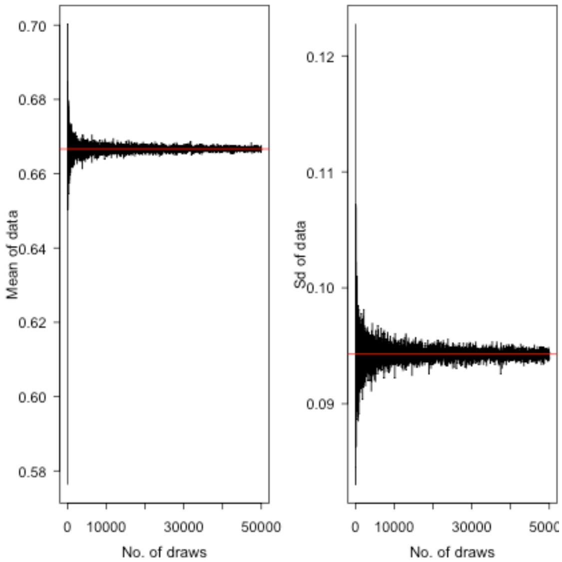
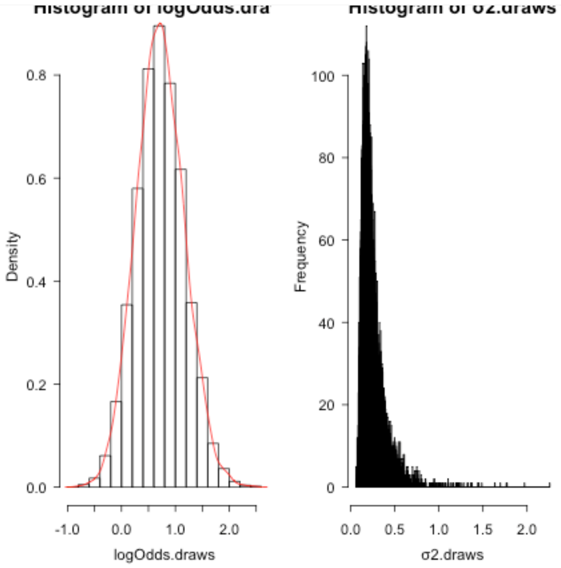
```
######################### Lab1 #########################
### Meta-info
## Beta distribution
## Simulations
## Gini coefficient
## Credible interval
## Highest Posterior Density HPD

# Task 1: Bernoulli ... again
# a) Draw random numbers from Beta distribution and graphical verification
# of posterior
# b) Simulate to compute posterior prob of Pr(theta < 0.4)
# c) Compute log-oods posterior distribution

# Task 2: Log-normal distribution and the Gini coefficient
# a) Simulate 1000 draws from posterior or theta2. Compare with real value
# b) Compute posterior distribution of Gini coefficient G
# c) Compute 95% equal tail credible interval of Gini coefficient G.
# Doing a kernal density estimate
# Compute 95% Highest Posterior Density interval (HPD) of G

# Task 3: Bayesian inference for the concentration parameter in the von Mises distributio
# a) Plot posterior distribution of kappa for wind direction data
# b) Find approximate posterior mode of kappa

############# Task 1 #############
# Bernoulli ... again
#a)
# Instrucitons
# Likelihood: y_1, ..., y_n | θ ~ Bern(θ)
# Prior: θ ~ Beta(alpha_0, beta_0), alpha_0 = beta_0 = 2
# Posterior: θ|y_1, ..., y_n ~ Beta(alpha_0 + s, beta_0 + f)
# s = 14
# n = 20
# f = 6

# Setup
n = 20
s = 14
f = n - s
nDraws = 50000
drawsInterval = 10
intervalVec <- seq(10, nDraws, drawsInterval)

# Prior
prior.alpha = 2
prior.beta = 2

# Posterior
posterior.alpha <- prior.alpha + s
posterior.beta <- prior.beta + f
posterior.draws_means = numeric()
posterior.draws_sd = numeric()

# For-loop - (10, 20, 30, ..., 49980, 49990)
for (i in intervalVec) {
  posterior.draws <- rbeta(n = i,
                           shape1 = posterior.alpha,
                           shape2 = posterior.beta) # Draw from beta

  posterior.draws_means <- c(posterior.draws_means, mean(posterior.draws)) # Add mean to vector of means
  posterior.draws_sd <- c(posterior.draws_sd, sd(posterior.draws)) # Add sd to vector of sd
}

# True values
posterior.true_mean <- posterior.alpha/(posterior.alpha + posterior.beta)
posterior.true_sd <- sqrt((posterior.alpha*posterior.beta)/((posterior.alpha + posterior.beta)^2 * (posterior.alpha + posterior.beta + 

# Plot
par(mfrow = c(1, 2))
plot(x = intervalVec,
     y = posterior.draws_means,
     type = 'l',
     xlab = 'No. of draws',
     ylab = 'Mean of data') # Plot means
abline(h = posterior.true_mean, col = 'red') # Add line of real mean to plot

plot(x = intervalVec,
     y = posterior.draws_sd,
     type = 'l',
     xlab = 'No. of draws',
     ylab = 'Sd of data') # Plot sd's
abline(h = posterior.true_sd, col = 'red')
```

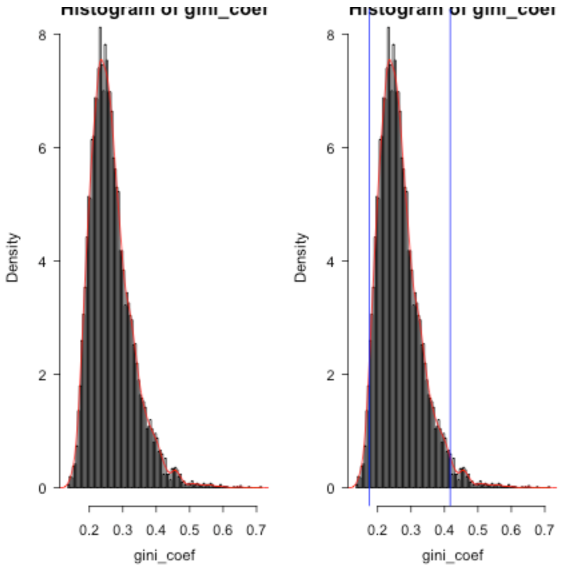



```
# b)
# Setup
n = 20
s = 14
f = n - s
nDraws = 10000

# Prior
prior.alpha = 2
prior.beta = 2

# Posterior
posterior.alpha <- prior.alpha + s
posterior.beta <- prior.beta + f
posterior.draws_means = numeric()
posterior.draws_sd = numeric()

# Draws
posterior.draws_10000 <- rbeta(n = nDraws,
                               shape1 = posterior.alpha,
                               shape2 = posterior.beta)

# Calculate probability
posterior.prob_0.4 <- length(which(posterior.draws_10000 < 0.4))/length(posterior.draws_10000)
posterior.prob_real<- pbeta(q = 0.4,
                            shape1 = posterior.alpha,
                            shape2 = posterior.beta)

# c)
# Functions
logOdds <- function(theta) {
  return (log(theta/(1-theta)))
}

# Setup
n = 20
s = 14
f = n - s
nDraws = 10000

# Prior
prior.alpha = 2
prior.beta = 2

# Posterior
posterior.alpha <- prior.alpha + s
posterior.beta <- prior.beta + f

# Draws
posterior.draws_10000 <- rbeta(n = nDraws,
                               shape1 = posterior.alpha,
                               shape2 = posterior.beta)

# Log-odds the draws
logOdds.draws <-logOdds(posterior.draws_10000)

# Hist and plot the density function of the log-odds draws
hist(logOdds.draws, probability = TRUE)
lines(density(logOdds.draws), col = 'red')

############# Task 2 #############
# Log-normal distribution and the Gini coefficient
# Likelihood: y_1, ..., y_n | μ, σ2 ~ log[ N(μ, σ2) ], μ known, σ2 unknown
# Prior: p(σ2) ∝ 1/σ2

# Posterior of σ2: Inv-X(n, tao^2)
# Tao^2 - The sample variance. Calculated as following:
# sum[ (log(y_i) - μ)^2 ]/n

# If X ~ N(0, 1)
# Y = exp(X) ~ log[ N(0,1) ] (lognormal)

# Setup
y <- c(14, 25, 45, 25, 30, 33, 19, 50, 34, 67)
```

```
nDraws = 10000
mu = 3.5

# a)
# Functions
scaled_rchisq <- function(Y, mu, nDraws) {
  n <- length(Y) # Length of data
  Z <- rchisq(n = nDraws, df = n) # Draw from Chi-squared distribution
  tao2 <- sum((log(Y) - mu)^2)/n # Calculate tau2 (sample standard diviation)

  return (n*tao2/Z)
}

σ2.draws <- scaled_rchisq(y, mu, nDraws) # Draw from Scaled inverse chi-squared distribution
hist(σ2.draws, breaks=1000)
```



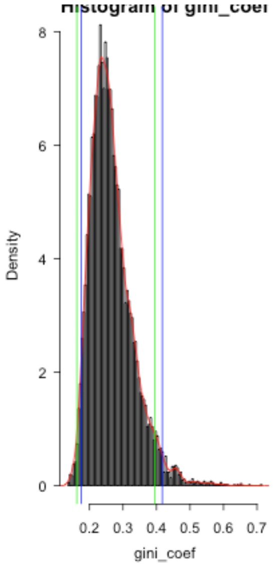

```
# b)
# Gini-coefficient measuers inequality (0: equal, 1: unequal)
# Uses the Lorenz curve where:
# x-axis: Cumulative share of people from low to high income
# y-axis: Cumulative share of income earned
#
# If a straight line, it's 100% equal
#
# The Gini-coefficient is the ratio between the area between the straight line and the Lorenz curve
# divided by the total area
#
# If the data follows an lognormal distribution (e.g wealth), the Gini-coef is calculated as follows:
# G = 2 * Φ(σ/√2) - 1

σ <- sqrt(σ2.draws) # Square σ2 to get σ
gini_coef <- 2 * pnorm(σ/sqrt(2), mean = 0, sd = 1) - 1 # Calculate the Gini-coefficients for each σ
hist(gini_coef, breaks=200, probability = TRUE) # Hist Gini-coefficients
lines(density(gini_coef), col='red') # Plot density curve

# c)
# Functions
eqtail_CI <- function(..X, interval) {
  lower <- (1-interval)/2
  upper <- 1 - lower
  n <- length(..X)
  X <- sort(..X) # Sort from smallest to largest value

  return (list(lower=X[n*lower], upper=X[n*upper]))
}

HPD <- function(density, interval) {
  gini_df <- data.frame(x = density$x, y = density$y)
  gini_df <- gini_df[with(gini_df, order(y)),]
  gini_df <- data.frame(x = gini_df$x, y = gini_df$y)
  n <- dim(gini_df)[1]
  lower <- 1 - interval
  print(lower)
  HPD_cumsum <- cumsum(gini_df$y)/sum(gini_df$y)
  HPD_lower <- which(HPD_cumsum >= lower)[1]
  gini_df_95 <- gini_df[(HPD_lower + 1):n, ]
  HPD_interval <- c(min(gini_df_95$x), max(gini_df_95$x))
  return (list(lower = HPD_interval[1], upper = HPD_interval[2]))
}

# 95% equal tail credible interval
gini_coef_CI <- eqtail_CI(gini_coef, 0.95)

hist(gini_coef, breaks=200, probability = TRUE) # Hist Gini-coefficients
lines(density(gini_coef), col='red') # Plot density curve
abline(v = gini_coef_CI$lower, col='blue') # Plot lower CI line
abline(v = gini_coef_CI$upper, col='blue') # Plot upper CI line
```

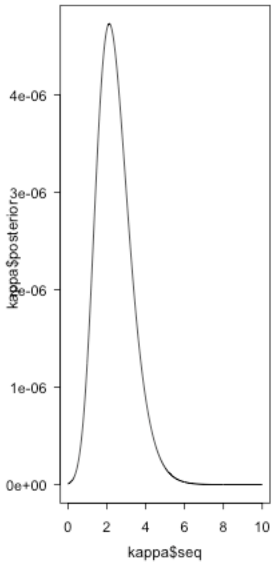



```
# Highest Posterior Density Interval
gini_density <- density(gini_coef)
HPD_interval <- HPD(gini_density, 0.95)
```

```
## [1] 0.05
```

```
# Plot histogram of Gini coefficients, 95% credible interval (blue) and 95% HPD interval (green)
hist(gini_coef, breaks=200, probability = TRUE) # Hist Gini-coefficients
lines(density(gini_coef), col='red') # Plot density curve
abline(v = gini_coef_CI$lower, col='blue') # Plot lower CI line
abline(v = gini_coef_CI$upper, col='blue') # Plot upper CI line
abline(v = HPD_interval$lower, col='green') # Plot lower HPD interval
abline(v = HPD_interval$upper, col='green') # Plot upper HPD interval

############# Task 3 #############
# von Mises distribution looks like a normal distribution with a spiky top and
# is a continues probability distribution on the circle, where theta is an angle.
# Kappa (ϰ): Concentration parameter. Large ϰ gives small variance around μ.
# Likelihood: p(y_1, ..., y_n | μ, ϰ) = exp[ ϰ * cos(y - μ) ]/(2πIo(ϰ))
# Prior: ϰ ~ Exp(λ = 1), mean = 1/λ

# Setup
# Wind-angles in degrees on 10 different days
# North is zero
#
y.degrees <- c(40, 303, 326, 285, 296, 314, 20, 308, 299, 296)
y.radians <- c(-2.44, 2.14, 2.54, 1.83, 2.02, 2.33, -2.79, 2.23, 2.07, 2.02)
mu <- 2.39 # Mean directon

# a)
# Prior
kappa <- data.frame(seq = seq(0, 10, 0.01),
                    posterior = 0
)

for (i in 1:dim(kappa)[1]) { # Loop over every kappa
  k <- kappa$seq[i] # Extract current kappa
  prior <- exp(-k) # Calculate prior with current kappa
  bessel <- besselI(x = k,
                    nu = 0) # Bessel-function
  likelihood <- prod(exp(k * cos(y.radians - mu))/(2*pi*bessel)) # Calculate von Mises probability
  kappa$posterior[i] <- likelihood * prior # Calculate posterior with current kappa
}

# Plot posterior for different kappas
plot(kappa$seq, kappa$posterior, type='l')
```






```r
# b)
index <- which.max(kappa$posterior) # Finds index with maximum posterior
kappa.mode <- kappa$seq[which.max(kappa$posterior)] # Extract kappa with maximum posterior

plot(kappa$seq, kappa$posterior, type='l')
abline(v = kappa.mode, col='red')
```