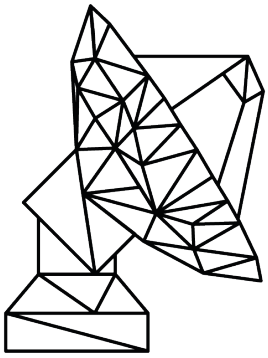
scousepy Documentation

Release 0.0.1.dev

Jonathan D. Henshaw

Feb 11, 2019

CONTENTS



scouse

The `scousepy` package provides a method by which a large amount of complex astronomical spectral line data can be fitted in a systematic way. A detailed description of the method (and the original IDL version of the code) can be found in Henshaw et al. 2016. In the following pages you will find a *brief introduction* to the method as well as a *tutorial*. The source code is available on GitHub and comments and contributions are very welcome.

CHAPTER

ONE

DOCUMENTATION

1.1 Installing scousepy

1.1.1 Requirements

scousepy requires the following packages:

- Python 3.x
- astropy >=3.0.2
- lmfit >=0.8.0
- matplotlib >=2.2.2
- numpy >=1.14.2
- pyspeckit >=0.1.21.dev2682
- spectral_cube >=0.4.4.dev1809

Note that for interactive fitting with pyspeckit you may need to customise your matplotlib configuration. Namely, if you're using scousepy on a Mac you will most likely need to change your backend from 'macosx' to 'Qt5Agg' (or equiv.). You can find some information about how to do this here.

1.1.2 Installation

(Available soon - stick to developer version for now - see below)

To install the latest stable release, you can type:

```
pip install scousepy
```

or you can download the latest tar file from PyPI and install it using:

```
python setup.py install
```

1.1.3 Developer version

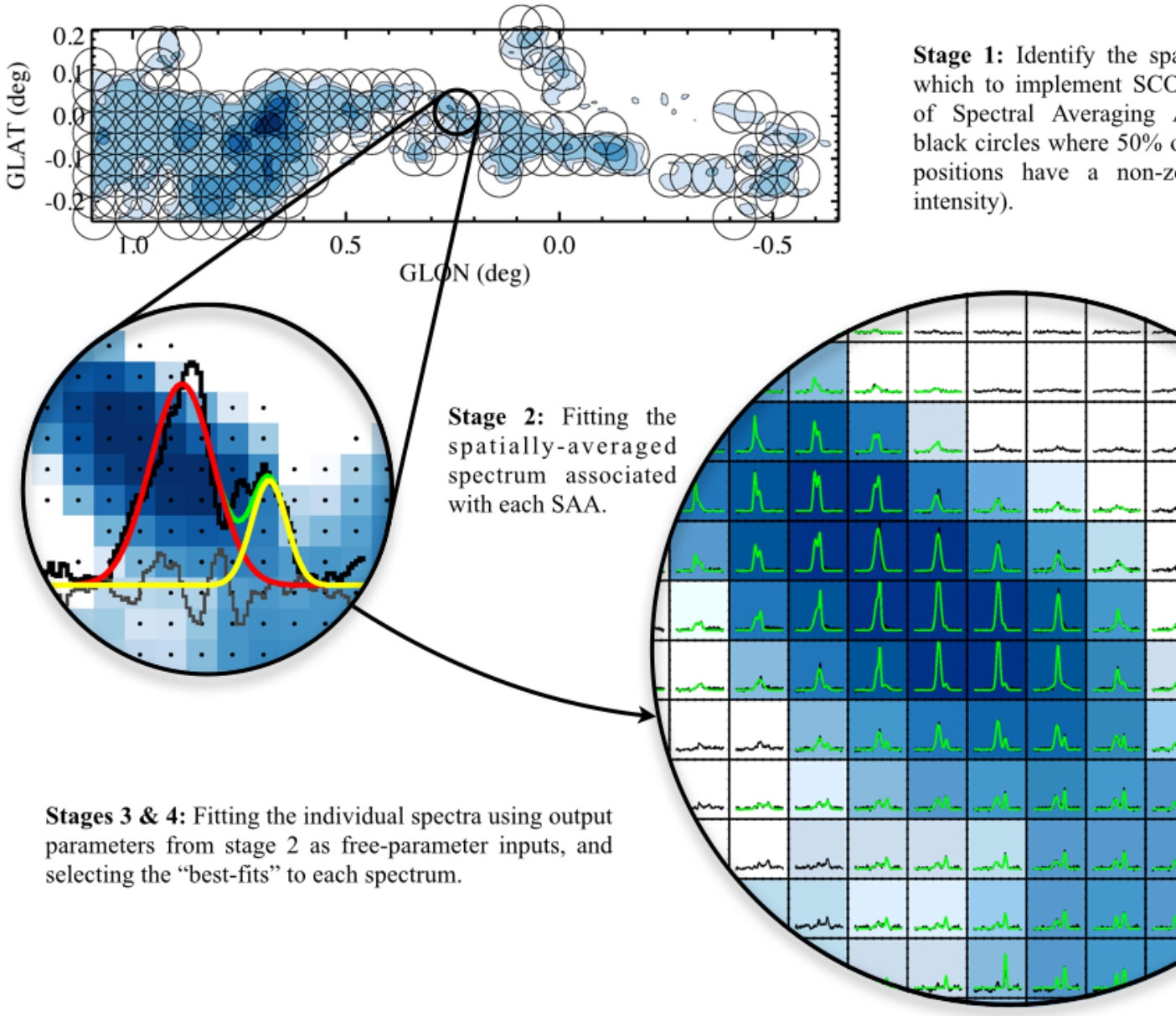
If you want to install the latest developer version of the scousepy, you can do so from the git repository:

```
git clone https://github.com/jdhenshaw/scousepy
cd scousepy
python setup.py install
```

You may need to add the `--user` option to the last line if you do not have root access.

1.2 A brief introduction to scousepy

The method has been updated slightly from the original IDL version of the code. It is now more interactive than before which should hopefully speed things up a bit for the user. The method is broken down into six stages in total. Each stage is summarised below.



Stage 1: Identify the spa
which to implement SCO
of Spectral Averaging
black circles where 50% o
positions have a non-z
intensity).

Stage 2: Fitting the spatially-averaged spectrum associated with each SAA.

Stages 3 & 4: Fitting the individual spectra using output parameters from stage 2 as free-parameter inputs, and selecting the "best-fits" to each spectrum.

1.2.1 Stage 1: defining the coverage

Here `scousepy` identifies the spatial area over which to fit the data. It generates a grid of spectral averaging areas (SAAs). The user is required to provide the width of the spectral averaging area. Extra refinement of spectral averaging areas (i.e. for complex regions) can be controlled using the keyword *refine_grid*.

1.2.2 Stage 2: fitting the spectral averaging areas

User-interactive fitting of the spatially averaged spectra output from stage 1. `scousepy` makes use of the pyspeckit package and is fully interactive.

1.2.3 Stage 3: automated fitting

Non user-interactive fitting of individual spectra contained within all SAAs. The user is required to input several tolerance levels to `scousepy`. Please refer to Henshaw et al. 2016 for more details on each of these.

1.2.4 Stage 4: selecting the best fits

Here `scousepy` selects the best-fits that are output in stage 3.

1.2.5 Optional Stages

Unfortunately there is no one-size-fits-all method to selecting a best-fitting solution when multiple choices are available (stage 4). SCOUSE uses the Akaike Information Criterion, which weights the chi-squared value of a best-fitting solution according to the number of free-parameters.

While AIC does a good job of returning the best-fitting solutions, there are areas where the best-fitting solutions can be improved. As such the following stages are optional but *highly recommended*.

This part of the process has changed significantly from the original code. The user is now presented with several diagnostic plots (see below), selecting different regions will display the corresponding spectra, allowing the user to check the fit quality.

Depending on the data a user may wish to perform a few iterations of Stages 5-6.

1.2.6 Stage 5: checking the best-fitting solutions

Checking the fits. Here the user is required to check the best-fitting solutions to the spectra. This stage is now fully interactive. The user is first presented with several diagnostic plots namely: *rms*, *residstd*, *redchi2*, *ncomps*, *aic*, *chi2*. These can be used to assess the quality of fits throughout the map. Clicking on a particular region will show the spectra associated with that location. The user can then select spectra for closer inspection or refitting as required.

1.2.7 Stage 6: re-analysing the identified spectra

In this stage the user is required to either select an alternative solution or re-fit completely the spectra identified in stage 5.

1.3 Tutorial

1.3.1 Data

This tutorial utilises observations of N2H+ (1-0) towards the Infrared Dark Cloud G035.39-00.33. This data set was first published in Henshaw et al. 2013.. These observations were carried out with the IRAM 30m Telescope. IRAM is supported by INSU/CNRS (France), MPG (Germany) and IGN (Spain).

1.4 License

CHAPTER

TWO

REPORTING ISSUES AND GETTING HELP

Please help to improve this package by reporting issues via GitHub. Alternatively, if you have any questions or if you are having any problems getting set up you can get in touch here.

CHAPTER

THREE

DEVELOPERS

This package was developed by:

- Jonathan Henshaw

Contributors include:

- Adam Ginsburg
- Manuel Reiner

CHAPTER

FOUR

CITING SCOUSEPY

If you make use of this package in a publication, please consider the following acknowledgement...

> Henshaw et al. 2018 (in prep. coming soon)

Please also consider acknowledgements to the required packages in your work.

CHAPTER

FIVE

PAPERS USING SCOUSEPY

- Henshaw et al. 2018, in prep.

CHAPTER

SIX

RECIPE

Recipe for a fine Liverpudlian Scouse pie.