PROCEEDINGS A

royalsocietypublishing.org/journal/rspa

Research

Cite this article: Liu Y, McCalla SG, Schaeffer H. 2023 Random feature models for learning interacting dynamical systems. *Proc. R. Soc. A* **479**: 20220835.
https://doi.org/10.1098/rspa.2022.0835

Received: 11 December 2022
Accepted: 15 June 2023

Subject Areas:
applied mathematics, artificial intelligence

Keywords:
interacting systems, sparsity, randomization, data discovery, random feature method

Authors for correspondence:
Scott G. McCalla
e-mail: scott.mccalla@montana.edu
Hayden Schaeffer
e-mail: hayden@math.ucla.edu

Random feature models for learning interacting dynamical systems

Yuxuan Liu[1], Scott G. McCalla[2] and Hayden Schaeffer[1]

[1]Mathematics, University of California, Los Angeles, CA, USA
[2]Mathematical Sciences, Montana State University, Bozeman, MT, USA

SGM, 0000-0003-0093-4658; HS, 0000-0003-1379-1238

Particle dynamics and multi-agent systems provide accurate dynamical models for studying and forecasting the behaviour of complex interacting systems. They often take the form of a high-dimensional system of differential equations parameterized by an interaction kernel that models the underlying attractive or repulsive forces between agents. We consider the problem of constructing a data-based approximation of the interacting forces directly from noisy observations of the paths of the agents in time. The learned interaction kernels are then used to predict the agents' behaviour over a longer time interval. The approximation developed in this work uses a randomized feature algorithm and a sparse randomized feature approach. Sparsity-promoting regression provides a mechanism for pruning the randomly generated features which was observed to be beneficial when one has limited data, in particular, leading to less overfitting than other approaches. In addition, imposing sparsity reduces the kernel evaluation cost which significantly lowers the simulation cost for forecasting the multi-agent systems. Our method is applied to various examples, including first-order systems with homogeneous and heterogeneous interactions, second-order homogeneous systems, and a new sheep swarming system.

THE ROYAL SOCIETY
PUBLISHING

1. Introduction

Agent-based dynamics typically employ systems of ordinary differential equations (ODEs) to model a wide range of complex behaviours such as particle dynamics, multi-body celestial mechanics, flocking dynamics, species interactions, opinion dynamics and many other physical or biological systems. Fundamentally, agent-based dynamical systems model the collective behaviour between multiple objects of interest by using a kernel that represents their interactions and thus provide equations that govern their motion. They are able to reproduce a variety of collective phenomena with a minimal set of parameters which make them especially accessible for modellers [1–5]. Such models of collective behaviour are numerically slow to solve and difficult to analyse which often leads to issues in trying to extract parameters, i.e. the kernels. This difficulty is typically circumvented by passing to continuum limits in order to aid computations [6] and understand facets of the dynamics of the discrete system such as emergent patterning [7–9] or properties of the asymptotic dynamics [10–16]. These interaction kernels provide information on the behaviour between agents as well as the agents' self-driven forces. However, in practice, these kernels may not be fully known and thus one needs to be able to approximate them from observation of the agents' states. In this work, we develop a sparse random feature model (RFM) that uses a randomized Gaussian basis to approximate the interaction kernels from the data and use it to forecast future states and behaviour.

Learning dynamical systems from data is an important modelling problem in which one approximates the underlying equations of motion governing the evolution of some unknown system [17,18]. This is essentially a model selection and parameter estimation problem, where the equations of interest are those that define a differential equation. One of the main challenges is to construct methods that do not overfit on the data, can handle noise and outliers, and can approximate a wide enough class of dynamics. While there are many works in the last decade on this topic, we highlight some of the related work along the direction of interpretable models, i.e. those that provide not only an approximation but a meaningful system of equations. For a detailed overview of the approximation and inference of physical equations from data see [19] and the citations within. The SINDy algorithm [20] is a popular method for extracting governing equations from data using a sparsity-promoting method. The key idea is to learn a sparse representation for the dynamical system using a dictionary of candidate functions (e.g. polynomials or trigonometric functions) applied to the data. This approach has lead to many related algorithms and applications, see [21–34] and the reference within. The sparse optimization framework for learning governing equations was proposed in [35] along with an approach to discovering partial differential equations using a dictionary of derivatives. The sparse optimization and compressive sensing approach for learning governing equations include ℓ^1 based approaches for high-dimensional ODE [36–38] and noisy robust ℓ^0 approaches using the weak form [39]. Another approach is the operator inference technique [40] which can be used to approximate high-dimensional dynamics by learning the ODEs that govern the coefficients of the dynamics projected onto a linear subspace of small dimension. This results in a model for the governing equation in the reduced space that does not require computing terms in the full dimension. While the methods of [20,35,40,41] can be applied to a wide range of problems, they are not optimal for the problems considered here.

A non-parametric approach for learning multi-agent systems was developed in [42] based on the assumption that the interaction kernel $g : \mathbb{R}_+ \to \mathbb{R}$ is a function of the pairwise distances between agents. Their method builds a piecewise polynomial approximation of the interaction kernel by partitioning the set of possible distances (i.e. the input space for g) and (locally) estimating the kernel using the least-squares method (see §2). The training problem is similar in setup to other learning approaches for dynamical systems [20,35,40,43] which use trajectory-based regression; however, since the ODE takes a special form, the training problem becomes a one-dimensional regression problem and thus piecewise estimators do not incur the curse-of-dimensionality. The theoretical results in [42] focused on the first-order homogeneous case (one type of agent), but recent works generalize and extend the theory and numerical model

to heterogeneous systems (i.e. different types of agents and different interaction kernels) [44], stochastic systems [45] and second-order systems [46]. In [47], the accuracy and consistency of the approach from [42] was numerically verified on a range of synthetic examples, including predicting emergent behaviours at large timescales. A theoretical analysis of the identifiability of interaction systems was presented in [48] which relies on a coercivity condition that is related to the conditions in [42,44,45].

In this work, we will construct an RFM for learning the interaction kernel and thus the governing system for multi-agent dynamics. RFMs are a class of non-parametric methods used in machine learning [49–51] which are related to kernel approximations and shallow neural networks. The standard RFM is a shallow two-layer fully connected neural network whose (sole) hidden layer is randomized and then fixed [49–51] while its output layer is trained with some optimization routine. From the perspective of dictionary learning, an RFM uses a set of candidate functions that are parameterized by a random weight vector as opposed to the standard polynomial or trigonometric basis whose constructions use a fixed set of functions. Theoretical results on RFM establish various error estimates, including uniform error estimates [50], generalization bounds related to an L^∞ like space [51], and generalization bounds for reproducing kernel Hilbert spaces [52–55]. A detailed comparison between the approaches, training problems and results can be found in [56].

2. Learning multi-agent systems

Consider the n-agent interacting system, define the n time-dependent state variables $\{\mathbf{x}_i(t)\}_{i=1}^n$, $\mathbf{x}_i(t) \in \mathbb{R}^d$ for all $1 \le i \le n$, whose dynamics are governed by the following system of ODEs:

$$\left.\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t}\mathbf{x}_i(t) &= \frac{1}{n}\sum_{i'=1}^{n} g(||\mathbf{r}_{i',i}(t)||_2)\,\mathbf{r}_{i',i}(t) \\ \text{and} \qquad \mathbf{r}_{i',i}(t) &= \mathbf{x}_{i'}(t) - \mathbf{x}_i(t), \end{aligned}\right\} \tag{2.1}$$

where the interaction kernel is $g : \mathbb{R}_+ \to \mathbb{R}$. The velocity of each agent $\mathbf{x}_i(t)$ defined by (2.1) depends on the g-weighted average of distances to all other agents. Note that the velocity only depends on the relative distances between agents and not on their overall locations in space or time. We will assume that g is continuously differentiable and has either compact support or has sufficient decay, which guarantees the well-posedness of (2.1). In addition, we assume that g is a positive definite function, which leads to a specific representation (see §2(a)) that is used to derive the proposed model although these assumption will be relaxed for applications. For simplicity, denote $\mathbf{x}(t) := (\mathbf{x}_1(t), \ldots, \mathbf{x}_n(t))$ and $\mathbf{r}_{:,i}(t) = (\mathbf{x}_1(t) - \mathbf{x}_i(t), \ldots, \mathbf{x}_n(t) - \mathbf{x}_i(t))$. Equation (2.1) can be derived from the gradient flow of the potential energy $U(\mathbf{x}(t)) = \sum_{i \ne i'} G(\mathbf{r}_{i',i}(t))$, where $g(r) := G'(r)/r$. Such gradient flow systems have proven effective as a minimal framework to model a variety of chemical and living systems such as for micelle formation [57] or aggregation and swarming of living things, even without the addition of stochastic noise. The pairwise interaction kernel can be derived from physical constraints for the potential energy in non-living systems, but this type of first principles approach becomes unreasonable to derive the governing equations for biological systems.

This section details our method. First, to train (2.1) from data, we develop a random feature approach for representing the unknown interacting kernel g. Then, the learning problem is written as a linear system in order to identify the coefficients of the model. This is done using either the least-squares approach or a sparse learning problem.

(a) New random features

Motivated by the characterization of radial positive definite functions on $\mathbb{R}^d$ from [58] and the earlier work on spherical RFMs [59], we propose approximating the interacting kernel g in (2.1) using a random radial feature space defined by Gaussians centred at zero.

royalsocietypublishing.org/journal/rspa Proc. R. Soc. A 479: 20220835

Theorem 2.1 ([58]). *A continuous function $g:\mathbb{R}_+\to\mathbb{R}$ is radial and positive definite on $\mathbb{R}^d$ for all d if and only if it is of the form $g(r)=\int_0^\infty e^{-r^2\omega^2}\,d\nu(\omega)$ where ν is a finite non-negative Borel measure on $\mathbb{R}_+$.*

We consider a weighted integral representation, namely, we assume that $g(r)=\int_0^\infty \alpha(\omega)e^{-r^2\omega^2}\,d\nu(\omega)$, then (2.1) becomes

$$\begin{aligned}\frac{d}{dt}\mathbf{x}_i(t)&=\frac{1}{n}\sum_{i'=1}^{n} g(||\mathbf{r}_{i',i}(t)||_2)\,\mathbf{r}_{i',i}(t)\\&=\frac{1}{n}\left(\sum_{i'=1}^{n}\int_0^\infty \alpha(\omega)\,e^{-||\mathbf{r}_{i',i}(t)||_2^2\,\omega^2}\,d\nu(\omega)\right)\mathbf{r}_{i',i}(t)\\&=\int_0^\infty\left(\frac{\alpha(\omega)}{n}\sum_{i'=1}^{n} e^{-||\mathbf{r}_{i',i}(t)||_2^2\,\omega^2}\,\mathbf{r}_{i',i}(t)\right)d\nu(\omega)\\&=\int_0^\infty \phi(\mathbf{r}_{1,i}(t),\dots,\mathbf{r}_{n,i}(t),\omega)\,d\nu(\omega)\\&=:F(\mathbf{r}_{1,i}(t),\dots,\mathbf{r}_{n,i}(t)),\end{aligned}\tag{2.2}$$

for all $i\in[n]:=\{1,\dots,n\}$. In (2.2), the integrand ϕ is parameterized by the scalar $\omega\in\mathbb{R}_+$ with respect to the measure ν. To approximate (2.2), we build a set of N random features by sampling ω from a user defined probability measure θ, i.e. $\omega_k\sim\theta(\omega)$ for $k\in[N]$. The function F does not explicitly depend on the agents' locations or time and instead is a function of the radial distances. Thus for simplicity consider $F(\mathbf{r}_1,\dots,\mathbf{r}_n)$, i.e. a function of n inputs, then the random feature approximation for this problem becomes

$$F_N(\mathbf{r}_1,\dots,\mathbf{r}_n)=\frac{1}{N}\sum_{k=1}^{N} c_k\,\phi_k(\mathbf{r}_1,\dots,\mathbf{r}_n),\tag{2.3}$$

where $\mathbf{c}=(c_1,\dots,c_N)$ are the coefficients of the expansion of F_N with respect to the proposed radial feature space

$$\left\{\phi_k\,\middle|\,\phi_k(\mathbf{r}_1,\dots,\mathbf{r}_n)=\frac{\alpha(\omega_k)}{n}\sum_{i=1}^{n} e^{-||\mathbf{r}_i||_2^2\,\omega_k^2}\,\mathbf{r}_i\right\}.\tag{2.4}$$

The collection of ϕ_k forms the dictionary used in the learning algorithm. The weight $\alpha(\omega)$ is chosen to normalize the basis functions ϕ_k, i.e. we set

$$\alpha(\omega)=\left(\int_0^\infty\frac{1}{n}\sum_{i=1}^{n} e^{-||\mathbf{r}||_2^2\,\omega^2}\,r_j\,dr_1,\dots,dr_n\right)^{-1}=\frac{2^n\omega^{n+1}}{\pi^{(n-1)/2}}.\tag{2.5}$$

The motivation for rescaling the terms is to avoid the ill-conditioning which occurs for larger ω_k, that is, the Gaussians become nearly zero which leads to poor generalization. We observed that rescaling by an increasing function of ω, e.g. ω^p for $p\geq 1$ was sufficient to obtain meaningful results; however, for theoretical consistency, we use the normalization factor (2.5). Although the random feature representation for the interaction kernel g is similar to [59], the approximation to F differs since we are considering a vectorized system of n-agents and a different functional form.

In figure 1, we compare the formulation (2.3) with random Fourier features (RFF) [49–51]. The global and oscillatory nature of RFF does not match the typical behaviour of the interaction kernels, i.e. the specific growth and decay properties either near the origin or in far-field. This leads to a smooth but oscillatory approximation of the kernel that causes a large accumulation of errors when forecasting the states using the learned dynamical system. In the zoomed-in plot (for region $r\in[40,60]$), we see that the RFF produces a non-monotone and 'noisy' fit. In this example, we see that our approach outperforms the RFF by matching the structural assumptions.

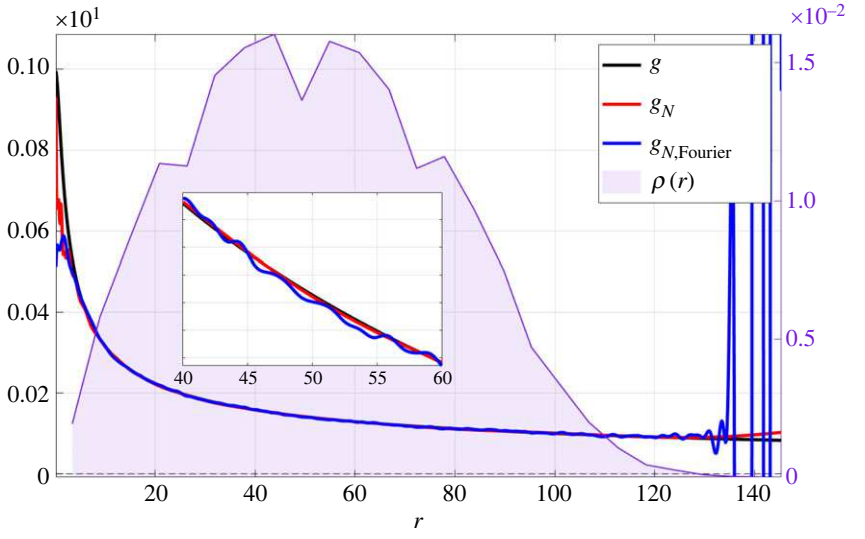

Figure 1. Illustrative example. Radial features versus Fourier features: the plot includes the true interaction kernel ($g(r) = (1 + r^2)^{-(1/4)}$, the black curve) and the learned interaction kernels using the radial features defined by (2.3) (the red curve) and the random Fourier features (blue curve). The purple area displays the empirical distribution of the radial distances estimated on the trajectories denoted by $\rho(r)$. See §4 for parameters and details.

(b) Least-squares learning problem

The training problem is to approximate F in (2.2) by F_N from (2.3) given a set of discrete trajectory paths. That is, given J observation timestamps $\{t_j\}_{j\in[J]}$ with $t_j \in [0, T]$ and $t_j < t_{j+1}$ for all $j \in [J]$ and a total of L initial conditions (IC) $\mathbf{x}^\ell(0) = (\mathbf{x}_1^\ell(0), \ldots, \mathbf{x}_n^\ell(0))$, we observe L trajectories indexed by $\ell \in [L]$ and stored as $\mathbf{X}^\ell = [\mathbf{x}_i^\ell(t_j)]_{j\in[J],i\in[n]} \in \mathbb{R}^{dnJ}$. The entire training set is the concatenation of the L trajectories $\mathbf{X} = [\mathbf{X}^\ell]_{\ell\in[L]} \in \mathbb{R}^{n_{\text{tot}}}$, where $n_{\text{tot}} = dnJL$, i.e. the product of the dimension of each agent, the number of agents, the number of timestamps and the total number of trajectories. The IC $\mathbf{x}^\ell(0)$ are drawn independently from a prescribed probability measure $\boldsymbol{\mu}_0$ on $\mathbb{R}^{dn}$. The velocity can be estimated by a finite difference method and concatenated to form the velocity dataset $\mathbf{V} = [\mathbf{V}_{i,j}^\ell]_{\ell\in[L]} \in \mathbb{R}^{n_{\text{tot}}}$, where $\mathbf{V}^\ell$ is an approximation to $\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{x}_i^\ell(t_j)$. We used the central difference formula in all examples unless otherwise stated.

To train the system, we want to minimize the risk with respect to the unknown non-negative Borel measure ν

$$\mathcal{L}(\nu) := \frac{1}{nLT} \sum_{i\in[n],\ell\in[L]} \int_0^T \left\| \frac{\mathrm{d}}{\mathrm{d}t}\mathbf{x}_i^\ell(t) - F(\mathbf{r}_{:,i}^\ell(t); \nu) \right\|_2^2 \mathrm{d}t,$$

which measures the error between the velocity and the RHS of the ODE (2.2), where $F(\cdot\,; \nu)$ is used to denote the dependence of F on ν. This is referred to as a *trajectory-based regression* problem. The empirical risk is given by

$$\mathcal{L}_N(\mathbf{c}) := \frac{1}{nJL} \sum_{i\in[n],j\in[J],\ell\in[L]} |\mathbf{V}_{i,j}^\ell - F_N(\mathbf{r}_{:,i}^\ell(t_j); \mathbf{c})|^2, \tag{2.6}$$

which measures the error between the estimated (or observed) velocity and the RFM $F_N(\cdot\,; \mathbf{c})$ from (2.3) (which depends on the vector c). The sum in (2.6) is evaluated on a finite set of timestamps. To simplify the expression for F_N, let $\mathbf{A} \in \mathbb{R}^{n_{\text{tot}}\times N}$ be the random radial feature matrix whose elements

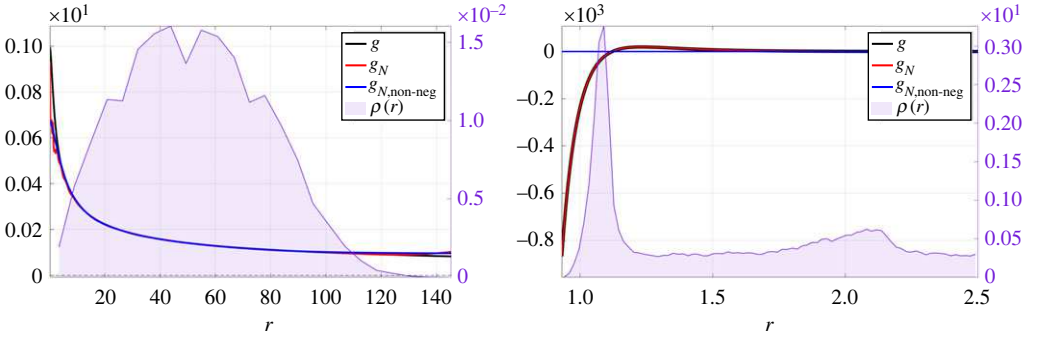

Figure 2. Comparing constraints: the plots compare the true interaction kernels (left $g(r) = (1 + r^2)^{-1/4}$ and right (4.2) the black curves) and the learned interaction kernels using the radial features with the non-negative constraints (the red curves) and with the non-negative constraints (the blue curves). The purple region plots the empirical distribution of the radial distances estimated on the trajectories. The first graph shows that (within the region of observed radial distances), the two kernels obtain similar accuracy when the true kernel satisfies the constraints. The second graph shows that if the true kernel does not satisfy the constraints, then the unconstrained approach is preferred. See §4 for parameters and details.

are defined by $a_{\tilde{i},k} = \phi_k(\mathbf{r}^{\ell}_{:,i}(t_j))$ (see (2.4)) where $\tilde{i}$ represents the triple (i, j, ℓ) after reindexing to a single index. Then (2.6) can be written as

$$\mathcal{L}_N(\mathbf{c}) = \frac{1}{njL}||\mathbf{V} - \mathbf{A}\mathbf{c}||_2^2, \tag{2.7}$$

using vector notation. We can obtain the trained coefficient vector by minimizing the empirical loss directly, i.e.

$$\mathbf{c} = \arg\min_{\mathbf{c}' \in \mathbb{R}^N} ||\mathbf{V} - \mathbf{A}\mathbf{c}'||_2^2, \tag{2.8}$$

i.e. the ordinary least-squares problem.

Note that, for a positive definite function, the coefficients $\mathbf{c}$ should be constrained to be non-negative based on theorem 2.1. However, by relaxing the assumptions, we find that the approximation is still valid in practice without the additional computational cost incurred by adding a constraint. In figure 2, we compare the approximation of two interaction kernels (the true kernels are the black curves) using the unconstrained optimization formulation (in red) and the non-negative constrained optimization formulation (in blue). The (empirical) density function of radial distances is represented by the purple region in all figures. The first plot in figure 2 shows that for a radial and positive definite kernel g, the non-negative constrained problem produces a smoother approximation that agrees better with the true kernel for small $r > 0$. However, the regions where g_N does not agree with g have low sampling density and are thus less likely events in the dynamics. The second plot in figure 2 provides an example where the true interaction kernel g does not satisfy the assumptions of theorem 2.1. In this case, the kernel $g(r) \to -\infty$ as $r \to 0^+$, thus for visualization we do not include the large (negative) values. The non-negative constrained solution is trivial since positive coefficients add more deviation for $0 < r < 1$ than the potential fit gained for the region $r > 1$. Based on this example, we argue that for positive definite and radial kernels the discrepancies between the two formulations are not statistically significant, while for non-positive definite kernels the non-negative constraint can lead to large errors. Thus without prior knowledge about the system, we use the unconstrained formulation.

The training problem (2.7) measures the 'stationary' error, since it compares the data and model using the observed velocity at each point in time. Another way to state this is that the features, i.e. the columns of $\mathbf{A}$, are applied to the data directly and do not depend on the learned solution generated by the trained system. To evaluate the effectiveness of the trained model, we

want to measure the error of the model (i.e. the equations of motion) and the error that is incurred over the path generated by the trained ODE system

$$\left.\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t}\tilde{\mathbf{x}}_i(t) &= F_N(\tilde{\mathbf{r}}_{1,i}(t), \ldots, \tilde{\mathbf{r}}_{n,i}(t); \mathbf{c}) \\ \text{and} \qquad \tilde{\mathbf{r}}_{i',i}(t) &= \tilde{\mathbf{x}}_{i'}(t) - \tilde{\mathbf{x}}_i(t). \end{aligned}\right\} \tag{2.9}$$

To define the generalization error of the model, consider the probability measure ρ on $\mathbb{R}_+$ that defines the distribution of pairwise distances between agents. This gives us the needed distribution for the values of r that define the input distribution to the true interaction kernel g and the trained interaction kernel defined by

$$g_N(\mathbf{r}) = \frac{2^n}{\pi^{(n-1)/2}\, N} \sum_{k=1}^{N} c_k\, \omega_k^{n+1}\, \mathrm{e}^{-||\mathbf{r}||_2^2\, \omega_k^2}. \tag{2.10}$$

Formally, for any interval $U \subseteq \mathbb{R}_+$, $\rho(U)$ is the probability that the pairwise distance between two agents is in U, for all IC sampled from $\boldsymbol{\mu}_0$ and for $t \in [0, T]$, i.e.

$$\rho(r) = \frac{1}{\binom{n}{2} T} \int_0^T \mathbb{E}_{\mathbf{x}(0)\sim\boldsymbol{\mu}_0} \left[\sum_{1\le i'<i\le n} \delta_{r_{i',i}(t)}(r) \right] \mathrm{d}t. \tag{2.11}$$

We define the $L^2(\rho)$ *kernel generalization error* as $||G' - G'_N||_{L^2(\rho)}$ where $G'(r) = g(r)r$ and $G'_N(r) = g_N(r)r$. In practice, the expectation and supremum cannot be computed directly, so instead one can consider the empirical probability density by replacing the integrals and expectations via averages. Lastly, the *path-wise generalization error* can be defined as

$$\mathbf{E}(c) = \mathbb{E}_{\mathbf{x}(0)\sim\boldsymbol{\mu}_0} \left[\sup_{t\in[0,\tilde{T}], i\in[N]} ||\mathbf{x}_i(t) - \tilde{\mathbf{x}}_i(t)||_2 \right], \tag{2.12}$$

where $\tilde{\mathbf{x}}$ is the simulated path using (2.9) and thus depends implicitly on c. When final forecast time $\tilde{T} = T$, $\mathbf{E}$ measures how close the simulated path is to the training data, noting that this is not the training error used in (2.7) since the training problem minimizes the stationary risk and not the discrepancy between the data and the simulated trajectory. For all experiments, we use $\tilde{T} > T$ to evaluate the performance of the model in the extrapolation regime, i.e. how well does the model predict future states using a test dataset. The error (2.12) is approximated numerically by averaging over a finite set of randomly sampled paths and maximized over a finite set of timestamps.

The errors depend on the ability of the training problem to approximate the interaction kernel g. For functions in the appropriate reproducing kernel Hilbert spaces, it was shown that under some technical assumptions, a random feature approximation obtained from minimizing the empirical risk with a ridge regression penalty (i.e. adding the term $\lambda||\mathbf{c}||_2^2$ to (2.7)) produces generalization errors that scale like $\mathcal{O}(m^{-1/2})$ when the number of random features scales like $N = \mathcal{O}(m^{1/2} \log m)$ where m is the total number of data samples [52]. For the ordinary least-squares case, [54,60] analysed the behaviour of the approximation and its associated risk as a function of the dimension of the data, the size of the training set, and the number of random features. In the overparameterized setting, i.e. when there are many more features than data samples, [54] concluded that the ordinary least-squares model may produce the optimal approximation among all kernel methods. In [60,61], the random feature matrix was shown to be well-conditioned in the sense that its singular values were bounded away from zero with high probability. In addition, the risk associated with underparameterized random feature methods scale like $\mathcal{O}(N^{-1} + m^{-1/2})$. Although the assumptions in [52,54,60] do not directly hold in our setting, their results do indicate the expected behaviour of using an RFM for learning interacting dynamics in the regime where the number of agents or the number of random initial conditions is sufficiently large (thus matching the random sampling needed in the theory for random feature methods). We leave a detailed theoretical examination of this problem for future work.

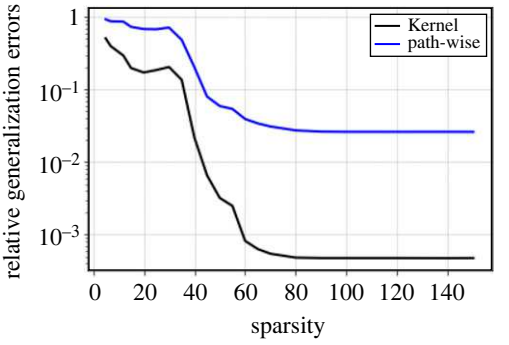

Figure 3. Motivation for sparsity: the plot shows the relative generalization errors for the kernel and learned trajectories as a function of the coefficient sparsity level s for the Lennard–Jones system. The errors plateau for $s \geq 80$ indicating that adding more features does not necessarily improve the error in this example.

(c) Sparse learning problem

The theory for RFMs suggests that N must be sufficiently large to obtain high-accuracy models. However, the cost of simulating the path using (2.9) requires at least one query of the RFM for each time step and thus scales linearly in N. Therefore, for computational efficiency one does not want N to be too large but for accuracy large N is needed. This motivates us to use a sparse random feature regression problem, in particular, we replace (2.8) with the ℓ^0 constrained least-squares problem

$$\mathbf{c} = \arg \min_{\mathbf{c}' \in \mathbb{R}^N, ||\mathbf{c}'||_0 \leq s} ||\mathbf{V} - \mathbf{A}\mathbf{c}'||_2^2, \tag{2.13}$$

where $||\mathbf{c}||_0$ is the number of non-zero entries of $\mathbf{c}$. To solve (2.13), we use a greedy approach called the hard thresholding pursuit (HTP) algorithm [62]

$$\left.\begin{aligned} S^{h+1} &= \{\text{indices of } s \text{ largest (in magnitude) entries of } \mathbf{c}^h + \mathbf{A}^*(\mathbf{V} - \mathbf{A}\mathbf{c}^h)\} \\ \text{and} \qquad \mathbf{c}^{h+1} &= \arg\min\{||\mathbf{V} - \mathbf{A}\mathbf{c}||_2^2, \quad \text{supp}(\mathbf{c}) \subseteq S^{h+1}\} \end{aligned}\right\} \tag{2.14}$$

which generates a sequence $\mathbf{c}^h$ index by $h \in [0, H]$. The two-step approach iteratively sparsifies and then re-fits the coefficients with respect to the random radial feature matrix $\mathbf{A}$ applied to the training dataset.

The motivation for using sparsity is that it enables us to find the most dominant modes from a large feature space. Theoretically, we expect that the sparse coefficients $\mathbf{c}$ concentrate near the largest values of $\nu(\omega)$ when taking into account the random sampling density $\theta(\omega)$. To verify the sparsity assumption, we measure the (relative) generalization errors produced by the random radial basis approximation obtained from the HTP algorithm (2.14) with various sparsity levels s in the range $[1, 150]$; see figure 3. Both relative errors generally decay as s increases and we see that they plateau for $s \geq 80$, thus no accuracy gains are made by increasing the number of terms used in the approximation. When s is close to 40, the errors are within an order of magnitude of the optimal error. Thus, it would be advantageous to choose a sparsity level s smaller than the dimension of the full feature space in order to lower the simulation cost of (2.9).

There have been several recent random feature methods that either use an ℓ^1 or an ℓ^0 penalty to obtain sparse representations. In [63], the sparse random feature expansion was proposed which used an ℓ^1 basis pursuit denoising formulation in order to obtain sparse coefficients with statistical guarantees, see also [60]. The results in [63] match the risk bounds of the ℓ^2 formulation, i.e. an error of $\mathcal{O}(N^{-1} + m^{-1/2})$, when the sparsity level is set to $s = N$ and the number of measurements $m = \mathcal{O}(N \log(N))$. When the coefficient vector is *compressible*, i.e. they can be well-represented by a sparse vector, then $s < N$ gives similar results with a lower model complexity

than in the ℓ^2 case and with fewer samples $m = \mathcal{O}(s\log(N))$. In [64], a step-wise LASSO approach was used to iteratively increase the number of random features by adding a sparse subset from multiple random samples. In [65], an iterative magnitude pruning approach was used to extract sparse representations for RFM. Our proposed model is related to the hard-ridge RFM [66] where it was shown that for certain feature functions ϕ_k with Gaussian data and weights, the iterative method converges to a solution with a given error bound, see also [67].

Although both the ℓ^2 and the sparse RFMs come with various theoretical properties, those results do not hold directly here. In the setting of learning multi-agent interaction kernels, the data samples are typically less ideal than in the theoretical setting, namely, trajectories are highly correlated and so they are not independent and identically distributed random variables. In fact, the randomness in the data is obtained from the sampling of various initial conditions. Additionally, the trajectories may contain noise and outliers which leads to non-standard biasing in the velocity estimates in the training problem. Empirically, we observe that the two approaches behave in a similar fashion, and that the sparsity-promoting approach often avoids overfitting on the data.

3. Numerical method

The datasets are created by approximating the solutions to the systems of ODE (see §4) in MATLAB using the ODE15s solver with relative tolerance 10^{-5} and absolute tolerance 10^{-6} to handle the stiffness of the system. The train and test datasets have the same size collection of L independent trajectories computed over the time interval $[0, T]$. The L initial conditions are independently sampled from the user-prescribed probability measure $\boldsymbol{\mu}_0$, i.e. $\mathbf{x}^\ell(0) \sim \boldsymbol{\mu}_0$ for $\ell \in [L]$ which is specified in each example. The observations are the state variables (i.e. we have measurements of $\mathbf{x}(t)$ and we do not have direct measurements of the velocity $(\mathrm{d}/\mathrm{d}t)\mathbf{x}(t)$ with multiplicative uniform noise of magnitude σ_{noise} at J equidistant timestamps in the interval $[0, T]$). An approximation of the unknown distribution $\rho(r)$ (defined in (2.11)) is obtained by using a large number L' ($L' \gg L$) of independent trajectories and averaging these trials to estimate $\rho(r)$ empirically. This is used for visualizing the distribution of the data and assessing the error of the learned kernel. The dictionary of ϕ_k's used in the training problem is defined in each section.

(a) Loss function for first-order homogeneous systems

Given n agents $\{\mathbf{x}_i(t)\}_{i=1}^n$, $\mathbf{x}_i(t) \in \mathbb{R}^d$ for all $i \in [n]$, the first-order homogeneous system is governed by the following system of ODEs:

$$\left.\begin{aligned} &\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{x}_i(t) = \frac{1}{n}\sum_{i'=1}^{n} g(||\mathbf{r}_{i',i}(t)||_2)\,\mathbf{r}_{i',i}(t) =: F(\mathbf{r}_{1,i}(t), \ldots, \mathbf{r}_{n,i}(t)) \\ \text{and} \quad &\mathbf{r}_{i',i}(t) = \mathbf{x}_{i'}(t) - \mathbf{x}_i(t), \end{aligned}\right\}$$

where g is the interaction kernel. Consider the radial feature space defined in (2.4). We seek to approximate F using a linear combination of elements from the above radial feature space

$$F_N(\mathbf{r}_1, \ldots, \mathbf{r}_n) = \frac{1}{N}\sum_{k=1}^{N} c_k\,\phi_k(\mathbf{r}_1, \ldots, \mathbf{r}_n).$$

Denote the (approximated) velocity dataset by $\mathbf{V} = [\mathbf{V}_{i,j}^\ell]_{\ell\in[L]} = [(\mathrm{d}/\mathrm{d}t)\mathbf{x}_i^\ell(t_j)] \in \mathbb{R}^{n_{\text{tot}}}$ and the radial feature matrix for space $\{\phi_k | k \in [N]\}$ by $\mathbf{A} = [a_{\tilde{i},k}] = [\phi_k(\mathbf{r}_{:,i}^\ell(t_j))] \in \mathbb{R}^{n_{\text{tot}}\times N}$ where $\tilde{i}$ represents the

triple (i, j, ℓ) after reindexing to a single index, the loss function is

$$\mathcal{L}_N(\mathbf{c}) = \frac{1}{nJL} \sum_{i\in[n], j\in[J], \ell\in[L]} |\mathbf{V}_{i,j}^{\ell} - F_N(\mathbf{r}_{:,i}^{\ell}(t_j); \mathbf{c})|^2$$
$$= \frac{1}{nJL} ||\mathbf{V} - \mathbf{A}\mathbf{c}||_2^2.$$

(b) Loss function for first-order heterogeneous systems

In a first-order heterogeneous system, each of the n agents $\{\mathbf{x}_i(t)\}_{i=1}^n$ has a type label $k_i \in \{1, 2\}$. We use n_1 and n_2 to denote the number of agents of type 1 and 2. The system is governed by the following system of ODEs:

$$\left.\begin{aligned} &\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{x}_i(t) = \sum_{i'=1}^{n} \frac{1}{n_{k_{i'}}} g_{k_i k_{i'}}(||\mathbf{r}_{i',i}(t)||_2)\, \mathbf{r}_{i',i}(t) =: F^{\text{heterog.}}(\mathbf{r}_{:,i}(t)) \\ \text{and} \quad &\mathbf{r}_{i',i}(t) = \mathbf{x}_{i'}(t) - \mathbf{x}_i(t), \end{aligned}\right\}$$

where $g_{k_i k_{i'}}$ is the interaction kernel governing how agents of type $k_{i'}$ influence agents of type k_i. Consider the following radial feature space:

$$\left\{ \phi_k^{\text{heterog.}} \,\middle|\, \phi_k^{\text{heterog.}}(\mathbf{r}_1, \ldots, \mathbf{r}_n) = \sum_{i=1}^{n} \frac{\beta(\omega_k)}{n_{k_i}} \mathrm{e}^{-||\mathbf{r}_i||_2^2 \omega_k^2} \mathbf{r}_i \right\}.$$

We seek to approximate $F^{\text{heterog.}}$ using a linear combination of elements from the above radial feature space

$$F_N^{\text{heterog.}}(\mathbf{r}_1, \ldots, \mathbf{r}_n) = \frac{1}{N} \sum_{k=1}^{N} c_k\, \phi_k^{\text{heterog.}}(\mathbf{r}_1, \ldots, \mathbf{r}_n),$$

where $\beta(\omega_k) = \sqrt{\pi}/2\omega_k$ is a normalization factor. Denote the (approximated) velocity dataset by $\mathbf{V} = [\mathbf{V}_{i,j}^{\ell}]_{\ell\in[L]} = [(\mathrm{d}/\mathrm{d}t)\mathbf{x}_i^{\ell}(t_j)] \in \mathbb{R}^{n_{\text{tot}}}$ and the radial feature matrix for space $\{\phi_k^{\text{heterog.}} | k \in [N]\}$ by $\mathbf{A} = [a_{\tilde{i},k}] = [\phi_k^{\text{heterog.}}(\mathbf{r}_{:,i}^{\ell}(t_j))] \in \mathbb{R}^{n_{\text{tot}}\times N}$, the loss function is

$$\mathcal{L}_N^{\text{heterog.}}(\mathbf{c}) = \frac{1}{nJL} \sum_{i\in[n], j\in[J], \ell\in[L]} |\mathbf{V}_{i,j}^{\ell} - F_N^{\text{heterog.}}(\mathbf{r}_{:,i}^{\ell}(t_j); \mathbf{c})|^2$$
$$= \frac{1}{nJL} ||\mathbf{V} - \mathbf{A}\mathbf{c}||_2^2.$$

(c) Loss function for second-order homogeneous systems

Given n agents $\{\mathbf{x}_i(t)\}_{i=1}^n, \mathbf{x}_i(t) \in \mathbb{R}^d$ for all $i \in [n]$, the second-order homogeneous system is governed by the following system of ODEs:

$$\left.\begin{aligned} &\frac{\mathrm{d}^2}{\mathrm{d}t^2}\mathbf{x}_i(t) = \frac{1}{n} \sum_{i'=1}^{n} g(||\mathbf{r}_{i',i}(t)||_2) \frac{\mathrm{d}}{\mathrm{d}t}\mathbf{r}_{i',i}(t) =: F^{\text{sec.}}(\mathbf{r}_{:,i}(t), \frac{\mathrm{d}}{\mathrm{d}t}\mathbf{r}_{:,i}(t)) \\ \text{and} \quad &\mathbf{r}_{i',i}(t) = \mathbf{x}_{i'}(t) - \mathbf{x}_i(t), \end{aligned}\right\}$$

where g is the interaction kernel. Consider the following radial feature space:

$$\left\{ \phi_k^{\text{sec.}} \,\middle|\, \phi_k^{\text{sec.}}(\mathbf{r}_1, \ldots, \mathbf{r}_n) = \frac{\beta(\omega_k)}{n} \sum_{i=1}^{n} \mathrm{e}^{-||\mathbf{r}_i||_2^2 \omega_k^2} \frac{\mathrm{d}}{\mathrm{d}t}\mathbf{r}_i \right\}.$$

We seek to approximate $F^{\text{sec.}}$ using a linear combination of elements from the above radial feature space

$$F_N^{\text{sec.}}\left(\mathbf{r}_{:,}\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{r}_{:,i}(t)\right)=\frac{1}{N}\sum_{k=1}^{N}c_k\,\phi_k^{\text{sec.}}\left(\mathbf{r}_{:,}\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{r}_{:,i}(t)\right).$$

Denote the (approximated) acceleration dataset by $\mathbf{V}=[\mathbf{V}_{i,j}^{\ell}]_{\ell\in[L]}=[(\mathrm{d}^2/\mathrm{d}t^2)\mathbf{x}_i^{\ell}(t_j)]\in\mathbb{R}^{n_{\text{tot}}}$ and the radial feature matrix for space $\{\phi_k^{\text{sec.}}|k\in[N]\}$ by $\mathbf{A}=[a_{\tilde{i},k}]=[\phi_k^{\text{sec.}}(\mathbf{r}_{:,i}^{\ell}(t_j),\frac{d}{dt}\mathbf{r}_{:,i}^{\ell}(t_j))]\in\mathbb{R}^{n_{\text{tot}}\times N}$, the loss function is

$$\begin{aligned}\mathcal{L}_N^{\text{sec.}}(\mathbf{c})&=\frac{1}{nJL}\sum_{i\in[n],j\in[J],\ell\in[L]}\left|\mathbf{V}_{i,j}^{\ell}-F_N^{\text{sec.}}\left(\mathbf{r}_{:,i}^{\ell}(t_j),\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{r}_{:,i}^{\ell}(t_j);\mathbf{c}\right)\right|^2\\&=\frac{1}{nJL}||\mathbf{V}-\mathbf{A}\mathbf{c}||_2^2.\end{aligned}$$

(d) Remarks

Given training set $\mathbf{X}^{\ell}=[\mathbf{x}^{\ell}(t_j)]_{j\in[J],i\in[n]}\in\mathbb{R}^{dnJ}$ for $\ell\in[L]$, we first obtain the velocity dataset $\mathbf{V}=[\mathbf{V}_{i,j}^{\ell}]_{\ell\in[L]}=[(\mathrm{d}/\mathrm{d}t)\mathbf{x}_i^{\ell}(t_j)]\in\mathbb{R}^{\text{tot}}$ using the central difference scheme

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{x}_i^{\ell}(t_j)\approx\frac{\mathbf{x}_i^{\ell}(t_{j+1})-\mathbf{x}_i^{\ell}(t_{j-1})}{t_{j+1}-t_{j-1}},$$

and we do not assume that the samples are evenly spaced in time. After having the state and (approximated) velocity information as training data, we randomly sample ω_k from $\mathcal{N}(0,\sigma^2)$ (where σ^2 is user-defined) to form the radial feature space for learning

$$\left\{\phi_k\,\middle|\,\phi_k(\mathbf{r}_1,\ldots,\mathbf{r}_n)=\frac{\alpha(\omega_k)}{n}\sum_{i=1}^{n}\mathrm{e}^{-||\mathbf{r}_i||_2^2\,\omega_k^2}\,\mathbf{r}_i\right\}.$$

We then assemble them into a random radial feature matrix $\mathbf{A}\in\mathbb{R}^{n_{\text{tot}}\times N}$ whose elements are defined by $a_{\tilde{i},k}=\phi_k(\mathbf{r}_{:,i}^{\ell}(t_j))$. The empirical loss $\mathcal{L}_N(\mathbf{c})=(1/nJL)||\mathbf{V}-\mathbf{A}\mathbf{c}||_2^2$ is minimized using either least squares or HTP with sparsity s to obtain coefficients $\mathbf{c}$, and we form the approximations $g_N\approx g$ and $F_N\approx F$ via

$$g_N(\mathbf{r})=\frac{2^n}{\pi^{\frac{n-1}{2}}N}\sum_{k=1}^{N}c_k\,\omega_k^{n+1}\,\mathrm{e}^{-||\mathbf{r}||_2^2\,\omega_k^2}$$

and

$$F_N(\mathbf{r}_1,\ldots,\mathbf{r}_n)=\frac{1}{N}\sum_{k=1}^{N}c_k\,\phi_k(\mathbf{r}_1,\ldots,\mathbf{r}_n).$$

These approximations allow us to compute kernel and path-wise generalization errors.

(e) Hyperparameters

In each of the examples, the following hyperparameters are specified in the associated table: the initial conditions' probability distributions $\boldsymbol{\mu}_0$, the number of training trajectories L, the number of trajectories used to approximate the empirical data distribution L', the number of timestamps (i.e. the observation times) J, the training time range T, the generalization time range $\tilde{T}$, the uniform multiplicative noise parameter σ_{noise}, the number of agents n, the number of features N, the probability distribution for the random features θ and the sparsity s. For the random features, the probability distribution is set to the normal distribution $\mathcal{N}(0,\sigma^2)$, and the tunable learning parameters are N, s and σ which were determined through a coarse grid search. All codes are available at https://zenodo.org/record/7992652.

4. Experimental results

For visualization purposes, we choose $d = 2$ for our examples. In all tests, we use the random radial expansion where the trained coefficients are obtained using the least-squares method (RRE) or using the sparse approximation (SRRE). The main indication of success is based on the path-wise error, which is reported for each experiment. Note that the data are acquired over a time interval T which is smaller than the total time $\tilde{T}$ used in the prediction. All of the experiments include uniform multiplicative noise on the state variables with scaling parameter σ_{noise}. In all tests, the algorithm does not know what the true interaction kernels are and thus must learn them from observations of the dynamics. We test the two approaches on known interacting kernels so that we can provide empirical error bounds.

The Lennard–Jones system is a first-order homogeneous interacting system that models intermolecular dynamics. The governing equations are given by the following system of ODEs [68–70]:

$$\left.\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t}\mathbf{x}_i(t) &= \frac{1}{n}\sum_{i'=1}^{n} g(||\mathbf{r}_{i',i}(t)||_2)\,\mathbf{r}_{i',i}(t) \\ \text{and} \qquad \mathbf{r}_{i',i}(t) &= \mathbf{x}_{i'}(t) - \mathbf{x}_i(t), \end{aligned}\right\} \tag{4.1}$$

where $g(r) = G'(r)/r$ and $G(r)$ is the Lennard–Jones potential given by

$$G(r) = 4\varepsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right], \tag{4.2}$$

where $\varepsilon = 10$ is the depth of the potential well (dispersion energy) and $\sigma = 1$ is the distance at which the potential between particles is 0. The r^{-12} term accounts for repulsion at short ranges and the r^{-6} term accounts for attraction at long ranges. The trajectories form a specific configuration of states, which can limit the available information when training on observations of the configuration (see figure 5 where each curve is the path of one agent in time). This is observed numerically in figure 4, where the empirical density of radial distances $\rho(r)$ (the purple region) concentrates around $r = 1$ with very few measurements outside of $r \in [(1/2), 5]$. Even though the data distribution does not provide a rich sampling of r, the RRE (blue curve on the left) and SRRE (blue curve on the right) provide close approximations within the entire interval of interest. Both the RRE and SRRE approximations produce similar relative kernel errors of 8.64×10^{-4} and 3.00×10^{-4}, respectively. Using the learned kernels from figure 4, in figure 5, the simulated paths are compared to the true paths starting from new initial data. The total prediction time is $50\times$ longer than the interval in which we observe the training trajectories. The path-wise errors using random ICs are $2.22 \times 10^{-2} \pm 6.48 \times 10^{-2}$ (RRE) and $8.20 \times 10^{-3} \pm 2.44 \times 10^{-2}$ (SRRE). The learned paths require at least one query of the learned kernels for each time step, thus the SRRE approach leads to significant improvements in run time cost for forecasting and predictions. For this experiment, the error is slightly lower in the SRRE case with a $6.67\times$ speed up in queries at each step. Table 1 contains the hyperparameters used for this experiment.

The Cucker–Smale system is a second-order homogeneous system. The dynamics are governed by the following systems of ODEs [71,72]:

$$\left.\begin{aligned} \frac{\mathrm{d}^2}{\mathrm{d}t^2}\mathbf{x}_i(t) &= \frac{1}{n}\sum_{i'=1}^{n} g(||\mathbf{r}_{i',i}(t)||_2)\,\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{r}_{i',i}(t) \\ \text{and} \qquad \mathbf{r}_{i',i}(t) &= \mathbf{x}_{i'}(t) - \mathbf{x}_i(t), \end{aligned}\right\} \tag{4.3}$$

where $g(r) = (1 + r^2)^{-(1/4)}$ is an alignment-based interaction kernel. Table 2 contains the hyperparameters used for the system. In this setting, the RRE obtains a smaller relative kernel error than the SRRE, that is 1.30×10^{-3} and 1.74×10^{-1}, respectively, since the SRRE kernel produces a small oscillation within the support set of the distribution of observed radial distance. This is likely due to an instability formed from the second-order model which can be resolved by increasing the sparsity. However, the learned and true trajectories shown in figure 6 indicate

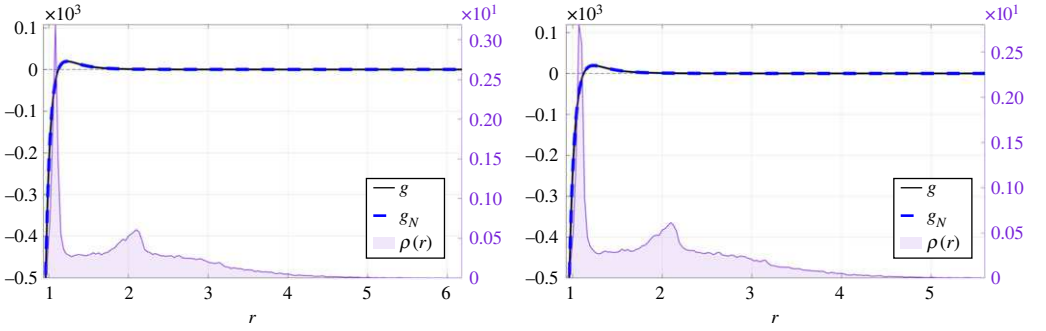

Figure 4. Lennard–Jones system: plots of the true interaction kernel (the black curves) and the learned interaction kernels (the blue curves). The first graph uses the RRE model and the second uses the SRRE model.

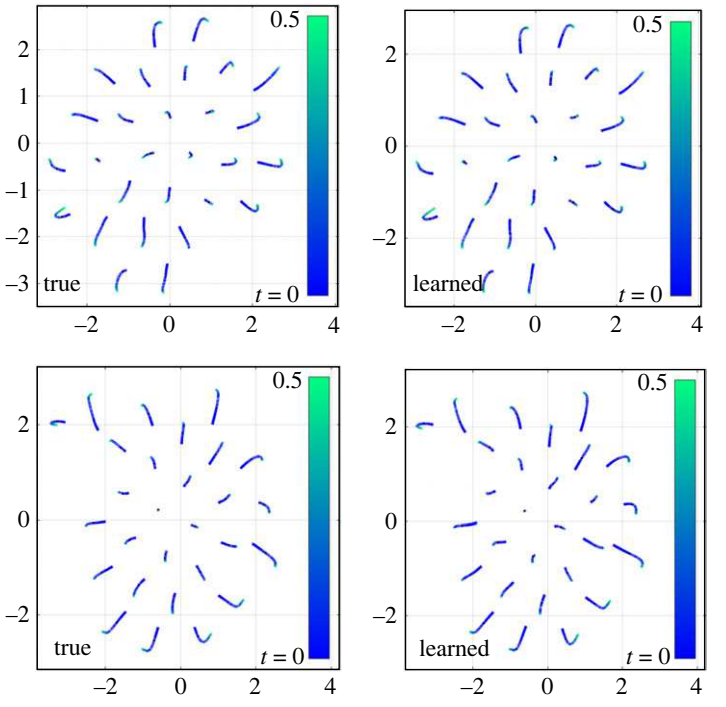

Figure 5. Lennard–Jones system: comparison of the true paths (first and third) and the learned paths (second is RRE and fourth is SRRE) on the test data using the trained interacting kernels. Each curve is the path of one agent in time (denoted by the colourbar).

Table 1. Hyperparameters used in the Lennard–Jones system example.

n	μ_0	L	L'	J	T	$\tilde{T}$	σ_{noise}
7	$\mathcal{N}(0, I_{2n})$	100	2000	150	0.01	0.5	0.001
N	θ	s					
1000	$\mathcal{N}(0, 35)$	150					

that the overall model agrees with the true governing system, see also figure 7. In fact, pathwise errors using random ICs are close to each other: $6.39 \times 10^{-1} \pm 5.40 \times 10^{-2}$ (RRE) and $6.36 \times 10^{-1} \pm 5.58 \times 10^{-2}$ (SRRE).

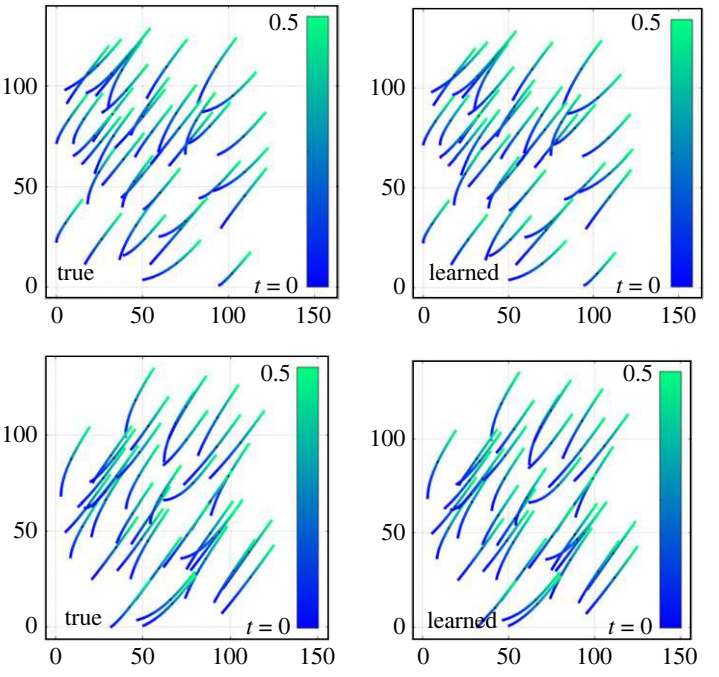

Figure 6. Cucker–Smale system: comparison of the true paths (first and third) and the learned paths (second is RRE and fourth is SRRE) on the test data using the trained interacting kernels. Each curve is the path of one agent in time (denoted by the colourbar).

Table 2. Hyperparameters used in the Cucker–Smale system example.

n	L	L'	J	T	$\tilde{T}$	σ_{noise}
10	50	2000	200	0.25	0.5	0.01
$\boldsymbol{\mu}_0 = \boldsymbol{\mu}_{0,\text{velocity}}$	N	θ	s			
unif. on $[0, 100]^2$	1000	$\mathcal{N}(0, 1)$	500			

Single predator-swarming prey interactions is an example of a first-order heterogeneous particle system with four interaction kernels. Each agent $\mathbf{x}_i$ has a type label $k_i \in \{1, 2\}$. We use n_1 and n_2 to denote the number of agents of type 1 (prey) and 2 (predator), and $g_{k_i k_{i'}}$ is the interaction kernel governing how agents of type $k_{i'}$ influence agents of type k_i. The dynamics are governed by the following system:

$$\left.\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t}\mathbf{x}_i(t) &= \sum_{i'=1}^{n} \frac{1}{n_{k_{i'}}} g_{k_i k_{i'}}(||\mathbf{r}_{i',i}(t)||_2)\, \mathbf{r}_{i',i}(t) \\ \text{and} \qquad \mathbf{r}_{i',i}(t) &= \mathbf{x}_{i'}(t) - \mathbf{x}_i(t), \end{aligned}\right\} \tag{4.4}$$

where the four interaction kernels (prey–prey, prey–predator, predator–prey, predator–predator) are given by

$$g_{11}(r) = 1 - r^{-2}, \quad g_{12}(r) = -2r^{-2}$$
$$g_{21}(r) = 3.5r^{-3}, \quad g_{22}(r) = 0,$$

and all terms will be learned. One of the potential difficulties with this example is that the path-wise error depends on learning each of the kernels to within the same accuracy. In addition,

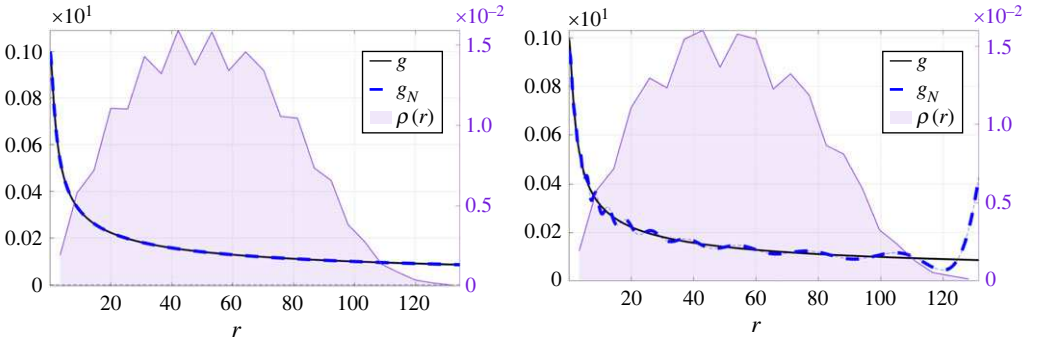

Figure 7. Cucker–Smale system: plots of the true interaction kernel (the black curves) and the learned interaction kernels (the blue curves). The first graph uses the RRE model and the second uses the SRRE model.

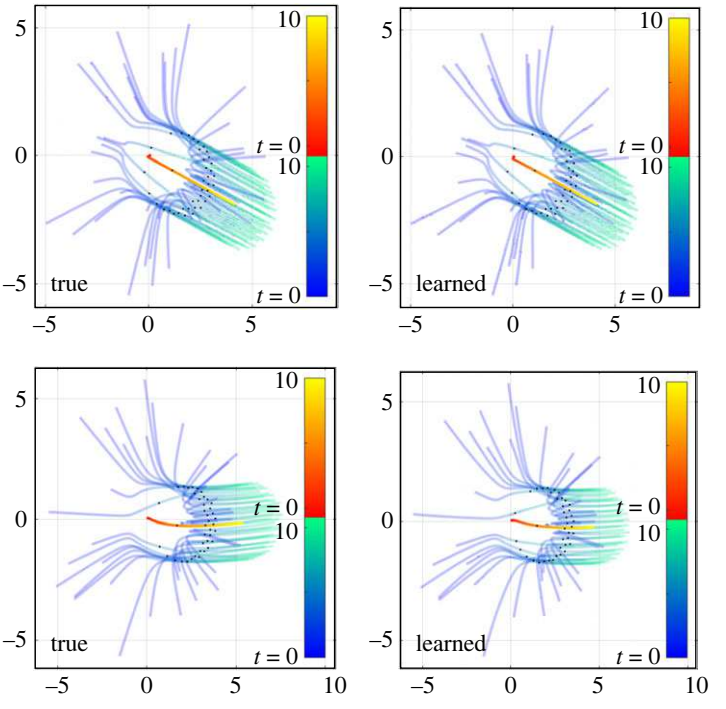

Figure 8. Single predator-swarming prey interactions: comparison of the true paths (first and third) and the learned paths (second is RRE and fourth is SRRE) on the test data using the trained interacting kernels. Each curve is the path of one agent in time (denoted by the colourbar). The warm tone is the predator and the cool tone is used for the prey. The dots represent the transition point between the length of time used in the training set and the additional time in the test.

the density of radial distances between the predator–prey (and by symmetry the prey–predator) concentrates at $r = 1.5$ with a rapid decay outside of the peak. This example tests the methods' ability to obtain accurate models with a limited sampling distribution.

Both the RRE and SRRE produce similar relative kernel errors for all kernels, with their pathwise errors using random ICs being comparable: $3.70 \times 10^{-2} \pm 2.49 \times 10^{-1}$ for RRE and $8.19 \times 10^{-3} \pm 4.63 \times 10^{-2}$ for SRRE. The true and learned trajectories are plotted in figure 8, where the dots represent the transition point between the length of time used in the training set and the additional time 'extrapolated' by the learned dynamical system. The data in figure 8 are a new set

Table 3. Hyperparameters used in the single predator-swarming prey system example.

n_1	n_2	L	L'	J	T	$\tilde{T}$	σ_{noise}
9	1	50	2000	200	5	10	0.001
$\boldsymbol{\mu}_0^1$	$\boldsymbol{\mu}_0^2$						
unif. on ring [0.5, 1.5]	unif. on disc at 0.1						
N	θ	s					
$\begin{bmatrix} 500 & 500 \\ 500 & 50 \end{bmatrix}$	$\mathcal{N}(0, 30)$	400					

of conditions that are not from the training set. Table 3 contains the hyperparameters used for the experiment in figure 9.

Sheep-food interacting system: Inspired by art which captures large flocks of sheep swarming various grain patterns [73], we propose a sheep-food interacting model and try to discover the interactions based on observations. Our approach to model the sheep is based off of the prey dynamics in [1]. Similarly, the sheep take on the role of predator in their interaction with the stationary food source. The sheep both want to cluster together for safety, but also are drawn to seek out the food at the cost of the group cohesion. We manually set the dynamics to be governed by a first-order heterogeneous system: (4.4) where

$$g_{11}(r) = -\frac{1.2}{r^4/4 + 0.05} + 0.02, \quad g_{12}(r) = 65\,\mathrm{e}^{-(r/0.45)},$$
$$g_{21}(r) = 0, \quad g_{22}(r) = 0,$$

where type 1 is the food and type 2 are the sheep. The food is the stationary 'prey' and are arranged in a 'heart' shape and are plotted as targets in figures 10 and 11. The sheep (the predator swarm) are initialized randomly below the food, near $y = -10$. The challenge is to correctly learn the dynamics from the earlier time interactions in order to correctly forecast the entire path the sheep take around the heart configuration. Note that although the training set only includes the interaction in the blue region, up to $T = 100$, the forecasted paths capture the full geometry. The errors for the kernels and paths are displayed in table 4. The RRE approximation produces smaller errors but the overall kernels are similar, see figure 11. The parameters are shown in table 5.

Comparison: Figure 1 provides a comparison of our approach and the standard RFF [49–51], showing that our randomized feature construction produces a more accurate approximation to the interacting kernels (table 5).

We compare our algorithm on the Lennard–Jones example with the approximation produced by the piecewise polynomial model from [42]. The set-up is as described in the previous Lennard–Jones example. The model from [42] using $L = 1000$ initial conditions for the training set produces a relative kernel error of 2.26×10^{-2} and a path-wise generalization error of 7.31×10^{-2}. The SRRE using $L = 100$ initial conditions for the training set produces a relative kernel error of 3.00×10^{-4} and a path-wise generalization error of 8.20×10^{-3}. This highlights two advantages of our approach. The RRE and SRRE methods need an order of magnitude fewer samples since they use a global smooth candidate set rather than a local basis. Secondly, the generalization error is lower due to the regularizing nature of the method, i.e. only retaining the dominate random features.

5. Discussion

We developed an RFM and learning algorithm for approximating interacting multi-agent systems. The method is constructed using an integral-based model of the interacting velocity field which is approximated by an RFM directly on observations of state variables. The resulting functions, i.e. the sums-of-Gaussians, led to a new radial randomized feature space

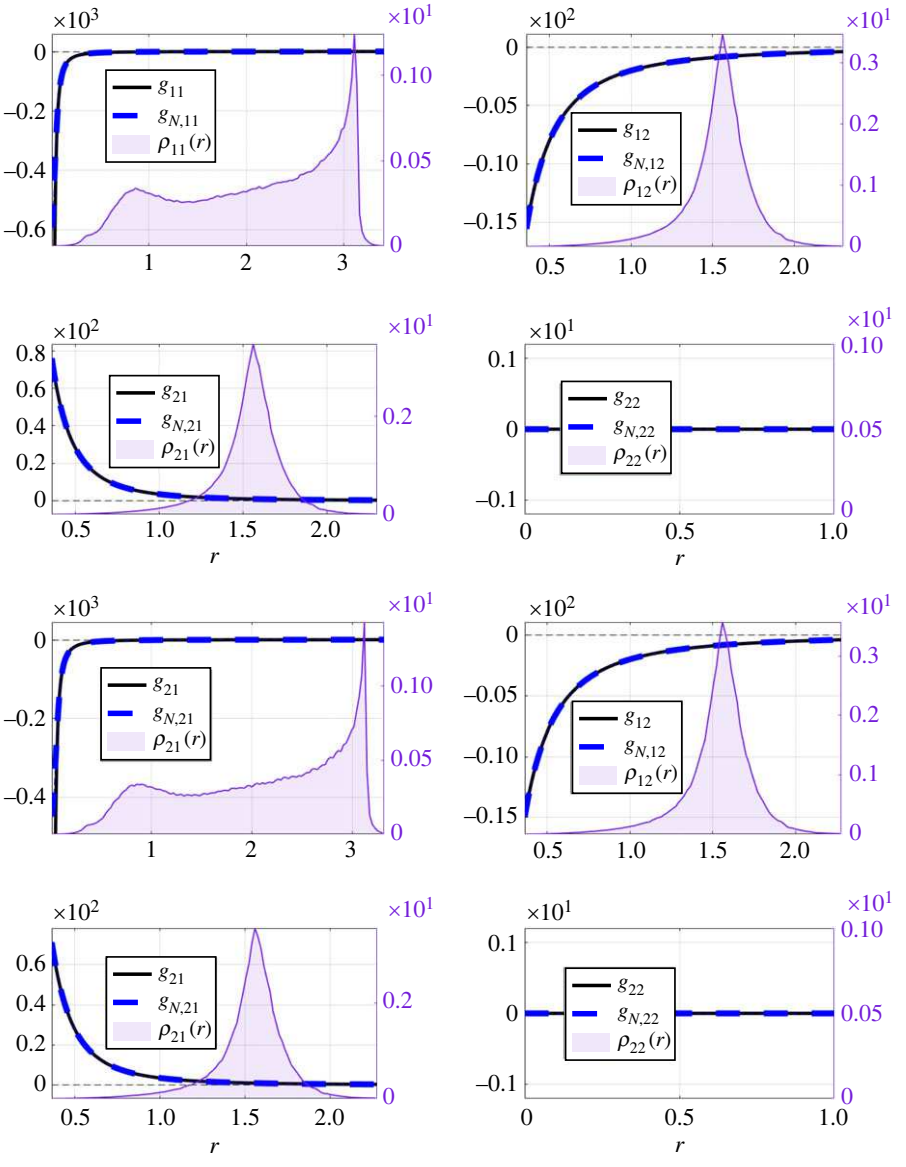

Figure 9. Single predator-swarming prey interactions: plots of the true interaction kernels (the black curves) and the learned interaction kernels (the blue curves). The first four graphs use the RRE model and the last four graphs use the SRRE model.

which is theoretically justified and was shown to produce a more consistent and stable result than other RFMs. We employ a randomized feature space rather than a pre-defined set of candidate functions, so that the algorithm would not require prior knowledge on the system or its interaction kernels. This is motivated by the applications, since manually modelling and training complex multi-agent systems is often computationally infeasible. When some terms or interactions are known by the user, they can be added to the model by incorporating them into the dynamical systems formulation. This can be seen by the different setups used for the loss function when considering homogeneous versus heterogeneous cases or first-order versus second-order systems.

We used a sparse random feature training algorithm based on the HTP algorithm. We note that when the sparsity parameter is set to the feature space size, i.e. $s = N$, the training reverts to

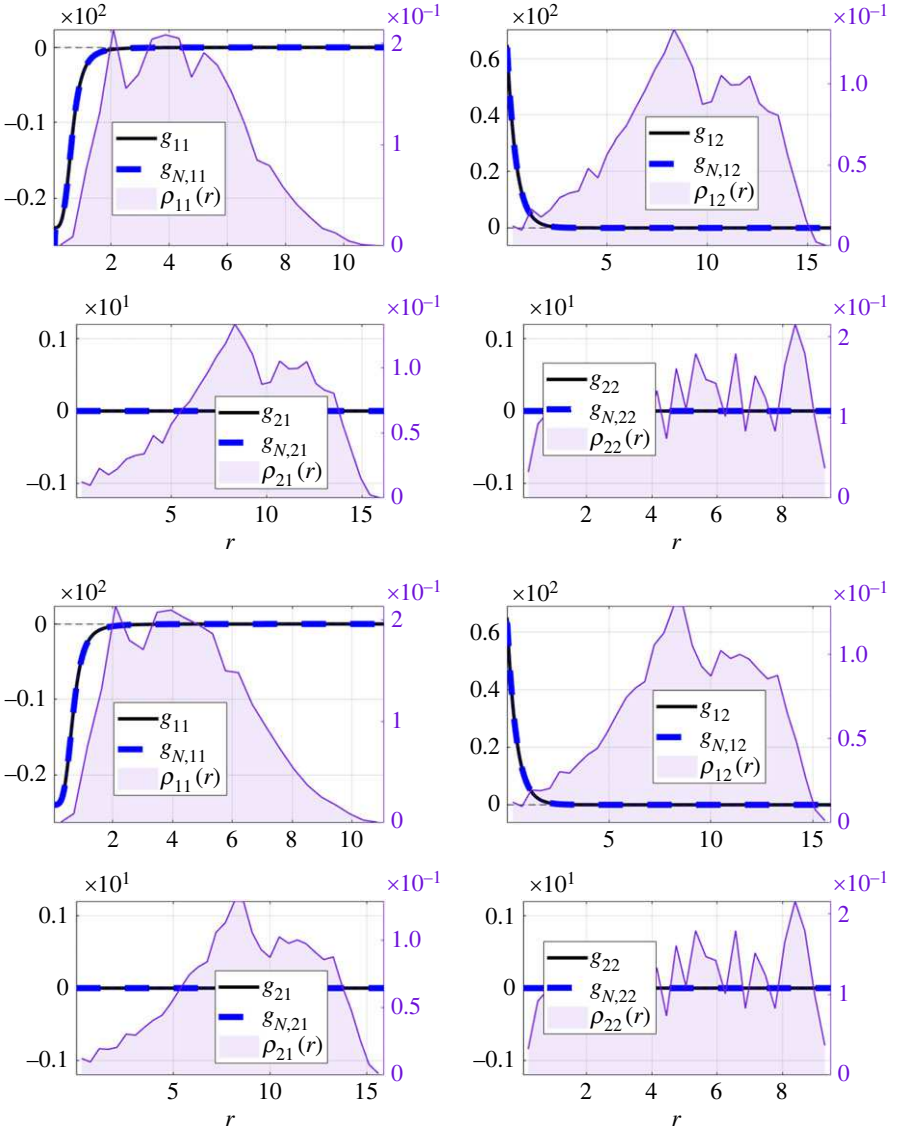

Figure 10. Sheep-food system: plots of the true interaction kernels (the black curves) and the learned interaction kernels (the blue curves). The first four graphs use the RRE model and the last four graphs use the SRRE model.

the least-squares problem and is applicable for some of the problems examined in this work. In particular, when one has a sufficiently rich dataset, least-squares-based RFMs produce accurate solutions. On the other hand, it is likely that there is only a limited set of observations of the system. In that case, there are several benefits to incorporating sparsity in the training of RFM, i.e. extracting an RFM that only uses a small number of randomized features. First, sparsity lowers the cost of forecasting for high-dimensional multi-agent system, since the cost scales with s rather than N. Since the trained systems are used to forecast the multi-agent dynamical system, a large number of repeated queries are needed, which would limit the use of standard RFM and overparameterized neural networks. Sparse regression also leads to less overfitting when one has a scarce or limited training set. Although SRRE may lead to models which are less interpretable than fixed candidate dictionaries such as [20,35,40], the SRRE approach does provide some insight

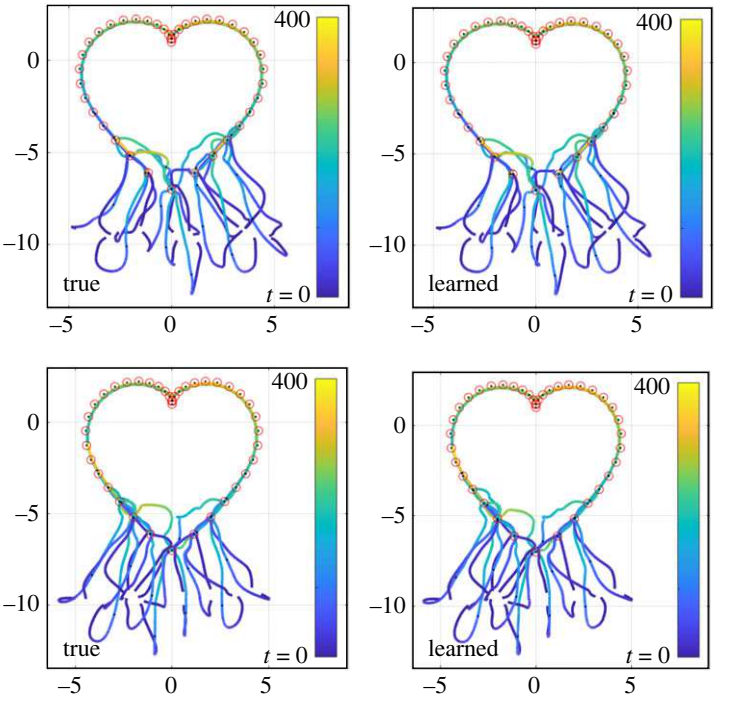

Figure 11. Sheep-food interacting system: comparison of the true paths (first and third) and the learned paths (second is RRE and fourth is SRRE) on the test data using the trained interacting kernels.

Table 4. Kernel and generalization errors of sheep-food interacting system example.

	RRE	SRRE
relative Kernel error	$\begin{bmatrix} 9.35 \times 10^{-3} & 5.93 \times 10^{-3} \\ 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 2.26 \times 10^{-2} & 1.43 \times 10^{-2} \\ 0 & 0 \end{bmatrix}$
path-wise error: training	$6.21 \times 10^{-1} \pm 6.89 \times 10^{-1}$	$1.69 \pm 9.23 \times 10^{-1}$
path-wise error: testing	$8.32 \times 10^{-1} \pm 9.53 \times 10^{-1}$	$1.72 \pm 9.49 \times 10^{-1}$

Table 5. Hyperparameters used in the sheep-food interacting system example.

n_1	n_2	L	L'	J	T	$\tilde{T}$	σ_{noise}
20	40	50	1000	600	100	400	0.001
μ_0^1	μ_0^2						
unif. on $[-9, -10] \times [-5, 5]$	heart of 'radius' 5 at 0						
N	θ	s					
$\begin{bmatrix} 500 & 500 \\ 50 & 50 \end{bmatrix}$	$\mathcal{N}(0, 10)$	600					

into the structure of the interaction kernel and can be used to guide analysis of the underlying system.

Our approach obtained accurate predictions for popular first-order system with homogeneous and heterogeneous interactions, second-order systems, and a new sheep-food interacting system from noisy measurements. A comparison with a similar approach showed that the SRRE

produced lower path-wise generalization error and lower kernel error while also needing an order of magnitude fewer training points. As with most approximation techniques for training dynamical systems, one limitation of the SRRE is with high levels of noise and outliers. One way to alleviate this is to preprocess the data, possibly using a neural network model, to approximate the underlying training paths before extracting its collective behaviour. However, the choice of preprocessing algorithms may bias the trained dynamics. Another limitation is its inability to train highly accurate predictive models when the underlying interacting system incorporates additional terms outside of the assumptions. A future direction may be to train only the interacting kernel component of the system and develop a closure approach for obtaining the remaining aspects of the dynamics.

Data accessibility. All codes are available from the Zenodo repository: https://zenodo.org/record/7992652 [74].

Authors' contributions. Y.L.: conceptualization, data curation, formal analysis, investigation, methodology, software, validation, visualization, writing—original draft, writing—review and editing; S.G.M.: conceptualization, data curation, formal analysis, funding acquisition, investigation, methodology, project administration, resources, software, supervision, validation, visualization, writing—original draft, writing—review and editing; H.S.: conceptualization, data curation, formal analysis, funding acquisition, investigation, methodology, project administration, resources, software, supervision, validation, visualization, writing—original draft, writing—review and editing.

All authors gave final approval for publication and agreed to be held accountable for the work performed therein.

Conflict of interest declaration. We declare we have no competing interests.

Funding. S.G.M. would like to acknowledge the support of NSF grant nos. 1813654, 2112085 and the Army Research Office (W911NF-19-1-0288). H.S. was supported in part by AFOSR MURI FA9550-21-1-0084, NSF DMS-2331100 and NSF DMS-1752116.

References

1. Chen Y, Kolokolnikov T. 2014 A minimal model of predator–swarm interactions. *J. R. Soc. Interface* **11**, 20131208. (doi:10.1098/rsif.2013.1208)
2. Motsch S, Tadmor E. 2011 A new model for self-organized dynamics and its flocking behavior. *J. Stat. Phys.* **144**, 923–947. (doi:10.1007/s10955-011-0285-9)
3. von Brecht JH, Uminsky DT. 2016 Anisotropic assembly and pattern formation. *Nonlinearity* **30**, 225. (doi:10.1088/1361-6544/30/1/225)
4. Mackey A, Kolokolnikov T, Bertozzi AL. 2014 Two-species particle aggregation and stability of co-dimension one solutions. *Discrete Continu. Dyn. Syst. B* **19**, 1411. (doi:10.3934/dcdsb.2014.19.1411)
5. Evers JH, Fetecau RC, Ryzhik L. 2015 Anisotropic interactions in a first-order aggregation model. *Nonlinearity* **28**, 2847. (doi:10.1088/0951-7715/28/8/2847)
6. Wang L, Short MB, Bertozzi AL. 2017 Efficient numerical methods for multiscale crowd dynamics with emotional contagion. *Math. Models Methods Appl. Sci.* **27**, 205–230. (doi:10.1142/S0218202517400073)
7. Kolokolnikov T, Sun H, Uminsky D, Bertozzi AL. 2011 Stability of ring patterns arising from two-dimensional particle interactions. *Phys. Rev. E* **84**, 015203. (doi:10.1103/PhysRevE.84.015203)
8. Bertozzi AL, Kolokolnikov T, Sun H, Uminsky D, Von Brecht J. 2015 Ring patterns and their bifurcations in a nonlocal model of biological swarms. *Commun. Math. Sci.* **13**, 955–985. (doi:10.4310/CMS.2015.v13.n4.a6)
9. Von Brecht JH, Uminsky D, Kolokolnikov T, Bertozzi AL. 2012 Predicting pattern formation in particle interactions. *Math. Models Methods Appl. Sci.* **22**, 1140002. (doi:10.1142/S0218202511400021)
10. Carrillo JA, Fornasier M, Rosado J, Toscani G. 2010 Asymptotic flocking dynamics for the kinetic Cucker–Smale model. *SIAM J. Math. Anal.* **42**, 218–236. (doi:10.1137/090757290)
11. Barbaro AB, Canizo JA, Carrillo JA, Degond P. 2016 Phase transitions in a kinetic flocking model of Cucker–Smale type. *Multiscale Model. Simul.* **14**, 1063–1088. (doi:10.1137/15M1043637)

12. Bertozzi A, Garnett J, Laurent T, Verdera J. 2016 The regularity of the boundary of a multidimensional aggregation patch. *SIAM J. Math. Anal.* **48**, 3789–3819. (doi:10.1137/15M1033125)
13. Morales J, Peszek J, Tadmor E. 2019 Flocking with short-range interactions. *J. Stat. Phys.* **176**, 382–397. (doi:10.1007/s10955-019-02304-5)
14. Tadmor E, Tan C. 2014 Critical thresholds in flocking hydrodynamics with non-local alignment. *Phil. Trans. R. Soc. A* **372**, 20130401. (doi:10.1098/rsta.2013.0401)
15. Ha SY, Tadmor E. 2008 From particle to kinetic and hydrodynamic descriptions of flocking. (http://arxiv.org/abs/0806.2182)
16. Balagué D, Carrillo JA, Laurent T, Raoul G. 2013 Dimensionality of local minimizers of the interaction energy. *Arch. Ration. Mech. Anal.* **209**, 1055–1088. (doi:10.1007/s00205-013-0644-6)
17. Bongard J, Lipson H. 2007 Automated reverse engineering of nonlinear dynamical systems. *Proc. Natl Acad. Sci. USA* **104**, 9943–9948. (doi:10.1073/pnas.0609476104)
18. Schmidt M, Lipson H. 2009 Distilling free-form natural laws from experimental data. *Science* **324**, 81–85. (doi:10.1126/science.1165893)
19. Ghattas O, Willcox K. 2021 Learning physics-based models from data: perspectives from inverse problems and model reduction. *Acta Numer.* **30**, 445–554. (doi:10.1017/S0962492921000064)
20. Brunton SL, Proctor JL, Kutz JN. 2016 Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proc. Natl Acad. Sci. USA* **113**, 3932–3937. (doi:10.1073/pnas.1517384113)
21. Champion K, Lusch B, Kutz JN, Brunton SL. 2019 Data-driven discovery of coordinates and governing equations. *Proc. Natl Acad. Sci. USA* **116**, 22 445–22 451. (doi:10.1073/pnas.1906995116)
22. Zhang L, Schaeffer H. 2019 On the convergence of the SINDy algorithm. *Multiscale Model. Simul.* **17**, 948–972. (doi:10.1137/18M1189828)
23. Rudy S, Alla A, Brunton SL, Kutz JN. 2019 Data-driven identification of parametric partial differential equations. *SIAM J. Appl. Dyn. Syst.* **18**, 643–660. (doi:10.1137/18M1191944)
24. Kaiser E, Kutz JN, Brunton SL. 2018 Sparse identification of nonlinear dynamics for model predictive control in the low-data limit. *Proc. R. Soc. A* **474**, 20180335. (doi:10.1098/rspa.2018.0335)
25. Hoffmann M, Fröhner C, Noé F. 2019 Reactive SINDy: discovering governing reactions from concentration data. *J. Chem. Phys.* **150**, 025101. (doi:10.1063/1.5066099)
26. Shea DE, Brunton SL, Kutz JN. 2021 SINDy-BVP: sparse identification of nonlinear dynamics for boundary value problems. *Phys. Rev. Res.* **3**, 023255. (doi:10.1103/PhysRevResearch.3.023255)
27. Messenger DA, Bortz DM. 2021 Weak SINDy for partial differential equations. *J. Comput. Phys.* **443**, 110525. (doi:10.1016/j.jcp.2021.110525)
28. Messenger DA, Dall'Anese E, Bortz DM. 2022 Online weak-form sparse identification of partial differential equations. (http://arxiv.org/abs/2203.03979)
29. Narasingam A, Kwon JSI. 2018 Data-driven identification of interpretable reduced-order models using sparse regression. *Comput. Chem. Eng.* **119**, 101–111. (doi:10.1016/j.compchemeng.2018.08.010)
30. Abdullah F, Wu Z, Christofides PD. 2021 Sparse-identification-based model predictive control of nonlinear two-time-scale processes. *Comput. Chem. Eng.* **153**, 107411. (doi:10.1016/j.compchemeng.2021.107411)
31. Bhadriraju B, Narasingam A, Kwon JSI. 2019 Machine learning-based adaptive model identification of systems: Application to a chemical process. *Chem. Eng. Res. Des.* **152**, 372–383. (doi:10.1016/j.cherd.2019.09.009)
32. Abdullah F, Alhajeri MS, Christofides PD. 2022 Modeling and control of nonlinear processes using sparse identification: using dropout to handle noisy data. *Ind. Eng. Chem. Res.* **61**, 17 976–17 992. (doi:10.1021/acs.iecr.2c02639)
33. Bhadriraju B, Kwon JSI, Khan F. 2021 Risk-based fault prediction of chemical processes using operable adaptive sparse identification of systems (OASIS). *Comput. Chem. Eng.* **152**, 107378. (doi:10.1016/j.compchemeng.2021.107378)
34. Bhadriraju B, Bangi MSF, Narasingam A, Kwon JSI. 2020 Operable adaptive sparse identification of systems: application to chemical processes. *AIChE J.* **66**, e16980. (doi:10.1002/aic.16980)

35. Schaeffer H. 2017 Learning partial differential equations via data discovery and sparse optimization. *Proc. R. Soc. A* **473**, 20160446. (doi:10.1098/rspa.2016.0446)
36. Schaeffer H, Tran G, Ward R, Zhang L. 2020 Extracting structured dynamical systems using sparse optimization with very few samples. *Multiscale Model. Simul.* **18**, 1435–1461. (doi:10.1137/18M1194730)
37. Schaeffer H, Tran G, Ward R. 2018 Extracting sparse high-dimensional dynamics from limited data. *SIAM J. Appl. Math.* **78**, 3279–3295. (doi:10.1137/18M116798X)
38. Schaeffer H, Tran G, Ward R. 2017 Learning dynamical systems and bifurcation via group sparsity. (http://arxiv.org/abs/1709.01558)
39. Schaeffer H, McCalla SG. 2017 Sparse model selection via integral terms. *Phys. Rev. E* **96**, 023302. (doi:10.1103/PhysRevE.96.023302)
40. Peherstorfer B, Willcox K. 2016 Data-driven operator inference for nonintrusive projection-based model reduction. *Comput. Methods Appl. Mech. Eng.* **306**, 196–215. (doi:10.1016/j.cma.2016.03.025)
41. Mangan NM, Askham T, Brunton SL, Kutz JN, Proctor JL. 2019 Model selection for hybrid dynamical systems via sparse regression. *Proc. R. Soc. A* **475**, 20180534. (doi:10.1098/rspa.2018.0534)
42. Lu F, Zhong M, Tang S, Maggioni M. 2019 Nonparametric inference of interaction laws in systems of agents from trajectory data. *Proc. Natl Acad. Sci. USA* **116**, 14 424–14 433. (doi:10.1073/pnas.1822012116)
43. Tran G, Ward R. 2017 Exact recovery of chaotic systems from highly corrupted data. *Multiscale Model. Simul.* **15**, 1108–1129. (doi:10.1137/16M1086637)
44. Lu F, Maggioni M, Tang S. 2021 Learning interaction kernels in heterogeneous systems of agents from multiple trajectories. *J. Mach. Learn. Res.* **22**, 1013–1067. (doi:10.1007/s10208-021-09521-z)
45. Lu F, Maggioni M, Tang S. 2021 Learning interaction kernels in stochastic systems of interacting particles from multiple trajectories. *Found. Comput. Math.* **22**, 1–55. (doi:10.1007/s10208-021-09521-z)
46. Miller J, Tang S, Zhong M, Maggioni M. 2020 Learning theory for inferring interaction kernels in second-order interacting agent systems. (http://arxiv.org/abs/2010.03729)
47. Zhong M, Miller J, Maggioni M. 2020 Data-driven discovery of emergent behaviors in collective dynamics. *Phys. D* **411**, 132542. (doi:10.1016/j.physd.2020.132542)
48. Li Z, Lu F, Maggioni M, Tang S, Zhang C. 2021 On the identifiability of interaction functions in systems of interacting particles. *Stoch. Process. Appl.* **132**, 135–163. (doi:10.1016/j.spa.2020.10.005)
49. Rahimi A, Recht B. 2007 Random features for large-scale kernel machines. In *Advances in neural information processing systems*, vol. 20, *Vancouver, CA, 3–6 December 2007*. Red Hook, NJ: Curran Associates.
50. Rahimi A, Recht B. 2008 Uniform approximation of functions with random bases. In *2008 46th Annual Allerton Conf. on Communication, Control, and Computing, Monticello, IL, 23–26 September 2008*, pp. 555–561. New York, NY: IEEE.
51. Rahimi A, Recht B. 2008 Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning. In *Advances in neural information processing systems*, vol. 21, *Vancouver, Canada, 8–11 December 2008*. Red Hook, NJ: Curran Associates.
52. Rudi A, Rosasco L. 2017 Generalization properties of learning with random features. In *Advances in neural information processing systems*, vol. 30, *Long Beach, CA, 4–9 December 2017*. Red Hook, NJ: Curran Associates.
53. Li Z, Ton JF, Oglic D, Sejdinovic D. 2019 Towards a unified analysis of random Fourier features. In *Int. Conf. on Machine Learning, Long Beach, CA, 9–15 June 2019*, pp. 3905–3914. PMLR.
54. Mei S, Misiakiewicz T, Montanari A. 2021 Generalization error of random feature and kernel methods: hypercontractivity and kernel matrix concentration. *Appl. Comput. Harmon. Anal.* **59**, 3–84. (doi:10.1016/j.acha.2021.12.003)
55. Bach F. 2017 On the equivalence between kernel quadrature rules and random feature expansions. *J. Mach. Learn. Res.* **18**, 714–751.
56. Liu F, Huang X, Chen Y, Suykens JA. 2020 Random features for kernel approximation: a survey on algorithms, theory, and beyond. (http://arxiv.org/abs/2004.11154)

57. Somoza A, Chacón E, Mederos L, Tarazona P. 1995 A model for membranes, vesicles and micelles in amphiphilic systems. *J. Phys.: Condens. Matter* **7**, 5753. (doi:10.1088/0953-8984/7/29/005)
58. Schoenberg IJ. 1938 Metric spaces and completely monotone functions. *Ann. Math.* **39**, 811–841. (doi:10.2307/1968466)
59. Pennington J, Yu FXX, Kumar S. 2015 Spherical random features for polynomial kernels. In *Advances in neural information processing systems*, vol. 28, *Montreal, Canada, 7–12 December 2015*. Red Hook, NJ: Curran Associates.
60. Chen Z, Schaeffer H. 2021 Conditioning of random feature matrices: double descent and generalization error. (http://arxiv.org/abs/2110.11477)
61. Chen Z, Schaeffer H, Ward R. 2022 Concentration of random feature matrices in high-dimensions. (http://arxiv.org/abs/2204.06935)
62. Foucart S. 2011 Hard thresholding pursuit: an algorithm for compressive sensing. *SIAM J. Numer. Anal.* **49**, 2543–2563. (doi:10.1137/100806278)
63. Hashemi A, Schaeffer H, Shi R, Topcu U, Tran G, Ward R. 2023 Generalization bounds for sparse random feature expansions. *Appl. Comput. Harmon. Anal.* **62**, 310–330. (doi:10.1016/j.acha.2022.08.003)
64. Yen IEH, Lin TW, Lin SD, Ravikumar PK, Dhillon IS. 2014 Sparse random feature algorithm as coordinate descent in hilbert space. In *Advances in neural information processing systems*, vol. 27, *Montreal, Canada, 8–13 December 2014*. Red Hook, NJ: Curran Associates.
65. Xie Y, Shi B, Schaeffer H, Ward R. 2021 SHRIMP: sparser random feature models via iterative magnitude pruning. (http://arxiv.org/abs/2112.04002)
66. Saha E, Schaeffer H, Tran G. 2022 HARFE: hard-ridge random feature expansion. (http://arxiv.org/abs/2202.02877)
67. Richardson N, Schaeffer H, Tran G. 2022 SRMD: sparse random mode decomposition. (http://arxiv.org/abs/2204.06108)
68. Jones JE. 1924 On the determination of molecular fields.–I. From the variation of the viscosity of a gas with temperature. *Proc. R. Soc. Lond. A* **106**, 441–462. (doi:10.1098/rspa.1924.0081)
69. Jones JE. 1924b On the determination of molecular fields.–II. From the equation of state of a gas. *Proc. R. Soc. Lond. A* **106**, 463–477. (doi:10.1098/rspa.1924.0082)
70. Lennard-Jones JE. 1931 Cohesion. *Proc. Phys. Soc. (1926-1948)* **43**, 461. (doi:10.1088/0959-5309/43/5/301)
71. Cucker F, Smale S. 2007 Emergent behavior in flocks. *IEEE Trans. Autom. Control* **52**, 852–862. (doi:10.1109/TAC.2007.895842)
72. Cucker F, Smale S. 2007 On the mathematics of emergence. *Jpn J. Math.* **2**, 197–227. (doi:10.1007/s11537-007-0647-x)
73. Harkins G. A farmer couldn't go to his aunt's funeral because of covid. So he used sheep to send a heart-shaped message. *The Washington Post*.
74. Liu Y, McCalla SG, Schaeffer H. 2023 Random feature models for learning interacting dynamical systems. *Zenodo*. (https://zenodo.org/record/7992652)