A&A, 690, A274 (2024)
https://doi.org/10.1051/0004-6361/202450995
© The Authors 2024

Astronomy & Astrophysics

A machine learning framework to generate star cluster realisations

George P. Prodan[1,8,⋆], Mario Pasquato[2,3], Giuliano Iorio[1,4,5], Alessandro Ballone[1,4,5], Stefano Torniamenti[1,4,6], Ugo Niccolò Di Carlo[7], and Michela Mapelli[1,4,6]

[1] Dipartimento di Fisica e Astronomia, Università di Padova, Vicolo dell'Osservatorio 3, 35122 Padova, Italy
[2] IASF Milano, via Alfonso Corti 12, Milano, Italy
[3] Ciela – Montreal Institute for Astrophysical Data Analysis and Machine Learning, Montréal, Canada
[4] INFN – Padova, Via Marzolo 8, 35131 Padova, Italy
[5] INAF – Osservatorio Astronomico di Padova, Vicolo dell'Osservatorio 5, Padova, Italy
[6] Institut für Theoretische Astrophysik, ZAH, Universität Heidelberg, Albert-Ueberle-Str. 2, 69120 Heidelberg, Germany
[7] SISSA – Scuola Internazionale Superiore di Studi Avanzati, via Bonomea 365, 34136 Trieste, Italy
[8] Faculty of Sciences, University of Craiova, A.I. Cuza 13, 200585 Craiova, Romania

Received 5 June 2024 / Accepted 4 September 2024

ABSTRACT

Context. Computational astronomy has reached the stage where running a gravitational N-body simulation of a stellar system, such as a Milky Way star cluster, is computationally feasible, but a major limiting factor that remains is the ability to set up physically realistic initial conditions.
Aims. We aim to obtain realistic initial conditions for N-body simulations by taking advantage of machine learning, with emphasis on reproducing small-scale interstellar distance distributions.
Methods. The computational bottleneck for obtaining such distance distributions is the hydrodynamics of star formation, which ultimately determine the features of the stars, including positions, velocities, and masses. To mitigate this issue, we introduce a new method for sampling physically realistic initial conditions from a limited set of simulations using Gaussian processes.
Results. We evaluated the resulting sets of initial conditions based on whether they meet tests for physical realism. We find that direct sampling based on the learned distribution of the star features fails to reproduce binary systems. Consequently, we show that physics-informed sampling algorithms solve this issue, as they are capable of generating realisations closer to reality.

Key words. gravitation – hydrodynamics – methods: numerical – open clusters and associations: general

1. Introduction

The target of humanity's first deliberate attempt at interstellar radio communication was the star cluster M13 (Staff at the National Astronomy & Ionosphere Center 1975), and this act is a testament to the importance of star clusters to astronomy, which can hardly be overstated. For instance, star clusters are the likely birthplace of most stars (Lada & Lada 2003).

Astronomers use gravitational N-body simulations with the purpose of studying the evolution of star clusters (Aarseth 2003; Spurzem & Kamlah 2023). The goals, among others, are to predict gravitational wave emission from black holes and neutron stars formed by cluster stars (Rastello et al. 2021; Dall'Amico et al. 2021; Iorio et al. 2023), constrain the place of origin of the Solar System (Pichardo et al. 2012; Pfalzner et al. 2015), determine the stability and habitability of exoplanets (Spurzem et al. 2009; Malmberg et al. 2011; Parker & Quanz 2012), and understand how star clusters contribute to the overall evolution of the Milky Way as part of the field called Galactic archaeology (Chung et al. 2019; Pang et al. 2021, 2022).

Star clusters are understood to form within molecular clouds. Molecular clouds are high-density, low-temperature regions in the interstellar medium predominantly composed of molecular hydrogen. Star formation in molecular clouds occurs in small regions where the density becomes high enough (e.g. by turbulent colliding flows) to start a fast runaway gravitational collapse (Evans 1999). Gas in molecular clouds is often animated by turbulent motions, which are ultimately responsible for the irregular and complex spatial distribution of young cluster stars (Krumholz et al. 2019; Krause et al. 2020; Ballone et al. 2020). The scientific usefulness of gravitational N-body simulations depends on the realism of initial conditions (primordial positions, velocities, and masses of stars), which hinges on the ability to properly model star formation in molecular clouds. Realistic initial conditions may be obtained by running hydrodynamical simulations. These are computationally expensive, with one simulation taking around 10^5 CPU hours (Ballone et al. 2021).

Machine learning can be leveraged to obtain samples of valid initial conditions at a fraction of the computational price by learning a generative model from the outputs of a limited number of simulations. The first attempt in this direction was presented by Torniamenti et al. (2022). Their work consists of a bespoke manipulation of hierarchical clustering models that is simple and effective but lacks theoretical guarantees. Here we propose a different approach based on Gaussian processes (GPs).

2. Gaussian processes

Gaussian processes serve as probabilistic models capable of predicting function values based on noisy observations

⋆ Corresponding author; prodangp9@gmail.com

Open Access article, published by EDP Sciences, under the terms of the Creative Commons Attribution License (https://creativecommons.org/licenses/by/4.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.
This article is published in open access under the Subscribe to Open model. Subscribe to A&A to support open access publication.

(Wang 2022). Especially valuable when data is limited, GPs present a more fitting alternative to deep neural networks given the constraints of our application (Griffiths 2022). For star cluster realisations, the distribution of stars in a cluster can be represented as a function of key physical parameters, such as star masses, coordinates, and velocities. Leveraging GPs, we can learn the star distribution in clusters in the space defined by these parameters, enabling the generation of synthetic clusters.

The essence of GP modelling lies in Bayesian inference, where model beliefs are updated upon receiving new observations. A GP model is characterised by a mean function, $\mu(x)$ and a kernel function, $K(x, x')$, with the kernel function reflecting the similarity degree between input points and influencing predictions (Rasmussen 2004). In our case, the kernel function aims to represent the spatial relations among stars in a cluster by capturing the relationships among the stars set at star formation and arising through dynamics via, for example, the effects of gravitational attraction between the individual stars.

The kernel function hyperparameters of GP models include the signal variance, σ_y; the length scale, l; and the noise level, σ_n. The signal variance measures signal amplitudes, the length scale indicates covariance decay distance, and the noise caters to errors. All of these parameters are trainable. Usually, the radial basis function kernel (RBF) is employed to compute the amplitude of the covariance function,

$$K(x, x') = \sigma_y^2 \exp\left[-\frac{(x - x')^2}{2l^2}\right]. \tag{1}$$

The hyperparameters of the mean function can be defined depending on the choice of the prior mean function. For instance, a constant mean function $\mu(x) = c$ leads to one additional trainable hyperparameter.

Training aims to minimize the log marginal likelihood (Rasmussen 2004) with respect to the above-mentioned hyperparatemers $\theta = \{\sigma_y, \sigma_n, l, ...\}$, namely

$$\log P(y \mid X, \theta) = -\frac{1}{2}(\mathbf{y} - \mu)^T (K + \sigma_n^2 I)^{-1}(\mathbf{y} - \mu) - \frac{1}{2}\log|K + \sigma_n^2 I| - \frac{n}{2}\log 2\pi. \tag{2}$$

The first term ensures data fitting, the second one is responsible for regularisation, and the last one is a constant. Regularisation is an important part of model training that prevents overfitting (where the model learns irrelevant patterns in the data instead of the underlying relationships). By incorporating regularisation terms, the model generalizes better on unseen data, thus improving its predictive performance. Training with noise is an essential part of this work. By inserting noise, we obtained new cluster realisations that at the same time preserve some of the features of the training samples. Our aim for the GP fitting is to create a model that can reproduce these features.

We drew the predictions from the posterior GP,

$$y \sim \mathcal{GP}(\mu, K + \sigma_n^2 I), \tag{3}$$

where μ, K are the posterior mean and kernel function and I is the identity matrix. One can draw two kinds of predictions. The first is the mean prediction, which generates the expected values that are similar to those determined for the training samples. This prediction is deterministic and represents the noiseless prediction of the GP. The second involves sampling from the GP's posterior distribution, incorporating noise. Here, the predictions are more diverse because they consider the inherent uncertainty in the model. This can lead to a broader range of potential outcomes and possibly to the formation of different clusters. A complete mathematical overview on GP models is presented in Rasmussen & Williams (2006).

3. Learning framework

We have introduced a learning framework that utilizes GP modelling to learn the feature space distributions of stellar clusters, followed by sampling from these distributions to produce new clusters. This is an inverse problem, as the framework obtains new cluster realisations via simulation-based inference (Cranmer et al. 2020; Lueckmann et al. 2021). While training, the inputs of the GP model are the parameters of each star, $\theta_{\mathbf{i}}^{(train)}$ with $i = \overline{1, N}$ for a cluster of N stars. The parameters are physical quantities of the stars, with each star having a specific mass, M; location, $\mathbf{r} = (x, y, z)$; and velocity, $\mathbf{v} = (v_x, v_y, v_z)$, with respect to the cluster's centre of mass and all of the parameters M, $\mathbf{r}$, and $\mathbf{v}$ are not normalised. The GP learns a distribution over the probability density function of the feature space, $f(\theta)$.

Once the GP model input features were determined, our next step involved computing the target distribution, $y = f(\theta)$. For this purpose, k-nearest neighbours density estimators (Mack & Rosenblatt 1979) were implemented to ascertain the probability density values, y_i, which are normalised accordingly,

$$y_i^{(train)} = \frac{N y_i}{\Sigma_{j=1}^N y_j}. \tag{4}$$

Further, we implemented Markov chain Monte Carlo (MCMC) approaches to simulate new cluster realisations based on the Metropolis-Hastings algorithm (Chib & Greenberg 1995; Eckhardt 1987; Robert & Casella 2011). The states proposed in the Markov Chain include a set of N features at a given iteration step. We considered an iteration step n corresponding to a state $S_n = \{\theta_1^{(n)}, \theta_2^{(n)}, ..., \theta_N^{(n)}\}$. The next candidate state, S_{n+1}, was obtained by a jump with probability $T(S_n \to S_{n+1})$. The acceptance probability of this jump is given by

$$\alpha = \min\left(1, \frac{\pi(S_{n+1})T(S_n \to S_{n+1})}{\pi(S_n)T(S_{n+1} \to S_n)}\right), \tag{5}$$

where π is the stationary distribution of the chain. The probability of the new state can be written as the joint probability of the candidate features,

$$\pi(S_n) = \prod_{i=1}^{N} f(\theta_i), \tag{6}$$

where f is a probability density function drawn from the GP model that is already trained on the features $\theta^{(train)}$ and the corresponding target values $\mathbf{y}^{(\mathbf{train})}$ of the density function.

This sequential method first crafts a GP statistical model based on the probability density function of the features space for the cluster stars. While sampling from a standard probability density function is generally straightforward, our specific problem requires meticulous adjustments due to the need for maintaining physical constraints and the presence of complex correlations. This ensures that the generated samples accurately reflect the underlying distribution and maintains the validity of the initial conditions. We employed traditional MCMC methods

based on the Metropolis algorithm to directly sample in a seven-dimensional space (DMCMC). Moreover, we propose a physics-informed sampling approach based on learning the energy space distribution of nearest-neighbour star pairs (EMCMC).

3.1. Direct sampling

The direct approach is to define a seven-dimensional feature space that includes all the physical quantities of the stars in the clusters, meaning that for every star at the iteration of state n, we have $\theta_i^{(n)} = \{M_i^{(n)}, x_i^{(n)}, y_i^{(n)}, z_i^{(n)}, v_{xi}^{(n)}, v_{yi}^{(n)}, v_{zi}^{(n)}\}$ with i in $\overline{1,N}$. Considering the large number of stars in the cluster, achieving convergence becomes challenging if all parameters are subject to change each time we propose a new candidate cluster. However, Eq. (5) remains valid also if we propose only a new candidate star $\theta_i^{(n+1)}$ such that the new state becomes S_{n+1} with

$$\theta_j^{(n+1)} = (1 - \delta_{ij})\theta_i^{(n)} + \delta_{ij}\theta_i^{(n+1)} \tag{7}$$

for j in $\overline{1,N}$, where δ_{ij} is the Kronecker delta, and it is equal to one only if $i = j$; otherwise it is zero. The acceptance rate will be

$$\alpha = \min\left(1, \frac{f\left(\theta_i^{(n+1)}\right)}{f\left(\theta_i^{(n)}\right)}\right). \tag{8}$$

Therefore, at each iteration a new subset of seven features $\theta_i^{(n+1)}$ is proposed with i chosen randomly. We drew new candidates using a normal distribution such that $\theta_{ik}^{(n+1)} \sim \mathcal{N}(\theta_{ik}^{(n)}, \sigma)$ for every feature k of θ_i, where σ is called the step size, and it is the amplitude of the perturbation applied on the previous state of the Markov chain and, mathematically, the standard deviation of the normal distribution centred on the previous sampled feature from which we sampled the new candidate.

The new candidate is accepted or rejected based on the rate established by Eq. (8) and employing the same exact density function f drawn from the GP model, meaning that the GP noise seed must be kept unchanged during the sampling of one cluster realisation. The last condition is that it is essential not to break the detailed balance of the Metropolis-Hastings algorithm.

One can notice that there is no conditioning on the global properties of the cluster and the sampler 'builds' the cluster by sampling stars individually. Therefore, this direct approach works similar to a black box that relies solely on the correlations captured by the GP model.

3.2. Physics-informed sampling

Sampling star systems, especially binary systems in close interaction, by generating new samples in a seven-dimensional space is challenging due to the low probability of proposing candidate stars in proximity within the physical space. This may result in erroneous modelling of star clusters at small scales, even if the sampling algorithm successfully replicates the density function given by the GP. Our solution to this problem is based on incorporating physical laws to the sampling process by focusing on binary stars creating chains of pairs based on the nearest neighbours and learning the distribution over the potential and kinetic energies of these pairs. Thus, the cluster is reduced to a chain of this kind, and from this chain we can reconstruct other clusters based on similar chains.

This approach substitutes learning the full distribution of the mutual potential energies that results from $N(N-1)/2$ pairs of stars. This can be categorised as feature dimensionality reduction through feature selection (Jia et al. 2022) based on the heuristic that most of the binding energy is concentrated in the binary systems of the cluster (Torniamenti et al. 2021).

We defined for each pair a feature set based on the mutual potential energy of the stars and their kinetic energies, $\theta_i = (U_{i,i+1}, K_i, K_{i+1})$ with i in $\overline{1,N-1}$. In this way, following the nearest neighbours chain, we get $N-1$ sets of features in the energy space. We considered that these sets of energy values fully define the energy state of the nearest neighbours chain. New energy states are sampled making use of a GP model trained on the probability density function of the energy space.

The sampling scheme therefore starts by proposing new energy states. The first step of the solution is the EMCMC algorithm that samples the energy state of the nearest neighbours chain. The energy state at the n-th iteration, $S_n = \{\theta_1^{(n)}, \theta_2^{(n)}, ..., \theta_N^{(n)}\}$, implies the mutual potentials of the nearest neighbours, $U_{12}^{(n)}$, $U_{23}^{(n)}$, ..., $U_{N-1,N}^{(n)}$, and all the kinetic energies, $K_1^{(n)}$, $K_2^{(n)}$, ..., $K_N^{(n)}$. As before, we sampled for a random i a new candidate $\theta_i^{(n+1)} = (U_{i,i+1}^{(n+1)}, K_i^{(n+1)}, K_{i+1}^{(n+1)})$ from a normal distribution based on the values of the previous iteration. This also implies a change for $\theta_{i-1}^{(n)}$ and $\theta_{i+1}^{(n)}$. In this case, applying Eqs. (5) and (6) leads to the following acceptance rate:

$$\alpha = \min\left(1, \frac{f\left(\theta_{i-1}^{(n+1)}\right) f\left(\theta_i^{(n+1)}\right) f\left(\theta_{i+1}^{(n+1)}\right)}{f\left(\theta_{i-1}^{(n)}\right) f\left(\theta_i^{(n)}\right) f\left(\theta_{i+1}^{(n)}\right)}\right), \tag{9}$$

where f is the probability density function drawn from the GP model trained on the energy space. We note that at the boundaries, only two sets of features are subject to change.

Using the new sampled chain, one needs to 'reconstruct' the cluster by defining its stars as entities with a position, velocity, and mass. The magnitudes of the relative distances between the stars, $\mathbf{r_{i,i+1}} = \mathbf{r_{i+1}} - \mathbf{r_i}$, and the velocities are derived according to the physical laws of Newtonian gravity,

$$|\mathbf{r_{i,i+1}}| = -\frac{GM_iM_{i+1}}{U_{i,i+1}} \tag{10}$$

$$|\mathbf{v_{i+1}}| = \sqrt{\frac{2K_{i+1}}{M_{i+1}}}, \tag{11}$$

where G is the universal gravitational constant and M_i and M_{i+1} are the masses.

Following the chain sequentially implies that the properties of the star i are known when deriving those of the next star in the chain (the star $i+1$). The mass of the next star, M_{i+1}, is sampled independently with the help of another GP model trained only on mass distributions. One can then determine the magnitudes of $\mathbf{r_{i,i+1}}$ and $\mathbf{v_{i+1}}$. The next step is to explore the full seven-dimensional feature space of the stars to determine the optimal directions of $\mathbf{r_{i,i+1}}$ and $\mathbf{v_{i+1}}$. For this, we used a GP model trained on the phase-space, $\mathcal{GP}(\boldsymbol{r}, \boldsymbol{v})$. The exploration consists of proposing several N_c candidate directions for each vector. The candidates were chosen randomly from a uniform distribution. Using the GP model predictions, we chose the most probable position in the phase-space to determine the next star $i+1$. The procedure was repeated until the end of the chain.

In Algorithm 1, we provide the pseudocode of the proposed algorithm that works based on three GP models: $\mathcal{GP}(\theta)$, $\mathcal{GP}(M)$, and $\mathcal{GP}(\boldsymbol{r}, \boldsymbol{v})$. The function 'randomDirection' is employed to generate a random direction based on two random generated

numbers, $\mathcal{R}_\theta$ and $\mathcal{R}_\phi$, which are used to generate pairs of angles (θ, ϕ) that correspond to directions sampled uniformly on a three dimensional sphere. These directions are used to generate new candidates.

We define two hyper-parameters, *maxJump* and N_c. The first parameter is used to control the maximum distance between the pairs of stars. In this way, the mass choice is conditioned on the spatial distribution of the cluster. Without this conditioning, the chain can be easily broken into multiple parts. This would lead to generating several smaller clusters. The other parameter, N_c, is the number of star candidates we generate randomly based on the values of $|\boldsymbol{r}_{ij}|$ and $|\boldsymbol{v}_j|$. We use $\mathcal{GP}(\boldsymbol{r}, \boldsymbol{v})$ afterwards to find the best candidate, that is, the star with the most probable position in the phase-space. As N_c is large enough, the resulting star features always point towards the densest region of the phase-space that is attainable at a given moment.

Algorithm 1 EMCMC algorithm in pseudocode to find and select new star candidates for the generated cluster.

Require: $\mathcal{GP}(\theta)$, $\mathcal{GP}(M)$, $\mathcal{GP}(\boldsymbol{r}, \boldsymbol{v})$, `randomDirection`, N, N_c, *maxJump*
  cluster ← empty array to store the parameters of N stars
  cluster[0] ← initialize parameters of the first star
  for i `from 0 to N-1`
    $U_{i,i+1}, K_i, K_{i+1} \leftarrow MCMC(\mathcal{GP}_{(\theta)})$
    $M_i, \boldsymbol{r}_i, \boldsymbol{v}_i \leftarrow cluster[i]$
    $|\boldsymbol{r}_{i,i+1}| \leftarrow \infty$
    while $|\boldsymbol{r}_{i,i+1}| > maxJump$
      $M_{i+1} \leftarrow MCMC(\mathcal{GP}_{(M)})$
      $|\boldsymbol{r}_{i,i+1}| \leftarrow \frac{M_i m_{i+1}}{U_{i,i+1}}$
    end while
    $|\boldsymbol{v}_{i+1}| \leftarrow \sqrt{\frac{2K_{i+1}}{m_{i+1}}}$
    candidates ← empty array to store N_c candidates
    for `each` *candidate* `in` *candidates*
      $\boldsymbol{\lambda}_r \leftarrow$ `randomDirection()`
      $\boldsymbol{\lambda}_v \leftarrow$ `randomDirection()`
      $\boldsymbol{r}_{i+1} = \boldsymbol{r}_i + |\boldsymbol{r}_{i,i+1}|\boldsymbol{\lambda}_r$
      $\boldsymbol{v}_{i+1} = |\boldsymbol{v}_{i+1}|\boldsymbol{\lambda}_v$
      $candidate \leftarrow M_{i+1}, \boldsymbol{r}_{i+1}, \boldsymbol{v}_{i+1}$
    end for
    $probs \leftarrow \mathcal{GP}_{(\boldsymbol{r},\boldsymbol{v})}(candidates)$
    $bestCandidateIdx \leftarrow argmax(probs)$
    $cluster[i+1] \leftarrow candidates[bestCandidateIdx]$
  end for

4. Results and discussions

4.1. Dataset

We utilized a dataset consisting of ten clusters extracted from hydrodynamical simulations conducted by Ballone et al. (2020). These simulations were designed to accurately reproduce not only the clumpiness but also the fractal nature observed in star-forming regions. Each cluster in the dataset has approximately 2500 to 4000 stars, with a cumulative mass ranging between 4000 and 40 000 $M_\odot$ (solar masses). For our analysis, we divided this dataset into a training subset, consisting of seven clusters, and a validation subset made up of the remaining three clusters.

Table 1. Training results.

Model	Best epoch	Time/epoch (s)	l	σ_n
$\mathcal{GP}(M, \boldsymbol{r}, \boldsymbol{v})$	32	7.2	1.6205	0.1445
$\mathcal{GP}(\boldsymbol{r}, \boldsymbol{v})$	53	6.1	1.4506	0.0752
$\mathcal{GP}(U_{ij}, K_i, K_j)$	12	1.4	0.5034	0.3965
$\mathcal{GP}(M)$	69	1.0	1.5523	0.0040

Notes. Columns: (1) Model; (2) best epoch; (3) training time per epoch; (4) scale length, l; (5) noise level, σ_n.

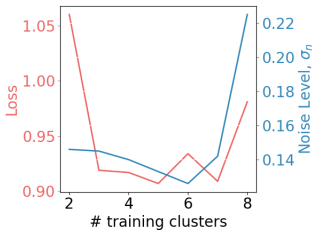

Fig. 1. Training parameters (loss and noise level) with respect to the number of clusters used in the training. One validation cluster was used in the case of two or three training clusters; otherwise, we used two validation clusters. The training parameters correspond to the model's version at early stopping.

4.2. Model training

We used the gpytorch library (Gardner et al. 2018) to implement our models. This library offers a modular and efficient framework for GPs and supports graphics processing unit (GPU) acceleration. For all of our experiments, we employed one NVIDIA GeForce RTX 3060 GPU. Our GP model was trained using the RBF kernel, employing the exact marginal log likelihood as the loss function and using the Gaussian likelihood to calculate the posterior distribution. All models were initialised with a zero-prior mean. The necessary features for model training were derived from simulation data. Our training approach for each model integrates cross-validation (Hastie et al. 2001) and early stopping (Prechelt 1996), the latter having a patience threshold set at ten epochs. The properties of each trained model are summarised in Table 1.

The models were trained on ten clusters, among which three are for validation. The training itself can be done using more or fewer clusters, depending on the context. We observed that the loss slightly increases when decreasing the number of clusters used in training (see Fig. 1), suggesting a slight decline of the model's performance. Also, we noticed a sudden increase of the loss when using eight clusters for training (and the other two for validation). This could be caused by underfitting, as suggested by the larger noise level. Therefore, we continued our experiments using a training validation ratio of seven to three.

4.3. Sampling

The sampling is based on a probability density function drawn from $\mathcal{GP}(M, \boldsymbol{r}, \boldsymbol{v})$ for DMCMC or $\mathcal{GP}(U_{ij}, K_i, K_j)$ for EMCMC. The initial state, in both cases, is initialised from a normal distribution $\mathcal{N}(\mu, \sigma)$, with μ and σ computed over the feature distributions retrieved from the simulation data.

The cluster size, that is, the number of stars in the cluster, N, is established when defining the initial state of MCMC proposals. The new candidate stars are proposed such that the state

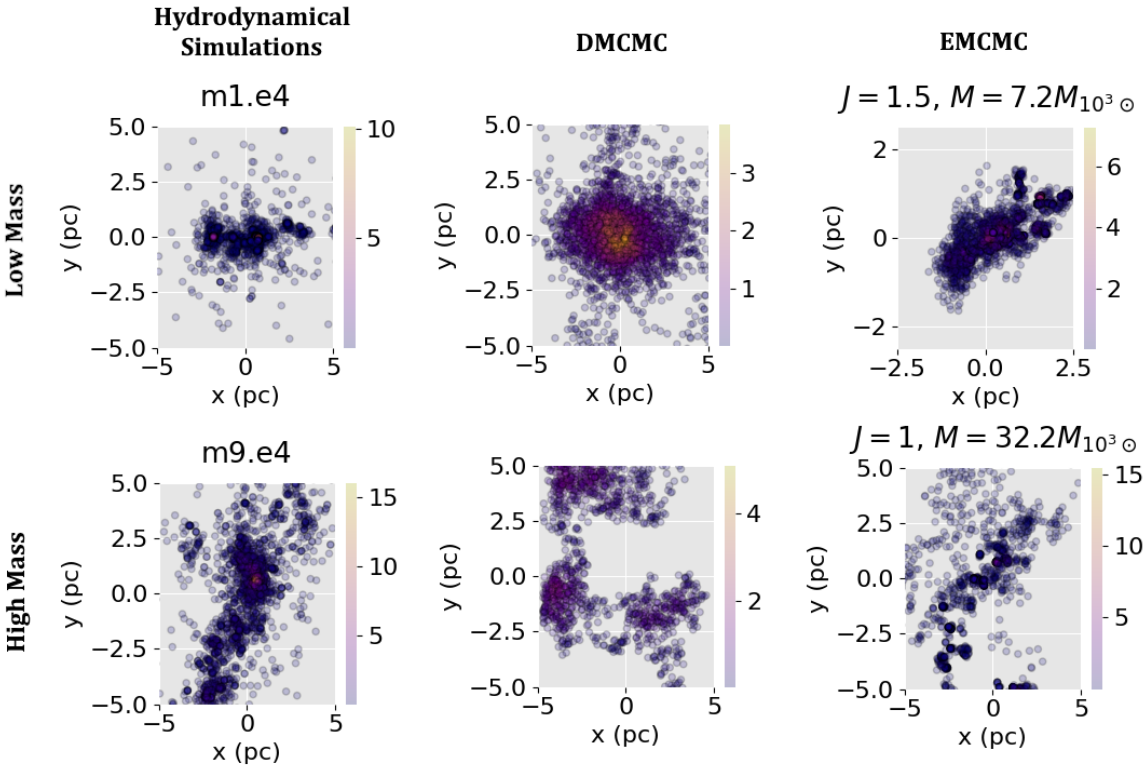

Fig. 2. Two-dimensional representation of one low-mass and one high-mass cluster generated with DMCMC and EMCMC, respectively. For comparison, the corresponding projections are illustrated also for the hydrodynamically simulated clusters *m*1.*e*4 and *m*9.*e*4 from the work of Ballone et al. (2020). The x and y coordinates are measured in parsecs. For EMCMC, we show the value of the hyper-parameter J (maxJump) and the total mass expressed in $M_{10^3\odot} = 1000$ solar masses.

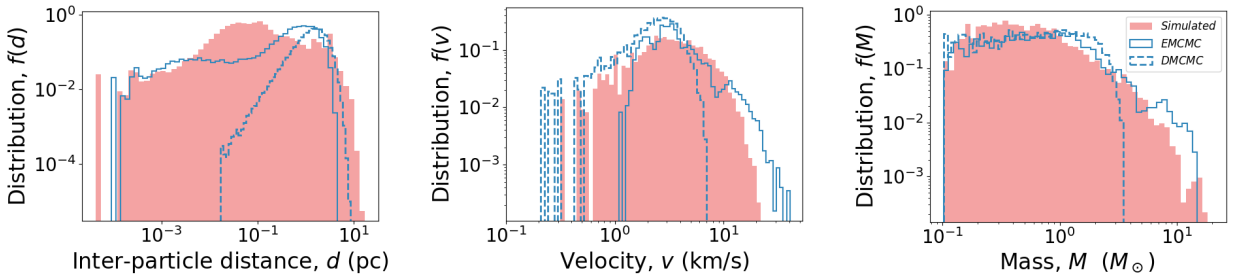

Fig. 3. Distributions of inter-particle distances, velocity, and mass for low-mass clusters with 3000 stars generated using DMCMC and EMCMC methods. The mass distribution has a lower bound of 0.1. A simulated cluster from Ballone et al. (2020) is represented for comparison.

perturbations are uniformly distributed along the stars' features. This means that we do not define a maximum number of iterations, but several accepted states, N_A. For instance, if $N_A = mN$, new features for each star will be accepted exactly m times. In this way, one can recover m cluster realisations or use the first $m - 1$ iterations as a burn-in strategy (Roy 2020).

4.4. Generated clusters

The newly generated clusters were evaluated based on the distributions of the inter-particle distance, velocity, and mass. In addition, we computed global physical quantities such as the virial ratio, total mass, and the number of stars that are part of a bound system. The evaluation was based on comparing our generated clusters to the hydrodynamically simulated ones retrieved from Ballone et al. (2020). These clusters are our baseline, and our aim was to reproduce similar features. A two-dimensional spatial projection of such clusters obtained with DMCMC and EMCMC on the Oxy plane is shown in Fig. 2.

We performed a set of experiments to sample a low-mass cluster (4000 $M_\odot$) of $N = 3000$ stars. The results are shown in Fig. 3. Comparing to the simulation data, for the inter-particle distance distribution of DMCMC, we noticed that there is a lack of stars at low inter-particle distances smaller than 10^{-2} pc. This suggests a lack of close interactions between stars (i.e. binary systems). However, when analysing the total energy of the star pairs, we observed that there is a significant number of stars that are part of bound systems. For instance, the low-mass DMCMC-generated cluster that corresponds to Figure 2 exhibits a number of 276 stars that are bound to other stars, whereas in the baseline training clusters there are between 400 and 800 binaries. According to the inter-particle distance distribution of the DMCMC clusters (Fig. 3), the orbital separations of this binary systems are large enough that gravitational encounters will likely unbind them during a cluster's evolution. This number is thus an

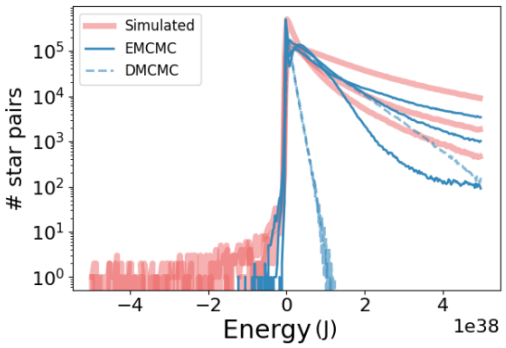

Fig. 4. Histogram with the energies of each pair of stars in the cluster. The energy for each pair was calculated as the sum of kinetic energies of the stars and their mutual potential energy. Negative energies correspond to gravitationally bound pairs. The resulting energy spectra for the clusters generated with EMCMC and DMCMC are compared to the clusters obtained via hydrodynamical simulations (Ballone et al. 2020).

upper limit to stable binaries, and the actual number of stable binaries could be significantly less, given the lack of close binaries. This is not the case for the EMCMC clusters, as there are also star pairs with low inter-particle distance that are able to keep the bound system intact.

On the other hand, the physics-informed algorithm excels at sampling clusters with a balanced spectrum of inter-particle distances, which direct sampling in a seven-dimensional feature space fails to achieve. We also note that the energy spectrum of DMCMC exhibits a linear trend that differs from those of the EMCMC-generated clusters or hydrodynamical simulations of clusters (see Fig. 4).

Other relevant physical quantities are presented in Fig. 5. In the left subfigure, it is shown how the number of stars varies with respect to the cluster's radius. We observed that there is not any specific trend in the distribution derived for the reference clusters. We noticed sudden jumps, meaning that there are regions of higher density, but there is one exception for one of the DMCMC clusters, where the number of stars increases smoothly. On the right side, we have plotted the histogram illustrating the distribution of angular momenta resulting after computing the magnitude of angular momentum of each star as

$$|\mathbf{L}| = \sqrt{(r_y v_z - r_z v_y)^2 + (r_z v_x - r_x v_z)^2 + (r_x v_y - r_y v_x)^2}. \quad (12)$$

Here, we note that DMCMC clusters lack stars with high angular momentum. Probably, this happens because of the lower sampling rate of stars with high velocity magnitudes (see Fig. 3).

We present a summary of additional experiments in Table 2. We show the global properties (virial ratio, estimated number of binary systems, and total mass) of the six realisations obtained via DMCMC and EMCMC. We performed the experiments in order to reproduce clusters with a low, intermediate, and high total mass.

When examining virial ratios, that is, the total kinetic energy divided by half of the total binding energy, we observed that the Metropolis algorithm delivers varied dynamical states. The EMCMC approach consistently produces realistic clusters; 60% of the clusters we sampled fall within a virial ratio between one and two, which is consistent with the training samples.

All the experiments were aimed at generating clusters containing 3000 stars, which would allow us to provide computational performance comparisons. The number $N_A = 3000$ was chosen arbitrarily; one can try different cluster sizes. However, the number of stars in the clusters should be the same order of magnitude as those used in the training (2000–5000 stars). More massive clusters could be sampled, but the framework performance has been tested primarily for lower-mass clusters, so applying it to significantly larger clusters might necessitate retraining or fine-tuning the model to ensure that it handles the increased star count effectively. Sampling with DMCMC takes around 15 minutes for $N_A = 3000$ accepted samples, while EMCMC samples 3000 new accepted states during the same time, but it needs additional time, around 10 minutes, to reconstruct the cluster from the chain. The running time will scale linearly with the number of stars, as the sampling is done sequentially. One would expect EMCMC sampling to take longer, as we are estimating the probability density for three stars at each iteration. However, the acceptance rate is also three times higher for EMCMC. The experiments were done using an NVIDIA GeForce RTX 3060 GPU and a 12th Gen Intel Core i7-12700H CPU at 4.70 GHz. The training, sampling, and evaluation routines are available online on GitHub[1], and there one can find several results that we have generated and the models we trained to obtain these results.

5. Conclusions

We have introduced a learning framework to generate new realisations of positions, velocities, and masses of star clusters to be used as initial conditions in N-body simulations. This framework allowed us to bypass the computational bottleneck represented by hydrodynamical simulations of star formation in molecular clouds, when provided an adequate training set. Based on well understood GPs, our method yields predictable results and can be readily integrated into a simulation pipeline.

The learning framework consists of two steps. First, a statistical model of the data is constructed through the GP, and second, new clusters are generated by sampling from the target distribution. The probability density function of the star features was employed to train the GP model. We tackled the problem in two ways: by sampling directly in the seven-dimensional feature space of the stars (DMCMC) and by sampling in a specific energy space (EMCMC). That was followed by reconstructing the cluster using the proposed algorithm.

In the pursuit of sampling clusters with realistic spatial distributions and dynamical states, the EMCMC algorithm has proven to be particularly effective, especially in addressing the crucial distribution of inter-particle distances, whereas DMCMC led to undesired distance distributions. The latter method and other conventional samplers work similar to a 'black box' in this case, relying only on the correlations learned by the GP model. The main difference between the two approaches is that EMCMC incorporates physical principles. This physics-informed algorithm ensures that the generated samples are not only mathematically plausible but that they also possess physical significance.

The DMCMC experiments showed that knowing the probability density function of the mass, position, and velocity is not sufficient to fully reproduce a cluster realisation. These properties are a consequence of physical interactions, and trying to reproduce them by sampling an estimated probability density function proved to be challenging due to a lack of binary stars. In this context, the learned distribution can inform us about the possible location of a star in the feature space, but it cannot

[1] https://github.com/prodangp/star-clusters-gen

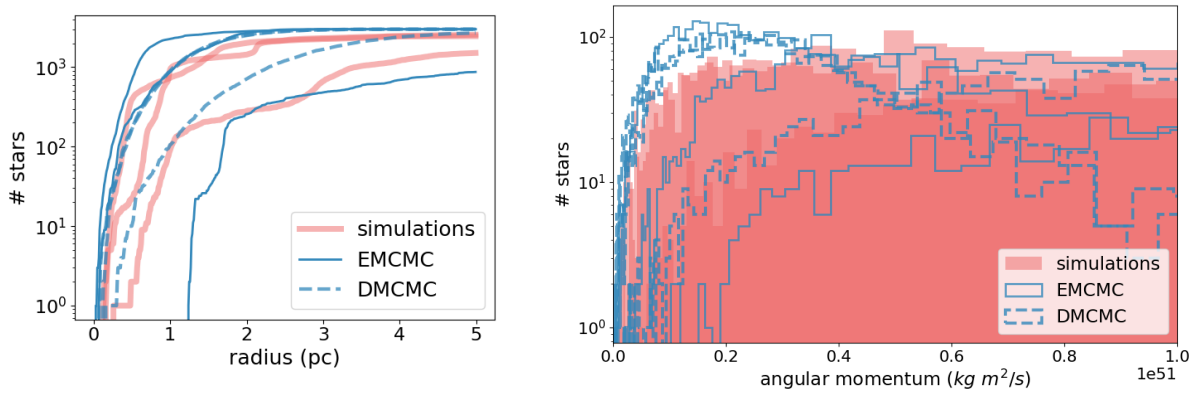

Fig. 5. The distributions of key physical paramters for the new sampled clusters. The left subfigure shows the number of stars as a function of the distance to the cluster's centre of mass for a set of simulated clusters, DMCMC and EMCMC (three clusters each). On the right subfigure is plotted a histogram of the angular momentum distribution of the stars for the same sets of clusters as in the left subfigure. The angular momentum is measured in the units of the international system.

Table 2. Monte Carlo sampling results using EMCMC and DMCMC algorithms.

#	Method	M_{lower} ($M_\odot$)	Step size	acc. rate	α_{vir}	# binaries	M (10^3 $M_\odot$)
1		0.1	1.50	0.06	1.47	214	3.7
2	DMCMC	0.3	1.50	0.07	3.52	75	10.2
3		0.5	1.50	0.07	1.26	261	24.0
4		0.1	0.2	0.33	1.69	2089	7.2
5	EMCMC	0.2	0.2	0.5097	1.08	1334	13.5
6		0.5	0.4	0.35	1.21	1401	32.2

Notes. The generated clusters contain 3000 stars. Columns: (1) #, cluster index; (2) method; (3) the lower bound for the mass distribution, M_{lower}, in solar masses ($M_\odot$) ; (4) the step size in the Metropolis algorithm; (5) the acceptance rate; (6) the virial ratio; (7) the number of binaries; (8) the total mass of the sampled cluster.

determine whether that star is part of a binary system, and it cannot reproduce its companion star. In contrast, with EMCMC, we did not change the sampling algorithm, but we sampled from a different target distribution that helps in answering both questions, namely, where the star is and whether it is a binary. The EMCMC generates clusters that not only match statistical properties but also reflect physical realities, such as binary star formation, making it a more robust and significant method for studies of stellar evolution and cluster dynamics.

Acknowledgements. This work acknowledges financial support from the European Research Council for the ERC Consolidator grant DEMOBLACK, under contract no. 770017 (PI: Mapelli). MM and ST also acknowledge financial support from the German Excellence Strategy via the Heidelberg Cluster of Excellence (EXC 2181 – 390900948) STRUCTURES.

References

Aarseth, S. J. 2003, Gravitational *N*-Body Simulations (Cambridge, UK: Cambridge University Press)
Ballone, A., Mapelli, M., Di Carlo, U. N., et al. 2020, MNRAS, 496, 49
Ballone, A., Torniamenti, S., Mapelli, M., et al. 2021, MNRAS, 501, 2920
Chib, S., & Greenberg, E. 1995, Am. Statist., 49, 327
Chung, C., Pasquato, M., Lee, S.-Y., et al. 2019, ApJ, 883, L31
Cranmer, K., Brehmer, J., & Louppe, G. 2020, PNAS, 117, 30055
Dall'Amico, M., Mapelli, M., Di Carlo, U. N., et al. 2021, MNRAS, 508, 3045
Eckhardt, R. 1987, Los Alamos Sci., 15, 131
Evans, Neal J., I. 1999, ARA&A, 37, 311
Gardner, J. R., Pleiss, G., Bindel, D., Weinberger, K. Q., & Wilson, A. G. 2018, GPyTorch: Blackbox Matrix-Matrix Gaussian Process Inference with GPU Acceleration
Griffiths, R.-R. 2022, Applications of Gaussian Processes at Extreme Lengthscales: From Molecules to Black Holes (Apollo - University of Cambridge Repository), https://doi.org/10.17863/CAM.93643
Hastie, T., Tibshirani, R., & Friedman, J. 2001, The Elements of Statistical Learning, Springer Series in Statistics (New York, NY, USA: Springer New York Inc.)
Iorio, G., Mapelli, M., Costa, G., et al. 2023, MNRAS, 524, 426
Jia, W., Sun, M., Lian, J., et al. 2022, Complex Intell. Syst., 8, 2663
Krause, M. G. H., Offner, S. S. R., Charbonnel, C., et al. 2020, Space Sci. Rev., 216, 64
Krumholz, M. R., McKee, C. F., & Bland-Hawthorn, J. 2019, Annu. Rev. Astron. Astrophys., 57, 227
Lada, C. J., & Lada, E. A. 2003, ARA&A, 41, 57
Lueckmann, J.-M., Boelts, J., Greenberg, D. S., Gonçalves, P. J., & Macke, J. H. 2021, Benchmarking Simulation-Based Inference
Mack, Y. P., & Rosenblatt, M. 1979, J. Multivariate Anal., 9, 1
Malmberg, D., Davies, M. B., & Heggie, D. C. 2011, MNRAS, 411, 859
Pang, X., Yu, Z., Tang, S.-Y., et al. 2021, ApJ, 923, 20
Pang, X., Tang, S.-Y., Li, Y., et al. 2022, ApJ, 931, 156
Parker, R. J., & Quanz, S. P. 2012, MNRAS, 419, 2448
Pfalzner, S., Davies, M. B., Gounelle, M., et al. 2015, Phys. Scr, 90, 068001
Pichardo, B., Moreno, E., Allen, C., et al. 2012, AJ, 143, 73
Prechelt, L. 1996, in Neural Networks
Rasmussen, C. E. 2004, Gaussian Processes in Machine Learning, eds. O. Bousquet, U. von Luxburg, & G. Rätsch (Berlin, Heidelberg: Springer), 63
Rasmussen, C. E., & Williams, C. K. I. 2006, Gaussian Processes for Machine Learning, Adaptive Computation and Machine Learning (MIT Press), I, 1
Rastello, S., Mapelli, M., Di Carlo, U. N., et al. 2021, MNRAS, 507, 3612
Robert, C., & Casella, G. 2011, Statist. Sci., 26
Roy, V. 2020, Annu. Rev. Statist. Appl., 7, 387
Spurzem, R., & Kamlah, A. 2023, Living Rev. Computat. Astrophys., 9, 3
Spurzem, R., Giersz, M., Heggie, D. C., & Lin, D. N. C. 2009, ApJ, 697, 458
Staff at the National Astronomy & Ionosphere Center 1975, Icarus, 26, 462
Torniamenti, S., Ballone, A., Mapelli, M., et al. 2021, MNRAS, 507, 2253
Torniamenti, S., Pasquato, M., Di Cintio, P., et al. 2022, MNRAS, 510, 2097
Wang, J. 2022, An Intuitive Tutorial to Gaussian Processes Regression