We've suggested here four tasks that, after completing the processing script for processing IRC+10246, will provide more of a demonstration of the current, and future capabilities of CASA SD. An accompanying document: "homework_ext_ans.pdf" contains a description of our answers, but answers may vary: there are occasionally a few ways of doing the same thing. Our intention in this case is to introduce capability by specifying the tasks we hope you would use.

Some general tips:
1.Do not forget to prefix each CASA session with *asap_init* to initialise the SD task suite.
2.To view spectral datasets, use sdplot(infile='*infilename*')
3.To view image datasets, use viewer(infile='*infilename*')

1. SDcal
The task SDcal is designed to remove a lot of the tedium of processing and calibrating SD data, by incorporating all calibration steps: averaging, bandpass removal, calibration and baselining in one almighty task. In fact, starting with cycle 1 data, SDcal compresses the entire process of calibration and gridding into to steps: calibration, then gridding.

While the IRC+10246 data do not suffer significantly from strongly-varying spectral baselines, its nonetheless a worthwhile exercise to experiment a little with it, starting from the calibration steps (stage #6) from the "workflow.py" script, i.e. using as input, X44.PM02.asap.nowvr. Recall that the cube data is in IF[1] and the total power data is in IF[0]. Using the Viewer, and: tools > spectral profile, it's possible to view the spectrum of IRC+10246 (note, the view is pretty dully, use the cross hairs selector at the top of the view, and click approximately in the middle of the view region until a spectrum is plotted).

2. SDbaseline
This task removes a frequency-varying "baseline" component. We've generated some baseline-affected synthetic data "fake1dspec.asap". (Note, these "data" are entirely synthetic. Any similarity to identifiable transition lines, alive or dead, is entirely coincidental) for you to test sdbaseline.
I suggest you test different polynomial orders, different sinusoid or spline fits. and different masking alternatives (e.g. 'auto', 'interactive'). Note that "auto" mode works better for spectra with only two or three transitions. Feel free to try a two-step process, or csplines with high numbers of pieces.

3. SDimprocess
This task helps to remove gain variations in scanned data. Working in the Fourier domain, scanning striations manifest as a comb function - by switching affected parts of the FFT map with the less-affected parts of a second map, scanned in a different direction, the data can be "corrected" and cleaned a little.
We've generated more dummy data - maps of a cloud (or continuum region), with strong gain variations in two orthogonal directions: hscan4d.image and vscan4d.image (no telescope front-ends were harmed in the making of these data).

use sdimprocess to generate a cleaned image from these two datasets. Note that the angle specified in the direction parameter, is positive clockwise, and EAST=0 deg.

4. SDtpimaging
This task removes a “baseline” in the image domain, from a total power image. we can remove a 1st order brightness offset from the moon_atf.ms dataset, by fitting to the edges of the map.
to simplify this process for you, these parameters are the most useful when processing the moon_atf.ms dataset, with sdtpimaging.

SDtpimaging *overwrites* the input file, so it might be handy to make two copies of the file for your review. you can do this with:

```
shutil.copytree('moon_atf.ms','moon_atf.bl.ms')
```

Note that warnings about “time dependent feed table encountered” can be ignored.

```
imsize=[200,200]
cell=['0.2arcmin','0.2arcmin']
phasecenter="AZEL 187d54m22s 41d03m0s"
ephemsrcname='Moon'
```