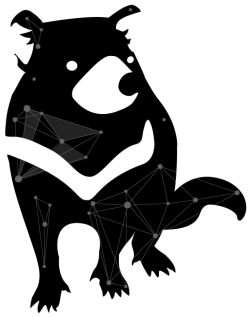

DEVILS

Deep Extragalactic VIsible Legacy Survey

Tool for Analysis and Redshifting (TAZ) Version 0.1

Luke Davies (5/12/2017)

1 Installation

1.1 External Packages

There are three main pieces of external software required to run TAZ. They are i) main coding language R, ii) the 2dfDR data reduction software provided by the the AAO, iii) the AAT targeting/fibre assignment software CONFIGURE.

1.1.1 Installing R

You can download the version of R specific to your operating systems and via your closest mirror here: `https://cran.r-project.org/mirrors.html`. Follow the instructions to install R. You can then check that R is installed correctly, by opening a terminal window and typing:

```
% R
```

If an R session starts in your terminal you are up and working fine.

1.1.2 Installing 2dfDR

The 2dfDR software is used to reduce data from the 2df+AAOmega system at the AAT. Firstly, you can find detailed information regarding the processes in 2dfDR here: `https://www.aao.gov.au/get/document/2dF-AAOmega-obs-manual-69-88.pdf`. In order to install 2dfDR go here: `https://www.aao.gov.au/science/software/2dfdr` and download the appropriate version for your operating system. You will then need to follow the installation instructions.
You will then need to set the location of the installed 2dfDR software to your path by add the following (or similar) to your .cshrc/.bashRC/.profile file. For me this is:

```
setenv PATH $PATH\:/Applications/2dfdr-6.46-MacOsX_ElCapitan/bin
```

For running TAZ we use the command line version of 2dfDR called AAORUN. To check this is running correctly, open a terminal and type:

```
% aaorun help
```

This should display the AAORUN command line options. You can also test the 2dfDR GUI software interface is working, but typing:

```
% drcontrol
```

This should give you a nice image of Saturn and tell you that there are no raw data files in your current directory.

1.1.3 Installing CONFIGURE

The AAT uses a piece of software called CONFIGURE to determine fibre assignments and produce the files required to drive 2df. You can find out about what CONFIGURE does here: `https://www.aao.gov.au/science/software/configure`. On this page follow the online instructions for installing configure.

You will also need to add the path to CONFIGURE:

```
setenv PATH $PATH\:/Applications/configure-8.4-MacOsX_ElCapitan_x86_64
```

and the environmental variable for CONFIGURE's data files:

```
setenv CONFIG_FILES /Applications/configure-8.4-MacOsX_ElCapitan_x86_64/data_files/
```

To your .cshrc, etc. You can then check that CONFIGURE is running correctly by opening a terminal and typing:

```
% configure
```

You should get a pop-up GUI asking which instrument you are using. Don't worry about this, you should not need to do things manually, but TAZ will now be able to call CONFIGURE to allocate fibres for observing.

1.2 Installing R packages

There are a number of R packages that need to be installed. To do this you should just be able to run the INSTALLTAZ.R script provided `https://github.com/ICRAR/DEVILS-TAZ/blob/master/InstallTAZ.R`. First move to the directory where INSTALLTAZ.R is located, open an R session in a terminal, using:

```
% R
```

then run:

```
> source('InstallTAZ.R')
```

This will install all of the required packages required to run TAZ. As a quick test type:

```
> installed.packages()
```

and you should see DEVILSTAZ on this list.

2 Setting Up the DEVILS Directory Structure

The first stage in running TAZ is to set up the the DEVILS directory structure. To do this, run the setUpDir() function. This function will create the directory structure including all of the calibrations, IDX files (parameters for 2dfDR), and observability plots/logs for each night you are observing. To run this structure, you will need to input the run and night range for which you wish to generate the data structure for. First open an R session. Then you will need to load the DEVILSTAZ package (this should be all you need to load whenever you start R):

```
> library('DEVILSTAZ')
```

The setUpDir() function is then run as:

```
> setUpDir(workingDir='.', runs=c('run1_2017_12','run2_2018_01'),
dateStart=c('2017_12_18','2018_01_09'),dateEnd=c('2017_12_26','2018_01_22'))
```

where,

`workingDir`=directory location you wish to build this structure.
`runs`=the run names. They must have this format. If you do not know the run you are on, ask Luke (luke.j.davies@uwa.edu.au).
`dateStart`=First night of each run in year_month_day format
`dateEnd`=Last night of each run in year_month_day format

Try running this code and explore the directory structure. You can generate this structure for any dates you like. You should something like have:

- **data/**
  - **biases/**
    - **run1_2017_12/, ...**
  - **calibrators/**
    - **AutoZTemp/**
    - **filters/**
    - **GuideStars/**
    - **sensfuncs/**
    - **SkyFibres/**
    - **stdstars/**
  - **darks/**

run1_2017_12/, ...
idxFiles/
logs/
observing/
D10_yrPlan2017.png, ...
run1_2017_12/, ...
2017_12_18/, ...
raw/
run1_2017_12/, ...
2017_12_18/, ...
reduced/
run1_2017_12/, ...
2017_12_18/, ...

You will populate the **biases/**, **darks/**, and **raw/** part of this structure with the data taken at the AAT, and TAZ will generate reduced data and analysis products in the other parts of the structure.

3 Adding Observations

When observing you will add raw data files to this directory structure. Firstly, at the start of each run, bias and dark frames will be taken. These need to be added to the **biases/*runName*** and **darks/*runName*** directories. Note that TAZ will aim to generate a master bias and master dark for all files in this directory, so if there are files that you do not wish to be used in making the master darks/biases please keep them in the **biases/*runName*/junk/** or **darks/*runName*/junk/** sub folders.
Next you will need to place the raw target data under in the relevant directory. This should be easily done as the date in in the file name produced by the AAT. Data for each night observed should be copped to the relevant **raw/*runName*/YYYY_MM_DD**. Once again, TAZ will reduce/analyse all data in this directory. So if there are files you do not wish to be included, add then to the **raw/*runName*/YYYY_MM_DD/junk** folder. You do not need to separate ARC, FLAT and TARGET files in this directory. TAZ will identify and match the correct files based on the configuration used in the FITS header.
For now, there is is an example small dataset you can get from here `https://www.dropbox.com/s/yutdugiwpi4jo3x/TAZRawExample.zip?dl=0`. Once you have this copy the biases, darks and raw data files to your **biases/**, **darks/**, and **raw/** folders under the correct run/date.

4 Running TAZ

Here I will just give a high level overview of running the TAZ software. More detailed descriptions of the individual functions can be accessed via the R help function by simply opening an R terminal and typing:

```
> ?*FUNCTION_NAME*
```

In most cases this code will by run end to end with one line. However, here i split it into various sections to explain what the code is doing and to allow the user to run each component separately.

4.1 Running the 2dfDR reduction

First, lets try and run TAZ on our test dataset to just perform the data reduction. To do this, make sure you are in the directory above the **/data** directory you just generated, and run:

```
> TAZ(user='NewUser', workingDir='.', verbose=2, doReduce=T, doExtract=F,
doStack=F, doAutoZ=F, cores=2, doUpdateMaster=F, doTiler=F, zeroPoint=T)
```

This should run TAZ over you current directory, reduce the data using 2dfDR. When the code finished, you should find that some of the **/data/reduced/** sub-folders are populated. Check that this is the case. The most important thing to check is that you have reduced files in the **/data/reduced/run1_2017_12/2017_12_18/** folder. If not, go back and check you followed the previous stages correctly, and if you are still experiencing problems contact Luke.

One additional option is available at this stage which will produce diagnostic plots of all bias and dark frames you are using. This is very useful for quality control and checking observations at the telescope. To do this simply add `doCalibQC=F` to the line running TAZ. This will produce plots in both **/data/darks/run#_YYYY_YY/** and **/data/biases/run#_YYYY_YY/** to aid in QC of the calibration files.

4.2 Running the 2dfDR reduction and 1D extraction

Next we can try and run both a reduction and extract 1D spectra from each of our reduced frames. As TAZ looks for directories that have already been reduced, you will need to delete the products you have previous made in the **/data/reduced/run1_2017_12/2017_12_18/** and **/data/reduced/run1_2017_12/2017_12_19/** directories, before rerunning this code. To do thew reduction and extraction simply set doExtract=T and run:

```
> TAZ(user='NewUser', workingDir='.', verbose=2, doReduce=T, doExtract=T,
doStack=F, doAutoZ=F, cores=2, doUpdateMaster=F, doTiler=F, zeroPoint=T)
```

This should run TAZ over you current directory, reduce the data using 2dfDR AND extract 1D spectra. The most important thing to check is that you have individual reduced spectra in the .Rdata format in the **/data/reduced/allSpec** directory. If not, go back and check you followed the previous stages correctly, and if you are still experiencing problems contact Luke.

As you may not want to re-run the reduction, you can tell TAZ to skip this part. However, you then have to tell it which files to extract 1D spectra from. This is achieved by setting doReduce=F and then adding the variable: toExtractFiles=*a string list of files to extract*. In our example, this would be:

```
> toExtractFiles<-c('data/reduced/run1_2017_12/2017_12_18/
2017_12_18_config_1_reduced.fits', 'data/reduced/run1_2017_12/2017_12_19/
2017_12_19_config_1_reduced.fits')
```

```
> TAZ(user='NewUser', workingDir='.', verbose=2, doReduce=F, doExtract=T,
toExtractFiles=toExtractFiles, doStack=F, doAutoZ=F, cores=2,
doUpdateMaster=F, doTiler=F, zeroPoint=T)
```

This will skip the reduction phase and extract 1D spectra from the reduced files provided.

4.3 Stacking extracted spectra

As DEVILS will observe the same spectra over multiple observations, these will need to be stacked to increase single to noise. TAZ can perform this stacking from previously extracted spectra. As TAZ writes all spectra to the **/data/reduced/allSpec/** directory, the staking procedure simply searches for all instances of a particular ID in this directory and inverse variance weights the spliced spectrum, individual CCD arms and continuum extracted spectra. As we do not wish to extract the spectra again, set `doExtract=F` and `doStack=T`. Once again, in order to run this, you must tell TAZ which IDs to stack. This can either be done by providing a vector list of ID (note our example data is from GAMA and all objects are appended with a 'G", in DEVILS this will be 'D'):

```
> toStackIDs<-c('G006014', 'G006158')

> TAZ(user='NewUser', workingDir='.', verbose=2, doReduce=F, doExtract=F,
doStack=T, toStackIDs=toStackIDs, doAutoZ=F, cores=2, doUpdateMaster=F,
doTiler=F, zeroPoint=T)
```

A second option is to tell TAZ to try and stack all IDs in the current **/allSpec** folder. To do this, simply set the `toStackIDs` value to 'all':

```
> TAZ(user='NewUser', workingDir='.', verbose=2, doReduce=F, doExtract=F,
doStack=T, toStackIDs='all', doAutoZ=F, cores=2, doUpdateMaster=F, doTiler=F,
zeroPoint=T)
```

Run this code now. TAZ should populate the **/data/reduced/stackedSpec/** folder of your directory structure with 1D spectra. You can now quickly look at one of these spectra manually to see this has worked, in R do:

```
> load('data/reduced/stackedSpec/G006158.Rdata')
> magplot(spec$wave, spec$flux, type='l')
```

and over-plot the blue and red arms separately:

```
> lines(spec$waveBlue, spec$fluxBlue, col='blue')
> lines(spec$waveRed, spec$fluxRed, col='red')
```

or plot the continuum extracted stacked spectrum:

```
> magplot(spec$wave, spec$fluxSub, type='l')
```

If you want to see all information available for this spectrum, simply type:

```
> names(spec)
```

and then your chosen parameter, *i.e.*:

```
> spec$EXP
```

If you look at the spec$z parameter, it will not currently have a value. This will be updated below.

4.4 Running AutoZ on Stacked Spectra

Next we will run AutoZ over all of our stacked spectra. This is done by setting the `doAutoZ=T` flag. As we do not wish to stack the spectra again, we will set `doStack=F`. Once again, in order to run this, you must tell TAZ which stacked spectra to run AuotZ over. This can either be done by providing a vector list of file locations.

```
> toAutoZStacks<-c('data/reduced/stackedSpec/G006014.Rdata',
'data/reduced/stackedSpec/G006158.Rdata')

> TAZ(user='NewUser', workingDir='.', verbose=2, doReduce=F, doExtract=F,
doStack=F, doAutoZ=T, toAutoZStacks=toAutoZStacks, cores=2, doUpdateMaster=F,
doTiler=F, zeroPoint=T)
```

Or as above, you can tell TAZ to try and run AutoZ for all IDs in the current **/stackedSpec** folder. To do this, simply set the `toAutoZStacks` value to 'all':

```
> TAZ(user='NewUser', workingDir='.', verbose=2, doReduce=F, doExtract=F,
doStack=F, doAutoZ=T, toAutoZStacks='all', cores=2, doUpdateMaster=F,
doTiler=F, zeroPoint=T)
```

This should run AutoZ over all spectra. Do this now. You should find that the code populates the **/data/reduced/stackedSpec/AutoZplots/** folder with figures. The code is also updating the meta information in all of the individual spectra .Rdata files in **/data/reduced/stackedSpec/**. If you now reload the spectrum above, you will find it has a redshift and a probability that that redshift is correct:

```
> load('data/reduced/stackedSpec/G006158.Rdata')

> spec$z

0.7750067

> spec$z

0.5725674
```

TAZ will also make plots of all of the sources you have run AutoZ over in the **/data/reduced/stackedSpec/AutoZplots/**. The spectrum for G006158 can be seen in Figure 1. These figures are useful for checking quality control during observations and for examining TAZ outputs.

4.5 Updating Master Catalogues with new redshifts and producing new observing catalogues

Once we have now redshifts, we will wish to update our master catalogues to reflect this, and produce new observing catalogues with updated priorities. In TAZ this is done by setting `doUpdateMaster=T`. If set, TAZ will:

• Find the most recent DEVILS Master Catalogue (DMCat) in the **data/catalogues/MASTERcats** directory.

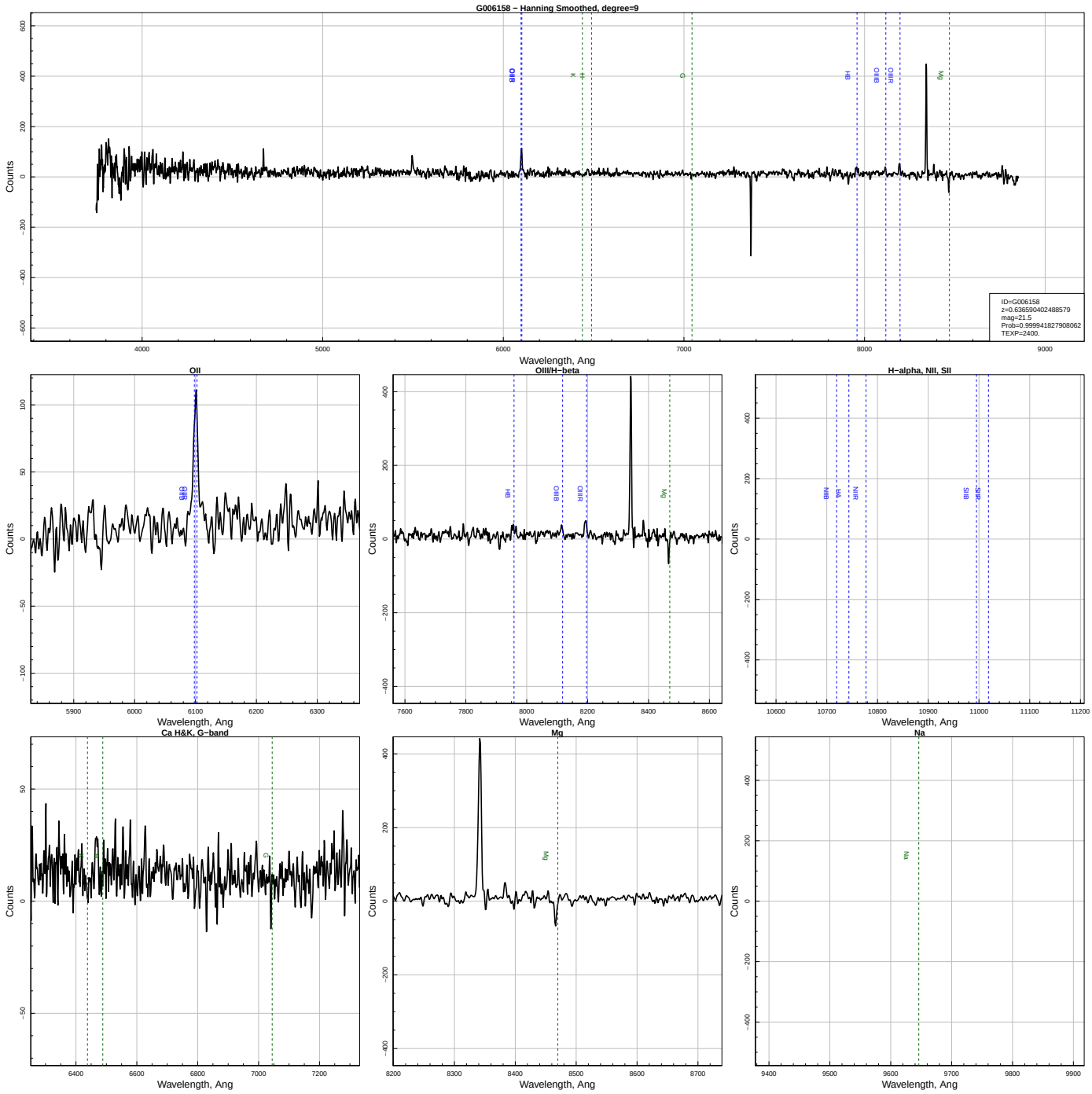

Figure 1: Example output of TAZ after running AutoZ. Top row shows the full spectrum, middle row shows the location of key emission features at the source's measured redshift, and the bottom row shows the same but for absorption features. Line positions at the fitted redshift are shown as dashed vertical lines (emission in blue and absorption in green). Details of the fitted redshift are given in the legend of the top panel. All spectra are Hanning smoothed.

• Read all of the spectra in the **/data/reduced/stackedSpec/** directory and update the DEVILS_z, DEVILS_prob,..... values in the catalogue.

• Save a new DMCat to **/data/catalogues/MASTERcats/** with the current date.

• Identifies the next observing date in your current directory structure (past the date you are running the code) and create a DEVILS observing catalogues (DOCats) directory here **data/observing/run#_YYYY_MM/YYYY_MM_DD/DOCats**. Within this file all of the target, guide, sky, standards catalogues that are required for the Tiling configuration are produced.

To generate to do this simply set `doUpdateMaster=T` run:

```
> TAZ(user='NewUser', workingDir='.', verbose=2, doReduce=F, doExtract=F,
doStack=F, doAutoZ=F, cores=2, doUpdateMaster=T, doTiler=F, zeroPoint=T)
```

Check that TAZ produces the relevant files.

4.6 Producing new fibre configuration files

The final part of TAZ is to generate new fibre configuration files for the next night's observing. This is done by first setting `doTiler=T`. One important extra parameter that needs to be set for this function to work, is the location of the directory containing the AAT's CONFIGURE software, that was installed at the start of this document. To do this simply add the directory location as the `confidir` parameter in TAZ. For modern Macs (and the default) this will be something like: `configdir='/Applications/configure-8.4-MacOsX_ElCapitan_x86_64'`. You will also need to tell TAZ which DO catalogues directory to use in the `DODir` parameter. Finally, we also need to tell TAZ how many fibre allocation files to make in each filed, and whether to start on plate 0 or 1 of 2dF. This is done in the following way:

```
> TAZ(user='NewUser', workingDir='.', verbose=2, doReduce=F, doExtract=F,
doStack=F, doAutoZ=F, doUpdateMaster=F, doTiler=T, DODir='data/observing/run
1_2017_12/2017_12_20/DOCats', zeroPoint=T, cores=4, N_D02A=1,N_D02B=1, N_D03=1,
N_D10=1, D02A_startPlate=0, D02B_startPlate=1, D03_startPlate=0, D10_startPlate
=1)
```

The final parameters of this line tell TAZ to produce 1 configuration in each of D02A, D02B, D03, and D10, and to start on plate 0 for D02A, 1 for D02B, 0 for D03 and 1 for D10. You can try and run this code now. Be aware that this will take some time. TAZ will generate the configuration files on one core per field if `cores`>3.
Within the supplied DOcats directory, TAZ will produce a sub-folder called **Tiling**, within this there will be one sub-folder per field requested, and in this one folder for each configuration, called **TargetFork....**. These files contain the .fld and .sds files produced by CONFIGURE and needed to assign fibres on 2dF. All of these files are date-stamped and copied to a directory called **Tiling/TilesFiles** so that they can be provided to the support astronomer for the next night's observing (see the DEVILS_Observing_Manual.pdf for further details).

4.7 Putting it all together

TAZ functions are rarely run in isolation as above. In practice, at the end of each night's observing the full code will be run end to end. This can be achieved by running something like:

```
    > TAZ(user='NewUser', workingDir=''.'', verbose=2, doCalibQC=F, doReduce
=T, doExtract=T, doStack=T, doAutoZ=T, doUpdateMaster=T, doTiler=T, zeroPoint
=T, cores=4, configdir='/Applications/configure-8.4-MacOsX_ElCapitan_x86_64'
N_D02A=0,N_D02B=0, N_D03=3, N_D10=3, D02A_startPlate=0, D02B_startPlate=0, D03
_startPlate=0, D10_startPlate=1)
```

This will run the full reduction and analysis pipeline from end to to using 4 cores. It will then produce three fibre configurations in D03 starting on plate 0 and three configurations in D10 starting on plate 1. If you want to run the pipeline from end to end on the test dataset. Make sure you delete the reduced data in each of the **/data/reduced/run#_YYYY_MM/ YYYY_MM_DD/** directories, otherwise TAZ will not re-reduce them.

5 TAZ inputs list

Below is a list of TAZ inputs and what they do:

• `user` - A string identifier which will be added to log files to show who ran the code.

• `workingDir` - The directory location where your previously generated **data/...** data structure is located.

• `verbose` - How much information to give you about TAZ, verbose=0,1,2.

• `doReduce` - TRUE/FALSE. Do you want to reduce new data? TAZ will look for new data as files where there is data in the **raw/** directory for a particular night, but nothing in the corresponding **reduced/** directory. For the advanced user, you can force this by running `run2dfDR(toReduce=toReduce)` where `toReduce` is a list of raw directories you want to reduce.

• `doCalibQC` - TRUE/FALSE. Produce bias/dark QC plots.

• `doExtract` - TRUE/FALSE. Do you want to extract 1D spectra from the files you have reduced? If you have set `doReduce=F` and `doExtract=T` you will need to provide a list of reduced files to extract.....

• `toExtractFiles` - A string vector list of reduced files for which you wish to extract 1D spectra. Set if `doReduce=F` and `doExtract=T`. These need to be the full directory path in the DEVILS data structure. *i.e.* data/reduced/run2_2018_01/2018_1_9/2018_1_9_config_1_reduced.fits.

• `doStack` - TRUE/FALSE. Do you want to stack extracted spectra? If you have set `doExtract=F` and `doStack=T` you will need to provide a list of spectra IDs to stack.....

• `toStackIDs` - A string vector list of ID for which you wish to stack. Here you just provide the IDs in a string. *I.e.* D10021811.

• doAutoZ - TRUE/FALSE. Do you want to run Auto-z over the stacked spectra? If you have set doStack=F and doAutoZ=T you will need to provide a list of stacked spectra files.....

• toAutoZStacks - A string vector list of stacked spectra files for which you wish to run Auto-z. These need to be the full directory path in the DEVILS data structure. *i.e.* data/reduced/stackedSpec/D10021811.Rdata.

• doUpdateMaster - TRUE/FALSE. Do you want to update the master catalogue with new redshift. This will produce a new master catalogue with the current date (DMcats). And produce the observing catalogues for the next observation data DOcats.

• doTiler - TRUE/FALSE. Do you want to generate new tile configurations for the next night of observations?

• DODir - If you have set doUpdateMaster=F and doTiler=T you need to tell the code which observation catalogues to use. Set this to a string with the DOCat directory location. Something like: ∼DEVILS/TAZ/data/observing/run1_2017_12/2017_12_18/DOCats/.

• N_D? - Number of new fibre configurations to generate in each field. Use for next night's tiling

• D?_startPlate - Starting plate number for each configuration (either 0 or 1)

• zeroPoint - TRUE/FALSE. Do you want to zero point flux calibrate the spectra? This isn't needed for the redshifting.

• cores - Number of core to uses when running the reduction, AutoZ fitting and fibre configuration.

• configdir - String containing the directory path location of your installation of the CONFIGURE software.

6 TAZ Outputs list

TAZ will have generated a large number of output files in different parts of the directory structure. Below is a list of key data products produced by TAZ such that you can easily find things in the data structure:

6.1 Raw Data Meta Information:

/data/raw/run#_YYYY_MM/YYYY_MM_DD/YYYY_MM_DD_metaData.Rdata
R data file containing meta data from FITS header of all raw files in the current directory. To view this data, move to the directory in which is is contained, open and R session and type:

```
> load('YYYY_MM_DD_metaData.Rdata')

> metaData
```

This will display the filename, ccd, field, RA/DEC centre, exposure time in seconds, the type of data, fibre configuration, date and grating for each file.

6.2 Spliced reduced data for each night/configuration:

/data/reduced/run#_YYYY_MM/YYYY_MM_DD/YYYY_MM_DD_config_?_reduced.fits
FITS file with the fully reduced and spliced data for a particular fibre configuration. These are found within the data structure under each night in the **reduced/** folder.

6.3 Blue arm reduced data for each night/configuration:

/data/reduced/run#_YYYY_MM/YYYY_MM_DD/ccd1/YYYY_MM_DD_config_?_reduced_blue.fits
FITS file with the fully reduced blue data for a particular fibre configuration. These are found within the data structure under each night in the **reduced/** folder (note that the blue and red arms are in **ccd1** and **ccd2** subdirectories).

6.4 Red arm reduced data for each night/configuration:

/data/reduced/run#_YYYY_MM/YYYY_MM_DD/ccd2/YYYY_MM_DD_config_?_reduced_blue.fits
FITS file with the fully reduced red data for a particular fibre configuration. These are found within the data structure under each night in the **reduced/** folder (note that the blue and red arms are in **ccd1** and **ccd2** subdirectories).

6.5 Meta Information for all reduced configurations:

/data/reduced/run#_YYYY_MM/YYYY_MM_DD/YYYY_MM_DD_metaData.Rdata
Meta information for all reduced data frames. To view this data, move to the directory in which is is contained, open and R session and type:

```
> load('YYYY_MM_DD_metaData.Rdata')

> metaData
```

This will display the filename, ccd, field, RA/DEC centre, exposure time in seconds, the type of data, fibre configuration, date and grating for each file.

6.6 Flux calibration meta information for each reduced configuration:

/data/reduced/run#_YYYY_MM/YYYY_MM_DD/ccd2/YYYY_MM_DD_config_?_reduced_zeroPoints.Rdata

R data file with flux calibration information for each reduced frame. To view this data, move to the directory in which is is contained, open and R session and type:

```
> load('run#_YYYY_MM/YYYY_MM_DD_config_?_reduced_zeroPoints.Rdata')

> zeroPoints
```

This will display:

ZP - Magnitude zero point of the frame calculated from standard stars
ZPMAD - Median average deviation of ZP
ZPRMS - RMS of ZP
ZPNUM - Number of standards used to calculate ZP
FLUXSC - The flux scaling to convert counts to ergs/sec/cm^2/Ang/
ZP_red ... - Same values for red and blue arm data individually.

6.7 Individual extracted spectra from corresponding reduced configuration/night:

/data/reduced/run#_YYYY_MM/YYYY_MM_DD/ccd2/YYYY_MM_DD_config_?_reduced_spec/

Directory containing all individually reduced spectra from the corresponding configuration. This directory will also contain 1D plots if requested in TAZ. Each spectrum is stored as an R data structure. To view this data, move to the directory in which is is contained, open and R session and type (for example):

```
> load('2017_12_18_G2271755.Rdata')

> spec
```

This structure contains various data and meta information for this observation, such a flux, wavelength, sky. It also retains the individual blue and red arms of the spectrum prior to splicing. Note that these are not flux calibrated, but contain the flux calibration information to convert pixels to ergs/sec/cm^2/A (fluxSc). To view the spectrum simply type:

```
> magplot(spec$wave, spec$flux, type='l')
```

or to list available meta information type:

```
> names(spec)
```

These can be displayed as, *i.e.*:

```
> spec$RA
```

All spectra extracted in this way are also copied to the **data/reduced/allSpec/** directory.

6.8 Individual spectra meta information for each configuration/night:

/data/reduced/run#_YYYY_MM/YYYY_MM_DD/ccd2/YYYY_MM_DD_config_?_reduced_meta
This is an R data file containing the meta information for each spectrum row in the corresponding reduced file. To view this data, move to the directory in which is is contained, open and R session and type:

```
> load('YYYY_MM_DD_config_?_reduced_meta.Rdata')

> meta
```

This will display:

ID - Source ID from input catalogue
RA - Fibre RA
DEC - Fibre DEC
X - Fibre X
Y - Fibre Y
XERR - Fibre X error
YERR- Fibre Y error
THETA - Fibre angle
TYPE - Type of source (Target, sky, standard, guide)
MAG - Magnitude of source from input cat
FIBRE - Fibre Number
FILE - The file name and location of the extracted spectrum
EXP - The exposure time of the spectrum

6.9 Individual stacked source spectrum files with redshifts:

data/reduced/stackedSpec/*ID*.Rdata
R data structure containing 1D stacked spectra and meta information. This can be accessed as, *i.e.*:

```
> load('data/reduced/stackedSpec/G2271755.Rdata')
```

Available data products can be displayed as:

```
> names(spec)
```

These are:

wave - vector of wavelength positions from spliced spectrum
flux - vector of counts from spliced spectrum
sn - vector of variance from spliced spectrum
sky - vector of sky values from spliced spectrum
ID - ID of spectrum
RA - Right Ascension of spectrum, J2000
DEC - Declination of spectrum, J2000
MAG - Magnitude of source provided to 2dF
xunit - x-axis unit (will be 'ang')
yunit - y-axis unit (will be 'ang' - used internally in TAZ and for plotting in package spec.tools)
z - redshift of spectrum
EXP - Total exposure time, sec
NStack - Number of spectra that went into the stack
waveBlue - vector of wavelength positions from blue ccd spectrum
fluxBlue - vector of counts from blue ccd spectrum
snBlue - vector of variance from blue ccd spectrum
skyBlue - vector of sky values from blue ccd spectrum
waveRed - vector of wavelength positions from Red ccd spectrum
fluxRed - vector of counts from Red ccd spectrum
snRed - vector of variance from Red ccd spectrum

skyRed - vector of sky values from Red ccd spectrum
fluxSub - continuum subtracted spliced spectrum
fluxSubBlue - continuum subtracted blue ccd spectrum
fluxSubRed - continuum subtracted red ccd spectrum
fluxSc - Scaling value to change counts (flux value) to ergs/sec/cm^2/ang
fluxScBlue - Scaling value to change counts (flux value) to ergs/sec/cm^2/ang
fluxScRed - Scaling value to change counts (flux value) to ergs/sec/cm^2/ang
file - the file and location of the stacked spectrum
prob - the redshift probability from AutoZ
cc - the peak cross-correlation value from AutoZ
z2 - the second best redshift from AutoZ
cc2 - the second peak cross-correlation value from AutoZ
temp - the template number of the best fit redshift from AutoZ

6.10 Final Spectrum/Redshifting diagnostic plots:

data/reduced/stackedSpec/AutoZplots/*ID*.pdf
These plots display the final stacked spectrum and AutoZ fits for each source. See example in Figure 1.

6.11 DEVILS Master Catalogues (DMCat):

data/catalogues/MASTERCats/DMCatYYYY-MM-DD.rda
DEVILS master observing catalogues which contain all data for sources in the DEVILS region including, source positions, previous redshift information, DEVILS redshifts, star-galaxy classes, mask flags and priorities. These data products will be described in detail in a DMU description file.

6.12 DEVILS Observing Catalogues (DOCat):

data/observing/run#_YYYY_MM/YYYY_MM_DD/DOCats/*
DEVILS observing catalogues used to define the fibre configurations on a given night. These are used as inputs to the tiling software. There are a number of files produced in this directory which are:

SurveyInfo.txt - A survey description file which explains the field positions and extent.
DGuideCat* - A guide star catalogue
DSkyCat* - A sky fibre position catalogue
DStdCat* - A standard star catalogue
DObjCat* - The most up-to-date target catalogue which is generated from the most recent DMCat.

6.13 Tiling Files:

data/observing/run#_YYYY_MM/YYYY_MM_DD/DOCats/Tiling/TilesFiles/*
The tiling files produced by CONFIGURE. These are passed to the support astronomer to run the telescope each night.