Aplicación en Scilab para el tratamiento de señales digitales con la Transformación de Fourier fraccionaria: Filtro de Wiener fraccionario

Presentado por:
MARCOS AMARIS GONZALEZ

Universidad del Magdalena
Facultad de Ingeniería
Programa de Ingeniería de Sistemas
Santa Marta D.T.C.H.
Abril de 2009

Aplicación en Scilab para el tratamiento de señales digitales con la transformación de Fourier fraccionaria: Filtro de Wiener fraccionario

Presentada por: Marcos Amaris González

Dirigida por: Dr. Rafael Ángel Torres Amaris

Trabajo de grado presentado como requisito para optar al titulo de Ingeniero de Sistemas

Universidad del Magdalena

Facultad de Ingeniera

Ingeniería de Sistemas

Abril de 2009

NOTA DE ACEPTACION

_____

_____

_____

Jurados

_____

Ms.c. German Sánchez Torres

_____

Ms. Omar Francisco Rodriguez Alvarez

Santa Marta, Colombia. Abril de 2009

A mi familia, la cual me ha

apoyado incondicionalmente.

Si he visto más lejos que otros hombres, es porque me he aupado a hombros de gigantes.

Albert Einstein

AGRADECIMIENTOS

A mi madre; que es mi pilar y que sin su sacrificio y amor no hubiese podido salir adelante.

A los profesores amigos, que fueron excelentes maestros y han sido ejemplo en mi formación como persona y profesional.

A los que me brindaron su mano, y me dieron confianza, exhortándome a soñar, Tito.

Al asesor de esta investigación por muchas razones; a Zandra y a su familia.

A los que me cerraron las puertas, porque me dieron la fortaleza de buscar otros horizontes.

A mis amigos y amigas, que sonrieron conmigo en momentos difíciles y con quienes compartí risas, lágrimas y experiencias de vida; porque el sol se oculta felizmente al atardecer detrás del mar, cuando se cuenta con la confianza y el apoyo de los amigos.

Tabla de Contenido

Lista de Figuras

Lista de Tablas

Resumen

“ ***El teorema de Fourier no es solamente uno de los más hermosos resultados del análisis moderno, sino que se ha convertido en instrumento indispensable de la física moderna.***”[1] .

En 1980 surgió una nueva operación para el tratamiento de señales inventada por V. Namías, desde ese momento surgieron operaciones como la convolución y correlación fraccionaria, el teorema de muestreo en dominios de Fourier fraccionarios y nuevos filtros en el dominio de esta transformación; también se encontraron sus propiedades y relaciones con otras funciones, de la cual surgió una de las más importantes, la cual es que la $FrFT$ es una rotación de la distribución de Wigner que se encuentra asociada a una señal, esto ha servido como una excelente representación de señales en tiempo-frecuencia y han surgido muchas implementaciones para el tratamiento de señales de muchas categorías entre ellas las no estacionarias.

En este documento se muestra la implementación de un algoritmo de la transformación de Fourier fraccionaria haciendo uso del teorema de muestreo en dominios fraccionarios, se demuestra la rotación de la distribución de Wigner asociada a una

[1] Lord Kelvin

señal por medio de la $FrFT$, y se implementa el filtro de Wiener fraccionario utilizando la convolución fraccionaria.

En el capitulo 2 se propone el tratamiento de señales con la $FrFT$ implementada en las operaciones necesarias, con el fin de realizar el filtro de Wiener fraccionario, en el capitulo 3 los antecedentes de esta investigación, en el 4 la teoría de la $FrFT$, en el 5 la justificación, en el 6 los objetivos, en el capitulo 7 se formulan algunas ecuaciones para implementarlas computacionalmente, en 8 la Metodología XP como metodología de ciclo de vida de desarrollo de software, en el 9 y en el 10 el desarrollo de las funciones y la demostración por medio de simulación. En el Capitulo 11 se exponen algunos inconvenientes y limitaciones durante la investigación y al final 12 las conclusiones.

Capítulo 1

Introducción

Desde la invención de la transformación de Fourier, esta siempre ha ocupado los primeros puestos en el tratamiento de señales [1], no obstante, en varias ocasiones esta ha tenido que acondicionarse para realizar un estudio de señales que requierren un análisis más especializado [2], anteriormente no se podía obtener una información real y concreta de señales que variaran su amplitud y frecuencia aperiódicamente, ya que como principal condición de la transformación de Fourier y de otras transformaciones adaptadas de ella, es que las señales a analizar deben ser periodicas. Las señales no estacionarias son una clase de señales que no cumplen con esta condición [3].

Hace poco tiempo surgió la invención de una nueva transformación [4], la cual tiene como caso particular la transformación de Fourier, la transformación de Fourier inversa y la misma función de entrada, su nombre es Transformación de Fourier fraccionaria (Fractional Fourier Transform, $FrFT$ por sus siglas en inglés). La $FrFT$ puede ser vista como una excelente representación de señales en tiempo-frecuencia [5] pudiendo obtener frecuencias presentes en la señal en tiempos dados. La $FrFT$ puede verse como una rotación de la distribución de Wigner asociada a una señal

[6], donde la rotación está dada por el angulo $\alpha = a\pi/2$, siendo a el $a-esimo$ orden fraccionario de la transformación.

Además, la $FrFT$ trajo consigo un nuevo enfoque fraccionario para el tratamiento de señales, entre ellos el teorema de muestreo en dominios de Fourier fraccionarios [7, 8], la convolución fraccionaria, la correlación fraccionaria [9, 10], filtro de Wiener fraccionario [11, 12] y seguirá avanzando, trayendo nuevos modelos para el tratamiento de señales.

Capítulo 2

Planteamiento del problema

La búsqueda de métodos para extraer y recuperar información que poseen las señales de cualquier tipo y categoría, con el objeto de conocer atributos relacionados a los procesos físicos que las generan, siempre ha sido tema de interés para ingenieros y matemáticos. La primera división natural de todas las señales son las estacionarias y las no estacionarias [2]. Las señales estacionarias son constantes en sus parámetros estadísticos (media aritmética, variación estándar, entre otras) y las señales no estacionarias son aquellas que sus parámetros de amplitud y/o frecuencia pueden variar a lo largo y ancho del tiempo [13], ejemplos de estas señales, son los exámenes médicos como electrocardiogramas, electroenfalogramas, señales de presión, análisis de sismología, entre otras.

Una de las técnicas más utilizadas ha sido representar las señales en el dominio del tiempo; esto ha permitido asociar cambios de la señal con los parámetros amplitud-tiempo de la señal. Posteriormente, el dominio de la frecuencia vino a complementar esta información. El análisis de Fourier informa acerca de la presencia o ausencia de determinadas frecuencias en la señal analizada [2]. La definición de la transforma-

ción de Fourier está dada por la siguiente ecuación.

$$F(y) = \int_{-\infty}^{\infty} f(x)e^{-2ixy}dx \tag{2.1}$$

Para todo x sea este número real, donde x es la variable del dominio directo de la función de entrada e y es la variable a ser representada en la frecuencia.

La transformación de Fourier (Fourier Transform) es una herramienta de gran utilidad para el Tratamiento de señales estacionarias, sobre todo desde el descubrimiento de la transformación rápida de Fourier [13], pero para el tratamiento de señales no estacionarias la transformación de Fourier estándar presenta más inconvenientes que ventajas ya que esta práctica común adolece de una limitada resolución en frecuencia haciéndola algunas veces inútil para el análisis de estas señales [14], ya que esta transformación supone que el espectro y la frecuencia de las señales son uniformes durante todo el periodo muestreado, y en la práctica podemos observar que muchas señales tanto la frecuencia como la amplitud varían en el tiempo, por tal razón al aplicar la FT en estas señales, ni la representación temporal ni la representación frecuencial arrojarán información sustanciosa del proceso.

La conversión de una señal análoga a digital se realiza a través de un proceso de muestreo, el teorema de Claude Shannon (padre de la Teoría de la Información), éste formula el proceso de muestreo de una señal continua como la frecuencia de muestreo mínima para garantizar condiciones óptimas de fidelidad de la señal tanto en su representación digital como en su reconstrucción análoga, esta frecuencia es comúnmente llamada frecuencia de Nyquist [15].

El teorema del muestreo de Shannon, también se conoce como teorema de muestreo

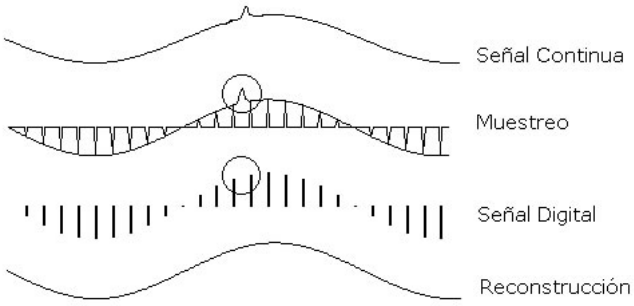

Figura 2.1: Fenómeno de aliasing durante un proceso de muestreo y reconstrucción

de Whittaker-Nyquist-Kotelnikov-Shannon, utilizado básicamente para el tratamiento de señales, el cual establece una frecuencia mínima de muestreo necesaria para evitar la superposición de los espectros "Aliasing"y hacer una perfecta reconstrucción análoga de una señal $f(x)$.

Según el anterior teorema, si al muestrear una señal con función $f(x)$, a una frecuencia menor que $1/\zeta$, siendo ζ el ancho de banda de la transformada asociada a la función, entonces al intentar reconstruir la señal original podemos obtener frecuencias que esta no contenía, es decir, podemos confundir una frecuencia $f1$ con otra $f2$, por ello a $f2$ se le llama el alias de f, a este fenómeno se le conoce en el tratamiento de señales como alliasing [16].

En la Figura 2.1, podemos observar el proceso de muestreo de una señal continua con puntos equidistantes entre sí, para su posterior reconstrucción. Pero el espectro resultante es ficticio, ya que contiene frecuencias diferentes, lo anterior debido a que la señal continua al ser reconstruida nuevamente no posee la frecuencia que se encuentra encerrada en el circulo, esto se debe a que la digitalización en el dominio de tiempo es muy grande y posee errores de amplitud o espacios de frecuencia los que

no puede llegar por su periocidad.

Bajo estas necesidades surge la representación tiempo-frecuencia [16], ya que bajo estas escalas se puede hacer un buen estudio de la frecuencia y la amplitud de la señales a lo largo del tiempo de las mismas, existen varias técnicas para la representación de señales en tiempo-frecuencia, estas técnicas varían en eficiencia y optimización de la información extraída, cabe destacar que estas técnicas son derivadas de la transformación de Fourier estandar [3].

La representación en tiempo-frecuencia provee de un puente entre estas representaciones simultáneamente, así en cualquier instante de tiempo podemos tener información del espectro frecuencial de la señal [17]. Según esto si se representan las señales en tiempo-frecuencia, entonces podemos obtener información de cómo evoluciona o cambia el espectro de una señal a través del tiempo, de esta forma se puede extraer información, analizar e interpretar una infinidad de procesos físicos y/o biofísicos que se dan en forma de señales no estacionarias; también podremos realizar un tratamiento de señales unidimensionales como electrocardiogramas y representarlos en tiempo-frecuencia, obteniendo información de real interés en cualquier intervalo de tiempo. La $FrFT$ ha sido implementada como una función para el análisis de señales en representación tiempo-frecuencia, siendo esta una de las más óptimas para el tratamiento de señales no estacionarias.

Como se dijo anteriormente, existen varias técnicas para el tratamiento de señales representadas en tiempo-frecuencia, la mayoría de estas extensiones de la Transformación de Fourier estándar, las cuales han sido rediseñadas para un tratamiento de

señales más especializados, entre las cuales están:

- Transformación de Tiempo Corto de Fourier (STFT por sus siglas en inglés): Este es el método más clásico para esta clase de representaciones, la $STFT$ fue propuesta por Gabor en 1946. La idea básica de este algoritmo es la introducción de una especie de ventaneo a lo largo y ancho de la señal [18], los cuales son trasladados en puntos equidistantes durante estos espacios, esta transformación se calcula con la siguiente ecuación:

$$G(\tau, y) = \int_{-\infty}^{\infty} f(x)g(x-\tau)e^{-2ixy}dx \tag{2.2}$$

  La ecuación 2.2 se puede interpretar como los resultado de la transformación de Fourier estándar para cada punto $G(\tau, y)$, donde y es el espectro frecuencial de la señal $f(x)$ para cada punto τ, siendo $f(x)$ la función de entrada y la variable a representar en su dominio directo, $g(x-\tau)$ la función de ventaneo utilizada en la ecuación, ver figura 3.1, la $STFT$ no puede mejorar el tiempo y la frecuencia debido a que la función de ventana es estática, además carece de varias propiedades de la transformación de Fourier estándar [19].

- Transformación Wavelet (WT por su siglas en inglés): es el Tratamiento de Fourier por medio de pequeñas ondas en un espacio determinado de tiempo llamadas wavelet, ver figura 2.2, la WT es otro rediseño de la FT la cual introduce una ventana dinámica que está dada por la ecuación 2.3:

$$w(t) = \frac{1}{\sqrt{|a|}} w\left(\frac{t-b}{a}\right) \tag{2.3}$$

  donde a es la escala y es b la traslación, esta ecuación presenta un análisis tiempo-frecuencia que entrega información redundante [20].

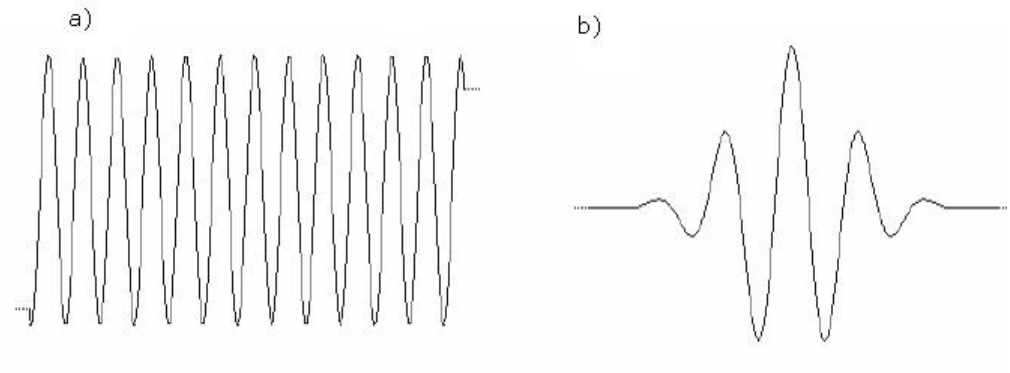

Figura 2.2: a) Onda senoidal b) Wavelet

Recientemente el área de la óptica de Fourier se ha extendido con nuevas contribuciones relativas a transformaciones no convencionales denominadas transformaciones fraccionarias. Una de sus principales herramientas es la Transformación de Fourier fraccionaria ($FrFT$ por sus siglas en inglés) [7]. Este es un nuevo método de representación de señales en tiempo frecuencia, con el cual se han llenado muchos vacíos en el procesamiento de señales no estacionarias que se venían dando con los métodos convencionales antes expuestos, con esta técnica podemos obtener un tratamiento de señales no estacionarias que provee de información verídica y de interés para interpretaciones y toma de decisiones acerca de estas, entre sus utilidades está el cambio y recuperación de la fase en un inetervalo de tiempo dado [21].

Se ha propuesto, por ejemplo, la $FrFT$ para filtrado espacialmente variante, reconocimiento de caracteres, encriptado, marca de agua, implementación de redes neuronales, solución de ecuaciones diferenciales, entre otras [8], las notaciones de esta función (según se interprete mejor para el caso) se pueden escribir de las sigu-

ientes maneras:

$$f_\alpha = \mathfrak{F}_\alpha[f(x)] = \mathfrak{F}_\alpha[f]. \tag{2.4}$$

Donde a la función de entrada $f(x)$ es realizada una transformación de Fourier fraccionaria $\mathfrak{F}_\alpha$ con un angulo $\alpha = a\pi/2$, a es el orden de fraccionalización de la operación.

La $FrFT$ tiene aproximadamente 6 definiciones todas ellas muy parecidas, ya que sus autores la han adaptado a sus conveniencias para diferentes áreas y clases de investigaciones, sin embargo la que mejor se adapta al procesamiento de señales representadas en tiempo-frecuencia es la propuesta por Victor Namias [4], este autor propone que la $FrFT$ de una función $f(x)$ está dada por la siguiente expresión:

$$f_a(y) = C_\alpha e^{i\pi y^2 \cot\alpha} \int f(x) e^{i\pi x^2 \cot\alpha} e^{-i\frac{2\pi}{\sin\alpha}xy} dx. \tag{2.5}$$

a es un número real (en otras definiciones puede tener un valor complejo), que determina el orden fraccionario de la transformación y C_α está dada por la siguiente función:

$$C_\alpha = \frac{e^{i(s(\alpha)\frac{\pi}{4} - \frac{\alpha}{2})}}{\sqrt{|\sin\alpha|}} \tag{2.6}$$

$s(\alpha)$ representa una función sign.

La $FrFT$ se ha establecido hoy en día como una de las herramientas más poderosas y de mayor uso para el tratamiento de señales representadas en tiempo-frecuencia, ya que tiene la facilidad de extraer información veraz del espectro [5] en cualquier instante de tiempo [21]. Además de esto ha traído consigo un óptimo tratamiento de diferentes clases de señales (unidimensionales y multidimensionales; continuas y discretas; y periódicas y no periódicas) [22].

La $FrFT$ ha sido implementada en filtros de restauración y mejora de señales [44], reconocimiento de patrones como huellas dactilares y una perfecta optimización de filtros adaptativos como el filtro de Wiener [23], esto último se basa en la posibilidad de la fraccionalización de la transformación en un número a, el cual es utilizado en el filtro de Wiener fraccionario para minimizar el error cuadrático medio (Mean Square Error MSE por su siglas en inglés) [24]; el $MMSE$ al hacer mínimo el error, y tender a 0, menor es la probabilidad de equivocación al momento de hacer una comparación entre señales, se conoce como $MMSE$ (Minimum Mean Square Error) el error cuadrático medio minimo.

La restauración de señales se debe a que al ser captada una señal, se presenta cierto ruido el cual necesitamos manejar para una mejor calidad de estas, este hecho lo podemos definir matemáticamente de la siguiente manera:

$$s(x) = e(x) + r(x) \tag{2.7}$$

— donde $s(x)$ es la señal; $e(x)$ es la escena y $r(x)$ es el ruido presente [1, 2].

El filtro de Wiener se utilizan comúnmente en la restauración de señales representadas en tiempo-frecuencia [8]. Un filtro de Wiener se diseña de tal manera que su salida sea lo más parecidamente posible a la señal de referencia para ello se considera que la señal y el ruido son procesos aleatorios [25]. El parecido entre la señal de referencia y la restaurada se puede comparar usando el error cuadrático medio [24].

El filtro de Wiener estándar presenta deficiencias en el dominio de Fourier fraccionario [20, 24], así como también el proceso de muestreo de Shannon que se venía

implementando en el tratamiento de señales [3]. Bajo estas deficiencias y/o necesidades surgieron el filtro de Wiener fraccionario [11] y el teorema de muestreo en dominios de Fourier fraccionarios [8]. Juntando estas tres herramientas hace que podamos extraer información de las señales y realizar un óptimo tratamientos de señales, según lo anterior se formula el siguiente interrogante ¿Por qué se hace necesario la implementación computacional de la Transformación de Fourier fraccionaria en el tratamiento de señales y cuales serían los beneficios de esta?

El desarrollo e implementación de este proyecto es realizado sobre Scilab, esta plataforma se puede encontrar sobre Internet, así como también, muchas contribuciones o las llamadas Toolbox, entre sus ventajas frente a otras se puede mencionar que es una plataforma gratis, de código abierto, presenta un lenguaje de programación de alto nivel para cálculo científico, disponible en múltiples sistemas operativos ($Unix$, $GNU/Linux$, $Windows$, $Solaris$, $Alpha$); y que todos los procesos que se generan para realizar las diferentes operaciones son ejecutados paralelamente, es decir, que a través de esta herramienta se podría llegar a realizar un ambiente totalmente distribuido utilizando las librerías e interfaces de diversos lenguajes de programación tales como $java$, C, $Fortran$ y diferentes sistemas operativos [26, 27], hasta llegar al caso donde un grupo de computadores puedan compartir sus dispositivos de hardware y/o software y realizar todos sus procesos en paralelo.

Y ya que Scilab es una herramienta de software libre el usuario puede personalizarlo o ampliarlo añadiendo sus propias funciones, y como la transformación de Fourier fraccionaria no existe en esta plataforma se propone su desarrollo e implementación en las funciones necesarias para hallar el filtro de Wiener fraccionario para el proce-

samiento de señales. El desarrollo de la $FrFT$ será de gran ayuda para el estudio y análisis de señales de diferentes señales de biofísica, electrofísica, geofísica, acústicas, de voz, fisiológicas, entre otras.

Capítulo 3

Antecedentes

Podemos definir señal como toda función $f(x)$ que contiene información que varía en el tiempo, en el espacio ó en ambos simultáneamente y la Transformada $F(y)$ es la relación de la amplitud que obtenemos de una variable que se esté analizando a lo largo del tiempo ó su dominio directo [13]. No existe alguna duda que la madre de todas las Transformaciones de señales es la Transformación de Fourier [2], la cual se denota como $\mathfrak{F}[f](x)$ donde x es el dominio directo de la señal a analizar y f la función de entrada a la Transformación, la ecuación de esta operación está dada en la ecuación 2.1.

3.1. Transformación de Fourier discreta

La transformación de Fourier discreta vista en muchas partes como DFT Discrete Fourier Transform por su siglas en inglés, es una función ampliamente empleada para el tratamiento de señales y campos a fines, con el fin de analizar las frecuencias que tienen un número de muestras de una señal o función, esta función puede calcularse eficientemente desde la Transformación rápida de Fourier FFT explicada

en la sección 3.1.1, la ecuación 2.1 se discretiza de la siguiente manera:

$$x = n\Delta x \qquad\qquad y = k\Delta y$$

$$\mathfrak{F}(k) = \sum_{n=N/2}^{n=N/2-1} f(n)e^{-2i\pi n\Delta x k\Delta y} \tag{3.1}$$

Los valores con Δ se refiere al espacio que debe haber entre cada una de las muestras con el fin de que la señal pueda ser reconstruida perfectamente, por lo cual según el teorema de Shannon si ξ es la duración de la señal el cual debe ser equivalente al tamaño de la señal, y ζ es el ancho de banda de la transformada, ver figura 7.1, entonces:

$$\Delta x = \frac{\xi}{N} \qquad\qquad \Delta y = \frac{\zeta}{N} \qquad\qquad \xi\zeta = N$$

con base en lo anterior la ecuación 3.1, la transformación de Fourier discreta queda de la siguiente forma:

$$\mathfrak{F}(k) = \sum_{N/2}^{N/2-1} f(n)e^{\frac{-2i\pi nk}{N}} \tag{3.2}$$

3.1.1. Transformación rápida de Fourier

EL algoritmo de la FFT lo que busca es hacer una aproximación lo más exacta de la DFT pues como puede verse en la ecuación 3.2 el factor de fase de la función hace que se tenga que hacer N^2 multiplcaciones entre n y k para hallar el resultado. El algoritmo de la FFT hace que se encuentren valores muy cercanos a la transformación de Fourier real, en $N * log_2(N)$ operaciones, en este algoritmo se encuentra una forma por medio de división modular para que no se repitan valores en el factor de fase de la ecuación 3.2, y así reducir el tiempo de ejecución de esta operación.

N	DFT	FFT	Mejora
2	16	4	4
4	256	32	8
8	4096	192	21.33
16	$6{,}554 * 10^4$	1024	64
32	$1{,}049 * 10^6$	5120	201.8
64	$1{,}678 * 10^7$	$2{,}458 * 10^4$	682.67
128	$2{,}684 * 10^8$	$1{,}147 * 10^5$	2341
256	$4{,}295 * 10^9$	$5{,}243 * 10^5$	8192
512	$6{,}872 * 10^{10}$	$2{,}359 * 10^6$	$2{,}913 * 10^4$
1024	$1{,}100 * 10^{12}$	$1{,}049 * 10^7$	$1{,}049 * 10^5$

Tabla 3.1: Comparación del número de operaciones entre la DFT y el algoritmo de la FFT de señales Bidimensionales cuadradas

3.2. $FrFT$ y otras representaciones de señales

A pesar de la gran utilidad de la Transformación de Fourier en el tratamiento de señales, esta presenta ciertas limitaciones en la obtención de la información de la señal de entrada, ya que al aplicarla no se puede saber si en las frecuencias que se obtuvieron en $F(y)$ aparece un valor determinado de la función $f(x)$, se puede saber si la frecuencia existe pero no el espacio en que el evento ocurre [14].

En la naturaleza hay muchas señales no estacionarias, en las cuales la amplitud y frecuencia varían en el tiempo. Así por ejemplo, en la voz, las frecuencias varían mucho si el tiempo es bastante grande y por tanto un análisis en el dominio de la frecuencia sin tener en cuenta el tiempo es muy ineficiente para este tipo de señales [16]. Por lo tanto la transformación de Fourier estándar no es una herramienta idónea para el análisis de señales no estacionarias [3]. Esta limitante hace que no se pueda obtener suficiente información de la función de entrada, por lo cual surgieron algunas adaptaciones para que esta función pueda ser representada en tiempo-frecuencia

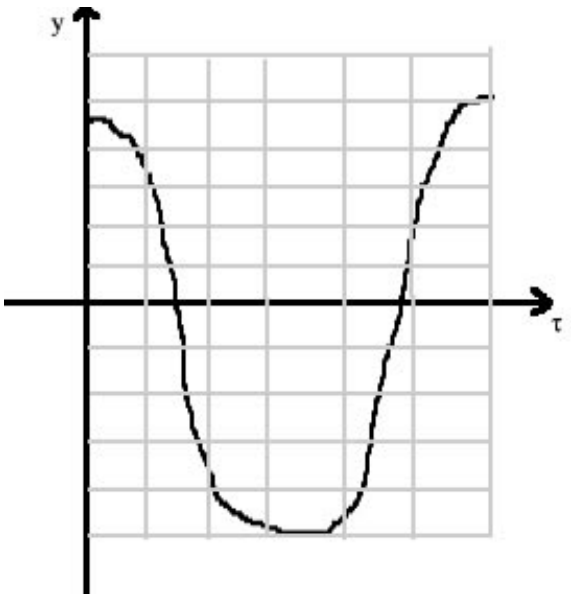

Figura 3.1: La $STFT$ obtiene el espectro frecuencial por cada punto de esta ventana Rectangular

[17, 5].

La Transformación de Fourier de tiempo corto $STFT$ por sus siglas en inglés, es la primera en introducir el concepto de ventaneo en la función a analizar, con el fin de hacer el tratamiento en sectores separados periodicamente y así obtener el valor de la Transformación de Fourier en cada uno de estos sectores, así se dividiría el resultado y se podrá saber el espectro de frecuencia en cada uno de estos puntos llamados por otros investigadores átomos de la ventana [19], Véase figura 3.1, dependiendo de la función de ventana que se selecione quedará la operación matemática, sin embargo continúan las mismas limitaciones que se mencionan anteriormente con la transformación de Fourier estándar.

Es decir, si la función a tratar es $f(x)$ y la ventana a utilizarse depende de la función $g(x)$ con un factor de traslación τ, entonces cada punto del espectro de

la frecuencia está dado por la ecuación 2.2 al ojo puede verse si se tienen ciertos conocimientos matemáticos que esta operación obtiene también la forma de una convolución por lo tanto se mantienen ciertas propiedades que se mencionan con estas operaciones.

Las técnicas de adaptación de la Transformación de Fourier se volvieron un objetivo para investigadores, principalmente por la necesidad de encontrar una representación de señales de todas las clases, entre ellas las no estacionarias las cuales son aquellas que pueden ser generadas aleatoriamente como las sísmicas, geofísica, biofísica, entre otras [18].

Una de las más grandes utilidades en la busqueda de representaciones de señales en tiempo frecuencia, es la descomposición de operaciones en familias de funciones, lo anterior sumado al basto procesamiento de los computadores modernos ha hecho posible nuevos descubrimientos en el tratamiento de señales digitales [20].

Aproximadamente en el año de 1985 un Francés de apellido *Meyer* inventó la Transformación Wavelet u ondeletas, la cual a su vez lleva su nombre como Wavelet de Meyer, este es un tratamiento realizado con base en pequeñas ondas que varían su amplitud periodicamente en el tiempo llamadas wavelets, de ahí proviene su nombre, véase figura 2.2.

Otra representación en tiempo frecuencia es la Distribución de Wigner [6], esta nos permite visualizar el soporte de una señal $f(x)$, cuya Transformada de Fourier es $F(y)$, es decir, con esta asociamos un valor de amplitud en cada punto $W(x, y)$

asociados a un una función $f(x)$ y a su transformada $F(y)$, esta operación se obtiene con la siguiente ecuación 3.3:

$$\mathcal{W}_f(x,y) = \int_{-\infty}^{\infty} f\left(x + \frac{x'}{2}\right) \overline{f\left(x - \frac{x'}{2}\right)} e^{-2\pi i x' y} dx' \tag{3.3}$$

donde x es la variable a evaluar de la función de entrada y x' es otra variable que está en el dominio de las x. Como puede observarse en la ecuación 3.3 el resultado de la distribución de wigner es una función bidimensional asociada a una función f. La gráfica de una distribución de Wigner de una función $f(x)$ puede obtenerse como se muestra en la figura 3.2, sin embargo en la mayoría de los casos es visualizada en forma de imagen. En esta figura la sección de cuadricula azul en los ejes x e y es el soporte de la distribución de Wigner, lo cuales tienen un valor de amplitud cada uno asociado a la función $f(x)$ en cada punto (x, y) de la ecuación 3.3, esta cuadricula se llama comunmente soporte compacto de la señal.

Ahora si se integra el resultado de la distribución de Wigner en el dominio di-

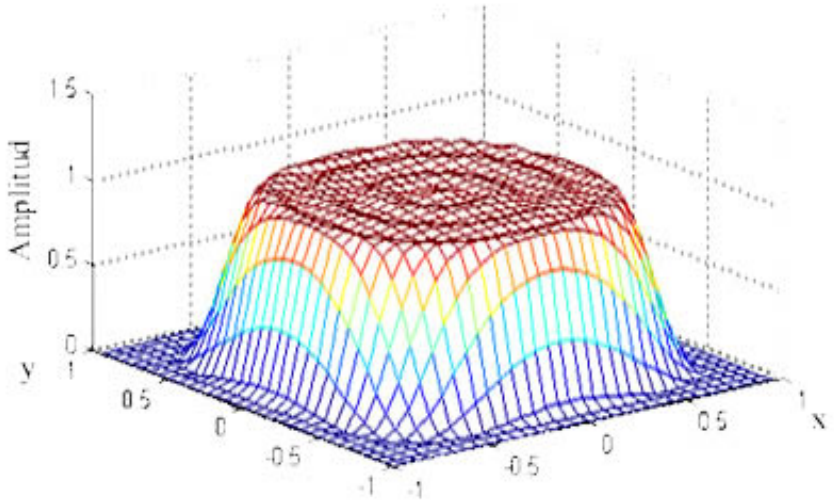

Figura 3.2: Distribución de Wigner asociada a una señal

recto de la señal y en el dominio de las frecuencias, se obtienen las expresiones

a	α =a $\pi/2$	Operador fraccionario	Operación sobre la señal
0 ó 4	0 ó 2π	$F^0 = F^4 = I$	Operador Identidad
1	$\pi/2$	$F^1 = F$	FT
2	π	$F^2 = FF = P$	Operador Paridad
3	$3\pi/2$	$F^3 = FF^2 = F^{-1}$	FT inversa

Tabla 3.2: Casos Particulares de la $FrFT$, según el valor del operador a

que se muestran en la ecuación 3.4, 3.5 y 3.6 las cuales son el módulo cuadrado de $f(x)$, el módulo cuadrado de $F(y)$ y el módulo cuadrado en el dominio de Fourier fraccionario dado por el angulo α.

$$|f\,(x)\,|^2 = \int_{-\infty}^{\infty} \mathcal{W}_f\,(x,y)\,dy \tag{3.4}$$

$$|F\,(y)\,|^2 = \int_{-\infty}^{\infty} \mathcal{W}_f\,(x,y)\,dx \tag{3.5}$$

$$|f_\alpha\,(x_\alpha)\,|^2 = \int_{-\infty}^{\infty} \mathcal{W}_f(x\cos\alpha - y\,\mathrm{sen}\,\alpha, x\,\mathrm{sen}\,\alpha + y\cos\alpha)dy \tag{3.6}$$

Esta transformación presenta 4 casos particulares los cuales dependen del grado de fraccionalización de la Transformación, ver tabla 3.2, los algoritmos que desarrollen esta operación de la $FrFT$ deben cumplir con estas condiciones.

Existen otras operaciones que se realizan en el dominio directo como lo es la convolución y la correlación [14], estas se aplican con el fin de encontrar la magnitud de la relación entre dos señales, la primera está dada por la siguiente integral

$$[f * g](x) = \int_{-\infty}^{\infty} f(y)g(x-y)dy \tag{3.7}$$

donde $f(x)$ es la señal a convolucionar con la función $g(x)$, y $x = x - y$ con el fin de desplazar la función y así al ser esta móvida se encontrará la relación de la

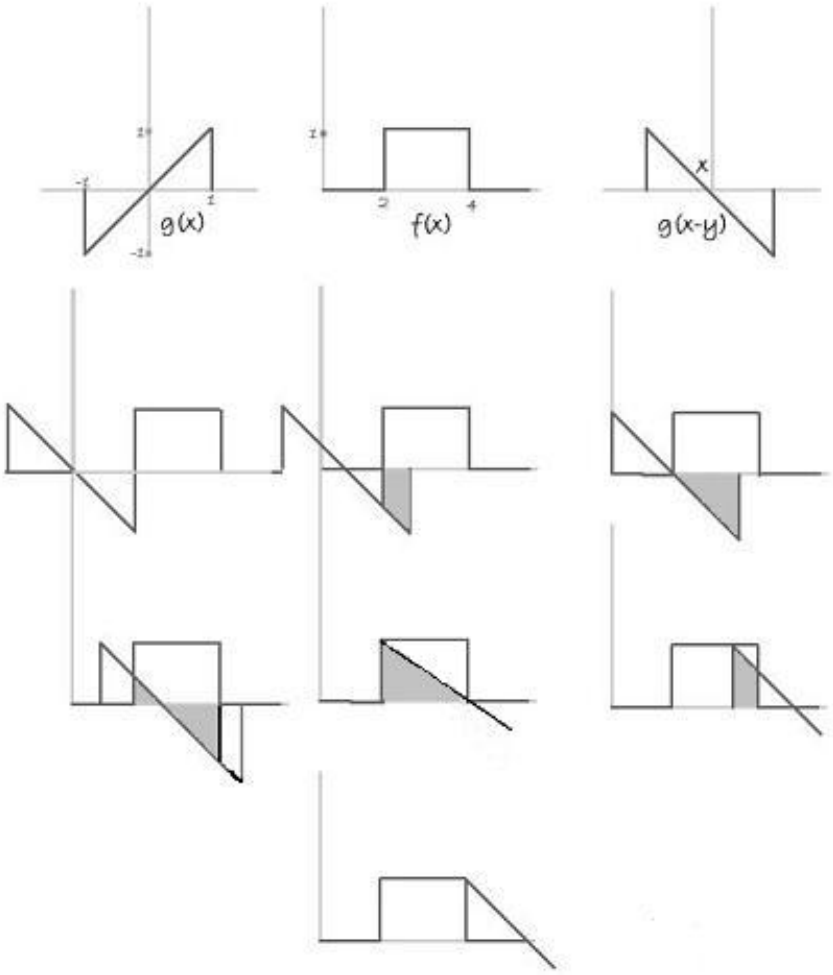

Figura 3.3: Proceso de convolución entre dos funciones

amplitud en todos los puntos de las dos señales, ver figura 3.3. En resúmen la figura 3.3 muestra la convolución entre la función $f(x)$ y $g(x)$, donde $g(x)$ se debe reflejar en su dominio directo e irla tarsladando en un valor y, con el fin de hallar la relación de la superpocisión de las dos funciones.

Teorema 3.1. Teorema de la Convolución Al aplicar la Transformación de Fourier a una convolución, se obtiene el producto de las dos transformadas de las funciones relacionadas en la convolución.

El teorema anterior se puede representar matemáticamente por la siguiente ecuación.

$$\mathfrak{F}[f * g](y) = F(y)G(y) \tag{3.8}$$

Si se aplica en ambos lados de la ecuación una transformación de Fourier inversa, hayaremos la convolución de las dos funciones originales, este proceso es el que se conoce como convólución rápida y se describe matemáticamente con la siguiente ecuación 3.9:

$$[f * g](y) = \mathfrak{F}^{-1}[F(y)G(y)] \tag{3.9}$$

Según el articulo [41] la ecuación de la transformación de Fourier fraccionaria puede verse sometida a un cambio algebraico y trigonométrico, con el fin de que esta pueda resolverse por medio de una convolución, partiendo desde la Ecuación 2.5, se ingresan todos los términos con exponenciales y al agrupar los terminos similares queda:

$$f_\alpha(y) = C_\alpha \int_\infty^\infty f(x) e^{i\pi(y^2 \cot\alpha + x^2 \cot\alpha - 2xy \csc\alpha)} dx \tag{3.10}$$

Luego se suman la siguientes expresiones, con el fin de completar cuadrados perfectos,

$$x^2 \csc\alpha - x^2 \csc\alpha + y^2 \csc\alpha - y^2 \csc\alpha = 0$$

y la ecuación 3.10 queda de la siguiente manera:

$$f_\alpha(y) = C_\alpha \int_\infty^\infty f(x) e^{i\pi(y^2(\cot\alpha - \csc\alpha) + x^2(\cot\alpha - \csc\alpha) + (x-y)^2 \csc\alpha)} dx \tag{3.11}$$

después de la comprobación de que $\cot\alpha - \csc\alpha = \tan\alpha/2$ se reemplaza esto en la ecuación 3.11 y finalmente se puede observar la ecuación creada por Namías, como un producto de una función Chirp por una convolución [29], tal como se muestra en la ecuación 3.12.

3.3. Descripción del algoritmo $fracF$

En esta investigación se analizó el código de la $FrFT$ descrito por el articulo [41], este se conoce comúnmente como $fracF$ explicado en la sección 3.3.1 página 22 de este documento, este código se puede hallar en el sitio Web [40].

3.3.1. Función $fracF$

`function [res]=fracF(fc,a)`, los parámetros de entrada de la función son fc y a, donde fc es la señal o muestras de la señal y a el $a - esimo$ valor a fraccionar la Transformación, la salida res, será la Transformada de Fourier fraccionaria de fc.

```
N = length(fc);
```

Se halla el tamaño de la señal, el cual debe ser impar como condición principal de este algoritmo.

```
if fix(N/2)  = N/2
   error('Length of the input vector should be even');
end;
```

En las plataformas de cálculo numérico es importante evitar los bucles for para

el uso de operaciones sobre elementos de un vector, en este caso la programación es ineficiente debido a que los bucles for son interpretados y esto hace lento el proceso, este bucle se debe utilizar como última opción, no con motivos de cálculo, sino en operaciones de control solamente, véase figura 3.4.

En estas plataformas de cálculo las operaciones matriciales son muy frecuentes por tal razón es recomendable utilizarlas en lugar del uso de bucles. El simbolo (:) ayuda a utilizar y comprender estas operaciones y es uno de los más utilizados, ya que ahorra el uso de bucles en la de espacios vectoriales de diferentes dimensiones.

En la siguiente sentencia, `fc = fc(:)` se ordena los valores de fc de tal forma

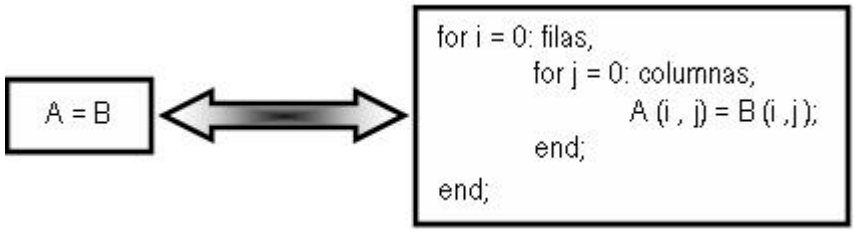

Figura 3.4: Operaciones matriciales en plataformas de cálculo

que todos formen una sola columna, representando todos sus elementos en una sola columna y se interpola la señal fc, por medio de la función `bizinter(fc)`, la salida de esta operación será la función interpolada a un tamaño $2N$.

```
fc = fc(:);
fc = bizinter(fc);
fc = [zeros(N,1); fc ; zeros(N,1)];
```

La siguiente parte es una sección de condicionales para mantener la potencia a en un intervalo con el fin de aplicar el kernel de este algoritmo. Se puede observar que

entre ciertos intervalos de a la variable `flag` toma valores entre 1 y 4 para después ser utilizada como condicional para realizar la función `corefrmod2(fc,a)` explicado en la sección 3.3.2 página 25.
Si a está entre 0 y 0,5 entonces $flag = 1$ y se hace $a = a - 1$
Si a está entre $-0{,}5$ y 0 entonces $flag = 2$ y se hace $a = a + 1$
Si a está entre 1,5 y 2 entonces $flag = 3$ y se hace $a = a - 1$
Si a está entre -2 y $-1{,}5$ entonces $flag = 4$ y se hace $a = a + 1$

Luego teniendo el valor de $flag$ y la potencia de a podemos preguntar por sus casos especiales, véase figura 3.2, para realizar la transformación de Fourier estándar, la inversa, el operador identidad y el de reflección según los casos que le corresponde a cada uno.

Si $flag = 1$ ó $flag = 3$ entonces realiza el núcleo con $a = 1$, es decir que el angulo tomaría un valor de $\alpha = \pi/2$.
Si $flag = 2$ ó $flag = 4$ entonces realiza la función `corefrmod2(fc,-1)`, es decir que el angulo tomaría un valor de $\alpha = -\pi/2$.
Por último si $a = 0$ entonces realiza el operador Identidad de la función de entrada fc.
Si $flag = 2$ ó $flag = -2$ entonces realiza el núcleo con $a = 1$, es decir que el angulo tomaría un valor de $\alpha = \pi/2$
De lo contrario se realiza la función `corefrmod2(res,a)` en el número fraccionario a

```
res = res(N+1:3*N);
res = bizdec(res);
```

```
res(1) = 2*res(1);
```

Como los datos que vienen tienen un tamaño de $4N$ se escogen los datos del centro, luego se decima por medio de la función `bizdec()`, y por último se multiplica por 2 el dato de la posición 1 de la transformada como caracteristica especial de este algoritmo.

3.3.2. función corefrmod2(fc,a)

Esta es la función de mayor interés en el algoritmo, ya que es la que desarrolla la operación de Kernel de la ecuación 2.5, $fracF$ se basa en una implementación computacional del teorema de la convolución, llamado convolución rápida y por medio de algunas operaciones algebragicas y trigonometricas, llevan la ecuación 2.5 a la forma de una convolución con la siguiente expresión:

$$F_\alpha(y) = C_\alpha e^{-i\pi \tan(\alpha/2)y^2} \int_{-\infty}^{\infty} e^{i\pi \csc\alpha(y-x)^2} [e^{-i\pi \tan(\alpha/2)x^2} f(x)]dx \quad . \tag{3.12}$$

Con base a la ecuación 3.12 hacen uso del teorema de la Convolución e implementan un algoritmo de convolución rápida, esto se explica en el Capitulo 7 en la página 40.

Los parámetros de entrada de la función son fc y a donde fc es el número de muestras de una función y a el valor con el que se evaluará el kernel o nucleo de la $FrFT$.

En la siguiente sentencia se observa la creación de una función Chirp $e^{-i\pi \tan(\alpha/2)x^2}$, y luego se multiplica por las muestras de la señal de entrada.

```
f1 = exp(-i*pi*tan(phi/2)*x.*x); f1 = f1(:);
```

```
fc = fc.*f1;
```

Posteriormente se crea otra función Chirp $e^{i\pi \cot \alpha y^2}$, luego se hace la variable `hlptcz` la variable `hlptc` recientemente creada para crear la segunda función chirp y se agregan ceros a la derecha de la función, lo mismo se hace con el resultado de la operación anterior, con el fin de convolucionar y de evitar la superposición entre las dos funciones.

```
hlptc =exp(i*pi*beta*t.*t);
hlptc = hlptc(:);
N2 = length(hlptc);
N3 = 2^(ceil(log(N2+N-1)/log(2)));
hlptcz = [hlptc;zeros(N3-N2,1)];
fcz = [fc;zeros(N3-N,1)];
Hcfft = ifft(fft(fcz).*fft(hlptcz));
```

Luego se crea la variable *res* al multiplicar el resultado de la convolución rápida, con la primera función chirp creada y la constante C_a, obteniendo así el resultado de la operación de la la transformación de Fourier fraccionaria por medio del algoritmo de convolución rápida mencionado en [41].

```
Hc = Hcfft(N:2*N-1);
Aphi=exp(-i*(pi*sign(sin(phi))/4-phi/2))/sqrt(abs(sin(phi)));
xx = [-ceil(N/2):fix(N/2)-1]/deltax1;
f1 = f1(:);
res = (Aphi*f1.*Hc)/deltax1;
```

Capítulo 4

Marco Teórico

Jean-Baptiste-Joseph Fourier matemático y físico francés, ver figura 4.1 conocido por sus trabajos sobre la descomposición de funciones periódicas en series trigonométricas convergentes llamadas series de Fourier, estás series están dadas de la siguiente forma:

$$y(x) = \frac{a_0}{2} + \sum_{n=1}^{\infty}[a_n \cos(nx) + b_n \sin(nx)] \tag{4.1}$$

donde a_0, a_n y b_n se denominan coeficientes de Fourier en las series de Fourier de la función $f(x)$, en términos teóricos la investigación de Fourier dice que toda función que se repite periódicamente puede ser expresada como la suma de senos y/o cósenos de diferentes frecuencias, cada uno multiplicado por un coeficiente diferente [2]. Más adelante se definió que aún funciones que no son periódicas, pero con un área finita bajo la curva, es decir, absolutamente integrables, pueden ser expresadas como la integral de senos y/o cósenos multiplicada por una función de ponderación o factor de fase, esta es la transformación de Fourier, la cual está dada en la ecuación 2.1, y su utilidad es aún más grande que la de las series de Fourier en muchos problemas prácticos [13].

Figura 4.1: Jean Baptiste Fourier, (1768-1830)

Estas teorías son utilizadas comúnmente para el tratamiento de señales y su implementación en esta área se llama normalmente óptica de Fourier, luego fue creado el teorema de muestreo de Shannon [15] y la transformación rápida de Fourier, lo cual permitió que el tratamiento de señales fuera realizado digitalmente sobre computadoras. Esta área ha estado sujeta a cambios debido a algunas señales que necesitan un tratamiento más especializado, para ello hay que hablar acerca de la principal división natural de las señales, estas son las estacionarias y no estacionarias.

Las señales estacionarias son constantes en sus parámetros estadísticos, es decir, si se observa una señal estacionaria, durante un instante de tiempo y después de una hora se le vuelve a observar, esencialmente se vería igual o muy parecida, en la figura 4.2 se puede ver una señal digital estacionaria.

Las señales estacionarias pueden ser representadas en el dominio del tiempo y pueden ser tratadas por la transformación de Fourier estándar, pero las señales no estacionarias no deben ser analizadas con esta transformación y deben ser representadas en tiempo-frecuencia, así por ejemplo, como puede observarse en la figura

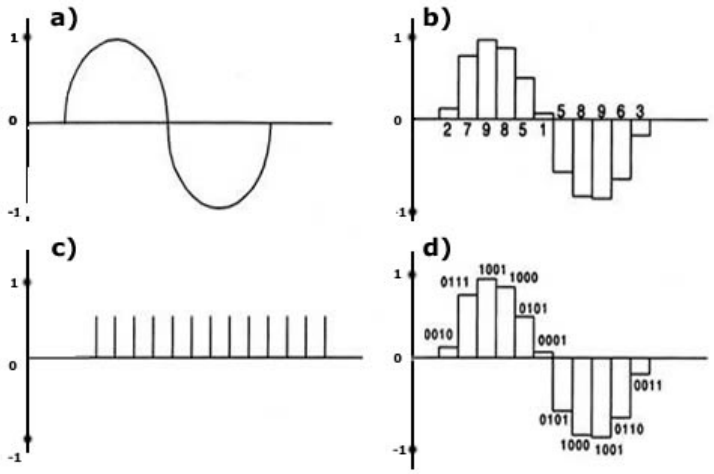

Figura 4.2: *a*) Señal digital, *b*) Señal digitalizada *c*) Reloj de muestreo *d*) Señal binaria cuantizada

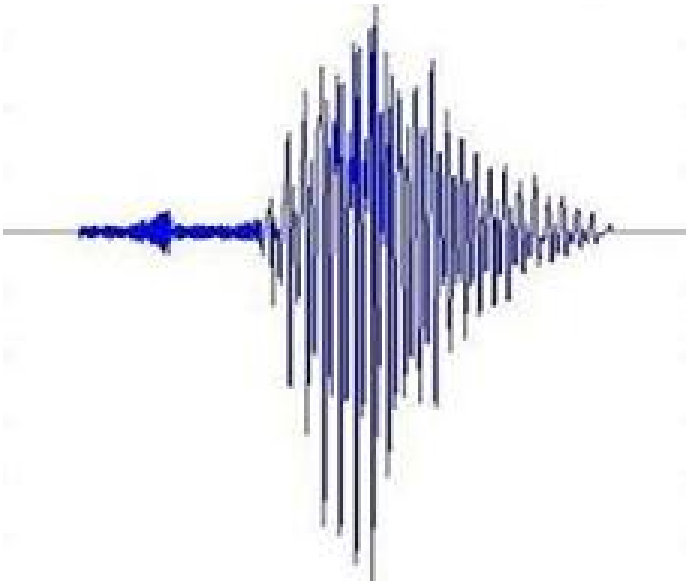

Figura 4.3: Señal no estacionaria

4.3, en una señal de audio, la frecuencia y amplitud de este varían mucho en un intervalo de tiempo módico y por lo tanto un análisis periódico en el dominio del tiempo sin tener en cuenta la frecuencia es muy ineficiente para este tipo de señales; la necesidad de una representación de señales en general ha sido un vacío que se ha ido cerrando con la creación de nuevos modelos matemáticos, la transformación de Fourier fraccionaria [5] es uno de los más recientes, la cual ha sido de gran utilidad para el tratamiento de señales de muchos tipos, trayendo un gran avance en la óptica de Fourier, generando con ello un nuevo paradigma de fracccionalización en el tratamiento de señales llamado óptica de Fourier fraccionaria.

El término de transformación de Fourier fraccionaria aparece por primera vez en 1929 en un artículo de Norbert Wiener [24], desde ese momento surgieron muchas aplicaciones de esta operación, en solución de ecuaciones diferenciales, mecánica y óptica quántica, teoría de difracción óptica y diferentes sistemas ópticos; debido a cierta complejidad en las relaciones entre sus variables esta transformación no podía utilizarse para el tratamiento de señales, la física tuvo que esperar hasta 1980, año en que Victor Namías [4] propuso una nueva definición. Inmediatamente la $FrFT$ se mostró como una herramienta útil, y los investigadores se interesaron por encontrar todas las propiedades, operaciones y teoremas que permitan el empleo de ella en un amplio rango de campos de las ciencias y las ingenierías [20]. Y casi simultaneamente en 1993 un grupo de investigadores Almeida, Mendlovic, Lohmann, Ozaktas, "reinventaron"la $FrFT$ [5], algunos la interpretaron como una rotación del plano tiempo-frecuencia, véase figura 4.4. Gracias a esta interpretación en la actualidad la Transformación de Fourier fraccionaria es implementada en el tratamiento de señales, y ha suplido muchas falencias de la Transformación de Fourier ordinaria

siendo la operación más utilizada para el tratamiento de señales dada sus ventajas sobre el tratamiento de señales no estacionarias [21]. El orden fraccionario de la f_a ha sido interpretado como la rotación de la distribución de Wigner que se encuentra asociada a la señal en un ángulo $a\pi/2$ en el tiempo-frecuencia. De ahí la gran relación que tiene esta transformación con la distribución de Wigner [23]. La Distribución de Wigner es una función que extrae la energía de una señal representada en tiempo-frecuencia [6]. La distribución de Wigner de una función $f(x)$ está definida según la ecuación 3.3 [23].

La $FrFT$ es una transformación lineal que generaliza la Trasformación de Fourier estándar [5]. entre las propiedades encontradas están:

- **Propiedad de Linealidad:** Esta propiedad dice que la $FrFT$ de una combinación lineal de dos funciones $f(x)$ y $g(x)$ de entrada se comportan de acuerdo a la definición de sistemas lineales. c_1 y c_2 son constantes.

$$F_\alpha[c_1 f(x) + c_2 g(x)] = c_1 F_\alpha[f(x)] + c_2 F_\alpha[g(x)]. \tag{4.2}$$

- **Teorema de Parseval ó Propiedad de la conservación de la energía:** Este teorema nos dice que la energía de una señal, calculada en el dominio del tiempo es igual a la calculada en el dominio de la frecuencia.

$$\int_{\Re} f(x) g*(x) dx = \int_{\Re} f_\alpha(y) g *_\alpha (y) dy. \tag{4.3}$$

- **Teorema del Corrimiento o propiedad de traslación en tiempo:** Este Teorema dice que si se traslada una función $f(x)$ en el dominio del tiempo, lo único que se altera es la distribución de fase vs. frecuencia. Siendo ς la variable de corrimiento.

$$F^a[f(x-\varsigma)] = f_a(y - \varsigma \cot\alpha) e^{i\pi \sin\alpha(\varsigma^2 \cos\alpha - 2y\varsigma)}. \tag{4.4}$$

- **Teorema de la Modulación o propiedad de cambio de la frecuencia:** Este teorema es de gran importancia para el tratamiento de la información en cualquiera de sus magnitudes, ya que esta permite hacer un cambio de la señal de entrada en su frecuencia. La propiedad de modulación es muy parecida a la de corrimiento en el tiempo. Siendo δ la variable de Modulación en la siguiente ecuación:

$$F^a[f(x)e^{i2\pi\delta x}] = f_a(y - \delta \sin\alpha)e^{-}i\pi \cos\alpha(\delta^2 \sin\alpha - 2\delta y). \tag{4.5}$$

- **Teorema del Escalamiento:** La importancia conceptual de este teorema radica en la relación tiempo-frecuencia, este dice que si se tiene una función $f(x)$ para expandir en tiempo la función obligatoriamente se debe comprimir la frecuencia de la misma; Intuitivamente esto es lógico ya que comprimir una señal en tiempo equivale a hacer sus cambios más bruscos, por lo tanto tienen que aparecer componentes de mayor frecuencia.

$$F^a[f(cx)] = \sqrt{\cos\beta/\cos\alpha}e^{\frac{1}{2}i(\alpha-\beta)}e^{i\pi y^2 \cot\alpha(1-\frac{\cos^2\beta}{\cos^2\alpha})}f_\beta(y\frac{\sin\beta}{c\sin\alpha}), \tag{4.6}$$

$$\text{donde } \tan\beta = c^2\tan\alpha.$$

El cálculo matemático de esta operación realizado en los computadores modernos hace de la transformación de Fourier fraccionaria una herramienta de gran importancia y utilidad para aplicaciones en el tratamiento de señales en las cuales es manipulada. Varios algoritmos han sido descritos, sin embargo estos algoritmos pueden ser clasificados de dos tipos: transformación rápida de Fourier fraccionaria [47] y transformación discreta de Fourier fraccionaria [38], los primeros son generados utilizando las funciones de la transformación rápida de Fourier y operaciones de convolución.

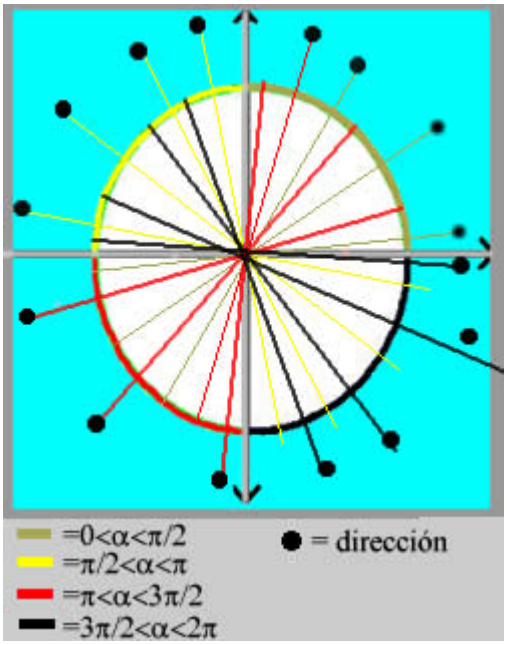

Figura 4.4: Rotación en el plano tiempo-frecuencia

El nuevo modelo matemático de la transformación de Fourier fraccionaria y sus aplicaciones para la recuperación de la fase, caracterización de rayos de luz, filtros, encriptación, marca de agua, entre otros [20, 11, 12, 28], ha llevado a que en la última década la rama de la óptica de Fourier halla avanzado en el tratamiento de una cantidad de señales.

Capítulo 5

Justificación

La línea de investigación del procesamiento de señales digitales enmarca tópicos interesantes tales como reconocimiento de voz, visión por computadora, procesamiento digital de señales unidimensionales y/o multidimensionales, estacionarias y no estacionarias, continuas y discretas, en tiempo real, en audio, en video, entre otras [2, 13].

En esta investigación se hace un tratamiento de señales en el dominio de Fourier fraccionario, todo esto por medio del desarrollo de funciones en una aplicaión realizada sobre la plataforma de cálculo numérico Scilab. Este tratamiento puede ser implementado en muchas áreas de investigación [22] y últimamente ha tenido mucho auge en el procesamiento de imágenes digitales [23, 45], gracias a la extracción de información de las señales representadas en tiempo-frecuencia en cualquier instante de tiempo [5] y su mayor ventaja se fundamenta en el tratamiento de señales no estacionarias.

El estudio de señales no estacionarias siempre ha representado mayor preocupación en el campo científico [16], ya que muchos factores en áreas de biofísica, electrofísica, geofísica, entre otras muchas, se generan aleatoriamente a lo largo del tiempo, el re-

diseño de la transformación de Fourier fraccionaria ha generado muchas utilidades y ha sido considerada como una óptima representación tiempo frecuencia al poder hacer una rotación en el espacio de una función por medio de la distribución de Wigner [6], la $FrFT$ trajo consigo nuevas operaciones como la convolución fraccionaria, correlación fraccionaria, teorema de muestreo en dominios de Fourier fraccionarios, filtros, entre otros [20].

Junto con la $FrFTD$ utilizando el teorema de muestreo en dominios fraccionarios para el tratamiento de señales unidimensionales y bidimensionales, se implementa también la distribución de Wigner para señales unidimensionales y un filtro para la restauración de señales llamado filtro de Wiener fraccionario [25, 11], para esto se requierre hacer un estudio detallado de los fundamentos matemáticos de estas operaciones, comprender su universalidad y los límites de su aplicabilidad en el área de la óptica y el tratamiento de señales [36].

5.1. Viabilidad técnica

La elección del software que se usa en una investigación debe ser una decisión fundamental, más aún, cuando la investigación que se realiza está enfocada en áreas de las ciencias de la computación. Para el desarrollo de este proyecto se utiliza la herramienta gratuita Scilab, ver figura 5.1, esta herramienta es un paquete de software de cálculo muy potente en lo que a matrices se refiere ya que fue creado para trabajar con ellas, también posee una extraordinaria versatilidad y capacidad para resolver problemas de matemática aplicada, física, ingeniería, procesamiento de señales y otras muchas aplicaciones. La utilización de esta herramienta produce algunas ventajas, entre ellas:

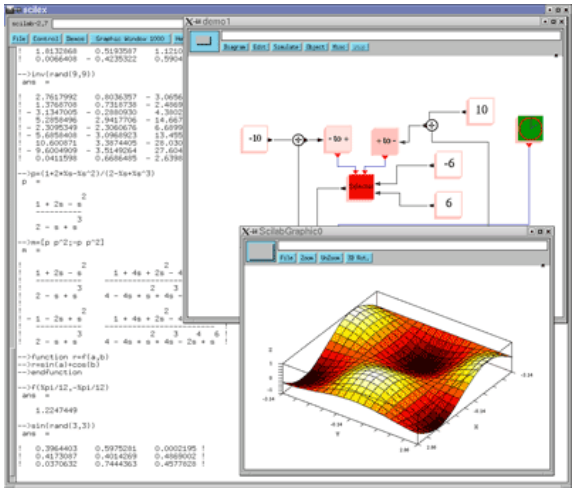

Figura 5.1: Interfaz grafica de Scilab.

- El software hasta ahora es gratis.
- La última versión del software está siempre disponible.
- El acceso al código fuente facilita el aprendizaje de esta herramienta.
- La información y documentación de calidad está siempre disponible.
- Sus procesos los realiza de forma paralela, permitiendo y facilitando la implementación de sistemas distribuidos para tareas que así lo requieran.

El código de la Transformación de Fourier fraccionaria está hecho en la plataforma Matlab, sin embargo, la forma en que es realizada esta función no tiene en cuenta el teorema de muestreo en dominios de Fourier fraccionarios [7] para la discretización del Kernel lo cual hace que no se pueda tener un estudio total y puntual sobre la frecuencia; Scilab tiene funciones que permiten exportar códigos de Matlab [27], sin embargo esta conversión la mayoría de las veces es imperfecta, ya que existen funciones de Matlab como eye, ones, size, sum, entre otras [37], que se comportan de diferente forma según la dimensión de sus argumentos y si Scilab no puede

deducir el código que está exportando, lo que hace es sustituir la función de llamada correspondiente dentro del fichero .m que se está traduciendo, por una función mltb-¡nombre-función¿en Scilab, y no todas las funciones tienen esta función de traducción.

Otro objetivo implícito que adquiere esta investigación es pretender arrojar un poco de luz sobre las caracteríssticas de LaTeX que lo hacen ideal para la escritura de textos extensos sustituyendo a los típicos procesadores gráficos WYSIWYG.

Una de las cosas que más parece útil de LaTeXrespecto a un procesador de texto WYSIWYG (What You See Is What You Get, en inglés, lo que ves es lo que obtienes, en español) es que permite concentrarse en el contenido del documento, más que del diseño y que la belleza aparente del mismo, aspectos que vienen por defecto en LaTeX. Las ideas que se quieren comunicar van a ser los elementos importantes de lo que se necesita decir, cómo estructurar el texto en las unidades que te permitan ir elaborando tus ideas, tener ya la estructura del texto misma e ir refinando el documento componiendolo por partes, es programar un documento. Y eso es lo que exactamente permite LaTeX. **¿Qué mejor herramienta para un programador?**

5.2. Viabilidad económica

Los Proyectos de investigación y desarrollo de software se caracterizan por ser económicos en cuanto a la adquisición de herramientas para llevar a cabo estas investigaciones. Scilab nos provee de muchas facilidades, entre ellas es que es libre y además es gratis; algunas veces se malentienden estos términos y algunas personas los confunden como sinónimos. Cuando se habla de software libre, se trata de

dar el código fuente de las aplicaciones, el precio está sometido a las utilidades y necesidades de los usuarios.

5.3. Viabilidad social

La investigación matemática puede y debe contribuir a todos los aspectos involucrados en el procesamiento de señales: tanto al desarrollo del modelo matemático y su implementación computacional. Para cubrir las necesidades que pueden suplir investigaciones matemáticas se requiere de equipos y de trabajo multidisciplinares integren a ingenieros y matemáticos.

La interacción con centros de investigación o grupos de investigación de instituciones académicas de educación superior, siempre ha sido bienvenida en este caso la Universidad Industrial de Santander *(UIS)* por medio del Grupo de Óptica y Tratamiento de Señales *(GOTS)*, la UIS siempre ha estado activa y participe a colaborar en los procesos académicos de la Universidad del Magdalena.

Capítulo 6

Objetivos

6.1. Objetivo general

Desarrollar una aplicación utilizando Scilab para realizar e implementar la transformación de Fourier fraccionaria y el filtro de Wiener fraccionario para el tratamiento de señales digitales.

6.2. Objetivos específicos

- Estudiar y analizar los fundamentos matemáticos de la transformación de Fourier fraccionaria para el procesamiento de señales digitales.
- Implementar el teorema de muestreo fraccionario en la transformación de Fourier fraccionaria.
- Desarrollar las diferentes funciones para obtener la transformación de Fourier fraccionaria y el filtro de Wiener fraccionario para el tratamiento de señales.
- Demostrar la relación existente entre la transformación de fourier fraccionaria y la distribución de Wigner.

Capítulo 7

Formulación & Hipótesis

En este capitulo se presentan las ideas principales de esta investigación, se formula la implementación del algoritmo de la transformación de fourier fraccionaria utilizando el kernel de la transformación de Fourier estándar, este algoritmo se utiliza para implementar la convolución fraccionaria en forma de operadores, la cual se utiliza para la implementación del filtro de Wiener fraccionario, este filtro actúa como un filtro causal-adaptativo en la extracción de un ruido aleatorio (no estacionario).

7.1. Algoritmo implementado

De la misma forma como la DFT tiene una forma de ser realizada rápidamente la $FrFT$ también tienen diferentes formas de ser implementada, ya sea la mencionada en la sección 3.3 u otras utilizando algoritmos de la FFT. La forma discreta de la ecuación 2.5 está dada por la siguiente expresión.

$$x = n\Delta x \qquad\qquad y = k\Delta y$$

$$f_\alpha(k) = C_\alpha e^{i\pi k^2 \Delta y^2 \cot\alpha} \sum_{n=N/2}^{n=N/2-1} f(n) e^{i\pi n^2 \Delta x^2 \cot\alpha} e^{\frac{-2i\pi n \Delta x k \Delta y}{\sin\alpha}} \tag{7.1}$$

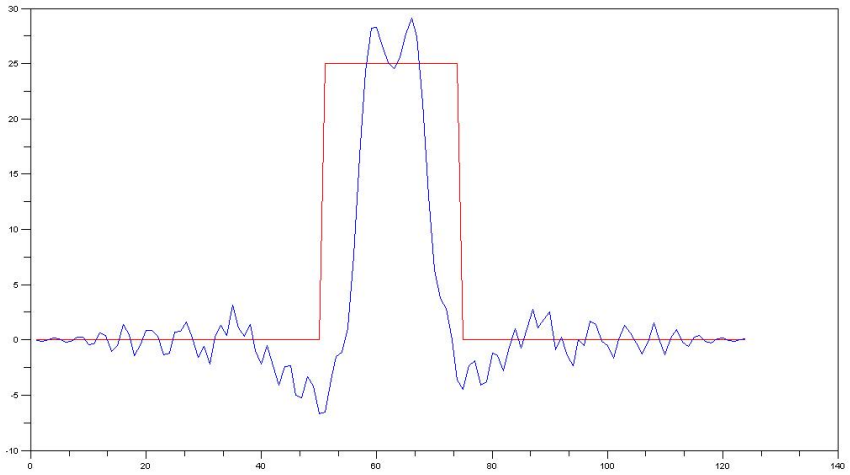

Figura 7.1: $\xi = \zeta$, el tamaño de la Señal es el mismo que el Ancho de Banda de la Transformada

$$\Delta x = \frac{\xi}{N} \qquad\qquad \Delta y = \frac{\zeta}{N}$$

y de esta forma la ecuación 7.1 queda de la siguiente forma:

$$f_\alpha(k) = C_\alpha e^{\frac{i\pi k^2 \zeta^2 \cot\alpha}{N^2}} \sum_{n=N/2}^{n=N/2-1} f(n) e^{\frac{i\pi n^2 \xi^2 \cot\alpha}{N^2}} e^{\frac{-2i\pi nk\zeta\xi}{N^2 \sin\alpha}} \tag{7.2}$$

Teorema 7.1 (Teorema de Muestreo en dominios de Fourier fraccionarios). Sea $f(x)$ una función tal que $f_\alpha[y]$ tiene soporte finito ζ, $f(x)$ puede ser muestreada y reconstruida perfectamente si las muestras se toman a una tasa $\Delta x \leq \sin\alpha/\zeta$ [8].

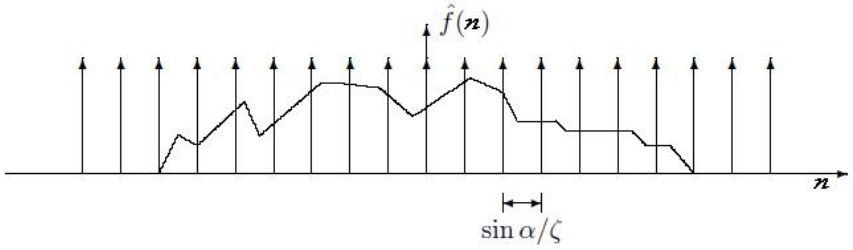

Figura 7.2: Δx en el teorema de muestreo en dominios de Fourier fraccionarios, imagen extraída de [8]

El teorema de muestreo estandar de Shannon-Whittaker se encuentra inmerso como caso particular de este, cuando $\alpha = \pi/2$; tomando en cuenta el Teorema de Muestreo en dominios de Fourier fraccionarios se infiere lo siguiente:

$$\frac{\xi}{N} = \frac{\sin\alpha}{\zeta} \qquad \Rightarrow \qquad \xi\zeta = N\sin\alpha \qquad \xi = \zeta = \sqrt{N\sin\alpha}$$

Según lo anterior se define una expresión para que sea implementada en un algoritmo sobre la plataforma numérica Scilab para el tratamiento de señales, el algoritmo implementado cumple las propiedades y caracteristicas de la ecuación 2.5, este puede solucionarse por diferentes métodos utilizando el algoritmo de la FFT u otras operaciones en su defecto [46], lo cual es explicado en el capitulo 10; al incluir este teorema del muestreo fraccionario la $DFrFT$ queda de la siguiente forma:

$$f_\alpha(k) = C_\alpha e^{\frac{i\pi k^2\cos\alpha}{N}} \sum_{n=N/2}^{n=N/2-1} f(n) e^{\frac{i\pi n^2\cos\alpha}{N}} \boldsymbol{e}^{\frac{\boldsymbol{-2i\pi nk}}{\boldsymbol{N}}} \tag{7.3}$$

El kernel discreto en la ecuación 7.3 contiene el kernel discreto de la transformación de Fourier estándar mostrado en la ecuación 3.2, permitiendo hacer los cálculos con base en la función FFT explicado en la sección 3.1.1 [50]. Como se muestra en la ecuación 7.3 al implementar el teorema de muestreo en dominios de Fourier fraccionarios el cual dice que la $\Delta x = \sin\alpha/\zeta$, siendo ζ el ancho de banda de la transformada de Fourier de la señal a muestrear. Teniendo en cuenta la formulación de la transformación de Fourier estándar descrita en la ecuación 2.1, la transformación de Fourier fraccionaria puede ser implementada utilizando el algoritmo de la Transformación de Fourier de la siguiente forma:

$$f_\alpha(k) = C_\alpha e^{\frac{i\pi k^2\cos\alpha}{N}} \mathfrak{F}\left[f(n) e^{\frac{i\pi n^2\cos\alpha}{N}}\right]. \tag{7.4}$$

Al implementar el kernel descrito en la ecuación 7.4, se puede obtener la $FrFT$ en un intervalo fraccionario $0.5 < |a| < 1.5$, para hallar los valores que no están en

este intervalo se aplican transformaciones de Fourier estándar e inversa, según sea el caso, y se suma o resta un escalar al orden fraccionario a; esto se explica con el siguiente código, el cual es implementado en la función $FrFT$ desarrollada en esta investigación y utilizada en la aplicación realizada en Scilab.

Al igual que el algoritmo explicado en la sección 3.3.1, los parámetros de entrada del algoritmo realizado llamado $FrFT$, la variable de salida es y la cual es la transformada de Fourier fraccionaria de las muestras de entrada, la señal debe ser par, el algoritmo de interpolación que se utiliza es la función sinc para aumentar el tamaño de las a $2N$ para luego colocar ceros a la izquierda y derecha de la señal, y así evitar la superposición de espectros y no provocar aliasing en la reconstrucción de la señal y al final de este proceso de decima para llevar el número de muestras al mismo número de muestras de la función de entrada.

Con el fin de reducir el intervalo fraccionario de la $FrFT$, lo primero que se hace es hallar el módulo del número a, `a=modulo(a,4)`. Al hacer esto se pregunta por el valor absoluto de a, y si este es igual a 2, se aplica la propiedad de reflexión el cual es un caso particular de la $FrFT$, de lo contrario se vuelve y reduce el intervalo de fraccionalización.

```
if(abs(a)==2)
   Fy =fc($:-1:1,:)
else
   a=modulo(a,2);
end;
```

Luego de tener el valor fraccionario de la $FrFT$ entre $0 \leq |a| < 2$, lo primero que hay que hacer es descartar los intervalos que no son posibles de realizar directamente con el kernel de la función. Otro caso particular es la propiedad identidad la cual se da cuando $a = 0$, el cual se nota en las dos primeras sentencias que se muestran en el siguiente código.

```
if (a==0)
   Fy = fc;
else
   if (a>-0.5) & (a<0)
      Fy = kernelFrFT(fftshift(fft(fftshift(fc))),-1-abs(a));
   end;
   if (a>0) & (a<0.5)
      y = kernelFrFT(fftshift(fft(fftshift(fc))),a-1);
   end;
   if (a>-2) & (a<-1.5)
      Fy = kernelFrFT(fftshift(fft(fftshift(fc),1)),1-abs(a));
   end;
   if (a>1.5) & (a<2)
      Fy = kernelFrFT(fftshift(fft(fftshift(fc),1)),a-3);
   end;
   if (abs(a)>=0.5) & (abs(a)<=1.5)
      Fy = kernelFrFT(fc,a);
   end;
end;
```

En las sentencias anteriores se observa una reducción en el intervalo del operador fraccionario de la $FrFT$ y se implementa el código de la Transformación rápida de Fourier, explicado en la sección 3.1.1, y se le suma algebraícamente el excedente para que realice la operación en el operador que se requiere. Por ejemplo, si se supone $a = 0{,}1$, este orden fraccionario entra en el segundo condicional, entonces se aplica una FFT a la función de entrada fc, y se le resta 1 al operador fraccionario, con esto el Kernel de la $FrFT$ realizará una transformación en el orden $-{,}9$ por la propiedad de aditividad que mantiene la $FrFT$, en forma de operadores será entendido mejor.

$$a = 0{,}1; \qquad 0{,}1 - 1 = -0{,}9; \qquad \mathfrak{F}_{1-0{,}9} = \mathfrak{F}_{0{,}1}$$

7.1.1. Función KernelFrFT

Como ya se había dicho anteriormente el kernel de la $FrFT$ puede ser implementado utilizando el algoritmo de la transformación de Fourier en su kernel, y se logró para esta investigación, este algoritmo da como resultado una aproximación muy buena de la $FrFT$ pero posee problemas en su fase, ver figura 7.3, y no cumple varias propiedades de esta Transformación de Fourier fraccionaria como es la aditividad, no obstante, en futuras investigaciones se podrá solucionar este inconveniente y se logrará utilizar la FFT en la $FrFT$, haciendo este algortmo más rápido y eficiente

En las siguientes sentencias de código se muestra el kernel que no funcionó para la función de la Transformación de Fourier fraccionaria, claramente se muestra la multiplicación de una función chirp por la función de entrada, para luego aplicar a este resultado una Transformación rápida de Fourier y luego multiplicar por otra función Chirp y la constante mostrada en la ecuación 2.6.

```
x = [-ceil(N/2):fix(N/2)-1];

chirp1 = exp((%i*%pi*cotg(alpha)*(x.*x))/N);

chirp1 = chirp1(:);

fch = fc.*chirp1;

Hcfft = fftshift(fft(fftshift(fch),-1));

Aphi=exp(-%i*(%pi*sign(sin(alpha))/4-alpha/2))/sqrt(abs(sin(alpha)));

y = (Aphi*chirp1.*Hcfft);
```

Por tal razón con el fin de solucionar la expresión de la ecuación 7.4 se hace uso de

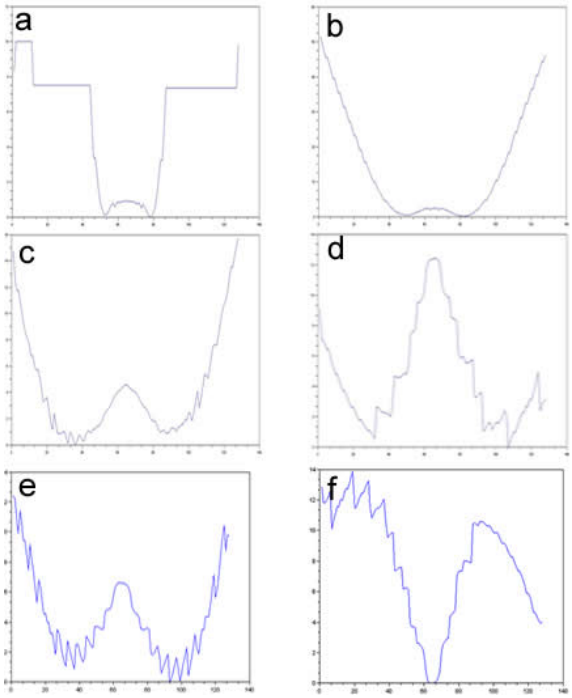

Figura 7.3: Fases de la $FrFT$ con el algoritmo de la FFT en el kernel; a) $\mathfrak{F}_{0,05+0,05}$; b) $\mathfrak{F}_{0,15+0,15}$; c) $\mathfrak{F}_{0,2+0,3}$; d) $\mathfrak{F}_{0,3+0,3}$; e) $\mathfrak{F}_{0,4+0,5}$; f) $\mathfrak{F}_{1+0,1}$.

la discretización total de la señal de la siguiente manera:

```
x = [-ceil(N/2):fix(N/2)-1];
```

```
x=x(:);
chirp1 = exp((%i*%pi*cotg(alpha)*(x.*x))/N);
chirp1 = chirp1(:); fc = fc.*chirp1;
W = exp(-2*%i*%pi*(x*x')/(sin(alpha)*N));
Hcfft=W*fc;
Aphi = exp(-%i*(%pi*sign(sin(alpha))/4-alpha/2))/sqrt(abs(sin(alpha)));
y = (Aphi*chirp1.*Hcfft);
```

En las sentencias de código arriba mostradas, se observa la declaración de una función Chirp de la siguiente forma $e^{\frac{i\pi \cot \alpha x^2}{N}}$, luego esta es multiplicada por la función de entrada y posteriormente este resultado es multiplicado por una matriz de $N \times N$, realizando una operación matricial, donde se obtiene una sumatoria del producto de cada una de las filas de la matriz W por el vector resultante. Después se multiplica por la constante y otra función Chirp similar a la descrita anteriormente.

7.2. Convolución fraccionaria

Gracias a la convolución en el dominio directo de las señales se pueden realizar infinidades de operaciones como filtros, detección de patrones, entre otras. Pero ahora, con el advenimiento del tratamiento de señales en el dominio de Fourier fraccionario llegaron nuevas operaciones como la convolución y la correlación fraccionaria [10, 51], la convolución fraccionaria de dos funciones $f(x)$ y $g(x)$ se denota de la siguiente manera:

$$[f \overset{\alpha}{*} g] = \mathfrak{F}_{-\alpha}[\mathfrak{F}_{\alpha}[f]\mathfrak{F}_{\alpha}[g]] \tag{7.5}$$

Teniendo en cuenta la tabla 3.2 se nota que cuando el angulo de la $FrFT$ es $\pi/2$ se realiza una Transformación de Fourier estándar por lo cual la convolución fraccionaria será una convolución estandar también, y si $\alpha = 0$ entonces se cumplirá el caso del operador identidad para ambas funciones y se realizará un simple producto entre estas dos funciones [12, 9]. La Convolución fraccionaria se soluciona en forma de operadores utilizando $FrFT$ como se muestra adelante.

$$[f \overset{*}{\alpha} g](x) = \mathfrak{F}_{-\alpha}[\mathfrak{F}_\alpha[f](x_\alpha)\mathfrak{F}_\alpha[g](x_\alpha)e^{i\pi x^2 \cot\alpha}](x). \tag{7.6}$$

Y la forma integral se muestra de la siguiente forma, ver capitulo 10:

$$[f \overset{*}{\alpha} g](x) = \int f(y)g(x-y)e^{2i\pi y(x-y)\cot\alpha}dy. \tag{7.7}$$

Como se menciona anteriormente en la sección 7.2, la convolución fraccionaria de la ecuación 7.5 presenta casos particulares según el grado de fraccionalización, de la misma forma el integral mostrado en la ecuación 7.8, la cual realiza esta operación puede ser solucionado de diferentes formas [9, 10] entre ellas: Al evaluar el factor de fase del integral resulta la expresión 7.8:

$$[f \overset{*}{\alpha} g](x) = \int f(y)g(x-y)e^{\pi i y^2 \cot\alpha}e^{-2\pi i x y \cot\alpha}dy \tag{7.8}$$

otra forma es la siguiente la mostrada en la ecuación 7.9

$$[f \overset{*}{\alpha} g](x) = e^{\pi i x^2 \cot\alpha}\,(f'_\alpha * g'_\alpha) \tag{7.9}$$

siendo

$$f'_\alpha = f(x)e^{\pi i x^2 \cot\alpha} \qquad y \qquad g'_\alpha = g(x)e^{\pi i x^2 \cot\alpha}$$

La forma más fácil y rápida de implementar la convolución fraccionaria se expone en la siguiente expresión:

$$h(k) = \sum_{n=N/2}^{n=N/2-1} f(n)g(k-n)e^{\frac{-2\pi i k(n-k)\cot\alpha}{N}} \tag{7.10}$$

otra forma se realiza tomando la forma de la ecuación 7.8, donde se observa que la forma del factor que multiplica ambas funciones se asemeja la transformación de Fourier estándar, con la diferencia que el factor de fase tendría la función $\cot\alpha$; esta forma quedaría de la siguiente manera:

$$h(k) = \sum_{n=N/2}^{n=N/2-1} [f(n). \times Chirp] \times W * [g(n)W]\, Chirp \tag{7.11}$$

siendo

$$W = e^{-2i\pi nk\cot\alpha} \qquad y \qquad Chirp = e^{-\pi in^2\cot\alpha}$$

después de realizar estos productos se aplica la función de Convolución estándar.

En esta investigación se lograron implementar estos algoritmos, pero no fue posible hacer que dieran los resultados esperados debido a incoherencias que se obtenían en su simulación. Por lo cual se toma la decisión de implementar la convolución fraccionaria aplicando la $FrFT$ por medio de operadores descrita en la ecuación 7.6 del capitulo 7 la cual sumple el teorema de la convolución planteado en el Teorema 3.1 del capitulo 3, este algoritmo se explica adelante.

```
function h=FrConvol(f,g,a)
alpha=a*N=length(f);
M=length(g);

if(N>M)
   g(N)=0;
   N1=N;
   elseif (M>N)
```

```
      f(M)=0;
      N1=M;
else
   N1=N;
end;
n = [-ceil(N1/2):fix(N1/2)-1];

f1=FrFT(f,a);
g1=FrFT(g,a);
Ch=exp(%i*%pi*(n.*n)*cotg(alpha)/N1);
h1=f1.*g1.*Ch';
h=FrFT(h1,-a);
endfunction
```

Los parámetros de entrada de la señal son `f,g,a`, donde el primero es el número de muestras de una señal f, el segundo el número de muestras de otra señal g y el tercero el orden de fraccionalización de la operación, en la segunda sentencia se halla el ángulo y posteriormente se almacenan en variables los tamaños de las señales de entrada, con el fin de rellenar con 0 la función de menor tamaño e igualar tamaños al momento del proceso de convolución.

Puede observarse que en la variable h1 es guardado el producto de `f1,g1,Ch`, siendo la primera la transformada de Fourier fraccionaria de la función de entrada `f`, la segunda el modulo cuadrado en el dominio fraccionario de Fourier de las muestras contenidas en el parámetro de entrada `g` y la tercera es un factor de fase cuadrático tal como se ve en la ecuación 7.6.

7.3. Filtro de Wiener fraccionario

El objetivo de esta sección es la implementación computacional de las funciones necesarias para crear un filtro $g(x)$ en Scilab, de modo que la salida $h(x)$ sea lo más parecidamente posible a una señal denominada referencia, estas funciones son expuestas en [12]. Sea $f(x)$ una señal que presenta un ruido aditivo aleatorio, tal que $f(x) = e(x) + r(x)$, siendo e la escena y r el ruido, entonces, se propone un filtro el cual a través de la operación de convolución fraccionaria explicada en la sección 7.2, elimina el ruido sumergido en la naturaleza de la señal, la función de filtro se muestra en la ecuación 7.12:

$$G^{\alpha}(v_{\alpha}) = \frac{E_S^{\alpha}(v\alpha)}{E_S^{\alpha}(v\alpha) + R_R^{\alpha}(v\alpha)} e^{-i\pi v_{\alpha}^2 \cot\alpha} \tag{7.12}$$

donde $E_S^{\alpha}(v\alpha)$ es el modulo cuadrado en el dominio de Fourier fraccionario de la señal sin el ruido y $R_R^{\alpha}(v\alpha)$ es el modulo cuadrado en el dominio fraccionario del ruido que distorsiona la señal. Este filtro debe ser implementado usando la convolución fraccionaria tal como se muestra en la ecuación 7.13.

$$\hat{H}_y(x) = [F_y \overset{*}{\alpha} g](x) \tag{7.13}$$

En la ecuación 7.13 se observa que el filtro aparece en el dominio directo de la señal y la señal se encuentra en el dominio de Fourier fraccionario, por lo cual antes de aplicar la convolución fraccionaria se debe realizar una $FrFT$ a la señal de referencia y al ruido. El proceso de distorsión de una función rectángulo por medio de ruido aleatorio aditivo de una función Chirp multiplicada por una Gauss, el filtro implmentado y la operación de convolución se realiza en la plataforma de cálculo numérico Scilab como se muestra en las siguientes sentencias de código.

```
fc=[zeros(1,400), ones(1,224), zeros(1,400)];
N=length(fc);
t=0;
a=.5;
x=[-12:.5:12.5];
y=(2*rand(1,50)-1)/10;
ruido=[zeros(1,150),5*exp(%i*(.02*%pi*x.*x)).*(exp(-(x.*x)/100+y)),
      zeros(1,824)];
ruido=ruido(:);
fy=FrFT(fc,a);
ff=[fy+ruido*25];
fc1=FrFT(ff,-a);
v=[-N/2:N/2-1];
FF=exp(%i*%pi*(v.*v)/N*cotg(alpha));
SB=FrFT(ruido,a);
G=(((fy)**2)/(((fy)**2)+((SB)**2)))*FF';
g=FrFT0(G,-a);
h=FrConvol(fy,g,a);
```

Así, este tratamiento consigue un filtrado de un ruido no estacionario aleatorio, debido a rotaciones que realiza la transformación de Fourier fraccionaria sobre la distribución de Wigner asociada a la señal, permitiendo tener conocimiento de las frecuencias de la señal durante todo el tiempo, extrayendo el ruido presente en el dominio directo de la señal de manera adaptable en el dominio de Fourier fraccionario, la demostración de este proceso se puede apreciar en el capitulo 10.

Capítulo 8

Metodología

Al considerar la implementación computacional de la transformación de Fourier fraccionaria y el filtro de Wiener fraccionario sobre la plataforma Scilab, se buscan las ventajas del tratamiento de señales en el dominio de Fourier fraccionario frente al análisis estándar y otras que se venían realizando en el tratamiento de señales representadas en tiempo frecuencia [48]. Esta práctica requiere de mucho sacrificio de tiempo en investigación en fases como análisis y requerimientos de la información para cumplir los objetivos propuestos en esta investigación.

8.1. Metodología XP

Tras buscar metodologías de desarrollo de software que se adaptaran a esta inevstigación, se pudo constatar que la todas presentan fases y conceptos en común, pora tal razón, para seleccionar la metodología en esta investigación se tomó en cuenta algunos aspectos de tiempo y personal presente durante el ciclo de vida de la aplciación, la metodología que se ajusta a esta investigación es la *Xtremme Programming* (XP por sus siglas en inglés). XP es una de las metodologías de desarrollo de software más exitosas en la actualidad utilizadas para proyectos de corto plazo, corto

equipo y cuyo plazo de entrega era ayer, está basada en una serie de valores y de prácticas de buenas maneras de programación que persigue el objetivo de aumentar la productividad a la hora de desarrollar aplicaciones de software, esta consta de 4 fases: análisis, diseño, desarrollo y pruebas.

La interacción de estas cuatro fases se pueden describir como un rompecabezas figura 8.1, en la medida en que se avanza por cada una de las fases siempre hay comunicación entre ellas, la fase de pruebas se encuentra inmersa durante todo el proceso de desarrollo, de esta forma siempre hay una retroalimentación continua a través de las fases que se enmarcan en la metodología XP. Utilizar esta técnica para el desarrollo de esta aplicación no solo es posible sino que también puede ser provechosa para nuestra experiencia en el desarrollo de aplicaciones en el campo del software libre. La documentación se hace al final de aplicación; a continuación se relatan las actividades que se realizarán durante la investigación según cada una de las fases de la metodología seleccionada.

8.1.1. Análisis

Esta fase es la más importante de todas ya que en ella se realiza el estudio de los fundamentos matemáticos de la transformación de Fourier fraccionaria, sus propiedades, y el filtro de Wiener fraccionario sobre señales digitales; de la misma forma se trazan a lapiz, los modelos de las diferentes funciones a utilizar para llevar a cabo las operaciones que permitan culminar los objetivos propuestos. También se toman en cuenta los requerimientos funcionales de la aplicación y se modelan los

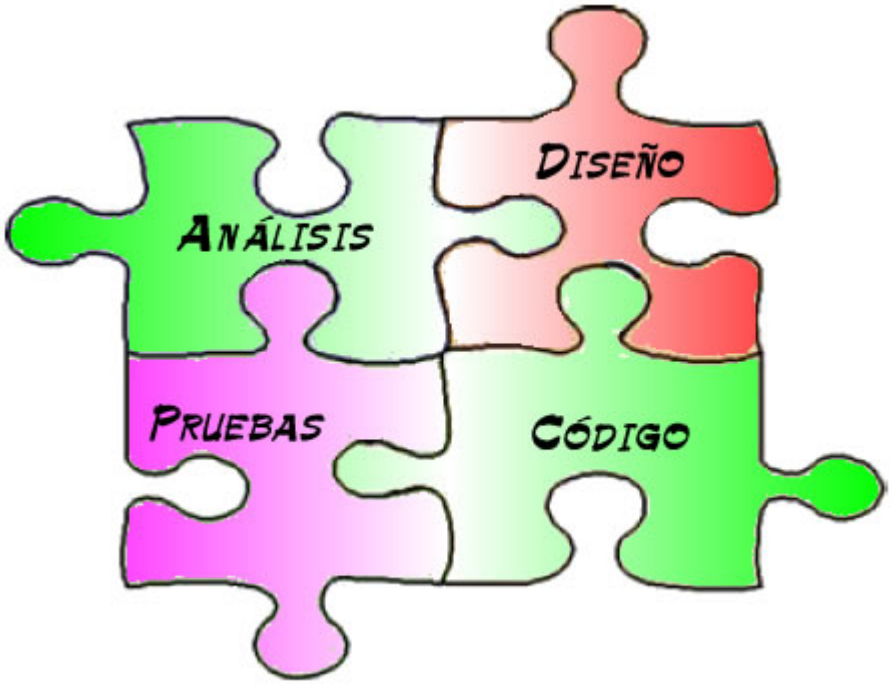

Figura 8.1: Metodología XP

procesos a través de las diferentes fases aquí expuestas.

8.1.2. Diseño

En esta fase se establece el diseño de la interfaz que interactuará con el usuario y el esquema de funciones que se necesitarán para llevar a cabo la Transformación de Fourier fraccionaria, la operación de la distribución de Wigner, la operación de convolución y el filtro de wiener fraccionario, quizás también otras transformaciones representadas en tiempo-frecuencia para obtener conclusiones experimentales en la investigación. En la aplicación se implementa un diseño simple, para enfocarse en los resultados científicos que se obtienen, y también porque una simplicidad del diseño, hará que sea más fácil añadir nuevas funcionalidades en un futuro.

8.1.3. Desarrollo

Teniendo en cuenta que en esta fase se ha entendido el fondo y la teoría matemática que abarca la $FrFT$ y el filtro de Wiener fraccionario, como también la discretización de cada una de las formulas a utilizar en estos algoritmos; se comienza a desarrollar el código de estas operaciones y también se establecen los mecanismos y pautas, para que sea revisado la fase de diseño de la investigación.

En esta fase se hace la integración de las funciones y los recursos necesarios para que el sistema funcione y se obtenga un óptimo procesamiento de señales digitales representadas en tiempo-frecuencia en el dominio de Fourier fraccionario. Si se sigue al pie de la letra las normas y técnicas de la presente metodología de desarrollo, al final de esta fase se obtendrá un código simple y funcional.

8.1.4. Pruebas

Scilab es una plataforma rápida que permite ejecutar rápidamente funciones en el interprete de código, por lo cual desde el primer momento en que se va programando y probando, se realizan pruebas en la aplicación, y de esta forma se asegura que el código escrito hasta entonces en algún momento es correcto y no estropea lo anterior.

Por tal razón esta fase no está precisamente al final sino durante todo el proceso del ciclo de vida del software, y al final de la fase de desarrollo estaría la aplicación debiera estar lista. Sin embargo, se hacen las suficientes pruebas para que la aplicación sea estable.

Capítulo 9

Desarrollo de la aplicación

Posterior al análisis matemático se procedió a realizar cada una de las funciones expuestas en el capitulo 7 y modelar la interfaz gráfica para la interacción con el usuario, ver figura 9.1, en esta aplicación el usuario tiene la posibilidad de cargar diferentes funciones y de aplicar el algoritmo de la tranformación de Fourier fraccionaria ($FrFT$), la Distribución de Wigner, graficar cada una de estas, entre otras funcionalidades.

La Metodología explicada en la sección 8.1, requiere que el grupo de trabajo que participa en el ciclo de vida del desarrollo del software trabajen en comunicación constante y participes a la solución de problemas que se presenten durante el desarrollo del sistema, por lo cual se hizo necesario una visita de investigación al Grupo de Óptica y Tratamiento de Señales (GOTS), de la Universidad Industrial de Santander (UIS). Durante la digitación del código se siguieron pautas de programación, comentando y tabulando el código de las diferentes funciones y algoritmos desarrollados.

Luego de tener un modelo de interfaz gráfica y los algoritmos de las operaciones

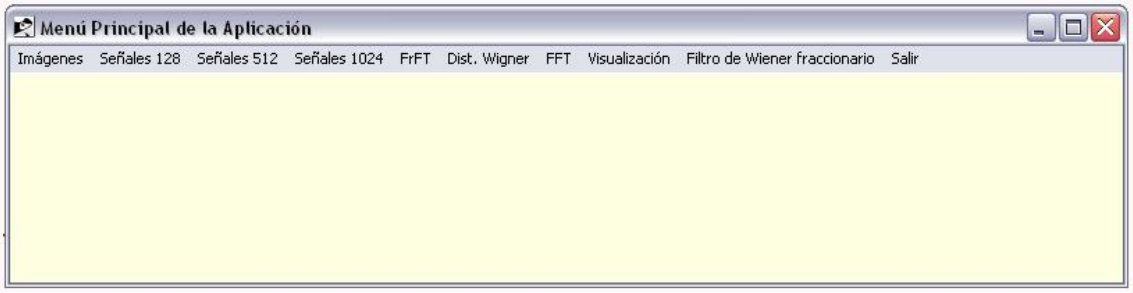

Figura 9.1: Menú Principal de la aplicación

analizadas en este trabajo, se fueron construyendo las funciones necesarias para la implementación de estas en la plataforma de cálculo numérico Scilab, y creando señales de diferentes escalas con el fin de corroborar las propiedades y caracteristicas de la $FrFT$ en el tratamiento de señales.

La aplicación realizada presenta algunos Menús en la parte superior, por medio de los cuales se puede acceder a submenús con el fin de generar imágenes y señales, a las cuales se les aplica el algoritmo de la $FrFT$ implementado en este trabajo, también se puede hallar la distribución de Wigner de la señal y de las transformadas. Esta aplicación se realiza para que el usuario pueda corroborar algunas propiedades de la $FrFT$, los menús de la aplicación se exponen de la siguiente manera:

1. Imágenes

    - Este menú presenta 3 submenú, el primero con el fin de abrir y guardar una imagen para luego tratarla con las opciones de la transformación de fourier fraccionaria, del segundo y tercer submenú, donde en el primero se realizan una $FrFT$ a la imagen guardada en el orden fraccionario ingresado en las filas y columnas de la imagen, en el tercero se realiza esta misma operación a la transformada de la imagen, pudiendo navegar

por así decirlo en el dominio de Fourier fraccionario de una señal bidimensional encontrando información valiosa y regresándose nuevamente al dominio directo de la imagen. Las variables que se crean en la aplicación para guardar las funciones que se utilizan en estos submenús son `fc2d`, `Fy2d` declaradas como globales en la aplicación realizada, `fc2d` es la señal bidimensional y `Fy2d` es la transformada de la variable anterior.

2. Señales de 128, 512 y 1024 muestras

   - En este numeral se exponen 3 menús, por medio de los cuales se generan y grafican señales de diferentes tamaños, con el fin de que se le apliquen las operaciones de $FrFT$, Distr. de Wigner y FFT; el objeto de que se carguen las mismas señales en los tres menús principales consiste en la demostración de la propiedad de escala de la $FrFT$ y la FFT, descrita en la ecuación 4.6. Las señales que permite generar la aplicación por medio de operaciones vectoriales como `zeros, ones, linespaces, exp,` entre otros son:
     * Rectángulo, ver figura 9.2.
     * Triangulo, ver figura 9.3.
     * Coseno, ver figura 9.4.
     * Seno, ver figura 9.5.
     * Chirp, ver figura 9.6.
     * Gaussiana, ver figura 9.7.
     * Chirp*Gauss, ver figura 9.8.

3. FrFT

   - En este tiene dos (2) opciones, aplicar la $FrFT$ a la señal y aplicar la

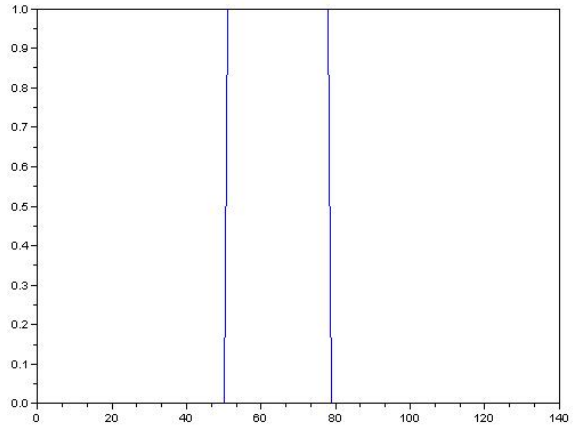

Figura 9.2: Señal rectángulo de 128 muestras

.

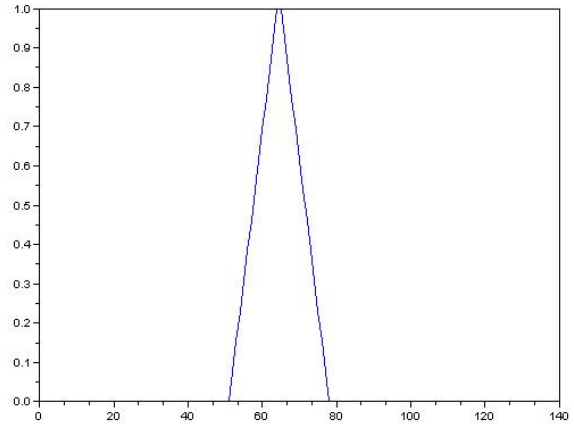

Figura 9.3: Señal triangulo de 128 muestras

.

$FrFT$ a la transformada de la señal, en ambas opciones la aplicación solicita la digitación del orden fraccionario de la operación y toma como función de entrada la variable `fc` creada en los menús de señales, la variable de salida es `fy` graficada al terminar el proceso de la evaluación computacional.

4. Distribución de Wigner

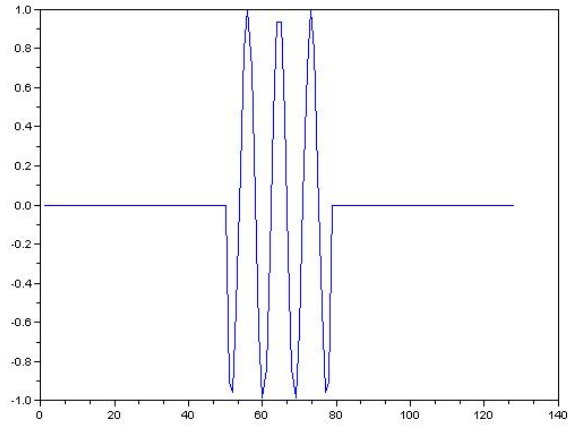

Figura 9.4: Señal coseno de 128 muestras

.

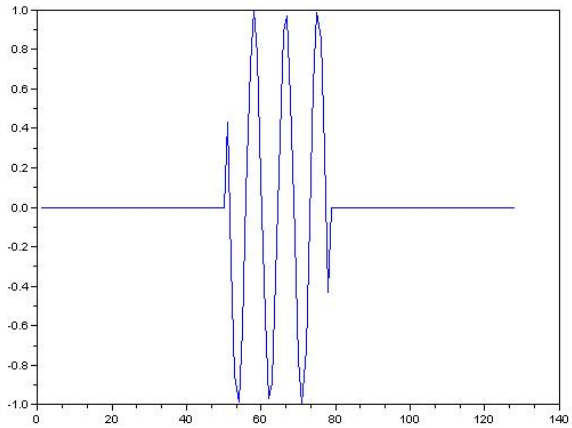

Figura 9.5: Señal Seno de 128 muestras

.

- Por medio de la segunda opción del menú anterior, se puede observar la rotación de la distribución de Wigner que se encuentra asociada a una función, con los submenús de este menú, ya que a medida de que el orden fraccionario a de la $FrFT$ se aplica la distribución de Wigner y esta se visualiza utilizando la sentencia `imshow` de la librería Scilab Image Processing, ya que como se dijo en la sección 3.3, esta función es una señal bidimensional.

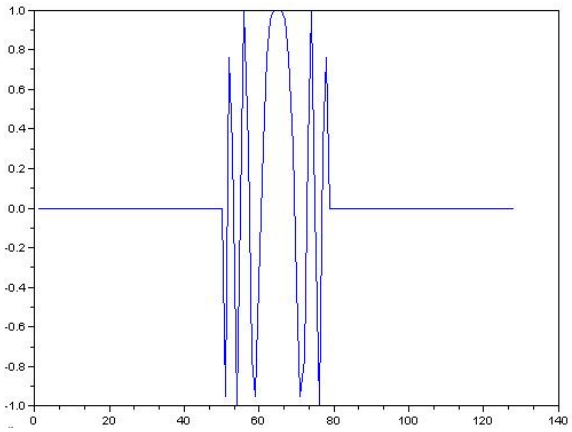

Figura 9.6: Señal Chirp de 128 muestras

.

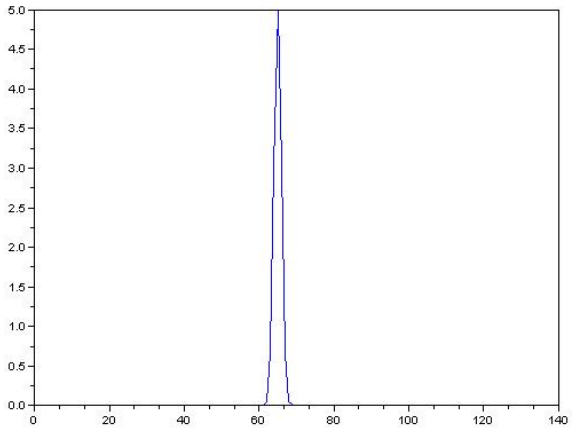

Figura 9.7: Señal Gaussiana de 128 muestras

.

5. Filtro de wiener fraccionario El proceso que se realiza al hacer clic sobre la opción que presenta este menú se explica en la sección 7.3 y la demostración de esta sección se aprecia en la sección 10.4.

Las diferentes funciones realizadas para llevar a cabo esta aplicación se organizaron por menús y se colocaron sobre una carpeta llamada *Function*, estas funciones son llamadas por cada uno de los submenús de la aplicación, cambiando el valor de las variables globales de la aplicación. Los tutoriales y la documentación que se

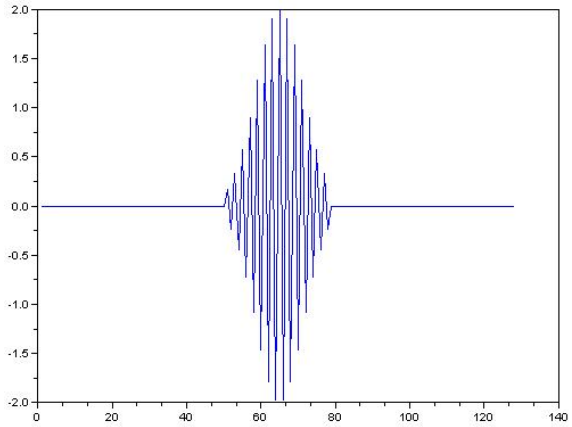

Figura 9.8: Señal Chirp*Gauss de 128 muestras

.

encuentra en internet acerca de Scilab ayudó mucho a la realización de este trabajo.

Capítulo 10

Demostración de hipótesis y resultados

Durante la etapa de análisis se realizó la busqueda de la información que sirviera como base teórica de la presente información, en esta busqueda surgieron dudas que llevaban a seguir investigando acerca de las bases matemáticas. En esta misma etapa se realizó un estudio de la $FrFT$ y las demás funciones que se relacionan con esta operación, como la Distribución de Wigner, la operación de convolución, la convolución fraccionaria, entre otras [5, 51, 9].

10.1. Señales Unidimensionales

El objetivo de crear señales con diferentes tamaños consiste en corroborar lo dicho en la ecuación 4.6 acerca del teorema de escalamiento, el cual dice que si una función es escalada entonces su transformación de Fourier fraccionaria será escalada también de manera inversa que la función y la $FrFT$ no será la misma ya que el angulo con el cual esta es realizada estaría dado por la potencia fraccionaria $\beta = \arctan(c^2 \tan\alpha)$, y el angulo estaría dado por $\beta = b\pi/2$. Algo similar ocurre con la Transformación de Fourier estándar con la diferencia que la segunda premisa es falsa y la FT de una

función escalada será la misma Transformada escalada inversamente a la función, es decir, al aplicar un factor de escala m a una función, su FT se escala inversamente, ver figura 10.1. matematicamente se describe esto de la siguiente manera:

$$f(x/m) \xrightarrow{\mathfrak{F}} |M|F(my).$$

Explicado esto en el lenguaje coloquial sería, al aumentar la escala de una función $f(x)$ entonces el espectro de frecuencia de su transformada $F(y)$ se reduce, es decir, el escalamiento es inverso o indirectamente proporcional.

En la figura 10.2 se puede observar que el operador entre más tiende a 0, se obtiene un espectro frecuencial muy parecido a la función de entrada, y de esta forma se puede tener la información real de la frecuencia de la señal durante todo su espacio o dominio directo de la misma.

10.2. Distribución de Wigner

Una de las propiedades y utilidades más importantes de la $FrFT$, es su representación como una rotación de la distribución de Wigner de una señal, en la figura 10.3 se puede observar la distribución de Wigner de una señal rectángulo de 1024 muestras, y en la figura 10.4 se muestra las rotaciones de esta función en la distribución de wigner aplicando la transformación de Fourier fraccionaria, empezando desde 0.1 y terminando en 2 de izquierda a derecha y de arriba a abajo, siendo el angulo de rotación $\alpha = a\pi/2$ y a el operador fraccionario de la $FrFT$ tal como se ha venido diciendo durante todo este trabajo.

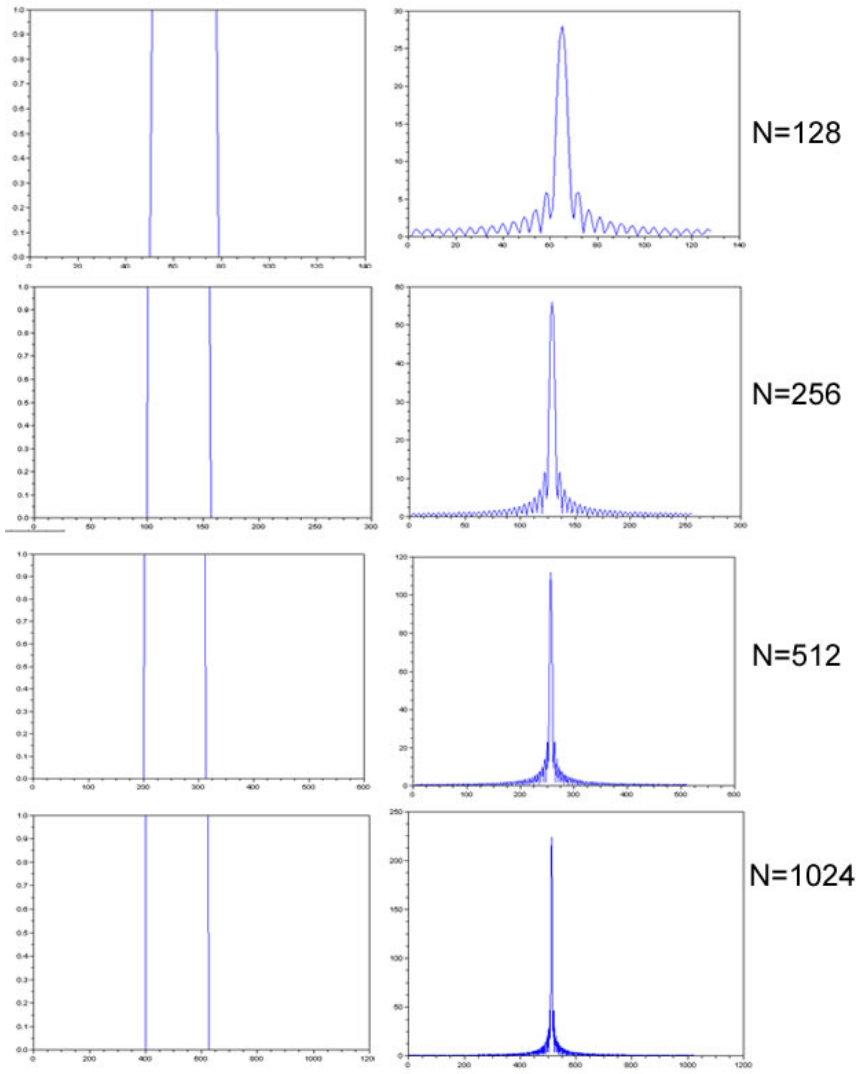

Figura 10.1: Función Rectángulo escalada y su Transformada de Fourier.

10.3. Señales Bidimensionales

De la misma forma se implementa la $FrFT$ para el tratamiento imágenes, aplicando el algorutmo realizado en las filas y columnas [45].

En la figura 10.8 se halló la transformada de Fourier estándar valiendose de la propiedad de aditividad de la $FrFT$ y se le aplica una potencia fraccionaria de $a = 0{,}5$ a la señal de la figura 10.6, siendo esta última la transformada de la figura

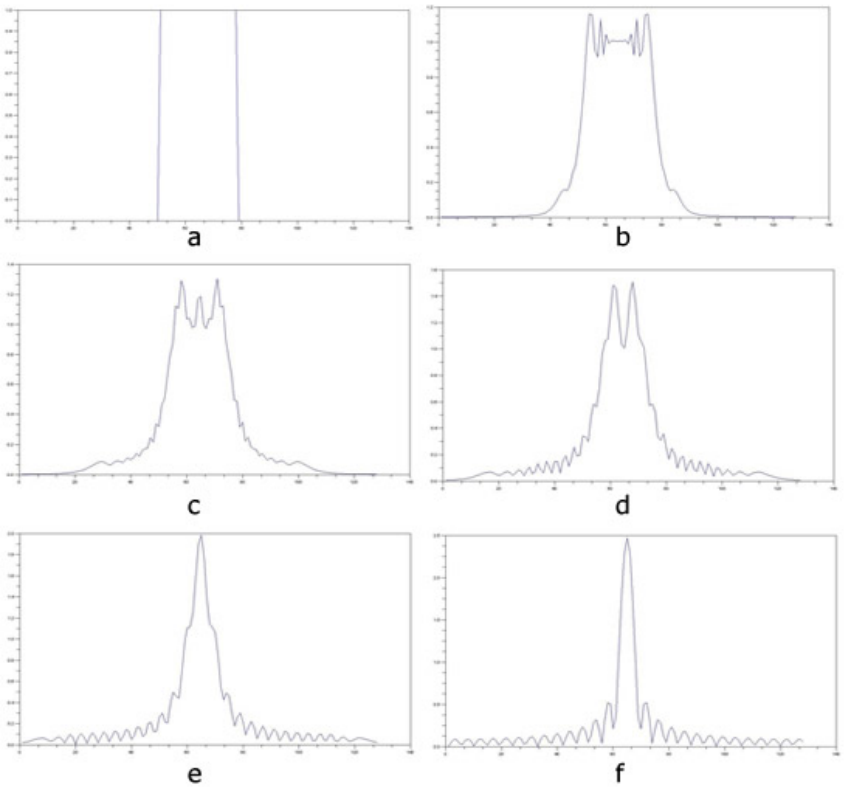

Figura 10.2: a) Función Rectángulo; b) $FrFT$ con a=0.1; c) $FrFT$ con a=0.3; d) $FrFT$ con a=0.5; e) $FrFT$ con a=0.7; f) $FrFT$ con a=0.9.

10.5. Con el fin de mostrar varios ejemplos con diferentes imágenes en este documento, utilizando la aplicación realizada se carga la imagen onion.jpg del Toolbox de procesamiento de imágenes de Matlab tal como se muestra en la figura 10.10, a esta imagen se le aplica la transformación de Fourier fraccionaria con $a = 0{,}25$ en las filas y con $a = 0{,}75$ en las columnas, tal como se aprecia en la figura 10.11 y luego se regresa la imagen a su dominio directo aplicando la $FrFT$ con $a = -0{,}25$ en las filas y con $a = -0{,}75$ en las columnas como se visualiza en la figura 10.12.

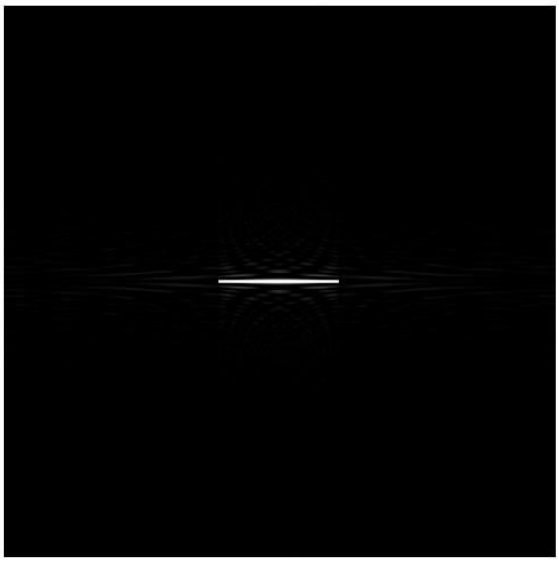

Figura 10.3: Distribución de Wigner de la señal rectángulo de 1024 muestras.

10.4. Filtro de Wiener fraccionario

El proceso de filtrar una señal $f(x)$ casi siempre se ha entendido en términos frecuenciales. Es decir, los filtros realizados a señales están fuertemente relacionados con la transformación de Fourier. Todo el proceso de filtrado consiste en si un filtro de función $g(x)$ es una versión paso bajo, paso banda o paso alto de frecuencias de la original [2].

Lo anterior ha restringido demasiado el proceso realizado por un filtro $g(x)$, el cual venía determinado por la viabilidad tecnológica de producir la salida $h(x)$, siendo esta otra versión de la señal de entrada. La puesta en marcha del tratamiento digital de señales no es más que el uso de la tecnología capaz de realizar cualquier operación matemática sobre una señal $f(x)$, más específicamente sobre su versión muestreada $f(n)$, esto abrió enormemente las perspectivas de los tipos de filtrado que podían ser realizados sobre una función. El filtro de Wiener fraccionario es un función por

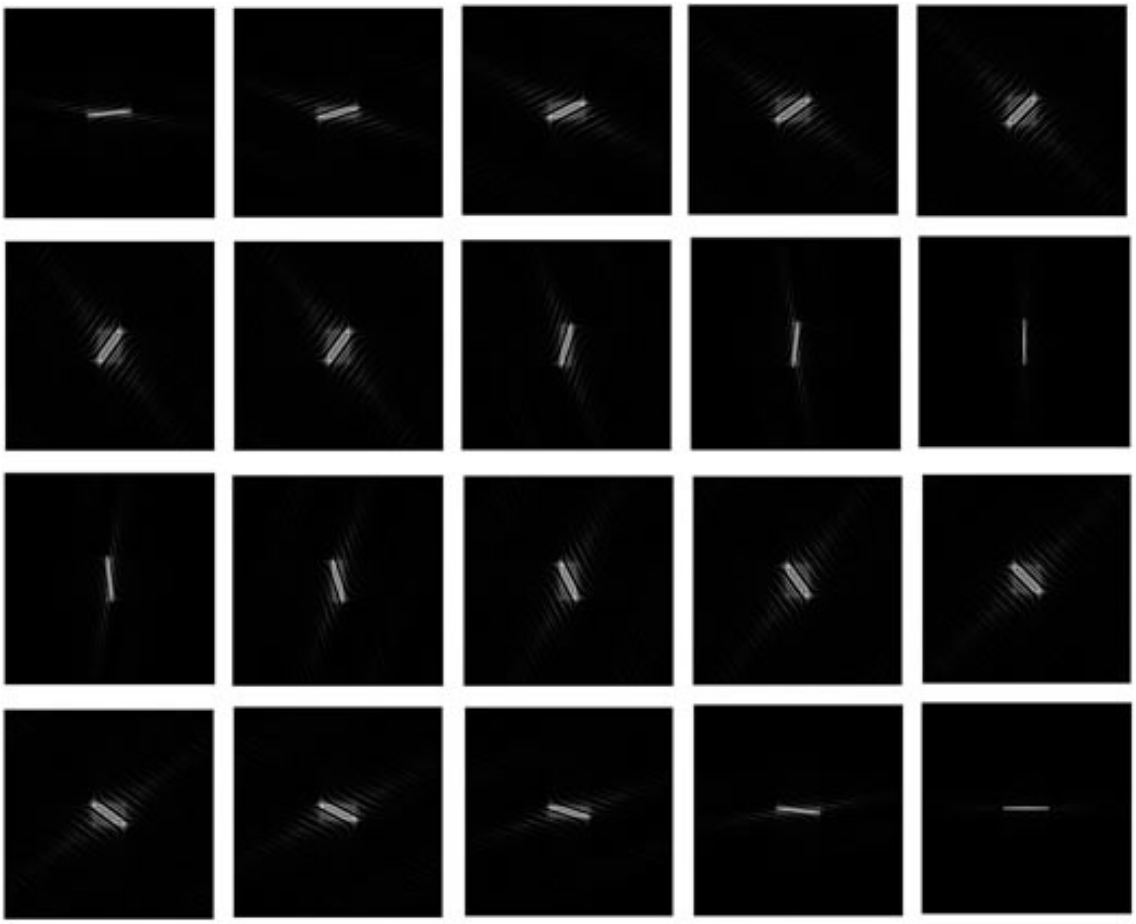

Figura 10.4: Rotación de Dis. Wigner por medio de la $FrFT$ de una función rectángulo de 1024 muestras.

medio de la cual se restaura una señal que ha sido distorsionada por un ruido aleatorio. Este proceso se describe así, en la figura 10.13 se muestra la transformada de una señal rectángulo de 1024 muestras con un operador fraccionario $a = -,5$ lo que representaría una rotación de la distribución de Wigner en un angulo $\alpha = -\frac{\pi}{4}$, ver figura 10.14, luego en el dominio de Fourier fraccionario se le agrega un ruido aditivo de una función chirp multiplicado por una función gaussiana afectada por un valor randómico obteniendo un ruido no estacionario de frecuencias y amplitudes linealmente variantes en el tiempo, el valor absoluto de este resultado se muestra en

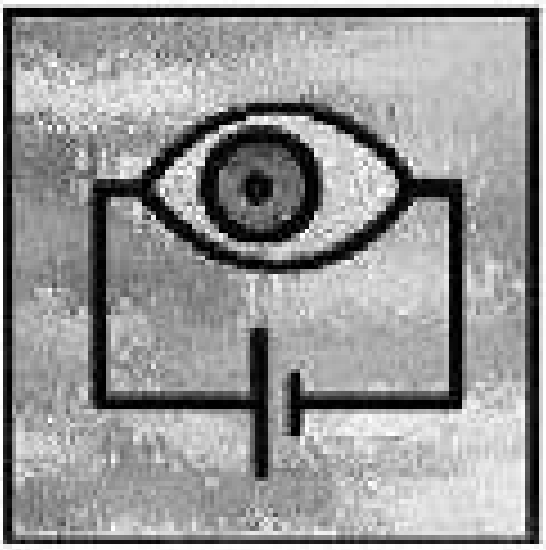

Figura 10.5: Imagen Tru.jpg de Scilab Image Processing.

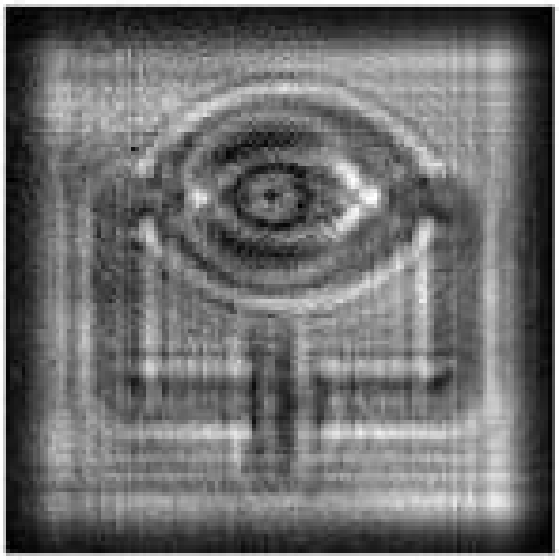

Figura 10.6: $FrFT$ de la imagen 10.5 con $a = 0{,}5$ en las filas y columnas.

la figura 10.15 y en la figura 10.16 su valor real donde se muestra perfectamente la diferencia entre las dos señales, al regresar la señal rectángulo a su dominio directo, el ruido se encontrará en el dominio de Fourier fraccionario de la señal, en este caso, al aplicar una $FrFT$ con $a = 0{,}5$ a la transformada de la figura 10.15, se obtiene

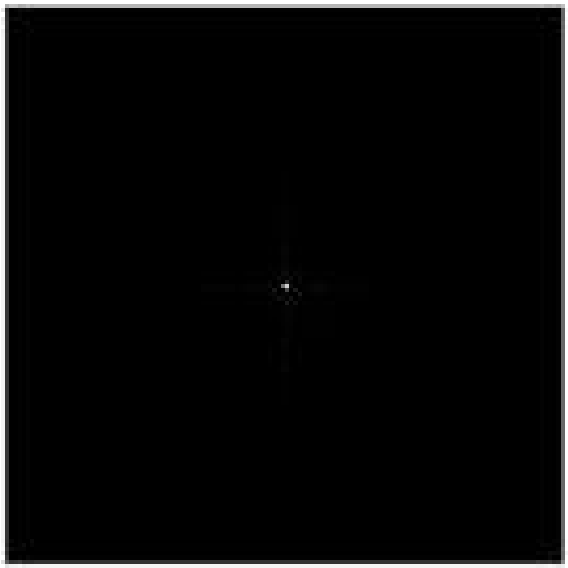

Figura 10.7: $FrFT$ de la transformada de la figura 10.6 con $a =$,5 (Aditividad=FT).

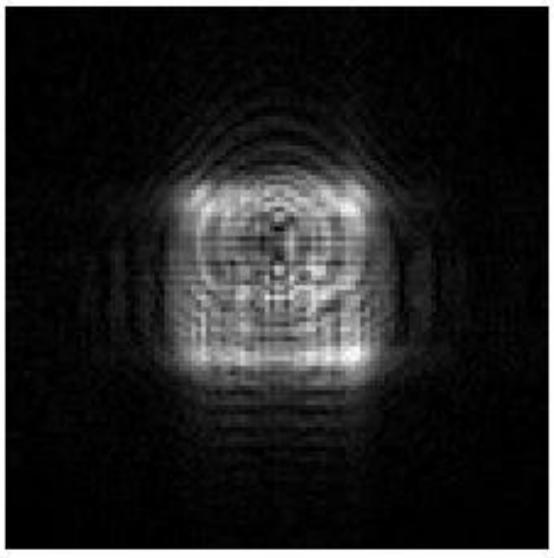

Figura 10.8: $FrFT$ de Tru.jpg con $a = 0{,}75$ en filas y columnas.

nuevamente la señal en su dominio directo afectada por el ruido aditivo tal como se muestra en la figura 10.17, luego se grafica la distribución de wigner y se observa el ruido en un dominio de Fourier fraccionario de la señal original, ver figura 10.18.

Figura 10.9: $FrFT$ de Tru.jpg con $a = 0{,}1$ en filas y $a = 0{,}9$ en las columnas.

Figura 10.10: Imagen onion de Matlab.

En la ecuación 7.12 se muestra una expresión con el fin de extraer y separar un ruido, el cual está inmerso en una señal de forma aditiva, este filtro permite reducir la probabilidad de equivocación al momento de la restauración de la señal, al utilizar la convolución fraccionaria con este filtro se extrae el ruido y la señal queda lo más

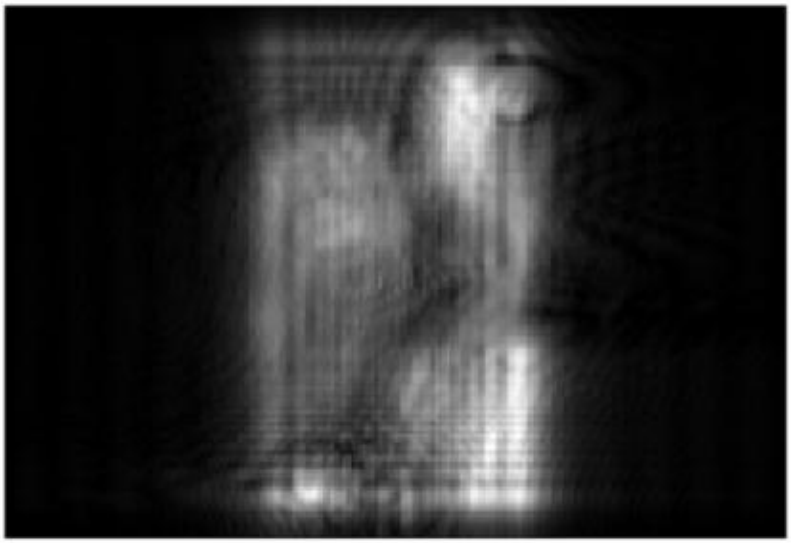

Figura 10.11: $FrFT$ de la imagen onion con $a = 0{,}25$ en la filas y $a = 0{,}75$ en las columnas.

Figura 10.12: imagen onion en su dominio directo, luego de haber estado en el dominio fraccionario.

parecidamente posible a la que se quiere recuperar, véase figura 10.20.

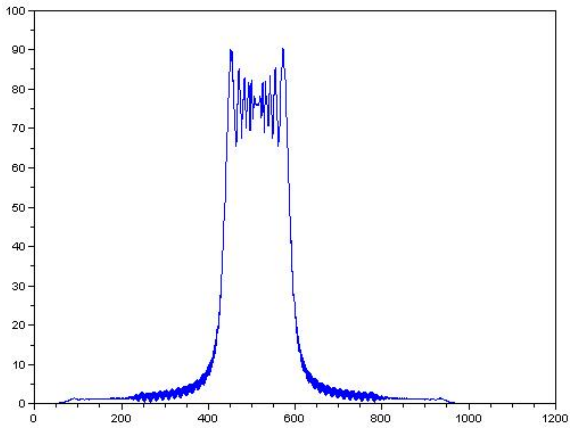

Figura 10.13: $FrFT$ con $a = -0{,}5$ de una función rectángulo de 1024 muestras.

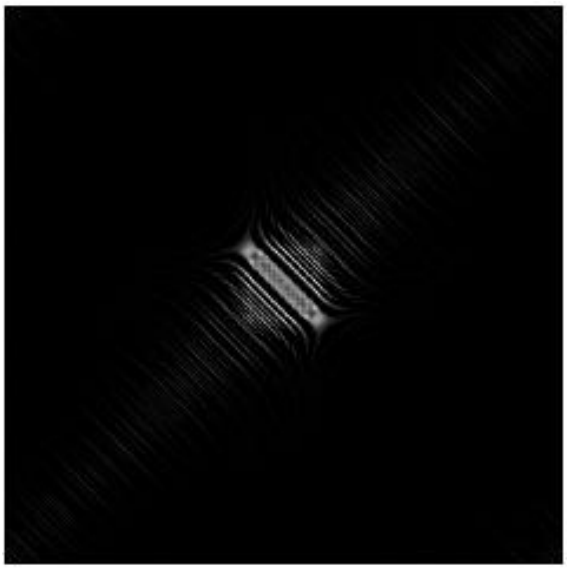

Figura 10.14: Distribución de Wigner de la $FrFT$ de una función rectángulo con $a = -{,}5$.

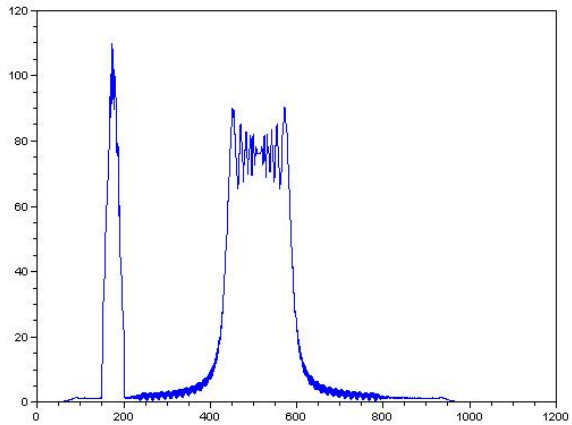

Figura 10.15: Módulo de la transformada de la señal + ruido aleatorio.

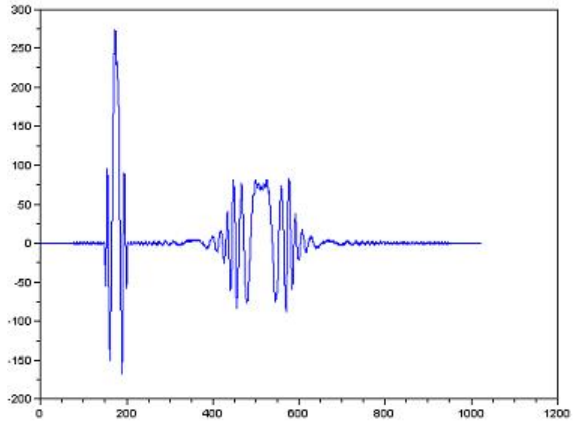

Figura 10.16: Valor real de transformada de la señal + ruido no estacionario.

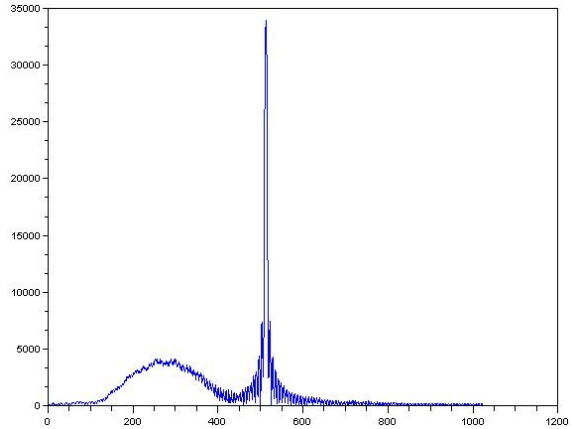

Figura 10.17: Señal + ruido en el dominio directo de la señal.

Figura 10.18: Distribución de Wigner de la señal en el dominio directo + ruido en el dominio fraccionario.

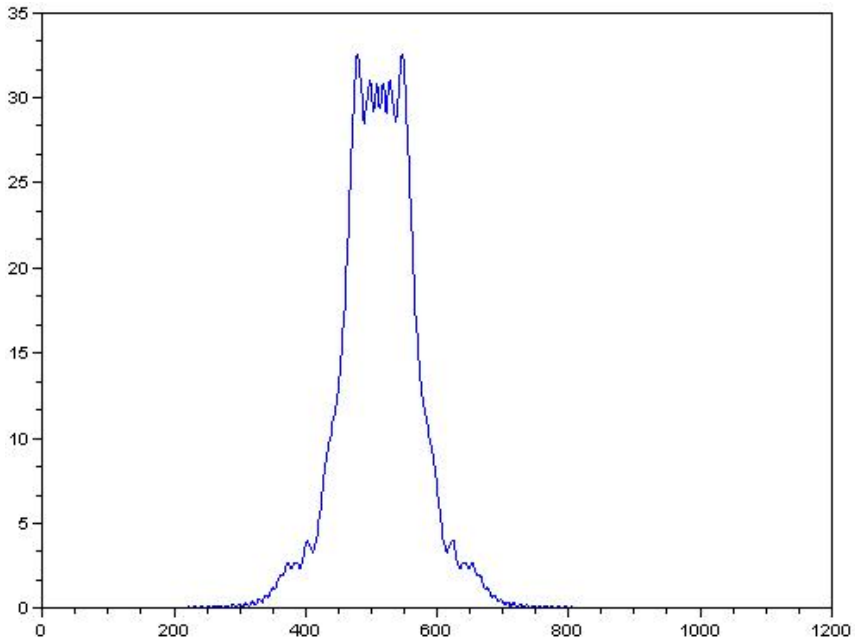

Figura 10.19: Filtro creado según la ecuación 7.12.

.

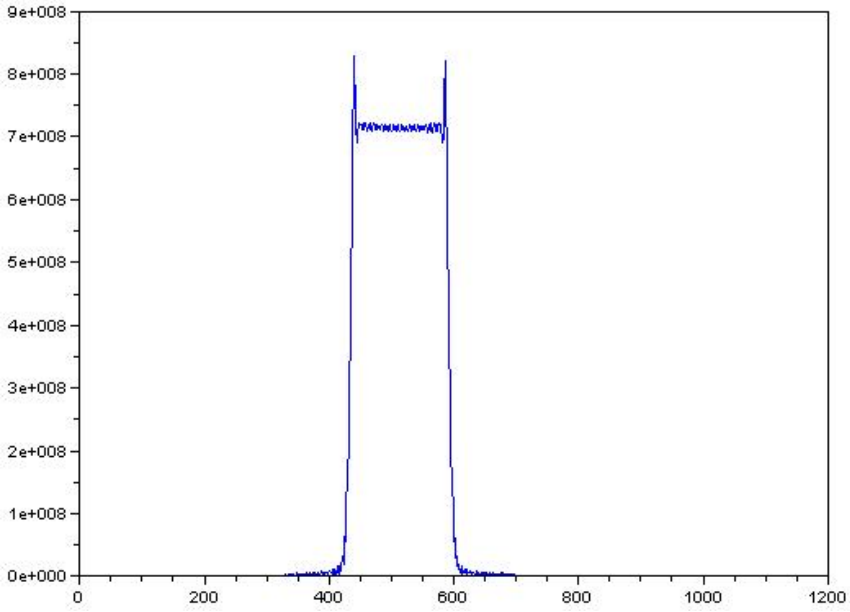

Figura 10.20: Luego del proceso de filtrado.

Capítulo 11

Limitaciones & inconvenientes

Con el fin de cumplir la atapa de análisis matemático de las ecuaciones que tenían relación con la Transformación de Fourier fraccionaria, se notaron deficiencias las cuales debieron ser sometidas a un rigoroso estudio para desarrollar las diferentes funciones de la aplicación realizada en esta investigación.

La incorporación de un nuevo editor de texto por medio de macros y/o funciones LaTeX [42], sugerido y requerido por el Director de la investigación, exhortando así el uso e implementación de aplicaciones de software libre, este fue creado por el matemático Leslie Lamport [43], por lo cual además de los programas utilizados para la aplicación en Scilab, se utilizó MikTeX y TeXnicCenter, con estos se pueden generar documentos de diferentes clases de archivos tales como .Pdf, .Dvi, .Ps y .Html por medio de un "lenguaje de programación ". - $I\,t_{hink}$ $LaTeX$ is $\mathfrak{f}^{u^n}$.

A pesar de implementarse diferentes soluciones para la Transformación de Fourier fraccionaria discreta, en los cuales aparece la expresión de la Transformación de Fourier la cual no fue posible implementar, debido a algunas incoherencias que se obtenían la fase de este algoritmo, ver figura 11.1. En algunas partes donde se men-

ciona este algoritmo se le llama $FrFT0$.

Por tal razón para la solución del algortimo se hace una operación matricial, entre una matriz $N \times N$ multiplicado por la función fc de tamaño N, la expresión de una operación matricial se muestra como en la ecuación 11.1, haciendo una sumatoria por cada uno de los elementos de las filas de la matriz por los elementos del vector columna. En el algoritmo propuesto esta operación puede llegar a ser tediosa para el tratamiento de medianas y grandes imágenes, donde puede llegar a durar más de una decena minutos para hallar el resultado de una imagen de altas magnitudes, ver tabla 11.1

$$\begin{pmatrix} w_{11} & w_{12} & \dots & a_{1n} \\ w_{21} & w_{22} & \dots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ w_{k1} & w_{k2} & \dots & a_{kn} \end{pmatrix} \begin{pmatrix} fc_1 \\ fc_2 \\ \vdots \\ fc_n \end{pmatrix} = \begin{pmatrix} Fy_1 \\ Fy_2 \\ \vdots \\ Fy_n \end{pmatrix} \tag{11.1}$$

No. de muestras	Dimensión	fracF	FrFT0	DFrFT
128	1	0	0	0.094
256	1	0.16	0.023	0.312
512	1	0.16	0.017	1.052
1024	1	0.18	0.016	4.657
128	2	1.265	1.219	20.937
256	2	2.813	2.828	177.984
512	2	6.672	7.141	1259.312
1024	2	17.515	19.843	6167.481

Tabla 11.1: Duración en segundos de transformaciones de Fourier fraccionarias a señales $1D$ y $2D$ de diferentes tamaños, con los algoritmos implementados

Algo similar ocurrió con la operación en integral de la convolución fraccionaria,

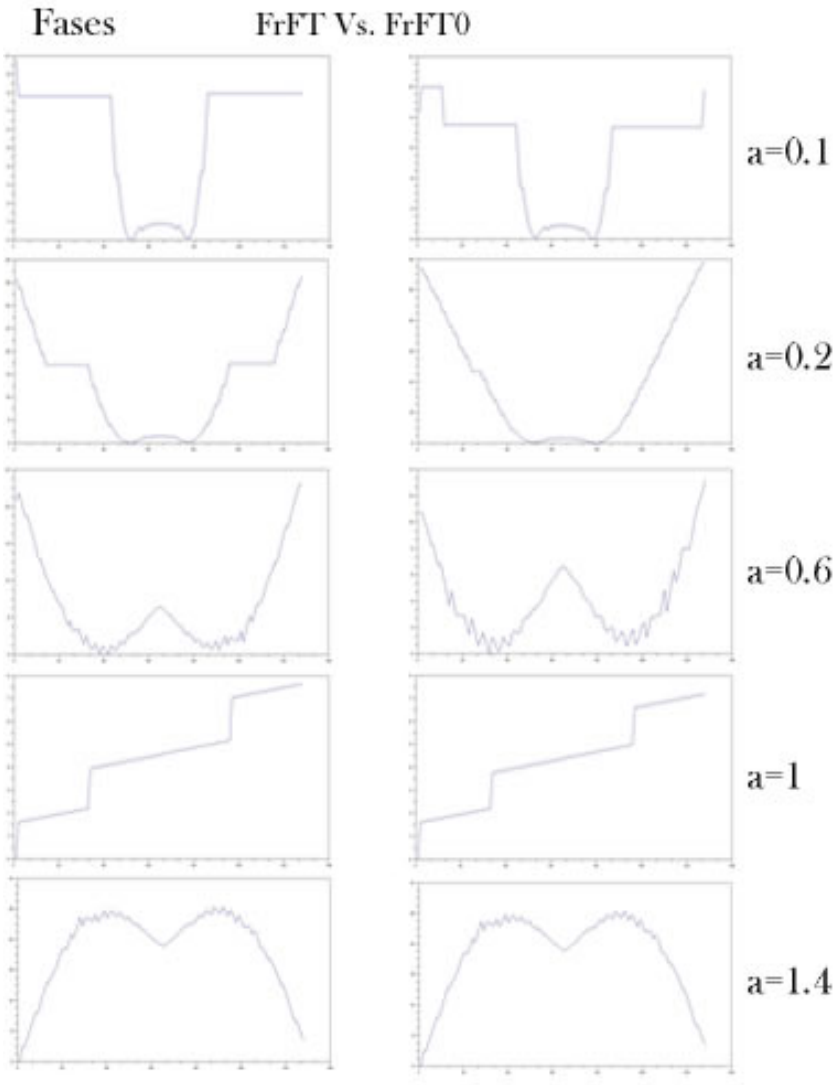

Figura 11.1: Valor de la fase de dos funciones de las $FrFT$ implementadas

mostrada en la ecuación 7.8, la cual fue resuelta en diferentes formas y se discretizaron todas estas funciones para la digitación de estos algoritmos en Scilab, pero los resultados que esta arrojaba no eran los esperados para la operación de convolución fraccionaria.

Se observa en la figura 11.2 que ambos algoritmos que realizan la operación de convolución fraccionaria obtienen el mismo resultado en valores absolutos, pero al graficar sus fases estas no corresponden y son muy diferentes, lo cual no permite seguir adelante en la realización de las función del filtro peopuesto en la sección 7.3.

Por tal motivo se implementa la ecuación 7.6 la cual resuelve la convolución fraccionaria por medio del teorema 3.1 utilizando la $FrFT$. Al hallar la Convolución fraccionaria por medio del algoritomo propuesto de la transformación de Fourier fraccionaria discreta y utilizarla según el teorema de la convolución no hay dudas que se obtendrá la convolución fraccionaria de dos funciones en un angulo $\alpha = a\pi/2$. La convolución fraccionaria es la operación por medio de la cual debe ser implementado el filtro de Wiener fraccionario y sin esta, este no puede obtenerse, ello llevó a la demora de la realización del Filtro pero al final se cumplieron todos los objetivos propuestos en la presente investigación.

El tamaño de memoria demandado por Scilab es dinamico realizado mediante las funciones `pwd, m=pwd(), y m=getcwd()`; por esto el número de muestras de las señal de entrada aceptado por los parámetros iniciales en scilab en la $FrFT$ es de 280 muestras y la función de entrada siempre debe ser una función par, por la discretización de la señal en el intervalo $(-N/2, N/2-1)$ 7.4, al realizar la petición

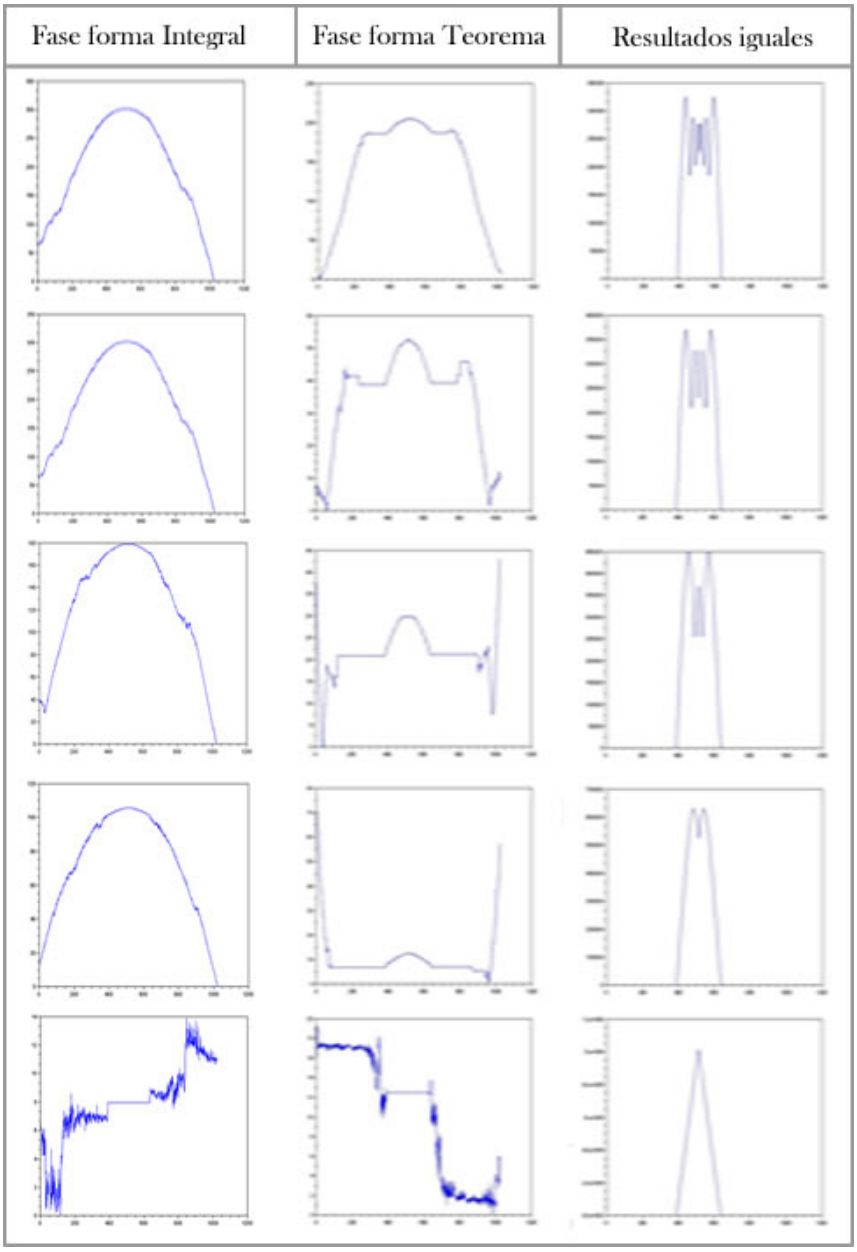

Figura 11.2: Valor de las fases de las funciones de convolución fraccionaria implementadas

de memoria por medio de la sentencia `stacksize(7e7)`; se aumenta el número de muestras que pueden ser analizadas por la $FrFT$ a un tamaño de 1044 muestras, sin embargo; la aplicación de esta función a imágenes medianas y grandes puede ser algo tedioso, ver tabla 11.1 ya que podría llegar a demorar mucho tiempo para hallar la transformada de la imagen, por tal razón se aconseja trabajar con imágenes pequeñas de tamaño igual o menor a 124 muestras para ambas dimensiones para que su proceso no sea tedioso.

Capítulo 12

Conclusiones

Las investigaciones relacionadas con áreas de las ciencias básicas resultan bastante interesante, por la retórica teoría que estas siempre presentan; más aún cuando se utilizan herramientas computacionales para realizar los cálculos de la investigación. El área de la óptica de Fourier ha estado avanzando desde su invención y seguirá avanzando aún más, gracias al poder de cálculo que se realizan con los computadores modernos. Al hacer uso de herramientas para el cálculo numérico, siempre hace falta más poder computacional para llegar a conclusiones más exactas, y nace la necesidad de realizar los cálculos sobre un ambiente distribuído, Scilab es una plataforma de cálculo numérico que ofrece estas posibilidades.

El estudio de señales no estacionarias en el campo científico siempre ha representado mayor preocupación, ya que muchos factores en áreas de biofísica, electrofísica, geofísica, sismología, entre muchas otras se generan aleatoriamente a lo largo del tiempo, no obstante, poder hacer una rotación en el espacio tiempo-frecuencia, y utilizar esta información para el análisis, estudio y tratamiento de señales son muchas de las ventajas que tiene la transformación de Fourier fraccionaria en el área del tratamiento de señales.

La implementación numérica de la $FrFT$ en Scilab hace posible la generación de nuevas operaciones y filtros como la convolución fraccionaria y el filtro de Wiener fraccionario, los cuales se implementan de tal forma que minimiza el error al momento de la separación de dos señales en el dominio de Fourier fraccionario, ya que la dependencia del filtro con el orden fraccionario a lo convierte en un filtro adaptativo [28]. Así, este tratamiento consigue un filtrado de una función no estacionaria aleatoria, debido a rotaciones que realiza la transformación de Fourier fraccionaria sobre la distribución de Wigner asociada a una señal, permitiendo extraer todo el ruido de manera adaptable en el dominio de Fourier fraccionario.

Bibliografía

[1] M. Perov y A. Yakimov. The Influence of Fast Fourier Transform on the signal-spectrum Estimation. Radiophysics and Quantum Electronics. 2002, Vol. 45, No.3, Pág. 239-245.

[2] W. Goodman. Introduction to Fourier Optics. Edit. McGraw-Hill. Segunda edición. 1996. 441 págs.

[3] R. Vio y W. Wamsteker. Joint Time-Frequency Analisis: a tool exploratory analysis and filtering of non-stationary time series. Abril de 2002. Articulo de Astronomía y AStrofísica.

[4] V. Namias. The Fractional Order Fourier Transform and its Applications to Quantum Mechanics, J. Inst. Maths. Applics. 1980 Vol. 25: 241-265.

[5] L. Almeida. The fractional Fourier transform and time-frequency representation. IEEE Trans. Sig. Proc. 1994. vol 42:3084-3091.

[6] D. Dragoman . Applications of the Wigner Distribution Function in signal Processing. EURASIP Hournal on Applied Signal Processing 2005 Hindawi Publishing Corporation. Vol. 10. Pág 1520-1534.

[7] R. Torres, Y. Torres y P. Pellat. Samplig Theorem for fractional bandlimited signals: A self-contained proof. application to digital holography. IEEE Sign. Proc. Lett 13 (11). 2006. Pág. 676-679.

[8] R. Torres, Y. Torres y P. Pellat. Sampling theorem in fractional Fourier domains. En "5^{th} Ibeoroamerican Metting on Optics and 8^{th} Latin American Metting on Optics, Lasers, ans their Applications "tomo 5622 de "Presented at the society of Photo-Optical Instrumentation Engineer (SPIE)". Pág. 1188-1192. A Marano, J. L. Paz (2004).

[9] L. Almeida. Product and convolution theorems for the fractional fourier transform. IEEE Signal Process. Lett. 4, 15-17. 1997.

[10] R. Torres, P. Pellat-Finet e Y. Torres. Fractional convolution, fractional correlation and their traslation invariance properties. Soumis *'a Signal Processing. An International Journal of the European Association for Signal Processing* (EURASIP) (2008).

[11] Z. Zalevsky and D. Mendlovic, "Fractional Wiener filter,". 1996. Appl. Opt. Vol. 35, Publicación 20. Pág. 3930-3936.

[12] R. Torres. Tratamiento de señales por Transformación de Fourier fraccionaria. aplicaciones a la holografía sintética y al filtrado óptico.[Tesis Doctoral] Agosto de 2008. Bucaramanga-Colombia. 134 pág.

[13] S. Mitra. Digital Signal Processing: A computer-Based Approach. Editorial Mc. Graw-Hill. Tercera Edición. Año 2006.

[14] A. Lara. Sobre la Transformación Tiempo-Frecuencia y la aplicación del proceso de Convolución a la dinámica de sistemas físicos. Revista de acústica ISSN 0210-3680. Vol. 38. No. 1-2, Julio-Agost de 2006. Pág. 7-13.

[15] R. Willett. Sampling Theory and Spline Interpolation. [Sitio Web]. Visitado el 24 de 2003 Disponible en: `http://cnx.org/content/m11126/2.3/`. Visitado el día 19 de Octubre de 2008.

[16] N. Brueller, N. Peterfreund y M. Porat. Non-stationary signals: optimal sampling and instantaneous bandwidth estimation. Proc. de la IEEE-SP International Symposium on Time-Frequency and Time-Scale Analysis, Pittsburgh, USA, 6-9 Oct 1998. Pág.113-115.

[17] Y. Cekic, A. Akan y L. Chaparro. A Discrete Fractional Gabor Expansion for Time-Frequency Signal Analysis. Istanbul University engineering Faculty, Journal of Electrical ámdpersanÉlectronics. 2002. Vol. 2. No. 2. Pág. 483-487.

[18] M. Greitans, Advanced Proccesing of Nonuniformly Sampled Non-Stationary Signals. Elektronika Ir Elektrotechnika. 2005. No. 3 Pág 42-45.

[19] J. Bouvrie y T. Ezzat. An Incremental Algorithm for Signal Reconstruction from Short-Time Fourier Transform Magnitude, pittsburg, USA. In INTERSPEECH-2006, paper 1691-Thu2BuP.9.

[20] H. Ozaktas y B. Billur. Convolution, filtering, and multiplexing in fractional Fourier domains and their relation to chirp and wavelet transform. Sociedad de Óptica de América. J. Opt. Soc. Am. A Vol. 11, No. 2/February 1994. Pág. 547-559.

[21] T. Alieva, M. Bastiaans, M^a. L. Calvo. Fractional Transform in optical Information Processing. J. Applied. 2005. Vol 10: 1498-1519.

[22] H. Ozaktas, Z. Zalevsky, et al. The Fractional Fourier Transform Applications in Optics and Signal Processing, 2001. John Wiley & Sons.

[23] R. Saxena y S. Kulbir. Fractional Fourier transform: A novel tool for signal processing. J. Indian Inst. Sci., ene-feb 2005, Vol 85: 11-26.

[24] N. Wiener. Hermitian polynomials and Fourier analysis. J. Math. Phys. (MIT), 8, 70-73, (1929).

[25] H. Chin. Iterative Wiener filters for image restoration. Sig. Proc IEEE. Vol. 39. Agosto de 1991 Pág. 1892 - 1899.

[26] Scilab Group (INRIA Meta2 Project/ENPC Cergrene). Introduction to Scilab: User's Guide, Rocquencourt(Francia): Sitio Web. `http://www-rocq.inria.fr/scilab`. 2007. 125 Pág. Visitado el día 8 de Agosto de 2008.

[27] Scilab Group. (INRIA Meta2 Project/ENPC Cergrene). Signal Proccesing With SciLab. Rocquencourt(Francia): Sitio Web: `http://www.scilab.org/product/index-product.php?page=old-documentation`; 2007. 205 Pág. Visitado el día 22 de Septiembre de 2008.

[28] K. Tsai-Sheng. Wiener Filter as an Optimal MMSE Interpolator. Taipei, Taiwan PWASET Vol. 24 Octubre de 2007 ISSN 1307-6884. 216-218.

[29] A. Bultheel y H. Martínez. Computation of the Fractional Fourier Transform. Dept. of Computer Science, Celestijnenlaan 200A, B-3001 Leuven.

[30] D. Bailey. The Fractional Fourier Transform and applications; SIAM Review 1995. Vol. 33: Pág. 389-404.

[31] J. Cruel. A Scilab Image Processing User Manual. Versión 0.3 (rev. 1). 2004. Armentires-Francia. 30 Pág.

[32] I. DJurovic, S. Stankovic y I. Pitas. Digital watermarking in the fractional Fourier transformation domain. Journal of Network and Computer Applications. 2001. Vol. 24. Pág. 167-173.

[33] X. Yang et al. Improved fast fractional Fourier Transform Algorithm. Journal Opt. Soc. Am. A. Vol 21. No. 9./Septiembre 2004.

[34] D. Mustard. Lie group imbeddings of the Fourier transform. School of Mathematics Preprint. A987. AM87/13, The university of New South Wales, Kensington, Australia.

[35] G. Cariolaro, T. Erseghe, P. Kraniaskas y N. Laurenti. Multiplicity of fractional Fourier Transforms ans their relationships. IEEE Sig. Proc. Vol 48. No. 1. Enero de 2000. Pág. 227-241.

[36] P. Pellat. Lecciones de óptica de Fourier. División Editorial y de Publicaciones Universidad Industrial de Santander (UIS). Bucaramanga. (2004).

[37] Scilab Group (INRIA Meta2 Project/ENPC Cergrene). Scilab References Manual: Online Documentation. Rocquencourt(Francia): Sitio Web `http://www-rocq.inria.fr/scilab`; 2007. 649 pág. Visitado el día 30 de Agosto de 2008.

[38] O. Arikan, M. Kutay, H. Özaktas, Ö. Akdemir. The Discrete Fractional fourier Transformation. 1996. TFTS. Pág. 205-207.

[39] I. Djurovi, S. Stankovic y I. Pitas . Digital watermaking in the fractional Fourier transformation domain, 2001. Journal of Network and Computer Applications Vol. 24, Pág. 167-173.

[40] M. Kutay. fracF: Fast Computation of the Fractional Fourier Transform, 1996; `http://www.ee.bilkent.edu.tr/~haldun/fracF.m`.

[41] H. Ozaktas, M. Kutay y G. Bozdagil. Digital computation of the fractional Fourier transform. IEEE Trans. Signal Process. 44 (1996) 2141-2150.

[42] T. Oetiker, H. Partl, I. Hyna y E. Schlegl. The Not So Short Introduction to LaTeX 2_ε. Or LaTeX 2_ε in 87 minutes. Version 3.2, Free Software Foundation, Inc. 675 Mass Ave, Cambridge, MA 02139, USA. September, 1998.

[43] L. Lamport. LaTeXA Document Preparation System. Addison-Wesley, California 1986.

[44] H. Emre. Signal recovery from partial fractional Fourier domain information and pulse design iterative projections. [Tesis de Maestría]. Departamento de Ingeniería Eléctrica y Electrónica, Instituto de Ingenierías y Ciencias, Universidad de Bilkent. Julio de 2005.

[45] P. Pellat y Y. Torres. Image formation with coherent light: The fractional Fourier approach. Journal of Modern Optics. Agosto de 1997. Vol. 44. Pág. 1581-594.

[46] A. Bultheel y H. Martínez. A shattered survey of the Fractional Fourier Transform. Department of Computer Science, K.U.Leuven. Bélgica. Reporte TW337, Abril de 2002.

[47] Yang et al. Improved fast fractional-Fourier-transform algorithm. Vol. 21, No. 9. Septiembre 2004. J. Opt. Soc. Am. A Pág. 1677-1681

[48] G. Cariolaro, T. erseghe, P. Kraniauskas y N. Laurenti. A Unified Framework for the Fractional Fourier Transform. Departamento de Electrónica & informática, Universidad de Padova. Italia. 19 Pág.

[49] A. Bultheel y A. Martínez H. An introducction to the Fractional Fourier Transform and friends. Preprint, February 25. Belgian.

[50] J. García, D. Mas y R. Dorsch. Fractional Fourier transform calculation throuh the fast Fourier transform algorithm. Revista Sociedad öptica de América. Diciembre de 1996. Vol 35. No. 35. Applied optics. Pág. 7013-7018.

[51] D. Mustard. Fractional Convoluction. J. Austral. Math. Soc. Ser. B 40(1998). Pag. 257-265.