Manual for CESS (version 0.8.5, 2024/12/30)

1. Download

There are two ways to download the latest version of CESS to the local environment.

1.1 'git' command

Using the 'git' command to clone the whole package.

```
1 git clone https://github.com/RainW7/CESS.git
```

1.2 Release

The CESS GitHub repository is at https://github.com/RainW7/CESS. Click the 'Release' at the right of the website and download the latest CESS. Unpacking the package to your path, like '~/CESS_/'.

2. Setup

Before running the executable script of CESS, there are three steps to set the CESS input parameters.

2.1 Create the morphological parameters libraries

Using the 'create_lib.py' file in '~/CESS_/basic_file/' to create the two morphological parameters libraries -- 'widthlib.pkl' and 'heightlib.pkl'. A detailed description of the morphological parameters library can be found in Section 3.3 in Wen et al. 2024.

The default libraries contain 20 n in the range of [0.5, 5], 20 R_e in the range of [0.3, 7.4], 10 PA in the range of [0◦, 90◦] with an interval of 10◦, and 10 b/a in the range of [0, 1] with an interval of 0.1.

Users can costume their morphological parameters libraries by updating the parameter series in 'create_lib.py'.

To create the morphological parameters libraries, using command:

```
python ~/CESS_<version>/basic_file/create_lib.py
```

```
nseries = np.array([0.5, 0.6, 0.7, 0.8, 0.9, 1. ,1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.8, 2. ,
                    2.5, 3., 3.5, 4., 4.5, 5.]) # 20
reseries = np.round(np.array([0.3, 0.5, 0.7, 0.9, 1, 1.2, 1.4, 1.6, 1.8, 2, 2.5, 3, 3.5,
                              4.5, 5, 5.5, 6, 6.5, 7, 7.4])/0.074) #20
paseries = np.array([0, 10, 20, 30, 40, 50, 60, 70, 80, 90]) # 10
baseries = np.array([0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1]) # 10
```

2.2 Update the 'emulator_parameters.json'

The beginning lines of the 'emulator_parameters.json' file are working directories of paths, and users should update the paths for their environment.

```
    "input_path": "/Users/rain/emulator/CESS_input/",
    "output_path": "/Users/rain/emulator/CESS_output/",

    "bkg_spec_path": "/Users/rain/emulator/CESS_v0.8.5/basic_file/bkg_spec.fits",
    "gems_path": "/Users/rain/emulator/CESS_v0.8.5/basic_file/gems_cat.fits",

    "widthlib_pkl": "/Users/rain/emulator/CESS_v0.8.5/basic_file/widthlib_20x20x10x10.pkl",
    "heightlib_pkl": "/Users/rain/emulator/CESS_v0.8.5/basic_file/heightlib_20x20x10x10.pkl",

    "gutp_path": "/Users/rain/emulator/CESS_v0.8.5/csst_sls_tp/GU.Throughput.1st.fits",
    "gvtp_path": "/Users/rain/emulator/CESS_v0.8.5/csst_sls_tp/GV.Throughput.1st.fits",
    "gitp_path": "/Users/rain/emulator/CESS_v0.8.5/csst_sls_tp/GI.Throughput.1st.fits",

    "nuvtp_path": "/Users/rain/emulator/CESS_v0.8.5/csst_phot_tp/csst_phot.nuv.fits",
    "utp_path": "/Users/rain/emulator/CESS_v0.8.5/csst_phot_tp/csst_phot.u.fits",
    "gtp_path": "/Users/rain/emulator/CESS_v0.8.5/csst_phot_tp/csst_phot.g.fits",
    "rtp_path": "/Users/rain/emulator/CESS_v0.8.5/csst_phot_tp/csst_phot.r.fits",
    "itp_path": "/Users/rain/emulator/CESS_v0.8.5/csst_phot_tp/csst_phot.i.fits",
    "ztp_path": "/Users/rain/emulator/CESS_v0.8.5/csst_phot_tp/csst_phot.z.fits",
    "ytp_path": "/Users/rain/emulator/CESS_v0.8.5/csst_phot_tp/csst_phot.y.fits",
```

'input_path' and 'output_path' are the paths of the input model spectra library (see appendix for details of input_model_spectra.hdf5 file) and the output CESS simulated spectra library.

'bkg_spec_path' and 'gems_path' are the paths of the sky background file and GEMS morphological catalog file (see appendix for details).

'widthlib_pkl' and 'heightlib_pkl' are the paths of the morphological libraries (see Section 2.1).

'g*tp_path' are the paths of filter throughputs of CSST slitless spectroscopic bands.

'*tp_path' are the paths of filter throughputs of CSST photometric bands.

```
    "radi": 100,
    "expt": 150,
    "expnum": 4,
    "arcsecperpix": 0.074,
    "dc_value": 0.02,

    "gu_Res": 225,
    "gu_wave_mid": 3375,
    "gu_dlambda": 15,
    "gu_ree80": 0.17,

    "gv_Res": 250,
    "gv_wave_mid": 5250,
    "gv_dlambda": 21,
    "gv_ree80": 0.19,

    "gi_Res": 250,
    "gi_wave_mid": 8100,
    "gi_dlambda": 33,
    "gi_ree80": 0.19,

    "gu_start_wl": 2200,
    "gu_end_wl": 4400,
    "gu_min_wl": 2450,
    "gu_max_wl": 4200,

    "gv_start_wl": 3600,
    "gv_end_wl": 6800,
    "gv_min_wl": 3800,
    "gv_max_wl": 6600,

    "gi_start_wl": 5900,
    "gi_end_wl": 12500,
    "gi_min_wl": 6100,
    "gi_max_wl": 10500,
```

In the middle of the 'emulator_parameters.json' file are the CSST slitless spectroscopy equipment parameters to date. Users can update the resolution, central wavelength, delta lambda, and R_EE80 of PSF (in arcsec) of three slitless bands. The current values are based on the latest CSST instrumental parameters. The 'g*_start/end/min/max_wl' means the starting wavelength of convolving, the ending wavelength of convolving, the lower wavelength cut of CSST slitless band, and the upper wavelength cut of the CSST slitless band of three bands, respectively.

```
    "INITIAL INDEX OF EL DETECTION IN GU BAND": " ",
    "gu_init_idx": 20,
    "END INDEX OF EL DETECTION IN GU BAND": " ",
    "gu_end_idx": 318,

    "INITIAL INDEX OF EL DETECTION IN GV BAND": " ",
    "gv_init_idx": 20,
    "END INDEX OF EL DETECTION IN GV BAND": " ",
    "gv_end_idx": 373,

    "INITIAL INDEX OF EL DETECTION IN GI BAND": " ",
    "gi_init_idx": 20,
    "END INDEX OF EL DETECTION IN GI BAND": " ",
    "gi_end_idx": 496,

    "EL DETECTION THRESHOLD IN NOISY SPECTRA": " ",
    "noisy_el_detect_th": 1.5,

    "EL DETECTION THRESHOLD IN INTRINSIC SPECTRA": " ",
    "intri_el_detect_th": 1.2,

    "EL CENTER WAVELENGTH DIFFERENCE BETWEEN INTRINSIC AND NOISY SPECTRA": " ",
    "el_detect_success_th": 50,

    "ASSUMED EL WIDTH IN GV BAND (IN INDEX)": " ",
    "gv_elwidth": 40,

    "ASSUMED EL WIDTH IN GI BAND (IN INDEX)": " ",
    "gi_elwidth": 30
```

The last lines describe the parameters of the emission line detection module. Currently, only the GV and GI band detects emission line information. Due to the low transmission at both ends of the filter, CESS will skip a small part at the ends and only detect the middle part of the simulated slitless spectrum. The parameters (e.g., detection threshold) used for emission line detection are also listed here.

2.3 Fill the input file variable

The input filenames should be added into the 'run.py' file at the ending lines (e.g., starting from line 759 in run.py file of CESS_0.8.5). Currently, we offer a simple way to collect and add the filenames with the collecting script 'glob_seedcat.py'. Users can display all the input filenames at the terminal and copy them to the 'run.py'.

```
python glob_seedcat.py <input_file_path>
```

The detailed output example in terminal is here:

```
(grizli) ~/emulator/CESS_v0.8.5  <master*> $ python glob_seedcat.py /Users/rain/emulator/CESS_input
'seedcat2_0420_120_DECaLS_0csp_sfh200_bc2003_hr_stelib_chab_neb_300r_i0100_2dal8_10.hdf5',
'seedcat2_0420_971_MzLS_0csp_sfh200_bc2003_hr_stelib_chab_neb_300r_i0100_2dal8_10.hdf5',
'seedcat2_0420_1064_MzLS_0csp_sfh200_bc2003_hr_stelib_chab_neb_300r_i0100_2dal8_10.hdf5',
'seedcat2_0420_1199_MzLS_0csp_sfh200_bc2003_hr_stelib_chab_neb_300r_i0100_2dal8_10.hdf5',
'seedcat2_0420_1204_MzLS_0csp_sfh200_bc2003_hr_stelib_chab_neb_300r_i0100_2dal8_10.hdf5',
'seedcat2_0420_727_MzLS_0csp_sfh200_bc2003_hr_stelib_chab_neb_300r_i0100_2dal8_10.hdf5',
'seedcat2_0420_1058_MzLS_0csp_sfh200_bc2003_hr_stelib_chab_neb_300r_i0100_2dal8_10.hdf5',
(grizli) ~/emulator/CESS_v0.8.5  <master*> $
```

Copy and paste the output filenames into the 'hdf5filenames' variable:

```
759   hdf5filenames = [
760   'seedcat2_0420_120_DECaLS_0csp_sfh200_bc2003_hr_stelib_chab_neb_300r_i0100_2dal8_10.hdf5',
761   'seedcat2_0420_971_MzLS_0csp_sfh200_bc2003_hr_stelib_chab_neb_300r_i0100_2dal8_10.hdf5',
762   'seedcat2_0420_1064_MzLS_0csp_sfh200_bc2003_hr_stelib_chab_neb_300r_i0100_2dal8_10.hdf5',
763   'seedcat2_0420_1199_MzLS_0csp_sfh200_bc2003_hr_stelib_chab_neb_300r_i0100_2dal8_10.hdf5',
764   'seedcat2_0420_1204_MzLS_0csp_sfh200_bc2003_hr_stelib_chab_neb_300r_i0100_2dal8_10.hdf5',
765   'seedcat2_0420_727_MzLS_0csp_sfh200_bc2003_hr_stelib_chab_neb_300r_i0100_2dal8_10.hdf5',
766   'seedcat2_0420_1058_MzLS_0csp_sfh200_bc2003_hr_stelib_chab_neb_300r_i0100_2dal8_10.hdf5',
767   ]
```

3. Run

3.1 The running command

After the setup, CESS can be run with the command

```
python run.py <NUM1> <NUM2> morph=yes wave_cal=yes photo=yes
el_detect=yes
```

The and here stands for the staring index and ending index of the 'hdf5filenames' variable. The expected computing source of one input file is one CPU and about 10GB memory, thus choosing a suitable and that fits the user's environment to avoid 'ArrayMemoryError' during the run.

Addtionally, the running script sets four calculation switches, which are controlled by the keywords in the running command. The 'morph' means adding morphological effects, 'wave_cal' means adding wavelength calibration error effects, 'photo' means using the photometric calculation module and 'el_detect' means using the emission line detection module. These calculation modules are off in default, users should set them with 'true' or 'yes' to switch on.

```
35  begin = int(sys.argv[1]) hdf5filename变量起始数字索引
36  end = int(sys.argv[2])   hdf5filename变量终止数字索引
37  apply_morphology_effect = False
38  apply_wavelength_calibration_error = False
39  apply_photometric = False                  四个计算模块 默认关闭
40  apply_el_detect = False
41
42  for arg in sys.argv[1:]:  四个计算模块的启用方式
43      if arg.lower() == 'morph=true' or arg.lower() == 'morph=yes':
44          apply_morphology_effect = True
45      elif arg.lower() == 'wave_cal=true' or arg.lower() == 'wave_cal=yes':
46          apply_wavelength_calibration_error = True
47      elif arg.lower() == 'photo=true' or arg.lower() == 'photo=yes':
48          apply_photometric = True
49      elif arg.lower() == 'el_detect=true' or arg.lower() == 'el_detect=yes':
50          apply_el_detect = True
51
```

3.2 The output file

The output file of CESS is named as 'CSST_grism_<input_file_name>_<length_of_the_file>.hdf5 ', e.g.,

```
CSST_grism_seedcat2_0420_727_MzLS_0csp_sfh200_bc2003_hr_stelib_chab_neb_300r_i0100_2dal8_10_23360.hdf5
```

4. Check output

CESS offers two ways 'demo.py' and 'demo.ipynb' to show the results of the output file, including showing simulated spectrum figures and hdf5 file structures.

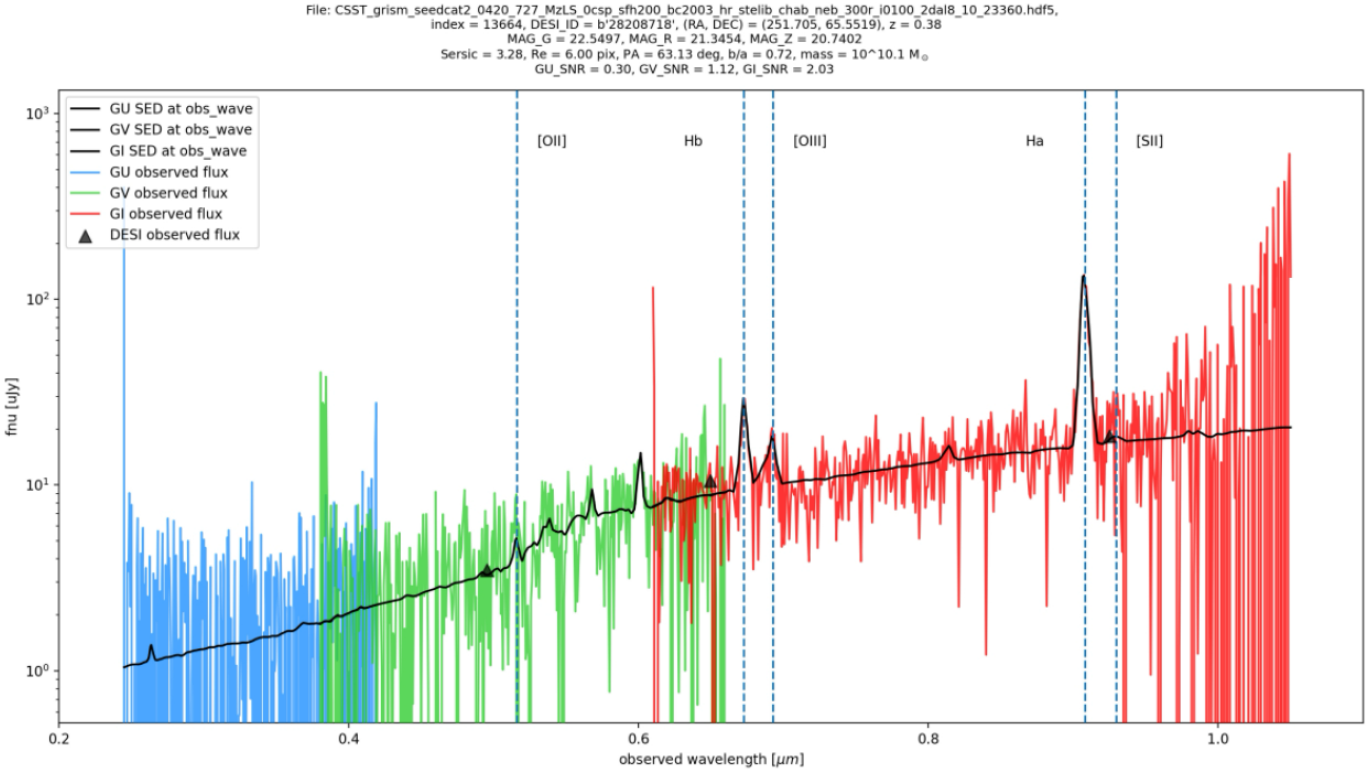
4.1 To display the spectrum

The 'demo.py' file is used to display the simulated CSST grism spectrum with detailed parameters and original model spectrum. The command is:

```
python demo.py <CESS_output_file_path> <NUM or random> <emissionline or none>
```

Where the <CESS_output_file_path> is the path of CESS output file, means a certain index within the length of CESS output file at the end (e.g., 23360 for 727_MzLS) or a random index using , and means showing intrinsic emission line information in the figure or not.

```
(grizli) ~/emulator/CESS_v0.8.5  $ python demo.py CSST_grism_seedcat2_0420_727_MzLS_0csp_sfh200_bc2003_hr_stelib_chab_neb_300r_i0100_2dal8_10_23360.hdf5 random emissionline
The spectrum (ID = 13664) is shown!
```



4.2 To show the file structure

The Jupyter file 'demo.ipynb' is used to show the detailed CESS input, output file structure and simulated spectra with simulated photometric data.

The input file (e.g., <seedcat2_0420_727_MzLS_0csp_sfh200_bc2003_hr_stelib_chab_neb_300r_i0100_2dal8_10.hdf5>) structure is like:

```
[3]  mzls_origin = h5py.File('/Users/rain/emulator/CESS_input/seedcat2_0420_727_MzLS_0csp_sfh200_bc2003_hr_stelib_chab_neb_300r_i0100_2dal8_10.hdf5')
     ✓ 0.0s

     for name in mzls_origin:
         print(name)
         # if group, show the group name and datasets
         if isinstance(mzls_origin[name], h5py.Group):
             for subname in mzls_origin[name]:
                 print(f"  {subname}")
[4]  ✓ 0.0s

...  ID
     best_fit
       spec_csp_sfh200_bc2003_hr_stelib_chab_neb_300r_i0100_2dal8[0,1]
       wavelength_rest
       z
     filters
       filter_0
       filter_1
       filter_10
       filter_11
       filter_12
       filter_13
       filter_14
       filter_2
       filter_3
       filter_4
       filter_5
       filter_6
       filter_7
       filter_8
       filter_9
     parameters
     parameters_name
```

While the output file (e.g., <CSST_grism_23514_seedcat2_0420_1204_MzLS_0csp_sfh200_bc2003_hr_stelib_chab_neb_300r_i0100_2dal8_10.hdf5>) structure is like:

```
mzls = h5py.File('/Users/rain/emulator/CESS_output/CSST_grism_23514_seedcat2_0420_1204_MzLS_0csp_sfh200_bc2003_hr_stelib_chab_neb_300r_i0100_2dal8_10.hdf5')
```

[5] ✓ 0.0s

```
for name in mzls:
    print(name)
    # if group, show the group name and datasets
    if isinstance(mzls[name], h5py.Group):
        for subname in mzls[name]:
            print(f"  {subname}")
```

[6] ✓ 0.0s

```
GI
  detect_el_elnumber
  detect_el_id
  detect_el_idx
  detect_el_snr
  detect_el_wave
  ferr
  flux_elec
  flux_ujy
  flux_ujy_with_noise
  intri_el_elnumber
  intri_el_id
  intri_el_idx
  intri_el_wave
  snr
  snr_mask
  spec_mask
  wave
GU
  ferr
  flux_elec
  flux_ujy
  flux_ujy_with_noise
  snr
  snr_mask
...
data_mask
parameters_desi
parameters_grism
parameters_phot
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

A demo for simulated CSST slitless spectrum with photometric data (conversion functions for AB mag and flux in μJy is offered) is like:

```
plt.errorbar(2877.52,mzls['parameters_phot'][0,idx], mzls['parameters_phot'][7,idx],c='k',marker='x')
plt.errorbar(3683.53,mzls['parameters_phot'][1,idx], mzls['parameters_phot'][8,idx],c='k',marker='x')
plt.errorbar(4729.27,mzls['parameters_phot'][2,idx], mzls['parameters_phot'][9,idx],c='k',marker='x')
plt.errorbar(6121.62,mzls['parameters_phot'][3,idx], mzls['parameters_phot'][10,idx],c='k',marker='x')
plt.errorbar(7578.78,mzls['parameters_phot'][4,idx], mzls['parameters_phot'][11,idx],c='k',marker='x')
plt.errorbar(8979.86,mzls['parameters_phot'][5,idx], mzls['parameters_phot'][12,idx],c='k',marker='x')
plt.errorbar(9607.87,mzls['parameters_phot'][6,idx], mzls['parameters_phot'][13,idx],c='k',marker='x')
plt.plot(mzls['GU']['wave'][:], uJy2mAB(mzls['GU']['flux_ujy'][idx]))
plt.plot(mzls['GV']['wave'][:], uJy2mAB(mzls['GV']['flux_ujy'][idx]))
plt.plot(mzls['GI']['wave'][:], uJy2mAB(mzls['GI']['flux_ujy'][idx]))
plt.ylabel('Ab magnitude',size=10)
plt.xlabel('Wavelength',size=10)
plt.show()
```

[18] ✓ 0.1s