# SurveySim User's Guide
# Version 1.0

Jed McKinney, Noah Kurinsky, Anna Sajina

Last updated: May 31, 2016

**Abstract**

SurveySim is a new MCMC package for constraining the evolution of galaxy luminosity functions. SurveySim is built to be very flexible and hence easily adaptable to a wide range of surveys. However, the current version 1.0 is limited to the infrared 3-1000$\mu$m regime. The core of the code is a series of C++ modules, which perform the MCMC. All changeable aspects, including the luminosity function parameters, the SED template library and the observations themselves are external to this core and are fed to it via fits files. We provide supporting Python scripts, including a graphical user's interface to help setup these files, run SurveySim and examine the output. This User Guide contains the download and installation instructions for SurveySim, as well as a guide to both interactive and command line use.

## Contents

**License**

# 1 Installation

The current version of SurveySim can be obtained via GitHub by following this link: `https://github.com/nkurinsky/SurveySim/raw/master/tags/current.tar.gz`. If you have any trouble with the above download, please email kurinsky@stanford.edu or anna.sajina@tufts.edu for alternative links.

## 1.1 System requirements

SurveySim requires compilers which support the full C++11 standard. We have successfully installed SurveySim on various versions of OSX up to 10.11 as well as Red Hat Linux (versions 5 and 6). We have also installed SurveySim on both the Stanford and Tufts clusters. Feel free to email us if you'd like to setup the code on your cluster and need tips for how to do so.

## 1.2 Compilation

### Step 1

Begin by extracting the SurveySim tar.gz with the following command:

```
tar xzvf current.tar.gz
```

### Step 2

**Only necessary if you do not already have GSL, CCfits, and cfitsio installed on your system**. If you do, skip to Step 3, if you don't, these libraries are contained within their corresponding folder lib_aux as tarballs. Extract each library in the designated order beginning with GSL. If GSL, CCfits and cfitsio are already installed, skip to step 5. Otherwise, begin with the following terminal commands in the lib_aux directory. However, note that if these libraries are already installed but using older compilers they may have to be re-installed to work with SurveySim (e.g. compilers older than gcc5.2 do not use the C++11 standard by default).

```
tar xzvf gsl.tar.gz
```

Now enter the GSL directory and input

```
./configure
make
make check 2>&1
sudo make install

tar xzvf cfitsio.tar.gz
```

Enter the cfitsio directory.

```
./configure --prefix=/usr/local
make
sudo make install

tar xzvf CCfits.tar.gz
```

Enter the CCfits directory.

```
./configure --with-cfitsio=/usr/local
make
sudo make install
```

**Step 3**

Once each library has been successfully installed in /usr/local, run the main compilation procedure in the surveysim-1.0 directory by entering the following:

```
./configure
make
sudo make install
```

Finally, include the following line in a .bashrc file to add the SurveySim directory to your path (if not using the bash shell, translate accordingly).

```
export PATH=\$PATH:.:/usr/local/surveysim-1.0
```

On systems with GSL, cfitsio and CCfits already installed, point to the existing versions with the following syntax:

```
./configure --with-GSL=/path/to/GSL --with-cfitsio=/path/to/cfitsio --with-CCfits=/path/to/CCfits
 make
 sudo make install
```

### 1.2.1 Important Notes for Mac Users

Make sure to install the XCode command line tools before attempting the main installation.

For OSX 10.9: enter the command ”xcode-select –install”

For OSX 10.7: Use clang and clang++ to compile and link all three libraries, as the llvm compiler is very buggy and the gsl checks will fail. To do this, add the following two lines to your ./configure commands.

```
CC = clang
CXX = clang++
```

These are not necessary for the final program, but make check will fail in gsl if ./configure is not properly modified.

### 1.2.2 Local Installation Procedure

A local install may be necessary for users without root access. Begin by extracting each tarball for GSL, cfitsio and CCfits as normal. Execute the following three commands beginning in GSL, then moving to cfitsio and finally in CCfits (cfitsio must be installed before CCfits).

```
./configure --prefix=/path/to/SurveySim
make
make install
```

CCFits will only install if it can find cfitsio. If it does not automatically do so, you may provide CCFits with the path to cfitsio with the following adjustment to ./configure

```
./configure --prefix=/path/to/SurveySim --with-cfitsio=/path/to/cfitsio
```

Before attempting the main compilation in SurveySim/trunk, you will need to set a series of flags with the appropriate commands.

```
export CPPFLAGS='-I/path/to/SurveySim/include'
export LDFLAGS='-L/path/to/SurveySim/lib/'
```

After the flags have been set, enter the SurveySim directory and proceed with the standard compilation procedure.

```
./configure --prefix=/path/to/SurveySim
make
make install
```

Note: make will fail unless your current compiler supports c++11, we recommend gcc 4.8.2 or higher for best (tested) results. We have tested compilation using gcc 4.8, 5.1, 5.2, and 5.3, and found that compilation is successful as long as libraries are compiled with the same gcc version (fairly standard), though increasing the gcc version does generate a larger number of compilation warnings, which can be ignored (they are mostly related to logging).

## 1.3 Python Dependencies

The SurveySim python interface requires the python packages pyfits, numpy, matplotlib, astropy, and seaborn. The SurveySim python interface can be installed into your local python installation in the python distribution like any other package by running (from the surveysim-1.0/python directory):

```
python setup.py install
```

in the SurveySim/python folder, using root privileges if necessary. If you do not have root privleges, you can run

```
python setup.pi install --user
```

and it will be installed in your home directory. By default, this will also check for the packages listed above, and attempt to install them if they are not found or the versions are lower than the minimum SurveySim requirements.

## 1.4 Troubleshooting

If the compilation fails, make sure the CCfits ColumnVectorData header file is correct; it may need to be updated from the CCfits site. ColumnVectorData.h is located in ../surveysim-1.0/lib_aux/CCfits. If any additional problems are encountered that you cannot fix, please start an issue in the github repo by going to `https://github.com/nkurinsky/SurveySim` .

## 2 Setting up a Survey Simulation

In this section we describe the various methods of running SurveySim, including both command line and graphical interfaces.

### 2.1 Calling sequence and Required Files

SurveySim is built such that, once installed, it can be run from anywhere in the system using:

```
    SurveySim modelfile templatefile [obsfile -o outputfile]
```

It is installed by default in /usr/local/bin/surveysim, but if you modified the install path during installation you need to be sure to add that path to your system path. Each of the input files are in the FITS format and contain anything that is likely to change depending on the specific case SurveySim is being applied to without the need to recompile the core. These include the SED library and luminosity function parameters, as well as settings for the fitting routine. These are described in turn below.

#### 2.1.1 Model File

The model file holds the desired values for the luminosity function parameters (see Table 1), including their allowed ranges as well as whether or not the parameter is to be fitted or held fixed. This file also contains parameters specific to the particular survey being simulated (see Table 2 *top*). Lastly, it contains parameters relevant to the MCMC (see Table 2 *bottom*), which we advise users to not modify unless they know what they are doing. All the above parameter settings are contained within the primary header of the model file. The first extension is an ASCII table containing the transmission profiles of the filters to be used in the simulation. The filter library we use is adopted from the Flexible Stellar Population Synthesis (FSPS) code (Conroy, Gunn & White 2009), but with a few long wavelength filters included (AzTEC 1.1mm, and ALMA band3, 5, and 7). We provide an example code for adding a filter to this library (python/add_filter.py).

The SurveySimFilters script allows a user to search the filter library for a particular instrument. For example, to find the DES filters in the library, the command would give the following output:

```
> SurveySimFilters DES
%Filter Location: /usr/local/surveysim/filters/
DES ['DES_g', 'DES_r', 'DES_i', 'DES_z', 'DES_Y']
```

To see all filters, just enter a blank character on the command line (”).

We have provided a python class that deals with handling and writing these fits files. The ModelFile class is defined in `ModelFile.py` under the `python/SurveySim` directory, and is described in more detail later in this section.

#### 2.1.2 SED Templates File

This is a FITS file with a header giving the different types of SEDs (currently ”SFG”, ”AGN” and ”COMP”), the luminosity and redshift bins of the templates included in the library. Each SED template is held in a binary table named SEDTYPE_z#, where z# refers to which redshift bin associated with the given template. Within these extensions, the first column gives the wavelength array in microns. Each subsequent column corresponds to a different luminosity bin and contains the $L_\nu$ array ($W\ Hz^{-1}$) associated with the particular total IR (TIR) luminosity. SurveySim uses linear interpolation to extract a flux at a given redshift, luminosity, and wavelength from the cube constructed from redshift evolved SEDs of each type. We do not currently interpolate between types. We provide the SED templates file described in Kurinsky et al. (2016, submitted) as templates/default_templates.fits. However, any FITS file following the above structure will work with SurveySim.

Table 1. Luminosity Function parameters

| Parameter | Description | Functional Form |
|---|---|---|
| LF_form | "0", "1", or "2" | 0=PL, 1=MS, 2=S |
| Phi0 | Log Normalization | $\Phi^*(z) = \boldsymbol{\Phi^*_o}(1+z)^p$ |
| L0 | Log Luminosity Knee | $L^*(z) = \boldsymbol{L^*_o}(1+z)^q$ |
| Alpha | Primary Slope | $\boldsymbol{\alpha}$ in $\Phi_S$, $\Phi_{MS}$, and $\Phi_{PL}$ |
| Beta | Secondary Slope | $\boldsymbol{\beta}$ in $\Phi_{PL}$ |
| P | Low z Density Evolution | $(z < z_{b,p}) : p = \boldsymbol{P}$ |
| P2 | High z Density Evolution | $(z > z_{b,p}) : \Phi^*(z) = \Phi^*_o(z_{b,p})(1+z-z_{b,p})^{\boldsymbol{P2}}$ |
| Q | Low z Luminosity Evolution | $(z < z_{b,q}) : q = \boldsymbol{Q}$ |
| Q2 | High z Luminosity Evolution | $(z > z_{b,q}) : q = \boldsymbol{Q2}$ |
| zbp | P break redshift | $z_{b,p}$ |
| zbq | Q break redshift | $z_{b,q}$ |
| fa0 | AGN fraction at LogL = 12, z = 0 | $f_{AGN,z} = \boldsymbol{f_{AGN,o}}[log(L_{IR})/12]^{f_P}(1+z)^t$ |
| t1 | Low z AGN fraction Evolution | $(z < z_{b,t}) : t = \boldsymbol{t1}$ |
| t2 | High z AGN fraaction Evolution | $(z > z_{b,t}) : f_{AGN,z} = f_{AGN,z_{b,t}}(1+z-z_{b,t})^{\boldsymbol{t2}}$ |
| zbt | AGN fraction Evolution break redshift | $z_{b,t}$ |
| zbc | SED Color Evolution break redshift | $z_{b,c}$ |
| fcomp | Fraction of AGN Composite Galaxies | |

### 2.1.3 Observations File

The observations file contains the photometry for the survey being modelled in FITS format. Each observation file must be properly formatted according to the examples found in `/surveysim-1.0/examples`. We provide a python script to construct SurveySim-compatible obsfiles called `SurveySimObsFile.py` located in `/surveysim/python`. Specifically, the primary header needs to contain the keyword (FHDU) which specifies the extension number where the data are contained. The extension header needs to associate columns in the table with band flux densities as well as errors. For example: F1COL='F24', EF1COL='e_F24', where 'F24' and 'e_F24' are column headings in the associated binary table. SurveySim uses these headings to assign the correct data to its "band 1 flux", "band 2 flux" etc.

The observation file is not required to run SurveySim. If it is not supplied, SurveySim automatically switches to "prediction mode" where it simply simulates a survey with given flux and/or color limits provided in the model file including provides a model color-magnitude distribution, and distributions in redshift, luminosity and galaxy types achieved by the given selection parameters.

### 2.1.4 Output File

This is a FITS file that holds the SurveySim output. If not supplied, it is by default "output.fits" in the working directory. The output file holds a number of extensions which are discussed in more detail in Section 3.

## 2.2 GUI

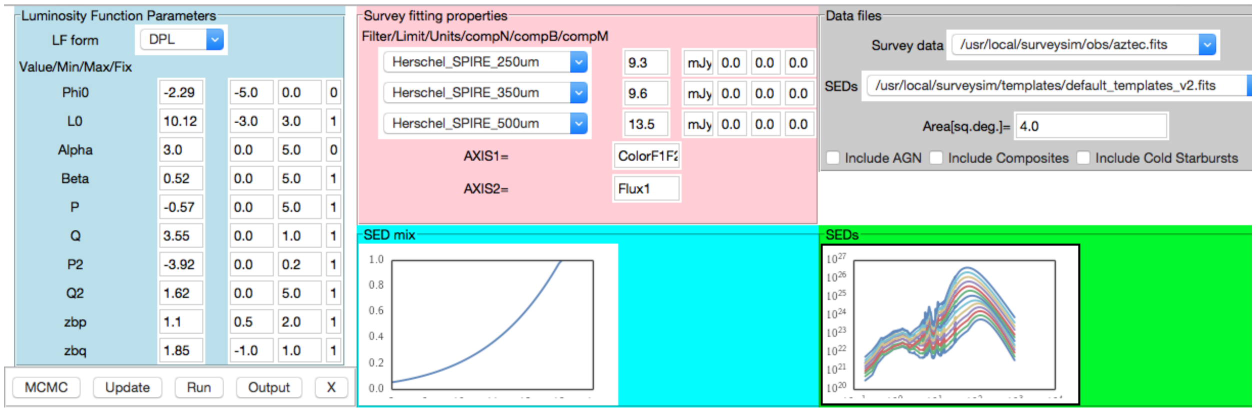
The following command launches the GUI, shown in Figure 1.

```
SurveySimGUI [-i modelfile -o outputfile] [-h]
```

If no model file is supplied, the GUI defaults all parameters to the settings given in ModelFile.py, and generates a new model.fits in the working directory.

Table 2. Parameters relating to the survey being simulated (*top*) as well as MCMC settings (*bottom*). Parameters containing # in this table refer to different bands, and run from 1 to 3.

| Parameter | Default Value | Description |
|---|---|---|
| AREA | 4.0 | Solid Angle of survey [sq.deg.] |
| COLSEL | 'None' | Survey color cut |
| AXIS1 | 'ColorF1F2' | 1st axis to be fit (color or flux) |
| AXIS2 | 'Flux1' | 2nd axis to be fit (color or flux) |
| UNITS# | 'mJy' | Flux units for each band. Can by Jy,mJy,or ABmag |
| LIMIT# | 9.3 | flux/magnitude limit |
| ERROR# | 3.1 | flux/magnitude error (gaussian error about 0) |
| SERROR# | 0 | flux/magnitude skew error (one-sided poissonian error) |
| COMP#N | -1.0 | Power of Completeness Curve $\left(1+\frac{flux-M}{B}\right)^{-n}$ |
| COMP#B | -1.0 | Slope of Completeness Curve $\left(1+\frac{flux-M}{B}\right)^{-n}$ |
| COMP#M | 0.0 | Median of Completeness Curve $\left(1+\frac{flux-M}{B}\right)^{-n}$ |
| ZMIN | 0.01 | Simulation minimum redshift |
| ZMAX | 5.0 | Simulation maximum redshift |
| DZ | 0.05 | Redshift Bin Width |
| RUNS | 10000 | Maximum number of runs (will terminate earlier if convergence is achieved) |
| NCHAIN | 5 | Chain Number |
| TEMP | 0.01 | Starting Anneal Temperature |
| LRATE | 1.0 | Annealing Learning Rate |
| ANN_PCT | 0.25 | Ideal acceptance Percentage |
| ANN_RNG | 0.05 | Range to maintain acceptance within, ANN_PCT $\pm$ ANN_RNG is sought. |
| CONV_CON | 0.9 | Convergence confidence interval (fraction), see Kurinsky 2016 for description |
| CONV_STE | 20 | Steps between convergence checks |
| BURN_STE | 10 | Steps between anneal calls in burn-in |
| BURNVRUN | 10 | Ratio of normal to burn-in steps |
| PRINT | 2 | Level of verbosity (0=silent,3=debug) |



Luminosity Function Parameters

LF form: DPL

| Value/Min/Max/Fix | | | | |
|---|---|---|---|---|
| Phi0 | -2.29 | -5.0 | 0.0 | 0 |
| L0 | 10.12 | -3.0 | 3.0 | 1 |
| Alpha | 3.0 | 0.0 | 5.0 | 0 |
| Beta | 0.52 | 0.0 | 5.0 | 1 |
| P | -0.57 | 0.0 | 5.0 | 1 |
| Q | 3.55 | 0.0 | 1.0 | 1 |
| P2 | -3.92 | 0.0 | 0.2 | 1 |
| Q2 | 1.62 | 0.0 | 5.0 | 1 |
| zbp | 1.1 | 0.5 | 2.0 | 1 |
| zbq | 1.85 | -1.0 | 1.0 | 1 |

Survey fitting properties

| Filter/Limit/Units/compN/compB/compM | | | | | |
|---|---|---|---|---|---|
| Herschel_SPIRE_250um | 9.3 | mJy | 0.0 | 0.0 | 0.0 |
| Herschel_SPIRE_350um | 9.6 | mJy | 0.0 | 0.0 | 0.0 |
| Herschel_SPIRE_500um | 13.5 | mJy | 0.0 | 0.0 | 0.0 |

AXIS1= ColorF1F2

AXIS2= Flux1

Data files

Survey data: /usr/local/surveysim/obs/aztec.fits

SEDs: /usr/local/surveysim/templates/default_templates_v2.fits

Area[sq.deg.]= 4.0

☐ Include AGN ☐ Include Composites ☐ Include Cold Starbursts

Figure 1 SurveySim's GUI.

Figure 1 shows everything that is updatable through the front-page of the GUI. After all desired changes are made, click **Update** on the bottom left of the window to update the working model file to include your changes. Clicking on the **MCMC** button opens a new windows that allows the user to update the MCMC settings as well (not recommended). Clicking the **Run** button, actually runs SurveySim with the given settings. Finally, clicking the **Output** button prints the best-fit parameters and their uncertainties in the terminal as well as generates a figure as the one shown in Figure 2.

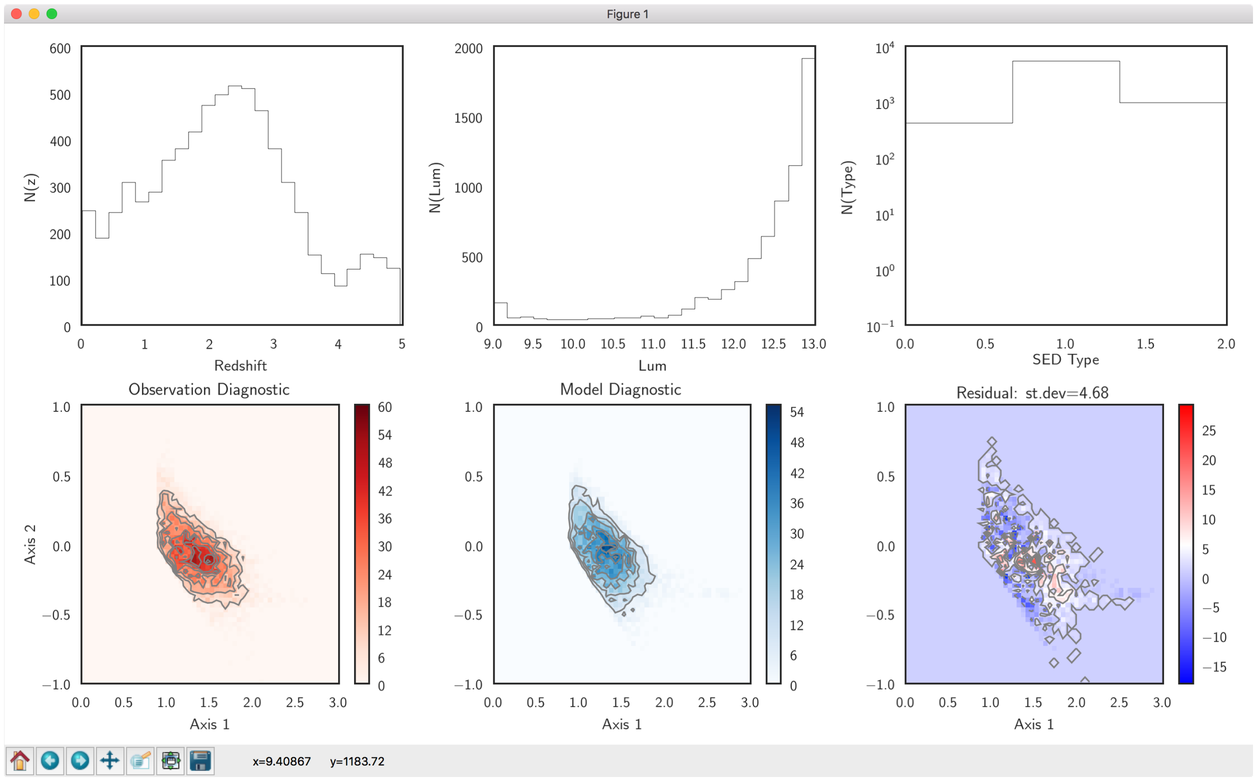


Figure 2 Standard GUI output from its Output button. These figures were generated from a SurveySim model of the HERMES catalogue using the SPIRE band-matched sources as described in Kurinsky et. al. 2016. *Top Row:* Number counts for redshift, luminosity and SED type. 0 = Star Forming, 1 = Composite, 2 = AGN. *Bottom Row:* Color-color or alternatively color-magnitude plots for observational diagnostic, the model sit and residual.

### 2.3 Command-line scripts

It is also possible, and sometimes quicker to modify the input file settings and run SurveySim via command line scripts. In the surveysim-1.0/examples directory, we include the script run for the models shown in Kurinsky et al. 2016, **run_Kurinsky2016_models.py**. As examples, we also include the model files generated by the above script as well as their associated output files. Note that the nature of MCMC runs is such that each run with the exact same initial setup will be slightly different (i.e. have slightly different best-fit parameter values). However, we have tested that the spread between the best-fit solutions of multiple runs is fairly small and indeed smaller than the statistical uncertainty on the parameters from each individual run (see Kurinsky et al. 2016).

## 3 Output

The output file holds the mean and standard deviation of the luminosity function parameters recorded in the MCMC chain in the header. SurveySim's diagnostic diagrams (color-color or color-magnitude) can be found under the Model Diagnostic and Observation Diagnostic extensions; the residual is under the first image extension. The **Best Fit Distributions** extension holds the simulated catalogue. This means for each simulated source - fluxes in three bands, IR luminosity, redshift, and source type. This extension also keeps best-fit model and observed number counts in three bands, but this is a beta version and is not recommended for usage at this point. The **MCMC Chain Record** extension holds the actually MCMC chains for all free parameters and for all chains. Lastly, the **Convergence** extension holds the results for the convergence testing that happens at regular intervals during the MCMC run. The frequency of the convergence tests is governed by the `CONV_STEP` keyword in the model file.

In the surveysim-1.0/examples directory, we include a few scripts for examining the SurveySim output files. Note, these are best run in the directory that hosts both the model and output files for the same run. These scripts are:

```
quick_look.py
plot_nice_images.py
plot_joint_contours.py
```

The first of these, **quick_look.py**, returns the best-fit parameter values as given in the output file header. It also produces a new directory with several *.png* image files which show the MCMC chain output for all free parameters. Specifically the *fit.png* image contains all parameters, *lffit.png* contains only the LF parameters, and *sedfit.png* contains the parameters associated with the galaxy population mix. **Caution!** all the output from `quick_look.py` is meant for a first look examination of the fit results, but are not themselves of science grade. Specifically, these results include the full chains, without removing the burn-in period, and also assume pure Gaussian distributions. These are therefore useful to just see roughly what parameters are coming out as well as examining the chain to see if the parameters have converged, whether there are any significant degeneracies, or signs of multiple solutions.

The second script, **plot_nice_images.py**, takes an output file and shows the model, observations and residual images. Note that at this point, the axis ranges and labels need to be specified within this code, although we are expecting to be able to generalize this in the near future. An example of the output of this code is given in Figure 3.

The last script, **plot_joint_contours.py**, returns the best-fit values after the MCMC chain is appropriately trimmed to remove the burn-in period. By default this code requires the names of two output files whose probability distributions are combined. An example of the output of this code is shown in Figure 4.

Note that while this scripts calls for 2 output files, it also returns the best-fit parameter values with their 68% confidence limits for each individual run as well as jointly. If only a single output file is available, this code can still be run, one simply needs to supply the same output filename twice. In such cases, one should use the "Output #1" best-fit parameters as the "Joint" best-fit parameters are meaningless.

All of the above scripts for setting up SurveySim runs as well as examining their output call upon classes

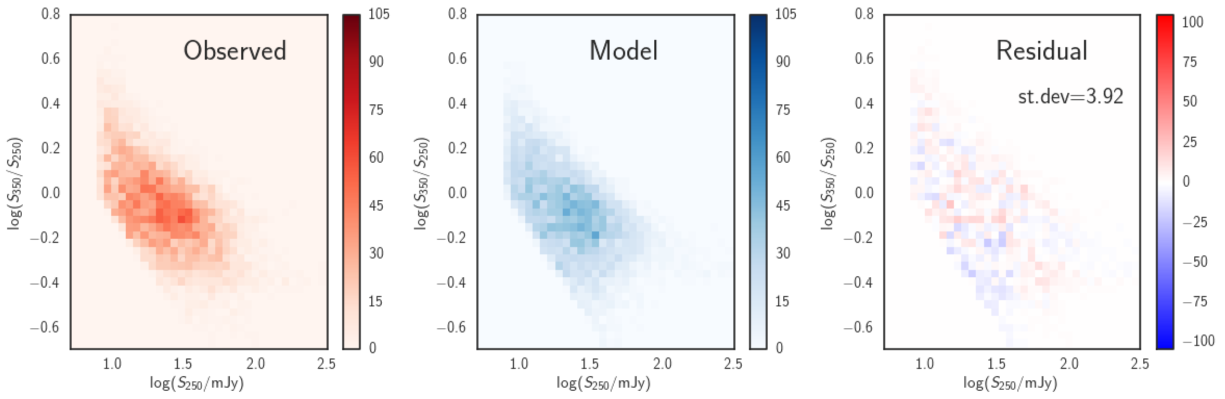


Figure 3 Example of observed, modeled and the residual color-color diagnostic plots based on the HerMES 250 $\mu$m-selected sources in the COSMOS field (Kurinsky et al. 2016).

defined under **python/SurveySim** which hold the necessary functionality. The above described scripts can therefore be used as templates if one wishes to write their own SurveySim scripts.

## 3.1 Development Collaboration and Bug Reports

If any bugs are found during the compilation process or in use, please post bug reports to the main github repository: `www.github.com/nkurinsky/SurveySim`. If interested in collaborating in further code development (including learning what we are already working on in this respect), please email kurinsky@stanford.edu or anna.sajina@tufts.edu .

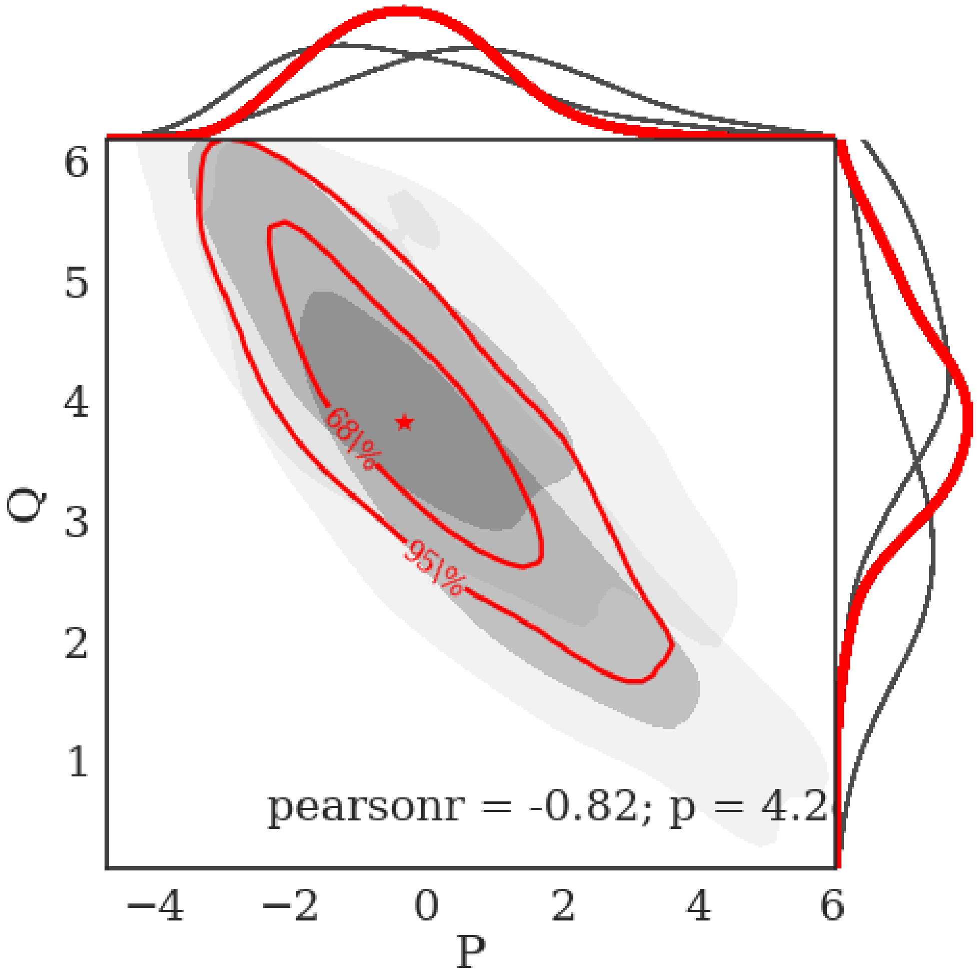


Figure 4 : Example MCMC joint posterior probability distribution (see Kurinsky et al. 2016). The 68% and 95% confidence levels for the individual SurveySim runs are show as respectively darker and lighter shades of grey. The joint probability distribution is marked with thick red contours. The same colors and linewidths apply to the 1d marginalized distributions shown on top and to the left of each panel. In all cases, the inner and outer contours represents respectively 68% and 95% confidence levels. The best joint value is plotted as a star.