Estimating Multivariate Stochastic Volatility Models with Particle MCMC

Taylor R. Brown PhD

Lecturer
Department of Statistics
University of Virginia

March 31, 2019

Overview

Motivation: The Big Picture

Problem Description

Want to forecast stock returns.

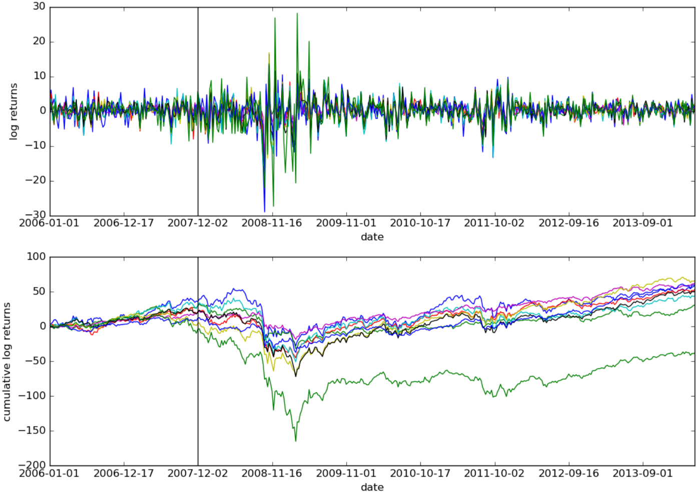
Want to estimate the sequence of forecast/posterior predictive distributions,

$$p(y_{t+1} \mid y_{1:t}),$$

where $y_{1:t} = \{y_i\}_{i=1}^{t}$ are the vector-valued **rates of return.**

Our Data

Select Sector SPDR ETFs: weekly log-returns $y_t \in \mathbb{R}^9$



Practical Considerations

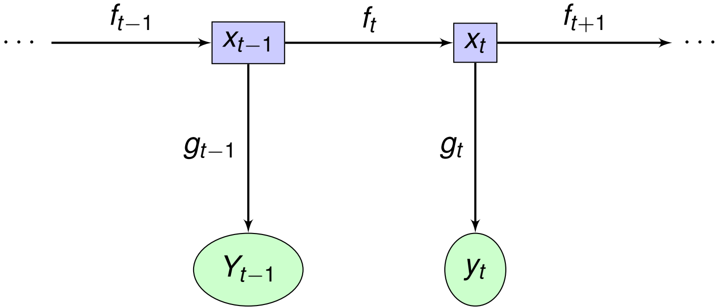
Stylized Features:

1. y_t is high-dimensional
2. $E[y_{t+1} \mid y_{1:t}]$ is "subtle"
3. $\mathrm{Var}[y_{t+1} \mid y_{1:t}]$ is a less-subtle matrix-valued stochastic process

Difficulties:

1. model estimation for general state space models can be computationally tricky
2. real-time forecasting can be tricky

Definition



Particle filtering yields:

1. Recursive formulas for $p(x_t \mid y_{1:t}, \theta)$
2. Estimates of the likelihood $\hat{p}(y_{1:T} \mid \theta)$

Particle Filtering Animation

A small example:

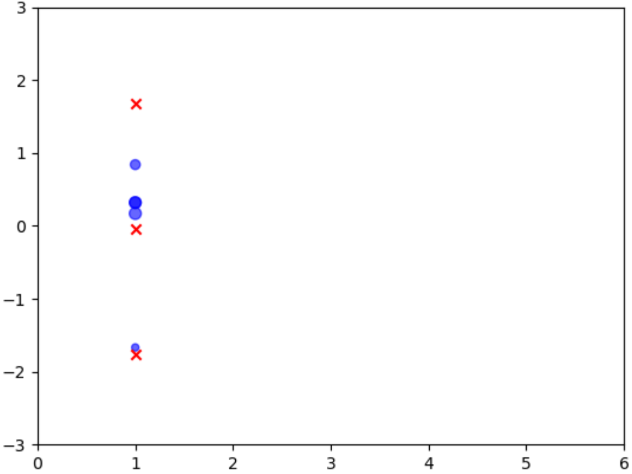
$$y_t = \exp(x_t/2)z_{1t} \tag{1}$$

$$x_t = c + \phi x_{t-1} + \sigma z_{2t} \tag{2}$$

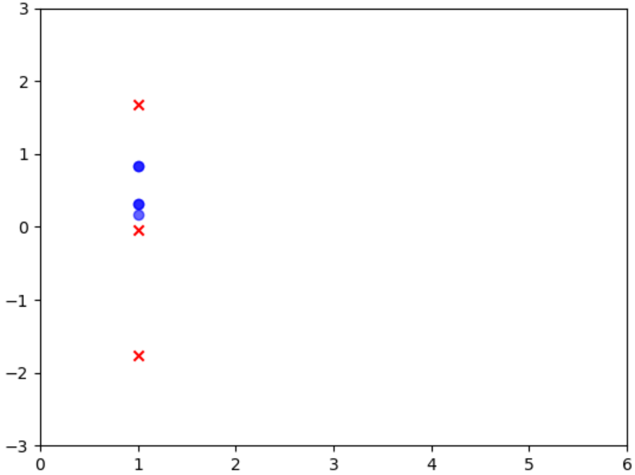
Turn samples for $p(x_{t-1} \mid y_{1:t-1})$ into samples for $p(x_t \mid y_{1:t})$.

1. propogate/mutate/extend the samples
2. assign weights
3. resample/select “good” samples and discard the “bad”

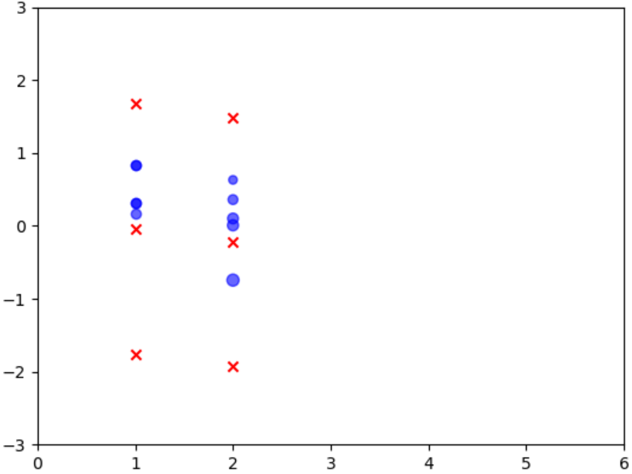
Particle Filtering Animation
$p(x_t \mid y_{1:t}, \theta)$ samples over time t



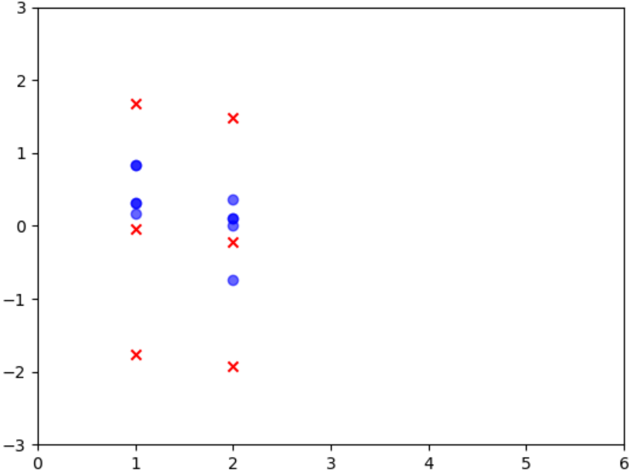
Particle Filtering Animation
$p(x_t \mid y_{1:t}, \theta)$ samples over time t



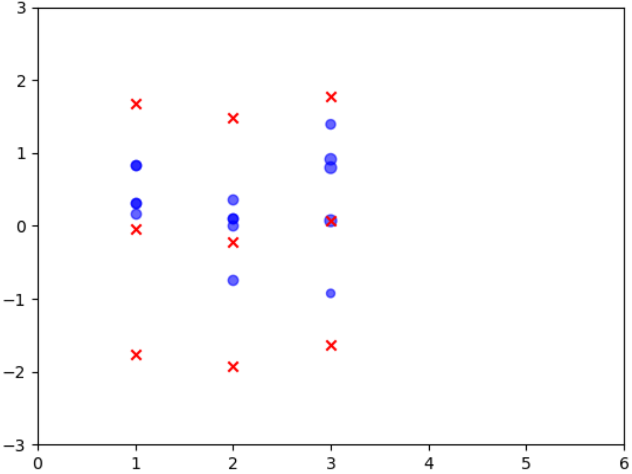
Particle Filtering Animation
$p(x_t \mid y_{1:t}, \theta)$ samples over time t



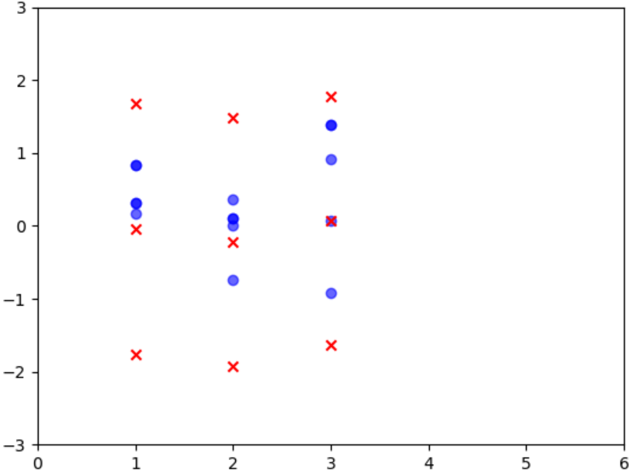
Particle Filtering Animation
$p(x_t \mid y_{1:t}, \theta)$ samples over time t



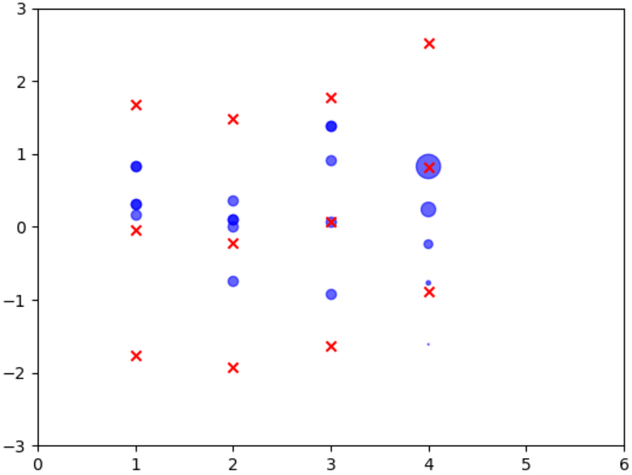
Particle Filtering Animation
$p(x_t \mid y_{1:t}, \theta)$ samples over time t



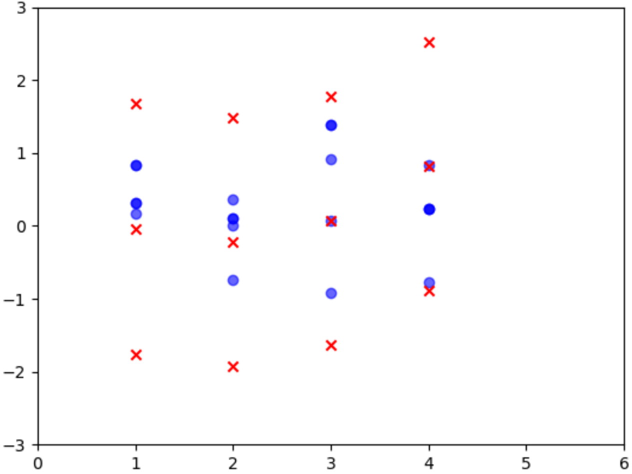
Particle Filtering Animation
$p(x_t \mid y_{1:t}, \theta)$ samples over time t



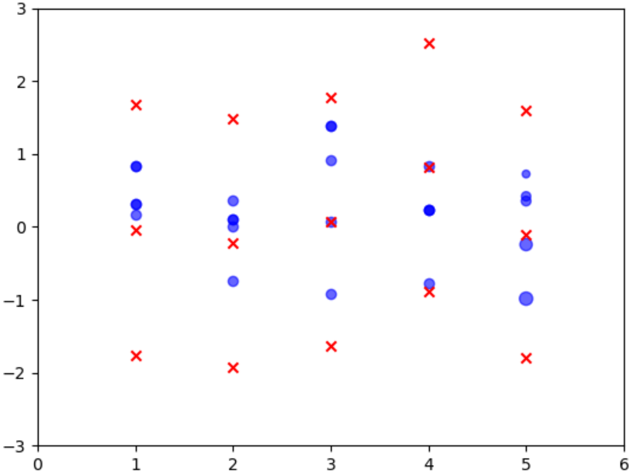
Particle Filtering Animation
$p(x_t \mid y_{1:t}, \theta)$ samples over time t



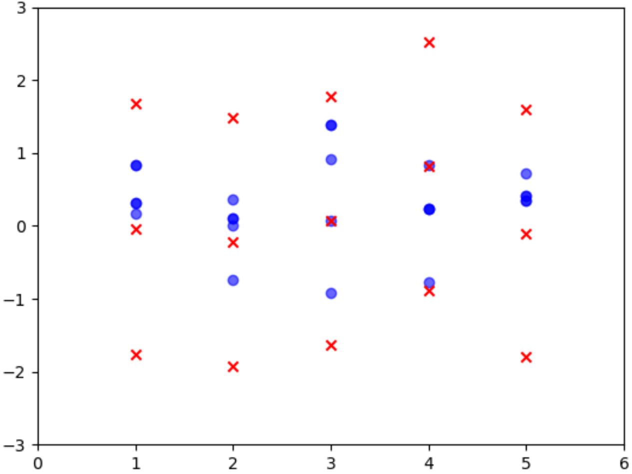
Particle Filtering Animation
$p(x_t \mid y_{1:t}, \theta)$ samples over time t



Particle Filtering Animation
$p(x_t \mid y_{1:t}, \theta)$ samples over time t



Particle Filtering Animation
$p(x_t \mid y_{1:t}, \theta)$ samples over time t



Batch Bayesian Parameter Estimation

Markov chain Monte Carlo

Goal: draw samples from

Target

$$
\begin{aligned}
& p(\theta \mid y_{1:T}) \\
& \propto \underbrace{p(y_{1:T} \mid \theta)}_{\text{intractable!}} p(\theta) \\
& = \int \underbrace{p(y_{1:T} \mid x_{1:T}, \theta) p(x_{1:T} \mid \theta)}_{\text{model's complete data likelihood}} p(\theta) dx_{1:T}
\end{aligned}
$$

From Marginal MH to Pseudo-Marginal MH

The marginal Metropolis-Hastings' acceptance ratio is

$$\frac{p(y_{1:T} \mid \theta')p(\theta')}{p(y_{1:T} \mid \theta)p(\theta)} \frac{q(\theta \mid \theta')}{q(\theta' \mid \theta)},$$

and the **pseudo-marginal Metropolis-Hastings** acceptance ratio is

$$\frac{\hat{p}(y_{1:T} \mid \theta')p(\theta')}{\hat{p}(y_{1:T} \mid \theta)p(\theta)} \frac{q(\theta \mid \theta')}{q(\theta' \mid \theta)}.$$

Particle Marginal Metropolis-Hastings

Rewrite it a little differently:

$$\frac{\overbrace{\hat{p}(y_{1:T} \mid u', \theta')p(\theta')\psi(u'|y_{1:T}, \theta')}^{\text{unnormalized target}}}{\hat{p}(y_{1:T}|u, \theta)p(\theta)\psi(u|y_{1:T}, \theta)} \underbrace{\frac{\psi(u|y_{1:T}, \theta)q(\theta|\theta')}{\psi(u'|y_{1:T}, \theta')q(\theta'|\theta)}}_{\text{proposal distribution}}$$

- The full normalized target is $p(\theta, u \mid y_{1:T}) = \frac{\hat{p}(y_{1:T}|u,\theta)\psi(u|y_{1:T},\theta)p(\theta|y_{1:T})}{p(y_{1:T}|\theta)}$
- When likelihood is unbiased, the marginal is $p(\theta \mid y_{1:T})$
- for us, u is the output generated by a particle filter (samples and ancestor indices)

Particle MCMC: The Catch

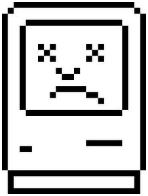
More particles

$\Longrightarrow$ increased accuracy of $\hat{p}(y_{1:T} \mid \theta')$

$\Longrightarrow$ faster MCMC algorithm (Andrieu and Vihola, 2016), but...

Particle MCMC: The Catch

More particles
$\implies$ increased accuracy of $\hat{p}(y_{1:T} \mid \theta')$
$\implies$ faster MCMC algorithm (Andrieu and Vihola, 2016), but...
$\implies$ more computing time



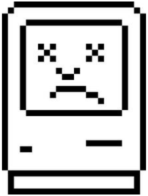
Particle MCMC: The Catch

More particles

$\Longrightarrow$ increased accuracy of $\hat{p}(y_{1:T} \mid \theta')$

$\Longrightarrow$ faster MCMC algorithm (Andrieu and Vihola, 2016), but...

$\Longrightarrow$ more computing time



1. Doucet et al., 2015 recommend increasing the number of particles until the std. dev. of the log-likelihood is around 1.2, but this can still be slow
2. We still have the classic problem of tuning $q(\theta' \mid \theta)$

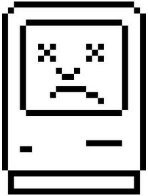
Particle MCMC: The Catch

More particles

$\Longrightarrow$ increased accuracy of $\hat{p}(y_{1:T} \mid \theta')$

$\Longrightarrow$ faster MCMC algorithm (Andrieu and Vihola, 2016), but...

$\Longrightarrow$ more computing time



1. Doucet et al., 2015 recommend increasing the number of particles until the std. dev. of the log-likelihood is around 1.2, but this can still be slow
2. We still have the classic problem of tuning $q(\theta' \mid \theta)$

What do we do?

Trick 1: faster code

Use faster code:

1. https://github.com/tbrown122387/pf
2. https://github.com/tbrown122387/ssme

Trick 2: Within-Chain Parallelization

Using the same parameter, run several particle filters in parallel at every iteration of the chain (Brown, 2019)

Reduced-Variance Likelihood Estimator

$$\tilde{p}(y_{1:T} \mid \theta) = n_p^{-1} \sum_{i=1}^{n_p} \hat{p}^i(y_{1:T} \mid \theta).$$

-threaded implementation in ssme

Trick 3: Rao-Blackwellization

Use a Rao-Blackwellized particle filter:

Lemma (Brown, 2019)

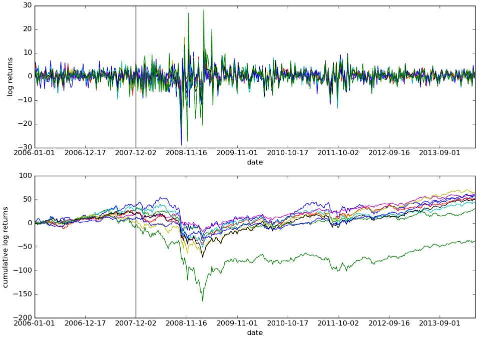
When multinomial resampling is conducted at every time point, the Rao-Blackwellized particle filter's likelihood estimate is unbiased.

-also suggested in (Andrieu and Vihola, 2016), anticipated earlier by (Jacob et al., 2009)

Factor Stochastic Volatility Models

Out-of-Sample Performance

1. Estimate $p(\theta \mid y_{1:100})$ on training data (slow, do it once)
2. Recursively approximate $p(y_{100+k+1} \mid y_{100+k})$ (fast)



A Starting Point

(Jacquier, Polson, and Rossi, 1999)

$$\mathbf{y}_t = \mathbf{B}\mathbf{f}_t + \mathbf{v}_t$$
$$\mathbf{f}_t = \exp(\mathrm{diag}[\mathbf{x}_t/2])\mathbf{z}_t$$
$$\mathbf{x}_t - \boldsymbol{\mu} = \boldsymbol{\Phi}\left[\mathbf{x}_{t-1} - \boldsymbol{\mu}\right] + \mathbf{w}_t$$

where $\{\mathbf{v}_t\} \overset{\text{iid}}{\sim} \mathrm{N}(0, \mathbf{R})$,
$\{\mathbf{z}_t\} \overset{\text{iid}}{\sim} \mathrm{N}(0, I_{d_x})$,
$\{\mathbf{w_t}\} \overset{\text{iid}}{\sim} \mathrm{N}(0, \mathrm{diag}(\sigma_1^2, \ldots, \sigma_{d_x}^2))$,
$\boldsymbol{\Phi} = \mathrm{diag}(\phi_1, \ldots, \phi_{d_x})$

For us, f_t will be univariate.

A New Model

The Markov-Switching Loadings Model (Brown, 2019)

$$\mathbf{y}_t = \mathbf{B}\mathbf{f}_{1t} + \mathbf{D}_{x_{1t}}\mathbf{B}\mathbf{f}_{2t} + \mathbf{v}_t$$

$$\begin{bmatrix} \mathbf{f}_{1t} \\ \mathbf{f}_{2t} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\lambda}_1 \\ \boldsymbol{\lambda}_2 \end{bmatrix} + \begin{bmatrix} \exp(\mathrm{diag}\{\mathbf{x}_{2t}\}) & \mathbf{0} \\ \mathbf{0} & \exp(\mathrm{diag}\{\mathbf{x}_{3t}\}) \end{bmatrix} \begin{bmatrix} \mathbf{z}_{1t} \\ \mathbf{z}_{2t} \end{bmatrix}$$

$$\begin{bmatrix} \mathbf{x}_{2,t} \\ \mathbf{x}_{3,t} \end{bmatrix} = \boldsymbol{\mu} + \boldsymbol{\Phi}\left(\begin{bmatrix} \mathbf{x}_{2,t-1} \\ \mathbf{x}_{3,t-1} \end{bmatrix} - \boldsymbol{\mu} \right) + \mathbf{w}_t$$

$$x_{1t} \in \{1, 2, \ldots, S_K\} \text{ is a discrete Markov chain}$$

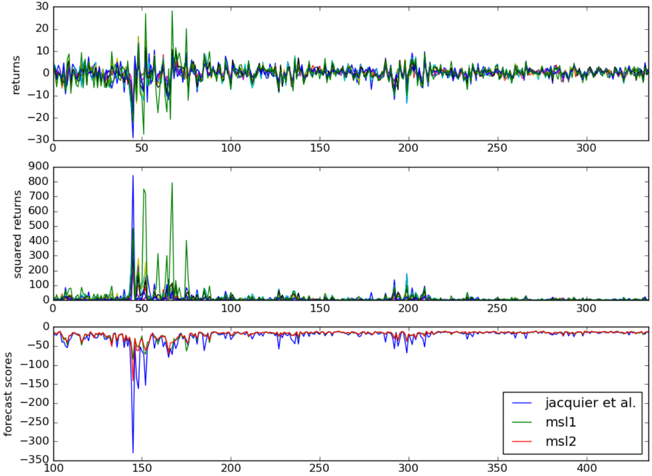
$\mathbf{D}_{x_{1t}}$ is diagonal with entries 0 or 1, and $S_K = \sum_{k=0}^{K} \binom{d_y}{k}$

-For us, f_{1t}, f_{2t} will be univariate,

-two different transition matrices for x_{1t} (e.g. MSL1 and MSL2).

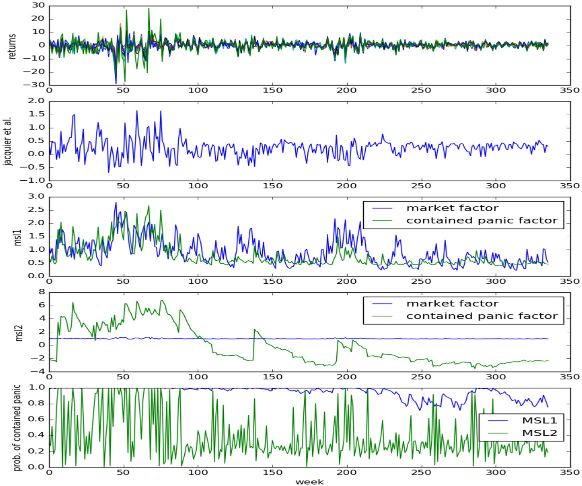
Out-Of-Sample Performance

$\hat{p}(y_{t+1} \mid y_{1:t})$ versus time and y_{t+1}...



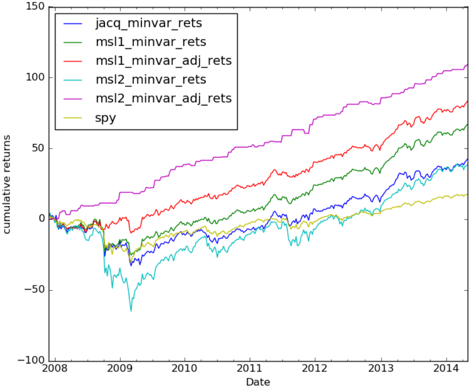
Out-Of-Sample Performance

Visualizing the hidden states...



Out-Of-Sample Performance

- rebalancing to minimum variance every week, no transaction costs or slippage
- "adj" = liquidate if forecast mean is negative (infeasible)



The End

That's it. Thanks!

Special thanks to

1. Dan Keenan (http://statistics.as.virginia.edu/faculty-staff/profile/dmk7b)
2. UVA Advanced Computing Research Services (https://arcs.virginia.edu/)

Links:

1. (preprint)https://arxiv.org/abs/1903.01841
2. (code)https://github.com/tbrown122387
3. (homepage)http://www.people.virginia.edu/~trb5me/

References I

Andrieu, Christophe and Matti Vihola (Oct. 2016). "Establishing some order amongst exact approximations of MCMCs". In: *Ann. Appl. Probab.* 26.5, pp. 2661–2696. DOI: 10.1214/15-AAP1158. URL: https://doi.org/10.1214/15-AAP1158.

Brown, Taylor R. (2019). *A Factor Stochastic Volatility Model with Markov-Switching Panic Regimes*. eprint: arXiv:1903.01841.

References II

Doucet, A. et al. (2015). “Efficient implementation of Markov chain Monte Carlo when using an unbiased likelihood estimator”. In: *Biometrika* 102.2, pp. 295–313. DOI: 10.1093/biomet/asu075. eprint: /oup/backfile/content_public/journal/biomet/102/2/10.1093_biomet_asu075/2/asu075.pdf. URL: +http://dx.doi.org/10.1093/biomet/asu075.

Jacob, P. et al. (Nov. 2009). “Comments on “Particle Markov chain Monte Carlo” by C. Andrieu, A. Doucet, and R. Hollenstein”. In: *ArXiv e-prints*. arXiv: 0911.0985 [stat.CO].

References III

Jacquier, Eric, Nicholas G. Polson, and Peter Rossi (1999). *Stochastic Volatility: Univariate and Multivariate Extensions*. Computing in Economics and Finance 1999 112. Society for Computational Economics. URL: http://EconPapers.repec.org/RePEc:sce:scecf9:112.

Kokkala, Juho and Simo Särkkä (2015). "Combining particle MCMC with Rao-Blackwellized Monte Carlo data association for parameter estimation in multiple target tracking". In: *Digital Signal Processing* 47, pp. 84–95. DOI: 10.1016/j.dsp.2015.04.004. URL: https://doi.org/10.1016/j.dsp.2015.04.004.