Intercomparison between the Euler and Runge-Kutta method to solve ordinary differential equations with initial value using Rstudio

Claudio H. González-Rojas, PhD.
Laboratorio de Computación Cuántica, Melipilla,
P.O. Box 9580000, RM, Chile, chgonzalezr@academicos.uta.cl

Abstract

A comparison between Euler and Runge-Kutta methods to solve differential equations y´= 2y with initial values is presented, in rstudio environment, according to a code in R language. The comparison is refered to mean value of second moment of residual (RMSE) and precision that reach each one, in function the number of division of intervals from ten in ten, for a interval defined, begin in ten subdivisions and ending in fifty.

The results reached allow to see that RMSE statistic appears as a good descriptor to decide between two numerical approximation methods such as Euler and fourth order Runge-Kutta, since the mean value of the second moment is at least four orders of magnitude smaller in the case of Runge-Kutta respect to Euler.

The convergence in the accuracy of the approximation is faster in the Runge-Kutta case with respect to Euler, for this differential equation.

Finally, the rapid convergence of the accuracy allows us to appreciate that by making the division into 50 sub-intervals the accuracy conceptually reaches full accuracy, assuming Runge-Kutta.

1 Introduction

The numerical methods for solving initial value differential equations are sufficiently well covered in the literature. One of them is the Euler method. Another procedure is the Runge-Kutta procedure. Both procedures can provide accurate and fast converging solutions.

However, how much does the numerical approximation improve when comparing the same equation by both methods? Moreover, there are quite a few codes used in the literature, but coding in R language has yet to be found. The present paper intends to address both questions.

2. Theoretical aspects

2.1 Euler Method

Now consider the class of first-order initial-value problems of the form

$$\frac{dy}{dx} = f\left(x, y\right) \quad , \quad y\left(x_0\right) = y_0$$

The problem is to find the solution y(x) of this equation that passes through the initial point (x_0,y_0) in the xy plane for values of x in the interval [a,b]. The starting point will be to examine the graphical basis of the method proposed by Euler in a step consisting of the following:

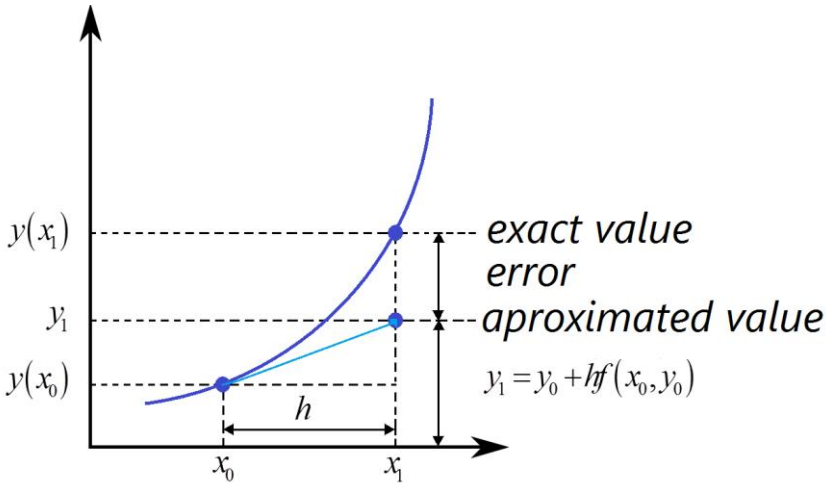


Figure 1. Graphical basis of Euler's one-step method

Figure 1 shows the differential equation (thick line). A range of size h (one step) is defined between the abscissae x0,x1, where the equation represents h:

$$x_1 = x_0 + h$$

As an initial value, we take $(x0,y(x0))$. A tangent to the differential equation is drawn at this coordinate. This tangent intersects both the curve of the equation and the point $(x0,y0)$. The line $x=x1$ is found to the right of $x0$. According to Figure 1, this line intersects the coordinate $(x1, y1(x1))$, which yields the approximate solution of the differential equation. This same line intersects the coordinate $(x1, y(x1))$, which is the exact solution of the differential equation, expressed according to:

$$y_1 = y_0 + hf(x_0, y_0)$$

According to what will be shown below. Between this approximation and the exact solution, there is an error, as can be seen in Figure 1. Now, $y1$ will be designated as the approximation of the solution $y(x1)$, which is exact. How do we know $y1$? Consider its derivative and approximate the derivative in the differential equation by its difference quotient, given by:

$$y' = \frac{dy}{dx} \approx \frac{\Delta y}{\Delta x} = \frac{y_1 - y_0}{h}$$

According to Figure 1, the slope of the tangent to the curve at $(x0,y0)$ is $y'(x0)=f(x0,y0)$, so we can write:

$$\frac{y_1 - y_0}{h} \approx f\left(x_0, y_0\right)$$

solving for y1, results in

$$y_1 = y_0 + hf\left(x_0, y_0\right)$$

How to improve the solution? By considering the same interval as above but subdividing it. This example subdivides it into two equidistant intervals between $x0,x1$ and $x1,x2$. See Figure 2. In Figure 2, three points of this type are presented, and the interval is divided into two sub-intervals of one step each.

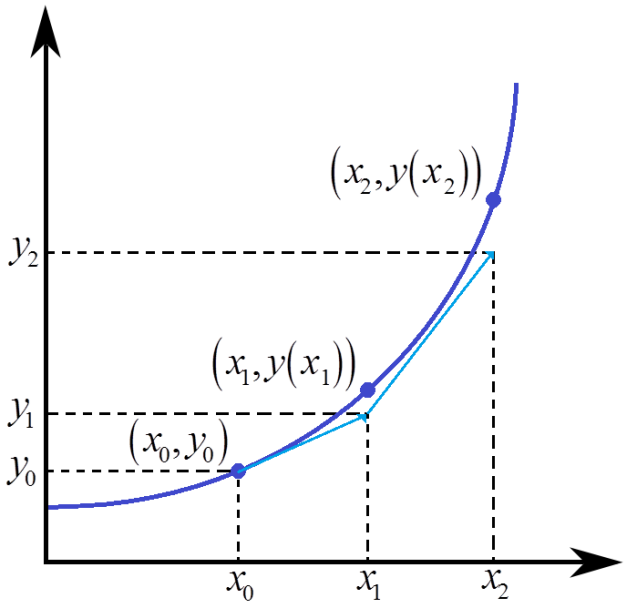


Figure 2: Graphical foundations of Euler's Method, two divisions.

In Euler's Method, again, the first step is to use the initial condition. This step is represented as a point on the solution curve (x0,y(x0)=(x0,y0), as seen in Figure 2. The curve is then plotted at the point (x0,y0), represented by a thick line. The next step is to move to the right of x0.

The solution equation according to Figure 1 is:

$$y_1 = y_0 + hf\left(x_0, y_0\right)$$

Next, another coordinate is moved to the right of x1 and x2 is fixed. This is the second stage; thus the equation for x2 is:

$$x_2 = x_1 + 2h$$

Assuming that h is constant. That is, equidistant abscissae.

What is the following step? Take advantage of the previous approximation y1. Using y1 as the intercept, plot a line to intersect the x2 abscissa.

Again, the solution at x=x2 has yet to be found. However, it is possible to determine the tangent line and find the point of intersection it makes with the vertical. According to Figure 2, this intersection point is close to the point on the curve of the equation. Then, y2 will be designated as the approximation of the solution y(x2) that is exact and, as in the previous step, return to determine y2, which equation is, reiterating the previous step:

$$y_2 = y_1 + hf\left(x_1, y_1\right)$$

Generalizing, for n equidistant subintervals of the form:

$$x_n = x_0 + nh$$

and repeating the previous procedure leads to the following algorithm for determining a numerical solution to the initial value problem:

$$y\left(x_0\right) = y_0$$

$$y_n = y_{n-1} + hf\left(x_{n-1}, y_{n-1}\right), n = 0,1,,,N$$

Given that each stage is solved in one step at a time, it is said that this is the one-step EULER method.

2.2 Fourth-order Runge_Kutta method

The Runge-Kutta method was first developed around 1900 by the German mathematicians C. Runge and M.W. Kutta for the approximation of solutions of ordinary differential equations, starting from an initial value. Several procedures exist, all starting from Euler's method but proposing, at the same time, several generic iterative, explicit, and implicit methods. The fourth-order Runge-Kutta method is a member of the Runge-Kutta family of methods. It is so widely used that it is often called "RK4" or the "Runge-Kutta method."

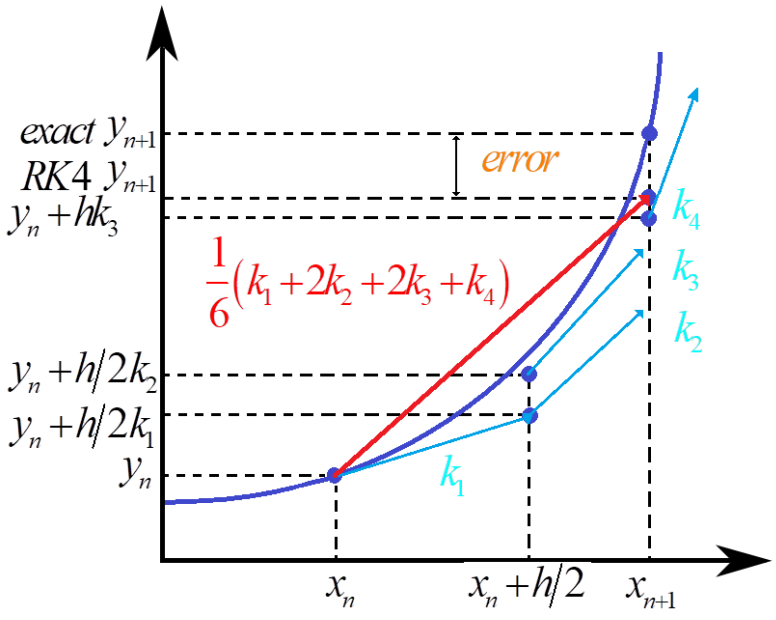


Figure 3: Graphical foundations of the fourth-order Runge Kutta Method

Let it be a differential equation:

$$y'(x) = f(x, y(x))$$

Let us define an initial value problem as:

$$y'(x) = f(x, y), y(x_0) = y_0$$

The first step, according to this method, is to plot a tangent through the point x_0,y_0, considering an interval given by:

$$x_1 = x_0 + \frac{h}{2}$$

$$k_1 = f(x_0, y_0)$$

and its equation is:

$$y_1 = y_0 + \frac{h}{2} k_1$$

See Figure 3

The second step is to move to the right of x_0, at $h/2$, and plot a line that will intersect with the previous equation. This intersection has coordinates.

$$\left(x_0 + \frac{h}{2}, y_0 + \frac{h}{2} k_1 \right)$$

The initial value of the second equation, from which the second tangent, k_2, is plotted, the value of which is

$$k_2 = f\left(x_0 + \frac{h}{2}, y_0 + \frac{1}{2} k_1 \right)$$

And the equation describing it is:

$$y_1 = y_0 + \frac{h}{2} k_2$$

The third step is to use the same previous abscissa as the origin coordinate and take half the value of the previous gradient k2, from which a straight line is plotted to intercept the next coordinate:

$$\left(x_0 + \frac{h}{2}, y_0 + \frac{h}{2} k_2 \right)$$

Serving as initial value to the third tangent that approximates the original function and whose value is:

$$k_3 = f\left(x_0 + \frac{h}{2}, y_0 + \frac{1}{2} k_2 \right)$$

where the equation describing it is:

$$y_1 = y_0 + hk_3$$

The fourth and final step is to consider the end of the interval given by:

$$x_1 = x_0 + h$$

The initial value of which is now:

$$\left(x_0 + h, y_0 + hk_3 \right)$$

And from the coordinate

$$\left(x_0 + h, y_0 + hk_3 \right)$$

A tangent is plotted to the original equation, taking the value of the third tangent whose value is now:

$$k_4 = f\left(x_0 + h, y_0 + k_3 \right)$$

where the equation describing it is:

$$y_1 = y_0 + hk_3$$

Thus, the next value (y_{i+1}) is determined by the present value (y_i) plus the product of the interval size (h) times an estimated gradient. The estimated gradient is a weighted average of gradients in the form:

$$y_{n+1} = y_n + \frac{1}{6}h\left(k_1 + 2k_2 + 2k_3 + k_4\right)$$

where:

k_1 is the gradient at the beginning of the interval.

K_2 is the gradient at the interval's midpoint, using k_1 to determine the value of y at the point $x_i + h/2$.

k_3 is again the midpoint gradient, but now using k_2 to determine the value of y

k_4 is the gradient at the end of the interval, using the value of k_3

This form of the Runge-Kutta method is a fourth-order method, which means that the error per step is of order $O(h^5)$, while the total cumulative error has order $O(h^4)$. Therefore, the convergence of the method is of order $O(h^4)$, which is why it is used in computational methods.

It is known that the numerical approximation in the Euler case improves with the subdivision of the intervals. The same is also true for the Runge-Kutta method. As a statistical indicator, let us demonstrate this by using the root mean square value or RMSE, or root mean second-moment value of the residual, defined by:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{i=n}\left(\hat{y}_i - y\left(i\right)\right)^2}$$

where:

$\hat{y}_i$ represents the predicted value

y(i) represents the exact value

While defining accuracy as the relationship:

$$precision = \left(1 - RMSE\right) \cdot 100 \text{ , \%}$$

Thus, both descriptors will be used to explore how much the approximate solution improves compared to the actual value, both in the Euler method and in the Runge-Kutta method, while comparing both methods to appreciate how much better the approximation achieved by Runge-Kutta is compared to Euler.

4. Results and Discussion

Euler Case

This study will begin by taking the Euler case for the equation

$$y' = 2y$$

with an initial value (0,1) and an interval divided into 10 sub-intervals first and then with the same interval divided into 50 sub-intervals.

The exact solution is:

$$y = e^{2x}$$

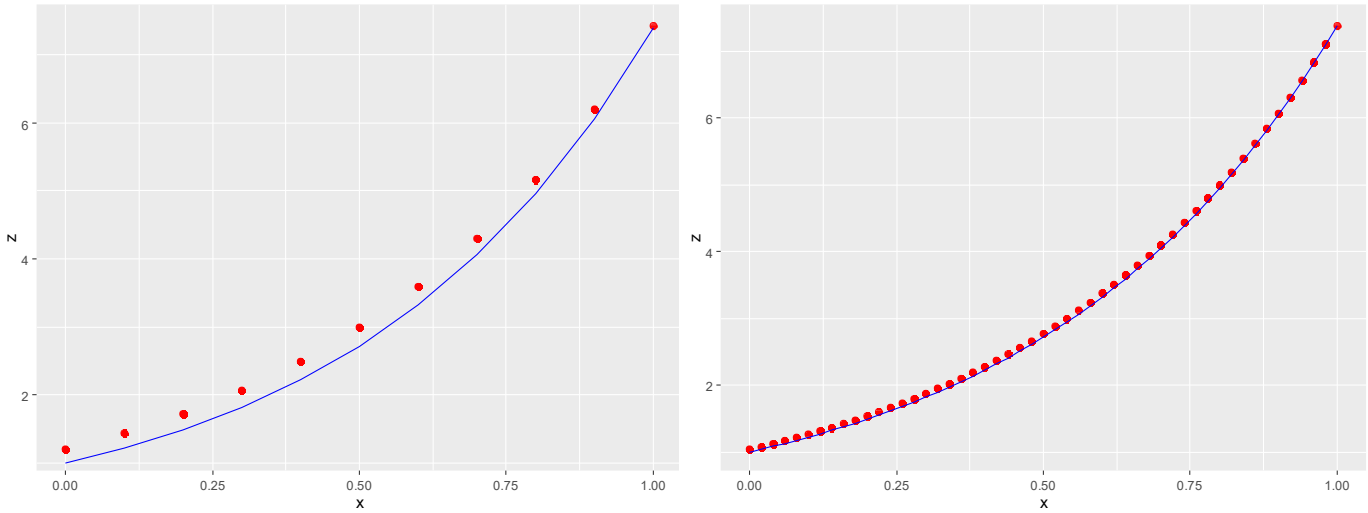


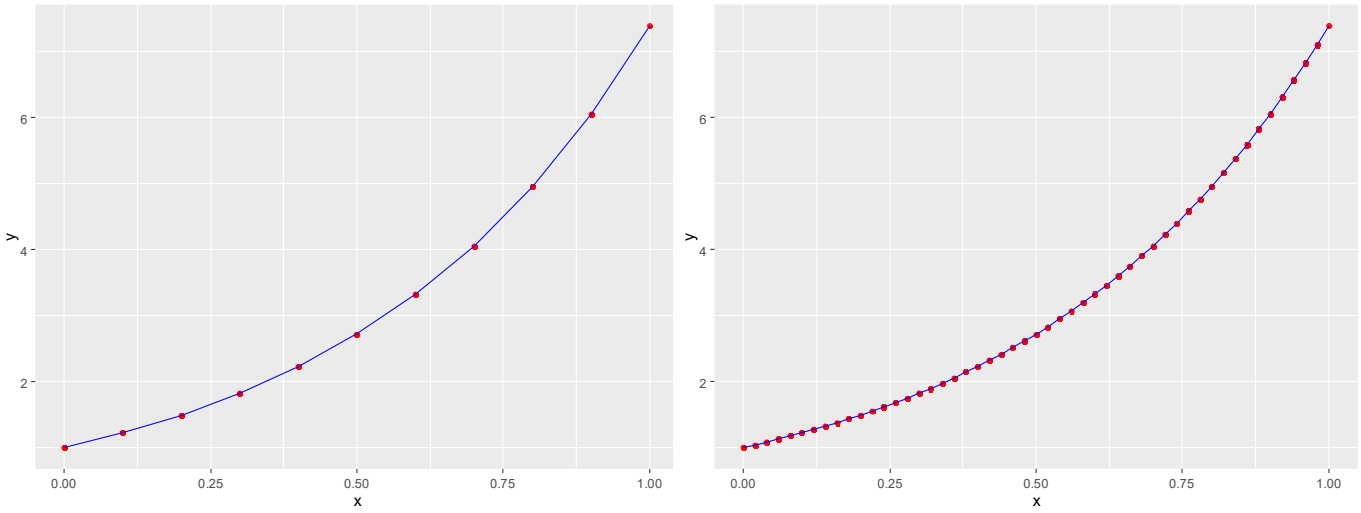
Figure 1. Numerical resolution of the differential equation: left 10 subintervals, right 50 subintervals in the case of Euler's method.

Table I. Summary of the number of partitions and values of the RMSE statistic

n	RMSE	Precision, %
10	0.0129740700	98.70259
20	0.0025030200	99.74970
30	0.0009367759	99.90632
40	0.0004635976	99.95364
50	0.0002679342	99.97321

Case Runge-Kutta

Next, continuing with the Runge-Kutta case, the same differential equation is considered and its exact solution, with an initial value (0,1) and an interval divided into 10 sub-intervals first and then with the same interval divided into 50 sub-intervals.



Numerical resolution of the differential equation: left: 10 sub-intervals, right: 50 sub-intervals in the case of the fourth-order Runge-Kutta method.

Table II. Summary of partition number and values of RMSE statistic and precision

n	RMSE	Precision, %
10	5.27649e-05	99.99472
20	2.533813e-06	99.99975
30	4.201494e-07	99.99996
40	1.167355e-07	99.99999
50	4.31244e-08	100

4.1 Comparison of the two methods

Table III. Summary of the RMSE statistic for both methods

n	Euler	Runge-Kutta
10	0.0129740700	5.27649e-05
20	0.0025030200	2.533813e-06
30	0.0009367759	4.201494e-07
40	0.0004635976	1.167355e-07
50	0.0002679342	4.31244e-08

From this Table, it is seen that the average value of the second moment of the residual, exemplified by the RMSE, is at least four orders of magnitude smaller for the Runge-Kutta of the residual, exemplified in the RMSE, is at least four orders of magnitude smaller in the Runge-Kutta case compared to Euler.

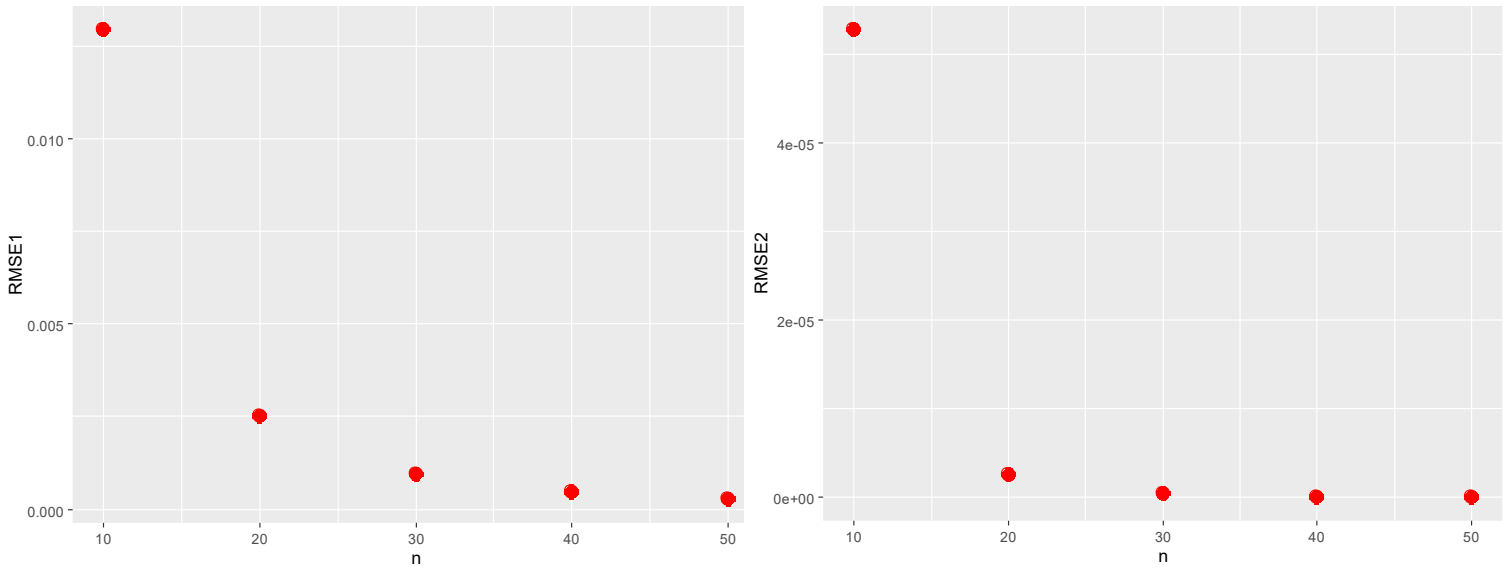


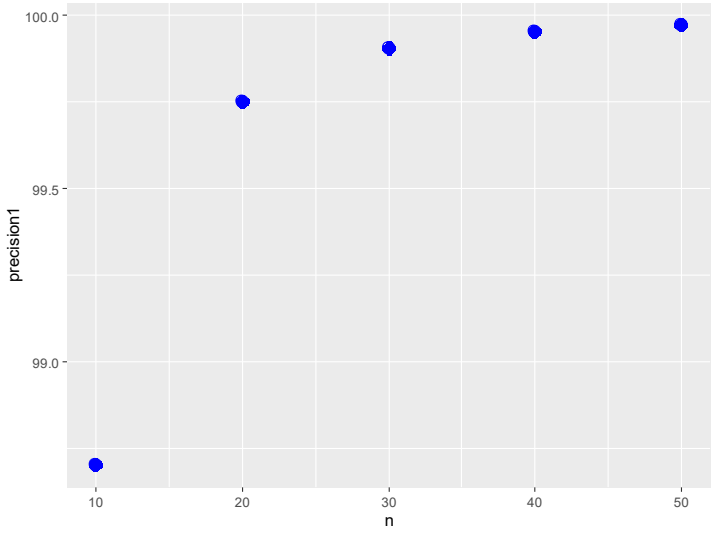
Figure 3. Left, RMSE v/s number of subintervals for the Euler case, and right, RMSE v/s number of subintervals for the Runge-Kutta case.

As shown in Figure 3, it can be observed that the second moment of the residual decreases significantly only when the approximation subintervals are doubled once, i.e., from 10 to 20 subintervals. The same can be observed for both numerical methods.

Table IV. Summary of the precision statistic for both methods

n	Euler	Runge-Kutta
10	98.70259	99.99472
20	99.74970	99.99975
30	99.90632	99.99996
40	99.95364	99.99999
50	99.97321	100

Table IV shows that for the same given interval between 0.1, considering 50 sub-intervals under the Euler approximation is equivalent, in the error committed, to a Runge-Kutta run considering 10 sub-intervals of the interval. In the same context, a run of 10 sub-intervals in the Runge-Kutta case gives a higher accuracy than a run of 50 sub-intervals in the case of the Euler method.



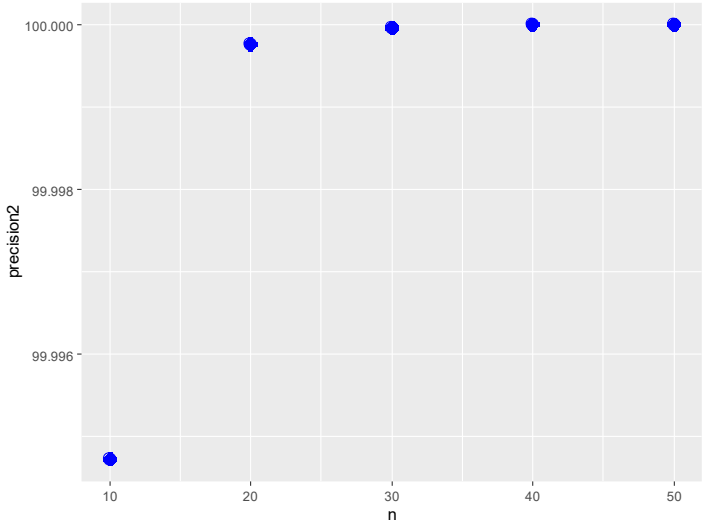


Figure 4. Left, Precision v/s number of subintervals for the Euler case. Right, Precision v/s the number of subintervals for the fourth-order Runge-Kutta case.

The convergence in the approximation accuracy is faster in the Runge-Kutta case for this equation and this interval, given that by dividing the equation into 50 subintervals, the accuracy conceptually reaches full accuracy.

5. Concluding remarks

The RMSE statistic appears as a valid descriptor to decide between two numerical approximation methods, such as the Euler method and the fourth-order Runge-Kutta method, as the average value of the second moment is, at least, four orders of magnitude smaller in the case of Runge-Kutta than in the case of Euler.

The RMSE allows discriminating that a run using the Runge-Kutta method with 10 divisions in the interval is almost equivalent to a run using the Euler method with 50 divisions, i.e., with a ratio of 1:5.

The convergence in the accuracy of the approximation is faster in the Runge-Kutta case concerning Euler for this differential equation and this interval. Furthermore, this rapid convergence of the precision makes it possible to appreciate that by dividing 50 sub-intervals, the precision conceptually reaches complete accuracy, assuming Runge-Kutta.

Acknowledgments

The author, CGR, acknowledges ***Laboratorio de Computación Cuántica*** *for dedicating time and lending the facilities during the development of the present work.*

ANNEX I

The syntax of both codes is shown below:

<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<>>>>>>>>>>>>>>>>>>>>>>>

Euler

```
# Programa para evaluar EDO con valor inicial segun Metodo de EULER,
un paso
# Intervalo a usar, h, min = 0  max = 1
# Cantidad de particiones N = 10
# Ecuacion diferencial bajo estudio y´ = 2y
# Condicion inicial (x0,y0) = (0,1)
install.packages("ggplot2")
library("ggplot2")

n <- 10 # Definiendo el numero de particiones del intervalo

a = 0
b = 1
h = (b-a)/n #Definiendo el Intervalo

# Introduciendo la ecuacion diferencial dy/dx = 2y

x <- vector("integer", n)    # 1 output
y <- vector("integer", n)    # 1 output
z <- vector("integer", n)    # 1 output
for (i in 1:n){              # 2 secuencia
  x[1]=0 # condicion inicial para x
  x[i+1] = x[i] + h          # 3 cuerpo
  y[1]=1 # condicion inicial para y
```

```
  y[i+1] = y[i] + h*2*y[i]   # 3 cuerpo
  z[1] = y[1+1]              # 3 cuerpo
  z[i+1] = z[i] + h*2*z[i]   # 3 cuerpo
}

# Construyendo la solucion exacta
# Ecuacion diferencial bajo estudio y´ = 2y
# Solución de la ecuación exacta y = exp(2x)

yy <- vector("integer", n)    # 1 output
for (i in 1:n+1){             # 2 secuencia
  yy[1]=1
  yy[i]=exp(2*x[i])           # 3 cuerpo
}

s1 = 0
s2 = 0
DIF = 0                   # Calculo del RMSE
for (i in 1:n+1){
  s1 = z[i]               # Valor calculado
  s2 = yy[i]              # Valor observado
  DIF = abs(s1-s2)^2
  S = DIF/n
}
RMSE = sqrt(S)
Precision = (1-RMSE)*100

Table <- data.frame(
  "intervalo" = x,
  "yn"= y,
  "yn 1" = z,
  "ya" = yy
)
Table
RMSE
```

```
precision

ggplot(data=Table,      aes(x=x,y=z))+      geom_point(color="red",
size=2)+geom_line(y=yy, color="blue")

<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<>>>>>>>>>>>>>>>>>>>>>>>>
```

Runge-Kutta

```
# Programa para evaluar EDO con valor inicial segun Metodo de Runge-
Kutta, cuarto orden
# Intervalo a usar, h, min = 0  max = 1
# Cantidad de particiones N = 10
# Ecuacion diferencial bajo estudio y´ = 2y
# Condicion inicial (x1,y1) = (0,1)
install.packages("ggplot2")
library("ggplot2")

n <- 10 # Definiendo el numero de particiones de intervalo

a = 0
b = 1
h = (b-a)/n #Definiendo el Intervalo

# Introduciendo la ecuacion diferencial dy/dx = 2y

x <- vector("integer", n)    # 1 output
y <- vector("integer", n)    # 1 output
for (i in 1:n){              # 2 secuencia
  x[1]=0 # condicion inicial para x
  x[i+1] = x[i] + h          # 3 cuerpo
}
k = 0
for (i in 1:n){
  y[1]=1 # condicion inicial para y
```

```
  k1 = 2*h*y[i]            # 3 cuerpo
  k2 = 2*h*(y[i]+k1/2)
  k3 = 2*h*(y[i]+k2/2)
  k4 = 2*h*(y[i]+k3)
  k  = (k1+2*k2+2*k3+k4)/6
  y[i+1]=y[i]+k
}

# Construyendo la solucion exacta
# Ecuacion diferencial bajo estudio y´ = 2y
# Solución de la ecuación exacta y = exp(2x)

yy <- vector("integer", n)    # 1 output
for (i in 1:n+1){             # 2 secuencia
  yy[1]=1
  yy[i]=exp(2*x[i])           # 3 cuerpo
}

s1 = 0
s2 = 0
DIF = 0                   # Calculo del RMSE
for (i in 1:n+1){
  s1 = y[i]               # Valor calculado
  s2 = yy[i]              # Valor observado
  DIF = abs(s1-s2)^2
  S = DIF/n
}
RMSE = sqrt(S)
Precision = (1-RMSE)*100

Table <- data.frame(
  "intervalo" = x,
  "yRK4"= y,
  "ya" = yy
)
```

```
Table
RMSE
precision
ggplot(data=Table,      aes(x=x,y=y))+      geom_point(color="red")+
geom_line(y=yy, color="blue")


<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<>>>>>>>>>>>>>>>>>>>>>>>>>>>>
install.packages("ggplot2")
library("ggplot2")
library("readr")
setwd("C:/Users/usuario/Dropbox/Mi PC (usuario-HP)/Desktop/MNC")
data <- read.table("data.csv", header = TRUE, sep = ",")
View(data)
ggplot(data=data, aes(x=n,y=a))+ geom_point(color="red", size=2)
ggplot(data=data, aes(x=n,y=c))+ geom_point(color="blue", size=2)
ggplot(data=data, aes(x=n,y=b))+ geom_point(color="red", size=2)
ggplot(data=data, aes(x=n,y=d))+ geom_point(color="blue", size=2)
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<>>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

ANNEX II

Calculated values of the approximations for 10 subintervals and 50 subintervals, according to the Euler and Runge-Kutta methods.

Caso Euler

Tabla I. Valor mínimo de la subdivisión n = 10 y Valor máximo de la subdivisión n = 50

n	yn	yn+1	y exactly
1	1.00000	1.20000	1.00000
2	1.20000	1.44000	1.22140
3	1.44000	1.72800	1.49183
4	1.72800	2.07360	1.82212
5	2.07360	2.48832	2.22554
6	2.48832	2.98598	2.71828
7	2.98598	3.58318	3.32012
8	3.58318	4.29982	4.05520
9	4.29982	5.15978	4.95303
10	5.15978	6.19174	6.04965
11	6.19174	7.43008	7.38906
	RMSE	0.01297407	

n	yn	yn+1	y exactly
1	1.00000	1.04000	1.00000
2	1.04000	1.08160	1.04081
3	1.08160	1.12486	1.08329
4	1.12486	1.16986	1.12750
5	1.16986	1.21665	1.17351
6	1.21665	1.26532	1.22140
7	1.26532	1.31593	1.27125
8	1.31593	1.36857	1.32313
9	1.36857	1.42331	1.37713
10	1.42331	1.48024	1.43333
11	1.48024	1.53945	1.49183
12	1.53945	1.60103	1.55271
13	1.60103	1.66507	1.61607
14	1.66507	1.73168	1.68203
15	1.73168	1.80094	1.75067
16	1.80094	1.87298	1.82212
17	1.87298	1.94790	1.89648
18	1.94790	2.02582	1.97388
19	2.02582	2.10685	2.05443
20	2.10685	2.19112	2.13828
21	2.19112	2.27877	2.22554
22	2.27877	2.36992	2.31637
23	2.36992	2.46472	2.41090
24	2.46472	2.56330	2.50929
25	2.56330	2.66584	2.61170
26	2.66584	2.77247	2.71828
27	2.77247	2.88337	2.82922
28	2.88337	2.99870	2.94468

29	2.99870	3.11865	3.06485
30	3.11865	3.24340	3.18993
31	3.24340	3.37313	3.32012
32	3.37313	3.50806	3.45561
33	3.50806	3.64838	3.59664
34	3.64838	3.79432	3.74342
35	3.79432	3.94609	3.89619
36	3.94609	4.10393	4.05520
37	4.10393	4.26809	4.22070
38	4.26809	4.43881	4.39295
39	4.43881	4.61637	4.57223
40	4.61637	4.80102	4.75882
41	4.80102	4.99306	4.95303
42	4.99306	5.19278	5.15517
43	5.19278	5.40050	5.36556
44	5.40050	5.61652	5.58453
45	5.61652	5.84118	5.81244
46	5.84118	6.07482	6.04965
47	6.07482	6.31782	6.29654
48	6.31782	6.57053	6.55351
49	6.57053	6.83335	6.82096
50	6.83335	7.10668	7.09933
51	7.10668	7.39095	7.38906
	RMSE	0.0002679342	

Caso Runge Kutta

Tabla II. Valor mínimo de la subdivisión n = 10 y Valor máximo de la subdivisión n = 50

n	yRK4	y exactly	n	yRK4	y exactly
1	1	1	1	1	1
2	1.221400	1.221403	2	1.040811	1.040811
3	1.491818	1.491825	3	1.083287	1.083287
4	1.822106	1.822119	4	1.127497	1.127497
5	2.225521	2.225541	5	1.173511	1.173511
6	2.718251	2.718282	6	1.221403	1.221403
7	3.320072	3.320117	7	1.271249	1.271249
8	4.055136	4.055200	8	1.323130	1.323130
9	4.952943	4.953032	9	1.377128	1.377128
10	6.049525	6.049647	10	1.433329	1.433329
11	7.388889	7.389056	11	1.491825	1.491825

RMSE	5.27649e-05	13	1.552707	1.552707
		14	1.616074	1.616074
		15	1.682028	1.682028
		16	1.750672	1.750673
		17	1.822119	1.822119
		18	1.896481	1.896481
		19	1.973878	1.973878
		20	2.054433	2.054433
		21	2.138276	2.138276
		22	2.225541	2.225541
		23	2.316367	2.316367
		24	2.410900	2.410900
		25	2.509290	2.509290
		26	2.611696	2.611696
		27	2.718282	2.718282
		28	2.829217	2.829217
		29	2.944679	2.944680
		30	3.064854	3.064854
		31	3.189933	3.189933
		32	3.320117	3.320117
		33	3.455613	3.455613
		34	3.596640	3.596640
		35	3.743421	3.743421
		36	3.896193	3.896193
		37	4.055200	4.055200
		38	4.220696	4.220696
		39	4.392946	4.392946
		40	4.572225	4.572225
		41	4.758821	4.758821
		42	4.953032	4.953032
		43	5.155169	5.155170
		44	5.365556	5.365556
		45	5.584528	5.584528
		46	5.812437	5.812437
		47	6.049647	6.049647
		48	6.296538	6.296538
		49	6.820958	6.820958
		50	7.099327	7.099327
		51	7.389056	7.389056
			RMSE	4.31244e-08