# Taller Euler y runge-kutta

Javier Marin- Andrés Mariño

26 de octubre de 2018

## R Punto 1

Considere un cuerpo con temperatura interna $T$ el cual se encuentra en un ambiente con temperatura constante $Te$ . Suponga que su masa $m$ concentrada en un solo punto. Entonces la transferencia de calor entre el cuerpo y el entorno externo puede ser descrita con la ley de Stefan-Boltzmann:

$$v(t) = \epsilon\gamma S(T\ 4\ (t) - Te\ 4\ )$$

Donde, $t$ es tiempo y $\epsilon$ es la constante de Boltzmann ($\epsilon$ = 5.6$x$10−8 $J/m2K\ 2\ s$), $\gamma$ es la constante de "emisividad" del cuerpo, $S$ el área de la superficie y $v$ es la tasa de transferencia del calor. La tasa de variación de la energía $dT/dt = -v(t)\ mC$ ($C$ indica el calor específico del material que constituye el cuerpo). En consecuencia,

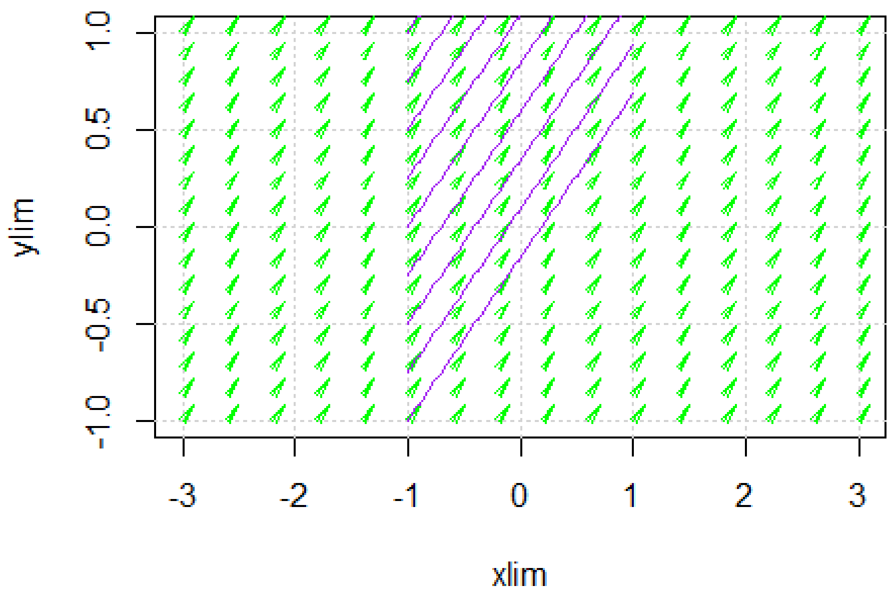
$$dT\ /dt = -\epsilon\gamma S(T\ 4\ (t) - Te\ 4\ /)\ mC$$

Usando el método de Euler (en R) y 20 intervalos iguales y t variando de 0 a 200 segundos, resuelva numéricamente la ecuación, si el cuerpo es un cubo de lados de longitud 1m y masa igual a 1Kg. Asuma, que T0 = 180K, Te = 200K, g = 0.5 y C = 100J/(Kg/K).

lo cual nos dio como resultado la siguiente gráfica con los siguientes puntos utilizando la emisividad del vidrio la cual es 0.94:

```r
library(pracma)
metodoEuler <- function(f, h, xi, yi, xf)
{
  N = (xf - xi) / h
  x = y = numeric(N+1)
  x[1] = xi;
  y[1] = yi;
  i = 1
  while (i <= N)
  {
    x[i+1] = x[i]+h
    y[i+1] = y[i]+(h*f(x[i],y[i]))
    i = i+1
  }
  return (data.frame(X = x, Y = y))
}

f <- function(x, y) {-(((5.6*10^(-8))*0.94*1)*(y^4*(x)-200^4))/100}

e1 = metodoEuler(f, 10, 0, 180, 200)
```

```
e1[nrow(e1),]

##      X        Y
## 21 200 54.29582

print(e1)

##      X         Y
## 1    0 180.00000
## 2   10 188.42240
## 3   20 130.49402
## 4   30 108.38772
## 5   40  95.01509
## 6   50  86.27635
## 7   60  80.11555
## 8   70  75.52624
## 9   80  71.95902
## 10  90  69.09005
## 11 100  66.71750
## 12 110  64.71013
## 13 120  62.97945
## 14 130  61.46400
## 15 140  60.11982
## 16 150  58.91470
## 17 160  57.82444
## 18 170  56.83051
## 19 180  55.91841
## 20 190  55.07662
## 21 200  54.29582

xx <- c(-3, 3); yy <- c(-1, 1)
vectorfield(f, xx, yy, scale = 0.1)
for (xs in seq(-1, 1, by = 0.25))
{
  sol <- rk4(f, -1, 1, xs, 100)
  lines(sol$x, sol$y, col="purple")
}
```




## R Punto 2

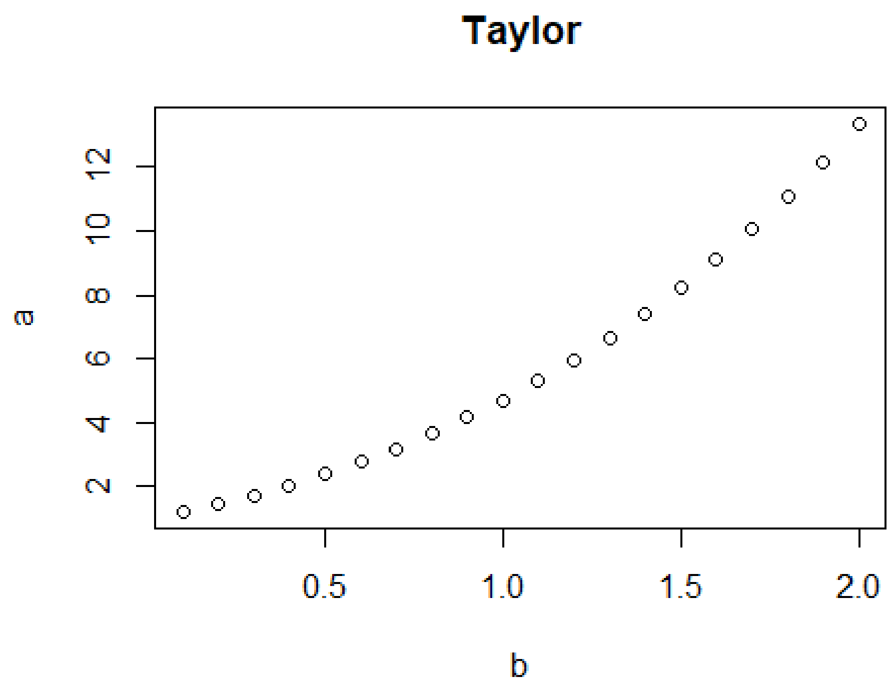
Obtenga cinco puntos de la solución de la ecuación, utilizando el método de Taylor (los tres primeros términos)con h=0.1 implemente en R

$$\frac{dy}{dx} - (x + y) = 1 - x^2; y(0) = 1$$

Grafique su solución y compare con la solución exacta, cuál es el error de truncamiento en cada paso

```
taylor = function(f,df,x,y,h,m){

  a=c()
  b=c()
  u=c()
  v=c()
  for (i in 1:m){
    y=y+h*f(x,y)+h**2/2*df(x,y)
    x=x+h
    u[i]=u+x
    v[i]=v+y
    cat("y ",round(y,6)," ","\n")
    cat("x ",round(x,6)," ")
    a[i]=y
    b[i]=x
```

```
  }
  plot(b,a,main="Taylor")



}

f = function(x,y){

  return (x-x**2+y+1)
}

df = function(x,y){

  return (y-x**2-x+2)
}


taylor(f,df,0,1,0.1,20)

## y  1.215
## x  0.1  y  1.461025
## x  0.2  y  1.739233
## x  0.3  y  2.050902
## x  0.4  y  2.397447
## x  0.5  y  2.780429
## x  0.6  y  3.201574
## x  0.7  y  3.662789
## x  0.8  y  4.166182
## x  0.9  y  4.714081
## x  1  y  5.309059
## x  1.1  y  5.953961
## x  1.2  y  6.651926
## x  1.3  y  7.406429
## x  1.4  y  8.221304
## x  1.5  y  9.100791
## x  1.6  y  10.04957
## x  1.7  y  11.07283
## x  1.8  y  12.17628
## x  1.9  y  13.36623
## x  2
```




## R Punto 3

Obtenga 20 puntos de la solución de la ecuación, utilizando el método de Euler (los tres primeros términos)con h=0.1

$$\frac{dy}{dx} - (x+y) = 1 - x^2; y(0) = 1$$

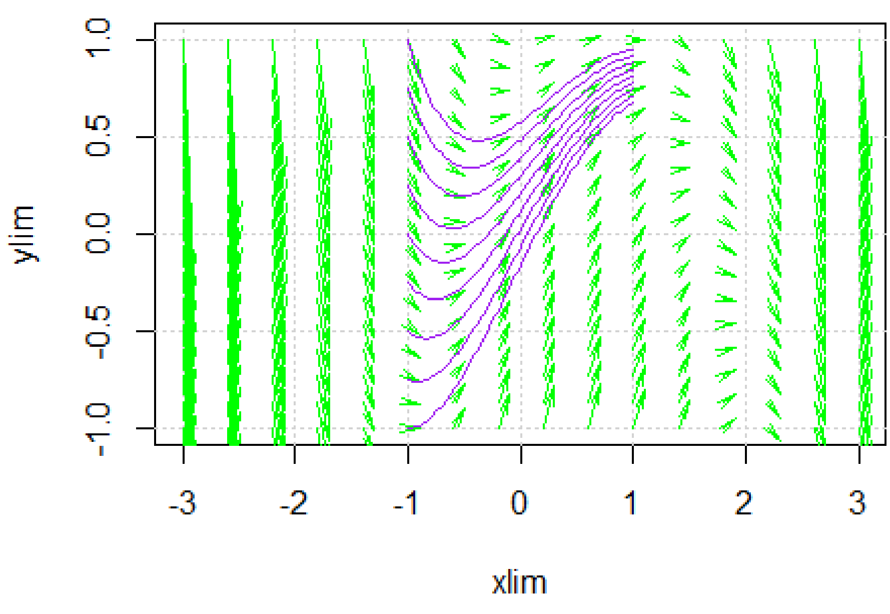
Grafique su solución y compare con la solución exacta, cuál es el error de truncamiento en cada paso

```
library(pracma)

metodoEuler <- function(f, h, xi, yi, xf)
{
  N = (xf - xi) / h
  x = y = numeric(N+1)
  x[1] = xi;
  y[1] = yi;
  i = 1
  while (i <= N)
  {
    x[i+1] = x[i]+h
    y[i+1] = y[i]+(h*f(x[i],y[i]))
    i = i+1
  }
  return (data.frame(X = x, Y = y))
}
```

```
f <- function(x, y) {
  1-(x^2) + x - y
}

e1 = metodoEuler(f, 0.1, 0, 1, 2)

e1[nrow(e1),]
```

```
##    X         Y
## 21 2 0.4525723
```

```
xx <- c(-3, 3); yy <- c(-1, 1)
vectorfield(f, xx, yy, scale = 0.1)
for (xs in seq(-1, 1, by = 0.25))
{
  sol <- rk4(f, -1, 1, xs, 100)
  lines(sol$x, sol$y, col="purple")
}
```




```
print(e1)
```

```
##      X         Y
## 1  0.0 1.0000000
## 2  0.1 1.0000000
## 3  0.2 1.0090000
## 4  0.3 1.0241000
```

```
## 5   0.4 1.0426900
## 6   0.5 1.0624210
## 7   0.6 1.0811789
## 8   0.7 1.0970610
## 9   0.8 1.1083549
## 10  0.9 1.1135194
## 11  1.0 1.1111675
## 12  1.1 1.1000507
## 13  1.2 1.0790457
## 14  1.3 1.0471411
## 15  1.4 1.0034270
## 16  1.5 0.9470843
## 17  1.6 0.8773759
## 18  1.7 0.7936383
## 19  1.8 0.6952744
## 20  1.9 0.5817470
## 21  2.0 0.4525723
```

## R Punto 4

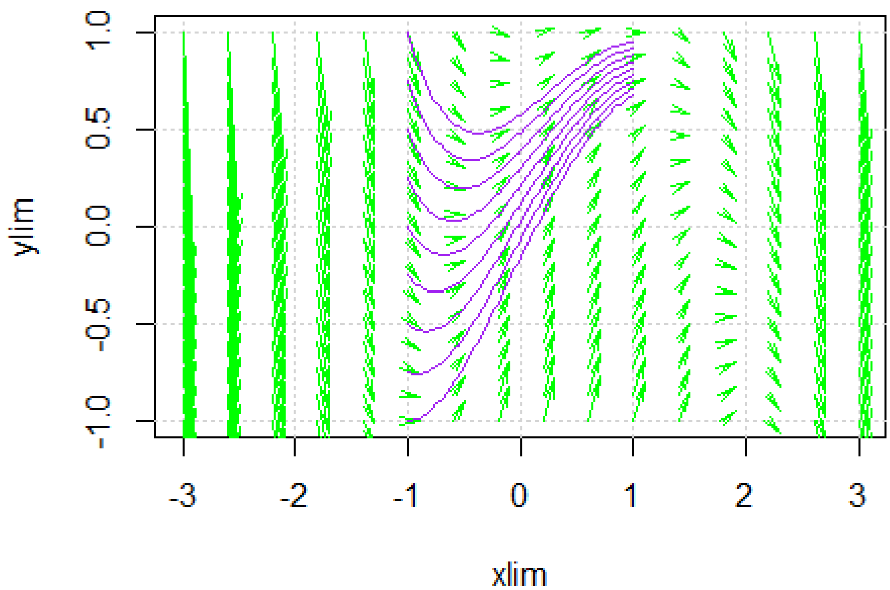
Implemente en R el siguiente algoritmo y aplíquelo para resolver la ecuación anterior

1) Defina f(x,y) y la condición incial $(x_0, y_0)$
2) Defina h y la cantidad de puntos a calcular m
3) Para i =1, 2, ..., m
4) $K_1 = hf(x_i, y_i)$
5) $K_2 = hf(x_i + h, y_i + K_1)$
6) $y_{i+1} = y_i + \frac{1}{2}(K_1 + K_2)$
7) $x_{i+1} = x_i + h$
8) fin

$$\frac{dy}{dx} - (x + y) = 1 - x^2; y(0) = 1$$

```r
f <- function(x, y) {
  1-(x^2) + x - y
}


h = 0.1
m = 20
x <- c(0)
y <- c(1)
for (i in 1:m)
{
  k1 = h*f(x[i],y[i])
  k2 = h*f(x[i]+h,y[i]+k1)
  yi = y[i] + (0.5)*(k1+k2)
  y <- append(y, yi)
  xi = x[i] + h
```

```r
  x <- append (x,xi)
}

xx <- c(-3, 3); yy <- c(-1, 1)
vectorfield(f, xx, yy, scale = 0.1)
for (xs in seq(-1, 1, by = 0.25))
{
  sol <- rk4(f, -1, 1, xs, 100)
  lines(sol$x, sol$y, col="purple")
}
```




```r
print(e1)

##      X         Y
## 1  0.0 1.0000000
## 2  0.1 1.0000000
## 3  0.2 1.0090000
## 4  0.3 1.0241000
## 5  0.4 1.0426900
## 6  0.5 1.0624210
## 7  0.6 1.0811789
## 8  0.7 1.0970610
## 9  0.8 1.1083549
## 10 0.9 1.1135194
## 11 1.0 1.1111675
## 12 1.1 1.1000507
## 13 1.2 1.0790457
```

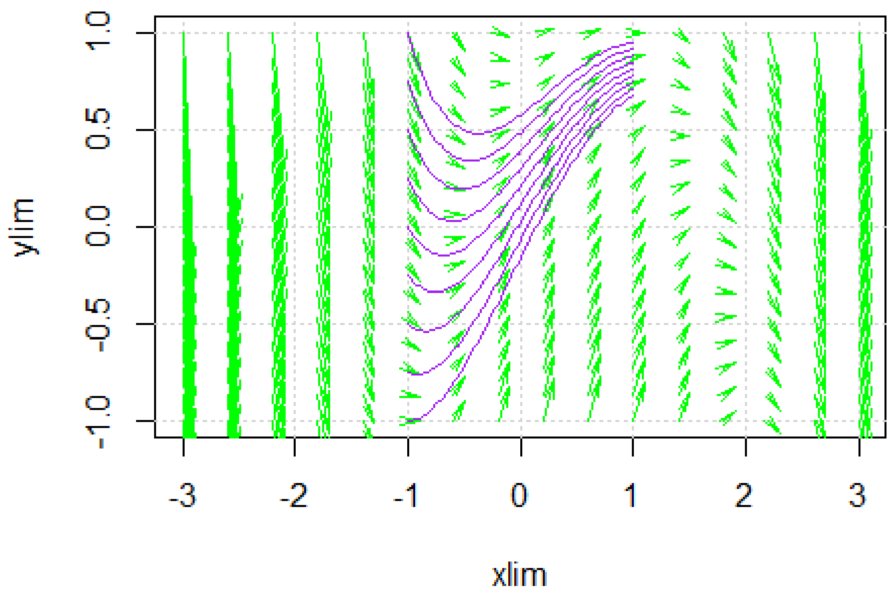
```
## 14 1.3 1.0471411
## 15 1.4 1.0034270
## 16 1.5 0.9470843
## 17 1.6 0.8773759
## 18 1.7 0.7936383
## 19 1.8 0.6952744
## 20 1.9 0.5817470
## 21 2.0 0.4525723
```

## R Punto 5

Utilizar la siguiente variación en el método de Euler, para resolver una ecuación diferencial ordinaria de primer orden, la cual calcula el promedio de las pendientes en cada paso

$$y_{i+1} = y_i + \frac{h}{2}\left(f(x_i, y_i) + f(x_{i+1}, y_{i+1})\right)$$

```r
library(pracma)
f <- function(x,y)
{
  x-y+1-(x^2)
}

x <- rnorm(10)
y <- rnorm(10)

m = 10
h = 0.1
yi <- c()
for (i in 1:m-1)
{
  l <- y[i] + (h/2)*( f(x[i],y[i]) + f(x[i+1],y[i+1]) )
  yi <- append (yi,l)
}

xx <- c(-3, 3); yy <- c(-1, 1)
vectorfield(f, xx, yy, scale = 0.1)
for (xs in seq(-1, 1, by = 0.25))
{
  sol <- rk4(f, -1, 1, xs, 100)
  lines(sol$x, sol$y, col="purple")
}
```




```
print (sol)

## $x
##   [1] -1.00 -0.98 -0.96 -0.94 -0.92 -0.90 -0.88 -0.86 -0.84 -0.82 -
0.80
##  [12] -0.78 -0.76 -0.74 -0.72 -0.70 -0.68 -0.66 -0.64 -0.62 -0.60 -
0.58
##  [23] -0.56 -0.54 -0.52 -0.50 -0.48 -0.46 -0.44 -0.42 -0.40 -0.38 -
0.36
##  [34] -0.34 -0.32 -0.30 -0.28 -0.26 -0.24 -0.22 -0.20 -0.18 -0.16 -
0.14
##  [45] -0.12 -0.10 -0.08 -0.06 -0.04 -0.02  0.00  0.02  0.04  0.06
0.08
##  [56]  0.10  0.12  0.14  0.16  0.18  0.20  0.22  0.24  0.26  0.28
0.30
##  [67]  0.32  0.34  0.36  0.38  0.40  0.42  0.44  0.46  0.48  0.50
0.52
##  [78]  0.54  0.56  0.58  0.60  0.62  0.64  0.66  0.68  0.70  0.72
0.74
##  [89]  0.76  0.78  0.80  0.82  0.84  0.86  0.88  0.90  0.92  0.94
0.96
## [100]  0.98  1.00
##
## $y
##   [1] 1.0000000 0.9609907 0.9239261 0.8887517 0.8554144 0.8238619
0.7940431
##   [8] 0.7659076 0.7394065 0.7144915 0.6911153 0.6692316 0.6487950
```

```
0.6297611
##  [15] 0.6120862 0.5957275 0.5806433 0.5667923 0.5541343 0.5426299
0.5322403
##  [22] 0.5229277 0.5146549 0.5073855 0.5010837 0.4957146 0.4912438
0.4876378
##  [29] 0.4848634 0.4828886 0.4816815 0.4812111 0.4814470 0.4823593
0.4839189
##  [36] 0.4860971 0.4888658 0.4921974 0.4960650 0.5004421 0.5053028
0.5106216
##  [43] 0.5163737 0.5225346 0.5290804 0.5359876 0.5432333 0.5507948
0.5586502
##  [50] 0.5667777 0.5751561 0.5837646 0.5925828 0.6015907 0.6107687
0.6200976
##  [57] 0.6295586 0.6391332 0.6488033 0.6585512 0.6683595 0.6782112
0.6880895
##  [64] 0.6979782 0.7078611 0.7177226 0.7275471 0.7373197 0.7470254
0.7566499
##  [71] 0.7661787 0.7755981 0.7848943 0.7940539 0.8030638 0.8119111
0.8205832
##  [78] 0.8290677 0.8373525 0.8454257 0.8532756 0.8608909 0.8682603
0.8753729
##  [85] 0.8822178 0.8887847 0.8950630 0.9010428 0.9067140 0.9120670
0.9170922
##  [92] 0.9217803 0.9261220 0.9301084 0.9337307 0.9369803 0.9398487
0.9423276
##  [99] 0.9444089 0.9460847 0.9473470
```

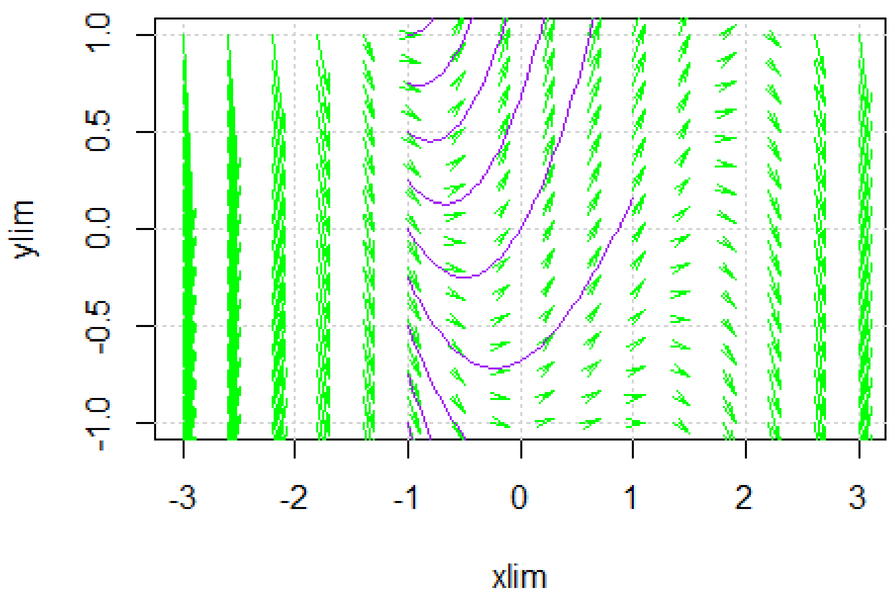
## R Punto 6

Implemente un código en R, para este método y obtenga 10 puntos de la solución con h=0.1, grafíquela y compárela con el método de Euler:

$$\frac{dy}{dx} - x\text{-}y - 1 + x^2 = 0; y(0) = 1$$

```
library(pracma)
f <- function(x,y)
{
  x+y+1-x^2
}

x <- rnorm(10)
y <- rnorm(10)

m = 10
h = 0.1
yi <- c()
for (i in 1:m-1)
```

```
{
  l <- y[i] + (h/2)*( f(x[i],y[i]) + f(x[i+1],y[i+1]) )
  yi <- append (yi,l)
}
xx <- c(-3, 3); yy <- c(-1, 1)
vectorfield(f, xx, yy, scale = 0.1)
for (xs in seq(-1, 1, by = 0.25))
{
  sol <- rk4(f, -1, 1, xs, 100)
  lines(sol$x, sol$y, col="purple")
}
```




```
print (sol)

## $x
##   [1] -1.00 -0.98 -0.96 -0.94 -0.92 -0.90 -0.88 -0.86 -0.84 -0.82 -
0.80
##  [12] -0.78 -0.76 -0.74 -0.72 -0.70 -0.68 -0.66 -0.64 -0.62 -0.60 -
0.58
##  [23] -0.56 -0.54 -0.52 -0.50 -0.48 -0.46 -0.44 -0.42 -0.40 -0.38 -
0.36
##  [34] -0.34 -0.32 -0.30 -0.28 -0.26 -0.24 -0.22 -0.20 -0.18 -0.16 -
0.14
##  [45] -0.12 -0.10 -0.08 -0.06 -0.04 -0.02  0.00  0.02  0.04  0.06 
0.08
##  [56]  0.10  0.12  0.14  0.16  0.18  0.20  0.22  0.24  0.26  0.28 
0.30
```

```
##  [67]  0.32  0.34  0.36  0.38  0.40  0.42  0.44  0.46  0.48  0.50
0.52
##  [78]  0.54  0.56  0.58  0.60  0.62  0.64  0.66  0.68  0.70  0.72
0.74
##  [89]  0.76  0.78  0.80  0.82  0.84  0.86  0.88  0.90  0.92  0.94
0.96
## [100]  0.98  1.00
##
## $y
##   [1] 1.000000 1.000601 1.002411 1.005437 1.009687 1.015171 1.021897
##   [8] 1.029874 1.039111 1.049617 1.061403 1.074477 1.088849 1.104530
##  [15] 1.121530 1.139859 1.159528 1.180548 1.202929 1.226685 1.251825
##  [22] 1.278362 1.306307 1.335674 1.366474 1.398721 1.432428 1.467607
##  [29] 1.504272 1.542438 1.582119 1.623328 1.666081 1.710392 1.756278
##  [36] 1.803753 1.852833 1.903536 1.955876 2.009872 2.065541 2.122900
##  [43] 2.181967 2.242761 2.305300 2.369603 2.435690 2.503581 2.573296
##  [50] 2.644856 2.718282 2.793595 2.870817 2.949971 3.031080 3.114166
##  [57] 3.199254 3.286368 3.375533 3.466774 3.560117 3.655588 3.753213
##  [64] 3.853021 3.955040 4.059297 4.165821 4.274643 4.385793 4.499302
##  [71] 4.615200 4.733520 4.854296 4.977560 5.103346 5.231689 5.362625
##  [78] 5.496190 5.632421 5.771356 5.913032 6.057490 6.204769 6.354911
##  [85] 6.507956 6.663947 6.822928 6.984943 7.150037 7.318256 7.489647
##  [92] 7.664258 7.842138 8.023337 8.207905 8.395894 8.587358 8.782351
##  [99] 8.980927 9.183143 9.389056
```

## R Punto 7

7. Pruebe el siguiente código en R del método de Runge Kutta de tercer y cuarto orden y obtenga 10 puntos de la solución con h=0.1, grafiquela y compárela con el método de Euler:

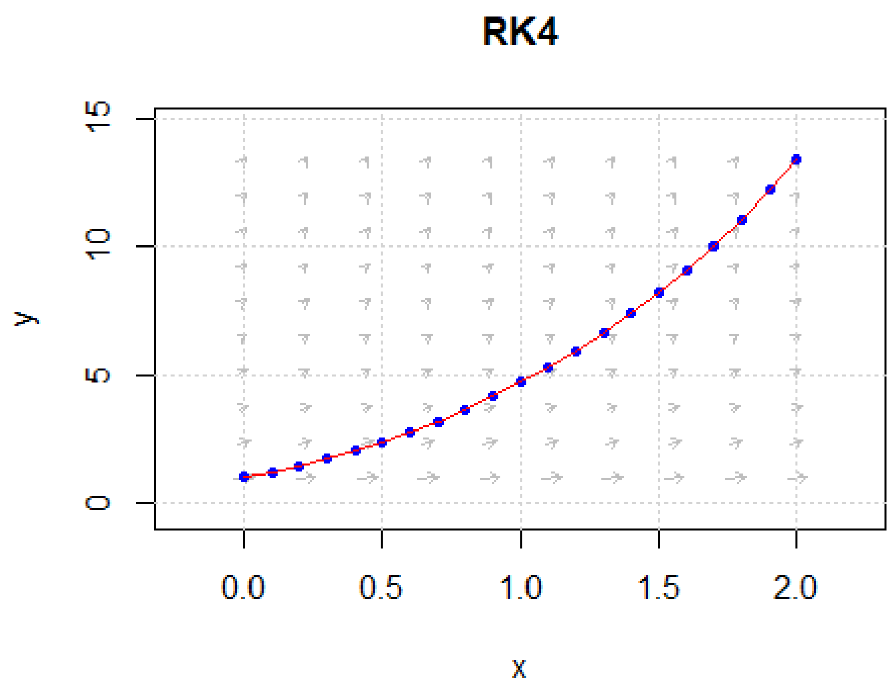
$$\frac{dy}{dx} - x - y - 1 + x^2 = 0; y(0) = 1$$

```
library(phaseR)

f<-function(fcn,x,y){
  return(eval(fcn))
}

# Solo para prueba con dy=x+y, y(0)=1
obtenerErrorAbsoluto<-function(x,y){
  solucion=exp(x)*((-x*exp(-x))-exp(-x)+2)
  return(abs(y-solucion))
}

graficarCampoPendiente<-function(x0, xn, y0, yn, fcn, numpendientes,
metodo){
  apma1 <- function(t, y, parameters){
    a <- parameters[1]
    dy <- a*(f(fcn, t, y))
```

```
    list(dy)
  }
  apma1.flowField <- flowField(apma1, x = c(x0, xn),
                               y    = c(y0, yn), parameters = c(1),
                               points = numpendientes, system =
"one.dim",
                               add = FALSE, xlab = "x", ylab = "y",
                               main = metodo)
  grid()
}

graficarSolucionNumerica<-function (x, y){
  points (x, y, pch=20, col="blue")
  for (i in 2:length(x)){
    segments(x[i-1], y[i-1], x[i], y[i], col="red")
  }
}

rk4<-function(dy, ti, tf, y0, h, graficar=TRUE, numpendientes=10){
  t<-seq(ti, tf, h)
  y<-c(y0)
  cat("x     |y          |k1         |k2         |k3         |k4        |error
absoluto\n")
  for(i in 2:length(t)){
    k1=h*f(dy, t[i-1], y[i-1])
    k2=h*f(dy, t[i-1]+h/2, y[i-1]+k1*(0.5))
    k3=h*f(dy, t[i-1]+h/2, y[i-1]+k2*(0.5))
    k4=h*f(dy, t[i-1]+h, y[i-1]+k3)
    y<-c(y, y[i-1]+1/6*(k1+2*k2+2*k3+k4))
    cat(t[i-1]," | ", y[i-1]," | ",k1," | ",k2," | ",k3," | ",k4," |
",obtenerErrorAbsoluto(t[i-1],y[i-1]),"\n")
  }
  if (graficar){
    graficarCampoPendiente(min(t), max(t), min(y), max(y), dy,
numpendientes, "RK4")
    graficarSolucionNumerica(t, y)
  }
  rta<-list(w=y, t=t)
}

rk3<-function(dy, ti, tf, y0, h, graficar=TRUE, numpendientes=10){
  t<-seq(ti, tf, h)
  y<-c(y0)
  cat("x     |y          |k1          |k2         |k3         |error
absoluto\n")
  for(i in 2:length(t)){
    k1=h*f(dy, t[i-1], y[i-1])
    k2=h*f(dy, t[i-1]+h/2, y[i-1]+k1*(0.5))
    k3=h*f(dy, t[i-1]+h, y[i-1]-k1+2*k2)
    y<-c(y, y[i-1]+1/6*(k1+4*k2+k3))
```

```
    cat(t[i-1]," | ", y[i-1]," | ",k1," | ",k2," | ",k3," |
",obtenerErrorAbsoluto(t[i-1],y[i-1]),"\n")
  }
  if (graficar){
    graficarCampoPendiente(min(t), max(t), min(y), max(y), dy,
numpendientes, "RK3")
    graficarSolucionNumerica(t, y)
  }
  rta<-list(w=y, t=t)
}

r<-rk4(expression(x+y+1-x^2), 0, 2, 1, 0.1)

## x     |y          |k1         |k2         |k3         |k4        |error
absoluto
## 0  |  1  |  0.2  |  0.21475  |  0.2154875  |  0.2305488  |  0
## 0.1  |  1.215171  |  0.2305171  |  0.2457929  |  0.2465567  |
0.2621727  |  0.1048288
## 0.2  |  1.461402  |  0.2621402  |  0.2779972  |  0.2787901  |
0.2950192  |  0.2185966
## 0.3  |  1.739858  |  0.2949858  |  0.3114851  |  0.31231  |  0.3292168
|  0.3401402
## 0.4  |  2.051823  |  0.3291823  |  0.3463914  |  0.3472519  |
0.3649075  |  0.4681739
## 0.5  |  2.398719  |  0.3648719  |  0.3828655  |  0.3837652  |
0.4022485  |  0.6012768
## 0.6  |  2.782116  |  0.4022116  |  0.4210722  |  0.4220152  |
0.4414132  |  0.7378787
## 0.7  |  3.20375  |  0.441375  |  0.4611937  |  0.4621846  |  0.4825934
|  0.8762442
## 0.8  |  3.665537  |  0.4825537  |  0.5034314  |  0.5044753  |
0.5260012  |  1.014455
## 0.9  |  4.169599  |  0.5259599  |  0.5480078  |  0.5491102  |
0.5718709  |  1.150392
## 1  |  4.718276  |  0.5718276  |  0.595169  |  0.5963361  |  0.6204612
|  1.281713
## 1.1  |  5.31416  |  0.620416  |  0.6451867  |  0.6464253  |  0.6720585
|  1.405827
## 1.2  |  5.960109  |  0.6720109  |  0.6983615  |  0.699679  |
0.7269788  |  1.519875
## 1.3  |  6.659288  |  0.7269288  |  0.7550252  |  0.75643  |  0.7855718
|  1.620694
## 1.4  |  7.41519  |  0.785519  |  0.8155449  |  0.8170462  |  0.8482236
|  1.70479
## 1.5  |  8.231677  |  0.8481677  |  0.8803261  |  0.881934  |
0.9153611  |  1.768299
## 1.6  |  9.113019  |  0.9153019  |  0.9498169  |  0.9515427  |
0.9874561  |  1.806954
## 1.7  |  10.06393  |  0.9873931  |  1.024513  |  1.026369  |  1.06503
|  1.816037
```

```
## 1.8  |  11.08963  |  1.064963  |  1.104961  |  1.106961  |  1.148659
|  1.790334
## 1.9  |  12.19587  |  1.148587  |  1.191767  |  1.193926  |  1.23898  |
1.724085
```




```
r2<-rk3(expression(x+y+1-x^2), 0, 2, 1, 0.1)

## x     |y          |k1           |k2          |k3         |error absoluto
## 0  |  1  |  0.2  |  0.21475  |  0.23195  |  0
## 0.1  |  1.215158  |  0.2305158  |  0.2457916  |  0.2636226  |
0.1048165
## 0.2  |  1.461376  |  0.2621376  |  0.2779945  |  0.2965227  |
0.2185703
## 0.3  |  1.739816  |  0.2949816  |  0.3114806  |  0.3307795  |
0.3400979
## 0.4  |  2.051763  |  0.3291763  |  0.3463851  |  0.3665357  |
0.4681134
## 0.5  |  2.398638  |  0.3648638  |  0.382857  |  0.4039488  |
0.6011956
## 0.6  |  2.782012  |  0.4022012  |  0.4210612  |  0.4431933  |
0.737774
## 0.7  |  3.203618  |  0.4413618  |  0.4611799  |  0.4844616  |
0.8761127
## 0.8  |  3.665375  |  0.4825375  |  0.5034144  |  0.5279667  |
1.014293
## 0.9  |  4.169402  |  0.5259402  |  0.5479872  |  0.5739437  |
1.150196
```

```
## 1   |  4.718041  |  0.5718041  |  0.5951443  |  0.6226526  |  1.281477
## 1.1  |   5.31388  |  0.620388  |  0.6451574  |  0.6743807  |  1.405548
## 1.2  |   5.95978  |  0.671978  |  0.6983269  |  0.7294456  |  1.519546
## 1.3  |   6.658902  |  0.7268902  |  0.7549847  |  0.7881981  |
1.620308
## 1.4  |   7.41474  |  0.785474  |  0.8154976  |  0.8510261  |  1.70434
## 1.5  |   8.231155  |  0.8481155  |  0.8802712  |  0.9183582  |
1.767776
## 1.6  |   9.112414  |  0.9152414  |  0.9497535  |  0.990668  |  1.80635
## 1.7  |   10.06323  |  0.9873235  |  1.02444  |  1.068479  |  1.81534
## 1.8  |   11.08883  |  1.064883  |  1.104877  |  1.15237  |  1.789534
## 1.9  |   12.19496  |  1.148496  |  1.19167  |  1.24298  |  1.723166
```