Using an iterated block particle filter via **spatPomp**

Ning Ning and Edward L. Ionides

December 14, 2022

The IBPF algorithm studied by Ning and Ionides (2021) and Ionides et al. (2022) has been contributed to the R package **spatPomp** (Asfaw et al., 2021a,b) as the function `ibpf`. This document introduces `ibpf` and validates its correctness on a simple Gaussian example which is tractable using the Kalman filter. We also test `ibpf` on simulated data for a measles transmission model.

In addition to the **spatPomp** code presented here, the full code to reproduce this document is available in the R Noweb (`.Rnw`) source file.

1 Correlated Gaussian random walks

Consider spatial units $1,\dots,U$ located evenly around a circle, where $\mathrm{dist}(u,\tilde{u})$ is the circle distance,

$$\mathrm{dist}(u,\tilde{u}) = \min\big(|u-\tilde{u}|, |u-\tilde{u}+U|, |u-\tilde{u}-U|\big).$$

The latent process is a U-dimensional Brownian motion $\boldsymbol{X}(t)$ having correlation that decays with distance. Specifically,

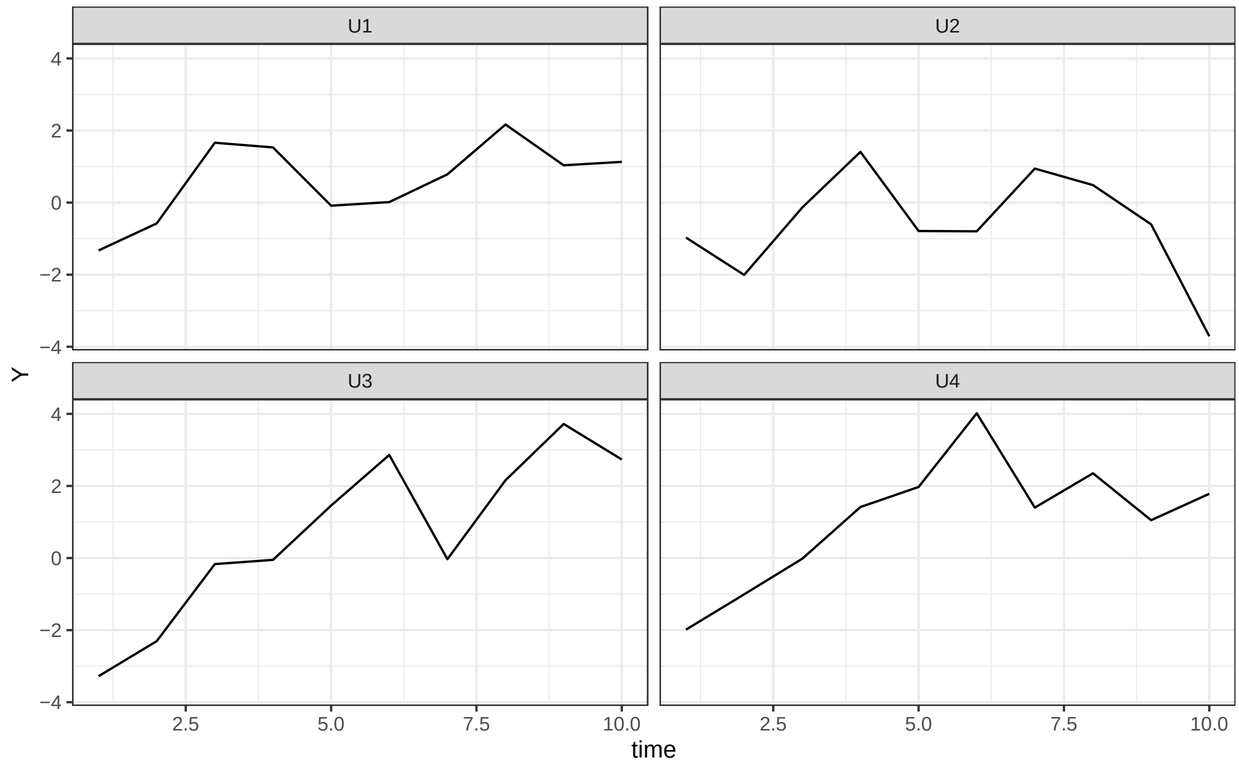
$$dX_u(t) = \sum_{\tilde{u}=1}^{U} \rho_u^{\mathrm{dist}(u,\tilde{u})}\, dW_{\tilde{u}}(t),$$

where $W_1(t),\dots,W_U(t)$ are independent Brownian motions with infinitesimal variance σ_u^2, and $|\rho_u|<1$. An observation Y_n is made at each time $t_n = n$ for $n=1,2,\dots,N$, and we write $X_n = X(t_n)$. We suppose our measurement model for discrete-time observations of the latent process is

$$Y_{u,n} = X_{u,n} + \eta_{u,n}$$

where $\eta_{u,n} \overset{\mathrm{iid}}{\sim} \mathrm{Normal}(0,\tau_u^2)$. The model is completed by providing the initial conditions, $\{X_u(0), u \in 1:U\}$, at time $t_0 = 0$. These initial conditions are specified as parameters. An instance of this model is generated below, using the `bm2` function.

```
R> library(spatPomp)
R> i <- 1
R> b <- bm2(U=4,N=switch(i,10,200),unit_specific_names="rho")
R> plot(b)
```



Here, `i` is a computational intensity switch which adjusts the code for varying run-time. We set `i=1` for testing and debugging, and `i=2` for higher quality results. For simplicity, we consider only one unit-specific parameter, ρ_u, with other parameters being fixed at a value shared between units. The simulation for `b` has $\rho_u = 0.4$ for all u, but the estimators do not know this. Before carrying out inference, we check likelihood evaluation. For this toy model, the **spatPomp** function `bm2_kalman_logLik` provides an exact log-likelihood via the Kalman filter. This study uses a sufficiently small number of units (`U=4`) that the particle filter is numerically tractable. We use the particle filter provided by the **pomp** package (King et al., 2016), taking advantage of the class structure where class '`spatPomp`' inherits from class '`pomp`'. We can readily validate the agreement between `bm2_kalman_logLik` and `pfilter`, and identify the likelihood cost of the block filter approximation in this situation.

```
R> kf_logLik <- bm2_kalman_logLik(b)
R> pf_logLik <- replicate(10,
+    logLik(pfilter(b,switch(i,10,1000)))
+  )
```

		KF	PF	BPF ($K = 1$)	BPF ($K = 2$)	BPF ($K = 4$)
Log-likelihood	mean	-72.31	-79.84	-79.17	-76.79	-76.75
	sd	0.00	2.75	4.46	3.84	2.73

Table 1: Likelihood evaluation for the `bm2` model object, `b`, using the Kalman filter (KF), particle filter (PF), and block particle filter (BPF) with varying numbers of blocks (K). For Monte Carlo filters, the mean and standard deviation are shown for 10 replicates.

```
R> bpf_logLik2 <- replicate(10,
+    logLik(bpfilter(b,switch(i,10,1000),block_size=2))
+  )
```

Table 1 shows the increasingly negative bias, and decreasing variance, of BPF as the number of blocks increases. For a single block, $K = 1$, the BPF algorithm matches PF. PF provides an unbiased estimate of the likelihood, and due to the convexity of the logarithm it has negative bias (approximately equal to half the variance) for estimating the log-likelihood. Subsequently, we investigate inference for $\rho_{1:4}$ with $K = 2$.

Ionides et al. (2021) investigated a range of values for U for this model in their Figure 1, and for an epidemiological model their Figure 3. The small scenario considered here, with $U = 4$, is designed for the following purposes: (i) to validate whether or not `ibpf` is correctly coded by comparison with direct calculations using the Kalman filter; (ii) to check whether or not the block approximation has considerable adverse effects on inference in this case. The inherent scalability of BPF and IBPF means that results for $U = 4$ are applicable to behavior on larger systems.

For our test of IBPF, we start searches at $\rho_u = u/5$ to investigate the effect (if any) on starting value.

```
R> rho_start <- seq(from=0.2,to=0.8,length=U)
R> params_start <- coef(b)
R> params_start[paste0("rho",1:U)] <- rho_start
R> ibpf_mle_searches <- foreach(reps=1:switch(i,3,10))%dopar%{
+    ibpf(b,params=params_start,
+      Nbpf=switch(i,2,50),Np=switch(i,10,1000),
+      rw.sd=rw.sd(rho1=0.02,rho2=0.02,rho3=0.02,rho4=0.02),
+      unitParNames="rho",
+      sharedParNames=NULL,
+      block_size=2,
+      spat_regression=0,
+      cooling.fraction.50=0.5
+    )
+  }
```

To assess the success of these searches, we evaluate the likelihood of the resulting parameter estimates using the Kalman filter. The highest likelihood found in these ten searches was -72.28 which is not far from the actual maximum of -70.90. However, the median of -72.34 reveals that substantial Monte Carlo maximization error is present. It can be intractable to increase computational effort to the point where the Monte Carlo error is negligible, and instead we emphasize methods that quantify and control this.

The MLE may be of less interest than marginal confidence intervals for each unit-specific parameter. Therefore, we compute a profile likelihood for each u. We calculate a profile likelihood for each value of u, using Monte Carlo adjusted profile methodology (Ionides et al., 2017; Ning et al., 2021). We compare this with an exact likelihood profile constructed by numerical optimization of the log-likelihood evaluated using the Kalman filter. The IBPF implementation is identical to the search above, except that the profiled parameter is fixed. For the profile shown in Figure 1, we first evaluate the likelihood using BPF rather than the Kalman filter, to present methodology applicable to non-Gaussian models. We then check against the likelihood evaluated via the Kalman filter for the IBPF estimates, and the profile computed directly from the Kalman filter. These reveal a distinct bias in the IBPF/BPF profile, apparently primarily to do with a bias in likelihood evaluation. The parameter in question describes a dynamic coupling between the units, and it seems that the blocking procedure breaks some of the coupling and thereby infers a higher value of the coupling parameter that used for the simulation. The bottom panel of Figure shows that we can also diagnose this effect using the particle filter, on this small example for which the particle filter is tractable.

The profile took 0.04 mins using 10 computing cores. Likelihood evaluation took 0.09, shared between BPF and PF.

We now do the same calculation for σ for comparison with ρ.

```
R> set.seed(20)
R> b_sig <- bm2(U=4,N=switch(i,10,200),unit_specific_names="sigma")
R>
R> bm2_sig_negLogLik <- function(sigma){
+      coef(b_sig,names(sigma)) <- unname(sigma)
+      -bm2_kalman_logLik(b_sig)
+    }
R>
R> stew(file=paste0(bm_dir,"kf_mle_sig.rda"),seed=256,{
+    sigma_init <- c(sigma1=1,sigma2=1,sigma3=1,sigma4=1)
+    bm2_sig_negLogLik(sigma_init)
+    bm2_sig_mle <- optim(sigma_init,bm2_sig_negLogLik)
+  })
```

The σ_1 profile took 0.04 mins using 10 computing cores. Likelihood evaluation took 0.08, shared between BPF and PF.

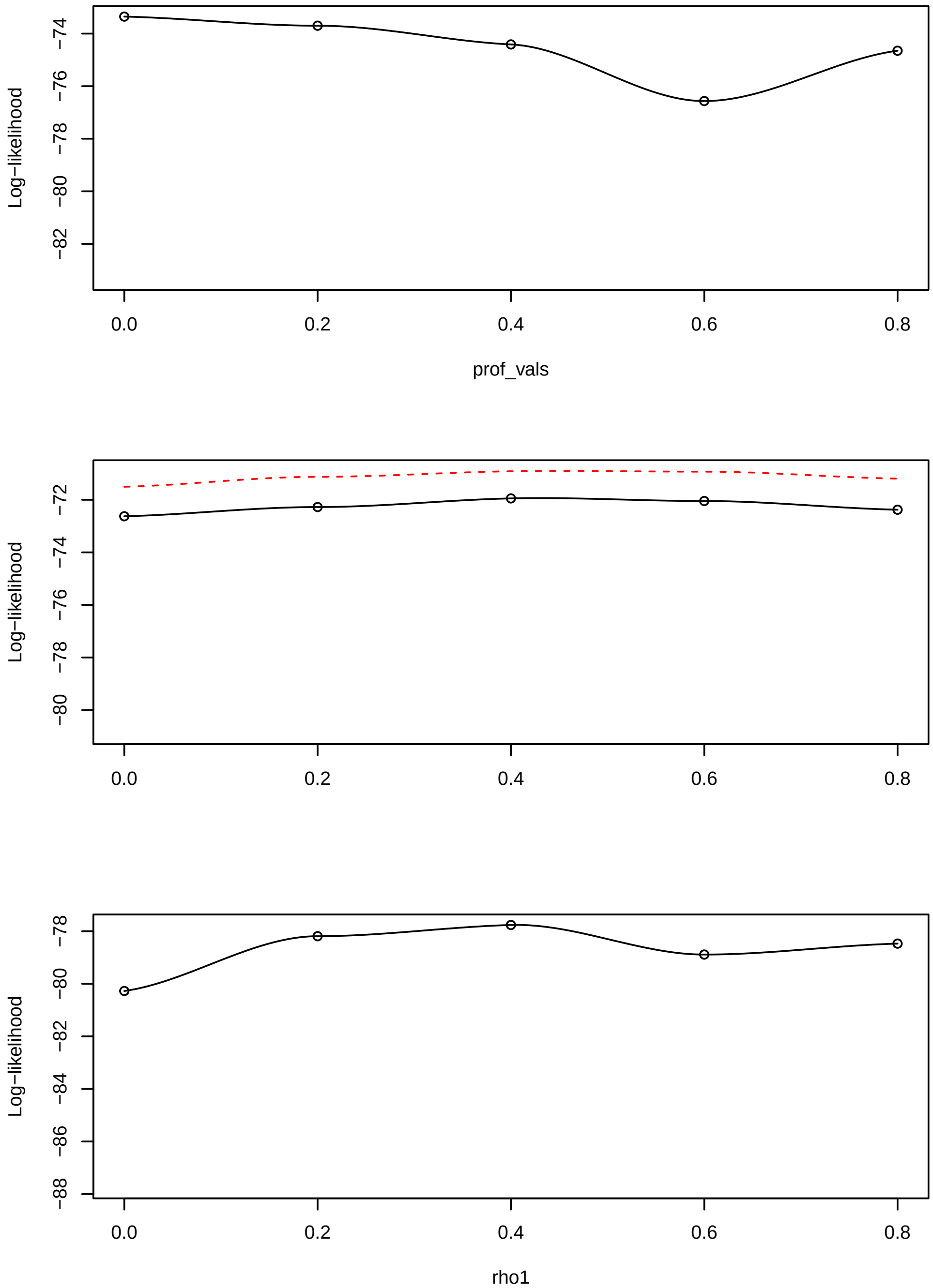


Figure 1: Top: profile for ρ_1 using an IBPF search with likelihood computed using BPF, for $K = 2$ blocks each having 2 units. Middle: Exact profile (dashed red line) and the same IBPF search with likelihood computed exactly using the Kalman filter. Bottom: The same IBPF search with likelihood computed using the particle filter.

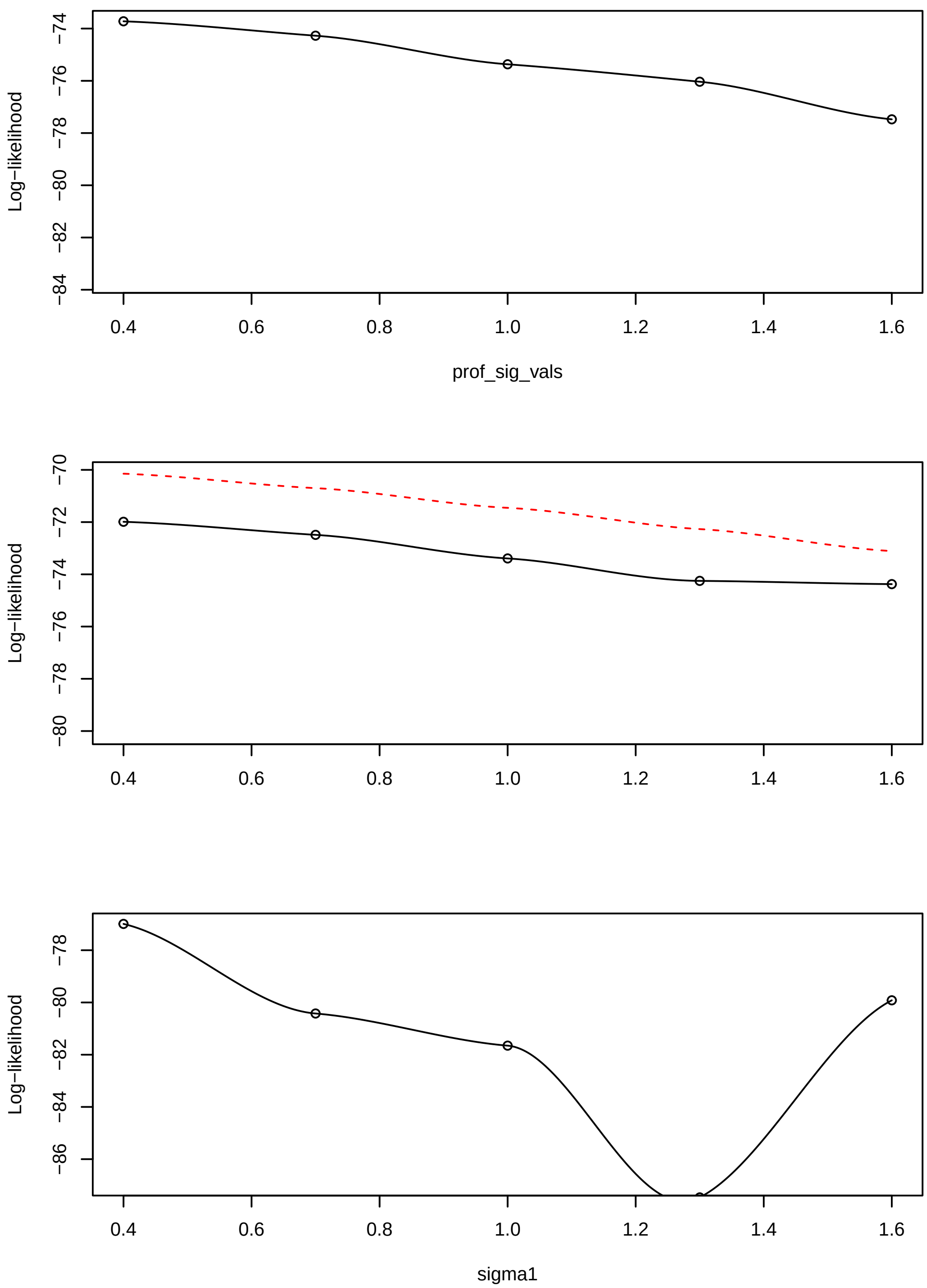


Figure 2: Top: profile for σ_1 using an IBPF search with likelihood computed using BPF, for $K = 2$ blocks each having 2 units. Middle: Exact profile (dashed red line) and the same IBPF search with likelihood computed exactly using the Kalman filter. Bottom: The same IBPF search with likelihood computed using the particle filter.

These results show that there is noticeable bias in using BPF for likelihood evaluation for this model. This bias leads to parameter estimation bias for IBPF. The same methods can be used to assess BPF and IBPF for a metapopulation model. We will see below that this model is well suited to BPF and IBPF. Intuitively, this may be because the coupling between units is weak—population movement between towns is critical to disease dynamics, but the vast majority of transmission occurs among residents of the same town.

2 A measles model

We now proceed to carry out a similar analysis for the measles model generated by `he10`. This is a susceptible-exposed-infected-recovered model for measles transmission, described by Ionides et al. (2022) and Asfaw et al. (2021b). For this model, exact likelihood evaluation is not available. However, for a relatively small number of units ($U = 4$) the particle filter provide an adequate approximation. Ionides et al. (2022) considered fitting this model to data using IBPF, with 20 cities and up to 20×13 parameters. Here, our task is to focus on a smaller, simulated dataset, estimating fewer parameters in order to assess more clearly whether or not the block approximation is leading to substantial bias. We choose two large towns (London and Birmingham) and two small towns (Cardiff and Hastings) since we expect that population movement from large towns to small towns is essential to explain disase persistence in small towns. Large towns can maintain an ongoing epidemic, but, below a critical community size local extinction of the disease is expected during epidemic troughs.

```
R> he10_model <- he10(U=4,dt=1/365,Tmax=switch(i,1964,1964),
+    expandedParNames=c("R0"),
+    towns_selected=c(1,2,11,12),
+    basic_params = c(
+      alpha =0.99,      iota=0,          R0=30,
+      cohort=0.5,  amplitude=0.3,     gamma=52,
+      sigma=52,           mu=0.02,  sigmaSE=0.05,
+      rho=0.5,           psi=0.1,         g=800,
+      S_0=0.036,         E_0=0.00007,   I_0=0.00006
+    )
+  )
R> m_seed <- 27
R> # plot(simulate(he10_model,seed=1))
R> m <- simulate(he10_model,seed=m_seed)
```

Likelihood evaluation took 0.08 mins. Recall that the bias-variance trade-off for likelihood evaluation becomes a tradeoff between two sources of bias for log-likelihood evaluation, due to Jensen's inqeuality. Table 2 shows that little likelihood is lost due to the block approximation for small numbers of units. Indeed, even for a large number of particles, a small block size gives higher log-likelihood estimates. This is on contrast to the results in Table 1 for the spatially correlated random walk example in Section 1.

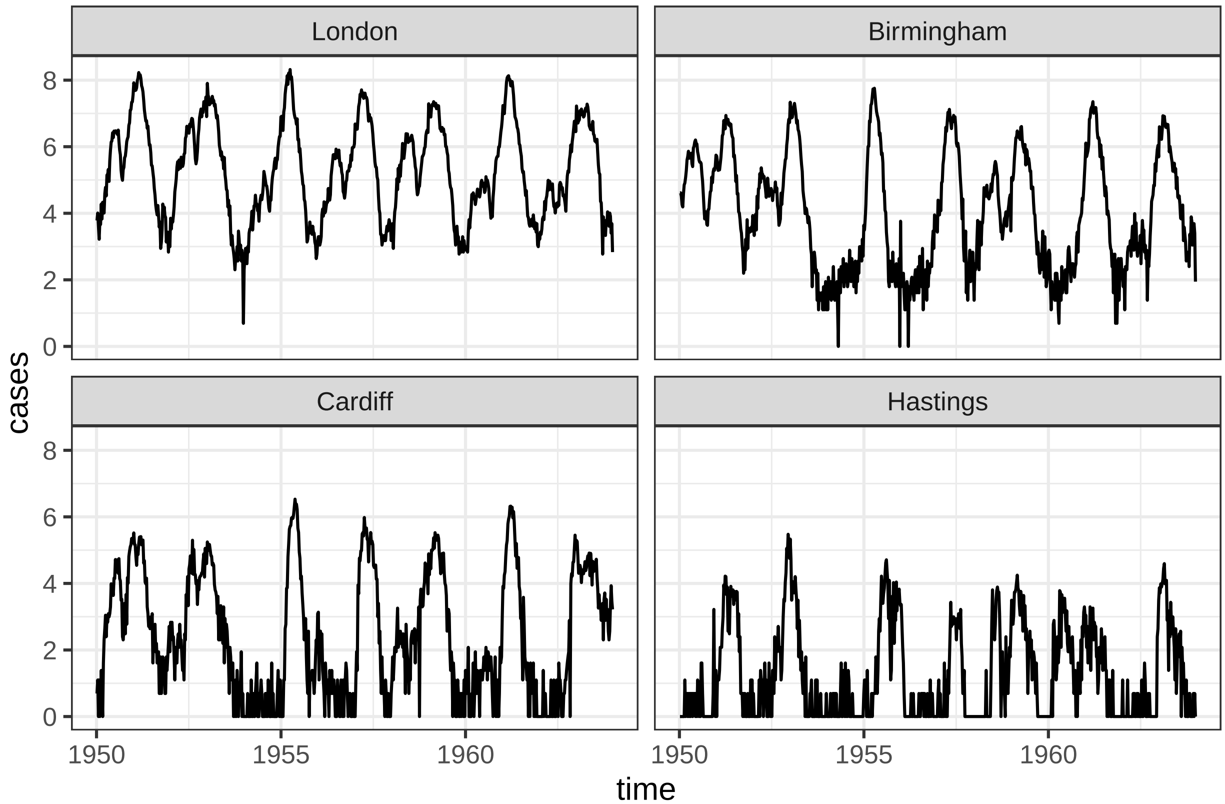


Figure 3: Weekly measles case report data for four UK towns.

		PF	BPF ($K = 1$)	BPF ($K = 2$)	BPF ($K = 4$)
Log-likelihood	mean	-209349.33	-207365.79	-92974.85	-34110.09
	sd	56864.95	45996.92	23207.28	8241.42

Table 2: Likelihood evaluation for the `he10` model object, `m`, using the particle filter (PF), and block particle filter (BPF) with varying numbers of blocks (K). The mean and standard deviation are shown for 10 replicates with 10^4 particles.

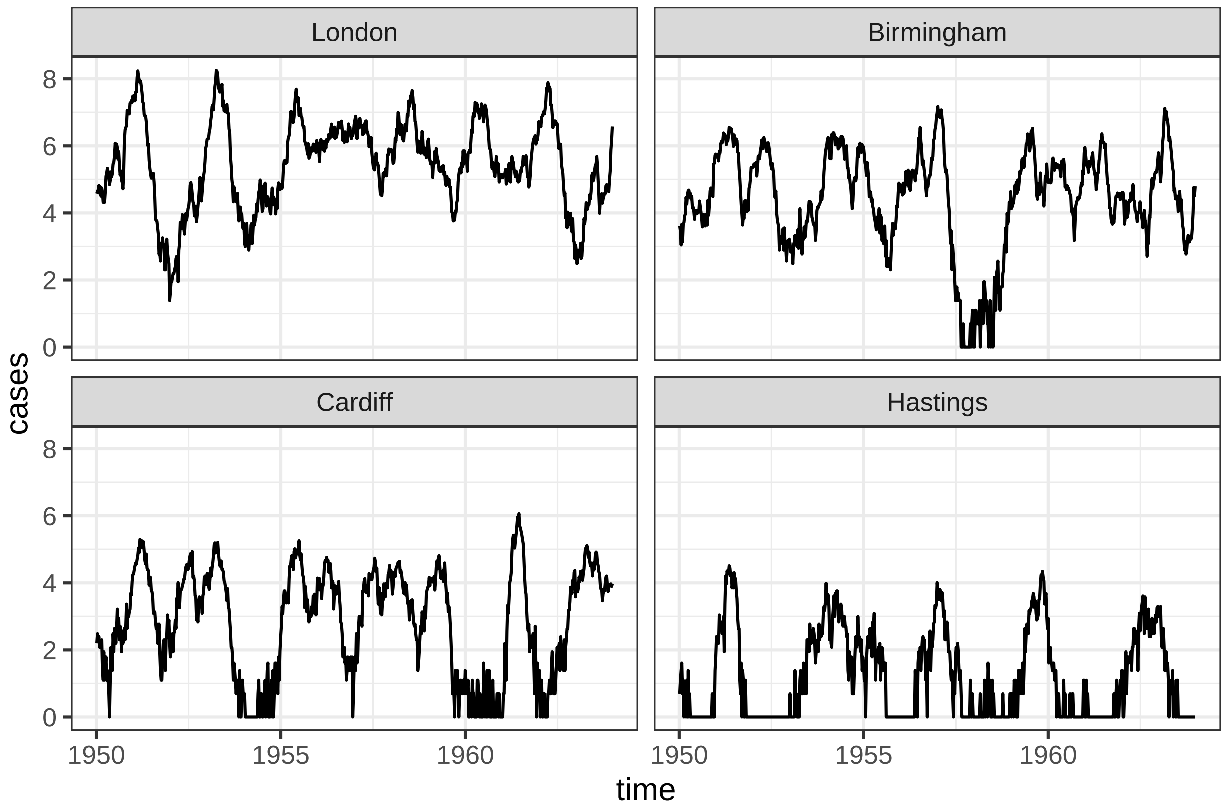


Figure 4: Simulated data from the `he10` model for four towns

Accurate BPF likelihood evalution for a small number of units suggests that the accuracy will persist for larger numbers of units. It is hard to directly test this for the measles model, since we do not have an alternative accurate evalation once PF becomes inapplicable. However, BPF has favorable scaling properties, and if the weak coupling that makes BPF accurate on the small system is also a feature of the big system, it may be reasonable to expect accuracy in situations where it cannot be directly tested.

The $R0_1$ profile took 0.28 mins using 10 computing cores. Likelihood evaluation took 0.13, shared between BPF and PF.

If IBPF is effectively maximizing the BPF approximation to the likelihood, then situations where BPF has low likelihood evaluation bias may correspond to situations where IBPF has low estimation bias. A profile likelihood for one paramter is presented to support this, in Figure 5. Here, the evaluation using PF gives a slightly tighter estimate of the profile, but this may be less accurate: PF is a higher variance algorithm, even with $U = 4$, and its variance increases as the model becomes increasingly misspecified. Thus, the profile likelihood estimate may have additional curvature due to increasing variance (and therefore increasing Jensen bias) away from the MLE.

The block approximation in BPF concerns dependence between blocks and therefore may have an effect on estimation of parameters describing the coupling between units. The measles metapopulation model has a so-called gravity model for coupling, with a parameter g_u controlling the rate of transmission from other cities into city u. A relatively straightforward way to investigate estimation of g_u using BPF and IBPF is to compute a likelihood slice through the true parameter value for a simulation, with only g_u being varied. Here, we investigate a slice for $u = 4$. A likelihood profile cannot take a lower value than a likelihood slice, since the profile has an additional optimization. Therefore, a flat slice implies a flat profile; the converse is not necessarily true. Clear evidence of bias in a slice for a small number of units would anticipate difficulties when undertaking the more time-consuming task of obtaining a profile with many units and many parameters. Figure 6 suggests that the BPF likelihood approximation leads to a low-bias estimate of g_u when compared with Figures 1 and 2. CHECK THAT THIS MAKES SENSE IN THE CONTEXT OF THE FINAL RESULTS.

The g_4 slice took 0.05 mins using 10 computing cores.

References

Asfaw, K., Ionides, E. L., and King, A. A. (2021a). `spatPomp`: R package for statistical inference for spatiotemporal partially observed Markov processes. `https://cran.r-project.org/web/packages/spatPomp`.

Asfaw, K., Park, J., Ho, A., King, A. A., and Ionides, E. L. (2021b). Statistical inference for spatiotemporal partially observed Markov processes via the R package spatpomp. *arXiv:2101.01157*.

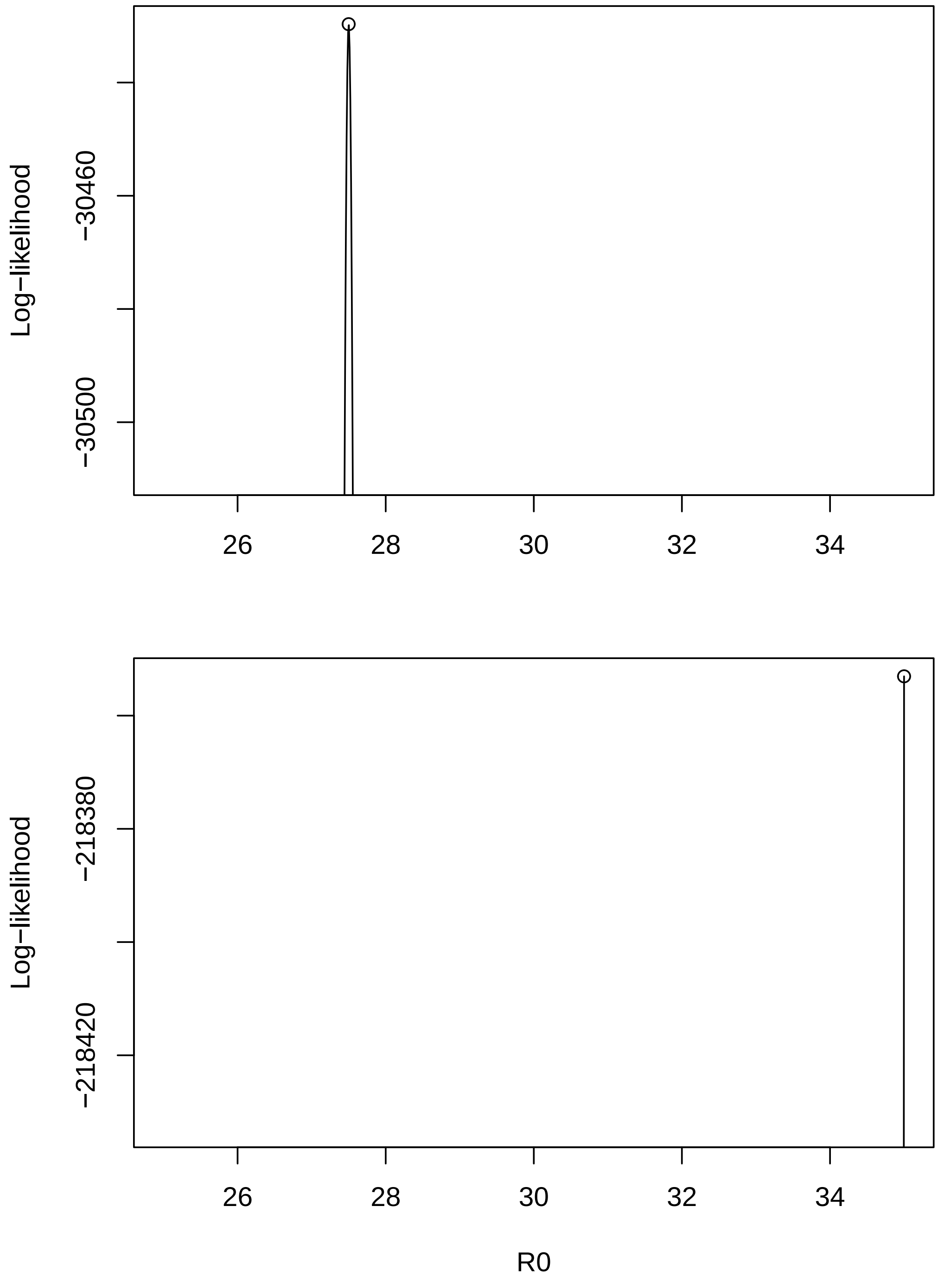


Figure 5: Top: profile for $R0_1$ using an IBPF search with likelihood computed using BPF, for $K = 4$ blocks each having 1 unit. Bottom: The same IBPF search with likelihood computed using the particle filter.

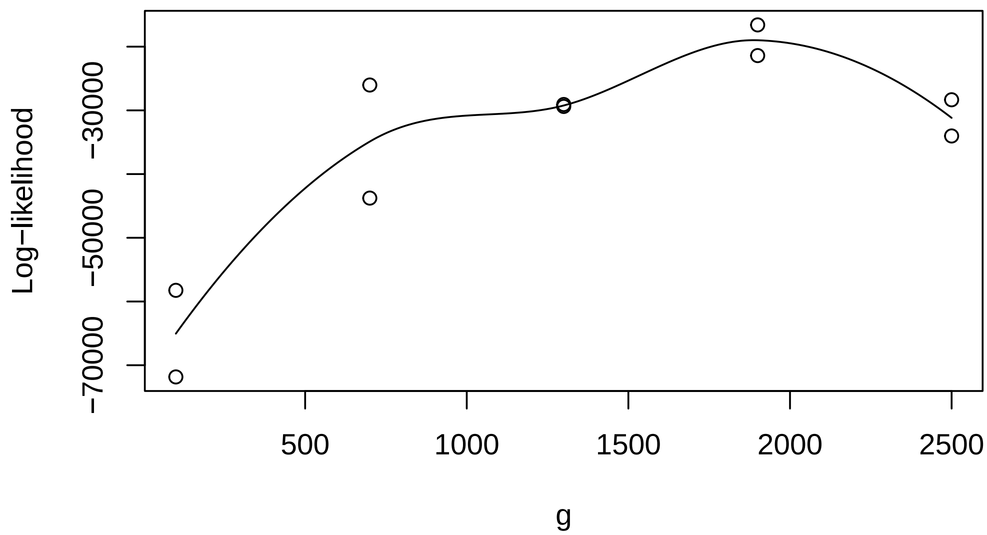


Figure 6: Slice for g_4 through the true parameter vector, using BPF evaluation with $K = 4$ blocks each having 1 unit.

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., 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., 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, pre-published online*.

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.

Ning, N. and Ionides, E. L. (2021). Iterated block particle filter for high-dimensional parameter learning: Beating the curse of dimensionality. *arXiv:2110.10745*.

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