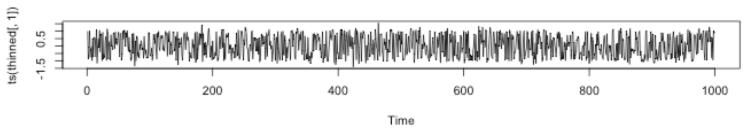
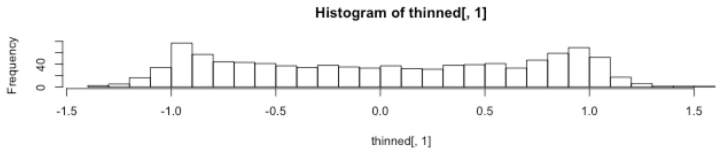
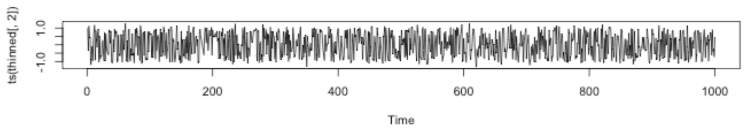
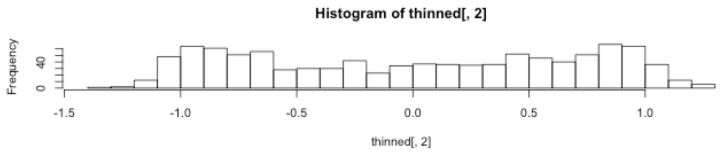
```
# MCMC - HW2 - Question 5.a, 5,b

# Helpful: http://www.macs.hw.ac.uk/~stan/f73bi/bayes_tut_5.pdf

RandomWalk <- function(init.state, n, sigma) {
  # Implements two-dimensional random-walk Metropolis MCMC with a N(0,sigma^2)
  # proposal distribution for each dimension.
  #
  # Args:
  #   init.state: Two-dimensional vector of initial values for x1, x2.
  #   n: Length of the chain to be simulated.
  #   sigma: Standard deviation of step size in proposal.
  #
  # Returns:
  #   y: Vector of length n+1 consisting of the initial
  #      state and the subsequent n steps of the Markov chain.
  #
  if(Target(init.state)==0) {
    cat("Please choose initial state with non-zero Target density\n")
    return()
  }

  # Compute Normal proposal steps on each of two dimensions.
  x1.steps <- rnorm(n,0,sigma)
  x2.steps <- rnorm(n,0,sigma)
  steps <- cbind(x1.steps, x2.steps)

  # Generate uniforms for acceptance decisions.
  u <- runif(n)

  # Create vector to hold successive states of chain.
  y <- matrix(NA, nrow=n+1, ncol=2)

  # Initialize chain.
  y[1,] <- init.state

  accept.count <- 0

  # Run MH.
  for(i in 1:n) {
    y[i+1,] <- y[i,] + steps[i,]
    accept.prob <- min(1,Target(y[i+1,])/Target(y[i,]))
    if (accept.prob < u[i]) {
      y[i+1,] <- y[i,]
    } else {
      accept.count <- accept.count+1
    }
  }
  return(y)
}

Target <- function(init.state) {
  # Evaluates the target distribution's probability of a given state.
  #
  # Args:
  #   init.state: Two-dimensional vector of initial values for x1, x2.
  #
  # Returns:
  #   prob: Probability of init.state, under target distribution.
  x1 <- init.state[1]
  x2 <- init.state[2]
```

```
  prob <- exp(-5*abs(x1^2+x2^2-1))
  return (prob)
}

# Run RandomWalk and plot results to test convergence and mixing.
init.state <- c(1, 1)
chain <- RandomWalk(init.state, 100000, 0.5)
thinned <- chain[seq(1,100000, 100),]

par(mfrow=c(4,1))
plot(ts(thinned[,1]))
hist(thinned[,1],30)
plot(ts(thinned[,2]))
hist(thinned[,2],30)

# FOR FUN.
dev.off()
x1 <- seq(-.5, .5, .1)
x2 <- seq(-.5, .5, .1)
zfun <- function(x1,x2) {exp(-5*abs(x1^2+x2^2-1))}
z <- outer(x1, x2, FUN="zfun")
persp(x1, x2, z)

plot(thinned[,1], thinned[,2])
hist(rowSums(y^2))

# Part B:
# A more efficient algorithm could be to use a Gibbs sample of one variable at
# a time, or to change x1, x2 into polar coordinates, and sample uniformly
# from the theta parameter, to retrieve original x1, x2's.
```

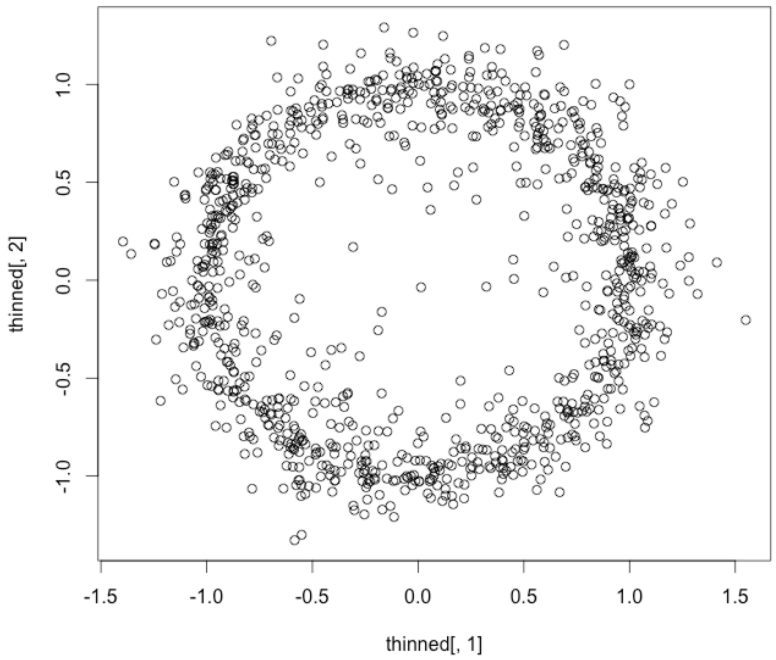




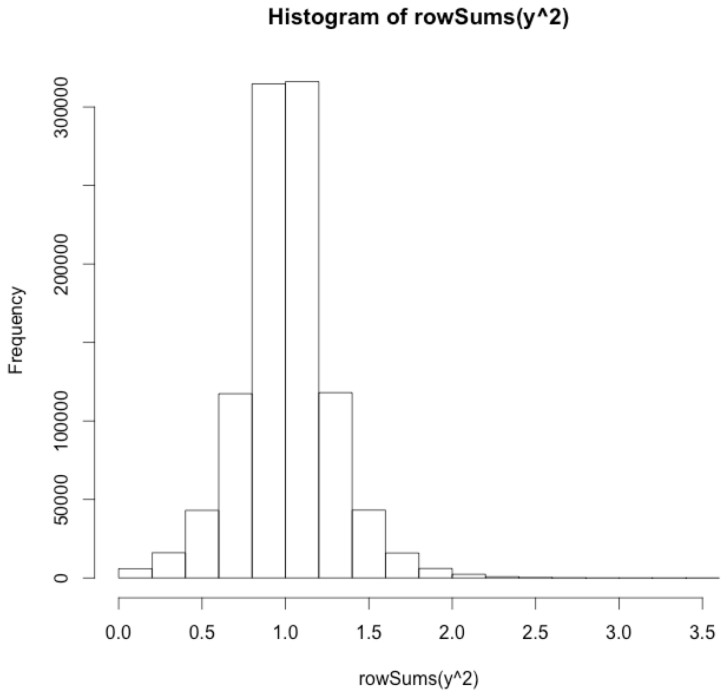




Mixing is decent after thinning by every thousand in the chain. Densities also concentrate around +/-1.



Similarly, the chain prefers points where x1 = x2, which produces locally higher probabilities, given the symmetry of the target probability density function.



The above histogram demonstrates the sum of squared parameters "wants" to be close to 1, to minimize the quantity in the absolute value, and therefore maximize the joint distribution of P(x1, x2).