1. System Requirements:

(1). ArcGIS 10.1 or higher, including Spatial Analyst Extension.
(2). Python 2.7
(3). Python packages: Enthought Python Distribution, Numpy, matplotlib, Scikit Learn.

Pythonxy is recommended, the Python 2.7 and the packages listed above are included in pythonxy. (https://code.google.com/p/pythonxy) Please uninstall Python before the installation of pythonxy. After installation, create an “arcgis.txt” file in the catalog of library (e.g. “C:\Program Files (x86)\Python\Lib”), add two rows of text like Fig. 1 in the txt file (the text is determined by the version of ArcGIS and where it is installed), and then change the extension from “txt” to “pth”.

```
C:\Program Files (x86)\ArcGIS\Desktop10.2\arcpy
C:\Program Files (x86)\ArcGIS\Desktop10.2\bin
```

Figure 1. An example of text.

2. Installation

Please keep “opfilter.tbx”, “Optimal_Filter_V0.1.py” and “Draw_ROC_Curve_V0.1.py” in a same folder. Use Catalog to load “opfilter.tbx”.
When an error like “ERROR 000576: Script associated with this tool does not exist” raises in application, right click the tool and make sure the script file in the correct place.

3. Application

3.1 Draw ROC Curve

Left double click to open “Draw ROC Curve” tool in “opfilter.tbx” to open the tool, the Fig. 2 shows the input interface.

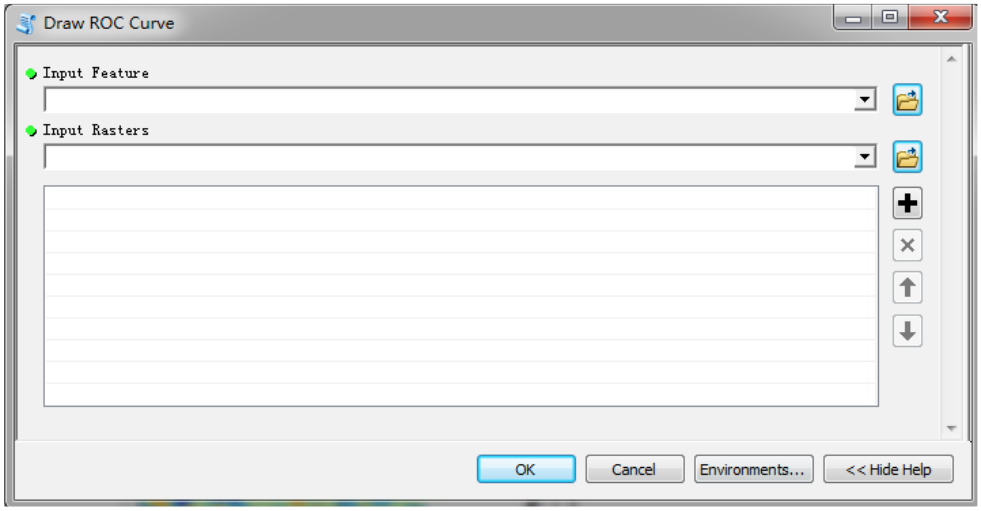

Figure 2. The input interface of Draw ROC Curve.

Input Feature: Input objective feature layer or file.
Input Raster: Input rasters, these rasters should have same properties (e.g. cell size, mask, extent, pixel type, nodata value, snap raster and spatial reference).
Step 1. Fill all parameters and click “OK”, after a few seconds, the results occurs (Fig. 3).

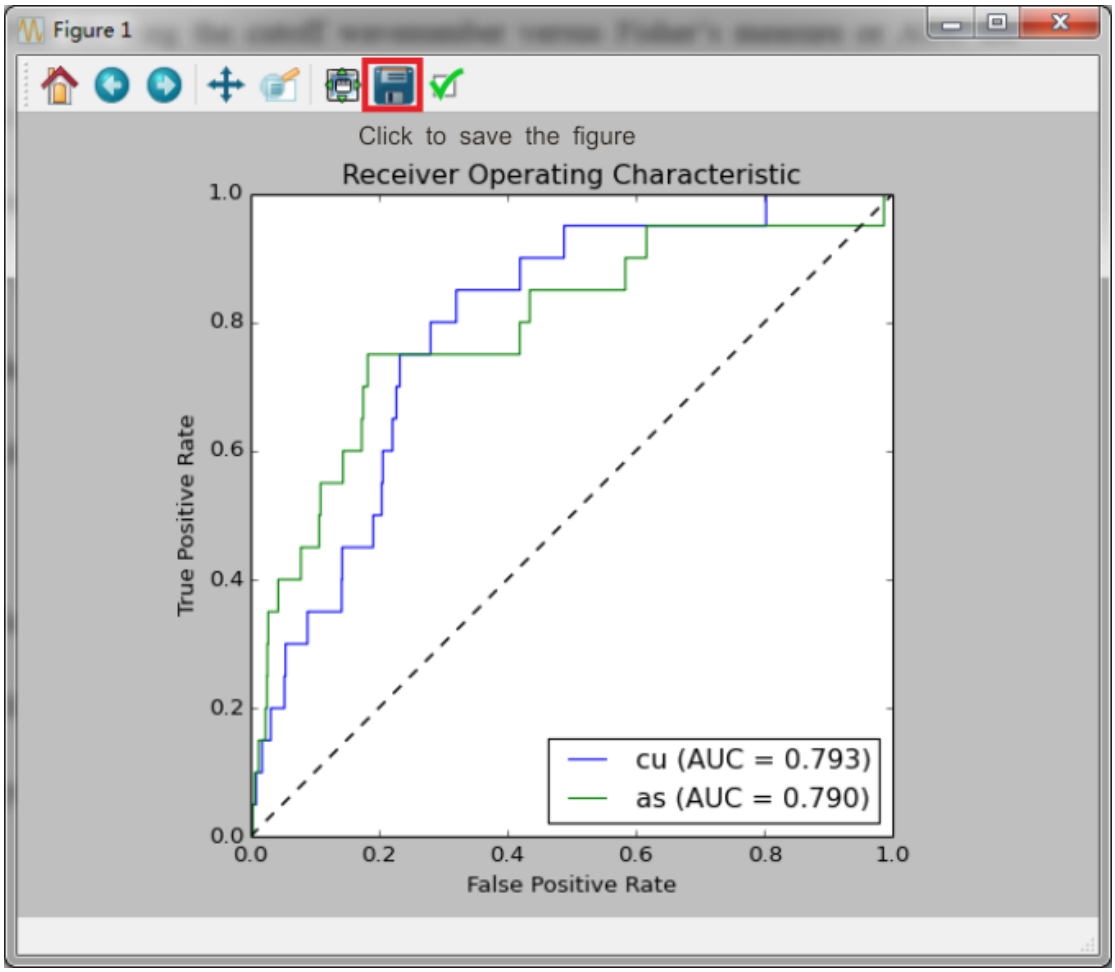

Figure 3. the output results**.**

Step 2. Click the icon in red rectangle to save the figure as raster image or vector graphics.

3.2 Optimal Filter

Left double click “Optimal Filter V0.1” tool in “opfilter.tbx” to open the tool, the Fig. 4 gives the input interface.

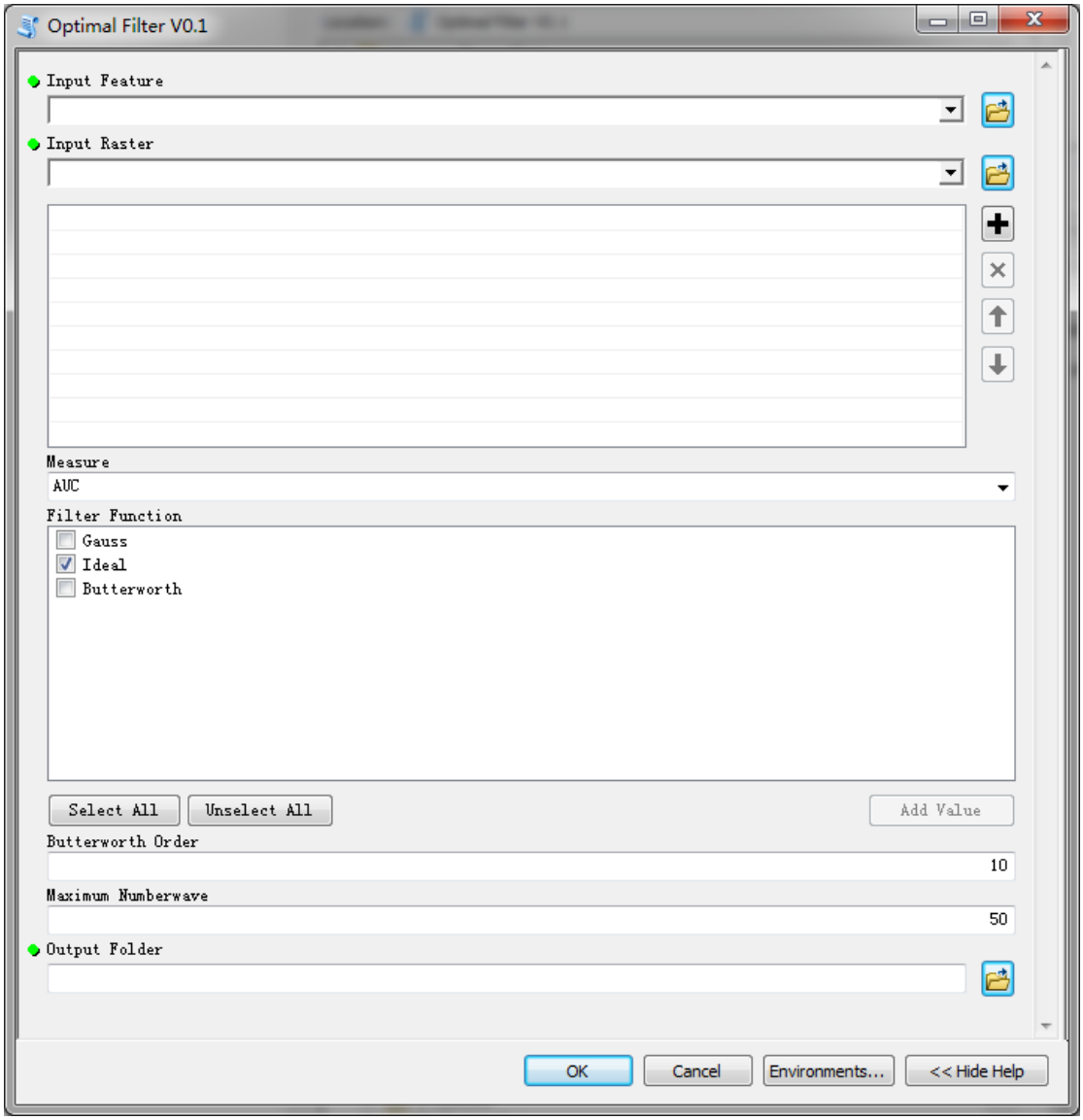

Figure 4. The input interface of optimal filtering.

Input Feature: Input objective feature layer or file.
Input Raster: Input rasters, these rasters should have same properties.
Measure: AUC or Fisher’s measure.
Filter Function: Multiple Choice.
Butterworth Order: The order of Butterworth function, only can work when Butterworth function is chosen.
Output Folder: The folder saving output rasters.

Step 1. Fill all parameters and click “OK”, the next interface (Fig. 5) will occur after a few minutes.

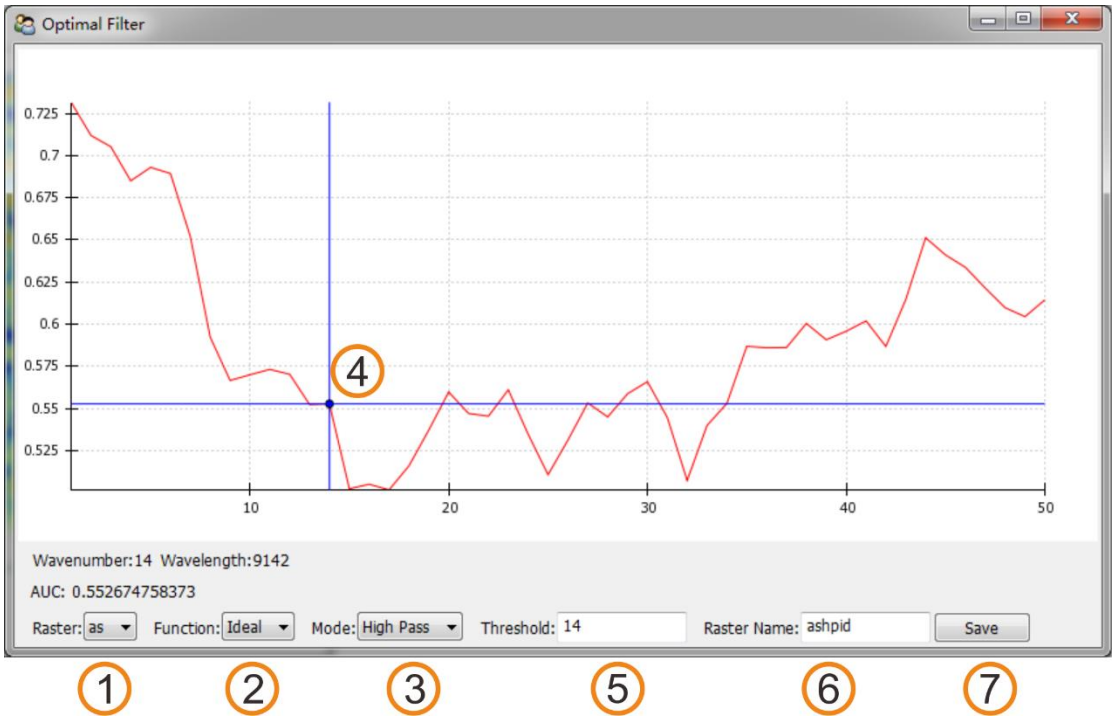

Figure 5. The interface of filtering

(1) Raster: The objective raster.
(2) Function: The filter function.
(3) Mode: High-pass or low-pass mode.
(4) The intersection: Can be dragged by left click to determine the threshold of wavenumber.
(5) Threshold: The threshold of wavenumber.
(6) Raster Name: The name of output raster, less than 13 characters.
(7) Save button: Save the result of filtering in memory.

Step 2. Choose filtering model (Raster, Function and Threshold), adjust cut-off wavenumber, fill Raster Name, and then click “Save”. When all results have been saved in memory, close the window, the program will save the results into rasters.

Notice

The optimal filter tool seems has cooldown time, which means an error will occur when the tool is applied twice time in a short time (a few minutes). You can try the tool third time to solve the problem. That bug might be from ArcGIS and I don’t know how to fix it.
Please feel free to report bugs or problems to sinixyang@gmail.com

Citing our works

If our work contributes to a project that leads to a scientific publication please acknowledge the work by citing the following literatures:

1. Yang, J. and Agterberg F. P., Cheng, Q., A novel filtering technique for enhancing mineralization associated geochemical and geophysical anomalies, Computers & Geosciences
2. Hunter, J.D., 2007. Matplotlib: A 2D Graphics Environment. Computing in Science & Engineering, 9(3): 90-95.doi:http://dx.doi.org/10.1109/MCSE.2007.55.
3. Pedregosa, F., Varoquaux, G., Varoquaux, l., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E., 2011. Scikit-learn: Machine Learning in Python. J. Mach. Learn. Res., 12: 2825-2830.
4. Walt, S.v.d., Colbert, S.C. and Varoquaux, G., 2011. The NumPy Array: A Structure for Efficient Numerical Computation. Computing in Science & Engineering, 13(2): 22-30.doi:http://dx.doi.org/10.1109/MCSE.2011.37.