
EM2Tools

Release 0.1.0

Frédéric PLEWNIAK

Aug 18, 2020

CONTENTS:

1	Getting started	1
1.1	Requirements	1
1.2	Installation	1
2	The EM2Tools API reference	3
2.1	The “seq_record” module	3
2.2	The “seq” module	5
2.3	The “seq_feature” module	6
2.4	The “seq_utils” module	10
2.5	The “argparse_em2” module	12
3	Indices and tables	13
	Python Module Index	15

GETTING STARTED

1.1 Requirements

The EM2Tools package requires at least Python 3.6 with Biopython, pandas and gffpandas packages.

1.2 Installation

1.2.1 Linux

1. Download and unzip the master branch zip file:

```
wget https://github.com/fplewniak/EM2Tools/archive/master.zip  
unzip master.zip
```

2. Install the EM2Tools Python package and the required Python packages with pip:

```
pip install EM2Tools-master/.
```


THE EM2TOOLS API REFERENCE

2.1 The “seq_record” module

Extension module to the Biopython Bio.SeqRecord module

class `em2lib.seq_record.SeqRecordEM2` (*seq*, ***kwargs*)

Extension to Biopython SeqRecord class

add_feature (***kwargs*)

Adds a feature to the current record according to arguments passed as ***kwargs*.

Parameters *kwargs* – keyword arguments to pass to SeqFeatureEM2 class

overlap (*start*, *end=None*, *strand=0*)

Retrieves features that overlap a given position range.

Parameters

- **strand** – strand specification of features to be returned. If strand is 0, then features on both strands are returned. If feature.strand is 0, then all strands will match.
- **start** – start of range
- **end** – end of range, if None, then end=start

Returns a list of overlapping features

feature_after (*position*, *strand=0*, *nearest=False*)

Retrieves the features immediately after (but not overlapping) the specified position, on one strand or both. If nearest is True, then only the nearest ones are returned.

Parameters

- **nearest** – if True, only the nearest features are returned. This only makes sense when strand is 0
- **strand** – strand specification of features to be returned. If strand is 0, then features on both strands are returned
- **position** – the position

Returns a list of features after the specified position

feature_before (*position*, *strand=0*, *nearest=False*)

Retrieves the features immediately before (but not overlapping) the specified position, on one strand or both. If nearest is True, then only the nearest ones are returned.

Parameters

- **nearest** – if True, only the nearest features are returned. This only makes sense when strand is 0
- **strand** – strand specification of features to be returned. If strand is 0, then features on both strands are returned
- **position** – the position

Returns a list of features before the specified position

surrounding_features (*position, strand=0, nearest=False*)

Retrieves all the features around a given position but not overlapping it. If nearest is True, then only the nearest features are returned.

Parameters

- **position** – the position
- **nearest** – if True, only the nearest features are returned.
- **strand** – strand specification of features to be returned. If strand is 0, then features on both strands are returned

Returns a list of features around the specified position

join (*other=None, offset=0, keepself=True*)

Joins two SeqRecordEM2 objects into a new one representing the resulting merged sequence

Parameters

- **keepself** – if True and overlapping subsequences are different, then keep sequence from self record, otherwise keep the sequence of other record.
- **other** – the other SeqRecordEM2 object
- **offset** – the offset of the two sequences. If the value is negative, then the two sequences overlap.

Returns the result of merging records as a new SeqRecordEM2 object

stitch (*other, fpos_in_self, fpos_in_other, feature_length, orientation=1, **kwargs*)

Stitches two records, that is, joins them according to an overlapping feature. The sequences may or may not overlap. If not, Ns or Xs are added to fill the gap. If they overlap, a warning is issued if sequences do not correspond exactly. The new record keeps track of the two original records as Features. By convention, the self record should contain the start position of the feature on the forward strand, the other contains the end position on the forward strand if orientation=1 or on the reverse strand if orientation=-1. If orientation is -1, then the other record is reversed/complemented before stitching and the position of the overlapping feature is modified accordingly. It is the user's responsibility to provide the records in the right order.

Parameters

- **other** – the other SeqRecordEM2 object to stitch
- **fpos_in_self** – feature position in self record (start position of feature)
- **fpos_in_other** – feature position in other record (end position of feature), according to FeatureLocation conventions for end position requiring that length = end - start, it is not included in the feature.
- **feature_length** – feature length
- **orientation** – the orientation of the other record relative to the self, either 1 if it is in the same orientation, -1 if other needs to be reversed before stitching, 0 if stranded but unknown, None for proteins

- **kwargs** – any additional parameters that may be passed to the constructor of the stitching feature in the new record

Returns the stitched record as a new SeqRecordEM2 object

reverse_complement (*id=False, name=False, description=False, features=True, annotations=False, letter_annotations=True, dbxrefs=False*)

Reverse-complement the record adjusting features and their positions accordingly. The record id is conserved but if name is not specified 'reversed' is appended. All other arguments are passed and handled by the parent method. Note that the main goal for this method is to replace SeqRecord and Seq objects by their SeqRecordEM2 and SeqEM2 equivalents when reverse/complementing.

Parameters

- **id** – the id for the reversed record
- **name** – the name for the reversed record
- **description** – the description for the reversed record
- **features** – keep and adjust location of features if True
- **annotations** – keep annotations if True
- **letter_annotations** – keep letter_annotations if True
- **dbxrefs** – keep dbxrefs if True

Returns a reversed copy of the record

orfs_to_features (*start=['ATG'], stop=None, filter=None, add=False*)

Determines all open reading frames in a sequence record. All the returned ORFs have a length that is a multiple of 3. Thus, for sequences without any stop codon, 3 ORFs are returned, one for each frame. Both strands are examined but it is possible to filter the ORFs by length, frame, etc. with the FeatureFilter defined by the filter argument.

Parameters

- **start** – a list of accepted start codons
- **stop** – a list of accepted stop codons
- **filter** – a FeatureFilter object defining a filter to select ORFs according to some criteria
- **add** – if True, the selected ORFs are added to the record's features

Returns a list of ORFs as SeqFeatureEM2 objects

2.2 The “seq” module

Extension of Bio.Seq.Seq class from Biopython to add or improve functionalities

class em2lib.seq.**SeqEM2** (*data, seqtype*)

SeqEM2 class providing extension to Bio.Seq.Seq class of BioPython package.

classmethod **dna** (*data*)

Creates a DNA sequence

Parameters **data** – the sequence string

Returns a SeqEM2 DNA instance

classmethod **protein** (*data*)

Creates a protein sequence

Parameters **data** – the sequence string

Returns a SeqEM2 protein instance

is_protein ()

Tests whether sequence was created as a protein

Returns boolean, True if sequence was created as a protein, False otherwise.

length_in_range (*minlength=None, maxlength=None*)

Checks whether the sequence length is with the specified range.

Parameters

- **minlength** – lower length bound
- **maxlength** – upper length bound

Returns True if sequence length is within specified range, False otherwise

re_search (*regex*)

Searches a sequence using a regular expression

Parameters **regex** – the regular expression

Returns a list of re.Match instances

search (*pattern*)

Searches sequence for a pattern specified with a fuzznuc or fuzzpro like syntax

Parameters **pattern** – the pattern to be searched for that is converted into a regular expression

Returns a list of re.Match objects

get_orfs (*start=['ATG'], stop=None*)

Determines all open reading frames in a sequence. It only examines the forward strand. All the returned ORFs have a length that is a multiple of 3. Thus, for sequences without any stop codon, 3 ORFs are returned, one for each frame.

Parameters

- **start** – list of accepted start codon or None if ORFs do not need to start at a start codon.
- **stop** – list of accepted stop codons

Returns a set of tuples (start, end) of orfs where start is the starting position of the orf and end its ending position, not including the stop codon

2.3 The “seq_feature” module

Extension of Bio.SeqFeature module from Biopython to add or improve functionalities

class em2lib.seq_feature.**SeqFeatureEM2** (*parent=None, **kwargs*)

SeqFeatureEM2 class providing extension to Bio.SeqFeature.SeqFeature class of BioPython package.

length_in_range (*minlength=None, maxlength=None*)

Checks whether the feature length is with the specified range.

Parameters

- **minlength** – lower length bound
- **maxlength** – upper length bound

Returns True if feature length is within specified range, False otherwise

lies_within (*start*, *end*)

Determines whether feature lies entirely with the specified range. Fuzzy positions are turned into integers.

Parameters

- **start** – start of range either int or ExactPosition
- **end** – end of range either int or ExactPosition

Returns True if feature boundaries lie with the specified range.

covers (*start*, *end*)

Determines whether feature covers the whole range specified by start and end

Parameters

- **start** – start of range either int or ExactPosition
- **end** – end of range either int or ExactPosition, if None then end=start

Returns True if feature covers the specified range

overlaps (*start*, *end=None*)

Determines whether feature overlaps a position range.

Parameters

- **start** – start of range either int or ExactPosition
- **end** – end of range either int or ExactPosition

Returns True if feature overlaps range

intersect (*other*, ***kwargs*)

Creates a new feature which is the intersection of feature and another one

Parameters **other** – the other feature

move (*offset*)

Moves a feature by a certain offset

Parameters **offset** – offset by which the feature must be moved

class `em2lib.seq_feature.FeatureFilter`

A class for the definition and application of a filter to a list of features. May be used to retrieve features from a record or any list of features according to length, location, type, strand and frame. It is possible to combine several criteria into one single filter.

keep (*keep=True*)

Set the keep attribute of the filter. If True, the features corresponding to the specified criteria will be kept, otherwise, they will be discarded.

Parameters **keep** – boolean, True if features consistent with the criteria should be kept.

Returns the current filter

type (*feat_type=None*)

Set the type of feature.

Parameters **feat_type** – the type of feature to select

Returns the current filter

length (*minlength=None*, *maxlength=None*)

Set the length range of the feature.

Parameters

- **minlength** – the minimum length of the feature or 0 if None
- **maxlength** – the maximum length of the feature or no limit if set to None

Returns the current filter

strand (*strand=0*)

Set the strand of the feature.

Parameters **strand** – the strand of the feature

Returns the current filter

frame (*frame=0, strand=1*)

Set the frame of the filter. By default, this is on the forward strand unless the strand is set to -1.

Parameters

- **frame** – the frame 0, 1 or 2
- **strand** – the strand 1 or - 1

Returns the current filter

covers (*start=None, end=None*)

Set a region that must be covered by the feature.

Parameters

- **start** – start position of the region
- **end** – end position of the region

Returns the current filter

overlaps (*start=None, end=None*)

Set a region that must overlap the feature.

Parameters

- **start** – start position of the region
- **end** – end position of the region

Returns the current filter

lies_within (*start=None, end=None*)

Set a region within which the feature must lie.

Parameters

- **start** – start position of the region
- **end** – end position of the region

Returns the current filter

type_applies (*feature*)

Test if type criterion applies to the feature and return a boolean stating whether the feature should be kept or rejected. If the criterion has not been set, then True is returned.

Parameters **feature** – the feature to test

Returns True the feature must be kept

length_applies (*feature*)

Test if length criterion applies to the feature and return a boolean stating whether the feature should be kept or rejected. If the criterion has not been set, then True is returned.

Parameters **feature** – the feature to test

Returns True the feature must be kept

covers_applies (*feature*)

Test if covers criterion applies to the feature and return a boolean stating whether the feature should be kept or rejected. If the criterion has not been set, then True is returned.

Parameters **feature** – the feature to test

Returns True the feature must be kept

overlaps_applies (*feature*)

Test if overlaps criterion applies to the feature and return a boolean stating whether the feature should be kept or rejected. If the criterion has not been set, then True is returned.

Parameters **feature** – the feature to test

Returns True the feature must be kept

lies_within_applies (*feature*)

Test if lies_within criterion applies to the feature and return a boolean stating the feature should be kept or rejected. If the criterion has not been set, then True is returned.

Parameters **feature** – the feature to test

Returns True the feature must be kept

location_applies (*feature*)

Test if location criterion applies to the feature and return a boolean stating whether the feature should be kept or rejected. If the criterion has not been set, then True is returned.

Parameters **feature** – the feature to test

Returns True the feature must be kept

strand_applies (*feature*)

Test if strand criterion applies to the feature and return a boolean stating whether the feature should be kept or rejected. If the criterion has not been set, then True is returned.

Parameters **feature** – the feature to test

Returns True the feature must be kept

frame_applies (*feature*)

Test if frame criterion applies to the feature and return a boolean stating whether the feature should be kept or rejected. If the criterion has not been set, then True is returned.

Parameters **feature** – the feature to test

Returns True the feature must be kept

apply (*features*)

Test if all defined criteria apply to the features in the list and return the list of features corresponding to the specified criteria.

Parameters **features** – the list of features to filter

Returns the filtered list of features

2.4 The “seq_utils” module

Extension module to the Biopython Bio.SeqUtils module

`em2lib.seq_utils.ambiguous2string (code, protein=False)`

Converts an ambiguous residue into a string with all compatible unambiguous residues. If the input code is not ambiguous, it is returned without any conversion.

Parameters

- **code** – the input code to be converted into a list of residues.
- **protein** – True if residue is amino-acid

Returns a string corresponding to the unambiguous residues compatible with the input code

`em2lib.seq_utils.isambiguous (code, protein=False)`

Checks code is an ambiguous residue specification or not.

Parameters

- **code** – the input code that must be checked for ambiguity
- **protein** – True if code is amino-acid code

Returns boolean, True if code is ambiguous, False otherwise

`em2lib.seq_utils.pattern2regex (pattern, protein=False)`

Converts a fuzznuc or fuzzpro-like pattern into a regular expression that can be used to search a sequence string.

- `[ABC]` => any of ABC residues,
- `{ABC}` => any residue except ABC,
- `<ABC...` => start of sequence,
- `...ABC>` => end of sequence,
- `A(n)(ABC)(n)` => repeat residue or subsequence n times,
- `A(n,m)(ABC)(n,m)` => repeat residue or subsequence from n up to m times.

Parameters

- **pattern** – the pattern definition (string)
- **protein** – True if pattern applies to a protein sequence, False otherwise.

Returns the regular expression pattern as a string

class `em2lib.seq_utils.SeqFilter`

A class for the creation of a sequence filter to specify filtering criteria and applying the filter to a list of sequence records.

length (*minlength=None, maxlength=None*)

Minimal and maximal length specification

Parameters

- **minlength** – minimal accepted length
- **maxlength** – maximal accepted length

Returns SeqFilter instance

pattern (*pattern=None*)
pattern specification

Parameters **pattern** – pattern that must be in the sequence

Returns SeqFilter instance

name (*name=None*)
sequence record name specification

Parameters **name** – name regular expression

Returns SeqFilter instance

keep (*keep=True*)
Boolean defining whether the matching sequences must be kept (True) or removed (False)

Parameters **keep** – True to keep positive sequences, False to remove them

Returns SeqFilter instance

length_applies (*rec*)
test whether length criterion applies to the sequence record

Parameters **rec** – the sequence record to test

Returns boolean True if criterion applies or False otherwise

pattern_applies (*rec*)
test whether parameter criterion applies to the sequence record

Parameters **rec** – the sequence record to test

Returns boolean True if criterion applies or False otherwise

name_applies (*rec*)
test whether name criterion applies to the sequence record

Parameters **rec** – the sequence record to test

Returns boolean True if criterion applies or False otherwise

apply (*records*)
Filters a list of SeqRecords instances, keeping only records satisfying the specified criteria of length, match of a pattern, name specification. It is possible to invert the filtering process by setting the keep boolean to False and thus only keep records which do not satisfy the criteria.

Parameters **records** – list of SeqRecord instances to apply

Returns the filtered list of records

class em2lib.seq_utils.**GFF** (*feature_list=None, input_df=None*)
Manipulation of features based upon gffpandas package

add_feature_list (*feature_list=None*)
Adds a list of feature to the list of an existing GFF object

Parameters **feature_list** – list of features to add to DataFrame

Returns the GFF object with feature list appended

static **df_from_feature** (*feature*)
Create a pandas DataFrame from a feature (SeqFeatureEM2 or SeqFeature)

Parameters **feature** – the feature to convert into a dataframe

Returns the resulting dataframe

to_feature_list (*parents=None*)

Converts features in a GFF object into a list of SeqFeatureEM2 objects

Parameters **parents** – list of references to parent SeqRecord objects or a single parent reference if all features are defined in the same parent. If it is a list, it should be of the same length as the dataframe, repeating references as needed to get the right number.

Returns a list of SeqFeatureEM2 objects

2.5 The “argparse_em2” module

Extension module to the standard argparse module. Adding custom actions and argument verification methods.

class `em2lib.argparse_em2.GetList` (*option_strings, dest, nargs='+', **kwargs*)

An argparse custom action to return a list from an argument containing a list of elements and/or file names. Files are supposed to contain one element of the list per line. There can be more than one file and the argument may take a combination of elements and files. In all cases, the returned list will contain all the specified elements without any checking for redundancy. If you need a non redundant set instead of a list, then use GetSet action instead.

static `arg2list` (*values*)

This method converts the argument values containing elements and/or files containing elements into a list of elements.

Parameters **values** – argument values, this is supposed to be a list of arguments or None (returns an empty list)

Returns the list of elements or an empty list if the argument was None

class `em2lib.argparse_em2.GetSet` (*option_strings, dest, nargs='+', **kwargs*)

An argparse custom action to return a set from an argument containing a list of elements and/or file names. Files are supposed to contain one element of the list per line. There can be more than one file and the argument may take a combination of elements and files. In all cases, the returned set will contain all the specified elements keeping only one copy of each element. If you do not want to remove redundancy, then use GetList action instead.

static `arg2set` (*values*)

This method converts the argument values containing elements and/or files containing elements into a set of elements.

Parameters **values** – argument values, this is supposed to be a list of arguments or None (returns an empty set)

Returns the set of elements or an empty set if the argument was None

INDICES AND TABLES

- `genindex`
- `modindex`
- `search`

PYTHON MODULE INDEX

e

- `em2lib`, [3](#)
- `em2lib.argparse_em2`, [12](#)
- `em2lib.seq`, [5](#)
- `em2lib.seq_feature`, [6](#)
- `em2lib.seq_record`, [3](#)
- `em2lib.seq_utils`, [10](#)