The S^3-Tools User's Guide

François Levrier

University of Oxford
LERMA - Ecole Normale Supérieure de Paris

April 2011

Contents

Acknowledgements

I started working on the SKA Simulated Skies and what were to become the S^3-Tools as a SKADS postdoc with the subdepartment of Astrophysics of the University of Oxford. I therefore wish to heartily thank Steve Rawlings, who was head of the subdepartment at the time, for his constant enthusiasm and support. I also wish to thank those (Ian Heywood and Hans-Rainer Klöckner, mostly) who helped debugging the code, kept asking for new functionalities and never failed to say thanks when I replied to their calls for help. Thanks also to Richard Wilman and Danail Obreschkow for providing the S^3 databases that this work builds upon, and to Rense Boomsma for his HI templates. As far as optimization and parallelization are concerned, I am deeply indebted to Mihai C. Duta, Matteo Turilli and Fred Dulwich, from the Oxford e-Research Center (OeRC).

Chapter 1

Introduction

1.1 The SKA Simulated Skies (S^3)

The SKA Simulated Skies project (S^3) aims at building mock radio skies suitable for planning science with the SKA[1] and the pathfinder experiments such as the South African MeerKAT[2] and the Asutralian ASKAP[3]. The S^3 project comprises a number of simulations, of which essentially two are relevant for the present guide :
• S^3-SEX (**S**emi-**E**mpirical e**X**tragalactic) : a large-scale simulation of the extragalactic radio continuum sky covering a sky area of $20° \times 20°$, out to redshift $z = 20$, and down to 10 nJy, with a frequency coverage going from 151 MHz to 18 GHz. It also includes HI gas masses for star-forming galaxies. For a complete description, see [Wilman et al. (2008), Wilman et al. (2010)].
• S^3-SAX (**S**emi-**A**nalytical e**X**tragalactic) : a smaller-scale simulation of the extragalactic HI and CO line emissions, derived from the Millenium simulation [Springel et al. (2005)], which comes in two flavours. S^3-SAX-Sky is a skyfield simulation giving the apparent properties of galaxies, with a field of view that depends on the maximum redshift, while S^3-SAX-Box is a simulation of a cubic volume giving their intrinsic properties. Throughout this guide, S^3-SAX shall be understood as meaning S^3-SAX-Sky. For a complete description, see [Obreschkow et al. (2009)].
The S^3 set of simulations, performed at the University of Oxford in the framework of the European Union's SKADS program[4] are publically available on the S^3 website

http://s-cubed.physics.ox.ac.uk

and the databases built from these may be queried there. The S^3-Tools may then be used on query results to build maps and spectral data cubes.

1.2 The S^3-Tools

The S^3-Tools are a set of Python and C routines whose purpose is to provide user-friendly access to the S^3 simulations, in particular to build 2-dimensional images and 3-dimensional spectral data cubes that may be used as input for instrument simulators.
In order to use the S^3-Tools, you will need a number of python modules, a C compiler, and - as far as S^3-SAX is concerned - a set of templates. The next chapter describes the installation procedure in detail.
The S^3-Tools were designed to meet the following requirements :

[1]http://www.skatelescope.org/
[2]http://www.ska.ac.za/
[3]http://www.atnf.csiro.au/SKA/
[4]http://www.skads-eu.org/

• Versatile and reliable mapping of S^3 sources (both line and continuum) over a certain area of the sky and a certain range in frequency, at a given resolution
• Positioning of the output maps and cubes at a user-specified position in the sky (so as to be observable by a given array, for instance)
• Possible addition of ancillary signals such as Cosmic Microwave Background (CMB) emission, diffuse Galactic emission, signal from the Epoch of Reionization (EoR)...
• Inclusion of noise and Gaussian beam smearing
• Batch mode with a hint of parallel computing for heavy map-making jobs

To achieve these goals while keeping a user-friendly approach, Graphical User Interfaces (GUI) are used, serving as frontends to scripts which handle the actual work. Thus, the S^3-Map GUI is the way users will specify which maps to make out of a set of S^3 sources. To be more specific, the state of the GUI is passed as arguments to a python script command, either `SEX_map_script.py` or `SAX_map_script.py`. It is this command that actually builds the maps using lower-level routines.

1.3 Using this User's Guide

1.3.1 Organization

This guide is organized as follows :

• Chapter 2 deals with the installation of the S^3-Tools and additional data required to make maps.
• Chapter 3 explains the standard tests that can be performed to check the S^3-Tools work correctly.
• Chapter 4 explains how to query the S^3 databases so as to obtain query results that hold the necessary information for the S^3-Tools to work.
• Chapter 5 details the practical steps involved to make images and cubes from S^3 query results using the `S3Map` GUI.
• Chapter 6 goes one step deeper and focuses on the non-interactive scripts `SEX_map_script.py` and `SAX_map_script.py`.
• Chapter 7 delves into all the gory details about the algorithms used in the lower-level routines.
• Chapter 8 describes the solutions devised to split up large map-making jobs into smaller sub-jobs, so the S^3-Tools can be run on a cluster or on cloud-computing resources.

1.3.2 Conventions used in this document

Throughout this document, different fonts are used for different types of items:

• `File and directory names, terminal output, command line input`
• Graphical User Interfaces, buttons, menus and entry fields

Chapter 2

Installation

2.1 Prerequisites

In order to use the S^3-Tools, a number of python modules are required : os, sys, getopt, math, numpy, scipy, pyfits, time, random, types, ctypes, stat, Image, Tkinter, tkFileDialog, tkFont, matplotlib. The install script `install.py` contained in the distribution and described below first looks for the necessary packages and warns the user if some cannot be found. Depending on which package is missing, the limitations could go from not being able to build the plots when running tests to downright not being able to make maps at all, so a missing package is not to be taken lightly. For Mac OS X users, the necessary packages should all be included in the SciSoft package (version 2008.3.1 onwards).

2.2 Getting the S^3-Tools

Although they should find a permanent home on the S^3 website in the future, the S^3-Tools are currently hosted on an external website at `http://www.lra.ens.fr/~levrier/Recherche/S3/`. To get them directly, the following command may be used

```
%wget http://www.lra.ens.fr/~levrier/Recherche/S3/S3Tools.tgz
```

Once the archive is retrieved, it should be uncompressed in a temporary directory :

```
%tar -zxvf S3Tools.tgz
```

The archive contains the following :

• Two files : `README`, which is a shortened version of this chapter of the User's Guide, and `install.py`, which is the installation script described in 2.3
• Eight subdirectories : `C/`, `Config/`, `Doc/`, `Jobs/`, `Output/`, `Routines/`, `Tests/`, `Tmp/`

Each subdirectory serves a specific purpose :

• `C/` contains the C components of the S^3-Tools (see 2.5)
• `Config/` contains configuration files, such as `Config_Path.py`, which is discussed in 2.4, and `Config_Map.py`, which is mentioned in chapter 7.
• `Doc/` contains documentation and help files
• `Jobs/` is dedicated to holding shell scripts (e.g. to run batch jobs, see chapter 8)
• `Output/` is the repository for both input and output files

• `Routines/` contains all routines, both low- and high-level
• `Tests/` contains files for test runs of the S^3-Tools (see 3)
• `Tmp/` is where input tarballs are uncompressed to access query results. Users may create subdirectories under `Tmp/` to hold single source cubes for parallel jobs (see chapter 8).

2.3 Automatic installation using `install.py`

Installation can be performed easily via a dedicated python script. Type

```
%python install.py
```

at the command line prompt and follow the instructions. The script first checks for necessary modules[1] :

```
Running tests for Python modules...
os...            [ok]
[...]
matplotlib...    [ok]
```

It then inquires whether a former installation of the S^3-Tools exists or not

```
Is there an existing installation of the S3Tools [Y/N] ?
```

then prompts for the existing or desired installation directory

```
Give full path to (existing/desired) installation directory :
```

Depending on the answer to the first question, the script will either fully install the S^3-Tools in the specified directory, or upgrade the existing installation towards which it is pointed.

2.3.1 Full installation

In that case, all files within the archive are copied to the target directory and the user is prompted as to whether the script should proceed with configuration

```
Installing S3Tools in /Users/levrier/Desktop/S3try/
Do you want to proceed with path configuration now [Y/N] ?
```

If the answer is no, then the script skips this part and proceeds with making symbolic links in `Jobs/` to the python routines and shell scripts within `Routines/`. In that case, manual configuration will be needed (see 2.4). If the answer is yes, on the other hand, the user is prompted for a number of installation-specific pieces of information :

• The python executable to use[2] :
```
Give python executable [python] :
```

• The PDF viewer that will be used to open this guide from the `S3Map` GUI[3] :

[1]Note that the installation script will not halt if it finds a missing module (signalled by a `[NO]`). It is up to the user to check whether modules are missing, and act on it.
[2]This is the command used to launch the python interpreter from the command line. This is configurable here so that users may choose between different python versions.
[3]This is also how the viewer is launched from the command line, as the python routines basically spawn a UNIX child process using that command.

```
Give PDF viewer command for help files [acroread] :
```

• The paths to external data and tools (HI and CO templates, Global Sky Model, WCSTools, EoR templates,... see 2.6). If any of these are not installed, simply hitting the RETURN key will leave the default values indicated (and of course the corresponding functionality of the S^3-Tools will not be available):

```
Give full path to directory under which to find the HI and CO templates for SAX made by
Danail Obreschkow [/Path/To/Oxford_Templates/] :
Give full path to directory under which to find the HI templates for SAX made by
Rense Boomsma [/Path/To/Kapteyn_Templates/] :
Give full path to directory under which to find the WCS binaries [/Path/To/WCS/bin/] :
Give full path to directory under which to find Global Sky Model [/Path/To/GSM/] :
Give full path to directory under which to find the IQU continuum templates for SAX made
by Tigran Arshakian [/Path/To/MPIFR_Templates/] :
Give full path to directory under which to find the EoR signal templates made by Mario
Santos [/Path/To/IST_Templates/] :
```

These are used to automatically edit the configuration file `Config/Config_Path.py`, the original file being moved to `Config_Path.py.bak`. This configuration can be done manually, as explained in 2.4.

The installation script then proceeds with making the already mentioned symbolic links in `Jobs/`, and finally reminds the user of a final mandatory step, which is explained in 2.5 :

```
For the S3Tools to work, you need to compile the file S3.c under Routines/ into a
libS3.so shared object library.  Please refer to the README file.
```

2.3.2 Upgrade

As already mentioned, the `install.py` script can also upgrade an existing installation. Although users may want to do a backup of the `Routines/` subdirectory, it is not necessary, as the script will always make copies of the existing python files and shell scripts with a `.bak` extension[4]. In any case, it is not advised to do a full backup of the entire installation, especially if the `Output/` subdirectory is swollen. When upgrading, the script looks through the `.py` and `.sh` files inside the `Routines/` subdirectory of the distribution source, and seeks a file of the same name in the existing installation :

```
Upgrading S3Tools in /Users/levrier/Desktop/S3try/
```

If no such file can be found, the file is copied to the target directory, for instance

```
Adding ipol.py to your distribution...
```

If a corresponding file is found, a `diff` command is performed on these. If a difference is found, the existing file is backed up with a `.bak` extension and the new one is installed in its place, e.g.

```
ipol.py needs upgrading...
```

For the sake of users of earlier versions of the S^3-Tools, the script also looks to see if you already have the `C/`, `Tests/` and `Jobs/` subdirectories, which are new additions as of March 2011. If not, it creates them and fills them with the relevant files, i.e. C-related files into `C/` and symbolic links to the python files in `Routines/` into `Jobs/`. The `Tests/` subdirectory is created empty, as in the case of a full installation.

[4]Be warned, however, that any existing `.py.bak` file, e.g. from a previous upgrade, will be lost in the process !

Note for users of S^3-Tools predating March 2011:

It is important to note that when upgrading an existing installation, the script will not alter any of the files inside the `Config/` subdirectory, so as to avoid breaking working configurations. However, for users of an older (predating March 2011) installation of the S^3-Tools, that also means that the `Config_Path.py` will be missing a couple lines, which need to be manually inserted. These are :

```
# Directory where reference files are located and basic functionality tests are run
TESTS_DIR=S3TOOLS_PATH+"Tests/"
# Directory where the shell script jobs are placed for parallel computing
JOBS_DIR=S3TOOLS_PATH+"Jobs/"
```

2.4 Manual installation

If the user opted out of automatic configuration when prompted by `install.py`, the `Config_Path.py` file found under the `Config/` subdirectory needs to be edited manually. For simplicity, we shall assume that the S^3-Tools were installed in `~/S3Tools/`, that is, there is a `S3Tools/` directory under the home folder, with subdirectories `C/`, `Config/`, `Doc/`, `Jobs/`, `Output/`, `Routines/`, `Tests/` and `Tmp/`.
The variables to edit in `Config_Path.py` are the following :

`S3TOOLS_PATH` : Should be set to the full path to the S^3-Tools distribution. In our example, this would be `S3TOOLS_PATH="/Users/levrier/S3Tools/"`

`python` : Should be set to the python executable to use. Default is `"python"`, but users with various versions of python may want to use a specific one.

`PDFViewer` : Should be set to a command used to open PDF files from the terminal. It will be called to display this guide from the `S3Map` GUI when clicking the Help button.

`SAX_TEMPLATES_DIR` : Should be set to the full path of the directory under which to find the HI and CO templates for SAX made by Danail Obreschkow [University of Oxford] (see 2.6.1).

`KAPTEYN_TEMPLATES_DIR` : Should be set to the full path of the directory under which to find the HI templates for SAX made by Rense Boomsma [Kapteyn Institute] (see 2.6.2).

`GSM_DIR` : Should be set to the full path to the Global Sky Model distribution (see 2.6.3).

`WCS_DIR` : Should be set to the full path to the WCSTools binaries (see 2.6.4).

`IST_EOR_TEMPLATES_DIR` : Should be set to the full path of the directory under which to find the EoR signal templates from Mario Santos (Instituto Superior Técnico, Lisbon). This is currently in the test phase.

`MPIFR_TEMPLATES_DIR` : Should be set to the full path of the directory under which to find the IQU templates for SAX made by Tigran Arshakian [Max-Planck Institut fur Radioastronomie]. This is not implemented yet.

2.5 C components of the S^3-Tools

The S^3-Tools now include a tiny bit of C to speed up some computations[5]. The single C file is `S3.c`, located under the `C/` subdirectory, and it needs to be compiled into a shared object library file called

[5]To be precise, the single C function is used to perform spatial resolution degrading of emission template cubes, see...

`libS3.so`. To do so, a typical UNIX command would be

```
%gcc -Wall -O2 -fPIC -c S3.c && gcc -shared -Wl,-soname,libS3.so -o libS3.so S3.o
```

Mac OS X users may need to install the free cross-platform make `CMake`[6] and do a little more work. Instead of the one-liner above, the following steps should yield the desired shared library object :

```
%cd C
%mkdir build
%cd build
%cmake ../
%make
```

If all goes well, a `libS3.so` should appear under `C/build/`. This file must then be copied over to `Routines/` and a symbolic link to it created in `Jobs/`. In the `gcc` case, that would be done like this :

```
%cp libS3.so ../Routines
%cd ../Jobs
%ln -s ../Routines/libS3.so
```

and in the `CMake` case :

```
%cp libS3.so ../../Routines
%cd ../../Jobs
%ln -s ../Routines/libS3.so
```

2.6 External data and tools

By themselves, the S^3-Tools can only make data cubes from S^3-SEX query results, as the sources for that simulation are either pointlike or ellipses, which can be built on-the-fly and require no emission template[7]. For other functionalities, the user needs to download and install other elements :

• To make S^3-SAX data cubes, a set of position-position-velocity (PPV) emission templates is required. For HI emission, there are two sets available : one by Danail Obreschkow (University of Oxford), and one by Rense Boomsma (Kapteyn Institute). These are described in 2.6.1 and 2.6.2, respectively. For emission in the rotational lines of CO ($J = 1 \rightarrow 0$ to $J = 10 \rightarrow 9$), users will require the set of Oxford templates.
• The S^3-Tools include the possibility to add diffuse Galactic emission from 10 MHz to 100 GHz, via the Global Sky Model[8] [de Oliveira-Costa et al. (2008)], which is derived from a number of publically available large-area radio surveys. To use it, the GSM data and software are required, as well as the WCSTools. See 2.6.3 and 2.6.4 for details.

2.6.1 HI and CO templates [University of Oxford]

Installation

The templates can be downloaded as an archive `SAX_templates.tgz` from the LRA ftp server

[6] `http://www.cmake.org/`
[7] That may change in the future when infrared emission is taken into account, as spectral energy distribution (SED) templates may have to be used in this frequency domain.
[8] Also dubbed S^3-GAL on the S^3 website.

```
%wget ftp://ftp.lra.ens.fr/outgoing/levrier/SAX_templates.tgz
```

The user should be aware that the archive weighs over 2 GB, and 11.6 GB when uncompressed. It contains four files and four directories :

```
drwxr-xr-x    398 levrier  levrier       13532 Feb 15 20:09 FITS
drwxr-xr-x   8282 levrier  levrier      281588 May  8  2009 ascii
drwxr-xr-x   8282 levrier  levrier      281588 May 11  2009 line_profiles
-rw-r--r--      1 levrier  levrier      157500 May 11  2009 line_widths_CO.txt
-rw-r--r--      1 levrier  levrier      157500 May 11  2009 line_widths_HI.txt
drwxr-xr-x   8282 levrier  levrier      281588 May 11  2009 major_axis_profiles
-rw-r--r--      1 levrier  levrier      144900 May 11  2009 spatial_widths_CO.txt
-rw-r--r--      1 levrier  levrier      144900 May 11  2009 spatial_widths_HI.txt
```

After untarring the archive, the `Config_Path.py` file needs to be edited, so that `SAX_TEMPLATES_DIR` is set to the directory containing the above hierarchy (see 2.4).

Note for users of S^3-Tools predating March 2011:
Older (predating March 2011) installations of these templates did not include the `FITS/` subdirectory, into which FITS files for the templates are placed : The original template files are ascii (under `ascii/`) but they take a while to load into arrays, much longer than when read from FITS files. Consequently, when making S^3-SAX cubes, the script first looks for a FITS file, and uses that if it can find it. Otherwise, it uses the ascii file and takes the opportunity to convert it to FITS and save it under `FITS/` for future use. The downside of that is that FITS files take up a much bigger disk space (typically 8 to 9 times larger) than ascii files, so storage issues may occur.

About the templates

TO BE COMPLETED

2.6.2 HI templates [Kapteyn Institute]

Installation

The templates can be downloaded as an archive `KI_templates.tgz` from the LRA ftp server

```
%wget ftp://ftp.lra.ens.fr/outgoing/levrier/KI_templates.tgz
```

This archive weighs 265 MB, but almost 2.7 GB after decompression. It contains a `Profiles/` subdirectory, 3 `.txt` files and 1150 FITS files of the form `template_type_vasymp_inclination.fits` :

```
drwxrwxrwx   1177 levrier  levrier       40018 May  7  2008 Profiles
-rw-r--r--      1 levrier  levrier       46026 Jun 20  2008 hi_widths_0.2.txt
-rw-r--r--      1 levrier  levrier       45968 Jun 20  2008 hi_widths_0.5.txt
-rw-r--r--      1 levrier  levrier       46337 Feb  1  2010 spatial_widths.txt
-rwxr-xr-x      1 levrier  levrier      812160 Jan 22  2008 template_S0-Sab_100_0.fits
[...]
-rwxr-xr-x      1 levrier  levrier     1713600 Jan 22  2008 template_Sdm_75_90.fits
```

After untarring the archive, the `Config_Path.py` file needs to be edited, so that `KAPTEYN_TEMPLATES_DIR` is set to the directory containing the above hierarchy (see 2.4).

About the templates

The emission cubes all have the same pixel size (4”) and are assumed to be at a redshift of $z = 0.0045$, so that the physical size of a pixel is about 0.36 kpc. The third dimension of the cubes is velocity, and channels are 5 km.s^{-1} wide, but the number of channels varies from one template to the next to save space, although it is always odd to have a central channel[9]. The right ascension and declination axes also have an odd number of pixels (127) to have a central pixel. All template galaxies have a null position angle (major axis along the vertical).
The filenames reflect the galaxy’s type, asymptotical rotation velocity and inclination :
• **Type** : There are 5 galaxy types : S0-Sab, Sb-Sbc, Sc-Scd, Sd and Sdm (dwarf galaxies). This is important for the diameter, as all large galaxies (S0-Sab to Sd) are generated to have a 22 kpc diameter (measured at $\Sigma = 1$ M$_\odot$/pc^2), but some extend much further because of the faint tail in the radial HI profile. The template dwarf galaxies (Sdm) have a 16 kpc diameter.
• **Asymptotical rotation velocity** : Five mass and luminosity ranges are defined using the asymptotical rotation velocity V_a, which can take the values 100, 150, 200, 250 or 300 km.s^{-1} for S0-Sab to Sd galaxies, and 50, 75, 100, 125 or 150 km.s^{-1} for Sdm galaxies.
• **Inclination** : Possible inclinations range from 0° (face-on) to 90° (edge-on) in steps of 2°.
We thus recover the expected number of templates, as $1150 = 46 \times 5 \times 5$.

2.6.3 Global Sky Model

Installation

The Global Sky Model [de Oliveira-Costa et al. (2008)] data and routines can be downloaded from the GSM website at `https://www.cfa.harvard.edu/~adeolive/gsm/`. The direct download link is :

```
%wget http://xte.mit.edu/angelica/gsm/gsm.tar.gz
```

This archive weighs 109 MB (351 MB uncompressed) and contains the following hierarchy :

```
-rw-r--r--   1 levrier  levrier  122683392 Mar 17  2008 component_maps_23klocked.dat
-rw-r--r--   1 levrier  levrier  122683392 Mar 17  2008 component_maps_408locked.dat
-rw-r--r--   1 levrier  levrier  122683392 Mar 17  2008 component_maps_5deg.dat
-rw-r--r--   1 levrier  levrier       1045 Mar 17  2008 components.dat
-rw-r--r--   1 levrier  levrier       5413 Mar 17  2008 gsm.f
```

After untarring the archive, the `gsm.f` F77 file in the distribution must be replaced with a non-interactive one[10] available from

```
%wget http://www.lra.ens.fr/~levrier/Recherche/S3/gsm.f
```

Compile that file into a `gsm` executable, e.g.

```
%g77 gsm.f -o gsm
```

The last step is to edit the `Config_Path.py` file so that `GSM_DIR` be set to the directory containing the above hierarchy. Now, in order for the S^3-Tools to be able to use the GSM, the WCS Tools must also be installed (see below and 2.6.4).

[9]This central channel corresponds to $v_z = 0$ in the galaxy’s rest-frame.
[10]This is because by default, the executable resulting from the compilation of `gsm.f` prompts the user for a frequency at which to make a map, and a name for the output file. Both of these things must be dealt with by the S^3 map-making routines without user intervention, thus requiring that the `gsm` executable be non-interactive.

About the GSM

The Global Sky Model uses Principal Component Analysis (PCA) **CITE** of 11 publically available large-scale radio surveys (from 10 MHz to 94 GHz), to allow the building of Galactic emission maps at any intermediate frequency via interpolation. The output maps are in Galactic coordinates using the HEALPIX scheme **CITE** with $N_{\rm side} = 512$. As the cubes built from S^3 data are in equatorial coordinates, conversion is performed by the WCSTools, which therefore need to be installed (see 2.6.4). Assuming this is the case and that the user has requested a spectral data cube be made with p frequency channels $(\nu_1, \nu_2, \ldots, \nu_p)$, p GSM maps need to be built. This is done internally (using routines that can be found in `GSM_routines.py`).

2.6.4 WCSTools

The WCSTools are required for the correct inclusion of the Global Sky Model in S^3 data cubes. They can be downloaded from `http://tdc-www.harvard.edu/wcstools/`. After untarring the `wcstools-xxx.tar.gz` archive[11] into a directory, compilation of the one program that is required, `xy2sky` is done (under UNIX) in that directory by typing

```
%make xy2sky
```

The resulting `xy2sky` executable is located in the `bin/` subdirectory, and it is this one that the `WCS_DIR` variable in `Config_Path.py` should be set to. To compile all programs in the WCSTools, the above command should be replaced by

```
%make all
```

Troubleshooting regarding installation of the WCSTools should be dealt with according to instructions on the WCSTools website.

[11]where `xxx` refers to the version of the software.

Chapter 3

Standard tests

The S^3-Tools installation contains (in the `Routines/` subdirectory) a few shell scripts that can be run to test the correct working of the S^3-Tools (more precisely of `SAX_map_script.py` and `SEX_map_script.py`, see chapter 6). These scripts basically build some data cubes, which then need to be compared to reference files. Consequently, to use these scripts, user should download an archive containing these reference files from the LRA ftp server :

```
%wget ftp://ftp.lra.ens.fr/outgoing/levrier/Tests.tgz
```

Uncompressed in the `Tests/` subdirectory of the installation target directory, it reveals a couple input files `Tests/single-sax.tar.gz` and `Tests/single-sex.tar.gz`, as well as a `Tests/Reference/` hierarchy containing a number of `.fits`, `.png` and `.log` files, which are to be used as references. The source in `single-sax.tar.gz` is a single galaxy and the one in `single-sex.tar.gz` is a radio source which is made up of two 8.7" $\times$1.9" lobes and a central pointlike core.

3.1 S^3-SEX

To test making cubes out of S^3-SEX query results, a test script `run-SEX-tests.sh` is available under `Routines/`. To run it from the command line, users should type

```
%sh run-SEX-tests.sh
```

This script performs a series of `python SEX_map_script.py` commands with varying arguments (see chapter 6), to test the making of continuum emission data cubes :

• Emission with varying spatial resolution both with and without Gaussian beam smearing (see Fig. 3.1)
• Emission spectra from both lobes and central core of the single test source (see Fig. 3.2)
• Spatial window splitting : Using spatially resolved (0.1" pixels) continuum emission, four maps are made corresponding to the four quadrants of the source. This is to test that we can split a map in different patches without affecting pixel values. The four submaps are shown on the left panel of Fig. 3.13, and should be compared with the full map at the same resolution, which is the top left panel of Fig. 3.1.

3.2 S^3-SAX

To test making maps out of S^3-SAX query results, two test scripts are available. To run these from the command line, the user should type

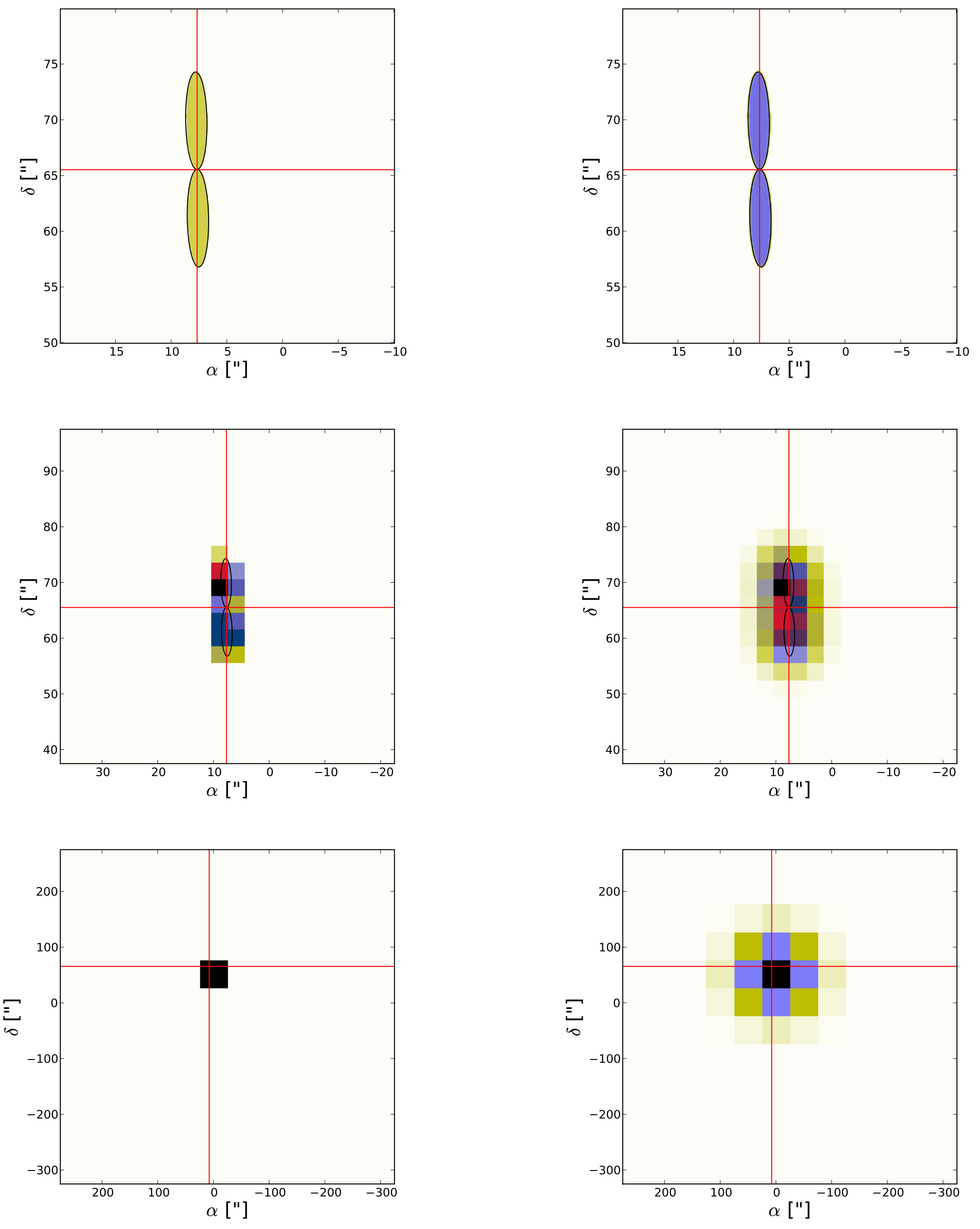


Figure 3.1: Continuum emission for the single source within `single-sex.tar.gz`, which is made up of two 8.7”×1.9” lobes and a central pointlike core, at varying spatial resolutions : pixel sizes are 0.1” (*top row*), 3” (*middle row*) and 40” (*bottom row*). In each row, the left figure shows the spatially resolved emission without Gaussian beam smearing, while the figure on the left includes the effect of a circular Gaussian beam with FWHM twice the size of the pixel. On each figure, two thin black ellipses represent the lobes and a red reticule shows the position of the core.

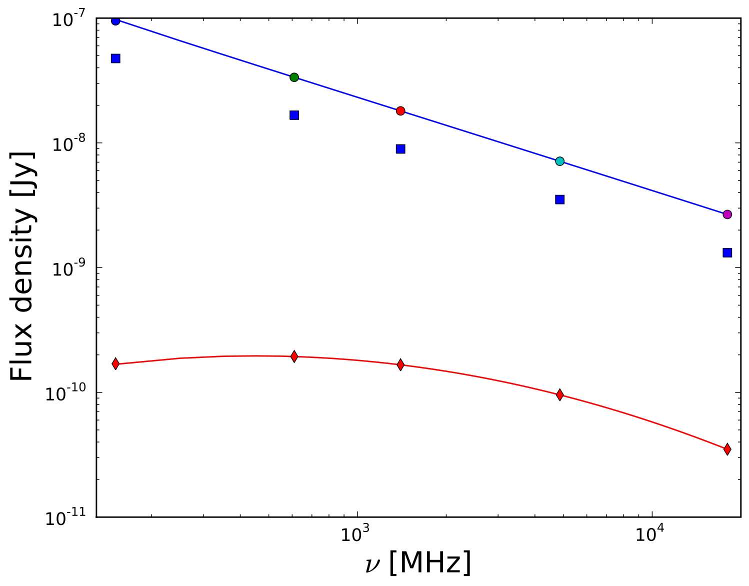


Figure 3.2: Spectral energy distribution (SED) over the full S^3-SEX frequency range 151 MHz - 18 GHz for the single source within `single-sex.tar.gz`. Represented are the flux densities for the entire field (blue line), the expected values at the reference frequencies for the lobes (blue squares) and the core (red diamonds), as well as the expected total intensities (colored circles). Also represented is the flux density for the pixel containing the sole core (red curve), as computed by the S^3 routines over the full frequency range (see chapter 7).

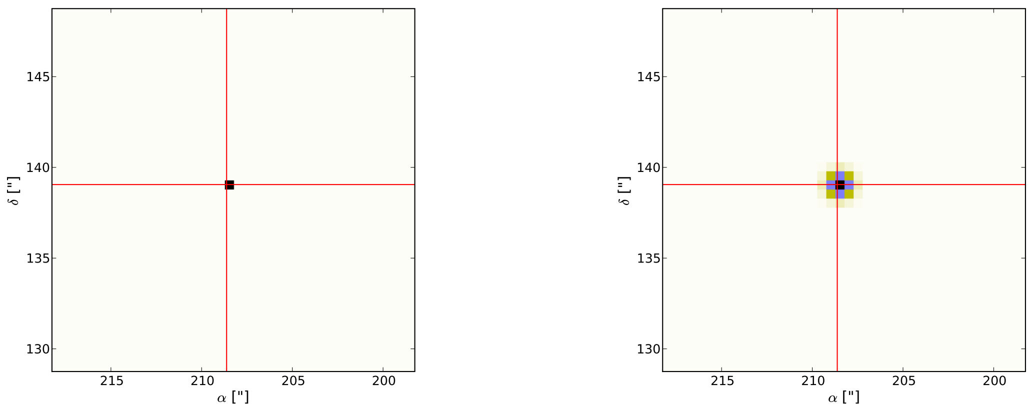


Figure 3.3: Spatially unresolved frequency-integrated HI emission for the single source within `single-sax.tar.gz`, using University of Oxford templates, with (*right*) and without (*left*) Gaussian beam smearing. The pixel size is 0.5" for a source with a 0.15" major axis, and the map size is 40×40 pixels. The circular Gaussian beam used on the right has a 1" full-width at half-maximum (FWHM). The red reticule shows the position of the source's center.

```
%cd Routines/
%sh run-SAX-Oxford-tests.sh
%sh run-SAX-Kapteyn-tests.sh
```

These scripts perform a series of `python SAX_map_script.py` commands with varying arguments (see chapter 6), to test the making of HI and CO emission data cubes with various spatial and spectral resolutions. The difference between the two being the set of templates used, either those made by Danail Obreschkow (University of Oxford), which can be used for both HI and CO, or those made by Rense Boomsma (Kapteyn Institute) for HI only. Obviously, for the scripts to work, the corresponding templates need to be installed and the `Config_Path.py` file configured so the scripts can find them (see 2.6.1 and 2.6.2)...

The tests performed by `run-SAX-Oxford-tests.sh` are the following :

• Spatially unresolved HI emission, with varying spectral resolution (channel widths 6, 60, 600 kHz and 2, 10 and 40 MHz), both with and without Gaussian beam smearing (see Figs. 3.3 and 3.4)
• Spatially resolved HI emission at varying spatial resolutions and constant 120 kHz spectral resolution, both with and without Gaussian beam smearing (see Fig. 3.5)
• Spatially unresolved CO emission in the $J = 1 \rightarrow 0$ to $J = 10 \rightarrow 9$ lines, with spectral resolution varying in proportion to rest frequency, without Gaussian beam smearing (see Fig. 3.6)
• Spatially resolved (1 milliarcsecond pixels) CO emission in the $J = 1 \rightarrow 0$ to $J = 10 \rightarrow 9$ lines, with spectral resolution varying in proportion to rest frequency, without Gaussian beam smearing (Fig. 3.6)
• Spatially integrated CO(1-0) emission spectra at varying spectral resolutions (see Fig. 3.7).
• Spatially unresolved HI and CO emission in all lines, with a 10 MHz - 200 GHz frequency coverage (see Fig. 3.8). The script also computes the integrated flux over the ranges in frequency correspondign to the various lines, for comparison with expected values :

```
* HI       ----> 2.38999999998e-08 Jy.km/s when expected integrated flux is 2.39e-08 Jy.km/s
* CO(1-0)  ----> 8.43e-06 Jy.km/s when expected integrated flux is 8.43e-06 Jy.km/s
* CO(2-1)  ----> 3.34999999999e-05 Jy.km/s when expected integrated flux is 3.35e-05 Jy.km/s
* CO(3-2)  ----> 7.47999999997e-05 Jy.km/s when expected integrated flux is 7.48e-05 Jy.km/s
```

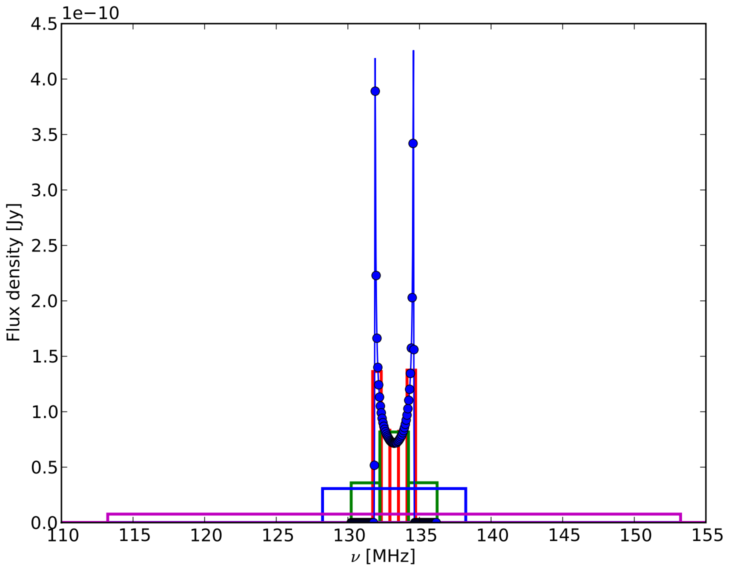


Figure 3.4: Spatially-integrated HI emission spectra for the single source within `single-sax.tar.gz`, using University of Oxford templates, for various spectral resolutions : 6 kHz (solid blue line), 60 kHz (blue circles), 600 kHz (red bars), 2 MHz (green bars), 10 MHz (blue bars) and 40 MHz (magenta bars).

```
* CO(4-3)  ----> 0.000131 Jy.km/s when expected integrated flux is 0.000131 Jy.km/s
* CO(5-4)  ----> 0.000201 Jy.km/s when expected integrated flux is 0.000201 Jy.km/s
* CO(6-5)  ----> 0.000284 Jy.km/s when expected integrated flux is 0.000284 Jy.km/s
* CO(7-6)  ----> 0.000376999999999 Jy.km/s when expected integrated flux is 0.000377 Jy.km/s
* CO(8-7)  ----> 0.000478999999999 Jy.km/s when expected integrated flux is 0.000479 Jy.km/s
* CO(9-8)  ----> 0.000547999999998 Jy.km/s when expected integrated flux is 0.000548 Jy.km/s
* CO(10-9) ----> 0.000441999999999 Jy.km/s when expected integrated flux is 0.000442 Jy.km/s
```

• Channel splitting : Using spatially unresolved HI emission with 60 kHz spectral resolution and 6 MHz bandwidth, three cubes are built corresponding to the lower 2.4 MHz of the band, the central 1.2 MHz and the upper 2.4 MHz. This is done to check that we recover the same emission as the one we get when building the whole 6 MHz cube in one go. This is essential for the correct working of the parallel map-making jobs, described in chapter 8 (see Fig. 3.9).
• Spatial window splitting : Using spatially resolved (1 mas pixels) HI emission at 120 kHz spectral resolution, four maps are made corresponding to the four quadrants of the source. This is to test that we can split a map in different patches without losing sources and that pixel values are not affected by the process. The four submaps are shown on the right panel of Fig. 3.13, and should be compared with the full map at the same resolution, which is the bottom left panel of Fig. 3.5.

The tests performed by `run-SAX-Kapteyn-tests.sh` are the following :

• Spatially unresolved HI emission, with varying spectral resolution (channel widths 6, 60, 600 kHz and 2, 10 and 40 MHz), both with and without Gaussian beam smearing (see Figs. 3.10 and 3.11)
• Spatially resolved HI emission at varying spatial resolutions and constant 120 kHz spectral resolution, both with and without Gaussian beam smearing (see Fig. 3.12)

These scripts use as an input the `single-sax.tar.gz` archive, which contains full data for a single galaxy from the S^3-SAX database. Some scripts take longer to complete than others, but the data cubes should

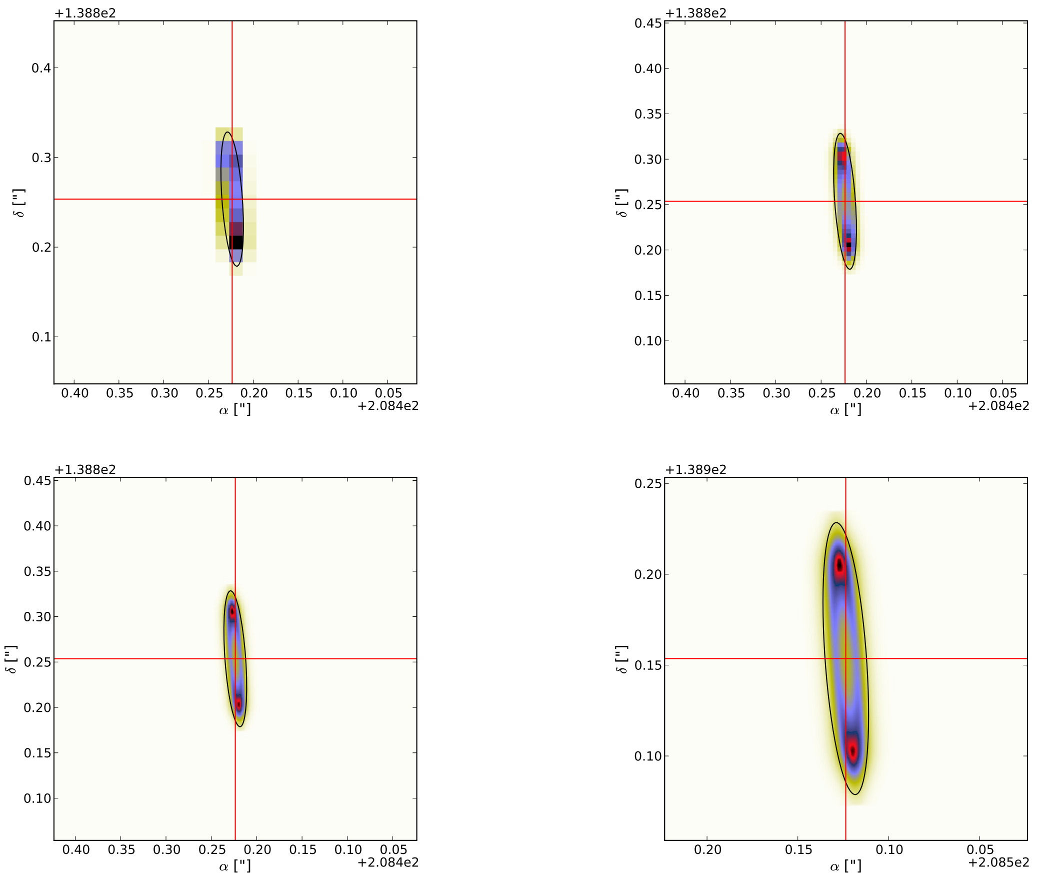


Figure 3.5: Spatially resolved frequency-integrated HI emission for the single source within `single-sax.tar.gz`, using University of Oxford templates. The pixel sizes are 15 milliarcseconds (*top left*), 5 milliarcseconds (*top right*), 1 milliarcsecond (*bottom left*) and 0.3 milliarcsecond (*bottom right*). The source has a 0.15" major axis and a 0.0228" minor axis, with a 4° position angle, and is represented by the thin black ellipse. The red reticule shows the position of the source's center.

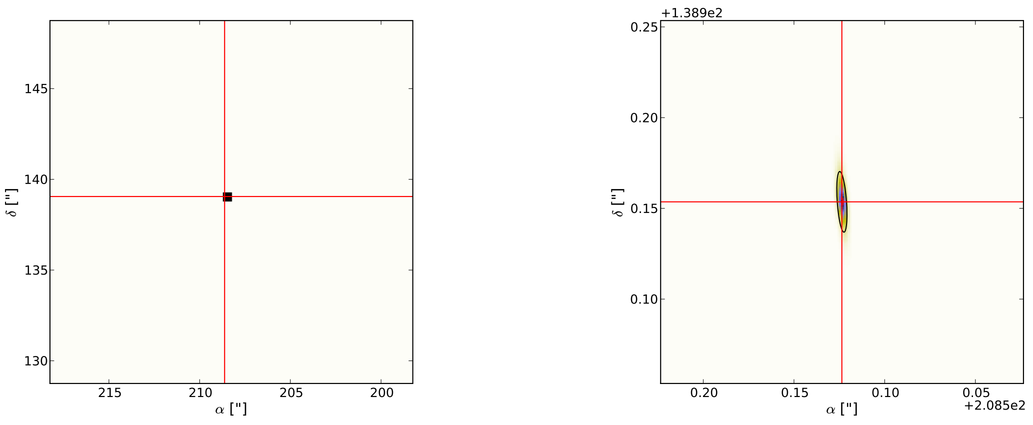


Figure 3.6: Spatially unresolved (*left*) and resolved (*right*) CO(1-0) frequency-integrated emission for the single source within **single-sax.tar.gz**. The pixel sizes are respectively 0.5 arcsecond and 1 milliarcsecond. The source has a 33.5 milliarcsecond major axis and a 5.1 milliarcsecond minor axis, with a 4° position angle, and is represented by the thin black ellipse. The red reticule shows the position of the source's center.

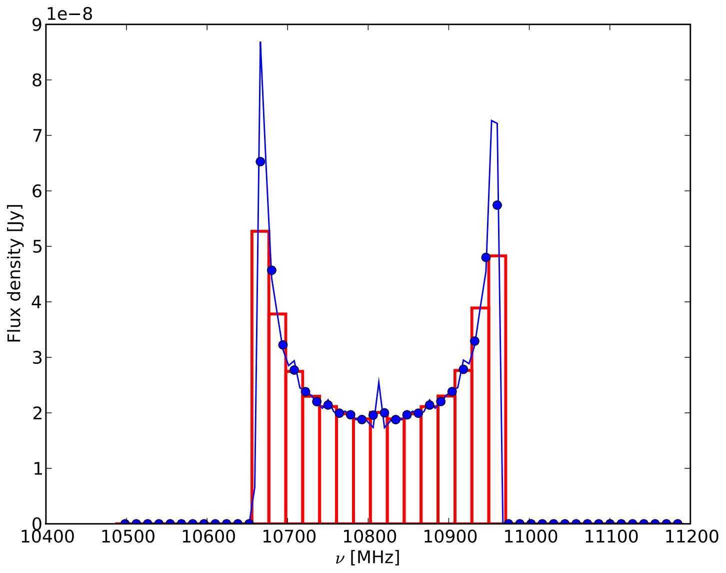


Figure 3.7: Spatially-integrated CO(1-0) emission spectra for the single source within **single-sax.tar.gz**, for various spectral resolutions : 7 MHz (solid blue line), 14 MHz (blue circles) and 21 MHz (red bars).

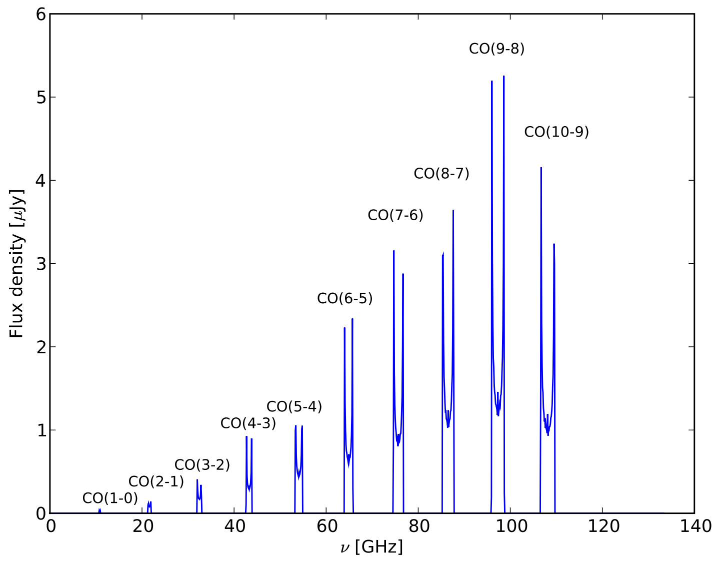


Figure 3.8: Spatially-integrated spectrum from the cube containing emission from all Hɪ and CO lines. The Hɪ line is too faint (see text) to be apparent on this plot.

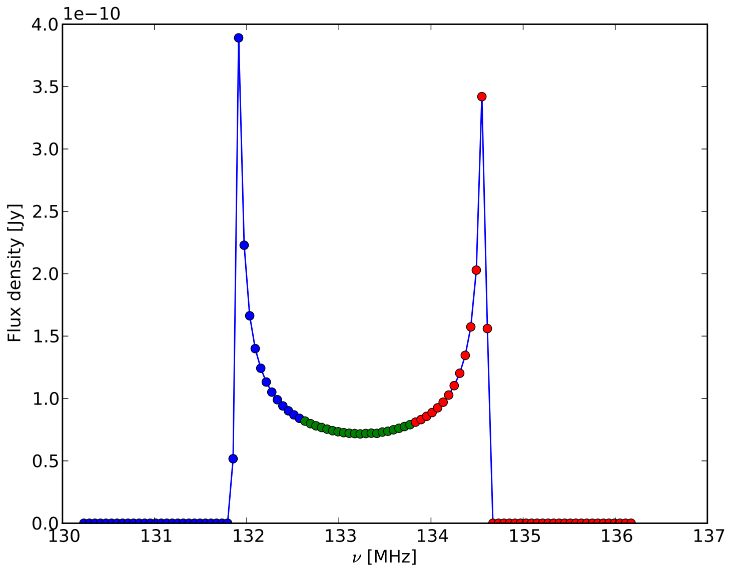


Figure 3.9: Spatially-integrated Hɪ emission spectra for the single source within `single-sax.tar.gz`, using University of Oxford templates, for various spectral coverages : full spectral line (solid blue line), lower frequency band (blue circles), central frequency band (green circles) and upper frequency band (red circles). The spectral resolution used is 60 kHz for all four data cubes.

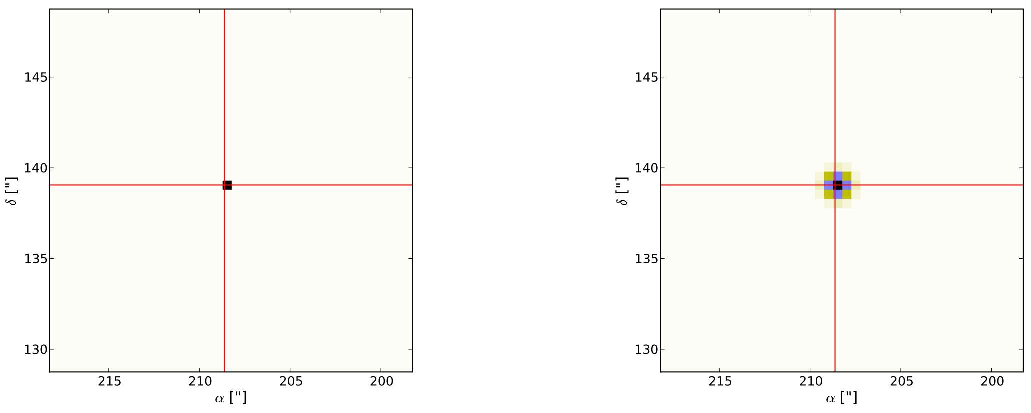


Figure 3.10: Spatially unresolved frequency-integrated HI emission for the single source within `single-sax.tar.gz`, using Kapteyn Institute templates, with (*right*) and without (*left*) Gaussian beam smearing. The pixel size is 0.5" for a source with a 0.15" major axis, and the map size is 40×40 pixels. The circular Gaussian beam used on the right has a 1" full-width at half-maximum (FWHM). The red reticule shows the position of the source's center.

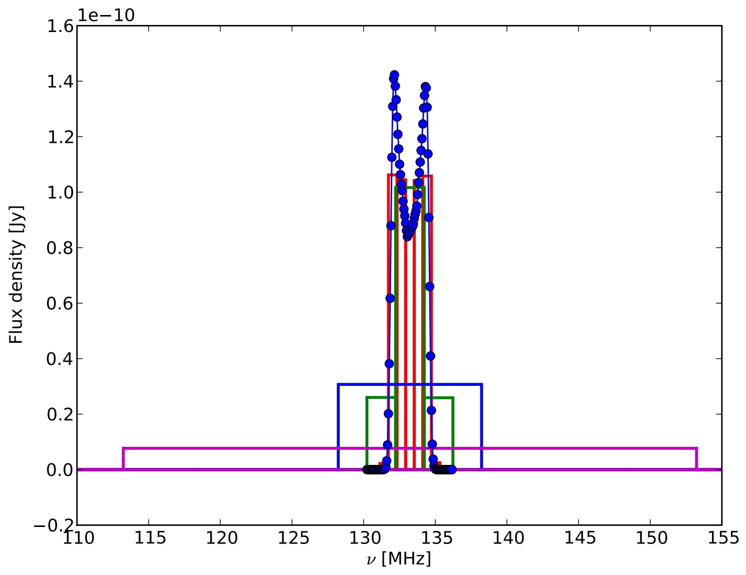


Figure 3.11: Spatially-integrated HI emission spectra for the single source within `single-sax.tar.gz`, using Kapteyn Institute templates, for various spectral resolutions : 6 kHz (solid blue line), 60 kHz (blue circles), 600 kHz (red bars), 2 MHz (green bars), 10 MHz (blue bars) and 40 MHz (magenta bars).

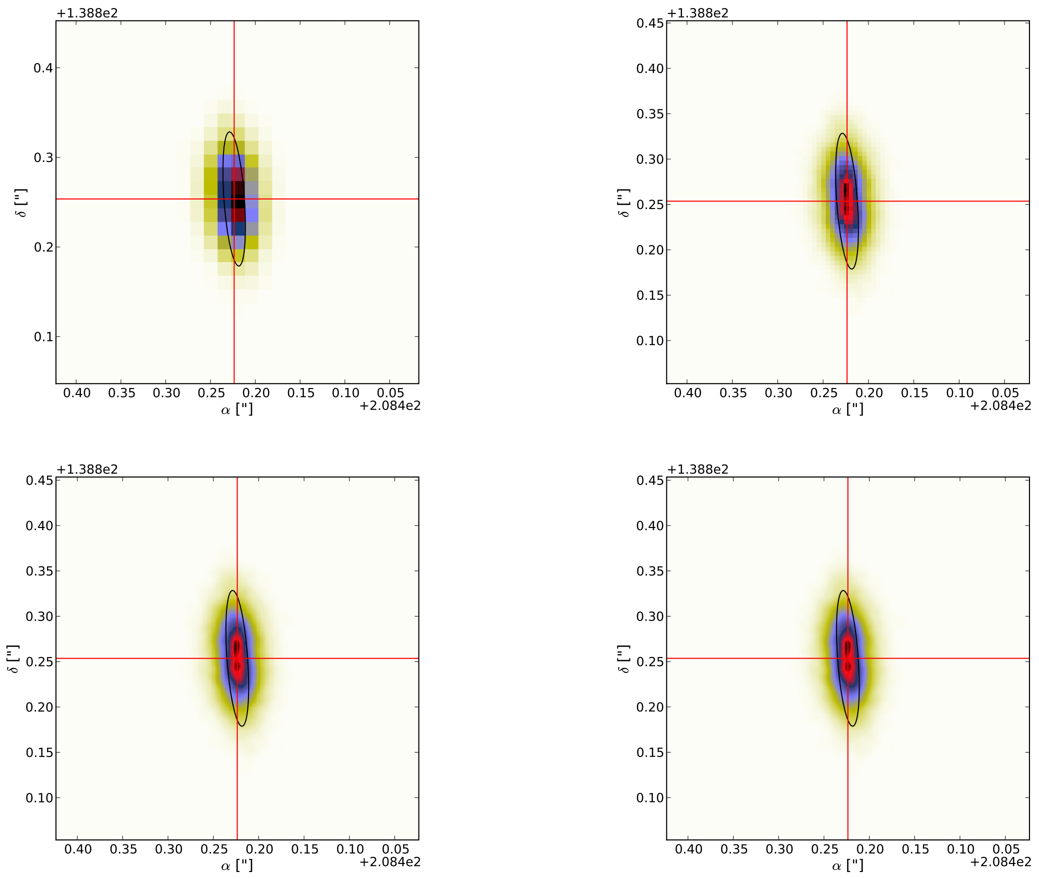


Figure 3.12: Spatially resolved frequency-integrated H I emission for the single source within `single-sax.tar.gz`, using Kapteyn Institute templates. The pixel sizes are 15 milliarcseconds (*top left*), 5 milliarcseconds (*top right*), 1 milliarcsecond (*bottom left*) and 0.3 milliarcsecond (*bottom right*). The source has a 0.15" major axis and a 0.0228" minor axis, with a 4° position angle, and is represented by the thin black ellipse. The red reticule shows the position of the source's center.

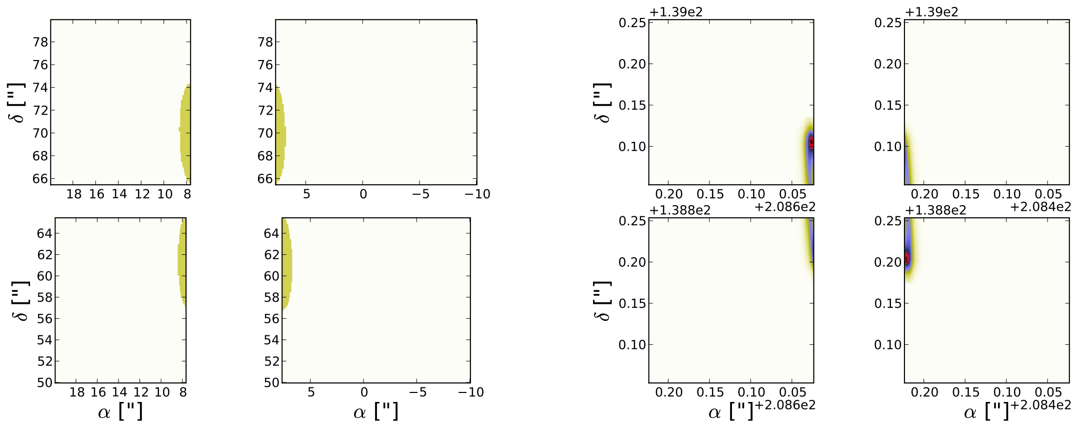


Figure 3.13: Demonstration of spatial window splitting on resolved continuum emission for the single source within `single-sex.tar.gz` (*left*) and on resolved HI frequency-integrated emission for the single source within `single-sax.tar.gz`, using University of Oxford templates (*right*). The four subplots show the four quadrants of the sources. The pixel size is 100 milliarcsecond in the S^3-SEX case and 1 milliarcsecond in the S^3-SAX case. Figures should be compared respectively with the top left panel of Fig. 3.1 and the bottom left panel of Fig. 3.5.

all be done in a few minutes. They are put in the `Tests/` subdirectory and then automatically compared - via UNIX `diff` commands - to the corresponding files in the `Tests/Reference/` subdirectory. If all goes well, there should not be any difference found. However, it is possible that small differences could be present (for instance due to different rounding schemes in floating point operations). Therefore, the scripts also build plots of spectra for the different cubes produced and save these plots as `.png` files in the `Tests/` subdirectory. Visual inspection of these plots with respect to the corresponding ones in the `Tests/Reference/` subdirectory should make it clear whether there is cause for alarm or not...
As a matter of fact, the figures in this chapter are simply `.eps` versions of these test plots, which can be built independently of making the FITS files, by running

`%python plot-SAX-Oxford-tests.py` or `%python plot-SAX-Kapteyn-tests.py`

Actually the shell scripts `run-SAX-Oxford-tests.sh` and `run-SAX-Kapteyn-tests.sh` simply run these python commands themselves after having built the FITS files.

3.3 Cosmic Microwave Background

To check that Cosmic Microwave Background (CMB) emission (a simple uniform blackbody emission at $T = 2.725$ K) can be correctly included in the output data cubes, the following shell script may be run

```
%sh run-CMB-tests.sh
```

This script builds a cube of unresolved emission centered on the S^3-SAX source `single-sax.tar.gz`, called `hi-co-cmb.fits`, covering all HI and CO lines from 100 MHz to 1 THz, including the CMB (i.e. the `--nocmb` flag has been removed from the calls to `SAX_map_script.py`), and a cube of the same size, called `cmb.fits`, centered on a different patch of sky, which thus recovers only CMB emission. The python script `plot-CMB-tests.py`, which is called at the end of `run-CMB-test.sh` then builds a difference cube and checks that correct integrated fluxes are recovered for each line :

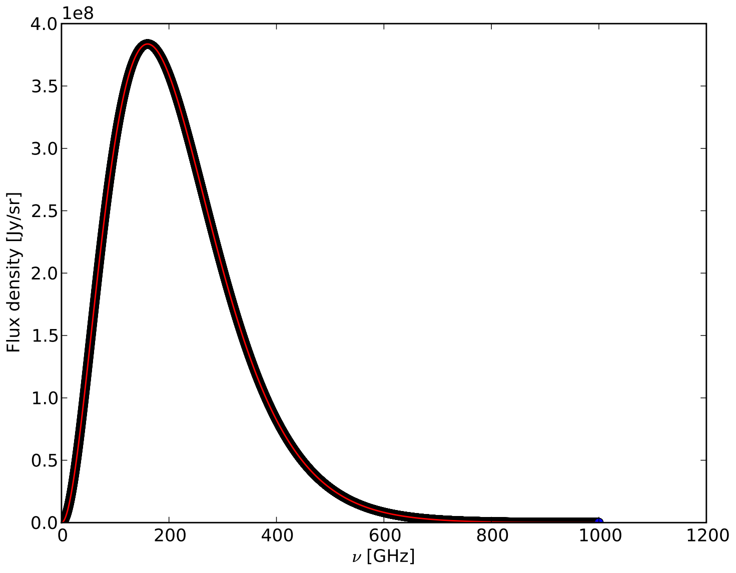


Figure 3.14: Flux density of the CMB emission computed from the CMB-only cube (blue circle) overlaid with the expected curve (solid red).

```
* HI       ----> 2.38999999998e-08 Jy.km/s when expected integrated flux is 2.39e-08 Jy.km/s
* CO(1-0)  ----> 8.43e-06 Jy.km/s when expected integrated flux is 8.43e-06 Jy.km/s
* CO(2-1)  ----> 3.34999999999e-05 Jy.km/s when expected integrated flux is 3.35e-05 Jy.km/s
* CO(3-2)  ----> 7.47999999997e-05 Jy.km/s when expected integrated flux is 7.48e-05 Jy.km/s
* CO(4-3)  ----> 0.000131 Jy.km/s when expected integrated flux is 0.000131 Jy.km/s
* CO(5-4)  ----> 0.000201 Jy.km/s when expected integrated flux is 0.000201 Jy.km/s
* CO(6-5)  ----> 0.000284 Jy.km/s when expected integrated flux is 0.000284 Jy.km/s
* CO(7-6)  ----> 0.000376999999999 Jy.km/s when expected integrated flux is 0.000377 Jy.km/s
* CO(8-7)  ----> 0.000478999999999 Jy.km/s when expected integrated flux is 0.000479 Jy.km/s
* CO(9-8)  ----> 0.000547999999998 Jy.km/s when expected integrated flux is 0.000548 Jy.km/s
* CO(10-9) ----> 0.000441999999999 Jy.km/s when expected integrated flux is 0.000442 Jy.km/s
```

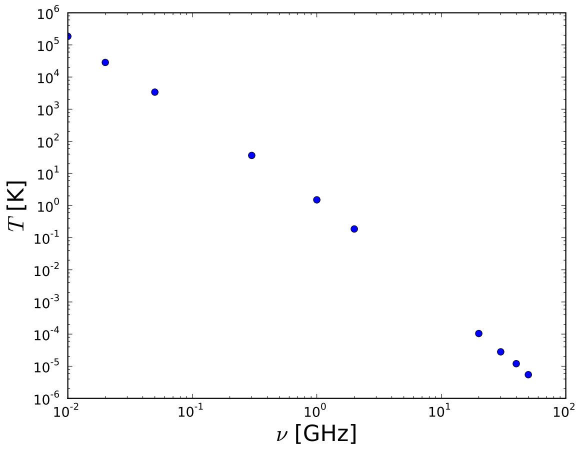
It also builds a plot of the emission from the CMB-only cube, in Jy/sr (see chapter 7), overlaid with the expected curve, and saves it under `cmb_spectrum.png` (Fig. 3.14).

3.4 Global Sky Model

To check that the Global Sky Model plugin for the S^3-Tools works, the following shell script may be run

```
%sh run-GSM-tests.sh
```

This script builds 10 single-pixel, single-frequency "images" at 10, 20, 50, 300 MHz and 1, 2, 20, 30 40 and 50 GHz, named `gsm-6am-10MHz.fits` to `gsm-6am-50GHz.fits`, centered on galactic coordinates $l = 11.3°$ and $b = 89.6°$, chosen to match those from Fig. 3 in [de Oliveira-Costa et al. (2008)] - and which are devoid of any signal from the single S^3-SAX test source. The spectrum derived from these (in brightness temperature [K] as a function of frequency [GHz] - see chapter 7) is saved into `gsm_spectrum.png`, which can be directly compared to the aforementioned figure (see the left panel of Fig. 3.15). Also built is a large scale (30° on a side) emission map from the GSM at 1 GHz. This is saved into `gsm-30deg-1GHz.fits`



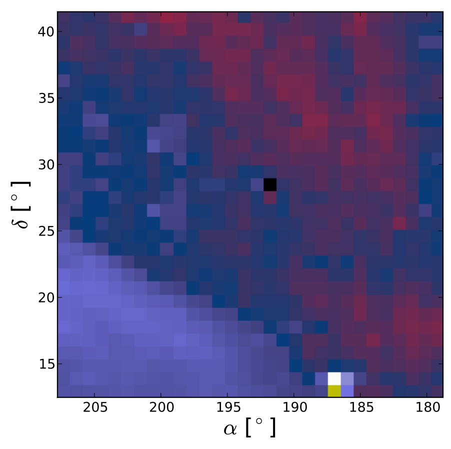
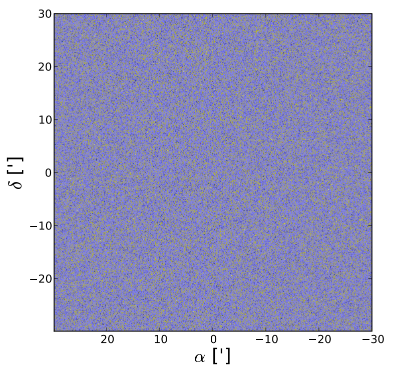


Figure 3.15: Flux density of the Global Sky Model emission at Galactic coordinates $(l, b) = (11.3°, 89.6°)$ (*left*) and Global Sky Model emission at $\nu = 1$ GHz on a $30° \times 30°$ sky area (*right*).



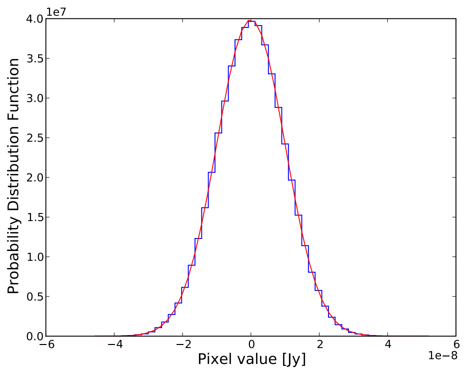


Figure 3.16: Example 10 nJy noise-only map over a square degree, with a 5" pixel size (*left*). Also shown (*right*) is the histogram (in blue) of the pixel values for that noise map, overlaid with a Gaussian PDF with $\sigma = 10$ nJy (red line).

and a figure of it is saved into `gsm_30deg.png` (see the right panel of Fig. 3.15). The `.png` files can be built separately once the FITS files are present under `Tests/` by running

```
%python plot-GSM-tests.py
```

which is actually run automatically at the end of `run-GSM-tests.sh`.

3.5 Noise

Noise can be added to cubes made via the S^3-Tools. This functionality can be tested with the following shell script :

```
%sh run-noise-tests.sh
```

This script uses `single-sax.tar.gz` as an input and build two maps, each covering the same square degree field outside the source, with a 10 nJy level noise (see chapter 7 for details). Both maps are 720×720 pixels, the pixel size being 5", and one is convolved with a 30" FWHM circular Gaussian beam. The maps are saved in `map-noise.fits` and `map-noise-smeared.fits`. On Fig. 3.16 are shown the unconvolved map and the histogram of its values, overlaid with a Gaussian PDF with $\sigma = 10$ nJy. These plots are built by the python script `plot-noise-tests.py` at the end of `run-noise-tests.sh`.

Chapter 4

Querying the S3 databases

4.1 Database structure

The S^3-SEX and S^3-SAX databases have similar structures, and consist of the following tables :

• For S^3-SEX : a Clusters table holding cluster properties, a Galaxies table holding galaxy properties (possibly including the cluster index to which they belong) and a Components table, which lists the properties of the several components that may make up a single galaxy, such as cores and radio lobes.
• For S^3-SAX : a galaxies_line table containing the apparent position and emission line properties of the galaxies and a galaxies_delu table containing the intrinsic properties of the galaxies as in the catalog from [de Lucia et al. (2006)].
For more information on the structure of the database and the attributes held in the various tables, please refer to the appropriate sections of the S^3 website.

4.2 Performing online queries

4.2.1 SQL query forms

The databases can be queried via SQL forms on the dedicated S^3-SEX and S^3-SAX sections of the S^3 website. More specifically, the query forms may be found at :

• `http://s-cubed.physics.ox.ac.uk/queries/new?sim=s3_sex` for S^3-SEX
• `http://s-cubed.physics.ox.ac.uk/queries/new?sim=s3_sax` for S^3-SAX

In both cases, queries should be written in the SQL format, which is quite straightforward to understand and implement, at least in this simple case. For instance, with S^3-SEX :

```
select * from Galaxies, Components where Galaxies.galaxy=Components.galaxy and Galaxies.right_ascension between
-0.02 and 0.02 and Galaxies.declination between -0.02 and 0.02;
```

will select all properties[1] (that's what the wildcard * stands for) from both Galaxies and Components tables where the center of the galaxy falls within the central $0.04^\circ \times 0.04^\circ$ of the simulation, and after identification of the object via the galaxy index, which is common to the Galaxies and Components tables.

Similarly, for S^3-SAX, a typical query line would be :

[1]These are called attributes in SQL-speak.

```
select * from galaxies_line where galaxies_line.ra between -0.02 and 0.02 and galaxies_line.decl between -0.02 and 0.02;
```

which would also select all properties from objects in the galaxies_line table which fall within the central $0.04^\circ \times 0.04^\circ$ of the simulation.

In both of these queries, the wildcard * after the select command means that all attributes will be returned. It can be replaced by a comma-separated list of attributes, keeping in mind that to make maps from these simulations, it is mandatory that some properties be included in the query, as the mapping algorithms require them. The next paragraph lists these mandatory attributes. For more complicated queries, users may refer to the S^3 website or the MySQL documentation at `http://www.mysql.com`

4.2.2 Mandatory attributes for mapping using the S^3-Tools

S^3-SEX

In the Galaxies table

- galaxy : Object identifier. This is the same index as the galaxy attribute in the Components table.
- redshift : Hubble flow redshift. At least one of redshift or modified_redshift is required
- modified_redshift : Apparent redshift (including peculiar velocities). At least one of redshift or modified_redshift is required
- distance : Comoving distance in Mpc. Only necessary if HI mapping is requested.
- sftype : Star formation type (None [0], Quiescent [1], Starburst [2])
- agntype : AGN type (None [0], Radio-Quiet [1], FRI [2], FRII [3], GPS [4])
- m_hi : Mass of HI gas in $M_\odot$. Only necessary if HI mapping is requested.

In the Components table

- galaxy : Object identifier. This is the same for all components of a single galaxy.
- right_ascension : Right ascension of the component's center, in degrees.
- declination : Declination of the component's center, in degrees.
- position_angle : Position angle of the component, in radians, counted positively east from north.
- major_axis : Major axis of the component, in arcseconds
- minor_axis : Minor axis of the component, in arcseconds
- i_151 : Base-10 logarithm $\log(I_{151})$ of the component's flux density at 151 MHz, in Jy.
- i_610 : Base-10 logarithm $\log(I_{610})$ of the component's flux density at 610 MHz, in Jy.
- i_1400 : Base-10 logarithm $\log(I_{1400})$ of the component's flux density at 1.4 GHz, in Jy.
- i_4860 : Base-10 logarithm $\log(I_{4860})$ of the component's flux density at 4.86 GHz, in Jy.
- i_18000 : Base-10 logarithm $\log(I_{18000})$ of the component's flux density at 18 GHz, in Jy.

S^3-SAX

In the galaxies_line table only

- ra : Right ascension of the galaxy's center, in degrees.
- decl : Declination of the galaxy's center, in degrees.
- zapparent : Apparent redshift (including peculiar velocities).
- diskpositionangle : Position angle of the galaxy, in radians, counted positively east from north.
- diskinclination : Inclination of the galaxy's disc on the line of sight, in radians (0 means face-on, $\pi/2$ means edge-on)
- distance : Comoving distance in Mpc.

To these, users should add the following for mapping the HI line :

- hiintflux : Velocity-integrated line flux of the HI line, in Jy.km.s^{-1}

• himajoraxis_10max : Radius at which the surface brightness in the line $\Sigma_{\rm HI}$ is at least 10% of its maximum value, in arcseconds.
• hiwidth50 : Width of the line at 50% of the peak luminosity, in km.s^{-1}.
• hiwidth20 : Width of the line at 20% of the peak luminosity, in km.s^{-1}.
• rmolc : H$_2$/HI surface density ratio at the disk center. Only necessary when using the University of Oxford templates (see 2.6.1).
• hubbletype : Numerical Hubble type along the RC2 sequence (-6,...,10). Only necessary when using the Kapteyn Institute templates (see 2.6.2).

For mapping the CO($J \rightarrow J-1$) line, users will obviously require the corresponding attributes :
• cointflux_1 : Velocity-integrated line flux of the CO(1-0) line, in Jy.km.s^{-1}
• cointflux_2 : Velocity-integrated line flux of the CO(2-1) line, in Jy.km.s^{-1}
• cointflux_3 : Velocity-integrated line flux of the CO(3-2) line, in Jy.km.s^{-1}
• cointflux_4 : Velocity-integrated line flux of the CO(4-3) line, in Jy.km.s^{-1}
• cointflux_5 : Velocity-integrated line flux of the CO(5-4) line, in Jy.km.s^{-1}
• cointflux_6 : Velocity-integrated line flux of the CO(6-5) line, in Jy.km.s^{-1}
• cointflux_7 : Velocity-integrated line flux of the CO(7-6) line, in Jy.km.s^{-1}
• cointflux_8 : Velocity-integrated line flux of the CO(7-7) line, in Jy.km.s^{-1}
• cointflux_9 : Velocity-integrated line flux of the CO(9-8) line, in Jy.km.s^{-1}
• cointflux_10 : Velocity-integrated line flux of the CO(10-9) line, in Jy.km.s^{-1}
• h2majoraxis_10max : Radius at which the surface brightness in the molecular lines $\Sigma_{\rm H_2}$ is at least 10% of its maximum value, in arcseconds. Needed for any J.
• cowidth50 : Width of the molecular lines at 50% of the peak luminosity, in km.s^{-1}. Needed for any J.
• cowidth20 : Width of the molecular lines at 20% of the peak luminosity, in km.s^{-1}. Needed for any J.
• rmolc : H$_2$/HI surface density ratio at the disk center. Needed for any J.

Finally, when parallel jobs are required, one should also add the id attribute to the list (see chapter 8).

4.3 Structure of query result files

When a query is completed, users are e-mailed with a link to download the results. These are saved in a gzipped tarball `idquery.tar.gz`, where `idquery` is a hash given by the S^3 database server at submission time. The tarball contains two plain text files `idquery.sql` and `idquery.result`.

In `idquery.sql`, the original SQL-formatted query is recalled, for instance :

```
select component,structure,right_ascension,declination,position_angle,major_axis,minor_axis
from Galaxies, Components where Galaxies.galaxy=Components.galaxy and Components.right_ascension
between -0.02 and 0.02 and Components.declination between -0.02 and 0.02;
```

`idquery.result` contains a first line listing the attributes retrieved, followed by the comma-separated list of query results proper. For instance :

```
component,structure,right_ascension,declination,position_angle,major_axis,minor_axis
53878311,1,0.00213,0.0182,0,0,0
53878312,2,0.00215,0.01942,0.016,8.748,1.917
53878313,2,0.00211,0.01699,0.016,8.748,1.917
```

Chapter 5

The S3Map GUI

5.1 The S3Map GUI and underlying scripts

For most users, making maps from query results may be done via S3Map, a graphical user interface (GUI) written in python, which is run from within the Routines/ subdirectory :

```
%cd Routines/
%python S3Map.py
```

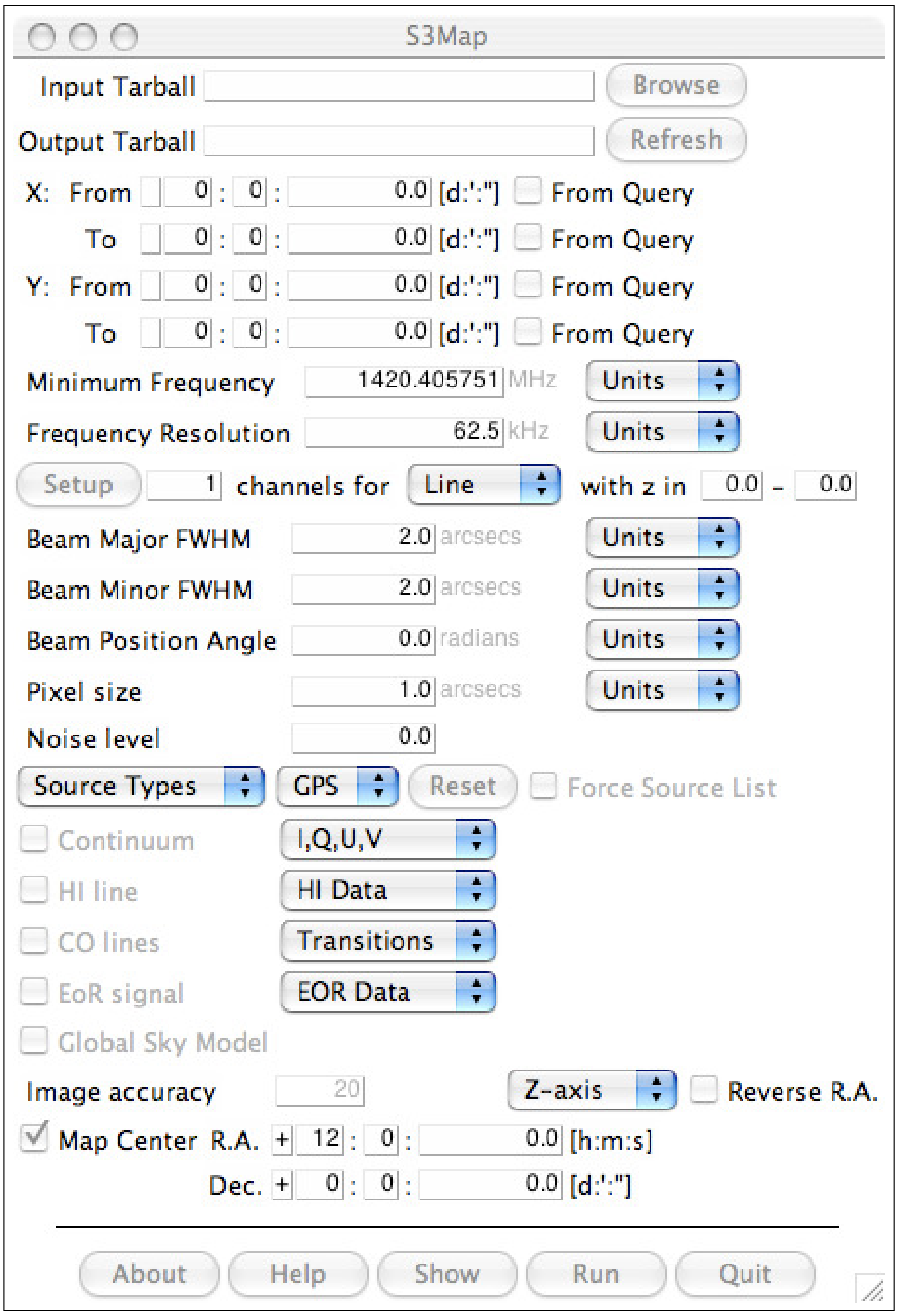
A window similar to the one presented in Fig. 5.1 should appear after a few moments. It is through this interface that users specify which maps to make. To be more specific, the state of the GUI is passed as arguments to a python script command, either SEX_map_script.py or SAX_map_script.py. It is this command that is in charge of actually building the maps using lower-level routines. The actual command that would be run in the current state of the GUI can be displayed in the terminal by clicking on the Show button. Users can then copy-paste this command into a terminal window and perform the same mapping non-interactively. This is a useful feature for building batch jobs or debugging purposes. See chapter 6 for more details on these scripts, and chapter 7 for lower-level routines.

5.2 S3Map options

The various buttons on the S3Map GUI have the following functions:

• Input Tarball entry field : Allows the user to specify the name of the tarball to use, which can be either a .tar or a .tar.gz file. This is the name without extension and by default the file is looked for in the directory given by the OUTPUT_DIR variable in Config_Path.py, unless the input tarball is selected by the Browse button (see below).
• Browse button : Allows the user to select the input tarball via a file selction popup window. The filename without extension is then reproduced in the Input Tarball entry field and the directory is retained to use for the output.
• Output Name entry field : Allows the user to specify the name of the output file(s). This is the name without any extension, as these are automatically added (only .fits for FITS files at the moment). The file is put in the same directory as the input tarball.
• X From, X To, Y From and Y To entry fields : Allow to specify the parameters of the rectangular area to map. Values are entered in degrees, arcminutes and arcseconds[1]. Be careful not to skip the sign entry field for each parameter.

[1]Be careful that right ascension is in degrees, arcminutes and arcseconds, and not hours, minutes, seconds...


S3Map

Input Tarball
Output Tarball
X: From 0 : 0 : 0.0 [d:':"] From Query
To 0 : 0 : 0.0 [d:':"] From Query
Y: From 0 : 0 : 0.0 [d:':"] From Query
To 0 : 0 : 0.0 [d:':"] From Query
Minimum Frequency 1420.405751 MHz
Frequency Resolution 62.5 kHz
1 channels for Line with z in 0.0 – 0.0
Beam Major FWHM 2.0 arcsecs
Beam Minor FWHM 2.0 arcsecs
Beam Position Angle 0.0 radians
Pixel size 1.0 arcsecs
Noise level 0.0
Source Types GPS Force Source List
Continuum I,Q,U,V
HI line HI Data
CO lines Transitions
EoR signal EOR Data
Global Sky Model
Image accuracy 20 Z-axis Reverse R.A.
✓ Map Center R.A. + 12 : 0 : 0.0 [h:m:s]
Dec. + 0 : 0 : 0.0 [d:':"]

Figure 5.1: The S3Map graphical user interface

• From Query checkboxes : If set, the adjacent entry field (one of X From, X To, Y From and Y To) is updated with the value read from the `WINDOW` keyword in the input `.query` file (see 5.3).
• Minimum Frequency entry field : Specifies the frequency of the center of the first spectral channel to consider. Units can be changed by using the Units drop-down menu.
• Frequency Resolution entry field : Specifies the width $\delta\nu$ of each spectral channel. Channels are assumed to be perfect bandpass filters[2]. Units can be changed by using the Units drop-down menu.
• Number of Channels entry field : Specifies the number of spectral channels to be considered.
• Setup button : Uses the value of the Number of Channels entry field and the range in redshifts specified by the with z in entry fields, along with the selection from the Line drop-down menu, to compute frequency coverage and update the Minimum Frequency and Frequency Resolution entry fields (see chapter 7).
• Beam Major FWHM entry field : Specifies the full width at half maximum (FWHM) of the (assumed elliptical Gaussian) beam to use, along its major axis. Units can be changed by using the Units drop-down menu.
• Beam Minor FWHM entry field : Specifies the full width at half maximum (FWHM) of the (assumed elliptical Gaussian) beam to use, along its minor axis. Units can be changed by using the Units drop-down menu.
• Beam Position Angle entry field : Specifies the position angle of the beam's major axis, counted positively east from north.Units can be changed by using the Units drop-down menu.
• Pixel size entry field : Specifies the size of a pixel's side. Pixels are considered square. Units can be changed by using the Units drop-down menu.
• Noise level entry field : Specifies the amount of (Gaussian) noise to be added to the map, in nJy. This value is the standard deviation of a zero-mean Gaussian random map which is then convolved by the same beam as the noise-free map
• Source Types drop-down menu : Allows the user to select the types of sources (Radio-quiet AGN, FRI radio-loud AGN, FRII radio-loud AGN, quiescent star-forming galaxies and starburst galaxies) that should be mapped. Of course, a source type that was not included in the query cannot be mapped. Only active when using S^3-SEX data.
• GPS drop-down menu : Allows the user to select whether GPS sources should be included. Only active when using S^3-SEX data containing such sources
• Continuum checkbox : If set, the output map or cube will include continuum emission. Currently only applies to S^3-SEX data cubes.
• I, Q, U, V drop-down menu : Allows to select whether the output map or cube will include an extra axis for polarization. Currently, only total (unpolarized) continuum intensity is available for S^3-SEX and no continuum is available for S^3-SAX.
• HI line checkbox : If set, the output map or cube will include HI line emission. In the case of S^3-SAX, the HI Data drop-down menu becomes active, allowing the user to choose between Oxford templates (see 2.6.1) and Kapteyn templates (see 2.6.2).
• CO lines checkbox : If set, the output map or cube will include CO line emission. Only active when using S^3-SAX data. In that case, the Transitions drop-down menu becomes active, allowing the user to choose which of the ten possible transitions to map. Only those present in the input tarball may be ticked on, and they are by default. The All and None buttons allow to select all CO lines or none of them.
• EoR signal checkbox : If set, the output cube will include the Epoch of Reionization signal. The data templates to be used can be selected via the EoR Data drop-down menu. Currently in the test phase.
• Global Sky Model checkbox : If set, the output cube will include the Global Sky Model signal (see 2.6.3).
• Image Accuracy entry field : Specifies the image accuracy parameter. Only applies to S^3-SEX data.
• Z-Axis drop-down menu : Allows to select the third axis units, the choice being velocity or frequency.
• Reverse R.A. Axis checkbox : If set, the output cube's right ascension axis will increase from left to right, instead of increasing from right to left, which is customary in astronomical images.
• Map Center Right Ascension and Map Center Declination entry fields : Allow to specify the parameters of the desired map center. Right ascension is entered in hours, minutes and seconds, declination in degrees,

[2]For a channel centered on ν_0, the bandpass is one for frequencies ν such that $|\nu - \nu_0| < \delta\nu/2$, and 0 otherwise.

arcminutes and arcseconds, recalling that 1 second in right ascension corresponds to 15 arcseconds. Be careful not to skip the sign entry fields. See **??** for more details.
• About button : Displays information about the S^3-Tools in a separate window.
• Help button : Launches the PDF viewer specified by the `PDFViewer` variable in `Config_Path.py` (see 2.4) and opens this User's guide.
• Show button : Displays, in the terminal, the full python command that will be issued should the user press the Run button with the current configuration of the GUI. This python command is based on the `SEX_map_script.py` and `SAX_map_script.py` scripts, as explained in chapter 6. Also displayed are the size of the would-be output map or cube, both in number of pixels and in estimated memory usage.
• Run button : Runs the map-making command with the current configuration of the GUI.
• Quit button : Exits the `S3Map` GUI.

5.3 Query files

During the develoment of the S^3-Tools, it appeared that some form of a "summary" for the `.result` files returned by database queries was necessary, e.g. to deactivate buttons on the GUI which are irrelevant for a certain set of results, or to initialize some fields with relevant values (such as spatial or redshift extents). These summaries are called query files and given the suffix `.query`. They are built by two dedicated routines `SEX_create_query_file` and `SAX_create_query_file`

Example of a S^3-SAX `.query` file :

```
DB=S3SAX
N_OBJECTS=1
CLUSTER_ITEMS=
GALAXY_ITEMS=id,galaxyid,box,ra,decl,distance
WINDOW=0.057951,0.057951,0.038626,0.038626
REDSHIFT=9.66001,9.66001
HI_RANGE=0.130232090735,0.136195883628
CO_1_0_RANGE=10.4982261246,11.1199930132
CO_2_1_RANGE=20.9964522492,22.2399860263
CO_3_2_RANGE=31.4946783739,33.3599790395
CO_4_3_RANGE=41.9929044985,44.4799720527
CO_5_4_RANGE=52.4911306231,55.5999650659
CO_6_5_RANGE=62.9893567477,66.719958079
CO_7_6_RANGE=73.4875828723,77.8399510922
CO_8_7_RANGE=83.985808997,88.9599441054
CO_9_8_RANGE=94.4840351216,100.079937119
CO_10_9_RANGE=104.982261246,111.199930132
```

Example of a S^3-SEX `.query` file :

```
DB=S3SEX
N_OBJECTS=1
CLUSTER_ITEMS=
GALAXY_ITEMS=galaxy,cluster,sftype,agntype,distance,modified_redshift
WINDOW=0.00211,0.00215,0.01699,0.01942
REDSHIFT=1.970476,1.970476
TYPES=FRI
GPS=NO
```

Chapter 6

Non-interactive scripts

6.1 General syntax

The S3Map GUI is actually a front-end to two python scripts called SEX_map_script.py (for S^3-SEX data) and SAX_map_script.py (for S^3-SAX data), whose arguments are derived from the state of the GUI. Consequently, it is possible to run map-making jobs from the command line without using S3Map. The arguments to SEX_map_script.py and SAX_map_script.py are described below in 6.4.2 and also in the corresponding help files, which can be printed out by issuing the following python command at the prompt:

```
%python SEX_map_script.py --help
```
or
```
%python SAX_map_script.py --help
```

The general syntax for these scripts is :

```
%python XXX_map_script.py --argument=value --keyword
```

where the arguments and keywords depend on which script is run.

6.2 Arguments and keywords to SEX_map_script.py

6.2.1 Arguments

The following lists the possible arguments to SEX_map_script.py and gives an explanation of the expected values for each :

• in : ID of the query results tarball to use as input. This is the filename minus the .tar.gz extension.
• out : Name of the FITS file to write the results to. This is to be given without the .fits extension.
• types [Optional] : A string of the form x,y,z,... giving the source type indices (eligible values are 1 to 5) to include in the map. Default is to include all source types that are present in the input tarball.
• window [Optional] : A string of the form min_x,max_x,min_y,max_y giving the spatial window to map. Values are to be given in arcseconds and the default is to use the query window from the input tarball (.query file see 5.3), in which case a conversion from degrees to arcseconds is performed internally.
• freqinfo : A string of the form min_freq,delta_freq,nchans giving the center frequency of the first channel in GHz, the width of each channel in GHz, and the number of channels.

• `bmaj` [Optional] : Full-width at half-maximum (FWHM) of the major axis of the (assumed Gaussian) beam, in arcseconds. Default is zero, which implies no beam convolution.
• `bmin` [Optional] : Full-width at half-maximum (FWHM) of the minor axis of the (assumed Gaussian) beam, in arcseconds. Default is zero, which implies no beam convolution.
• `bpa` [Optional] : Position angle of the beam, counted positively east from north, in radians. Default is zero.
• `noise` [Optional] : Level of the Gaussian noise to add, in nJy. The value specified is the standard deviation of a Gaussian-distributed random map with zero mean. The noise map is thereafter convolved by the same beam as the noise-free map. Default is zero (no noise).
• `pixelsize` : Size of a pixel's side, in arcseconds. Pixels are considered square.
• `dir` [Optional] : Full path to the directory where the input tarball should be found and the output cube should be put. The default is to use the directory specified by the `OUTPUT_DIR` variable in `Config_Path.py`.
• `continuum` [Optional] : A comma-separated list of Stokes parameters (I, Q, U, V) to specify which type of continuum is to be mapped. Currently, only `--continuum=I` is possible.
• `imageaccuracy` [Optional] : Minimum half-size, in pixels, of all single-source images. Default is 20.
• `rarefpos` [Optional] : Value of the right ascension at the image center, in degrees. Default is 180, that is 12 hours.
• `decrefpos` [Optional] : Value of the declination at the image center, in degrees. Default is 0.
• `zrange` [Optional] : A string of the form `z_min,z_max` specifying which range of redshift should be mapped. Default is to use the query range from the input tarball (`.query` file see 5.3).
• `ztype` [Optional] : Can be either `redshift` or `modified_redshift`, depending on whether the redshift to be used is the Hubble flow one or the apparent one. Default is to use the latter.
• `flux_limit` [Optional] : Allows to specify a flux limit, in nJy, so that any source whose fluxes at the reference frequencies are all below that cutoff are not mapped. Currently inactive.

6.2.2 Keywords

The following lists the possible keywords to `SEX_map_script.py` and gives an explanation of their effects when set :

• `hi` [Optional] : Set this option to map HI line emission.
• `freqaxis` [Optional] : Set this option to have a third axis in units of frequency (Hz). This is the default.
• `veloaxis` [Optional] : Set this option to have a third axis in units of velocity (km/s). This is only possible in conjunction with the `--hi` keyword.
• `gsm` [Optional] : Set this option to include the Global Sky Model emission (see 2.6.3).
• `reversera` [Optional] : Set this option to have a right ascension axis increasing from left to right. Default is the customary convention in astronomical images (right acension increases from right to left).
• `separate_maps` [Optional] : Set this option to make separate cubes for the different source types and non-S^3 data (CMB, Global Sky Model).
• `userefpos` [Optional] : Set this option to effectively use the map center direction specified via the `rarefpos` and `decrefpos` arguments.
• `nocmb` [Optional] : Set this option to skip the inclusion of the CMB emission (see 3.3) in the output cube. This is an option as the default is to always include it.

6.3 Arguments and keywords to `SAX_map_script.py`

6.3.1 Arguments

The following lists the possible arguments to `SAX_map_script.py` and gives an explanation of the expected values for each :

• `in` : ID of the query results tarball to use as input. This is the filename minus the `.tar.gz` extension.

• `out` : Name of the FITS file to write the results to. This is to be given without the `.fits` extension.
• `window` [Optional] : A string of the form `min_x,max_x,min_y,max_y` giving the spatial window to map. Values are to be given in arcseconds and the default is to use the query window from the input tarball (`.query` file see 5.3), in which case a conversion from degrees to arcseconds is performed internally.
• `freqinfo` : A string of the form `min_freq,delta_freq,nchans` giving the center frequency of the first channel in GHz, the width of each channel in GHz, and the number of channels.
• `bmaj` [Optional] : Full-width at half-maximum (FWHM) of the major axis of the (assumed Gaussian) beam, in arcseconds. Default is zero, which implies no beam convolution.
• `bmin` [Optional] : Full-width at half-maximum (FWHM) of the minor axis of the (assumed Gaussian) beam, in arcseconds. Default is zero, which implies no beam convolution.
• `bpa` [Optional] : Position angle of the beam, counted positively east from north, in radians. Default is zero.
• `noise` [Optional] : Level of the Gaussian noise to add, in nJy. The value specified is the standard deviation of a Gaussian-distributed random map with zero mean. The noise map is thereafter convolved by the same beam as the noise-free map. Default is zero (no noise).
• `pixelsize` : Size of a pixel's side, in arcseconds. Pixels are considered square.
• `dir` [Optional] : Full path to the directory where the input tarball should be found and the output cube should be put. The default is to use the directory specified by the `OUTPUT_DIR` variable in `Config_Path.py`.
• `hi` [Optional] : Set to either `Oxford` or `Kapteyn` to specify which set of HI emission templates are to be used. Default is not to request HI mapping, which corresponds to `--hi=0`
• `rarefpos` [Optional] : Value of the right ascension at the image center, in degrees. Default is 180, that is 12 hours.
• `decrefpos` [Optional] : Value of the declination at the image center, in degrees. Default is 0.
• `zrange` [Optional] : A string of the form `z_min,z_max` specifying which range of redshift should be mapped. Default is to use the query range from the input tarball (`.query` file see 5.3).
• `hash` [Optional] : Set to a user-chosen string which is meant to identify several map-making jobs as part of the same project. See chapter 8 for more details.

6.3.2 Keywords

The following lists the possible keywords to `SAX_map_script.py` and gives an explanation of their effects when set :

• `co_1_0` [Optional] : Set this option to map CO(1-0) line emission.
• `co_2_1` [Optional] : Set this option to map CO(2-1) line emission.
• `co_3_2` [Optional] : Set this option to map CO(3-2) line emission.
• `co_4_3` [Optional] : Set this option to map CO(4-3) line emission.
• `co_5_4` [Optional] : Set this option to map CO(5-4) line emission
• `co_6_5` [Optional] : Set this option to map CO(6-5) line emission.
• `co_7_6` [Optional] : Set this option to map CO(7-6) line emission.
• `co_8_7` [Optional] : Set this option to map CO(8-7) line emission.
• `co_9_8` [Optional] : Set this option to map CO(9-8) line emission.
• `co_10_9` [Optional] : Set this option to map CO(10-9) line emission.
• `freqaxis` [Optional] : Set this option to have a third axis in units of frequency (Hz). This is the default.
• `veloaxis` [Optional] : Set this option to have a third axis in units of velocity (km/s). This is only possible when one and only one of `hi`, `co_1_0`, `co_2_1`, `co_3_2`, `co_4_3`, `co_5_4`, `co_6_5`, `co_7_6`, `co_8_7`, `co_9_8` or `co_10_9` is set.
• `gsm` [Optional] : Set this option to include the Global Sky Model emission (see 2.6.3).
• `reversera` [Optional] : Set this option to have a right ascension axis increasing from left to right.
• `separate_maps` [Optional] : Set this option to make separate maps for the S^3 data and non-S^3 data (CMB, Global Sky Model).

• `userefpos` [Optional] : Set this option to effectively use the map center direction specfied via the `rarefpos` and `decrefpos` arguments.
• `nocmb` [Optional] : Set this option to skip the inclusion of the CMB emission (see 3.3) in the output cube. This is an option as the default is to always include it.

6.4 Running a script

6.4.1 Typical script commands

6.4.2 Terminal output while running a script

What the `SEX_map_script.py` and `SAX_map_script.py` scripts actually do is to first look up the input. This means a `.result` file with the right name (i.e. that specified by the `--in` argument's value) in the right directory (specified by the `--dir` argument's value). If one is found, the corresponding `.sql` file must be there too :

```
Looking for a .result file...
A .result file was found. Looking for a .sql file...
A .sql file was found.
```

If no relevant `.result` file can be found, the scripts look for a `.tar` or `.tar.gz` file of the same name (also in the same directory). This is typically the case when using database query results directly. In that case, the files in the tarball (which should include the `.result` and `.sql` files) are extracted :

```
Looking for a .result file...
No .result file found. Looking for a tar ball...
Tar ball found. Processing it...
```

If not even a tarball can be found, the script exits cleanly and warns the user :

```
Looking for a .result file...
No .result file found. Looking for a tar ball...
No tar ball found. Check your inputs (especially the --dir and --in arguments) !
```

Assuming proper input was found, the scripts then look up a corresponding (i.e. with the same name) `.query` file (see 5.3) :

```
Looking for a .query file...
A .query file was found.
```

If no such file can be found, one is built automatically (via the already mentioned routines `SEX_create_query_file` and `SAX_create_query_file`, which are described in chapter 7) :

```
Looking for a .query file...
No .query file found. Building one...
```

The scripts use these to read in the spatial and redshift extents covered by the data, to use as default values if the `--window` or `--zrange` arguments are missing. In that case, the user is notified :

```
Using query window [7.596,7.74,61.164,69.912] (arcseconds) as mapping window
Using query redshift [1.96970937,1.96970937] as mapping redshift range
```

Also in the `.query` file is the number of objects inside the full results, which is printed to the screen :

```
File contains 1 objects
```

If the beam size, specified by the values of the `--bmin` and `--bmaj` arguments, is not greater than the requested pixel size (specified by `--pixelsize`), then no convolution will be done :

```
Beam size too small with respect to pixel size, I will not be performing convolution
```

Computation of the image size is performed, so the user knows what to expect - and may abort the run if the output map size is too big (see chapter 7):

```
Will build 720 x 720 x 1 image
```

Then the actual map-making proceeds, via the `SEX_build_map` and `SAX_build_map` routines, which are described extensively in chapter 7,

```
Actually making S3 map takes 0.118352890015 seconds
```

Finally, the S^3 data cube is written out to a FITS file, whose name is specified by the value of the `--out` argument, and placed in the same directory as the input :

```
Writing to FITS takes 0.172261953354 seconds
```

If some additional data is to be considered (Cosmic Microwave Background, Global Sky Model,...), it is treated as a post-processing step. For instance :

```
Making CMB signal
Added CMB signal to my_map.fits
Making GSM map(s)
outfile_1420.405751 already exists. No need to rebuild it.
Added GSM signal to my_map.fits
```

For more details on these, see chapter 7

Chapter 7

Lower-level routines

TO BE COMPLETED....

Chapter 8

Parallel extensions

TO BE COMPLETED....

References

[de Lucia et al. (2006)] de Lucia, G., Springel, V., White, S.D.M., Croton, D., Kauffmann, G., 2006, MNRAS, 366, 499

[de Oliveira-Costa et al. (2008)] de Oliveira-Costa, A., Tegmark, M., Gaensler, B.M., Jonas, J., Landecker, T.L., Reich, P., 2008, MNRAS, 388, 247

[Levrier et al. (2009)] Levrier, F., Wilman, R.J., Obreschkow, D., Klöckner, H.-R., Heywood, I., Rawlings, S., 2009, in "Widefield Science and Technology for the SKA" SKADS Conference 2009, edited by S.A. Torchinsky, A. vanArdenne, T. van den Brink-Havinga, A. van Es, A.J. Faulkner

[Obreschkow et al. (2009)] Obreschkow, D., Klöckner, H.-R., Heywood, I., Levrier, F., Rawlings, S., 2009, ApJ, 703, 1890

[Springel et al. (2005)] Springel, V., White, S.D.M., Jenkins, A., Frenk, C.S., Yoshida, N., Gao, L., Navarro, J., Thacker, R., Croton, D., Helly, J., Peacock, J.A., Cole, S., Thomas, P., Couchman, H., Evrard, A., Colberg, J., Pearce, F., 2005, Nature, 435, 629

[Wilman et al. (2008)] Wilman, R.J., Miller, L., Jarvis, M.L., Mauch, T., Levrier, F., Abdalla, F.B., Rawlings, S., Klöckner, H.-R., Obreschkow, D., Olteanu, D., Young, S., 2008, MNRAS, 388, 1335

[Wilman et al. (2010)] Wilman, R.J., Jarvis, M.L., Mauch, T., Rawlings, S., Hickey, S., 2010, MNRAS, 405, 447