Taller de Desarrollo de Aplicaciones

Práctica No.1

Método predictivo en tiempo

Giovanni Josué Venegas Ramírez
Castro Bouquet Ildefonso

Abstract. - Mediante el método de Euler de inferencia de sistemas en tiempo se realizará la presente practica donde se aplicará el método de predicción de Euler que ayuda a conocer tiempos futuros a partir de un estado actual, así como las fuerzas que ejercen en dicho contexto a analizar de un individuo en un Bunge jump.

I. Introducción. –

Hoy en día, los nuevos métodos de inteligencia artificial nos permiten encontrar relaciones y patrones incomprensibles y nos ayudan a comprender el comportamiento de muchos fenómenos complejos. Una de estas aplicaciones es un sistema que intenta predecir el estado de los fenómenos durante un período de tiempo. A partir de la sugerencia de comportamiento, expresada matemáticamente, el valor inicial se define por el estado inicial y los valores pasados.

Como seres humanos nos podemos preguntar qué es lo podría suceder si se realizan ciertas acciones, y como tal no hay una manera de saber concretamente el efecto a lo realizado, por lo que se hacen propuestas para poder hacer un estimado a lo que se cree que podría ser el resultado, uno de estos métodos nos permite observar el patrón de comportamiento para poder generar un valor de predicción de nuestro interés.

Este tipo de sistemas siguen evolucionando, de diferentes maneras y diferentes usos, todo a partir de un historial para generar posibles soluciones.

En matemática y computación, el método de Euler, llamado así en honor a Leonhard Euler, es un procedimiento de integración numérica para resolver ecuaciones diferenciales ordinarias (EDO) a partir de un valor inicial dado. El método de Euler es el más simple de los métodos numéricos para resolver un problema de valor inicial, y el más simple de los Métodos de Runge-Kutta.

El método de Euler es un método de primer orden, lo que significa que el error local es proporcional al cuadrado del tamaño del paso, y el error global es proporcional al tamaño del paso. También sirve como base para construir métodos más complejos.

La idea es que a pesar de que la curva es desconocida en un principio, su punto de comienzo, al cual denotamos por A0, es conocido. Entonces, de la ecuación diferencial se puede calcular la pendiente de la curva en el punto A0 y por lo tanto la recta tangente a la curva.

Ahora, dando un pequeño paso sobre dicha recta, podemos tomarnos un nuevo punto A1 y suponer que dicho punto pertenece a la curva, entonces seguimos el mismo razonamiento aplicado anteriormente y volvemos a calcular la pendiente de la recta tangente a la curva en el punto A1. Luego de varios pasos tendremos formada una curva poligonal A0A1A2A3... En general esta curva que obtenemos al aplicar el método no diverge lejos de la curva original, además el error entre ambas curvas se puede minimizar si se dan pasos muy pequeños al avanzar sobre la recta tangente a la curva y además el intervalo sobre el que trabajamos es finito.

II. Contexto

Como parte de nuestro programa, debemos obtener el comportamiento de una persona expulsada de un Bungee para definir instantáneamente su trayectoria después del inicio. El método de Euler es aplicado en la práctica para predecir el comportamiento temporal de la ecuación en obtenida en la práctica como modelo matemático a seguir. Por lo tanto, por las condiciones Iniciales, podemos acercarnos a todos los intervalos de tiempo que se ejecutan en cada iteración teniendo en cuenta que la seguridad de la persona se basa en estos cálculos matemáticos.

III. Análisis

Se diseñará un software a través de modularidad y modelo MVC que prediga un movimiento con trayectoria de un bongee, junto a la información generada se hará una gráfica con GNUplot.

Para ello se utilizarían mínimo tres módulos, uno donde se encuentra el software principal (MOTOR), se usará un ".h" para saber qué procesos se harán mediante qué funciones, y por último un ".c" donde se desarrollen las funciones a utilizar.

Primero hay que analizar nuestro problema, necesitamos variables de entrada, en este caso variables para sustituir en la ecuación de Euler:

→ **ΔT,** Es el tiempo discreto
→ **Masa (M),** La masa de una persona en kg
→ **K,** constante del bongee jump
→ **g,** la constante de gravedad: 9.81 m/s^2
→ Condiciones iniciales

También necesitamos saber qué valores de salida daremos:

→ Se estrella o no contra el suelo
→ Valores para gráfica GNUplot
→ Archivo CSV

Ya con entradas y salidas podemos saber el proceso a realizar, en este caso:

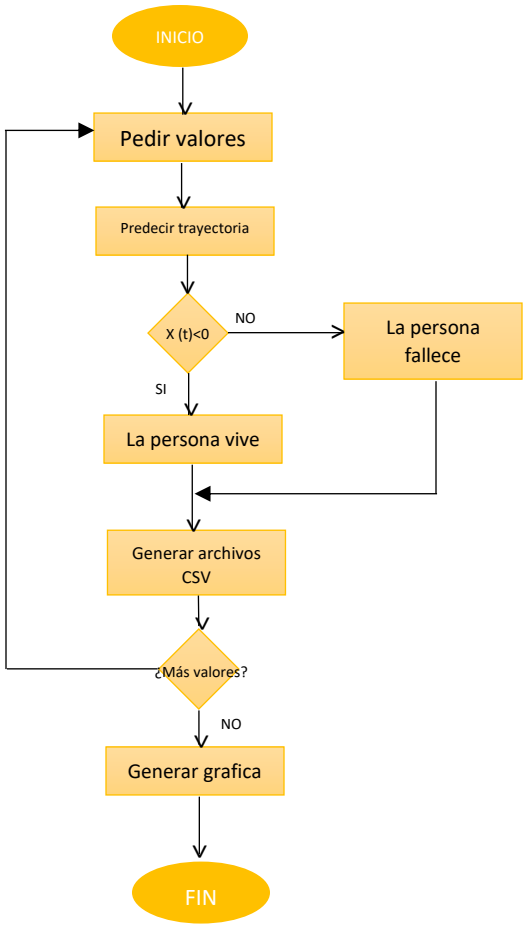
$$x(t+2\Delta t) = -x(t)\left[1+\frac{k\Delta t^2}{m}\right] + 2x(t+\Delta t) + \Delta t^2 g$$

IV. Diseño

El presente desarrollo se basa en el modelo MVC es decir en el modelo-vista-controlador el cual es un estilo de arquitectura de software que separa los datos de aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintitos.

→ **Modelo:** Enfocado a la predicción de la trayectoria.
→ **Vista:** Despliega un menú con las instrucciones del programa para así poder asignar valores
→ **Controlador:** Comparación de posiciones.

Diagrama de flujo de datos (DFD)



V. Pseudocódigo

Se concibe fragmentar el código en módulos que permitan tener una organización optima.

Para este mismo se desarrollarán 3 módulos.

Un maestro, otro para enviar la información a los archivos y un último para el proceso y desarrollo de las ecuaciones.

El módulo principal será la vista, donde colocara un menú, los usuarios podrán insertar el valor(es) de la ecuación. Es decir, el individuo, la constante de elasticidad del Bungee Jump, el intervalo del tiempo (▲t) y la duración total del juego.

funciones.c

```
Inicio
SubProceso MENU(DATOS *DATOS)
Hacer
Escribir
'=================================
================'
Escribir '=                MENU                ='
Escribir '=1 Set values for Dt, Weight, K or cuantity
='
Escribir '=2 Calculate Graphic                     ='
Escribir '=                                        ='
Escribir
'=================================
================'
Escribir 'Input an option: '
Leer x
Si ((x!=1) y (x!=2))
Escribir 'Thats not a valid value.'
FinSi
Si (x==2)
FinSi
Si (x == 1)
Hacer
Escribir 'Insert value for Dt s:'
Leer datos->delta
Si (datos->delta<=0)
Escribir 'Value out of range >0-infinity'
FinSi
Hasta Que (datos->delta<=0)
Hacer
Escribir 'Introduce weight value kg:'
Leer datos->masa
Si (datos->masa<=0)
Escribir 'Value out of range >0-infinity'
FinSi
Hasta Que (datos->masa<=0)
Hacer
Escribir 'Introduce value for k constant of the
bongeeN/m^2:'
Leer datos->k
Si (datos->k<=0)
Escribir 'Value out of range >0-infinity'
FinSi
Hasta Que (datos->k<=0)
Hacer
Escribir 'Introduce how many dots are going to be
calculated:'
Leer datos->cantDelta
Si (datos->cantDelta<=0)
Escribir 'Value out of range >0-infinity'
FinSi
Hasta Que (datos->cantDelta<=0)
//es el valor del for, quiero 15 medidas, osea de x
hasta x+15deltas
FinSi
Hasta Que (x!=2)
FinSubProceso

SubProceso MENU(DATOS *DATOS)
Hacer
Escribir
'=================================
================'
Escribir '=                MENU                ='
Escribir '=1 Set values for Dt, Weight, K or cuantity
='
Escribir '=2 Calculate Graphic                     ='
Escribir '=                                        ='
Escribir
'=================================
================'
Escribir 'Input an option: '
Leer x
Si ((x!=1) y (x!=2))
Escribir 'Thats not a valid value.'
FinSi
```

```
Si (x==2)
FinSi
Si (x == 1)
Hacer
Escribir 'Insert value for Dt s:'
Leer datos->delta
Si (datos->delta<=0)
Escribir 'Value out of range >0-infinity'
FinSi
Hasta Que (datos->delta<=0)
Hacer
Escribir 'Introduce weight value kg:'
Leer datos->masa
Si (datos->masa<=0)
Escribir 'Value out of range >0-infinity'
FinSi
Hasta Que (datos->masa<=0)
Hacer
Escribir 'Introduce value for k constant of the
bongeeN/m^2:'
Leer datos->k
Si (datos->k<=0)
Escribir 'Value out of range >0-infinity'
FinSi
Hasta Que (datos->k<=0)
Hacer
Escribir 'Introduce how many dots are going to be
calculated:'
Leer datos->cantDelta
Si (datos->cantDelta<=0)
Escribir 'Value out of range >0-infinity'
FinSi
Hasta Que (datos->cantDelta<=0)
//es el valor del for, quiero 15 medidas, osea de x
hasta x+15deltas
FinSi
Hasta Que (x!=2)
FinSubProceso

SubProceso CREATEFILE(FILE **FILE)
*file=fopen"graphic.dat","w"
Si (*file==NULL)
SiNo
FinSi
FinSubProceso
```

```
void FillFile(FILE *file,float array[][2],DATOS
*datos)
  int i;
  Para (i=0;i<datos->cantDelta;i++)
    Imprimir (file,"%f, %f\n",array[i][0],array[i][1])
Fin Para

void DesarrolloEcuacion(DATOS *datos,float
array[][2])

  float t,n;
  int i;
  t=0;
  Para (i=0; i<datos->cantDelta; i++)
    if(i==0)
      array[i][1]=10;
      array[i][0]= t; //time value
      array[i+1][1]=10;
    if(i>=1)
      array[i+1][1]=0-(array[i-1][1])*((1)+(((datos-
>k)*(datos->delta)*(datos->delta))/datos-
>masa))+(0.2)*(array[i][1])*(datos->delta)+(datos-
>delta)*(datos->delta)*9.81;/*agregar 0.2*/

    t=t+datos->delta;
    array[i+1][0]= t; //time value
fin para
Fin
```

tipo.h

```
Estructura de  DATOS
  float delta,masa,k
  int cantDelta
Fin de la estructura
```

main.c

```
datos.delta=0.1
datos.masa=60
datos.k=500
datos.cantDelta=100
menu(&datos)
float array[datos.cantDelta][2]
DesarrolloEcuacion(&datos,array)
i=CreateFile(&file)
{
Escribir 'ERROR File couldnt me created'
```

exit1
FillFilefile,array,&datos
Si (i==1)
Cerrar archivo

VI. Código

funciones.c

```c
//
//  funciones.c
//
//
//  Created by Castro Bouquet Ildefonso on 27/09/2020.
//  Created by Venegas Ramirez Giovanni Josue on 27/09/2020.
//

/* Libraries */
#include<stdio.h>
#include<stdlib.h>

#include"tipo.h"

/* funciones.c -- Functions */

/**
*This function generates a menu for the program, asking if user wants to set
values or use the predetermined.
*
*@param
*   datos(DATOS *)
*
*/
void menu(DATOS *datos)
{
  float x;
  do
  {
    printf("================================================\n");
    printf("=                      MENU                    =\n");
    printf("=1) Set values for Dt, Weight, K or cuantity    =\n");
    printf("=2) Calculate Graphic                           =\n");
    printf("=                                               =\n");
    printf("================================================\n");
    printf("Input an option: ");
    scanf("%f",&x);
    if((x!=1)&&(x!=2))
    {
      printf("Thats not a valid value.\n");
    }
    if(x==2)
    {
      return;
    }
    if(x == 1)
    {
      do
      {
        printf("Insert value for Dt (s):");
        scanf("%f",&datos->delta);
        if(datos->delta<=0)
        {
          printf("Value out of range (>0-infinity)\n");
        }
      }while(datos->delta<=0);
      do
      {
        printf("Introduce weight value (kg):");
        scanf("%f",&datos->masa);
        if(datos->masa<=0)
        {
          printf("Value out of range (>0-infinity)\n");
        }
      }while(datos->masa<=0);
      do
      {
        printf("Introduce how many dots are going to be calculated:");
        scanf("%d",&datos->cantDelta);
        if(datos->cantDelta<=0)
        {
          printf("Value out of range (>0-infinity)\n");
        }
      }while(datos->cantDelta<=0);//es el valor del for, quiero 15 medidas,
      //osea de x hasta x+15deltas
    }
  }while(x!=2);
}

/**
*This function creates a file and saves the location on a variable.
*
*@param
*   file(FILE *)
*@returns
*   0 if cant create a file
*   1 if file had been created correctly
*/
int CreateFile(FILE **file)
{
  *file=fopen("graphic.dat","w");
  if(*file==NULL)
  {
    return 0;
  }
  else
  {
    return 1;
  }
}

/**
*This function takes the array and print the information from it to the file.
*
*@param
*   files(FILE *)
*   array[][](float *)
*
*/
void FillFile(FILE *file,float array[][2],DATOS *datos)
{
  int i;
  for(i=0;i<datos->cantDelta;i++)
  {
    fprintf(file,"%f, %f\n",array[i][0],array[i][1]);
  }
}

/**
*This function takes the variables and replace them in the main ecuation, solve
// it, and save the results in the array.
*
*@param
*   datos(DATOS *)
*   array[][](float *)
*
*/
void DesarrolloEcuacion(DATOS *datos,float array[][2])
{
  float t,n;
  int i;
  t=0;
  for (i=0; i<datos->cantDelta; i++)
  {
    if(i==0)
    {
      array[i][1]=10;
      array[i][0]= t; //time value
      array[i+1][1]=10;
    }
    if(i>=1)
    {
      array[i+1][1]=0-(array[i-1][1])*((1)+(((datos->k)*(datos->delta)*(datos->
    }
    t=t+datos->delta;
    array[i+1][0]= t; //time value
  }
}
```

main.c

```c
//
//  main.c
//
//
//  Created by Castro Bouquet Ildefonso on 27/09/2020.
//  Created by Venegas Ramirez Giovanni Josue on 27/09/2020.
//

/* Libraries */
#include<stdio.h>
#include<stdlib.h>

/* Custom Libraries */
#include"tipo.h"
#include"funciones.h"

/* Main */
int main(void)
{
  system("clear");
  int i;
  FILE *file,*gnu_socket;
  DATOS datos;
  /*Predetermined values, set by the problem, can be changed from menu function*/
  datos.delta=0.1;
  datos.masa=60;
  datos.k=500;
  datos.cantDelta=100;
  /**/
  menu(&datos);
  float array[datos.cantDelta][2];
  DesarrolloEcuacion(&datos,array);
  i=CreateFile(&file);
  if(i==0)/*Validate if file cant be created*/
    {
      printf("ERROR File couldnt me created\n");
      exit(1);
    }
  FillFile(file,array,&datos);
  if(i==1)
  {
    fclose(file);
  }
  char *nombre="grafica";
  gnu_socket=popen("gnuplot -persist","w");
  fprintf(gnu_socket,"plot \"%s.dat\" using 1:2 with lines\n",nombre);/*Command
  // for calling gnuplot internally*/
  pclose(gnu_socket);
  /*for(i=0;i<datos.cantDelta;i++)
  {
    printf("%f, %f\n",array[i][0],array[i][1]);
  }*/
}
```

tipo.h

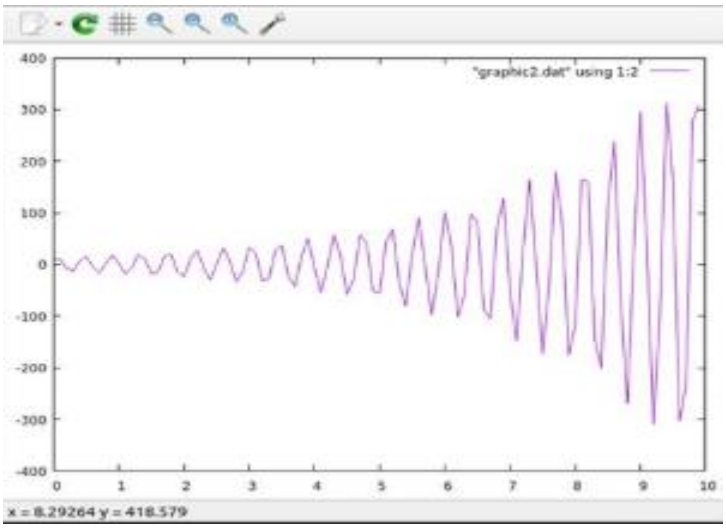
```c
//
//  tipo.h
//
//
//  Created by Castro Bouquet Ildefonso on 27/09/2020.
//  Created by Venegas Ramirez Giovanni Josue on 27/09/2020.
//

/* Libraries */
#include<stdio.h>
#include<stdlib.h>

/* Variables */
typedef struct DATOS
{
  float delta,masa,k;
  int cantDelta;
}DATOS;
```

funciones.h

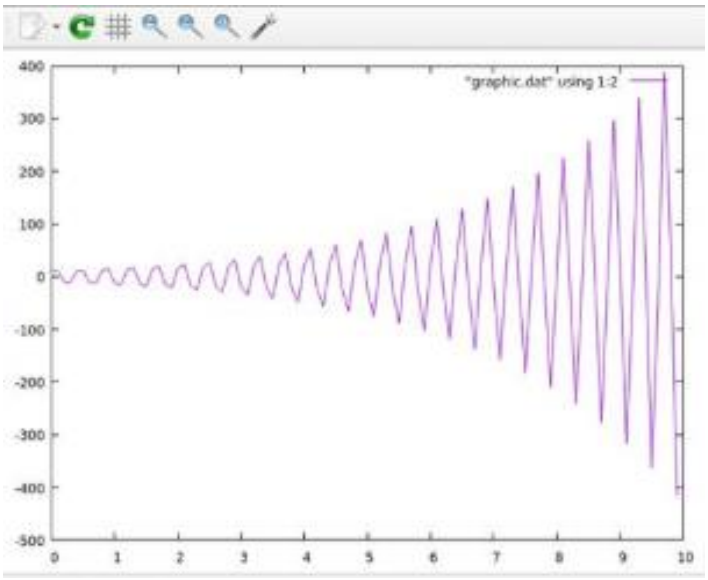
```c
//
//  funciones.h
//
//
//  Created by Castro Bouquet Ildefonso on 27/09/2020.
//  Created by Venegas Ramirez Giovanni Josue on 27/09/2020.
//
#ifndef funciones_h
#define funciones_h

#include <stdio.h>

#ifdef funciones_IMPORT
    #define EXTERN
#else
    #define EXTERN extern
#endif

/* funciones.h -- Function prototypes */

/**
*This function generates a menu for the program, asking if user wants to
// set values or use the predetermined.
*
*@param
*   datos(DATOS *)
*
*/
EXTERN void menu(DATOS *datos);

/**
*This function creates a file and saves the location on a variable.
*
*@param
*   file(FILE *)
*@returns
*   0 if cant create a file
*   1 if file had been created correctly
*/
EXTERN int CreateFile(FILE **file);

/**
*This function takes the array and print the information from it to the file.
*
*@param
*   files(FILE *)
*   array[][](float *)
*
*/
EXTERN void FillFile(FILE *file,float array[][2],DATOS *datos);

/**
*This function takes the variables and replace them in the main ecuation,
// solve it, and save the results in the array.
*
*@param
*   datos(DATOS *)
*   array[][](float *)
*
*/
EXTERN void DesarrolloEcuacion(DATOS *datos,float array[][2]);


#undef funciones_IMPORT
#undef EXTERN

#endif /* funciones_h */
```

VII. Graficas de GNUPLOT

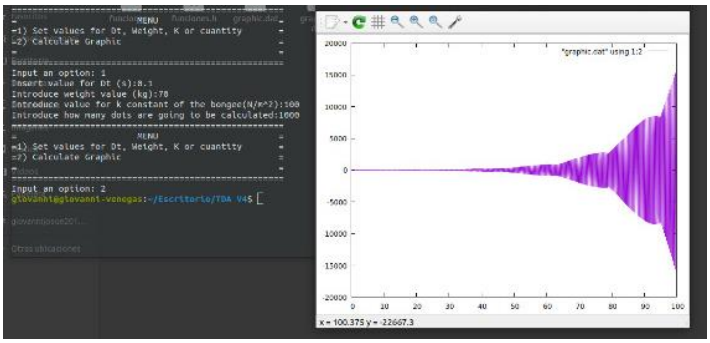
Usando 2x

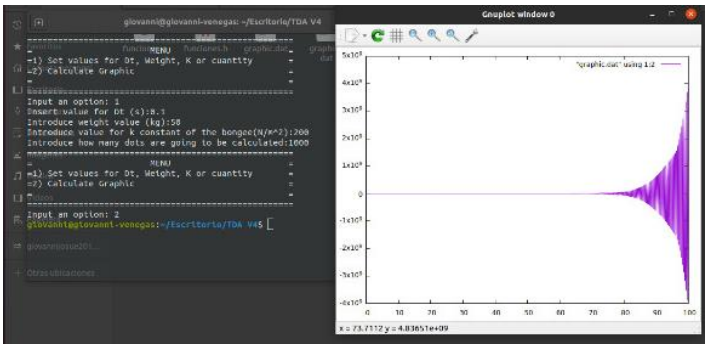


Usando 0.2x



Aumentando el número de puntos N y bajando el valor de K





VIII. PREGUNTAS

¿Como se vería una persona de cierto peso en el bongee si en vez de saltar, lo jalaran desde abajo, tensando la cuerda y dejándola ir?

R= Si se jala desde abajo, sería más corta la trayectoria ya que X0 es menor a como si se jalara desde arriba, ya que está más cerca del suelo.

Imaginemos que esta personita la jalan de los brazos una cierta distancia X0 y luego lo sueltan, ¿Cuál sería su trayectoria?

R= Seria la misma trayectoria ya que es el mismo patrón de subida y bajada.