GAS Documentation

Release 0.5.dev555

Jaime Pineda, Erik Rosolowsky and Rachel Friesen

Mar 24, 2017

Contents

The GAS package provides a full data reduction pipeline of the data obtained by the GouldBelt Ammonia Survey (GAS) using the Green Bank Telescope.

It provides a uniform framework to reduce each region observed in a consistent fashion. At the moment, the GAS pipeline does the following:

- It runs the official calibration pipeline from GBT (https://github.com/nrao/gbt-pipeline/) on the raw data. This is done through an wrapper.
- Each spectral window and region (as defined in the observing log) are stored in a different directory.
- The imaging is done with our gridder, which generates a data cube using all the observations for a given window.
- A first look functionality. This allows for a quick look of the data, useful for quality assesment.
- Line fitting for all data is done using `pyspeckit`. In the case of NH3 the proper hyperfine model is used, while for the other lines a single Gaussian model is used.

Part I

Getting started

CHAPTER 1

Calibration

The calibration is done in a region by region basis, as listed in the observations summary. Go to the directory of the region and open an ipython session,

```
cd /lustre/pipeline/scratch/GAS/region_name
import GAS
GAS.gasPipeline.wrapper( region=region_name, startdate='2015-11-10', enddate='2015-12-10')
```

if redoing this step and need to download the latest observation summary file, remove the .csv file in the directory you are running it first).

The first command will load the calibration pipeline functions, while the second downloads an up-to-date version of the observations summary and reduces all spectral windows and beams observed for the requested region. Also, there is a function that runs the data reduction for all regions, just run the following commands,

```
cd /lustre/pipeline/scratch/GAS/
import GAS
GAS.gasPipeline.reduceAll()
```

The pipeline puts the calibrated data in named directories for each region (i.e., for Orion A, the calibrated data are in /lustre/pipeline/scratch/GAS/OrionA). Within the named directories, there will be folders for each molecular line in the observing setup.

Important: Check against the observing logs to see if all the sessions/scans are being reduced for all lines. If not, take note and then we can check what to do, or if we need to reduce them manually.

CHAPTER 2

Imaging

Go to the directory with the survey's images for the region to generate:

```
>>> cd /lustre/pipeline/scratch/GAS/images/region_name
```

and run the function to image the region want. This is an example for Serpens_Aquila

```
cd /lustre/pipeline/scratch/GAS/images/Serpens_Aquila
import GAS
GAS.run_grid_regions.grid_SerAqu()
```

The pipeline puts the cubes in an 'images' directory, with subdirectories for each region (i.e., for Orion A, the calibrated data are in /lustre/pipeline/scratch/GAS/images/OrionA).

Please do check what is the proper channel range needed to include all the data, while avoiding too many empty channels. Once you know the prefered range then modify the grid function in your local distribution of the pipeline and commit. Don't modify the files at Green Bank computers.

To know which imaging functions are available just type

```
import GAS
GAS.run_grid_regions.<Press TAB>
GAS.run_grid_regions.fits          GAS.run_grid_regions.grid_L1688    GAS.run_grid_regions.
↪grid_OrionB
GAS.run_grid_regions.grid_B18      GAS.run_grid_regions.grid_NGC1333  GAS.run_grid_regions.
↪grid_SerAqu
GAS.run_grid_regions.grid_L1455    GAS.run_grid_regions.grid_OrionA   GAS.run_grid_regions.
↪gridregion
```

CHAPTER 3

First Look

We have also created a first look pipeline that will allow us to baseline all the cubes and to create integrated intensity maps, peak temperature and first moment maps.

This is carried out with wrapper functions for each observed region, which can be found here `First Look wrappers`, where the underlying first look functionality is defined in the `First Look subpackage`.

Go to the directory with the survey's images and open an ipython session:

```
cd /lustre/pipeline/scratch/GAS/images/
import GAS.run_first_look
GAS.run_first_look.FirstLook_SerAqu(file_extension='_DR1')
```

Here the `file_extension` keyword is used to work with different versions of the data, e.g., DR1 and DR2.

To know which first look functions are available just type:

```
import GAS.run_first_look
GAS.run_first_look.<Press TAB>
```

Important: The output from these functions are FITS files which are good enough to assess the quality of the data. A higher quality product is (will be) created using the results from the line fitting as input to determine an optimal mask.

What does it do for you?

The First Look pipeline will take a FITS cube and remove the a polynomial baseline, for this you need to give the range of line free channels to be used and the polynomial order to be fit. Notice that we use a single channel range and polynomial order for the entire cube. Also, we use this same selection of channels to estimate he rms in each pixel.

Also, the pipeline will determine what is the line peak brightness at each pixel, for this you need to provide a single range of channels where the emission is constrained (this is done to avoid noise spikes). We also calculate the *first look* integrated intensity map, using the same channel range as for the peak brightness. *Caution* this integrated intensity map only takes into account the channel range provided, which is OK for typical lines however it means that the satelite components will be missed from the NH_3 maps. The solution for this is provided by `update_NH3_moment0`.

All products (integrated intensity, rms, and peak brightness maps as well as the baselined cube) are exported as FITS files.

How to use it on non-GAS data?

If you are using this package for your own non-GAS data, then you need to do the following.

TODO

CHAPTER 4

Line Fitting

The line fitting is done using pyspeckit We use the NH3 model included in `pyspeckit` to simultaneously fit NH_3 (1,1) and (2,2) to derive the parameters of NH3. This includes the centroi velocity, velocity dispersion, excitation temperature, kinetic temperature, and column density. Although we also observed NH_3 (3,3) we do not attempt to fit for the ortho-para ratio of NH_3, by default we set the ortho-para ratio to 0, therefore, we only report the para-NH3 column density.

Fitting GAS data

We have setup a couple of convenience functions to do the line fitting. For this, it assumes that the files are stored in a directory named 'region_name'. For example, if we want to fit the data for OrionA, then we do the following:

```
import GAS.PropertyMaps
GAS.PropertyMaps.cubefit(region='OrionA', vmin=5.6, vmax=13.7, do_plot=False, snr_min=3.0,
        multicore=40, file_extension='base_DR1', mask_function = None)
```

If you want to make some quick plots showing the fit results, then do the following

```
import astropy.units as u
import GAS.PropertyMaps
GAS.PropertyMaps.plot_cubefit(region='OrionA', distance=450*u.pc, dvmin=0.05, dvmax=0.7,
            vcmin=5.7, vcmax=12.7, file_extension='base_DR1')
```

The result from this line fitting will most likely include pixels that don't have a good fit (because of the default value for the minimum snr), however, we only need to do this fit once and now we clean-up the parameter fit to keep only those that are reliable.

CHAPTER 5

Improved integrated intensity maps

Given the constrains on the First Look functions to keep things simple, the integrated intensity map of the NH_3 lines only include the emission from the main component. As a solution, we have implemented a function that will use the results from the previous line fitting to determine the channel range with signal for the integrated intensity calculation. The function `update_NH3_moment0` is within `PropertyMaps` and it will also create a new FITS file for each NH_3 integrated intensity map.

Here is an example of its usage:

```
import GAS
import GAS.PropertyMaps
GAS.PropertyMaps.update_NH3_moment0(region_name='OrionA', file_extension='_DR1', threshold=0.
↪0125, save_masked=True)
```

Here the key parameter is the `threshold` used to identify the channels to select. Given the best fit model for the entire map we produce the cube with the emission predicted by the model. Then, we select all the voxels (3D "pixels") with brightness higher than `threshold`.

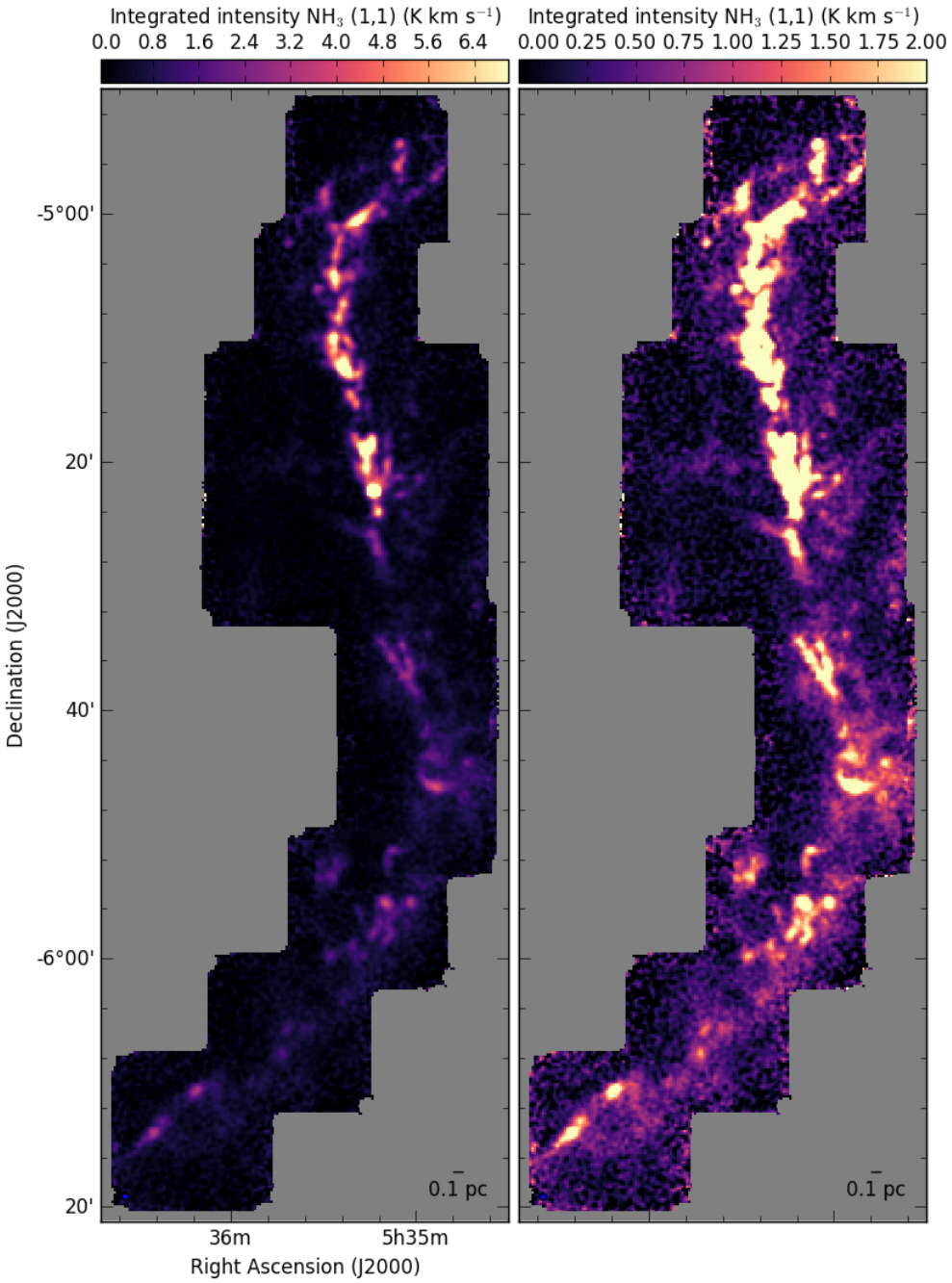
In addition, the function will calculate the uncertainty at each pixel, since the channel range will depend on the line profile.

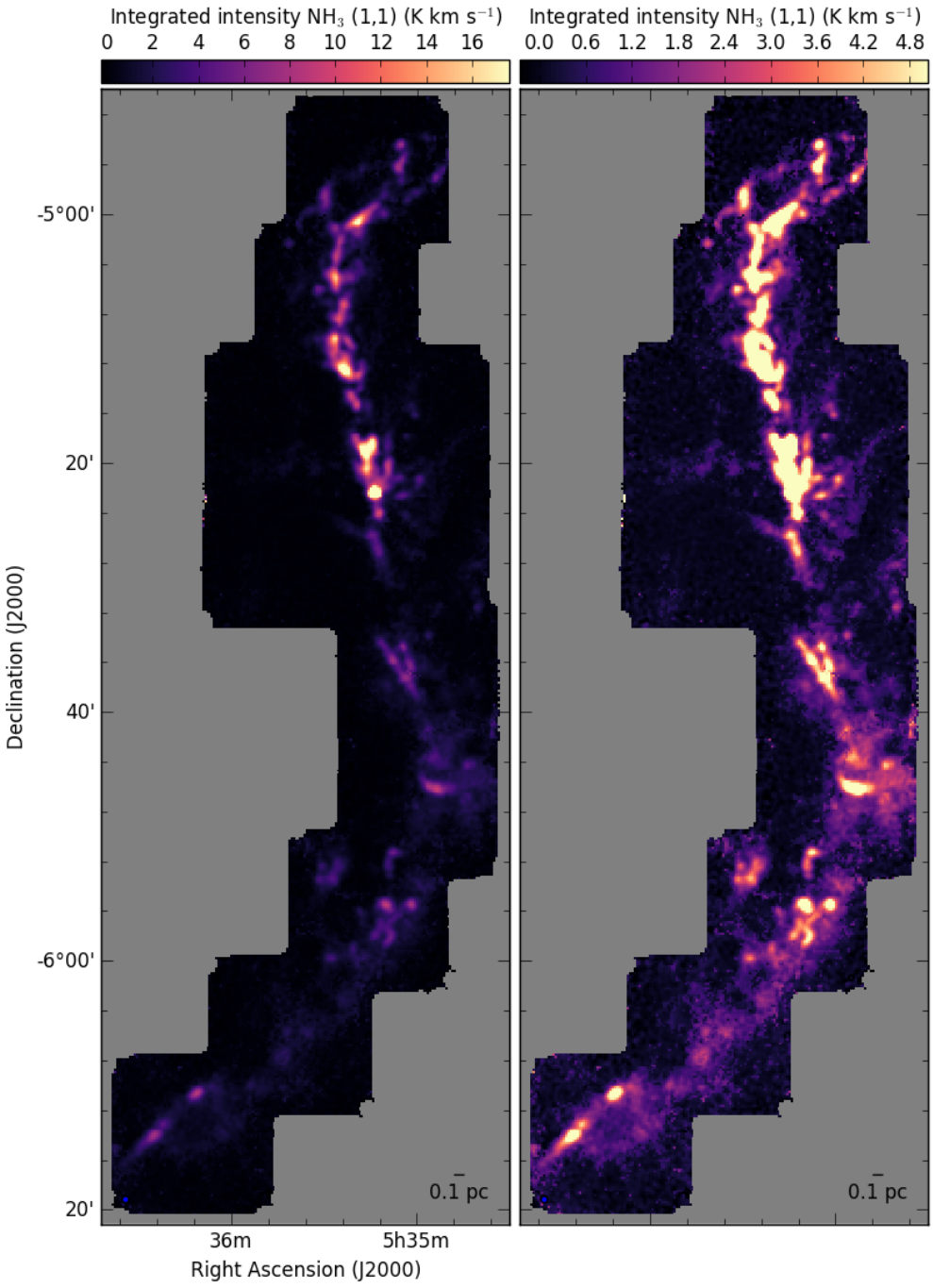
Is it really better?

Is this method really better than just selecting a fixed range of channels? Specially for the lower brightness lines.

We have taken the OrionA map to better show the difference. Here, we show the NH_3 integrated intensity map generated by the first look pipeline with two different color ranges: *Left* panel shows the map with a wide color range to show the full dynamic range of the region, *Right* panel shows the map with a narrower color range to enhance the fainter emission.

In comparison, similar color stretches are are shown for the improved inetgrated intensity maps, showing a much better noise level at the low brightness emission region:

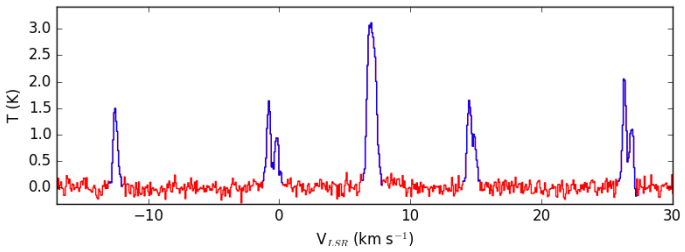

Integrated intensity NH3 (1,1) (K km s−1)
0.0 0.8 1.6 2.4 3.2 4.0 4.8 5.6 6.4
Integrated intensity NH3 (1,1) (K km s−1)
0.00 0.25 0.50 0.75 1.00 1.25 1.50 1.75 2.00
-5°00'
20'
40'
-6°00'
20'
Declination (J2000)
0.1 pc
0.1 pc
36m
5h35m
Right Ascension (J2000)


Integrated intensity NH3 (1,1) (K km s−1)
0 2 4 6 8 10 12 14 16
Integrated intensity NH3 (1,1) (K km s−1)
0.0 0.6 1.2 1.8 2.4 3.0 3.6 4.2 4.8
-5°00'
20'
40'
-6°00'
20'
Declination (J2000)
0.1 pc
0.1 pc
36m
5h35m
Right Ascension (J2000)

Caveats

This method for improving the integrated intensity also have some limitations. It depends on having a good model, which is usually the case, however, there are places where the emission is not absolutelly well fitted with a single component. In those cases, this method losses some of the flux.

Here we show an example of the method used on the OrionA data. We see that in some regions this works very well,



while in others the masking does not catch all the emission from faint components at different velocities.