ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE CIENCIAS

ESCUELA DE FÍSICA Y MATEMÁTICAS

CARRERA DE BIOFÍSICA

"ESTUDIO DE TÉCNICAS DE ANÁLISIS ESPECTRAL Y SU APLICACIÓN EN EL PROCESAMIENTO DIGITAL DE SEÑALES"

TÉSIS DE GRADO

Previa a la obtención del título de:

BIOFÍSICA

RICARDO FABIÁN SIZA GUALPA

RIOBAMBA - ECUADOR

2013

AGRADECIMIENTOS

En primer lugar agradezco a Dios y a mis padres por darme el privilegio de ser su hijo, al igual que a mi familia ya que gracias a su apoyo y paciencia me han guiado hasta la presentación de esta tesis de grado. Soy afortunado de tener mucho que agradecer a las personas que me han ayudado de diferentes modos. A **PhD. Dennis Cazar** y **Mat. Marcelo Cortez,** por toda la ilusión y el esfuerzo que cada día han puesto en este trabajo. Han sido un excelente Director y Asesor de Tesis, pero además han sido excelentes amigos y compañeros.

Y a mis compañeros de aula y amigos, por hacer que me sienta en la ciudad de Riobamba como en mi casa.

DEDICATORIA

Dedico esta tesis a mi querida familia quienes me apoyaron en todas las formas posibles y en especial a mi hermosa madre Eva Mercedes Gualpa Villavicencio quien se desvive por darme apoyo moral, económico y tantas cosas que es imposible enumerarlas aquí para que yo obtenga mi título profesional

También a Don Alfonso por ser un apoyo y pilar fundamental de mí querida madre sabiendo guiarme y aconsejarme en toda mi estancia de estudiante politécnico.

NOMBRE	FIRMA	FECHA
Dr. Silvio Álvarez Luna	_____	_____
Decano de la Facultad de Ciencias		
Dra. Jenny Orbe	_____	_____
Directora Escuela de Física y Matemática		
PhD. Dennis Cazar	_____	_____
Director de Tesis		
Dr. Richard Pachacama	_____	_____
Miembro del Tribunal		
Tlg. Carlos Rodríguez	_____	_____
Director Departamento de Documentación		

NOTA DE TESIS ESCRITA: _____

DERECHOS DE AUTORIA

“Yo Ricardo Fabián Siza Gualpa declaro que soy la autor del presente trabajo de tesis el cual fue elaborado por mi persona bajo la dirección del PhD. Dennis Cazar y colaborador Mat. Marcelo Cortez, haciéndome responsable de las ideas y métodos expuestos en esta Tesis, y el patrimonio intelectual de la Tesis de grado le pertenece a la Escuela Superior Politécnica de Chimborazo”

RICARDO FABIAN SIZA GUALPA
CI.050335608-1

RESUMEN

En el Laboratorio GETNano (Grupo Ecuatoriano para el Estudio Experimental y teórico de Nanosistemas) ubicado en la Escuela de Física y Matemáticas de la Escuela Superior Politécnica de Chimborazo.

Se desarrolló un trabajo titulado “Estudio de Técnicas de Análisis Espectral y su aplicación en el Procesamiento Digital de Señales” con el fin de estudiar la teoría del Análisis de Fourier: su significado, propiedades, y se presenta una aplicación en el campo del diagnóstico médico, se desarrollan también módulos de enseñanza auto consistentes que pueden ser la base para un curso práctico a nivel de pregrado.

Se utilizó los métodos inductivo para la investigación y deductivo para el desarrollando ejemplos y aplicaciones, simplificando su cálculo al utilizar herramientas computacionales como Hardware portátil y un Software Libre SCILAB (Programa de Calculo Numérico) programa muy útil especialmente para el análisis de señales.

Entre los ejemplos desarrollados encontramos desde la generación de señales básicas, convolución de señales, transformada discreta de Fourier, transformada rápida de Fourier, filtros digitales, bancos de filtros, muestreo.

Finalmente se presenta una simulación de ECG (Electrocardiograma Ideal) que permite analizar el origen matemático de las señales eléctricas cardíacas para su aplicación en la identificación y reconocimiento de ciertas patologías. El sistema de simulación emplea un algoritmo de generación de señales biológicas que se

basa principalmente en el desarrollo de datos a través de SCILAB generando señales ECG considerando los distintos intervalos.

Esto tendrá como finalidad la comparación gráfica con otros tipos de electrocardiogramas con el fin de obtener un diagnóstico de patologías del músculo cardíaco mediante la visualización de las variaciones de sus intervalos con respecto al electrocardiograma ideal.

Se recomienda realizar el estudio centrándose en la parte que se utilizara en la parte final del trabajo. Esto se da, debido a que el estudio de Análisis Espectral abarca un sin número de temas que por lo general no es necesario adentrarse.

ABSTRACT

“Study of spectral analysis techniques and its application in the digital signs process”

The development of mathematical algorithms and the study of spectral analysis techniques are very helpful in the generation of biological signs in medical diagnosis.

The Ecuadorian Group for the Nanosystems Experimental and Theoretical Studies (Spanish acronym GETNano), located in the Mathematics and Physics Department of the Escuela Superior Politécnica de Chimborazo, developed works to study the theory of Fourier Analysis. This theory presents applications I the medical diagnosis. The institution also develops self-consistent courses that can be the foundation for a practical course at an undergraduate education level.

The objectives of this research are the following:

- To study the theory of Discrete Fourier transform (DFT).
- To develop an application for DFT using SCILAB.
- To implement a DFT application.

The methodology used in the research was inductive and deductive for the specific applications and examples that simplify calculations. They use computational tools like laptop and free software SCILAB (numerical analysis program) which is very useful in the signs analysis.

As a result, the research generated a biological sign, convolution between signals using the Fourier analysis.

The research found that the spectral analysis and generated a biological sign of cardiac muscle known as Electrocardiography.

INDICE DE ABREVIATURAS

T:	Periodo
DFT:	Transformada Discreta de Fourier
FFT:	Transformada Rápida de Fourier
STFT:	Ventana de la Transformada de Fourier
FT:	Función De Transferencia
EE:	Espacio De Estados
ECG:	Electrocardiograma
FIR:	Respuesta Finita al Impulso
IIR:	Respuesta Infinita al Impulso
SA:	Seno-auricular
AV:	Aurículo-Ventricular

INDICE GENERAL

AGRADECIMIENTOS I
DEDICATORIA II
DERECHOS DE AUTORIA iV
RESUMEN V
ABSTRACT VII
INDICE DE ABREVIATURAS IX
INDICE GENERAL X
INDICE DE FIGURAS XIV
INDICE DE TABLAS XV
INTRODUCCION 6
OBJETIVOS 8
CAPITULO I 9
1 ANÁLISIS ESPECTRAL 9
1.1 FUNDAMENTO TEÓRICO 9
1.2 ANÁLISIS DE FOURIER Y FUNCIONES ORTOGONALES 11
1.3 TRANSFORMADA DISCRETA DE FOURIER 14
1.4 TRANSFORMADA RÁPIDA DE FOURIER 16
1.5 ESPECTRO DE POTENCIA DE UN PÉNDULO IMPULSADO 20
1.6 TRANSFORMADA DE FOURIER EN DIMENSIONES SUPERIORES 21
1.7 ANÁLISIS WAVELET 22
1.8 VENTANA DE LA TRANSFORMADA DE FOURIER 23
1.9 TRANSFORMADA WAVELET CONTINUA 26
1.10 TRANSFORMADA WAVELET DISCRETA 28
1.11 ANALISIS DE MULTIRESOLUCIÓN 30
1.12 ESCALA DE LA FUNCIÓN 32
1.13 DISEÑO DE FILTROS 37
1.14 DISEÑO DE FILTROS DE RESPUESTA IMPULSIONAL FINITA 38
1.14.1 Técnicas de Ventana (Windowing) 38

1.15 DISEÑO DE FILTROS DE RESPUESTA IMPULSIONAL INFINITA 39
1.15.1 Filtros Analógicos 39
1.15.2 Filtros Butterworth 40
1.16 DISEÑO DE FILTROS IIR A PARTIR DE FILTROS ANALÓGICOS 40
1.17 DISEÑO DE FILTROS PASA BAJO 41
1.18 BANCO DE FILTROS Y EL ALGORITMO DE LA PIRAMIDE 42

CAPITULO II 45
2 DESARROLLO DE EJEMPLOS APLICANDO LA TRANSFORMADA DE FOURIER UTILIZANDO EL SOFTWARE SCILAB 45
2.1 DESCRIPCIÓN GENERAL 45
2.2.1 INTRODUCCIÓN 45
2.2 SCILAB Y PROCESAMIENTO DE SEÑALES 46
2.3 TRATAMIENTO DE SEÑALES 47
2.3.1 Frecuencia de la señal 47
2.3.2 Amplitud de la señal y componente continua 49
2.3.3 Modificación de la fase 49
2.3.4 Suma de ondas senoidales 50
2.3.5 Series de Fourier 51
2.4 REPRESENTACIÓN DE DATOS 52
2.4.1 Sistema lineal por su función de transferencia (FT) 52
2.4.2 Espacio de estados (EE) de un sistema lineal 52
2.5 CAMBIO DE REPRESENTACIÓN 53
2.5.1 Paso de FT al EE y viceversa 53
2.5.2 Discretización de sistemas continuos 54
2.5.3 Filtrado y su representación gráfica 55
2.6 LA FFT Y LA DFT 57
2.7 CONVOLUCIÓN 57
2.8 DISEÑO DE FILTROS CON SCILAB 58
2.8.1 Diseño de filtros FIR 58
2.9 MUESTREO 60
2.10 TÉCNICAS DE MUESTREO DE FRECUENCIA 61
2.11 EJEMPLOS APLICANDO LA TEORIA 64

2.11.1 Generación de una secuencia exponencial compleja 64
2.11.2 Generación de una secuencia exponencial real 65
2.11.3 Generación de señales y ruido aleatorio 66
2.11.4 Suavizamiento de una señal por un filtro de promedio movil 67
2.11.5 Autocorrelación de una secuencia senosoidal comprimida por ruido68
2.11.6 Transformada de fourier de tiempo discreto 69
2.11.7 Transformada de fourier 70
2.11.8 Calculo de la DFT inversa 71
2.11.9 Calculo de la DFT para 512 datos 72
2.11.10 Muestreo y conversion de datos 72

CAPITULO III 76
3 ANÁLISIS DE SEÑALES CARDIOVASCULARES UTILIZANDO TÉCNICAS DE PROCESAMIENTO DIGITAL DE SEÑALES 76
3.1 INTRODUCCIÓN 76
3.2 SISTEMA DE CONDUCCIÓN CARDÍACO 78
3.2.1 Nodo Sinusal 78
3.2.2 Nodo Aurículo-Ventricular 79
3.2.3 Conducción a Nivel Auricular 79
3.2.4 Has de His 79
3.2.5 Ramas y Fibras de Purkinje 80
3.3 EL ELECTROCARDIOGRAMA ECG 81
3.4 EL CORAZÓN CON RELACIÓN AL ECG 82
3.5 PARTES DE UN ECG 83
3.5.1 Vía eléctrica normal 83
3.6 ALGORITMO DETECTOR DE ONDAS 85
3.6.1 Frecuencia Cardíaca 85
3.7 SIMULACIÓN DE LA SEÑAL ECG 86
3.5.1 Cálculos 86
3.7.1 Generación del complejo QRS 88
3.7.2 Generación de ondas sinusoidales P,T y U 91
3.8 IMPLEMENTACIÓN EN SCILAB 94
3.8.1 Código de datos 95

3.8.2 Onda Sinusoidal P 95
3.8.3 Onda Sinusoidal Q 96
3.8.4 Complejo QRS 97
3.8.5 Onda Sinusoidal S 98
3.8.6 Onda Sinusoidal T 99
3.8.7 Onda Sinusoidal U 100
3.8.8 Implementación en SCILAB 102
3.9 ECG NORMAL Y ANATOMÍA CARDÍACA 102
3.9.1 ECG normal 103
3.9.2 Ruta de estimulos en los ritmos sinusales 103
3.9.3 Variación del complejo QRS 104

CAPITULO IV 107
4 CONCLUSIONES Y RECOMENDACIONES SOBRE EL PROCESO 107
4.1 CONCLUSIONES 107
4.2 RECOMENDACIONES 108
BIBLIOGRAFIA 110
ANEXOS 116
DESARROLLO DE PROGRAMAS EN EL SOFTWARE SCILAB 116

INDICE DE FIGURAS

Figura 2.1. Consola de Software SCILAB……..…….……46

Figura 2.2. Representación de la frecuencia de una señal…….……..48

Figura 2.3. Modificación de la frecuencia de una señal.…….……48

Figura 2.4. Representación de la variación de la Amplitud……..…..….....49

Figura 2.5. Representación de la variación del tiempo…..……..……...50

Figura 2.6. Verificación de una función periódica..…..……..……..…….....*51*

Figura 2.7. Señal de salida del filtro…….……..…...56

Figura 2.8. Señal de entrada X del filtro…….…….…....56

Figura 2.9. Función coseno sin FFT…….…….……...……...57

Figura 2.10. Función con filtro pasa banda……..62

Figura 2.11. Señal con filtro pasa bajo tipo 1 y 2…….…….……..….63

Figura 2.12. Generación secuencia exponencial compleja……..…..……...64

Figura 2.13. Generación secuencia exponencial real……65

Figura 2.14. Generación de señales y ruido aleatorio……..……..66

Figura 2.15. Suavizamiento de una señal.……...67

Figura 2.16. Secuencia sinusoidal corrompida por ruido…….....…….….68

Figura 2.17. Transformada de Fourier de tiempo discreto……..…..69

Figura 2.18. Transformada Discreta de Fourier……..……70

Figura 2.19. Calculo de la DFT Inversa …..……71

Figura 2.20. DFT en una función generada con 512 puntos……....72

Figura 2.21. Muestreo y conversión de datos(a)……73

Figura 2.22. Muestreo y conversión de datos (b)……..……74

Figura 2.23. Muestreo y conversión de datos (c)……..……..….75

Figura 3.1. Sistema de conducción cardiaco……..……...……80

Figura 3.2. Formas de onda de ECG……...……..81

Figura 3.3. Trazado de una señal de ECG…….….82

Figura 3.4. Representación onda P con SCILAB……..……...……96

Figura 3.5. Representación onda Q con SCILAB...……...….97

Figura 3.6. Representación del complejo QRS con SCILAB……...…98

Figura 3.7. Representación onda S con SCILAB...……...…..99

Figura 3.8. Representación onda T con SCILAB…….100

Figura 3.9. Representación onda U con SCILAB...……..101

Figura 3.10. Implementación de ECG con SCILAB...……..102

Figura 3.11. Variación de la amplitud del complejo QRS en SCILAB……...105

Figura 3.12. ECG con la variación del complejo QRS...……...105

Figura 3.13. Comparación ECG basal y ECG patológico……106

INDICE DE TABLAS

Tabla 3.1. Muestra de datos de ondas para ECG……..……..……..95

INTRODUCCION

En la actualidad el análisis de señales se ha extendido y especializado en colaboración con diversas áreas, una de ellas es el sector salud.

En 1807, Fourier, establece en los trabajos presentados en el instituto de Francia que: cualquier señal periódica puede ser representada por una serie de sumas trigonométricas en senos y cosenos relacionadas armónicamente.

Los argumentos establecidos por Fourier eran imprecisos y en 1829 Dirichlet proporcionó las condiciones precisas para que una señal periódica pueda ser representada por una serie de Fourier.

Fourier obtuvo además, una representación para señales no periódicas, no como suma de ondas sinusoidales relacionadas armónicamente, sino como integrales de ondas sinusoidales las cuales no todas están relacionadas armónicamente. Al igual que las series de Fourier, la integral de Fourier, llamada Transformada de Fourier, es una de las herramientas más poderosas para el análisis de sistemas LTI (Sistema Lineal Invariante en el Tiempo).

El presente trabajo estudia los métodos de análisis de Fourier, y su aplicación en el procesamiento digital de señales, realizando un estudio teórico y de ejercicios con la ayuda del Software SCILAB.

En la práctica siempre se trabaja con una cantidad finita de señales, por esto hay pérdida de información al realizar la DFT. En cualquier forma, generalmente se toman medidas para que estas pérdidas de señales estén acotadas y que estén fuera de rangos críticos en las aplicaciones. Gracias a esto es que podemos hablar por celular y, si todo funciona bien, entender nítidamente las conversaciones.

El programa que ayudara al desarrollo de la infinidad de cálculos se denomina SCILAB fue creado en 1990 por investigadores del INRIA y la EcoleNationale des Ponts et Chaussees (ENPC). El Consorcio SCILAB se formó en mayo de 2003 para ampliar las contribuciones y promover a SCILAB como software de referencia en todo el mundo académico y de la industria.

En junio de 2010, el Consorcio anunció la creación de SCILAB ENTERPRISES. SCILAB ENTERPRISES desarrolla y comercializa, directamente o a través de una red internacional de proveedores de servicios aliados, un conjunto integral de servicios para los usuarios de SCILAB. En colaboración con el Consorcio Scilab, SCILAB ENTEPRISES ofrece su experiencia para empresas y académicos. El objetivo final de SCILAB ENTERPRISES es contribuir a mejorar el uso de SCILAB y hacerlo aún más eficaz y más fácil. En septiembre de 2010, SCILAB ENTERPRISES anunció una alianza mundial con EQUALIS para proporcionar servicios de asistencia en línea.

OBJETIVOS

Objetivo General

- Estudio de las técnicas de análisis espectral y su aplicación en el procesamiento digital de señales.

Objetivos Específicos

- Estudio de la teoría de la Transformada de Fourier.

- Desarrollo de ejemplos de aplicación de la Transformada de Fourier utilizando el software libre SCILAB.

- Estudio e implementación de una aplicación de la Transformada de Fourier al procesamiento digital de señales.

CAPITULO I

1 ANÁLISIS ESPECTRAL

1.1 *FUNDAMENTO TEORICO*

La naturaleza a menudo se comporta de manera diferente a como nuestra intuición lo predice. Por ejemplo, cuando observamos un movimiento armónico, inmediatamente podemos calcular el período, pero no la estructura detallada de los datos dentro de cada período. Las series de Fourier proporcionan una herramienta para analizar la respuesta de estado estacionario de sistemas para una señal periódica de entrada. En este capítulo se extenderá la idea del análisis de Fourier para tratar con funciones periódicas o no. Se hace esto introduciendo la Transformada de Fourier.

Conforme se desarrolle la teoría se verá como la forma exponencial compleja de la representación de la serie de Fourier de una función periódica surge como un caso especial de la Transformada de Fourier. Mientras las transformadas de Fourier encontraron en un principio la mayoría de sus aplicaciones en la solución de ecuaciones diferenciales parciales, es válido decir que hoy en día los métodos de la transformadas de Fourier se usan extensivamente en el análisis de señales y sistemas [1]-[10].

Era Fourier quien primero señaló que una función periódica arbitraria *f (t),* con un período *T*, se puede descomponer en una suma de términos armónicos simples, que son funciones periódicas que son múltiplos de la frecuencia fundamental *1 / T* de la función *f (t)*. Cada coeficiente de la suma está dado por un integrante del producto de la función y el complejo conjugado de ese término armónico.

Suponiendo que tenemos una función dependiente del tiempo *f (t)* con un período *T*, es decir, *f (t + T) = f (t)*, el teorema de Fourier se puede escribir como una sumatoria que se conoce comúnmente como la serie de Fourier.

$$f(t) = \sum_{j=-\infty}^{\infty} g_j e^{-ij\omega t}, \qquad \textbf{(1.1)}$$

Aquí $\omega = 2\pi / T$ es la frecuencia angular fundamental y g_j son los coeficientes de Fourier, que vienen dados por:

$$g_j = \frac{1}{T}\int_0^T f(t) e^{ij\omega t} dt. \qquad \textbf{(1.2)}$$

El teorema de Fourier se puede derivar de las propiedades de las funciones exponenciales en

$$\phi_j(t) = \frac{1}{\sqrt{T}} e^{-ij\omega t}, \qquad \textbf{(1.3)}$$

Esta forma una base ortonormal establecida en la región de un período de *f (t)*: es decir,

$$\int_{t_0}^{t_0+T} \phi_j^*(t)\phi_k(t)dt = \langle j|k\rangle = \delta_{jk}, \qquad \textbf{(1.4)}$$

Donde t_0 es un punto de partida arbitrario, $\emptyset_j^*(t)$ es el complejo conjugado de $\emptyset_k(t)$ y δ_{jk} es la función δ de Kronecker, es decir vale 1 si son iguales, y 0 si son diferentes [1].

El coeficiente de Fourier g_j de la ecuación **(1.2)** se obtiene entonces multiplicando la ecuación **(1.1)** por $e^{-ik\omega t}$ y luego la integración de ambos lados de la ecuación de más de un período de *f (t)*. Por conveniencia se puede siempre definir con facilidad de cálculo.

1.2 *ANÁLISIS DE FOURIER Y FUNCIONES ORTOGONALES*

Podemos generalizar el teorema de Fourier a una función no periódica definida en una región *x ∈ [a, b]*, si tenemos una base completa de un conjunto de funciones ortonormales $\phi_k(x)$ con

$$\int_a^b \phi_j^*(x)\phi_k(x)dx = \langle j|k\rangle = \delta_{jk}, \qquad \textbf{(1.5)}$$

Para cualquier función arbitraria *f(x)* definida en la región $x \in [a, b]$, siempre se puede escribir

$$f(x) = \sum_j g_j \phi_j(x) \qquad \textbf{(1.6)}$$

Si la función es cuadrado integrable, es decir

$$\int_a^b |f(x)|^2 dx < \infty. \qquad \textbf{(1.7)}$$

La sumatoria de la series es sobre todo los estados posibles en el juego completo, y el coeficiente g_j viene dado por

$$g_j = \int_a^b \phi_j^*(x) f(x) dx = \langle j | f \rangle. \qquad \textbf{(1.8)}$$

La transformada de Fourier continua se obtiene si se restringe la serie a la región de $t \in [-T/2, T/2]$ y luego se extiende el período T hasta el infinito.

Necesitamos redefinir $j\ \omega$ como ω y $\sum_j$ como $\left(1/\sqrt{2\pi}\right)\int d\omega$. Entonces la suma se convierte en un integrante que comúnmente se conoce como la función de la integral de Fourier.

$$f(t) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} g(\omega) e^{-i\omega t} d\omega, \qquad \textbf{(1.9)}$$

El coeficiente de Fourier de la función viene dada por

$$g(\omega) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(t) e^{i\omega t} dt. \qquad \textbf{(1.10)}$$

Las ecuaciones **(1.9)** y **(1.10)** definen una parte integral de transformación y su inversa, que comúnmente se conoce como la transformada de Fourier y la

transformada inversa de Fourier. Se puede demostrar que las ecuaciones **(1.9)** y **(1.10)** son coherentes, es decir, se puede obtener la ecuación **(1.10)** multiplicando la ecuación **(1.9)** por $e^{i\omega t}$ y luego su integración. Se debe utilizar la función δ de Dirac durante este proceso.

La función δ de Dirac está definida como

$$\delta(x-x') = \begin{cases} \infty & si\ x = x' \\ 0 & en.otra.parte \end{cases} \qquad \textbf{(1.11)}$$

y

$$\int_{-\infty}^{\infty} \delta(x-x')dx = \lim_{\in \to +0} \int_{x'-\in}^{x'+\in} \delta(x-x')dx = 1. \qquad \textbf{(1.12)}$$

Así la función *δ* Dirac *δ(ω)* también se puede interpretar como la transformada de Fourier de una función constante $f(t) = 1/\sqrt{2\pi}$.

La transformada de Fourier también se puede aplicar a otros tipos de variables o de dimensiones superiores. Por ejemplo, la transformada de Fourier de una función *f(r)* en tres dimensiones está dada por:

$$f(r) = \frac{1}{(2\pi)^{3/2}} \int g(q) e^{iq.r} dq, \qquad \textbf{(1.13)}$$

Donde el coeficiente de Fourier

$$g(q) = \frac{1}{(2\pi)^{3/2}} \int f(q) e^{-iq.r} dr. \qquad \textbf{(1.14)}$$

Nótese que **(1.13)** y **(1.14)** son integrales tridimensionales. El espacio definido por *q* se suele llamar el espacio de momentos.

1.3 *TRANSFORMADA DISCRETA DE FOURIER (DFT)*

La transformada unidimensional de Fourier se define por las ecuaciones **(1.9)** y **(1.10)**; como siempre, tenemos que convertir las variables continuas en variables discretas antes de desarrollar un procedimiento numérico. Consideremos *f(x)* como una cantidad física espacio-dependiente partir de las mediciones experimentales. Si las mediciones se llevan a cabo entre $x = 0$ y $x = L$, *f(x)* es distinto de cero sólo para $x \in [0, L]$.

Para simplificar nuestro problema, podemos suponer que los datos han sido tomados en puntos espaciados uniformemente a cada intervalo de $h = L / (N - 1)$, donde *N* es el número total de puntos de datos. Se supone que los datos periódicamente repetidos fuera de la región de $x \in [0, L]$, lo que equivale a la imposición de la condición de contorno periódicas en el sistema finito. El número de onda correspondiente en el espacio de momentos es entonces muy discreta, con un intervalo de $\kappa = 2\pi / L$.

La DFT de tal conjunto de datos puede entonces ser expresada en términos de una sumatoria.

$$f_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} g_j e^{i 2\pi jk/N}, \qquad \textbf{(1.15)}$$

Con los coeficientes de Fourier dados por:

$$g_j = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} f_k e^{-i2\pi jk/N}, \qquad \textbf{(1.16)}$$

Donde se ha utilizado la convención que $f_k = f(t = k\tau)$ y $g_j = g(\omega = j\nu)$. Podemos demostrar que estas dos sumas son consistentes, lo que significa que la transformada inversa de los coeficientes de Fourier dará los valores exactos de *f(t)* en *t=0, τ,..., (N - 1) τ*. Sin embargo, esta transformada inversa de Fourier no garantiza suavidad en la función de los datos recuperados. Nótese que las funciones exponenciales en la serie forman una base discreta de un conjunto de funciones ortogonales. Es decir:

$$\left\langle \phi_j \middle| \phi_m \right\rangle = \sum_{k=0}^{N-1} \frac{1}{\sqrt{N}} e^{-i2\pi k j/N} \frac{1}{\sqrt{N}} e^{i2\pi km/N}$$

$$\left\langle \phi_j \middle| \phi_m \right\rangle = \frac{1}{N} \sum_{k=0}^{N-1} e^{i2\pi k(m-j)/N} = \delta_{jm}. \qquad \textbf{(1.17)}$$

Antes de introducir la Transformada Rápida de Fourier, vamos a examinar cómo podemos aplicar la DFT de la ecuación **(1.16)** de una manera sencilla.

Podemos separar la parte real *(Re)* y la parte imaginaria *(Im)* con grupos de los coeficientes, por simplicidad.

$$\mathrm{Re} \quad g_j = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \left(\cos \frac{2\pi jk}{N} \mathrm{Re} \quad f_k + sen \frac{2\pi jk}{N} \mathrm{Im} \quad f_k \right), \qquad \textbf{(1.18)}$$

$$\mathrm{Im} \quad g_j = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \left(\cos \frac{2\pi jk}{N} \mathrm{Im} \quad f_k - sen \frac{2\pi jk}{N} \mathrm{Re} \quad f_k \right), \qquad \textbf{(1.19)}$$

Nótese que no es necesario separar en la práctica, aunque la mayoría de la gente suele hacerlo. La separación de las partes real e imaginaria es conveniente, porque entonces sólo tenemos que tratar con números reales.

Un tema importante aquí es la precisión de la función de los datos recuperados en lugares distintos de los puntos. Cuando los datos de la transformada de Fourier o su inversa se realizan, sólo puede tener un conjunto discreto de puntos de datos. Así que una interpolación es inevitable si queremos saber cualquier valor que no está en los puntos de datos.

Se puede aumentar el número de puntos para reducir los errores en la interpolación de los datos, pero también hay que tomar en cuenta el crecimiento de los errores de redondeo con el número de puntos utilizados.

1.4 *TRANSFORMADA RAPIDA DE FOURIER (FFT)*

Como podemos ver claramente, el sencillo algoritmo de la transformada discreta de Fourier presentada en la sección **(1.3)** no es muy eficiente, ya que el tiempo de cálculo necesario es proporcional a N^2. Con el fin de resolver este problema, se ha desarrollado FFT. El elemento clave de la FFT es la de reorganizar los términos de la serie y tener la suma realizada de una manera jerárquica. Por ejemplo, podemos realizar una serie de adiciones pairwise para llevar a cabo la suma si el número de puntos de datos es de la potencia de 2: es decir, $N = 2^M$ donde M es

un número entero. La idea detrás de la FFT había sido estudiada desde hace mucho tiempo, incluso antes que la primera computadora fuera construida. Gauss desarrolló una versión de la transformada rápida de Fourier y publicó su trabajo en América neoclásico [1]. Sin embargo, nadie se dio cuenta de la idea de Gauss, conectado a la computación moderna. El algoritmo de la transformada rápida de Fourier fue descubierto formalmente y puesto en práctica por Cooley y Tukey (1965) [1]. Aquí daremos una breve descripción de su idea.

El más simple algoritmo de la FFT se logra con la observación de que podemos separar los términos pares e impares en la transformación discreta de Fourier como:

$$g_j = \sum_{k=0}^{N/2-1} f_{2k} e^{-i2\pi j(2k)/N} + \sum_{k=0}^{N/2-1} f_{2k+1} e^{-i2\pi j(2k+1)/N},$$
$$g_j = x_j + y_j e^{-i2\pi j/N}, \qquad \textbf{(1.20)}$$

Donde;

$$x_j = \sum_{k=0}^{N/2-1} f_{2k} e^{-i2\pi jk/(N/2)} \qquad \textbf{(1.21)}$$

y

$$y_j = \sum_{k=0}^{N/2-1} f_{2k+1} e^{-i2\pi jk/(N/2)}. \qquad \textbf{(1.22)}$$

Lo que hemos hecho es volver a escribir la transformada de Fourier de una suma de n términos de dos sumas, cada una de N / 2 términos. Este proceso se puede repetir continuamente hasta obtener dos términos en cada suma si N =2^M, donde

M es un número entero. Hay una simetría más entre g_k para $k < N/2$ y g_k para $k \geq N/2$: es decir,

$$g_j = x_j + w^j y_j, \quad \textbf{(1.23)}$$

$$g_j + N/2 = x_j - w^j y_j, \quad \textbf{(1.24)}$$

Donde $w = e^{-i2\pi/N}$ y $j = 0, 1, ..., N/2 - 1$. Esto es muy importante en la práctica, porque ahora hay que realizar la transformación sólo hasta $j = N/2 - 1$, y los coeficientes de Fourier con mayor ***j*** se obtienen con la ecuación anterior, al mismo tiempo. Hay dos ingredientes importantes en el rápido algoritmo de transformada de Fourier. Después de la suma se descompone M veces, tenemos que añadir puntos de datos individuales, en parejas. Sin embargo, debido a la clasificación de los términos pares e impares en cada nivel de descomposición, los puntos en cada par en el primer nivel de adiciones pueden ser muy separados en la cadena de datos original. Sin embargo, Cooley y Tukey (1965) [1] encontraron que si se registra el índice de la cadena de datos con un número binario, un bit de orden inverso, se puso cada par de puntos de datos junto a la otra de las sumas en el primer nivel. Tomemos un conjunto de 16 puntos de datos $f_0, f_1, \ldots, f_{15}$ como un ejemplo. Si los grabamos con un índice binario, tenemos 0000, 0001, 0010, 0011,..., 1111, para todos los puntos de datos.

El orden se logra si se invierte el orden de cada cadena binaria. Por ejemplo, el orden inverso de bits de 1000 es 0001. Así que el orden de los datos después de

la reversión de bits es $f_0, f_8, f_4, f_{12}, \ldots, f_3, f_{11}, f_7, f_{15}$. Entonces el primer nivel de las adiciones se realiza entre $f_0\ y\ f_8, f_4\ y\ f_{12}, \ldots, f_3\ y\ f_{11}, f_7\ y\ f_{15}$.

Las ecuaciones **(1.23)** y **(1.24)** se pueden aplicar repetidamente para sumar los 2^{l-1} espacios de separación, en la corriente de bits de datos invertida. Aquí ***l*** indica el nivel de las adiciones, por ejemplo, el primer conjunto de adiciones corresponde a *l = 1.*

En cada nivel de adiciones, dos puntos de datos se crean a partir de cada par. Nótese que w^jestá asociado con el segundo término de **(1.23).** Con el rápido algoritmo de la transformada de Fourier, el tiempo de cálculo necesario para un gran conjunto de puntos de datos se reduce enormemente. Esto lo podemos ver mediante el examen de los pasos de cálculo necesarios en la transformación. Supongamos que $N = 2^M$, de modo que después de la inversión, es necesario llevar a cabo M niveles de adiciones y $N/2^l$ adiciones al *l-esimo* nivel.

Un análisis cuidadoso muestra que el tiempo de cálculo total en el algoritmo de la FFT es proporcional a N $\log_2$ N el lugar de N^2, como es el caso de la recta de la transformada discreta de Fourier. Algoritmos similares se pueden idear para N de la potencia de 4, 8, y así sucesivamente.

Muchas versiones y variaciones de los programas de la FFT están disponibles en Fortran (Burrus y Parques, 1985) y en otros lenguajes de programación. Uno de los primeros programas de Fortran de FFT fue escrito por Cooley, Lewis y Welch

(1969) [1]. Ahora muchas computadoras vienen con una biblioteca de FFT, que se suele escribir en lenguaje de máquina y diseñado específicamente para la arquitectura del sistema.

La mayoría de las rutinas de la FFT se escriben con variables complejas. Sin embargo, es más fácil tratar con sólo las variables reales. Siempre se pueden separar las partes reales e imaginarias en las ecuaciones **(1.20)-(1.24)**, como se discutió anteriormente. En problemas reales, α puede ser un poco superior a dos para una máquina escalar. Sin embargo, la ventaja de la vectorización en FFT no es tan significativa como en la recta hacia delante de la DFT. Así que, en general, es posible que necesitemos para examinar el problema de estudio, y la FFT sin duda una herramienta importante, ya que los recursos informáticos disponibles son siempre limitados.

1.5 *ESPECTRO DE POTENCIA DE UN PÉNDULO IMPULSADO*

Como se sabe un péndulo impulsado con amortiguación pueden presentar ya sea un comportamiento periódico o caótico. Una forma de analizar la dinámica de un sistema no lineal es el estudio de su espectro de potencia. El espectro de potencia de una variable dinámica se define como el cuadrado del módulo de la función del coeficiente de Fourier,

$$S(\omega)=|g(\omega)|^2, \qquad \textbf{(1.25)}$$

Donde *g(ω)* viene dada por la ecuación **(1.10)** Con la disponibilidad de este esquema, la evaluación del espectro de potencia de una variable dependiente el tiempo se vuelve sencillo. El péndulo impulsado con amortiguamiento es descrito por;

$$\frac{dy_1}{dt} = y_2, \tag{1.26}$$

$$\frac{dy_1}{dt} = -qy_2 - seny_1 + b\cos\omega_0 t, \tag{1.27}$$

Donde $y_1(t) = \theta(t)$ es el ángulo entre el péndulo y la vertical y_2(t)=dθ(t)/dt es su velocidad angular correspondiente, *q* es el coeficiente de amortiguación, y *b* y ω_0 son la amplitud y la frecuencia angular de la fuerza. El espectro de potencia del ángulo dependiente del tiempo y la velocidad angular en función del tiempo se puede obtener fácilmente mediante la realización de una DFT.

1.6 *TRANSFORMADA DE FOURIER EN DIMENSIONES SUPERIORES*

La transformada de Fourier se puede obtener de una forma muy directa en dimensiones más altas, si nos damos cuenta de que podemos transformar cada coordenada, como si se tratara de un problema unidimensional con todos los índices de coordenadas, las demás se mantiene constante. Tomemos el caso de dos dimensiones como un ejemplo.

Supóngase que los datos son de un dominio rectangular con puntos de malla N_1en una dirección y N_2 en la otra. Así que el número total de puntos es N = $N_1 N_2$. La DFT es:

$$g_{jk} = \frac{1}{\sqrt{N}} \sum_{l=0}^{N_1-1} \sum_{m=0}^{N_2-1} f_{lm} e^{-i2\pi(jl/N_1 + km/N_2)}$$
$$g_{jk} = \frac{1}{\sqrt{N_1}} \sum_{l=0}^{N_1-1} e^{-i2\pi jl/N_1} \frac{1}{\sqrt{N_2}} \sum_{m=0}^{N_2-1} f_{lm} e^{-i2\pi km/N_2}. \quad \textbf{(1.28)}$$

De este modo, podemos obtener la primera transformación de todos los términos ***m*** bajo un índice ***l-esimo*** fijo y luego por todos los términos ***l*** con un índice de ***k*** fijo.

1.7 ANALISIS WAVELET

El análisis Wavelet se introdujo por primera vez por Haar (1910) [1], pero no se reconoce como una poderosa herramienta matemática hasta la década de 1980. Fue Morlet [1] quien utilizó por primera vez el enfoque en el análisis de wavelet de datos sísmicos. La transformada wavelet contiene información espectral a diferentes escalas y ubicaciones de la secuencia de datos, por ejemplo, la intensidad de una señal en torno a una frecuencia específica y un tiempo específico.

Esto está en contraste con el análisis de Fourier, en el que un determinado coeficiente de transformada contiene información acerca de una escala específica

o la frecuencia del espacio de datos completos sin referirse a su importancia para la ubicación en el flujo de datos originales. El método wavelet es extremadamente útil en el análisis de señales de tiempo corto, datos transitorios, o la complejidad del diseño.

El desarrollo y las aplicaciones de análisis wavelet desde 1980 han demostrado que muchas más aplicaciones surgirán en el futuro. Aquí damos una breve introducción al tema y señalamos sus posibles aplicaciones.

1.8 VENTANA DE LA TRANSFORMADA DE FOURIER

Es un esfuerzo por corregir la diferencia presentada en **(1.7)**. Denis Gabor adapto la Transformada de Fourier para poder analizar una pequeña sección de la señal en un determinado tiempo (mediante una especie de ventana). Esta adaptación es la que conoce como Ventana de la Transformada de Fourier (STFT), cual lleva una señal del plano del tiempo al plano bidimensional de tiempo y frecuencia.

Es importante mencionar que la STFT representa una especie de compromiso entre el dominio del tiempo y el de la frecuencia de una señal, ya que provee algo de información acerca de cuándo y a que frecuencia de una señal ocurre un determinado evento. Sin embargo, solamente se puede obtener dicha información con una precisión limitada, la cual está acotada por el tamaño de la ventana. Es

deseable en muchas aplicaciones que la estructura local en un conjunto de datos pueda ser amplificada y analizada.

Esto es debido a que no puede ser capaz de obtener los datos a lo largo de todo el espacio sino solo en una escala específica de interés particular. A veces también queremos filtrar el ruido en torno a los límites de los datos. Una ventana de la transformada de Fourier se puede utilizar para seleccionar la información de los datos en un lugar específico. Se puede definir como la ventana de la transformada de Fourier de la función *f(*t) en la ventana de la función *w (t - τ*).

$$g(\omega,\tau)=\frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty} f(t)w(t-\tau)e^{i\omega t}dt \qquad \textbf{(1.29)}$$

La función ventana *w (t - τ*) se utiliza aquí para extraer información acerca de *f(t)* en las cercanías *t = τ*. La función de ventana *w(t)* es comúnmente elegido para ser un real, incluso la función, y satisface la propiedad:

$$\int_{-\infty}^{\infty} w^2(t)dt=1. \qquad \textbf{(1.30)}$$

Las funciones típicas de ventana incluyen la función de ventana triangular

$$w(t)=\begin{cases}\dfrac{1}{\sqrt{N}}\left(1-\dfrac{|t|}{\sigma}\right) & si\ |t|<\sigma \\ 0 & si\ |t|\geq\sigma\end{cases} \qquad \textbf{(1.31)}$$

La función de ventana gaussiana es:

$$w(t) = \frac{1}{\sqrt{N}} e^{-t^2/2\sigma^2}, \tag{1.32}$$

Donde *N* es la constante de normalización y σ es una medida de la anchura de la ventana. Tan pronto como σ se selecciona, *N* puede ser evaluada con la ecuación **(1.31)**. De la definición, también podemos recuperar los datos de sus coeficientes de Fourier a través de la transformada inversa como en la transformada de Fourier convencional.

$$f(t) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} g(\omega, \tau) w(\tau - t) e^{-i\omega t} d\omega d\tau \tag{1.33}$$

El producto interno, es decir, la integral sobre el cuadrado de la amplitud de datos, es igual a la integral sobre el cuadrado de su amplitud del coeficiente de la transformada, es decir,

$$\int_{-\infty}^{\infty} |f(t)|^2 dt = \int_{-\infty}^{\infty} |g(\omega, \tau)|^2 d\omega d\tau. \tag{1.34}$$

La ventaja de la transformada de Fourier afina los datos de modo que podemos obtener la estructura local de los datos o suprimir los efectos no deseados en la cadena de datos. Sin embargo, la ventana de la transformada de Fourier trata con un conjunto uniforme datos y no sería capaz de distinguir las estructuras detalladas de estos datos a diferentes escalas.

1.9 TRANSFORMADA WAVELET CONTINUA

La transformada de Fourier con ventana puede proporcionar información en un lugar determinado en el tiempo, pero no proporciona los datos con una escala específica en la ubicación seleccionada. Se puede obtener información acerca de un conjunto de datos a nivel local y también a diferentes escalas a través del análisis wavelet. La transformada wavelet continua de una función *f (t)* se define mediante la integral,

$$g(\lambda,\tau)=\int_{-\infty}^{\infty} f(t)u_{\lambda\tau}^{*}(t)dt, \qquad \textbf{(1.35)}$$

Donde $u_{\lambda\tau}^{*}(t)$ es el complejo conjugado de la wavelet

$$u_{\lambda\tau}(t)=\frac{1}{\sqrt{|\lambda|}}u\left(\frac{t-\tau}{\lambda}\right), \qquad \textbf{(1.36)}$$

Los parámetros λ y τ suelen ser elegidos para ser reales, y seleccionar, respectivamente, la escala y la ubicación de la secuencia de datos durante la transformación. La función *u(t)* es el generador de todas las ondas $u_{\lambda\tau}(t)$ y se llama wavelet madre o simplemente wavelet.

Hay algunas restricciones en la selección de un sentido wavelet *u(t).* Por ejemplo, con el fin de tener la transformada inversa definida, que debe tener

$$Z=\int_{-\infty}^{\infty}\frac{1}{|\omega|}\left|z(\omega)\right|^{2}d\omega<\infty, \qquad \textbf{(1.37)}$$

Donde *z(ω)* es la transformada de Fourier de *u(t).* La restricción antes mencionada se llama la condición de admisibilidad de la wavelet [1], equivale;

$$z(0)=\frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty}u(t)dt=0, \qquad \textbf{(1.38)}$$

Si *u(t)* es de cuadrado integrable y decae cuando t → ± ∞, la razón *u(t)* se denomina wavelet, es decir, una onda pequeña, en comparación con una onda plana típica, que satisface la condición anterior, pero no es cuadrado integrable. La wavelet *u(t)* cumple la condición de normalización

$$\langle u|u\rangle=\int_{-\infty}^{\infty}|u(t)|^2\,dt=1 \qquad \textbf{(1.39)}$$

Por conveniencia veamos un sencillo wavelet

$$u(t)=\Theta(t)-2\Theta\left(t-\frac{1}{2}\right)+\Theta(t-1), \qquad \textbf{(1.40)}$$

Se llama la wavelet Haar, con la función de paso

$$\Theta(t)=\begin{cases}1 & si\ t>0\\ 0 & si\ t\le 0\end{cases} \qquad \textbf{(1.41)}$$

Los coeficientes de la transformada wavelet g (λ, τ) se obtienen a través de la ecuación. **(1.35)** con la onda dada por

$$u_{\lambda\tau}(t)=\frac{1}{\sqrt{|\lambda|}}\left[\Theta(t-\tau)-2\Theta\left(t-\tau-\frac{\lambda}{2}\right)+\Theta(t-\tau-\lambda)\right]. \qquad \textbf{(1.42)}$$

Lo dificultad aquí es tener una forma analítica de g (λ, τ), incluso para una simple forma de *f(t).* En general, la integración debe llevarse a cabo numéricamente.

De la definición de la transformada wavelet continua y las limitaciones en la selección de la wavelet *u(t)*, se puede demostrar que la función de los datos se pueden recuperar de la transformada wavelet inversa;

$$f(t) = \frac{1}{Z}\int_{-\infty}^{\infty} \frac{1}{\lambda^2} g(\lambda,\tau) u_{\lambda\tau}(t) d\lambda d\tau. \qquad \textbf{(1.43)}$$

También puede demostrarse que la transformada wavelet satisface una identidad similar a la de la transformada de Fourier o la ventana transformada de Fourier con

$$\int_{-\infty}^{\infty} |f(t)|^2 dt = \frac{1}{Z}\int_{-\infty}^{\infty} \frac{1}{\lambda^2} |g(\lambda,\tau)|^2 d\lambda d\tau. \qquad \textbf{(1.44)}$$

Analíticamente, la transformada wavelet continua es más fácil de tratar que la transformada wavelet discreta. Sin embargo, la mayoría de los datos obtenidos son discretos en la naturaleza. Más importante aún, es mucho más fácil aplicar el análisis de los datos numéricamente si la transformación se define con variables discretas.

1.10 TRANSFORMADA WAVELET DISCRETA

Desde la transformada wavelet continua, podemos obtener información detallada sobre los datos en diferentes ubicaciones y escalas. El inconveniente es que la

información recibida es redundante porque efectivamente se descompone una secuencia unidimensional de datos en un flujo de dos dimensiones de datos.

Para eliminar esta redundancia, podemos tomar la onda en ciertas escalas seleccionadas (escalas diádicas en λj = 2j para j entero) y los lugares determinados (k puntos de diferencia en la escala de λj). La transformación se puede lograr nivel por nivel con el análisis de multiresolución de Mallat [1]-[9] bajo un esquema de pirámide. En la finalidad de llevar a cabo el análisis de Fourier en términos de la serie de Fourier, se expande la secuencia de tiempo como:

$$f(t) = \sum_{j=-\infty}^{\infty} \sum_{k=-\infty}^{\infty} c_{jk} u_{jk}(t), \qquad \textbf{(1.45)}$$

Donde el conjunto base,

$$u_{jk}(t) = 2^{j/2} u\left(2^{j} t - k\right) \qquad \textbf{(1.46)}$$

La transformada se obtiene a partir del argumento general de $u_{jk}(t)$ que forma un juego completo, de base ortonormal.

$$c_{jk} = \left\langle u_{jk} \middle| f \right\rangle = \int_{-\infty}^{\infty} f(t) u_{jk}(t) dt, \qquad \textbf{(1.47)}$$

La expansión de la ecuación **(1.45)** es también conocida porque la síntesis puede reconstruir la secuencia de tiempo *f(t)* si todos los coeficientes c_{jk} son conocidos.

En consecuencia, la transformación de la ecuación. **(1.47)** se denomina de análisis. Para simplificar nuestro problema, hemos tomado la base fija para ser ortonormal a pesar de que la integridad es la única condición necesaria. También se ha supuesto que el conjunto de base es real y ortonormal por simplicidad, aunque la integridad es la única condición necesaria requerida. Podemos seguir el enfoque que hemos utilizado en la sección anterior para la transformada wavelet continua para obtener todas las integrales en los coeficientes de transformación. Sin embargo, la estructura jerárquica de la Transformada Discreta Wavelet nos permite utilizar un método mucho más eficiente para obtener la transformada sin la preocupación de todas las integrales que intervienen.

1.11 ANÁLISIS DE MULTI RESOLUCIÓN

En primer lugar se puede definir un conjunto de espacios lineares de vectores W_j, con cada ajuste en el espacio cubierto por la función u_{jk} para - ∞ <k <∞, lo que nos permite descomponer *f(t)* en los componentes que residen con W_j.

$$f(t) = \sum_{j=-\infty}^{\infty} d_j(t), \qquad \textbf{(1.48)}$$

Donde $d_j(t)$ se llama el detalle de *f (t)* en W_j y viene dada por:

$$d_j(t) = \sum_{k=-\infty}^{\infty} c_{jk} u_{jk}(t). \qquad \textbf{(1.49)}$$

A continuación, podemos introducir otra serie de espacios lineales de vectores V_j, con cada adición de su sub espacio $V_{j-1} \subset V_j$ y W_{j-1}. Podemos visualizar esto examinando en el espacio tridimensional de Euclides.

Entonces la proyección de *f(t)* en el espacio V_j puede ser escrito como:

$$a_j(t) = \sum_{k=-\infty}^{j-1} \sum_{l=-\infty}^{\infty} c_{kl} u_{kl}(t), \tag{1.50}$$

Los espacios aproximados W_j, V_j, y $a_j(t)$ y los detalles de $d_j(t)$ simplemente están relacionados por:

$$V_{j+1} = V_j \oplus W_j, \tag{1.51}$$

$$a_{j+1}(t) = a_j(t) + d_j(t). \tag{1.52}$$

Veamos un ejemplo sencillo de la descomposición de una función *f(t)* dada en un determinado espacio de V_j.

La jerarquía de espacio se da como

$$\begin{aligned} V_j &= V_{j-1} \oplus W_{j-1} \\ V_j &= V_{j-2} \oplus W_{j-2} \oplus W_{j-1} \\ V_j &= V_{j-3} \oplus W_{j-3} \oplus W_{j-2} \oplus W_{j-1}, \end{aligned} \tag{1.53}$$

que nos permite ampliar la función como

$$\begin{aligned} f(t) &= A_1(t) + D_1(t) \\ f(t) &= A_2(t) + D_2(t) + D_1(t) \\ f(t) &= A_3(t) + D_3(t) + D_2(t) + D_1(t), \end{aligned} \tag{1.54}$$

Donde $A_1(t)$ es la aproximación del primer nivel o la proyección de *f(t)* en V_{j-1}, $A_2(t)$ es la aproximación de segundo nivel o la proyección en V_{j-2}, $D_1(t)$ es el detalle en primer nivel W_{j-1}, $D_2(t)$ es el detalle del segundo nivel en W_{j-2}, y así sucesivamente.

La esencia del análisis multiresolución es por lo tanto, para descomponer una secuencia de datos en los niveles de diferencia de aproximaciones y detalles.

1.12 ESCALA DE LA FUNCIÓN

El análisis multiresolución de *f(t)* se puede realizar de una forma concisa, si existe un conjunto de funciones $\upsilon_{jk}(t)$ que forman una base completa existente en el espacio ortogonal de V_j con

$$\left\langle \upsilon_{jk}(t) \middle| \upsilon_{jl}(t) \right\rangle = \int_{-\infty}^{\infty} \upsilon_{jk}(t)\upsilon_{jl}(t)dt = \delta_{kl}. \qquad \textbf{(1.55)}$$

Aquí $\upsilon_{jk}(t)$ se denominan funciones de escala las cuales satisfacen la relación

$$\upsilon_{jk}(t) = 2^{j/2}\upsilon\left(2^j t - k\right), \qquad \textbf{(1.56)}$$

Donde la función de escalado *v(t)* a veces también se conoce como la función de escalado padre o wavelet padre. Nótese que $\upsilon_{jk}(t)$ con *j* diferente no se puede hacer ortogonales entre sí debido a que los espacios V_j están anidados en lugar

de ser ortogonales, a diferencia de la W_j espacios. Más importante aún, V_0 y W_0 son tanto los subespacios de V_1 porque $V_0 \oplus W_0 = V_1$.

Por lo tanto, se puede expandir una función de base en V_0, *v (t)* = V_{00} *(t),* y una función de base de W_0, *u (t)* = U_{00}*(t),* en términos de la base que figuran en V_1 como:

$$\upsilon(t) = \sum_{k=-\infty}^{\infty} h(k)\upsilon_{lk}(t) = \sum_{k=-\infty}^{\infty} h(k)\sqrt{2\upsilon}(2t-k), \qquad \textbf{(1.57)}$$

$$u(t) = \sum_{k=-\infty}^{\infty} g(k)\upsilon_{lk}(t) = \sum_{k=-\infty}^{\infty} g(k)\sqrt{2\upsilon}(2t-k), \qquad \textbf{(1.58)}$$

Donde *h(k)* y *g(k)* se conocen como filtros. Las relaciones de dos escalas que figuran en las ecuaciones **(1.58)** y **(1.59)** son fundamentales para el desarrollo de un algoritmo eficiente para el cálculo de la transformada wavelet discreta. Hay ciertas propiedades que se pueden derivar de las condiciones ortogonales de las funciones de escala y wavelets. Por ejemplo, la integración en el tiempo de la relación de escala, para la función de escalado se obtiene

$$\sum_{k=-\infty}^{\infty} h(k) = \sqrt{2}. \qquad \textbf{(1.59)}$$

Además, dado que v (t) | v (t - l) = δ_{0l}, también tenemos

$$\sum_{k=-\infty}^{\infty} h(k)h(k+2l) = \delta_{0l}, \qquad \textbf{(1.60)}$$

Después de aplicar la ecuación de escala para la función de escalado se puede demostrar, a partir de la transformada de Fourier que

$$\int_{-\infty}^{\infty} \upsilon(t)dt = 1. \qquad \textbf{(1.61)}$$

Utilizando la condición de admisibilidad

$$\int_{-\infty}^{\infty} u(t)dt = 0 \qquad \textbf{(1.62)}$$

como consecuencia que W_0 es ortogonal a V_0, también podemos obtener

$$g(k) = (-1)^k h(r-k), \qquad \textbf{(1.63)}$$

Donde k es un número entero arbitrario, impar. Sin embargo, si el número cero de *h(k)* es finito e igual a n, se debe tener *r = n - 1*.

A fin de tener todos los *h(k)* para *k = 0, 1, ..., n – 1* determinados, necesitamos un total de *n* ecuaciones independientes. Las ecuaciones **(1.59)** y **(1.60)** proporcionan un total de *n / 2 + 1* ecuaciones.

Por lo tanto, son libres de imponer más condiciones sobre *h(k).* Si queremos recuperar los datos de polinomios hasta el *(n/2 -1)* orden, los momentos en los filtros deben cumplir.

$$\sum_{k=0}^{n-1} (-1)^k k^l h(k) = 0, \qquad \textbf{(1.64)}$$

Para *l* = 0, 1,..., n /2 - 1, lo que proporciona otras *n/2* ecuaciones. Nótese que el caso de *l = 0*, ya sea en la ecuación. **(1.60)** o la ecuación. **(1.64)** se puede derivar de la ecuación **(1.59)** y otras relaciones dadas.

Por lo tanto las ecuaciones **(1.59)**, **(1.60)**, y **(1.64)**, juntas proporcionan *n* ecuaciones independientes para *n* coeficientes *h(k),* y por lo tanto puede ser determinadas de manera única para un *n* dado. Ahora vamos a utilizar un ejemplo sencillo para ilustrar lo señalado anteriormente. Consideremos el caso de función wavelet Haar de escalado para *n = 2*.

La función Haar de escala es un cuadro de entre 0 y 1, a saber, *v (t)* = *Θ(t)* - *Θ (t - 1),* donde *Θ (t)* es la función escalón. Entonces tenemos:

$$\upsilon(t) = \upsilon(2t) + \upsilon(2t-1), \qquad \textbf{(1.65)}$$

Que da h(0)= h(1)= $1/\sqrt{2}$ y h(k)= 0 para k≠ 0, 1.

También tenemos g(0) = $1/\sqrt{2}$, g(1) = - $1/\sqrt{2}$, y g (k) = 0 si k ≠ 0, 1, que viene de $g(k) = (-1)^k h(n-1-k)$ = $(-1)^k h(1-k)$, con n = 2. Considerando ahora n = 4, tenemos:

$$\begin{aligned} h(0) &= \frac{1+\sqrt{3}}{4\sqrt{2}}; h(1) = \frac{3+\sqrt{3}}{4\sqrt{2}}; \\ h(2) &= \frac{3-\sqrt{3}}{4\sqrt{2}}; h(3) = \frac{1-\sqrt{3}}{4\sqrt{2}}, \end{aligned} \qquad \textbf{(1.66)}$$

Lo que se conoce comúnmente como la wavelet D4 [1]. Para n más grande, sin embargo, puede que tengamos que resolver las ecuaciones acopladas n de *h(k)* en forma numérica. Un mejor esquema se puede diseñar a través de z_k de un polinomio dentro del círculo unitario.

$$p(z)=\sum_{k=0}^{n/2-1}\frac{(n/2-1+k)!}{k!(n/2-1)!}\frac{(z-1)^{2k}}{(-z)^{k}} \qquad \textbf{(1.67)}$$

Los coeficientes *h(k)* están dados por *h(k)* = $b_k/\sqrt{N}$, donde b_k son los coeficientes de la expansión:

$$(z+1)^{n/2}\prod_{k=1}^{n/2-1}(z-z_k)=\sum_{k=0}^{n-1}b_k z^{n-k-1} \qquad \textbf{(1.68)}$$

Y la constante de normalización

$$N=\sum_{k=0}^{n-1}b_k^2 \qquad \textbf{(1.69)}$$

En general, podemos expresar las funciones de base para los espacios de V_{j-1} y W_{j-1} en términos de los correspondientes al espacio como V_j.

$$\upsilon_{j-1k}(t)=\sum_{l=-\infty}^{\infty}h(l)\upsilon_{j2k+1}(t), \qquad \textbf{(1.70)}$$

$$u_{j-1k}(t)=\sum_{l=-\infty}^{\infty}g(l)\upsilon_{j2k+1}(t), \qquad \textbf{(1.71)}$$

Después con las ecuaciones **(1.46)**, **(1.56)**, **(1.57)**, y **(1.58).** Estas proporcionan un medio para descomponer un conjunto de datos de nivel por nivel, a partir de cualquier resolución elegida.

1.13 DISEÑO DE FILTROS

La teoría general del diseño de Filtros Digitales nos lleva al agrupamiento de las funciones para el diseño en cuatro métodos [2]-[3]:

- Diseño de filtros IIR usando prototipos analógicos.
- Diseño de filtros IIR directo.
- Diseño de filtros FIR directo.
- Diseño de filtro inverso.

Los tres primeros producen filtros selectivos de frecuencias que son diseñados a partir de las especificaciones de funcionamiento de la respuesta en magnitud. Las técnicas de la última categoría encuentran en los coeficientes del filtro responsables de los datos de la respuesta temporal o de la respuesta en frecuencia dados.

Hay varias formas de especificar el filtro que se necesita. Una especificación amplia puede ser “para un sistema con una velocidad de muestreo de 500 Hz, se necesita un filtro que elimine el ruido por encima de 30 Hz “. Una especificación más rigurosa es dar especificaciones de rizado en banda pasante, atenuación en la banda de rechazo o anchura en la transición. Una especificación muy precisa pediría conseguir los objetivos de funcionamiento con el filtro de orden mínimo o una forma de magnitud determinada o requerir un filtro FIR.

1.14 DISEÑO DE FILTROS DE RESPUESTA IMPULSIONAL FINITA (FIR)

1.14.1 Técnicas de Ventana (Windowing) [2]-[8].

En teoría el diseño de filtros FIR es sencillo. Se toma la transformada inversa de Fourier de la respuesta en frecuencia deseada y se obtiene la respuesta impulsional discreta en el dominio temporal, de acuerdo con:

$$h(n) = \frac{1}{2\pi}\int_{-\pi}^{\pi} H(\omega)e^{j\omega n}d\omega \qquad -\infty < n < \infty \tag{1.72}$$

En la práctica el problema es que muchos de los filtros de interés tienen una respuesta impulsional infinita y una relación causa-efecto no casual. Un ejemplo claro de esto, viene dado por este Filtro Pasa Bajo con una frecuencia de corte ω_c:

$$H(\omega|\omega_c) = \begin{cases} 1 & |\omega| \le \omega_c \\ 0 & |\omega| > \omega_c \end{cases} \tag{1.73}$$

Su respuesta impulsional se obtiene introduciendo $H(\omega|\omega_c)$ en la fórmula **(1.71)** se obtiene:

$$h(n|\omega_c) = \frac{1}{\pi n}\sin|\omega_c n| \qquad -\infty < n < \infty \tag{1.74}$$

Para obtener una longitud finita es necesario implementar una técnica mediante la cual se pueda tomar N elementos de **(1.74)** y que centrado en *n=0*, elimina los elementos sobrantes. Esta operación puede ser representada mediante la

multiplicación de la secuencia de elementos por una Ventana Rectangular de la siguiente forma:

$$R_N(n) = \begin{cases} 1 & 0 \le n \le N-1 \\ 0 & N-1 \le n \le 1 \end{cases} \quad \textbf{(1.75)}$$

1.15 DISEÑO DE FILTROS DE RESPUESTA IMPULSIONAL INFINITA (IIR)

1.15.1 Filtros Analógicos

En esta sección recordaremos algunos de los filtros analógicos (tiempo continuo) clásicos. Estos son definidos en el dominio frecuencial mediante su función de transferencia racional de la siguiente forma:

$$H(s) = \frac{\sum_{i=0}^{m} b_i s^i}{1 + \sum_{i=1}^{n} a_i s^i} \quad \textbf{(1.76)}$$

El problema radica en determinar los coeficientes a_i y b_i o de forma equivalente los ceros z_i y los polos $p_i$de H de tal forma que se consigan las especificaciones de la magnitud de respuesta cuadrada definida como:

$$h^2(\omega) = |H(i\omega)|^2 = H(s)H(-s)|_{s=i\omega} \quad \textbf{(1.77)}$$

Siendo *h(ω)* el espectro de salida de un filtro lineal que admite un ruido blanco como entrada.

1.15.2 Filtros Butterworth [2]

Su magnitud de respuesta cuadrada viene dada por:

$$h_n^2(\omega|\omega_c) = \frac{1}{1+\left(\frac{\omega}{\omega_c}\right)^{2n}} \quad \textbf{(1.78)}$$

Siendo ω_c es la frecuencia de corte y *n* el orden del filtro.

1.16 DISEÑO DE FILTROS IIR A PARTIR DE FILTROS ANALÓGICOS.

Una forma de diseñar filtros IIR es a partir de aproximaciones discretas de los filtros analógicos. Partiendo de un filtro analógico estable y casual, el filtro digital obtenido a partir de él mediante la técnica de aproximación debería ser también estable y casual. Por ejemplo, un diseño tal que su localización de la frecuencia de corte, ancho de la banda de transición y la cantidad de error en la banda pasante y de rechazo. La aproximación debería preservar estas especificaciones de diseño.

$$H(z)\Big|_{z=\epsilon} = \frac{1}{T}\sum_{k=-\infty}^{\infty} H\left(s + j\frac{2\pi k}{T}\right) \quad \textbf{(1.79)}$$

Como vemos consiste en la superposición de las variaciones de *H(s)* a lo largo de los ejes *jω*, pudiendo ocurrir Aliasing si el filtro analógico no tiene la banda limitada. Existen técnicas de aproximación de filtros analógicos para evitar el problema del Aliasing, que son los conocidos como aproximación de la derivada y aproximación de la integral. La mayoría de los filtros de interés sus magnitudes permanecen constante y no les afecta la frecuencia distorsionadora.

1.17 DISEÑO DE FILTROS PASA BAJO

La transformación bilineal es la mejor para convertir filtros analógicos en digitales. Si consideramos que:

$$s = \frac{2j}{T}\tan(\omega/2) = \sigma + j\Omega \qquad \textbf{(1.80)}$$

La frecuencia *ω* para el filtro digital le corresponde una para el filtro analógico que es:

$$\Omega = \frac{2}{T}\tan(\omega/2) \qquad \textbf{(1.81)}$$

Para el diseño de un filtro digital pasa bajo con una frecuencia de corte ω_c se requiere una frecuencia de corte de diseño del filtro analógico ω_c:

$$\Omega_c = 2\tan(\omega_c/2) \qquad \textbf{(1.82)}$$

Cualquiera de los filtros pasa bajo analógicos pueden ser usadas para obtener el diseño de un filtro digital pasa bajo.

1.18 BANCO DE FILTROS Y EL ALGORITMO DE LA PIRÁMIDE

Pasemos ahora al análisis de los datos de V_j espacios y W_j. Si empezamos por la aproximación de la función de datos *f(t)* para A_0(t) = a_j(t) con un *j* razonablemente grande, podemos realizar el nivel de análisis wavelet completo.

En primer lugar, se descomponen A_0(t) una vez para tener

$$A_0(t) = \sum_{k=-\infty}^{\infty} a^{(0)}(k)\upsilon_{jk}(t) = A_1(t) + D_1(t), \qquad \textbf{(1.83)}$$

Dónde:

$$A_1(t) = \sum_{k=-\infty}^{\infty} a^{(1)}(k)\upsilon_{j-1k}(t), \qquad \textbf{(1.84)}$$

$$D_1(t) = \sum_{k=-\infty}^{\infty} d^{(1)}(k)u_{j-1k}(t), \qquad \textbf{(1.85)}$$

Esta es la consecuencia de $V_j = V_{j-1} \oplus W_{j-1}$, y $\upsilon_{jk}(t)$ para todos los enteros *k* que forman la base fijada para V_j y $u_{jk}(t)$ para todos los enteros *k* que constituyen la base fijada para W_j. Los coeficientes de la expresión anterior se obtienen a partir de

$$a^{(1)}(k) = \langle \upsilon_{j-1k} | A_1 \rangle = \langle \upsilon_{j-1k} | A_0 \rangle = \sum_{l=-\infty}^{\infty} h(1-2k)a^{(0)}(l), \qquad \textbf{(1.86)}$$

$$d^{(1)}(k) = \langle u_{j-1k} | D_1 \rangle = \langle u_{j-1k} | A_0 \rangle = \sum_{l=-\infty}^{\infty} g(1-2k)a^{(0)}(l), \qquad \textbf{(1.87)}$$

Tan pronto como tenemos la función de escalado y sus correspondientes filtros especificados, también tenemos la onda de la segunda dos y la escala de relación con los coeficientes $a^{(0)}(0) = \langle v(t-k) | f(t) \rangle$.

Un filtro digital se define por la convolución, que es un operador matemático que transforma dos funciones f y g en una tercera función que en cierto sentido representa la magnitud en la que se superponen f y una versión trasladada e invertida de g.

$$a * b(t) = \sum_{k=-\infty}^{\infty} a(k) b(t-k) = \sum_{k=-\infty}^{\infty} a(k-1) b(t), \qquad \textbf{(1.88)}$$

Donde *a(t)* se llama el filtro y la secuencia de tiempo *b(t)* es la función de los datos que se filtra por *a(t).* De las ecuaciones **(1.86)** y **(1.87)**, se ve que los filtros asociados con el análisis wavelet de tiempo invertido y saltar cada punto de datos a otros durante el filtrado, que se llama downsampling es decir la técnica que reduce la frecuencia de muestreo de una señal con el fin de disminuir la cantidad de datos adquiridos. Por lo tanto se puede representar cada nivel de análisis por dos operaciones, el filtrado y disminución de la resolución.

La disminución de la resolución es equivalente a la conversión de la función *x(k)* en *x(2k).* El análisis se puede continuar con el mismo par de filtros para el siguiente nivel con una descomposición de $A_1(t)$ en $A_2(t)$ y $D_2(t)$, y así sucesivamente, formando una pirámide multietapa de banco de filtros. Podemos diseñar un algoritmo piramidal para llevar a cabo los n niveles de

descomposiciones para obtener $\{a^{(n)}, d^{(n)}, ..., d^{(1)}\}$, para *n* ≥ *1.* La transformada inversa (síntesis) se puede obtener con el mismo par de filtros sin invertir el tiempo. Desde la expansión de la ecuación **(1.83)**, tenemos

$$a^{(0)}(k) = \langle \upsilon_{jk}(t) | A_0(t) \rangle$$
$$a^{(0)}(k) = \sum_{k=-\infty}^{\infty} a^{(l)}(l) h(k-2l) + \sum_{k=-\infty}^{\infty} d^{(l)}(l) g(k-2l), \quad \textbf{(1.89)}$$

Esta ecuación puede ser interpretada como una combinación de un sobremuestreo y una operación de filtrado. El sobremuestreo es equivalente a la conversión de la función *x(k)* en *x(k/2)* para incluso *k* y en cero para *k* impar. Por supuesto, el proceso puede continuarse hasta el segundo nivel de la síntesis, el tercer nivel de síntesis, y así sucesivamente, hasta que se tienen la secuencia de tiempo completo reconstruida.

Al igual que la DFT, la transformada wavelet discreta también se puede utilizar en espacios de más dimensiones.

CAPITULO II

2 DESARROLLO DE EJEMPLOS APLICANDO LA TRANSFORMADA DE FOURIER UTILIZANDO EL SOFTWARE SCILAB

2.1 *DESCRIPCIÓN GENERAL*

2.1.1. INTRODUCCIÓN

En este capítulo se describirá el principio de funcionamiento del Software Scilab, se estudiaran las principales funciones utilizadas para el análisis espectral y se ilustraran sus potencialidades mediante ejemplos [12].

El interfaz básico de SCILAB que se utilizara es el siguiente:

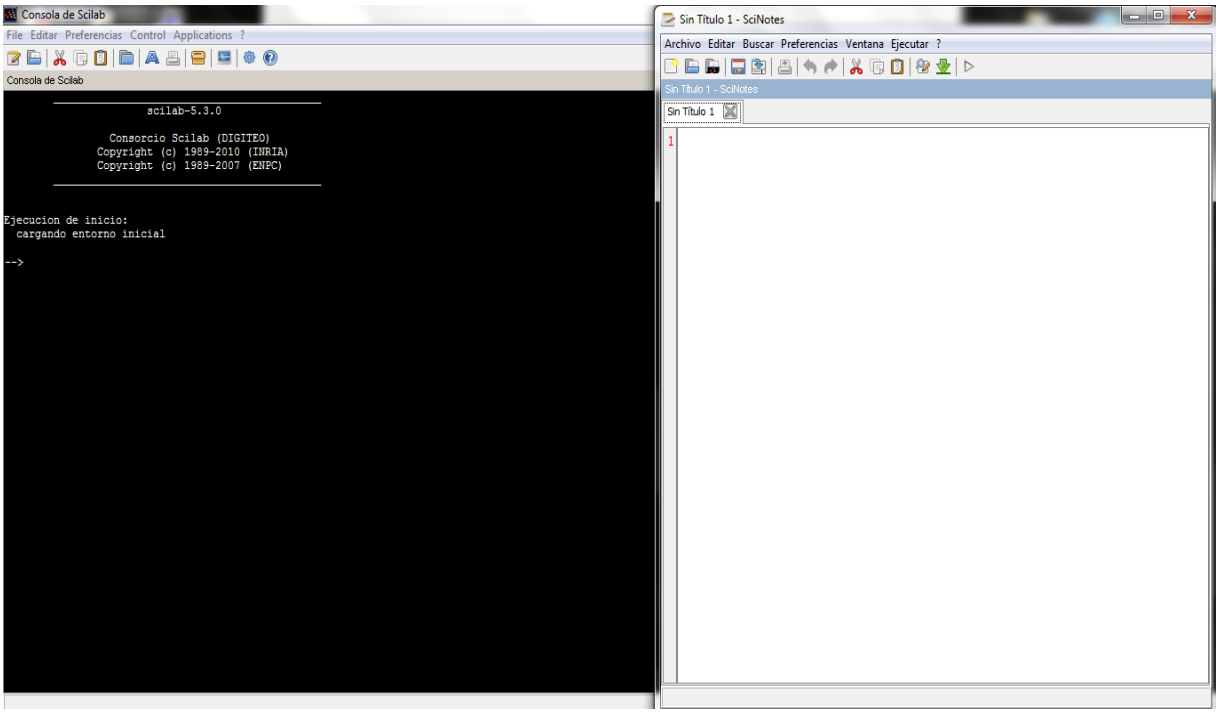


Figura 2.1 Consola de Software SCILAB de código abierto para computación científica.

2.2 SCILAB Y EL PROCESAMIENTO DE SEÑALES

Entre las funciones para el tratamiento de señales, SCILAB posee una gran variedad de funciones, como por ejemplo la convolución, la inversión de matrices, la FFT, DFT, entre otras.

Las herramientas encontradas en SCILAB permiten el análisis e implementación de filtros digitales, incluyendo la respuesta en frecuencia, retardo de grupo y retardo de fase. Además la implementación de filtros puede ser directa utilizando técnicas en el dominio de la frecuencia basadas en la FFT. También se establece el diseño de filtros. Otras propiedades de SCILAB son el diseño de filtros FIR y su

optimizado, el procesamiento de la FFT incluyendo la transformada base-2 y su inversa, y las transformadas para potencias que no sean de dos, estimación espectral (espectro de potencia), filtro optimo y suavizado [11]-[12].

2.3 TRATAMIENTO DE SEÑALES

En SCILAB es posible llevar a cabo distintas funciones que permiten realizar la convolución y representación de sistemas lineales a través de su función de transferencia o su espacio de estados.

Entre algunas de las funciones generadoras de señales SCILAB posee las siguientes: • sin • cos • asin y acos • exp • sqrt

Existen otras funciones que desarrollan otras tareas como: • disp. • Title

Debido al gran número de funciones que posee SCILAB solo veremos las más importantes funciones y aplicaciones en el procesamiento de señales, las cuales se muestran a continuación.

2.3.1 Frecuencia De La Señal

Modificaremos la frecuencia de la señal senoidal multiplicando la variable t (dentro del paréntesis del seno).

También podemos observar que el gráfico que une los puntos del muestreo cada vez se parece menos a la onda inicial contínua [2].

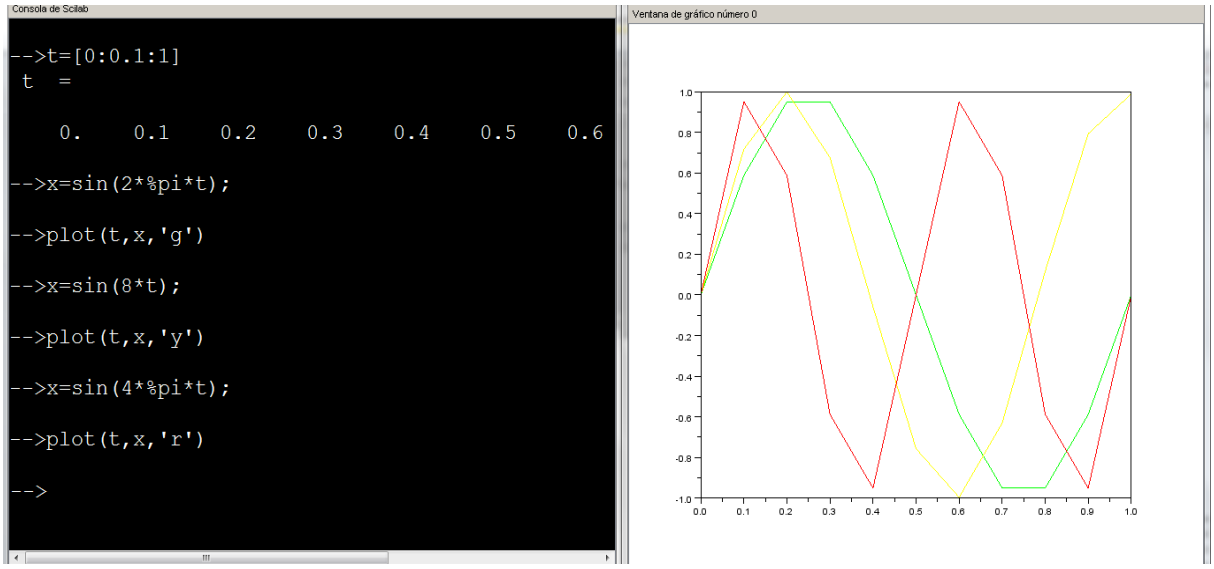


Figura 2.2 Representación de la frecuencia de una señal, utilizando 7 y 10 datos generados en el Software.

Para una mejor representación de la curva de la función, añadiremos más puntos, vamos a probar con 100 intervalos en lugar de 10.

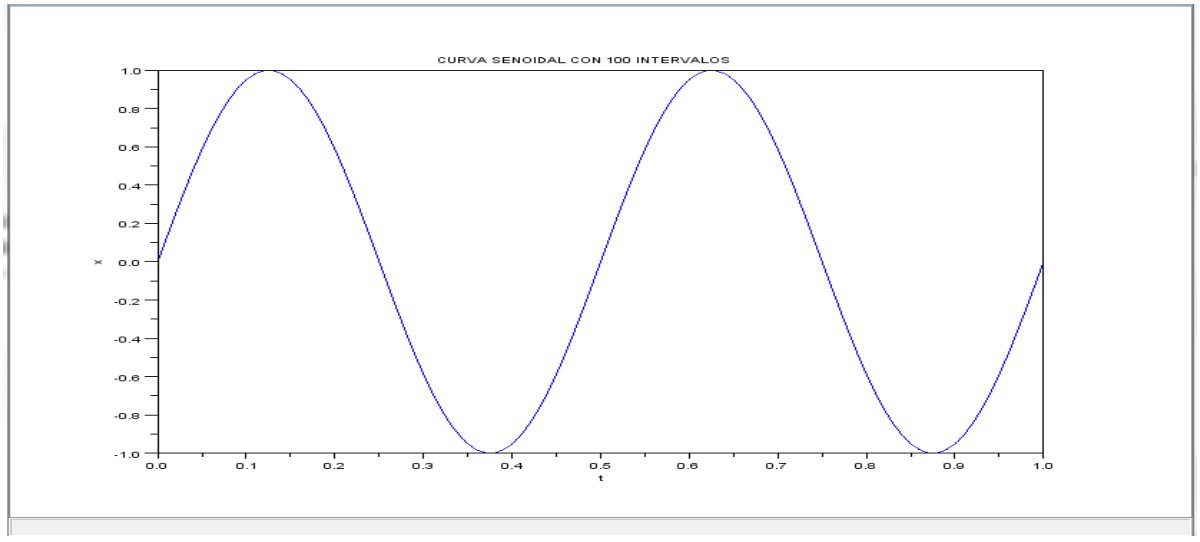


Figura 2.3 Modificación de la frecuencia de una señal, utilizando 100 datos generados en el Software.

2.3.2. Amplitud de la Señal y Componente Continua

Si multiplicamos el valor de la función seno por una constante, estaremos multiplicando cada uno de los valores de la señal por ese número, y estaremos cambiando la amplitud de variación de la curva. También podemos representar gráficamente varias curvas juntas usando plot (t,x1,t,x2) [2].

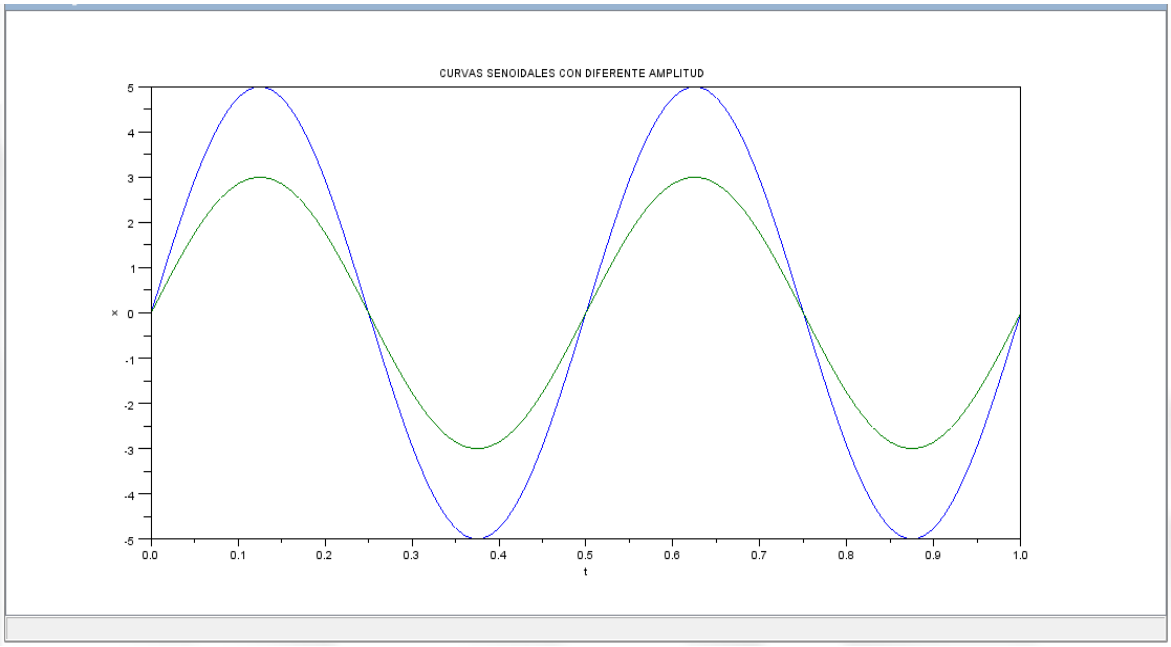


Figura 2.4 Representación de la variación de la Amplitud en una misma función.

2.3.3. Modificación de la Fase

Si ahora sumamos un valor constante a la escala de tiempo (dentro del paréntesis de la función seno) veremos que estamos desplazando la señal, que ya no comenzará en cero.

Esa diferencia temporal la llamaremos fase (a veces también desfase) [2].

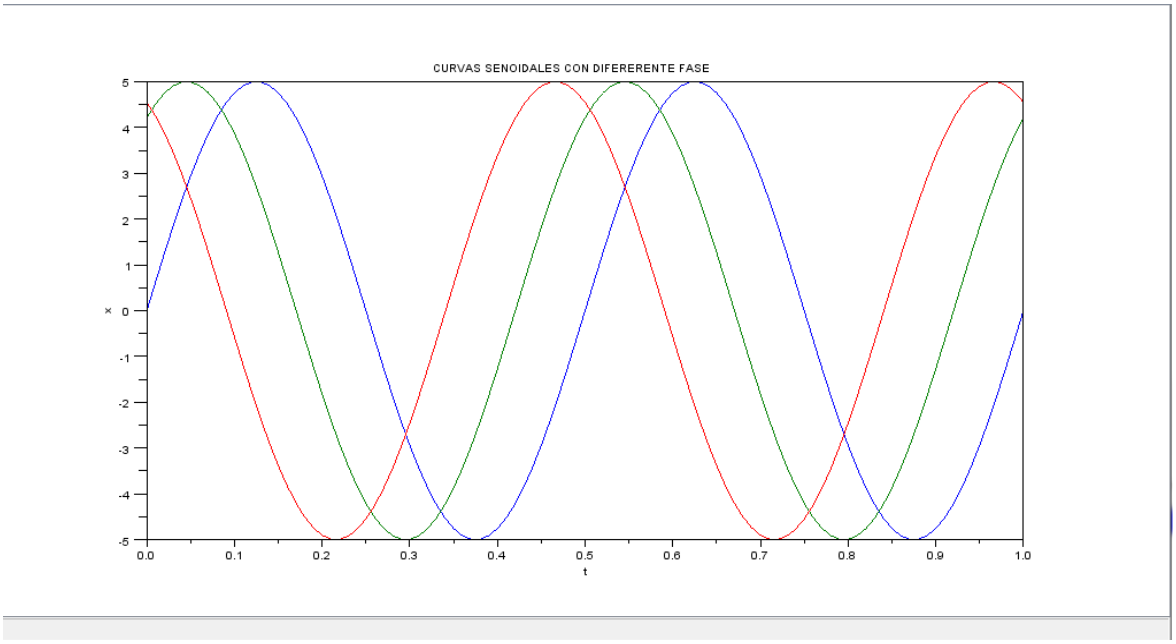


Figura 2.5 Representación de la variación del tiempo en la misma función periódica.

La última que hemos dibujado tiene Amplitud 5, Frecuencia 2 y Fase 2. Y como siempre *t* puede ser un número o bien una tabla de valores para la cual queremos visualizar el resultado y obtener un gráfico.

2.3.4. Suma de Ondas Senoidales

Al igual que sumamos números, en SCILAB podemos sumar tablas. La única condición es que deben tener el mismo número de elementos, para así ser sumados uno a uno.

Si las tablas representan señales con la misma escala de tiempo (este último es muy importante), estaremos sumando señales [2].

2.3.5. Series de Fourier

La siguiente figura muestra una función periódica de periodo T = 2; Para verificar que la función es en efecto periódica, tratamos las siguientes evaluaciones: *f(0),f(2),f(4).*

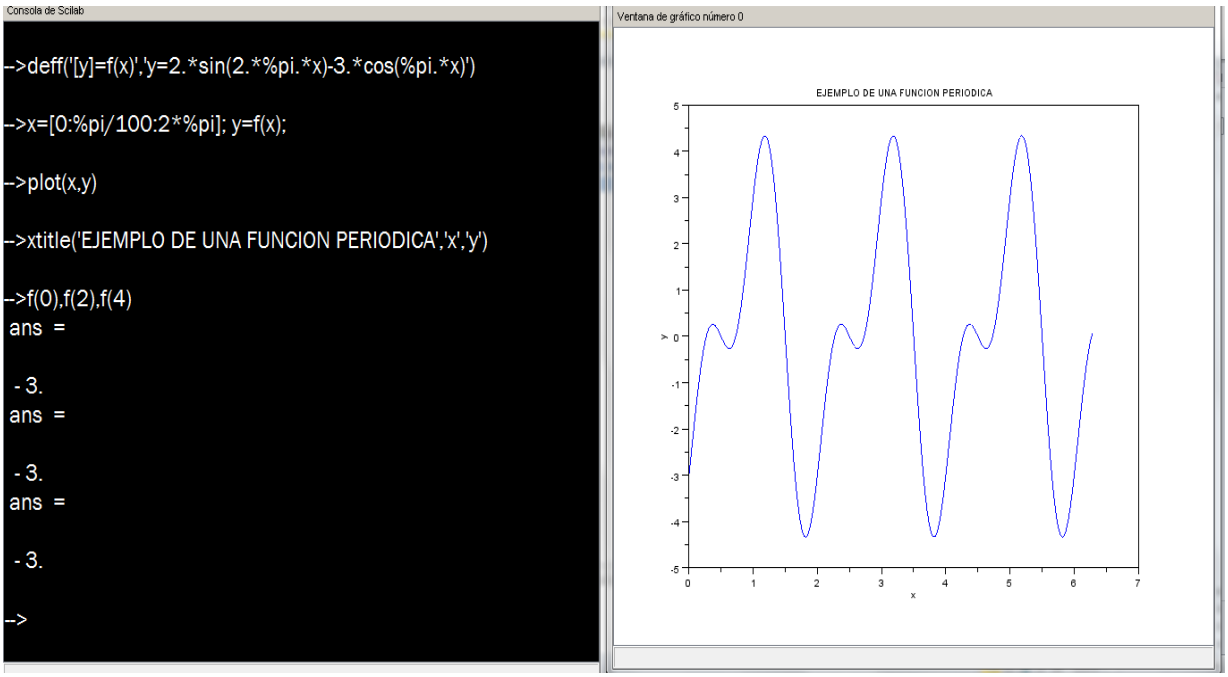


Figura 2.6 Verificación de una función periódica en un intervalo de tiempo determinado.

2.4 REPRESENTACION DE DATOS

2.4.1. Representación de un Sistema Lineal por su Función de Transferencia (FT)

El poder representar un sistema lineal por su función de transferencia nos permite tomar distintas decisiones sobre el comportamiento de dicho sistema así como la posibilidad de modificar este agregando filtros u otros modificadores al sistema. En SCILAB se utilizan las expresiones racionales polinómicas para describir funciones de transferencia. Estas funciones de transferencia pueden representar señales en el dominio temporal continuo o discreto. Pero como ya sabemos las señales son sistemas continuos que pueden ser transformados en discretos mediante el muestreo. Para describir la función de transferencia en SCILAB se utiliza el comando ***syslin*** el cual presenta la siguiente sintaxis.

--> sl=syslin(dominio, numerador, denominador);

En la sintaxis anterior se define el dominio de la función de transferencia, el numerador de la función de transferencia y el denominador de la misma.

2.4.2. Representación de un Espacio de Estados (EE) de un Sistema Lineal

Existen dos tipos de representaciones clásicas en el Espacio de Estados, la continua y la discreta. Siendo A, B, C y D matrices y Xo un vector. Para

representar en el Espacio de Estados, SCILAB vuelve a utilizar la función syslin, pero de la siguiente forma:

--> sl=syslin(dominio,a,b,c,[,d[,x0]])

Siendo el valor de retorno de sl una lista con los siguientes elementos:

-->S=list('lss',a,b,c,d,x0,dominio)

2.5 CAMBIO DE REPRESENTACIÓN

2.5.1. Paso de Función de Transferencia (FT) al Espacio de Estados (EE) y Viceversa

En los problemas normales de tratamiento de señales nos puede interesar una u otra representación, que son totalmente equivalentes. Para pasar de la función de transferencia a espacio de estados se usa el comando ***tf2ss*** cuya sintaxis es:

-->sl=tf2ss(h)

Donde sl es el espacio de estados esperado y h es la función de transferencia a transformar. Para pasar de espacio de estados a función de transferencia se usa el comando **ss2ft** que muestra la sintaxis:

-->h=ss2ft (sl)

2.5.2. Discretización de Sistemas Continuos

En SCILAB un sistema lineal continuo en el tiempo representado por su EE o su FT, puede ser convertido en discreto dentro del mismo dominio temporal, esto por medio de la función dscr. La sintaxis de la función dscr es:

-->[f,g[,r]]=dscr(a,b,dt[,m])

En donde a y b son matrices asociadas al espacio de estados continuo, mientras que f y g las matrices resultantes para el espacio de estados discreto. Si el argumento de entrada en la función dscr fuera el espacio de estados continuo como una lista sl, seria:

-->[sld[,r]]=dscr(sl,dt[,m])

Siendo sld la lista que se representa en el espacio de estado discreto. En el caso en que el espacio de estados continuos este representado por su función de transferencia h, esta será el argumento de entrada de la función hd (FT del discreto) [2]:

-->[hd]=dscr(h,dt)

2.5.3. Filtrado y su Representación Gráfica

El filtrado de señales mediante filtros representados por su espacio de estados o por su función de transferencia se realiza con la función ***flts*** la cual tiene dos formatos: En el caso de un sistema lineal dado por la ecuación de estados la sintaxis es:

-->[y[,x]]=flts(u,sl[,0])

Cuando el sistema lineal está dado por su función de transferencia:

-->y=flts(u,h[,past])

SCILAB utiliza el comando plot para representar gráficamente cualquier señal. Ahora veremos un ejemplo del uso de los comandos plot y flts.

Para ello generamos dos señales sinusoidales X1 y X2, luego definimos un filtro de respuesta impulsional finita wfir, sumamos las dos señales de entrada cuyo resultado es la señal x, obtenemos la función de transferencia del filtro la cual llamaremos hz, aplicamos el filtro a la señal x y obtenemos la salida [2]-[15].

yhz: La señal de salida del filtro yhz es la mostrada en la siguiente figura:

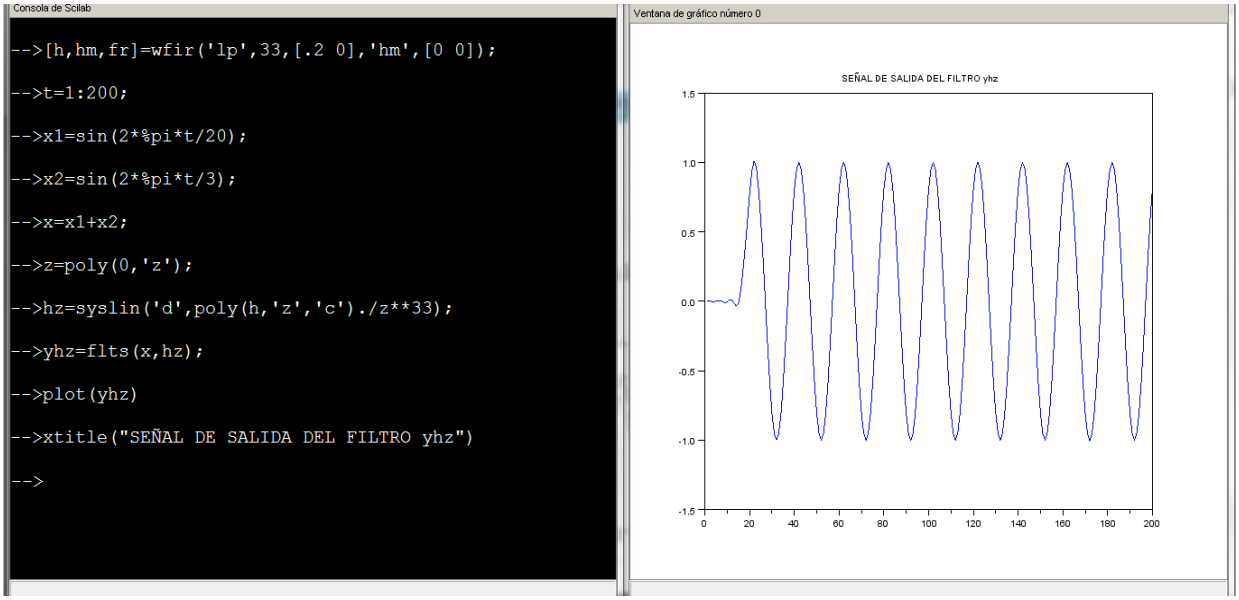


Figura 2.7 Representación sumatoria de dos funciones de onda y su señal de salida de filtro.

La señal de entrada X del filtro se muestra en la siguiente figura:

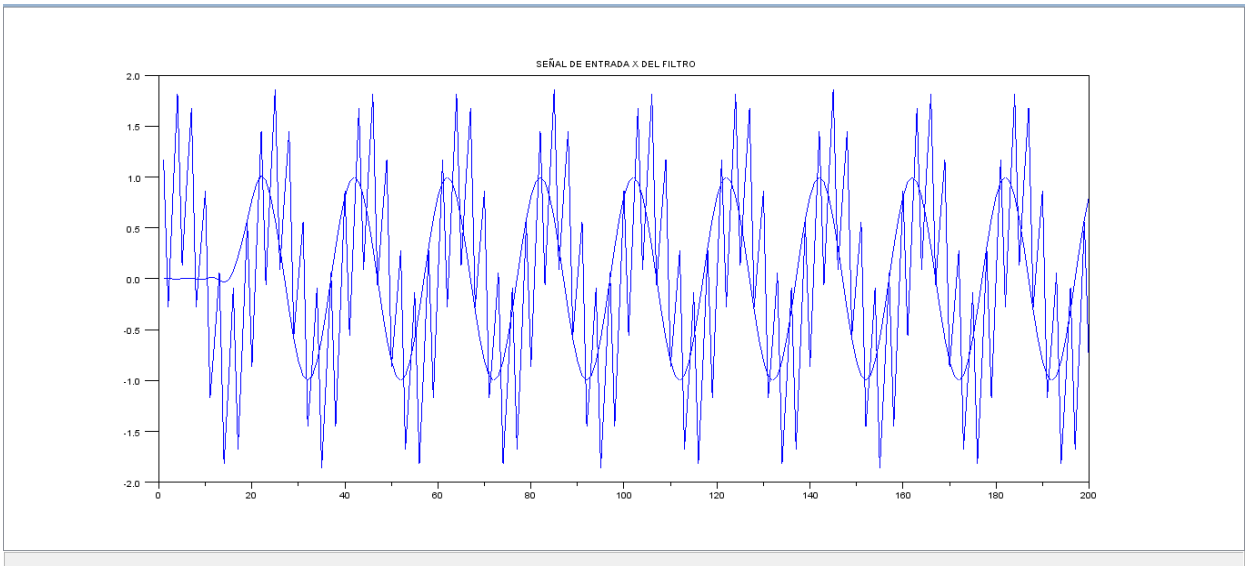


Figura 2.8 Representación de la sumatoria de dos funciones de onda y su señal de entrada X del filtro.

2.6 LA FFT Y LA DFT

Scilab para el cálculo de la FFT utiliza la función fft:

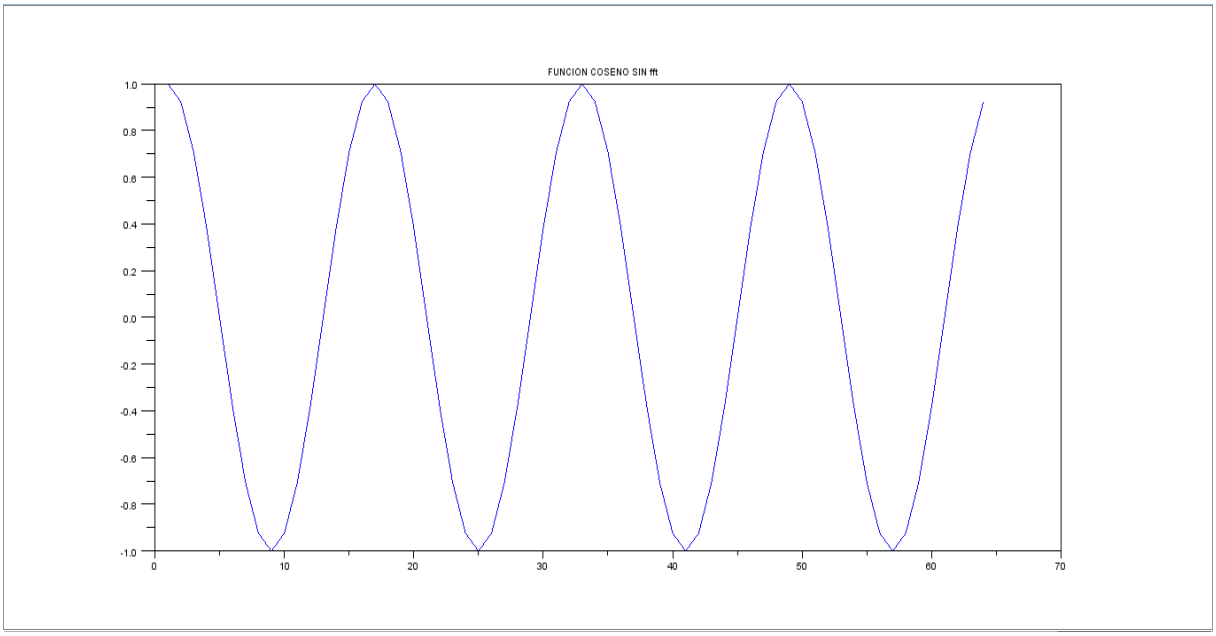


Figura 2.9 Muestra la señal de la función coseno sin haber aplicado la FFT.

2.7 CONVOLUCIÓN

Scilab estima la convolución usando el método de la suma coincidente. Este es un algoritmo basado en la definición de convolución y es implementado por un producto de polinomios, el cual es mucho más eficiente si x es un arreglo extenso.

Para poder realizar la convolución SCILAB tiene la función ***convol*** la cual, presenta tres tipos de sintaxis distintas.

-->y=convol(h,x)

-->[y]=convol(h,x)

-->[y,e1]=convol(h,x,e0)

En donde:

h: vector de la primera secuencia de entrada.

x: vector de la segunda secuencia de entrada.

e0: un vector, ultima cola de la suma en solaparse (no se usa en el primer llamado).

e1: nuevo vector de la suma en solaparse (no usado en el último llamado)

2.8 DISEÑO DE FILTROS CON SCILAB

La teoría general del diseño de Filtros Digitales nos conduce al agrupamiento de las funciones para el diseño de estos.

2.8.1. Diseño De Filtros de Respuesta Finita (FIR)

Esta función puede tener dos tipos de sintaxis, la primera de ellas:

--> [wft,wfm,fr]=wfir()

Donde el paréntesis es una parte requerida del nombre. Este tipo de función es interactiva y propone el uso de parámetros de entrada requeridos, en relación al tipo de filtro (lp = paso bajo, hp = paso alto bp = paso banda, sp = rechaza banda).

La segunda sintaxis de la función es la siguiente.

--> [wft,wfm,fr] =wfir(ftype,forder,cfreq,wtype,fpar)

Esta forma de la función no es interactiva y todos los parámetros de entrada deben ser dados como argumentos de la función.

Ftype: indica el tipo de filtro a construir y puede tomar valores de lp, hp, bp y sb representando paso bajo, paso alto, paso banda y rechaza banda respectivamente.

Forder: es un integer positivo que da el orden del filtro elegido.

Cfreq: es un vector de 2 dimensiones para el cual solo el primer elemento es usado para el caso de filtros paso bajo y paso alto. Bajo estas circunstancias cfreq(1) es la frecuencia de corte en Hertz del filtro escogido. Para el pasa banda y el rechaza banda ambos elementos de cfreq son usados, el primero de ellos es la frecuencia baja de corte y el segundo la frecuencia alta de corte del filtro. Ambos valores de cfreq deben estar entre 0 y 0.5 correspondiendo a los posibles valores de una respuesta de frecuencia discreta.

Ftype indica el tipo de ventana escogida y puede tomar valores de re, tr, hm, hn, kr, o ch representando respectivamente las ventanas rectangular, triangular, Hamming, Hanning, Kaiser y Chebyshev.

Fpar es un vector de dos valores para el cual solo el primer valor es usado para el caso del uso de la ventana Kaiser y solo en la ventana Chebyshev se usan ambos elementos.

En el caso de la ventana Kaiser el primer elemento indica el cambio relativo entre el lóbulo principal de la respuesta en frecuencia de la ventana y el ancho del lado del lóbulo (debe ser un entero positivo). Para la ventana Chebyshev se puede especificar el ancho del lóbulo principal de la ventana o el tamaño de los lados del lóbulo. El primer elemento de fpar indica el tamaño del lado del lóbulo y toma valores con rangos entre 0 y 1 y el segundo elemento da el ancho del lóbulo principal el cual puede tener valores entre 0 y 0.5.

Debido a las propiedades lineales de los filtros de fase lineal FIR no es posible diseñar un filtro paso alto de un ancho regular o un filtro rechaza banda [2]-[15].

2.9 MUESTREO

Dada una señal continua *x(t)*, el proceso de muestreo consiste en tomar muestras equiespaciadas de la señal mediante otra señal denominada señal muestreadora *p(t)*.

a. **Muestreo Ideal:** En este caso la señal muestreadora p(t) es un tren de deltas de periodo Ts.

b. **Teorema De Nyquist:** Dada una señal limitada en banda $X(\omega) = 0$

$\forall|\omega| > \omega m$ podemos recuperar la señal a partir de sus muestras si se cumple:

$$\omega_s \geq 2\omega_m$$

A la frecuencia $\omega_s = 2\omega_m$ se la conoce normalmente como la frecuencia de Nyquist. En la práctica debemos muestrear por encima de la frecuencia de Nyquist pues es imposible realizar un filtro ideal.

2.10 TÉCNICAS DE MUESTREO DE FRECUENCIA

Las técnicas de muestreo de frecuencia están basadas en un grupo especificado de muestras que se eligen de la respuesta en frecuencia tomada con N puntos espaciados uniformemente alrededor de un círculo unitario, donde N es la longitud del filtro.

Esto significa que una aproximación de respuesta en frecuencia continua se obtiene tomando un muestreo en frecuencia con N puntos equidistantes alrededor del círculo unitario (la frecuencia de muestreo), e interpolando entre estos obteniendo la respuesta en frecuencia continua. De esta forma el error de aproximación podría ser exactamente cero en las frecuencias de muestreo y finito entre los demás. Este factor tiene que ser relacionado a la construcción de una función continua para estas muestras.

La principal deficiencia de esta técnica es la carencia de flexibilidad al especificar la transición del ancho de banda, el cual es igual al número de muestras elegidas en el tiempo π/N y de esta manera está fuertemente relacionado a N.

Esto genera un pequeño rizado cerca del flanco de la banda.

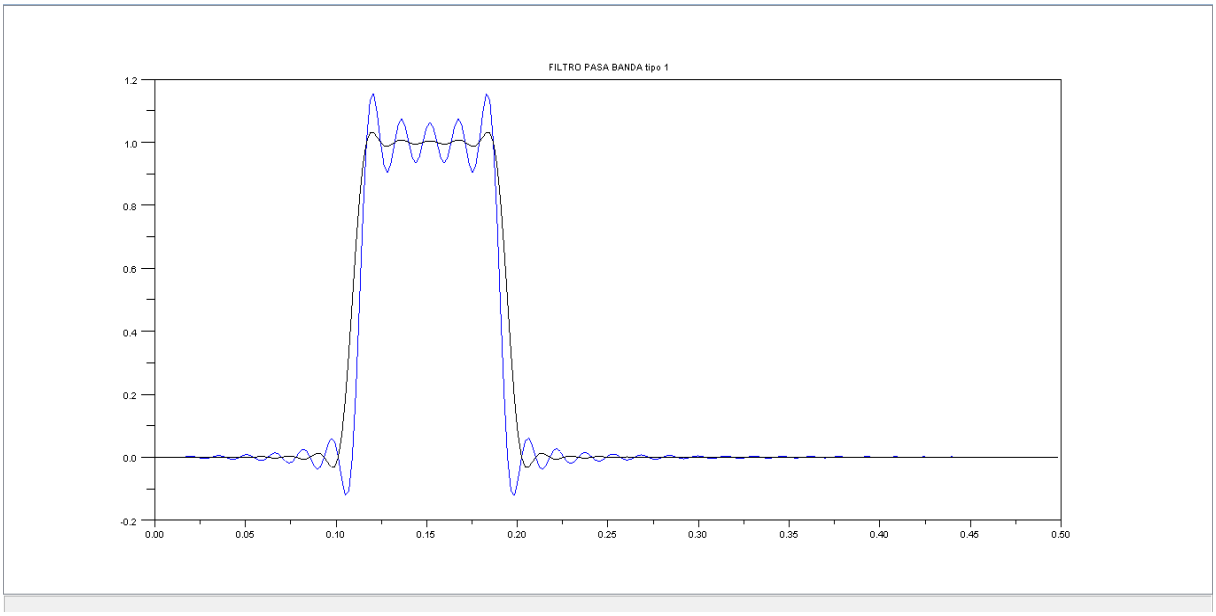


Figura 2.10 La figura muestra la respuesta obtenida por un filtro paso banda.

La primera respuesta es obtenida sin muestras en la banda de transición y la segunda con un muestreo de magnitud 0.5 en cada una de las bandas. Dependiendo de dónde ocurra la frecuencia de muestreo, se dan dos distintos grupos de frecuencias de muestreo correspondientes a los llamados filtros FIR tipo 1 y tipo 2.

El tipo de diseño deseado depende de la aplicación. Por ejemplo, un corte de banda puede ser más cerrado en los puntos de frecuencia muestreada en un filtro tipo 1 que en un filtro tipo 2.

La siguiente figura muestra estos puntos para un filtro paso bajo con una longitud 64 y sin muestrear en la banda de transición [4].

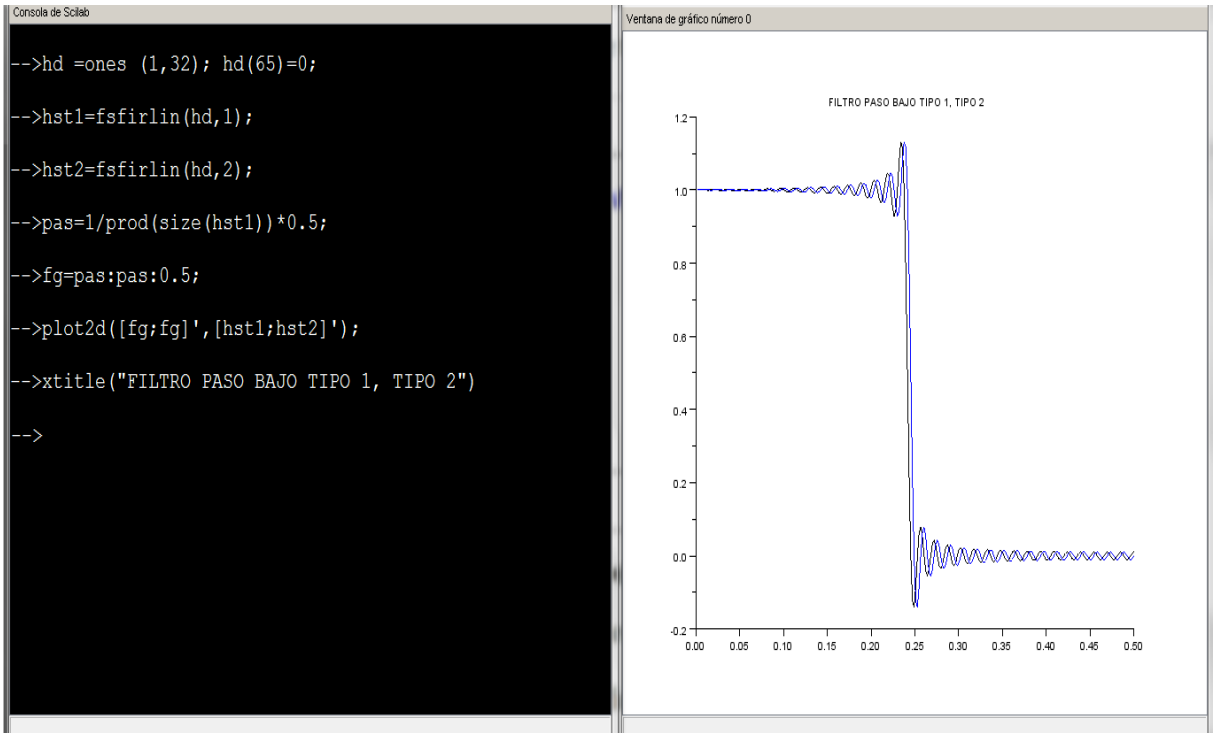


Figura 2.11 La figura muestra la respuesta obtenida por un filtro pasa bajo tipo 1 y tipo 2. La línea rellena da la respuesta aproximada para el filtro lineal FIR tipo 1.

2.11 EJEMPLOS DE APLICACIÓN

2.11.1. Programa 1 (Generación de una Secuencia Exponencial Compleja)

En este primer ejemplo se muestra como generar algunas señales y sus graficas en forma discreta, en si se generan secuencias exponenciales complejas, real, imaginaria y señales de ruido. Además se realiza el suavizamiento de una señal por un filtro de promedio móvil.

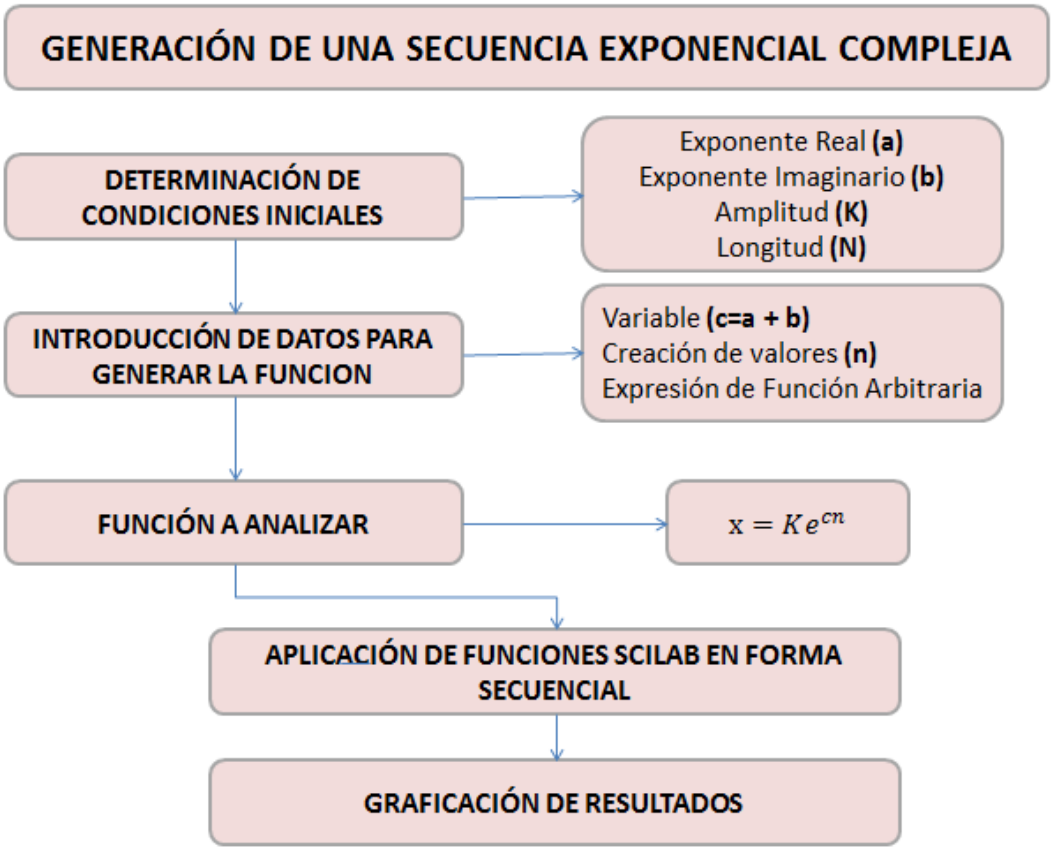


Figura 2.12 Esta secuencia de pasos se utilizar para generar una secuencia exponencial compleja o real, estos resultados se los puede observar en **Anexos de PROGRAMA 1** [4].

2.11.2. Programa 2 (Generación de una Secuencia Exponencial Real)

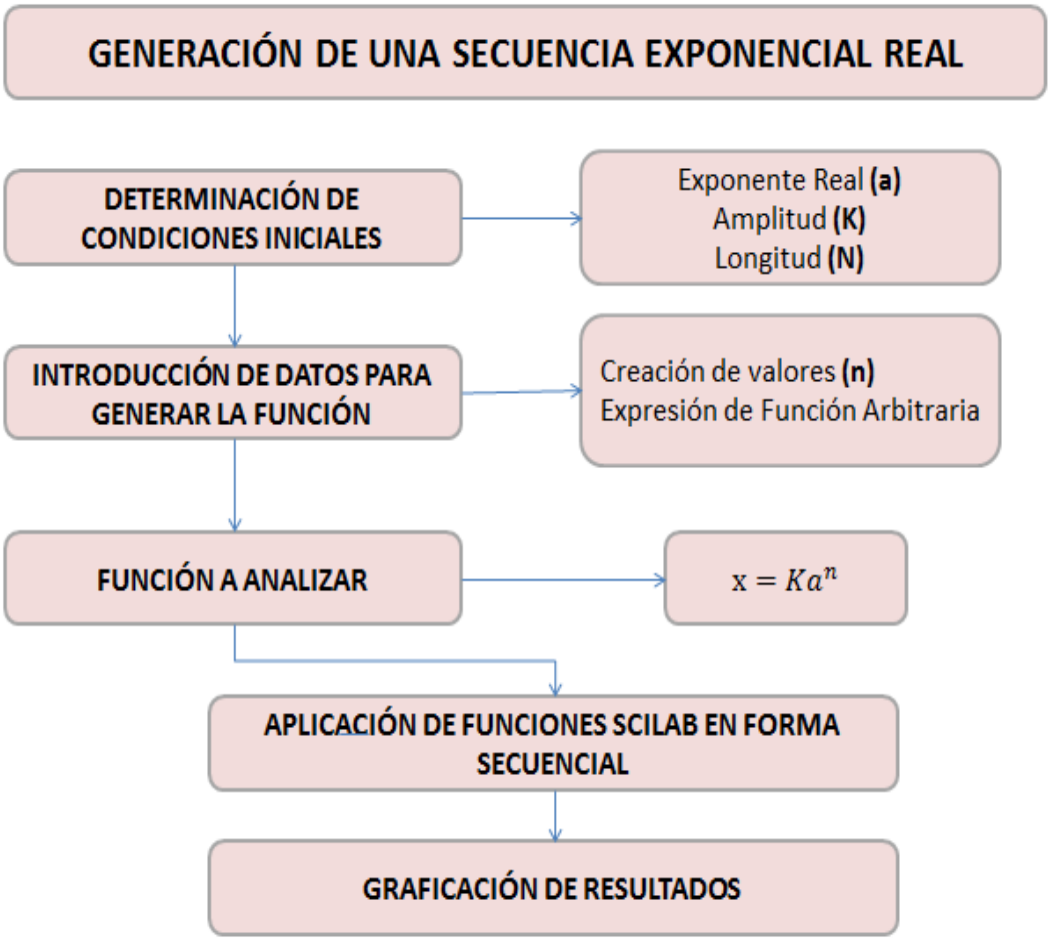


Figura 2.13 Esta secuencia de pasos se utilizar para generar una secuencia exponencial real, estos resultados se los puede observar en **Anexos de PROGRAMA 2** [4].

2.11.3. Programa 3 (Generación de Señales y Generación de Ruido Aleatorio)

Es conveniente reducir el efecto del ruido d[n] para obtener un mejor estimado de s[n] de x[n]. Para este fin, el filtro de promedio móvil da resultados de cuantificación razonablemente buenos.

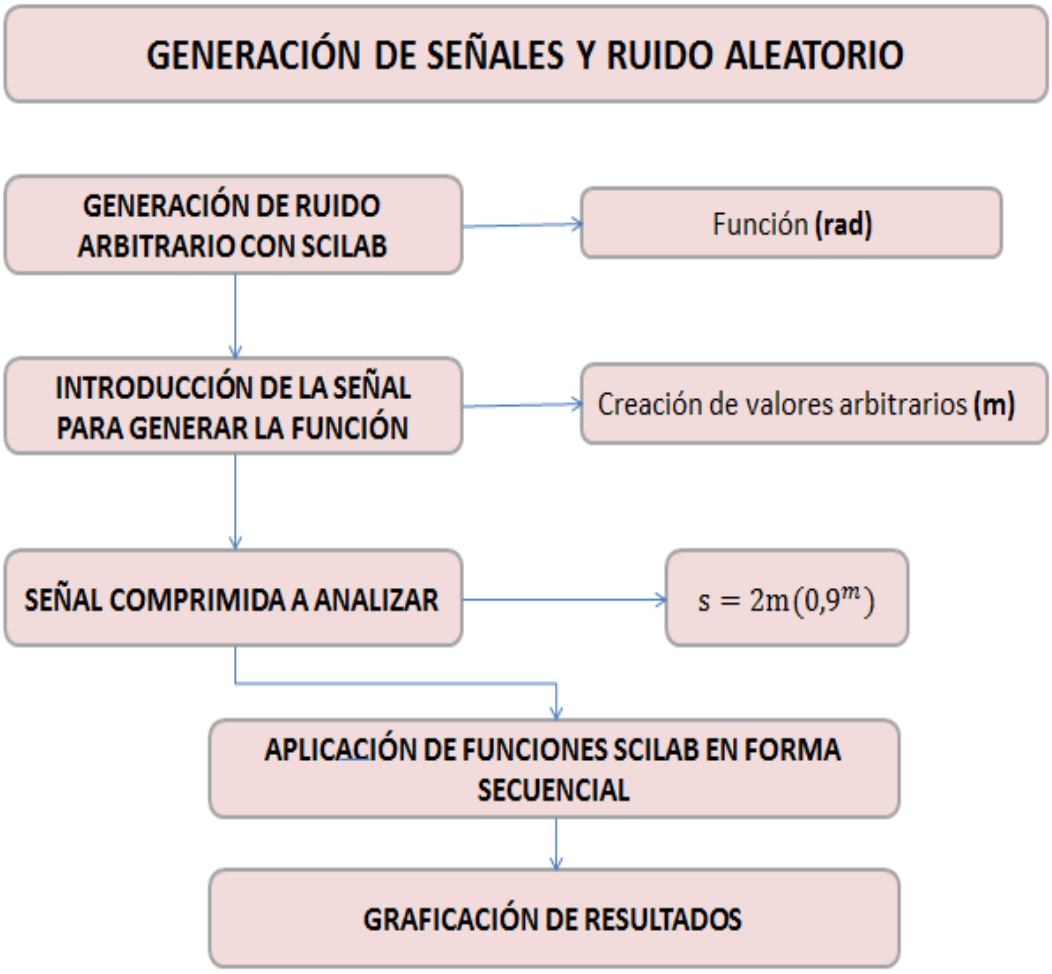


Figura 2.14 Esta secuencia de pasos se utiliza para generar una señal con ruido aleatorio, los resultados se los puede observar en **Anexos de PROGRAMA 3** [4].

2.11.4. Programa 4 (Suavizamiento de una Señal por un Filtro de Promedio Móvil)

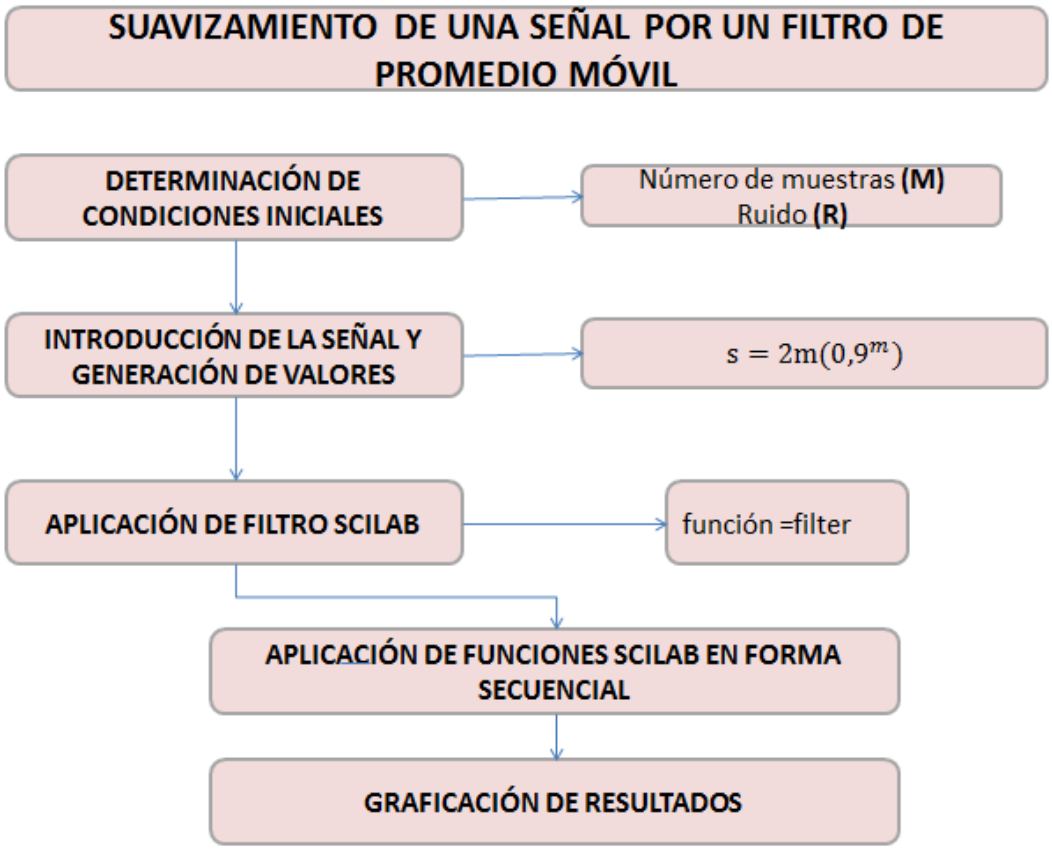


Figura 2.15 El siguiente programa se usa para generar la salida 'suavizada' y[n] de la señal original corrompida con ruido usando el sistema de promedio móvil [4].

Durante su ejecución, el programa solicita los datos de entrada, que son el número deseado M de muestras de entrada a ser promediadas. Para enfatizar el efecto del ensuavizamiento del ruido, las señales de tiempo discreto se grafican

como curvas continuas usando la función plot, estos resultados se los puede observar en **Anexos de PROGRAMA 4**.

2.11.5. Programa 5 (Calculo de la Auto correlación de una Secuencia Sinusoidal Corrompida por Ruido)

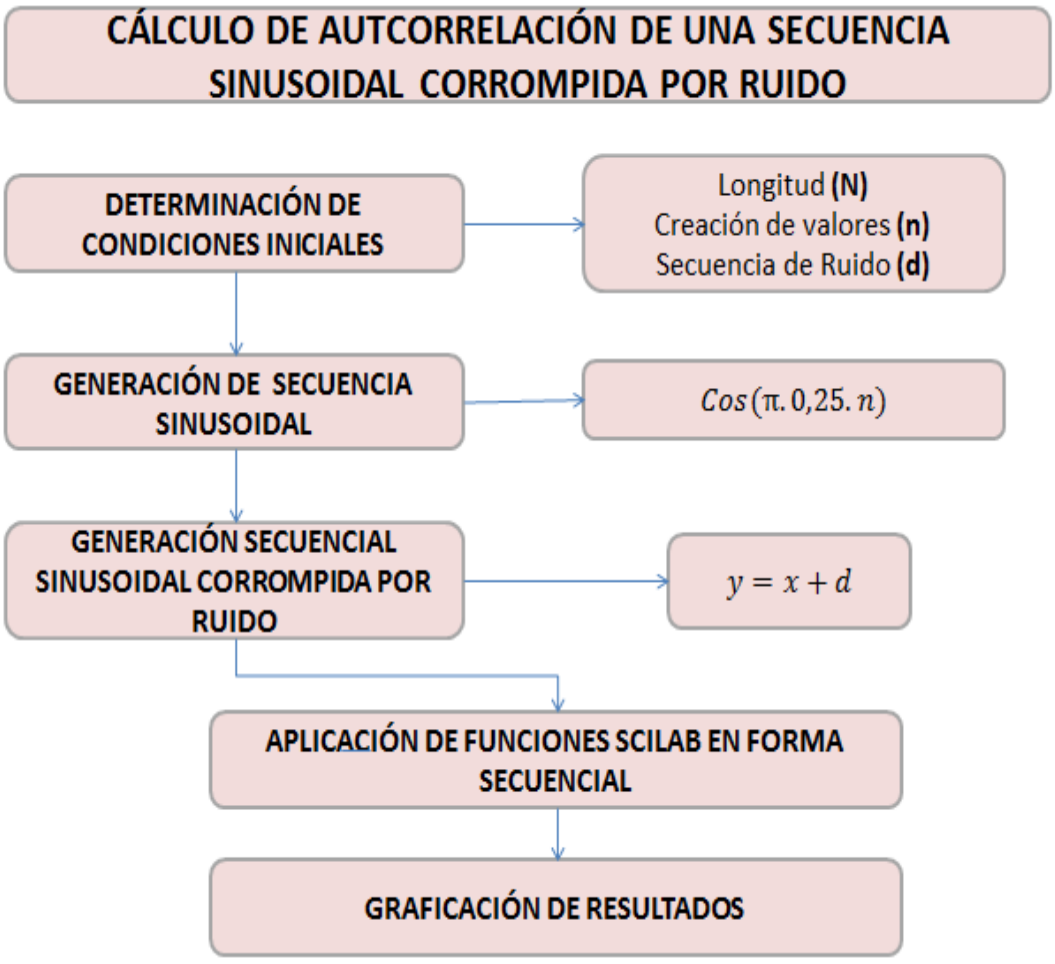


Figura 2.16 Siguiendo las instrucciones anteriores se genera el cálculo de autocorrelación de una secuencia sinusoidal corrompida por ruido, los resultados se los puede observar en **Anexos de PROGRAMA 5** [4].

2.11.6. Programa 6 (Transformada de Fourier de Tiempo Discreto)

El siguiente programa puede emplearse para determinar los valores de la DTFT de una secuencia real descrita como una función racional de $e^{-j\omega}$.

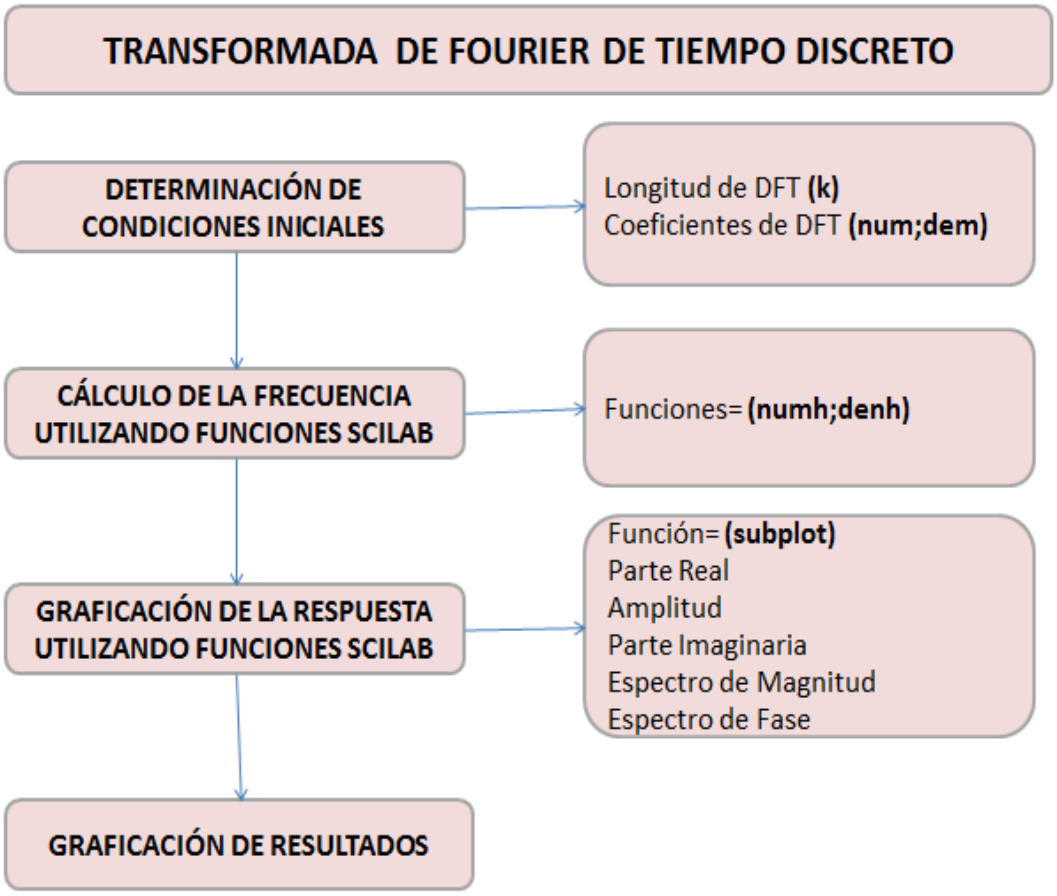


Figura 2.17 Mediante el desarrollo de esta secuencia se puede observar la DFT **Anexos de PROGRAMA 6** [4].

Los datos de entrada solicitados por el programa son el número de puntos de frecuencia *k* donde la DTFT va a ser calculada y los vectores *num* y *den* que contienen los coeficientes del numerador y del denominador de la DTFT, respectivamente, proporcionados en orden ascendente de potencias de $e^{-j\omega}$.

Estos vectores deben ser introducidos dentro de paréntesis cuadrados. El programa calcula los valores de la DTFT en los puntos de frecuencia prescritos y grafica las partes real e imaginaria, junto con los espectros de fase y magnitud. Debe notarse que debido a las relaciones de simetría de la DTFT de una secuencia real, esta es calculada únicamente en *k* valores igualmente espaciados de ω entre 0 y π.

2.11.7 Programa 7 (Transformada Discreta de Fourier)

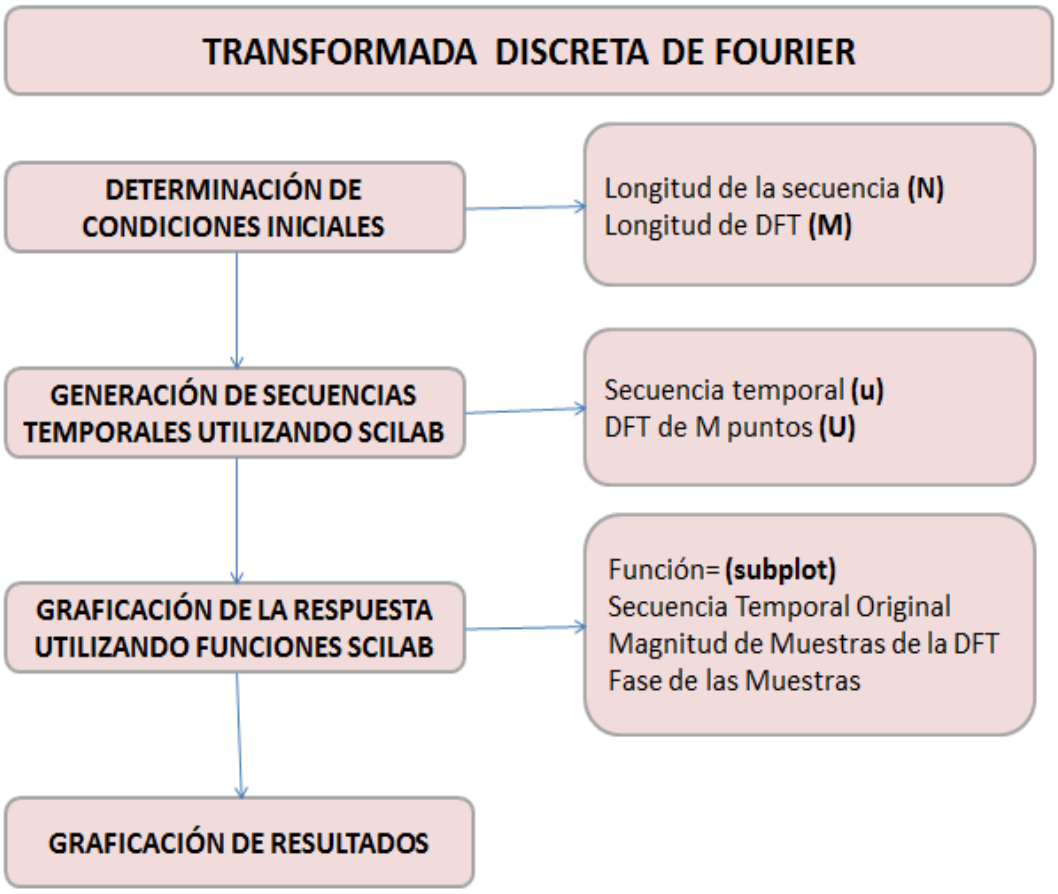


Figura 2.18 Mediante el desarrollo de esta secuencia se puede observar la DFT en caso diferente **Anexos de PROGRAMA 7** [4].

Este programa solicita los datos de entrada *N* y *M*. Para asegurar que la DFT de valores correctos, *M* debe ser mayor que o igual a *N*. Después que son introducidos tales valores, calcula la DFT de *M* puntos y gráfica la secuencia original de *N* puntos, junto con la magnitud y fase de la secuencia de la DFT se los puede observar en **Anexos de PROGRAMA 7**.

.

2.11.8. Programa 8 (Cálculo de la DFT Inversa)

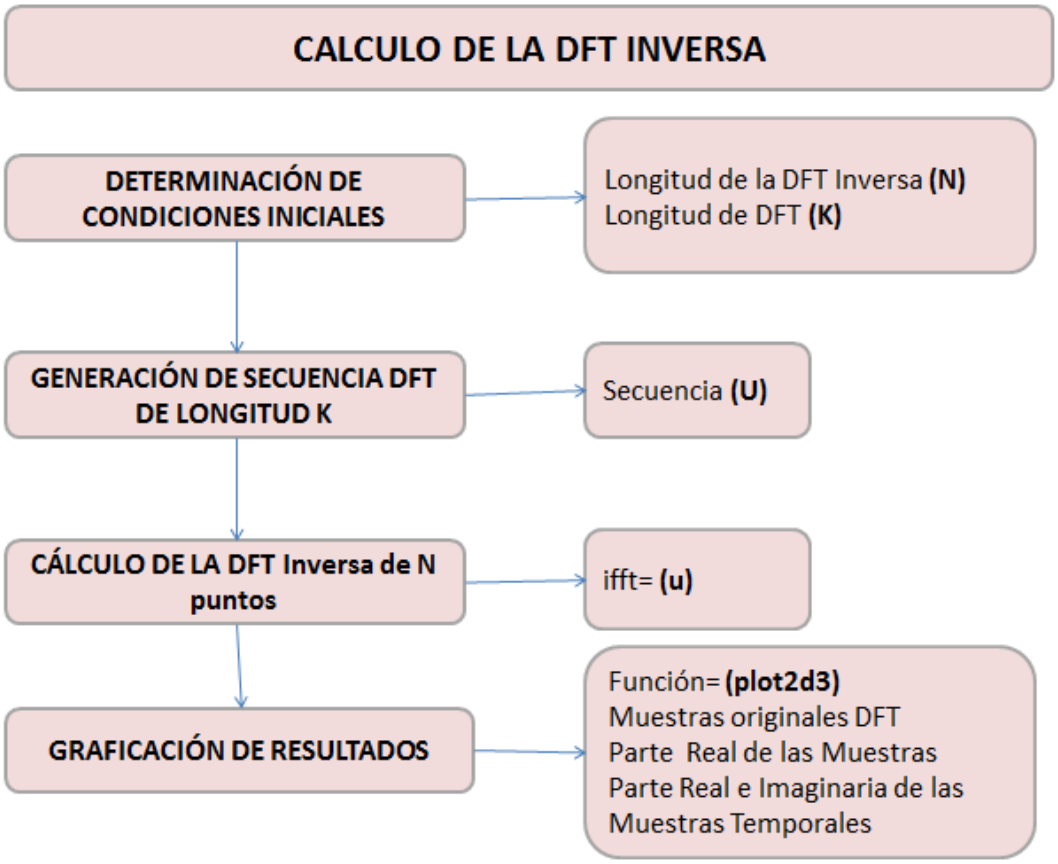


Figura 2.19 Esta secuencia de pasos se utiliza para realizar el cálculo de la DFT inversa, los resultados se los puede observar en **Anexos de PROGRAMA 8** [4].

2.11.9. Programa 9 (Calculo De La DFT Para 512 Puntos)

Se determinara la DTFT de la secuencia x[n] de longitud 16 calculando una DFT de 512 puntos usando el siguiente programa.

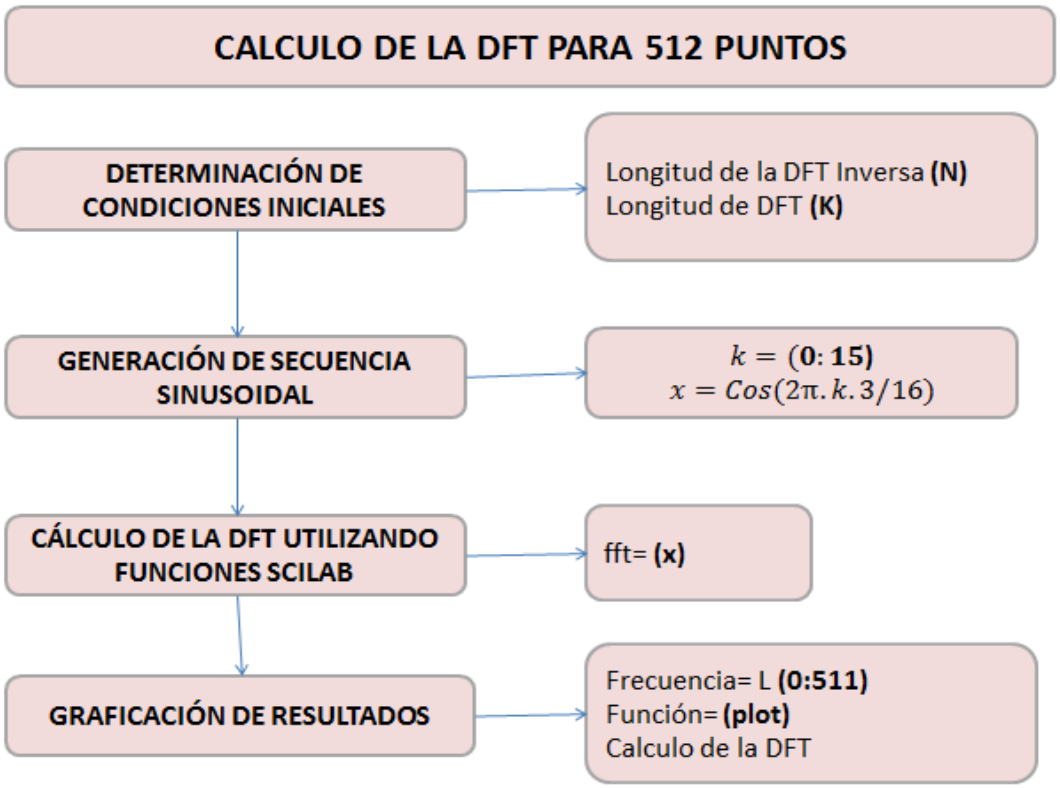


Figura 2.20 Esta secuencia de pasos se utiliza para realizar el cálculo de la DFT de cierto intervalo de puntos, los resultados se los puede observar en **Anexos de PROGRAMA 9** [4].

2.11.10. Programa 10 (Muestreo y Conversión De Datos)

En esta práctica se trata el tema de muestreo y conversión digital analógica, el traslape y la interpolación y decimación.

a. Muestreo y Conversión Digital/Analógica

Se simulará el muestreo de señales de tiempo continuo o analógico para producir señales de tiempo discreto. Puesto que SCILAB solo puede procesar señales de tiempo discreto (vectores), no se puede utilizar realmente señales analógicas sin el uso de dispositivos externos de conversión analógico/digitales (A/D). En su lugar, por consiguiente, se simulará el muestreo analógico mediante el muestreo de las secuencias de tiempo discreto correspondientes, que se conoce como decimación de las secuencias. Recíprocamente, se simulará la conversión digital-analógica (D/A) mediante la interpolación de secuencias de tiempo discreto. La interpolación y la decimación son operaciones útiles, aparte de su uso en la simulación de las conversiones A/D y D/A [4].

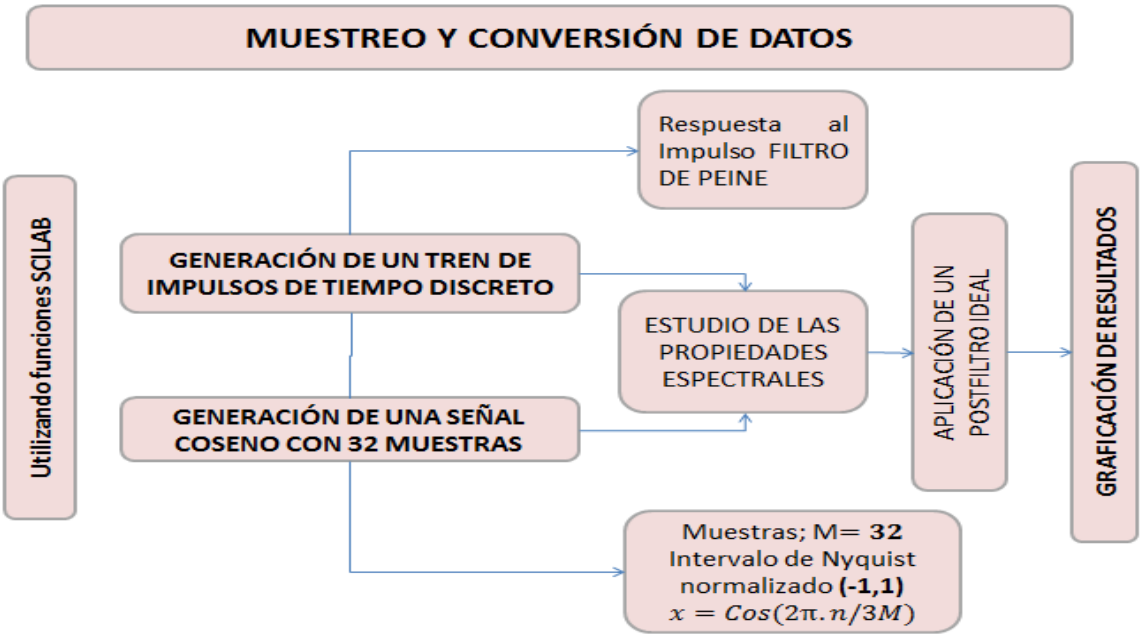


Figura 2.21 El procedimiento y los resultados se los puede observar en **Anexos de PROGRAMA 10-a**.

b. Traslape ("Aliasing")

Para un primer ejemplo de una señal de banda no limitada y el “aliasing” resultante después del muestreo, se utilizará una señal de onda cuadrada. Para sintetizar ocho ciclos de una onda cuadrada de tiempo discreto y para simular su espectro de magnitud DTFT, se realiza:

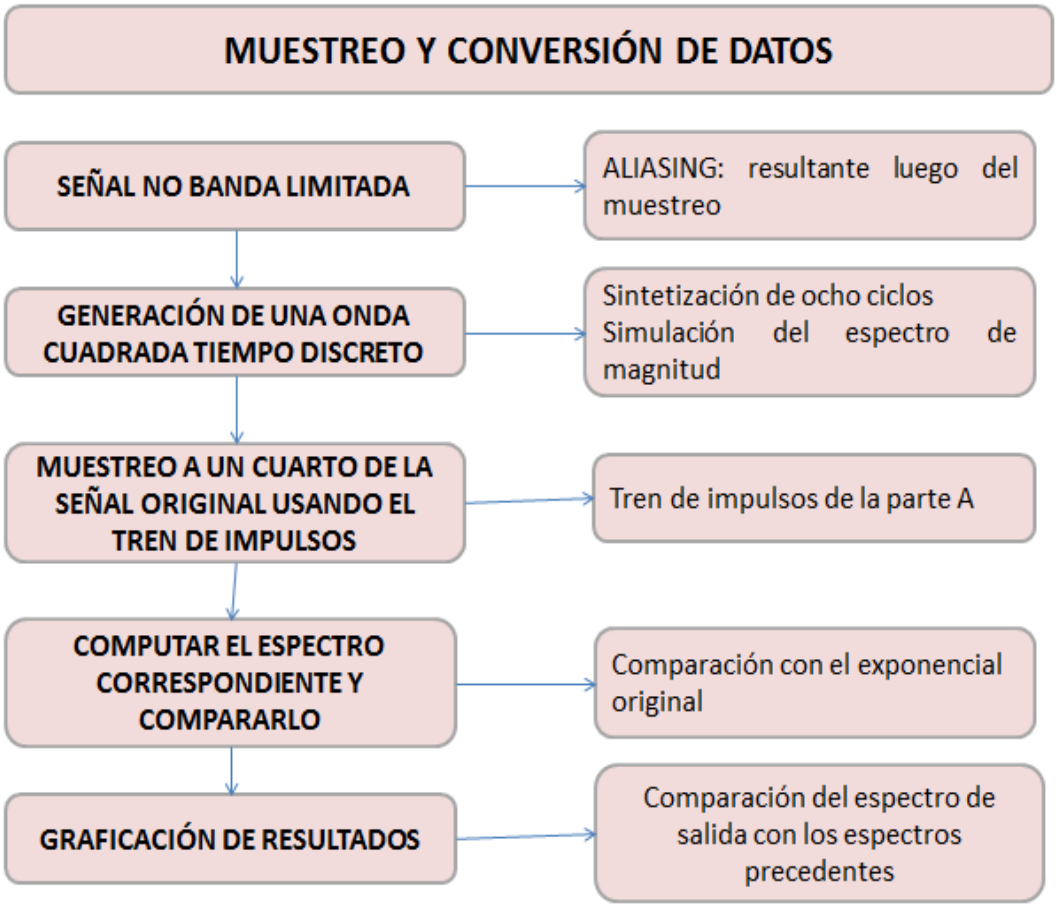


Figura 2.22 El procedimiento y los resultados se los puede observar en **Anexos de PROGRAMA 10-b.**

c. Interpolación y Decimación

Se puede interpolar o decimar secuencias usando los comandos interp y decimate. Por ejemplo, considérese la siguiente interpolación de un senoide por L=4:

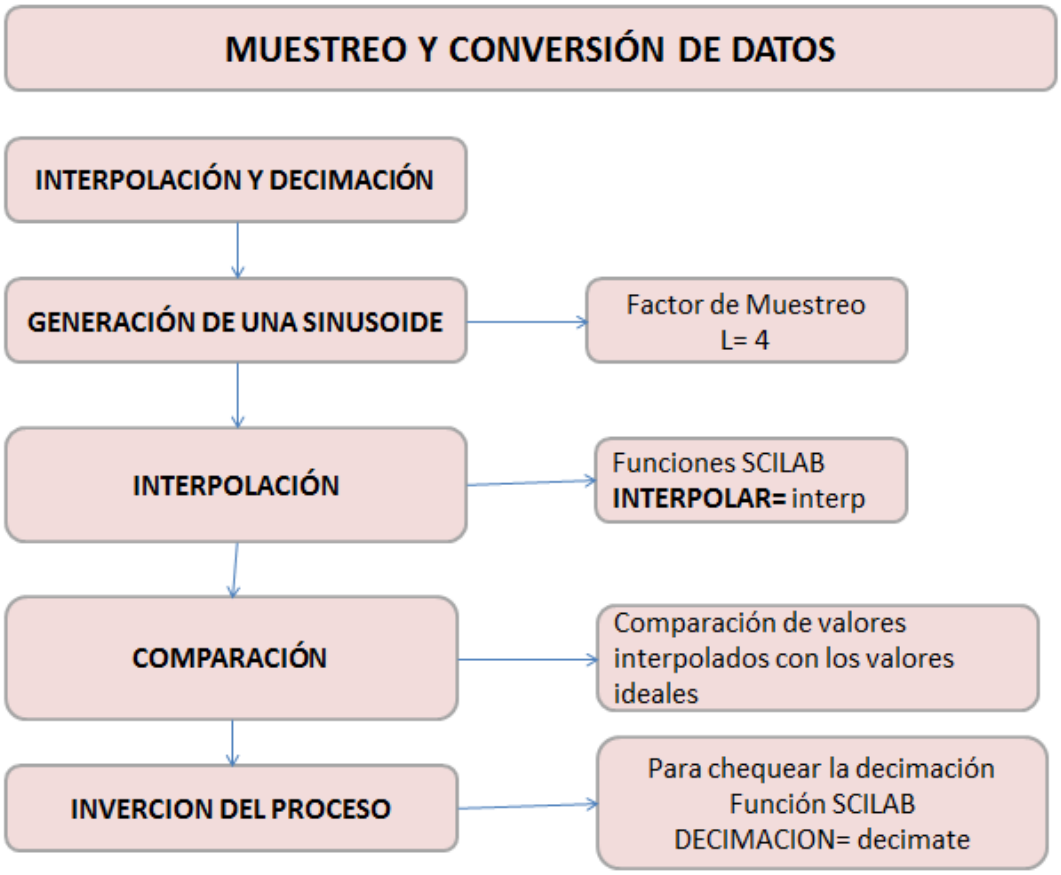


Figura 2.23 El procedimiento y los resultados se los puede observar en **Anexos de PROGRAMA 10-c**.

Nótese que con los desplazamientos de tiempo apropiados, la decimación ejecutada sobre la sinusoide es casi perfecta.

CAPITULO III

3 ANÁLISIS DE SEÑALES CARDIOVASCULARES UTILIZANDO TÉCNICAS DE PROCESAMIENTO DIGITAL DE SEÑALES

3.1 INTRODUCCION

El desarrollo de modelos matemáticos para generar señales sintéticas de electrocardiograma (ECG) es un tema ampliamente investigado, cuyo principal desafío consiste en que las señales de ECG obtenidas a partir de los mismos, posean una gama amplia de morfologías, espectros de potencia y cambios en la variabilidad de la frecuencia cardiaca, según el comportamiento deseado.

La Electrocardiografía (ECG) permite visualizar la actividad eléctrica del corazón, la cual entrega información vital al momento de conocer el estado del músculo cardiaco. Para obtener dicha señal eléctrica es necesario emplear análisis matemático, el cual está compuesto por una serie de algoritmos matemáticos los cuales generan la onda de electrocardiograma.

El objetivo del algoritmo es producir una onda de ECG IDEAL para poder utilizarla para la comparación con diferentes tipos de ECG, permitiendo dar a conocer con su medición la variabilidad de los datos reflejados en sus derivaciones de cada uno de los segmentos como sea posible.

El ahorro de tiempo en cálculos gracias a SCILAB es de gran ayuda para la eliminación de procesos repetitivos en la teoría de cálculos. El simulador ECG nos permite analizar y estudiar formas de onda de ECG normal y anormal sin usar la máquina de ECG.

Este capítulo comienza con una revisión de trabajos similares existentes en esta área, con el fin de encontrar soportes de ayuda para la aplicación a estudiarse, posteriormente se exponen cálculos necesarios para determinar la frecuencia de muestreo de la señal discreta generada para cualquier simulador de ECG. Finalmente se describe que es un electrocardiograma, las ondas que lo conforman, las patologías más conocidas asociadas y la variabilidad de los ECG en presencia de un estándar creado en Scilab.

Se representa un algoritmo para realizar el análisis de la señal electrocardiográfica de forma automática mostrando los diferentes intervalos que ofrece una señal de este tipo.

Se representa cada una de las morfologías de onda típicas de un ECG tales como las ondas P, Q, R, S, T, consiguiendo generar los patrones escogidos.
El conocimiento de la capacidad de software SCILAB es de vital importancia ya que nos ayuda a obtener las componentes de una serie de datos que representa la imagen ideal del musculo cardiaco.

3.2 SISTEMA DE CONDUCCIÓN CARDIACO

El sistema de conducción cardíaca es un grupo de músculos cardíacos especializados ubicados en las paredes del corazón que envían señales al músculo cardíaco que hacen que se contraiga, esta señal crea una corriente eléctrica que puede ser observada en un gráfico llamado electrocardiograma (ECG). Los componentes principales del sistema de conducción cardiaca son:

3.2.1. Nodo Sinusal (Senoauricular SA)

Nodo SA, está situado en la parte superior de la aurícula derecha, ligeramente lateral a la unión de la orejuela, en condiciones normales, este nódulo genera un estímulo eléctrico cada vez que el corazón late, el cual viaja a través de las vías

de conducción y hacen que las cavidades bajas del corazón se contraigan y bombeen la sangre hacia fuera [6].

3.2.2. Nodo Aurículo-Ventricular (AV)

El Nodo AV, localizado en la aurícula derecha en su parte baja y el anillo fibroso central exactamente encima de los ventrículos, aquí llega el impulso eléctrico proveniente del nodo SA, es aquí en el nodo AV donde se retrasan los impulsos durante unos breves instantes para continuar por la vía de conducción a través del Haz de His hacia los ventrículos (marcapaso fisiológico).

3.2.3. Conducción a Nivel Auricular

El modo de conducción de los impulsos a las aurículas ha sido un tema de bastante controversia. A nivel de aurículas no existe un “verdadero” tejido de conducción (a diferencia de los ventrículos que si cuentan con tejido conductivo: el Haz de His y fibras de Purkinje).El impulso se transmite en forma radial y sincitial (la más rápida) desde el nodo SA al nodo AV, de manera que se admite que hay tres áreas de fibras musculares de conducción más rápida.

3.2.4. Haz de His

Pequeña banda de fibras miocárdicas especializadas que conduce la onda de contracción proveniente de las aurículas a los ventrículos. Cruza el triángulo fibroso pasando por la parte posterior inferior del septo membranoso y se dirige en dirección anterior y medial.

Tiene una longitud aproximada de 1cm, antes de dividirse siendo el “Haz no ramificado”.

3.2.5. Ramas y Fibras de Purkinje

La porción ramificada, comienza con las fibras de “cascada”. La rama izquierda se divide en dos ramas principales:

- La rama antero-superior, que se dirige hacia arriba y adelante terminando en el musculo papilar anterior.
- Rama postero-inferior, que se dirige hacia atrás papilar posterior, que se dirige hacia atrás y abajo, terminando en el músculo papilar anterior.

Después de dar las ramas para formar “la rama izquierda” el Haz continua como “la rama derecha”. Tanto la trayectoria “no ramificado”, para final se conectara con el endocardio ventricular [6].

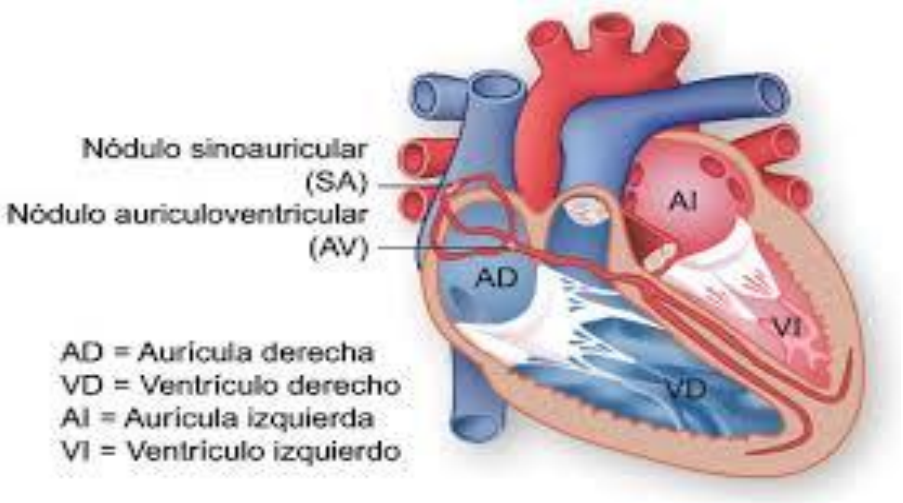


Figura 3.1 Muestra el sistema de conducción cardíaco con las partes que intervienen en la generación del electrocardiograma ECG.

3.3 EL ELECTROCARDIOGRAMA (ECG)

Un ECG es un registro visible de la actividad eléctrica del corazón, es un registro gráfico de los potenciales eléctricos durante el ciclo cardíaco, esta representación consiste en una línea base, varias deflexiones y ondas, que se obtienen colocando electrodos en diversas posiciones del cuerpo, y conectándolos a un aparato electrocardiográfico. De la actividad eléctrica del corazón, se puede obtener una representación gráfica o un trazado de la actividad eléctrica del corazón dando forma a una onda que nos indicarán la transmisión del impulso eléctrico que contrae al musculo cardiaco y por tanto, lo hace realizar su trabajo (expulsar sangre por una parte especifica).

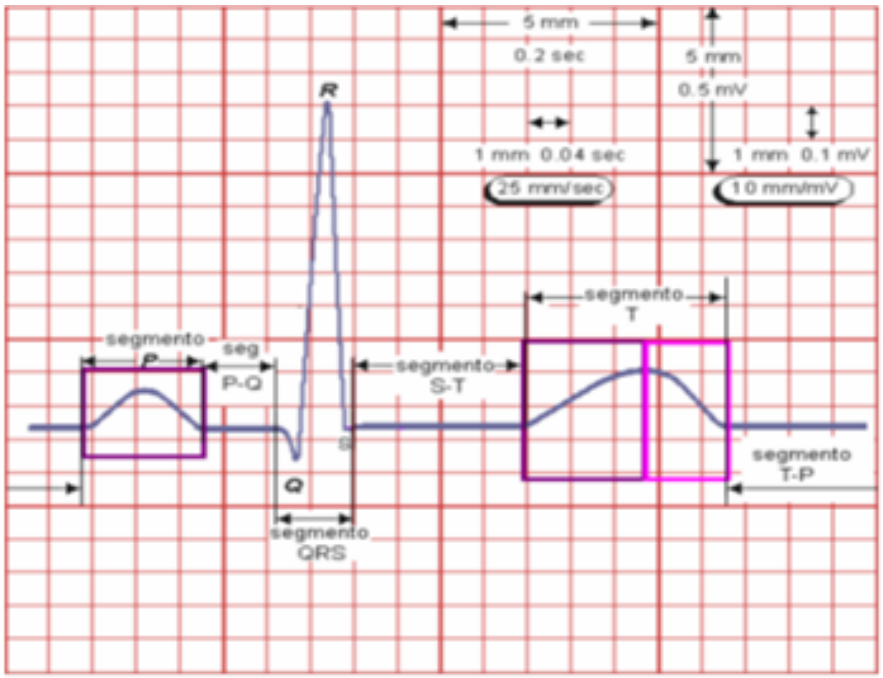


Figura 3.2 Muestra las características significativas de la forma de onda P, Q, R, S, y T, la duración de cada onda, y ciertos intervalos de tiempo tales como la PR, ST, y los intervalos QT [7]-[13].

Lo que se pretende con este análisis es conocer sobre el origen de la actividad eléctrica del corazón, las ondas que este genera y el análisis matemático de la generación de este tipo de ondas.

3.4 EL CORAZON CON RELACION AL ECG

Cada onda representa la transmisión de un impulso eléctrico que contrae al músculo cardíaco y por lo tanto lo hace expulsar sangre por una parte específica del corazón.

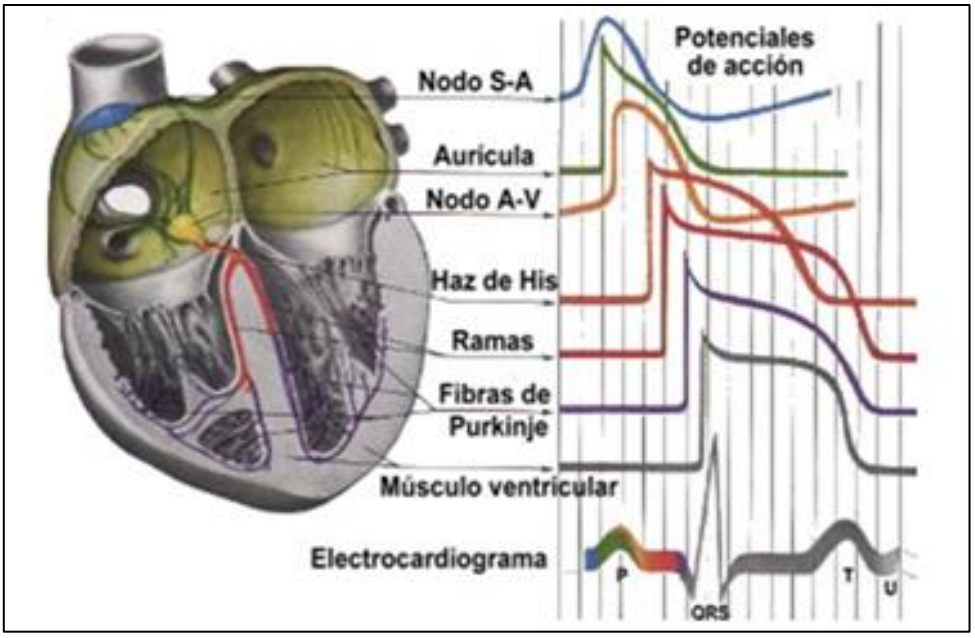


Figura 3.3 Muestra el trazado típico de una señal de electrocardiograma ECG registrando un latido cardíaco normal, formado por una onda P, complejo QRS, onda T **[7].**

3.5 PARTES DE UN ECG

3.5.1. Vía Eléctrica Normal

La onda P (onda auricular) empieza en el nodo SA (marcapaso fisiológico normal), localizado en la parte alta de la aurícula derecha. El complejo QRS (onda ventricular), empieza en el nodo AV, localizado en la parte superior de los ventrículos. Ambos nodos están inervados por el sistema simpático, que aumenta la frecuencia cardiaca, y por el sistema parasimpático (nervio vago) que disminuye la frecuencia cardiaca.

Ya conocidas las medidas básicas, estamos familiarizados con la relación entre las ondas del ECG y la anatomía del corazón, veamos cual es el significado de cada onda e intervalo [6].

a. **Onda P:** Esta onda representa la contracción auricular, su ensanchamiento indica agrandamiento de la aurícula, como puede producirse en la estenosis mitral (la aurícula crece porque la abertura del orificio valvular mitral, entre la aurícula y el ventrículo izquierdo, es pequeña, obligando a la sangre a estancarse y a la pared auricular a expandirse). La onda P suele considerarse aumentada si se tiene una altura mayor de dos y medio pequeños cuadros, una anchura mayor de tres pequeños cuadros o ambas características.
b. **Intervalo PR:** Este se extiende desde el comienzo de la onda P al de la onda Q. Tiene importancia principalmente porque este intervalo aumenta de duración en la cardiopatía arterioesclerosa y en la fiebre reumática. Este alargamiento se produce porque el tejido cardiaco, cuya actividad está representada por el intervalo PR

(aurícula y zona del nodo AV), esta inflamado o es cicatrizal, y el impulso se propaga con menor velocidad. En términos generales, el intervalo PR normal no dura más de 0,20s.

c. **Complejo QRS:** Está formado por tres deflexiones: onda Q, el primer desplazamiento hacia abajo; onda R, en el desplazamiento hacia arriba, y onda S, el ultimo desplazamiento hacia abajo.

Una onda Q grande puede indicar infarto de miocardio antiguo. Una onda R alta suele indicar crecimiento ventricular. La onda S tiene poca significación para la actual exposición. Aunque no siempre se registren complejos QRS con onda Q y con onda S, es costumbre usar la denominación compleja QRS para indicar que es un impulso ventricular.

a. **Segmento ST:** Empieza al final de la onda S y finaliza al principio de la onda T. Se eleva cuando existe infarto de miocardio agudo. Esta hundido cuando: a) El musculo cardiaco no recibe su provisión normal de oxígeno, b) El paciente recibe digital.

b. **Onda T:** Representa la recuperación eléctrica de la contracción ventricular. (Los electrones se desplazan para recuperar sus posición normal, el reposo). La onda T se aplana cuando el corazón no recibe suficiente oxígeno, como en la cardiopatía arterioesclerosa.

Puede ser alta cuando la concentración sérica de potasio es elevada. La onda T normal no excede de 5 cuadrados pequeños (5mm).

3.6 ALGORITMO DETECTOR DE ONDAS

3.6.1. Frecuencia Cardiaca

Como ya se ha indicado cada cuadrado grande en el papel del ECG representa 0.20s. Por tanto 300 cuadros representa un minuto (0,20*300 = 60s). Para determinar en forma rápida pero aproximada la frecuencia cardiaca hay que contar el número de cuadrados grandes entre una y otra onda R (complejo QRS) del ECG y dividir 300 por esta cifra. Por ejemplo, si en una muestra hay tres cuadrados grandes entre dos onda R.

Una señal ideal de ECG es útil para la comparación gráfica con registros de varias personas midiendo las características del patrón de cada una de las señales que origina el corazón.

Se propone, como concepto inicial, que cada patrón de una señal ECG se puede entender como el resultado de la composición de diferentes tipos de ondas (P, Q, R, S, T y U), así como su naturaleza (triangular y sinusoidal) que fácilmente se puede observar en la **figura 3.1** Considera (Q, R, S) como ondas triangulares y (P, T, U) como ondas sinusoidales.

Para graficar este estudio se utiliza las funciones del software SCILAB tomando como base los cálculos desarrollados con Fourier, esta nos brinda una gran cantidad de funciones que conviene utilizar para la simplificación de cálculos **[9].**

Este estudio se basa en tres procedimientos fundamentales

- Detección de QRS: se obtendrá el ciclo de marcadores cardiacos.
- Obtener inicios y finales de P y T: se obtendrán segmentos, duraciones y amplitudes.
- Procesado final de los datos y deducciones.

3.7 SIMULACIÓN DE LA SEÑAL ECG

En esta parte se desarrollará la simulación de la señal ECG a partir de ondas triangulares, medias ondas y segmentos rectos [5].

3.7.1. Cálculos

Se afirma que la máxima frecuencia de un ECG es 150 Hz, al filtrase todas las frecuencias por encima de los 150 Hz. Aplicando el Teorema de muestreo de Nyquist-Shannon , el simulador debe generar una señal ECG discretizada con 300 muestras por segundo, pero por cuestiones técnicas siempre se elegirá una frecuencia de muestreo potencia de dos.

Con lo cual se usa una frecuencia de muestreo de 512 Hz.

$$F_{Max}ECG = 150\ Hz$$
$$F_{Simple} = 2^n \geq F_{Max}ECG,$$
$$F_{Simple} = 2^9 = 512\ Hz$$

Se expresa cada una de los tipos de ondas como serie de Fourier para que una vez generadas cada una de las partes que forma el patrón de un ECG, sólo puede sumar cada onda generada y así obtener la señal del ECG resultante. La señal R del patrón queda centrada, y se usa como punto de referencia para el desplazamiento del resto de ondas que forman el patrón del ECG [7].

Adaptando las fórmulas de la serie de Fourier para generar las señales ECG:

$$F = \frac{hb}{60}$$
$$T = \frac{60}{hb} = 2l \quad \text{Donde; hb} \rightarrow \text{latido por minuto}$$
$$l = \frac{30}{hb}$$

Se obtiene:
$$f(x) = \frac{a_0}{2} + \sum_{n=1}^{\infty} a_n \cos\left(\frac{n\pi}{l}x\right) + \sum_{n=1}^{\infty} b_n \sin\left(\frac{n\pi}{l}x\right) \qquad \textbf{(3.1)}$$

3.7.2. Generación del Complejo QRS

Para poder generar esta forma de onda, hay que encontrar el sumatorio de señales sinusoidales que genera esta forma de onda. Se trata de construir un pulso triangular de T = 2l, de amplitud máxima a y de duración d**.**

Definición analítica de la función:

$$f(x) = -\frac{ab}{L}x + a \qquad 0 < x < \frac{L}{b}$$

$$f(x) = \frac{ab}{L}x + a \qquad -\frac{L}{b} < x$$

Aplicando las series de Fourier tenemos:

$$a_0 = \frac{b}{L}\int_{-\frac{L}{b}}^{\frac{L}{b}} f(x)dx = \frac{ba}{L}\int_{-\frac{L}{b}}^{0}\left(\frac{b}{L}x+1\right)dx + \frac{ba}{L}\int_{0}^{\frac{L}{b}}\left(-\frac{b}{L}x+1\right)dx = a \qquad \textbf{(3.2)}$$

Para a_n se tiene que:

$$a_n = \frac{ba}{L}\int_{-\frac{L}{b}}^{\frac{L}{b}} f(x)\cos\left(\frac{nb\pi x}{L}\right)dx$$

$$a_n = \frac{ba}{L}\left[\int_{-\frac{L}{b}}^{0}\left(\frac{b}{L}x+1\right)\cos\left(\frac{nb\pi x}{L}\right)dx + \int_{0}^{\frac{L}{b}}\left(-\frac{b}{L}x+1\right)\cos\left(\frac{nb\pi x}{L}\right)dx\right] \qquad \textbf{(3.3)}$$

Teniendo presente que la integral de

$$\int x\cos(px)dx = \frac{\cos(px)}{p^2} + \frac{xsen(px)}{p}$$

$$\int_{-\frac{L}{b}}^{0}\left(\frac{b}{L}x+1\right)\cos\left(\frac{nb\pi x}{L}\right)dx = \frac{b}{L}\int_{-\frac{L}{b}}^{0} x\cos\left(\frac{nb\pi x}{L}\right)dx + \int_{-\frac{L}{b}}^{0}\cos\left(\frac{nb\pi x}{L}\right)dx =$$

$$\left[\frac{b\cos\left(\frac{nb\pi x}{L}\right)}{L\left(\frac{nb\pi}{L}\right)^2} + \frac{xsen\left(\frac{nb\pi x}{L}\right)}{n\pi} + \frac{sen\left(\frac{nb\pi x}{L}\right)}{\frac{nb\pi}{L}}\right]_{-\frac{L}{b}}^{0} = \frac{L}{b(n\pi)^2}\left[1-\cos(-n\pi)\right]$$

$$\int_{0}^{\frac{L}{b}}\left(-\frac{b}{L}x+1\right)\cos\left(\frac{nb\pi x}{L}\right)dx = -\frac{b}{L}\int_{0}^{\frac{L}{b}} x\cos\left(\frac{nb\pi x}{L}\right)dx + \int_{0}^{\frac{L}{b}}\cos\left(\frac{nb\pi x}{L}\right)dx =$$

$$\left[\frac{-b\cos\left(\frac{nb\pi x}{L}\right)}{L\left(\frac{nb\pi}{L}\right)^2} - \frac{xsen\left(\frac{nb\pi x}{L}\right)}{n\pi} + \frac{sen\left(\frac{nb\pi x}{L}\right)}{\frac{nb\pi}{L}}\right]_{0}^{\frac{L}{b}} = \frac{L}{b(n\pi)^2}\left[-\cos(n\pi)+1\right]$$

Por tanto

$$a_n = \frac{a}{b(n\pi)^2}\left[2-\cos(-n\pi)-\cos(n\pi)\right]$$

$$a_n = \begin{cases} 0 & \text{n par} \\ \dfrac{2a}{b(n\pi)^2} & \text{n impar} \end{cases}$$

$$b_n = 0$$

$$f(x) = a + \sum_{n=1}^{\infty}\frac{4a}{b(n\pi)^2}\cos\left(\frac{n\pi bx}{L}\right) \qquad \textbf{(3.4)}$$

Para generar las ondas del electrocardiograma se realiza el cálculo mediante:

$$f(x)=\begin{cases}(-\frac{ab}{L}x)+a & \text{en } 0<x<\frac{L}{b}\\ (\frac{ab}{L}x)+a & \text{en } -\frac{L}{b}<x<0\\ \text{o} & \text{en el resto}\end{cases}$$

Calculamos los coeficientes para aplicar **(3.1)**

$$a_0=\frac{1}{L}\int_{-\frac{L}{b}}^{\frac{L}{b}} f(x)$$

$$a_0=\frac{1}{L}\left(\int_{-\frac{L}{b}}^{0}(\frac{ab}{L}x+a)dx+\int_{0}^{\frac{L}{b}}(-\frac{ab}{L}x+a)dx\right)$$

$$a_0=\left(\frac{a}{b}\right)(2-b)$$

$$a_n=\frac{1}{L}\int_{-\frac{L}{b}}^{\frac{L}{b}} f(x)\cos\left(\frac{n\pi}{L}x\right)dx$$

$$a_n=\frac{1}{L}\left(\int_{-\frac{L}{b}}^{0}(\frac{ab}{L}x+a)\cos\left(\frac{n\pi}{L}x\right)dx+\int_{0}^{\frac{L}{b}}(-\frac{ab}{L}x+a)\cos\left(\frac{n\pi}{L}x\right)dx\right)$$

$$a_n=\left(\frac{2ab}{n^2\pi^2}\right)\left(1-\cos\left(\frac{n\pi}{b}\right)\right)$$

$$b_n=\frac{1}{L}\int_{-\frac{L}{b}}^{\frac{L}{b}} f(x)\sin\left(\frac{n\pi}{L}x\right)dx$$

$$b_n=\frac{1}{L}\left(\int_{-\frac{L}{b}}^{0}(\frac{ab}{L}x+a)\sin\left(\frac{n\pi}{L}x\right)dx+\int_{0}^{\frac{L}{b}}(-\frac{ab}{L}x+a)\sin\left(\frac{n\pi}{L}x\right)dx\right)$$

$b_n=0$ (la onda es par)

Por lo que obtenemos la siguiente función:

$$f(x) = \frac{a_0}{2} + \sum_{n=1}^{\infty} a_n \cos\left(\frac{n\pi}{L}x\right)$$

$$f(x) = \frac{a_0}{2}(2-b) + \sum_{n=1}^{\infty} \frac{2ab}{n^2\pi^2}\left(1-\cos\left(\frac{n\pi}{b}\right)\right)\cos\left(\frac{n\pi}{L}x\right) \quad \textbf{(3.5)}$$

Esta función se usa para generar las ondas triangulares de un ECG, es decir simula el complejo QRS de la señal ECG. La modificación de los parámetros a y d permitiendo permite modelar tanto su amplitud como su duración.

El último parámetro a modificar, el desplazamiento respecto el pico R, es simplemente una traslación en el eje del tiempo, producida por sumar el desplazamiento hacia x de la función para avanzar el recorrido o restando el desplazamiento para retrasarlo [7].

3.7.3. Generación de Ondas Sinusoidales P, T y U

Las ondas P, T y U de una señal de ECG se pueden representar como ondas sinusoidales. Hay que modelar las ondas P, T y U como la forma de onda. Para poder generar esta forma de onda, hay que encontrar el sumatorio de señales sinusoidales que genera esta forma de onda.

Se trata de construir un pulso sinusoidal de T = 2L, de amplitud máxima a y de duración d:

$$f(x)=\cos\left(\frac{b\pi}{2L}x\right) \qquad \text{Entre} \qquad -\frac{L}{b}<x<\frac{L}{b}$$

Pero hay que tener en cuenta, que al usar una función sinusoidal, su amplitud máxima por definición será 1. Por lo tanto, hay que escalar la onda multiplicando por la amplitud indicada. Este último paso, se deja para cuando se tenga la función expresada como una serie de Fourier. Aplicando las series de Fourier a una señal de este tipo, se obtiene:

Desarrollo de los coeficientes de Fourier:

$$a_0=\frac{1}{L}\int_T \mathrm{f}(\mathrm{x})\mathrm{dx}$$

$$a_0=\frac{1}{L}\int_{-\frac{L}{b}}^{\frac{L}{b}}\cos\left(\frac{b\pi}{2L}x\right)dx$$

$$a_0=\frac{2}{L}$$

Los dos coeficientes restantes quedan expresados de la siguiente forma:

$$a_n = \frac{1}{L}\int_{\mathrm{T}} \mathrm{f}(\mathrm{x})\cos\left(\frac{n\pi}{L}x\right)\mathrm{dx}$$

$$a_n = \frac{1}{L}\int_{-\frac{L}{b}}^{\frac{L}{b}} \cos\left(\frac{b\pi}{2L}x\right)\cos\left(\frac{n\pi}{L}x\right)dx$$

$$a_n = \frac{2}{\pi}\left(\frac{sen\left(\frac{\pi}{2b}(b-2n)\right)}{b-2n} + \frac{sen\left(\frac{\pi}{2b}(b+2n)\right)}{b+2n}\right)$$

$$b_n = \frac{1}{L}\int_{\mathrm{T}} \mathrm{f}(\mathrm{x})\mathrm{sen}\left(\frac{\mathrm{n}\pi}{\mathrm{L}}x\right)\mathrm{dx}$$

$$b_n = \frac{1}{L}\int_{-\frac{L}{b}}^{\frac{L}{b}} \cos\left(\frac{b\pi}{2L}x\right)\mathrm{sen}\left(\frac{\mathrm{n}\pi}{\mathrm{L}}x\right)dx$$

$$b_n = 0\,(\text{porque la forma de onda es par})$$

Al aplicar **(3.1)** se obtiene:

$$f(x) = \frac{a_0}{2} + \sum_{n=1}^{\infty} a_n \cos\left(\frac{n\pi}{L}x\right)$$

$$f(x) = \frac{1}{L} + \sum_{n=1}^{\infty}\frac{2}{\pi}\left(\frac{sen\left(\frac{\pi}{2b}(b-2n)\right)}{b-2n} + \frac{sen\left(\frac{\pi}{2b}(b+2n)\right)}{b+2n}\right)\cos\left(\frac{n\pi}{L}x\right)$$

En este punto es cuando se efectúa el cambio de la amplitud máxima.

Al multiplicar la serie de Fourier por la amplitud indicada, finalmente se obtiene:

$$f(x)=a\left[\frac{1}{L}+\sum_{n=1}^{\infty}\frac{2}{\pi}\left(\frac{sen\left(\frac{\pi}{2b}(b-2n)\right)}{b-2n}+\frac{sen\left(\frac{\pi}{2b}(b+2n)\right)}{b+2n}\right)\cos\left(\frac{n\pi}{L}x\right)\right] \quad \textbf{(3.6)}$$

La fórmula se usa para generar las ondas sinusoidales de un ECG, como lo son las ondas P, T y U.

La modificación de los parámetros a y d permite modelar tanto su amplitud como su duración. El último parámetro a modificar, el desplazamiento respecto el pico R, es simplemente una traslación en el eje del tiempo producida por sumar el desplazamiento a la x de la función para avanzar el recorrido, o restando el desplazamiento para retrasarlo.

3.8. IMPLEMENTACION EN SCILAB

Para cada onda se definen tres valores básicos que servirán para generar cada una de las ondas de la señal ECG hasta completarla.

• Amplitud

• Duración

• Retardo respecto el pico R

3.8.1. Código De Datos

```
      li=30/72;
a= Amplitud
d= Duracion
t= Tiempo
```

PARAMETROS GENERALES ELECTROCARDIOGRAMA							
GENERACION	**Onda P**	**Onda Q**	**Complejo QRS**	**Onda S**	**Onda T**	**Onda U**	**Unidades**
AMPLITUD	0,25	0,025	1,6	0,25	0,35	0,035	MiliVolito(mV)
DURACION	0,09	0,066	0,11	0,066	0,142	0,0476	Segundos(s)
TIEMPO	0,16	0,166		0,09	0,2	0,433	Segundos(s)

Tabla 3.1. Esta tabla muestra los datos de ondas, intervalos y segmentos del ECG [7]-[8].

Para empezar con la simulación del ECG aplicamos las series (3.5) y (3.6) calculadas con anterioridad en la consola de SCILAB.

3.8.2. Onda Sinusoidal P

Para generar la onda P ingresamos el código:

```
x=0.01:0.01:2;
l=1;
a=0.25
x=x+(1/1.8);
b=3;
n=100;
p1=1/l
p2=0
for i = 1:n
   harm1=(((sin((%pi/(2*b))*(b-(2*i))))/(b-
(2*i))+(sin((%pi/(2*b))*(b+(2*i))))/(b+(2*i)))*(2/%pi))*cos((i*%pi*x)/l);
   p2=p2+harm1;
end
onda1p=p1+p2;
ondap=a*onda1p;
```

```
plot(x, ondap)
xgrid(21)
xtitle("ONDA SINUSOIDAL P")
```

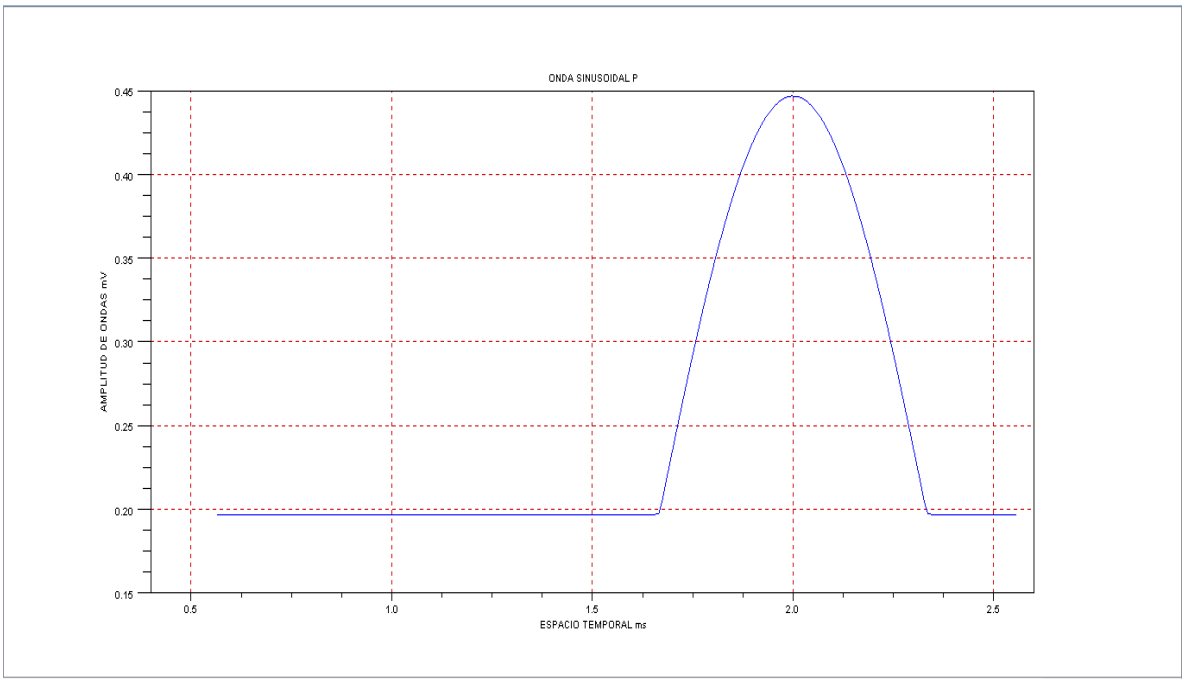


Figura 3.4. Representación de Onda sinusoidal P utilizando software libre SCILAB

3.8.3. Onda Sinusoidal Q

Para generar la onda Q ingresamos el código:

```
x=0.01:0.01:2;
l=1;
x=x+l/6
a=0.025;
b=15;
n=100;
q1=(a/(2*b))*(2-b);
q2=0
for i = 1:n
   harm5=(((2*b*a)/(i*i*%pi*%pi))*(1-cos((i*%pi)/b)))*cos((i*%pi*x)/l);
   q2=q2+harm5;
end
```

```
ondaq=-1*(q1+q2);

plot(x, ondaq)
xgrid(21)
xtitle("ONDA SINUSOIDAL Q")
```

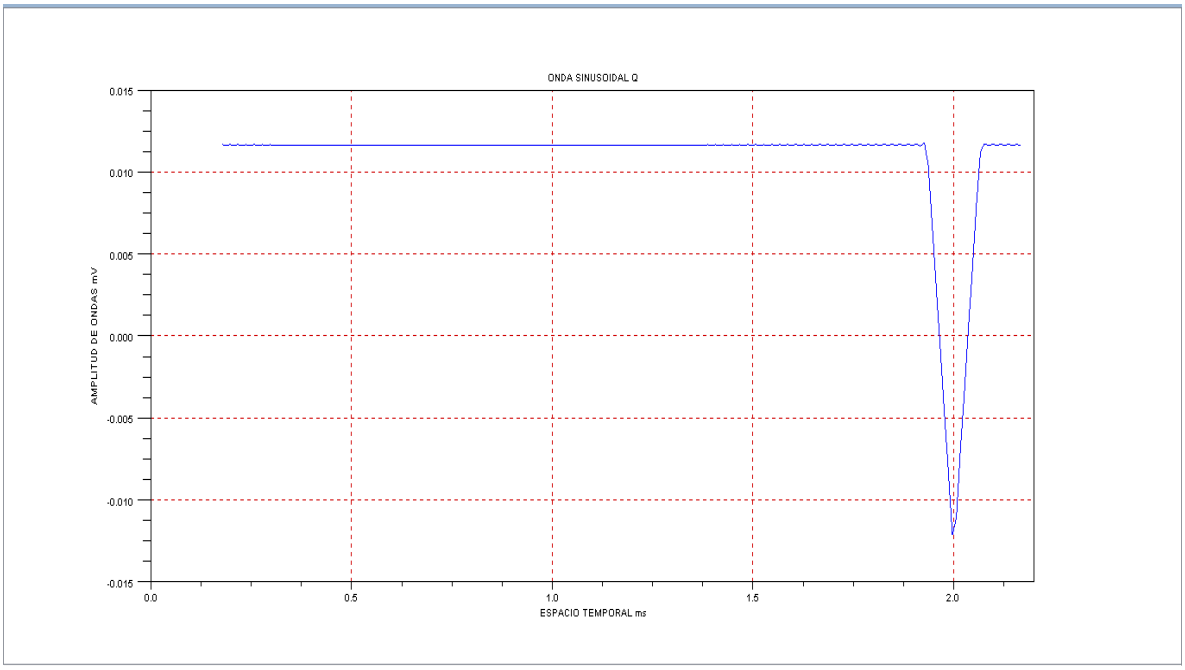


Figura 3.5. Representación de Onda sinusoidal Q utilizando software libre SCILAB

3.8.4. Complejo QRS

Para generar el complejo QRS ingresamos el código:

```
x=0.01:0.01:2;
l=1;
a=1.5;
b=5;
n=100;
qrs1=(a/(2*b))*(2-b);
qrs2=0
for i = 1:n
   harm=(((2*b*a)/(i*i*%pi*%pi))*(1-cos((i*%pi)/b)))*cos((i*%pi*x)/l);
   qrs2=qrs2+harm;
end
```

```
ondaqrs=qrs1+qrs2;
plot(x, ondaqrs)
xgrid(21)
xtitle("COMPLEJO QRS")
```

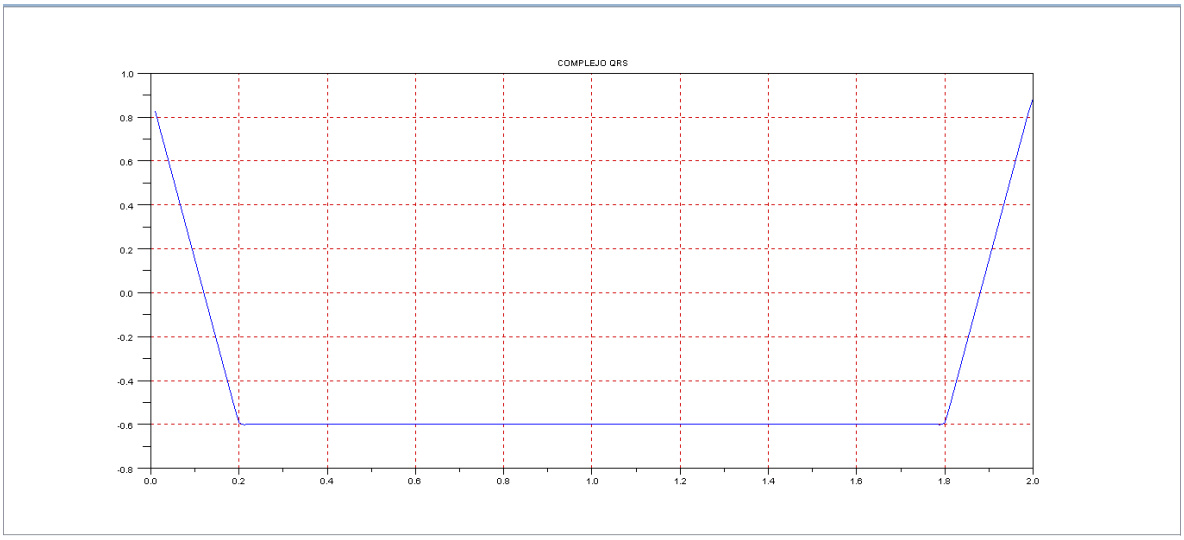


Figura 3.6. Representación del complejo QRS utilizando software libre SCILAB

3.8.5. Onda Sinusoidal S

Para generar la onda S ingresamos el código:

```
x=0.01:0.01:2;
l=1;
x=x-l/6
a=0.25;
b=15;
n=100;
s1=(a/(2*b))*(2-b);
s2=0
for i = 1:n
   harm3=(((2*b*a)/(i*i*%pi*%pi))*(1-cos((i*%pi)/b)))*cos((i*%pi*x)/l);
   s2=s2+harm3;
end
```

```
ondas=-1*(s1+s2);
plot(x, ondas)
xgrid(21)
xtitle("ONDA SINUSOIDAL S")
```

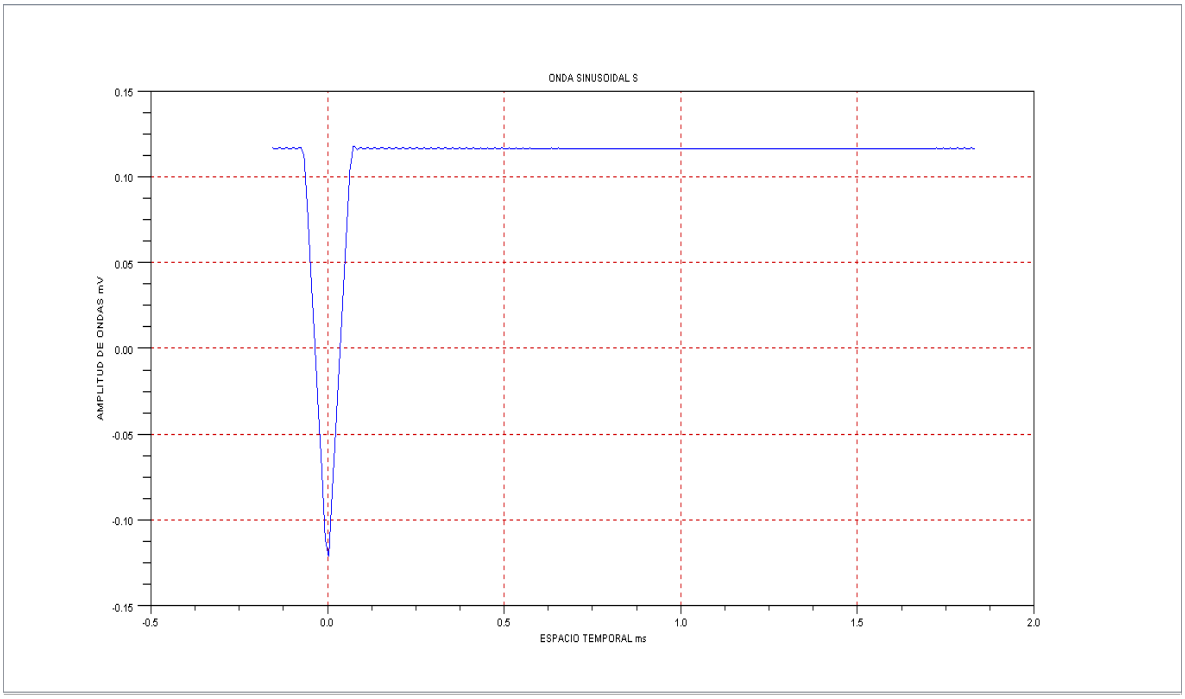


Figura 3.7. Representación de Onda sinusoidal S utilizando software libre SCILAB

3.8.6. Onda Sinusoidal T

Para generar la onda T ingresamos el código:

```
x=0.01:0.01:2;
l=1;
a=0.35
x=x-(1/1.8);
b=7;
n=20;
t1=1/l
t2=0
for i = 1:n
```

```
    harm2=(((sin((%pi/(2*b))*(b-(2*i))))/(b-
(2*i))+(sin((%pi/(2*b))*(b+(2*i))))/(b+(2*i)))*(2/%pi))*cos((i*%pi*x)/l);
    t2=t2+harm2;
end
ondat1=t1+t2;
ondat=a*ondat1;
plot(x, ondat)
xgrid(21)
xtitle("ONDA SINUSOIDAL T")
```

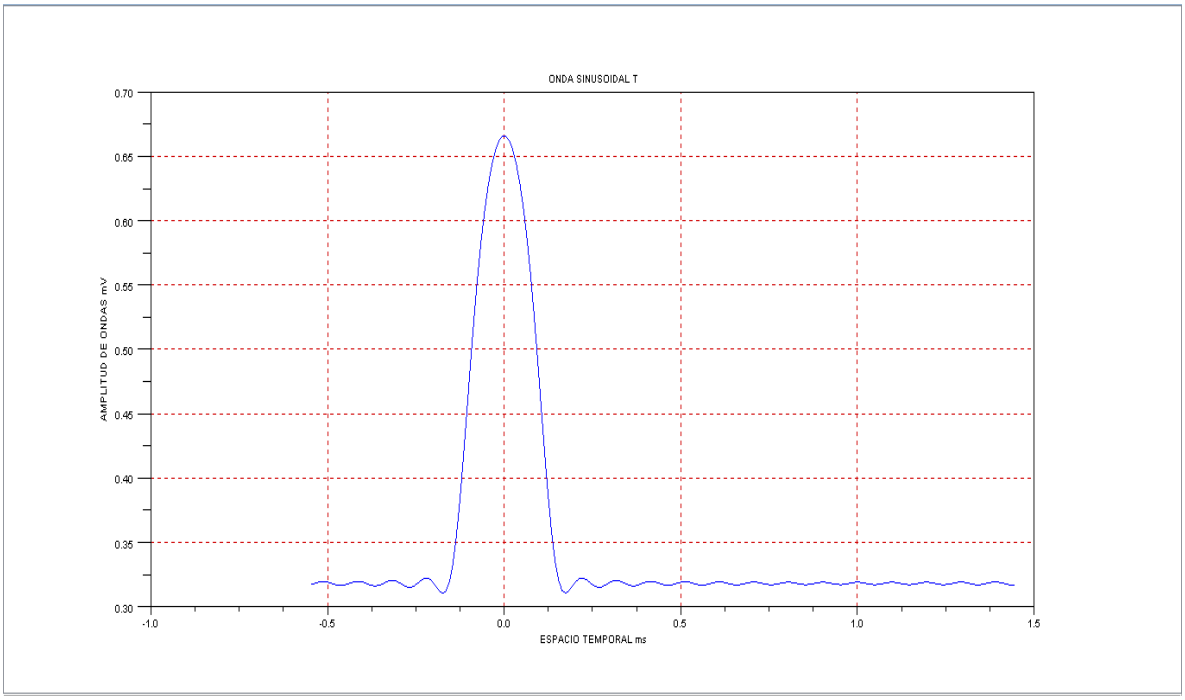


Figura 3.8. Representación de Onda sinusoidal T utilizando software libre SCILAB

3.8.7. Onda Sinusoidal U

Para generar la onda U ingresamos el código:

```
x=0.01:0.01:2;
l=1;
a=0.03;
x=x-(1/1.1);
b=21;
n=100;
u1=1/l;
```

```
u2=0;
for i = 1:n
   harm4=(((sin((%pi/(2*b))*(b-(2*i))))/(b-
(2*i))+(sin((%pi/(2*b))*(b+(2*i))))/(b+(2*i)))*(2/%pi))*cos((i*%pi*x)/l);
   u2=u2+harm4;
end
ondau1=u1+u2;
ondau=a*ondau1;
plot(x, ondau)
xgrid(21)
xtitle("ONDA SINUSOIDAL U")
```

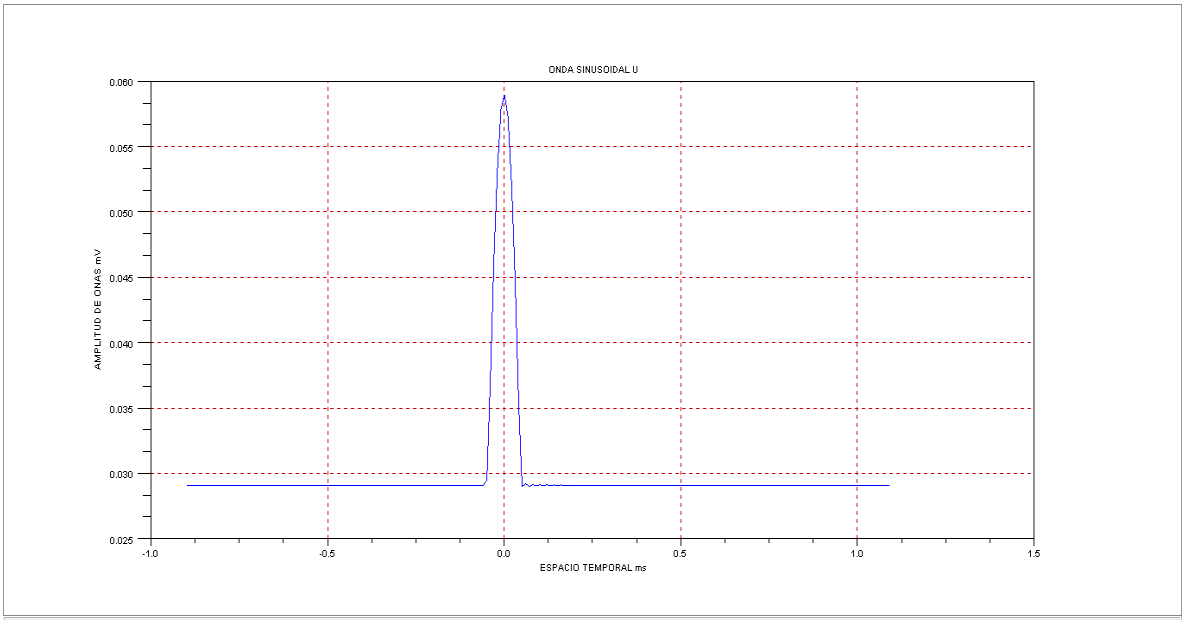


Figura 3.9. Representación de Onda sinusoidal U utilizando software libre SCILAB

Para poder generar todas las características que modela un ECG, se procesa una estructura de datos que describen la morfología completa de una señal de ECG. Esto quiere decir que detalla la forma de cada onda del ECG (P, Q, R, S, T y U).

3.8.8. Implementación en SCILAB

Para generar el electrocardiograma ECG ingresamos el código:

```
plot(ondap+ondaq+ondaqrs+ondas+ondat+ondau+ondap)
xgrid(21)
xtitle("ONDA SINUSOIDAL U")
```

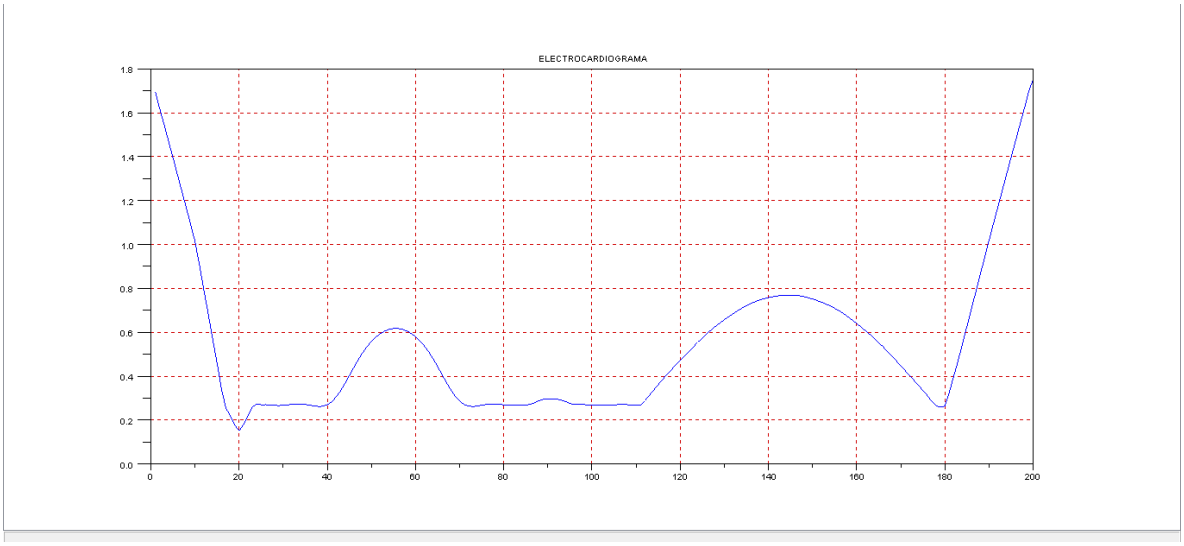


Figura 3.10. Representación gráfica de las líneas de un ECG utilizando software libre SCILAB

3.9 ECG NORMAL Y ANATOMÍA CARDÍACA

El electrocardiograma como ya se había mencionado es un registro relativo de la actividad eléctrica del corazón, es el procedimiento más sencillo y rápido para evaluar dicha actividad, el ECG está compuesto por varias ondas e “intervalos” que representan el comportamiento del corazón de las cuales se hablara en los próximos numerales.

3.9.1 ECG Normal

Todos los latidos cardíacos aparecen con morfologías similares, separados por espacios iguales; cada uno está formado por tres unidades principales: Onda P, complejo QRS y onda T. Aunque para muchas aplicaciones se hace necesario estudiar el complejo QRS como ondas separadas tomando segmentos y analizando sus características.

Durante la despolarización y repolarización miocárdica, aparecen las ondas del *ECG*. Las distancias entre deflexiones u ondas se denominan segmentos o intervalos. Un periodo del *ECG* perteneciente a un individuo sano, consiste en una onda P, el complejo *QRS*, la onda *T* y la onda *U* [10]. Los resultados anormales de ECG pueden indicar lo siguientes tipos de patologías.

3.9.2 Ruta de estímulo en los ritmos sinusoidales

Las tres arritmias que se originan en el nodo SA son: La arritmia sinusal, la taquicardia sinusal y la bradicardia sinusal. La vía que siguen sus impulsos eléctricos es exactamente la de un ritmo sinusal normal (ECG normal), según se indica. En consecuencia, la onda P (auricular) y el complejo QRS (ventricular) tienen la misma configuración que en el ritmo normal.

Ahora que ya conocemos algo de la electrocardiografía básica vamos a considerar un parámetro que se ha convertido en objeto reciente de investigación, es la onda R. Los cambios en la amplitud de dicha onda en el ejercicio del ECG han sido descritos durante la isquemia y el infarto agudo de miocardio. Sin embargo, no se ha estudiado ampliamente en un escenario controlado.

Algunos autores han supuesto que un incremento significante en esta amplitud, ocurre durante la isquemia miocárdica transmural temprana [11]

3.9.3 Variación de complejo QRS

Para generar la variación del complejo QRS, aumentamos su amplitud a un valor arbitrario (Amplitud=2) y observamos su variación.

Ingresamos el código:

```
x=0.01:0.01:2;
l=1;
a=2;
b=5;
n=100;
qrs1=(a/(2*b))*(2-b);
qrs2=0
for i = 1:n
   harm=(((2*b*a)/(i*i*%pi*%pi))*(1-cos((i*%pi)/b)))*cos((i*%pi*x)/l);
   qrs2=qrs2+harm;
end
ondaqrs=qrs1+qrs2;
```

```
plot(x, ondaqrs)
xgrid(24)
xtitle("VARIACION  DEL COMPLEJO QRS")
```

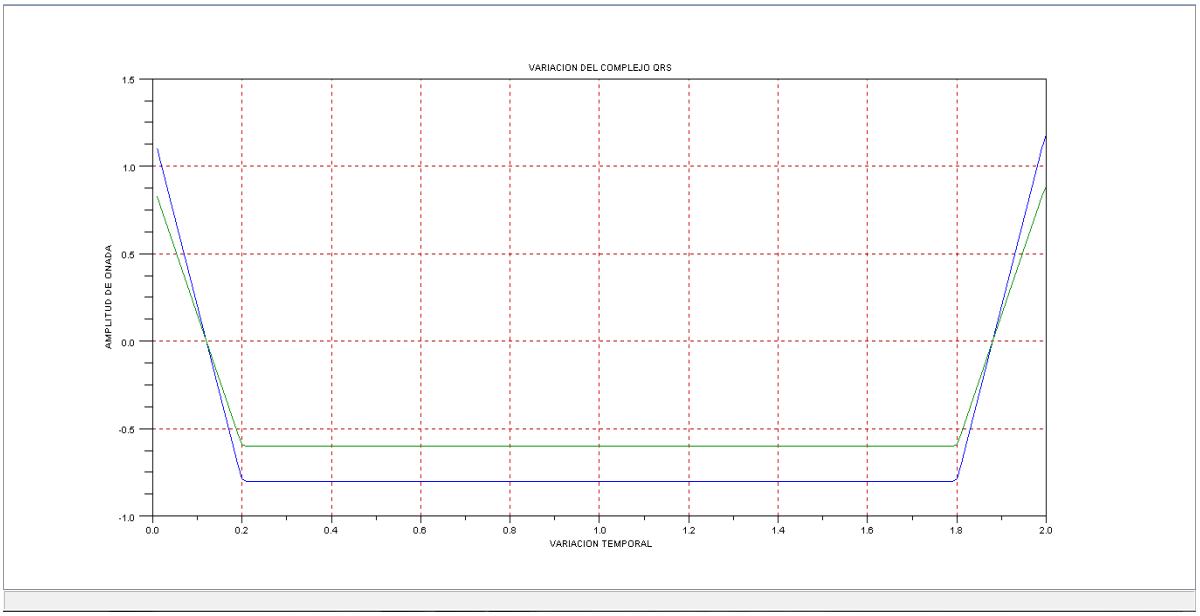


Figura 3.11. Representación gráfica de la variación de amplitud del complejo QRS. La línea verde muestra el complejo original del ECG y la línea azul muestra el complejo QRS alterado.

Por lo que, el nuevo ECG que dibujado de la siguiente manera.

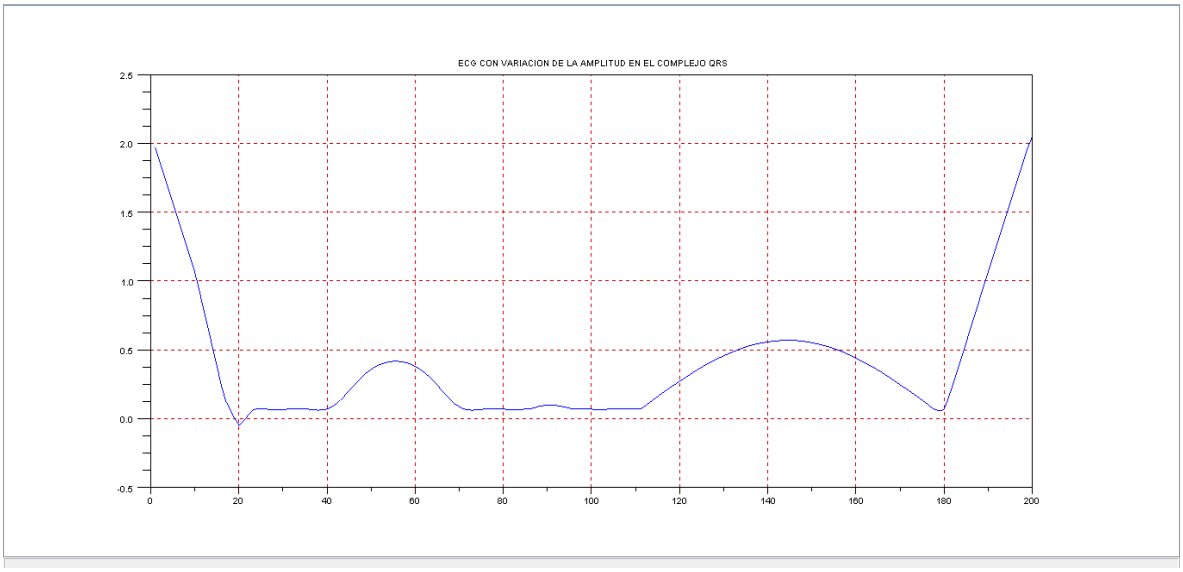


Figura 3.12. Representación gráfica de la variación de amplitud en el complejo QRS del ECG

Aparentemente no se distingue variación al ECG original, pero con una sobre posición de las figuras **(3.10)** y **(3.12)** se observa lo contrario.

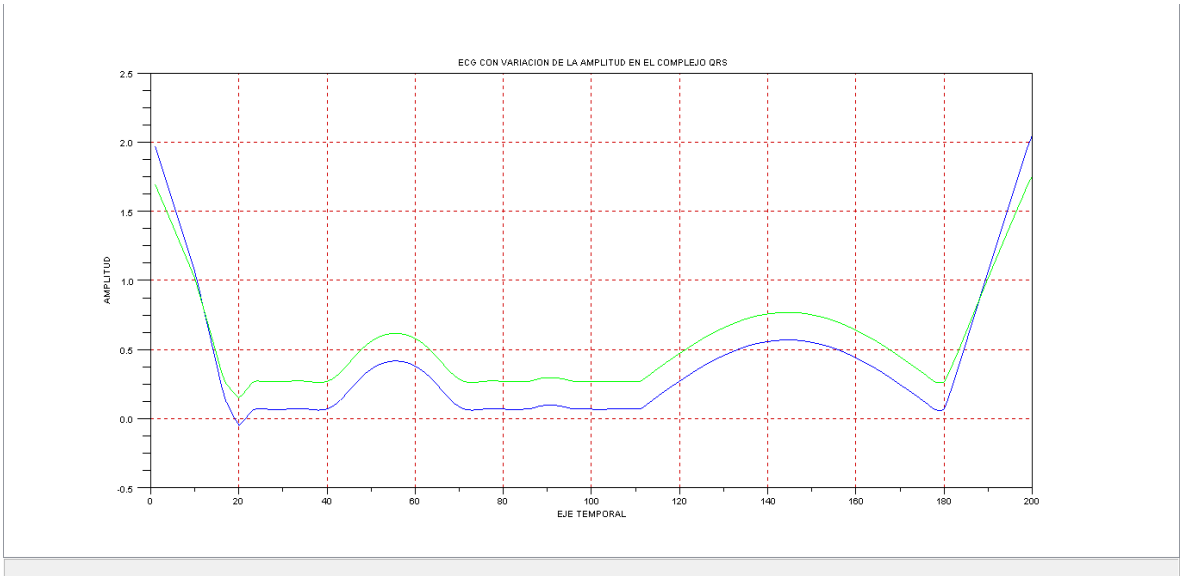


Figura 3.13. Representación gráfica de la variación de un ECG basal y un ECG con patología. La línea verde muestra un trazado de ECG ideal, mientras que la línea de color azul muestra un ECG con isquemia.

Por lo que, el incremento podría ser consistente con la expansión de la cavidad ventricular izquierda durante la isquemia y/o alteraciones en la conducción los cuales son intrínsecos al miocardio. Dicho fenómeno debería ser investigado empleando este tipo de estudios con el fin de mejorar la predicción de esta patología.

CAPITULO IV

4 CONCLUSIONES Y RECOMENDACIONES

4.1 CONCLUSIONES

- La Transformada de Fourier permitió el reconocimiento y desarrollo del Análisis Espectral, esto se aplicó en casos de tratamiento de señales apoyándose con el uso del Software gratuito SCILAB.

- Se logro aplicaciones con la Transformada de Fourier y el software Libre SCILAB, determinando el algoritmo matemático que contribuye a verificar el

estudio de análisis espectral y profundizar en el análisis de señales cardiovasculares.

- El estudio del ECG utilizando las reacciones generadas por los biopotenciales ayuda a poner en práctica las herramientas de Análisis Espectral y SCILAB.

- La utilización de Software ayudo al desarrollo de elementos que mejoran el diagnóstico médico, visualizando con mayor detalle las patologías del músculo cardíaco.

4.2 RECOMENDACIONES

- Este estudio puede ser utilizado para un curso ingenierístico introductorio de procesamiento de señales y a la vez de aprendizaje del Software SCILAB poniendo en práctica en el desarrollo de una gran infinidad de procesos.

- Se recomienda mayor enfoque la teoría que se utilizara en la aplicación del trabajo, esto debido a que el estudio de análisis espectral abarca un sin número de temas y aplicaciones que por lo general no es necesario adentrarse.

- Se recomienda al momento de desarrollar un algoritmo matemático, se utilice una metodología ordenada, esto con el fin de evitar cálculos innecesarios e irse centrando de buena manera en los resultados a obtenerse.

- Se recomienda profundizar al mejoramiento de los parámetros de las ondas del ECG tomando como base lo desarrollado.

BIBLIOGRAFIA

[1].-PAG, TAO., Spectral Analysis., *An Introduction to Computational Physics.*, 2a. ed., Universisty of Nevada, Las vegas., 2006., Pp. 164-196.

[2].-DELICADO, BERNARDO., Tratamiento de señales con Scilab., *Manual de introduccion al tratamiento de señales con Scilab para usuarios de Matlab.*, Madrid-España., 2012., Pp 21-44.

[3].-ALVARADO, JOSE., Procesamiento Digital de Señales., *Causalidad y sus implicaciones.*, Sistemas LTI como filtros selectivos en frecuencia., Cartago-Costa Rica., 2006., Pp 209-228.

[4].-PEREZ, ARTHUR., Diseño de practicas de Procesamiento digital de Señales utilizando SCILAB y SCICOS., Introduccion a SCILAB., Rodrigo Facio-Costa Rica., 2009., Pp 5-124.

[5].- AGUILAR, VALENTI., Método de Musicalización de ECG., Simulación de Electrocardiograma (ECG)., Bellaterra-España., 2007., Pp 33-64.

[6].-ANÁLISIS DE SEÑAL DEL IMPULSO CARDÍACO PARA EL MEJORAMIENTO DEL DIAGNÓSTICO DE PATOLOGÍAS DEL CORAZÓN

http://repositorio.utp.edu.co/dspace/bitstream/11059/1873/1/6161207547P661.pdf

2013-01-05

[7].-SISTEMA SIMULADOR ECG PARA EL ESTUDIO DE SEÑALES CARDIACAS

http://www.iiis.org/CDs2012/CD2012SCI/CISCI_2012/PapersPdf/CA769GN.pdf

2012-12-27

[8].-TRATAMIENTO DE DATOS EN LAS TÉCNICAS INSTRUMENTALES

http://www.culturacientifica.org/textosudc/unidad_didactica_fft.pdf

2013-03-15

[9].-TUTORIA INTRODUCTORIO A LA TEORIA WAVELET

http://www2.elo.utfsm.cl/~elo377/documentos/Wavelet.pdf

2012-06-01

[10].-INTRODUCCION AL ANALISIS ESPECTRAL

http://www.astormastering.com.ar/Clase1_Introducci%C3%B3n_al_analisis_espectral.pdf

2012-03-19

[11].-MANUAL DE INTRODUCCION AL TRATAMIENTO DE SEÑALES CON SCILAB PARA USUARIOS DE MATLAB

http://www.virtual.unal.edu.co/cursos/ingenieria/2001619/lecciones/descargas/senal.pdf

2011-11-29

[12].- INTRODUCCION A SCILAB

http://www.ing.una.py/DIREC_PPAL/ACADEMICO/APOYOcalculo%20numerico/Scilab/CURSO%20DE%20SCILAB.pdf

2013-04-31

[13].-FILTRADO DE SEÑAL DE ELECTROCARDIOGRAMA

http://ebookbrowse.com/2012-practica-2-filtrado-en-ma_tlabpdf-d347705306

2013-02-08

[14].-ANÁLISIS Y TRATAMIENTO DE LA SEÑAL ELECTROCARDIOGRÁFICA PARA LA DETECCIÓN DE PARÁMETROS DE NORMALIDAD BAJO LA PLATAFORMA LABVIEW "ADPAN-ECG" (FCV - UPB - COLCIENCIAS)

http://especiales.universia.net.co/dmdocuments/Tesis_Miguel_Ingenieria_Electronica_UPB.pdf

2012-12-06

[15].-APRENDA MATLAB 7.0

http://mat21.etsii.upm.es/ayudainf/aprendainf/Matlab70/matlab/70primero.pdf

2012-03-13

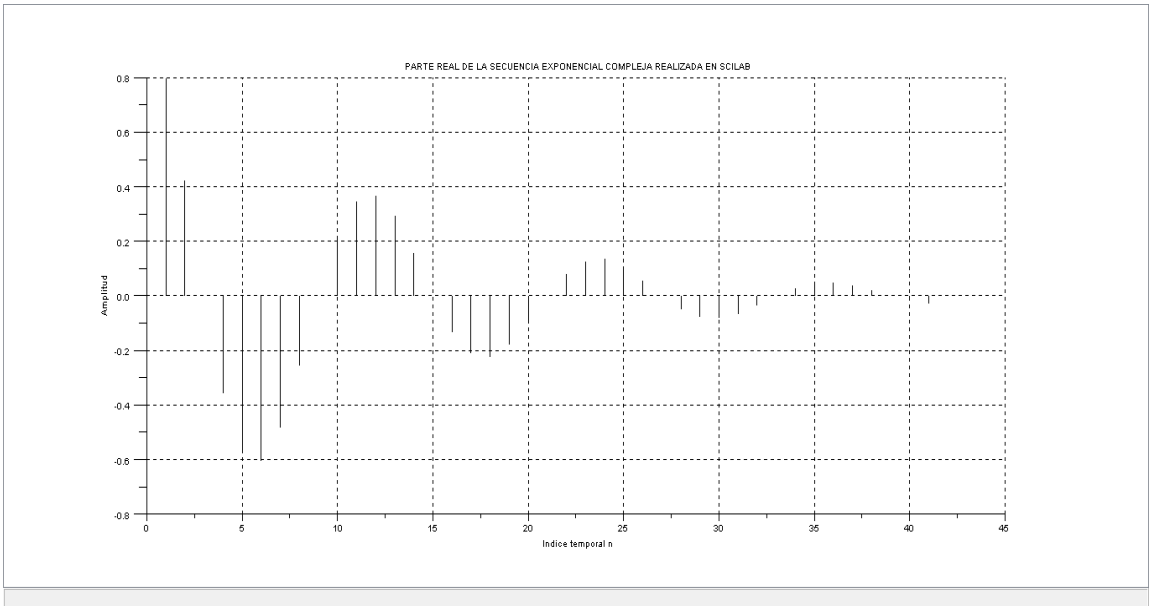
ANEXOS

DESARROLLO DE PROGRAMAS EN EL SOFTWARE SCILAB

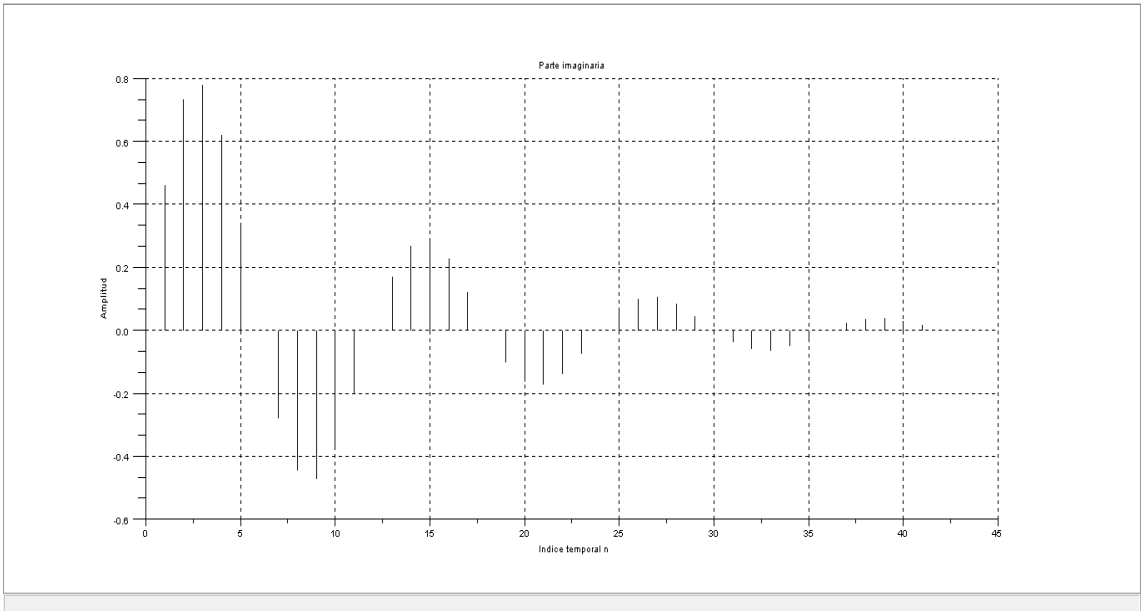
- **PROGRAMA 1**

```
// Generación de una secuencia exponencial compleja

// a = -1/12; b = %pi/6; K = 1;N = 41;

a = input('Introduzca exponente real = ');
b = input('Introduzca exponente imaginario = ');
c = a + b*%i;
K = input('Introduzca la constante de ganancia = ');
N = input('Introduzca longitud de secuencia = ');
n = 1:N;
x = K*exp(c*n);
f0=scf(0); //crea la figura 0
plot2d3('gnn',n,real(x));xgrid;
xlabel('Indice temporal n');ylabel('Amplitud');
title('Parte real');
disp('Teclee ENTER para la parte imaginaria');
pause
f1=scf(1);
scf(f1);
```

```
plot2d3('gnn',n,imag(x));xgrid;
xlabel('Indice temporal n');ylabel('Amplitud');
title('Parte imaginaria');
```



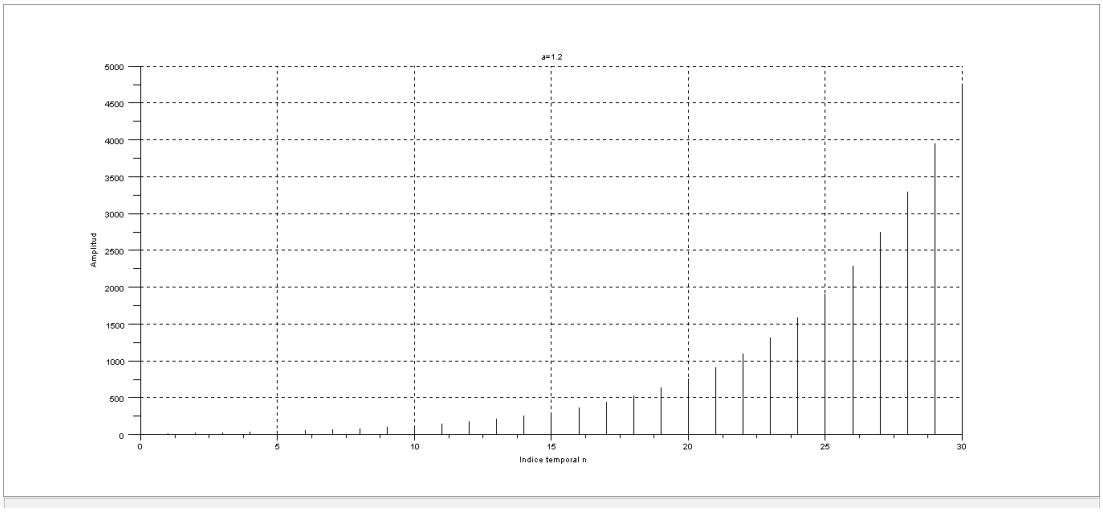
La figura muestra la parte real de la secuencia exponencial compleja realizada en SCILAB.



Muestra la parte imaginaria de la secuencia exponencial compleja realizada en SCILAB.

PROGRAMA 2

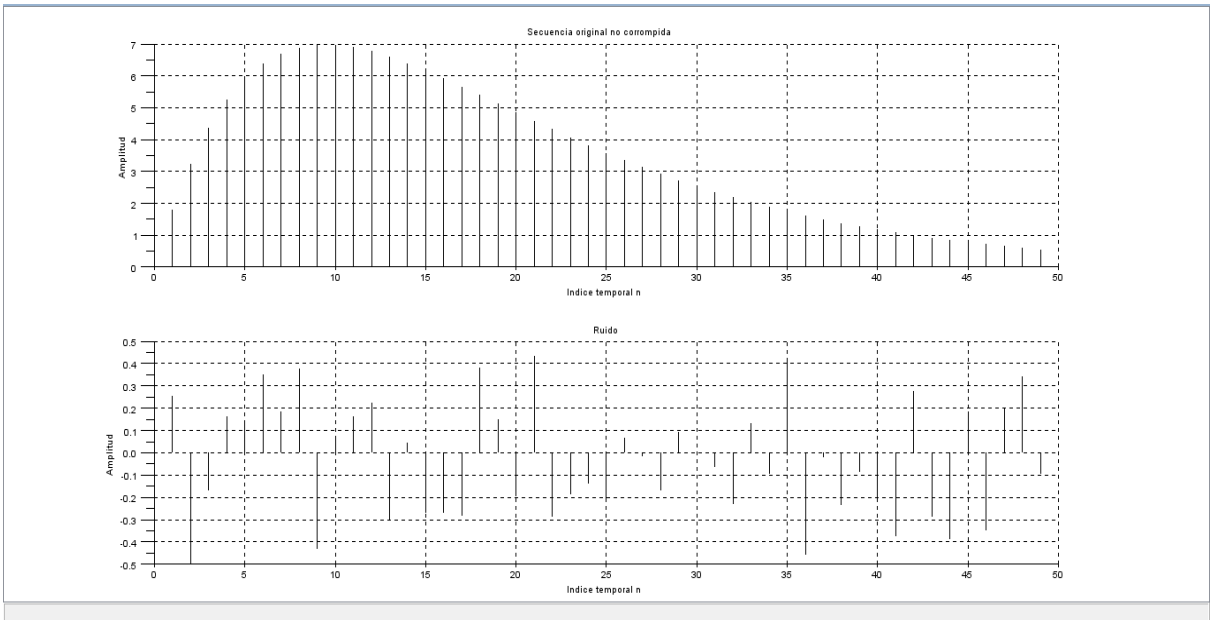
```
// Generacion de secuencia exponencial real

// a = 1.2; K = 0.2; N = 30;

// Pruebe tambien con

// a = 0.9; K = 20; N = 30

a = input('Introduzca exponente = ');
K = input('Introduzca la constante de ganancia = ');
N = input('Introduzca la longitud de la secuencia = ');
n = 0:N;
x = K*a.^n;
f0=scf(0);club;
plot2d3('gnn',n,x); xgrid;
xlabel('Indice temporal n');ylabel('Amplitud');
title('a=1.2');
disp('Teclee ENTER para continuar');
pause
```



Muestra la generación de una secuencia exponencial real de una función arbitraria.

PROGRAMA 3

```
R = 50;
d = rand(R,1) - 0.5;
// Para generar la señal corrompida

m = [0:1:R-1]';
s = 2*m.*(0.9.^m);
f0=scf;clf;
subplot(2,1,1);
plot2d3('gnn',m,s);xgrid;
xlabel('Indice temporal n');ylabel('Amplitud');
title('Secuencia original no corrompida');
subplot(2,1,2);
plot2d3('gnn',m,d);xgrid;
xlabel('Indice temporal n');ylabel('Amplitud');
title('Ruido');
disp('Teclee ENTER para continuar');
pause
```

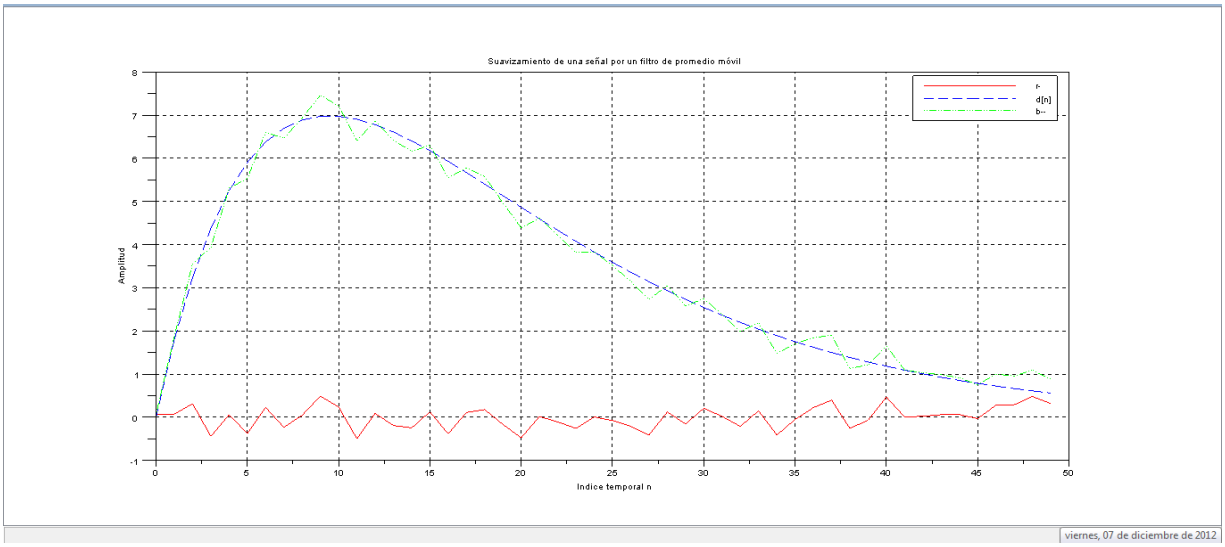


Muestrala Generación de Señales y Ruido Aleatorio

PROGRAMA 4

```
M = 3;
R = 50;
d = rand(R,1) - 0.5;
m = [0:1:R-1]';
s = 2*m.*(0.9.^m);
x = s + d;
f0=scf(0);clf;
plot(m,d,'r-',m,s,'b--',m,x,'g:');xgrid;
xlabel('Indice temporal n');ylabel('Amplitud');
legend('r-','d[n]','b--','s[n]','g:','x[n]');
disp('Teclee ENTER para realizar el suavizamiento');
pause
M = input('Numero de muestras de entrada = ');
b = ones(M,1)/M;
// y = filter(b,1,x);
y = filter(b,1,x);
f1=scf(1);clf;
scf(1);
plot(m,s,'r-',m,y,'b--');grid;
```

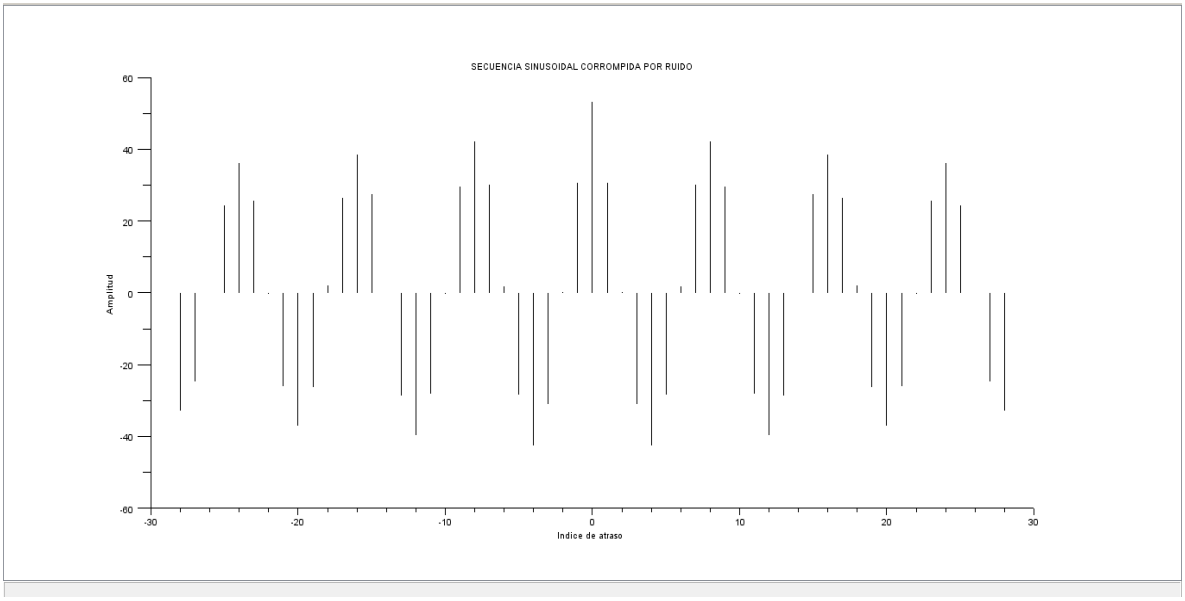
```
legend('r-','s[n]','b--','y[n]');
xlabel('Indice temporal n');ylabel('Amplitud');
xtitle("Suavizamiento de una señal por un filtro de promedio móvil")
```



Muestra el Suavizamiento de una Señal por un Filtro de Promedio Móvil.

PROGRAMA 5

```
N = 96;
n = 1:N;
x = cos(%pi*0.25*n); // Se genera la secuencia sinusoidal
d = rand(1,N) - 0.5; // Se genera la secuencia de ruido
y = x + d; // Se genera la secuencia sinusoidal corrompida por ruido
r = convol(y,mtlb_fliplr(y)); // Calcula la autocorrelacion
k = -28:28;
plot2d3('gnn',k,r(68:124));
xlabel('Indice de atraso');ylabel('Amplitud');
```



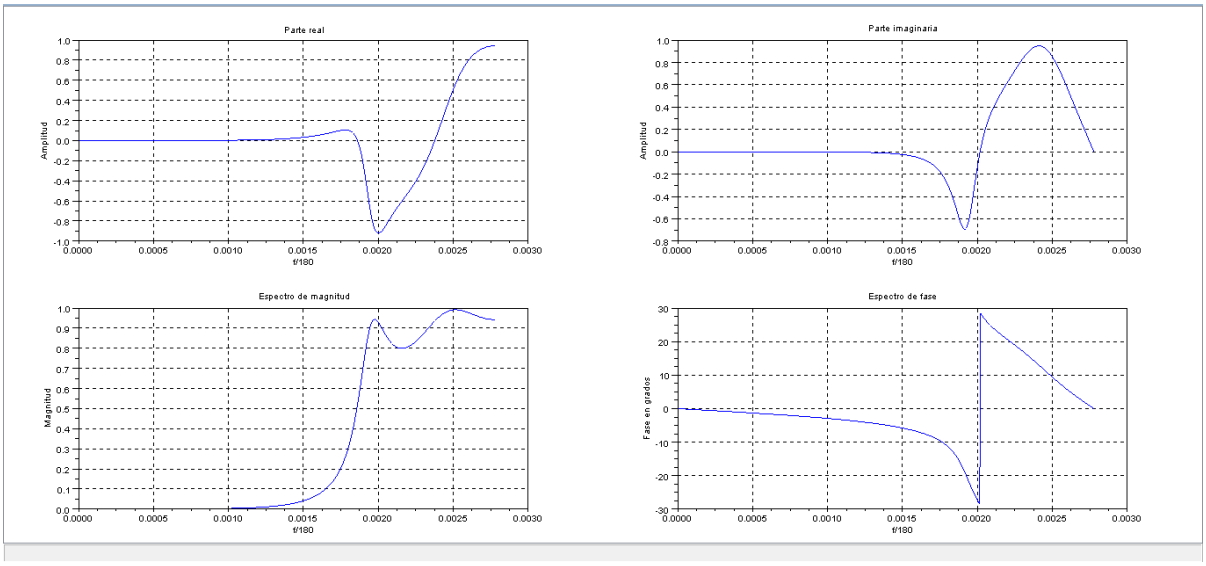
Muestra la autocorrelación de una secuencia sinusoidal corrompida por ruido

PROGRAMA 6

```
k = 256;
num = [0.008 -0.033 0.05 -0.033 0.008];
den = [1 2.37 2.7 1.6 0.41];

//Cálculo de la transformada de Fourier de tiempo discreto

// Lea la longitud deseada de la DFT

k = input('Numero de puntos de frecuencia = ');
// Lea los coeficientes del numerador y del denominador
num = input('Coeficientes del numerador = ');
den = input('Coeficientes del denominador = ');
```

```
// Calcule la respuesta en frecuencia
numh=poly(num,"z","coeff");
denh=poly(den,"z","coeff");
h=syslin('d',numh/denh);
[frq,bnds,split]=calfrq(h,0.001,%pi);
rf=repfreq(h,frq);
// Grafique la respuesta en frecuencia
subplot(2,2,1)
plot(frq/180,real(rf));xgrid;
title('Parte real');
xlabel('f/180');ylabel('Amplitud');
subplot(2,2,2)
plot(frq/180,-imag(rf));xgrid;
title('Parte imaginaria');
xlabel('f/180');ylabel('Amplitud');
subplot(2,2,3);
plot(frq/180,abs(rf));xgrid;
title('Espectro de magnitud');
xlabel('f/180');ylabel('Magnitud');
subplot(2,2,4);
plot(frq/180,-atand(imag(rf),real(rf))/(2*%pi));xgrid;
```

```
title('Espectro de fase');
xlabel('f/180');ylabel('Fase  en grados');
```

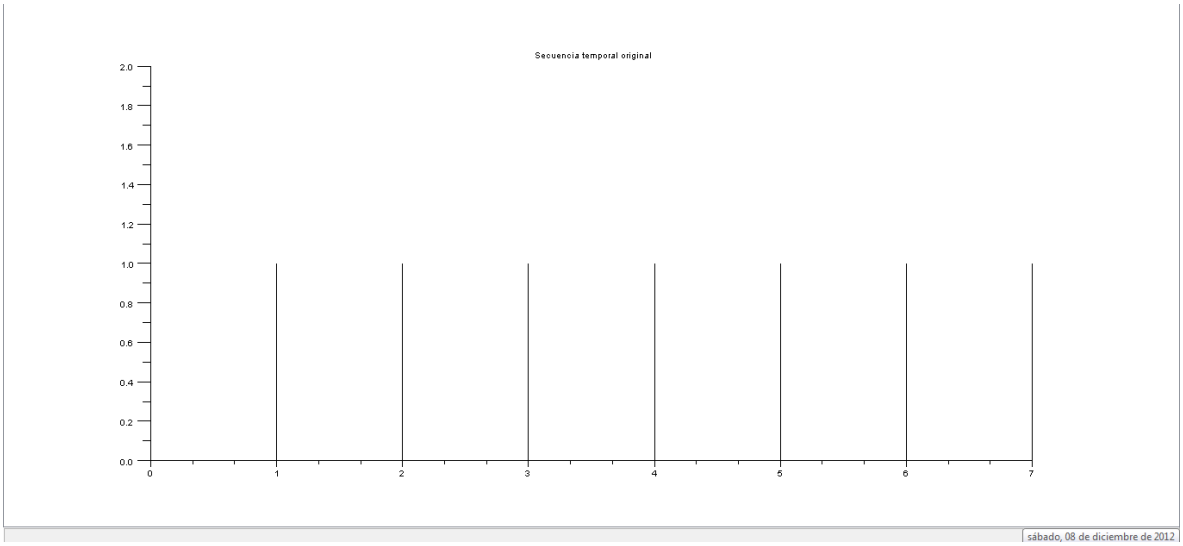


Muestra la Transformada De Fourier De Tiempo Discreto considerando su parte real e imaginaria (parte superior) como su espectro de magnitud y de fase (parte inferior)

PROGRAMA 7

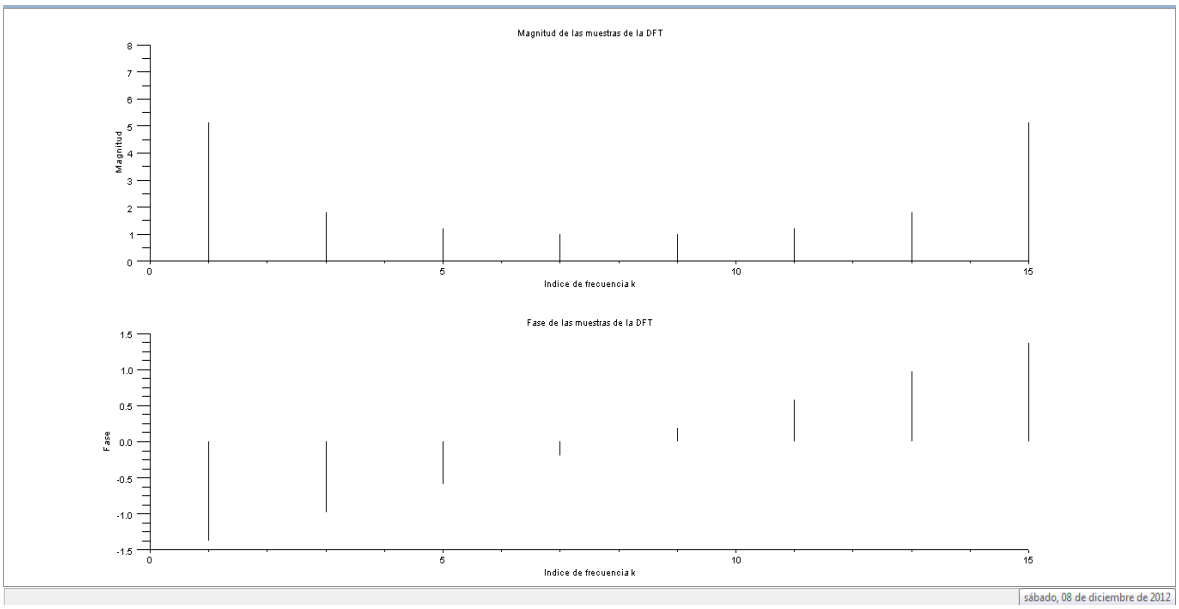
```
// Ilustración del cálculo de la DFT
// N=8; M=16;
// Lea la longitud N de la secuencia y la longitud deseada M de la DFT
N = input('Introduzca la longitud de la secuencia = ');
M = input('Introduzca la longitud de la DFT = ');
// Genere la secuencia temporal de longitud N
u = [ones(1,N)];
// Calcule su DFT de M puntos
U = mtlb_fft(u,M);
// Grafique la secuencia temporal y su DFT
t = 0:1:N-1;
plot2d3('gnn',t,u);
title('Secuencia temporal original');
xlabel('Indice temporal n');ylabel('Amplitud');
pause
subplot(2,1,1);
k = 0:1:M-1;
plot2d3('gnn',k,abs(U));
title('Magnitud de las muestras de la DFT');
xlabel('Indice de frecuencia k');ylabel('Magnitud');
subplot(2,1,2);
plot2d3('gnn',k,atan(imag(U),real(U)) );
title('Fase de las muestras de la DFT');
xlabel('Indice de frecuencia k');ylabel('Fase');
```

Gráfico 1:



Muestra la secuencia temporal original de la transformada discreta de Fourier generado por SCILAB.

Gráfico 2:

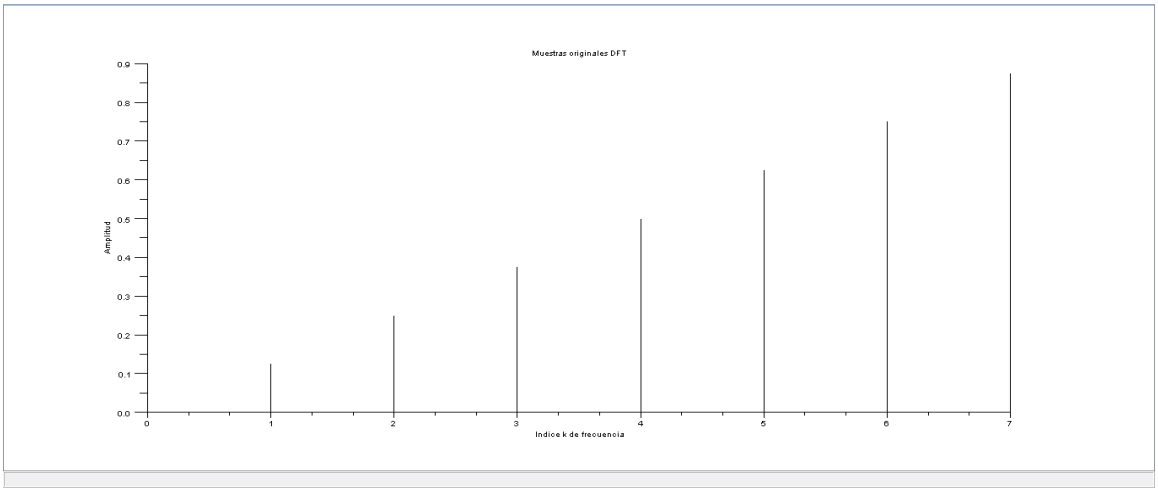


Muestra la magnitud de las muestras de una DFT (parte superior) y la fase de la muestra en la parte inferior.

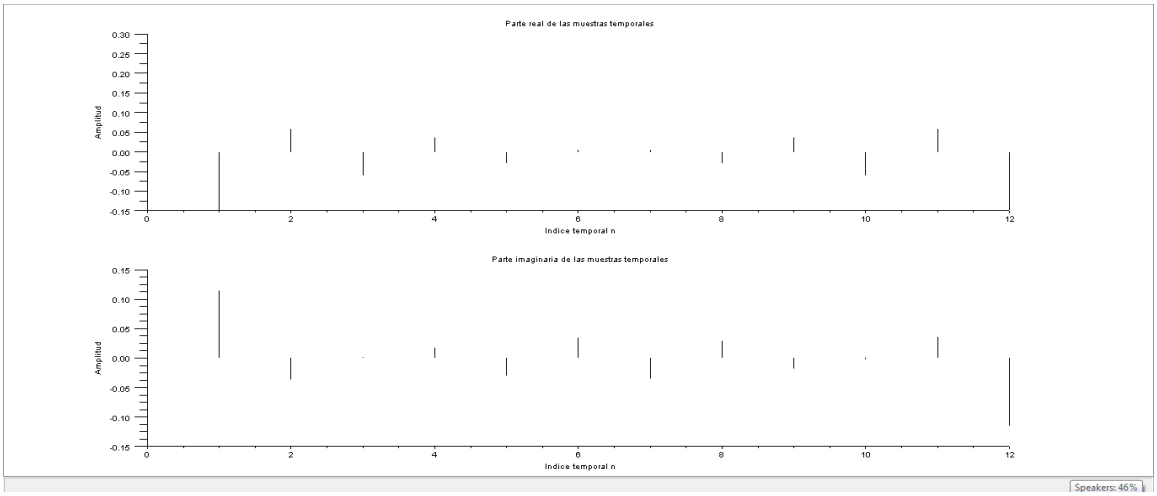
Programa 8

```
// Ilustración del cálculo de la DFT inversa

// K=8; N=13;

// Lea la longitud K de la DFT y la longitud deseada N de la DFT inversa

K = input('Introduzca la longitud de la DFT = ');
N = input('Introduzca la longitud de la DFT inversa = ');
// Genere la secuencia DFT de longitud K
// Se trata de una rampa discreta
k = 1:K;
U = (k-1)/K;
// Calcule su DFT inversa de N puntos
u = mtlb_ifft(U,N);
// Grafique la DFT y la DFT inversa
k = 1:K;
plot2d3('gnn',k-1,U);
xlabel('Indice k de frecuencia');ylabel('Amplitud');
title('Muestras originales DFT')
pause
subplot(2,1,1)
n = 0:1:N-1;
plot2d3('gnn',n,real(u));
title('Parte real de las muestras temporales');
xlabel('Indice temporal n');ylabel('Amplitud');
subplot(2,1,2)
plot2d3('gnn',n,imag(u));
```

```
title('Parte imaginaria de las muestras temporales');
xlabel('Indice temporal n');ylabel('Amplitud');
```



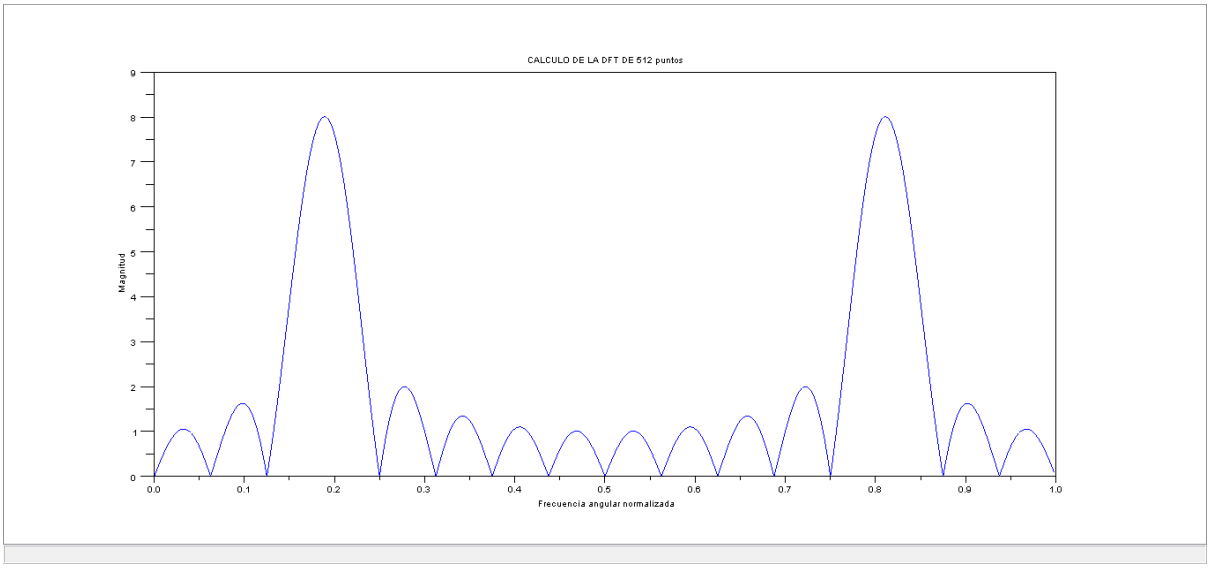
Muestras originales de la DFT generados en SCILAB.



Muestra la parte real de las muestras temporales (parte superior) y la parte imaginaria de las muestras temporales (parte inferior).

PROGRAMA 9

```
// Cálculo numérico de la DTFT usando la DFT

// Genere la secuencia sinusoidal de longitud 16

k = 0:15;
x = cos(2*%pi*k*3/16);
// Calcule su DFT de 512 puntos
X = mtlb_fft(x);
XE = mtlb_fft(x,512);
// Grafique la respuesta en frecuencia
L = 0:511;
plot(L/512,abs(XE));
mtlb_hold
plot(k/16,abs(X),'o');
xlabel('Frecuencia angular normalizada');
ylabel('Magnitud');
title("CALCULO DE LA DFT DE 512 puntos")
```



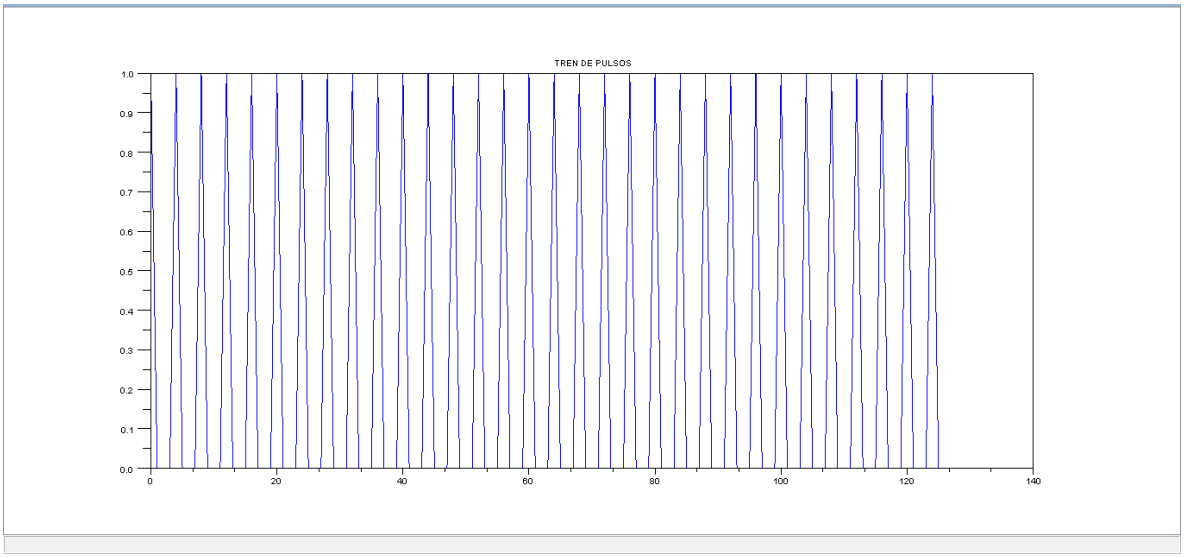
Muestra el cálculo de la DFT en una función generada con 512 puntos.

PROGRAMA 10

En esta práctica se trata el tema de muestreo y conversión digital analógica, el traslape y la interpolación y decimación.

a. Muestreo y Conversión Digital/Analógica

```
a = [1 0 0 0 -1];
d = [1 zeros(1,127)];
train = filter(1,a,d);
n = 0:127;
plot(n,train);
//Grafiquese el vector train contra el vector n.
```

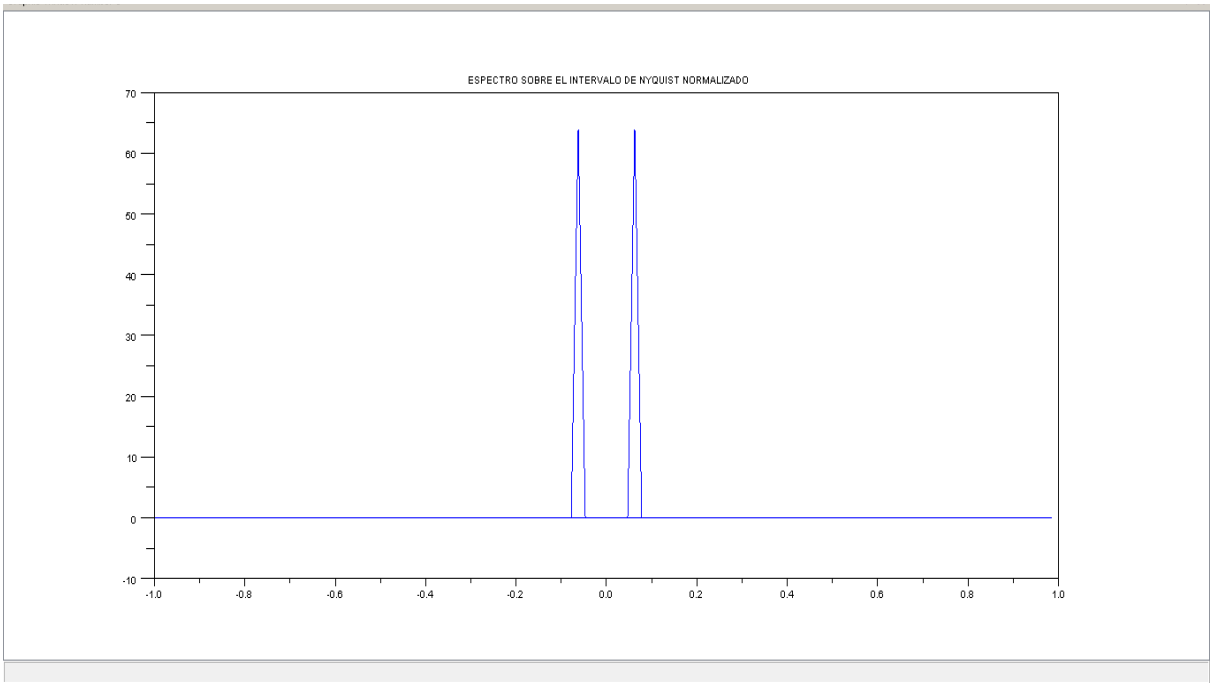


Generación un tren de impulsos de tiempo discreto para su empleo en muestreo.

Generación de una señal coseno con 32 muestras por ciclo mediante la función x.

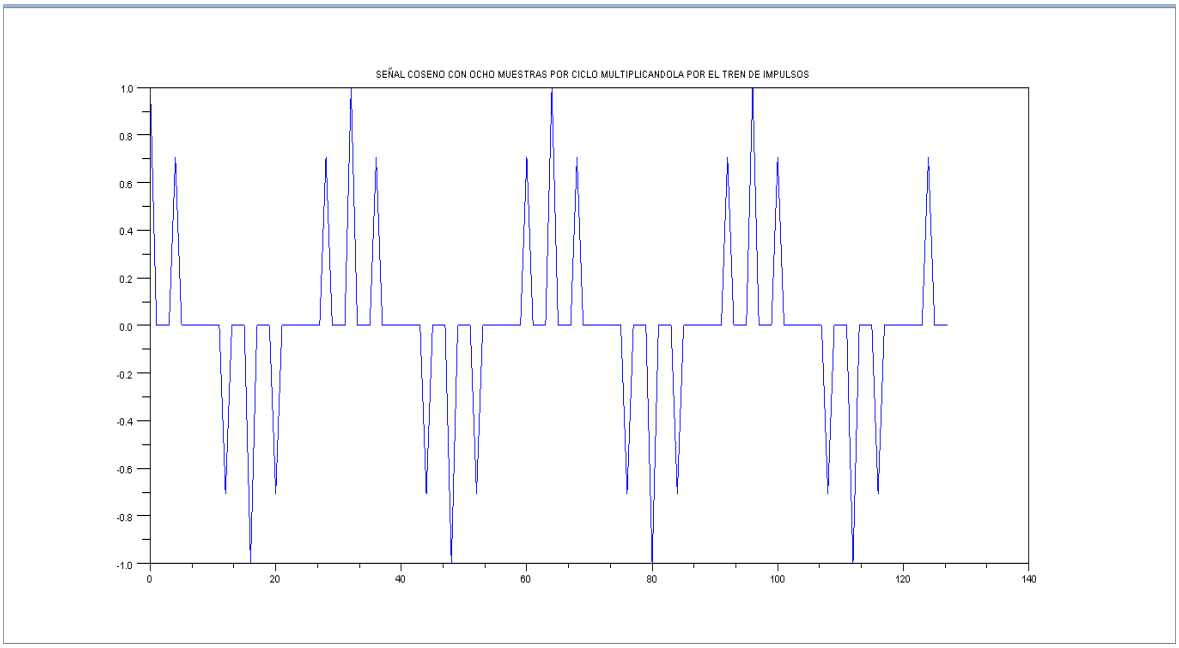
El espectro correspondiente sobre el intervalo de Nyquist normalizado (-1,1).

```
x = cos(2*%pi*n/32);
s=1; nfft =128;b=x;a=1;
nb = length(b);
na = length(a);
a = [a zeros(1,nb-na)];
b = [b zeros(1,na-nb)];
n = length(a);
h = (mtlb_fft(b,s*nfft)./mtlb_fft(a,s*nfft)).';
h = h(1:nfft);
h = h(:);
hh = h;
deltaF = (2*%pi)/nfft;
w = linspace(0,2*%pi-deltaF,nfft);
w = w(:);
wp = mtlb_fftshift(w)/%pi  -1;
plot(wp,hh);
```



Generación de una señal coseno con 32 muestras por ciclo.

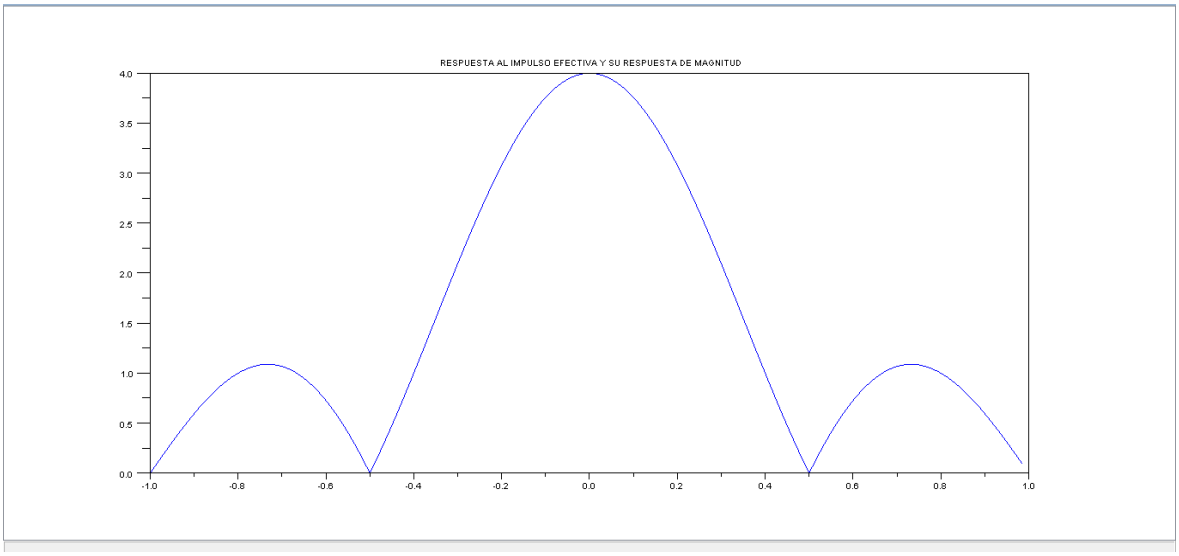
```
n = 0:127;
a = [1 0 0 0 -1];
d = [1 zeros(1,127)];
train = filter(1,a,d);
x = cos(2*%pi*n/32);
y = x.*train;
M=8;
yy=intdec(y,M/32);
nn=[0:length(yy)-1];
plot2d3('gnn',nn,yy-3);
plot(n,y);
title("SEÑAL COSENO CON OCHO MUESTRAS POR CICLO
MULTIPLICANDOLA  POR EL TREN DE IMPULSOS")
```



Generación de una señal coseno con ocho muestras por ciclo multiplicada por el tren de impulsos.

Respuesta al impulso efectiva y su respuesta de magnitud:

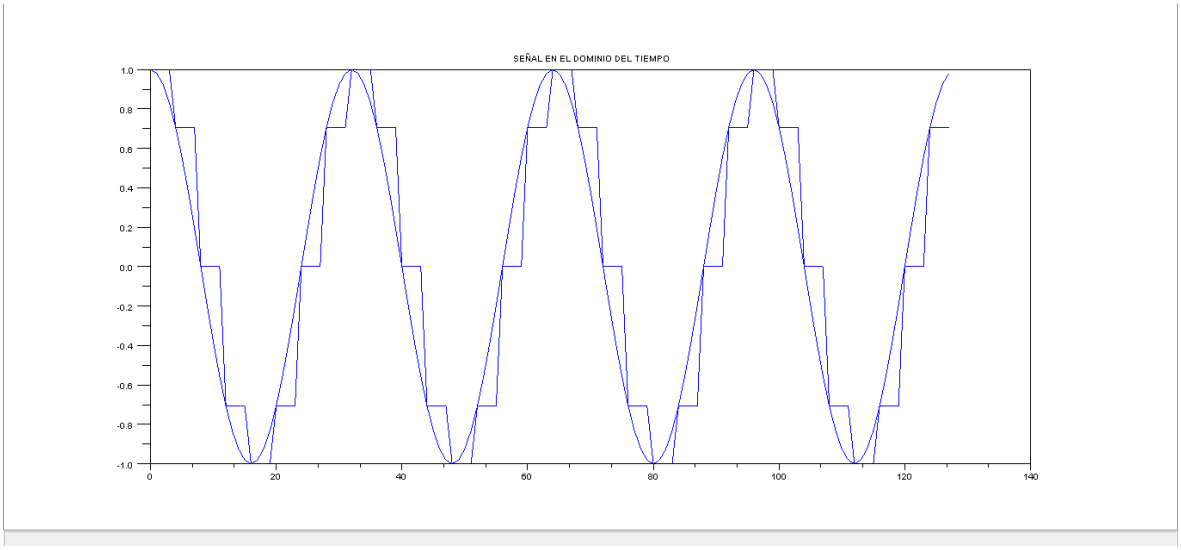
```
h = ones(1,4);
H = mtlb_fft(h,128);
plot(mtlb_fftshift(wp),mtlb_fftshift(abs(H)));mtlb_hold;
```



Respuesta al impulso efectivo y su respuesta de magnitud

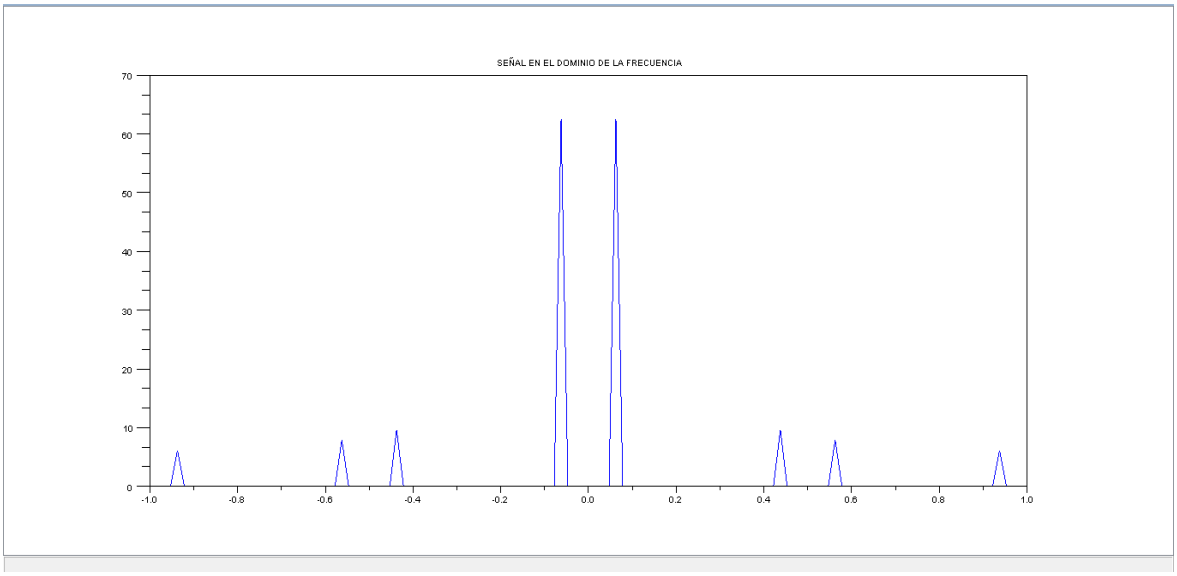
Aplicando este filtro D/A a la señal muestreada.

```
yd = filter(h,1,y);
plot(n,yd);plot(n,x)
```



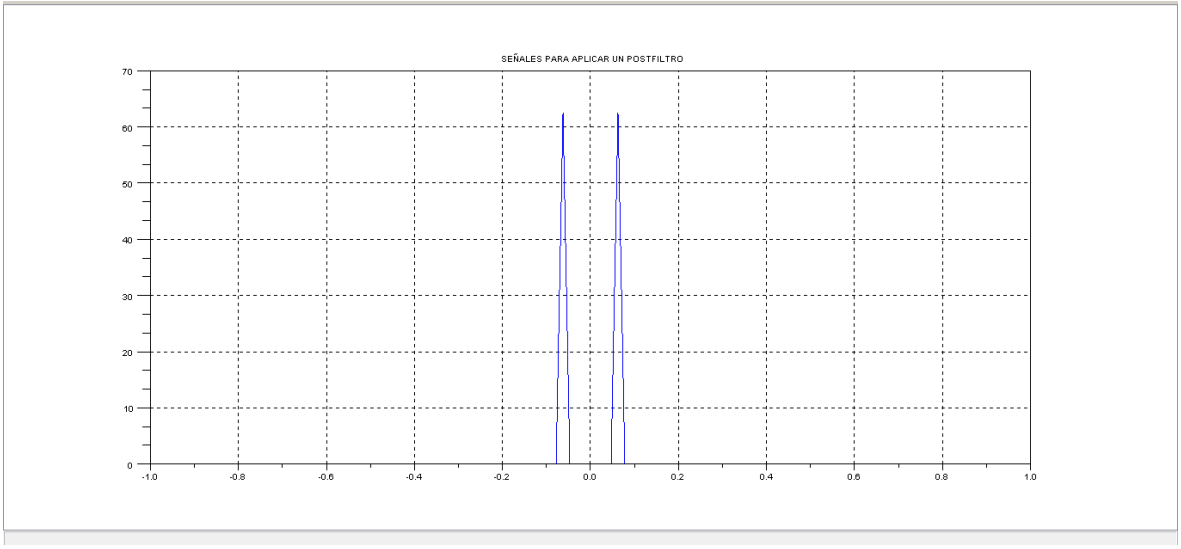
Señal muestreada en el dominio del tiempo.

```
Yd = mtlb_fft(yd);
plot(wp,abs(Yd));mtlb_hold;
```



Señal muestreada en el dominio de la frecuencia.

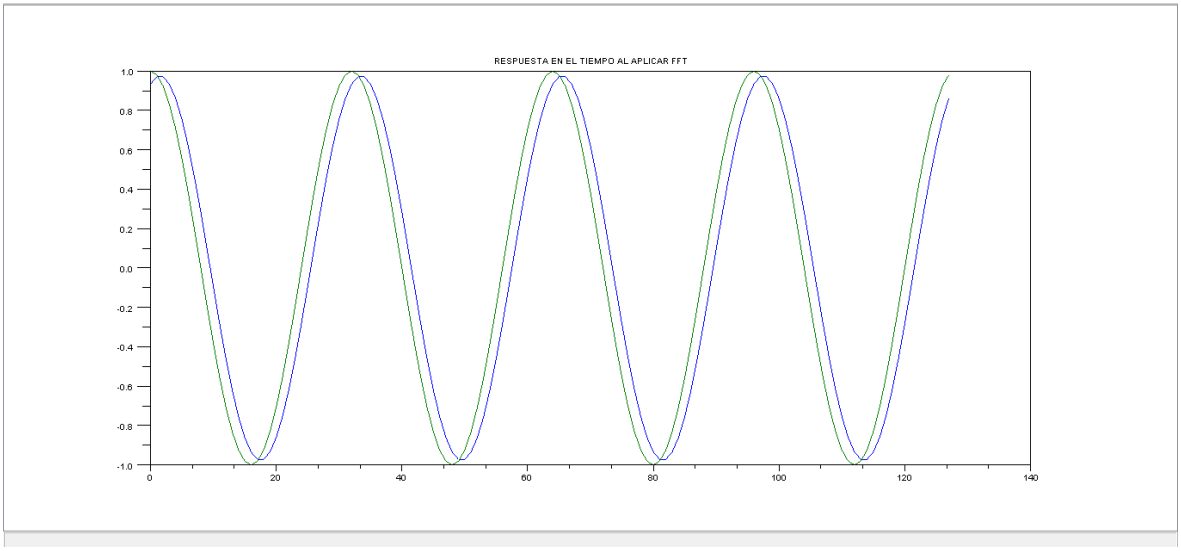
```
Yc = Yd;Yc(17:112)  = zeros([0:95]);
plot(wp,abs(Yc));xgrid;
title("SEÑALES PARA APLICAR UN POSTFILTRO")
```



Señal con un postfiltro ideal y una frecuencia de corte agregada

Entonces la respuesta en el tiempo al aplicar FFT es la siguiente:

```
yc = mtlb_ifft(Yc);
plot(n,yc,n,x);mtlb_hold;
title("RESPUESTA EN EL TIEMPO AL APLICAR FFT")
```



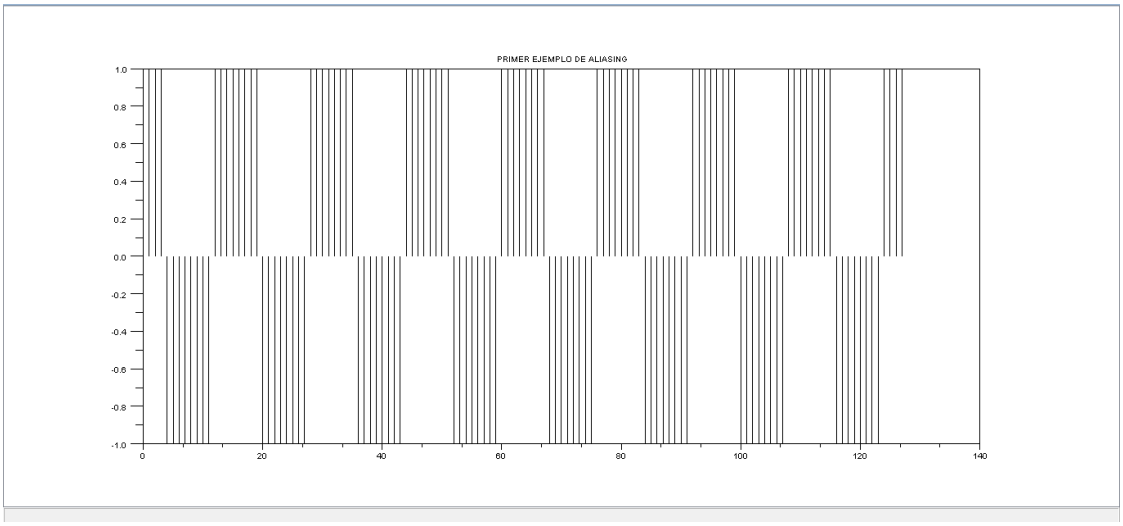
Respuesta en el tiempo al aplicar FFT.

Nótese el ligero desplazamiento de fase y la atenuación en la amplitud en la salida del postfiltro contra la señal original causada por el convertidor D/A.

b. Traslape (“Aliasing”)

El pequeño desplazamiento de fase (0.01) asegura un signo correcto.

```
n = 0:127;
x = cos(2*%pi*n*8/128 + 0.01);
y = sign(x);
plot2d3('gnn',n,y);
title("PRIMER  EJEMPLO DE ALIASING")
```

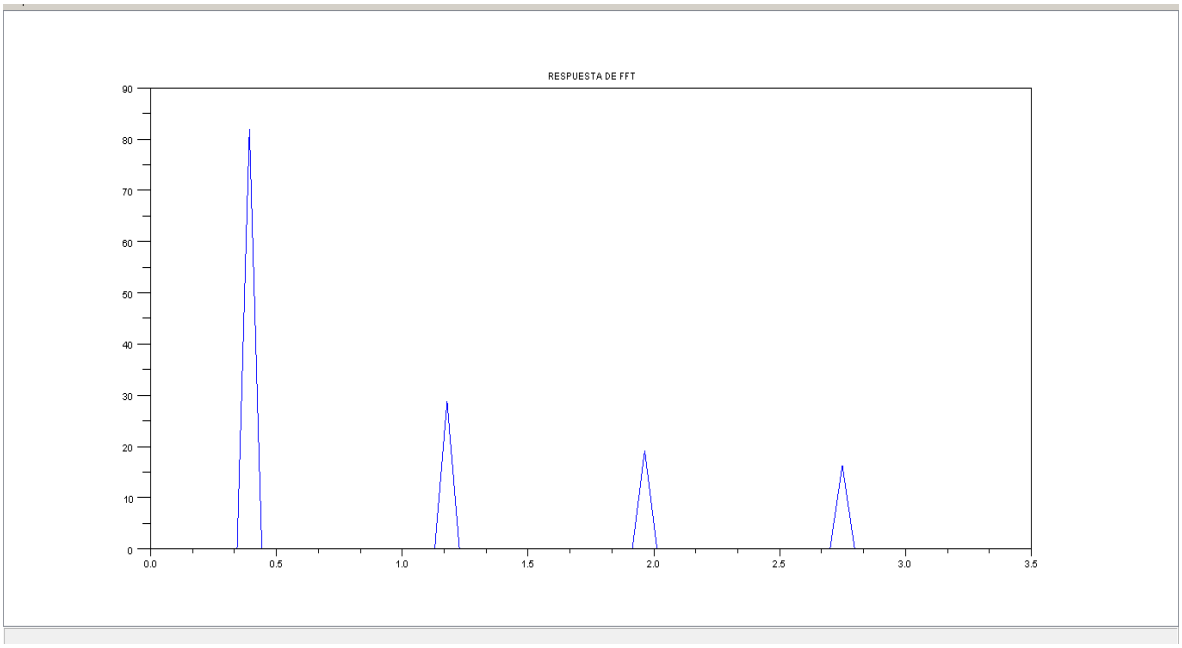


Generación de una onda cuadrada en tiempo discreto (Primer ejemplo de ALIASING)

Nótese que la onda cuadrada tiene solo armónicos impares, que decaen de la forma 1/ k (pero no tan exactamente, pues las armónicas superiores para k = 9, 11, 13,.., caen encima de las correspondientes para k = 1, 3, 5, 7).

```
b=y;a=1;s=2; nfft =64;
nb = length(b);
113
na = length(a);
a = [a zeros(1,nb-na)];
b = [b zeros(1,na-nb)];
n = length(a);
```

```
h = (mtlb_fft(b,s*nfft)./mtlb_fft(a,s*nfft)).';
h = h(1:nfft);
h = h(:);
hh = h;
deltaF = %pi/nfft;
w = linspace(0,%pi-deltaF,nfft);
w = w(:);
mag = abs(hh);
plot(w,mag);
title("RESPUESTA DE FFT")
```

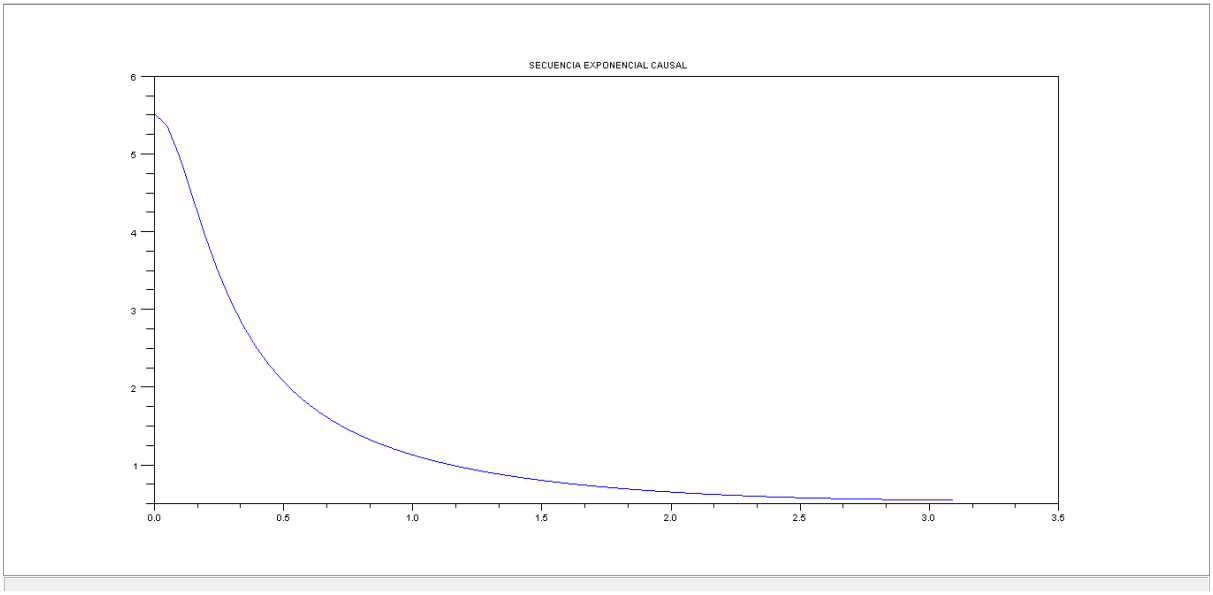


Respuesta de FFT

Genérese ahora una secuencia exponencial causal y su espectro de magnitud mediante

```
n=0:127;
x = exp(-n/5);
b=x;a=1;s=2; nfft =64;
nb = length(b);
na = length(a);
a = [a zeros(1,nb-na)];
b = [b zeros(1,na-nb)];
n = length(a);
```

```
h = (mtlb_fft(b,s*nfft)./mtlb_fft(a,s*nfft)).';
h = h(1:nfft);
h = h(:);
hh = h;
deltaF = %pi/nfft;
w = linspace(0,%pi-deltaF,nfft);
w = w(:);
magX = abs(hh);
plot(w,magX);
title("SECUENCIA EXPONENCIAL CAUSAL")
```



Generación de una secuencia exponencial causal y su espectro de magnitud.

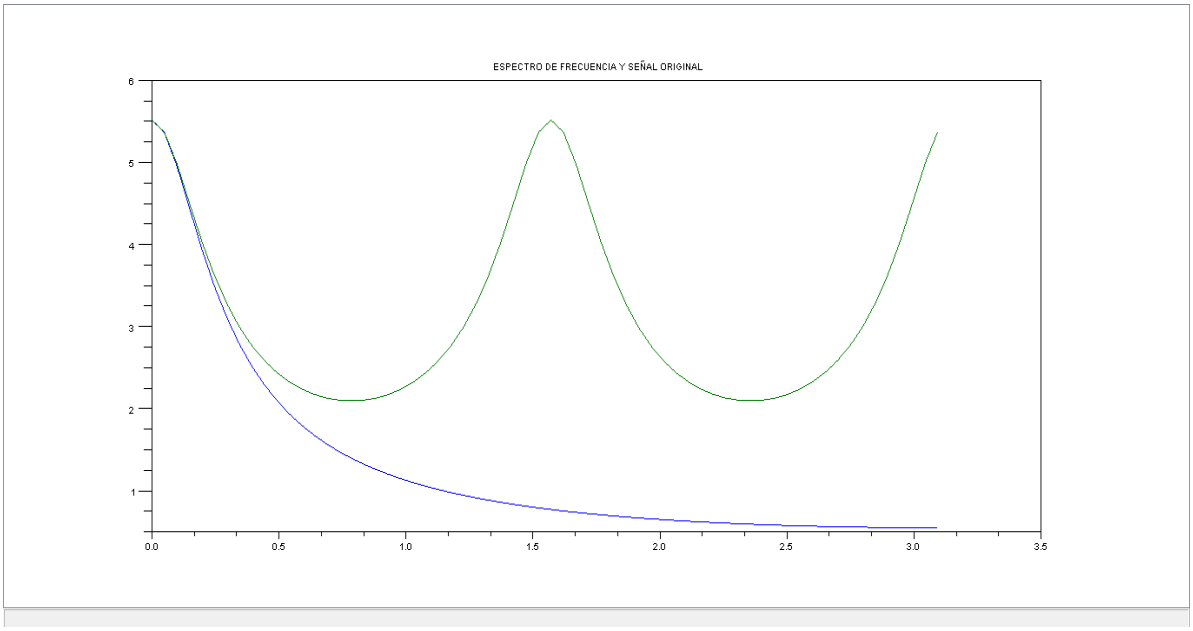
Espectro de frecuencia y señal original se grafica de la siguiente forma:

```
a = [1 0 0 0 -1];
d = [1 zeros(1,127)];
train = filter(1,a,d);
y = x.*train;
```

Para computar el espectro correspondiente y para compararlo con el del exponencial original (después de un escalamiento adecuado), se ejecuta los mandatos,

```
b=x;a=1;s=2; nfft =64;
nb = length(b);
na = length(a);
a = [a zeros(1,nb-na)];
b = [b zeros(1,na-nb)];
n = length(a);
h = (mtlb_fft(b,s*nfft)./mtlb_fft(a,s*nfft)).';
h = h(1:nfft);
h = h(:);
hh = h;
deltaF = %pi/nfft;
w = linspace(0,%pi-deltaF,nfft);
w = w(:);
magX = abs(hh);
by=y;ay=1;sy=2; nffty =64;
nby = length(by);
nay = length(ay);
ay = [ay zeros(1,nby-nay)];
by = [by zeros(1,nay-nby)];
ny = length(ay);
116
yy = (mtlb_fft(by,sy*nffty)./mtlb_fft(ay,sy*nffty)).';
yy = yy(1:nffty);
yy = yy(:);
Y = yy;
deltaF = %pi/nfft;
w = linspace(0,%pi-deltaF,nfft);
w = w(:);
magY = abs(Y);
magY = magY*magX(1)/magY(1);
```

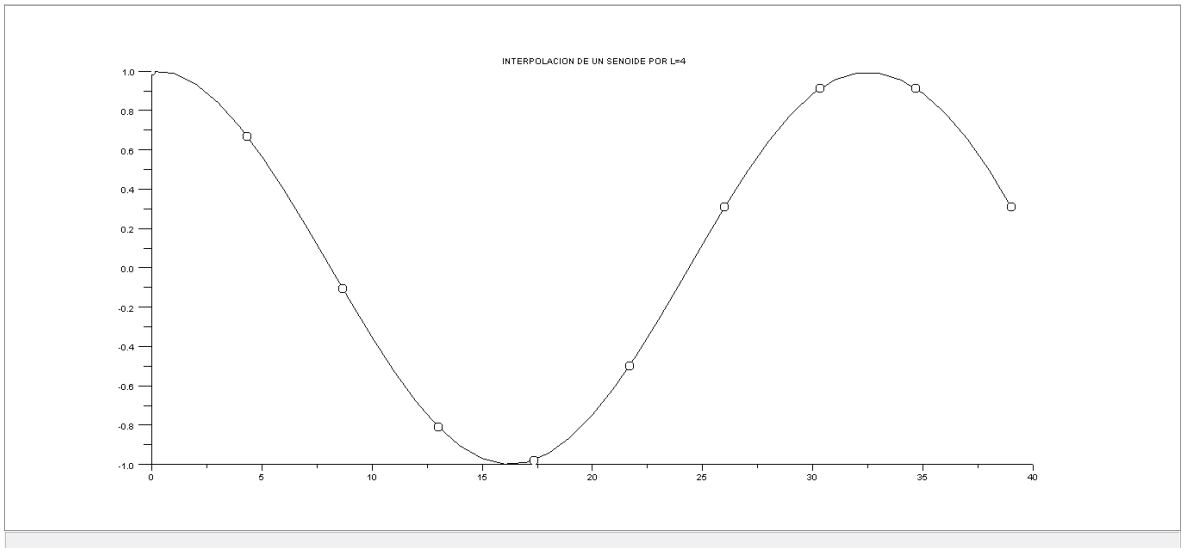
```
plot(w,magX,w,magY);
title("ESPECTRO DE FRECUENCIA Y SEÑAL ORIGINAL")
```



Espectro de frecuencia y señal original

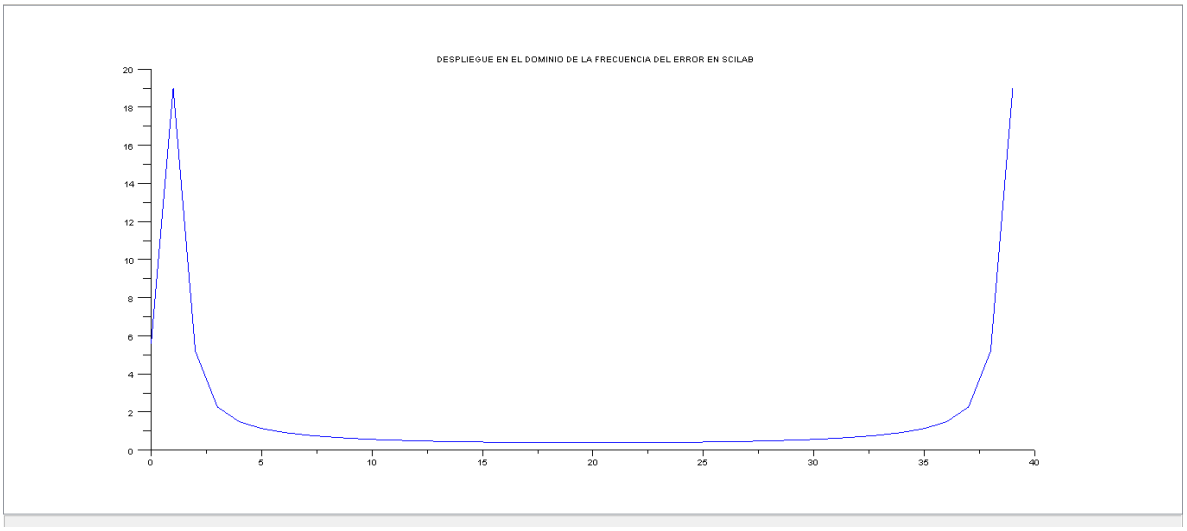
c. Interpolación Y Decimación

```
x = linspace(0,39,10)';
y = cos(2*%pi*x/5);
dk = splin(x,y);
xx = linspace(0,39,40)';
[yyk, yy1k, yy2k] = interp(xx, x, y, dk);
plot2d(xx, [yyk])
plot2d(x, y, -9)
```



Interpolación y decimación de secuencias de un senoide.

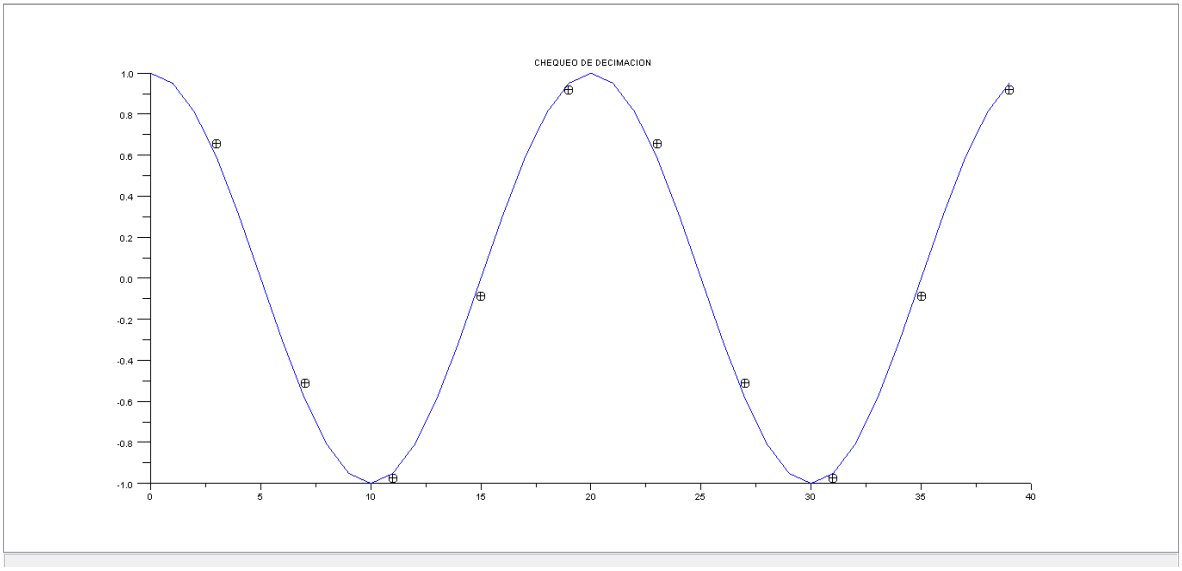
```
Y = mtlb_fft(yyk);
plot2d(m,abs(Y),2);mtlb_hold;
```



Despliegue en el dominio de la frecuencia del error.

Nótese las respuestas al impulso y de frecuencia (variantes con el tiempo) del filtro interpolador. Ahora se invierte el experimento para chequear la decimación, o sea,

```
m = 0:39; n = 0:9;
xi = cos(2*%pi*m/20);
xii = cos(2*%pi*m/20-5.43);
yy=intdec(xii,1/4);
plot2d(4*n+3,yy,-3);plot(m,xi);
```



Resultado de la decimación e interpolación de la sinusoide.

Todos los ejercicios de esta tesis fueron desarrollados utilizando el Software SCILAB, el cual está diseñado bajo las normativas de software libre, lo que garantiza que SCILAB siempre se mantendrá de fuente abierta entre muchas otras cosas.