

Barnacle Manual

Package version 0.2

June 19, 2009

Contents

1	Introduction	1
2	Installation	2
2.1	Prerequisites	2
2.1.1	Mocapy is included as a sub-package	2
2.2	Installation procedure	2
3	Using the model	2
3.1	Importing Barnacle and constructing a <code>Barnacle</code> object	2
3.2	Sampling and partial resampling	3
3.3	Accessing a sampled sequence and structure	3
3.4	Calculating the likelihood of the sampled structure	4
3.5	Undoing the sampling	4
3.6	Model parameters	5
4	Command line script	5
4.1	Example	5
5	License	5
	References	5

1 Introduction

This Python package contains a reference implementation of the probabilistic model of RNA conformational space called Barnacle. The full details of the model have been described in the literature [1]; in this manual we will only describe the functionality of the package.

The main feature of Barnacle is the ability to sample RNA structures that have an “RNA-like” geometry on a local length scale. It is possible to calculate a likelihood for each sample.

2 Installation

2.1 Prerequisites

The Barnacle package is written in Python and works with Python 2.4 (or newer). The only non-standard Python packages required by Barnacle are `numpy` and `Biopython` [2, 3].

Barnacle runs on any platform where Python and `numpy` can be installed. It has been successfully tested on common Linux, Unix and Windows platforms (including Debian Linux, Ubuntu Linux, Mac OS X and Cygwin for Windows).

2.1.1 Mocapy is included as a sub-package

Note that the Barnacle package includes minimal version of Mocapy 0.726 [4] as a sub-package.

2.2 Installation procedure

Barnacle makes use of the standard tool `Distutils` for installation. To install the package, you must first unpack it using the command:

```
$ tar -xzf Barnacle-0.2.tar.gz
```

Once the package has been unpacked, the install script `setup.py` is located in the extracted directory. The package can now be installed using the command:

```
$ python setup.py install
```

This will install the package in the standard path for Python package. If you want to install the Barnacle in a non-standard location, it can be done using the command:

```
$ python setup.py install --install-lib MYPATH
```

See the help text for `setup.py` for further options on alternative installation paths.

3 Using the model

3.1 Importing Barnacle and constructing a Barnacle object

After successful installation, the main class `Barnacle` can be imported from the package using the standard commands:

```
>>> from Barnacle import Barnacle
```

The `Barnacle` class is extensively documented using *docstrings* and a help message for the class can be displayed using the command:

```
>>> help(Barnacle)
```

A `Barnacle` object can be constructed using the class constructor:

```
>>> model = Barnacle("GGGCGCAAGCCU")
```

The constructor takes a nucleotide sequence as an argument. The object can then generate angle sequences and RNA structures for this sequence.

In the rest of this manual, we will assume that the class `Barnacle` has been imported and that the `Barnacle` object `model` has been created.

3.2 Sampling and partial resampling

Once a `Barnacle` object has been constructed, we can sample structures that are compatible with the given sequence, using the `sample` method. If no arguments are given to the method, all dihedral angles for all nucleotides in the sequence are sampled:

```
>>> model.sample()
```

Optionally, this method can also be used to partially resample a subsequence of the angles, using the arguments `start` and/or `end`.

```
>>> model.sample(start=2, end=6)
```

This will resample the dihedral angles for the nucleotides in the sequence from position index `start` and up to index `end` (but not including). The angles are sampled conditionally upon the angles in the remaining nucleotides, sampled previously using the `sample` method.

3.3 Accessing a sampled sequence and structure

After the `sample` method has been invoked, the sampled angle sequence and structure can be accessed using a set of methods from the `Barnacle` class.

The sampled angles can be extracted using the `get_angles` method:

```
>>> model.get_angles()
[(5.0693, 2.9940, 0.9832, 1.5042, 3.3909, 4.4710, 3.2939),
 (2.5254, 3.1806, 3.2267, 1.5898, 3.6029, 4.8716, 3.2674),
 ...]
```

The angles are returned as a list of tuples. Each tuple in the list contains the angles for the nucleotide at the corresponding position in the nucleotide sequence (specified upon creation of the object). The order of the angles in each tuple is:

$$[(\alpha_0, \beta_0, \gamma_0, \delta_0, \epsilon_0, \zeta_0, \chi_0), (\alpha_1, \beta_1, \gamma_1, \delta_1, \epsilon_1, \zeta_1, \chi_1), \dots]$$

Note that the angles are measured in radians. The 7 dihedral angles are illustrated on figure 1.

The sampled structure can be returned as a `Bio.PDB.Structure` object using the `get_structure` method. It can also be saved in PDB-format using the `save_structure` method:

```
>>> model.save_structure("output.pdb")
```

The PDB file format follows the naming conventions specified by the PDB format versions 3 [5, 6] and has been tested to work correctly in most common PDB-viewers (PyMol, RasMol, etc.).

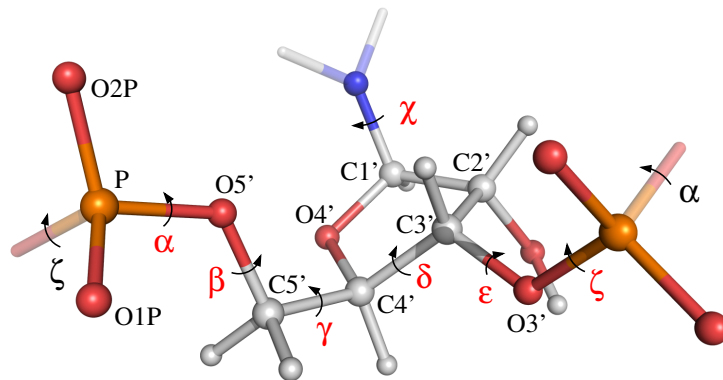


Figure 1: Ball-and-stick representation of an RNA fragment. The seven relevant dihedral angles in the central nucleotide (α to ζ) are indicated with red labels. Each label is placed on the central bond of the four consecutive atoms that define the dihedral angle. The χ angle describes the rotation of the base relative to the RNA backbone, while the six other angles define the course of the backbone.

3.4 Calculating the likelihood of the sampled structure

The log likelihood of the sampled structure can be calculated using the method `get_log_likelihood`, as shown below:

```
>>> model.get_log_likelihood()
55.7817
```

3.5 Undoing the sampling

The last call to the `sample` method can be undone using the `undo` method. This method brings the `Barnacle` object back to the state it was in, before last call to the `sample` method.

```
>>> model.sample()
>>> model.get_angles()
[(1.1500, 3.2873, 1.0878, 1.6322, 2.9815, 0.8866, 3.1110), ...]
>>> model.sample()
>>> model.get_angles()
[(1.5622, 3.0059, 2.9356, 1.3923, 3.5417, 4.9928, 3.1683), ...]
>>> model.undo()
>>> model.get_angles()
[(1.1500, 3.2873, 1.0878, 1.6322, 2.9815, 0.8866, 3.1110), ...]
```

The `undo` method also works, if the last call was a partial resampling. However, the `undo` method can only be invoked once for each use of the `sample` method.

The `undo` method is essential, if the model is to be used in a MCMC framework.

3.6 Model parameters

The model parameters are available in the module `model_parameters`. See the file for further comments.

4 Command line script

We have provided a command line script as a simple demonstration of the package. The command line script can be used to sample RNA structures for a nucleotide sequence.

The command line script is available without any installation. The only requirements for the command line script is Python 2.4 (or newer) and the `numpy` package. To use the script, you must first obtain the Barnacle package and unpack it using the command:

```
$ tar -xzf Barnacle-0.2.tar.gz
```

The script `BarnacleSampler.py` is located in the extracted directory; this is the command line script. A help message for the script can be displayed using the command:

```
$ python BarnacleSampler.py -h
```

4.1 Example

We want to sample 5 structures that are compatible with the sequence GGGCG-CAAGCCU and save the output in `testXXXXX.pdb`. This this can be done using the command:

```
$ python BarnacleSampler.py -n 5 -s GGGCGCAAGCCU -l 0 -o test
Starting sampling:
test00000.pdb: ll=25.664357
test00001.pdb: ll=-28.728932
test00002.pdb: ll=-4.257696
test00003.pdb: ll=36.157002
test00004.pdb: ll=25.160863
DONE
```

The option `'-l 0'` means that in each iteration all angles are sampled. The option `'-l 2'` would mean that in each iteration the angles for two randomly chosen consecutive nucleotides are resampled.

5 License

The Barnacle package is published under the GNU General Public License version 3. See the file `LICENSE` or <http://www.gnu.org/licenses/> for full details.

References

- [1] Frelsen J, Moltke I, Thiim M, Mardia KV, Ferkinghoff-Borg J, Hamelryck T (2009) A Probabilistic Model of RNA Conformational Space. *PLoS Comput Biol* 5(6): e1000406.
- [2] Hamelryck T, Manderick B (2003) PDB file parser and structure class implemented in Python. *Bioinformatics* 19: 2308–2310.
- [3] <http://biopython.org/>
- [4] <http://sourceforge.net/projects/mocapy/>
- [5] Berman HM, Henrick K, Nakamura H (2003) Announcing the worldwide Protein Data Bank. *Nat Struct Biol* 10:980–980.
- [6] <http://www.wwpdb.org/docs.html>