Lab Assignment 2

Shwetha Vandagadde, Suhani Ariga and Hoda Fakharzadeh

11/16/2020

Question 1: Optimizing parameters

1

```
parabolic_inter = function(a,x){
  a0 = a[1]
  a1 = a[2]
  a2 = a[3]
  res = a0 + a1*x + a2*(x^2)
  return(res)
}
sum_square_error = function(a,x,func){
  x0 = x[1]
  x1 = x[2]
  x2 = x[3]
  res = sum((func(x0)-parabolic_inter(a,x0))^2,(func(x1)-parabolic_inter(a,x1))^2,
            (func(x2)-parabolic_inter(a,x2))^2)
  return(res)
}
opt = function(x,func){
  a = c(0,0,0)
  res = optim(a,sum_square_error,x = x,func = func)
  res = res$par
  return(res)
}
```

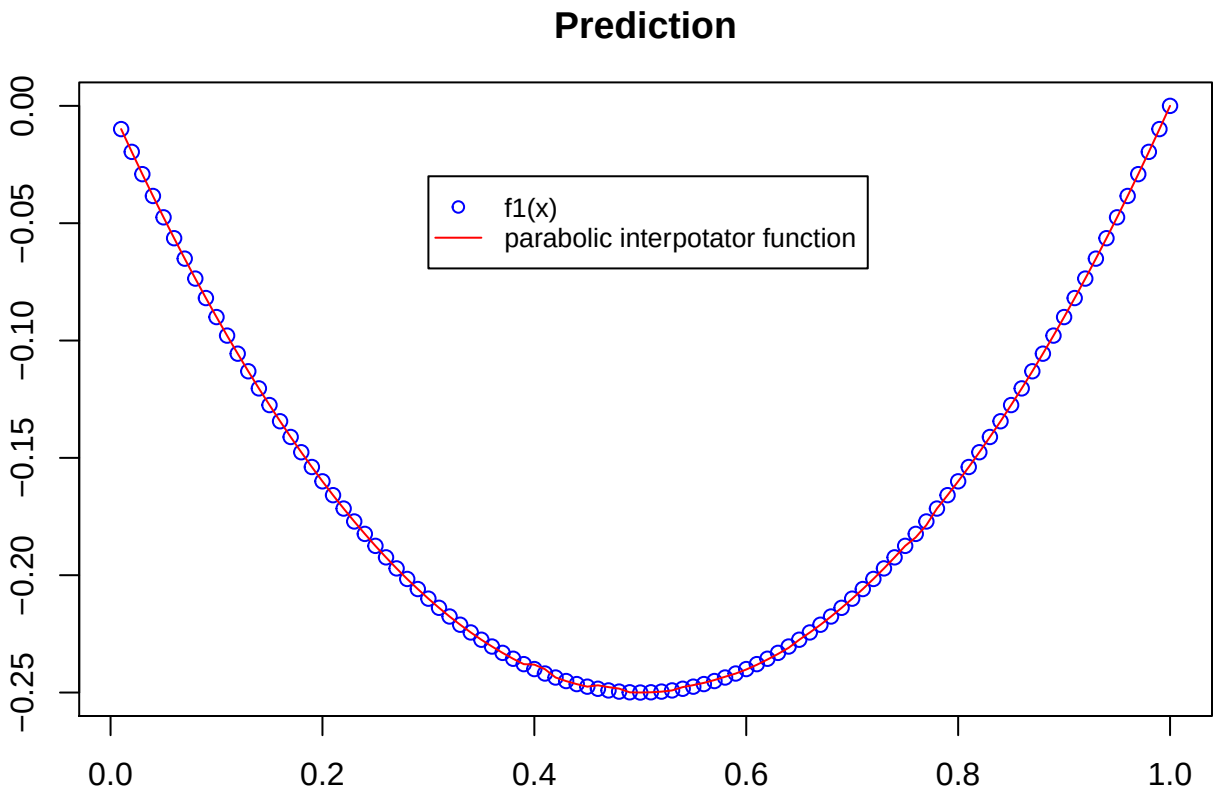
2

```
approx = function(n,func){
 interval = 1/n
 midpoint = 1/(2*n)
 res = data.frame()
 for(i in 1:n){
   end_point = i/n
   x = c((end_point - interval),(end_point - midpoint),end_point)
   res = append(res,as.data.frame(opt(x,func)))
 }
```

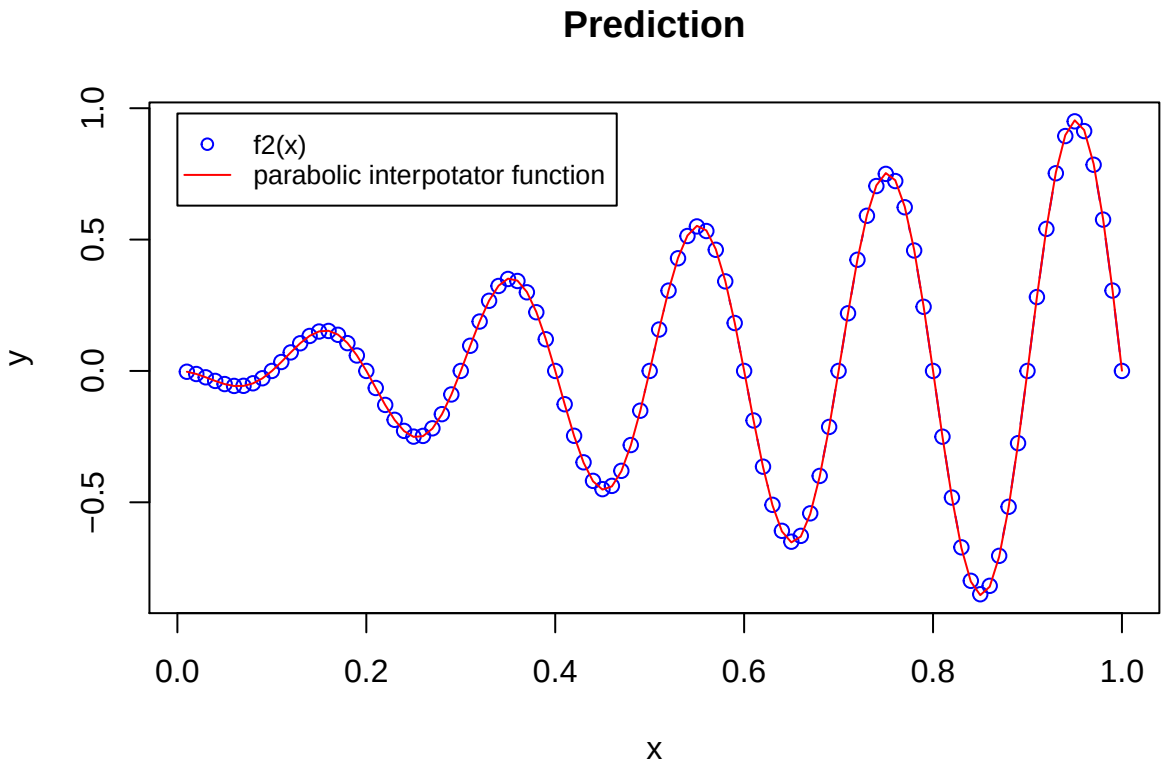
```
 return(res)
}
```

3

Comparing the result from f1(x) and parabolic interpolator function for 100 subintervals.



Comparing the result from f2(x) and piecewise parabolic interpolator function for 100 subintervals.



Observing the above two plots , we can say that the prediction from piecewise parabolic interpolater was fair as the predictions has very low errors.

Appendix

```
knitr::opts_chunk$set(echo = TRUE)
parabolic_inter = function(a,x){
  a0 = a[1]
  a1 = a[2]
  a2 = a[3]
  res = a0 + a1*x + a2*(x^2)
  return(res)
}
sum_square_error = function(a,x,func){
  x0 = x[1]
  x1 = x[2]
  x2 = x[3]
  res = sum((func(x0)-parabolic_inter(a,x0))^2,(func(x1)-parabolic_inter(a,x1))^2,
            (func(x2)-parabolic_inter(a,x2))^2)
  return(res)
}
opt = function(x,func){
  a = c(0,0,0)
```

```
  res = optim(a,sum_square_error,x = x,func = func)
  res = res$par
  return(res)
}
approx = function(n,func){
 interval = 1/n
 midpoint = 1/(2*n)
 res = data.frame()
 for(i in 1:n){
   end_point = i/n
   x = c((end_point - interval),(end_point - midpoint),end_point)
   res = append(res,as.data.frame(opt(x,func)))
 }
 return(res)
}
f1 = function(x){
  res = -x *(1-x)
  return(res)
}
f2 = function(x){
  res = -x * sin(10*pi*x)
  return(res)
}
x = c()
for(i in 1:100){x[i] = i/100}

approx1 = approx(100,f1)
interpolate_result1 = c()
for(i in 1:length(approx1)){
  a = as.vector(approx1[[i]])
  interpolate_result1 = append(interpolate_result1,parabolic_inter(a,x[i]))
}
par(mar = c(3,3,3,0))
plot(x,f1(x),xlab = "x",ylab = "y",col = "blue", type = "b")
lines(x,interpolate_result1,col = "red")
title("Prediction")
legend(0.3, -0.03, legend=c("f1(x)", "parabolic interpotator function"),
       col=c("blue", "red"), pch=c(1,NA),lty=c(0,1), cex=0.8)

approx2 = approx(100,f2)
interpolate_result2 = c()
for(i in 1:length(x)){
  a = as.vector(approx2[[i]])
  interpolate_result2 = append(interpolate_result2,parabolic_inter(a,x[i]))
}
plot(x,f2(x),xlab = "x",ylab = "y",col = "blue", type = "b")
lines(x,interpolate_result2,col = "red")
title("Prediction")
legend(0, 0.98, legend=c("f2(x)", "parabolic interpotator function"),
       col=c("blue", "red"), pch=c(1,NA),lty=c(0,1), cex=0.8)
```