Examen 1

IBERO
Ciudad de México • Tijuana

Decodificación de señales a partir de series de Fourier

Integrantes del equipo

César Mauricio Arellano Velásquez

Profesor

César Arturo Ángeles Ruiz

Materia

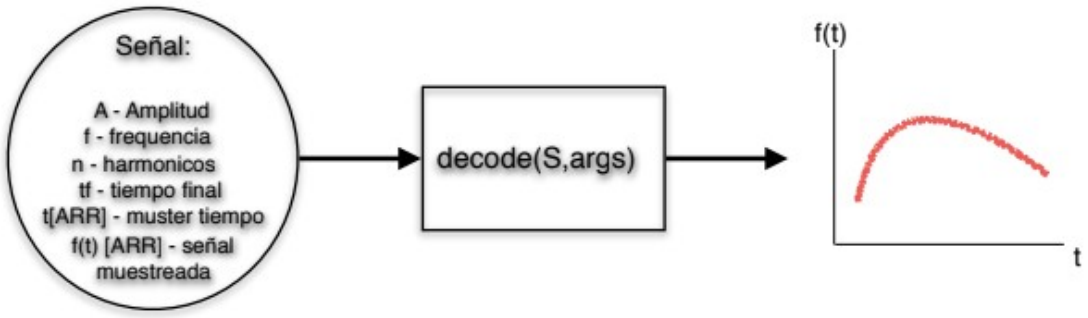
Taller de Desarrollo de Aplicaciones

Introducción:

En la teoría de señales y sistemas, una manera de describir una señal periódica infinita, por medio de una finita, es por medio de su expansión armónica conocida como serie de Fourier.
Ecuación Inicial:

$$f(t) = \frac{A}{2} + \sum_{n=0}^{\infty} -\frac{A}{\pi n}\left[cos(\pi n) - 1\right] sin(2\pi n f t)$$

- Una estructura que contenga los parámetros de la señal:
  - A: amplitud, f: frecuencia, n: número de harmonicos, tf: tiempo final, un arreglo que almacene el muestreo de tiempo, t, y un arreglo que almacene f(t).



Objetivo:

Decodificar una señal a través de valores iniciales dados por el usuario:

- Amplitud
- Frecuencia
- Tiempo Final
- Armónicos
- Tamaño de muestra.

Desarrollo:

1.c

```c
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <math.h>

//Declaración Estructura de Valores de la Señal;
typedef struct def_Senial
{
  double Amplitud, Frecuencia, TFinal,*MusterTiempo,*Evaluacion;
  double TamMuestra,NumHarmonicos;
} TipoSenial;

void PedirDatos(TipoSenial *Valores); // Pedimos datos iniciales de la señal;
void Decode(TipoSenial *Valores); // Decodifica la señal a partir de los valores iniciales de la señal
void ImprimirArch(TipoSenial Valores); // Imprimir el muestreo del tiempo y de todos los valores de la funcion evaluada en determinada t;
void Graficar(); // Grafica en gnuplot el archivo de texto que tiene los valores de la señal decodificada;
void Cargando(char Mensaje[]); // Menú de carga -> estado en el cual se encuentra el programa;

int main (void)
{
  TipoSenial Senal;  // Instanciamos la estructura;
  PedirDatos(&Senal);
  Senal.MusterTiempo = (double *) malloc (sizeof (TipoSenial)*Senal.TamMuestra); // Generamos el arreglo dinámico de muestreo de tiempo de la estructura;
  Senal.Evaluacion = (double *) malloc (sizeof (TipoSenial)*Senal.TamMuestra); // Generamos el arreglo dinámico donde se van a almacenar la funcion evaluada en determinada t de la estru
  Cargando("Decodificando señal\n"); // Estado: Decodificando señal;
  Decode(&Senal);
  Cargando("Imprimiendo datos en archivo"); // Estado: Imprimir datos en archivo;
  ImprimirArch(Senal);
  Cargando("Graficando con GNUPlot"); // Estado: Graficando con GNUPlot;
  Graficar();
  return 0;
}
```

```c
void PedirDatos(TipoSenial *Valores)
{
  printf("Introduzca los siguientes datos:\n\n");
  printf("Amplitud (A): \n");
  scanf(" %lf",&Valores -> Amplitud);
  printf("Frecuencia (f): \n");
  scanf(" %lf",&Valores -> Frecuencia);
  printf("Harmónicos (n): \n");
  scanf(" %lf",&Valores -> NumHarmonicos);
  printf("Tiempo Final (tf): \n");
  scanf(" %lf",&Valores -> TFinal);
  printf("Tamaño de muestra (ARR): \n");
  scanf(" %lf",&Valores -> TamMuestra);
}

void Decode(TipoSenial *Valores)
{
  double Incremento = Valores->TFinal / Valores->TamMuestra;
  double Res, T, PI=3.14159;
  int i=0;
  for(T = 0; T <= Valores->TFinal; T += Incremento)
    {
      Res = 0;
      Valores->MusterTiempo[i]= T;
      for(int j = 1; j<Valores->NumHarmonicos; j++)
      {
         Res += ((Valores->Amplitud / (PI * j)) * (cos (PI * j) - 1) * (sin (2 * PI * j * Valores->Frecuencia * T))); // Evaluacion en la señal;
      }
      Valores->Evaluacion[i] = (Valores->Amplitud / 2) - Res;
      i++;
    }
}
```

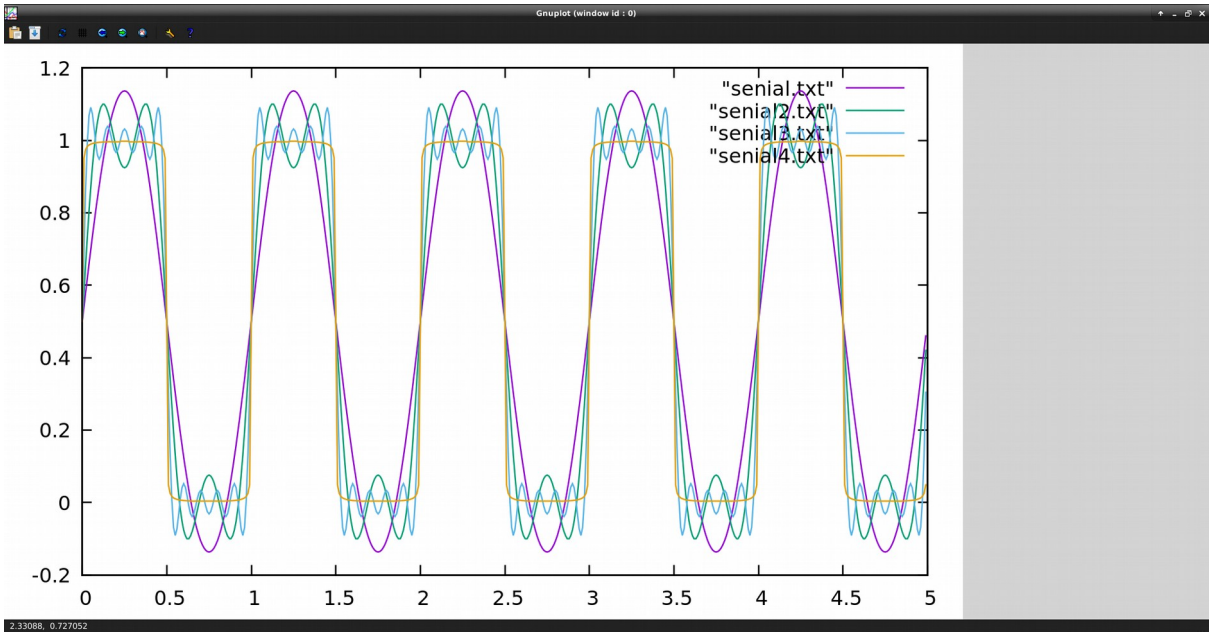
```c
void ImprimirArch(TipoSenial Valores)
{
  int i;
  char NombreArchivo[100];
  FILE *Archivo;
  Archivo = fopen("senial.txt","wt");
  for(i = 0; i<Valores.TamMuestra;i++)
    {
      fprintf(Archivo,"%lf %lf\n",Valores.MusterTiempo[i],Valores.Evaluacion[i]);
    }

  fclose(Archivo);
}

void Graficar()
{
  int i;
  char *AbrirGnuPlot[] = {"set title \"f(t) con n harmonicos\"",
        "set ylabel \"----Señal Muestreada--->\"",
        "set xlabel \"----Muster Tiempo--->\"",
        "plot \"senial.txt\" with lines"
  };
  FILE *VentanaGnuPlot = popen ("gnuplot -persist", "w");
  for (i=0; i<4; i++)
    fprintf(VentanaGnuPlot, "%s \n", AbrirGnuPlot[i]);
}
```

```c
void Cargando (char Mensaje[])
{
  system ("clear");
  puts (Mensaje);
  printf ("\n");
  system ("sleep 0.15");
  system ("clear");
  puts (Mensaje);
  printf (".\n");
  system ("sleep 0.15");
  system ("clear");
  puts (Mensaje);
  printf ("..\n");
  system ("sleep 0.15");
  system ("clear");
  puts (Mensaje);
  printf ("...\n");
  system ("sleep 0.15");
}
```

Gráfica / Evidencia:



Nombre de archivos	Parámetros
senial.txt	Amplitud: 1 Frecuencia: 1 Armónico: 3 Tiempo final: 5 Muster time: 500
senial2.txt	Amplitud: 1 Frecuencia: 1 Armónico: 5 Tiempo final: 5 Muster time: 500
senial3.txt	Amplitud: 1 Frecuencia: 1 Armónico: 10 Tiempo final: 5 Muster time: 500
senial4.txt	Amplitud: 1 Frecuencia: 1 Armónico: 100 Tiempo final: 5 Muster time: 500

Conclusión:

Se puede concluir que una serie infinita de datos puede procesarse y convertirse a una serie finita para lograr un decodificación de una señal específica, haciendo series de Fourier a través de su número armónico, lo que a su vez nos permite interpretar de mejor forma como se comportan ciertas señales bajo ciertos valores iniciales respaldados por un cierto análisis previo.

Punto Extra:

¿Qué sucedió al ir aumentando n?, ¿Cuál es la señal adquirida?
R = Los picos de la señales se van haciéndose más planas de lo normal, haciendo que pase una señal senoidal a una gráfica casi cuadrada.