ctools - Bug #2136

csresmap significance computation throws an error if model map contains zero entries

06/26/2017 04:37 PM - Eschbach Stefan

Status:	Closed	**Start date:**	06/26/2017
Priority:	Normal	**Due date:**	
Assigned To:	Cardenzana Josh	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	1.3.1		

Description

When choosing the SIGNIFICANCE option for csresmap, it often throws an error stating that it cannot compute the root of a negative value (see attached picture).

Josh looked into it and found the reason. When the value from the model map is 0, the computation goes wrong:

"When modelmap is 0, the division in the computation of 'logmap' returns a 0 (which it should to prevent infinities). However, now in the computation of 'signif_squared_map' you end up setting the value to '-self._resmap', which is a problem when the data map has a non-zero value in the same bin resulting in a negative number. The fix is to instead loop over the bins and fill the values with 0 if the modelmap is zero in that bin."

History

#1 - 06/27/2017 10:10 AM - Cardenzana Josh

- Status changed from New to Pull request

- % Done changed from 20 to 100

I've implemented a fix for the above. The algorithm was originally doing the following:

```
# Compute sign map
signmap = (self._resmap - modelmap).sign()

# Compute logarithm of map. For that we mask pixels
#logmap  = self._resmap/modelmap
logmap  = self._resmap.copy()   # Python 3.x kluge
logmap /= modelmap
for i in range(logmap.npix()):
    if logmap[i] > 0.0:
        logmap[i] = math.log(logmap[i])
    else:
        logmap[i] = 0.0

# Compute significance map
signif_squared_map  = (self._resmap*logmap) + modelmap  - self._resmap
signif_squared_map *= 2.0
unsigned_resmap     = signif_squared_map.sqrt()
self._resmap        = unsigned_resmap * signmap
```

which was having trouble when the model map value was zero. The algorithm runs as:

```
# Compute sign map
signmap = (self._resmap - modelmap).sign()

# Loop over every bin in the map
for i in range(self._resmap.npix()):

    # If the model value > 0.0 do the computation as normal
    model_val = modelmap[i]
    if model_val > 0.0:

        # If the data value is also > 0 then proceed as normal
        data_val = self._resmap[i]
```

```
            if data_val > 0.0:
                # Compute the significance^2 and save it to the map
                log_val = math.log(data_val/model_val)
                self._resmap[i] = (data_val*log_val) + model_val  - data_val

            # If the data value is less than zero, then compute the
            # reduced value of the above expression. This is necessary
            # to avoid computing log(0).
            else:
                self._resmap[i] = model_val

        # If the model value is zero, hard-code the significance to 0
        else:
            self._resmap[i] = 0.0

    # Compute significance map
    self._resmap *= 2.0
    self._resmap  = self._resmap.sqrt()
    self._resmap  = self._resmap * signmap
```

The computation of a 1500x500 pixel map ran in about 8 seconds with no errors.

Pull branches (from gitlab):
joshcardenzana/ctools: 2136-csresmap_significance_fix

#2 - 06/30/2017 04:12 PM - Knödlseder Jürgen

- Status changed from Pull request to Closed

- Target version set to 1.3.1

Merged into bugfix-1.3.1 branch.

Files

csresmap_error.png	59.9 KB	06/26/2017	Eschbach Stefan