Dan Higgins
9 Aug 2005

Actors For Use in ENM/GARP Workflows

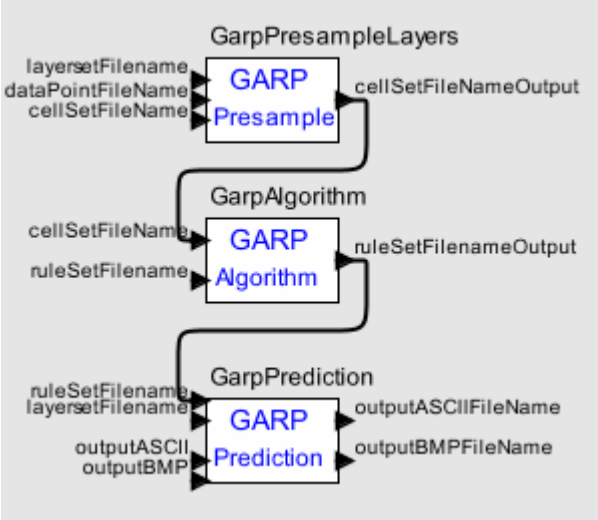
GARP Actors

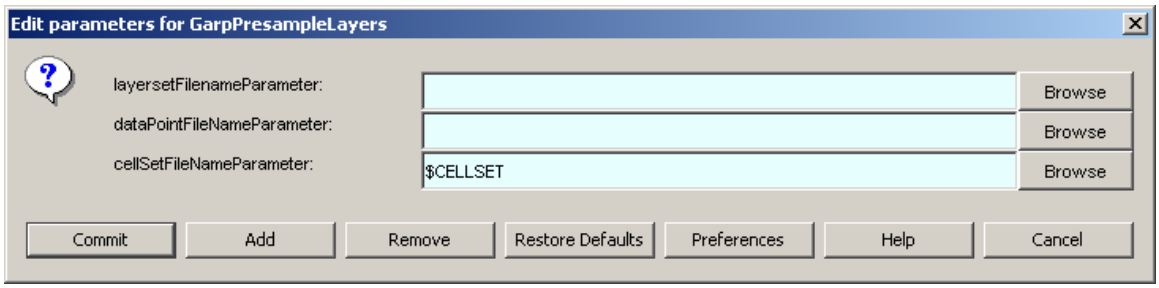
There are three basic GARP actors for predicting species locations, as indicated in Figure 1.

The GarpPresampleLayers has three input ports: the layersetFileName, the dataPointFileName, and the cellSetFileName. The layersetFileName is the name of an XML document, which summarizes all the spatial layer information. The dataPointFile is a file with known species location in longitude,latitude pairs. The cellSetFileName is just the name to be given to the output file. Note that the input ports need not be connected; values can be specified as internal parameters.

The GarpPresample actor will usually be connected to the GarpAlgorithm actor. The output of the two actors is the ruleSetFilenameOutput. This ruleset file can be saved and then used with new (or old) sets of spatial data to predict species locations.

The GarpPrediction actor takes ruleset and spatial layer information (along with desired output file name) as inputs and actually creates output spatial grids (maps) in both ascii raster and BMP file formats.





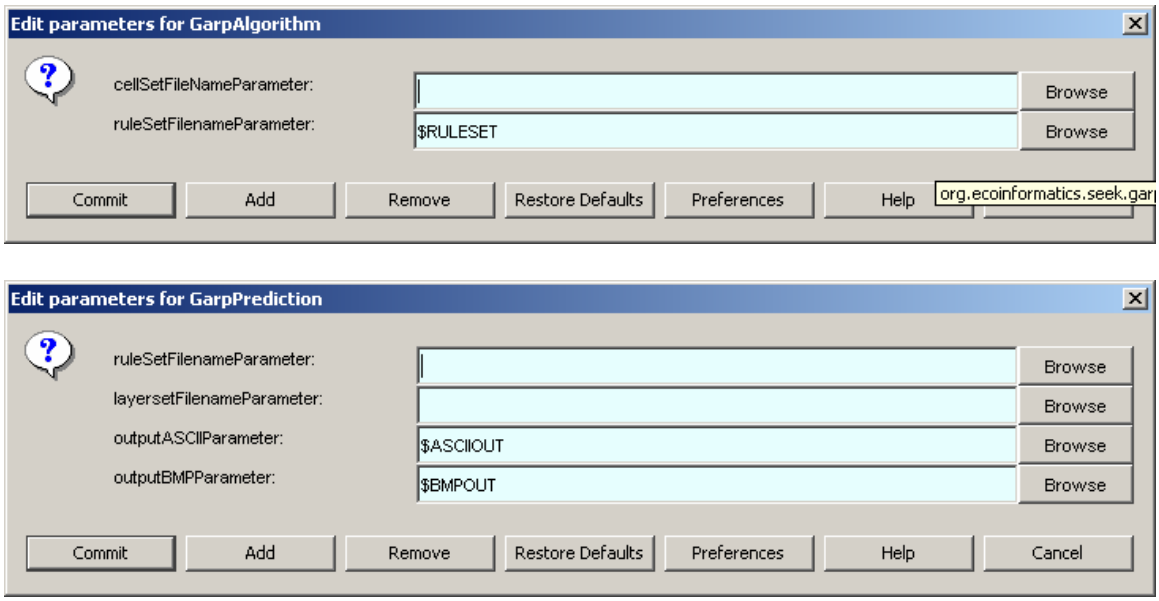
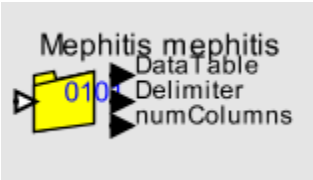


Figure 1 – GARP Actors and Parameters

Darwin Core Data Source

The primary source of existing species location data is the DigiR network of Darwin Core data from a variety of museums. These sources are obtained by doing an ecogrid data search on species name. A typical actor icon resulting from such a search is shown in Figure 2. Note that the output can be configured in different ways, so the output ports may vary. This source is usually configured to return only the longitude, latitude known species location pairs.



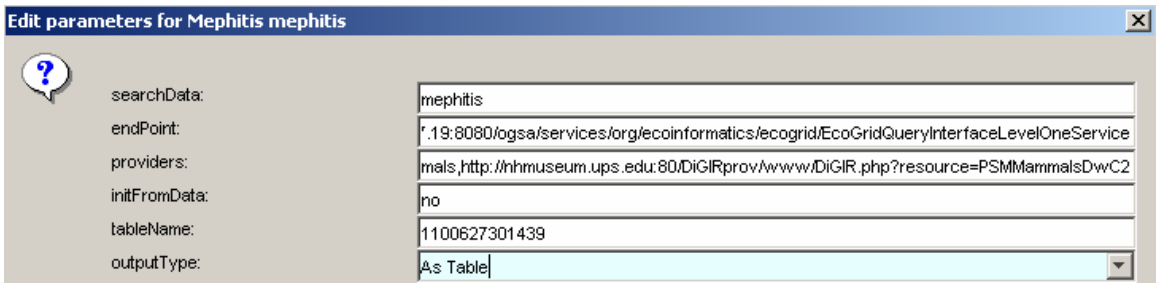
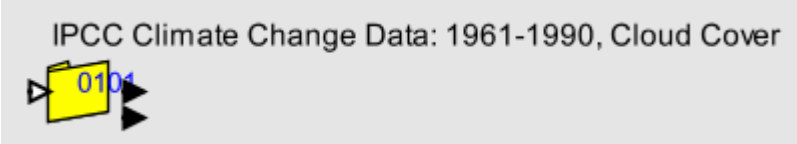
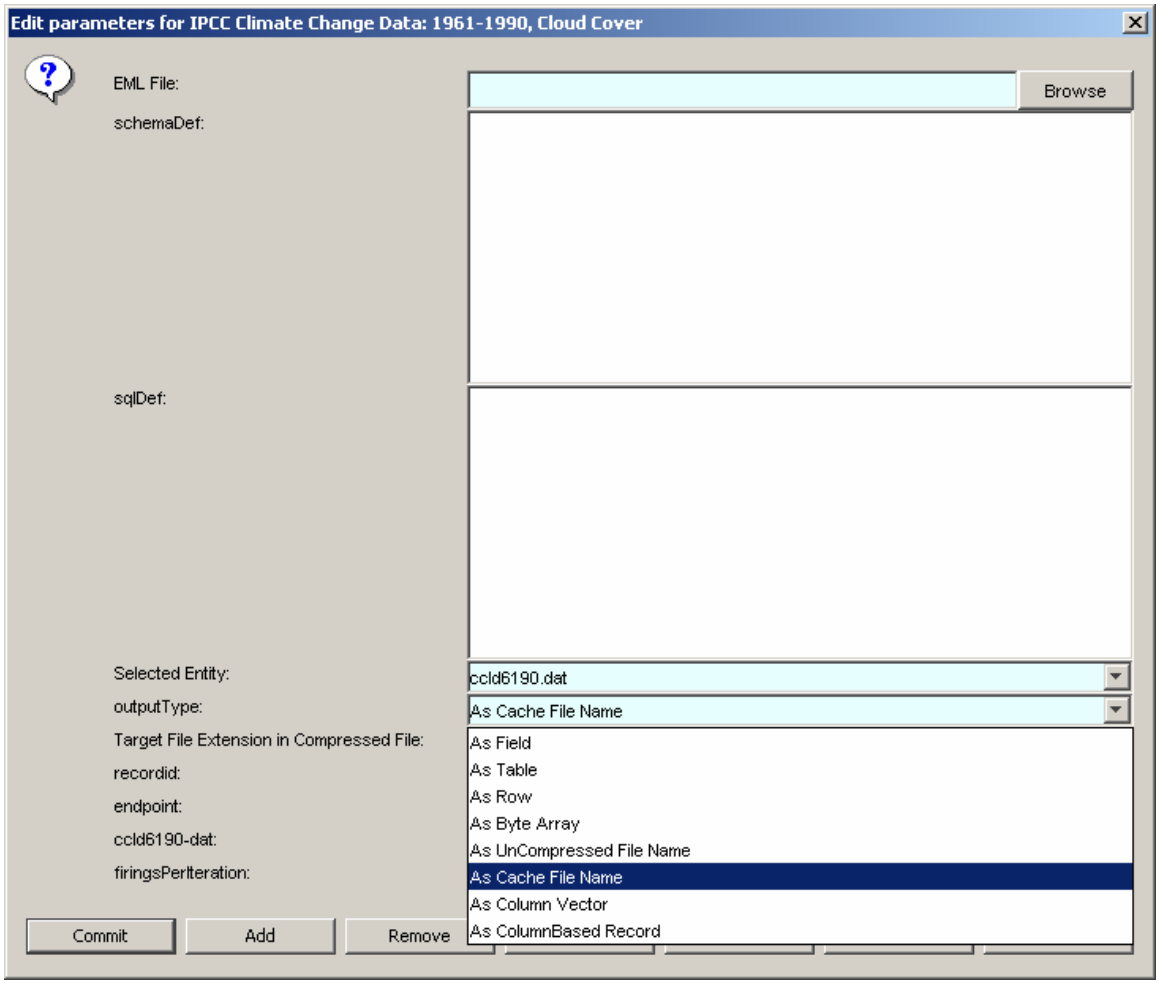


Figure 2 – Darwin Core Data Source – Output Type is ‘As Table’

EML 2 Data Source

Spatial layer data can be obtained from the ecogrid using the EML 2 DataSource, shown in Figure 3. There are a variety of data outputs available, as indicated in the outputType parameter. EML 2 Data Sources have extensive associated metadata. This can be viewed by right clicking to bring up a popup menu and selecting “Get Metadata”. A typical metadata display is shown below as part of Figure 3.





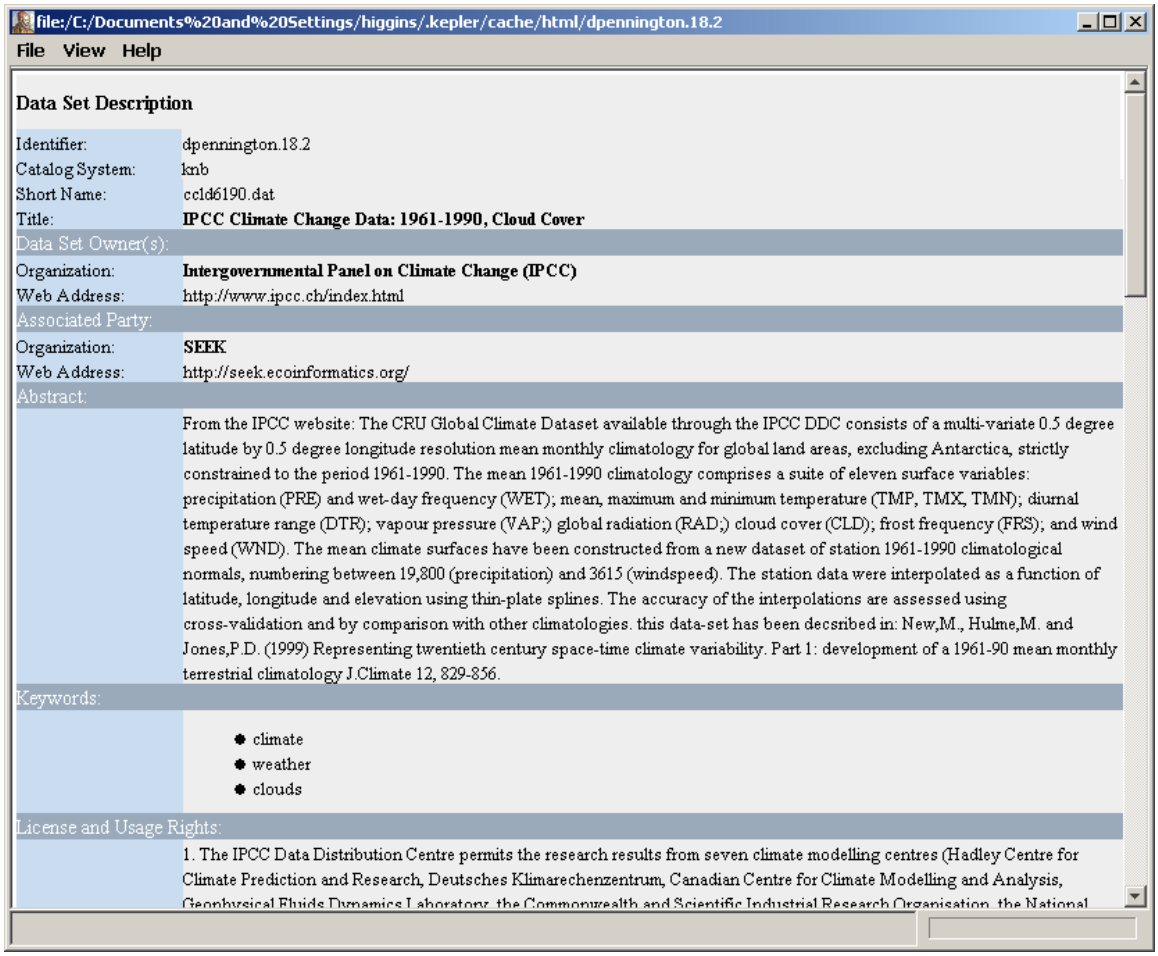


Figure 3 – EML200DataSource

GridRescaler

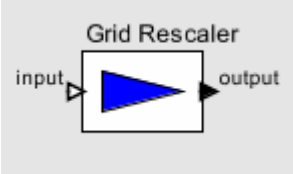
This actor converts one (or more) ASC raster grid files into other ASC grid files with different extent and cell spacing. Both input and output are file names of ASC grid files. As indicated in the Edit parameters screen, one must set the x and y values for the lower left corner of the output grid (usually as longitude and latitude), the cell size, and the number of desired rows and columns.

The 'algorithm' parameter refers to how output grid values are calculated from the input grid values. There are currently two choices: 'nearest neighbor' or 'inverse distance weighted'. The first simply finds the nearest cell in the input grid and uses its value; the second interpolates using a weighting based in the inverse distance to surrounding cells.

Note that multiple input file names can be attached to the input port in able to rescale multiple raster grids at once. This results in a sequence of output filenames equal to the number of inputs. If the output file name parameter is empty, then the output file name is just the input file name plus the suffix "`.outn`" (where n is an index). If the

output file name parameter is a directory, all output is put in that directory; otherwise it is put in the same directory as the input file.

Note also the 'use Existing File' checkbox in the Edit parameters dialog. If that checkbox is selected, then the actor will see if a file with the output file name already exists. If it does, then the actor skips all actions except for returning the file name. This can be used to avoid re-doing lengthy re-scaling calculations that have already be completed. If the checkbox is not selected, any existing output file with the same name will simply be overwritten.



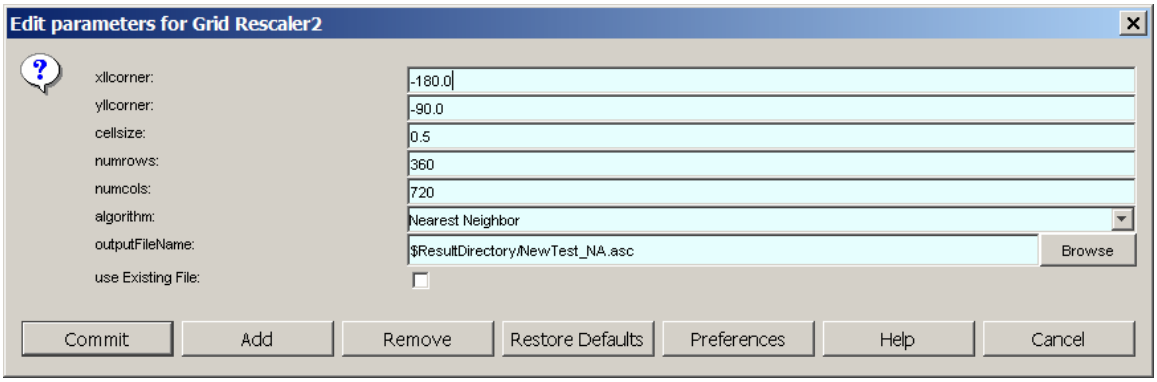


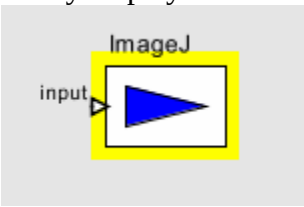
Figure 4

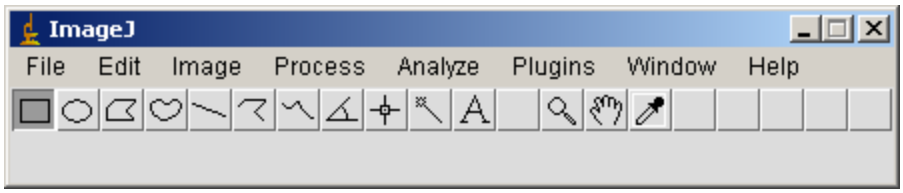
ImageJ

ImageJ is an image processing system created by the NIH (http://rsb.info.nih.gov/ij/). It has been added to Kepler for displaying GIS raster grids (and other images) and perhaps providing means for further image processing. It will display many types of images by just connecting the file name to the input.

For additional information, see the ImageJ Help menu or refer to http://rsb.info.nih.gov/ij/.

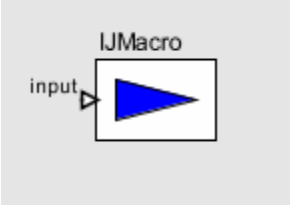
Note that the ImageJ window shown below will stay open even after the Kepler workflow that opened it is closed. This is to allow it to be used for additional processing of any displayed.

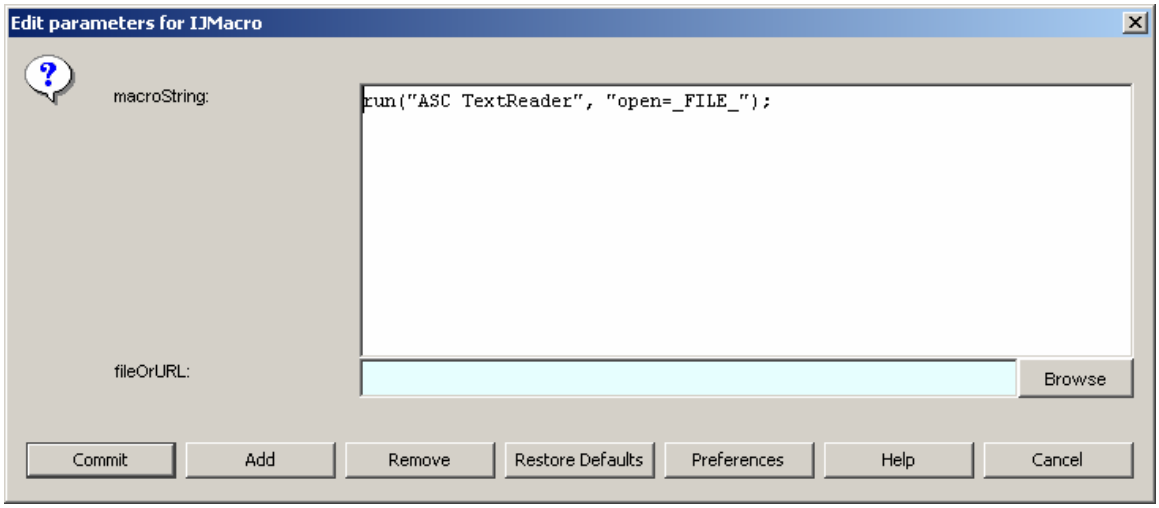




IJMacro

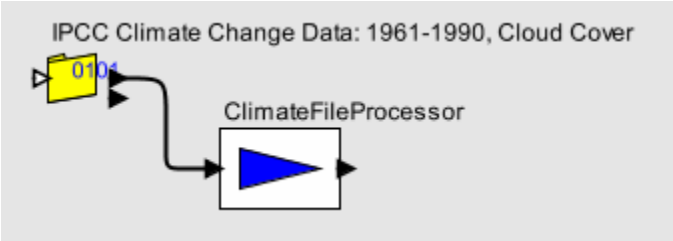
This is the ImageJ actor modified to run ImageJ (http://rsb.info.nih.gov/ij/) macros. This allows a variety of actions that can be run using the macro language of ImageJ (which has an macro recorder built-in.) In particular, this example uses this variation to display *.ASC ascii grid files as images. Note that if the expression "_FILE_" is included in this string, it is replaced by the input parameter or fileOrUrl parameter string, enabling the insertion of the input image file.

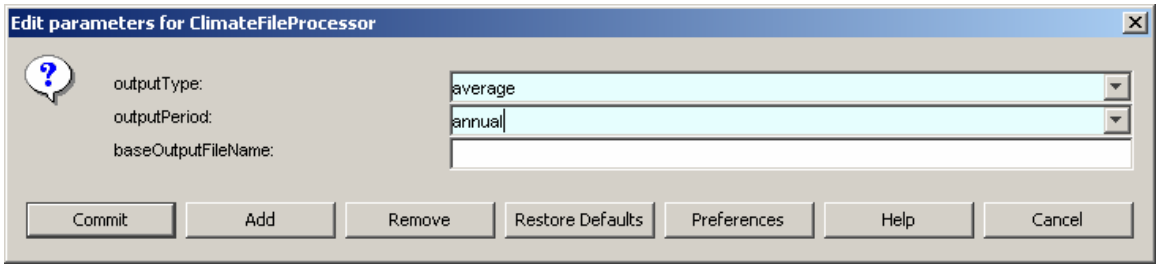




ClimateFileProcessor

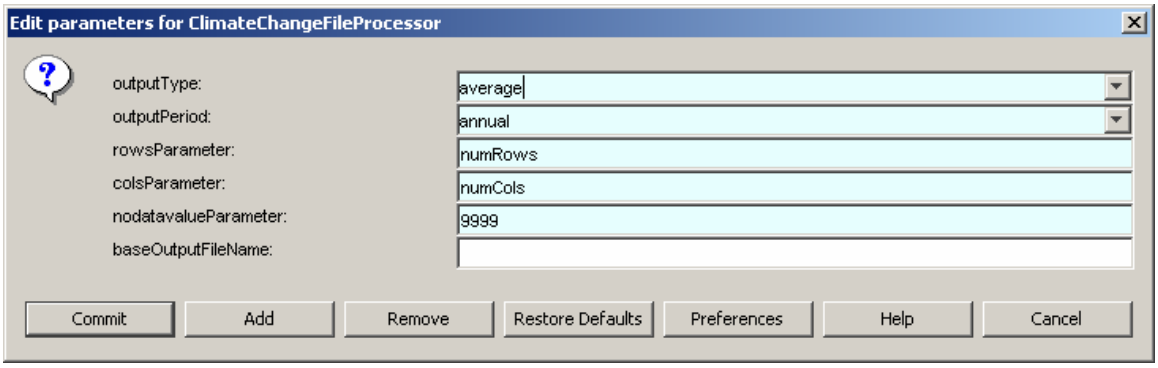
This actor is designed for specific use with IPCC historical climate information files. It creates ASC grids for the chosen parameters. IPCC data is stored in a file that has data organized by months. This actor processes that data to return file(s) that are seasonal (fall, winter, summer, or spring) and/or annual. The 'outputPeriod' parameter sets the season. Minimum, maximum, or average values can be placed in the output file. The type of values is set with the 'outputType' parameter. If the 'baseOutputFileName' parameter is left empty, the resulting output file is placed in the same directory as the input IPCC file, with some text added to the filename to indicate its type and period. Otherwise, the text in 'baseOutputFileName' is assumed to be the base for the output file path (and text indicating type and period is added). This allows the output to be sent to arbitrary local locations. In any case, the resulting output is an ASC grid filename.





ClimateChangeFileProcessor

This actor is very similar to the ClimateFileProcessor actor except it is designed to work with predicted climate change layers which have different formats than that the historical (1961-1990) IPCC spatial grids.



<u>AscToRaw</u>

This actor can take an array of ASC grid file names and create corresponding RAW files (the format needed by GARP) and output an XML files (*.dxl) that summarizes the layers.

Note that an ASC file is a text file, which has six header lines defining resolution and spatial position. These header lines, followed by one row of ASCII space-delimited values per line. Typical header lines are:

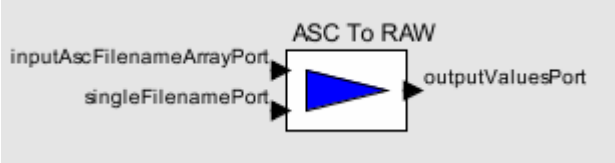
```
* ncols         720
* nrows         360
* xllcorner     -180.0
* yllcorner     -90
* cellsize      0.5
* NODATA_value  -9999
```

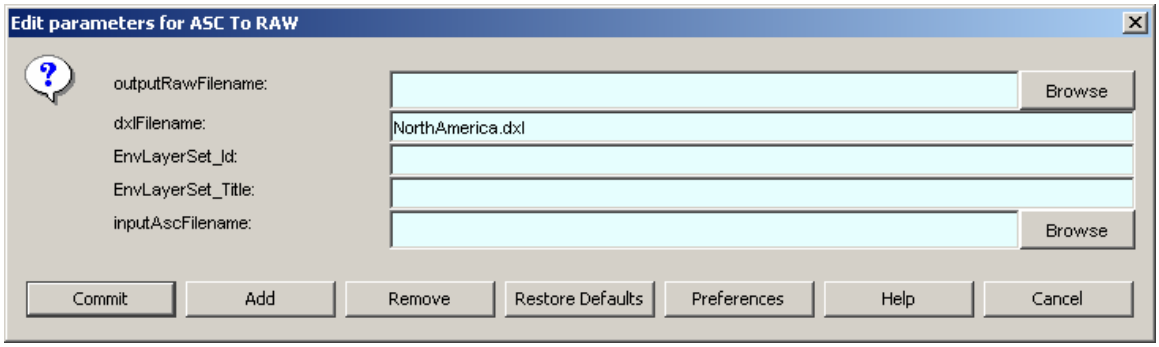
The raw file is just a sequence of row x col bytes, scaled appropriately.

A single ASC file can be converted to a RAW file by just connecting the ASC file name to the ‘singleFilenamePort’. The corresponding RAW file name will appear on the ‘outputValuesPort’.

In many cases, however, the input will be an array of spatial layer file names ASC format) and this array will be connect to the ‘inputAscFilenameArrayPort’. It is assumed that all the layers have the same extent and resolution. When an array of layers is input,

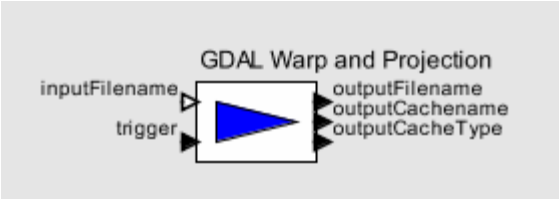
an XML file (*.dxl) is created which summarized the array of layers in a format used by GARP. In most cases, the only required parameter is the dxlFilename, which can be used to characterize the layer set.

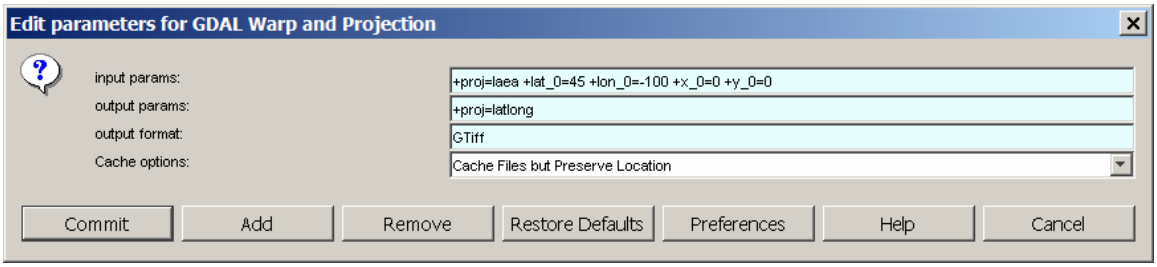




GDAL Warp and Projection

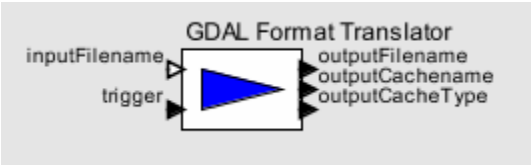
GDAL is an open source software package designed to read, write, and manipulate the wide variety of GIS raster grid files that are in common use. This actor uses a JNI connection to the GDAL C code. In particular, it can read various file types and apply spatial projections to the geo-referenced data. Details of GDAL are documented at http://www.gdal.org/index.html. In particular, the gdalwarp utility is described at http://www.gdal.org/gdal_utilities.html#gdalwarp. Currently, a Macintosh version of this actor is not available.

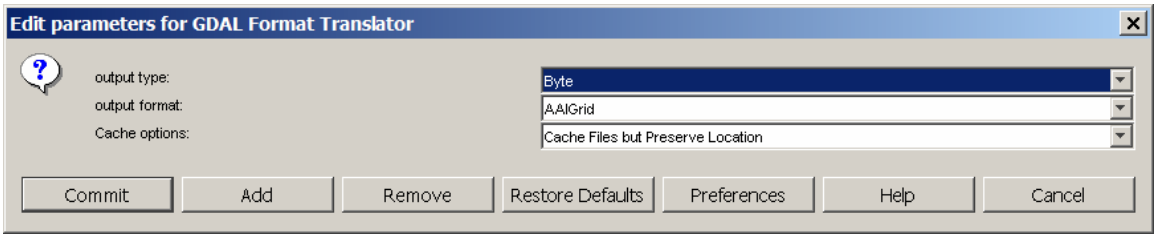




GDAL Format Translator

The GDAL Format Translator translates a spatially referenced raster file from one format to another. Refer to http://www.gdal.org/gdal_utilities.html for further information. Currently, a Macintosh version of this actor is not available.





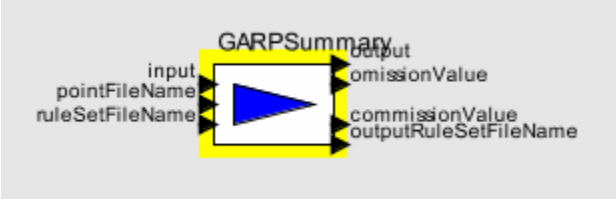
GARPSummary

GARP predictions of species distributions are probabilistic and will thus predict somewhat different spatial distributions of species every time a calculation is executed. The GARPSummary actor generates measures of how well a given run does in predicting distributions, based on a ‘test set’ of known species locations. The specific parameters used to evaluate GARP performance are called ‘omission’ and ‘commission’.

Omission is the fraction of test set points that are not predicted by the calculation. Commission, when we don't have real absence points (our case), is the proportion of area predicted present with regard to the total area of interest, not counting masked pixels.

The ‘input’ port should be connected to a file name for an ASC grid file such as the output of the GARPPrediction actor. The ‘pointFileName’ port takes a file name for a test set of species location in a (longitude, latitude) format with one location per line of a tab-delimited text file. The ‘ruleSetFileName’ input is just used to provide the ruleset file name which is passed to the output port named ‘outputRuleSetFileName’. (The ruleSetFileName is needed when the omission and commission values are used to pick the ‘best’ rulesets so that predictions can be applied for future climate predictions.)

Results can be obtained in several forms. The ‘output’ generates a string with tab-delimited omission, commission, and ruleSetName values. The numerical values of omission and commission are output as doubles on output ports of the corresponding names, and the ouputRuleSetFileName is copied from the input as a string.

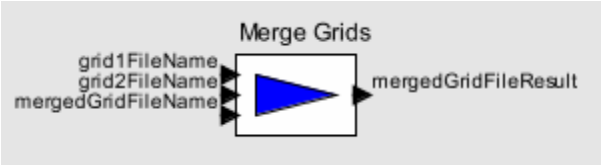


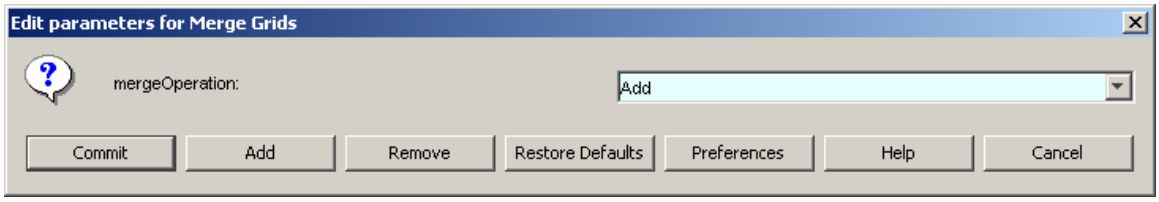
MergeGrids

The purpose of this actor is to 'merge' two ASC grids. The precise meaning of 'merge' will depend on the 'merge' operator. One example is the combination of 2 grids into a new grid whose extent is a rectangle that includes both input bounding box rectangles, averaging values from both inputs. Simple math operations (add, subtract) are other examples of the ‘merge’ operator.

Order of the input grids may be significant(e.g. for subtraction). Extent of the output will always include the combined extent of the inputs, but the cell size will match that of the first grid.

One use of this actor is to combine several regions into a large region such as combining a grid covering North America with one for South America to create a raster grid for the western hemisphere.



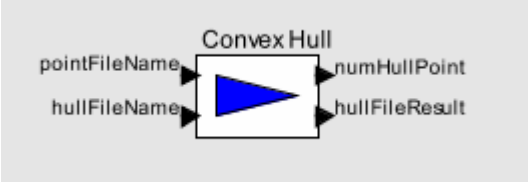


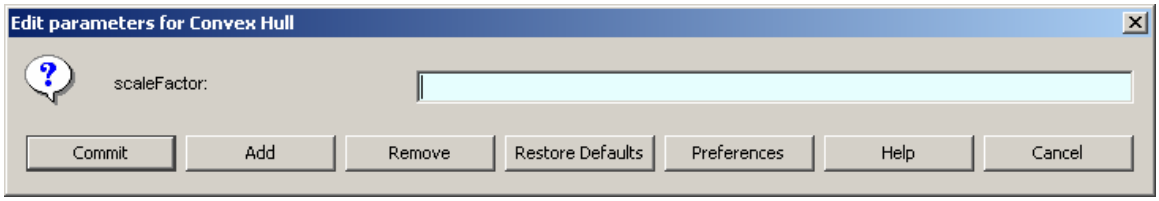
ConvexHull

The ‘Convex Hull’ actor takes a set of 2-D points and calculates those points that define the minimum polygon that encloses all of the input points. (One way to think about this is to consider the convex hull to be the region inside a tight rubber band placed about all the points; the output is the set of points that hold the rubber band from collapsing further, with all points inside the rubber band.) A convex hull in a ‘minimum’ bounding region.

The ‘pointFileName’ is a text file with tab-delimited (x,y) values on each line. (In most ENM cases, this is the species location set of (longitude, latitude) values. ‘hullFileName’ is just the name to be given to the subset of input points that define the convex hull region. The output ‘hullFileResult’ is a new point list with only those points that define the convex hull polygon, while ‘numHullPoint’ is the number of such points.

The parameter ‘scaleFactor’ is a multiplier that can be used to make the Convex Hull bigger or smaller. The polygon determined by bounding points is scaled linearly by this factor with the center of the bounding box kept at the same location. If the ‘scaleFactor’ parameter is left empty, a default value of 1.0 is used.

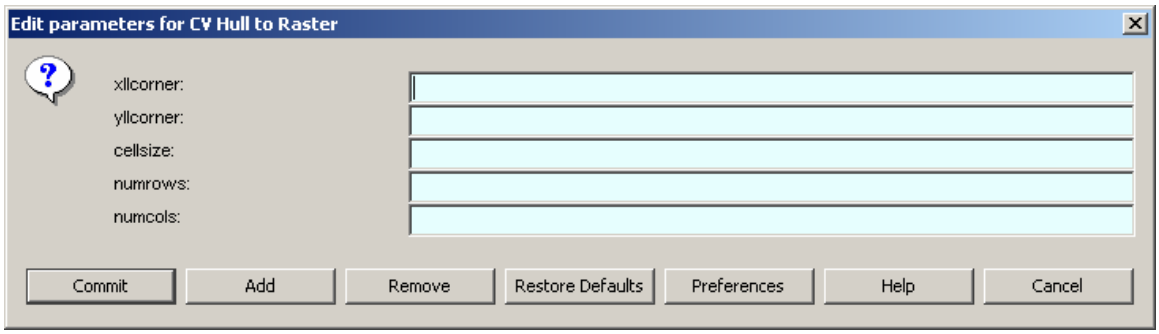




CV Hull to Raster

The Convex Hull is used to define a minimal region including all known occurrence points of a species. One use is to limit the region where ENM calculations occur by creating a 'mask' grid. This actor thus takes the Convex Hull points and creates an ASC raster grid file with "NO_DATA" values for all points outside the convex hull region. Any points inside the convex hull are set to a value of '1'.

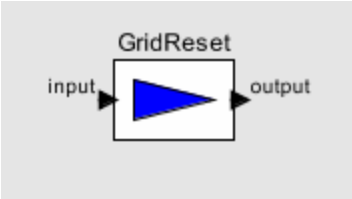
Note that the location, size, and resolution of the resulting ASC grid can be set using the parameters `xllcorner`, `yllcorner`, `cellsize`, `numrows`, and `numcols`. If these parameters are not set, the default grid is set to the minimum bounding box enclosing the convex hull.

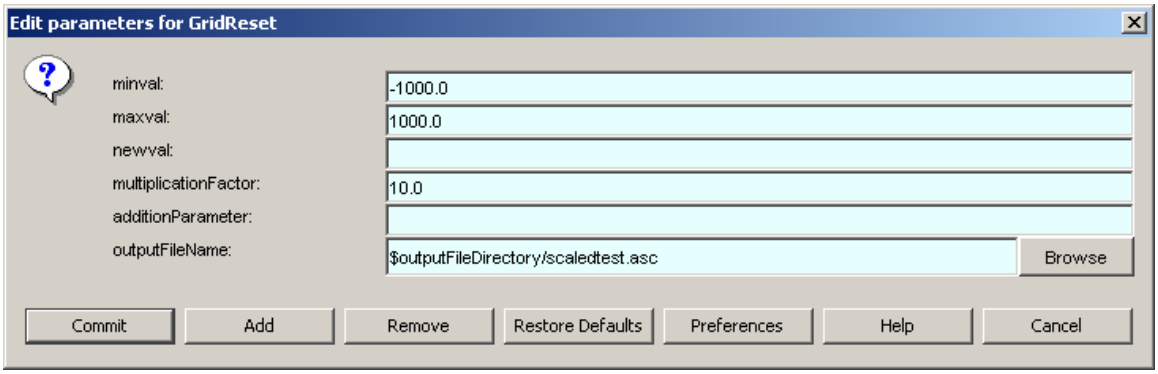


Grid Reset

The Grid Reset actor is used for resetting values of cells in an ASC grid file without making any changes is cell size or extent of the grid. The input is the name of the input ASC file and the output is the name of the output file.

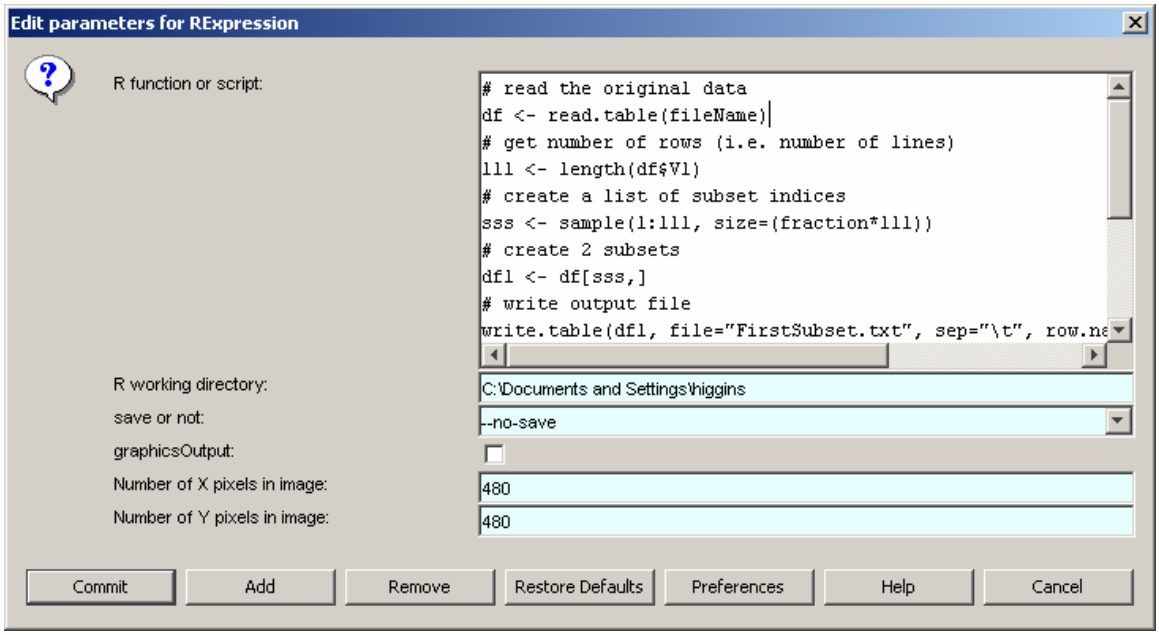
The parameters '`minval`' and '`maxval`' set the range of existing cell values that should be considered. If '`newval`' is specified, all exiting values in that range are set to '`newval`'. If '`newval`' is blank, the values in the specified range are multiplied by '`multiplicationFactor`' and '`additionParameter`' is added to the result. [The default for '`multiplicationFactor`' is `1.0`.]





Custom RExpression Actor – Random Training/Testing Points

This is a custom version of the RExpression actor that uses a simple R script to subdivide a location point file into two subset – one to be used for 'training' a program like GARP and the other to be used for 'testing' and evaluating the result. The actual R script is shown below. The R "sample" function is used to randomly sample the points. The parameter '`fraction`' is set to 0.5 (`fraction <- 0.5`) which means that the original set of points is divided into two approximately equal subsets. (Edit the value to change this.)



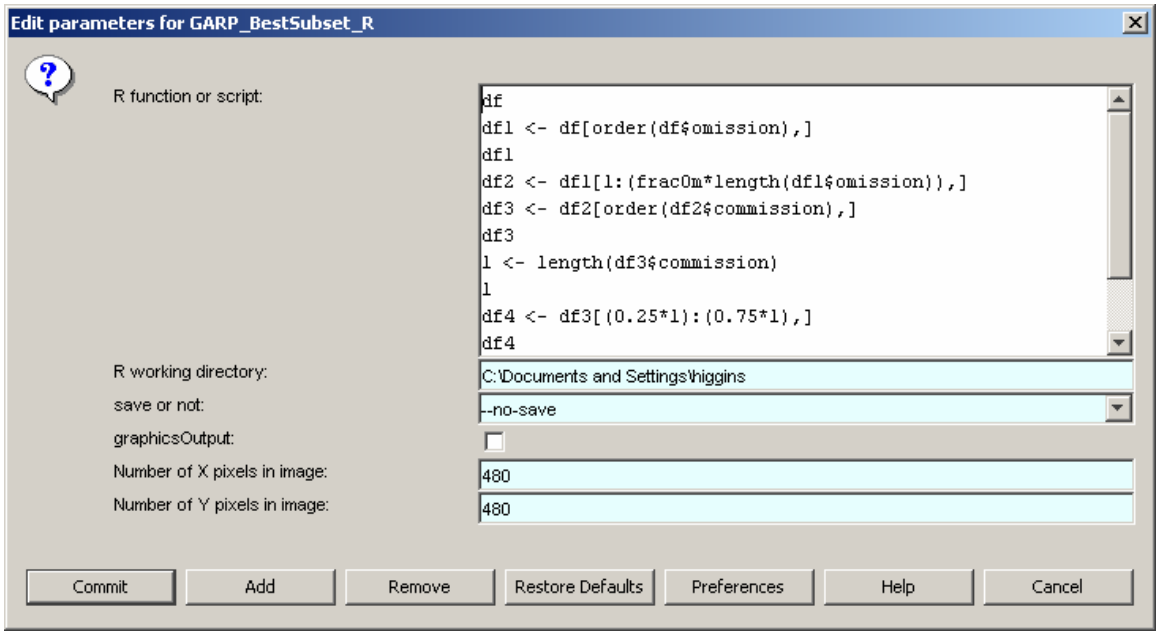
Script:

```
# read the original data
df <- read.table(fileName)
# get number of rows (i.e. number of lines)
lll <- length(df$V1)
# create a list of subset indices
sss <- sample(1:lll, size=(fraction*lll))
# create 2 subsets
df1 <- df[sss,]
# write output file
write.table(df1, file="FirstSubset.txt", sep="\t", row.names=FALSE,
col.names=FALSE)
trainingLocations <- paste(getwd(),"/FirstSubset.txt",sep="")
trainingLocations
df2 <- df[-(sss),]
# write output file
write.table(df2, file="SecondSubset.txt", sep="\t", row.names=FALSE,
col.names=FALSE)
testingLocations <- paste(getwd(),"/SecondSubset.txt",sep="")
```

```
df1
df2
```

Custom RExpression Actor – Best Ruleset Selection

This custom RExpression actor takes the omission/commission table created by running GARP numerous times and finds the 'best' rulesets. It first sorts the results table by the value of '`commission`' with smaller values first. It then takes the top 10% of those and then sorts by the value of '`omission`'. The middle half of the resulting table (0.25 to 0.75) is then returned.



Script:

```
df
df1 <- df[order(df$omission),]
df1
df2 <- df1[1:(fracOm*length(df1$omission)),]
df3 <- df2[order(df2$commission),]
df3
l <- length(df3$commission)
l
df4 <- df3[(0.25*l):(0.75*l),]
df4
cnt
vec <- as.vector(df4$ruleset)
bestRulesets <- vec[1:cnt]
```