Publications of the Astronomical Society of Australia (2020), **37**, e016, 12 pages
doi:10.1017/pasa.2020.8

CAMBRIDGE UNIVERSITY PRESS

Research Paper

SIMSPIN—Constructing mock IFS kinematic data cubes

Katherine E. Harborne[1,2], Chris Power[1,2] and Aaron S. G. Robotham[1,2]

[1]International Centre for Radio Astronomy (ICRAR), M468, The University of Western Australia, 35 Stirling Highway, Crawley, WA 6009, Australia and [2]ARC Centre of Excellence for All Sky Astrophysics in 3 Dimensions (ASTRO 3D)

Abstract

We present SIMSPIN, a new, public, software framework for generating integral field spectroscopy (IFS) data cubes from *N*-body/hydrodynamical simulations of galaxies, which can be compared directly with observational datasets. SIMSPIN provides a consistent method for studying a galaxy's stellar component. It can be used to explore how observationally inferred measurements of kinematics, such as the spin parameter λ_R, are impacted by the effects of, for example, inclination, seeing conditions, distance. SIMSPIN is written in R and has been designed to be highly modular, flexible, and extensible. It is already being used by the astrophysics community to generate IFS-like cubes and FITS files for direct comparison of simulations to observations. In this paper, we explain the conceptual framework of SIMSPIN; how it is implemented in R; and we demonstrate SIMSPIN's current capabilities, providing as an example a brief investigation of how numerical resolution affects how reliably we can recover the intrinsic stellar kinematics of a simulated galaxy.

Keywords: galaxies: evolution – galaxies: kinematics and dynamics – methods: numerical – virtual observatory tools

(Received 3 February 2020; revised 7 March 2020; accepted 17 March 2020)

1. Introduction

Over the last few decades, we have seen vast improvements in our understanding of galaxy evolution by combining photometric measurements of galaxies with observed, projected, stellar kinematics. While photometry provides clues about a galaxy's assembly history, the inclusion of kinematics has revealed a whole new perspective that highlights the imprints of accretion and merger events. These imprints can be quantified and connected to the mass and environment of a galaxy. This new perspective has opened up new avenues to investigate the drivers of galactic evolution (Binney 2005; Emsellem et al. 2007; Cappellari et al. 2011; Cortese et al. 2016; van de Sande et al. 2017a). During the same period, advances in numerical simulations of galaxy formation and evolution have enabled comparable kinematic measurements to be made of simulated galaxies. This has provided a physically motivated framework to understand how galactic structure and kinematics are entwined and to interpret the astrophysical significance of observed kinematic signatures (Jesseit et al. 2009; Naab et al. 2014; Teklu et al. 2015; Lagos et al. 2018a).

As observations and simulations have grown in both scope and sophistication, the question of how to compare them in a faithful manner has become more important. The now well-established standard approach is to generate synthetic data products from theoretical datasets. This substantially reduces the inherent uncertainties in translating from an observed dataset to an estimate of the physical quantity of interest. The mock images produced can be passed through the same software tools that observers use to give consistent comparisons and also allows for the incorporation of observational limitations, such as the effects of the atmosphere that can artificially distort the observed line-of-sight (LOS) velocities. This approach is already being pursued by the SAMI (the Sydney-AAO Multi-object Integral field spectrograph survey; Croom et al. 2012; Bryant et al. 2015) and MaNGA (Mapping Nearby Galaxies at Apache Point; Bundy et al. 2015; Blanton et al. 2017) teams (see, e.g., Lagos et al. 2018b; Bassett & Foster 2019; Duckworth et al. 2020). This alone suggests that a tool for creating such data products in a publicly accessible and repeatable way is advantageous for the community.

Mock data products from large cosmological simulations such as MILLENIUM (Springel et al. 2005) and ILLUSTRIS (Vogelsberger et al. 2014) are available via corresponding 'observatories'—the Millenium Run Observatory (MRObs; Overzier et al. 2013) and the Illustris Simulation Observatory (Torrey et al. 2015). Tools such as Simulating IFU Star Cluster Observations (SISCO; Bianchini et al. 2015) have also been used to generate integral field spectroscopy (IFS) observations of globular clusters for exploring kinematic signatures of intermediate-mass black holes (De Vita et al. 2017). However, similar complex data products for galaxy-scale models can be difficult and time-consuming to produce, and so often the focus has been on bulk physical properties (Overzier et al. 2013), such as the angular momentum and structure of galaxies (Genel et al. 2015; Teklu et al. 2015; Pedrosa & Tissera 2015).

This approach is no longer viable, however; not only does it limit the complexity of the observational data that can be compared to, but it also limits how these data can be used to benefit theoretical modelling. Mock observations not only assist our interpretation of observable kinematics, but also allow us to tune our sub-grid physics models within simulations. To understand whether our models of feedback in simulations are sensible, we need to investigate the more detailed gas and stellar

Author for correspondence: Katherine E. Harborne, E-mail: katherine.harborne@icrar.org

Cite this article: Harborne KE, Power C and Robotham ASG. (2020) SIMSPIN—Constructing mock IFS kinematic data cubes. *Publications of the Astronomical Society of Australia* **37**, e016, 1–12. https://doi.org/10.1017/pasa.2020.8

© Astronomical Society of Australia 2020; published by Cambridge University Press

https://doi.org/10.1017/pasa.2020.8 Published online by Cambridge University Press

kinematics as well as compare to the cutting edge HI (Papastergis & Ponomareva 2017) and IFS surveys (van de Sande et al. 2019). Sub-grid recipes used in the latest galaxy formation simulations tend to be based on older stellar formation and feedback models, for example, ILLUSTRIS (Genel et al. 2014), and its successor ILLUSTRISTNG (Pillepich et al. 2018), models the star-forming inter-stellar medium (ISM) gas as an effective equation of state, first proposed by Springel & Hernquist (2003). This approach is common in simulations where ISM structure is below the resolution limit of the model (Ascasibar et al. 2002; Few et al. 2012). However, as simulations drive towards higher resolutions and start to model, for example, the star-forming ISM in more detail, the kinds of comparisons required to verify the utility of these models must similarly become more sophisticated.

These considerations have led us to develop SIMSPIN, a framework to allow for a fair comparison of simulated and observed datasets. SIMSPIN is a modular R-package that takes a particle model and creates a synthetic stellar-kinematic data cube from which we can generate mock flux, LOS velocity, and LOS velocity dispersion images using the specifications of any IFS. Observational effects, such as the resolution of the cube and distortions in the atmosphere, can be incorporated. From these images, we can study the specific effects that observing has on the kinematic properties recovered, given that we have access to the intrinsic properties of the model under scrutiny. Each simulated galaxy can be analysed many times from a range of projected distances and angles.

SIMSPIN is designed to be quick and repeatable, allowing a small number of models to produce a large number of observations. This is the first open-source package of its kind [registered with the Astrophysics Source Code Library (Harborne 2019)] and allows any astronomer to generate mock kinematic images for comparison with real observations. This code can work with simple N-body models, but also has facilities to incorporate simple stellar population (SSP) synthesis models for processing hydrodynamic simulations. All data products can be output in a FITS file format similar to what would be produced by an observation. Furthermore, it is written in a highly modular fashion that allows modifications and extensions to be easily added in the future, i.e., radiative transfer outputs, dust screens, telescope specifics, and further kinematic data manipulation.

The purpose of SIMSPIN is to ease the communication between practical and theoretical astronomers and accelerate our progress in understanding how specific stellar kinematic features evolve over time. Here, we present the initial framework for creating synthetic IFS kinematic data cubes from simulated galaxies. In Section 2, we briefly describe the methodology of the code and show how each function of the package can be implemented in Section 3. Full astronomical examples can be found in Section 4. Finally, we discuss possible extensions of this work and give a summary in Section 5.

2. Methodology

The purpose of the SIMSPIN package is to take a simulation of a galaxy and to produce a data cube corresponding to that which would be obtained if it had been observed using an IFS—spatial information in projection with kinematic information along the LOS. A kinematic data cube can be produced using the functions in this package, from which 'observables' can be measured and compared to the true (i.e., intrinsic) kinematic properties of the simulation.

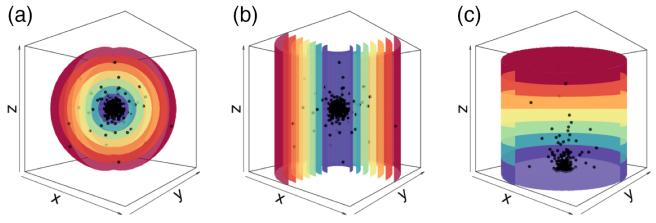

Figure 1. (a) `bin_type = "r"` Demonstrating the 3D spherical bins. (b) `bin_type = "cr"` The 2D circular annuli bins that spread out radially along the plane of the disc. (c) `bin_type = "z"` The 2D circular bins that grow in 1D out of the plane of the disc.

In this section, we present the methodology chosen to achieve this in a consistent and repeatable way:

1. Understand the true kinematics of the model.
2. Construct the simulation particle data into an 'observable format' and bin data into a 3D kinematic cube.
3. Convolve the data cube with a point spread function (PSF) in order to replicate the effects of the atmosphere.
4. Construct synthetic images from 3D mock data cube.
5. Calculate observable properties from the images produced (i.e., measure the effective radius of the galaxy, calculate the observable spin parameter within a given radius, etc.)

We will briefly address the approach to each of these matters in turn.

2.1. Understanding intrinsic model properties

It is necessary to understand the inherent nature of the galaxy model in question in order to assess how observation impacts kinematic measurements. Hence, SIMSPIN provides a method for analysing the phase space information of the particles within a simulation before constructing the mock observables.

Particle-based simulations provide the user with the particle IDs, positions (x, y, z), velocities (v_x, v_y, v_z), and masses. These properties are used within SIMSPIN to describe the intrinsic physical and kinematic profiles of the galaxy. To construct profiles, we take each particle and compute several additional properties. The particle phase space distribution is centred by subtracting the median in position and velocity space. The radial distribution of the physical properties can then be mapped. We add the spherical polar coordinates for each particle (r, θ, ϕ), the corresponding velocities (v_r, v_θ, v_ϕ), and components of the angular momentum (J_x, J_y, J_z).

Particles are divided into bins (spherical shells, cylindrical shells, or stacks—as shown in Figure 1) and the following properties are computed:

- the mass distribution,
- the log(density) distribution,
- the circular and rotational velocity distributions,
- the velocity anisotropy (β) distribution (Binney & Tremaine 2008),
- the Bullock spin parameter (λ) distribution, (Bullock et al. 2001).

https://doi.org/10.1017/pasa.2020.8 Published online by Cambridge University Press

These reflect the true nature of the system and allow us to explore the limitations of the synthetic observables, for example, when seeing conditions become more severe.

2.2. Creating the 'observable' format

In order to generate a projected galaxy image, as if the simulation is being observed in the sky, a few additional properties are added to each particle, such as the projected quantities of position and LOS velocity at inclination, i, to the observer.

$$z_{obs} = z\sin(i) + y\cos(i), \tag{1}$$

$$v_{los} = v_z\cos(i) - v_y\sin(i), \tag{2}$$

$$r_{obs} = \sqrt{x^2 + z_{obs}^2}, \tag{3}$$

where $i = 0°$ is the galaxy projected face on and 90° is edge on.

SIMSPIN then accounts for the physical properties of the observing telescope. Particulars such as the size and shape of the field of view and the size of the galaxy within that aperture are specified. Further instrument specifics, such as charge coupled device (CCD) noise and detailed fibre arrangements, are not included in the current implementation, but we intend to add these in later iterations of the code. Using the `celestial` package[a], we compute the angular diameter size, d_A, of the galaxy when projected at a supplied redshift distance using equation (4). The reference cosmology in this case is the most recent Planck data ($H_0 = 68.4$, $\Omega_M = 0.301$, $\Omega_L = 0.699$, $\Omega_R = 8.98 \times 10^{-5}$, $\sigma_8 = 0.793$; Planck Collaboration et al. 2018).

$$d_A = \frac{S_k(r)}{1+z}, \tag{4}$$

where r is the comoving distance and,

$$S_k(r) = \begin{cases} \frac{\sin(\sqrt{-\Omega_k}H_0 r)}{H_0\sqrt{|\Omega_k|}}, & \text{if } \Omega_k < 0 \\ r, & \text{if } \Omega_k = 0 \\ \frac{\sin(\sqrt{\Omega_k}H_0 r)}{H_0\sqrt{|\Omega_k|}}, & \text{if } \Omega_k > 0 \end{cases}$$

where $\Omega_k = 1 - \Omega_M - \Omega_L - \Omega_R$ is the curvature density and H_0 is the Hubble parameter today.

Using this, we determine how large the galaxy appears within the aperture. A selection of aperture shapes are available (circular, hexagonal, or square) to mimic the current layouts of modern IFS surveys, such as SAMI and MaNGA. The size of these apertures is user-defined and specified by the diameter in units of arc-seconds. We remove any particles belonging to the galaxy that fall outside of the imaged region.

Each remaining particle is then assigned a luminosity, L. This can be done by specifying a mass-to-light ratio for each luminous particle type (bulge, disc, or star) and scaling the luminosity to a flux, F, with respect to the luminosity distance at a given redshift, $D_L = \sqrt{L/4\pi F}$; alternatively, if the user has computed a spectrum for each stellar particle, these can also be supplied to the code and used to calculate more accurate fluxes within a chosen filter using PROSPECT[b] (Robotham et al. 2020), a high-level spectral generation package designed to create spectral energy distributions (SEDs) for the semi-analytic code, SHARK (Lagos et al. 2018c). PROSPECT combines stellar synthesis libraries, such as Bruzual & Charlot (2003) (BC03 hereafter) and/or EMILES (Vazdekis et al. 2016) with dust attenuation (Charlot & Fall 2000) and re-emission models (Dale et al. 2014). In this case, we use PROSPECT in a purely generative mode, using the SED generated for each stellar particle to calculate the flux contribution of each within a given filter.

The dimensions of the data cube are constructed to contain just the remaining luminous particles. We leave the specifics to the discretion of the user, for example, the apparent pixel size for SAMI data cubes is 0.5 arcsec with a spectral sampling scale of 1.04 Å (Green et al. 2018). These parameters are used to determine the widths of the bins in each direction. Physical pixel size is computed by multiplying the spatial sampling scale by the angular diameter scale calculated above, as this is dependent on the distance at which the galaxy is projected; the velocity pixel size is approximated by $v \sim c\Delta\lambda/\lambda$, where we take the central wavelength of the filter to be λ and the spectral scale as $\Delta\lambda$. Particles are filtered into their correct positions within the position-velocity cube and output as a 3D array.

At this stage, we make a key assumption that each particle in the simulation has some inherent uncertainty in its velocity. This mimics the idea that, when astronomers observe emission lines, those lines have a width representing an uncertainty in the true speed of the host environment where the line originated. This uncertainty is encapsulated numerically by the line spread function (LSF), `lsf_fwhm`, which is caused by a spectral response of the observing telescope to a point like source. We use the LSF to fix the 'width' of the particle's velocity. As the LSF of IFS instruments can be well approximated as Gaussian, we model the velocity of each particle as a Gaussian centred on the known velocity of the simulated particle with a width corresponding to the LSF associated with the mock observation telescope (van de Sande et al. 2017a).

Each particle's associated Gaussian is scaled by the flux of that particle and then summed with portions of each distribution contributing to several bins in velocity space. We fully bin the particles in this manner within both projected spatial coordinates and velocity space to construct the IFS kinematic data cube. An infographic of this process is shown in Figure 2.

These arrays can be output in FITS file format or passed to further functions for the addition of atmospheric effects and kinematic analysis.

2.3. Mimicking the effects of the atmosphere

Ground-based optical observations are limited by the blurring effects of our atmosphere. In order to compare like-for-like, we replicate these 'beam smearing' effects within our synthetic observations. SIMSPIN does this by convolving each spatial plane within the data cube with a PSF.

The user can specify the shape of this PSF—either a Gaussian or Moffat kernel (Moffat 1969)—and the full-width half-maximum (FWHM) of the kernel. Each x–y spatial plane is convolved with the generated PSF, using functions from PROFIT (Robotham et al. 2017) which follow the method:

$$F_{obs} = F_i \circledast \text{PSF}, \tag{5}$$

where F_i is the flux within each pixel in each spatial plane, i, and $\circledast$ represents convolution.

[a] https://CRAN.R-project.org/package=celestial
[b] https://github.com/asgr/ProSpect

https://doi.org/10.1017/pasa.2020.8 Published online by Cambridge University Press

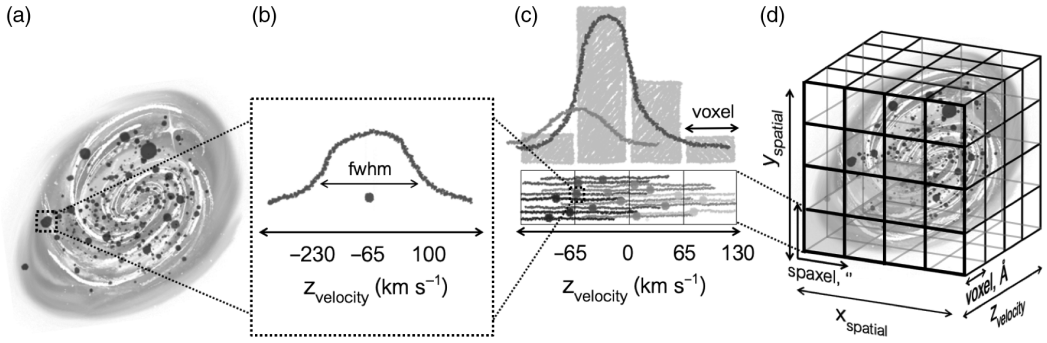

Figure 2. Illustrating the method in which each kinematic data cube is constructed. (a) We take each particle within the simulation—which will have some known velocity along the projected LOS—and (b) convolve each with a Gaussian kernel such that it has a velocity distribution with width dictated by the LSF. (c) This velocity distribution is then binned in velocity space along each spatial pixel such that a single particle can occupy several velocity bins. (d) Each pixel is then arranged in the cube to reconstruct the galaxy image.

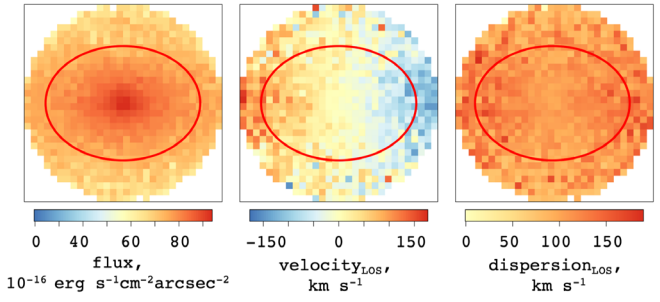

Figure 3. Demonstrating the mock images produced through SimSpin observations of the S0 example model inclined to 70° with added Sky RMS noise. The red line demonstrates 1 R_{eff}, within which λ_R is measured.

2.4. Constructing synthetic images

Having generated a realistic kinematic data cube, SimSpin can be used to process images and observable kinematics. Flux images and maps of the LOS velocity and velocity dispersion are generated by collapsing the cube along the z-axis.

To generate the flux maps, the contribution of flux from each velocity plane is summed, F_i:

$$F = \sum_{i=1}^{v_{\max}} F_i, \tag{6}$$

where $v_{\max}$ is the last velocity bin along that pixel in the cube. The LOS velocity and LOS velocity dispersion are given by flux-weighted statistics:

$$V = \frac{\sum v_i \times F_i}{\sum F_i}, \tag{7}$$

$$\sigma = \frac{\sum F_i \times (v_i - V)^2}{\sum F_i}. \tag{8}$$

Here, v_i is the velocity assigned to each velocity bin, i, weighted by the flux in each bin, F_i, and V is the mean velocity along that pixel in the cube given by equation (7). An example of such images can be seen in Figure 3.

Sky noise can optionally be added to the images using a sample of random, normally distributed values. The appropriate level is determined using the specified magnitude threshold, zero point, and spatial pixel scale. It is possible to export these images for Voronoi binning. While this is not supported at this time within the SimSpin code, Cappellari & Copin (2003) provides a standard method for binning these images so that the signal to noise is consistent across pixels. Vorbin[c] is a Python code that can be downloaded directly from PyPi. In future versions of this code, we intend for re-binned images to be added back into SimSpin for calculating the observable properties.

2.5. Calculating observable properties

The synthetic images can then be used to calculate various observational kinematic properties of the galaxy in question. SimSpin has been used to investigate the observable spin parameter, λ_R (Emsellem et al. 2007; Harborne et al. 2019), and can further evaluate the V/σ parameter (Cappellari et al. 2007).

The user can specify the radius within which the kinematic measurements are made. Often these are made within an effective radius, $R_{\rm eff}$, but we give the user the freedom to fully specify the size and ellipticity of this measurement radius. Either, the second-order moments are calculated from the flux distribution by diagonalising the inertia tensor and assuming the galaxy ellipticity from this axial ratio, q. This ellipse is then grown from the centre until half the total flux is contained within the radius. Alternatively, software like ProFound[d] can be used to generate concentric isophotes that contain equal amounts of flux within each (Robotham et al. 2017). This axial ratio information can be used to specify the ellipse within which the kinematics will be calculated. Only the pixels whose midpoints are contained within this ellipse will be used for further calculations.

Currently, it is possible to calculate two kinematic properties: λ_R (Emsellem et al. 2007) and V/σ (Cappellari et al. 2007). λ_R is calculated using equation (9):

$$\lambda_R = \frac{\sum_{i=1}^{n_p} F_i R_i |V_i|}{\sum_{i=1}^{n_p} F_i R_i \sqrt{V_i^2 + \sigma_i^2}}, \tag{9}$$

where F_i is the observed 'flux' taken from the flux image, R_i is the circularised radial position, V_i is the LOS velocity taken from the LOS velocity image, σ_i is the LOS velocity dispersion taken from the LOS dispersion image per pixel, i, and summed across the total number of pixels, n_p. We also give the option to compute the parameter using the elliptical radius where we define R_i^{ϵ} as the semi-major axis of an ellipse that would pass through that pixel.

[c] http://www-astro.physics.ox.ac.uk/ mxc/software/
[d] https://github.com/asgr/ProFit

https://doi.org/10.1017/pasa.2020.8 Published online by Cambridge University Press

Similarly, V/σ is calculated as:

$$V/\sigma = \sqrt{\frac{\sum_{i=1}^{n_p} F_i V_i^2}{\sum_{i=1}^{n_p} F_i \sigma_i^2}}. \quad (10)$$

SimSpin computes these parameters in a consistent manner to observable software, such as pPXF (Cappellari 2017), for simple and consistent comparison with real observations.

3. Implementation

Having outlined the methodology, we present the fully documented and tested R-package, SimSpin. This code can be downloaded from the github repository[e] and the package can be installed directly into R using the following commands:

```
> install.packages("devtools")
> library(devtools)
> install_github("kateharborne/SimSpin")
```

To load the package into your R session,

```
> library(SimSpin)
```

In Figure 4, we show the designed flow of the code. While it is possible to use each sub-function within this package independently and examine the output at each stage, there are three basic analysis functions designed to give the output information in a user friendly format.

1. `sim_analysis()`—This function outputs the inherent kinematic properties of the galaxy model. This provides the comparison to the kinematic observables produced in the following functions.
2. `build_datacube()`—This function produces the kinematic data cube prior to kinematic analysis. This allows the user to take the cubes to use in some other form of analysis without having to calculate λ_R.
3. `find_kinematics()`—This function produces a kinematic data cube and calculates the observed spin parameter, with both circularised and elliptical radii, V/σ, ellipticity, inclination and the corresponding flux, LOS velocity and LOS velocity dispersion images. For individual λ_R or V/σ, we provide two functions (`find_lambda()`/`find_vsigma()`) that output their named kinematics.

For further information about the implementation of these functions, we direct the reader to the repository and to RPubs where we present a series of vignette examples[f]. Each function is fully documented with an demonstrated example.

4. Examples

In this section, we demonstrate two ways in which this package may be used. Section 4.1 shows the simple analysis of an N-body model containing disc and bulge components and examines the effect of particle resolution on the profiles recovered. Section 4.2 expands further on this exercise to examine a hydrodynamic EAGLE simulation with stellar particles and histories.

4.1. N-body model example

Here, we use SimSpin to measure the inherent kinematics and the observable spin parameter for five different N-body realisations of galaxies. We use repeated observations to explore how the particle resolution of an N-body model can impact the precision and reliability of the physical and observable kinematics recovered.

These models have been constructed in two phases: first, initial conditions are generated using GalIC (Yurin & Springel 2014) which constructs isolated particle distributions in equilibrium by solving the collision-less Boltzmann equation; we then evolve these initial conditions using a modified version of GADGET-2 (Springel 2005) in which the 'live' dark matter (DM) halo is replaced with its 'static' analytical form. This is done to maintain a stable, well-resolved galactic disc at reasonable computational cost. The details of this simulated catalogue can be found in Table 1 and is described in greater detail within (Harborne et al. 2019).

First, we convert the simulation snapshots into SimSpin compatible HDF5 input files[g] which can be read into R using the SimSpin function `sim_data()`, and the output data frame is processed using `sim_analysis()` and `find_lambda()`. In each case, we examine our models at full particle resolution initially and assume this to be an ideal case by which we benchmark.

4.1.1. Simulation properties

We begin by examining the kinematic properties inherent to the simulated models. This is done using the code below. First we load in the simulation data, considering all components and then the disc and bulge components separately.

```
> all_data = sim_data("S0.hdf5")
> disc_data = sim_data("S0.hdf5", ptype=2)
> bulge_data = sim_data("S0.hdf5", ptype=3)
```

Next, we run the `sim_analysis()` function for each loaded dataset, supplying the information about the DM profile that has been removed throughout the evolution and the number of radial bins that we wish to examine the profile across. If DM particles were present in the supplied file, the DM profile parameter would not be necessary.

```
> all_analysis = sim_analysis(all_data, rbin =
1 000, DM_profile = list(profile="Hernquist",
DM_mass=184.996, DM_a=34.5))
> disc_analysis = sim_analysis(disc_data, rbin
= 1 000, DM_profile = list(profile="Hernquist",
DM_mass=184.996, DM_a=34.5))
> bulge_analysis = sim_analysis(bulge_data, rbin
= 1 000, DM_profile = list(profile="Hernquist",
DM_mass=184.996, DM_a=34.5))
```

There are several outputs from the `sim_analysis()` function, as described in Section 2. Some of these are shown in Figure 5. We demonstrate how simply we can examine different simulation components in the top panel, where the mass profile of the disc and bulge are plotted separately for each galaxy in the catalogue.

[e] https://github.com/kateharborne/SimSpin
[f] https://rpubs.com/kateharborne
[g] https://github.com/kateharborne/create_SimSpinFile

https://doi.org/10.1017/pasa.2020.8 Published online by Cambridge University Press

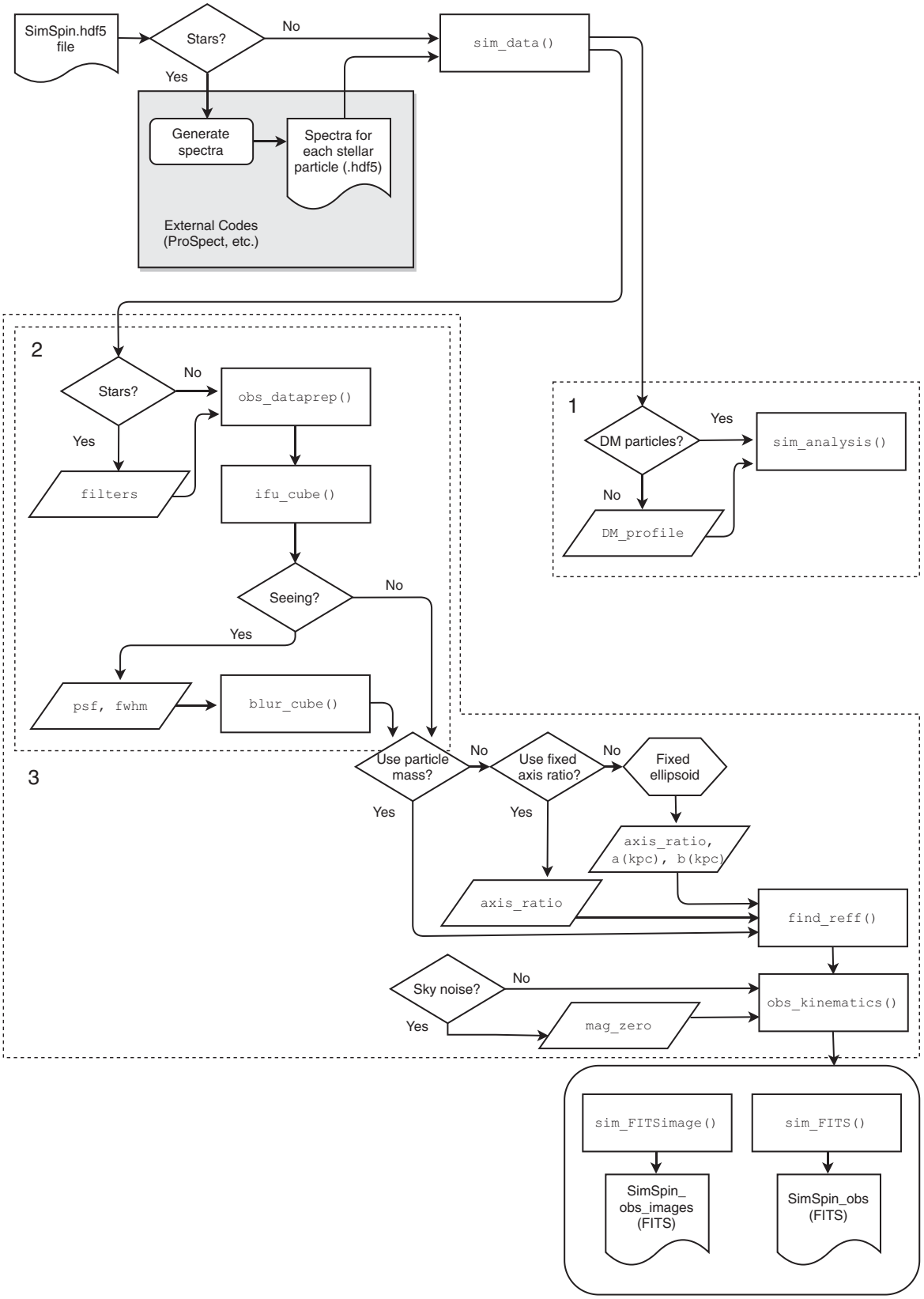

Figure 4. Demonstrating the individual functions and queries used when running a simulated galaxy through the SimSpin package. Three over-arching functions are identified that link these sub-functions together: (1) `sim_analysis()` - as explained in Section 2, (2) `build_datacube()` - as explained in Section 2.1 and (3) `find_kinematics()` - as explained in Section 2.4

https://doi.org/10.1017/pasa.2020.8 Published online by Cambridge University Press

Table 1. Outlining the properties of each N-body galaxy model in the catalogue explored throughout the examples in Section 4.1.

	B/T	b/kpc	n	N_{disc}	N_{bulge}
S0	0.60	2.14	2.84	1 000 000	1 500 000
Sa	0.40	1.38	2.26	1 500 000	1 000 000
Sb	0.25	0.90	1.64	1 875 000	625 000
Sc	0.05	0.17	0.99	2 375 000	125 000
Sd	0.02	0.07	0.97	2 450 000	50 000

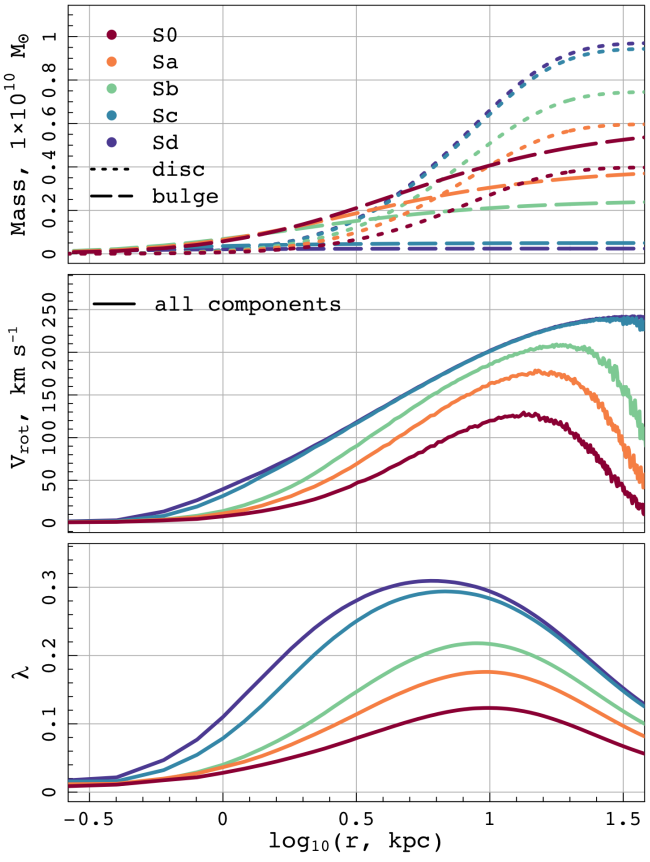

Figure 5. Showing a selection of the profiles provided by the `sim_analysis()` function: Mass (top), rotational velocity (middle), Bullock spin parameter (bottom). We demonstrate the flexibility of the code in analysing galaxy components separately.

We have reduced the resolution of our simulations in order to examine what effect this has on the recovered kinematics. For each test, we have rerun the GALIC initial conditions of the galaxy but with a fraction of the original number of particles. These have then been evolved for 10 Gyrs using GADGET-2. We have considered a further 12 iterations of each galaxy in the catalogue, from 0.01% N_{total} increasing in increments to the full N_{total}. This gives a sample of 65 simulations to analyse overall.

The results of this experiment are shown for the measured Bullock spin parameter in Figure 6 for the Sa galaxy. As we expect, the lower the resolution, the poorer our results become. Clearly, for the lowest resolution considered, at 250 particles, we have very noisy variations ($\pm$0.1) between measured lambda and the benchmark measurement made at full resolution. However, these variations are much less significant at 10% N_{total}, where we only see $\pm$0.005 and only within the inner radii. In all but the worst

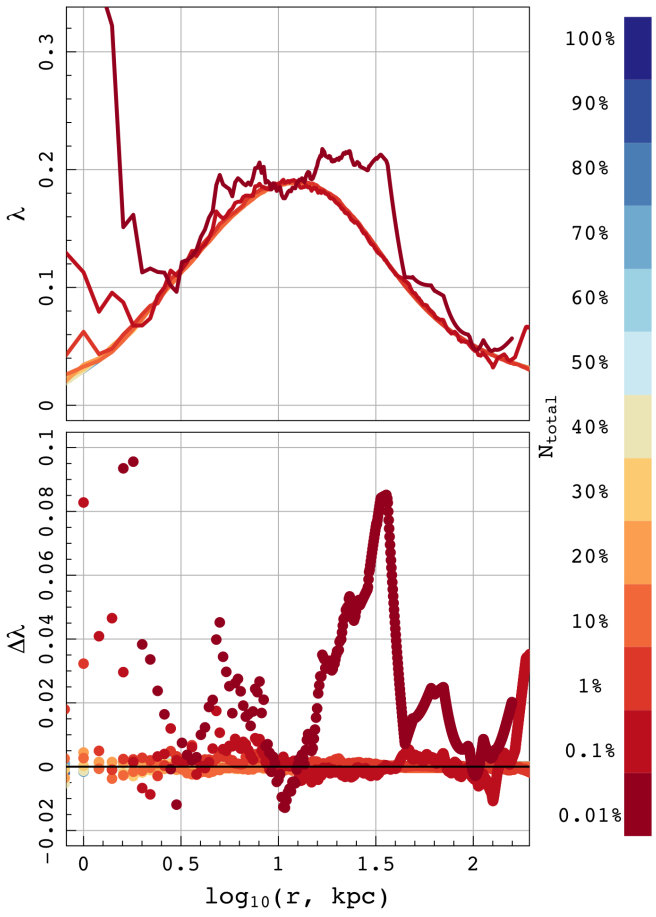

Figure 6. Demonstrating the effect of reduced particle resolution on the recovery of the Bullock spin parameter, λ. In the upper panel, we show the spin parameter radial profile for the Sa galaxy at 13 different resolutions, as described by the colours shown on by the colour bar on the right. The percentages describe the fraction of the N_{total} particles included in each simulation. Residuals from the 100% N_{total} case are shown in the lower panel.

case, the expected shape of the lambda profile is still recovered. The effect of particle resolution is akin to adding numerical noise. While the general trend is returned, we see the effective signal-to-noise drop as the number of particles used to describe the mass distribution is reduced. It is interesting to investigate how the observable kinematics are affected by similar reductions in resolution.

4.1.2. *Synthetic observable properties*

Using the `find_lambda()` function, we can generate synthetic observations of our models and measure kinematic properties. The following code generates images as if our simulations were observed using the blue arm of SAMI, which is the default output of the SIMSPIN observation functions. The projected distance, z, is set to 0.05 and the projected inclination for these observations is 70°.

```
> all_data = sim_data("S0.hdf5")
> lambda = find_lambda(all_data)
```

We run this function on the 12 degraded iterations of each galaxy model in the catalogue, examining each spin parameter at resolutions from 0.01% to 100% of the benchmark. In each case,

https://doi.org/10.1017/pasa.2020.8 Published online by Cambridge University Press

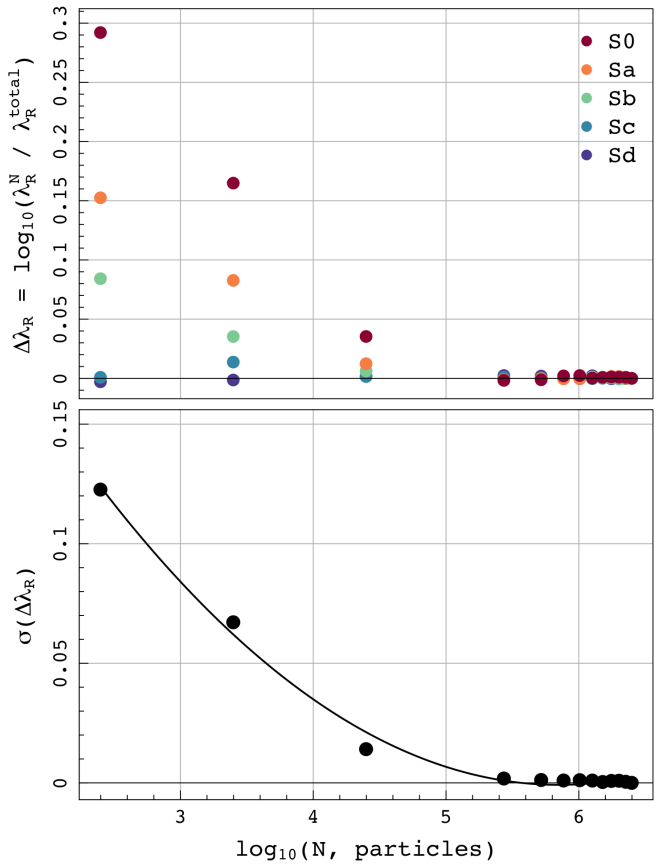

Figure 7. Considering how the observed spin parameter, λ_R, changes when the number of particles within the simulated model is reduced from full resolution down to 0.01% N_{total}. In the upper panel, we consider the log difference between λ_R at that resolution with respect to the value measured at the benchmark resolution. In the lower panel, we measure the scatter of those measurements at each resolution and find that it is well fit by an exponential profile.

the recovered λ_R is plotted against the particle resolution; this is shown in Figure 7. In the upper panel of the figure, we examine the log difference $\Delta\lambda_R = \log_{10}(\lambda_R^N/\lambda_R^{total})$, i.e., between the value of λ_R observed at a degraded resolution and the value measured at the highest resolution. In the lower panel of this figure, we consider the spread of the $\Delta\lambda_R$ values at each resolution. Within each bin we measure the standard deviation, σ, in order to give an indication of how confident we are that the model returns the correct λ_R. This assumes that the full resolution, 2.5×10^6-particle model is the 'true' value.

From the lower panel in Figure 7, we can see that at resolutions lower than 5×10^5 particles, the measure becomes exponentially more noisy. At 0.01% N_{total} particles, σ peaks at ∼0.12. However, the change in $\sigma(\Delta\lambda_R)$ is quite sudden, when the number of particles in the model approaches 5×10^5. At higher resolutions than this, we see that the benchmark λ_R is recovered with a precision of ∼0.002, which is much lower than normal observational uncertainties. The assumption that the 2.5×10^6 particle model returns a benchmark value also seems reasonable. The fairly flat trend in $\sigma(\Delta\lambda_R)$ from 30% to 100% N_{total} suggests that we have reached an asymptotic value.

We also see that as the number of particles drops, the morphology of the model becomes important for the associated uncertainty. At 0.01%, the S0 galaxy has the greatest uncertainty. This makes sense when you consider that dispersion will dominate in the bulge component; reducing the number of bulge particles that sample this velocity distribution will have a greater impact on the measurement of σ, and hence λ_R and so we see the largest uncertainties in galaxies with larger bulge components.

Overall, this demonstrates that galaxies represented by smaller numbers of particles will give less accurate measurements, though in this example the uncertainty on the measurement at N > 250 000 is ∼0.002, increasing to ∼0.02 as we approach smaller N, ∼25 000. In reality, even at this number of particles, the related uncertainty is still much smaller than the uncertainty generally associated with observational limitations (i.e., seeing and beam smearing) (D'Eugenio et al. 2013; van de Sande et al. 2017a, 2017b, Greene et al. 2018; Graham et al. 2018; Harborne et al. 2019).

4.2. Full hydrodynamical model example

It is important to understand how uncertainties are associated with particle resolution because when we examine galaxies from larger cosmological models, the number of particles making up individual galaxies tends to be lower. In the following example, we use a small galaxy from the EAGLE simulation, `GalaxyID = 1056` taken from the `RefL0100N1504` simulation. As a proof of concept, we will demonstrate that we can process these galaxies using the same functions as the simple N-body models in Section 4.1.

The EAGLE project (Schaye et al. 2015; Crain et al. 2015; McAlpine et al. 2016) is a suite of cosmological hydrodynamical simulations that allow us to investigate the formation and evolution of galaxies. We have been using the publicly available `RefL0100N1504` simulation run; this is a cubic volume with a side length of 100 co-moving Mpc, of intermediate resolution with initial baryonic particle masses of $m_g = 1.81 \times 10^6$ $M_\odot$, and maximum gravitational softening lengths of $\epsilon_{\rm prop} = 0.70$ pkpc.

The galaxy chosen, `GalaxyID = 1056`, contains 21 174 stellar particles. In EAGLE, each stellar particle is initialised with a stellar mass described by the Chabrier (2003) initial mass function; their metallicities are inherited from their parent gas particle and their ages recorded from their formation to current snapshot time. We can use this information to generate particle luminosities assuming a SSP. When providing SIMSPIN with stellar particles, it is optional as to whether you would like to provide SEDs generated using stellar population synthesis models or just assume a mass-to-light ratio for all stellar particles. For this example, we have used PROSPECT to generate SEDs for each stellar particle by interpolating the BC03 tables in both age and metallicity. A subset of these generated distributions are shown in Figure 8.

We store these SEDs in a SIMSPIN compatible stellar file and provide this information along with the simulation data. This stellar file is in an HDF5 format that contains a `PartType4` group with two datasets, `Luminosity` and `Wavelength`. When loading this data into SIMSPIN, we specify this stellar file as below.

```
> eagle_data = sim_data("eagle.hdf5",
SSP="eagle_stars.hdf5")
```

This prepares a structure that contains the luminosity at a range of wavelengths for the stellar particles. We then pass this data structure on to the `find_vsigma()` function as we would any other `sim_data()` output.

https://doi.org/10.1017/pasa.2020.8 Published online by Cambridge University Press

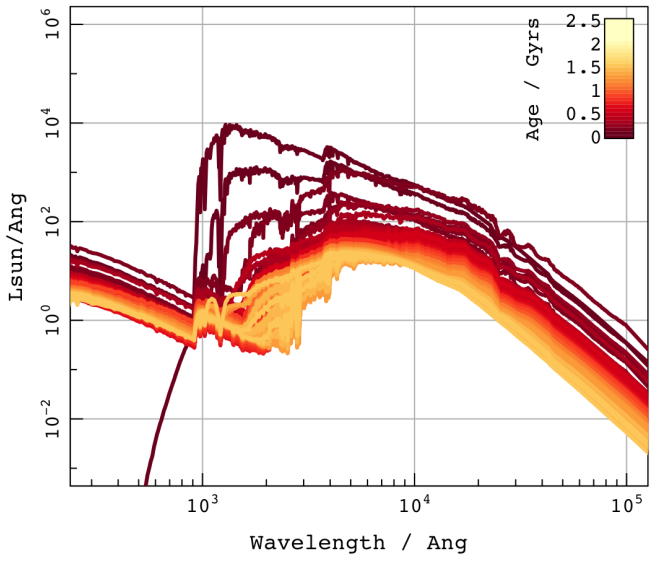

Figure 8. Showing the SEDs generated using ProSpect for 500 of the stellar particles within Eagle GalaxyID = 1056. The colour of each line reflects the age of the stellar particle in Gyr. These particles have been randomly selected from the 21 174 within the model to show a representative number of the stellar distributions in this galaxy.

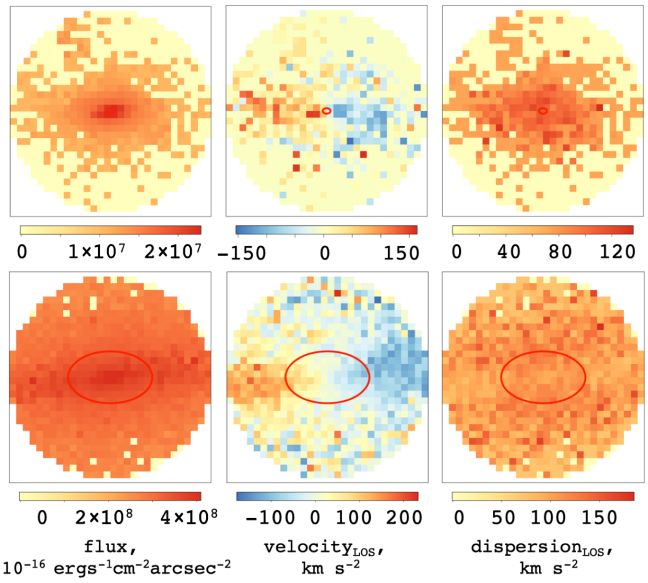

Figure 9. The synthetic observations of the Eagle galaxy (GalaxyID = 1056) inclined edge-on, produced by SimSpin at a projected redshift distance $z = 0.0005$ (above) and $z = 0.00005$ (below).

```
> eagle_vsigma = find_vsigma(eagle_data,
z=0.0005, inc_deg=90)
```

The presence of these luminosity/wavelength tables means that, once passed to the SimSpin analysis functions, the particle fluxes will be calculated using ProSpect within a given filter. In this case, given that the central wavelength of the blue arm of SAMI (∼4 800 Å), we use the Sloan Digital Sky Survey g-filter to assign a flux to each particle at the specified redshift. Currently, this process is the most computationally expensive stage because the flux calculation takes on average ∼0.004 s per particle. When increasing the number of particles above the small number in this simulation, this becomes the main source of computation, mostly due to the large arrays of luminosity and wavelength for each particle. For the purpose of proving that SimSpin can perform with hydrodynamic simulations, we believe this to be sufficient, but highlight this as an area for further work that would benefit from parallelisation.

Because the galaxy in question is very small ($R_{eff} \sim 0.003$ kpc), we have had to put this galaxy very nearby in order to generate a suitable image. We show the images produced in Figure 9. Inclining the galaxy to 90°, we see that there is a rotating disc component present. However, the image resolution is insufficient to make a measurement of V/σ within a half-mass radius, as this makes up less than 4 pixels at the centre of the image. If instead, we bring the galaxy closer again:

```
> eagle_vsigma_close = find_vsigma(eagle_data,
z=0.00005, inc_deg=90),
```

we now recover the $V/\sigma = 0.46$. Both projections are shown in Figure 9, with the fitted half-mass radius shown.

This is another way in which resolution can effect our recovery of the internal kinematics. With instruments like MUSE, there are a large number of poorly resolved objects at distant redshifts for which we have a few pixels worth of kinematic information (Guérou et al. 2017). Using SimSpin, we can examine what effect this has on the recovered kinematics. With this in mind, we varied the projected distance of the galaxy inclined edge on from $z = 0.00001$ to the point at which the measurement radius is too small to return a value (i.e., at the point when the semi-major axis of the measurement ellipse is less than the size of one pixel in the image). We present this in units of cz for clarity (i.e., from $cz = 3\,000 - 93\,000$). This is shown in Figure 10.

There are several interesting features in this plot. Firstly, there are large discontinuities where the V/σ measure drops suddenly. Within each of the discrete chunks, there is a gradual positive inclination of points. To explain these features, we have looked more closely at the geometry of the images being produced for each measurement.

Throughout this experiment, we have fixed the shape of the measurement ellipse in physical space ($a = 0.003$ kpc and $b = 0.002$ kpc) as was measured using the second-order moments of the galaxy projected at 90°. Therefore, the only thing changing as we modify the projected redshift distance is the size of the galaxy within the aperture. As the galaxy is moved further and further from the telescope, a smaller number of pixels fit inside the measurement ellipse. Also, each pixel begins to contain more particles within that system, which explains the gradual increases we see within the discrete bins; to explain the discrete bins themselves, we first considered the number of pixels that are contained within the ellipse in each case. As can be seen by the shaded blocks behind the measurements, the jumps occur at the transitions when the ellipse crosses the midpoints of new pixels to be included in the measurement.

Finally, we see that the jumps can be positive or negative, and that in general these jumps become larger the further away we go. To understand this effect, we looked at the arrangement of additional particles in each of three cases. In the first, we consider the transition from point 1 to 2, as shown in Figure 10. Here we see V/σ drop considerably from 0.48 to 0.35; the corresponding maps produced show that at this point, we go from 16 to 12 pixels, and specifically we loose pixels along the semi-major

https://doi.org/10.1017/pasa.2020.8 Published online by Cambridge University Press

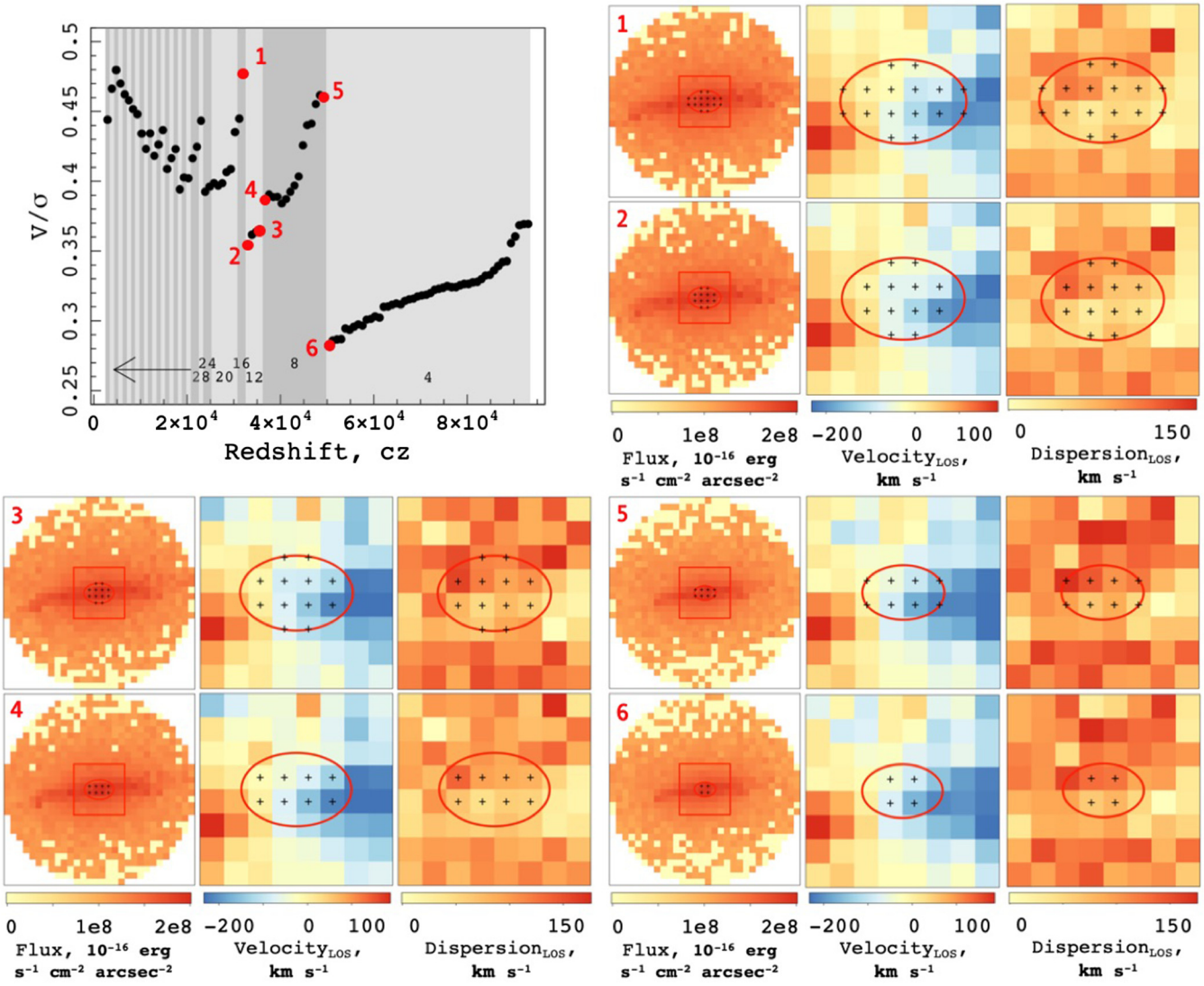

Figure 10. Investigating the scatter in the measurement of V/σ with 100 redshift distances from $z = 0.00001$ to 0.00031 for the EAGLE galaxy (`GalaxyID = 1056`).

axis. We have aligned the major velocity axis horizontally, so the removal of the pixels in these positions reduces the velocity component much more than the dispersion component. Therefore, in a relative sense, the dispersion becomes larger and V/σ drops. Conversely, if we remove pixels from the vertical semi-minor axis, we reduce the dispersion component relative to the velocity and so V/σ rises; this is shown in the highlighted 3 to 4 points. The most significant drop is between points 5 and 6. At this stage, we are hitting the limit of our resolution because we cannot calculate V/σ with fewer than 4 pixels. This jump is the largest because more particles are contained within these 4 pixels than at the closer projections.

This result will be dependent on several other factors, such as the inclination and morphology of the galaxy being observed. The magnitude of these jumps will also be changed by the manner in which you choose to centre your pixels (e.g., if the centre of your measurement ellipse is at the centre of a single pixel), although we would still expect to see these jumps due to the discrete nature of pixel resolution. In all cases, this is pause for thought when calculating kinematic measures with only a very small number of pixels at the centre of your system.

5. Further work and conclusions

We have introduced SIMSPIN, a flexible and versatile framework for generating and analysing IFS data cubes from N-body/hydrodynamical simulations, which can be compared directly with observations from surveys such as SAMI and MaNGA.

We offer this current version of SIMSPIN via direct download from GitHub or a web application[h]. The benefit of using `R` is the ease with which an app can be created, and the `R Shiny` application makes it simple to generate a standard `JavaScript`, web

[h]The app is housed on the Nimbus server at the Pawsey Supercomputing Centre, http://simspin.icrar.org/

https://doi.org/10.1017/pasa.2020.8 Published online by Cambridge University Press

browser compatible Graphical User Interface. The current version of this web app is designed for simplicity, and so there are fewer code options, described by a series of drop down panels and slider bars, than available via the standard R interface. The outputs generated are basic, but allow you to intuitively explore how observational effects such as seeing conditions and projected distance affect the observable kinematics. This application can also be downloaded to your own system and run from there, available on GitHub[i].

There are several aspects that we wish to develop further and in this section we will outline a few of these ideas.

In the near term, we will

- increase the efficiency and reduce the run time of the conversion between intrinsic spectrum and flux, which currently requires a large portion of computation time, as highlighted in the EAGLE galaxy example in Section 4.2.
- add further kinematic observables, such as higher order Gauss–Hermite coefficients, h_3 and h_4, and output images, such as gas maps as we continue to implement improvements for processing hydrodynamical simulations.

In the longer term, we plan to:

- offer mock data cube outputs in a format that more closely mimics their observed counter parts, that is, cubes with (x, y, λ) rather then the current $(x, y, v_{\rm LOS})$. With the progression of tools such as PROSPECT, each particle in a model can be associated with a spectra, as demonstrated in Section 4.2. With these spectra used to describe the third dimension of the mock-cube, we can generate a product that can be fully processed using observational pipelines.
- further signal-to-noise treatments could be implemented at the level of the individual spectra. Signal-to-noise values are important for observational kinematic measurements and the current implementation of SIMSPIN does not yet incorporate this factor in a physically meaningful way.
- additional instrument specifics and uncertainties can be implemented—CCD read-out noise, fibre arrangements, dithering patterns, etc.
- provide multi-language implementations of SIMSPIN, in, for example, `python` and `Julia`. Such versions are under development. Ideally, we would like several wrappers to allow different users to generate the same results easily from different platforms, such as a `python` wrapper using the `r2py` interface.

Finally, one of the key requirements of this code has been that it should be easily extendable. All of the software is freely available through the `GitHub` repository and, with the use of this paper, examples within the package and `RPubs`, we hope that general users can extend this work to address many questions beyond those we have considered here.

In conclusion, we have demonstrated the versatility and simplicity of the SIMSPIN code. While this tool was initially created to measure the observational effects that come into play when measuring kinematics such as λ_R, its extensibility makes the scope of this tool enormous. We have demonstrated examples in which we consider how the resolution of simulated models and the resolution of the observing instrument used may impact measurements of galactic kinematics. These two examples barely scratch the surface of all the parameter variations that could be considered.

We believe that SIMSPIN has uses within both the theoretical and observational astronomical communities. The code has the ability to produce synthetic data products in FITS file format that allows direct comparison between simulations and observations. This code has already been used to examine how the observed kinematics vary with seeing conditions (Harborne et al. 2019), and further projects include using synthetic SIMSPIN observations to design an empirical correction to counteract these effects (Harborne et al. submitted). We present version 1.1.1 with the capabilities outlined within this paper and invite further collaboration to extend the reach of this tool.

Acknowledgements. We would like to thank the anonymous referee for their kind and prompt review of this work. KH is supported by the SIRF and UPA awarded by the University of Western Australia Scholarships Committee. CP and AR acknowledge the support of ARC Discovery Project grant DP140100395. Parts of this research were also conducted by the Australian Research Council Centre of Excellence for All Sky Astrophysics in 3 Dimensions (ASTRO 3D), through project number CE170100013. This work was supported by resources provided by the Pawsey Supercomputing Centre with funding from the Australian Government and the Government of Western Australia. We acknowledge the Virgo Consortium for making their simulation data available. The EAGLE simulations were performed using the DiRAC-2 facility at Durham, managed by the ICC, and the PRACE facility Curie based in France at TGCC, CEA, Bruyèresle-Châtel.

References

Ascasibar, Y., Yepes, G., Gottlöber, S., & Müller, V. 2002, A&A, 387, 396
Bassett, R., & Foster, C. 2019, MNRAS, 487, 2354
Bianchini, P., Norris, M. A., van de Ven, G., & Schinnerer, E. 2015, MNRAS, 453, 365
Binney, J. 2005, MNRAS, 363, 937
Binney, J., & Tremaine, S. 2008, Galactic Dynamics. Princeton Series in Astrophysics (2nd edn.; Princeton University Press)
Blanton, M. R., et al. 2017, AJ, 154, 28
Bruzual, G., & Charlot, S. 2003, MNRAS, 344, 1000
Bryant, J. J., et al. 2015, MNRAS, 447, 2857
Bullock, J. S., Dekel, A., Kolatt, T. S., Kravtsov, A. V., Klypin, A. A., Porciani, C., & Primack, J. R. 2001, ApJ, 555, 240
Bundy, K., et al. 2015, Overview of the SDSS-IV MaNGA survey: Mapping Nearby Galaxies at Apache Point Observatory, 10.1088/0004-637X/798/1/7
Cappellari, M. 2017, MNRAS, 466, 798
Cappellari, M., & Copin, Y. 2003, MNRAS, 342, 345
Cappellari, M., Emsellem, E., Bacon, R., et al. 2007, MNRAS, 379, 418
Cappellari, M., Emsellem, E., Krajnovic, D., et al. 2011, MNRAS, 416, 1680
Chabrier, G. 2003, PASP, 115, 763
Charlot, S., Fall, S. M. 2000, AJ, 539, 718
Cortese, L., Fogarty, L. M. R., Bekki, K., et al. 2016, MNRAS, 463, 170
Crain, R. A., et al. 2015, MNRAS, 450, 1937
Croom, S. M., et al. 2012, MNRAS, 421, 872
D'Eugenio, F., Houghton, R. C. W., Davies, R. L., et al. 2013, MNRAS, 429, 1258
Dale, D. A., Helou, G., Magdis, G. E., Armus, L., Díaz-Santos, T., & Shi, Y. 2014, AJ, 784, 83
De Vita, R., Trenti, M., Bianchini, P., Askar, A., Giersz, M., & van de Ven, G. 2017, MNRAS, 467, 4057
Duckworth, C., Tojeiro, R., & Kraljic, K. 2020, MNRAS, 492, 1869
Emsellem, E., Cappellari, M., Krajnovic, D., et al. 2007, MNRAS, 379, 401
Few, C. G., Courty, S., Gibson, B. K., Kawata, D., Calura, F., & Teyssier, R. 2012, MNRAS, 424, L11
Genel, S., et al. 2014, MNRAS, 445, 175

[i]https://github.com/kateharborne/SimSpin_app

Genel, S., Fall, M., Hernquist, L., et al. 2015, ApJ, 804, 7pp
Graham, M. T., Cappellari, M., Li, H., et al. 2018, MNRAS, 477, 4711
Green, A. W., et al. 2018, MNRAS, 475, 716
Greene, J. E., Leathaud, A., Emsellem, E., et al. 2018, ApJ, 852, 36
Guérou, A., et al. 2017, A&A, 608, 24
Harborne, K. 2019, Astrophysics Source Code Library, record ascl:1903.006
Harborne, K. E., Power, C., Robotham, A. S. G., Cortese, L., & Taranu, D. S. 2019, MNRAS, 483, 249
Jesseit, R., Cappellari, M., Naab, T., et al. 2009, MNRAS, 397, 1202
Lagos, C. D. P., Stevens, A. R. H., Bower, R. G., et al. 2018a, MNRAS, 473, 4956
Lagos, C. D. P., Schaye, J., Bahé, Y., Van de Sande, J., Kay, S. T., Barnes, D., Davis, T. A., & Vecchia, C. D. 2018b, MNRAS, 476, 4327
Lagos, C. D. P., Tobar, R. J., Robotham, A. S., Obreschkow, D., Mitchell, P. D., Power, C., & Elahi, P. J. 2018c, MNRAS, 481, 3573
McAlpine, S., et al. 2016, Astronomy and Computing, 15, 72
Moffat, A. F. J. 1969, A&A, 3, 455
Naab, T., Oser, L., Emsellem, E., et al. 2014, MNRAS, 444, 3357
Overzier, R., Lemson, G., Angulo, R. E., Bertin, E., Blaizot, J., Henriques, B. M., Marleau, G. D., & White, S. D., 2013, MNRAS, 428, 778
Papastergis, E., & Ponomareva, A. A., 2017, A&A, 601, 6 pp.
Pedrosa, S. E., & Tissera, P. B. 2015, A&A, 584, 8
Pillepich, A., et al. 2018, MNRAS, 473, 4077
Planck Collaboration, et al. 2018, arXiv:1807.06209
Robotham, A. S. G., Taranu, D. S., Tobar, R., et al. 2017, MNRAS, 466, 1513
Robotham, A. S. G., Bellstedt, S., Lagos, C. D. P., Thorne, J. E., Davies, L. J., Driver, S. P., & Bravo, M., 2020, arXiv:2002.06980
Schaye, J., et al., 2015, MNRAS, 446, 521
Springel, V. 2005, MNRAS, 364, 1105
Springel, V., & Hernquist, L. 2003, MNRAS, 339, 289
Springel, V., White, S. D. M., Jenkins, A., et al. 2005, Nature, 435, 629
Teklu, A. F., Remus, R.-S., Dolag, K., et al. 2015, ApJ, 812, 24 pp.
Torrey, P., et al. 2015, MNRAS, 447, 2753
Vazdekis, A., Koleva, M., Ricciardelli, E., Röck, B., & Falcón-Barroso, J. 2016, MNRAS, 463, 3409
Vogelsberger, M., Genel, S., Springel, V., et al. 2014, Nature, 509, 177
Yurin, D., & Springel, V. 2014, MNRAS, 444, 62
van de Sande, J., Bland-Hawthorn, J., Brough, S., et al. 2017a, MNRAS, 472, 1272
van de Sande, J., Bland-Hawthorn, J., Fogarty, L. M. R., et al. 2017b, ApJ, 835, 35 pp
van de Sande, J., et al. 2019, MNRAS, 484, 869

https://doi.org/10.1017/pasa.2020.8 Published online by Cambridge University Press