So how about every week we try to set up analytics milestones that u try really hard to meet?  For the rest of the week i want u to be able to simulate a hidden markov model with known transition probabilities.

First simulate x.  So I give u the "P" matrix from notes (or it might be called "Q") and I give u the prior "pi" on the state at time 1.  With those two pieces of information u should be able to simulate a markov chain of length n time points.  U randomly draw the x1 from pi.  Then given u know what x1 is, (let us say it was 2), u draw x2 from Q(x2 | x1 = 2).  Then keep moving until u have x1 to xn in memory. Then simulate y.  That is simple.  Each xt gets randomly mapped to yt using the R matrix.  This should be easy.  
  
In both cases, (a) going from x at time t to x at time t+1 using "P" or "Q" or whatever the name of the transition matrix is,  or (b) going from x at time t to y at time t using R,  
  
Note that u have a black box w a discrete input and u are randomly mapping that to a discrete output in a manner that is consistent with the Q matrix or the R matrix.  U can do this very easily with an "if then else" statement. Suppose that x could be 0 or 1.  And the Q matrix was of the form  
  
Q(1|0) = 3/4 and Q(0|0)=1/4.  
Q(1|1) = 7/8 and Q(0|1) = 1/8  
  
Then this means that IF  x at time t (call it the state variable "xprevious") was zero then assign x at time t to be 1 (call it the state variable "xnext") with probability 3/4 and assign it to be 0 with probability 1/4.  That is easy to do. Draw a z to be uniform [0,1] using the Matlab command rand.  If z is between 0 and 3/4 then assign xnext to 1; else assign xnext = 0.

Now suppose xprevious  was 1.  And suppose that Q(1|1) = 1/8 and Q(0|1) = 7/8.  Then u know the analogy.  
  
So your pseudo code should be of the form  
  
  
- assign x(1) to be drawn from PMF "pi".  Do this by saying "z=rand; if z < pi(1) then x(1)=0; else x(1)=1;"  
  
- assign state variable: xprevious=x(1).  
- assign n= 30 (let us say)  
  
For t = 2 to n,  
   - assign state variable xnext given xprevious using Q matrix (use if then else code I mentioned above)  
   - assign x(t) = xnext  
   - assign state variable ynext given xnext using R matrix (again assume for now y is binary and it will be the exact same if then else code I mentioned above)  
    - assign y(t) = ynext  
   - update state variable: xprevious = xnext  
  
End  
- Then u should have x(1) ... x(n) and y(1) ... y(n) and u can visualize them.  Then u will have simulated your HMM from known statistics (pi, Q, R).  U can then change pi, change Q, and change R and see what happens when u visualize x and y.  Feel free to ask Sanggyun or Diego if u have questions w part of this but u should be able to do this in a few hours if u dedicate time to it.  This will help u understand the generative process of an HMM with known statistics.  Here u r acting like god where u r generating a sample (x,y) from known statistics (pi, Q, R).  Let us make this the milestone for this week and remember to ask for help if u need it, but really try and get it done. And for next week, by the end of the week, u should set a milestone to get the filter totally implemented.  
  
Let us also remember that once we get u to this level of building filters for not only discrete processes but also continuous ones, u will be able to pivot and do cool data analysis for so many problems that might interest u.  But until that point, we won't be able to do more than talk about how cool it could be to associate this w that.  Let's push hard to the analytics going!