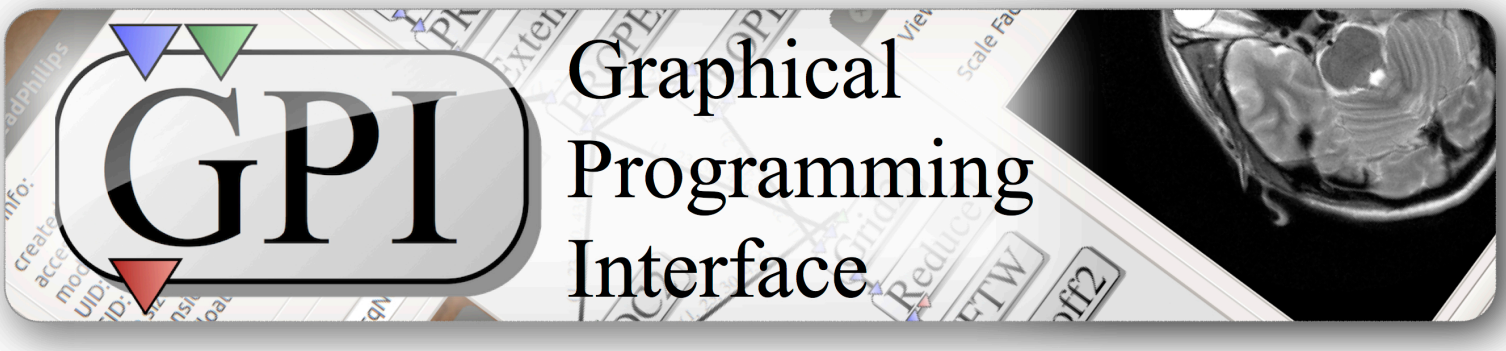
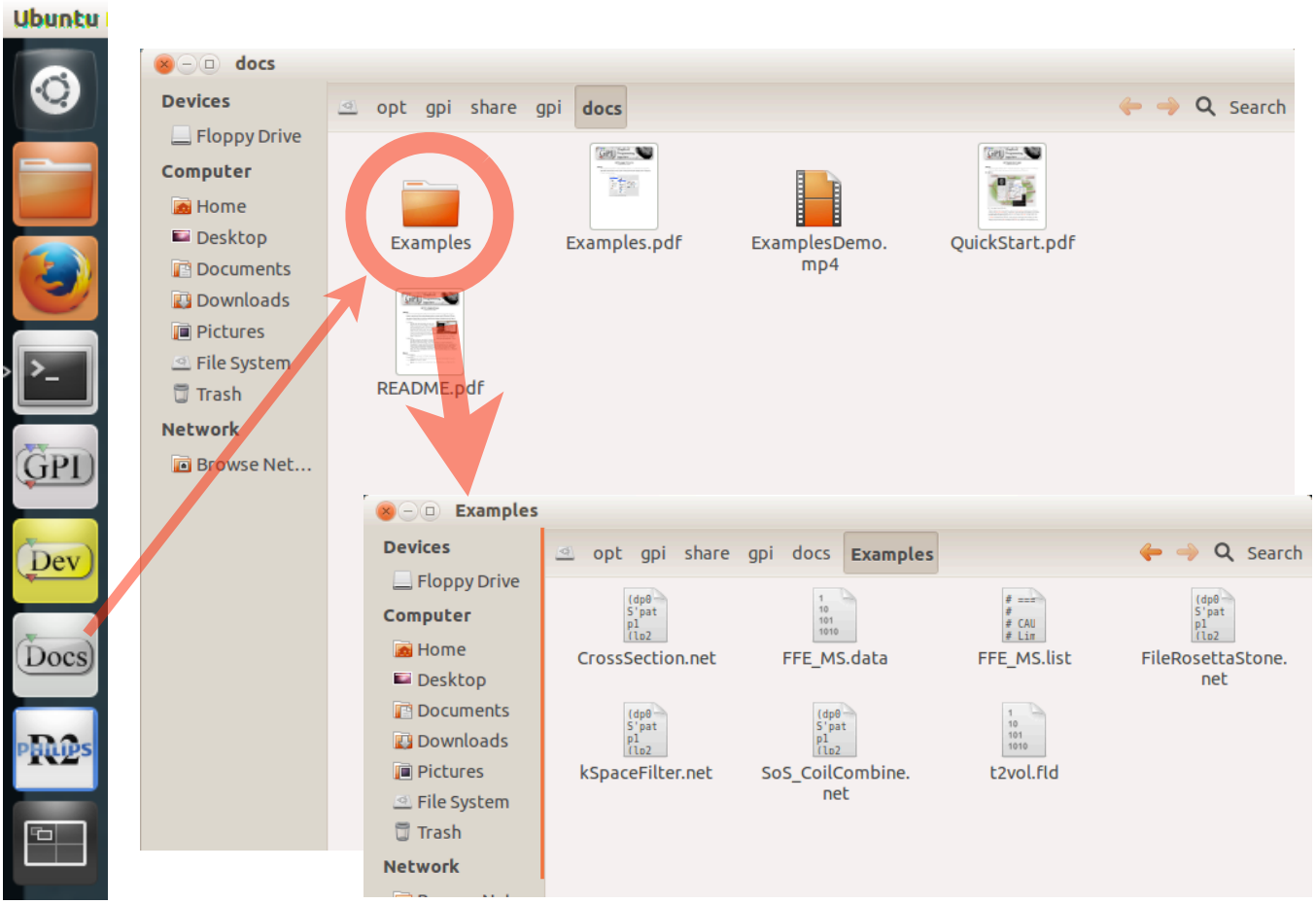


GPI Example Networks

Nicholas R. Zwart, Ph.D.

Summary:

GPI was packaged with a few demo networks to illustrate some of the use cases and features that might not be immediately apparent. This document gives a description of each network and points out some of the key nodes that would be adjusted during a working session. The example networks (the files with the '.net' extension) can be drag-n-dropped from the file browser to the GPI network canvas. To locate the example networks, click the 'Docs' icon on the left side toolbar, then double-click the 'Examples' sub-folder.

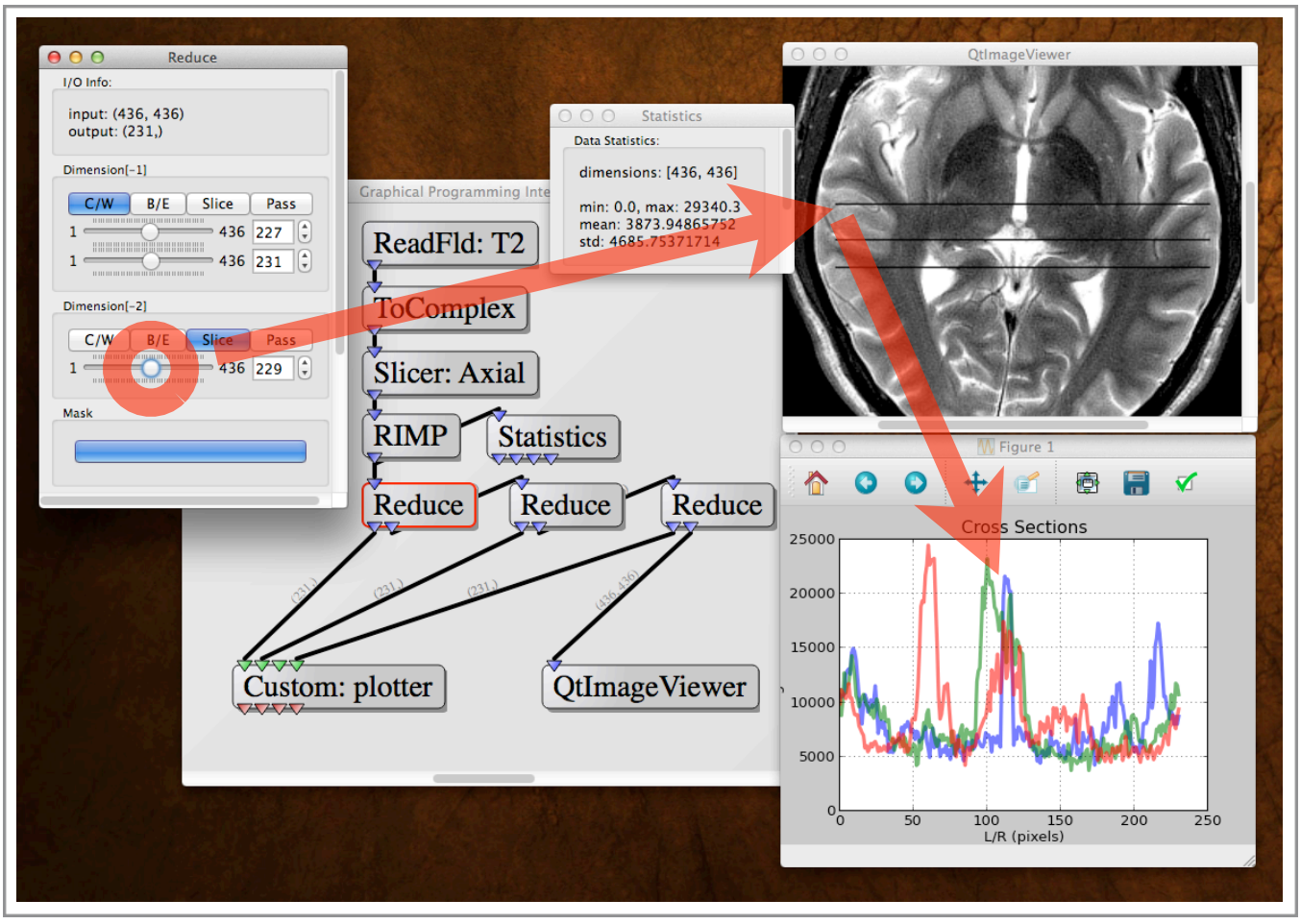


The demo network location in the documents folder.

CrossSection_GPI.net:

The Cross-Section example network highlights the versatility of the Reduce node. Reduce can be used to slice through multidimensional arrays and simultaneously crop dimensions. The three Reduce nodes are making use of crop and slice operations simultaneously with the 'Mask' feature to extract cross-sections of the reconstructed image data. The slice axis slider (screenshot) is highlighted with a red circle. This slider can be used to vary the y-axis position of the cropped cross-section shown in the image viewer which is simultaneously updated in the plotter. The cropping slider, located above the slicing slider, can be adjusted to change the cross-sectional width and x-axis position.

The edge connections between nodes show the dimensionality of the numpy array that is being passed between them. Hovering over a port shows this dimension information and information about the port connection requirements. The statistics node can also be used to get the dimensionality and basic information about the data such as the min, max, mean, and standard deviation.

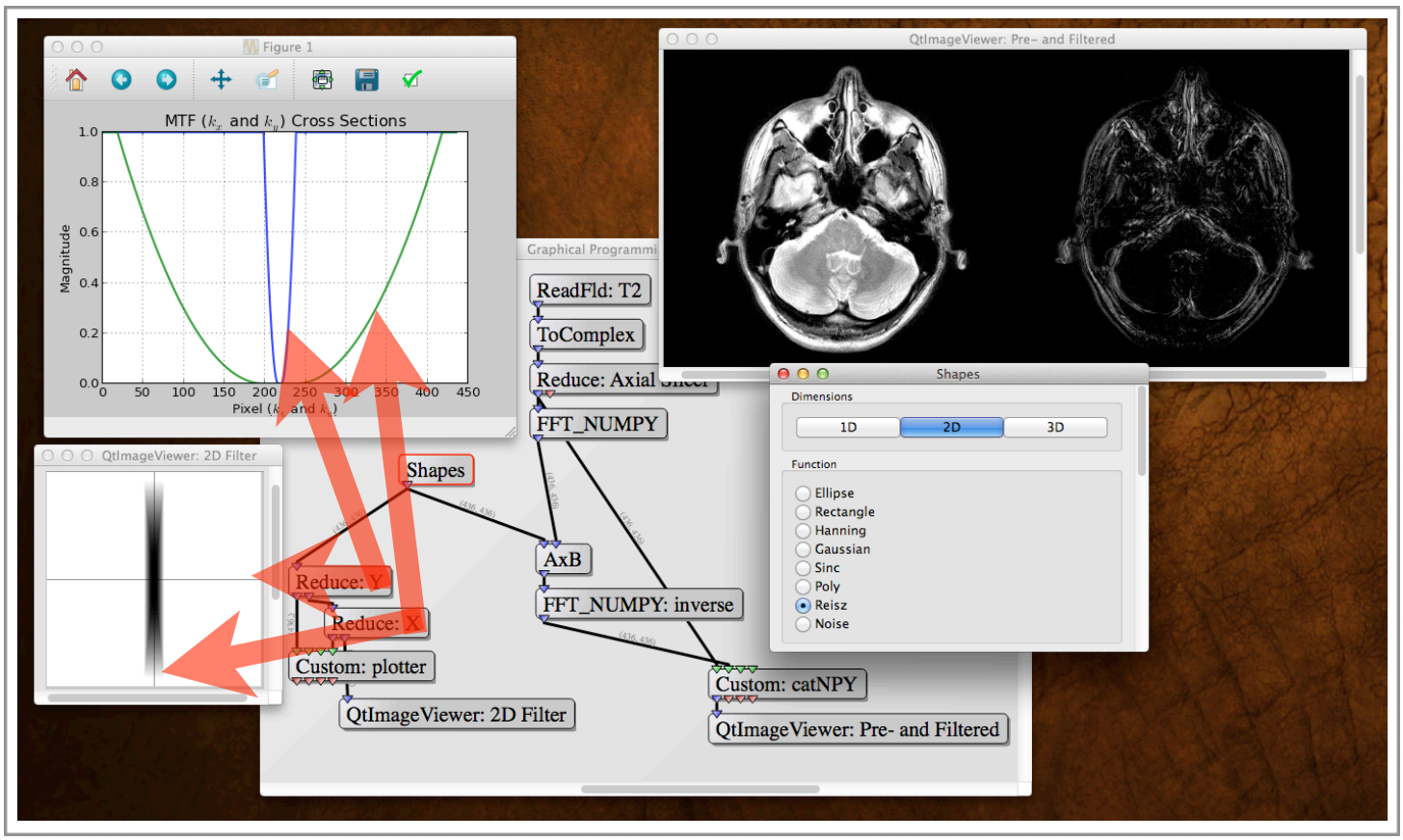


A screen-capture of the example network CrossSection_GPI.net.

kSpaceFilter_GPI.net:

This network demonstrates how the Shapes node can be used to generate a k-space filter. As shown in the screen-shot image, the Shapes node has a list of predefined shapes that can be further modified with the hidden widgets further down in the Node Menu. Shapes takes a chosen dimensionality, the dimension sizes (in this case the size of the image matrix), and other parameters that modify the filter window length, transitions, peak and stop values. In this example, a 2D filter is generated, as shown in the viewer widget, and its x and y cross-sections are plotted. The 2D filter array is multiplied by the Fourier transform of the 2D image. The image is then transformed back into the image domain to reveal a high-pass filter.

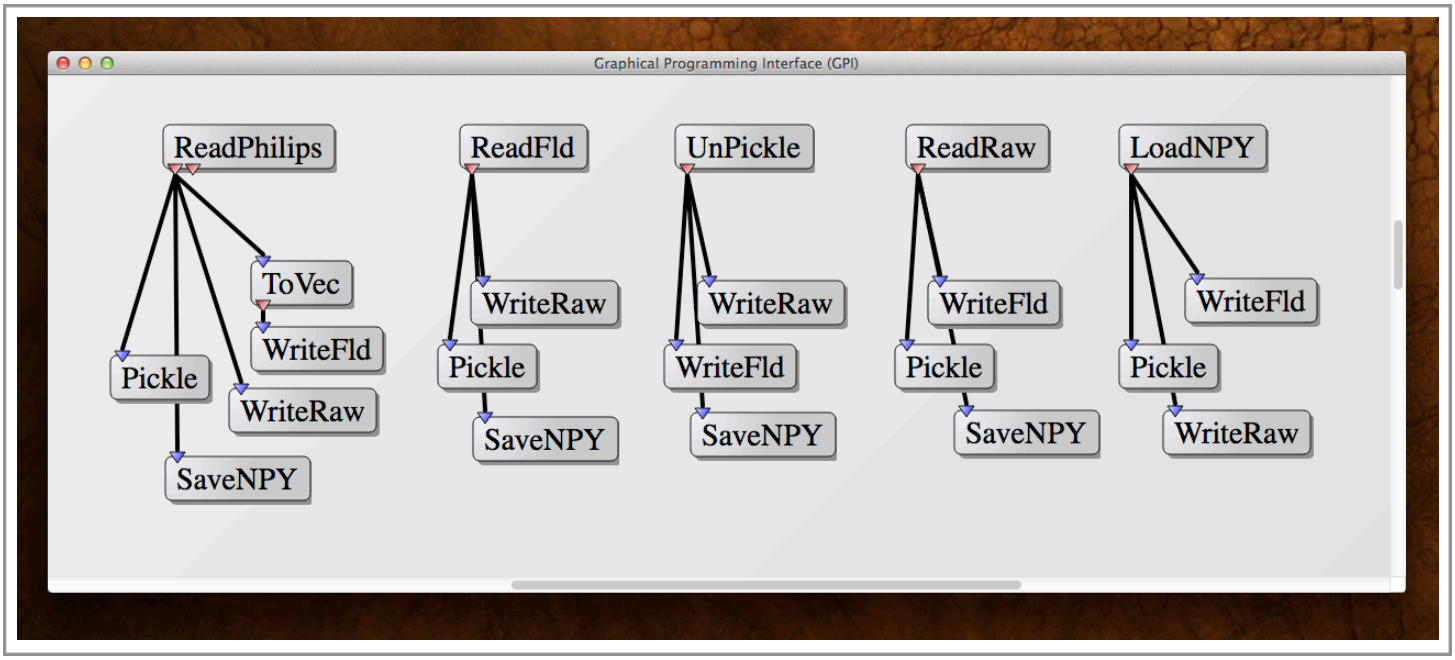
The original image and filtered image are concatenated using the numpy.concatenate() function implemented in the Custom node and are then passed to the viewer node. The Custom node is also used to implement the matplotlib plotter package. This custom code can be modified to set the title, axis labels, legends, etc...



A screen-capture of the example network kSpaceFilter_GPI.net. The red arrows indicate the cross-section each reduce is taking.

FileRosettaStone_GPI.net:

GPI can use any python bindable datatype to transmit data between connected nodes. Most of the time, numpy arrays are used as the common data format for this communication. When reader and writer nodes are written to support a common format (such as numpy arrays) the GPI framework can be used as a rosetta stone for file formats. In this way, node developers benefit by only having to write file I/O nodes for their specific needs and can then share data with others in formats available in existing node packages.

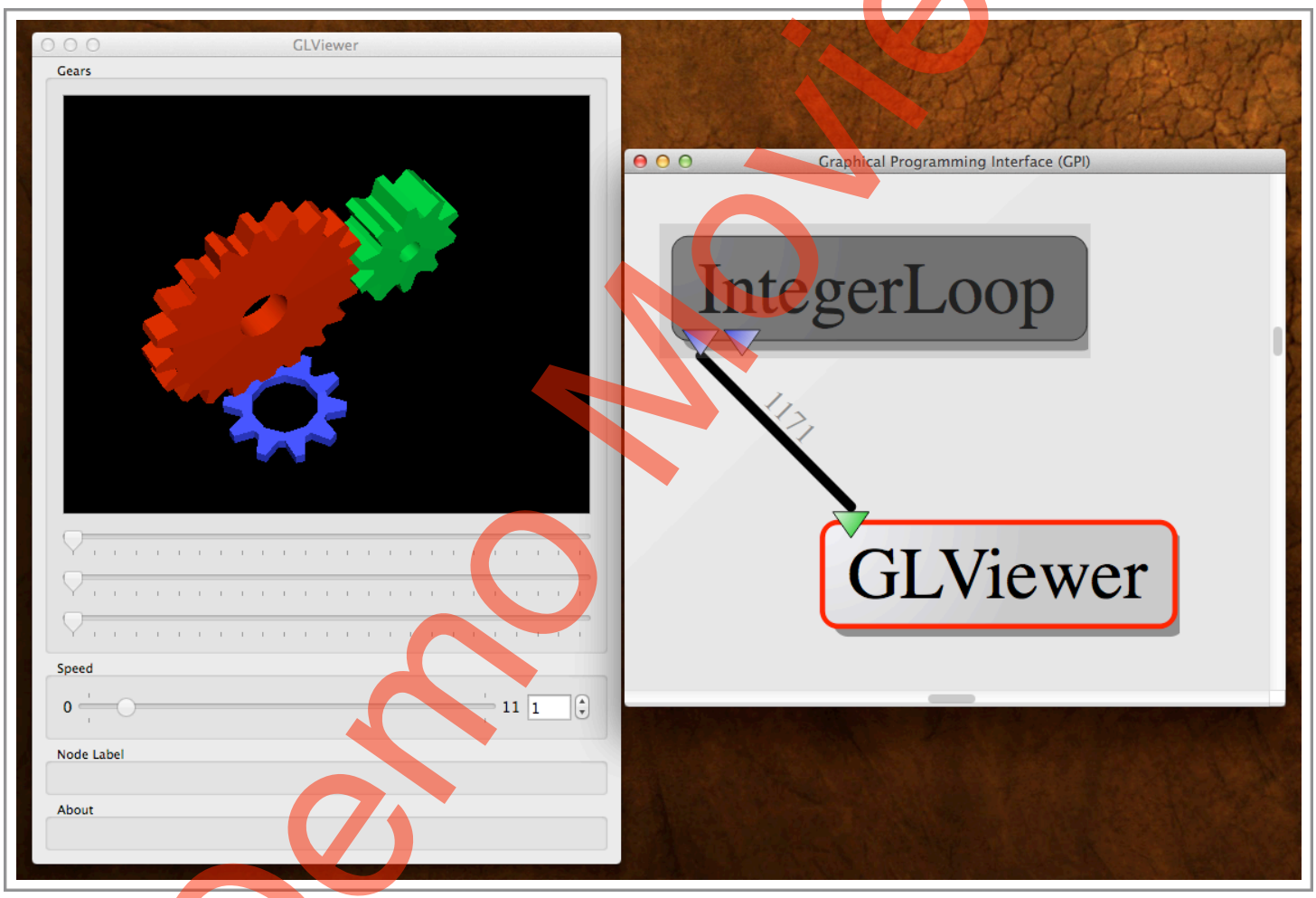


A screen-capture of the example network FileRosettaStone_GPI.net.

EmbeddedGLDemo_GPI.net:

Currently, there are plans to integrate GL objects into the GPI framework (as a plugin) which would allow data to be rendered and explored in a 3D interactive window. This is often useful for visualizing 3D k-space trajectories, level curves, spin simulations, etc... This example shows that a GL window can be imbedded within a GPI widget without any internal framework dependence on the PyOpenGL package. The GLViewer node itself holds the dependencies.

In this network, the GLViewer takes an event as a queue to proceed to the next frame of the gear animation. The IntegerLoop node is a special node that sends an integer downstream and waits for all subsequent nodes to execute. Once the downstream nodes are finished, the IntegerLoop reenters the event loop causing subsequent events until it reaches its termination criteria (in this case its a max index like a for loop).

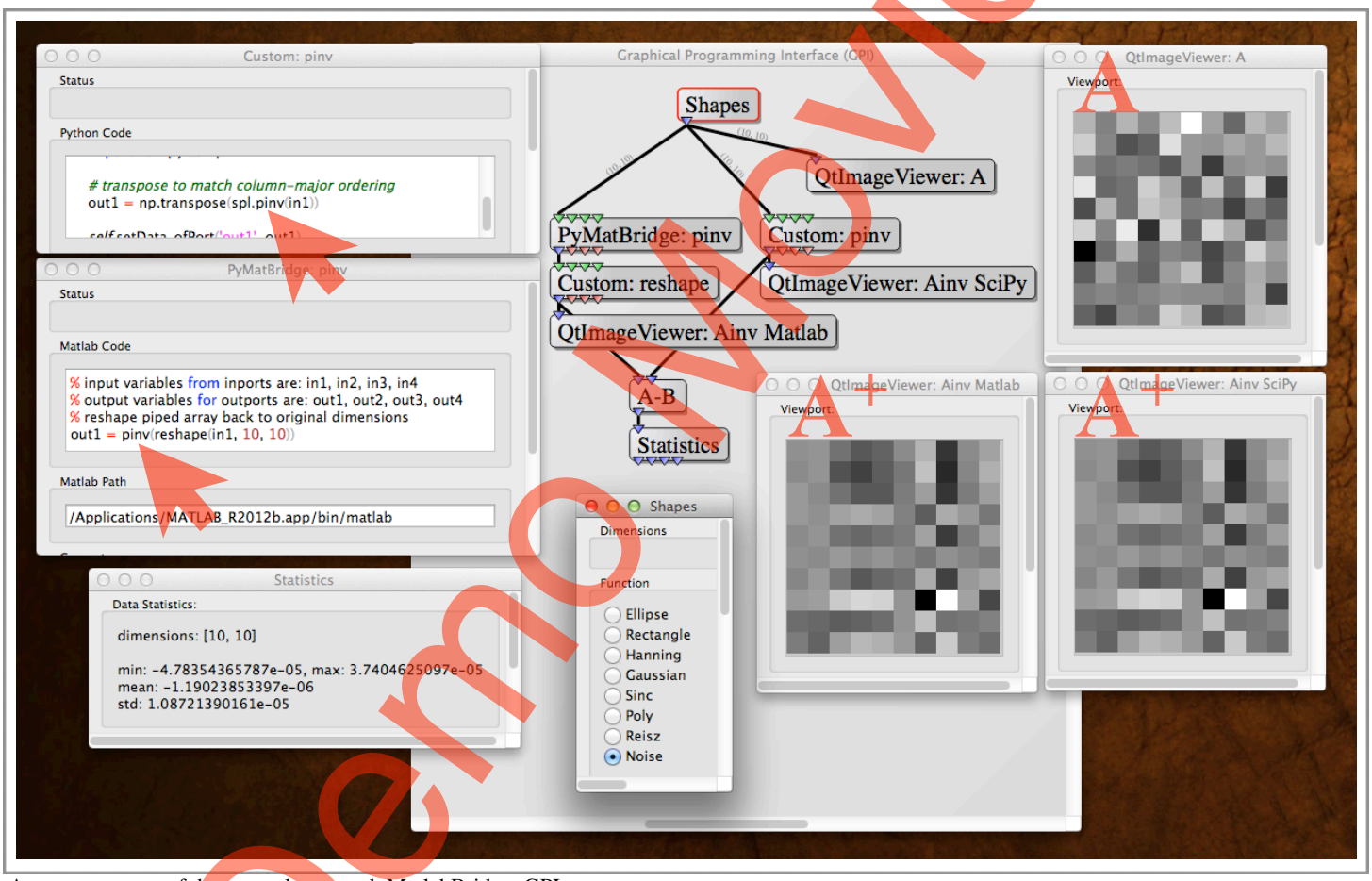


A screen-capture of the example network EmbeddedGLDemo_GPI.net.

MatlabBridge_GPI.net:

This network demonstrates the PyMatBridge node, named after the python package it implements, pymatbridge. The Matlab bridge node allows communication between Python and Matlab. It requires an installed Matlab release on the same host as the GPI session. This example compares the Matlab pseudo-inverse result to the SciPy pseudo-inverse result of a 2D matrix populated with gaussian noise (generated by the Shapes node). The resulting arrays are differenced and displayed in the Statistics node. Custom code can be entered in the PyMatBridge node editor widget allowing the user to effectively pull shared Matlab scripts or Matlab specific functionality into a GPI session.

Note: The pymatbridge module uses a web-server to send and receive data to Matlab. As a result, this communication can be slow, therefore, a Status widget (shown at the top of the PyMatBridge Node Menu) gives user feedback from the process. It is recommended that the GPI session be started from a terminal window to provide more feedback.



A screen-capture of the example network MatlabBridge_GPI.net.