LA TRANSFORMADA DE FOURIER Y SUS APLICACIONES EN LA FÍSICA

J. E. Loera,[1] C. E. Valladares[1] E. A. Leija[1]

RESUMEN

En este artículo se estudian las series y transformadas de Fourier y su aplicación en la Física, específicamente nos centraremos en la teoría del caos, el atractor de Lorentz y en el problema de la partícula libre cuántica no-relativista, utilizando diversos algoritmos enfocados en la obtención de la transformada de Fourier mediante métodos matemáticos. Entre ellos tenemos la Transformada Discreta de Fourier o DFT (Discrete Fourier Transform) y la Transformada Rápida de Fourier, mejor conocida por la abreviatura FFT (Fast Fourier Transform) el cual es considerado como uno de los mejores diez algoritmos desarrollados durante el siglo XX.

ABSTRACT

In this article we study the Fourier series and transforms and their application in Physics, specifically we will focus on chaos theory, the Lorentz attractor and the problem of the non-relativistic quantum free particle, using various algorithms focused on the obtaining the Fourier transform with some mathematical methods. Among them we have the Discrete Fourier Transform or DFT and the Fast Fourier Transform, better known by the abbreviation FFT which is considered one of the best ten algorithms developed during the 20th century.

Key Words: DFT — FFT — Transformada de Fourier — Espectro — Algoritmo

1. INTRODUCCIÓN

Conforme se avanza en el estudio de la física, llega un momento donde es inevitable encontrarse con la Transformada de Fourier para la simplificación y solución de problemas, es una herramienta muy útil dentro de la física. Sus aplicaciones van desde el análisis armónico de señales periódicas hasta la solución de ecuaciones diferenciales y la descripción de fenómenos ondulatorios en mecánica clásica, electrodinámica, mecánica cuántica y muchas más.

En este trabajo la estudiaremos e implementaremos computacionalmente para resolver problemas de la física.

2. LA TRANSFORMADA DE FOURIER

Las transformadas integrales como lo son la transformada de Laplace o la transformada de Fourier, tienen una gran aplicación en muchas ramas de la ciencia, en este trabajo nos centraremos en la transformada de Fourier que puede ser usada para simplificar complejos problemas matemáticos. De manera poco rigurosa podemos decir que nos lleva una función del espacio en el que está definida al espacio de las frecuencias.

[1]Universidad Autónoma de Nuevo León, Facultad de Ciencias Físico Matemáticas, San Nicolás de los Garza, Nuevo león, México

Sea f(x) una función que satisface las condiciones de Dirichlet, entoces se dice que tiene una representación en series de Fourier.

Teorema 2.1 (Teorema de Dirichlet) *Si se cumple que:*

1. *Si $f(x)$ tiene periodo $2L$ y es univaluada entre $-L$ y L.*
2. *Tiene un número finito de máximos y mínimos.*
3. *Tiene un número finito de discontinuidades.*
4. *Y si $\int_{-L}^{L} \|f(x)\|dx$ es finita.*

Entonces f(x) tiene representación en series de Fourier y converge a ella en todos los puntos donde $f(x)$ es continua y al punto medio donde $f(x)$ es discontinua.

Definición 2.1 (Series de Fourier) *Si $f(x)$ satisface las condiciones de Dirichlet entonces su representación en series de Fourier está dada por:*

$$f(x) = \sum_{n=-\infty}^{\infty} c_n e^{i(n\pi x/L)} \tag{1}$$

Donde los coeficientes $\{c_n\}$ suelen ser llamados como el *espectro de f(x)* .

El espectro anterior es un espectro discreto que puede representar ciertas funciones periodicas, sin embargo, existen otras funciones periodicas o ciertos fenomenos físicos en los que se necesita de un espectro continuo. Notemos como la serie de Fourier está dada por una suma infinita, por lo cual no debería sorprendernos que bajo ciertas condiciones podamos reemplazarla por la integral de Fourier.

Definición 2.2 (La transformada de Fourier) *Si una función $f(x)$ satisface las condiciones de Dirichlet y si $\int_{-\infty}^{\infty} \|f(x)\|dx$ es finita entonces:*

$$F(k) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(x)e^{ikx}dx \tag{2}$$

Y la transformada inversa de Fourier está dada por:

$$f(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} F(k)e^{-ikx}dk \tag{3}$$

3. LA TRANSFORMADA DE DISCRETA DE FOURIER DFT

Notemos que una transformada de Fourier implica la evaluación de integrales y cierto manejo del álgebra, sin embargo, es posible implementar un algoritmo que nos permita encontrar la transformada de Fourier a un conjunto de datos discreto que contenga la información de la función a transformar. A este algortimo le llamamos la *Transformada Discreta de Fourier o DFT.*

Definición 3.1 (DFT) *Sea X_n un arreglo con N terminos que contiene la información de una función, la Transformada Discreta de Fourier (DFT) de cada elemento del arreglo numérico está por:*

$$X_k = \sum_{n=1}^{N-1} x_n e^{-i2\pi kn/N} \tag{4}$$

Y la Transformada Inversa Discreta de Fourier (tambíen denotada por DFT inversa o IDFT) está dada por:

$$x_n = \frac{1}{N} \sum_{n=1}^{N-1} X_k e^{i2\pi kn/N} \tag{5}$$

Resaltamos que tanto x_n como X_k pueden tomar valores complejos.

4. LA TRANSFORMADA RAPIDA DE FOURIER DFT

Por lo general, tenemos que lidiar con una gran cantidad de puntos de datos, y la velocidad del algoritmo para la transformada de Fourier se convierte en un tema muy importante. El algoritmo de la DFT tiene un costo computacional muy grande, conforme crece el tamaño del conjunto de datos, de hecho su complejidad es de $O(n^2)$ (donde n es el tamaño del conjunto de datos) .

La Transformada Rápida de Fourier (FFT) es un algoritmo eficiente para calcular la DFT de una secuencia. Se describe por primera vez en el artículo clásico de Cooley y Tukey en 1965, aunque se hizo referencia a este por primera vez en un trabajo no publicado de Gauss en 1805. Este algoritmo reduce considerablemente los costos computacionales pues su complejidad es de $O(nlog(n))$ (donde n es el tamaño del conjunto de datos) .

Este algoritmo se aprovecha de varias simetrías a la hora de calcular la DFT, y será implementado en Python para el desarrollo de este trabajo.

5. EL ATRACTOR DE LORENTZ

Uno de los sistemas más icónicos cuando se empieza el estudio de la teoría del caos es el de las *ecuaciones de Lorenz* .

Definición 5.1 (Ecuaciones de Lorenz)

$$\dot{x} = \sigma(y - x) \tag{6}$$

$$\dot{y} = x(\rho - z) - y \tag{7}$$

$$\dot{z} = xy - \beta z \tag{8}$$

El atractor de Lorenz es un concepto introducido por Edward Lorenz en 1963. Se trata de un sistema dinámico determinista tridimensional no lineal derivado de las ecuaciones simplificadas de rollos de convección que se producen en las ecuaciones dinámicas de la atmósfera terrestre.

Al parametro σ se le suele denominar el número de Prandtl y ρ se llama el número de Rayleigh.

Fijemos los parametros $\sigma = 10$ y $\beta = 8/3$ y variemos el número de Rayleigh entre 1 y 100 para hacer una inspección sobre de la sensibilidad a las condiciones iniciales del atractor de Lorenz. Emplearemos la siguiente metodología:

1. Resolver numéricamente el sistema de ecuaciones diferenciales.

2. Aplicar la transformada de Fourier a la solución de la ecuación diferencial.

3. Analizar gráficamente el espectro y la fase.

Hemos encontrado algunos puntos donde la sensibilidad a las condiciones iniciales de hace muy presente y estos son algunos de ellos:

1. $\rho \in (14.5500, 14.5599)$
2. $\rho \in (24.97, 24.98)$
3. $\rho \in (27.08, 27.09)$

En cada uno de estos cambios se observó una considerable variación en el comportamiento del sistema, ante minimas variaciones del número de Rayleigh, hay otros casos en los cuales también se observan perturbaciones y estos son solo algunos de dichos casos.

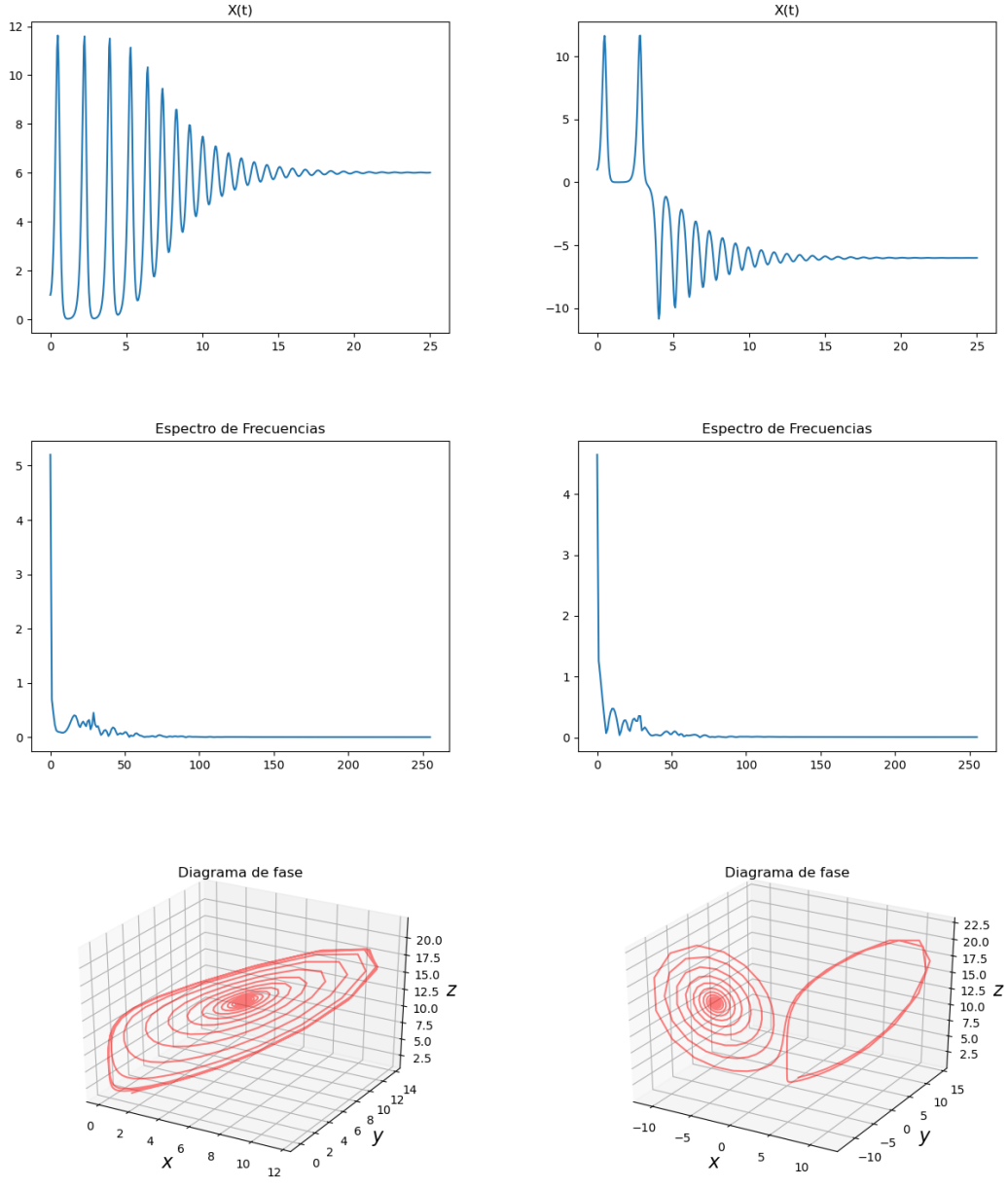

Fig. 1. $\rho = 14.5500 \Longrightarrow \rho = 14.5899$

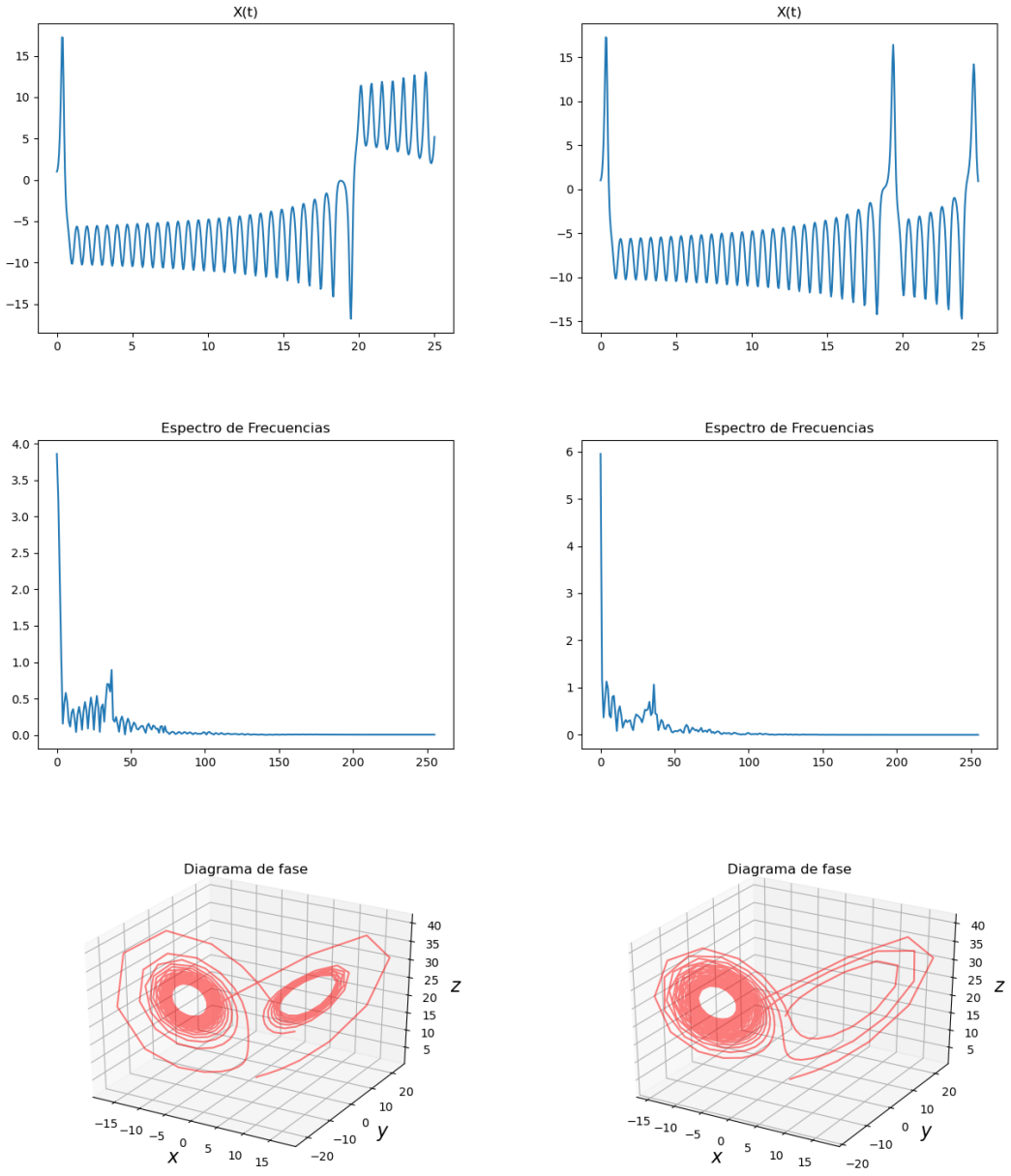

Fig. 2. $\rho = 24.97 \Longrightarrow \rho = 24.98$

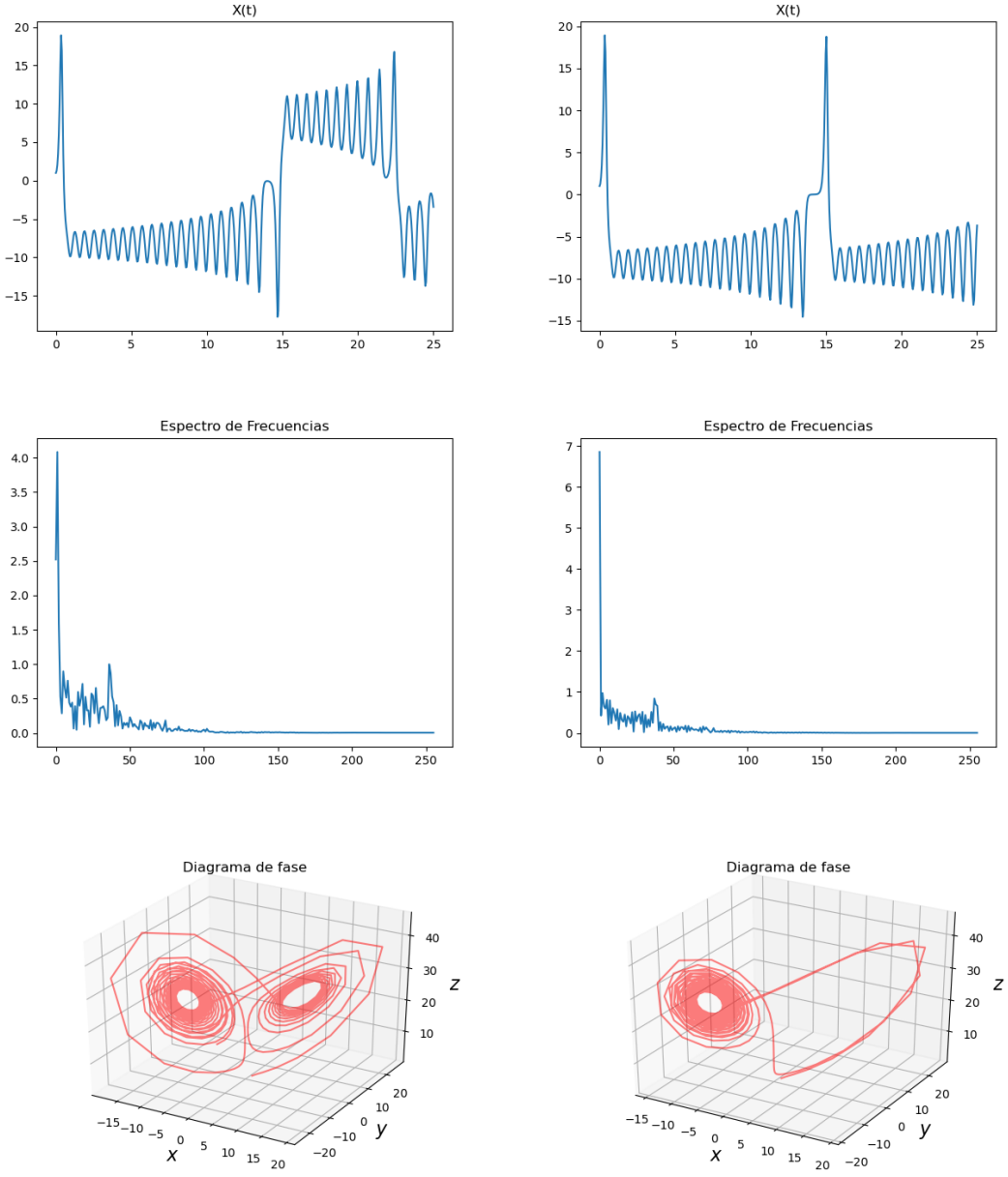

Fig. 3. $\rho = 27.08 \Longrightarrow \rho = 27.09$

6. LA PARTÍCULA LIBRE

En el contexto de la mecánica cuántica, las partículas cuánticas muestran características tanto de ondas como de partículas (clásicamente hablando). Es por eso que es necesario implementar un esquema matemático adecuado que englobe ambas características de forma simultánea y satisfactoria.
En la física clásica, una partícula está bien localizada en el espacio, dado que su posición y velocidad pueden ser calculadas simultáneamente con precisión arbitraria. Mientras que en la mecánica cuántica, una partícula se describe mediante una función de onda correspondiente a la onda de materia asociada con la partícula *(Hipótesis de De Broglie)*. Dada la situación, una partícula que sea localizable en una cierta región del espacio puede ser descrita por una onda de materia cuya amplitud sea grande en esa región y cero fuera de ella. De esta manera, la onda de materia será localizada dentro de la región del espacio donde la partícula esté físicamente confinada.
Una función de onda localizable es llamada un *paquete de ondas*. Un paquete de ondas consiste entonces de una superposición de un grupo de ondas de ligeramente distintas longitudes de onda, con fases y amplitudes tales que interfieran constructivamente dentro de una pequeña región del espacio y destructivamente fuera de ella.

Matemáticamente es posible llevar a cabo dicha superposición de ondas gracias a la *Transformada de Fourier*. Por simplicidad, se considerará un paquete de ondas unidimensional con la intención de describir a una partícula "clásica" confinada a una región unidimensional.
Es posible construir $\Psi(x,t)$ superponiendo ondas planas viajeras de distintas frecuencias (o longitudes de onda):

$$\Psi(x,t) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \phi(k) e^{i(kx-\omega t)}\, dk \qquad (9)$$

Donde $\phi(k)$ es la amplitud del paquete de ondas.
Es importante tomar en cuenta que ω no es una constante dentro de la integral, de hecho ω es una función de k, i.e. $\omega = \omega(k)$.
De las relaciones de De Broglie:

$$\omega = \frac{E}{\hbar} = \frac{p^2/2m}{\hbar} = \frac{(\hbar k)^2}{2m\hbar} = \frac{\hbar k^2}{2m}$$

El objetivo es hallar la evolución temporal de este paquete de ondas. Para lograr esto, se analiza primero la forma del paquete en un tiempo dado. Por decir, en $t = 0$:

$$\Psi(0,t) = \psi_0(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \phi(k) e^{ikx}\, dk \qquad (10)$$

De lo revisado en la *sección 2*, se pueden hallar las amplitudes $\phi(k)$ si se aplica una transformada de Fourier a $\psi_0(x)$:

$$\phi(k) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \psi_0(x) e^{-ikx}\, dx \qquad (11)$$

De esta manera, si se conoce la forma inicial de la función de onda, será posible encontrar las amplitudes $\phi(k)$ con (11), y finalmente la función de onda con su dependencia temporal en (9). Más formalmente se puede resumir este algoritmo como trasladar la función de onda $\psi_0(x)$ del *espacio de posiciones* a su representación $\phi(k)$ en el *espacio-k*, también conocido como *espacio de momentos*, para finalmente volver al *espacio-x* agregando la dependencia temporal de la función de onda. Cabe destacar que tanto $\psi_0(x)$ como $\phi(k)$ contienen la misma información y son descripciones igualmente válidas para un estado cuántico.

Considérese por ejemplo a una partícula libre con la función de onda inicial:

$$\Psi(x,0) = Ae^{-}ax^2 \qquad (12)$$

Donde A y a son constantes (a es real y positiva).

Lo primero que se busca hacer es normalizar la función de onda para que su modulo al cuadrado adquiera un sentido estadístico apropiado de acuerdo con la interpretación de Born.

$$\begin{aligned}\int_{-\infty}^{\infty} |\Psi(x,0)|^2\, dx &= |A|^2 \int_{-\infty}^{\infty} e^{-2ax^2}\, dx \\ &= 2|A|^2 \int_{0}^{\infty} e^{-2ax^2}\, dx\end{aligned}$$

Efectuando cambios de variable apropiados:

$$\begin{aligned}\int_{-\infty}^{\infty} |\Psi(x,0)|^2\, dx &= \frac{1}{\sqrt{2a}} |A|^2 \int_{0}^{\infty} t^{-1/2} e^{-t}\, dt \\ &= \frac{1}{\sqrt{2a}} |A|^2\, \Gamma(\frac{1}{2}) = \sqrt{\frac{\pi}{2a}} |A|^2\end{aligned}$$

De la condición de normalización:

$$\int_{-\infty}^{\infty} |\Psi(x,0)|^2\, dx = \sqrt{\frac{\pi}{2a}}|A|^2 = 1$$
$$\therefore A = \left(\frac{2a}{\pi}\right)^{1/4}$$

Una vez normalizada la función de onda inicial $\Psi(x,0) = \left(\frac{2a}{\pi}\right)^{1/4} e^{-ax^2}$, se obtiene $\phi(k)$ con la Transformada de Fourier según la ec. (11):

$$\begin{aligned}\phi(k) &= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \psi_0(x) e^{-ikx}\, dx \\ &= \left(\frac{a}{2\pi^3}\right)^{1/4} \int_{-\infty}^{\infty} e^{-(ax^2+ikx)}\, dx\end{aligned}$$

Con el apoyo de la librería *Scipy* utilizando la función de transformada de Fourier, se obtiene el resultado:

$$\phi(k) = \frac{1}{(2\pi a)^{1/4}} e^{-\frac{k^2}{4a}} \tag{13}$$

Luego, de la ec. (9):

$$\begin{aligned}\Psi(x,t) &= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \phi(k) e^{i(kx - \frac{\hbar k^2}{2m}t)}\, dk \\ &= \frac{1}{(8\pi^3 a)^{1/4}} \int_{-\infty}^{\infty} e^{-\left[\left(\frac{1}{4a} + \frac{i\hbar}{2m}t\right)k^2 - ixk\right]}\, dk\end{aligned}$$

Evidentemente esta es una integral algo laboriosa de hacer, así que se recurre una vez más a la librería *Scipy*, utilizando esta vez la función de cálculo de integrales.
Se obtiene así la expresión general para la función de onda:

$$\Psi(x,t) = \left(\frac{2a}{\pi}\right)^{1/4} \frac{1}{\sqrt{1 + \frac{2i\hbar a}{m}t}} e^{-\frac{ax^2}{1+\frac{2i\hbar a}{m}t}}$$

O de forma más simplificada haciendo $\theta = \frac{2\hbar a}{m}t$:

$$\Psi(x,t) = \left(\frac{2a}{\pi}\right)^{1/4} \frac{1}{\sqrt{1+i\theta}} e^{-\frac{ax^2}{1+i\theta}} \tag{14}$$

Apéndice A

Método en python con el algoritmo de la DFT

```
def DFT(F, N):
Im_F = F[0]
Re_F = F[1]
Im_G = []
Re_G = []
G = []
Freq = []

for k in range(N):
im_g = 0
re_g = 0
for n in range(N):
re_g = (
re_g
+ np.cos((2 * np.pi * k * n) / (N)) * Re_F[n]
+ np.sin((2 * np.pi * k * n) / (N)) * Im_F[n]
)
im_g = (
im_g
+ np.cos((2 * np.pi * k * n) / (N)) * Im_F[n]
- np.sin((2 * np.pi * k * n) / (N)) * Re_F[n]
)
Im_G = np.append(Im_G, im_g)
Re_G = np.append(Re_G, re_g)
G = np.append(G, [re_g, im_g])
Freq = np.append(Freq, k)

return G, Re_G, Im_G, Freq
```

Apéndice B

Método en python con el algoritmo de la FFT

```
def FFT(f):
N = len(f)
if N <= 1:
return f

# division
even = FFT(f[0::2])
odd = FFT(f[1::2])

# store combination of results
G = np.zeros(N).astype(np.complex64)

# only required to compute for half the frequencies
# since u+N/2 can be obtained from the symmetry property
for u in range(N // 2):
G[u] = even[u] + np.exp(-2j * np.pi * u / N) * odd[u]  # conquer
G[u + N // 2] = even[u] - np.exp(-2j * np.pi * u / N) * odd[u]  # conquer

return G
```

Apéndice C

Código donde se simuló el atractor de Lorenz

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import odeint
from Methods import FFT

"""
El atractor de Lorentz
"""


def Lorentz_ED(xyz, t, sigma, rho, beta):
x, y, z = xyz
return [sigma * (y - x), x * (rho - z) - y, x * y - beta * z]


"""
Discretizamos la variable dependiente en N=2**m terminos,
el cual es un requerimento de la Transformada Rapida de Fourier
"""
m = 9
N = 2 ** m


"""
Condiciones iniciales y parametros del Atractor de Lorentz
"""

# Asignamos valores a los parametros
sigma, Rho, beta = 10, np.linspace(24, 25, 100), 8.0 / 3.0

# Condicion inicial y valores de t sobre los que calcular
xyz0 = [1.0, 1.0, 1.0]
xi = 0.0
xf = 25.0
t = np.linspace(xi, xf, N)
iter = 1

for rho in Rho:

# Resolvemos las ecuaciones
xyz = odeint(Lorentz_ED, xyz0, t, args=(sigma, rho, beta))

# Graficamos las soluciones
from mpl_toolkits.mplot3d.axes3d import Axes3D

fig, (ax1) = plt.subplots(1, 1, subplot_kw={"projection": "3d"})
ax1.plot(xyz[:, 0], xyz[:, 1], xyz[:, 2], "r", alpha=0.5)
ax1.set_xlabel("$x$", fontsize=16)
ax1.set_ylabel("$y$", fontsize=16)
ax1.set_zlabel("$z$", fontsize=16)
plt.title("Diagrama de fase")
plt.savefig("images_numpy_24_25/fase/fase" + str(iter) + ".png")
plt.close()

"""
Calculamos la FFT de X(t) de las ecns de Lorentz
"""
G_Lorentz_X = FFT(xyz[:, 0])

"""
Creamos un arreglo con los espectros de potencia de la FFT
"""
S_fft = []
for i in range(N):
s = (1.0 / N) * abs(G_Lorentz_X)
S_fft = np.append(S_fft, s)
```

```
    """
    Graficamos la funcion Fx_fft
    """
    plt.plot(t, xyz[:, 0])
    plt.title("X(t)")
    plt.savefig("images_numpy_24_25/eje_x/eje_x" + str(iter) + ".png")
    plt.close()

    """"
    Ahora Graficamos S vs frecuencia (Espectro)
    """
    Freq = np.arange(0, N // 2)
    plt.plot(Freq, S_fft[: N // 2])
    plt.title("Espectro de Frecuencias")
    plt.savefig("images_numpy_24_25/espectro/espectro" + str(iter) + ".png")
    plt.close()

    iter = iter + 1
```

REFERENCES

Scherer J. (2010) Computational Physics
Boas M. (2006) Mathematical methods in the physical sciences
Butkov E. (1988) Física Matemática
Stickler, D. (2016) Basic concepts in computational physics
Pang T. (1999) An introduction to computational physics