```
////////////////////////////////////////////////////////////
//  Daniel  ////////////// Computational //////////////  Final  //
// Richford //////////////    Physics    ////////////// Project //
// 20171031 //////////////   Fall 2017   //////////////  Plans  //
////////////////////////////////////////////////////////////

////////////////////////////////////////////////////////////
//                                                        //
// Language: Python (or C++ ?)                            //
//  Project: Distribution Unfolding                       //
//      Due: End of Fall 2017 semester                    //
//                                                        //
////////////////////////////////////////////////////////////

////////////////////////////////////////////////////////////
//                                                        //
// Overview: An althorithm to determine the relative contrib- //
//           utions of two unknown distributions that make    //
//           up a combination distribution, using random      //
//           distributions and checking their contributions   //
//           to the combination distribution iteratively,     //
//           until two distributions are settled upon.        //
//                                                        //
//  Details: We're looking for the relative yield (that is,   //
//           the number of electrons created from decays) of  //
//           two mesons, D, which has charm quarks, and B,    //
//           which has bottom quarks. The fractions of each   //
//           of these yields compared with the total yield    //
//           indicates the number of charm and bottom quarks  //
//           actually created during a heavy-ion collision.   //
//           Since these two distributions are not known, we  //
//           have to unfold (or un-convolve) their contrib-   //
//           utions from the total yield distribution (after  //
//           subtracting known contributions from mesons and  //
//           hadrons composed of lighter quarks. Finally, the //
//           best way to approach these two mesons is from    //
//           the distributions for the distance of closest    //
//           approach that the projected trajectories of the  //
//           decay electrons make with the primary collision  //
//           point, since the D-meson is far longer lived     //
//           than the B-meson. The algorithm randomly creates //
//           D.C.A. distributions for the D- and B-mesons,    //
//           checks if they fit into the combined D.C.A.      //
//           distribution, and then calculates the fractional //
//           yield when good-fitting candidates are reached.  //
//                                                        //
////////////////////////////////////////////////////////////

////////////////////////////////////////////////////////////
//                                                        //
// Figure:                                                //
//                                                        //
// Daughter   --<--\     /-->-- e- <- Path of decay electron, //
//  Hadron          \   /             bent by magnetic field, //
//                   \ /              from 2ndary vertex      //
// Secondary Vertex   *            <- Secondary Vertex (decay //
//                    |\               of charmed or bottom    //
//                    |                meson)                  //
//                  ^ |  \ _ _ _   <- Traced-back (projected) //
//                  | |   :            path of decay electron  //
//                    |   :        <- Solid line (|): path of //
//                  M |  :             meson generated from    //
//                  E ^  : D           primary vertex          //
//                  S | :  C       <- Dotted line (:):        //
//                  O | :  A           distance of closest ap- //
//                  N |:               proach from the decay-  //
//                    |:               electron's traced-back  //
//                    |                path to the primary vtx //
//          n --->--- * ---<--- n  <- Primary vertex (col-    //
//             Primary Vertex          lision of two nucleons) //
//                                                        //
////////////////////////////////////////////////////////////
```

```
////////////////////////////////////////////////////////////
//                                                        //
//     Flow:                                              //
//           1. Setup                                     //
//              1.1 Dimensionality                        //
//              1.2 Input/Output directories              //
//              1.3 Files for later use                   //
//                  1.3.1 Probability matrices            //
//                  1.3.2 Error matrices/error-bar limits //
//                  1.3.3 Combination distribution's matrix //
//           2. Preliminary Work                          //
//              2.1 Exclude certain DCA bins              //
//              2.2 Simulation for initial values, normal- //
//                  ization, comparison with final result //
//              2.3 Initialization for Markov chains      //
//              2.4 Prior probability vector              //
//              2.5 Modifications to individual DCA matrices //
//           3. Markov-Chain Monte-Carlo Sampler (emcee)  //
//              3.1 Regularization, paramter limits, starting //
//                  points for random walk                //
//              3.2 Perform a random walk, allow the matrices //
//                  to evolve from the starting point     //
//                (3.2+1/2 Animation ?)                   //
//              3.3 Get posterior probability percentiles //
//              3.4 Compare posterior probabilty percentiles //
//                  with covariance matrix                //
//           4. Check Results                             //
//              4.1 Re-fold to check                      //
//              4.2 Calculate covariance from combined and //
//                  constituent distributions             //
//              4.3 Get fractional contribution yield from //
//                  the two constituent distributions     //
//              4.4 Calculate likelihood for the two com- //
//                  ponants                               //
//                  4.4.1 Likelihood of getting the expected //
//                        fractional contribution (Poisson?) //
//                  4.4.2 Likelihood from the unfolded     //
//                        results (Monte Carlo sampling)  //
//               4.5 Plot and save results to output files //
//                                                        //
////////////////////////////////////////////////////////////

////////////////////////////////////////////////////////////
//                                                        //
// Other Thoughts:                                        //
//                                                        //
//           1. I'm having trouble understanding the co-  //
//              variance thing -- there might be a slower, //
//              more computationally-intensive way to go  //
//         * 2. Requires ROOT module for python -- which  //
//              I am having trouble installing (it should //
//              work with Python 3.6, but I keep getting  //
//              missing-module error when I say "import   //
//              ROOT"                                     //
//           3. Might be best to first determine the frac- //
//              tional yield of one particle (the B-meson) //
//              rather than two (both the B- and D-mesons) //
//           4.                                           //
//                                                        //
//                                                        //
//                                                        //
////////////////////////////////////////////////////////////

////////////////////////////////////////////////////////////
```