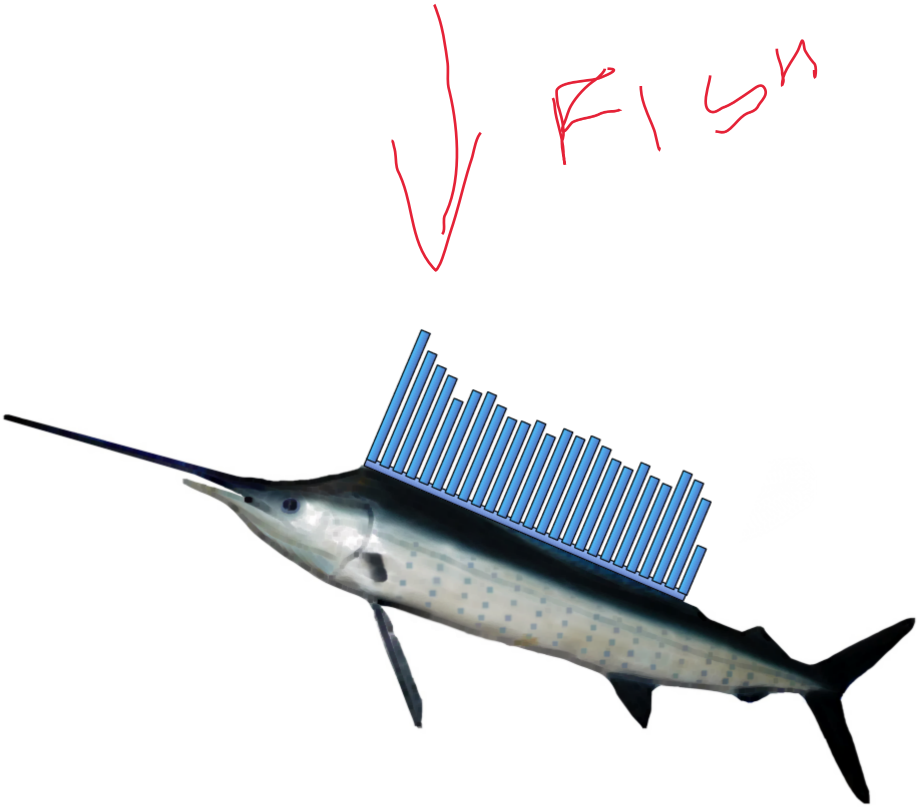

Sailfish: User Guide v0.6.3

Author: Rob Patro

Overview

Sailfish is a tool for transcript quantification from RNA-seq data. It requires a set of target transcripts (either from a reference or *de-novo* assembly) to quantify. All you need to run Sailfish is a fasta file containing your reference transcripts and a (set of) fasta/fastq file(s) containing your reads. Sailfish runs in two phases; indexing and quantification. The indexing step is independent of the reads, and only need to be run one for a particular set of reference transcripts and choice of k (the k-mer size). The quantification step, obviously, is specific to the set of RNA-seq reads and is thus run more frequently.

Indexing

To quantify the abundance, in a sample, of a set of target transcripts, you must first create a Sailfish index of this transcript set. The Sailfish index is actually just a collection of different files, kept together inside of a directory (the directory itself is referred to as the index), that allow Sailfish to efficiently access the information it needs about the target transcripts.

The generation of the Sailfish index is performed via the Sailfish `index` command. Like all user-level commands in Sailfish, `index` is a subcommand of the main Sailfish program. Thus, the index command is invoked as:

```
> sailfish index [options]
```

The list of options are as follows:

- **-v | –version** Print the version of Sailfish being used and exit
- **-h | –help** Print the help message describing the parameters
- **-t | –transcripts** A FASTA format file containing the transcripts on which the index will be built.
- **-m | –tgmap** Provide a transcript-to-gene map (currently unused)
- **-k | –kmerSize** The size of the k-mer on which the index is built. There is a tradeoff here between the distinctiveness of the k-mers and their robustness to errors. The shorter the k-mers, the more robust they will be to errors in the reads, but the longer the k-mers, the more distinct they will be. We generally recommend using a k-mer size of at least 20. Because of the way k-mers are encoded in Sailfish, the current maximum k-mer size is 32, but this may change in the future.
- **-o | –out** The directory in which the Sailfish index will be placed.

- **-p | –threads** The maximum number of concurrent threads to use when building the index.
- **-f | –force** If `-o` is provided with a directory that already exists, then force the re-building of the index, replacing the current contents of that directory.

To generate the Sailfish index for your reference set of transcripts, for example, you would run a command like the following:

```
> sailfish index -t <ref_transcripts> -o <out_dir> -k <kmer_len>
```

This will build a Sailfish index for k-mers of length <kmer_len> for the reference transcripts provided in the file <ref_transcripts> and place the index under the directory <out_dir>.

Quantification

Now that you have generated the Sailfish index (say that it's the directory <index_dir> — this corresponds to the <out_dir> argument provided in the previous step), you can quantify the transcript expression for a given set of reads. Just like the indexing is performed via the `index` sub-command of Sailfish, quantification is performed via the `quant` sub-command. The quantification command is invoked as follows:

```
> sailfish quant [options]
```

The list of options are as follows:

- **-v | –version** Print the version of Sailfish being used and exit
- **-h | –help** Print the help message describing the parameters
- **-i | –index** The path to the Sailfish index built on the set of target transcripts.
- **-l | –libtype** A string describing the library type of the provided reads. For a description of the different possible library strings and their meanings, see the library string section below.
- **-r | –unmated_reads** A list of one or more FASTA or FASTQ format files (or a named pipe providing reads in one of these formats) containing unpaired reads. This option should directly follow the `-l` option, and is only valid if the library format is of type single end (`SE`).

- **-1 | –mates1** A list of one or more FASTA or FASTQ format files (or a named pipe providing reads in one of these formats) containing the first mate for a set of reads. This option should directoy follow the `-l` option, and is only valid if the library format is of the paired-end type (`PE`).
- **-2 | –mates2** A list of one or more FASTA or FASTQ format files (or a named pipe providing reads in one of these formats) containing the second mate for a set of reads. This option should directoy follow the `-1` option, and is only valid if the library format is of the paired-end type (`PE`). This list should contain the same number of files (paired read-for-read) with the mates provided by the `-1` option.
- –**no_bias_correct** Normally, Sailfish outputs two quantification files in the requested output directory, `quant.sf` and `quant_bias_corrected.sf`. If this option is provided, bias-correction is not performed and the bias-corrected file is not produced.
- **-m | –min_abundance** Set to 0 the abundance of any transcripts with a computed K-mers Per Kilobase per Million mapped k-mers (KPKM) lower than the provided value.
- **-o | –out** The output directory where the quantification results (and other relevant files) are written.
- **-n | –iterations** The maximum number of iterations of the EM step to carry out. The optimization algorithm that computes the transcript estimates will terminate when either the convergence critera specified by the `-d` option (below) is met, or when this number of iterations has been performed.
- **-d | –delta** The maximum allowable delta between consecutive iterations of the optimization procedure. If the maximum relative change in any transcripts' abundance is less than this value between two consecutive iterations of the optimization, then the procedure will be considered to have converged and the optimization will terminate.
- **-p | –threads** The maximum number of threads to use when counting k-mers and computing transcript abundance.
- **-f | –force** By default, if the output folder provided to the `-o` option already exists, the k-mer counts recorded by the previous run will be used and only the quantification will be performed again. Passing in this option forces both a re-counting of the k-mers and a re-quantification of the target transcripts.
- **-a | –polya** If this flag is set, then polyA/polyT k-mers will not be counted.

So, a typical invocation of th the Sailfish `quant` command will look something like the following:

```
> sailfish quant -i <index_dir> -l "<libtype>" \
  {-r <unmated> | -2 <mates1> -2 <mates2>} -o <quant_dir>
```

Where <index_dir> is, as described above, the location of the Sailfish index, <libtype> is a string describing the format of the read library (see library string below) <unmated> is a list of files containing unmated reads, <mates{1,2}> are lists of files containing, respectively, the first and second mates of paired-end reads. Finally, <quant_dir> is the directory where the output should be written.

When the quantification step is finished, the directory <quant_dir> will contain a file named "quant.sf". This file contains the result of the Sailfish quantification step. This file contains a number of columns (which are listed in the last of the header lines beginning with '#'). Specifically, the columns are (1) Transcript ID, (2) Transcript Length, (3) Transcripts per Million (TPM), (4) Reads Per Kilobase per Million mapped reads (RPKM), (5) K-mers Per Kilobase per Million mapped k-mers (KPKM), (6) Estimated number of k-mers (an estimate of the number of k-mers drawn from this transcript given the transcript's relative abundance and length) and (7) Estimated number of reads (an estimate of the number of reads drawn from this transcript given the transcript's relative abnundance and length). The first two columns are self-explanatory, the next four are measures of transcript abundance and the final is a commonly used input for differential expression tools. The Transcripts per Million quantification number is computed as described in [1], and is meant as an estimate of the number of transcripts, per million observed transcripts, originating from each isoform. Its benefit over the K/RPKM measure is that it is independent of the mean expressed transcript length (i.e. if the mean expressed transcript length varies between samples, for example, this alone can affect differential analysis based on the K/RPKM.) The RPKM is a classic measure of relative transcript abundance, and is an estimate of the number of reads per kilobase of transcript (per million mapped reads) originating from each transcript. The KPKM should closely track the RPKM, but is defined for very short features which are larger than the chosen k-mer length but may be shorter than the read length. Typically, you should prefer the KPKM measure to the RPKM measure, since the k-mer is the most natural unit of coverage for Sailfish.

Library Format String

The library format string is given as a parameter to the `quant` step of Sailfish. Since Sailfish works with the reads directly and not alignments, the purpose of this string is to inform Sailfish of relevant information about the reads in the library. Not all of this information is *currently* used, but some of it is and other pieces of it may be in the future.

The library format string consists of 3 parts (one of which is sometimes optional), provided as key-value pairs. The relevant keys and possible value options are:

`(T|TYPE)=(PE|SE)`

This option specifies the “paired-end” status of the read library. If the reads are paired end, then this should be set to `PE`, and the library format string should be followed by the `-1` and `-2` options with the respective mate-pair reads. If the reads are unpaired, then this should be set to `SE` and the library format string shoujld be followed by the `-r` option and list of files containing unpaired reads.

`(O|ORIENTATION)=(>>|<>|><)`

This option secifies the relative orientation of reads within a pair. If the library consists of unpaired reads, then this key-value pair can and should be ignored. If the library consists of paired end reads, this key-value pair should be provided. Note, this denotes the *realtive orientation* of the reads, not their absolute directionality with respect to the reference. The options are meant to denote, visually, how the reads could be oriented.

The first option `>>` denotes that the mates are oriented in the same direction — e.g. if the 5’ end of mate 1 is upstream from the 3’ end, then the 5’ end of mate 2 is upstream from its 3’ end and vice-versa.

The second option `<>` denotes that the mates are oriented away from each other. This implies that start of mate1 is closer to start of mate 2 than the end of mate 2, etc.

The third option `><` is, perhaps, the most common relative orientation. It denotes that the mates are oriented toward each other, so that the start of mate 1 is farther from the start of mate 2 than it is from the end of mate 2 and vice-versa.

`(S|STRAND)=(AS|SA|S|A|U)`

This option specifies the strandedness of the reads. If the type is `SE` the only allowable options are `S`, `A`, and `U`, which denote, respectively, that the reads come from the sense strand, the antisense strand, or are of unknown strandedness (in which case both strands are tried and the one resulting in more matching k-mers is used).

If the type of the read library is `PE`, then any of the options are valid. The `S`, `A` and `U` options given in the above paragraph have the same meaning. The `AS` option specifies that mate 1 is from the antisense strand and mate2 is from the sense strand, while `SA` specifies that mate 1 is from the sense strand and mate 2 is from the antisense strand.

Because of the way argument parsing works, the library format string must be offset by quotations. An example format string specifying that the read library consists of unpaired reads with unknown orientation is:

```
-l "T=SE:S=U"
```

Alternatively, a format string specifying that the read library consists of paired-end reads, oriented toward each other where mate 1 is from the sense strand and mate 2 is from the antisense strand is:

```
-l "T=PE:O=><:S=SA"
```

Many, but not all, combinations of the three options (type, orientation and strandedness) are possible. Sailfish will perform a coarse-grained sainty check to ensure that the provided library string is not impossible (e.g. `T=SE:O=><:S=A`, which is not possible because unpaired reads can't have a relative orientation).

License

References

[1] Li, Bo, et al. "RNA-Seq gene expression estimation with read mapping uncertainty." Bioinformatics 26.4 (2010): 493-500.