

# Real-Time Bayesian Forecasting with State-Space Models



Taylor R. Brown PhD

Department of Statistics  
University of Virginia

March 19, 2021

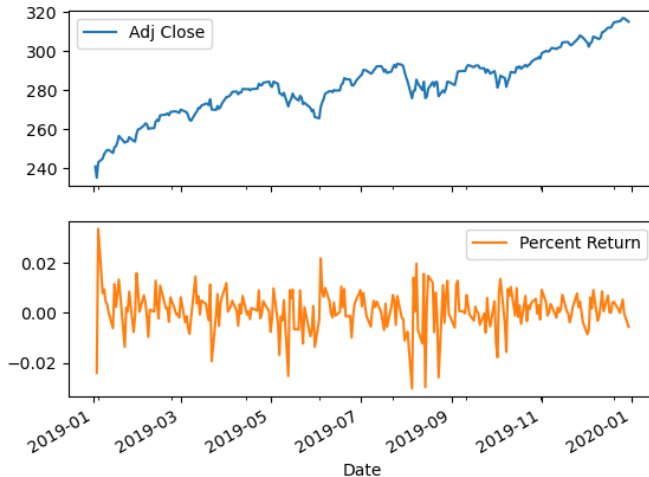
# Overview

- 1 Motivation For State-Space Models
- 2 Forecasting in a Bayesian Way
- 3 Option 2: The Two Step Approach
- 4 Option 1: On-Line Sampling of Both States and Parameters
- 5 Option 2 (With My Approach)
- 6 Numerical Experiments

## Motivation For State-Space Models

# Motivation

S&P500 Index (SPY ETF)



# Our State-Space Model

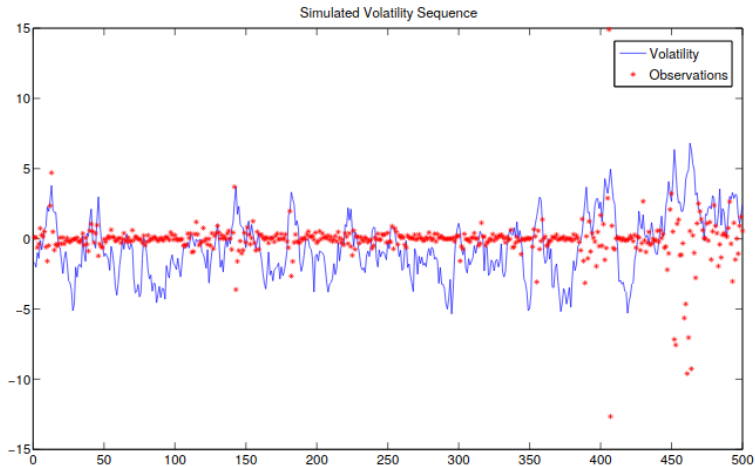
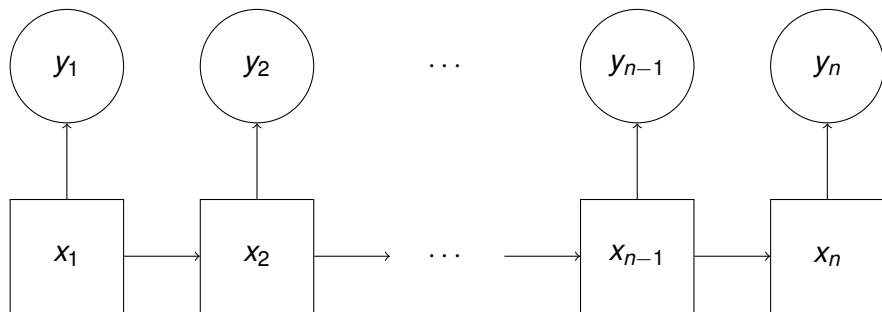


Figure: image from Doucet and Johansen, 2011

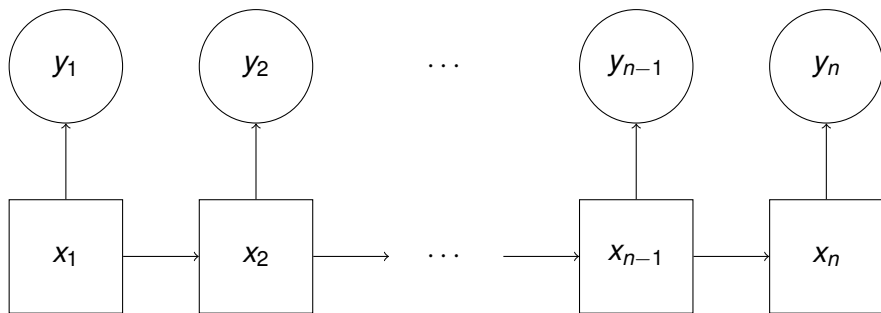
# State-Space Models



vertical arrows:  $g(y_t | x_t, \theta)$

horizontal arrows:  $f(x_t | x_{t-1}, \theta)$

# Our State-Space Model



$$g(y_t | x_t, \theta) = \text{Normal}(0, \beta^2 e^{x_t})$$

$$f(x_t | x_{t-1}, \theta) = \text{Normal}(\phi x_{t-1}, \sigma^2)$$

$x_t$  is log-volatility, represents how “active” the market is...

(Taylor, 1982)

## Forecasting in a Bayesian Way



# Our Goal

Forecasting in a Bayesian way—the posterior predictive distribution:

$$p(y_{t+1} \mid y_{1:t}) = \int p(y_{t+1} \mid \theta, y_{1:t}) p(\theta \mid y_{1:t}) d\theta$$

- $y_{1:t}$  are the percent returns
- $\theta$  is the vector of model parameters for the assumed true model
- $t \rightarrow \infty$
- can't condition on unknown parameters or unknown states
- could predict further into the future, too

# Our Goal

With state-space models, there are two general approaches to approximating

$$p(y_{t+1} \mid y_{1:t}) = \int p(y_{t+1} \mid \theta, y_{1:t}) p(\theta \mid y_{1:t}) d\theta$$

- option 1: the one step approach
- option 2: the two step approach

# Option 1: A Very Difficult Task!

We want

$$\begin{aligned} p(y_{t+1} \mid y_{1:t}) &= \iint p(y_{t+1} \mid x_t, \theta, y_{1:t}) p(x_t, \theta \mid y_{1:t}) dx_t d\theta \\ &= \iint p(y_{t+1} \mid x_t, \theta) p(x_t, \theta \mid y_{1:t}) dx_t d\theta \end{aligned}$$

- 1 Sampling both states and parameters **from one on-line algorithm**  $p(x_t, \theta \mid y_{1:t})$
- 2  $p(y_{t+1} \mid x_t, \theta)$  is easy to deal with

## Option 2: Two Difficult Tasks in One

Option 2:

$$p(y_{t+1} \mid y_{1:t}) = \int p(y_{t+1} \mid \theta, y_{1:t}) p(\theta \mid y_{1:t}) d\theta$$

- 1  $p(y_{t+1} \mid \theta, y_{1:t})$  involves filtering
- 2  $p(\theta \mid y_{1:t})$  is standard posterior inference, but now for every  $t$

We focus on this first, and briefly summarize option 1 later...

## Option 2: The Two Step Approach

## Option 2: Two Difficult Tasks in One

Option 2 was based on:

$$p(y_{t+1} \mid y_{1:t}) = \int p(y_{t+1} \mid \theta, y_{1:t}) p(\theta \mid y_{1:t}) d\theta$$

**1** filtering:

$$\begin{aligned} p(y_{t+1} \mid \theta, y_{1:t}) \\ = \int g(y_{t+1} \mid x_{t+1}, \theta) f(x_{t+1} \mid x_t, \theta) \underbrace{p(x_t \mid y_{1:t}, \theta)}_{\text{filtering distribution}} dx_{t:t+1} \end{aligned}$$

**2**  $p(\theta \mid y_{1:t})$  (posterior inference for every  $t$ )

We describe filtering first...

## A Quick Word on Filtering

$$p(x_t | y_{1:t}, \theta) = \frac{g(y_t | x_t, \theta)p(x_t | y_{1:t-1}, \theta)}{p(y_t | y_{1:t-1}, \theta)}$$

where  $p(x_t | y_{1:t-1}, \theta) = \int f(x_t | x_{t-1}, \theta)p(x_{t-1} | y_{1:t-1}, \theta)dx_{t-1}$   
comes from the previous filtering distribution.

On-line algorithms include

- 1 (Kalman, 1960) filtering for linear Gaussian models
- 2 Hidden Markov Model filter for finite state space models
- 3 approximate filtering otherwise (e.g. extended Kalman filter, unscented Kalman filter, particle filters, etc.)

# Particle Filtering (The General Strategy)

Choose a large  $N$ . Then

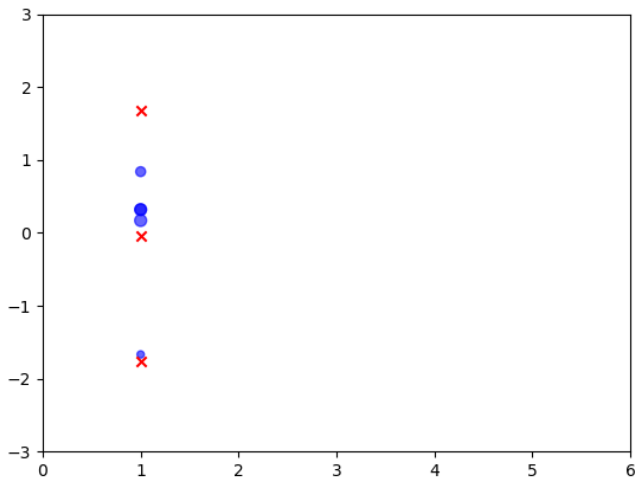
- sample  $x_1^1, \dots, x_1^N \overset{\text{approx. iid}}{\sim} p(x_1 | y_1, \theta)$
- compute weights then resample those into  $x_2^1, \dots, x_2^N \overset{\text{approx. iid}}{\sim} p(x_2 | y_{1:2}, \theta)$
- compute weights then resample those into  $x_3^1, \dots, x_3^N \overset{\text{approx. iid}}{\sim} p(x_3 | y_{1:3}, \theta)$
- etc. etc.





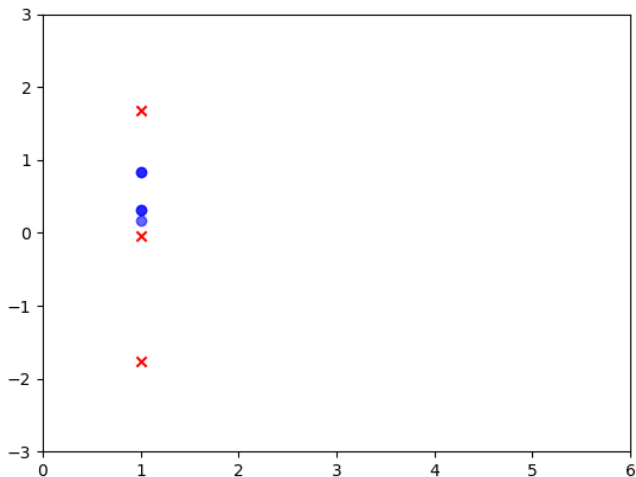
# Particle Filtering

Red xs: true mean  $\pm 2$  std. deviations



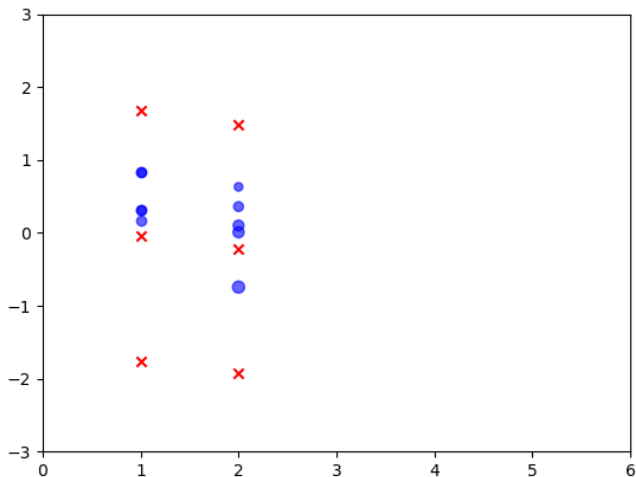
# Particle Filtering

Red xs: true mean  $\pm 2$  std. deviations



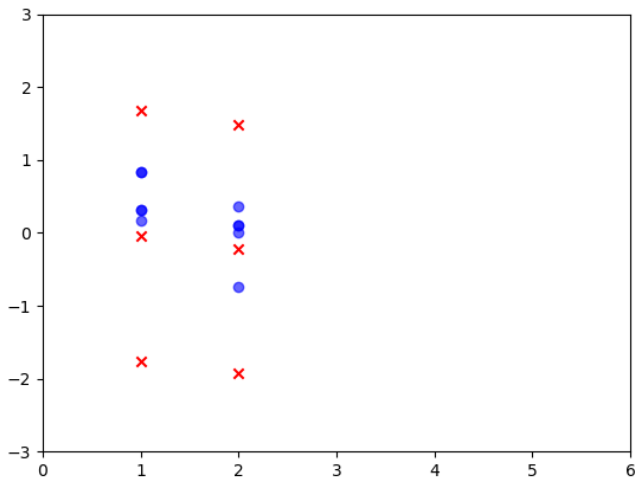
# Particle Filtering

Red xs: true mean  $\pm 2$  std. deviations



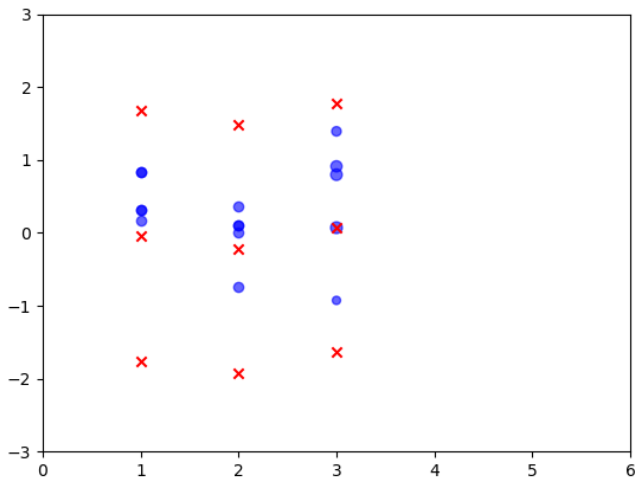
# Particle Filtering

Red xs: true mean  $\pm 2$  std. deviations



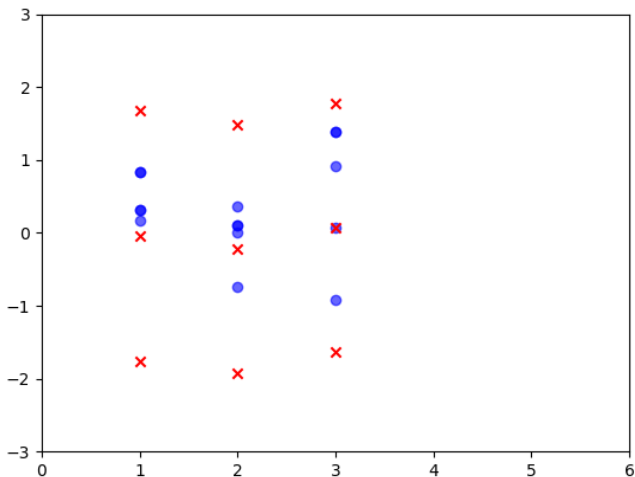
# Particle Filtering

Red xs: true mean  $\pm 2$  std. deviations



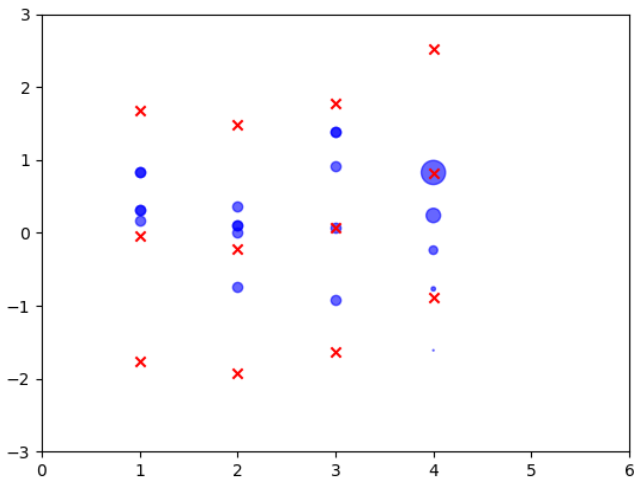
# Particle Filtering

Red xs: true mean  $\pm 2$  std. deviations



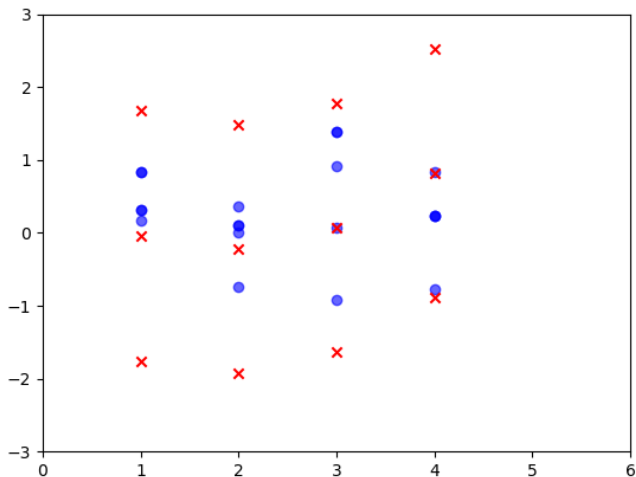
# Particle Filtering

Red xs: true mean  $\pm 2$  std. deviations



# Particle Filtering

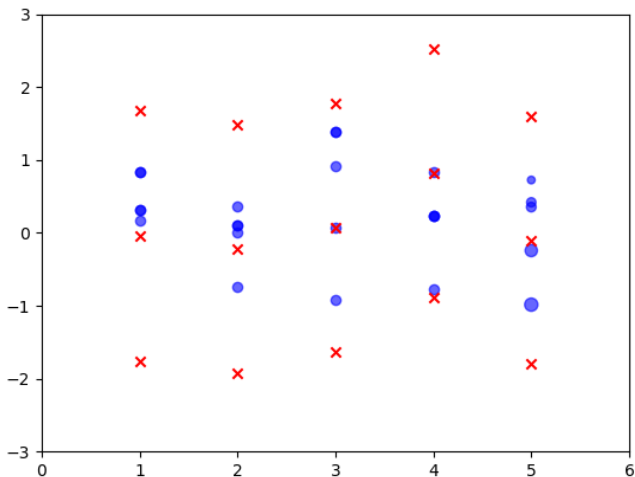
Red xs: true mean  $\pm 2$  std. deviations





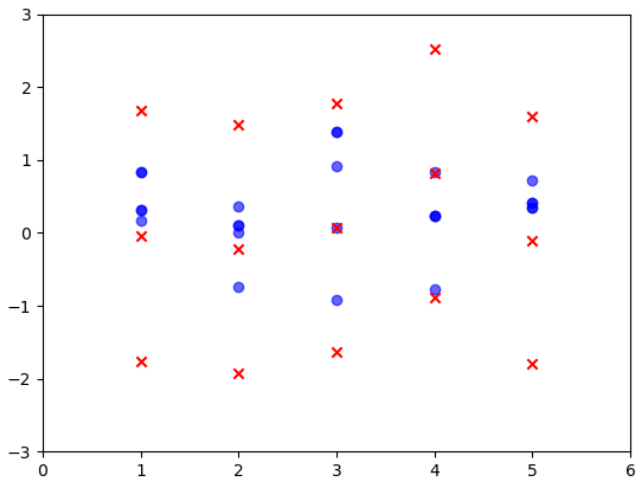
# Particle Filtering

Red xs: true mean  $\pm 2$  std. deviations



# Particle Filtering

Red xs: true mean  $\pm 2$  std. deviations



## The other half of Option 2: Posteriors for $\theta$

Option 2 was motivated by

$$p(y_{t+1} \mid y_{1:t}) = \int \overbrace{p(y_{t+1} \mid \theta, y_{1:t})}^{\text{particle filter}} \underbrace{p(\theta \mid y_{1:t})}_{?} d\theta$$

How do we use particle filtering, but incorporate uncertainty about  $\theta$ ?

## The other half of Option 2: Posteriors for $\theta$

There are many ways to sample from one posterior  $p(\theta \mid y_{1:t})$

...but they can all be either time-consuming, difficult to tune or program, or impossible to use on data sets with large  $t$ .

Is there a way to avoid having to do this at every time step on an increasingly large data set?

Even if there was, how do we use parameter samples together with a particle filter?

This is why so much work has been done on Option 1...

## Option 1: On-Line Sampling of Both States and Parameters

## Option 1: On-Line Sampling From $p(x_t, \theta \mid y_{1:t})$

This is just a quick review.

It won't describe any of these in detail, but we list out a few names...

## Option 1: On-Line Sampling From $p(x_t, \theta \mid y_{1:t})$

- 1 Run one particle filter on a model with extended state  $\tilde{x}_t = (x_t, \theta)$  (Kitagawa, 1998)
- 2 Do the above, but exploit sufficient statistics of  $p(\theta \mid x_{1:t}, y_{1:t})$  (i.e. the Storvik Filter (Storvik, 2002), (Fearnhead, 2002) and Particle Learning (Carvalho et al., 2010) )
- 3 Run one particle filter on a model with extended state  $\tilde{x}_t = (x_t, \theta_t)$ , assume  $\theta_t$  is an (artificial) random walk ( the Liu-West Filter (Liu and West, 2001) and the original proposal in (Kitagawa, 1998))
- 4 Other recursive (but not on-line) algorithms: The Resample-Move algorithm (Gilks and Berzuini, 2001) and SMC<sup>2</sup> (Chopin, Jacob, and Papaspiliopoulos, 2013)

## Option 1: On-Line Sampling From $p(x_t, \theta \mid y_{1:t})$

Option 1 approaches are all generally “fast”, but they suffer from some combination of the following:

- 1 high variance (due to **particle degeneracy**),
- 2 have a nonquantifiable bias, or they are
- 3 not on-line.

So, what do we do...



Option 2 (With My Approach)

## Back To Option 2: The Particle Swarm Filter

My approach: don't always update the posterior, and aggregate particle filters' forecasts.

$$\begin{aligned} p(y_{t+1} \mid y_{1:t}) &= \int p(y_{t+1} \mid \theta, y_{1:t}) p(\theta \mid y_{1:t}) d\theta \\ &\approx \int p(y_{t+1} \mid \theta, y_{1:t}) p(\theta \mid y_{1:s}) d\theta \end{aligned}$$

where  $0 \leq s < t$ .  $p(\theta \mid y_{1:s})$  is either a prior, or an “old” posterior that you can't afford to update every moment.

- Sample  $\{\theta^j\}_j$  **once** (or occasionally); run many particle filters with these plugged in; keep averaging predictions.

# The Particle Swarm Filter: Theoretical Guarantees

Summary of some of the theoretical results in (Brown, 2021):

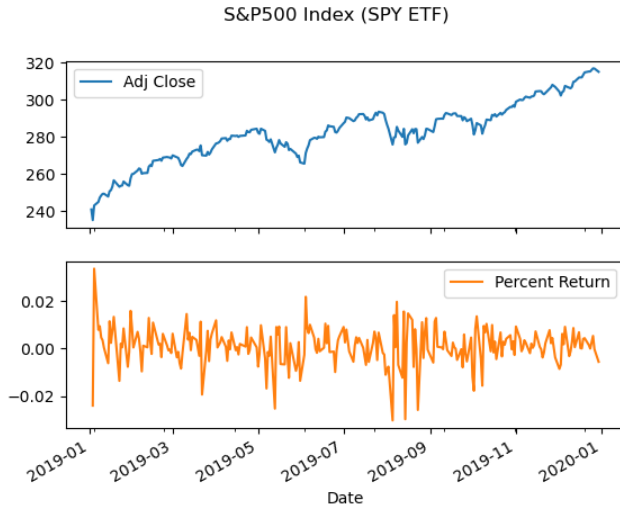
- 1 Consistency to the proxy distribution under strong assumptions on  $\Theta$ ,
- 2 asymptotic normality also under strong assumptions on  $\Theta$ ,
- 3 no uniform-in-time bounds for either the bias or for the asymptotic variances (caveat emptor)

$$\begin{aligned} & \overbrace{p(y_{t+1} \mid y_{1:t})}^{\text{ideal fcast density}} - \overbrace{\int p(y_{t+1} \mid \theta, y_{1:t}) p(\theta \mid y_{1:s}) d\theta}^{\text{our forecast}} \\ &= \int p(y_{t+1} \mid \theta, y_{1:t}) \left\{ \frac{p(\theta \mid y_{1:t})}{p(\theta \mid y_{1:s})} - 1 \right\} p(\theta \mid y_{1:s}) d\theta \quad (1) \end{aligned}$$

- Using a prior is not encouraged (always make sure to examine forecasts produced!).
- For fixed  $s$ ,  $\frac{p(\theta|y_{1:t})}{p(\theta|y_{1:s})}$  (probably) grows in  $t$ .
- Occasionally updating posterior is better than never updating.
- Easy to extend to averaging over models, too!

# Numerical Experiments

# The S&P 500 Index



What about options contracts on this?

# Options on the S&P 500 Index

- 1 a **call (put) option** is a derivative contract that allows one to buy (sell) a fixed quantity at some **strike price**.
- 2 you can either buy or sell this contract
- 3 It has a finite lifetime—it expires on the **expiration date**.
- 4 you can buy and sell these contracts, or you can **exercise** the privilege of buying (selling) the underlying shares.
- 5 **European style** options will only allow you to exercise on the expiration date (SPX).
- 6 **American style** options allow an exercise at any time before expiration (SPY).

# Options: A Few Takeaways From Mathematical Finance

The option's price should be tightly linked with the price of the SPY, but we are not doing **pricing** here. We're not trying to come up with a hedge and identify arbitrages (if we were we would need SDEs and stochastic calculus)—we want to take bets without trading the underlying too actively.

Option prices tend to

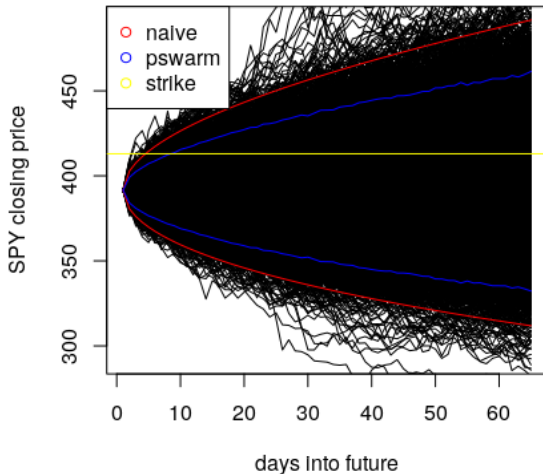
- 1 decrease in value **over time**
- 2 change in value if the underlying moves towards or away from the strike (nonlinearly with respect to the underlying)
- 3 decrease in value if volatility decreases



# Running the Particle Swarm

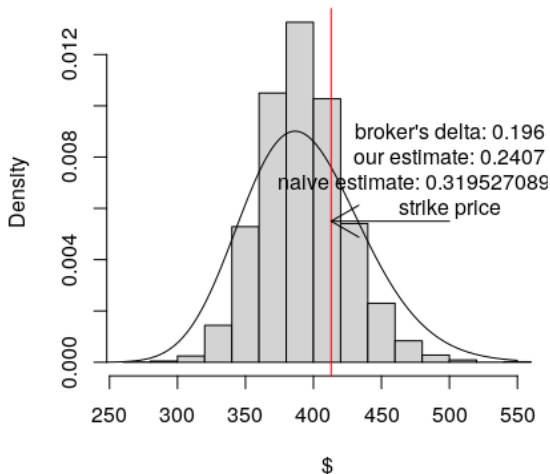
- 1 Use old posterior samples from a pseudo-marginal Metropolis-Hastings algorithm
- 2 Sample 100 parameter vectors "from" those
- 3 Each particle filter maintains 100 state samples
- 4 Each particle filter runs through the past 807 trading days (2018-01-01 until now)
- 5 Each particle filter is used to simulate forward in time 64 days (until expiration day)
- 6 Code: [github.com/tbrown122387/pf](https://github.com/tbrown122387/pf) (Brown, 2020), [github.com/tbrown122387/ssme](https://github.com/tbrown122387/ssme)

# Running the Particle Swarm

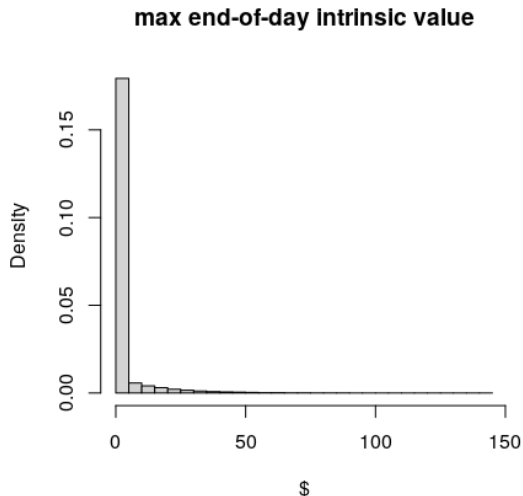


# Running the Particle Swarm

## SPY on expiration (May 21 '21)



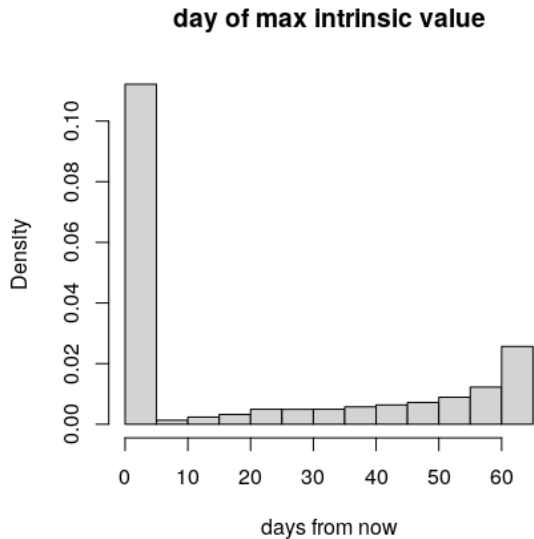
# Running the Particle Swarm



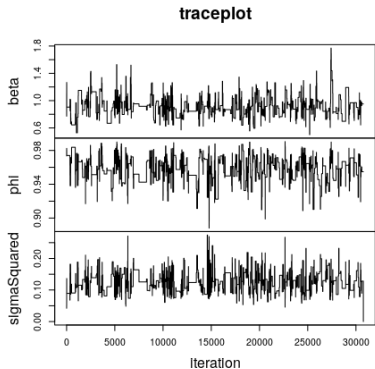
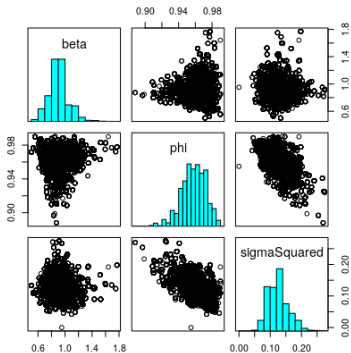
for American-style

\$2.61-\$2.68 at March 18 close.

# Running the Particle Swarm






# Running the Particle Swarm



Thanks!



# References I

-  Brown, Taylor R. (2020). “A Short Introduction to PF: A C++ Library for Particle Filtering”. In: *Journal of Open Source Software* 5.54, p. 2599. DOI: [10.21105/joss.02599](https://doi.org/10.21105/joss.02599). URL: <https://doi.org/10.21105/joss.02599>.
-  — (2021). *Approximating Posterior Predictive Distributions by Averaging Output From Many Particle Filters*. arXiv: [2006.15396](https://arxiv.org/abs/2006.15396) [stat.ME].
-  Carvalho, Carlos M. et al. (Feb. 2010). “Particle Learning and Smoothing”. In: *Statist. Sci.* 25.1, pp. 88–106. DOI: [10.1214/10-STS325](https://doi.org/10.1214/10-STS325). URL: <https://doi.org/10.1214/10-STS325>.



# References II



Chopin, N., P. E. Jacob, and O. Papaspiliopoulos (June 2013). “SMC2: an efficient algorithm for sequential analysis of state space models”. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 75.3, pp. 397–426. DOI: [10.1111/j.1467-9868.2012.01046.x](https://doi.org/10.1111/j.1467-9868.2012.01046.x). URL: <http://dx.doi.org/10.1111/j.1467-9868.2012.01046.x>.



Doucet, Arnaud and Adam M. Johansen (2011). *A tutorial on particle filtering and smoothing: fifteen years later*.

# References III



Fearnhead, Paul (2002). “Markov Chain Monte Carlo, Sufficient Statistics, and Particle Filters.”. In: *Journal of Computational and Graphical Statistics* 11.4, pp. 848 –862. ISSN: 10618600. URL: <http://proxy01.its.virginia.edu/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=edsjsr&AN=edsjsr.1391165&site=eds-live>.

# References IV







Gilks, Walter R. and Carlo Berzuini (2001). “Following a moving target—Monte Carlo inference for dynamic Bayesian models”. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 63.1, pp. 127–146. DOI: <https://doi.org/10.1111/1467-9868.00280>. eprint: <https://rss.onlinelibrary.wiley.com/doi/pdf/10.1111/1467-9868.00280>. URL: <https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/1467-9868.00280>.



Kalman, Rudolph Emil (1960). “A New Approach to Linear Filtering and Prediction Problems”. In: *Transactions of the ASME—Journal of Basic Engineering* 82.Series D, pp. 35–45.

# References V

-  Kitagawa, Genshiro (Sept. 1998). “A Self-Organizing State Space Model”. In: 93.443, pp. 1203–1215. ISSN: 0162-1459 (print), 1537-274X (electronic). URL: [http://www.amstat.org/publications/jasa/abstracts\\_98/sept/kitagawa.html](http://www.amstat.org/publications/jasa/abstracts_98/sept/kitagawa.html).
-  Liu, Jane and Mike West (2001). “Combined Parameter and State Estimation in Simulation-Based Filtering”. In: *springer*. Chap. 10.
-  Storvik, G. (2002). “Particle filters for state-space models with the presence of unknown static parameters”. In: *IEEE Trans. Signal Process.* 50, pp. 281–289.
-  Taylor, Stephen (Jan. 1982). “Financial Returns Modelled by the Product of Two Stochastic Processes, a Study of Daily Sugar Prices 1961-79”. In: 1.