Calibrating Agent-based Models to Microdata with Graph Neural Networks

Joel Dyer [1 2] **Patrick Cannon** [3] **J. Doyne Farmer** [1 2 4] **Sebastian M. Schmon** [3 5]

Abstract

Calibrating agent-based models (ABMs) to data is among the most fundamental requirements to ensure the model fulfils its desired purpose. In recent years, simulation-based inference (SBI) methods have emerged as powerful tools for performing this task when the model likelihood function is intractable, as is often the case for ABMs. In some real-world use cases of ABMs, both the observed data and the ABM output consist of the agents' states and their interactions over time. In such cases, there is a tension between the desire to make full use of the rich information content of such granular data on the one hand, and the need to reduce the dimensionality of the data to prevent difficulties associated with high-dimensional learning tasks on the other. A possible resolution is to construct lower-dimensional time-series through the use of summary statistics describing the macrostate of the system at each time point. However, a poor choice of summary statistics can result in an unacceptable loss of information from the original dataset, dramatically reducing the quality of the resulting calibration. In this work, we instead propose to learn parameter posteriors associated with granular microdata directly using temporal graph neural networks. We will demonstrate that such an approach offers highly compelling inductive biases for Bayesian inference using the raw ABM microstates as output.

1. Introduction

Agent-based models (ABMs) are becoming a popular modelling tool in a variety of disciplines, from economics (Baptista et al., 2016) to epidemiology (Ferguson et al., 2020). They offer domain experts a high degree of flexibility in modelling complex systems, for example by naturally incorporating interactions between, and heterogeneity across, agents in the system.

Typically, ABMs are stochastic, dynamic models in which the states $\mathbf{z}^t = (\mathbf{z}_1^t, \dots, \mathbf{z}_N^t)$ of a set of N interacting agents, labelled $i = 1, \dots, N$, are simulated over time $t \in [0, T]$. We assume here that the ABM progresses in discrete[1] timesteps $t = 0, 1, \dots, T$, and that the agent states may be multidimensional such that $\mathbf{z}_i^t \in \mathbb{R}^K$ for some $K \geq 1$. ABMs further rely on a potentially time-varying graph structure – represented as an adjacency matrix $\mathbf{w}^t \in \mathbb{R}^{N \times N}$ – that reflects, for example, the strength and/or valence of the relationship between pairs of agents, or the set of pairwise interactions that can take place during the simulation. Once a set of parameters $\boldsymbol{\theta} \in \boldsymbol{\Theta} \subset \mathbb{R}^D$ and the initial states $\mathbf{z}^0$ and $\mathbf{w}^0$ are specified, the agent behaviours and interactions are simulated, and a time-series $\mathbf{x} = (\mathbf{x}^1, \dots, \mathbf{x}^T)$ is generated as output. Typically, $\mathbf{x}^t \in \mathbb{R}^M$ for some $M \geq 1$ and, owing to the stochastic nature of many ABMs, the output time-series $\mathbf{x}$ may furthermore differ between simulations generated by the same $\boldsymbol{\theta}$, $\mathbf{z}^0$, and $\mathbf{w}^0$.

The model output $\mathbf{x}$ is often taken to be some aggregate statistics describing the macrostate of the ABM over time, that is $\mathbf{x} = g(\mathbf{w}, \mathbf{z})$ for some aggregation function g. In some cases, this is done out of necessity: it is sometimes the case that only aggregate data is available from the real-world system the ABM is designed to mirror and, consequently, the ABM can only be compared to reality through the lens of this aggregate data. Under these circumstances, two natural inference tasks arise:

1. parameter inference – i.e. inferring the fixed parameters $\boldsymbol{\theta}$; and

2. latent state inference (filtering and smoothing) – i.e. inferring either or both the latent agent states $\mathbf{z}^t$ and the latent agent-agent relationships $\mathbf{w}^t$ over time.

Both tasks are complicated by the fact that the relevant marginal and conditional likelihood functions are in general unavailable to compute, due to the complexity of ABMs. Intractable likelihoods are encountered widely across

[1]Institute for New Economic Thinking, Oxford [2]Mathematical Institute, University of Oxford [3]Improbable, London [4]Santa Fe Institute [5]Durham University. Correspondence to: Joel Dyer <joel.dyer@maths.ox.ac.uk>.

Spotlight paper at the ICML 2022 workshop on *Artificial Intelligence for Agent-based Modelling*. Copyright 2022 by the author(s).

[1]We discuss in Section 5 how this assumption may be relaxed.

model types and application domains and, consequently, significant research effort within the statistics and machine learning communities has been directed towards developing likelihood-free, simulation-based inference (SBI) procedures that act as more convenient substitutes to their likelihood-based counterparts. For parameter inference, approaches such as approximate Bayesian computation (ABC) (Pritchard et al., 1999; Beaumont et al., 2002; Dyer et al., 2021a) have seen significant success, while more modern neural network approaches to density (Papamakarios & Murray, 2016; Lueckmann et al., 2017; Greenberg et al., 2019) and density ratio (Thomas et al., 2021; Hermans et al., 2020) estimation show promise as a means to dramatically reduce the simulation burden in SBI procedures; see Dyer et al. (2022a) for a recent overview of these methods and their application to ABMs in the social sciences. Similarly, variants of the Kalman filter and sequential Monte Carlo methods have been developed and applied to the problem of ABM latent state inference (see e.g. Ward et al., 2016; Lux, 2018), although this has received less attention within the ABM community than the problem of parameter inference.

In contrast to this formulation, the increasing availability of granular, longitudinal microdata on agent behaviours and interactions in real-world social systems (e.g. on social media) raises the possibility of dispensing with the structure described above, in which it is often necessary to perform two inference tasks to properly "fit" the ABM. Indeed, in some crucial applications, the ABM is "fully-observed", i.e. $g(\cdot) = identity(\cdot)$ and $\mathbf{x}^t = (\mathbf{w}^t, \mathbf{z}^t)$. Under these circumstances, the filtering and smoothing problems vanish and only the problem of parameter inference remains. In such cases, the process of fully calibrating the ABM to observed data is greatly simplified as a result of the granularity of the data.

Approaches to performing parameter inference for fully-observed ABMs are currently lacking in the SBI literature. Importantly, modellers are lacking approaches to SBI that incorporate useful inductive biases that reflect the natural dynamic graph structure of the ABM and the data. The absence of such methods prevents us from properly capitalising on the availability, and full information content, of such granular data.

In this paper, we address this gap by demonstrating how (recurrent) graph neural networks (GNNs) may be combined with neural SBI procedures to flexibly and automatically accommodate high-dimensional, high-resolution data describing the evolution of the microstate of a social system. We show that GNNs provide useful inductive biases for the use of such microdata as observables against which parameters $\boldsymbol{\theta}$ are calibrated in the case of "fully observed" ABMs, with promising performance on test cases modelling the coevolution of opinions and network structure in a social system.

2. Background & Motivation

2.1. Simulation-based Parameter Inference

Simulation-based inference (SBI) is a set of algorithms in which likelihood-based parameter inference procedures – such as Bayesian inference – are approximated through training on *iid* data $(\mathbf{x}, \boldsymbol{\theta}) \sim p(\mathbf{x} \mid \boldsymbol{\theta})\pi(\boldsymbol{\theta})$, where $\pi(\boldsymbol{\theta})$ is a prior density and $p(\mathbf{x} \mid \boldsymbol{\theta})$ is the likelihood function associated with the simulation model. This is done by first sampling $\boldsymbol{\theta} \sim \pi(\boldsymbol{\theta})$ and subsequently forward simulating from the simulator at $\boldsymbol{\theta}$, represented as $\mathbf{x} \sim p(\mathbf{x} \mid \boldsymbol{\theta})$. Once trained, SBI algorithms then, given some observation $\mathbf{y}$, yield estimates of parameter posteriors $\pi(\boldsymbol{\theta} \mid \mathbf{y})$ (Papamakarios & Murray, 2016; Lueckmann et al., 2017; Greenberg et al., 2019), data likelihood functions $p(\mathbf{y} \mid \boldsymbol{\theta})$ (Papamakarios et al., 2019), or likelihood-to-evidence ratios $p(\mathbf{y} \mid \boldsymbol{\theta})/p(\mathbf{y})$ (Thomas et al., 2021; Hermans et al., 2020; Dyer et al., 2022b).

Of particular interest to the current work is the first of these three alternatives, commonly referred to as neural posterior estimation (NPE). In NPE algorithms, a conditional density estimator – such as a mixture density network (Bishop, 1994) or a normalising flow (Tabak & Vanden-Eijnden, 2010; Tabak & Turner, 2013; Rezende & Mohamed, 2015) – is trained to approximate the map $\mathbf{x} \mapsto \pi(\cdot \mid \mathbf{x})$ using *iid* training data $(\mathbf{x}, \boldsymbol{\theta}) \sim p(\mathbf{x} \mid \boldsymbol{\theta})\pi(\boldsymbol{\theta})$. This provides the experimenter with immediate access to the posterior density estimated by the neural network, which can furthermore be constructed to operate directly on raw data $\mathbf{x}$ through the incorporation of appropriate inductive biases into the network architecture.

2.2. Graph Neural Networks

Recent years have seen considerable progress in the development of graph neural networks (GNNs) in machine learning (e.g. Yao et al., 2019; Zhang et al., 2020; Baek et al., 2021). In many cases, the design of a GNN consists of generalising a convolution operator from regular, Euclidean domains – as appears in convolutional neural networks – to graphs. This has predominantly proceeded by constructing a convolution in the spatial domain (see e.g. Masci et al., 2015; Niepert et al., 2016) or by exploiting the convolution theorem and performing a multiplication in the graph Fourier domain (see e.g. Bruna et al., 2014). A recent review of GNNs and their design can be found in Zhou et al. (2020).

The problem of extending GNNs to dynamic graphs has also recently received significant attention. In this vein, Li et al. (2017) introduce Diffusion Convolutional Recurrent Neural Networks, with applications to traffic flow prediction. In addition, Seo et al. (2018) propose Graph Convolutional

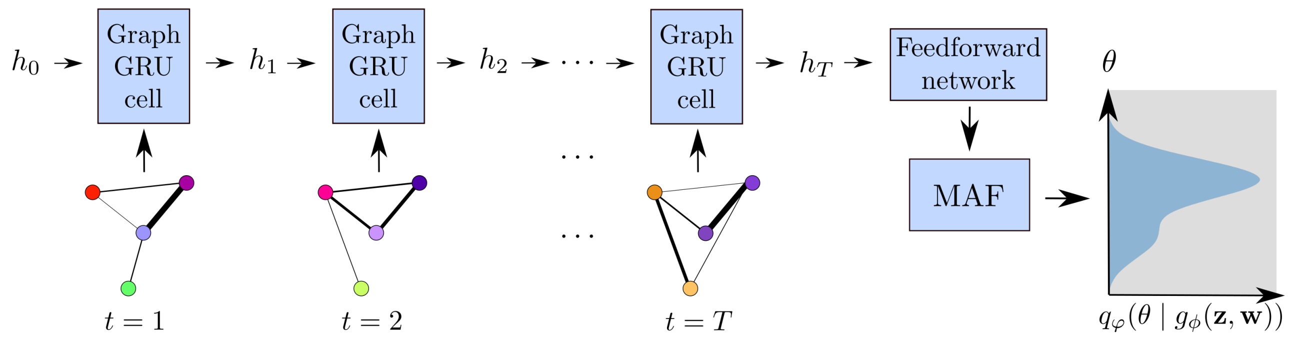

Figure 1. A schematic of our posterior estimation pipeline. The ABM – shown as the dynamic graph with evolving node states (node colors) $\mathbf{z}$ and edge weights (line widths) $\mathbf{w}$ – is embedded into a low-dimensional space with a graph GRU and a feedforward network applied to the GRU's final hidden state, h_T. This representation, $g_\phi(\mathbf{z}, \mathbf{w})$, is fed to the MAF to estimate the posterior as $q_\varphi(\boldsymbol{\theta} \mid g_\phi(\mathbf{z}, \mathbf{w}))$.

Recurrent Networks, an adaptation of standard recurrent networks to operate on sequences of graphs via graph convolutional operators. Further examples of recurrent graph neural network architectures exist; a broader survey of neural networks for dynamic graphs can be found in Wu et al. (2021, Section 7).

3. Methods

In this paper, we consider the problem of parameter inference for the case of fully observed ABMs, where the data the experimenter observes is the complete trace of agent states $\mathbf{z}^t$ and their relationships $\mathbf{w}^t$ over all timesteps $t = 0, \dots, T$. Under these circumstances, the experimenter requires approaches to parameter inference that accommodate the dynamic graph structure of this data.

To address this, we construct a neural posterior estimator in which a recurrent GNN g_ϕ and a neural conditional density estimator q_φ are paired to approximate the map $(\mathbf{z}, \mathbf{w}) \mapsto \pi(\cdot \mid \mathbf{z}, \mathbf{w})$, where ϕ and φ are the parameters of the respective neural networks. In particular, we take g_ϕ to be a feedforward network applied to the final hidden state of the recurrent GNN and q_φ to be a normalising flow. The posterior may then be learned from this low-dimensional representation of the original high-dimensional sequence of graphs as $q_\varphi(\cdot \mid g_\phi(\mathbf{z}, \mathbf{w}))$.

While many choices of architecture are available, we apply a feedforward network to the final hidden state of a graph convolutional gated recurrent unit (GRU) (Seo et al., 2018) to construct g_ϕ and use a masked autoregressive flow (MAF) (Papamakarios et al., 2017) with 5 transforms of 50 hidden features for q_φ in this paper. In Figure 1, we show a schematic of the pipeline that results from the particular choice of recurrent GNN and conditional density estimator used for our experiments, although the exact modules appearing in this experimental setup may be substituted for others without fundamentally altering the overall pipeline.

We train the network parameters ϕ and φ concurrently and on the same loss function as the MAF, such that the parameters of the graph sequence embedding and the posterior estimator are learned simultaneously. Further details on the architectures and training procedure we employ can be found in the supplement.

4. Experimental Results

To test the approach, we consider a task based on the Hopfield model of social dynamics proposed by Macy et al. (2003) which describes the coevolution of opinions and the social network structure, and the emergence of polarisation, in a population of N agents. At each time step $t = 1, \dots, T$, each agent is equipped with $N - 1$ undirected ties to the remaining agents in the population, and the strength and valence of the tie between agents i and j is characterised by $w_{ij}^t \in [-1, 1]$. Each agent is also equipped with a state vector $\mathbf{z}_i^t \in \{-1, 1\}^K, i = 1, \dots, N$, which may represent the opinion status of agent i on each of a number $K \geq 1$ of topics at time t. The *social pressure* that agent i experiences on topic k at time t is then modelled as

$$P_{ik}^t = \frac{1}{N-1} \sum_{j \neq i} w_{ij}^t z_{ik}^t, \tag{1}$$

and i's corresponding propensity to adopt the positive opinion is taken to be

$$\pi_{ik}^t = \frac{1}{1 + e^{-\rho \cdot P_{ik}^t}}, \tag{2}$$

where $\rho > 0$ is a free parameter of the model. Agent i then adopts the positive opinion on topic k at time t (i.e. $z_{ik}^{t+1} = 1$) if

$$\pi_{ik}^t > 0.5 + \epsilon U_i^t, \tag{3}$$

where $\epsilon \in [0, 1]$ is a further free parameter of the model and $U_i^t \sim \mathcal{U}(-0.5, 0.5)$. Finally, the ties between agents evolve

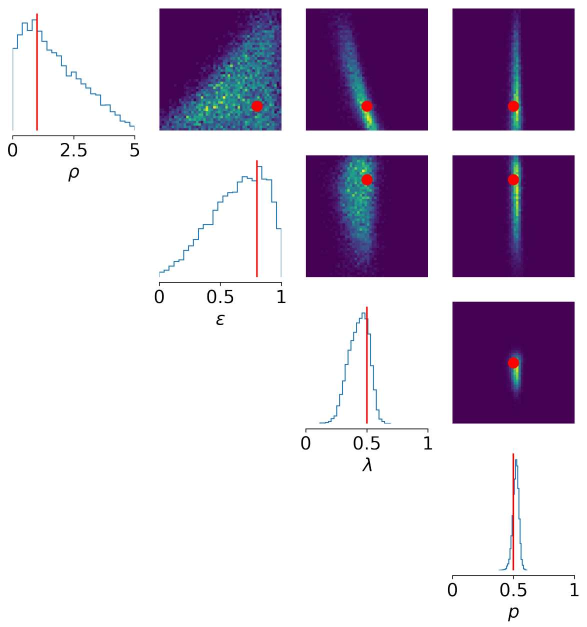

Figure 2. Approximate posterior for the Hopfield model obtained with a masked autoregressive flow and recurrent graph embedding network. Red lines/points show the ground truth parameters for the dataset.

as

$$w_{ij}^{t+1} = (1-\lambda)w_{ij}^{t} + \frac{\lambda}{K}\sum_{k=1}^{K} z_{ik}^{t+1} z_{jk}^{t+1}, \tag{4}$$

where $\lambda \in [0,1]$ is a third free parameter of the model.

Taking the initial proportion $p \in [0,1]$ of all opinion state entries as the final free parameter of the model, we assume the goal of approximating a posterior density for $\boldsymbol{\theta} = (\rho, \epsilon, \lambda, p)$. Specifically, we assume prior densities $\rho \sim \mathcal{U}(0,5)$, $\epsilon \sim \mathcal{U}(0,1)$, $\lambda \sim \mathcal{U}(0,1)$, $p \sim \mathcal{U}(0,1)$, and further assume that the ABM is fully observed in the sense that all w_{ij}^t and z_{ik}^t are observed. To generate a pseudo-true dataset, we simulate the model for 25 time steps with $N = 20$ at ground-truth parameter values $\boldsymbol{\theta}^* = (1, 0.8, 0.5, 0.5)$.

We show the approximate posterior obtained with a MAF posterior estimator, recurrent graph convolutional embedding network, and a budget of 1000 simulations in Figure 2. The diagonal subplots show the marginal posterior densities, while the off-diagonal subplots show the 2-dimensional projections of the joint densities. We show the ground truth parameters $\boldsymbol{\theta}^*$ with red lines and points. The approximated posterior assigns high posterior density to the ground truth parameters, providing evidence that a reasonable degree of accuracy has been achieved by the posterior estimator.

5. Discussion

In this paper, we address the problem of how to learn parameter posteriors for "fully-observed" ABMs, that is, when the full trace of the agents' states and interactions are observed. We propose the use of temporal graph neural networks in neural SBI algorithms as a way to incorporate useful inductive biases reflecting the natural dynamic graph structure of ABMs. Through experiments performed on an ABM modelling the coevolution of agent opinions and relationship strengths in a dynamic social network, we demonstrated that such an approach can generate approximate Bayesian parameter posteriors in which the ground-truth parameter is assigned a high-posterior density, suggesting that the approximate posterior is accurate to some degree. In future work, we will conduct a more thorough assessment of the quality of the estimated posteriors following the guidelines discussed in Dyer et al. (2022a), for example through the use of posterior predictive checks to assess the predictive power of the inferences drawn, or simulation-based calibration (Talts et al., 2020) to assess the quality of the uncertainty quantification provided by the overall shape of the posterior. In addition, we will consider extensions of this approach to more general settings, for example continuous-time settings through the use of architectures that are compatible with continuous-time data (see e.g. Rossi et al., 2020).

Acknowledgements

The authors thank the anonymous reviewers for their helpful comments and feedback. JD is supported by the EPSRC Centre For Doctoral Training in Industrially Focused Mathematical Modelling (EP/L015803/1) in collaboration with Improbable.

References

Baek, J., Kang, M., and Hwang, S. J. Accurate Learning of Graph Representations with Graph Multiset Pooling. In *ICLR*, 2021. URL `https://openreview.net/forum?id=JHcqXGaqiGn`.

Baptista, R., Farmer, J. D., Hinterschweiger, M., Low, K., Tang, D., and Uluc, A. Macroprudential policy in an agent-based model of the UK housing market. *Bank of England Working Paper*, 2016.

Beaumont, M. A., Zhang, W., and Balding, D. J. Approximate Bayesian computation in population genetics. *Genetics*, 162(4):2025–2035, 2002.

Bishop, C. M. Mixture density networks. 1994.

Bruna, J., Zaremba, W., Szlam, A., and LeCun, Y. Spectral networks and deep locally connected networks on graphs. In *2nd International Conference on Learning Representations, ICLR 2014*, 2014.

Dyer, J., Cannon, P., and Schmon, S. M. Approximate Bayesian Computation with Path Signatures. *arXiv preprint arXiv:2106.12555*, 2021a.

Dyer, J., Cannon, P. W., and Schmon, S. M. Deep signature statistics for likelihood-free time-series models. In *ICML Workshop on Invertible Neural Networks, Normalizing Flows, and Explicit Likelihood Models*, 2021b.

Dyer, J., Cannon, P., Farmer, J. D., and Schmon, S. Black-box Bayesian inference for economic agent-based models. *arXiv preprint arXiv:2202.00625*, 2022a.

Dyer, J., Cannon, P. W., and Schmon, S. M. Amortised Likelihood-free Inference for Expensive Time-series Simulators with Signatured Ratio Estimation. In *International Conference on Artificial Intelligence and Statistics*, pp. 11131–11144. PMLR, 2022b.

Ferguson, N. M., Laydon, D., Nedjati-Gilani, G., Imai, N., Ainslie, K., Baguelin, M., Bhatia, S., Boonyasiri, A., Cucunubá, Z., Cuomo-Dannenburg, G., et al. Impact of non-pharmaceutical interventions (NPIs) to reduce COVID-19 mortality and healthcare demand, 2020.

Greenberg, D. S., Nonnenmacher, M., and Macke, J. H. Automatic posterior transformation for likelihood-free inference. *36th International Conference on Machine Learning, ICML 2019*, 2019-June:4288–4304, 2019.

Hermans, J., Begy, V., and Louppe, G. Likelihood-free MCMC with Amortized Approximate Ratio Estimators. In III, H. D. and Singh, A. (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 4239–4248. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/hermans20a.html.

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Li, Y., Yu, R., Shahabi, C., and Liu, Y. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. *arXiv preprint arXiv:1707.01926*, 2017.

Lueckmann, J.-M., Goncalves, P. J., Bassetto, G., Öcal, K., Nonnenmacher, M., and Macke, J. H. Flexible statistical inference for mechanistic models of neural dynamics. In *Advances in Neural Information Processing Systems*, 2017.

Lux, T. Estimation of agent-based models using sequential Monte Carlo methods. *Journal of Economic Dynamics and Control*, 91:391–408, 2018. ISSN 0165-1889. doi: https://doi.org/10.1016/j.jedc.2018.01.021. URL https://www.sciencedirect.com/science/article/pii/S0165188918300356. Special Issue in Honour of Prof. Carl Chiarella.

Macy, M. W., Kitts, J. A., Flache, A., and Benard, S. Polarization in dynamic networks: A Hopfield model of emergent structure. *Dynamic Social Network Modelling and Analysis*, pp. 162–173, 2003.

Masci, J., Boscaini, D., Bronstein, M. M., and Vandergheynst, P. Geodesic Convolutional Neural Networks on Riemannian Manifolds. In *2015 IEEE International Conference on Computer Vision Workshop (ICCVW)*, pp. 832–840, 2015. doi: 10.1109/ICCVW.2015.112.

Niepert, M., Ahmed, M., and Kutzkov, K. Learning convolutional neural networks for graphs. In *International conference on machine learning*, pp. 2014–2023. PMLR, 2016.

Papamakarios, G. and Murray, I. Fast epsilon-free Inference of Simulation Models with Bayesian Conditional Density Estimation. In *Advances in Neural Information Processing Systems*, 2016.

Papamakarios, G., Pavlakou, T., and Murray, I. Masked autoregressive flow for density estimation. *Advances in neural information processing systems*, 30, 2017.

Papamakarios, G., Sterratt, D., and Murray, I. Sequential neural likelihood: Fast likelihood-free inference with autoregressive flows. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 837–848. PMLR, 2019.

Pritchard, J. K., Seielstad, M. T., Perez-Lezaun, A., and Feldman, M. W. Population growth of human Y chromosomes: a study of Y chromosome microsatellites. *Molecular biology and evolution*, 16(12):1791–1798, 1999.

Rezende, D. and Mohamed, S. Variational Inference with Normalizing Flows. In Bach, F. and Blei, D. (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1530–1538, Lille, France, 07–09 Jul 2015. PMLR. URL https://proceedings.mlr.press/v37/rezende15.html.

Rossi, E., Chamberlain, B., Frasca, F., Eynard, D., Monti, F., and Bronstein, M. Temporal graph networks for deep learning on dynamic graphs. *arXiv preprint arXiv:2006.10637*, 2020.

Rozemberczki, B., Scherer, P., He, Y., Panagopoulos, G., Riedel, A., Astefanoaei, M., Kiss, O., Beres, F., , Lopez, G., Collignon, N., and Sarkar, R. PyTorch Geometric Temporal: Spatiotemporal Signal Processing with Neural Machine Learning Models. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management*, pp. 4564–4573, 2021.

Seo, Y., Defferrard, M., Vandergheynst, P., and Bresson, X. Structured sequence modeling with graph convolutional recurrent networks. In *International Conference on Neural Information Processing*, pp. 362–373. Springer, 2018.

Tabak, E. G. and Turner, C. V. A family of nonparametric density estimation algorithms. *Communications on Pure and Applied Mathematics*, 66(2):145–164, 2013.

Tabak, E. G. and Vanden-Eijnden, E. Density estimation by dual ascent of the log-likelihood. *Communications in Mathematical Sciences*, 8(1):217–233, 2010.

Talts, S., Betancourt, M., Simpson, D., Vehtari, A., and Gelman, A. Validating Bayesian Inference Algorithms with Simulation-Based Calibration, 2020.

Tejero-Cantero, A., Boelts, J., Deistler, M., Lueckmann, J.-M., Durkan, C., Gonçalves, P. J., Greenberg, D. S., and Macke, J. H. sbi: A toolkit for simulation-based inference. *Journal of Open Source Software*, 5(52):2505, 2020. doi: 10.21105/joss.02505. URL https://doi.org/10.21105/joss.02505.

Thomas, O., Dutta, R., Corander, J., Kaski, S., and Gutmann, M. U. Likelihood-Free Inference by Ratio Estimation. *Bayesian Analysis*, pp. 1 – 31, 2021. doi: 10.1214/20-BA1238. URL https://doi.org/10.1214/20-BA1238.

Ward, J. A., Evans, A. J., and Malleson, N. S. Dynamic calibration of agent-based models using data assimilation. *Royal Society open science*, 3(4):150703, 2016.

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., and Yu, P. S. A Comprehensive Survey on Graph Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems*, 32(1):4–24, 2021. doi: 10.1109/TNNLS.2020.2978386.

Yao, L., Mao, C., and Luo, Y. Graph convolutional networks for text classification. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pp. 7370–7377, 2019.

Zhang, Y., Chen, X., Yang, Y., Ramamurthy, A., Li, B., Qi, Y., and Song, L. Efficient Probabilistic Logic Reasoning with Graph Neural Networks. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rJg76kStwH.

Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., and Sun, M. Graph neural networks: A review of methods and applications. *AI Open*, 1:57–81, 2020. ISSN 2666-6510. doi: https://doi.org/10.1016/j.aiopen.2021.01.001. URL https://www.sciencedirect.com/science/article/pii/S2666651021000012.

A. Further experimental details

The first module in the embedding network g_ϕ is a graph convolutional gated recurrent unit proposed in (Seo et al., 2018). Taking $L \in \mathbb{R}^{N \times N}$ as the normalised graph Laplacian for the order-N graph, this component operates on the sequence $(\mathbf{x}^t)_{t=1}^T$ of node states $\mathbf{x}_t \in \mathbb{R}^{N \times K}$ – where $K \geq 1$ is the dimensionality of each node state – to find a running embedding $\mathbf{h}^t \in \mathbb{R}^{N \times d_\mathbf{h}}$ of each subsequence $(\mathbf{x}_{t'})_{t'=1}^t$, $t = 1, \ldots, T$, as follows:

$$\begin{aligned}
\mathbf{s} &= \sigma\left(W_{\mathbf{sx}}(L) \cdot \mathbf{x}^t + W_{\mathbf{sh}}(L) \cdot \mathbf{h}^{t-1}\right), \\
\mathbf{r} &= \sigma\left(W_{\mathbf{rx}}(L) \cdot \mathbf{x}^t + W_{\mathbf{rh}}(L) \cdot \mathbf{h}^{t-1}\right), \\
\tilde{\mathbf{h}} &= \tanh\left(W_{\mathbf{hx}}(L) \cdot \mathbf{x}^t + W_{\mathbf{hh}}(L) \cdot (\mathbf{r} \odot \mathbf{h}^{t-1})\right), \\
\mathbf{h}^t &= \mathbf{s} \odot \mathbf{h}^{t-1} + (1 - \mathbf{s}) \odot \tilde{\mathbf{h}}
\end{aligned}$$

for $t = 1, \ldots, T$. Here, $W_{\cdot\mathbf{x}} : \mathbb{R}^{N \times N} \to \mathbb{R}^{Q \times d_\mathbf{h} \times K}$ and $W_{\cdot\mathbf{h}} : \mathbb{R}^{N \times N} \to \mathbb{R}^{Q \times d_\mathbf{h} \times d_\mathbf{h}}$ are graph convolution operators with $d_\mathbf{h} K$ filters that are parameterised by Q Chebyshev coefficients, taking the form

$$G(L) = \sum_{q=1}^{Q} a_q H_q(L), \tag{5}$$

where $H_q(L)$ is the q-th order Chebyshev polynomial evaluated at L. We use $Q = 3$ Chebyshev coefficients in the graph filtering operation and choose a hidden state size of $d_\mathbf{h} = 64$, such that the hidden state of each agent is a 64-dimensional vector. A single linear layer reduces this $N \times 64$ matrix into an N-vector, where N is the number of agents in the system. An embedding of the entire graph then proceeds by passing this N-vector through a feedforward network with layer sizes $32, 16, 16$. In our experiments, we take $N = 20$ and simulate for $T = 25$ time steps.

To construct the posterior estimator, we use a masked autoregressive flow (Papamakarios et al., 2017) with 5 transforms and 50 hidden features.

To train the neural networks described above, we follow e.g. Lueckmann et al. (2017); Greenberg et al. (2019); and Dyer et al. (2021b) and train the parameters for the embedding network and the posterior estimator concurrently on the same loss function (the sample mean of the the negative log-likelihood of the parameters). We use a learning rate of 5×10^{-4} and a training batch size of 50. Furthermore, we reserve 10% of the training data for validation and cease training if the validation loss fails to decrease after 20 epochs to prevent overfitting. Throughout, we make use of the `sbi` (Tejero-Cantero et al., 2020) and PyTorch Geometric Temporal (Rozemberczki et al., 2021) `python` packages.