Homework 02 - Statistical Methods for Data Science

Victor Plesco, Matteo Zambon

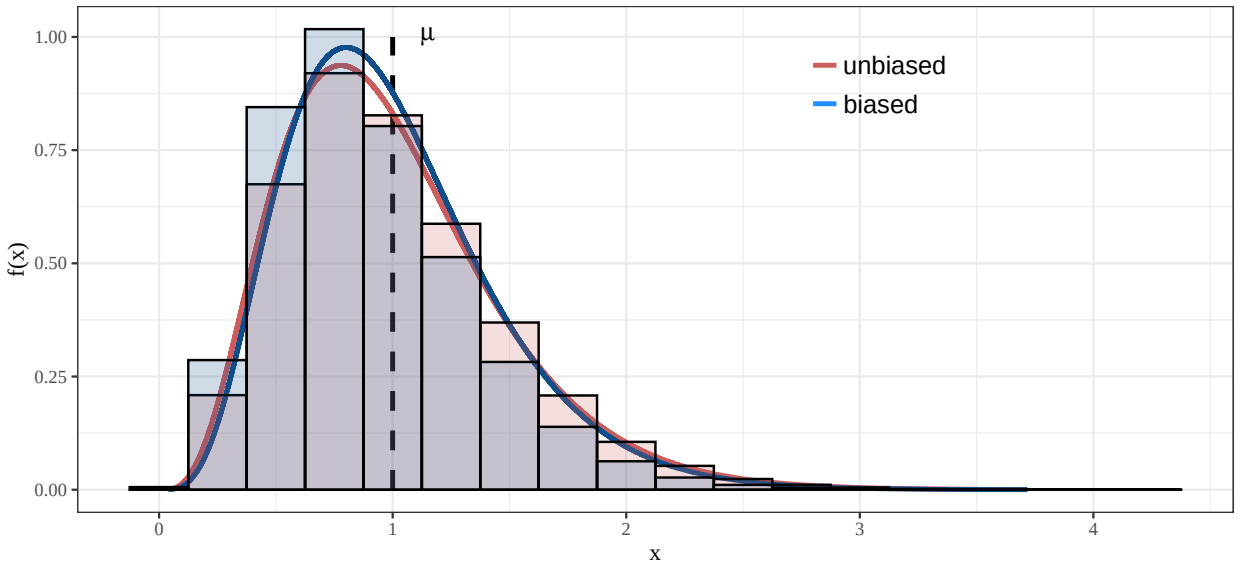
LAB Exercises

Exercise 1

Check the biased nature of s_b^2 via MC simulation, generating $n = 10$ i.i.d. values from a normal distribution. Plot also s^2 and comment the difference.

```
set.seed(1)
require(ggplot2)
require(extrafont)

R <- 100000; n <- 10;
samples     <- matrix(NA, nrow = n, ncol = R);

vector_s   <- rep(NA, 100000);
vector_sb  <- rep(NA, 100000);

for(i in 1:R)
{
  samples[, i]  <- rnorm(n, 0, 1);
  vector_s[i]   <- var(samples[, i]);
  vector_sb[i]  <- 1/n * sum((samples[, i] - mean(samples[, i]))^2);
}

ggplot() +

  # True Mean
  geom_line(aes(x = rep(1, 2), y = c(0, 1)),
            color      = "black",
            size       = 1,
            linetype = "dashed") +

  geom_text(aes(x = 1.15, y = 1),
           label = expression(mu),
           size = 4) +

  # Density (s)
  geom_line(aes(x = vector_sb, y = (n - 1)/1 * dchisq(vector_sb * (n - 1)/1, n - 1)),
            colour = "indianred",
            size   = 1) +

  # Density (sb)
  geom_line(aes(x = vector_sb, y = (n)/1 * dchisq(vector_sb * (n)/1, n)),
```

```
          colour = "dodgerblue4",
          size   = 1) +

# Histogram (s)
geom_histogram(aes(x = vector_s, y = ..density..),
               binwidth = 0.25,
               colour   = "black",
               fill     = "indianred",
               alpha    = 0.2) +

# Histogram (sb)
geom_histogram(aes(x = vector_sb, y = ..density..),
               binwidth = 0.25,
               colour   = "black",
               fill     = "dodgerblue4",
               alpha    = 0.2) +

# Custom Label
geom_line(aes(x = c(2.8, 2.9), y = rep(0.9375, 2)),
          color = "indianred",
          size  = 1) +
geom_text(aes(x = 3.15, y = 0.9375),
          label = "unbiased",
          size = 4) +

geom_line(aes(x = c(2.8, 2.9), y = rep(0.8525, 2)),
          color = "dodgerblue",
          size  = 1) +
geom_text(aes(x = 3.090, y = 0.8525),
          label = "biased",
          size = 4) +
labs(x = "x", y = "f(x)") +
theme_bw(base_size = 10, base_family = "Times")
```

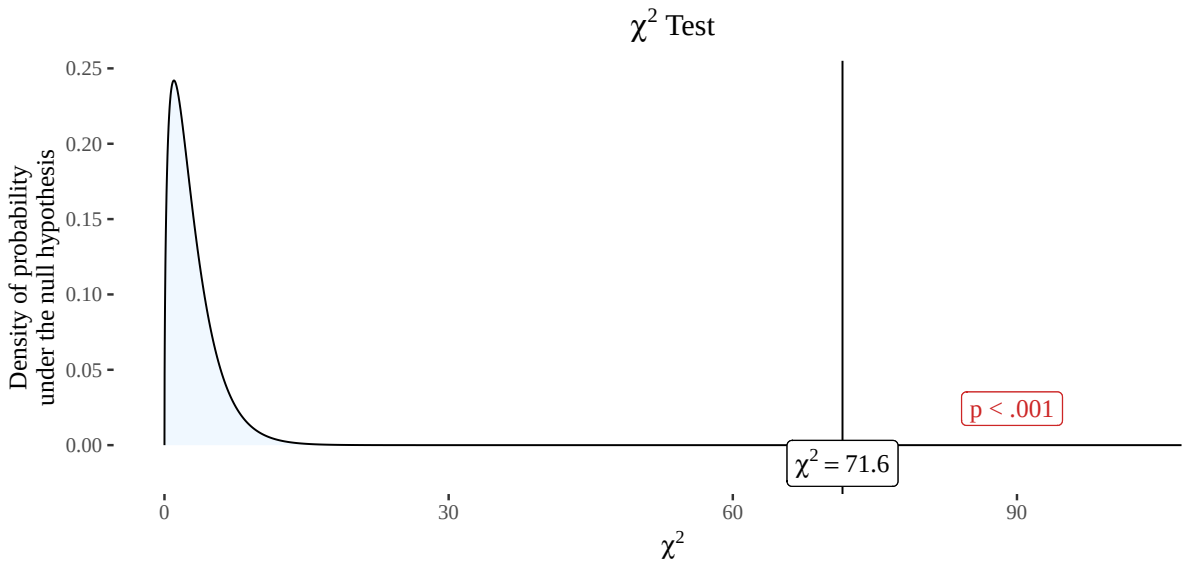


Considering the small size of n, the correction introduced within the unbiased estimator of the variance has considerably improved it's estimation of the true variance. In fact, by looking at the histogram of the two distributions, the unbiased one tends more to the real distribution of the variance, while the biased one appears to have heavier tails, which lower the overall variance but increase the bias of the estimation. This difference disappears when increasing the sample size due to the two estimators being consistent.

Exercise 2

What happens if a great player decides to join you, now? Try to simulate the data and perform the test again.

```
set.seed(1)
require(nhstplot)

n <- 50; K <- 4;
y3 <- sample(1:K, n, replace = TRUE, prob = c(3/16, 4/16, 5/16, 4/16))
observed <- table(y3)
plotchisqtest(chisq.test(observed, p = c(7/16, 5/16, 3/16, 1/16)))
```



Starting from a premise, in our opinion the assumption of i.i.d. samples is not observed in this example. The reason is due to the target correction of the players on each shot subsequent to the first one. However, considering an hypothetical probability distribution related to an experienced dart player, whose goal is to center the target, a comparison of the latter with our prior belief through the chi-square test for goodness of fit allows us to reject the H_0 in favor of the alternative hypothesis that the underneath probability of the experienced dart player is different from the proposed one.

Exercise 3

Consider now some of the most followed instagram accounts in 2018: for each of the owners, we report also the number of Twitter followers (in millions). Are the instagram and Twitter account somehow associated? Perform a correlation test, compute the p-value and give and answer.

```r
set.seed(1)

Instagram <- c(69, 98, 107, 123, 110, 118, 135, 67)
Twitter   <- c(109, 106, 86, 72, 59, 57, 56, 56)

n <- 8
r <- cor(Instagram, Twitter);
T <- r * sqrt((n - 2)/(1 - r^2))

# p-value is > 0.05, we don't reject H_0.
p_value <- 2 * (1 - pt(abs(T), n - 2));
cat("p-value: ", p_value)
```

```
p-value:  0.2956669
```

```r
# Confidence interval for correlation - manual
z_r <- (1/2) * log((1 + r)/(1 - r))
SE  <- 1/(sqrt(n - 3))

z_L <- z_r - pnorm(1 - 0.05/2) * SE
z_U <- z_r + pnorm(1 - 0.05/2) * SE

r_L <- (exp(2 * z_L) - 1)/(exp(2 * z_L) + 1)
r_U <- (exp(2 * z_U) - 1)/(exp(2 * z_U) + 1)

# Confidence interval for correlation - cor.test()
CI <- cor.test(Instagram, Twitter,
               method = ("pearson"),
               alternative = c("two.sided"),
               conf.level = 0.95)
cat(CI$conf.int[1], " < ", r, " < ", CI$conf.int[2]);
```

```
-0.8689006  <  -0.4235845  <  0.4006898
```

We obtain $(-0.87, 0.40)$ as the confidence interval for population's correlation coefficient. Because this interval covers most of the possible values and is distributed around 0, we further confirm our previous conclusion, that it's not possible to associate a specific trend to the correlation of the samples we're analyzing.

DAAG Exercises

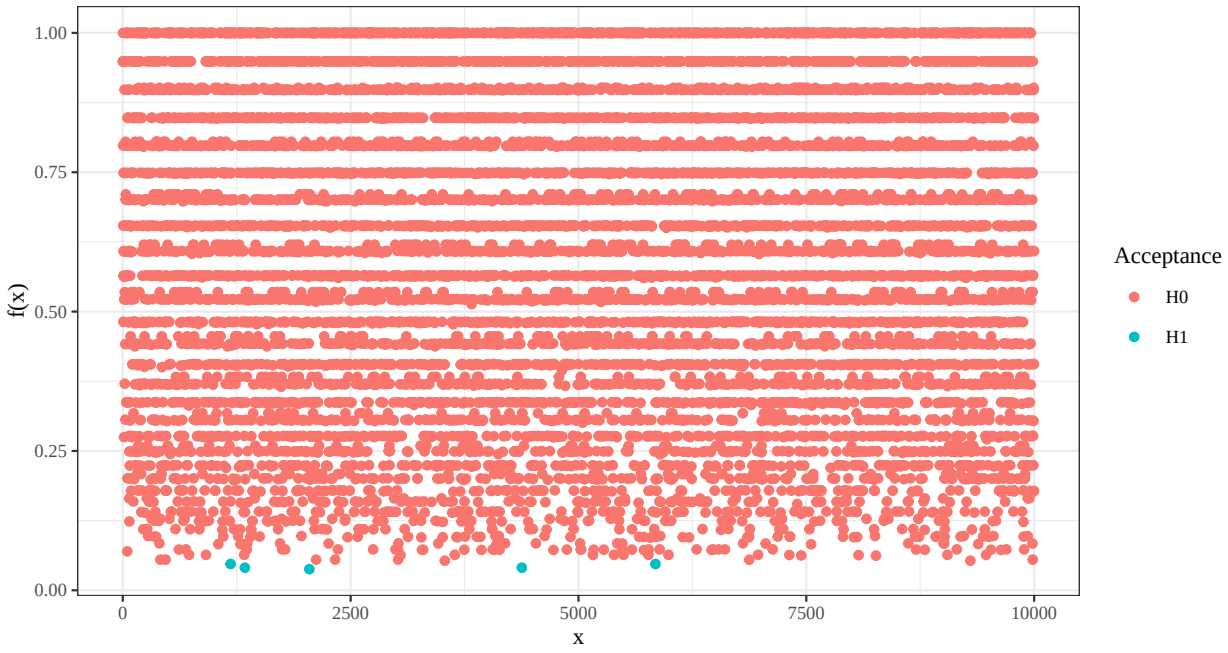
Exercise 11, Chapter 3

```r
x <- c(87, 53, 72, 90, 78, 85, 83);
x_mean <- rep(NA, 100000); x_var  <- rep(NA, 100000);

for(i in 1:100000)
{
  x_mean[i] = mean(rpois(7, 78.3))
  x_var[i]  = var(rpois(7, 78.3))
```

```
}

inf <- 78.3 - qnorm(1 - 0.05/2) * sqrt(78.3)/sqrt(7);
sup <- 78.3 + qnorm(1 - 0.05/2) * sqrt(78.3)/sqrt(7);
cat(inf, " < ", mean(x), " < ", sup);
```

```
71.74489  <  78.28571  <  84.85511
```

```
wilcox_t <- data.frame(p_values = rep(NA, 10000))
for(i in 1:10000){wilcox_t[i, 1] = wilcox.test(rpois(7, 78.3), x)$p.value}
wilcox_t$Acceptance <- ifelse(wilcox_t[, 1] > 0.05, "H0", "H1")

ggplot() +
  geom_point(data = wilcox_t, aes(x   = c(1:10000),
                                  y   = p_values,
                                  col = Acceptance)) +
  labs(x = "x", y = "f(x)") +
  theme_bw(base_size = 10, base_family = "Times")
```



Exercise 13, Chapter 3

```
set.seed(1)
require(zoo)
require(grid)
require(lattice)
require(latticeExtra)
```

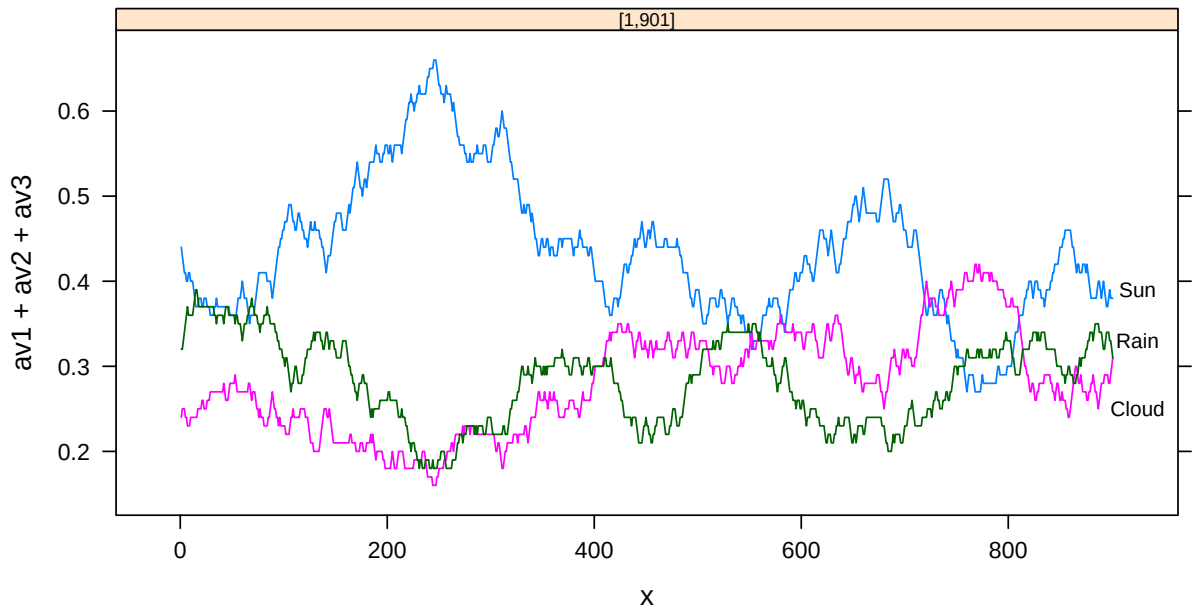
```
Pb <- rbind(c(0.6, 0.2, 0.2), c(0.2, 0.4, 0.4), c(0.4, 0.3, 0.3))
print(Pb)
```

```
     [,1] [,2] [,3]
[1,]  0.6  0.2  0.2
[2,]  0.2  0.4  0.4
[3,]  0.4  0.3  0.3
```

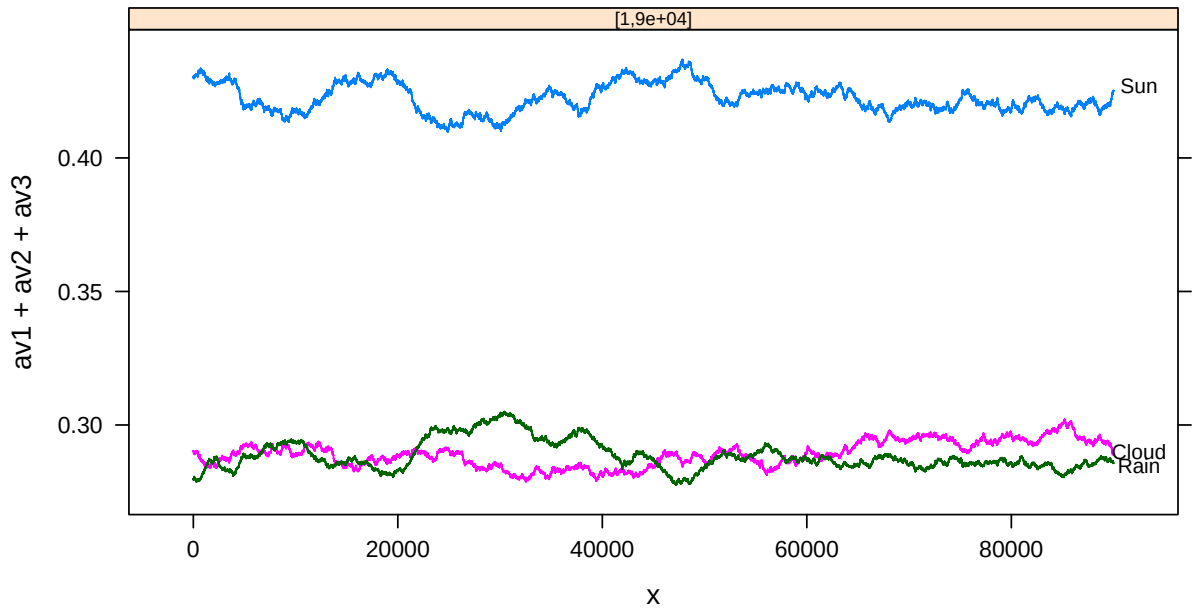
```
Markov <- function(n = 1000, start = 1, transition = Pb)
{
  x <- c(start, numeric(n - 1));
  for(i in 2:n)
  {x[i] <- sample(1:ncol(Pb), size = 1, prob = transition[x[i-1], ])};

  return(x);
}
cat(as.vector(table(Markov(1000, 1, Pb))/
              sum(table(Markov(1000, 1, Pb))))==c(0.641, 0.208, 0.151))
```

```
FALSE FALSE FALSE
```

```
plotmarkov <- function(n = 1000, start = 1, window = 100, transition = Pb, npanels = 1)
{
  xc2  <- Markov(n, start, transition)
  mav1 <- rollmean(as.integer(xc2==1), window)
  mav2 <- rollmean(as.integer(xc2==2), window)
  mav3 <- rollmean(as.integer(xc2==3), window)

  npanel <- cut(1:length(mav1),
                breaks = seq(from = 1, to = length(mav1), length = npanels + 1),
                include.lowest=TRUE)

  df <- data.frame(av1 = mav1, av2 = mav2, av3 = mav3,
                   x = 1:length(mav1), gp = npanel)
  return(df);
}

df <- plotmarkov();
xyplot(av1 + av2 + av3 ~ x | gp, data = df,
       layout = c(1, 1),
       type = "l",
       par.strip.text = list(cex = 0.65),
       scales = list(x = list(relation = "free")),
       panel = function(x, y, ...){panel.xyplot(x, y, ...);
         ltext(rep(925, 3), c(0.39, 0.25, 0.33),
         labels = c("Sun", "Cloud", "Rain"), cex = 0.75)})
```

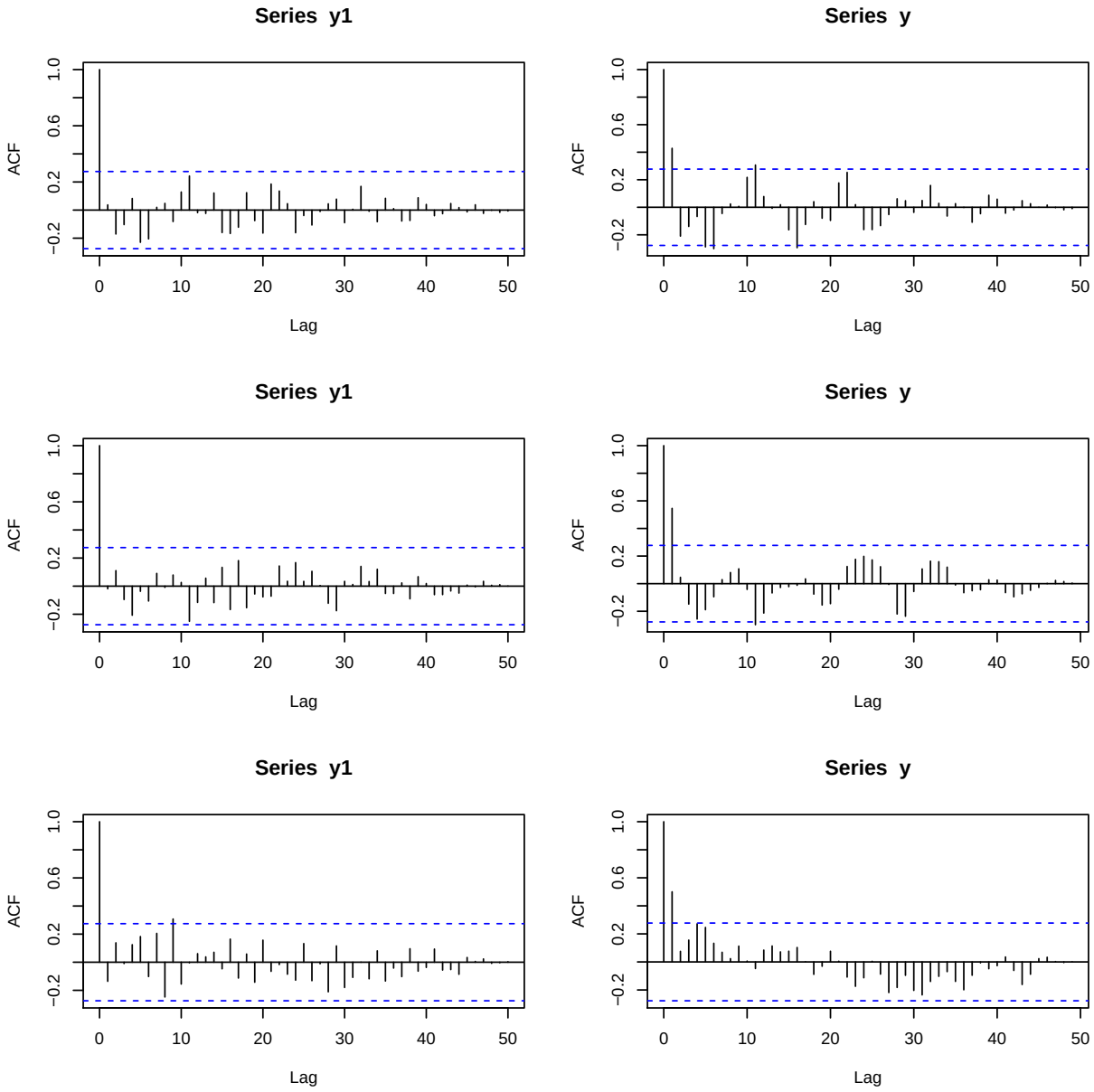



```
df <- plotmarkov(100000, 1, 10000)
xyplot(av1 + av2 + av3 ~ x | gp, data = df,
       layout = c(1, 1),
       type = "l",
       par.strip.text = list(cex = 0.65),
       scales = list(x = list(relation = "free")),
       panel = function(x, y, ...){panel.xyplot(x, y, ...);
         ltext(rep(92500, 3), c(0.427, 0.29, 0.285),
         labels = c("Sun", "Cloud", "Rain"), cex = 0.75)})
```



Exercise 6, Chapter 4

```
set.seed(1)
require(forecast)

par(mfrow = c(3, 2), oma = c(0, 0, 0, 0))
for(i in 1:3)
{
  y1 <- rnorm(51)
  y  <- y1[-1] + y1[-51]

  acf(y1, lag.max = 50)
  acf(y,  lag.max = 50)
}
```



In the above acf() plots, we can notice that there is a significant spike at a lag 1 of *Series y* in all of the analyzed samples, which may suggest that a first-order autoregression model would likely be feasible for this data set, and much lower spikes for the subsequent lags. The general trend of *Series y* appears to follow a specific pattern, with subsequent lags to the first having predictable behavior. On the other side, although some specific lags may give us some information about existing autocorrelation between data, the overall behavior in *Series y*1 appears to be quite randomic and less consistent in consecutive values.

Exercise 7, Chapter 4

```
set.seed(1)

series <- function(x)
```

```
{
  y1 = rnorm(x);
  y  = y1[-1] + y1[-x];
  return(y);
}

av <- rep(NA, 25); v <- rep(NA, 25);

for(i in 1:25)
{
  y     = series(51);
  av[i] = mean(y);
  v[i]  = var(y);
}

cat(var(av))
```

```
> 0.08020313
```