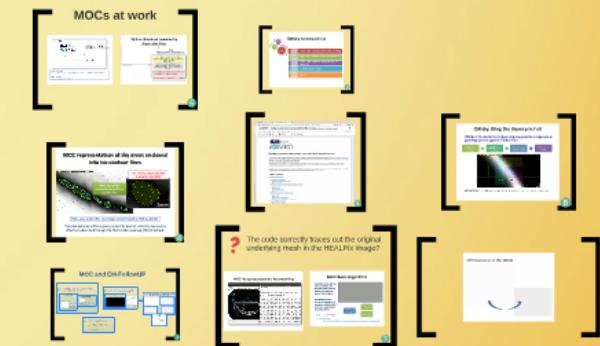
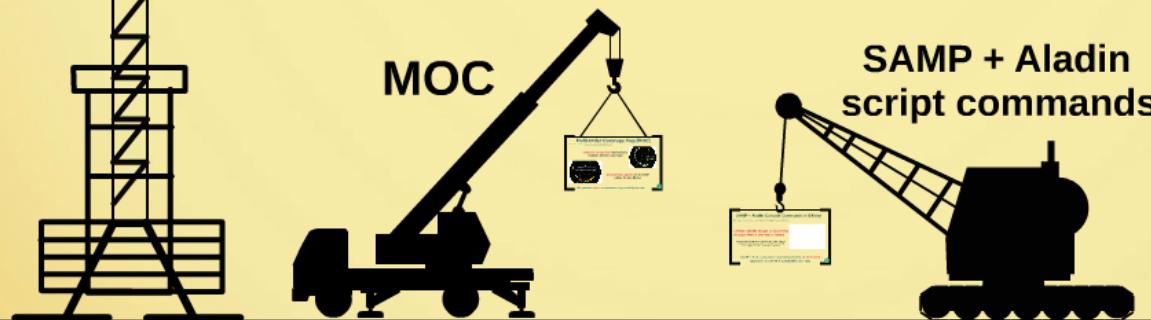


BUILDING (VO) IDEAS FOR O2 RUN

G. Greco, E. Chassande-Mottin, M. Branchesi, T. Boch, P. Fernique, G. Stratta and many others

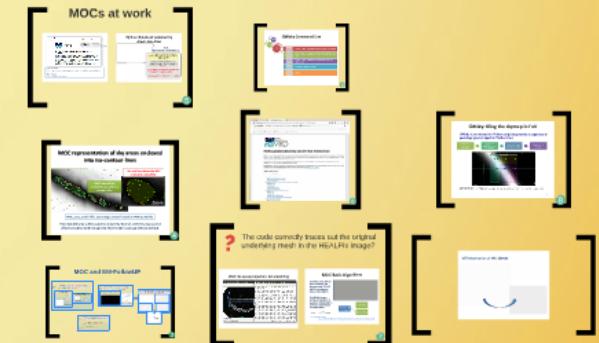
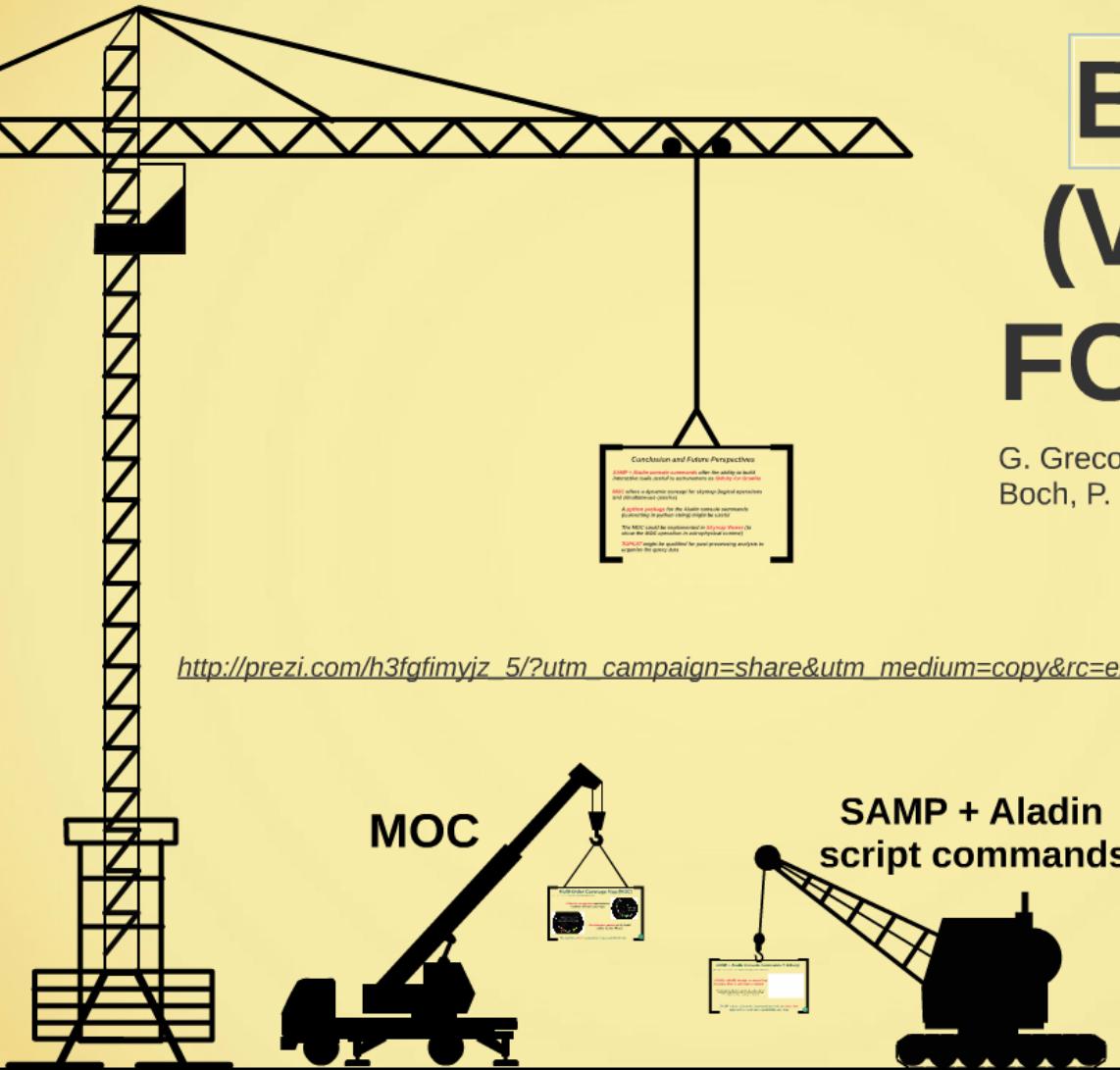
http://prezi.com/h3fgfimyjz_5/?utm_campaign=share&utm_medium=copy&rc=ex0share



BUILDING (VO) IDEAS FOR O2 RUN

G. Greco, E. Chassande-Mottin, M. Branchesi, T. Boch, P. Fernique, G. Stratta and many others

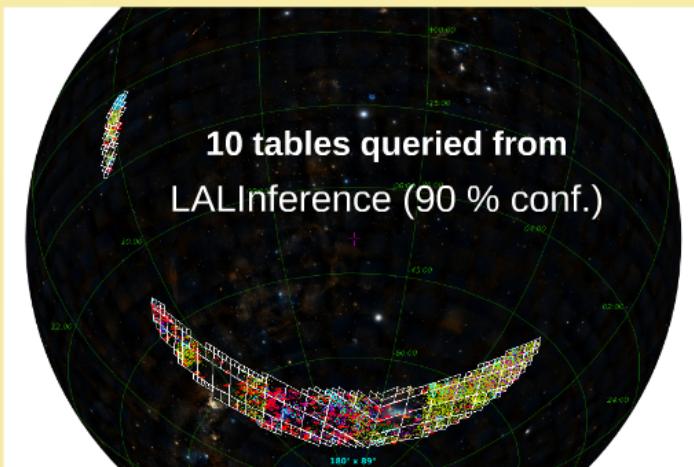
http://prezi.com/h3fgfimyjz_5/?utm_campaign=share&utm_medium=copy&rc=ex0share



Multi-Order Coverage Map (MOC)

Fernique et al. (2015); <http://arxiv.org/pdf/1505.02937v1.pdf>

- Effective comparison mechanisms between different sky maps



- Simultaneous queries of all VizieR tables (16.000 MOCs)

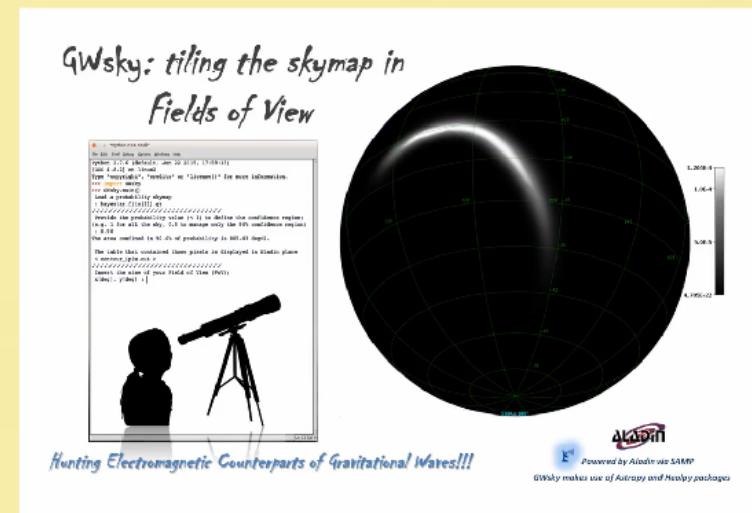
MOC provides a dynamic approach to manage probability sky maps

SAMP + Aladin Console Commands = GWsky

M.B. Taylor, T. Boch and J. Taylor <https://arxiv.org/pdf/1501.01139v1.pdf>

GWsky: quickly design an observing strategy when a sky map is issued

The sky map of a GW event - represented by white region of interest on a dark background - is tiled with multiple EM observations, each one targeting a colored tile



SAMP + Aladin Console Commands provide an **interactive** approach to work with a probability sky map

MOC and EM-FollowUP

The MOC method is based on the **HEALPix** tessellation algorithm and it is essentially a simple way to map irregular and complex sky regions into hierarchically grouped predefined cells.

Probability sky map

From Berger et al. 2004, The Astrophysical Journal, 612

- Sky localization of several hundred square degrees
- Source location can be characterized by disjoint sky areas
- Ring shape as a result of “triangulation” based on triangulation of the sky between detector sites
- We can describe complex sky region using just a list of cells based on Healpix

GW skymap: from 2009 to 2015

The original pipelines repeat the indicated portion of each candidate GW event as a skymap, a set of probability ellipses assigned to pixels in a grid covering the sky.

The 90% confidence level for each event (blue) is shown as a set of nested ellipses. This set is centered on the best-fit point (red dot). The ellipses are oriented along the principal axes of the posterior distribution.

A 90% confidence interval for the first five binary black hole events is shown in green.

HEALPix pixelization is introduced in the Advanced LIGO. The implementation is manageable!

The operations between the MOC maps (union, intersection, subtraction, difference) are extremely **simple and fast** (generally a few milliseconds) even for very complex sky regions.

The probability skymaps are disseminated using a sequence of algorithms with increasing accuracy and computational cost.

Distributed sky maps of GW 150914

CWB • $8.0 \times 10^{-10} \text{ rad}^2$ • $\text{Area} = 223 \text{ deg}^2$ (10% cont.)	LIGO • $4.0 \times 10^{-10} \text{ rad}^2$ • $\text{Area} = 250 \text{ deg}^2$ (10% cont.)	minimal assumptions of waveform morphology Bayesian inference assuming dissociated modulated Gaussian signal triangulating times, amplitudes and phases
SAYTEAT • $2.63 \times 10^{-12} \text{ rad}^2$ • $\text{Area} = 404 \text{ deg}^2$ (10% cont.)	triangulating times, amplitudes and phases CWB waveform, IRN gains and detector calibration uncertainties	
GAIA reference	Final localization	



GW Alerts and Elapsed Time

Initial Event Localization → Probability (posterior) → Localization at any stage

We need fast tool to compare them!



Some dataserver, such as VizieR, can be queried by MOC in order to return data (galaxy catalogs/list of images) only inside the MOC coverage.

To identify likely host galaxies of a GW event we need to collect as much information as possible

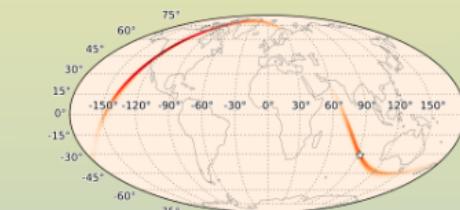


TOPCAT

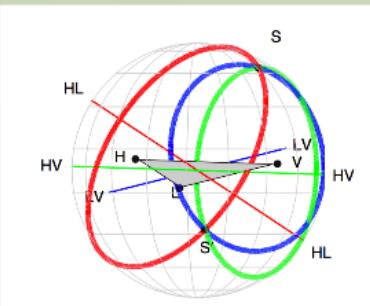


The MOC method is based on the **HEALPix** tessellation algorithm and it is essentially a simple way to map irregular and complex sky regions into hierarchically grouped predefined cells.

Probability sky map



From Singer et al. 2014, *The Astrophysical Journal*, 795



From Abbott et al. 2016, *Living Rev. Relativity* 19, 1

Sky localization of several hundred square degrees

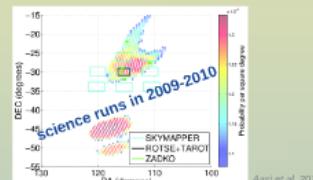
Source location can be characterized by disjoint sky areas

Ring shape as a result of "triangulation" based on arrival time delay between detector sites.

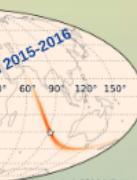
We can describe complex sky region using just a list of cells based on Healpix

GW skymap: from 2009 to 2015

The trigger pipelines report the estimated position of each candidate GW event as a skymap, a list of probability densities assigned to pixels in a grid covering the sky.



The gravitational wave skymap for event G19377 which occurred on 16th September 2010. This event was later revealed to be a blind injection.



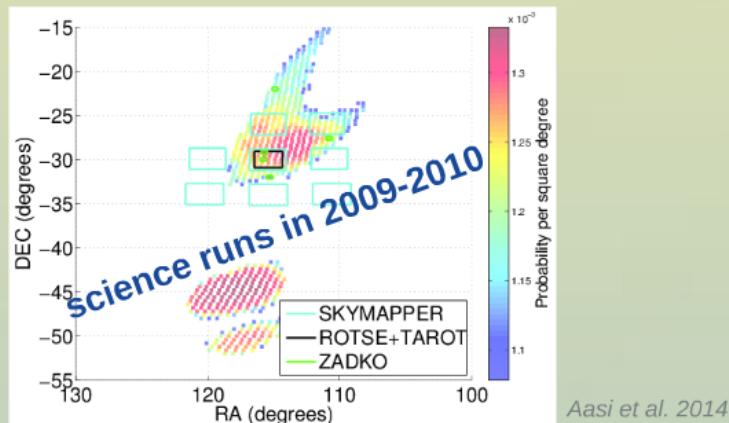
Singer et al. 2014 & Derry et al. 2025

ID 4532; data release for The First Two Years of Electromagnetic Follow-Up with Advanced LIGO and Virgo.

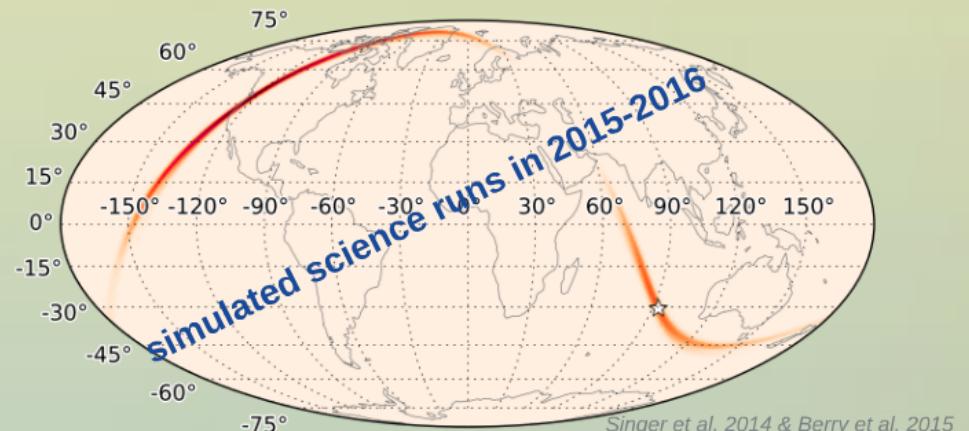
HEALPix pixelization is introduced in the Advanced LVC run.
The implementation is manageable!

GW skymap: from 2009 to 2015

The trigger pipelines report the estimated position of each candidate GW event as a skymap, a list of probability densities assigned to pixels in a grid covering the sky.



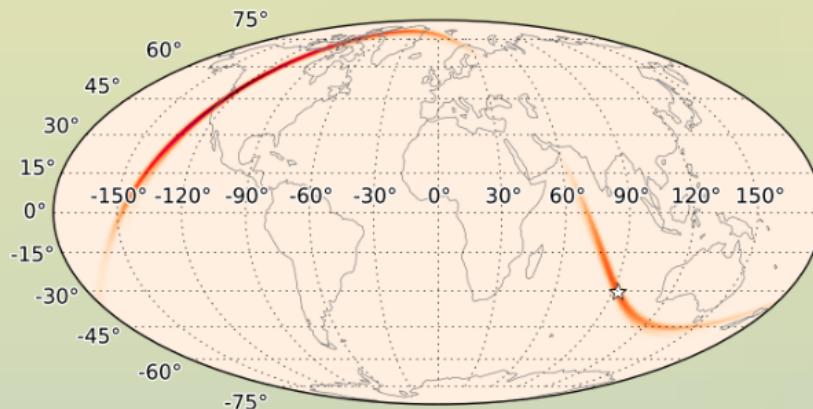
The gravitational wave skymap for event G19377 which occurred on 16th September 2010. This event was later revealed to be a blind injection.



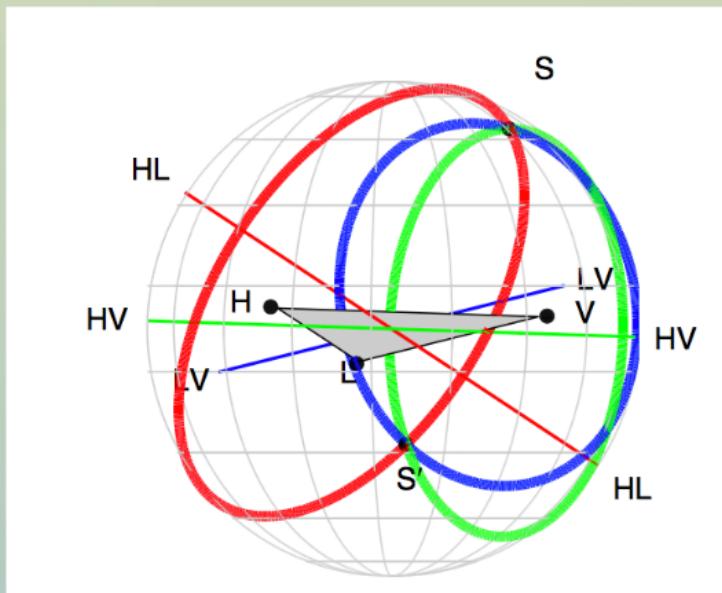
ID 4532; data release for The First Two Years of Electromagnetic Follow-Up with Advanced LIGO and Virgo.

**HEALPix pixelization is introduced in the Advanced LVC run.
The implementation is manageable!**

Probability sky map



From Singer et al. 2014, *The Astrophysical Journal*, 795



From Abbott et al. 2016, *Living Rev. Relativity* 19, 1

Sky localization of several hundred square degrees

Source location can be characterized by disjoint sky areas

Ring shape as a result of “triangulation” based on arrival time delay between detector sites.

We can describe complex sky region using just a list of cells based on Healpix



The **operations** between the MOC maps (union, intersection, subtraction, difference) are extremely **simple and fast** (generally a few milliseconds) even for very complex sky regions.

The probability skymaps are disseminated using a sequence of algorithms with increasing accuracy and computational cost.

Distributed sky maps of GW 150914

cWB

- Initially shared
- Area = 310 deg^2 (90% conf.)

LIB

- Initially shared
- Area = 750 deg^2 (90% conf.)

BAYESTAR

- 2016 Jan. 13
- Area = 400 deg^2 (90% conf.)

LALInference

- 2016 Jan. 13
- Area = 590 deg^2 (90% conf.)

minimal assumptions of waveform morphology

Bayesian inference assuming sinusoidally modulated Gaussian signal

triangulating times, amplitudes and phases

CBC waveform, BH spin and detector calibration uncertainties

Final localization

GW Alerts and Elapsed Time

Preliminary
[3 – 5 min]

Initial
[5 – 10 min]

Update
[Hours/days]

Human Event validation

Retraction at any stage

We need fast tool to compare them!

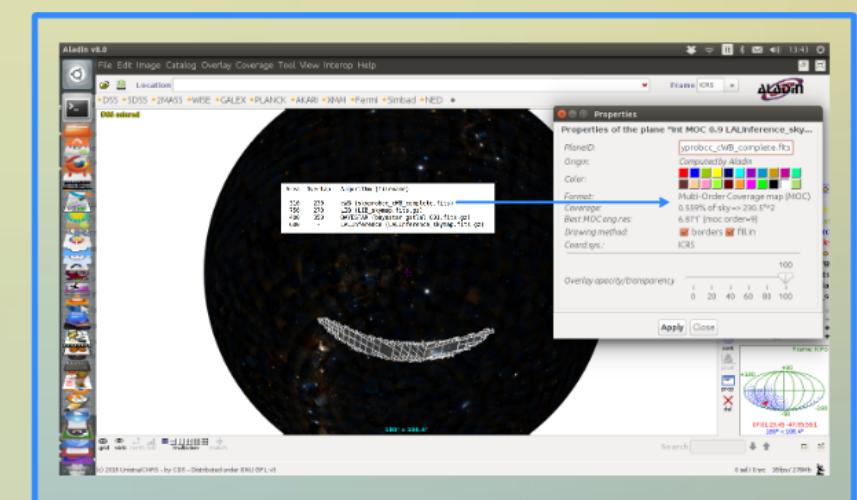
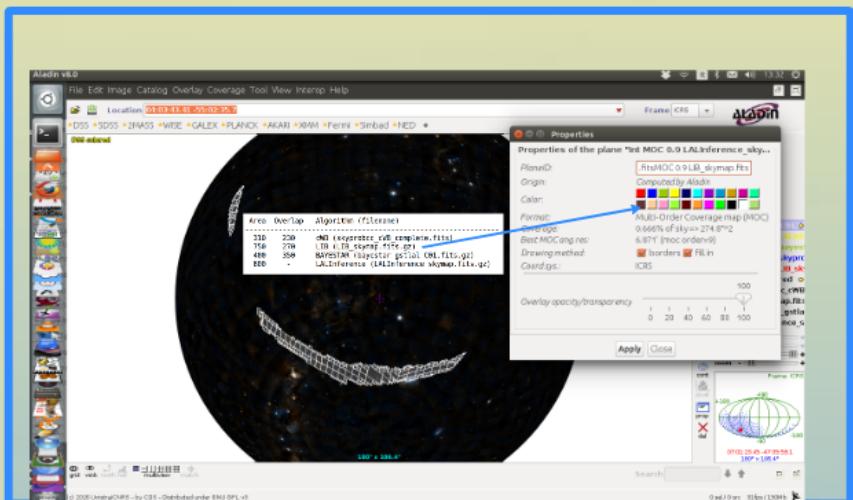
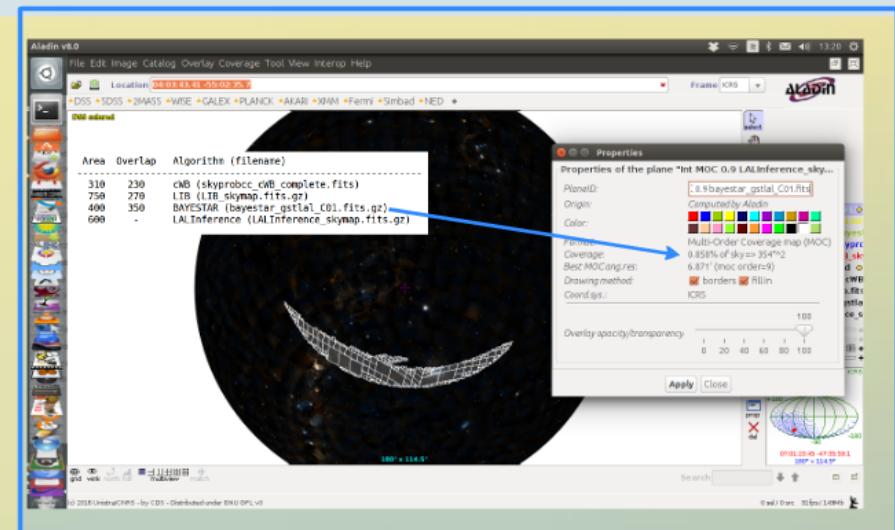
The GCN Circular (18858) shows the overlap regions between the sky maps published with the GW 150914.

The same regions are visualized applying the MOC method.

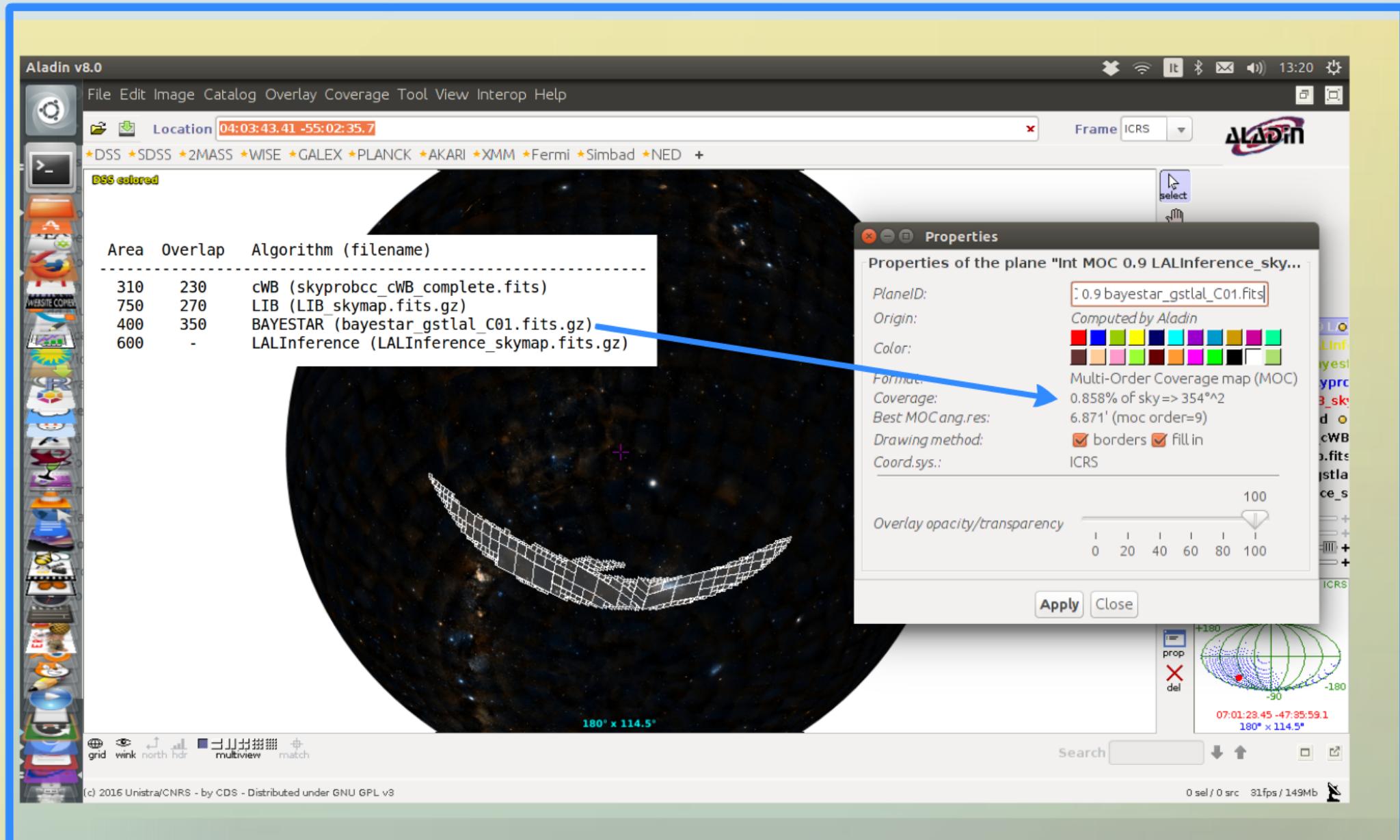
Both of the sky maps above agree with the initial LALinference Burst (LIB) localization (GCN 18330, LIB_skymap.fits.gz) on favoring the southern portion of the annulus determined by an arrival time difference between LIGO Hanford and LIGO Livingston of about 7 ms.

The table below presents a quantitative comparison of the available localizations along the lines of Sec. 4.5 of Essick et al. (2015, <http://adsabs.harvard.edu/abs/2015ApJ...800...81E>). The first column gives the area in deg² of the 90% credible region, and the second column gives the area in deg² of the overlap with the LALInference 90% credible region.

Area	Overlap	Algorithm (filename)
310	230	cWB (skyprobcc_cWB_complete.fits)
750	270	LIB (LIB_skymap.fits.gz)
400	350	BAYESTAR (bayestar_gstlal_C01.fits.gz)
600	-	LALInference (LALInference_skymap.fits.gz)

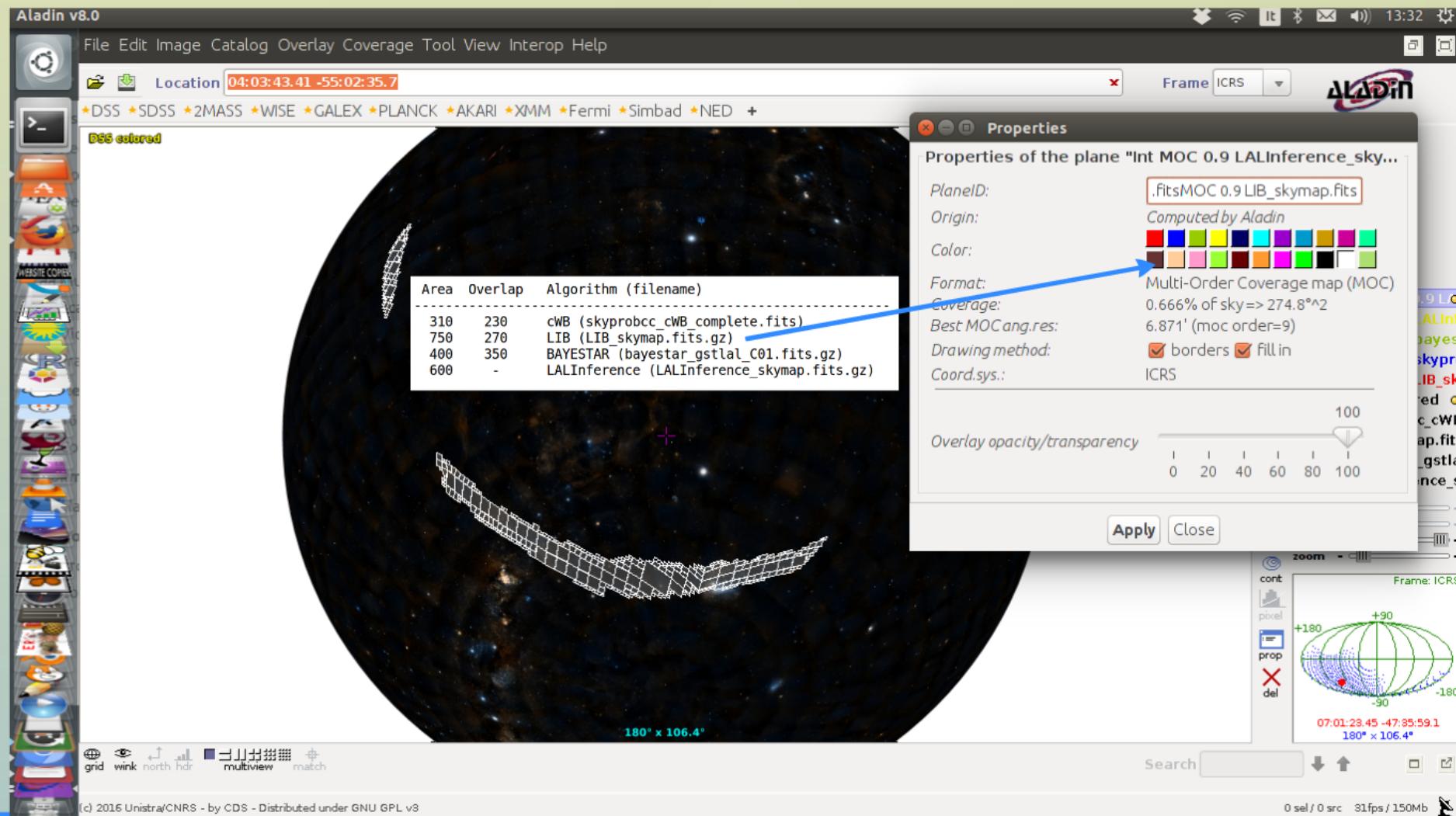


ed applying the MOC method.



400 350
600 -

BAYESTAR (bayestar_gstlal_C01.fits.gz)
LALInference (LALInference_skymap.fits.gz)



Aladin v8.0

File Edit Image Catalog Overlay Coverage Tool View Interop Help

13:43

Location

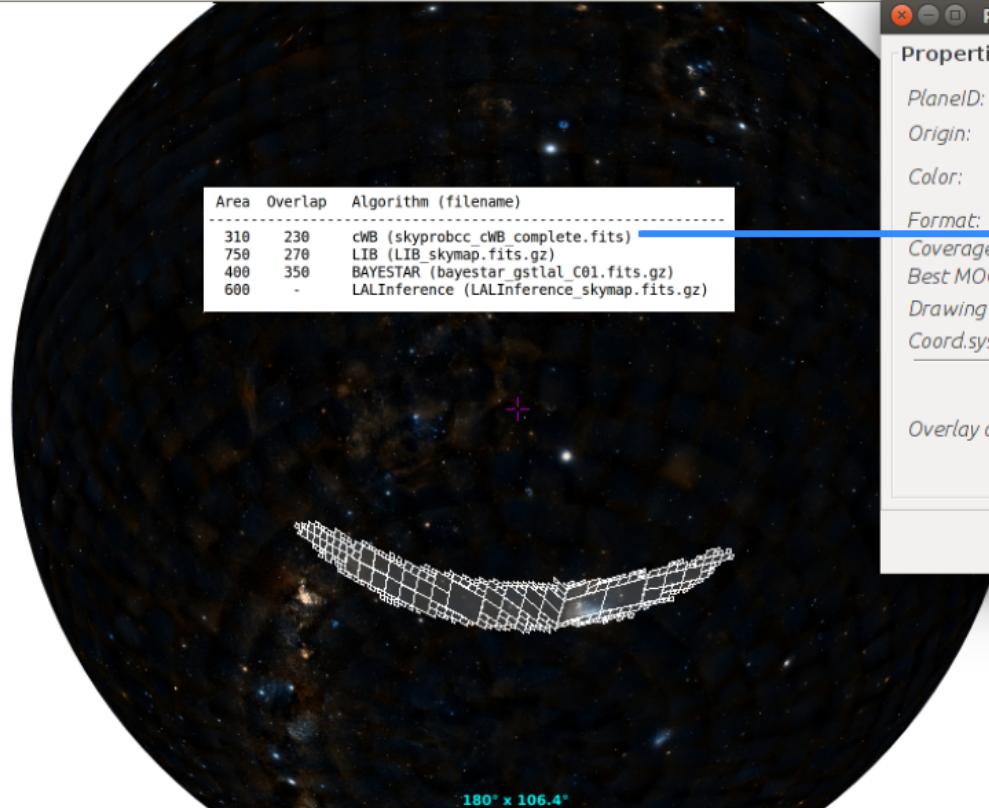
Frame ICRS



+DSS +SDSS +2MASS +WISE +GALEX +PLANCK +AKARI +XMM +Fermi +Simbad +NED +

DSS colored

Area	Overlap	Algorithm (filename)
310	230	cWB (skyprobcc_cWB_complete.fits)
750	270	LIB (LIB_skymap.fits.gz)
400	350	BAYESTAR (bayestar_gstlal_C01.fits.gz)
600	-	LALInference (LALInference_skymap.fits.gz)



Properties

Properties of the plane "Int MOC 0.9 LALInference_sky..."

PlaneID:

yprobcc_cWB_complete.fits

Origin:

Computed by Aladin

Color:



Format:

Multi-Order Coverage map (MOC)

Coverage:

0.559% of sky => 230.5°^2

Best MOCang.res:

6.871' (moc order=9)

Drawing method:

borders fill in

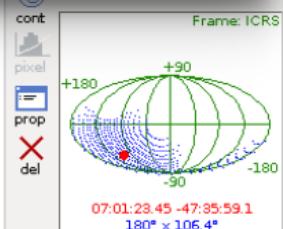
Coord.sys.:

ICRS

Overlay opacity/transparency



Apply Close



Search

0 sel / 0 src 26fps / 276Mb

(c) 2016 Unistra/CNRS - by CDS - Distributed under GNU GPL v3

Skymap Viewer

A sky atlas for understanding LIGO-Virgo skymaps. Help [here](#), or [ask a question](#). Skymaps [here](#). If you do not see the big dark sky map, look below the sky.

LIGO-Virgo Skymaps [?](#)

This is skymap GW150914:CWB.
50% area = 98.19 sq deg
90% area = 308.2 sq deg

South North [?](#)

Show Weighted Galaxies (or [table](#)).

Time and Place [?](#)

Universal time
2015-09-14T09:50:45 [Now](#)

E Longitude [east lon] Latitude [latitude]

[Show Sky](#)

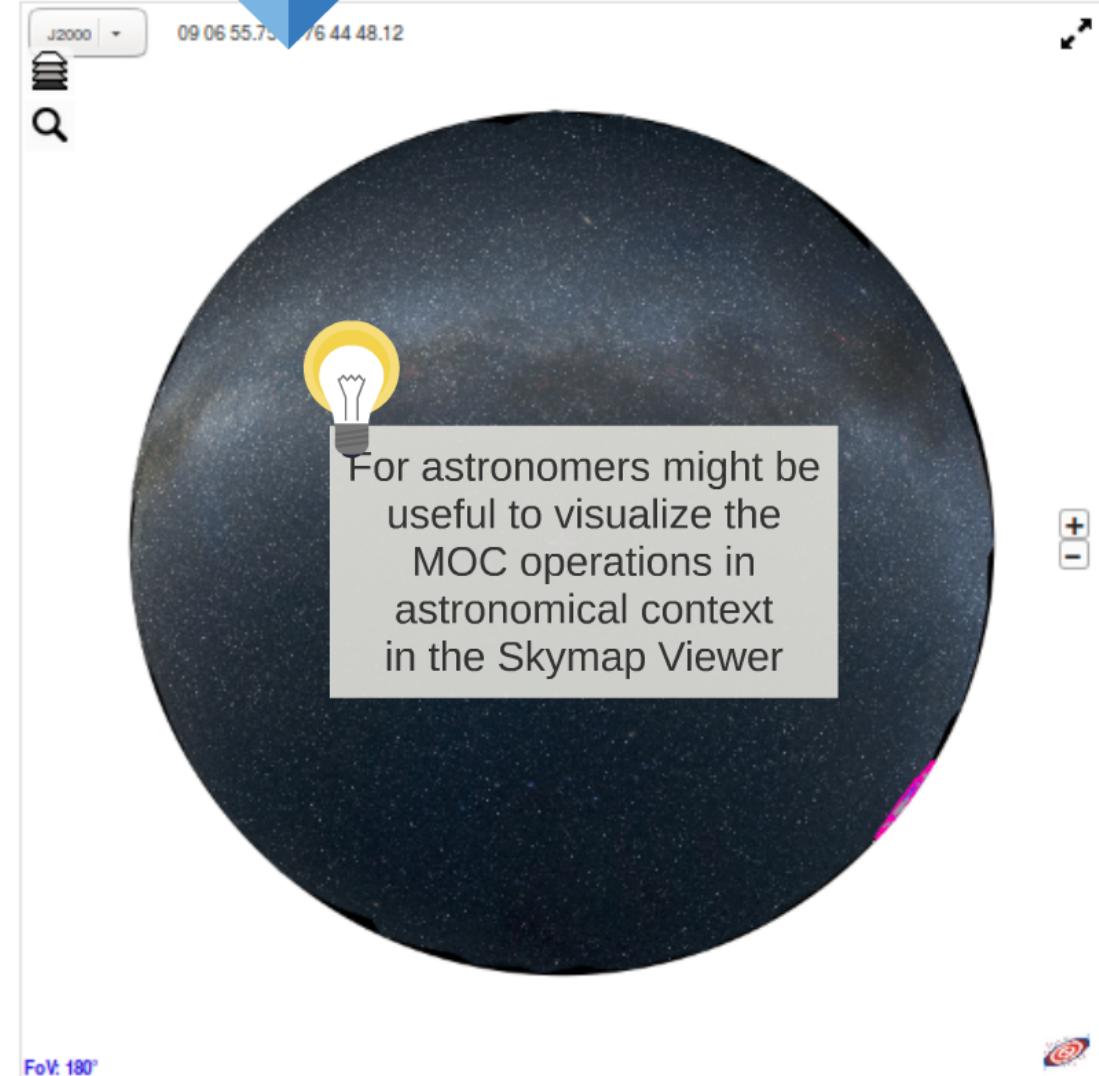
Sun = and = Moon

Catalog Sources [?](#)

Click the Layers icon to switch on catalogs. If you click on the sources on the sky, information will appear here with links to Simbad and NED.

Zoomable Multiwavelength Sky

Zoom in on the sky with the mouse or the +/- icons





Some dataserver, such as VizieR, can be queried by MOC in order to return data (galaxy catalogs/list of images) only inside the MOC coverage.

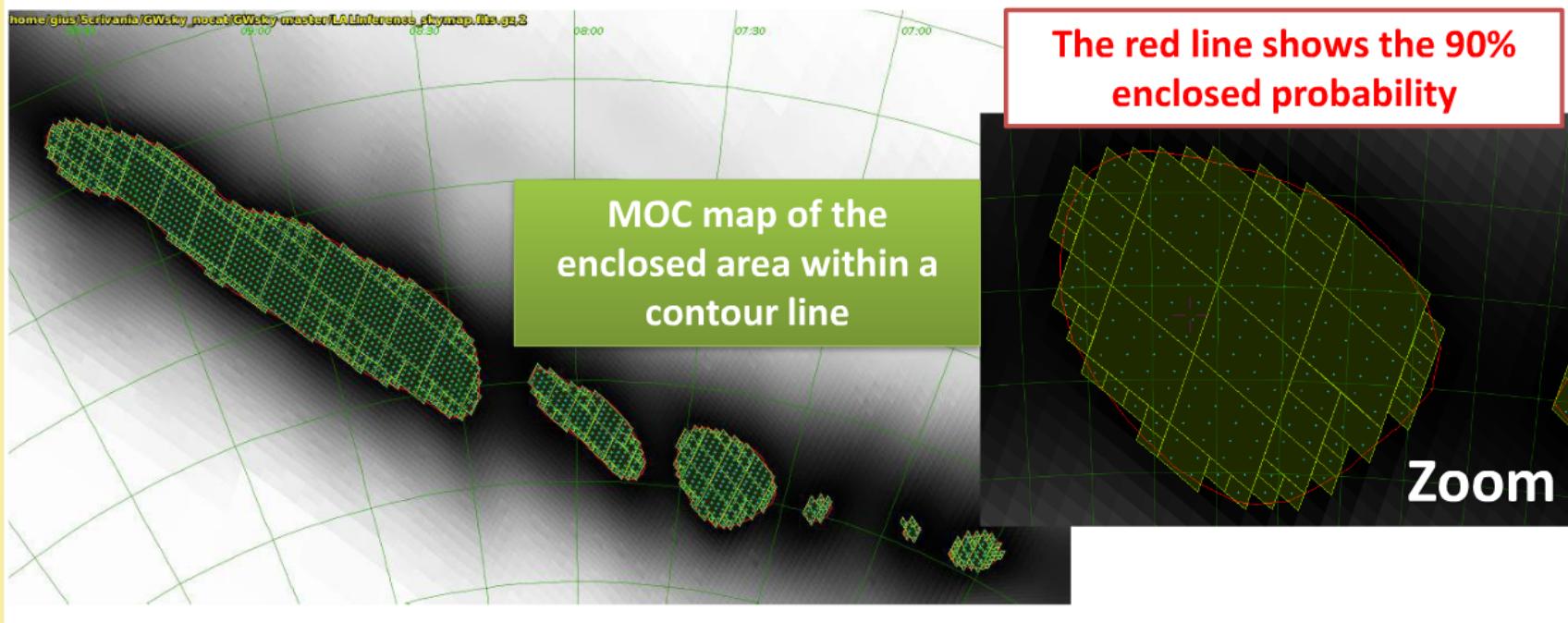
*To identify likely host galaxies of a GW event
we need to collect as much information as possible*



TOPCAT

post-processing: to organize data from queries.

MOC representation of sky areas enclosed into iso-contour lines



`MOC_area_prob(infile, percentage, output)` based on MOCpy module

The enclosed area within a given probability level of a GW sky map can be effectively described through the Multi-Order Coverage (MOC) method.

MOCs at work

This screenshot shows a Jupyter nbviewer page. At the top, there are logos for jupyter, nbviewer, LSC (LIGO Scientific Collaboration), and VIRGO. The main title is "IP[y]: IPython Interactive Computing". Below the title, the document is titled "Handling gravitational-wave sky maps with Multi-Order Coverage". It contains text about Multi-Order Coverage (MOC) maps based on HEALPix sky tessellation, mentioning the union and intersection of different sky maps and their combination with catalogs of galaxies. It also discusses sample code in Python and a video tutorial.

Ipython tutorial:
<http://nbviewer.jupyter.org/gist/ggreco77/5fc0cc2777f9edd446b459459db830e9>

Video Tutorial:
<https://vimeo.com/167173587>

This screenshot shows an IPython Notebook interface. The title bar says "IPython Notebook powered by Aladin Sky Atlas". The notebook cell contains Python code: "In [1]: from IPython.core.display import HTML". Below the code, there is a screenshot of the Aladin Java applet displaying a sky map. To the right of the notebook, there is a diagram showing two people, one labeled "Aladin" and one labeled "python", connected by a red arrow pointing towards each other, with "SAMP" (Simple Application Messaging Protocol) represented by blue speech bubbles above them. Several callout boxes provide details: "The Aladin java applet is embedded in the ipython cell, as the code runs the results are displayed in real time in Aladin planes"; "The interoperability between Aladin and ipython notebook is obtained using the SAMPIintegratedClient with a set of dedicated functions"; and "The Aladin Console commands are 'converted' in python string" followed by the code "rename_plane = 'rename' + ' + plane".



Handling gravitational-wave sky maps with Multi-Order Coverage

This document explains how gravitational wave sky maps can be easily and efficiently visualized and processed using [Multi-Order Coverage \(MOC\)](#) maps based on [HEALPix](#) sky tessellation. We compute the union and intersection of different sky maps and we combine them with catalogs of galaxies. For this tutorial we use the sky maps published with GW150914.

We provide sample code in Python; you can download this document and run the code samples in [IPython Notebook](#). The results are displayed in real time in [Aladin Sky Atlas](#) which is embedded in the document.

Caution. While the method is effective for having fast catalog queries, the use of the MOC leads a loss of information. By reducing a map in only a single confidence region, the probability distribution within that region is irreversible lost; see also [Essick et al. \(2015\)](#).

Ipython tutorial:

<http://nbviewer.jupyter.org/gist/ggreco77/5fc0cc2777f9edd446b459459db830e9>

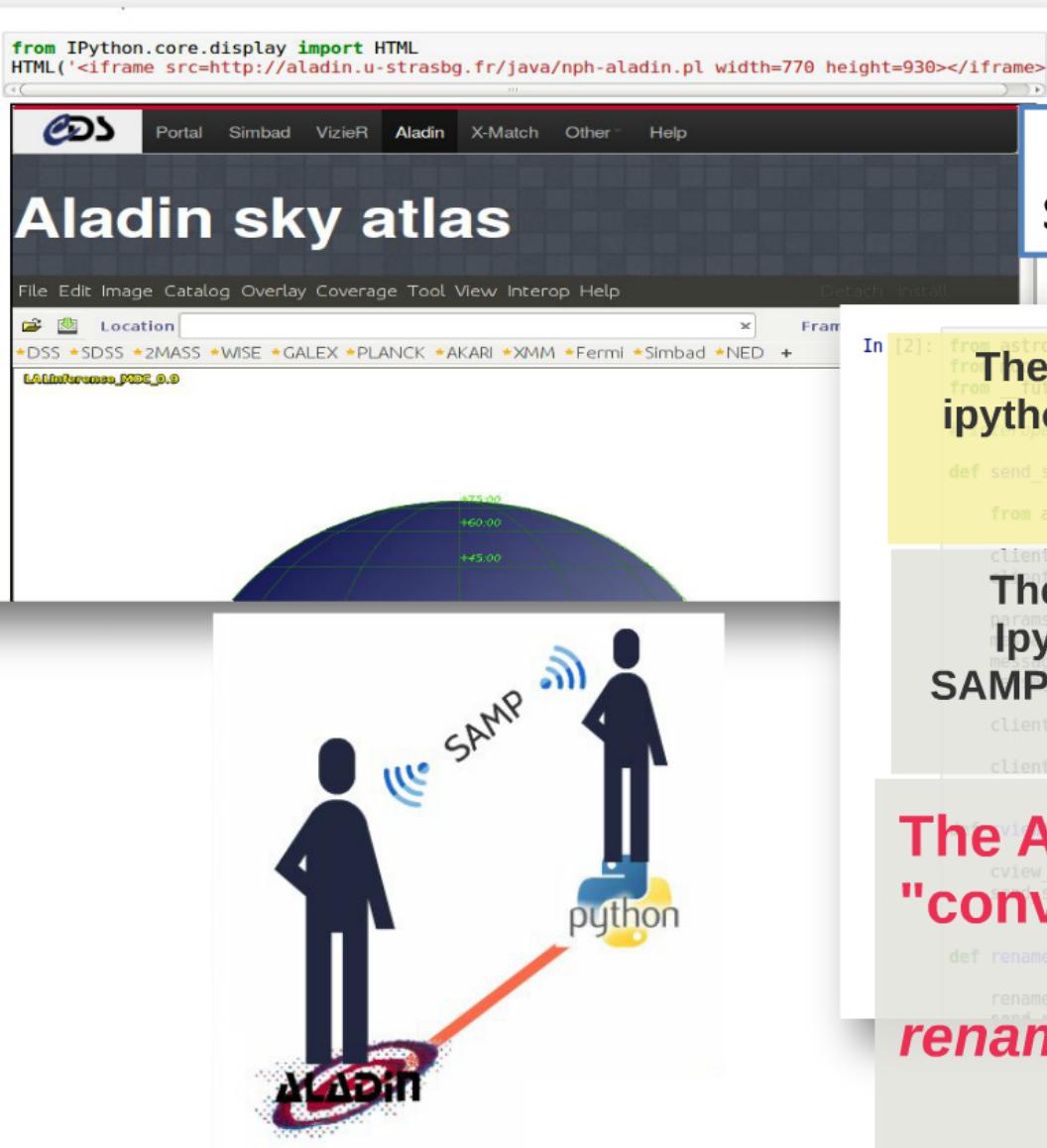
Video Tutorial:

<https://vimeo.com/167173587>

IPython Notebook powered by Aladin Sky Atlas

In [1]: `from IPython.core.display import HTML
HTML('<iframe src=http://aladin.u-strasbg.fr/java/nph-aladin.pl width=770 height=930></iframe>')`

Out[1]:



The screenshot shows an IPython Notebook cell with the output of an 'In' cell. The output is an 'HTML' cell containing an 'iframe' tag with the source set to 'http://aladin.u-strasbg.fr/java/nph-aladin.pl' and attributes 'width=770' and 'height=930'. The resulting output is a screenshot of the Aladin Java applet, which is a map of the sky with various astronomical objects and coordinates (RA and Dec) visible.

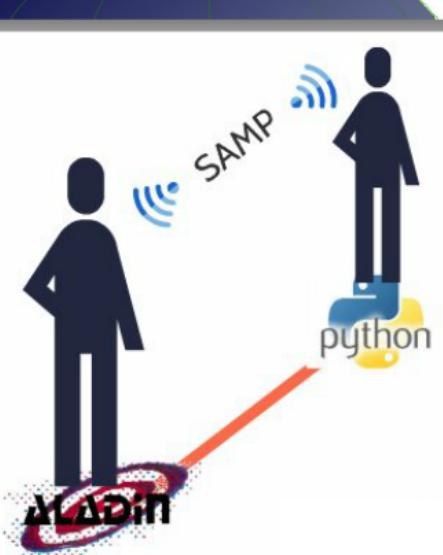
SAMP:
Simple Application Messaging Protocol

The Aladin java applet is embedded in the ipython cell, as the code runs the results are displayed in real time in Aladin planes

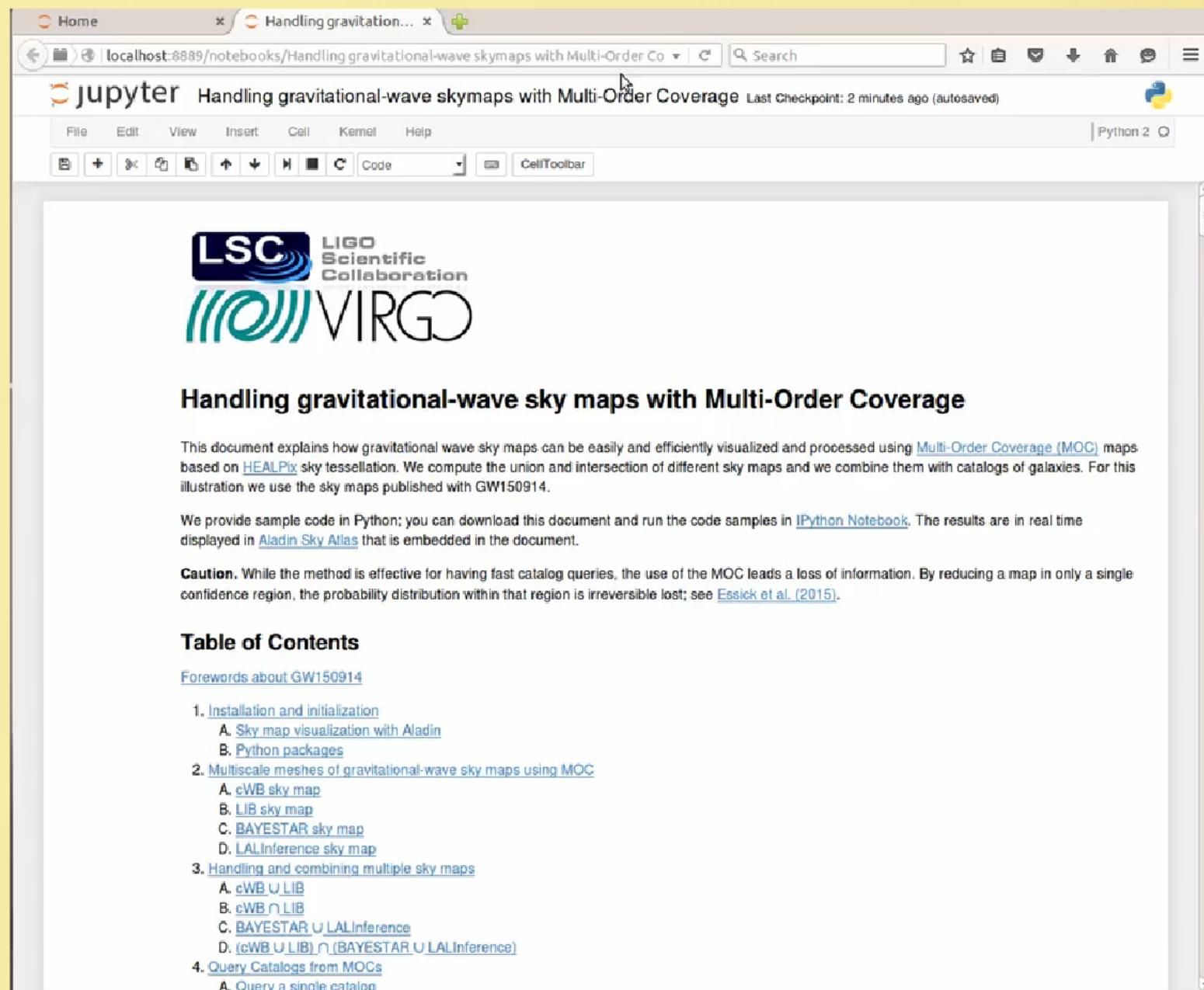
The interoperability between Aladin and Ipython notebook is obtained using the **SAMPIntegratedClient** with a set of dedicated functions

The Aladin Console commands are "converted" in python string

```
cview_url = 'cview' + ' ' + url
rename_plane = 'rename' + ' ' + plane
```



A diagram illustrating the SAMP integration. It shows two dark blue silhouettes of people. Between them is a white rectangular box containing the text 'SAMP'. Below this box is a yellow Python logo icon. A red arrow points from the Python icon towards the left silhouette, which is standing on a small circular base labeled 'Aladin'.



Handling gravitational-wave sky maps with Multi-Order Coverage

This document explains how gravitational wave sky maps can be easily and efficiently visualized and processed using [Multi-Order Coverage \(MOC\)](#) maps based on [HEALPix](#) sky tessellation. We compute the union and intersection of different sky maps and we combine them with catalogs of galaxies. For this illustration we use the sky maps published with GW150914.

We provide sample code in Python; you can download this document and run the code samples in [IPython Notebook](#). The results are in real time displayed in [Aladin Sky Atlas](#) that is embedded in the document.

Caution. While the method is effective for having fast catalog queries, the use of the MOC leads a loss of information. By reducing a map in only a single confidence region, the probability distribution within that region is irreversible lost; see [Essick et al. \(2015\)](#).

Table of Contents

[Forewords about GW150914](#)

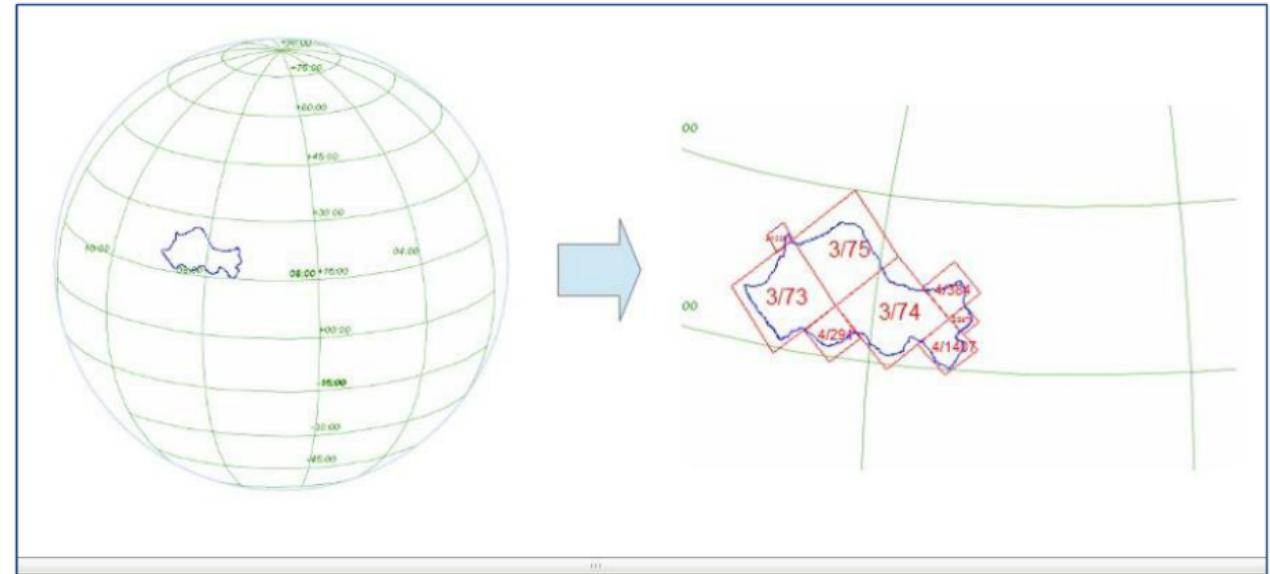
1. [Installation and initialization](#)
 - A. [Sky map visualization with Aladin](#)
 - B. [Python packages](#)
2. [Multiscale meshes of gravitational-wave sky maps using MOC](#)
 - A. [cWB sky map](#)
 - B. [LIB sky map](#)
 - C. [BAYESTAR sky map](#)
 - D. [LALInference sky map](#)
3. [Handling and combining multiple sky maps](#)
 - A. [cWB U LIB](#)
 - B. [cWB ∩ LIB](#)
 - C. [BAYESTAR U LALInference](#)
 - D. [\(cWB U LIB\) ∩ \(BAYESTAR U LALInference\)](#)
4. [Query Catalogs from MOCs](#)
 - A. [Query a single catalog](#)

MOC Basic Algorithm

Each MOC cell is defined by two numbers: the hierarchy level (**HEALPIX ORDER**) and the pixel index (**HEALPIX NPIX**).

The **NUNIQ** scheme defines an algorithm for packing an (**ORDER, NPIX**) pair into a single integer for compactness:

$$uniq = 4 \times 4^{order} + npix$$



Encoding
Formats

FITS MOC

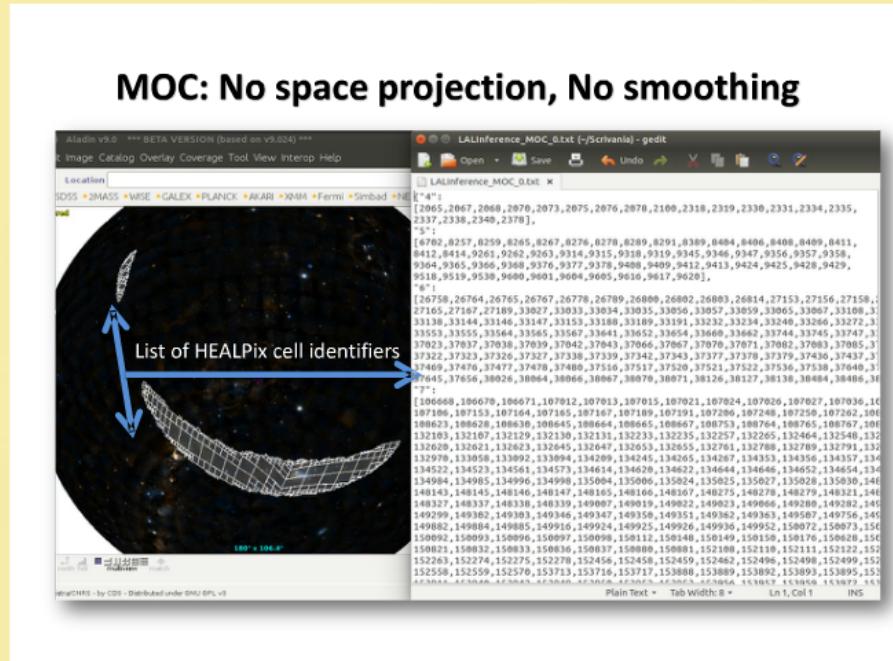
JSON MOC

Fernique et al. 2014

<http://ivoa.net/documents/MOC/20140602/REC-MOC-1.0-20140602.pdf>



The code correctly traces out the original underlying mesh in the HEALPix image?

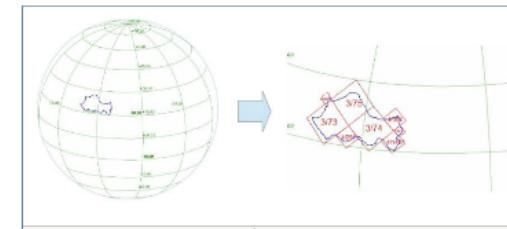


MOC Basic Algorithm

Each MOC cell is defined by two numbers: the hierarchy level (HEALPIX ORDER) and the pixel index (HEALPIX NPIX).

The NUNIQ scheme defines an algorithm for packing an (ORDER, NPIX) pair into a single integer for compactness:

$$uniqu = 4 \times 4^{order} + npix$$

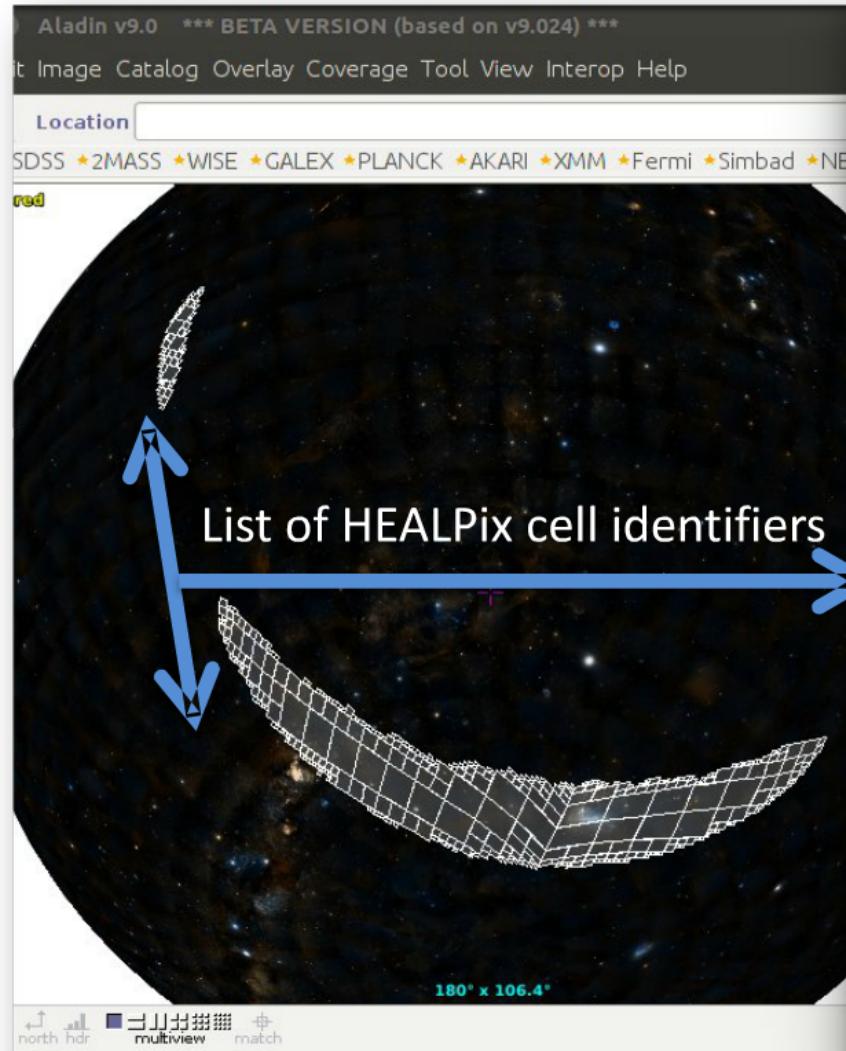


```
graph LR; A[Encoding Formats] --> B[FITS MOC]; A --> C[JSON MOC]
```

Fernique et al. 2014

<http://ivoa.net/documents/MOC/20140602/REC-MOC-1.0-20140602.pdf>

MOC: No space projection, No smoothing



LALInference_MOC_0.txt (~/Scrivania) - gedit

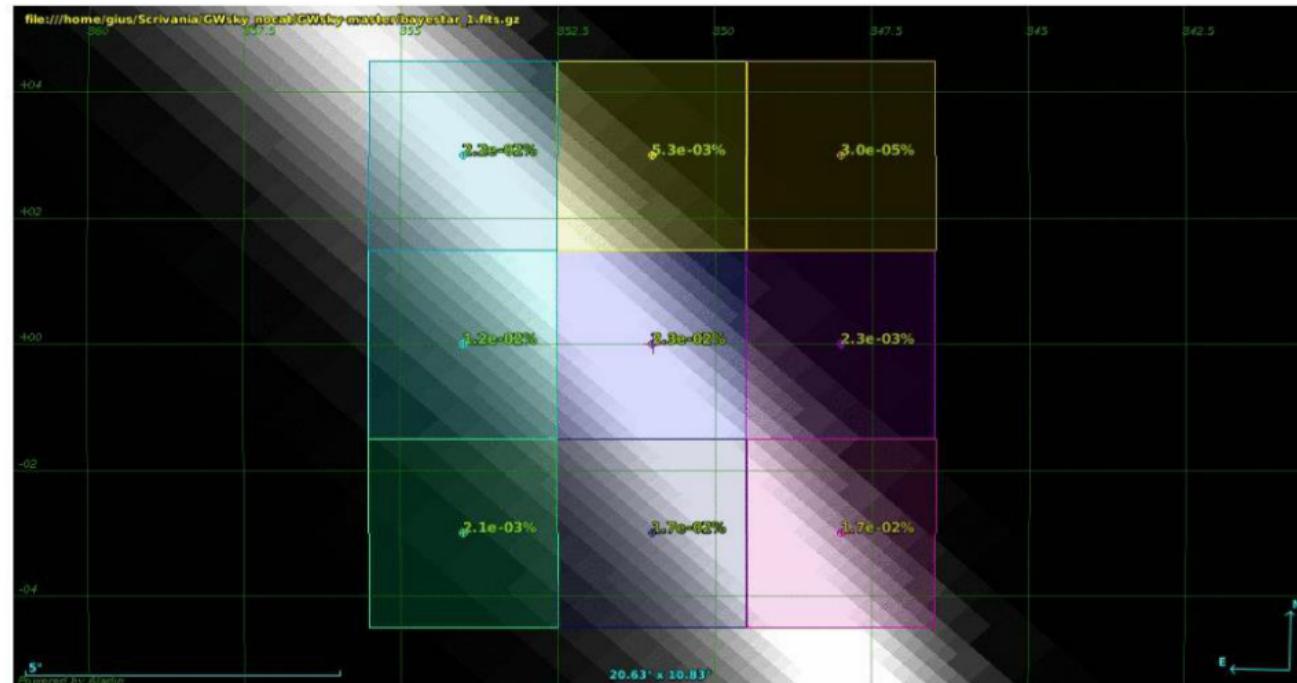
Open Save Undo Redo Cut Copy Paste Find Replace

LALInference_MOC_0.txt x

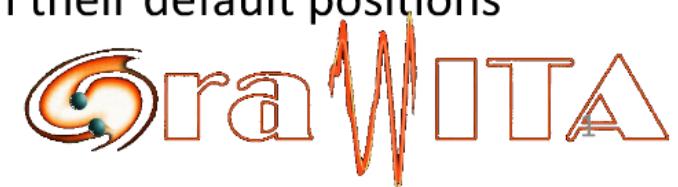
```
[{"4": [2065, 2067, 2068, 2070, 2073, 2075, 2076, 2078, 2100, 2318, 2319, 2330, 2331, 2334, 2335, 2337, 2338, 2340, 2378], "5": [6702, 8257, 8259, 8265, 8267, 8276, 8278, 8289, 8291, 8389, 8404, 8406, 8408, 8409, 8411, 8412, 8414, 9261, 9262, 9263, 9314, 9315, 9318, 9319, 9345, 9346, 9347, 9356, 9357, 9358, 9364, 9365, 9366, 9368, 9376, 9377, 9378, 9408, 9409, 9412, 9413, 9424, 9425, 9428, 9429, 9518, 9519, 9530, 9600, 9601, 9604, 9605, 9616, 9617, 9620], "6": [26758, 26764, 26765, 26767, 26778, 26789, 26800, 26802, 26803, 26814, 27153, 27156, 27158, 27165, 27167, 27189, 33027, 33033, 33034, 33035, 33056, 33057, 33059, 33065, 33067, 33108, 33138, 33144, 33146, 33147, 33153, 33188, 33189, 33191, 33232, 33234, 33240, 33266, 33272, 33553, 33555, 33564, 33565, 33567, 33641, 33652, 33654, 33660, 33662, 33744, 33745, 33747, 37023, 37037, 37038, 37039, 37042, 37043, 37066, 37067, 37070, 37071, 37082, 37083, 37085, 37322, 37323, 37326, 37327, 37338, 37339, 37342, 37343, 37377, 37378, 37379, 37436, 37437, 37469, 37476, 37477, 37478, 37480, 37516, 37517, 37520, 37521, 37522, 37536, 37538, 37640, 37645, 37656, 38026, 38064, 38066, 38067, 38070, 38071, 38126, 38127, 38138, 38484, 38486, 38745, 38746, 38747, 38748, 38749, 38750, 38751, 38752, 38753, 38754, 38755, 38756, 38757, 38758, 38759, 38760, 38761, 38762, 38763, 38764, 38765, 38766, 38767, 38768, 38769, 38770, 38771, 38772, 38773, 38774, 38775, 38776, 38777, 38778, 38779, 38780, 38781, 38782, 38783, 38784, 38785, 38786, 38787, 38788, 38789, 38790, 38791, 38792, 38793, 38794, 38795, 38796, 38797, 38798, 38799, 38701, 38702, 38703, 38704, 38705, 38706, 38707, 38708, 38709, 38710, 38711, 38712, 38713, 38714, 38715, 38716, 38717, 38718, 38719, 38720, 38721, 38722, 38723, 38724, 38725, 38726, 38727, 38728, 38729, 38730, 38731, 38732, 38733, 38734, 38735, 38736, 38737, 38738, 38739, 38740, 38741, 38742, 38743, 38744, 38745, 38746, 38747, 38748, 38749, 38750, 38751, 38752, 38753, 38754, 38755, 38756, 38757, 38758, 38759, 38760, 38761, 38762, 38763, 38764, 38765, 38766, 38767, 38768, 38769, 38770, 38771, 38772, 38773, 38774, 38775, 38776, 38777, 38778, 38779, 38780, 38781, 38782, 38783, 38784, 38785, 38786, 38787, 38788, 38789, 38790, 38791, 38792, 38793, 38794, 38795, 38796, 38797, 38798, 38799, 38701, 38702, 38703, 38704, 38705, 38706, 38707, 38708, 38709, 387010, 387011, 387012, 387013, 387014, 387015, 387016, 387017, 387018, 387019, 387020, 387021, 387022, 387023, 387024, 387025, 387026, 387027, 387028, 387029, 387030, 387031, 387032, 387033, 387034, 387035, 387036, 387037, 387038, 387039, 387040, 387041, 387042, 387043, 387044, 387045, 387046, 387047, 387048, 387049, 387050, 387051, 387052, 387053, 387054, 387055, 387056, 387057, 387058, 387059, 387060, 387061, 387062, 387063, 387064, 387065, 387066, 387067, 387068, 387069, 387070, 387071, 387072, 387073, 387074, 387075, 387076, 387077, 387078, 387079, 387080, 387081, 387082, 387083, 387084, 387085, 387086, 387087, 387088, 387089, 387090, 387091, 387092, 387093, 387094, 387095, 387096, 387097, 387098, 387099, 3870100, 3870101, 3870102, 3870103, 3870104, 3870105, 3870106, 3870107, 3870108, 3870109, 3870110, 3870111, 3870112, 3870113, 3870114, 3870115, 3870116, 3870117, 3870118, 3870119, 3870120, 3870121, 3870122, 3870123, 3870124, 3870125, 3870126, 3870127, 3870128, 3870129, 3870130, 3870131, 3870132, 3870133, 3870134, 3870135, 3870136, 3870137, 3870138, 3870139, 3870140, 3870141, 3870142, 3870143, 3870144, 3870145, 3870146, 3870147, 3870148, 3870149, 3870150, 3870151, 3870152, 3870153, 3870154, 3870155, 3870156, 3870157, 3870158, 3870159, 3870160, 3870161, 3870162, 3870163, 3870164, 3870165, 3870166, 3870167, 3870168, 3870169, 3870170, 3870171, 3870172, 3870173, 3870174, 3870175, 3870176, 3870177, 3870178, 3870179, 3870180, 3870181, 3870182, 3870183, 3870184, 3870185, 3870186, 3870187, 3870188, 3870189, 3870190, 3870191, 3870192, 3870193, 3870194, 3870195, 3870196, 3870197, 3870198, 3870199, 38701910, 38701911, 38701912, 38701913, 38701914, 38701915, 38701916, 38701917, 38701918, 38701919, 38701920, 38701921, 38701922, 38701923, 38701924, 38701925, 38701926, 38701927, 38701928, 38701929, 38701930, 38701931, 38701932, 38701933, 38701934, 38701935, 38701936, 38701937, 38701938, 38701939, 38701940, 38701941, 38701942, 38701943, 38701944, 38701945, 38701946, 38701947, 38701948, 38701949, 38701950, 38701951, 38701952, 38701953, 38701954, 38701955, 38701956, 38701957, 38701958, 38701959, 38701960, 38701961, 38701962, 38701963, 38701964, 38701965, 38701966, 38701967, 38701968, 38701969, 38701970, 38701971, 38701972, 38701973, 38701974, 38701975, 38701976, 38701977, 38701978, 38701979, 38701980, 38701981, 38701982, 38701983, 38701984, 38701985, 38701986, 38701987, 38701988, 38701989, 38701990, 38701991, 38701992, 38701993, 38701994, 38701995, 38701996, 38701997, 38701998, 38701999, 387019100, 387019101, 387019102, 387019103, 387019104, 387019105, 387019106, 387019107, 387019108, 387019109, 387019110, 387019111, 387019112, 387019113, 387019114, 387019115, 387019116, 387019117, 387019118, 387019119, 387019120, 387019121, 387019122, 387019123, 387019124, 387019125, 387019126, 387019127, 387019128, 387019129, 387019130, 387019131, 387019132, 387019133, 387019134, 387019135, 387019136, 387019137, 387019138, 387019139, 387019140, 387019141, 387019142, 387019143, 387019144, 387019145, 387019146, 387019147, 387019148, 387019149, 387019150, 387019151, 387019152, 387019153, 387019154, 387019155, 387019156, 387019157, 387019158, 387019159, 387019160, 387019161, 387019162, 387019163, 387019164, 387019165, 387019166, 387019167, 387019168, 387019169, 387019170, 387019171, 387019172, 387019173, 387019174, 387019175, 387019176, 387019177, 387019178, 387019179, 387019180, 387019181, 387019182, 387019183, 387019184, 387019185, 387019186, 387019187, 387019188, 387019189, 387019190, 387019191, 387019192, 387019193, 387019194, 387019195, 387019196, 387019197, 387019198, 387019199, 3870191100, 3870191101, 3870191102, 3870191103, 3870191104, 3870191105, 3870191106, 3870191107, 3870191108, 3870191109, 3870191110, 3870191111, 3870191112, 3870191113, 3870191114, 3870191115, 3870191116, 3870191117, 3870191118, 3870191119, 3870191120, 3870191121, 3870191122, 3870191123, 3870191124, 3870191125, 3870191126, 3870191127, 3870191128, 3870191129, 3870191130, 3870191131, 3870191132, 3870191133, 3870191134, 3870191135, 3870191136, 3870191137, 3870191138, 3870191139, 3870191140, 3870191141, 3870191142, 3870191143, 3870191144, 3870191145, 3870191146, 3870191147, 3870191148, 3870191149, 3870191150, 3870191151, 3870191152, 3870191153, 3870191154, 3870191155, 3870191156, 3870191157, 3870191158, 3870191159, 3870191160, 3870191161, 3870191162, 3870191163, 3870191164, 3870191165, 3870191166, 3870191167, 3870191168, 3870191169, 3870191170, 3870191171, 3870191172, 3870191173, 3870191174, 3870191175, 3870191176, 3870191177, 3870191178, 3870191179, 3870191180, 3870191181, 3870191182, 3870191183, 3870191184, 3870191185, 3870191186, 3870191187, 3870191188, 3870191189, 3870191190, 3870191191, 3870191192, 3870191193, 3870191194, 3870191195, 3870191196, 3870191197, 3870191198, 3870191199, 38701911000, 38701911001, 38701911002, 38701911003, 38701911004, 38701911005, 38701911006, 38701911007, 38701911008, 38701911009, 38701911010, 38701911011, 38701911012, 38701911013, 38701911014, 38701911015, 38701911016, 38701911017, 38701911018, 38701911019, 38701911020, 38701911021, 38701911022, 38701911023, 38701911024, 38701911025, 38701911026, 38701911027, 38701911028, 38701911029, 38701911030, 38701911031, 38701911032, 38701911033, 38701911034, 38701911035, 38701911036, 38701911037, 38701911038, 38701911039, 38701911040, 38701911041, 38701911042, 38701911043, 38701911044, 38701911045, 38701911046, 38701911047, 38701911048, 38701911049, 38701911050, 38701911051, 38701911052, 38701911053, 38701911054, 38701911055, 38701911056, 38701911057, 38701911058, 38701911059, 38701911060, 38701911061, 38701911062, 38701911063, 38701911064, 38701911065, 38701911066, 38701911067, 38701911068, 38701911069, 38701911070, 38701911071, 38701911072, 38701911073, 38701911074, 38701911075, 38701911076, 38701911077, 38701911078, 38701911079, 38701911080, 38701911081, 38701911082, 38701911083, 38701911084, 38701911085, 38701911086, 38701911087, 38701911088, 38701911089, 38701911090, 38701911091, 38701911092, 38701911093, 38701911094, 38701911095, 38701911096, 38701911097, 38701911098, 38701911099, 38701911100, 38701911101, 38701911102, 38701911103, 38701911104, 38701911105, 38701911106, 38701911107, 38701911108, 38701911109, 38701911110, 38701911111, 38701911112, 38701911113, 38701911114, 38701911115, 38701911116, 38701911117, 38701911118, 38701911119, 387019111100, 387019111101, 387019111102, 387019111103, 387019111104, 387019111105, 387019111106, 387019111107, 387019111108, 387019111109, 387019111110, 387019111111, 387019111112, 387019111113, 387019111114, 387019111115, 387019111116, 387019111117, 387019111118, 387019111119, 3870191111100, 3870191111101, 3870191111102, 3870191111103, 3870191111104, 3870191111105, 3870191111106, 3870191111107, 3870191111108, 3870191111109, 3870191111110, 3870191111111, 3870191111112, 3870191111113, 3870191111114, 3870191111115, 3870191111116, 3870191111117, 3870191111118, 3870191111119, 38701911111100, 38701911111101, 38701911111102, 38701911111103, 38701911111104, 38701911111105, 38701911111106, 38701911111107, 38701911111108, 38701911111109, 38701911111110, 38701911111111, 38701911111112, 38701911111113, 38701911111114, 38701911111115, 38701911111116, 38701911111117, 38701911111118, 38701911111119, 387019111111100, 387019111111101, 387019111111102, 387019111111103, 387019111111104, 387019111111105, 387019111111106, 387019111111107, 387019111111108, 387019111111109, 387019111111110, 387019111111111, 387019111111112, 387019111111113, 387019111111114, 387019111111115, 387019111111116, 387019111111117, 387019111111118, 387019111111119, 3870191111111100, 3870191111111101, 3870191111111102, 3870191111111103, 3870191111111104, 3870191111111105, 3870191111111106, 3870191111111107, 3870191111111108, 3870191111111109, 3870191111111110, 3870191111111111, 3870191111111112, 3870191111111113, 3870191111111114, 3870191111111115, 3870191111111116, 3870191111111117, 3870191111111118, 3870191111111119, 38701911111111100, 38701911111111101, 38701911111111102, 38701911111111103, 38701911111111104, 38701911111111105, 38701911111111106, 38701911111111107, 38701911111111108, 38701911111111109, 38701911111111110, 38701911111111111, 38701911111111112, 38701911111111113, 38701911111111114, 38701911111111115, 38701911111111116, 38701911111111117, 38701911111111118, 38701911111111119, 387019111111111100, 387019111111111101, 387019111111111102, 387019111111111103, 387019111111111104, 387019111111111105, 387019111111111106, 387019111111111107, 387019111111111108, 387019111111111109, 387019111111111110, 387019111111111111, 387019111111111112, 387019111111111113, 387019111111111114, 387019111111111115, 387019111111111116, 387019111111111117, 387019111111111118, 387019111111111119, 3870191111111111100, 3870191111111111101, 3870191111111111102, 3870191111111111103, 3870191111111111104, 3870191111111111105, 3870191111111111106, 3870191111111111107, 3870191111111111108, 3870191111111111109, 3870191111111111110, 3870191111111111111, 3870191111111111112, 3870191111111111113, 3870191111111111114, 3870191111111111115, 3870191111111111116, 3870191111111111117, 3870191111111111118, 3870191111111111119, 38701911111111111100, 38701911111111111101, 38701911111111111102, 38701911111111111103, 38701911111111111104, 38701911111111111105, 38701911111111111106, 38701911111111111107, 38701911111111111108, 38701911111111111109, 38701911111111111110, 38701911111111111111, 38701911111111111112, 38701911111111111113, 38701911111111111114, 38701911111111111115, 38701911111111111116, 38701911111111111117, 38701911111111111118, 38701911111111111119, 387019111111111111100, 387019111111111111101, 387019111111111111102, 387019111111111111103, 387019111111111111104, 387019111111111111105, 387019111111111111106, 387019111111111111107, 387019111111111111108, 387019111111111111109, 387019111111111111110, 387019111111111111111, 387019111111111111112, 387019111111111111113, 387019111111111111114, 387019111111111
```

GWsky: tiling the skymap in FoV

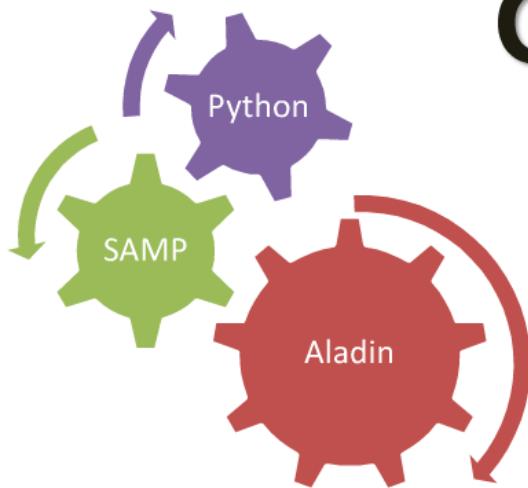
GWsky is an interactive Python script to generate a sequence of pointings given a specific Field of View



USER OPTION: the FoVs can be overlaid or separated from their default positions



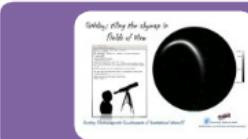
GWsky Command Line



C runs a new sequence *changing* the FoV center



I runs a new sequence without drawing the *input* FoV



L runs a new sequence starting from the *last* drawn FoV



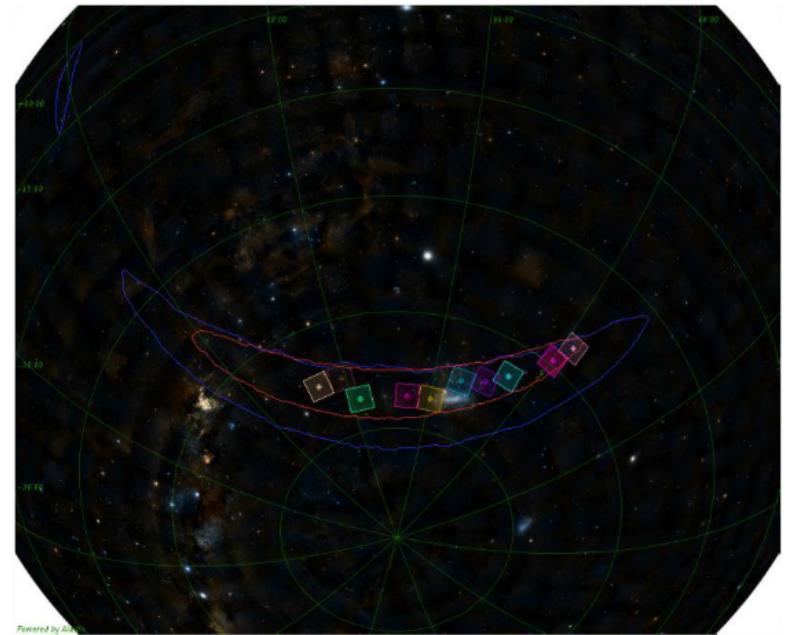
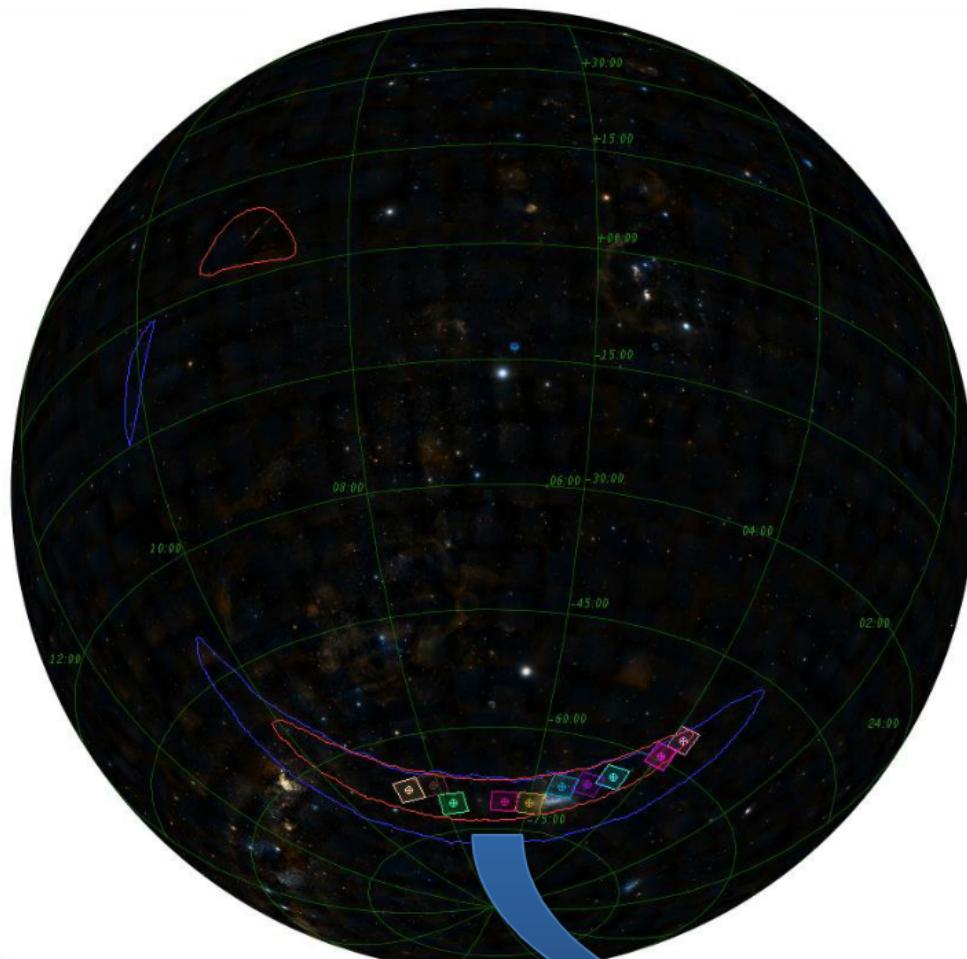
R repeats the last action



Q quit



VST Observation of GW 150914



Survey Area Definition Tool (VST)

File Options Help

Survey ID vst_survey

Survey Areas

Type	Lon	Lat	Diameter (d...)	Angle (d...)	System	Exclude
Coordinate Range	30.0	-2.0	35.0	1.2	0 Galactic	<input type="checkbox"/>
Coordinate Range	19:10:00	-02:00:00	19:30:00	+02:00:00	0 FK5 (J20...	<input type="checkbox"/>
Geodesic Rectangle	19:20:00	-07:00:00	5.0	4.0	-20 FK5 (J20...	<input type="checkbox"/>
Circle	26.0	-2.5	4.5		0 Galactic	<input type="checkbox"/>

Add Survey Area Delete Survey Area

Select Dither Pattern:
OMEGACAM_Dither_diag_5

Select Catalogue
GSC-2 at ESO

View / Update Areas Start / Resume Reset Plastic

Gravitational Wave Observatory

Conclusion and Future Perspectives

SAMP + Aladin console commands offer the ability to build interactive tools useful to astronomers as **GWsky for Grawita**

MOC offers a dynamic concept for skymap (logical operations and simultaneous queries)

A **python package** for the **Aladin console commands** (converting in python string) might be useful

The **MOC** could be implemented in **Skymap Viewer** (to show the MOC operation in astrophysical context)

TOPCAT might be qualified for post-processing analysis to organize the query data