Jet Propulsion Laboratory
California Institute of Technology

Using FALCO with the Phase C Roman CGI PROPER Model

A.J. Riggs

November 3, 2021

Jet Propulsion Laboratory, California Institute of Technology

Overview

1. Software packages used

2. Changes from Phase B to Phase C
   - For the PROPER model
   - For FALCO

3. Installation and setup
   - In Matlab
   - In Python

4. Example scripts and options

Software Packages Involved

- PROPER
  - **General-purpose library** of optical propagation functions
  - Available in **IDL, Matlab, & Python**

- roman_phasec_proper
  - **Diffraction model** of the Roman CGI in Phase C. Uses PROPER.
  - Does not include models of the Wollaston prisms or Amici prisms
  - Includes telescope & CGI optics, aberrations, polarization, DMs, and masks
  - Available in **IDL, Matlab, & Python**

- CGISim *(not utilized here)*
  - **Python-only wrapper** around roman_phasec_proper Python model
  - Includes stellar spectra and flux prediction
  - Produces intensity images, optionally with EMCCD noise
  - Primarily created for for single-image generation to investigate phase retrieval and image morphologies for exposure time estimation

- FALCO
  - **Package for performing** wavefront sensing and control (**WFSC**) for several coronagraph types.
  - Includes example scripts to run pair-wise probing and EFC.
    - Can be used as a **wrapper for PROPER models**
    - *Due to extra complexity of CGISim compared to the PROPER model, FALCO cannot currently be used as a wrapper for CGISim.*
  - Coming soon: algorithms for alignment and calibration of masks and deformable mirrors
  - (Also includes Zernike WFS mode, but not with a PROPER model.)
  - Available in **Matlab** & **Python**

Changes from Phase B to Phase C

PROPER model changes:

- Slightly different Roman pupil and baseline CGI masks.
- Data for all high-contrast mask configurations are now included (even unsupported ones).
    - The model allows for loading unsupported masks with different flags.
- More realistic mirror surfaces or measurements (when allowed) are included.

FALCO upgrades:

- Added unit tests and continuous integration to verify functionality.
- Code in whole package is much cleaner now.
- Added capability for (and example of) multi-star wavefront estimation and control.
- Added option for peak-normalized EFC, which minimizes *normalized* intensity instead of just intensity. CGI will do peak-normalized EFC.
- New implementation of tied DM actuators and the “neighbor rule”. Same as the CGI algorithm.
- Basic detector noise is now an option for simulated images.
- New compact model option that eliminates rotation between conjugate planes. (Makes it easier to tell if masks and DMs are oriented correctly.)

Matlab Setup

- Download *roman_phasec_v*.zip* and its manual from CGISim. Unzip the file where you want to keep the folders.

- In a Unix/Linux terminal, **clone** the falco-matlab repo into the folder you want, and then create and checkout a new branch. (PROPER is already included in falco-matlab.):
- `>> cd folder_to_hold_falco` *(choose the folder you want)*
- `>> git clone https://github.com/ajeldorado/falco-matlab.git`
- `>> git branch new-branch-name` *(choose your own branch name)*
- `>> git checkout new-branch-name`

- Go into the subdirectory falco-matlab/roman/ and open the files starting with names of EXAMPLE_*.m.
  - Near the top of the file EXAMPLE_main_Roman_CGI_any.m, replace the file path in the line:
    `addpath(genpath('~/Documents/Sim/cgi/public/roman_phasec_v1.2.4/matlab/'));`
    with the correct file path on your computer system.
  - Similarly, in all the config files (named EXAMPLE_config*.m), replace the file path in the line:
    `mp.full.data_dir = '/Users/ajriggs/Documents/Sim/cgi/public/roman_phasec_v1.2.4/phasec_data/';`
    with the correct file path on your computer system.

- Now you should be able to run EXAMPLE_main_Roman_CGI_any.m. To switch mask configurations, uncomment the config file (actually another script) named for the mode you want:

```
% EXAMPLE_config_Roman_CGI_HLC_NFOV_Band1()
% EXAMPLE_config_Roman_CGI_SPC_Spec_Band3()
EXAMPLE_config_Roman_CGI_SPC_WFOV_Band4()
```

Python Setup

- Download and unzip the .zip file containing PROPER and its manual from Sourceforge. Follow the brief Python installation instructions in the PROPER manual.

- Download and unzip the *roman_phasec_v*.zip* and its manual from CGISim. Follow the brief Python installation instructions in the roman_phasec manual.

- In a Unix/Linux terminal, **clone** the falco-python repo into the folder you want, and then create and checkout a new branch.:
- `>> cd folder_to_hold_falco  (choose the folder you want)`
- `>> git clone https://github.com/ajeldorado/falco-python.git`
- `>> git branch new-branch-name    (choose your own branch name)`
- `>> git checkout new-branch-name`

- Add the falco-python/ folder to your PYTHONPATH. (In Linux, this might be in your ~/.profile file. On a Mac, it might be in your ~/.zshrc file. Google it to be sure.)

- Go into the subdirectory falco-python/roman/

- Now you should be able to run EXAMPLE_main_Roman_CGI_any.m. To switch mask configurations, uncomment the config file named for the mode you want:

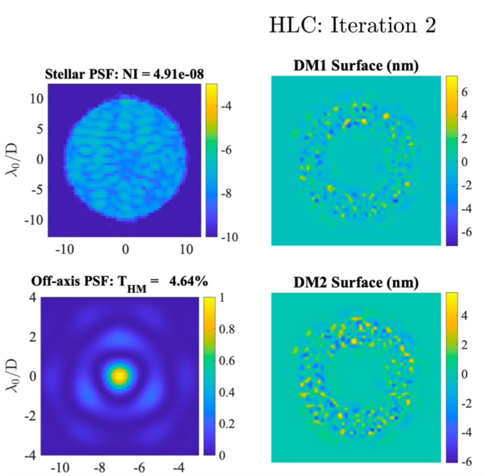
```
# import EXAMPLE_config_Roman_CGI_HLC_NFOV_Band1 as CONFIG
# import EXAMPLE_config_Roman_CGI_SPC_Spec_Band3 as CONFIG
import EXAMPLE_config_Roman_CGI_SPC_WFOV_Band4 as CONFIG
```

Example Output

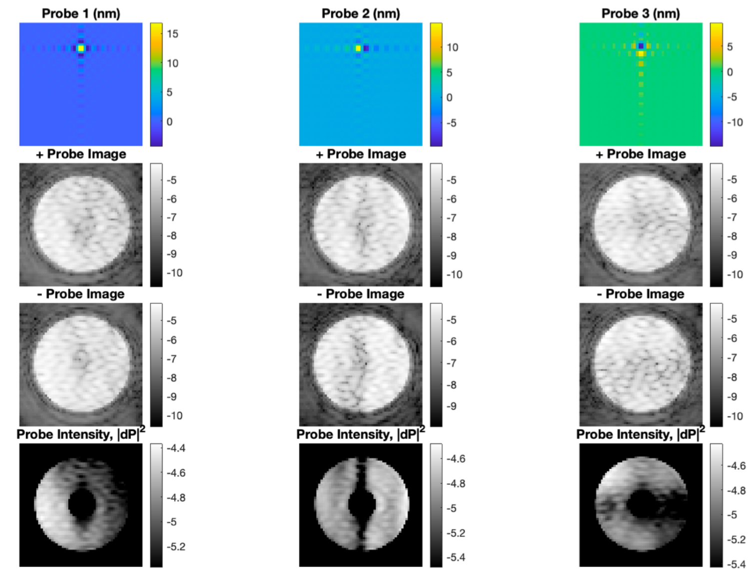
Command line reporting of progress

```
>> EXAMPLE_main_Roman_CGI_any
 Using 3 discrete wavelength(s) in each of 3 sub-bandpasses over a 10.0% total bandpass
Sub-bandpasses are centered at wavelengths [nm]:         555.83  575.00  594.17

DM 1-to-2 Fresnel number (using radius) = 932.0391
  Influence function padded from 84 to 84 points for A.S. propagation.
Computing datacube of DM influence functions... done.  Time = 1.3s
  Influence function padded from 84 to 84 points for A.S. propagation.
Computing datacube of DM influence functions... done.  Time = 1.3s
Saved the config file:  /Users/ajriggs/Repos/falco-matlab/data/brief//
Series0001_Trial0001_Roman_CGI_SPC_WFOV__config.mat

Beginning Trial 1 of Series 1.
WFSC Iteration: 1/5
Zernike modes used in this Jacobian:    1
DMs to be used in this iteration = [ 1 2 ]
Core throughput within the half-max isophote(s) = 4.66%         at separation = (7.0, 0.0)
lambda0/D.
Computing control Jacobian matrices ...
...done.  Time = 37.89
Estimating electric field with batch process estimation ...
Wavelength: 1/3 ... Mode: 1/3 ... Measured unprobed Inorm (Corr / Score): 2.85e-06
2.82e-06
Chosen probe intensity: 1.34e-05
Actual Probe 1+ Contrast is: 1.49e-05
Actual Probe 1- Contrast is: 1.99e-05
Actual Probe 2+ Contrast is: 1.67e-05
Actual Probe 2- Contrast is: 1.72e-05
Actual Probe 3+ Contrast is: 1.68e-05
Actual Probe 3- Contrast is: 1.16e-05
*** Mean measured Inorm for probe #1  = 1.454e-05
*** Mean measured Inorm for probe #2  = 1.409e-05
*** Mean measured Inorm for probe #3  = 1.135e-05
15 of 1428 pixels were given zero probe amplitude.
Wavelength: 2/3 ... Mode: 2/3 ... Measured unprobed Inorm (Corr / Score): 2.47e-06
2.42e-06
Chosen probe intensity: 1.27e-05
Actual Probe 1+ Contrast is: 1.30e-05
Actual Probe 1- Contrast is: 1.76e-05
Actual Probe 2+ Contrast is: 1.43e-05
Actual Probe 2- Contrast is: 1.53e-05
Actual Probe 3+ Contrast is: 1.45e-05
Actual Probe 3- Contrast is: 1.01e-05
*** Mean measured Inorm for probe #1  = 1.283e-05
*** Mean measured Inorm for probe #2  = 1.234e-05
*** Mean measured Inorm for probe #3  = 9.839e-06
12 of 1428 pixels were given zero probe amplitude.
Wavelength: 3/3 ... Mode: 3/3 ... Measured unprobed Inorm (Corr / Score): 2.13e-06
2.07e-06
```

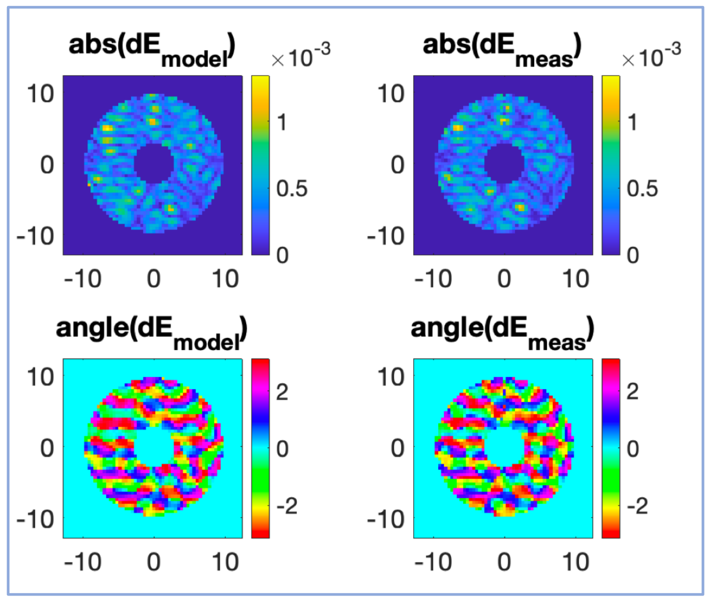
Visual reporting of progress



Pair-wise probing



Measured vs Model-Based Change in E-field



Settable Options

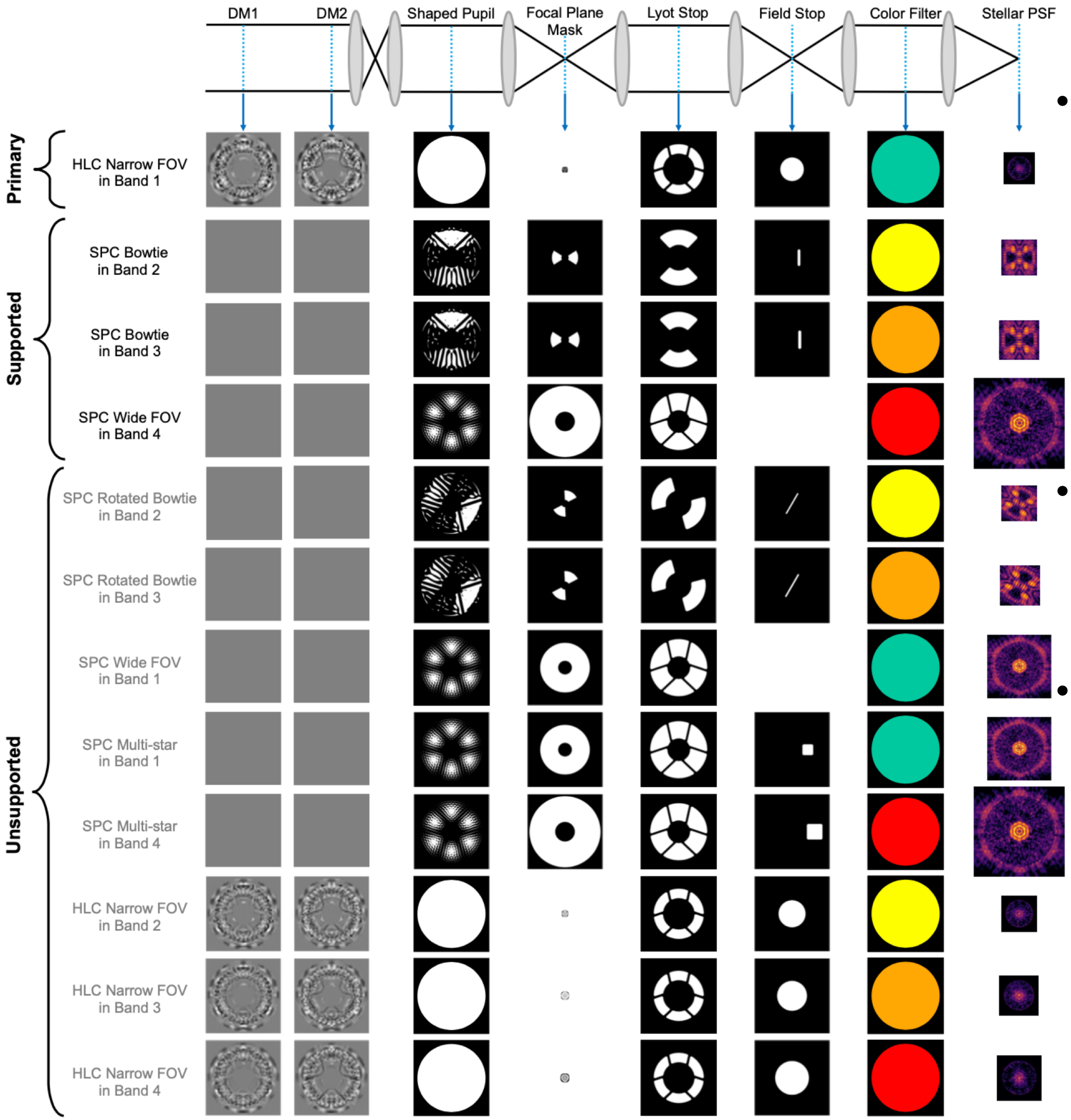
- The script *EXAMPLE_main_Roman_CGI_any* is pretty short.
  - This code block overwrites several config file settings to make the example run faster. Comment this block out when you’re ready.

```
%% SETTINGS FOR QUICK RUN: SINGLE WAVELENGTH, SINGLE POLARIZATION, AND NO PROBING
mp.fracBW = 0.01; %--fractional bandwidth of the whole bandpass (Delta lambda / lambda0)
mp.Nsbp = 1; %--Number of sub-bandpasses to divide the whole bandpass into for estimation and control
mp.Nwpsbp = 1; %--Number of wavelengths to used to approximate an image in each sub-bandpass
mp.full.pol_conds = 10;% [-2,-1,1,2]; %--Which polarization states to use when creating an image.
mp.estimator = 'perfect';
mp.flagParfor = false; %--whether to use parfor for Jacobian calculation
```

- Most options are set in the config files, *EXAMPLE_config_**
  - WFSC options and tuning parameters

- Lesser-used, optional parameters are defined in falco_set_optional_variables.m in Matlab or falco.setup.set_optional_variables() in Python.
  - (Not ideal, but done to maintain backwards compatibility with older scripts.)

Selectable Mask Configurations



- From arXiv, download the conference paper detailing all the mask configurations: "Flight mask designs of the Roman Space Telescope Coronagraph Instrument" https://arxiv.org/abs/2108.05986
- Only the high-contrast mask configs are options right now (← all the ones shown here).
- The low-contrast, traditional Lyot coronagraph mask configs could be done with the PROPER model but would not correctly model effects at large angular separations (aberrations, distortion, vignetting).

Unfinished Features (all to be added ASAP)

- More realistic control strategies and starting calibration settings

- Config files for unsupported mask configurations.
  - Mostly just need to computed initial DM settings.
  - The rest is just copy-paste and changing some file names.

- Some falco-python features lagging behind falco-matlab
  - Some newer features not included in Python version yet
    (e.g., MSWC, peak-normalized EFC, newer DM constraints)

- Issue with the *multiprocessing* package on Macs running Python >=3.8

Backup Slides

Jet Propulsion Laboratory
California Institute of Technology

Useful Links

- Paper showing all the Roman CGI mask configurations: https://arxiv.org/abs/2108.05986
- PROPER: https://sourceforge.net/projects/proper-library/
- CGISim: https://sourceforge.net/projects/cgisim/
- FALCO:
    - https://github.com/ajeldorado/falco-matlab
    - https://github.com/ajeldorado/falco-python
- lowfssim: https://github.com/nasa-jpl/lowfssim

Capabilities of FALCO + roman_phasec_proper

Capabilities

- Wavefront estimation with pairwise probing.
- Wavefront control with electric field conjugation (EFC).
- Inter-actuator voltage constraints on deformable mirrors
- Basic detector noise

What it cannot do:

- Any physics not in the PROPER model (e.g., observatory jitter and drift, dispersion from prisms)
- Low-order wavefront sensing for the Roman CGI. (See lowfssim instead)
- Mask and DM alignment and calibration. I am working on getting this code released as open-source.
- EMCCD noise or photon counting. These are part of CGISim and are not currently included.
- Truly-wide field-of-view imaging.
    - To avoid aliasing, the wavefront error maps do not contain info beyond 80 cycles/aperture, so the PSF will be near-perfect at large separations.
    - Most light baffles not included in PROPER model to save time, so vignetting at large separations not modeled correctly.