Parallelization and Optimization Jobs on Rivanna

(from https://www.rc.virginia.edu/userinfo/rivanna/queues/)

Partition	Max time / job	Max nodes / job	Max cores / job	Max cores / node	Max memory / core	Max memory / node / job	SU Charge Rate
standard	7 days	1	40	40	9GB	375GB	1.00
parallel	3 days	25	1000	40	9GB	375GB	1.00
largemem	4 days	1	16	16	60GB	975GB	1.00
gpu	3 days	4	10	10	32GB	375GB	3.00 *
dev	1 hour	2	8	4	6GB	36GB	0.00

Sample slurm scripts: https://www.rc.virginia.edu/userinfo/rivanna/slurm/

Numba Information/Notes:

1. https://numba.pydata.org/numba-doc/latest/user/5minguide.html
2. https://numba.pydata.org/numba-doc/dev/reference/numpysupported.html

Multiprocessing Example

1. Let's say we need to parallelize 10 parallel tasks in each job, and let's say we need to submit 20 jobs (so 200 tasks in total)

   Main.py >>>

```
def run_replica(i):
    job_number = sys.argv[1]
    replica_number = 10*int(sys.argv[1]) + i


if __name__ == '__main__':
    jobs = []
    for i in range(10):
        p = multiprocessing.Process(target=run_replica, args=(i,))
        jobs.append(p)
        p.start()
```

   job.slurm >>>

```
#!/bin/sh
#SBATCH --nodes=20
#SBATCH --ntasks-per-node=10
#SBATCH --time=10:00:00
#SBATCH --output=slurm.out
#SBATCH --error=slurm.err
#SBATCH --partition=parallel
#SBATCH -A spinquest_standard
#SBATCH --array=0-20


module load openmpi


srun python3 Main.py $SLURM_ARRAY_TASK_ID
```