Computing HW 4

2020324011
민동은

5.1. $p_i(x) = f(x_i) + (x-x_i)\frac{f(x_{i+1})-f(x_i)}{x_{i+1}-x_i}$

By Taylor's expansion. $f(x_{i+1}) = f(x_i) + f'(x_i)(x_{i+1}-x_i) + \frac{f''(x_i)}{2!}(x_{i+1}-x_i)^2 + \cdots$

$f(x_{i+1}) - f(x_i) = f'(x_i)(x_{i+1}-x_i) + \frac{f''(x_i)}{2!}(x_{i+1}-x_i)^2 + \cdots$

$\rightarrow p_i(x) = f(x_i) + (x-x_i)\left\{f'(x_i) + \frac{f''(x_i)}{2!}(x_{i+1}-x_i) + \frac{f'''(x_i)}{3!}(x_{i+1}-x_i)^2 + \cdots\right\}$

$= f(x_i) + f'(x_i)(x-x_i) + \frac{f''(x_i)}{2!}(x_{i+1}-x_i)(x-x_i) + \underbrace{\frac{f'''(x_i)}{3!}(x_{i+1}-x_i)^2(x-x_i) + \cdots}_{=O(\frac{1}{n^3})}$

$\therefore$ (5.14) is proved.

5.3. (a), (b) : R codes attached next page.

(c) $u = \frac{e^\mu}{1+e^\mu}$. $\rightarrow \frac{u}{1-u} = e^\mu$. $\log(\frac{u}{1-u}) = \mu$.

Integral transform : $(Tf)(u) = \int_{t_1}^{t_2} f(t)K(t,u)\,dt$.

Since $f(\mu) = \sqrt{\frac{7}{18\pi}}\cdot\exp(-\frac{7}{18}(x-\mu)^2)$, transformed : $\int_{\frac{e^3}{1+e^3}}^{1} f(\log(\frac{u}{1-u}))/(\frac{u}{1-u})\,du$.

(d) $u = \frac{1}{\mu}$. transformed : $\int_0^{\frac{1}{3}} f(\frac{1}{u})\cdot(\frac{1}{u})^2\,du$.

5.4. $E(Y) = E(\frac{a-1}{X})$. $X \sim Unif(1,a)$. $(a>1)$.

$= \int_{\frac{a-1}{a}}^{a-1} \frac{a-1}{x}\cdot\frac{1}{a-1}\,dx = \int_{\frac{a-1}{a}}^{a-1}\frac{1}{x}\,dx = [\log x]_{\frac{a-1}{a}}^{a-1} = \log a$.

$\rightarrow$ Check if $\int_{\frac{a-1}{a}}^{a-1}\frac{1}{x}\,dx$ is the same as $\log a$.

Using Romberg's integration of trapezoidal rule,

$\hat{T}(2n) = \frac{1}{2}\hat{T}(n) + \frac{h}{2}\sum_{i=1}^{n} f(a+(i-\frac{1}{2})h)$ and $\hat{T}_{i,0} = \hat{T}(2^i)$ for $i = 0, \cdots, m$.

$\hat{T}_{i,j} = \frac{4^j\hat{T}_{i,j-1} - \hat{T}_{i-1,j-1}}{4^j-1}$ for $j = 1, \cdots, i$ and $i = 1, \cdots, m$.

Computing HW4

Dongeun Min

11/25/2020

```r
library("Rcpp")
current_path = rstudioapi::getActiveDocumentContext()$path
setwd(dirname(current_path))
sourceCpp("Computing_HW4.cpp")
```

Question 5.3 (a)

```r
x = c(6.52, 8.32, 0.31, 2.82, 9.96, 0.14, 9.64)
xbar = mean(x)
k <- 7.84654
trapezoid(xbar, -100, 100, 10000)
```

```
## [1] 0.1274447
```

```r
k * trapezoid(xbar, -100, 100, 10000)
```

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

When k=7.84654 is multiplied with the integration (calculated with the trapezoidal rule), the integrated value becomes 1. Therefore, 7.84654 is the right proportionality constant.

Question 5.3 (b)

```r
k * riemann(xbar, 2, 8, 500)
```

```
## [1] 0.9960067
```

```r
k * trapezoid(xbar, 2, 8, 500)
```

```
## [1] 0.9960543
```

```r
k * simpson(xbar, 2, 8, 500)
```

```
## [1] 0.9960547
```

Question 5.3 (c)

First: ignore the singularity at 1

```
lower = exp(3)/(1+exp(3))
upper = 1
k * criemann(xbar, lower, upper, 500)
```

```
## [1] 0.990889
```

```
k * csimpson(xbar, lower, upper, 500)
```

```
## [1] NaN
```

when using simpson's rule, there exists a singularity problem at the upper bound 1.

Second: fix the singularity at 1 by fixing the upper bound as a number slightly smaller than 1

```
lower = exp(3)/(1+exp(3))
upper = 1 - 0.00001
k * csimpson(xbar, lower, upper, 500)
```

```
## [1] 0.9908565
```

Question 5.3 (d)

```
lower = 10^(-10)  #use a lower bound very slightly bigger than 0
upper = 1/3
k * dsimpson(xbar, lower, upper, 1000)
```

```
## [1] 0.9908595
```

Question 5.4

```
a = 5  #arbitrarily choose value for a
lower = (a-1)/a
upper = a-1
t_i0 = c(1:7)
for(i in 1:7){
  t_i0[i] = gtrapezoid(lower, upper, 2^(i-1))
}
t_i1 = c(1:6)
for(i in 1:6){
  t_i1[i] = (4*t_i0[i+1] - t_i0[i]) / (4-1)
}
t_i2 = c(1:5)
for(i in 1:5){
  t_i2[i] = (4*4*t_i1[i+1] - t_i1[i]) / (4^2-1)
```

```
}
t_i3 = c(1:4)
for(i in 1:4){
  t_i3[i] = ((4^3)*t_i2[i+1] - t_i2[i]) / (4^3-1)
}
t_i4 = c(1:3)
for(i in 1:3){
  t_i4[i] = ((4^4)*t_i3[i+1] - t_i3[i]) / (4^4-1)
}
t_i5 = c(1:2)
for(i in 1:2){
  t_i5[i] = ((4^5)*t_i4[i+1] - t_i4[i]) / (4^5-1)
}
t_66 = ((4^6)*t_i5[2] - t_i5[1]) / (4^6-1)
log(a)
```

```
## [1] 1.609438
```

```
t_i0
```

```
## [1] 2.400000 1.866667 1.683333 1.628968 1.614406 1.610686 1.609750
```

```
t_i1
```

```
## [1] 1.688889 1.622222 1.610847 1.609552 1.609446 1.609438
```

```
t_i2
```

```
## [1] 1.617778 1.610088 1.609466 1.609439 1.609438
```

```
t_i3
```

```
## [1] 1.609966 1.609456 1.609438 1.609438
```

```
t_i4
```

```
## [1] 1.609454 1.609438 1.609438
```

```
t_i5
```

```
## [1] 1.609438 1.609438
```

```
t_66
```

```
## [1] 1.609438
```

From the R codes of the previous page, a triangular array is made like this :

$$\begin{pmatrix} 2.4 & & & & & & \\ 1.867 & 1.689 & & & & & \\ 1.683 & 1.622 & 1.6128 & & & & \\ 1.629 & 1.611 & 1.610 & 1.6099 & & & \\ 1.614 & 1.6095 & 1.60946 & 1.609456 & 1.60945 & & \\ 1.610 & 1.6094 & 1.609439 & 1.609438 & 1.609438 & 1.609438 & \\ 1.6097 & 1.609438 & 1.609438 & 1.609438 & 1.609438 & 1.609438 & 1.609438 \end{pmatrix}$$

$(a=5, m=6)$.

$\log a = 1.609438$, which matches the results of $m=5$, $m=6$ rows.

(highlighted above).