NeSSie: nucleic-acids elements of sequence symmetry identification

NeSSie is a c/c++ 64 bit program that allows to perform fast patterns search and sequence analyses on DNA strings using the NeSSie library.

The tool currently offers the following types of analyses:

- *De novo* search for motifs with a *mirror* or *palindromic* symmetry, as well as motifs with a *DNA-triplex* forming potential. Both *perfect* and *degenerate* motifs (motifs containing gaps and mismatches) can be detected.
- Exhaustive search for all k-mers in the sequence.
- Calculation of the sequence complexity and entropy on the full sequence, or using a sliding window of given size and shift.
- Exact search for motifs in a provided list.

Contacts

Michele Berselli, berselli.michele@gmail.com

Install from source

A make file is available in the NeSSie folder to easily compile the program. To set things up:

- Download the repository
- Unpack the repository and position inside the folder
  `unzip path/to/packed/folder` and `cd path/to/unpacked/folder`
- Compile the program
  `make`

This will generate the binary file nessie inside the folder.

Using the library

To use the library simply `#include "Nessie.h"` into your code.

Quick guide

This is a quick reference guide, to see more details on specific analyses check the relative sections.

Input

The program accepts in input both *fasta* and *multi-fasta* files.

***note**: the program can handle uppercase and lowercase letters, as well as the presence of N*

in the input sequence. It cannot handle IUPAC symbols different from canonical bases A, C, T, G.

Basic command lines

- *De novo* search for exact motifs with *mirror* (**-M**) or *palindromic* (**-P**) symmetry or motifs with a *DNA-triplex* forming potential (**-T**). To allow for degeneration use additional parameters as explained below.
  `nessie -I path/input/file -O path/output/file {-P | -M | -T} -k N [ADDITIONAL ARGUMENTS]`
- Search for all k-mers in the sequence (**-A**)
  `nessie -I path/input/file -O path/output/file -A -k N [ADDITIONAL ARGUMENTS]`
- Analysis of the sequence complexity (**-L**) and entropy (**-E**)
  `nessie -I path/input/file -O path/output/file {-E | -L} [ADDITIONAL ARGUMENTS]`
- Exact search (**-N**) for motifs provided in *path/file/w/motifs*.
  `nessie -I path/input/file -O path/output/file -N path/file/w/motifs [ADDITIONAL ARGUMENTS]`

where **-k** is the length of the motifs or k-mers to be searched and *N* is a positive integer.

***note**: the shown arguments are required in the correct order, the additional arguments can be instead specified in any order!*

Additional arguments for all searches

- A sub-interval of the sequence can be used for the analysis. **-b** *N* is used to define the starting index of the interval and **-e** *N* is used to define the ending index of the interval. *N* is a positive integer.
- The reverse complement of the sequence can be used for the analysis using the **-C** flag.

Additional arguments for -P/-M/-A/-L/-T

- The minimum and maximum length of the motifs or k-mers to be searched or to be used for the complexity calculation can be defined using **-k** *N* for the minimum length and **-K** *N* for the maximum length. *N* is a positive integer.

  ***note**: a maximum length is required if the -MAX parameter is used, a minimum length is required for -P/-M/-A/-T searches*.

Additional arguments for -P/-M/-T

- The tool is flexible enough to allow for the presence of mismatches and gaps that impair the symmetry of the motifs. **-m** *N* is used to define the percentage of

permitted mismatches while **-g** *N* is used to define the percentage of permitted gaps. It is also possible to define a cumulative percentage for both mismatches and gaps using the **-t** *N* parameter. *N* is a positive integer.

- The **-MAX** flag activate the search mode for the longest motif only at each position. **-k** *n* and **-K** *N* are used to specify the length range for the motif [*n*...*N*]. *n* and *N* are positive integers and $n < N$.

Additional arguments for -E/-L

- A sliding window of given length and shift can be used to calculate the linguistic complexity and entropy. **-l** *N* is used to define the window length and **-s** *N* is used to define the window shift. *N* is a positive integer.

Additional arguments for -T

- **-p** *N* can be used to define the percentage of permitted non-purine bases in the motifs for the DNA-triplexes search. *N* is a positive integer.

Output format

Different output files are generated depending on the type of analysis.

- Standard output for perfect motifs

```
>SEQUENCE_1_NAME
$|12|AGAAGAAGAAGA
@counts: 6
@indexes: 2|5|8|11|14|17|
$|10|TCTTCCTTCT
@counts: 2
@indexes: 30|42
```

- Output for exact-motifs search

```
>SEQUENCE_1_NAME
!MOTIF_1_NAME
$|3|AAA
@counts: 3
@indexes: 3|4|34|
!MOTIF_2_NAME
$|3|CCC
@counts: 2
@indexes: 12|20|
```

where `$|12|AGAAGAAGAAGA` reports the retrieved motif and its length, `@counts: 6` reports the number of occurrences for the motif and `@indexes: 2|5|8|11|14|17|`

reports the indexes at which the motif was found (*i.e.* positions in the sequence). A new block starting with `>SEQUENCE_NAME` is created for each of the target sequences if a *multi-fasta* is provided as input. `!MOTIF_NAME` is the name of the motif to be searched as provided in the *fasta* / *multi-fasta* file with motifs.

- Standard output for degenerate motifs

```
>SEQUENCE_NAME
$|21|AAAAAAAATAGATCAAATAAA|01010110010101100010111
@counts: 1
@indexes: 61577|
$|10|AAAAAAAATA|0110010101
@counts: 2
@indexes: 133061|805355|
```

If degenerate motifs are searched, the additional field `01010110010101100010111` is reported. This represents the encoding of the best alignment retrieved for the corresponding sequence, and it is used by the NeSSie Output Parser (see next section) to explicitly print the alignment if desired. 00 and 11 represent indels, 01 represent a match, 10 represent a mismatch.

- Output for complexity and entropy when calculated on the entire sequence

```
>SEQUENCE_1_NAME
@0-91: 0.891170431211499
>SEQUENCE_2_NAME
@0-30: 0.9459459459459459
```

where `@0-91` is the interval on which the score `0.891170431211499` is calculated.

- Output for complexity calculation using a sliding window of give size and shift

```
>SEQUENCE_1_NAME
@0-30
0       0.7755102040816326
5       0.9795918367346939
10      0.8775510204081632
15      0.9387755102040817
20      0.8775510204081632
```

where `@0-30` is the total length of the interval on which the scores are calculated. `0 0.7755102040816326` reports the relative starting index of the window and the corresponding calculated score. A new block starting with `>SEQUENCE_NAME` is created for each of the target sequences if a *multi-fasta* is provided as input.

- Output for entropy calculation using a sliding window of give size and shift

```
>SEQUENCE_1_NAME
@0-30
0       0.4854752972273343      A:6     C:0     G:4     T:0
5       0.8427376486136672      A:5     C:1     G:3     T:1
10      0.9609640474436811      A:2     C:2     G:4     T:2
15      0.7609640474436811      A:0     C:4     G:4     T:2
20      0.6804820237218405      A:0     C:4     G:1     T:5
```

where `@0-30` is the total length of the interval on which the scores are calculated. `0 0.4854752972273343 A:6 C:0 G:4 T:0` reports the relative starting index of the window, the corresponding calculated score and the bases composition of the sequence. A new block starting with `>SEQUENCE_NAME` is created for each of the target sequences if a *multi-fasta* is provided as input.

To reduce the output file, the **-c** flag can be used to report only counts while the **-i** flag can be used to report only indexes.

A log file that contains information on errors occurred during the analysis is produced as well as output in the working directory.

NeSSie Output Parser (NessieOutParser.py)

Together with NeSSie, a python script is also provided that can be used to better organize the raw output obtained for the search of *mirror* and *palindromic* motifs, as well as the motifs with a *DNA-triplex* forming potential.

In the presence of *N* the sequence is splitted into blocks and each blocks analysed separately. If the same motif is detected in different blocks, the hit will be reported for every block with the associated indexes at which it is found in that block. This can lead to a redundancy of some hits in the results. The parser allows to join this redundant motifs together under one hit while ordering the results. The results can be ordered:

- by indexes (lowest to highest) as a default
- by counts (highest to lowest) using the **-c** flag
- by score (highest to lowest) using the **-s** flag

***note**: the score is calculated as the length of the motif minus the number of mismatches and gaps. Mismatch or gap opening scores -2, while mismatch or gap extension scores -1.*

The parser allows also to generate an output where the retrieved best alignments are explicitated using the **-a** flag. Finally the **-g** flag will generate a GFF format file to simplify the visualization of the results using a genome browser.

***note**: when generating the GFF, a color code is assigned to motifs based on their score. From lower to higher scores the colors are red, yellow, blue, green in the order*.

`NessieOutParser.py -i path/input/file -o path/output/file [-c] [-a] [-g] [-s]`

***note**: Python 2.7 is required*.

To wig (to_wig.py)

Together with NeSSie, a python script is also provided that can be used to better visualize the raw output obtained for the analyses of the sequence entropy and complexity. The script allows to generate from the output a WIG format file that can be visualized using a genome browser.

```
to_wig.py -i path/input/file -o path/output/file
```

***note**: Python 2.7 is required*.

To tab formatted file (to_tabformat.py)

Together with NeSSie, a python script is also provided that can be used to organize the raw output obtained for the search of *mirror*, *palindromic* and *DNA-triplex* forming motifs in a tab formatted file. The results can be ordered by indexes (default), by score (**-s**) or by counts (**-c**).

```
to_tabformat.py -i path/input/file -o path/output/file [-c] [-s]
```

***note**: Python 2.7 is required*.

De novo search for motifs with *mirror* or *palindromic* symmetry

The program allows to perform a *de novo* search for motifs with a *mirror* or *palindromic* symmetry in a DNA sequence. Both *perfect* and *degenerate* motifs can be detected since the program is flexible enough to allow for the presence of mismatches and gaps that impair the symmetry. The search is exhaustive and all the motifs corresponding to the defined parameters are reported.

This is a basic command line that allows to detect all the perfect *mirror* (**-M**) or *palindromic* (**-P**) motifs of length *n* in the sequence. The input must be a file in *fasta* or *multi-fasta* format. *n* is a positive integer.

```
nessie -I path/input/file -O path/output/file {-P | -M} -k n
```

To search for all motifs in a range of length [*n*...*N*] the **-K** *N* parameter can be added to the **-k** *n* parameter. *n* and *N* are positive integers and $n < N$.

```
nessie -I path/input/file -O path/output/file {-P | -M} -k n -K N
```

To search only for the longest motif in range [*n*...*N*] at each position it is possible to use the **-MAX** flag. This will speed up the search and only the longest non-contained motifs identified at each index will be reported.

```
nessie -I path/input/file -O path/output/file {-P | -M} -k n -K N -MAX
```

To allow for the presence of gaps the parameter **-g** *G* can be used to define the percentage of gaps allowed. *G* is a positive integer.

`nessie -I path/input/file -O path/output/file {-P | -M} -k n [-K N -MAX] -g G`

-m *M* can be used in combination with **-g** or alternatively to define the percentage of mismatches allowed. *M* is a positive integer.
`nessie -I path/input/file -O path/output/file {-P | -M} -k n [-K N -MAX -g G] -m M`

To specify a cumulative percentage for both mismatches and gaps the **-t** *T* parameter can be used. This allows to detect motifs with a total maximum percentage *T* of gaps and mismatches. *T* is a positive integer.
`nessie -I path/input/file -O path/output/file {-P | -M} -k n [-K N -MAX] -t T`

To limit the search to a sub-string of the sequence it is possible to use **-b** *B* to define the starting index of the interval and **-e** *E* to define the ending index of the interval. *B* and *E* are positive integers.

De novo search for motifs with *DNA-triplex* forming potential

The program allows to perform a *de novo* search for motifs with the potential to form *DNA-triplexes* in a DNA sequence. *DNA-triplexes* can form from homopurine motifs characterised by a *mirror* symmetry. Both *perfect* and *degenerate* motifs can be detected since the program is flexible enough to allow for the presence of mismatches and gaps that impair the symmetry as well as mixed purine-pyrimidine motifs. The search is exhaustive and all the motifs corresponding to the defined parameters are reported.

This is a basic command line that allows to detect all the perfect potential *DNA-triplex* forming motifs (**-T**) of length *n* in the sequence. The input must be a file in *fasta* or *multi-fasta* format. *n* is a positive integer.
`nessie -I path/input/file -O path/output/file -T -k n`

To search for all motifs in a range of length [*n*...*N*] the **-K** *N* parameter can be added to the **-k** *n* parameter. *n* and *N* are positive integers and $n < N$.
`nessie -I path/input/file -O path/output/file -T -k n -K N`

To search only for the longest motif in range [*n*...*N*] at each position it is possible to use the **-MAX** flag. This will speed up the search and only the longest non-contained motifs identified at each index will be reported.
`nessie -I path/input/file -O path/output/file -T -k n -K N -MAX`

To allow for the presence of gaps the parameter **-g** *G* can be used to define the percentage of gaps allowed. *G* is a positive integer.
`nessie -I path/input/file -O path/output/file -T -k n [-K N -MAX] -g G`

-m *M* can be used in combination with **-g** or alternatively to define the percentage of mismatches allowed. *M* is a positive integer.

`nessie -I path/input/file -O path/output/file -T -k n [-K N -MAX -g G] -m M`

To specify a cumulative percentage for both mismatches and gaps the **-t** *T* parameter can be used. This allows to detect motifs with a total maximum percentage *T* of gaps and mismatches. *T* is a positive integer.
`nessie -I path/input/file -O path/output/file -T -k n [-K N -MAX] -t T`

To allow for mixed purine-pyrimidine motifs the parameter **-p** *P* can be used to define the percentage of allowed non-purine bases.
`nessie -I path/input/file -O path/output/file -T -k n [-K N -MAX -t T] -p P`

To limit the search to a sub-string of the sequence it is possible to use **-b** *B* to define the starting index of the interval and **-e** *E* to define the ending index of the interval. *B* and *E* are positive integers.

Exhaustive search for all k-mers in the sequence

The program allows to perform an exhaustive search for all the different k-mers (words of length k) in the sequence.

This is a basic command line that allows to detect all the different k-mers of length *n* in the sequence. The input must be a file in *fasta* or *multi-fasta* format. *n* is a positive integer.
`nessie -I path/input/file -O path/output/file -A -k n`

To search for all k-mers in a range of length [*n*...*N*] the **-K** *N* parameter can be added to the **-k** *n* parameter. *n* and *N* are positive integers and $n < N$.
`nessie -I path/input/file -O path/output/file -A -k n -K N`

To limit the search to a sub-string of the sequence it is possible to use **-b** *B* to define the starting index of the interval and **-e** *E* to define the ending index of the interval. *B* and *E* are positive integers.

Analysis of the sequence linguistic complexity and entropy

The program allows to perform an analysis of the sequence complexity and entropy. The linguistic complexity and the Shannon entropy are used respectively as measures.

This is a basic command line that allows to calculate the sequence complexity (**-L**) or entropy (**-E**) for the entire sequence.
`nessie -I path/input/file -O path/output/file {-E | -L}`

To use a sliding window of given length and shift instead of the entire sequence the **-l** *L* parameter can be used to define the window length while the **-s** *S* parameter can be used to define the window shift. *L* and *S* are positive integers.

`nessie -I path/input/file -O path/output/file {-E | -L} -l L -s S`

It is possible to define a range for the length of the k-mers that are used for the calculation of the sequence complexity. To consider only k-mers of minimum length *n* the parameter **-k** *n* can be used. To consider only k-mers of maximum length *N* the **-K** *N* parameter can be used. To consider only k-mers in a range of length [*n*...*N*] the **-k** *n* and the **-K** *N* parameters can be combined.
`nessie -I path/input/file -O path/output/file -L [-l L -s S] [-k n -K N]`

***note**: by default k-mers in range [1..20] are used for the complexity calculation. If **-K** N is used, also **-k** n must be specified*.

To limit the search to a sub-string of the sequence it is possible to use **-b** *B* to define the starting index of the interval and **-e** *E* to define the ending index of the interval. *B* and *E* are positive integers.

Exact-motifs search

The program allows to perform a search for exact-motifs in the sequence.

This is a basic command line that allows to search for the motifs in `path/file/w/motifs` in the sequence. Both the input files must be in *fasta* or *multi-fasta* format.
`nessie -I path/input/file -O path/output/file -N path/file/w/motifs`

To limit the search to a sub-string of the sequence it is possible to use **-b** *B* to define the starting index of the interval and **-e** *E* to define the ending index of the interval. *B* and *N* are positive integers.

License

Libraries

NeSSie uses the libraries:

- BitArray: https://github.com/noporpoise/BitArray
- BITSCAN https://github.com/psanse/bitscan