Lab2

Jun Rao

9/4/2019

## Probability distributions in R and Stan

# Task 1 (8 points)

Study the previous code provided on probability distribution functions in R (see the file on BbLearn). Your task is to choose a continuous probability distribution other than the Gaussian distribution and complete the following (2 points each). You can use the Stan functions to help.

1.Create a custom function to calculate the relative probability density of a given observation of a continuous random variate.

#Uniform Distribution  
pdf\_func = function(x,lower,upper){  
 if((x>= lower) & (x<=upper)){  
 result = 1/(upper-lower);  
 }  
 else{  
 result = 0;  
 }  
 return(result)  
}  
  
pdf\_func(x=0,lower=-10,upper=10)

## [1] 0.05

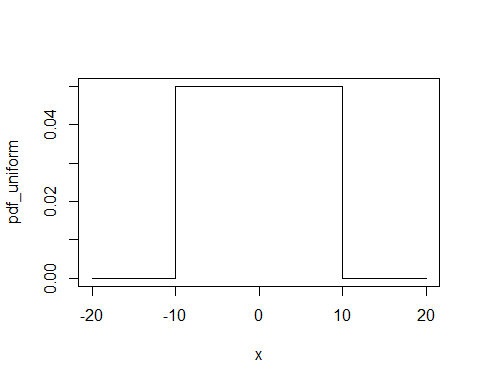
2.Compare your function to R’s built-in density function (i.e., show that they give the same output).

#Uniform Distribution  
  
dunif(0, min = -10, max = 10)

## [1] 0.05

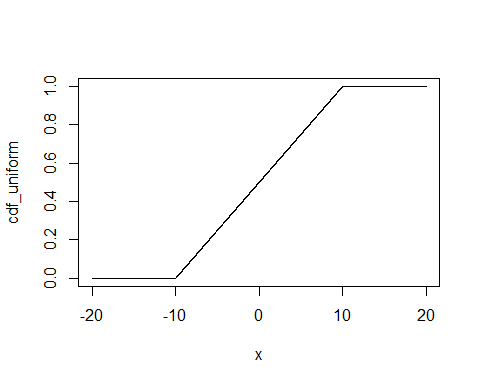
3.Plot the probability density function P(y|Θ), where Θ is a vector of user-specified parameters that define the probability distribution of interest.

#Uniform Distribution  
x = seq(-20,20,length.out = 10000)  
  
pdf\_uniform = dunif(x,-10,10)  
  
plot(pdf\_uniform ~ x,type = "l")



4. Plot the cumulative density function F(y|Θ), as above.

#Uniform Distribution  
x = seq(-20,20,length.out = 10000)  
  
cdf\_uniform = punif(x,-10,10)  
  
plot(cdf\_uniform ~ x,type = "l")



# Task 2 (12 points)

Your task is to now choose a discrete probability distribution, and use the guided code above to complete the following tasks. Again, see the Stan functions documentation for assistance.

# Load the rstan package  
library(rstan)

# Set some useful options  
options(mc.cores = parallel::detectCores())  
rstan\_options(auto\_write = TRUE)

1.Create a pmf\_sim.stan file that generates the PMF of your chosen distribution and that draws random values from the distribution (5 points).

# Define data inputs  
  
## Number of random draws:  
n\_draws = 1000  
## Number of points used for PMF:  
n\_seq = 101  
## Sequence of observed y, for PMF:  
y\_pmf = seq(0, 100, length.out = n\_seq)  
  
# Store the required data in a list  
sim\_data = list(n\_draws = n\_draws,  
 n\_seq = n\_seq,  
 y\_pmf = y\_pmf)  
  
  
#------------------  
#------------------  
# Run the simulation:  
sim\_fit =  
 stan(file="pmf\_sim.stan",  
 data=sim\_data,  
 iter=1,  
 chains=1,  
 algorithm="Fixed\_param"  
 )

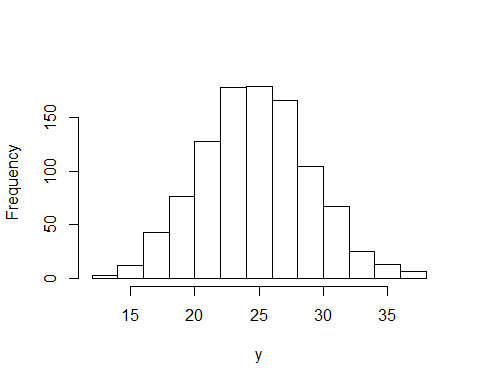
##   
## SAMPLING FOR MODEL 'pmf\_sim' NOW (CHAIN 1).  
## Chain 1: Iteration: 1 / 1 [100%] (Sampling)  
## Chain 1:   
## Chain 1: Elapsed Time: 0 seconds (Warm-up)  
## Chain 1: 0 seconds (Sampling)  
## Chain 1: 0 seconds (Total)  
## Chain 1:

# The function extract() will create a list of output  
# Note that "lp\_\_" is irrelevant here, but will become  
# very important later in the course.  
  
sim\_out = extract(sim\_fit)  
str(sim\_out)

## List of 3  
## $ log\_lik : num [1, 1:101] -28.8 -25.3 -22.5 -20.1 -18 ...  
## ..- attr(\*, "dimnames")=List of 2  
## .. ..$ iterations: NULL  
## .. ..$ : NULL  
## $ rand\_binomial: num [1, 1:1000] 22 24 37 25 18 25 21 31 23 27 ...  
## ..- attr(\*, "dimnames")=List of 2  
## .. ..$ iterations: NULL  
## .. ..$ : NULL  
## $ lp\_\_ : num [1(1d)] 0  
## ..- attr(\*, "dimnames")=List of 1  
## .. ..$ iterations: NULL

2.Plot a histogram of the random draws (2 points).

# Plot the random draws:  
hist(sim\_out$rand\_binomial[, 1:n\_draws], main = "", xlab = "y")



mean(sim\_out$rand\_binomial[, 1:n\_draws])

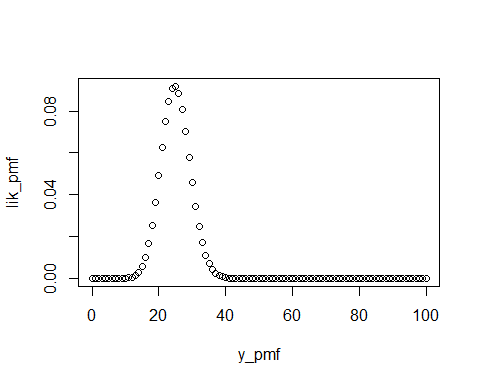
## [1] 25.237

var(sim\_out$rand\_binomial[, 1:n\_draws])

## [1] 18.27711

3.Plot the PMF using bars or points (3 points).

# Plot the PMF  
## First get back to usual scale:  
lik\_pmf = exp(sim\_out$log\_lik[, 1:n\_seq])  
  
## Now plot:  
plot(lik\_pmf ~ y\_pmf, type = "p",)



4.Compare the PMF generated in Stan to the PMF generated by R’s built-in functions (2 points).

# Plot the PMF  
lik\_pmf\_R = dbinom(y\_pmf,size=100, prob = 0.25)  
plot(lik\_pmf\_R ~ y\_pmf, type = "p")

