Manual

Transient Sky Simulator (tss)

Jennifer Barnes and Bob Jacobs

jlb2331@columbia.edu

b.p.j.jacobs@uva.nl

Table of Contents

1

SECTION

Introduction

The *Transient Sky Simulator* is an open-source Monte Carlo code (written in Python 3.7) that has the ability to simulate any patch of the sky including extragalactic *as well as* Galactic transients. As such TSS allows you to simulate an observing run with several professional telescopes. TSS uses empirical transient number densities and light curves to simulate the transient sky. The library of light curves that we have collected for TSS is also unique in that it contains good multi-color light curves of transients that did not have any light curves publicly available yet.

For more information on TSS than is written in this manual, please have a look at Bob Jacobs' Master's thesis here

SECTION

2 Installation

2.1 Prerequisites

The following Python (3.7) packages are required:

- numpy
- scipy
- matplotlib
- time
- glob
- h5py
- csv
- astropy
- astLib
- itertools
- dustmaps

The following packages may be necessary depending on how you use TSS:

- FFmpeg (necessary if you are animating the output)

- astroplan (recommended for observation plan creation)
- sncosmo (Only necessary if you want to add extragalactic transients or color filters to the preexisting ones)

For the creation of observation plans you will also want to install Jupyter Notebooks.

It is strongly recommended to use the online version of the dustmaps package. For the purpose of TSS this is much faster. In case you don't have an internet connection while running TSS, you can also use the off-line version of dustmaps. Before first time off-line usage one needs to download the dust maps 'bayestar' and 'sfd' for the dustmaps package as described here.

2.2 Installation

To install TSS please clone the Github directory:

```
git clone https://github.com/jL-barnes/TSS.git
```

SECTION 3

Transients and filter systems

3.1 Transients already installed

Below is a table with the transients we have included, the transient light curve file name prefix and the source of the light curves.

Transient	Subtype	File name	Source
Classical nova		NovaDel13	[1]
Dwarf nova	U Gem	SSCyg	[2]
	SU UMa	YZCnc	[2]
	Z Cam	RXAnd	[2]
Supernova	Ia	nugent-sn1a	[3]
	Ib	snana-2007y	[4]
	Ic	snana-2004fe	[4]
	IIP	snana-2007kw	[4]
	IIL	s11-2004hx	[5]
	IIn	[1]	
M-dwarfs	M3	AD_Leo	[6]
	M3.5	EV_Lac	[6]
	M4	YZ_CMi	[6]

Table 1: [1]: Skopal et al. (2019) [2]: AAVSO database [3]: sncosmo/Nugent, Kim Permutter (2002) [4]: sncosmo/Extracted from SNANA's SNDATA_ROOT on 5 August 2014. [5]: Sako et al. (2011) [6]: Private communcations with Adam Kowalski.

3.2 Filter systems already installed

TSS has five filter systems already installed. They are:

- The Bessell filters U,B,V,R and I. We adopt the filter transmission curves as they are in the `sncosmo` package. Their source is Bessell (1990) Table 2
- The SDSS filters u,g,r,i and z. We adopt the filter transmission curves as they are in the `sncosmo` package. Their source is Doi et al. (2010) Table 4
- The LSST filters u,g,r,i,z and y. We adopt the filter transmission curves as they are in the `sncosmo` package. Their source is `https://github.com/lsst/throughputs/tree/7632edaa9e93d06576e34a065ea4622de8cc48d0/baseline`.
- The BlackGEM filters u,g,r,i,z and q. We have privately obtained these preliminary filter transmissions from the BlackGEM team with the permission to use and redistribute them in this open-source software package. Please be aware that they are preliminary and do not include effects from for example the mirror coating.
- The ZTF filter g,r and i. We have privately obtained these filter transmissions from Dr. Daniel Goldstein with the permission to use them and redistribute them in this open-source software package.

[1]We model Supernovae IIn with the same light curves as IIPs and IILs but with lower peak magnitudes. After all, they are photometrically nearly indistinguishable.

SECTION 4

Getting started

A very simple and easy first run can be done with:

```
python3 main.py
```

The arguments are explained in Section 4.2.
If it runs correctly, all packages are installed correctly.

4.1 Output

The code above will create an output file with the default name `output.dat`. This contains all data that a telescope will be able to see plus the types of the transients found. The first row contains a list of all transients that were visible with an integer before them. This integer corresponds to the integers in the `type` column below. So, if the top row shows `0 kilonova` this means that all transients detected whose `type` is `0` are kilonovae.
From the second row onwards a table is printed. The columns are:

- `iD`: The number that is attached to the transient.
- `type`: The transient type encapsulated in an integer (see above).
- `time`: The time after start of observations in seconds.
- `RA`: Position of transient in RA.

- `DEC`: Position of transient in DEC.
- `mag`: Apparent magnitude of the transient at this time in this filter.
- `Qmag`: Quiescent magnitude of the transient in this filter. If the transient does not have a quiescent magnitude it is displayed as 1000.000.
- `band`: The filterband of this observation.

The output file serves as the main result of TSS. The data in this file can subsequently be processed in whatever way one wishes. Some options are displayed in Section 9

4.2 `main.py` Arguments

One can choose a few options when running

```
python main.py [Arguments]
```

they are all optional:

- `[-h] [--help]` Print help function
- `[-f] [--file]` The Obstimes file (see Section 7) default: `Obstimes.dat`
- `[-p] [--params]` Parameter file to use (see Section 5) default: `params.py`
- `[-l] [--offline]` Execute the calculations fully offline. This usually takes a lot longer!
- `[-d] [--nodust]` Exclude dust extinction in the calculation
- `[-c] [--colorsys]` Color system to use. `ABVega` will return UBVRI measurements in Vega and ugriz measurements in AB. Other options are `Vega` (returns everything in Vega measurements) and `AB`. default:`ABVega`
- `[-O] [--output]` Name of the output file. Defaults to the outputfile in `params.py`
- `[-o] [--option]` `Animate` and/or `ColorColor` (see Section 9)

These arguments will override any arguments in params.py

4.3 Restrictions

4.3.1 Filter bands

TSS is capable of handling two different filter systems simultaneously: the Bessell system plus a second system which is by default the SDSS system. The latter system can be swapped for the LSST filter system, the BlackGEM system or the ZTF system. Any new light curve files should be similar to the ones already installed: light curve files for the Bessell system should have `_UBVRI` as suffix; in other systems the files should have `_sdss`, `_lsst`, `_blackgem`, `_ztf`, or the new filter system *in lowercase* as suffix.

4.3.2 Telescope aperture

In TSS the telescope aperture is rectangular. As of version 1.0 it cannot cope with circular apertures. It is also assumed that the aperture is constantly aligned with the RA/DEC grid such that one side of the aperture goes along a line of constant declination (this is generally the case with equatorially mounted telescopes).

4.3.3 Snapshot duration

Measurements of the brightness of transients in TSS are done in a single snapshot: the flux of a transient is not averaged over the duration of the exposure time, but rather taken as the instantaneous flux at the start of the exposure. This should not matter for most transients and most exposure times, but it can become important for longer exposures (e.g. BlackGEM) and short transients (e.g. M-dwarfs).

SECTION 5

Observation parameters

In `params.py` one can define many parameters that correspond to a telescope's performance and some settings for the particular run of TSS.
Here we list the options one can change:

- `n_RA` and `n_DEC`: The number of cells in the RA and DEC directions of a field-of-view in TSS. The higher the number, the more accurate, but slower. For field-of-views away from the Galactic bulge this should generally not be changed.
- `nCells_D` and `nCells_xgal`: The number of cells in the distance direction (Galactic/extragalactic).
- `mag_limit` Limiting magnitude in each filter band given as a dictionary.
- `mag_resolution`: Resolution of the telescope in magnitudes (in most sensitive band).
- `aperture_DEC` and `aperture_RA`: Telescope aperture in DEC and RA direction (in degrees)
- `color_system`: The secondary filter system (next to the Bessell's). Can be `sdss`, `lsst`, `blackgem`, `ztf`, or your own new filter system *in lowercase*.

Next `params.py` contains a list of switches for pre-installed transients. One can include them in a calculation by setting the switch to 1.

One can also include their own transients by filling its name (exactly the same as in the light curve files) into `Extra_transients`

Furthermore `params.py` contains parameters for the kilonova:

- `maxLIGODist`: The maximum distance in centimeters up to which LIGO/Virgo can observe the gravitational waves from this merger event.
- `k_tMerge`: Time of merger relative to observation start in seconds.
- `k_dist`: The distance at which the merger takes place. Can be set to `None` to set the distance randomly within LIGO/Virgo limits.
- `k_mvRed`: A set of data in form [a,b] with a the Ejecta mass and b the velocity of the red kilonova ejecta component.
- `k_mvBlue`: A set of data in form [a,b] with a the Ejecta mass and b the velocity of the blue kilonova ejecta component. To only have a red(blue) component, set `k_mvRed(k_mvBlue)=None`
- `k_HGE`: The host galaxy extinction probability distribution scale. Choose from (`no`, `G%f`, `F%f`) for (no dust extinction, a Gaussian with $\sigma = \%f$, an exponential with $\sigma = \%f$).

The last parameters in `params.py` denote things like the output files and options for color-color plots and an animation in case those options are turned on.

SECTION 6

Transient parameters

Different transients have different parameters like the duration of a light curve or a peak magnitude. The source of information on these parameters for TSS is in the `dataTransients.dat` file. This file contains a table with a row for every type of transient that TSS may handle [2]. The columns are:

- The name of the transient. This should be the same as in `params.py`.
- A parameter on the nature of the transient. Choices are (0,1,2,3,4) corresponding to (non-recurrent Galactic transients, recurrent Galactic transients, M-dwarfs, extragalactic transients that require K-corrections[3], extragalactic transients without K-corrections).
- Four parameters on whether the transient can be found in the Galactic thin disk, in the thick disk, in the bulge or in the halo. Sometimes the literature gives a volumetric density for a transient as a function of a single disk profile with a single scale height. If this is the case, disable the thick disk.
- The prefix of the name of the light curve files that should be used for this type of transient.
- The volumetric transient density in the solar neighborhood in kpc^{-3} for Galactic transients. For extragalactic transients this column contains the number of transients that occur for $1 M_{\odot}$ of formed stars. [4]

[2] The actual decision on *whether* a transient is used is found in the `params.py` file (see Section 5)
[3] These transients should also have K-correction files in the `LightCurveFiles\Kcorrections` folder.
[4] TSS simulates extragalactic volumetric densities with a star forming history and a delay-time distri-

- The scale height of the thin disk.
- The duration of the light curve in the light curve file. For dwarf novae this is somewhat more complicated: here the duration of the normal outburst and the super outburst (excluding data on the quescent state) in the LC files are required. This is used to scale these outbursts. For M-dwarfs the duration of the decay time of the light curve file is used.
- The outburst frequency in days. This is only required for recurrent transients. Here the output of a `scipy` lognormal distribution to the data is required. The data entry format in this column is (`s,loc,scale`) where all three parameters correspond to the output of the `scipy` lognormal distribution. There may be no spaces between the three parameters and the commas.
- The mean peak absolute magnitude in the R-band in the rest frame of the transient. In the case of Galactic transients this can be a simple estimate.
- The standard deviation of the mean peak magnitude above.
- (For M-dwarfs only): the flare energy in the U-band of the flare in the light curve file.
- The power-law parameter α for the Delay-time distribution t^{α}. If there is no delay-time (which is the case for core-collapse supernovae), this parameter should be set to 0.0. The same is true for Galactic transients.
- The host galaxy extinction parameter. One can choose from (`no,G%f,E%f`) where $\%f$ is a float. These correspond to (no host galaxy extinction, host galaxy extinction with a single-sided Gaussian distribution with $\%f$ the standard deviation, host galaxy extinction with an exponential distribution with exponential parameter $\%f$).

bution similarly to https://ui.adsabs.harvard.edu/abs/2012MNRAS.426.3282M/abstract. See Bob Jacobs' thesis (here) for more information.

7

SECTION

Setting observation times, fields and filters

Details of an observation schedule are by default given in `Obstimes.dat`.[5] One can of course use your own differently named file by changing the `Obstimes` parameter in `params.py` or by using the `--file` option when running the program. An 'Obstimes file' should contain four columns with a row for every single telescope exposure. Each row should contain:

- The time of observation in Julian Date
- The filter used for this exposure. It should have the exact same name as in the light curve files.
- The center coordinates of the frame in Right Ascension in J2000.
- The center coordinates of the frame in Declination

If you have many different center frame coordinates that differ only slightly (say an arcminute) it is recommended to average these center coordinates and enter the averages. For every pair of different center coordinates TSS creates a new data cube that can become a real memory hog if you have too many of them.

[5]The data in this file is non-specific to telescopes. It can be used for multiple telescopes. Telescope specific data (like aperture size) can be changed in `params.py`

8

SECTION

Adding data

8.1 Adding transients

It is very easy to add new transients. Here is a short description on how to do this for Galactic and extragalactic transients.[6]

8.1.1 Galactic transients

Let us assume we want to add a transient called 'Zodeion'.

The most important part is the light curve. One needs at least data in the Bessell-system. The light curve data needs a *single* time-array for all the colors. If you have data in multiple filter systems, the time-arrays need to be exactly the same for the Bessell-system as in the other system.
It needs to be packaged in an `hdf5` file that follows the naming convention of the files already in the `LightCurveFiles` folder. So, one should make an hdf5 file for every filter system (for which you have data)[7]. It can be insightful to open a light curve file already installed in the folder `LightCurveFiles`.

[6]Any more complicated transients should either be added analogously to their counterpart (e.g. dwarf novae), or require adding an extra transient class in `transient.py` similar to the classes already present. If you require some help adding more complicated transients the writers of TSS are happy to help you. Please email us.

[7]The filter transmission curves for LSST, SDSS and Bessell that were used for the light curves in TSS can be found in the `sncosmo` Python package. The transmission curves for BlackGEM and ZTF can be found in `Auxiliary\Bandpasses` The BlackGEM transmission curves are preliminary and do not account for effects from e.g. the mirror coating.

The hdf5 file should contain an array with absolute magnitudes measurements for as many color filters in the filter system of the file as possible. For non-recurrent transients R-band photometry is compulsory and for dwarf novae V-band photometry is compulsory. These arrays should have the exact same name as the color filter (e.g. 'u'). An extra array with the times of the measurements (in seconds) is required called `times`. This array should be equal in length to the arrays of the absolute magnitude measurements. This means that all magnitude measurements should have either been done simultaneously (since there is only *one* array with times) or be an interpolation of a light curve. When the light curve files are done they must be named e.g. `zodeion_LC_UBVRI.hdf5` and placed in the `LightCurveFiles` folder. The `UBVRI` suffix should be replaced with the name of the filter system you use.

Next, you should add an extra row in `dataTransients.dat`. Please follow Section 6 for this with the transient name `zodeion` and light curve file name prefix `zodeion_LC`.

All that is left to do is to add the name of the transient (the same as in `dataTransients.dat`) in the `Extra_transients` list in `params.py`.

8.1.2 Extragalactic transients

Adding extragalactic transients goes similarly to adding Galactic transients. The difference is that you will also need K-corrections. Luckily there is a help program. One can open `Auxiliary\Kcor.ipynb` or any of the telescope-specific files with a Jupyter notebook and follow the instructions inside that notebook. Especially the third code box is important. This will create the correct hdf5 light curve and K-correction files.

In case the light curves/type of extragalactic transient is not present in the `sncosmo` python package, you will first need to install your transient in this package. To do this, please see the sncosmo manual.

8.2 Adding a filter system

Even though it is not recommended to add a new filter system (it can be tedious), it is possible. Please be aware that any new filter names should be in lowercase letters.
The `Filterchar.py` file contains a few dictionaries with filter characteristics. One needs to add an extra entry to them. This should be self-explanatory.

In the `observ.py` file in the `set_Colors` function in the `observation` class one should add an extra row under the `if`-statement:

```
elif col.lower() == 'filtersystem': colr = col.lower()
```

with `filtersystem` the name of the filter system that will also be used in the light curve file names.

If you desire to use the `Animation.py` script, you should add a corresponding color to the dictionary called `band_colors` in that python file.

New light curve files (and K-correction files) need to be made for every transient that you want to use. For extragalactic transients this can easily be done by adjusting the `Auxiliary\Kcor.ipynb` file for the new filter system and running the notebook for every transient that you want to use. Light curve files for transients not generated with this script should have a suffix with the filter system's name *in lowercase*.

If you are using kilonovae, you will also need to convolve the Kasen models of kilonova spectrum evolutions (`https://github.com/dnkasen/Kasen_Kilonova_Models_2017/tree/master/systematic_kilonova_model_grid`) with your new color filters. The resulting files go into the `kilonovaLib` folder and need to be named similarly to the files already present in the directory.

If you are using M-dwarfs and the name of your new filters aren't *u,g,r,i,z,q* or *y*, you will need to add a line to `data_MDwarfs.dat` with the quiescent luminosities of M3, M3.5 and M4 dwarfs in erg/s/Å. In `transient.py` you will also need to add your filter in the `self.Lq` dictionary in the M-dwarf class in `transient.py`.

Also, don't forget to change the `color_system` parameter in `params.py`

SECTION 9

Processing

9.1 Color-color plot

With TSS one can produce color-color plots of the results of a survey. One can do this with the `--option ColorColor` argument of `main.py` or by running the `colorcolor.py` code with Python 3.7.
To run the `--option ColorColor` argument one should first choose what three color filters to have in this plot. This is done by changing the `CC_bands` parameter in `params.py`. This parameter is a list of three or four filters. If you fill in `['A', 'B', 'C']`, this will create a color-color plot for $A-B$ against $B-C$. If you enter `['A', 'B', 'C', 'D']`, this will create a plot for $A-B$ against $C-D$. In the same file one can choose the output location under the `CC_outfile` parameter.

By running `main.py` with the `--color` option TSS will first generate an observation survey and simultaneously create a color-color diagram. One can also choose to let `colorcolor.py` run on a previously generated output file. This code takes four optional arguments:

- `[-c] [--colors]`: The colors/filters to create a color-color plot for. If you fill in `A B C`, this will create a plot for $A-B$ against $B-C$, if you enter `A B C D`, this will create a plot for $A-B$ against $C-D$. Please enter them in the order of observation.
- `[-p] [--params]`: Parameter file to use. The default is `params.py`.
- `[-f] [--file]`: The file (that was output by TSS) to make a color-color plot from. The default is the `outfile` in the parameter file.

- `[-o] [--output]`: The output file destination. default: `CC.png`.

These arguments override anything in `params.py`

9.2 Animation

Similarly to a color-color diagram one can also make an animation of the data. The animation script in `Animation.py` generates an animation with a panel for every color filter requested.

One can run `Animation.py` with Python 3.7 with the five optional arguments:

- `[-c] [--colors]`: The colors/filters to animate. These can be multiple. Add spaces between filter names.
- `[-p] [--params]`: Parameter file to use. The default is `params.py`.
- `[-f] [--file]`: The file (that was output by TSS) to animate. The default is the `outfile` in the paramaters file.
- `[-r] [--samefr]`: If this option is given the framerate between two epochs that are far apart does not change. By default it does change to account for the fact that often several observations are made on the same night and then there is a day between nights in which no observations can be made.
- `[-o] [--output]`: The output file destination. This is by default `Animation.mp4`.

One can also run the `--option Animate` argument to `main.py`. This will automatically generate an animation for the data that was generated by TSS through `main.py`. The color filters for which the animation is generated can be found in the `Ani_bands` parameter in `params.py`. In the same file the output file destination is defined in `Ani_outfile`. The `--samefr` option mentioned in the paragraph above can be found under `Ani_samefr` in `params.py`

10

SECTION

Contributors

We would like to dearly thank the people who contributed to this project:

- Dr. Samaya Nissanke
- Dr. Deanne Coppejans
- Dr. Adam Kowalski
- Prof. Dr. Christian Knigge
- Dr. Anna Francesca Pala
- Dr. Jan van Roestel
- Dr. Augustin Skopal
- Dr. Daniel Goldstein
- The BlackGEM-team