Project 1

Ola Rasmussen

222222222222# Problem 1:

a)

$$P(X_0 = k) = \alpha_k, \quad k = -1, 0, 1$$
$$P(X_n = k | X_{n-1} = j) = \beta_{jk}, \quad j, k = -1, 0, 1$$
$$Z_n = \sum_{s=0}^{n} X_s$$

$\{X_n\}_{s=0}^n$ is a markov chain, and F_n contains all the info of $\{X_n\}_{s=0}^n$ and $\{Z_n\}_{s=0}^n$.

Our goal in this problem is to find the restrictions of α_k and β_{jk} so that $\{Z_n\}_{s=0}^n$ is a zero-mean martingale.

In other words, we need to show that $E[Z_n | F_{n-1}] = Z_{n-1}$.

$$\begin{aligned}
E[Z_n|F_{n-1}] &= E\left[\sum_{s=0}^{n} X_s \middle| F_{n-1}\right] \\
&= \sum_{s=0}^{n-1} X_s + E[X_n|F_{n-1}], \quad \textit{using the markov property} \\
&= Z_{n-1} + E[X_n|X_{n-1}] \\
&= Z_{n-1} + \sum_{k=-1}^{1} kP(X_n = k|X_{n-1}) \\
&= Z_{n-1} + \Big(-1 \cdot P(X_n = -1|X_{n-1}) + 0 \cdot P(X_n = 0|X_{n-1}) + 1 \cdot P(X_n = 1|X_{n-1})\Big) \\
&= Z_{n-1} + \Big(-\Big(P(X_n = -1|X_{n-1} = -1)P(X_{n-1} = -1) + P(X_n = -1|X_{n-1} = 0)P(X_{n-1} = 0) \\
&\qquad + P(X_n = -1|X_{n-1} = 1)P(X_{n-1} = 1)\Big) \\
&\quad + \Big(P(X_n = 1|X_{n-1} = -1)P(X_{n-1} = -1) + P(X_n = 1|X_{n-1} = 0)P(X_{n-1} = 0) \\
&\qquad + P(X_n = 1|X_{n-1} = 1)P(X_{n-1} = 1)\Big)\Big) \\
&= Z_{n-1} + \Big(-\Big(\beta_{-1,-1}\alpha_{-1} + \beta_{0,-1}\alpha_0 + \beta_{1,-1}\alpha_1\Big) + \Big(\beta_{-1,1}\alpha_{-1} + \beta_{0,1}\alpha_0 + \beta_{1,1}\alpha_1\Big)\Big) \\
&= Z_{n-1} + \alpha_{-1}(\beta_{-1,1} - \beta_{-1,-1}) + \alpha_0(\beta_{0,1} - \beta_{0,-1}) + \alpha_1(\beta_{1,1} - \beta_{1,-1})
\end{aligned}$$

So we need $\alpha_{-1}(\beta_{-1,1} - \beta_{-1,-1}) + \alpha_0(\beta_{0,1} - \beta_{0,-1}) + \alpha_1(\beta_{1,1} - \beta_{1,-1}) = 0$, i.e. we need $\beta_{j,1} = \beta_{j,-1}$ and $\beta_{j,0} = 1 - 2\beta_{j,1} = 1 - 2\beta_{j,-1}$ where $j = -1, 0, 1$, and we need that $\sum_{j=-1}^{1} \beta_{jk} = 1$. We also need $\sum_{k=-1}^{1} \alpha_k = 1$.

Using $\alpha_{-1} = \alpha_1 = 0$, $\alpha_0 = 1$, and β_{jk}'s given by the probability transition matrix

$$P = \begin{bmatrix} 0.48 & 0.4 & 0.48 \\ 0.01 & 0.98 & 0.01 \\ 0.49 & 0.2 & 0.49 \end{bmatrix}$$

we also get

$$\begin{aligned} E[Z_n|F_{n-1}] &= \sum_{s=0}^{n-1} X_s + 0 + (0.01 - 0.01) + 0 \\ &= \sum_{s=0}^{n-1} X_s \\ &= Z_{n-1} \end{aligned}$$

b)

The formula for the predictable variation process is $\langle Z \rangle_n = \sum_{i=1}^{n} E[(Z_i - Z_{i-1})^2|F_{n-1}]$, and the formula for the optional variation process is $[Z]_n = \sum_{i=1}^{n}(Z_i - Z_{i-1})^2$. Using these we find;

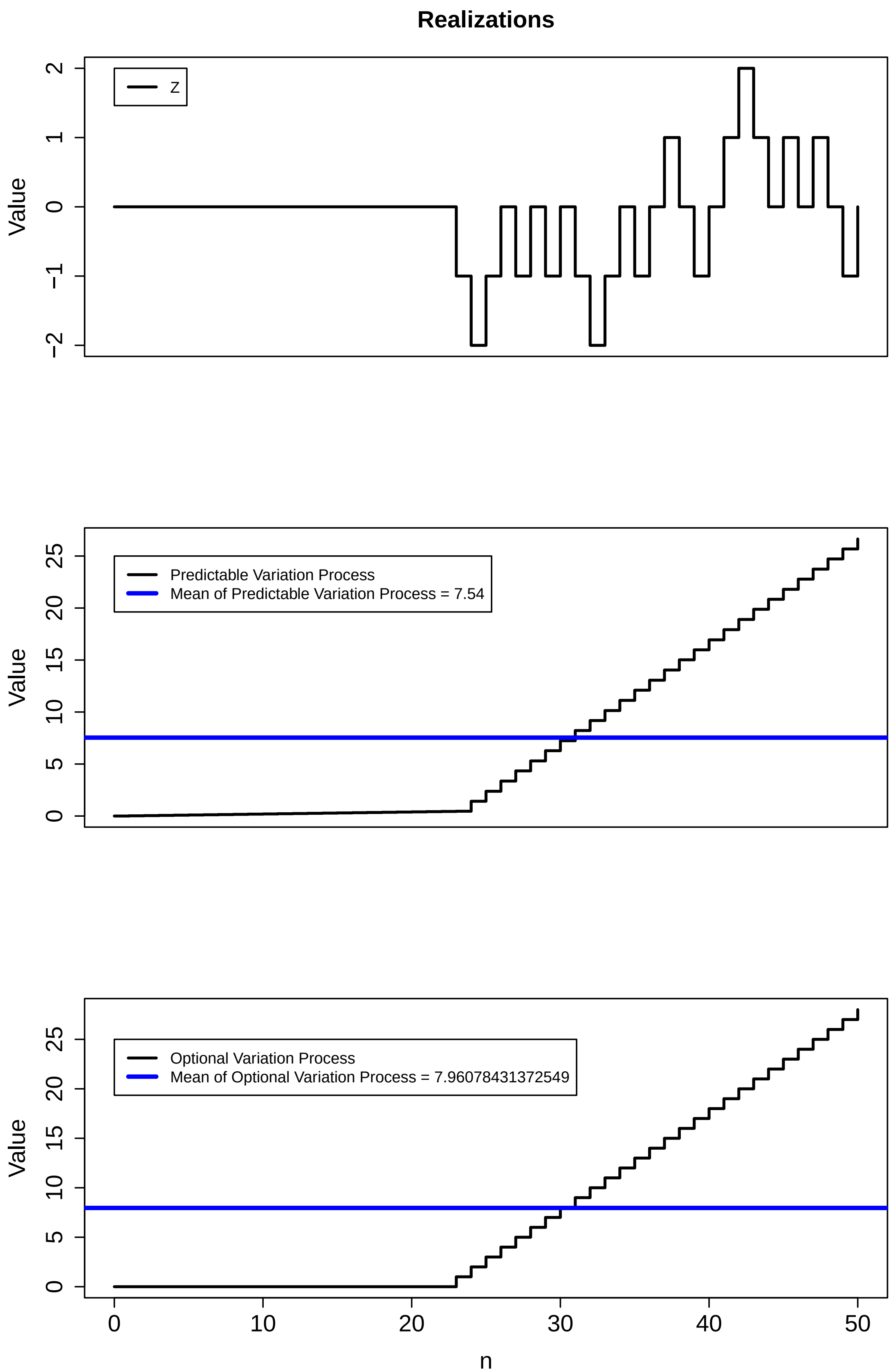
$$\begin{aligned} \langle Z \rangle_n &= \sum_{i=1}^{n} E[(Z_i - Z_{i-1})^2|F_{n-1}] \\ &= \sum_{i=1}^{n} E[X_i^2|F_{n-1}] \\ &= \sum_{i=1}^{n} E[X_i^2|X_{n-1}], \quad \textit{using the markov property} \\ &= \sum_{i=1}^{n} \sum_{k=0}^{1} kP(X_i^2 = k|X_{i-1}), \quad \textit{using the markov property} \\ &= \sum_{i=1}^{n} P(X_i^2 = 1|X_{i-1}) \\ &= \sum_{i=1}^{n} \Big(P(X_i = -1|X_{i-1}) + P(X_i = 1|X_{i-1}) \Big) \\ &= \sum_{i=1}^{n} (\beta_{i,-1} + \beta_{i,1}) \end{aligned}$$

$$\begin{aligned} [Z]_n &= \sum_{i=1}^{n} (Z_i - Z_{i-1})^2 \\ &= \sum_{i=1}^{n} X_i^2 \end{aligned}$$

c)

```r
simFun <- function(N, alpha, beta) {
    x <- rep(0, N + 1)

    # Initial state
    u <- runif(1)
    if (u <= alpha[1]) {
        x[1] <- 0
    } else if (u <= alpha[2] + alpha[1]) {
        x[1] <- 1
    } else if (u >= 1 - alpha[3]) {
        x[1] <- 2
    }

    # Simulate next steps
    for (n in 1:N) {
        x[n + 1] <- sample.int(3, size = 1, replace = TRUE, prob = beta[x[n] +
            1, ]) - 1
    }
    x <- x - 1

    Z <- rep(0, N + 1)
    for (i in 1:(N + 1)) {
        Z[i] <- sum(x[1:i])
    }

    predZ <- rep(0, N + 1)
    test <- rep(0, N + 1)
    for (i in 2:(N + 1)) {
        test[i] <- beta[x[i - 1] + 2, 1] + beta[x[i - 1] + 2,
            3]
        predZ[i] <- sum(test[1:i])
    }

    optZ <- rep(0, N + 1)
    for (i in 1:(N + 1)) {
        optZ[i] <- sum(x[1:i]^2)
    }

    return(cbind(x, Z, predZ, optZ))
}
```

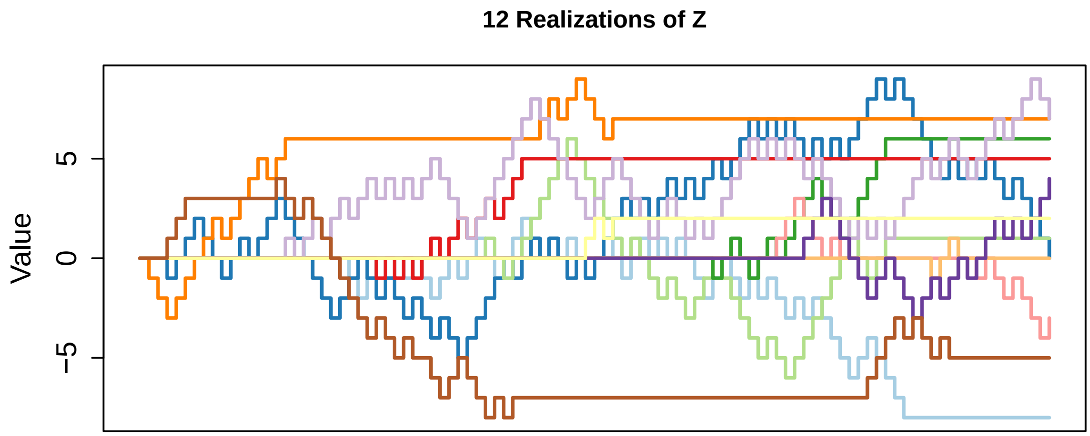
```r
set.seed(98)

N <- 50
alpha <- c(0, 1, 0)
beta <- matrix(c(0.48, 0.04, 0.48, 0.01, 0.98, 0.01, 0.49, 0.02,
    0.49), nrow = 3, byrow = T)

sim <- simFun(N, alpha, beta)

par(mfrow = c(3, 1))

plot(0:N, sim[, 2], type = "s", lwd = 2, cex.axis = 1.5, main = "Realizations",
    xlab = "", ylab = "Value", cex.lab = 1.5, cex.main = 1.5,
    xaxt = "n")
legend(0, 2, c("Z"), col = c("black"), lty = c(1), lwd = c(2))

plot(0:N, sim[, 3], type = "s", lwd = 2, cex.axis = 1.5, xlab = "",
    ylab = "Value", cex.lab = 1.5, cex.main = 1.5, xaxt = "n")
abline(h = mean(sim[, 3]), col = "blue", lwd = 3)
legend(0, 25, c("Predictable Variation Process", paste("Mean of Predictable Variation Pr
    mean(sim[, 3]))), col = c("black", "blue"), lty = c(1, 1),
    lwd = c(2, 3))

plot(0:N, sim[, 4], type = "s", lwd = 2, cex.axis = 1.5, xlab = "n",
    ylab = "Value", cex.lab = 1.5, cex.main = 1.5)
abline(h = mean(sim[, 4]), col = "blue", lwd = 3)
legend(0, 25, c("Optional Variation Process", paste("Mean of Optional Variation Process
    mean(sim[, 4]))), col = c("black", "blue"), lty = c(1, 1),
    lwd = c(2, 3))
```

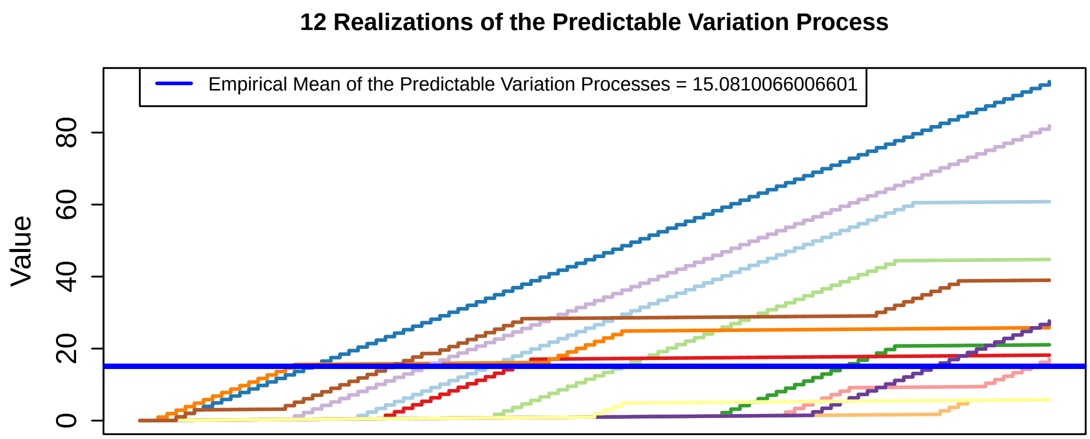

Realizations
Value
z
2
1
0
−1
−2
Value
Predictable Variation Process
Mean of Predictable Variation Process = 7.54
0
5
10
15
20
25
Value
Optional Variation Process
Mean of Optional Variation Process = 7.96078431372549
0
5
10
15
20
25
0
10
20
30
40
50
n

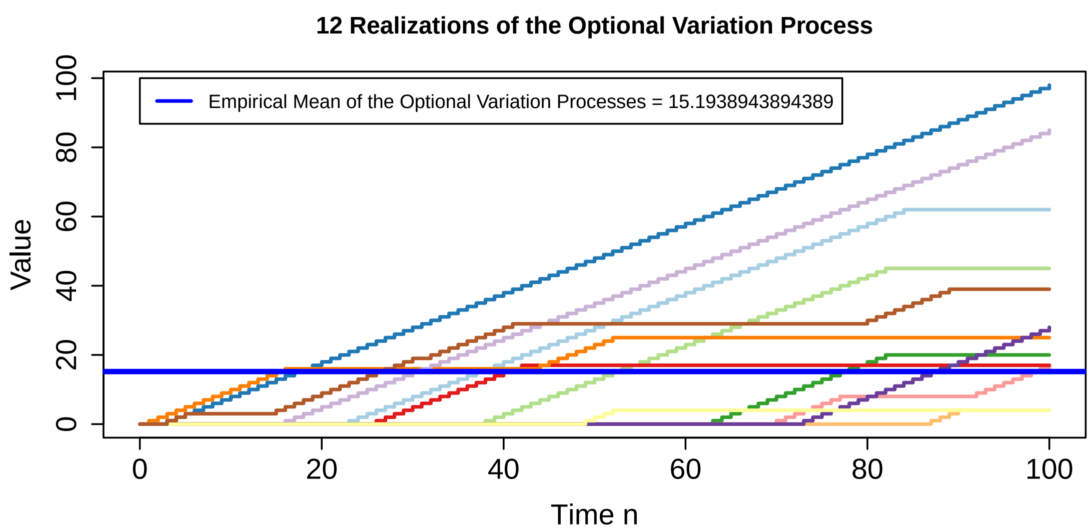
d)

```r
N <- 100
M <- 12
Zmatr <- matrix(0, nrow = M, ncol = N + 1)
for (i in 1:M) {
    set.seed(97 + i)
    Zmatr[i, ] <- simFun(N, alpha, beta)[, 2]
}
empMeanpred <- c()
predZmatr <- matrix(0, nrow = M, ncol = N + 1)
for (i in 1:M) {
    set.seed(97 + i)
    predZmatr[i, ] <- simFun(N, alpha, beta)[, 3]
    empMeanpred <- append(empMeanpred, mean(simFun(N, alpha,
        beta)[, 3]))
}

empMeanopt <- c()
optZmatr <- matrix(0, nrow = M, ncol = N + 1)
for (i in 1:M) {
    set.seed(97 + i)
    optZmatr[i, ] <- simFun(N, alpha, beta)[, 4]
    empMeanopt <- append(empMeanopt, mean(simFun(N, alpha, beta)[,
        4]))
}

cols <- brewer.pal(M, "Paired")
par(mfrow = c(3, 1))
plot(NULL, NULL, xlim = c(0, N), ylim = c(min(Zmatr), max(Zmatr)),
    main = paste(M, "Realizations of Z"), xlab = "", ylab = "Value",
    cex.axis = 1.5, cex.lab = 1.5, lwd = 1.5, xaxt = "n")
for (i in 1:M) {
    lines(0:N, Zmatr[i, ], col = cols[i], lwd = 2, type = "s")
}
plot(NULL, NULL, xlim = c(0, N), ylim = c(min(predZmatr), max(predZmatr)),
    main = paste(M, "Realizations of the Predictable Variation Process"),
    xlab = "", ylab = "Value", cex.axis = 1.5, cex.lab = 1.5,
    lwd = 1.5, xaxt = "n")
for (i in 1:M) {
    lines(0:N, predZmatr[i, ], col = cols[i], lwd = 2, type = "s")
}
abline(h = (sum(empMeanpred)/M), col = "blue", lwd = 3)
legend(0, 100, paste("Empirical Mean of the Predictable Variation Processes =",
```

```r
    (sum(empMeanpred)/M)), col = "blue", lty = c(1, 1), lwd = c(2,
    3))

plot(NULL, NULL, xlim = c(0, N), ylim = c(min(optZmatr), max(optZmatr)),
    main = paste(M, "Realizations of the Optional Variation Process"),
    xlab = "Time n", ylab = "Value", cex.axis = 1.5, cex.lab = 1.5,
    lwd = 1.5)
for (i in 1:M) {
    lines(0:N, optZmatr[i, ], col = cols[i], lwd = 2, type = "s")
}
abline(h = (sum(empMeanopt)/M), col = "blue", lwd = 3)
legend(0, 100, paste("Empirical Mean of the Optional Variation Processes =",
    (sum(empMeanopt)/M)), col = "blue", lty = c(1, 1), lwd = c(2,
    3))
```


12 Realizations of Z
Value
5
0
−5


12 Realizations of the Predictable Variation Process
Empirical Mean of the Predictable Variation Processes = 15.0810066006601
Value
0
20
40
60
80


12 Realizations of the Optional Variation Process
Empirical Mean of the Optional Variation Processes = 15.1938943894389
Value
0
20
40
60
80
100
Time n

Here we can see that Z tends to stay within a range of 10 from zero

e)

$$U_n = \sum_{s=0}^{n} \frac{X_s}{n-s+1}$$

Our goal now is to find the Doob decomposition of $\{U_n\}_{n=0}^{\infty}$. i.e. $U_n = E[U_n|F_{n-1}] + \Delta M_n$, where $\{M_n\}_{n=0}^{\infty}$ is a zero-mean martingale and $\Delta M_n = M_n - M_{n-1}$ for $n = 0, 1, ...$

$$\begin{aligned} E[U_n|F_{n-1}] &= E\left[\sum_{s=0}^{n} \frac{X_s}{n-s+1}\middle| F_{n-1}\right] \\ &= \sum_{s=0}^{n-1} \frac{X_s}{n-s+1} + E\left[\frac{X_s}{n-n+1}\middle| F_{n-1}\right] \\ &= U_{n-1} + E[X_n|X_{n-1}], \quad \textit{using the markov property} \end{aligned}$$

We know that $E[X_n|X_{n-1}] = 0$ because $\sum_{s=0}^{n-1} X_s + E[X_n|X_{n-1}] = Z_{n-1} = \sum_{s=0}^{n-1} X_s \Rightarrow E[X_n|X_{n-1}] = 0$, which means that $E[U_n|F_{n-1}] = U_{n-1}$.

$$\begin{aligned} U_n - E[U_n|F_{n-1}] &= U_n - U_{n-1} \\ &= X_n \\ &= \Delta M_n \end{aligned}$$

So the Doob decomposition is:

$$U_n = U_{n-1} + X_n$$

Problem 2:

a)

We have:

$$N(t) = \sum_{i=1}^{n} N_i(t)$$

$$Y(t) = 1 + \sum_{i=2}^{n} Y_i(t)$$

$$dN_i(t) = N_i((t+dt)^-) - N_i(t^-)$$

$$P(dN_i(t) = 1 | N_i(s), Y_i(s); s \in [0,t]) = Y_i(t)\alpha(t)dt$$

The general formulation of the Doob-Meyer decomposition is $N(t) = N^*(t) + M(t)$. $N(t)$ is trivially a semi-martingale because is always increasing or staying the same, $E[N(t)|F_s] \geq N(s)$.

In particular, we want to use $dN^*(t) = E[dN(t)|F_{t^-}]$ to show that the compensator $N^*(t)$ of $N(t)$ is $N^*(t) = \int_0^t \alpha(t)Y(t)dt$.

$$\begin{aligned}
dN^*(t) &= E[dN(t)|F_{t^-}] \\
&= E[N((t+dt)^-) - N(t^-)|F_{t^-}] \\
&= E\left[\sum_{i=1}^{n}(N_i((t+dt)^-) - N_i(t^-))|F_{t^-}\right] \\
&= \sum_{i=1}^{n} E[N_i((t+dt)^-) - N_i(t^-)|F_{t^-}] \\
&= \sum_{i=1}^{n} E[dN_i(t)|F_{t^-}] \\
&= \sum_{i=1}^{n}\Big(1 \cdot P(dN_i(t) = 1|F_{t^-}) + 0 \cdot P(dN_i(t) = 0|F_{t^-})\Big) \\
&= \sum_{i=1}^{n} P(dN_i(t) = 1|F_{t^-}) \\
&= \sum_{i=1}^{n} Y_i(t)\alpha(t)dt \\
&= Y(t)\alpha(t)dt
\end{aligned}$$

Using that $\int_0^t dN^*(t) = N^*(t)$, we then get $N^*(t) = \int_0^t Y(t)\alpha(t)dt$.

So the Doob-Meyer decomposition is then $N(t) = \int_0^t Y(t)\alpha(t)dt + M(t)$.

b)

The Doob-Meyer decomposition on increment form is:

$$\begin{aligned} dN(t) &= dN^*(t) + dM(t) \\ &= Y(t)\alpha(t)dt + dM(t) \end{aligned}$$

$$Deviding \;\; by \;\; Y(t), \;\; we \;\; get:$$

$$\frac{dN(t)}{Y(t)} = \alpha(t)dt + \frac{dM(t)}{Y(t)}$$

$$Then \;\; we \;\; integrate \;\; both \;\; sides:$$

$$\int_0^t \frac{dN(s)}{Y(s)} = A(t) + \int_0^t \frac{dM(s)}{Y(s)}$$

$\int_0^t \frac{dM(s)}{Y(s)}$ is a zero-mean martingale because $M(t)$ is a zero-mean martingale. This trivially means that $\hat{A}(t) = \int_0^t \frac{dN(s)}{Y(s)}$ is unbiased, because $E[\hat{A}(t)] = A(t)$.

c)

$$\begin{aligned} \hat{A}(t) &= \sum_{j|T_j<t} \frac{1}{Y(T_j)} \\ &= \int_0^t \frac{dN(s)}{Y(s)} \end{aligned}$$

$$\mathbb{I} = \int_0^t \frac{dM(s)}{Y(s)}$$

$$[M](t) = N(t)$$

We know that $\mathbb{I}(t) = \int_0^t H(s)dM(s)$ and that $[\int HdM] = \int H^2 d[M] = \int H^2 dN$. This means that:

$$
\begin{aligned}
[\mathbb{I}](t) &= \left[\int_0^t \frac{dM(s)}{Y(s)}\right]\Big(t\Big) \\
&= \int_0^t \frac{1}{Y^2(s)} d[M](s) \\
&= \int_0^t \frac{1}{Y^2(s)} dN(s) \\
&= \sum_{j|T_j<t} \frac{1}{Y^2(T_j)}
\end{aligned}
$$

To show that $\hat{\sigma}^2 = \sum_{j|T_j<t} \frac{1}{Y(T_j)^2}$ is an unbiased estimator of $\sigma^2 = Var[\hat{A}(t)]$, we need to show that $E[\hat{\sigma}^2] = \sigma^2$.

We know that $E[\hat{\sigma}^2] = E[[\mathbb{I}](t)]$, and that $\sigma^2 = Var[\hat{A}(t)] = E[[\hat{A}](t)]$. In other words, we need to show that $[\mathbb{I}](t) = [\hat{A}](t)$.

$$
\begin{aligned}
[\hat{A}](t) &= \lim_{n\to t} \sum_{k=1}^{n} (\hat{A}_k - \hat{A}_{k-1})^2 \\
&= \lim_{n\to t} \sum_{k=1}^{n} \left(\frac{1}{Y(T_k)}\right)^2 \\
&= \sum_{j|T_j<t} \frac{1}{Y^2(T_j)} \\
&= [\mathbb{I}](t)
\end{aligned}
$$

d)

In the following problem i will use the following inversion function to use in the inversion sampling method:

$$
\begin{aligned}
\alpha(t) &= \frac{t}{600}, \quad t \le 60 \\
&= 1/10, \quad elsewhere \\
S(t) &= e^{-\int_s^t \alpha(x)dx}, \quad s < t \le 60 \\
&= e^{-\frac{t^2-s^2}{1200}}, \quad s < t \le 60 \\
F(t) &= 1 - S(t) \\
&= 1 - e^{-\frac{t^2-s^2}{1200}}, \quad s < t \le 60 \\
\Rightarrow F^{-1}(u) &= \sqrt{s^2 - 1200 \cdot ln(1-u)}, \quad u \le F(60)
\end{aligned}
$$

```r
# Fail of component 2 to n
failureP <- function(alpha, v, n, tau) {
    en <- c()
    nul <- c()

    while (sum(en) + sum(nul) <= 60) {
        test <- sum(en) + sum(nul)
        en <- append(en, alphas(test))
        nul <- append(nul, rexp(1, v))
    }

    while (sum(en) + sum(nul) <= tau) {
        en <- append(en, rexp(1, 1/10))
        nul <- append(nul, rexp(1, v))
    }

    if (length(en) >= length(nul)) {
        blabla <- length(en)
    } else {
        blabla <- length(nul)
    }

    TT <- c(0)
    TT[2] <- en[1]
    i <- 2
    while (i <= (blabla)) {
        TT[i + 1] <- sum(en[1:(i)]) + sum(nul[1:(i - 1)])
        i <- i + 1
    }

    S <- rep(0, blabla)
    S[1] <- TT[2] + nul[1]
    for (i in 2:(blabla)) {
        S[i] <- TT[i + 1] + nul[i]
    }

    return(cbind(TT, S, en, nul))
}
```

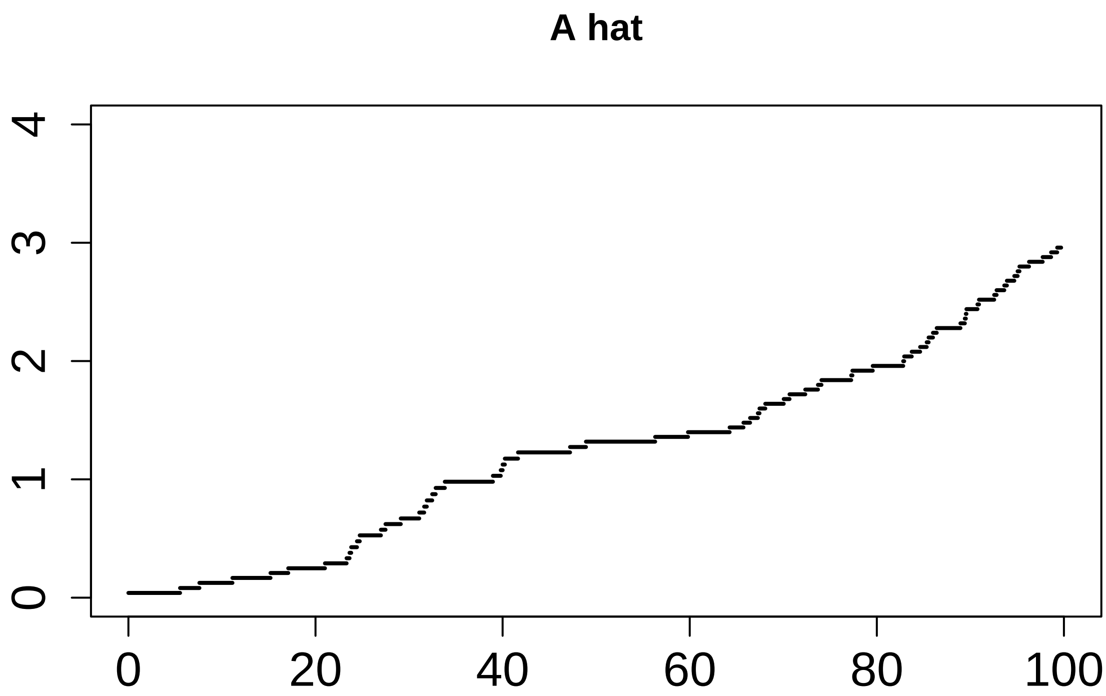
```r
alphas <- function(S) {
    u <- runif(1)
    integral <- (60^2 - S^2)/1200
    if (u <= (1 - exp(-integral))) {
        t <- sqrt(S^2 - (1200 * log(1 - u)))
```

```
    } else {
        t <- (60 - S) + rexp(1/10)
    }
    return(t)
}
v <- 0.2
n <- 24
tau <- 100
```

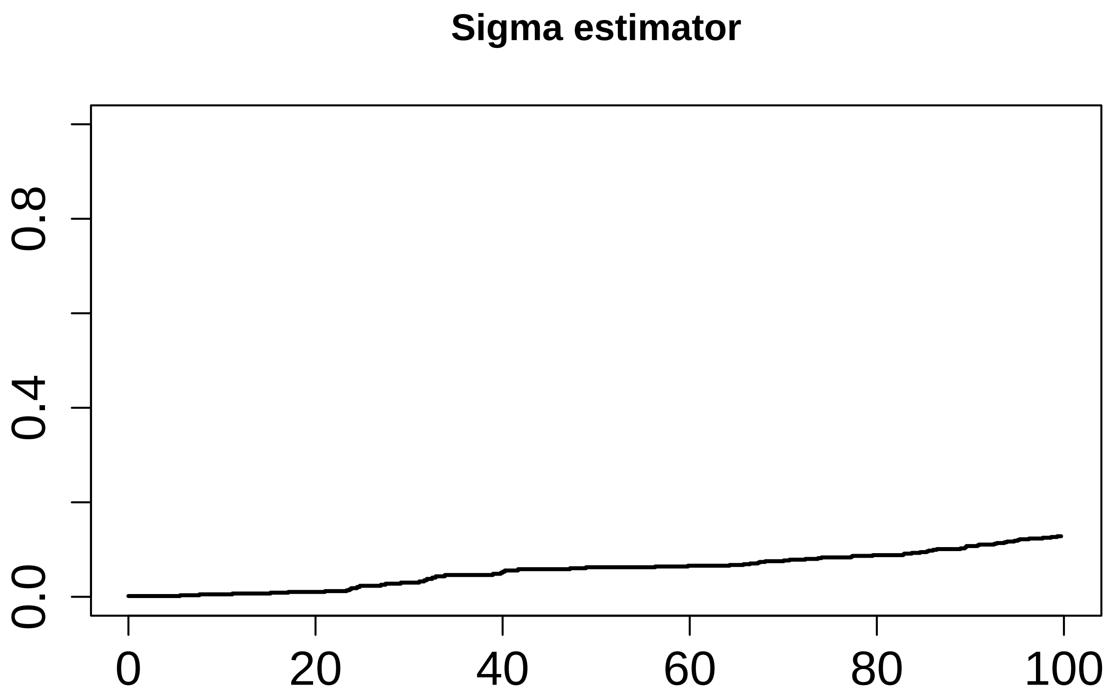
```
TTMATR <- matrix(NA, nrow = n, ncol = 25)
SMATR <- matrix(NA, nrow = n, ncol = 25)
TTT <- c()
for (i in 1:n) {
    set.seed(97 + i)
    fai <- failureP(alphas, v, n, tau)
    TT <- fai[, 1]
    S <- fai[, 2]

    TTMATR[i, ] <- append(TT, rep(NA, ncol(TTMATR) - length(TT)))
    SMATR[i, ] <- append(S, rep(NA, ncol(SMATR) - length(S)))
    TTT <- append(TTT, TT)
}

TTMATR <- TTMATR[1:n, 2:25]

TTMATR[is.na(TTMATR)] <- 0
SMATR[is.na(SMATR)] <- 0

TTT <- unique(sort(TTT))

TTT[TTT >= tau] <- NA

TTT <- na.omit(TTT)

TTT <- TTT[2:length(TTT)]

Y <- c()
hatA <- c()
sigm <- c()
for (j in 1:n) {
    for (k in 1:length(TTT)) {
        Ytid <- c()
        for (i in 1:n) {
            if (((TTT[k] > TTMATR[i, j]) && (TTT[k] < SMATR[i,
```

```
                j])) == T) {
                Ytid <- append(Ytid, 1)
            }
        }
        Y <- append(Y, 1 + (24 - sum(Ytid)))
    }
}

# plot(Y, xlim = c(0,100), type = 's')
```

```
A <- c()
for (i in 1:length(Y)) {
    A <- append(A, 1/Y[i])
}
hatA <- c()
for (i in 1:length(Y)) {
    hatA <- append(hatA, sum(A[1:i]))
}

plot(NULL, NULL, xlim = c(0, tau), ylim = c(0, 4), cex.axis = 1.5,
    cex.lab = 1.5, main = "A hat", xlab = "", ylab = "")
lines(c(0, TTT[1]), c(hatA[1], hatA[1]), lwd = 2)
for (i in 2:length(TTT)) {
    lines(c(TTT[i - 1], TTT[i]), c(hatA[i - 1], hatA[i - 1]),
        lwd = 2)
}
```




```
A <- c()
for (i in 1:length(Y)) {
    A <- append(A, 1/(Y[i])^2)
}
sigm <- c()
for (i in 1:length(Y)) {
    sigm <- append(sigm, sum(A[1:i]))
}

plot(NULL, NULL, xlim = c(0, tau), ylim = c(0, 1), cex.axis = 1.5,
    cex.lab = 1.5, main = "Sigma estimator", xlab = "", ylab = "")
lines(c(0, TTT[1]), c(sigm[1], sigm[1]), lwd = 2)
for (i in 2:length(TTT)) {
    lines(c(TTT[i - 1], TTT[i]), c(sigm[i - 1], sigm[i - 1]),
        lwd = 2)
}
```


Sigma estimator
0.0
0.4
0.8
0
20
40
60
80
100