Running the Rp-Bp pipeline

The Rp-Bp pipeline consists of an index creation step, which must be performed once for each genome and set of annotations, and a two-phase prediction pipeline, which must be performed for each sample.

In the first phase of the prediction pipeline, a filtered genome profile is created. In the second phase, the ORFs which show evidence of translation in the profile are predicted.

This document describes the steps to run each of these phases. It shows some sample calls. For all programs, the `--help` option can be given to see the complete list of parameters.

- Creating reference genome indices
- Running the Rp-Bp pipeline
  - Creating filtered genome profiles
  - Predicting translated open reading frames
- Logging options
- Parallel processing options (SLURM options)

Creating reference genome indices

This section describes the steps necessary to prepare a reference genome and matching annotation for use in the Rp-Bp pipeline. The process must only be run once for each reference genome and set of annotations.

Please see creating reference indices for more details about choosing the appropriate files from Ensemble, etc., for Rp-Bp.

The entire index creation process can be run automatically using the `prepare-rpbp-genome` scripts. It reads most of the required paths from a YAML configuration file. Additionally, it automatically creates some of the output paths.

The script accepts a `--overwrite` flag. Unless this is given, then steps for which the output file already exists will be skipped.

It also accepts a `--do-not-call` flag. If this flag is given, then the commands below will be printed but not executed.

Logging options can be given to this script.

Parallel processing options can be given to this script.

Configuration file keys

The following keys are read from the configuration file. Keys with [`brackets`] are optional.

- `gtf`. The path to the reference annotations
- `fasta`. The path to the reference genome sequence
- `ribosomal_fasta`. The path to the ribosomal sequence
- [`de_novo_gtf`]. An additional GTF containing annotaions constructed from a *de novo* assembly (for example, from StringTie). See below for how *de novo* assemblies are handled.
- `genome_base_path`. The path to the output directory for the transcript fasta and ORFs
- `genome_name`. A descriptive name to use for the created files
- `ribosomal_index`. The base output path for the Bowtie2 index of the ribosomal sequence
- `star_index`. The base output path for the STAR index for the reference genome sequence. This STAR index *does not* include the transcript annotations. They will be incorporated later while running the specific samples.
- [`orf_note`]. An additional description used in the filename of the created ORFs
- [`start_codons`]. A list of strings that will be treated as start codons when searching for ORFs. default: [`ATG`]
- [`stop_codons`]. A list of strings that will be treated as stop codons when searching for ORFS. default: [`TAA`, `TGA`, `TAG`]
- [`sjdb_overhang`]. The value to use for splice junction overlaps when constructing the STAR index. default: 50

ORF labels

As part of the index creation phase, ORFs are labeled according to their location relative to the annotated, canonical transcripts, that is, annotated CDS regions. We use the following labels (essentially, the same as those given in the supplement of the paper):

- `canonical`. an ORF which exactly coincides with an annotated transcript
- `canonical_extended`. an ORF which starts upstream of a `canonical` ORF, but otherwise has the same structure

- `canonical_truncated`. an ORF which starts downstream of a `canonical` ORF, but otherwise has the same structure
- `five_prime`. an ORF which is completely in the annotated 5' leader region of a transcript and does not overlap other `canonical` ORFs on the same strand (such as from another isoform)
- `three_prime`. an ORF in the annotated 3' trailer region of a transcript and does not overlap other `canonical` ORFs on the same strand
- `noncoding`. an ORF from a transcript not annotated as coding, such as a lncRNA or pseudogene.
- `five_prime_overlap`. an ORF in the annotated 5' leader region of a transcript but which overlaps a `canonical` ORF; this could either be out-of-frame with respect to the `canonical` ORF of the same transcript or otherwise overlap a `canonical` ORF from another isoform.
- `three_prime_overlap`. an ORF in the annotated 3' trailer region of a transcript but which overlaps a `canonical` ORF
- `suspect`. an ORF which partially overlaps the interior of a `canonical` ORF. These can result from things like retained introns.
- `within`. an out-of-frame ORF in the interior of a `canonical` ORF

Input files

The required input files are those suggested by the configuration file keys.

- `gtf`. The GTF/GFF3 file containing the reference annotations. The file must follow standard conventions for annotating transcripts. This means at least the `exon` features must be present, and the transcript identifiers (`transcript_id` attribute) must match for exons from the same transcript. Furthermore, the ORFs are labeled based on their positions relative to annotated coding sequences. For it to work correctly, the `CDS` features must also be present in the annotation file. The start codon should be included in the `CDS`, but the stop codon should not. Following standard conventions, the genomic locations are taken to be base-1 and inclusive. Other feature types, such as `gene`, `start_codon` and `stop_codon` may be present in the file, but they **are not used** for extracting ORFs.

- `fasta`. The input fasta file should contain all chromosome sequences. The identifiers must match those in the GTF file. Typically, the "primary assembly" file from Ensembl contains the appropriate sequences and identifiers. In particular, the "top level" Ensembl genome assembly includes haplotype information that is not used by Rp-Bp; for human, this is quite large and can substantially slow down the pipeline. Please see Ensembl for more information about the differences between assemblies.

- `ribosomal_fasta`. The ribosomal DNA sequence is typically repeated many times throughout the genome. Consequently, it can be difficult to include in the genome assembly and is often omitted. Therefore, a separate

fasta file is required for these sequences (which are later used to filter reads). This file could also include other sequences which should be filtered, such as tRNA sequences downloaded from GtRNAdb, for example.

- `de_novo_gtf`. The GTF/GFF3 file containig the *de novo* assembled transcripts.

de novo assembled transcripts

In principle, there is no difference between "standard" annotated transcripts and *de novo* assembled transcripts. In both cases, ORFs are extracted from the transcripts based on the given `start_codons` and `stop_codons`. However, it is often of scientific interest to distinguish between ORFs from annotated transcripts and "novel" ones from the *de novo* assembly.

Both to avoid reporting spurious "novel" translated ORFs and to avoid redundant calculations, we remove each ORF identified in the *de novo* assembly which exactly match an ORF using only the annotations.

We then label ORFs from the *de novo* assembly as follows, with respect to their relationship to the annotated transcripts. Due to the first filtering step, the following categories include only ORFs which are at least partially only supported in the *de novo* assembly.

- `novel`. The ORF does not overlap the annotations at all.
- `novel_canonical_extended`. an ORF which starts upstream of a `canonical` ORF, but otherwise has the same structure.
- `novel_noncoding`. an ORF from a transcript not annotated as coding, such as a lncRNA or pseudogene.
- `novel_five_prime_overlap`. an ORF in the annotated 5' leader region of a transcript but which overlaps a `canonical` ORF; this could either be out-of-frame with respect to the `canonical` ORF of the same transcript or otherwise overlap a `canonical` ORF from another isoform.
- `novel_three_prime_overlap`. an ORF in the annotated 3' trailer region of a transcript but which overlaps a `canonical` ORF
- `novel_suspect`. an ORF which partially overlaps the interior of a `canonical` ORF. These can result from things like retained introns.
- `novel_within`. an out-of-frame ORF in the interior of a `canonical` ORF

The other labels, such as `novel_canonical` or `novel_five_prime` are guaranteed to be filtered in the first step based on their definitions.

Output files

- `<genome_base_path>/ribosomal_index/`. The bowtie2 index files (`ribosomal_index`.1.bt2, etc.) for the ribosomal_fasta file

- `<genome_base_path>/star_index/`. The STAR index files (`SA`, `Genome`, etc.) for the `fasta` file
- `<genome_base_path>/<genome_name>.annotated.bed.gz`. A bed12 file containing all transcripts. For coding transcripts, the thick_start and thick_end columns give the start and end of the coding region; the start codon *is* included in the thick region, but the stop codon *is not*. For noncoding transcripts, thick_start and thick_end are both -1. This seems to behave as expected in IGV.

From annotations

- `<genome_base_path>/transcript-index/<genome_name>.transcripts.annotated.fa`. The sequences of all annotated transcripts.
- `<genome_base_path>/transcript-index/<genome_name>.genomic-orfs.annotated.<orf_note>.bed` A bed12+ file containing all ORFs from annotated transcripts. Besides the standard bed12 columns, this file includes columns giving the orf_type, orf_length, and orf_num. The ORF ids are of the form: `transcript_seqname:start-end:strand`. The start codon *is* included in the ORF, but the stop codon *is not*. The thick_start and thick_end are always the same as start and end.
- `<genome_base_path>/transcript-index/<genome_name>.orfs-exons.annotated.<orf_note>.bed.g` A bed6+2 file containing each exon from each ORF. The "id" of an exon corresponds exactly to the ORF to which it belongs; exons from the same ORF have the same "id". The extra columns are "exon_index", which gives the order of the exon in the transcript, and "transcript_start", which gives the start position of that index in transcript coordinates. Exons are always sorted by "lowest start first", so the order is really reversed for ORFs on the reverse strand.

From *de novo* assembly.

The semantics of these files are the same of those from the annotations, but created using the `de_novo_gtf` files. N.B., ORFs which completely overlap annotations are not included.

- `<genome_base_path>/transcript-index/<genome_name>.transcripts.de-novo.fa`
- `<genome_base_path>/transcript-index/<genome_name>.genomic-orfs.de-novo.<orf_note>.bed.g`
- `<genome_base_path>/transcript-index/<genome_name>.orfs-exons.de-novo.<orf_note>.bed.gz`

From both annotations and *de novo* assembly.

The semantics are again the same as above. If a *de novo* assembly was not provided, these are simply symlinks to the respective "annotations" files. Otherwise, they are the concatenation of the respective "annotation" and "*de novo*" files.

- <genome_base_path>/transcript-index/<genome_name>.genomic-orfs.<orf_note>.bed.gz
- <genome_base_path>/transcript-index/<genome_name>.orfs-exons.<orf_note>.bed.gz

```
prepare-rpbp-genome WBcel235.79.chrI.yaml --num-cpus 2 --mem 4G --overwrite --logging-level
```

Back to top

Running the Rp-Bp pipeline

The entire Rp-Bp pipeline can be run on a set of riboseq samples which all use the same genome indices using a sample sheet-like YAML configuration file with the `run-all-rpbp-instances` program. The configuration file respects all of the optional configuration options specified below.

The script accepts the following options:

- `--tmp <loc>`. If this flag is given, then all relevant calls will use `<loc>` as the base temporary directory. Otherwise, the program defaults will be used.
- `--overwrite`. Unless this is given, then steps for which the output file already exists will be skipped.
- `--keep-intermediate-files`. Unless this flag is given, large intermediate files, such as fastq files output by flexbar after removing adapters, will be deleted.
- `--flexbar-format-option`. Older versions of flexbar used `format` as the command line option to specify the format of the fastq quality scores, while newer versions use `qtrim-format`. Depending on the installed version of flexbar, this option may need to be changed. Default: `qtrim-format`
- `--star-executable`. In principle, `STARlong` (as opposed to `STAR`) could be used for alignment. Given the nature of riboseq reads (that is, short due to the experimental protocols of degrading everything not protected by a ribosome), this is unlikely to be a good choice, though. Default: `STAR`
- `--star-read-files-command`. The input for `STAR` will always be a gzipped fastq file. `STAR` needs the system command which means "read a gzipped text file". As discovered in Issue #35, the name of this command is different on OSX and ubuntu. The program now attempts to guess the name of this command based on the system operating system, but it can be explicitly specified as a command line option. Default: `gzcat` if `sys.platform.startswith("darwin")`; `zcat` otherwise. Please see python.sys documentation for more details about attempting to guess the operating system.

Logging options can be given to this script.

Parallel processing options can be given to this script.

Using replicates

The Rp-Bp pipeline handles replicates by adding the (smoothed) ORF profiles. The Bayes factors and predictions are then calculated based on the combined profiles. The `--merge-replicates` flag indicates that the replicates should be merged. By default, if the `--merge-replicates` flag is given, then predictions will not be made for the individual datasets. The `--run-replicates` flag can be given to override this and make predictions for both the merged replicates as well as the individual datasets.

Configuration file keys

The following keys are read from the configuration file. Their semantics is exactly as described below. (This text is in both places for convenience.)

- `riboseq_data`. The base output location for all created files.
- [`note`]. An optional string which will be added to all filenames. It should not contain spaces or any other special characters.
- [`models_base`]. The base path to the compiled models, the base path to the compiled models. The models specified in the paper are included with the source distribution and compiled/pickled as part of the installation process. They are installed in an operating system-specific location (in particular, `user_data_dir` from the appdirs package. This location is determined during installation and does not normally need to be changed. For development, they may be in some alternative location.

Reference genome options

These options should be exactly the same as those used in the configuration file used to create the reference indices.

- `gtf`. The path to the reference annotations
- `genome_base_path`. The path to the output directory for the transcript fasta and ORFs
- `genome_name`. A descriptive name to use for the created files
- `ribosomal_index`. The base path for the Bowtie2 index of the ribosomal sequence
- `star_index`. The base path to the STAR index
- `fasta`. The path to the reference genome sequence

Samples specification

- `riboseq_samples`. A dictionary in which each entry specifies a sample. The key is an informative name about the sample, and the value gives the complete path to the sequencing file (a fastq or fastq.gz file). The names

will be used to construct filenames, so they should not contain spaces or other special characters.

- `riboseq_biological_replicates`. A dictionary in which each entry species one condition and all samples which are replicates of the condition. The key of the dictionary is a string description of the condition, and the value is a list that gives all of the sample replicates which belong to that condition. The names of the sample replicates must match the dataset names specified in `riboseq_samples`.

```
# do not merge replicates
run-all-rpbp-instances c-elegans-test.yaml --tmp /scratch/bmalone/ --num-cpus 10 --logging-l

# merging the replicates, do not calculate Bayes factors and make predictions for individual
run-all-rpbp-instances c-elegans-test.yaml --tmp /scratch/bmalone/ --overwrite --num-cpus 2

# merging the replicates and calculating Bayes factors and making predictions for individual
run-all-rpbp-instances c-elegans-test.yaml --tmp /scratch/bmalone/ --overwrite --num-cpus 2
```

Back to top

Creating ORF profiles

The entire profile creation process can be run automatically using the `create-orf-profiles` script. It reads most of the required paths from a YAML configuration file. Additionally, it automatically creates some of the output paths.

The script accepts a `--overwrite` flag. Unless this is given, then steps for which the output file already exists will be skipped.

It also accepts a `--do-not-call` flag. If this flag is given, then the commands below will be printed but not executed.

The script accepts a `--keep-intermediate-files` flag. Unless this flag is given, large intermediate files, such as fastq files output by flexbar after removing adapters, will be deleted.

Logging options can also be given to this script.

Parallel processing options can be given to this script.

Configuration file keys

The following keys are read from the configuration file. Keys with [`brackets`] are optional.

- `riboseq_data`. The base output location for all created files.

- [note]. An optional string which will be added to all filenames. It should not contain spaces or any other special characters.
- [models_base]. The base path to the compiled models, the base path to the compiled models. The models specified in the paper are included with the source distribution and compiled/pickled as part of the installation process. They are installed in an operating system-specific location (in particular, user_data_dir from the appdirs package. This location is determined during installation and does not normally need to be changed. For development, they may be in some alternative location.

Reference genome options

These options should be exactly the same as those used in the configuration file used to create the reference indices.

- gtf. The path to the reference annotations
- genome_base_path. The path to the output directory for the transcript fasta and ORFs
- genome_name. A descriptive name to use for the created files
- ribosomal_index. The base path for the Bowtie2 index of the ribosomal sequence
- star_index. The base path to the STAR index

Flexbar options

- [adapter_file]. A fasta file containing a set of adapter sequences used by flexbar. default: None
- [max_uncalled]. The maximum number of Ns to permit in a read without filtering. default 1
- [pre_trim_left]. The number of bases to remove from the 5' end of all reads: default: 0
- [adapter_sequence]. A single sequence used to remove adapters within flexbar. default: None
- [quality_format]. The quality format of the reads in the raw fastq file. default: "sanger"

STAR options

- [align_intron_min]. default: 20
- [align_intron_max]. default: 100000
- [out_filter_intron_motifs]. default: RemoveNoncanonicalUnannotated
- [out_filter_mismatch_n_max]. default: 1
- [out_filter_mismatch_n_over_l_max]. default: 0.04
- [out_filter_type]. default: BySJout
- [out_sam_attributes]. default: AS NH HI nM MD

- [sjdb_overhang]. default: 50

Multimapper options

- [keep_riboseq_multimappers]: If this variable is present in the config file with any value (even something like "no" or "null" or "false"), then multimapping riboseq reads *will not* be removed. They will be treated as "normal" reads in every place they map. That is, the weight of the read will not be distributed fractionally, probabilistically, etc.

Metagene periodicity options

- [seqids_to_keep]. If this list is given, then only transcripts appearing on these identifiers will be used to construct the metagene profiles (and other downstream analysis). The identifiers must match exactly (e.g., "2" and "chr2" do not match)
- [metagene_profile_start_upstream]. The number of bases upstream of the translation initiation site to begin constructing the metagene profile. default: 50
- [metagene_profile_start_downstream]. The number of bases downstream of the translation initiation site to end the metagene profile. default: 20
- [metagene_profile_end_upstream]. The number of bases upstream of the translation termination site to begin constructing the metagene profile. default: 50
- [metagene_profile_end_downstream]. The number of bases downstream of the translation termination site to end the metagene profile. default: 20
- [periodic_offset_start]. The position, relative to the translation initiation site, to begin calculating periodicity Bayes factors (inclusive)
- [periodic_offset_end]. The position, relative to the translation initiation site, to stop calculating periodicity Bayes factors (inclusive)
- [metagene_profile_length]. The length of the profile to use in the models. metagene_profile_length + periodic_offset_end must be consistent with the length of the extracted metagene profile
- [metagene_profile_iterations]. The number of iterations to use for each chain in the MCMC sampling. The first half of the iterations are discarded as burn-in samples. All of the remaining samples are used to estimate the posterior distributions. That is, we do not use thinning. default: 500

Periodicity and offset options

- [use_fixed_lengths]. If this variable is present in the config file with any value (even something like "no" or "null" or "false"), then the estimated periodic read lengths and offsets will not be used. Instead, fixed values given by lengths and offsets (below) will be used.
- [lengths]. A list of read lengths which will be used for creating the profiles if the use_fixed_lengths option is given. Presumably, these are lengths that have periodic metagene profiles.
- [offsets]. The P-site offset to use for each read length specifed by --lengths if the use_fixed_lengths option is given. The number of offsets must match the number of lengths, and they are assumed to match. For example lengths of 26, 29 with offsets 9, 12 means only reads of lengths 26 bp and 29 bp will be used to create the profiles. The 26 bp reads will be shifted by 9 bp in the 5' direction, while reads of length 29 bp will be shifted by 12 bp.
- [min_metagene_profile_count]. If fixed lengths are not used: the minimum number of reads for a particular length in the filtered genome profile. Read lengths with fewer than this number of reads will not be used. default: 1000
- [min_metagene_bf_mean]. If fixed lengths are not used: if max_metagene_profile_bayes_factor_var is not None, then this is taken as a hard threshold on the estimated Bayes factor mean.

If min_metagene_profile_bayes_factor_likelihood is given, then this is taken as the boundary value; that is, a profile is "periodic" if:

```
[P(bf > min_metagene_bf_mean)] > min_metagene_bf_likelihood
```

If both max_metagene_bf_var and min_metagene_bf_likelihood are None, then this is taken as a hard threshold on the mean for selecting periodic read lengths.

If both max_metagene_bf_var and min_metagene_bf_likelihood are given, then both filters will be applied and the result will be the intersection.

default: 5

- [max_metagene_bf_var]. If fixed lengths are not used: if given, then this is taken as a hard threshold on the estimated Bayes factor variance. default: None, i.e., this filter is not used. (null in yaml)
- [min_metagene_bf_likelihood]: If fixed lengths are not used: if given, then this is taken a threshold on the likelihood of periodicity (see min_metagene_bf_mean description for more details). default: 0.5

Smoothing options

- [smoothing_fraction]. The fraction of the profile to use for smoothing within LOWESS. default: 0.2
- [smoothing_reweighting_iterations]. The number of reweighting iterations to use within LOWESS. Please see the statsmodels documentation for a detailed description of this parameter. default: 0
- [min_orf_length]. If this value is greater than 0, then ORFs whose length (in nucleotides) is less than this value will not be smoothed, and neither the Bayes factor estimates nor the chi-square p-value will be calcualted. default: 0
- [max_orf_length]. If this value is greater than 0, then ORFs whose length (in nucleotides) is greater than this value will not be smoothed, and neither the Bayes factor estimates nor the chi-square p-value will be calcualted. default: 0
- [min_signal]. ORFs for which the number of in-frame reads is less than this value will not be smoothed, and neither the Bayes factor estimates nor the chi-square p-value will be calculated. default: 5

Shared MCMC options

These affect the MCMC both for estimating metagene profile periodicity and ORF translation Bayes factors.

- [seed]. The random seed for the MCMC sampling. default: 8675309
- [chains]. The number of chains to use in the MCMC sampling. default: 2

Input files

The required input files are only those suggested by the configuration file keys.

- ribosomal_index.
- gtf.
- The STAR index
- The periodic models, taken as all .pkl files located in <models_base>/periodic.
- The nonperiodic models, taken as all .pkl files located in <models_base>/nonperiodic.

Output files

This script primarily creates the following files. (STAR also creates some temporary and log files.)

N.B. If the keep_riboseq_multimappers configuration option is given, then the "-unique" part will not be present in the output filenames.

- Trimmed and quality filtered reads

  - **trimmed and filtered reads**. A fastq.gz file containing the reads after removing adapters and low-quality reads. `<riboseq_data>/without-adapters/<sample-name>[.<note>].fastq.gz`
- Reads aligning to ribosomal sequences
  - **discarded reads**. A fastq.gz file containing reads which align to the ribosomal index. They are recorded but not used in later processing. `<riboseq_data>/with-rrna/<sample-name>[.<note>].fastq.gz`
- Reads not aligning to ribosomal sequences
  - **retained reads**. A fastq.gz file containing reads which do not align to the ribosomal index and are used in further processing. `<riboseq_data>/without-rrna/<sample-name>[.<note>].fastq.gz`
- Aligned reads
  - **sorted reads aligned to the genome**. A sorted bam file containing all alignments of reads to the genome. `<riboseq_data>/without-rrna-mapping/<sample-name>[.<` Additionally, `<riboseq_data>/without-rrna-mapping/<sample-name>[.<note>].bam`, which is a symlink to the `Aligned.sortedByCoord.out.bam` file.
  - **aligned reads which map uniquely to the genome**. A sorted bam file containing all alignments of reads to the genome with multimapping reads filtered out. `<riboseq_data>/without-rrna-mapping/<sample-name>[.<note>]-`
- Metagene profiles
  - **metagene profiles**. A gzipped csv file containing the metagene profiles for all read lengths which occur in the uniquely-aligning reads. It includes the metagene profile around both the annotated translation initiation site and translation termination site. `<riboseq_data>/metagene-profiles/<sample-name>[.<note>]-unique.metagene-profile.csv`
  - **periodicity estimations**. A gzipped csv file containing the Bayes factor estimates for all P-site offsets specified by the configuration, as well as information about the number of reads in the respective profile. `<riboseq_data>/metagene-profiles/<sample-name>[.<note>]-unique.metagene-periodicity`
  - **estimated P-site offsets**. A gzipped csv file containing the selected P-site offset for each read length. All read lengths are included, even if the estimates do not meet the criteria specified in the configuration file. (The filtering occurs later.) `<riboseq_data>/metagene-profiles/<sample-name>[.<note>]-unique.periodic-offsets.csv`
- ORF profiles
  - **unsmoothed ORF profiles**. A gzipped, sparse matrix market file containing the profiles for all ORFs. **N.B.** The matrix market format uses base-1 indices. `<riboseq_data>/orf-profiles/<sample-name>[.<note>]-unique.length-<`
  - **smoothed ORF profiles**. The smoothed profiles are not explicitly stored.

Indices are also created for the bam files. STAR creates parameter files in the location `riboseq_data`/without-rrna-mapping/`sample-name`_STARgenome.

Difference from paper

The fifth step of creating the base genome profile in the paper is "Everything except the 5' end of the remaining reads is removed." This profile is not explicitly constructed in the pipeline. The `<sample_name>-unique.bam` file already contains the necessary information.

```
create-orf-profiles test-chrI.fastq.gz c-elegans-test.yaml c-elegans-chrI --num-cpus 2 --tmp
```

Back to top

Predicting translated open reading frames

The entire profile creation and translation prediction process can be run automatically using the `predict-translated-orfs` script. It reads most of the required paths from a YAML configuration file. Additionally, it automatically creates some of the output paths.

The script accepts a `--overwrite` flag. Unless this is given, then steps for which the output file already exists will be skipped.

It also accepts a `--do-not-call` flag. If this flag is given, then the commands below will be printed but not executed.

Logging options can also be given to this script.

Parallel processing options can be given to this script.

Configuration file keys

The following keys are read from the configuration file. Keys with [`brackets`] are optional.

- `riboseq_data`. The base output location for all created files
- [`note`]. An optional string which will be added to all filenames. It should not contain spaces or any other special characters.
- [`models_base`]. The base path to the compiled models, the base path to the compiled models. The models specified in the paper are included with the source distribution and compiled/pickled as part of the installation process. They are installed in an operating system-specific location (in particular, `user_data_dir` from the appdirs package. This location is determined during installation and does not normally need to be changed. For development, they may be in some alternative location.

Reference genome options

These options should be exactly the same as those used in the configuration file used to create the reference indices.

- `fasta`. The path to the reference genome sequence
- `genome_base_path`. The path to the output directory for the transcript fasta and ORFs
- `genome_name`. A descriptive name to use for the created files

Bayes factor estimate options

- [`chi_square_only`]. If this flag is in the config file with any value, then only the Rp-chi pipeline will be performed; namely, the translation models will not be fit to the data, and the posterior distributions will not be estimated.
- [`translation_iterations`]. The number of iterations to use for each chain in the MCMC sampling. The first half of the iterations are discarded as burn-in samples. All of the remaining samples are used to estimate the posterior distributions. (That is, we do not use thinning.) default: 200

Selecting predicted ORFs options

- [`min_bf_mean`]. The minimum value for the estimated Bayes factor mean to "predict" that an ORF is translated. This value is used in conjunction with both min_bf_mean and min_bf_likelihood.

If `max_bf_var` is a positive value, then this is taken as a hard threshold on the estimated Bayes factor mean. ORFs must meet both the `min_bf_mean` and `max_bf_var` filters to be selected as "translated."

If `min_bf_likelihood` is given, then this is taken as the boundary value; that is, an ORF is selected as "translated" if:

```
[P(bf > min_bf_mean)] > min_bf_likelihood
```

If both `max_bf_var` and `min_bf_likelihood` are `None` (null in YAML), then this is taken as a hard threshold on the mean for selecting translated ORFs.

If both `max_bf_var` and `min_bf_likelihood` are given, then both filters will be applied and the result will be the intersection.

default: 5

- [`max_bf_var`]. The maximum value value for the estimated Bayes factor variance to "predict" that an ORF is translated. ORFs must meet both the `min_bf_mean` and `max_bf_var` filters to be predicted. See the description for `min_bf_mean` for more details. default: null (i.e., this filter is not used by default)

- [min_bf_likelihood]. The minimum probability of the BF exceeding min_bf_mean to select an ORF as translated. See the description for min_bf_mean for more details. default: 0.5
- [chisq_significance_level]. For the chi-square test, this value is first Bonferroni corrected based on the number of ORFs which pass the smoothing filters. It is then used as the significance threshold to select translated ORFs. default: 0.01

Shared MCMC options

These affect the MCMC both for estimating metagene profile periodicity and ORF translation Bayes factors.

- [seed]. The random seed for the MCMC sampling. default: 8675309
- [chains]. The number of chains to use in the MCMC sampling. default: 2

Input files

This script requires several files created during the previous steps in the pipeline, as well as a few external files.

- External files
  - **genome fasta file**. The genome fasta file. This is the same file used for prepare-rpbp-genome.
  - **orfs**. The ORFs (gzipped bed12+ file) created by prepare-rpbp-genome. It must be located at <genome_base_path>/transcript-index/<genome_name>.genomic-orfs.<or
  - **exons**. The ORF exons (gzipped bed6+ file) created by prepare-rpbp-genome. It must be located at <genome_base_path>/transcript-index/<genome_na
  - **models of translation**. Some compiled, pickled Stan model files must be located in the <models_base>/translated folder.
  - **models of lack of translation**. Some compiled, pickled Stan model files must be located in the <models_base>/untranslated folder.
- Metagene profiles
  - **metagene profiles**. A gzipped csv file containing the metagene profiles for all read lengths which occur in the uniquely-aligning reads. It includes the metagene profile around both the annotated translation initiation site and translation termination site. <riboseq_data>/metagene-profiles/<sample-name>[.<note>]-unique.metagene-profile.csv
  - **periodicity estimations**. A gzipped csv file containing the Bayes factor estimates for all P-site offsets specified by the configuration, as well as information about the number of reads in the respective profile. <riboseq_data>/metagene-profiles/<sample-name>[.<note>]-unique.metagene-periodicity
  - **estimated P-site offsets**. A gzipped csv file containing the selected P-site offset for each read length. All read lengths are included, even if the estimates do not meet the criteria

specified in the configuration file. (The filtering occurs later.) `<riboseq_data>/metagene-profiles/<sample-name>[.<note>]-unique.periodic-offsets.csv`

- ORF profiles
  - **unsmoothed ORF profiles**. A gzipped, sparse matrix market file containing the profiles for all ORFs. **N.B.** The matrix market format uses base-1 indices. `<riboseq_data>/orf-profiles/<sample-name>[.<note>]-unique.length-<`

Output files

If replicates are merged, then these files will be created for each condition. Otherwise, they will be created for each sample (or both if the appropriate options are given).

- Estimates
  - **the Bayes factor estimates**. A BED12+ file which contains the estimated values for all ORFs (which pass the thresholds mentioned above). The first 12 columns are valid BED12 entries that are simply copied from the `orfs` BED file. `<riboseq_data>/orf-predictions/<sample_name>[.<note>]-unique.length-<lengths>.offse`
- Predictions
  - **Rp-Bp predictions** are made using the methodology described in the paper.
    - **the ORFs**. A BED12+ file containing the ORFs in the final prediction set. `<riboseq_data>/orf-predictions/<sample_name>[.<note>]-unique.length-<`
    - **the DNA sequences**. A fasta file containing the DNA sequences of the predicted ORFs. The fasta header matches the 'id' column in the BED files. `<riboseq_data>/orf-predictions/<sample_name>[.<note>]-unique.leng`
    - **the protein sequences**. A fasta file containing the protein sequences of the predicted ORFs. The fasta header matches the 'id' column in the BED files. `<riboseq_data>/orf-predictions/<sample_name>[.<note>]-uniqu`
  - **Rp-chi predictions** are made using a simple chi-square test.
    - **the ORFs**. A BED12+ file containing the ORFs in the final prediction set. `<riboseq_data>/orf-predictions/<sample_name>[.<note>]-unique.length-<`
    - **the DNA sequences**. A fasta file containing the DNA sequences of the predicted ORFs. The fasta header matches the 'id' column in the BED files. `<riboseq_data>/orf-predictions/<sample_name>[.<note>]-unique.leng`
    - **the protein sequences**. A fasta file containing the protein sequences of the predicted ORFs. The fasta header matches the 'id' column in the BED files. `<riboseq_data>/orf-predictions/<sample_name>[.<note>]-uniqu`

Furthermore, there are "unfiltered" and "filtered" versions of the files. The "filtered" versions result from performing the filtering described in the paper (taking the longest predicted ORF for each stop codon, and then selecting the ORF with the highest expected Bayes factor among each group of overlapping ORFs). The "unfiltered" version contains all predictions.

```
predict-translated-orfs c-elegans-test.yaml c-elegans-chrI --num-cpus 2  --tmp /scratch/bmal
```

Back to top

Logging options

All of the driver scripts mentioned above, and many of the internal scripts as well, allow detailed specification of logging options on the command line. Interally, logging is handled using the standard python logging system. The following options are allowed.

The allowed logging levels are: `NOTSET`, `DEBUG`, `INFO`, `WARNING`, `ERROR`, `CRITICAL`. Most of the scripts output possibly-useful information at the `INFO` level; some scripts also output `DEBUG` messages, but as the name indicates, they are typically only useful for tracking down problems which do not cause actual errors.

- `--log-file`. This option specifies a file to which logging statements will be written (in addition to stdout and stderr, if specified). default: False
- `--log-stdout`. If this flag is present, then logging statements will be written to stdout (in addition to a file and stderr, if specified). default: False
- `--log-stderr`. If this flag is present, then logging statements will be written to stderr (in addition to a file and stdout, if specified). default: True
- `--logging-level`. If this value is specified, then it will be used for all logs. default: WARNING
- `--file-logging-level`. The logging level to be used for the log file, if specified. This option overrides `--logging-level`. default: NOTSET
- `--stdout-logging-level`. The logging level to be used for the stdout log, if specified. This option overrides `--logging-level`. default: NOTSET
- `--stderr-logging-level`. The logging level to be used for the stderr log, if specified. This option overrides `--logging-level`. default: NOTSET

Back to top

Parallel processing options

All of the scripts accept options for running code in parallel. Furthermore, the scripts are designed to seamlessly integrate with the SLURM scheduler. The following options are allowed. SLURM-specific options are specified by [`brackets`].

- `--num-cpus`. The number of CPUs to use. The definition of a "CPU" varies somewhat among the programs. For example, for STAR, these are actually threads. For many of the python scripts, this number is translated into the number of processes to spawn. None of the code parallelizes across machines, so the value should not be greater than the number of cores on the machine on which the programs are executed. When used with SLURM, this will be translated into an sbatch request like: "–ntasks 1 –cpus-per-task <num-cpus>". default: 1
- `--mem`. For STAR genome indexing, the amount of RAM to request. The rest of the programs do not use this value. When used with SLURM, this will be translated into an sbatch request like: "–mem=<mem>". default: 10G
- `--do-not-call`. If this flag is present, then the commands will not be executed (but will be printed). This can be useful to ensure paths in configuration files are correct.
- [`--use-slurm`]. If this flag is present, then the commands will be submitted to SLURM via sbatch. default: By default, each command is executed sequentially within the current terminal.
- [`--time`]. The amount of time to request. This will be translated into an sbatch request like: "–time <time>. default: 0-05:59
- [`--partitions`]. The partitions to request. This will be translated into an sbatch request like: "-p <partitions>". default: general (N.B. This value should be a comma-separated list with no spaces, for example: `--partitions general,long`)
- [`--no-output`]. If this flag is present, stdout will be redirected to /dev/null. This will be translated into an sbatch request like: "–output=/dev/null". default: If the flag is not present, then stdout will be directed to a log file with the job number. This corresponds to "–output=slurm-%J.out" in the sbatch call.
- [`--no-error`]. If this flag is present, stderr will be redirected to /dev/null. This will be translated into an sbatch request like: "–error=/dev/null". default: If the flag is not present, then stderr will be directed to a log file with the job number. This corresponds to "–error=slurm-%J.err" in the sbatch call.
- [`--stdout-file`]. If this is present and the –no-output flag is not given, then stdout will be directed to this file. This corresponds to "–output=`stdout-file`" in the sbatch call. default: slurm-%J.out
- [`--stderr-file`]. If this is present and the –no-error flag is not given, then stderr will be directed to this file. This corresponds to "–error=`stderr-file`" in the sbatch call. default: slurm-%J.err

- [`--mail-type`]. When to send an email notifcation of the job status. See official documentation for a description of the values. If a mail-user is not specified, this will revert to 'None'. Defaut: FAIL TIME_LIMIT
- [`--mail-user`]. To whom an email will be sent, in accordance with mail-type. default: None

```
# This will submit the prepare-genome script to SLURM as a single job. That job
# will request 10 CPUs and 100G of RAM.

prepare-rpbp-genome WBcel235.79.chrI.yaml --num-cpus 10 --mem 100G --overwrite --logging-lev

# This will submit each sample as a separate job to SLURM. Each submitted job will request 1
# and 100G of RAM. For example, if c-elegans-test.yaml specifies 5 samples in the riboseq_sa
# value, then 5 jobs will be submitted to SLURM, one for each sample.

# If the --merge-replicates flag is given, then all of the profiles are first created. Then,
# combined profiles are created in accordance with riboseq_biological_replicates from the co
# file. Finally, the last phase of the pipeline (predict-translated-orfs) is called.

# If any step fails, then later phases will not be started.

run-all-rpbp-instances c-elegans-test.yaml --tmp /scratch/bmalone/ --num-cpus 10 --mem 100G
```

Back to top