VAMP Documentation

Release 0.9.0

Lance Parsons

May 23, 2014

Contents

1 Introduction **3**

2 Installation **5**
- 2.1 Install Prequisites . . . 5
- 2.2 Install VAMP . . . 5
- 2.3 Non-root users . . . 6

3 Usage **7**
- 3.1 Align Contigs to Reference . . . 7
- 3.2 Stitch Together Scaffold . . . 7
- 3.3 Genome Annotation and Comparison . . . 7

4 Commands **9**
- 4.1 compare_genomes.py . . . 9
- 4.2 fastq_to_fasta.py . . . 9
- 4.3 find_contig_deletions.py . . . 10
- 4.4 gff2gtf_simple.py . . . 10
- 4.5 maf_net.py . . . 11
- 4.6 makePairedOutput2EQUALfiles_vamp.pl . . . 12
- 4.7 makePairedOutput2UNEQUALfiles_vamp.pl . . . 13
- 4.8 TQSfastq_vamp.py . . . 13
- 4.9 translate_cds.py . . . 13

5 Modules **15**
- 5.1 `vamp.utils` . . . 15
- 5.2 `seq_utils.convert_coordinates` . . . 19
- 5.3 `seq_utils.fasta_from_gff` . . . 19
- 5.4 `seq_utils.summarize_alignments` . . . 19
- 5.5 `seq_utils.utils` . . . 20

6 Indices and tables **23**

Python Module Index **25**

Contents:

CHAPTER 1

Introduction

The Virus AsseMbly Pipeline (VAMP) is a set of tools designed to assist with reference guided assembly of viral genomes from paired-end Illumina sequence data.

The pipeline portion of VAMP has been replaced by VirGA (paper in progress, Moriah Szpara and Lance Parsons).

The main tools used by VirGA are *maf_net.py* and *compare_genomes.py*. There are a number of *other tools* which may be useful.

CHAPTER 2

Installation

2.1 Install Prequisites

1. **Bedtools**
    (a) Bedtools Installation Instructions
2. **Cython**
    (a) `pip install cython`

2.2 Install VAMP

1. Download VAMP from https://bitbucket.org/szparalab/vamp/downloads
2. **Unarchive into directory**
    (a) `tar xzvf vamp-x.x.tar.gz.`
3. **Install VAMP (plus python dependencies):**
    (a) `cd vamp-x.x`
    (b) `python setup.py install`
4. **(optional) Build documentation**
    (a) `cd docs`
    (b) `make html`

2.2.1 Notes

- **OSX**
    - XCode and Command Line Utilities must be installed prior to installing many of the required tools for VAMP. See the MacPorts XCode installtion instructions for more information.

2.2.2 Python Dependencies

By default, VAMP's setup.py installs the required python dependencies listed below:

- Cython
- BioPython
- bx-python
- pybedtools
- argparse (only if Python version is < 2.7)

2.3 Non-root users

- If you are not root or just want to install this locally, one option is to use the `--user` parameter when installing. e.g.:

```
pip install --user cython
python setup.py install --user
```

CHAPTER 3

Usage

3.1 Align Contigs to Reference

The first step is to align contigs to a reference genome and output the result in a MAF formatted file. There are many options for alignment tools, however, we have had success with Mugsy, a very fast multiple whole genome alignment tool.

3.2 Stitch Together Scaffold

Once you have aligned the contigs to the reference, the next step is to stitch together the various alignment blocks into a scaffold. The *maf_net.py* utility does this by reassembling the reference sequence from the MAF blocks and using the highest scoring block for each location in the genome to assemble a scaffold genome.

3.3 Genome Annotation and Comparison

Once a draft of a genome has been completed, it can be useful to migrate annotations from an annotated reference to the new genome. In addition, this step generates a summary of the changes at the nucleic acid as well as amino acid level.

Run *compare_genomes.py* to migrate annotations and generate a list of differences between two species. The script requires an aligned fasta file (typically use the one generated from the previous scaffold stitching step) and a GFF file of features (genes, exons, etc.) to migrate.

The coding sequences can be checked by translating them to protein sequences using *translate_cds.py*. Translation errors such as missing start or stop codons, extra stop codons, etc. will be printed to STDERR.

CHAPTER 4

Commands

4.1 compare_genomes.py

Compares genomes using an aligned fasta file and migrates annotations from a reference to the other sequences in the alignment

Usage:

```
compare_genomes.py [-r REFERENCE] [--align_format FORMAT] [-o PREFIX]
                       [--gff_feature_types GFF_FEATURE_TYPES]
                       [--gff_attributes GFF_ATTRIBUTES] [-v] [--version]
                       [-h]
                       aligned_fasta gene_gff
```

Required Arguments:

```
aligned_fasta          An aligned fasta file
gene_gff               An gff file with features to be migrated
-r REFERENCE, --reference REFERENCE
                       Sequence id of reference sequence in aligned fasta
                       file
```

Optional Arguments:

```
--align_format FORMAT
                       Alignment format (default: fasta)
-o PREFIX, --output PREFIX
                       Output prefix (default: compare_genomes_output/)
--gff_feature_types GFF_FEATURE_TYPES
                       Comma separated list of gff feature types toparse
                       (default: CDS,exon,gene,mRNA,stem_loop)
--gff_attributes GFF_ATTRIBUTES
                       Comma separated list of feature attributes tocarry
                       over (default: ID,Parent,Note,gene,function,product)
-v, --verbose          verbose output
--version              show program's version number and exit
-h, --help             show this help message and exit
```

4.2 fastq_to_fasta.py

Convert a FASTQ file to a FASTA file

Usage:

```
fastq_to_fasta.py [-h] [-w WRAP] [-v] [--version] fastq_file fasta_file
```

Required Arguments:

```
fastq_file
fasta_file
```

Optional Arguments:

```
-h, --help            show this help message and exit
-w WRAP, --wrap WRAP  Maximum length of lines, 0 means do not wrap (default:
                      0)
-v, --verbose         verbose output
--version             show program's version number and exit
```

4.3 find_contig_deletions.py

Find contigs with deletions from the contig composition file output from *compare_genomes.py*

Usage:

```
find_contig_deletions.py [-h] [-o OUTPUT_DIR] [-q] [-v] [--version]
                         contig_composition aligned_fasta contigs_fasta
```

Find contigs with deletions from the contig composition file output from compare_genomes.py

Required Arguments:

```
contig_composition    Contig composition file output from compare_genomes.py
aligned_fasta         Aligned FASTA file
contigs_fasta         Contigs FASTA file
```

Optional Arguments:

```
-h, --help            show this help message and exit
-o OUTPUT_DIR, --output_dir OUTPUT_DIR
                      Directory to store output files, default is
                      aligned_fasta directory
-q, --quiet           Quiet, replace all deletions found, no prompts
-v, --verbose         verbose output
--version             show program's version number and exit
```

4.4 gff2gtf_simple.py

Simple conversion of GFF files to GTF files.

Usage:

```
gff2gtf_simple.py [-h] [-v] [--version] gff_file
```

Required Arguments:

```
gff_file       GFF file to convert
```

Optional Arguments:

```
-h, --help     show this help message and exit
-v, --verbose  verbose output
--version      show program's version number and exit
```

4.5 maf_net.py

Output an aligned fasta file by stitching together a specified reference sequence in the MAF file and using the highest scoring block for each section.

Usage:

```
maf_net.py [-r REFERENCE] [-c CHROMOSOME] [-s SPECIES] [-o OUTPUT_DIR]
           [--consensus_sequence] [--reference_fasta REFERENCE_FASTA]
           [-v] [--version] [-h]
           maf_file
```

Required Arguments:

```
maf_file              MAF file to stitch together
-r REFERENCE, --reference REFERENCE
                      Reference species (e.g. scerevisiae)
-c CHROMOSOME, --chromosome CHROMOSOME
                      Sequence ID of the chromosome for which to generate
                      the alignment net (e.g. chrI)
-s SPECIES, --species SPECIES
                      List of species to include, comma separated (e.g.
                      scerevisiae,sbayanus)
```

Optional Arguments:

```
-o OUTPUT_DIR, --output_dir OUTPUT_DIR
                      Directory to store output file, default is maf file
                      directory
--consensus_sequence  Output "consensus sequence" for each species in files
                      named [species].[chromosome].consensus.fasta
--reference_fasta REFERENCE_FASTA
                      Check MAF file against this fasta (for
                      troubleshooting, debugging)
-v, --verbose         verbose output
--version             show program's version number and exit
-h, --help            show this help message and exit
```

Output:

- **Aligned Fasta File**: `BASENAME.net.afa`

  This file contains an aligned fasta file created by stitching together MAF blocks based on the reference sequence. Where two blocks overlap, the higher scoring block is used.

Optional Output (one per species):

- **Consensus Sequence**: `SPECIES.consensus.fasta`

  A FASTA file containing the consensus sequence for this species. N's in the sequence represent sections where no contigs mapped to a section of the reference (i.e. potential gaps in the scaffold).

- **Consensus Contig Composition GFF**: `SPECIES.consensus_contig_composition.gff`

GFF formatted file describes intervals in the SPECIES genome. The attributes contain information about the contigs used to determine the sequence in this interval. The attributes are:

- src_seq
- src_seq_start
- src_seq_end
- src_strand
- src_size
- maf_block
- block_start
- block_end
- ref_src
- ref_start
- ref_end
- ref_strand

- **Consensus Contig Composition Summary**: `SPECIES.consensus_contig_composition_summary.txt`

  Tab delimited file with the following columns that describes intervals in the SPECIES genome and the contigs that were used for the sequence.

  - *seq* - sequence id of the interval in the SPECIES genome
  - *start* - start position of the interval
  - *end* - end position of the interval
  - *contig* - contig id that was used to "build" this interval. If *None*, that means no contig was found for the analogous region in the reference.
  - *contig_start* - the start position of the contig that aligned to this start interval
  - *contig_end* - the end position of the contig that aligned to the end position of this interval
  - *contig_strand* - the direction that the contig aligned to the reference (if '-', the reverse complement of the contig aligned to the reference in this interval)
  - *contig_size* - the full size of the contig (including those bases that did not aligned to this interval)

4.6 makePairedOutput2EQUALfiles_vamp.pl

Modified versions of scripts provided by SSAKE. They are used to prepare two separate paired end fastq files for use by SSAKE. The modifications made were to accommodate new Illumina style sequence identifiers introduced with CASAVA 1.8.:

```
Usage: makePairedOutput2EQUALfiles_vamp.pl <fasta file 1> <fasta file 2> <library insert size>
       --- ** Both files must have the same number of records & arranged in the same order
```

4.7 makePairedOutput2UNEQUALfiles_vamp.pl

See *makePairedOutput2EQUALfiles_vamp.pl*:

```
Usage: makePairedOutput2UNEQUALfiles_vamp.pl <fasta file 1> <fasta file 2> <library insert size>
          --- files could have different # of records & arranged in different order but template ids
```

4.8 TQSfastq_vamp.py

Preforms quality trimming as per the original SSAKE script. It was modified to accommodate larger, zipped fastq files.

Usage:

```
TQSfastq_vamp.py [options]
```

Optional Arguments:

```
-h, --help            show this help message and exit
-f FASTQFILE, --fastq file=FASTQFILE
                      Sanger encoded fastq file - PHRED quality scores,
                      ASCII+33
-t THRESHOLD, --Phred quality threshold=THRESHOLD
                      Base intensity threshold value (Phred quality scores 0
                      to 40, default=10)
-c CONSEC, --consec=CONSEC
                      Minimum number of consecutive bases passing threshold
                      values (default=20)
-v, --verbose         Runs in Verbose mode.
-q, --qualities       Outputs Qualities to FASTQ file (default is FASTA)
-z, --zip             Compress output with gzip
-o OUTPUT_BASE, --output=OUTPUT_BASE
                      Output filename base
```

4.9 translate_cds.py

Extracts the coding sequences (CDS) regions from a fasta reference and gff file and translates them into amino acid sequences, output in FASTA format to STDOUT

Usage:

```
translate_cds.py [--notrans] [-i IDATTR] [-t FEATURETYPE]
                 [--table TABLE] [-v] [--version] [-h]
                 gff_file fasta_file
```

Required Arguments:

```
gff_file              GFF file containing CDS records to be translated
fasta_file            FASTA file containing the nucleotide sequences
                      referenced in the GFF file
```

Optional Arguments:

```
--notrans             Do not translate to amino acid sequence, output DNA
-i IDATTR, --idattr IDATTR
                      GFF attribute to use as gene ID. Features with the
                      same ID will be considered parts of the same gene. The
                      default "gene_id" is suitable for GTF files.
-t FEATURETYPE, --featuretype FEATURETYPE
                      GFF feature type(s) (3rd column) to be used. Specify
                      the option multiple times for multiple feature types.
                      The default is "CDS" for GFF files and "CDS" and
                      "stop_codon" for GTF files.
--table TABLE         NCBI Translation table to use when translating DNA
                      (see http://www.ncbi.nlm.nih.gov/Taxonomy/Utils/wprint
                      gc.cgi). Default: 1.
-v, --verbose         verbose output
--version             show program's version number and exit
-h, --help            show this help message and exit
```

CHAPTER 5

Modules

5.1 vamp.utils

Utilities for working with multiple sequence alignments and MAF objects

class vamp.utils.**ContigComposition**

Represents the composition of one interval by another interval.

Association of two genomic intervals, used to represent the composition of one interval by another.

seq str
Sequence id of the interval being described

start str
The start position of the interval being described (1-based)

end str
The end position of the interval being described (1-based)

contig str
The sequence id of the second interval

contig_start str
The start position of the second interval (1-based)

contig_end str
The end position of the second interval (1-based)

strand str
The strand ('+' or '-') of the interval being described

contig_size str
The complete length of the sequence of the second interval.

static tab_headings()
Static method that retuns a tab delimited string of header names

Returns A tab delimited string representing the headers in the order used by the to_tab() method.

Return type string

to_tab()
Return a tab delimited string of the ContigComposition

Returns A tab delimited string representing the ContigComposition.

Return type string

vamp.utils.**find_deletions**(*contig_composition_list*, *verbose=False*)
Find contigs with deletions.

From a contig composition list, find contigs that have deleted sections. When a contig has deleted sections, the pieces of the contig may be replaced by a contiguous section. This function returns tuples containing the indices of the contigs pieces to be replaced along with a replacement `ContigComposition` object consisting of the contiguous section.

e.g.:

```
([2, 3],
 {'seq': 'chr', 'start': 3, 'end': 5, 'contig': 'contig1',
  'contig_start': 5, 'contig_end': 10, 'strand': '+',
  'contig_size': 20})
```

indicates that we may wish to replace `contig_composition_list[2:3]` with the new `ContigComposition` specified.

Parameters

- **contig_composition_list** (*list*) – A list of `ContigComposition` objects.
- **verbose** (*bool, optional*) – If true, output additional debug info (default is False).

Returns A list of tuples with the indices of `ContigComposition` objects in the list to be replaced along with replacement `ContigComposition` objects.

Return type list

vamp.utils.**get_block_by_label**(*maf_filename*, *label*)
Return the MAF block with the specified label

Parameters

- **maf_filename** (*string*) – The name of the MAF file.
- **label** (*string*) – The label of the block in the MAF file to search for.

Returns The first block found in the MAF file that has the given label

Return type block

vamp.utils.**get_sequence_length_from_maf**(*maf_file*, *reference_species*, *sequence_id*)
Return length of the reference_species.sequence_id

Parameters

- **maf_filename** – The filename of the MAF file.
- **reference_species** – The name species used as the reference.
- **sequence_id** – The sequence_id used as the reference. The format of sequence names in the MAF file is assumed to be 'species.sequence_id' (e.g. 'scerevisiae.chrI')

Returns The length of the specified sequence in the first component containing that sequence in the MAF file, or None if no matching componenets were found in the MAF file.

Return type integer

vamp.utils.**get_sequence_net_alignment**(*maf_filename*, *reference_species*, *sequence_id*, *species*, *verbose=False*)

Return the alignment created by stitching MAF blocks together

Stitches MAF blocks together along an entire reference sequence (including gaps). For regions covered by more than one block, the highest scoring block is used.

Parameters

- **maf_filename** – The filename of the MAF file.
- **reference_species** – The name species used as the reference.
- **sequence_id** – The sequence_id used as the reference. The format of sequence names in the MAF file is assumed to be 'species.sequence_id' (e.g. 'scerevisiae.chrI')
- **species** – A list of the species names to be returned
- **verbose** (*bool, optional*) – If True, print debug information (default: False)

Returns A tuple containing a Bio.Align.MutlipleSeqAlignment object and a list of intervals. The multiple sequence alignments contains each the alignment of each species from the MAF file created by stitching blocks together based on the specified reference sequence The list of intervals is relative to the alignment that indicate the MAF block, block start, and block end of the source of that piece of the alignment.

Return type tuple

vamp.utils.**get_vamp_home**()

Return the directory where the VAMP module is installed

vamp.utils.**read_contig_composition_summary**(*filename*)

Generator that reads a contig composition summary file and returns attributes.

Parameters **filename** (*string*) – The name of contig composition summary file as output by compare_genomes.py.

Yields *ContigComposition* – A `ContigComposition` object for each line in the contig composition summary file.

vamp.utils.**replace_alignment_with_block**(*alignment*, *block*, *reference_species*, *sequence_id*, *verbose=False*)

Update the multiple sequence alignment with the specified MAF block

Use the MAF block alignment to replace the appropriate section of the given multiple sequence alignment by using the specified reference species and sequence as guide

Parameters

- **block** (*maf block*) – MAF block
- **reference_species** (*str*) – The name species used as the reference.
- **sequence_id** (*str*) – The sequence_id used as the reference. The format of sequence names in the MAF file is assumed to be 'species.sequence_id' (e.g. 'scerevisiae.chrI')
- **verbose** (*bool, optional*) – If True, print debug information (default: False)

Returns The updated alignment and a Pybedtools interval of the section of the alignment that was replaced. The interval contains the following attributes: *maf_block; block_start; block_end* which indicate the MAF block label and start and end position on the block used in the replacement

Return type tuple

vamp.utils.**subtract_intervals**(*interval1*, *interval2*)
Subtract two pybedtools intervals, return list of resulting intervals

Parameters

- **interval1** – A pybedtools interval
- **interval2** – A pybedtools interval to subtract from interval1

Returns A list of pybedtools intervals that contain the region(s) of interval1 that are not overlapped by interval2

Return type list

vamp.utils.**summarize_contig_composition**(*interval_list*, *src_tag*, *start_tag*, *end_tag*, *strand_tag*, *source_size_tag*)
Summarize the contig composition of a stitched MAF file.

Parameters

- **interval_list** (*list*) – A list of Pybedtools interval objects
- **src_tag** (*string*) – The attribute containing the contig name
- **start_tag** (*string*) – The attribute containing the start position in the contig
- **end_tag** (*string*) – The attribute containing the end position in the contig
- **strand_tag** (*string*) – The attribute containing the strand
- **source_size_tag** (*string*) – The attribute containing the contig size

Returns A list of dictionaries with the following keys: (seq, start, end, contig, contig_start, contig_end, strand, contig_size)

Return type list

vamp.utils.**update_contig_composition_summary**(*contig_composition_summary*, *replacements*)
Update list of `ContigComposition` objects with replacements.

Replacements are a list of tuples containing a list of indices of contigs to be replaced along with replacements. The replacements must be non-overlapping and sorted.

Parameters

- **contig_compostion_summary** (*list*) – List of dictionaries as returned by `summarize_contig_composition()`
- **replacements** (*list*) – A list of `ContigCompostion` objects

Retuns: list: A updated contig composition summary

vamp.utils.**update_sequence_with_replacements**(*seq*, *replacements*, *replacement_seq_dict*)
Update Seq object with replacements.

The replacements specified by `ContigComposition` objects and must be non-overlapping and sorted.

Parameters

- **seq** (*Bio.Seq*) – The sequence object to be updated.
- **replacements** (*list*) – A list `ContigComposition` objects
- **replacement_seq_dict** (*dict*) – A dictionary to the replacement sequences.

Returns Bio.Seq: An updated sequence object with replacements made

vamp.utils.**verify_maf_fasta**(*maf_filename*, *reference_species*, *fasta_filename*, *verbose=False*)
Verify the consistency between the sequence in a MAF and a FASTA file

Checks all compenents in all blocks of the MAF file for the specified species and checks that the sequence matches that in the FASTA file.

Parameters

- **maf_filename** (*string*) – The name of the MAF file.
- **reference_species** (*string*) – The species to select from the MAF file.
- **fasta_filename** (*string*) – The name of the FASTA file to check against.
- **verbose** (*bool, optional*) – If true, output additional debugging info (default is False).

Returns Prints to STDOUT if there is a mismatch.

Return type None

5.2 seq_utils.convert_coordinates

Convert coordinates from GFF or BED file using multi-fasta alignments

seq_utils.convert_coordinates.**find_aligned_position**(*gap_positions*, *pos*)
Update position by adding preceeding gaps

Parameters

- **gap_positions** (*list*) – list of gaps (must include start and end methods to return the start and end of a gap, typically they are re.MatchObjects)
- **pos** (*int*) – the position to adjust by adding preceding gaps

Returns The new position, accounting for preceeding gaps

Return type int

5.3 seq_utils.fasta_from_gff

Extract fasta sequences from regions defined in GFF/BED file and output fasta to stdout

5.4 seq_utils.summarize_alignments

Summarize the differences between sequences in an aligned FASTA file.

This script will output summarize the differences between sequences in an aligned FASTA file.

Usage:

```
summarize_alignments.py aligned_fasta reference_sequence [-h,--help]
    [-v,--verbose] [--version]
```

seq_utils.summarize_alignments.**main**()
Runs summary_of_alignment function on input files from the command line.

seq_utils.summarize_alignments.**mismatch_string**(*mismatches*)
Generate a string from a list of mismatches.

Parameters **mismatches** (*list*) – A list of mismatches. A mismatch is a dictionary with a position (pos), reference genotype (ref), and alternate genotype (alt).

Returns A comma separated string of the mismatches

Return type string

`seq_utils.summarize_alignments.`**`parse_event`**(*event*, *reference_sequence*, *alternate_sequence*)

Parse an event (sequence of differences) for VCF output.

Parse a simple event with reference_position, reference_base, and new_base and determine the type and add padding if necessary (for VCF compatibility)

Parameters

- **event** (*dictonary*) – An event has at least a position (pos), reference genotype (ref), and aternate genotype (alt). May also have a flag indicating if it is a snp (snp).
- **reference_sequence** (*str*) – The complete reference sequence
- **alternate_sequence** (*str*) – The complete alternate sequence

Returns An event with additional padding to the start of the variant and an added type attribute, for VCF compatibility

Return type dictonary

`seq_utils.summarize_alignments.`**`summary_of_alignment`**(*alignment*, *reference_sequence_id*)

Summarizes changes in given alignment

Parameters

- **alignment** (*Bio.AlignIO object*) – Alignment object
- **reference_index** (*int*) – index of the reference sequence in alignment (default is 1)

Returns

A dctionary with a key for each non-reference sequence in the alignment

Each entry is another dictionary with the following keys:

- *match_count*: The number of matching bases
- *mismatch_count*: The number of mismatching bases, including indels
- *mismatches*: list of mismatches by base: *RefBase(RefPos)NewBase*
- ***contiguous_change_count*: the number of contiguous change** "events"

Return type dictionary

5.5 seq_utils.utils

Utility classes and methods for working with sequence data

`seq_utils.utils.`**`convert_interval_gapped_to_nongapped`**(*seq*, *start*, *end*)

Take position with gaps and return position without gaps

Uses 0-based positions

Parameters

- **seq** (*str*) – sequence string (with gaps included)

- **start** (*int*) – starting position of interval (including gaps)
- **end** (*int*) – ending postion of interval (including gaps)

Returns (start, end) the start and end positions after removing gaps in the sequence

Return type tuple

seq_utils.utils.**convert_interval_nongapped_to_gapped**(*seq*, *start*, *end*, *include_end_gaps=False*)

Take position without gaps and return position with gaps

Uses 0-based positions

Parameters

- **seq** (*str*) – sequence string (with gaps added)
- **start** (*int*) – starting position of interval (excluding gaps)
- **end** (*int*) – ending postion of interval (excluding gaps)
- **include_end_gaps** (*bool, optional*) – if true, include gap positions that directly follow the end positions in the new interval, default is False and such end positions are not included

Returns (start, end) the start and end positions after accouting for gaps in the sequence

Return type tuple

CHAPTER 6

Indices and tables

- *genindex*
- *modindex*
- *search*

S

seq_utils.convert_coordinates, 19
seq_utils.fasta_from_gff, 19
seq_utils.summarize_alignments, 19
seq_utils.utils, 20

V

vamp.utils, 15