

# MOC Usage in TOPCAT and STILTS

Mark Taylor (Bristol)

IVOA Interop  
Heidelberg

14 May 2013

`$Id: moc.tex,v 1.7 2013/05/09 12:31:19 mbt Exp $`

# Outline

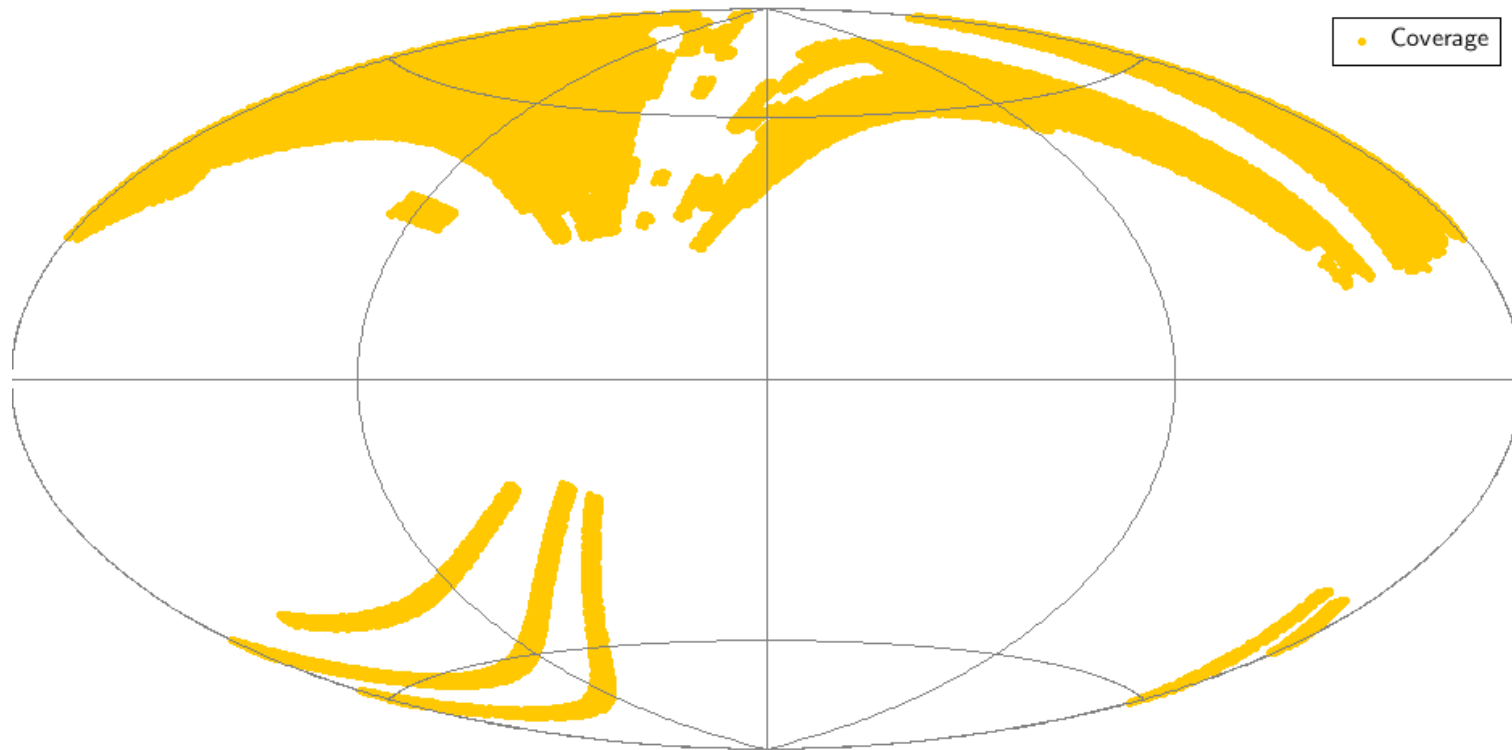
## MOC usage in TOPCAT and STILTS:

- Use CDS MOC coverage service to eliminate redundant queries in multi-cone
- New expression language function `inMoc` to query coverage
- New STILTS command `pixfoot` generates a MOC FITS file from a list of positions

# Multi-Cone Coverage

## Eliminate unnecessary cone-search queries in multi-cone

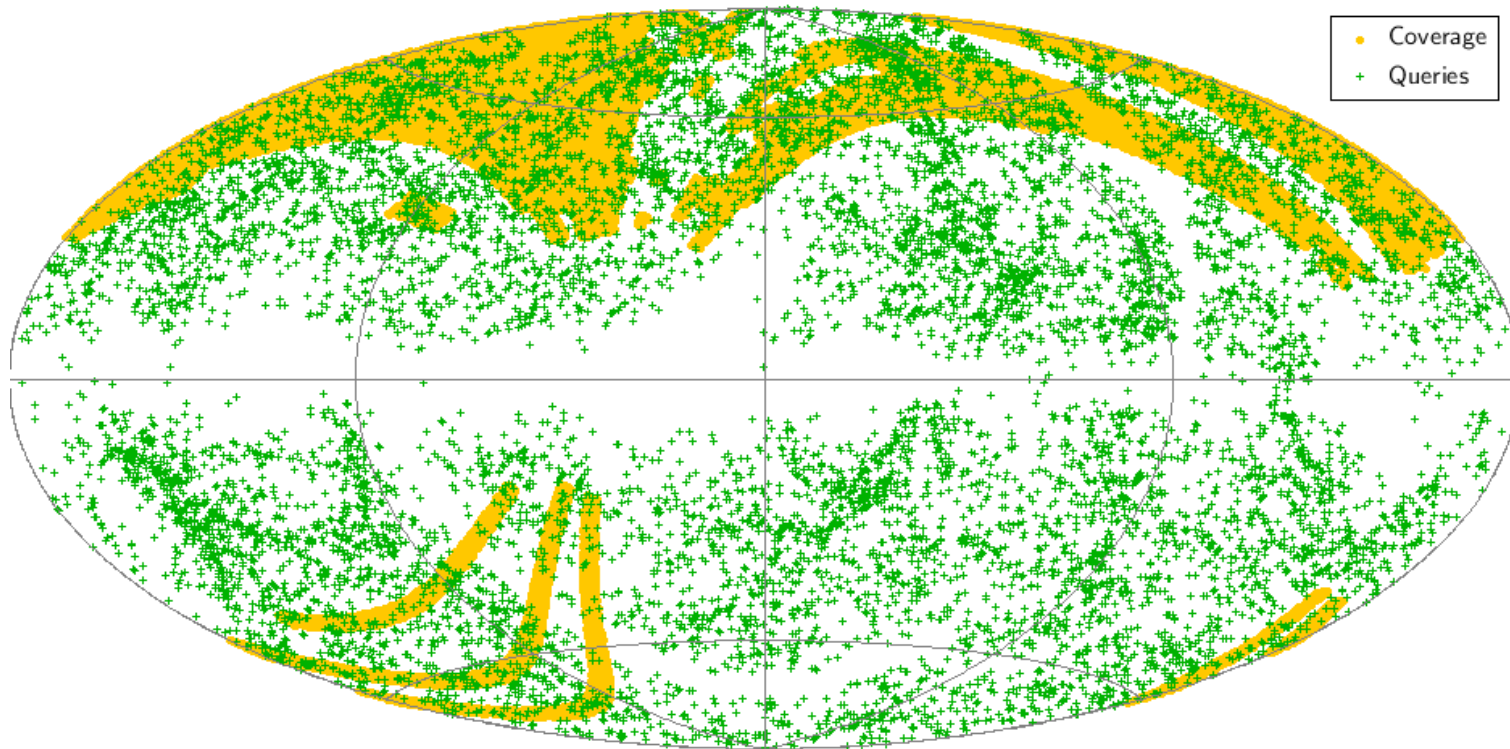
- Multi-cone normally has to make one cone search query for each point
- With coverage information (MOC), can avoid queries known to be outside remote catalogue coverage
- Only works for CDS (non-standard MOC service)



# Multi-Cone Coverage

## Eliminate unnecessary cone-search queries in multi-cone

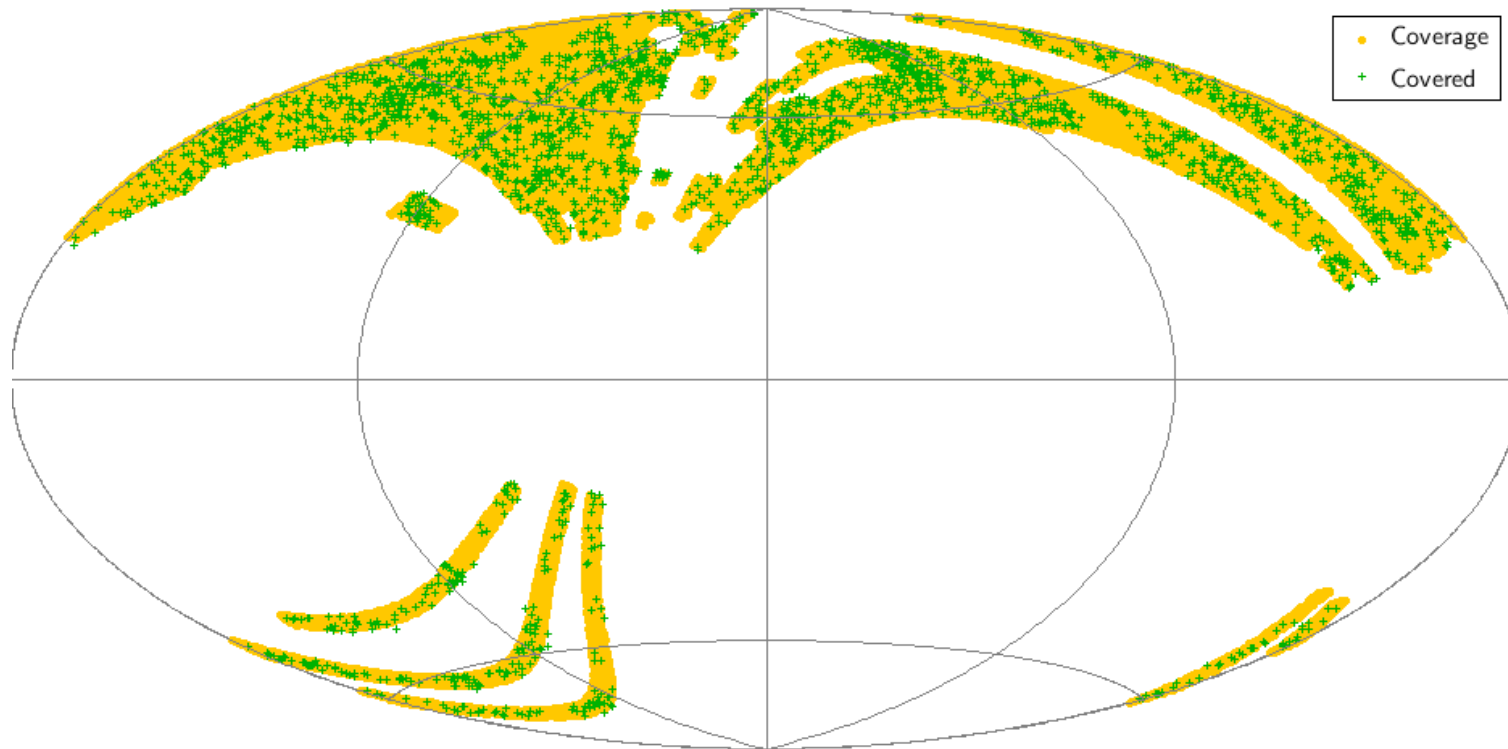
- Multi-cone normally has to make one cone search query for each point
- With coverage information (MOC), can avoid queries known to be outside remote catalogue coverage
- Only works for CDS (non-standard MOC service)



# Multi-Cone Coverage

## Eliminate unnecessary cone-search queries in multi-cone

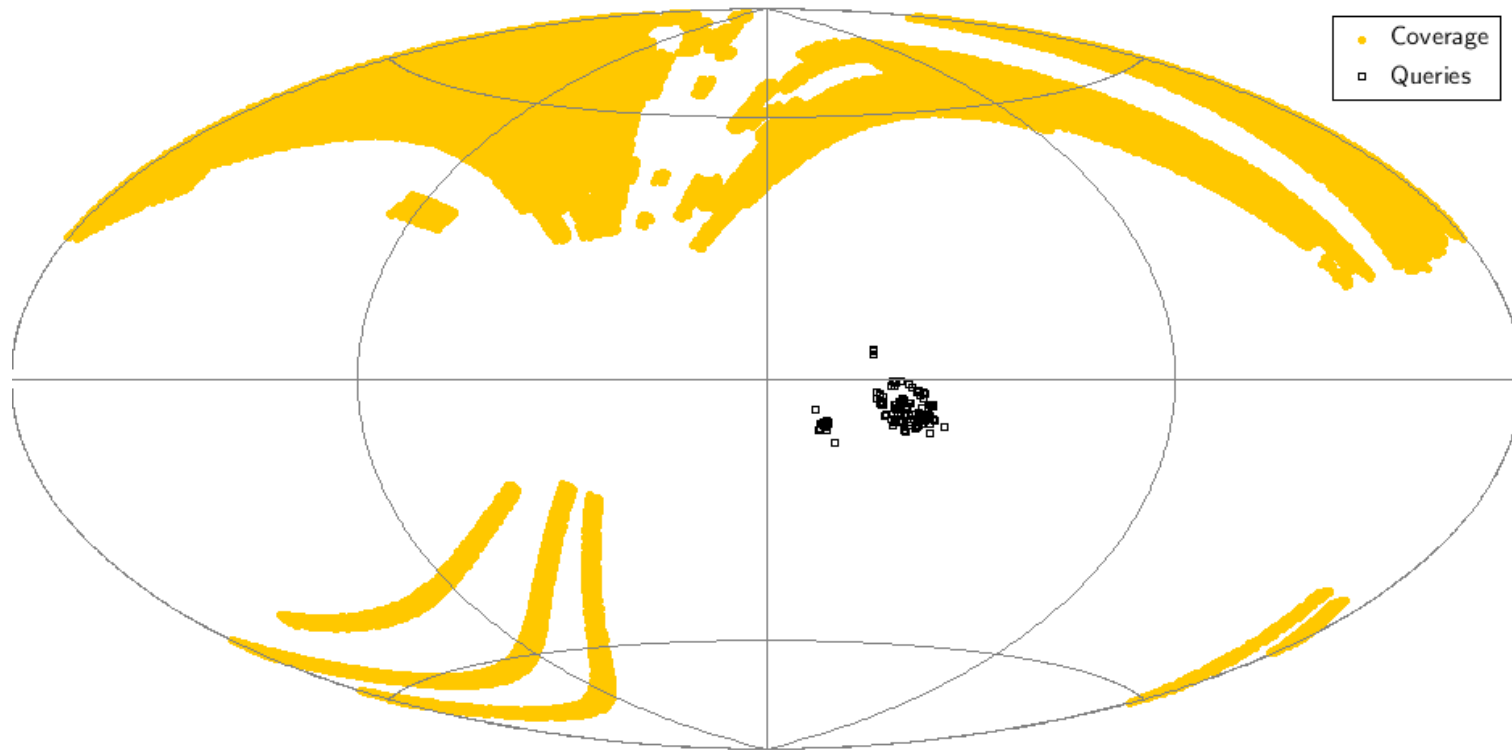
- Multi-cone normally has to make one cone search query for each point
- With coverage information (MOC), can avoid queries known to be outside remote catalogue coverage
- Only works for CDS (non-standard MOC service)



# Multi-Cone Coverage

## Eliminate unnecessary cone-search queries in multi-cone

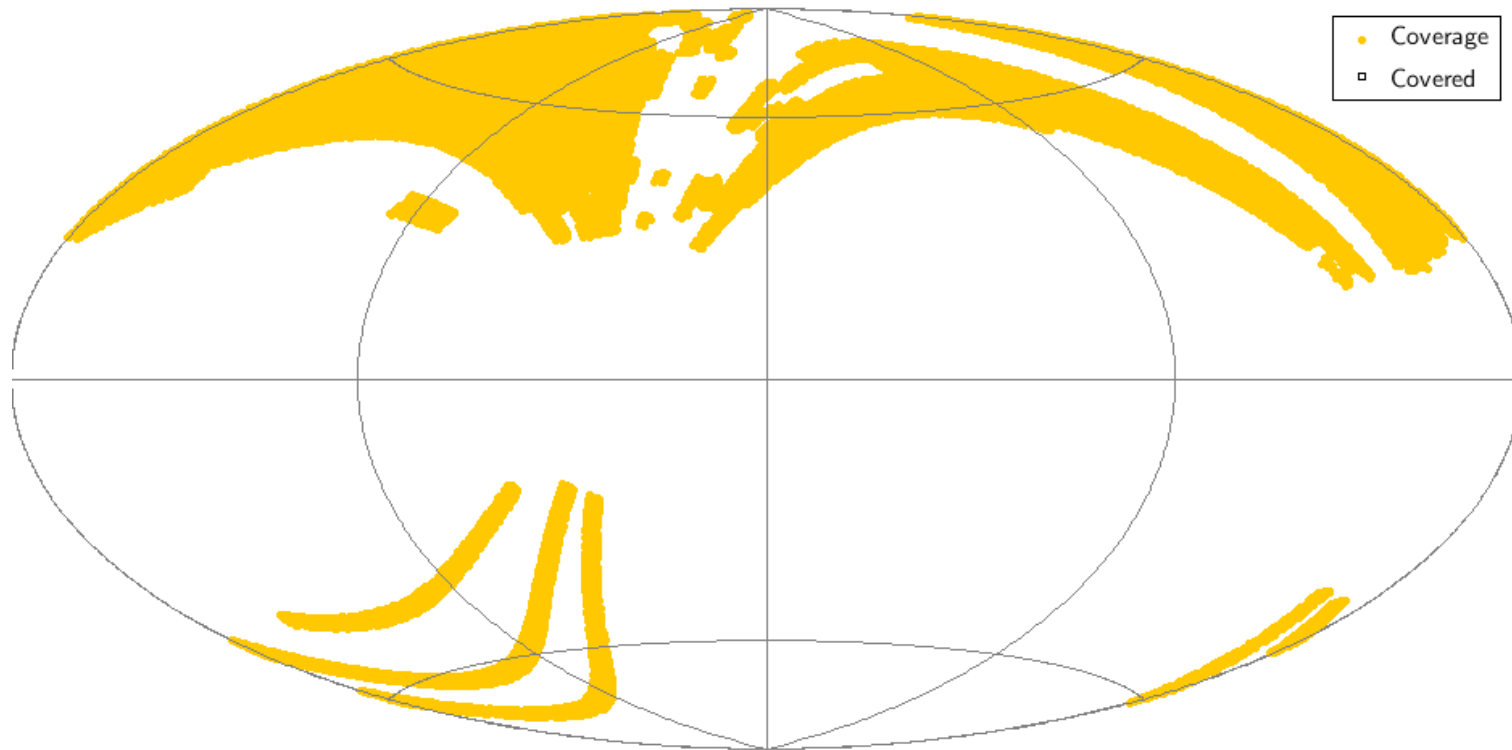
- Multi-cone normally has to make one cone search query for each point
- With coverage information (MOC), can avoid queries known to be outside remote catalogue coverage
- Only works for CDS (non-standard MOC service)



# Multi-Cone Coverage

## Eliminate unnecessary cone-search queries in multi-cone

- Multi-cone normally has to make one cone search query for each point
- With coverage information (MOC), can avoid queries known to be outside remote catalogue coverage
- Only works for CDS (non-standard MOC service)



# Multi-Cone with MOC

## TOPCAT multi-cone implementation:

- User selects local (TOPCAT table) and remote (cone search service) catalogues
- TOPCAT attempts to acquire MOC for cone search service
- If it gets a MOC:
  - ▷ For each local point, only query service if position is covered (else no match)
- If no MOC:
  - ▷ Have to query every point as before

## STILTS offers a similar option

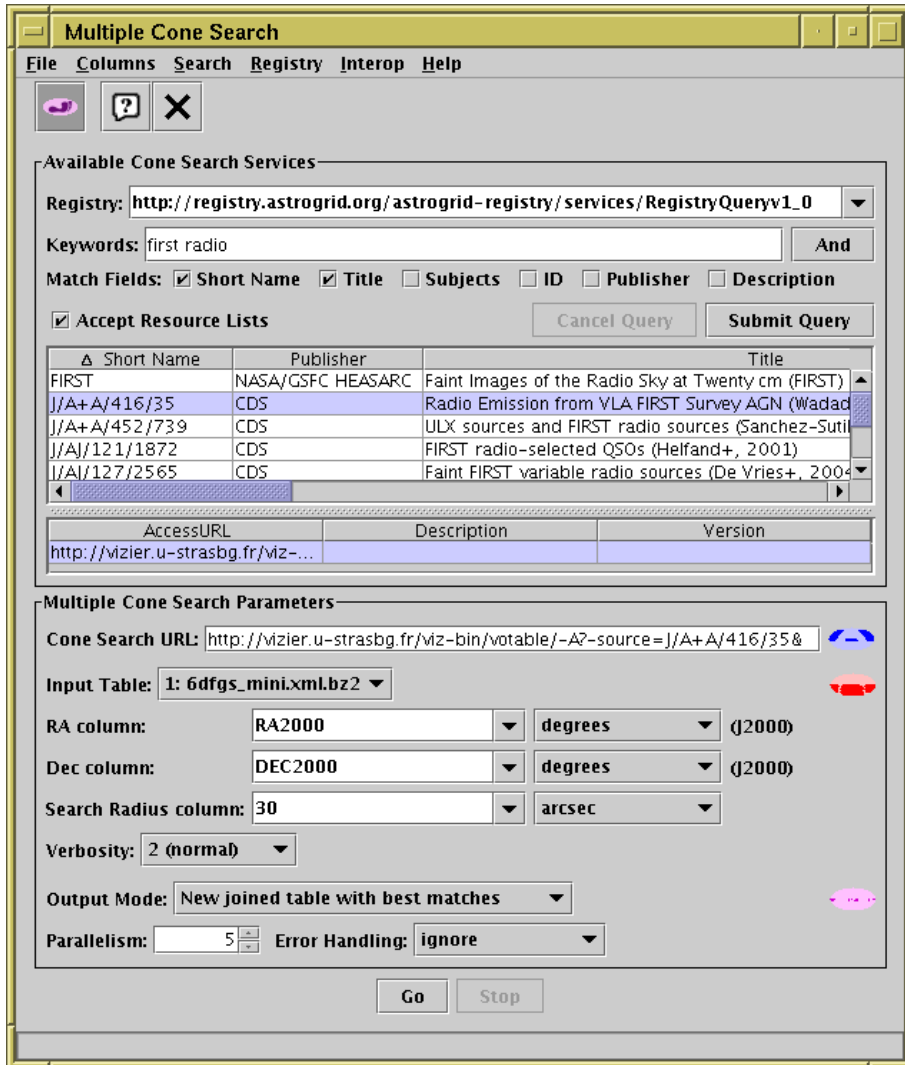
- Command `coneskymatch` has new parameters
- Parameters `usefoot`, `footinside`

## MOCs currently only available for Vizier

- MOC service at [http://alasky.u-strasbg.fr/footprints/getMoc?<service\\_url>](http://alasky.u-strasbg.fr/footprints/getMoc?<service_url>)



# Multi-Cone GUI



Little icons show (Mollweide, Equatorial):

← Remote catalogue coverage (from MOC)

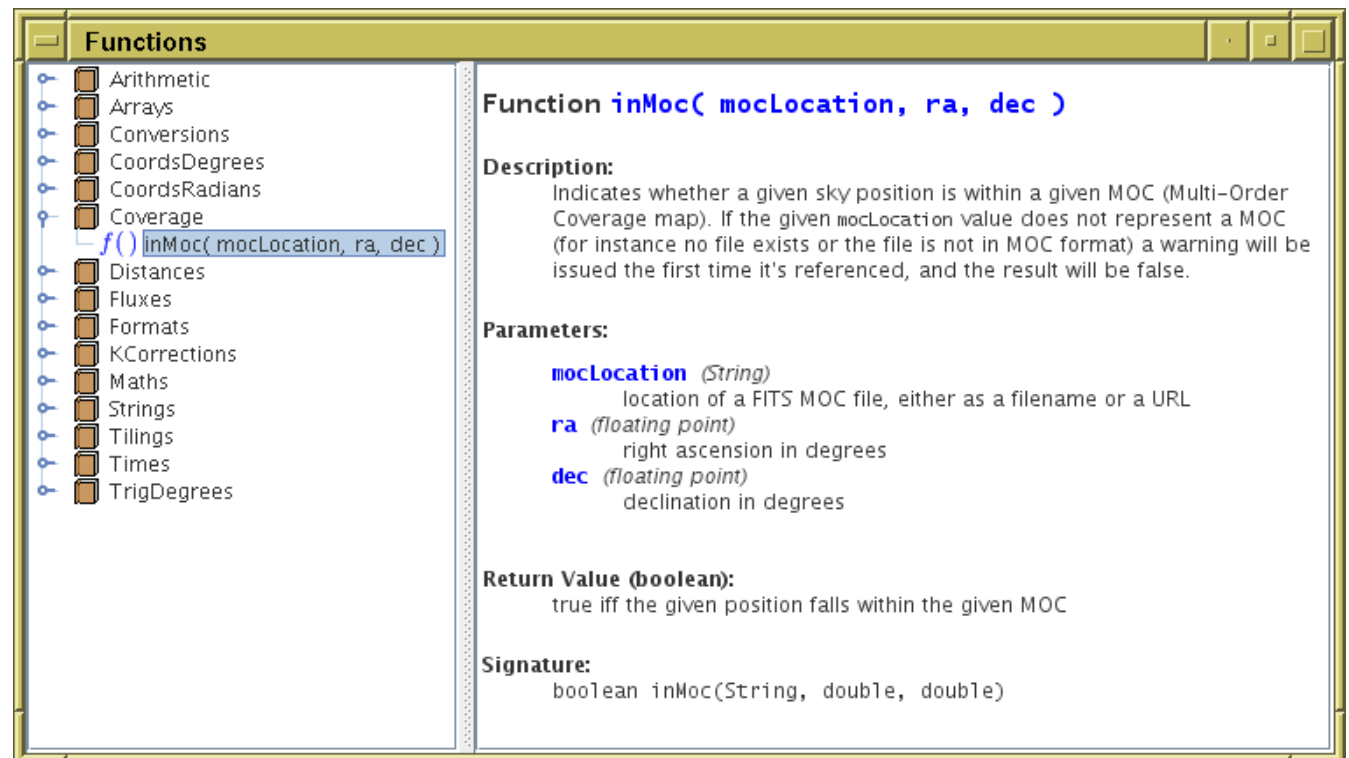
← Local catalogue coverage

← Intersection

# Coverage function

## New function `inMoc(mocLocation, ra, dec)`

- Usable from TOPCAT/STILTS expression language if MOC FITS file is available
- Reference MOC file by filename or URL
- E.g. new subset: `inMoc("dr7.moc", RA2000, DE2000)`
- Coverage read from MOC file read when first encountered, cached by filename/URL for subsequent calls



**Functions**

- Arithmetic
- Arrays
- Conversions
- CoordsDegrees
- CoordsRadians
- Coverage
  - `f() inMoc(mocLocation, ra, dec)`
- Distances
- Fluxes
- Formats
- KCorrections
- Maths
- Strings
- Tilings
- Times
- TrigDegrees

**Function `inMoc(mocLocation, ra, dec)`**

**Description:**  
Indicates whether a given sky position is within a given MOC (Multi-Order Coverage map). If the given `mocLocation` value does not represent a MOC (for instance no file exists or the file is not in MOC format) a warning will be issued the first time it's referenced, and the result will be false.

**Parameters:**

- `mocLocation` (*String*)  
location of a FITS MOC file, either as a filename or a URL
- `ra` (*floating point*)  
right ascension in degrees
- `dec` (*floating point*)  
declination in degrees

**Return Value (boolean):**  
true iff the given position falls within the given MOC

**Signature:**  
`boolean inMoc(String, double, double)`

# MOC Creation with STILTS

## New command `pixfoot`

- Generates MOC FITS file from a catalogue
- Parameters:
  - ▷ `in`: input table
  - ▷ `ra, dec`: input positions
  - ▷ `radius`: size associated with each position (may be fixed, zero, or variable)
  - ▷ `order`: HEALPix order (MOC resolution)
  - ▷ `out`: output MOC FITS file
- Example:

```
stilts pixfoot in=survey.vot ra=RA2000 dec=DEC2000
              order=8 mocfmt=fits out=sfoot.fits
```
- May be useful for data centers wanting to generate MOC files for Vizier-MOC-like service?