Supplement to "Machine Learning for Mechanistic Models of Metapopulation Dynamics"

Jifan Li, Edward L. Ionides, Aaron A. King, Mercedes Pascual and Ning Ning

Compiled November 9, 2023, using R 4.2.3, `metapoppkg` 0.1.15, `spatPomp` 0.33.0, and `pomp` 5.4.2.0.

Supplementary Content

S1 Specification of the models

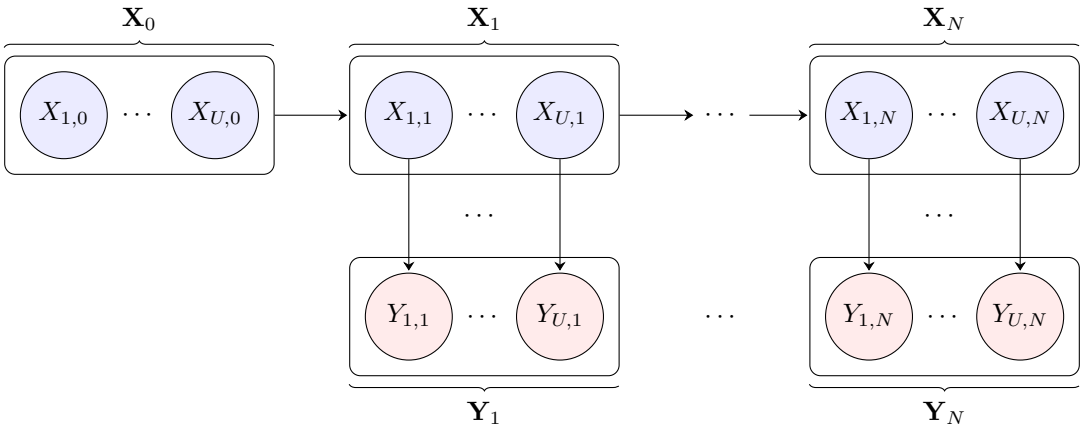

Figure S-1: Diagram for a spatiotemporal partially observed Markov process (SpatPOMP) model, adapted from (Asfaw et al., 2023). The latent dynamic process is a continuous-time Markov chain taking value $\mathbf{X}(t) = \big(X_0(t), \dots, X_U(t)\big)$ at time t. At observation time t_n, the value of the latent process is denoted by $\mathbf{X}(t_n) = \mathbf{X}_n = \big(X_{1,n}, \dots, X_{U,n}\big)$. Noisy and/or incomplete observations of the latent process at this time are modeled by $\mathbf{Y}_n = \big(Y_{1,n}, \dots, Y_{U,n}\big)$.

We give the mathematical description of the COVID-19 models described in Figure 1 and Table 1 of the main text. These metapopulation models are partially observed Markov process (POMP) models with additional spatial structure, known as SpatPOMP models (Asfaw et al., 2023). The diagram in Figure S-1 defines a general SpatPOMP corresponding to a metapopulation with U spatial units. The latent dynamic state is $\mathbf{X}(t) = \big(X_1(t), \dots, X_U(t)\big)$, which we also write as $\mathbf{X}(t) = X_{1:U}(t)$. The observation model at time t_n is $\mathbf{Y}_n = \big(Y_{1,n}, \dots, Y_{U,n}\big) = Y_{1:U,n}$, for n taking values in $1:N$. The data, $\mathbf{y}^*_n = y^*_{1:U,n}$, are modeled as a realization of the random variable $Y_{1:U,n}$.

For each city, u in $1{:}U$, with $U = 373$, we model the state at time t as

$$X_u(t) = \big(S_u(t), E_u(t), A_u(t), I_u(t), R_u(t), C^a_u t, C^b_u(t), C_u(t)\big), \tag{S1}$$

where each individual in the city is in exactly one of the compartments: susceptible (S_u), exposed (E_u), infected and infectious but asymptomatic (A_u), infected and infectious and symptomatic (I_u), and recovered or removed (R_u). The additional case reporting compartments, C^a_u, C^b_u and C_u are used to describe reporting delay. Individuals entering I_u are simultaneously added to C^a_u, from which they transition to C^b_u and subsequently to the observable compartment, C_u. For notational convenience, we introduce a transport compartment, T, which accounts for all individuals traveling between cities. The complete collection of compartments is therefore

$$\mathbb{C} = \big\{S_{1:U}, E_{1:U}, A_{1:U}, I_{1:U}, R_{1:U}, C^a_{1:U}, C^b_{1:U}, C_{1:U}, T\big\}.$$

We let $N_{VW}(t)$ count the directional transitions between V to W for any pair of compartments in $\mathbb{C}$, and we write dN_{VW} for an infinitesimal increment, $N_{VW}(t+dt) - N_{VW}(t)$. We can write $\mathbf{X}(t) = \big(X_1(t), \dots, X_U(t)\big)$

in terms of its value $\mathbf{X}(t_0)$ at an initial time t_0 together with the flow equations:

$$
\begin{aligned}
dS_u &= -dN_{S_uE_u} + dN_{TS_u} - dN_{S_uT}, && \text{(S2)}\\
dE_u &= dN_{S_uE_u} - dN_{E_uA_u} - dN_{E_uI_u} + dN_{TE_u} - dN_{E_uT}, && \text{(S3)}\\
dA_u &= dN_{E_uA_u} + dN_{TA_u} - dN_{A_uT} - dN_{A_uR_u}, && \text{(S4)}\\
dI_u &= dN_{E_uI_u} - dN_{I_uR_u}, && \text{(S5)}\\
dR_u &= dN_{I_uR_u} + dN_{A_uR_u}, && \text{(S6)}\\
dC_u^a &= dN_{E_uI_u} - dN_{C_u^aC_u^b}, && \text{(S7)}\\
dC_u^b &= dN_{C_u^aC_u^b} - dN_{C_u^bC_u}, && \text{(S8)}\\
dC_u &= dN_{C_u^bC_u}. && \text{(S9)}
\end{aligned}
$$

In this model, individuals travel between cities only when in compartments S (susceptible), A (infected and infectious but unreported, generally asymptomatic or mildly symptomatic), and E (exposed with a latent infection). Also, note that we do not match up each individual entering and leaving T, so there can be small stochastic variation in the total population.

Each transition dN_{VW} has an associated rate, μ_{VW}, which may depend on the state of other compartments, or on covariate processes, on parameters, or on time. For all compartments other than the source/sink compartment, T, it is convenient to specify rates per capita. For transitions which enter a compartment from T, we specify a total rate. The non-zero transition rates are therefore as follows:

$$
\begin{aligned}
\mu_{S_uE_u} &= \beta S_u(t)\left(\frac{I_u(t)+\mu A_u(t)}{P_u(t)}\right) d\Gamma_u/dt, && \text{(S10)}\\
\mu_{S_uT} = \mu_{E_uT} = \mu_{A_uT} &= \vartheta \sum_j \frac{M_{uj}}{P_u - I_u}, && \text{(S11)}\\
\mu_{TS_u} &= \vartheta \sum_j \frac{M_{ju}S_j}{P_j - I_j}, && \text{(S12)}\\
\mu_{TE_u} &= \vartheta \sum_j \frac{M_{ju}E_j}{P_j - I_j}, && \text{(S13)}\\
\mu_{TA_u} &= \vartheta \sum_j \frac{M_{ju}A_j}{P_j - I_j}, && \text{(S14)}\\
\mu_{E_uA_u} = \mu_{E_uI_u} &= \alpha/Z, && \text{(S15)}\\
\mu_{A_uR_u} = \mu_{I_uR_u} &= 1/D, && \text{(S16)}\\
\mu_{C_u^aC_u^b} = \mu_{C_u^bC_u} &= 2T_d. && \text{(S17)}
\end{aligned}
$$

We define $d\Gamma_u/dt$ in (S10) as non-negative multiplicative gamma white noise with variance parameter $\sigma_{SE,u}$. That is, $\Gamma_u(t)$ is a gamma process with stationary independent increments, such that $\Gamma_u(t)-\Gamma_u(s)$ is gamma distributed with mean $t-s$ and variance $\sigma_{SE}(t-s)$. Equations (S2)-(S17) specify an overdispersed continuous time Markov process via the limit of a discrete time Euler approximation as the discretization step approaches zero (Bretó et al., 2009; Bretó and Ionides, 2011). The parameters are described in Table S-1.

Data for city u at time t_n is an official report $y^*_{u,n}$ recording new cases since time t_{n-1}. The data are modeled as a realization of a random variable $Y_{u,n}$ which measures $C_u(t_n) - C_u(t_{n-1})$. The measurement model asserts that $Y_{u,n}$ is a discretized normal random variable with mean

$$C_{u,n} = C_u(t_n) - C_u(t_{n-1}), \qquad \text{(S18)}$$

and variance

$$V_{u,n} = C_{u,n} + \tau^2 C_{u,n}^2, \qquad \text{(S19)}$$

including both Poisson scale variability and the possibility of overdispersion. Thus,

$$\text{Prob}\big(Y_{u,n} = y_{u,n} \,|\, C_{u,n}\big) = \Phi\big(y_{u,n}+0.5\,;C_{u,n},V_{u,n}\big) - \Phi\big(y_{u,n}-0.5\,;C_{u,n},V_{u,n}\big),$$

where Φ is the normal cumulative distribution function. If $y_{u,n}=0$, we replace $\Phi\big(y_{u,n}-0.5\,;C_{u,n},V_{u,n}\big)$ by $\Phi\big(-\infty\,;C_{u,n},V_{u,n}\big)=0$.

Our models M_1, M_2, M_5 and M_6 are extensions of the model of Li et al. (2020). The original model of Li et al. (2020), which we call `li20`, represents the dynamic model by a system of ordinary differential equations with random rates, as discussed further in Section S1.3. The general model including M_1, M_2, M_5 and M_6, which we call `li23`, represents the models as continuous-time Markov chains. In addition to this change, `li23` considers two model aspects not investigated by Li et al. (2020): (i) overdispersed process noise; (ii) an adjusted movement matrix to ensure that all cities are connected. In M_1, these two features are turned off, and so this model is very similar to `li20`, as documented subsequently in Section S1.3 (and specifically Figure S-7) where we discuss `li20` in more detail. M_1 and M_2 have the constraint that $\sigma_{SE}=0$. M_1 has an additional constraint that $\tau=0$ since this parameter was not included in the dynamic model of Li et al. (2020). Another difference between these models is that M_1 uses the mobility data from (Li et al., 2020) whereas M_2 (together with M_5 and M_6) use the modification described in Section S1.2. M_5 involves estimation of all the parameters estimated by Li et al. (2020), with the addition of σ_{SE} and τ. M_6 adds the additional constraints that $Z^{\mathrm{af}}=Z^{\mathrm{be}}$ and $D^{\mathrm{af}}=D^{\mathrm{be}}$.

All the model parameters were specified to be shared between units. The model can be extended to define distinct unit-specific values for each parameter, and in some situations this is helpful (Ionides et al., 2022; Whitehouse et al., 2023). We developed an R package `metapoppkg` to provide a data analysis environment for our numerical work, described further in Section S8. The `li23` and `li20` models are implemented by the `li23()` and `li20()` constructor functions in `metapoppkg`. The resulting model objects have class `spatPomp` which provides access to the inference and visualization tools in the R package `spatPomp` (Asfaw et al., 2023) as well as `pomp` (King et al., 2016). Here, we focus on likelihood evaluation via the block particle filter, implemented as `bpfilter`, and parameter estimation via the iterated block particle filter, `ibpf` (discussed in Section S3). The ensemble Kalman filter and iterated ensemble Kalman filter, as employed by (Li et al., 2020), are implemented as `enkf` and `ienkf`, respectively, and are discussed further in Section S4.

Table S-1 shows some substantial differences between our parameter estimates and those of Li et al. (2020). Model M_5, which has considerably higher likelihood than M_1 due to the inclusion of dynamic process noise, obtains its highest likelihoods when the pre-lockdown infectious period parameter is unrealistically large. In this model, a long duration of infection pre-lockdown does not cause problems because individuals leave the infected class quickly once lockdown arrives. Constraining the durations of latency and infection to be the same before and after lockdown changes this, without having substantial effects on other parameter estimates. This occurs at the expense of $-9088.2-(-9116.5)=28.3$ units of log-likelihood. We cannot readily see why the data prefer a mechanistically implausible infectious period pre-lockdown. However, weak identifiability is not surprising in a complex model with many parameters that is required to fit fairly sparse amounts of data pre-lockdown. Some model misspecification is also inevitable for a mathematical model of a biological system. When weak identifiability and model misspecification co-occur, one possible result is scientifically implausible parameter estimates.

Our estimate of the relative transmissibility of unreportable to reportable cases is considerably higher than Li et al. (2020). Indeed, our estimate during the first time period is $\mu^{\mathrm{be}}=1.00$. The data are compared to simulations from models M_1 and M_5 in Figure S-2. The greater number of cases for model M_5 compared to M_1 is explained by its higher initial $\mathcal{R}_0$. The variability in M_5 is explicitly included in the model and fitted to the data; it therefore matches the variability in the data more closely than M_1. If many simulations are made from M_5, the pointwise 10^{th} percentile is similar to a simulation from M_1 (Figure S-3). The similarity between model M_1 and `li20` is evident by comparing Figure S-2(B) with Figure S-7 in Section S1.3. The code and parameters for the simulation from `li20` are taken directly from Li et al. (2020).

	M_1	CI	M_5	CI	M_6	CI	interpretation & units
ϑ	1.36	(1.27,1.45)	2.34	(2.07,2.54)	2.87	(2.67,3.19)	Mobility factor
τ	0.5	fixed	0.28	(0.27,0.33)	0.32	(0.29,0.35)	Measurement noise
σ_{SE}	0	fixed	2.08	(1.93,2.27)	1.77	(1.58,1.92)	Dynamic noise ($\text{day}^{1/2}$)
E_0	0–2000	NA	2712	(2381,3122)	3477	(2730,3846)	Initial E in Wuhan
A_0	0-2000	NA	0	(0,307)	0	(0,440)	Initial A in Wuhan
β^{be}	1.12	(1.06,1.09)	0.73	(0.70,0.78)	0.97	(0.93,1.05)	Transmission rate (day^{-1})
μ^{be}	0.55	(0.46,0.62)	1.00	(0.96,1.00)	1.00	(0.93,1.00)	Relative transmission μ
Z^{be}	3.69	(3.30,3.96)	0.55	(0.28,0.83)	*0.72	(0.48,0.97)	Latent period (day)
D^{be}	3.47	(3.15,3.73)	35.0	(5.0,35.0)	†3.87	(3.69,4.10)	Infectious period (day)
α^{be}	0.14	(0.10,0.18)	0.11	(0.09,0.12)	0.08	(0.07,0.08)	Reported fraction
$\mathcal{R}_0^{\text{be}}$	2.38	(2.03,2.77)	13.07	(11.96,13.72)	3.51	(3.31,3.72)	Basic reproductive number
T_d^{be}	9.00	fixed	9.00		9.00		Diagnosis delay (day)
β^{af}	0.35	(0.28,0.45)	0.24	(0.21,0.26)	0.22	(0.21,0.25)	Transmission rate (day^{-1})
μ^{af}	0.43	(0.31,0.61)	0.61	(0.52,0.82)	0.78	(0.66,0.90)	Relative transmission
Z^{af}	3.42	(3.30,3.65)	4.23	(3.39,5.04)	*0.72	(0.48,0.97)	Latent period (day)
D^{af}	3.31	(2.96,3.88)	2.36	(2.16,2.51)	†3.87	(3.69,4.10)	Infectious period (day)
α^{af}	0.69	(0.65,0.72)	0.38	(0.34,0.47)	0.48	(0.39,0.52)	Reported fraction
$\mathcal{R}_0^{\text{af}}$	0.95	(0.83,1.16)	0.51	(0.49,0.58)	0.70	(0.65,0.77)	Basic reproductive number
T_d^{af}	6.00	fixed	6.00		6.00		Diagnosis delay (day)

Table S-1: Parameter estimates and their confidence intervals (CIs). The parameter estimates and confidence intervals for M_1 come from Li et al. (2020). The values for M_5 and M_6 come from profile likelihood plots shown in Sections S5 and S6 respectively. Top block of rows: parameters constant through time. Middle block: parameters estimated for Jan 10-Jan 23. Bottom block: parameters estimated for Jan 24-Feb 8. Parameters without specified units are dimensionless. For M_6, $Z^{\text{be}} = Z^{\text{af}}$, so the two values marked by * are constrained to be equal; † denotes the other constraint $D^{\text{be}} = D^{\text{af}}$. We calculated $\mathcal{R}_0^{\text{be}} = \big(\alpha^{\text{be}} + (1-\alpha^{\text{be}})\mu^{\text{be}}\big)D^{\text{be}}\beta^{\text{be}}$ and $\mathcal{R}_0^{\text{af}} = \big(\alpha^{\text{af}} + (1-\alpha^{\text{af}})\mu^{\text{af}}\big)D^{\text{af}}\beta^{\text{af}}$.

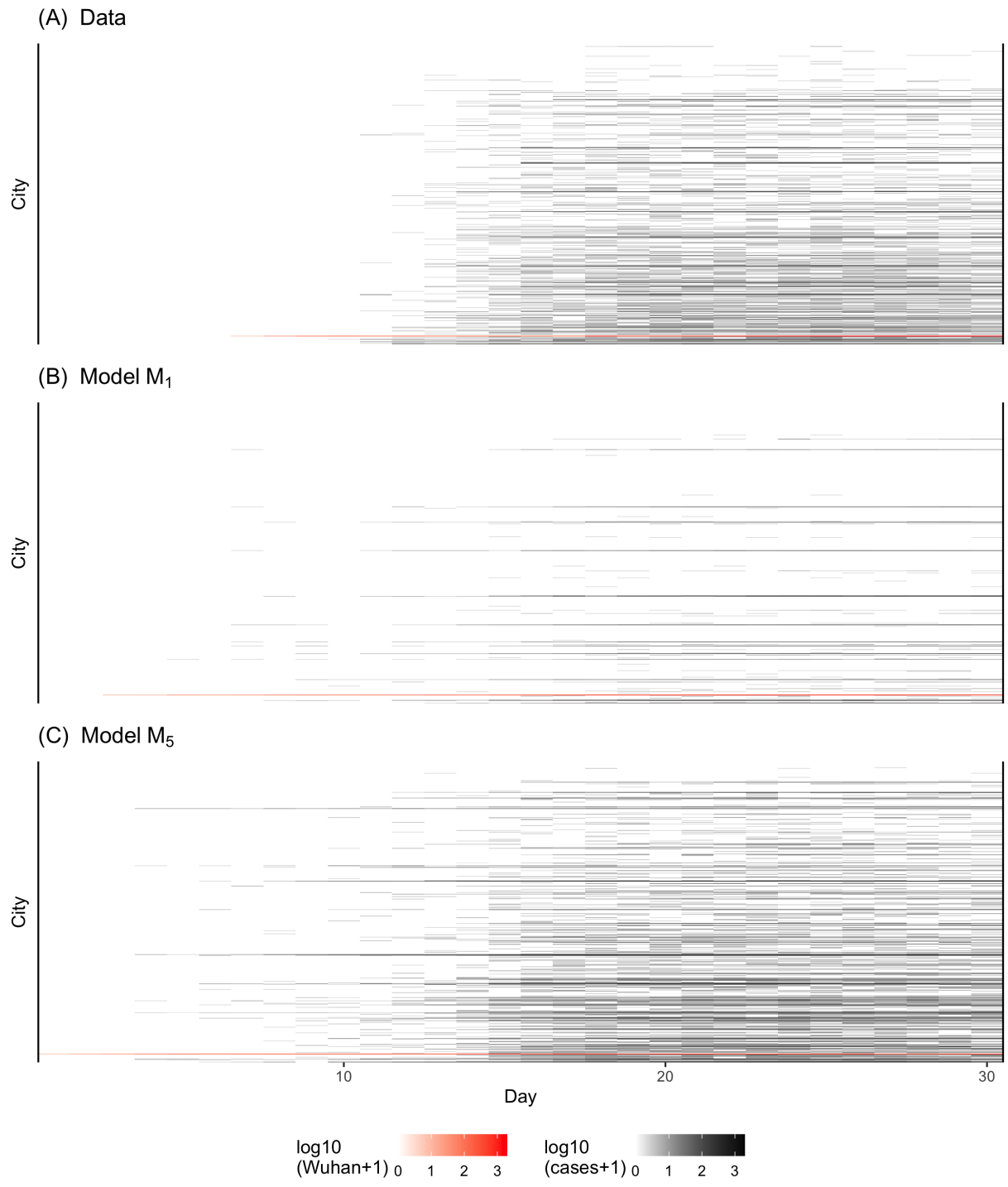

Figure S-2: Daily case report time series for 373 cities: (A) the real data; (B) a simulation from model M_1; (C) a simulation from model M_6. Within each panel, cities are ordered by population, largest on the bottom row.

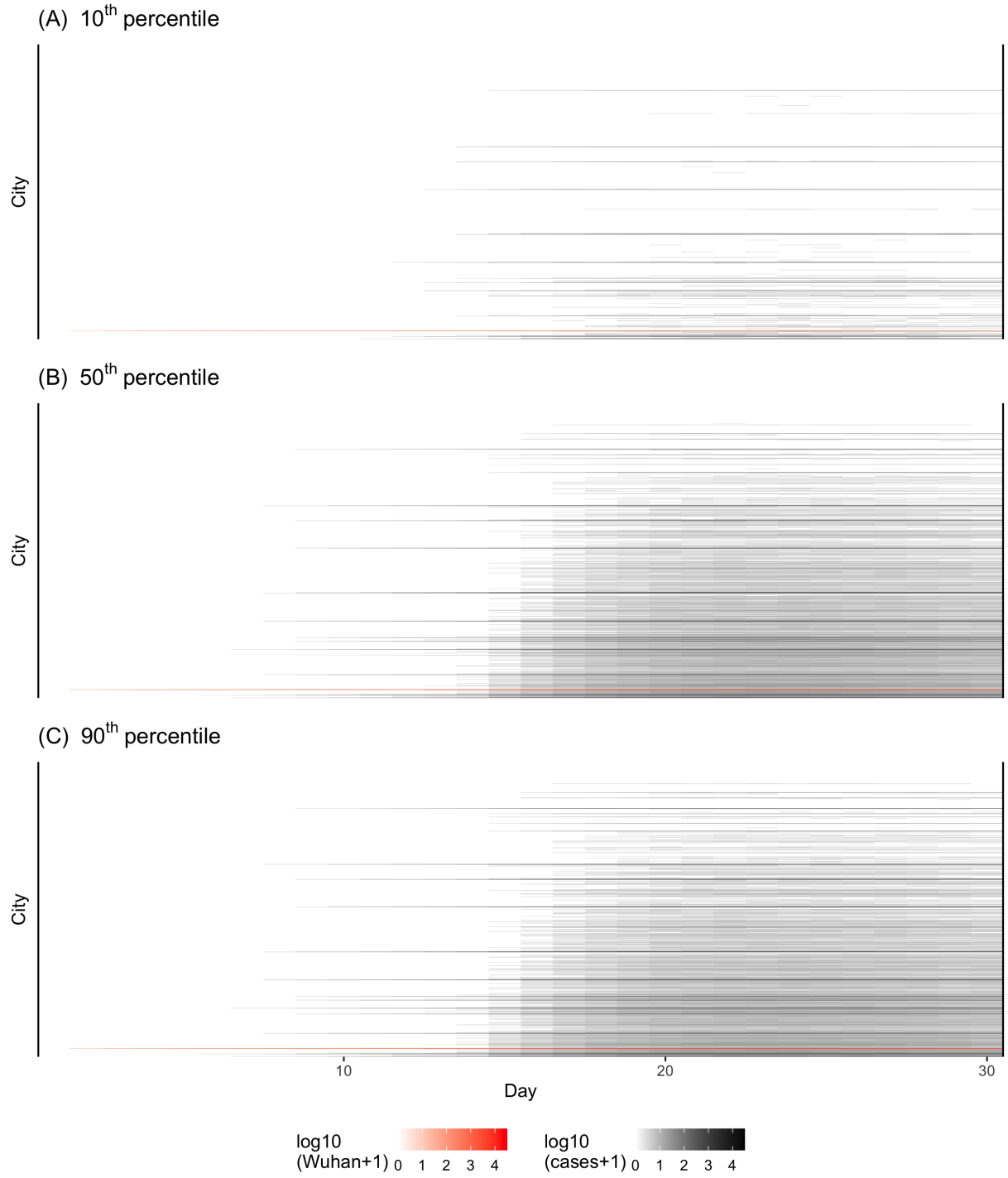

Figure S-3: Simulated daily case reports for model M_5, showing the 10th, 50th and 90th percentiles. Within each panel, cities are ordered by population, largest on the bottom row.

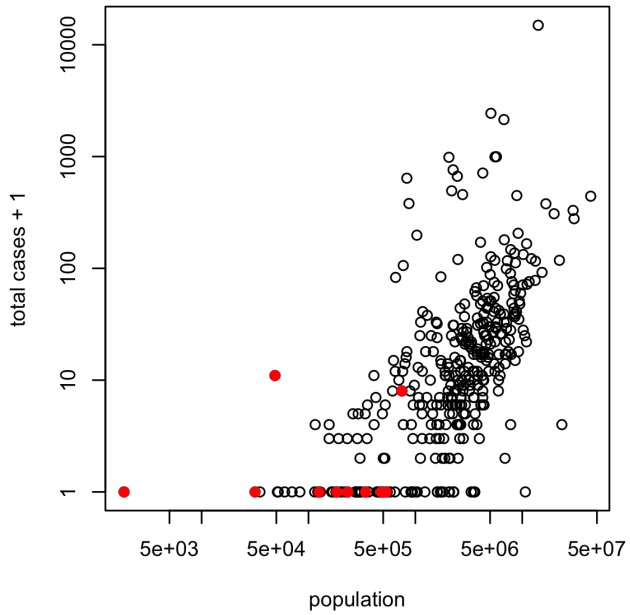

Figure S-4: Total cases, January 10 to February 8, for each city, plotted against mean population size. Cities with no arriving travelers recorded in the mobility data are shown as solid red points.

S1.1 Reporting Delay

The `li23` and `li20` models describe the reporting process via the case report compartments, C_u^a, C_u^b and C_u for each city, u. When an individual transitions from E_u to the reportable infectious state, I_u, an individual is also added to the start of the reporting process by incrementing C_u^a. The counts in compartments C_u^a, C_u^b and C_u do not affect the transmission dynamics; they only part of the measurement model. For this reason, they are denoted by octagons rather than squares in the diagrammatic representation (Figure 1, main text).

Li et al. (2020) described transitions from C_u^a to C_u using a gamma delay model, with each individual arriving in C_u^a transitioning to C_u after a random delay distributed as $G(a, T_d/a)$, the gamma distribution with mean T_d and variance T_d^2/a. Based on analysis of early confirmed cases, and preliminary exploration of the model, they specified $a = 1.85$, $T_d = 9$ before January 23, and $T_d = 6$ after January 23. We use their values of T_d but use $a = 2$ in order to obtain a Markovian representation, where the gamma delay is represented as the sum of two exponential delays, formalized using a compartment C_u^b intermediate between C_u^a and C_u.

Our interpretation of reporting delay leads to a small discrepancy between our `li20` model and the model actually specified by Li et al. (2020). However, the discrepancy is small. Further, the Markovian property is necessary for inference using either the ensemble Kalman filter or block particle filter. Thus, this discrepancy closes a small gap between the model specified by Li et al. (2020) and the methods which they (and we) use to analyze the model.

S1.2 Mobility Data

Figure S-4 shows the 10 cities which have no incoming travelers in the mobility dataset compared to other cities. We see that the cities modeled as having no sources by Li et al. (2020) did have relatively few reported cases for their city size, but not a complete absence of caess.

To capture individual movement among the 373 cities simulated in the metapopulation model, Li et al. (2020) used human mobility data from the Tencent location-based service used in popular Tencent mobile phone applications, such as Wechat, QQ, and Baidu Maps. High resolution Tencent data were available for 2018, so they assumed the travel patterns captured in 2018 during the New Year celebrations (Chunyun) are similar to those of the analogous time period during 2020, prior to January 23 travel restrictions. In total, 92,248 inter-city travel records were used to represent travel during January 10-23. In the Tencent mobility data, for each day, the top 10 outflows from each of 373 Chinese cities were recorded. For city-to-city connections for which only some of the days in this two-week time period rank in the top 10, Li et al. (2020) linearly interpolated missing daily outflow values.

This procedure resulted in reasonable mobility estimates for most cities, but some cities remained disconnected, with no estimated incoming travelers (Figure S-5, A and B). Several small cities with few cases might be expected not to have a large impact on the overall analysis. However, if their case reports have likelihood 0 under a model then they can lead to a log-likelihood of $-\infty$ even for an otherwise suitable model.

We therefore added a small amount of additional movement between cities based on a gravity model,

$$M_{uj} = M_{uj}^{\text{li20}} + \frac{\mathcal{F}\,\bar{d}}{\bar{P}(t_0)} \times \frac{P_u(t_0)\,P_j(t_0)}{d_{uj}}, \tag{S20}$$

where M_{uj}^{li20} is the movement from city u to j used by Li et al. (2020), $P_u(t_0)$ is the initial population in city u; $\bar{P}(t_0)$ is the average population across all 373 cities; d_{uj} is the distance from city u to city j; $\bar{d}$ is the average of this distance over all $(373 \times 372)/2$ pairs; $\mathcal{F}$ is a mobility correction factor which we took as $\mathcal{F} = 20$ based on assessment of diagnostic plots. Figure S-6 shows that this modification does not provide a major distortion to the pattern of travel from the movement data. This figure displays only day 1 (January 10), but other days show similar patterns. Figure S-5 gives further evidence for this; the modification is sufficient to move the zero travel records toward the main body of data, but not enough to result in other qualitative changes.

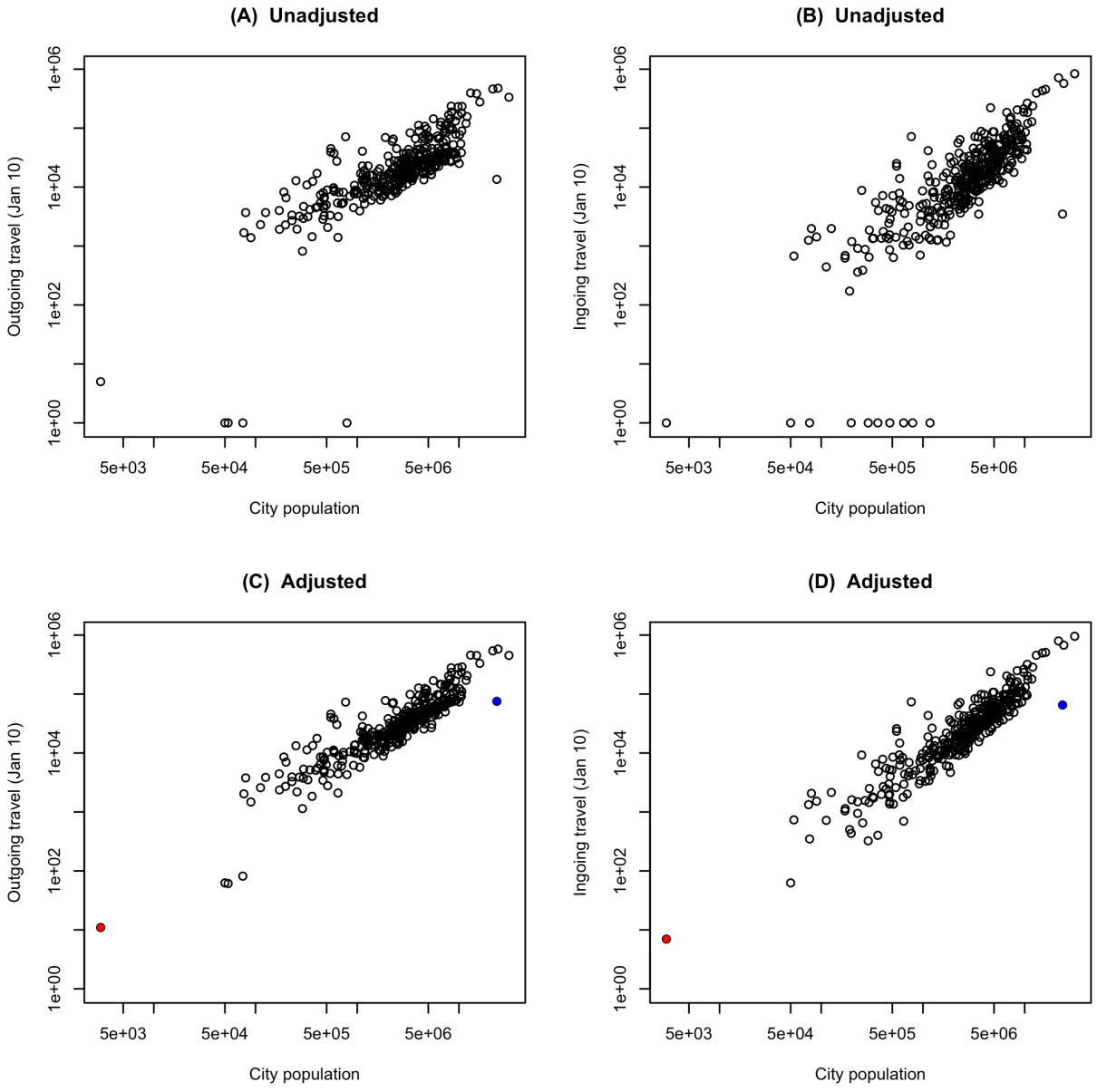

Figure S-5: Total ingoing and outgoing travel plotted against city size: (A,B) without an adjustment to ensure connectivity; (C,D) with the adjustment. The remaining outliers after adjustment are Sansha (red) and Taiwan (blue)

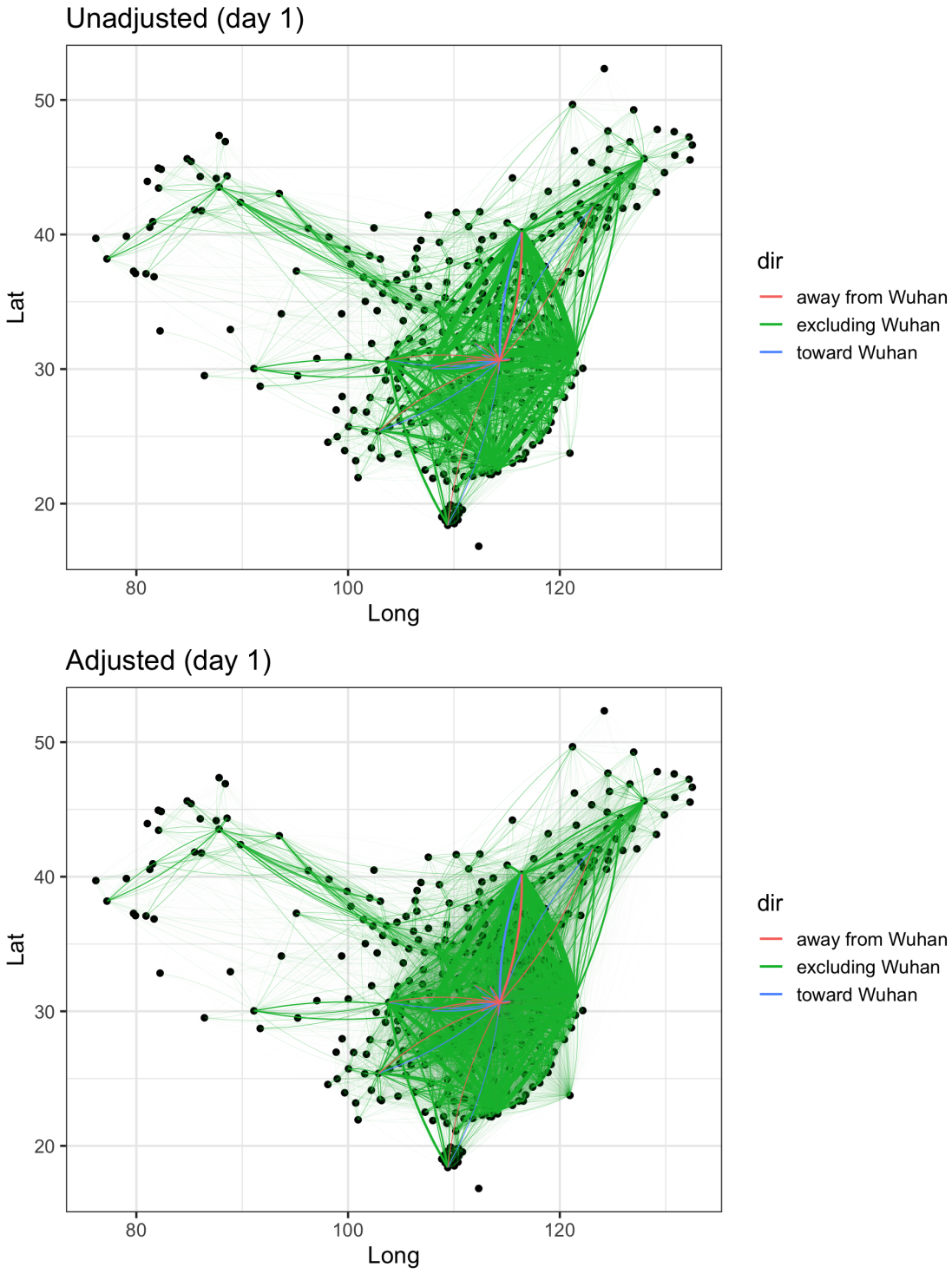

Figure S-6: Mobility graph for day 1. Top: without adjustment to ensure full connectivity. Bottom: with adjustment. Edge thickness is proportional to movement.

S1.3 Comparison with the model and data of Li et al. (2020)

Li et al. (2020) specified the compartment model as a set of ordinary differential equations, with Poisson noise on rates, solved using a 4th-order Runge-Kutta scheme. This approach constructs a discrete-time SpatPOMP, with the time discretization corresponding to the measurement times. The continuous-time SpatPOMP representation of `li23` has some practical advantages:

1. It leads to simpler code. For example, compare the representation of our dynamic model in the R package `metapoppkg` function `li23` with either the code provided by Li et al. (2020) or our direct adaptation of this code in the function `li20` in our `metapoppkg` package. Code simplicity facilitates debugging and consideration of model variations.

2. It allows inclusion of white noise on transmission rates to generate dynamic overdispersion (Bretó et al., 2009; He et al., 2010; Bretó and Ionides, 2011; Stocks et al., 2020). This can lead to better fits to data (measured by likelihood) as well as avoiding over-confident predictions resulting from models that cannot adequately explain the variability in the data.

Potentially, similar principles could be incorporated into `li20`, but the SpatPOMP framework makes the generalization more straightforward.

We implemented the `li20` model of Li et al. (2020) within the framework of the R package `spatPomp` (Asfaw et al., 2023). The model is constructed by the `metapoppkg` function `li20()`. This permits reproduction of the methodology used by Li et al. (2020) via the `spatPom` function `ienkf`. Our implementation of `li20` incorporates code adapted from supplementary information provided by Li et al. (2020). Simulation from `li20` using the parameters of Li et al. (2020) is therefore essentially equivalent to the simulation used by Li et al. (2020). In Figure S-7, we compare a simulation from the code of Li et al. (2020) with simulations from models M_1 and M_2. We see that `li20` and M_1 look similar; comparing with Figure S-2 we see that both have distincly less variability than the data. The additional variability in M_2 makes it look superficially more like the data, but the additional stochasticity is all ascribed to reporting variability in M_2; dynamic noise allows a better fit to the data, as documented formally in Table 1 and apparent from Figure S-2.

The dataset analyzed by Li et al. (2020) included 375 cities, but we study only 373. We found that two of the 375 cities are duplicates, with two slightly different names for the same town having the same location. Also, the island of Hainan was included together with its separate counties—Hainan has a different administrative structure from other Chinese provinces, and does not have prefecture cities. We removed the aggretated region of Hainan. We modified some of the population values used by Li et al. (2020). When there was a major discrepancy between the value in their dataset and the prefecture population reported by Wikipedia, we took the latter. The largest change was updating the population of Ezhou to 1,079,353 from 59,500. Complete details of all our modifications to the dataset used by Li et al. (2020) are reported in the `metapop` package.

S2 Benchmark statistical models

Basic statistical models, such as linear regression models or autoregressive-moving average (ARMA) time series models, or even independent random sample models, provide a baseline estimate of the predictability of the system under investigation. A standard measure of this predictability is the log-likelihood (Gneiting and Raftery, 2007), and it is therefore appropriate to compare log-likelihoods for different models calculated for the same data. A sophisticated mechanistic model might be expected to have higher predictive skill, and therefore a higher log-likelihood, than a simple statistical model. However, mechanistic models may be

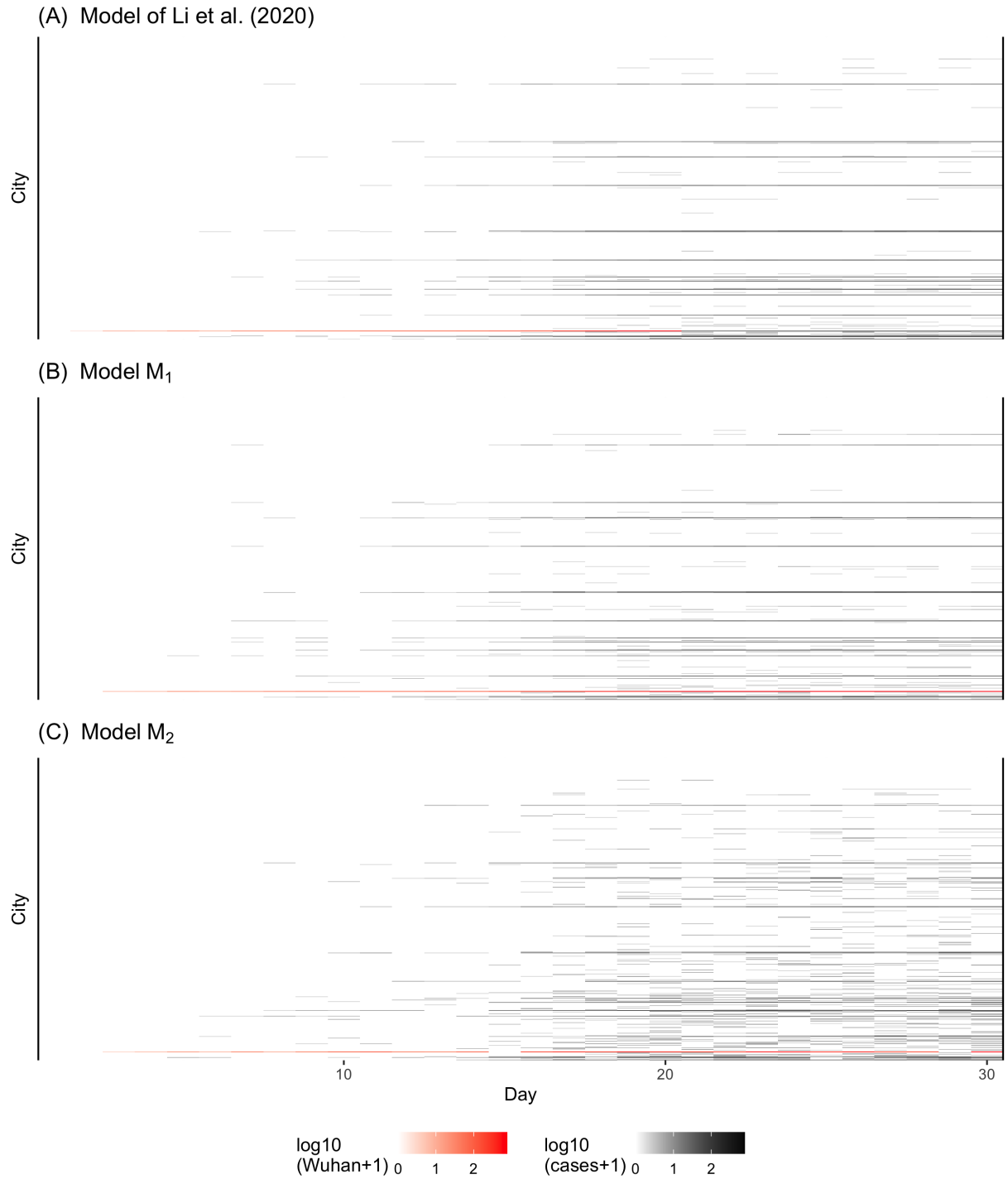

Figure S-7: Simulations for 373 cities, comparing (A) a simulation from the model code and parameters provided by Li et al. (2020); (B) a simulation from model M_1; (C) a simulation from model M_2. Within each panel, cities are ordered by population, largest on the bottom row.

informative for what they cannot explain as well as what they can: so far as a mechanistic model captures current understanding of the science of a system, we are interested to know when and where this science is inadequate to explain the data. By contrast, statistical models are designed solely to provide a statistical fit. If a mechanistic model fits substantially worse than a simple statistical model, one may infer that there is considerable room to improve the mechanistic model. If the mechanistic model is competitive with non-mechanistic alternatives, regardless of whether its likelihood is actually higher, we infer that the mechanistic model provides a plausible explanation of the data. Plausible mechanistic models can then be compared against each other by likelihood, with protection from the concern of infering support for one model by comparison against a weak "straw man" alternative. The use of benchmark likelihoods is demonstrated by (King et al., 2008; He et al., 2010; Wheeler et al., 2023). Here, we use two benchmarks:

(i) A negative binomial model which is independent and identically distributed (IID) for each time point with a unit, with a unit-specific mean. We parameterize the model in terms of its mean and variance, as

$$\mathbb{E}[Y_{u,n}] = \mu_u \quad \text{and} \quad \mathrm{Var} Y_{u,n} = \mu_u + \mu_u^2/s, \tag{S21}$$

where s is a scale parameter.

(ii) An autoregressive negative binomial model, where the count $Y_{u,n}$ is modeled as negative binomial conditional on $Y_{u,n-1}$ with mean and variance given by

$$\mathbb{E}\big[Y_{u,n}|Y_{u,n-1}\big] = \mu_u + \phi Y_{u,n-1} \quad \text{and} \quad \mathrm{Var}\big[Y_{u,n}|Y_{u,n-1}\big] = \mu_u + \phi Y_{u,n-1} + \big(\mu_u + \phi Y_{u,n-1}\big)^2/s, \tag{S22}$$

with the convention that $Y_{u,0} = 0$.

We did not adopt the previously used log-ARMA benchmark, because it is inappropriate for count data with many zeros. The likelihood was optimized using `optim` in R.

S3 The block particle filter and iterated block particle filter

We use the block particle filter of Rebeschini and van Handel (2015) implemented as `bpfilter` in the `spatPomp` package (Asfaw et al., 2023). A filter can evaluate the likelihood function but is not directly concerned with parameter estimation. Iterated block particle filters for parameter estimation were developed by Ning and Ionides (2023) and Ionides et al. (2022) and are implemented as `ibpf` in `spatPomp`. Here, we give in informal introduction to `bpfilter` and `ibpf`.

The particle filter (Arulampalam et al., 2002; Doucet and Johansen, 2011) can be heuristically understood as Darwinian evolution operating on a swarm of particles. Between consecutive observation times, each particle follows a random trajectory of the stochastic dynamic system. This randomness is analogous to Darwinian mutation. At an observation time, the particles are resamples with weights corresponding to the conditional density of the data given the location of the particle. The weights are Darwinian fitness, and the resampling is Darwinian natural selection.

Iterated filtering algorithms add perturbations to the parameters of each particle. Thus, the natural selection of particles favors parameter values consistent with the data, giving rise to algorithms that approach the maximum likelihood estimate (Ionides et al., 2006, 2015). A diagram representing an iterated block particle filter is showh in Figure S-8.

The performance of particle filters decays rapidly with the dimension of the latent state (Bengtsson et al., 2008). Iterated particle filters suffer from the same curse of dimensiounality. Block particle filters avoid this curse by partitioning the latent states into weakly dependent units, and resampling separately on

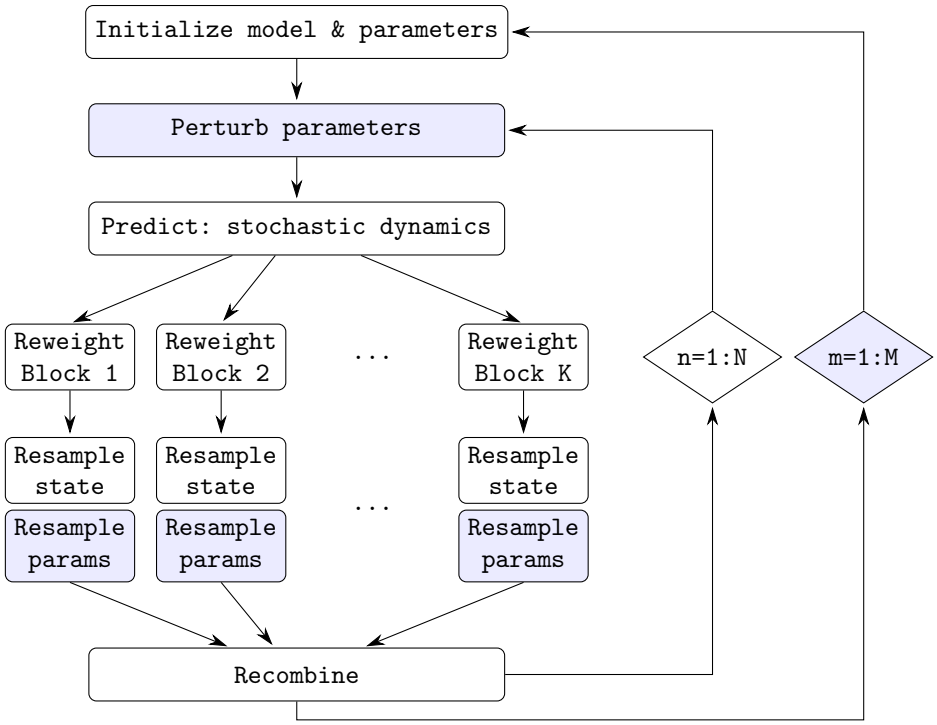

Figure S-8: A flow diagram for an iterated block particle filter. The outer loop ($m = 1, \ldots, M$) enables parameter estimation, when combined with the perturbation and resampling of the parameters.

each unit. This is analogous to recombination in sexual reproduction, with each block corresponding to a chromosome. The recombination allows successful blocks of one particle to join up with different successful blocks from another particle. If this approximation permits effective high-dimensional filtering then the parameter perturbation strategy can be employed for parameter estimation, just as for basic particle filters.

S3.1 Log-likelihood estimation via filtering

For our primary goal of carrying out inference on unknown model parameters, the principal motivation for filtering is to obtain an estimate of the log-likelihood. The log-likelihood is the probability density of the model, evaluated at the data, viewed as a function of the model parameters, defined as

$$\ell(\theta) = \log\left(f_{Y_{1:N}}(y^*_{1:N};\theta)\right), \tag{S23}$$

This quantity is fundamental for likelihood-based inference, including maximum likelihood estimation and Bayesian inference (via combination of the likelihood with prior beliefs).

The recursive nature of a filtering algorithm suggests using a likelihood decomposition

$$f_{Y_{1:N}}(y^*_{1:N};\theta) = \prod_{n=1}^{N} f_{Y_n|Y_{1:n-1}}(y^*_n|y^*_{1:n-1};\theta). \tag{S24}$$

The requirement of a filtering algorithm is to provide an approximation to

$$f_{X_n|Y_{1:n}}(x_n|y^*_{1:n};\theta),$$

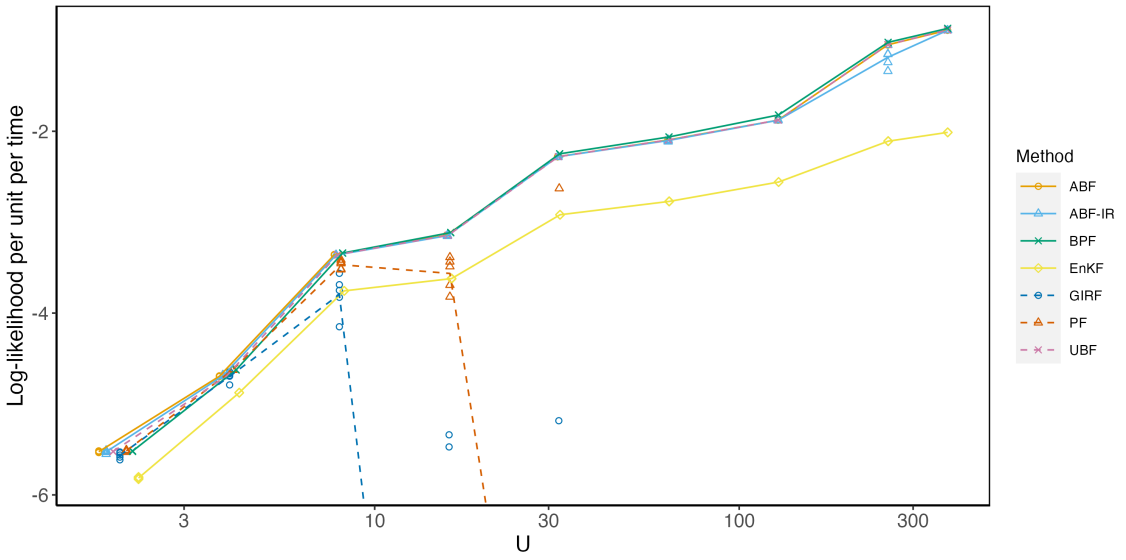

Figure S-9: Comparing filters on simulated data from model `li23`. ABF is the adapted bagged filter (Ionides et al., 2021); ABF-IR is the adapted bagged filter with intermediate resampling (Ionides et al., 2021); BPF is the block particle filter; EnKF is the ensemble Kalman filter; GIRF is the guided intermediate resampling filter (Park and Ionides, 2020); PF is the particle filter; UBF is the unadapted bagged filter (Ionides et al., 2021).

though many (including EnKF) also provide an approximation to the one-step prediction distribution, say

$$f^P_{X_{n+1}|Y_{1:n}}(x_{n+1}|y^*_{1:n};\theta) \approx f_{X_{n+1}|Y_{1:n}}(x_{n+1}|y^*_{1:n};\theta).$$

The approximations $f^P_{X_{n+1}|Y_{1:n}}$ together with a measurement density can be used to construct a model defined by a joint probability density,

$$f^P(y_{1:N};\theta) = \int \prod_{n=1}^{N} f_{Y_n|X_n}(y_n|x_n;\theta) f^P_{n+1}(x_{n+1}|y_{1:n};\theta)\, dx_{1:N}, \tag{S25}$$

with corresponding likelihood and log-likelihood functions,

$$L^P(\theta) = f^P(y^*_{1:N};\theta), \qquad \ell^P(\theta) = \log L^P(\theta).$$

Since $f^P_{X_{n+1}|Y_{1:n}}$ is an approximation, $L^P(\theta)$ does not exactly match the likelihood of the proposed model. We call $L^P(\theta)$ the predictive likelihood of the filtering algorithm applied to the model. It can be viewed as the exact likelihood of the approximate model for the data defined by, rather than the approximate likelihood of the original target model. This perspective implies that, if the proposed model is correct, the expected value of $\ell^P(\theta)$, viewed as a random function of $Y_{1:N}$, is lower than $\ell(\theta)$ since log-likelihood is a proper scoring rule (Gneiting and Raftery, 2007). Therefore, it is appropriate to compare filtering methods by their predictive likelihoods.

To investigate the suitability of the block particle filter for this data analysis, we compared log-likelihood evaluation from various filters available in the `spatPomp` package. We used simulated data from our maximum likelihood estimate so that the comparison is made on a correctly specified model, because we do not want to assess filters based on how well they compensate for model misspecification. For a correctly specified

model, the best possible expected log-likelihood arises for an ideal filter; in other words, log-likelihood is a proper scoring method for filters (Gneiting and Raftery, 2007). Figure S-9 shows that the block particle filter performs well on this task. The particle filter is effective for up to $U = 5$ cities, but (as theory predicts) its performance starts declining rapidly as dimension increases. The particle filter gives an unbiased and consistent Monte Carlo estimate of the likelihood, so we can tell from Figure S-9 that the block approximation is effective for small U since is it does not fall far behind the particle filter in situations where the latter is known to give essentially exact results. However, the block particle filter continues to operate successfully for large U, when the particle filter fails. Two bagged filters (ABF and UBF) also perform well, however their structure is not well suited to parameter estimation via iterated perturbed filtering (Ionides et al., 2021). The particle filters using guided intermediate resampling (ABF-IR and GIRF) are computationally expensive; in principle, they have good scalability properties, however, their `spatPomp` implementation performs less well than the unguided filters in this experiment.

The algorithmic settings for Figure S-9 were set for comparable computing times in the `spatPomp` implementation. Full details are available in the published source code. Different algorithms have different demands in terms of number of computations, memory requirements, and parallelizability. All these algorithms have various tuning parameters, so we do not rule out the possibility that the numerical comparisons are dependent on details of the implementation. Figure S-9 is similar to Figure 3 of Ionides et al. (2021), that shows an equivalent filter comparison for a longer collection of time series of an endemic viral disease, namely pre-vaccination measles. Different relative performance results can be obtained on other model classes, for example in a linear Gaussian model (Figure 1 of Ionides et al. (2021)) or the Lorenz 96 geophysical model (Figure S3 of Ionides et al. (2021)). Figure S-9 therefore adds to the growing body of evidence that block particle filters are well suited to metapopulation models.

S3.2 Limitations of IBPF

Simulation-based methods are computationally intensive. IBPF requires thousands of simulations in each of hundreds of filtering iterations. In practice, this limits the applicability to a moderate scale, say, hundreds of units.

IBPF approximates a full-information maximum-likelihood analysis, yet both the likelihood evaluation and maximization are subject to errors that could become scientifically significant. IBPF builds on the block particle filter, and so IBPF cannot expect to succeed in situations where BPF fails. However, it is possible for IBPF to fail in situations where BPF succeeds, We discuss separately these two requirements for success of IBPF.

BPF has good scalability properties, but an approximation error which can be large when the model is not a good match for the method. The theory within which BPF has small approximation error assumes that dependence decays suitably quickly with distance between units (Rebeschini and van Handel, 2015). The theory for IBPF, in the case where all parameters are unit-specific, has a similar requirement, but on an extended model where the latent state at each unit is augmented with a parameter vector carrying out a random walk. There is not yet a comparable theory for the shared parameter extension of IBPF proposed and demonstrated by Ionides et al. (2022), but a similar requirement is probably needed.

The size of an initial infected population at a single source unit, Wuhan for COVID-19, is an example of a potentially problematic parameter to estimate via IBPF. In the extended model, this parameter is local to the state at Wuhan, and yet it is important for the dynamics of other units. Fortunately, the initialization procedure does award this parameter some direct effect on the initial number of infections in other units, which the algorithm can harness. Without this, the IBPF algorithm would lose any power for events outside the block containing Wuhan to inform the initial state in Wuhan.

Algorithm 1: EnKF algorithm (adapted from Asfaw et al., 2023)

input: simulator for the transition density, $f_{\mathbf{X}_n|\mathbf{X}_{n-1}}(\mathbf{x}_n \,|\, \mathbf{x}_{n-1}\,;\theta)$, and initial density, $f_{\mathbf{X}_0}(\mathbf{x}_0\,;\theta)$; evaluator for expectation of $Y_{u,n}$ given $X_{u,n}=x$, $e_u(x,\theta)$, and corresponding variance, $V_u(x,\theta)$; parameter, θ; data, $\mathbf{y}^*_{1:N}$; number of particles, J.

1 initialize filter particles, $\mathbf{X}_0^{F,j} \sim f_{\mathbf{X}_0}(\,\cdot\,;\theta)$ for j in $1{:}J$

2 **for** n in $1{:}N$ **do**

3 prediction ensemble, $\mathbf{X}_n^{P,j} \sim f_{\mathbf{X}_n|\mathbf{X}_{n-1}}\big(\,\cdot\,|\mathbf{X}_{n-1}^{F,j};\theta\big)$ for j in $1{:}J$

4 centered prediction ensemble, $\tilde{\mathbf{X}}_n^{P,j} = \mathbf{X}_n^{P,j} - \frac{1}{J}\sum_{q=1}^{J}\mathbf{X}_n^{P,q}$ for j in $1{:}J$

5 forecast ensemble, $\hat{\mathbf{Y}}_n^j = e_u(X_{u,n}^{P,j},\theta)$ for j in $1{:}J$

6 forecast mean, $\overline{\mathbf{Y}}_n = \frac{1}{J}\sum_{j=1}^{J}\hat{\mathbf{Y}}_n^j$

7 centered forecast ensemble, $\tilde{\mathbf{Y}}_n^j = \hat{\mathbf{Y}}_n^j - \overline{\mathbf{Y}}_n$ for j in $1{:}J$

8 forecast measurement variance, $R_{u,\tilde{u}} = \mathbb{1}_{u,\tilde{u}}\,\frac{1}{J}\sum_{j=1}^{J}V_u\big(\mathbf{X}_{u,n}^{P,j},\theta\big)$ for $u,\tilde{u}$ in $1{:}U$

9 forecast estimated covariance, $\Sigma_Y = \frac{1}{J-1}\sum_{j=1}^{J}(\tilde{\mathbf{Y}}_n^j)(\tilde{\mathbf{Y}}_n^j)^T + R$

10 prediction and forecast sample covariance, $\Sigma_{XY} = \frac{1}{J-1}\sum_{j=1}^{J}(\tilde{\mathbf{X}}_n^{P,j})(\tilde{\mathbf{Y}}_n^j)^T$

11 Kalman gain, $K = \Sigma_{XY}\Sigma_Y^{-1}$

12 artificial measurement noise, $\epsilon_n^j \sim \text{Normal}(\mathbf{0}, R)$ for j in $1{:}J$

13 errors, $\mathbf{r}_n^j = \hat{\mathbf{Y}}_n^j - \mathbf{y}_n^*$ for j in $1{:}J$

14 filter update, $\mathbf{X}_n^{F,j} = \mathbf{X}_n^{P,j} + K\big(\mathbf{r}_n^j + \epsilon_n^j\big)$ for j in $1{:}J$

15 $\ell_n = \log\big[\phi\big(\mathbf{y}_n^*;\overline{\mathbf{Y}}_n,\Sigma_Y\big)\big]$ where $\phi(\cdot\,;\mu,\Sigma)$ is the Normal(μ,Σ) density.

16 **end**

output: filter sample, $\mathbf{X}_n^{F,1:J}$, for n in $1{:}N$; log likelihood estimate, $\ell^{\text{EnKF}} = \sum_{n=1}^{N}\ell_n$

Determining the success of a filter, in a particular application, is an empirical task. Many filters, including BPF and the basic particle filter, evaluate the log-likelihood by constructing a sequence of one-step predictive distributions which do not have access to future data. The log-likelihood is a proper scoring rule for such forecasts (Gneiting and Raftery, 2007), meaning that, if the model is correct, no other predictive distribution can have higher expected log-likelihood. This motivates comparison of filters by their estimated log-likelihood: the higher, the better.

Caution is required in the presence of model misspecification. For example, the model may place zero probability on specific latent state values, yet these states may become possible in a block particle filter due to the block resampling. If the data favor these inconsistent values (an indication of model misspecification) then the block particle filter may have much higher likelihood that an ideal filter. For this reason, we recommend that experimentation to determine the choice of filter should be carried out on simulated data as well as the actual data. If the conclusions are different in these two scenarios, that is an indication of model misspecification.

S4 The ensemble Kalman filter (EnKF) and its use for metapopulation models

EnKF algorithms (Evensen et al., 2022) have proved effective for moderately nonlinear, non-Gaussian data assimilation tasks with large amounts of data. Algorithm 1 gives pseudocode for an EnKF algorithm, using notation for SpatPOMP models consistent with the `spatPomp` R package (Asfaw et al., 2023). EnKF algorithms update each member of an ensemble using a Kalman gain, constructed in line 11 of Algorithm 1.

This linear update rule corresponds to an ideal filter (i.e., the Kalman filter) when the observations and latent states are jointly linear and Gaussian. The nonlinear and non-Gaussian behavior permitted in the prediction step (line 3) makes some appropriate adaptation for general SpatPOMP models, but does not in general guarantee a good approximation to the ideal nonlinear filter. We see in Figure S-9 that the performance of EnKF on the `li23` model falls substantially below some filters with nonlinear update rules.

A technical requirement for proper likelihood-based comparison of two models is that the likelihoods are calculated with respect to the same base measure. This technical consideration can become important in the context of EnKF since this algorithm is motivated by a continuous, real-valued probability distribution (the Gaussian distribution) yet is applied to discrete, integer-valued metapopulation models. When population counts are not small, evaluating a Gaussian probability density function at integer values may be a close approximation to the probability mass function of a discrete model. However, when counts are small, we may encounter a situation where the prediction variance is small, in which case the Gaussian probability density is unbounded. By contrast, a valid probability mass function (i.e., a discrete probability density with respect to counting measure) can never attain values higher than 1.

Li et al. (2020) prevented this issue by putting a lower bound of 4 on the observation variance for their EnKF implementation. We implemented their approach in our EnKF calculations for Figure S-9, by replacing $V_{u,n}$ in equation (S19) with

$$V_{u,n} = \min(4, C_{u,n}^2/4) \tag{S26}$$

when carrying out inference via the ensemble Kalman filter. Further, Li et al. (2020) introduce a measure of discrepancy between the fitted model and the data which they call log-likelihood but which is not an approximation to the statistical quantity $L(\theta)$. The quantity shown in their Figure 1, and described in their supplementary Section 8, could be viewed as an approximation to a pseudo-log-likelihood (Besag, 1974). However, this quantity is not the predictive log-likelihood of any model, and cannot properly be compared with log-likelihoods from alternative models. An alternative approach for employing EnKF algorithms with discrete data is to embed EnKF within a Markov chain Monte Carlo algorithm (Katzfuss et al., 2020).

S4.1 Mismatches between the model and the EnKF specification

Modifications to the EnKF implementation may improve filtering but break the correspondence between the algorithm and the postulated model. For example, (Li et al., 2020) use a measurement variance that depends on the observation itself, which superficially suggests the mathematically inconsistent expression

$$\mathrm{Var}(Y_n|X_n) = \min(4, Y_n^2/4).$$

Filtering may proceed with this variance specification, but it does not correspond to a valid predictive distribution since a conditional variance for Y_n cannot depend on Y_n.

Some mismatch between the predictive likelihood and the actual model likelihood occurs whenever using numerical methods. Inference is necessarily based on the numerically implementation of the filtered model and its likelihood, implying that the predictive likelihood is a proper measure of fit for the procedure actually implemented. As pointed out in Sec. S3.1, a filter generating a legitimate predictive likelihood is penalized for infelicity to the intended model, so far as the intended model fits the data. This permits likelihood-based comparison candidate filters as well as candidate models. However, a non-predictive likelihood approximation calculated requires additional care in its interpretation; use of future information could lead to higher likelihoods than can be obtained by even an ideal filter. If it is expedient to use a non-predictive likelihood approximation, this issue requires care.

S5 Estimation for the unconstrained model, M_5

We calculated profile likelihoods for each parameter in M_5, in order to assess identifability and obtain confidence intervals. Profile likelihood involves fixing one parameter at a range of values while maximizing the log-likelihood with respect to all the other estimated parameters. We use Monte Carlo adjusted profile (MCAP) methodology which provides a way to construct likelihood-based confidence intervals in situations where Monte Carlo variability in maximization and evaluation of the log-likelihood is too large to ignore (Ionides et al., 2017; Ning et al., 2021). An estimate of the profile likelihood is obtained by applying a smoothing algorithm (such as a smoothing spline) to these noisy evaluations. The MCAP confidence interval selects the region of the estimated profile above a cutoff value, where the cutoff is chosen to combine the statistical uncertainty of the ideal (inaccessible) likelihood function with the Monte Carlo variability of the available estimate of the likelihood function. MCAP is not appropriate when the maximum likelihood occurs on the boundary of the parameter space, which occurs here for the initial unobserved cases, A_0, and the initial relative transmissibility, μ^{be}. For these parameters we therefore used a basic likelihood ratio test on the smoothed likelihood, unadjusted for Monte Carlo error.

The resulting profiles are shown in Figures S-10, S-11 and S-12. We see from Figure S-10 that D^{be} is weakly identified, and arbitrarily high values of this parameter can be consistent with the data. In this model, the mean 9-day distributed delay in reporting means that the initial dynamics cannot have many visible consequences in the early data. This is consistent with the absence of reported cases in the first 6 days of the dataset. However, as a consequence, the data have limited ability to identify model parameters, increasing the possibility that certain parameters, or combinations of parameters, have large statistical uncertainty. We resolved this situation by adding two constraints, $D^{\mathrm{be}} = D^{\mathrm{af}} = D$ and $Z^{\mathrm{be}} = Z^{\mathrm{af}} = Z$, to obtain the constrained model, M_6.

S6 Estimation for the constrained model, M_6

Figures S-13, S-14 and S-15 graph the profile likelihood evaluations and construct the resulting confidence intervals, following the same procedures used for Figures S-10, S-11 and S-12. For both models M_5 and M_6, the initial count of unreportable infections in Wuhan, A_0, is indistinguishable from $A_0 = 0$. Evidently, the model prefers to explain the data by placing the initial cases in the latent state, E. However, the evidence is not strong: the profiles for A_0 show compatibility with $A_0 = 5000$ for a cost of around 25-50 units of log likelihood. That is strong statistical evidence in the context of the model under investigation, since a 95% confidence interval contains only values within around 2 log units.

Formally, confidence intervals (like other forms of model-based statistical inference) are constructed based on a class of models under consideration. The meaning of these intervals in the context of models outside the class under consideration is generally unclear. However, the flatter the profile, the easier for some relatively small, unmodeled phenomenon to affect the estimate. Thus, to understand the robustness of the results to model misspecification, it can be helpful to consider the effect of larger likelihood cutoffs.

Standard robust statistical methods concern proper inference when aspects of the model are statistically inadequate, but comparison against appropriate benchmarks provides protection against this type of model misspecification. For example, we do not have to be excessively concerned about the possible effect of inappropriate modeling of dependence on confidence intervals if our mechanistic model has a likelihood comparing favorably against associative models having flexible specification of dependence. A different type of model misspecification arises when important explanatory variables are missing, or the postulated causal structures in the model class do not adequately represent reality. Such unknowns cannot readily be accounted for in standard error estimates. The curvature of the profile likelihood may indicate how robust the results

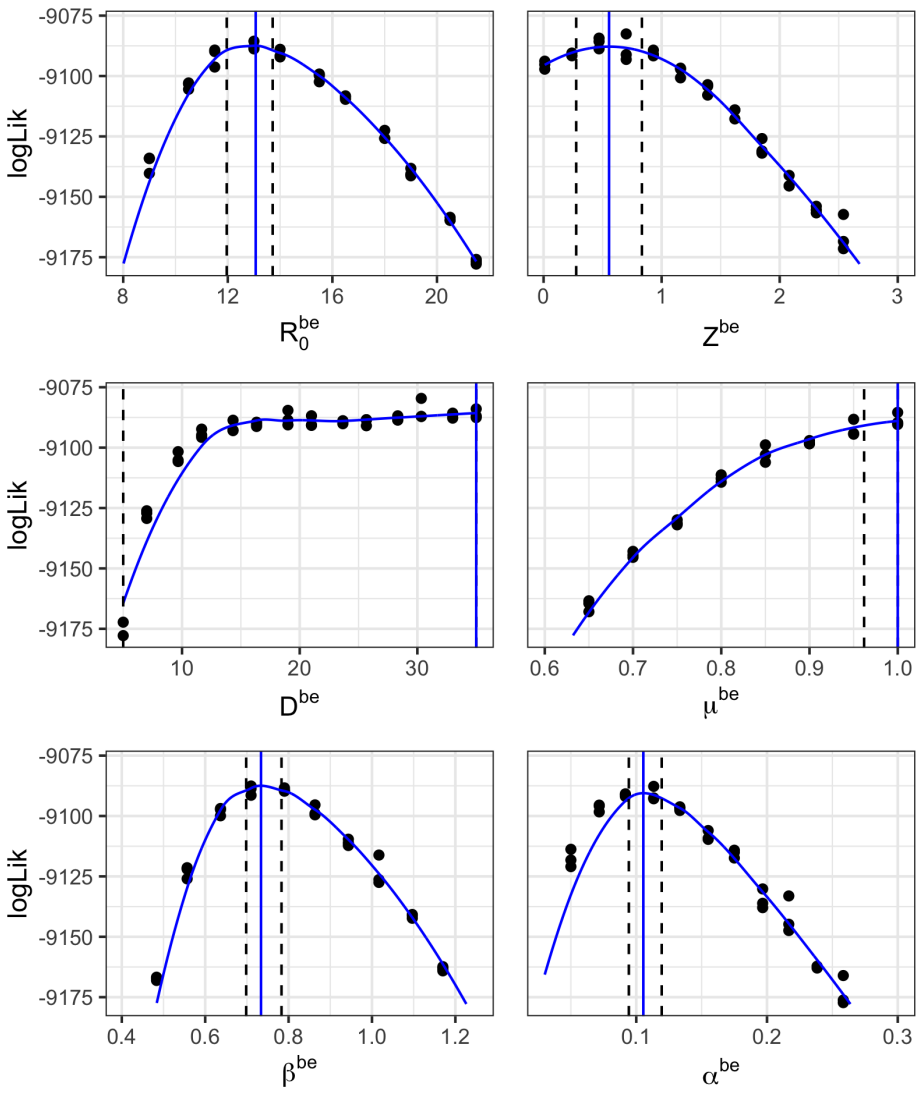

Figure S-10: Profile log-likelihood for model M_5 parameters before lockdown: $\mathcal{R}_0^{be}$, Z^{be}, D^{be}, μ^{be} and β^{be}.

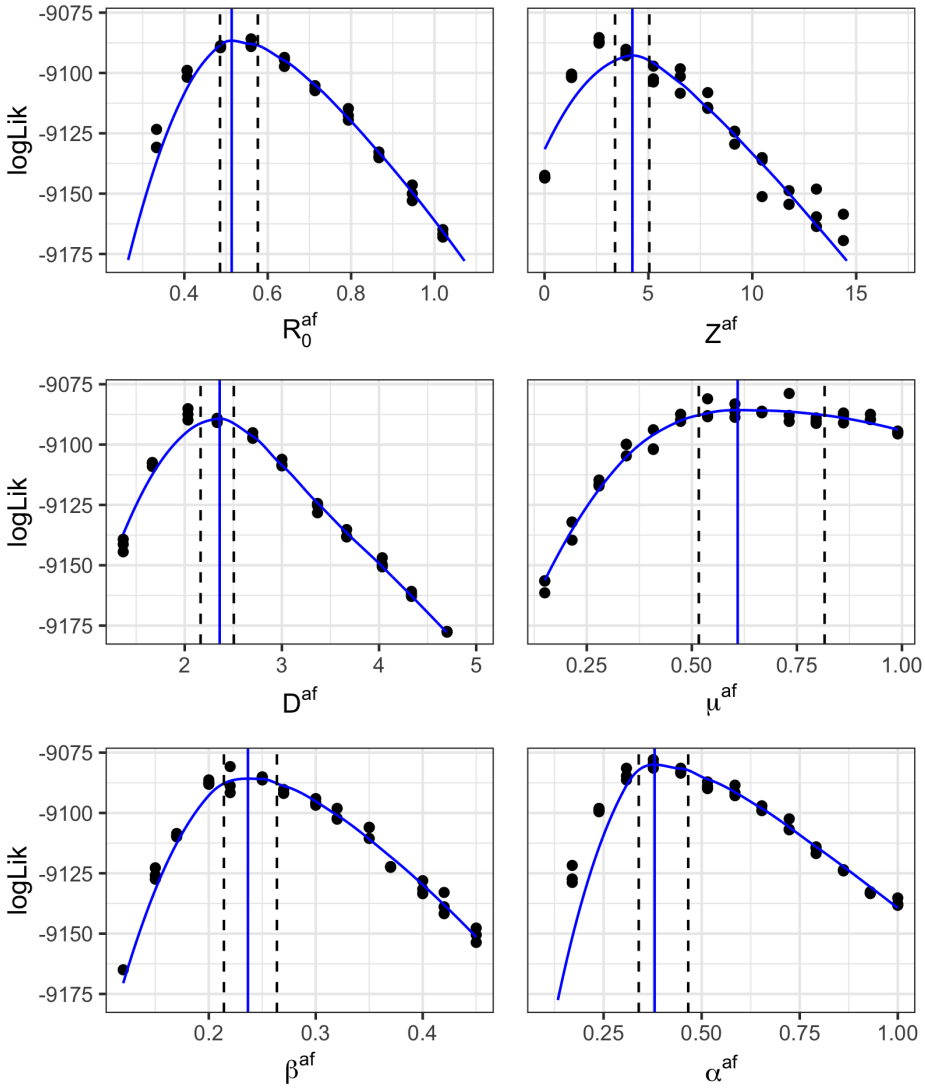

Figure S-11: Profile log-likelihood for model M_5 parameters after lockdown: $\mathcal{R}_0^{af}$, Z^{af}, D^{af}, μ^{af}, β^{af} and α^{af}.

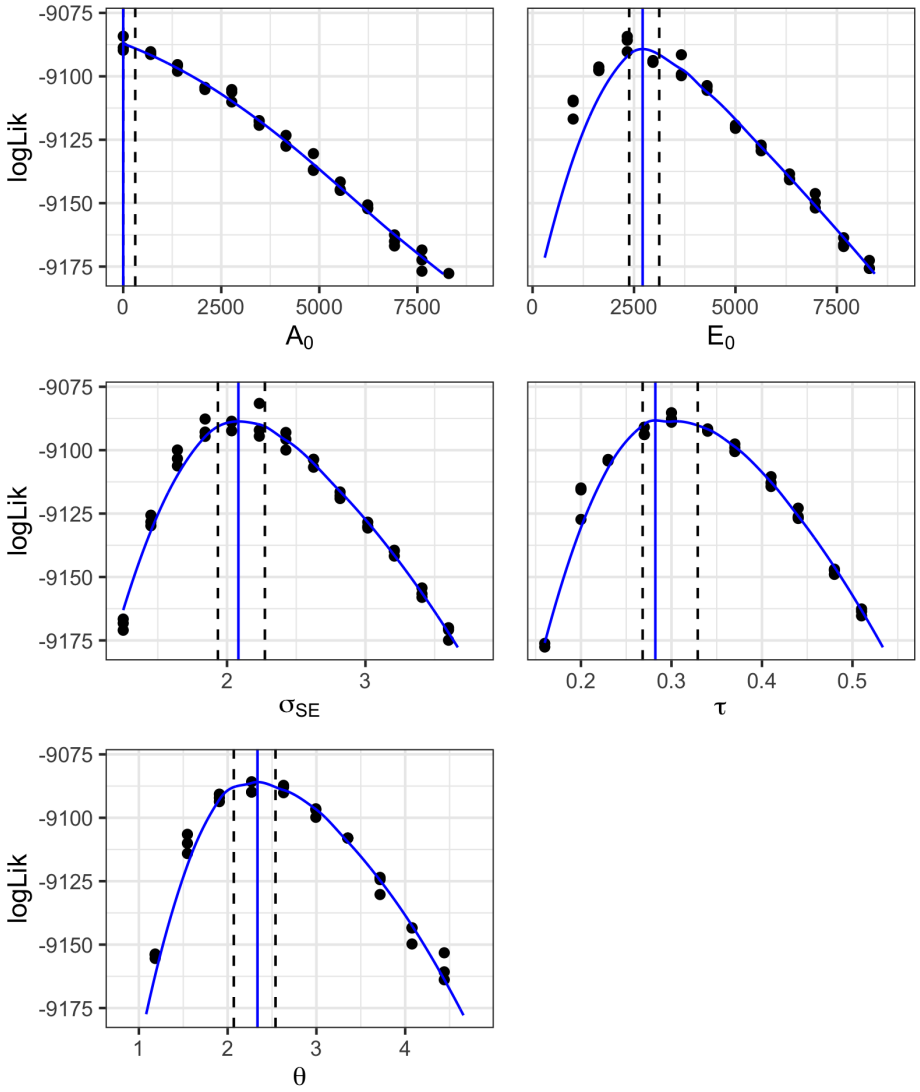

Figure S-12: Profile log-likelihood for model M_5 parameters which are unaffected by lockdown: τ, θ, σ_{SE}, A_0 and E_0.

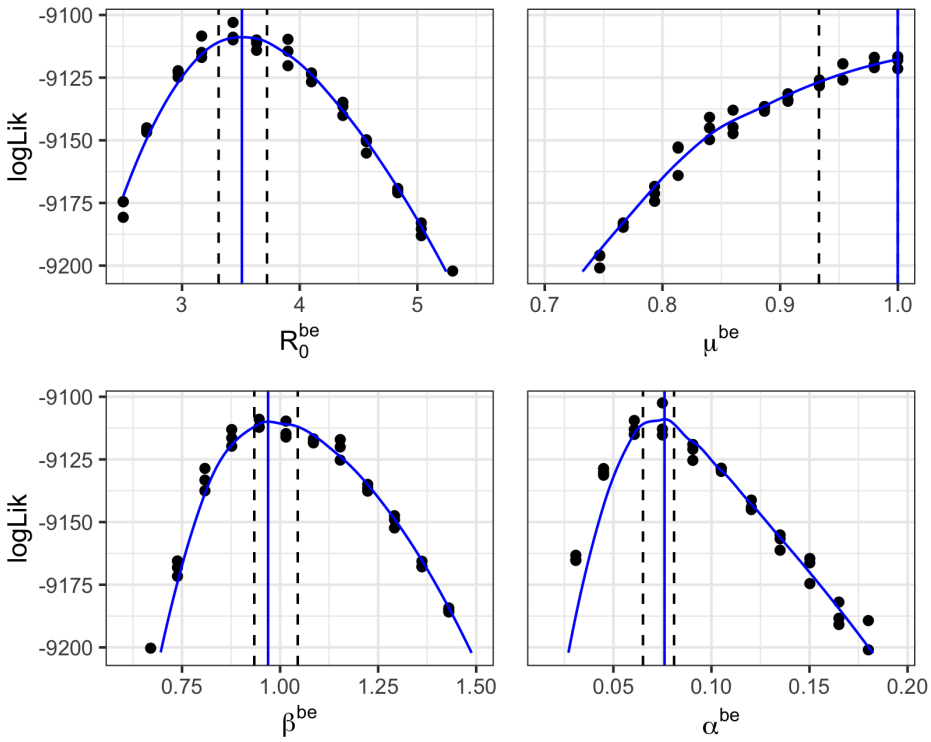

Figure S-13: Profile log-likelihood for model M_6 parameters before lockdown: $\mathcal{R}_0^{be}$, μ^{be} and β^{be}.

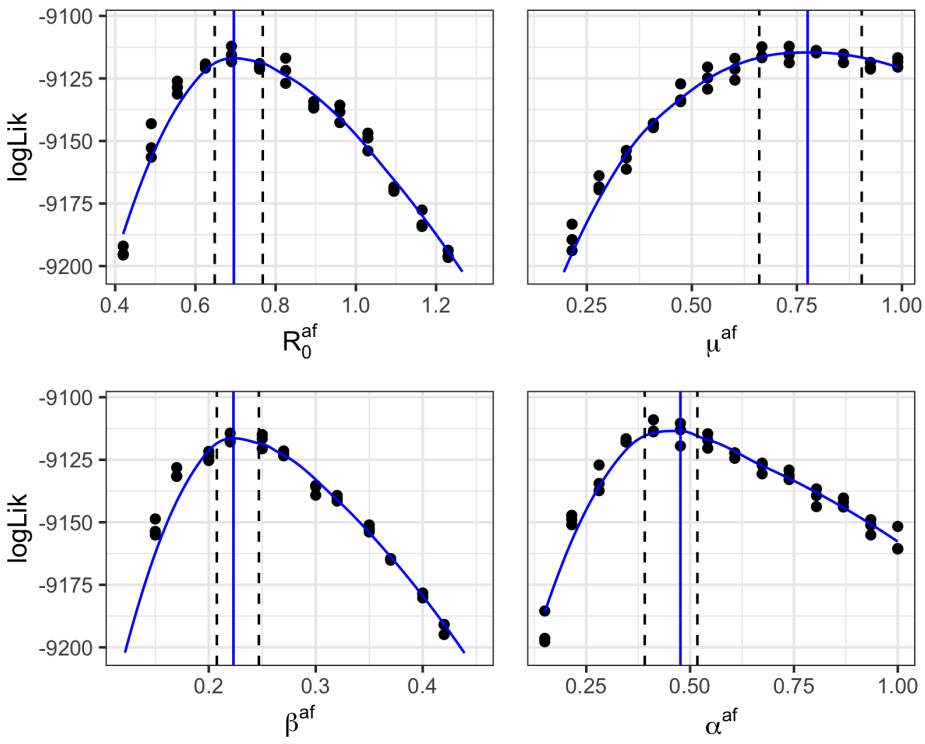

Figure S-14: Profile log-likelihood for model M_6 parameters after lockdown: $\mathcal{R}_0^{af}$, μ^{af}, β^{af} and α^{af} in model M_6

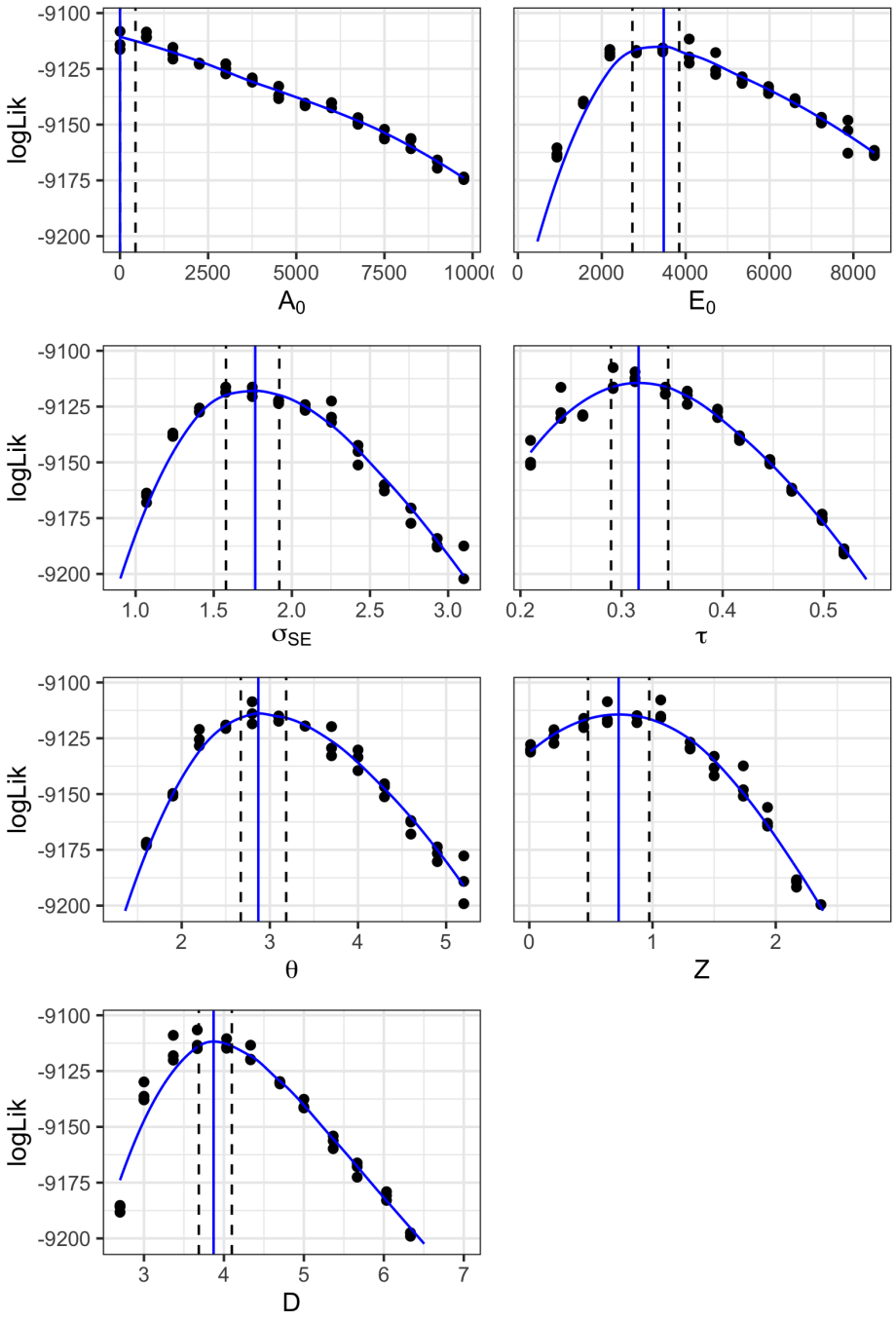

Figure S-15: Profile log-likelihood for model M_6 parameters which are unaffected by lockdown: τ, θ, Z, D, σ_{SE}, A_0 and E_0.

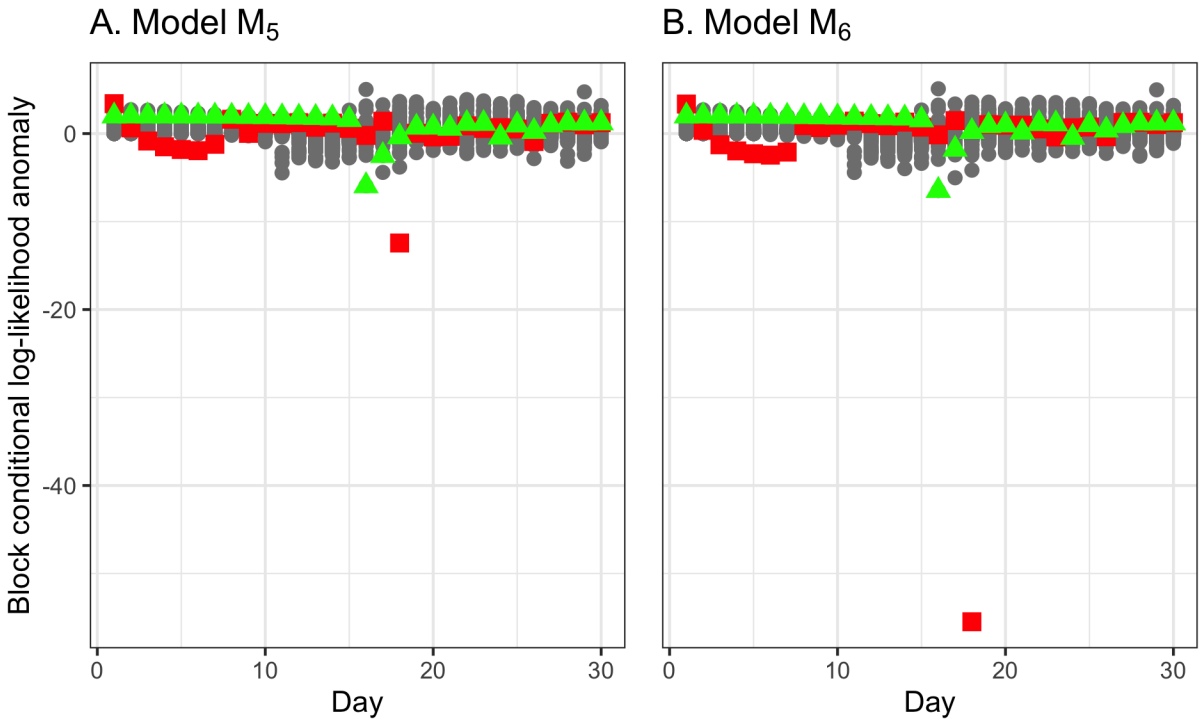

Figure S-16: Conditional log-likelihood anomaly for each city at each time point. Panel A corresponds to the best fit for M_5 and panel B the best fit for M_6. Points for Wuhan are red squares, and Huangshi are green triangles.

are to small misspecifications. For large misspecifications, parameters in differing models may have entirely different causal interpretation, and the respective likelihoods provide a measure of support from the data concering each hypothesis.

S7 Anomaly analysis

The block particle filter log-likelihood estimate can be decomposed into block conditional log-likelihoods for each city at each time point. The total estimated log-likelihood for the full dataset is the sum of these block conditional log-likelihoods. Likelihood depends on the units of the measured quantity, leading to a scale-dependent additive constant in the log-likelihood. To remove this constant, and to compare the model with a simple statistical prediction, we consider the log-likelihood anomaly, defined to be the model log-likelihood minus the benchmark log-likelihood. The log-likelihood anomaly at each time for each block is the corresponding difference for the block conditional log-likelihood. These log-likelihood anomalies can be investigated to look for patterns of interest; they are analogous to the residuals (i.e., observations minus predicted values) used for diagnostic analysis of regression models. An anomaly for an observation much smaller than -1 suggests that the data point is poorly explained by the mechanistic model, in which case it is called an outlier.

The largest outlier for both models M_5 and M_6 arises for Wuhan on day 18. This corresponds to the dramatic increase in reported cases on that day, shown in Figure S-17. This feature is a much more sever anomaly for

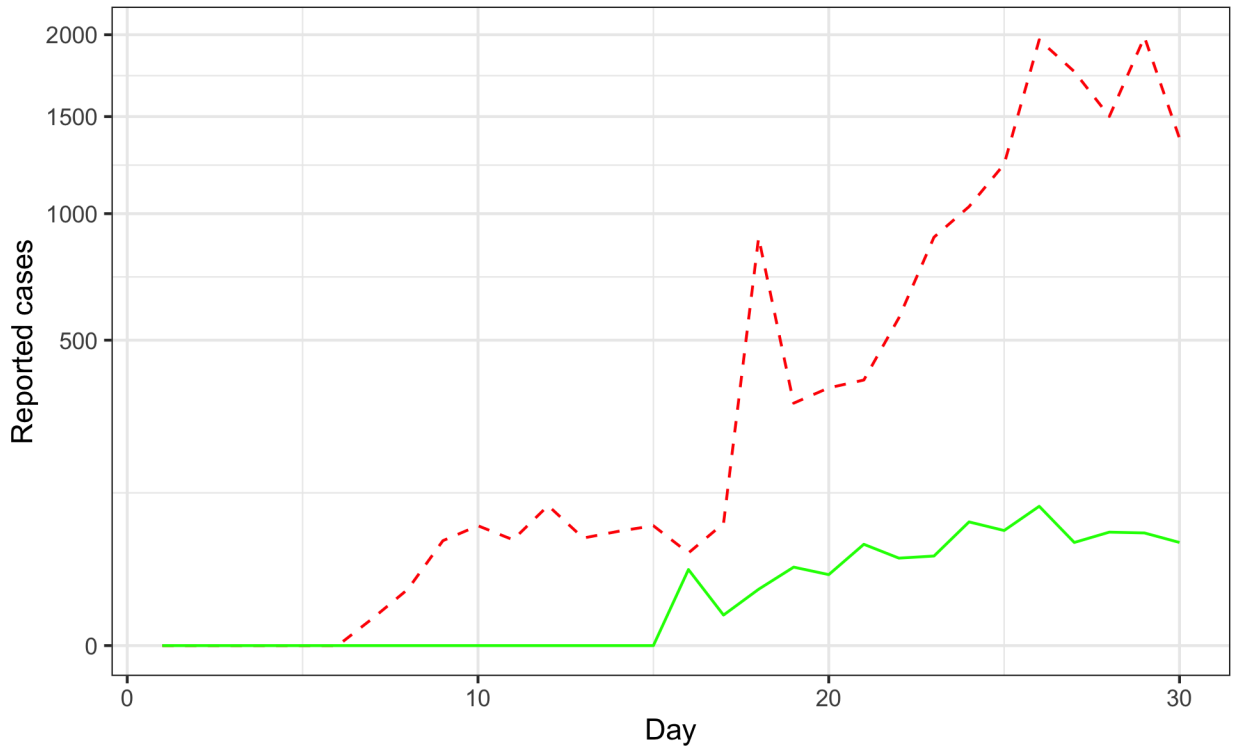

Figure S-17: Time plot of reported cases in Wuhan (red,dashed) and Huangshi (green,solid).

the constrained model, M_6. Indeed, the difference in the anomalies, which is 43.0 log-likelihood units, is more than enough to explain the difference in maximized log-likelihood between these two models. Evidently, the dynamics of the fitted model M_5 manage to better explain this outlier by postulating a long latency period and high $\mathcal{R}_0$ before the lockdown so that there is a larger supply of cases ready to explain the dramatic increase in reported cases on day 18. We see that one extreme outlier, which may represent an idiosyncrasy of the reporting process rather than a feature of the underlying dynamics, can have substantial consequences on the fit and the resulting conclusions. Essentially, the reported Wuhan cases on day 18 are inconsistent with model M_6; it pays a heavy price for this in terms of log-likelihood, but that may not be a scientific concern since it may indeed be the case that the reported peak on day 18 was not a genuine feature of the dynamics. Diagnostic protocols for COVID-19 were in their infancy, and changing rapidly, so the anomaly can be explained as a unique event in the reporting system.

The second largest outlier occured on day 16 in Huangshi, a city only 100km from Wuhan. Again, this corresponds to a sudden surge in reported cases (shown in Figure S-17) beyond what the model can account for.

Here, we do not show investigations of anomalies that were carried out while developing our model and correcting errors in the data. The need to correct certain population values described in Section S1.3 was identified by looking to explain why some cities had large anomalies. Discrepancies between the model and data may be (i) a problem with the model; (ii) an error in the data; (iii) an unavoidable consequence of limitations of the model or data, without being a major flaw in either. The first task of data analysis is to identify such discrepancies, since that is prerequisite for evaluating what should be learned from them.

S8 The metapoppkg R package

Source code reproducing the numerical results in the article and this supplement is available at https://github.com:jifanli/metapop_article. The code builds on a software package, `metapoppkg`, available at https://github.com:jifanli/metapoppkg. This package provides the dataset and models under consideration, as well as some useful data analysis operations. The documentation and unit tests for `metapoppkg` help to make the data analysis extendable: they reduce the overhead for subsequent investigators to adapt the analysis we present with variations to the models, data or statistical methods. Extendable data analysis is discussed by Wheeler et al. (2023).

The central component of `metapoppkg` is the function `li23` builds the model described in Section S1. The arguments permit specification of the number of spatial units and number of observation times. `li20` is similar to `li23` but aims to replicate the model form of Li et al. (2020), as described in S1.3. `R0` evaluates the value of $\mathcal{R}_0$ for a given set of parameter values. `incidence`, `mobility` and `population` import the corresponding epidemiological datasets uses for the data analysis. The `metapoppkg` package builds heavily on the `spatPomp` package (Asfaw et al., 2023), which in turn builds on the `pomp` package (King et al., 2016). Generic plotting methods for data, simulations, and diagnostic plots are provided by these packages. We also use plotting functions designed specifically for the analysis presented here, and made available as the `plot_dist` and `plot_li` functions in `metapoppkg`.

Supplementary References

Arulampalam, M. S., Maskell, S., Gordon, N., and Clapp, T. (2002). A tutorial on particle filters for online nonlinear, non-Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing*, 50:174 – 188.

Asfaw, K., Park, J., King, A. A., and Ionides, E. L. (2023). Statistical inference for spatiotemporal partially observed Markov processes via the R package spatPomp. *arXiv:2101.01157v3*.

Bengtsson, T., Bickel, P., and Li, B. (2008). Curse-of-dimensionality revisited: Collapse of the particle filter in very large scale systems. In Speed, T. and Nolan, D., editors, *Probability and Statistics: Essays in Honor of David A. Freedman*, pages 316–334. Institute of Mathematical Statistics, Beachwood, OH.

Besag, J. E. (1974). Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society, Series B (Statistical Methodology)*, 36:192–225.

Bretó, C., He, D., Ionides, E. L., and King, A. A. (2009). Time series analysis via mechanistic models. *Annals of Applied Statistics*, 3:319–348.

Bretó, C. and Ionides, E. L. (2011). Compound Markov counting processes and their applications to modeling infinitesimally over-dispersed systems. *Stochastic Processes and their Applications*, 121:2571–2591.

Doucet, A. and Johansen, A. (2011). A tutorial on particle filtering and smoothing: Fifteen years later. In Crisan, D. and Rozovsky, B., editors, *Oxford Handbook of Nonlinear Filtering*. Oxford University Press.

Evensen, G., Vossepoel, F. C., and van Leeuwen, P. J. (2022). *Data assimilation fundamentals: A unified formulation of the state and parameter estimation problem*. Springer Nature.

Gneiting, T. and Raftery, A. E. (2007). Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378.

He, D., Ionides, E. L., and King, A. A. (2010). Plug-and-play inference for disease dynamics: Measles in large and small towns as a case study. *Journal of the Royal Society Interface*, 7:271–283.

Ionides, E. L., Asfaw, K., Park, J., and King, A. A. (2021). Bagged filters for partially observed interacting systems. *Journal of the American Statistical Association*, 0(ja):1–33.

Ionides, E. L., Bretó, C., and King, A. A. (2006). Inference for nonlinear dynamical systems. *Proceedings of the National Academy of Sciences of the USA*, 103:18438–18443.

Ionides, E. L., Breto, C., Park, J., Smith, R. A., and King, A. A. (2017). Monte Carlo profile confidence intervals for dynamic systems. *Journal of the Royal Society Interface*, 14:1–10.

Ionides, E. L., Nguyen, D., Atchadé, Y., Stoev, S., and King, A. A. (2015). Inference for dynamic and latent variable models via iterated, perturbed Bayes maps. *Proceedings of the National Academy of Sciences of the USA*, 112(3):719—-724.

Ionides, E. L., Ning, N., and Wheeler, J. (2022). An iterated block particle filter for inference on coupled dynamic systems with shared and unit-specific parameters. *Statistica Sinica*, pages pre–published online.

Katzfuss, M., Stroud, J. R., and Wikle, C. K. (2020). Ensemble Kalman methods for high-dimensional hierarchical dynamic space-time models. *Journal of the American Statistical Association*, 115(530):866–885.

King, A. A., Ionides, E. L., Pascual, M., and Bouma, M. J. (2008). Inapparent infections and cholera dynamics. *Nature*, 454:877–880.

King, A. A., Nguyen, D., and Ionides, E. L. (2016). Statistical inference for partially observed Markov processes via the R package pomp. *Journal of Statistical Software*, 69:1–43.

Li, R., Pei, S., Chen, B., Song, Y., Zhang, T., Yang, W., and Shaman, J. (2020). Substantial undocumented infection facilitates the rapid dissemination of novel coronavirus (SARS-CoV-2). *Science*, 368(6490):489–493.

Ning, N. and Ionides, E. L. (2023). Iterated block particle filter for high-dimensional parameter learning: Beating the curse of dimensionality. *Journal of Machine Learning Research*, 24:1–76.

Ning, N., Ionides, E. L., and Ritov, Y. (2021). Scalable Monte Carlo inference and rescaled local asymptotic normality. *Bernoulli*, 27:2532–2555.

Park, J. and Ionides, E. L. (2020). Inference on high-dimensional implicit dynamic models using a guided intermediate resampling filter. *Statistics & Computing*, 30:1497–1522.

Rebeschini, P. and van Handel, R. (2015). Can local particle filters beat the curse of dimensionality? *The Annals of Applied Probability*, 25(5):2809–2866.

Stocks, T., Britton, T., and Höhle, M. (2020). Model selection and parameter estimation for dynamic epidemic models via iterated filtering: Application to rotavirus in Germany. *Biostatistics*, 21(3):400–416.

Wheeler, J., Rosengart, A. L., Jiang, Z., Tan, K., Treutle, N., and Ionides, E. L. (2023). Informing policy via dynamic models: Cholera in Haiti. *arXiv:2301.08979*.

Whitehouse, M., Whiteley, N., and Rimella, L. (2023). Consistent and fast inference in compartmental models of epidemics using Poisson Approximate Likelihoods. *Journal of the Royal Statistical Society, Series B*, To appear.