Homework2

Jieying Jiao

2018-09-13

Contents

Chapter 1

Exercise 1.2

1.1 Use Monte Carlo to estimate $\Phi(t)$

The Monte Carlo methods gives:

$$\hat{\Phi}(t) = \frac{1}{n}\sum_{i=1}^{n} I(X_i \leqslant t)$$

where X_i are random normal sample drawn from $N(0,1)$.

```
library(xtable)
phi.MC <- function(n, t) {
  phi.hat <- matrix(0, nrow = length(n), ncol = length(t))
  for (i in 1:length(n)) {
    for (j in 1:length(t)) {
      X <- rnorm(n[i], 0, 1)
      phi.hat[i, j] <- mean(as.numeric(X <= t[j]))
    }
  }
truth <- pnorm(t)
phi.hat <- rbind(phi.hat, truth)
}
n <- c(10^2, 10^3, 10^4)
t <- c(0.0, 0.67, 0.84, 1.28, 1.65, 2.32, 2.58, 3.09, 3.72)
set.seed(1)
phi.hat <- phi.MC(n, t)
rownames(phi.hat) <- c(paste0("n = 10e", 2:4), "truth")
colnames(phi.hat) <- paste0("t = ", t)
t.MC <- xtable(phi.hat, digits = 4, caption = "Monte Carlo estimation",
               label = "MC result")
```

The estimation results are shown in Table 1.1.

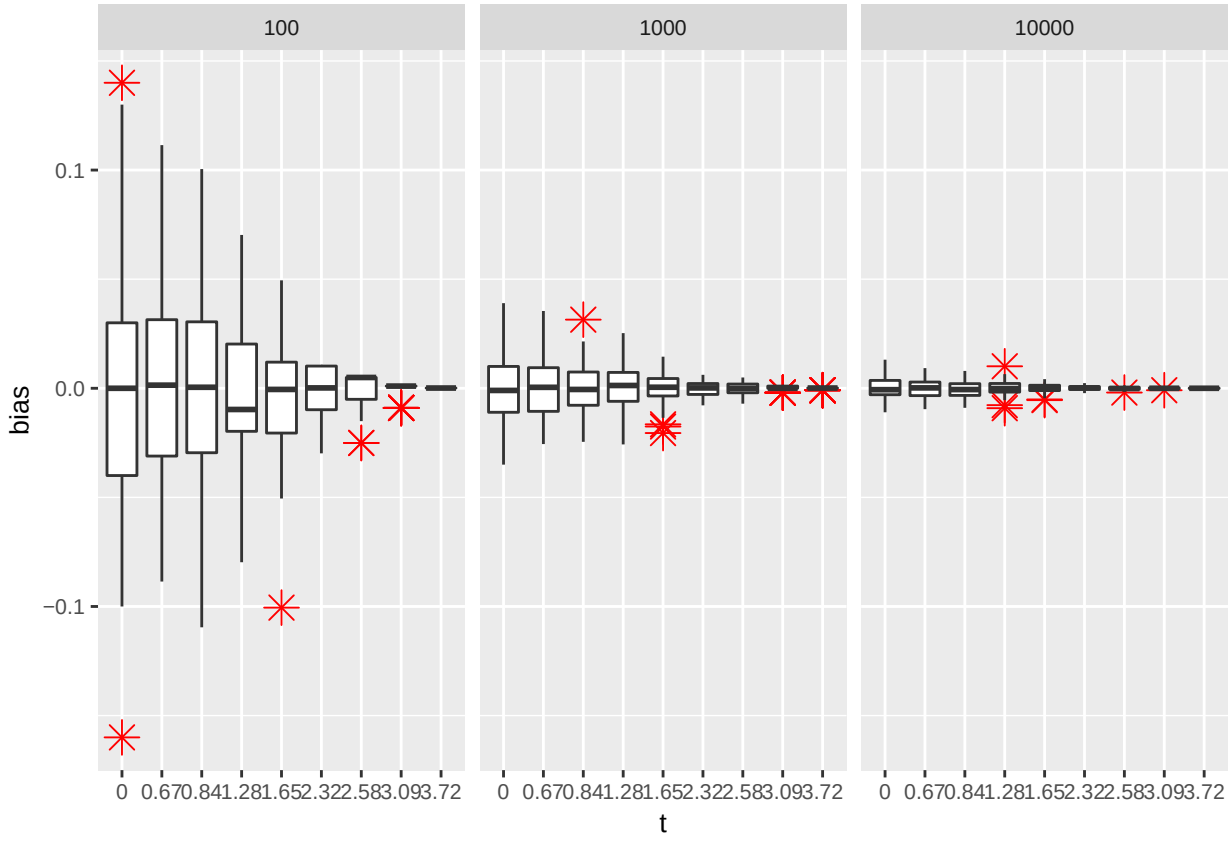
1.2 Boxplots of bias

Repeat the above expriment 100 times and plot bias of Monte Carlo methods at every time point t using boxplots.

	t = 0	t = 0.67	t = 0.84	t = 1.28	t = 1.65	t = 2.32	t = 2.58	t = 3.09	t = 3.72
n = 10e2	0.4600	0.7700	0.7800	0.8700	0.9200	1.0000	0.9900	1.0000	1.0000
n = 10e3	0.5110	0.7370	0.7710	0.9080	0.9500	0.9860	0.9990	0.9980	1.0000
n = 10e4	0.5061	0.7467	0.8021	0.8981	0.9527	0.9913	0.9947	0.9985	0.9998
truth	0.5000	0.7486	0.7995	0.8997	0.9505	0.9898	0.9951	0.9990	0.9999

Table 1.1: Monte Carlo estimation

```
library(ggplot2)
nsim <- 100
bias <- data.frame(bias = rep(0, nsim * length(t) * length(n)),
                   t = rep(t, length(n) * nsim),
                   n = rep(rep(n, each = length(t)), nsim))
for (i in 1:nsim) {
  phi.hat <- phi.MC(n, t)
  bias$bias[((i-1)*length(t)*length(n)+1):(i*length(t)*length(n))] <-
    as.vector(t(phi.hat[1:length(n), ]) - phi.hat[length(n) + 1, ])
}
bias$t <- as.factor(bias$t)
ggplot(data = bias, aes(x = t, y = bias)) +
  geom_boxplot(outlier.colour="red", outlier.shape=8, outlier.size=4) +
  facet_wrap( ~ n) + theme(text = element_text(size = 10))
```



Chapter 2

Exercise 1.3

```
.Machine$double.xmax
```

```
## [1] 1.797693e+308
```

```
.Machine$double.xmin
```

```
## [1] 2.225074e-308
```

```
.Machine$double.eps
```

```
## [1] 2.220446e-16
```

```
.Machine$double.neg.eps
```

```
## [1] 1.110223e-16
```

Floating number is represented in computer as :

$$(-1)^{x_0}(1+\sum_{i=1}^{t} x_i 2^{-i})2^k$$

In a 64 bits machine, x_0 takes 1 sign bites, significant takes 52 bits, so t can be 52 at most. Exponent k takes 11 bits, so $2^11 = 2048$ possible values. With shifting to negative side, k can be from -1022 to 1023, with both -1023 (all zeros) amd 1024 (all ones) are left for special numbers.

".Machine$double.xmax" is the largest floating number that computer can display, it is $(1+\sum_{i=1}^{52} 2^{-i}) \times 2^{1023}$

".Machine$double.xmin" is the smallest floating numer that computer can display, it is 2^{-1022}.

".Machine$double.eps" is the smallest positive floating number that the computer can tell the difference by adding it. It is actually the smallest significant, that is 2^{-52}.

".Machine$double.neg.eps" is the smallest positive floating number that the computer can tell the difference by substracting it. It is 2^{-53}.