Solutions to computer class 1

Richard Wilkinson

16 February 2018

Question 1

Suppose $X_1, \dots, X_{10}$ are iid $U[0,1]$ random variables. Find the distribution of

$$R(X) = \max_i \{X_i\} - \min_i \{X_i\}.$$

What is $\mathbb{P}(R(X) > 0.99)$?

Solution

Finding the distribution in a non-parametric setting essentially means generating a histogram in order to estimate the pdf. We can easily do this with Monte Carlo sampling. Let's start by creating a function to simulate R.

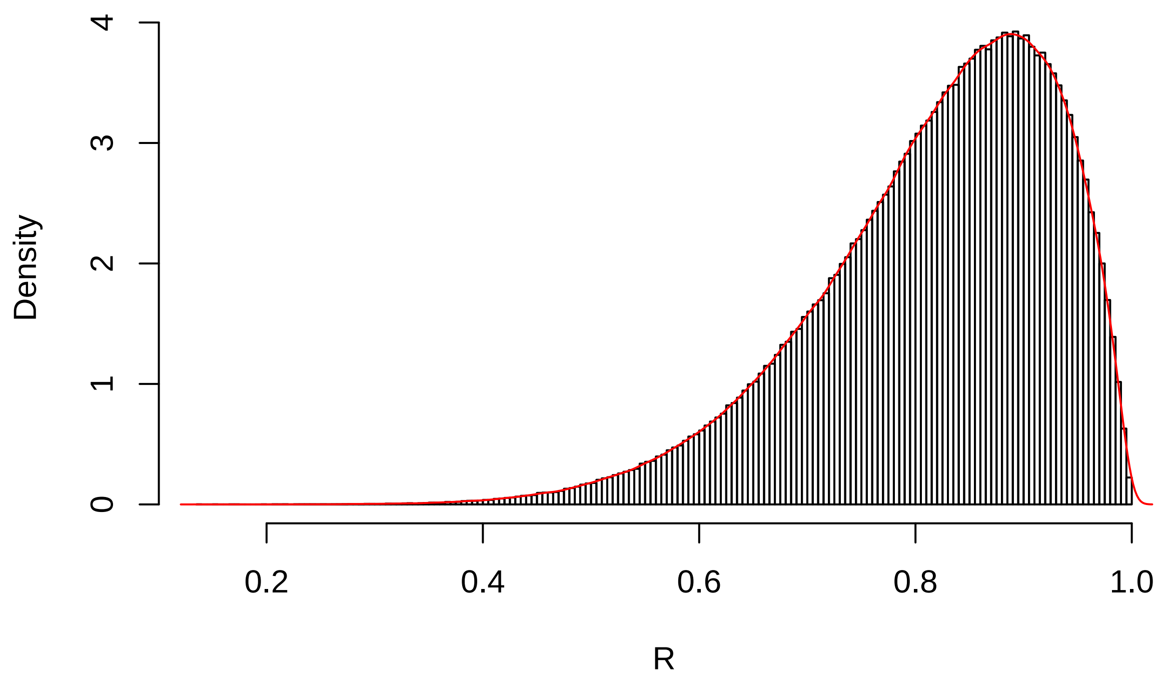
```
simR <- function(){
  X <- runif(10) # simulate 10 U[0,1] rvs
  R <- max(X) - min(X)
  return(R)
}
```

Now let's simulate 10^6 realisations of R

```
Rvals <- replicate(10^6, simR())
```

We now just need to find the histogram of these values. I've drawn the kernel density estimate of the pdf over the top.

```
library(MASS)
truehist(Rvals, col=0, xlab = 'R', ylab='Density')
lines(density(Rvals), col=2)
```



To find $\mathbb{P}(R(X) > 0.99)$ we just count the proportion of the samples that are above 0.99.

```
sum(Rvals>0.99)/10^6
```

```
## [1] 0.004259
```

Note that this is quite a small number, thus to estimate it well it was necessary to take a very large number of samples of R.

Question 2

Estimate the integral

$$I = \int_0^{10} \frac{1}{(1+x^2)} dx$$

using at least two different choices for the proposal density $g(\cdot)$.

Find 95% confidence intervals for your estimates using the central limit theorem.

Solution

We could do this in several different ways. They all amount to choosing the importance function $g(\cdot)$ and using the identity

$$I = \int_0^{10} h(x)g(x)dx \text{ where } h(x) = \frac{f(x)}{g(x)}$$

and then approximating I by

$$\hat{I} = \frac{1}{n}\sum_{i=1}^{n} h(X_i) \text{ where } X_i \sim g(\cdot)$$

Let's start by letting $g(x)$ be the pdf of a $U[0, 10]$ random variable. Then, $h(x) = \frac{10}{1+x^2}$, and we can estimate this in R using

```r
X <- runif(10^6, 0, 10)
hX <- 10/(1+X^2)
mean(hX)
```

```
## [1] 1.470506
```

Lets compare this with a numerical estimate

```r
integrate(function(x) 1/(1+x^2), lower=0, upper=10)
```

```
## 1.471128 with absolute error < 5.1e-06
```

An alternative choice for g would be to let it be the Cauchy distribution

$$g(x) = \frac{1}{\pi(1+x^2)} \text{ for } x \in \mathbb{R}$$

Then

$$h(x) = \pi \mathbb{I}_{x\in[0,10]}$$

```r
X <- rcauchy(10^6)
hX <- pi* (0<X & X<10)
mean(hX)
```

```
## [1] 1.470677
```

Better still, we could let

$$g(x) = \frac{2}{\pi(1+x^2)} \text{ for } x \in \mathbb{R}^+$$

```r
X <- abs(rcauchy(10^6))
hX <- pi* (0<X & X<10)/2
mean(hX)
```

```
## [1] 1.471191
```

We'll find a 95% CI for this last integral. We know from the CLT that

$$\frac{1}{n}\sum h(X_i) \sim N(I, \sigma^2/n) \text{ where } \sigma^2 = \mathbb{V}\text{ar}h(X_1)$$

we can estimate $\mathbb{V}\text{ar}h(X_1)$ as

```r
var(hX)
```

```
## [1] 0.1465392
```

giving a 95% CI of

```r
mean(hX) + qnorm(0.025)*sqrt(var(hX)/10^6)
```

```
## [1] 1.47044
```

```r
mean(hX) + qnorm(0.975)*sqrt(var(hX)/10^6)
```

```
## [1] 1.471941
```

which as it happens does contain the true value of the integral. Note that we could have calculate a 95% CI using Monte Carlo sampling, although this begins to get numerically expensive.

```r
Ihat <- function(){
  X <- abs(rcauchy(10^6))
  hX <- pi* (0<X & X<10)/2
  return(mean(hX))
}
Ihat.vals <- replicate(100, Ihat())
quantile(Ihat.vals, 0.025)
```

```
##     2.5%
## 1.470432
```

```r
quantile(Ihat.vals, 0.975)
```

```
##    97.5%
## 1.471895
```

Question 3

Let

$$I_1 = \int_0^1 e^{-x^2} \mathrm{d}x \quad \text{and} \quad I_2 = \int_0^1 (\cos(50x) + \sin(20x))^2 \mathrm{d}x$$

Estimate both of these integrals using Monte Carlo (using an estimator of your choice). Show that the root mean square error of your estimator scales as $O(n^{-1/2})$. To do this, you will need to repeat the analysis multiple times for a range of values of n (i.e., for $n = 10, 50, 100, 500, 1000, 5000, 10000$ estimate the integral 100 times and calculate the standard error).

Calculate these integrals using the mid-ordinate rule. Show that the error now scales as $O(n^{-2})$.

Solution

We can estimate both using random samples $X \sim U[0,1]$. I'll use a large number of samples here to get a precise estimate we can use as the truth.

```r
f1 <- function(x){exp(-x^2)}
f2 <- function(x){
  (cos(50*x)+sin(20*x))^2
}

X <- runif(10^7)
(I1hat <- (pnorm(1,0,1/sqrt(2))-pnorm(0,0,1/sqrt(2)))*sqrt(pi))
```

```
## [1] 0.7468241
```

```r
(I2hat <- mean(f2(X)))
```

```
## [1] 0.9645358
```

We can estimate the error in our estimator by repeating it.

```r
n = 100
errs <- replicate(10^2, {
  X <- runif(n)
  error <- abs(mean(f1(X))-I1hat)
  })
```

```
sqrt(mean(errs^2))
```

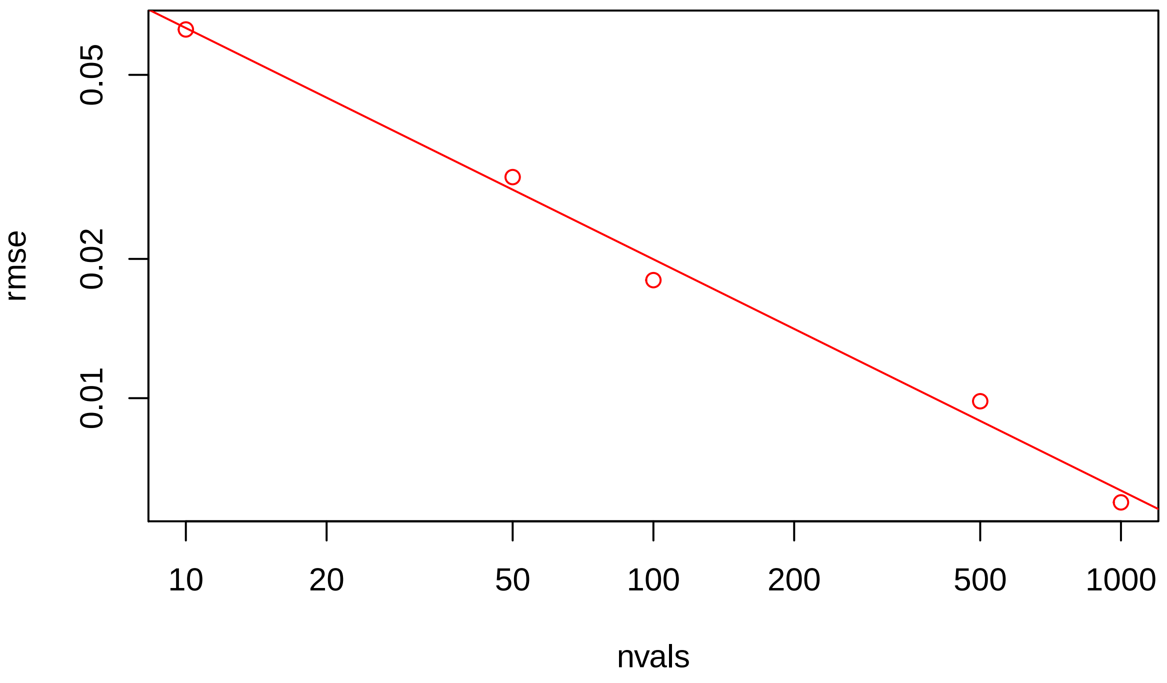
```
## [1] 0.02096357
```

To show how the RMSE scales, lets try for a variety of different values. I've plotted these here on a log-log plot. The line is the theoretical line showing how the RMSE should decay with n, i.e., with the slope set to -1/2. The intercept is the variance of $exp(-X^2)$.

```
nvals <- c(10,50,100,500,1000)
rmse <- c()
i <- 1
for(n in nvals){
  errs <- replicate(10^2, {
    X <- runif(n)
    error <- abs(mean(f1(X))-I1hat)
    })

  rmse[i] <- sqrt(mean(errs^2))
  i <- i+1
}

plot(nvals, rmse, log='xy', col=2)
lm(log10(abs(rmse))~ log10(nvals))
```

```
##
## Call:
## lm(formula = log10(abs(rmse)) ~ log10(nvals))
##
## Coefficients:
##  (Intercept)  log10(nvals)
##      -0.7001       -0.5002
```

```
abline(log10(sd(f1(runif(1000)))), -0.5, col=2)
```



Let's now look at the mid-ordinate rule.

```r
n = 10
h = 1/n
x = (1:n-0.5)*h
sum(f1(x)*h)
```

```
## [1] 0.7471309
```

```r
sum(f2(x)*h)
```

```
## [1] 0.8980558
```

We can also see how this error scales with n.

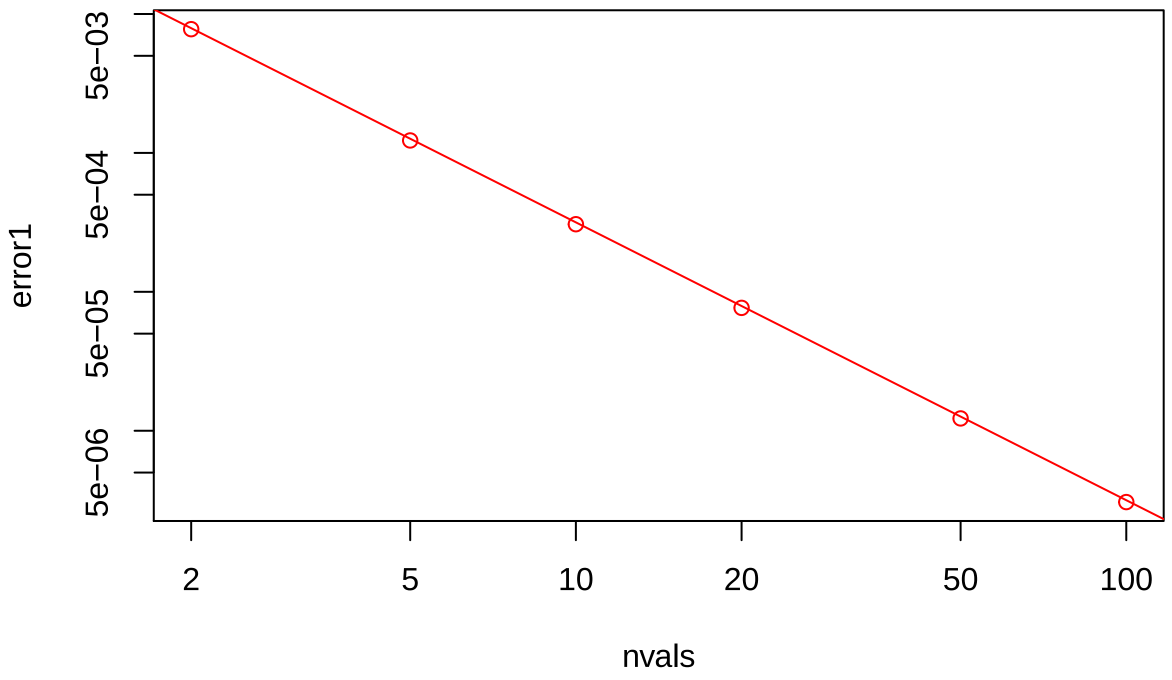
```r
nvals <- c(2,5,10,20,50,100)
rmse <- c()
i <- 1
error1 <- c()

for(n in nvals){
  h = 1/n
  x = (1:n-0.5)*h
  error1[i] <- abs(I1hat - sum(f1(x)*h))
   i <- i+1
}
```

Lets plot these errors

```r
plot(nvals, error1, log='xy', col=2)
lm(log10(error1)~log10(nvals))
```

```
##
## Call:
## lm(formula = log10(error1) ~ log10(nvals))
##
## Coefficients:
##  (Intercept)  log10(nvals)
##       -1.509        -2.003
```

```r
abline(-1.5,-2,col=2)
```

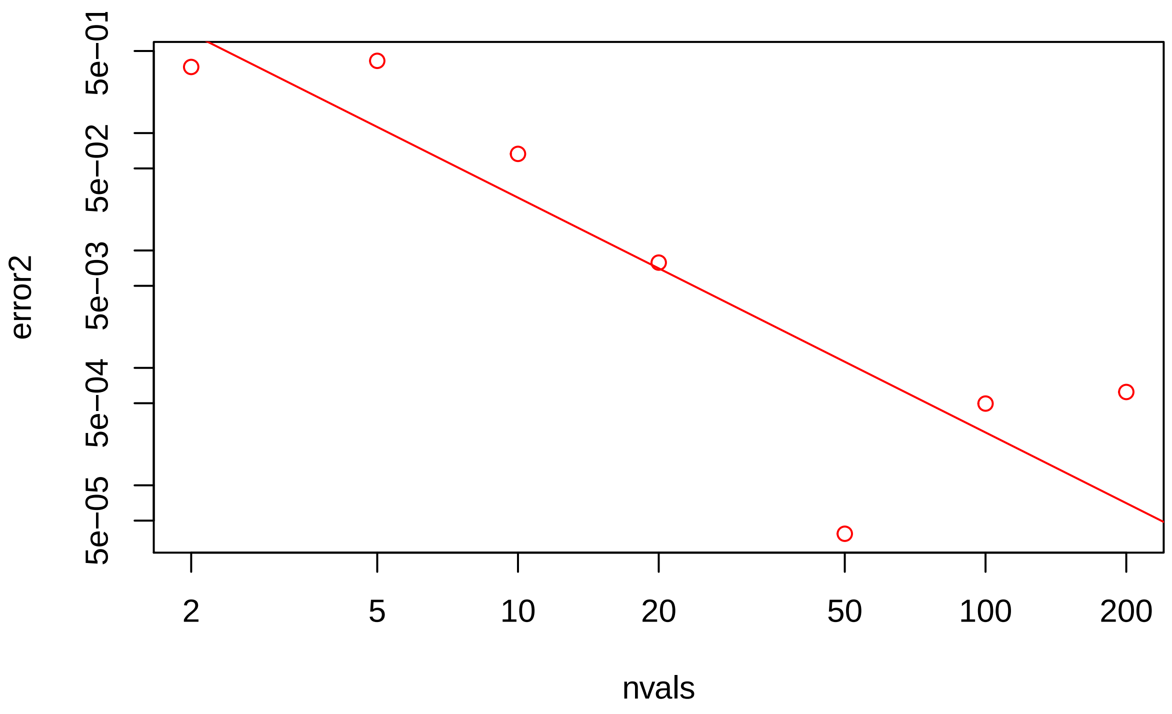


Let's do the same for I_2.

```
nvals <- c(2,5,10,20,50,100,200)
rmse <- c()
i <- 1
error2 <- c()

for(n in nvals){
  h = 1/n
  x = (1:n-0.5)*h
  error2[i] <- abs(I2hat - sum(f2(x)*h))
   i <- i+1
}
plot(nvals, error2, log='xy', col=2)
lm(log10(error2)~log10(nvals))
```

```
##
## Call:
## lm(formula = log10(error2) ~ log10(nvals))
##
## Coefficients:
##  (Intercept)  log10(nvals)
##       0.3574       -1.8843
```

```
abline(0.45,-2,col=2)
```


error2
5e−01
5e−02
5e−03
5e−04
5e−05
2
5
10
20
50
100
200
nvals