The Astrophysical Journal, 876:78 (15pp), 2019 May 1 https://doi.org/10.3847/1538-4357/ab16e5

Measuring Star Formation Histories, Distances, and Metallicities with Pixel Color–Magnitude Diagrams. I. Model Definition and Mock Tests

B. A. Cook[1], Charlie Conroy[1], Pieter van Dokkum[2], and Joshua S. Speagle[1]
[1] Center for Astrophysics | Harvard & Smithsonian, 60 Garden St., Cambridge, MA 02138, USA; bcook@cfa.harvard.edu
[2] Astronomy Department, Yale University, 52 Hillhouse Ave, New Haven, CT 06511, USA
Received 2019 February 28; revised 2019 March 28; accepted 2019 April 6; published 2019 May 7

Abstract

We present a comprehensive study of the applications of the pixel color–magnitude diagram (pCMD) technique for measuring star formation histories (SFHs) and other stellar population parameters of galaxies, and we demonstrate that the technique can also constrain distances. SFHs have previously been measured through either the modeling of resolved-star CMDs or of integrated-light spectral energy distributions, yet neither approach can easily be applied to galaxies in the "semi-resolved regime." The pCMD technique has previously been shown to have the potential to measure stellar populations and SFHs in semi-resolved galaxies. Here we present Pixel Color–Magnitude Diagrams with Python (`PCMDPy`), a graphics processing unit (GPU)-accelerated package that makes significant computational improvements to the original code and includes more realistic physical models. These advances include the simultaneous fitting of distance, modeling a Gaussian metallicity distribution function, and an observationally motivated dust model. GPU acceleration allows these more realistic models to be fit roughly $7\times$ faster than the simpler models in the original code. We present results from a suite of mock tests, showing that with proper model assumptions, the code can simultaneously recover SFH, [Fe/H], distance, and dust extinction. Our results suggest the code, applied to observations with *Hubble Space Telescope*-like resolution, should constrain these properties with high precision within 10 Mpc and can be applied to systems out to as far as 100 Mpc. pCMDs open a new window to studying the stellar populations of many galaxies that cannot be readily studied through other means.

Key words: galaxies: photometry – galaxies: stellar content – techniques: photometric

1. Introduction

The mass build-up of galaxies is sensitive to many physical processes. Mergers, active galactic nucleus activity, supernova feedback, and the accretion of pristine gas from cosmological filaments will all affect a galaxy's star formation history (SFH) and chemical enrichment, and it is a major goal of modern astrophysics to constrain their relative impacts. Hydrodynamical simulations (e.g., Hopkins et al. 2014; Vogelsberger et al. 2014) are one powerful way to study the evolution of galaxies, but their results require reliable observational constraints on the diversity of SFHs and stellar populations observed in the universe.

To date, measurements of stellar populations and SFHs largely rely on two distinct techniques, each with limitations on the systems they can be applied to and the robustness of the results. These approaches can be considered in a single framework through considering the typical number of stars per resolution element, $N_{\rm pix}$ (van Dokkum & Conroy 2014; Conroy & van Dokkum 2016).

The first technique, resolved-star photometry, typically requires $N_{\rm pix} \lesssim \mathcal{O}(10^{-1})$ in order to fully resolve each individual star in a system. The fluxes of these stars in at least two bands are converted into a color–magnitude diagram (CMD), and the stellar populations are derived from fits to stellar evolution models (e.g., Dolphin 2002; Weisz et al. 2011; Lewis et al. 2015; Williams et al. 2015). As long as stars down to the oldest main-sequence turnoff are resolved, this method recovers highly precise and robust measurements of the SFH and is considered the gold standard.

In practice, the number of systems where the oldest main-sequence turnoff can be resolved is small, even with the resolution of *Hubble Space Telescope* (*HST*)'s Advanced Camera for Surveys (ACS). Several dozen dwarf galaxies in the Local Group have sufficiently low stellar densities to allow well-measured SFHs (e.g., Weisz et al. 2011, 2014). But even our nearest massive neighbor, M31, is so crowded that only rare, massive main-sequence stars (Lewis et al. 2015) can be individually resolved in most fields, limiting SFH recovery to relatively recent star formation. In the inner regions of M31, with $N_{\rm pix} \sim \mathcal{O}(10^{1})$, SFH can be measured using the resolved red giant branch or red clump stars, but the results are subject to significant systematics due to disagreement between models of these phases of stellar evolution (Williams et al. 2017), with the oldest ages the most uncertain. The inner bulge of M31 is so crowded that not even red giant branch (RGB) stars can be resolved with *HST*, making any SFH measurement impossible with this technique.

The second well-established method, spectral energy distribution (SED) modeling, analyzes the integrated light of all stars in a (typically entirely unresolved) galaxy. The broadband photometry and spectra are modeled using stellar population synthesis techniques to recover stellar populations and SFHs (e.g., Walcher et al. 2011; Conroy 2013). These methods typically assume a fully populated initial mass function (IMF), ignoring Poisson fluctuations of rare, evolved stars. Such a regime requires $N_{\rm pix} \gtrsim \mathcal{O}(10^{6})$, usually a valid approximation for distant, unresolved galaxies.

Yet even if such fluctuations can be ignored, the light from the oldest, lowest-mass main-sequence stars is usually dwarfed by the light of rare, evolved stars, an effect known as "outshining," which can lead to an underestimate of the total stellar mass and oldest ages of star formation (Maraston et al. 2010; Pforr et al. 2012; Sorba & Sawicki 2015). The SFHs

recovered from SED modeling are also highly dependent on the underlying stellar evolution models (e.g., Conroy 2013; Hunt et al. 2019), with systematic uncertainties much larger than for resolved-star SFHs. Carnall et al. (2019) and Leja et al. (2019) also study the prior bias introduced from assuming various functional forms for the SFH.

In between the realm of resolved stars and integrated light, there is a substantial volume of the universe in the so-called *semi-resolved regime* (van Dokkum & Conroy 2014; Conroy & van Dokkum 2016). This regime can be defined loosely as $\mathcal{O}(10^1) \lesssim N_{\rm pix} \lesssim \mathcal{O}(10^6)$, where stars cannot be individually resolved because of crowding, but surface-brightness fluctuations due to Poisson sampling of rare, bright stars in each pixel cannot be ignored. This semi-resolved regime extends from $\sim$1 Mpc out to nearly 100 Mpc for massive galaxies observed with *HST*.

The surface-brightness fluctuation (SBF; Tonry & Schneider 1988) distance technique is one example of a method for studying galaxies in the semi-resolved regime. In this approach, distances to a galaxy are estimated by measuring the scale of the pixel-to-pixel surface-brightness fluctuations, studied in Fourier space in order to isolate the instrumental point-spread function (PSF). The underlying stellar populations are largely treated as nuisance parameters and accounted for through calibration. Other examples include the "disintegrated" light analysis of stellar halos (Mould 2012), fluctuation spectroscopy (van Dokkum & Conroy 2014), and pixel-level time variability (Conroy et al. 2015).

A third technique for measuring SFHs and stellar populations, specifically applicable to semi-resolved galaxies, was introduced in Conroy & van Dokkum (2016). The pixel color–magnitude diagram (pCMD) method measures the fluxes in multiple (typically two) photometric bands within every pixel in an image, plotting the resulting magnitudes in a CMD. The surface-brightness fluctuations across the image represent real variations in the number of rare, bright stars at each pixel. Therefore, the distribution of these pixel fluxes (the pCMD) holds important information about the underlying stellar populations. Critically, as we show in this work, pCMD distributions show distinct sensitivities to many key physical parameters, including metallicity, SFH, distance, and dust content. With a proper accounting of important observational artifacts, including PSF convolution, sky subtraction, and shot noise, these key parameters can be inferred through a forward-modeling procedure.

The pCMD technique can therefore be considered a generalization of SBF, as in this work we demonstrate that both distances and stellar populations can be simultaneously measured using the fluctuations in multiple bands of well-aligned photometry. SBF is still an important rung on the distance ladder to this day, used to derive distances to nearby dwarf galaxies (Cohen et al. 2018; Carlsten et al. 2019) and massive galaxies alike (Greene et al. 2019). SBF also remains the best distance estimator to the host galaxy of GW170817, the first binary neutron star merger detected in gravitational waves (Cantiello et al. 2018). Modeling the pCMDs of systems such as these offers the prospect of constraining (rather than assuming) their stellar populations, while more robustly accounting for their uncertainties in deriving distances.

This work presents the first results from Pixel Color–Magnitude Diagrams with Python (`PCMDPy`), an open-source package developed to fully implement the pCMD fitting method and allow for modeling pCMDs with more realistic, flexible physical models. The package extends significantly the original framework outlined in Conroy & van Dokkum (2016), allowing for a variety of metallicity, dust extinction, and SFH models, and for the first time extends the pCMD method to simultaneously fit for galactic distances. The package is written in Python and hosted on GitHub,[3] and posterior estimation is implemented with the dynamic nested sampling code `dynesty` (J. S. Speagle 2019, in preparation).[4] The computationally challenging simulation procedure has been accelerated using graphics processing units (GPUs), allowing for more accurate and complex models to be generated more rapidly and for fits to converge in less time.

This paper is outlined as follows. In Section 2 we provide an outline of the `PCMDPy` code. In Section 3 we describe a suite of mock tests where simulated pCMDs generated with the code are then fit to study the code's constraining power, and we conclude the paper in Section 4.

2. Pixel Color–Magnitude Diagrams with Python (`PCMDPy`)

We describe here the procedure for generating and modeling a pCMD. Section 2.1 discusses the general approach and primary assumptions made, while Section 2.2 details the physical models implemented in `PCMDPy`, including metallicity, SFH, distance, and dust extinction. Section 2.3 demonstrates that each free parameter in these models has a unique effect on pCMD distributions. In Section 2.4 we describe the computational architecture of the code. Section 2.5 describes our likelihood model for comparing pCMDs and the nested sampling approach for fitting model posteriors over these free parameters.

2.1. Overview of the pCMD Technique

A pCMD (Conroy & van Dokkum 2016) represents the distribution in pixel-by-pixel photometry of a galaxy or galactic region, projected onto magnitude space. A pCMD is generated by measuring the flux in each pixel of two bands of photometric images and converting those fluxes to apparent magnitudes, representing their distribution in color–magnitude space as a Hess diagram. When doing so, we are necessarily discarding any spatial correlations between the pixels. This contrasts with the SBF method, where the Fourier transform preserves spatial information and can therefore isolate the effect of the PSF. The distribution of the pCMD is thus shaped both by the physical properties (stellar populations) of the stars residing in the image and by the dust extinction and the optical properties of the telescope, such as the PSF and photometric noise.

The pCMD technique endeavors to constrain the underlying stellar populations through a forward-modeling approach: given a model for stellar photometry and our knowledge of the telescope's optical properties, we can create a simulated pCMD derived from a specified set of populations and compare it to real data. This procedure is summarized visually in Figure 1. Comparing simulated pCMDs over a range of stellar populations allows us to infer the most likely populations to have generated in the data, as long as our stellar and observational models are reasonable approximations of the truth.

[3] https://github.com/bacook17/pcmdpy; final API and documentation are still in the development stages.
[4] https://github.com/joshspeagle/dynesty

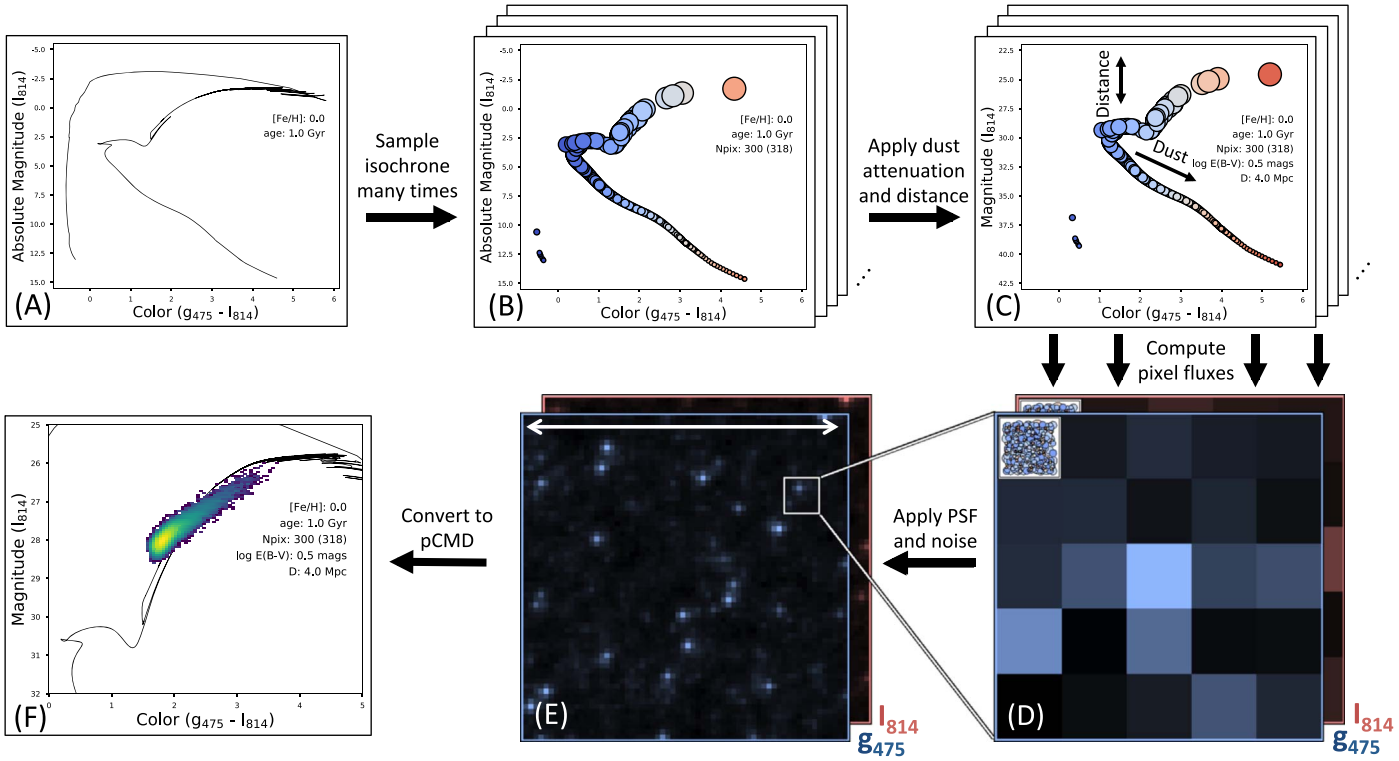

Figure 1. Overview of the pCMD modeling procedure. (A) Metallicity and SFH models are chosen (for demonstration purposes, a single stellar population and corresponding isochrones are drawn from MESA Isochrones & Stellar Tracks (MIST; Choi et al. 2016). (B) Stars are randomly sampled from the isochrones for each pixel in the simulated image, with the mean number of stars in a pixel equal to $N_{\rm pix}$. (C) The fluxes of each star are reddened by dust extinction and adjusted by the distance modulus of the model. (D) The simulated images are generated, with each pixel flux equal to the sum of fluxes from all stars residing in that pixel, as shown in the top left pixel. Here, images in two filters are shown. (E) The simulated images are convolved with the PSF models, and sky and shot noise are added. (F) The pixel color–magnitude diagram is computed by converting pixel fluxes to magnitudes. The original isochrone track is shown for reference.

In practice, the possible combinations of stars residing in a given image are limitless, requiring us to make simplifying assumptions for the problem to be tractable. We adopt a probabilistic approach: we assume there is a global underlying distribution of stars within the image, from which each pixel represents a unique random realization. Therefore, we care only about the overall distribution of stellar populations and pixel magnitudes (the pCMD), not about matching each specific pixel to a model of the stars it contains.

Specifically, we assume that the number of stars in a pixel is drawn from a Poisson distribution, with the mean number of stars constant across the image and equal to the free parameter $N_{\rm pix}$. Therefore, the surface-brightness fluctuations across pixels are a true signal, representing the Poisson noise in stars per pixel, the magnitude of which is determined by $N_{\rm pix}$ and the stellar populations. Thus, $N_{\rm pix}$ can be estimated from the column density $N_{\rm col}$ of stars, the observed spatial resolution θ, and the distance d to the source and is approximately

$$N_{\rm pix} \approx 6 \frac{\rm stars}{\rm pixel} \left(\frac{N_{\rm col}}{10^3\ {\rm pc}^{-2}} \right) \left(\frac{\theta}{0\farcs05} \right)^2 \left(\frac{d}{1\ {\rm Mpc}} \right)^2. \quad (1)$$

The representative values above assume a galaxy with average stellar density of one star per cubic parsec and a thickness of 1 kpc ($N_{\rm col} = 10^3\ {\rm pc}^{-2}$) and observations with the spatial resolution of *HST*-ACS.

To determine the properties of those $\approx N_{\rm pix}$ stars in each pixel, we assume a model for the distribution of their ages (the SFH model), metallicities (given as iron abundance, [Fe/H]), and initial masses (an IMF). This is shown in panel (A) of Figure 1.

In our approach, we divide the age–[Fe/H]–mass space into discrete bins, assigning each point a weight according to the models above and normalizing the weights to equal $N_{\rm pix}$. For every pixel in the simulated image (of size $N_{\rm im} \times N_{\rm im}$ pixels), we randomly draw the number of stars in each bin according to a Poisson distribution with the given weights.[5] We then derive the absolute magnitudes of each star in the observed filters from stellar evolution models (isochrones), although this could also be implemented with well-calibrated observational catalogs. This is shown in panel (B) of Figure 1.

The intrinsic fluxes of each star are attenuated by dust extinction, potentially by both foreground (Milky Way) and source (host galaxy) dust, requiring reddening curves and an assumed model for the dust abundance. The stellar magnitudes are converted to fluxes according to the distance to the host galaxy and are summed into their respective pixels. These steps are panels (C) and (D) of Figure 1.

Finally, the telescope and instrumental signatures must be accounted for (panel (E) of Figure 1). The raw images are convolved with models for the PSF in each filter.[6] Sky flux can be added at this stage. Poisson shot noise is added, assuming the data are in electrons, such that noise $= \sqrt{\rm counts}$. The

[5] Given many Poisson-distributed numbers n_i with weight parameters λ_i, the sum $N = \sum_i n_i$ is also Poisson distributed, with weight parameter $\lambda = \sum_i \lambda_i$.

[6] The PSF convolution breaks the assumption that each pixel represents an independent draw from an underlying distribution, because the fluxes in neighboring pixels are highly correlated. This makes writing down an exact likelihood or computing it using probabilistic programming intractable.

Table 1
Modeling a pCMD Requires an Assumed Physical Model for Metallicity, Star Formation History, Dust Extinction, and Distance

No.	Model Name	Free Parameters	Hyperparameters
	Metallicity Models		
M1	Single [Fe/H]	[Fe/H]	
M2	Gaussian MDF (Fixed Width)	[Fe/H]	$\sigma_{\rm [Fe/H]}$
M3	Gaussian MDF	[Fe/H], $\sigma_{\rm [Fe/H]}$	
	Star Formation History Models		
S1	Single Stellar Population	log $N_{\rm pix}$, logage	
S2	Constant SFR	log $N_{\rm pix}$	
S3	Tau Model	log $N_{\rm pix}$, τ	
S4	Delayed-Tau Model	log $N_{\rm pix}$, τ	
S5	Nonparametric SFH	log $\rm SFH_0$, log $\rm SFH_1$,⋯	$N_{\rm SFH}$, Bin Edges
	Dust Extinction Models		
E1	Fixed dust screen	log $E(B-V)$	$F_{\rm dust}$
E2	Log-normal Dust Screen (Fixed Width)	log $E(B-V)$	$F_{\rm dust}$, $\sigma_{\rm dust}$
E3	Log-normal Dust Screen	log $E(B-V)$, $\sigma_{\rm dust}$	$F_{\rm dust}$
	Distance Models		
D1	Fixed Distance		μ_d
D2	Variable Distance	μ_d	

Note. The physical model implementations in `PCMDPy` and their parameters are listed here. Free parameters are fit to data, while hyperparameters are set prior to fitting.

resulting pCMD is computed by converting the pixel counts to magnitudes (panel (F) of Figure 1).

We additionally note that it is possible to both measure and simulate pCMDs in any arbitrary number of photometric bands. In this work we only consider the case of two filters, primarily due to the challenge of comparing pCMD distributions in more than two dimensions.

2.2. *PCMDPy* Model Choices

The `PCMDPy` physical model for a region contains four primary model components: a metallicity model, an SFH model, a dust extinction model, and a distance model. When fitting a pCMD, the free parameters of the fit all correspond to one of these four components, and we assume flat priors over each. Each component (with the exception of distance) could be modeled in a variety of complex ways, and we detail here the implementations available in `PCMDPy`. The functional forms included here may not necessarily be adequate representations of the true shapes of the distributions of interest in particular galaxies, but they are designed to provide reasonable approximations. In Section 3.4, we discuss the effects of incorrect model assumptions.

All of these physical model components are summarized in Table 1. All other parameters or constants assumed in the model are considered hyperparameters and are held fixed throughout the fit. These are summarized in Table 2.

Three metallicity models are available, which specify the metallicity distribution function (MDF, in terms of [Fe/H]) of the stars in each pixel:

Single [Fe/H]: all stars have the same metallicity, equal to the single free parameter [Fe/H].
Gaussian MDF (fixed width): stellar metallicities are drawn from a Gaussian distribution, with one free parameter corresponding to the mean ([Fe/H]). The standard deviation ($\sigma_{\rm [Fe/H]}$) is a hyperparameter and is held fixed.
Gaussian MDF: as in M2, but $\sigma_{\rm [Fe/H]}$ is a second free parameter.

We adopt the nonrotating isochrones of the MIST project, v1.2 (Choi et al. 2016). The metallicities provided in the MIST isochrones are discretized to a grid with spacing 0.25 dex. For model M1, we interpolate the isochrones between these grid points to recover metallicities not on the grid. For models M2 and M3, we use only the metallicities on the grid and weight their abundances by the specified Gaussian distribution. Grid points with weights less than 1% are removed, resulting in 5 to 10 metallicity points for most Gaussian models.

Five SFH models are implemented, which specify the distribution of stars as a function of age. The number of stars per pixel $N_{\rm pix}$ is either an explicit free parameter of the model or computed as the sum of the SFH.

Single stellar population (SSP): all stars are the same age. The two free parameters are the age of stars (in log years) and $N_{\rm pix}$.
Constant star formation rate (SFR): assumes constant SFR at all times. The one free parameter is $N_{\rm pix}$, to which the total SFH sums.
Tau SFH: an exponentially falling SFR (SFR $\propto e^{-t/\tau}$). The two free parameters are τ (in Gyr) and $N_{\rm pix}$.
Delayed-tau SFH: a linearly rising SFR followed by an exponential falloff (SFR $\propto te^{-t/\tau}$). The two free parameters are τ (in Gyr) and $N_{\rm pix}$.
Nonparametric SFH: the SFH is binned into several (by default five) independent bins, within which the SFR is constant. The free parameters are $\{\log \rm SFH_i\}$, the logarithm of total star formation in each bin, in units of stars per pixel. The number and edges of the bins are hyperparameters.

With the exception of model S1, stellar ages are discretized using a grid of ages. By default, the grid uses 21 equally spaced bins in log age from 1 Myr to 14 Gyr, but this is a hyperparameter $N_{\rm age}$ that can be adjusted. The age points are taken as the midpoints of each bin ($10^{6.1}$, $10^{6.3}$, …, $10^{10.1}$ yr), and the SFR is assumed to be constant within the bin.

When simulating complex models represented by many metallicity and age points (ex: M2+S3), we downsample the MIST isochrones in mass by a factor of five (a hyperparameter). This downsampling factor improves computation time dramatically while not significantly affecting the resulting pCMDs, as confirmed through internal tests.

In the current implementation of `PCMDPy`, SFH and metal abundance are modeled independently: the metallicity of a star does not depend on its age. In reality, the abundances of stars are known to evolve with age as previous generations enrich the interstellar matter (ISM) out of which stars form. Future work could model both jointly, but this is outside the scope of the current work.

Table 2
Global Hyperparameters of the PCMDPy Modeling Procedure

Parameter Name (Default)	Number of Parameters	Description
Filters	$N_{\rm bands}$	Specify which observational filters to simulate. Includes zero points and PSF models.
$N_{\rm im}$ (512)	1	Size of simulated image plane ($N_{\rm im}$ pix$\times$ $N_{\rm im}$ pix)
$N_{\rm age}$(21)	1	Number of isochrones to draw from in an SFH
IMF (Salpeter)	1	Stellar initial mass function
Sky Level (0)	$N_{\rm bands}$	Level of background sky noise to add to each band
Exposure Time	$N_{\rm bands}$	Exposure time of each image (important for modeling shot noise)
Downsampling (5)	1	Factor to downsample isochrone points
Hess Binning (0.05, 0.05)	2	Width of Hess bins used to compute log-likelihood

Note. These parameters remain fixed throughout fitting. This does not include parameters of the sampling algorithm (see dynesty documentation).

We adopt a Salpeter IMF (Salpeter 1955) by default, but a Kroupa IMF (Kroupa 2001) is also implemented in PCMDPy. In addition to determining the distribution of stars by mass, the IMF also determines the conversion from $N_{\rm pix}$ to $M_{\rm pix}$, the stellar mass formed in each pixel. The total stellar mass in the image, $M_\star$, requires a solution for SFH to account for stellar mass loss from post-MS stellar evolution.

Three dust extinction models are implemented, which determine the dust extinction in units of the reddening parameter, $\log E(B-V)$. We adopt the framework of Dalcanton et al. (2015), who studied dust extinction in M31. See their Section 3 for details.

Constant dust screen: all pixels have a constant amount of extinction, equal to the one free parameter: $\log E(B-V)$.
Log-normal dust screen (fixed width): the dust extinction in each pixel is drawn from a log-normal distribution, with one free parameter: the median extinction $\log E(B-V)$. The dimensionless width parameter $\sigma_{\rm dust}$ is a hyperparameter and is held fixed.
Log-normal dust screen: as in E2, except $\sigma_{\rm dust}$ is an additional free parameter.

Each dust model assumes a single thin screen of dust. The geometry of the screen is specified by a hyperparameter $F_{\rm dust}$, which determines the fraction of stars that are reddened by dust. Here, $F_{\rm dust}=1.0$ represents a foreground screen of dust (all stars are reddened), while our default choice of $F_{\rm dust}=0.5$ corresponds to a midplane disk of dust, with half the stars reddened and half unobscured. This model assumes that in practical applications, any foreground dust from the Milky Way can be accounted for in data reduction. Future work could extend to more complex dust geometries, as preliminary tests indicate neither of these models may be sufficient to model the complex and dense dust-lane structures found in disk galaxies at the scale of interest. We convert $\log E(B-V)$ to magnitudes of extinction in each optical band using the $R_V=3.1$ reddening law from Schlafly & Finkbeiner (2011, their Table 6).

The final physical model component is distance. We implement two distance models, simply representing whether or not distance (in units of distance modulus, μ_d) is included in the fit:

Distance fixed: distance is assumed known; μ_d is a fixed hyperparameter.
Distance free: μ_d is a free parameter.

A physical model is specified with one of each of these model components. We will occasionally refer to the entire model used to simulate a pCMD using the alphanumeric codes above. For example, our "fiducial-τ" model (Section 3.1) is described as M2+S3+E2+D2. This translates to a Gaussian MDF (with fixed $\sigma_{\rm [Fe/H]}$), a tau SFH, a log-normal dust screen (with fixed $\sigma_{\rm dust}$), and distance free.

A major hyperparameter of the fitting technique is $N_{\rm im}$, the 1D size of the simulated image plane (the total number of pixels in the simulated pCMD is therefore ${N_{\rm im}}^2$). Note that $N_{\rm im}$ does not have to match the size of the data being fit, as we can compare the relative number of pixels in the pCMD. Larger simulated images are more computationally expensive but reduce the inherent stochasticity of the likelihoods (see Section 2.5) by providing more samples of the rare fluctuations in surface brightness. We choose a default image size of $N_{\rm im}=512$, which we find to be the optimal size for reducing stochasticity while allowing for convergent fits in a reasonable time.

We must match the observational conditions under which the data were taken as closely as possible, requiring several well-calibrated hyperparameters representing each observed filter. This includes the exposure time in each filter, the photometric zero points, and a model for the PSF. We have specifically modeled observations with the *HST*-ACS camera, but the model can be generally applied to ground-based observations as well. The exposure time is taken from the FITS header of the imaging data. Zero points are computed using the PySynphot (Lim et al. 2015) package, as detailed in the ACS Handbook.[7]

PSF models are taken from the Tiny Tim (Krist et al. 2011) web interface.[8] To simulate subpixel PSF effects, we subdivide each image into a 4×4 grid and apply to each a different PSF convolution, shifted by fractions of a pixel. Tests showed our approach is a reasonable approximation to the effects of truly subsampling each pixel and applying a subpixel PSF model, and it is substantially faster.

2.3. Sensitivity of pCMDs to Model Parameters

In Figure 2, we show that the detailed structures of pCMDs are sensitive to each of the physical parameters of interest, in ways that allow constraints on each from only two bands of semi-resolved photometry. In each panel, we show a simulated pCMD for a "baseline model" in gray, with a comparison for a model with one physical parameter changed superimposed in black. The baseline model has a single-[Fe/H] metallicity model (M1, $[{\rm Fe/H}]=-0.5$), a tau SFH (S3, $N_{\rm pix}=10^3$, $\tau=3$ Gyr), a constant dust screen (D1, $\log E(B-V)=-0.5$), and a distance of 1 Mpc ($\mu_d=25$).

[7] http://www.stsci.edu/hst/acs/analysis/zeropoints
[8] http://www.stsci.edu/hst/observatory/focus/TinyTim

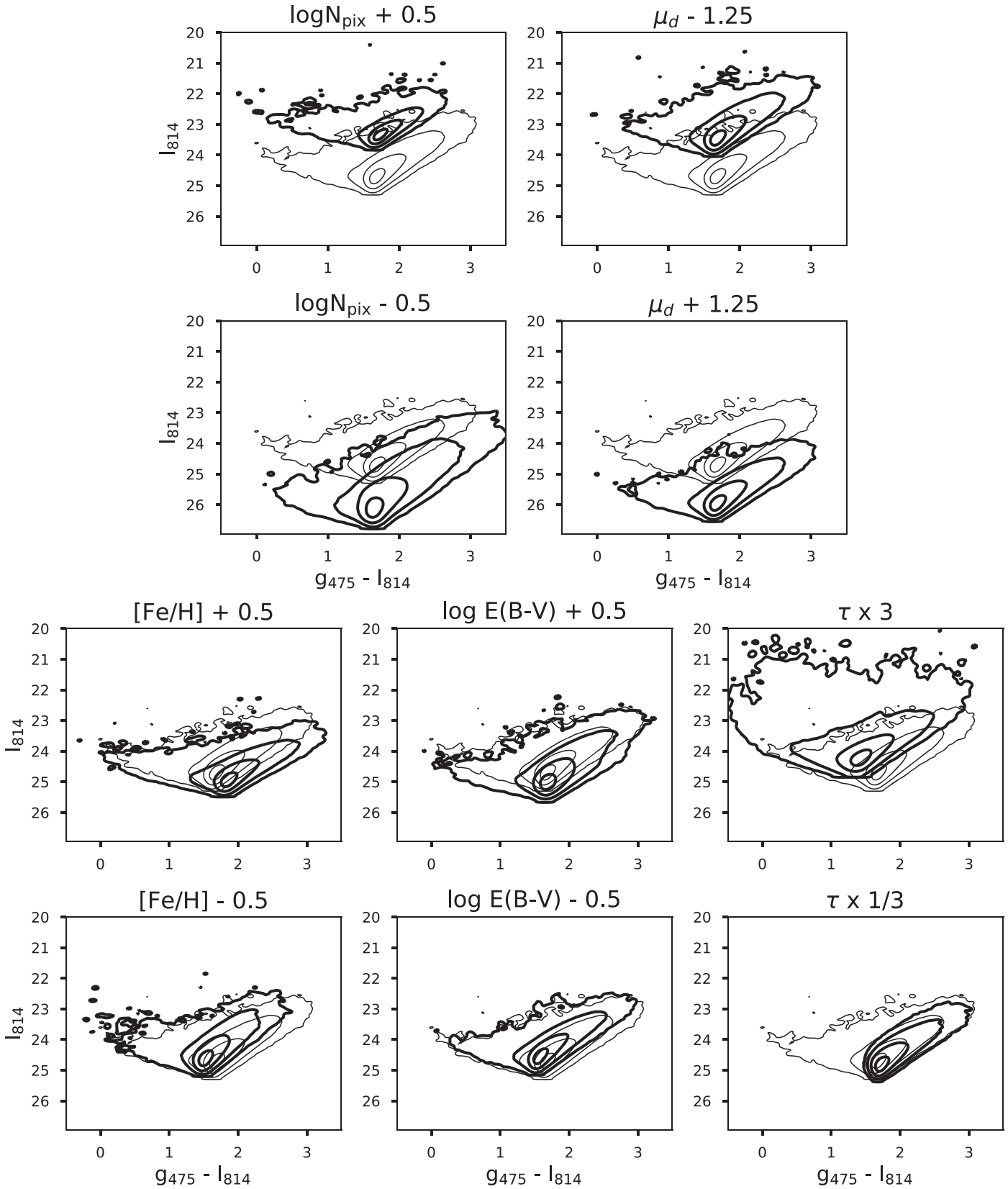

Figure 2. Sensitivity of the pCMD distribution to the various model parameters. The baseline model, shown in thin gray, is a fiducial-τ model (see Section 3.1). The contours show the bounds within which 39%, 87%, 99%, and 99.9% of the points lie (the 1σ, 2σ, 3σ, and 4σ contours, respectively). Varying each model parameter, shown in dark black, has significant effects on the shape and location of the pCMD. In the top two rows, the changes in $N_{\rm pix}$ and μ_d were chosen such that the average flux is equal in each row.

Of particular interest are the effects of $N_{\rm pix}$ (proxy for stellar surface density) and distance. The upper left column of Figure 2 shows that in addition to changing the average luminosity, $N_{\rm pix}$ also affects the dispersion of the pCMD distribution, due to the increase in Poisson surface-brightness fluctuations. We compare this to variations in distance modulus in the upper right column, which simply shifts the distribution in the vertical (luminosity) direction, with no effect on color. We show variations in $N_{\rm pix}$ and μ_d that each result in the same average luminosity; the dispersion of the distribution can be used to break the degeneracy between luminosity and distance. This represents a reformulation of the SBF method: as the same physical system is moved toward larger distance, surface brightness remains constant but the magnitude of fluctuations decreases as $N_{\rm pix}$ rises. This hints at the utility of pCMDs to simultaneously recover distances and stellar populations for galaxies in the semi-resolved regime. We demonstrate this conclusively in Section 3.

Varying metallicity shifts the peak of the distribution and has a notable effect on the slope of the upper-right wing. The pixels in that region contain RGB stars, and the pCMD is therefore sensitive to the metallicity-dependent slope of the RGB (Choi et al. 2016). The effects of increasing dust broaden the overall distribution but leave the slope of the RGB feature relatively intact. As we show in Section 3.2, the model is able to constrain dust and [Fe/H], although there remains a slight degeneracy between the two.

Changing the age parameter, τ, has significant effects on the blue wing of the pCMD. This represents the relative abundance of young, massive main-sequence stars. These hot stars are only abundant when there has been recent star formation (high τ), while only the old main-sequence and RGB stars contribute when recent star formation is suppressed (low τ).

2.4. Computational Infrastructure

Even when downsampling the isochrones by a factor of five, simulating a pCMD with the simplest SSP model requires $\mathcal{O}(10^8)$ random Poisson calls for a 512×512 image. Increasing the complexity and realism of a model by incorporating more age and metallicity points increases the computation time nearly linearly with the number of isochrones. With 21 age points and about five metallicity points, the more realistic physical models described in Section 2.2 require $\mathcal{O}(10^{10})$ Poisson draws and quickly become computationally infeasible on a traditional CPU.

`PCMDPy` includes an optional *GPU-accelerated* backend, which dramatically reduces the computational time required to simulate a pCMD. Given the isochrones representing a particular physical model, each GPU thread independently samples the stars from those isochrones into an individual pixel. This accelerates the simulation time of an individual model pCMD by a factor of $\sim$30$\times$ compared to the CPU implementation. Figure 3 shows the computation time required to simulate a pCMD of an SSP (model M1+S1) as a function of $N_{\rm im}$. The CPU tests were run on an Intel Xeon E5-2620 processor (2.1 GHz) and the GPU tests on an Nvidia Tesla K20Xm.[9]

The original code of Conroy & van Dokkum (2016) simulated an $N_{\rm im} = 256$ pCMD for a relatively simple physical model (M1+S2+E1+D1) in $\approx$1 s. The same computation in `PCMDPy` takes only $\approx$0.25 s on an Nvidia Tesla K20Xm. For our preferred model with a Gaussian MDF and 21 age bins (see Section 3.1), `PCMDPy` simulates a pCMD with $N_{\rm im} = 512$ in $\approx$2 s. GPU acceleration allows for simulating both more complex (realistic) physical models and larger image sizes (less stochasticity in likelihoods; see Section 2.5) than would be feasible with CPUs alone.

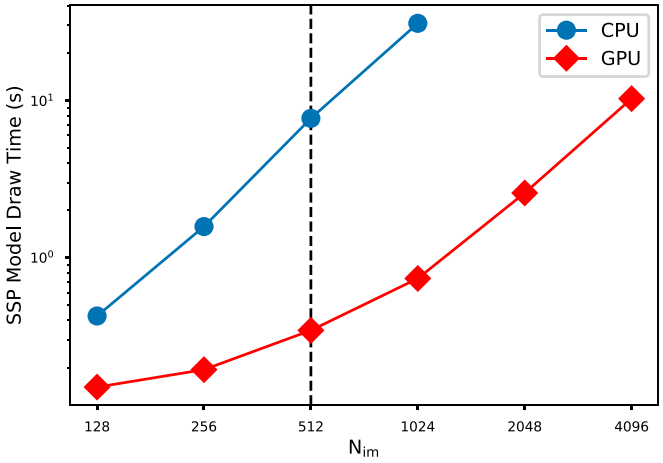

Figure 3. Computation time for drawing an SSP model pCMD is shown for the CPU and GPU-accelerated versions of the code, as a function of the simulated image size. For our fiducial model size of $N_{\rm im} = 512$, the GPU-accelerated code results in nearly 30$\times$ speedup, and it could be 4$\times$ larger still with more modern GPU chips.

2.5. Likelihoods and Posterior Sampling

We evaluate the likelihood of a pCMD given a model with a binned Hess diagram and Gaussian statistics. With this approach, we create a binned 2D histogram of pixels in the pCMD and compare the relative number of counts in the two distributions, normalized to the total number of pixels. We choose bins of width 0.05 mag in each dimension, a width chosen to be roughly equivalent to the observational uncertainties, but we show in Section 3.2 that the resulting posteriors are fairly insensitive to this choice. We compute the log-likelihood $\mathcal{L}$ using the counts of data pixels d_i and model pixels m_i in each Hess bin, as follows:

$$\mathcal{L} \propto \sum_i \left[-\frac{\left(\frac{d_i}{N_d} - \frac{m_i}{N_m}\right)^2}{2\sigma_i^2} \right] + \mathcal{C}, \tag{2}$$

where the uncertainty σ_i is approximated by the square root of the number of Hess bin counts, added in quadrature:

$$\sigma_i = \max\left(\sqrt{d_i^2 + m_i^2}, 2\right). \tag{3}$$

We apply a floor to the uncertainty to down-weight very rare bins in the Hess diagram, where differences in simulated and data image sizes may unintentionally bias the likelihood. This is one aspect of a general problem of evaluating likelihoods in Hess diagram space: how to handle data points when there are no (or very few) model points. For instance, a Poisson likelihood model would return zero likelihood if the model predicts zero pixels in a bin with even a single data pixel.

[9] These are the default GPUs available to us. The latest generation of Nvidia GPUs (the Tesla V100) could lead to an additional $\sim$4$\times$ speedup.

We add an additional likelihood term, corresponding to the difference in mean color C and magnitude M between the two distributions, with error of 0.05 mag:

$$\mathcal{C} = -\frac{(C_d - C_m)^2}{2(0.05)^2} - \frac{(M_d - M_m)^2}{2(0.05)^2}. \quad (4)$$

Without this term, two pCMDs that do not overlap at all ($d_i = 0 \ \forall \ i$ s.t. $m_i \neq 0$) are equally poor fits regardless of whether they are offset by an average of 1 or 10 mag. The addition of this term gives slight preference to models where the center of the distributions roughly aligns with the data, without substantially affecting best-fit estimates because the relative magnitude of the term is quite small.

We sample the posterior using a new Python package for dynamic nested sampling, `dynesty` (J. S. Speagle 2019, in preparation). Nested sampling (e.g., Skilling 2004; Feroz et al. 2009, 2013; Handley et al. 2015) is an approach similar to the commonly used Markov chain Monte Carlo (MCMC) technique, representing the posterior through a collection of samples from the distribution. Unlike MCMC, nested sampling efficiently computes the Bayesian evidence (also called marginal likelihood), allowing for principled model comparisons. Nested sampling algorithms also allow for sophisticated handling of multimodal distributions. See the references above for more details on nested sampling.

Throughout this work, parameter estimates are reported as the median of the marginalized posterior probability function, and error bars are reported as the 68% equal-tailed credible interval, unless otherwise stated.

The pCMD likelihoods are stochastic, meaning that recalculating the likelihood multiple times for the same input parameters will result in a different log-likelihood. This presents a statistical challenge for any posterior sampling algorithm, and without proper accounting it leads to an underestimate in uncertainties (see, however, the pseudomarginal MCMC approach; Andrieu & Roberts 2009). We discuss this further in Appendix A and detail a method for postprocessing the results to recover reasonable posteriors.

Fitting a pCMD with models of size $N_{\rm im} = 512$ takes $\sim$100 GPU hours for most cases. This is a speedup of a factor of 7$\times$ compared to the original code of Conroy & van Dokkum (2016), which required around 700 CPU hours to fit a posterior using $N_{\rm im} = 256$. This was largely due to the need to combine the posteriors of 10 independent MCMC runs to overcome the stochasticity of the likelihoods. The larger simulated image sizes allowed by GPU acceleration decrease the inherent stochasticity of the likelihoods, making combining multiple fits unnecessary.

3. A Suite of Mock Tests

3.1. Mock pCMD Models

To evaluate the capability of the code to recover model parameters, we run a series of mock tests where we fit models to simulated pCMDs generated from the code. In most cases, we fit models with the same physical components (i.e., same metallicity, SFH, and dust model) and hyperparameters as used to generate the data. We model observations in the F814W (I_{814}) and F475W (g_{475}) bands of ACS, use a mock image size of $N_{\rm im} = 256$ and model image size of $N_{\rm im} = 512$, and assume no sky noise. Exposure times for F814W and F475W are set to 3235 and 3620 s, respectively, corresponding to the typical values of data from the PHAT survey (Dalcanton et al. 2012).

The fiducial model we frequently study, which we denote the "fiducial-τ" model, has a fixed-width Gaussian MDF, tau-SFH, and fixed-width log-normal dust screen, and the distance is allowed to vary (components M2+S3+E2+D2; see Section 2.2 for details). Unless otherwise specified, the free parameters used to generate the mock pCMDs are set to [Fe/H] $= -0.25$, $\log E(B-V) = -0.5$, $N_{\rm pix} = 10^2$, $\tau = 3$ Gyr, and $\mu_d = 26.0$. The width of the MDF is set to $\sigma_{\rm [Fe/H]} = 0.2$, and the width of the log-normal dust model is set to $\sigma_{\rm dust} = 0.1$.

We also study a "fiducial-nonparametric" model, where the SFH is fit with a five-bin nonparametric model (S5). The five SFH bins correspond to the following ages:

1. SFH0: 1–100 Myr
2. SFH1: 100 Myr–1 Gyr
3. SFH2: 1–3 Gyr
4. SFH3: 3–10 Gyr
5. SFH4: 10–14 Gyr

Unless otherwise specified, the same free parameters are used to generate mock pCMDs as above, with the exception of the SFH, which is a constant-SFR model with $N_{\rm pix} = 10^2$.

In both models, we assume flat priors over all parameters. In most cases, we assume [Fe/H] $\in [-0.5, +0.25]$, $\log E(B-V) \in [-1, 0]$, $\log N_{\rm pix} \in [2, 5]$, $\tau \in [0.1, 8.0]$ Gyr, and $\mu_d \in [22, 26]$. In the case of the nonparametric SFH, we assume flat priors in log SFH, of width ± 1 dex around the true underlying SFH.

3.2. Recovery of Nonparametric SFHs

Figure 4 shows the posterior probability distribution for a fiducial-nonparametric model fit and demonstrates that `PCMDPy` can simultaneously recover the input model parameters well. The metallicity, dust content, distance, and total $N_{\rm pix}$ (computed as the sum of all SFH bins) are all recovered to within 1σ, as shown in the marginalized histograms. There is a notable degeneracy between $N_{\rm pix}$ and μ_d, but the distance modulus is still constrained to within 0.1 dex.

The oldest bins of SFH (bins 3 and 4) are somewhat degenerate, but the total star formation older than 3 Gyr is fairly well constrained, as shown in the marginalized contours of those two bins. Figure 4 also shows the derived constraints on the SFR in each pixel, as a function of age. Compared to the original prior bounds allowed (± 1 dex in each bin), the model has recovered reasonably tight constraints on the SFH.

We also examine the recovered SFH for various input SFHs. The results are shown in Figure 5, for constant-SFR and $\tau = 3$ Gyr models and increasing $N_{\rm pix}$ from 10^2 to 10^5. Each model is able to recover the underlying SFH, with the true SFH contained within the 68% credible interval in nearly all cases. In constant-SFR models, the oldest ages of star formation have the largest uncertainties, while the relatively higher SFR in the τ models is easier to constrain precisely.

3.3. Evaluation of Model Choices

We use the suite of mock tests to evaluate the effect of various hyperparameter choices and model families on the recovered parameters. For simplicity, these tests are performed using the fiducial-τ model, and the results are shown in Figure 6. These findings may not generalize fully to all models or regions of parameter space (especially higher $N_{\rm pix}$), and we

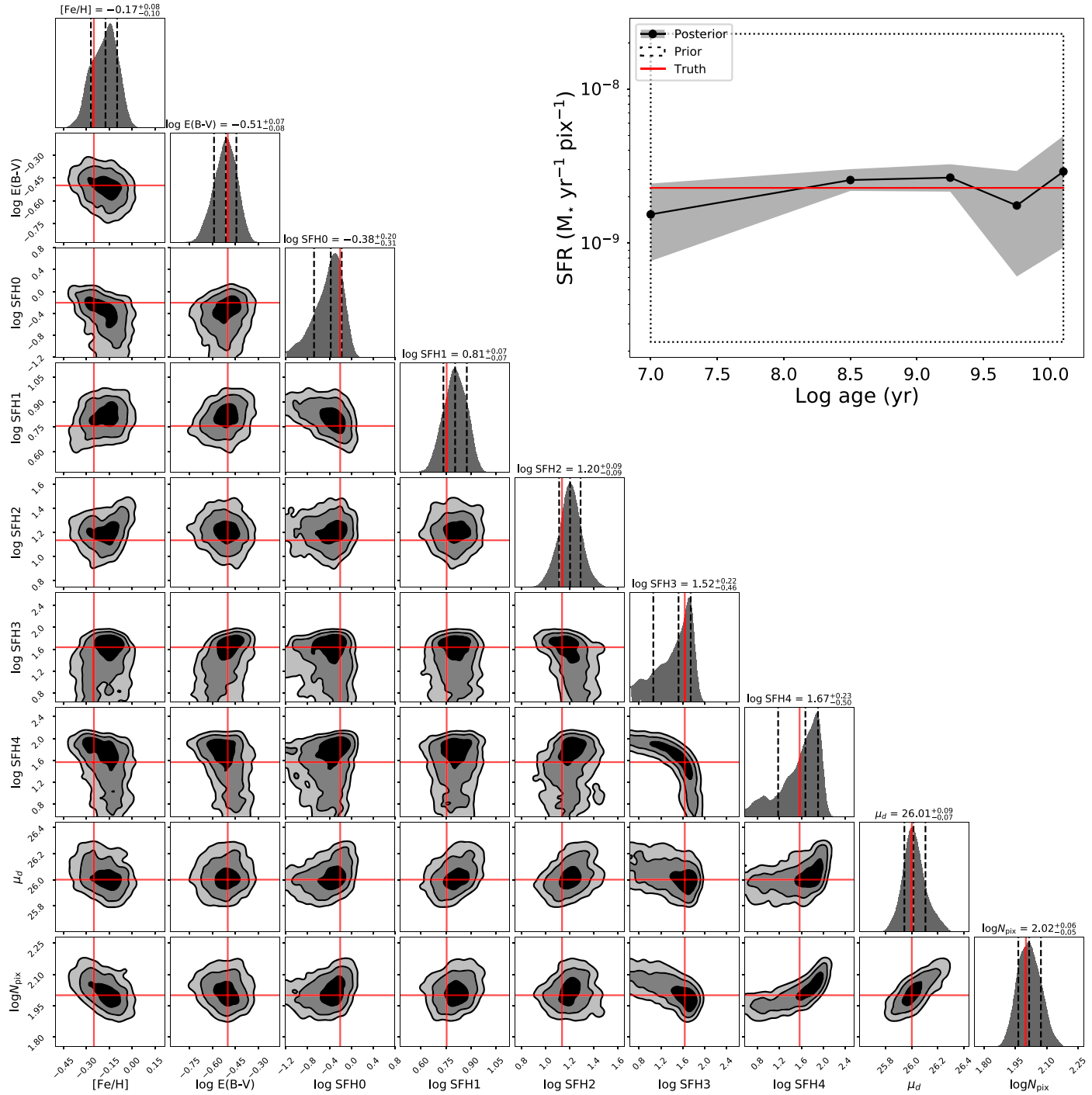

Figure 4. Recovered posterior probability distribution from a mock test, using a five-bin nonparametric SFH model to fit a constant star formation rate model. Here, log $N_{\rm pix}$ is a quantity derived from the five SFH bins. The 1σ, 2σ, and 3σ contours are shown. The upper right panel shows the recovered posterior estimates of star formation rates as a function of time. The true values used to generate the mock pCMD are shown with red lines. The full model is specified in Section 3.1 (fiducial-nonparametric model). Every input parameter is recovered within the 68% credible interval.

caution all users of PCMDPy to think carefully about all hyperparameter choices before fitting.

The first column of Figure 6 shows the effect of fitting distance as a free parameter (D1 versus D2). We find we can recover the distance modulus to ± 0.1 mag. With the exception of $N_{\rm pix}$ (which is slightly degenerate with μ_d), estimates of other parameters are just as well constrained regardless of whether distance is included in the fit.

The second column shows the effect of fitting for the width of the MDF and dust distributions (simple: M2+E2; complex: M3+E3). We find the model is unable to constrain these widths, which leads to a small bias and inflated uncertainties in the estimates of other parameters. We therefore recommend against fitting for these width parameters, unless available information suggests a fairly informative prior is warranted.

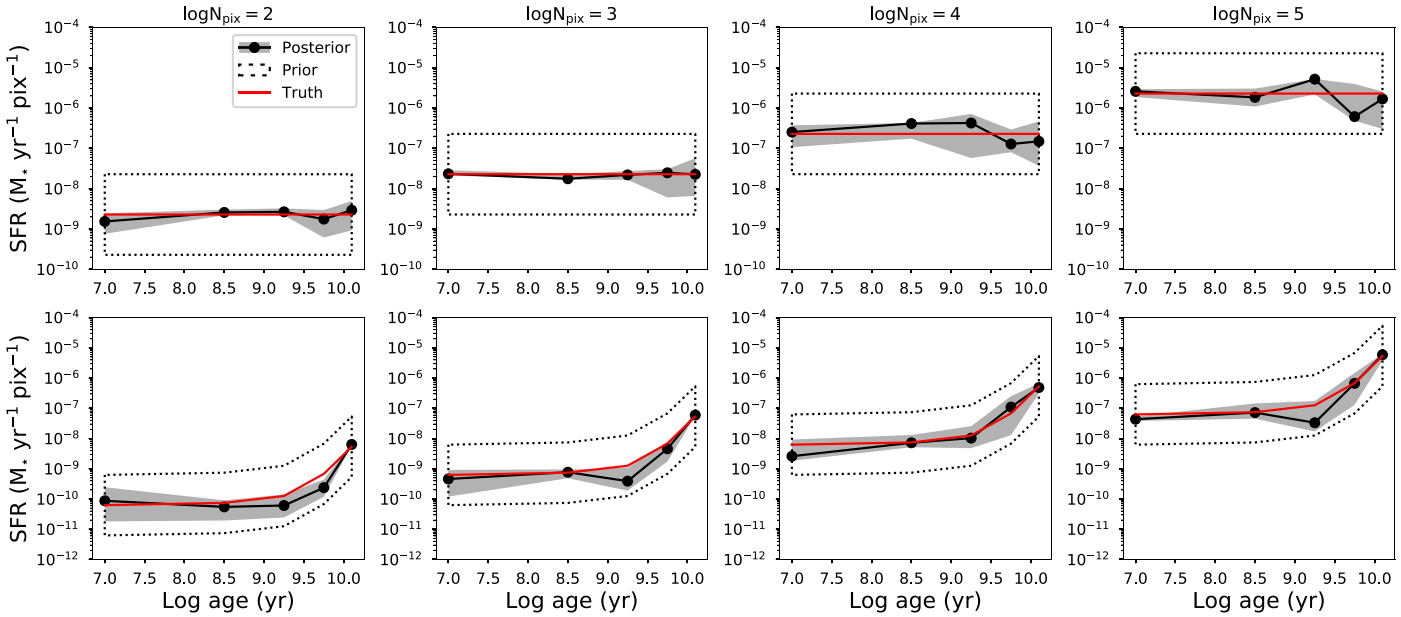

Figure 5. Recovered SFH for several input models, as a function of $N_{\rm pix}$. The input SFH is shown in red, while the posterior estimates (median and 68% credible interval) are shown in gray. The prior, shown as a dotted outline, is assumed flat within ± 1 dex of the true SFH. Top: the input SFH has a constant star formation rate. Bottom: a tau-SFH model with $\tau = 3$ Gyr.

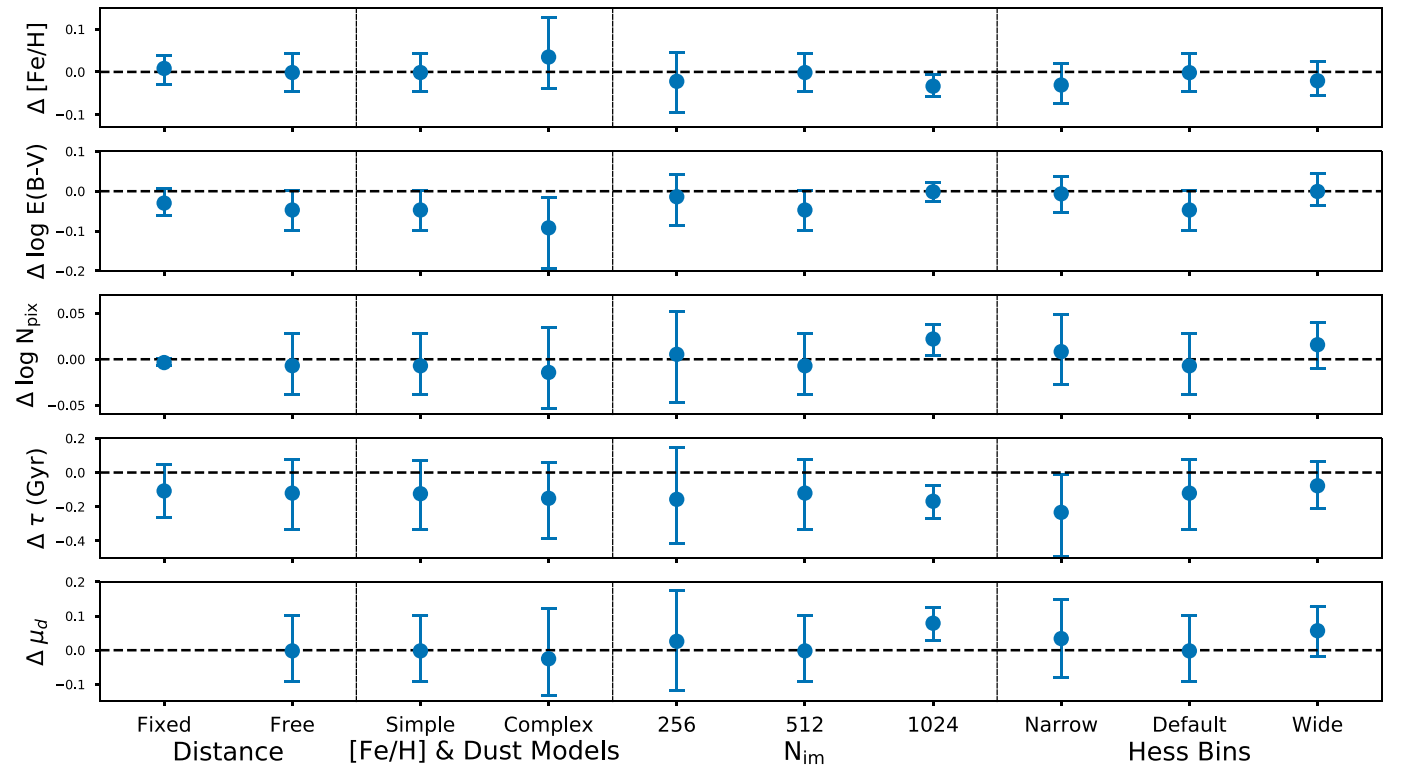

Figure 6. Error in best fit (posterior median) and 68% credible interval in each parameter for several mock tests with various modeling choices. The baseline model assumed is specified in Section 3.1 (fiducial-τ). Column 1: distance is held fixed in the first model. Column 2: the simple models correspond to M2+E2, while the complex models correspond to M3+E3 (the widths $\sigma_{\rm [Fe/H]}$ and $\sigma_{\rm dust}$ are fit as free parameters). Column 3: the size of the model image, $N_{\rm im}$. Column 4: the widths of the likelihood Hess bins are either 0.02 dex (narrow), 0.05 dex (default), or 0.1 dex (wide).

In the third column, we vary the model image size $N_{\rm im}$. Smaller image sizes ($N_{\rm im} = 256$) lead to inflated uncertainties, despite taking just as long to fit as the default size of $N_{\rm im} = 512$. The additional stochasticity in the likelihoods at small $N_{\rm im}$ leads to more severe drops in sampling efficiency (see Appendix A). Larger image sizes ($N_{\rm im} = 1024$) take significantly longer to fit, requiring fewer nested sampling live points to converge in a reasonable time (fewer than 300 GPU hours), which results in underestimated errors.

The final column shows various choices for the width of the Hess bins used to evaluate likelihoods. The bin size has no significant effect on the recovery of the posterior, although we

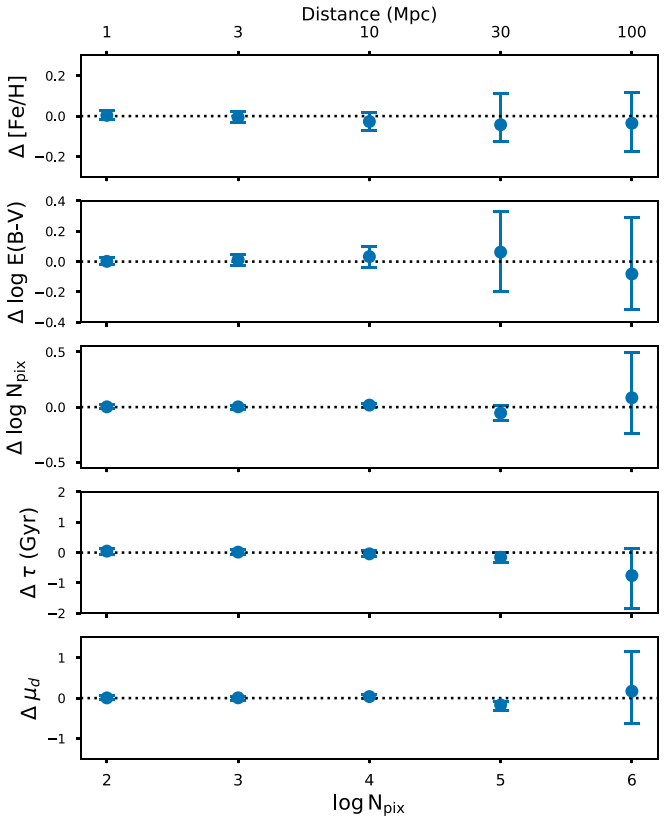

Figure 7. Same as Figure 6, where the same physical system is modeled at increasing distance. Here, $N_{\rm pix}$ increases with distance, while $N_{\rm im}$ of the mock pCMD is decreased to keep the simulated mass enclosed ($\approx 10^8 M_\star$) constant. The model recovers highly precise estimates of all parameters out to $\approx$10 Mpc, but the true results remain within the errors at distances as large as 100 Mpc.

note that this may not hold true at higher $N_{\rm pix}$, when the pCMD distribution is more compact.

In Figure 7, we study the ability of `PCMDPy` to model galaxies as a function of distance. At larger distances, the magnitude of surface-brightness fluctuations decreases (since $N_{\rm pix}$ is larger), making the models less sensitive to the underlying stellar populations. In addition, the same physical size region will correspond to far fewer pixels, and gradients in surface brightness and stellar populations will make the modeling assumptions invalid over too many pixels.

To simulate these effects, we begin with a fiducial-τ model with $N_{\rm pix} = 10^2$, modeled at distance of 1 Mpc, roughly equivalent to the disk of a galaxy like M31 (Conroy & van Dokkum 2016; see also B. A. Cook et al. 2019, in preparation). The mock pCMD for this system is generated with $N_{\rm im} = 2048$, corresponding to a region 500pc on a side with total mass $M_\star \approx 10^8\, M_\odot$. We then approximate the effects of observing the same physical system at larger distance and correspondingly higher $N_{\rm pix}$, while decreasing $N_{\rm im}$ of the mock pCMD to keep the physical size and total mass constant. Five such mock data sets (from 1 to 100 Mpc) are fit to the same model (with simulated $N_{\rm im} = 512$).

The fits from this experiment are in excellent agreement with the true models until $D \gtrsim 10$ Mpc ($N_{\rm pix} \gtrsim 10^4$), at which point the uncertainty rises sharply. But even out to 100 Mpc, the true parameters remain within the 68% credible interval.

The pCMD method should therefore provide interesting constraints of stellar populations and distance out to large

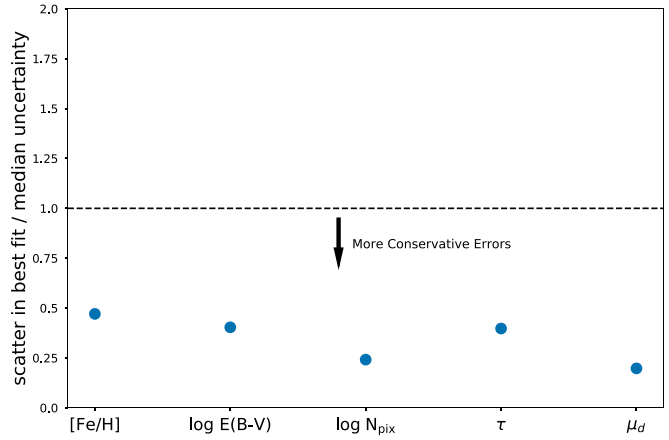

Figure 8. The scatter in posterior median estimates of each model parameter is shown for eight mock tests, evaluated on different realizations of the input model and normalized by the median uncertainty in each parameter. The fact that all values are below 1.0 implies slightly overestimated uncertainties.

distances and may serve as a useful complement to the existing SED modeling technique.

To test the stability of the results, we run eight fits on different realizations of pCMDs generated with the same underlying model. The scatter in the estimates of each fit should fall within the typical uncertainty, or else the uncertainties are likely underestimated. Figure 8 shows the results of this test. For all parameters, the scatter (standard deviation) in median estimates is significantly less than the typical uncertainty. This suggests that the model uncertainties are likely overestimated by a factor of roughly $2\times$. It could be possible to correct for this with a more careful selection of $\mathcal{L}_{\rm max}$, defined in Appendix A, but we are comfortable with overestimating our uncertainties, given the relatively loose method for dealing with the stochastic likelihoods detailed in the appendix.

3.4. Model Mismatch Tests

We investigate the effect of fitting a pCMD with a different model than used to generate the data in Figure 9. First, we show fits where the physical model is incorrect. Incorrectly modeling the metallicity distribution function may lead to a minor systematic bias in [Fe/H] and τ, but for $\sigma_{\rm [Fe/H]}$ on the order of a few tenths of a dex, the effects are small and within the uncertainties.

Slight biases also arise from incorrectly modeling the SFH or the structure of the dust. We fit a model with $F_{\rm dust} = 1.0$ to data generated with $F_{\rm dust} = 0.5$, effectively ignoring that half of the stars should lie above the screen of dust. This results in an underestimate of the dust content, as well as bias in τ and $N_{\rm pix}$. Accurately accounting for the geometries of stars relative to the dust is therefore important to recovering accurate measurements.

We also show the effect of modeling a τ-SFH with a nonparametric model. For comparison to the τ models, we compute the average age of the inferred SFHs and convert to an effective τ. The nonparametric model appears slightly biased toward preferring old ages (lower τ). This results in slight positive bias in $N_{\rm pix}$ and μ_d in order to result in the same total luminosity. The age bias could be mitigated by increasing the number of SFH bins modeled, but this becomes computationally demanding. Incorrectly specifying the physical model can lead to subtle systematic biases, and it is not always easy to

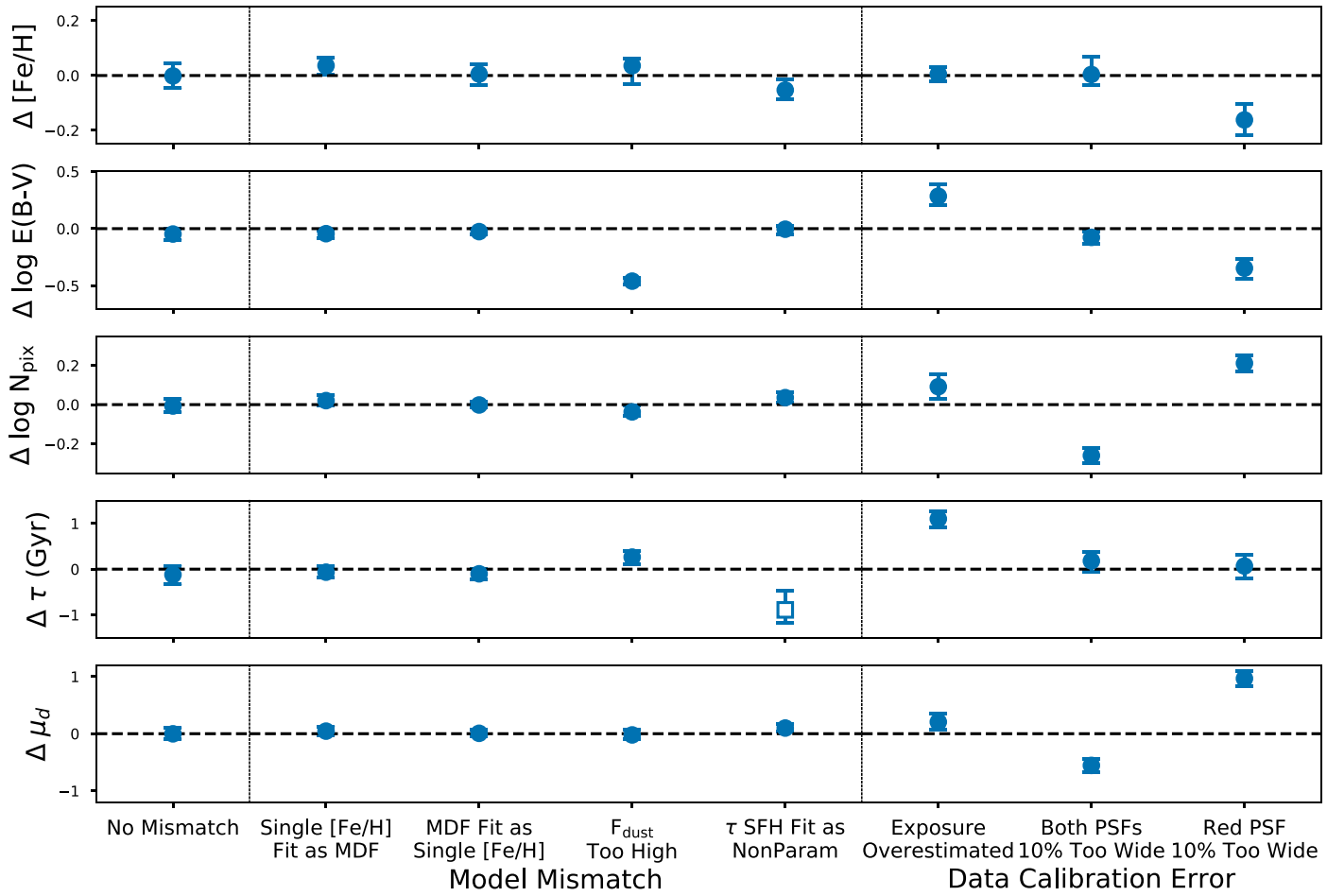

Figure 9. Same as Figure 6, for mock tests where there is a mismatch between the models used to generate the mock pCMD and to fit it. Column 1: there is no model mismatch. Column 2: cases where the physical model assumed is incorrect. The white box shows the approximate τ derived from the nonparametric SFH. Column 3: cases where observational hyperparameters are miscalibrated.

diagnose such a mismatch. One possible approach could be to compare the Bayesian evidence (computed through the nested sampling algorithm) for multiple model choices, or alternatively by studying the residual patterns in Hess diagram space. In future work, we additionally intend to study the effect of different SFH priors, such as those suggested by Leja et al. (2019).

Other possible mismatches between data and model include errors in the calibration of the observational data. If the exposure time of images is overestimated (the model shown overestimates the true exposure by $2\times$), the model greatly overpredicts the age parameter τ, and most other parameters are also biased from their true values. When extracting pCMDs from observations, it is important to properly understand the observing conditions and whether multiple exposures were coadded to produce the final photometry.

The final two examples showcase the complex effects of poor PSF calibrations. If both PSF models are too wide (FWHM overestimated by 10%), best-fit models are strongly biased in all parameters. Yet if only one PSF model is miscalibrated (here, the F814W filter), many of the biases go in the opposite direction. We investigate this effect more fully in upcoming work (B. A. Cook et al., 2019, in preparation) when we discuss the important practical challenges in selecting observational data to analyze with the pCMD technique.

4. Conclusions

We have presented a study of the application of pixel color–magnitude diagrams (pCMDs) for studying galaxies in the semi-resolved regime, as first introduced by Conroy & van Dokkum (2016). We developed the open-source, GPU-accelerated Python package `PCMDPy` and detailed its primary modeling assumptions and physical models implemented. We also presented the results of a suite of mock tests that demonstrate the potential for using `PCMDPy` to constrain important physical parameters of galaxies.

The main results of this work are as follows:

1. We have developed the new package `PCMDPy` for modeling pCMDs and inferring the physical properties of galaxies. It contains a number of improved physical models compared to the original code of Conroy & van Dokkum (2016), including a Gaussian MDF and a log-normal dust extinction model, and allows for simultaneous fitting of distance modulus. The code uses GPU acceleration to allow the modeling of pCMDs with these complex models and larger simulated image sizes in significantly less time. Posteriors are sampled using the dynamic nested sampling code `dynesty`, and fitting a model requires about one-seventh as much computational time as the original code.

2. We demonstrated that pCMDs are sensitive to the distance to a galaxy and that our code can simultaneously recover accurate estimates of distance, SFH, metallicity, and dust content.
3. We show that highly precise measurements of distance and stellar populations should be recoverable for massive galaxies at least to 10 Mpc with *HST*-like resolution, and reasonable constraints are possible out to 100 Mpc.
4. The model is relatively robust against small biases in the assumed metallicity distribution function. Mischaracterized dust or SFH models can lead to small systematic biases in recovered parameters. Additionally, it is crucial to have good estimates of optical properties such as exposure time and well-calibrated PSF models. Future work will discuss other important considerations for selecting observational data to fit with `PCMDPy`.
5. The effects of the stochastic likelihood model present challenges for traditional sampling methods. The larger images that can be simulated using GPUs reduce this burden somewhat, but additional steps must be taken to account for the bias toward positive-likelihood fluctuations. Our chosen approach may result in overestimating the derived uncertainties by a factor of order $2\times$.

We thank Aaron Dotter and Jieun Choi for helpful discussions around the details of the MIST models. B.C. acknowledges support from the NSF Graduate Research Fellowship Program under grant DGE-1144152. This work is supported in part by HST-AR-14557. The computations in this paper were run on the Odyssey cluster supported by the FAS Division of Science, Research Computing Group at Harvard University.

Software: This research has made use of NASA's Astrophysics Data System, as well as the following software packages: PyCUDA (Klöckner et al. 2012), Dynesty (J. S. Speagle 2019, in preparation), NumPy (van der Walt et al. 2011), Matplotlib (Hunter 2007), IPython (Pérez & Granger 2007), Jupyter (Kluyver et al. 2016), SciPy (Jones et al. 2001), Pandas (McKinney 2010), and Astropy (The Astropy Collaboration et al. 2013, 2018).

Appendix A
Stochastic Likelihoods

The forward-modeling procedure for generating pCMDs is stochastic, and therefore so are our likelihoods. The simulated pCMD is a random realization of the underlying model, and therefore evaluating the likelihood of the same point in parameter space multiple times will produce different results. Our tests indicate that the distribution of likelihoods, given fixed model parameters, has a well-defined center but a long positive tail, essentially resulting from "overfitting" to the noise in the data. Larger simulated image sizes $N_{\rm im}$ produce pCMDs that average over the many rare fluctuations, and therefore decrease the inherent stochasticity.

Any level of stochasticity poses significant problems for nested sampling or MCMC, because a positive fluctuation in likelihood will result in a model point being given larger weight than it truly deserves. In either MCMC or nested sampling, this has the detrimental effect of significantly decreasing sampling efficiency. The sampling algorithms will take longer to find a comparably good fit, resulting in decreased sampling efficiency and a cascading effect where most subsequently sampled points are also drawn from the positive-fluctuation tails.

Our tests have identified two other detrimental effects of nested sampling in a stochastic likelihood model. The weights assigned to the sampled points are biased toward only weighting the largest fluctuations, often resulting in only a single point being assigned a nontrivial weight. Since nested sampling integrates toward increasing likelihood, the final many points returned by the algorithm, and those assigned highest weight, almost all represent positive likelihood fluctuations, rather than actually better models.

Furthermore, the autostopping criterion used by many algorithms, including `dynesty`, relies on $\Delta \ln Z$, the estimated evidence remaining (e.g., Feroz et al. 2009). This value is biased high by the positive fluctuations, leading the algorithm to continue sampling longer than would be desired. Combined with the decrease in sampling efficiency, our `dynesty` fits often become stuck, taking thousands of likelihood calls to return a new sample.[10] In practice, we stop the model fit after a fixed number of iterations. As discussed below, we find that we can later account for the bias in likelihoods, and when we do so the fits are almost always well converged by the $\Delta \ln Z$ criterion.

We show the effect of this likelihood bias in Figure 10. The blue line shows the measured log-likelihood of the `dynesty` samples from a mock test. The upturn at the final points shows extreme positive fluctuations: using the results straight from `dynesty`, the final point has 99% of the weight, producing an implausibly peaked posterior. To demonstrate that these points all represent positive likelihood fluctuations, we recomputed the likelihood of several of the points in parameter space, and we show the measured distribution of likelihoods as error bars.

It is clear that the likelihoods returned by `dynesty` are biased high from the average likelihood for each model. We can estimate what the maximum realistic likelihood should be, by computing many realizations of the "true" model used to generate the pCMD and evaluating the fits. This is shown in the gray band. As should be expected, all of the mean likelihoods for the resampled points lie roughly within this band. We therefore have a means for estimating which likelihoods are trustworthy and which are likely biased high by fluctuations.

We use this insight to adopt a postprocessing procedure to account for this likelihood bias. At the completion of each `dynesty` run, we recompute the likelihood of the best-fit model 100 times and then adopt the median value as a likelihood threshold, $\mathcal{L}_{\rm max}$. Any likelihoods above that threshold are capped to that value, and we recompute the weights and convergence statistics using the updated likelihoods.

We find that this approach is successful at eliminating the bias from likelihood fluctuations, by down-weighting the final handful of samples that are most subject to this stochastic bias. Figure 11 shows the mean and uncertainties in model parameters recovered as a function of this $\mathcal{L}_{\rm max}$. When $\mathcal{L}_{\rm max}$ is too high, the mean estimates vary sharply, as only one or a few sampled points are given all of the posterior weight. Using our adopted $\mathcal{L}_{\rm max}$ ceiling, we are averaging over sufficiently many sampled points to recover stable means and reasonable uncertainties. When the remaining-evidence criterion is

[10] An example of this principle, for a simplified case of a stochastic likelihood, is included as a demonstration in the Dynesty repository: https://github.com/joshspeagle/dynesty/blob/master/demos/Examples%20–%20Noisy%20Likelihoods.ipynb.

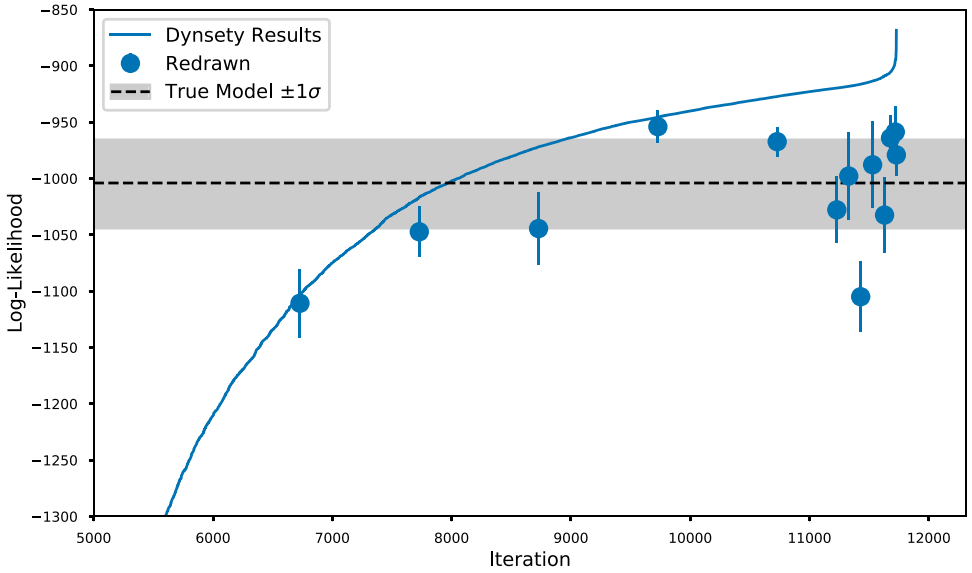

Figure 10. Demonstration of the bias induced by sampling from a stochastic likelihood function. The blue line shows the log-likelihood reported for each nested sampling point of a mock test as a function of iteration. The blue error bars show the distribution of log-likelihood from recomputing the log-likelihood of the same points in parameter space 10 times. The distribution of log-likelihoods computed using the true (known) model parameters is shown in black, and any log-likelihood above that range is simply a rare positive fluctuation that is overfitting the stochastic effects.

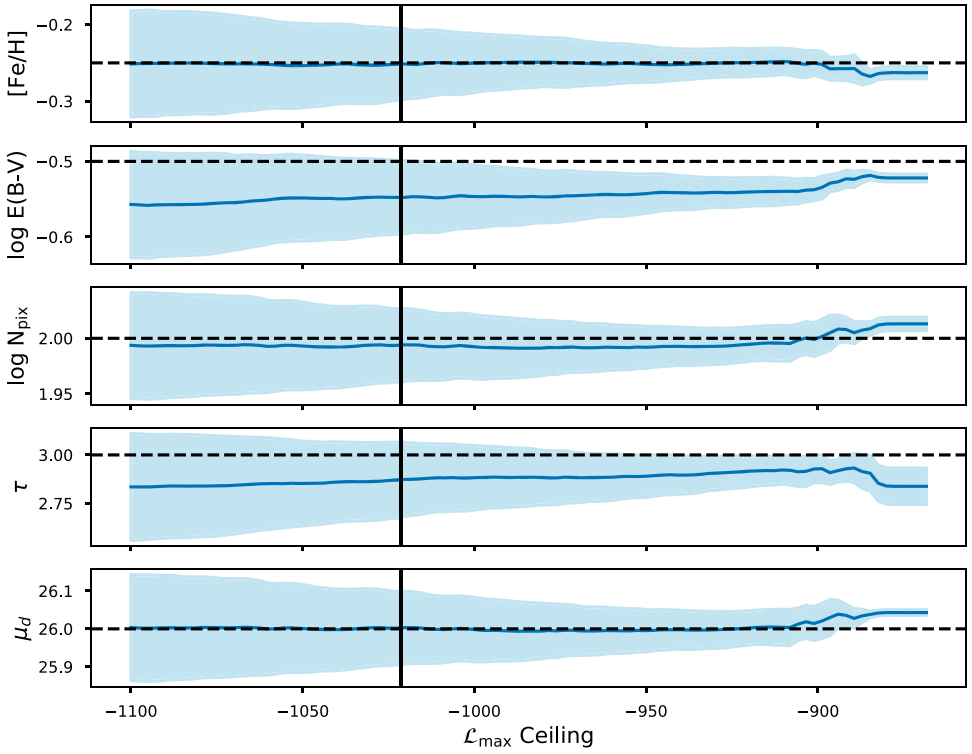

Figure 11. Marginalized mean estimates (solid line) and uncertainties in each parameter as a function of the maximum log-likelihood ceiling applied. The solid black line shows the adopted threshold: the median log-likelihood of the best-fit model recomputed 100× against the data. The dashed lines show the true parameters.

recomputed using the adjusted likelihoods, the fits are almost always extremely well converged by standard stopping criteria ($\Delta \ln Z \lesssim 0.01$).

This method for handling the stochastic likelihoods is a rough, heuristic approach, so the exact statistical uncertainties must be treated cautiously. The mock tests shown in this work are therefore important for validating that the approach recovers reasonable errors. As shown in Figure 8, the scatter in mean estimates between multiple independent trials is smaller than the typical uncertainty, indicating that if anything this relatively ad hoc procedure is somewhat overly conservative in the model uncertainties.

Appendix B
Cloud Computing and GPUs

The `PCMDPy` GPU acceleration is written in `CUDA`, a proprietary language that only operates on Nvidia-brand GPUs.

This limits the systems that can utilize the GPU-accelerated version of PCMDPy. Most personal computers and laptops (especially Macs) do not have Nvidia cards; Intel and AMD are common competitors, but CUDA code cannot be compiled to run on their architecture. However, the GPUs installed in most computer clusters hosted at research institutions are Nvidia, and cloud computing offers an alternative option for obtaining GPU-accelerated computational resources.

To make PCMDPy's GPU capabilities more broadly applicable, we have also developed a Docker container architecture for fitting pCMDs in the cloud using AWS-Batch. Using this framework, pCMD models can be fit by renting GPU-enabled instances on demand, with the benefit of having practically limitless scaling potential (large numbers of jobs can be run simultaneously). As of this writing, such instances are available for around $1/GPU hour (with faster GPUs available for higher cost, in nearly linear proportion to their speedup). As PCMDPy typically takes around 100 GPU hours to fit a model, this translates to around $100 per model fit. Full details and examples will be detailed later with the public PCMDPy code release paper. For a detailed example on using AWS in astronomical research, see Williams et al. (2018).

ORCID iDs

B. A. Cook https://orcid.org/0000-0002-4637-760X
Charlie Conroy https://orcid.org/0000-0002-1590-8551
Pieter van Dokkum https://orcid.org/0000-0002-8282-9888
Joshua S. Speagle https://orcid.org/0000-0003-2573-9832

References

Andrieu, C., & Roberts, G. O. 2009, AnSta, 37, 697
Cantiello, M., Jensen, J. B., Blakeslee, J. P., et al. 2018, ApJL, 854, L31
Carlsten, S., Beaton, R., Greco, J., & Greene, J. 2019, ApJ, submitted (arXiv:1901.07575)
Carnall, A. C., Leja, J., Johnson, B. D., et al. 2019, ApJ, 873, 44
Choi, J., Dotter, A., Conroy, C., et al. 2016, ApJ, 823, 102
Cohen, Y., Dokkum, P. v., Danieli, S., et al. 2018, ApJ, 868, 96
Conroy, C. 2013, ARA&A, 51, 393
Conroy, C., & van Dokkum, P. G. 2016, ApJ, 827, 9
Conroy, C., van Dokkum, P. G., & Choi, J. 2015, Natur, 527, 488
Dalcanton, J. J., Fouesneau, M., Hogg, D. W., et al. 2015, ApJ, 814, 3
Dalcanton, J. J., Williams, B. F., Lang, D., et al. 2012, ApJS, 200, 18
Dolphin, A. E. 2002, MNRAS, 332, 91
Feroz, F., Hobson, M. P., & Bridges, M. 2009, MNRAS, 398, 1601
Feroz, F., Hobson, M. P., Cameron, E., & Pettitt, A. N. 2013, arXiv:1306.2144
Greene, J. E., Veale, M., Ma, C.-P., et al. 2019, ApJ, 874, 66
Handley, W. J., Hobson, M. P., & Lasenby, A. N. 2015, MNRAS, 453, 4384
Hopkins, P. F., Kereš, D., Oñorbe, J., et al. 2014, MNRAS, 445, 581
Hunt, L. K., De Looze, I., Boquien, M., et al. 2019, A&A, 621, 51
Hunter, J. D. 2007, CSE, 9, 90
Jones, E., Oliphant, T., Peterson, P., et al. 2001, SciPy: Open Source Scientific Tools for Python, https://www.scipy.org/citing.html
Klöckner, A., Pinto, N., Lee, Y., et al. 2012, ParC, 38, 157
Kluyver, T., Ragan-Kelley, B., Pérez, F., et al. 2016, in Proc. 20th Int. Conf. Electron. Publish., Positioning and Power in Academic Publishing: Players, Agents and Agendas, ed. F. Loizides & B. Schmidt (Amsterdam: IOS Press), 87
Krist, J. E., Hook, R. N., & Stoehr, F. 2011, Proc. SPIE, 8127, 81270J
Kroupa, P. 2001, MNRAS, 322, 231
Leja, J., Carnall, A. C., Johnson, B. D., Conroy, C., & Speagle, J. S. 2019, ApJ, 876, 3
Lewis, A. R., Dolphin, A. E., Dalcanton, J. J., et al. 2015, ApJ, 805, 183
Lim, P. L., Diaz, R. I., & Laidler, V. 2015, pysynphot: Synthetic photometry software package, Astrophysics Source Code Library, ascl:1303.023
Maraston, C., Pforr, J., Renzini, A., et al. 2010, MNRAS, 407, 830
McKinney, W. 2010, in Proc. 9th Python Conf., Data Structures for Statistical Computing in Python, ed. S. van der Walt & J. Millman (Austin, TX: SciPy), 51
Mould, J. 2012, ApJL, 755, L14
Pérez, F., & Granger, B. E. 2007, CSE, 9, 21
Pforr, J., Maraston, C., & Tonini, C. 2012, MNRAS, 422, 3285
Salpeter, E. E. 1955, ApJ, 121, 161
Schlafly, E. F., & Finkbeiner, D. P. 2011, ApJ, 737, 103
Skilling, J. 2004, in AIP Conf. Proc. 735, 24th International Workshop on Bayesian Inference and Maximum Entropy Methods in Science and Engineering, ed. R. Fischer, R. Preuss, & U. von Toussiant (Melville, NY: AIP), 395
Sorba, R., & Sawicki, M. 2015, MNRAS, 452, 235
The Astropy Collaboration, Price-Whelan, A. M., Sipőcz, B. M., et al. 2018, AJ, 156, 123
The Astropy Collaboration, Robitaille, T. P., Tollerud, E. J., et al. 2013, A&A, 558, A33
Tonry, J., & Schneider, D. P. 1988, AJ, 96, 807
van der Walt, S., Colbert, S. C., & Varoquaux, G. 2011, CSE, 13, 22
van Dokkum, P., & Conroy, C. 2014, ApJ, 797, 56
Vogelsberger, M., Genel, S., Springel, V., et al. 2014, MNRAS, 444, 1518
Walcher, J., Groves, B., Budavári, T., & Dale, D. 2011, Ap&SS, 331, 1
Weisz, D. R., Dalcanton, J. J., Williams, B. F., et al. 2011, ApJ, 739, 5
Weisz, D. R., Dolphin, A. E., Skillman, E. D., et al. 2014, ApJ, 789, 147
Williams, B. F., Dalcanton, J. J., Dolphin, A. E., et al. 2015, ApJ, 806, 48
Williams, B. F., Dolphin, A. E., Dalcanton, J. J., et al. 2017, arXiv:1708.02617
Williams, B. F., Olsen, K., Khan, R., Pirone, D., & Rosema, K. 2018, ApJS, 236, 4