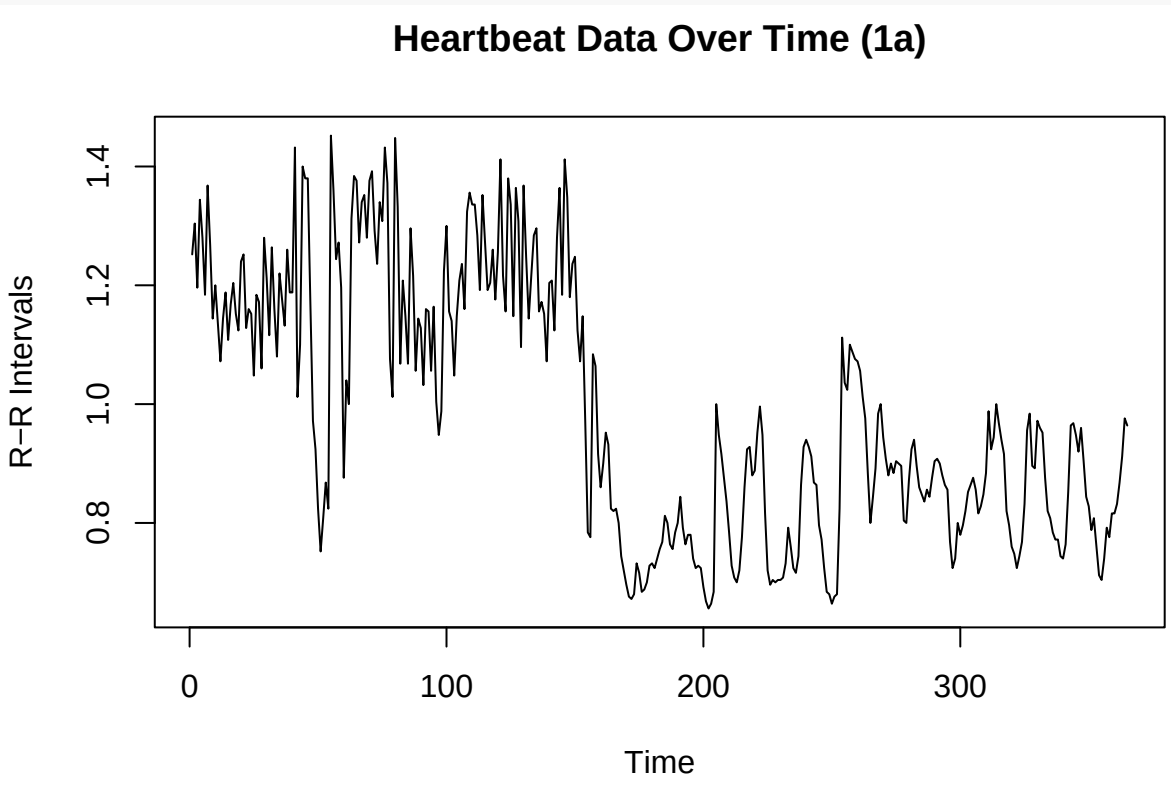
568Project

Reetom Gangopadhyay

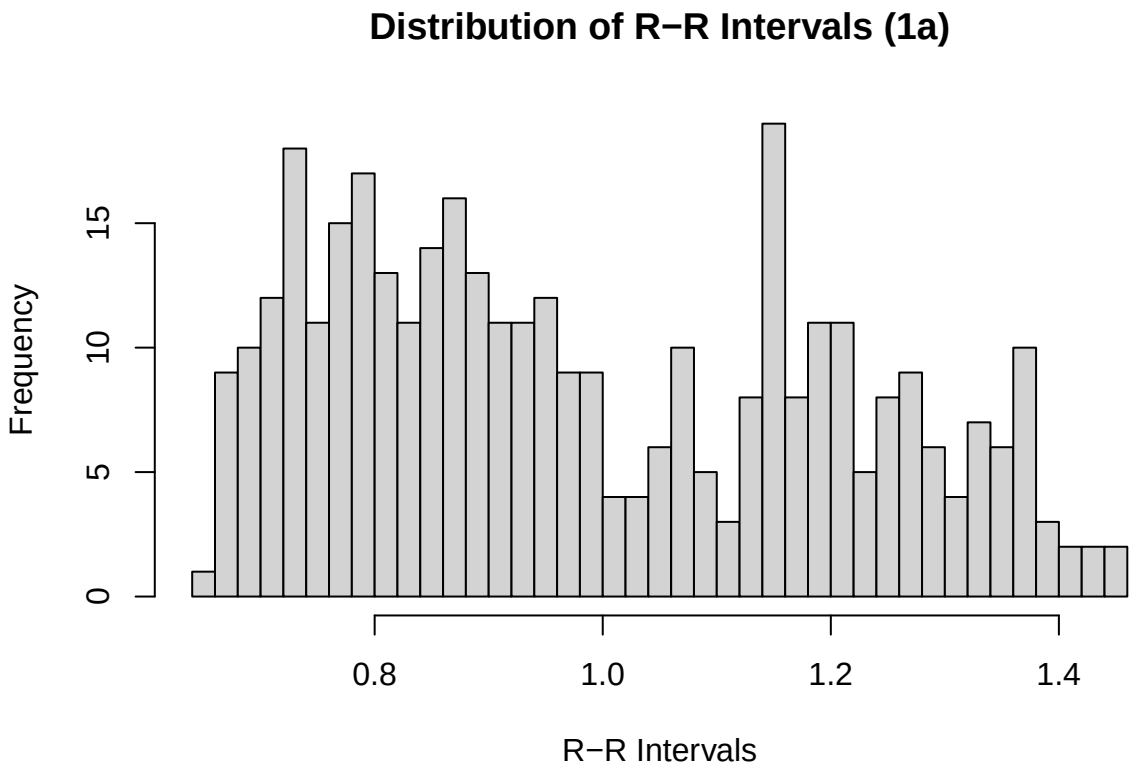
2023-11-29

```
## subject 1
data1a <- read.table('data/subj1a.txt', header = FALSE)
data1b <- read.table('data/subj1b.txt',header = FALSE)

## subject 2
data2a <- read.table('data/subj2a.txt', header = FALSE)
data2b <- read.table('data/subj2b.txt',header = FALSE)

## subject 3
data3a <- read.table('data/subj3a.txt', header = FALSE)
data3b <- read.table('data/subj3b.txt',header = FALSE)

## subject 4
data4a <- read.table('data/subj4a.txt', header = FALSE)
data4b <- read.table('data/subj4b.txt',header = FALSE)
```

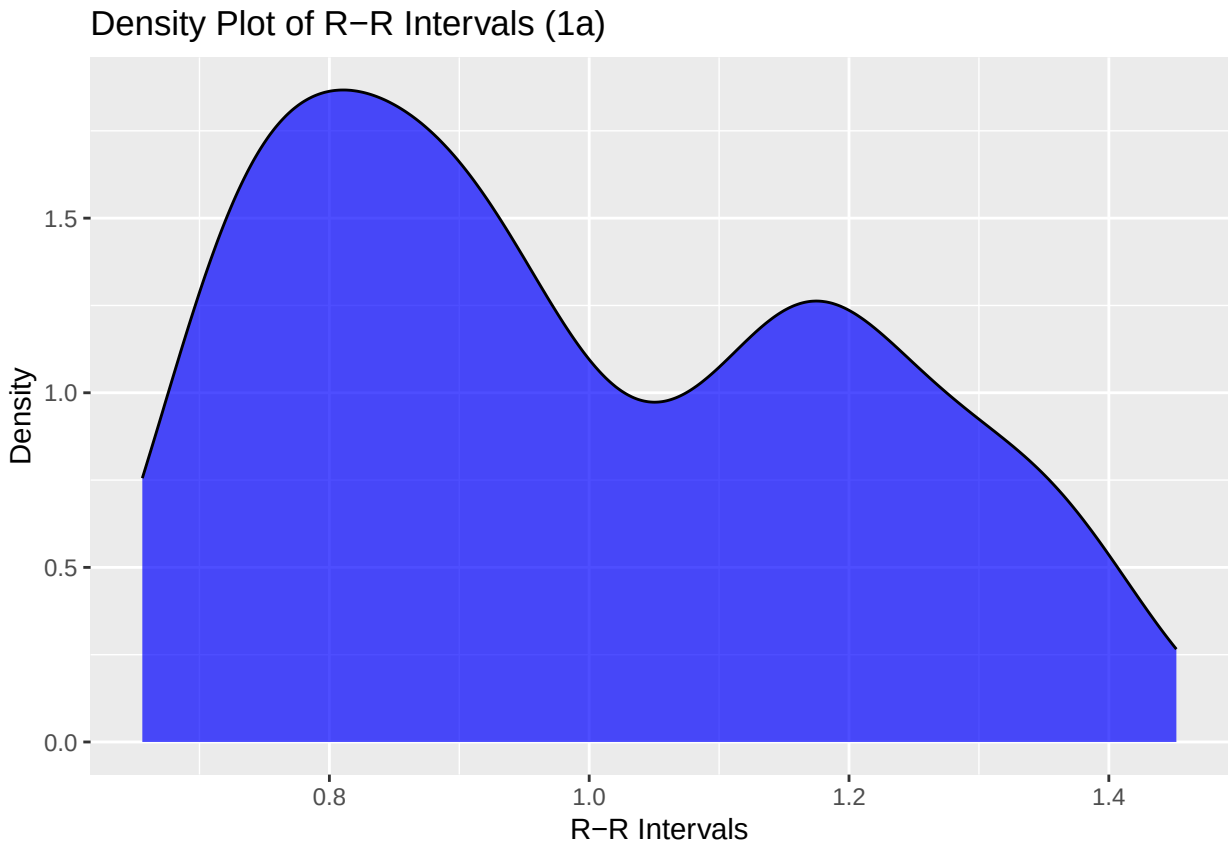
```
plot(data1a$V2, type = "l", xlab = "Time", ylab = "R-R Intervals", main = "Heartbeat Data Over Time (1a)
```



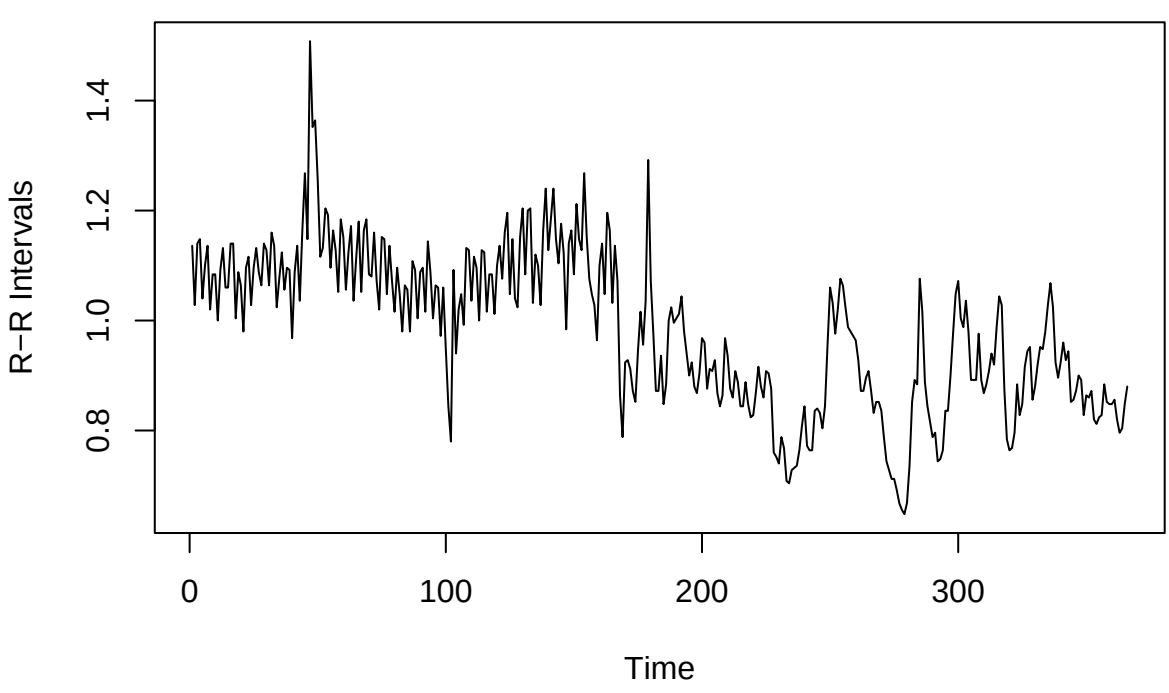
```r
hist(data1a$V2, breaks = 30, xlab = "R-R Intervals", main = "Distribution of R-R Intervals (1a)")

library(ggplot2)
```



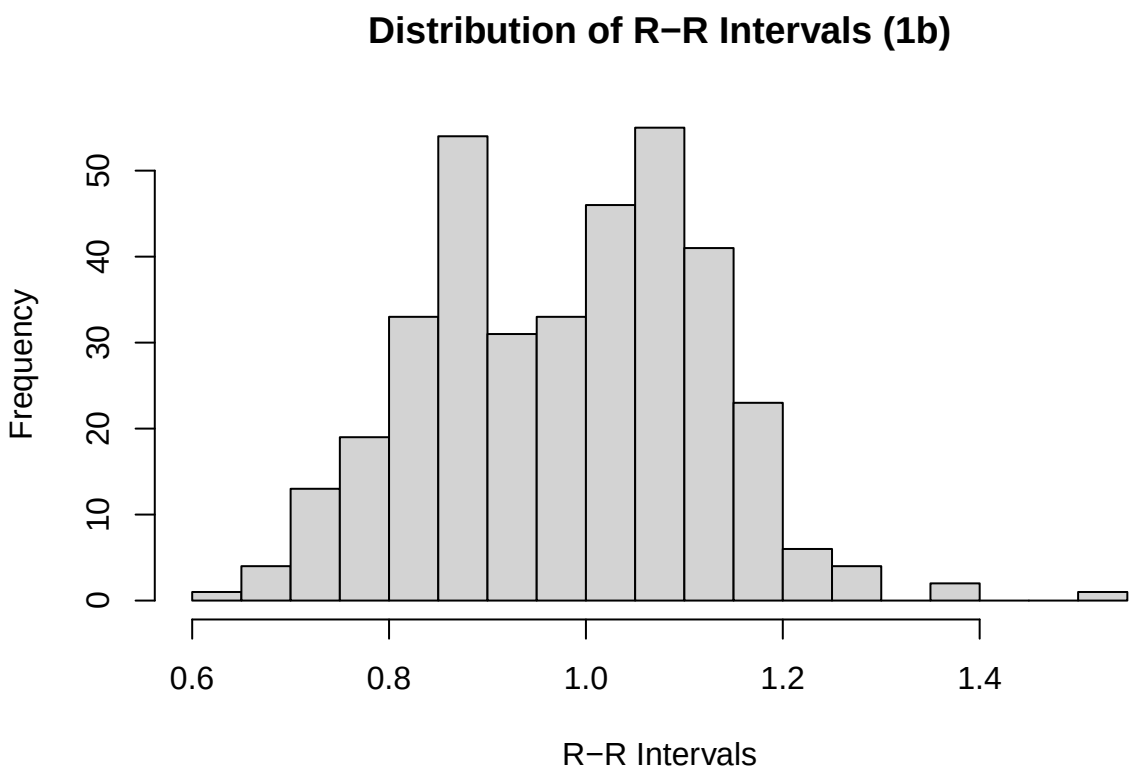

```r
ggplot(data1a, aes(x = V2)) + geom_density(fill = "blue", alpha = 0.7) +
  labs(x = "R-R Intervals", y = "Density", title = "Density Plot of R-R Intervals (1a)")
```



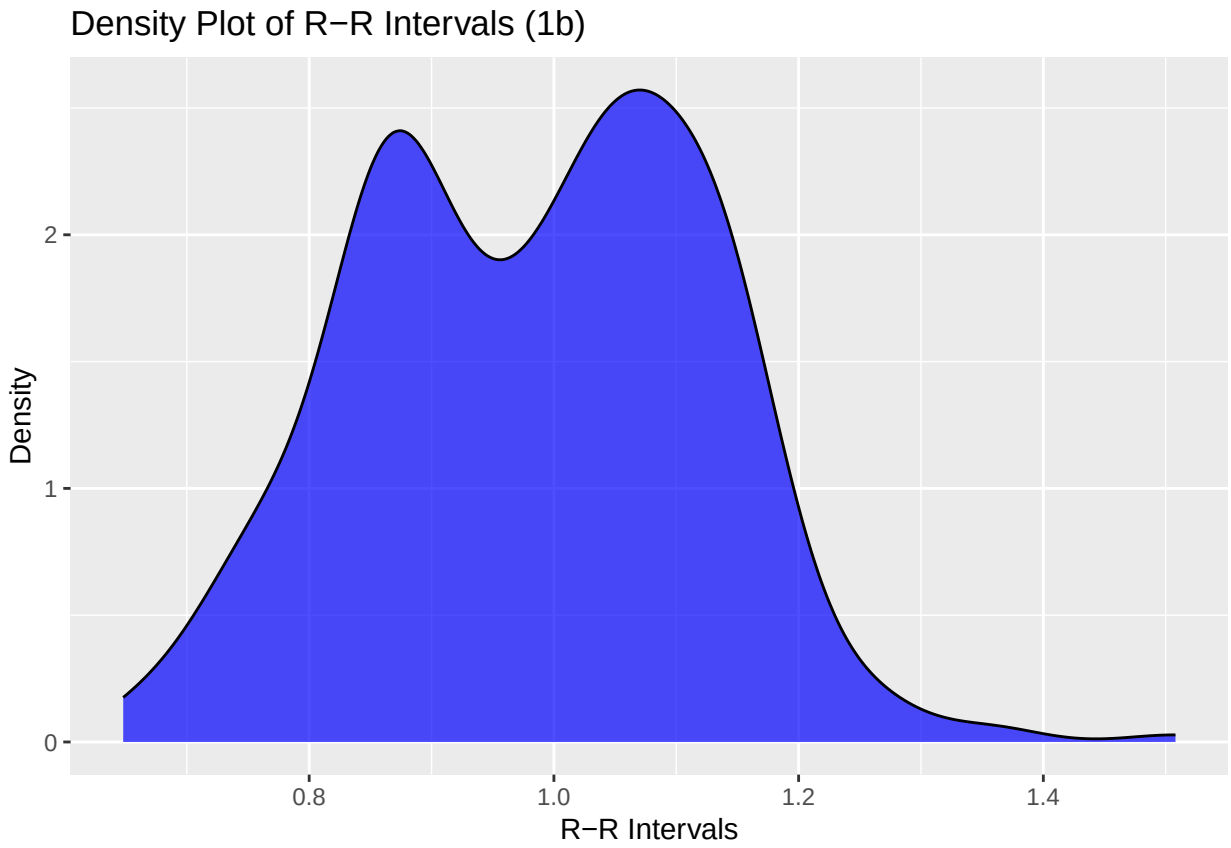

```
plot(data1b$V2, type = "l", xlab = "Time", ylab = "R-R Intervals", main = "Heartbeat Data Over Time (1b)
```



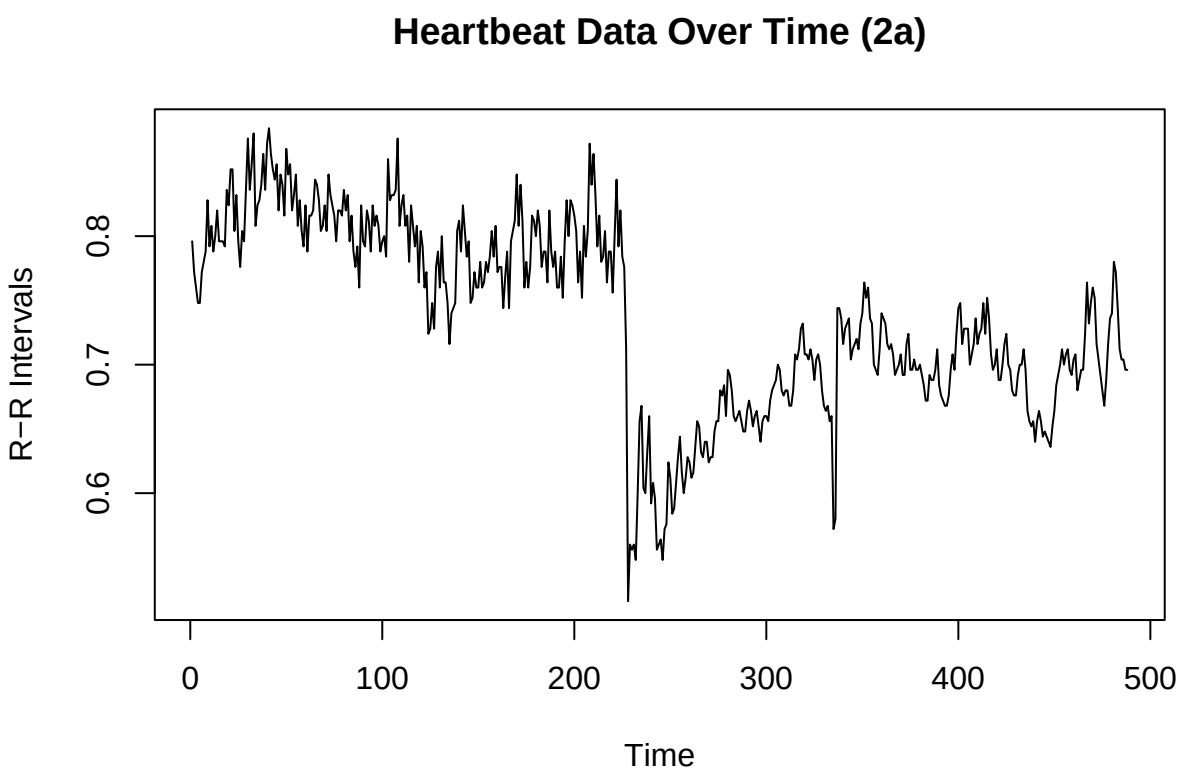
```r
hist(data1b$V2, breaks = 30, xlab = "R-R Intervals", main = "Distribution of R-R Intervals (1b)")
```



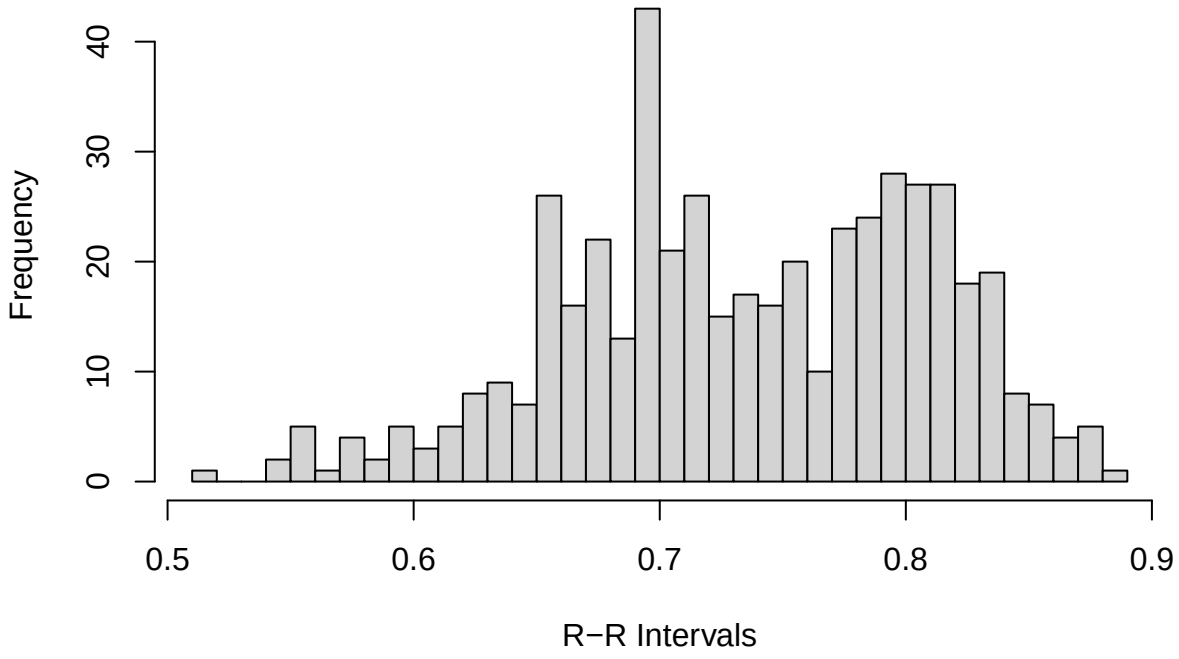

```r
library(ggplot2)
ggplot(data1b, aes(x = V2)) + geom_density(fill = "blue", alpha = 0.7) +
  labs(x = "R-R Intervals", y = "Density", title = "Density Plot of R-R Intervals (1b)")
```



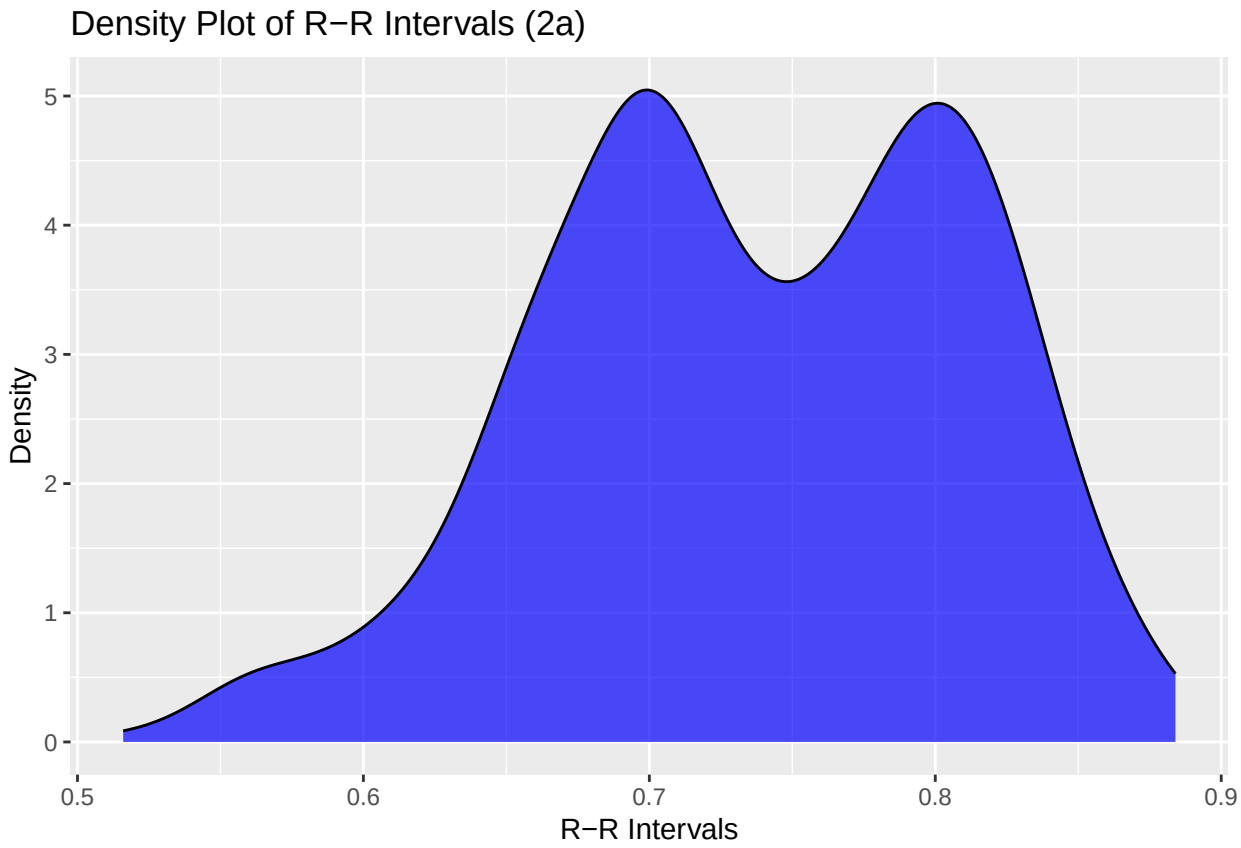

```
plot(data2a$V2, type = "l", xlab = "Time", ylab = "R-R Intervals", main = "Heartbeat Data Over Time (2a)
```



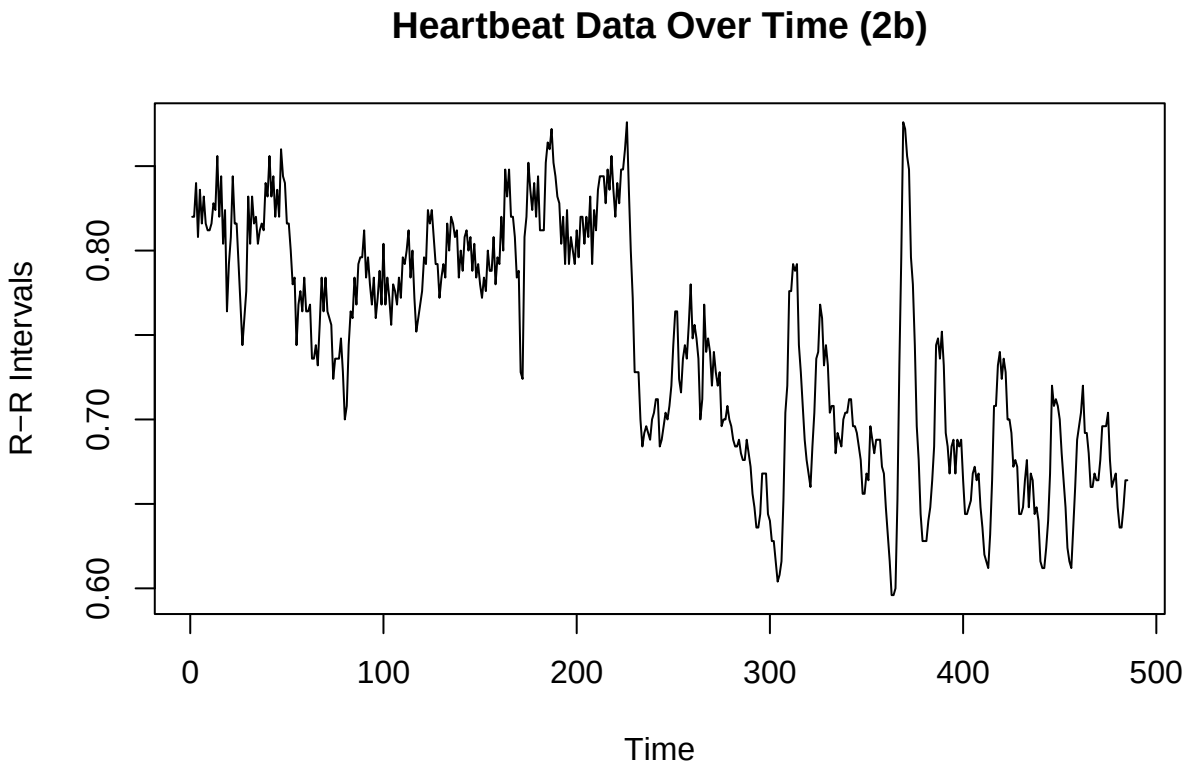
```r
hist(data2a$V2, breaks = 30, xlab = "R-R Intervals", main = "Distribution of R-R Intervals (2a)")
```



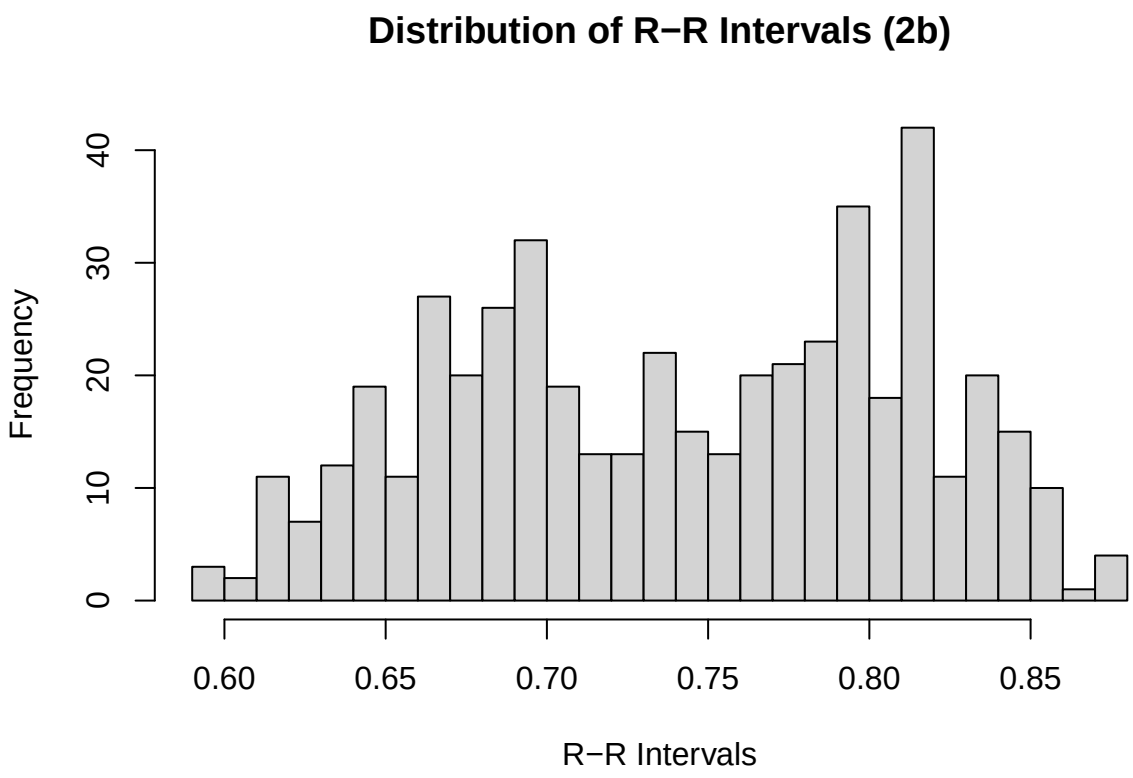

```r
library(ggplot2)
ggplot(data2a, aes(x = V2)) + geom_density(fill = "blue", alpha = 0.7) +
  labs(x = "R-R Intervals", y = "Density", title = "Density Plot of R-R Intervals (2a)")
```



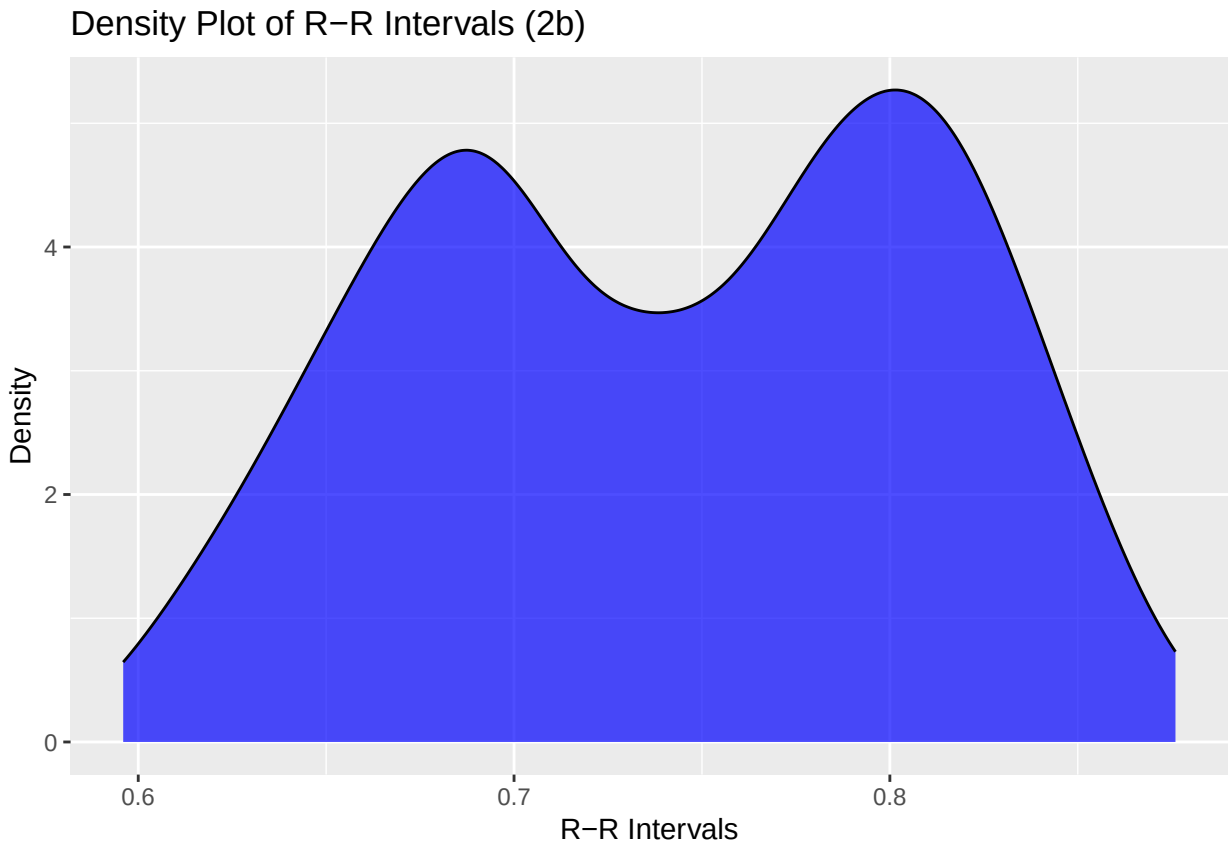

```
plot(data2b$V2, type = "l", xlab = "Time", ylab = "R-R Intervals", main = "Heartbeat Data Over Time (2b)
```



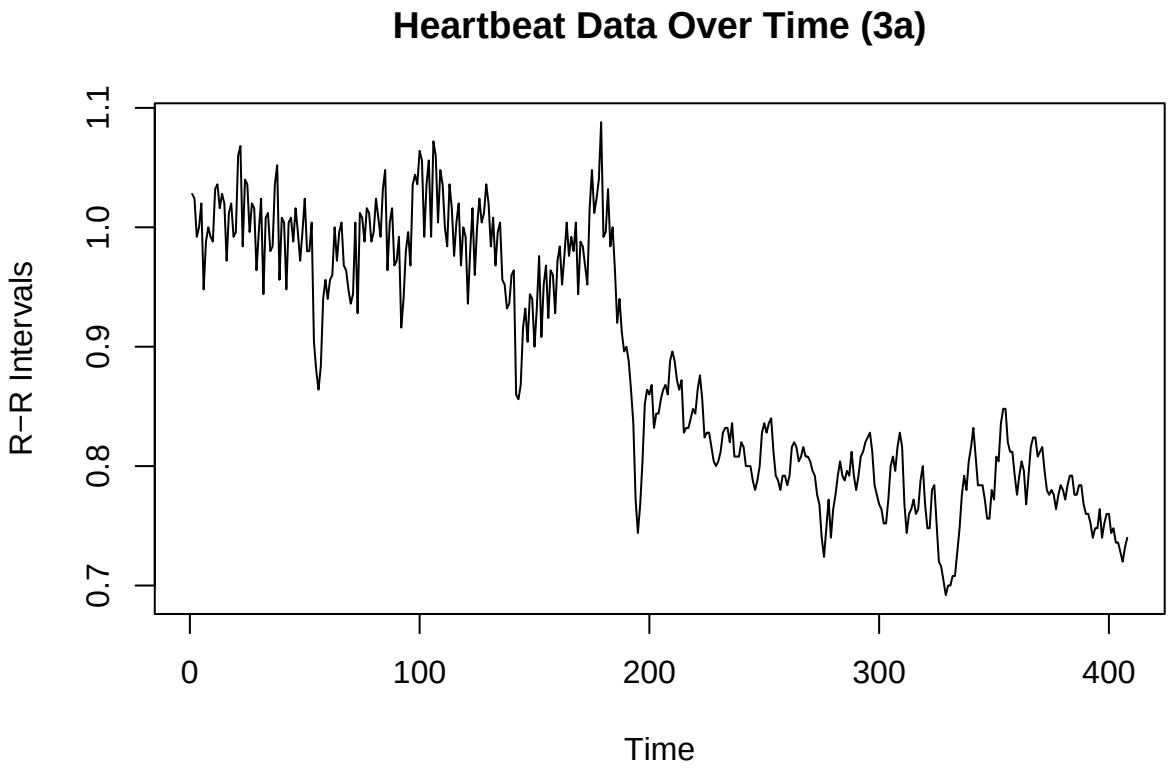
```r
hist(data2b$V2, breaks = 30, xlab = "R-R Intervals", main = "Distribution of R-R Intervals (2b)")
```



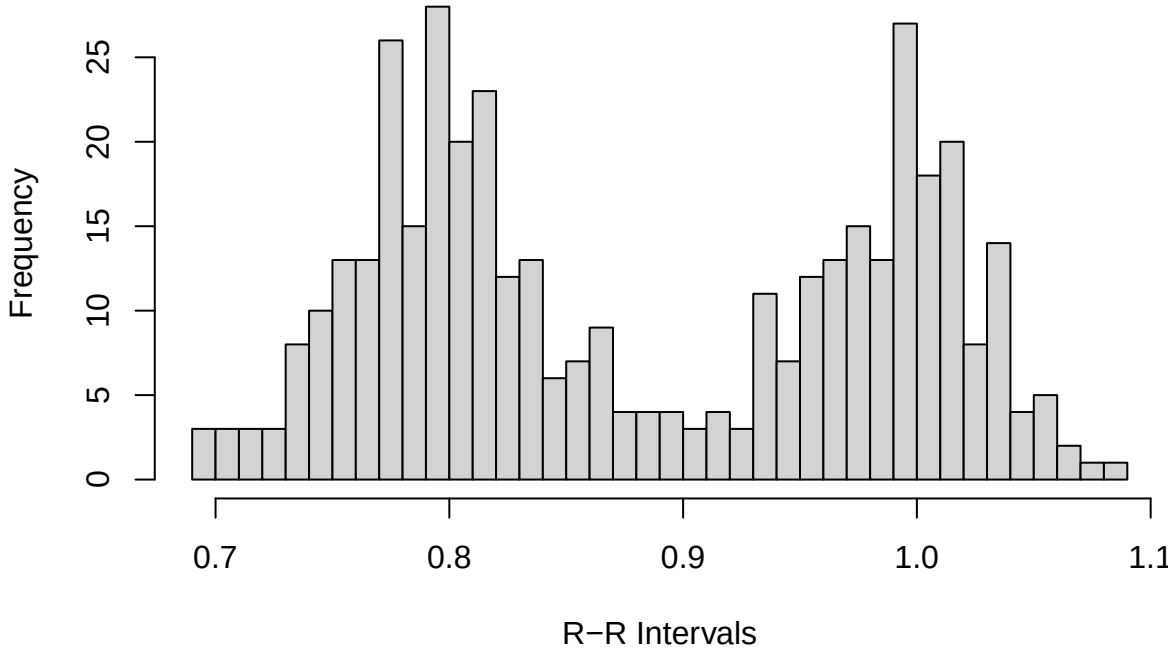

```r
library(ggplot2)
ggplot(data2b, aes(x = V2)) + geom_density(fill = "blue", alpha = 0.7) +
  labs(x = "R-R Intervals", y = "Density", title = "Density Plot of R-R Intervals (2b)")
```



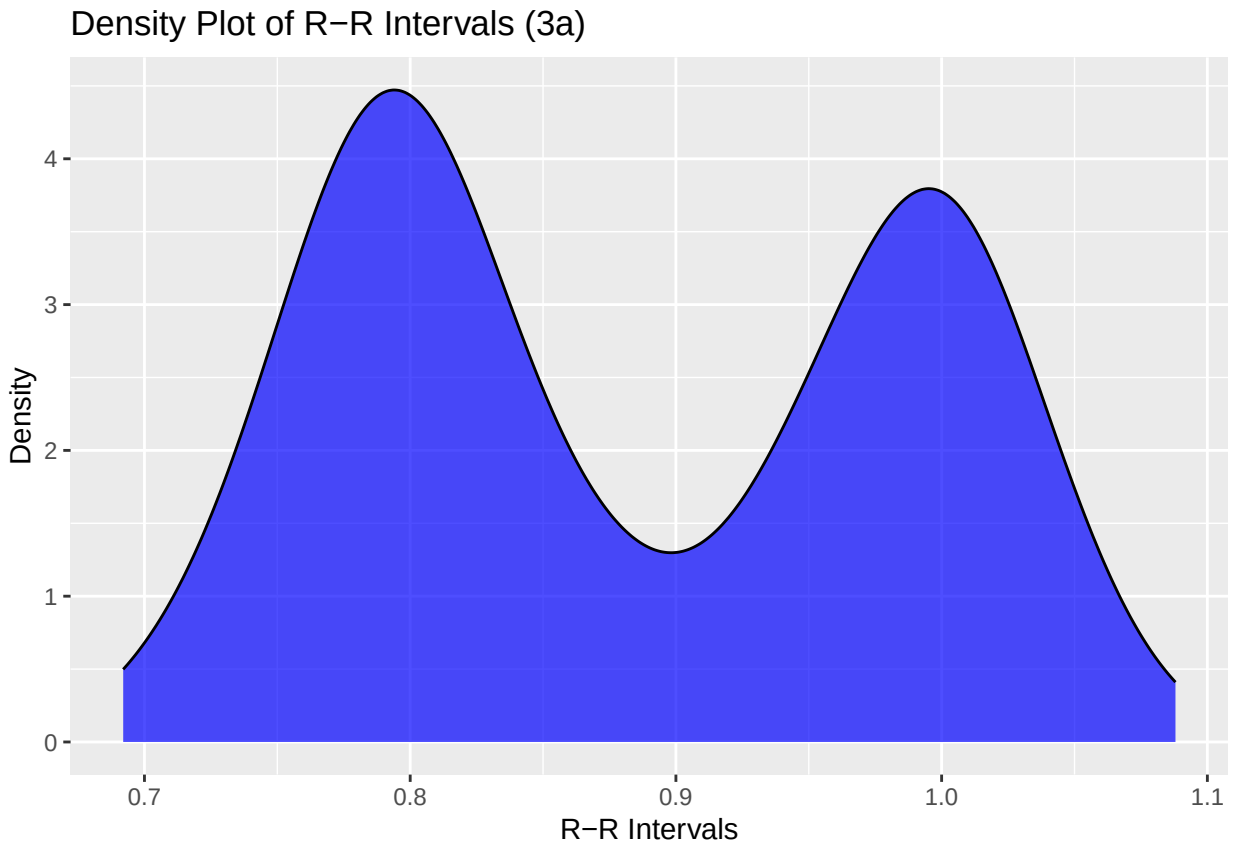

```
plot(data3a$V2, type = "l", xlab = "Time", ylab = "R-R Intervals", main = "Heartbeat Data Over Time (3a)
```



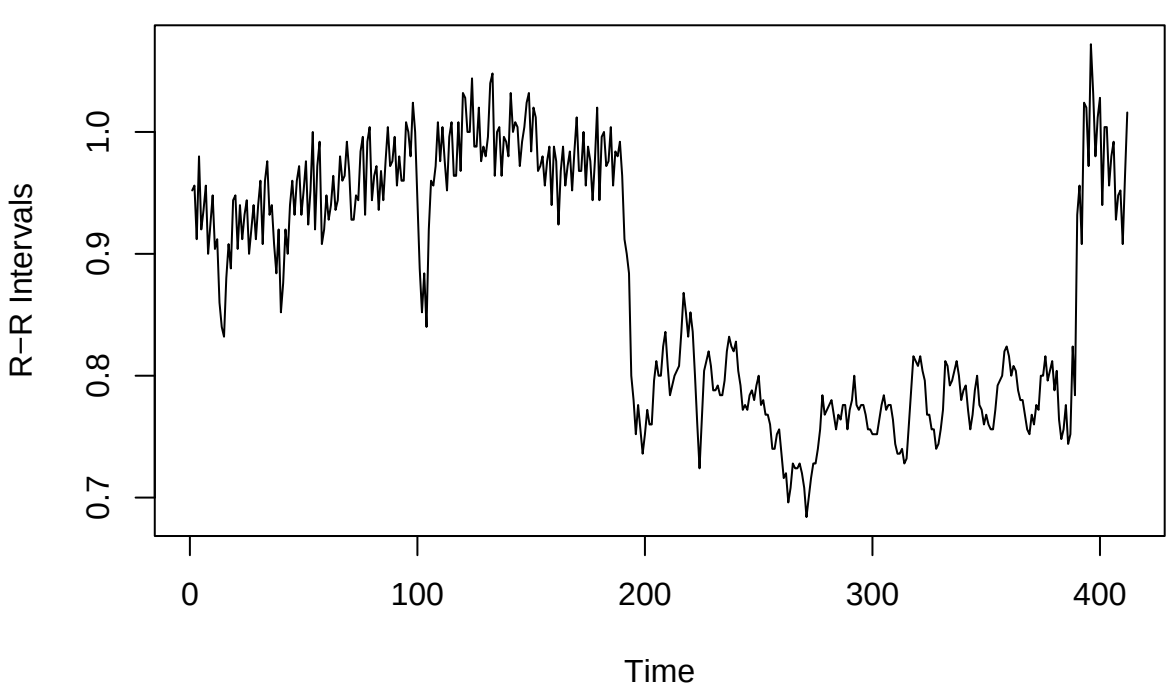
```r
hist(data3a$V2, breaks = 30, xlab = "R-R Intervals", main = "Distribution of R-R Intervals (3a)")
```



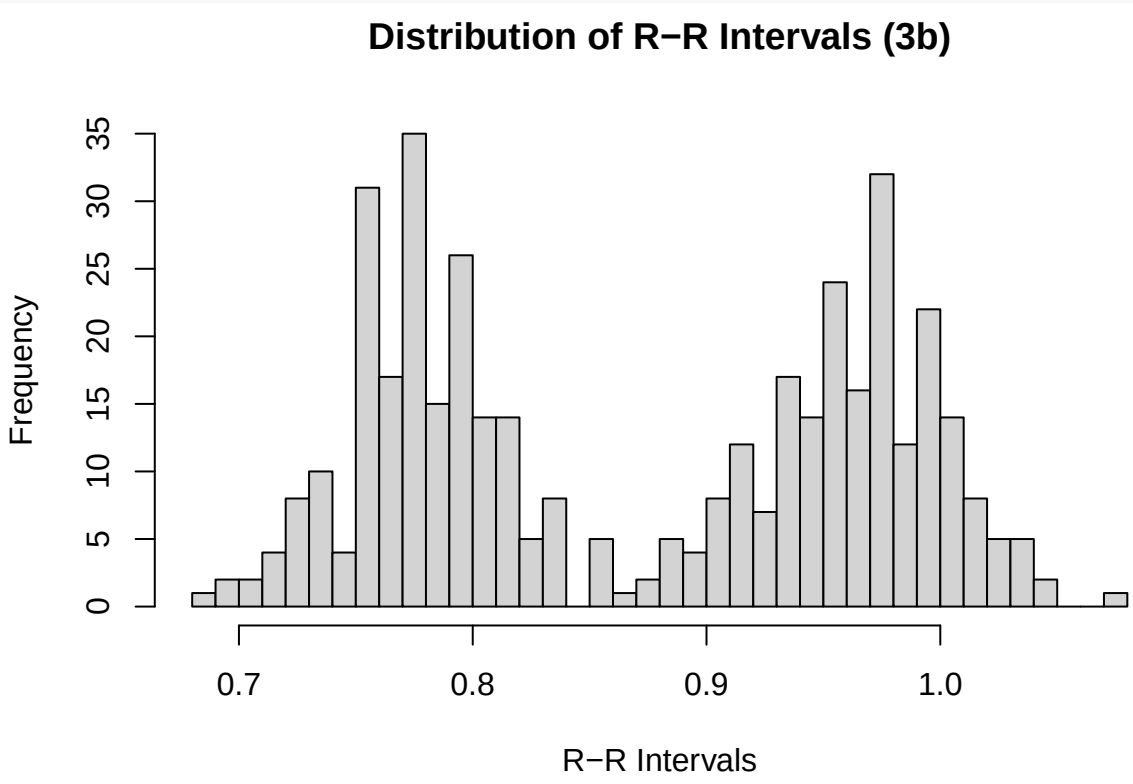

```r
library(ggplot2)
ggplot(data3a, aes(x = V2)) + geom_density(fill = "blue", alpha = 0.7) +
  labs(x = "R-R Intervals", y = "Density", title = "Density Plot of R-R Intervals (3a)")
```



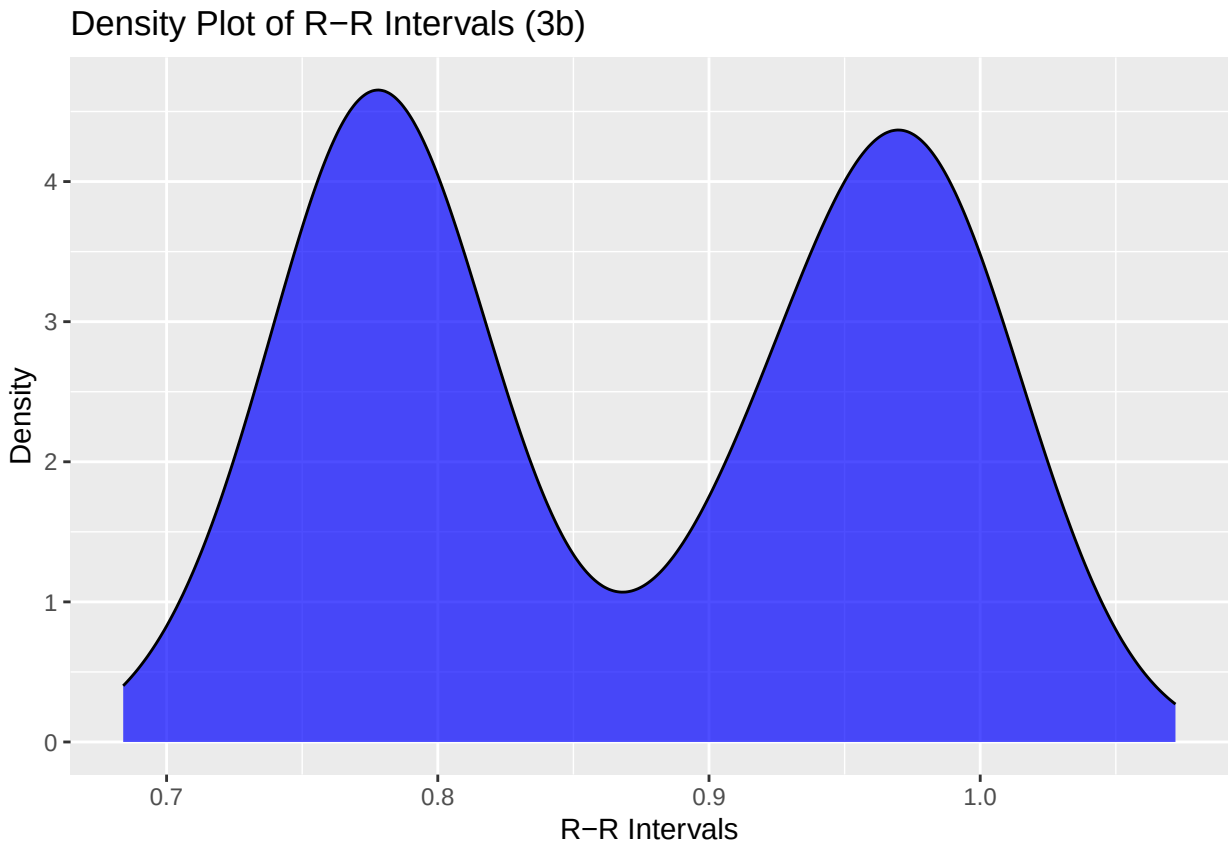

```
plot(data3b$V2, type = "l", xlab = "Time", ylab = "R-R Intervals", main = "Heartbeat Data Over Time (3b)
```



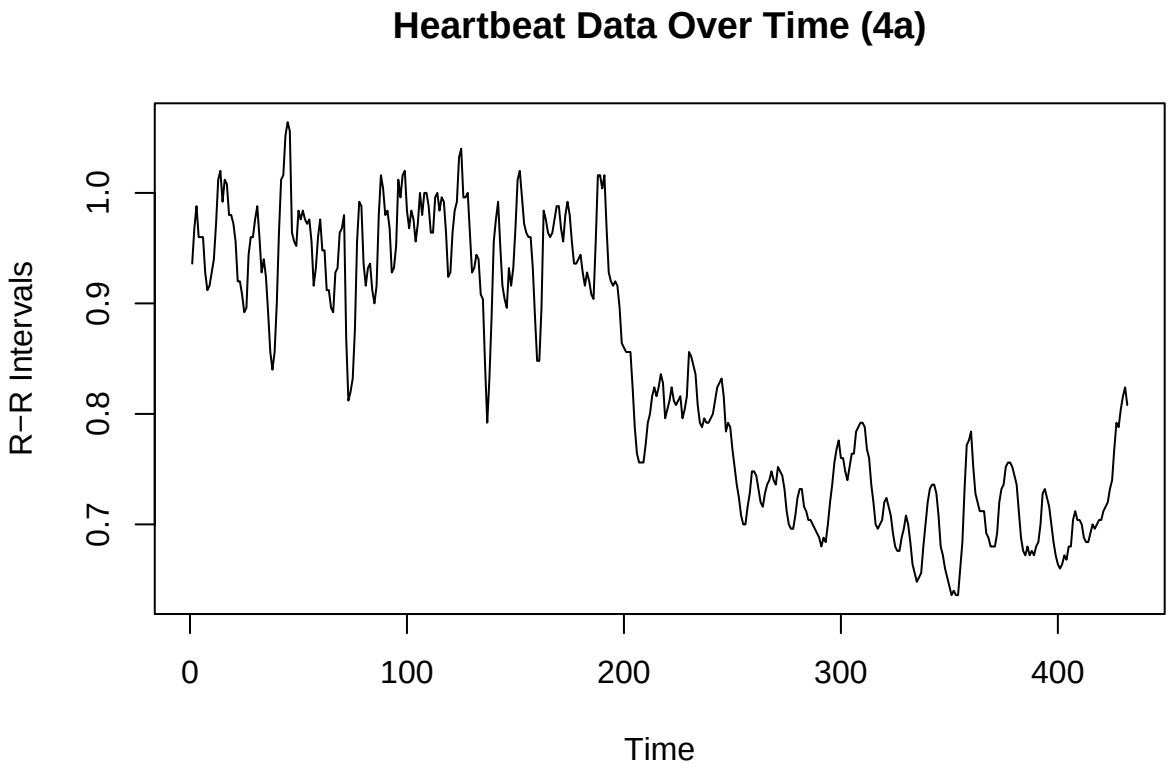
```r
hist(data3b$V2, breaks = 30, xlab = "R-R Intervals", main = "Distribution of R-R Intervals (3b)")
```



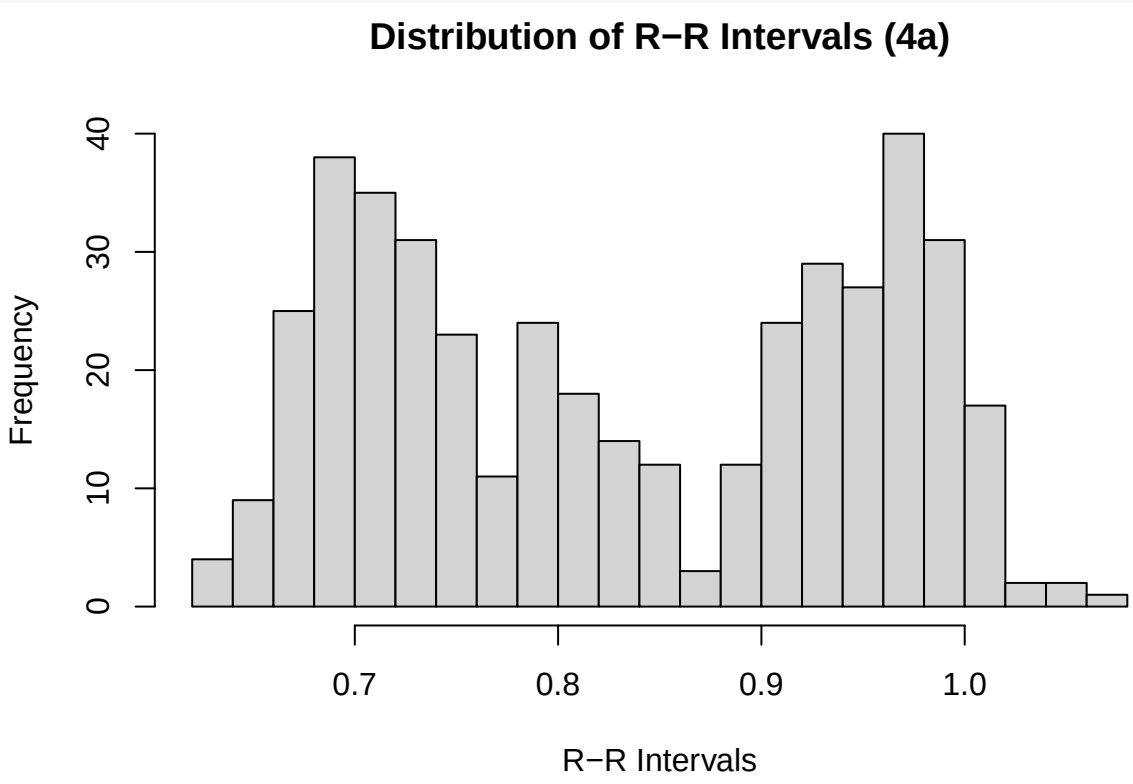

```r
library(ggplot2)
ggplot(data3b, aes(x = V2)) + geom_density(fill = "blue", alpha = 0.7) +
  labs(x = "R-R Intervals", y = "Density", title = "Density Plot of R-R Intervals (3b)")
```



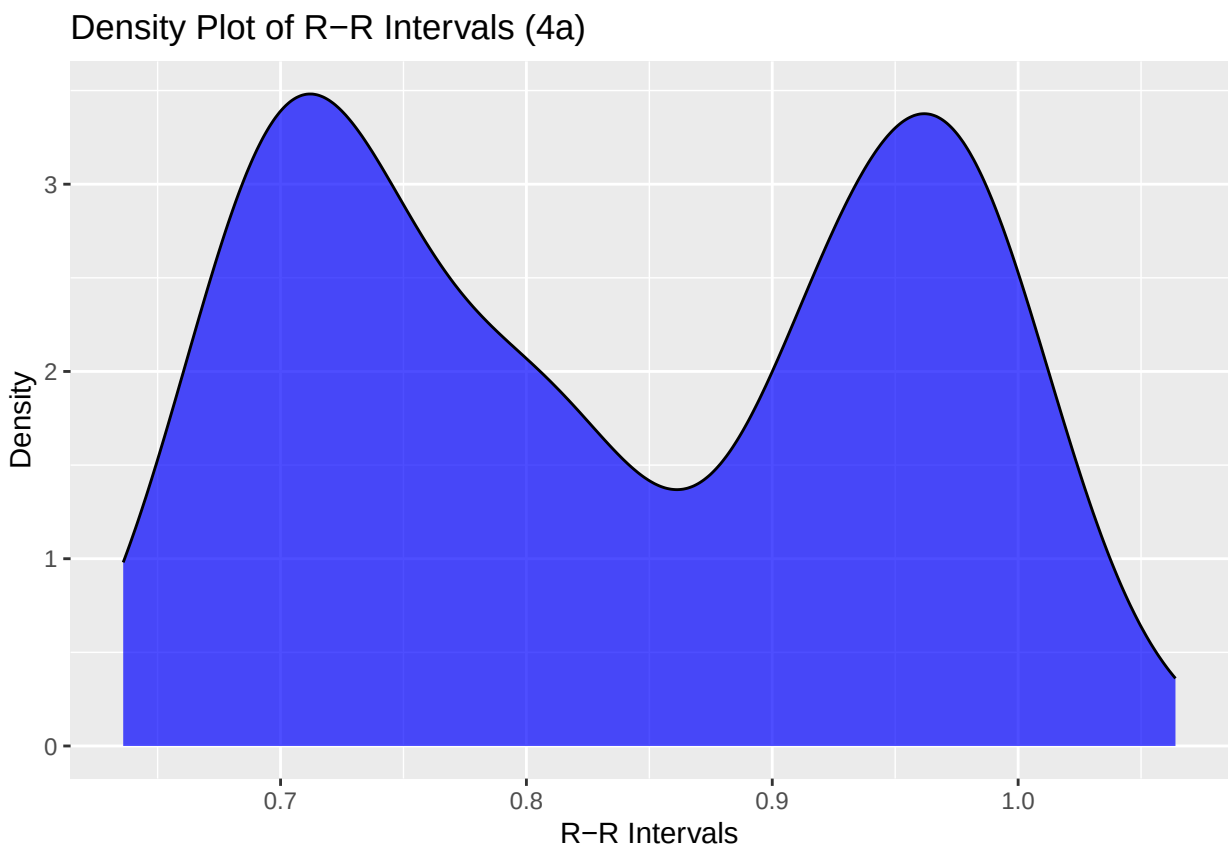

```
plot(data4a$V2, type = "l", xlab = "Time", ylab = "R-R Intervals", main = "Heartbeat Data Over Time (4a)
```



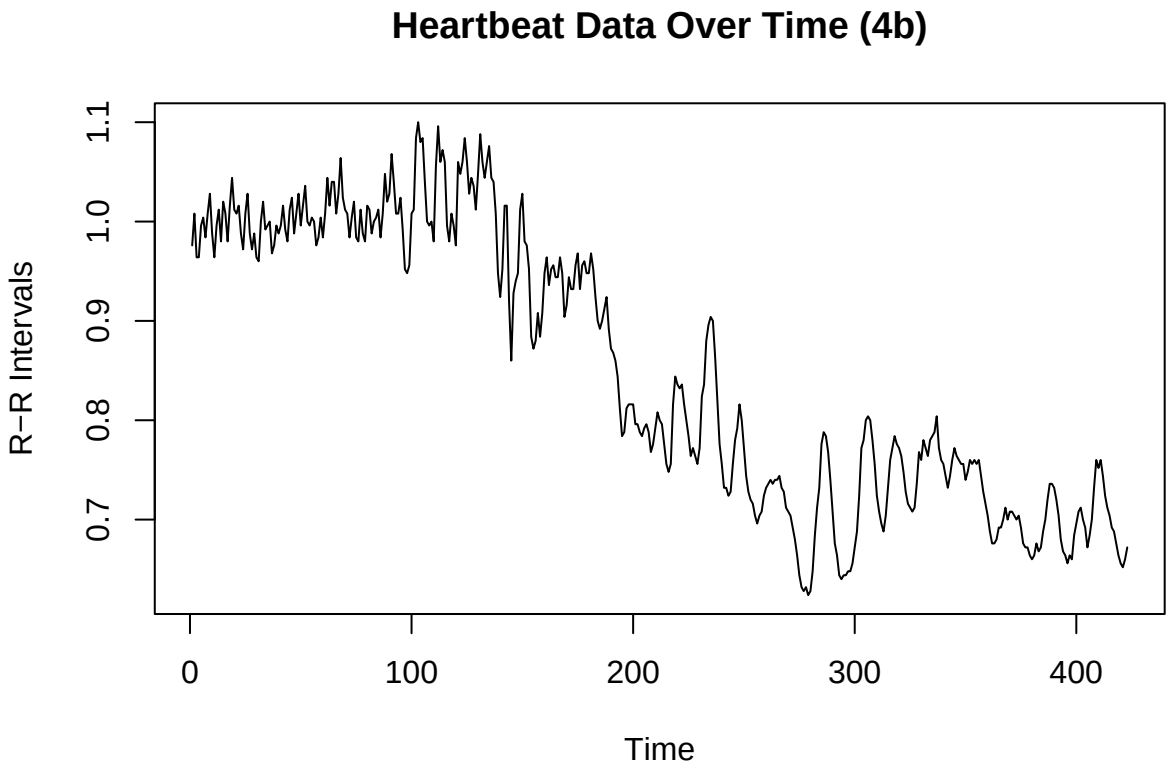
```r
hist(data4a$V2, breaks = 30, xlab = "R-R Intervals", main = "Distribution of R-R Intervals (4a)")
```



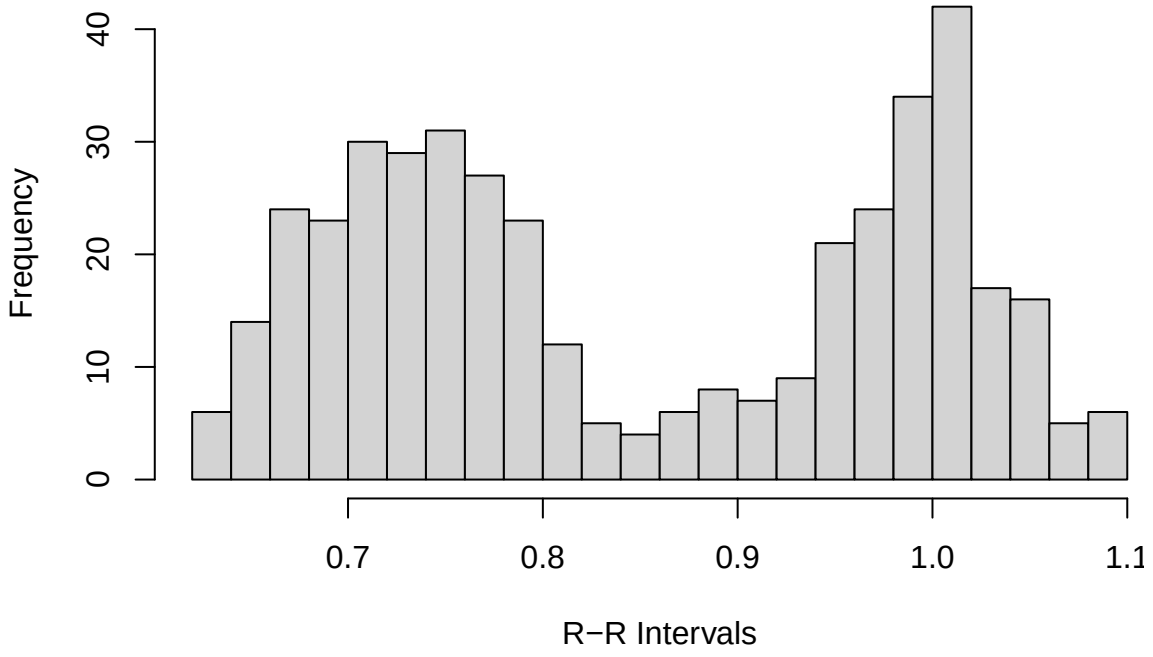

```r
library(ggplot2)
ggplot(data4a, aes(x = V2)) + geom_density(fill = "blue", alpha = 0.7) +
  labs(x = "R-R Intervals", y = "Density", title = "Density Plot of R-R Intervals (4a)")
```



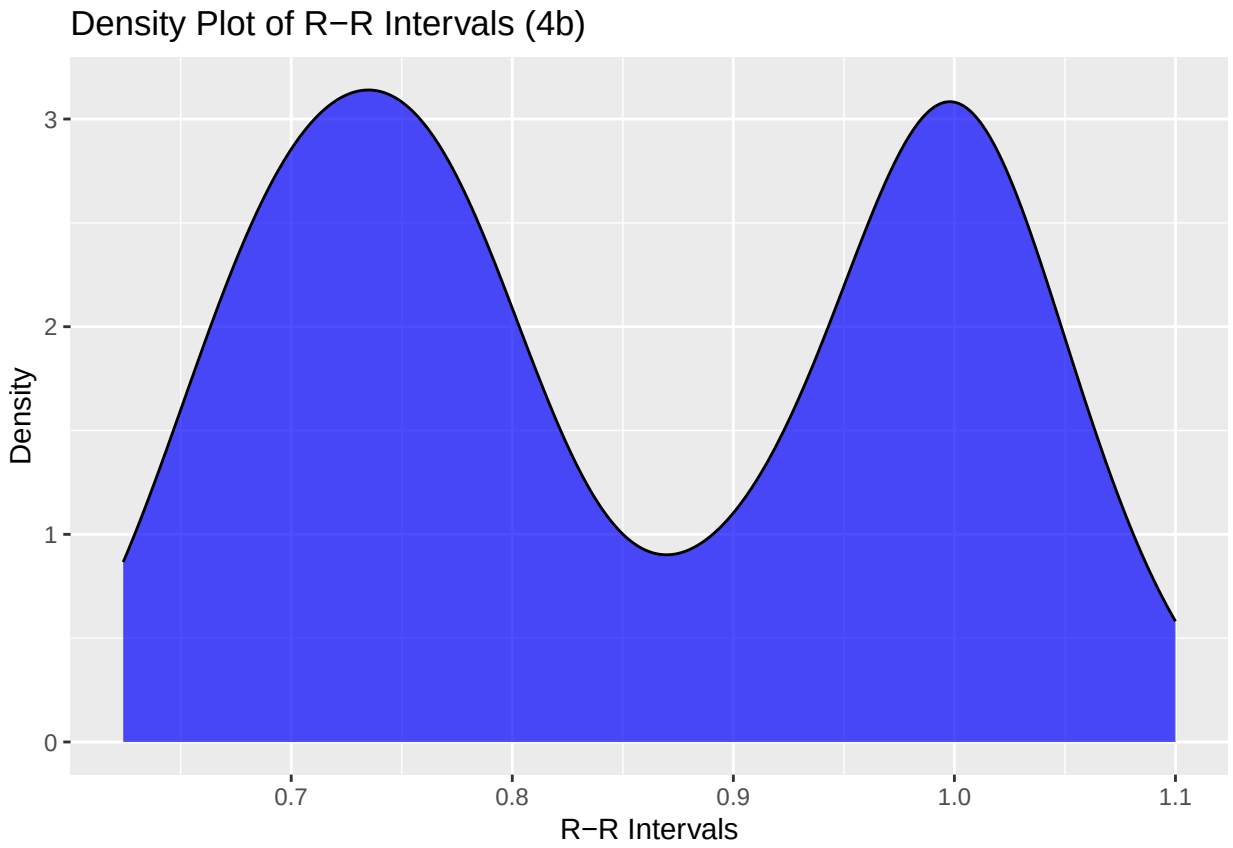

```
plot(data4b$V2, type = "l", xlab = "Time", ylab = "R-R Intervals", main = "Heartbeat Data Over Time (4b)
```



```r
hist(data4b$V2, breaks = 30, xlab = "R-R Intervals", main = "Distribution of R-R Intervals (4b)")
```



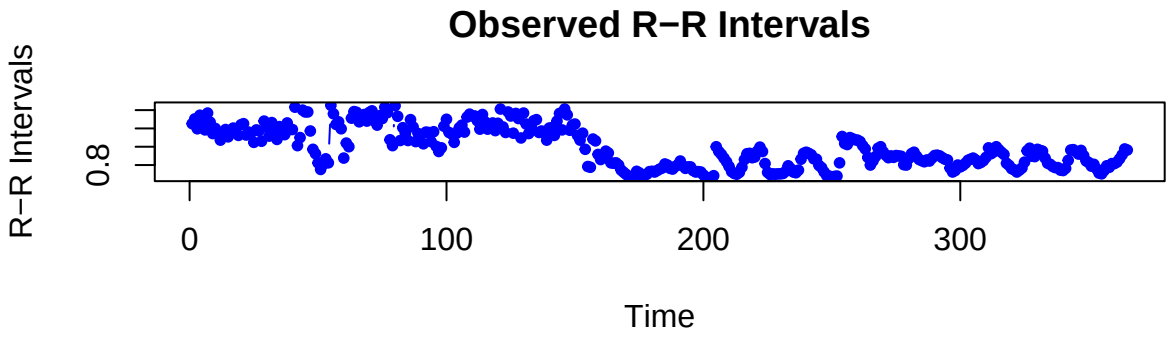

```r
library(ggplot2)
ggplot(data4b, aes(x = V2)) + geom_density(fill = "blue", alpha = 0.7) +
  labs(x = "R-R Intervals", y = "Density", title = "Density Plot of R-R Intervals (4b)")
```

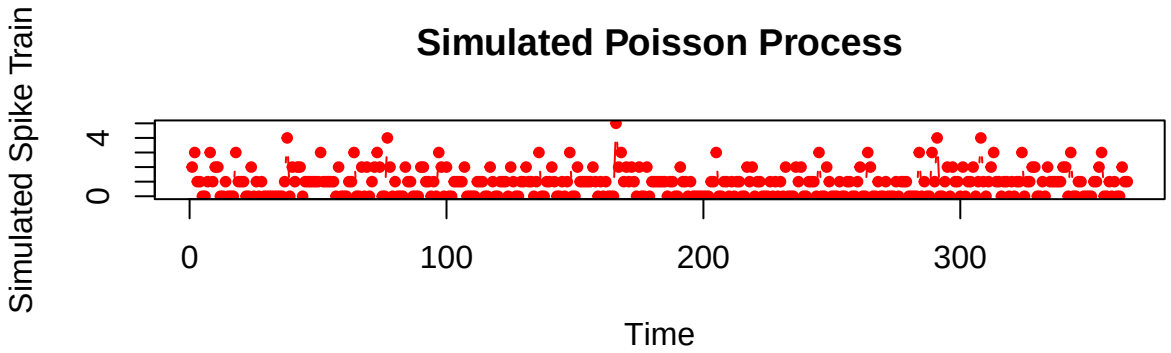



```
library(stats)
rr1a <- data1a$V2

lambda_est <- mean(rr1a)

cat("Estimated rate parameter (lambda):", lambda_est, "\n")
```

```
## Estimated rate parameter (lambda): 0.9876822
```

```
simulated_spike_train <- rpois(length(rr1a), lambda = lambda_est)

par(mfrow = c(2, 1))
plot(rr1a, type = 'b', col = 'blue', pch = 20,
     xlab = 'Time', ylab = 'R-R Intervals',
     main = 'Observed R-R Intervals')

plot(simulated_spike_train, type = 'b', col = 'red', pch = 20,
     xlab = 'Time', ylab = 'Simulated Spike Train',
     main = 'Simulated Poisson Process')
```






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


# Perform KS test
ks_result <- ks.test(rr1a, "pexp", rate = 1/lambda_est)

# Display the results of the KS test
cat("KS Test Results:\n")
```

```
## KS Test Results:
```

```r
print(ks_result)
```

```
##
##  Asymptotic one-sample Kolmogorov-Smirnov test
##
## data:  rr1a
## D = 0.48672, p-value < 2.2e-16
## alternative hypothesis: two-sided
```

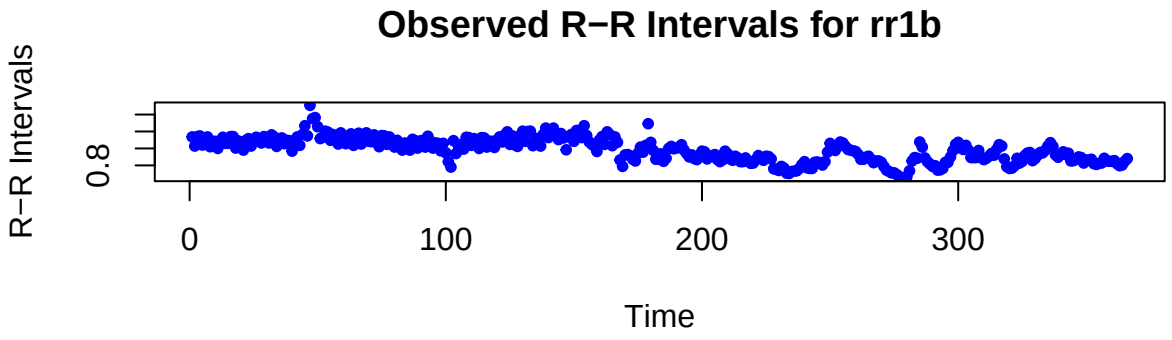
DISTRIBUTIONS NOT IDENTICAL. POISSON PROCESS IS NOT A GOOD FIT.

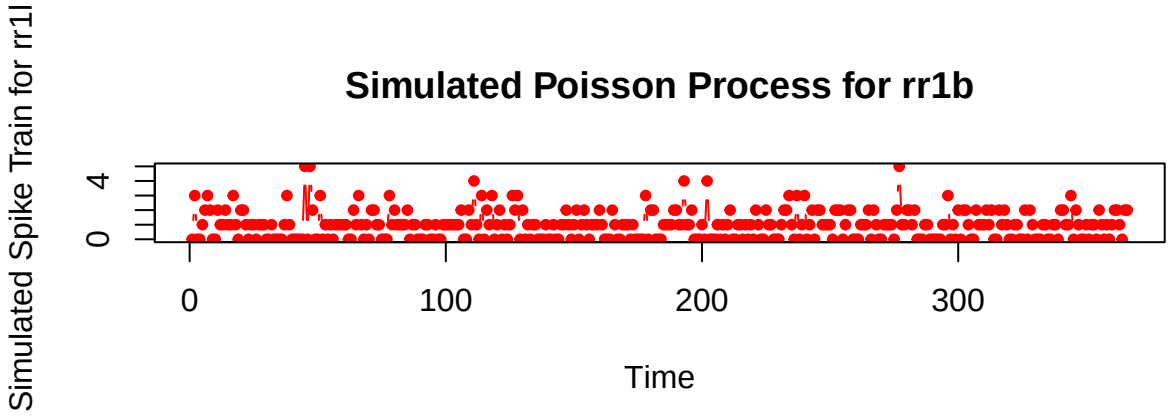
```r
# Load the required library
library(stats)

# Extract the R-R intervals from the second column and rename the variable
rr1b <- data1b$V2

# Calculate the mean R-R interval as an estimate for the rate parameter (lambda)
lambda_est_b <- mean(rr1b)

# Display the estimated rate parameter
```

```r
cat("Estimated rate parameter (lambda) for rr1b:", lambda_est_b, "\n")
```

```
## Estimated rate parameter (lambda) for rr1b: 0.9828197
```

```r
# Simulate a Poisson process based on the estimated lambda
simulated_spike_train_b <- rpois(length(rr1b), lambda = lambda_est_b)

# Plot the observed R-R intervals and the simulated spike train for rr1b
par(mfrow = c(2, 1))
plot(rr1b, type = 'b', col = 'blue', pch = 20,
     xlab = 'Time', ylab = 'R-R Intervals',
     main = 'Observed R-R Intervals for rr1b')

plot(simulated_spike_train_b, type = 'b', col = 'red', pch = 20,
     xlab = 'Time', ylab = 'Simulated Spike Train for rr1b',
     main = 'Simulated Poisson Process for rr1b')
```






```r
# Reset the plotting layout
par(mfrow = c(1, 1))

# Perform KS test for rr1b
ks_result_b <- ks.test(rr1b, "pexp", rate = 1/lambda_est_b)

# Display the results of the KS test for rr1b
cat("KS Test Results for rr1b:\n")
```

```
## KS Test Results for rr1b:
```

```r
print(ks_result_b)
```

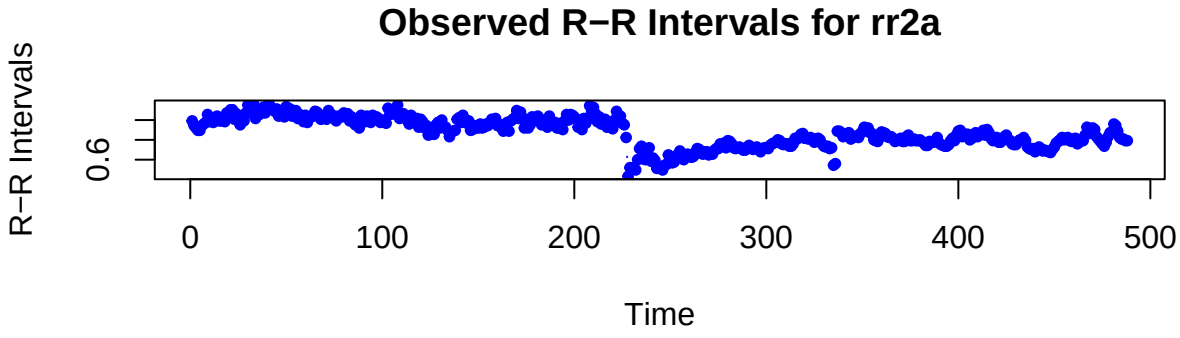
```
##
```

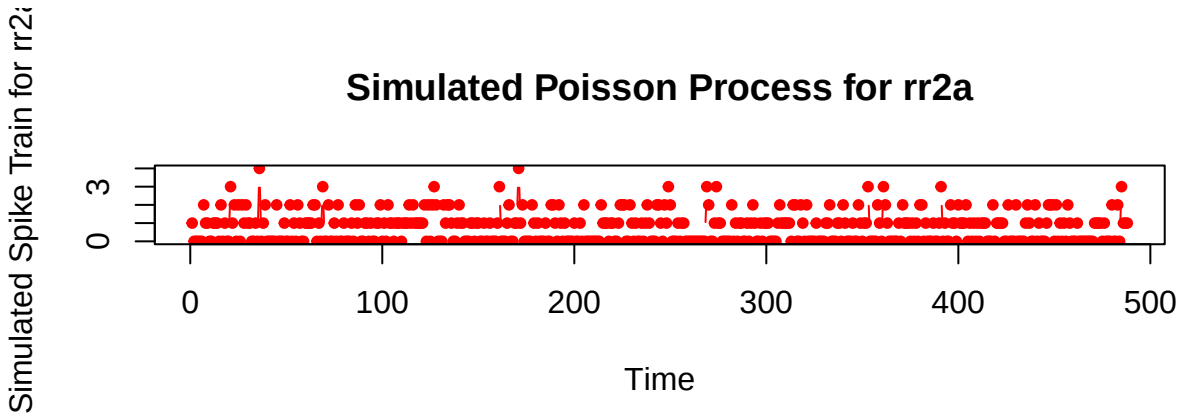
```
##  Asymptotic one-sample Kolmogorov-Smirnov test
##
## data:  rr1b
## D = 0.49864, p-value < 2.2e-16
## alternative hypothesis: two-sided
```

```r
# Extract the R-R intervals from the second column and rename the variable
rr2a <- data2a$V2

# Calculate the mean R-R interval as an estimate for the rate parameter (lambda)
lambda_est_2a <- mean(rr2a)

# Display the estimated rate parameter
cat("Estimated rate parameter (lambda) for rr2a:", lambda_est_2a, "\n")
```

```
## Estimated rate parameter (lambda) for rr2a: 0.7369344
```

```r
# Simulate a Poisson process based on the estimated lambda
simulated_spike_train_2a <- rpois(length(rr2a), lambda = lambda_est_2a)

# Plot the observed R-R intervals and the simulated spike train for rr2a
par(mfrow = c(2, 1))
plot(rr2a, type = 'b', col = 'blue', pch = 20,
     xlab = 'Time', ylab = 'R-R Intervals',
     main = 'Observed R-R Intervals for rr2a')

plot(simulated_spike_train_2a, type = 'b', col = 'red', pch = 20,
     xlab = 'Time', ylab = 'Simulated Spike Train for rr2a',
     main = 'Simulated Poisson Process for rr2a')
```





```r
# Reset the plotting layout
par(mfrow = c(1, 1))

# Perform KS test for rr2a
ks_result_2a <- ks.test(rr2a, "pexp", rate = 1/lambda_est_2a)

# Display the results of the KS test for rr2a
cat("KS Test Results for rr2a:\n")
```

```
## KS Test Results for rr2a:
```

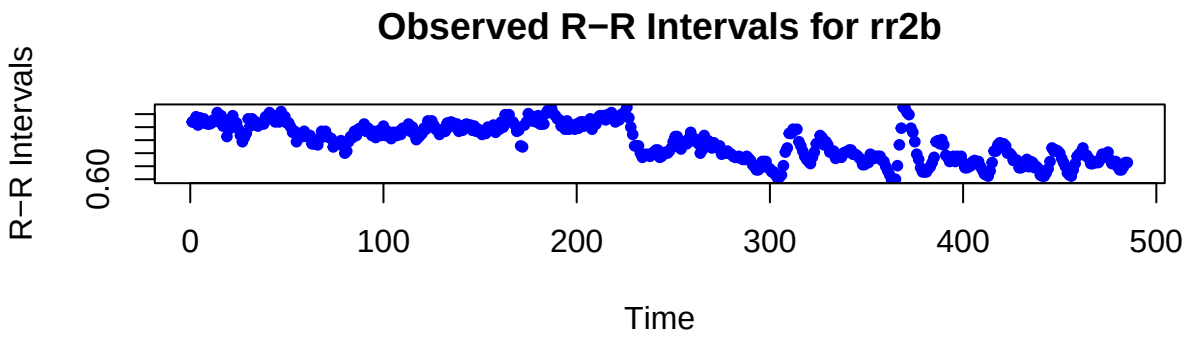
```r
print(ks_result_2a)
```

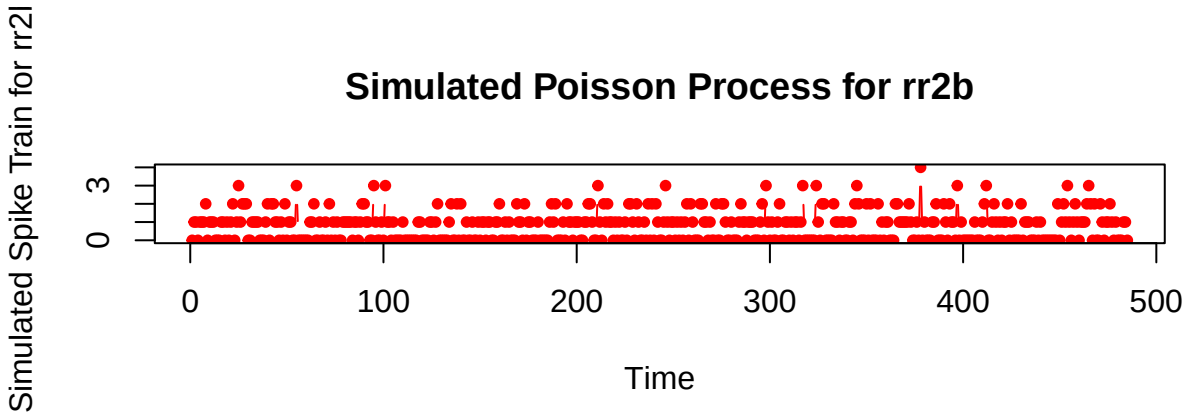
```
##
##  Asymptotic one-sample Kolmogorov-Smirnov test
##
## data:  rr2a
## D = 0.5236, p-value < 2.2e-16
## alternative hypothesis: two-sided
```

```r
# Load data for subject 2b
data2b <- read.table('data/subj2b.txt', header = FALSE)

# Extract the R-R intervals from the second column and rename the variable
rr2b <- data2b$V2

# Calculate the mean R-R interval as an estimate for the rate parameter (lambda)
lambda_est_2b <- mean(rr2b)

# Display the estimated rate parameter
cat("Estimated rate parameter (lambda) for rr2b:", lambda_est_2b, "\n")
```

```
## Estimated rate parameter (lambda) for rr2b: 0.7423423
```

```r
# Simulate a Poisson process based on the estimated lambda
simulated_spike_train_2b <- rpois(length(rr2b), lambda = lambda_est_2b)

# Plot the observed R-R intervals and the simulated spike train for rr2b
par(mfrow = c(2, 1))
plot(rr2b, type = 'b', col = 'blue', pch = 20,
     xlab = 'Time', ylab = 'R-R Intervals',
     main = 'Observed R-R Intervals for rr2b')

plot(simulated_spike_train_2b, type = 'b', col = 'red', pch = 20,
     xlab = 'Time', ylab = 'Simulated Spike Train for rr2b',
     main = 'Simulated Poisson Process for rr2b')
```






```
# Reset the plotting layout
par(mfrow = c(1, 1))

# Perform KS test for rr2b
ks_result_2b <- ks.test(rr2b, "pexp", rate = 1/lambda_est_2b)

# Display the results of the KS test for rr2b
cat("KS Test Results for rr2b:\n")
```

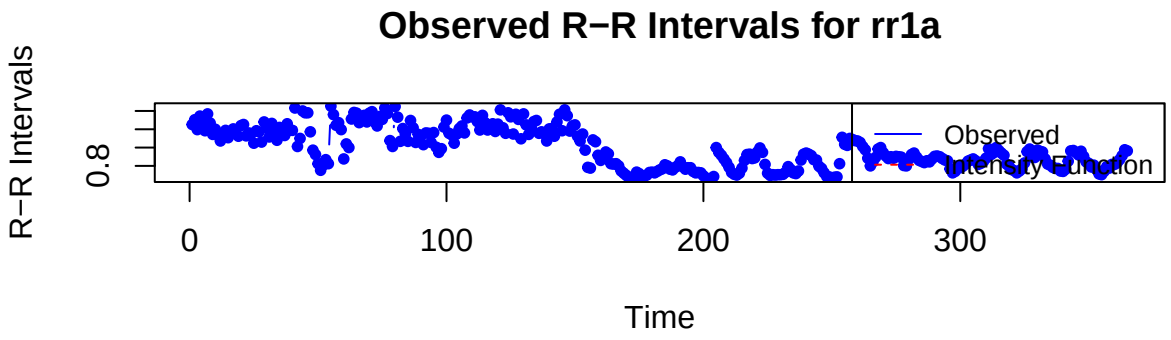
```
## KS Test Results for rr2b:
```

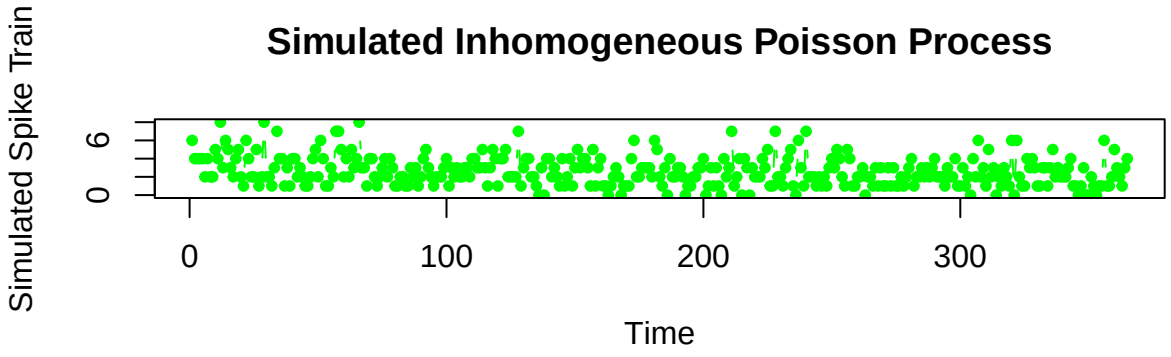
```
print(ks_result_2b)
```

```
##
##  Asymptotic one-sample Kolmogorov-Smirnov test
##
## data:  rr2b
## D = 0.55196, p-value < 2.2e-16
## alternative hypothesis: two-sided
```

```
# Load the required library
library(stats)

# Load data for subject 1a
data1a <- read.table('data/subj1a.txt', header = FALSE)

# Extract the R-R intervals from the second column and rename the variable
rr1a <- data1a$V2

# Create a time variable (assuming time increments of 1, adjust as needed)
time <- 1:length(rr1a)
```

```r
# Fit a linear model to estimate the intensity function
model <- lm(rr1a ~ time)

# Extract the coefficients
beta0 <- coef(model)[1]
beta1 <- coef(model)[2]

# Define the estimated intensity function
lambda_est <- exp(beta0 + beta1 * time)

# Simulate an inhomogeneous Poisson process based on the estimated intensity
simulated_spike_train <- rpois(length(rr1a), lambda = lambda_est)

# Plot the observed R-R intervals and the estimated intensity function
par(mfrow = c(2, 1))
plot(rr1a, type = 'b', col = 'blue', pch = 20,
     xlab = 'Time', ylab = 'R-R Intervals',
     main = 'Observed R-R Intervals for rr1a')

lines(time, lambda_est, col = 'red', lty = 2)
legend("topright", legend = c("Observed", "Intensity Function"),
       col = c("blue", "red"), lty = c(1, 2), cex = 0.8)

# Plot the simulated inhomogeneous Poisson process
plot(simulated_spike_train, type = 'b', col = 'green', pch = 20,
     xlab = 'Time', ylab = 'Simulated Spike Train',
     main = 'Simulated Inhomogeneous Poisson Process')
```





```r
# Reset the plotting layout
par(mfrow = c(1, 1))

# Perform KS test for inhomogeneous Poisson process
ks_result_inh <- ks.test(rr1a, lambda_est)
```

```
## Warning in ks.test.default(rr1a, lambda_est): p-value will be approximate in
## the presence of ties
```

```r
# Display the results of the KS test for inhomogeneous Poisson process
cat("KS Test Results for Inhomogeneous Poisson Process:\n")
```

```
## KS Test Results for Inhomogeneous Poisson Process:
```

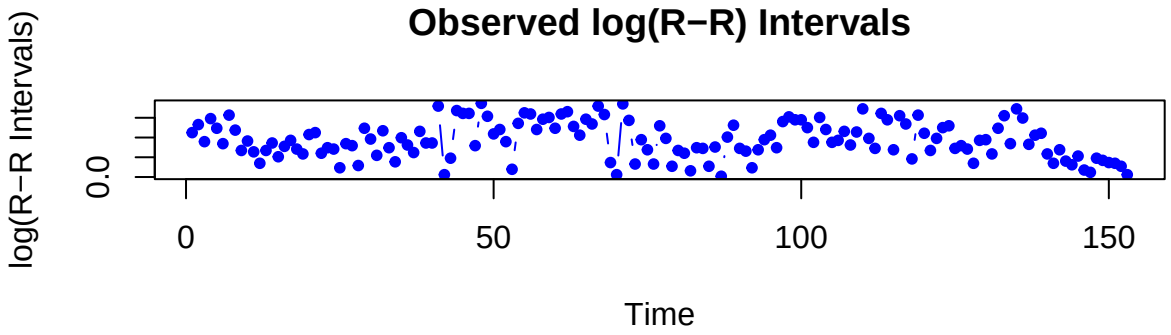
```r
print(ks_result_inh)
```

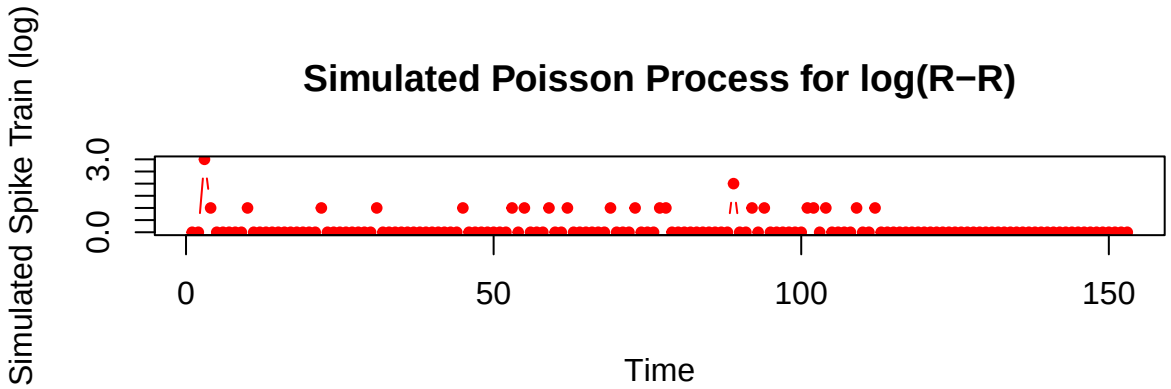
```
##
##  Asymptotic two-sample Kolmogorov-Smirnov test
##
## data:  rr1a and lambda_est
## D = 1, p-value < 2.2e-16
## alternative hypothesis: two-sided
```

```r
library(stats)

# Transform R-R intervals to log(R-R)
log_rr1a <- log(rr1a)

# Remove non-positive values
log_rr1a <- log_rr1a[log_rr1a > 0]

# Check if there are still non-positive values
if (any(log_rr1a <= 0)) {
  stop("After log transformation, there are still non-positive values in log(R-R) intervals.")
}

# Estimate rate parameter for the transformed data
lambda_est_log <- mean(log_rr1a)

cat("Estimated rate parameter (lambda) for log(R-R):", lambda_est_log, "\n")
```

```
## Estimated rate parameter (lambda) for log(R-R): 0.1874978
```

```r
# Simulate Poisson process for log(R-R)
simulated_spike_train_log <- rpois(length(log_rr1a), lambda = lambda_est_log)

# Plot the observed log(R-R) intervals and the simulated spike train
par(mfrow = c(2, 1))
plot(log_rr1a, type = 'b', col = 'blue', pch = 20,
     xlab = 'Time', ylab = 'log(R-R Intervals)',
     main = 'Observed log(R-R) Intervals')

plot(simulated_spike_train_log, type = 'b', col = 'red', pch = 20,
     xlab = 'Time', ylab = 'Simulated Spike Train (log)',
     main = 'Simulated Poisson Process for log(R-R)')
```






```r
# Reset the plotting layout
par(mfrow = c(1, 1))

# Perform KS test for log(R-R)
ks_result_log <- ks.test(log_rr1a, "pexp", rate = 1/lambda_est_log)
```

```
## Warning in ks.test.default(log_rr1a, "pexp", rate = 1/lambda_est_log): ties
## should not be present for the Kolmogorov-Smirnov test
```

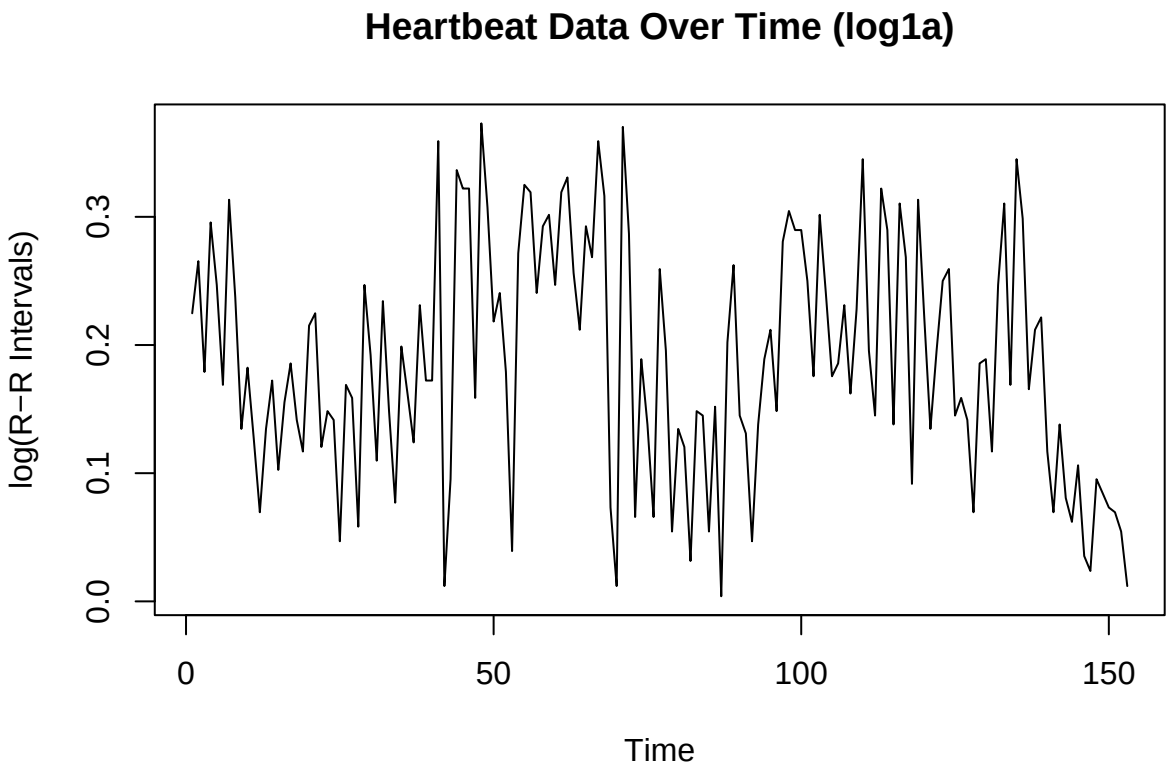
```r
# Display the results of the KS test for log(R-R)
cat("KS Test Results for log(R-R):\n")
```

```
## KS Test Results for log(R-R):
```

```r
print(ks_result_log)
```

```
##
##  Asymptotic one-sample Kolmogorov-Smirnov test
##
## data:  log_rr1a
## D = 0.25475, p-value = 4.746e-09
## alternative hypothesis: two-sided
```

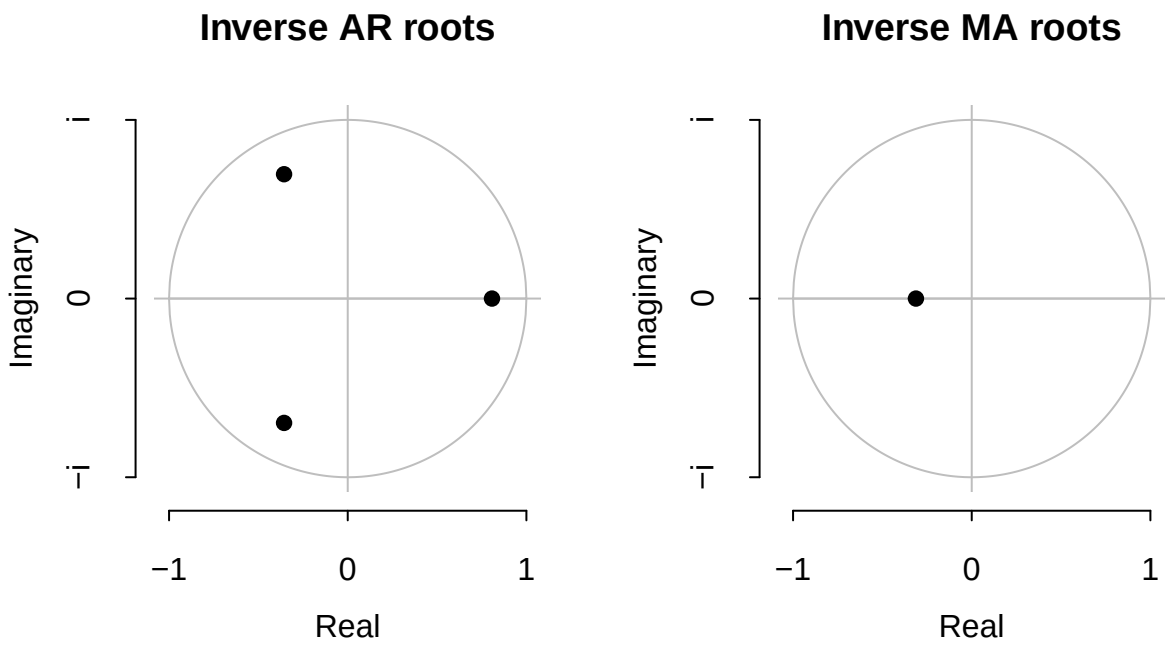
```r
plot(log_rr1a, type = "l", xlab = "Time", ylab = "log(R-R Intervals)", main = "Heartbeat Data Over Time
```



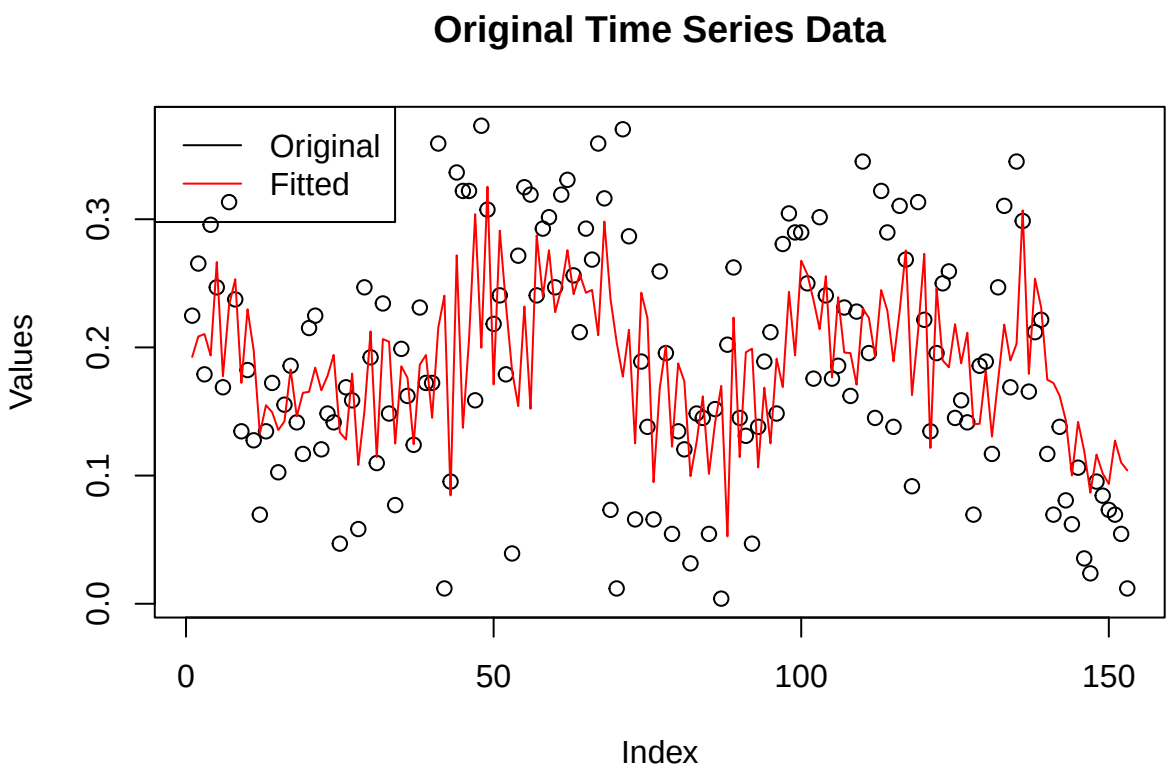

```
library(forecast)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
res <- auto.arima(log_rr1a)

plot(res)
```



```r
# Plot the original time series data
plot(log_rr1a, main = "Original Time Series Data", ylab = "Values")

# Add the fitted values to the plot
lines(fitted(res), col = "red")
legend("topleft", legend = c("Original", "Fitted"), col = c("black", "red"), lty = 1)
```

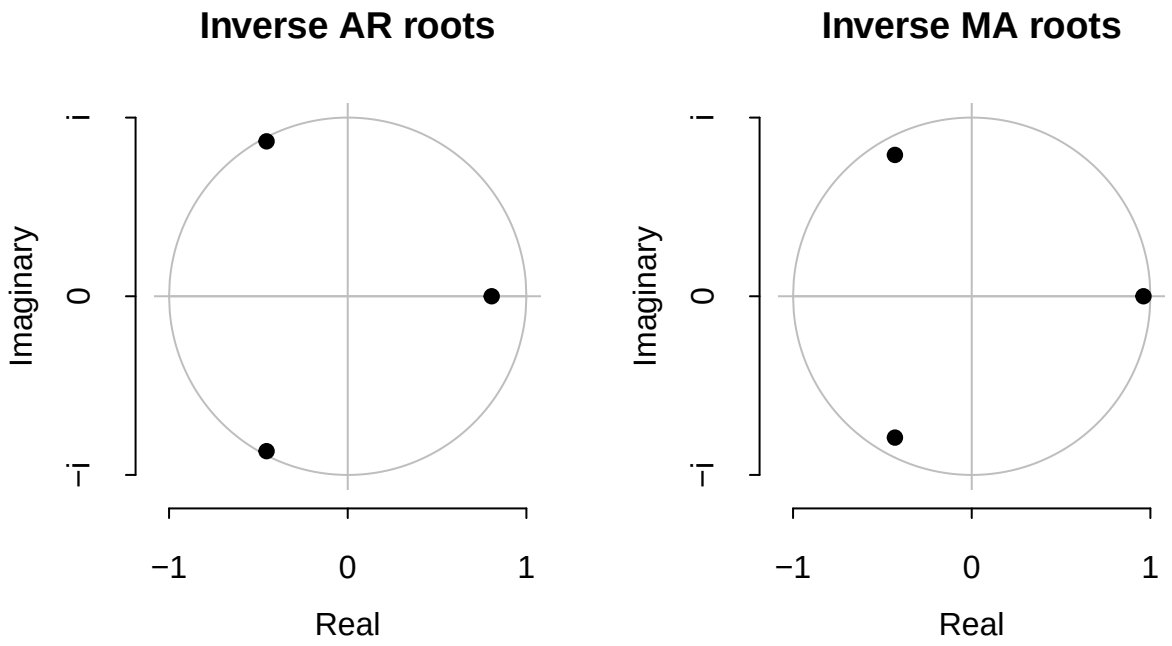



```r
summary(res)
```

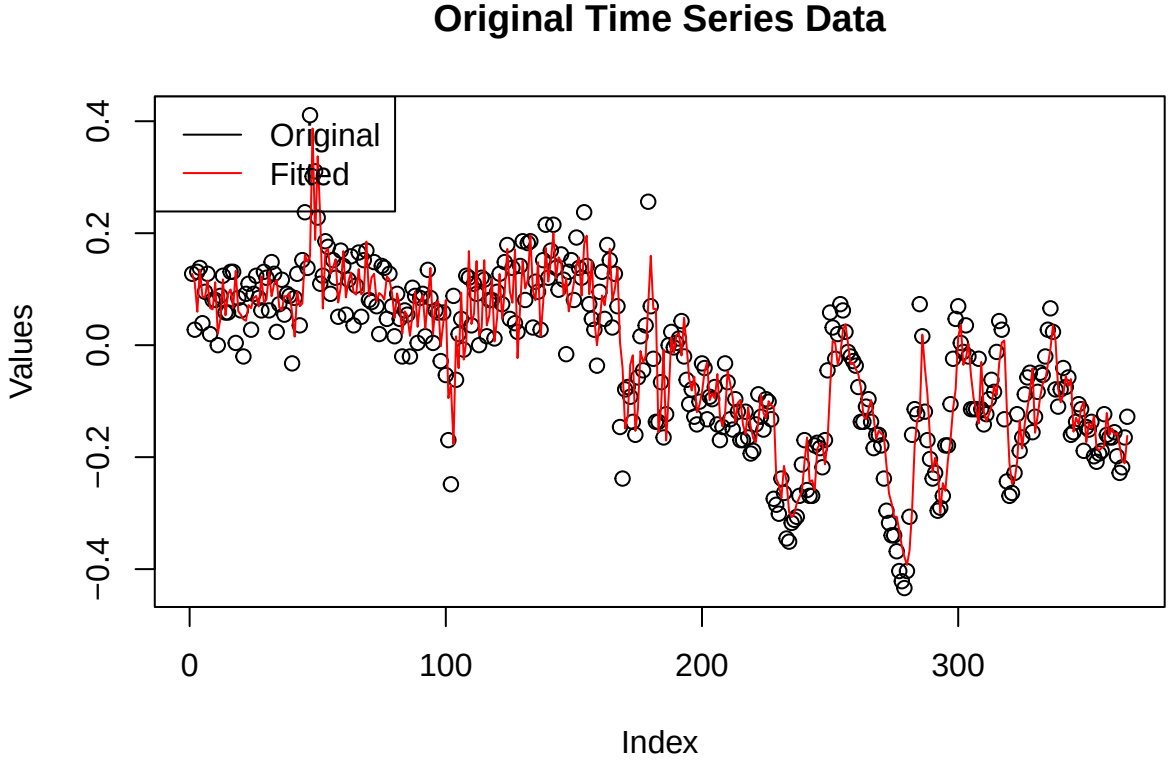
```
## Series: log_rr1a
## ARIMA(3,0,1) with non-zero mean
##
## Coefficients:
##          ar1      ar2     ar3     ma1    mean
##       0.0949  -0.0360  0.4936  0.3123  0.1846
## s.e.  0.1354   0.0846  0.0719  0.1556  0.0171
##
## sigma^2 = 0.005612:  log likelihood = 181.4
## AIC=-350.8   AICc=-350.23   BIC=-332.62
##
## Training set error measures:
##                         ME      RMSE        MAE       MPE     MAPE      MASE
## Training set -0.0003254295 0.0736823 0.05801808 -74.22299 95.18879 0.7177017
##                      ACF1
## Training set -0.007719581
```

```r
# Transform R-R intervals to log(R-R)
log_rr1b <- log(rr1b)

resb <- auto.arima(log_rr1b)
```

```r
plot(resb)
```



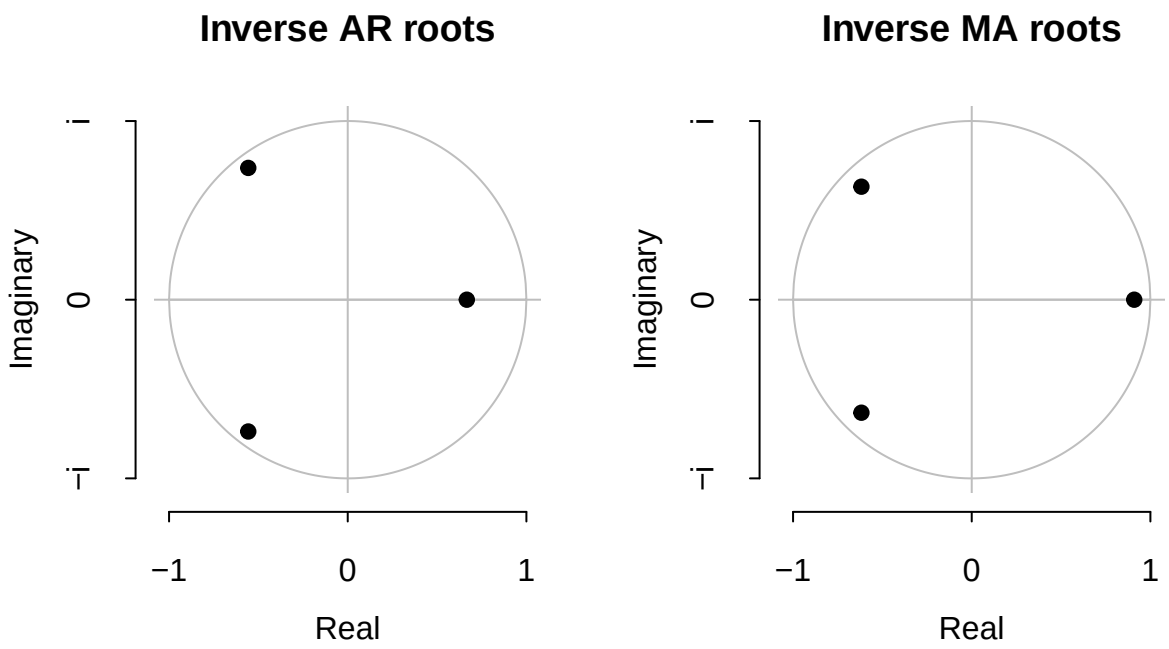

```r
# Plot the original time series data
plot(log_rr1b, main = "Original Time Series Data", ylab = "Values")

# Add the fitted values to the plot
lines(fitted(resb), col = "red")
legend("topleft", legend = c("Original", "Fitted"), col = c("black", "red"), lty = 1)
```



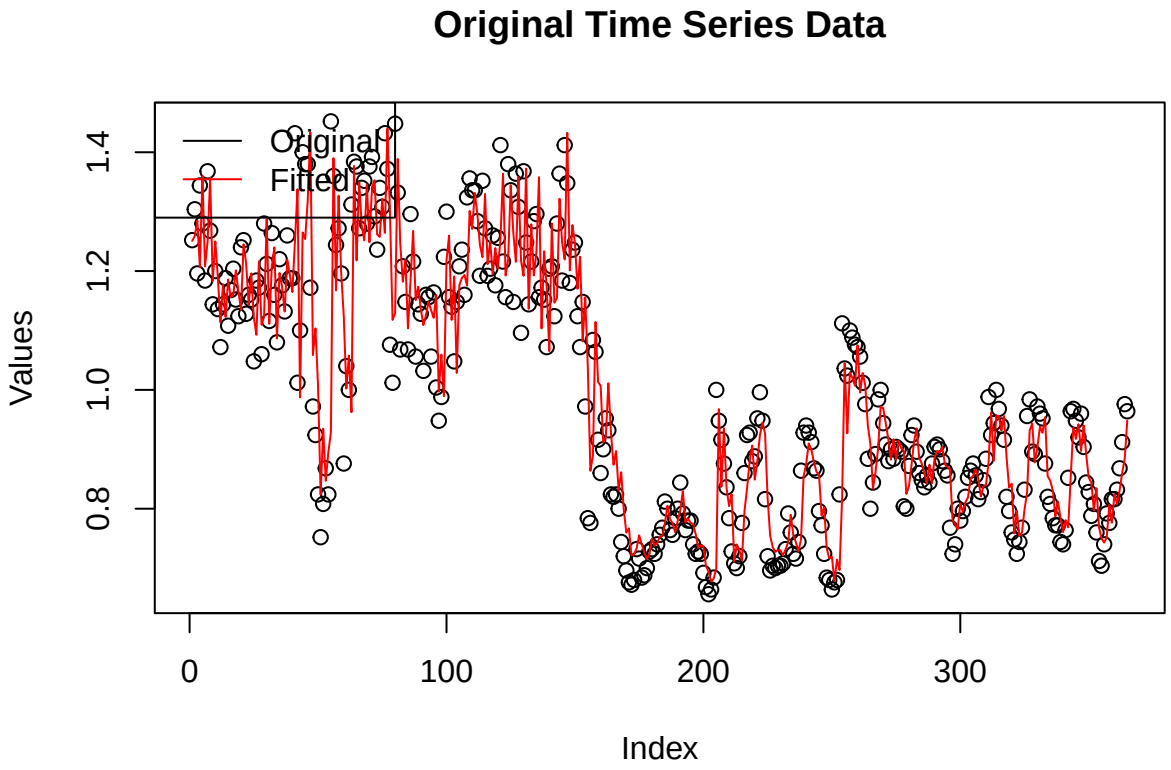
```r
summary(resb)
```

```
## Series: log_rr1b
## ARIMA(3,1,3)
##
## Coefficients:
##           ar1      ar2     ar3      ma1      ma2      ma3
##       -0.1030  -0.2262  0.7720  -0.1015  -0.0158  -0.7791
## s.e.   0.0503   0.0452  0.0447   0.0495   0.0534   0.0484
##
## sigma^2 = 0.003821:  log likelihood = 500.2
## AIC=-986.4   AICc=-986.08   BIC=-959.1
##
## Training set error measures:
##                       ME       RMSE        MAE MPE MAPE     MASE      ACF1
## Training set -0.00347232 0.06122252 0.04618269 NaN  Inf 0.808039 0.0182391
```

```r
library(forecast)

res1a <- auto.arima(rr1a)

plot(res1a)
```



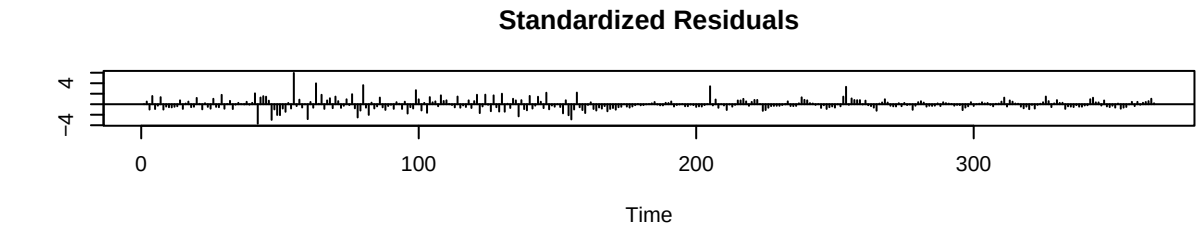

```r
# Plot the original time series data
plot(rr1a, main = "Original Time Series Data", ylab = "Values")

# Add the fitted values to the plot
lines(fitted(res1a), col = "red")
legend("topleft", legend = c("Original", "Fitted"), col = c("black", "red"), lty = 1)
```

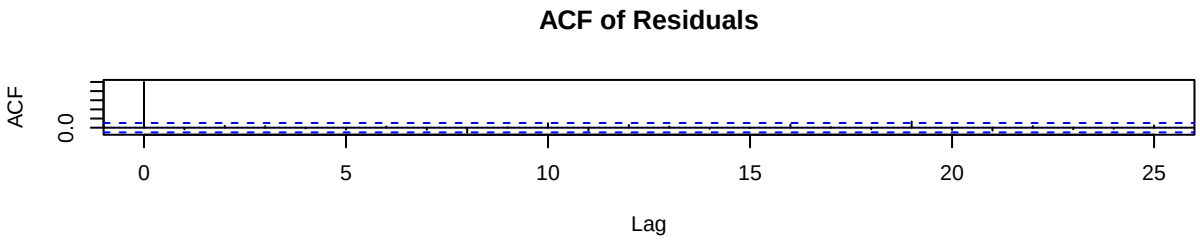


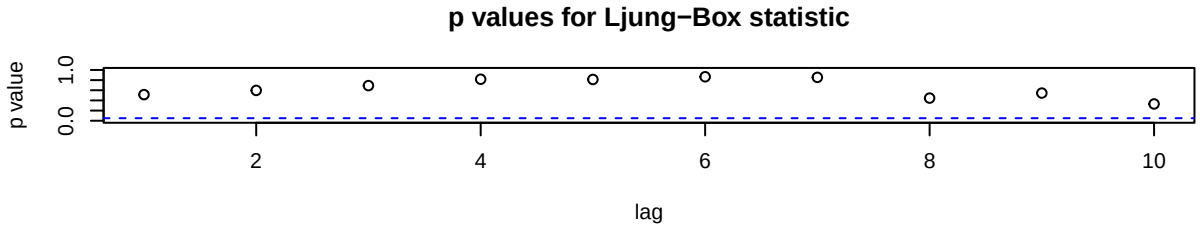

```
summary(res1a)
```

```
## Series: rr1a
## ARIMA(3,1,3)
##
## Coefficients:
##           ar1      ar2     ar3     ma1      ma2      ma3
##       -0.4470  -0.1119  0.5694  0.3254  -0.3419  -0.7106
## s.e.   0.1188   0.0758  0.0692  0.1201   0.0534   0.0820
##
## sigma^2 = 0.007718:  log likelihood = 371.19
## AIC=-728.37   AICc=-728.06   BIC=-701.09
##
## Training set error measures:
##                        ME       RMSE        MAE        MPE     MAPE      MASE
## Training set -0.003635937 0.08700585 0.06114694 -0.9751233 6.029105 0.9197308
##                     ACF1
## Training set -0.03399185
```

```
tsdiag(res1a)
```


Standardized Residuals
4
−4
0
100
200
300
Time


ACF of Residuals
ACF
0.0
0
5
10
15
20
25
Lag


p values for Ljung−Box statistic
p value
1.0
0.0
2
4
6
8
10
lag