

Universidad Autónoma de Chiapas  
Facultad de Física y Matemáticas  
Física Computacional  
Tarea 2  
Series De Tiempo

---

Carlos Castillo García

Licenciatura en Física

6° B

---

Dr. Roberto Arceo Reyes.  
03 de Mayo del 2025.

# 1. Introducción

Este trabajo se aborda un problema relacionado con el análisis de series de tiempo mediante el uso de herramientas computacionales. Se parte de una señal generada de manera artificial, que combina una función de tipo gaussiano con una onda cosenoidal modulada. Esta señal busca representar un fenómeno que cambia con el tiempo, permitiendo estudiar sus características.

A continuación, se utiliza la Transformada Rápida de Fourier (FFT) para transformar la señal desde el dominio del tiempo hacia el dominio de la frecuencia. Esta técnica permite identificar las frecuencias que componen la señal original, lo cual es especialmente útil para detectar ciclos repetitivos, patrones periódicos o componentes predominantes dentro de la serie.

Haciendo uso de programación en Fortran, creamos un código que nos pueda resolver nuestro problema, para que en seguida nos pueda dar una serie de datos.

Finalmente, se obtiene una representación gráfica del espectro de frecuencias, donde se visualiza la intensidad o energía presente en cada frecuencia, facilitando la interpretación y el análisis de la dinámica temporal del fenómeno simulado.

# 2. Código de Fortran

```
common /time/ f
complex f(0:10000), ai, q, g(0:127), c, c2
pi = acos(-1.0)
ai = (0.0, 1.0)
n = 128
delta = 0.4
q = cos(2.0 * pi/n) + ai * sin(2.0 * pi/n)
w = (2.0 * pi/n) / delta

do j = 1, n
  t = (j - n/2) * delta
  th1 = j * pi * (real(n - 1)/real(n))
  c = cos(th1) + ai * sin(th1)
  f(j) = c * (exp(-4.0 * t**2.0) + 0.1 * cos(5.0 * t))
enddo

if (n .eq. 4) then
  call four(q, g, 1, 0)
else if (n .eq. 8) then
  call eight(q, g, 1, 0)
```

```

else if (n .eq. 16) then
call sixteen(q, g, 1, 0)
else if (n .eq. 32) then
call thirtytwo(q, g, 1, 0)
else if (n .eq. 64) then
call sixtyfour(q, g, 1, 0)
else if (n .eq. 128) then call onetwentyeight(q, g, 1, 0)
else
print *, 'Valor de n no soportado.'
stop
endif

```

```

do j = 0, n-1
th2 = (j + 1) * pi * (real(n - 1)/real(n))
c2 = cos(th2) + ai * sin(th2)
g(j) = g(j) * c2
print *, j, w * (j - n/2), abs(g(j))
enddo

```

```

stop
end

```

```

subroutine four(t, x, k, i)
common /time/ f
complex t, f(0:10000), x(0:3), s1, s2, s3, s4
s1 = f(i) + f(i + 2*k)
s2 = f(i + k) + f(i + 3*k)
s3 = f(i) - f(i + 2*k)
s4 = f(i + k) - f(i + 3*k)
x(0) = s1 + s2
x(1) = s3 + t * s4
x(2) = s1 - s2
x(3) = s3 - t * s4
return
end

```

```

subroutine eight(q, g, k, i)
common /time/ f
complex q, g(0:7), x(0:3), y(0:3), t, f(0:10000)

```

```

call four(q**2, x, 2*k, i)
call four(q**2, y, 2*k, i + k)
t = 1.0
do j = 0, 3
g(j) = x(j) + t * y(j)
g(j + 4) = x(j) - t * y(j)
t = t * q enddo
return
end

```

```

subroutine sixteen(q, g, k, i)
common /time/ f
complex q, g(0:15), x(0:7), y(0:7), t, f(0:10000)
call eight(q**2, x, 2*k, i)
call eight(q**2, y, 2*k, i + k)
t = 1.0
do j = 0, 7
g(j) = x(j) + t * y(j)
g(j + 8) = x(j) - t * y(j)
t = t * q
enddo
return
end

```

```

subroutine thirtytwo(q, g, k, i)
common /time/ f
complex q, g(0:31), x(0:15), y(0:15), t, f(0:10000)
call sixteen(q**2, x, 2*k, i)
call sixteen(q**2, y, 2*k, i + k)
t = 1.0
do j = 0, 15
g(j) = x(j) + t * y(j)
g(j + 16) = x(j) - t * y(j)
t = t * q
enddo
return
end

```

```

subroutine sixtyfour(q, g, k, i)
common /time/ f

```

```

complex q, g(0:63), x(0:31), y(0:31), t, f(0:10000)
call thirtytwo(q**2, x, 2*k, i)
call thirtytwo(q**2, y, 2*k, i + k)
t = 1.0
do j = 0, 31
  g(j) = x(j) + t * y(j)
  g(j + 32) = x(j) - t * y(j)
  t = t * q
enddo
return
end

```

```

subroutine onetwentyeight(q, g, k, i)
common /time/ f
complex q, g(0:127), x(0:63), y(0:63), t, f(0:10000)
call sixtyfour(q**2, x, 2*k, i)
call sixtyfour(q**2, y, 2*k, i + k)
t = 1.0
do j = 0, 63
  g(j) = x(j) + t * y(j)
  g(j + 64) = x(j) - t * y(j)
  t = t * q
enddo
return end

```

### 3. Datos Obtenidos

0 -7.85398197 9.68566015E-02  
1 -7.73126364 9.68520641E-02  
2 -7.60854483 8.63548443E-02  
3 -7.48582649 0.112222075  
4 -7.36310816 8.02608728E-02  
5 -7.24038982 0.133014560  
6 -7.11767101 7.81609938E-02  
7 -6.99495268 0.159976795  
8 -6.87223434 7.95545578E-02  
9 -6.74951553 0.194102168  
10 -6.62679720 8.37250352E-02  
11 -6.50407887 0.236953318  
12 -6.38136053 8.93865302E-02  
13 -6.25864172 0.290877044  
14 -6.13592339 9.41522121E-02  
15 -6.01320505 0.360140562  
16 -5.89048672 9.27696228E-02  
17 -5.76776791 0.453490853  
18 -5.64504957 7.23850727E-02  
19 -5.52233124 0.593696237  
20 -5.39961243 8.13186355E-03  
21 -5.27689409 0.868452549  
22 -5.15417576 0.378536761  
23 -5.03145742 2.28940105  
24 -4.90873861 6.28331041  
25 -4.78602028 0.637709975  
26 -4.66330194 1.19227600  
27 -4.54058313 0.143828332  
28 -4.41786480 0.976188004  
29 -4.29514647 0.398846060  
30 -4.17242813 0.957380235  
31 -4.04970932 0.569211483  
32 -3.92699099 0.997116506  
33 -3.80427241 0.713880360  
34 -3.68155408 1.06443357  
35 -3.55883551 0.849214554  
36 -3.43611717 1.14794207  
37 -3.31339860 0.981031895  
38 -3.19068027 1.24191630  
39 -3.06796169 1.11133802  
40 -2.94524336 1.34266400  
41 -2.82252479 1.24041522  
42 -2.69980621 1.44730306

43 -2.57708788 1.36765146  
44 -2.45436931 1.55334353  
45 -2.33165097 1.49190402  
46 -2.20893240 1.65838480  
47 -2.08621407 1.61167717  
48 -1.96349549 1.76013136  
49 -1.84077704 1.72527051  
50 -1.71805859 1.85634637  
51 -1.59534013 1.83092797  
52 -1.47262168 1.94489717  
53 -1.34990311 1.92684591  
54 -1.22718465 2.02374458  
55 -1.10446620 2.01132631  
56 -0.981747746 2.09101963  
57 -0.859029293 2.08283901  
58 -0.736310840 2.14510655  
59 -0.613592327 2.13997340  
60 -0.490873873 2.18464899  
61 -0.368155420 2.18172193  
62 -0.245436937 2.20858288  
63 -0.122718468 2.20727301  
64 0.00000000 2.21624660  
65 0.122718468 2.21624804  
66 0.245436937 2.20727468  
67 0.368155420 2.20860243  
68 0.490873873 2.18170786  
69 0.613592327 2.18464422  
70 0.736310840 2.13999200  
71 0.859029293 2.14510202  
72 0.981747746 2.08283401  
73 1.10446620 2.09101391  
74 1.22718465 2.01135445  
75 1.34990311 2.02373457  
76 1.47262168 1.92686856  
77 1.59534013 1.94491172  
78 1.71805859 1.83094001  
79 1.84077704 1.85637116  
80 1.96349549 1.72528422  
81 2.08621407 1.76014686  
82 2.20893240 1.61168385  
83 2.33165097 1.65838253  
84 2.45436931 1.49192131  
85 2.57708788 1.55334020  
86 2.69980621 1.36767268  
87 2.82252479 1.44731987

88 2.94524336 1.24042916  
89 3.06796169 1.34266222  
90 3.19068027 1.11134505  
91 3.31339860 1.24192786  
92 3.43611717 0.981043398  
93 3.55883551 1.14796317  
94 3.68155408 0.849227846  
95 3.80427241 1.06444371  
96 3.92699099 0.713882983  
97 4.04970932 0.997118413  
98 4.17242813 0.569232345  
99 4.29514647 0.957377672  
100 4.41786480 0.398866475  
101 4.54058313 0.976201117  
102 4.66330194 0.143839240  
103 4.78602028 1.19228256  
104 4.90873861 0.637688577  
105 5.03145742 6.28333998  
106 5.15417576 2.28938937  
107 5.27689409 0.378503174  
108 5.39961243 0.868455708  
109 5.52233124 8.10957141E-03  
110 5.64504957 0.593693376  
111 5.76776791 7.23969340E-02  
112 5.89048672 0.453494191  
113 6.01320505 9.27865505E-02  
114 6.13592339 0.360154599  
115 6.25864172 9.41429138E-02  
116 6.38136053 0.290896267  
117 6.50407887 8.93971249E-02  
118 6.62679720 0.236957490  
119 6.74951553 8.37188959E-02  
120 6.87223434 0.194121361  
121 6.99495268 7.95744732E-02  
122 7.11767101 0.159960747  
123 7.24038982 7.81757832E-02  
124 7.36310816 0.133017063  
125 7.48582649 8.02773237E-02  
126 7.60854483 0.112228982  
127 7.73126364 8.63629580E-02



## 4. Gráfica

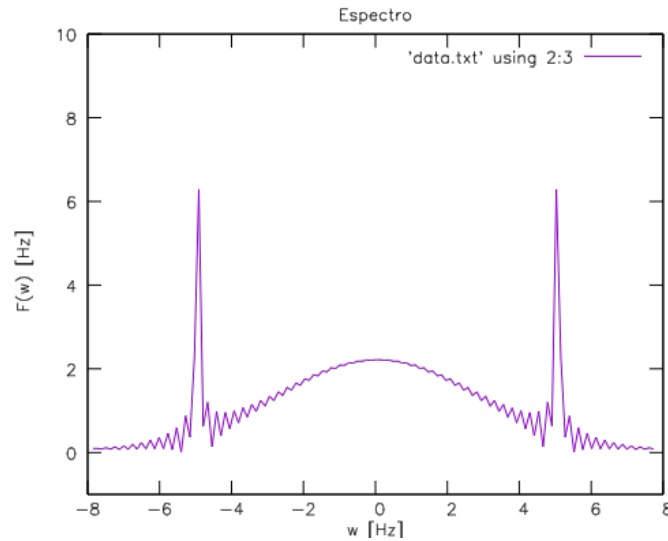


Figura 1: Gráfica Obtenida

## 5. Conclusión

El desarrollo de esta actividad permitió comprender de manera práctica cómo se puede analizar una serie de tiempo utilizando herramientas computacionales, en particular la Transformada Rápida de Fourier (FFT). A partir de una señal construida artificialmente, se logró descomponer sus componentes en el dominio de la frecuencia, identificando las frecuencias dominantes que definen su comportamiento temporal.

El uso del lenguaje de programación Fortran fue clave para implementar el algoritmo de la FFT de manera eficiente, así como para generar y procesar los datos necesarios. La estructura modular del código facilitó su comprensión y adaptación para distintos tamaños de datos, evidenciando la versatilidad de este enfoque.

Finalmente, la interpretación del espectro obtenido nos brindó una visión más clara de los elementos que componen la señal original, validando la utilidad de la FFT como herramienta fundamental en el análisis de señales y series de tiempo en física computacional.