Paul Cézanne, *Moulin sur la Couleuvre à Pontoise, 1881,*
Staatliche Museen zu Berlin, Nationalgalerie

Programming into Slicer3

Sonia Pujol, Ph.D.

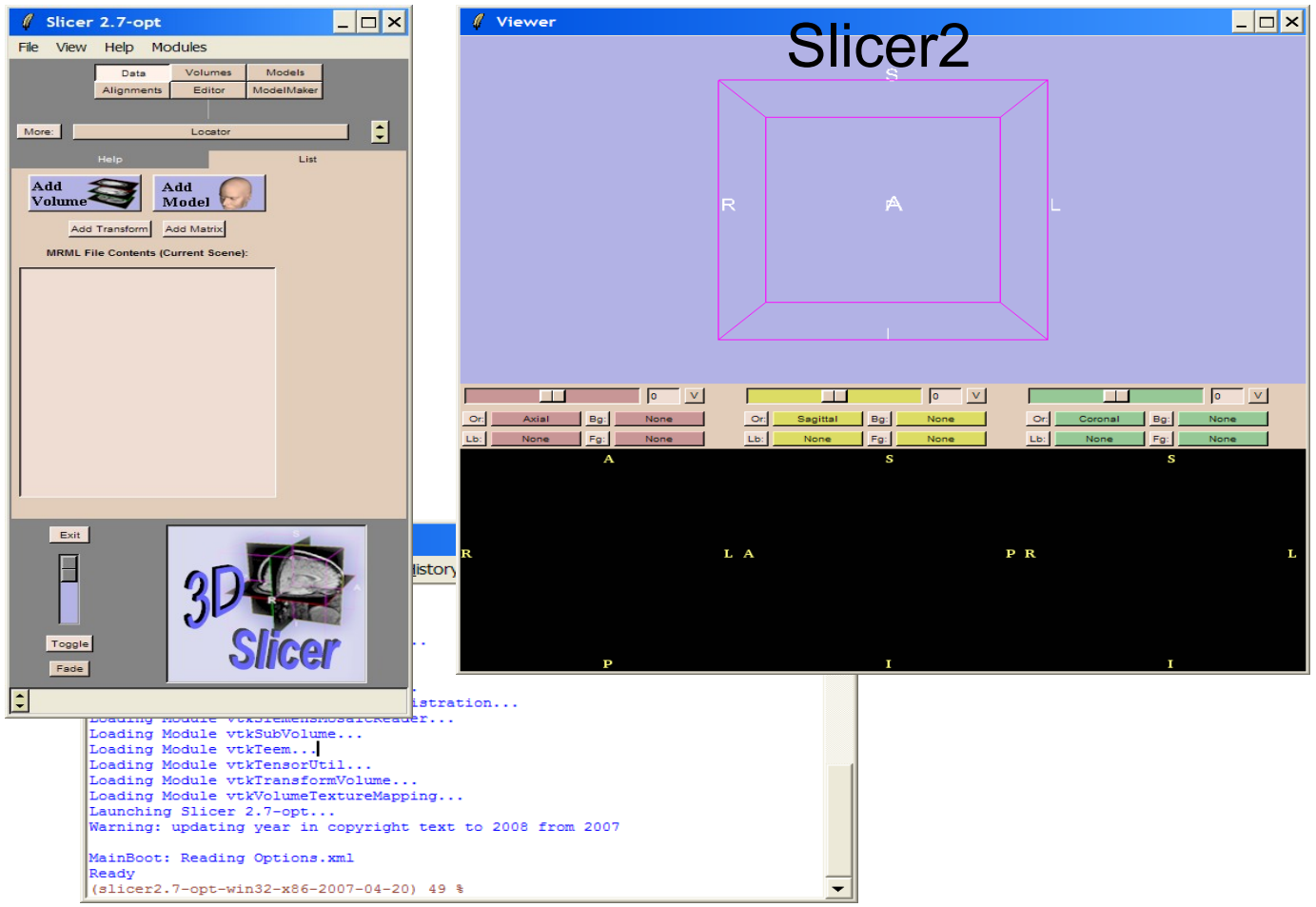
Surgical Planning Laboratory
Harvard University

The NA-MIC Kit

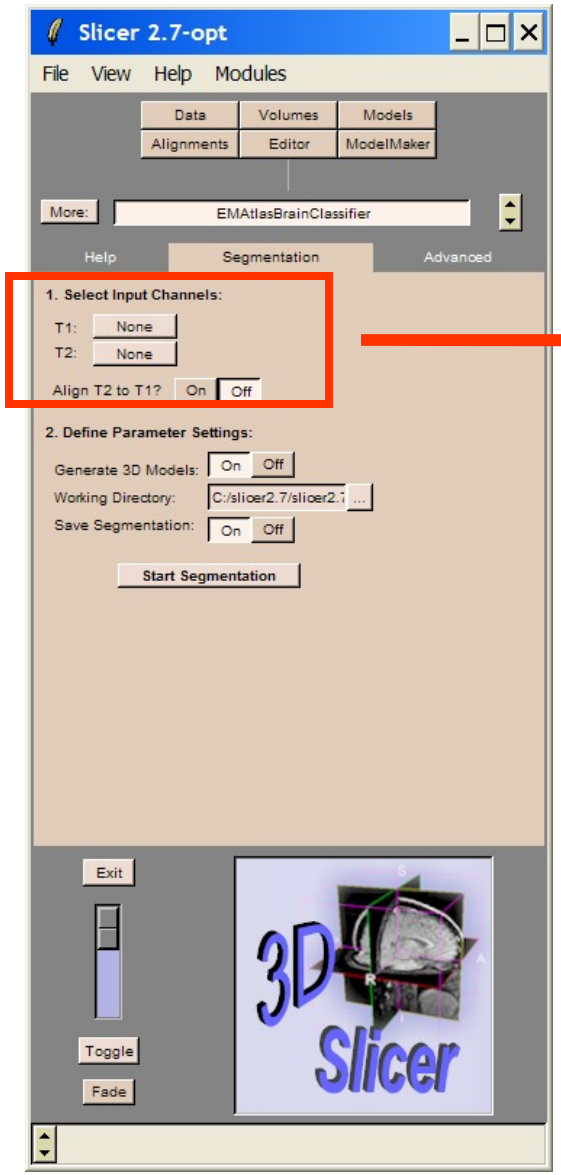
Slicer3

- An end-user application for image analysis
- An open-source environment for software development
- A software platform that is both easy to use for clinical researchers and easy to extend for programmers

Before Slicer3

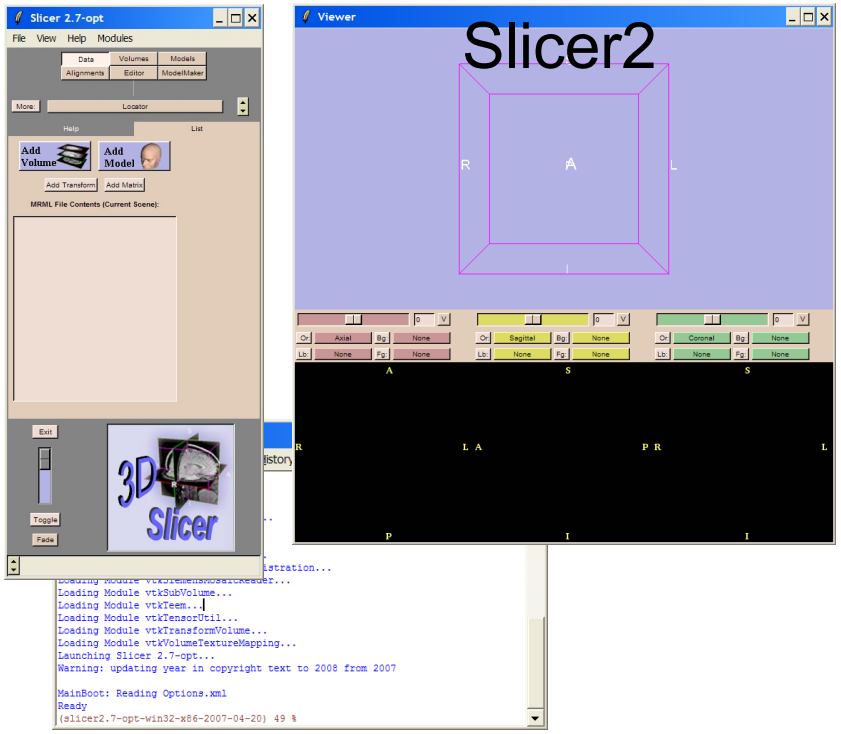


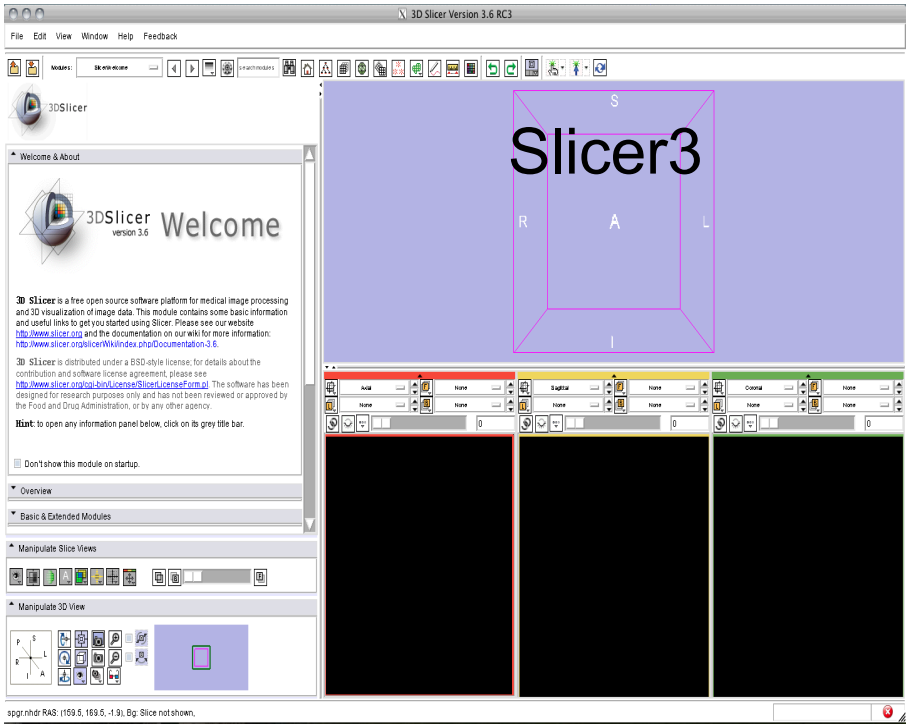
Programming into Slicer2



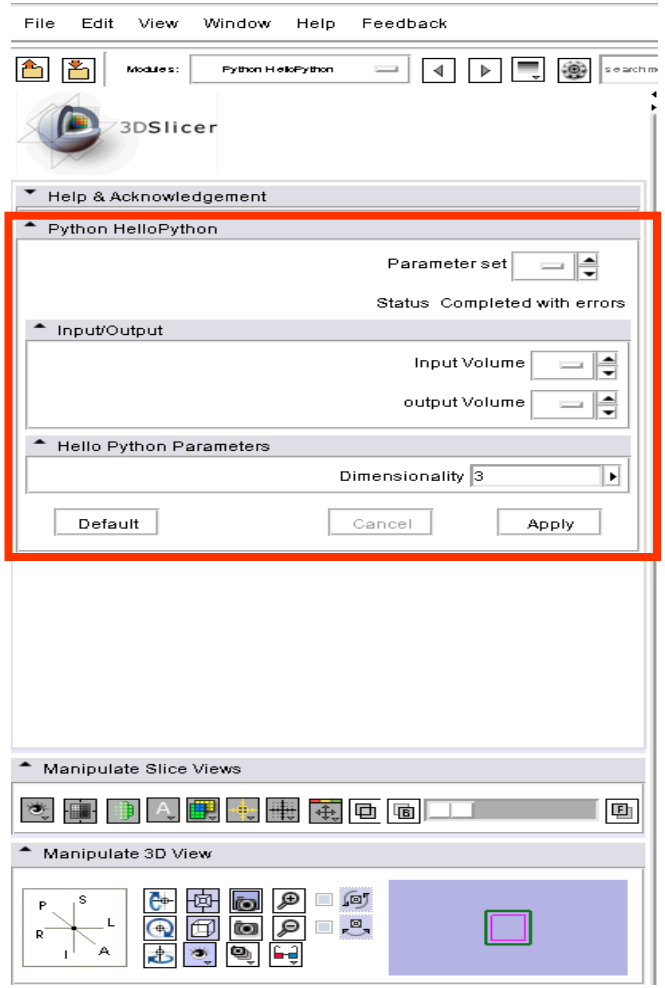

```
#-------------------------------------------
# 1. Step
#-------------------------------------------
set f $fSeg.fStep1
DevAddLabel $f.lTitle "1. Select Input Channels: " WTA
pack $f.lTitle -side top -padx $Gui(pad) -pady 1 -anchor w
frame $f.fInput -bg $Gui(activeWorkspace)
pack $f.fInput -side top -padx 0 -pady 0  -anchor w
foreach frame "Left Right" {
    frame $f.fInput.f$frame -bg $Gui(activeWorkspace)
    pack $f.fInput.f$frame -side left -padx 0 -pady $Gui(pad) }
foreach LABEL "T1 T2" Input "SPGR T2W" {
    DevAddLabel $f.fInput.fLeft.l$Input "  ${LABEL}:"
    pack $f.fInput.fLeft.l$Input -side top -padx $Gui(pad) -pady 1 -anchor w
    set menubutton   $f.fInput.fRight.m${Input}Select
    set menu         $f.fInput.fRight.m${Input}Select.m
            eval {menubutton $menubutton -text [Volume($EMAtlasBrainClassifier(Volume,${Input}),node)
GetName] -relief raised -bd 2 -width 9 -menu $menu} $Gui(WMBA)
    eval {menu $menu} $Gui(WMA)
    TooltipAdd $menubutton "Select Volume defining ${Input}"
    set EMAtlasBrainClassifier(mbSeg-${Input}Select) $menubutton
    set EMAtlasBrainClassifier(mSeg-${Input}Select) $menu
    # Have to update at UpdateMRML too
    DevUpdateNodeSelectButton Volume EMAtlasBrainClassifier Seg-${Input}Select Volume,$Input
    pack $menubutton -side top  -padx $Gui(pad) -pady 1 -anchor w }
frame $f.fAlign -bg $Gui(activeWorkspace)
 TooltipAdd  $f.fAlign "If the input T1 and T2 are not aligned with each other set flag here"
 pack $f.fAlign -side top -padx 0 -pady 2  -padx $Gui(pad) -anchor w
 DevAddLabel $f.fAlign.lAlign "Align T2 to T1? "
 pack $f.fAlign.lAlign -side left -padx $Gui(pad) -pady 1 -anchor w
 foreach value "1 0" text "On Off" width "4 4" {
    eval {radiobutton $f.fAlign.r$value -width $width -indicatoron 0\
            -text "$text" -value "$value" -variable EMAtlasBrainClassifier(AlignInput) } $Gui(WCA)
    pack $f.fAlign.r$value -side left -padx 0 -pady 0  }
```

From Slicer2 to Slicer3





The New Execution Model



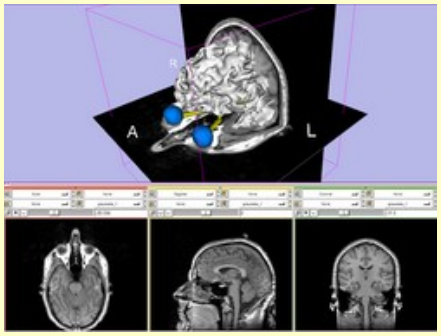

```
<?xml version="1.0" encoding="utf-8"?>
<executable>
 <category> Demonstration </category>
 <title> Python HelloPython </title>
 <description> Slicer Developer Course </description>
 <version>  1.0     </version>
 <documentation-url>  </documentation-url>
 <license></license>
 <contributor>
        Sonia Pujol, Ph.D., Surgical Planning Laboratory, Harvard Medical School
</contributor>
 <acknowledgements> National Alliance for Medical Image Computing (NAMIC), Grant
U54 EB005149. </acknowledgements>
<parameters>
  <label>Input/Output</label>
  <description>Input/output parameters</description>
  <image>
   <name>helloPython</name>
   <label>Input Volume</label>
   <channel>input</channel>
   <index>0</index>
   <default>None</default>
   <description>Input volume</description>
  </image>
  <image>
   <name>helloPythonOutputVolume</name>
   <label>Output Volume</label>
   <channel>output</channel>
   <index>1</index>
   <default>None</default>
   <description>Output filtered</description>
  </image>
</parameters>
</executable>
```

Slicer3 Execution Model

- This course is based on the Execution Model which provides a mechanism for incorporating command line programs as Slicer modules.
- Jim Miller, Dan Blezek, Bill Lorensen (GE)
- This course uses the Python interpreter that has been integrated to Slicer.

Pre-requisite

- This course supposes that you have taken the following tutorial:



Slicer3 Data Loading and Visualization, *Sonia Pujol,*

- The tutorial is available on the Slicer3.6 101 compendium:

http://www.slicer.org/slicerWiki/index.php/Slicer3.6:Training#Software_tutorials

Material

This course requires the following material

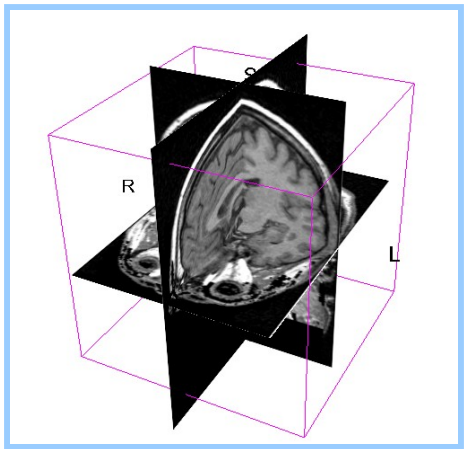
- Slicer3.6
- HelloPython.zip

Disclaimer

It is the responsibility of the user of 3DSlicer to comply with both the terms of the license and with the applicable laws, regulations and rules.

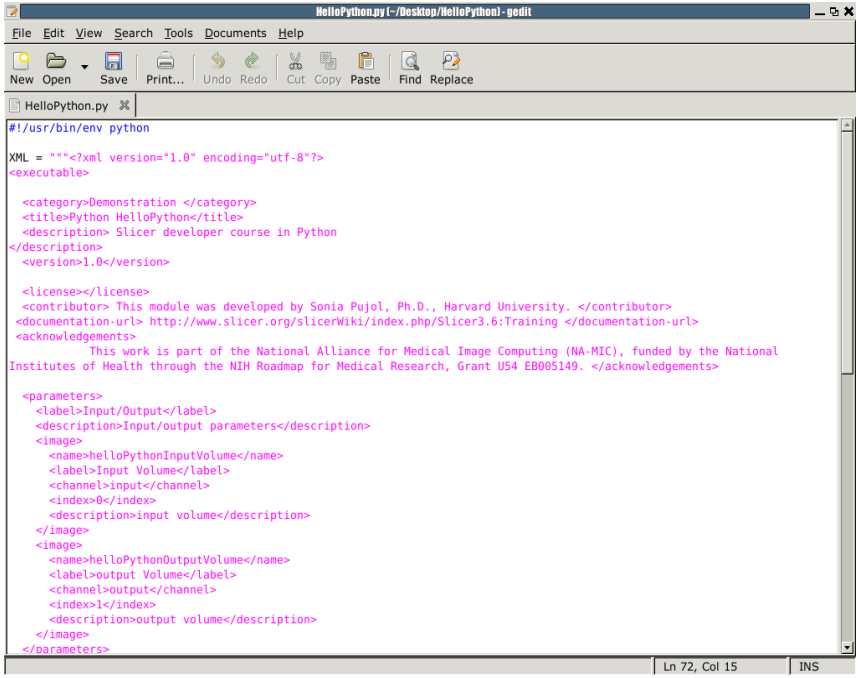
HelloPython Course Material

Unzip the HelloPython.zip archive



spgr.nhdr spgr.raw.gz

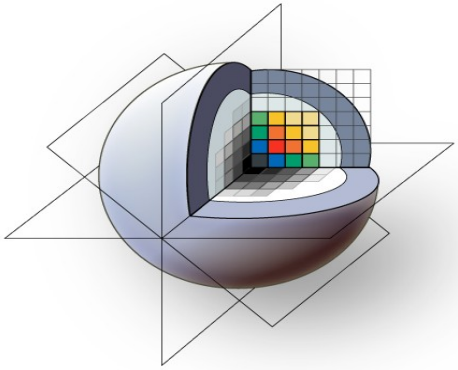
(124 SPGR images)



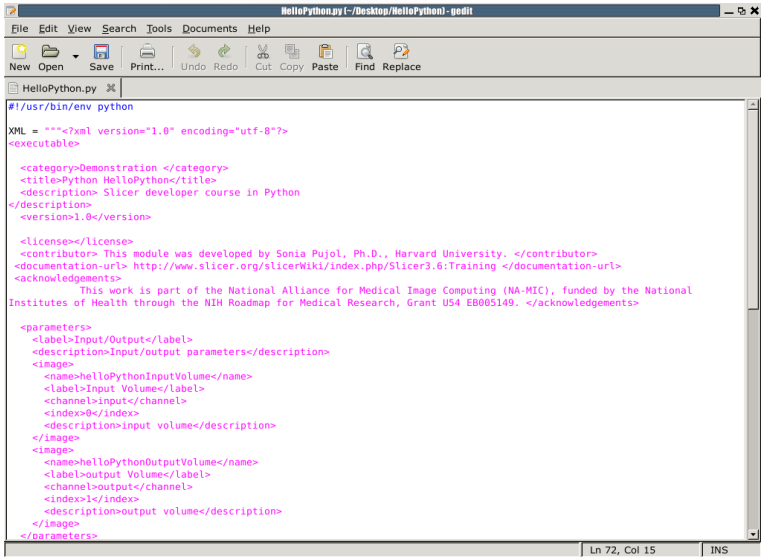
HelloPython.py

Overview

- Part A: Integration of the HelloPython.py program into Slicer3
- Part B: Implementation of the Laplace operator in the Python HelloPython module
- Part C: Image Sharpening using the Laplace operator

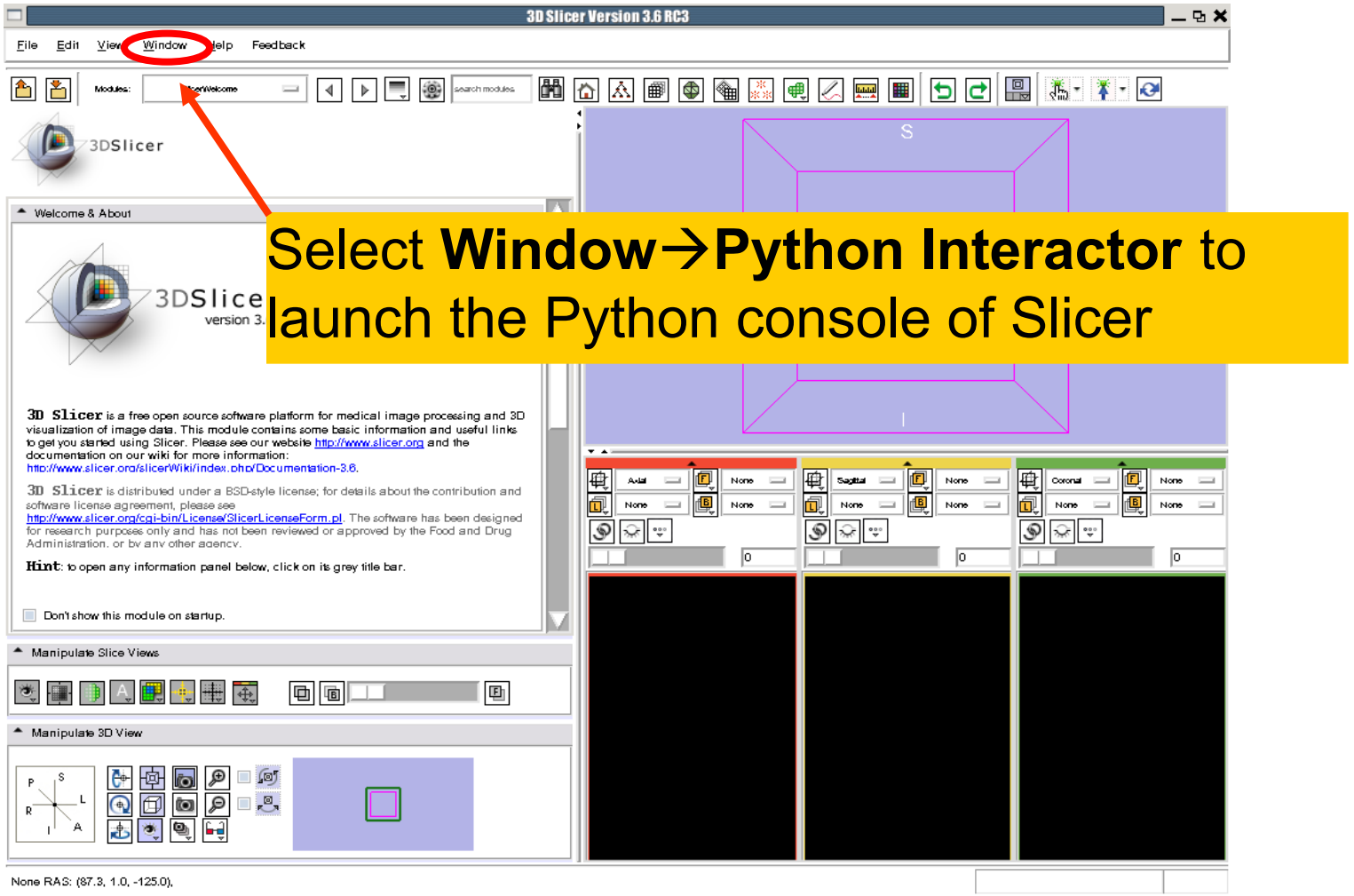


3DSlicer

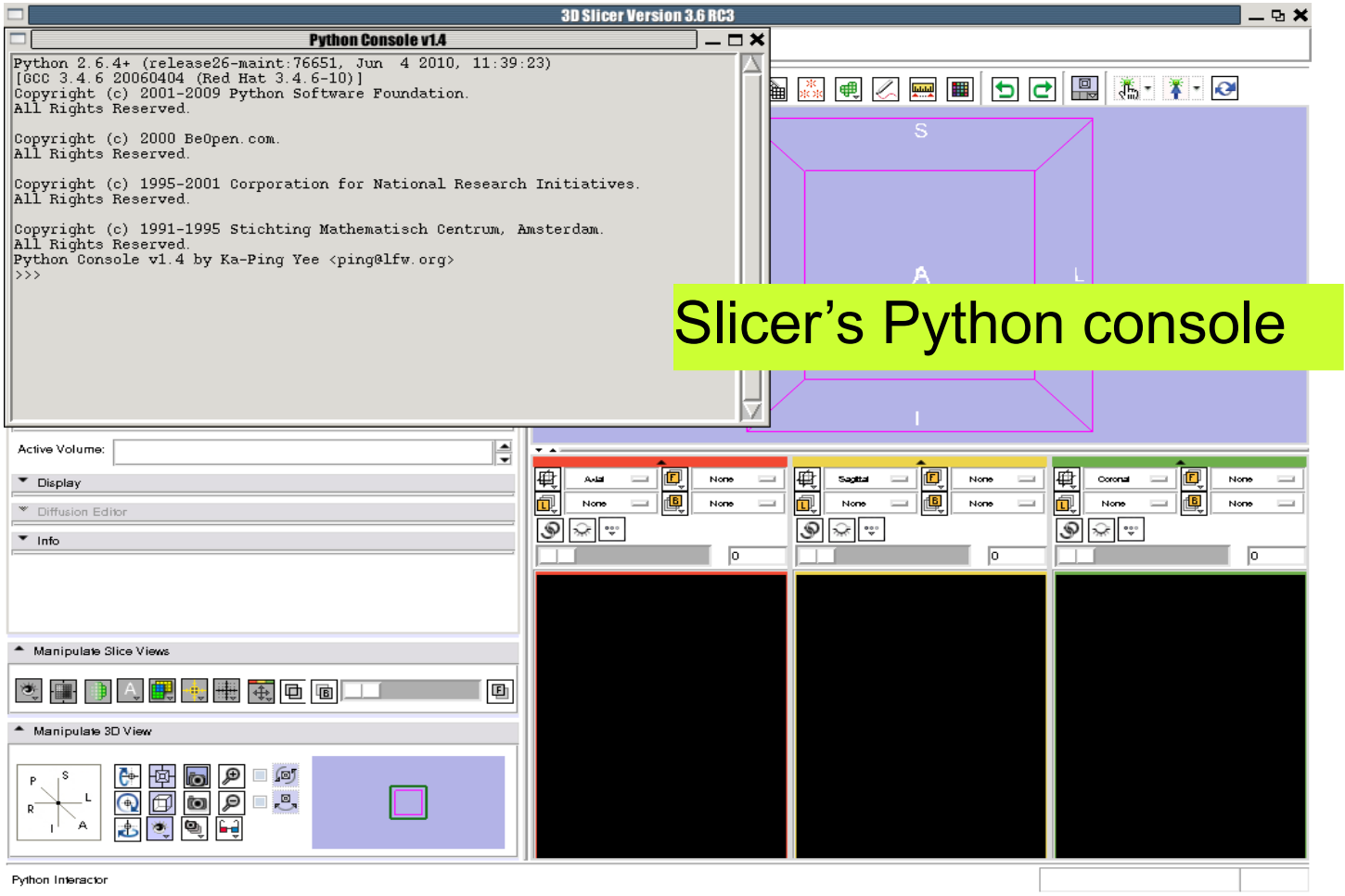


Part A: Integrating HelloPython into Slicer3

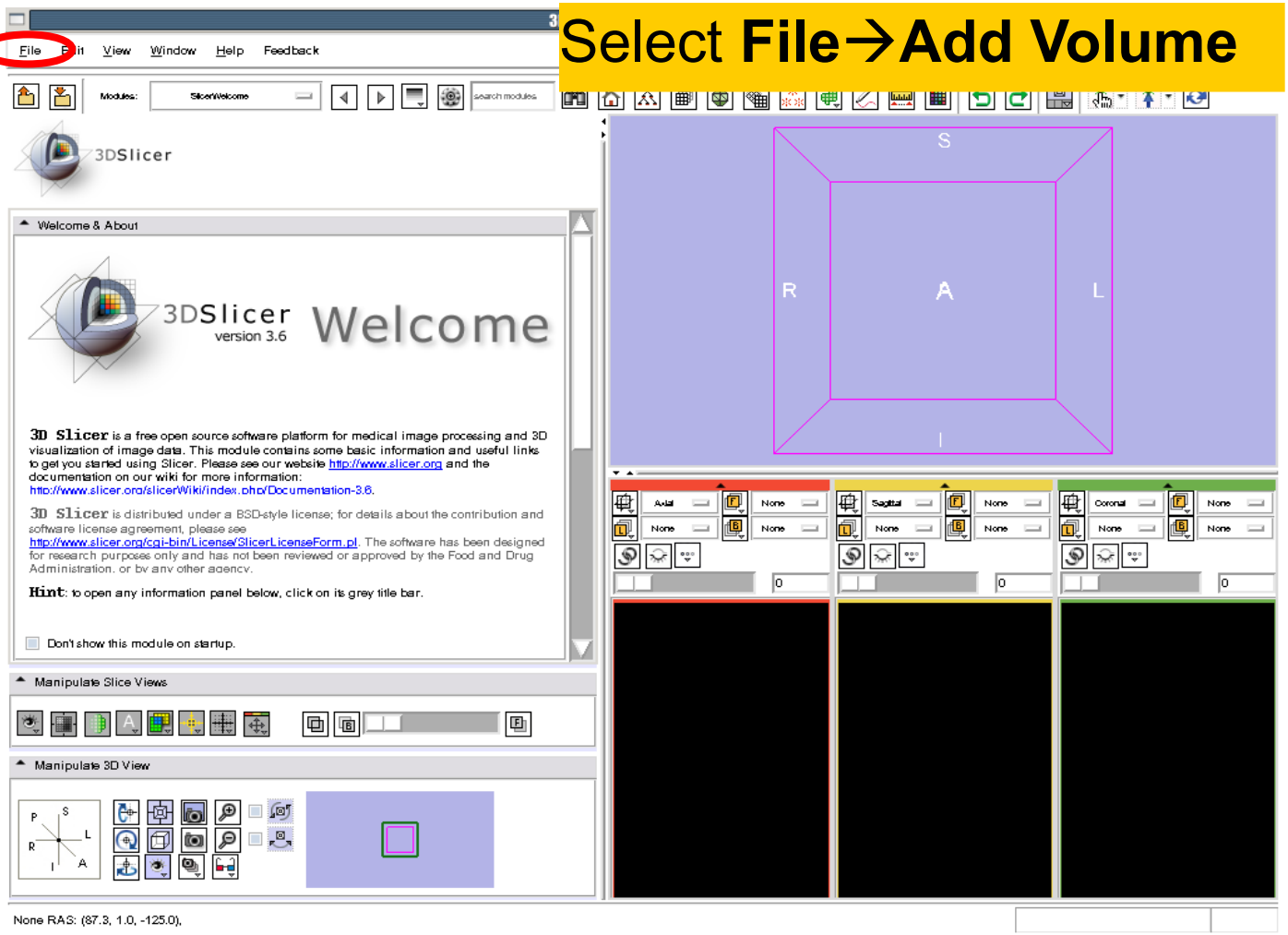
Python Console



Python Console

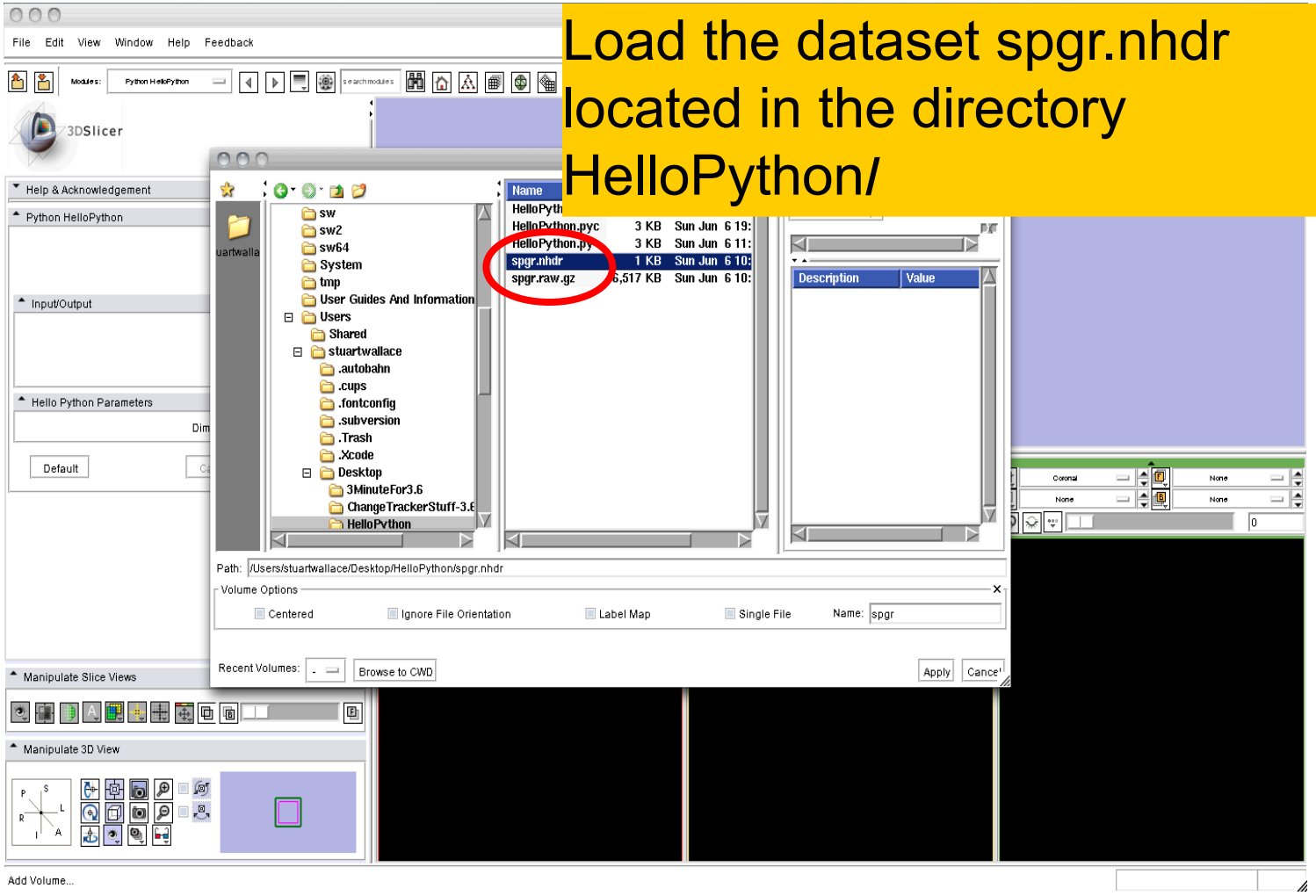


Python Console

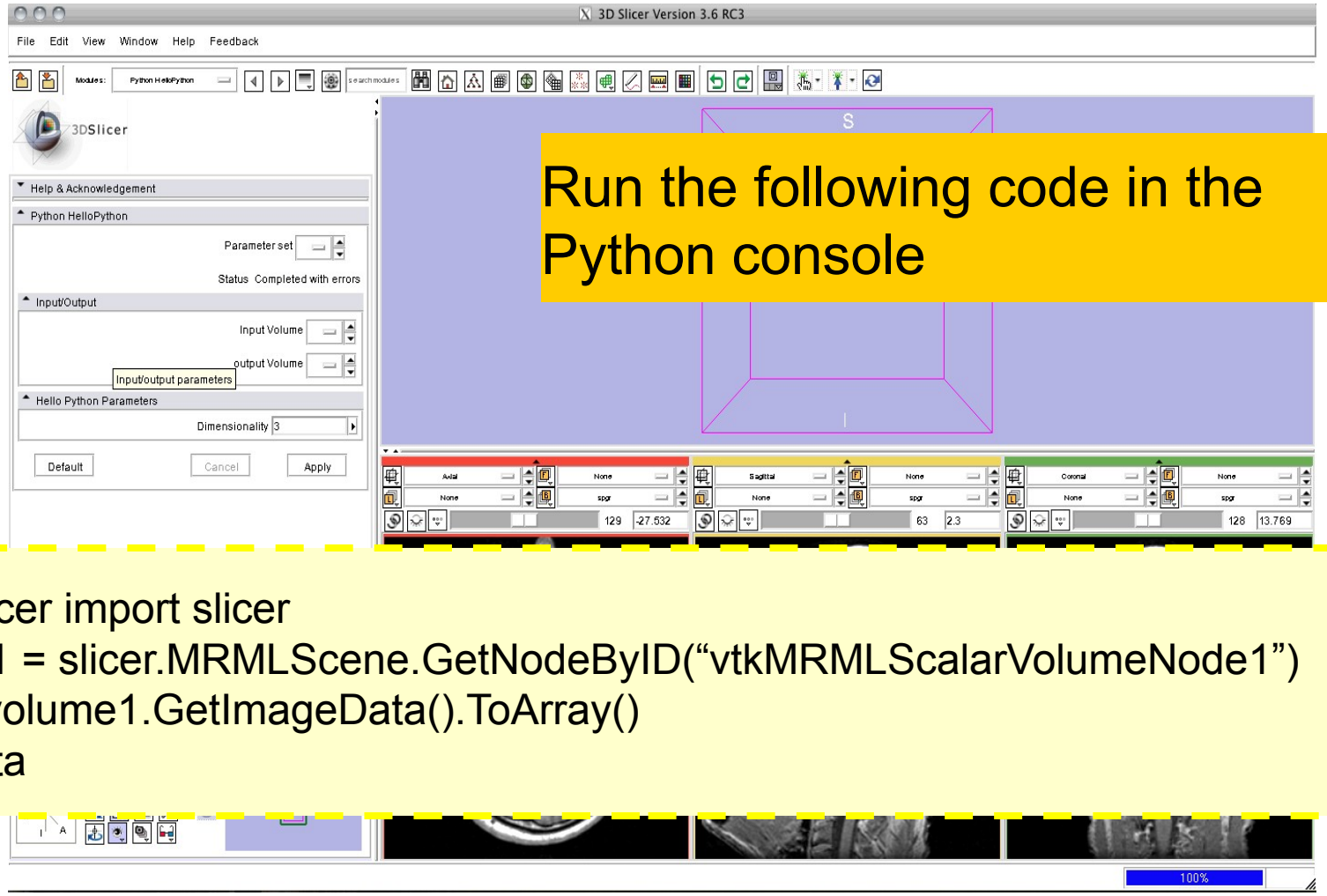


Python Console

Load the dataset spgr.nhdr located in the directory HelloPython/

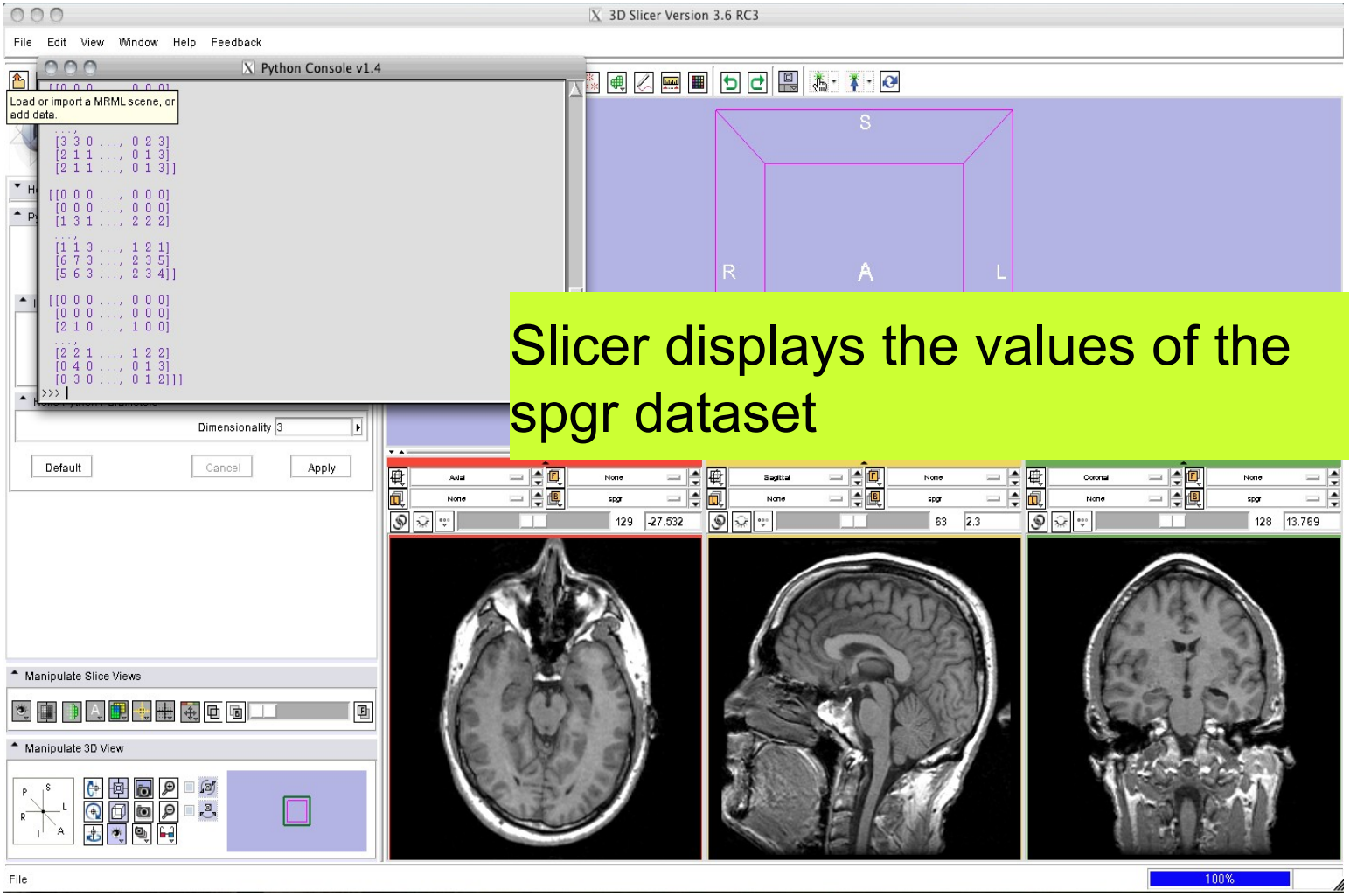


Python Console



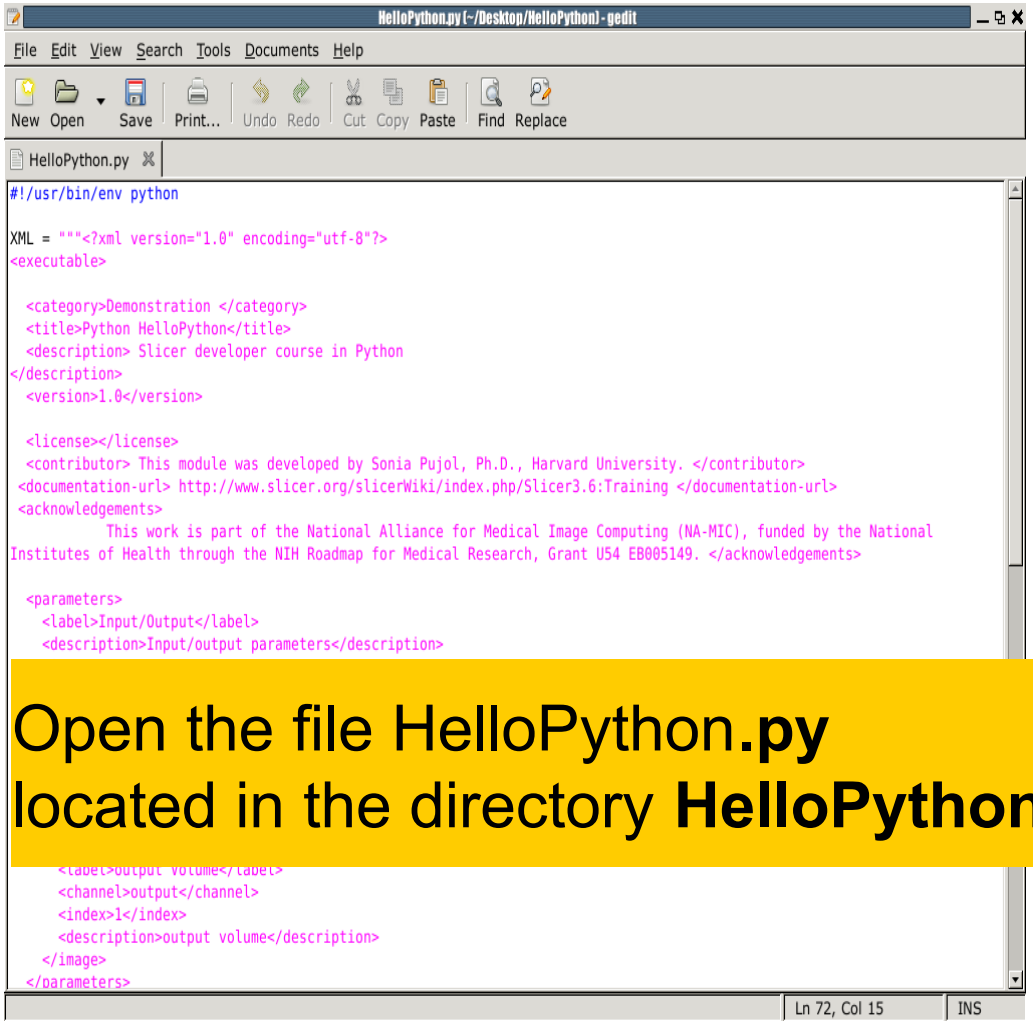

```
from Slicer import slicer
volume1 = slicer.MRMLScene.GetNodeByID("vtkMRMLScalarVolumeNode1")
data = volume1.GetImageData().ToArray()
print data
```

Python Console



Slicer displays the values of the spgr dataset

HelloPython.py



Open the file HelloPython**.py**
located in the directory **HelloPython**

HelloPython.py

Module Description

Module Parameters

Execute function

HelloPython.py

```
#!/usr/bin/env python

XML = """<?xml version="1.0" encoding="utf-8"?>
<executable>

  <category>Demonstration </category>
  <title>Python HelloPython</title>
  <description> Slicer developer course in Python
</description>
  <version>1.0</version>

  <license></license>
  <contributor> This module was developed by Sonia Pujol, Ph.D., Harvard University. </contributor>
 <documentation-url> http://www.slicer.org/slicerWiki/index.php/Slicer3.6:Training </documentation-url>
 <acknowledgements>
          This work is part of the National Alliance for Medical Image Computing (NA-MIC), funded by the National Institutes of
Health through the NIH Roadmap for Medical Research, Grant U54 EB005149. </acknowledgements>

  <parameters>
    <label>Input/Output</label>
    <description>Input/output parameters</description>
    <image>
      <name>helloPythonInputVolume</name>
      <label>Input Volume</label>
      <channel>input</channel>
      <index>0</index>
      <description>input volume</description>
    </image>
    <image>
      <name>helloPythonOutputVolume</name>
      <label>output Volume</label>
      <channel>output</channel>
      <index>1</index>
      <description>output volume</description>
    </image>
  </parameters>
<parameters>
    <label>Hello Python Parameters</label>
    <description>Parameters of the Python Hello Python module </description>
     <integer>
      <name>dimensionality</name>
      <longflag>dimensionality</longflag>
      <description>Dimensionality of the Laplace operator</description>
      <label>Dimensionality</label>
      <default>3</default>
      <constraints>
        <minimum>2</minimum>
        <maximum>3</maximum>
      </constraints>
    </integer>
</parameters>


</executable>
"""


def Execute ():
        Slicer = __import__("Slicer")
        slicer = Slicer.slicer
        scene = slicer.MRMLScene

        return
```

Module Description

```
#!/usr/bin/env python
XML = """<?xml version="1.0" encoding="utf-8"?>
<executable>
<category>Demonstration </category>
 <title>Python HelloPython</title>
<description> Slicer developer course in Python </description>
 <version>1.0</version>
 <license></license>
 <contributor> This module was developed by Sonia Pujol, Ph.D., Harvard University. </contributor>
<documentation-url> http://www.slicer.org/slicerWiki/index.php/Slicer3.6:Training </documentation-url>
<acknowledgements>
        This work is part of the National Alliance for Medical Image Computing (NA-MIC),
        funded by the National Institutes of Health through the NIH Roadmap for Medical Research,
        Grant U54 EB005149.
</acknowledgements>
```

Module Parameters

```
<parameters>
  <label>Input/Output</label>
  <description>Input/output parameters</description>
  <image>
    <name>HelloPythonInputVolume</name>
    <label>Input Volume</label>
    <channel>input</channel>
    <index>0</index>
    <description>input volume</description>
  </image>

    <image>
     <name>HelloPythonOutputVolume</name>
     <label>Output Volume</label>
     <channel>output</channel>
     <index>1</index>
     <description>output volume</description>
    </image>
</parameters>
```

Input Volume

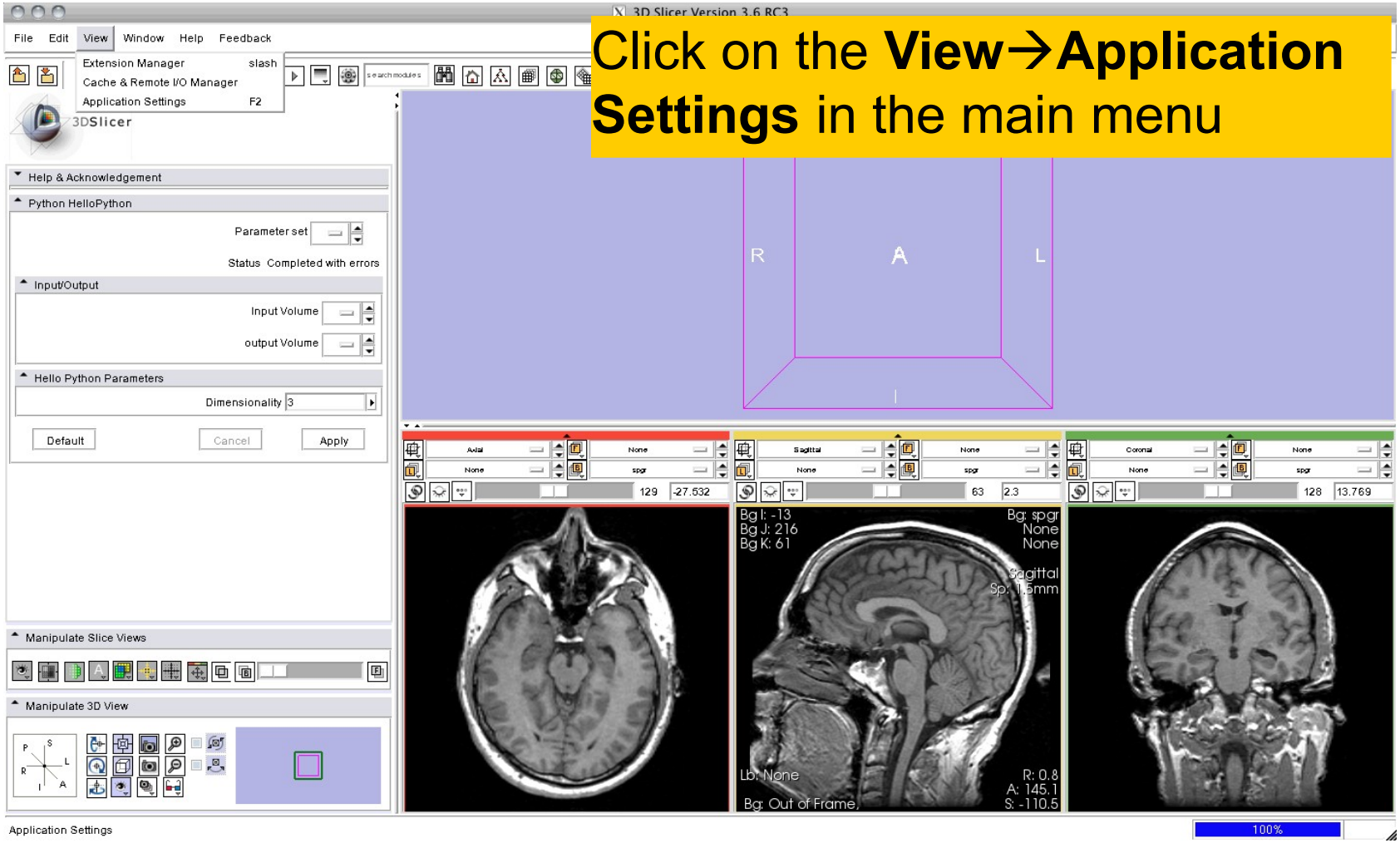
Output Volume

A file that specifies the image

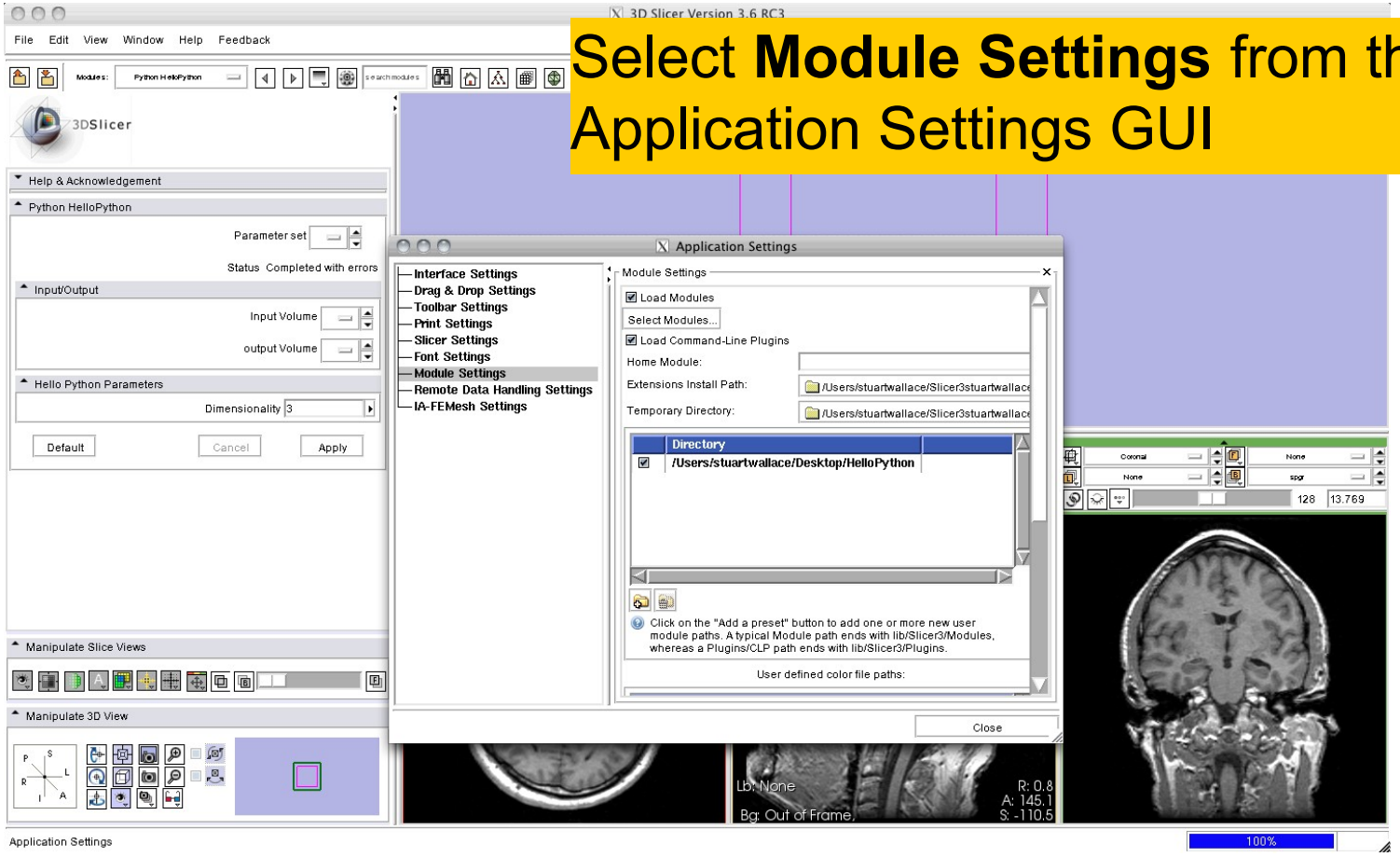
Execute Function

```
def Execute ():

    Slicer = __import__("Slicer")
    slicer = Slicer.slicer
    scene = slicer.MRMLScene

    return
```

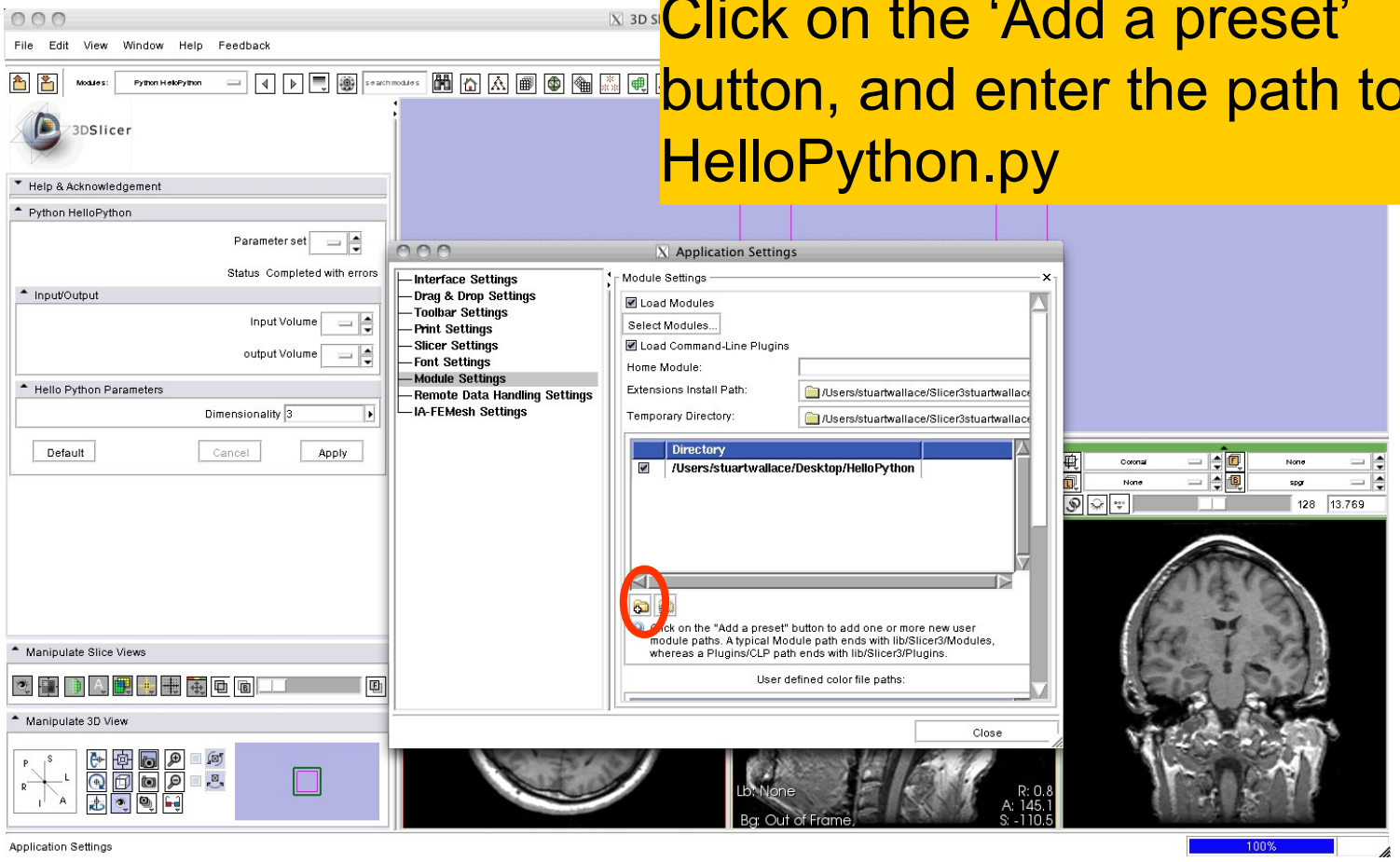
Integrating HelloPython to Slicer3



Integrating HelloPython to Slicer3

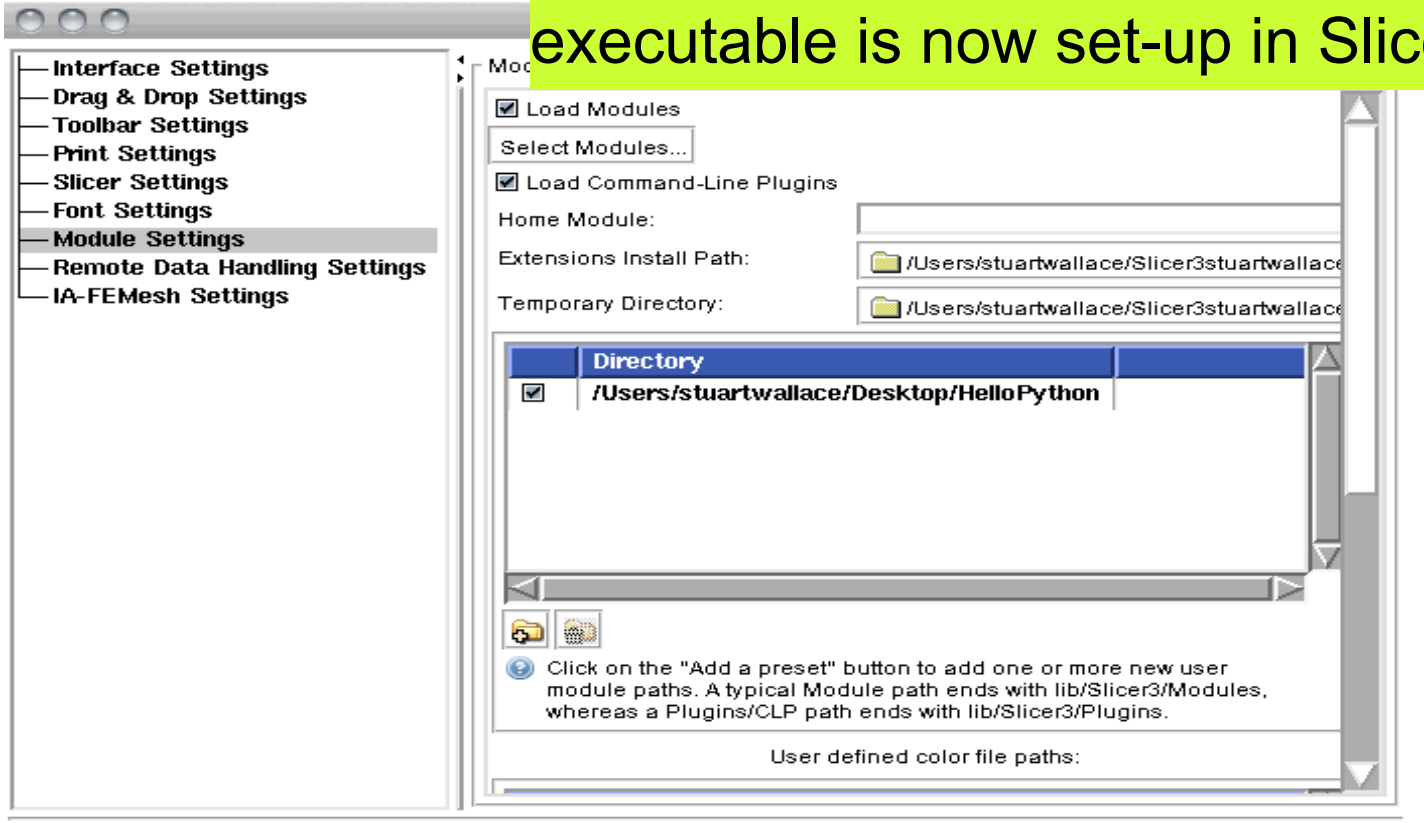


Integrating HelloPython to Slicer3



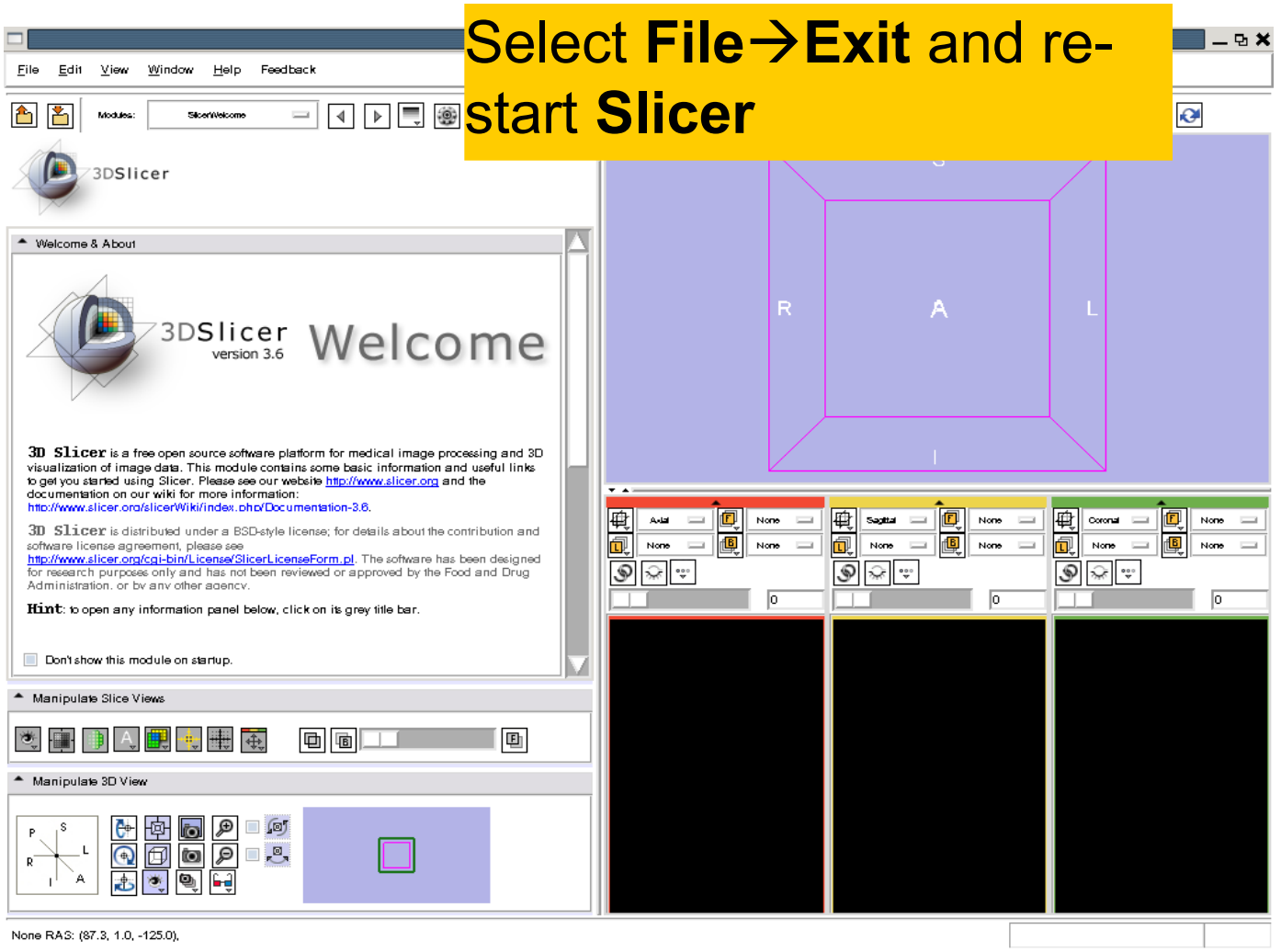
Integrating HelloPython to Slicer3

The path to the HelloPython.py executable is now set-up in Slicer3.

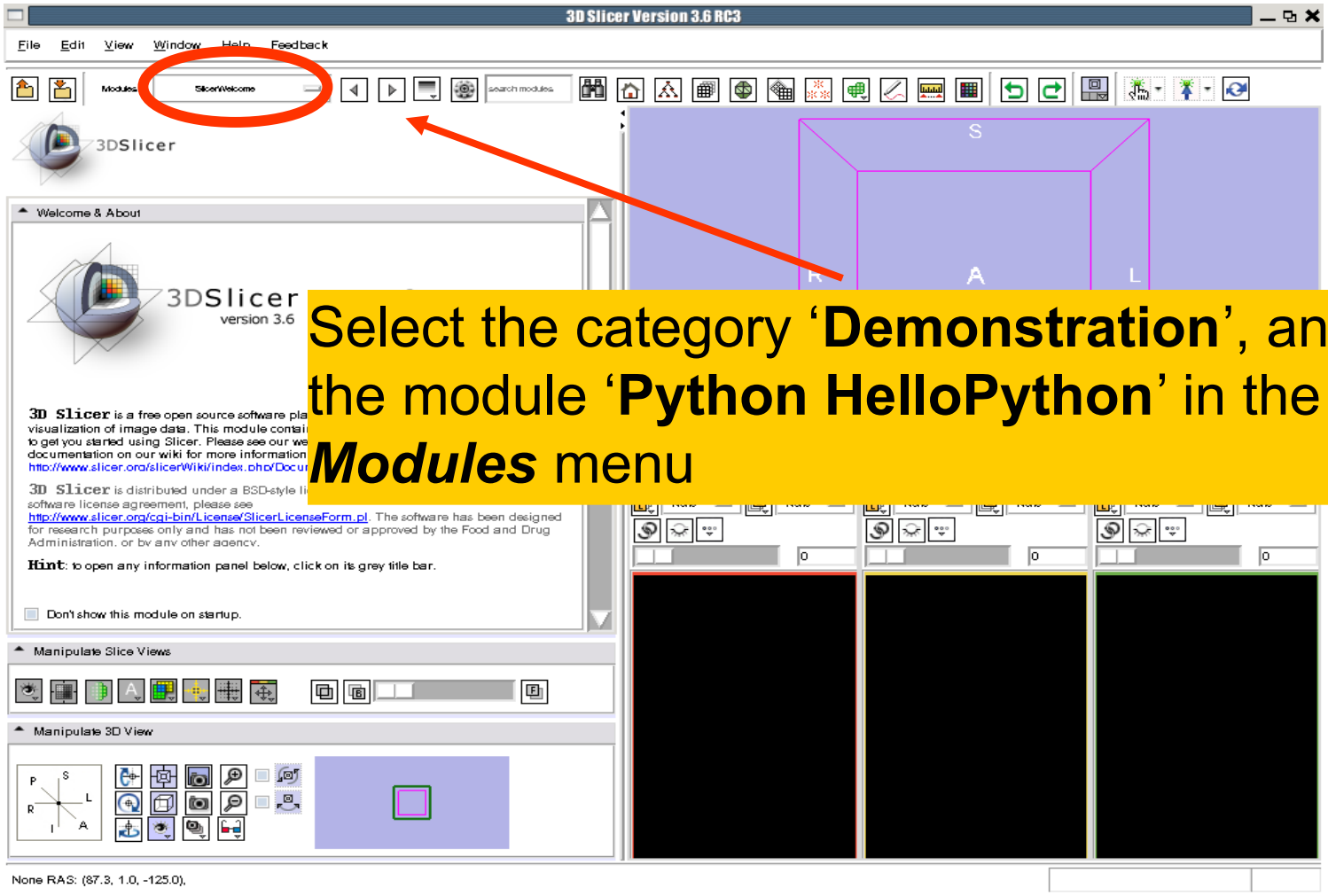


Click on the **Close** to exit the Application Settings window.

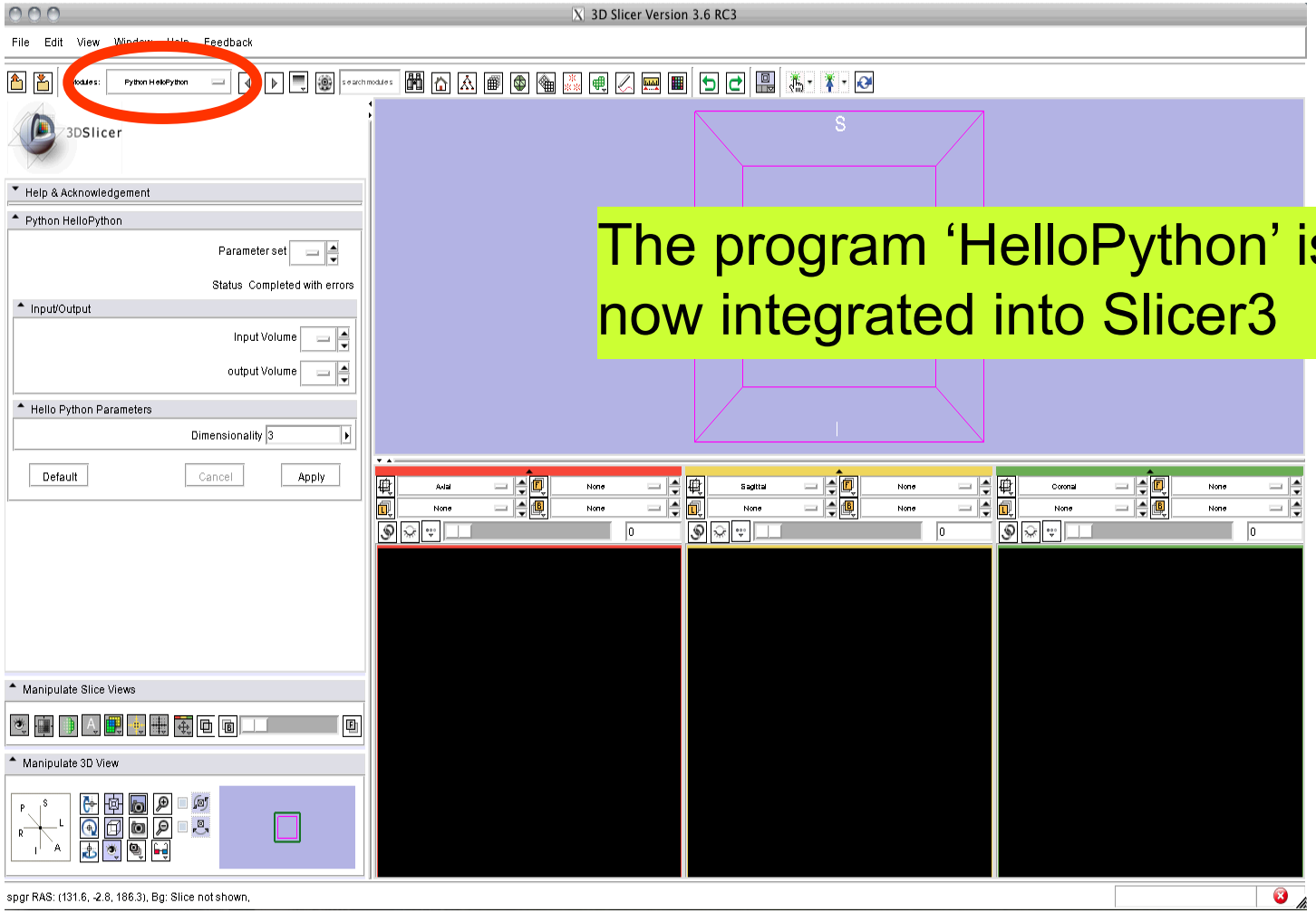
Integrating HelloPython to Slicer3



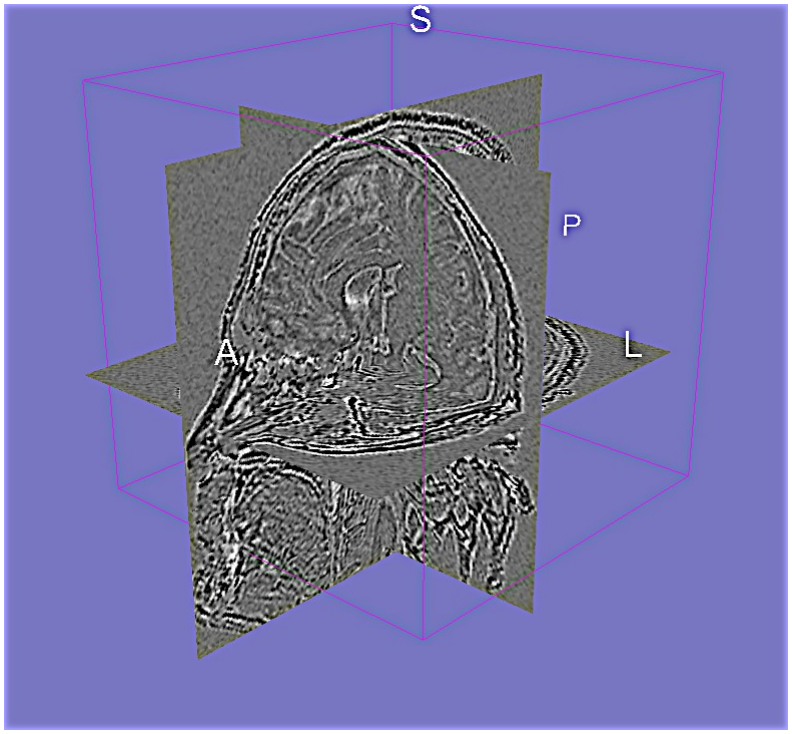
HelloPython module



HelloPython Module



The program 'HelloPython' is now integrated into Slicer3



Part B: Implementing the Laplace* Operator

*named after Pierre-Simon, marquis de Laplace (1749-1827)

Execute Function

```
def Execute (HelloPythonInputVolume, HelloPythonOutputVolume):

    Slicer = __import__("Slicer")
    slicer = Slicer.slicer
    scene = slicer.MRMLScene
    inputVolume = scene.GetNodeByID(HelloPythonInputVolume)
    outputVolume = scene.GetNodeByID(HelloPythonOutputVolume)

    return
```

Add the I/O code

Laplace Operator

```
def Execute (HelloPythonInputVolume, HelloPythonOutputVolume):

    Slicer = __import__("Slicer")
    slicer = Slicer.slicer
    scene = slicer.MRMLScene
    inputVolume = scene.GetNodeByID(HelloPythonInputVolume)
    outputVolume = scene.GetNodeByID(HelloPythonOutputVolume)
    laplacian = slicer.vtkImageLaplacian()
    laplacian.SetInput(inputVolume.GetImageData()

    return
```

Add the Laplace operator

Laplace Operator

```
<parameters>
  <label>Input/Output</label>
  <description>Input/output parameters</description>
  <image>
    <name>HelloPythonInputVolume</name>
    <label>Input Volume</label>
    <channel>input</channel>
    <index>0</index>
    <description>input volume</description>
   </image>
   <image>
     <name>HelloPythonOutputVolume</name>
     <label>Output Volume</label>
     <channel>output</channel>
     <index>1</index>
     <description>output volume</description>
    </image>
</parameters>
```

```
<parameters>
  <label>Hello Python Parameters</label>
  <description> Parameters of the Python Hello Python module </description>
</parameters>
```

Add a new parameter group for the Laplace operator

Laplace Operator

```
<parameters>
  <label>Hello Python Parameters</label>
  <description>Parameters of the Python Hello Python module </description>
   <integer>
    <name>dimensionality</name>
    <longflag>dimensionality</longflag>
    <description>Dimensionality of the Laplace operator</description>
    <label>Dimensionality</label>
    <default>3</default>
    <constraints>
      <minimum>2</minimum>
      <maximum>3</maximum>
    </constraints>
  </integer>
</parameters>
```

Add the Laplace operator's dimensionality

Laplace Operator

```
def Execute (HelloPythonInputVolume, HelloPythonOutputVolume,
dimensionality=3):
    Slicer = __import__("Slicer")
    slicer = Slicer.slicer
    scene = slicer.MRMLScene
    inputVolume = scene.GetNodeByID(HelloPythonInputVolume)
    outputVolume = scene.GetNodeByID(HelloPythonOutputVolume)
    laplacian = slicer.vtkImageLaplacian()
    laplacian.SetInput(inputVolume.GetImageData())
    laplacian.SetDimensionality(dimensionality)

    return
```

Set-up the corresponding dimensionality parameter in the Python code

Laplace Operator

```
def Execute (HelloPythonInputVolume, HelloPythonOutputVolume,
dimensionality=3):
    Slicer = __import__("Slicer")
    slicer = Slicer.slicer
    scene = slicer.MRMLScene
    inputVolume = scene.GetNodeByID(HelloPythonInputVolume)
    outputVolume = scene.GetNodeByID(HelloPythonOutputVolume)
    laplacian = slicer.vtkImageLaplacian()
    laplacian.SetInput(inputVolume.GetImageData()
    laplacian.SetDimensionality(dimensionality)
    laplacian.Update()
    outputVolume.SetAndObserveImageData(laplacian.GetOutput())
    return
```

Add code to get the output of the Laplace operator

Laplace Operator

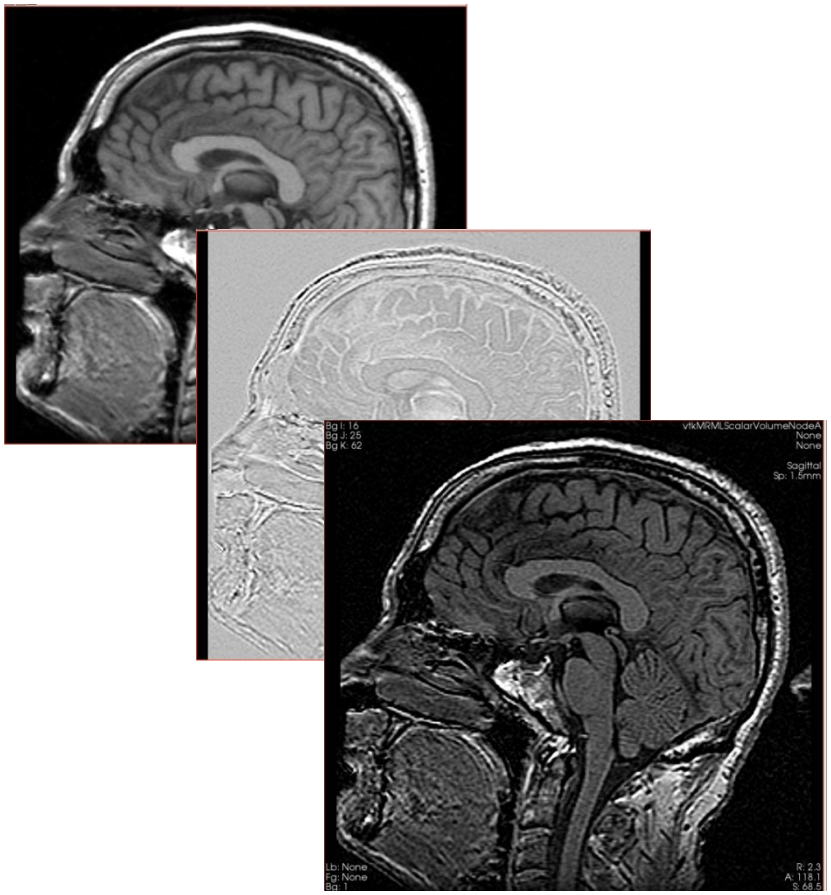
```
def Execute (HelloPythonInputVolume, HelloPythonOutputVolume,
dimensionality=3):
    Slicer = __import__("Slicer")
    slicer = Slicer.slicer
    scene = slicer.MRMLScene
    inputVolume = scene.GetNodeByID(HelloPythonInputVolume)
    outputVolume = scene.GetNodeByID(HelloPythonOutputVolume)
    laplacian = slicer.vtkImageLaplacian()
    laplacian.SetInput(inputVolume.GetImageData()
    laplacian.SetDimensionality(dimensionality)
    laplacian.Update()
    outputVolume.SetAndObserveImageData(laplacian.GetOutput())
    matrix = slicer.vtkMatrix4x4()
    inputVolume.GetIJKToRASMatrix(matrix)
    outputVolume.SetIJKToRASMatrix(matrix)
    return
```

Place back the Laplacian of the image in the RAS reference system.

Integrating HelloPython to Slicer3

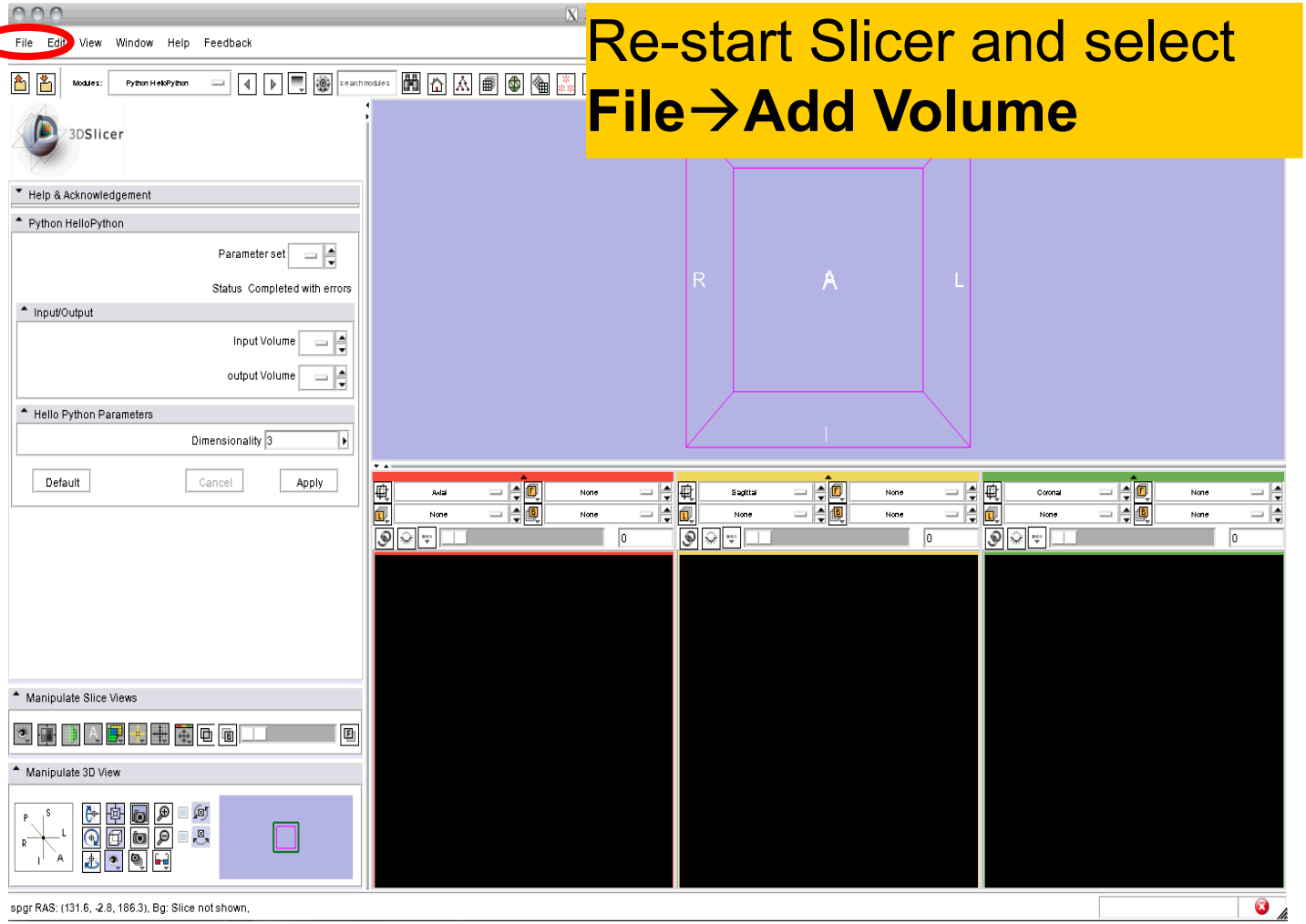
```
def Execute (HelloPythonInputVolume, HelloPythonOutputVolume,
dimensionality=3):
        Slicer = __import__("Slicer")
        slicer = Slicer.slicer
        scene = slicer.MRMLScene
        inputVolume = scene.GetNodeByID(HelloPythonInputVolume)
        outputVolume = scene.GetNodeByID(HelloPythonOutputVolume)
        laplacian = slicer.vtkImageLaplacian()
        laplacian.SetInput(inputVolume.GetImageData()
        laplacian.SetDimensionality(dimensionality)
        laplacian.Update()
        outputVolume.SetAndObserveImageData(laplacian.GetOutput())
        matrix = slicer.vtkMatrix4x4()
        inputVolume.GetIJKToRASMatrix(matrix)
        outputVolume.SetIJKToRASMatrix(matrix)
        return
```

Save the HelloPython.py file and exit Slicer.



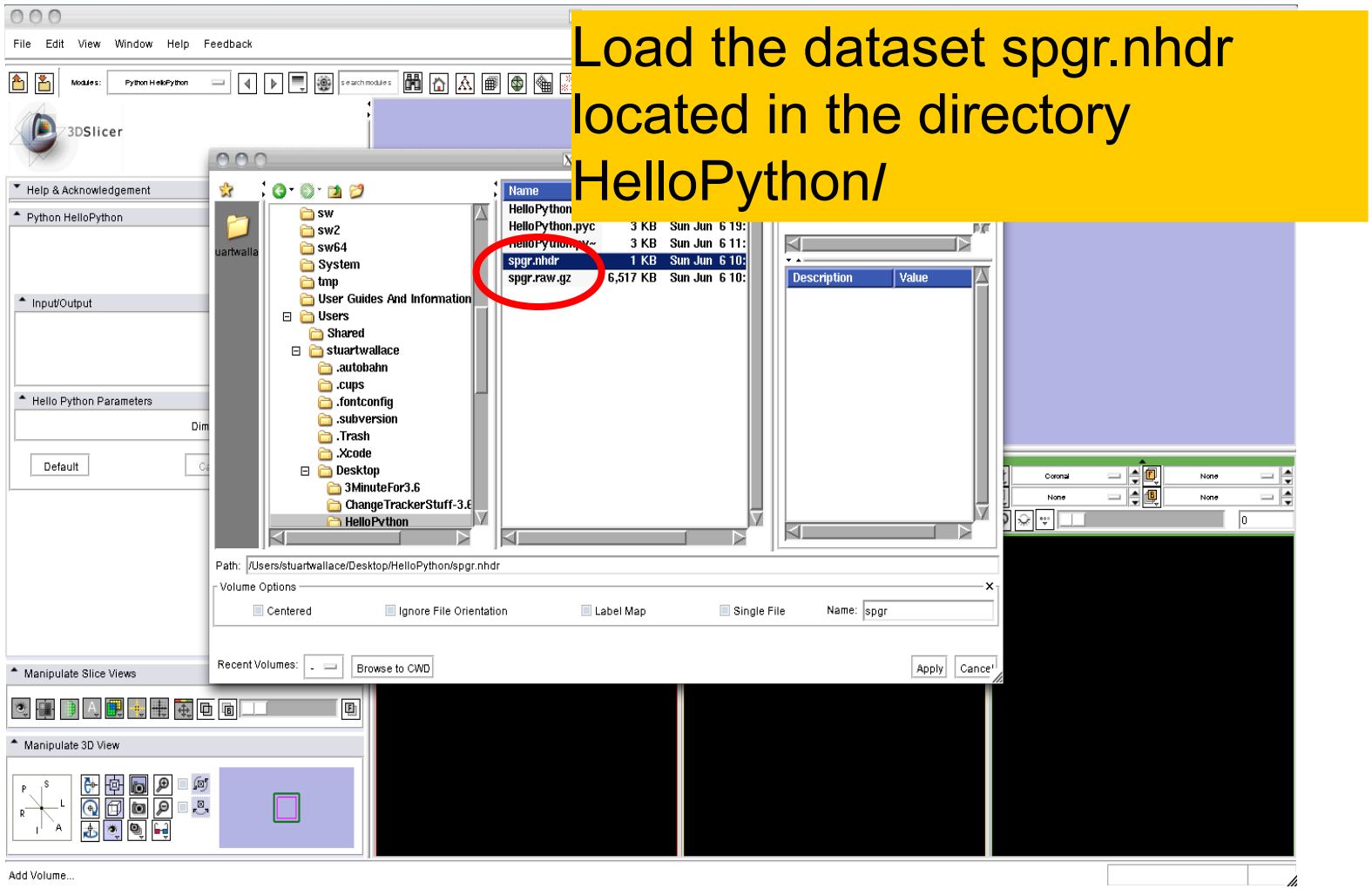
Part C: Image Sharpening with the Laplace Operator

Running the Laplace Operator

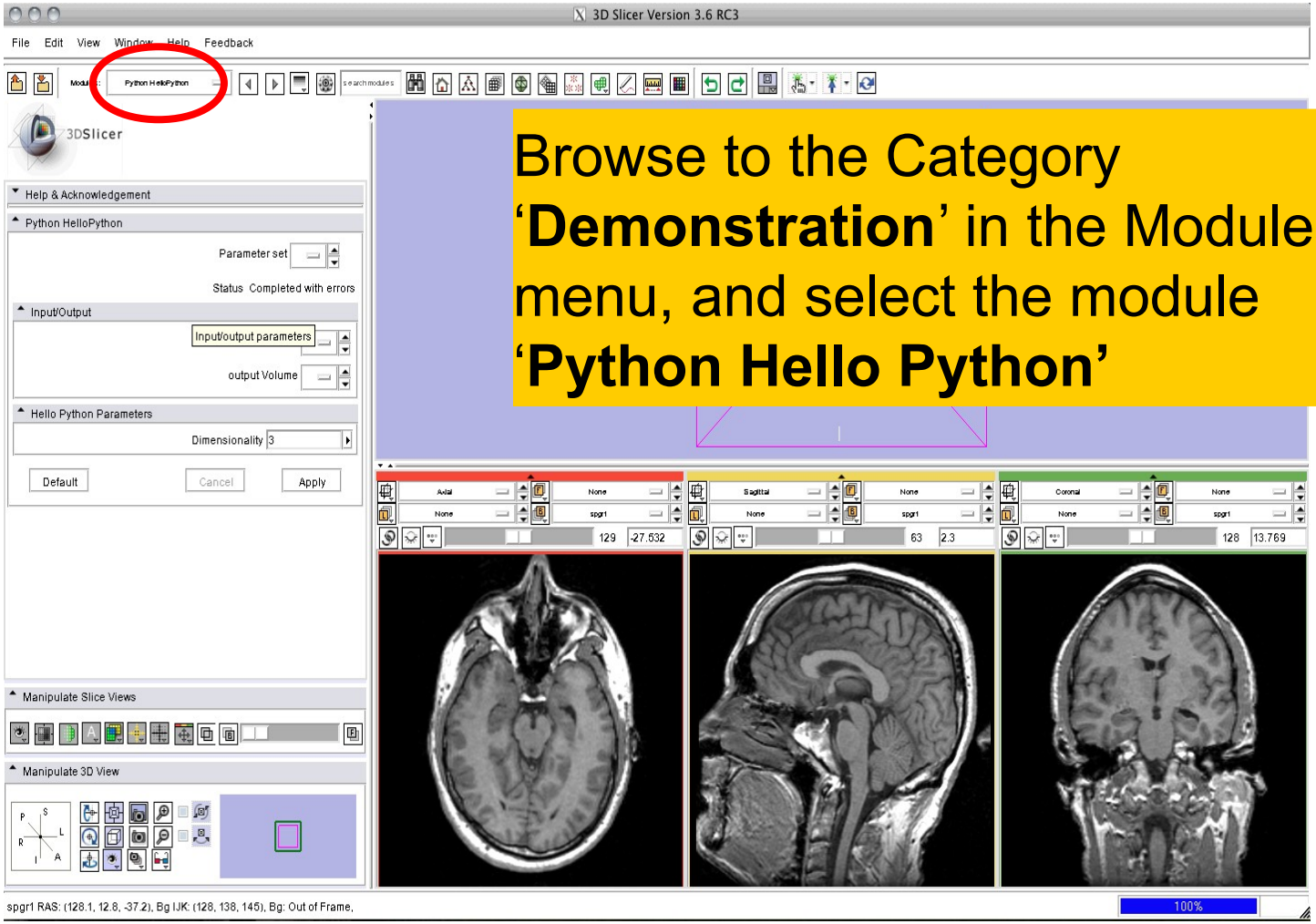


Re-start Slicer and select
File→Add Volume

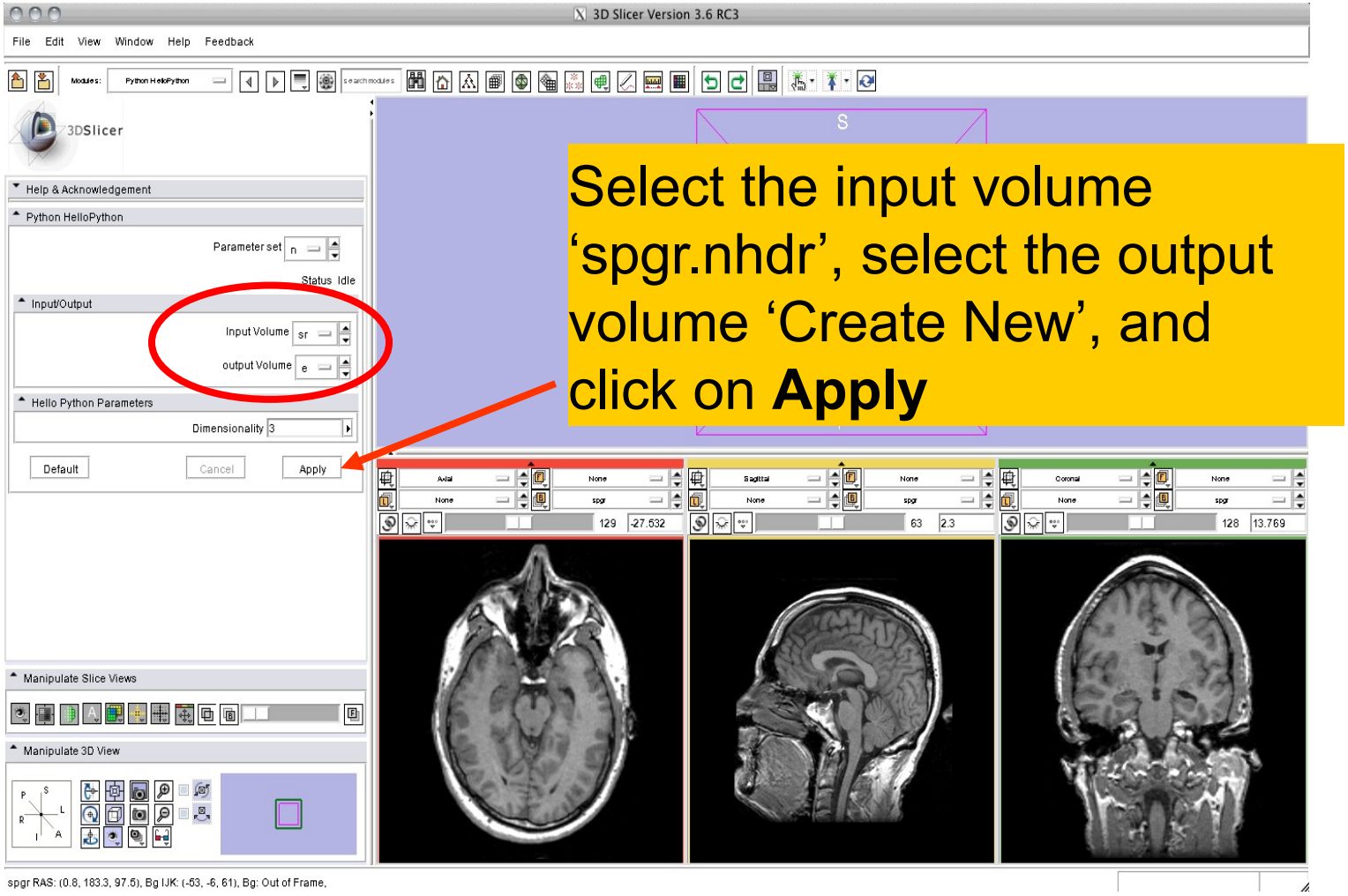
Running the Laplace Operator



Running the Laplace Operator

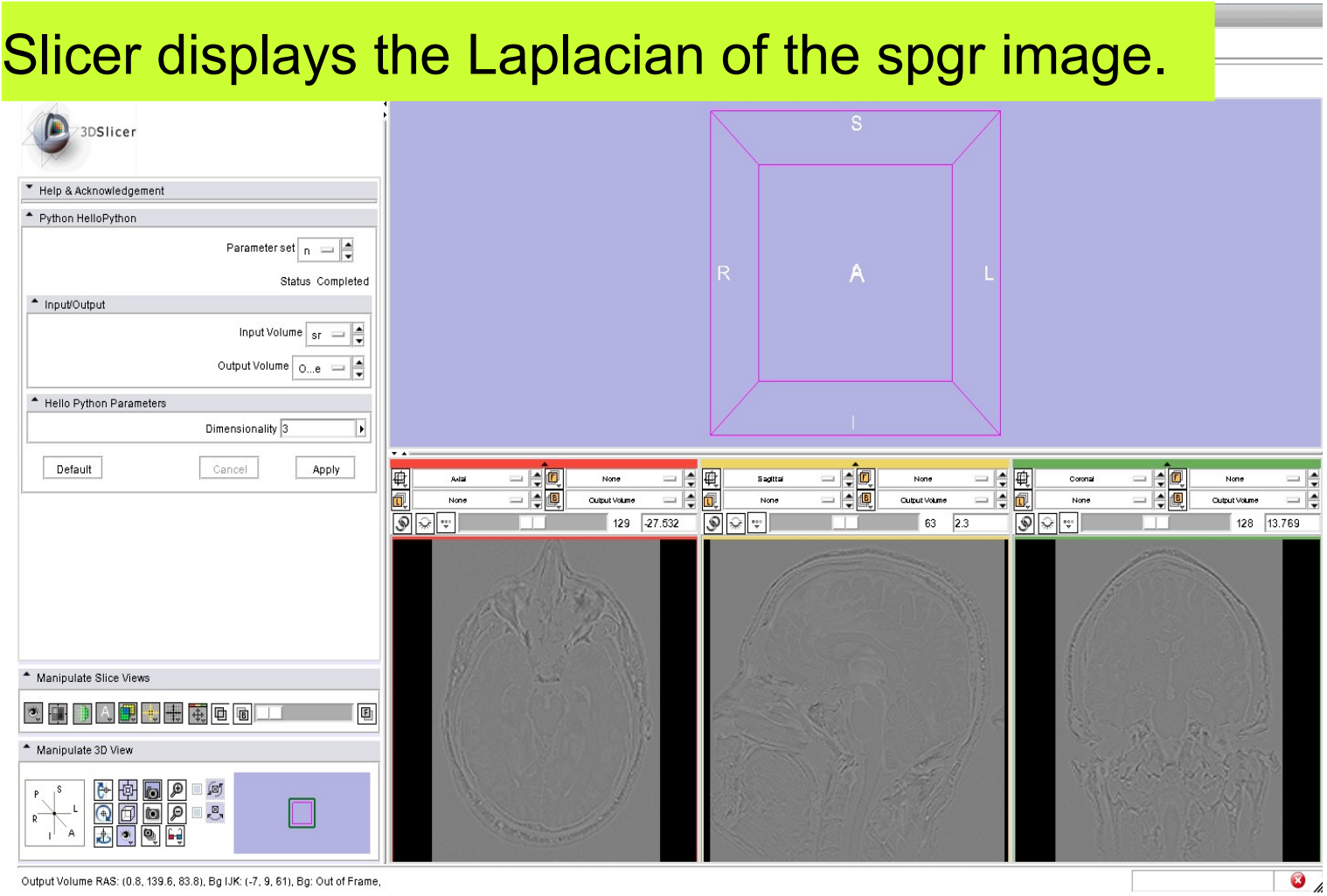


Running the Laplace Operator



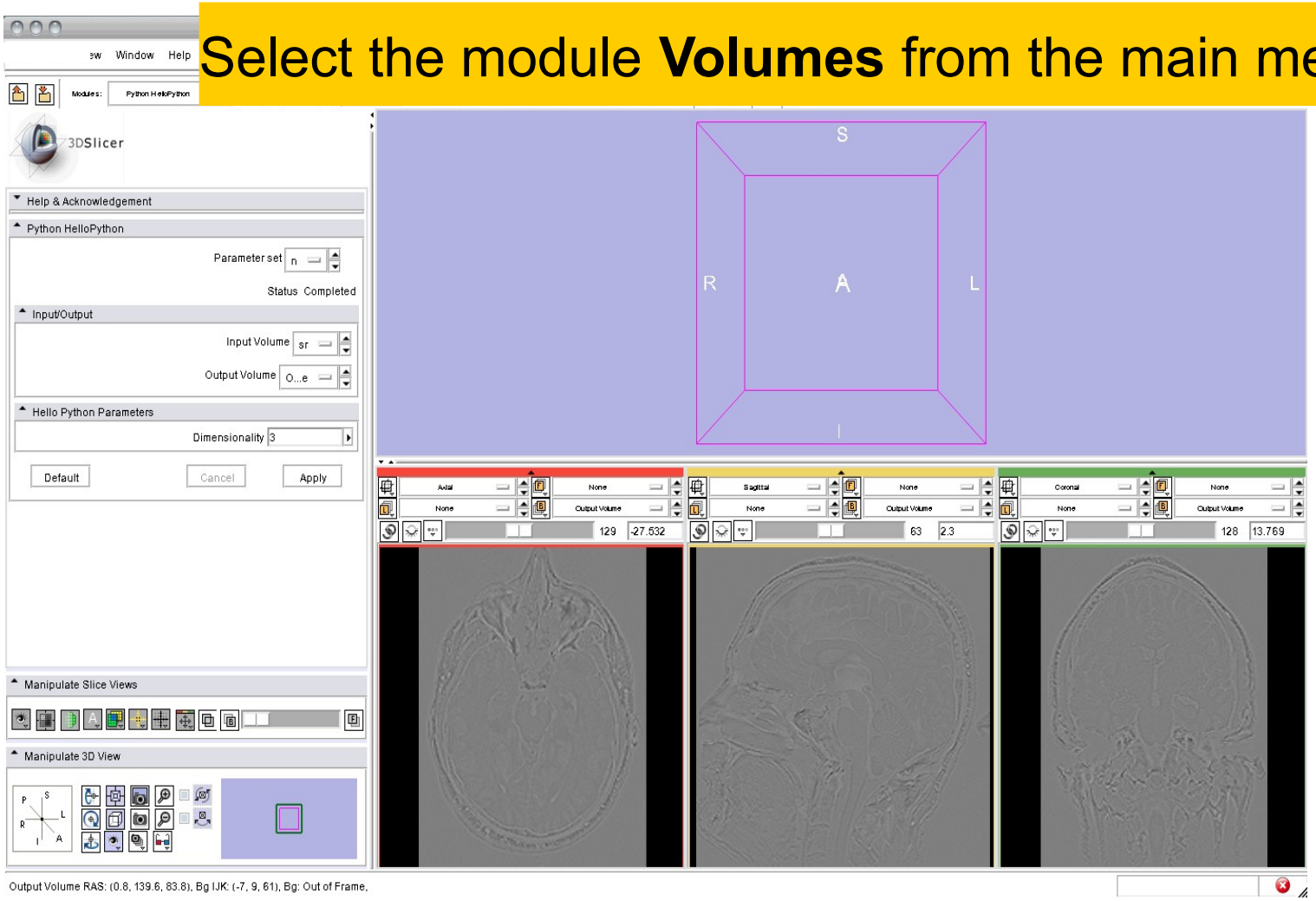
Running the Laplace Operator

Slicer displays the Laplacian of the spgr image.



Laplacian of the image

Select the module **Volumes** from the main menu



Laplacian of the image

Set the Active Volume to **Output Volume** and adjust the Window/Level parameters

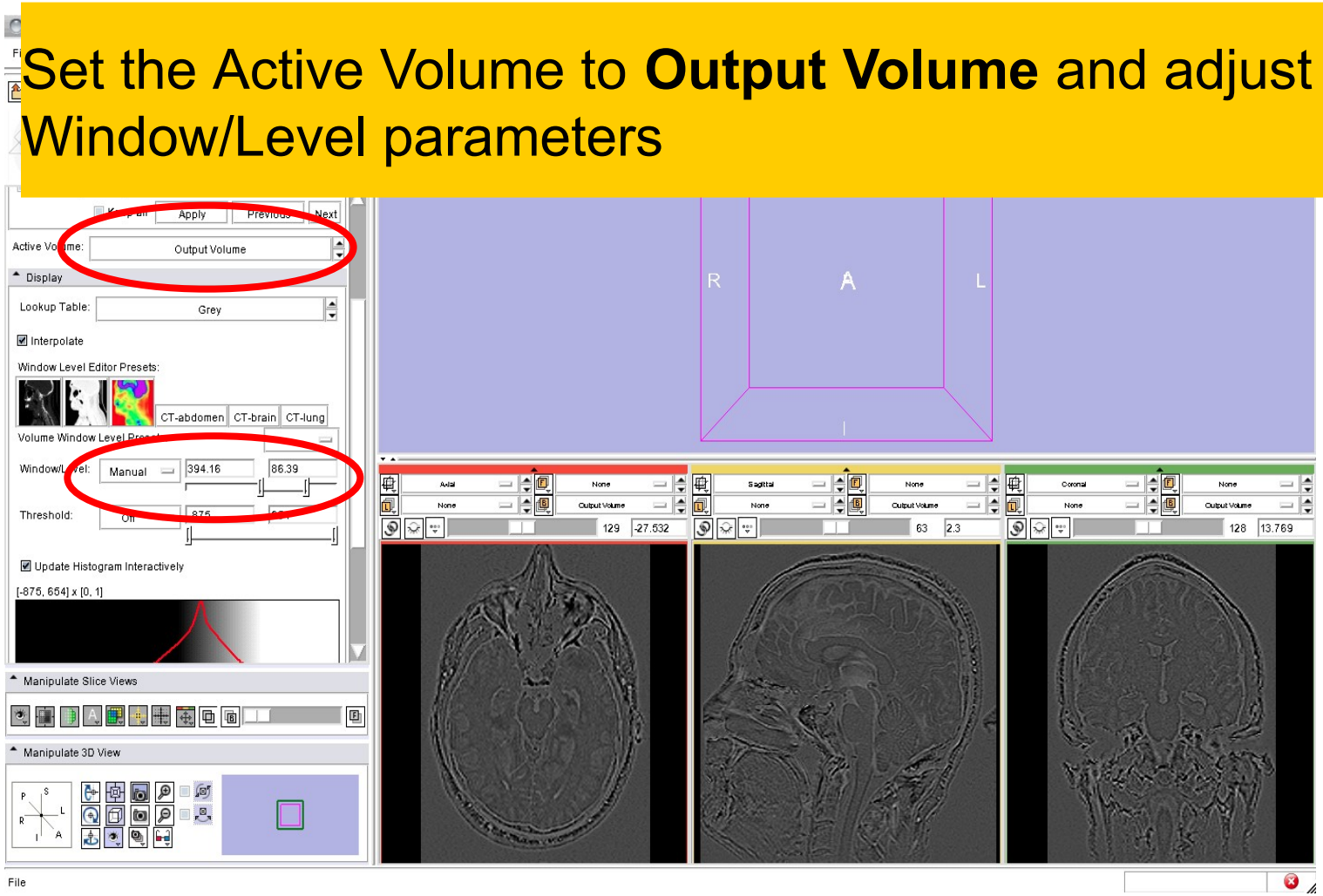
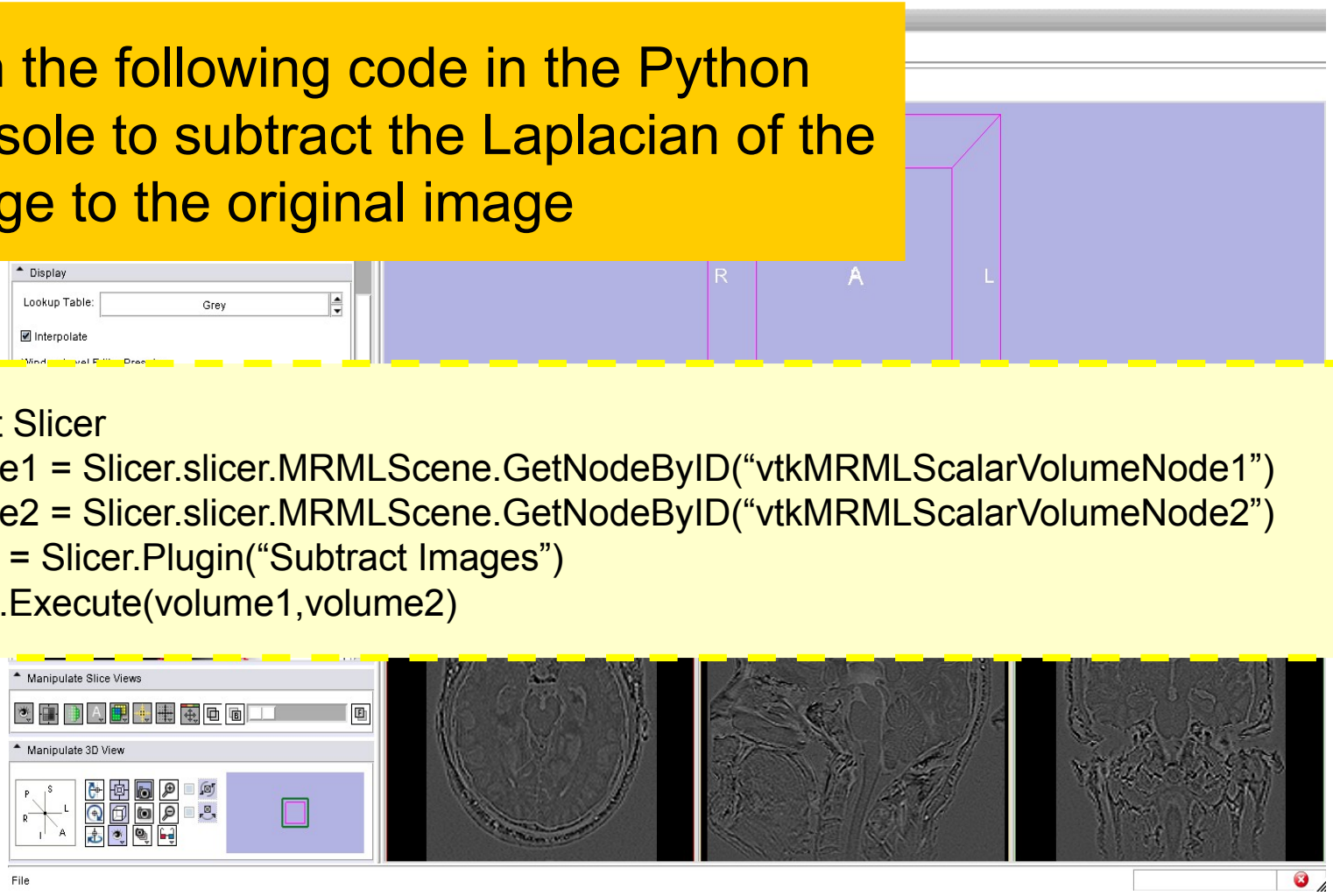


Image Sharpening

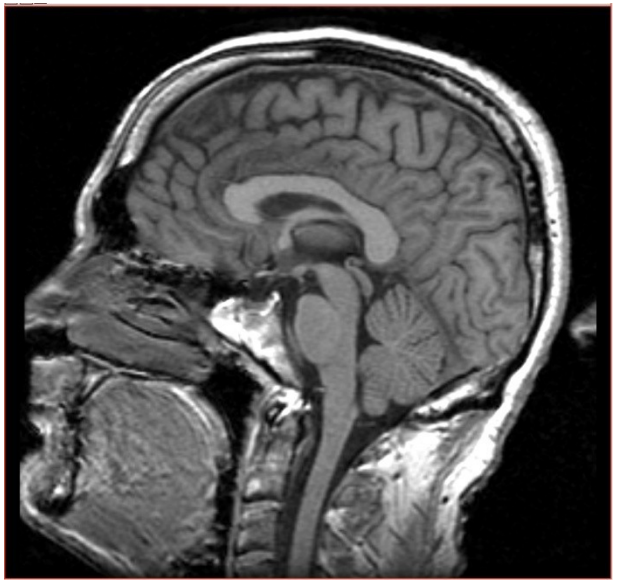
Run the following code in the Python console to subtract the Laplacian of the image to the original image



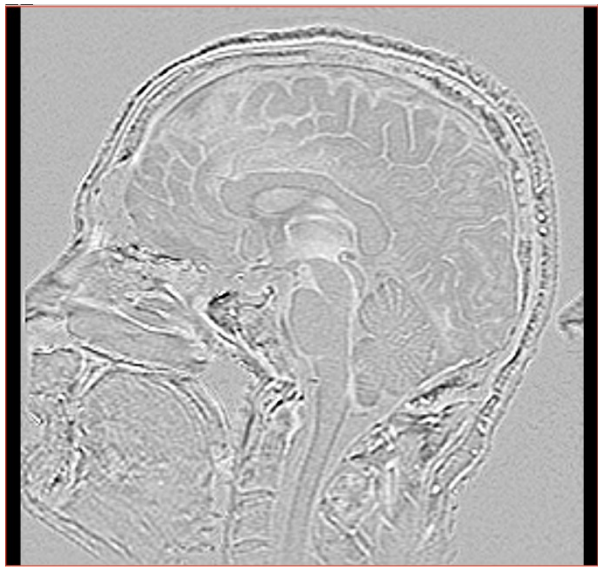

```
import Slicer
volume1 = Slicer.slicer.MRMLScene.GetNodeByID("vtkMRMLScalarVolumeNode1")
volume2 = Slicer.slicer.MRMLScene.GetNodeByID("vtkMRMLScalarVolumeNode2")
plugin = Slicer.Plugin("Subtract Images")
plugin.Execute(volume1,volume2)
```

Image Sharpening

original



Laplacian



Laplacian filtered

Conclusion

- This course demonstrates how to integrate an external program in Python within Slicer3

- The Execution Model of Slicer3 provides a simple mechanism for incorporating command line programs as Slicer modules in Python.

Acknowledgments

National Alliance for Medical Image Computing

NIH U54EB005149

Neuroimage Analysis Center

NIH P41RR013218