Características de las funciones

Hola, en esta breve clase te presentaré algunas características de las funciones reales que estoy seguro te serán de utilidad en un futuro. Pero antes que nada, ¿por qué se llaman funciones reales? Esto es muy fácil de contestar.

Se les llama *funciones reales* porque tanto su **dominio** como el **codominio** `(recuerda que al codominio también se le puede llamar rango o imagen)` están contenidos en el conjunto de los números reales.
Es decir, el conjunto que contiene a los números racionales e irracionales. En otras palabras cualquier número que se te ocurra que no sea imaginario.

Todos los ejemplos que veamos a lo largo de este curso serán sobre números reales. Una vez que ya tenemos claro cuáles son las funciones reales y por qué se les llama así, pasemos a describir algunas de sus características.

Función par:

Una función es par si cumple la siguiente relación a lo largo de su dominio $[x]$.

$$f(x) = f(-x)$$

Si lo notaste, esta relación nos dice que **una función es par** si es **simétrica al eje vertical (eje Y)**. Por ejemplo una parábola

$$y = x^2$$

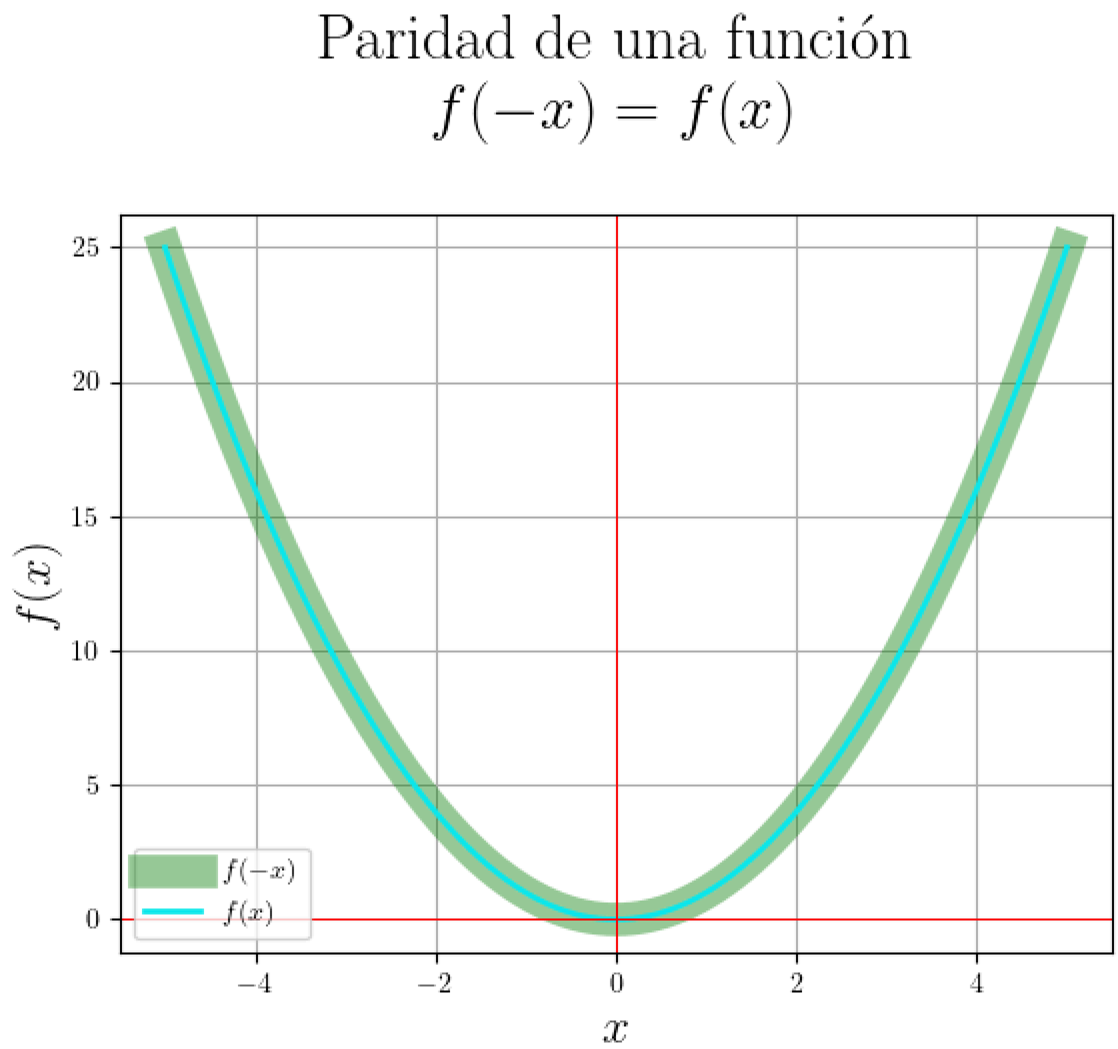
In [ ]:
```
import numpy as np
import matplotlib.pyplot as plt

#Configurando Latex
# Configuración de Matplotlib para usar LaTeX
plt.rcParams.update({
    "text.usetex": True,
    "font.family": "serif",
    "font.serif": ["Computer Modern Roman"],
    "text.latex.preamble": r"\usepackage{amsmath}"
})
```

In [ ]:
```
# definiendo funcion parabola
def f(x):
  return x**2

#Definiendo variables
N = 1000
c = 4
x = np.linspace(-5,5, num=N)

#Creando graficas
fig, ax = plt.subplots()
ax.plot(x,f(-x),label=r'$f(-x)$',color='green',alpha=0.4,linewidth=12)
ax.plot(x,f(x),label=r'$f(x)$',color='#0EE9ED',linewidth=2)
ax.grid()
ax.axhline(y=0, color='r',linewidth=.7)
ax.axvline(x=0, color='r',linewidth=.7)
plt.title('Paridad de una función\n$f(-x)=f(x)$\n',fontsize=23)
plt.xlabel(r'$x$',fontsize=18)
plt.ylabel(r'$f(x)$',fontsize=18)
plt.legend()
plt.show()
```



Como se puede observar en la gráfica tanto $f(x)$ como $f(-x)$ tienen los mismos valores. Entonces podemos decir que:

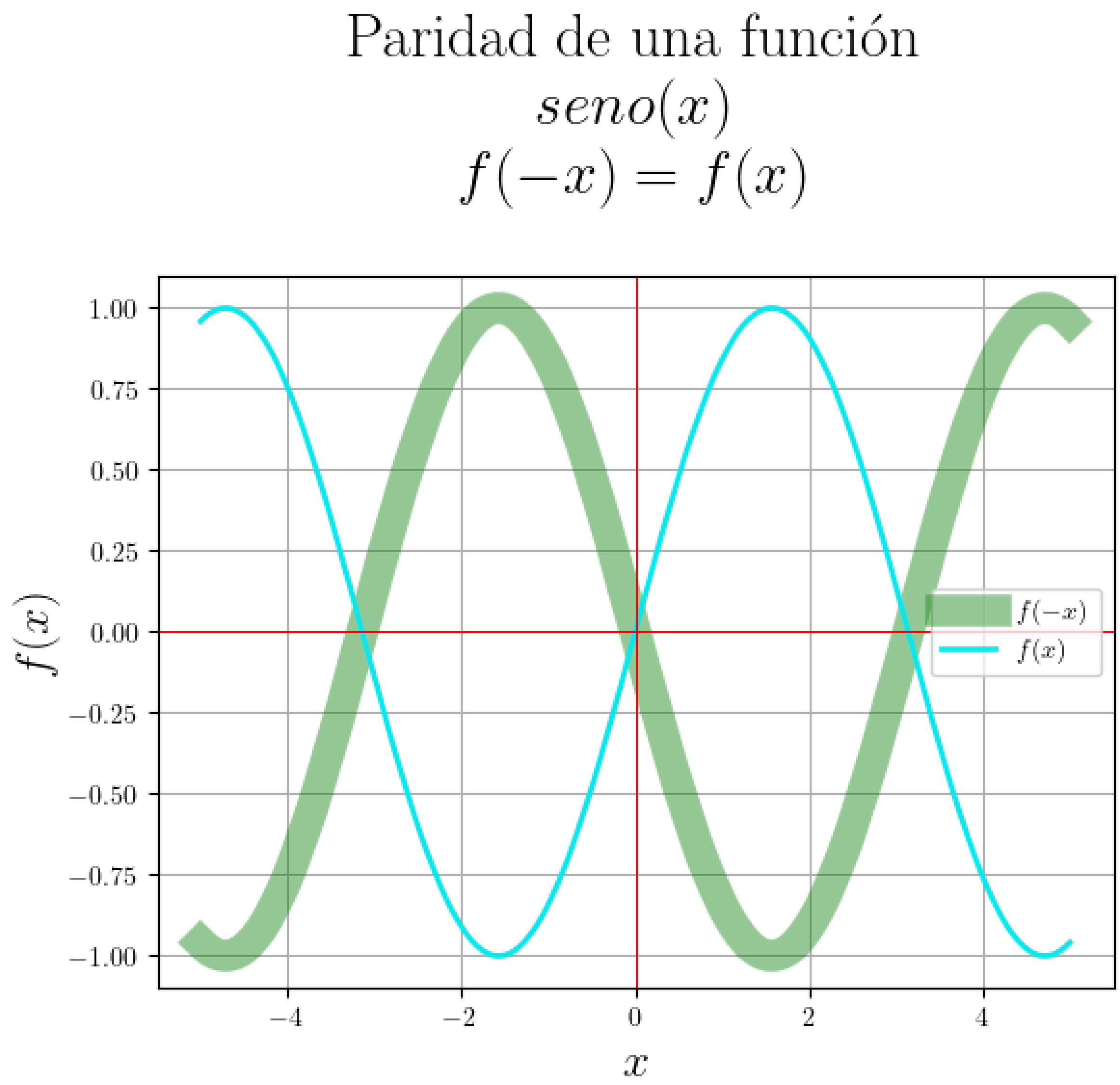
$$x^2 \text{ Es una función par}$$

Apliquemosle el criterio a la función $seno(x)$

In [ ]:
```python
# definiendo función seno
def f(x):
  return np.sin(x)

#Definiendo variables
N = 1000
c = 4
x = np.linspace(-5,5, num=N)

#Creando graficas
fig, ax = plt.subplots()
ax.plot(x,f(-x),label=r'$f(-x)$',color='green',alpha=0.4,linewidth=12)
ax.plot(x,f(x),label=r'$f(x)$',color='#0EE9ED',linewidth=2)
ax.grid()
ax.axhline(y=0, color='r',linewidth=.7)
ax.axvline(x=0, color='r',linewidth=.7)
plt.title('Paridad de una función\n$seno(x)$\n$f(-x)=f(x)$\n',fontsize=23)
plt.xlabel(r'$x$',fontsize=18)
plt.ylabel(r'$f(x)$',fontsize=18)
plt.legend()
plt.show()
```



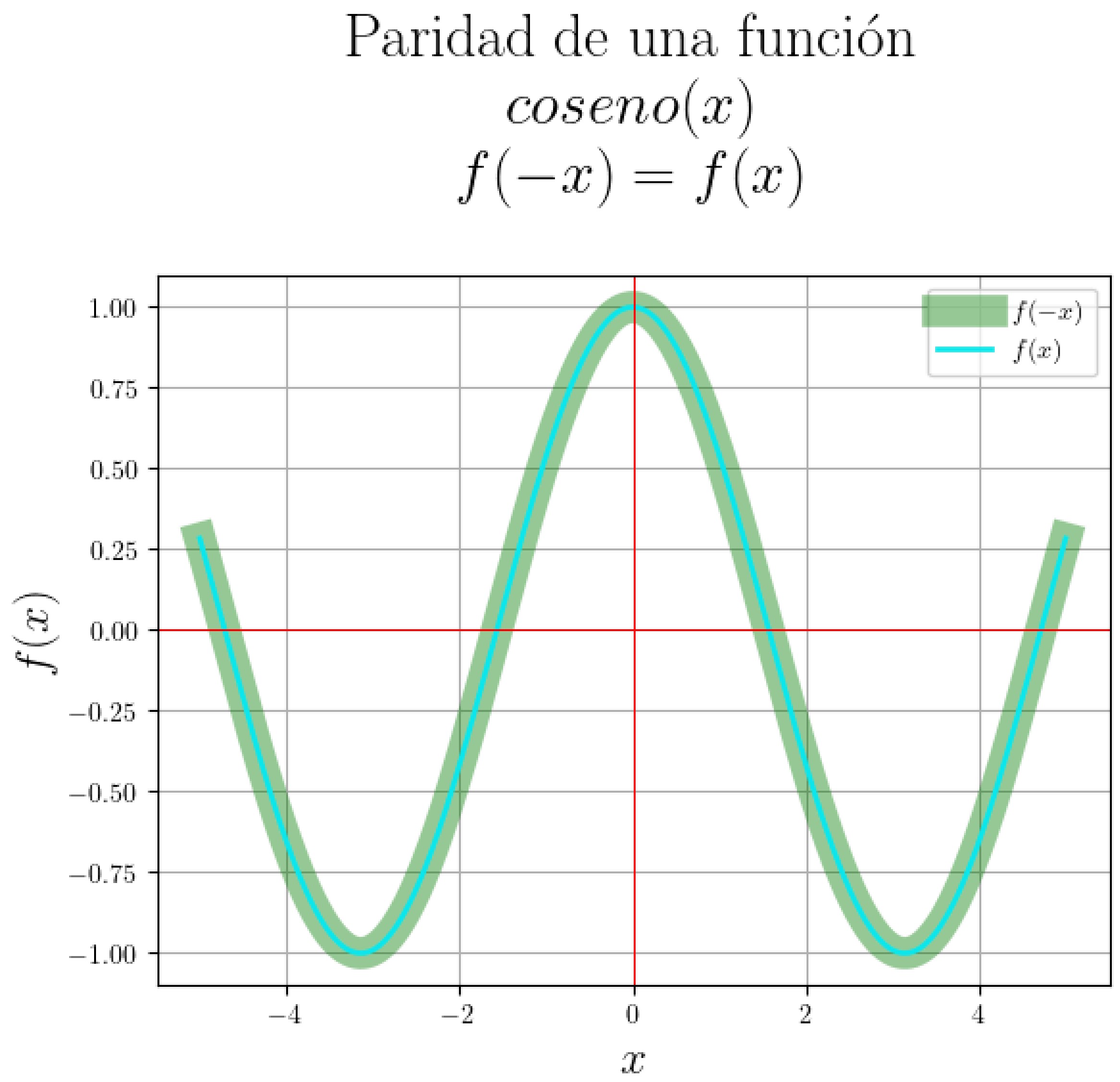
Cómo se puede ver $seno(x)$ no cumple con esta condición. Veamos si el $coseno(x)$ cumple

In [ ]:
```
# definiendo función coseno
def f(x):
  return np.cos(x)

#Definiendo variables
N = 1000
c = 4
x = np.linspace(-5,5, num=N)

#Creando graficas
fig, ax = plt.subplots()
ax.plot(x,f(-x),label=r'$f(-x)$',color='green',alpha=0.4,linewidth=12)
ax.plot(x,f(x),label=r'$f(x)$',color='#0EE9ED',linewidth=2)
ax.grid()
ax.axhline(y=0, color='r',linewidth=.7)
ax.axvline(x=0, color='r',linewidth=.7)
plt.title('Paridad de una función\n$coseno(x)$\n$f(-x)=f(x)$\n',fontsize=23)
plt.xlabel(r'$x$',fontsize=18)
plt.ylabel(r'$f(x)$',fontsize=18)
plt.legend()
plt.show()
```

Paridad de una función



El $coseno(x)$ si cumple, entonces quiere decir que es una **función par**.

Función impar

Una función es impar si cumple la siguiente relación a lo largo de su dominio $[x]$:

$$f(-x) = -f(x)$$

Esta relación nos indica que una **función es impar** si es **simétrica al eje horizontal (eje X)**.
Por ejemplo, una función cúbica es impar.

$$f(x = x^3)$$

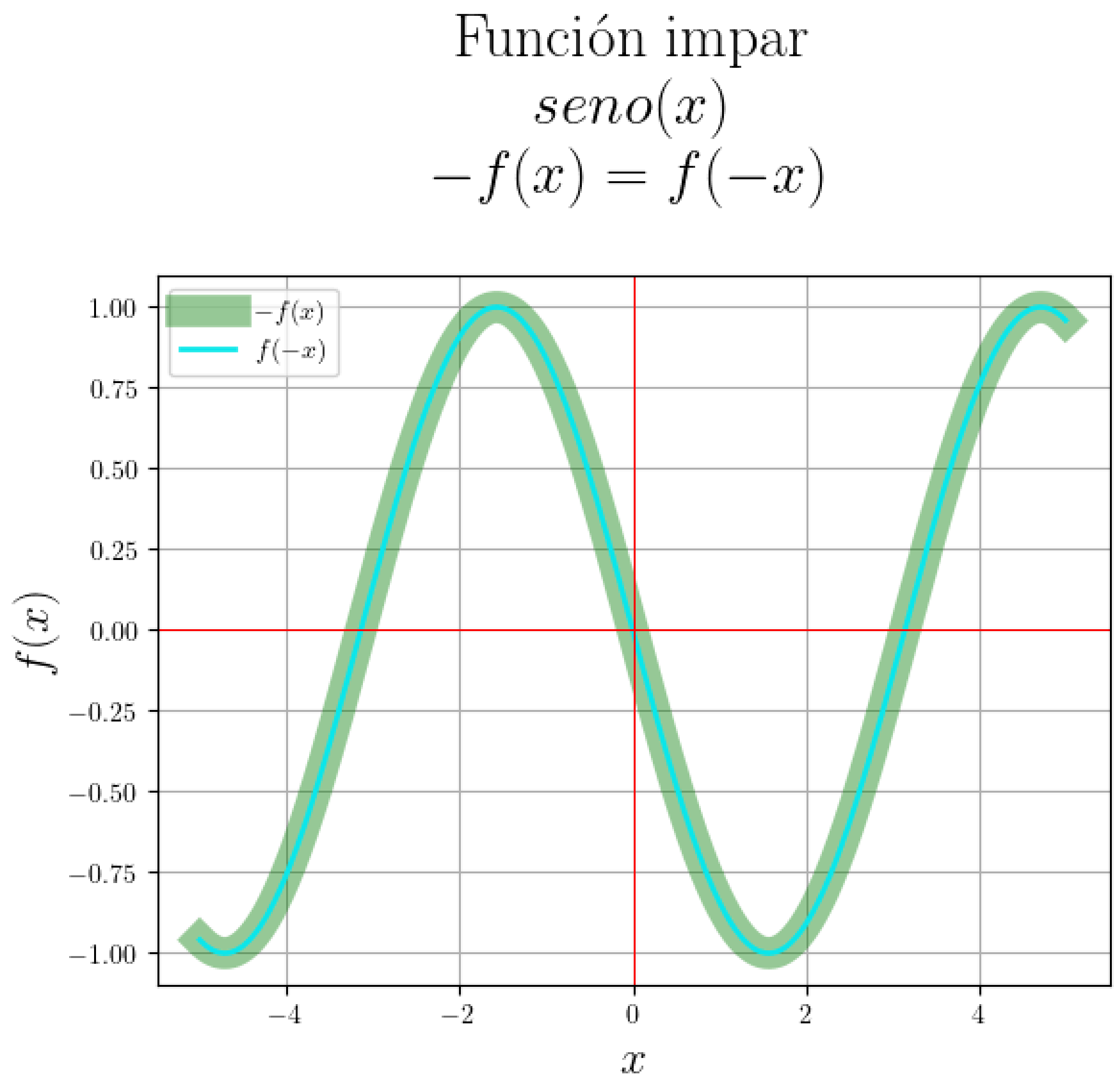
Ahora

Recordemos que el $seno(x)$ no cumplió con el criterio anterior, ahora veamos si cumple con este.

```
In [ ]: # definiendo función seno
def f(x):
  return np.sin(x)

#Definiendo variables
N = 1000
c = 4
x = np.linspace(-5,5, num=N)

#Creando graficas
fig, ax = plt.subplots()
ax.plot(x,-1*f(x),label=r'$-f(x)$',color='green',alpha=0.4,linewidth=12)
ax.plot(x,f(-x),label=r'$f(-x)$',color='#0EE9ED',linewidth=2)
ax.grid()
ax.axhline(y=0, color='r',linewidth=.7)
ax.axvline(x=0, color='r',linewidth=.7)
plt.title('Función impar\n$seno(x)$\n$-f(x)=f(-x)$\n',fontsize=23)
plt.xlabel(r'$x$',fontsize=18)
plt.ylabel(r'$f(x)$',fontsize=18)
plt.legend()
plt.show()
```



Importante:

> Las dos características o propiedades anteriores están muy relacionadas con la simetría.

Puedes comprobarlas tú mismo con diferentes funciones e incluso hacer una pequeña rutina en Python que compruebe si una función es par o impar.

Función acotada

Una función es acotada si su codominio (también conocido como rango o imagen) se encuentra entre dos valores, es decir, está acotado.

Esta definición se define como;
hay un número m que para todo valor del dominio de la función se cumple que:

$$-m \leq f(x) \leq m$$

Por ejemplo:

> La función $seno(x)$ y $coseno(x)$ están acotadas en el intervalo $[-1, 1]$, dentro de su co-dominio.

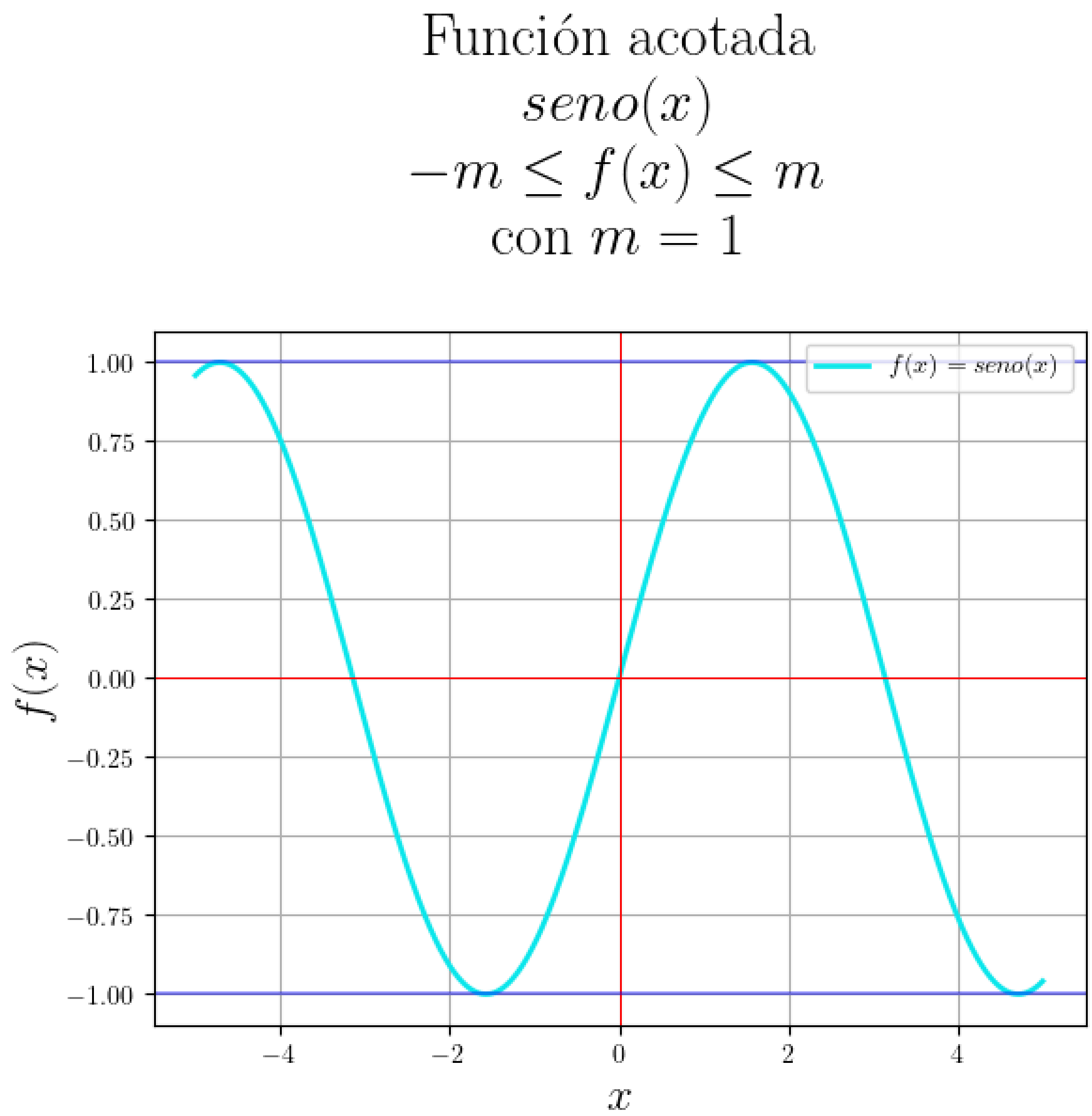
In [ ]:

```python
# definiendo función seno
def f(x):
  return np.sin(x)

#Definiendo variables
N = 1000
c = 4
x = np.linspace(-5,5, num=N)

#Creando graficas
fig, ax = plt.subplots()
ax.plot(x,f(x),label=r'$f(x)=seno(x)$',color='#0EE9ED',linewidth=2)
ax.grid()
ax.axhline(y=0, color='r',linewidth=.7)
ax.axvline(x=0, color='r',linewidth=.7)
ax.axhline(y=1, color='blue',linewidth=1.2,alpha=0.5)
ax.axhline(y=-1, color='blue',linewidth=1.2,alpha=0.5)
plt.title('Función acotada\n$seno(x)$\n$-m\leq f(x)\leq m$\ncon $m=1$\n',fontsize=23)
plt.xlabel(r'$x$',fontsize=18)
plt.ylabel(r'$f(x)$',fontsize=18)
plt.legend()
plt.show()
```

```
<>:18: SyntaxWarning: invalid escape sequence '\l'
<>:18: SyntaxWarning: invalid escape sequence '\l'
/tmp/ipykernel_1270/799568745.py:18: SyntaxWarning: invalid escape sequence '\l'
  plt.title('Función acotada\n$seno(x)$\n$-m\leq f(x)\leq m$\ncon $m=1$\n',fontsize=23)
```



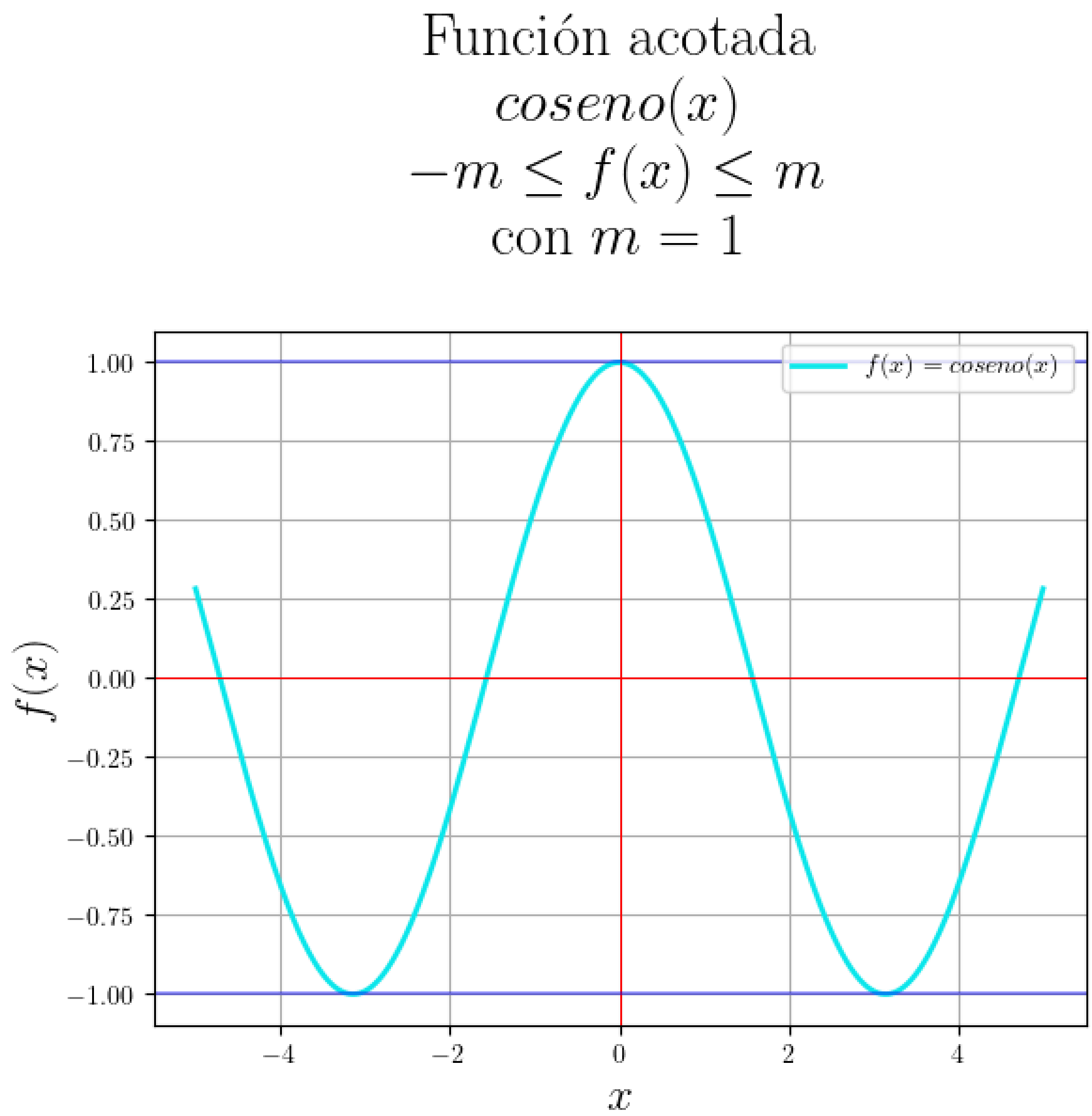
In [ ]:

```
# definiendo función coseno
def f(x):
  return np.cos(x)

#Definiendo variables
N = 1000
c = 4
x = np.linspace(-5,5, num=N)

#Creando graficas
fig, ax = plt.subplots()
ax.plot(x,f(x),label=r'$f(x)=coseno(x)$',color='#0EE9ED',linewidth=2)
ax.grid()
ax.axhline(y=0, color='r',linewidth=.7)
ax.axvline(x=0, color='r',linewidth=.7)
ax.axhline(y=1, color='blue',linewidth=1.2,alpha=0.5)
ax.axhline(y=-1, color='blue',linewidth=1.2,alpha=0.5)
plt.title('Función acotada\n$coseno(x)$\n$-m\leq f(x)\leq m$\ncon $m=1$\n',fontsize=23)
plt.xlabel(r'$x$',fontsize=18)
plt.ylabel(r'$f(x)$',fontsize=18)
plt.legend()
plt.show()
```

```
<>:18: SyntaxWarning: invalid escape sequence '\l'
<>:18: SyntaxWarning: invalid escape sequence '\l'
/tmp/ipykernel_1270/3376066378.py:18: SyntaxWarning: invalid escape sequence '\l'
  plt.title('Función acotada\n$coseno(x)$\n$-m\leq f(x)\leq m$\ncon $m=1$\n',fontsize=23)
```



Función monótona

Estas funciones son útiles de reconocer o analizar debido a que nos permiten saber si una función crece o decrece en alguno de sus intervalos.

Que algo sea monótono significa que no tiene variaciones. Entonces las funciones monótonas son aquellas que dentro de un *intervalo (I)*, perteneciente a los números reales, cumple alguna de estas propiedades:

1.- La función es monótona y estrictamente creciente:

Si para todo $x1, x2 \in I : x1 > x2 \leftrightarrow f(x1) < f(x2)$

No te preocupes si es un poco difícil de leer esta definición que para eso estoy aquí para explicarte.

La forma correcta de leer esto es:

> "si para todo $x1$ y $x2$ que pertenecen al intervalo I, tal que $x1$ sea menor a $x2$, si y solo si $f(x1)$ sea menor a $f(x2)$".

En palabras mucho más sencillas, lo que nos dice esta definición es que $x1$ siempre tiene que ser menor que $x2$ en nuestro intervalo I, y que al evaluar $x2$ en la función el resultado de esto siempre será mayor que si evaluamos la función en $x1$.

Para las siguientes tres definiciones restantes no cambia mucho la forma en la que se interpretan.

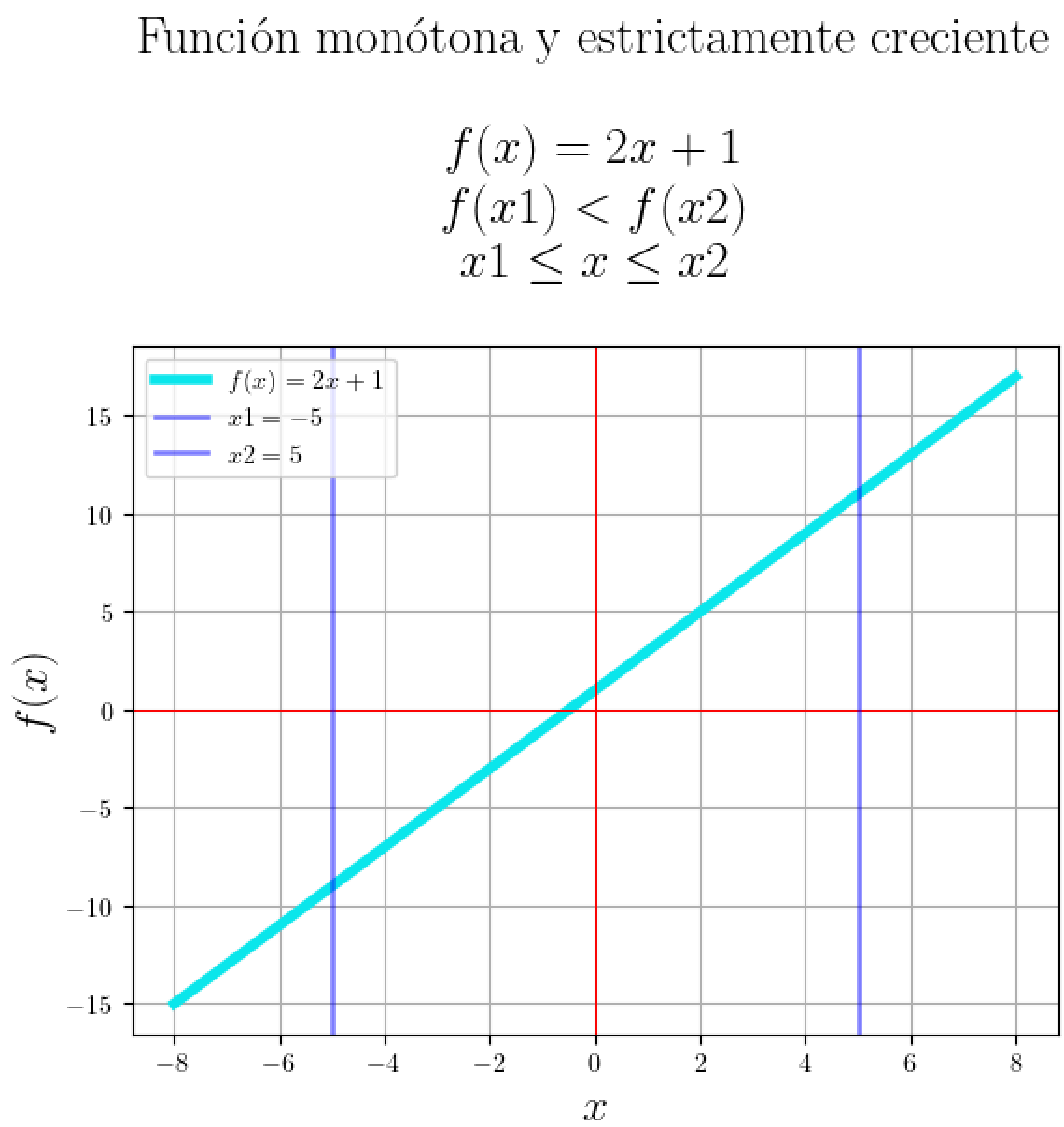
In [ ]:

```
# definiendo función
def f(x):
  return 2*x+1

#Definiendo variables
N = 1000
c = 4
x1=-5
x2=5
x = np.linspace(x1-3,x2+3, num=N)

#Creando graficas
fig, ax = plt.subplots()
ax.plot(x,f(x),label=r'$f(x)=2x+1$',color='#0EE9ED',linewidth=4)
ax.grid()
ax.axhline(y=0, color='r',linewidth=.7)
ax.axvline(x=0, color='r',linewidth=.7)
ax.axvline(x=x1, color='blue',linewidth=1.8,alpha=0.5,label=f'$x1={x1}$')
ax.axvline(x=x2, color='blue',linewidth=1.8,alpha=0.5,label=f'$x2={x2}$')
plt.title(f'Función monótona y estrictamente creciente\n\n$f(x)=2x+1$\n$f(x1)<f(x2)$\n$x1\leq x\leq x2$\n',fontsize=
plt.xlabel(r'$x$',fontsize=18)
plt.ylabel(r'$f(x)$',fontsize=18)
plt.legend()
plt.show()
```

```
<>:20: SyntaxWarning: invalid escape sequence '\l'
<>:20: SyntaxWarning: invalid escape sequence '\l'
/tmp/ipykernel_1270/3143365803.py:20: SyntaxWarning: invalid escape sequence '\l'
  plt.title(f'Función monótona y estrictamente creciente\n\n$f(x)=2x+1$\n$f(x1)<f(x2)$\n$x1\leq x\leq x2$\n',fontsize
=20)
```



2.- La función es monótona y estrictamente decreciente:

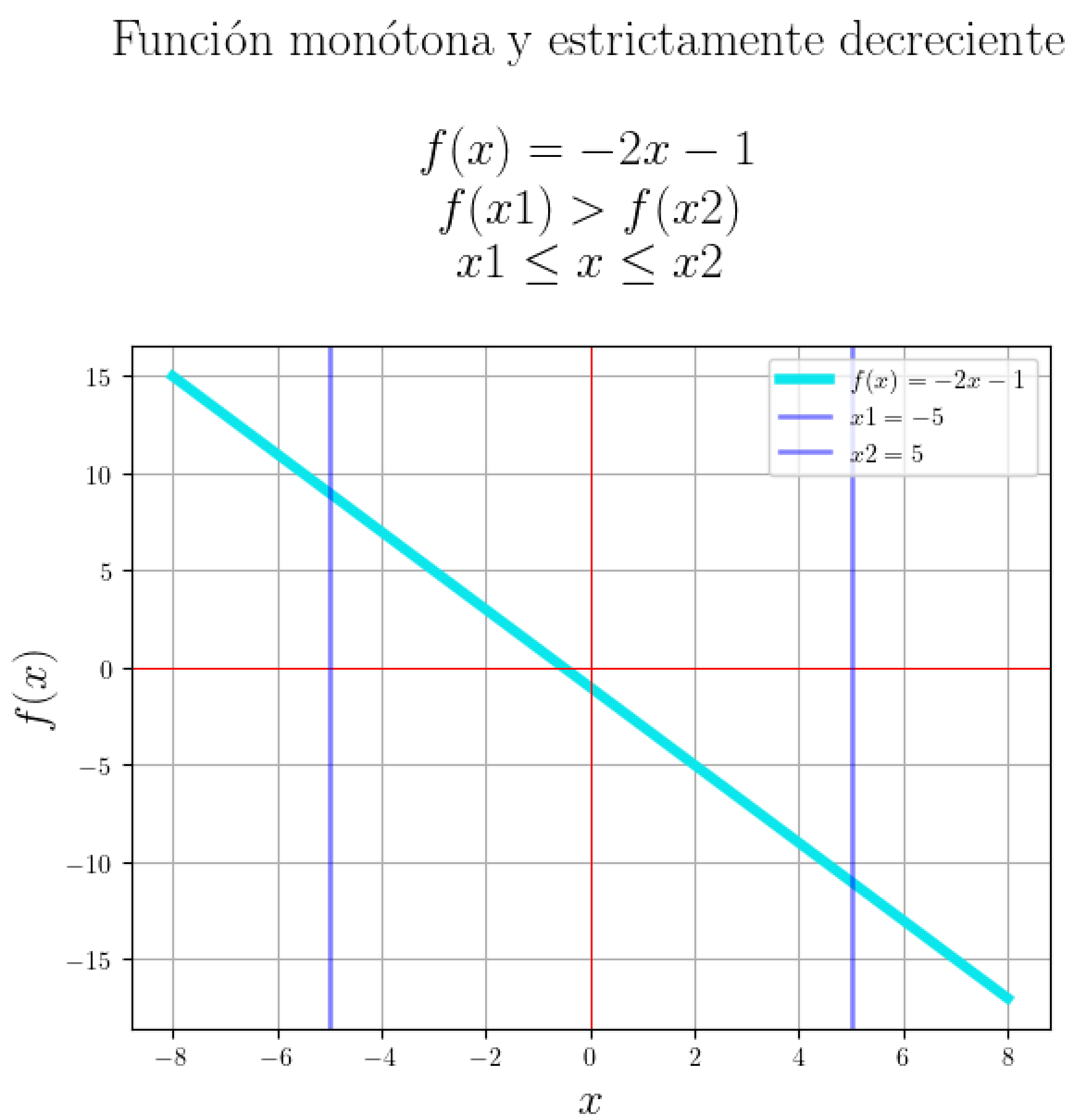
Si para todo $x1, x2 \in I : x1 > x2 \leftrightarrow f(x1) > f(x2)$

In [ ]:
```python
# definiendo función parabóla
def f(x):
  return -2*x-1

#Definiendo variables
N = 1000
c = 4
x1=-5
x2=5
x = np.linspace(x1-3,x2+3, num=N)

#Creando graficas
fig, ax = plt.subplots()
ax.plot(x,f(x),label=r'$f(x)=-2x-1$',color='#0EE9ED',linewidth=4)
ax.grid()
ax.axhline(y=0, color='r',linewidth=.7)
ax.axvline(x=0, color='r',linewidth=.7)
ax.axvline(x=x1, color='blue',linewidth=1.8,alpha=0.5,label=f'$x1={x1}$')
ax.axvline(x=x2, color='blue',linewidth=1.8,alpha=0.5,label=f'$x2={x2}$')
plt.title(f'Función monótona y estrictamente decreciente\n\n$f(x)=-2x-1$\n$f(x1)>f(x2)$\n$x1\leq x\leq x2$\n',fontsi
plt.xlabel(r'$x$',fontsize=18)
plt.ylabel(r'$f(x)$',fontsize=18)
plt.legend()
plt.show()
```

```
<>:20: SyntaxWarning: invalid escape sequence '\l'
<>:20: SyntaxWarning: invalid escape sequence '\l'
/tmp/ipykernel_1270/4293691386.py:20: SyntaxWarning: invalid escape sequence '\l'
  plt.title(f'Función monótona y estrictamente decreciente\n\n$f(x)=-2x-1$\n$f(x1)>f(x2)$\n$x1\leq x\leq x2$\n',fonts
ize=20)
```



3.- La función es monótona y creciente:

Si para todo $x1, x2 \in I : x1 > x2 \leftrightarrow f(x1) \leq f(x2)$

4.- La función es monótona y decreciente:

Si para todo $x1, x2 \in I : x1 > x2 \leftrightarrow f(x1) \geq f(x2)$

Funciones periódicas

Las funciones periódicas son aquellas que se repiten cada cierto periodo, este periodo se denomina con la letra T.
La relación que debe cumplir la función para ser periódica es la siguiente.

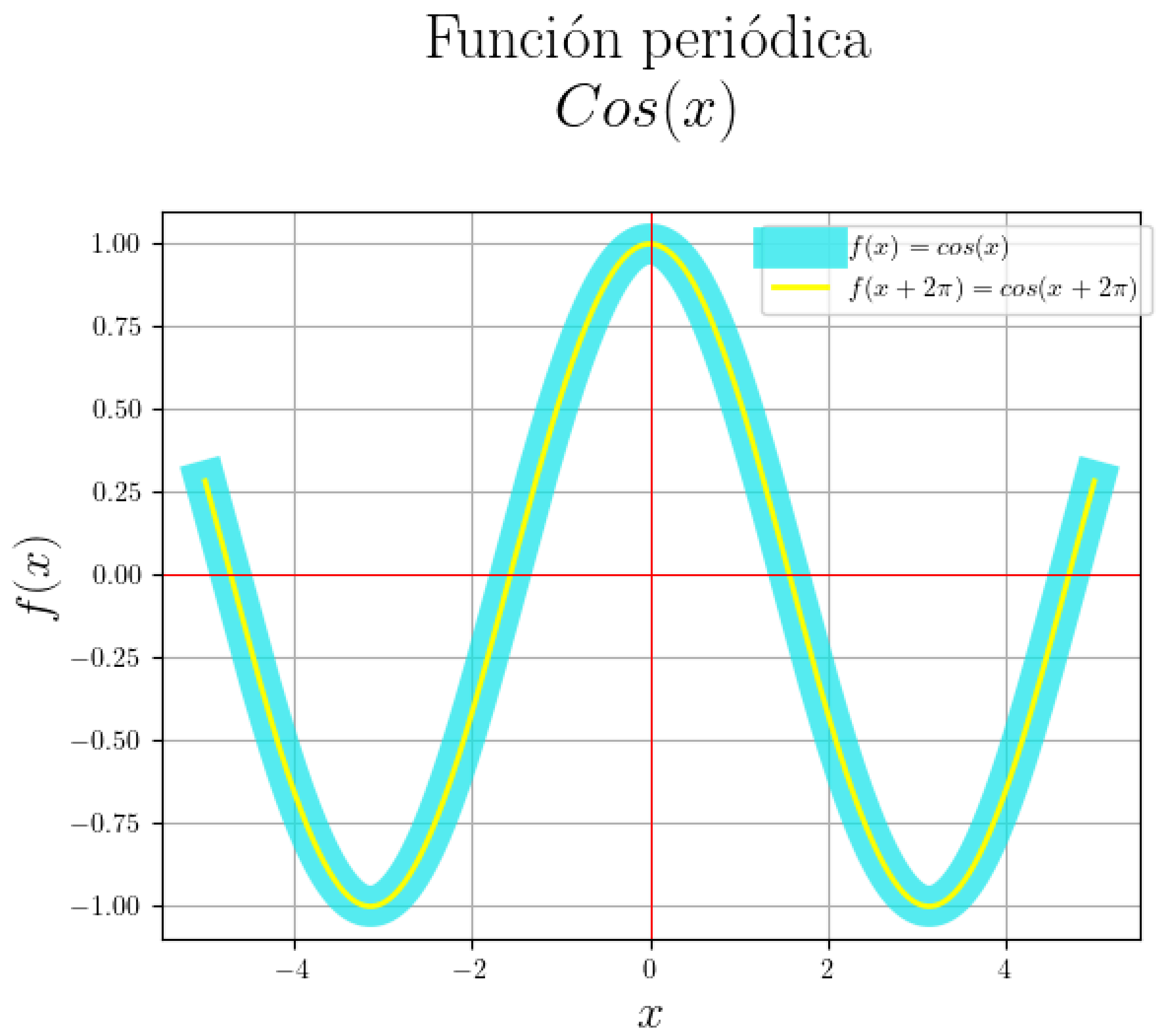
$$f(x) = f(x+T), T \neq 0$$

Por ejemplo, la función $seno(x)$ y $coseno(x)$ son funciones periódicas con un periodo $T = 2\pi$. Es decir que si nosotros calculamos $f(x)$ y calculamos $f(x+2\pi)$ en la función $seno(x)$ el valor que nos den ambas expresiones es el mismo.

In [ ]:
```
# definiendo función coseno
def f(x):
  return np.cos(x)

#Definiendo variables
N = 1000
c = 4
x = np.linspace(-5,5, num=N)

#Creando graficas
fig, ax = plt.subplots()
ax.plot(x,f(x),label=r'$f(x)=cos(x)$',color='#0EE9ED',linewidth=15,alpha=0.7)
ax.plot(x,f(x+(2*np.pi)),label=r'$f(x+2\pi)=cos(x+2\pi)$',color='yellow',linewidth=2)
ax.grid()
ax.axhline(y=0, color='r',linewidth=.7)
ax.axvline(x=0, color='r',linewidth=.7)
plt.title('Función periódica\n$Cos(x)$\n',fontsize=23)
plt.xlabel(r'$x$',fontsize=18)
plt.ylabel(r'$f(x)$',fontsize=18)
plt.legend(bbox_to_anchor=(0.6, 1))
plt.show()
```



Funciones convexas y cóncavas

La forma de demostrar la concavidad de una función se puede hacer a través del análisis de derivadas consecutivas, pero aún no llegamos a eso, así que no te preocupes. A continuación, te dejo una forma súper intuitiva de ver si una función es cóncava o convexa.

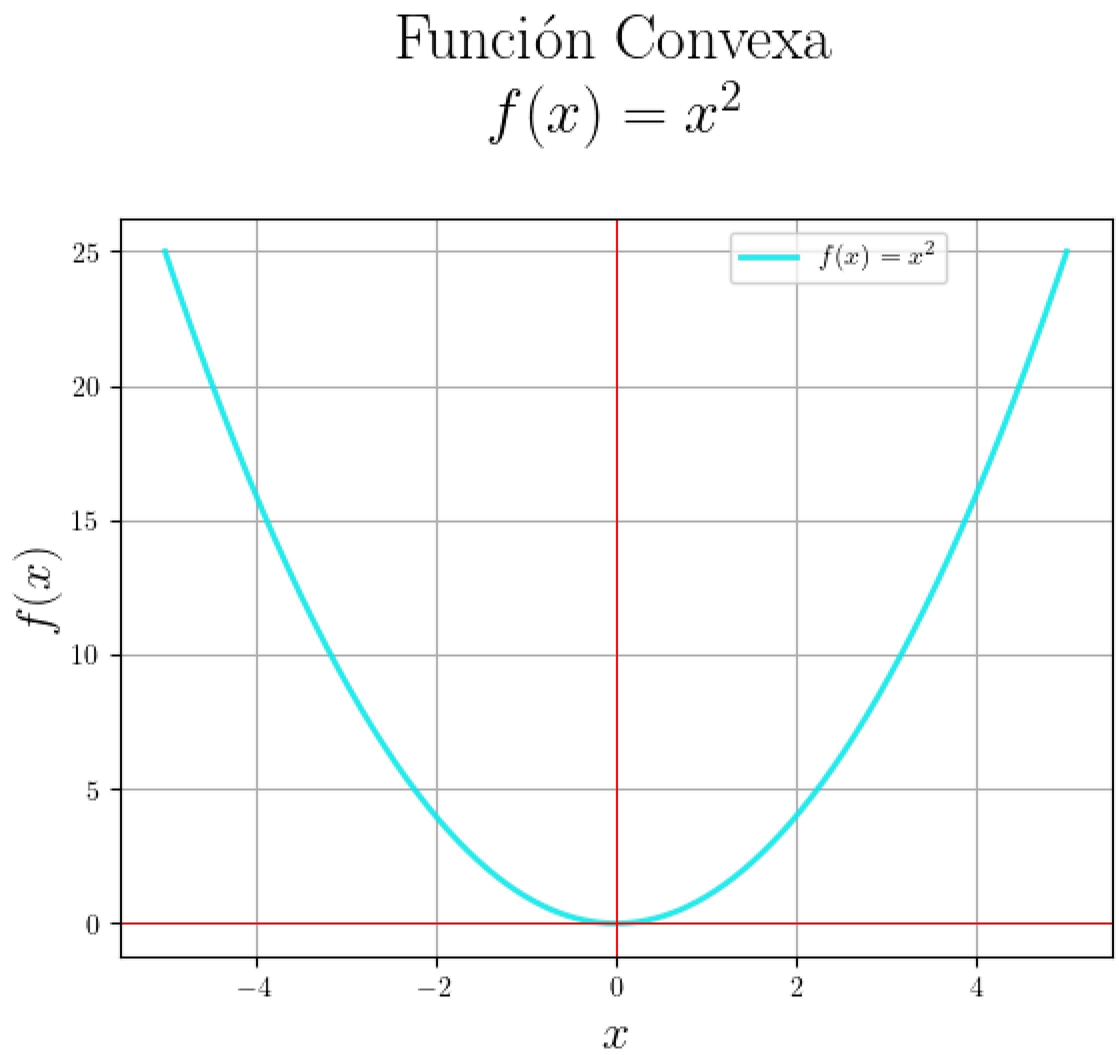
Convexa

Se dice que una función dentro de un intervalo es convexa si la función "abre hacia arriba" $\smile$. Es decir si se ve la siguiente manera:

In [ ]:
```
# definiendo función coseno
def f(x):
  return x**2

#Definiendo variables
N = 1000
c = 4
x = np.linspace(-5,5, num=N)

#Creando graficas
fig, ax = plt.subplots()
ax.plot(x,f(x),label=r'$f(x)=x^2$',color='#0EE9ED',linewidth=2,alpha=0.9)
ax.grid()
ax.axhline(y=0, color='r',linewidth=.7)
ax.axvline(x=0, color='r',linewidth=.7)
plt.title('Función Convexa\n$f(x)=x^2$\n',fontsize=23)
plt.xlabel(r'$x$',fontsize=18)
plt.ylabel(r'$f(x)$',fontsize=18)
plt.legend(bbox_to_anchor=(0.6, 1))
plt.show()
```



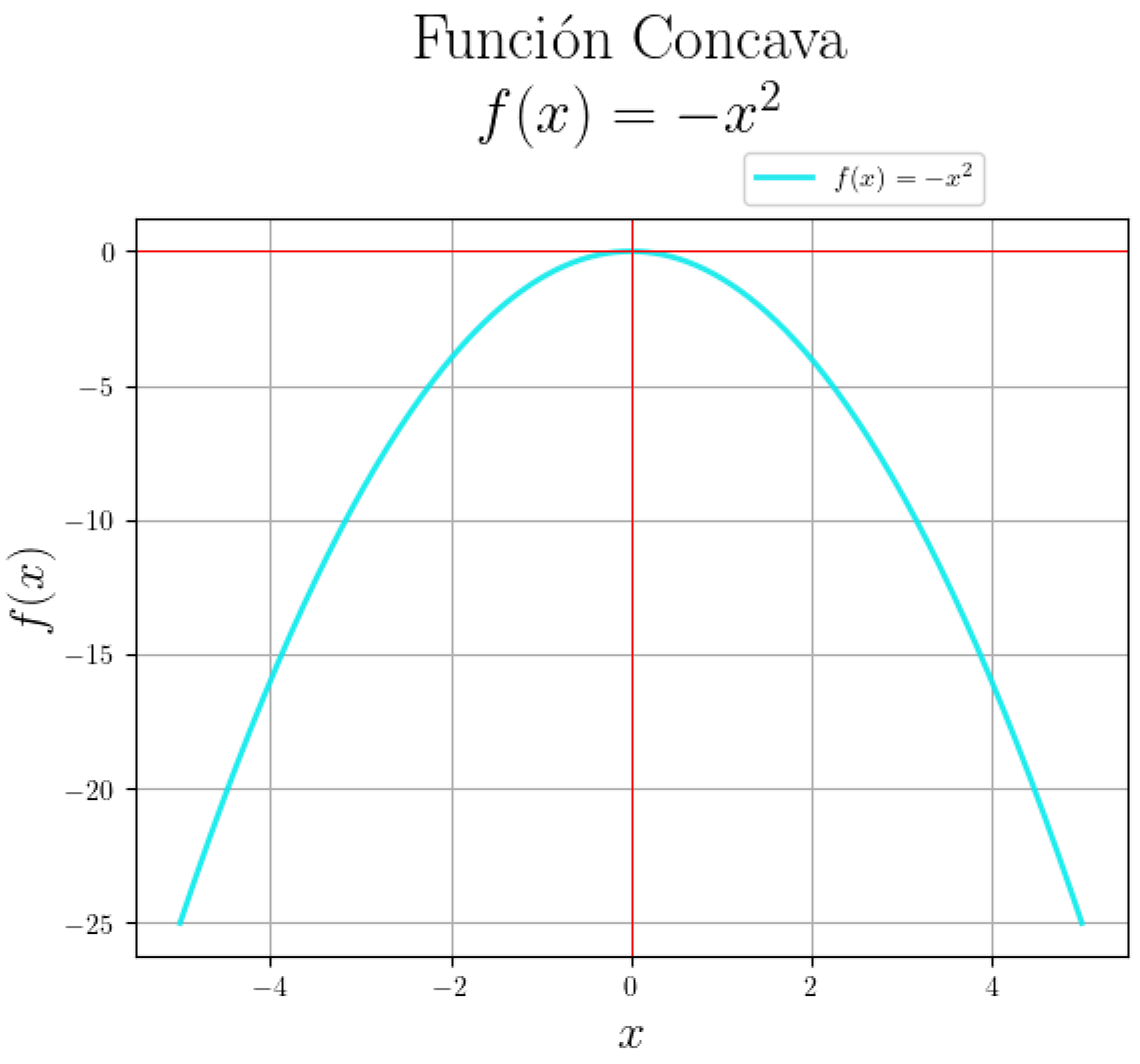
Cóncava

Ahora, ¿qué sería una función cóncava? Pues así es, lo contrario de una convexa. Se dice que una función dentro de un intervalo es concava si la función "abre hacia abajo" $\frown$. Es decir si se ve la siguiente manera:

In [ ]:
```
# definiendo función coseno
def f(x):
  return -x**2

#Definiendo variables
N = 1000
c = 4
x = np.linspace(-5,5, num=N)

#Creando graficas
fig, ax = plt.subplots()
ax.plot(x,f(x),label=r'$f(x)=-x^2$',color='#0EE9ED',linewidth=2,alpha=0.9)
ax.grid()
ax.axhline(y=0, color='r',linewidth=.7)
ax.axvline(x=0, color='r',linewidth=.7)
plt.title('Función Concava\n$f(x)=-x^2$\n',fontsize=23)
plt.xlabel(r'$x$',fontsize=18)
plt.ylabel(r'$f(x)$',fontsize=18)
plt.legend(bbox_to_anchor=(0.6, 1))
plt.show()
```



Como ves, identificar si una función es cóncava o convexa a través de su gráfica es muy sencillo.

Pero si quieres enfrentar un reto, te dejo que cuando sepas que es una derivada regreses a esta clase y busques la forma de comprobar si una función es cóncava o convexa de forma analítica.

> Como pista, se hace a través del análisis de la segunda derivada.

Con esta última característica hemos finalizado esta clase 🥳. Te espero en la siguiente clase en donde aprenderás cómo se compone una neurona.

Spoiler, está hecha de funciones.