



Tutorial de Desenvolvimento do *Slicer*: Programando no *Slicer*

Sonia Pujol, Ph.D.

Professora Assistente de Radiologia
Diretora de Treinamento e Educação do 3D Slicer
Brigham and Women's Hospital
Escola de Medicina de Harvard

Steve Pieper, Ph.D.

Arquiteto-chefe do 3D Slicer
Isomics Inc.

Objetivo do Tutorial



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



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

Este tutorial é uma introdução ao *Python interactor* e ao conjunto de *widgets Qt* na versão 5 do 3D Slicer.

Roteiro do Tutorial



Parte 1: Visão Geral dos Módulos do 3D Slicer



Parte 2: Familiarizando-se com o ambiente Python no 3D Slicer

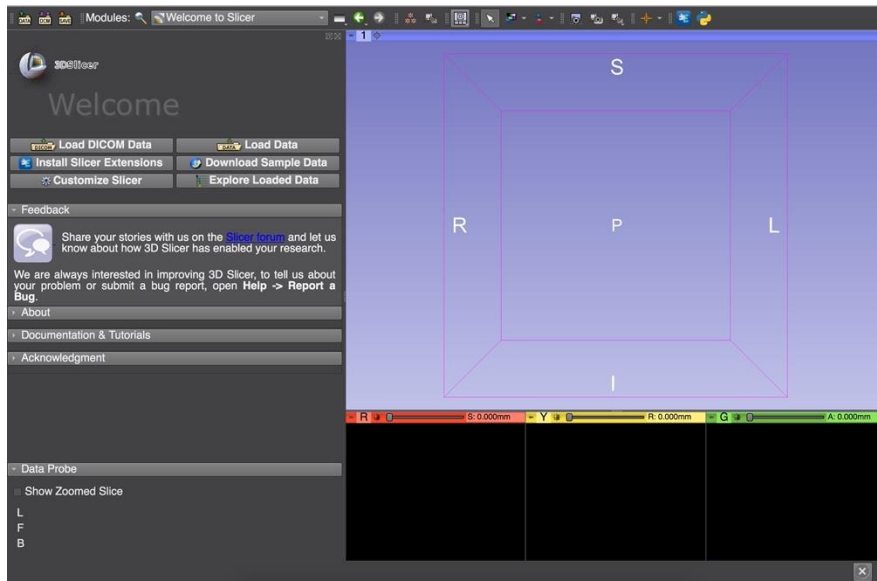


Parte 3: Familiarizando-se com o conjunto de widgets Qt no 3D Slicer

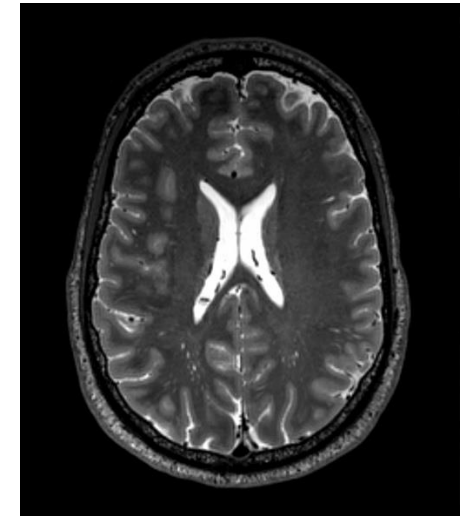
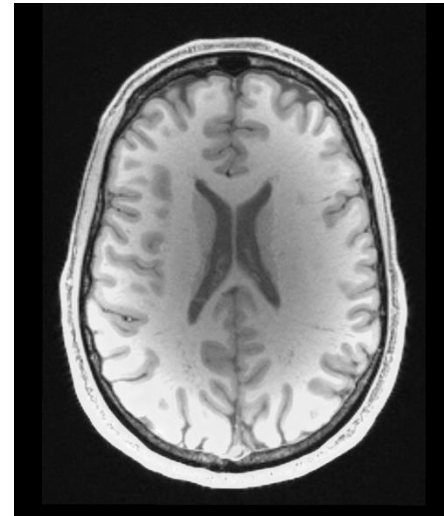
Aviso

- O 3D Slicer é um *software* livre de código aberto distribuído sob uma licença no estilo BSD.
- O *software* não é aprovado pela FDA nem possui marcação CE, sendo destinado apenas para uso em pesquisa.

Materiais de Tutorial



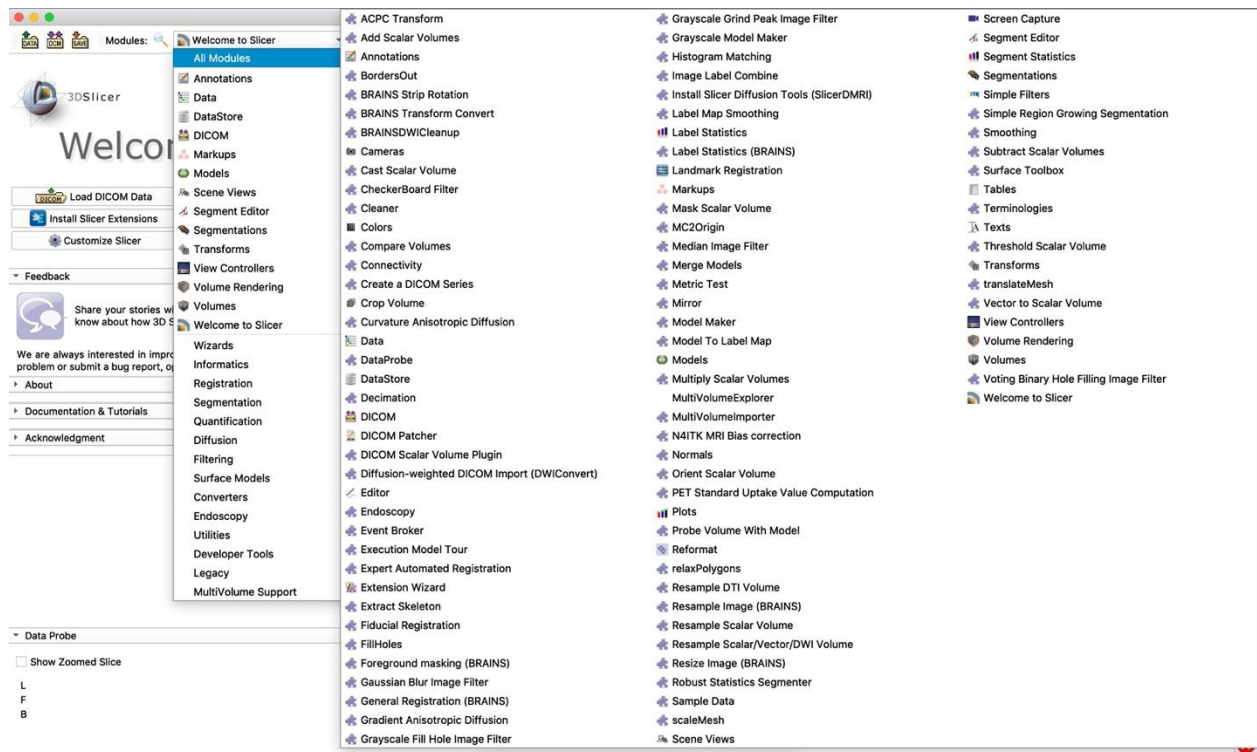
3D Slicer versão 5



SlicerProgrammingTutorialData.zip

Parte 1

Um panorama dos módulos do Slicer



3D Slicer

<> Code Issues 544 Pull requests 52 Actions Wiki Security Insights

Slicer Public Watch 43 Fork 575 Star 1.8k

main 12 Branches 35 Tags Go to file Add file Code

JamesButler and jcf: ENH: Make NSIS Windows installer prettier with applic... #869775 · 15 hours ago 29,614 Commits

.github	COMP: Bump github/codeql-action from 3.28.1 to 3.28.5	3 days ago
Applications	ENH: Make NSIS Windows installer prettier with application ...	15 hours ago
Base	COMP: Fix Windows build errors by explicitly including Windo...	16 hours ago
CMake	ENH: Make NSIS Windows installer prettier with application ...	15 hours ago
Docs	DOC: Update Transforms module API documentation adding...	5 days ago
Extensions	COMP: Update CLI modules for compatibility with modern IT...	2 days ago

About Multi-platform, free open source software for visualization and image computing.

www.slicer.org

python c-plus-plus qt image-processing medical-imaging registration neuroimaging segmentation vtk itk national-institutes-of-health medical-image-computing 3d-printing 3d-slicer tractography image-guided-therapy nih computed-tomography kitware tcia-dac

- O 3D Slicer é uma plataforma de código aberto para análise e visualização de dados de imagens médicas.
- O 3D Slicer é compilado e testado todos os dias nas plataformas Windows, Mac e Linux.
- O código-fonte está disponível gratuitamente no GitHub em <http://github.com/Slicer/Slicer>.

Módulos do Slicer

O 3D Slicer suporta três tipos de módulos:

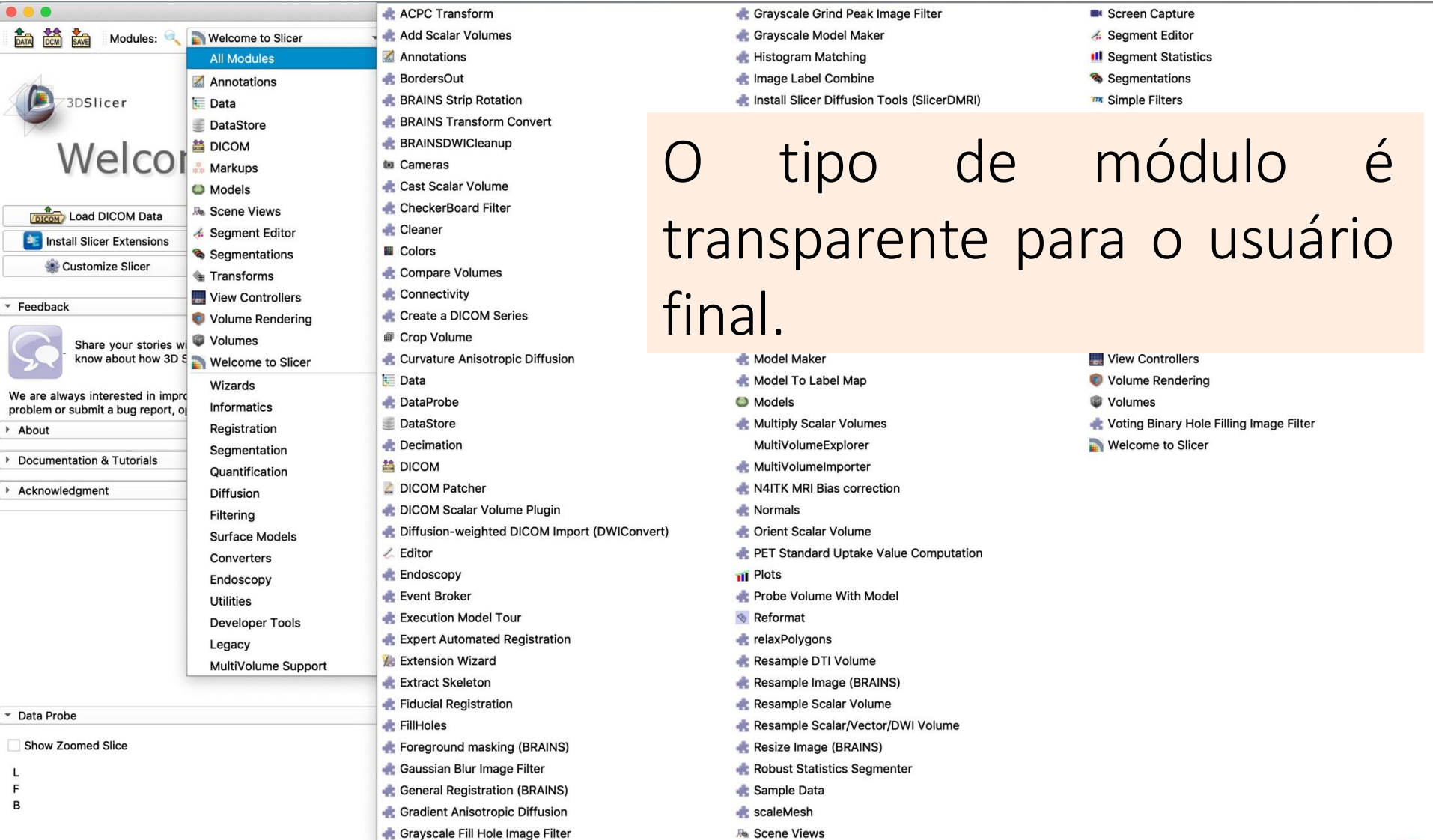
Command-line Interface (CLI) [Interface de Linha de Comando]: executável independente com argumentos de entrada/saída limitados.

Módulos Carregáveis (Plugins em C++): otimizados para computação pesada.

Foco deste tutorial

- Módulos com *scripts* (Python): recomendados para prototipagem rápida e desenvolvimento de fluxos de trabalho.

Módulos do Slicer



The image shows the Slicer software interface with the 'Modules' list open on the left. The list includes various modules such as 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, Scene Views, Screen Capture, Segment Editor, Segment Statistics, Segmentations, Simple Filters, View Controllers, Volume Rendering, Volumes, Voting Binary Hole Filling Image Filter, and Welcome to Slicer.

O tipo de módulo é transparente para o usuário final.

Extensões do Slicer

Uma Extensão do Slicer é um pacote de distribuição que agrupa um ou mais módulos do Slicer.



SwissSkullStripper
Bill Lorensen (Noware...
★★★★★ (0)

INSTALL



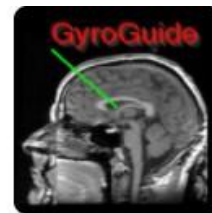
PETTumorSegmenta...
Christian Bauer (Univ...
★★★★★ (0)

INSTALL



SlicerOpenIGTLink
Junichi Tokuda (SPL), ...
★★★★★ (0)

INSTALL



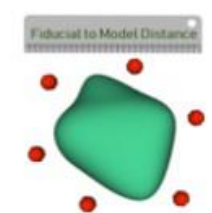
GyroGuide
Ruifeng Chen, Luping...
★★★★★ (0)

INSTALL



PET-IndiC
Ethan Ulrich (Universi...
★★★★★ (0)

INSTALL



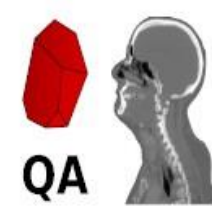
FiducialsToModelDi...
Jesse Reynolds (Cante...
★★★★★ (0)

INSTALL



Slicer-Wasp
Thomas Lawson (MR...
★★★★★ (0)

INSTALL

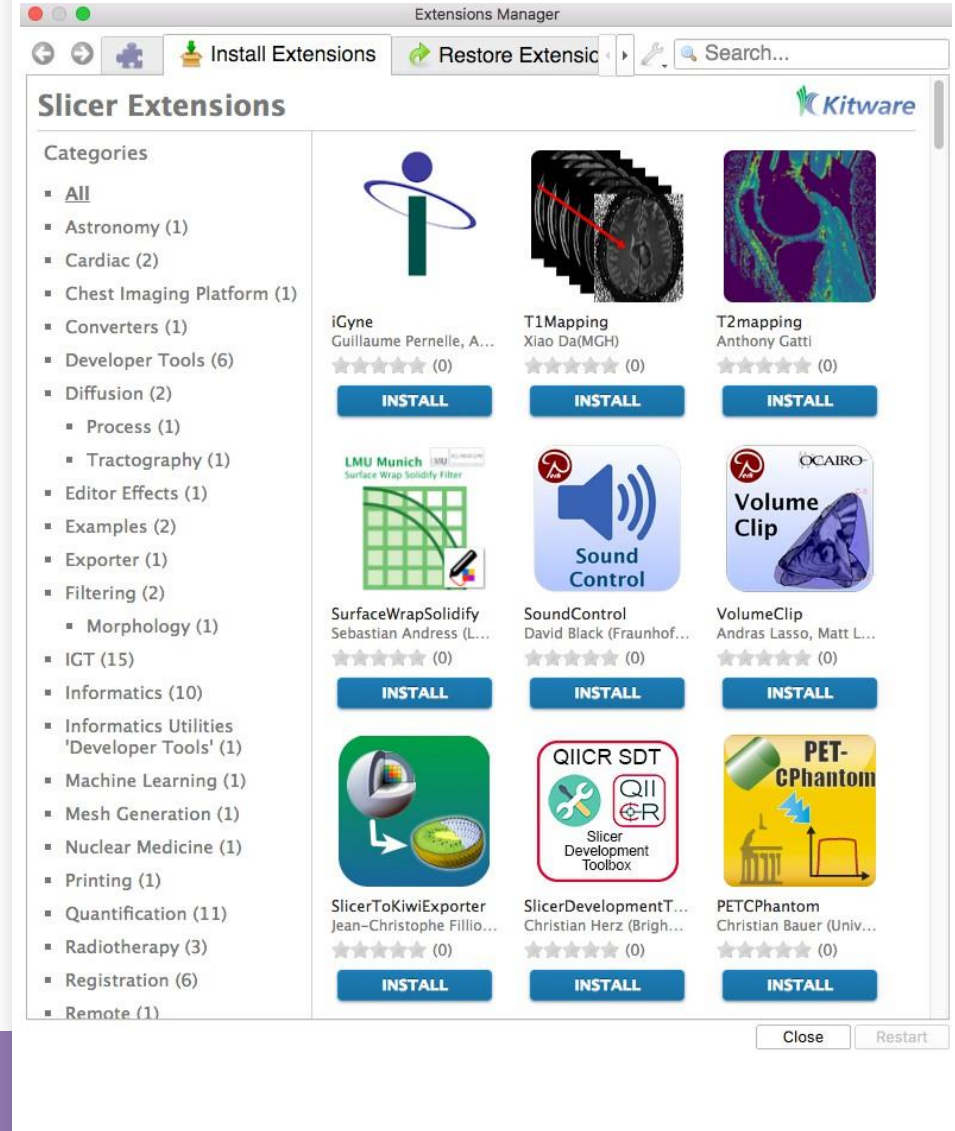


ImageCompare
Paolo Zaffino (Magna ...
★★★★★ (0)

INSTALL

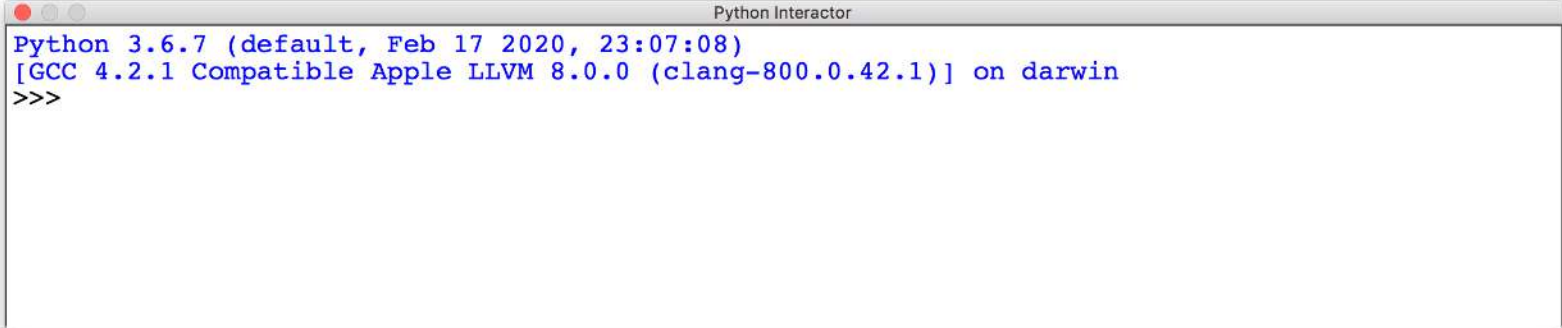
Gerenciador de Extensões do Slicer

- O Gerenciador de Extensões do Slicer oferece uma plataforma do tipo “loja de aplicativos” para o ecossistema do 3D Slicer.
- O Gerenciador de Extensões permite a criação e a instalação fácil de extensões do Slicer.
- A versão de lançamento do Slicer 5 inclui mais de 130 extensões.



Parte 2

Familiarizando-se com o ambiente Python no 3D Slicer



```
Python Interactor
Python 3.6.7 (default, Feb 17 2020, 23:07:08)
[GCC 4.2.1 Compatible Apple LLVM 8.0.0 (clang-800.0.42.1)] on darwin
>>>
```

Python no Slicer

O Slicer v. 5 funciona com Python 3 e um conjunto rico de bibliotecas padrão.



O NumPy é o pacote fundamental para a computação científica com Python.



O VTK é uma biblioteca de código aberto para manipulação e apresentação de dados científicos.



O ITK é uma biblioteca de código aberto para análise de imagens.



CTK é uma biblioteca de código aberto para a computação de imagens biomédicas.



PythonQt é uma ligação Python para Qt.

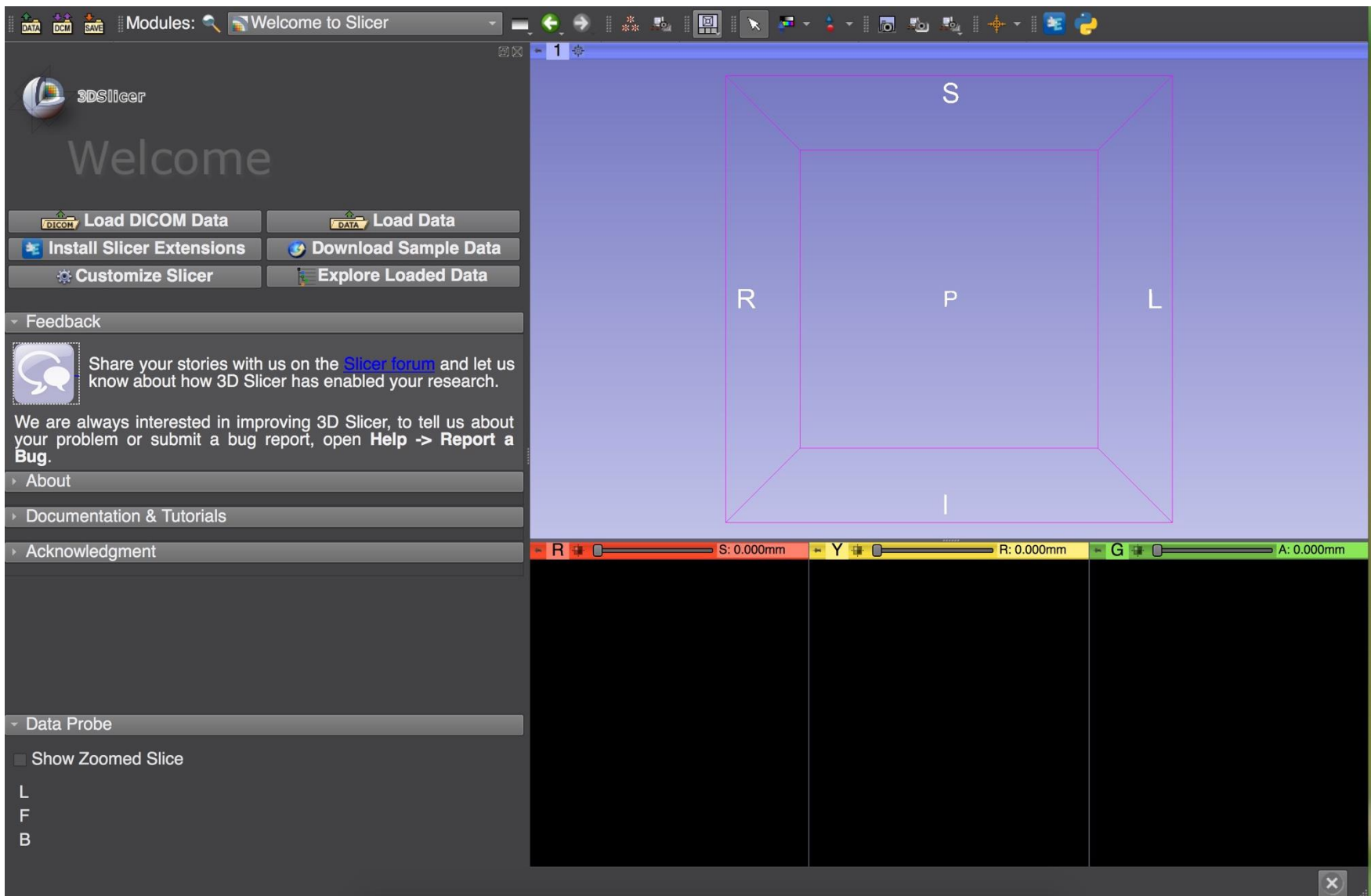


O Qt é uma estrutura multiplataforma utilizada como um conjunto de ferramentas gráficas.

Python no Slicer



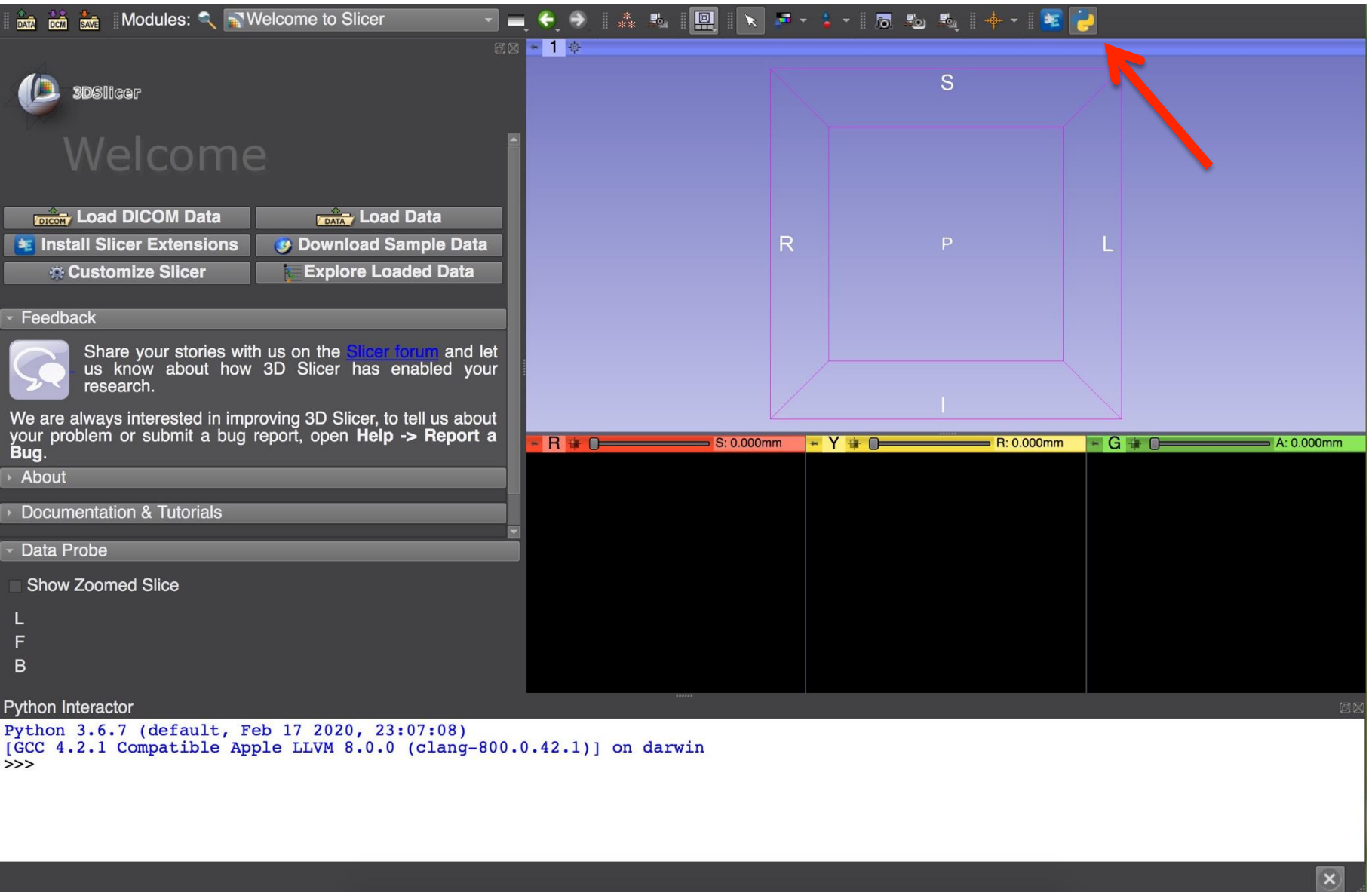
- O índice de pacotes Python (PyPi) dá acesso a mais de 200.000 pacotes Python adicionais (<http://pipy.org>).
- O comando *pip install* no Slicer permite que os programadores instalem as ferramentas de computação científica mais comuns (por exemplo, *TensorFlow*, *SciPy*, *PyTorch*, *Pandas* etc.).
- O Slicer pode ser usado como um *kernel* do *Jupyter Notebook*.
- PyCharm e outras ferramentas de desenvolvimento Python podem ser usadas com o Slicer.



A versão 5 do Slicer integra Python3, VTK5 e ITK5

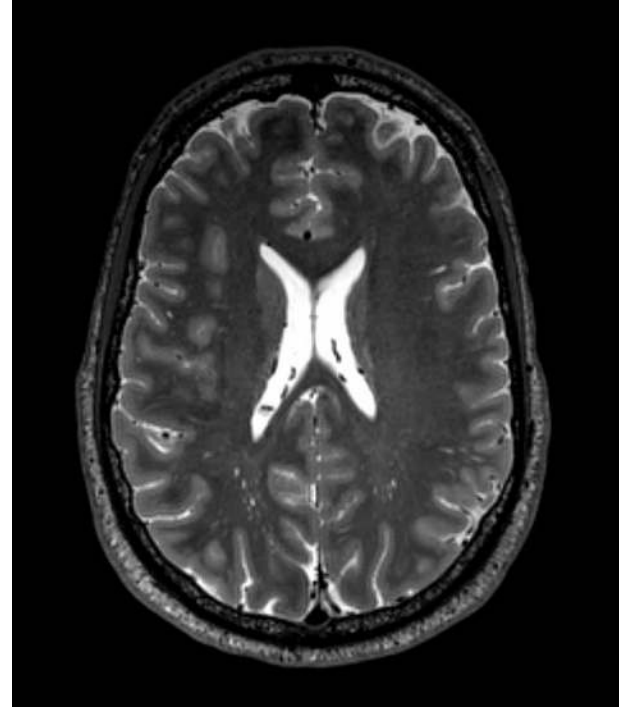
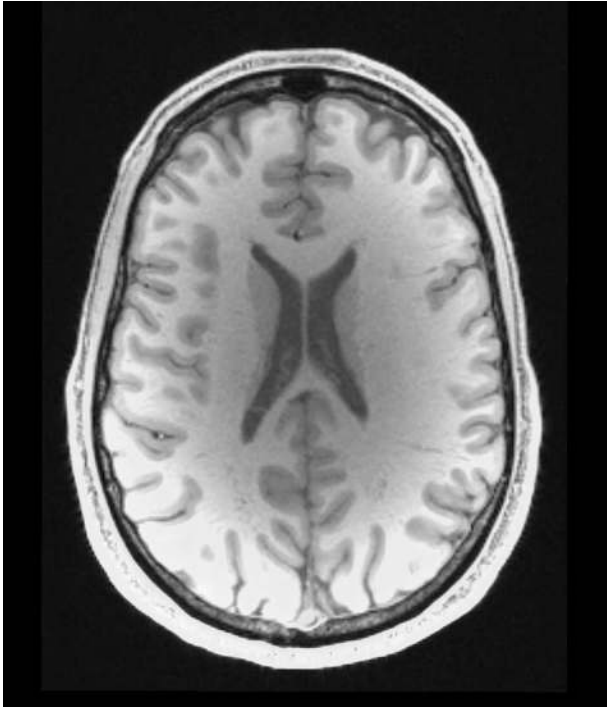
O console Python no Slicer

O Python Interactor é um console baseado em Qt que permite acesso direto aos Nós MRML do Slicer, a bibliotecas (NumPy, VTK, ITK, CTK) e ao Qt.



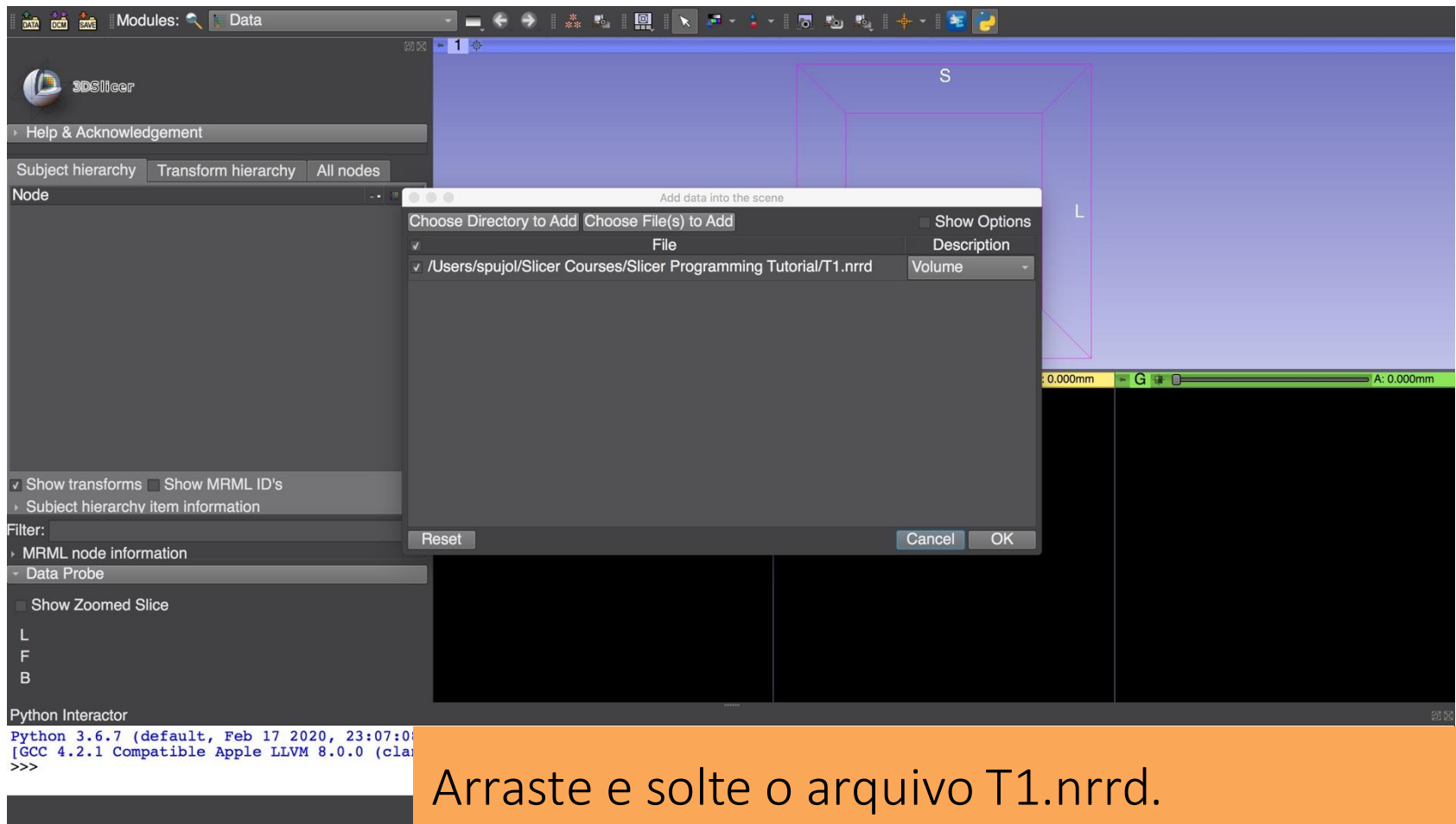
Para acessar o Python Interactor, clique no ícone do Python no menu da barra superior do Slicer.





O conjunto de dados do tutorial de programação do Slicer inclui exames de ressonância magnética ponderados em T1 e T2 de um sujeito saudável.

Conjunto de dados do tutorial



Arraste e solte o arquivo T1.nrrd.

Clique em OK para carregar o arquivo no Slicer.

Conjunto de dados do tutorial

The screenshot displays the 3D Slicer software interface. The main window shows a 3D coordinate system with axes labeled S (Superior), I (Inferior), R (Right), and L (Left). Below the 3D view, three orthogonal slices are visible: an axial slice (left), a sagittal slice (middle), and a coronal slice (right). The slice positions are indicated by sliders at the bottom of the 3D view, with values: S: 12.971mm, R: 3.818mm, and A: 21.967mm.

The left sidebar contains the following panels:

- Help & Acknowledgement
- Subject hierarchy (selected), Transform hierarchy, All nodes
- Node: T1
- Filter: MRML node information, Data Probe
- Show Zoomed Slice
- L, F, B
- Python Interactor

The Python Interactor at the bottom shows the following text:

```
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  
>>>
```

Visão Geral

- O Slicer é um *software* livre e de código aberto.
- Há milhares de imagens médicas sofisticadas disponíveis na *internet* que você pode visualizar e analisar com o 3D Slicer.

Modelo de Dados do Slicer



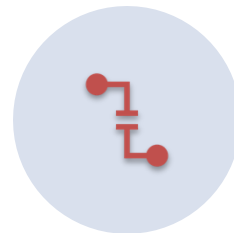
O Modelo de Dados do Slicer é baseado na Estrutura de Dados da Cena do Slicer.



A *Medical Reality Markup Language* (MRML) [Linguagem de Marcação da Realidade Médica] é uma linguagem baseada em XML usada para serializar o conteúdo da cena do Slicer no disco (scene.mrml).



Uma cena do Slicer é uma coleção de imagens, anotações, modelos 3D, transformações espaciais, marcas de referência e câmeras.



Cada elemento de uma cena é chamado de nó MRML.

Nós MRML do Slicer: Tipos Básicos



Nó de dados: Armazena os dados brutos



Nó de exibição: Descreve como os dados devem ser visualizados



Nó de armazenamento: Descreve como os dados devem ser armazenados no disco

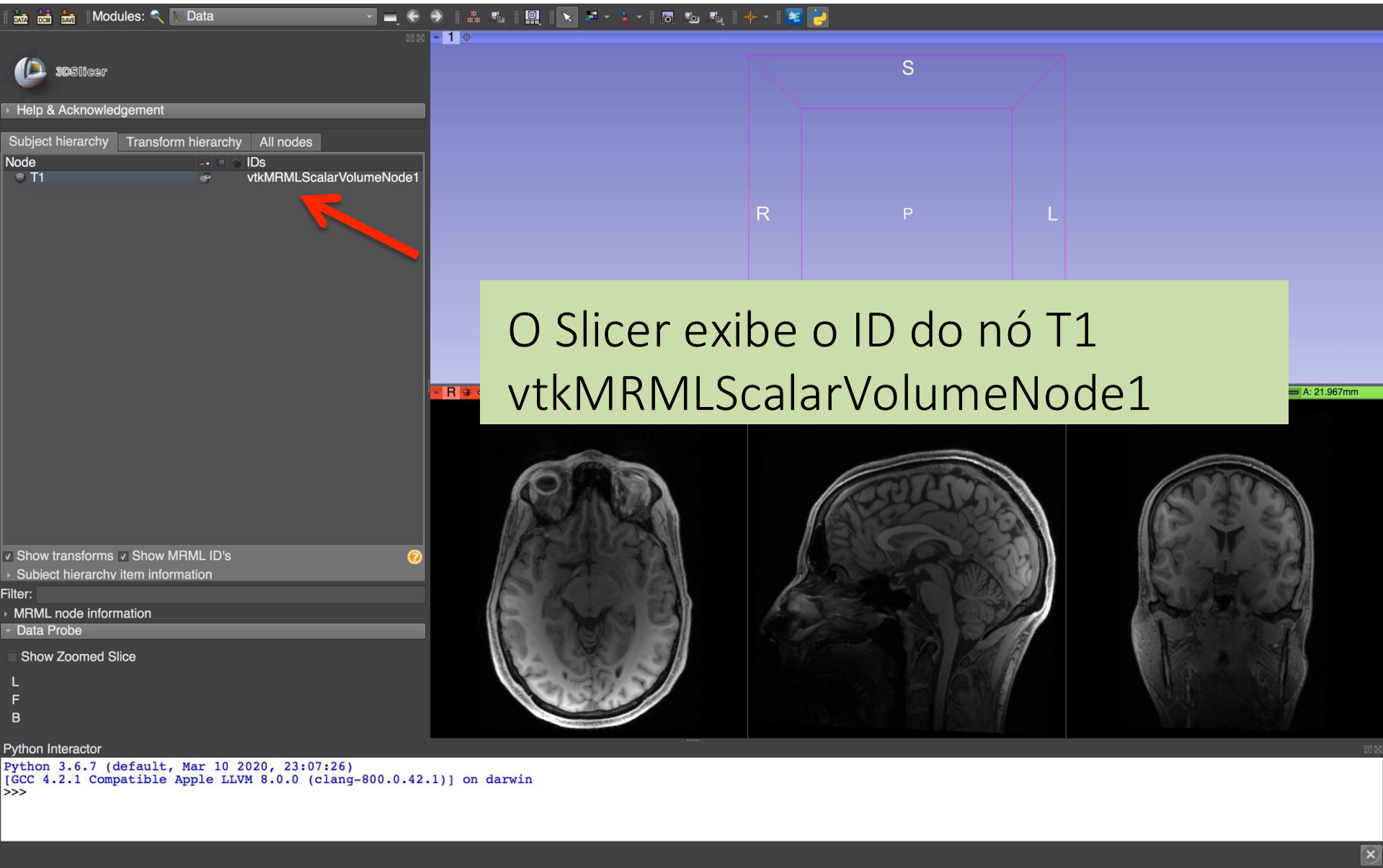
Conjunto de dados do Tutorial

The screenshot displays the 3DSlicer software interface. At the top, there is a menu bar with 'Modules' and 'Data'. Below it is a toolbar with various icons. The main window shows a 3D coordinate system with axes labeled S (Superior), I (Inferior), R (Right), and L (Left). Below the 3D view are three sliders for R (Red), Y (Yellow), and G (Green) channels, with values S: 12.971mm, R: 3.818mm, and A: 21.967mm. At the bottom, there are three MRI brain scan slices: an axial view on the left, a sagittal view in the middle, and a coronal view on the right. On the left side, there is a sidebar with 'Help & Acknowledgement', 'Subject hierarchy', 'Transform hierarchy', and 'All nodes'. Under 'Node', there is a 'T1' node. Below this, there are checkboxes for 'Show transforms' (checked) and 'Show MRML ID's' (unchecked). A red arrow points to the 'Show MRML ID's' checkbox. Below the sidebar, there is a 'Filter:' section with 'MRML node information' and 'Data Probe'. At the bottom left, there is a 'Python Interactor' window showing the following text:

```
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
>>>
```

Selezione Mostrar IDs MRML.

Modelo de Dados do Slicer

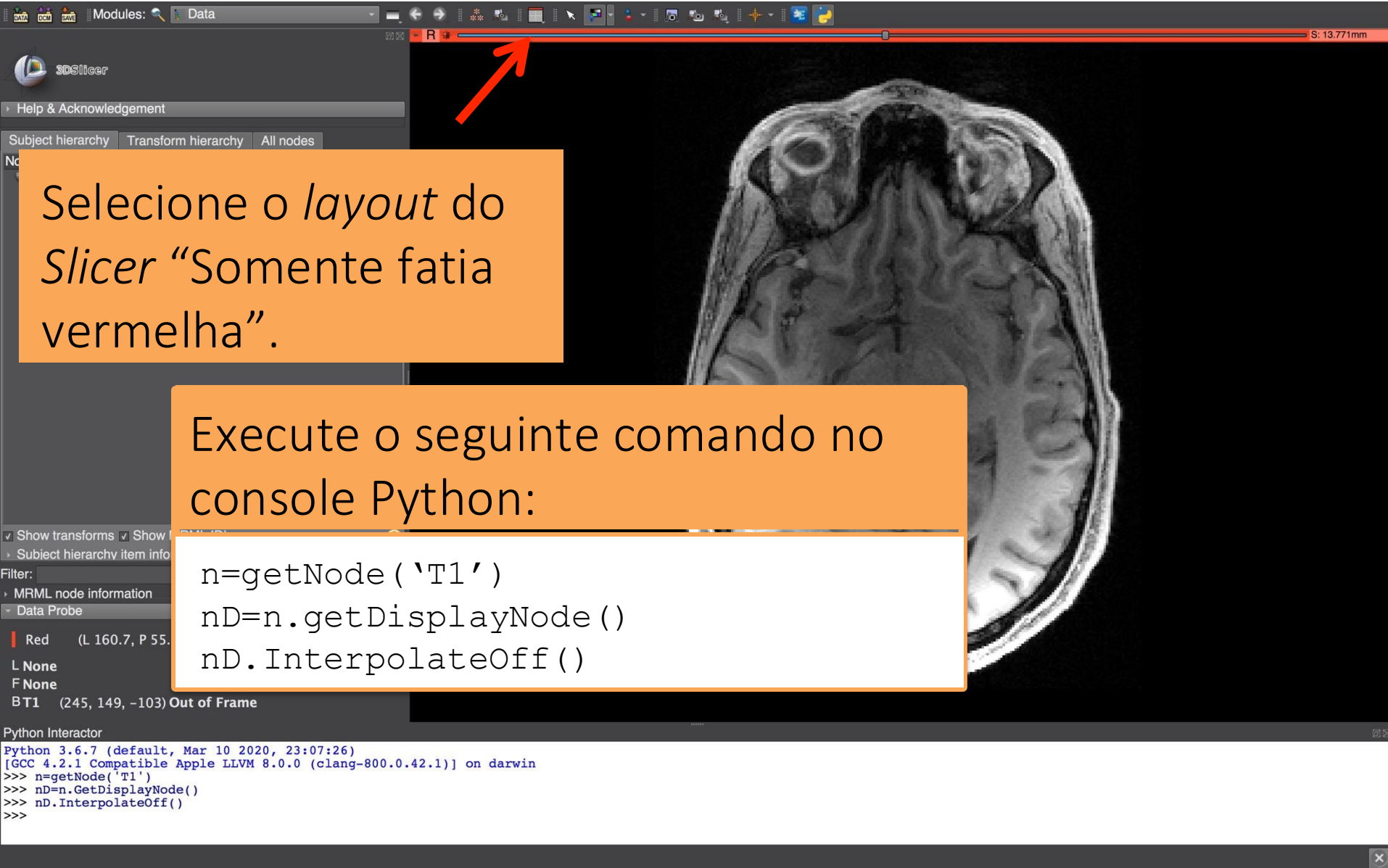


The image shows the Slicer software interface. On the left, the 'All nodes' panel is visible, showing a tree structure with a node 'T1' highlighted. A red arrow points to the ID 'vtkMRMLScalarVolumeNode1' associated with 'T1'. The main 3D view shows a purple volume with axes labeled 'S' (Superior), 'R' (Right), and 'L' (Left). Below the 3D view, three MRI slices are displayed: an axial slice on the left, a sagittal slice in the middle, and a coronal slice on the right. A green text box is overlaid on the 3D view, containing the text: 'O Slicer exibe o ID do nó T1 vtkMRMLScalarVolumeNode1'. At the bottom, a Python Interactor window shows the Python version (3.6.7) and the operating system (darwin).

O Slicer exibe o ID do nó T1
vtkMRMLScalarVolumeNode1

```
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  
>>>
```

Acessando nós MRML a partir do Python interactor



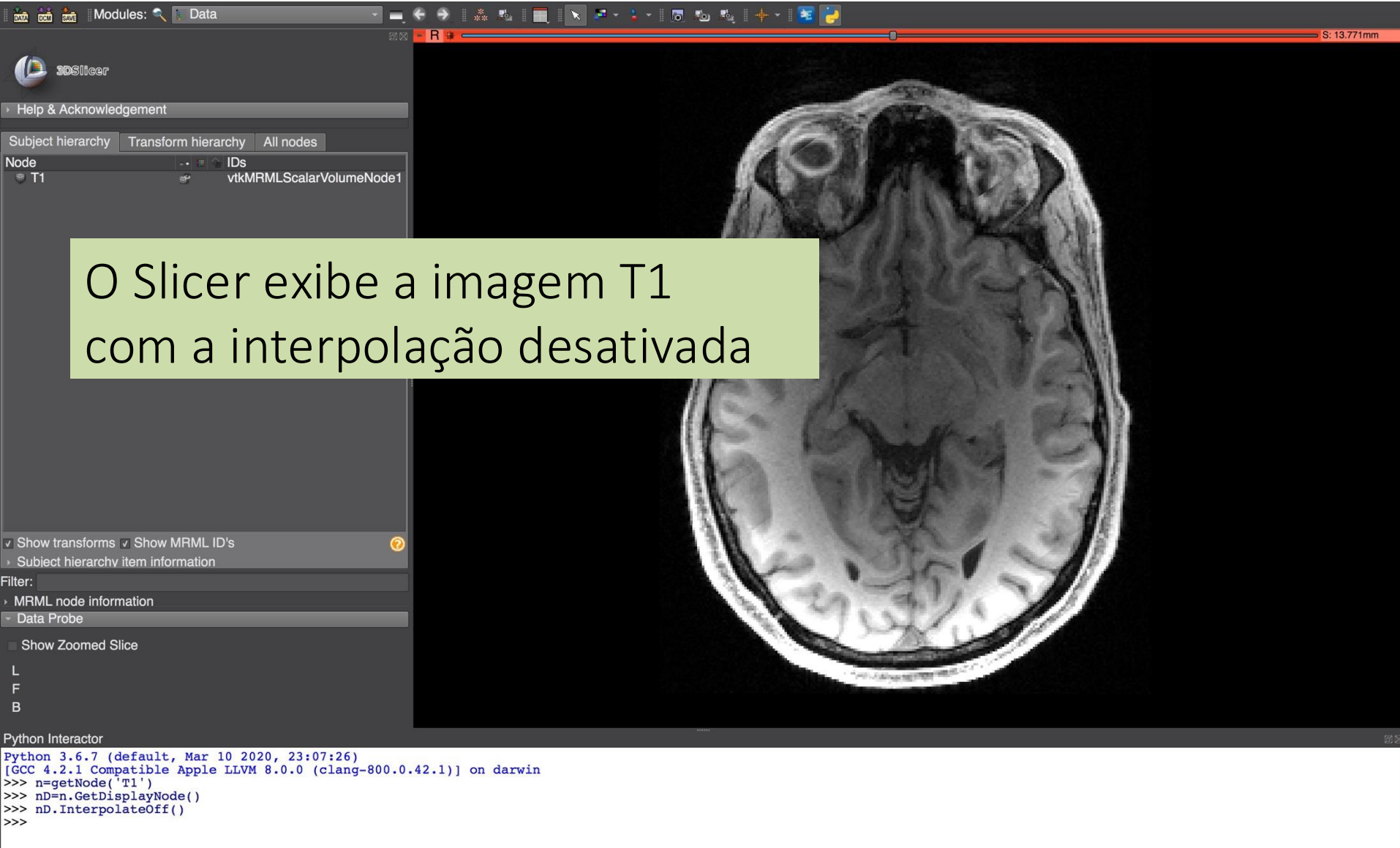
Selecione o *layout* do *Slicer* “Somente fatia vermelha”.

Execute o seguinte comando no console 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()
>>>

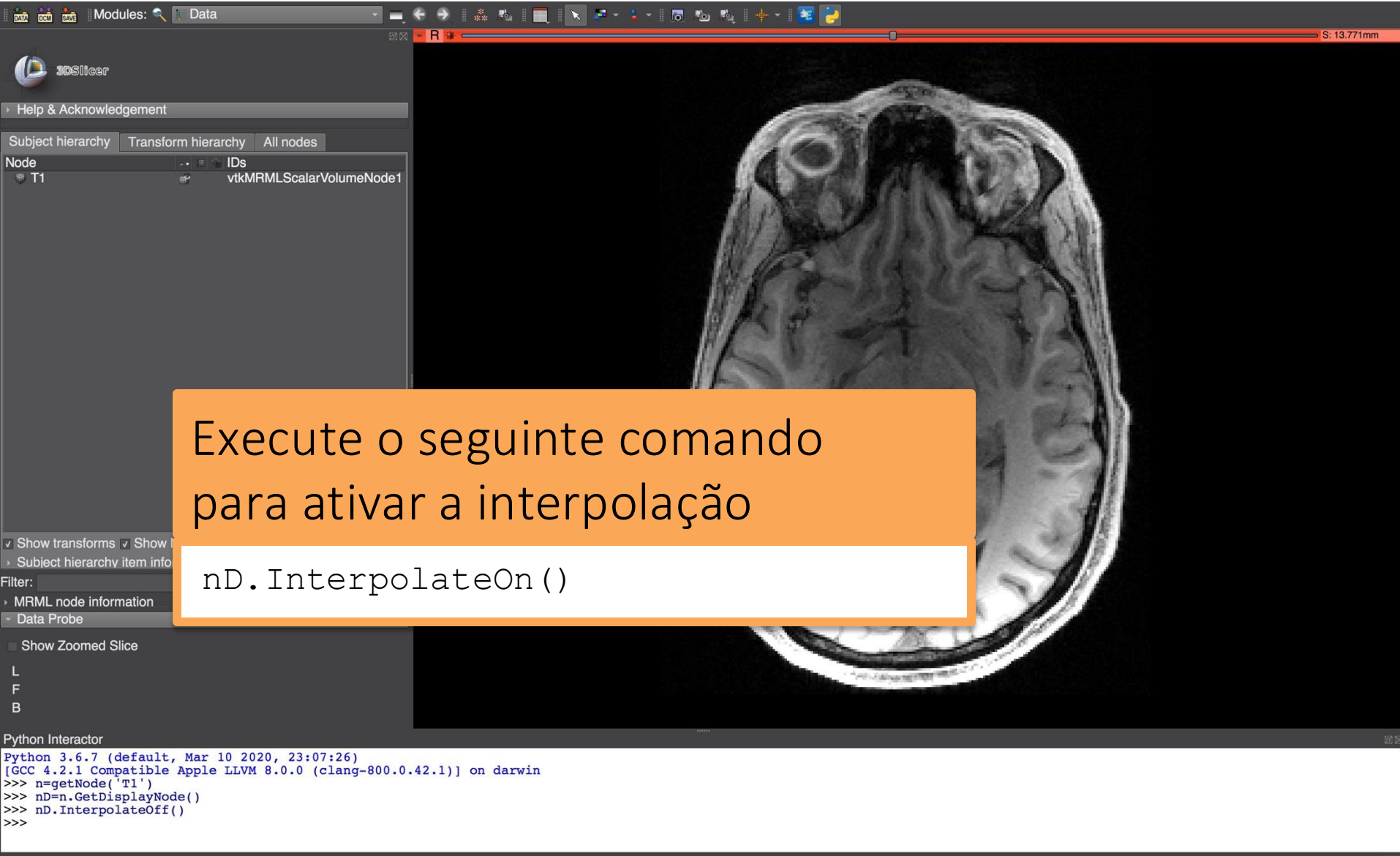
Acessando nós MRML a partir do Python interactor



O Slicer exibe a imagem T1 com a interpolação desativada

```
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()
>>>
```

Acessando nós MRML a partir do Python interactor

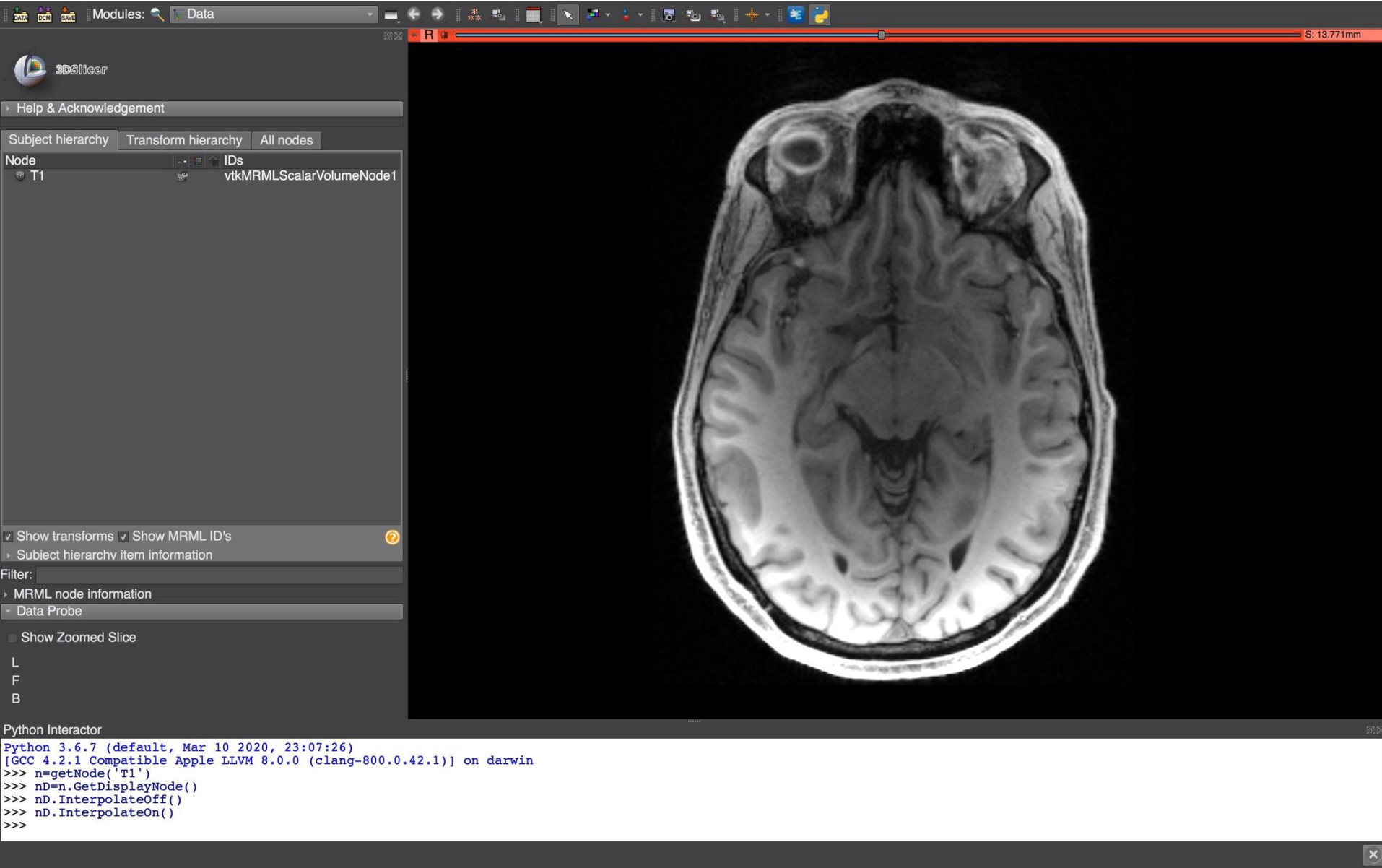


Execute o seguinte comando para ativar a interpolação

```
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()
>>>
```

Acessando nós MRML a partir do Python interactor



The screenshot displays the 3D Slicer software interface. The main window shows an axial MRI scan of a brain. The left sidebar contains a 'Node' list with 'T1' selected, showing its ID as 'vtkMRMLScalarVolumeNode1'. Below the node list, there are checkboxes for 'Show transforms', 'Show MRML ID's', and 'Show Zoomed Slice'. The bottom of the interface features a 'Python Interactor' terminal window with the following code:

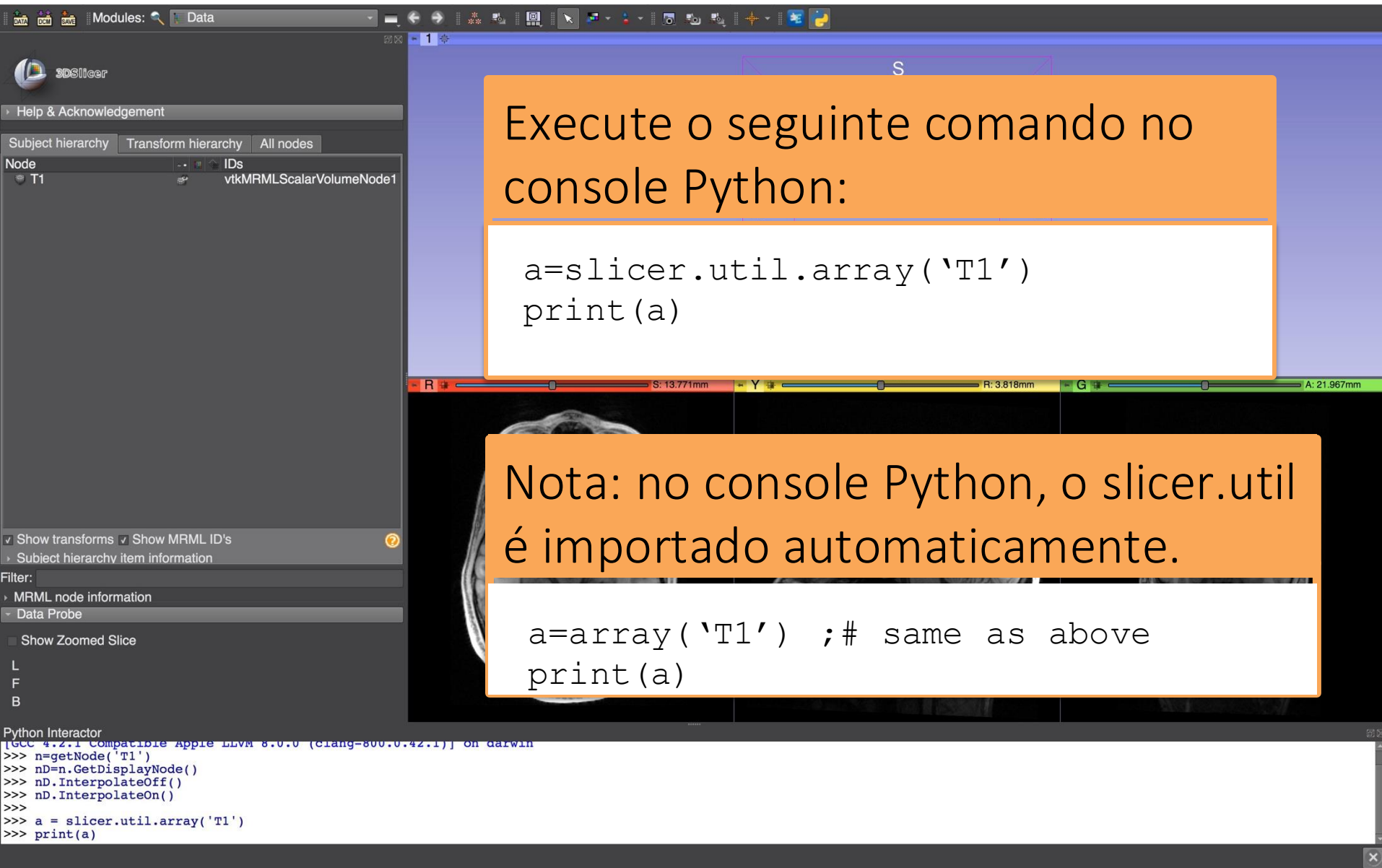
```
Python 3.6.7 (default, Mar 10 2020, 23:07:26)
[GCC 4.2.1 Compatible Apple LLVM 8.0.0 (clang-800.0.42.1)] on darwin
>>> n=getNode('T1')
>>> nd=n.GetDisplayNode()
>>> nd.InterpolateOff()
>>> nd.InterpolateOn()
>>>
```

Acessando vóxeis em um volume

- O pacote `slicer.util` oferece acesso aos volumes como *arrays* multidimensionais do *NumPy*.
- Os volumes podem ser modificados utilizando métodos padrão do *NumPy*.



Acessando vóxeis em um volume



The image shows a screenshot of the Slicer software interface. On the left, there is a sidebar with a 'Subject hierarchy' panel showing a tree structure with a node 'T1' of type 'vtkMRMLScalarVolumeNode1'. Below this is a 'Python Interactor' window with a terminal-like interface. The main window displays a 3D volume with a red box labeled 'S' indicating a slice. The bottom of the interface shows a coordinate system with axes R (13.771mm), Y, and G (21.967mm).

Execute o seguinte comando no console Python:

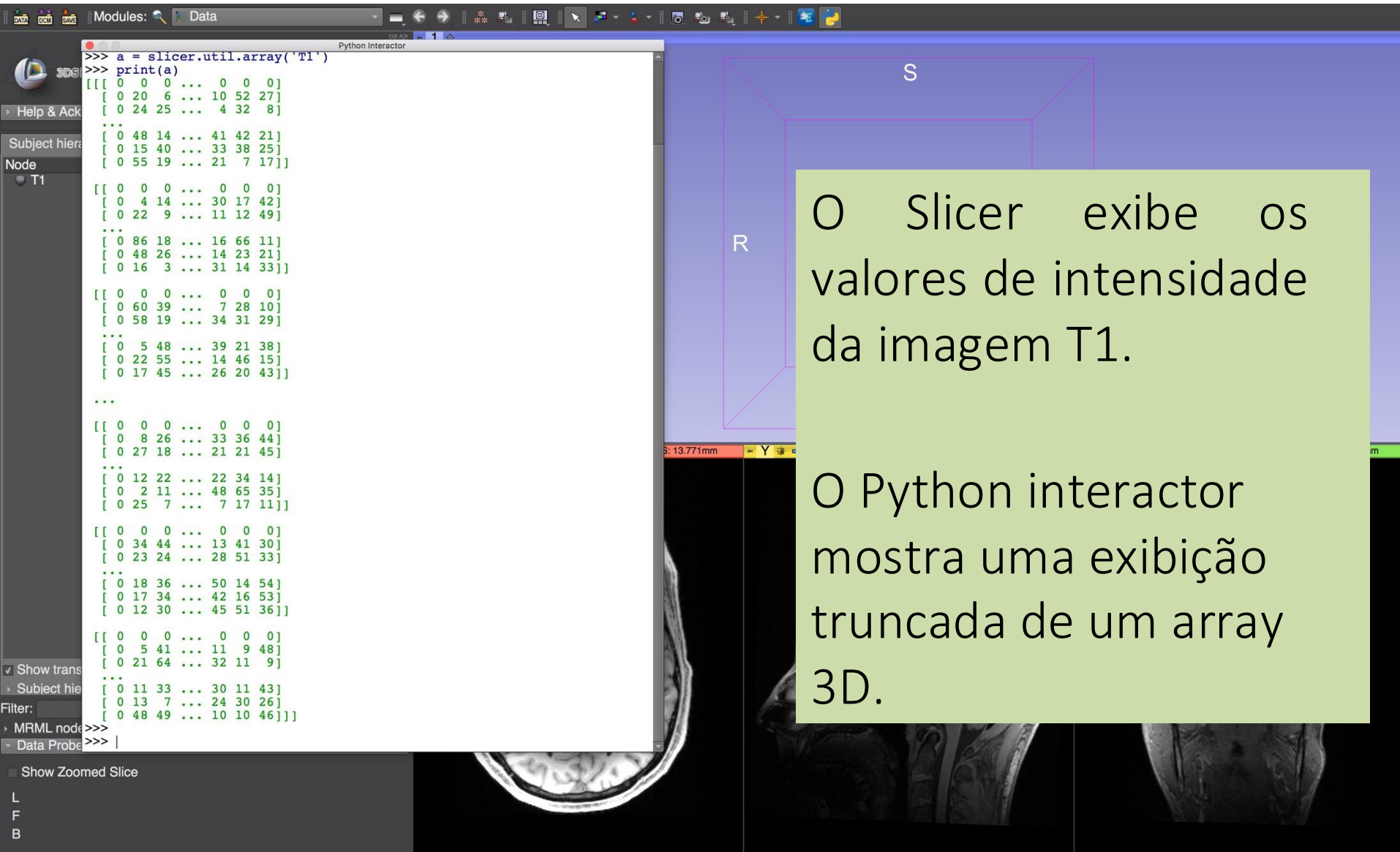
```
a=slicer.util.array('T1')  
print(a)
```

Nota: no console Python, o slicer.util é importado automaticamente.

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

```
Python Interactor  
[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)
```

Acessando vóxeis em um volume



The image shows a screenshot of the Slicer software interface. On the left, a Python Interactor window displays the following code and output:

```
>>> 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]]]

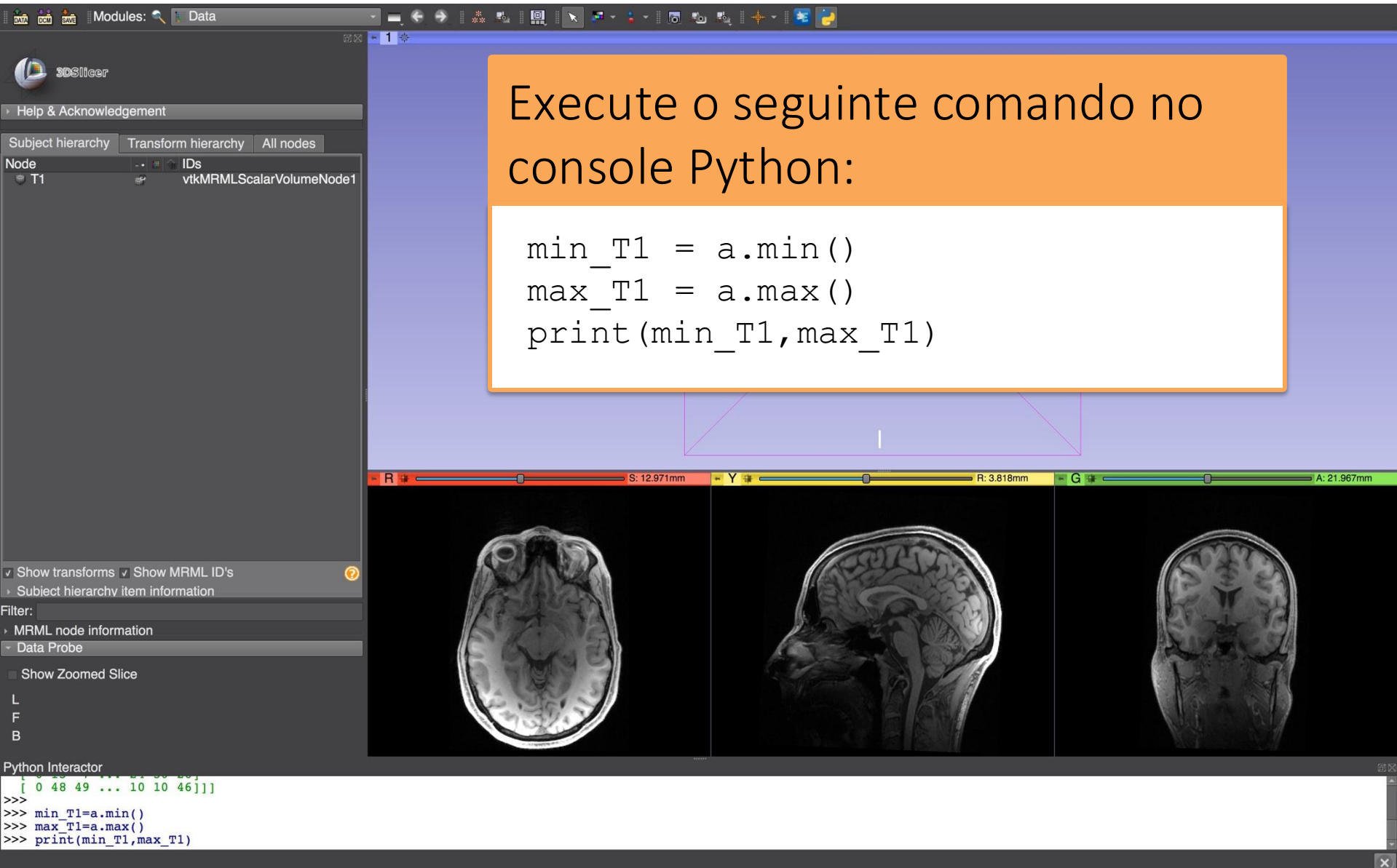
[[[ 0  0  0 ...  0  0  0]
 [ 0  5 41 ... 11  9 48]
 [ 0 21 64 ... 32 11  9]
 ...
 [ 0 11 33 ... 30 11 43]
 [ 0 13  7 ... 24 30 26]
 [ 0 48 49 ... 10 10 46]]]
>>>
```

On the right, a 3D volume rendering of a brain slice is shown. A purple bounding box is overlaid on the volume, with the letter 'S' at the top and 'R' on the left side. A green text box is overlaid on the 3D view, containing the following text:

O Slicer exibe os valores de intensidade da imagem T1.

O Python interactor mostra uma exibição truncada de um array 3D.

Acessando vóxeis em um volume



The image shows a screenshot of the 3D Slicer software interface. The main window displays three orthogonal MRI slices of a brain: axial (left), sagittal (middle), and coronal (right). Above the slices are sliders for slice position: S: 12.971mm, R: 3.818mm, and A: 21.967mm. A large orange box in the center of the main window contains the text: "Execute o seguinte comando no console Python:". Below this box, a white box contains the following Python code:

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

The Python Interactor at the bottom of the interface shows the execution of the code, with the output: [0 48 49 ... 10 10 46]]].

On the left side, the "Subject hierarchy" panel shows a tree structure with "Node" and "T1" (vtkMRMLScalarVolumeNode1). The "Data Probe" panel is also visible, showing "Show Zoomed Slice" and "L F B" options.

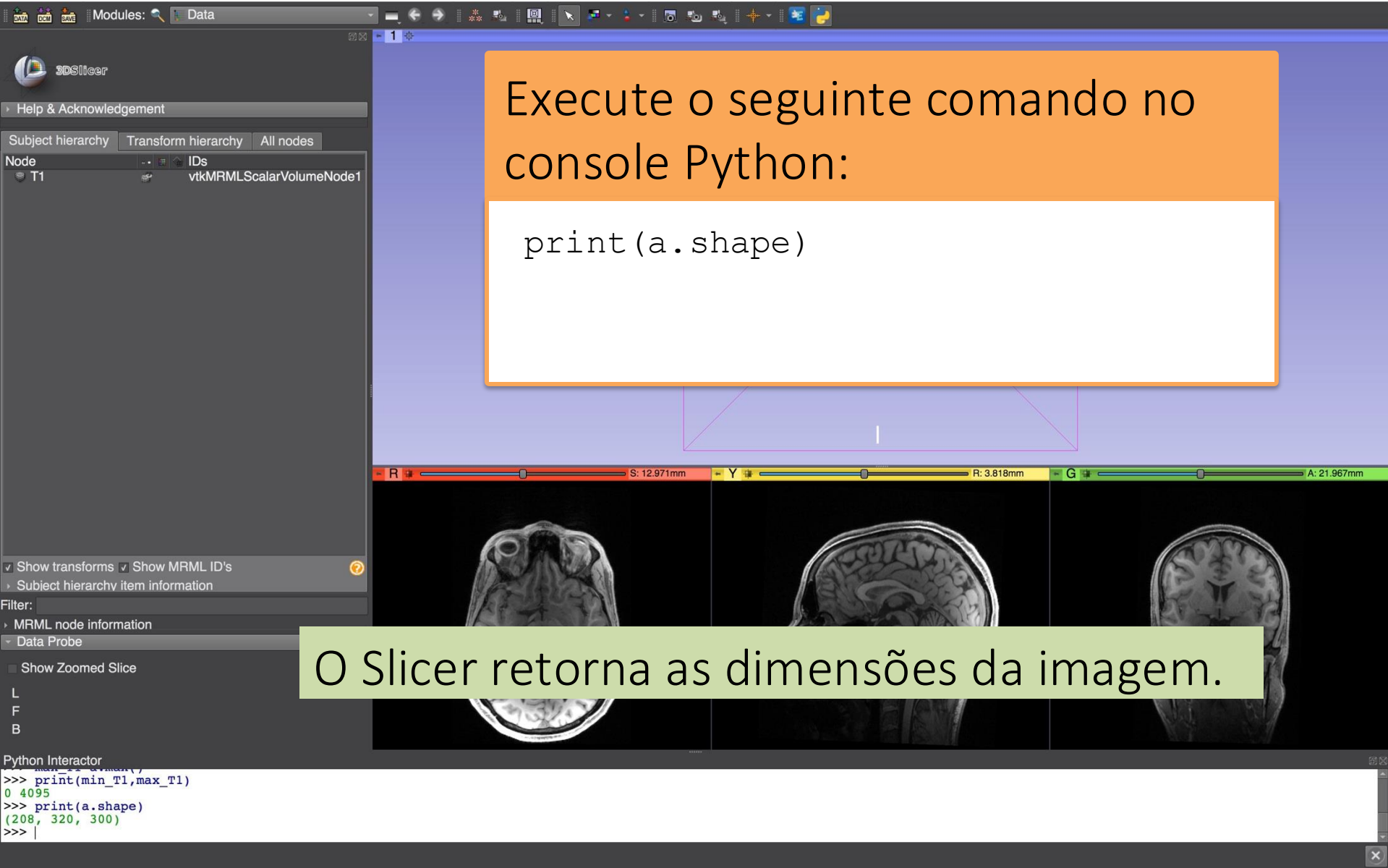
Acessando vóxeis em um volume

The screenshot displays the Slicer software interface. The main window shows a 3D 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 3D view, there are three orthogonal slices: an axial slice on the left, a sagittal slice in the middle, and a coronal slice on the right. The interface includes a sidebar on the left with 'Subject hierarchy' and '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 banner at the bottom of the image contains the text: "O Slicer retorna os valores mín. e máx. da imagem T1."

Modificando vóxeis em um volume



The image shows the 3D Slicer software interface. On the left, there is a sidebar with a 'Subject hierarchy' panel showing a tree view with 'Node' and 'T1' (vtkMRMLScalarVolumeNode1). Below it are checkboxes for 'Show transforms', 'Show MRML ID's', and 'Subject hierarchy item information'. At the bottom left is a 'Python Interactor' window with a terminal-like interface. The main window displays three orthogonal MRI slices of a brain (axial, sagittal, and coronal) with a purple wireframe box overlaid on the axial slice. Above the slices, there are sliders for 'S: 12.971mm', 'R: 3.818mm', and 'A: 21.967mm'. A large orange box in the center contains the text 'Execute o seguinte comando no console Python:' followed by a white box containing the code `print(a.shape)`. A green box at the bottom contains the text 'O Slicer retorna as dimensões da imagem.'

Execute o seguinte comando no console Python:

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

O Slicer retorna as dimensões da imagem.

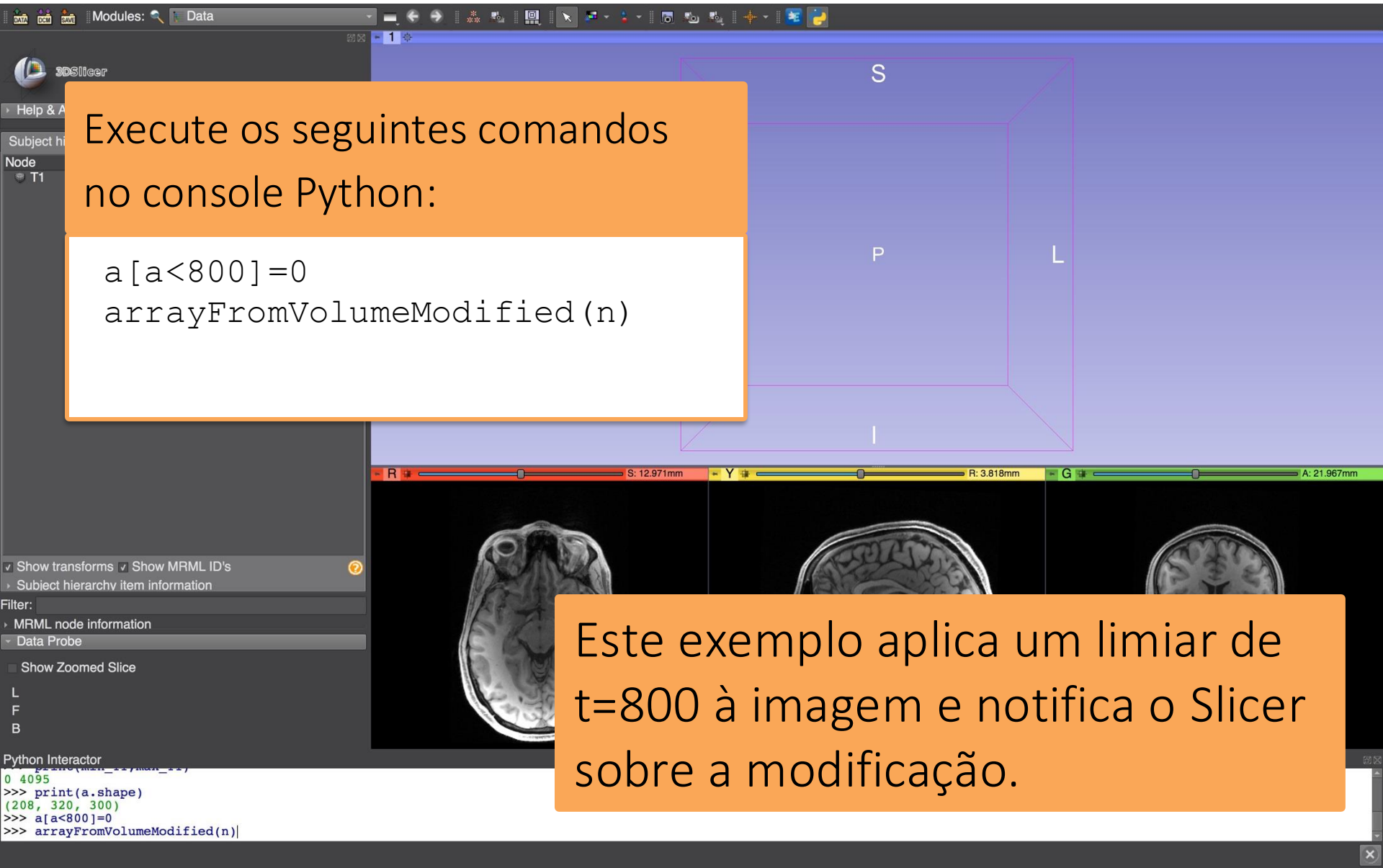
```
>>> print(min_T1,max_T1)
0 4095
>>> print(a.shape)
(208, 320, 300)
>>>
```

Modificando vóxeis em um volume

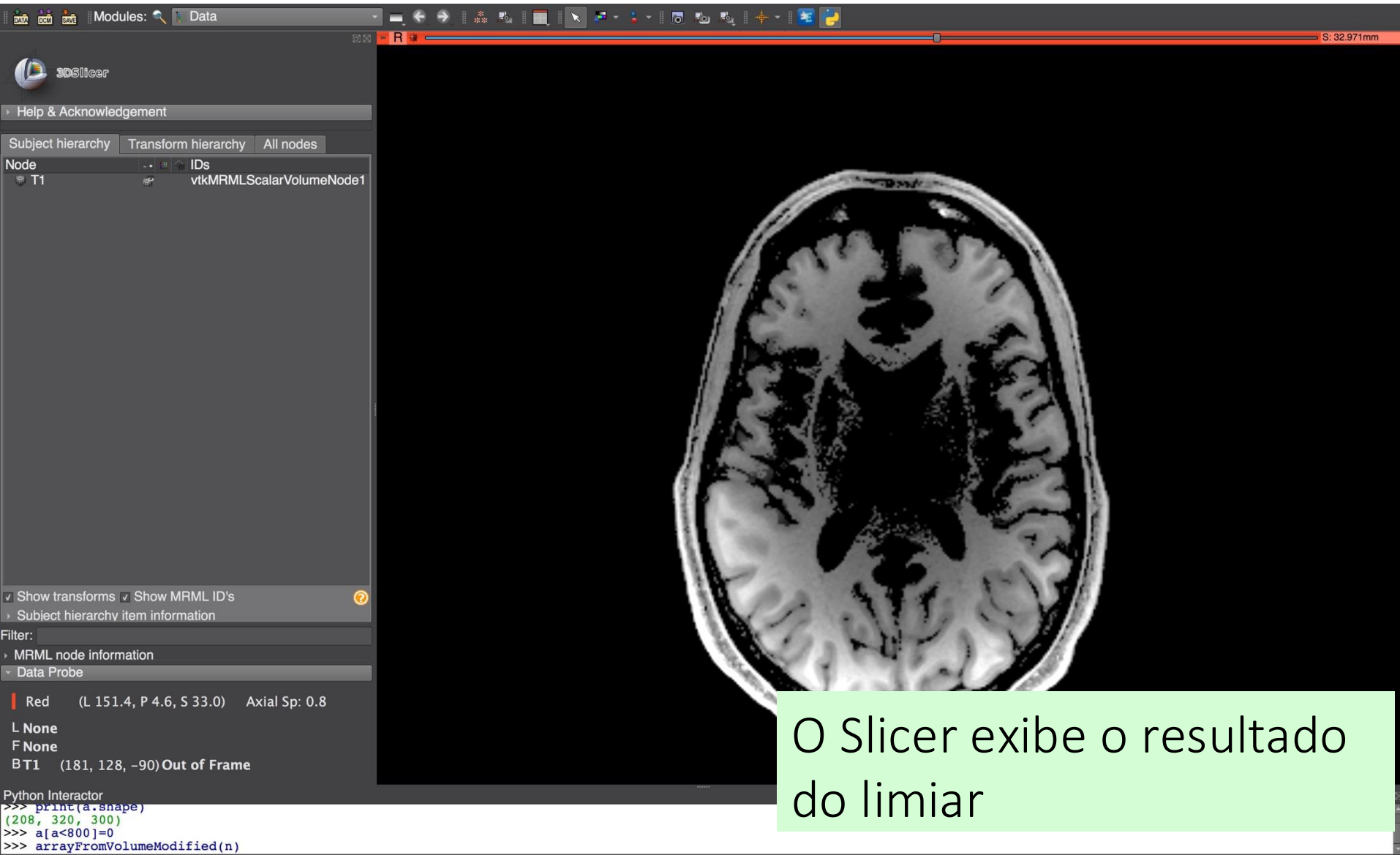
Execute os seguintes comandos no console Python:

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

Este exemplo aplica um limiar de $t=800$ à imagem e notifica o Slicer sobre a modificação.



Modificando vóxeis em um volume



The screenshot displays the 3D Slicer software interface. The main window shows an axial MRI slice of a brain with a threshold filter applied, resulting in a binary image where the brain tissue is white and the background is black. The sidebar on the left contains a 'Subject hierarchy' panel with a tree view showing a 'T1' node. Below this, there are checkboxes for 'Show transforms' and 'Show MRML ID's', and a 'Filter:' section with 'MRML node information' and 'Data Probe' expanded. The 'Data Probe' section shows the current slice location: 'Red (L 151.4, P 4.6, S 33.0) Axial Sp: 0.8'. At the bottom, the 'Python Interactor' shows the following code and output:

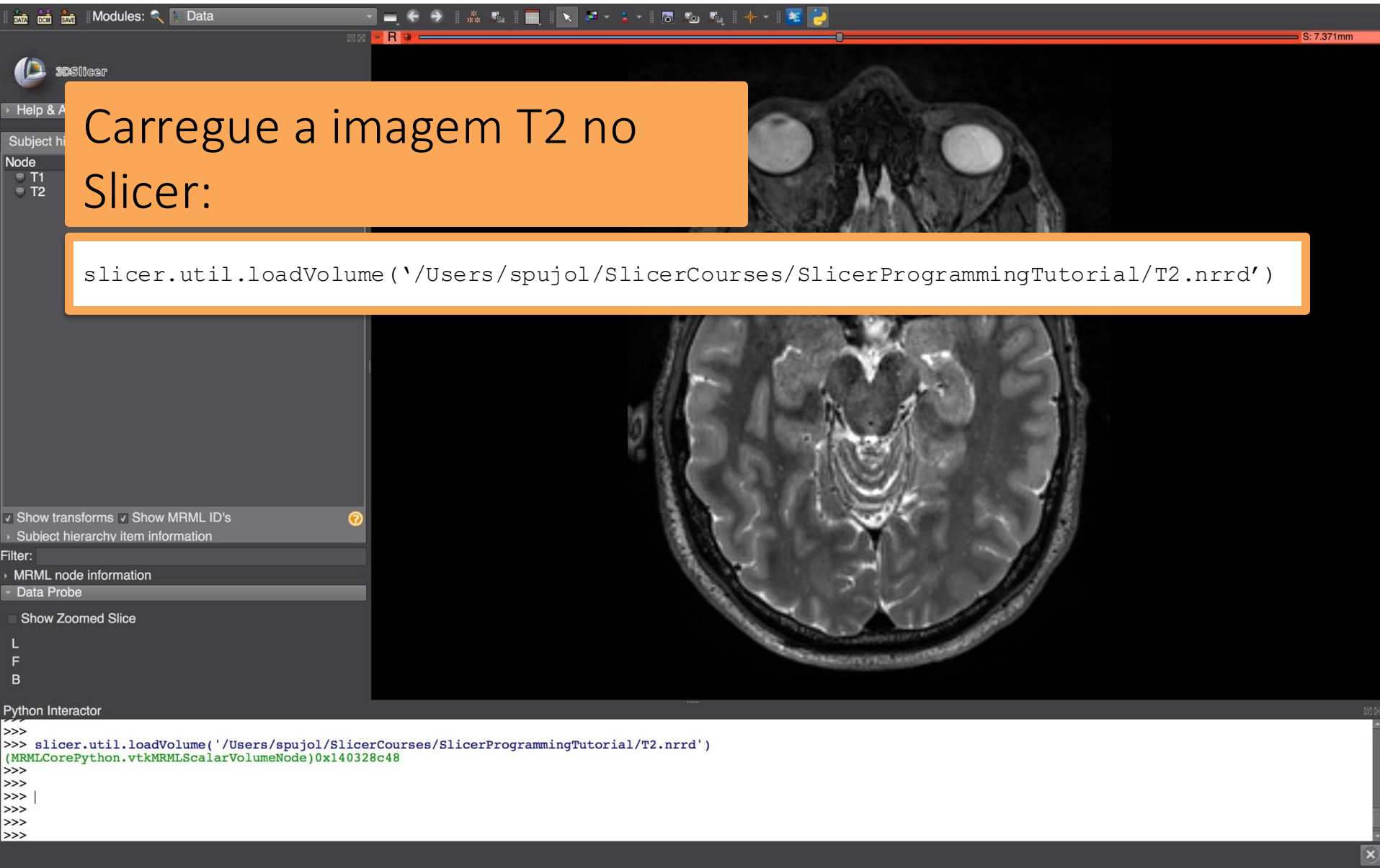
```
>>> print(a.shape)
(208, 320, 300)
>>> a[a<800]=0
>>> arrayFromVolumeModified(n)
```

O Slicer exibe o resultado do limiar

Carregando o volume T2

Carregue a imagem T2 no Slicer:

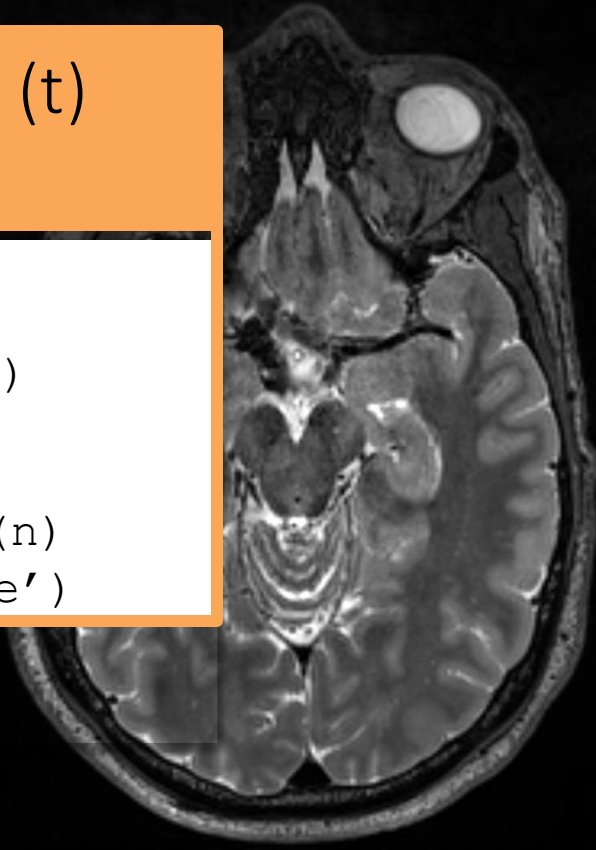
```
slicer.util.loadVolume('/Users/spujol/SlicerCourses/SlicerProgrammingTutorial/T2.nrrd')
```



Função Python: threshold [limiar]

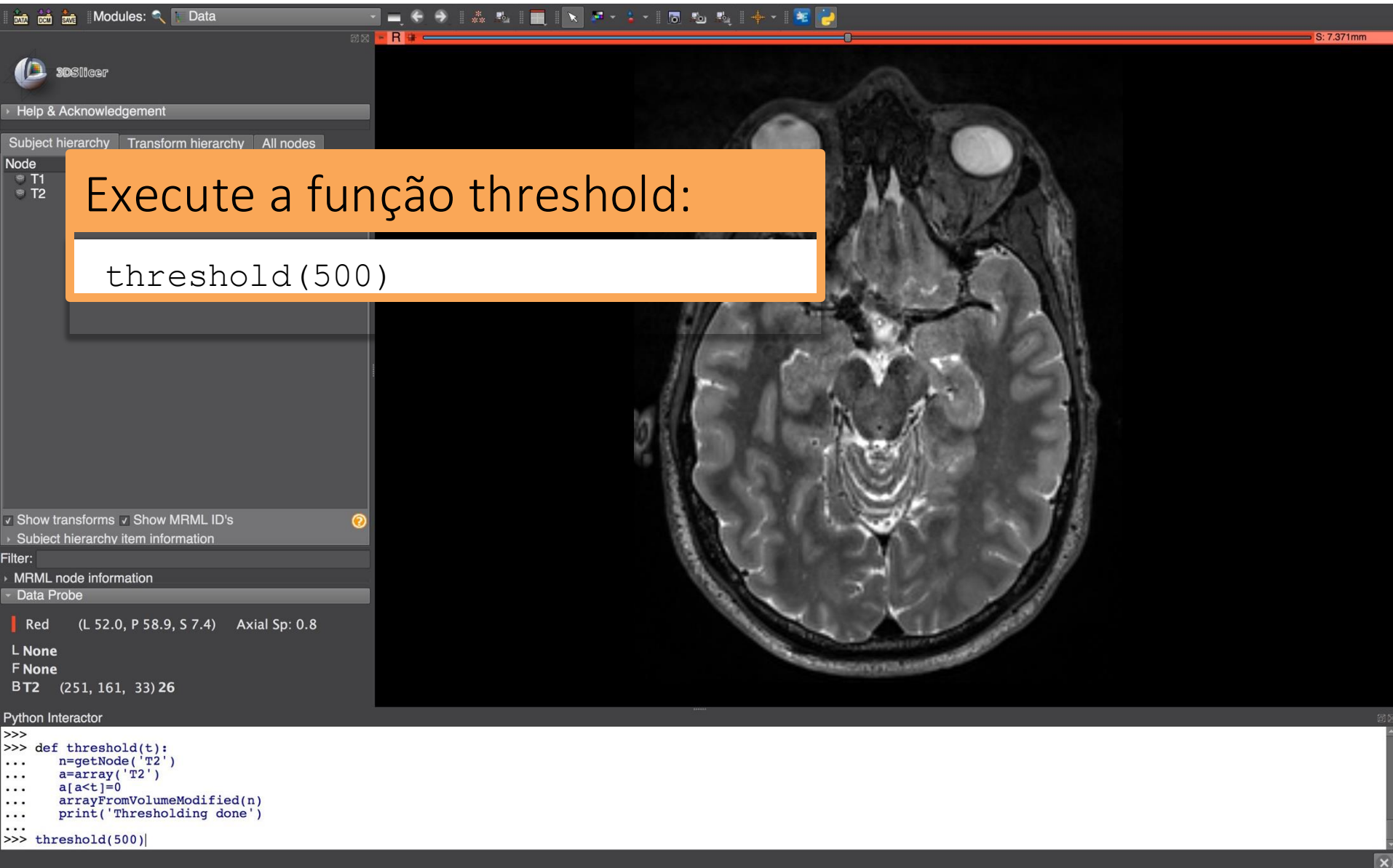
Crie uma função threshold (t)
no Python interactor:

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



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

Função Python: threshold

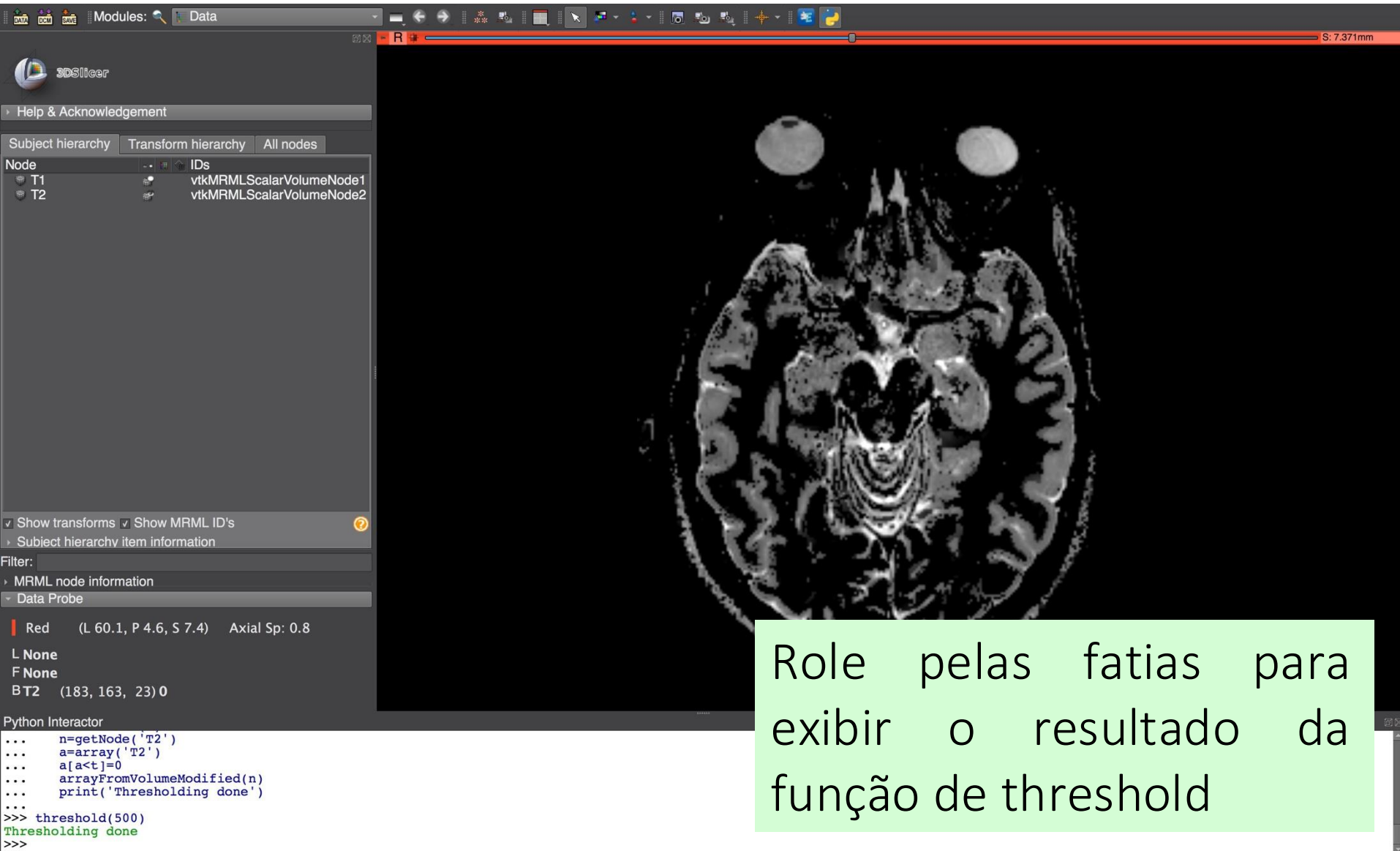


Execute a função threshold:

```
threshold(500)
```

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

Função Python: threshold



The screenshot displays the 3D Slicer software interface. On the left, the 'Subject hierarchy' panel shows two nodes: 'T1' (vtkMRMLScalarVolumeNode1) and 'T2' (vtkMRMLScalarVolumeNode2). Below this, the 'Data Probe' section shows the current slice parameters: 'Red (L 60.1, P 4.6, S 7.4) Axial Sp: 0.8'. At the bottom, the 'Python Interactor' contains the following code:

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

The central view shows an axial MRI slice of a brain. The image is mostly black, with some white and gray structures visible, indicating that a threshold has been applied to the data. A green text box in the bottom right corner contains the text: 'Role pelas fatias para exibir o resultado da função de threshold'.

Visão Geral

- O Slicer oferece fácil acesso para analisar e modificar tipos de dados complexos.
- O Slicer é compatível com uma ampla gama de pacotes de computação científica em Python.
- O Slicer é um ambiente de pesquisa para realizar experimentos de imagens médicas.

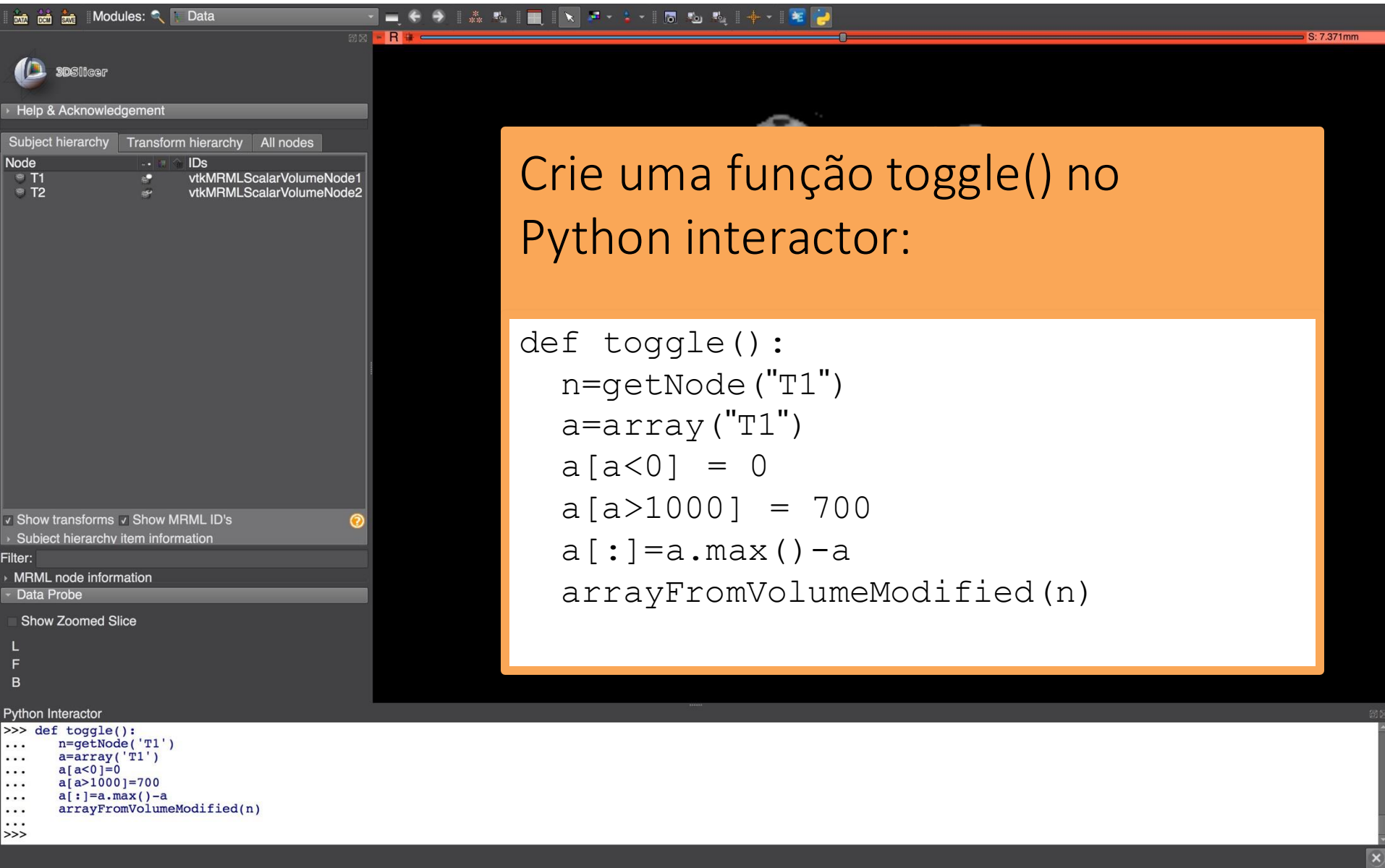
Parte 3

Familiarizando-se com o Qt no Slicer

Qt & PythonQt

- O Qt é a principal ferramenta no Slicer para criar *widgets*, diálogos, campos de texto etc.
- O PythonQt expõe a maioria das funcionalidades do Qt e é acessível por meio do *Python interactor* no Slicer.
- Interfaces de usuário podem ser criadas rapidamente para prototipagem e depuração.

Função Python: toggle [alternar]



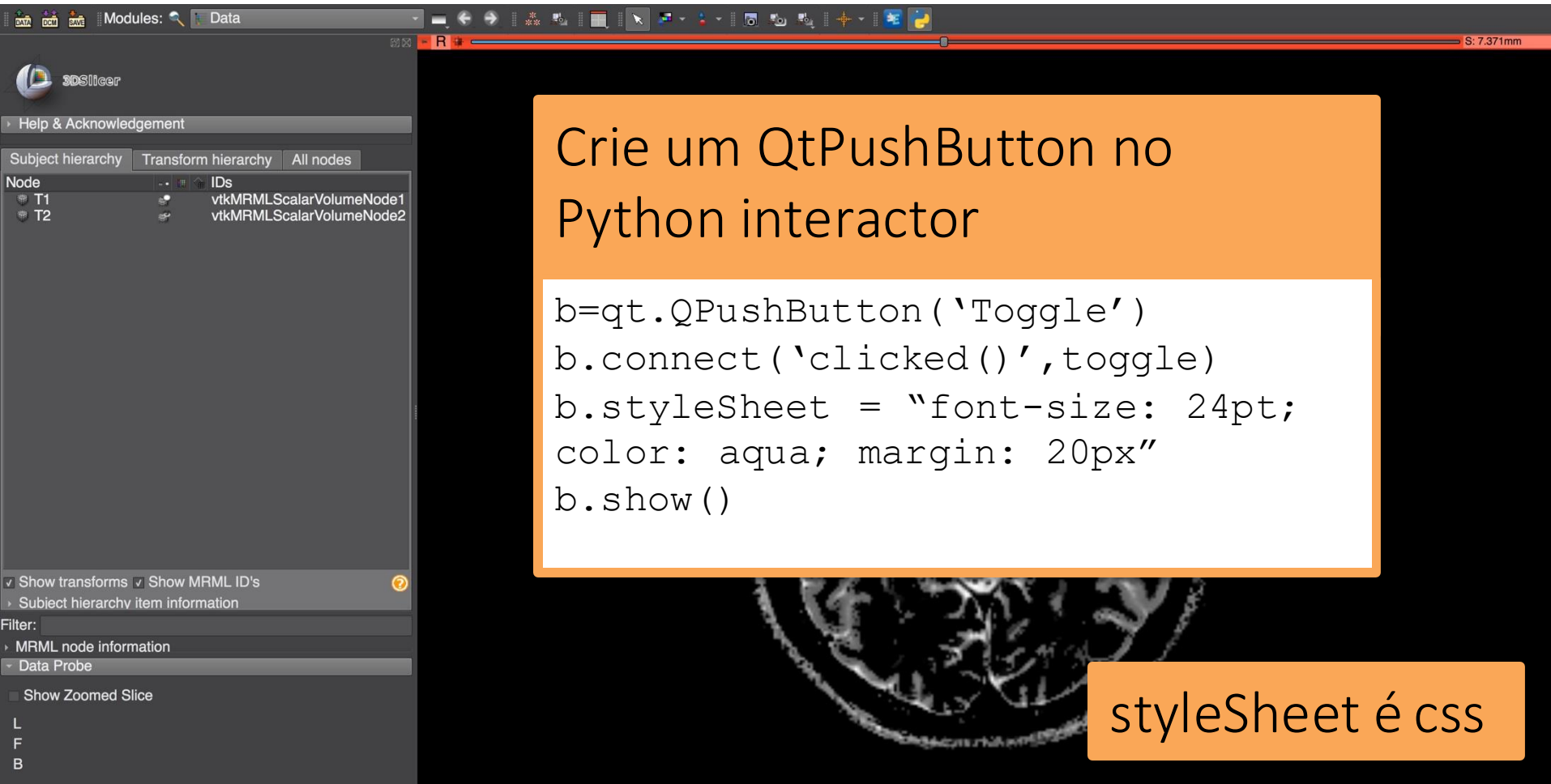
The image shows a screenshot of the 3D Slicer software interface. The main window displays a subject hierarchy with nodes T1 and T2. The Python Interactor window at the bottom shows the following code:

```
>>> 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 `toggle()` that performs the following operations:

- Retrieves the node `T1` using `getNode('T1')`.
- Creates an array `a` from the volume `T1` using `array('T1')`.
- Clips the array to non-negative values: `a[a<0]=0`.
- Clips the array to values less than 700: `a[a>1000]=700`.
- Reverses the array: `a[:]=a.max()-a`.
- Triggers a volume update: `arrayFromVolumeModified(n)`.

Criando um Botão de *Push* do Qt



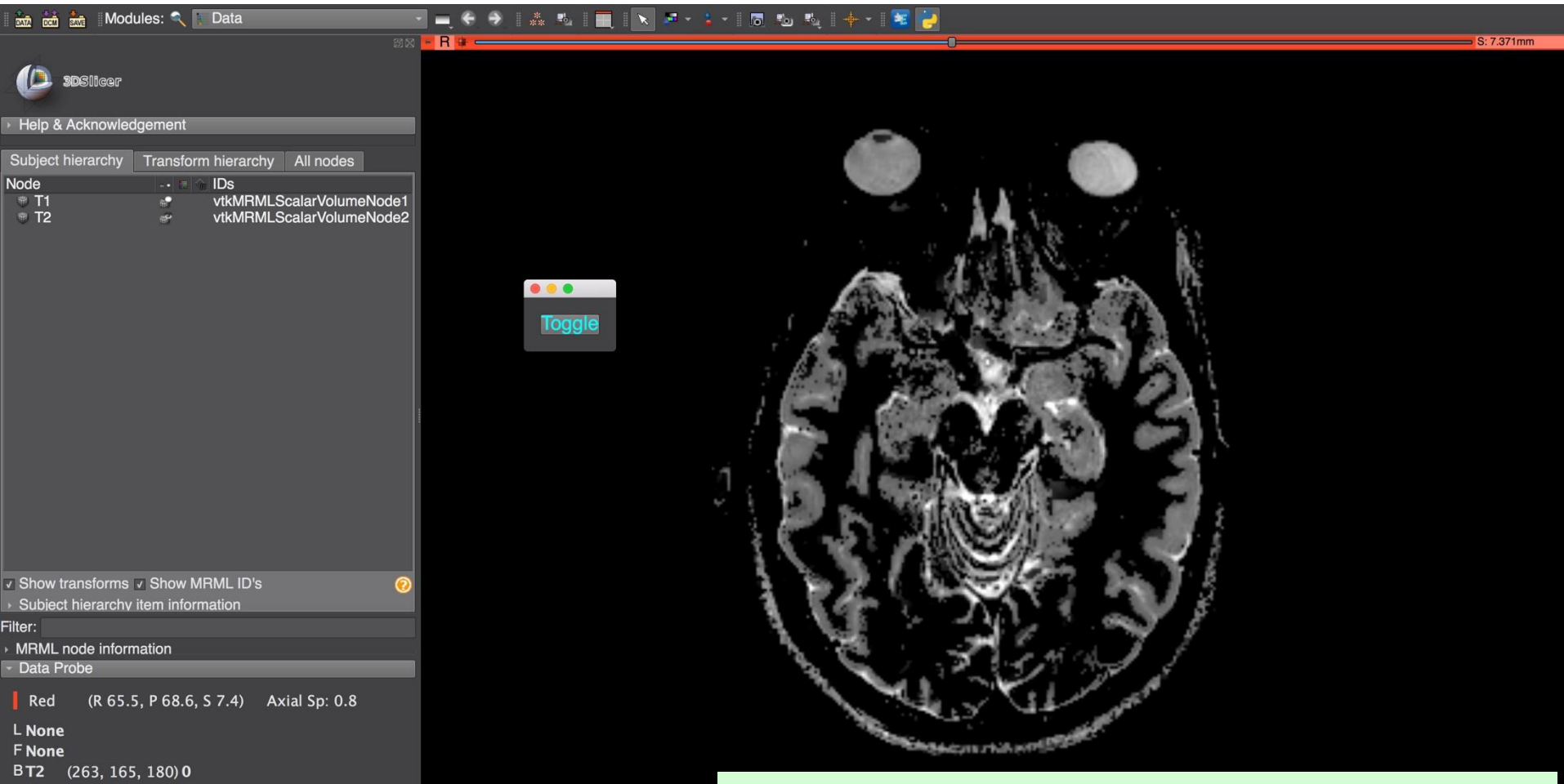
Crie um QPushButton no Python interactor

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

styleSheet é css

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

Criando um Botão de *Push* do Qt

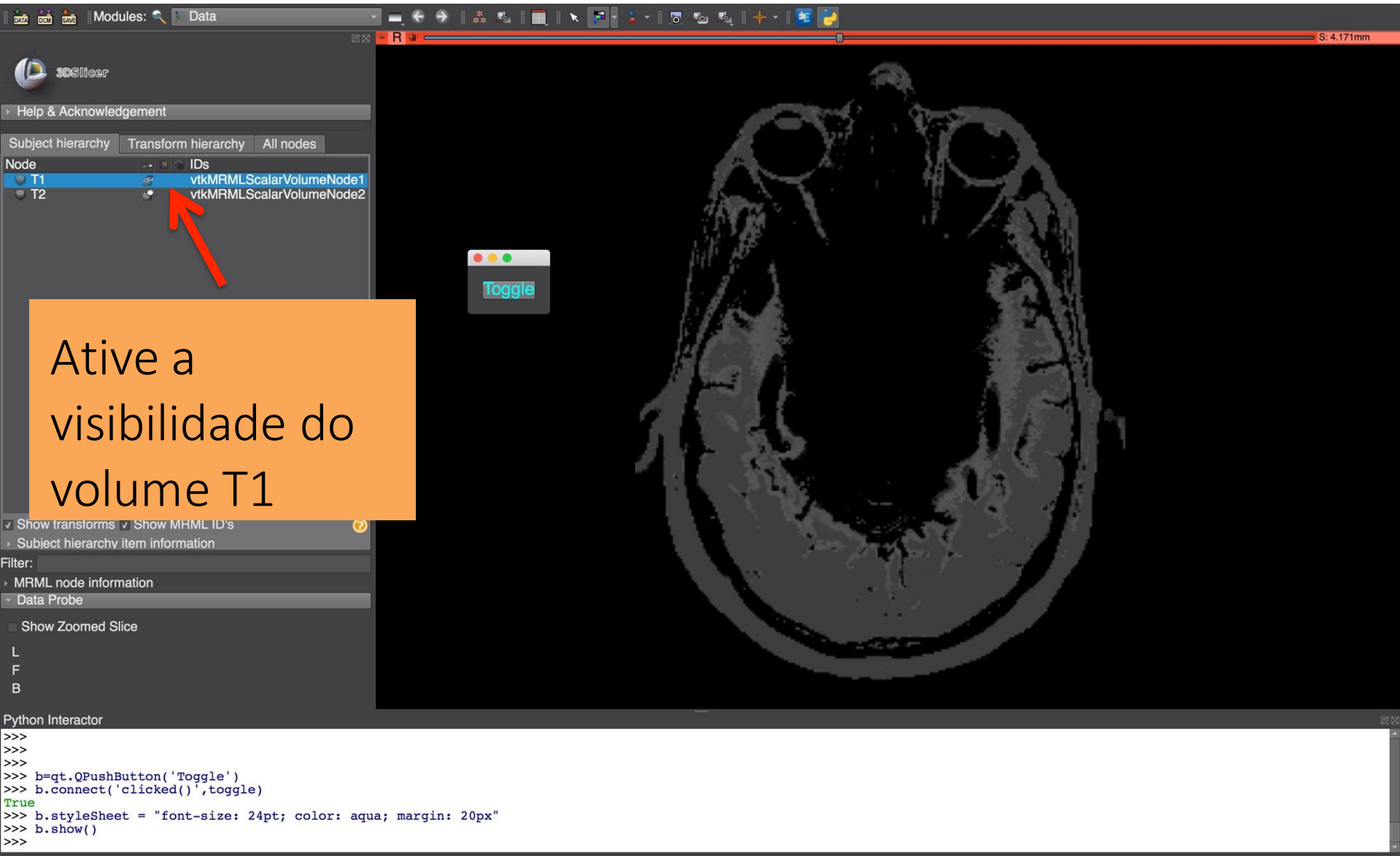


The screenshot shows a Qt application window titled '3DSlicer'. The main area displays an axial MRI scan of a brain. A small window with a title bar and a 'Toggle' button is overlaid on the scan. The left sidebar contains a 'Subject hierarchy' panel with a tree view showing 'Node' containing 'T1' and 'T2'. Below the tree are checkboxes for 'Show transforms' and 'Show MRML ID's', a 'Filter:' field, and a 'Data Probe' section showing 'Red (R 65.5, P 68.6, S 7.4) Axial Sp: 0.8'. At the bottom, a 'Python Interactor' shows the following code:

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

O botão de *toggle* [alternar] aparece

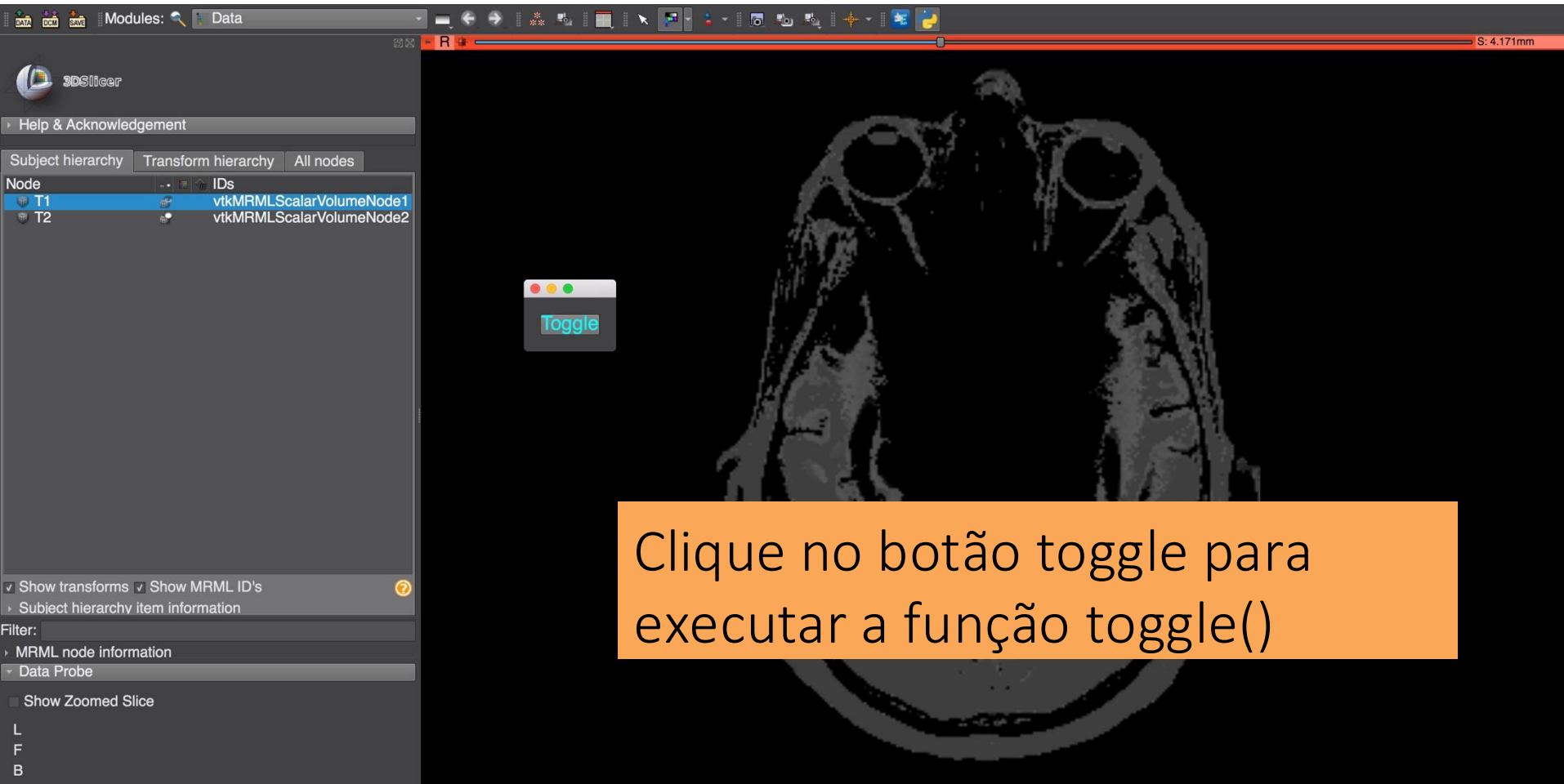
Criando um Botão de *Push* do Qt



Ative a visibilidade do volume T1

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

Criando um Botão de *Push* do Qt



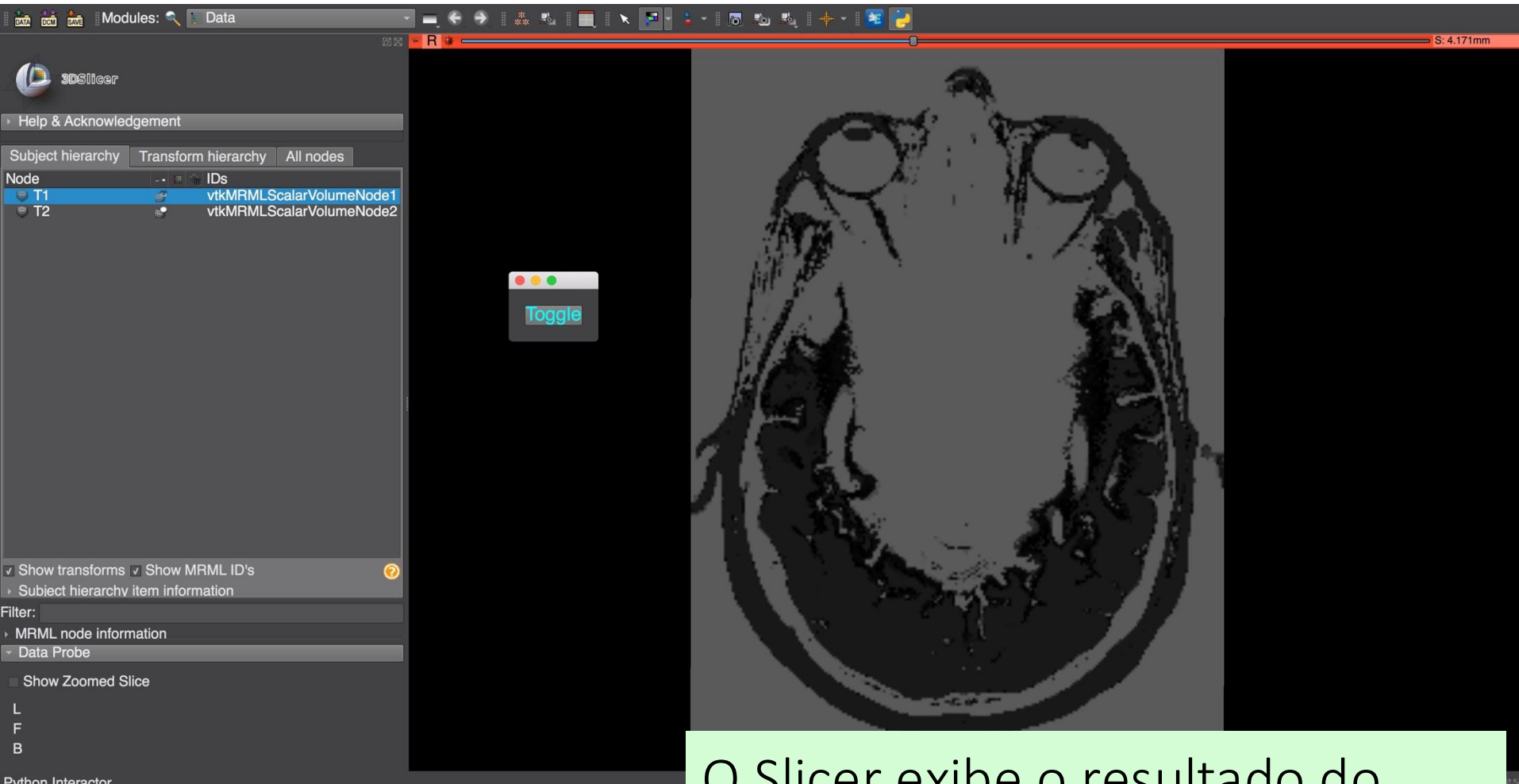
The screenshot shows the 3D Slicer interface. On the left, there is a sidebar with a 'Subject hierarchy' panel containing nodes 'T1' and 'T2'. Below it is a 'Python Interactor' window with the following code:

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

In the center, a 3D view of a brain MRI slice is shown. A small window with a red, yellow, and green title bar and the text 'Toggle' is overlaid on the image.

Clique no botão toggle para executar a função toggle()

Criando um Botão de *Push* do Qt



The screenshot displays the Slicer software interface. On the left, there is a sidebar with a 'Subject hierarchy' panel showing 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 visible, containing a blue button with the text 'Toggle'. The main view shows a grayscale axial MRI slice of a brain. The top status bar indicates 'S: 4.171mm'.

O Slicer exibe o resultado do processamento de imagem.

Exemplos de módulos com *scripts*

- O tutorial demonstra como criar uma interface simples em Python.
- O Slicer integra muitos módulos com *scripts* sofisticados, como Segment Statistics, Sample Data, módulo de Endoscopia etc.
- Para mais informações, consulte o Repositório de Scripts do Slicer:
<https://www.slicer.org/wiki/Documentation/Nightly/ScriptRepository>

Conclusão

- O Slicer permite que os desenvolvedores criem interfaces complexas que são otimizadas para os usuários-alvo.
- A plataforma de software oferece possibilidades ilimitadas de personalização.
- O Slicer dá acesso a bibliotecas avançadas subjacentes por meio de um pacote multiplataforma que é fácil de implementar para os usuários finais.

Agradecimentos



Neuroimage Analysis Center

[Centro de Análise de Neuroimagem]

(NIBIB P41 EB015902)



Sylvain Bouix, Ph.D.

Psychiatry Neuroimaging Laboratory

[Laboratório de Neuroimagem Psiquiátrica]