

UADE



FACULTAD DE INGENIERIA Y CIENCIAS EXACTAS

Modelado y Simulación

Trabajo Práctico

Curso: 3.1.016/25	Aula: A706	Turno: noche
Alumno	Franco Daniel Timpone	LU:1052849
Profesor	Fernando Acero	
Fecha: 12/05/2020	Cuatrimestre: primero	

Indice

3	___	Introducción
3	___	Implementación
3	___	Compilación
3	___	Ejecución
4	___	Aplicación
4	___	Menú: Inicio -> Graficar
4	___	Menú: Inicio -> Opciones
5	___	Menú: Inicio -> Ayuda
5	___	Menú: Agregar Gráfico -> Función Simple
6	___	Menú: Agregar Gráfico -> Problema de Valor Inicial -> Método de Euler
6	___	Menú: Agregar Gráfico -> Problema de Valor Inicial -> Método Euler Mejorado
7	___	
		Menú: Agregar Gráfico -> Problema de Valor Inicial -> Método de Runge - Kutta
7	___	Menú: Agregar Gráfico -> Integración Numérica -> Trapecios
8	___	Menú: Agregar Gráfico -> Integración Numérica -> Montecarlo
9	___	Como escribir operaciones matemáticas en la aplicación
9	___	Constantes
9	___	Operaciones simples
10	___	Operaciones Avanzadas
11	___	Graficando elementos
13	___	Utilidades durante la graficación
14	___	Ejemplos de Problema de Valor Inicial
16	___	Ejemplos de Integración Numérica

Introducción

La aplicación entregada es una implementación de los métodos de Problema de Valor Inicial e Integración Numérica vistos durante la primera clase de la cursada.

Implementación

La aplicación fue escrita en Java 8, y para graficar las funciones en pantalla se usó la librería libGDX.

Compilación

Para la facilidad de la prueba la aplicación ya se entrega compilada y lista para ejecutarse, para correrlo solo se requiere [Java 8](#) (o mayor), pero indico abajo como compilarlo a modo informativo.

Para compilar la aplicación se requiere de [Apache Maven](#) y [Java Development Kit 8](#) (o mayor). Con ambos productos instalados alcanza con pararse dentro de la carpeta del proyecto (done se encuentra el archivo **pom.xml**) y lanzar el siguiente comando:

```
mvn install
```

Si la compilación es exitosa saldrá el siguiente mensaje:

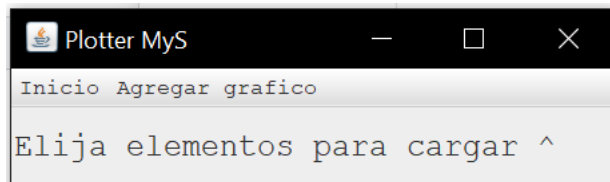
```
[INFO] Reactor Summary for Plotter Parent 1.0-SNAPSHOT:
[INFO]
[INFO] Plotter Parent ..... SUCCESS
[INFO] Plotter Core ..... SUCCESS
[INFO] Plotter desktop ..... SUCCESS
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
```

La primera instalación va a demorar por que debe descargar las librerías y dependencias.

Con ese “BUILD SUCCESS” tengo la confianza de que la compilación fue exitosa. Y puedo encontrar el compilado en la ruta `plotter/desktop/target`, con el nombre **plotter-desktop-1.0-SNAPSHOT-jar-with-dependencies.jar**.

Ejecución

En el entregable ya se encuentra el compilado **plotter.jar**, que es el resultado final de la compilación explicada anteriormente. Para correrlo alcanza con darle doble click y nos abrirá una ventana:



A continuación se detalla la funcionalidad de cada uno de los menús.

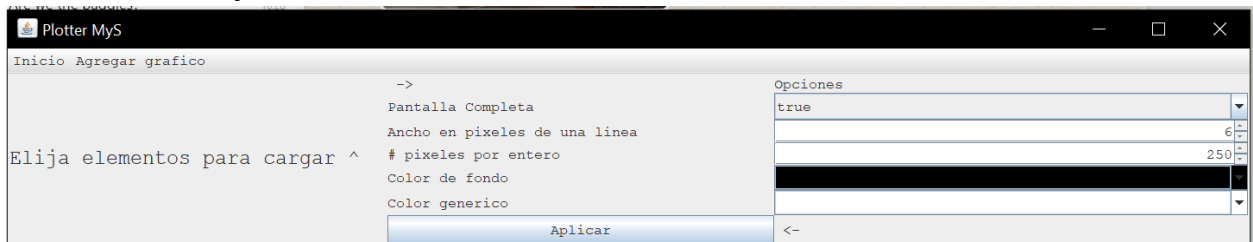
Aplicación

La aplicación inicialmente funciona como un graficador de curvas y figuras en dos dimensiones, entre las que pueden ser curvas definidas como funciones u el resultado de los métodos aprendidos en la materia. Uno carga los elementos que desea ver graficados y una vez que esté conforme con la elección le da al botón Inicio -> Graficar y la herramienta las mostrará por pantalla.

Menú: Inicio -> Graficar

Iniciará al graficación de los componentes

Menú: Inicio -> Opciones

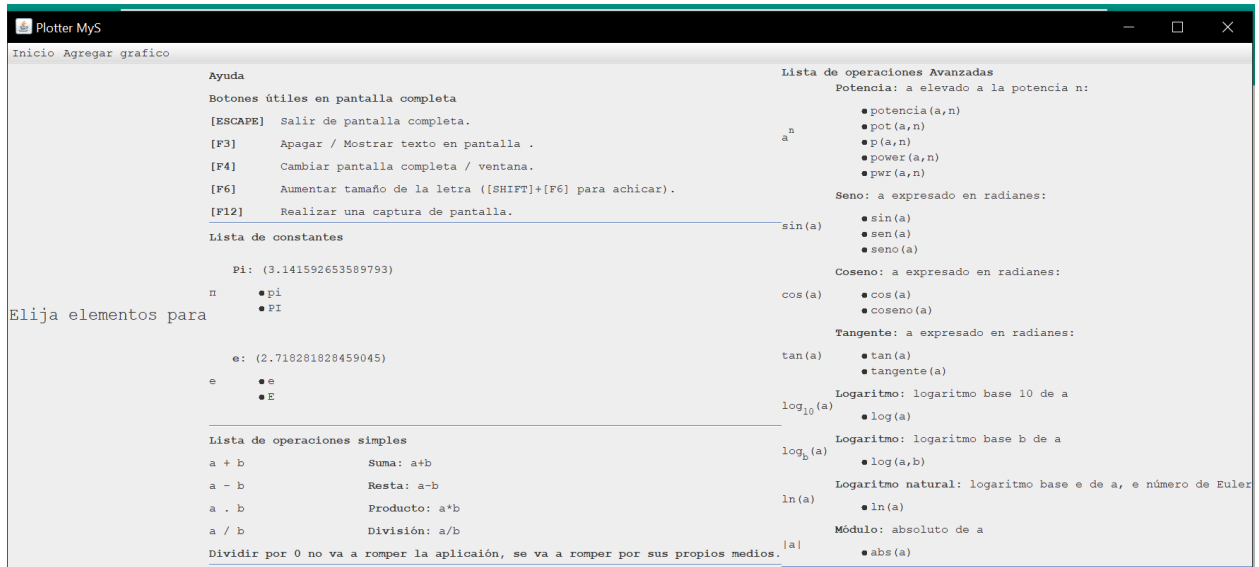


Abrirá la sub pantalla de opciones de la ventana.

Las opciones que pueden elegirse son:

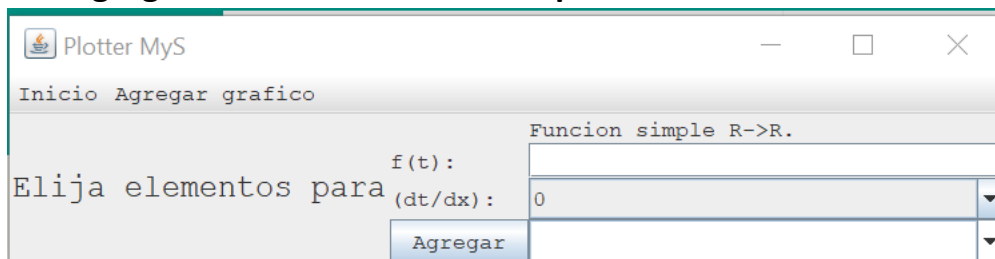
- Pantalla completa: Si la aplicación va a funcionar a pantalla completa al momento de graficar.
- Ancho en pixeles de una línea: El grosor de las líneas dibujadas en pantalla
- # pixeles por entero: Cuantos pixeles de la pantalla habrá entre cada punto del dominio. Ej: Si mi pantalla tiene una resolución horizontal de 1920 pixeles y elijo que se muestren 192 píxeles por entero entonces en mi pantalla tendrá 10 enteros de ancho, o dicho en otras palabras, mi eje horizontal mostrará el intervalo de -5 a 5. Si no se conoce la resolución de la pantalla durante la graficación se muestra esta información.
- Color de fondo: El color que hay debajo de lo graficado.
- Color genérico: El color que toman los ejes y el texto en pantalla.

Menú: Inicio -> Ayuda



Abrirá la sub pantalla de ayuda, donde se recuerda la [función de las teclas especiales](#) y [como usar operaciones matemáticas dentro de las funciones](#).

Menú: Agregar Gráfico -> Función Simple



Abrirá una sub pantalla para agregar funciones simples de R->R

- $f(t)$ es una función simple de la forma $x(t)$.
- dx/dt es la cantidad de veces que debe derivarse la función antes de graficarse.
- A la derecha del botón “Agregar” está el color con el que se dibujará la función.
- El botón Agregar validará únicamente que los campos no estén vacíos, y se agregará la función a la bolsa de “Elementos a cargar”

Menú: Agregar Gráfico -> Problema de Valor Inicial -> Método de Euler

Metodo de Euler. Solo ingrese h o N, no ambos.	
f(t,x):	<input type="text"/>
t ₀	<input type="text"/>
x ₀	<input type="text"/>
T:	<input type="text"/>
h:	<input type="text"/>
N:	<input type="text"/>
Agregar	<input type="text"/>

Abrirá una sub pantalla para aplicar el método de Euler a una función.

- $f(t,x)$ es una función que es derivada de una primitiva que no conocemos y deseamos aproximar.
- t_0 es el punto desde el que aproximo a la primitiva.
- x_0 es el resultado de evaluar la función primitiva en el punto t_0 .
- T es el punto final donde aproximaré a la primitiva.
- h es el tamaño de los intervalos entre los cuales calculo busco aproximaciones. No puede usarse con N .
- N es la cantidad de intervalos que tomaré entre t_0 y T . No puede usarse con h .
- A la derecha del botón “Agregar” se encuentra el color con el que se graficará la primitiva aproximada que se encontró.
- El botón Agregar validará únicamente que los campos no estén vacíos y que no estén h y N cargados al mismo tiempo. Al pulsarlo se agregará la función a la bolsa de “Elementos a cargar”.

Menú: Agregar Gráfico -> Problema de Valor Inicial -> Método Euler Mejorado

Metodo de Euler Mejorado. Solo ingrese h o N, no ambos.	
f(t,x):	<input type="text"/>
t ₀	<input type="text"/>
x ₀	<input type="text"/>
T:	<input type="text"/>
h:	<input type="text"/>
N:	<input type="text"/>
Agregar	<input type="text"/>

Abrirá una sub pantalla para aplicar el método de Euler Mejorado a una función.

- $f(t,x)$ es una función que es derivada de una primitiva que no conocemos y deseamos aproximar.
- t_0 es el punto desde el que aproximo a la primitiva.
- x_0 es el resultado de evaluar la función primitiva en el punto t_0 .
- T es el punto final donde aproximaré a la primitiva.
- h es el tamaño de los intervalos entre los cuales calculo busco aproximaciones. No puede usarse con N .
- N es la cantidad de intervalos que tomaré entre t_0 y T . No puede usarse con h .
- A la derecha del botón “Agregar” se encuentra el color con el que se graficará la primitiva aproximada que se encontró.
- El botón Agregar validará únicamente que los campos no estén vacíos y que no estén h y N cargados al mismo tiempo. Al pulsarlo se agregará la función a la bolsa de “Elementos a cargar”.

Menú: Agregar Gráfico -> Problema de Valor Inicial -> Método de Runge - Kutta

Metodo de Runge Kutta de 4to orden. Solo ingrese h o N, no ambos.

f(t,x):	
t ₀ :	
x ₀ :	
T:	
h:	
N:	
Agregar	

Abrirá una sub pantalla para aplicar el método de Runge Kutta de cuarto orden a una función.

- $f(t,x)$ es una función que es derivada de una primitiva que no conocemos y deseamos aproximar.
- t_0 es el punto desde el que aproximo a la primitiva.
- x_0 es el resultado de evaluar la función primitiva en el punto t_0 .
- T es el punto final donde aproximaré a la primitiva.
- h es el tamaño de los intervalos entre los cuales calculo busco aproximaciones. No puede usarse con N .
- N es la cantidad de intervalos que tomaré entre t_0 y T . No puede usarse con h .
- A la derecha del botón "Agregar" se encuentra el color con el que se graficará la primitiva aproximada que se encontró.
- El botón Agregar validará únicamente que los campos no estén vacíos y que no estén h y N cargados al mismo tiempo. Al pulsarlo se agregará la función a la bolsa de "Elementos a cargar".

Menú: Agregar Gráfico -> Integración Numérica -> Trapecios

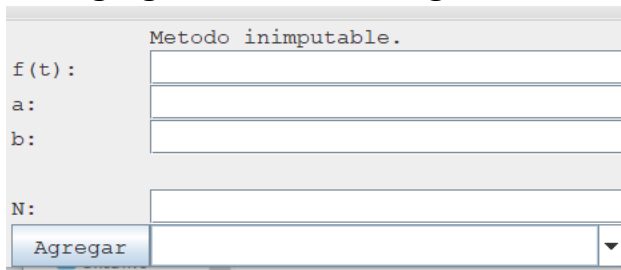
Metodo por Trapezoides. Solo ingrese h o N, no ambos.

f(t):	
a:	
b:	
h:	
N:	
Agregar	

Abrirá una sub pantalla para aplicar el método de integración numérica por trapecios a una función.

- $f(t)$ es una función de la que deseamos aproximar su área.
- a es el punto desde el que aproximo.
- b es el punto final donde aproximo.
- h es el tamaño de los intervalos entre los cuales calculo busco aproximaciones. No puede usarse con N .
- N es la cantidad de intervalos que tomaré entre a y b . No puede usarse con h .
- A la derecha del botón "Agregar" se encuentra el color con el que se graficarán los trapecios.
- El botón Agregar validará únicamente que los campos no estén vacíos y que no estén h y N cargados al mismo tiempo. Al pulsarlo se agregará la función a la bolsa de "Elementos a cargar".

Menú: Agregar Gráfico -> Integración Numérica -> Montecarlo



Abrirá una sub pantalla para aplicar el método de integración numérica de Montecarlo trapecios a una función.

- $f(t)$ es una función de la que deseamos aproximar su área.
- a es el punto desde el que aproximo.
- b es el punto final donde aproximo.
- N es la cantidad de puntos que evaluaré aleatoriamente entre a y b .
- A la derecha del botón “Agregar” se encuentra el color con el que se graficarán los trapecios.
- El botón Agregar validará únicamente que los campos no estén vacíos y que no estén h y N cargados al mismo tiempo. Al pulsarlo se agregará la función a la bolsa de “Elementos a cargar”.

Como escribir operaciones matemáticas en la aplicación

Preámbulo:

- Usar la coma (",") llevará a un error interno, que concluirá que no haya graficación del elemento que la posea.
- Los números con decimales deben expresarse con punto (".").
- Las operaciones pueden separarse entre paréntesis.
- Las operaciones se hacen por términos, y se respeta que la suma y la resta separa términos.

Constantes

- Pi
 - Longitud de la circunferencia de un círculo de radio (3.14159265358979323846):
 - "pi"
 - "PI"
- E
 - Numero de Euler, base del logaritmo natural (2.7182818284590452354):
 - "e"
 - "E"
- Números
 - Cualquier combinación de números con y sin decimales, o una de las constantes anteriores.
 - No llevan espacios.
 - Ej: "12.25", "3", "pi", ".13"

Operaciones simples

Suma

- Se expresa como dos constantes, dos operaciones, o una combinación de ambas separadas por el carácter +.
- Ej.: "2+3", "1+3+2"

Resta

- Se expresa como dos constantes, dos operaciones, o una combinación de ambas separadas por el carácter -.
- Ej.: "2-3", "1-3+2"

Producto

- Se expresa como dos constantes, dos operaciones, o una combinación de ambas separadas por el carácter *.
- No puede hacer producto sin el conector * entre los términos.
- Ej.: "2*3", "1-3*2"

División

- Se expresa como dos constantes, dos operaciones, o una combinación de ambas separadas por el carácter /.
- La aplicación no va a romperse por dividir por 0, va a romperse por sus propios medios.
- Ej.: "2/3", "1-3/2"

Operaciones Avanzadas

Son funciones internas que reciben parámetros que pueden ser constantes u operaciones.

Potencia (a^n)

- Para elevar el parámetro **a** a la potencia **n**, debo consumir la función potencia, que me retornará el resultado.
- Se puede llamar a esta función de las siguientes maneras:
 - `potencia(a,n)`
 - `pot(a,n)`
 - `p(a,n)`
 - `power(a,n)`
 - `pwr(a,n)`
- Ej.: “`p(2,3)`”, “`potencia(2*7,21-3)`”

Seno

- `sin(a)` -> El parámetro **a** es un ángulo expresado en radianes.
- Se puede llamar a esta función de las siguientes maneras:
 - `sin(a)`
 - `sen(a)`
 - `seno(a)`
- Ej.: “`sen(0)`”, “`sin(pi)`”, “`seno(pi/2)`”

Coseno

- `cos(a)` -> El parámetro **a** es un ángulo expresado en radianes.
- Se puede llamar a esta función de las siguientes maneras:
 - `cos(a)`
 - `cosen(a)`
- Ej.: “`cos(0)`”, “`coseno(pi/2)`”

Tangente

- `tan(a)` -> El parámetro **a** es un ángulo expresado en radianes.
- Se puede llamar a esta función de las siguientes maneras:
 - `tan(a)`
 - `tangente(a)`
- Ej.: “`tan(0)`”, “`tangente(pi/2)`”

Logaritmo base 10

- `log10(a)`
- Se puede llamar a esta función de las siguiente manera:
 - `log(a)`
- Ej.: “`log(10)`”

Logaritmo base b

- `logb(a)`
- Se puede llamar a esta función de las siguiente manera:
 - `log(a,b)`
- Ej.: “`log(4,2)`”

Logaritmo natural

- $\ln(a)$
- Se puede llamar a esta función de la siguiente manera:
 - $\ln(a)$
- Ej.: “ $\ln(0.158)$ ”, “ $\ln(0.158+e)$ ”

Modulo

- $|a|$ -> El valor positivo de a
- Se puede llamar a esta función de la siguiente manera:
 - $\text{abs}(a)$
- Ej.: “ $\text{abs}(58)$ ”

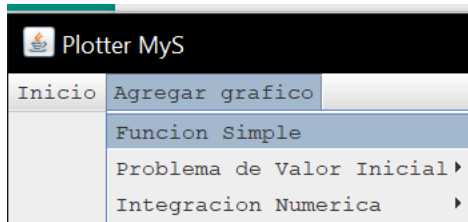
Graficando elementos

Supongamos que quiero graficar la siguiente función:

$$f(t)=e^{-t^2}$$

Dentro de la aplicación voy a elegir en el menú

Agregar gráfico -> Función Simple



La función anterior expresada en operaciones que reconoce la aplicación será:

“ $p(e,-p(t,2))$ ”

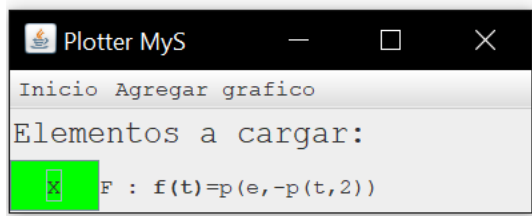
Donde t es la variable de la función.



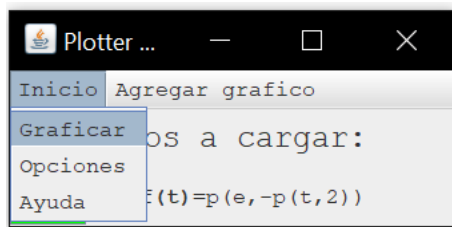
Y le elijo un color para diferenciarla una vez se grafique.

Una vez esté conforme con lo que cargué voy a darle al botón “Agregar”

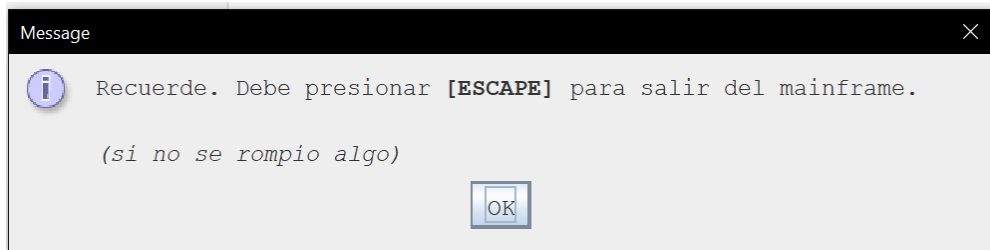
Luego de apretado el botón vamos a apreciar que esta nueva función se agregó a la bolsa de “Elementos a cargar”. Si quiero removerla puedo darle al botón con la ‘X’ para quitarla. Si no quito estos elementos seguirán guardados sin importar que se cierre la aplicación.



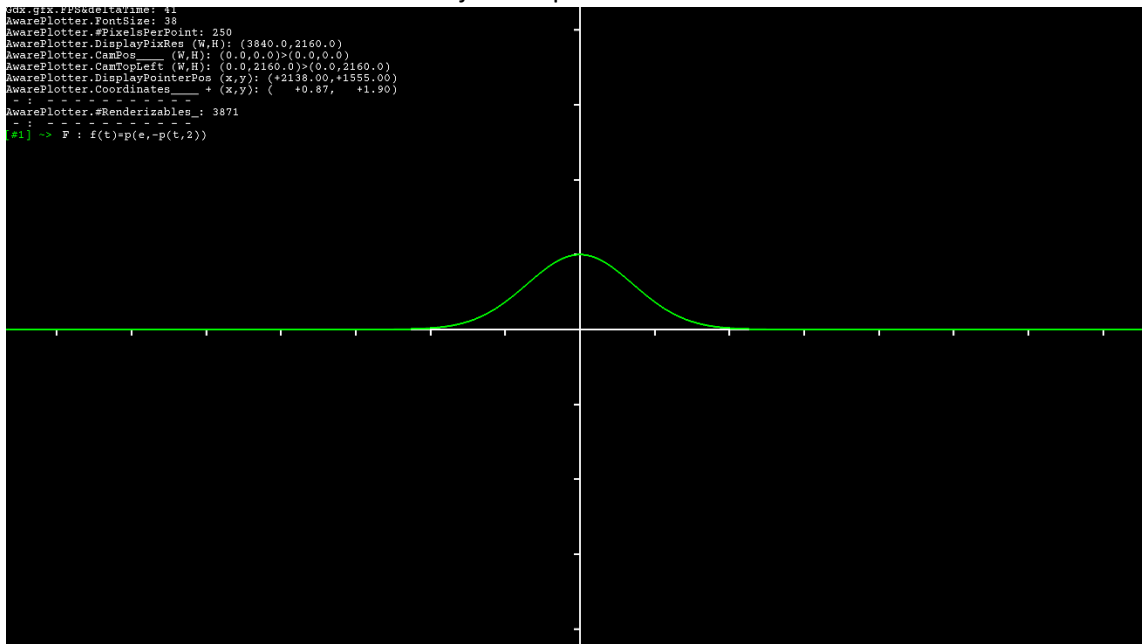
Ahora para ver nuestra creación graficada le damos a Inicio > Graficar



Y nos recibirá este cartel, que es un amable recordatorio.



Le damos a OK. Y vemos como se dibuja en la pantalla nuestra función en el mainframe.



Para salir, como indicó el cartel, presionamos la tecla **[ESCAPE]**.

Utilidades durante la graficación

Mientras estamos frente a los gráficos podemos apreciar que en la parte superior izquierda de la pantalla tenemos texto sobre la pantalla.

```
Gdx.gfx.FPS&deltaTime: 41
AwarePlotter.FontSize: 38
AwarePlotter.#PixelsPerPoint: 250
AwarePlotter.DisplayPixRes (W,H): (3840.0,2160.0)
AwarePlotter.CamPos_____ (W,H): (0.0,0.0)>(0.0,0.0)
AwarePlotter.CamTopLeft (W,H): (0.0,2160.0)>(0.0,2160.0)
AwarePlotter.DisplayPointerPos (x,y): (+2138.00,+1555.00)
AwarePlotter.Coordinates_____ + (x,y): (    +0.87,    +1.90)
- : - - - - -
AwarePlotter.#Renderizables_: 3871
- : - - - - -
[#1] ~> F : f(t)=p(e,-p(t,2))
```

De estos elementos nos interesan...

FontSize: Es el tamaño de la tipografía en pantalla, puedo aumentarlo con la tecla **[F6]** y disminuirlo con **[SHIFT] + [F6]**.

#PixelsPerPoint: Es la cantidad de pixeles por cada entero del gráfico.

DisplayPixRes: Es la resolución del monitor donde estoy graficando.

Coordinates: Me indica sobre qué punto del gráfico se encuentra parado el mouse.

Otros botones útiles

[ESCAPE]: Termina el programa.

[F3]: Activa y desactiva el texto en pantalla.

[F4]: Cambia entre el modo pantalla completa y ventana.

[F12]: Guarda lo graficado en una foto en la ruta:

%USERPROFILE%\plotter\screenshots para Windows

~/plotter/screenshots para UNIX

(En mi caso que mi equipo es Windows y mi nombre de usuario es “FTimpone” la ruta donde me guarda las capturas es C:\Users\FTimpone\plotter\screenshots)

Ejemplos de Problema de Valor Inicial

Supongamos que quiero graficar que tan aproximan los métodos de Euler, Euler Mejorado y Runge-Kutta a una primitiva manteniendo los mismos parámetros.

Función original $x(t)=\ln(t)$ (los métodos no la van a conocer, pero la agrego para que se vea en pantalla la original)

Parámetros: $dx/dt=1/t$; $x(t_0)=0$; $t_0=1$; $N=6$; $T=7$

Cargo función original en Azul (Agregar gráfico -> Función Simple).

Funcion simple R->R.	
f(t):	ln(t)
(dt/dx):	0
Agregar	java.awt.Color[r=0,g=0,b=255]

Cargo la aproximación por Euler en Amarillo (Agregar gráfico -> Problema Valor Inicial -> Método de Euler)

Metodo de Euler. Solo ingrese h o N, no ambos.	
f(t,x):	1/t
t_0	1
x_0	0
T:	7
h:	
N:	6
Agregar	

Cargo la aproximación por Euler Mejorado en Rojo (Agregar gráfico -> Problema Valor Inicial -> Método de Euler Mejorado)

Metodo de Euler Mejorado. Solo ingrese h o N, no ...	
f(t,x):	1/t
t_0	1
x_0	0
T:	7
h:	
N:	6
Agr...	

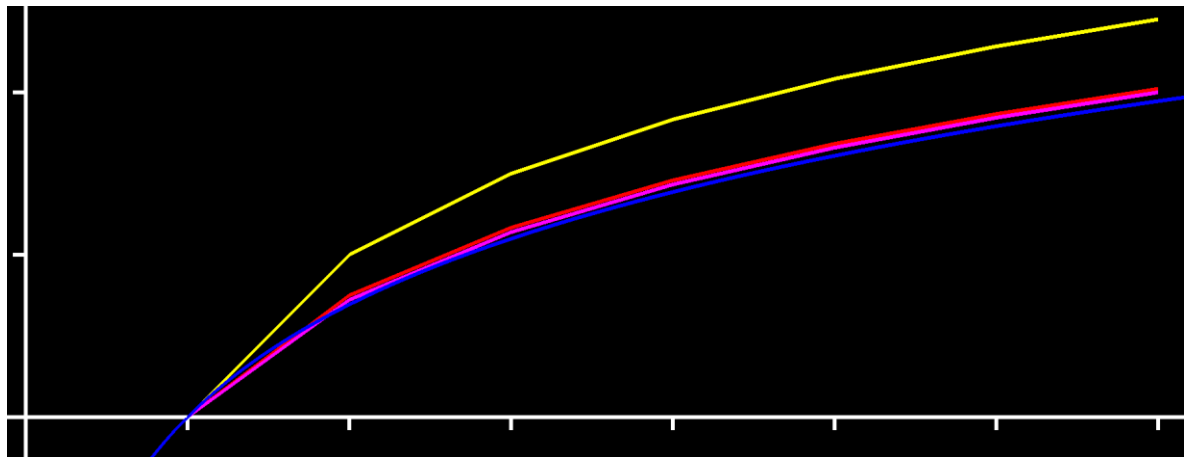
Cargo la aproximación por Runge-Kutta en violeta (Agregar gráfico -> Problema Valor Inicial -> Método de Runge Kutta)

Metodo de Runge Kutta de 4to orden. Solo ingrese h o N, no ambos.	
$f(t, x):$	$1/t$
t_0	1
x_0	0
T:	7
h:	
N:	6
<input type="button" value="Agregar"/>	

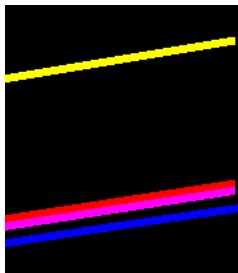
Veo los cuatro elementos que voy a graficar

Elementos a cargar:	
<input type="checkbox"/>	F : $f(t)=\ln(t)$
<input checked="" type="checkbox"/>	E : $f(t, x)=1/t; t_0= 1.0; x_0= 0.0; T=7.0; h=1.0; N=6$
<input checked="" type="checkbox"/>	EM: $f(t, x)=1/t; t_0= 1.0; x_0= 0.0; T=7.0; h=1.0; N=6$
<input checked="" type="checkbox"/>	RK: $f(t, x)=1/t; t_0= 1.0; x_0= 0.0; T=7.0; h=1.0; N=6$

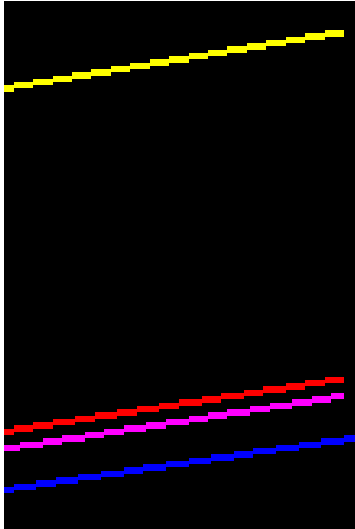
Y le doy a Inicio > Graficar



Cuesta mucho verlo, pero con un buen zoom...



Todavía cuesta verlo, pero si afino el trazo de las curvas (en la pantalla de opciones)



Ahora si se ve mejor.

En azul la función original.

La mejor aproximación es Runge-Kutta en rosa.

Muy cerca le sigue Euler Mejorado en rojo.

Y finalmente queda Euler en amarillo

Ejemplos de Integración Numérica

Supongamos que quiero encontrar el área dentro de la función seno entre 0 y 2π .

Repitamos el proceso y carguemos la función seno y las dos aproximaciones al área que puede realizarle.

Agrego la función de seno en verde (Agregar gráfico > Función simple)

Funcion simple R->R.	
f(t):	sin(t)
(dt/dx):	0
Agregar	

Agrego la aproximación por trapecios en azul (Agregar gráfico > Integración Numérica > Trapecios)

Metodo por Trapezoides. Solo ingrese h o N, no ambos.	
f(t):	sin(t)
a:	0
b:	6.283185307
h:	
N:	10
Agregar	




Aproximo el área entre 0 y 2π con 10 trapecios

Agrego la aproximación por Montecarlo en rojo (Agregar gráfico > Integración Numérica > Montecarlo)

Metodo inimputable.	
f(t):	sin(t)
a:	-6.283185307
b:	0
N:	10
<input type="button" value="Agregar"/> [Redacted]	

Aproximo el área entre -2π y 0 con 10 tiros.

Me quedarán estos elementos:

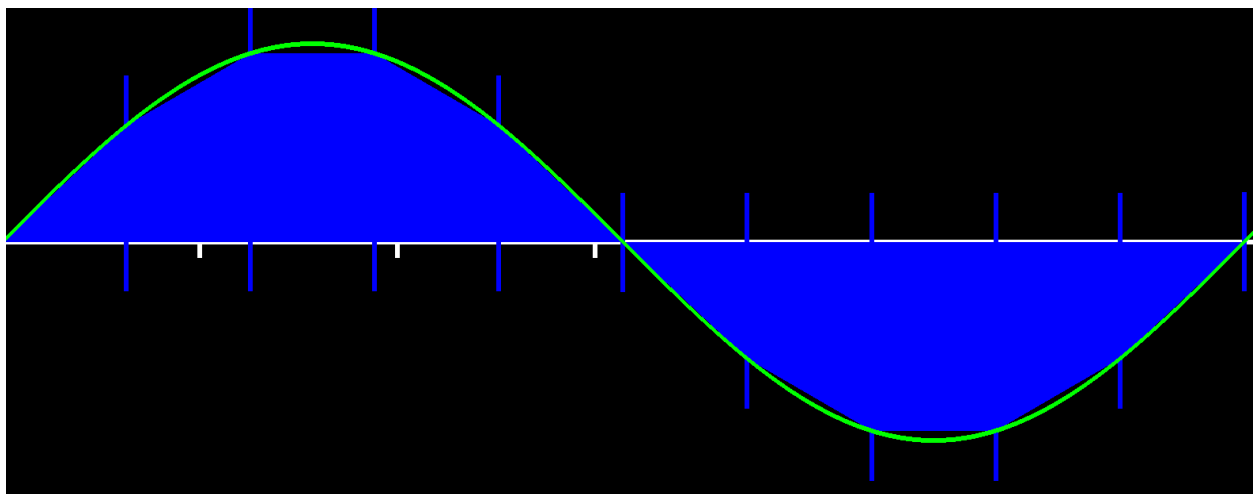
Elementos a cargar:	
	F : f(t)=sin(t)
	T : f(t)=sin(t); a= 0.0; b= 6.2831855; h=0.62831855; N=10
	R : f(t)=sin(t); a= -6.2831855; b= 0.0; h=0.62831855; N=10

Y los paso a graficar (Inicio > Graficar)

Resultado

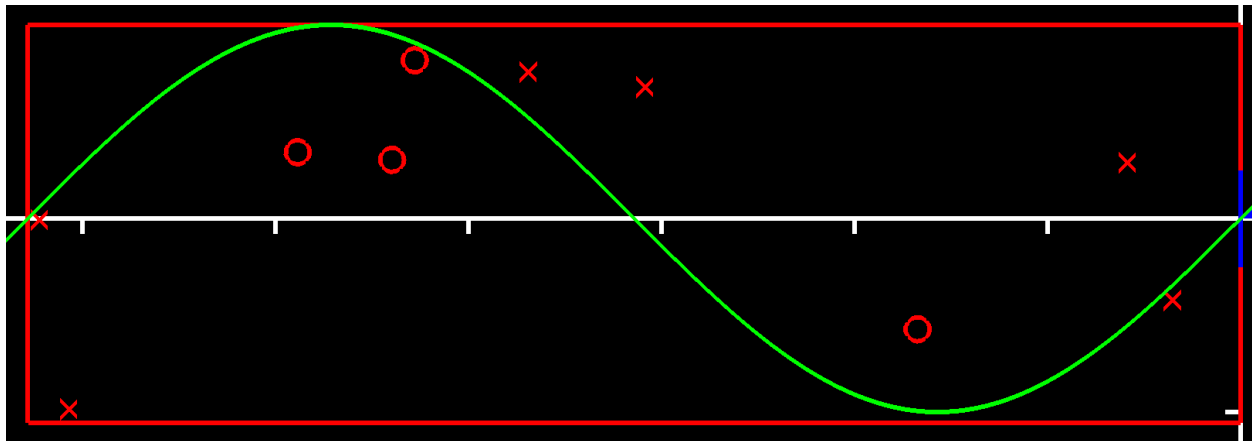
```
[#1] ~> R : f(t)=sin(t); a=-6.2831855; b=0.0; N=10; m=-1.05; M=1.00; area=2.58
[#2] ~> T : f(t)=sin(t); a=0.0; b=6.2831855; N=10; area=-0.00
[#3] ~> F : f(t)=sin(t)
```

Trapezios dio 0 (hay decimales truncados)



Fui muy amable al darle un corte tan cerca de su raíz, por lo que los trapezios son casi idénticos en ambos lados.

Montecarlo dio 2.58. La aleatoriedad no le dio mucha suerte...



3 aciertos positivos, 1 negativo, 10 tiros en total

$$A \sim \frac{(3 - 1)}{10} \cdot (2\pi \cdot 2)$$

Vamos a tratar de aumentar esa suerte dando más disparos.

Metodo inimputable.	
f(t):	sin(t)
a:	-6.283185307
b:	0
N:	100
<input type="button" value="Agregar"/>	

100!

Elementos a cargar:

```

F : f(t)=sin(t)
T : f(t)=sin(t); a= 0.0; b= 6.2831855; h=0.62831855; N=10
R : f(t)=sin(t); a= -6.2831855; b= 0.0; h=0.06283186; N=100
  
```

¡Ahora sí! Montecarlo también alcanzó el 0.00

```
[#1] -> T : f(t)=sin(t); a=0.0; b=6.2831855; N=10; area=-0.00  
[#2] -> R : f(t)=sin(t); a=-6.2831855; b=0.0; N=100; m=-1.05; M=1.00; area=0.00  
[#3] -> F : f(t)=sin(t)
```

