STAT230 HW 3
University of California, Berkeley

Thibault Doutre, Student ID 26980469

February 22, 2016

1 Theory

1.1 A1

$$\begin{aligned} E(Y|X) &= E(X\beta + \epsilon|X) && (1)\\ &= E(X\beta|X) + E(\epsilon|X) && (2)\\ &= E(X\beta|X) && (3)\\ &= X\beta && (4) \end{aligned}$$

$$\begin{aligned} Cov(Y|X) &= E(YY^T|X) - E(Y|X)E(Y|X)^T && (5)\\ &= X\beta\beta^T X^T - 2E(X\beta\epsilon^T|X) + E(\epsilon\epsilon^T|X) - X\beta(X\beta)^T && (6)\\ &= -2X\beta E(\epsilon^T|X) + E(\epsilon\epsilon^T|X) && (7)\\ &= \sigma^2 I && (8) \end{aligned}$$

$$\begin{aligned} \hat{\beta} &= (X^T X)^{-1} X^T Y && (9)\\ &= (X^T X)^{-1} X^T (X\beta + \epsilon) && (10)\\ &= \beta + (X^T X)^{-1} X^T \epsilon && (11) \end{aligned}$$

Therefore:

$$\begin{aligned} E(\hat{\beta}|X) &= E(\beta + (X^T X)^{-1} X^T \epsilon|X) && (12)\\ &= E(\beta|X) + (X^T X)^{-1} X^T E(\epsilon|X) && (13)\\ &= \beta && (14) \end{aligned}$$

$$
\begin{aligned}
Cov(\hat{\beta}|X) &= E(\hat{\beta}\hat{\beta}^T|X) - E(\hat{\beta}|X)E(\hat{\beta}|X)^T && (15)\\
&= E((X^TX)^{-1}X^TYY^TX(X^TX)^{-1}|X) && (16)\\
&- E((X^TX)^{-1}X^TY)E((X^TX)^{-1}X^TY|X)^T && (17)\\
&= (X^TX)^{-1}X^TE(YY^T|X)X(X^TX)^{-1} - \beta\beta^T && (18)\\
&= (X^TX)^{-1}X^TX\beta\beta^TX^TX(X^TX)^{-1} && (19)\\
&+ (X^TX)^{-1}X^TE(\epsilon\epsilon^T|X)X(X^TX)^{-1} - \beta\beta^T && (20)\\
&= (X^TX)^{-1}X^T(\sigma^2I)X(X^TX)^{-1} && (21)\\
&= \sigma^2(X^TX)^{-1} && (22)
\end{aligned}
$$

1.2 B2

$$
\begin{aligned}
E(Y|X) &= E(X\beta + \epsilon|X) && (23)\\
&= E(X\beta|X) + E(\epsilon|X) && (24)\\
&= E(X\beta|X) && (25)\\
&= X\beta && (26)
\end{aligned}
$$

$$
\begin{aligned}
Cov(Y|X) &= E(YY^T|X) - E(Y|X)E(Y|X)^T && (27)\\
&= X\beta\beta^TX^T - 2E(X\beta\epsilon^T|X) + E(\epsilon\epsilon^T|X) - X\beta(X\beta)^T && (28)\\
&= -2X\beta E(\epsilon^T|X) + E(\epsilon\epsilon^T|X) && (29)\\
&= G && (30)
\end{aligned}
$$

$$
\begin{aligned}
\hat{\beta} &= (X^TX)^{-1}X^TY && (31)\\
&= (X^TX)^{-1}X^T(X\beta + \epsilon) && (32)\\
&= \beta + (X^TX)^{-1}X^T\epsilon && (33)
\end{aligned}
$$

Therefore:

$$
\begin{aligned}
E(\hat{\beta}|X) &= E(\beta + (X^TX)^{-1}X^T\epsilon|X) && (34)\\
&= E(\beta|X) + (X^TX)^{-1}X^TE(\epsilon|X) && (35)\\
&= \beta && (36)
\end{aligned}
$$

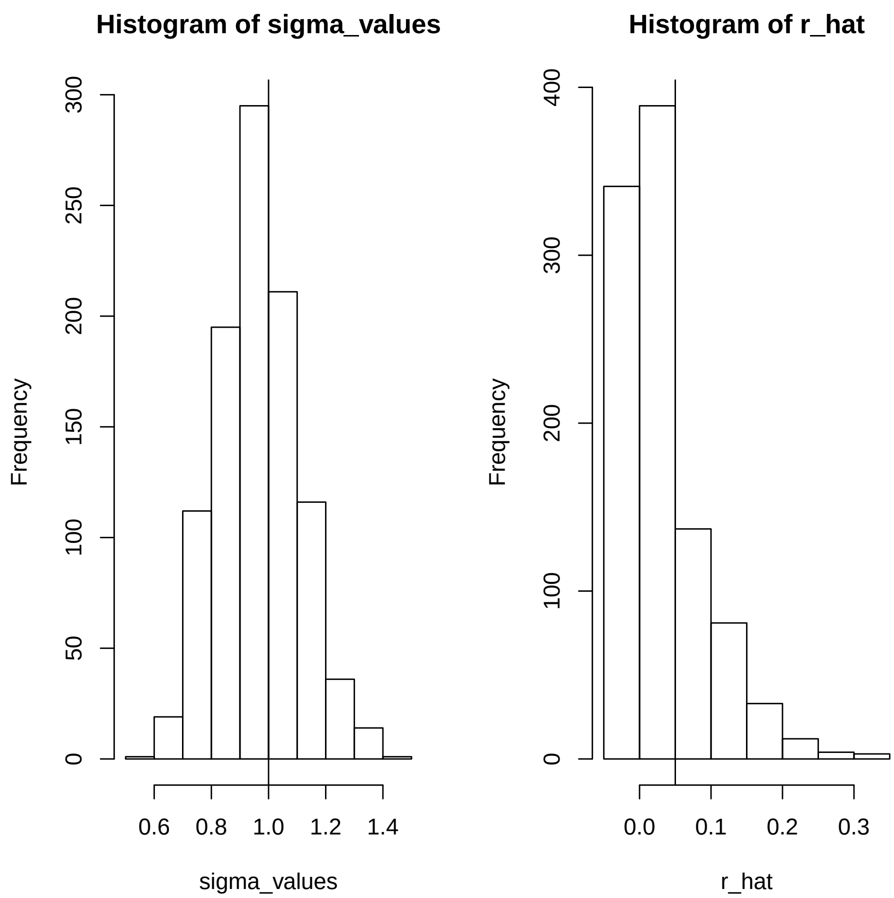
$$
\begin{aligned}
Cov(\hat{\beta}|X) &= E(\hat{\beta}\hat{\beta}^T|X) - E(\hat{\beta}|X)E(\hat{\beta}|X)^T && (37)\\
&= E((X^TX)^{-1}X^TYY^TX(X^TX)^{-1}|X) && (38)\\
&- E((X^TX)^{-1}X^TY)E((X^TX)^{-1}X^TY|X)^T && (39)\\
&= (X^TX)^{-1}X^TE(YY^T|X)X(X^TX)^{-1} - \beta\beta^T && (40)\\
&= (X^TX)^{-1}X^TX\beta\beta^TX^TX(X^TX)^{-1} && (41)\\
&+ (X^TX)^{-1}X^TE(\epsilon\epsilon^T|X)X(X^TX)^{-1} - \beta\beta^T && (42)\\
&= (X^TX)^{-1}X^TGX(X^TX)^{-1} && (43)
\end{aligned}
$$

2 Code

2.1 Generate correlated errors

```
### 1. Generate correlated errors.

n = 100
r = 0.05
G = matrix(r,n,n) + diag(1-r,n,n)
chol_G = chol(G)

set.seed(12345)
n_replications = 1000
epsilons = matrix(rnorm(n*n_replications),n,n_replications)
epsilons = as.data.frame(chol_G %*% epsilons)
head(epsilons)[,1:6]

##            V1         V2          V3        V4          V5         V6
## 1  1.7822384  0.4388947 -1.59539633 1.5725475  0.36061310 -1.0285078
## 2  1.8131346 -0.8953112 -0.75002287 1.0083173 -0.25226318 -1.9738353
## 3  0.9490665  0.6496476  0.11600482 0.5325667  0.02823925  0.9287940
## 4  0.5787599 -1.0447429  0.88759743 2.0722387 -1.19905224 -0.4686916
## 5  1.5649402  0.3983768  0.63320327 0.7274321 -0.78107700  3.6242797
## 6 -0.8137008 -0.2645429 -0.08610409 0.8463991  0.63117078  0.1398695

S_n = colSums(epsilons)
sigma_values = diag(cov(epsilons))
```

```r
# Var(Sn)
var(S_n)

## [1] 481.5239

par(mfrow = c(1,2))

# Empirical value of sigma
sigma_hat = mean(sigma_values)
sigma_hat

## [1] 0.9628051

hist(sigma_values)
abline(v = 1)

# Empirical value of r
non_diag_terms = (matrix(1,n,n_replications)-diag(1,n,n_replications))>0
var_hat = vector(length = 1000)
r_hat = vector(length = 1000)
for(i in 1:1000){
  cov_hat    = epsilons[,i] %*% t(epsilons[,i])
  var_hat[i] = mean(diag(cov_hat))
  r_hat[i] = mean(cov_hat[lower.tri(cov_hat, diag = FALSE)]/var_hat[i])
}
mean(r_hat)

## [1] 0.03507652

hist(r_hat)
abline(v = r)
```



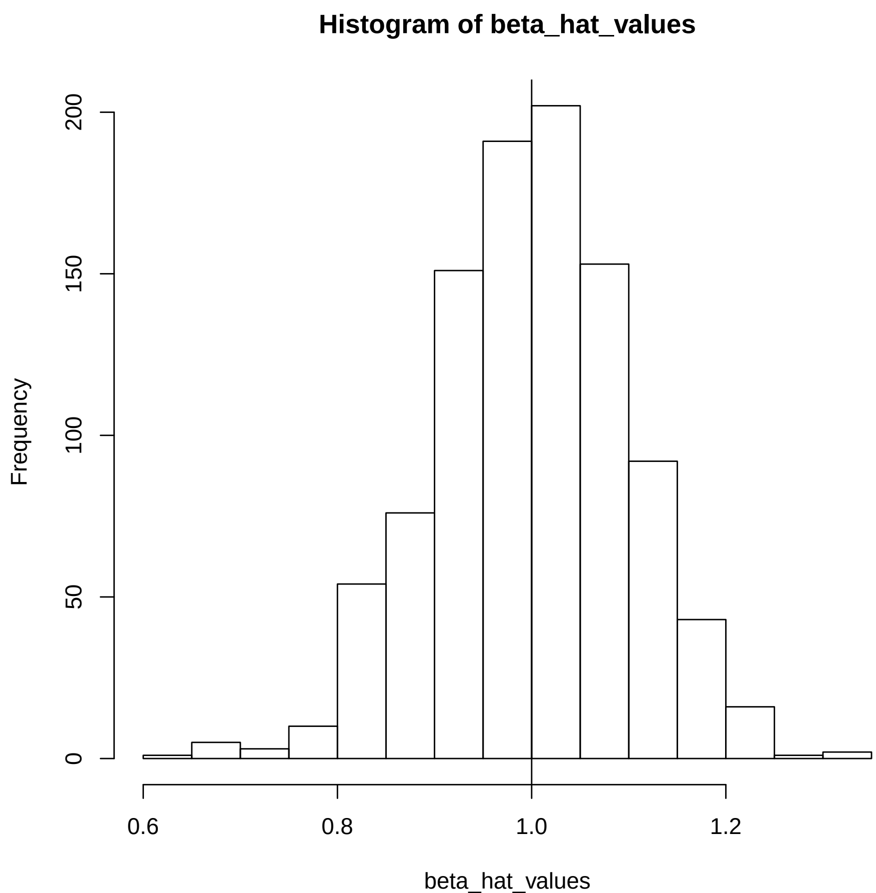

```
par(mfrow = c(1,1))
```

```
# Relationship between var(Sn), r, sigma satisfied?
abs(var(S_n)-n*sigma_hat^2-n*(n-1)*sigma_hat^2*r)/var(S_n)

## [1] 0.1454494

# True value of S_n
true_VarSn = n*1+n*(n-1)*1*r
true_VarSn

## [1] 595
```

```
# Relative error
abs(true_VarSn- var(S_n))/true_VarSn

## [1] 0.1907162
```

2.2 OLS

```
X = matrix(rnorm(100000),100,1000)
Y = X + epsilons

ols = function(X,Y){
  n = nrow(X)
  p = ncol(X)
  beta_hat_values = vector(length = p)
  epsilon_values = matrix(nrow = n, ncol = p)
  for (i in 1:p){
    Xi = X[,i]
    Yi = Y[,i]
    beta_hat_values[i] = solve(t(Xi) %*% Xi) %*% t(Xi)%*%Yi
    epsilon_values[,i] = Yi-beta_hat_values[i] %*% Yi
  }
  list(beta_hat_values = beta_hat_values, epsilons = epsilon_values)
}

beta_hat_values = ols(X,Y)$beta_hat_values

# Estimate of beta
mean(beta_hat_values)

## [1] 0.9997003

hist(beta_hat_values)
abline(v = 1)
```



Here, OLS does pretty well since the estimatonr of β is not biased and the variance is relatively low, as the histogram shows.

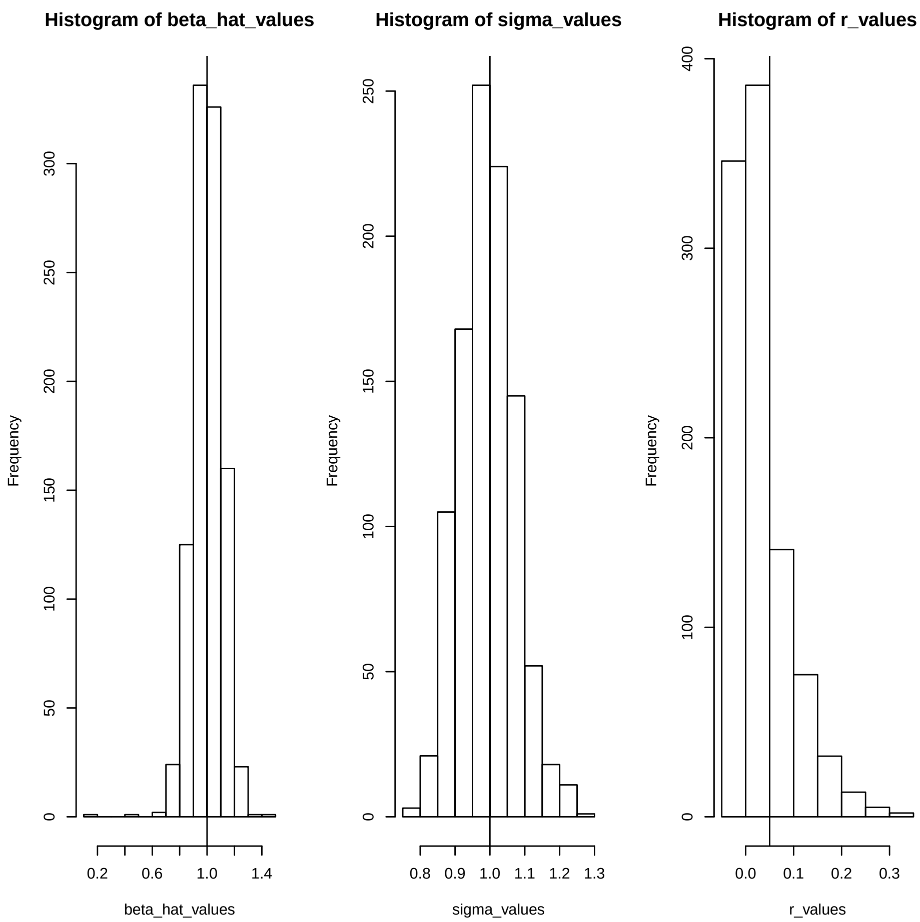
2.3 One Step GLS

```
X = matrix(rnorm(100000),100,1000)
Y = X + epsilons

gls = function(X,Y,epsilons, beta_hat_ols){
  n = nrow(X)
  p = ncol(X)
  non_diag_terms = (matrix(1,n,p)-diag(1,n,p))>0
```

```
  beta_hat_values = beta_hat_ols
  var_hat_values = vector(length = p)
  r_hat_values = vector(length = p)
  epsilons_update = epsilons
  for (i in 1:p){
    # Residuals
    e = Y[,i] - X[,i]*beta_hat_values[i]
    # Empirical value of cov
    cov_hat = e %*% t(e)
    # Empirical value of var
    var_hat_values[i] = mean(diag(cov_hat))
    # Empirical value of r
    r_hat_values[i] = mean(cov_hat[
      lower.tri(cov_hat, diag = FALSE)]/var_hat_values[i])
    # Estimate G
    G_hat = matrix(r_hat_values[i],n,n) +
      diag(var_hat_values[i]-r_hat_values[i],n,n)
    G_hat_inv = solve(G_hat)
    # Add beta hat value
    Xi = X[,i]
    Yi = Y[,i]
    beta_hat_values[i] = solve(t(Xi) %*% G_hat_inv %*% Xi) %*%
      t(Xi) %*% G_hat_inv %*% Yi
    # Update epsilon
    epsilons_update[,i] = G_hat_inv %*% e
  }
  return(list(beta_hat_values = beta_hat_values,
              epsilons = epsilons_update,
              r_hat_values = r_hat_values,
              var_hat_values = var_hat_values))
}

beta_hat_ols = ols(X,Y)$beta_hat_values
gls_out = gls(X,Y,epsilons,beta_hat_ols)
```

Here, we see that the estimators of β, σ and r are not biased. For β, we can say that the estimator is as good as the OLS.

```r
par(mfrow = c(1,3))
# Estimate of beta
beta_hat_values = gls_out$beta_hat_values
mean(beta_hat_values)

## [1] 1.001561

hist(beta_hat_values)
abline(v = 1)

# Empirical value of sigma
sigma_values = sqrt(gls_out$var_hat_values)
mean(sigma_values)

## [1] 0.9923265

hist(sigma_values)
abline(v = 1)

# Empirical value of r
r_values = gls_out$r_hat_values
mean(r_values)

## [1] 0.03493974

hist(r_values)
abline(v = r)
```

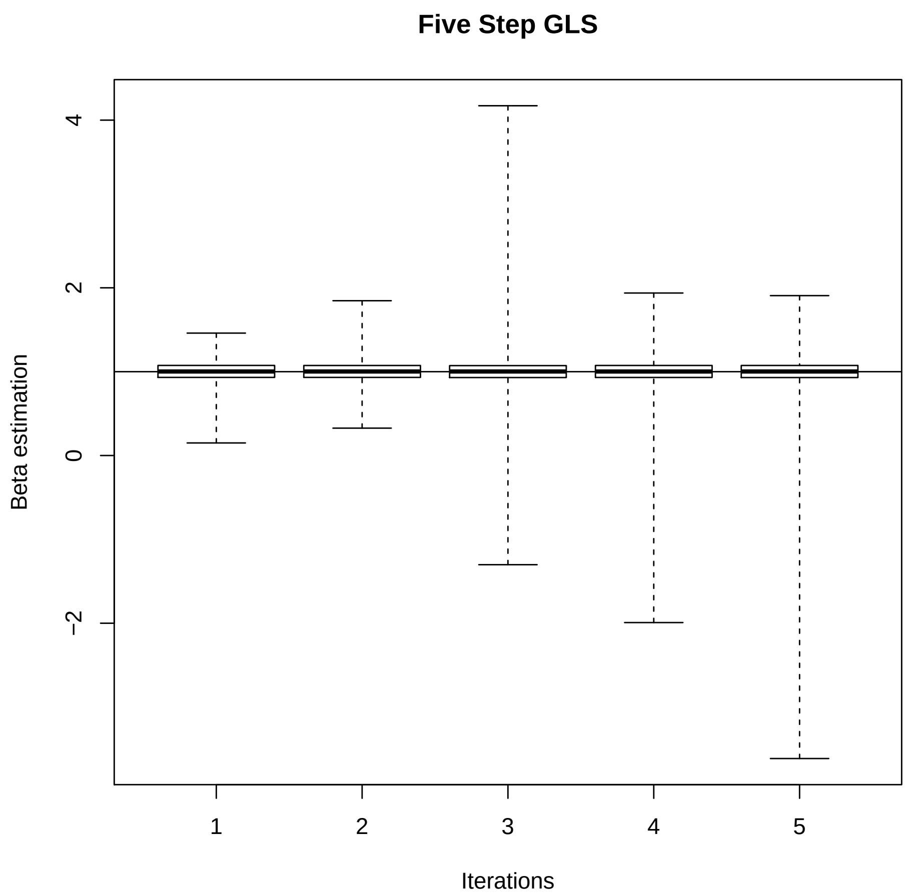



```
par(mfrow = c(1,1))
```

2.4 Five Step GLS

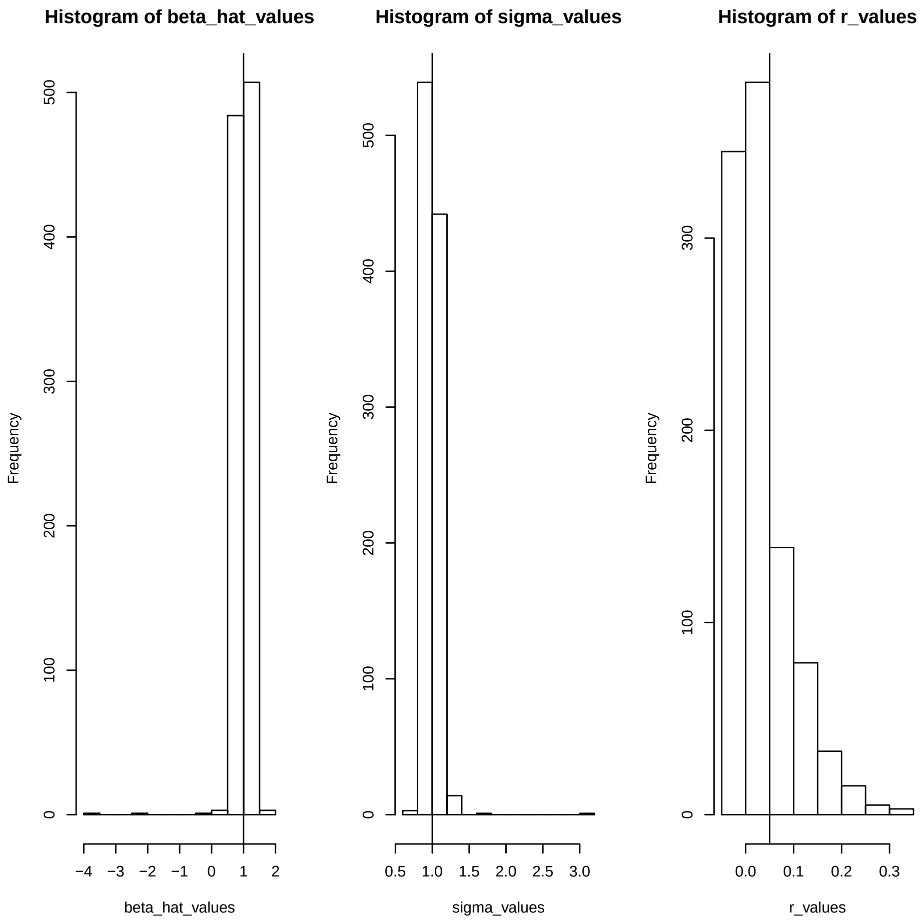
```
### 4. Five Step GLS

gls_it = function(X,Y,n_it){
  # Step zero: ols
  ols_out = ols(X,Y)
  epsilons_update = ols_out$epsilons
```

```r
  beta_hat_update = ols_out$beta_hat_values
  # Iterations
  beta_hat_values_it = matrix(nrow = 1000, ncol = n_it)
  var_hat_values_it = matrix(nrow = 1000, ncol = n_it)
  r_hat_values_it = matrix(nrow = 1000, ncol = n_it)
  for (j in 1:n_it){
    gls_out = gls(X,Y,epsilons_update,beta_hat_update)
    beta_hat_values_it[,j] = gls_out$beta_hat_values
    var_hat_values_it[,j] = gls_out$var_hat_values
    r_hat_values_it[,j] = gls_out$r_hat_values
    epsilons_update = gls_out$epsilons
    beta_hat_update = beta_hat_values_it[,j]
    cat('Step ',j,' done.\n')
  }
  list(beta_hat_values_it = beta_hat_values_it,
       var_hat_values_it = var_hat_values_it,
       r_hat_values_it = r_hat_values_it)
}
gls_it_out = gls_it(X,Y,5)

## Step  1  done.
## Step  2  done.
## Step  3  done.
## Step  4  done.
## Step  5  done.

beta_hat_values_it = gls_it_out$beta_hat_values_it
var_hat_values_it = gls_it_out$var_hat_values_it
r_hat_values_it = gls_it_out$r_hat_values_it

boxplot(beta_hat_values_it, main = "Five Step GLS",
        xlab = "Iterations", ylab = "Beta estimation",
        range=0)
abline(h=1)
```




```r
par(mfrow = c(1,3))
# Estimate of beta
beta_hat_values = beta_hat_values_it[,5]
mean(beta_hat_values)

## [1] 0.9913742

hist(beta_hat_values)
abline(v = 1)

# Empirical value of sigma
sigma_values = sqrt(var_hat_values_it[,5])
```

```r
sigma_hat = mean(sigma_values)
sigma_hat

## [1] 0.996248

hist(sigma_values)
abline(v = 1)

# Empirical value of r
r_values = r_hat_values_it[,5]
r_hat = mean(r_values)
r_hat

## [1] 0.03568166

hist(r_values)
abline(v = r)
```




```r
par(mfrow = c(1,1))
```

Here, the estimate is unbiased, and we see that the first estimation gives pretty accurate results. Moreover, the 5 steps make no improvement. We have more extreme values at the fith step than at the first one. It makes sense because here the G matrix is highly constrained and by updating it we allow more variance in the estimate.