



Tutorial para desarrolladores de Slicer: Programación en Slicer

Sonia Pujol, Ph.D.

Profesora adjunta de radiología

Directora de formación y educación en 3D Slicer

Hospital de mujeres de Brigham

Facultad de Medicina de Harvard

Steve Pieper, Ph.D.

Arquitecto jefe de 3D Slicer

Isomics Inc.

Objetivo del tutorial



```
def threshold(t):  
    n=getNode('T2')  
    a=array('T2')  
    a[a<t]=0  
    arrayFromVolumeModified(n)  
    print('Thresholding done')
```



Este tutorial es una introducción a la consola de Python y al widget Qt de la versión 5 de 3D Slicer



```
b=qt.QPushButton('Toggle')  
b.connect('clicked()',toggle)  
b.setStyleSheet = "font-size: 24pt; color:  
aqua; margin: 20px"  
b.show()
```

Esquema del Tutorial



Parte 1: Visión general de los módulos de 3D Slicer



Parte 2: Familiarizarse con el entorno Python en 3D Slicer

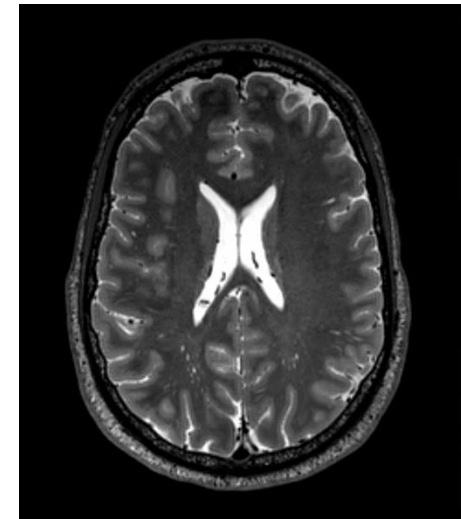
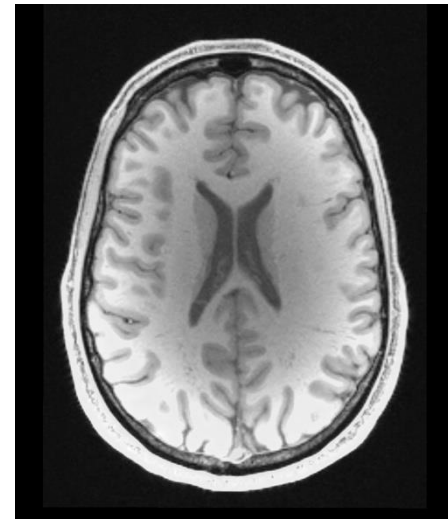
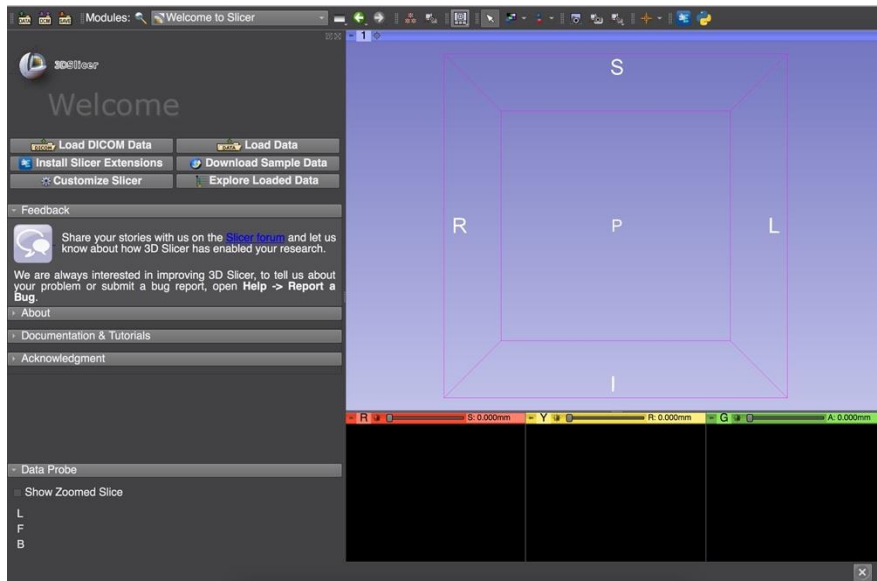


Parte 3: Familiarizarse con el conjunto de herramientas de widgets Qt en 3D Slicer

Descargo de responsabilidad

- 3D Slicer es una aplicación de software libre de código abierto distribuida bajo una licencia de estilo BSD.
- El software no cuenta con la aprobación de la FDA ni el marcado CE y es para uso exclusivo en investigación.

Materiales didácticos



SlicerProgrammingTutorialData.zip
[link](#)

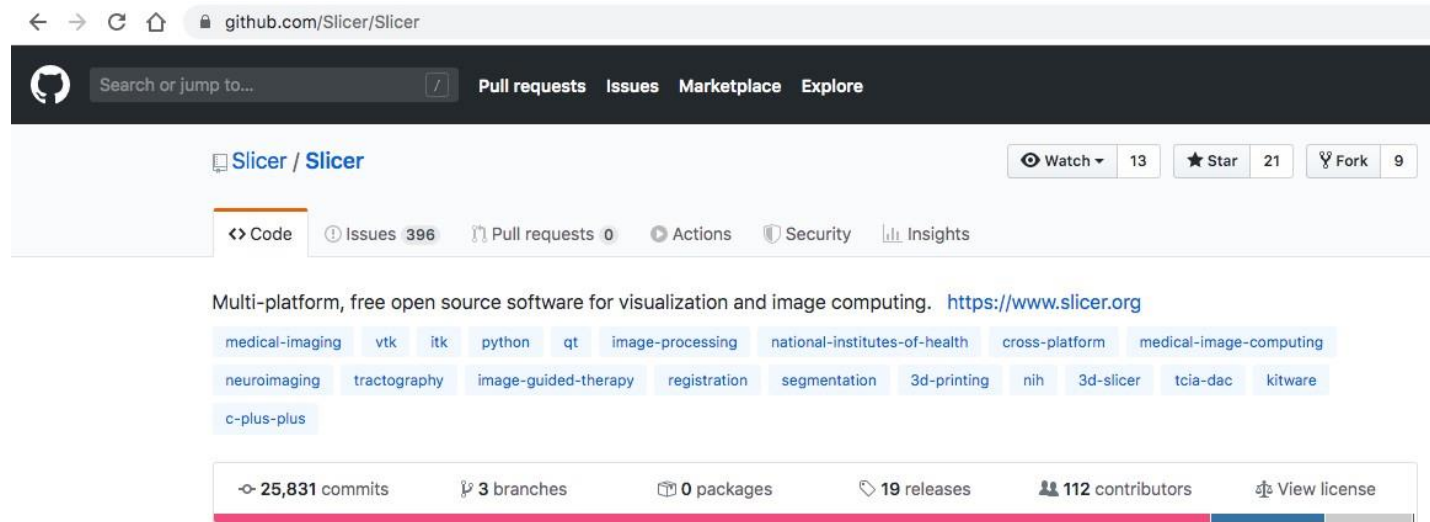
3D Slicer versión 4.11

Parte 1

Visión general de los módulos de Slicer



3D Slicer



- 3D Slicer es una plataforma de código abierto para el análisis y la visualización de datos de imágenes médicas.
- 3D Slicer se compila y se prueba a diario en plataformas Windows, Mac y Linux
- El código fuente está disponible gratuitamente en GitHub en <http://github.com/Slicer/Slicer>

Módulos Slicer

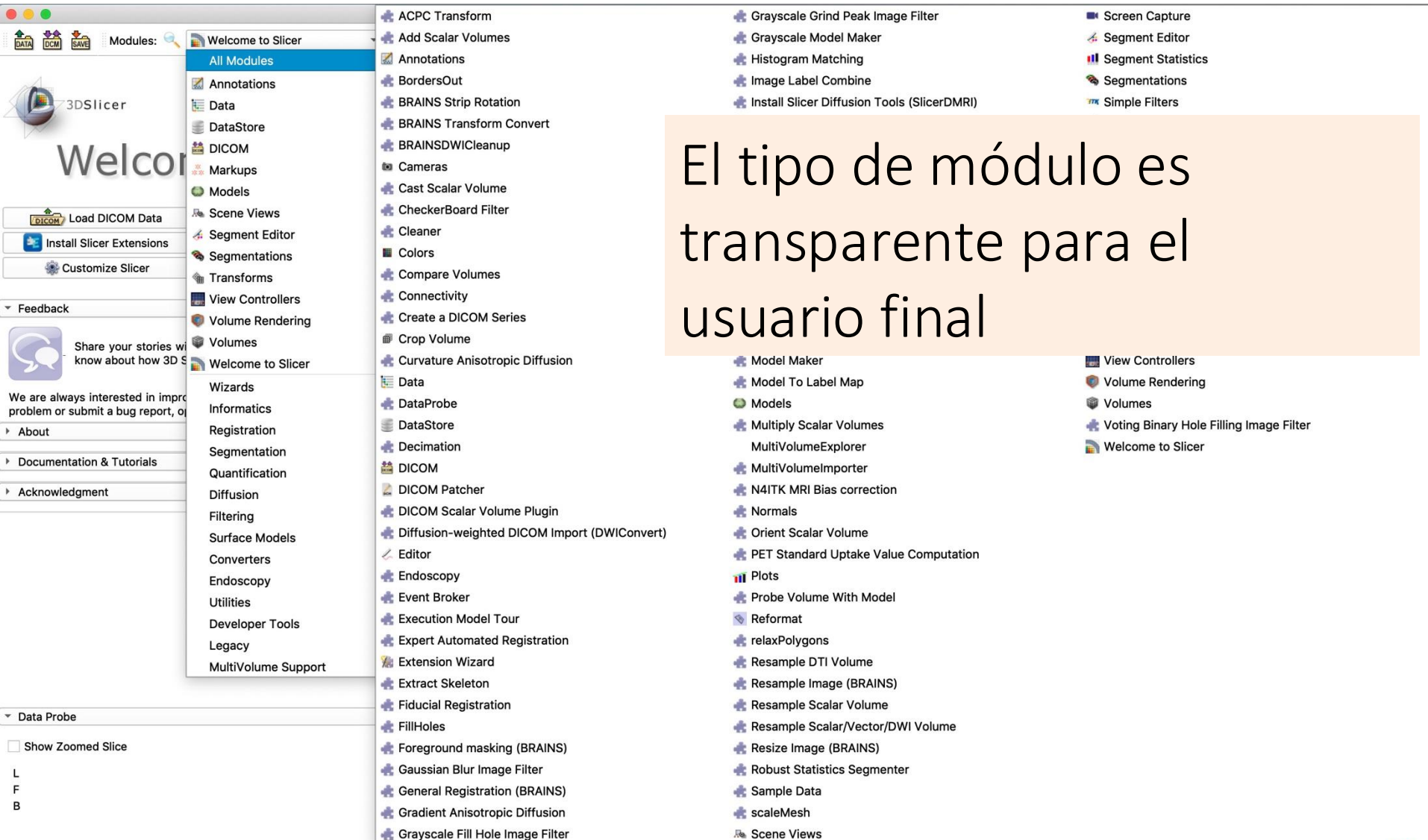
3D Slicer admite tres tipos de módulos:

- Interfaz de línea de comandos (CLI): ejecutor independiente con argumentos de entrada/salida limitados.
- Módulos cargables (C++ Plugins): optimizados para cálculos pesados.

Es el enfoque de este tutorial

- Módulos con script (Python): recomendados para la creación rápida de prototipos y el desarrollo de flujo de trabajo.

Módulos Slicer



The image shows the Slicer software interface with the 'Modules' menu open. The menu lists various modules, and a large orange box highlights the text 'El tipo de módulo es transparente para el usuario final'.

Modules:

- DATA
- DICOM
- SAVE

Welcome to Slicer

- All Modules
- Annotations
- Data
- DataStore
- DICOM
- Markups
- Models
- Scene Views
- Segment Editor
- Segmentations
- Transforms
- View Controllers
- Volume Rendering
- Volumes
- Welcome to Slicer
- Wizards
- Informatics
- Registration
- Segmentation
- Quantification
- Diffusion
- Filtering
- Surface Models
- Converters
- Endoscopy
- Utilities
- Developer Tools
- Legacy
- MultiVolume Support

Module List:

- ACPC Transform
- Add Scalar Volumes
- Annotations
- BordersOut
- BRAINS Strip Rotation
- BRAINS Transform Convert
- BRAINSDWICleanup
- Cameras
- Cast Scalar Volume
- CheckerBoard Filter
- Cleaner
- Colors
- Compare Volumes
- Connectivity
- Create a DICOM Series
- Crop Volume
- Curvature Anisotropic Diffusion
- Data
- DataProbe
- DataStore
- Decimation
- DICOM
- DICOM Patcher
- DICOM Scalar Volume Plugin
- Diffusion-weighted DICOM Import (DWIConvert)
- Editor
- Endoscopy
- Event Broker
- Execution Model Tour
- Expert Automated Registration
- Extension Wizard
- Extract Skeleton
- Fiducial Registration
- FillHoles
- Foreground masking (BRAINS)
- Gaussian Blur Image Filter
- General Registration (BRAINS)
- Gradient Anisotropic Diffusion
- Grayscale Fill Hole Image Filter
- Grayscale Grind Peak Image Filter
- Grayscale Model Maker
- Histogram Matching
- Image Label Combine
- Install Slicer Diffusion Tools (SlicerDMRI)
- Model Maker
- Model To Label Map
- Models
- Multiply Scalar Volumes
- MultiVolumeExplorer
- MultiVolumeImporter
- N4ITK MRI Bias correction
- Normals
- Orient Scalar Volume
- PET Standard Uptake Value Computation
- Plots
- Probe Volume With Model
- Reformat
- relaxPolygons
- Resample DTI Volume
- Resample Image (BRAINS)
- Resample Scalar Volume
- Resample Scalar/Vector/DWI Volume
- Resize Image (BRAINS)
- Robust Statistics Segmenter
- Sample Data
- scaleMesh
- Screen Capture
- Segment Editor
- Segment Statistics
- Segmentations
- Simple Filters
- View Controllers
- Volume Rendering
- Volumes
- Voting Binary Hole Filling Image Filter
- Welcome to Slicer

Data Probe

- Show Zoomed Slice

Feedback

- Share your stories with us and help us know about how 3D Slicer is used in your lab.
- We are always interested in improving 3D Slicer. If you find a problem or submit a bug report, or suggest a new feature, please contact us.

About

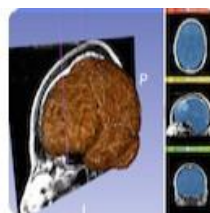
- Documentation & Tutorials
- Acknowledgment

Scene Views

- L
- F
- B

Extensiones Slicer

Una extensión Slicer es un paquete de entrega que incluye uno o varios módulos de Slicer



SwissSkullStripper
Bill Lorensen (Noware...)

★★★★★ (0)

INSTALL



PETTumorSegmenta...
Christian Bauer (Univ...)

★★★★★ (0)

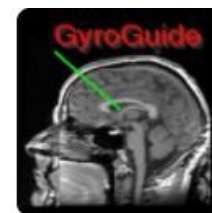
INSTALL



SlicerOpenIGTLink
Junichi Tokuda (SPL), ...

★★★★★ (0)

INSTALL



GyroGuide
Ruifeng Chen, Luping...

★★★★★ (0)

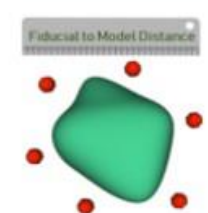
INSTALL



PET-IndiC
Ethan Ulrich (Universi...)

★★★★★ (0)

INSTALL



FiducialsToModelDi...
Jesse Reynolds (Cante...)

★★★★★ (0)

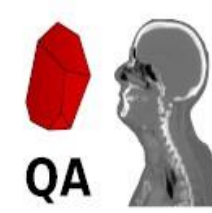
INSTALL



Slicer-Wasp
Thomas Lawson (MR...)

★★★★★ (0)

INSTALL



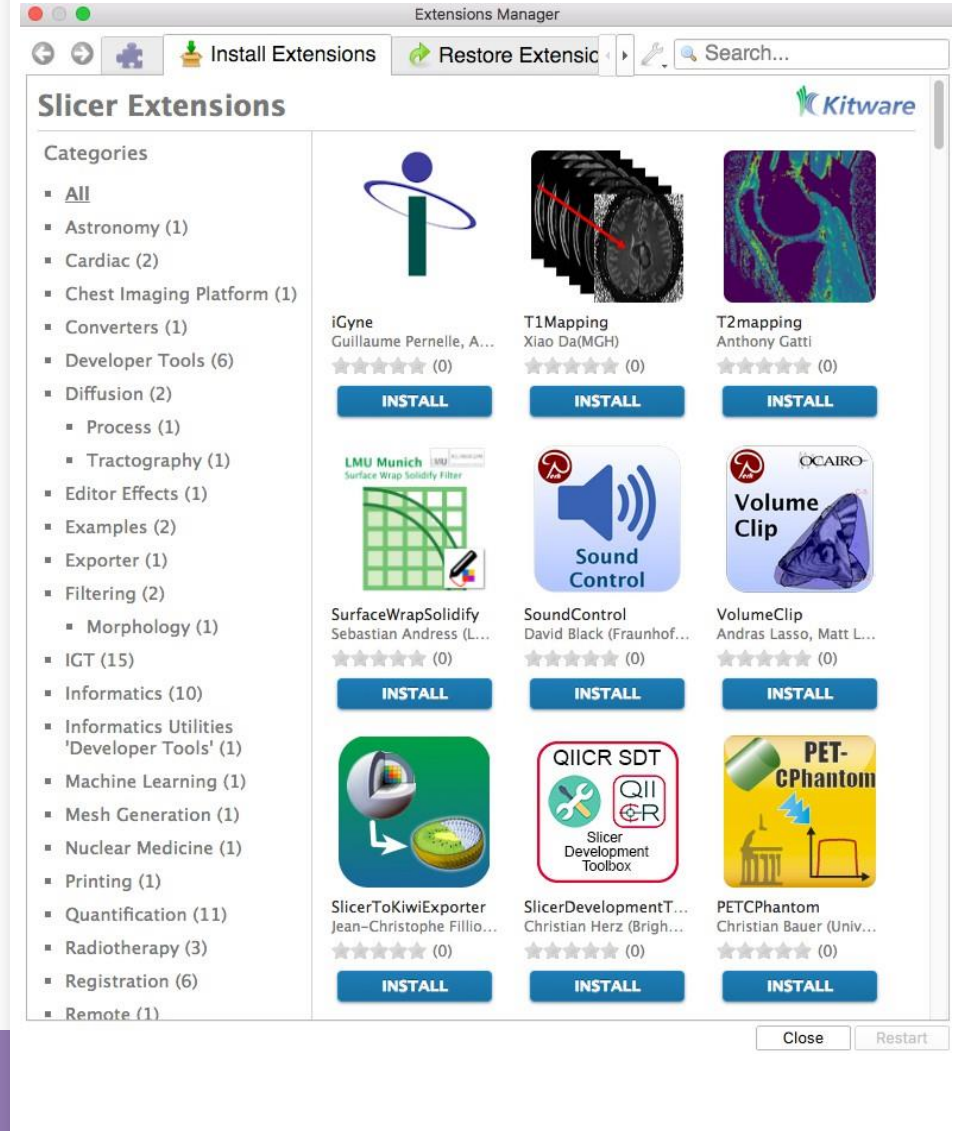
ImageCompare
Paolo Zaffino (Magna ...)

★★★★★ (0)

INSTALL

Administrador de extensiones de Slicer

- El administrador de extensiones de Slicer proporciona una plataforma de “tienda de aplicaciones” para el ecosistema de 3D Slicer
- El Extension Manager facilita la creación e instalación de extensiones de Slicer.
- La versión 5 de Slicer incluye más de 130 extensiones.



Parte 2

Familiarizarse con el entorno Python en 3D Slicer







Consola Python



```
Python 3.9.10 (main, Jan 23 2025, 23:04:06) [MSC v.1942 64 bit (AMD64)] on win32  
>>>
```

Python en Slicer

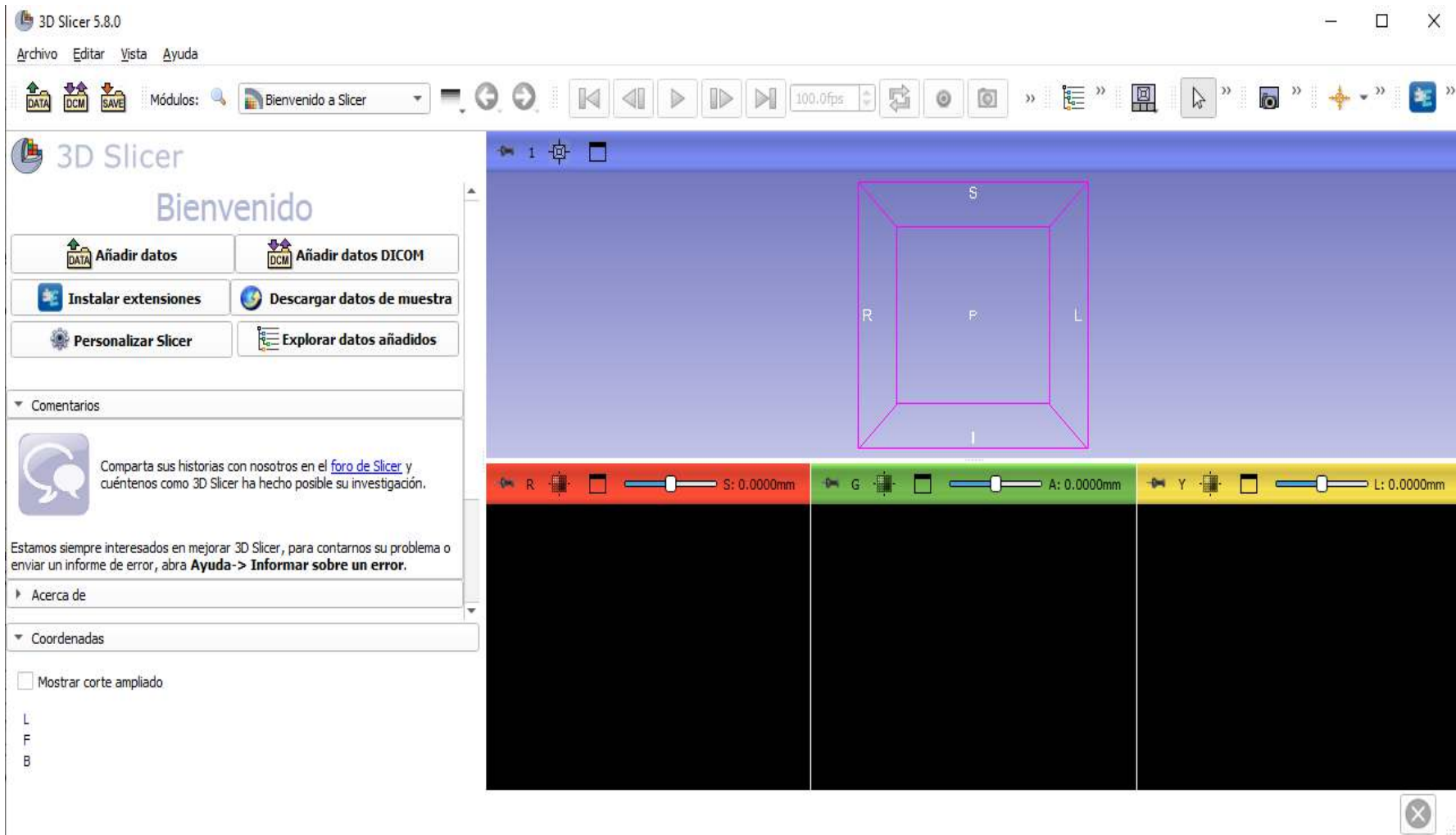
Slicer v.4.11 funciona con Python3 y un conjunto de bibliotecas estándar

	NumPy, el cual es el paquete fundamental para la computación científica con Python.
	VTK es una biblioteca de código abierto para la manipulación y visualización de datos científicos.
	ITK es una biblioteca de código abierto para el análisis de imágenes.
	CTK es una biblioteca de código abierto para el cálculo de imágenes biomédicas.
	PythonQT es un enlace de Python para Qt.
	Qt es un entorno de multiplataforma utilizado como conjunto de herramientas gráficas.

Python en Slicer

- El índice de paquetes de Python (PyPi) da acceso a más de 200.000 paquetes adicionales de Python (<http://pipy.org>).
- El comando “pip install” en Slicer permite a los desarrolladores instalar las herramientas de computación científica más comunes (por ejemplo, TensorFlow, SciPy, PyTorch, Pandas, etc.).
- Slicer puede utilizarse como núcleo de Jupyter notebook
- PyCharm y otras herramientas de desarrollo de Python se pueden utilizar con Slicer

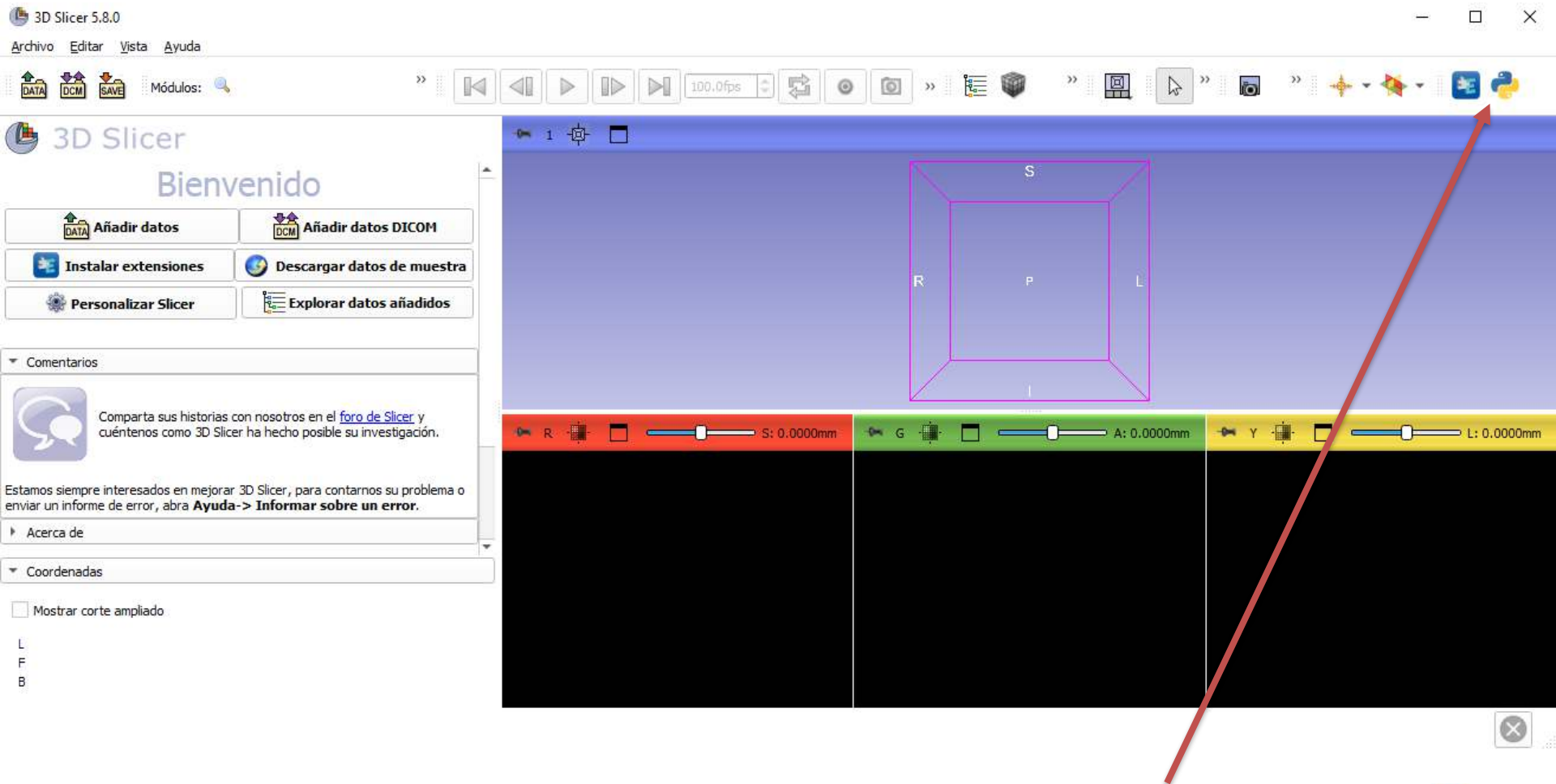




La version 5 de Slicer integra Python3, VTK5 and ITK5

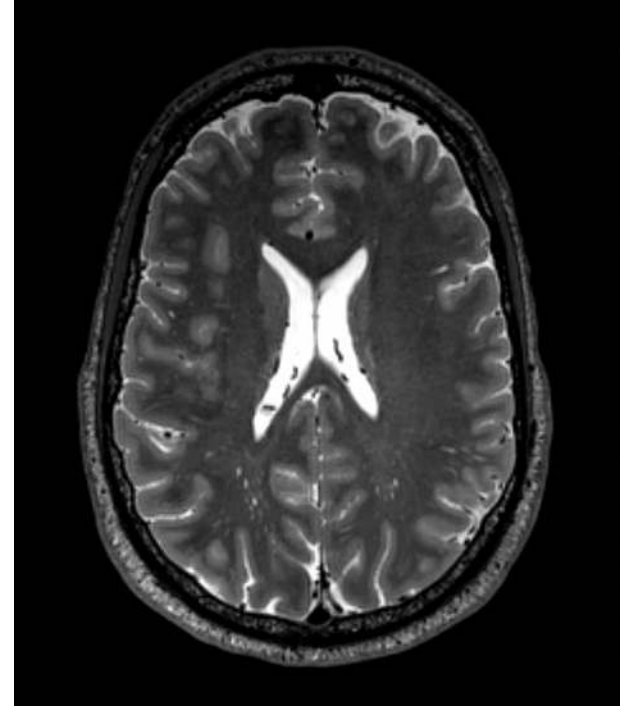
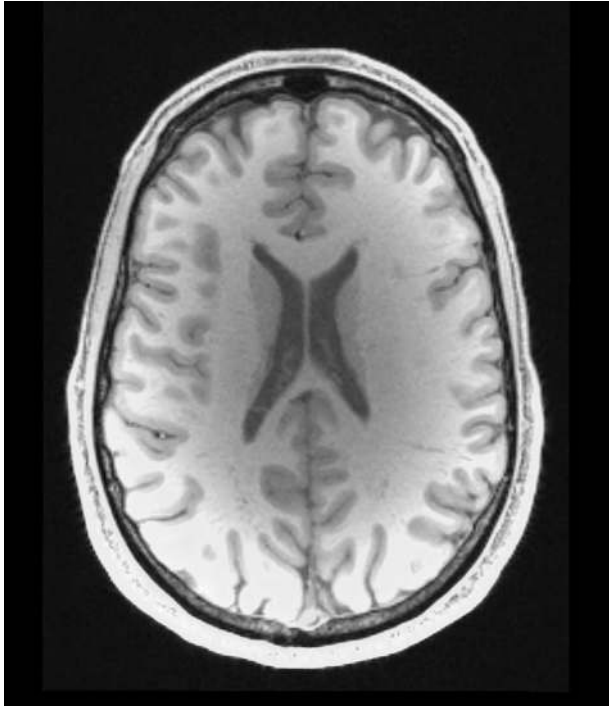
Consola de Python en Slicer

La consola de Python esta basada en Qt y permite un acceso directo a Slicer MRML Nodos, librerias (NumPy, VTK, ITK, CTK) y Qt.



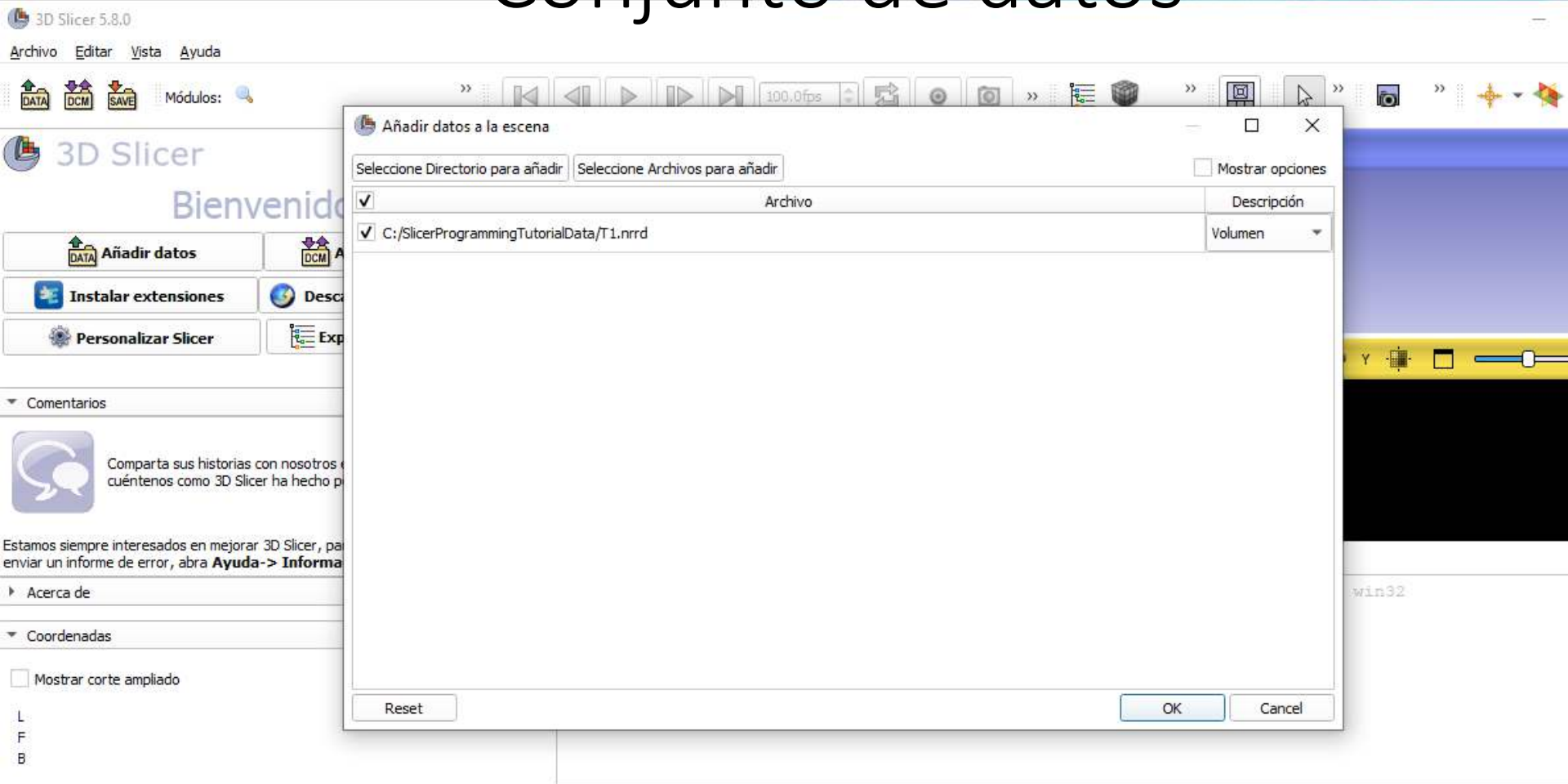
Para acceder a la Consola de Python, haga clic en el icono python en el menú de la barra superior de Slicer





El conjunto de datos del tutorial de programación Slicer incluye una resonancia magnética ponderada en T1 y otra en T2 de un sujeto sano.

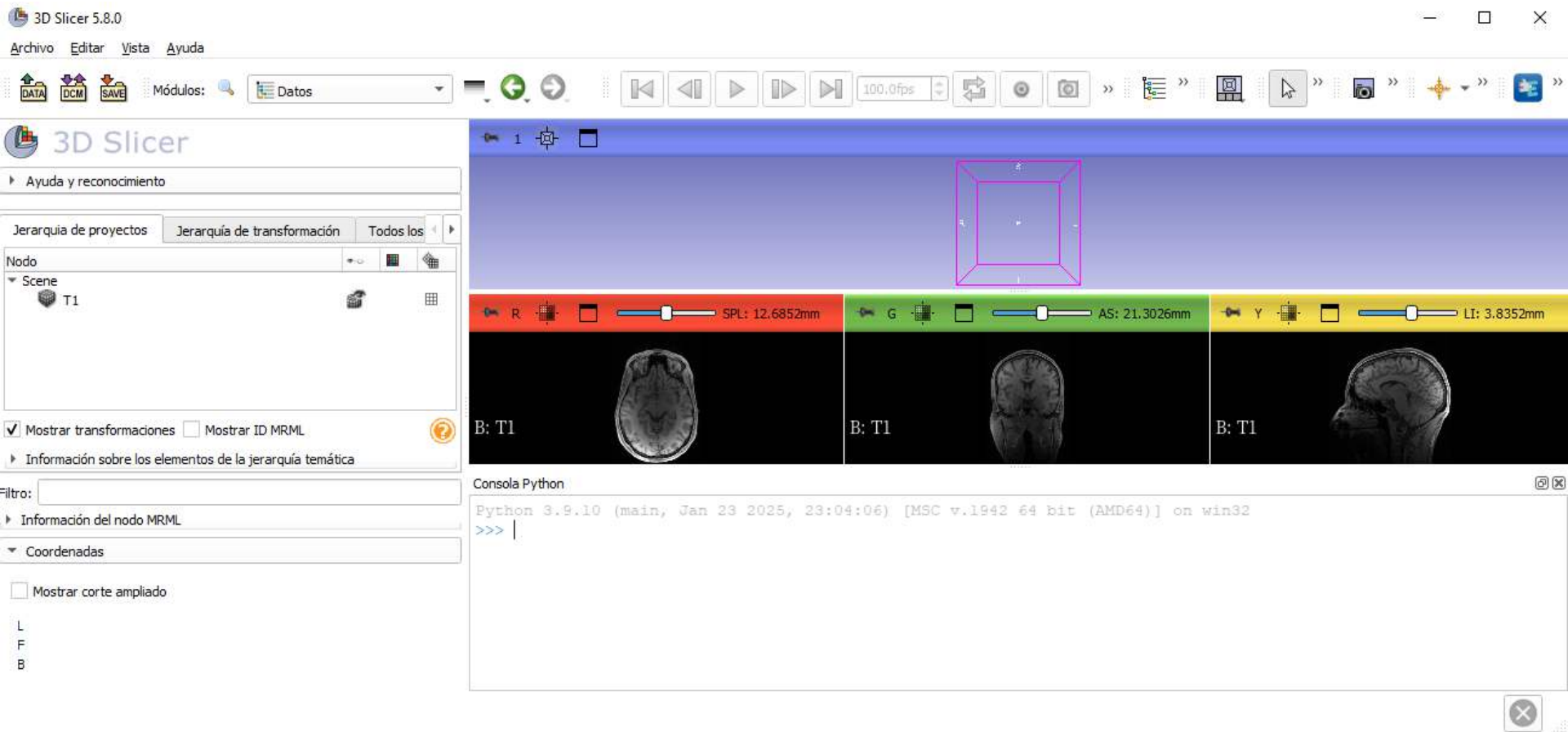
Conjunto de datos



Arrastre y suelte el archivo T1.nrrd

Haga clic en Aceptar para cargar el archivo en Slicer

Tutorial dataset



Panorama general

- Slicer es un software gratuito y de código abierto
- Hay miles de imágenes médicas sofisticadas disponibles en Internet que puede visualizar y analizar con 3D Slicer.

Modelo de datos de Slicer



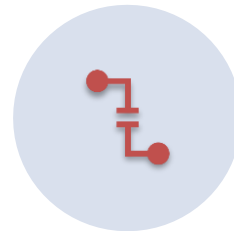
El modelo de datos de Slicer se basa en la estructura de datos de escena de Slicer.



Una escena Slicer es una colección de imágenes, anotaciones, modelos 3D, transformaciones espaciales, fiduciales y cámaras.



El lenguaje MRML (Medical Reality Markup Language) es un lenguaje basado en XML que se utiliza para serializar el contenido de la escena Slicer en disco (scene.mrml).



Cada elemento de una escena se denomina nodo MRML

Nodos MRML de Slicer: Tipos Básicos



Nodo de datos:
Almacena los datos
en bruto.

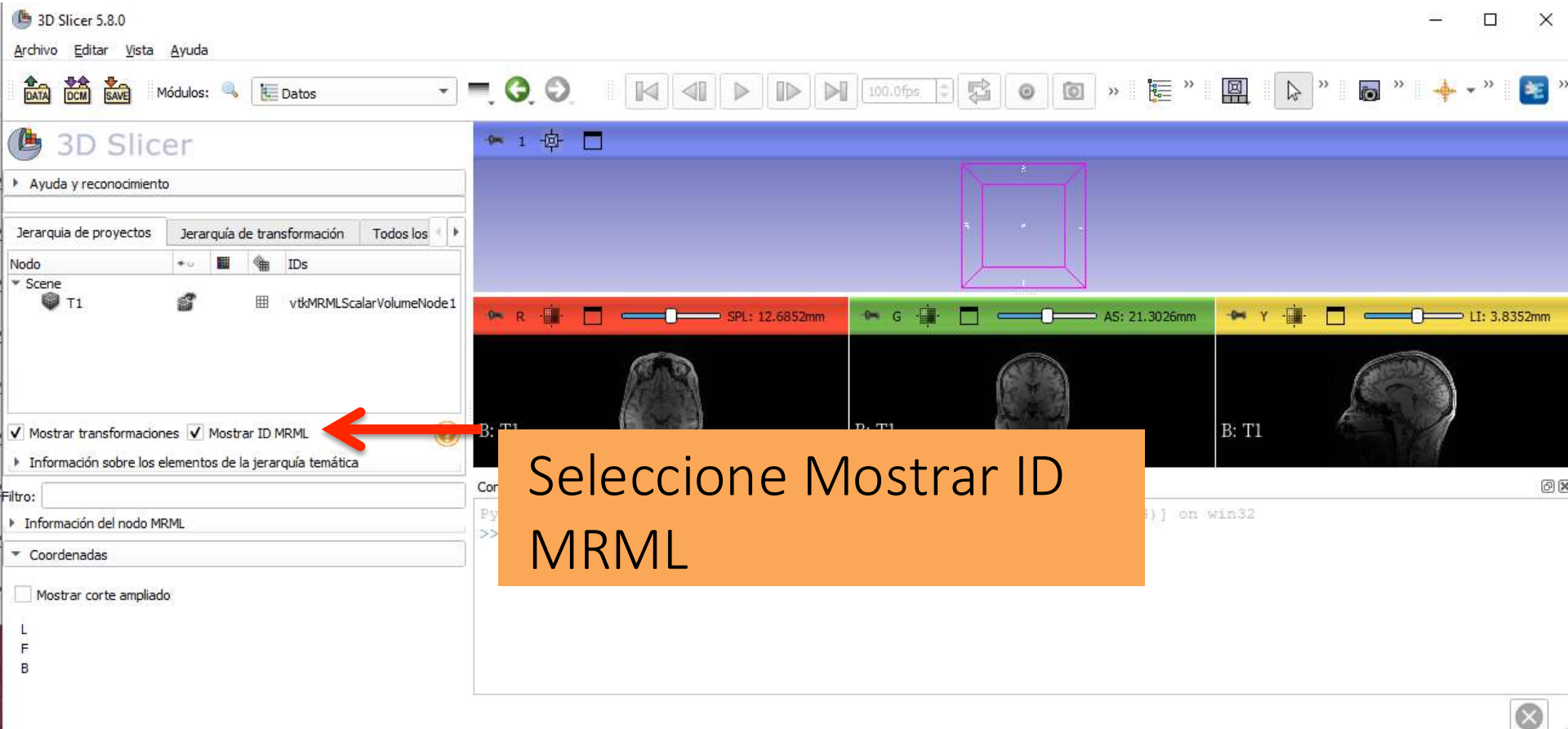


Nodo de
visualización:
Describe cómo
deben visualizarse los
datos.

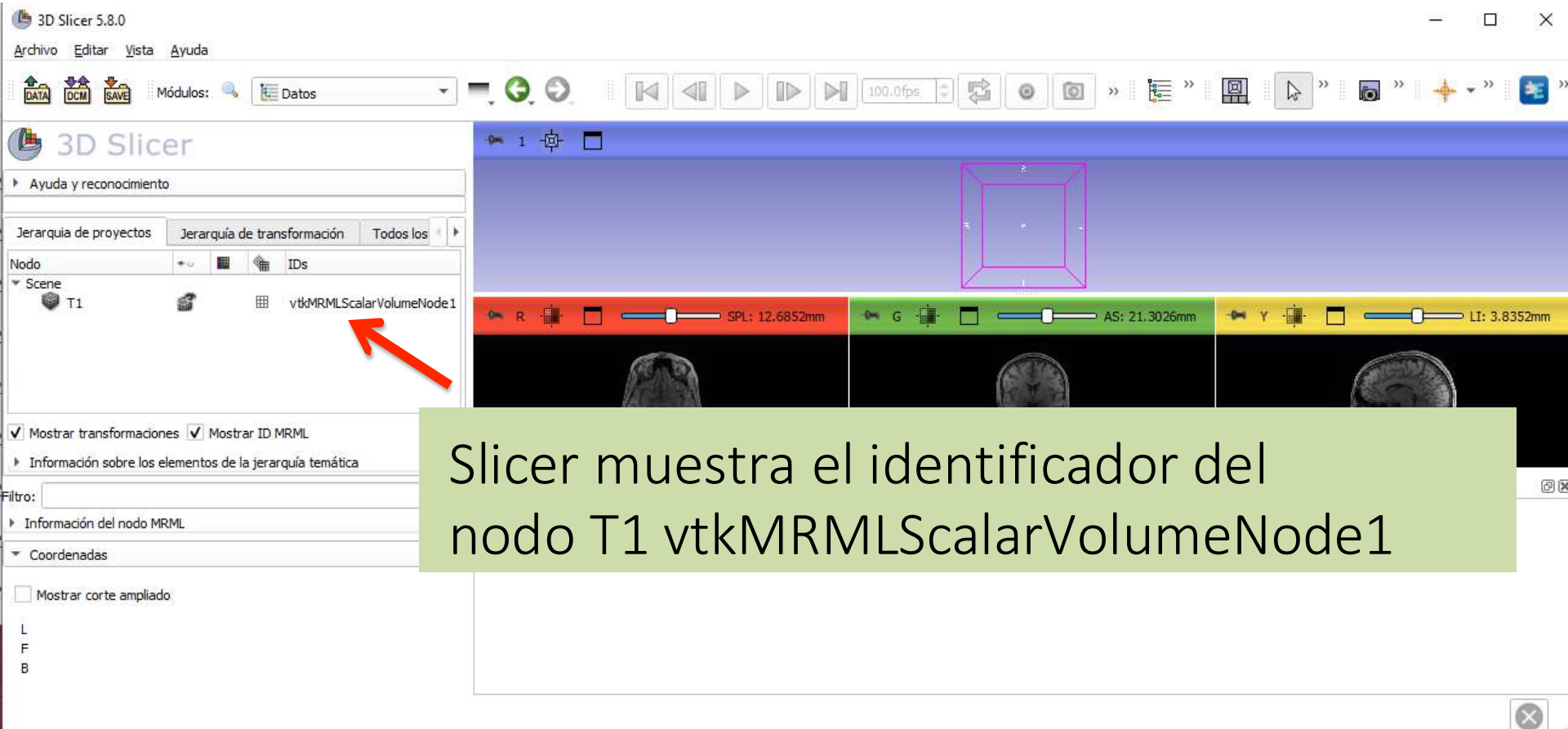


Nodo de
almacenamiento:
Describe cómo deben
almacenarse los
datos en el disco.

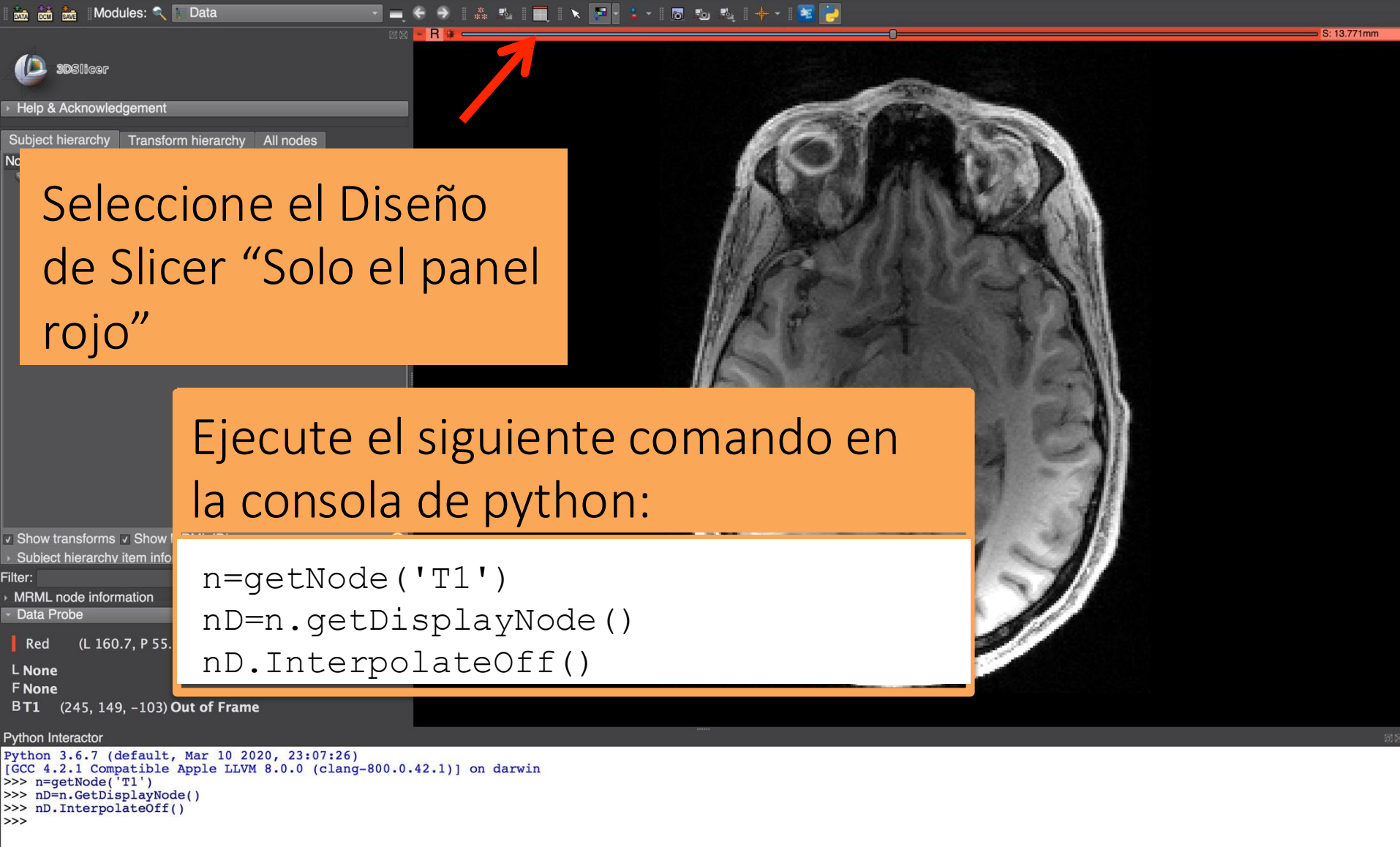
Conjunto de datos del tutorial



Slicer Data Model



Acceso a los nodos MRML desde la consola Python



The image shows a screenshot of the Slicer software interface. The main window displays a brain MRI scan. A red arrow points to the 'R' icon in the top toolbar. An orange box contains the text: 'Seleccione el Diseño de Slicer "Solo el panel rojo"'. Another orange box contains the text: 'Ejecute el siguiente comando en la consola de python:'. Below this, a white box contains the Python code: `n=getNode('T1')`, `nD=n.getDisplayNode()`, and `nD.InterpolateOff()`. At the bottom, a Python Interactor window shows the execution of these commands.

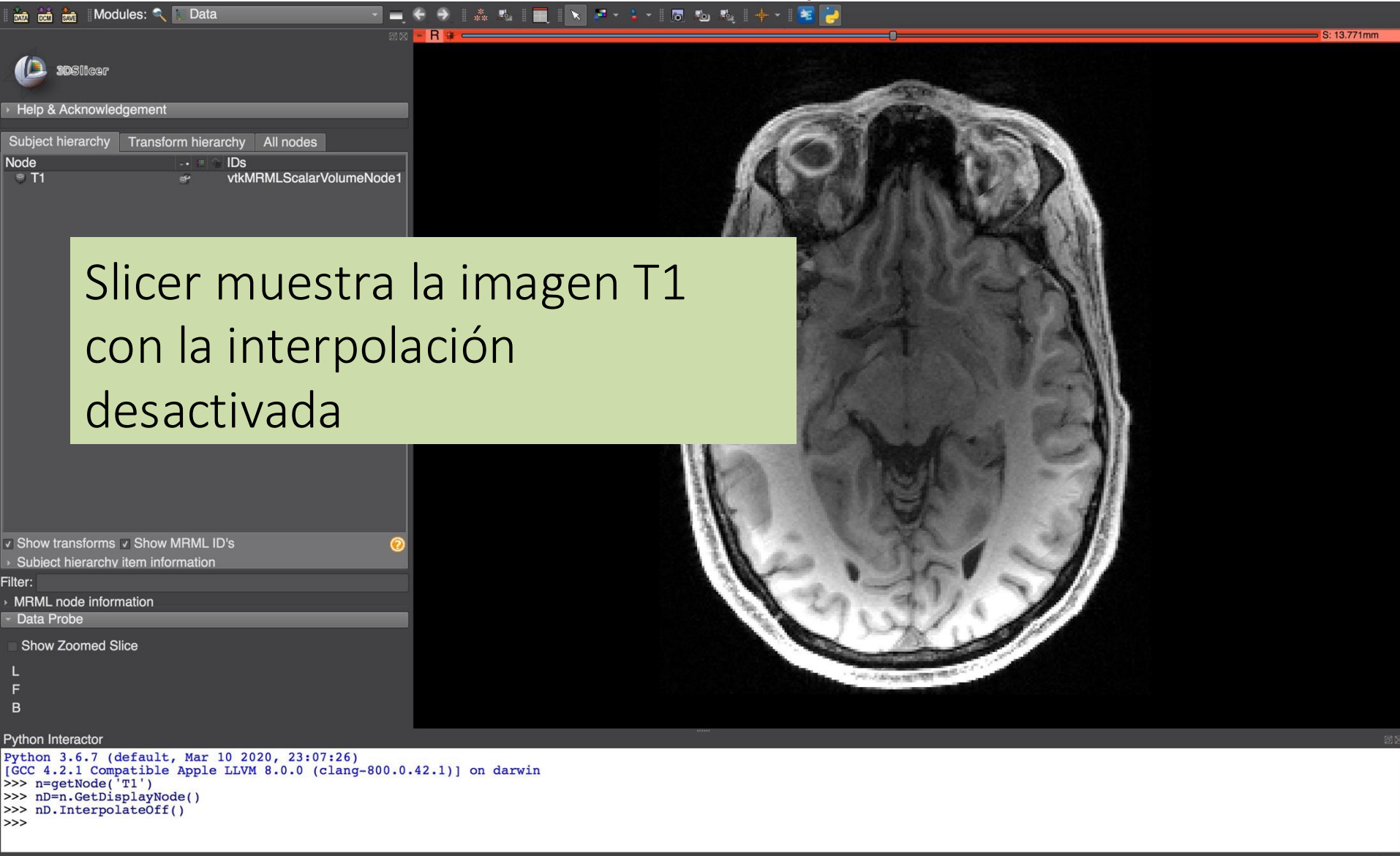
Seleccione el Diseño de Slicer "Solo el panel rojo"

Ejecute el siguiente comando en la consola de python:

```
n=getNode('T1')
nD=n.getDisplayNode()
nD.InterpolateOff()
```

```
Python Interactor
Python 3.6.7 (default, Mar 10 2020, 23:07:26)
[GCC 4.2.1 Compatible Apple LLVM 8.0.0 (clang-800.0.42.1)] on darwin
>>> n=getNode('T1')
>>> nD=n.GetDisplayNode()
>>> nD.InterpolateOff()
>>>
```

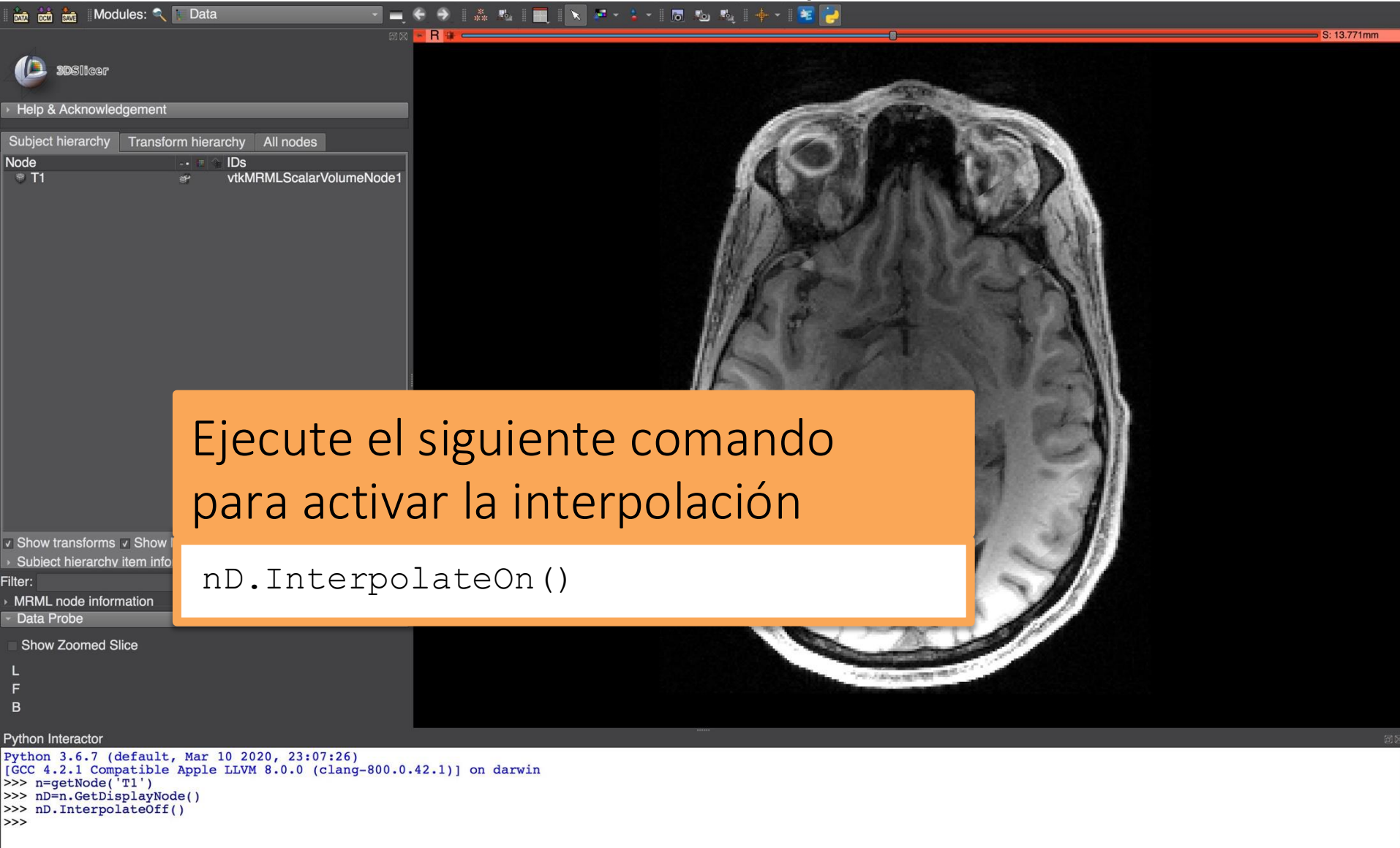
Acceso a los nodos MRML desde la consola Python



Slicer muestra la imagen T1 con la interpolación desactivada

```
Python Interactor
Python 3.6.7 (default, Mar 10 2020, 23:07:26)
[GCC 4.2.1 Compatible Apple LLVM 8.0.0 (clang-800.0.42.1)] on darwin
>>> n=getNode('T1')
>>> nD=n.GetDisplayNode()
>>> nD.InterpolateOff()
>>>
```

Acceso a los nodos MRML desde la consola Python

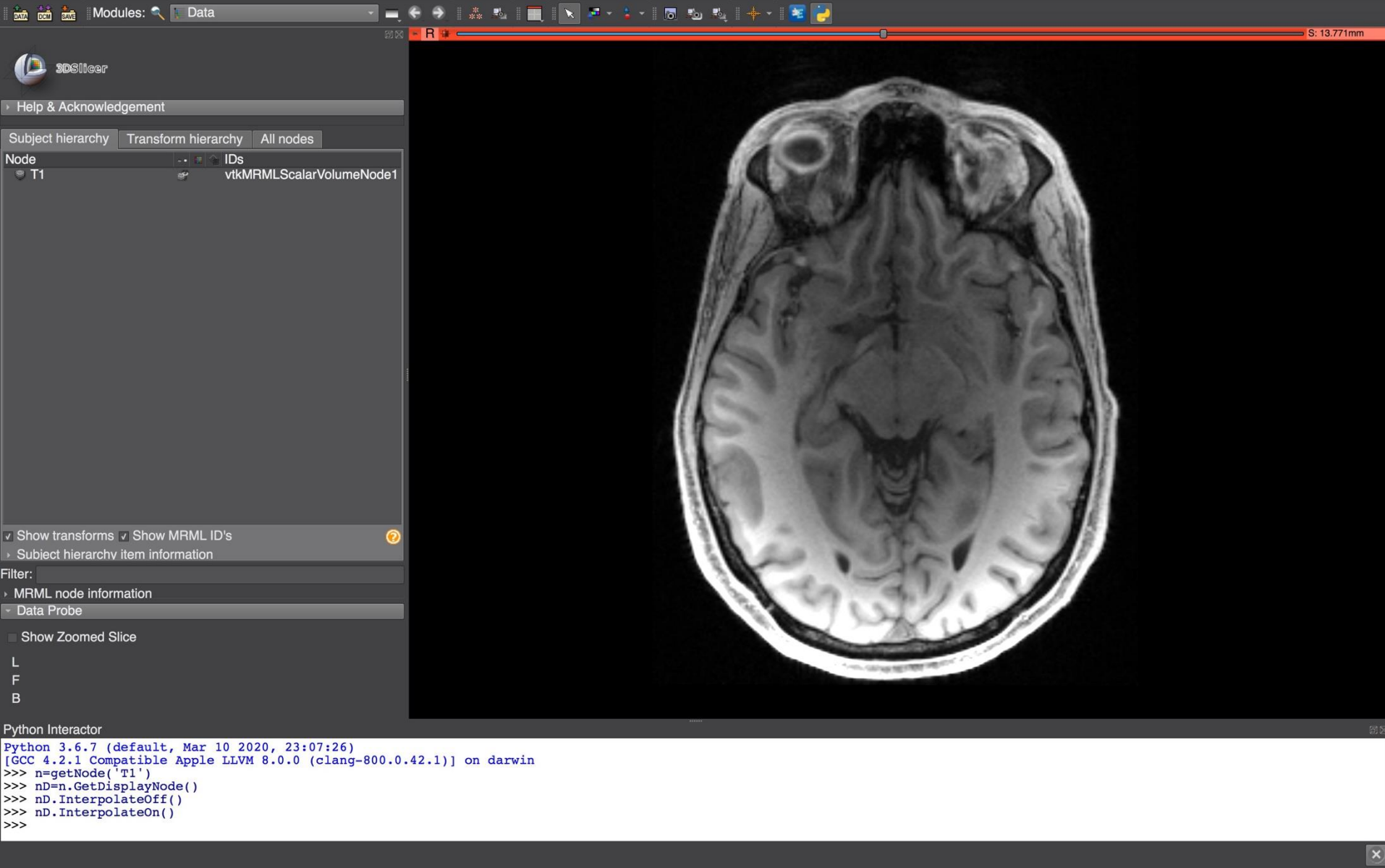


Ejecute el siguiente comando para activar la interpolación

```
nD.InterpolateOn()
```

```
Python 3.6.7 (default, Mar 10 2020, 23:07:26)
[GCC 4.2.1 Compatible Apple LLVM 8.0.0 (clang-800.0.42.1)] on darwin
>>> n=getNode('T1')
>>> nD=n.GetDisplayNode()
>>> nD.InterpolateOff()
>>>
```

Acceso a los nodos MRML desde la consola Python



The screenshot displays the 3D Slicer software interface. The main window shows an axial MRI slice of a brain. On the left, the 'Subject hierarchy' panel is visible, showing a tree structure with a node named 'T1' of type 'vtkMRMLScalarVolumeNode1'. Below this, there are checkboxes for 'Show transforms', 'Show MRML ID's', and 'Show Zoomed Slice'. At the bottom, a 'Python Interactor' window shows the following code:

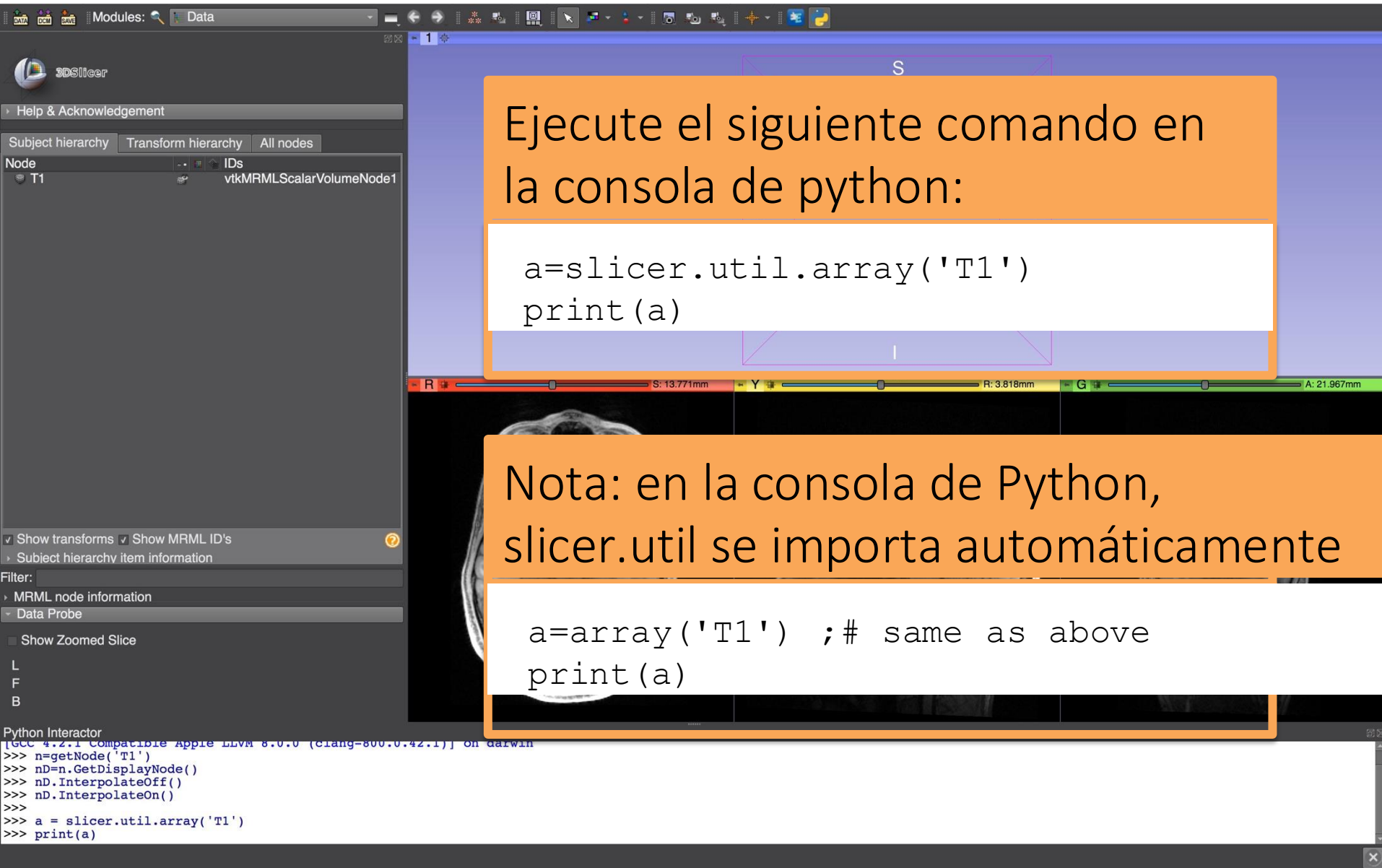
```
Python 3.6.7 (default, Mar 10 2020, 23:07:26)
[GCC 4.2.1 Compatible Apple LLVM 8.0.0 (clang-800.0.42.1)] on darwin
>>> n=getNode('T1')
>>> nd=n.GetDisplayNode()
>>> nd.InterpolateOff()
>>> nd.InterpolateOn()
>>>
```

Acceso a los vóxeles de un volumen

- El paquete `slicer.util` da acceso a los volúmenes como matrices multidimensionales NumPy
- Los volúmenes pueden modificarse utilizando métodos estándar de NumPy



Acceso a los vóxeles de un volumen



The image shows a screenshot of the Slicer software interface. The main window displays a 3D volume of a brain slice. The left sidebar contains a 'Subject hierarchy' panel with a tree view showing 'Node T1' and 'vtkMRMLScalarVolumeNode1'. Below this is a 'Python Interactor' window showing a terminal with the following code:

```
[GCC 4.2.1 Compatible Apple LLVM 8.0.0 (clang-800.0.42.1)] on darwin
>>> n=getNode('T1')
>>> nD=n.GetDisplayNode()
>>> nD.InterpolateOff()
>>> nD.InterpolateOn()
>>>
>>> a = slicer.util.array('T1')
>>> print(a)
```

Two orange callout boxes are overlaid on the interface. The top box contains the text: 'Ejecute el siguiente comando en la consola de python:' followed by a code block with the same Python code as shown in the console. The bottom box contains the text: 'Nota: en la consola de Python, slicer.util se importa automáticamente' followed by a code block with a simplified version of the Python code: 'a=array('T1') ;# same as above' and 'print(a)'. The background shows the Slicer interface with a 3D volume and a coordinate system (R, Y, G) at the bottom.

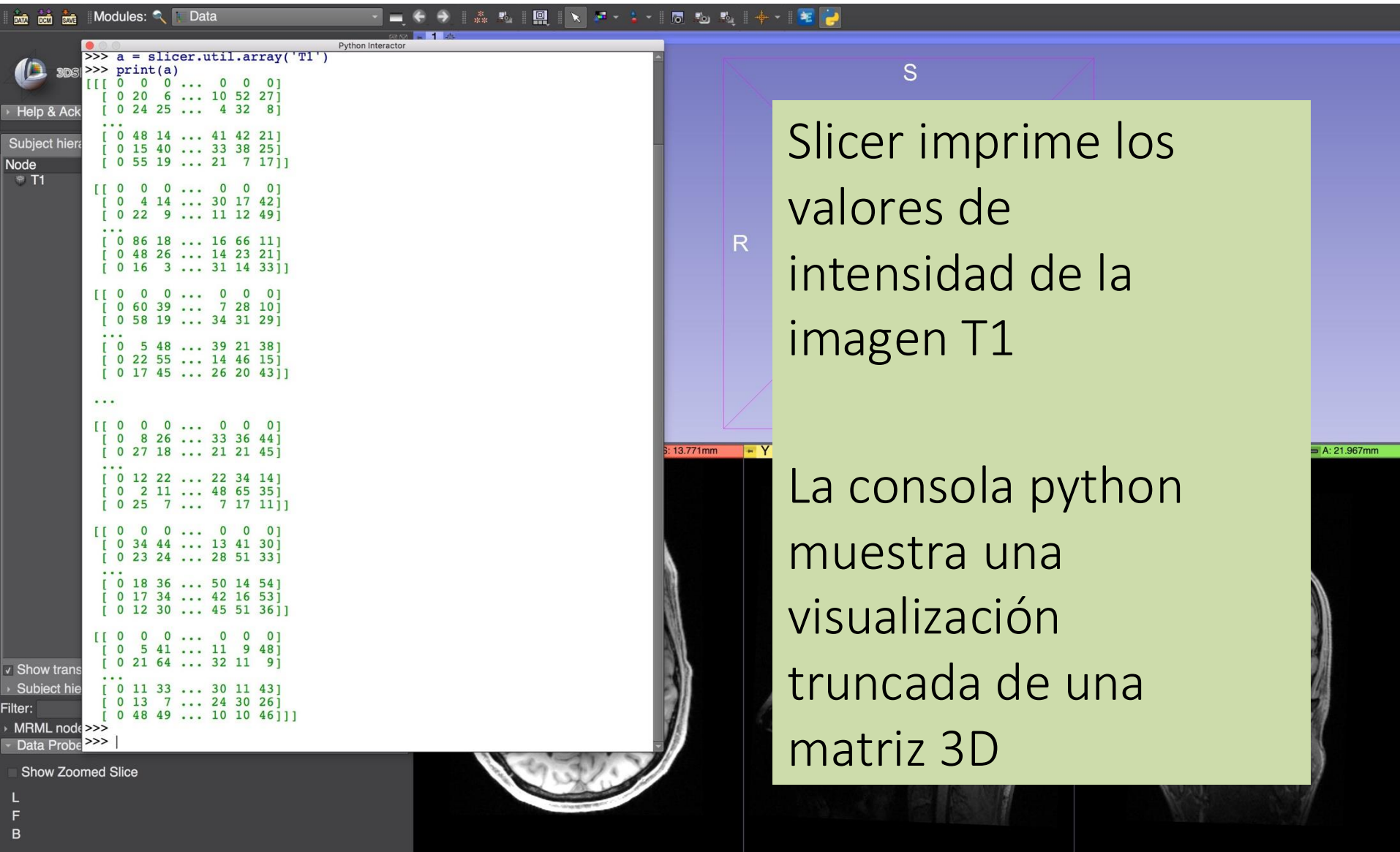
Ejecute el siguiente comando en la consola de python:

```
a=slicer.util.array('T1')
print(a)
```

Nota: en la consola de Python, slicer.util se importa automáticamente

```
a=array('T1') ;# same as above
print(a)
```

Acceso a los vóxeles de un volumen



The image shows a screenshot of the Slicer software interface. On the left, a Python console window displays the output of a script that prints a 3D matrix of intensity values for a T1-weighted MRI slice. The matrix is a 10x10x10 grid of integers. On the right, a 3D visualization of a brain slice is shown with a purple bounding box indicating the slice's extent. The axes are labeled S (Superior-Inferior), R (Right-Left), and Y (Anterior-Posterior).

```
>>> a = slicer.util.array('T1')
>>> print(a)
[[[ 0  0  0 ...  0  0  0]
 [ 0 20  6 ... 10 52 27]
 [ 0 24 25 ...  4 32  8]
 ...
 [ 0 48 14 ... 41 42 21]
 [ 0 15 40 ... 33 38 25]
 [ 0 55 19 ... 21  7 17]]

[[[ 0  0  0 ...  0  0  0]
 [ 0  4 14 ... 30 17 42]
 [ 0 22  9 ... 11 12 49]
 ...
 [ 0 86 18 ... 16 66 11]
 [ 0 48 26 ... 14 23 21]
 [ 0 16  3 ... 31 14 33]]

[[[ 0  0  0 ...  0  0  0]
 [ 0 60 39 ...  7 28 10]
 [ 0 58 19 ... 34 31 29]
 ...
 [ 0  5 48 ... 39 21 38]
 [ 0 22 55 ... 14 46 15]
 [ 0 17 45 ... 26 20 43]]

...

[[[ 0  0  0 ...  0  0  0]
 [ 0  8 26 ... 33 36 44]
 [ 0 27 18 ... 21 21 45]
 ...
 [ 0 12 22 ... 22 34 14]
 [ 0  2 11 ... 48 65 35]
 [ 0 25  7 ...  7 17 11]]

[[[ 0  0  0 ...  0  0  0]
 [ 0 34 44 ... 13 41 30]
 [ 0 23 24 ... 28 51 33]
 ...
 [ 0 18 36 ... 50 14 54]
 [ 0 17 34 ... 42 16 53]
 [ 0 12 30 ... 45 51 36]]

[[[ 0  0  0 ...  0  0  0]
 [ 0  5 41 ... 11  9 48]
 [ 0 21 64 ... 32 11  9]
 ...
 [ 0 11 33 ... 30 11 43]
 [ 0 13  7 ... 24 30 26]
 [ 0 48 49 ... 10 10 46]]]
>>>
```

Slicer imprime los valores de intensidad de la imagen T1

La consola python muestra una visualización truncada de una matriz 3D

Acceso a los vóxeles de un volumen

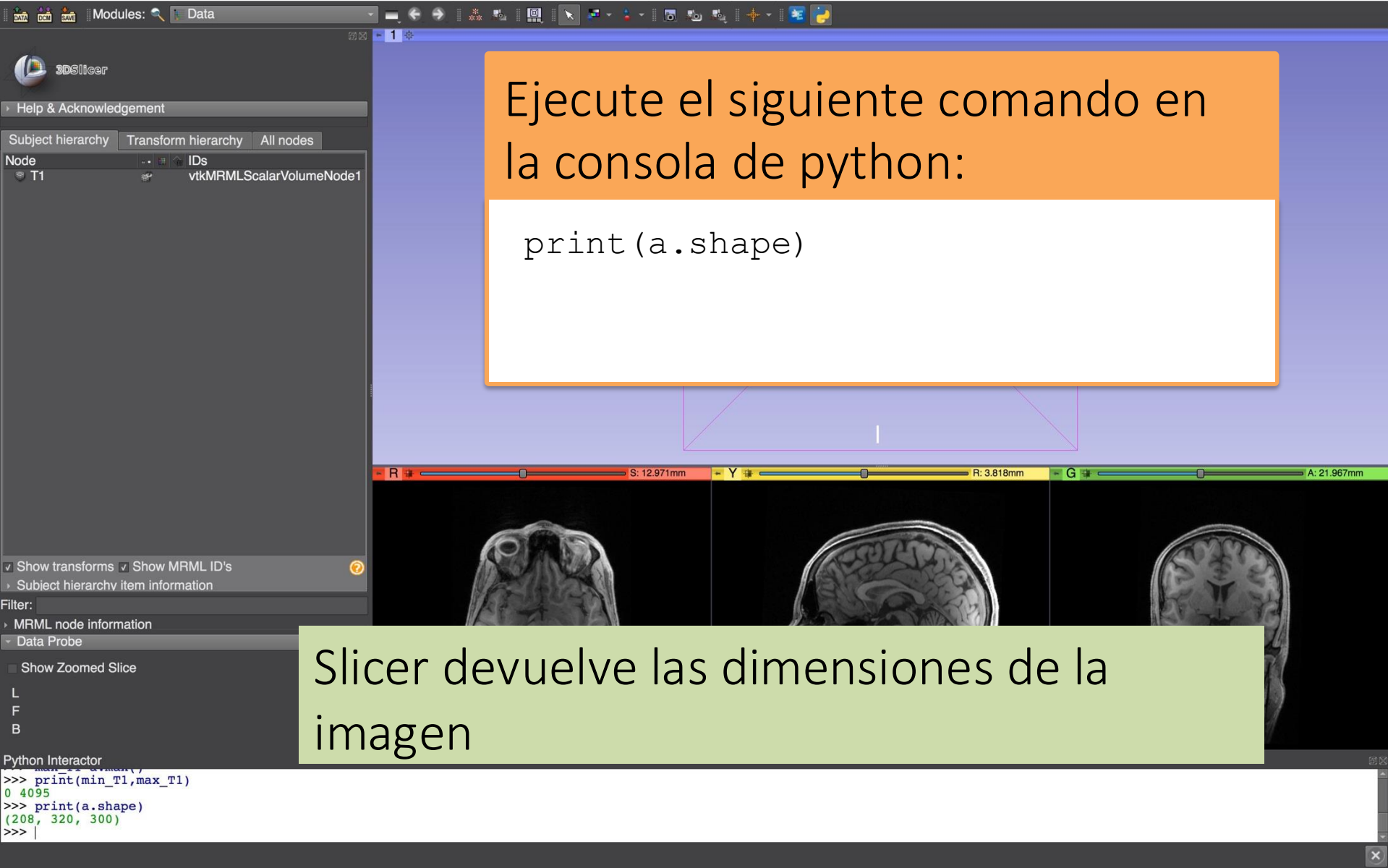


The screenshot displays the 3D Slicer software interface. The main window shows a T1 MRI volume with a purple bounding box. The bounding box is labeled with 'S' at the top, 'I' at the bottom, 'R' on the left, and 'L' on the right. Below the main window, there are three orthogonal views: an axial view on the left, a sagittal view in the middle, and a coronal view on the right. The interface includes a sidebar on the left with a subject hierarchy, a data probe, and a Python Interactor. The Python Interactor shows the following code and output:

```
>>> min_T1=a.min()
>>> max_T1=a.max()
>>> print(min_T1,max_T1)
0 4095
>>>
```

A green text box at the bottom of the image contains the text: "Slicer devuelve los valores mínimo y máximo de la imagen T1".

Modificación de vóxeles en un volumen



The screenshot shows the 3D Slicer software interface. On the left, the 'Subject hierarchy' panel shows a 'T1' node of type 'vtkMRMLScalarVolumeNode1'. The main window displays three orthogonal views of a brain MRI volume. Below the views, a status bar shows the dimensions: S: 12.971mm, R: 3.818mm, and A: 21.967mm. A Python Interactor at the bottom shows the following code and output:

```
>>> print(min_T1,max_T1)
0 4095
>>> print(a.shape)
(208, 320, 300)
>>>
```

Ejecute el siguiente comando en la consola de python:

```
print(a.shape)
```

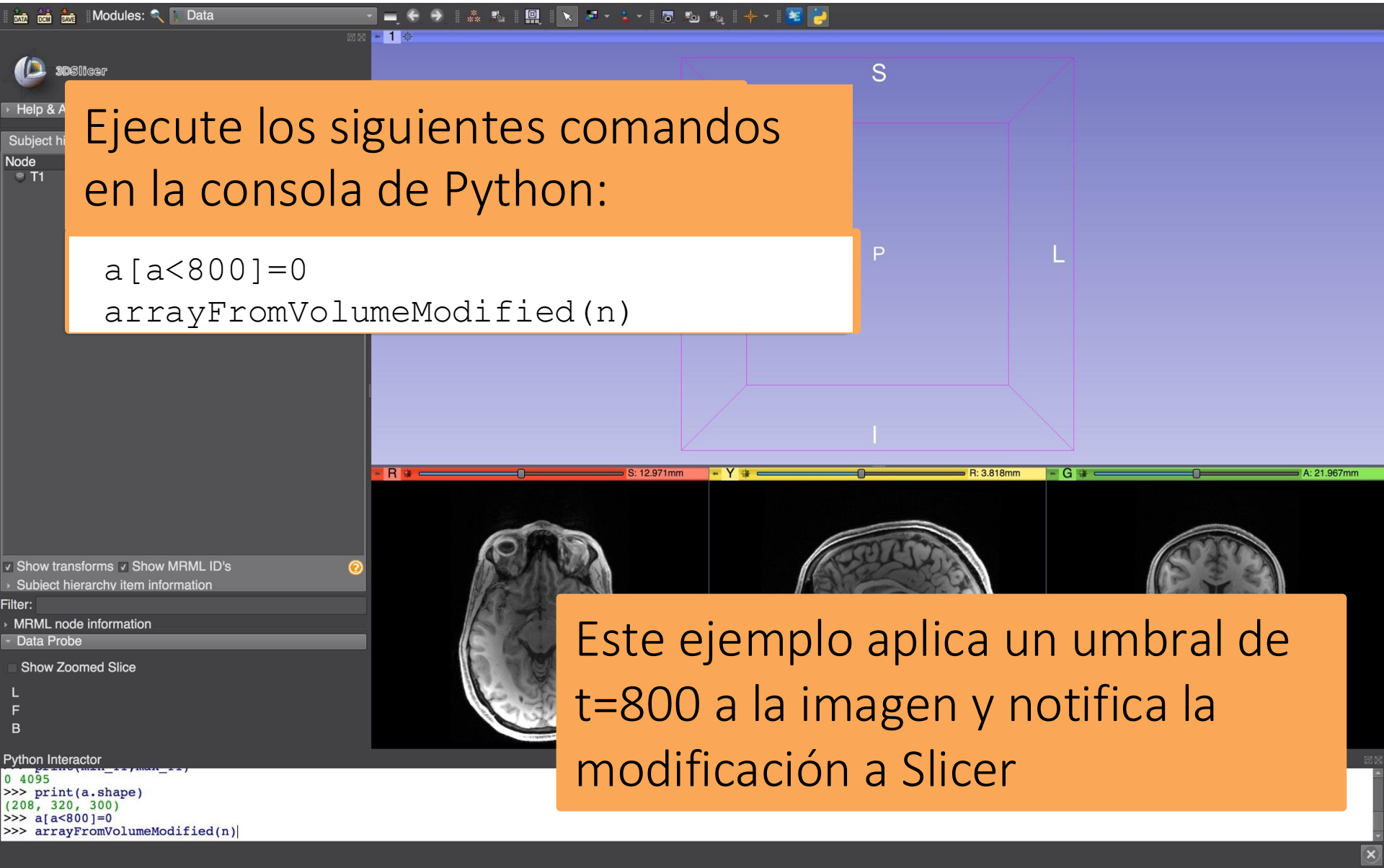
Slicer devuelve las dimensiones de la imagen

Modificación de vóxeles en un volumen

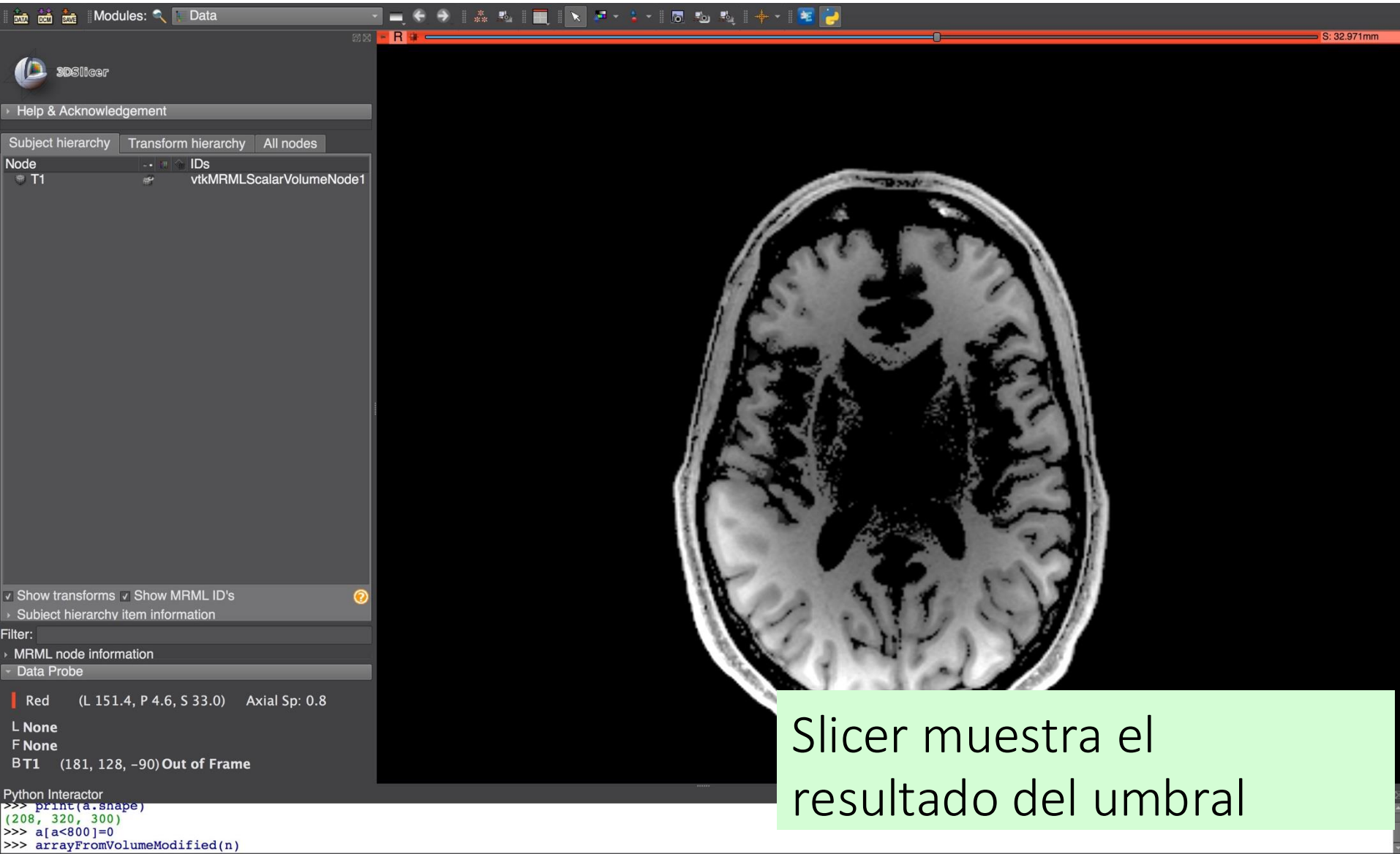
Ejecute los siguientes comandos en la consola de Python:

```
a[a<800]=0  
arrayFromVolumeModified(n)
```

Este ejemplo aplica un umbral de $t=800$ a la imagen y notifica la modificación a Slicer



Modificación de vóxeles en un volumen



The image shows the Slicer software interface. On the left, the 'Subject hierarchy' panel shows a node 'T1' of type 'vtkMRMLScalarVolumeNode1'. Below it, the 'Data Probe' section shows the following information:

- Red (L 151.4, P 4.6, S 33.0) Axial Sp: 0.8
- L None
- F None
- BT1 (181, 128, -90) Out of Frame

At the bottom, the Python Interactor shows the following code and output:

```
>>> print(a.shape)
(208, 320, 300)
>>> a[a<800]=0
>>> arrayFromVolumeModified(n)
```

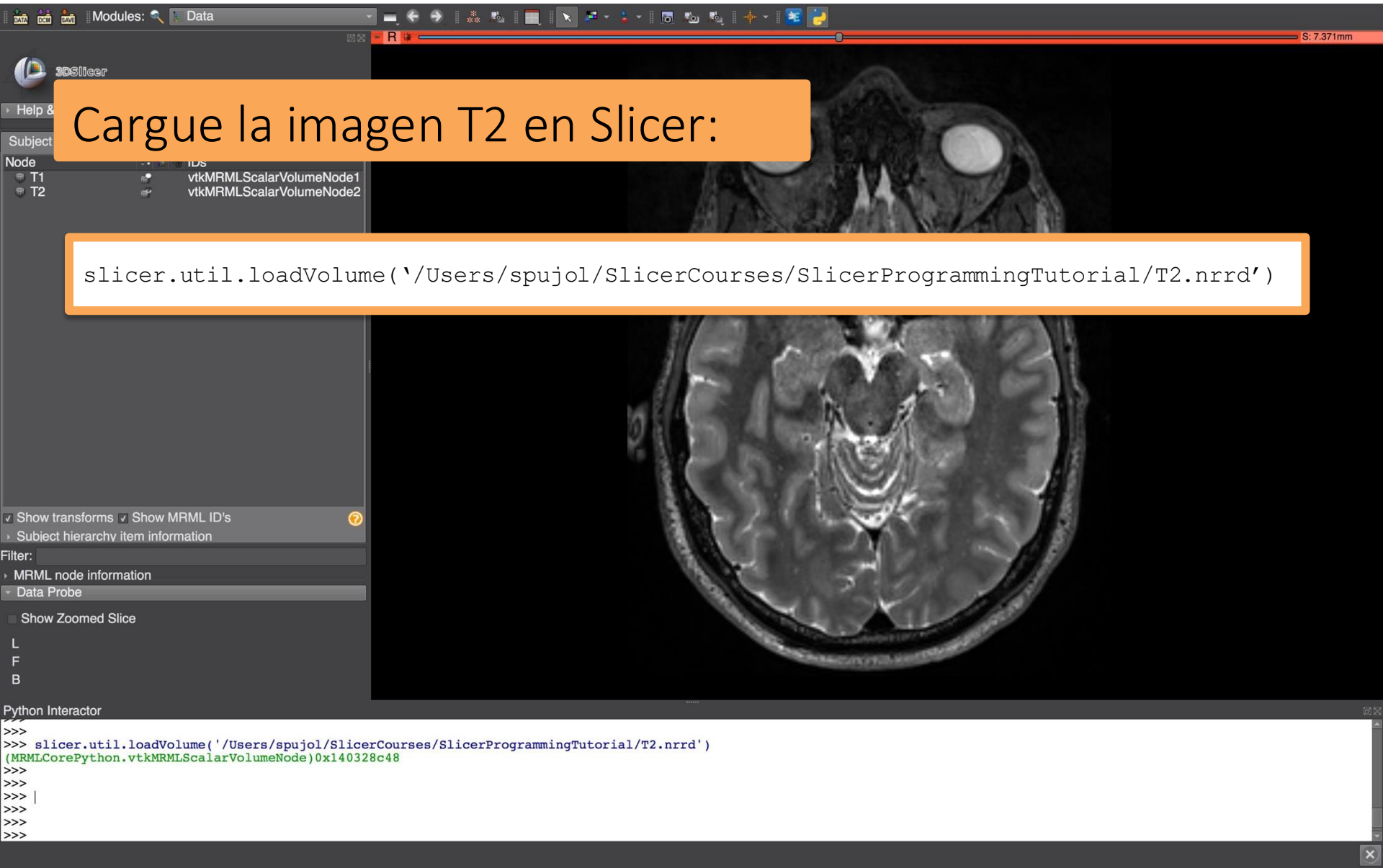
In the center, an axial MRI slice of a brain is displayed. The slice shows a threshold filter applied, resulting in a high-contrast, binary-like appearance of the brain tissue.

Slicer muestra el resultado del umbral

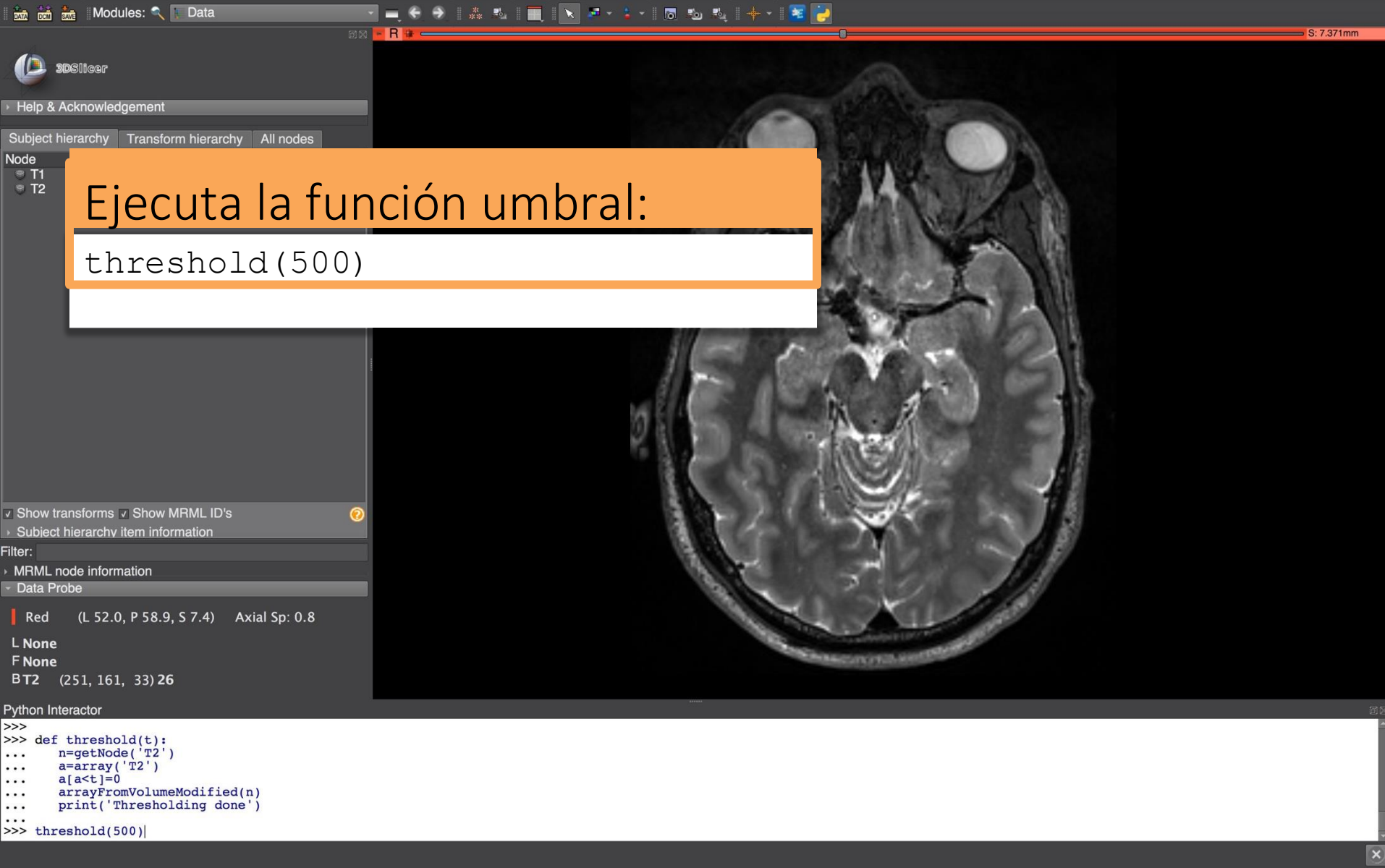
Cargar el volumen T2

Cargue la imagen T2 en Slicer:

```
slicer.util.loadVolume('/Users/spujol/SlicerCourses/SlicerProgrammingTutorial/T2.nrrd')
```



Función Python: umbral

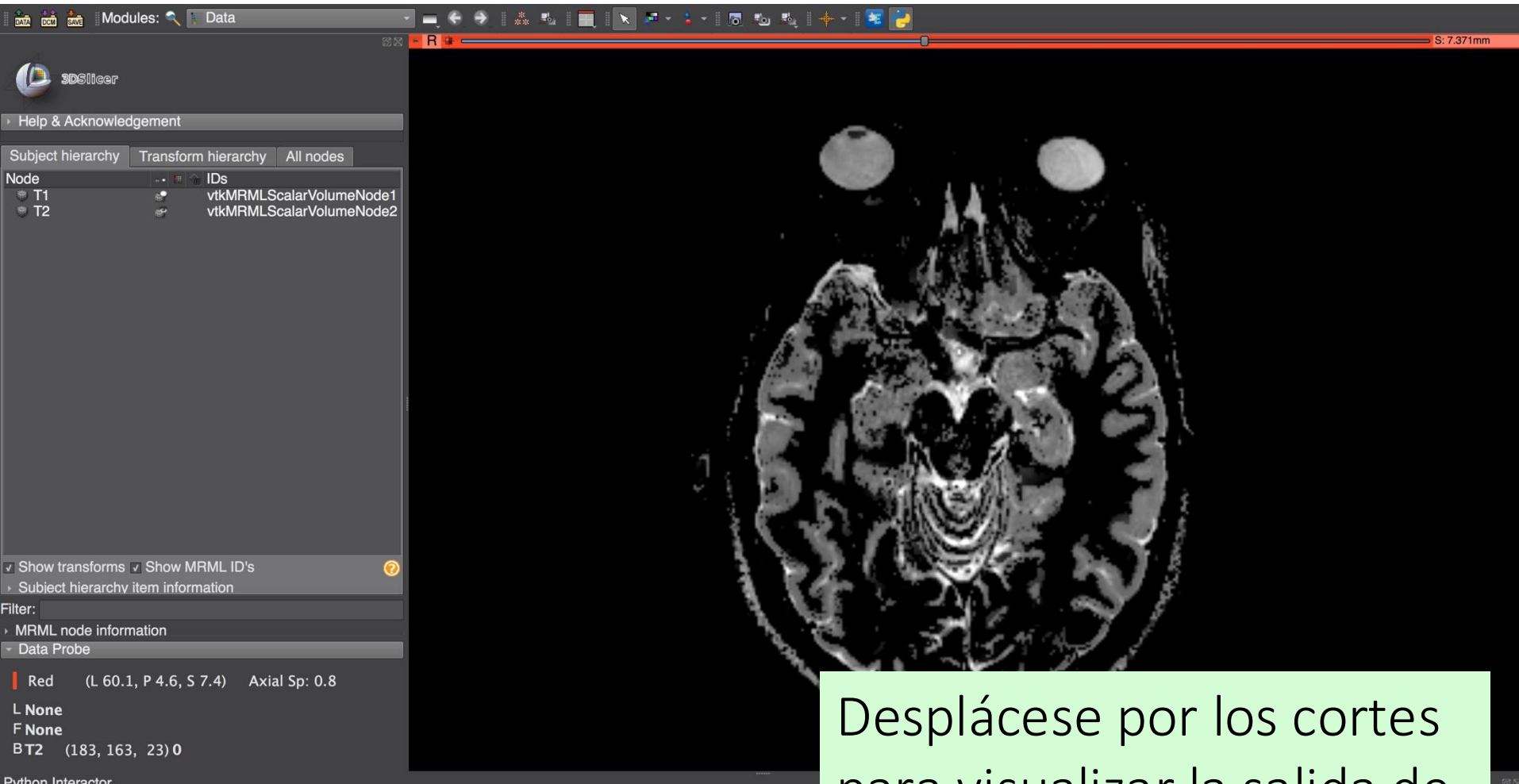


The image shows the 3D Slicer software interface. The main window displays an axial MRI brain scan. On the left, the 'Node' panel shows 'T1' and 'T2' nodes. Below it, the 'Data Probe' section shows 'Red (L 52.0, P 58.9, S 7.4) Axial Sp: 0.8'. At the bottom, the 'Python Interactor' window contains the following code:

```
>>>
>>> def threshold(t):
...     n=getNode('T2')
...     a=array('T2')
...     a[a<t]=0
...     arrayFromVolumeModified(n)
...     print('Thresholding done')
...
>>> threshold(500)
```

An orange callout box is overlaid on the interface, containing the text: "Ejecuta la función umbral: threshold(500)".

Función Python: umbral



The screenshot displays the 3D Slicer software interface. The main window shows an axial MRI slice of a brain with a thresholding filter applied, resulting in a binary (black and white) image. The left sidebar contains the 'Subject hierarchy' and 'Transform hierarchy' panels. The 'Subject hierarchy' panel shows two nodes: 'T1' and 'T2'. The 'Data Probe' panel shows the current slice parameters: 'Red (L 60.1, P 4.6, S 7.4) Axial Sp: 0.8'. The 'Python Interactor' window at the bottom left shows the following code and output:

```
... n=getNode('T2')
... a=array('T2')
... a[a<t]=0
... arrayFromVolumeModified(n)
... print('Thresholding done')
...
>>> threshold(500)
Thresholding done
>>>
```

Desplácese por los cortes para visualizar la salida de la función umbral

Panorama General

- Slicer facilita el acceso para analizar y modificar tipos de datos complejos
- Slicer es compatible con una amplia gama de paquetes informáticos científicos de Python.
- Slicer es un entorno de investigación para realizar experimentos con imágenes médicas

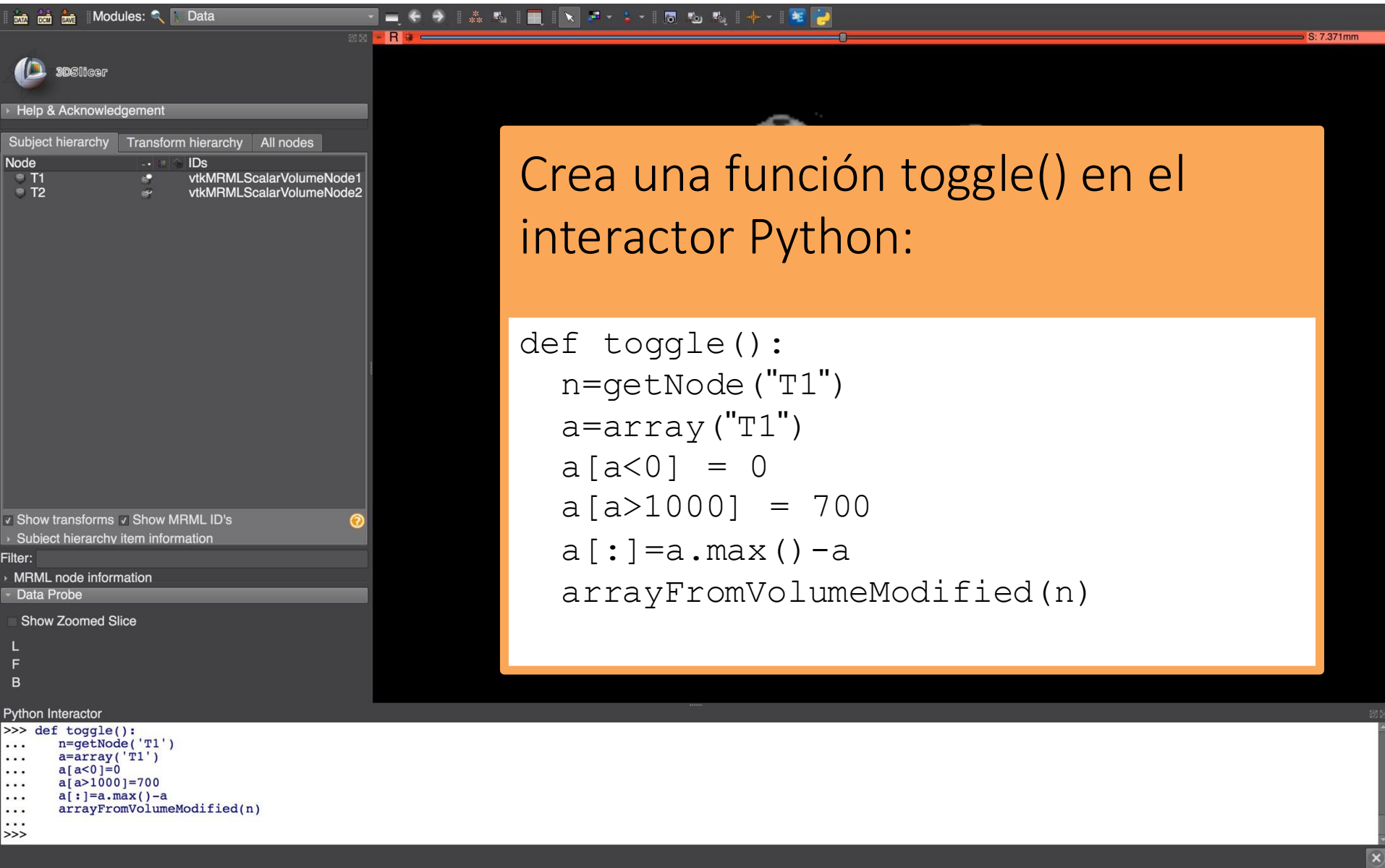
Parte 3

Familiarizarse con Qt en Slicer

Qt & PythonQt

- Qt es la herramienta principal de Slicer para crear widgets, cuadros de diálogo, entradas de texto, etc.
- PythonQt expone la mayoría de las funcionalidades de Qt y es accesible a través del interactuador Python en Slicer.
- Las interfaces de usuario se pueden crear sobre la marcha para una rápida creación de prototipos y depuración.

Función Python: toggle

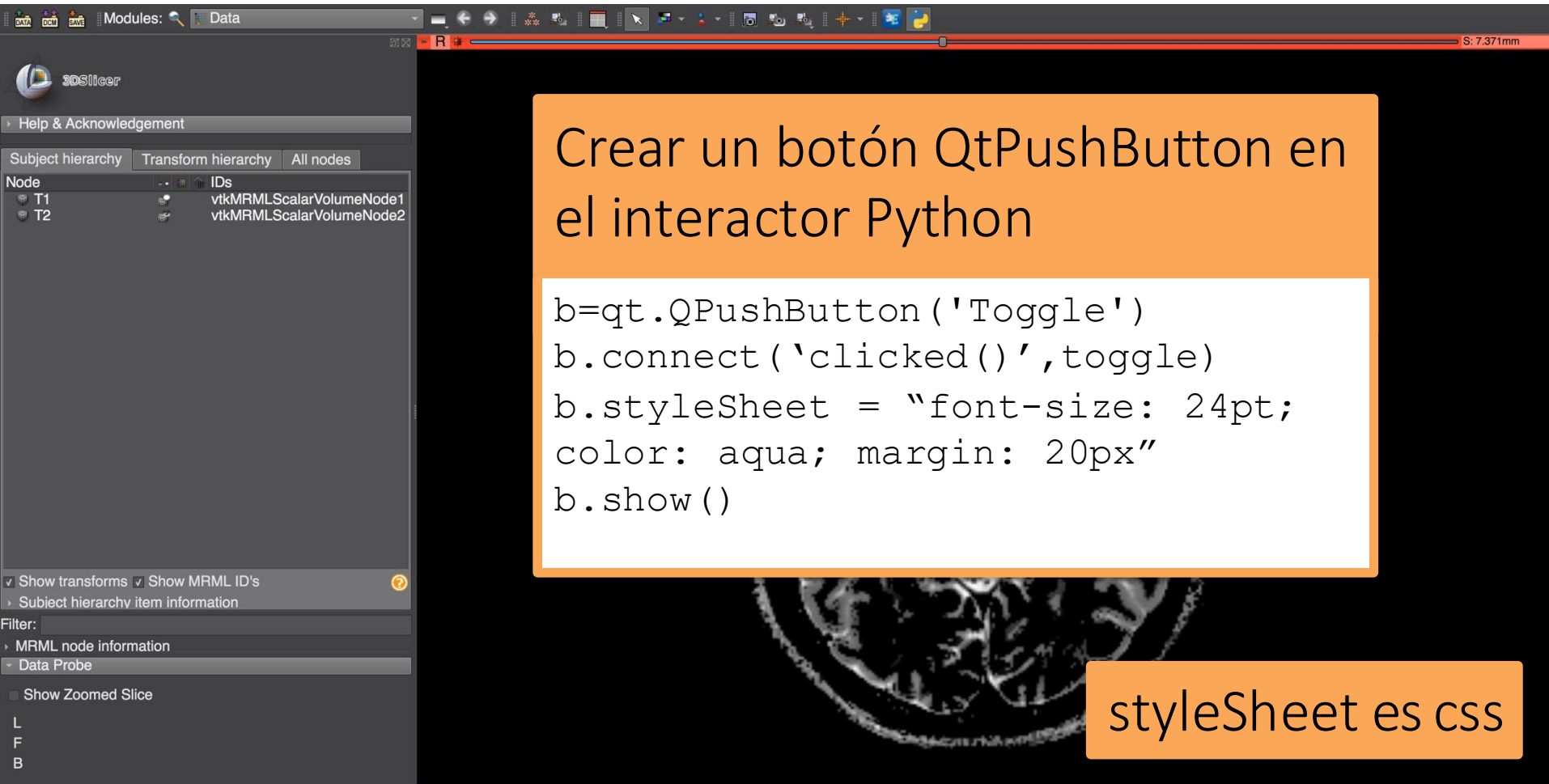


The image shows a screenshot of the 3D Slicer software interface. The top toolbar includes icons for Data, Modules, and various tool functions. The left sidebar contains a 'Subject hierarchy' panel with a tree view showing nodes T1 and T2, and their corresponding MRML IDs. Below this are sections for 'Filter:' and 'MRML node information'. The bottom panel is the 'Python Interactor', which displays the following Python code:

```
>>> def toggle():
...     n=getNode('T1')
...     a=array('T1')
...     a[a<0]=0
...     a[a>1000]=700
...     a[:]=a.max()-a
...     arrayFromVolumeModified(n)
...
>>>
```

The code defines a function named `toggle()` that performs several operations: it retrieves a node named 'T1', creates an array from it, sets values less than 0 to 0, sets values greater than 1000 to 700, and then applies the modified array back to the volume. The function concludes with `arrayFromVolumeModified(n)`.

Creación de un botón Qt



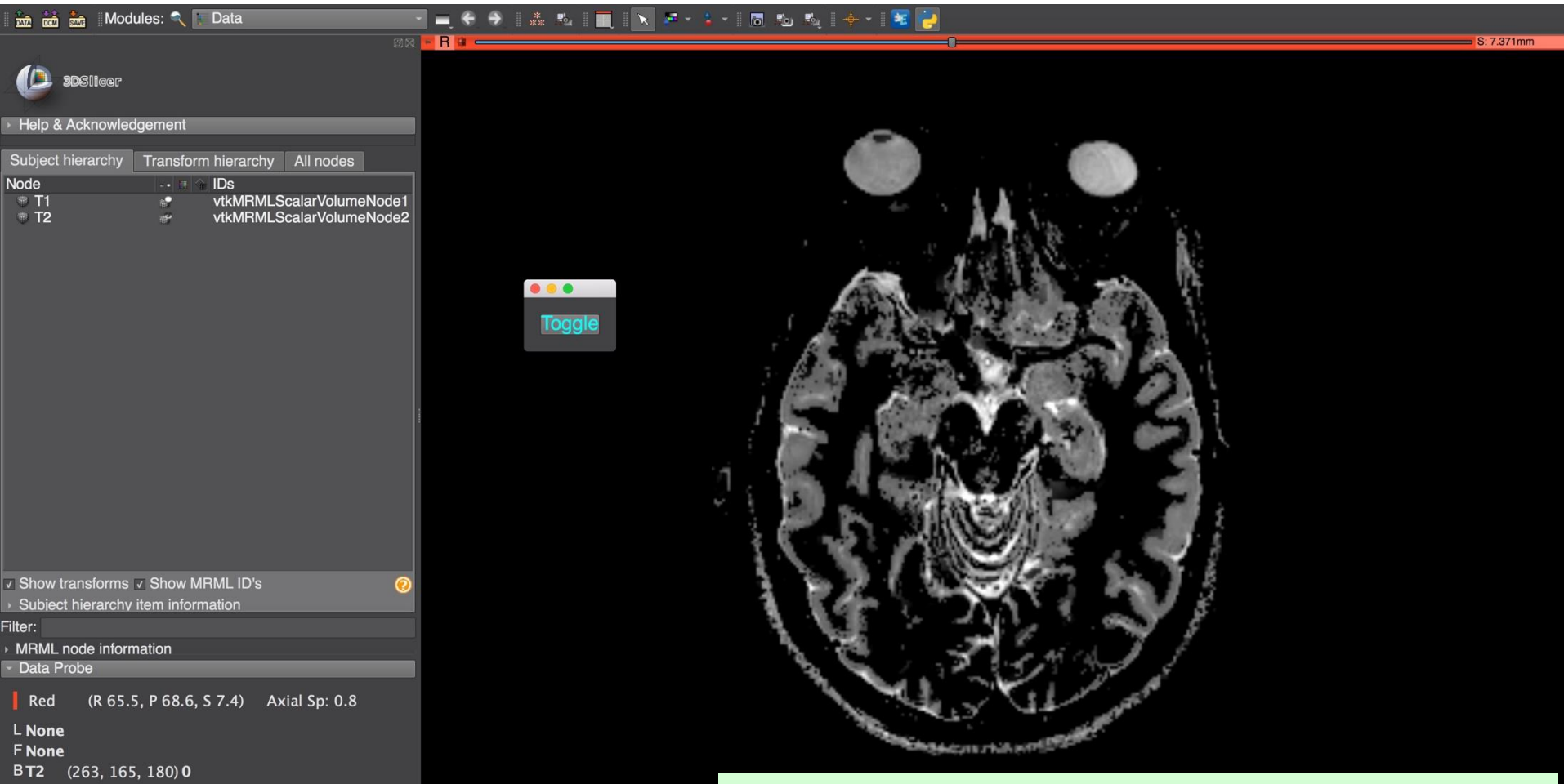
Crear un botón QPushButton en el interactor Python

```
b=qt.QPushButton('Toggle')  
b.connect('clicked()',toggle)  
b.styleSheet = "font-size: 24pt;  
color: aqua; margin: 20px"  
b.show()
```

styleSheet es css

```
Python Interactor  
>>>  
>>>  
>>>  
>>> b=qt.QPushButton('Toggle')  
>>> b.connect('clicked()',toggle)  
True  
>>> b.styleSheet = "font-size: 24pt; color: aqua; margin: 20px"  
>>> b.show()
```

Creación de un botón Qt

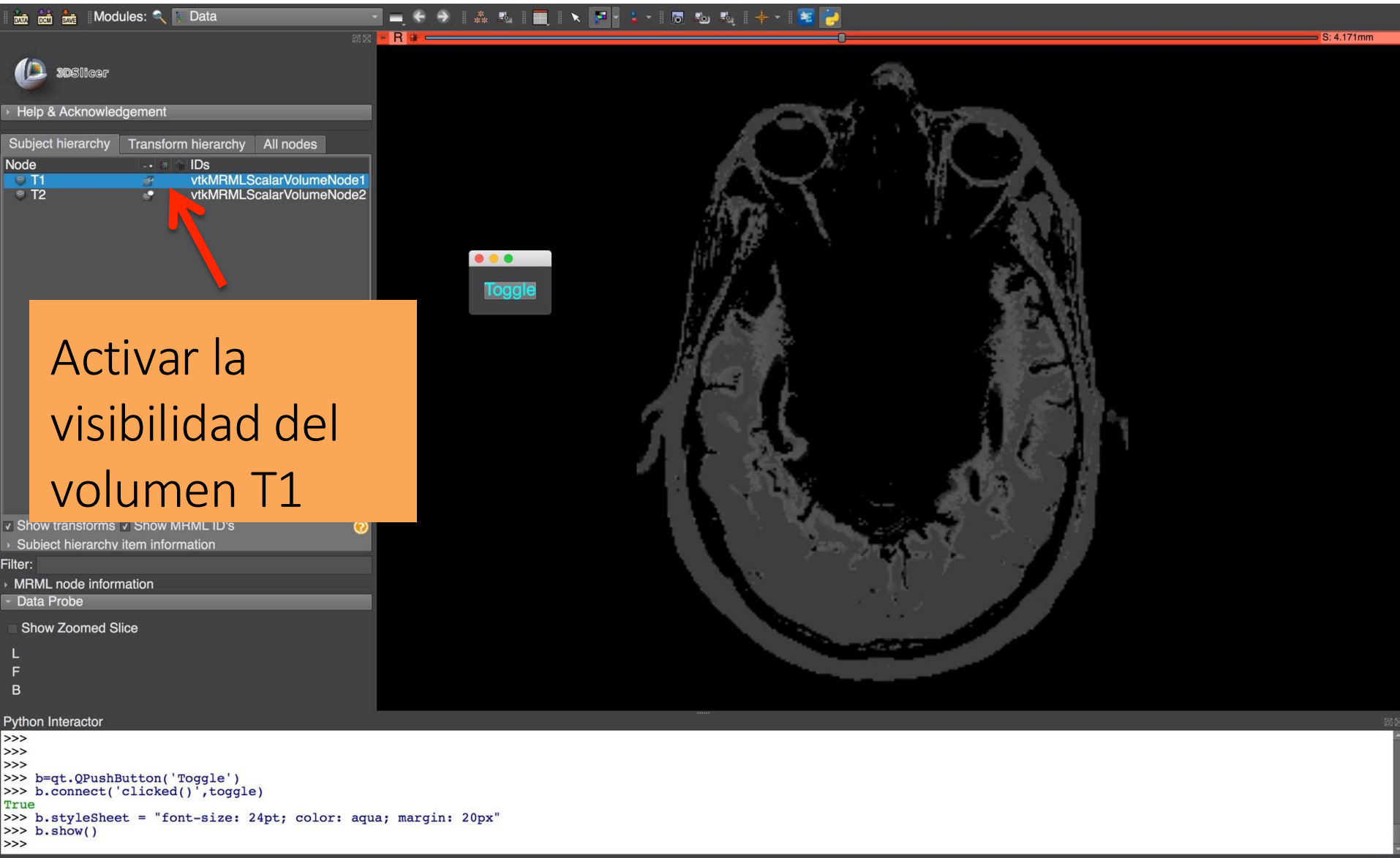


The screenshot shows the 3D Slicer interface. On the left, the 'Subject hierarchy' panel lists nodes T1 and T2. The main view displays an axial MRI brain scan. A small window with a 'Toggle' button is overlaid on the scan. The bottom of the interface shows a Python Interactor with the following code:

```
>>>  
>>>  
>>> b=qt.QPushButton('Toggle')  
>>> b.connect('clicked()',toggle)  
True  
>>> b.setStyleSheet = "font-size: 24pt; color: aqua; margin: 20px"  
>>> b.show()  
>>>
```

Aparece el botón Alternar

Creación de un botón Qt



3D Slicer

Help & Acknowledgement

Subject hierarchy Transform hierarchy All nodes

Node IDs

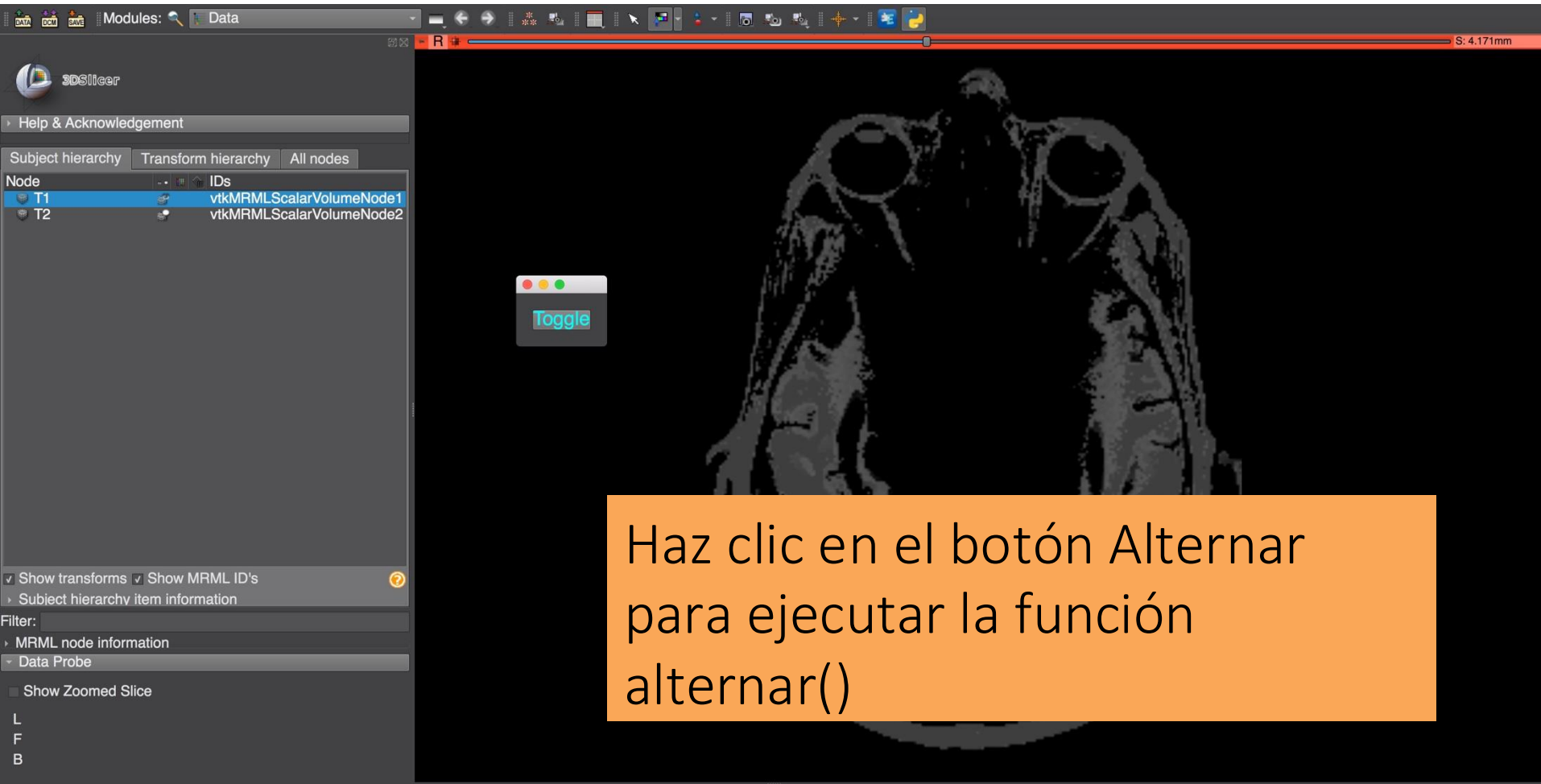
- T1 vtkMRMLScalarVolumeNode1
- T2 vtkMRMLScalarVolumeNode2

Toggle

Activar la visibilidad del volumen T1

```
>>>
>>>
>>> b=qt.QPushButton('Toggle')
>>> b.connect('clicked()',toggle)
True
>>> b.setStyleSheet = "font-size: 24pt; color: aqua; margin: 20px"
>>> b.show()
>>>
```

Creación de un botón Qt

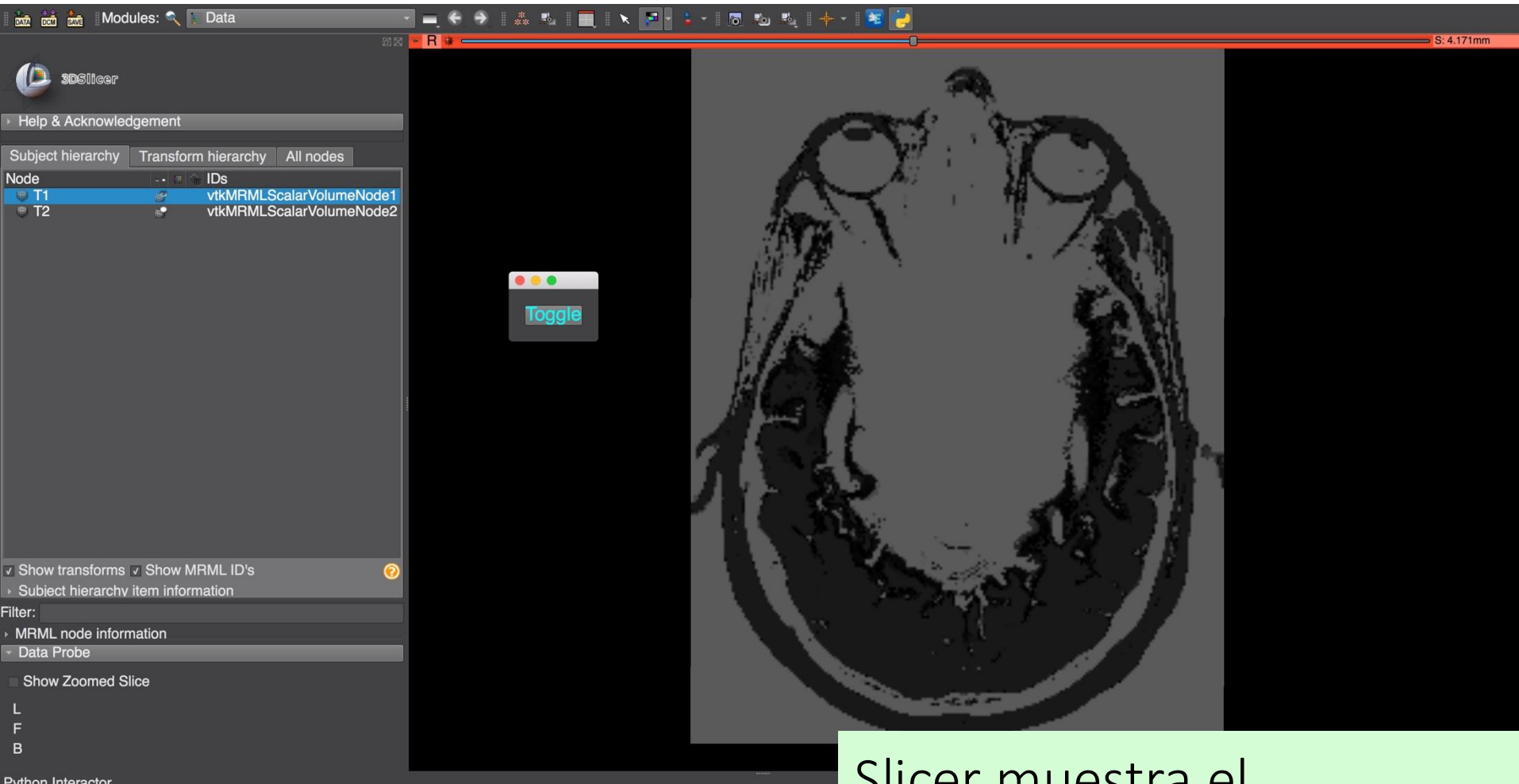


The screenshot shows the 3D Slicer interface. On the left, the 'Subject hierarchy' panel lists two nodes: 'T1' (vtkMRMLScalarVolumeNode1) and 'T2' (vtkMRMLScalarVolumeNode2). The main 3D view displays a grayscale volume rendering of a human torso. A small Qt window with a 'Toggle' button is overlaid on the 3D view. An orange text box is positioned over the bottom right of the 3D view.

Haz clic en el botón Alternar para ejecutar la función `alternar()`

```
Python Interactor
>>>
>>>
>>> b=qt.QPushButton('Toggle')
>>> b.connect('clicked()',toggle)
True
>>> b.setStyleSheet = "font-size: 24pt; color: aqua; margin: 20px"
>>> b.show()
>>>
```

Creación de un botón Qt



The screenshot shows the Slicer software interface. On the left, there is a sidebar with a 'Subject hierarchy' panel containing two nodes: 'T1' (vtkMRMLScalarVolumeNode1) and 'T2' (vtkMRMLScalarVolumeNode2). Below this is a 'Python Interactor' panel with the following code:

```
>>>
>>>
>>> b=qt.QPushButton('Toggle')
>>> b.connect('clicked()',toggle)
True
>>> b.setStyleSheet = "font-size: 24pt; color: aqua; margin: 20px"
>>> b.show()
>>>
```

In the center of the main view, a small window titled 'Toggle' is displayed, containing a blue button with the text 'Toggle'. The main view shows a grayscale axial MRI slice of a brain. The top right corner of the window displays 'S: 4.171mm'.

Slicer muestra el resultado del tratamiento de la imagen

Ejemplos de módulos con secuencias de comandos

- El tutorial muestra cómo crear una interfaz sencilla en Python.
- Slicer integra muchos módulos de secuencias de comandos sofisticados, como estadísticas de segmentos, datos de muestras, módulo de endoscopia, etc.
- Para más información, consulte el repositorio de secuencias de comandos de Slicer:

<https://www.slicer.org/wiki/Documentation/Nightly/ScriptRepository>

Conclusión

- Slicer permite a los desarrolladores crear interfaces complejas y ágiles para los usuarios
- La plataforma de software ofrece posibilidades de personalización ilimitadas
- Slicer da acceso a bibliotecas subyacentes avanzadas a través de un paquete multiplataforma que es fácil de desplegar a los usuarios

Agradecimientos



Neuroimage Analysis Center
(NIBIB P41 EB015902)



Sylvain Bouix, Ph.D.

Psychiatry Neuroimaging Laboratory