

## State-Space Models

In State-Space Modelling, we consider two random variables:  $\{X_t\}_{t \geq 0}$ , the latent variable, this is a Markov process which we do not observe.  $\{Y_t\}_{t \geq 1}$  the observed variable, this is dependent on the Markov process. This is formalised as follows,

$$\begin{aligned} X_0 &\sim \mu(x_0), \\ X_t | (X_{t-1} = x_{t-1}) &\sim f(x_t | x_{t-1}, \theta), \\ Y_t | (X_t = x_t) &\sim g(y_t | x_t, \theta). \end{aligned} \quad (1)$$

This leads to two major questions,

- **Filtering:** What is the latent variable at current time given as new observations are made,  $p(x_T | y_{1:T}, \theta)$ ?
- **Parameter Inference:** Can we make inference on the parameter values  $\theta$ ,  $\pi(\theta)$ ?

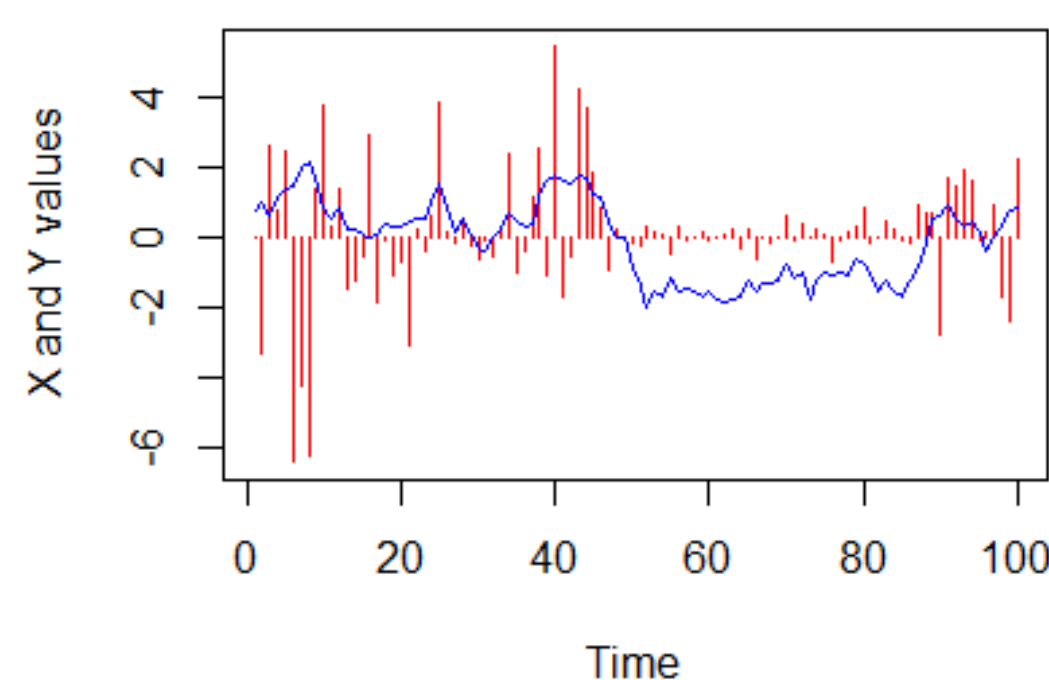
These problems are mostly intractable, thus we use **Monte Carlo** methods to make inferences on these distributions.

## Stochastic Volatility

One area which we can apply this is within finance, where we model the log-volatility as a latent Markov process  $X$  and log-returns as  $Y$  such that,

$$\begin{aligned} X_t | (X_{t-1} = x_{t-1}) &\sim \mathcal{N}(\gamma x_t, \sigma^2), \\ Y_t | (X_t = x_t) &\sim \mathcal{N}(0, \exp(x_t)). \end{aligned} \quad (2)$$

Below is a simulation over 100 time periods. Red represents log-returns and blue represents log-volatility.



## Bootstrap Filter

One way to approach **filtering** is with an algorithm known as bootstrap filter. This is described as follows,

1. **Initialise** particles by sampling  $x_0^{(i)} \sim \mu(x_0)$ ,
2. Set  $t \leftarrow 1$ ,
3. **Propagate** particles by sampling  $\tilde{x}_t^{(i)} \sim f(\tilde{x}_t | x_{t-1}^{(i)})$ ,
4. **Assign weights** by calculating  $\tilde{w}_t^{(i)} = g(y_t | x_t^{(i)})$  and normalise to  $w_t^{(i)} \propto \tilde{w}_t^{(i)}$  where  $\sum_i w_t^{(i)} = 1$ ,
5. **Resample** the particles with probability  $w_t^{(i)}$  corresponding to particle  $\tilde{x}_t^{(i)}$  to obtain  $x_t^{(i)}$ ,
6. Set  $t \leftarrow t + 1$  and repeat until  $t > T$  after this step.

## Dangers of Resampling

In the bootstrap filter, (multinomial) resampling can be wasteful when weights are nearly uniform. For example, if  $\{w_t^{(i)}\}_{i=1}^n = 1/n$ , approximately **37% of particles are lost**. A simple solution is to monitor the effective sample size (ESS) and resample only when it falls below a threshold.

## Particle MCMC

To approach the **parameter inference** problem, we use method known as the particle marginal Metropolis-Hastings (PMMH) algorithm, which we can use to obtain  $p(x_{1:T}, \theta | y_{1:T})$ .

Define  $u$  as an auxiliary random variables generated by a **particle filter**  $q_*(u | \theta)$  which contains particles  $x_{1:T}^{1:n}$  and ancestor indices from resampling  $a_{1:T}^{1:n}$ . Each iteration of the PMMH algorithm can be described as follows,

1. Set parameters from previous iterations,  
 $\theta \leftarrow \theta_{i-1}, u \leftarrow u_{i-1}, x_{1:T} \leftarrow x_{i-1,1:T}$
2. Propose a new parameter  $\theta'$  from proposal  $q(\theta' | \theta)$ ,
3. Sample  $u'$  from  $q(u' | \theta')$  and estimate  $\hat{\pi}(\theta'; u')$ ,
4. Sample  $k'$  from  $q(k' | u', \theta')$ , find the backwards path associated with  $x_T^{(k')}$  denote the full path  $x'_{1:T}$ .
5. Calculate acceptance probability,

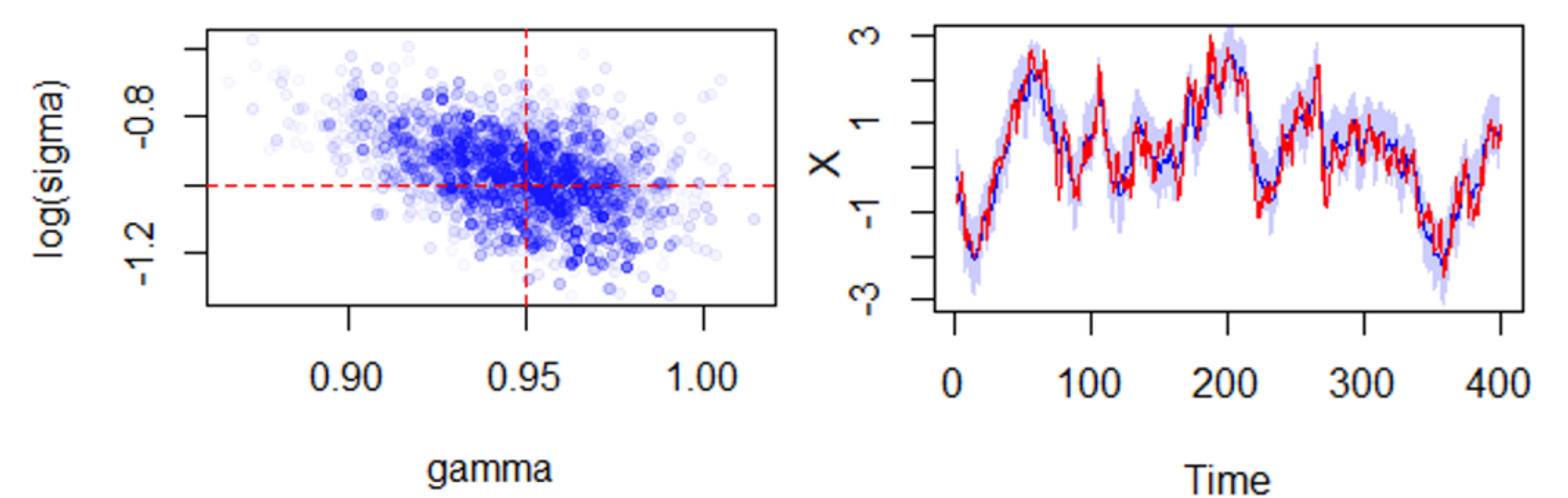
$$\alpha(\theta, u; \theta', u') = 1 \wedge \frac{\hat{\pi}(\theta'; u') q(\theta | \theta')}{\hat{\pi}(\theta; u) q(\theta' | \theta)},$$

6. Generate  $U \sim \text{Unif}(0, 1)$ . **If**  $U \leq \alpha$ , we **accept** the move: Set  $\theta_i \leftarrow \theta', u_i \leftarrow u', x_{i,1:T} \leftarrow x'_{1:T}$ . **Else**, we **reject** the move: Set  $\theta_i \leftarrow \theta, u_i \leftarrow u, x_{i,1:T} \leftarrow x_{1:T}$ .

## Empirical Experiment

We test the PMMH algorithm on the stochastic volatility model with parameters  $\theta = \{\gamma, \log \sigma\} = \{0.95, -1\}$ . We set  $\theta_0 = \{0.5, 0\}$ , used **random walk proposal**  $q(\theta' | \theta) \sim \mathcal{N}(\theta, hI)$  with  $h = 0.005$ . Each particle filter runs with  $n = 500$  particles, and we ran this over 10,000 iterations.

After removing burn-in samples  $b = 100$ , we obtain the following plots. On the left is the joint likelihood for  $p(\theta | y_{1:T})$ . On the right plot is the latent variables posterior distribution  $p(x_{1:T} | y_{1:T})$  with red line denoting the true latent trajectory.



## Further Work

- **Filtering:** Improve propagation by considering future observed value.
- **Parameter Inference:** Explore implementing the algorithm within a Gibbs sampling framework

## References

[Fearnhead and Sherlock, 2025] Fearnhead, P. and Sherlock, C. (2025). MCMC for State Space Models. In *Handbook of Markov Chain Monte Carlo Volume 2*.