Adaptive Particle Markov Chain Monte Carlo for Jump-Diffusion Models

And a Shift to Differential Particle Filters

Michelle Ko

Supervised By
Dr. Martin Lysy

University of Waterloo

May 29, 2023

Table of Contents

Motivation

Goal: Recast asset price jump-diffusion as a state-space model to

- Recover the latent (unobserved) volatility, and
- Estimate model parameters,

based on observed asset price using a "particular" computational technique.

Example of a jump-diffusion model for asset price:

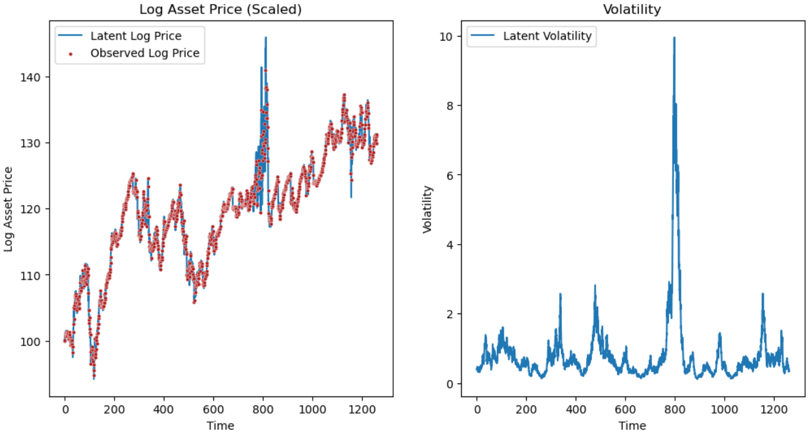
Exponential Ornstein-Uhlenbeck (ExpOU)

Log asset price: $dX_t = \alpha dt + \exp(Z_t)^{\frac{1}{2}} dW_t^x + V_t^x dN_t$

Log latent volatility: $dZ_t = \kappa(\theta - Z_t)dt + \sigma dW_t^z + V_t^z dN_t$

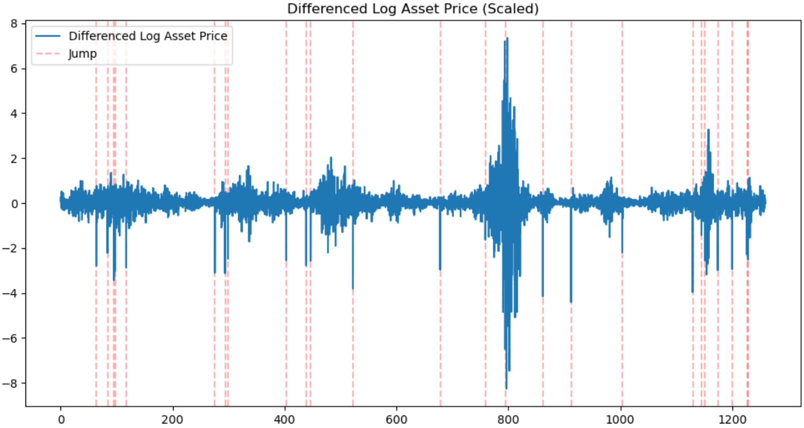
Working Example

Consider a synthetic stock's daily price over 5 years that follows an ExpOU + Jump model.



Working Example

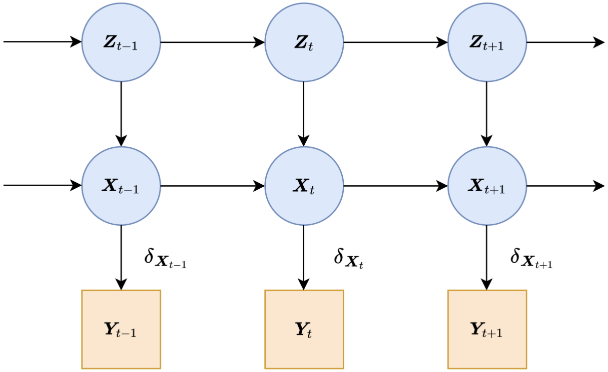
The differenced log price gives a close approximation for daily return percentage.



State-Space Representation

Volatility and log asset price, Z_t, X_t, are both continuous latent processes. Not a problem.

The observed log price, Y_t, is measured error-free $\Rightarrow$ observation density is a Delta function. This becomes a problem later.



Particle Filtering

Estimation method for the filtering distribution $p(Z_{1:T} \mid X_{1:T}, \Theta)$ [Gordon et al., 1993, Del Moral et al., 2001, Golightly and Wilkinson, 2008, Johannes et al., 2009].

1. **Propagate**: Sample from proposal $Z_t^{(i)} \sim q(Z_t \mid \tilde{Z}_{t-1}, X_t, \Theta)$
2. **Re-weight**: Calculate incremental importance weights:

$$w_t^{(i)} = \frac{p_{\text{obs}}(X_t \mid Z_t^{(i)}, \Theta) p_{\text{trans}}(Z_t^{(i)} \mid \tilde{Z}_{t-1}^{(i)}, \Theta)}{q(Z_t^{(i)} \mid \tilde{Z}_{t-1}, X_t, \Theta)} \tag{1}$$

3. **Resample**: Resample particles with probability $p_i \propto w_t^{(i)}$:

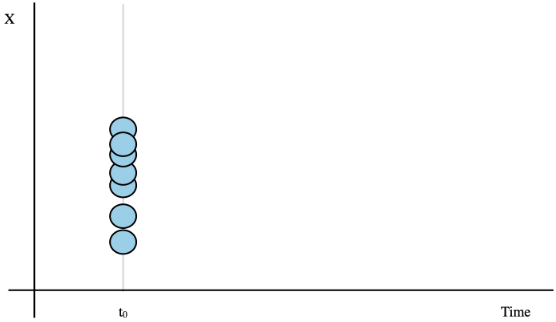
$$\tilde{Z}_t \sim Resampler(Z_t, w_t^{(i)}) \tag{2}$$

Importantly: The marginal log-likelihood of Θ given $X_{1:T}$ "can" be unbiasedly estimated,

$$\hat{\ell} = \sum_{t=1}^{T} \log(avg_i(w_t^{(i)})) \tag{3}$$

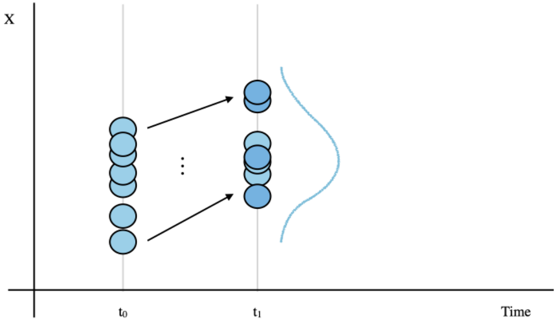
Particle Filtering Visualized

Prepare initial particles.



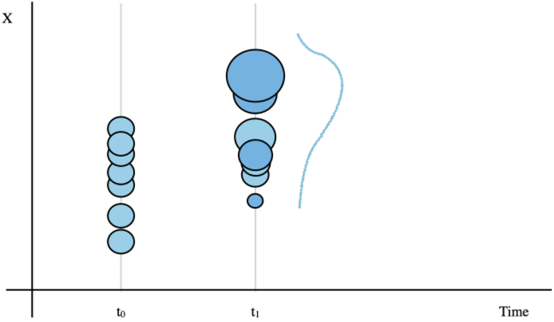
Particle Filtering Visualized

Propagate particles with transition density.



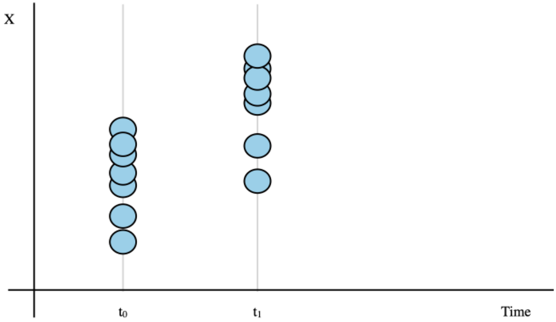
Particle Filtering Visualized

Re-weigh particles according to importance weights.



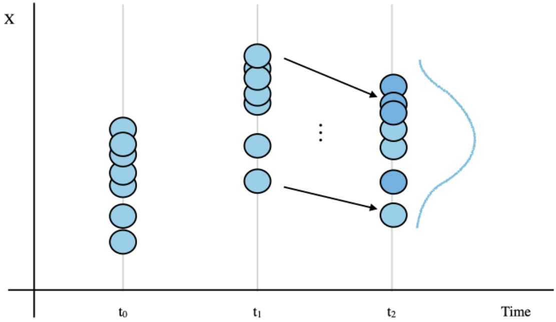
Particle Filtering Visualized

Resample particles based on importance weights.



Particle Filtering Visualized

Repeat!



Transition Density: Discretizing Jump-Diffusions

Revisiting ExpOU with Jump...

- $m-1$ inter-observations between each pair of observations
- Approximate the SDE using Euler-Maruyama scheme
- Use a Bernoulli distribution for jumps

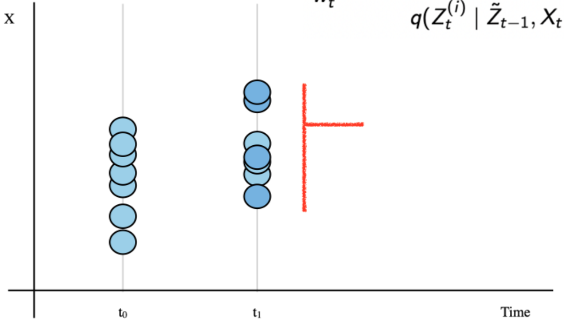
Sampling from the transition density follows as [Golightly, 2009]:

1. Sample jump $J \sim Bernoulli(\lambda \Delta t)$ and jump sizes V^x, V^z
2. Sample (X_{i+1}, Z_{i+1}) conditional on $X_i, Z_i, J, V^x, V^z, \Theta$ using:

$$\pi(Z_{i+1} \mid Z_i, \Theta) = N(Z_i + \kappa(\theta - Z_i)\Delta t + V^z_{i+1} J_{i+1}, \sigma^2 \Delta t)$$
$$\pi(X_{i+1} \mid X_i, Z_i, \Theta) = N(X_i + \alpha \Delta t + V^x_{i+1} J_{i+1}, \exp(Z_i)\Delta t)$$

But what happens with no-error measurement?

$$w_t^{(i)} = \frac{\delta(X_t) p_{\text{trans}}(Z_t^{(i)} \mid \tilde{Z}_{t-1}^{(i)}, \Theta)}{q(Z_t^{(i)} \mid \tilde{Z}_{t-1}, X_t, \Theta)}$$



Proposal Density: Diffusion Bridge with Jumps

How to deal with this?

- Create a bridge conditioned on the next observation $Y_{t+1} = X_{t+1}$
- This prevents all particle weights reducing to zero

The latent process in $(t_j, t_M]$ is proposed [Golightly, 2009]: For $i = j, \ldots, M-1$, first simulate M jumps and jump sizes. Then simulate Z^*_{i+1} recursively from its transition density. Then draw:

$$X^*_{i+1} \sim N(X^*_i + \frac{x_M - X^*_i}{M-i} + V^*_{i+1} J^*_{i+1} - \frac{\sum_{k=i+1}^{M} V^{x*}_k J^*_k}{M-i}, \\ \frac{M-i-1}{M-i} Z^*_i \Delta t)$$

Takeaway

We represent jump-diffusion model as a state-space model to:

- Use a particle filter.

We build a particle filter to:

- Recover the latent states given observation,
- Integrate over the latent states to obtain the marginal log-likelihood, which is almost always analytically intractable.

We construct a diffusion bridge to:

- Do the above even when observations are error-free.

Particle MCMC

Particle Markov Chain Monte Carlo $\Rightarrow$ When particle filtering is used within Markov chain Monte Carlo for drawing the posterior of the parameters [Andrieu et al., 2010]

- **Particle Marginal Metropolis-Hastings (PMMH)**:
  Metropolis-Hastings that uses the marginal likelihood estimated by particle filtering in the acceptance ratio
- **Particle Gibbs (PG)**:
  Alternate sampling between parameter and latent state using particle filters, i.e. draw parameter, draw a path based on parameter drawn, draw parameter based on path drawn...

Other options are Particle Metropolis-within-Gibbs, SMC^2, etc.

Adaptive PMCMC

PMCMC algorithms usually have tuning parameters:

- **PMMH**:
  The step size σ_{rw} if the parameter proposal is a random walk
- **Particle Gibbs (PG)**:
  Dependent on how parameter conditioned on path is sampled

With methods proposed in [Roberts and Rosenthal, 2009], parameter tuning can be automatically done based on target acceptance rates, empirical covariance structure, etc.

Preliminary Results

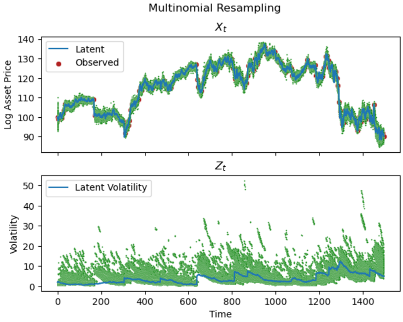
With a synthetic dataset generated by Heston + Jump model, Adaptive Particle Gibbs (APG) ran for around 9 minutes with:

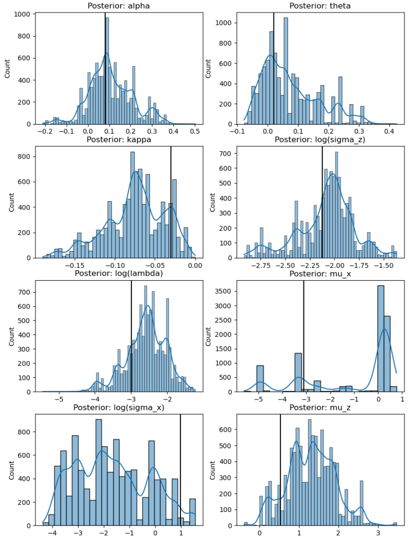
- Number of observations: 300
- Resolution: 5
- Number of particles: 100
- $\Rightarrow$ 150,000 operations per one marginal log-likelihood evaluation
- Number of MCMC iterations: 20,000

Obstacles:

- Extremely low acceptance rate on some parameters
- Computationally expensive to obtain plausible posterior

Preliminary Results





New Focus on Differentiable Particle Filters

Why do we want this?

- If the PF-estimated marginal log-likelihood is differentiable, it turns into an optimization problem
- Gradient-based methods can be used, Autograd already well-implemented JAX, a high performance Python package [Bradbury et al., 2018]

How can we get this?

1. Resampling method in the particle filter is differentiable
2. Random variables in the model have differentiable densities

1. Differentiable Resampler

Multinomial resampling is one of the traditional methods in particle filtering for the resampling step, i.e. choosing particle "ancestors":

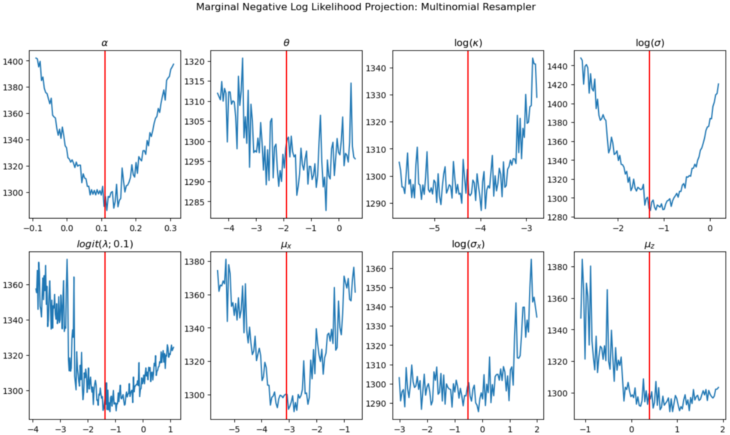
- $Multinomial(\{w_t^{(i)}\})$
- Unbiased but NOT differentiable!

Consider instead a Gaussian approximation of the weighted particle distribution:

- $N(mean(\{Z_t^{(i)}\}, \{w_t^{(i)}\}), var(\{Z_t^{(i)}\}, \{w_t^{(i)}\}))$
- May be biased in some cases but differentiable!

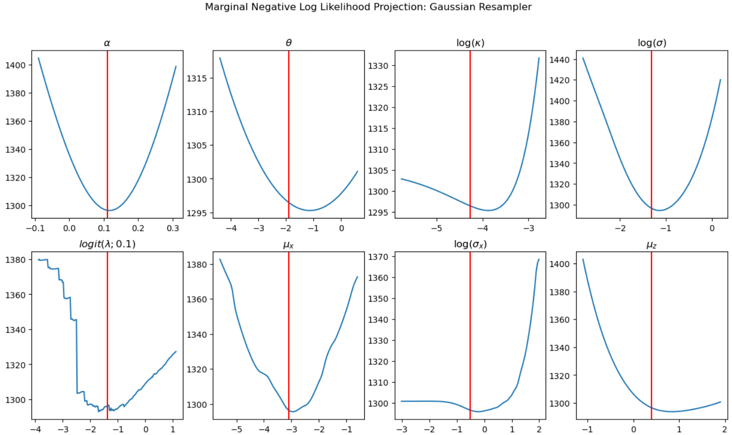
1. Differentiable Resampler

When particles are resampled with Multinomial distribution:



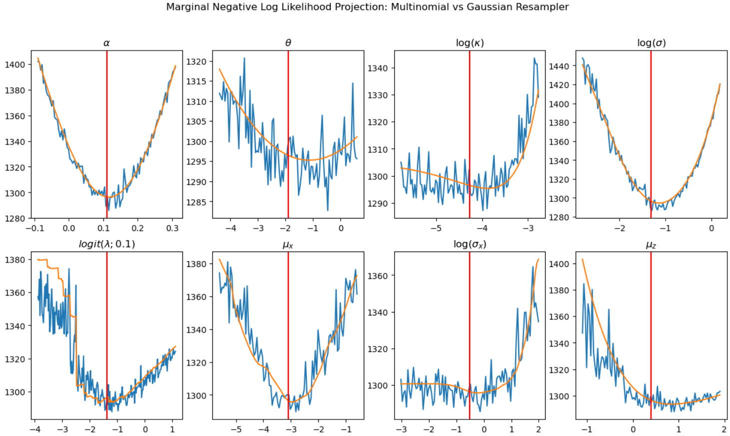
1. Differentiable Resampler

When particles are resampled with Gaussian approximation:

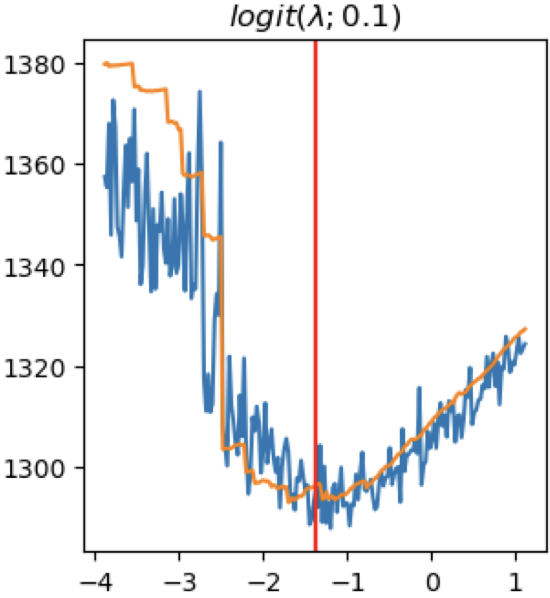


1. Differentiable Resampler

The two together:



Why still jagged?



2. Differentiable Densities

Jump occurrence is a Bernoulli random variable:

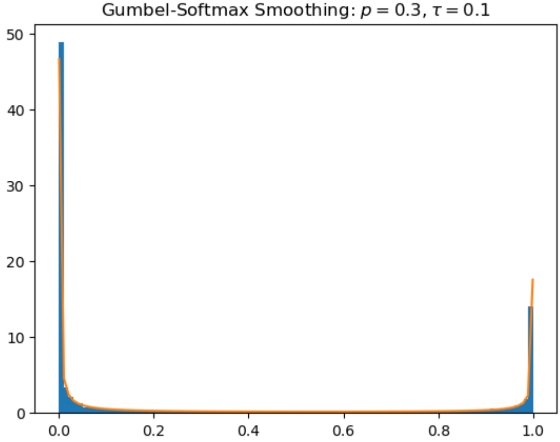
- Definitely NOT differentiable!
- Explains the jagged-ness of the marginal negative log-likelihood

We can employ a reparameterization trick:

- Gumbel-Softmax—comes with a tuning parameter τ [Jang et al., 2017]
- As $\tau \to 0$, the distribution becomes Bernoulli
- Adds bias but differentiable!

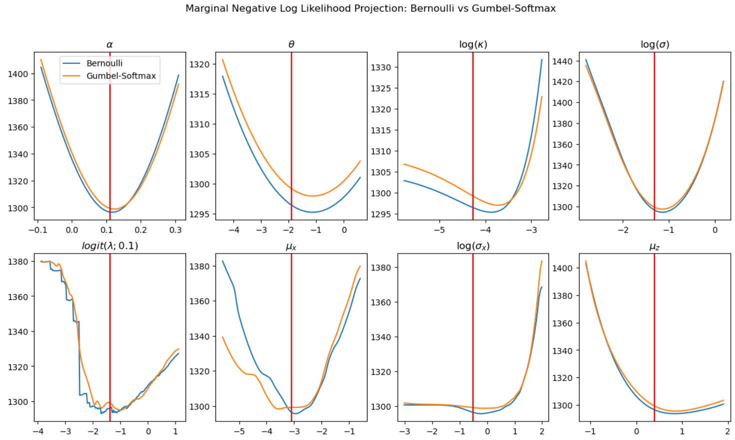
2. Differentiable Densities

Distribution of $(1 + \exp((L + \log \frac{1-p}{p})\tau^{-1}))^{-1}$, $L \sim \textit{logistic}(0, 1)$



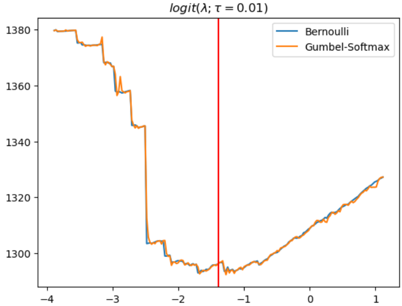
2. Differentiable Densities

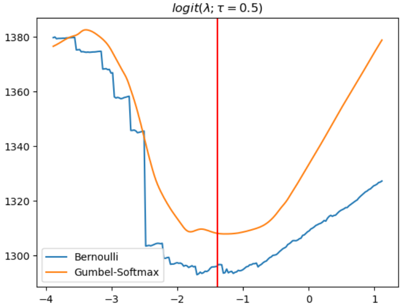
With $\tau = 0.1$, the minima is slightly shifted in some projection plots.



2. Differentiable Densities

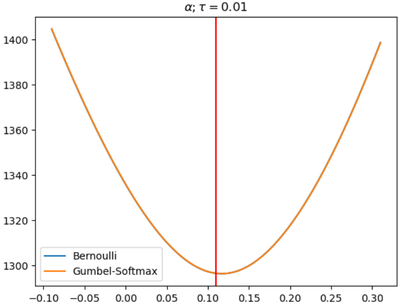
Increasing τ means smoother marginal in λ.

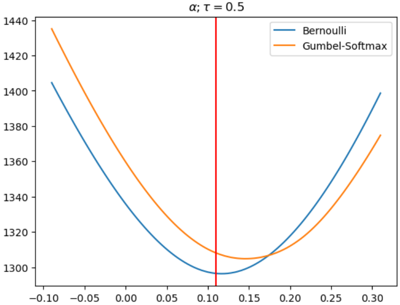




2. Differentiable Densities

Bias is prominent in α as τ increases.





Results

With a slight modification of λ, Gradient Descent ran for around 10 minutes with:

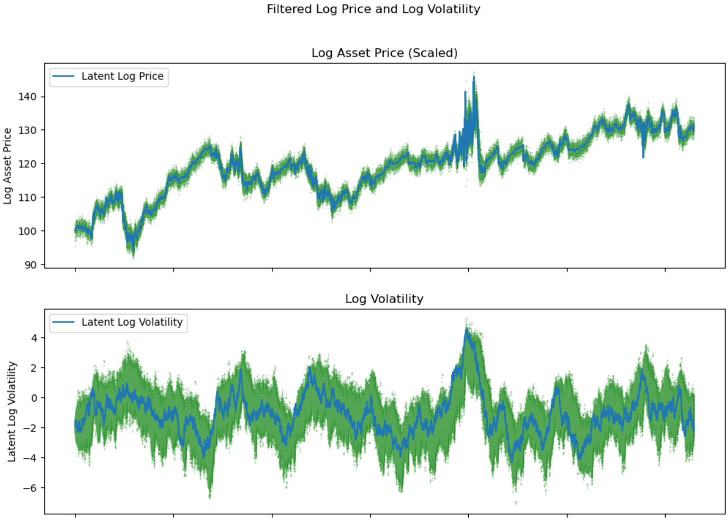
- Number of observations: $252 \times 3 = 756$
- Resolution: 5
- Number of particles: 200
- $\Rightarrow$ 756,000 operations for one marginal log-likelihood evaluation

Parameter	α	θ	κ	σ
True	0.11	-1.9	0.014	0.27
Estimated	0.18	-2.75	0.017	0.22

Parameter	λ	μ_x	σ_x	μ_z
True	0.050	-3.1	0.60	0.64
Estimated	0.063	-3.75	1.43	0.59

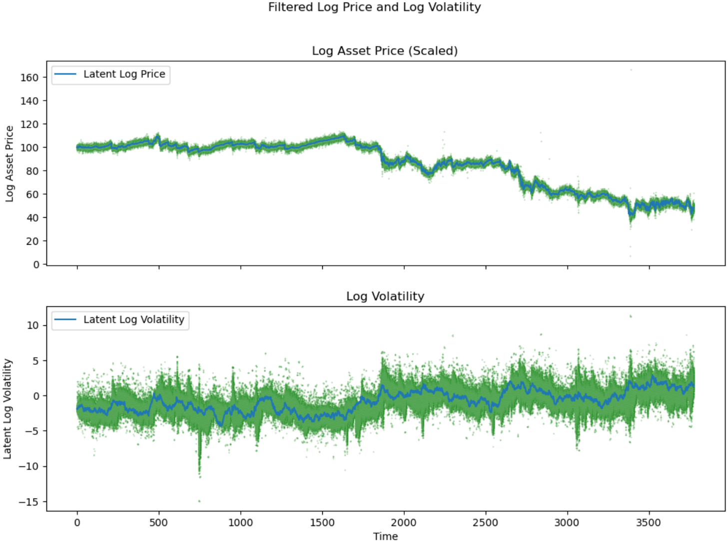
Filtered Latent States

With true parameters:



Filtered Latent States

With parameter estimates obtained from Reparameterized model:



Discussion

Cases that did not work well:

- Small λ: Jumps are too rare
- Small τ: Not enough smoothing
- Small μ_x, μ_z: Jump sizes are negligent
- Other factors: Wrong model specification, "bad seed", etc.

Areas of further investigation:

- Tradeoff between τ and bias: Bias correction?
- Distribution of $\hat{\Theta}$: Run optimizer with different seeds?
- Real life data: Daily SP 500 data?
- Correlated processes: Between X_t, Z_t or between jump sizes?
- Portfolio: Multiple assets that follow jump diffusion?

Acknowledgement

The PFJAX team:

- Martin Lysy
- Jonathan Ramkissoon
- Pranav Subramani
- Mohan Wu
- Kanika Chopra

References I

Andrieu, C., Doucet, A., and Holenstein, R. (2010).
Particle markov chain monte carlo methods.
Journal of the Royal Statistical Society: Series B (Statistical Methodology), 72(3):269–342.

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., and Zhang, Q. (2018).
Jax: composable transformations of Python+NumPy programs.

Del Moral, P., Jacod, J., and Protter, P. (2001).
The monte-carlo method for filtering with discrete-time observations.
Probability Theory and Related Fields, 120:346–368.

References II

Golightly, A. (2009).
Bayesian filtering for jump-diffusions with application to stochastic volatility.
Journal of Computational and Graphical Statistics, 18:384–400.

Golightly, A. and Wilkinson, D. (2008).
Bayesian inference for nonlinear multivariate diffusion models observed with error.
Computational Statistics and Data Analysis, 52:1674–1693.

Gordon, N., Salmond, D., and Smith, A. (1993).
Novel approach to nonlinear/non-gaussian bayesian state estimation.
IEE Proceedings-F, 140:107–113.

Jang, E., Gu, S., and Poole, B. (2017).
Categorical reparameterization with gumbel-softmax.

References III

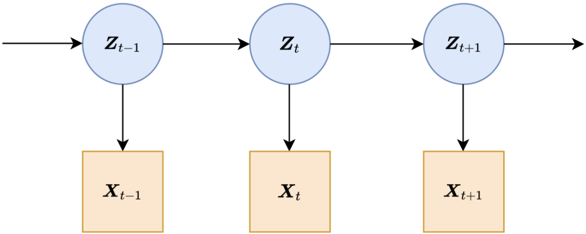
Johannes, M., Polson, N., and Stroud, J. (2009).
Optimal filtering of jump diffusions: Extracting latent states from asset prices.
Review of Financial Studies, 22:2759–2799.

Roberts, G. and Rosenthal, J. (2009).
Examples of adaptive mcmc.
Journal of Computational and Graphical Statistics, 18:349–367.

Background

State-space model is specified by:

- Latent state with transition density: $p_{\text{trans}}(Z_t \mid Z_{t-1}, \Theta)$
- Observation density: $p_{\text{obs}}(X_t \mid Z_t, \Theta)$



We are interested in the estimation of the model parameter Θ and latent state $Z_{1:T}$.