Standards for PDRT model files & directories

Version Date: Feb 04, 2021

Files

The fundamental storage unit for a PDR model to be used in the PDR Toolbox is a FITS file. The axes of the FITS file are hydrogen nucleus volume density *n* (AXIS1) and radiation field strength F_{FUV} (AXIS2). The pixel values are predicted ratio of two spectral lines, e.g [O I] 63 μm/[CII] 158 μm or the predicted intensity, e.g. [CI] 609 μm. Calculation is typically done in 0.125 steps in the log10, that is, the axes are logarithmic with CDELT*n* = 0.125. All files shall conform to the FITS 4.0 Standard as much as possible. Units will be in CGS: *n* shall be cm^{-3} and of F_{FUV} shall be erg s^{-1} cm^{-2}. Intensity units should be erg cm^{-2} s^{-1} sr^{-1}. The Toolbox will convert F_{FUV} to G_0 Habing, χ Draine, etc units if user requests.

Data are stored in IEEE floating point; therefore use of BSCALE and BZERO is not recommended. Table 1 lists the recommended keywords for PDR model files. It is expected that model authors will provide as many of these as are relevant to code, but some may be peculiar to a given code (e.g., MASS). See the FITS Standard for more details on general keywords.

File names should be consistent across models internally within a production and externally between productions. The file name format for intensity ratios is *numerator_denominator.fits,* where the ratio data in the file are *numerator/denominator,* e.g., OI63_CII158.fits for [O I] 63 μm/[C II] 158 μm, OI145+CII158_FIR.fits for ([O I] 145 μm+[C II] 158 μm)/FIR or CO76_CO43.fits for CO(J=7-6)/CO(J=4-3). Isotopic numbers should be put in the usual place, e.g. 13CO for ^{13}CO or C17O for C^{17}O. So, e.g., 13CO43_13CO21.fits for ^{13}CO(J=4-3)/^{13}CO(J=2-1). The file name format for intensities is similar, e.g. 13CO32.fits for ^{13}CO(J=3-2).

We must deal with models of varying metallicity Z. In the classic PDRT models we encoded Z into the file name, e.g., siii35feii26z1, siii35feii26z3, but only when there were multiple metallicity models available. In the new structure, different metallicities are in subdirectories and similarly for constant density and constant pressure models (see below). The actual lookup of what file contains what lines and metallicities is driven by an external table. We don't want to have to read in a FITS file to know what

metallicities are present. The same holds if we choose to make models that vary something else such as maximum extinction (AVMAX).

Storage on Disk and Access

The models are distributed as part of the PDRT python installation (http://pdrtpy.readthdocs.io). They are in a *models* subdirectory within the Python module. We further break these apart by origin and version, with a symbolic link *latest* pointing to the latest version, e.g.

```
models/
    wolfirekaufman/
                version2006
                    constant_density
                        z=1
                            CI370_CI609.fits
                            ...
                            models.tab
                        z=3
                        ...
                    constant_thermal_pressure
                        ...
                version2020
                latest -> version2020
    kosma-tau/
                version2019a[1]
                version2016
                latest -> version2019a
```

How versions are numbered is up to the software authors, but they should try to follow an internally consistent convention.

In the lowest leaf of the tree are the FITS files for each ratio and intensity and a file called *models.tab* which is an IPAC table format ASCII file giving the parameters of each file. See here for an example.

The index of all possible model sets is in another IPAC format ASCII table *all_models.tab* in the *tables* directory of the code.

[1] This is just an example. We don't have a full kosma-tau model set yet.

Table 1: Required FITS Keywords for PDRT Models

Generic FITS Keywords		**Model Specific Keywords**	
KEYWORD	**Explanation or Proposed Value**	**KEYWORD**	**Explanation or Proposed Value**
SIMPLE	T	VERSION	Version of models or modeling software
BITPIX	-32 or -64	DVDOP	Turbulent Doppler Velocity (km/s)
NAXIS	2 or 3, if 3 then NAXIS3 shall be 1	ABUNDC	Carbon Abundance (C/H)
NAXISn	Dimensions of each axis	ABUNDO	Oxygen Abundance (O/H)
CDELTn	Coordinate scale for each axis	ABUNDSI	Silicon Abundance (Si/H)
CRPIXn	Reference pixel of each axis	ABUNDS	Sulfur Abundance (S/H)
CRVALn	Value at reference pixel for each axis	ABUNDFE	Iron Abundance (Fe/H)
CUNITn	cm-3 for n, erg s-1 cm-2 for F_{FUV}	ABUNDMG	Magnesium Abundance (Mg/H)
CTYPEn	log10(n) for AXIS1 log10(F_{FUV}) for AXIS2	ABUNDPAH	PAH Abundance (PAH/H)
TITLE	e.g. '[O I] 63 micron/[C II] 158 micron'	ABUNDDST	Dust Abundance wrt local ISM
BUNIT	‘ ‘ empty string for ratios [unitless] ‘erg cm-2 s-1 sr-1’ for intensities	ABUNDZ	Dust and Metals wrt local ISM. Z=1 means solar
ORIGIN	institution responsible for creating	ABUND13C	Carbon 13 Abundance (13C/H)
AUTHOR	who compiled the data in the header	ABUNDF	Fluorine Abundance (F/H)
DATAMIN	Minimum of the data	ABUNDN	Nitrogen Abundance (N/H)
DATAMAX	Maximum of the data	ABUNDHE	Helium Abundance (He/H)
DATE	ISO-8601 format *YYYY-MM-DDThh:mm:ss* **UTC**	AVMAX	Maximum Visual optical depth (mag)
HISTORY	Processing history (multi-line)	MASS	Clump Mass (solar mass)
OPTIONAL KEYWORDS:		TAUUVV	Ratio of tau_UV/tau_V
COMMENT	Additional comments (multi-line)	TAUFUVV	Ratio of tau_FUV/tau_V
REFERENC	Bibliographic DOI or ADS identifier	HEAT	Photoelectric Heating model, e.g. “BTF’ for Bakes & Tielens Flat PAHs
		MODELTYP	Description of pixel values, e.g. “Ratio” or “Intensity”

I have been thinking about how best to accommodate models that have a variety of parameters (z, clump/non-clumpy, maximum clump mass, constant thermal pressure, constant density, etc), in the ModelSet API. Use of variable keyword arguments (**kwargs in Python) seems like a good method.

ModelSet(name=,z=,medium="clumpy, non-clumpy,pressure,density" mass=100, etc)

wk2006 = ModelSet(name="wk2006",z=1,type="density")
ktau = ModelSet(name="WD01-7",z=1,mass=100,medium="clumpy")

Keywords have the nice property that unused ones can be ignored, so e.g.

wk2 = ModelSet(name="wk2020",z=1,mass=100,medium="constant pressure")

The mass keyword would be ignored (either silently or we could warn)
We can make name, z and medium required keywords the rest optional with sensible defaults.
Or non-optional depending on medium - e.g. you have to give mass if medium="clumpy"