How to calculate ΔG and Tm values for exome probe regions using CoNVex?

Requirements:

You may require some or all of the following scripts, libraries or packages to calculate ΔG and Tm values. You may also use an alternative method for doing the same. This document describes the steps with the actual Unix commands within boxes. Please ignore the bits highlighted in red colour, which help you configure options related to the farm (Unix/Linux cluster). These may be specific to the Sanger Institute. Requirements include:

- Ensembl Perl API – to retrieve sequences from Ensembl for a given list of genomic (bait) regions
- UNAfold – to calculate ΔG and Tm by computationally hybridizing sequences in the fasta file
- Scripts (Perl/Unix shell) are bundled with the R package (/CoNVex/scripts folder)

Steps:

1) Generate a fasta file with bait sequences (this file can be split into multiple smaller files for parallel processing in the farm later in Step 2).
2) Use FastaRevComp.java for generating *rna.fa AND *revcomp.fa - This script splits the original fasta file into multiple fasta files for running them in parallel in the farm. Change values accordingly (example below).
3) Run the unafold_wrapper.sh (or unafold_farm_wrapper.sh) script to call and execute UNAfold (download/configure UNAfold – link is given below). Note: Please copy melt.pl to the local folder to make it easy. Change the script options to configure the number of (parallel) jobs to execute based on the number of fasta files (detailed examples below). This script outputs unix commands that will have to be run separately.
4) UNAfold output would created ~6 files for each run. These will end with .fa.65.ext, .fa.65.plot, .fa.asc, etc. Make sure each (farm) run (output in the *.out) and the output files are successful.
5) Run, Hybridization java program by amending the details of the above UNAfold output and fasta files.

Links

- Ensembl Perl API: http://www.ensembl.org/info/docs/api/
- UNAfold: http://mfold.rna.albany.edu/
- Unix/Perl scripts: 'scripts' folder within the CoNVex installation folder. getLibPath() shows its location

Step 1: Generate a fasta file with bait sequences

If you have the probe regions, you can get their sequences from Ensembl using its Perl API. There are several alternatives to create a fasta sequence file from a given set of genomic regions. This script uses the Ensembl Perl API and may need to be modified to specify the location of the API. Please edit the lines 'use lib /path/to/ensemble/api' in the script to specify your local Ensembl API location. Please use the extension '.fa' (not '.fasta') for fasta files.

```
perl Ensembl_GetSeq.pl <genomic_regions_input_file>
```

Step 2: Generate reverse complementary DNA and RNA sequences

The command, 'java FastaRevComp <options>', does the following:

- Reads a fasta file with DNA sequences – usually of probe regions and their sequences
- Generates RNA and reverse complemented sequences – to be used as input UNAfold RNA hybridization
- Splits it to <N> number of approximately equal parts – using -split_into AND -max_seq options
- Creates fasta files – <N> each (N=10 in the following example command): i.e. 10 RNA files and 10 reverse complemented files
- Saves them in the same folder in which the original DNA fasta file exists

Please make sure that all jar files bundled with CoNVex are in $CLASSPATH environment variable.

```
Shell#: bsub -q normal -R'select[mem>2000] rusage[mem=2000]' -M2000000 -o FRC.out -J
'FRC' java FastaRevComp -fasta_file /nfs/analysis_bakeoff/MOUSE
Agilent_MouseAllExonV1_S0276129.fa -max_seq 171461 -split_into 10
```

In the command above, the highlighted blue part is the actual command and the red part is specific to the farm (linux cluster) at the Sanger Institute.

Step 3: Generate Unix commands to run UNAfold

The example command and its output are given below. It's a Unix shell script that takes command line options and outputs execution commands to run UNAfold. The options are listed below:

It takes 5 values in command line options:

- Prefix for the output file (dG, Tm values are stored here) – option 1
- Prefix for RNA fasta file(s) – option 2
- Prefix for RevComp (reverse complemented) fasta file(s) – option 3
- How many times (as per the number of input fasta files) should it run? – option 4
- Include 'wait' to insert a wait command between each Unix command (useful for sequential run) – option 5

The prefix for the output file (option 1) is used to create a set of multiple files – N (option 4) number of times. Use option 5 (use 'wait') to insert a 'wait' command in between each Unix command. You can run this two ways as described in Step 4.

Execute this shell script first to check whether it outputs the correct commands (check $PATH and probably copy UNAfold's melt.pl to the local folder) and make sure that you have properly configured UNAfold execution binary). You may later redirect the shell script's output to a file (e.g., *.sh file) and execute it.

Print commands in shell (test):

```
Shell#: # WITHOUT WAIT - EXAMPLE
Shell#:./unafold_wrapper.sh hybrid Agilent_MouseAllExonV1_S0276129_rna
Agilent_MouseAllExonV1_S0276129_revcomp 10
Shell#: # WITH WAIT - EXAMPLE
Shell#:./unafold_wrapper.sh hybrid Agilent_MouseAllExonV1_S0276129_rna
Agilent_MouseAllExonV1_S0276129_revcomp 10 wait
```

```
Shell#:./unafold_wrapper.sh hybrid Agilent_MouseAllExonV1_S0276129_rna
Agilent_MouseAllExonV1_S0276129_revcomp 10
perl melt.pl --NA=RNA --temperature=65 --Ct=0.00001 Agilent_MouseAllExonV1_S0276129_rna1.fa
Agilent_MouseAllExonV1_S0276129_revcomp1.fa > hybrid1.txt
perl melt.pl --NA=RNA --temperature=65 --Ct=0.00001 Agilent_MouseAllExonV1_S0276129_rna2.fa
Agilent_MouseAllExonV1_S0276129_revcomp2.fa > hybrid2.txt
perl melt.pl --NA=RNA --temperature=65 --Ct=0.00001 Agilent_MouseAllExonV1_S0276129_rna3.fa
Agilent_MouseAllExonV1_S0276129_revcomp3.fa > hybrid3.txt
perl melt.pl --NA=RNA --temperature=65 --Ct=0.00001 Agilent_MouseAllExonV1_S0276129_rna4.fa
Agilent_MouseAllExonV1_S0276129_revcomp4.fa > hybrid4.txt
perl melt.pl --NA=RNA --temperature=65 --Ct=0.00001 Agilent_MouseAllExonV1_S0276129_rna5.fa
Agilent_MouseAllExonV1_S0276129_revcomp5.fa > hybrid5.txt
perl melt.pl --NA=RNA --temperature=65 --Ct=0.00001 Agilent_MouseAllExonV1_S0276129_rna6.fa
Agilent_MouseAllExonV1_S0276129_revcomp6.fa > hybrid6.txt
perl melt.pl --NA=RNA --temperature=65 --Ct=0.00001 Agilent_MouseAllExonV1_S0276129_rna7.fa
Agilent_MouseAllExonV1_S0276129_revcomp7.fa > hybrid7.txt
perl melt.pl --NA=RNA --temperature=65 --Ct=0.00001 Agilent_MouseAllExonV1_S0276129_rna8.fa
Agilent_MouseAllExonV1_S0276129_revcomp8.fa > hybrid8.txt
perl melt.pl --NA=RNA --temperature=65 --Ct=0.00001 Agilent_MouseAllExonV1_S0276129_rna9.fa
Agilent_MouseAllExonV1_S0276129_revcomp9.fa > hybrid9.txt
perl melt.pl --NA=RNA --temperature=65 --Ct=0.00001 Agilent_MouseAllExonV1_S0276129_rna10.fa
Agilent_MouseAllExonV1_S0276129_revcomp10.fa > hybrid10.txt
```

Step 4: Run Unix commands

You will have to run the Unix commands generated in step 3 in your local environment. The non-farm version includes 'wait' command, while the farm version excludes it.

Non-Farm version:

To run the commands one-by-one sequentially, please insert a 'wait' command in between each command. This will make sure that each command waits for the previous command to finish execution. You may like to customize this to your Linux environment.

```
Shell#: ./unafold_wrapper.sh hybrid Agilent_MouseAllExonV1_S0276129_rna
Agilent_MouseAllExonV1_S0276129_revcomp 10 wait > execute_all.sh
Shell#: chmod 755 execute_all.sh
### 'wait' OPTION INSERTS 'wait' COMMAND IN BETWEEN EACH COMMAND TO RUN THEM ONE-BY-ONE
Shell#: ./execute_all.sh
```

Farm version (recommended):

The 'unafold_wrapper.sh' script and its options are exactly the same as the above. The *only* difference is that you exclude the 'wait' option, which excludes the insertion of the 'wait' command.

```
Shell#: ./unafold_wrapper.sh hybrid Agilent_MouseAllExonV1_S0276129_rna
Agilent_MouseAllExonV1_S0276129_revcomp 10 > execute_all_in_parallel.sh
Shell#: bsub -o hybrid1.txt -q normal -R'select[mem>3000] rusage[mem=3000]' -M3000000 -J
UNA1 /path/to/submit_job_array.pl execute_all_in_parallel.sh
```

Step 5: Extract ΔG and Tm values from UNAfold output

The Hybridization java program extracts ΔG and Tm values from the UNAfold output and writes out two files: a fasta file with ΔG and Tm in the sequence titles, and a tab-delimited file with the following columns: chr, start, end, dG and Tm. The fasta file has the raw ΔG and Tm, but the tab-delimited file has the *normalized* ΔG using the formula below (RL/PL is approximately equal to the number of probes).

$$\Delta Gnorm = \frac{\Delta G}{(RL/PL)}$$

where ΔGnorm is the normalized ΔG, RL is the length of the region (merged overlapping/nearby probe regions) and PL is the fixed length of each individual probe (e.g., 120bp for Agilent probes). Tm is not normalized.

```
Shell#: java Hybridization -unafold_file_prefix <file_prefix> -regions_file
<regions_file> -input_fasta_file /path/to/file.fa -file_count <N> -probe_length <PL>
Shell#: java Hybridization -help
"-help" is not a valid option
 -file_count VAL          : Number of UNAfold output files - also used for
                            generating file name (integer N>=1)
 -input_fasta_file VAL    : Original input fasta file (generated from the
                            regions file)
 -probe_length VAL        : Length of the probe in bp (e.g., Agilent's typical
                            probe length 120 bp)
 -regions_file VAL        : Input regions file
 -unafold_file_prefix VAL : Prefix (starting file name) of a UNAfold file
```

```
Shell#: java Hybridization -unafold_file_prefix hybrid -regions_file
/path/to/genomic_regions_file.txt -input_fasta_file /path/to/original_fasta_file.fa
-file_count 10 -probe_length 120
```