translation-prediction

April 18, 2016

0.1 Predicting translated open reading frames

This document describes the steps necessary to predict open reading frames (ORFs) which are translated using Rp-Bp and Rp-chi. It assumes the filtered genome profiles and reference sequence indices are already available.

0.1.1 Extracting ORF profiles

One of the main steps of the reference sequence index creation is the extraction of all ORFs in all annotated transcripts. So, somewhat in contrast to the description in the paper, the set of ORFs is already available. Thus, it only remains to extract the profile for each ORF from the filtered genome profile.

The `extract-orf-profiles` script performs this operation. Roughly, it quickly reads in the filtered reads using pysam, discards all of the read except for the 5' end and accounts for the P-site offsets with pandas, and finds the reads which map to each ORF using pybedtools. Custom python code then constructs the ORF profiles and saves them as a sparse matrix (in matrix market format).

Logging options can be given to this script.

TODO: This script contains logic for mapping between relative and genomic coordinates, given the splicing information! Extract this and make a tool!

Command line options

This script does not use the configuration file. It accepts the following command line options.

- `bam`. The `sample-name`-unique.bam file containing the (unshifted) filtered genome profile.
- `orfs`. The orfs.bed.gz file created during reference sequence index construction.
- `out`. The output sparse matrix file containing the profiles for all ORFs. N.B. The matrix market format uses base-1 indices. scipy.sparse automatically accounts for this, but other scripts may not.
- [`--lengths`]. A white-space delimited list of read lengths which will be used for creating the profiles. Presumably, these are lengths that have periodic metagene profiles. If no lengths are specified, then all lengths are used in profile construction.
- [`--offsets`]. The P-site offset to use for each read length specifed by `--lengths`. The number of offsets must match the number of lengths, and they are assumed to match. For example `--lengths 26 29 --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.
- [`--seqname-prefix`]. If the seqname column of the BED file for ORFs does not match the seqname to which the reads are aligned, then pybedtools will not find any reads for any of the ORFs. For example, if the ORF seqnames are like "2" and the alignment seqnames are like "chr2", then profile construction will fail (silently; the sparse matrix will just consist entirely of 0's). If this option is given, then the string will be prepended to the ORF seqnames. So, in the example, `--seqname-prefix chr` will ensure the profiles are constructed as expected.
- [`--tmp`]. By design, pybedtools writes BED files to a temporary location on disk. The option specifies this location. default: `/tmp`

- [`--num-procs`]. The profile extraction is embarassingly parallel about the ORFs, so it can easily be parallelized. This option gives the number of processes to use. It should not exceed the number of available processors. default: 2

- [`--num-groups`]. The script displays a progress bar which gives a rough indication of how many ORFs have been processed and how much time remains. For technical reasons, it is only updated after one of the parallel calls completes. This option controls the number of parallel calls (but NOT the number of processes). More calls means the progress bar is updated more frequently and the time estimates are more accurate, but incurs more overhead because of additional communication. default: 100

```
In [ ]: extract-orf-profiles /path/to/my/input/`sample-name`-unique.bam /path/to/my/input/orfs.bed.gz /
```

0.1.2 Estimating ORF Bayes factors

After the ORF profiles are constructed, the Bayes factor and chi-square p-values can be estimated. The `estimate-orf-bayes-factors` scripts makes these estimations. It uses PyStan to interface with Stan, which implements the No-U-Turn Sampler for Hamiltonian Markov chain Monte Carlo (MCMC). A (normal) posterior distribution over model marginal likelihoods is estimated from the MCMC results. This script reports these likelihood posteriors, as well as posteriors over model parameters such as the mixture model component means and variances.

By default, both the Bayes factor and chi-square p-values are reported. Thus, this script is used for both the Rp-Bp and Rp-chi pipelines.

Logging options can be given to this script.

Command line options

This script does not use the configuration file. It accepts the following command line options.

- `profiles`. The ORF profiles extracted from the filtered genome profile in the previous step.
- `regions`. The orfs.bed.gz file created during reference sequence index construction.
- `out`. The output file BED12+ file which contains the estimated values for all ORFs (which pass certain thresholds defined below). The first 12 columns are valid BED12 entries that are simply copied from the `regions` file.

TODO: The model inputs cannot be arbitrary; they should be specified somewhere.

- [`--translated-models`]. A white-space delimited list of models which somehow represent a translated ORF profile. Presumably, the mixture model described in the paper. This should be a list of paths to pickled StanModel objects. At least one model must be specified. For each ORF, the script reports only the estimates from the model with the highest mean marginal likelihood.

- [`--untranslated-models`]. A white-space delimited list of models which somehow represent an untranslated ORF profile. Presumably, the naive Bayes model described in the paper. This should be a list of paths to pickled StanModel objects. At least one model must be specified. For each ORF, the script reports only the estimates from the model with the highest mean marginal likelihood.

- [`--chi-square-only`]. If this flag is given, then only the chi-square test will be performed; the models will not be fit to the data, and the posterior distributions will not be estimated.

- [`--min-length`]. If this value is greater than 0, then ORFs whose length (in nucleotides) is less than this value will not be evaluated. Neither the Bayes factor estimates nor the chi-square p-value will be calcualted. default: 0

- [`--max-length`]. If this value is greater than 0, then ORFs whose length (in nucleotides) is greater than this value will not be evaluated. Neither the Bayes factor estimates nor the chi-square p-value will be calcualted. default: 0

- [`--min-signal`]. The Bayes' factor of ORFs for which the number of **in-frame** reads is less than this value will not be estimated. The chi-square p-value **will be** calculated for these ORFs, though. default: 0

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

TODO: The ORF types need to be documented somewhere!

- [`--orf-types`]. An optional white-space delimited list of orf types. If this list is given, then only ORFs annotated with the given types will be evaluated. The list of available types are:
- canonical
- canonical_extended
- canonical_truncated
- five_prime
- five_prime_overlap
- three_prime
- three_prime_overlap
- within
- noncoding
- suspect_overlap
- [`--num-procs`]. The Bayes facter estimations are embarassingly parallel about the ORFs, so it can easily be parallelized. This option gives the number of processes to use. It should not exceed the number of available processors. default: 1
- [`--num-groups`]. The script displays a progress bar which gives a rough indication of how many ORFs have been processed and how much time remains. For technical reasons, it is only updated after one of the parallel calls completes. This option controls the number of parallel calls (but NOT the number of processes). More calls means the progress bar is updated more frequently and the time estimates are more accurate, but incurs more overhead because of additional communication. default: 100

```
In [ ]: estimate-orf-bayes-factors /path/to/my/input/orf-profiles.mtx /path/to/my/input/orfs.bed.gz /pa
```

0.1.3 Selecting final prediction set

Depending on the type of analysis, the BED12+ file produced in the previous step may be appropriate for use. Nevertheless, the paper describes a process for joining predictions. In particular, for each stop codon, the longest ORF which passes the given prediction criteria (either BF mean and variance or chi-square p-value) are selected.

This procedure is implemented in the `select-final-prediction-set` script. Roughly, it first filters ORFs by the selected criteria. It then selects the longest unfiltered ORF for each stop codon. pybedtools is used to extract the DNA sequence for the long ORFs, and biopython is used to translate those sequences into protein sequences. Both the DNA and protein sequences are written to disk as fasta files.

Logging options can be given to this script.

Command line options

This script does not use the configuration file. It accepts the following command line options.

- `bayes_factors`. The BED12+ file with the Bayes factor estimates and chi-square p-values created in the previous step.

- `fasta`. The original **genome** fasta file
- `predicted_orfs`. A BED12+ file containing the ORFs in the final prediction set (the longest ORF for each stop codon which meets the filtering criteria).
- `predicted_dna_sequences`. A fasta file containing the DNA sequences of the predicted ORFs. The fasta header matches the 'id' column in the BED files.
- `predicted_protein_sequences`. A fasta file containing the protein sequences of the predicted ORFs. The fasta header matches the 'id' column in the BED files.
- [`--all`]. If this flag is given, then no filter is used. This flag is used to extract all of the DNA and protein sequences for all ORFs in the `bayes_factor` file.
- [`--minimum-profile-sum`]. The minimum sum **across all reading frames** to consider an ORF as predicted. This filter differs from `--min-signal` in the `estimate-orf-bayes-factors` because it includes the entire profile, while the `--min-signal` filter considers only the first reading frame. It is possible to select these filters such that they are incompatible, in a sense. For example, `--min-signal 10` and `--minimum-profile-sum 5` are incompatible in the sense that ORFs could meet the latter filter without meeting the first one. Thus, their Bayes' factor would not be estimated in the first place. So care must be taken when selecting these filters such that their combination is sensible. The chi-square p-value is calcualted for all ORFs which meet the length thresholds, so there is not a large chance for inconsistency when using Rp-chi.
- [`--use-chi-square`]. If this flag is present, then the chi-square p-values will be used to select translated ORFs. Otherwise, the Bayes factor estimates will be used.
- [`--min-bf-mean`]. The minimum value for the estimated Bayes factor mean to "predict" that an ORF is translated. ORFs must meet both the –min-bf-mean and –max-bf-var filters to be predicted. 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. default: 5
- [`--chisq-significance-level`]. If using the chi-square test, then this value is first Bonferroni corrected based on the number of ORFs which pass the `--minimum-profile-sum` filter. It is then used as the significance threshold to select translated ORFs. default: 0.01

```
In [ ]: select-final-prediction-set /path/to/my/input/orf-estimates.bed.gz /path/to/my/input/reference-s
```