SISSIz Version 3.0 - Manual

Lorenz Perschy[1] and Tanja Gesell[1]

February 21, 2018

[1]Department for Computational Biology and Biomolecular NMR Spectroscopy, University of Vienna, 1030 Vienna, Austria

Contents

1 Set up

1.1 Installation

After having downloaded the SISSIz tar file open your terminal and go to your download directory.

```
tar -xzvf SISSIz.tar.gz # extract tar file
cd SISSIz
./configure # configure installation
make  # compile the source code
make install # install locally, for users without root rights
sudo make install # install as root user
```

This installs the SISSIz binary into /usr/local/bin and additional data files and examples in /usr/local/share/SISSIz.

If you have no root rights on your system or prefer to install SISSIz into a self-contained directory run configure for example like this:

```
./configure --prefix=/opt/programs/SISSIz --datadir=/opt/programs/SISSIz/share
```

The 'configure' shell script attempts to guess correct values for various system-dependent variables used during compilation. It uses those values to create a 'Makefile' in each directory of the package. It may also create one or more '.h' files containing system-dependent definitions. Finally, it creates a shell script 'config.status' that you can run in the future to recreate the current configuration, a file 'config.cache' that saves the results of its tests to speed up reconfiguring, and a file 'config.log' containing compiler output (useful mainly for debugging 'configure').

If you need to do unusual things to compile the package, please try to figure out how 'configure' could check whether to do them, and mail diffs or instructions to the address given in the 'README' so they can be considered for the next release. If at some point 'config.cache' contains results you don't want to keep, you may remove or edit it.

The file 'configure.in' is used to create 'configure' by a program called 'autoconf'. You only need 'configure.in' if you want to change it or regenerate 'configure' using a newer version of 'autoconf'.

1.2 Uninstall

To uninstall the program open your terminal and go to the SISSIz source code directory. Then run:

```
sudo make uninstall
```

You can remove the program binaries and object files from the source code directory by typing 'make clean'. To also remove the files that 'configure' created (so you can compile the package for a different kind of computer), type 'make distclean'. There is also a 'make maintainer-clean' target, but that is intended mainly for the package's developers. If you use it, you may have to get all sorts of other programs in order to regenerate files that came with the distribution.

1.3 Compiler Options

Some systems require unusual options for compilation or linking that the 'configure' script does not know about. You can give 'configure' initial values for variables by setting them in the environment. Using a Bourne-compatible shell, you can do that on the command line like this:

```
CC=c89 CFLAGS=-O2 LIBS=-lposix ./configure
```

Or on systems that have the 'env' program, you can do it like this:

```
env CPPFLAGS=-I/usr/local/include LDFLAGS=-s ./configure
```

1.4 Installation Names

By default, 'make install' will install the package's files in '/usr/local/bin', '/usr/local/man', etc. You can specify an installation prefix other than '/usr/local' by giving 'configure' the option '--prefix=PATH'.

You can specify separate installation prefixes for architecture-specific files and architecture-independent files. If you give 'configure' the option '--exec-prefix=PATH', the package will use PATH as the prefix for installing programs and libraries. Documentation and other data files will still use the regular prefix.

In addition, if you use an unusual directory layout you can give options like '--bindir=PATH' to specify different values for particular kinds of files. Run 'configure --help' for a list of the directories you can set and what kinds of files go in them.

If the package supports it, you can cause programs to be installed with an extra prefix or suffix on their names by giving 'configure' the option '--program-prefix=PREFIX' or '--program-suffix=SUFFIX'.

Further information can be found in `SISSIz`'s README file.

2 Using SISSIz

2.1 Description

SISSIz randomizes multiple sequence alignments while preserving the average dinucleotide content by using an adapted version of the SISSI (Simulating Site Specific Interactions, Gesell and von Haeseler (2006)). It uses a simulation procedure guided by a phylogenetic tree.

In combination with the RNAalifold algorithm (Hofacker et al. (2002), Bernhart et al. (2008)) it can be used to detect functional RNA structures in multiple alignments. To this end, SISSIz calculates the consensus folding energy of the original data and compares it to the consensus folding energy distribution of the random alignments. The significance of the prediction is measured as a z-score (Equation 1):

$$z = \frac{native_{MFE} - \mu_{MFE}}{\sigma_{MFE}} \tag{1}$$

Negative z-scores indicate secondary structures that are more conserved/stable than expected by chance.

Thus, SISSIz can be used in two ways, namely, the simulation of alignments and the prediction of RNA genes:

1. If SISSIz is used in simulation mode alignments are simulated with on average the same mono and di nucleotide content, Mean Pairwise Identity (MPI), local conservation and gap patterns. All of the mentioned statistics affect the Minimum Free Folding Energy (MFE) of the alignment's consensus secondary structure (Gesell and Washietl, 2008). SISSIz simulated alignments may be useful for creating negative control datasets for benchmarking, e.g. Gruber et al. (2010), or to estimate the FDR (False Discovery Rate) in ncRNA screens, e.g. Will et al. (2013).
2. When SISSIz is employed as an RNA gene finder the simulated alignments serve as a null model for the prediction as shown above. In genome wide screens the genomic alignments are split into alignments with a certain length and overlap (usually 200 and 100, respectively).

2.2 SISSIz Versions

Table 1 shows the history of SISSIz. The first release of SISSIz (Gesell and Washietl, 2008) used version 1.6.1 of RNAalifold and the Vienna RNA package (Lorenz et al., 2011), the second version of SISSIz (Smith et al., 2013), v.1.85 and the currently latest release (Perschy and Gesell, 2018), v.2.4.1. Since version 2.0 the ribosum matrix can be used in SISSIz as an alternative to the default covariation model. The third version of SISSIz added support for probing data in the prediction of ncRNAs and many other options.

Version	Publication	Source Code
1.0	Gesell and Washietl (2008)	https://github.com/wash/sissiz
2.0	Smith et al. (2013)	http://www.martinalexandersmith.com/ECS/SISSIz-2.tar.gz
3.0	Perschy and Gesell (2018)	XXXX

Table 1: Available SISSIz versions.

2.3 Usage

2.3.1 Input alignment

The input alignment must be in CLUSTAL W format or MAF format.

2.3.2 Program call

```
SISSIz [OPTIONS] [FILE]
```

2.3.3 Options

Most options have a long (--long) and a short form (-x). Table 2 provides an overview of all command line options.

Option	Description
`-s, --simulate`	This switches to "simulation only" mode, i.e no RNA analysis is carried out and the alignments is just randomized.
`-n, --num-samples`	Number of samples that are used for calculating the z-score. In simulation mode, the number of random alignments to be produced. Default is 100 for z-scores, and 1 for simulations.
`-i, --mono`	Choose mono-nucleotide background model. Default is the di-nucleotide model.
`-d, --di`	Choose di-nucleotide background model. Default is the di-nucleotide model
`-v, --verbose`	Produce verbose output that gives an overview of the algorithm, intermediate results and statistics.
`-t, --tstv`	Use a model that has different rates for transitions and transversions.
`-k, --kappa`	Set the transition/transversion rate ration parameter for the model. If not set, it is estimated from the data. If `--tstv` not set this option can be ignored.
`-p, --precision`	Limit the deviation of the mono-nucleotide content in the simulated alignments. The value given here is the maximum Euclidean distance between the mono-nucleotide frequencies in the original alignment and the simulated alignment. In other words, it is the square-root of the sum of the squared differences between the four frequencies of As, Cs, Ts, and Gs. For example if you set to 0.01 you will get almost exactly the same mono-nucleotide content in the simulations as in the original alignment. This makes it slower but more accurate.
`-m, --num-regression`	Pairwise alignments of 25 different distances are simulated to estimate the relationship between observed and genetic distances. For each of these points the average of X samples is used, where X is the number given here. Default is 10.

`-f, --flanks`	In most cases you can ignore this option. In the algorithm additional sites are used as "buffer sites" to overcome problems with saturation of mutations that make it impossible to estimate distances. Default is 1.5 x (number of sites in original alignment). If your simulation fails because of a negative logarithm you can raise this value.
`--dna, --rna`	Use T or U in the output alignments. Default is `--dna`, i.e Ts are printed.
`--clustal, --maf`	Output alignment format, either CLUSTAL W format (default) or MAF format. Only relevant in simulation mode (`--simulate`).
`-h, --help`	Print short overview of options.
`-V, --version`	Print version information.
`-b, --print-tree`	Print estimated tree, BIONJ-Tree into aln.tree.
`-a, --oldAliEn`	Use old alifold energies (with gaps) (default: new version)
`-j, --ribo`	Ribosome matrices of Alifold are used. Default: OFF
`-- shape`	Use SHAPE reactivity data to guide structure predictions =file1,file2,...
`--shapeMethod`	Specify the method how to convert SHAPE reactivity data to pseudo energy contributions =D[mX][bY]
`-y, --seed_with_pID`	Seed with process ID. Default: OFF
`-z, --print_seeds`	Print seed values to seed.txt. Default: OFF
`--read_seeds`	Read seeds from seed_in.txt file. Default: OFF
`--sci`	Calculate SCI. Default: OFF
`-h, --help`	Help screen
`-V, --version`	Print version

Table 2: **SISSIz command line options**

2.3.4 Output

Simulation
In simulation mode (`-s`) one or more alignments in CLUSTAL W or MAF format are printed. By default only one alignment is simulated. The number can be set using the parameter `-n`; note that the alignments are then printed in a concatenated stream where each alignment is delimited by a "CLUSTAL W (SISSIz 3.0 simulation)" line.

If `--verbose` is given lots of additional information is printed (these lines start with a # so that they can be filtered easily).

RNA gene finding
When using `SISSIz` as an RNA genefinder (without `-s` option) the results are by default given in one tab separated line. Each column corresponds to the following parameters (also fig. 1):

1. Model name (either "sissiz-mono" or "sissiz-di")

2. Name of the input file
3. Number of sequences in the input alignment
4. Length of alignment
5. Mean Pairwise Identity (MPI) of the input alignment
6. Average MPI of the sampled alignments.
7. Standard deviation of the MPIs of the sampled alignments
8. Structural Conservation Index (SCI)
9. GC-Content
10. RNAalifold consensus Minimum Free Energy (MFE) of the original alignment.
11. Average consensus MFE in the sampled alignments
12. Standard deviation of the consensus MFE in the sampled alignments
13. z-score calculated from 7. 8. and 9.

2.4 Scripts

2.4.1 window_SHAPE.R

The source code folder of `SISSIz` contains the R-Script `window_SHAPE.R` in the `scripts` directory. It was created with the idea of facilitating the incorporation of SHAPE data in genomic screens, since then not only the input alignment needs to be divided into overlapping windows but also the corresponding SHAPE files for each sequence in the alignment. **Usage:**

```
Rscript window_SHAPE
        <Input alignment file in FASTA format>
        <window size> <window overlap>
        <SHAPE file for 1st sequence>
        <SHAPE file for 2nd sequence>
        <SHAPE file for ...>
```

The parameters must be entered in one line and in the correct order without brackets <>. Firstly, the alignment file must be provided in FASTA format, secondly, the window size, thirdly the window overlap. From here on, the parameters become optional, i.e. it is also possible to divide the input alignment into windows without SHAPE data. The fourth parameter would provide the name of the SHAPE file corresponding to the first sequence in the alignment, the fifth parameter, the name of the SHAPE file for the the second sequence in the alignment and so on. It is not necessary to provide SHAPE files for all sequences in the alignment. The SHAPE file should contain a tab separated table with a header e.g. "Index \t nt_id \t 1m7_react \t 1m7_std_err"

The alignment windows and SHAPE windows are then printed to various files with the extensions _w_i.fasta and _w_i.txt, respectively, where i refers to the index of the window file. The FASTA alignment windows need to be converted to CLUSTAL prior to use with `SISSIz`.

2.5 Examples

The examples folder of the `SISSIz` source code directory provides a few sample alignments on which you can test `SISSSz`. In the following a few example `SISSSIz` runs are shown.

2.5.1 Simulating alignments

Default simulation

```
SISSIz --simulate rRNA.aln
```

The simplest command to create a simulated alignment with the same average mono and di nucleotide content as the input alignment.

Simulation with set sample size and transition/transversion model

```
SISSIz --simulate --tstv -n 1000 rRNA.aln
```

Simulate 1000 random alignments with the same average di nucleotide content and a transition/transversion model.

Simulating with the mono nucleotide model

```
SISSIz --simulate --mono --tstv -n 1000 rRNA.aln
```

The same as above with mono nucleotide model, i.e the simulated alignments have only the same average mono nucleotide content but not the same average di nucleotide content.

Simulation with more detailed output

```
SISSIz --verbose --simulate --mono --tstv -n 1000 rRNA.aln
```

The same as above, but with a more detailed output.

Tree output

```
SISSIz --verbose --simulate -b rRNA.aln
```

Simulate an alignment with verbose output and print the tree, that guided the simulation, to aln.tree file.

2.5.2 RNA gene finding

Default prediction

```
SISSIz rRNA.aln
```

The simplest command to obtain a z-score (last column of output line) for a given input alignment using the default covariance model and the di-nucleotide background model (see section 2.5.1). The z-score serves as the predictor variable and indicates the likelihood of an alignment containing structurally conserved RNAs, the more negative the value, the higher the likelihood. Figure 1 explains the whole output line of a `SISSIz` program call.

Using the ribosum matrix

```
SISSIz --ribo rRNA.aln
```

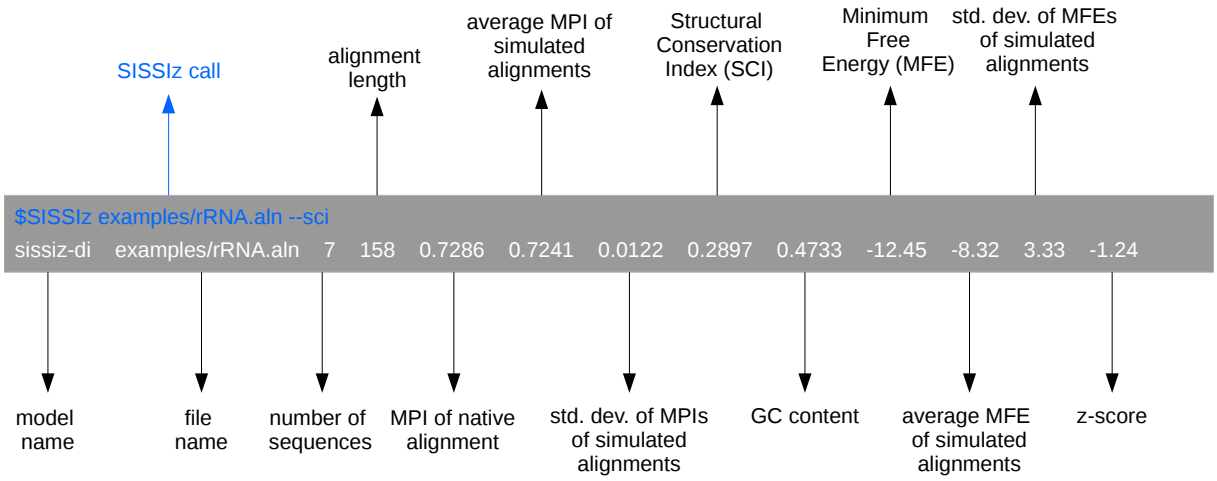

Figure 1: **SISSIz output example explained**

Calculate z-score using the ribosum covariance model of `RNAalifold`, which usually delivered better results than the default covariance model.

Using advanced sampling and higher precision

```
SISSIz --verbose --tstv --precision 0.05 -n 1000 rRNA.aln
```

Calculate z-score with a sample size of 1000. Slower but more accurate settings.

Output of SCI

```
SISSIz --ribo --sci rRNA.aln
```

Calculate z-score using the ribosum covariance model of `RNAalifold` and additionally calculate and output the SCI. By default the SCI is not calculated since it increases the runtime and therefore is printed as NA.

Using additional probing data for ncRNA prediction

```
SISSIz hiv_w_19.clu --shape=hiv_aug2013_w_19.txt,sivcpz_1m7_w_19.txt,
    sivmac_1m7_w_19.txt --shapeMethod="D"
```

Prediction guided by additional experimental SHAPE data (examples/SHAPE). The SHAPE files must be provided in the same order as the sequences in the alignment. Different reactivity to pseudo energy conversion parameters can be set. In this example "D" was chosen according to the approach of Deigan et al. (2009). The output of `SISSIz` retains the same format as in the above examples with the z-score printed in the last column too. For more information on how to incorporate SHAPE data (on genomic alignments) see section 2.4.1.

2.5.3 Advanced seeding options and reproduction of results

Since the simulation of alignments and as a result the z-score calculation occurs in a (pseudo) random fashion, that depends on a PRNG (Pseudo Random Number Generator), it is necessary

to provide certain seed values if results need to be reproduced. The examples shown below can thus be used for both simulation and RNA gene finding.

Using advanced seeding and seed output

```
src/SISSIz examples/rRNA.aln --simulate --print_seeds --seed_with_pID
```

Create a simulated alignment with the same average mono and di nucleotide content as the input alignment. Additionally, the process ID is used to seed the PRNG (necessary for parallel runs) and the seed values are printed to a seed.txt file which is automatically appended with every `SISSIz` call (one line corresponds to one `SISSIz` run).

Using advanced seeding and seed output

```
src/SISSIz examples/rRNA.aln --simulate --read_seeds=389650868,16063
```

Reproduction of simulated alignments with two seeds values taken from the seed.txt file. Using the same seed values will always produce the same simulated alignments and z-score, respectively. Hence, the reproduction of results is possible.

3 Literature

SISSIz and the new di-nucleotide randomization procedure: Gesell and Washietl (2008)

Basic idea of calculating stability z-score for multiple alignments: Washietl and Hofacker (2004)

The theoretical foundations for the simulation Model: Gesell and von Haeseler (2006)

RNAalifold algorithm: Hofacker et al. (2002)

Application of SISSIz-2.0 and RNAz-2.0 to multiple genome alignments of 35 mammal: Smith et al. (2013)

4 Source:

Adapted from SISSIz INSTALL file (probably `http://web.mit.edu/gnu/doc/html/autoconf_10.html`) and README file.

References

S. H. Bernhart, I. L. Hofacker, S. Will, A. R. Gruber, and P. F. Stadler. RNAalifold: improved consensus structure prediction for RNA alignments. *BMC Bioinformatics*, 9:474, Nov 2008. [PubMed Central:PMC2621365] [DOI:10.1186/1471-2105-9-474] [PubMed:19014431].

K. E. Deigan, T. W. Li, D. H. Mathews, and K. M. Weeks. Accurate SHAPE-directed RNA structure determination. *Proc. Natl. Acad. Sci. U.S.A.*, 106(1):97–102, Jan 2009. [PubMed Central:PMC2629221] [DOI:10.1073/pnas.0806929106] [PubMed:19109441].

T. Gesell and A. von Haeseler. In silico sequence evolution with site-specific interactions along phylogenetic trees. *Bioinformatics*, 22(6):716–722, Mar 2006. [DOI:10.1093/bioinformatics/bti812] [PubMed:16332711].

T. Gesell and S. Washietl. Dinucleotide controlled null models for comparative RNA gene prediction. *BMC Bioinformatics*, 9:248, May 2008. [PubMed Central:PMC2453142] [DOI:10.1186/1471-2105-9-248] [PubMed:18505553].

A. R. Gruber, S. Findeiss, S. Washietl, I. L. Hofacker, and P. F. Stadler. RNAz 2.0: improved noncoding RNA detection. *Pac Symp Biocomput*, pages 69–79, 2010. [PubMed:19908359].

I. L. Hofacker, M. Fekete, and P. F. Stadler. Secondary structure prediction for aligned RNA sequences. *J. Mol. Biol.*, 319(5):1059–1066, Jun 2002. [DOI:10.1016/S0022-2836(02)00308-X] [PubMed:12079347].

R. Lorenz, S. H. Bernhart, C. Honer Zu Siederdissen, H. Tafer, C. Flamm, P. F. Stadler, and I. L. Hofacker. ViennaRNA Package 2.0. *Algorithms Mol Biol*, 6:26, Nov 2011. [PubMed Central:PMC3319429] [DOI:10.1186/1748-7188-6-26] [PubMed:22115189].

Lorenz Perschy and Tanja Gesell. SISSIz in Shape. *In Preparation*, 2018.

M. A. Smith, T. Gesell, P. F. Stadler, and J. S. Mattick. Widespread purifying selection on RNA structure in mammals. *Nucleic Acids Res.*, 41(17):8220–8236, Sep 2013. [PubMed Central:PMC3783177] [DOI:10.1093/nar/gkt596] [PubMed:23847102].

S. Washietl and I. L. Hofacker. Consensus folding of aligned sequences as a new measure for the detection of functional RNAs by comparative genomics. *J. Mol. Biol.*, 342(1):19–30, Sep 2004. [DOI:10.1016/j.jmb.2004.07.018] [PubMed:15313604].

S. Will, M. Yu, and B. Berger. Structure-based whole-genome realignment reveals many novel noncoding RNAs. *Genome Res.*, 23(6):1018–1027, Jun 2013. [PubMed Central:PMC3668356] [DOI:10.1101/gr.137091.111] [PubMed:23296921].