PONTIFICIA UNIVERSIDAD JAVERIANA
SISTEMAS DE ECUACIONES DIFERENCIALES

ANÁLISIS NUMÉRICO

PROFESORA: EDDY HERRERA DAZA

AUTORES
PABLO ANDRES MILLAN ARIAS
OSCAR GIOVANNI FONSECA NEIRA

BOGOTÁ 13 DE NOVIEMBRE DE 2018

ENTREGA FINAL PROYECTO DE ANÁLISIS NUMÉRICO

Para el proyecto de Análisis Numérico se construyó una librería en RStudio llamada *SistemasMillanFonseca*, con el fin de que tenga la capacidad para resolver sistemas de ecuaciones diferenciales utilizando distintos métodos de solución.

Modo de uso/instalación

1. Abrir la aplicación de RStudio
2. En la esquina inferior derecha del programa, ir a la opción packages y luego ir a la opción install.
3. Donde se encuentran las opciones para instalación, cambiar de install from Repository (CRAN) a Package Archive.
4. Seleccionar un archivo con extensión .zip o .tar gz, de lo contrario no cargará el paquete. Una vez hecho esto, ya se habrá instalado el paquete.
5. 5. Ejecutar el comando install para phaseR (Paquete requeridos para hacer las respectivas operaciones)
6. Ejecutar cualquiera de las funciones diseñadas para el paquete (Se puede ayudar del archivo *prueba.r* para hacer pruebas de las funciones del paquete).

Funciones

Con respecto a las funciones para cumplir el objetivo de la librería, podremos encontrar dentro de la librería las siguiente funciones:

ODE_Sys

Es la función principal y es la que debe ser implementada por el usuario, ya que a partir de de esta función se realiza llamadas a las demás funciones. Para ver más sobre la función, puede ingresar dentro de la librería el comando *?ODE_Sys* para ver la documentación de la función.

Solución

Función que resuelve el sistema de ecuaciones diferenciales y retorna su correspondiente solución dependiendo del método que se desea. Para ver más sobre la función, puede ingresar dentro de la librería el comando *?Solucion* para ver la documentación de la función.

Error

Función que calcula el error de la solución obtenida en la anterior función, comparando dicha solución con la solución analítica del sistema. Para ver más sobre la función, puede ingresar dentro de la librería el comando *?error* para ver la documentación de la función.

PlotField

Función que grafica el campo y la solución correspondiente del sistema calculado en la función *Solución,* únicamente funciona para sistemas con dos ecuaciones. Para ver más sobre la función, puede ingresar dentro de la librería el comando *?plotfield* para ver la

documentación de la función. Así mismo dentro de esta función también se encuentra ubicada la función ***Field***, para ver en qué consiste puede usar el comando *?Field.*

Métodos de solución

Ahora bien, los métodos que se utilizaron para resolver los sistemas de ecuaciones son los siguientes.

Método de Euler

El método de Euler es un procedimiento que permite construir aproximaciones a las soluciones de un problema con valor inicial con una ecuacion diferencial ordinaria de primer orden.

Método de Mid Point

El método del punto medio consiste en aproximar la derivada por la fórmula de dos puntos

$$y'(x) \sim \frac{y(x+h)-y(x-h)}{2h}$$

con lo que obtenemos la siguiente relación de recurrencia

$$y(x+h) = y(x-h) + 2hf(x, y(x))$$

Método de Runge-Kutta de 4 orden (RK4)

Es otro método que podemos utilizar para encontrar soluciones numéricas a Ecuaciones diferenciales, y que incluso es más preciso que el método de Euler. En el cual la relación de recurrencia va a estar dada por un promedio ponderado de términos.

Pruebas de funcionalidad de la librería

A continuación se mostrará un ejemplo sobre cómo funciona la librería con el siguiente sistema de ecuaciones diferenciales:

$$"-x+y+1" \text{ y } "x-y"$$

Para hallar los errores correspondientes, se conoce la solución analítica del sistema, que es:

$$x = \tfrac{1}{4}(2t - 9e^{-2t} - 3) \text{ y } y = \tfrac{1}{4}(2t + 9e^{-2t} - 5)$$

PRIMER EJEMPLO

Condiciones

- Método RK4
- Puntos iniciales $x(0) = -3$ y $y(0) = 1$
- $h = 0.1$
- 500 puntos
- $t = 0.62$

Representación en RStudio

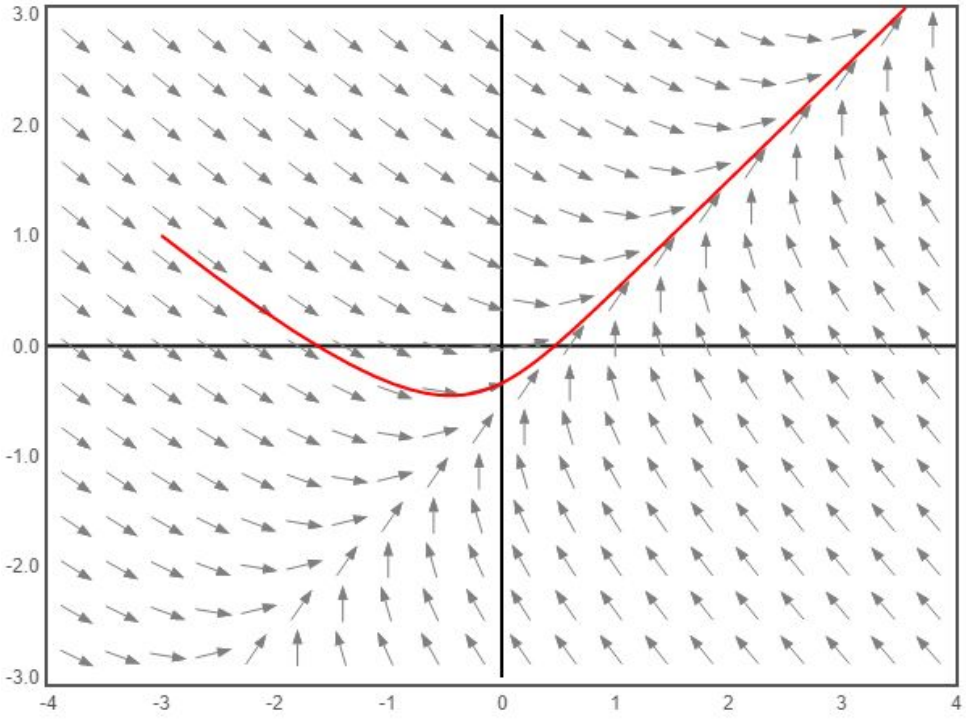
```
sol = ODE_Sys(c("-x+y+1","x-y"),c("x","y","t"),c(-3,1,0),0.1,c(-4,4,-3,3),500,"rk4",0.62)
```

Solución y errores

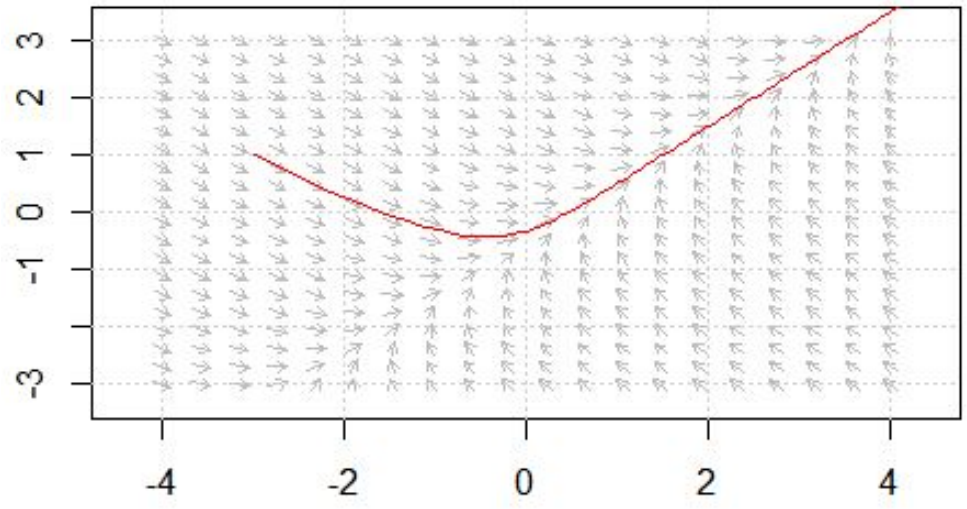
```
$answer
[1] -1.1167201 -0.2382827  0.6000000

$`errores en x y y`
[1] 0.01096689 0.03403033
```

Gráfica en Bluffton



Gráfica obtenida con RK4



SEGUNDO EJEMPLO

Condiciones

- Método de Euler
- Puntos iniciales $x(0) = -3$ y $y(0) = 1$
- $h = 0.01$
- 700 puntos
- $t = 0.9$

Representación en RStudio

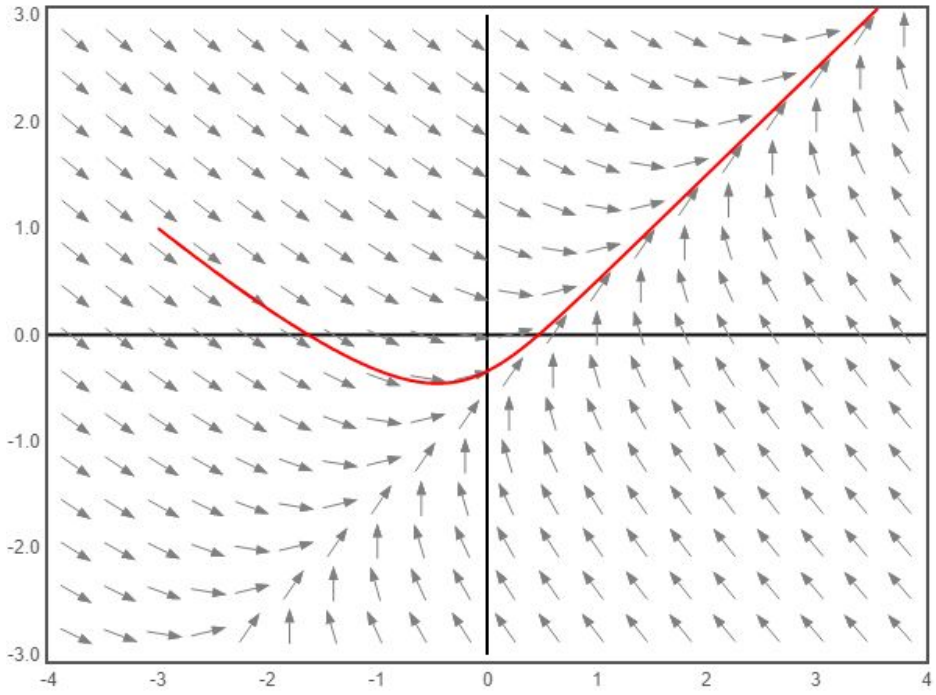
```
sol = ODE_Sys(c("-x+y+1","x-y"),c("x","y","t"),c(-3,1,0),0.01,c(-4,4,-3,3),700,"euler",0.9)
```

Solución y errores

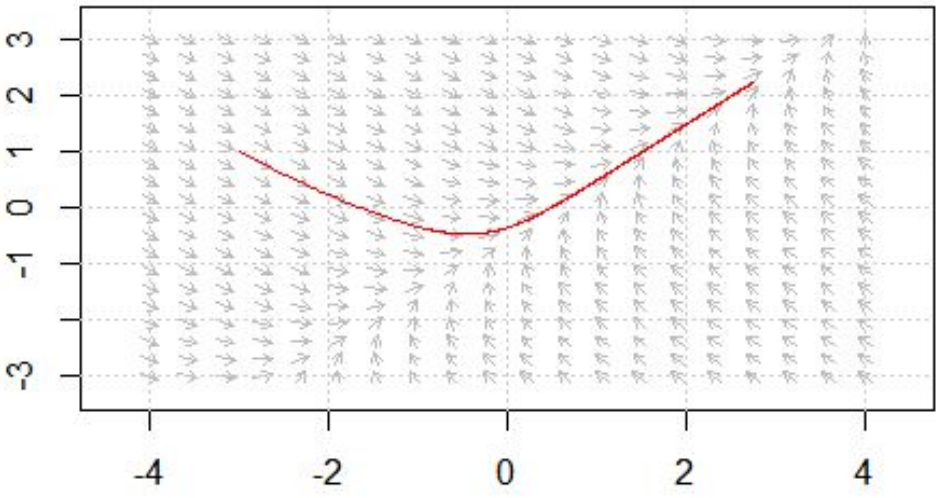
```
$answer
[1] -0.6651988 -0.4348012  0.9000000

$`errores en x y y`
[1] 0.006723708 0.006723708
```

Gráfica en Bluffton



Gráfica obtenida con RK4



TERCER EJEMPLO

Condiciones

- Método de Midpoint
- Puntos iniciales $x(0) = -4$ y $y(0) = 1$
- $h = 0.001$
- 400 puntos
- $t = 0.1$

Representación en RStudio

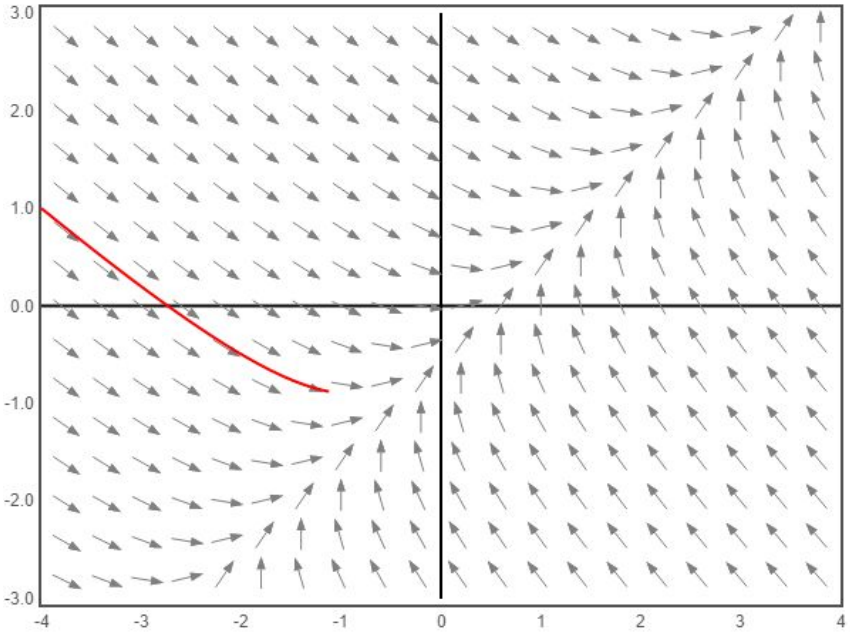
```
sol = ODE_Sys(c("-x+y+1","x-y"),c("x","y","t"),c(-4,1,0),0.001,c(-4,4,-3,3),400,"midpoint",0.1)
```

Solución y errores

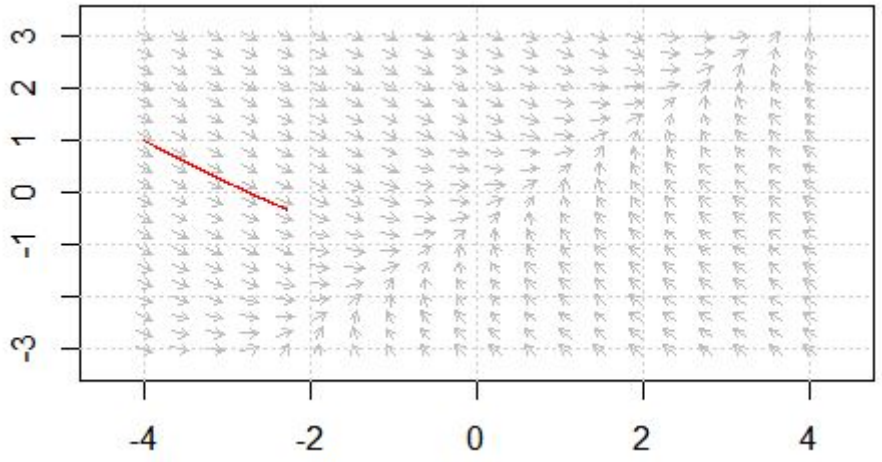
```
$answer
[1] -3.4510587  0.5510587  0.1000000

$`errores en x y y`
[1] 0.90891452 0.09108548
```

Gráfica en Bluffton



Gráfica obtenida con Midpoint



Todos los ejemplos anteriores se encuentran en el archivo *prueba.R,* solo descomente el que quiera implementar y ejecutar.

Conclusión

Comparando las gráficas obtenidas con las de Bluffton y viendo los errores obtenidos, se concluye el método más eficiente es de Runge-Kutta de orden 4 (RK4).