

IEE352 - Laboratorio 00

Gabriel Ramirez Orihuela

PUCP

Procesamiento Digital de Señales

Contenido

- 1 Jupyter Notebooks
- 2 Python
- 3 NumPy
- 4 Matplotlib
- 5 Conclusiones

- Los cuadernos Jupyter son una herramienta popular para la computación interactiva y el análisis de datos.

- Los cuadernos Jupyter son una herramienta popular para la computación interactiva y el análisis de datos.
- Permiten escribir y ejecutar código en un navegador web, combinando código, texto y visualizaciones de datos en un solo documento.

- Los cuadernos Jupyter son una herramienta popular para la computación interactiva y el análisis de datos.
- Permiten escribir y ejecutar código en un navegador web, combinando código, texto y visualizaciones de datos en un solo documento.
- Los cuadernos admiten muchos lenguajes de programación, pero se usan más comúnmente con Python.

- Los cuadernos Jupyter son una herramienta popular para la computación interactiva y el análisis de datos.
- Permiten escribir y ejecutar código en un navegador web, combinando código, texto y visualizaciones de datos en un solo documento.
- Los cuadernos admiten muchos lenguajes de programación, pero se usan más comúnmente con Python.
- Para empezar con Jupyter, se puede usar la distribución Anaconda o instalarlo directamente con pip.

- Python es un lenguaje de programación popular para el cómputo científico y el análisis de datos.

- Python es un lenguaje de programación popular para el cómputo científico y el análisis de datos.
- Tiene una sintaxis simple y expresiva, lo que facilita escribir y leer código.

- Python es un lenguaje de programación popular para el cómputo científico y el análisis de datos.
- Tiene una sintaxis simple y expresiva, lo que facilita escribir y leer código.
- Python tiene una comunidad grande y activa, con muchas bibliotecas y herramientas disponibles para varias tareas.

- Python es un lenguaje de programación popular para el cómputo científico y el análisis de datos.
- Tiene una sintaxis simple y expresiva, lo que facilita escribir y leer código.
- Python tiene una comunidad grande y activa, con muchas bibliotecas y herramientas disponibles para varias tareas.
- Algunas bibliotecas de Python populares para el cómputo científico y el análisis de datos incluyen NumPy y Matplotlib.

- 1 Ir a la página de descargas de Python:
`https://www.python.org/downloads/`.

Descargando Python

- 1 Ir a la página de descargas de Python:
<https://www.python.org/downloads/>.
- 2 Seleccionar la versión de Python que se desea descargar.

Descargando Python

- 1 Ir a la página de descargas de Python:
<https://www.python.org/downloads/>.
- 2 Seleccionar la versión de Python que se desea descargar.
- 3 Seleccionar la opción de descarga para Windows.

Descargando Python

- 1 Ir a la página de descargas de Python:
<https://www.python.org/downloads/>.
- 2 Seleccionar la versión de Python que se desea descargar.
- 3 Seleccionar la opción de descarga para Windows.
- 4 Esperar a que se descargue el archivo ejecutable (.exe).

- 1 Hacer doble clic en el archivo ejecutable (.exe) descargado.

Instalando Python

- 1 Hacer doble clic en el archivo ejecutable (.exe) descargado.
- 2 Seleccionar la opción "Agregar Python 3.10 al PATH".

Instalando Python

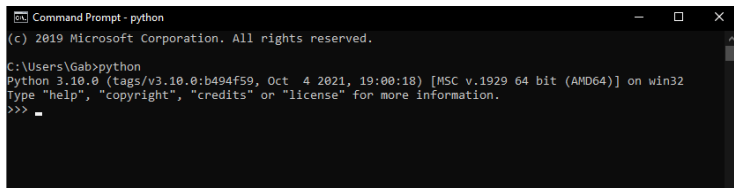
- 1 Hacer doble clic en el archivo ejecutable (.exe) descargado.
- 2 Seleccionar la opción "Agregar Python 3.10 al PATH".
- 3 Hacer clic en "Install" y esperar a que finalice la instalación.

Instalando Python

- 1 Hacer doble clic en el archivo ejecutable (.exe) descargado.
- 2 Seleccionar la opción "Agregar Python 3.10 al PATH".
- 3 Hacer clic en "Install" y esperar a que finalice la instalación.
- 4 Hacer clic en "Close".

Verificando la instalación

Para verificar que Python se ha instalado correctamente, se puede abrir una ventana de línea de comandos y escribir "python" (sin comillas). Debería aparecer algo como esto:



```
Command Prompt - python
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\Gab>python
Python 3.10.0 (tags/v3.10.0:b494f59, Oct 4 2021, 19:00:18) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> _
```

Esto significa que Python se ha instalado correctamente y se puede comenzar a usar.

Agregar Python a PATH manualmente en Windows

- 1 Abre el menú de inicio de Windows y busca "Configuración avanzada del sistema".

Agregar Python a PATH manualmente en Windows

- 1 Abre el menú de inicio de Windows y busca "Configuración avanzada del sistema".
- 2 Haz clic en "Variables de entorno".

Agregar Python a PATH manualmente en Windows

- 1 Abre el menú de inicio de Windows y busca "Configuración avanzada del sistema".
- 2 Haz clic en "Variables de entorno".
- 3 Busca la variable "PATH" en la sección "Variables del sistema" y selecciónala.

Agregar Python a PATH manualmente en Windows

- 1 Abre el menú de inicio de Windows y busca "Configuración avanzada del sistema".
- 2 Haz clic en "Variables de entorno".
- 3 Busca la variable "PATH" en la sección "Variables del sistema" y selecciónala.
- 4 Haz clic en "Editar".

Agregar Python a PATH manualmente en Windows

- 1 Abre el menú de inicio de Windows y busca "Configuración avanzada del sistema".
- 2 Haz clic en "Variables de entorno".
- 3 Busca la variable "PATH" en la sección "Variables del sistema" y selecciónala.
- 4 Haz clic en "Editar".
- 5 En la ventana "Editar variables del sistema", haz clic en "Nuevo" y escribe la ruta de la carpeta de Python que deseas agregar a PATH: "C:\Users\USER\AppData\Local\Programs\Python\Python3X" y "C:\Users\USER\AppData\Local\Programs\Python\Python3X\scripts".

Agregar Python a PATH manualmente en Windows

- 1 Abre el menú de inicio de Windows y busca "Configuración avanzada del sistema".
- 2 Haz clic en "Variables de entorno".
- 3 Busca la variable "PATH" en la sección "Variables del sistema" y selecciónala.
- 4 Haz clic en "Editar".
- 5 En la ventana "Editar variables del sistema", haz clic en "Nuevo" y escribe la ruta de la carpeta de Python que deseas agregar a PATH: "C:\Users\USER\AppData\Local\Programs\Python\Python3X" y "C:\Users\USER\AppData\Local\Programs\Python\Python3X\scripts".
- 6 Haz clic en "Aceptar" para guardar los cambios.

Instalación de Jupyter Notebook, NumPy y Matplotlib con cmd

- 1 Abre la ventana del símbolo del sistema (cmd) como administrador.

Instalación de Jupyter Notebook, NumPy y Matplotlib con cmd

- 1 Abre la ventana del símbolo del sistema (cmd) como administrador.
- 2 Para instalar Jupyter Notebook, escribe el siguiente comando y presiona Enter:

```
pip install jupyter numpy matplotlib
```

Instalación de Jupyter Notebook, NumPy y Matplotlib con cmd

- 1 Abre la ventana del símbolo del sistema (cmd) como administrador.
- 2 Para instalar Jupyter Notebook, escribe el siguiente comando y presiona Enter:

```
pip install jupyter numpy matplotlib
```

- 3 Espera a que se complete cada instalación y cierra la ventana del símbolo del sistema.

Instalación de Jupyter Notebook, NumPy y Matplotlib con cmd

- 1 Abre la ventana del símbolo del sistema (cmd) como administrador.
- 2 Para instalar Jupyter Notebook, escribe el siguiente comando y presiona Enter:

```
pip install jupyter numpy matplotlib
```

- 3 Espera a que se complete cada instalación y cierra la ventana del símbolo del sistema.
- 4 Abre Jupyter Notebook desde el menú de inicio o escribiendo `jupyter notebook` en la línea de comandos.

- NumPy es una biblioteca de Python para el cómputo numérico.

- NumPy es una biblioteca de Python para el cómputo numérico.
- Proporciona un objeto de matriz poderoso, que le permite realizar operaciones matemáticas complejas en conjuntos de datos grandes.

- NumPy es una biblioteca de Python para el cómputo numérico.
- Proporciona un objeto de matriz poderoso, que le permite realizar operaciones matemáticas complejas en conjuntos de datos grandes.
- NumPy también incluye muchas funciones útiles para álgebra lineal, transformadas de Fourier y generación de números aleatorios.

- NumPy es una biblioteca de Python para el cómputo numérico.
- Proporciona un objeto de matriz poderoso, que le permite realizar operaciones matemáticas complejas en conjuntos de datos grandes.
- NumPy también incluye muchas funciones útiles para álgebra lineal, transformadas de Fourier y generación de números aleatorios.
- NumPy es una herramienta fundamental para muchas otras bibliotecas de Python, incluyendo Matplotlib.

Importando NumPy

Para utilizar NumPy en Python, primero debes importar la biblioteca:

```
import numpy as np
```

A partir de ahora, podemos usar la abreviatura "np" para referirnos a la biblioteca NumPy.

Creando un array NumPy

Para crear un array NumPy, podemos utilizar la función `numpy.array()`.

```
import numpy as np
```

Crear un array de una lista

```
a = np.array([1, 2, 3])
```

```
print(a)
```

Crear un array de dos dimensiones

```
b = np.array([[1, 2], [3, 4]])
```

```
print(b)
```

Podemos acceder a elementos individuales de un array NumPy utilizando su índice.

```
import numpy as np
```

```
a = np.array([1, 2, 3])
```

Acceder al primer elemento

```
print(a[0])
```

Acceder a un rango de elementos

```
print(a[1:3])
```

Operaciones con arrays

Podemos realizar operaciones aritméticas y otras operaciones matemáticas con arrays NumPy.

```
import numpy as np
```

```
a = np.array([1, 2, 3])
```

```
b = np.array([4, 5, 6])
```

Sumar dos arrays

```
c = a + b
```

Multiplicar un array por un escalar

```
d = 2 * a
```

Calcular la media de un array

```
e = np.mean(a)
```

Operaciones con arrays

```
import numpy as np

# Crear un array
a = np.array([1, 2, 3, 4, 5])

# Encontrar el valor máximo
max_val = np.max(a)

# Encontrar el valor mínimo
min_val = np.min(a)

# Sumar los valores del array
sum_val = np.sum(a)
```

Operaciones con arrays

```
import numpy as np

a = np.array([1, 2, 3])
b = np.array([4, 5, 6])

# Multiplicación elemento a elemento
c = np.multiply(a, b)

# Producto punto (dot product)
d = np.dot(a, b)

# Producto cruz (cross product)
e = np.cross(a, b)

# Multiplicación matricial
f = a @ b
```

Operaciones con arrays

```
import numpy as np

# np.arange()
arr1 = np.arange(0, 10, 2)
print(arr1)  # Output: [0 2 4 6 8]

# np.zeros()
arr2 = np.zeros(5)
print(arr2)  # Output: [0. 0. 0. 0. 0.]

# np.linspace()
arr3 = np.linspace(0, 10, num=5)
print(arr3)  # Output: [ 0.   2.5  5.   7.5 10. ]
```


Normas de vectores

```
import numpy as np

v = np.array([1, 2, 3, 4, 5])

# Norma 1 (suma de valores absolutos)
norm1 = np.linalg.norm(v, ord=1)

# Norma 2 (raiz cuadrada de la suma de cuadrados)
norm2 = np.linalg.norm(v, ord=2)

# Norma infinita (máximo valor absoluto)
norm_inf = np.linalg.norm(v, ord=np.inf)
```

- Matplotlib es una biblioteca de Python para visualización de datos.

- Matplotlib es una biblioteca de Python para visualización de datos.
- Proporciona una amplia gama de herramientas de trazado, incluyendo gráficos de líneas, gráficos de dispersión, gráficos de barras y más.

- Matplotlib es una biblioteca de Python para visualización de datos.
- Proporciona una amplia gama de herramientas de trazado, incluyendo gráficos de líneas, gráficos de dispersión, gráficos de barras y más.
- Matplotlib es altamente personalizable, lo que le permite crear figuras de calidad profesional con solo unas pocas líneas de código.

- Matplotlib es una biblioteca de Python para visualización de datos.
- Proporciona una amplia gama de herramientas de trazado, incluyendo gráficos de líneas, gráficos de dispersión, gráficos de barras y más.
- Matplotlib es altamente personalizable, lo que le permite crear figuras de calidad profesional con solo unas pocas líneas de código.
- Matplotlib funciona perfectamente con matrices NumPy, lo que lo convierte en una herramienta poderosa para explorar y visualizar datos.

Tamaño de letras de título, leyenda y ejes

- En Matplotlib, el tamaño de las fuentes se puede ajustar para los títulos, la leyenda y los ejes.

Tamaño de letras de título, leyenda y ejes

- En Matplotlib, el tamaño de las fuentes se puede ajustar para los títulos, la leyenda y los ejes.
- Para cambiar el tamaño de la fuente de un título, se puede usar el parámetro `fontsize` de la función `title`.

Tamaño de letras de título, leyenda y ejes

- En Matplotlib, el tamaño de las fuentes se puede ajustar para los títulos, la leyenda y los ejes.
- Para cambiar el tamaño de la fuente de un título, se puede usar el parámetro `fontsize` de la función `title`.
- Para cambiar el tamaño de la fuente de la leyenda, se puede usar el parámetro `fontsize` de la función `legend`.

Tamaño de letras de título, leyenda y ejes

- En Matplotlib, el tamaño de las fuentes se puede ajustar para los títulos, la leyenda y los ejes.
- Para cambiar el tamaño de la fuente de un título, se puede usar el parámetro `fontsize` de la función `title`.
- Para cambiar el tamaño de la fuente de la leyenda, se puede usar el parámetro `fontsize` de la función `legend`.
- Para cambiar el tamaño de la fuente de los ejes, se puede usar el parámetro `fontsize` de las funciones `xlabel` y `ylabel`.

Tamaño de letras de título, leyenda y ejes

- En Matplotlib, el tamaño de las fuentes se puede ajustar para los títulos, la leyenda y los ejes.
- Para cambiar el tamaño de la fuente de un título, se puede usar el parámetro `fontsize` de la función `title`.
- Para cambiar el tamaño de la fuente de la leyenda, se puede usar el parámetro `fontsize` de la función `legend`.
- Para cambiar el tamaño de la fuente de los ejes, se puede usar el parámetro `fontsize` de las funciones `xlabel` y `ylabel`.
- Estos parámetros toman un número entero que representa el tamaño de la fuente en puntos.

- En Matplotlib, se pueden utilizar diferentes tipos de marcadores para representar los datos en un gráfico.

Tipos de marker

- En Matplotlib, se pueden utilizar diferentes tipos de marcadores para representar los datos en un gráfico.
- Los marcadores se especifican utilizando el parámetro marker en las funciones de trazado como plot o scatter.

Tipos de marker

- En Matplotlib, se pueden utilizar diferentes tipos de marcadores para representar los datos en un gráfico.
- Los marcadores se especifican utilizando el parámetro `marker` en las funciones de trazado como `plot` o `scatter`.
- Algunos ejemplos de marcadores incluyen: `o` para círculos, `s` para cuadrados, `*` para estrellas...

Tipos de marker

- En Matplotlib, se pueden utilizar diferentes tipos de marcadores para representar los datos en un gráfico.
- Los marcadores se especifican utilizando el parámetro `marker` en las funciones de trazado como `plot` o `scatter`.
- Algunos ejemplos de marcadores incluyen: `o` para círculos, `s` para cuadrados, `*` para estrellas...
- También se pueden especificar el tamaño y el color del marcador utilizando los parámetros `markersize` y `markeredgecolor`, respectivamente.

Tipos de marker

- En Matplotlib, se pueden utilizar diferentes tipos de marcadores para representar los datos en un gráfico.
- Los marcadores se especifican utilizando el parámetro `marker` en las funciones de trazado como `plot` o `scatter`.
- Algunos ejemplos de marcadores incluyen: `o` para círculos, `s` para cuadrados, `*` para estrellas...
- También se pueden especificar el tamaño y el color del marcador utilizando los parámetros `markersize` y `markeredgecolor`, respectivamente.
- Los marcadores también pueden ser personalizados utilizando la función `Path` de la biblioteca `matplotlib.path`.

- En Matplotlib, se pueden cambiar los colores de las líneas, los marcadores y las áreas rellenas en un gráfico.

- En Matplotlib, se pueden cambiar los colores de las líneas, los marcadores y las áreas rellenas en un gráfico.
- Los colores se especifican utilizando el parámetro color en las funciones de trazado como plot, scatter...

- En Matplotlib, se pueden cambiar los colores de las líneas, los marcadores y las áreas rellenas en un gráfico.
- Los colores se especifican utilizando el parámetro color en las funciones de trazado como plot, scatter...
- Algunos ejemplos de colores incluyen: b para azul, g para verde, r para rojo, c para cian, m para magenta, y para amarillo, y k para negro.

- En Matplotlib, se pueden cambiar los colores de las líneas, los marcadores y las áreas rellenas en un gráfico.
- Los colores se especifican utilizando el parámetro color en las funciones de trazado como plot, scatter...
- Algunos ejemplos de colores incluyen: b para azul, g para verde, r para rojo, c para cian, m para magenta, y para amarillo, y k para negro.
- También se pueden especificar colores utilizando valores RGB (rojo-verde-azul) o hexadecimal.

- En Matplotlib, se pueden cambiar los colores de las líneas, los marcadores y las áreas rellenas en un gráfico.
- Los colores se especifican utilizando el parámetro color en las funciones de trazado como plot, scatter...
- Algunos ejemplos de colores incluyen: b para azul, g para verde, r para rojo, c para cian, m para magenta, y para amarillo, y k para negro.
- También se pueden especificar colores utilizando valores RGB (rojo-verde-azul) o hexadecimal.
- Además, Matplotlib proporciona una serie de mapas de colores predefinidos que se pueden utilizar con la función colormap.

Varios gráficos en una imagen

A veces queremos mostrar varios gráficos en una sola imagen. Para hacer esto, podemos utilizar la función `subplot()`. Esta función nos permite dividir la figura en una cuadrícula de subgráficos y especificar en cuál subgráfico queremos dibujar nuestro gráfico.

Sintaxis de `subplot()`

```
subplot(nrows, ncols, index)
```

donde `nrows` es el número de filas de subgráficos, `ncols` es el número de columnas de subgráficos y `index` es el índice del subgráfico actual.

Diferentes tipos de gráficos

Matplotlib ofrece una amplia variedad de gráficos para visualizar diferentes tipos de datos. Algunos de los gráficos más comunes incluyen:

- Gráficos de línea

Cada tipo de gráfico tiene su propio conjunto de parámetros y opciones de estilo que se pueden ajustar para personalizar el gráfico.

Diferentes tipos de gráficos

Matplotlib ofrece una amplia variedad de gráficos para visualizar diferentes tipos de datos. Algunos de los gráficos más comunes incluyen:

- Gráficos de línea
- Gráficos de barras

Cada tipo de gráfico tiene su propio conjunto de parámetros y opciones de estilo que se pueden ajustar para personalizar el gráfico.

Diferentes tipos de gráficos

Matplotlib ofrece una amplia variedad de gráficos para visualizar diferentes tipos de datos. Algunos de los gráficos más comunes incluyen:

- Gráficos de línea
- Gráficos de barras
- Gráficos de dispersión

Cada tipo de gráfico tiene su propio conjunto de parámetros y opciones de estilo que se pueden ajustar para personalizar el gráfico.

Diferentes tipos de gráficos

Matplotlib ofrece una amplia variedad de gráficos para visualizar diferentes tipos de datos. Algunos de los gráficos más comunes incluyen:

- Gráficos de línea
- Gráficos de barras
- Gráficos de dispersión
- Gráficos de áreas

Cada tipo de gráfico tiene su propio conjunto de parámetros y opciones de estilo que se pueden ajustar para personalizar el gráfico.

- Los cuadernos Jupyter, Python, NumPy y Matplotlib son herramientas esenciales para la informática científica y el análisis de datos en Python.

- Los cuadernos Jupyter, Python, NumPy y Matplotlib son herramientas esenciales para la informática científica y el análisis de datos en Python.
- Proporcionan un entorno flexible para explorar y visualizar datos, y se pueden utilizar para una amplia gama de aplicaciones.

- Los cuadernos Jupyter, Python, NumPy y Matplotlib son herramientas esenciales para la informática científica y el análisis de datos en Python.
- Proporcionan un entorno flexible para explorar y visualizar datos, y se pueden utilizar para una amplia gama de aplicaciones.
- Si eres nuevo en estas herramientas, hay muchos tutoriales y recursos disponibles en línea para ayudarte a comenzar.

¡¡Gracias por su atención!!