RF Simulation

2025-10-24

Contents

On Data simulated from Cox

Survival

Simulate data

```
set.seed(373)
beta = matrix(c(1, 0,0.5,1), ncol = 2, nrow = 2)
data <- simSurvData(N = 1000, beta = beta, cens = 0)
```

Fit model

```
#tune.nodesize(Surv(Time,Delta) ~ L0 + A0, data = data)
RF_fit <- rfsrc(Surv(Time, Delta) ~ L0 + A0, data = data, nodesize = 40)
```

Simulate new data

```
list_old_vars <- list(L0 = data$L0, A0 = data$A0)
new_data <- simEventRF(2*10^4, RF_fit, list_old_vars = list_old_vars)
covars <- data.frame(L0 = c(0.5,0.5), A0 = c(1,0))
```

Fit Cox models

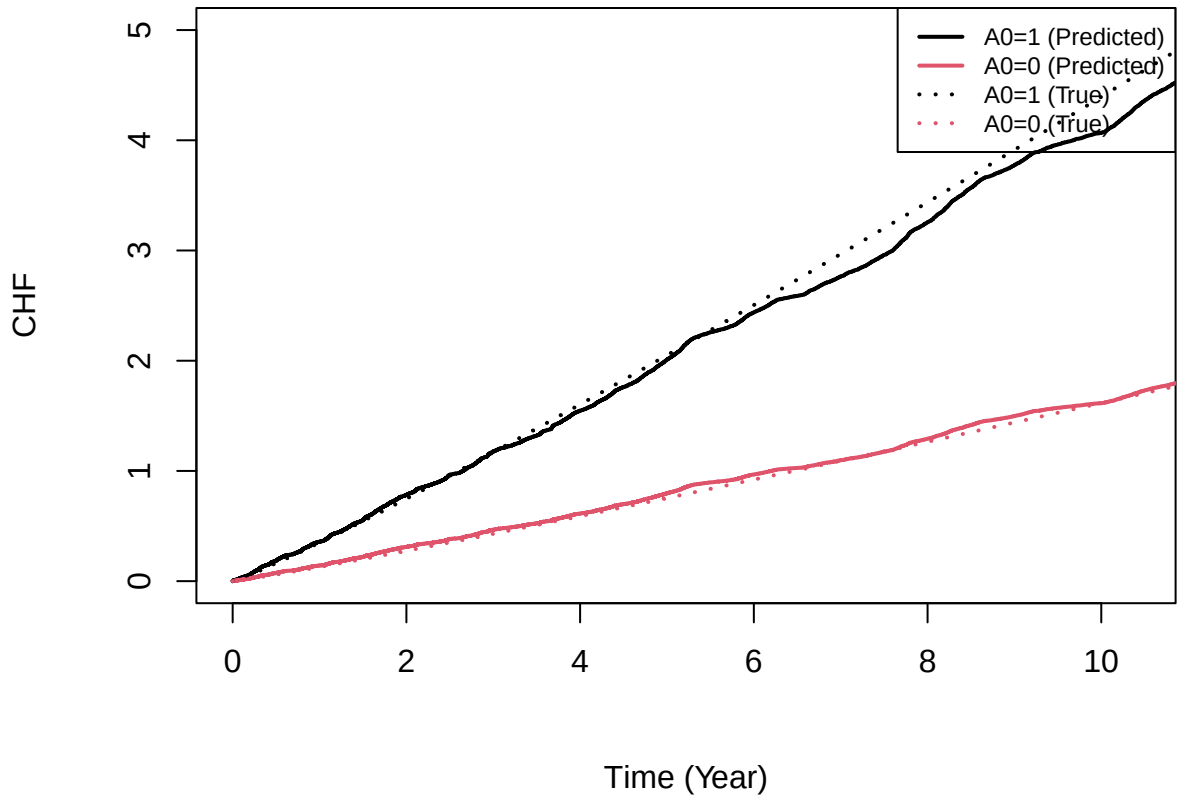
```
cox1 <- coxph(Surv(Time, Delta == 1) ~ L0 + A0, data = new_data)
basehazz1 <- basehaz(cox1, centered = TRUE)
cox_term1 <- exp(predict(cox1, newdata=covars, type="lp"))
cum_int1 <- outer(basehazz1[['hazard']], cox_term1, "*")
```

Evaluate the true survival functions at these time points

```r
true_chf_func1 <- function(t) 0.1  * t^(1.1) * exp(sum(covars[1,] * c(0.5,1)))
true_chf_func2 <- function(t) 0.1  * t^(1.1) * exp(sum(covars[2,] * c(0.5,1)))
time_points <- basehazz1[['time']]
true_chf1 <- true_chf_func1(time_points)
true_chf2 <- true_chf_func2(time_points)
```

Plot

```r
par(mfrow = c(1,1), cex.axis = 1.0, cex.lab = 1.0, cex.main = 1.0, mar = c(6.0,6,1,1), mgp = c(4, 1, 0))
plot(time_points, cum_int1[,1], type="l", xlab="Time (Year)",
     ylab="CHF", col=1, lty=1, lwd=2, xlim = c(0,quantile(new_data$Time, 0.9)),
     ylim = c(0,5))
lines(time_points, cum_int1[,2], col=2, lty=1, lwd=2)
# Add lines for the true survival functions
lines(time_points, true_chf1, col=1, lty=3, lwd=2)
lines(time_points, true_chf2, col=2, lty=3, lwd=2)
legend("topright",
       legend=c("A0=1 (Predicted)", "A0=0 (Predicted)", "A0=1 (True)", "A0=0 (True)"),
       col=c(1, 2, 1, 2),
       lty=c(1, 1, 3, 3),
       cex=0.75,
       lwd=2)
```



Competing Risk

Simulate data

```
set.seed(373)
beta = matrix(c(0.5,-1,-0.5,0.5,0,0.5), ncol = 3, nrow = 2)
data <- simCRdata(N = 3000, beta = beta)
```

Fit RSF model

```
#tune.nodesize(Surv(Time,Delta) ~ L0 + A0, data = data)
RF_fit <- rfsrc(Surv(Time, Delta) ~ L0 + A0, data = data, nodesize = 150)
```

New Data

```
list_old_vars <- list(L0 = data$L0, A0 = data$A0)
new_data <- simEventRF(10^4, RF_fit, list_old_vars =list_old_vars,
                       n_event_max = c(1,1), term_events = c(1,2))
```

Fit Cox models

```
cox1 <- coxph(Surv(Time, Delta == 1) ~ L0 + A0, data = new_data)
cox2 <- coxph(Surv(Time, Delta == 2) ~ L0 + A0, data = new_data)
basehazz1 <- basehaz(cox1, centered = TRUE)
basehazz2 <- basehaz(cox2, centered = TRUE)
cox_term1 <- exp(predict(cox1, newdata=covars, type="lp"))
cox_term2 <- exp(predict(cox2, newdata=covars, type="lp"))
cum_int1 <- outer(basehazz1[['hazard']], cox_term1, "*")
cum_int2 <- outer(basehazz2[['hazard']], cox_term2, "*")
```

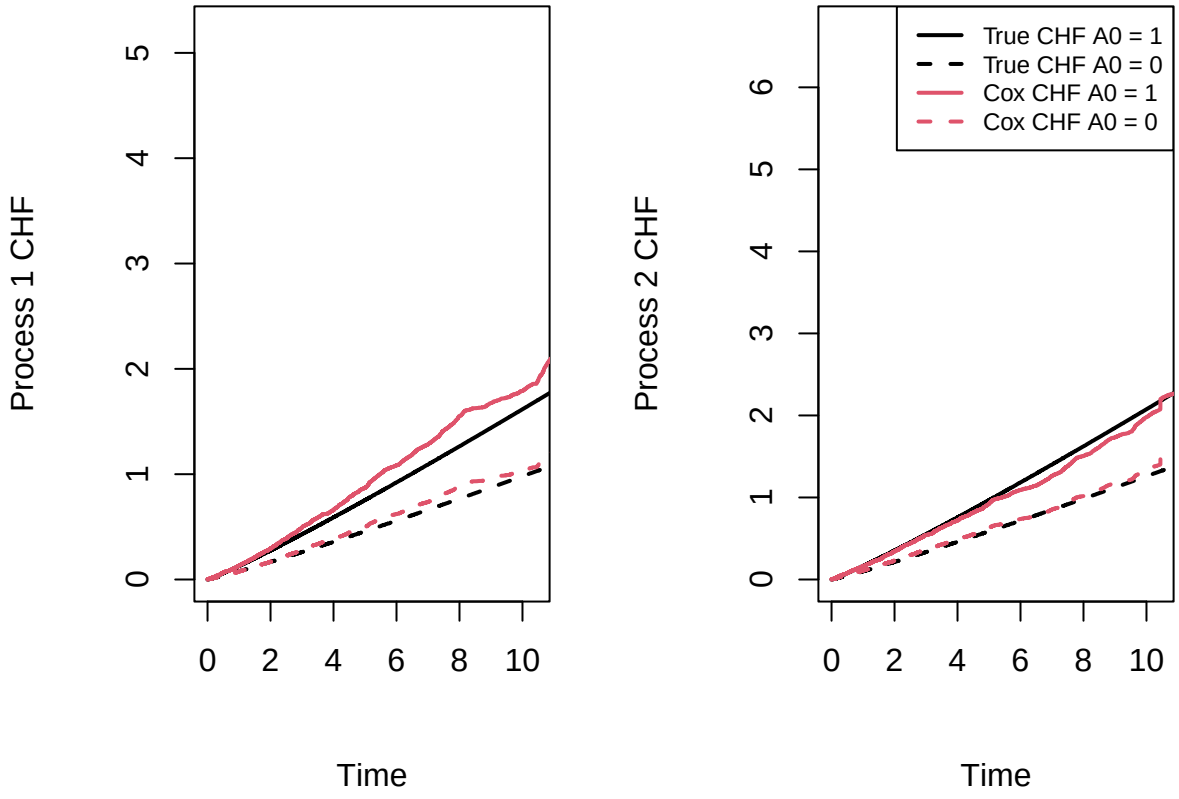
True CHF

```
true_chf_func1 <- function(t, cov)  0.1  * t^(1.1) * exp(sum(cov * c(-0.5,0.5)))
true_chf_func2 <- function(t, cov)  0.1  * t^(1.1) * exp(sum(cov * c(0,0.5)))

# Evaluate the true survival functions at time points
times <- basehazz1[['time']]
true_chf11 <- true_chf_func1(times, covars[1,])
true_chf12 <- true_chf_func1(times, covars[2,])
true_chf21 <- true_chf_func2(times, covars[1,])
true_chf22 <- true_chf_func2(times, covars[2,])
```

Plot

```
par(mfrow = c(1,2), cex.axis = 1.0, cex.lab = 1.0, cex.main = 1.0, mar = c(6.0,6,1,1), mgp = c(4, 1, 0))
# Process 1
plot(times, true_chf11, type="l", xlab="Time",
     ylab="Process 1 CHF", col=1, lty=1, lwd=2, xlim = c(0,quantile(new_data$Time, 0.95)))
lines(times, true_chf12, col=1, lty=2, lwd=2)
lines(times, cum_int1[,1], col=2, lty=1, lwd=2)
lines(times, cum_int1[,2], col=2, lty=2, lwd=2)
```

```r
# Process 2
plot(times,true_chf21, type="l", xlab="Time",
     ylab="Process 2 CHF", col=1, lty=1, lwd=2, xlim = c(0,quantile(new_data$Time, 0.95)))
lines(times, true_chf22, col=1, lty=2, lwd=2)
lines(times, cum_int2[,1], col=2, lty=1, lwd=2)
lines(times, cum_int2[,2], col=2, lty=2, lwd=2)

legend("topright",
       legend = c("True CHF A0 = 1", "True CHF A0 = 0",
                  "Cox CHF A0 = 1", "Cox CHF A0 = 0"),
       col = c(1, 1, 2, 2),
       lty = c(1, 2, 1, 2),
       lwd = 2,
       cex = 0.75)
```



Comparing Simulation Procedures

We interested in how well the simulated data from respectively `simCRdata` and `simEventRF` aligns with "observed" data. We will investigate this by simulating large data sets using both procedures and approximating the cumulative incidence function in a grid of points $t_1, \ldots, t_M$ using Monte Carlo methods. We approximate the cumulative incidence function for cause j as

$$\hat{f}_j(t) = \hat{P}(T \leq t, \Delta = j) = \frac{1}{N} \sum_{i=1}^{N} 1(T_i \leq t, \Delta = j)$$

When we have calculated the approximated CIF in the grid of time points, we can sum the squared differences of the true CIF and the approximated CIF, thereby defining the metric:

$$Q_j = \sum_{i=1}^{M} \left(\hat{f}_j(t_i) - f_j(t_i) \right)^2$$

As a measure of how well our simulated data aligns with the underlying data.

On Data from Proportional Hazard Model

We simulate the "observed" data

```
set.seed(371)
beta = matrix(c(0.5,-1,-0.5,0.5,0,0.5), ncol = 3, nrow = 2)
data <- simCRdata(N = 3000, beta = beta, cens = 0)
```

Approximating the true CIFs

We simulate data from the true distribution:

```
N <- 10^5
true_data <- list()
for(i in 1:10){
  true_data[[i]] <- simCRdata(N = N, beta = beta, cens = 0)
}
true_data <- dplyr::bind_rows(true_data)
```

We approximate the true cumulative incidence function using the simulated data

```
tis <- seq(0.1,max(data$Time)/2, length.out = 20)
CIF_true1 <- numeric(10)
CIF_true2 <- numeric(10)
i <- 1

for(ti in tis){
  CIF_true1[i] <- mean(true_data[,(Time <= ti) & Delta == 1])
  CIF_true2[i] <- mean(true_data[,(Time <= ti) & Delta == 2])
  i <- i + 1
}
```

Approximating the CIFs using data genereted by Cox

Fit Cox models

```
cox1 <- coxph(Surv(Time, Delta == 1) ~ L0 + A0, data = data)
cox2 <- coxph(Surv(Time, Delta == 2) ~ L0 + A0, data = data)
cox_fits <- list("Process 1" = cox1, "Process 2" = cox2)
```

Simulate 10 large datasets using `simEventCox`

```
N <- 3*10^4
new_data <- list()
cox_fitss1 <- list()
cox_fitss2 <- list()
for(i in 1:10){
  new_data[[i]] <- simEventCox(N, cox_fits, L0_old = data$L0, A0_old = data$A0)
  cox_fitss1[[i]] <- confint(coxph(Surv(Time, Delta == 1) ~ L0 + A0, data = new_data[[i]]))
  cox_fitss2[[i]] <- confint(coxph(Surv(Time, Delta == 2) ~ L0 + A0, data = new_data[[i]]))
}
new_data <- dplyr::bind_rows(new_data)
```

Approximate the cumulative incidence function using the Cox data

```
CIF_cox1 <- numeric(10)
CIF_cox2 <- numeric(10)
i <- 1

for(ti in tis){
  CIF_cox1[i] <- mean(new_data[,(Time <= ti) & Delta == 1])
  CIF_cox2[i] <- mean(new_data[,(Time <= ti) & Delta == 2])
  i <- i + 1
}
```

Approximating the CIFs using data genereted by RF

Fit RF

```
RF_fit <- randomForestSRC::rfsrc(Surv(Time, Delta) ~ L0 + A0, data = data, nodesize = 150)
```

Simulate 10 large datasets using `simEventRF`

```
N <- 2*10^4
new_data2 <- list()
RFcox_fitss1 <- list()
RFcox_fitss2 <- list()
list_old_vars <- list(L0 = data$L0, A0 = data$A0)
for(i in 1:10){
  new_data2[[i]] <- simEventRF(N, RF_fit, list_old_vars = list_old_vars, term_events = c(1,2))
  RFcox_fitss1[[i]] <- confint(coxph(Surv(Time, Delta == 1) ~ L0 + A0, data = new_data2[[i]]))
  RFcox_fitss2[[i]] <- confint(coxph(Surv(Time, Delta == 2) ~ L0 + A0, data = new_data2[[i]]))
}
new_data2 <- dplyr::bind_rows(new_data2)
```

Approximate the cumulative incidence function using the RF data

```
CIF_RF1 <- numeric(10)
CIF_RF2 <- numeric(10)
i <- 1

for(ti in tis){
  CIF_RF1[i] <- mean(new_data2[,(Time <= ti) & Delta == 1])
  CIF_RF2[i] <- mean(new_data2[,(Time <= ti) & Delta == 2])
  i <- i + 1
}
```

Calculating the Qs

```
Q_cox1 <- sum((CIF_cox1 - CIF_true1)^2)
Q_cox2 <- sum((CIF_cox2 - CIF_true2)^2)
Q_RF1 <- sum((CIF_RF1 - CIF_true1)^2)
Q_RF2 <- sum((CIF_RF2 - CIF_true2)^2)
print(Q_cox1); print(Q_RF1)
```

```
## [1] 0.05892252
```

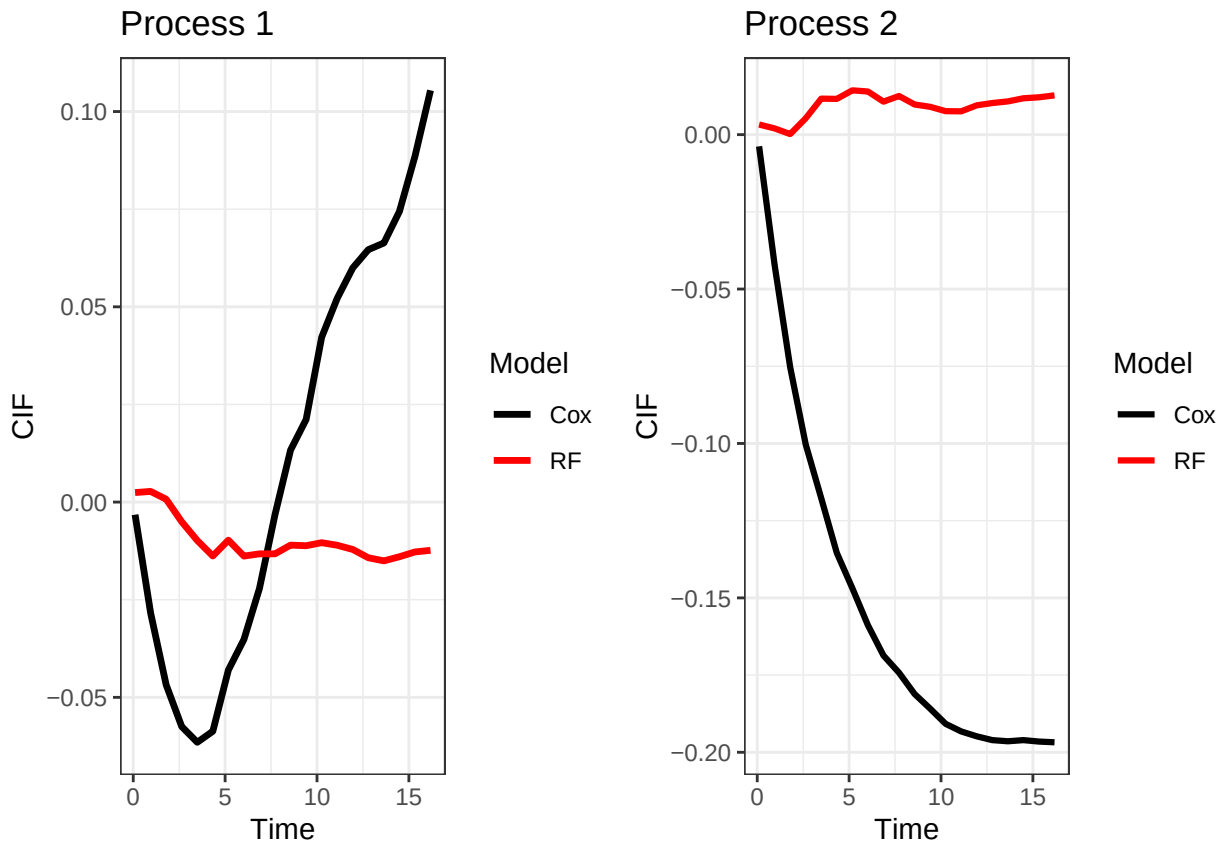
```
## [1] 0.002529797
```

```
print(Q_cox2); print(Q_RF2)
```

```
## [1] 0.5272147
```

```
## [1] 0.002030302
```

```
df1 <- data.frame(
  tis = tis,
  True = CIF_true1,
  Cox = CIF_cox1 - CIF_true1,
  RF = CIF_RF1 - CIF_true1
) %>%
  pivot_longer(cols = c(Cox, RF), names_to = "Model", values_to = "CIF")

df2 <- data.frame(
  tis = tis,
  Cox = CIF_cox2 - CIF_true2,
  RF = CIF_RF2 - CIF_true2
) %>%
  pivot_longer(cols = c(Cox, RF), names_to = "Model", values_to = "CIF")

p1 <- ggplot(df1, aes(x = tis, y = CIF, color = Model)) +
  geom_line(linewidth = 1.2) +
  ylab("CIF") + xlab("Time") +
  theme_bw() +
  ggtitle("Process 1") +
  scale_color_manual(values = c("black", "red", "green4"))

p2 <- ggplot(df2, aes(x = tis, y = CIF, color = Model)) +
  geom_line(linewidth = 1.1) +
  ylab("CIF") + xlab("Time") +
  theme_bw() +
  ggtitle("Process 2") +
  scale_color_manual(values = c("black", "red", "green4"))

gridExtra::grid.arrange(p1,p2, nrow = 1)
```



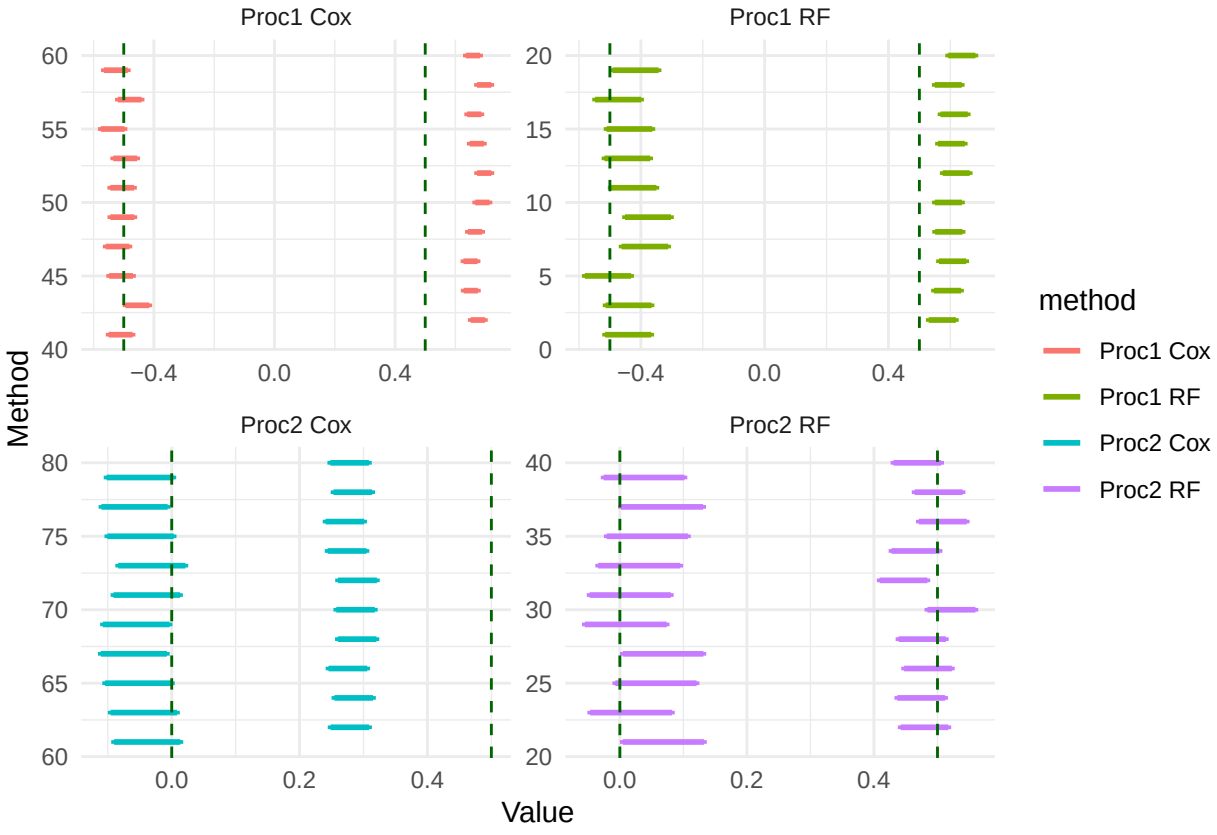
Additionally we have fitted cox models to all generated data and calculated the confidence intervals for the coefficients. This has resulted in the following graph:

```
proc1_RF <- do.call(rbind, RFcox_fitss1)
proc2_RF <- do.call(rbind, RFcox_fitss2)
proc1_cox <- do.call(rbind, cox_fitss1)
proc2_cox <- do.call(rbind, cox_fitss2)
df <- tibble(
  method = rep(c("Proc1 RF", "Proc2 RF", "Proc1 Cox", "Proc2 Cox"),each = 20),
  ci_lower = c(proc1_RF[,1],proc2_RF[,1], proc1_cox[,1], proc2_cox[,1]),
  ci_upper = c(proc1_RF[,2],proc2_RF[,2], proc1_cox[,2], proc2_cox[,2]),
  true_value = rep(c(rep(beta[,2], 10), rep(beta[,3],10)), 2),
  y = 1:80
)

ggplot(df, aes(y = y, color = method)) +
  geom_errorbarh(
    aes(xmin = ci_lower, xmax = ci_upper),
    position = position_dodge(width = 0.5),
    height = 0.25,
    size = 1
  ) +
  geom_vline(
    aes(xintercept = true_value),
    linetype = "dashed",
    color = "darkgreen"
```

```
  ) +
  theme_minimal() +
  labs(
    x = "Value",
    y = "Method"
  )+
  facet_wrap("method", scales = "free")
```

```
## Warning: Using ‘size‘ aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use ‘linewidth‘ instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

```
## Warning: ‘position_dodge()‘ requires non-overlapping x intervals.
## ‘position_dodge()‘ requires non-overlapping x intervals.
## ‘position_dodge()‘ requires non-overlapping x intervals.
## ‘position_dodge()‘ requires non-overlapping x intervals.
```



On Data not from Proportional Hazard Model

We simulate the “observed” data

```
beta <- matrix(c(-0.5 , 0.5, 0, 0, 0.5, -0.5, 0, 0), ncol = 2)
beta_TV <- matrix(c(1 , -1.5, 0, 0, -1, 1, 0, 0), ncol = 2)
t_prime <- 2
data <- simEventTV(3000, beta = beta, term_deltas = c(0,1), tv_eff = beta_TV,
                   t_prime = t_prime)
data[, Delta := Delta + (Delta == 0) * 2]
```

Approximating the true CIFs

We simulate data from the true distribution:

```
N <- 10^5
true_data <- list()
for(i in 1:10){
  true_data[[i]] <- simEventTV(N, beta = beta, term_deltas = c(0,1), tv_eff = beta_TV,
                               t_prime = t_prime)
}
true_data <- dplyr::bind_rows(true_data)
true_data[, Delta := Delta + (Delta == 0) * 2]
```

We approximate the true cumulative incidence function using the simulated data

```
tis <- seq(0.1,5, length.out = 10)
CIF_true1 <- numeric(10)
CIF_true2 <- numeric(10)
i <- 1

for(ti in tis){
  CIF_true1[i] <- mean(true_data[,(Time <= ti) & Delta == 1])
  CIF_true2[i] <- mean(true_data[,(Time <= ti) & Delta == 2])
  i <- i + 1
}
```

Approximating the CIFs using data genereted by Cox

Fit Cox models

```
cox1 <- coxph(Surv(Time, Delta == 1) ~ L0 + A0, data = data)
cox2 <- coxph(Surv(Time, Delta == 2) ~ L0 + A0, data = data)
cox_fits <- list("Process 1" = cox1, "Process 2" = cox2)
```

Simulate 10 large datasets using `simEventCox`

```
N <- 10^5
new_data <- list()
for(i in 1:10){
  new_data[[i]] <- simEventCox(N, cox_fits, L0_old = data$L0, A0_old = data$A0)
}
new_data <- dplyr::bind_rows(new_data)
```

Approximate the cumulative incidence function using the Cox data

```
CIF_cox1 <- numeric(10)
CIF_cox2 <- numeric(10)
i <- 1

for(ti in tis){
  CIF_cox1[i] <- mean(new_data[,(Time <= ti) & Delta == 1])
  CIF_cox2[i] <- mean(new_data[,(Time <= ti) & Delta == 2])
  i <- i + 1
}
```

Approximating the CIFs using data genereted by Cox

Fit RF

```
RF_fit <- randomForestSRC::rfsrc(Surv(Time, Delta) ~ L0 + A0, data = data, nodesize = 150)
list_old_vars <- list(L0 = data$L0, A0 = data$A0)
```

Simulate 10 large datasets using `simEventRF`

```
N <- 1.5*10^4
new_data2 <- list()
for(i in 1:10){
  print(i)
  new_data2[[i]] <- simEventRF(N = N, RF_fit = RF_fit, list_old_vars = list_old_vars, term_events = c(1
}
```

```
## [1] 1
## [1] 2
## [1] 3
## [1] 4
## [1] 5
## [1] 6
## [1] 7
## [1] 8
## [1] 9
## [1] 10
```

```
new_data2 <- dplyr::bind_rows(new_data2)
```

Approximate the cumulative incidence function using the RF data

```
CIF_RF1 <- numeric(10)
CIF_RF2 <- numeric(10)
i <- 1

for(ti in tis){
  CIF_RF1[i] <- mean(new_data2[,(Time <= ti) & Delta == 1])
  CIF_RF2[i] <- mean(new_data2[,(Time <= ti) & Delta == 2])
  i <- i + 1
}
```

Calculating the Qs

```r
Q_cox1 <- sum((CIF_cox1 - CIF_true1)^2)
Q_cox2 <- sum((CIF_cox2 - CIF_true2)^2)
Q_RF1 <- sum((CIF_RF1 - CIF_true1)^2)
Q_RF2 <- sum((CIF_RF2 - CIF_true2)^2)
print(Q_cox1); print(Q_RF1)
```

```
## [1] 0.01277755
```

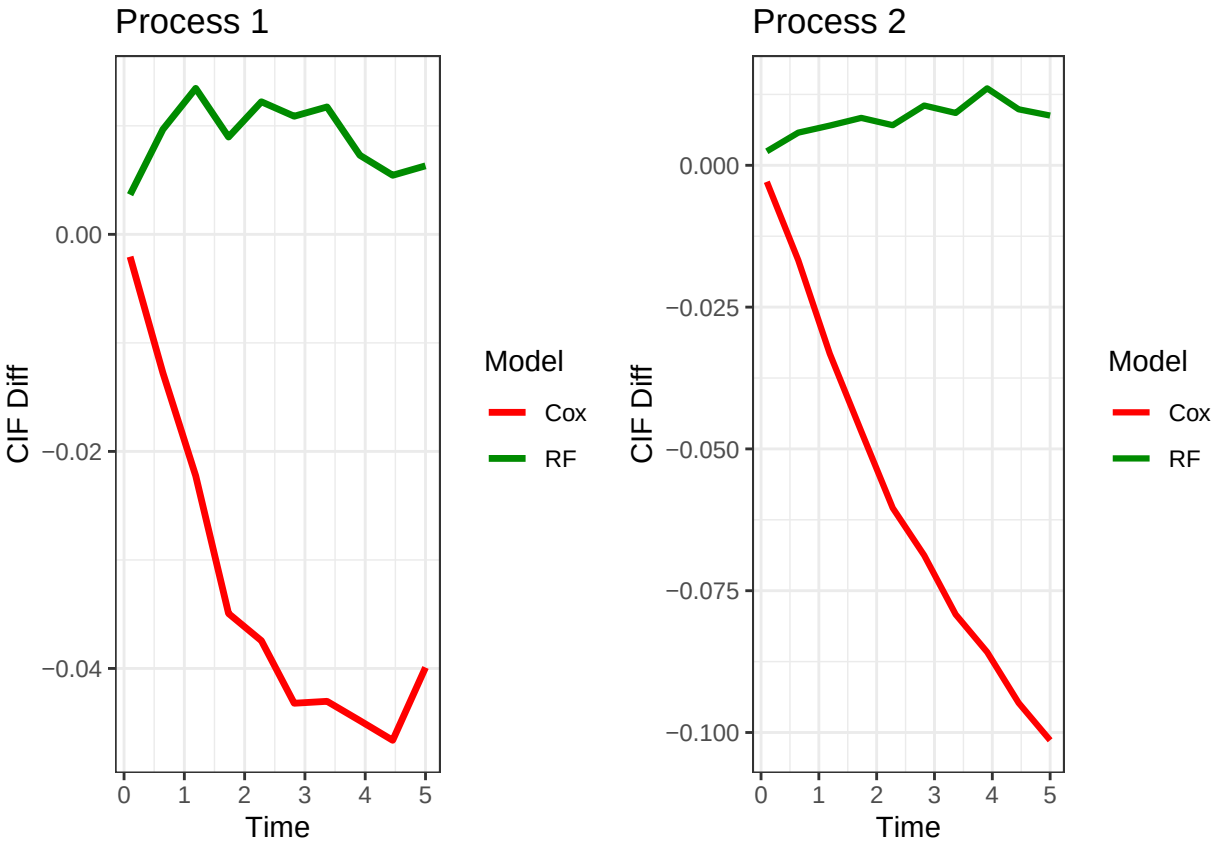
```
## [1] 0.0008964046
```

```r
print(Q_cox2); print(Q_RF2)
```

```
## [1] 0.04486901
```

```
## [1] 0.000763182
```

```r
df1 <- data.frame(
  tis = tis,
  True = CIF_true1,
  Cox = CIF_cox1 - CIF_true1,
  RF = CIF_RF1 - CIF_true1
) %>%
  pivot_longer(cols = c(Cox, RF), names_to = "Model", values_to = "CIF")

df2 <- data.frame(
  tis = tis,
  Cox = CIF_cox2 - CIF_true2,
  RF = CIF_RF2 - CIF_true2
) %>%
  pivot_longer(cols = c(Cox, RF), names_to = "Model", values_to = "CIF")

p1 <- ggplot(df1, aes(x = tis, y = CIF, color = Model)) +
  geom_line(linewidth = 1.2) +
  ylab("CIF Diff") + xlab("Time") +
  theme_bw() +
  ggtitle("Process 1") +
  scale_color_manual(values = c("red", "green4"))

p2 <- ggplot(df2, aes(x = tis, y = CIF, color = Model)) +
  geom_line(linewidth = 1.1) +
  ylab("CIF Diff") + xlab("Time") +
  theme_bw() +
  ggtitle("Process 2") +
  scale_color_manual(values = c( "red", "green4"))

gridExtra::grid.arrange(p1,p2, nrow = 1)
```


Process 1
CIF Diff
0.00
−0.02
−0.04
0
1
2
3
4
5
Time
Model
Cox
RF
Process 2
CIF Diff
0.000
−0.025
−0.050
−0.075
−0.100
0
1
2
3
4
5
Time
Model
Cox
RF