Daniel Andrés Mendoza

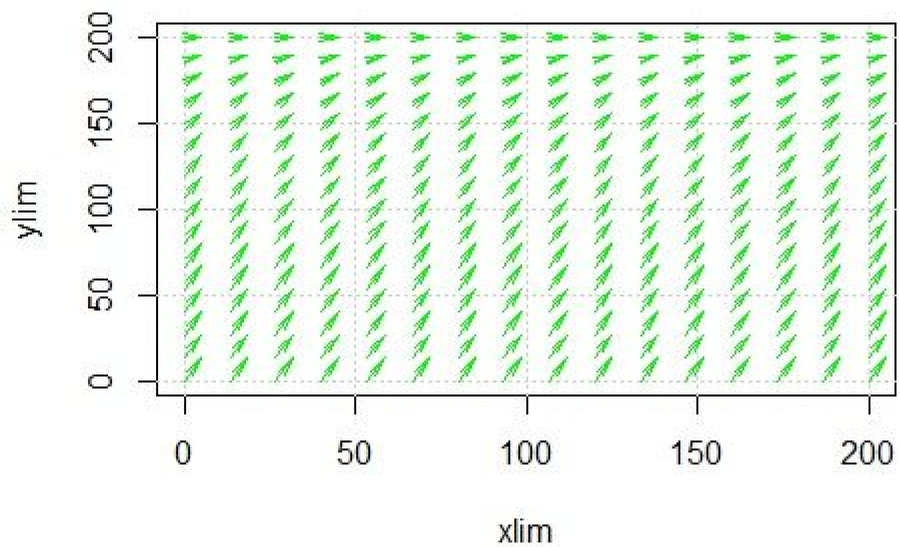
Camilo Serrano

Taller EDO

1.

Al aplicar el método de Euler para solucionar $\frac{dT}{dt} = \frac{-\epsilon\gamma S(T\ 4\ (t) - Te\ 4\ )}{mC}$ solo se grafica el campo de pendientes, mas no la solución.

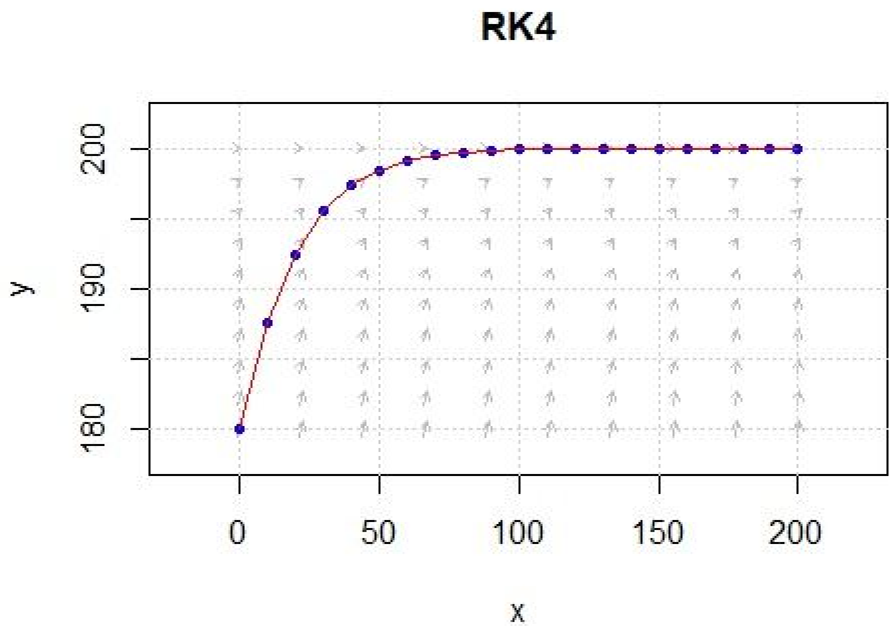
```
metodoEuler <- function(f, h, xi, yi, xf)
{
  N = (xf - xi) / h
  x = y = numeric(N+1)
  x[1] = xi;
  y[1] = yi;
  i = 1
  while (i <= N)
  {
    x[i+1] = x[i]+h
    y[i+1] = y[i]+(h*f(x[i],y[i]))
    i = i+1
  }
  return (data.frame(X = x, Y = y))
}
```



Por otro lado, el método de runge kutta nos da la siguiente solución:

```
rk4<-function(dy, ti, tf, yo, h, graficar=TRUE, numpendientes=10){
  t<-seq(ti, tf, h)
  y<-c(yo)
  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)
}
```



3.

Ejecutar el método de Euler imprimiendo los valores, obtenemos el siguiente resultado.

```
metodoEuler <- function(dy, 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
    cat("y(",x[i],") = ", y[i],"\n")
    #print(x[i])
    #print(y[i])
  }
  return (data.frame(X = x, Y = y))
}
```

```
y( 0.1 ) =  1
y( 0.2 ) =  1.01
y( 0.3 ) =  1.03
y( 0.4 ) =  1.06
y( 0.5 ) =  1.1
y( 0.6 ) =  1.15
y( 0.7 ) =  1.21
y( 0.8 ) =  1.28
y( 0.9 ) =  1.36
y( 1 ) =  1.45
y( 1.1 ) =  1.55
y( 1.2 ) =  1.66
y( 1.3 ) =  1.78
y( 1.4 ) =  1.91
y( 1.5 ) =  2.05
y( 1.6 ) =  2.2
y( 1.7 ) =  2.36
y( 1.8 ) =  2.53
y( 1.9 ) =  2.71
y( 2 ) =  2.9
```

4.

La implementación del método de Euler quedó así:

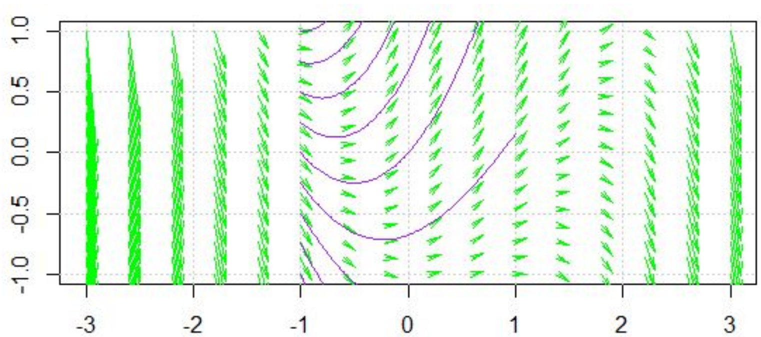
```
metodoEuler <- function(dy, h, xi, yi, xf,m)
{
  i = 1
  while (i <= m)
  {
    k1 = h*dy(xi,yi)
```

```
    k2 = h*dy(xi+h,yi+k1)

    yi = yi + (k1+k2)/2

    xi = xi + h

    cat("y(",xi,") = ", yi,"\n")

    #print(x[i])

    #print(y[i])

    i = i +1

  }

}
```

5.

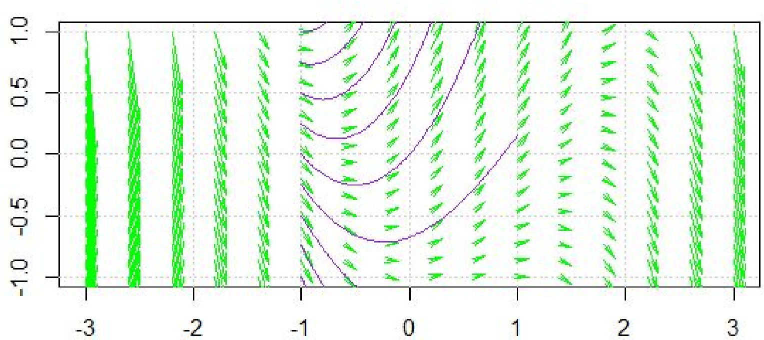
Al correr el método de Euler modificado se obtienen los siguientes valores:

```
y( 0.1 ) =  1.2145
y( 0.2 ) =  1.459973
y( 0.3 ) =  1.73757
y( 0.4 ) =  2.048564
y( 0.5 ) =  2.394364
y( 0.6 ) =  2.776522
y( 0.7 ) =  3.196757
y( 0.8 ) =  3.656966
y( 0.9 ) =  4.159248
y( 1 ) =  4.705919
```



Comparado con los valores de Euler original:

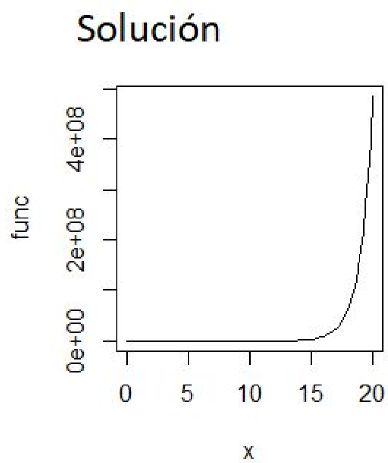
```
x=  0   , y= 1
x=  0.1   , y= 1.2
x=  0.2   , y= 1.429
x=  0.3   , y= 1.6879
x=  0.4   , y= 1.97769
x=  0.5   , y= 2.299459
x=  0.6   , y= 2.654405
x=  0.7   , y= 3.043845
x=  0.8   , y= 3.46923
x=  0.9   , y= 3.932153
```

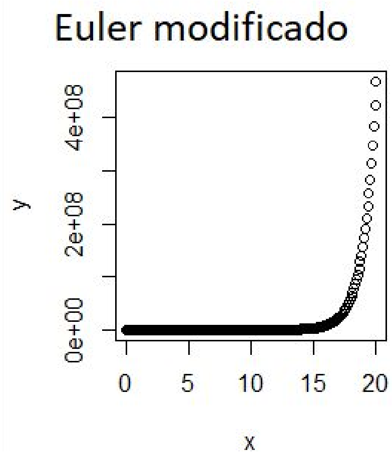


La respuesta a la ecuación diferencial es:

$y = e^x(e^2e^{-x} + xe^{-x} + 1)$

El método modificado se asemeja más a la respuesta, en el m[etodo original el error de y es más grande a medida que x avanza.

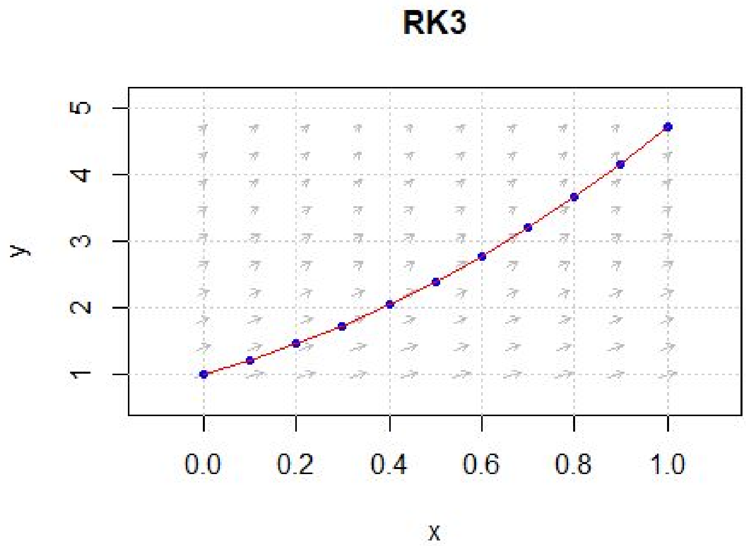




7.

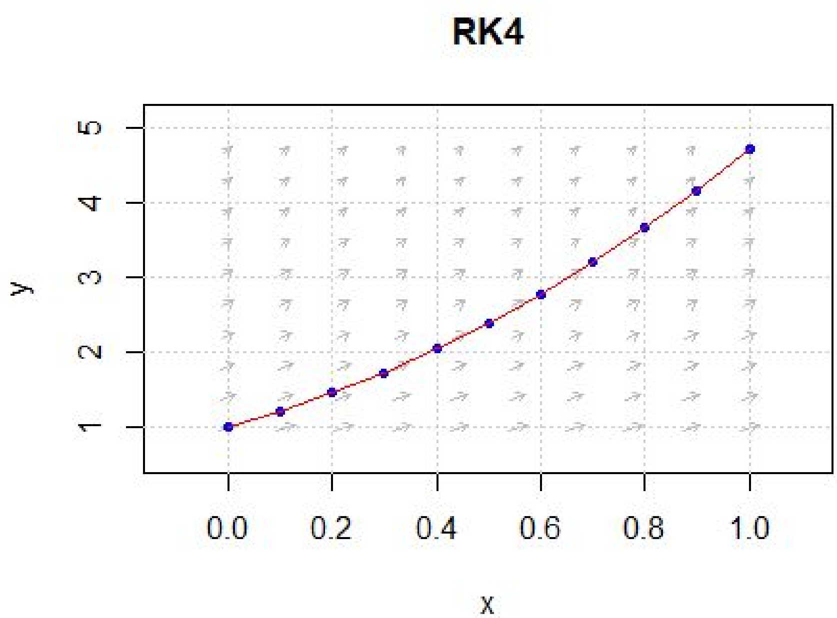
Aplicando el método de Runge Kutta de tercer orden se obtienen los siguientes datos y la siguiente gráfica:

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



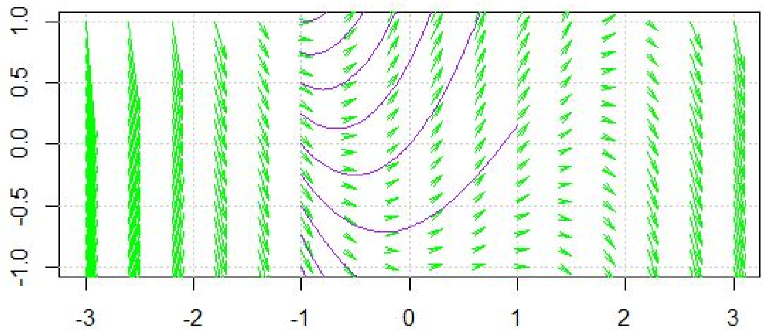
Aplicando el método de Runge Kutta de cuarto orden se obtienen los siguientes datos y la siguiente gráfica:

```
> r<-rk4(expression(-x^2+x+y+1), 0, 1, 1, 0.1)
x     |y          |k1         |k2          |k3          |k4        |error absoluto
0   |  1   |  0.2  |  0.21475  |  0.2154875  |  0.2305487  |  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
```



Aplicando el método de Euler:

```
x=  0   , y= 1
x=  0.1  , y= 1.2
x=  0.2  , y= 1.429
x=  0.3  , y= 1.6879
x=  0.4  , y= 1.97769
x=  0.5  , y= 2.299459
x=  0.6  , y= 2.654405
x=  0.7  , y= 3.043845
x=  0.8  , y= 3.46923
x=  0.9  , y= 3.932153
```



En ambos casos, el método de Runge Kutta fue más preciso que el método de Euler.