Creación de un ambiente conda para trabajar con Jupyter notebook

Profesor: Joaquín Fontbona T.
Auxiliares: Pablo Zúñiga Rodríguez-Peña, Arie Wortsman Z., Camilo Carvajal Reyes

Conda es un sistema de manejo de paquetes de código abierto, originalmente diseñado para el lenguaje de programación *python*. *Conda* nos permite crear y administrar ambientes que contengan las dependencias sobre las cuales construiremos nuestro código.

A continuación mostramos los pasos básicos a seguir para la instalación de miniconda, según el sistema operativo, además de la creación de un ambiente con bibliotecas que serán útiles para el curso, incluyendo jupyter.

1. Instalación

La instalación de conda depende del sistema operativo.

1.1. Windows

1. Descargar el instalador apropiado según las características del computador (64 o 32 bits) en este enlace.
2. Hacer doble click en el archivo .exe instalado, donde se abrirá una ventana como en la figura 1 y seguir las instrucciones, con los parámetros de instalación por defecto.

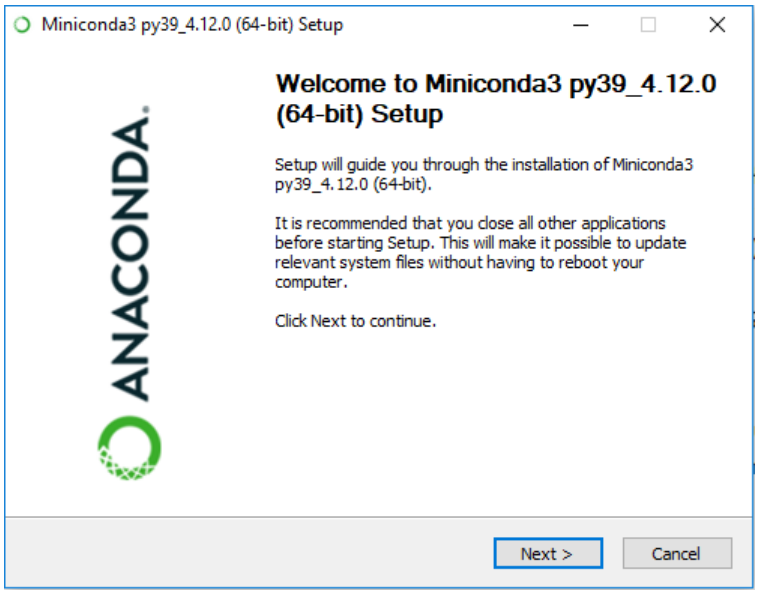

Figura 1: Instalador de *miniconda*

.

3. En la barra de búsqueda de windows escribir “anaconda” y seleccionar **Anaconda Prompt** como se muestra en la figura 2.

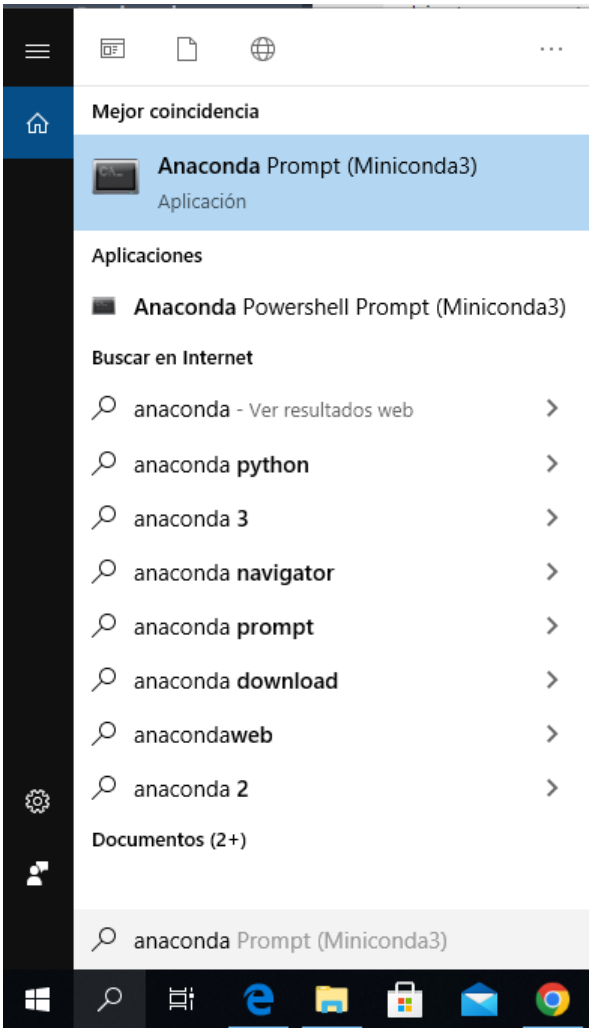

Figura 2: Selección de Anaconda Prompt

. Esto abrirá una terminal para usar *miniconda*.

1.2. macOS

1. En la barra inferior presionar el ícono Launchpad, luego escribir “terminal” en el campo de búsqueda, como se muestra en la figura 3

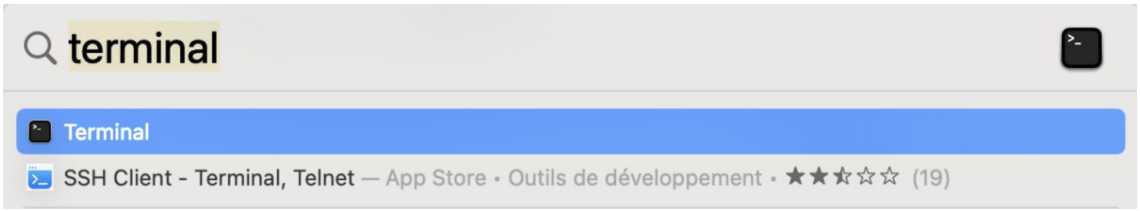

Figura 3: Abriendo la terminal de macOS

.

2. Descargar el instalador para macOS. Hay dos opciones para esto, una es usar directamente la terminal al correr lo siguiente:

```
wget https://repo.anaconda.com/miniconda/Miniconda3-latest-MacOSX-x86_64.sh -O ~/miniconda.sh
```

Esto requiere tener instalado *wget*, que permite la descarga de archivos usando la linea de comando.
Otra opción es descargar el instalador *bash* manualmente desde este enlace. Supongamos que el archivo fue descargado a la carpeta "~/mi/directorio/", entonces ejecutamos

```
mv ~/mi/directorio/Miniconda3-latest-MacOSX-x86_64.sh ~/miniconda.sh
```

3. Procedemos a la instalación ejecutando

```
bash ~/miniconda.sh -b -p \$HOME/miniconda
```

Este modo de instalación asume que se acepta la licencia de uso. El instalador preguntará si ce acepta inicializar miniconda3 usando "conda init" . Necesariamente debemos aceptar esta opción.

4. Para finalizar la instalación debemos cerrar la terminal.

2. Creación de un ambiente Conda

En esta sección usaremos miniconda para crear un ambiente *python*. Usaremos una terminal, que en el caso de Windows puede ser Anaconda Prompt. Para sistemas Unix podemos usar la terminal integrada siempre y cuando hayamos instalado conda/miniconda (ver sección anterior).

1. Crearemos un ambiente conda para *python* 3.9, que contará con las bibliotecas *numpy* . En este caso este se llamará *MA4402*, pero pueden cambiar este nombre si lo desean. Para esto debemos abrir una terminal y escribir

```
conda create --name MA4402 python=3.9 numpy
```

La terminal debería verse como en la figura 4.

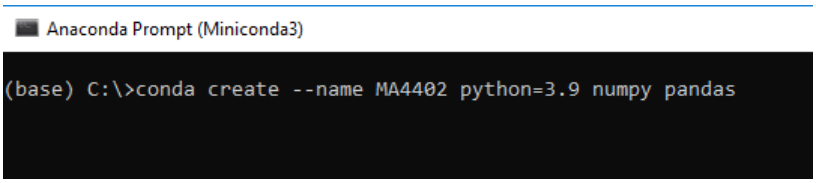

Figura 4: Creación de un ambiente

Luego, al presionar enter, la terminal se verá como en la figura 5:

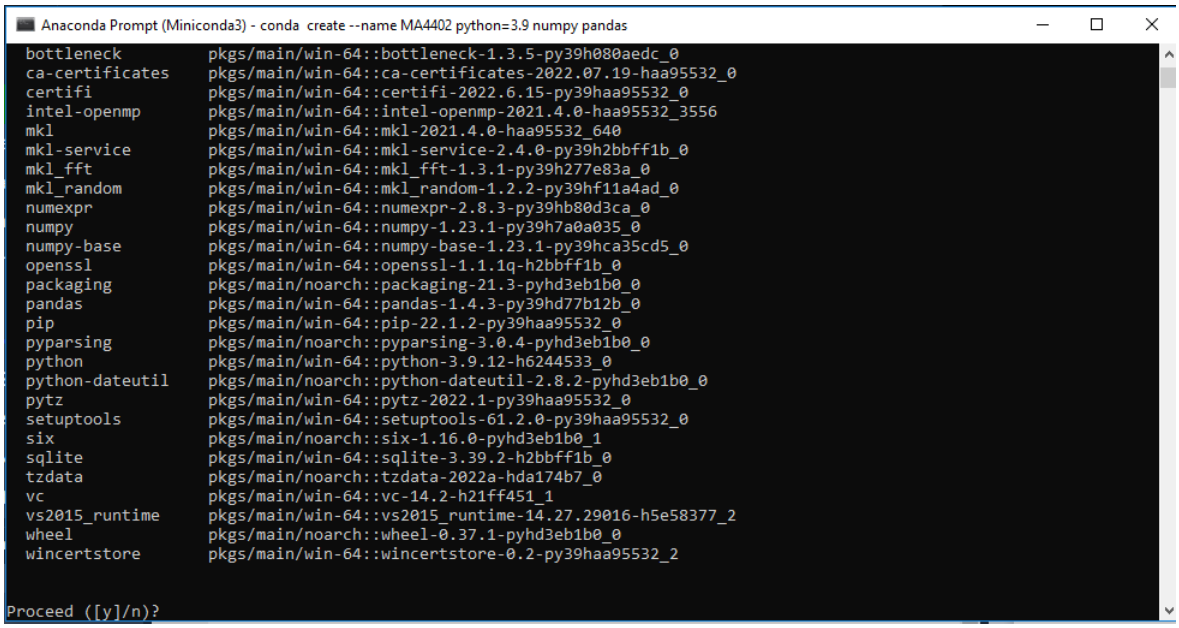

Figura 5: Instalación del ambiente conda

Debemos presionar la tecla “y” y luego enter. Al finalizar la instalación de nuestro ambiente deberíamos ver lo que se muestra en la figura 6.

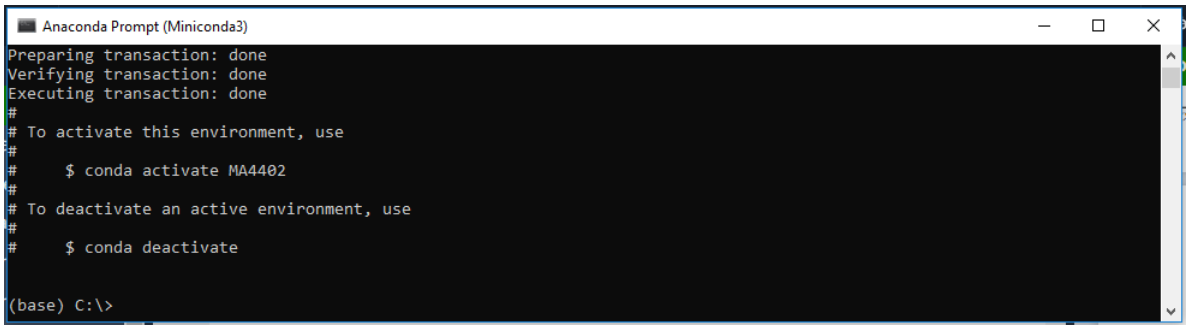

Figura 6: Finalización de instalación del ambiente conda

2. Cuando queramos acceder a nuestro proyecto y activar el uso de nuestro ambiente lo haremos como muestra la figura 7.

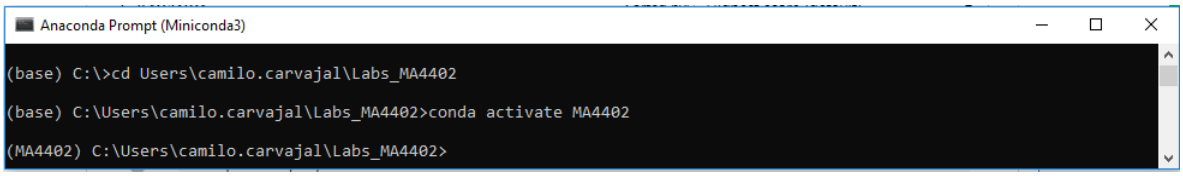

Figura 7: Activación del ambiente conda

En este caso nuestro directorio de trabajo se llama Labs_MA4402. En general ejecutaremos:

```
cd ruta_directorio
conda activate MA4402
```

El comando *cd* nos permite acceder a distintos directorios.

3. Ahora instalaremos más bibliotecas. Para aquello podemos usar distintos "canales de instalación". El más común es pip, aunque también podemos usar conda como instalador. Notar que algunos paquetes pueden estar disponibles en algunos instaladores y otros no.

   Como ejemplo, instalaremos la biblioteca *scipy* usando pip:

   ```
   pip install scipy
   ```

   Si todo sale bien veremos algo como en la figura 8.

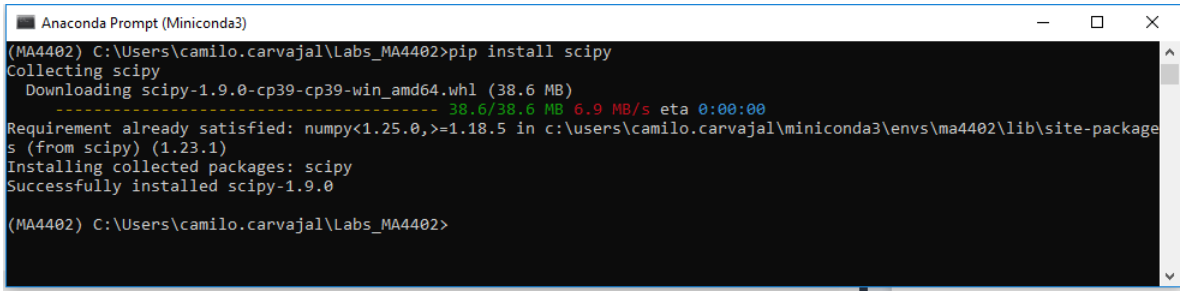

Figura 8: Instalación de *scipy* usando *pip*

4. Los comandos para ver que paquetes están instalados en el ambiente son:

   ```
   conda list
   pip list
   ```

3. Instalación de Jupyter

Jupyter notebooks nos permite combinar texto y código en un mismo documento. Usaremos esto para los informes de laboratorios.

1. Desde una terminal con nuestro ambiente activado ejecturaremos

```
pip install notebook
```

2. Una vez instalado el Notebook ejecutaremos

```
conda install ipykernel
```

3. Teniendo *ipykernel* instalado podremos hacer que Jupyter reconozca nuestro ambiente python para ejecutar nuestros Notebooks.

```
ipython kernel install --user --name=MA4402
```

4. Luego de haber realizado lo anterior, al ejecutar

```
jupyter kernelspec list
```

deberíamos ser capaces de ver nuestro ambiente, tal como en la figura 9.

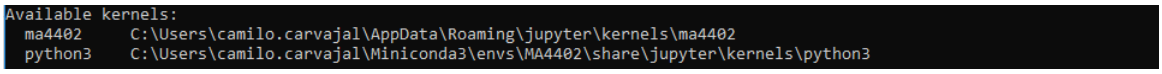

Figura 9: Kernels disponibles para Jupyter

5. Enseguida procedemos a abrir la interfaz de jupyter con el comando

```
jupyter notebook
```

En la terminal se observará algo como lo mostrado en la figura 10.

```
(MA4402) C:\Users\camilo.carvajal\Labs_MA4402>jupyter notebook
[I 17:12:13.419 NotebookApp] Serving notebooks from local directory: C:\Users\camilo.carvajal\Labs_MA4402
[I 17:12:13.419 NotebookApp] Jupyter Notebook 6.4.12 is running at:
[I 17:12:13.419 NotebookApp] http://localhost:8888/?token=cd5e98e572c66b56c89a8473e425efef412fad7b63a378ce
[I 17:12:13.419 NotebookApp]  or http://127.0.0.1:8888/?token=cd5e98e572c66b56c89a8473e425efef412fad7b63a378ce
[I 17:12:13.419 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 17:12:13.544 NotebookApp]

    To access the notebook, open this file in a browser:
        file:///C:/Users/camilo.carvajal/AppData/Roaming/jupyter/runtime/nbserver-14480-open.html
    Or copy and paste one of these URLs:
        http://localhost:8888/?token=cd5e98e572c66b56c89a8473e425efef412fad7b63a378ce
     or http://127.0.0.1:8888/?token=cd5e98e572c66b56c89a8473e425efef412fad7b63a378ce
```

Figura 10: Terminal al abrir jupyter

Como se menciona ahí, el comando para cerrar la sesión de jupyter es ctrl + c. Al mismo tiempo, se debió abrir la interfaz de jupyter notebook en el navegador predeterminado, como es el caso de la figura 11

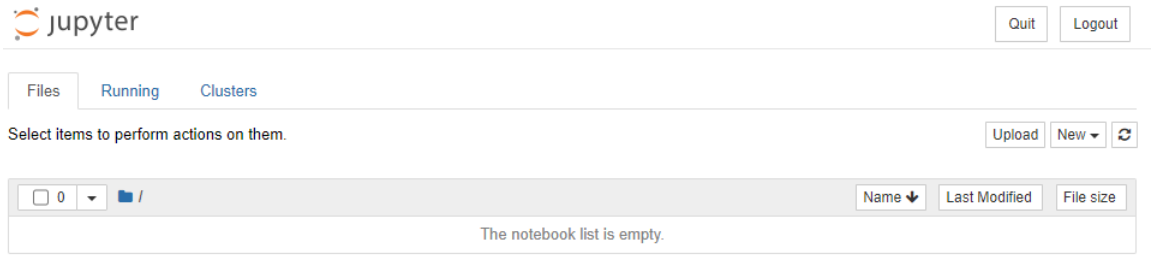

Figura 11: Interfaz de usuario de jupyter notebook en el navegador

Notar que esto no está usando internet, simplemente se trata de una interfaz tipo web que se abre en el navegador.

6. Podemos crear un nuevo notebook presionando “new” y seleccionando la kernel que acabamos de crear, como en la figura 12.

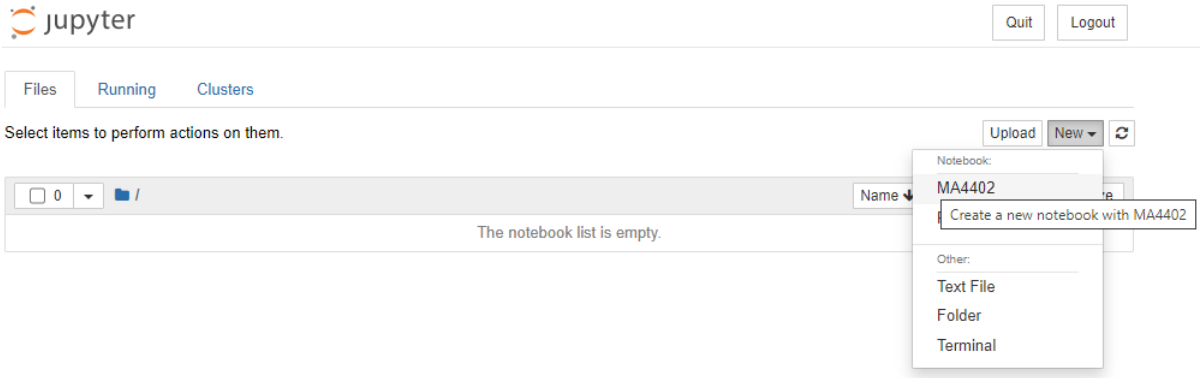

Figura 12: Creación de un jupyter notebook

Lo anterior abrirá una ventana como en la figura 13.

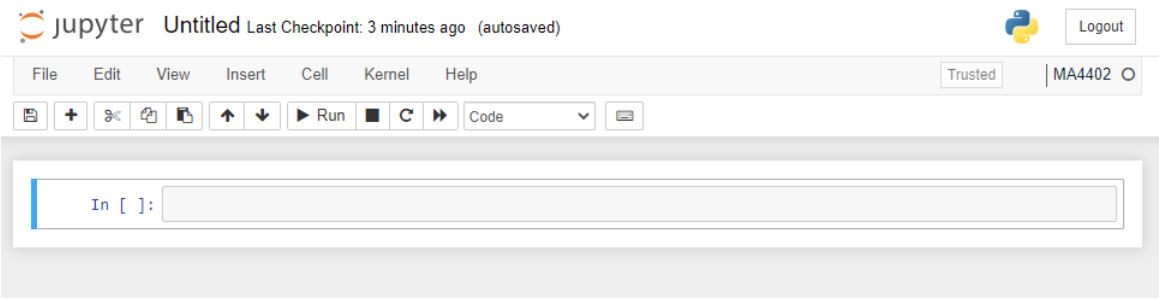

Figura 13: Nuevo jupyter notebook

7. Por último podemos editar nuestro notebook. En la figura 14 hemos cambiado el nombre de nuestro notebook a "notebook_test". Cambiamos una celda a modo *markdown*, en el cual podemos escribir cosas. Por último, importamos una de las bibliotecas instaladas, imprimimos un string y ejecutamos un cálculo.

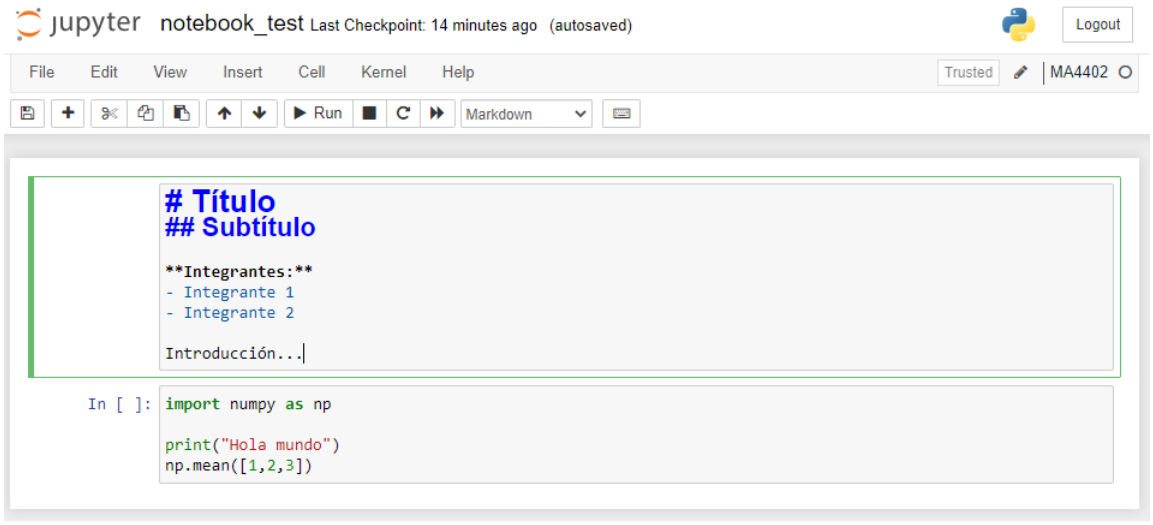

Figura 14: Nuevo jupyter notebook editado

El resultado de ejecutar las celdas correspondientes (con el botón superior *run* o bien con ctrl + enter) se muestra en la figura 15.

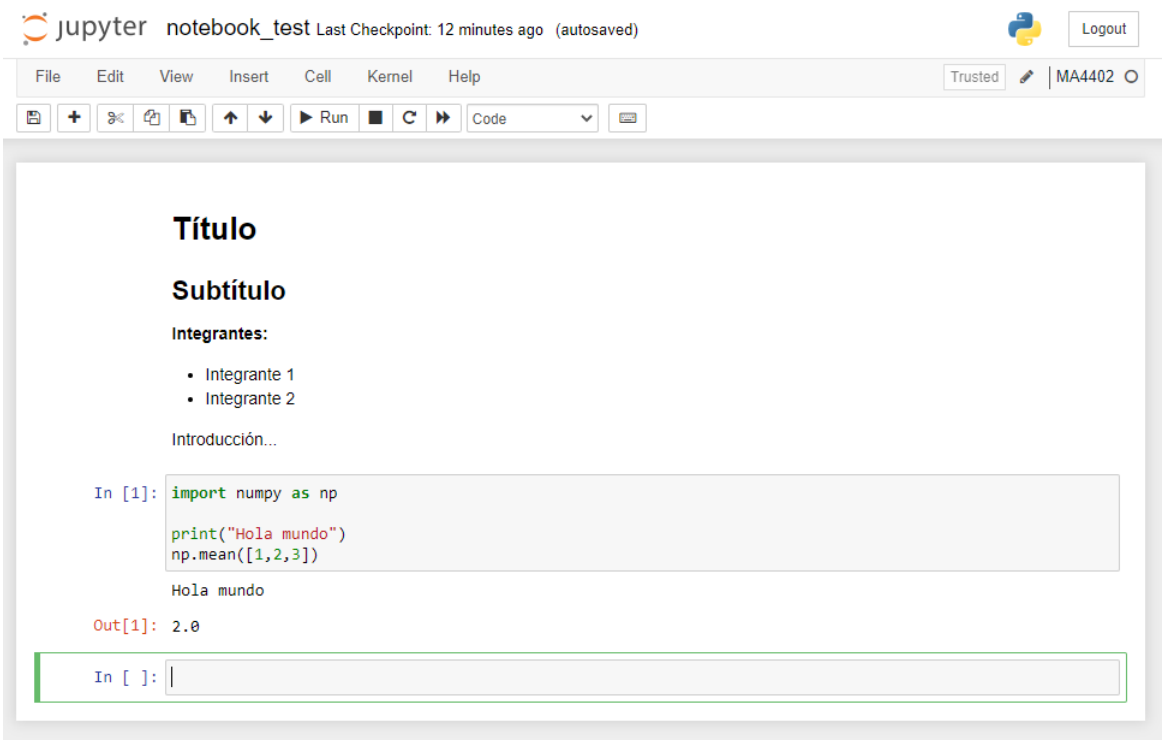

Figura 15: Nuevo jupyter notebook editado con celdas ejecutadas