1 LSMTool: the LOFAR Local Sky Model Tool[1]

LSMTool is a Python package which allows for the manipulation of sky models in the `makesourcedb` format (understood by BBS and NDPPP). Note that LSMTool is still in beta. Please report bugs to drafferty@hs.uni-hamburg.de. To initialize your environment for LSMTool, users on CEP1 and CEP2 should run the following commands:

```
use LofIm
source ~rafferty/init_lsmtool
```

Note that the Pythonlibs LOFAR package includes an older version of astropy that conflicts with LSMTool and cannot be used in conjunction with it.

1.1 Usage

LSMTool can be run as follows:

```
Usage: lsmtool.py <skymodel> <parset> [<beam MS>]
Options:
  --version   show program's version number and exit
  -h, --help  show this help message and exit
  -q          Quiet
  -v          Verbose
```

The parset specifies the operations to perform and their parameters. These are described in the next sections.

1.2 Operations

These are the operations that LSMTool can perform:

SELECT : Select sources by source or patch properties

REMOVE : Remove sources by source or patch properties

TRANSFER : Transfer a patch scheme from one sky model to another

GROUP : Group sources into patches

UNGROUP : Remove patches

MOVE : Move a source or patch position

MERGE : Merge two or more patches into one

CONCATENATE : Concatenate two sky models

ADD : Add a source

SETPATCHPOSITIONS : Calculate and set patch positions

PLOT : Plot a simple representation of the sky model

[1] This section is maintained by David Rafferty (`drafferty@hs.uni-hamburg.de`).

1.3 Example parset

This is an example parset that filters on the flux, adds a source, and then groups the sources into patches:

```
LSMTool.Steps = [selectbright, addsrc, grp, setpos]

# Select only sources above 1 mJy
LSMTool.Steps.selectbright.Operation = SELECT
LSMTool.Steps.selectbright.FilterExpression = I > 1.0 mJy

# Add a source
LSMTool.Steps.addsrc.Name = new_source
LSMTool.Steps.addsrc.Type = POINT
LSMTool.Steps.addsrc.Ra = 277.4232
LSMTool.Steps.addsrc.Dec = 48.3689
LSMTool.Steps.addsrc.I = 0.69

# Group using tessellation to a target flux of 50 Jy
LSMTool.Steps.grp.Operation = GROUP
LSMTool.Steps.grp.Algorithm = tessellate
LSMTool.Steps.grp.TargetFlux = 50.0 Jy
LSMTool.Steps.grp.Method = mid

# Set the patch positions to their midpoint and write final skymodel
LSMTool.Steps.setpos.Method = mid
LSMTool.Steps.setpos.Outfile = grouped.sky
```

In the first line of this parset the step names are defined. In the next sections, the step parameters for every step are defined. Steps are applied sequentially, in the same order defined in the list of steps. A list of step-specific parameters is given in Table 1.

1.4 Interactive use and scripting

LSMTool can also be used interactively (in IPython, for example) or in Python scripts without the need for a parset. To use LSMTool in a Python script or interpreter, import it as follows:

```
>>> import lsmtool
```

A sky model can then be loaded with, e.g.:

```
>>> LSM = lsmtool.load('skymodel.sky')
```

All of the operations described in Section 1.2 are available as methods of the resulting sky model object (with the same name as the corresponding operation). For example, the following commands with duplicate the steps done in the example parset given in Section 1.3:

```
>>> LSM.select('I > 1.0 mJy')
>>> LSM.add({'Name':'new_source', 'Type':'POINT', 'Ra':277.4232, 'Dec':48.3689, 'I':0.69})
>>> LSM.group(algorithm='tesselate', targetFlux='10.0 Jy')
>>> LSM.setPatchPositions(method='mid')
```

In many cases, the methods accept parameters with the same names as those used in a parset (see the full documentation for details). The sky model can then written to a new file with:

```
>>> LSM.write('grouped.sky')
```

Additionally, sky models can be written out as ds9 region files and kvis annotation files (as well as all the formats supported by the astropy.table package, such at VOTable, HDF5, and FITS):

Var Name	Format	Example	Comment
Operation	string	SELECT	An operation among those defined in Sec. 1.2
OutFile	string	out_sky_model.sky	Name of output file
SELECT and REMOVE			
FilterExpression	string	I > 10.0 Jy	Filter for selection
Aggregate	bool	False	Filter by aggregated patch property
ApplyBeam	bool	True	If true, apparent fluxes will be used
TRANSFER			
PatchFile	string	sky_model_with_patches.sky	File with patches that will be transfered
GROUP			
Algorithm	string	tessellate	One of tessellate, cluster, single, every
TargetFlux	string	10.0 Jy	Target total flux of patches (tessellate only)
NumClusters	int	100	Number of clusters (cluster only)
ApplyBeam	bool	True	If true, apparent fluxes will be used
UNGROUP			
MOVE			
Name	string	src1	Name of source or patch to move.
Position	list of floats	[12.3, 23.4]	RA and Dec in degrees to move to
Shift	list of floats	[0.001, 0.0]	RA and Dec in degrees to shift by
MERGE			
Patches	list of strings	[bin1, bin2, bin3]	Patch names to merge
Name	string	merged_patch	Name of new merged patch
SETPATCHPOSITIONS			
Method	string	mid	Set patch positions to mid, mean, or wmean positions
CONCATENATE			
Skymodel2	string	in_sky_model2.sky	Name of second sky model to concatenate
MatchBy	string	position	Identify duplicates by position or name
Radius	string	30 arcsec	Radius within which matches are identified
Keep	string	all	If two sources match, keep: all, from1, or from2
ADD			
Name	string	src1	Name of source; required
Type	string	POINT	Type; required
Patch	string	new_patch	Patch name; required if sky model has patches
RA	float or string	12:45:30.4	RA; required
Dec	float or string	+76.45.02.48	Dec; required
I	float	0.69	Flux in Jy; required
AnyValidColumnName		value	Any valid column name can be specified
PLOT			

Table 1: Definition of variables in the LSMTool parset.

```
>>> LSM.write('outskymodel.reg', format='ds9')
>>> LSM.write('outskymodel.ann', format='kvis')
>>> LSM.write('outskymodel.fits', format='fits')
>>> LSM.write('outskymodel.hdf5', format='hdf5')
>>> LSM.write('outskymodel.vo', format='votable')
```

In addition to the operations described above, a number of other methods are available:

LSM.copy() : Return a copy of the sky model object

LSM.info() : Print information about the sky model

LSM.more() : Print the sky model to the screen, using more-like controls

LSM.getColNames() : Returns a list of the column names in the sky model

LSM.getColValues() : Returns a numpy array of column values

LSM.getRowIndex() : Returns the row index or indices for a source or patch

LSM.getRowValues() : Returns a table or row for a source or patch

LSM.getPatchPositions() : Returns patch RA and Dec values

LSM.getDefaltValues() : Returns column default values

LSM.getPatchSizes() : Returns an array of patch sizes

LSM.setColValues() : Sets column values

LSM.setRowValues() : Sets row values

LSM.setDefaultValues() : Sets default column values

For details on these methods, please see the full documentation.