# Use the Monte Carlo method to approximate $N(0,1)$

*Xiaokang Liu*

*13 September 2018*

**Abstract**

This report aims to record a small experiment to use the Monte Carlo method to approximate the cumulative distribution function of the standard normal distribution. The results will be displayed in tables and graphs.

## Contents



## 1 Introduction

Consider approximation of the distribution function of $N(0,1)$,

$$\Psi(t) = \int_{-\infty}^{t} \frac{1}{\sqrt{2\pi}} e^{-y^2/2} dy, \tag{1}$$

by

$$\widehat{\Psi}(t) = \frac{1}{n} \sum_{i=1}^{n} I(X_i \leq t), \tag{2}$$

where $X_i$'s are i.i.d. $N(0,1)$ variables. Experiments with the approximation at $n \in \{10^2, 10^3, 10^4\}$ at $t \in \{0, 0.67, 0.84, 1.28, 1.65, 2.32, 2.58, 3.09, 3.72\}$ will be displayed by a table. The experiment will be repeated for 100 times. The bias at all $t$ will be showed in some boxplots.

## 2 Implementation and Results

### 2.1 R codes of conducting experiments

```
n <- c(100, 1000, 10000)
t <- c(0,0.67,0.84,1.28,1.65,2.32,2.58,3.09,3.72)
results <- array(dim = c(100, 3, 9))
for (i in 1:100){
  for (j in 1:3){
    for (k in 1:9){
```

Table 1: Summary of the experiment(part 1)

| | 0.0 | 0.67 | 0.84 | 1.28 | 1.65 |
|---|---|---|---|---|---|
| true | 0.500000 | 0.7485711 | 0.7995458 | 0.8997274 | 0.9505285 |
| n=100 | 0.499900 | 0.7478000 | 0.7999000 | 0.8989000 | 0.9512000 |
| n=1000 | 0.500430 | 0.7498600 | 0.7996200 | 0.9009600 | 0.9494300 |
| n=10000 | 0.500402 | 0.7481770 | 0.7992960 | 0.8995590 | 0.9506210 |

```
      results[i,j,k] <- sum(rnorm(n[j])<=t[k])/n[j]
    }
  }
}

# summarize the results from 100 repeatations
sum.results <- matrix(nrow = 4, ncol = 9)
sum.results[1,] <- pnorm(t)
sum.results[2,] <- apply(results[,1,], 2, mean)
sum.results[3,] <- apply(results[,2,], 2, mean)
sum.results[4,] <- apply(results[,3,], 2, mean)
colnames(sum.results) <- c("0.0","0.67","0.84","1.28","1.65","2.32","2.58",
                           "3.09","3.72")
rownames(sum.results) <- c("true","n=100","n=1000","n=10000")
```

## 2.2 Results

### 2.2.1 Tables of mean estimation values

The following two tables including the results averaged from 100 repetations for each situation. By comparing the results of the 2, 3 and 4 row with the 1st row, we can find that larger the sample size, smaller the difference between the approximated probability and the true probability.

```
# table
knitr::kable(sum.results[,c(1:5)], booktabs = TRUE,
             caption = 'Summary of the experiment(part 1)')
```

```
knitr::kable(sum.results[,-c(1:5)], booktabs = TRUE,
             caption = 'Summary of the experiment(part 2)')
```

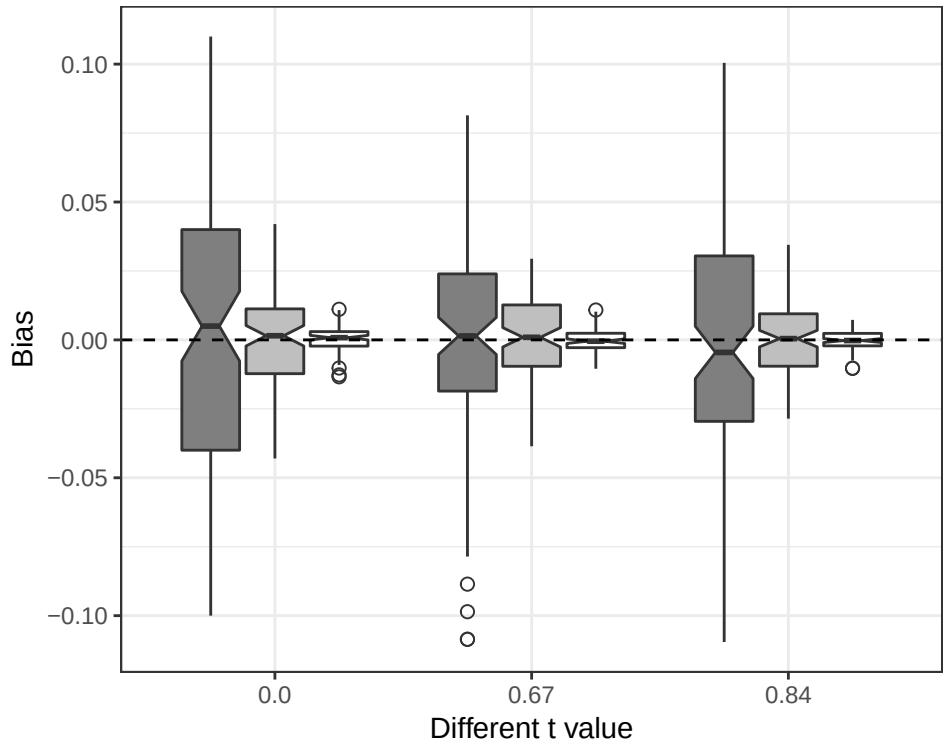
### 2.2.2 Box plots of bias at all $t$

```
# get the bias
bias <- array(dim = c(100, 3, 9))
truep <- t(matrix(rep(pnorm(t),3), nrow = 9, ncol = 3))
for (i in 1:100){
```

Table 2: Summary of the experiment(part 2)

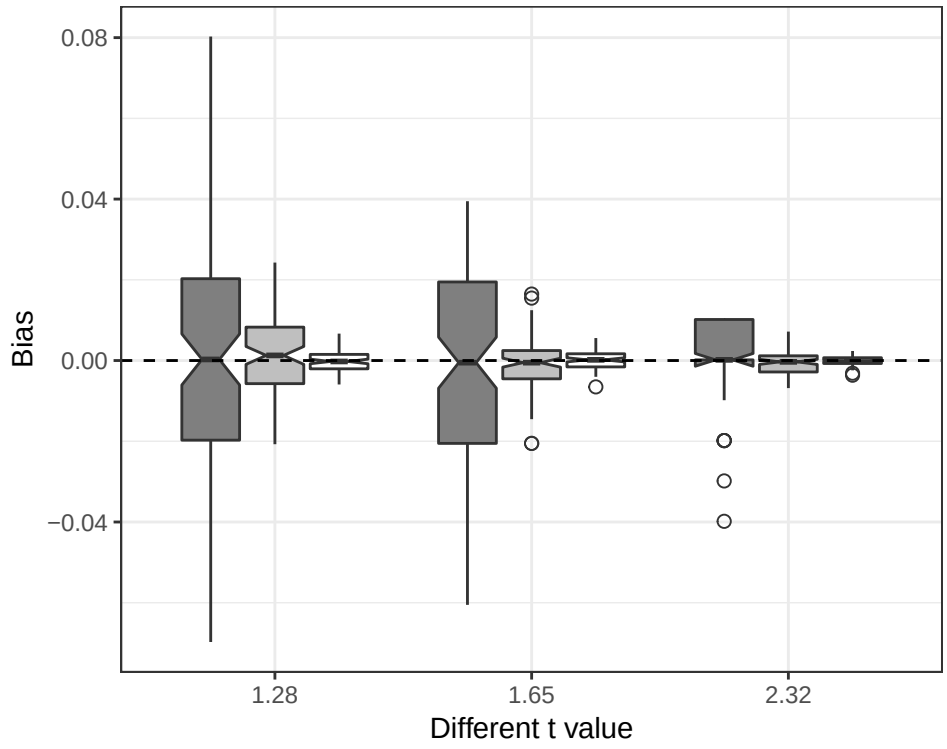
| | 2.32 | 2.58 | 3.09 | 3.72 |
|---|---|---|---|---|
| true | 0.9898296 | 0.995060 | 0.9989992 | 0.9999004 |
| n=100 | 0.9911000 | 0.995600 | 0.9986000 | 0.9998000 |
| n=1000 | 0.9892700 | 0.995160 | 0.9989800 | 0.9999300 |
| n=10000 | 0.9897490 | 0.995026 | 0.9990450 | 0.9998950 |

```
  bias[i,,] <- results[i,,]-truep
}

####### for t=0.0, 0.67, 0.84
prg1 <- vector()
for(a in 1:3){
  for (b in 1:3){
    prg1 <- c(prg1,sort(bias[,a,b]))
  }
}

nprg <- 100
f1 <- rep(c(rep("0.0",nprg),rep("0.67",nprg),rep("0.84",nprg)),3)
f2 <- c(rep(100,nprg*3),rep(1000,nprg*3),rep(10000,nprg*3))

prgdata1 <- data.frame(b=factor(f1),
                       Correlation=factor(f2),
                       PRG=prg1,geom="point")

#postscript(paste("1to3.eps",sep=""), width = 4, height = 4,horizontal=FALSE)
ggplot(aes(y = PRG, x = b, fill = Correlation), data = prgdata1) +
  geom_boxplot(notch=TRUE,notchwidth=0.3,outlier.size=2,outlier.shape=1) +
  scale_fill_manual(name = "Correlation",
                    values = c("grey50", "grey75","white"))+
  ylab("Bias") +
  xlab("Different t value")+
  theme_bw()+
  guides(fill=FALSE)+
  geom_hline(aes(yintercept=0), colour="black", linetype="dashed")
```

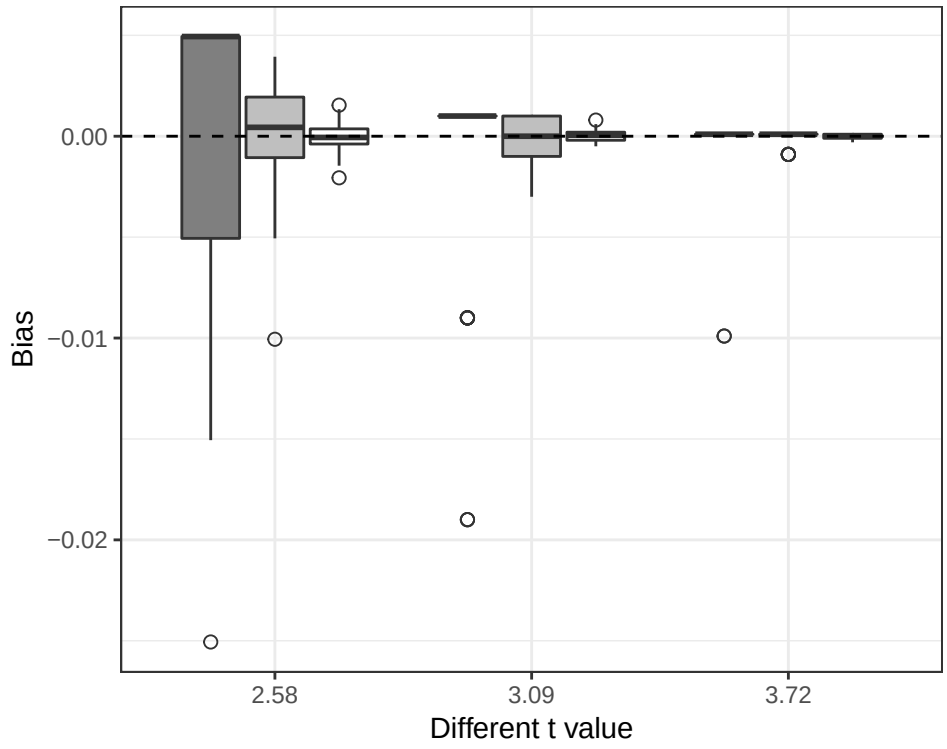



```
###### for t=1.28, 1.65, 2.32
prg2 <- vector()
for(a in 1:3){
  for (b in 4:6){
    prg2 <- c(prg2,sort(bias[,a,b]))
  }
}

f3 <- rep(c(rep("1.28",nprg),rep("1.65",nprg),rep("2.32",nprg)),3)
prgdata2 <- data.frame(b=factor(f3),
                       Correlation=factor(f2),
                       PRG=prg2,geom="point")

#postscript(paste("1to3.eps",sep=""), width = 4, height = 4,horizontal=FALSE
ggplot(aes(y = PRG, x = b, fill = Correlation), data = prgdata2) +
  geom_boxplot(notch=TRUE,notchwidth=0.3,outlier.size=2,outlier.shape=1) +
  scale_fill_manual(name = "Correlation",
                    values = c("grey50", "grey75","white"))+
  ylab("Bias") +
  xlab("Different t value")+
  theme_bw()+
  guides(fill=FALSE)+
  geom_hline(aes(yintercept=0), colour="black", linetype="dashed")
```

```
## notch went outside hinges. Try setting notch=FALSE.
```




```
##### for t=2.58, 3.09, 3.72
prg3 <- vector()
for(a in 1:3){
  for (b in 7:9){
    prg3 <- c(prg3,sort(bias[,a,b]))
  }
}
f4 <- rep(c(rep("2.58",nprg),rep("3.09",nprg),rep("3.72",nprg)),3)
prgdata3 <- data.frame(b=factor(f4),
                       Correlation=factor(f2),
                       PRG=prg3,geom="point")

#postscript(paste("1to3.eps",sep=""), width = 4, height = 4,horizontal=FALSE)
ggplot(aes(y = PRG, x = b, fill = Correlation), data = prgdata3) +
  geom_boxplot(notch=FALSE,notchwidth=0.3,outlier.size=2,outlier.shape=1) +
  scale_fill_manual(name = "Correlation",
                    values = c("grey50", "grey75","white"))+
  ylab("Bias") +
  xlab("Different t value")+
  theme_bw()+
  guides(fill=FALSE)+
  geom_hline(aes(yintercept=0), colour="black", linetype="dashed")
```




```
dev.off()
```

```
## null device
##           1
```