CODE PERFORMANCE AND SCALING

De Novo Genome assembly:

Pending available funds, we plan to continue assembling genomes *de novo* for several more species of *Nucella* using the velvet module. Velvet is run in two steps with velvetg (2nd step) using the majority of the SU's. We previously ran into memory limits running on Mason and hope that the new Bridges Resource will allow us to run a larger portion of the data set in one run. From previous Illumina HiSeq PE250 runs we expect to have data sets with around 200 Million reads for each species.

Code:

```
module load velvet/1.2.10-longseq
jobid=`echo $PBS_JOBID|cut -f 1 -d'.'`
nthreads=15
export OMP_NUM_THREADS=$nthreads
cd /N/dc2/scratch/xd-bbcameron/VG61_80m

velveth . 61 -shortPaired -fastq -separate /N/dc2/scratch/xd-bbcameron/15NoR1aa /N/dc2/scratch/xd-bbcameron/15NoR2aa >&VHspl8_61.$jobid.out
```

Code:

```
module load velvet/1.2.10-longseq
jobid=`echo $PBS_JOBID|cut -f 1 -d '.'`
nthreads=16
export OMP_NUM_THREADS=$nthreads
cd /N/dc2/scratch/xd-bbcameron/VG61_80m

velvetg . -cov_cutoff 3 -ins_length 600 -exp_cov 16 -min_contig_lgth 150 -read_trkg yes -amos_file yes >&Velg_nucaa61.$jobid.out
```

Number of reads (millions)	Greenfield SUs	Number of Bridges (Regular) SUs
45M	6,892	12,850
295M	48,271	90,000

De Novo Transcriptome assembly:

We have assembled four transcriptomes *de novo* for four species using trinityrnaseq. Trinity remains the most computationally efficient program for assembling high quality transcriptomes without a reference. We have run Trinity on several data sets to determine the amount of computational time we require:

Code:

```
Trinity --seqType fq --CPU 16 --max_memory 100G --normalize_reads --bflyCPU 16
--bflyGCThreads 16 --left left.fastq --right right.fastq --SS_lib_type RF > output.log
```

Number of reads (millions)	Number of Greenfield SUs	Number of Bridges (Regular) SUs
40M	4,200	7,831
80M	7,920	14,767
120M	12,850	23,958
205M	19,200	35,798

Population genetic analyses

We implement python scripts on the Xsede platform that run the population genetics program *dadi,* these scripts take varying amounts of time, and are run multiple times to ensure adequate model fitting. In some instances, the model fit can be achieved in less than an hour on one node, using approximately 4SUs:

Code:

```
Module load python
cd /crucible/bi4ifup/cornwell/Symdadi/Basic
python 3PopBasic.py
```

When fitting more complex models with more populations, the scripts can run for ~4 days on a single node (approximately 1,536 SUs):

Code:

```
Module load python
cd /crucible/bi4ifup/cornwell/Symdadi/Basic
python 3PopScriptBNAEAXmod.py
```

We run these scripts multiple times per model with different start parameters to ensure that we adequately explore parameter space. Additionally, we run multiple models in order to determine which demographic best fits the populations and species in question:

Number of Models to Fit	Number of Greenfield SUs	Number of Bridges (Regular) SUs
6	5,000	9,322