STA 478 Assignment #2 Solutions

Dr. Robert Buscaglia

September 12, 2025

Inverse CDF

```r
# @param iCDF : inverse CDF function that accepts 'u'
#     Example: my.iCDF <- function(u) qnorm(u, 0, 1)
# @param n : desired sampled size

inverse.CDF <- function(iCDF, n){
  u <- runif(n)
  return(iCDF(u))
}

# @example
# my.iCDF <- function(u) qnorm(u, 0, 1)
# x <- inverse.CDF(iCDF = my.iCDF, n=1000)
# hist(x)
```

Accept-Reject

```r
# @param target.pdf : the target probability density function as a function of 'x'
#   Example : target.pdf <- function(x) dbeta(x, 12, 6)
# @param lower : target pdf lower domain
# @param upper : target pdf upper domain
# @param n : desired sample size

accept.reject <- function(target.pdf, lower, upper, n)
{
  ### Estimate M
  M <- max(target.pdf(seq(lower, upper, length.out = 1e5)))
  ### Storage
  x <- numeric()
  ### Loop
  while(length(x) < n)
  {
    x.star <- runif(1, lower, upper)
    y <- runif(1, 0, M)
    if(y <= target.pdf(x.star)) x <- c(x, x.star)
  }
  return(x)
}

# @example
# target.pdf <- function(x) dbeta(x, 12, 6)
# x <- accept.reject(target.pdf = target.pdf, lower = 0, upper = 1, n = 1000)
# hist(x)
```

Monte Carlo Markov Chain

```
# @param target.df : the target probability density function as a function of 'x'
#   Example : target.df <- function(x) dexp(x, 1)
# @param prop.fun : proposal function that generates a perturbation from a given 'x'
#   Example : prop.fun <- function(x) x + rnorm(1, 0, 0.1)
# @param start : a numerical input; starting point of MCMC chain
# @param nsteps : a numerical input; number of MCMC steps
#
# @note : This MCMC has a slightly different method for generating results.
#         It is written for multiple dimensions.  You may use it in this form,
#         but should only give one-dimensionsal proposal functions.

MCMC.metropolis <- function(target.df, prop.fun, start, nsteps)
{
  step <- function(x, target.df, prop.fun) {
    x.prop <- prop.fun(x)
    alpha <- target.df(x.prop) / target.df(x)
    if(runif(1) < alpha) x <- x.prop
    return(x)
  }

  x<-start
  results<-matrix(NA, nsteps, length(start))

  for(i in seq_len(nsteps))
  {
    results[i,] <- x <- step(x, target.df, prop.fun)
  }

  return(results)
}

# @example
# target.df <- function(x) dexp(x, 1)
# prop.fun <- function(x) x + rnorm(1, 0, 0.5)
# x <- MCMC.metropolis(target.df = target.df, prop.fun = prop.fun,
#                      start=runif(1, 0, 3), nsteps=1e3)
# hist(x)
```

Gelman Diagnostics

```r
# @param x : matrix of size n x m; m columns represent different MCMC chains

Gelman <- function(x)
{
  n<-dim(x)[1]
  m<-dim(x)[2]
  W<-mean(apply(x, 2, var))
  B<-(n/(m-1))*sum((apply(x,2,mean)-mean(x))^2)
  sigma2.hat <- ((n-1)/n)*W + (1/n)*B
  R.hat <- sqrt(sigma2.hat/W)
  n.eff <- m*n*sigma2.hat/B
  return(data.frame(W = W, B = B, sigma2.hat=sigma2.hat, R.hat=R.hat, n.eff=n.eff))
}

# @example
# See the Gelman PDF where I use this function.
```