Publicly Available Tools and the Overall Process for Generating Observing Scenario (OS) 11 Polarization Datasets for the Wide-field-of-view Shaped Pupil Coronagraph

Jessica Gersh-Range
Princeton University
May 2022

Publicly Available Tools

- PROPER (John Krist)
  - An optical propagation library available at http://proper-library.sourceforge.net
- Roman Phase C PROPER Prescription (John Krist)
  - PROPER model for the final CGI layout and coronagraph masks
  - Available at https://sourceforge.net/projects/cgisim/files/
- CGISim (John Krist)
  - Wrapper code for the Roman Phase C PROPER Prescription that defines many of the necessary parameters and generates an image
  - Most of the setup happens in cgisim.rcgisim.py, which is called by a wrapper function that contains the list of parameters
  - Includes examples
  - Available at https://sourceforge.net/projects/cgisim/files/
- EMCCD Detect (Bijan Nemati and Sam Miller)
  - Detector model that can be used for adding detector noise
  - Available at https://sourceforge.net/projects/cgisim/files/

Implementing a Time Series Simulation

- A time series simulation is implemented by a wrapper function that defines all of the parameters; iteratively calls, for each time step, a modified version of cgisim.rcgisim.py that returns an array of star-centered electric fields; applies jitter using an approximation developed by John Krist; and constructs the normalized intensity. (This process is not optimized for speed.)

- When defining the parameters, this wrapper function:
  - Specifies which coronagraph mode, coronagraph type, and bandpass to use (cgi_mode = 'excam', cor_type = 'spc-wide', bandpass = '4')
  - Specifies which polarization states are used (polaxis = 10 generates an incoherent image for each of the four polarization components)
  - Specifies the spectral type (star_spectrum) and V magnitude (star_vmag) of the star for each time step. The "star" column of the OS 11 inputs file, spc_wfov_os11_inputs.fits (available via https://roman.ipac.caltech.edu/sims/Coronagraph_public_images.html), identifies reference star observations with 2 and target star observations with 1.
  - Specifies the deformable mirror (DM) pistons, dm1 and dm2, ('use_dm1':1, 'dm1_m':dm1, 'use_dm2':1, 'dm2_m':dm2), with DM drift implemented as

$$DM_{new}\left[x,y\right]=DM_{old}\left[x,y\right]\left(1+0.026\Delta_{temp,K}\right).$$

    There are four DM solutions provided in the "examples" folder of Roman Phase C PROPER that can be used to generate the initial dark hole, and the DM temperatures are provided in spc_wfov_os11_inputs.fits.

The parameters are detailed in the documentation for the Roman Phase C PROPER Prescription and in the documentation for CGISim:
John Krist, "Roman Phase C PROPER Prescription for IDL, Python, and Matlab," Public version 1.2.8, California Institute of Technology (2021).
John Krist, "CGISim (Roman coronagraph) Simulator," version 3.0.1, Jet Propulsion Laboratory, California Institute of Technology (2021).

Implementing a Time Series Simulation

- The wrapper function also incorporates multiple errors by setting parameters in CGISim:
  - Optic fabrication and alignment errors
    ('use_errors' : 1)
  - Wavefront error changes from thermal drift
    ('zindex':np.arange(4,46), 'zval_m' set to the values in columns 4-45 of the appropriate row of spc_wfov_os11_inputs.fits, with the column numbers starting at 0)
  - Pupil shear
    ('cgi_x_shift_m' and 'cgi_y_shift_m' set to the values in columns 66 and 67, respectively, with the column numbers starting at 0)
  - DM shear
    ('DM1_x_shear', 'DM1_y_shear', 'DM2_x_shear', and 'DM2_y_shear' set to the values in columns 68-71)
  - SPAM (shaped pupil mask) shear
    ('SPAM_x_shear' and 'SPAM_y_shear' set to the values in columns 74 and 75, respectively)
  - LSAM (Lyot stop) shear
    ('LS_x_shear' and 'LS_y_shear' set to the values in columns 76 and 77, respectively)

- The focus correction mechanism offset to compensate Z4 is incorporated by setting 'foc_mech_corr' equal to the value in column 80.

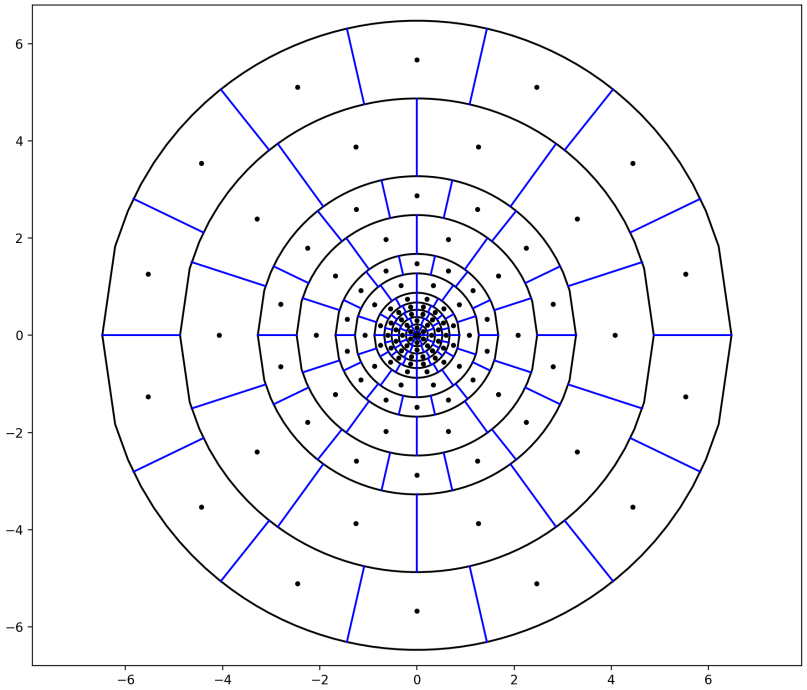
- Adding jitter is a more involved process than simply setting a parameter. The jitter distribution is represented by a set of source offsets, and the intensity for each offset is approximated and weighted. The weighted intensities are then combined to form the final intensity.

Jitter approximation reference: John Krist, private communication, (2022).

Procedure for Adding Jitter

Setup (Only done once):

Source offsets and the areas they represent



- Determine a set of source offsets (x,y) to represent the jitter distribution, and calculate the area of the region, A(x,y), represented by each offset.
  This set contains 125 offsets,
  with offsets <= 0.6 mas spaced by 0.15 mas
  and with larger offsets spaced by
  up to 1.6 mas.

- Calculate the star-centered electric field $E_0\left(0,0,\lambda,p\right)$ for the first time step for each wavelength λ and each polarization *p*.

- Create a library of the differences between an offset electric field for the first time step and the star-centered electric field for the first time step:

$$\Delta E_0\left(x,y,\lambda,p\right)=E_0\left(x,y,\lambda,p\right)-E_0\left(0,0,\lambda,p\right)$$

- It is also possible to precompute all of the weights for the offset electric fields at each time step.

Jitter approximation reference: John Krist, private communication, (2022).

Procedure for Adding Jitter

Calculating the jittered intensity:

For each time step *i*,

1. Calculate the jitter-free star-centered electric field, $E_i\left(0,0,\lambda,p\right)$.

2. For each offset (x,y), calculate a broadband intensity $I_i\left(x,y,p\right)$ by calculating

$$I_i\left(x,y,\lambda,p\right)=\left|E_i\left(0,0,\lambda,p\right)+\Delta E_0\left(x,y,\lambda,p\right)\right|^2,$$

   weighting each intensity in the array by the counts for the appropriate wavelength, and combining the images with the same polarization component but different wavelengths. Then, multiply the broadband intensity by the predetermined weight for the appropriate offset.

3. Calculate $I_i\left(p\right)$ by summing the weighted broadband images with the same polarization component but different offsets.

This jitter calculation process is an approximation that reduces the calculation time to a reasonable amount.

Jitter approximation reference: John Krist, private communication, (2022).

Assembling the Final Datasets

In CGISim, polaxis is the parameter that specifies which polarization components to calculate:

- polaxis = 10: “All-polarization” mode
  CGISim calculates the electric fields for all four polarization components
  (-45 deg in, Y out; -45 deg in, X out; 45 deg in, X out; and 45 deg in, Y out).

- polaxis = 5: “x-polarized” mode
  CGISim calculates the electric fields for the two relevant polarization components
  (-45 deg in, X out and 45 deg in, X out).

- polaxis = 6: “y-polarized” mode
  CGISim calculates the electric fields for the two relevant polarization components
  (-45 deg in, Y out and 45 deg in, Y out).

The CGISim code is written to provide the final intensity, but it can be modified to provide the electric fields for each polarization component at each wavelength along with the counts for each wavelength.

The transmission in the polarized modes (polaxis = 5 or 6) is 45% of the transmission in the all-polarization mode.

This means, for example, that the intensity of the -45 deg in, Y out polarization component for a y-polarized simulation, $I_{ypol_{-45in,Yout}}$, is thus 0.45 of the intensity of the -45 deg in, Y out polarization component for the all-polarization simulation, $I_{allpol_{-45in,Yout}}$: $I_{ypol_{-45in,Yout}} = 0.45 I_{allpol_{-45in,Yout}}$.

Reference: John Krist, “CGISim (Roman coronagraph) Simulator,” version 3.0.1, Jet Propulsion Laboratory, California Institute of Technology (2021).

Assembling the Final Datasets

The final datasets are all based off of a time series simulation with polaxis = 10.

For each time step, a modified version of CGISim is used to determine the star-centered electric field for each polarization component at each wavelength. These electric fields are used to calculate the jittered intensity for each of the four polarization components.

These intensities are labeled as the “unpolarized” intensities for the polarization components, and they are multiplied by 0.45 to obtain the “polarized” versions.

The final intensity for the unpolarized case, I_{unpol}, is calculated by averaging the four intensity components. The calculations are repeated for a version of the coronagraph without the focal plane mask (FPM), and the normalized intensity is calculated by dividing I_{unpol} by the maximum value of the final intensity for the “noFPM” case, $I_{unpol,noFPM}$:

$$NI = \frac{I_{unpol}}{max(I_{unpol,noFPM})}.$$

The final intensities for the polarized cases are calculated similarly, except that only the two relevant intensity components are averaged instead of all four.