Lab3

Zhipeng Chen

10/1/2021

```
rm(list = ls())
sessionInfo()
```

```
## R version 4.1.1 (2021-08-10)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 19042)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=English_United States.1252
## [2] LC_CTYPE=English_United States.1252
## [3] LC_MONETARY=English_United States.1252
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.1252
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## loaded via a namespace (and not attached):
##  [1] compiler_4.1.1  magrittr_2.0.1  fastmap_1.1.0   tools_4.1.1
##  [5] htmltools_0.5.2 yaml_2.2.1      stringi_1.7.4   rmarkdown_2.11
##  [9] knitr_1.34      stringr_1.4.0   xfun_0.25       digest_0.6.27
## [13] rlang_0.4.11    evaluate_0.14
```

Case study on numerical integration

1

```
midpoint <- function(f, a, b) {
  result <- (b - a) * f((a + b)/2)
  return(result)
}

trapezoid <- function(f, a, b) {
  result <- ((b - a)/2) * (f(a) + f(b))
  return(result)
}

midpoint(sin, 0, pi)
```

```
## [1] 3.141593
```

```
trapezoid(sin, 0, pi)
```

```
## [1] 1.923607e-16
```

2

```
midpoint.composite <- function(f, a, b, n = 10) {
  points <- seq(a, b, length = n + 1)

  area <- 0
  for (i in seq_len(n)) {
    area <- area + midpoint(f, a, b)
  }
  return(area)
}

trapezoid.composite <- function(f, a, b, n = 10) {
  points <- seq(a, b, length = n + 1)

  area <- 0
  for (i in seq_len(n)) {
    area <- area + trapezoid(f, a, b)
  }
  return(area)
}

midpoint.composite(sin, 0, pi, n = 10)
```

```
## [1] 31.41593
```

```
midpoint.composite(sin, 0, pi, n = 100)
```

```
## [1] 314.1593
```

```
midpoint.composite(sin, 0, pi, n = 1000)
```

```
## [1] 3141.593
```

```
trapezoid.composite(sin, 0, pi, n = 10)
```

```
## [1] 1.923607e-15
```

```
trapezoid.composite(sin, 0, pi, n = 100)
```

```
## [1] 1.923607e-14
```

```r
trapezoid.composite(sin, 0, pi, n = 1000)
```

```
## [1] 1.923607e-13
```

There is a loop in the function so we will get different results per loop times.

3

```r
midpoint.composite.vectorize <- function(f, a, b, n = 10) {
  points <- seq(a, b, length = n + 1)
  areas <- midpoint(f, points[rep(1,n)], points[rep((n+1),n)])
  return(sum(areas))
}

trapezoid.composite.vectorize <- function(f, a, b, n = 10) {
  points <- seq(a, b, length = n + 1)
  areas <- trapezoid(f, points[rep(1,n)], points[rep((n+1),n)])
  return(sum(areas))
}

midpoint.composite.vectorize(sin, 0, pi, n = 10)
```

```
## [1] 31.41593
```

```r
midpoint.composite.vectorize(sin, 0, pi, n = 100)
```

```
## [1] 314.1593
```

```r
midpoint.composite.vectorize(sin, 0, pi, n = 1000)
```

```
## [1] 3141.593
```

```r
trapezoid.composite.vectorize(sin, 0, pi, n = 10)
```

```
## [1] 1.923607e-15
```

```r
trapezoid.composite.vectorize(sin, 0, pi, n = 100)
```

```
## [1] 1.923607e-14
```

```r
trapezoid.composite.vectorize(sin, 0, pi, n = 1000)
```

```
## [1] 1.923607e-13
```

The results are the same as above.

4

```
system.time(midpoint.composite(sin, 0, pi, n = 10000))
```

```
##    user  system elapsed
##    0.02    0.00    0.02
```

```
system.time(trapezoid.composite(sin, 0, pi, n = 10000))
```

```
##    user  system elapsed
##    0.01    0.00    0.02
```

```
system.time(midpoint.composite.vectorize(sin, 0, pi, n = 10000))
```

```
##    user  system elapsed
##       0       0       0
```

```
system.time(trapezoid.composite.vectorize(sin, 0, pi, n = 10000))
```

```
##    user  system elapsed
##       0       0       0
```

Normal Equations

```
my.normal.equations <- function(X, Y) {
  if (!is.vector(Y)) {
    stop("Y is not a vector!")
  }

  if (!is.matrix(X)) {  # force X to be a matrix for now
    stop("X is not a matrix!")
  }

  if (dim(X)[1] != length(Y)) {
    stop("Dimension mismatch between X and Y!")
  }

  return(solve(t(X) %*% X) %*% t(X) %*% Y) # finish the calculation for beta
}

set.seed(7360)
sample.size <- 100
num.col <- 2
X <- matrix(rnorm(sample.size * num.col), nrow = sample.size, ncol = num.col)
X <- cbind(1, X)
Y <- rnorm(sample.size)

system.time(result.lm <- lm(Y ~ X[, 2] + X[, 3]))
```

```
##    user  system elapsed
##       0       0       0
```

```
summary(result.lm)
```

```
##
## Call:
## lm(formula = Y ~ X[, 2] + X[, 3])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.92783 -0.58015  0.05852  0.57220  1.82080
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.04356    0.09662   0.451   0.6532
## X[, 2]      -0.05051    0.09222  -0.548   0.5852
## X[, 3]       0.17642    0.09189   1.920   0.0578 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9171 on 97 degrees of freedom
## Multiple R-squared:  0.04138,    Adjusted R-squared:  0.02162
## F-statistic: 2.094 on 2 and 97 DF,  p-value: 0.1288
```

```
system.time(result.my.normal.equations <- my.normal.equations(X, Y))
```

```
##    user  system elapsed
##       0       0       0
```

```
result.my.normal.equations
```

```
##             [,1]
## [1,]  0.04355651
## [2,] -0.05050778
## [3,]  0.17641649
```

The results are matching.