


Integrating Simulation and Signal Processing with Stochastic Social Kinetic Model

Fan Yang and Wen Dong()

Department of Computer Science and Engineering,
State University of New York at Buffalo, Buffalo, USA
`wendong@buffalo.edu`

Abstract. Data that continuously track the dynamics of large populations have spurred a surge in research on computational sustainability. However, coping with massive, noisy, unstructured, and disparate data streams is not easy. In this paper, we describe a particle filter algorithm that integrates signal processing and simulation modeling to study complex social systems using massive, noisy, unstructured data streams. This integration enables researchers to specify and track the dynamics of complex social systems by building a simulation model. To show the effectiveness of this algorithm, we infer city-scale traffic dynamics from the observed trajectories of a small number of probe vehicles uniformly sampled from the system. The experimental results show that our model can not only track and predict human mobility, but also explain how traffic is generated through the movements of individual vehicles. The algorithm and its application point to a new way of bringing together modelers and data miners to turn the real world into a living lab.

1 Introduction

In this paper, we describe a particle filter approach to integrate signal processing and simulation modeling in the study of complex social systems using massive, noisy, unstructured data streams. As a result of this integration, researchers will be able to specify the dynamics of complex social systems by building a simulation model—a model that projects system-state changes over time—and applying the model to track complex systems and conduct thought experiments. We demonstrate this predictive ability in an application that tracks road transportation dynamics at city scale from the trajectories of probe vehicles and a state-of-the-art transportation simulator.

Data that continuously track the dynamics of large populations [1] have spurred a surge in research on social diffusion [4, 5], social network dynamics [8], and human mobility [12]. These data are often massive, noisy, and unstructured. Processing such data and turning them into useful knowledge is not easy. Traditional signal processing and pattern recognition models for capturing the dynamics in the data often lack intuitions about how component behaviors lead to predicted system behavior in a complex system, and have difficulty incorporating the effect of non-recurrent conditions.

Simulation modeling [2] is a method that captures system dynamics by simulating and collecting runtime system states. It is a widely adopted method for solving problems about complex systems, which is characterized by complex interdependence among components and nonlinear relationships between system behavior and component behavior. These simulated models have found widespread application in biology, industrial and systems engineering, economics, the social sciences, and education. A simulation model is easier to implement than an analytical model, and is intuitive to understand. However, it is a significant challenge to verify and validate a simulation model, and it is complicated to translate such a model from theoretical speculation into an enabling tool.

Our approach is to identify the simulation model of a complex system driven by a collection of events as a Markov process; to define the massive, noisy, and unstructured data streams as observations about this system; and to develop machine learning algorithms to track and learn the evolution of the latent state Markov process from these noisy observations. The key observation behind this approach is that a simulation model generates different sample paths with different probabilities, which are unambiguously identified by a sequence of events and the corresponding times at which these events occur. As such, the simulation defines a stochastic process with a probability measure assigned to the space of the sample paths that describe the interactions among elements in the system.

In this paper, we introduce the stochastic kinetic model to specify the dynamics of a complex social system with a set of events described by production rules, and design a particle filter algorithm to track the individuals in the system from noisy and partial observations and learn the system dynamics. A dynamic Bayesian network (DBN) [3, 16] can alternatively represent the dynamics of a complex system because the values of the random variables describing the system at time t are probabilistically dependent on those at time $t - 1$. In comparison with a DBN, an event-based model describes the system dynamics more succinctly by factoring the conditional probability of the system variables at two time steps into the probabilities for a sequence of events between these two time steps that incrementally change the system variables in simple ways. A deep neural network can potentially represent the arbitrarily complex dynamics of a system through a huge number of synaptic weights [9]; However, it requires a very large set of training data and huge computational resources to train these weights, and does not tell us how complex systems work microscopically, or what the consequences are of non-recurrent events. Variational inference is an alternative algorithm to track and learn complex system dynamics [20]. But it suffers from numerical stability issues when applied to complex systems.

The most creative and innovative aspect of this work is the integration of simulation modeling and signal processing in the study of complex social systems. This approach has not been explored because the intersection of the signal processing community and the simulation community is presently very small. However, this intersection is nonetheless very powerful because it affords an intuitive interpretation of the information extracted from massive, noisy, unstructured data streams. It has the potential to revolutionize how researchers use

simulation models, from running computer programs and analyzing program outputs to making inferences about a real-world system. For example, by integrating an agent-based transportation model with signal processing, we can not only simulate traffic jams during rush hour but also predict from the trajectories of probe vehicles whether today's traffic jams will be formed earlier or last longer than usual, and help drivers use the road network more efficiently. This kind of practical, transformative result is what happens when we bring together modelers and data miners to turn the real world into a living lab.

2 Stochastic Social Kinetic Model

We introduce the stochastic kinetic model to capture the dynamics of a complex social system driven by a set of events. A *stochastic kinetic model* is a biochemist's way of describing the temporal evolution of a biological network with M species driven by V mutually independent events [7, 19], where the stochastic effects are particularly prevalent (e.g., a transcription network or signal transduction network). Let $\mathbf{X} = (\mathbf{X}^{(1)}, \dots, \mathbf{X}^{(M)})$ denote the M species in the network. An event (chemical reaction) v is specified by a production

$$\alpha_v^{(1)} \mathbf{X}^{(1)} + \dots + \alpha_v^{(M)} \mathbf{X}^{(M)} \xrightarrow{c_v} \beta_v^{(1)} \mathbf{X}^{(1)} + \dots + \beta_v^{(M)} \mathbf{X}^{(M)}. \quad (1)$$

The production is interpreted as having *rate constant* c_v (probability per unit time, as time goes to 0). $\alpha_v^{(1)}$ individuals of species 1, $\alpha_v^{(2)}$ individuals of species 2 ... interact according to event v , resulting in their removal from the system. $\beta_v^{(1)}$ individuals of species 1, $\beta_v^{(2)}$ individuals of species 2 ... are introduced into the system. Hence event v changes the populations by $\Delta_v = (\beta_v^{(1)} - \alpha_v^{(1)}, \dots, \beta_v^{(M)} - \alpha_v^{(M)})$, and $S = (\Delta_1^\top, \dots, \Delta_V^\top)$ is therefore the *stoichiometry matrix*. The species on the left side of the production are *reactants*, the species on the right side of the production are *products*, and the species m with $\alpha_v^{(m)} = \beta_v^{(m)}$ are *catalysts*.

At the system level, let $x_t = (x_t^{(1)}, \dots, x_t^{(M)})$ be the populations of the species in the system at time t . A stochastic kinetic process initially in state x_0 at time $t = 0$ can be simulated through the Gillespie algorithm [7] by iteratively (1) sampling the time τ to the next event according to exponential distribution $\tau \sim \text{Exponential}(h_0(x_t, c))$, where $h_0(x, c) = \sum_{v=1}^V h_v(x_t, c_v)$ is the rate of all events and $h_v(x_t, c_v)$ is the rate of event v , (2) simulating the event v according to categorical distribution $v \sim (\frac{h_1}{h_0}, \dots, \frac{h_V}{h_0})$ conditional on event time τ , and accordingly (3) updating the system time $t \leftarrow t + \tau$ and populations $x \leftarrow x + \Delta_v$, until the termination condition is satisfied. In this algorithm, event rate $h_v(x_t, c_v)$ is the rate constant c_v multiplying a total of $\prod_{m=1}^M (x_t^{(m)})^{\alpha_v^{(m)}}$ different ways for individuals to interact in the system, assuming homogeneous populations. Exponential distribution is the maximum entropy distribution given the rate constant, and consequently is favored by the nature. The stochastic kinetic model thus assigns a probabilistic measure to a sample path induced by a sequence of events v_1, \dots, v_n happening between times 0 and T , $0 < t_1 < \dots < t_n < T$, which is

$$P(v_{1:n}, t_{1:n}, x_{1:n}) = \prod_{i=1}^n h_{v_i}(x_{t_{i-1}}, c_{v_i}) \exp\left(-\sum_{i=1}^n h_0(x_{t_{i-1}}, c)(t_i - t_{i-1})\right), \quad (2)$$

where

$$h_v(x, c_k) = c_v g_v(x) \text{ for } v = 1, \dots, V, \text{ and } h_0(x, c) = \sum_{v=1}^V h_v(x, c_v). \quad (3)$$

The stochastic kinetic model is one way to define a discrete event process, and its equivalents in other fields include the stochastic Petri net [10, 13], the system dynamics model [6], the multi-agent model specified through a flow chart or state chart [2], and the production rule system. As such, we argue that the stochastic kinetic model can capture the dynamics in not only biological networks but also social networks. For example, epidemiologists use productions such as “ $S + I \rightarrow 2I$ ” and “ $I \rightarrow R$ ” to represent infection and recovery events, and economists have developed the predator-prey model (“prey $\rightarrow 2$ prey,” “prey + predator $\rightarrow 2$ predator,” and “predator $\rightarrow \emptyset$ ”) to represent the interactions among different industries. In our research, we use events of the form “ $p_i \circ l_j \rightarrow p_i \circ l_k$ ” to specify the dynamics in a road transportation network in terms of vehicle movements: a vehicle previously at location j moves to location k (Fig. 1).

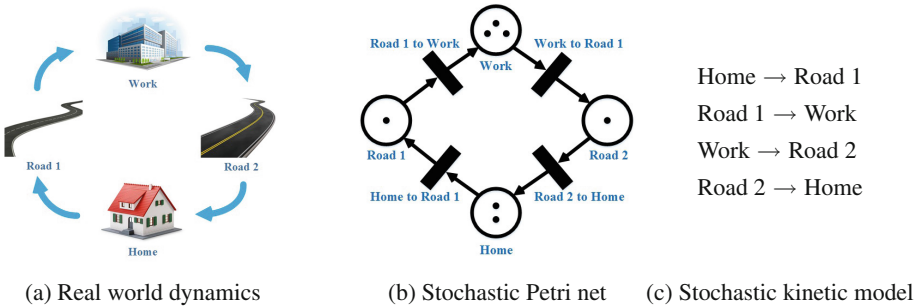


Fig. 1. Several representations of a complex system. (a) Real-world complex system dynamics, (b) a stochastic Petri net representation, (c) a stochastic kinetic model representation.

Computational social scientists often specify the complex dynamics of a social system with discrete event simulator software. To identify such a discrete event simulator as a Markov process and use the simulator to track real-world social systems with continued observations, we exploit the fact that all discrete event simulators (at least, to the best of our knowledge) have a way to dump the events happening in a simulation run. As such, we can reconstruct simulation runs according to the event sequences and so reconstruct the stochastic discrete event model from simulation runs outside the simulator—instead of unearthing the source code of a specific simulator over many man-months. For example, rather than hacking through the 140 thousand lines of code for MATSIM (which is a

state-of-the-art multi-agent transportation simulator) to make real-time inferences with real-world data, we can dump four events: vehicle leaving a building, vehicle entering a link, vehicle leaving a link, and vehicle entering a building. From these four events, we can construct a state transition matrix to represent vehicle dynamics.

Although the stochastic kinetic model is a continuous time model, we work with a discrete time stochastic model in the rest of this paper because our goal is to track stochastic kinetic dynamics from observations of populations or individuals with countably many computational steps. To this end, we approximate the continuous time process with a discrete time process on a countable set of equally spaced time points $0, \tau, 2\tau, \dots$, with a time interval so small that the probability of more than one event happening in the interval τ is negligible. This approximation works because the state transition kernel from time 0 to time τ is $p(x_0 \rightarrow x_\tau) = \sum_{n=0}^{\infty} \left(I + \frac{Q}{\gamma}\right)^n \exp(-\gamma\tau) \frac{(\gamma\tau)^n}{n!}$ according to the uniformization method [11], where γ is a uniformization rate, I the identity matrix and Q the infinitesimal generator defined by $h_k, k = 1, \dots, V$. With $\gamma \rightarrow \infty$ and $\gamma\tau = 1$, we get a first-order approximation of the state transition kernel $I + Q \cdot \tau$.

Specifically, let v_1, \dots, v_T be a sequence of events in the discrete time stochastic kinetic system, x_1, \dots, x_T a sequence of states (populations of species), and y_1, \dots, y_T a set of observations about the populations. Our goal is to make inferences about $\{v_t, x_t : t = 1, \dots, T\}$ from $\{y_t : t = 1, \dots, T\}$ according to the following probability measure, where indicator function $\delta(x_t - x_{t-1} = \Delta_{v_t})$ is 1 if the previous state is x_{t-1} and the current state is $x_t = x_{t-1} + \Delta_{v_t}$, and 0 otherwise.

$$P(v_{1,\dots,T}, x_{1,\dots,T}, y_{1,\dots,T}) = \prod_{t=1}^T P(x_t, y_t, v_t | x_{t-1}), \quad (4)$$

$$\text{where } P(x_t, y_t, v_t | x_{t-1}) = P(v_t | x_{t-1}) \delta(x_t - x_{t-1} = \Delta_{v_t}) P(y_t | x_t), \quad (5)$$

$$\text{and } P(v_t | x_{t-1}) = \begin{cases} c_k \tau \cdot g_k(x_{t-1}) & \text{if } v_t = k \\ 1 - \sum_j c_j \tau g_j(x_{t-1}) & \text{if } v_t = \emptyset \end{cases}. \quad (6)$$

3 Particle Filter with Stochastic Kinetic Model

We apply a particle filter to track the dynamics of a stochastic kinetic process. The particle filter maintains a collection of particles x_t^k for $k = 1, \dots, N$ and $t = 1, \dots, T$ to represent the likelihood of the latent state of a stochastic process x_t at different regions of the state space with each particle representing a system state, given noisy and partial observations $y_{1,\dots,T}$. It tracks the evolution of a stochastic process by alternately mutating the collection of particles according to stochastic process dynamics $P(x_t | x_{t-1})$ and selecting the particles according to observations $P(y_t | x_t)$. In comparison with a particle filter, a simulation run uses only one particle and does not use observations to perform particle selection.

Specifically, let x_t^k for $k = 1, \dots, N$ be the collection of particle positions and v_t^k the corresponding events from particle mutation, and $i_t^k \in \{1, \dots, N\}$ be the

collection of particle indexes from particle selection. To make inferences about the latent state x_t of a stochastic process starting at state x_0 from observations $y_{1:t}$, we initialize particle positions and indexes as $x_0^1, \dots, x_0^N = x_0$ and $i_0^1 = 1, \dots, i_0^N = N$, and iteratively sample the next event v_t^k according to how likely it is that different events will occur conditioned on system state $x_{t-1}^{i_{t-1}^k}$ for $k = 1, \dots, N$ (Eq. 7), then update $x_t^k = x_{t-1}^k + \Delta_{v_t^k}$ accordingly (Eq. 8) and resample these events per their likelihoods with regard to the observation y_t for $t = 1, \dots, T$ (Eq. 9).

$$v_t^k | x_{t-1}^{i_{t-1}^k} \sim \text{Categorical}\left(1 - \frac{h_0(x_{t-1}^{i_{t-1}^k})}{\gamma}, \frac{h_1(x_{t-1}^{i_{t-1}^k})}{\gamma}, \dots, \frac{h_V(x_{t-1}^{i_{t-1}^k})}{\gamma}\right), \quad (7)$$

$$x_t^k = x_{t-1}^{i_{t-1}^k} + \Delta_{v_t^k}, \quad (8)$$

$$i_t^k | (x_t^{1:N}, y_t) \sim \text{Categorical}(p(y_t | x_t^1), \dots, p(y_t | x_t^N)). \quad (9)$$

To determine a particle trajectory from the posterior distribution of a stochastic kinetic process with respect to observations, we trace back the events that lead to the particles x_T^k for $k = 1, \dots, N$:

$$x_0, v_1^{j_1^k}, x_1^{j_1^k}, \dots, v_T^{j_T^k}, x_T^{j_T^k}, \text{ where } j_T^k = i_T^k, j_{T-1}^k = i_{T-1}^k, \dots, j_1^k = i_1^k. \quad (10)$$

To learn the rate constants (the parameters) of a stochastic kinetic model, we sample from the posterior distribution of these parameters conditioned on the particle trajectory, using a beta distribution as conjugate prior. Let v_1, \dots, v_T be the sequence of events in a particle trajectory (Eq. 10). The posterior probability distribution of a rate constant is a beta distribution that matches the expected number of events in a sample path with the number of events that actually occur, where a_v and b_v are hyper-parameters:

$$c_v | v_{1:T}, x_{1:T} \sim \text{Beta}(a_v + \sum_{i=1}^T \delta_v(v_t), b_v + \sum_{t=0}^T g_v(x_t) - \sum_{t=1}^T \delta_v(v_t)). \quad (11)$$

Overall, then, we have developed a particle-based algorithm to make inferences about a complex system using a stochastic kinetic model and noisy observations (Algorithm 1).

The discrete event model developed in this paper can be extended into a Markov discrete event decision process (MDEDP) to study how individuals make decisions probabilistically in order to jointly maximize a time-discounted future reward according to their perceptions of the world. To this end, we introduce action variables that are probabilistically dependent on the current system state for individuals to control the event rates, and specify the rewards for an individual to be in different states for a unit time. To tractably solve the MDEDP, we can first reduce its state value function into the probability of receiving a reward in a mixture of dynamics Bayesian networks [18], then search into the future for the best individual actions with the particle filter algorithm.

Algorithm 1. Particle-Based Inference with Stochastic Kinetic Model

Input: Observations y_1, \dots, y_T of a stochastic kinetic process (Eq. 4) specified by a set of events (Eq. 1 and 6 for $v = 1, \dots, V$).

Output: Resampled particles $(v_t^{i_k}, x_t^{i_k})_{t=1:T}^{k=1:N}$ from particle filter, particle trajectories $(v_t^{j_k}, x_t^{j_k})_{t=1:T}^{k=1:N}$ from particle smoother.

Procedure:

- Initialize $x_0^1 = \dots = x_0^N = x_0, i_0^1 = 1, \dots, i_0^N = N$.
- (Filtering) For t in $1, \dots, n$ and k in $1, \dots, N$, sample v_t^k and i_t^k according to Eq. 7, 8 and 9, where $p(y_t|x_t)$ is defined in Eq. 4.
- (Smoothing) Back-track particle trajectory from $x_T^{i_k}$ according to Eq. 10, for $k = 1, \dots, N$.
- (Parameter Learning) Sample rate constants according to Eq. 11.

4 Tracking City-Scale Transportation Dynamics

In this section, we evaluate the performance of the particle filter in continuously tracking current and future traffic conditions at city scale from an event-based transportation model and the sporadically observed locations of probe vehicles uniformly sampled from the system.

4.1 Modeling Traffic Dynamics

We model road traffic dynamics through a single type of event— $p_i \circ l_j \rightarrow p_i \circ l_k$ —a vehicle i moving from link/building j to link/building k with rate constant c_{l_j, l_k} , changing its location from $X_t^{(p_i)} = l_j$ to $X_{t+1}^{(p_i)} = l_k$, changing the number of vehicles on link l_j from $X_t^{(l_j)} = x_t^{(l_j)}$ to $X_{t+1}^{(l_j)} = x_t^{(l_j)} - 1$, and changing the number of vehicles on link l_k from $X_t^{(l_k)} = x_t^{(l_k)}$ to $X_{t+1}^{(l_k)} = x_t^{(l_k)} + 1$. According to this model, a vehicle stays at link/building j for an average duration $1/\sum_k c_{l_j, l_k}$ and on exit chooses a downstream link/building with a probability proportional to the rate constant $c_{l_j, l_k}/\sum_{k'} c_{l_j, l_{k'}}$. Here we use “ \circ ” to represent a bond: person i binds to location j before the event and binds to location k after the event.

We assume that probe vehicles are uniformly sampled from the system. Let x_{ttl} be the total number of vehicles in the system and y_{ttl} be the total number of observed vehicles. The probability of observing $y_t^{(l_j)}$ probe vehicles at location j conditioned on that there are $x_t^{(l_j)}$ vehicles in total is $p(y_t^{(l_j)}|x_t^{(l_j)}) = \binom{x_t^{(l_j)}}{y_t^{(l_j)}} \binom{x_{\text{ttl}} - x_t^{(l_j)}}{y_{\text{ttl}} - y_t^{(l_j)}} / \binom{x_{\text{ttl}}}{y_{\text{ttl}}}$. When the total number of vehicles in the system is large, the percentage of probe vehicles at a given link/building is roughly the percentage in the system.

4.2 Experimental Setup

Here we compare the performance of the particle filter against other algorithms on three datasets of human mobility: SynthTown, Berlin and Dakar. The SynthTown dataset is comprised of a synthesized network of one home location, one work location, and 23 single-direction road links, with the trips of 2000 synthesized inhabitants going to work in the morning and going home in the evening [14]. This dataset is small enough for studying how different algorithms work. The Berlin dataset is comprised of a network of 24,000 single-direction road links derived from Open Street Map and the trips of 9000 synthesized vehicles representing the travel behavior of one million vehicles. The trips in the Berlin dataset were carefully validated with survey and sensor network data, and thus provide the ground truth for evaluating algorithms in a semi-realistic configuration [21]. The Dakar dataset is comprised of a network of 8000 single-direction road links derived from Open Street Map and 12,000 real-world vehicle trips derived from Data for Development call detail records [15].

To evaluate the effectiveness of our model, we compare the stochastic kinetic model (PFSKM) with a deep neural network (DNN) [9], with a recurrent neural network (RNN) [9], and with a dynamic Bayesian network (EKF) [17]. The vanilla neural networks represent the power of a general-purpose non-parametric model that does not involve a problem-specific structure. The dynamic Bayesian network characterizes the problem-specific structure through probability dependence among its random variables, and includes a suite of inference and structure learning algorithms. All models are trained with 30 days of traffic data and evaluated with a separate single day of testing data.

We use two metrics to evaluate the performance of our model: coefficient of determination (R^2) and mean squared error (MSE). We use R^2 to evaluate the goodness of fit between a time series of the estimated vehicle counts at a location and the ground truth. Let f_t be the estimated vehicle count at time t , y_t the ground truth and \bar{y} the average of y_t . We define $R^2 = 1 - \sum_t (f_t - y_t)^2 / \sum_t (y_t - \bar{y})^2$. A higher R^2 indicates a better fit between the estimated time series and the ground truth, with $R^2 = 1$ indicating a perfect fit and $R^2 < 0$ a fit worse than using the average. We use MSE to measure the average squared error difference between the estimated vehicle counts at all locations at a given time t and the ground truth. Let $f^{(i)}$ be the estimated vehicle count at location i and $y^{(i)}$ the ground truth. We define $\text{MSE} = \frac{1}{n} \sum_{i=1}^n (y^{(i)} - f^{(i)})^2$. A lower MSE indicates an estimation closer to the ground truth.

4.3 Evaluation Results

Figure 2 compares the summary MSE and R^2 performance statistics for the four models in vehicle tracking—i.e., estimating the numbers of vehicles up to now, short-term prediction (10 min) and long-term prediction (1 h) on all data sets. The Dakar dataset is too large for DNN, RNN, and EKF, which indicates the better scalability of PFSKM. PFSKM has the lowest MSE across different times of day, which is followed by DNN, EKF, and RNN in order (top row, lower

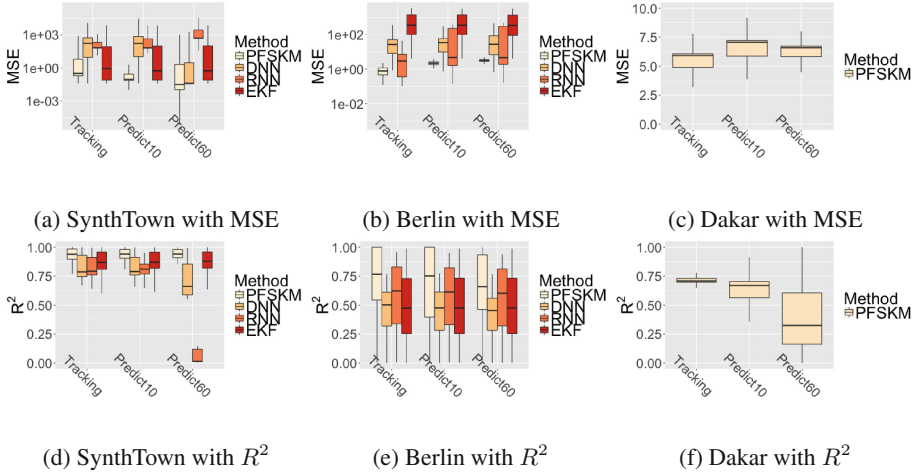


Fig. 2. Performance of PFSKM, DNN, RNN and EKF on the SynthTown, Berlin and Dakar datasets using MSE (top, lower MSE indicating better performance) and R^2 (bottom, higher R^2 indicating better performance).

is better). PFSKM also has the highest R^2 across different locations, which is followed by DNN, EKF, and RNN (bottom row, higher is better). PFSKM outperforms RNN and DNN because it can explicitly leverage problem-specific structures such as road topology. It outperforms EKF because it can work with arbitrary probability distributions, and sometimes Gaussian assumption is not a good approximation for real-world applications.

Figure 3 illustrates the output of PFSKM with four snapshots of the Dakar road network (bottom row) at 6am, noon, 6pm, and midnight, corresponding to four points in the particle trajectory. Each particle trajectory is an uninterrupted

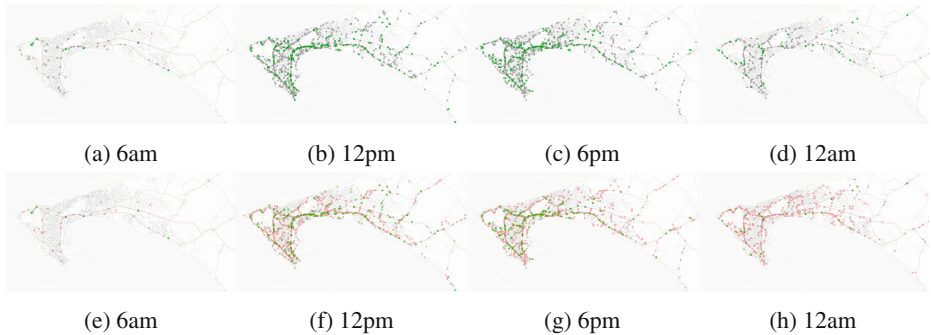


Fig. 3. Dakar region traffic estimation and ground truth at four time points. The bottom figures are the estimation result, and the top figures are the ground truth. (Color figure online)

simulation in which the vehicles move from one location to another, explaining how traffic is generated with the movements of individual vehicles and what the consequences of non-recurrent events are. We have also provided the ground truth snapshots at the same time for reference (top row).

In these snapshots, probe vehicles are shown as green dots, simulated vehicles as red dots, and non-probe vehicles from the ground truth as black dots. There is no correspondence between the non-probe vehicles in the particle trajectory and those in the ground truth, because the non-probe vehicles are simply not observable. However, the vehicle densities at different locations of the road network and in the ground truth agree with each other, and both are proportional to the densities of probe vehicles. This is because for the particle filter, we have continually selected the most likely system evolution directions conditioned on the probe vehicle movements, and we backtrack these system evolution in particle smoother.

5 Conclusions

In this paper, we have described a particle filter algorithm for our stochastic social kinetic model that integrates signal processing and simulation modeling to study complex social systems using massive, noisy, unstructured data streams. Our method outperforms neural networks and the extended Kalman filter in inferring city-scale road traffic from continued observations of a small number of probe vehicles uniformly sampled from the system. This method points to a new way of bringing together modelers and data miners by turning the real world into a living lab.

References

1. Blondel, V.D., Decuyper, A., Krings, G.: A survey of results on mobile phone datasets analysis (2015). arXiv preprint: [arXiv:1502.03406](https://arxiv.org/abs/1502.03406)
2. Borshchev, A.: The Big Book of Simulation Modeling: Multimethod Modeling with AnyLogic 6. AnyLogic North America, Chicago (2013)
3. Boyen, X.: Inference and learning in complex stochastic processes. Ph.D. thesis, Stanford University (2002)
4. Castellano, C., Fortunato, S., Loreto, V.: Statistical physics of social dynamics. *Rev. Mod. Phys.* **81**(2), 591–646 (2009)
5. Dong, W., Heller, K., Pentland, A.S.: Modeling infection with multi-agent dynamics. In: Yang, S.J., Greenberg, A.M., Endsley, M. (eds.) SBP 2012. LNCS, vol. 7227, pp. 172–179. Springer, Heidelberg (2012). doi:[10.1007/978-3-642-29047-3_21](https://doi.org/10.1007/978-3-642-29047-3_21)
6. Forrester, J.W.: Industrial Dynamics. MIT Press, Cambridge (1961)
7. Gillespie, D.T.: Stochastic simulation of chemical kinetics. *Annu. Rev. Phys. Chem.* **58**, 35–55 (2007)
8. Goldenberg, A., Zheng, A.X., Fienberg, S.E., Airolidi, E.M.: A survey of statistical network models. *Found. Trends® Mach. Learn.* **2**(2), 129–233 (2010)
9. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press, Cambridge (2016)

10. Goss, P.J., Peccoud, J.: Quantitative modeling of stochastic systems in molecular biology by using stochastic petri nets. *Proc. Natl. Acad. Sci.* **95**(12), 6750–6755 (1998)
11. Grassmann, W.K.: Transient solutions in Markovian queueing systems. *Comput. Oper. Res.* **4**, 47–53 (1977)
12. Guan, T., Dong, W., Koutsonikolas, D., Qiao, C.: Fine-grained location extraction and prediction with little known data. In: *Proceedings of the 2017 IEEE Wireless Communications and Networking Conference*. IEEE Communications Society (2017)
13. Marsan, M.A., Balbo, G., Conte, G., Donatelli, S., Franceschinis, G.: *Modelling with Generalized Stochastic Petri Nets*. Wiley, New York (1994)
14. MATSim Development Team (eds.): *MATSIM-T: aims, approach and implementation*. Technical report, IVT, ETH Zürich, Zürich (2007)
15. de Montjoye, Y.A., Smoreda, Z., Trinquart, R., Ziemlicki, C., Blondel, V.D.: D4D-Senegal: the second mobile phone data for development challenge (2014). arXiv preprint: [arXiv:1407.4885](https://arxiv.org/abs/1407.4885)
16. Murphy, K.P.: *Dynamic Bayesian networks: representation, inference and learning*. Ph.D. thesis, University of California, Berkeley (2002)
17. Smith, G.L., Schmidt, S.F., McGee, L.A.: Application of statistical filter theory to the optimal estimation of position and velocity on board a circumlunar vehicle. National Aeronautics and Space Administration (1962)
18. Toussaint, M., Storkey, A.: Probabilistic inference for solving discrete and continuous state Markov decision processes. In: *Proceedings of the 23rd International Conference on Machine Learning*, pp. 945–952. ACM (2006)
19. Wilkinson, D.J.: *Stochastic Modelling for Systems Biology*. CRC Press, Boca Raton (2011)
20. Xu, Z., Dong, W., Srihari, S.N.: Using social dynamics to make individual predictions: variational inference with stochastic kinetic model. In: *Advances in Neural Information Processing Systems*, pp. 2775–2783 (2016)
21. Ziemke, D., Nagel, K., Bhat, C.: Integrating CEMDAP and MATSim to increase the transferability of transport demand models. *Transp. Res. Rec. J. Transp. Res. Board* **2493**, 117–125 (2015)