Geometric Transformation

Story

I want a way to quickly and efficiently go from un-dispersed to dispersed images reference frame. It is therefore an (x,y) (dx,dy) transformation but wavelength factors in this, as well as where on the detector we are. It is therefore, in a more general way a transformation from (x,y,lambda) (dx,dy, lambda), where (x,y) is the location on the detector to use to account for any field dependency.
I can thing of several things I would use this for:

- Simulating: In this case I want to be able to compute the offsets (dx,dy) coordinates in the dispersed image where light originating from (x,y) in the undispersed image and at a particular wavelength lambda ends up.
- Extracting: This is a reverse operation. I know the coordinates in a dispersed spectrum (x',y') and therefore (dx,dy) and I want to know what the wavelength of that light is since I know it originates from the pixel (x,y) in the direct image
- In between cases: Some times, I want to be able to go back and forth from (x,y,lamba) to (dx,dy,lambda) with missing values of x, y or lambda, or dx,dy,lambda so I need functions that give me the flexibility to solve any of the in between problems, equally as fast.

I envision that I will need to compute this several millions on times so I want to keep things as fast as possible. It could be a 2D polynomial solution, which can be inverted either analytically or numerically in the case of higher polynomial orders (as in UVIS).

I want this system to be flexible and extendable to highly distorted spectra so I do not want the calibration to be dependent on the x or y coordinates, or the pathlength along the trace (which is difficult to compute and invert) or lambda directly. The choise of x or y is a bad one for example in cases where the spectra curve dramatically and either of these coordinates become a poor ways to measure where the trace is. Instead I want a system where each of this transformation is given as a function of an intermediate variable t which can then be defined as best fit a particular disperser. Hence the functions will be dx = fx(x,y,t) dy= fy(x,y,t) lambda=fl(x,y,t) and the inverse functions x = fx'(dx,dy,t), y = fy'(dx,dy,t), lambda=fl'(dx,dy,t). The variable t can be defined as t = x in the case of a simple x-direction disperser. It can also be defined as t = lambda, or t = (lambda-l0)/(l1-l0) where l0 an dl1 and the minimum and maximum wavelength in the dispersed spectrum. The latter results in 0<t<1 which could be beneficial to some.

Inputs

- A grism configuration file that described the transformations
- A set of coordinates and wavelenghts, etc...

Outputs

- As set of coordinates and wavelengths, etc...

Computations

- Computations are handles by a small module which simply implements the equation described above, using parameters values determined during the course of the instrument calibration.

Drawbacks

- The use of the variable t provides more flexibility but some bad choices can still be made. How to parametrized t should be defined by balancing adequacy, speed, as well as the ability of the instrument team to calibrate the grism in that manner