**Please, upload your answers in Canvas: assignment PC LAB – May 10**

Name: Vu The Doan

Student nr: 12918687

**PC5b – Numerical Integration**

**1.** In this question, you are going to investigate the Trapezium rule for integration. The idea is to divide the interval  into *N* subintervals each with a length of  For each subinterval, the function is approximated by linear interpolation. The more subintervals are used, the more accurate the area is approximated.

a. First, we consider the integral  Below, the situation is shown by splitting the interval (0,1) into 2 subintervals each with a length of 0.5.



Write an R-script that generates the output:

Area Area

Interval: Exact: Approximation: Error:

==============================================================

(1): ( 0.00,0.50) 0.39062500 0.40625000 -0.01562500

(2): ( 0.50,1.00) 0.35937500 0.40625000 -0.04687500

==============================================================

Total: 0.75000000 0.81250000 -0.06250000

Check: h/2\*sum(w.\*f)=0.81250000

Hints:

* Use the R-script PC5b\_Q1\_incomplete.R (see **Canvas**); the commented lines (i.e. **#**) have to be coded
* Define an R-function

fprimitive\_1a <- function(x)

that calculates .

* For checking, define the vector *w* that contains the weights, i.e.



You only have to set the interior elements to 2 in the R-script.

Complete the following table:

|  |  |  |  |
| --- | --- | --- | --- |
| *k* | *N* | Error | Factor = error(*k*–1*)/*error(*k*) |
| 1 | 2 | -0.06250000 | \* |
| 2 | 4 | -0.01562500 | 4.0000 |
| 3 | 8 | -0.00390625 | 4.0000 |
| 4 | 16 | -0.00097656 | 4.0000 |

b. Next, do the same but now for 



Area Area

Interval: Exact: Approximation: Error:

==============================================================

(1): ( 1.00,1.50) 0.23022893 0.23279578 -0.00256686

(2): ( 1.50,2.00) 0.11043470 0.11499694 -0.00456224

==============================================================

Total: 0.34066362 0.34779272 -0.00712910

Check: h/2\*sum(w.\*f)=0.34779272

Use for the antiderivative the following function:

primitive\_1b <- function(x){

return(sqrt(2\*pi)\*pnorm(x))

}

Again, complete the following table:

|  |  |  |  |
| --- | --- | --- | --- |
| *k* | *N* | Error | Factor = error(*k*–1*)/*error(*k*) |
| 1 | 2 | -0.00712910 | \* |
| 2 | 4 | -0.00175737 | 4.0567 |
| 3 | 8 | -0.00043782 | 4.0139 |
| 4 | 16 | -0.00010936 | 4.0035 |

What can you say about the error when *h* halves?

**2.** We now turn to integration using quadratic polynomials. We consider the function  on the interval  Using the script PC5b\_Q2\_fig.R (see **Canvas**) you can see the quadratic approximation (in red lines) based on *M* subintervals.



a. Watch how  (the blue curve) is increasingly better approximated by the quadratic polynomials by varying the number of subintervals *M* from 1 to 8.

b. We take  and consider the first subinterval  For this subinterval, we have  and  Determine using polyfit the coefficients of the quadratic polynomials that goes through these 3 points, i.e. determine  such that  passes through . Determine the exact area under this quadratic polynomial using (substitute the found coefficients for ):

library(Ryacas)

t <- ysym("t")

int.exact <- integrate(a\*t^2+b\*t+c,t,0,pi/3)

eval(as\_r(int.exact))

Does this coincide with ?

c. Modify the script of question 1 such that it carries out Simpson’s 1/3 rule over the subintervals, i.e. determine the area using .

Area Area

Interval: Exact: Approximation: Error:

==============================================================

(1): ( 0.00,1.05) 0.10000000 0.19791590 -0.09791590

(2): ( 1.05,2.09) 0.20000000 0.39583181 -0.19583181

(3): ( 2.09,3.14) 0.10000000 0.19791590 -0.09791590

==============================================================

Total: 0.40000000 0.79166361 -0.39166361

Complete the following table:

|  |  |  |
| --- | --- | --- |
| *M* | Error | Factor = error(*M*–1*)/*error(*M*) |
| 1 | -1.69439510 | \* |
| 2 | 1.35736220 | -1.2483 |
| 3 | -0.39166361 | -3.4656 |
| 4 | 0.50738705 | -0.7719 |
| 5 | 1.58112098 | 0.3209 |
| 6 | 1.08467481 | 1.4577 |

**3**. [Exercise 3 of Ch.11 spuRs]

The standard normal distribution function is given by

For , the 100p standard normal percentage point is defined as that for which

Using the function **Phi(z)** from Example 11.2.1, calculate for and .

Hint: use the R-functions " **newtonraphson.r**" and "**simpson\_n.r**" of the spuRs package and express the problem as a root-finding problem.

4. [Exercise 4 of Ch.11 spuRs]

Consider

.

Let be the approximation to given by the trapezoid rule with a partition of size and let be the approximation given by Simpson’s 1/3-rule with a partition of size .

Let be the smallest value of for which and let be the smallest value of for which . Plot and against for, (see figure below).

Hint: use the R-functions "**trapezoid.r**" and "**simpson\_n.r**" of the spuRs package.

