Simulating the WFIRST coronagraph Integral Field Spectrograph

Maxime J. Rizzo^{a,e}, Tyler D. Groff^a, Neil T. Zimmerman^a, Qian Gong^a, Avi M. Mandell^a,
Prabal Saxena^{a,e}, Michael W. McElwain^a, Aki Roberge^a, John Krist^b, A J Eldorado Riggs^b,
Eric J. Cady^b, Camilo Mejia Prada^b, Timothy Brandt^c, Ewan Douglas^d, Kerri Cahoy^d

^{a}NASA Goddard Space Flight Center, 8800 Greenbelt Road, Greenbelt, MD, 20771, USA
^{b}Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, USA
^{c}Institute for Advanced Studies, Princeton, NJ, USA
^{d}Massachussetts Institute of Technology, Cambridge, MA, USA
^{e}NASA Postdoctoral Fellow

ABSTRACT

A primary goal of direct imaging techniques is to spectrally characterize the atmospheres of planets around other stars at extremely high contrast levels. To achieve this goal, coronagraphic instruments have favored integral field spectrographs (IFS) as the science cameras to disperse the entire search area at once and obtain spectra at each location, since the planet position is not known a priori. These spectrographs are useful against confusion from speckles and background objects, and can also help in the speckle subtraction and wavefront control stages of the coronagraphic observation. We present a software package, the Coronagraph and Rapid Imaging Spectrograph in Python (`crispy`) to simulate the IFS of the WFIRST Coronagraph Instrument (CGI). The software propagates input science cubes using spatially and spectrally resolved coronagraphic focal plane cubes, transforms them into IFS detector maps and ultimately reconstructs the spatio-spectral input scene as a 3D datacube. Simulated IFS cubes can be used to test data extraction techniques, refine sensitivity analyses and carry out design trade studies of the flight CGI-IFS instrument. `crispy` is a publicly available Python package and can be adapted to other IFS designs.

Keywords: Integral field spectroscopy, coronagraphy, high-contrast, spectrograph

1. INTRODUCTION

Recent progress in the maturation of high-contrast techniques have put the goal of directly imaging extrasolar planets from space within reach. WFIRST is the first space observatory to baseline an active coronagraph for high-contrast imaging of exoplanets. It leverages the extreme stability of the space environment and the lack of atmospheric perturbations to reach raw contrasts levels orders of magnitude deeper than what can be achieved from the ground. This allows us to observe a class of planets which have eluded researchers until now: cold, giant exoplanets. Although a number of these planets have been detected through radial velocity techniques, they are too dim to be directly imaged at any wavelength with the state-of-the-art ground-based facilities. Characterizing the composition of this family of objects is a key step towards learning about systems which could harbor habitable, rocky planets, and will pave the way for missions with the ability to directly image exoEarths, like LUVOIR or HabEx.

The WFIRST Coronagraphic Instrument (CGI) will operate in several bands between 400 and 970 nm, observing the starlight that scatters off of the planet's atmosphere. In addition to being scientifically compelling, this wavelength range also leverages recent progress in photon counting detectors which offer the best possible noise characteristics to date,[1] as well as progress in coronagraphic mask designs which allow dark holes with bandpasses as wide as $\Delta\lambda/\lambda = 18\%$.[2] In order to reach extreme contrasts, CGI features two deformable mirrors operated in closed loop, with data measured directly at the focal plane, and using techniques such as electric field conjugation.[3]

CGI will also feature a lenslet array-based Integral Field Spectrograph (IFS)[4] as a backend instrument to the coronagraph. This instrument is planned to operate in three 18% bands between 600 and 970 nm. The IFS yields information on the spectral content of the source over the entire band without having to cycle through individual narrow-band filters. This allows us to extract spectra at each point within the entire field of view and presents two fundamental advantages: first, for the locations in the field of view where our planetary targets are located, we can obtain reflected light spectra with a modest spectral resolution over a wide band, hence measuring atmospheric absorption features and determining elemental abundances. Second, the information can be used during operations to actually reach the desired contrasts in the dark hole faster than traditional methods, as both the spatial and spectral dependence of the speckle field are measured at once. Until this method was successfully demonstrated by the PISCES instrument[5,6] at JPL's High Contrast Imaging Testbed,[7] the standard approach to reach broadband contrast was to cycle through narrow band filters spread over the band, which is costly both in terms of filter slots and in terms of time on target.

Because CGI requires a closed loop to successfully reach deep contrast and observe its targets, an extensive amount of work has been achieved to accurately simulate CGI and its components. This is critical to determine the sensitivity of the instrument to perturbations, the tolerances of its elements, and its overall on-sky science yield. In addition, because coronagraphic data products typically require a significant amount of post-processing to reach the contrasts that the science requires, it is difficult to assess the scientific yield of the instrument without properly modelling its data products. For this simulation effort, we need to simulate an actual CGI observation scenario as a time series of focal plane maps, which include perturbations from the telescope and coronagraph as well as detector noise over the entire duration of the observation. Until recently, this effort has largely been carried out by JPL[8] and was mostly focused on the Hybrid Lyot Coronagraph (HLC) design which is one of the two types of coronagraphs that is baselined for CGI. The IFS is currently baselined to be used exclusively with the Shaped Pupil Coronagraph (SPC).

In this paper, we discuss our work in extending the simulation efforts to the SPC and IFS. We use the existing simulation framework to simulate spatially and spectrally resolved focal plane time series, and propagate them through our own parametric IFS instrument simulator in order to generate noisy IFS detector maps. Because in an IFS the spectral and spatial information are encoded onto a single IFS detector map, we also need an algorithm to extract the 2D map back into a spatio-spectral cube. These cubes are the elemental IFS data products that can be made available to scientists during science operations of the observatory. By producing realistic time series of IFS data products, we can identify post-processing methods, observing scenarios, and instrumental parameters that optimize the science yield.

We organize our paper in two sections. Section 2 describes the open-source software package that we wrote to simulate the IFS and decode its detector plane. Section 3 discusses how we applied this software to the CGI-IFS, including a fiducial observing scenario and the results of a trade-off on the lenslet array sampling.

2. SOFTWARE DESCRIPTION

We develop the Coronagraph and Rapid Imaging Spectrograph in Python (`crispy`)*, an open-source software package which can produce end-to-end IFS simulations. It consists of three main parts: the IFS propagation from the lenslet plane to the IFS detector plane (Section 2.2), the detector model (Section 2.3), and the IFS extraction (Section 2.4). The software is constructed in the context of the WFIRST IFS, but is adaptable to all lenslet array-based IFS designs. The extraction module of `crispy` is also routinely used for PISCES operations at HCIT.[7] Before tackling the software, we first give a brief overview of the IFS instrument design.

2.1 IFS instrument design

The lenslet array-based IFS design and concept is presented in the PISCES publications.[5,6] An overall layout is repeated in Fig. 1. The CGI focal plane is sampled by a square lenslet array, to the left of Fig. 1. Each square lenslet focuses onto a pinhole array to limit crosstalk between lenslets. The entire field, composed of many spots corresponding to each lenslet/pinhole pair, is then dispersed by a prism and re-imaged onto a detector. Each lenslet now has a corresponding micro-spectrum on the final detector. By choosing a proper clocking angle of
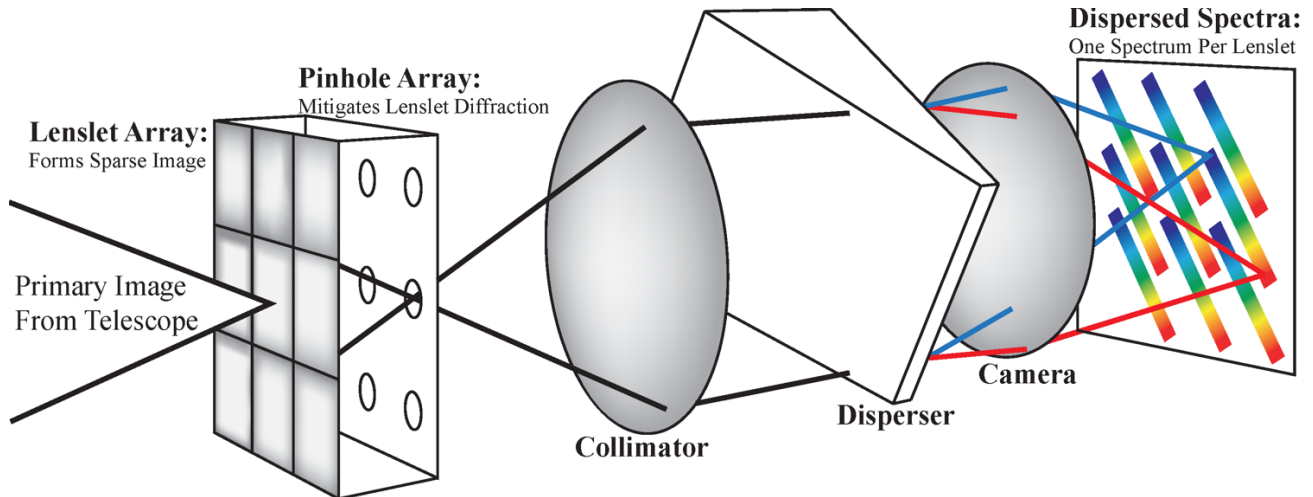
---

*https://github.com/mjrfringes/crispy

Figure 1. Layout of an lenslet array-based IFS. The focal plan of the coronagraph is sampled by the lenslet array. Each lenslet focuses a portion of the focal plane onto a pinhole (for stray light control and cross-talk considerations). The pinhole array plane is then collimated, dispersed with a prism and re-imaged onto the IFS detector array. Each lenslet, which corresponds to a small section of the coronagraphic image plane, now has its own micro-spectrum on the detector and a spatio-spectral cube can be reconstructed. The prism and lenslet array are clocked to avoid overlapping the microspectra and optimize the detector coverage.

the lenslet array with respect to the prism and detector, the microspectra can be aligned in order not to overlap with their neighbors. In the limit where the focal plane at the lenslet array is sampled at the telescope's Nyquist spatial frequency for a given wavelength, and sampled at the Nyquist spectral frequency by the prism/detector, the field at the IFS detector contains the same amount of information as the incident focal plane, but the light has been re-arranged.

Fig. 2 shows example detector maps of the IFS in the case of monochromatic and broadband flatfields. In the monochromatic image, the grid of spots represents light from all lenslet/pinhole pairs. Each monochromatic spot is called a PSFLet. In the broadband case, each lenslet has its own microspectrum. While the inherent monochromatic PSFLets are similar for neighboring lenslets, they are sampled differently by the detector pixel grid. In this particular example, the lenslet array is clocked 26 degrees with respect to the dispersion direction, to avoid overlap of the microspectra. This is best seen in the broadband image. The dispersion direction is aligned with the pixel lines for convenience.

2.2 IFS propagation

In order to simulate the IFS optics, the complex wavefront at each of the lenslets should be treated independently, propagated to focus, cropped with a pinhole mask, and further propagated through the collimator, disperser and camera optics (Fig. 1). This would represent a large amount of computing power and is impractical to use in simulations where hundreds of IFS propagations would be needed.

Instead, our approach is to consider only the total field intensity within a given lenslet, and multiply it by a PSFLet template at the correct location on the detector. This effectively ignores the complex nature of the local electric field, an assumption which is discussed below. Both the PSFLet templates and PSFLet locations can be determined from an optical ray trace software, from lab data (such as PISCES), or by using analytical functions.

We can thus reproduce monochromatic fields by placing the PSFLets at their location and appropriately discretizing them on the detector grid (such as on the left of Fig. 2). By summing monochromatic grids of spots at a much finer spectral sampling than the spectrometer's resolution, we can simulate detector maps with smooth broadband input (such as on the right of Fig. 2).

To achieve this, `crispy`'s propagator requires input datacubes representing the field at the lenslet array (e.g. see 3 slices out of a 45-slices input cube, Fig. 3). If the absolute fluxes are of importance to the user, the input
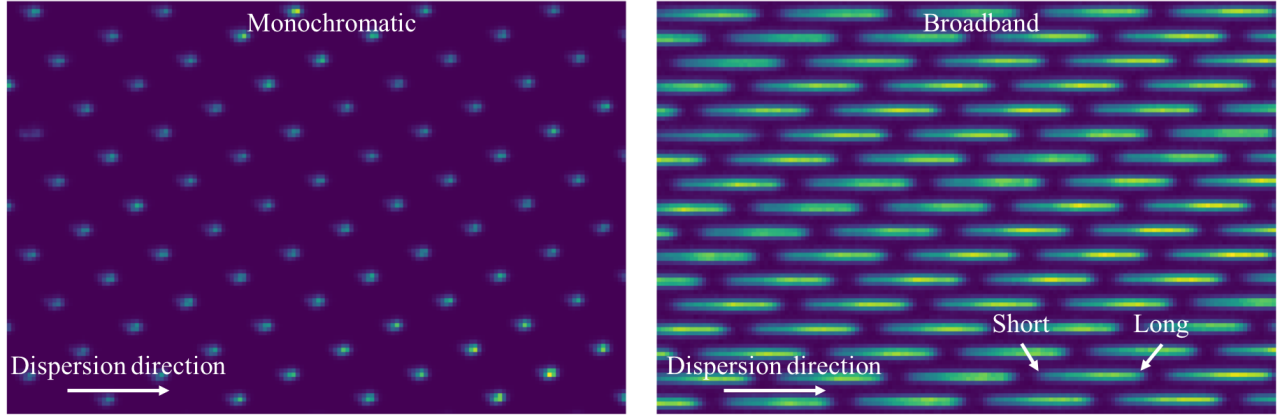
Figure 2. Left: Monochromatic grid of spots on the detector. This image is real data from the PISCES testbed and corresponds to a quasi-uniformly illuminated lenslet array in monochromatic light at 637 nm. Right: broadband microspectra grid from PISCES, corresponding to a quasi-uniformly illuminated lenslet array with broad spectrum from 600 to 720 nm. In both images, the dispersion direction is horizontal. The edges of the microspectrum correspond to the short and long end of the band.

cubes should be in units of photon flux density: each pixel represents the number of photons per second per nanometer of bandwidth within that pixel. The software calculates the appropriate bandwidth from the number of cube slices, assuming the provided list of wavelengths correspond to the band centers for each cube slice. Each slice is treated separately; `crispy` will rotate and bin the input pixel map onto the lenslet array using the input cube pixel scale, while ensuring flux conservation. The photon flux entering each lenslet, integrated over the short band from a given slice, is used to multiply the corresponding lenslet PSFLet, placed at the correct location on the detector (see the corresponding detector map in Fig. 4). Because each detector pixel can see photons from multiple wavelengths, all wavelength-dependent effects such as quantum efficiency (QE) and optical efficiency should be included within the input cube itself. Once the detector map is constructed, it is not possible to distinguish between photons from different wavelengths without running an extraction routine.
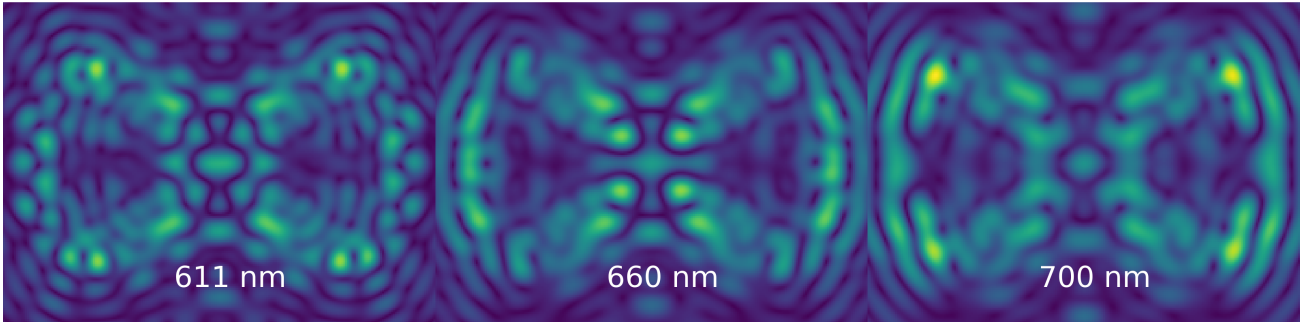

Figure 3. Input cube slices into the IFS, showing the coronagraphic PSF for WFIRST's Shaped Pupil bowtie mask for a 1000 s time step. The dark bowtie zones form the "dark hole", nominally from $\sim 3 - 9\lambda/D$ from the center of image. These models were generated by JPL's simulation pipeline for the "OS5" observing scenario (see Section 3.1). The three different panels show different wavelength slices, illustrating the change in the speckle pattern with wavelength. In the case of this example, the input cube has a total of 45 wavelength slices.

This provides a fast method to generate IFS cubes, and it can be fully parallelized since each slice is treated separately. The resulting detector maps that `crispy` generates are now in units of photo-electrons per second. This is the signal that an noiseless detector would read out.

The processing speed represents a vast advantage over other methods that require propagating each lenslet through all the optical elements individually. A previous version of Goddard's IFS simulation software uses PROPER[9] to simulate each lenslet, but it was taking ~11 h to run a single simulation on a standard 12-core

machine. We have shown excellent agreement between running the full complex propagation for each lenslet, using template PSFLets. The maximum change in the phase gradient across each lenslet is small, even including events required to dig the dark hole, such as DM probes. The resulting wavefront tilt would make the PSFLet coordinates shift by values much smaller than 1 detector pixel. On CHARIS, a lenslet array-based IFS at the Subaru telescope, poor atmospheric Strehl ratio can cause phase tilts across lenslets that can be considerably larger than the ones we expect on WFIRST; nevertheless, the PSFLet map is very stable from one exposure to the next and individual PSFLets do not appear to be shifting over such timescales.
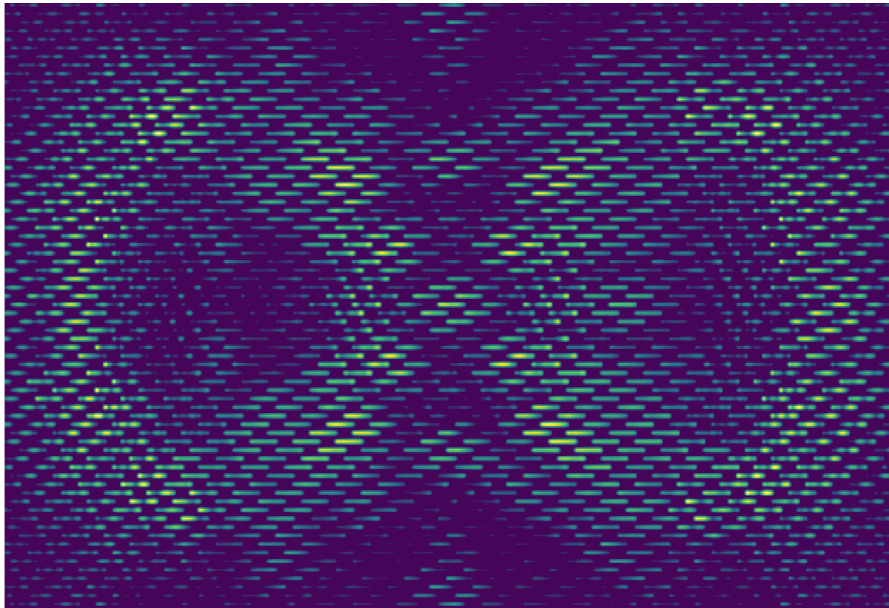

Figure 4. IFS detector map corresponding to the input cubes from Fig. 3. The cube is rotated, rebinned onto the lenslet array slice-by-slice, and the flux within each lenslet multiplies a lenslet PSF which location on the detector depends on the wavelength. For a broadband spectrum, each lenslet thus has an elongated microspectrum. For spatial and spectral Nyquist sampling, the information in this image is the same as the one in the input cube, and the input cube can be reconstructed. This detector map corresponds to a simulation of an observing scenario targeting 47 Uma c, which is visible in the right-hand side of the bowtie, close to the inner working angle. We use the same example throughout the paper, and discuss the details in Section 3.1.

2.3 Stochastic EMCCD model

WFIRST has baselined an e2v Electron-Multiplying CCD (EMCCD) for its flight operations. This type of detectors can be operated in photon-counting mode, which can get rid of the read noise and offers overall signal-to-noise ratio (SNR) improvements. In order to model the EMCCD, we use a step-by-step stochastic approach.[1, 10–13] First, we multiply the exposure time t by the photo-electron arrival rate map p obtained with `crispy`, which takes into account wavelength-dependent quantum efficiency and optical efficiencies. We add a dark current d term and a clock-induced charge (CIC) term c, and apply a Poisson distribution to this map $pt + dt + c$. We feed the obtained map of integer electrons to a gain register model, which uses the gamma function to represent the electron multiplying stage.[11, 13] We add an arbitrary bias b and a zero-mean gaussian process of standard deviation σ to describe the read noise of the electron-multiplying register. We finally apply a threshold at $b + \tau$ (typically $\tau > 5.5\sigma$). This allows us to count individual electrons and eliminate the gain register read noise. This model is approximate as it does not take into account effects such as spurious charge that can occur in the EM stage itself.[11] This algorithm is implemented using Python's `numpy.random` functions in the `crispy` source code. The set of parameters that we use for the flight simulations is shown in Table 1. All the components of the noise model can be turned off in `crispy`'s master parameter file.

In order to retrieve the actual photometry in photon-counting mode, we have to account for several effects. The first is to subtract from the average frame a local background level which is sampled in regions of the image

Table 1. Suggested detector parameter settings.

Parameter	Symbol	Value	Units
Clock-induced charge	c	0.01	$\mathrm{e^-\,pixel^{-1}\,frame^{-1}}$
Dark current	d	2×10^{-4}	$\mathrm{e^-\,pixel^{-1}\,s^{-1}}$
Electron-multiplying gain	g	2500	N/A
Readout bias	b	200	$\mathrm{e^-\,pixel^{-1}\,frame^{-1}}$
Read noise	σ	100	$\mathrm{e^-\,pixel^{-1}\,frame^{-1}}$
Threshold	τ	5.5σ	N/A

with no input flux. The second is to apply a photometric correction caused by coincidence losses at the input of the EM register, which can occur when more than 1 photon is converted into electrons in a single frame. This imposes the frame to be low flux (typically less than 0.1 electron/pixel/frame), in order to keep the coincidence losses from impacting the SNR, which in turns puts constraints on exposure times. A photometry correction for this effect exists[10] and is $I_{\mathrm{corrected}} = -\ln(1 - I_{\mathrm{original}})$, where I_{original} and $I_{\mathrm{corrected}}$ are the images before and after correction, respectively.

A third and final photometric correction occurs, as there is non-zero chance that a photon event might be lower than the threshold applied during photon-counting mode. The likelihood of this happening is $\exp(-\tau/g)$, where g is the EM gain, so we can divide the image by this amount. An example of a noisy map average for a $\sim 27\,\mathrm{h}$ observation is shown in Fig. 5.

The EMCCD suffers from radiation sensitivity in the space environment.[1] High energy particles can damage the silicon lattice and create "traps" which can capture and release electrons at random times during charge transfers to the gain register. As a result, single traps in the regular CCD sensitive area can affect electrons on the entire column, while charge traps that are located in the serial register can affect whole rows. This reduces the effective QE and generates random electron events. Unfortunately, accurately modeling the effects of charge traps is complicated, and also computationally expensive.[14] The process cannot be easily parallelized since each new frame requires knowledge of the state of all the traps from the previous frame.

At this moment, `crispy` is not modeling all of the effects of traps, although the authors are working to increase the speed of existing models.[14] At present, we simplify the problem of traps into an overall reduction in QE, using an empirical model developed by JPL during EMCCD testing. This overall QE reduction is a function of the fraction of the mission time the detector spent in the space environment, where a fraction of 1 corresponds to a 6-year mission at the L2 Lagrange point. This model is applied to the average photo-electron rate p before adding the dark current and CIC contributions.

2.4 IFS extraction

Once a map of the IFS detector has been generated and the detector readout has been simulated, it remains to extract the original datacube from it. In order to achieve this, `crispy` extraction routines are heavily based on the work done on the CHARIS data reduction pipeline.[15] We include two methods for extraction: one that relies on fitting 1-D Gaussian functions to each pixel across a microspectrum ('optimal' extraction[16]); and one that considers a least squares fit of the microspectrum as a sum of quasi-monochromatic PSFLet models (least squares extraction,[15] see the left-hand side of Fig. 6). The two algorithms are comparable in terms of execution speed, but the least squares extraction requires more calibration data and has a few more parameters. It is also the better of the two algorithms when it comes to producing science-grade reduced datacubes.

Both algorithms rely on knowing precisely the centroids of all the PSFLets for a given spectral bin. This can be achieved using monochromatic flatfields to uniformly illuminate the lenslet array, hence create a quasi uniform grid of spots on the IFS detector (as in the left of Fig. 2). The grid can be fitted by low-order 2D polynomials in both the x and y detector coordinates. By cycling through various wavelengths, it is then possible to map exactly where the light is supposed to fall at any wavelength in any microspectrum.

The least squares extraction method uses a high-resolution model of all the PSFLets across the field. This model can either be an analytical function, a set of templates provided by the user, or obtained in the lab
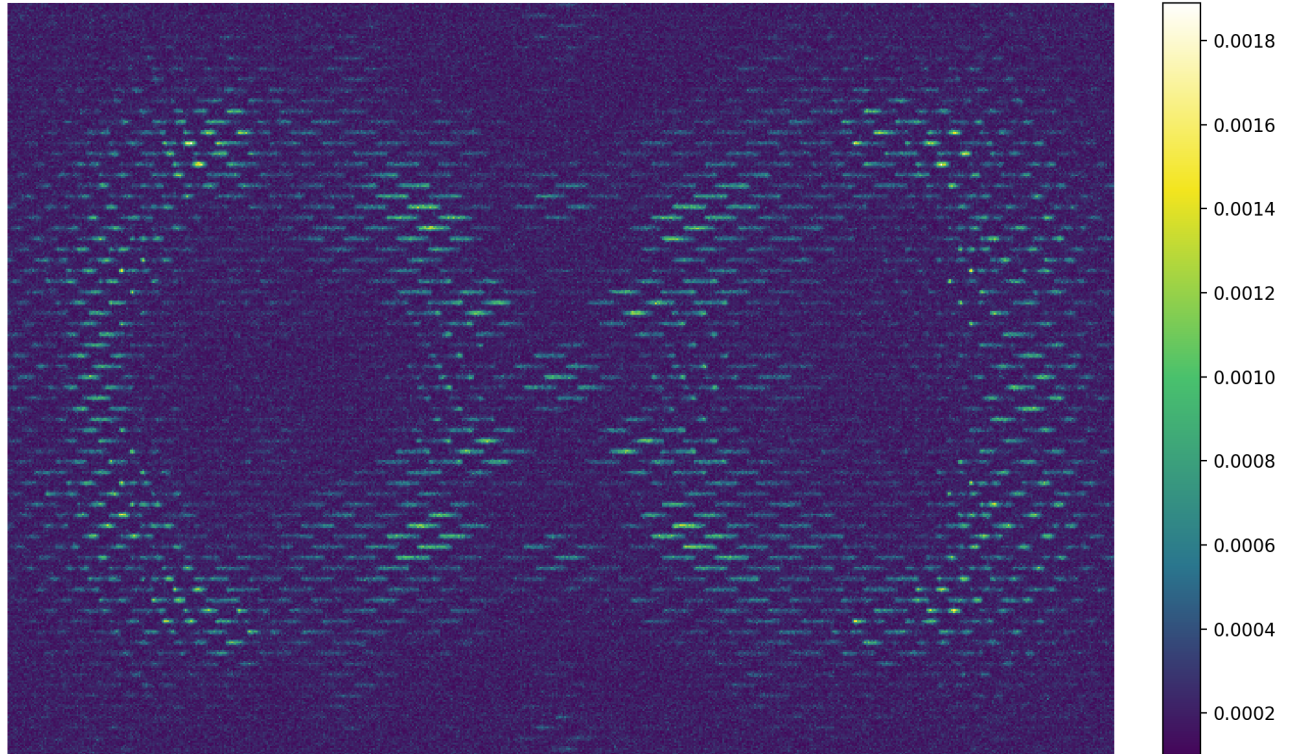
Figure 5. Average IFS detector map in electrons per second corresponding to a 27 hr integration time. This average is determined from 1000 individual 100 s exposures, in photon counting mode. The photometry has been corrected to reflect real electron rates.

by dividing up the detector in several regions and deconvolving PSFLets models in each region.[15] While the latter is implemented in `crispy` and its preliminary implementation on the PISCES testbed has proved to be successful, we plan on using detectors with a fine pixel pitch to calibrate the WFIRST IFS on the ground. The deconvolution method is implemented successfully on the CHARIS IFS, but their detector has 4 times the area and 4 times the number of PSFLets, hence improving the PSFLet sampling diversity and as a result, the PSFLet models. One of the main advantage of the least squares extraction method is that it provides residual maps over the entire detector area. These residual maps can be used for calibration purposes, which will be discussed in a later paper.

`crispy` is also capable of simulating the wavelength calibration step, a critical aspect of WFIRST operations for the IFS. As shown on CHARIS,[17] the PSFLet shapes do not significantly change with time, unlike their location on the detector. As part of their routine observing schedule, single-wavelength updates are used to account for the bulk shift and rotation of the PSFLet grid on the detector. A polynomial fit of the PSFLet grid is done at a single monochromatic wavelength, and the linear and rotational offset from the previous calibration at that wavelength is recorded. These offsets represent the overall translation and rotation of the PSFLet grid as a rigid body, and are thus applied identically to all other wavelengths in the calibration set.

2.5 Software usage and parameters

The publicly released version of `crispy` is tuned to apply for the WFIRST IFS, but can be applied to any other square lenslet array-based IFS. Hexagonal lenslet geometries are a possible future add-on. The software is open source and publicly available on GitHub[†]. A documentation page is associated with the repository, which has examples and tutorials on how to use the software.

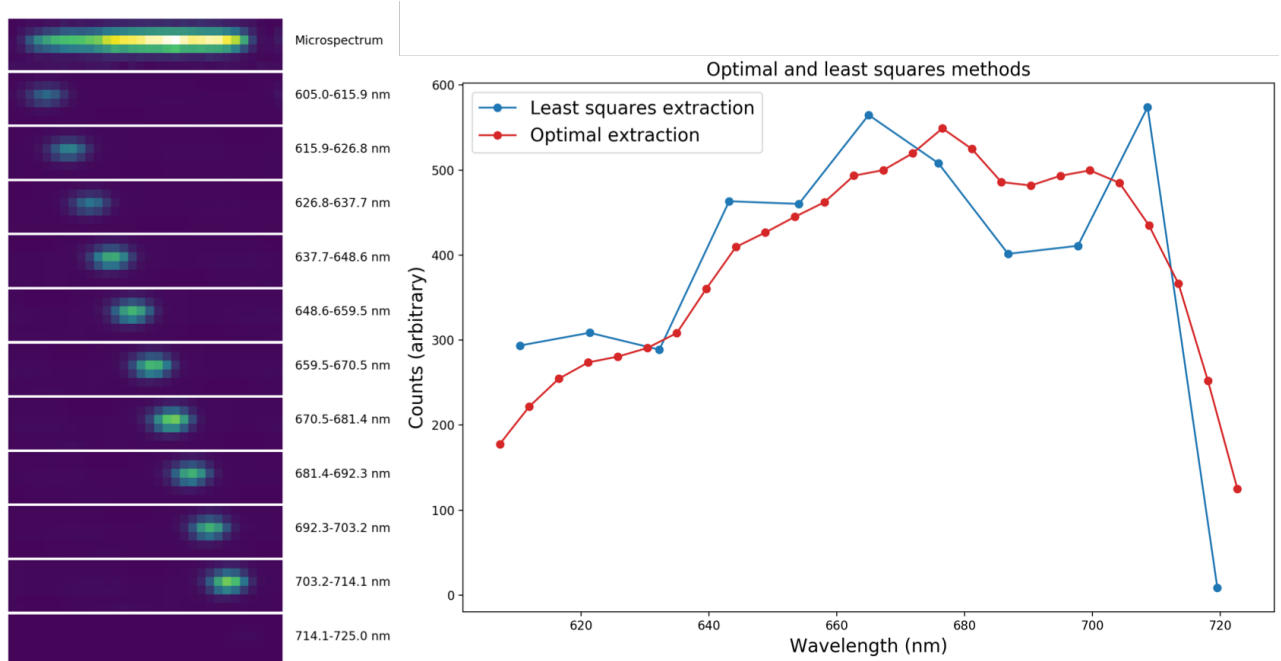[†]`https://github.com/mjrfringes/crispy`

Figure 6. Left: A microspectrum (top) is fitted to a weighted sum of template PSFLets in the least squares extraction method, here shown individually. Each PSFLet template is slightly broadened along the dispersion direction to account for finite-width bins in the recovered spectrum. Right: Comparison of the same microspectrum extracted using the least squares method (blue) and the optimal extraction method (red). In the latter, one-dimensional Gaussian functions are fitted along each pixel column of the microspectrum. In this particular configuration (PISCES), the optimal extraction has 26 bins when the least squares extraction only has 11. Note that the bins in the optimal extraction spectrum are not uncorrelated, unlike in the least squares spectrum, which is why the optimal extraction spectrum looks smoother and appears to be taken at a lower resolving power. In order to plot both spectra on top of each other on the same scale, the native optimal extraction spectrum has been scaled by a factor of 1/4. The counts in the recovered spectrum are arbitrary, as each method would have its own specific flux calibration procedure.

In order to use `crispy`, the user must choose the basic IFS parameters defined in Table 2. These define the overall geometry of the lenslet array and detector, as well as the spectral properties of the IFS. In addition, the detector parameters mentioned in Section 2.3 can be populated, with the ability to turn on and off all sources of noise.

Without any a priori knowledge other than those parameters, `crispy` generates a set of monochromatic flatfields at several user-defined wavelengths in the band. The nominal centroids for `FWHMlam` are determined using an ideal polynomial function with no geometric aberration. The users are also able to manually input their own distortions (e.g. to simulate a distortion that was measured on a testbed). The spots are 2D Gaussian PSFLets (`gaussian` set to `True`). It assumes the resolving power `R` is constant across the spectrum and the detector samples a spectral element with `npixperdlam` pixels for the central wavelength `FWHMlam`. The deviation in pixels along the dispersion direction from the centroid at wavelength `FWHMlam` is set by the following equation:

$$\Delta\mathrm{pix}(\lambda) = \texttt{npixperdlam} \times \texttt{R} \times \ln\frac{\lambda}{\texttt{FWHMlam}}. \tag{1}$$

`crispy` then runs a wavelength calibration routine on the set of PSFLet maps. This produces an array of polynomial coefficients for each wavelength, which fully characterize the PSFLet spot locations in the 2D detector coordinates. `crispy` generates all the required calibration products which are very similar to the ones used by the CHARIS data reduction pipeline. Importantly, it is possible to use noisy images for this step, which allows us to better understand what is the impact of errors in the wavelength calibration phase on final data

Table 2. `crispy` parameters.

Parameter	Default	Description
Lenslet array		
nlens	108	Number of lenslets across array
pitch	174×10^{-6}	Lenslet pitch in meters
interlace	2	Clocking is $\arcsin\left[(\text{interlace}^2 + 1)^{-1/2}\right]$
lensletsampling	0.5	Lenslet size in units of λ/D
lensletlam	660	Wavelength at which sampling is defined (in nm)
Detector		
npix	1024	Number of detector pixels on a side
pixsize	13×10^{-6}	Pixel size in meters
FWHM	2	Pixels across full-width-at-half-maximum of PSFLet
FWHMlam	660	Wavelength at which the FWHM is defined
gaussian	True	If True, uses a Gaussian kernel for PSFLets; if False, a kernel needs to be provided
Spectrograph		
BW	0.18	Fractional bandwidth
R	50	Spectral resolving power
npixperdlam	2	Pixel sampling of individual spectral channel
nchanperspec_lstsq	1	Number of channels per native spectral element

products. During this phase, the user is allowed to tweak the spectral resolution to produce more or less slices in the extracted cube.

Once the calibration step is finished, `crispy` can process input cubes into IFS detector maps, and can extract these detector maps into spectral datacubes. Input cubes need to provide certain FITS header keywords such as the wavelength and the pixel scale, which `crispy` will use to rotate and rebin the input cube to match the lenslet array geometry. The IFS map is then generated by combining monochromatic maps for each wavelength of the input cube.

The detector noise parameters will determine the SNR of the extracted cube. Although `crispy` uses a EMCCD noise model, individual noise contributors can be turned off in order to mimic a regular CCD as well. In the case where an EMCCD model is chosen, the user needs to pay attention to the electron flux at the final detector map which should not exceed $\sim 0.1\,\mathrm{e}^-\,\mathrm{pixel}^{-1}\,\mathrm{frame}^{-1}$ if one wants to avoid coincidence losses.

The extraction algorithm can finally be applied to the noisy detector maps. It will use the calibration products that were generated during the wavelength calibration phase. Using both the optimal extraction and the least squares extraction method, a spectral datacube is generated.

`crispy` is evolving fast and several improvements are planned that will make it easier for users to input their own distortion maps, their own field-dependent PSFLet templates, and facilitate other operational aspects of the software.

3. APPLICATION TO WFIRST CGI-IFS

`crispy` is being developed for WFIRST and the CGI Science Investigation Teams as the primary IFS simulator. Because of its intricacies, it warrants a standalone and focused effort, the development of which has been accelerated by the IFS similarities to existing ground-based IFS instruments such as CHARIS. We have been able to enjoy a productive collaboration with the CHARIS team and were able to apply most of their IFS pipeline for the WFIRST IFS.

`crispy` is intended to be used largely in three main ways at this stage of the WFIRST mission. First, it is used routinely on the PISCES testbed for coronagraph closed-loop control. This is discussed extensively in a companion paper in these proceedings.[7]

Second, `crispy` is intended to be used to simulate realistic data products, similar to the ones that will be produced by the flight instruments. This is discussed in Section 3.1.

Finally, `crispy` is used by the Goddard IFS team to study and improve the IFS design. As it is capable of simulating science observations and can determine quantities such as SNR, it can be used as a good metric to establish tolerances on the IFS and to explore the design parameter space. This is illustrated in Section 3.2.

3.1 CGI observing scenarios

Realistic data products that mimic an observing scenario are helpful to prepare post-processing pipelines, assess observing strategies, and, in the particular case of a coronagraph, understand how much can be gained by post-processing, which can relax the requirements on the instrument's raw contrast goals.
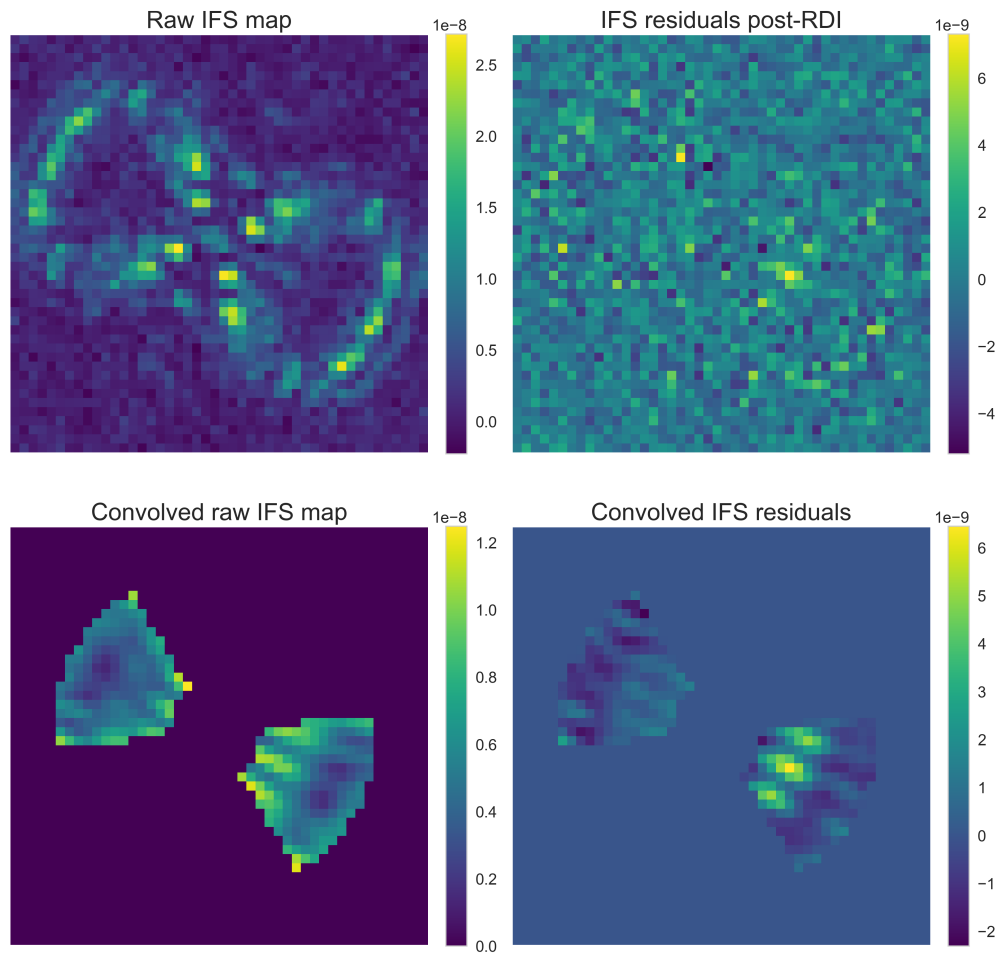

Figure 7. Sample of `crispy` products for an observing scenario. For each of the four panels, only the middle slice (660 nm) is shown. The top left shows the raw IFS map averaged over 1000 individual exposures of 100 s. The planet is barely visible on the right half of the bowtie, close to the inner working angle. The top right corresponds to the residuals after RDI, which uses a nearby reference star to measure the speckle pattern. The bottom left shows the top IFS map convolved with a matched filter that was constructed using an ideal off-axis PSF. The planet is now more visible in this image. The bottom right shows the convolved residuals. At this point the planet is clearly visible many standard deviations above the local noise.

`crispy` can ingest a realistic WFIRST CGI observing scenario time series of spectrally and spatially resolved cubes. The fiducial WFIRST scenario mimics an observation of the giant planet 47 Uma c that was discovered through radial velocity techniques, using β Uma as a close-by reference star. This scenario is labeled "OS5" in the rest of this section. It divides the integration time into chunks of 1000 s (subsequently 1 ks), during which we assume that the speckle field does not change. 30 ks are spent on the reference star, β Uma, and 100 ks are spent on the target star, 47 Uma. Importantly the simulation includes realistic noise expected from the observatory, as well as the low-order wavefront control response.

The module propagates all of the individual cubes into IFS maps representing the photon arrival rate at each discrete time step, both for the reference and target stars. For each of these maps, we select the correct exposure time that ensures less than 0.1 electron/frame in most areas of the frame. Using this exposure time, typically about 10 s for the reference and 100 s for the target, `crispy` computes as many noise realizations as needed to amount to the total exposure time for this time step. For example, if each time step corresponds to 1000 s, then 10 frames of 100 s are simulated per step for the target. All of the frames from all time steps are averaged separately for the reference and target, background-subtracted using non-illuminated parts of the detector, and the photometry is corrected as discussed in Section 2.3.

Once a noisy average is determined for both the reference and the target star, both maps are extracted into noisy IFS cubes. We implement a reference differential imaging technique (RDI) to subtract from the target cube the linear combination of the reference cube that minimizes the least squares residuals within the dark hole. Each slice is treated independently. The residual cube is then convolved by a matched filter cube and normalized to contrast units using an ideal stellar off-axis PSF. The planet's spectrum corresponds to the spectrum of the pixel which matches the planet's location, assumed known in this scenario.

Several by-products of this module are shown in Fig. 7, where one slice of a noisy simulation is shown. In this case, the planet is at the brightest location of the convolved IFS residuals (bottom right panel), and its contrast is the maximum pixel value.
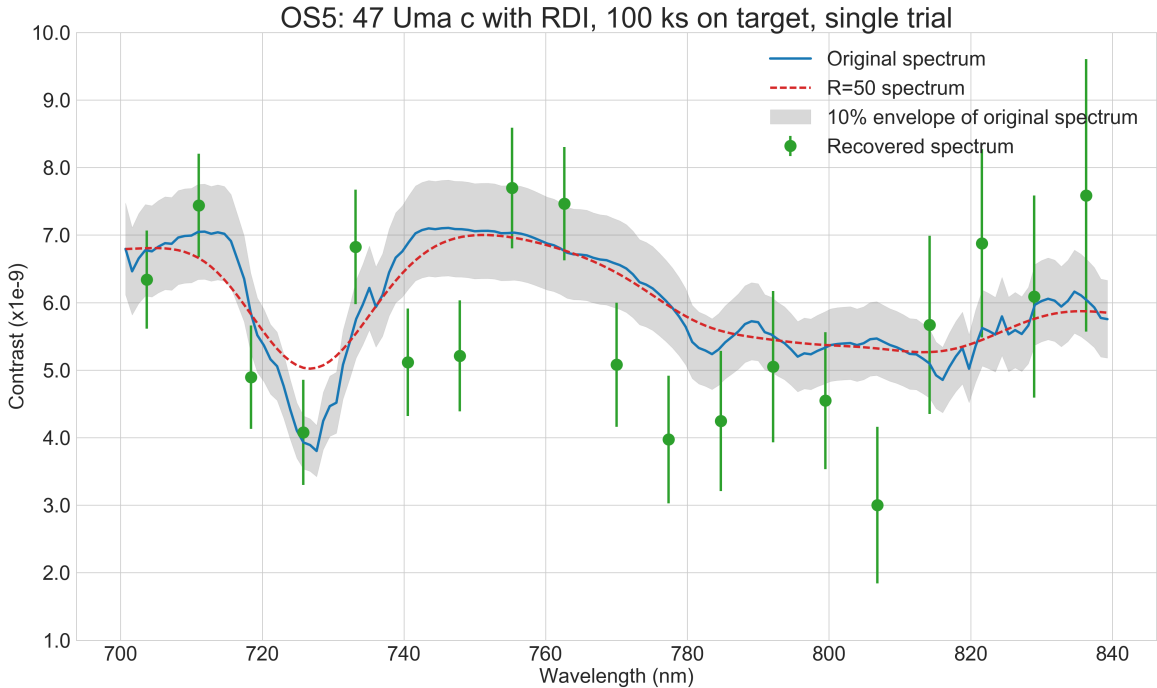

Figure 8. Extracted spectrum[18] from an OS5 trial of 27 h exposure time. The recovered spectrum is shown along with the original and convolved spectrum. The gray area indicates the SNR=10 confidence envelope of the original spectrum for visual reference. If the SNR was 10, the error bars would be the size of this envelope.

While the SNR in the bottom right panel of Fig. 7 appears large, it is not representative of the actual measurement uncertainty. Indeed, when running multiple realizations of the entire OS5 scenario, the variation in the contrast values that we obtained post-RDI varied significantly more than the SNR within a single realization would predict. As a result, we conclude that the SNR within a single OS5 trial is not capturing the uncertainty correctly: for example it is not capturing photon noise from the source. In order to determine an accurate SNR and put correct error bars on the recovered spectra, we choose to run our 100 OS5 trials, and use the standard deviation in the determined contrast for each slice as our error bar. A recovered spectrum[18] for the OS5 scenario is shown in Fig. 8 with representative error bars. The SNR obtained with our method agrees with parametric SNR estimates for the CGI-IFS to within 20%.

3.2 Trade-offs and discussion

An IFS simulator like `crispy` is a good tool to explore design trades, as it allows the use of accurate metrics to assess performance. The sensitivity of the chosen metric to various design parameters can be assessed and used to establish tolerances on the instrument. The simulator also provides a useful guide to adjust the design in order to maximize the chosen metric.

As an example, we show here the result of a simulation concerning the lenslet sampling at the lenslet array. It had been assumed that the lenslet array would sample the incoming beam at the Nyquist sampling ($0.5\lambda/D$) for the shortest wavelength in the IFS, 600 nm, which is the same as saying we will have 2.2 lenslets per λ/D at 660 nm. This is justified mostly by the need to interpolate the IFS cubes between the reference and target star in order to subtract the reference PSF, in the case that they are offset from each other. The concern is that if we sample coarser than Nyquist, any offset would cause issues with the interpolation and would dramatically reduce the potential post-processing gains that can be achieved. This lenslet sampling parameter has far-reaching implications, since it sets the number of lenslets that will be used to sample a planet's PSF, which then spreads the planet's light onto more detector pixels, which will in turn affect our SNR.
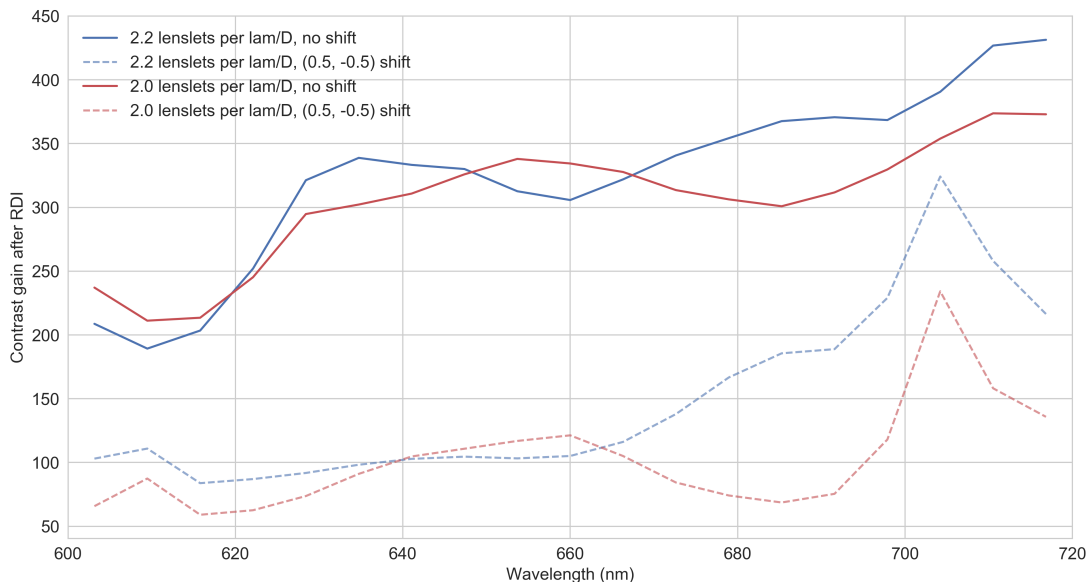

Figure 9. Contrast gain curves showing how much the standard deviation of the convolved speckle is improved by our RDI method. This is done for the case where both reference and target stars are aligned, and when the reference star is misaligned with respect to the target star in the worst possible case (0.5 lenslet in both directions). This is plotted for both 2.2 and 2.0 lenslets per λ/D at 600 nm. Even with the coarser sampling, the post-processing gain is still larger than 50 even at the shortest wavelength, which indicates that it will likely not be a limitation once real noise is taken into account.

`crispy` allows us to model this entire process and quantify the impact of the lenslet sampling on the post-processing gain. With this example, we are trying to determine whether it is possible to use a bit coarser sampling and not degrade post-processing gain too much. We set up the experiment with a noiseless detector, in order to understand what is the maximum possible post-processing gain that can be achieved. We add a known offset to the reference star with respect to the target star on the lenslet array. This is feasible with very little error since the fine sampling of the input cube from JPL's integrated simulation pipeline is typically given at 10 pixels per λ/D, significantly more than the Nyquist sampling. Both the offset reference cube and aligned target cube are propagated through the IFS and extracted back using our IFS extraction tool. The IFS cube of the reference star is then counter-shifted by the correct amount (assumed known), but now it has a 2.2 lenslets per λ/D at 660 nm (instead of 10 when we did the first interpolation), which will introduce interpolation noise. RDI is applied using the countershifted reference IFS cube, and we measure the ratio of the convolved target star residuals with RDI and the convolved target star without RDI (similar to Fig. 7 but without any noise and without any planet in the field of view). We repeat this process for the case of a slightly coarser sampling of 2 lenslets per λ/D at 660 nm.

The wavelength-dependent contrast gains in the noiseless case is shown in Fig. 9 for both sampling cases. In this figure, `lam` is equal to 660 nm. At the finer sampling of 2.2 lenslets per λ/D, the contrast gain is > 200 for the case where both reference and target are aligned on the lenslet array. In the worst interpolation case possible, which corresponds to an interpolation in each direction of the lenslet array by 0.5 lenslet, the contrast gain drops to ~65 for the coarser sampling and ~100 for the finer sampling. While this represents a loss in contrast gain by a factor of ~4, the remaining contrast gain in the case of the coarser sampling is still large, and will not limit our ability to remove the speckles. In any real-world instrument, our maximum achievable contrast gain will likely not exceed 10 or 20, which is the state-of-the-art improvement achievable by ground-based instruments.
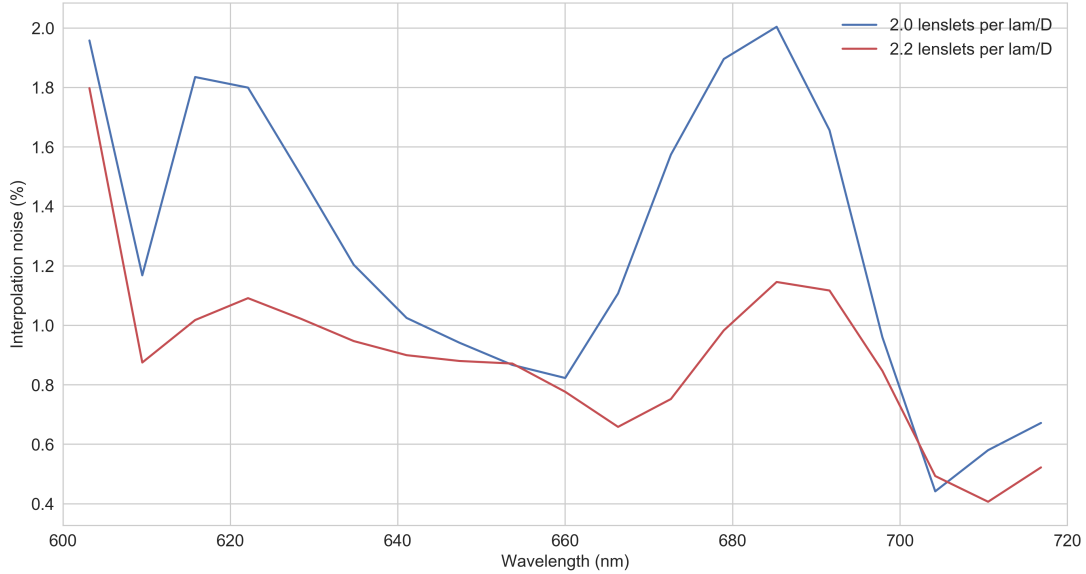

Figure 10. Interpolation noise caused by shifting the reference star by half a lenslet in both directions, which corresponds to the worst possible interpolation case. The noise is calculated by computing the RMS of the difference between the cubes when the aligned reference is propagated, and when the reference is shifted, propagated then countershifted. This is a measure of how much difference there is between the two cubes with and without the shift. Dividing by the RMS of the propagated aligned reference allows us to normalize this quantity. Between the two lenslet sampling values, the interpolation noise is increased by less than a factor of 2, but always remains at less than 2%, which indicate that the maps are very much alike despite the slightly coarser sampling.

The amount of interpolation noise is assessed by computing the RMS of the difference between the cubes when the aligned reference is propagated, and when the reference is shifted, propagated then countershifted. This is a measure of how much difference there is between the two cubes with and without the shift. We divide by the RMS of the aligned reference cube for normalization (Fig. 10).

Thus, this simulation offers strong indication that sampling the lenslet array a 2 lenslets per λ/D at $\lambda = 660$ nm will not limit our ability to use algorithms like RDI. It will however allow for an overall number of detector pixels per planet PSF reduced by $\sim 20\%$, thus improving the SNR of our observation. A more comprehensive study that analyses even coarser sampling is underway to find the optimal parameter.

4. CONCLUSIONS

`crispy` is a versatile IFS simulator and extractor for the WFIRST IFS. It can be used to simulate the IFS sensitivity and performance, study trade-offs, generate realistic data products, and is also routinely used for operations on the PISCES testbed at JPL. The software is open source and publicly available on GitHub[‡].

`crispy` uses a simplified propagation model to construct IFS images instead of propagating the complex field through each lenslet. It uses a stochastic model to represent the noise and response of electron-multiplying CCDs, and implements photon-counting algorithms for optimal performance. Finally, it is founded on the CHARIS data reduction pipeline to extract IFS maps back into spectral datacubes.

We used the software to simulate a realistic observing scenario for the WFIRST coronagraph. We implemented a reference differential imaging technique and recovered accurate planet photometry with SNR that is close to the parametric estimates used by the CGI yield calculation team. We also constructed data products that can be used for assessment and improvement of post-processing techniques for the CGI instrument. We used `crispy` to determine that changing the lenslet sampling on the IFS lenslet array would not undermine our efforts to obtain good post-processing gains. This is the first of multiple trade-off studies that we plan to do in the upcoming months and years for WFIRST.

ACKNOWLEDGMENTS

REFERENCES

1. L. K. Harding, R. T. Demers, M. Hoenk, P. Peddada, B. Nemati, M. Cherng, D. Michaels, L. S. Neat, A. Loc, N. Bush, D. Hall, N. Murray, J. Gow, R. Burgon, A. Holland, A. Reinheimer, P. R. Jorden, and D. Jordan, "Technology advancement of the CCD201-20 EMCCD for the WFIRST coronagraph instrument: sensor characterization and radiation damage," *Journal of Astronomical Telescopes* **2**, pp. 011007–011007, Jan. 2016.
2. N. T. Zimmerman, A. J. E. Riggs, N. J. Kasdin, A. Carlotti, and R. J. Vanderbei, "Shaped pupil Lyot coronagraph designs for WFIRST/AFTA," in *Proceedings of the SPIE*, pp. 96050A–96050A–14, Princeton Univ. (United States), International Society for Optics and Photonics, Sept. 2015.
3. A. Give'On, "A unified formailism for high contrast imaging correction algorithms," *Techniques and Instrumentation for Detection of Exoplanets IV. Edited by Shaklan* **7440**, pp. 74400D–74400D–6, Aug. 2009.
4. A. M. Mandell, Q. Gong, T. D. Groff, M. J. Rizzo, N. T. Zimmerman, P. Saxena, and M. W. McElwain, "Current science requirements and planned implementation for the WFIRST-CGI Integral Field Spectrograph (IFS)," in *These proceedings*, SPIE, 2017.

[‡]`https://github.com/mjrfringes/crispy`

5. P. Saxena, M. J. Rizzo, C. Meija Praga, J. Llop Sayson, Q. Gong, E. Cady, A. M. Mandell, T. D. Groff, and M. W. McElwain, "Commissioning and performance results of the PISCES instrument," in *These proceedings*, SPIE, 2017.

6. M. W. McElwain, A. M. Mandell, Q. Gong, J. Llop-Sayson, T. Brandt, V. J. Chambers, B. Grammer, B. Greeley, G. Hilton, M. D. Perrin, K. R. Stapelfeldt, R. Demers, H. Tang, and E. Cady, "PISCES: an integral field spectrograph technology demonstration for the WFIRST coronagraph," in *Proceedings of the SPIE*, H. A. MacEwen, G. G. Fazio, M. Lystrup, N. Batalha, N. Siegler, and E. C. Tong, eds., pp. 99041A–99041A–18, NASA Goddard Space Flight Ctr. (United States), SPIE, July 2016.

7. T. D. Groff, C. Meija Praga, E. Cady, M. J. Rizzo, A. M. Mandell, Q. Gong, N. T. Zimmerman, P. Saxena, M. W. McElwain, and O. Guyon, "Current science requirements and planned implementation for the WFIRST-CGI Integral Field Spectrograph (IFS)," in *These proceedings*, SPIE, 2017.

8. J. Krist, B. Nemati, and B. Mennesson, "Numerical modeling of the proposed WFIRST-AFTA coronagraphs and their predicted performances," *Journal of Astronomical Telescopes* **2**, pp. 011003–011003, Jan. 2016.

9. J. E. Krist, "PROPER: an optical propagation library for IDL," *Optical Modeling and Performance Predictions III. Edited by Kahan* **6675**, pp. 66750P–66750P–9, Sept. 2007.

10. A. G. Basden, C. A. Haniff, and C. D. Mackay, "Photon counting strategies with low-light-level CCDs," *Monthly Notices of the Royal Astronomical Society* **345**, pp. 985–991, Nov. 2003.

11. M. Hirsch, R. J. Wareham, M. L. Martin-Fernandez, M. P. Hobson, and D. J. Rolfe, "A Stochastic Model for Electron Multiplication Charge-Coupled Devices - From Theory to Practice," *PLoS ONE* **8**, pp. e53671–, Jan. 2013.

12. T. Plakhotnik, A. Chennu, and A. V. Zvyagin, "Statistics of single-electron signals in electron-multiplying charge-coupled devices," *IEEE Transactions on Electron Devices* **53**(4), pp. 618–622, 2006.

13. O. Daigle, J.-L. Gach, C. Guillaume, S. Lessard, C. Carignan, and S. Blais-Ouellette, "CCCP: a CCD controller for counting photons," *Ground-based and Airborne Instrumentation for Astronomy II. Edited by McLean* **7014**, pp. 70146L–70146L–10, July 2008.

14. B. Nemati, R. Effinger, R. Demers, L. Harding, P. Morrissey, N. Bush, D. Hall, and J. Skottfelt, "The effect of radiation-induced traps on the WFIRST coronagraph detectors," in *Proceedings of the SPIE*, pp. 99150M–99150M–12, Jet Propulsion Lab. (United States), International Society for Optics and Photonics, Aug. 2016.

15. T. D. Brandt, M. Rizzo, T. Groff, J. Chilcote, J. P. Greco, N. J. Kasdin, M. A. Limbach, M. Galvin, C. Loomis, G. Knapp, M. W. McElwain, N. Jovanovic, T. Currie, K. Mede, M. Tamura, N. Takato, and M. Hayashi, "Data Reduction Pipeline for the CHARIS Integral-Field Spectrograph," *arXiv.org*, June 2017.

16. K. Horne, "An optimal extraction algorithm for CCD spectroscopy," *Astronomical Society of the Pacific* **98**, pp. 609–617, June 1986.

17. T. D. Groff, J. Chilcote, T. D. Brandt, M. Galvin, C. Loomis, M. J. Rizzo, G. Knapp, O. Guyon, N. Jovanovic, J. Lozi, N. Takato, and M. Hayashi, "First light of the CHARIS high-contrast integral-field spectrograph," in *These proceedings*, SPIE, 2017.

18. K. L. Cahoy, M. S. Marley, and J. J. Fortney, "Exoplanet Albedo Spectra and Colors as a Function of Planet Phase, Separation, and Metallicity," *The Astrophysical Journal* **724**, pp. 189–214, Nov. 2010.