* **1) What native features of Python support parallel programming?**
  + To support parallel programming in Python, one can use the *multiprocessing* module in Python’s Standard Library.
    - Within multiprocessing, the class “Pool” makes it “easier to submit tasks to a pool of worker processes”
    - Threading in the “mainstream Python interpreter” faces the issue of “global lock” – however, the multiprocessing module “uses sub processes” to avoid the issue of the global interpreter lock
  + Additionally, one can work with *pathos*, or *parallel python* if/when working with clusters
  + *Sources:*
    - [*http://stackoverflow.com/questions/2987980/parallelism-in-python?rq=1*](http://stackoverflow.com/questions/2987980/parallelism-in-python?rq=1)
    - [*https://pypi.python.org/pypi/processing*](https://pypi.python.org/pypi/processing)
* **2) What native features of R support parallel programming?**
  + After registering a “parallel backend” with the number of cores available, creating a cluster, there are also various features that support parallel programming in R such as:
    - *doMC* (built on multicore, unix)
    - *doSNOW* (built on snow, windows)
    - *doParallel* (built on parallel, both)
  + *Sources:*
    - [*http://michaeljkoontz.weebly.com/uploads/1/9/9/4/19940979/parallel.pdf*](http://michaeljkoontz.weebly.com/uploads/1/9/9/4/19940979/parallel.pdf)
* **3) What Python libraries support parallel programming?**
  + There are various Python libraries that support parallel programming through shared memory or communication streams, or “employing multiple CPUs or multicore CPUs in a symmetric multiprocessing or shared memory environment, or potentially huge number of computers in a cluster or grid environment.”
  + Examples within *Symmetric*:
    - *Multiprocessing* (Python 2.6): process-based multithreading; “uses either fork on Unix or the subprocess module on Windows, implementing an API like the standard library's threading API and providing familiar objects such as queues and semaphores. Can use native semaphores, message queues or can use of a manager process for sharing objects”
    - Others: *dispy, delegate*
    - Advantages: convenient process creation and ability to share resources
  + Examples within *Cluster Computing*:
    - *IPython*: supports interactive parallel computing across multiple IPython instances
    - Others: *ScientificPython, batchlib, Celery*
    - Advantages: offer high scalability within the constraint of shared resources
  + Examples in *Cloud Computing* (similar to cluster computing):
    - *Google App Engine, StarCluster*
  + Examples in *Grid Computer*:
    - *Ganga, pyGlobus*
  + *Sources:*
    - [*http://calcul.math.cnrs.fr/Documents/Ecoles/2010/cours\_multiprocessing.pdf*](http://calcul.math.cnrs.fr/Documents/Ecoles/2010/cours_multiprocessing.pdf)
    - [*https://wiki.python.org/moin/ParallelProcessing*](https://wiki.python.org/moin/ParallelProcessing)
* **4) What R libraries support parallel programming?**
  + R application libraries that “deal with parallelism on their own” include: *gbm, boot, vtreat*
  + “Parallelization abstraction frameworks” include*: Thrust/Rth*
  + “Specialized parallel extensions” include: RevoScaleR, h2o.ai
  + Other parallel libraries: BLAS/LAPACK
  + *Sources:*
    - [*http://www.win-vector.com/blog/2016/01/parallel-computing-in-r/*](http://www.win-vector.com/blog/2016/01/parallel-computing-in-r/)
* **5) Can GPUs be used with Python and R? If yes, then how?**
  + Projects such as PyCUDA and PyOpenCL allow for CUDA code to be written within *Python*/integrated, which allows for use of the GPU
  + GPU operations can be accessed with *R* through packages such as *gputools*, *gmatrix*, and *Rth*, available on CRAN and through GitHub, however “perform only specialized operations” such as “matrix manipulation and sorting”
  + *Sources:*
    - [*http://blog.revolutionanalytics.com/2015/01/parallel-programming-with-gpus-and-r.html*](http://blog.revolutionanalytics.com/2015/01/parallel-programming-with-gpus-and-r.html)
    - [*https://cran.r-project.org/web/views/HighPerformanceComputing.html*](https://cran.r-project.org/web/views/HighPerformanceComputing.html)
    - [*http://stackoverflow.com/questions/5957554/python-gpu-programming*](http://stackoverflow.com/questions/5957554/python-gpu-programming)