July 4 notes from Todd re: Simulating our initial states document

**Random Variables and Mapping**

on page 3, u have the right intuition but I think some things you can think about more clearly

Z is nothing but a unif(0,1) random variable.

period

so you might want to say that "therefore we can CALCULATE (not define) the probabilty that Z falls within a given interval. For any interval [c,d] that is a subset of the [0,1] line, the probability that Z lies in [c,d] is simply d-c."

you are trying to construct a random variable X with a given pmf pi

X represents the state we are in initially. In this example, X can take on the values of 1 and 2

so i think, to make things easier, say you are trying to create a mapping

yes - X is 1 or 2. it is the state. So we are trying to create a mapping S that maps any point on the (0,1) line to either the value 1 or 2

**inconsistent use of variables:**

i would NOT use a and b in the two different ways u r using it

on page 2, you talk about how if U were a random variable that was uniformly distributed between a and b,

then we know his PDF is given by f\_U(v) = 1/(b-a).

because they are all equally likely and the area must integrate to 1.

here, a and b are parameters that describe the PDF. they are the endpoints: the smallest value U can take and the largest value U can take

clear?

so using a and b there is fine

 but then I changed them to represent my interval endpoints. so its inconsistent use of the same variables

**Terminology**

**{{{{{{{{{**

we say that U is a random variable (namely, a mapping from the sample space to the real line) whose statistics are given by the uniform distribution on (a,b) : **what are we *mapping* here? Aren’t all of the values in the sample space already on the real line? Our sample space happens to have the same values as the real line values we are mapping it to.**

so you could say U ~ unif(a,b) :

U is the random variable (which is a mapping)

and when I say "unif(a,b)", I am telling you about the statistics of U

I am saying that U is equally likely to be between a and b

so on page 2, you might wanna re-write and replace "U(a,b)" with "U ~ unif(a,b)"

**}}}}}: DONE**

so Z is a uniform(0,1) random variable that matlab gave you

you are trying to construct a random variable X with a given pmf pi

X represents the state we are in

X is 1 or 2. it is the state. So we are trying to create a mapping S that maps any point on the (0,1) line to either the value 1 or 2

if U were a random variable that was uniformly distributed between a and b, then we know his PDF is given by f\_U(v) = 1/(b-a). b ecause they are all equally likely and the area must integrate to 1

so you migh twant to say that "therefore we can CALCULATE (not define) the probabilty that Z falls within a given

interval. For any interval [c,d] that is a subset of the [0,1] line, the probability that Z lies in [c,d] is simply d-c.

you are not DEFINING that probability. rather, you are calculating it. the fact that Z is unif[0,1] is the defining property of the statistics. from there, you are simply doing a calculation via the axiooms of probability

we want to define our intervals on Z so that the PDF in each interval is equal to our given pi matrix

probabilities

1. there are no "intervals on Z". she is just a random variable.
2. "the PDF in each interval" should be replaced with "the area under the PDF in each interval". see why i say that? the PDF of Z evaluated at any u is simply equal to 1. but the area under the curve of the PDF, for any interval, will then be equal to the length of that interval.

you are trying to create a function S.

* S has a domain (inputs) that is the [0,1] line
* its range (possible ouputs) is {1,2}
* So every point u lying in [0,1] gets mapped either to the point 1 or to the point 2
* we are trying to DESIGN a funcion S so that:
  + the random variable X = S(Z) has the property that P\_X(1) = 1/4 and P\_X(2) = 3/4.
  + that is the fundamental engineering problem in front of us

so in the notes, i would write that i am trying to create a map S that maps the [0,1] line to the points {1,2} and the question becomees, how do I build it?

* + S is just a function, so clearly, there could be many possible S maps.
  + let us pick a simple one though. one simple idea is to have the function say:
    - I am a black box
    - my inputs can be between 0 and 1
    - my outputs can be either 1 or 2
    - here is my assignment rule
      * if the input (call it "u") lies between 0 and some threshold c, then declare output (call it "x") to be equal to 1.
      * else (i.e. u lies between c and 1), then declare x to be equal to 2.
      * this is a class of functions: for different thresholds c, the function behaves differently
        + for example, let c = 0.2, then note that S\_c(0.1) = 1.
        + Now imagine c = 0.05, then note that S\_c(0.1)=2.
        + **so in general, the behavior of the function S\_c depends on the threshold c**
      * **Note: not intervals on Z, really what they are, is they are the intervals for which the output of the map S equals 1. or 2.**
        + **This is at the core of optimal transport shit i do with Rui**
* now we have Z ~ unif[0,1]
* and we want to create a map S\_c, so that X = S\_c(Z)
* AND that we can guarantee that P\_X(1) = 1/4, P\_X(2) = ¾
* you might imagine that c depends on the P\_Z and P\_X, and in general it does!
  + so let us just pick c arbitrarily and see what the hell happens. Let me pick c to be 0.45
  + Question: what is P\_X(k)? and does it match our pre-specified pi?
    - note that P\_X(1) is by definition the probability that X = 1
    - so let us slow down for a second and think about the event {X=1}
      * I argue that event is EQUIVALENT to the event that {0 <= Z <=c}
        + this holds because we are declaring that X = S\_c(Z), and S\_c has that "if then else " code inside of it i mentioned above
        + so now we have that P\_X(1) = P({X=1}) = P({ 0 <= Z <= c}) = c

all these events are equal to “c” because we know that the probability that Z falls within a certain interval is equivalent to the area under the PDF in that interval, and furthermore, because the height of the interval is 1, we know that P({ 0 <= Z <= c}) = c-0=c.

but c = 0.45! so we have found that P\_X(1) = 0.45

so what is interesting is that you have to pick c based upon P\_X somehow

and what we find is that we should pick c to be equal to P\_X(1), or equivalently, pi(1)

so we are engineering a map so that the statistics of its output have a pre-defined structure

equivalence of events is the key step

* can Z be equal to c? or can it only be less than or greater than c?
  + Z is a continuous random variable, so the probability she EXACTLY takes on any value is 0
  + note that the event {0 <= Z <=c} = {0<= Z < c} UNION {Z = c}
    - P({0 <= Z <=c}) = P({0<= Z < c}) + P({Z = c})
    - Disjoint union so you can add them together b/c the two events don’t intersect
    - And since P({Z = c})=0, the two events are statistically equivalent
* also note that this black box where I give you P\_Z, I give you P\_X, and you are trying to build a map S so that P\_Z gets transformed into P\_X:
  + that is EXACYLY the same black box you use for the rest of the code
  + remembber we used the same functions!
    - P\_Z is always unif(0,1)
    - and P\_X is context-specific
    - but it s always the same type of black box u r calling!
  + so now u know how to be God. u know how to generate data from pre-specified statistics for an HMM
    - both the x process and the y process
* next step is to try and build the filter.
  + remember it has 2 ocmponetns: one-step prediction update and the bayes rule
  + always done sequentially, in a for loop
  + when u r trying to be the filter, u are no longer god, u r now an observer
  + (x,y) were generated using pi, Q, and R
    - but then u only get to observe y