A Numerical Study on Sparse Learning of Interaction Laws in Homogeneous Multiparticle Systems

Hao-Tien Chuang[†] , Dongyang Li[‡] , Shelby Malowney , and Ritwik Trehan.[‡]

Project advisor: Sui Tang [§‡]

Abstract. Multi-agent systems have found wide applications in science and engineering ranging from opinion dynamics to predator-prey systems. A grand challenge encountered in these areas is to reveal the interaction laws between individual agents leading to collective behaviors. In this article, we consider a system of ODEs that is often used in modeling opinion dynamics, where the laws of the interaction are dependent on pairwise distances. We leverage recent advancements in sparsity-promoted algorithms and propose a new approach to learning the interaction laws from a small amount of data. Numerical experiments demonstrate the effectiveness and robustness of the proposed approach in a small, noisy data regime and show the superiority of the proposed approach.

1. Introduction.

Multi-agent systems are ubiquitous in science and engineering. The individual interactions among agents produce a rich variety of compelling patterns, such as crystallization of particles, clustering of peoples' opinions, and coordinated movements of ants, fish, birds, or cars. However, the interaction laws of agents often remain elusive. It has been a long-standing problem in various disciplines to reveal the links between the collective behavior and the individual interaction laws [3, 15, 19, 9].

A common belief in scientific discovery is that complicated collective behaviors are consequences of simple interaction rules. In the mathematical modeling community, there have been tremendous research efforts to model collective dynamics using interaction laws based on pairwise distances. Researchers derive the governing equations for multi-agent systems by combining the fundamental physical law with relatively simple parametric families of elementary interaction functions. The goal is to find certain conditions on the interaction functions such that the underlying dynamical systems are well-posed and the asymptotic behavior of the solutions reproduce similar qualitative macroscopic patterns as observed in reality, such as flocking [7], crystallization [12, 20], clustering[10, 4, 17], and milling [6, 1, 2].

Recent rapid advancements in data information technology (especially in digital imaging and high-resolution lightweight GPS devices and particle tracking methods) allow gathering high-resolution trajectories of individual agents. This inspires the urgent need for devising efficient data-driven methods to uncover the governing interaction laws. The objective is to turn data into interaction functions that are not just predictive, but also provide physical insight into the nature of the underlying system from which the data was generated. Sparse regression techniques have drawn a lot of recent attention since many complex systems have simple algebraic representations corresponding to a sparse representation in high dimensional nonlinear functional spaces. By applying sparse regression to dynamical systems, we may de-

[†] Department of Mathematics, University of California, Los Angeles, CA 90095, USA (tchuang@ucla.edu).

[‡] Department of Mathematics, Applied probability and Statistics, Physics, University of California, Santa Barbara, CA 93106, USA (dongyang_li@ucsb.edu, smalowney@ucsb.edu,ritwik@ucsb.edu).

[§] Department of Mathematics, University of California, Santa Barbara, CA 93106, USA (suitang@ucsb.edu).

termine the fewest terms in the dynamic governing equations required to accurately represent the data. This results in parsimonious models that balance accuracy with model complexity to avoid over-fitting.

In this work, we leverage the recent advancement in the mathematical modeling of multi-agent systems and sparse regression techniques for identifying nonlinear dynamical systems called "SINDy" (Sparse Identification of Nonlinear Dynamics) [5]. We consider the data-driven discovery of interaction laws in a family of homogeneous agent systems that are derived from fundamental physical laws with an unknown interaction function modeling the pairwise-distance-based interactions. We propose sparsity-promoting techniques to approximate or discover the interaction functions from a large library, i.e., a set of candidate functions.

1.1. Relevant Work.

This work is built on the recent progress on the data-driven discovery of interacting particle system [14, 21, 13], where a least square regression approximation is applied to learn the interaction kernel ϕ from trajectory data. One can also refer to [11, 16] for recent advancements on the mean-field equation. In particular, [16] proposed a weak SINDy approach to identify the interaction kernel in the mean-field equation from the particle data.

1.2. Our Contribution.

Using a similar spirit in [5], we formulate the learning problem as a sparse regression problem. We use both the Sequential Least Square (SLS) and the Least Absolute Shrinkage and Selection Operator (LASSO) approach, as done in [5], to solve the corresponding sparse regression problem. We conduct an extensive numerical study. The numerical results support the effectiveness and robustness of the proposed approaches to opinion dynamics. In particular, we perform the comparison with the Least Squares estimator proposed in the previous work [14]. The numerical results show the superiority of leveraging the sparsity. We remark that our learning approach is different from [5]. In [5], one looks for the sparse coefficient for each row of the governing equations in a common dictionary. Here, our governing equations are coupled and connected by the interaction law, or kernel, ϕ. We look for the sparse coefficient vector for ϕ. Then together with the structure of the governing equations, one learns the form of the governing equations.

2. Problem Statements.

2.1. Formulation of Sparse Regression Problem.

We start with a first-order homogeneous agent system consisting of N agents in $\mathbb{R}^d$, interacting according to

$$\dot{\mathbf{x}}_i(t) = \frac{1}{N}\sum_{i'=1}^{N} \phi(\|\mathbf{x}_i(t) - \mathbf{x}_{i'}(t)\|)(\mathbf{x}_i(t) - \mathbf{x}_{i'}(t)), \tag{2.1}$$

where $i = 1, 2, ..., N$; $\mathbf{x}_i(t) \in \mathbb{R}^d$ is the state of the i^{th} particle; $\|\mathbf{x}_i(t) - \mathbf{x}_{i'}(t)\|$ is the Euclidean norm, and ϕ is the interaction law.

The form of the governing equation is derived using the Newton's second law where the mass of agents is assumed to be zero. The evolution of agents is governed by minimizing the potential energy function, depending on pairwise distance,

$$U(\mathbf{x}_1, \cdots, \mathbf{x}_N) = \frac{1}{2N}\sum_{i,i'=1,1}^{N,N} \Phi(\|\mathbf{x}_i - \mathbf{x}_{i'}\|)$$

so that the interaction force is the derivative of the potential function U and $\phi(r) = \frac{\Phi'(r)}{r}$. This kind of system has found applications in modeling opinion dynamics in social science [18, 8].

In modeling problems, a key challenge lies in the choice of ϕ to induce the desired collective behaviors, since we have very limited knowledge of the underlying complex system and there is no canonical choice of potential functions for many complex agents systems. One important observation is that many ad-hoc potential functions that have found successful applications have extremely simple parametric forms, such as the Leonard Jones potential in the form of $\Phi(r) = \frac{A}{r^p} - \frac{B}{r^q}$ (p, q are some positive integers) and the power-law potential in the form of $\Phi(r) = Cr^{-p}$, where A, B, C are constants. In our numerical section, potentials involving both polynomials and sinusoidal terms, like (3.2), are considered. They typically take the form of a linear combination of only a few items, as opposed to an infinite series, making it sparse in the space of possible functions. This inspired the idea of using the sparse regression techniques to look for a sparse representation of ϕ in a large library of candidate functions.

2.1.1. Observational data regime. We consider the observational data consisting of positional and velocity data, $\{\mathbf{x}_i^{(m)}(t_l), \dot{\mathbf{x}}_i^{(m)}(t_l)\}_{i=1,m=1,l=1}^{N,M,L}$. There are four parameters controlling the size of data: L denotes the instances of time $0 = t_1 < t_2 \cdots < t_L = T$; M denotes the number of trajectories, and N, d denote the number of agents in $\mathbb{R}^d$. The initial positions for each trajectory are drawn i.i.d from a probability measure $\mu_0 = \text{Unif}([0, 10]^{Nd})$ defined on the state space $\mathbb{R}^{dN}$,i.e., the uniform distribution over a cube with the side length 10.

For demonstration purpose, we begin with the case of $M = 1$ and show how this learning problem can be formulated as a sparse regression problem. In this case, we collect the time history of one trajectory, $\mathbf{X}(t) = \left[\mathbf{x}_1(t), \cdots, \mathbf{x}_N(t)\right] \in \mathbb{R}^{dN}$ and either measure its derivative $\dot{\mathbf{X}}(t) \in \mathbb{R}^{dN}$ or numerically approximate it. The trajectory data for each agent is sampled at different time instances $t_1, t_2, \cdots, t_L$, and organized into state vector $\mathbf{X}$ and velocity vector $\dot{\mathbf{X}}$, which contain all agents evolved over time :

$$\mathbf{X} = \begin{bmatrix} \mathbf{X}(t_1) \\ \vdots \\ \mathbf{X}(t_L) \end{bmatrix} \in \mathbb{R}^{dNL} \quad \text{and} \quad \dot{\mathbf{X}} = \begin{bmatrix} \dot{\mathbf{X}}(t_1) \\ \vdots \\ \dot{\mathbf{X}}(t_L) \end{bmatrix} \in \mathbb{R}^{dNL} \tag{2.2}$$

Next, we construct a library Θ consisting of candidate nonlinear functions , $\{\varphi_1(r), \cdots, \varphi_n(r)\}$, where r belongs to the domain of pairwise distance among agents. For example, say $\Theta = \{1, r, r^2, \cdots, r^{10}, \sin r, \cos r, \cdots, \sin(10r), \cos(10r)\}$. For any 1D function φ defined in the positive axis of the real line, we introduce a map $\mathbf{f}$ so that $\mathbf{f}_\varphi$ defines an interaction force field of the same form with the right hand side of (2.1) where we replace ϕ with φ. In other words, for $\mathbf{X}(t) \in \mathbb{R}^{dN}$, the $i-$th component of $\mathbf{f}_\varphi(\mathbf{X}(t)) \in \mathbb{R}^{dN}$ is written as

$$[\mathbf{f}_\varphi(\mathbf{X}(t))]_i = \frac{1}{N} \sum_{i'=1}^{N} \varphi(\|\mathbf{x}_i(t) - \mathbf{x}_{i'}(t)\|)(\mathbf{x}_i(t) - \mathbf{x}_{i'}(t)) \in \mathbb{R}^d,$$

which is used to construct

$$\mathbf{f}_\varphi(\mathbf{X}) = [\mathbf{f}_\varphi(\mathbf{X}(t_1))^\top, \mathbf{f}_\varphi(\mathbf{X}(t_2))^\top, \cdots, \mathbf{f}_\varphi(\mathbf{X}(t_L))^\top]^\top \in \mathbb{R}^{dNL} \tag{2.3}$$

for each φ. Then we build the dictionary matrix

$$\mathbf{f}_\Theta(\mathbf{X}) = \begin{bmatrix} | & | & \cdots & | \\ \mathbf{f}_{\varphi_1}(\mathbf{X}) & \mathbf{f}_{\varphi_2}(\mathbf{X}) & \cdots & \mathbf{f}_{\varphi_n}(\mathbf{X}) \\ | & | & \cdots & | \end{bmatrix} \in \mathbb{R}^{dNL\times n}. \tag{2.4}$$

where n is the number of candidate functions φ in the library Θ. A crucial note is that if ϕ has a sparse representation in Θ, then so does the interaction force function $\mathbf{f}_\phi$ in the dictionary $\mathbf{f}_\Theta$. Leveraging this observation, we may set up a sparse regression problem for learning ϕ by solving the system of equations

$$\dot{\mathbf{X}} = \mathbf{f}_\Theta(\mathbf{X})\mathbf{c}, \tag{2.5}$$

where $\mathbf{c}$ is a sparse vector of coefficients that determine which nonlinear functions are included in the approximation.

Realistically, often only the data $\mathbf{X}$ is available, and $\dot{\mathbf{X}}$ must be approximated numerically, via the finite difference method or something similar. We will also investigate the case where our data is contaminated with noise, solving the noise perturbed version of (2.5):

$$\dot{\mathbf{X}} = \mathbf{f}_\Theta(\mathbf{X})\mathbf{c} + \sigma\mathbf{Z}, \tag{2.6}$$

where $\mathbf{Z}$ is a matrix of independently identically distributed Gaussian entries with zero mean, and σ is the noise magnitude and we assume that $\mathbf{Z}$ is also independent of μ_0. The noise model we adapt here can be viewed as a discretization of corresponding SDEs (Stochastic Differential Equations) with homogeneous Brownian noise and is used to model the random effects of the environment on the measurement of velocities. We point out that this assumption has appeared in [5].

In both cases, the sparse regression problem can be formulated as

$$\text{argmin}_{\mathbf{c}\in\mathbb{R}^n, \mathbf{c}\ \text{sparse}} \frac{1}{N}\|\dot{\mathbf{X}} - \mathbf{f}_\Theta(\mathbf{X})\mathbf{c}\|^2. \tag{2.7}$$

The above formulation can be easily generalized if we have multiple trajectory data starting at different initial conditions, which we shall specify it below. Now consider for general M, we have $\mathbf{X}^{(m)}(t) = \left[\mathbf{x}_1^{(m)}(t), \cdots, \mathbf{x}_N^{(m)}(t)\right] \in \mathbb{R}^{dN}$ and its corresponding derivative $\dot{\mathbf{X}}^{(m)}(t) \in \mathbb{R}^{dN}$, where $1 \leq m \leq M$. Then, for each trajectory, the state and the velocity matrix are

$$\mathbf{X}^{(m)} = \begin{bmatrix} \mathbf{X}^{(m)}(t_1) \\ \vdots \\ \mathbf{X}^{(m)}(t_L) \end{bmatrix} \in \mathbb{R}^{dNL} \quad \text{and} \quad \dot{\mathbf{X}}^{(m)} = \begin{bmatrix} \dot{\mathbf{X}}^{(m)}(t_1) \\ \vdots \\ \dot{\mathbf{X}}^{(m)}(t_L) \end{bmatrix} \in \mathbb{R}^{dNL}, \tag{2.8}$$

which are combined to form

$$\mathbf{X} = \begin{bmatrix} \mathbf{X}^{(1)} \\ \vdots \\ \mathbf{X}^{(M)} \end{bmatrix} \in \mathbb{R}^{dNML} \quad \text{and} \quad \dot{\mathbf{X}} = \begin{bmatrix} \dot{\mathbf{X}}^{(1)}(t_1) \\ \vdots \\ \dot{\mathbf{X}}^{(M)}(t_L) \end{bmatrix} \in \mathbb{R}^{dNML}. \tag{2.9}$$

Then, the sparse regression problem becomes

$$\operatorname{argmin}_{\mathbf{c}\in\mathbb{R}^n,\mathbf{c}\text{ sparse}}\frac{1}{NM}\sum_{m=1}^{M}\|\dot{\mathbf{X}}^{(m)}-\mathbf{f}_\Theta(\mathbf{X}^{(m)})c\|^2, \tag{2.10}$$

where the function library Θ and dictionary matrix $\mathbf{f}_\Theta(\mathbf{X}^{(m)})$ is defined as in (2.3). The (2.10) can be further simplified to

$$\operatorname{argmin}_{\mathbf{c}\in\mathbb{R}^n,\mathbf{c}\text{ sparse}}\frac{1}{NM}\sum_{m=1}^{M}\|\dot{\mathbf{X}}-\mathbf{f}_\Theta(\mathbf{X})c\|^2, \tag{2.11}$$

where $\mathbf{f}_\Theta$ is applied component wise to $\mathbf{X}$ so that $\mathbf{f}_\Theta(\mathbf{X})\in\mathbb{R}^{dNML\times n}$.

2.2. Methodology for Solving the Sparse Regression Problem. Let $X\in\mathbb{R}^{m\times n}, Y\in\mathbb{R}^n, \beta\in\mathbb{R}^m$. In this section, we review the approaches that find the sparse solution β such that

$$X\beta\approx Y. \tag{2.12}$$

2.2.1. Least Squares (LS). In the first regression method, the key in the calculation of the coefficient vector is the matrix division. The equation (2.12) is solved as a system of linear equation. For $M=1$, we used the "\" command in MATLAB to find the coefficient vector. For $M\geq 2$, we solve (2.10) by finding the normal equation and obtain the coefficient vector via

$$c=\left(\sum_{m=1}^{M}\sum_{l=1}^{L}(f_\Theta(\mathbf{X}^{(m)}(t_l)))^T f_\Theta(\mathbf{X}^{(m)}(t_l))\right)^{\dagger}\left(\sum_{m=1}^{M}\sum_{l=1}^{L}(f_\Theta(\mathbf{X}^{(m)}(t_l)))^T\dot{\mathbf{X}}^{(m)}(t_l)\right),$$

where the inverse ($^{-1}$) is chosen as the pseudo-inverse (†) in the code. Note that Least Squares is not a sparsity promoting technique, though it is used as a comparison to show the importance of sparsity promoting.

2.2.2. Sequential Least Squares (SLS). Sequential Least-Squares or sequential threshold least-squares is a modified version of Least Squares. An initial guess β is made by taking the Least Squares. As a result, SLS would perform very well if β is close enough to the true coefficient vector. Next, the algorithm checks for any coefficients in the coefficient vector that are below a selected threshold λ. Functions corresponding to these small coefficients are removed from the approximation by setting the coefficients exactly to zero and the Least Squares approximation is performed again and again with the remaining functions $\mathbf{f}_\varphi$. After no coefficients are below the threshold, the coefficient vector has been found. Note that, we pick the value of λ in the range from 1×10^{-13} to 1 (1 is not included), because λ larger than 1 would remove significant coefficients, and a threshold below than 1×10^{-13} would offer limited help since it is close to the machine precision. To find the fitting λ, we pick 50 values from this range, where the log of these λ's are equally spaced in the log range from -13 to 0.

Then, for each example or setting that uses SLS in the Numerical Section, we run through all the 50 λ's to select the one with smallest error, defined in equation (3.1). We repeat the above procedure about 20 times to get a set of $\lambda's$ with smallest error. Then, we use the mean of this set as our threshold in SLS for that specific setting. In practice, one can use the cross-validation approach described in [5] to pick λ. Here our rationale is to find λ in a range of values that yields best performance as a SLS estimator. We will show even in this case, they are not as good as Lasso estimators in data regime we considered.

2.2.3. Least Absolute Shrinkage and Selection Operator (LASSO).

The Least Absolute Shrinkage and Selection Operator is a sparse regression technique that uses an L_1-norm penalized term to balance model complexity with accuracy. In contrast to Least Squares regression, which returns only nonzero coefficients, LASSO can ensure that the β vector is sparse. Section 3 can be referenced to show a concrete example of the difference between Least Squares and LASSO regression. LASSO finds a sparse coefficient vector β by minimizing

$$\|Y - X\beta\|_2^2 + \lambda\|\beta\|_1.$$

In our code, we used MATLAB built-in 2-fold cross validation, where cross-validation is a method of segmenting data randomly to perform the regression on multiple data subsets for higher accuracy.

3. Numerical Results.

Numerical Setup. We consider a system with N agents in $\mathbb{R}^d$, and observe M trajectories, where the initial condition for each trajectory is sampled i.i.d according to the probability measure $\mu_0 = \text{Unif}([0, 10]^{Nd})$. We generated trajectory data over the time interval $[0, T]$ at L equidistant time instances. All systems are evolved using ode15s in MATLAB with a relative tolerance at 10^{-5} and absolute tolerance at 10^{-6}. Across all experiments, the parameters listed below remained constant, and we pick the value of threshold λ in SLS as the way described in section 2.2.2. The software used in this experiment is made available here [1].

Choice of Dictionary. In our numerical examples, the dictionary consists of single variable monomial functions and trigonometric functions. We selected a wide dictionary, 806 functions so that the regression was not feasible by manual calculation. We also restricted monomial functions to have exponents less than or equal to 5, $r^p, p \leq 5$, because MATLAB has a numerical limit of 2^{64}. As bounded functions, trigonometric functions are not impeded by this numerical limit.

- The specific library is

$$\{1, r, ..., r^5, \sin(r), \sin(2r)..., \sin(400r), \cos(r), \cos(2r), ..., \cos(400r)\}.$$

The large size of the dictionary decreases the probability of accidentally correct approximations. In this wide dictionary, the regression methods are forced to test more functions. That means that the regression method that performs best with the large dictionary will be the regression method that can perform even without intimate knowledge of the characteristic functions of a system.

[1] https://github.com/TomDongyangLi/HomogeneousSparseLearning

Evaluation Metrics. To evaluate the performance, we choose different number of agents N, different sample sizes M, L, and various noise levels σ; for each choice of $\{d, N, M, L, \sigma\}$, we report the results using the following metrics:

- **Interaction Kernel Estimation.** Performance of the regression techniques is seen primarily in the magnitude of the error norm of the true coefficient vector β and the approximate coefficient vector $\hat{\beta}$ as

$$E_{\hat{\beta}} = \frac{\|\beta - \hat{\beta}\|}{\|\beta\|}. \tag{3.1}$$

  We chose to calculate the relative error $E_{\hat{\beta}}$ as it is the standard practice among the scholars in the field. The threshold for a successful coefficient approximation is $E_{\hat{\beta}} < 1 \cdot 10^{-4}$.

- **Trajectory Prediction.** We compare the discrepancy between the true trajectories (evolved using ϕ) and approximated trajectories (evolved using $\hat{\phi}$) using the relative max-in-time trajectory error over time interval $[T_0, T_1]$ that is defined as

$$\left.\frac{\left\|\mathbf{X} - \hat{\mathbf{X}}\right\|}{\|\mathbf{X}\|}\right|_{\mathrm{TM}([T_0, T_1])} = \frac{\sup_{t \in [T_0, T_1]} \left\|\mathbf{X}(t) - \hat{\mathbf{X}}(t)\right\|}{\sup_{t \in [T_0, T_1]} \|\mathbf{X}(t)\|}.$$

3.1. Interaction Law as Linear Combination in Library.

We consider the interaction law given by

$$\phi(r) = r + 2\sin(r) - 4\cos(2r), \tag{3.2}$$

which is a continuous function. The studied model find applications in opinion dynamics [14, 17] where ϕ is often chosen as a positive function bounded above by 1. In our example, ϕ models short range repulsion and long range strong attraction, which can model aggregation of particles. Then the true coefficient vector β, with a length of 806, for the given library is

$$\beta(2) = 1,\ \beta(7) = 2,\ \beta(408) = -4,\ \text{else } \beta(n) = 0, \tag{3.3}$$

The true interaction law is displayed in Figure 1.

In this example, we choose the training time interval $[0, 0.5]$. The settings of the system are determined by the parameters $\{d, N, M, L, \sigma\}$. We have a linear system of size $dNML \times 806$ and therefore it is underdetermined as long as $dNML < 806$. We remark that the values of ϕ are not observed directly (we only observe a functional value of ϕ). Even in the case where $dNML \geq 806$, it is still a possible ill-posed problem. We refer the reader to [13] for the detailed analysis from the perspective of the inverse problem. We are interested in sparse, noisy data regimes, where the regression methods have fewer data to make their approximations, making the approximations more difficult to complete.

3.1.1. Constant Parameters Experiment.

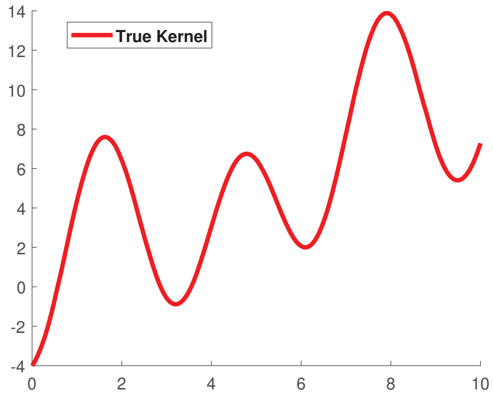

Figure 1. *Within the bounds of* $0 \leq r \leq 10$*, this interaction law has a minimum value of -4 and a maximum value of 13.8. This law is a continuous function over the bounds, and there are 6 inflection points,*

Experiment with a set of noise-free data. We began with $dNML = 240$ to find how little data could be provided to a regression method while still performing an accurate approximation. Specifically, the first system used the settings $\{d, N, M, L, \sigma\} = \{2, 10, 4, 3, 0\}$. This system was tested with ideal data capture, i.e., the position and velocity data are exact. Later, the noise is added to velocity observations to simulate a realistic setting. Note that following the procedure described in section 2.2.2, we have selected λ to be 2.23×10^{-4} in SLS.

The interaction law is oscillatory and increases on average. The bounds of the interaction law within the domain of $0 \leq r \leq 10$ are $-5 < \phi(r) < 15$.

Numerical results interpretation. In the ideal data case, only the LASSO regression could provide a satisfactory approximation of the interaction law. The Figure 2 makes it glaringly obvious that the LS and SLS methods have significant approximation errors, and we impute the failure of SLS to the fact that LS provided a poor initialization. Both the LS and SLS approximations exist far beyond the bounds of the interaction law. This large error can be attributed to the number of the nonzero coefficients of the LS and SLS approximated coefficient vector, $\hat{\beta}$. There are 3 nonzero terms in the true interaction law out of the 806 available functions in the library. For the chosen regression methods' $\hat{\beta}$, SLS and LS failed to produce sparse solutions. One can refer to Table 1 for numerical errors of three estimators. The errors $E_{\hat{\beta}}$ of LS and SLS is much larger than the one of LASSO regression. So in the setting $\{d, N, M, L, \sigma\} = \{2, 10, 4, 3, 0\}$, LASSO has the best performance. To confirm the performance of the approximations, we have plotted the trajectories for a given system of particles with the approximated interaction laws.

The Figure 3 is a collection of trajectory visualizations for this noise-free scenario. The top left plot is the visualization of the trajectory of the training IC (Initial Conditions) using the true interaction law. The top-right plot is the visualization of the trajectory of the training IC with the LASSO regression approximation. The regression was performed over data collected from the time interval $[0, 0.5]$ but the trajectory displayed is on time interval $[0, 10]$. This means that the interaction law learned over the smaller initial interval $[0, 0.5]$ was used to

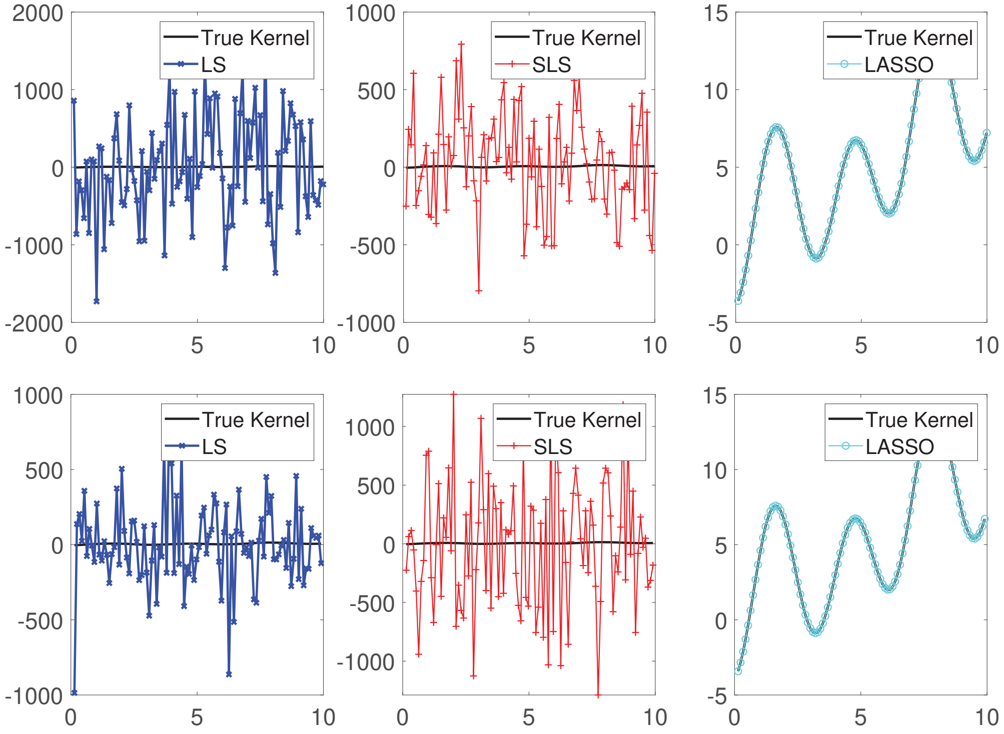

Figure 2. *In these 6 plots, the true interaction law is overlaid by the approximated interaction law from the 3 regression methods. The top 3 graphs are the approximations of the ideal data capture scenario. The bottom 3 graphs are the approximations in a system where 0.001 noise is added to data capture. The graphs from left to right are LS, SLS, and LASSO approximations. LS and SLS approximations have many fluctuations and larger magnitudes than the true interaction law. LASSO can perfectly mimic the original interaction law.*

Table 1

The normalized coefficient error of LS, SLS and LASSO estimators in Figure 2

Regression Method	Ideal System $E_{\hat{\beta}}$	Noisy System $E_{\hat{\beta}}$
Least Squares	$3.4 \cdot 10^2$	$6.9 \cdot 10^2$
Sequential Least Squares	$2.3 \cdot 10^2$	$3.7 \cdot 10^2$
LASSO	$9.4 \cdot 10^{-16}$	$4.0 \cdot 10^{-5}$

predict the trajectory in the larger second time interval $[0.5, 10]$. These two plots are visually identical, so LASSO was able to properly approximate the trajectory over the learning time interval $[0, 0.5]$, and was also able to predict the trajectory over the prediction time interval $[0.5, 10]$. Trajectory visualization was also performed for the LS and SLS methods, but these methods were entirely unable to provide a proper approximation. These plots have been omitted.

The bottom two graphs are also trajectory visualizations. The bottom left is a trajectory

Table 2

The trajectory prediction errors of LS, SLS and LASSO estimators in Figure 3

Regression Method	Training IC $[0, 0.5]$	Training IC $[0.5, 10]$	New IC $[0, 0.5]$	New IC $[0.5, 10]$
Least Squares	$4.2 \cdot 10^{-1}$	$4.7 \cdot 10^{-1}$	$3.2 \cdot 10^{-1}$	$3.6 \cdot 10^{-1}$
Sequential Least Squares	$4.2 \cdot 10^{-1}$	$4.8 \cdot 10^{-1}$	$3.2 \cdot 10^{-1}$	$3.6 \cdot 10^{-1}$
LASSO	$2.7 \cdot 10^{-13}$	$5.1 \cdot 10^{-13}$	$2.0 \cdot 10^{-13}$	$3.1 \cdot 10^{-13}$

visualization with the given interaction law on a new set of initial data, the New ICs. The bottom right is the trajectory visualization of the New ICs using the same approximation of $\phi(r)$ from the original ICs. This means that the previous LASSO estimator generalizes well: when given a new system, LASSO was still able to properly approximate the trajectory. However, SLS and LS again had no ability to properly approximate the trajectory. Again, the plots for these two methods have been omitted. From this figure, we can understand that incorrect approximations of the $\hat{\beta}$ like the ones for SLS and LS are unable to produce proper trajectories for an evolving system. One can refer to Table 2 for the trajectory prediction error of three estimators.

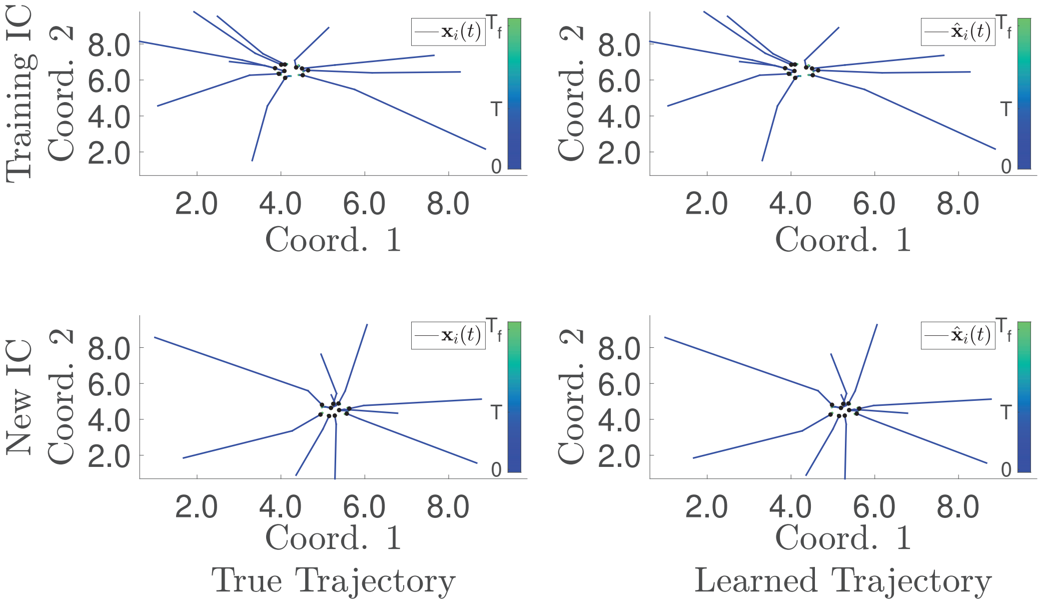

Figure 3. *Trajectory predictions produced by the LASSO estimator. Each line shows the evolution of the position (represented in 2D coordinates) for an agent in time interval* $[0, 10]$*. Top two panels are the true dynamics (left) and trajectory prediction by LASSO estimator (right) given an initial condition chosen from training data . Bottom two panels are the true dynamics (left) and LASSO-predicted trajectory given a new initial condition. The color bar (from blue to green) represents the start,* $t = 0$*, and ending point (displayed as green dots at the center of each panel),* $t = T_f$*, of the trajectory. The black dots indicate the time* $t = T = 0.5$*. One can refer to Table 2 for numerical trajectory prediction errors for three estimators.*

Experiment with a set of noisy data. To see the quality of these approximations in more practical situations, we add additive noise to the velocity data. Specifically, we chose $\mathbf{Z}$ in (2.6), to be the standard Gaussian vector with the unit variance, and we set σ to be 10^{-3}. This amount of noise was chosen to have a definite impact on the approximation, but not such a large noise that no approximation could be made. This experiment was executed for the same parameters $\{d, N, M, L\}$, though different initial conditions were used. This experiment also uses the same interaction law as in the ideal data capture experiment. We expected the addition of noise to bring more error into the approximations. Also, the value of λ in SLS is 5×10^{-2} for this setting.

Numerical results interpretation. Similar to the noise-free experiment, only LASSO regression was successful, and the LS and SLS regression methods again included most of the candidate functions to build the coefficient vector.

However, in calculations, LASSO's precision was reduced significantly as well. In this noisy scenario, the LASSO method found an error around 10^{-5}, this approximation is still considered a success. However, the increasing magnitude of the error suggests that there is a limit to how much noise can be added to a system before LASSO also fails. LS and SLS also had an increase in the magnitude of $E_{\hat{\beta}}$ in their systems, though not as sharp an increase as in the LASSO case. As seen by the Figure 2 plots, there is no improvement in these SLS and LS approximations of $\phi(r)$, and the SLS plot appears to have larger fluctuations in the noisy case. The LASSO approximation does not visibly appear to have less accuracy. In this noisy system, LASSO is the superior method of approximation.

3.1.2. Successful Rate of Recovery for Varying M.

Experiments with varying M . We then sought to calculate the success rate of each method as the size of observational data varies. Now we fix other parameters except for M and run the experiment for 100 independent trials. We say that a reconstruction of the kernel is successful if the relative reconstruction error, $E_{\hat{\beta}}$, is below 10^{-4}. We count how many trials yield successful reconstructions for three methods and calculate the corresponding probabilities. The results are summarized in Figure 4: the left panel is for noise-free data and the others are for noisy data.

Numerical results discussion. In the ideal data scenario, LASSO has a nonzero successful rate even at $M = 1$ and reach 100% when $M = 15$. SLS and LS succeed as the M rise in this experiment, where SLS succeeds sooner than LS. SLS reaches 100% at $M = 25$, so LASSO can provide a certain approximation of the interaction law with 60% of the data. Another notable difference is that LASSO reaches near certainty while the system is still underdetermined, but SLS and LS reach near certainty after $dNML > 806$. In the noisy scenario, for both thresholds (10^{-4} and 10^{-2}), LASSO and SLS improve similarly to the ideal data scenario as M increases. With the standard threshold 10^{-4}, LS is unable to have any success. It begins and ends the experiment with no successful trials. With a looser threshold 10^{-2}, LS is certainly successful when M is above 40. We observe that, if the reconstruction error of LS is sufficiently small, then SLS could produce a very accurate reconstruction.

One thing to note is that LASSO succeeds at $M = 15$ during the ideal case, but at the smaller $M = 3$ with noise $\sigma = 5 \cdot 10^{-4}$ under both thresholds. We impute this phenomenon to the randomness of the data. The existence of noise should enlarge the relative reconstruction

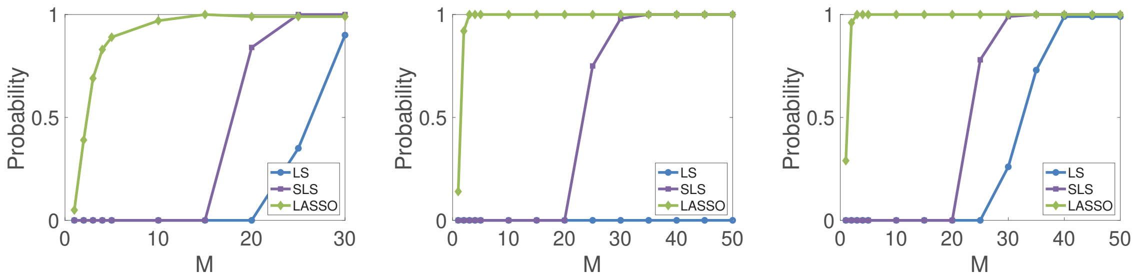

Figure 4. *This graphic displays probability of successful reconstruction by the three regression methods as M increases. The experiment settings are $\{d, N, M, L\}$=$\{2, 10, M, 3\}$, where $1 \leq M \leq 4, M = 5:5:30$. The training time interval is [0,0.5]. The green dot, purple square, and blue diamond lines are representing the LASSO, SLS, and LS regression methods respectively. The left graphic is a system where there is no noise, the following two graphics are systems with noise level $\sigma = 5 \cdot 10^{-4}$. In these noisy cases, M has been extended, so $1 \leq M \leq 4, M = 5:5:50$. The middle shows the probability of success if the relative reconstruction error is below 10^{-4}, and the right graph shows the success rate if the error is below 10^{-2}.*

error, which means LASSO should need higher M in the noisy case than in the ideal case.

LS and SLS decrease in accuracy with added noise. SLS success rate increases as M rises and requires a higher M in the noisy case as opposed to the ideal case.

3.1.3. Experiments over Varying d, N, M, L, σ.

Varying Parameters Experiment. In previous sections, LASSO has been shown as a strong regression method for the interaction law $\phi(r) = r + 2\sin(r) - 4\cos(2r)$. To collect more evidence, we tested in this section with various settings such as more particles, increasing noise, and increasing quantity of initial conditions. The tables are generated by the method with the best performance. In this case, it is always LASSO.

In the following Table 3, we display the normalized coefficient vector approximation error of LASSO regression for the given parameters. The results are the mean and standard deviation of $E_{\hat{\beta}}$'s calculated by 100 independent trials with i.i.d initial conditions sampled from μ_0. To further test LASSO performance in approximating the interaction law, we calculated the trajectory prediction error of LASSO over different parameters. The approximation is calculated within the training window of $[0, T]$, the prediction is made in the window of $[T, T_f]$. This prediction is called the trajectory approximation. Shown in Table 4, the results are the mean and standard deviation of trajectory prediction errors for the same 100 independent trials in calculating coefficient approximation error.

Numerical results interpretation. In Table 3, the top 7 lines are for $N = 10$ and the next 8 lines are for $N = 15$. In both sections, an increase in M will improve the accuracy of the approximation $E_{\hat{\beta}}$. The relative coefficient error drops significantly until it reaches the level of 10^{-15}. The existence of noise increases the error from 10^{-15} past 10^{-4}. Considering the threshold for a successful trial is 10^{-4}, this suggests the noise threshold is below 0.01. Also, in the $N = 15$ case, the impact of L on the regression was considered. The increase does force an improvement in accuracy, as seen by the lowest $E_{\hat{\beta}} = 6.4 \cdot 10^{-16}$, which can be considered as zero since machine precision is in 10^{-16}.

In Table 4, the pattern of trajectory error follows the pattern of coefficient error in Table

Table 3

Means and standard deviations of kernel reconstruction errors for different settings

$\{d, N, M, L, \sigma\}$	$E_{\hat{\beta}}$
$\{2, 10, 1, 3, 0\}$	$9.7 \cdot 10^{-1} \pm 4.3 \cdot 10^{-1}$
$\{2, 10, 2, 3, 0\}$	$4.6 \cdot 10^{-2} \pm 2.0 \cdot 10^{-1}$
$\{2, 10, 3, 3, 0\}$	$4.9 \cdot 10^{-3} \pm 4.9 \cdot 10^{-2}$
$\{2, 10, 5, 3, 0\}$	$8.6 \cdot 10^{-16} \pm 4.7 \cdot 10^{-16}$
$\{2, 10, 5, 3, 0.05\}$	$1.5 \cdot 10^{-3} \pm 9.0 \cdot 10^{-4}$
$\{2, 10, 5, 3, 0.01\}$	$2.9 \cdot 10^{-4} \pm 1.5 \cdot 10^{-4}$
$\{2, 10, 5, 3, 0.005\}$	$1.5 \cdot 10^{-4} \pm 8.2 \cdot 10^{-5}$
$\{2, 15, 1, 3, 0\}$	$3.8 \cdot 10^{-1} \pm 5.6 \cdot 10^{-1}$
$\{2, 15, 2, 3, 0\}$	$1.6 \cdot 10^{-15} \pm 1.8 \cdot 10^{-15}$
$\{2, 15, 3, 3, 0\}$	$1.4 \cdot 10^{-15} \pm 1.4 \cdot 10^{-15}$
$\{2, 15, 5, 3, 0\}$	$1.3 \cdot 10^{-15} \pm 1.1 \cdot 10^{-15}$
$\{2, 15, 5, 3, 0.05\}$	$1.8 \cdot 10^{-3} \pm 9.8 \cdot 10^{-4}$
$\{2, 15, 5, 3, 0.01\}$	$3.4 \cdot 10^{-4} \pm 2.1 \cdot 10^{-4}$
$\{2, 15, 5, 5, 0\}$	$6.8 \cdot 10^{-16} \pm 3.6 \cdot 10^{-16}$
$\{2, 15, 5, 11, 0\}$	$6.4 \cdot 10^{-16} \pm 3.3 \cdot 10^{-16}$

3. A similar theme was seen in Table 3, where an increase in noise would decrease the accuracy of the experiment. This sharp rise can be attributed to the magnitude of the noise. The apparent limits of the accuracy, in this case, were around 10^{-12}. What is shocking is that there were instances like line 2 and line 9 that had better accuracy in the new initial conditions than the training initial conditions. However, the standard occurrence was that the trajectory approximation during the training interval $[0, T]$ was more accurate than the trajectory approximation during the prediction interval $[T, T_f]$. Also, the trajectory approximation was more accurate during the original Training IC than the trajectory approximation for the New IC. However, these differences were not large, almost always within the same order of magnitude. This performance shows that a LASSO approximation of an interaction law learned from one set of data then can be used to accurately predict the future of that set or can be used to describe the evolution of the second set of data.

From these tables, we again conclude that LASSO is the most effective manner of regression in these underdetermined systems.

3.2. Interaction Law Out of the Dictionary.

In this example, we consider the interaction law given by

$$\phi(r) = e^{-r}, \tag{3.4}$$

Table 4

The mean and standard deviations of trajectory prediction errors for selected lines of Table 3

$\{d, N, M, L, \sigma\}$	Training IC $[0, 1]$	Training IC $[1, 10]$	New IC $[0, 1]$	New IC $[1, 10]$
$\{2, 10, 1, 3, 0\}$	$1.0 \cdot 10^{-1} \pm 8.0 \cdot 10^{-2}$	$1.1 \cdot 10^{-1} \pm 1.0 \cdot 10^{-1}$	$2.1 \cdot 10^{-1} \pm 6.1 \cdot 10^{-2}$	$2.6 \cdot 10^{-1} \pm 7.3 \cdot 10^{-2}$
$\{2, 10, 2, 3, 0\}$	$5.8 \cdot 10^{-3} \pm 2.5 \cdot 10^{-2}$	$9.0 \cdot 10^{-3} \pm 4.3 \cdot 10^{-2}$	$3.8 \cdot 10^{-12} \pm 1.8 \cdot 10^{-11}$	$8.0 \cdot 10^{-12} \pm 5.3 \cdot 10^{-11}$
$\{2, 10, 3, 3, 0\}$	$1.1 \cdot 10^{-4} \pm 1.1 \cdot 10^{-3}$	$1.3 \cdot 10^{-4} \pm 1.3 \cdot 10^{-3}$	$2.1 \cdot 10^{-12} \pm 6.4 \cdot 10^{-12}$	$4.0 \cdot 10^{-12} \pm 1.2 \cdot 10^{-11}$
$\{2, 10, 5, 3, 0\}$	$4.1 \cdot 10^{-12} \pm 2.8 \cdot 10^{-12}$	$9.8 \cdot 10^{-12} \pm 9.6 \cdot 10^{-12}$	$5.6 \cdot 10^{-12} \pm 4.3 \cdot 10^{-12}$	$1.1 \cdot 10^{-12} \pm 6.0 \cdot 10^{-12}$
$\{2, 10, 5, 3, 0.05\}$	$7.6 \cdot 10^{-4} \pm 1.6 \cdot 10^{-3}$	$4.7 \cdot 10^{-3} \pm 2.5 \cdot 10^{-2}$	$1.2 \cdot 10^{-3} \pm 2.4 \cdot 10^{-3}$	$7.5 \cdot 10^{-3} \pm 3.0 \cdot 10^{-2}$
$\{2, 10, 5, 3, 0.01\}$	$3.9 \cdot 10^{-4} \pm 1.0 \cdot 10^{-3}$	$2.3 \cdot 10^{-3} \pm 1.5 \cdot 10^{-2}$	$3.2 \cdot 10^{-4} \pm 1.0 \cdot 10^{-3}$	$2.7 \cdot 10^{-3} \pm 2.0 \cdot 10^{-2}$
$\{2, 10, 5, 3, 0.005\}$	$8.9 \cdot 10^{-5} \pm 2.0 \cdot 10^{-4}$	$7.2 \cdot 10^{-5} \pm 3.7 \cdot 10^{-4}$	$5.9 \cdot 10^{-5} \pm 7.3 \cdot 10^{-5}$	$3.8 \cdot 10^{-5} \pm 1.5 \cdot 10^{-4}$
$\{2, 15, 1, 3, 0\}$	$3.4 \cdot 10^{-2} \pm 5.8 \cdot 10^{-2}$	$3.7 \cdot 10^{-2} \pm 7.9 \cdot 10^{-2}$	$9.1 \cdot 10^{-2} \pm 5.0 \cdot 10^{-2}$	$7.3 \cdot 10^{-2} \pm 6.2 \cdot 10^{-2}$
$\{2, 15, 2, 3, 0\}$	$1.5 \cdot 10^{-12} \pm 3.2 \cdot 10^{-12}$	$3.6 \cdot 10^{-12} \pm 9.2 \cdot 10^{-12}$	$1.7 \cdot 10^{-12} \pm 5.6 \cdot 10^{-12}$	$3.6 \cdot 10^{-12} \pm 8.8 \cdot 10^{-12}$
$\{2, 15, 3, 3, 0\}$	$1.5 \cdot 10^{-12} \pm 3.7 \cdot 10^{-12}$	$3.2 \cdot 10^{-12} \pm 8.6 \cdot 10^{-12}$	$2.1 \cdot 10^{-12} \pm 7.7 \cdot 10^{-12}$	$5.9 \cdot 10^{-12} \pm 4.8 \cdot 10^{-12}$
$\{2, 15, 5, 3, 0\}$	$4.1 \cdot 10^{-12} \pm 3.1 \cdot 10^{-12}$	$9.8 \cdot 10^{-12} \pm 5.8 \cdot 10^{-12}$	$9.3 \cdot 10^{-12} \pm 1.1 \cdot 10^{-12}$	$1.3 \cdot 10^{-12} \pm 1.7 \cdot 10^{-12}$
$\{2, 15, 5, 3, 0.05\}$	$9.5 \cdot 10^{-4} \pm 1.4 \cdot 10^{-3}$	$1.5 \cdot 10^{-3} \pm 1.1 \cdot 10^{-2}$	$6.1 \cdot 10^{-4} \pm 1.0 \cdot 10^{-3}$	$2.5 \cdot 10^{-4} \pm 1.1 \cdot 10^{-3}$
$\{2, 15, 5, 3, 0.01\}$	$1.7 \cdot 10^{-4} \pm 3.6 \cdot 10^{-4}$	$1.4 \cdot 10^{-4} \pm 6.2 \cdot 10^{-4}$	$2.4 \cdot 10^{-5} \pm 5.1 \cdot 10^{-5}$	$9.3 \cdot 10^{-5} \pm 5.0 \cdot 10^{-4}$
$\{2, 15, 5, 5, 0\}$	$1.5 \cdot 10^{-12} \pm 9.5 \cdot 10^{-12}$	$3.2 \cdot 10^{-12} \pm 2.6 \cdot 10^{-12}$	$8.0 \cdot 10^{-12} \pm 6.3 \cdot 10^{-13}$	$1.9 \cdot 10^{-12} \pm 1.9 \cdot 10^{-12}$
$\{2, 15, 5, 11, 0\}$	$8.7 \cdot 10^{-12} \pm 1.2 \cdot 10^{-12}$	$2.0 \cdot 10^{-12} \pm 2.0 \cdot 10^{-12}$	$9.1 \cdot 10^{-12} \pm 1.0 \cdot 10^{-12}$	$9.4 \cdot 10^{-12} \pm 1.9 \cdot 10^{-11}$

which models homophilous opinion interactions. That is, each opinion tends to align with those that are similar. The Taylor series expansion of this interaction law is given as

$$\phi(r) = e^{-r} = \Sigma_{k=0}^{\infty} \frac{(-r)^k}{k!}. \tag{3.5}$$

This infinite series suggests that no finite dictionary can perfectly approximate this function. We are interested in this example, since ϕ does not lie in the span of the dictionary. We investigate this interaction law for two parameters, first $\{d, N, M, L\} = \{2, 10, 10, 3\}$ then $\{2, 10, 25, 3\}$. The first case will form an underdetermined matrix for regression, as the product $dNML$ is lesser than the number of candidate functions in the functions library. The second case will provide an overdetermined matrix for regression. We briefly investigate the performance of the proposed approaches.

Numerical results interpretation. The infinite series of this interaction law is made entirely of rational terms of the form $\frac{(-r)^k}{k!}, k \in \mathbb{Z}$. In the candidate function library, there are 6 rational terms and 800 trigonometric functions. In the domain $0 \leq r \leq 10$, the interaction law is bound as $e^{-10} \leq e^{-r} \leq 1$.

In the first case, we have an underdetermined linear system, the two approximated interaction laws from both LS and SLS exist beyond the bounds of the true interaction law. In contrast, the LASSO estimator tended to be zero. The interaction law grows steeper as the pairwise distance decreases. As the pairwise distance increases, the accuracy of the LASSO estimator improved.

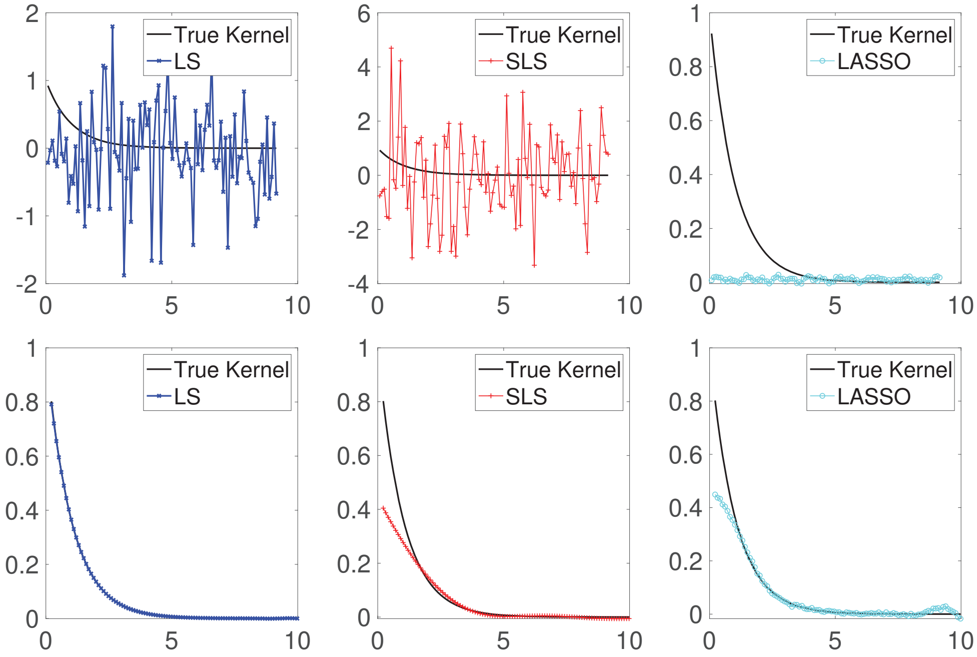

Figure 5. *This graphic displays reconstructions of the kernels by the three regression methods. We choose the training time interval [0,0.5].The blue star, red plus and cyan circle lines are representing the LS, SLS, and LASSO regression methods respectively. Top panel:* $\{d,N,M,L\} = \{2,10,10,3\}$. *Bottom panel:*$\{d,N,M,L\} = \{2,10,25,3\}$.

In the second case, we have an overdetermined system, and both LS and SLS have a great improvement in the accuracy of the interaction law approximation. In comparison to the first case, this is 250% more data in the second case. With this increase in data, the approximations through the conventional method of matrix division in least-squares find more success. The two approximations are contained within the bounds of the interaction law, and also take the shape of the interaction law. The same can be said for LASSO to an extent. In the underdetermined case, LASSO did not have the upward curve as the pairwise distance approached zero $r \to 0$. This curve does not take the exact shape very near zero, but it is closer to true than the SLS approximation.

In the underdetermined case, neither LS nor SLS generated sparse solutions, and the coefficient vector from LASSO only has a few non-zeros in the polynomial terms. In the overdetermined case, the coefficient vector from LS is still non-sparse while coefficients from SLS and LASSO are sparse. In all cases, none of the three methods generated the coefficient for term r^k that was close to $\frac{(-1)^k}{k!}$. This is a bit surprising, as LS provided a very accurate approximation of the true interaction law as shown in Figure 5.

However, instead of Taylor series, we might use Fourier series to explain the surprising accurate approximation of LS. The fourier series of e^{-r} consists of a constant term and infinite $\sin(nr)$ and $\cos(nr)$ terms. In our library Θ, n ranges from 1 to 400, so it could give us an approximation of e^{-r} with great accuracy. Also, there are only five terms, $r, r^2, \cdots, r^5$, that are not included in the fourier series, which means that the coefficient vector should not be sparse. Since there is no sparsity promoting techniques shown in LS, it then could have a more accurate result than SLS and LASSO.

3.3. Discussion. For the regression problem, we considered three regression techniques: LS, SLS, and LASSO. In low data systems, only LASSO would provide accurate approximations of the interaction law, and the LS and SLS approximations were unsatisfactory. In the experiments where the interaction law was a linear combination of candidate functions in the library, LASSO would need 4 variables for a 3 term polynomial ϕ, SLS and LS would make use of the entire library of 800+ functions. We also found that SLS is always as or sparser than LS. However, as seen in the case where the interaction law cannot be written as a linear combination of candidate functions, this sparsity can also reduce the accuracy of the approximation.

Across these experiments, the LASSO method was superior in the approximation of a coefficient vector to describe the interaction law in an underdetermined case, or where the size of the candidate functions library is larger than the product of the parameters $dNML$. These underdetermined experiments are important to find a more flexible regression method, as fewer data given to a regression method means less time for computation. We see that LASSO can work in a variety of underdetermined systems as in Table 3. As shown in Figure 3, the approximations calculated by LASSO are successful approximations of an underdetermined system and also can accurately predict the trajectory with new initial conditions. As such, we conclude that LASSO is the regression method that can correctly determine an interaction law in low-data systems.

4. Conclusions.

We considered a homogeneous agent system represented by a system of first-order ODEs. Agents in the system interact with one another through an unknown interaction law ϕ dependent on pairwise distances. This model has applications from opinion dynamics in social science. Given the parameters needed to create the system and an interaction law, we generate position data $\mathbf{X}$ and its derivative $\dot{\mathbf{X}}$, which will then be used to simulate data collected from a real-world scenario. With only these two matrices, position and trajectory, we may set up a sparse regression problem to learn ϕ from a library consisting of nonlinear candidate functions φ.

With the examples examined in this paper, we found that the success rate of the coefficient approximation increases as the number of sets of initial data increases. Among the three methods, LASSO performed very well even in the low data regime, while SLS and LS required more data to be successful. Specifically, when the true kernel does have a sparse representation in the dictionary, such as $\phi(r) = r + 2\sin(r) - 4\cos(r)$ in the first example, LASSO performed very well in the underdetermined case while LS and SLS performed very badly. After we increased M so the system became overdetermined, LS and SLS performed significantly better. When the true kernel ϕ does not have an exact sparse representation in the dictionary, such as $\phi(r) = \mathrm{e}^{-r}$ in the second example, all three methods performed poorly in the underdetermined case, and LS is the only one that performs well in the overdetermined case. One can refer to Table 5 for a concise summary.

Table 5
Summary of the performances of three methods for data regimes when kernel is in the library

Regression Method	Noise-free and Small M	Noise-free and Large M	Noisy and Small M	Noisy and Large M
LS	✗	okay	✗	okay
SLS	✗	✓	✗	okay
LASSO	✓	✓	✓	✓

We confirm that with large dictionaries and little data, of the choices Least Squares, Sequential Least-Squares, and Least Absolute Shrinkage and Selection Operator regression methods, the Least Absolute Shrinkage and Selection Operator regression method performs the strongest. For an arbitrary first-order homogeneous system, LASSO is likely to have more successful calculations than the other two methods.

Though not presented in this work, our research has also continued in this subject for the heterogeneous case by adding a second type of particle with separate interaction laws. Additionally, we have begun to add particle systems that do not have distinct particles but instead distribution functions that characterize particle position.

5. Acknowledgments. We would like to thank two anonymous reviews for their constructive feedback. We sincerely thank Dr. Sui Tang for being our mentor for this project and providing valuable insight and feedback. Part of this work was done in REU 2020 program at UCSB, in which Dr. Jea-Hyun Park served as a co-mentor. Our sincere gratitude also goes to Dr. Maribel Bueno for organizing the summer REU program where this research began. Also, we extend our thanks to Dr. Steven Brunton and his collaborators for their resourceful PNAS paper to reference. This research group acknowledges the University of California, Santa Barbara as its host institution. Hao-Tien Chuang and Shelby Malowney acknowledge the support through the National Science Foundation grant DMS-1850663. All of the authors were supported by NSF DMS-2111303.

REFERENCES

[1] N. Abaid and M. Porfiri, *Fish in a ring: spatio-temporal pattern formation in one-dimensional animal groups*, Journal of The Royal Society Interface, 7 (2010), pp. 1441–1453.

[2] G. Albi, D. Balagué, J. A. Carrillo, and J. von Brecht, *Stability analysis of flock and mill rings for second order models in swarming*, SIAM Journal on Applied Mathematics, 74 (2014), pp. 794–818.

[3] M. Ballerini, N. Cabibbo, R. Candelier, A. Cavagna, E. Cisbani, I. Giardina, V. Lecomte, A. Orlandi, G. Parisi, A. Procaccini, et al., *Interaction ruling animal collective behavior depends on topological rather than metric distance: Evidence from a field study*, Proceedings of the national academy of sciences, 105 (2008), pp. 1232–1237.

[4] V. D. Blondel, J. M. Hendrickx, and J. N. Tsitsiklis, *On krause's multi-agent consensus model with state-dependent connectivity*, IEEE transactions on Automatic Control, 54 (2009), pp. 2586–2597.

[5] S. L. Brunton, J. L. Proctor, and J. N. Kutz, *Discovering governing equations from data by sparse identification of nonlinear dynamical systems*, Proceedings of the national academy of sciences, 113 (2016), pp. 3932–3937.

[6] Y.-L. Chuang, M. R. D'orsogna, D. Marthaler, A. L. Bertozzi, and L. S. Chayes, *State transitions and the continuum limit for a 2d interacting, self-propelled particle system*, Physica D: Nonlinear

Phenomena, 232 (2007), pp. 33–47.

[7] F. Cucker and S. Smale, *Emergent behavior in flocks*, IEEE Transactions on automatic control, 52 (2007), pp. 852–862.

[8] M. Fornasier, J. Haškovec, and J. Vybíral, *Particle systems and kinetic equations modeling interacting agents in high dimension*, Multiscale Modeling & Simulation, 9 (2011), pp. 1727–1764.

[9] Y. Katz, K. Tunstrøm, C. C. Ioannou, C. Huepe, and I. D. Couzin, *Inferring the structure and dynamics of interactions in schooling fish*, Proceedings of the National Academy of Sciences, 108 (2011), pp. 18720–18725.

[10] U. Krause et al., *A discrete nonlinear and non-autonomous model of consensus formation*, Communications in difference equations, 2000 (2000), pp. 227–236.

[11] Q. Lang and F. Lu, *Learning interaction kernels in mean-field equations of 1st-order systems of interacting particles*, arXiv preprint arXiv:2010.15694, (2020).

[12] M. Lewin and X. Blanc, *The crystallization conjecture: a review*, EMS Surveys in Mathematical Sciences, 2 (2015), pp. 255–306.

[13] F. Lu, M. Maggioni, and S. Tang, *Learning interaction kernels in heterogeneous systems of agents from multiple trajectories*, Journal of Machine Learning Research, 22 (2021), pp. 1–67.

[14] F. Lu, M. Zhong, S. Tang, and M. Maggioni, *Nonparametric inference of interaction laws in systems of agents from trajectory data*, Proceedings of the National Academy of Sciences, 116 (2019), pp. 14424–14433.

[15] R. Lukeman, Y.-X. Li, and L. Edelstein-Keshet, *Inferring individual rules from collective behavior*, Proceedings of the National Academy of Sciences, 107 (2010), pp. 12576–12580.

[16] D. A. Messenger and D. M. Bortz, *Learning mean-field equations from particle data using wsindy*, arXiv preprint arXiv:2110.07756, (2021).

[17] S. Motsch and E. Tadmor, *Heterophilious dynamics enhances consensus*, SIAM review, 56 (2014), pp. 577–621.

[18] S. Patterson and B. Bamieh, *Interaction-driven opinion dynamics in online social networks*, in Proceedings of the First Workshop on Social Media Analytics, 2010, pp. 98–105.

[19] D. J. Sumpter, *Collective animal behavior*, Princeton University Press, 2010.

[20] E. Vedmedenko, *Competing interactions and pattern formation in nanoworld*, John Wiley & Sons, 2007.

[21] M. Zhong, J. Miller, and M. Maggioni, *Data-driven discovery of emergent behaviors in collective dynamics*, Physica D: Nonlinear Phenomena, 411 (2020), p. 132542.