Challenges when Implementing the Planck Virtual Image Service with SIA 2

Walter Landry
IRSA/IPAC/Caltech

Wei Mi, Xiuqin Wu, Angela Zhang,
Peregrine McGhee, Steve Groom

Planck Mission

- The ESA Planck satellite (2009-2013) was designed to observe the cosmic microwave background over the entire sky.
- It has 72 different detectors covering 9 different frequencies.
- The detectors are single pixel sensors taking data about 100 times a second.
- Coverage of the sky is provided by rotation of the spacecraft.

Planck TOI Data

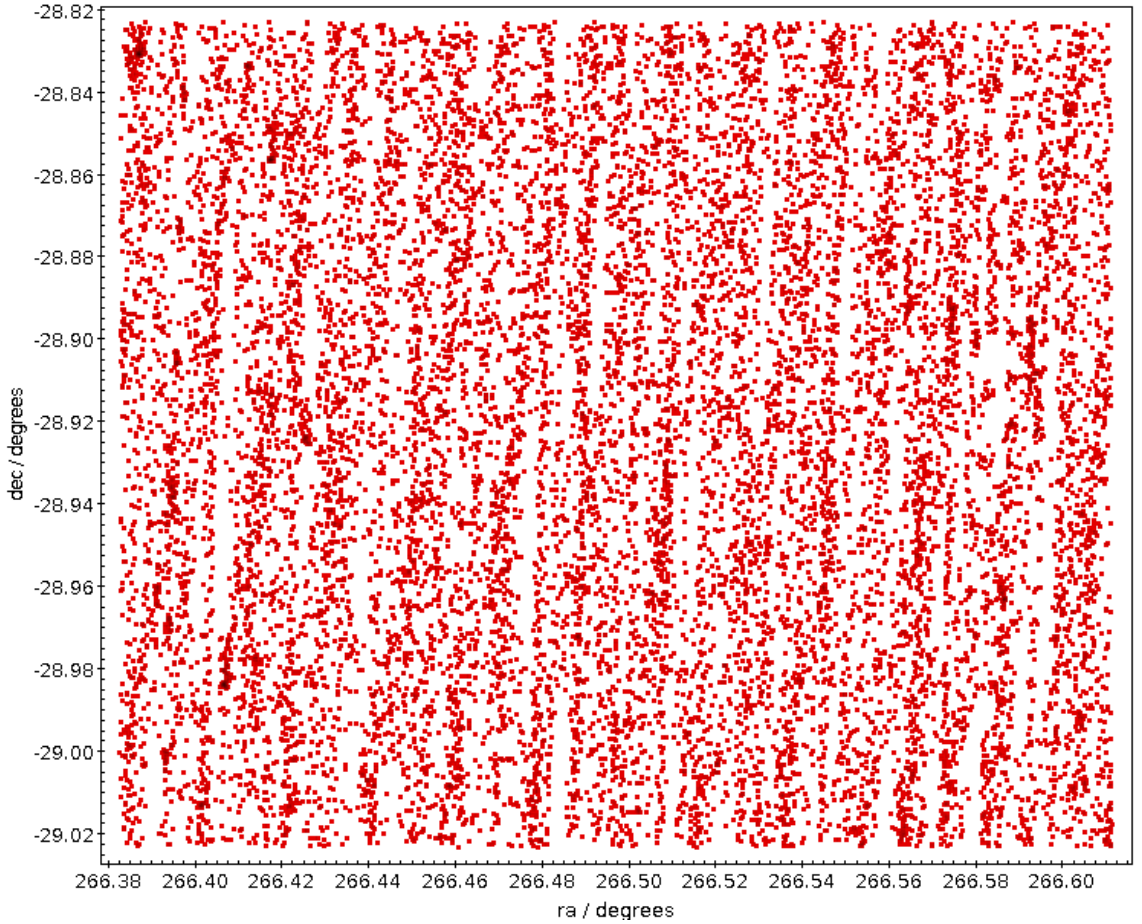
- The end result of this is a series of very large tables (40 TB total) of TOI's (Time Ordered Information)

RA	DEC	MJD	PSI	SIGNAL
121.177	-21.5810	56036	-149.28	.0022
121.174	-21.5807	56244	9.808	-.0023
121.184	-21.5732	56401	-149.26	.0024

- We store the tables in a special purpose file format (HTM).

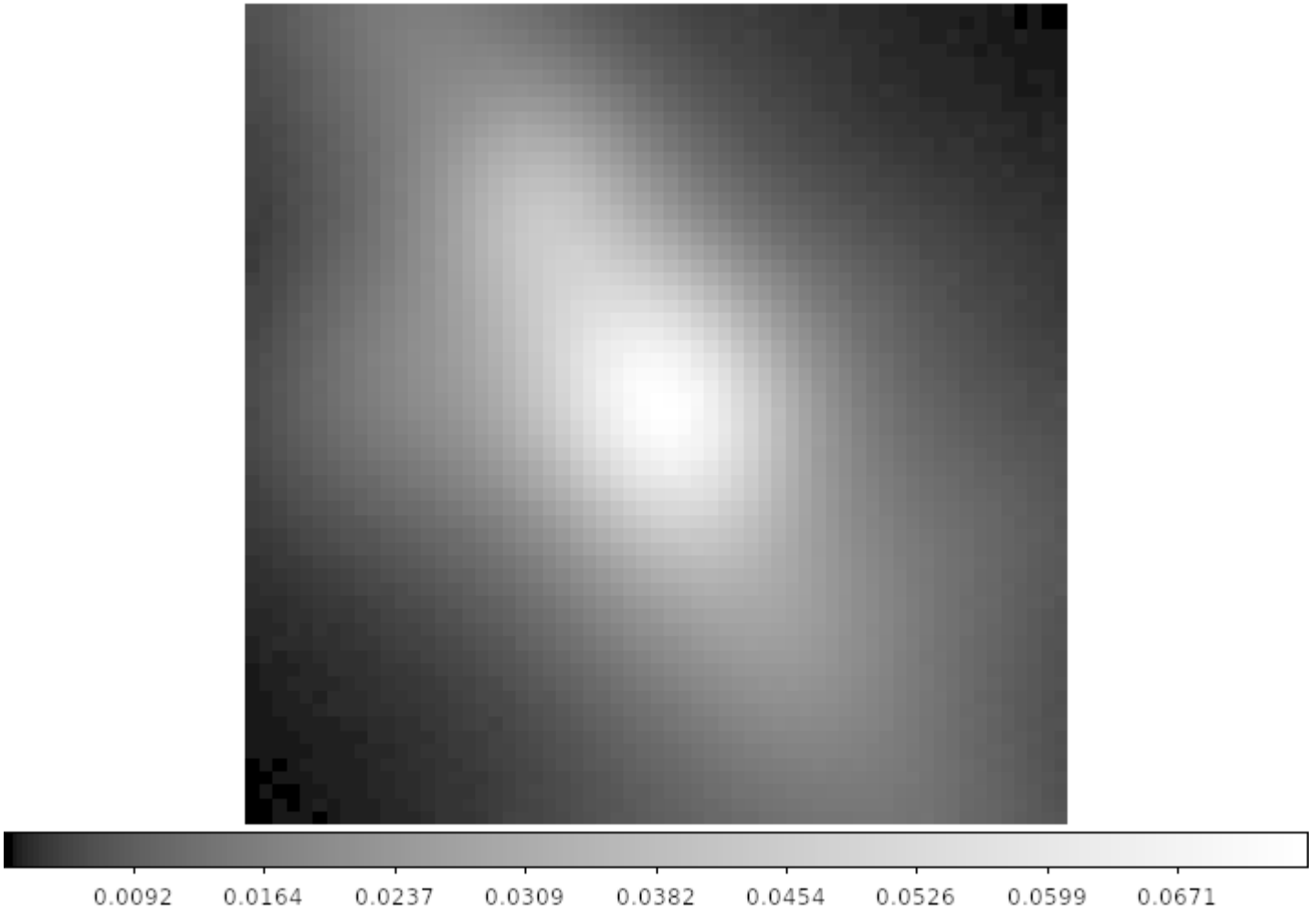
Planck Virtual Image Service

- We provide a stripped down TAP service to get TOI's.



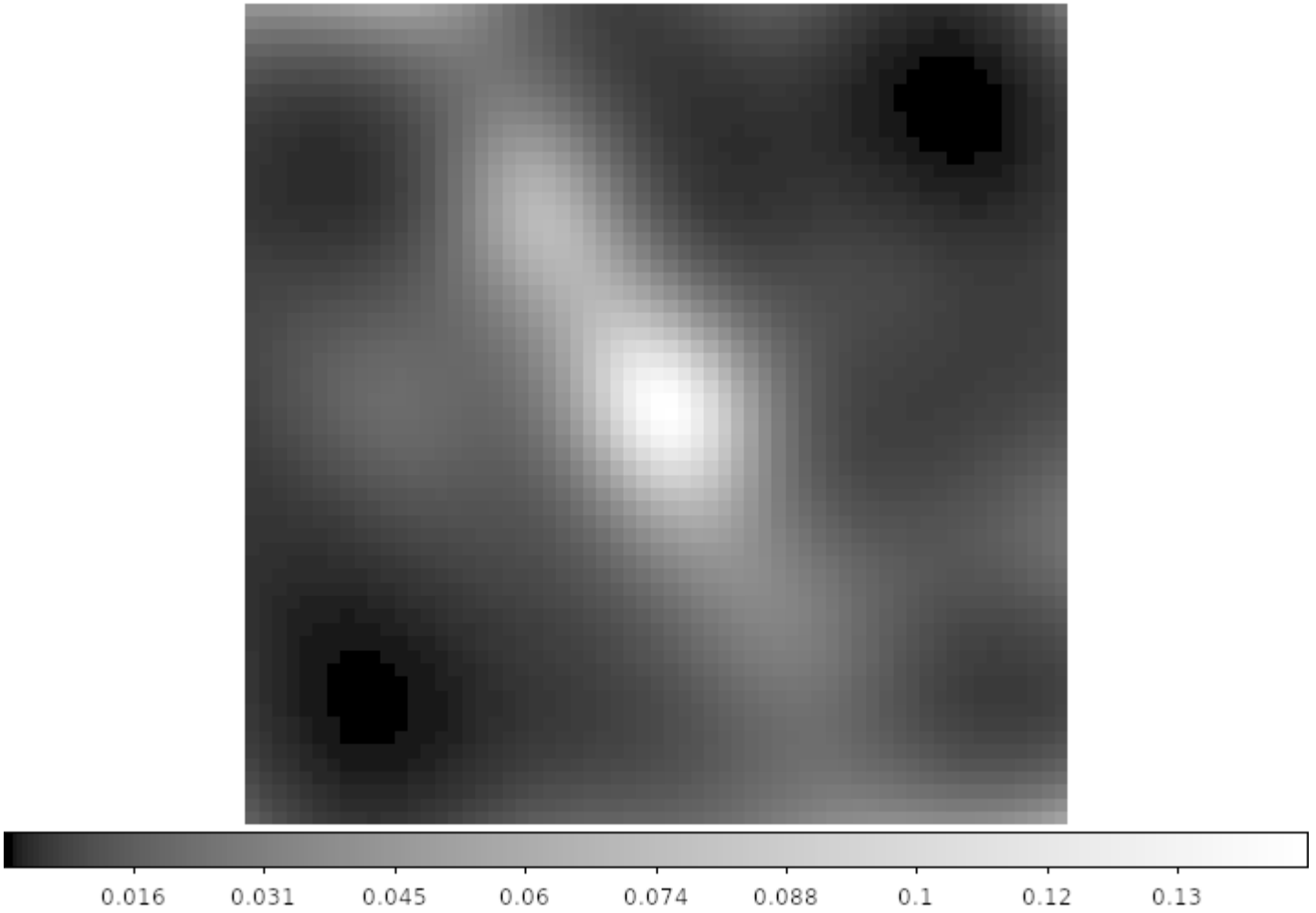
Minimap

- We also provide an image generation service.



Hires

- An image deconvolution service if you are willing to wait for it.



SIA

- Version 1 of SIA accommodated this kind of use case with support for Image Mosaicing Services.
- I tried implementing the service with SIA 2 protocols, but ran into problems.
- To be concrete, I will make some recommendations.

RANGE

- The backend HTM format makes it fast to search for shapes defined by great circle lines.
- Cone searches are a little more work, but not too bad.
- It would be some work to implement RANGE.

RANGE

- The backend HTM format makes it fast to search for shapes defined by great circle lines.
- Cone searches are a little more work, but not too bad.
- It would be some work to implement RANGE.
- RANGE is not necessary
  - It does not give a square on the sky, but users might think it does.

RANGE

- The backend HTM format makes it fast to search for shapes defined by great circle lines.
- Cone searches are a little more work, but not too bad.
- It would be some work to implement RANGE.
- RANGE is not necessary
  - It does not give a square on the sky, but users might think it does. Error-prone

RANGE

- The backend HTM format makes it fast to search for shapes defined by great circle lines.
- Cone searches are a little more work, but not too bad.
- It would be some work to implement RANGE.
- RANGE is not necessary
  - It does not give a square on the sky, but users might think it does. Error-prone
  - Tiling the sky with polygons is almost as easy as tiling the sky with ranges.

Recommendation

Remove RANGE

BOX

- The most common type of image that users will want are rectangles centered on a particular object.

BOX

- The most common type of image that users will want are rectangles centered on a particular object.
- SIA 1 only supported boxes and small circles.

BOX

- The most common type of image that users will want are rectangles centered on a particular object.
- SIA 1 only supported boxes and small circles.
- SIA 2 supports small circles, and polygons, but boxes must be constructed by making a polygon.

BOX

- The most common type of image that users will want are rectangles centered on a particular object.
- SIA 1 only supported boxes and small circles.
- SIA 2 supports small circles, and polygons, but boxes must be constructed by making a polygon.
  - Creating a polygon is error-prone, especially near the poles.

Recommendation

Add a BOX geometry with the same semantics of ADQL

COORD

- There are a number of additional parameters needed when generating images (e.g. iterations, detectors).
- I, or someone else, could easily end up using a keyword that has incompatible semantics with a future revision of SIA (e.g. DETECTOR).
- AccessData has the concept of COORD, where services could define keywords willy-nilly.

Recommendation

Add something like COORD to SIA 2 and forbid any extra parameters

Syntax

- POS parameters require spaces
  - With no URL-encoding (e.g curl's default), spaces can cause the rest of the query to be silently discarded.

Syntax

- POS parameters require spaces
  - With no URL-encoding (e.g curl's default), spaces can cause the rest of the query to be silently discarded.
  - Silent breakage is the worst kind of breakage.

Syntax

- POS parameters require spaces
  - With no URL-encoding (e.g curl's default), spaces can cause the rest of the query to be silently discarded.
  - Silent breakage is the worst kind of breakage.
- Polygon uses a straight list of numbers

Syntax

- POS parameters require spaces
  - With no URL-encoding (e.g curl's default), spaces can cause the rest of the query to be silently discarded.
  - Silent breakage is the worst kind of breakage.
- Polygon uses a straight list of numbers
  - Polygons are actually a list of pairs of numbers

Syntax

- POS parameters require spaces
  - With no URL-encoding (e.g curl's default), spaces can cause the rest of the query to be silently discarded.
  - Silent breakage is the worst kind of breakage.
- Polygon uses a straight list of numbers
  - Polygons are actually a list of pairs of numbers
  - Error prone

Syntax

- POS parameters require spaces
  - With no URL-encoding (e.g curl's default), spaces can cause the rest of the query to be silently discarded.
  - Silent breakage is the worst kind of breakage.
- Polygon uses a straight list of numbers
  - Polygons are actually a list of pairs of numbers
  - Error prone
- Arrays of strings are not supported
  - Planck users must be able to selectively choose a set of detectors '27m', '28s', ...

Structure

- Virtual image generation services can depend on input parameters that are best expressed as a structure.
  - Minimap and Hires have different parameters

Recommendation

Map key-value pairs to JSON5

JSON5

- JSON5 is a proposed extension to JSON that aims to make it easier for humans to write and maintain by hand
- JSON5 is a strict subset of ECMAScript 5
- Valid JSON is valid JSON5.

http://github.com/aseemk/json5

JSON5 Modifications to JSON

- Objects can be unquoted if they are valid identifiers
- Objects and arrays can have trailing commas
- Strings can be single quoted (').
- Numbers can include plus signs
- Infinity is a valid number
- Comments are allowed
- ...

JSON5 Example

```
{
    foo: 'bar',
    while: true,

    this: 'is a \
multi-line string',

    // this is an inline comment
    here: 'is another', // inline comment

    /* this is a block comment
       that continues on another line */

    hex: 0xDEADbeef,
    half: .5,
    delta: +10,
    to: Infinity,   // and beyond!

    finally: 'a trailing comma',
    oh: [
        "we shouldn't forget",
        'arrays can have',
        'trailing commas too',
    ],
}
```

Mapping key-value pairs to JSON5

- Because JSON5 removes a lot of the unnecessary baggage in JSON, it becomes feasible to use JSON5 syntax within a URL.
  - Put braces around the whole expression
  - Replace ampersands '&' with commas ','
  - Replace equals '=' with colon ':'

Mapping key-value pairs to JSON5

http://example.org/SIA2?TARGET='m101'&
POS=POLYGON:[[1,1],[1,2],[2,2]]&
BAND=[300,Infinity]&
TIME=['2012-01-01','2012-01-10']

```
{
    TARGET:'m101',
    POS:POLYGON:[[1,1],[1,2],[2,2]],
    BAND:[300,Infinity],
    TIME:['2012-01-01','2012-01-10']
}
```

Benefits

- This preserves the simple syntax that users like.
- The syntax scales to even very complex inputs
  - You could even post JSON5 documents wholesale
- In principle, we can validate it with json schema (http://json-schema.org/)

Benefits/Drawbacks

- This preserves the simple syntax that users like.
- The syntax scales to even very complex inputs
  - You could even post JSON5 documents wholesale
- In principle, we can validate it with json schema (http://json-schema.org/)
- Not many implementations of JSON5 parsers
  - Javascript
  - C++
  - Objective C
- Yet another way to interact with VO services