

VarGibbs user manual

VarGibbs Version 1.0 user manual

Gerald Weber gweberbh@gmail.com

Departamento de Física, Universidade Federal de Minas Gerais, Brazil

July 25, 2014

This work is licensed under the Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-nc-sa/3.0/>.



CONTENTS

1	Introduction	3
2	Installation	5
2.1	What will be installed?	5
2.2	Specific instructions	5
2.2.1	OpenSUSE	5
2.3	Compiling the source files	5
2.4	Additional software	6
2.5	If things go wrong	6
2.5.1	If you don't have a Linux system	6
2.5.2	License	6
3	Usage	7
3.1	General options	7
3.1.1	-o=<basename>	7
3.1.2	-par=<filename>	7
3.1.3	-parid=<identifier>	7
3.1.4	-data=<filename>	7
3.1.5	-duplextype=<DNA or RNA>	7
3.1.6	-rs=<seed>	8
3.1.7	-salt=<salt concentration> (reserved for future use)	8
3.1.8	-ct=<species concentration>	8
3.1.9	-mct=<factor>	8
3.1.10	-dict=<list of nucleotides>	8
3.1.11	-calc=<type of calculation>	8
3.2	Specific options for temperature prediction (-calc=prediction)	8
3.2.1	-seq=<nucleotide sequence>	9
3.2.2	-cseq=<nucleotide sequence>	9
3.3	Options for minimization (-calc=minlocal or -calc=minglobal)	9
3.3.1	-minmethod=<GSL or GBC>	9
3.3.2	-var=<list of parameters>	9
3.3.3	-limit=<numerical value>	9
3.3.4	-cl=<numerical value>	9
3.3.5	-me=<integer value>	9
3.3.6	-mm=<integer value>	10
3.4	Options for global minimization (-calc=minglobal) only	10
3.4.1	-ee=<experimental error>	10
3.4.2	-randomizedataset=<0 or 1>	10
3.4.3	-randominitial=<fraction>	10
3.4.4	-mrr=<integer value>	10
3.4.5	-adl=<integer value>	10
3.4.6	-asl=<integer value>	10
3.4.7	-gn=<0 or 1>	11
4	Simple melting temperature prediction -calc=prediction	12
4.1	Files generated for prediction	12
4.1.1	Extension .dat sequence information and results	12
4.1.2	Extension .ver quality of the prediction and verification	12
4.1.3	Recalculating temperatures for different species concentrations	13

5	Obtaining parameters from minimization	14
5.1	Why would you want to do this?	14
5.2	What is the idea?	14
5.3	Global and local minimizations	14
5.4	Local minimization (<code>-cal=minlocal</code>)	14
5.4.1	Files generated for local minimization	15
5.5	Global minimizations (<code>-calc=minglobal</code>)	17
5.5.1	Varying the initial parameters (<code>-randominitial</code>)	17
5.5.2	Varying the experimental temperatures (<code>-randomizedataset</code>)	17
5.5.3	Files generated during global minimization	18
5.5.4	How to stop VarGibbs when in <code>-calc=global</code> mode?	19
5.5.5	Work flow suggestion for global minimization	19
6	NN model parameters	20
7	Parameter specification	21
7.1	Generic * parameters	21
7.2	Independent parameters	21
7.3	NN parameters	21
7.4	Parameter precedence	22
8	Files included in the package	23
8.1	Folder <code>/usr/share/data/</code>	23
8.1.1	Sequences and melting temperatures	23
8.1.2	Parameters	23
8.2	Folder <code>/usr/share/example/</code>	24
8.2.1	Local minimizations	24
8.2.2	Global minimizations	24
8.2.3	Temperature prediction	24
8.3	Folder <code>/usr/share/example/verify</code>	24

1 INTRODUCTION

VarGibbs provides the minimization method for calculating nearest-neighbour thermodynamic parameters. This is a new method and is presently not yet formally published in a peer-reviewed journal.¹

What will this software do for you? Given a set of experimental melting temperatures, measured for at any species concentration, it will calculate the relevant nearest-neighbour entropies and enthalpies, as well as initiation parameters. Once you have this parameters you may also use it to do simple melting temperature prediction.

The method works by minimizing predicted temperatures for the nearest-neighbour model in a very similar way as used previously by us for the Peyrard-Bishop model [1–3, 3].

Evidently, this is work in progress. The method is new and as such there may be aspects of it which are not yet completely understood.

I would find it truly helpful indeed if you would let me know if this software is of any use to you. Showing a list of interested users to funding agencies often helps to secure the necessary resources to keeping such projects running. So, please, if you find this software useful let me know and if you use it for your scientific work please cite the appropriate papers which are listed at the end of this manual.

I wish you all the best in using VarGibbs

Gerald

Belo Horizonte, July 25, 2014

¹A paper describing VarGibbs is presently under review.

CHANGES

Version 1.0, July 2014

First public release of VarGibbs.

2 INSTALLATION

The easiest way to install is to visit the OpenSUSE Build service (<http://build.opensuse.org>) and search VarGibbs for your Linux distribution, this should take care of obtaining the correct libraries which are needed for VarGibbs to run.

2.1 What will be installed?

Typically, there will be at least a binary executable file

`/usr/bin/vargibbs`

and further files, such as model parameter files, pre-calculated regression parameters and example scripts are to be found in

`/usr/share/vargibbs`

the documentation (which you are reading right now) should be located at

`/usr/share/doc/packages/vargibbs`

2.2 Specific instructions

2.2.1 OpenSUSE

Installing via repository Using the graphical interface Yast2 or the command line zypper add the following repository URL

`http://download.opensuse.org/repositories/home:/drgweber/openSUSE_12.2/`

then search for the package VarGibbs and select install. If you have an older OpenSUSE then change the last numbers to your installed version accordingly. The installation via repository has the advantage that you may simply update for future versions instead of repeating the whole installation procedure.

Command line download/install Download the appropriate package for your system from

`http://download.opensuse.org/repositories/home:/drgweber/openSUSE_12.2/`

for example if your system is 64bits, you may download the package

`http://download.opensuse.org/repositories/home:/drgweber/openSUSE_12.2/x86_64/VarGibbs-1.0-1.21.1.x86_64.rpm`

note: version numbers may vary from this example. Then install

`zypper install VarGibbs-1.0-1.21.1.x86_64.rpm`

2.3 Compiling the source files

Please read this section if you are unable to find the packages for your specific Linux distribution.

This software was developed and tested under OpenSUSE Linux 12.2 and depends on some libraries to function properly:

1. `libboost.filesystem1.49.0` `libboost.system1.49.0` <http://www.boost.org>
2. `gsl` <http://www.gnu.org/software/gsl/>
3. `libblas3` <http://www.netlib.org/lapack/>

which means that you will need at *least* these specialized packages in addition to the usual gcc and g++ compiler.

Download the source package from the OpenSUSE build service at

<https://build.opensuse.org/package/show?package=VarGibbs&project=home%3Adrgweber>

or from my personal webpage

<https://sites.google.com/site/geraldweberufmg/vargibbs>

Typically the package is called something like `vargibbs-1.0.tar.bz2` After unpacking the `tar` package

```
tar -xvjf vargibbs-1.0.tar.bz2
```

change into the unpacked folder

```
cd vargibbs-1.1
```

if all necessary packages are available you should try to compile using the `make` command

```
make
```

If the compilation is successful, you should see something like

```
g++ -O3 -o vargibbs -Isrc src/Options.cpp src/JobControl.cpp src/Nucleotide.cpp src/vargibbs.cpp -lz  
-lgsl -lgslcblas -lm -lboost_filesystem -lboost_system
```

and nothing else, that is it! This generates the binary file `vargibbs`, which you may copy to your main installation at `/usr/bin` (you will need root permission) or into your local folder `/home/user/bin` (replace `user` with you actual user name).

2.4 Additional software

Although not required for `vargibbs` to run, you may find useful to install GRI which is a scientific graphing language which can be found at

<http://gri.sourceforge.net/>

note that some Linux distributions include GRI already, for OpenSUSE try

```
zypper install gri
```

2.5 If things go wrong

Most problems will come from missing library packages or from erroneous usage of your system. It is not possible for me to cover everything that may go wrong, so please feel free to contact me. Please include a detailed description of error messages, which system you are using and a step by step description of what you tried to do. Please understand that I will need as much information as possible. I can do nothing with messages simply saying “VarGibbs is not working on Ubuntu”.

2.5.1 If you don't have a Linux system

You may copy an OpenSUSE Live system on a CD or onto a USB memory stick and reboot your computer with this system. If you do, proceed as you were using a normal OpenSUSE 12.2 system, following the instructions above. See installation instructions for Live OpenSUSE at http://en.opensuse.org/Live_USB_stick

2.5.2 License

This software is published under the GNU General Public License version 3 (GPLv3), the complete text of this license can be found in the documentation folder. If you wish to use this software under a different license or wish to make changes to the software without distributing it under the GPLv3 please contact me so that we can arrange for a specific license.

3 USAGE

VarGibbs takes all its program option from the command line, therefore you should invoke the binary `vargibbs` with some of the arguments which are described below. Example shell scripts are provided and I recommend you study them as they are the best source to illustrate how to use VarGibbs. The following section details every available program option for VarGibbs.

In this section we provide all options for VarGibbs. Please note that some options will only be used in conjunction with other options. Please see 5 on how to obtain new parameters.

3.1 General options

3.1.1 `-o=<basename>`

specifies the output basename, that is, all files which are generated start with `basename`

3.1.2 `-par=<filename>`

specifies the input parameter file name. This can be a list of files `-par=file1.par,file2.par,file3.par`. In case of multiple specifications of the same model parameter, the last one supersedes previous parameters. For example if `file1.par` has the parameter `AT_AT:enthalpy -10.6708` and `file3.par` has `AT_AT:enthalpy -10.4256`, the final value for `AT_AT:enthalpy` will be `-10.4256`.

3.1.3 `-parid=<identifier>`

this is an identification string which is added to newly generated parameter files. Its only function is to help with the organization of your files.

(default) `-parid=vargibbs`

3.1.4 `-data=<filename>`

specifies the input file containing nucleotide sequences and melting temperatures. VarGibbs was not created with very large datasets in mind, and all sequences are loaded into memory. A safe limit is of the order of 50000 sequences which will require about 1GB of memory to run.

The file format is composed of columns

```
1 temperature
2 ATCAATCATA TAGTTAGTAT 21.3 69 2
3 TTGTAGTCAT AACATCAGTA 24.7 69 2
4 GAAATGAAAG CTTTACTTTC 22.1 69 2
```

The first column is the sequence (from 5' → 3') and the second column is the secondary strand (from 3' → 5'). The third column is the melting temperature in °C, the fourth column is the salt concentration (in mM or mol/L). The last column the species concentration (in μM). The secondary strand does not necessarily need to be the complementary of the main strand as long as there are parameters for the mismatched pairs. If VarGibbs fails to find parameters for your sequences it will complain loudly.

3.1.5 `-duplextype=<DNA or RNA>`

Selects the type of duplex we should expect, this is important for selecting the base pair complementarity. Note that IUPAC codes cannot be used since we need to know which nucleotide parameters to use, that is, we cannot use N for example since we would not know which parameters to use.

(default) `-duplextype=DNA` , will expect A, C, G and T base pairs and will consider A complementary to T and C complementary to G.

`-duplextype=RNA` , same as for DNA but considers A and U as complementaries [4].

3.1.6 `-rs=<seed>`

Sets the seed of the random number generator (C function `srand`) which is used to modify the dataset. If you use the same seed you will get exactly the same random modifications.

3.1.7 `-salt=<salt concentration>` (reserved for future use)

At present this option is not used, as no salt correction is yet included with VarGibbs.

3.1.8 `-ct=<species concentration>`

Specifies the species concentration C_t given in 10^{-6} mol/L (or mM). Usually it is not necessary to provide this as this value should be contained in the dataset file, unless of course you would like a different concentration from what is given in the file. For files containing symmetry adjusted species concentration you may prefer to use `-mct`.

3.1.9 `-mct=<factor>`

Multiply the species concentration of all sequences by a certain factor. This is useful for files containing symmetry adjusted melting temperatures. For example for files where the concentration is 100 mM for self-complementary sequences and 400 mM otherwise.

3.1.10 `-dict=<list of nucleotides>`

When using non-canonical nucleotides, for example inosine, you must tell VarGibbs which characters to use in addition to its usual dictionary of A,T,C,G and U. You should also tell VarGibbs if there is a complementary pair to this new nucleotide. If there is none, simply repeat the character. In the following example we are adding the letter I for inosine

```
-dict=I:I
```

here we are saying: consider I and its complementary I as new letters. If you wish to add two or more new letters simply make a comma-separated list,

```
-dict=I:I,J:J
```

but do not add blank spaces after the comma.

3.1.11 `-calc=<type of calculation>`

This sets the type of calculation. See section 5 for details on how to perform minimizations.

(default) `-calc=prediction` performs melting temperature predictions for given set of parameters.

`-calc=minlocal` performs one local minimization

`-calc=minglobal` performs several local minimizations by varying either the initial variables or by varying the dataset. This option may take a *very* long time to run depending on the stop .

3.2 Specific options for temperature prediction (`-calc=prediction`)

Instead of providing a data file containing sequences you may wish to specify a single sequence for predicting its melting temperature. This is achieved with the options `-seq` and optionally `-cseq`. In this case you may wish to adjust also the value of species concentration with `-ct`.

3.2.1 -seq=<nucleotide sequence>

You should give the main strand from 5' to 3', the complementary sequence will be worked out automatically (see also `-dict` for non-canonical nucleotides).

(example) `-seq=ACGTTGAATT`

3.2.2 -cseq=<nucleotide sequence>

You should provide a 3' → 5' sequence if your sequence is not perfectly complementary, say like in a sequence with nucleotide mismatches. Note that you also need to provide the necessary parameters for this.

(example) `-cseq=TGCTACTTAA` from 3' → 5'

3.3 Options for minimization (`-calc=minlocal` or `-calc=minglobal`)

3.3.1 -minmethod=<GSL or GBC>

This selects which algorithmic implementation of multidimensional minimization to use. Currently GSL (GNU Scientific Library) produces better results.

(default) `-minmethod=GSL` GNU Scientific Library functions

`-minmethod=GBC` C++ textbook-like implementation

3.3.2 -var=<list of parameters>

The list of parameters for the minimization is given as in the following example

`-var=AT_AT:entropy,AT_CG:entropy,AT_GC:entropy,AT_TA:entropy`

you should use the same notation as in your `.par` file. See section 6 for a full description of available nearest neighbour parameters.

3.3.3 -limit=<numerical value>

Specifies when to give up minimizing. When your local minimization fails to improve with a value larger than `-limit` the minimization should stop. If you specify a very small value your minimization may take a very long time to complete. Note that the minimization will stop earlier depending on the value of `-me`.

(default) `-limit=0.01`

3.3.4 -cl=<numerical value>

This is the characteristic length used in the multidimensional minimization. Essentially, a larger `-cl` means that a wider region will be covered for minimization and potentially avoiding local minima. On the other hand, a larger value will also mean longer processing times. The exact value is not critical but should typically be of the order of magnitude of the variables listed in `-var`. For a better understanding of this parameter please see numerical computation text book on downhill simplex methods [5].

(default) `-cl=10`

3.3.5 -me=<integer value>

Maximum evaluations per minimization round. If this number is reached before the limit `-cl`, the minimization will stop.

(default) `-me=<a really large integer>` in other words: do not stop. Implemented by C++ method `std::numeric_limits<int>::max()`

3.3.6 -mm=<integer value>

This controls the number of minimization steps. Each round stops when either -c1 or -me is reached. The minimization then immediately starts over, using the previous optimized parameters as new initial variables.

(default) -mm=2 , in other words, do at least one round.

3.4 Options for global minimization (-calc=minglobal) only

3.4.1 -ee=<experimental error>

This sets the experimental error (in °C) of the melting temperature set. If given, the data set (provided through the -data option) will be modified by small positive or negative amounts such that the standard deviation is close to the experimental error. See also -randomizedataset.

(example) -ee=0.5 will modify the dataset to within 0.5 °C

3.4.2 -randomizedataset=<0 or 1>

When -randomizedataset=1 is selected the randomization of the dataset is activated. The dataset given with -data=<filename> will be changed by small amounts such that the standard deviation from the original dataset equals -ee=<experimental error>. See section 5 for details.

(default) -randomizedataset=0 do not modify dataset

3.4.3 -randominitial=<fraction>

When given the initial values of the variables specified in the file given in -par will be randomly modified. For example

-randominitial=0.2

means that all initial values will be modified within $\pm 20\%$ every time a new local minimization starts. For example if your parameter file contains

GC.CG:enthalpy -11.1

and this parameter is listed in -var a value of -randominitial=0.2 will randomly choose a new initial value for GC.CG:enthalpy within -8.88 and -13.32. This is applied to all variables listed in -var, but only to those variables.

3.4.4 -mrr=<integer value>

Controls how many complete rounds of local minimizations are performed.

(default) -mrr=10 which is a relatively small number and should run in a couple of minutes. A more thorough search should use values larger than 100 which may take several hours to run.

3.4.5 -adl=<integer value>

Stops the minimization if the limit for total average difference given in -adl is reached. VarGibbs will compare the results of the current minimization with the previous one and compute the average differences of the parameters. A value such as -adl=0.1 means that the minimization will stop if on average the differences between the previous parameters and the new parameters is less than 1%.

3.4.6 -asl=<integer value>

Similar to -adl but considers the standard deviation instead of the average difference. In this sense -asl=0.1 is a much more stricter stop criteria than -adl=0.1.

3.4.7 `-gn=<0 or 1>`

Controls the generation of files for `-calc=minglobal`. Global minimization generates a very large number of files, easily reaching several Gb. If you are not interested in those files you may turn this option off `-gn=0`. Note that if afterwards you change your mind you may simply run again and get exactly the same results as long as you use the same seed for the random number generator `-rs`.

(default) `-gn=1` , generate all files.

4 SIMPLE MELTING TEMPERATURE PREDICTION -CALC=PREDICTION

This is the simplest application of VarGibbs. In the following example we show how to obtain the melting temperatures from the SantaLucia paper Ref. [6] with all sequences from the same paper:

```
examples/D-SL98-P-SL98.sh
1 vargibbs -calc=prediction -o=./D-SL98-P-SL98 -par=./data/P-SL98.par \
2 -data=./data/D-SL98.dat
```

Instead of a data file with sequences you may simply provide the sequence info directly in the command line

```
examples/sequence-AOP-CMB.sh
1 vargibbs -o=./sequence-AOP-CMB -par=./data/AOP-CMB.par \
2 -calc=prediction -v=1 -seq=ACGTTTAAAGGTTTC -cseq=TGCAAAATCCAAG -ct=200
```

All results are collected in the file with extension .dat described in the next section.

4.1 Files generated for prediction

4.1.1 Extension .dat sequence information and results

The file with extension .dat shows all sequences (format of this file is identical with extension .ver of local/global minimization).

```
examples/verify/D-SL98-P-SL98.dat
1 Main/Complementary alpha salt_concentration species_concentration temperature.measured ...
2 CCGG/GGCC 1 1000 100 16.6 16.6 12.8709 0 0 -26.4 0 0 -74 0 0 0
3 CGCG/GCGC 1 1000 100 23.7 23.7 22.7167 0 0 -30.8 0 0 -85.8 0 0 0
4 AGCCG/TCGGC 4 1000 400 35.6 35.6 35.5231 0 0 -33.8 0 0 -91.2 0 0 0
5 CACAG/GTGTC 4 1000 400 19.1 19.1 19.6603 0 0 -33 0 0 -94.4 0 0 0
6 ACCGCA/TGGCGT 4 1000 400 43.1 43.1 48.0789 0 0 -40.7 0 0 -108.4 0 0 0
7 AGTTGC/TCAACG 4 1000 400 37.3 37.3 35.2518 0 0 -40 0 0 -111.4 0 0 0
8 ATGCGC/TACGCG 4 1000 400 44.5 44.5 46.4657 0 0 -43.5 0 0 -117.8 0 0 0
9 CCAACG/GGTTGC 4 1000 400 37.7 37.7 39.2123 0 0 -43.2 0 0 -120 0 0 0
10 CCGCGG/GGCGCC 1 1000 100 55.2 55.2 52.0737 0 0 -46.8 0 0 -125.6 0 0 0
```

The first column are the sequences, the second column indicates if the sequence is self-complementary (4) or not (1). Then comes the salt concentration (1000) in mM, the species concentration (100 or 400) in μ M, the measured melting temperatures (columns 5 and 6), the predicted melting temperature (column 7). The predicted total enthalpy and total entropy variations are given in columns 10 and 13, respectively. The remaining columns (all with null values) are not used in the context of VarGibbs and maintained for compatibility with other programs.

For the case when a single sequence was given this file will look like this

```
examples/verify/sequence-AOP-CMB.dat
1 Main/Complementary alpha salt_concentration species_concentration temperature.measured ...
2 ACGTTTAAAGGTTTC/TGCAAAATCCAAG 4 0 200 0 0 65.0278 0 0 -90.868 0 0 -249.021 0 0 0
```

4.1.2 Extension .ver quality of the prediction and verification

```
examples/verify/D-SL98-P-SL98.ver
1 1.88066 1.57065 648.413 2.45027
```

where the first number average difference in melting temperature prediction

$$\langle \Delta T \rangle = \frac{1}{N} \sum_{i=1}^N |\Delta T_i|, \quad (4.1)$$

the second is the standard deviation of ΔT_i ,

$$\delta(\Delta T) = \sqrt{\frac{1}{N} \sum_{i=1}^N [\Delta T_i - \langle \Delta T \rangle]^2} \quad (4.2)$$

the third is χ^2 given by Eq. (5.1), and the last is

$$\Delta T_{RMS} = \sqrt{\frac{1}{N} \sum_{i=1}^N [\Delta T_i]^2} = \sqrt{\frac{\chi^2}{N}}. \quad (4.3)$$

4.1.3 Recalculating temperatures for different species concentrations

Sometimes you may need melting temperatures for different species concentration C_t . In this case the options `-ct` and `-mct` may be useful. The option `-ct` defines a new C_t common to all sequences, while `-mct` multiplies C_t by a certain factor. The second option is useful for files containing symmetry adjusted melting temperatures, where self-complementary sequences are 1/4 of the C_t of other sequences.

5 OBTAINING PARAMETERS FROM MINIMIZATION

5.1 Why would you want to do this?

You may wish to apply VarGibbs in the following practical situations:

1. You have a number of melting temperatures, measured at several species concentrations and you need the best nearest-neighbour enthalpies and entropies.
2. You have a number of melting temperatures, and you do not have the total entropies and enthalpies and you need the best nearest-neighbour enthalpies and entropies.
3. You already have the nearest-neighbour enthalpies and entropies from other methods, but wish to obtain a new set with hopefully better predictive capabilities.
4. You already have the nearest-neighbour enthalpies and entropies from other methods and would like to estimate the uncertainty of these parameters.

5.2 What is the idea?

The basic idea behind the implementation of VarGibbs is to minimize

$$\chi^2 = \sum_i [T_i - T'_i(P_k)] \quad (5.1)$$

where T_i are the measured melting temperatures for the i th sequence, and T'_i are the predicted temperatures according a set of parameters P_k . We start with an initial set of parameters P_0 and let the parameters P_k vary until we minimize as much as possible χ^2 . After N steps of minimizations the resulting set P_N represent the parameters which best reproduce the melting temperatures for model M .

5.3 Global and local minimizations

VarGibbs knows about two types of minimizations, global and local. In the following sections we will describe these in details, but essentially a global minimization is a number of local minimization. We will first describe the local minimization which is easier to understand and introduce the main concepts. Figure 5.1 shows an overview of the two types on minimizations and some of the main options related to each step.

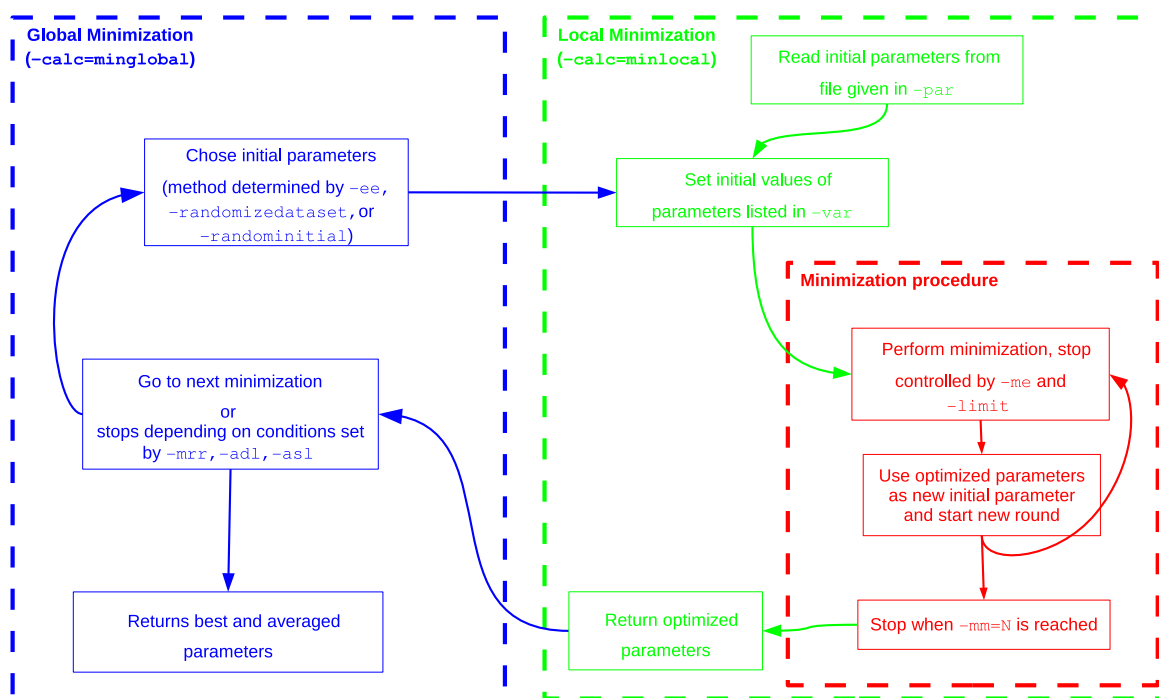
5.4 Local minimization (`-cal=minlocal`)

The local minimization is to use the unmodified T_i experimental melting temperatures and a set of initial parameters P_0 which are read from the parameter file given by `-par`. An example command line would be

```
vargibbs -var=AT_AT:entropy,AT_CG:entropy,AT_GC:entropy,AT_TA:entropy -data=experimental.dat \
-calc=minlocal -o=results -par=test.par
```

The quality of results will depend very much on your initial guess of variables in file `test.par`. The reason for this are local minima, that is the function χ^2 has numerous minima and your minimization may get “stuck” at non-optimal values. One solution, which *may* yield acceptable results is to carefully select generic initial parameters. In the following example this is achieved by a parameter file like this

```
1 init-s98
2 *:entropy -23
3 *:enthalpy -8
4 terminal_CG_enthalpy 2.0
```



S

Figure 5.1

Workflow and dependencies of global and local minimizations in VarGibbs.

```

5 | terminal_CG_entropy -5.0
6 | terminal_AT_enthalpy 5.0
7 | terminal_AT_entropy 10.0
8 | symmetry_correction_entropy -1.0

```

where we selected an entropy of -23 and an enthalpy of -8 for any nearest-neighbour context.

The complete example script, which should take about 20 s to run, is

```

1 | vargibbs -cl=10 -me=3000 -minmethod=gs1 -mm=3 -o=./SL98-local \
2 | -par=../data/init_s98.par \
3 | -parid=test -calc=minlocal -data=../data/D-SL98.dat \
4 | -var=AT_AT:entropy,AT_CG:entropy,AT_GC:entropy,AT_TA:entropy,CG_AT:entropy,CG_CG:entropy,...
5 |
6 | vargibbs -o=./SL98-local-prediction -par=./SL98-local.par \
7 | -calc=prediction -data=../data/D-SL98.dat

```

lines 6–7 calculate the prediction from the optimised parameter file `SL98-local.par` which has the following quality of the prediction file

```

1 | 1.50292 1.1753 393.129 1.9079

```

see Sec. 4.1.2 for details about these numbers.

5.4.1 Files generated for local minimization

Extension .dat: melting temperatures and melting index results

One of the main result files has the extension `.dat` and contains the optimized parameters. However these files are not easy to read, you may prefer to inspect the parameter file `.par` instead.

Here is an output of the previous example:

```

1 | 393.129 -7.74456 -21.9304 -10.2102 -28.7311 -6.27733 -16.4984 -3.29831 -8.39649 ...

```

The first number is the final χ^2 , the remaining numbers are the optimized parameters which were given in `-var` (in lexical order).

Extension *.par*: reusable parameter file

The optimized parameters are also stored in a file with extension *.par* in addition to initial generic parameters

examples/verify/SL98-local.par

```
1 test
2 *:enthalpy -8
3 *:entropy -23
4 AT_AT:enthalpy -7.74456
5 AT_AT:entropy -21.9304
6 AT_CG:enthalpy -10.2102
7 AT_CG:entropy -28.7311
8 AT_GC:enthalpy -6.27733
9 AT_GC:entropy -16.4984
10 AT_TA:enthalpy -3.29831
11 AT_TA:entropy -8.39649
12 CG_AT:enthalpy -11.1112
13 CG_AT:entropy -31.3799
14 CG_CG:enthalpy -8.56919
15 CG_CG:entropy -21.8905
16 CG_GC:enthalpy -9.64746
17 CG_GC:entropy -24.6633
18 GC_AT:enthalpy -8.04907
19 GC_AT:entropy -21.867
20 GC_CG:enthalpy -11.626
21 GC_CG:entropy -30.7114
22 TA_AT:enthalpy -3.04064
23 TA_AT:entropy -8.19177
24 symmetry_correction_entropy -0.292102
25 terminal_AT_enthalpy 2.17672
26 terminal_AT_entropy 4.50153
27 terminal_CG_enthalpy 1.79407
28 terminal_CG_entropy 3.35923
29
```

This parameter file can be reused in other minimization rounds or for the global minimizations discussed in the next section.

Extension *.log*: minimization history

The file *.log* records the complete minimization history and therefore is quite extensive.

examples/verify/SL98-local.log

```
100 192 3536.39 -8.07002 -23.1125 -8.29075 -21.5581 -8.31565 -22.2659 -7.89482 -22.6483 ...
101 193 3536.39 -8.07002 -23.1125 -8.29075 -21.5581 -8.31565 -22.2659 -7.89482 -22.6483 ...
102 195 3527.44 -7.98539 -23.0308 -8.16344 -21.5478 -8.1843 -22.1835 -7.80971 -22.5493 ...
103 197 3110.49 -8.14102 -22.9354 -8.32862 -21.4805 -8.34865 -22.0992 -7.7504 -22.4574 ...
104 198 3110.49 -8.14102 -22.9354 -8.32862 -21.4805 -8.34865 -22.0992 -7.7504 -22.4574 ...
105 200 3110.49 -8.14102 -22.9354 -8.32862 -21.4805 -8.34865 -22.0992 -7.7504 -22.4574 ...
106 201 3110.49 -8.14102 -22.9354 -8.32862 -21.4805 -8.34865 -22.0992 -7.7504 -22.4574 ...
107 203 2753.13 -8.18274 -23.1518 -8.22241 -21.2686 -8.26537 -22.2213 -7.90655 -22.8356 ...
108 205 2753.13 -8.18274 -23.1518 -8.22241 -21.2686 -8.26537 -22.2213 -7.90655 -22.8356 ...
109 206 2753.13 -8.18274 -23.1518 -8.22241 -21.2686 -8.26537 -22.2213 -7.90655 -22.8356 ...
110 208 2753.13 -8.18274 -23.1518 -8.22241 -21.2686 -8.26537 -22.2213 -7.90655 -22.8356 ...
```

The first column is the number of evaluation steps which stops at a maximum defined by *-mm*. The remaining columns are as in the file with extension *.dat*. Please note that the file ending with *.log.best* is always empty.

Extension *.ver* quality of the prediction and verification

The file with extension *.ver* shows all sequences .

examples/SL98-local.ver

```
221 AAAAAAAAA/TTTTTTTTT 4 1000 400 39.4 39.4 38.6545 0 0 -57.603 0 0 -166.44 0 0 0
222 ATAACGGC/TATTGACCG 4 1000 400 54 54 51.8572 0 0 -57.9065 0 0 -159.869 0 0 0
```

223	ATCTATCCG/TAGATAGGC	4	1000	400	51.1	51.1	52.1312	0	0	-46.2586	0	0	-123.91	0	0	0
224	CAAAAAAAG/GTTTTTTTC	4	1000	400	44.2	44.2	42.3168	0	0	-60.2677	0	0	-172.742	0	0	0
225	CAAACAAAG/GTTTGTTTC	4	1000	400	47.3	47.3	45.7217	0	0	-66.0999	0	0	-188.992	0	0	0
226	CAAAGAAAG/GTTTCTTTC	4	1000	400	45.1	45.1	45.3936	0	0	-59.105	0	0	-167.247	0	0	0
227	CAAATAAAG/GTTTATTTTC	4	1000	400	41.4	41.4	38.9791	0	0	-51.1175	0	0	-145.469	0	0	0
228	CGCTGTTAC/GCGACAATG	4	1000	400	54.2	54.2	54.2851	0	0	-66.2793	0	0	-184.119	0	0	0
229	GCCAGTTAA/CGGTCAATT	4	1000	400	52.5	52.5	52.1057	0	0	-62.3528	0	0	-173.403	0	0	0
230	AAAAAAAAAA/TTTTTTTTTT	4	1000	400	40.9	40.9	43.0411	0	0	-65.3476	0	0	-188.37	0	0	...

The first column are the sequences, the second column indicates if the sequence is self-complementary (4) or not (1). Then comes the salt concentration (1000) in mM, the species concentration (100 or 400) in μM , the measured melting temperatures (columns 5 and 6), the predicted melting temperature (column 7). The predicted total enthalpy and total entropy variations are given in columns 10 and 13, respectively. The remaining columns (all with null values) are not used in the context of VarGibbs and maintained for compatibility with other programs.

5.5 Global minimizations (-calc=minglobal)

One way to overcome the problem of local minima is to perform several local minimization. The option `-calc=minglobal` automates this process, see Figure 5.1 for an overview and how local and global minimization relate to each other. There are two basic global minimization strategies, one which varies the initial parameters randomly and the other varies the experimental data by small random amounts.

5.5.1 Varying the initial parameters (-randominitial)

The idea here is to start the local minimization anew each time with a different set of initial parameters $P_{0,l}$, where l is the index for a given set of initial parameters. VarGibbs will then start the minimization by randomizing the initial parameters. In the next example we modified the example of the previous section by changing `-calc=minglobal` and `-randominitial=0.2`, which modifies the initial parameters by up to 20%.

```
vargibbs -var=AT_AT:entropy,AT_CG:entropy,AT_GC:entropy,AT_TA:entropy -data=experimental.dat \
-calc=minglobal -o=results -randominitial=0.2
```

VarGibbs will carry on repeating the minimization over and over, every time for a new set of initial parameters $P_{0,l}$ until you tell VarGibbs to stop or until the maximum number of minimizations given by `-mrr` is reached.

VarGibbs will show a standard output when variables are changed:

```
Warning:replacing AT_AT:enthalpy old value=-9.3522 new value=-8.41043
Warning:replacing AT_AT:entropy old value=-26.487 new value=-18.8472
```

A complete example which should take few minutes to run:

```
examples/SL96-global.sh
1 vargibbs -calc=minglobal -cl=10 -data=./data/D-SL96.dat -gn=1 \
2 -me=3000 -minmethod=gs1 -mm=6 -mrr=10 \
3 -o=./SL96-global/BOP-SL96 -par=./data/init_s96.par \
4 -parid=BOP-SL96 -randominitial=0.5 -randomizedataset=0 -rs=1234 \
5 -var=AT_AT:entropy,AT_CG:entropy,AT_GC:entropy,AT_TA:entropy,CG_AT:entropy,\
6 CG_CG:entropy,CG_GC:entropy,GC_AT:entropy,GC_CG:entropy,TA_AT:entropy,\
7 AT_AT:enthalpy,AT_CG:enthalpy,AT_GC:enthalpy,AT_TA:enthalpy,CG_AT:enthalpy,\
8 CG_CG:enthalpy,CG_GC:enthalpy,GC_AT:enthalpy,GC_CG:enthalpy,TA_AT:enthalpy,\
9 initiation_CG_entropy,initiation_AT_entropy,\
10 terminal_5TA3_enthalpy,symmetry_correction_entropy
```

5.5.2 Varying the experimental temperatures (-randomizedataset)

This type of minimization should be carried out once you have a fairly good set of optimized parameters, therefore we do not recommend to use this type of minimization as your first attempt. This step is mainly used to evaluate the influence of the reported experimental uncertainty over the calculated parameters.

Here, we use a fixed set of initial parameters P_0 , as in the simple minimization, but for every round of minimization we slightly change the experimental melting temperatures T_i by random amounts ΔT_i . The standard deviation between the original set and the new artificial set will be close to the specified experimental error given in `-ee`.

```

12      examples/SL96-global.sh
13  vargibbs -asl=0.1 -calc=minglobal -cl=10 -data=./data/D-SL96.dat -gn=1 -ee=0.3 ...
14  -o=./SL96-global/AOP-SL96 -par=./SL96-global/BOP-SL96.par.best -parid=AOP-SL96 \
15  -randomizedataset=1 -rs=1234 \
16  -var=AT_AT:entropy,AT_CG:entropy,AT_GC:entropy,AT_TA:entropy,CG_AT:entropy,\
17  CG_CG:entropy,CG_GC:entropy,GC_AT:entropy,GC_CG:entropy,TA_AT:entropy,\
18  AT_AT:enthalpy,AT_CG:enthalpy,AT_GC:enthalpy,AT_TA:enthalpy,CG_AT:enthalpy,\
19  CG_CG:enthalpy,CG_GC:enthalpy,GC_AT:enthalpy,GC_CG:enthalpy,TA_AT:enthalpy,\
20  initiation_CG_entropy,initiation_AT_entropy,\
  terminal_5TA3_enthalpy,symmetry_correction_entropy

```

here we are using as initial parameters the best optimization

AOP-SL96.par.best

from the example in the previous section.

5.5.3 Files generated during global minimization

For global minimization you will get the same files for local minimizations listed in section 5.4.1 but appended by a number like .dat.00013.

```

      list of files for multiple minimization
BOP-SL96.dat.00003  BOP-SL96.reg.00003      BOP-SL96.ver.00003
BOP-SL96.par.00003  BOP-SL96.var.00003
BOP-SL96.dat.00004  BOP-SL96.reg.00004      BOP-SL96.ver.00004
BOP-SL96.par.00004  BOP-SL96.var.00004

```

The number indicates the local minimization round, otherwise these files are the same as for the simple minimization files described in section 5.4.1. The multiple minimization produces a very large number of files, therefore is it recommended to run in a separate folder. Alternatively, one may set `-gn=0` which will suppress the output of the files appended by numbers.

The additional extension `.best` are the results for the lowest χ^2 found, and extension `.par.average` contains the average parameter values. These files are updated at every round.

One file here is of special importance, the one ended with extension `.dat` will contain all results together.

```

      examples/verify/SL96-global/BOP-SL96.dat
1  529.123 -7.78225 -22.0386 -6.69507 -17.386 -2.52987 -4.87858 -0.30404 1.3981 -8.31749 ...
2  526.166 -9.85229 -27.997 -9.83689 -26.2417 -5.23629 -12.4947 -3.739 -8.55061 -8.56265 ...
3  525.7 -8.34932 -23.7104 -8.19589 -21.7873 -3.99421 -9.25101 -3.63315 -8.67596 -7.08778 ...
4  522.633 -7.67022 -21.7605 -9.56615 -26.0226 -6.01695 -15.4755 -7.06485 -18.9828 ...
5  525.7 -9.56528 -27.1325 -10.0257 -26.703 -5.94954 -14.6551 -5.38924 -13.3807 -8.05901 ...
6  526.799 -9.14595 -26.0312 -8.13573 -21.4327 -3.76715 -8.42463 -2.46223 -5.02819 ...
7  525.65 -9.84787 -28.0093 -9.37822 -24.8516 -4.95642 -11.7463 -3.838 -8.88293 -7.93385 ...
8  525.32 -7.23913 -20.438 -8.23796 -21.9436 -5.172 -12.886 -4.8005 -11.9916 -5.97586 ...
9  526.163 -10.8941 -31.0525 -10.0609 -26.9002 -3.17534 -6.13705 -3.37308 -7.63218 ...
10 525.294 -9.30893 -26.348 -10.0388 -26.796 -5.78086 -14.1626 -4.97548 -12.1637 -7.78641 ...

```

Where are your best results?

Your best results, that is, those with smallest χ^2 are easily accessible by locating the files ending in `.best`. Please note that the file `.log.best` will be empty as this file is written after the minimization has ended.

```

      example of list of files related to smallest  $\chi^2$ 
BOP-SL96.dat.best  BOP-SL96.par.best  BOP-SL96.reg.best  BOP-SL96.var.best  BOP-SL96.ver.best

```

Where are your average results?

It may be more sensible to work with average results as your may be overfitting your data. When using `-cal=minglobal` the average values are listed in the files ending with `.average`.

5.5.4 How to stop VarGibbs when in `-calc=global` mode?

If you did not specify the maximum rounds of minimization `-mrr`, or `-asl/-adl` then VarGibbs will run non-stop. You may analyse your intermediate results at any time and then stop VarGibbs when you feel that you have gathered enough data. Of course, when this happen you may simply kill the process if you like. But a cleaner way to do this is to set the `.proceed` file to 0. Suppose your results file start all with `test` (`-o=test`)

```
echo 0 > test.proceed
```

before restarting the next minimization VarGibbs will check this file, if it is 0 it will then stop and cleanly close all files.

5.5.5 Work flow suggestion for global minimization

- 1 Get the melting temperatures** either from a paper or format your own data according to section 3.1.4. Make sure you have the correct values for species concentration C_t in 10^{-6} mol/L (μM). You can mix data with different species concentrations, but not with different salt concentrations.
- 2 Define your initial parameters** , ideally start the the most generic values possible, such as in file `data/gibbs-av.par`.
- 3 Run global minimizations with varying initial parameters** . For this step follow the recommendations in section 5.5.1. Let it run for as many times as practically possible, but at least 100 times (2 h). It will take about two days if you let it run 1000. You may also monitor your results continuously as described in section 5.5.4, and then interrupt when you are satisfied.
- 4 Get the best parameters** as described in section 5.5.3.
- 5 Run global minimizations to obtain error estimates** as described in section 5.5.2. Use the best parameters from step 4 as initial parameters.
- 6 Use the averages** obtained in the last step (see 5.5.3 on how to get them). They should represent the data you are looking for.

6 NN MODEL PARAMETERS

There are basically two groups of parameters in the nearest neighbour model: context-dependent parameters and initiation parameters. The context-dependent parameters are the ones that give this model its name, they are dependent on the nearest neighbours. All other parameters are dependent on some specific features of a given sequence, for example a parameter which adds some amount to the entropy if there is an AT base pair at the end. These parameters are generally called initiation parameters and largely represent the idea that the melting may be initialized by the presence of one of these features. Table 6.1 summarizes the parameters known to VarGibbs.

Parameter	Type	Description
entropy	NN	the hybridization entropy for a given NN pair
enthalpy	NN	the hybridization enthalpy for a given NN pair
initiation_CG_entropy	init.	if the sequence contains a <i>at least</i> one CG base pair [7]
initiation_CG_enthalpy	init.	if the sequence contains a <i>at least</i> one CG base pair
initiation_AT_entropy	init.	if the sequence contains <i>only</i> AT (or AU) base pairs [7]
initiation_AT_enthalpy	init.	if the sequence contains <i>only</i> AT (or AU) base pairs
terminal_CG_enthalpy	init.	for each terminal CG base pair [6]
terminal_CG_entropy	init.	for each terminal CG base pair [6]
terminal_AT_enthalpy	init.	for each terminal AT base pair [6]
terminal_AT_entropy	init.	for each terminal AT base pair [6]
terminal_AU_enthalpy	init.	for each terminal AU base pair
terminal_AU_entropy	init.	for each terminal AU base pair
terminal_5TA3_enthalpy	init.	for each terminal 5'-TA-3' base pair [7]
symmetry_correction_entropy	init.	correction applied to self-complementary sequences [6,7]
symmetry_correction_enthalpy	init.	correction applied to self-complementary sequences
initiation_entropy	init.	a fixed correction applied to all sequences
initiation_enthalpy	init.	a fixed correction applied to all sequences

Table 6.1

Summary of parameters known to VarGibbs. Entropies are given in cal/mol and enthalpies in kcal/mol.

7 PARAMETER SPECIFICATION

This chapter explains how to write a parameter file.

Let's start with the most simple possible parameter file

```
1 gibbs
2 *:entropy -23.6
3 *:enthalpy -8.4
4 initiation_CG_entropy -5.9
5 initiation_AT_entropy -20
6 terminal_5TA3_enthalpy 0.4
7 symmetry_correction_entropy -20
```

The first line holds a simple identifier, this identifier can be letters, numbers and some other characters but should contain no spaces.

The second line starts with an asterisk which means that this parameter should apply to any base pair. In case of a DNA sequence this will apply to AT and CG base-pairs.

7.1 Generic * parameters

The generic base * specification can be used when a given parameter should be applied to any base pair (see previous example). However, if the parameter should be applied to a specific base pair you should specify either in BP or NN form (see next section).

7.2 Independent parameters

Independent parameters are those which do not depend on context changes of the sequences. Examples are `initiation_CG_entropy` and `terminal_5TA3_enthalpy`.

7.3 NN parameters

Nearest neighbour (NN) parameters follow the same conventions as usually found in linear regression models [6, 8]. The convention used is of type `AB_CD`, where `AB` is the first base pair and `CD` the second base pair. Note that `DC_BA` is equivalent to `AB_CD`, for example, `AT_AT` is the same as `TA_TA`. When specifying a NN sequence always specify in lexical ordering, that is write `AT_AT` and not `TA_TA`.

The following example shows all 10 irreducible N parameters for `finite_enthalpy.DeltaH`.

```
1 AT_AT:enthalpy -10.6708
2 AT_CG:enthalpy -11.6525
3 AT_GC:enthalpy -5.95646
4 AT_TA:enthalpy -7.89543
5 CG_AT:enthalpy -6.11506
6 CG_CG:enthalpy -10.4258
7 CG_GC:enthalpy -7.21118
8 GC_AT:enthalpy -5.35005
9 GC_CG:enthalpy -12.2045
10 TA_AT:enthalpy -2.12685
```

7.4 Parameter precedence

Since the program can read more than one parameter file, the last parameter read is the final value. There may also be several specification in the same file as well. Consider the following example

Precedence example 1

```
1 *:enthalpy -6.0
2 AT_AT:enthalpy -10.0
```

the first line says that all nearest-neighbours should a **enthalpy** value of -6.0. The second line however says that **AT_AT** nearest-neighbours should use -10.0. In this case **GC_AT** base pairs for instance will use -6.0 since nothing different was specified.

However, a generic nearest-neighbours ***** does not supersedes a specific nearest-neighbours as in the following example

Precedence example 2

```
1 AT_AT:enthalpy -10.0
2 *:enthalpy -6.0
```

in this case **AT_AT** nearest-neighbours will continue using -10.0, not -6.0. You should understand the generic nearest-neighbours ***** as: *if nothing else matches, use this value.*

8 FILES INCLUDED IN THE PACKAGE

VarGibbs comes with a number of files included.

8.1 Folder `/usr/share/data/`

This folder contains the input data such as parameter files (extension `.par`) and melting temperature data (extension `.dat`)

8.1.1 Sequences and melting temperatures

`D-SL96.dat` from Ref. 7

`D-SL98.dat` from Ref. 6 and supplementary data from Ref. 9

`D-OW04-[salt].dat` UV data from Ref. 10

`D-OW04DSC-[salt].dat` DSC data from supplementary material of Ref. 10. Note that `D-OW04DSC-1000.dat` uses uncorrected species concentrations.

`D-CMB.dat` combined dataset D-SL98, D-OW04-1024 and D-OW04DSC-1020.

8.1.2 Parameters

`gibbs-av.par` generic set of initial parameters using initiation parameters as in Ref. 7

`init_s96.par` set of initial parameters used to optimize D-SL96 data.

`init_s98.par` set of initial parameters used to optimize D-SL98 data.

`init_ow04.par` set of initial parameters used to optimize D-OW04 data.

`P-SL96.par` original parameters from Ref. 7

`P-SL98.par` original parameters from Ref. 6

`AOP-CMB.par`, `BOP-CMB.par` average optimized parameters (AOP) and best optimized parameters (BOP) generated from dataset D-CMB with parameters.

`AOP-OW04DSC.par` generated from dataset D-OW04DSC.

`BOP-OW04DSC.par` generated from dataset D-OW04DSC.

`AOP-OW04.par` generated from dataset D-OW04-1020.

`BOP-OW04.par` generated from dataset D-OW04-1020.

`AOP-SL96.par`, `BOP-SL96.par` generated from dataset D-SL96. See script `SL96-global.sh` for settings.

`AOP-SL98.par`, `BOP-SL98.par` generated from dataset D-SL98. See script `SL98-global.sh` for settings.

8.2 Folder /usr/share/example/

This folder contains example scripts. Processing times listed here are for an Intel(R) Xeon(R) CPU model E5310 1.60GHz.

To run these scripts on your machine proceed as follows:

1. locate the script, usually you will find them at /usr/share/VarGibbs/examples
2. locate the data files, usually you will find them at /usr/share/VarGibbs/data
3. run

```
/usr/share/VarGibbs/examples/D-SL98-AOP-SL98.sh /usr/share/VarGibbs/data
```
4. alternatively copy the script to your local folder and then run locally

```
cd /usr/share/VarGibbs/examples/D-SL98-AOP-SL98.sh  
./D-SL98-AOP-SL98.sh ./usr/share/VarGibbs/data
```

8.2.1 Local minimizations

SL96-local.sh for dataset D-SL96, processing time ≈ 1 min.

SL98-local.sh for dataset D-SL98, processing time: ≈ 1 min.

CMB-local.sh for D-CMB, processing time: ≈ 8 min.

8.2.2 Global minimizations

The following scripts contain complete global minimization procedures. They will take considerable time to run.

SL96-global.sh for dataset D-SL96, processing time: ≈ 16 min.

SL98-global.sh for dataset D-SL98, processing time: ≈ 28 min.

CMB-global.sh for dataset D-CMB, processing time: ≈ 1.5 h.

OW04-global.sh for all salt concentrations of dataset D-OW04, processing time: ≈ 4 h.

OW04DSC-global.sh for all salt concentrations of dataset D-OW04DSC, processing time: ≈ 1 h.

If you wish to reproduce the AOP and BOP files included in folder **data** then edit the scripts, locate and modify the following to lines to

```
export MRR=1000  
export ASL=0.001
```

see options **-mrr** and **-asl** for their meaning. Note that as the processing times will increase accordingly you may prefer using a dedicated desktop or server for this purpose.

8.2.3 Temperature prediction

D-SL98-P-SL98.sh self-prediction of dataset D-SL98 from parameters P-SL98.

D-SL98-AOP-SL98.sh self-prediction of dataset D-SL98 from parameters AOP-SL98.

D-CMB-AOP-CMB.sh self-prediction of dataset D-CMB from parameters AOP-CMB.

sequence-AOP-CMB.sh prediction of a single DNA sequence.

8.3 Folder /usr/share/example/verify

This folder contains all output data of the scripts include in this package. This allows you to verify if your local implementation is running correctly. If you run the scripts described here they should give you identical results to the ones in this folder.

Note that some files are gzipped to save space. To view them unzip as in the following example

```
gunzip AOP-SL96.log.00001.gz
```

BIBLIOGRAPHY

- [1] Weber, G., Essex, J. W., and Neylon, C. Probing the microscopic flexibility of DNA from melting temperatures. *Nature Physics* **5**, 769–773 (2009).
- [2] Weber, G. Finite enthalpy model parameters from DNA melting temperatures. *Europhys. Lett.* **96**, 68001 (2011).
- [3] Weber, G. Mesoscopic model parametrization of hydrogen bonds and stacking interactions of RNA from melting temperatures. *Nucl. Acids. Res.* **41**, e30 (2013).
- [4] Weber, G. Mesoscopic model parametrization of hydrogen bonds and stacking interactions of RNA from melting temperatures. *Nucl. Acids. Res.* **40**, 1 (2012).
- [5] Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. *Numerical Recipes in C*. Cambridge University Press, Cambridge, (1988).
- [6] SantaLucia, Jr., J. A unified view of polymer, dumbbell, and oligonucleotide DNA nearest-neighbor thermodynamics. *Proc. Natl. Acad. Sci. USA* **95**, 1460–1465 (1998).
- [7] SantaLucia, Jr., J., Allawi, H. T., and Seneviratne, P. A. Improved nearest-neighbour parameters for predicting DNA duplex stability. *Biochem.* **35**, 3555–3562 (1996).
- [8] Breslauer, K. J., Frank, R., Blocker, H., and Marky, L. A. Predicting DNA duplex stability from the base sequence. *Proc. Natl. Acad. Sci. USA* **83**, 3746–3750 (1986).
- [9] Allawi, H. T. and SantaLucia Jr, J. Thermodynamics and NMR of internal G·T mismatches in DNA. *Biochemistry* **36**, 10581–10594 (1997).
- [10] Owczarzy, R., You, Y., Moreira, B. G., Manthey, J. A., Huang, L., Behlke, M. A., and Walder, J. A. Effects of sodium ions on DNA duplex oligomers: Improved predictions of melting temperatures. *Biochem.* **43**, 3537–3554 (2004).