**Most important Files (in order of importance) –**

1. **HMMEditor -**  Basic function - read in HMM files in HMMER format, change probabilities of substitutions, print out HMM files in HMMER format.
   1. Common method parameters
      1. Top threshold – how high the probability should be of seeing the highest index. This is set to 96% by default which was picked arbitrarily after looking at some example HMMs
      2. Bottom threshold – how low the probability of substituting to other bases. When things have to be recalibrated this is typically the cause. Note by being recalibrated I just mean the reallocateExtras method is called.
      3. Proportional – Indicator to say whether to add or remove % bases from the substitute bases proportionally to their current odds. If false, the changes are made uniformly to all 3.
   2. Other assumptions built in
      1. The original index should always be at worst tied for the most likely base to occur regardless of the amount of change asked for.
   3. Other things of Note
      1. Originally I designed this class to work for any alphabet e.g. could work on proteins too. However, I stopped considering in that in changes as time past. It may work or it may not with other alphabets.
      2. This file does nothing in the ways of changing insertion and deletion probabilities. That would need to be added.
2. **DNAPartitioner –** Basic function create partitions in DNA files based on the predetermined number of repeat bases in each partition.
   1. Common method parameters
      1. repBasesPerPart – number of repeat bases that need to be in a partition before a new partition is created
      2. maxBases – how long a partition can span in terms of absolute number of bases
      3. maxGap – number of maximum bases between two repeats in a partition. Meant to ensure the centroid of Ns in chromosome do not cross a repeat.
      4. Rate – the amount that probabilities should be shifted given as a percentage of D value change. Default value of .65 is based off a linear regression of d values to percentages of diagonal in P matrix
      5. Buffer – the amount of flexibility around the searched HMM that repeats expected D value can be to count as repeat bases in the partition. For example, if searched for HMM is MIR with expected D value of .35. Then a buffer value of .1 would mean repeats with an expected D value of .25 to .45 would be considered when counting repeat bases for a partition and calculating the change in D values.
   2. Assumptions built in:
      1. This file assumes we are working with DNA. No consideration was made for other alphabets
3. **FileMaker.py**: Makes temporary directory, Creates files to be used by HMMER for each partition, parses reads and summarizes result files
   1. Common method parameters
      1. This file needs all the big picture inputs to the HMMAR program.
         1. Starting HMM file
         2. Starting Fasta File
         3. PSM file
         4. Results file
   2. Other things of node:
      1. This class contains a method to write PSM files in a form that can be used by the HMMAR program
         1. It is called createPickledFiles which intern calls something from RPTMatrixMod
         2. This method will need to be edited to contain the path to the PRM files
4. **Hmmeraroc:** The starting point of the program. Gathers input from the command line. Calls FileMaker. Calls HmmerInterface to fork a process to run HMMER on each of the partitions.
   1. Common method parameters – See how to run HMMAR document
   2. Other things of note:
      1. You may notice that I fork two processes at a time. It may be better to do one at a time since HMMER is already multithreaded and actually capable of being run on a cluster due to MPI code. I have not ever run it on the cluster since it does not take that long anyway, but it may be worth it someday.
5. **HmmerInterface:** forks a process that runs the hmmer software on.

**Utility Files (In order of usefulness):**

1. **SimulationFileGenerator:** creates a simulated fasta sequence and simulate PSM file to run the program on.
   1. Common method parameters –
      1. Start – the probability of change at beginning of file
      2. End – the probability of change at end of file
      3. Incr - how much to increment the partition for each loop
      4. OtherRepeatFileNames – the repeats that are used to populate the PSM files as well as the fasta file that are not being searched for
         1. These were originally chosen for their close d values. I keep the list of the following ["./HMMs/LTR16a.hmm","./HMMs/MARNA.hmm","./HMMs/MER115.hmm","./HMMs/mlt1l.hmm", "./HMMs/tigger8.hmm"] because they are convenient for getting 50k bases in a partition at one increment level.
      5. Repeat to find – the repeat you are going to search for. Note this repeat is not used in populating the PSM files
      6. Repeat indices file – this will tell you where the repeat you are searching for actually appears in the files.
   2. Other Things of Note –
      1. There are a lot of randomly generated numbers if this file. If you want consistency in the results run to run you will need to seed the random number generator.
      2. Right now the repeats that generate the PSM and repeat to be found are being forced to change at the same rate.
         1. This makes sense on one level, but it would be better to have these repeats change the same relative to their expected rate of change makes a lot of sense too.
2. **SimulationResultsSummary –** Takes the results from a simulation and compares it against the know places where a repeat exists from the simulation indices file generated by simulationfilegenerator. It then generates report based on this data.
   1. Common parameters needed
      1. Location of the results folder
      2. Location of the file defining the indices
   2. Other notes –
      1. Results from the HMM come in order of significance of the match not in order appearing on the chromosome. Therefore, it is necessary to search the possible matches to find the closest one. Unfortunately, the start and end of the matches vs. placed in repeats hardly ever are exact. Therefore, I decided that I would allow for a 400 base buffer in the start coordinates. If nothing matches those then the closest lower starting matching chromosome is chosen.
3. **HMMDownloader –** downloads all the HMM files from the dfam database. I believe this represents all the known repeats in the that have HMMs.
   1. Assumptions
      1. <http://www.dfam.org/download/model/> is the path to trigger the download of a file
      2. Repeats are sequentially listed in the database starting at 1 and going to 1338. The format of a database listing is DF000 followed by a number that is padded on the front to be 4 zeros. E.g. DF0000009 for number 9
      3. The name of the repeat matches the name in the hmm file.
4. **HMMSummaryCreator –** loops through all HMMs, determines the theoretical d value of the HMM of repeat, counts number of bases in the repeat, tries to look up the age of the repeat from the GlobalIDs.txt file(from Michelle’s strand symmetry project) .
   1. **Assumptions**
      1. All the HMMs are located in a folder ./AllHmms
      2. The name in the globalids.txt file matches that from the HMM
         1. I think there is some room for improvement here, but I didn’t try to find subtle differences that could result in mismatches
   2. **Other Notes**
      1. Only 503 of the 1338 HMMs found a match in the GlobalIDs.txt file. There are 638 entries in the globals.txt file so there are probably roughly 100 other connections that could be made.
5. **Sequence generator –** The purpose of this file was to make multiple alignments that could be plugged into hmmalign and hmmbuild to build theoretical HMMs. This was a preliminary file only.

**Files from Previous Projects – mostly strand symmetry**

1. **asymm\_tools –** various matrix manipulations, calculations of p and q matrices and d values
2. **fp\_check –** test if floating point number is within a certain tolerance of other numbers
3. **RMfileReader –** regex on regex on regex. Used to extra repeat masker data from files. Never touched this and have no desire to.
4. **RPTMatrixMod –** bridge between RMfileReader and PSM files. Used mostly to understand attributes of PSM and PRM files. Altered slightly to make new PSM files with correct imports. Used in order to make simulations as well.