Work Log for January

Logan Brown

January 9, 2015

Contents

1 Goals for the Month

1. Generate new genomes
2. Speed up NSE model
3. Move to Newton

2 Progress/Notes

2.1 Generate new genomes

1/5: Fixed a small bug in the genome code.

Now that the code actually updates the codons, not just the codon index, we can generate new genomes.

2.1.1 Potential Error – Reading small chunks of the script as commands?

I’ve seen this type of error before, and I’m still confused by it.

```
lbrown@gauley:~/cubfits/preston$ tail test*
==> test.ModOutput <==
Simulating YPR194C . . .
Simulating YPR196W . . .
Simulating YPR198W . . .
Simulating YPR199C . . .
Simulating YPR200C . . .
Simulating YPR201W . . .
Simulating YPR202W . . .
Simulating YPR203W . . .
Error: unexpected ’)’ in "x.simulation.type = ’M’)"
Execution halted

==> test.output <==
Simulating YPR194C . . .
Simulating YPR196W . . .
Simulating YPR198W . . .
Simulating YPR199C . . .
Simulating YPR200C . . .
Simulating YPR201W . . .
Simulating YPR202W . . .
Simulating YPR203W . . .
Error: unexpected ’)’ in "sta)"
Execution halted
```

This is quite confusing. There are no problems in those lines, and the commands the error indicates don't... exist?

My best judgement is that the first error is the tail chunk of

```
sim.genome <- simulate.data.all.genes(parallel='lapply', obs.genome = obs.genome, obs.phi
```

And the code is just trying to execute x.simulation.type = 'M') as a standalone command, which does produce the shown error. Similarly, for the second error, I think it's using the tail end of

```
write.fasta(sequences = sim.seqs, names = seq.ids, file.out = out.fasta)
```

And just running sta) as a command, which produces the shown error.

Google doesn't seem to show other people having this error. Is it just a problem with Rscript? RAM running out? (Doubtful, Gauley is powerful). Perhaps a problem with lapply/mclapply? One of the processes finishes early, and this... breaks the R interpreter?

I started using mclapply instead of lapply and the problem didn't happen, but I don't want to say that the problem has been 'fixed'.

2.1.2 Genome with no Omega

To debug the genome generation process, we want to look at some very simple genomes. One such genome would be one that is totally dominated by Mutation Bias, and see if the simulation correctly creates the gnome across all phi values.

2.1.3 Genome we can work out by hand

This genome would likely be structured so that each protein was a synonym of each other protein. Each gene would probably be one that only has two synonyms, and would only be about 9 amino acids long, and would have all the same amino acids in the same positions. We'd set the mutation bias to 0, and set the phi values to easily calculatable values.

2.2 Speed up NSE model

2.2.1 Move logarithm into C?

Right now, the code exponentiates the values in the C code (to be normalized), then later, in the R code, calulates the logarithm of those values, to correct this. Since C code is generally faster than R code, I thought it may be worthwhile to move that calculation into the C code.

I ran a quick test case (the code can be found in data/cLogTest.tar.gz), and it seems like making that change would save about 10 nanoseconds per gene. Our simulated yeast has 2.8 million genes, so it's only about a .3 second improvement per MCMC proposal. Basically, this change is too minor.

2.2.2 change from division to subtraction?

Using logarithm rules, we could subtract out ln(sum(exp)) instead of dividing the exponent by the sum of the exponents

2.3 Move to Newton

2.3.1 How to Install Packages

```
[Newton]$ export R_LIBS="$HOME/path_to_cubsrc/Dependencies"
[Newton]$ R
> install.packages('doSNOW')
> install.packages('coda')
> install.packages('seqinr')
> install.packages('EMCluster')
> install.packages('VGAM')
> install.packages('psych')
> install.packages('getopt')
> q("no")
```

2.3.2 How to Install CUBfits

```
export R_LIBS_USER="$HOME/path_to_cubsrc/Dependencies/"
R CMD build cubfits --no-build-vignettes --no-manual
R CMD INSTALL cubfits_0.1-2.tar.gz --library=$HOME/cubfits
```

2.3.3 How to run CUBfits on Newton

Run 'qsub run.sge' in the cubmisc/R folder. It goes as follows

```
[Newton:zeta00 R]$ cat run.sge
#$ -N Cubfits
#$ -q medium*
#$ -cwd
#$ -pe openmpi* 32
#$ -v R_LIBS_USER=/lustre/home/lbrown60/cubsrc/Dependencies/:/lustre/home/lbrown60/cubfit

echo $R_LIBS_USER
nohup Rscript run_loganYeast.r
```

3 Goals for next Month

1. Future Goal