires 64-bit Linux. A Java Runtime Environment is required for running the trimming option. Check with your distribution provider for details on installing Java.

ECCsplorer is implemented in Python 3 and R, using several additional open-source components developed and copyright by 3rd parties.

Installation

Once you have downloaded the pipeline and installed python packages, R libraries and 3rd party tools, the ECCsplorer.py script can be run in place. Optionally, the pipeline directory can be added to the `$PATH` environment variable.

Python (3.4 or higher) is required with the following additional packages:

- numpy
- biopython
- scipy
- pyRserve

Install the additional python packages using the pip command:

```
$cd /.../ECCsplorer # pipeline directory
$pip install -r requirements.txt
```

R (3.4.3 or higher) is required with the following additional libraries:

- ggplot2
- ggrepel
- grid & gridExtra
- dplyr

Install the additional libraries using the following command:

```
$R
>install.packages(c("ggplot2", "ggrepel", "grid", "gridExtra", "dplyr"))
```

Install 3rd party tools following their installation instructions:

- [NCBI blast+](#) (v2.2.28 or higher)
- [Trimmomatic](#) (optional)
 - needed for using `-trim/--trim_reads` option and using compressed input files
- [segemehl](#) (including haarz)
- [samtools](#) (1.9 or higher)
- [bedtools](#) (v2.28.0 or higher)
- [RepeatExplorer2](#)

After installing 3rd party tools you might need to add them to the `$PATH` environment variable or specify the location of their executables in the `lib/config.py` file (for details see page 16). The ECCsplorer pipeline will check if all 3rd party tools are available before starting its modules.

What does the ECCsplorer do in detail?

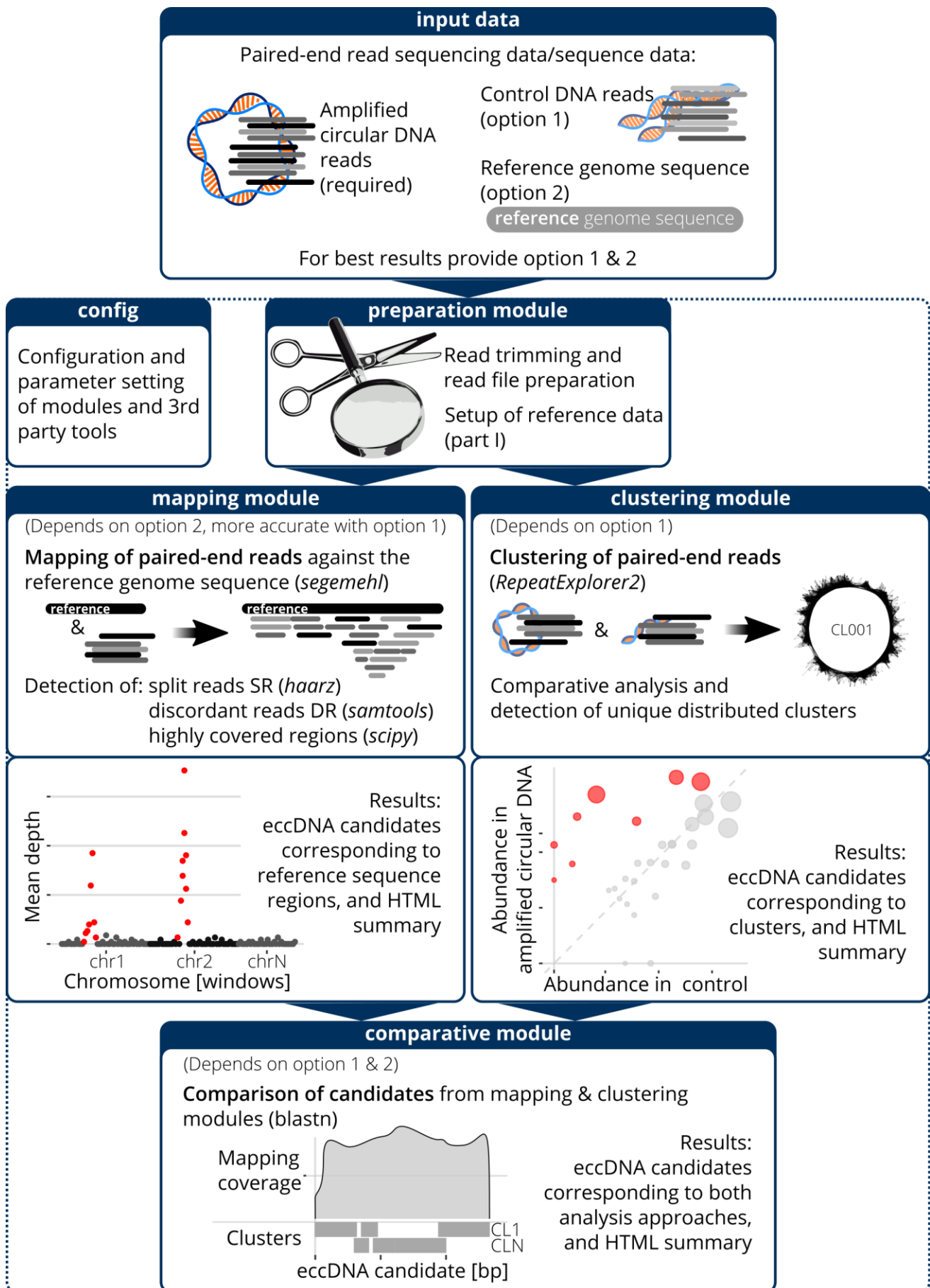


Figure 1: Pipeline workflow scheme.

Command line input files

Paired-end nucleotide sequences in FASTA or FASTQ format can be used as input for the ECCsplorer pipeline.

Compressed input is supported using either gzip or bzip2 formats when specifying the `-trm/--trim_reads` option.

Decompression is applied automatically when the appropriate file extensions are used (.gz or .bz2).

If FASTA files are used note that no trimming is performed within the pipeline. Trimming reads is crucial.

Alternatively, to the integrated solution with *trimmomatic* you might trim reads yourself before running the pipeline with FASTA sequences using your preferred trimmer (e.g. *cutadapt*, *trim_galore*, etc.).

The paired-end end reads should be placed in separated files. Sequencing services usually will provide separated files.

If reads are download via the SRA-toolkit make sure to use the `--split-files` option.

In order to run ECCsplorer provide two or four read files. Read files are positional arguments:

```
$python ECCsplorer.py <file1A> <file2A> [<file1B>] [<file2B>]
```

Please make sure to not mix up the order data sets A and B and forward (R1) and reverse (R2) read files. The first given file will be handled as forward read file R1 of the first data set A. The second given file will be handled as reverse read file R2 of the first data set A. The third given file will be handled as forward read file R1 of the second data set B. The fourth given file will be handled as reverse read file R2 of the second data set B.

In addition to the read files you can provide a reference (genome) sequence using the `-ref/--reference_genome` option. The reference (genome) sequence should be in FASTA format. It might contain unknown bases (N) but no other ambiguity bases. It should look like:

```
>chr1
AGTTAGCGCGT...
>chr2
NNNCGCTAGT...
...
>chrN
NNNN...
```

Note that given chromosome names will be used in the images and should be in appropriate length.

Besides the reference (genome) sequence you might also use an annotation database(s) for the blast annotation of eccDNA candidates. Currently the blast annotation with additional sequences is only used within the mapping module. By default, the pipeline uses the *RepeatExplorer2* DNA database for annotation if available. You can provide additional reference sequences using the `-d/--database` option. The file(s) should be in FASTA format and might contain multiple sequences. You also might provide multiple files separated by a space:

```
$python ECCsplorer.py -d <file.fasta> [<file2.fasta> ...]
```


Command line optional arguments

`-h/--help` option:

- Shows the help message and exit.

`-ref/--reference_genome <file>` option:

- Specifies the reference (genome) sequence FASTA file for mapping analysis.
- For details see page 7.

`-d/--database <file>` option:

- Specifies reference sequence(s) FASTA file(s) for blast annotations (annotation database).
- Provide multiple files separated by a space or multiple sequences in one FASTA file.
- For details see page 7.

`-out/--outpt_dir <dir>` option:

- Specifies your project output directory. The default is eccpipe and it is created in the current working directory. Note that existing non-reference files will be overwritten.

`-trm/--trim_reads <opt>` option:

- Performs a read trimming with trimmomatic. Read trimming is crucial and strongly recommended.
- Please specify the sequencing adapter for the trimming by entering an option from the following:
 - `nex` (Nextera),
 - `tru2` (TruSeq2), `tru3` (TruSeq3), `tru3-2` (TruSeq3-2)
 - `custom` (see trimmomatic manual, name `UserAdapter.fa`)

`-img/--image_format <opt>` option:

- Chooses the desired image output format.
- Please specify the format by entering an option from the following:
 - `png` (default), `jpeg`, `bmp`, `tiff` or `pdf`
 - Some formats will not be previewed in HTML summaries.
 - PDFs are vector graphics and might be edited with inkscape or similar tools (note the high number of data points, vector graphics may be very slow!)

`-dsa/--preA <txt>` and `-dsb/--preB <txt>` options:

- Sets a prefix for your data set A and B. The prefix is used in the visual output to determine the data sets. In the following the set prefixes will be referred to as PREFIX.
- Prefixes are also used for the comparative clustering.
- They should be at maximum 10 characters long and be equal in length (they also should not be the same).
- The default prefix for data set A is `TR` (treatment) and `CO` (control) for data set B.

`-cnt/--read_count <int/opt>` option:

- Specifies the number of reads to be used in the clustering analysis.
- The read count for the clustering analysis should not exceed 0.1x – 0.2x genome coverage.
- The maximum number of available reads is used by default (max. reads = smallest number of reads in any data set).
- Use auto to use a 0.1 x genome coverage if a reference genome sequence is provided or the approximate genome size is set manually in configuration file (see page 16).

`-win/--window_size <int>` option:

- Window size used for the mapping analysis whereas the reference (genome) sequence average coverage is calculated over windows. The coverage is later used to find peaks and also visualized.
- The default window size is 100 bp. Smaller window size increases memory usage and computational time but might also increase sensitivity. Larger window size shortens memory and time usage and is recommend for bigger genomes.

`-rgs/--genome_size <int>` option:

- Set the genome size of your organism in base pairs [bp].
- Only needed if `-cnt` set to `auto`.

`-tax/--taxon <opt>` option:

- Specifies taxon using `vir` for Viridiplantea (default) or `met` for Metazoa.
- Especially important for the clustering analysis to define the protein database to be used.

`-cpu/--max_threads <int>` option:

- Specify the number of threads (CPU cores) to use during analysis. By default, all available are used. If running into memory issues, try decreasing used threads.

`-log` option:

- Prints logging to file.
- If logging is not set it is printed to console (`stdout`).

`-m/--mode <opt>` option:







- Options:
 - `all` (default, run all modules), `map` (run only mapping module), `clu` ((run only clustering module), `PRExer` (run only preparation module, for a manual RepeatExplorer run)
- Can be used with `-dsa/--preA`, `-dsb/--preB`, `-trm/--trim_reads` and `-cnt/--read_count`.
- The pipeline will perform no software checkup and stop after the read preparation steps.
- The file for a separate RepeatExplorer run can be found in `output_dir/read_data/` named `REPEATEXPLORER_READY.fa`.

Output files

All output files except for the logging file (if `-log` set) will be written into the `--output_dir` (default: `eccpipe`) directory containing the sub directories `reference_data`, `read_data` and `eccpipe_results`. The `eccpipe_results` directory will include the directories `mapping_results`, `clustering_results`, `comparative_results` and the HTML summary files. Each module creates several output files. The logging file will be written in the current working directory. Depending on used input files or chosen options some of these files might be missing. Note that BED files might have modified optional fields shown with H: [column name]. For further explanation on the modified BED format including used abbreviations see page 15.

Output files per directory

Symbols:

- <ITALIC>* depending on input or option
-  output directory (default: `eccpipe`)
-  sub directory
-  text file (e.g. FASTA/Q)
-  table like file (e.g. CSV or BED, tab separated)
- H: Header (column names, not included in BED files)
-  image file (e.g. PNG, defined by `-img` option)
-  summary file (HTML)

H: Header abbreviations

- chr reference genome sequence identifier (e.g. chromosome)
- start start position of the feature
- end end position of the feature
- n count
- id identifier of feature (e.g. read name or candidate identifier)
- s strand (+/-)













eccpipe → reference_data	
Reference files	■ <i>REFERENCE.fasta</i> User input reference (genome) sequence.
	■ <i>REFERENCE.fasta.fai</i> <i>Samtools</i> related index of reference sequence.
	■ <i>REFERENCE.idx</i> <i>segemehl</i> related index of reference sequence.
	■ <i>REFERENCE.chrSize.txt</i> Chromosome size file of reference sequence.
	■ <i>REFERENCE.winWIN.bed</i> Windowed reference sequence.
	■ <i>x00winWIN-part.bed</i> Partial windowed reference sequence. Used for multi-threading.
	■ <i>DATABASE.fasta</i> User input reference sequences database.
	■ <i>DATABASE.fasta.nhr</i>
	■ <i>DATABASE.fasta.nin</i> <i>Blast+</i> database related files of user input reference database.
	■ <i>DATABASE.fasta.nsq</i>
	■ <i>combinedDB.fas.nal</i> <i>Blast+</i> combined databases file.

eccpipe → read_data	
Read files	■ <i>PREa-R1_trim.fq</i> Trimmed user input paired-end reads of data set A.
	■ <i>PREa-R2_trim.fq</i>
	■ <i>PREa-R1_utrim.fq</i> Trimmed user input unpaired reads of data set A.
	■ <i>PREa-R2_utrim.fq</i>
	■ <i>PREa-R1_trim.fa</i> Trimmed and converted user input paired-end reads of data set A.
	■ <i>PREa-R2_trim.fa</i>
	■ <i>PREb-R1_trim.fq</i> Trimmed user input paired-end reads of data set B.
	■ <i>PREb-R2_trim.fq</i>
	■ <i>PREb-R1_utrim.fq</i> Trimmed user input unpaired reads of data set B.
	■ <i>PREb-R2_utrim.fq</i>
	■ <i>PREb-R1_trim.fa</i> Trimmed and converted user input paired-end reads of data set B.
	■ <i>PREb-R2_trim.fa</i>
	■ <i>REPEATEXPLORER_READY.fa</i> Interlaced and concatenated read files for usage with <i>RepeatExplorer2</i> .







eccpipe → eccpipe_results	
Summary files	→ <i>mapping_results</i> Results from mapping module.
	→ <i>clustering_results</i> Results from clustering module.
	→ <i>comparative_results</i> Results from comparative module.
	☑ <i>eccMap_summary.html</i> Summary from mapping module.
	☑ <i>eccCL_summary.html</i> Summary from clustering module.
	☑ <i>eccComp_summary.html</i> Summary from comparative module.







■ eccpipe → eccpipe_results → mapping_results

→ candidates	Results from mapping for individual candidates.
→ temp	Temporary plotting files. Can be deleted occasionally.
■ <i>PREa</i> .sam	
■ <i>PREb</i> .sam	Alignment files in SAM or BED format. Paired-end reads from different data sets mapped to the reference sequence.
■ <i>PREa</i> .bed	H: [chr] [start] [end] [id] [n] [s]
■ <i>PREb</i> .bed	
■ <i>PREa_unmapped</i> .sam	Alignment files in SAM format. Unmatched paired-end reads from different data sets.
■ <i>PREb_unmapped</i> .sam	
■ <i>PREa</i> -sam-stats.txt	
■ <i>PREb</i> -sam-stats.txt	Statistic information of mappings.
■ <i>PREa_aligned</i> -SR.bed	Aligned split reads (SR).
■ <i>PREb_aligned</i> -SR.bed	H: [chr] [start] [end] [id+1] [n] [s]
■ <i>PREa_haarz</i> -SR.bed	SR regions according to aligned SR (<i>haarz</i> tool). H: [chr] [start] [end] [n] [quality] [file] [n] [n] ¹
■ <i>PREa_regions</i> -SR.bed	Merged SR regions. H: [chr] [start] [end]
■ <i>PREa_aligned</i> -DR.bed	Aligned discordant mapping reads (all). H: [chr] [start] [end] [n] [s] [n]
■ <i>PREa_aligned</i> -DR-nF2.bed	Aligned discordant mapping reads (no proper pair). H: [chr] [start] [end] [n] [s]
■ <i>PREa_aligned</i> -DR-F83.bed	Aligned discordant mapping reads (unusual orientation). H: [chr] [start] [end] [n] [s]
■ <i>PREa_aligned</i> -DR-F163.bed	
■ <i>PREb_aligned</i> -DR.bed	Aligned discordant mapping reads (all). H: [chr] [start] [end] [n] [s] [n]
■ <i>PREb_aligned</i> -DR-nF2.bed	Aligned discordant mapping reads (no proper pair). H: [chr] [start] [end] [n] [s]
■ <i>PREb_aligned</i> -DR-F83.bed	Aligned discordant mapping reads (unusual orientation). H: [chr] [start] [end] [n] [s]
■ <i>PREb_aligned</i> -DR-F163.bed	
■ <i>PREa_peak-region</i> -all.bed	Summarized regions with high general coverage. H: [chr] [start] [end]
■ <i>PREa_peak-region</i> -SR.bed	Summarized regions with high SR coverage. H: [chr] [start] [end]
■ <i>PREa_peak-region</i> -DR.bed	Summarized regions with high DR coverage. H: [chr] [start] [end]
■ <i>PREb_peak-region</i> -all.bed	Summarized regions with high general coverage. H: [chr] [start] [end]
■ <i>PREb_peak-region</i> -SR.bed	Summarized regions with high SR coverage. H: [chr] [start] [end]
■ <i>PREb_peak-region</i> -DR.bed	Summarized regions with high DR coverage. H: [chr] [start] [end]
■ <i>PREa_hicnfdc-ecc</i> -regions.bed	High confident eccDNA regions (3/3). H: [chr] [start] [end]

	 <i>PREa_enr_hicnfdc-ECC-REGIONS.bed</i>	Enriched eccDNA regions with high confidence (3/3). H: [chr] [start] [end] [enrichment score] [id]
	 <i>PREa_cnfdc-ECC-regions.bed</i>	Summary of eccDNA regions with lower confidence (2/3) H: [chr] [start] [end]
	 <i>PREa_cnfdc-ECC-regions_SR-all.bed</i>	Regions with SR and general high coverage. H: [chr] [start] [end]
	 <i>PREa_cnfdc-ECC-regions_DR-all.bed</i>	Regions with DR and general high coverage. H: [chr] [start] [end]
	 <i>PREa_cnfdc-ECC-regions_SR-DR.bed</i>	Regions with SR and DR. H: [chr] [start] [end]
	 <i>PREa-PREb_coverage-summary_RAW.csv</i>	Mean coverage over reference sequence windows.
	 <i>PREa-PREb_coverage-summary_NORM.csv</i>	Normalized mean coverage over reference sequence windows.
	 <i>PREa-PREb_region-coverage-summary_RAW.csv</i>	Mean coverage over high confident eccDNA regions.
	 <i>PREa-PREb_region-coverage-summary_NORM.csv</i>	Normalized mean coverage over eccDNA regions with high confidence and enrichment values.
	 <i>PREa-PREb_chr_manhattan-plot.png</i>	Manhattan plot over reference sequence (Figure 2).
	 <i>PREa-PREb_ECC-SEQUENCES.fasta</i>	Sequences of eccDNA regions with high confidence.
	 <i>*trns.txt & *mult.bed</i>	<i>segemehl</i> related files ¹ .

¹ for further details see *segemehl* manual

Candidate files	 eccpipe → eccpipe_results → mapping_results → candidates → eccCand_000	
	 eccCand_000.fasta	Candidate sequences.
	 eccCand_000_blast.m6	Candidate <i>Blast+</i> results.
	 eccCand_000_coverage-summary_RAW.csv	Per base pair coverage over candidates.
	 eccCand_000_coverage-summary_NORM.csv	Normalized per base pair coverage over candidates.
	 eccCand_000_collage-plot.png	Coverage plots over candidates (Figure 3).

Clustering files	 eccpipe → eccpipe_results → clustering_results	
	 cluster_table_sngl-lined.csv	Summary of clusters in a single line per cluster format (equivalent to the CLUSTER_TABLE.csv).
	 comparative_cluster_table.csv	Summary of clusters including read counts.
	 COMPARATIVE_CLUSTER_TABLE_eccCANDIDATES.csv	Summary of eccDNA candidate clusters including annotation and read counts.
	 comp_cl_tab_eccCandidates_list.csv	List with all eccDNA candidates cluster number.
	 summary_scatter.png	Abundance scatter plot (Figure 4).
	* other files & directories	
	<i>RepeatExplorer2</i> related files. For more information please refer to the <i>RepeatExplorer2</i> manual.	

eccpipe → eccpipe_results → comparative_results		
Comparative files	→ candidates	Results from comparison for individual candidates.
	eccMapper_cand.fa	Mapping resulting sequences.
	eccMapper_cand.fa.nhr	
	eccMapper_cand.fa.nin	Blast+ database related files.
	eccMapper_cand.fa.nsq	
	blast_CL000.m6	Cluster Blast+ result. Cluster sequences against mapping resulting sequences.
	comparative-summary_manhattan.png	Comparative Manhattan plot (Figure 2).
	comparative-summary_scatter.png	Comparative abundance scatter plot (Figure 4).
eccpipe → eccpipe_results → comparative_results → candidates → eccCand_000		
	eccCand_000_comparative-blast.csv	Blast+ results. Mapping candidate related clusters.
	eccCand_000_comparative-plot.png	Coverage plots over candidate and cluster annotation (Figure 5).

Special output formats and image explanation

Interlaced and concatenated reads format

If input are four files or the `-PRExer` option is set, there will be a `REPEATEXPLORER_READY.fa` file. This file will contain the interlaced, concatenated (if four files input), trimmed (if `-trm/--trim_reads` set), sub sampled (if `-cnt/--read_count` set) and converted (if input in FASTQ) paired-end reads from the input data set(s). In addition all reads will be cut to equal length and read identifiers will be extended.

Extended read identifiers: `PREFIX_read.id_#PAIR-TAG`

```
$cat REPEATEXPLORER_READY.fa
```

```
>TR_read.1_#0/1
AGTTAGCGCGT...
>TR_read.1_#0/2
TGCATGCGCTA...
...
>TR_read.n_#0/1
GTGAGTGATAC...
>TR_read.n_#0/2
TTGCGTATGTA...
...
>CO_read.1_#0/1
GTACGTGCGCT...
>CO_read.1_#0/2
GTATATCGCAT...
...
```

```
>CO_read.n_#0/1
ACTGCTAGCAT...
>CO_read.n_#0/2
ACTGGATGCAT...
```

Modified BED format

The BED format consists of one line per feature, each containing 3+ tab separated columns of data.

Columns one to three are essential and provide information on reference sequence (e.g. chromosome), starting position and ending position of features.

Additional columns might contain further information on each feature, such as strand, name of feature, or enrichment scores.

```
$cat file.bed

[chr] [start]      [end] [information] # header not in files
chr2  0          100  feature_1
chr4  200        5331 feature_2
```

Abbreviations for BED file headers:

H	modified BED format header (not in file included)
chr	reference genome sequence identifier (e.g. chromosome)
start	start position of the feature
end	end position of the feature
n	count
id	identifier of feature (e.g. read name or candidate identifier)
s	strand (+/-)

Windowed coverage plot (Manhattan plot)

Image (png, pdf, etc.) format with file ending: `_manhattan-plot.img`

Point plot for coverage visualization over a whole genome.

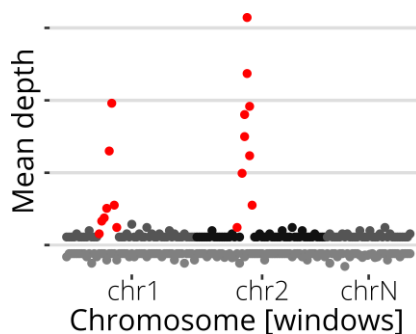


Figure 2: Schematic Manhattan plot. Dots ● represents the mean coverage over a window on the reference genome sequence (window size is set with the `-win/--window_size` option). Windows containing eccDNA candidates are colored in red ●. Control data ● is plotted below the x-axis. Coverage points are normalized to mapped base pairs per million mapped base pairs (BPM).

Coverage plot

Image (png, pdf, etc.) format with file ending: `_collage-plot.img`

Area plot for coverage visualization over eccDNA candidate regions.

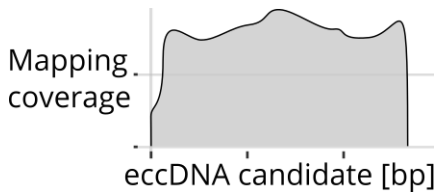


Figure 3: Schematic representation of a normalized mapping coverage plot over an eccDNA candidate region. Output Image contains three to six plot panels.

The top plot represents the general coverage (`_map.all`), the middle plot shows only split read coverage (`_map.SR`) and at the bottom the discordant read coverage (`_map.DR`) is shown. The right panels represents the control data if applicable.

Abundance scatter plot

Image (png, pdf, etc.) format with file ending: `_summary-scatter.img`

Scatter plot for visualization of abundance of clusters in data sets.

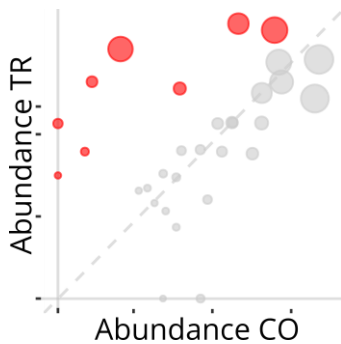


Figure 4: Schematic representation of an abundance scatter plot. Dots represents a cluster. Dot size shows the total number of reads in the cluster. The abundance of the cluster is the logarithmic read count from one data set. An abundance of zero indicates empty clusters. Clusters where more than 80% (default) of the reads inherit from the treatment data set are shown in red and used for further analysis.

Coverage plot with annotation

Image (png, pdf, etc.) format with file ending: `_comparative-plot.img`

Area plot for coverage visualization over mapping candidates and annotation of related clusters.

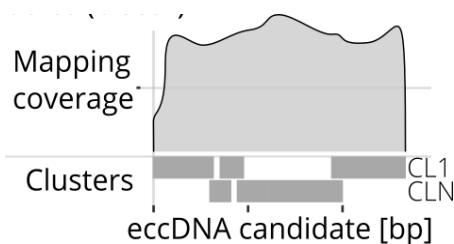


Figure 5: Schematic representation of a normalized mapping coverage plot over an eccDNA candidate sequence and corresponding clusters. Cluster boxes represent blast hits with sequence similarities.

config.py: pipeline configuration

The config.py contains the configuration for the ECCsplorer pipeline including names of directories, PATH values, commands and set parameters of 3rd party bioinformatic tools and other analysis parameters.

In case to run the ECCsplorer pipeline properly you might edit the PATH values of several 3rd party tool. The default values are:

```
TOOL_PATH = 'tool'
```

Example (*segemehl*):

```
SEGEMEHL_PATH = 'segemehl.x'
```

If you did not add the *segemehl* tool to the `$PATH` environment variable you need to change this value to look like:

```
SEGEMEHL_PATH = '/directory/path/to/segemehl-0.3.4/segemehl.x'
```

The 3rd party tool parameters and the pipeline parameters might be edited to fit individual needs. The following parameters can be edited in the configuration file:

3rd party tool parameters (*segemehl*, *Blast+*, *RepeatExplorer2*, *Trimmomatic*). For detailed information please refer to the corresponding manual.

Pipeline parameters (to be edited to fit individual needs):

- Approximate eccDNA length: `MAX_eccDNA_LENGTH` & `MIN_eccDNA_LENGTH`
- Merging distance of two candidates: `MERGE_CLOSE_REGIONS`
- Peak finder parameters: `PEAK_THRESHOLD` & `PEAK_DISTANCE`
- Enrichment threshold for high confident eccDNA candidate regions: `ENRICH_THRESHOLD`
- Reference sequence window size (default value): `WINDOW_SIZE` (~ `-win/--window_size`)
- Minimal cluster proportion for cluster candidates: `REPEX_ECC_PROPORTION`
- Approximate genome size in bp: `REPEX_GENOME_SIZE`
- SAMflags for discordant mapping reads: `SAMFLAGS_DR` & `SAMFLAGS_DR_not`
- Image parameters: `IMAGE_RES`, `IMAGE_WIDTH`, `IMAGE_HEIGHT` & `IMAGE_POINTS`
- Rserve port: `PORT`