**WFIRST Phase B PROPER Prescription for IDL, Python, and Matlab**

V1.2, 28 May 2019

John Krist

# Introduction

There are two PROPER prescriptions, the "full" and "compact" versions: `wfirst_phaseb` and `wfirst_phaseb_compact`. The full version has every optic with aberrations, source offsets, Zernike aberrations, polarization aberrations, defocus and pupil imaging lenses, a pinhole at the FPM, and the ability to shift masks or the CGI. It uses PROPER to propagate from optic to optic. The compact version only allows source offsets and the specification of a file containing the FPM exit pupil field computed previously by the full one. It starts at DM1 and uses PROPER to propagate to DM2 and then back to DM1, after which FFTs are used to go from pupils to focal planes (MFTs are used for part of this in the case of the SPC). NOTE: The full prescription has a small amount of defocus that the compact one does not. This can be compensated with some Zernike defocus, as demonstrated in the *run_hlc* example.

# Setting Up

> *Note*: Both IDL and Python versions of PROPER support using FFTW or the Intel math library's FFT, and you are strongly encouraged to install either and activate PROPER's use of one of them, as detailed in the PROPER manual. Matlab already uses a version of FFTW.

### *Required data files*

The WFIRST CGI Phase B optical error maps, polarization aberration tables, and coronagraph mask files are distributed as a separate package, available from wfirst.ipac.caltech.edu. These must be installed prior to using the Phase B prescription.

> *Note*: Users are strongly encouraged to install the Phase B package on a local disk, otherwise accessing the optical error maps and coronagraph mask files over the network will lead to significant slowdowns.

### *IDL:*

PROPER must, of course, be installed, and it requires the IDL Astronomy User's Library. The PROPER library directory must be added to IDL_PATH using the IDL **`pref_set`** command, as described in the PROPER manual.

The user needs to edit the `wfirst_phaseb.pro` and `wfirst_phaseb_compact.pro` files and change the **`data_dir`** variable to point to the directory containing the Phase B maps and mask files (this directory should contain subdirectories like hlc_20190210, maps, pol, etc.).

The `wfirst_phaseb.pro` and `wfirst_phaseb_compact.pro` files can only be executed using PROP_RUN or PROP_RUN_MULTI from within the current directory. The user should copy all of the .pro files from the wfirst_phaseb distribution into their working directory.

### *Python:*

PROPER and its prerequisite package must be installed first. The Phase B data files must also be installed.

The Python version is distributed as the **wfirst_phaseb_proper** package. It uses the **numpy**, **scipy**, **astropy**, and **proper** packages, plus, if you want to run the demo scripts, **matplotlib**. To install, uncompress and untar the distribution and then switch to the *wfirst_phaseb_proper_v1.1* directory. In there, issue the following command:

```
python setup.py install
```

This will install the package into your local Python library (under *~/.local/lib*).

In order for the package to know where the maps and masks data files are located, you first need to switch to that directory (it should contain subdirectories like hlc_20190210, maps, pol, etc.), get into Python, and issue the commands:

```
import wfirst_phaseb_proper
wfirst_phaseb_proper.set_data_dir()
```

This will define the `data_dir` variable in the `__init__.py` file in your local library used by the package. This value can be overridden using the `data_dir` optional parameter.

To run either the full `wfirst_phaseb.py` or compact `wfirst_phaseb_compact.py` prescription, its source code needs to be copied from your local library into your working directory using the **`wfirst_phaseb.copy_here()`** command (the `wfirst_phaseb.py` or `wfirst_phaseb_compact.py` source code in the local library is never directly accessed):

```
import wfirst_phaseb_proper
wfirst_phaseb_proper.copy_here()
```

**NOTE:** Whatever versions of `wfirst_phaseb.py` or `wfirst_phaseb_compact.py` that you may currently have in your working directory will get overwritten.

**NOTE:** If you are using the **wcgisim** routine, it will copy the full prescription to the current directory itself.

The **`wfirst_phaseb_proper`** module must be imported when running the prescription. To execute the prescription using **`prop_run`** or **`prop_run_multi`**, the **`proper`** module must be imported.

### *Matlab:*

PROPER and its prerequisite package must be installed first. The Phase B data files must also be installed.

The user needs to edit the `wfirst_phaseb.m` and `wfirst_phaseb_compact.m` files and change the **`data_dir`** variable to point to the directory containing the Phase B maps and mask files (this directory should contain subdirectories like hlc_20190210, maps, pol, etc.).

The `wfirst_phaseb.pro` and `wfirst_phaseb_compact.pro` files can only be executed using PROP_RUN or PROP_RUN_MULTI from within the current directory. The user should copy all of the .m files from the wfirst_phaseb distribution into their working directory.

# General assumptions

This software represents the flight, not testbed, system. The effective DM tilts (about the Y axis) are each 5.7° (rotation about the Y axis) for the model's perfectly circular beam, which produce the same footprint on the DMs as the real system's slightly non-circular pupil and 9.65° DM tilts. The PROPER DM model uses the DM32.5 influence function with 0.9906 mm spacing. The wavefront is centered at the joint of the four central actuators.

## Default coronagraphs

Each design has particular hard-coded settings that may change with new designs: DM geometry (tilts, spacing, alignment), available FPM wavelengths (HLC), FPM dimensions (SPC), pupil sampling (e.g., 309 pixels for HLC, 1000 pixels for SPC), computational grid sizes, and bandpass central wavelength (*lambda0_m*). ***Be sure to take note of which wavelengths the HLC FPM is defined at. Use only these wavelengths when running HLC, otherwise it will use the FPM file for the closest wavelength.***

***HLC 20190210****:* Dwight's design for a 10% bandpass centered at 575 nm. This is the default coronagraph but can also be explicitly selected by setting the *cor_type* parameter to 'hlc'. It uses an r = 2.8 $\lambda_c$/D FPM and has a DM-solution-limited OWA of 9 $\lambda_c$/D. It has separate S and P modes computed for a 10° FPM tilt (though since the tilt has been reduced, the effective difference between the two is insignificant and only one should be used; the default is P). A r = 9 $\lambda_c$/D field stop at the back end is included in the full prescription (there is no stop in the compact model). By default it produces fields at a sampling of 309/1024 = 0.302 λ/D per pixel.

The FPM files are defined only for 19 evenly-spaced wavelengths spanning the 10% band (which also happen to include 7 evenly-spaced wavelengths), which are (μm):

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0.54625000 | 0.54944444 | 0.55263889 | 0.55583333 | 0.55902778 | 0.56222222 | 0.56541667 | 0.56861111 |
| 0.57180556 | 0.57500000 | 0.57819444 | 0.58138889 | 0.58458333 | 0.58777778 | 0.59097222 | 0.59416667 |
| 0.59736111 | 0.60055556 | 0.60375000 | | | | | |

Users should run at only these wavelengths; the prescription looks for the closest wavelength to the requested one.

Dwight provided DM wavefront solution maps for an unaberrated system; by default, they are NOT used, but if you want to use them, set *use_hlc_dm_patterns* to 1. DM actuator settings for the HLC solution (*hlc_dm*.fits*) in an unaberrated field are provided in the *examples* subdirectory of the Phase B distribution. Post-EFC DM settings for an aberrated system (including X+Y polarization) are also there (*hlc_with_aberrations_dm*.fits*). These are used by the *run_hlc* example. In Python, you can see where the distribution directory is with the following commands:

```
import wfirst_phaseb_proper
print( wfirst_phaseb_proper.lib_dir )
```

***HLC 20190206_v3:*** Erkin's design for a 10% bandpass centered at 575 nm. It is selected by setting *cor_type* to 'hlc_erkin'. It uses an r = 2.8 $\lambda_c$/D FPM and has a DM-solution-limited OWA of 9 $\lambda_c$/D. It has a 9 $\lambda_c$/D field stop in the full prescription. By default it produces fields at a sampling of 310/1024 = 0.303 λ/D per pixel. The FPM is defined only at the same 19 wavelengths specified above for HLC 20190210. At the moment there are no DM piston settings for this design.

***SPC 20190130:*** A.J.'s shaped pupil coronagraph for the IFS. It has an r = 2.6 – 9 $\lambda_c$/D bow-tie FPM with a 65° opening angle and is designed for an 18% bandpass. It is selected by setting *cor_type* to 'spc-ifs_short' for the $\lambda_c$ = 660 nm band and 'spc-ifs_long' for the 730 nm one. The default output sampling is 1000/4096 = 0.244 λ/D per pixel. At the moment there are no DM piston settings for this design.

***SPC 20181220:*** A.J.'s shaped pupil coronagraph for wide field of view. It is selected by setting *cor_type* to 'spc-wide'. Is has a r = 5.4 – 20 $\lambda_c$/D annular FPM. It operates in a 10% bandpass centered at 825 nm. The default output sampling is 1000/4096 = 0.244 λ/D per pixel. At the moment there are no DM piston settings for this design.

When running a SPC mask, matrix Fourier transforms (MFTs) are used to/from the FPM to provide fine (0.025 λ/D) sampling there.

# Execution

The WFIRST prescription is run like any other PROPER prescription, though it has its own set of optional parameters and the grid size parameter is not the same as typically used by other codes.

## *Calling Sequence*

The user passes to `prop_run` or `prop_run_multi` the name of the prescription ('wfirst_phaseb' or 'wfirst_phaseb_compact'), the wavelength (or array of wavelengths) at which fields will be generated, the output grid size in pixels, and any optional parameters as a structure/dict. The prescription returns the E-field and the sampling in meters (which is always 0 if running the compact model).

---

### *IDL*

The IDL call is (replace *wfirst_phaseb* with *wfirst_phaseb_compact*, as necessary):

`prop_run, 'wfirst_phaseb'`, *field, lambda_um, gridsize, returned_sampling_m, PASSVALUE*={…}

*or*

`prop_run_multi, 'wfirst_phaseb'`, *fields, lambdas_um, gridsize, returned_samplings_m, PASSVALUE*={…}

### *Python*

The Python call is, after importing the **proper** and **wfirst_phaseb_proper** modules and copying the prescription file to the local directory using `wfirst_phaseb_proper.copy_here()`,:

*field, sampling_m* = `proper.prop_run( 'wfirst_phaseb'`, *lambda_um, gridsize, PASSVALUE*={…} )

*or*

*fields, samplings_m* = `proper.prop_run_multi( 'wfirst_phaseb',` *lambdas_um, gridsize, PASSVALUE*={…} )

### *Matlab*

The Matlab call is (replace *wfirst_phaseb* with *wfirst_phaseb_compact*, as necessary):

[*field, sampling_m*] = `prop_run('wfirst_phaseb'`, *lambda_um, gridsize, 'passvalue', ….. )*

*or*

[*fields, samplings_m*] = `prop_run_multi('wfirst_phaseb'`, *lambdas_um, gridsize, 'passvalue', ….. )*

---

Note that the grid size is not the size of the computational grid, as is commonly assumed by PROPER. Instead, the computational grid size is hard-coded in the prescription and this parameter is instead the dimension of the output result (but only if the output is at the final image plane, otherwise it is ignored). **This value MUST be a power of two, since PROPER assumes it is and checks for it.**

In the case of **`prop_run`**, *lambda_um* is a single value and *PASSVALUE* points to a structure. When using the parallel processing of **`prop_run_multi`**, *lambdas_um* may be an array of wavelengths and/or *PASSVALUE* is an array of structures/dicts. You can either run different sets of parameters for a single wavelength, multiple wavelengths for a single set of parameters, or matched multiple wavelengths and parameters (in which case *lambdas_um* and *PASSVALUE* have same number of array elements).

*PASSVALUE* is a list of optional keyword:value pairs that override default values. The available parameters are listed in Table 1. An example of the syntax is:

**IDL**: PASSVALUE={use_errors:1, cor_type:'spc-ifs_long', use_dm1:1, dm1_m:dm1_array}

**Python**: PASSVALUE={'use_errors':1, 'cor_type':'spc-ifs_long', 'use_dm1':1, 'dm1_m':dm1_array}

**Matlab**:
```
optval.use_errors = 1
cor_type = 'spc-ifs_long'
use_dm1 = 1
dm1_m = dm1_array
field = prop_run( …….., 'passvalue', optval )
```

## Returned Values: Field properties and sampling

The prescription returns the complex-valued electric field and the sampling in meters. Be aware that both depend on which plane the prescription is propagating up to, depending on the parameters set by the user. If **`prop_run`** is used then the field will be a 2-D complex-valued array and *sampling_m* is a scalar. If **`prop_run_multi`** is used, the field will be a 3-D array ( [*n, n, nlam*] in IDL and Matlab, [nlam, n, n] in Python ) and *sampling_m* will be a vector of corresponding samplings (note here that *n* depends on where the propagation stops and may or may not be *gridsize*, and *nlam* is either the number of wavelengths or the number of PASSVALUE structures passed).

The default behavior is to propagate from the primary (full prescription) or DM1 (compact) through to the final image plane. The sampling there can be set using the *final_sampling_lam0* parameter in units of $\lambda_0$/D ($\lambda_0$ is the central wavelength of the bandpass, also known here as $\lambda_c$); if it is not specified, then the default sampling, which depends on wavelength, is used and so the user should then pay close attention to *sampling_m*. The field at the final image plane will be returned as a *gridsize* by *gridsize* pixel array. There are options to propagate just up to the CGI entrance (the FSM, without FSM aberrations applied) or the FPM exit pupil (actually the Lyot stop plane without aberrations between the FPM and Lyot stop applied). In these cases the returned arrays ignore the *gridsize* and *final_sampling_lam0* specifications. Instead, the dimensions and sampling are set by the pupil dimensions assumed in the code (the user should inspect the code to see what `pupil_diam_pix` is for the chosen coronagraph).

## DM Settings

Parameters are provided to alter the default DM actuator spacing (*dm_sampling*), tilt *(_xtilt_deg, _ytilt_deg, _ztilt_deg*), and centration relative to the wavefront center *(_xc_act, _yc_act*). The DM is the default PROPER model DM that used the Xinetics 32.5 influence function. The DM piston arrays (dm1_m, dm2_m) are 48x48 arrays giving piston in meters. They specify the height of delta functions that are convolved with the subsampled influence function and then resampled as needed. An isolated actuator with 1.0 nm of piston would have a maximum surface deformation of 1.0 nm. All of the actuators pistoned by 1.0 nm will produce a surface 1.48 nm high.

DM actuator piston maps are provided for selected modes in the *examples* subdirectory of the library directory. The library directory can be found in Python using:

```
import wfirst_phaseb_proper
print( wfirst_phaseb_proper.lib_dir )
```

Available DM Setting Files

| Filename | Purpose |
|---|---|
| hlc_dm1.fits, hlc_dm2.fits | HLC (Dwight's) DM solutions for unaberrated system |
| hlc_aberrated_dm1.fits, hlc_aberrated_dm2.fits | HLC (Dwight's) DM solutions for aberrated system, including X+Y polarization (polaxis=10); system error correction evenly split among DMs |
| errors_polaxis10.fits | DM solution (wavefront flattening) for aberrated system, including X+Y polarization (polaxis=10); for DM1 |

## Precomputing Aberrated Input Fields for the Compact Model

In cases where one wishes to use the compact model but with some representation of the system aberrations, the aberrated field at the exit pupil of the FPM can be computed as saved to a file using the full prescription. The compact model can then read this in.

To do so, run the full model with *end_at_fpm_exit_pupil* = 1 and *output_field_rootname* set to the rootname of the output file (the code will automatically add the wavelength and the .fits extension). Also, because you don't want the default HLC DM patterns nor the FPM included in the output files, you need to set *use_hlc_dm_patterns = 0* and *use_fpm = 0*. After generating the files (which are real and imaginary), you can run the compact model with *input_field_rootname* set to the same rootname; the code will read it in an use it as the entrance pupil (DM1) field.

## Offsets and Shears

Parameters (see Table 1) are provided to offset the source or shift a mask or the entire CGI. Source offsets can be made by introducing tip/tilt at either the primary or FSM. The offsets can be specified either in mas or $\lambda_0$/D, and produce the same direction of shift as seen in the final image plane. Pupil mask shifts (SPC pupil mask, Lyot stop) and the CGI as a whole (bulk shift at FSM) are specified as the fraction of the pupil diameter or meters. The FPM can be offset in X and/or Y in units of $\lambda_0$/D and along the optical (z) axis in meters.

The focus correction mechanism can be pistoned to adjust focus. Note that this only alters focus; any induced changes in other aberrations have not been accounted for.

## Example Programs

Some example programs are provided to demonstrate the use of the prescriptions. In Python you can copy the programs into the current working directory with:

```
import wfirst_phaseb_proper
wfirst_phaseb_proper.copy_examples_here()
```

*run_hlc*

Run Dwight's HLC through the full system over a 10% bandpass at $\lambda_0$ = 575 nm without any errors but using Dwight's DM wavefront maps, without error but using DM actuator pistons for the HLC solutions, and with errors and a post-EFC solutions. Also, compute the source offset by 7 $\lambda_0$/D, using its peak value to convert intensity to normalized intensity. The fields have a sampling of 0.1 $\lambda_0$/D. Note that 0.19 nm RMS of defocus is included – the full prescription has a small amount of defocus by default, and this compensates for it so that it produces the as-designed HLC result.

*run_hlc_erkin*

Run Erkins's HLC through the full system over a 10% bandpass at $\lambda_0$ = 575 nm without any errors and using Erkins's DM wavefront maps.

*run_spc_ifs,*
*run_spc_wide*

Like *run_hlc*, except for the SPC IFS mask over an 18% bandpass centered at 730 nm or the SPC wide field-of-view mask over a 10% bandpass centered at 825 nm.

*run_hlc_input_fields*

Precomputed aberrated input fields for the compact model by first running through the full model with errors turned on and the HLC DM patterns & FPM turned off, and the output file rootname defined. Then run these through the compact model. Compare the results with the full model (note that the full model has a field stop, while the compact model does not).

*run_flatten*

Compute contrast in the HLC without and with wavefront flattening using a previously-derived DM1 pattern.

Table 1. PASSVALUE Parameters

parameters also available to compact model are in blue

parameters NOT available to wcgisim are marked with (*)

## Central wavelength, final image sampling, coronagraph type, data directory

| Parameter | Valid value or type | Default | Description |
|---|---|---|---|
| lambda0_m (*) | meters | 575e-9 (HLC)<br>730e-9 (SPC IFS)<br>825e-9 (SPC wide) | Bandpass center wavelength; used to define source offsets & sampling; default is 575e-9 for HLC, 660e-9 for SPC IFS short, 730e-9 for SPC IFS long, and 825e-9 for SPC wide. |
| lam0 (*) | microns | | Alternative in microns to lambda0_m |
| final_sampling_m (*) | meters | 0 | If 0, sampling is dependent on coronagraph and wavelength; this value overrides final_sampling_lam0 if both are specified |
| final_sampling_lam0 (*) | $\lambda_0/D$ | 0 | If 0, sampling is dependent on coronagraph and wavelength |
| cor_type (*) | 'hlc', 'hlc_erkin', 'spc-ifs_short', 'spc-ifs_long', 'spc-wide', 'none' | 'hlc' | Coronagraph type |
| data_dir | string | User-specified | Directory containing Phase B error maps & mask subdirectories |

## Use/exclude CGI components

| Parameter | Valid value or type | Default | Description |
|---|---|---|---|
| use_aperture | 0 or 1 | 0 | 1 = use apertures (not recommended) |
| use_hlc_dm_patterns | 0 or 1 (HLC only) | 0 | 1 = Use designer's DM WFE maps with solution |
| use_fpm | 0 or 1 | 1 | 1 = use FPM |
| use_lyot_stop | 0 or 1 | 1 | 1 = apply Lyot stop |
| use_dm1<br>use_dm2 | 0 or 1 | 0 | Use DM pistons provided in dm1_m, dm2_m |
| use_field_stop | 0 or 1 (HLC only) | 1 | 1 = apply field stop |

## Aberrations

| Parameter | Valid value or type | Default | Description |
|---|---|---|---|
| polaxis (*) | -2 = -45°in,+Yout<br>-1 = -45°in,+Xout<br>0 = none<br>+1 = +45°in,+Xout<br>+2 = +45°in,+Yout<br>5 = mean of ±45°in,+Xout<br>6 = mean of ±45°in,+Yout<br>10 = mean of all axes | 0 | Defines axis of polarization aberrations. WFC should solve for 5, 6, or 10, then the solution evaluated separately using -1 & 1 for 5, -2 & 2 for 6, and -2, -1, 1, & 2 for 10. NOTE: With the compact model, this parameter must be used with input _field_rootname to read in the corresponding field produced by the full prescription. |
| use_errors | 0 or 1 | 1 | 1 = use optic fabrication & alignment errors |
| use_hlc_dm_patterns | 0 or 1 (HLC only) | 1 | 1 = Use Dwight's DM WFE maps with solution |
| zindex | integer ≥ 1 | 0 | Vector of Zernike polynomial indices (Noll) |
| zval_m | meters RMS WFE | 0 | Vector of Zernike coefficients (Noll) |

DM parameters

| Parameter | Valid value or type | Default | Description |
|---|---|---|---|
| use_dm1<br>use_dm2 | 0 or 1 | 0 | Use DM pistons provided in dm1_m, dm2_m |
| dm1_m<br>dm2_m | meters | 0 | 48x48 arrays containing actuator pistons |
| dm_sampling_m | meters | 0.9906e-3 | DM actuator spacing |
| dm1_xc_act, dm1_yc_act<br>dm2_xc_act, dm2_yc_act | actuators | 23.5 | Center of wavefront on DM in units of actuators (0,0 = center of 1st actuator) |
| dm1_xtilt_deg, dm2_xtilt_deg<br>dm1_ytilt_deg, dm2_ytilt_deg<br>dm1_ztilt_deg, dm2_ztilt_deg | degrees | 0<br>5.7<br>0 | Rotation of DM about specified axis |

Focal plane mask parameters

| Parameter | Valid value or type | Default | Description |
|---|---|---|---|
| use_fpm | 0 or 1 | 1 | 1 = use FPM |
| fpm_x_offset<br>fpm_y_offset | $\lambda_0/D$ | 0 | Shift of FPM in $\lambda_0/D$ |
| fpm_x_offset_m<br>fpm_y_offset_m | meters | 0 | Shift of FPM in meters |
| fpm_z_shift_m | meters | 0 | Shift of FPM along optical axis |
| pinhole_diam_m | diameter in meters | 0 | Pinhole at FPM plane (if non-zero) |

Source offsets (wavefront tilt at primary, FSM tilt)

| Parameter | Valid value or type | Default | Description |
|---|---|---|---|
| source_x_offset_mas<br>source_y_offset_mas | milliarcsec | 0 | Offset of source caused by tilt introduced at primary mirror |
| source_x_offset<br>source_y_offset | $\lambda_0/D$ | 0 | Offset of source caused by tilt introduced at primary mirror |
| fsm_x_offset_mas<br>fsm_y_offset_mas | milliarcsec | 0 | Offset of source caused by tilt introduced at FSM |
| fsm_x_offset<br>fsm_y_offset | $\lambda_0/D$ | 0 | Offset of source caused by tilt introduced at FSM |

Component offsets (CGI box, SPC mask, FOCM, FPM, Lyot stop)

| Parameter | Valid value or type | Default | Description |
|---|---|---|---|
| cgi_x_shift_pupdiam<br>cgi_y_shift_pupdiam | -1 to +1 as fraction of pupil diameter | 0 | Bulk shear of CGI at FSM relative to IC |
| cgi_x_shift_m<br>cgi_y_shift_m | meters | 0 | Bulk shear of CGI at FSM relative to IC in meters |
| focm_z_shift_m | meters | 0 | Piston of focus corrector mirror (note: only alters focus, not other aberrations) |
| fpm_x_offset<br>fpm_y_offset | $\lambda_0/D$ | 0 | Shift of FPM in $\lambda_0/D$ |
| fpm_x_offset_m<br>fpm_y_offset_m | meters | 0 | Shift of FPM in meters |
| fpm_z_shift_m | meters | 0 | Shift of FPM along optical axis |
| lyot_x_shift_pupdiam<br>lyot_y_shift_pupdiam | -1 to +1 as fraction of pupil diameter | 0 | Lyot stop shift relative to pupil diameter |
| lyot_x_shift_m<br>lyot_y_shift_m | meters | 0 | Lyot stop shift in meters |
| mask_x_shift_pupdiam<br>mask_y_shift_pupdiam | -1 to +1 as fraction of pupil diameter (SPC only) | 0 | SPC pupil mask shift relative to pupil diameter |
| mask_x_shift_m<br>mask_y_shift_m | meters (SPC only) | 0 | SPC pupil mask shift in meters |

## Phase retrieval parameters (defocus & pupil imaging lenses, pinhole)

| Parameter | Valid value or type | Default | Description |
|---|---|---|---|
| pinhole_diam_m | diameter in meters | 0 | Pinhole at FPM plane (if non-zero) |
| end_at_fpm_exit_pupil (*) | 0 or 1 | 0 | End propagation at FPM exit pupil |
| use_pupil_lens | 0 or 1 | 0 | 1 = Use pupil imaging lens |
| use_defocus_lens | 0, 1, 2, 3, 4 | 0 | 0 = no defocus lens, 1 = +18 waves P-V, 2 = +9 waves, 3 = -4 waves, 4 = -8 waves (@ 550 nm) |
| defocus | waves P-V @ 550 nn -8.99 to +55.9 | 0 | Lens inserted to provide this much defocus (alternative to use_defocus_lens) |

## Intermediate field input/output

| Parameter | Valid value or type | Default | Description |
|---|---|---|---|
| end_at_fpm_exit_pupil (*) | 0 or 1 | 0 | End propagation at FPM exit pupil |
| end_at_fsm (*) | 0 or 1 | 0 | End propagation at FSM |
| output_field_rootname (*) | string | null | Rootname of FPM exit pupil field |
| input_field_rootname (*) | string | null | Rootname of pre-computed FPM exit field (compact prescription only) |

Change log:

V1.0 – Jan 3, 2019: Initial release

V1.0a – Jan 9, 2019: Enabled *polaxis* parameter for the compact model; it can only be used with *input_field_rootname* to read in a field generated by the full prescription with *polaxis* defined and *output_field_rootname* specified. Changed the full prescription so that after the output field (*output_field_rootname*) is generated propagation will continue to the final image plane, unless *exit_at_fpm_exit_pupil* is set. Updated manual to note that the output grid size must be a factor of two due to the checking done by PROPER.

V1.0c – Feb 13, 2019: HLC design is now HLC 20190210 (*cor_type = 'hlc'*), which has separate S and P axes that can be selected using the fpm_axis parameter; SPC IFS design is now SPC 20190130 (*cor_type = 'spc-ifs'*) and its default central wavelength now 730 nm with an 18% bandpass; SPC wide field-of-view design is now SPC20181220 (*cor_type = 'spc-wide'*) with a central wavelength of 825 nm and a 10% bandpass. Changed DM tilts to 5.7° around the Y axis to agree with pupil orientation.

V1.1 – May 8, 2019 (Python) Created the wfirst_phaseb_proper package. Added support for mask shifts in meters.

V1.2 – May 28, 2019: Added Matlab version; added data_dir parameter; took maps and mask files out of Python distribution and put them into a separate archive