Received: 5 December 2024 | Revised: 26 February 2025 | Accepted: 9 March 2025

DOI: 10.1002/pro.70107

TOOLS FOR PROTEIN SCIENCE

THE PROTEIN SOCIETY WILEY

Tutorial on integrative spatiotemporal modeling by integrative modeling platform

Andrew P. Latham [1,2,3] | **Miha Rožič** [1,2,3] | **Benjamin M. Webb** [1,2,3] | **Andrej Sali** [1,2,3]

[1]Quantitative Biosciences Institute, University of California, San Francisco, San Francisco, California, USA

[2]Department of Bioengineering and Therapeutic Sciences, University of California, San Francisco, San Francisco, California, USA

[3]Department of Pharmaceutical Chemistry, University of California, San Francisco, San Francisco, California, USA

Correspondence
Andrej Sali and Andrew P. Latham, Quantitative Biosciences Institute, University of California San Francisco, San Francisco, California 94158, USA.
Email: sali@salilab.org; aplatham@salilab.org

Funding information
American Slovenian Education Foundation Junior Fellowship; National Institutes of Health, Grant/Award Numbers: F32GM150243, R01GM083960

Review Editor: Nir Ben-Tal

Abstract

Cells function through dynamic interactions between macromolecules. Detailed characterization of the dynamics of large biomolecular systems is often not feasible by individual biophysical methods. In such cases, it may be possible to compute useful models by integrating multiple sources of information. We have previously developed an integrative method to model dynamic processes by computing biomolecular heterogeneity at fixed time points, then generating static integrative structural modes for each of these heterogeneity models, and finally connecting these static models to produce a scored trajectory model that depicts the process. Here, we demonstrate how to compute, score, and assess these integrative spatiotemporal models using our open-source Integrative Modeling Platform (IMP) program (https://integrativemodeling.org/).

KEYWORDS

biomolecular processes, integrative structure modeling, molecular dynamics, structural biology

1 | INTRODUCTION

Structural models of biomolecules have provided tremendous insight into cellular function. However, static structures alone cannot capture the full complexity of biology, as biomolecules are constantly in motion. Techniques such as Förster resonance energy transfer (FRET) spectroscopy and microscopy (Lerner et al., 2021), Nuclear magnetic resonance (NMR) spectroscopy (Alderson & Kay, 2021), small-angle X-ray scattering (SAXS) (Schneidman-Duhovny & Hammel, 2018), atomic force microscopy (AFM) (Heath & Scheuring, 2018), and physics-based molecular dynamics simulations (Hollingsworth & Dror, 2018; Karplus & McCammon, 2002; Pak & Voth, 2018), allow us to characterize the dynamics of biomolecules. Yet, many biological processes involve large-scale rearrangements of multi-component systems on long time scales, which cannot be described comprehensively by any of these individual approaches.

For static structures, integrative modeling has proven to be a powerful approach to model systems recalcitrant to individual techniques (Alber, Dokudovskaya, Veenhoff, Zhang, Kipper, Devos, Suprapto, Karni-Schmidt, Williams, Chait, Rout, & Sali, 2007a; Braitbard et al., 2019; Hummer & Köfinger, 2015; Koukos & Bonvin, 2020; Latham & Zhang, 2019; Rantos et al., 2022; Rout & Sali, 2019; Sali, 2021). Integrative modeling combines information from multiple methods, including various types of experimental data, statistical preferences, physical theories, and other prior models. Integrative modeling aims to maximize the accuracy, precision, and completeness of the output model by maximizing the amount of input information and by optimally converting this information into the output model. Many structural

Andrew P. Latham and Miha Rožič contributed equally.

models have been computed with an integrative approach, including those of the nuclear pore complex (NPC) (Alber, Dokudovskaya, Veenhoff, Zhang, Kipper, Devos, Suprapto, Karni-Schmidt, Williams, Chait, Sali, & Rout, 2007b; Kim et al., 2018; Singh et al., 2024), the structural maintenance of chromosomes 5/6 complex (Yu et al., 2021), and open reading frame 2 protein (Baldwin et al., 2024).

Recently, we have developed an integrative spatiotemporal method to generalize integrative modeling of static structures to modeling biomolecular dynamics (Latham et al., 2025; Otsuka et al., 2023). We illustrated the method by computing a model of the assembly pathway of the human NPC in the context of the reforming nuclear envelope during mitotic cell division, based on live-cell correlated electron tomography, bulk fluorescence correlation spectroscopy-calibrated quantitative live imaging, and a structural model of the fully assembled NPC (Latham et al., 2025). The method begins by generating a set of "heterogeneity models" at each time point. Each heterogeneity model describes a likely biomolecular composition, with each component assigned to a specific coarse location in the final state, such as the fully assembled complex when modeling the assembly process. For each heterogeneity model in the set, a "snapshot model" is computed. A snapshot model contains a set of alternative standard static integrative structure models based on the composition specified by the corresponding heterogeneity model and other information available for the corresponding time point. Next, trajectories are enumerated by connecting alternative snapshot models at adjacent time points, followed by scoring each trajectory based on its fit to the input information, including both snapshot model and transition scores. This modeling process results in a set of trajectories ranked by their agreement with input information (i.e., the "trajectory model"). The method is applicable to cases where data is available at discrete time points throughout a biological process, as is the case for the NPC assembly process (Latham et al., 2025; Otsuka et al., 2016; Otsuka et al., 2018; Otsuka et al., 2023).

To ease the barrier for computing integrative models, we have developed the open-source integrative modeling platform (IMP) program, freely available at https://integrativemodeling.org/ (Russel et al., 2012; Yang et al., 2012). IMP is a modular and flexible program that allows us to mix model representations, scoring terms based on varied types of input information, and model sampling algorithms. Multiple tutorials for computing static integrative structure models are already available (Saltzberg et al., 2019; Saltzberg et al., 2021; Webb et al., 2011; Webb et al., 2018). Here, we describe how to use IMP for integrative spatiotemporal modeling.

2 | INTEGRATIVE SPATIOTEMPORAL MODELING WORKFLOW

In general, integrative modeling consists of three steps (Alber, Dokudovskaya, Veenhoff, Zhang, Kipper, Devos, Suprapto, Karni-Schmidt, Williams, Chait, Rout, & Sali, 2007a; Rout & Sali, 2019; Sali, 2021). In the first step, information about the system of interest is gathered. In the second step, this input information is converted into an output model, using a statistical modeling approach (Gelman et al., 1995; McElreath, 2018). This approach requires specifying a model representation, a scoring function, and a search scheme as follows. A model representation contains variables whose values are computed by modeling. A scoring function quantifies a match between a model and input information. Finally, a search scheme finds good-scoring models. In the third step, the model is assessed. This three-step process can be optionally iterated until assessment indicates the model is suitable for addressing biological questions of interest.

While the workflow above is in principle able to accommodate producing any type of a model based on any input information, the search space may be too large for sufficiently efficient computation. The efficiency of searching can be improved by using a composite workflow, in which multiple instances of statistical modeling are performed such that solutions of simpler modeling problems provide prior models for solving more complex modeling problems. For example, structural models of subunits in a complex can be computed first before they are assembled into a structural model of the complex, as was done for the static NPC model (Alber, Dokudovskaya, Veenhoff, Zhang, Kipper, Devos, Suprapto, Karni-Schmidt, Williams, Chait, Sali, & Rout, 2007b; Kim et al., 2018; Singh et al., 2024).

Here, we represent a biomolecular process as a trajectory through a series of snapshot models (Figure 1). To model a process represented this way, we apply a composite workflow consisting of separate integrative modeling workflows to compute (i) heterogeneity models, (ii) snapshot models, and (iii) the trajectory model. For example, modeling the NPC assembly process included three independent instances of integrative modeling: (i) computing heterogeneity models at each time point, (ii) computing a snapshot model for each heterogeneity model, and (iii) computing a trajectory model by connecting snapshot models.

In this tutorial, we walk through all of the steps for integrative spatiotemporal modeling and demonstrate how to perform them in IMP. For illustration purposes, we rely on a hypothetical assembly mechanism of the Bmi1/Ring1b-UbcH5c complex (Bentley et al., 2011). The dynamic information used in this study is purely synthetic; thus, the model is not meant to depict the actual assembly mechanism of the complex. This tutorial case was selected to provide a known benchmark for evaluating model

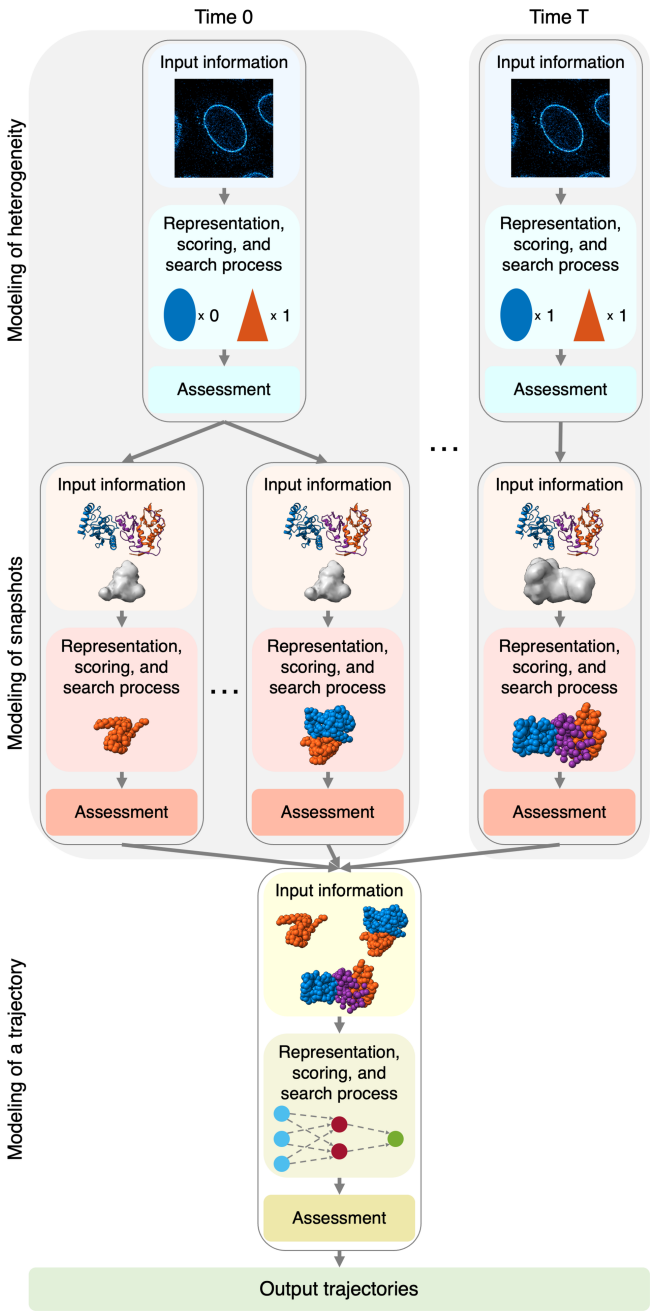

FIGURE 1 Overview of integrative spatiotemporal modeling. Spatiotemporal modeling consists of a composite modeling protocol consisting of three steps: modeling of copy numbers, modeling of snapshots, and modeling of a trajectory. Each modeling workflow is an individual integrative modeling problem, which proceeds through three steps: (i) gathering information, (ii) representing, scoring, and searching for models, and (iii) assessing alternative models. In the case of complex assembly, the final time point (T) is assumed to be the fully assembled structure, and thus only has one heterogeneity model. The output of modeling is a set of weighted trajectories describing the biological process.

accuracy and to be computationally simple. The tutorial can also be followed online at https://integrativemodeling.org/tutorials/spatiotemporal/, and also includes a Google Colab notebook with many of the key steps. The complete code is available at https://github.com/salilab/imp_spatiotemporal_tutorial, which includes both a complete, worked example of the tutorial in the `modeling` folder and the necessary code and input information without the resulting model in the `modeling_empty` folder.

2.1 | Modeling of heterogeneity

The first modeling workflow produces heterogeneity models (Figure 1). For the Bmi1/Ring1b-UbcH5c complex used in this tutorial, heterogeneity modeling only includes protein copy number. In general, other information, such as the coarse location in the final state, could also be modeled in a similar manner, as was done for the NPC assembly process (Latham et al., 2025).

2.1.1 | Heterogeneity modeling step 1: gathering of information

In the first step of heterogeneity modeling, information about the process of interest is gathered. To model the heterogeneity of the system, we utilize the X-ray crystal structure of the complete Bmi1/Ring1b-UbcH5c complex from the Protein Data Bank (PDB) (Bentley et al., 2011; Burley et al., 2023) (Figure 2a) and synthetically generated protein copy numbers during the assembly process (Figure 2c). The PDB structure of the complex informs the final heterogeneity model and constrains the maximum copy number for each protein, while the protein copy number data informs the stoichiometry of the complex during assembly.

2.1.2 | Heterogeneity modeling step 2: representation, scoring function, and search process

In the second step of heterogeneity modeling, we represent, score, and search for heterogeneity models. These operations are implemented by the `heterogeneity_modeling.py` script in the `Heterogeneity/Heterogeneity_Modeling` folder of the tutorial. The script represents each heterogeneity model as a series of copy numbers for each protein, scores those copy numbers based on their fit to synthetically generated protein copy number data, and searches for models through enumeration.

Specifically, these steps are performed by the `prepare_protein_library` function, which enumerates all protein copy numbers between a single protein and the fully assembled state, ranks the most likely of these protein copy numbers at each time point based on the experimental copy number data, and selects a user-

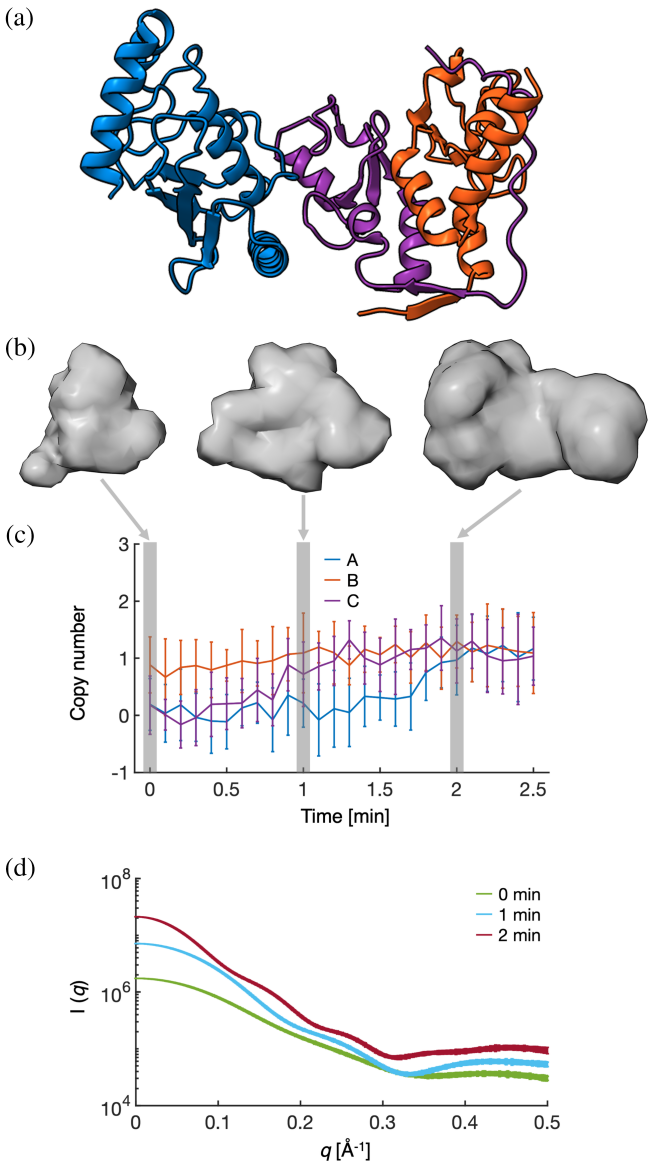

FIGURE 2 Input information for the integrative spatiotemporal modeling tutorial. This integrative spatiotemporal modeling tutorial focuses on building a model for the hypothetical assembly mechanism of the Bmi1/Ring1b-UbcH5c complex with four forms of information: (a) structural model of the fully assembled structure, (b) density maps of the assembling complex at 0, 1, and 2 min, (c) protein copy number data as a function of time, and (d) SAXS profiles of the assembling complex at 0, 1, and 2 min.

specified number of these models at each time point. In this tutorial, we chose to select the three most likely protein combinations at each time point, except for the final time point, which we restricted to having the same protein copy number as the PDB structure. The `prepare_protein_library` function then writes IMP topology files to prepare each heterogeneity model for snapshot modeling, based on the topology file of the full complex, `spatiotemporal_topology.txt`. We will discuss this topology file in more detail in Section 2.2.2.

2.1.3 | Heterogeneity modeling step 3: assessment

In the third step of heterogeneity modeling, we assess the heterogeneity models. Assessment of a model should include calculating the sampling precision of the model (Viswanath et al., 2017), comparing the model to data used to construct it, validating the model against data not used to construct it, and quantifying the precision of the model (Sali, 2021). For now, we focus on comparing the model to experimental data, as other assessments will be performed later, when the trajectory model is assessed.

Specifically, the `plot_heterogeneity.py` script in `Heterogeneity/Heterogeneity_Assessment` plots a comparison between the modeled and experimental copy numbers (Figure 3). We observe that the most likely copy numbers are sampled, indicating that we are ready to proceed to snapshot modeling.

2.2 | Modeling of snapshots

The second modeling workflow produces snapshot models (Figure 1). Each instance of snapshot modeling produces a set of structural models for a single heterogeneity model, corresponding to a single snapshot model. Snapshot modeling is performed independently for each heterogeneity model. The procedure for producing a snapshot model mirrors that of static structural modeling (Alber, Dokudovskaya, Veenhoff, Zhang, Kipper, Devos, Suprapto, Karni-Schmidt, Williams, Chait, Rout, & Sali, 2007a; Rout & Sali, 2019; Sali, 2021).

2.2.1 | Snapshot modeling step 1: gathering of information

In the first step of snapshot modeling, information about the process of interest is gathered. To model a static snapshot model, we use a heterogeneity model, the X-ray crystal structure of the complete Bmi1/Ring1b-UbcH5c complex from the PDB (Bentley et al., 2011; Burley et al., 2023) (Figure 2a), a synthetically generated electron tomography (ET) density map during the assembly process (Figure 2b), and physical principles. The input heterogeneity model informs protein copy numbers for the snapshot model. The PDB structure of the complex informs the structure of the individual proteins. The time-dependent ET data inform the size and shape of the assembling complex. Physical principles inform connectivity and excluded volume.

In addition to the information used in this tutorial, IMP can also convert many other types of input information into model representation or spatial restraints (Russel et al., 2012; Yang et al., 2012), including chemical cross-links (Shi et al., 2014), FRET measurements

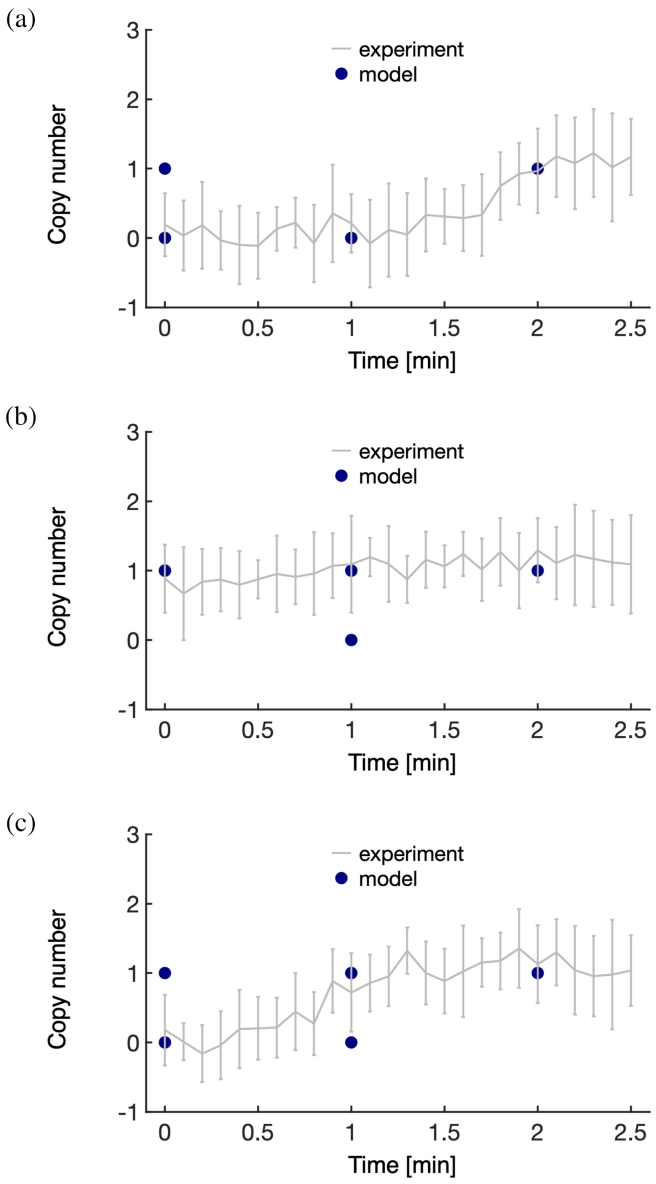

FIGURE 3 Assessing heterogeneity models. Comparison between experimental and protein copy numbers from heterogeneity modeling for proteins A (a), B (b), and C (c). Error bars represent standard deviations over multiple experiments.

(Bonomi et al., 2014), hydrogen/deuterium exchange mass spectrometry (HDX-MS) spectra (Saltzberg et al., 2017), comparative protein structure models (Sali & Blundell, 1993), and deep-learning structure models (Jumper et al., 2021).

2.2.2 | Snapshot modeling step 2: representation, scoring function, and search process

In the second step of snapshot modeling, we represent, score, and search for snapshot models. These three steps are performed in the `Snapshots/Snapshots_Modeling` folder. In this folder, there are two Python scripts: `static_snapshot.py`, which represents, scores, and searches for a single snapshot model, and `start_sim.py`, which automates the creation of multiple snapshot models at each time point.

To begin, `static_snapshot.py` is similar to other IMP scripts used for integrative modeling of static structures (Saltzberg et al., 2021; Webb et al., 2018). We start by choosing the model representation from a topology file, such as `spatiotemporal_topology.txt`, located in the `Heterogeneity/Heterogeneity_Modeling` folder. Here, we choose a multiscale representation, where the protein is represented by coarse-grained beads and Gaussians. Each bead depicts one residue. The Gaussian representation will facilitate comparison to ET data through a Gaussian mixture model (GMM) (Bonomi et al., 2019; Kawabata, 2008). Gaussians within the GMM are assigned density corresponding to approximately ten protein residues through a divide-and-conquer approach. We then divide each protein into rigid bodies and flexible strings of beads connecting them. The rigid bodies will not change their shape during sampling. In contrast, the beads will be free to move relative to each other, although they will be restrained by sequence connectivity. The choice of rigid bodies and flexible strings of beads should be commensurate with both the input information and the application of interest; increasing the number of rigid bodies and beads may allow for models to improve their fit to restraints but will likely require an increased amount of input information and sampling. Generally, known protein domains or secondary structure elements are set as rigid bodies, and disordered or flexible regions are set to be flexible beads (Kim et al., 2018).

Next, in `static_snapshot.py`, a variety of restraints are implemented to score the model. In this application, we choose three restraints: connectivity, excluded volume, and a match of a model to ET data. The complete scoring function is the sum of each of these three terms. Connectivity restrains beads adjacent in sequence to be close together in space. Excluded volume penalizes structures in which beads overlap in space. Finally, the ET density restraint enforces overlap between the experimental ET map and a calculated density map of an evaluated complex model. To reduce the computational cost of this calculation, GMMs are used to represent both the experimental and computed density maps.

Finally, `static_snapshot.py` implements one instance of searching for good-scoring models. The search process is performed through Markov Chain Monte Carlo sampling with temperature replica exchange (Habeck et al., 2005; Metropolis & Ulam, 1949). Sixteen geometrically spaced replicas with temperatures between 1.0 k_BT and 2.5 k_BT are used. 10^5 sampling steps are performed in each run, with frames saved every 200 steps.

While the `static_snapshot.py` script implements a single instance of sampling for a snapshot model, multiple instances of sampling are needed for sufficient sampling. Further, this process needs to be repeated for each heterogeneity model. All of these sampling tasks are performed by running `start_sim.py`. Specifically, the script uses the topology files for each heterogeneity model made by `heterogeneity_modeling.py` and the `static_snapshot.py` script to generate 50 independent sampling runs for each of the seven heterogeneity models.

2.2.3 | Snapshot modeling step 3: assessment

In the third step of snapshot modeling, we assess the snapshot models. Similarly to the heterogeneity models, we will focus only on calculating the sampling precision of snapshot models, as a more complete assessment of the model will be performed later, when the trajectory models are assessed. Assessment is performed by the `snapshot_assessment.py` script in `Snapshots/Snapshots_Assessment`.

First, models are filtered to select those that meet certain parameter thresholds. In this example, we use `imp_sampcon select_good` to select models whose cross-correlation with ET data is better than the median value; a more rigorous Bayesian weighting of different restraints remains a research topic.

Next, the script plots each score for each snapshot model using `imp_sampcon plot_score`. These plots can be visually inspected to debug the modeling protocol. For example, multimodal distributions may be indicative of insufficient sampling.

To assess the sampling protocol more rigorously, the script runs the `imp_sampcon exhaust` function on each snapshot model, which has been thoroughly discussed previously (Saltzberg et al., 2021; Viswanath et al., 2017). Briefly, the function performs three explicit checks on sampling, providing a test for sufficient sampling and an estimate of the sampling precision. First, it displays the best score of the model from bootstrapping the score distribution, which can be analyzed for convergence (Figure 4a). Second, it runs a non-parametric Kolmogorov–Smirnov (KS) two-sample test on two independently sampled sets of scores to determine whether the scores were drawn from the same parent

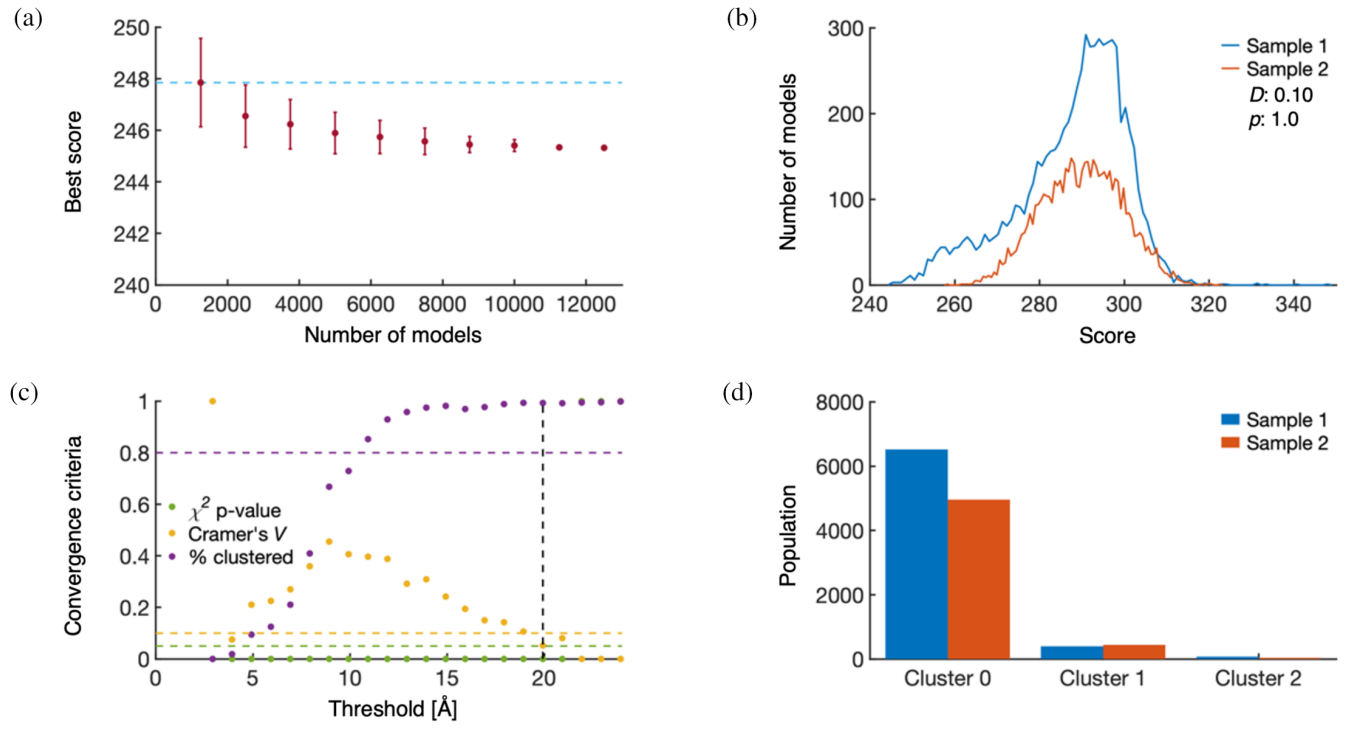

FIGURE 4 Testing for sampling exhaustiveness of a single snapshot model. (a) Testing the convergence of the lowest scoring model. Error bars represent standard deviations of the best scores, estimated by selecting different subsets of models 150 times. The light-blue line helps compare the highest average top score to the top score from other subsets of models. (b) Testing that the scores of two independently sampled models come from the same distribution. The difference between the two distributions, as measured by the KS test statistic (D) and KS test p-value (p) indicates that the difference is both statistically insignificant ($p > 0.05$) and small in magnitude ($D < 0.3$). (c) Determining the structural precision of a snapshot model. RMSD clustering is performed at 1 Å intervals until the clustered population (% clustered) is greater than 80%, and either the χ^2 p-value is greater than 0.05 or Cramér's V is less than 0.1. The sampling precision is indicated by the dashed black line. (d) Populations from sample 1 and sample 2 are shown for each cluster.

TABLE 1 Sampling convergence of snapshot models.

Snapshot	Time	*D*	*p*-value	Sampling precision (Å)
1	0 min	0.00822	1.00	1.02
2	0 min	0.0997	1.00	17.0
3	0 min	0.0613	1.00	16.0
1	1 min	0.114	1.00	16.8
2	1 min	0.0411	1.00	3.00
3	1 min	0.0459	1.00	5.27
1	2 min	0.103	1.00	19.9

Note: For each snapshot model, the KS two-sample test statistic (*D*), the KS test *p*-value, and the sampling precision are provided.

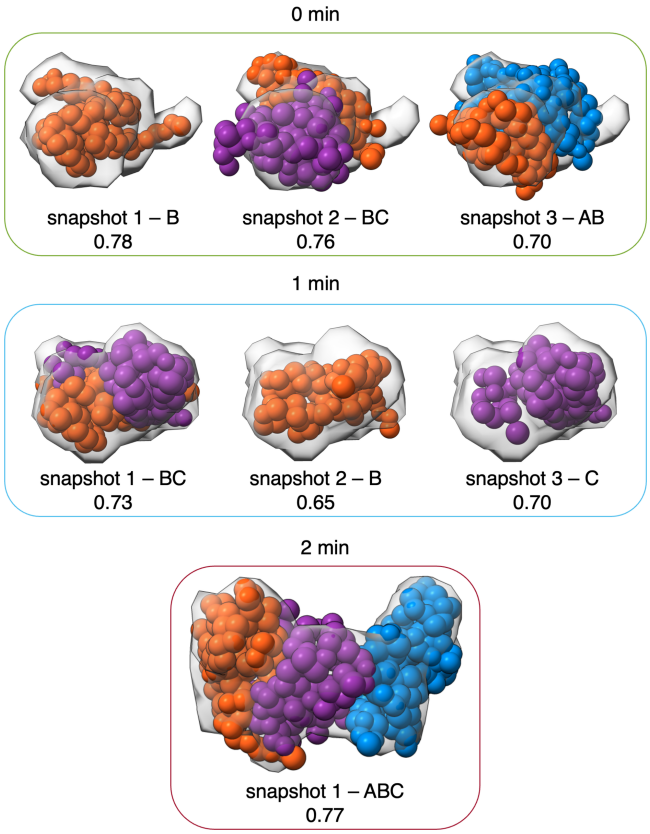

FIGURE 5 Snapshot models for the assembly process. For each snapshot model, the centroid protein structural model from the most populated cluster (A—blue, B—orange, and C—purple) is shown, along with the experimental ET density (gray). Snapshot models are numbered from most to least likely according to experimental copy number data. The number below the snapshot represents the cross-correlation coefficient between the experimental ET density and the forward density of the set of sufficiently good-scoring modeled structures.

distribution (Figure 4b). Third, it estimates the sampling precision of the model by calculating the smallest clustering threshold where each structural cluster includes models from each sample proportionally to its size (Figure 4c,d). To summarize the results from this test, the `snapshot_assessment.py` script outputs a table with the results of the KS test and sampling precision for each snapshot model (Table 1). Further, additional checks can be performed by ensuring sufficient overlap between the localization density of each cluster, which are also automatically output; the localization density is obtained by converting the ensemble of protein structures into the probability of any volume element being occupied by the protein (Alber, Dokudovskaya, Veenhoff, Zhang, Kipper, Devos, Suprapto, Karni-Schmidt, Williams, Chait, Rout, & Sali, 2007a). Finally, the snapshot models can be visualized with Chimera or ChimeraX (Goddard et al., 2018; Pettersen et al., 2004). In particular, `imp_sampcon exhaust` outputs the centroid model for each model cluster, which can serve as a representative model for visualization (Figure 5).

2.3 | Modeling of a trajectory

The third modeling workflow produces a trajectory model (Figure 1). The trajectory model is a set of weighted trajectories that describe the process of interest, including the uncertainty corresponding to the variability among the trajectories.

2.3.1 | Trajectory modeling step 1: gathering of information

In the first step of trajectory modeling, information about the process of interest is gathered. To model a trajectory, we use snapshot models, physical principles of macromolecular dynamics, and computed SAXS profiles at sampled snapshot time points. Snapshot models inform transitions between sampled time points, and their scores inform trajectory scores. Physical principles of macromolecular dynamics inform transitions between sampled time points. More specifically, these physical principles inform both the Markovian framework used to represent the model and the optional transition scoring term, which constrains the assembly process to be associative. A SAXS profile informs the size and shape of the assembling complex and is used only for validation.

2.3.2 | Trajectory modeling step 2: representation, scoring function, and search process

In the second step of trajectory modeling, we represent, score, and search for trajectories in a trajectory model. Each trajectory is a series of snapshot models, with one snapshot model at each sampled time point. In this representation, the complete set of trajectories can be viewed as a directed acyclic graph, where a node in the graph represents a snapshot model and an edge

represents connections between two snapshot models at neighboring time points.

Each trajectory score is the product of the scores of the snapshot models visited along the path and the scores of transitions between them:

$$w(x)\alpha\prod_{t=0}^{T}P(x_t|D_t)\bullet\prod_{t=0}^{T-1}W(x_{t+1}|D_{t,t+1}),\qquad(1)$$

where t indexes time from 0 to the final modeled snapshot (T); X_t is the snapshot model at time t; D_t is the experimental data at time t; $P(X_t|D_t)$ is the snapshot model score; and $W(X_{t+1}|X_t,D_{t,t+1})$ is the transition score. A trajectory weight ($W(\chi)$) is normalized so that the sum of all trajectory weights is 1.0. Transition scores are currently based on a simple metric that either allows or disallows a transition; a transition is allowed only if all proteins in the first snapshot model are included in the second snapshot model. More elaborate transition scores could take into account experimental information and/or physical models of macromolecular dynamics (Ghosh et al., 2020; Husic & Pande, 2018; Lindorff-Larsen et al., 2011; Noé et al., 2020).

Multiple trajectories are constructed by enumerating all connections between snapshot models at adjacent time points and scoring them according to Equation (1). This enumeration results in a set of weighted trajectories (i.e., a trajectory model).

The representation, scoring, and search process steps for computing a trajectory model are performed by the `create_trajectories.py` script in `Trajectories/Trajectories_Modeling`. The script uses the `create_DAG` function to create a set of trajectories from the snapshot models and the experimental copy number data. This set can be visualized as either a directed acyclic graph or by rendering the centroid of the most likely cluster of structural models in each snapshot model visited along the most likely trajectory (Figure 6).

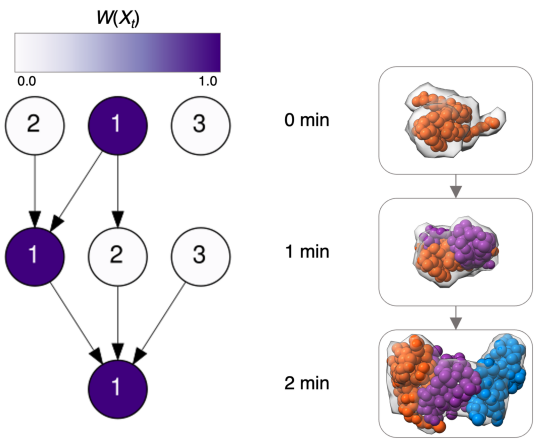

FIGURE 6 Hypothetical assembly mechanism of the Bmi1/Ring1b-UbcH5c complex. The integrative spatiotemporal model of the protein complex assembly process is represented as a directed acyclic graph (left) and a series of centroid structures along the highest weighted trajectory (right). Each row corresponds to a different time point in the assembly process (0, 1, and 2 min). Each node represents a single snapshot model, is numbered according to its ID in Figure 5, and is shaded according to its weight in the final model ($W(X_t)$). Protein A, blue; protein B, orange; and protein C, purple.

2.3.3 | Trajectory modeling step 3: assessment

In the third step of trajectory modeling, we assess the trajectory model. Ideally, we would like to know the model accuracy (i.e., the difference between the model and the truth). However, it is generally not possible to estimate the accuracy of a model because the truth is not known. Instead, we can assess a model in at least four ways: estimating the sampling precision, quantifying the model precision, comparing the models against data used to construct them, and validating the models against data not used to construct them. Each of these steps is performed by the `trajectories_assessment.py` script in the `Trajectories/Trajectories_Assessment` folder.

First, the script quantifies the sampling precision (uncertainty) of the trajectory model with the `analysis.temporal_precision` function. The only source of this sampling uncertainty in the current workflow is the sampling precision of the snapshot models, as trajectories are enumerated. The sampling precision of the trajectory model is defined by the difference between two trajectory models, each computed from two independent samples of the snapshot models. This temporal precision metric will approach 1.0 when the overlap between independently sampled trajectory models is perfect and 0.0 when there is no overlap between the independently sampled trajectory models. For the tutorial example, the temporal precision of 1.0 indicates that the sampling of snapshot models fully converged.

Second, the script quantifies the model precision with the `analysis.precision` function. This function calculates how many trajectories contribute significantly to the trajectory model. The precision will approach 1.0 when one trajectory has a much higher weight than the rest of them; the precision will approach the reciprocal of the number of trajectories when all trajectories are weighted equally. For the tutorial example, the precision of 1.0 indicates that a single trajectory dominates the trajectory model (Figure 6).

Third, the script quantifies the agreement between the trajectory model and the information used to construct it. For each sampled time point, it first calculates the cross-correlation between the computed density of each snapshot model and the corresponding

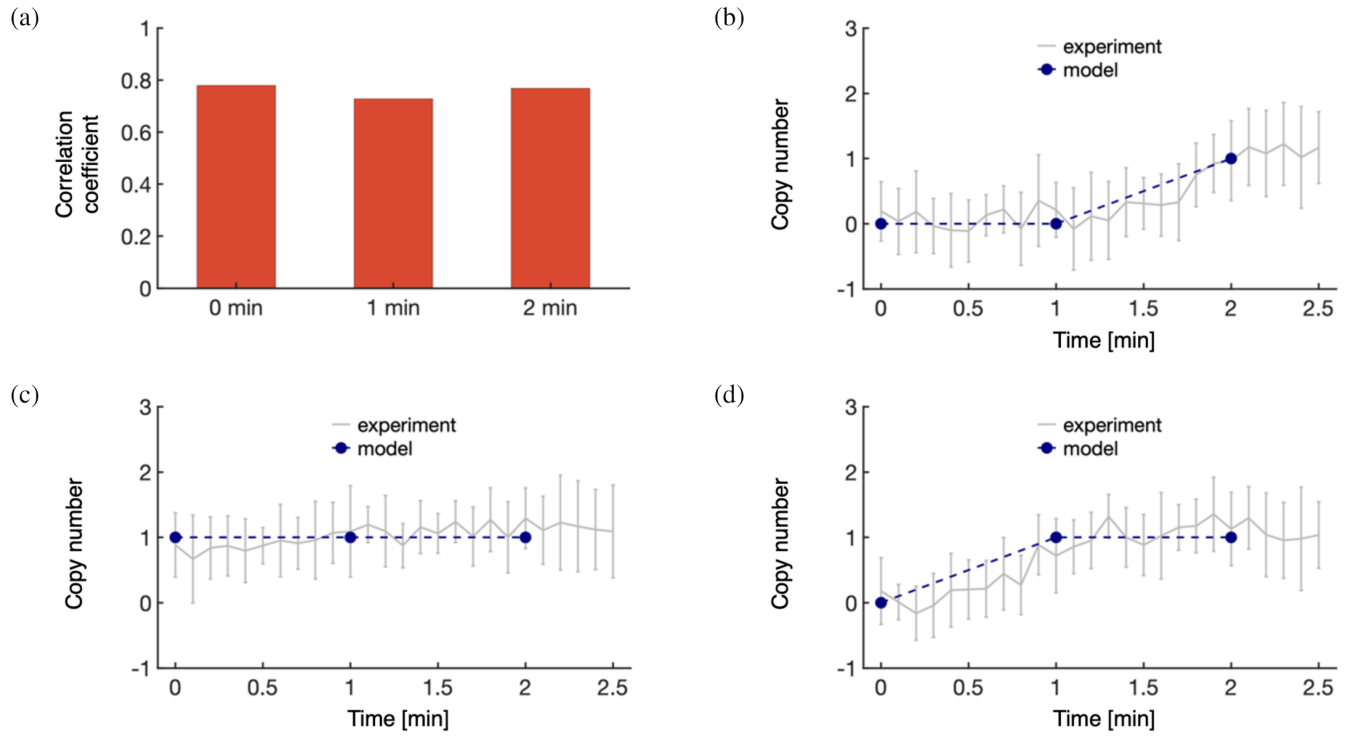

FIGURE 7 Assessing a spatiotemporal model with data used to compute it. (a) Cross-correlation coefficient between the experimental EM density and the forward density of the set of sufficiently good scoring modeled structures in the highest weighted trajectory. Comparison between experimental and modeled protein copy numbers for A (b), B (c), and C (d). Error bars represent standard deviations over multiple experiments or weighted trajectories and are smaller than the symbols when not visible.

experimental ET density (Figure 5). For the tutorial example, the snapshot models along the most likely trajectory show particularly good agreement with the experimental density (Figure 7a). Next, the script calculates the protein copy number for the trajectory model, which is then compared to the experimental copy number (Figure 7b–d). For the tutorial example, the copy numbers of the trajectory model lie well within the experimental error at all time points, indicating satisfactory agreement between the model and the data used to compute it.

Fourth, the script compares the model to data not used in trajectory model construction. In particular, the experimental SAXS profiles were reserved for validation. For each snapshot model, the scattering profile of the centroid structural model is computed by FoXS (Schneidman-Duhovny et al., 2016), followed by quantifying its similarity to the experimental profile using the χ^2 value (Figure 8a–c). For the tutorial example, the χ^2 values indicate that the highest weighted trajectory (Snapshot 1) fits the SAXS profiles well and better than the alternative trajectories (Table 2), providing an independent validation of the model.

Finally, the use of synthetic data in this example provides knowledge of the truth and allows for an evaluation of model accuracy. First, we evaluate the accuracy of the trajectory model by assessing the weight of the true trajectory in the trajectory model. The true trajectory is the highest weighted trajectory, with the weight approaching 1.0, while all other trajectories account for only $\sim 10^{-19}$ of the trajectory weight. Second, the `trajectories_assessment.py` script calculates the average root-mean-square deviation (RMSD) between the structures in the snapshot model at each time point and the final assembled structure. This RMSD is a useful measure of model accuracy, especially when the sampling precision of snapshot models is higher than the RMSD. For the tutorial example, the assembled structure lies within 1.5 Å of the sampling precision at all time points, indicating that the modeling comes close to predicting the true structure for all time points in the assembly process (Figure 8d).

3 | DISCUSSION

In this tutorial, we constructed heterogeneity models at sampled time points along a biological process of interest, computed a snapshot model for each of these heterogeneity models, connected the snapshot models to produce a set of trajectories (a trajectory model), and assessed the trajectory model to titrate our confidence in its predictions. If the precision of the trajectory model was insufficient to answer biological questions of interest, the model assessment could help inform new

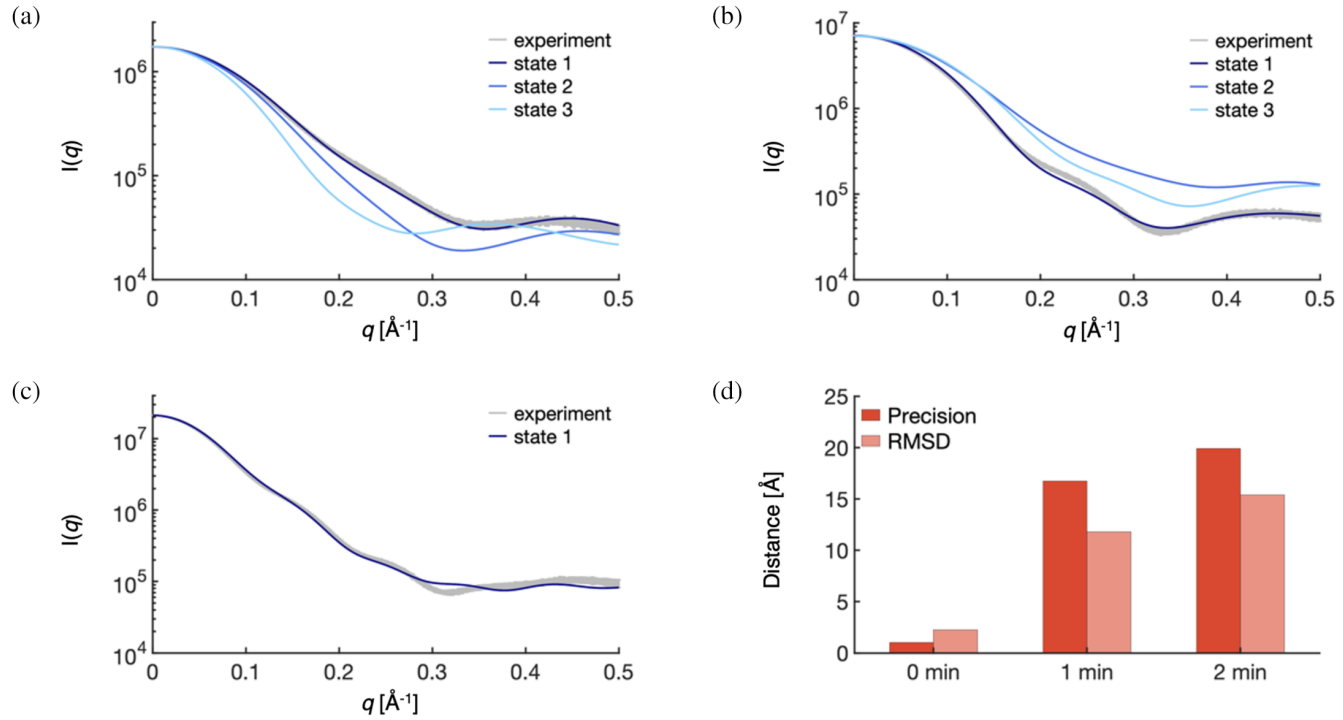

FIGURE 8 Assessing a spatiotemporal model with data not used to compute it and evaluating the model accuracy. Comparison between the centroid structure of the most populated cluster in each snapshot model at each time point and the experimental SAXS profile for 0 min (a), 1 min (b), and 2 min (c). The χ^2 value for each fit is provided in Table 2. (d) Comparison between the model sampling precision (dark red) and the RMSD to the PDB structure (light red) for each snapshot in the highest weighted trajectory.

TABLE 2 Fit of SAXS profiles to experimental data.

Time	Snapshot 1 χ^2	Snapshot 2 χ^2	Snapshot 3 χ^2
0 min	0.87	43.97	107.34
1 min	3.22	873.84	333.01
2 min	5.49		

Note: Measurement of χ^2 for fitting modeled structures to experimental SAXS profiles, as shown in Figure 8a– Snapshot models are labeled according to Figure 5.

experiments to improve the next iteration of trajectory modeling.

If a model is ready for publication, it should be archived so that others can reproduce the results or use the model. The model, data, and protocols should be deposited in the PDB, which has recently been extended for the deposition of integrative structures (Vallat et al., 2021). A straightforward way to deposit such models is through the use of `python-ihm`, which has been described in a previous IMP tutorial (https://integrativemodeling.org/tutorials/deposition/).

In conclusion, modeling biological processes is of high interest because biomolecular systems are dynamic (Russel et al., 2009). We demonstrated an integrative spatiotemporal modeling workflow that aims to maximize the amount of input information imposed on the model by construction. We envision this approach will be useful for modeling large-scale rearrangements of multicomponent systems, which remain challenging to describe by a single technique due to the large spatial and long temporal scales. For example, exocytosis (Jahn & Südhof, 1999; Zhang & Hughson, 2021), viral assembly mechanisms (Perlmutter & Hagan, 2015; Yao et al., 2020), gene transcription (Cramer, 2019; Lin et al., 2021), and ribosome biogenesis (Jiao et al., 2023; Thomson et al., 2013) are well-studied processes where sufficient data may already exist to produce coarse-grained integrative spatiotemporal models capable of providing new mechanistic insight. This tutorial, implemented in IMP, may facilitate creating, assessing, and depositing such models.

4 | MATERIALS AND METHODS

The scripts and input information used in this tutorial are available at the IMP spatiotemporal tutorial Github repository: https://github.com/salilab/imp_spatiotemporal_tutorial.

This tutorial requires a variety of open-sourced packages, specifically:

- IMP version 2.22 or later (https://integrativemodeling.org/)
- numpy (https://numpy.org/)

- pandas (https://pandas.pydata.org/)
- matplotlib (https://matplotlib.org/)
- pyrmsd (https://github.com/salilab/pyRMSD)
- graphviz (https://pypi.org/project/graphviz/)

The computational resources needed for this tutorial are listed for each script, assuming 1 core of an Intel Xeon or AMD EPYC CPU, unless otherwise noted:

- `heterogeneity_modeling.py`—1.1 s
- `plot_heterogeneity.py`—1.0 s
- `start_sim.py`—2775 total CPU hours (approximately 29.7 min per instance of `static_snapshot.py`, which is run 50 times for each snapshot and uses 16 CPU cores simultaneously).
- `snapshot_assessment.py`—5.6 h
- `create_trajectories.py`—56 s
- `trajectories_assessment.py`—2.3 h.

The computational cost of this tutorial (approximately 8 CPU hours without snapshot sampling: approximately 2783 CPU hours for the entire workflow) is much less than the computational cost of modeling the NPC assembly process (Latham et al., 2025; Otsuka et al., 2023) (approximately 1,164,800 total CPU hours), which represents a particularly complex case due to the number of snapshot models (91) and the size of the NPC (~100 MDa in humans) (Lin & Hoelz, 2019). The tutorial also includes a Jupyter notebook and a Google Colab notebook, which can replicate many of the key steps in approximately 5 min of compute time.

4.1 | Generation of synthetic data

Synthetic data was generated to describe the hypothetical assembly mechanism of the Bmi1/Ring1b-UbcH5c complex (Bentley et al., 2011). All code necessary to generate the input information is in the `modeling/Input_Information` folder. To start, we chose protein copy numbers at each time point in the assembly process. Specifically, we chose the true trajectory to be protein B, proteins B–C, and proteins A–B–C at 0, 1, and 2 min, respectively. Then, we created a ground truth structure at each time point by modifying the original structure of the Bmi1/Ring1b-UbcH5c complex to agree with the chosen protein copy numbers.

Synthetic ET data was generated for each time point in the assembly process using its corresponding ground truth structure. First, a density map was computed using the forward density of the ground truth structure at each time point. Second, this density map was converted into a GMM with 40 Gaussians. Third, noise was added by multiplying the mean value of each Gaussian in the GMM by a random variable uniformly distributed between 0.9 and 1.1. The resulting GMMs with noise had reasonable overlap with the original GMMs, with cross-correlation values of 0.86, 0.87, and 0.86 for 0, 1, and 2 min, respectively.

Synthetic copy number data was generated for each protein. The ground truth distribution was

$$n_{i,t} = \frac{1}{2}\tanh(5*(t-(T_i+0.5))), \tag{2}$$

where t is the time at which the copy number is being computed, while T_i is the time at which each protein (i) is added to the complex and takes the value of 0, 1, and 2 for proteins B, C, and A, respectively. Based on this ground truth distribution, synthetic data was computed by generating ten samples from a Gaussian distribution whose means were given by Equation (2) and whose standard deviation was 0.5. Data was generated at 0.1 min intervals between 0.0 and 2.5 min.

Synthetic SAXS data was computed at each time point by generating the forward model of the ground truth structure. The experimental error was approximated for each point in the scattering profile as

$$E(I(q),q) = 0.03*I(q)*5*(q+0.001)*\left(\left(\frac{k}{10}-1\right)+1\right), \tag{3}$$

where $I(q)$ is the scattering intensity, q is the scattering angle, and k is a Poisson random variable with a Poisson parameter of 10.

AUTHOR CONTRIBUTIONS

Andrew P. Latham: Conceptualization; software; investigation; writing – original draft; writing – review and editing; funding acquisition; visualization. **Miha Rožič:** Software; investigation; writing – original draft; writing – review and editing; visualization. **Benjamin M. Webb:** Software; writing – review and editing. **Andrej Sali:** Conceptualization; writing – original draft; writing – review and editing; funding acquisition.

ACKNOWLEDGMENTS

This work was supported by grants from NIH/NIGMS R01GM083960 (A.S.). A.P.L. was further supported by NIH/NIGMS F32GM150243. M.R. was further supported by the American Slovenian Education Foundation Junior Fellowship. We thank Ignacia Echeverria for help implementing our code in IMP, and Catherine Sue for testing the tutorial. We would further like to thank the current and former contributors to IMP.

CONFLICT OF INTEREST STATEMENT

The authors declare no conflicts of interest.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are openly available in imp_spatiotemporal_tutorial at https://github.com/salilab/imp_spatiotemporal_tutorial.

ORCID

Andrew P. Latham https://orcid.org/0000-0002-9338-7253
Benjamin M. Webb https://orcid.org/0000-0003-3360-4540
Andrej Sali https://orcid.org/0000-0003-0435-6197

REFERENCES

Alber F, Dokudovskaya S, Veenhoff LM, Zhang W, Kipper J, Devos D, et al. Determining the architectures of macromolecular assemblies. Nature. 2007a;450:683–94.
Alber F, Dokudovskaya S, Veenhoff LM, Zhang W, Kipper J, Devos D, et al. The molecular architecture of the nuclear pore complex. Nature. 2007b;450:695–701.
Alderson TR, Kay LE. NMR spectroscopy captures the essential role of dynamics in regulating biomolecular function. Cell. 2021;184:577–95.
Baldwin ET, van Eeuwen T, Hoyos D, Zalevsky A, Tchesnokov EP, Sánchez R, et al. Structures, functions and adaptations of the human LINE-1 ORF2 protein. Nature. 2024;626:194–206.
Bentley ML, Corn JE, Dong KC, Phung Q, Cheung TK, Cochran AG. Recognition of UbcH5c and the nucleosome by the Bmi1/Ring1b ubiquitin ligase complex. EMBO J. 2011;30:3285–97.
Bonomi M, Hanot S, Greenberg CH, Sali A, Nilges M, Vendruscolo M, et al. Bayesian weighing of electron cryo-microscopy data for integrative structural modeling. Structure. 2019;27:175–188.e6.
Bonomi M, Pellarin R, Kim SJ, Russel D, Sundin BA, Riffle M, et al. Determining protein complex structures based on a Bayesian model of in vivo Förster resonance energy transfer (FRET) data. Mol Cell Proteomics. 2014;13:2812–23.
Braitbard M, Schneidman-Duhovny D, Kalisman N. Integrative structure modeling: overview and assessment. Annu Rev Biochem. 2019;88:113–35.
Burley SK, Bhikadiya C, Bi C, Bittrich S, Chao H, Chen L, et al. RCSB protein data Bank (RCSB.Org): delivery of experimentally-determined PDB structures alongside one million computed structure models of proteins from artificial intelligence/machine learning. Nucleic Acids Res. 2023;51:D488–508.
Cramer P. Organization and regulation of gene transcription. Nature. 2019;573:45–54.
Gelman A, Carlin JB, Stern HS, Rubin DB. Bayesian data analysis. New York: Chapman and Hall/CRC; 1995.
Ghosh K, Dixit PD, Agozzino L, Dill KA. The maximum caliber variational principle for nonequilibria. Annu Rev Phys Chem. 2020;71:213–38.
Goddard TD, Huang CC, Meng EC, Pettersen EF, Couch GS, Morris JH, et al. UCSF ChimeraX: meeting modern challenges in visualization and analysis. Protein Sci. 2018;27:14–25.
Habeck M, Nilges M, Rieping W. Replica-exchange Monte Carlo scheme for bayesian data analysis. Phys Rev Lett. 2005;94:18105.
Heath GR, Scheuring S. High-speed AFM height spectroscopy reveals μs-dynamics of unlabeled biomolecules. Nat Commun. 2018;9:1–11.
Hollingsworth SA, Dror RO. Molecular dynamics simulation for all. Neuron. 2018;99:1129–43.
Hummer G, Köfinger J. Bayesian ensemble refinement by replica simulations and reweighting. J Chem Phys. 2015;143:243150.
Husic BE, Pande VS. Markov state models: from an art to a science. J Am Chem Soc. 2018;140:2386–96.
Jahn R, Südhof TC. Membrane fusion and exocytosis. Annu Rev Biochem. 1999;68:863–911.
Jiao L, Liu Y, Yu X-Y, Pan X, Zhang Y, Tu J, et al. Ribosome biogenesis in disease: new players and therapeutic targets. Signal Transduct Target Ther. 2023;8:15.
Jumper J, Evans R, Pritzel A, Green T, Figurnov M, Ronneberger O, et al. Highly accurate protein structure prediction with AlphaFold. Nature. 2021;596(7873):583–9. https://doi.org/10.1038/s41586-021-03819-2
Karplus M, McCammon JA. Molecular dynamics simulations of biomolecules. Nat Struct Biol. 2002;9:646–52.
Kawabata T. Multiple subunit fitting into a low-resolution density map of a macromolecular complex using a gaussian mixture model. Biophys J. 2008;95:4643–58.
Kim SJ, Fernandez-Martinez J, Nudelman I, Shi Y, Zhang W, Raveh B, et al. Integrative structure and functional anatomy of a nuclear pore complex. Nature. 2018;555(7697):475–82. https://doi.org/10.1038/nature26003
Koukos PI, Bonvin AMJJ. Integrative modelling of biomolecular complexes. J Mol Biol. 2020;432:2861–81.
Latham AP, Tempkin JOB, Otsuka S, Zhang W, Ellenberg J, Šali A. Integrative spatiotemporal modeling of biomolecular processes: application to the assembly of the nuclear pore complex. Proc Natl Acad Sci U S A. 2025;122:e2415674122.
Latham AP, Zhang B. Improving coarse-grained protein force fields with small-angle X-ray scattering data. J Phys Chem B. 2019;123(5):1026–34. https://doi.org/10.1021/acs.jpcb.8b10336
Lerner E, Barth A, Hendrix J, Ambrose B, Birkedal V, Blanchard SC, et al. FRET-based dynamic structural biology: challenges, perspectives and an appeal for open-science practices. Elife. 2021;10:e60416.
Lin DH, Hoelz A. The structure of the nuclear pore complex (an update). Annu Rev Biochem. 2019;88:725–83.
Lin X, Qi Y, Latham AP, Zhang B. Multiscale modeling of genome organization with maximum entropy optimization. J Chem Phys. 2021;155:10901.
Lindorff-Larsen K, Piana S, Dror RO, Shaw DE. How fast-folding proteins fold. Science. 2011;334:517–20.
McElreath R. Statistical rethinking: a Bayesian course with examples in R and Stan. New York: Chapman and Hall/CRC; 2018.
Metropolis N, Ulam S. The Monte Carlo method. J Am Stat Assoc. 1949;44:335–41.
Noé F, Tkatchenko A, Müller K-R, Clementi C. Machine learning for molecular simulation. Annu Rev Phys Chem. 2020;71:361–90.
Otsuka S, Bui KH, Schorb M, Hossain MJ, Politi AZ, Koch B, et al. Nuclear pore assembly proceeds by an inside-out extrusion of the nuclear envelope. eLife. 2016;5:e19071. https://doi.org/10.7554/eLife.19071
Otsuka S, Steyer AM, Schorb M, Hériché J-K, Hossain MJ, Sethi S, et al. Postmitotic nuclear pore assembly proceeds by radial dilation of small membrane openings. Nat Struct Mol Biol. 2018;25:21–8.
Otsuka S, Tempkin JOB, Zhang W, Politi AZ, Rybina A, Hossain MJ, et al. A quantitative map of nuclear pore assembly reveals two distinct mechanisms. Nature. 2023;613:575–81.
Pak AJ, Voth GA. Advances in coarse-grained modeling of macromolecular complexes. Curr Opin Struct Biol. 2018;52:119–26.
Perlmutter JD, Hagan MF. Mechanisms of virus assembly. Annu Rev Phys Chem. 2015;66:217–39.
Pettersen EF, Goddard TD, Huang CC, Couch GS, Greenblatt DM, Meng EC, et al. UCSF chimera—a visualization system for exploratory research and analysis. J Comput Chem. 2004;25:1605–12.
Rantos V, Karius K, Kosinski J. Integrative structural modeling of macromolecular complexes using Assembline. Nat Protoc. 2022;17:152–76.
Rout MP, Sali A. Principles for integrative structural biology studies. Cell. 2019;177:1384–403.

Russel D, Lasker K, Phillips J, Schneidman-Duhovny D, Velázquez-Muriel JA, Sali A. The structural dynamics of macromolecular processes. Curr Opin Cell Biol. 2009;21:97–108.
Russel D, Lasker K, Webb B, Velázquez-Muriel J, Tjioe E, Schneidman-Duhovny D, et al. Putting the pieces together: integrative modeling platform software for structure determination of macromolecular assemblies. PLoS Biol. 2012;10:e1001244.
Sali A. From integrative structural biology to cell biology. J Biol Chem. 2021;296:100743.
Sali A, Blundell TL. Comparative protein modelling by satisfaction of spatial restraints. J Mol Biol. 1993;234:779–815.
Saltzberg DJ, Broughton HB, Pellarin R, Chalmers MJ, Espada A, Dodge JA, et al. A residue-resolved Bayesian approach to quantitative interpretation of hydrogen-deuterium exchange from mass spectrometry: application to characterizing protein-ligand interactions. J Phys Chem B. 2017;121:3493–501.
Saltzberg D, Greenberg CH, Viswanath S, Chemmama I, Webb B, Pellarin R, et al. Modeling biological complexes using integrative modeling platform. In: Bonomi M, Camilloni C, editors. Biomolecular simulations: methods and protocols. New York, NY: Springer New York; 2019. p. 353–77.
Saltzberg DJ, Viswanath S, Echeverria I, Chemmama IE, Webb B, Sali A. Using integrative modeling platform to compute, validate, and archive a model of a protein complex structure. Protein Sci. 2021;30:250–61.
Schneidman-Duhovny D, Hammel M. Modeling structure and dynamics of protein complexes with SAXS profiles. Methods Mol Biol. 2018;1764:449–73.
Schneidman-Duhovny D, Hammel M, Tainer JA, Sali A. FoXS, FoXS-Dock and MultiFoXS: single-state and multi-state structural modeling of proteins and their complexes based on SAXS profiles. Nucleic Acids Res. 2016;44:W424–9.
Shi Y, Fernandez-Martinez J, Tjioe E, Pellarin R, Kim SJ, Williams R, et al. Structural characterization by cross-linking reveals the detailed architecture of a coatomer-related heptameric module from the nuclear pore complex. Mol Cell Proteomics. 2014;13:2927–43.
Singh D, Soni N, Hutchings J, Echeverria I, Shaikh F, Duquette M, et al. The molecular architecture of the nuclear basket. Cell. 2024;187:5267–81.
Thomson E, Ferreira-Cerca S, Hurt E. Eukaryotic ribosome biogenesis at a glance. J Cell Sci. 2013;126:4815–21.
Vallat B, Webb B, Fayazi M, Voinea S, Tangmunarunkit H, Ganesan SJ, et al. New system for archiving integrative structures. Acta Crystallogr D Struct Biol. 2021;77:1486–96.
Viswanath S, Chemmama IE, Cimermancic P, Sali A. Assessing exhaustiveness of stochastic sampling for integrative modeling of macromolecular structures. Biophys J. 2017;113:2344–53.
Webb B, Lasker K, Schneidman-Duhovny D, Tjioe E, Phillips J, Kim SJ, et al. Modeling of proteins and their assemblies with the integrative modeling platform. In: Cagney G, Emili A, editors. Network biology: methods and applications. Totowa, NJ: Humana Press; 2011. p. 377–97.
Webb B, Viswanath S, Bonomi M, Pellarin R, Greenberg CH, Saltzberg D, et al. Integrative structure modeling with the integrative modeling platform. Protein Sci. 2018;27:245–58.
Yang Z, Lasker K, Schneidman-Duhovny D, Webb B, Huang CC, Pettersen EF, et al. UCSF chimera, MODELLER, and IMP: an integrated modeling system. J Struct Biol. 2012;179:269–78.
Yao H, Song Y, Chen Y, Wu N, Xu J, Sun C, et al. Molecular architecture of the SARS-CoV-2 virus. Cell. 2020;183:730–738.e13.
Yu Y, Li S, Ser Z, Sanyal T, Choi K, Wan B, et al. Integrative analysis reveals unique structural and functional features of the Smc5/6 complex. Proc Natl Acad Sci U S A. 2021;118(19):e2026844118. https://doi.org/10.1073/pnas.2026844118
Zhang Y, Hughson FM. Chaperoning SNARE folding and assembly. Annu Rev Biochem. 2021;90:581–603.

How to cite this article: Latham AP, Rožič M, Webb BM, Sali A. Tutorial on integrative spatiotemporal modeling by integrative modeling platform. Protein Science. 2025;34(4):e70107. https://doi.org/10.1002/pro.70107