HW2

Qi Qi

9/14/2018

Exercise 1

Approximation of distribution function of standard normal

Abstract

Monte Carlo methods have been used to approximate the distribution function of standard normal. Box plots of bias are also established. After the comparison, I conclude that when sample size is enlarged, the approximations will get close to the true values.

Method

The distribution function of $N(0,1)$ is

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

I compute the true values at $t \in \{0.0, 0.67, 0.84, 1.28, 1.65, 2.32, 2.58, 3.09, 3.72\}$.

To calculate the approximated values, I generate random samples with sample size $n \in \{10^2, 10^3, 10^4\}$ and evaluate the approximation at all values of t by Monte Carlo method:

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

Comparison of true values and approximations

I use `pnorm` function to calculate the true values of distribution function at all the values of t and then compute the approximations by Monte Carlo method described previously. Table 1 is generated and contains t, true values and approximations at different sample size n.

```r
t<- c(0.0, 0.67, 0.84, 1.28, 1.65, 2.32, 2.58, 3.09, 3.72)
true_value<- pnorm(t, 0, 1)
apprx1<- NA
apprx2<- NA
apprx3<- NA
# n=10^2
x<- rnorm(100, 0, 1)
for (i in 1:length(t)){
  apprx1[i]<- sum(x<=t[i])/100
}

# n=10^3
```

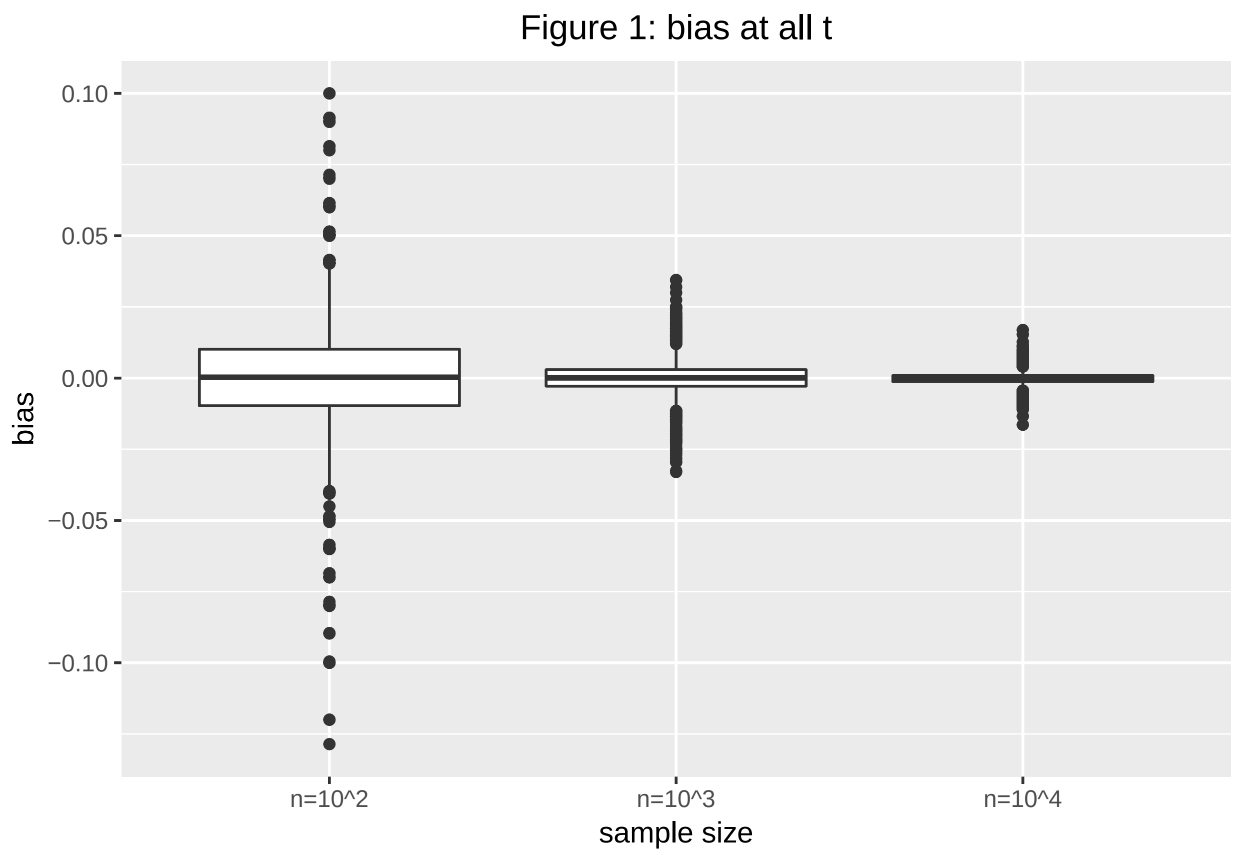
```r
y<- rnorm(1000, 0, 1)
for (i in 1:length(t)){
  apprx2[i]<- sum(y<=t[i])/1000
}

# n=10^4
z<- rnorm(10000, 0, 1)
for (i in 1:length(t)){
  apprx3[i]<- sum(z<=t[i])/10000
}

tab<- cbind(t, true_value, apprx1, apprx2, apprx3)
knitr::kable(tab, col.names = c("t", "True value", "Approx at n=10^2", "Approx at n=10^3", "Approx at n=
```

Table 1: Comparison of true values and approximation

t	True value	Approx at n=10^2	Approx at n=10^3	Approx at n=10^4
0.00	0.5000000	0.56	0.508	0.4954
0.67	0.7485711	0.75	0.761	0.7468
0.84	0.7995458	0.77	0.800	0.7994
1.28	0.8997274	0.90	0.902	0.8958
1.65	0.9505285	0.94	0.947	0.9505
2.32	0.9898296	0.98	0.997	0.9901
2.58	0.9950600	1.00	0.998	0.9954
3.09	0.9989992	1.00	1.000	0.9991
3.72	0.9999004	1.00	1.000	1.0000

Further, I repeat the experiment 100 times and create box plots of bias at all t (Figure 1).

```r
library(ggplot2)
bias1<- NA
bias2<- NA
bias3<- NA
bias_all<- NA
for (j in 1:100){
  for (i in 1:length(t)){
     x<- rnorm(100, 0, 1)
     bias1[i]<- sum(x<=t[i])/100- true_value[i]
     y<- rnorm(1000, 0, 1)
     bias2[i]<- sum(y<=t[i])/1000- true_value[i]
     z<- rnorm(10000, 0, 1)
     bias3[i]<- sum(z<=t[i])/10000- true_value[i]
  }
  bias<-cbind(bias1, bias2, bias3)
  bias_all<-rbind(bias_all, bias)
}
bias_all<- as.data.frame(na.omit(bias_all))
ggplot(stack(bias_all), aes(x = ind, y = values))+
  geom_boxplot()+
  scale_x_discrete(labels=c("n=10^2", "n=10^3", "n=10^4"), name="sample size")+
  scale_y_continuous(name="bias")+
  ggtitle("Figure 1: bias at all t")+
  theme(plot.title = element_text(hjust = 0.5))
```



Conclusion

According to Table 1, we can notice that when $n = 10^4$ the approximations are most close to the true values. After repeating experiment 100 times, we can state the bias are more close to 0 with larger n from Figue 1. Thus, this indicate that enlarging sample size will reduce the error of approximation.

Exercise 2

`.Machine$double.xmax` is the largest normalized floating-point number. Normally, it is $1.797693e + 308$.

`.Machine$double.xmin` is the smallest non-zero normalized floating-point number. Normally, it is $2.225074e - 308$.

`.Machine$double.eps` is the smallest positive floating-point number x such that $1 + x! = 1$. Normally, it is $2.220446e - 16$.

`.Machine$double.neg.eps` is a small positive floating-point number x such that $1 - x! = 1$. Normally, it is $1.110223e - 16$.