mutalyzer-backtranslate

Release 1.0.0

LUMC, Jeroen F.J. Laros

Nov 21, 2021

CONTENTS:

This library provides functions for back translation of amino acid changes to nucleotide changes.

Features:

- Support for all known codon tables.
- Back translation of amino acid changes using codon reference information.
- Back translation of amino acid changes using amino acid reference information.
- Function to determine all amino acid substitutions of which the back translation can be improved by adding codon information.

Please see ReadTheDocs for the latest documentation.

CHAPTER

ONE

QUICK START

The `BackTranslate` class provides functionality for back translation.

```
>>> from mutalyzer_backtranslate import BackTranslate
>>> bt = BackTranslate()
```

An amino acid change from a Leucine to a Phenylalanine can be explained by five substitutions.

```
>>> bt.without_dna('L', 'F')
{2: {('A', 'T'), ('A', 'C'), ('G', 'C'), ('G', 'T')}, 0: {('C', 'T')}}
```

If codon information is present, the same substitution can only be explained by one substitution.

```
>>> bt.with_dna('CTT', 'F')
{0: {('C', 'T')}}
```

1.1 Introduction

Back translation or Reverse translation is the conversion of a protein sequence into a nuclear sequence. In general, there are multiple nuclear sequences that can be translated into the same protein sequence, so the reverse of this operation usually results in a large number of candidate nuclear sequences.

A related problem is the back translation of an amino acid *change* to a nuclear *change*. In this case, we know a reference amino acid and an observed amino acid and we are interested in the nuclear variant that gave rise to this change. Since there are infinitely many ways of making such a transformation, we restrict ourselves to substitutions of one nucleotide that may explain the observed amino acid change.

For example, we might be interested in which nucleotide substitutions will transform a Tryptophan into a stop codon. It turns out that there are two possible one nucleotide substitutions that have this effect; replace either `G` with an `A`.

This form of back translation can be done by making use of reference information. This information can be provided in the form of a reference amino acid, or a reference codon. In general, using a reference codon will yield fewer possibilities when compared to using a reference amino acid.

For example, there are five one nucleotide substitutions that can transform a Leucine to a Phenylalanine. If we happen to know that the reference codon was `CTT` though, then there is only one substitution that can explain this transformation.

This library provides functionality to back translate amino acid changes using either a reference amino acid or a reference codon. Furthermore, it provides a function that, given a codon table, will list all amino acid substitutions of which the back translation can be improved by adding codon information.

1.2 Installation

The software is distributed via PyPI, it can be installed with `pip`:

```
pip install mutalyzer-backtranslate
```

1.2.1 From source

The source is hosted on GitHub, to install the latest development version, use the following commands.

```
git clone https://github.com/mutalyzer/backtranslate.git
cd backtranslate
pip install .
```

1.3 Usage

Use the command `backtranslate` to find substitutions that explain an amino acid change:

```
$ backtranslate with_dna -o 210 data/mhv.fa - 1 Leu
1       A       C
1       A       T
```

If no reference is available, use the `without_dna` subcommand:

```
$ backtranslate without_dna - Asp 92 Tyr
274     G       T
```

The command `find_stops` finds a list of positions and substitutions that lead to stop codons. This list of destructive substitutions are useful when analysing a pool of viral transcripts. Counting the appropriate nucleotides at the given positions gives insight into how many transcripts are active.

```
$ backtranslate find_stops -o 210 data/mhv.fa -
216     A       T
225     A       T
230     C       A
230     C       G
243     A       T
...
```

1.4 Library

This library provides functions for back translation from amino acid changes to nucleotide changes.

```
>>> from mutalyzer_backtranslate import BackTranslate
>>>
>>> # Create a class instance, optionally giving the translation table id.
>>> bt = BackTranslate()
>>>
```

(continues on next page)

(continued from previous page)

```
>>> # Find all substitutions that transform the codon 'TGG' into a stop codon.
>>> bt.with_dna('TGG', '*')
{1: {('G', 'A')}, 2: {('G', 'A')}}
```

Sometimes we do not have access to the DNA sequence so we have to find possible substitutions from the amino acids directly.

```
>>> # Find all substitutions that transform a Tryptophan into a stop codon.
>>> bt.without_dna('W', '*')
{1: {('G', 'A')}, 2: {('G', 'A')}}
```

To find out which substitution predictions can be improved by adding codon information, use the following function.

```
>>> bt.improvable()
{('N', 'K'), ('R', 'W'), ('L', 'F'), ('D', 'E'), ('E', 'D'), ('C', 'W'),
 ('K', 'N'), ('Q', 'H'), ('C', '*'), ('I', 'L'), ('G', 'R'), ('F', 'L'),
 ('S', '*'), ('T', 'S'), ('*', 'S'), ('S', 'R'), ('R', 'S'), ('V', 'L'),
 ('R', 'G'), ('Y', '*'), ('S', 'T'), ('*', 'L'), ('*', 'W'), ('H', 'Q'),
 ('L', 'I'), ('I', 'M'), ('L', 'V'), ('L', 'M'), ('L', '*'), ('R', '*'),
 ('S', 'C'), ('*', 'Y')}
```

To get substitutions in a readable format, we can use the following:

```
>>> from mutalyzer_backtranslate.util import subst_to_cds
>>>
>>> substitutions = bt.without_dna('W', '*')
>>>
>>> # Transform the substitutions to CDS coordinates.
>>> subst_to_cds(substitutions, 12)
{(14, 'G', 'A'), (15, 'G', 'A')}
```

1.5 API documentation

1.5.1 Main library

class mutalyzer_backtranslate.backtranslate.**BackTranslate**(*table_id=1*)

Back translation.

Parameters **table_id** (*int*) – Translation table id.

improvable()

Calculate all pairs of amino acid substututions that can be improved by looking at the underlying codon.

Returns list List of improvable substitutions.

with_dna(*reference_codon*, *amino_acid*)

Find single nucleotide substitutions that given a reference codon explains an observed amino acid.

Parameters

- **reference_codon** (*str*) – Original codon.
- **amino_acid** (*str*) – Observed amino acid.

Returns dict Set of single nucleotide substitutions indexed by position.

without_dna(*reference_amino_acid*, *amino_acid*)
Find single nucleotide substitutions that given a reference amino acid explains an observed amino acid.

Parameters

- **reference_amino_acid** (*str*) – Original amino acid.
- **amino_acid** (*str*) – Observed amino acid.

Returns dict Set of single nucleotide substitutions indexed by position.

mutalyzer_backtranslate.backtranslate.**cmp_subst**(*subst_1*, *subst_2*)
Compare two substitution sets.

Parameters

- **subst_1** (*dict*) – Substitution set.
- **subst_2** (*dict*) – Substitution set.

Returns bool True if *subst_1* equals *subst2*, False otherwise.

mutalyzer_backtranslate.backtranslate.**reverse_translation_table**(*table_id=1*)
Calculate a reverse translation table.

Parameters **table_id** (*int*) – Translation table id.

Returns dict Set of possible codons indexed by amino acid.

1.5.2 Utils

mutalyzer_backtranslate.util.**subst_to_cds**(*substitutions*, *offset*)
Convert a set of substitutions to CDS coordinates.

Parameters

- **substitutions** (*dict*) – Set of single nucleotide substitutions indexed by position.
- **offset** (*int*) – Codon position in the CDS.

Returns set Substitutions in CDS coordinates.

1.6 Contributors

- Jeroen F.J. Laros <jlaros@fixedpoint.nl> (Original author, maintainer)
- Martijn Vermaat <martijn@vermaat.name> (Continuous integration setup)

Find out who contributed:

```
git shortlog -s -e
```

PYTHON MODULE INDEX

m

INDEX