FabFos fosmid processing pipeline – user guide

Connor Morgan-Lang

August 25, 2016

1 Introduction

FabFos is simply a repository for fosmid data - everything from the raw sequence reads to descriptors of fosmid well coordinates. These data are meant to be stored and accessed from a central location, and to keep things as easy as possible this occurs on a server. Data are added to FabFos using a single python script – FabFos.py. Following the directions below, outputs will be added to the central FabFos directory and sorted based on metadata provided in a pre-decided format as described below. The python script and documentation are stored on GitHub at https://github.com/hallamlab/FabFos.

2 The pipes

The pipeline schematic (Figure 1) shows the four main steps in the pipeline.
BWA MEM (v0.7.12-r1039) is used to align the fosmid reads to the backbone/contaminant database with default parameters. The output SAM file is converted to a BAM with samtools, and the reads that align to the database are removed by bam2fastq using the '–no-aligned' flag. Trimmomatic (v0.35) trims Illumina adapter sequences and low-quality sequences with the parameters 'ILLUMINACLIP:TruSeq3-PE.fa:2:3:10 LEADING:3 TRAILING:3 SLIDINGWINDOW:4:15 MINLEN:36'. The MEGAHIT (v1.0.6) command uses k-min=71, k-max=241, and k-step=10. The '–no-mercy' flag is used since these sequence datasets are typically high coverage (exceeding 100X). Assembly stats (N50, N90, number of contigs $>$ 27kbp, etc.) are determined using a combination of functions within FabFos.py and custom code written in C++.
If the end-sequence FASTA file is not provided, the pipeline ends here. Otherwise the pipeline continues to align the end-sequences to the sequences in contigs.fasta using blastn. blastn writes a tabular file with 'sseqid, slen, qseqid, pident, length, sstrand, sstart, send, bitscore' BLAST field codes (these mean subject sequence name, subject length, query sequence name, percent identity, alignment length, subject strand, subject alignment start, subject alignment end, and bitscore, respectively). The minimum percent identity of the end-sequences to a fosmid contig is 95%. These alignments are parsed and those with an alignment length $<$ 200 or a bitscore $<$ 500 are ignored. Clones (pairs of fosmid ends) are assigned to the contigs with 4 categories: single-pair, multi-pair, single-orphan, multi-orphan. These are explained in detail in Table 1. Additionally if mate-pairs (forward and reverse) are located on two separate fosmid contigs these are classified as 'twin-orphan multi-contig' and the contigs

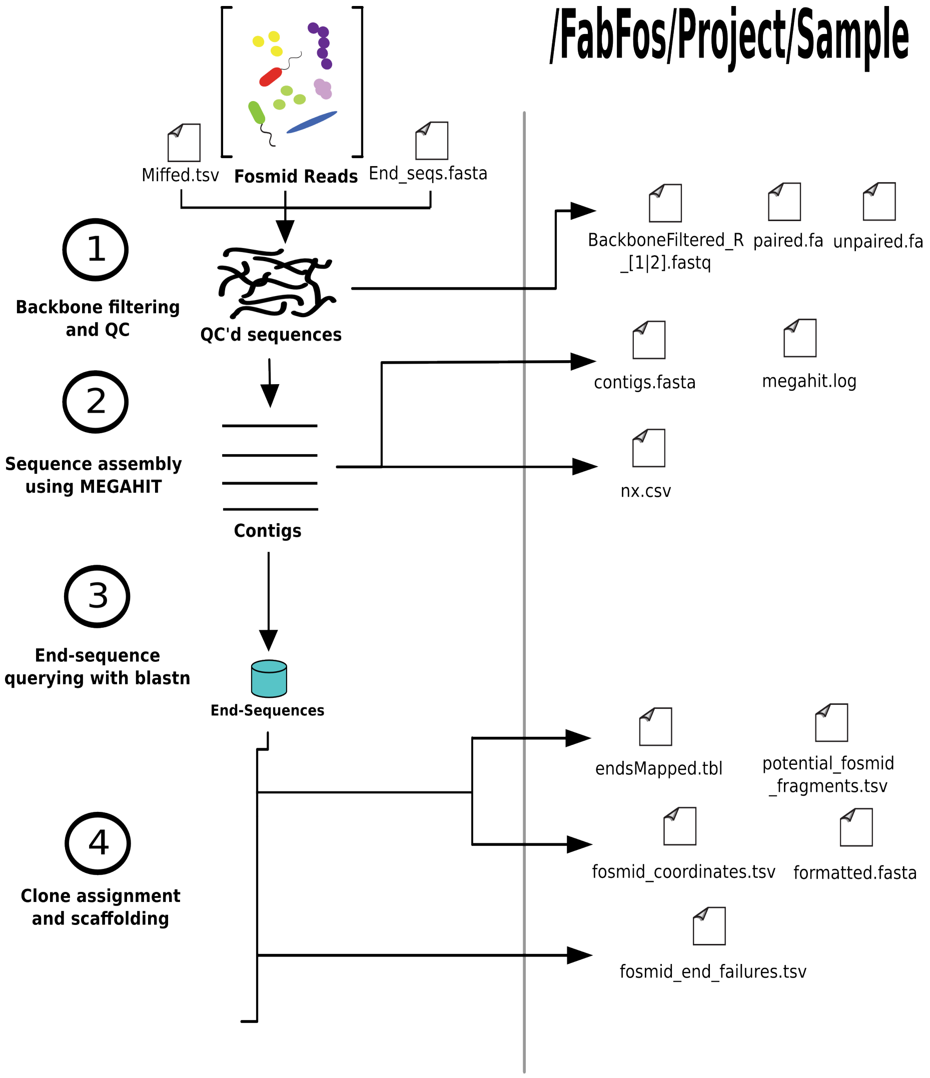

Figure 1: The pipeline showing inputs, outputs, and processing steps. The inputs are entered and sequences are QC'd to remove low-quality reads and reads that share homology with sequences in the backbone database. QC'd reads are then assembled using MEGAHIT. Finally, end-sequences are aligned to the contigs and the alignments are parsed to determine well location and potential scaffolds.

are scaffolded with 100 'N's gluing them together. 'No hit' is assigned to contigs that were not aligned to; these could represent misassemblies, contaminants, or short, internal fosmid sequences.

3 Running the pipeline

The pipeline requires the following inputs:

1. **Raw fosmid reads in FASTQ format**. The pipeline is meant to handle paired-end Illumina MiSeq (300bp) reads by default. If the forward and reverse reads are in separate files, these paths are given to FabFos.py with the '-r' and '-2' arguments, respectively. Alternatively, if the reads are contained in an interleaved file (i.e., the forward read is given, then the reverse read, and this is repeated through the file) then the flag '–interleaved' is to be used and the file is given only with the '-r' argument.

2. **Fosmid backbone and contaminating sequences in FASTA format**. These are sequences that will be used to filter the raw fosmid reads. There is no default file to be used, but one containing pCC1 backbone and the *E. coli* K-12 genome is found on the GitHub page and it can be modified for personal use.

3. **CSV file containing the minimum information for fosmid environmental data (MIFFED)**. A template table is provided on the GitHub page. The miimum information required are "Sample Name", "Project", "Human selector" (this may be synonymous with project lead), "Vector Name", and "Screen [in silico | functional]". miffed_template.csv is provided to guide users in entering the required and optional parameters to the pipeline.

The following inputs are optional, but recommended:

1. A FASTA formatted file containing the paired-end fosmid end sequences.

2. Number of threads to use, indicated by either '-T' or '–threads'. This will be used for assembly with MEGAHIT, and alignment with BLAST – if the end sequences are provided.

4 Outputs - and what to do with them

First off are the read files containing reads that are not homologous to sequences in the backbone – **_BackboneFiltered_R_[1|2].fastq.* The high-quality reads, according to their associated Phred scores, without Illumina adapter sequences are found in **_paired.fa* and **_unpaired.fa.* Most users will not need to use these sequences, but they remain just in case.

Secondly, **_contigs.fasta* (which is really just the suffix of the output file and the sample name is the prefix) contains the contigs assembled by MEGAHIT. The smallest possible contig in this file is 200bp. These can be used directly for many other bioinformatic pipelines e.g., Metapathways for annotation, minimus2 for extra further assembling, Maxbin2 for binning, etc.

Class	Definition
Single-pair	Two fosmid end sequences aligned to a contig – the mate-pairs (forward and reverse sequences) of a single fosmid end. They also aligned on opposite strands of the contig
Multi-pair	Multiple fosmid end sequences aligned to a contig but at least two of these reads are mate-pairs from the same clone
Single-orphan	A single end of one clone aligned
multi-orphan	A single end of multiple clones aligned
Twin-orphan multi-contig	mate-pairs (forward and reverse) are located on two contigs and scaffolded together with 100 'N's in between

Table 1: The five different clone assignment categories.

The *_endsMapped.tbl file is the raw output from the blastn command. potential_fosmid_fragments.tsv maps the orphaned reads to all the contigs they aligned to. *_fosmid_coordinates.tsv is a tabular file with the fields 'Fosmid', 'Contig', 'Evidence', 'Ends', and 'Sequence'. Fosmid is the clone that a contig has been assigned to for that row, unless no fosmid end reads aligned to it in which case Fosmid is None. Contig is the contig name, Evidence is one of the five potential classifications in Table 1. Ends are either F, R, or both depending on the direction of the fosmid end read(s) that aligned to the contig. Sequence is the trimmed sequence interior of the fosmid end alignemnt. For example, if both ends aligned to a contig (single-pair case) the Sequence will be all bases between and including the aligned positions. If an orphaned read aligns to a contig beginning at position 100, the first 100 bases are trimmed. *_formatted.fasta is the fasta version of *fosmid_coordinates.tsv* where the first four fields are in the header. Lastly, *_fosmid_end_failures.tsv contains the fosmid clone name in the first field and its failure status in the second. Fosmid ends can be flagged as failed for either not aligning to a contig, in which case resequencing of fosmids from that well may need to be done, or one or both fosmid end reads were less than 100bp.