Forecasting High-Frequency Futures Returns Using Online Langevin Dynamics

Hugh L. Christensen, James Murphy, and Simon J. Godsill, *Member, IEEE*

***Abstract*—Forecasting the returns of assets at high frequency is the key challenge for high-frequency algorithmic trading strategies. In this paper, we propose a jump-diffusion model for asset price movements that models price and its trend and allows a momentum strategy to be developed. Conditional on jump times, we derive closed-form transition densities for this model. We show how this allows us to extract a trend from high-frequency finance data by using a Rao–Blackwellized variable rate particle filter to filter incoming price data. Our results show that even in the presence of transaction costs our algorithm can achieve a Sharpe ratio above 1 when applied across a portfolio of 75 futures contracts at high frequency.**

***Index Terms*—Futures trading, online learning, particle filter, quantitative finance, tracking.**

I. Introduction

ALGORITHMIC trading of financial securities is big business, with an estimated 45% of trading volume in futures contracts from computer generated trading decisions [1].

In this paper, we propose a new trading algorithm from the popular *trend-following* or *momentum* class of trading strategies. In these an agent attempts to identify an ongoing price trend and then to take a market position in order to benefit from its continuation. Our algorithm is based on the ideas of physical object tracking and uses similar Bayesian filtering techniques to extract a trend from price observations.

Momentum strategies have been the source of much academic debate (e.g., [2]–[6]) because they appear to defy even the weak form of the *Efficient Market Hypothesis* (EMH) of [7], which states that prices should not be predictable from analyzing their past history. This form of the EMH is consistent with random-walk behavior of asset prices and suggests that no form of *technical analysis* (price prediction based solely on studying previous price history) can generate above average returns without taking on above average risk [2]. However, technical analysis remains in widespread use in public markets [8] and various forms have been shown to have at least some predictive power [8]–[11]. Momentum effects in particular have been extensively studied (e.g., [4], [12]–[16], among others) and have been found to exist in a number of markets including foreign exchange [15], commodities [17] and equities [14]. Proposed explanations of these effects include the non-instantaneous reaction of the market to news events [12], meaning that the effects of events take place over several trading periods, and herding behavior [18], in which, for example, investors clamor for assets that have recently performed well, further increasing their price. Though this herding behavior has been termed "irrational exuberance" [18], irrationality of investors is not required to explain momentum effects [5], [19]. Some advocates of market efficiency contest that trading costs would wipe out any practical benefit of momentum trading [2], [6]. We therefore aim to show that our algorithm delivers positive performance even in the face of trading costs. Despite efficient market objections to momentum trading, the continuing existence and profitability of momentum-based funds (for example, AHL, the world's largest quantitative hedge fund [20]) has led to obvious ongoing interest.

Momentum effects have been found at a range of frequencies from multi-monthly [4], to intraday [21], though this latter study found them to be more profitable at higher (intraday) frequencies. The algorithm we propose can be applied to observations at any frequency (though it is likely that not all frequencies will be profitable), and, since it does not rely on regularly spaced observations could be applied to asynchronous *tick data* (high-frequency data listing all market transactions).

Any momentum strategy relies on its ability to determine a trend to follow and numerous ways of doing so have been proposed. Some profitable strategies have been as simple as buying shares that performed well over a previous period [4]. A more sophisticated methodology (developed by practitioners in the late 1970s) consists of filtering the price though two low-pass filters (for example moving averages with different window lengths), the difference enabling trend calculation and the filter removing high-frequency noise [22]. Later work has looked at wavelet approaches to approximating bandpass filters [23] and replication of the payoff structures using look-back options [16]. While these methods have many common features they differ in their step response, sensitivity to high-frequency noise and computational cost.

In our algorithm, we propose obtaining trend information by using tracking algorithms akin to those used for physical object tracking. Such tracking algorithms are usually based around a motion model for the target being tracked. This model must be able to make predictions about the target motion and its accuracy has a substantial effect on the tracking performance [24]. The underlying dynamics of financial asset values are much

Manuscript received July 20, 2011; revised November 18, 2011; accepted March 03, 2012. Date of publication March 20, 2012; date of current version July 13, 2012. H. L. Christensen and J. Murphy are equal contributors to this paper. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Ali Akansu.

The authors are with the Signal Processing and Communications Laboratory, Engineering Department, Cambridge University, CB2 1PZ, U.K. (e-mail: hlc54@cam.ac.uk; jm362@cam.ac.uk; sjg30@cam.ac.uk).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JSTSP.2012.2191532

less clear than in the physical case, despite extensive study of the behavior of asset prices, e.g., [25]–[31]. Though the "stylized facts" established in this literature can help point out shortcomings of an asset price model, they do not prescribe a specific model of asset dynamics. Numerous models have been proposed, largely dependent on the application and their analytical tractability; popular models include GARCH models in econometrics [32], exponential Brownian motion models in Black–Scholes option pricing [33], stochastic volatility models to fit volatility clustering and option price smiles [34] and jump-diffusion and Lévy process models to fit heavy tails, e.g., [35] and [36], respectively. Such random walk models, however, do not allow for a predictable trend in asset prices. Since it is the aim of this work to determine such a trend we must move away from these models above and introduce a "trend" term. The model we describe in Section II can be seen as similar in spirit to the "near constant velocity" physical tracking models described in [24]. It can also be viewed as an extension of the Langevin dynamics used in [37].

Our model also addresses the issue of trend changes, which can cause difficulty for momentum strategies [38], by allowing jumps in our trend process. Jumps in the trend process can model sharp changes of sentiment in the market and allow our predictions to reflect these more quickly than models that simply smooth the price series. Aside from the jump timings we assume that the asset price and trend processes are Gaussian. By conditioning on the jump times we can obtain an analytically tractable model allowing for fast and accurate prediction using the Kalman filter. The non-Gaussian jump times can be inferred using a particle filter. This idea is similar to those in [37], [39], and [40], which aim to track physical objects such as aircraft capable of executing rapid, sharp manoeuvres by introducing jumps in their acceleration process.

The remainder of the paper is structured as follows. In Section II, we present the observation and dynamics models we propose to use for asset prices within our tracking algorithm. We show how these can be used to derive the observation and state transition densities required for the Bayesian state-space filtering techniques we wish to apply.

Section III gives details of the variable-rate particle filter algorithm that we use to infer trend information from our observations of asset prices. We show how to use the output of the algorithm to make predictions about the next observation. We also detail how we turn these model predictions into trading decisions.

Section IV presents the results of our algorithm applied to both daily and intraday futures prices. We show that the algorithm, in the presence of a suitable framework and simulation methodology, manages significantly positive performance, even after accounting for trading costs.

Section V concludes and suggests possible avenues of future research coming from this study. We find the algorithm to be a viable methodology for trading futures algorithmically at high frequency.

II. Model Components

The trend-following algorithm we propose uses sequential Bayesian Monte Carlo inference to obtain information about asset price behavior. Such inference techniques (described in detail in Section III) require that we can model the system generating the observations as a state-space system, where the system has some state at each point in time, with observations generated according to the underlying system state at the observation time. Such state-space systems require two models to define them: a state-transition or dynamics model, which describes the evolution of the system state over time, and an observation model, which describes how observations are generated from the current state. Both of these models must be probabilistic, giving a density over successor states or observations, respectively, given the current state. These densities are the key features of our model, so we derive them in detail below.

A. State Transition Model

The model we propose has a "value" x_1 and "trend" x_2 component. The trend is a mean-reverting random process subject to two different types of random innovation: Gaussian noise of constant volatility and random jumps. The value component x_1 is given by integrating the trend process with respect to time. Observed prices (y) are modeled as observations of the value process x_1 subject to Gaussian noise.

For clarity of exposition, we first consider the process without jump innovations. In this case, the system we propose can be written in matrix-vector form as

$$\begin{bmatrix} dx_{1,t} \\ dx_{2,t} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & \theta \end{bmatrix} \begin{bmatrix} x_{1,t} \\ x_{2,t} \end{bmatrix} dt + \begin{bmatrix} 0 \\ \sigma \end{bmatrix} dW_t \tag{1}$$

where the dW_t is a Gaussian noise processes and $x_{i,t}$ is the value of the x_i process at time t. We require the mean reversion coefficient θ to be non-positive and the noise standard deviation σ to be positive. This can be written as

$$dX_t = AX_t dt + b dW_t \tag{2}$$

where we have defined

$$X_t = \begin{bmatrix} x_{1,t} \\ x_{2,t} \end{bmatrix}, \quad A = \begin{bmatrix} 0 & 1 \\ 0 & \theta \end{bmatrix}, \quad b = \begin{bmatrix} 0 \\ \sigma \end{bmatrix}. \tag{3}$$

Our model has the advantage of being analytically tractable, while still retaining the salient features necessary to execute a trend following strategy. For such a purpose we require some notion of trend, excluding random walk models like Brownian motion and geometric Brownian motion. Including mean reversion in our model reflects our view that trends will fade over time. The inclusion of jumps in the trend process allows trends to reverse or disappear rapidly. Fig. 1 illustrates the types of changes in trend and price that can be accommodated by such a model.

In order to apply our tracking algorithm, we need to go from this statement of the model to an expression for the distribution of the system state at a future time T, given its state (or state distribution) at the current time S $(S < T)$.

The system in (2) is linear time-invariant (LTI) Gaussian, meaning it can be solved in closed form using Itô calculus [41] (for our model this is a well known procedure but for completeness we outline the solution here as this allows us to deal with

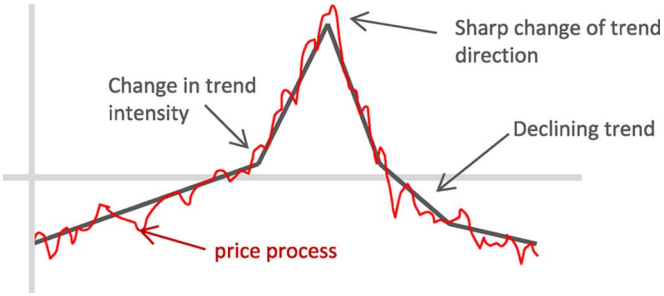

Fig. 1. Sketch of price dynamics that can be captured with our model, showing (from left to right) the effect of a change in the intensity of a trend, a sharp trend change, and a declining trend.

jumps later on). The solution (integrating from time S to T) is given by the stochastic integral

$$X_T = e^{A(T-S)}\left[X_S + \int_S^T e^{-At} b dW_t\right]. \tag{4}$$

If X_S is Gaussian distributed then X_T is itself Gaussian distributed because the stochastic integral is also a Gaussian random variable. Gaussian distributions are fully determined by their first two moments, so X_T is fully specified by its expectation and covariance.

Since the expectation of the stochastic integral is zero the expectation of X_T is

$$\mathbb{E}(X_T) = e^{A(T-S)}\mathbb{E}(X_S). \tag{5}$$

In order to find the variance-covariance of X_T we note the independence of X_S and $\int_S^T e^{-At} dW_t$, and the fact that the latter integral has an expectation of 0 in all components, so that some of the terms in the following expansion are equal to zero:

$$\operatorname{cov}(X_T) = \mathbb{E}((X_T - \mathbb{E}(X_T))(X_T - \mathbb{E}(X_T))'). \tag{6}$$

Substituting in the expressions for X_T and $\mathbb{E}(X_T)$ in (4) and (5), respectively, and, for notational convenience, defining

$$Q(r,s) = \mathbb{E}\left(\left(\int_r^s e^{-At} b dW_t\right)\left(\int_r^s e^{-At} b dW_t\right)'\right)$$

we get

$$\operatorname{cov}(X_T) = e^{A(T-S)}[Q(S,T) + \operatorname{cov}(X_S)]\left(e^{A(T-S)}\right)'. \tag{7}$$

Using a multivariate instance of the Itô isometry, applied to a deterministic process we are able to find the required expectation to calculate $Q(r,s)$:

$$Q(r,s) = \int_r^s e^{-At} bb'(e^{-At})' dt \tag{8}$$

which gives us a deterministic expression for variance-covariance of X_T. This calculation of the integral expression for $Q(r,s)$ in (8) is not completely trivial but can be obtained using matrix fraction decomposition [42] or by series expansion of the exponential functions [37]. The series expansion is only plausible for low dimensional systems and so while it could be applied here to our 2-D system, we use the former method for its generality.

Without jumps, then, the transition density we are seeking is given by

$$p(X_T \mid X_S) \sim \mathcal{N}(\mathbb{E}(X_T \mid X_S), \operatorname{cov}(X_T \mid X_S)) \tag{9}$$

with the conditional expectation and covariances in (9) given by the expressions in (5) and (7), respectively, given a known value for X_S (so that $\mathbb{E}(X_S) = X_S$ and $\operatorname{cov}(X_S) = 0$).

The system in (1) can be written as a standard Langevin equation. This equation has been used to model the motion of a particle in a fluid subject to stochastic forces [43]. In this case, θ has a physical interpretation as the ratio of the coefficient of resistance to particle mass. This allows for an analogous interpretation of the θ parameter as a trend resistance term, causing trends to revert to 0.

We are now able to build on this model by including jump terms, giving a governing SDE of the form

$$\begin{bmatrix} dx_{1,t} \\ dx_{2,t} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & \theta \end{bmatrix} \begin{bmatrix} x_{1,t} \\ x_{2,t} \end{bmatrix} dt + \begin{bmatrix} 0 \\ \sigma \end{bmatrix} dW_t + \begin{bmatrix} 0 \\ 1 \end{bmatrix} dJ_t \tag{10}$$

where dJ_t is the jump process.

No closed form solution of the jump system exists in general. However, by conditioning on jump times we can think of the system as if the jump times are known *a priori* (though our algorithm does include the jump times as random variables). Then we are able to separate the system into a tractable LTI Gaussian part (between the jumps), solved as above, and a nonlinear, non-Gaussian part (the jumps).

The jumps follow a Gauss–Poisson process, with jump times τ_k following a Poisson arrival process and jump sizes S_k distributed as a multivariate Gaussian, so that

$$S_k \sim \mathcal{N}\left(\mu_J, \sigma_J^2\right) \tag{11}$$

$$\tau_k - \tau_{k-1} \sim \text{exponential}(\lambda_J) \tag{12}$$

where μ_J is the mean jump size and σ_J^2 is the jump size variance. This allows us to define our jump process J_t as

$$J_t = \sum_{k \in \{k \mid \tau_k < t\}} S_k \tag{13}$$

so that $dJ_t = S_k$ at τ_k^+ (i.e., the moment just after τ_k). Note however that there is no necessity in our framework to limit ourselves to an exponential inter-arrival distribution; for example, gamma distributions were used in some earlier tracking work [37].

In the model thus described there are six parameters: the mean reversion coefficient θ, the Gaussian noise variance σ, the jump rate λ_J, he jump mean μ_J, the jump size variance σ_J^2, and the observation noise variance σ_{obs}^2.

In the presence of jumps the calculation of the state transition density is more complicated but can be accomplished if we condition on the jump times. Consider again the state transition from time S to T. If a single jump occurs between S and T, e.g.,

at time $\tau \in (S,T]$, then we can think of the transition in three parts: the pre-jump diffusion, the jump itself and the post-jump diffusion. By conditioning on the jump time we can think of τ as a known value rather than a random variable. For cases with no jumps or multiple jumps between S and T this processes can be modified appropriately by considering a single diffusion section from S to T or multiple diffusion and jump sections (e.g., for two jumps, diffusions from S to τ_1, τ_1 to τ_2, and τ_2 to T), respectively [37]. Since we are trying to develop a transition density conditional on the state at time S, we will also condition on X_S.

Following the diffusion from S to τ (where τ is the instant before the jump occurs) we calculate the distribution of the state exactly as in the non-jumping case above, giving a Gaussian state distribution with first and second moments given by

$$\begin{aligned}\mathbb{E}(X_\tau \,|\, \tau, X_S) &= e^{A(\tau-S)}X_S \\ \operatorname{cov}(X_\tau \,|\, \tau, X_S) &= e^{A(\tau-S)}Q(S,\tau)\left(e^{A(\tau-S)}\right)'\end{aligned}$$

using the formulae in (5) and (7), respectively.

Across the period of the jump, from time τ to τ^+ (where τ^+ is the instant immediately after the jump occurs) we can calculate the distribution by noting that we assume normally distributed jump sizes, independent of any other innovations as in (12). This leads to a post-jump distribution that is also Gaussian with mean and covariance given by

$$\begin{aligned}\mathbb{E}(X_{\tau^+} \,|\, \tau, X_S) &= \mathbb{E}(X_\tau \,|\, \tau, X_S) + \mu_J \\ \operatorname{cov}(X_{\tau^+} \,|\, \tau, X_S) &= \operatorname{cov}(X_\tau \,|\, \tau, X_S) + \sigma_J^2.\end{aligned}$$

Using this as a starting point, we can then use the expressions in (5) and (7) to calculate the first two moments of the state distribution following the diffusion from τ^+ to T. Since the post-jump distribution is Gaussian, the state distribution at T (conditional on jump times) will also be Gaussian and so will be fully specified by

$$\begin{aligned}\mathbb{E}(X_T) &= e^{A(T-\tau)}\mathbb{E}(X_{\tau^+} \,|\, \tau, X_S), \\ \operatorname{cov}(X_T) &= e^{A(T-\tau)}[Q(\tau,T)+\operatorname{cov}(X_{\tau^+} \,|\, \tau, X_S)]\left(e^{A(T-\tau)}\right)'.\end{aligned}$$

By substitution, we can now write expressions for these moments in terms of X_S (or its distribution), i.e., with no reference to the state at intermediate times τ and τ^+. These expressions are

$$\mathbb{E}(X_T \,|\, \tau, X_S) = e^{A(T-S)}X_S + e^{A(T-\tau)}\mu_J \tag{14}$$

and

$$\begin{aligned}\operatorname{cov}(X_T \,|\, \tau, X_S) = e^{A(T-\tau)}\left[Q(\tau,T) + \sigma_J^2\right]\left(e^{A(T-\tau)}\right)' \\ + e^{A(T-S)}Q(S,\tau)\left(e^{A(T-S)}\right)'.\end{aligned} \tag{15}$$

These allow us to define the state transition density function (conditional on knowing the jump times) as

$$\begin{aligned}&p(X_T \,|\, \{\tau\}_{S:T}, X_S) \\ &\quad \sim \mathcal{N}(\mathbb{E}(X_T \,|\, \{\tau\}_{S:T}, X_S), \operatorname{cov}(X_T \,|\, \tau_{S:T}, X_S))\end{aligned} \tag{16}$$

where $\tau_{S:T} = \{\tau_k \mid S < \tau_k \leq T\}$, the set of jump times between times S and T. Given the jump times, this gives us the state transition density we need.

The jump times on which we must condition are, however, still unknown. To fully specify our state transition model, we also need to specify the jump time distribution. In this work, in the absence of any specific conditioning information on jump times, we choose the memoryless exponential distribution. The jump times are thus distributed as

$$p(\tau_i \,|\, \tau_{1:i-1}) \sim \text{exponential}(\lambda_J). \tag{17}$$

Since this distribution is not Gaussian, these must be inferred using a method able to cope with non-Gaussian distributions, such as the particle filter, which we develop below for this model.

An alternative approach would be to calculate the state transition function from time S to T, including jumps at unknown times, directly. This transition function would be non-Gaussian, as it would include the exponentially distributed (in time) jumps. We would therefore need to apply a method able to cope with non-Gaussian transition functions to the entire state process. The approach outlined above, in contrast, allows much of the state to be estimated using the computationally efficient and optimal Kalman filter while the unknown jump times are sampled in a particle filter.

B. Observation Model

For our observation model, we assume that the ith price observation, y_i, observed at time t_i, is the result of an observation of the value process x_{1,t_i}, perturbed by Gaussian noise of a fixed variance

$$y_i = x_{1,t_i} + v_{t_i}, \quad v_{t_i} \sim \mathcal{N}\left(0, \sigma_{\text{obs}}^2\right). \tag{18}$$

We assume that the observation y_i is conditionally independent of all other observations and elements of the state process given x_{1,t_i}.

This gives an observation density, conditional on the state at time t_i, of

$$y_i \sim \mathcal{N}\left(x_{1,t}, \sigma_{\text{obs}}^2\right). \tag{19}$$

III. Inference Algorithm

Given a state-space model and an observation function, the filtering problem is that of estimating the state of the underlying model (perhaps as a probability distribution over possible states known as a *posterior filtering distribution*) at time t given a sequence of observations up to time t. Since the number of observations increases with time we would like to calculate our new state estimate *sequentially* using only the previous estimate and the new observation, since then we are not required to store all previous observations and this is likely to be computationally efficient.

The objective of our inference algorithm is to take a series of price observations $y_{1:N} = \{y_i \mid i \in 1, \ldots, N\}$ (with the observation y_i made at time t_i) and use these to infer the underlying state X_t at time t.

Conditional on the jump times, our model has a Gaussian distribution [(14) and (15)] that depends in a linear way on the state at some previous time. This is a sufficient condition for us to be able to apply the Kalman filter to obtain the state distributions *given we already know the jump times*. In reality, we do *not* already know the jump times so we must apply another filter to infer these. Since their governing dynamics are not linear Gaussian, we cannot use the Kalman filter and so opt instead to use a particle filter, able to cope with non-Gaussian state distributions. This will give us a weighted sample-based approximation of the jump-time distribution, with each sample (or *particle*) being a collection of jump times. For each particle we can then use this set of jump times as conditioning information in order to infer a set of (Gaussian) probability distributions for our state. Combining these state distributions according to the sample weights then gives an approximate joint posterior filtering distribution for the state and jump times. This is an example of *Rao–Blackwellization* [44], [45] in which the state is separated into a linear Gaussian portion, which can be inferred using the Kalman filter, and a nonlinear and/or non-Gaussian portion, inferred using an alternative method. This has great advantages for computation speed and accuracy of the particle filter [45].

In this section, we briefly review the Kalman and particle filters before showing how we use these in combination to do state inference with the proposed asset price model.

A. Kalman Filter

In the case of linear state models with additive Gaussian noise terms, the Kalman filter solves the filtering problem (in the sense that it can be shown to be the optimal linear estimator of the state in a mean squared error sense [46]). It can also be derived as a Bayesian filter [47], with the previous filtering distribution up to an observation time t_i forming the prior for calculating the filtering distribution at the next observation time $t_{i+1} > t_i$. This relies on the fact that the Gaussian distribution is its own conjugate prior, meaning that the posterior filtering distributions are always Gaussian for such models.

The model with states $X_{t_{1:N}} = \{X_{t_i} \mid i \in 1, \ldots, N\}$ (with X_{t_i} being the state at the time t_i corresponding to observation y_i) and observations $y_{1:N}$ can be written as

$$X_{t_i} = F_i X_{t_{i-1}} + u_i, \quad u_i \sim \mathcal{N}(0, C_{u_i}) \tag{20}$$

$$y_i = G_i X_{t_i} + v_i, \quad v_i \sim \mathcal{N}(0, C_{v_i}) \tag{21}$$

where F_i and G_i are the state transition and observation matrices, respectively, at the ith observation time, and C_{u_i} and C_{v_i} are the corresponding state transition and observation covariances. For our model these will be specified by the state transition and observation densities in (16) and (19), respectively, i.e.,

$$F_i = e^{A(t_i - t_{i-1})} \tag{22}$$

$$G_i = [1\ 0] \tag{23}$$

$$C_{u_i} = \operatorname{cov}\left(X_{t_i} \mid \tau_{t_{i-1}:t_i}, X_{t_{i-1}}\right) \tag{24}$$

$$C_{v_i} = \sigma^2_{\text{obs}} \tag{25}$$

where $\tau_{s:t} = \{\tau_k \mid \tau_k \in (s, t]\}$, i.e., the set of jumps between times s and t. These give distributions for X_{t_i} and y_i that match with the state transition and observation densities above (assuming the jump-size mean μ_J is 0, though this assumption can easily be relaxed).

Another important feature of the Kalman filter that we rely on in this work is its ability to sequentially calculate the observation likelihood $p(y_{1:n})$ via the *prediction error decomposition* (PED) [48]. The PED gives us the observation probability for the most recent observation, given all others up to that point:

$$p\left(y_i \mid y_{1:i-1}, \tau_{t_0:t_i}\right) \sim \mathcal{N}\left(y_i \mid \mu_{y_i}, C_{y_i}\right) \tag{26}$$

where

$$\mu_{y_i} = G_i \mu_{i \mid 0:i-1} \tag{27}$$

$$C_{y_i} = G_i C_{i \mid 0:i-1} G_i' + C_{v_i}. \tag{28}$$

Here $\mu_{i \mid 0:i-1}$ and $C_{i \mid 0:i-1}$ are the predicted mean and covariance of the state distribution at the time of the ith observation given the first $i-1$ observations. They are obtained from the previous step of the Kalman filter

$$\mu_{i \mid 0:i-1} = F_i \mu_{i-1 \mid 0:i-1} \tag{29}$$

$$C_{i \mid 0:i-1} = F_i C_{i-1 \mid 0:i-1} F_i' + C_{u_i} \tag{30}$$

where $\mu_{i-1 \mid 0:i-1}$ and $C_{i-1 \mid 0:i-1}$ are the mean and covariance of the filtering distribution at the time of the $(i-1)$th observation y_{i-1}, derived from the previous "prediction" through the "correction" step of the filter (see, e.g., [49]), given by

$$K_i = C_{i \mid 0:i-1} G_i' \left(G_i C_{i \mid 0:i-1} G_i' + C_{v_i}\right)^{-1} \tag{31}$$

$$\mu_{i \mid 0:i} = \mu_{i \mid 0:i-1} + K_i (y_i - G_i \mu_{i \mid 0:i-1}) \tag{32}$$

$$C_{i \mid 0:i} = (I - K_i G_i) C_{i \mid 0:i-1}. \tag{33}$$

In order to start the filter we need a prior state distribution specifying our belief about the underlying system state at t_0. We take this to be Gaussian (and this is necessary for the Kalman filter) and can therefore specify it by its mean μ_0 and covariance C_0, which we use in the first "predict" step of the filter ((29) and (30)) as $\mu_{0 \mid 0:0}$ and $C_{0 \mid 0:0}$, respectively.

B. Variable Rate Particle Filter (VRPF)

Were the jump times known *a priori*, the Kalman filter above would be sufficient to solve our filtering problem. However, without known jump times our model does not have a Gaussian state transition density. For such systems there is no longer any guarantee that the Kalman filter will give the conditional mean of the state distribution. We therefore require a filter that can cope with such non-Gaussian models. Variants of the Kalman filter algorithm such as the extended Kalman filter (EKF) and unscented Kalman filter (UKF) offer approximate methods, but we choose to use the theoretically appealing *particle filter*. This can be shown to converge asymptotically to the correct filtering distribution as the number of particles increases to infinity.

The standard particle filter algorithm uses importance sampling to represent the posterior filtering distribution after each observation using a collection of weighted samples (or *particles*). The samples are drawn from an easy-to-sample distribution over the space of interest, with weights chosen so as to

approximate the posterior filtering distribution. In the standard case the posterior filtering density is given by

$$p\left(X_{t_j} \mid y_{1:j}\right) \approx \sum_{p \in P_{t_j}} W_p(t_j) X^p_{t_j} \delta_{X^p_{t_j}} \tag{34}$$

where $X^p_{t_j}$ is the state of particle p at time t_j, $W_p(t_j)$ is the weight of particle p at time t_j, P_{t_j} is the collection of all particles at time t_j and δ_x is a delta function with mass 1 at the point x. This gives a sum-of-deltas approximation to the posterior filtering distribution, with nonzero values at the location of the particles $X^p_{t_j}$. A recent survey of particle filtering is given in [45].

The idea of Rao–Blackwellization allows us to separate our filter into a particle filter for the inference of jump times and a Kalman filter for the rest of the (conditionally linear Gaussian) state inference, which uses the jump times from the particle filter as conditioning information. Since our non-Gaussian state is simply the collection of jump times, each particle in our filter can be represented by a set of jump times. The distribution of the conditionally linear Gaussian state (the values of $x_{1,t}$ and $x_{2,t}$ in our system) corresponding to each particle can be inferred by using the Kalman filter described above with each particle's collection of jump times as conditioning information. The joint posterior filtering distribution (jump times and system state) is given by the sum of jump time and state distributions (these latter given by the Kalman filter), weighted by the particle weights. This gives an approximate underlying state distribution given by

$$p\left(X_{t_j} \mid y_{1:j}\right) \approx \sum_{p \in P_{t_j}} W_p(t_j) p\left(X^p_{t_j} \mid y_{1:t_j}, \tau^p_{t_0:t_j}\right) \tag{35}$$

where $p(X^p_{t_j} \mid y_{1:t_j}, \tau^p_{t_0:t_j})$ is the posterior filtering density obtained from the Kalman filter conditioned on particle p's set of jump times between t_0 and t_j, denoted $\tau^p_{t_0:t_j}$. This filtering density from the Kalman filter is Gaussian, with mean $\mu^p_{j\,|\,0:j}$ and covariance $C^p_{j\,|\,0:j}$ given by (32) and (33), respectively. This means that the approximate posterior state distribution is a weighed sum of Gaussians, given by

$$p\left(X_{t_j} \mid y_{1:j}\right) \approx \sum_{p \in P_{t_j}} W_p(t_j) \mathcal{N}\left(\mu^p_{j\,|\,0:j}, C^p_{j\,|\,0:j}\right). \tag{36}$$

Though the set of jump times are sufficient to define a particle (since the rest of the underlying state can be inferred by running the Kalman filter over all available observations conditioned on them), for efficiency we also store the current mean and covariance of the underlying state for each particle, $\mu^p_{j\,|\,0:j}$ and $C^p_{j\,|\,0:j}$, respectively. These can then be sequentially updated using the Kalman filtering (29)–(33). We can therefore think of a particle as a collection of four pieces of data: the collection of jump times $\tau^p_{t_0:t_j}$, the posterior mean $\mu^p_{j\,|\,0:j}$ and covariance $C^p_{j\,|\,0:j}$ of the underlying state given the jump times and all available observations up to time t_j, and the current particle weight $W_p(t_j)$. We can write this as

$$p = \left\{\tau^p_{t_0:t_j}, \mu^p_{j\,|\,0:j}, C^p_{j\,|\,0:j}, W_p(t_j)\right\}. \tag{37}$$

In what follows, we assume that we have a collection of such particles representing the posterior filtering density at t_{j-1}, the time of the $(j-1)$th observation (in the initial case where $j = 1$ this will be a collection of evenly weighted particles representing our prior probability density, the jump-time sets of which will be empty, and the state mean and covariance of which will be set to their initial prior values, μ_0 and C_0, respectively). Our aim, therefore, is to update this particle collection to a new one that represents the posterior filtering density after making the next observation at time t_j. Fig. 2 shows this process in outline for a single particle in the collection and Algorithm 1 briefly outlines the algorithm.

The first step in updating each particle is to decide the number of successor particles (children) the particle will have in the next generation (step 1 in Fig. 2 and Algorithm 1). This is part of a resampling step in which some high-weighted particles are propagated multiple times in the next generation (and low-weighted particles might not be propagated at all). It is necessary to combat the problem of *particle weight degeneracy*, which afflicts particle filters, whereby almost all particle weight accumulates on a single particle [45]. The number of offspring of a particle does not matter (though we want to avoid the number of particles exploding or going to zero), as long as we divide the weight of the parent particle between them. This requirement means that we cannot arbitrarily select zero offspring, but must use probabilistic resampling or require at least one offspring for low-weighted particles. Probabilistic resampling for particles with less than one offspring is achieved by propagating these particles with a probability proportional to their ideal number of offspring (<1). Alternatively, if we require particles to have at least one offspring, the number of offspring M^p_j can be given by

$$M^p_j = \max(1, \lfloor M W_p(t_{j-1}) \rfloor). \tag{38}$$

Initially, each child particle is a copy of its parent, with jumps in the same positions from time t_0 to t_{j-1} and we divide the parent's weight evenly, so that the weight of each offspring particle q is

$$W_q(t_{j-1}) = \frac{W_p(t_{j-1})}{M^p_j}. \tag{39}$$

Once we have chosen the number of children a particle will have we must sample a new proposed jump time τ^q_* for each child q (step 2 in Fig. 2 and Algorithm 1). We do this by sampling from the jump-time distribution from our model, given that we have arrived at time t_{j-1} having previous jump times given by the parent particle's set of jump times $\tau^p_{t_0:t_{j-1}}$, so that

$$\tau^q_* \sim p_{\text{jump-time}}\left(\tau_* \mid t_{j-1}, \tau^p_{t_0:t_{j-1}}\right). \tag{40}$$

Some of these newly sampled jump times will be before the current observation time t_j (e.g., those of the first and last particle in Fig. 2); we call these particles *jumping* particles, since they propose the occurrence of a jump between the previous and current observations. The other particles have proposed jump times beyond the current observation time; we call these *non-jumping*. In fact, for jumping particles we continue to sample jumps until

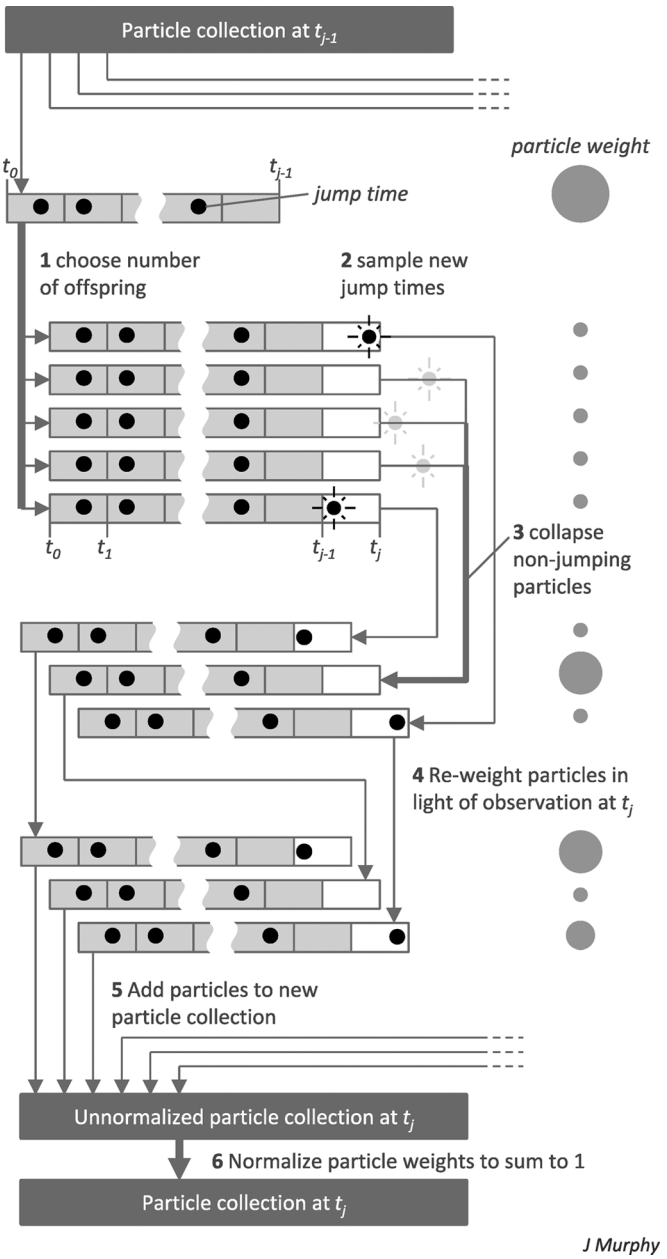

Fig. 2. Update step for a single particle of the variable rate particle filter. Gray rectangles represent particles, which consist of collections of jump times (black dots). Representative particle weights are illustrated along the right-hand side by gray circles, with diameter corresponding to weight. See text for further information on numbered steps.

Variable rate particle filter algorithm - outline steps

Input: Observations of the process
Output: Posterior filtering density estimate of jump times and system state

Initialization ($j = 0$): Create a collection P_{t_0} of N particles, each with no jumps, so that, in the notation of equation (37), each identical particle $p_i \in P_{t_0}$ is $p_i = \{\emptyset, \mu_0, C_0, 1/N\}$
while *observations available* **do**
  $j = j + 1$
  Observe y_j
  0 Initialize a new (empty) particle collection, $P_{t_j} = \emptyset$
  foreach *particle* $p \in P_{t_{j-1}}$ *(the previous particle collection)* **do**
    1a Choose the number of successor particles M_j^p for p (equation (38))
    1b Assign an equal share of particle p's weight to all children (equation (39))
    Initialize a set of particle p's children, $Q_p = \emptyset$
    Initialize count of non-jumping children $N_0 = 0$
    foreach *successor particle* q **do**
      2a Sample a new jump time τ_*^q for the particle from the model's jump time distribution (equation (40))
      if $\tau_*^q < t_j$ *(current observation time)* **then**
        2b add the jump to the particle's set of jumps
        2c add the particle to children i.e. $Q_p = Q_p \cup q$
      else
        2d note the particle as non-jumping; $N_0 = N_0 + 1$
      end
    end
    3a Collapse non-jumping successor particles into a single particle. i.e. create a particle q_0 with the same jumps as its parent p and weight equal to the share of the parent's weight due to non-jumping children.
    (i.e. $\tau_{t_0:t_j}^{q_0} = \tau_{t_0:t_{j-1}}^{p}$ and $W_{q_0}(t_{j-1}) = {}^{W_p(t_{j-1})N_0}/_{M_j^p}$)
    3b add this particle q_0 to the set of children, $Q_p = Q_p \cup q_0$
    foreach *child particle* $q \in Q_p$ *(set of children)* **do**
      4a Re-weight particle q in light of current observation
      – use Kalman filter conditioned on particle q's jump times to calculate observation likelihood via PED (see equation (26))
      – Set new weight to product of likelihood and current weight (equation (43))
      4b Update the current mean and covariance of particle q to those from the Kalman filter update equations (32) and (33)
    end
    5 Add all child particles to new particle collection
    $P_{t_j} = P_{t_j} \cup Q_p$
  end
  6 Normalize particle weights in new particle collection so they sum to 1
  Result: New particle collection represents posterior filtering density after seeing given observation; this becomes current particle collection
end

Algorithm 1: Variable rate particle filter (VRPF) algorithm outline

we propose one beyond the current observation time, allowing for the possibility of multiple jumps from time t_{j-1} to t_j.

The next step (step 3 in Fig. 2 and Algorithm 1) collapses all offspring particles with no jumps between t_{j-1} and t_j into a single offspring particle. This is possible since all these non-jumping particles are identical up to the time of the current observation t_j, with a set of jump times the same as that of their parent particle. This step is different from most resampling schemes (e.g., those in [50]) because in most applications there are not large numbers of effectively identical offspring. Here, however, this fact allows us to retain more particle diversity during resampling by reducing the number of children of highly weighted particles (meaning that, for example, requiring at least one offspring is plausible without particle explosion). The weight of the single particle into which all non-jumping particles is collapsed is the sum of the weight of all non-jumping particles.

Once we have obtained our reduced particle set (containing a number of jumping particles and at most one non-jumping particle), we can re-weight it in light of our observation at t_j (step 4 in Fig. 2 and Algorithm 1). The standard particle filter weight update equation is

$$W_q(t_j) \propto W_q(t_{j-1}) \frac{p\left(X_{t_j}^* \middle| X_{t_{j-1}}^*\right) p\left(y_j \middle| X_{t_j}^*\right)}{q\left(X_{t_j}^* \middle| X_{t_{j-1}}^*, y_j\right)} \tag{41}$$

where X_t^* is the complete system state at t for particle q, and $q(X_{t_j}^* \mid X_{t_{j-1}}^*, y_j)$ is the proposal function for the state at t_j given that at t_{j-1}. In our case, thanks to Rao–Blackwellization we only need to consider the non-Gaussian part of the state, the jump times. This gives us the weight update equation

$$W_q(t_j) \propto W_q(t_{j-1}) \times \frac{p\left(\tau^q_{t_{j-1}:t_j} \middle| \, t_{j-1}, \tau^q_{t_0:t_{j-1}}\right) p\left(y_j \mid y_{1:j-1}, \tau^q_{t_0:t_j}\right)}{q\left(\tau^q_{t_{j-1}:t_j} \middle| \, t_{j-1}, \tau^q_{t_0:t_{j-1}}\right)} \quad (42)$$

where $\tau^q_{s:t}$ is the set of jump times of particle q between times s and t, and $p(\tau^q_{t_{j-1}:t_j} \mid t_{j-1}, \tau^q_{t_0:t_{j-1}})$ is the jump time distribution, given we are at time t_{j-1} with the set of jumps $\tau^q_{t_0:t_{j-1}}$. The denominator here is the proposal density for the jump times given in (40). Since we choose the jump time proposal function to be equal to the conditional jump time distribution in the numerator we get a *bootstrap* particle filter and can simplify our weight update equation to

$$W_q(t_j) \propto W_q(t_{j-1}) p\left(y_j \mid y_{1:j-1}, \tau^q_{t_0:t_j}\right). \quad (43)$$

The term $p(y_j \mid y_{1:j-1}, \tau^q_{t_0:t_j})$ is the observation likelihood given the jump time collection and can be obtained from the PED of the Kalman filter as given in (26). This filter also infers the underlying state distributions for the particle (i.e., conditional on its particular set of jump times) at the observation times, which are used in our prediction step, below. So, in order to update the weights of our offspring particles in light of the latest observation y_j, we calculate the likelihood of the new observation using the Kalman filter conditioned on the particle's set of jump times and then multiply this by the previous particle weight (from the resampling step).

Once this has been done for all offspring particles, they can be added to the new particle collection for time t_j (step 5 in Fig. 2 and Algorithm 1).

After doing this for all particles in the original particle collection we will obtain a complete new particle collection for time t_j. Because the particle weight update in (43) is a proportionality relationship rather than an equality, the particle collection thus formed will have particle weights that do not sum to 1 as required. They can be made correct, however, by normalizing them so that they do (step 6 in Fig. 2 and Algorithm 1). This particle collection is now a particle representation of the posterior filtering distribution at time t_j. We can repeat this process for each new observation received.

In order to generate a trading signal, we use the expected value of our posterior estimate of the underlying processes X_{t_j}, which we denote $\hat{X}_{t_j}$. This can be calculated by taking the expectation of the posterior filtering density for the state in (36). Since this is a weighted sum of Gaussians, the expected value is just a weighted sum of their means, so

$$\hat{X}_{t_j} = \mathbb{E}\left(X_{t_j} \mid y_{1:j}\right) \approx \sum_{p \in P_{t_j}} W_p(t_j) \mu^p_{j \mid 0:j}. \quad (44)$$

The expected values of the price and trend processes are then just the first and second components, respectively, of the $\hat{X}_{t_j}$ vector, which we denote $\hat{x}_{1,t_j}$ and $\hat{x}_{2,t_j}$.

IV. Experiments and Results

A. Jump Time Estimation

To illustrate the detection of jumps in the trend process, the variable rate filter was run on synthetic data, generated from the filter's model with known jump times. Fig. 3 shows the result of running the filter on this data, with the trend process and the filtering density shown in the upper graph and the jump detection shown in the lower graph. This latter shows the total proportion of particle weight held by particles having a jump between adjacent observation times. Since the particle filter targets the posterior density of $X_{1:t_i}$ at t_i it acts as a (fairly crude) smoother at fixed lags and thus a smoothed state estimate (including jump times) at lag l is given by the marginal posterior distribution of $X_{t_{i-l}}$ at time t_i. This allows the smoothed total jumping weight to be calculated for various lags; Fig. 3 shows this for the filtering distribution (zero lag) and for a lag of ten observation periods. As expected the total jumping weight is much more sharply peaked for the lagged case, with many periods having nearly zero or nearly one total jumping weight indicating no or all significant particles jumping, respectively. In the filter case, the total jumping weight is frequently around 0.15 for periods with no jump (and significantly higher at jumps). This shows many jumps are being proposed and considered (some of which might have small amplitude). With the arrival of more observations the presence of a jump is either confirmed or refuted and so the lagged jumping weights are more sharply defined (though at long lags the degenerate histories of particles will also be partly responsible; there may only be one ancestor particle). In this example all strong jumps and almost all of the weaker ones are detected (in that they are deemed probable in the smoothed estimate). Some weaker jumps are not detected but these are also difficult to distinguish by eye from standard diffusion moves. Where jumps happened roughly midway between observations the jumping weight is often split between two neighboring observations.

B. Real Data

In order to evaluate our approach in a realistic scenario we use time series data for 75 of the world's most liquid futures contracts for the period 01 January 2006 to 01 January 2011, where a liquid security is one which can be traded without causing significant price movements. The data used is detailed in [51].

Owing to the very large size of these evaluation datasets, prior to particle filtering we first resample the asynchronous high-frequency data onto regular time grids. This is carried out by using the most recent available price in each asynchronous time series as the next data-point on the regular time grid. It is worth noting though that for very liquid markets (such as the U.S. 10-year note), the mean number of contracts traded is $\sim$2 000 per minute, and peak rates can be several multiples greater than this [52]. Our continuous-discrete time filtering methods from previous sections are of course valid for any time discretization and could also be applied directly to the asynchronous raw data (see also [37]); this is a development that we will study in future work on this topic.

Various discretization rates are evaluated. We carry out testing on daily data for the full 5-year data set, which forms a

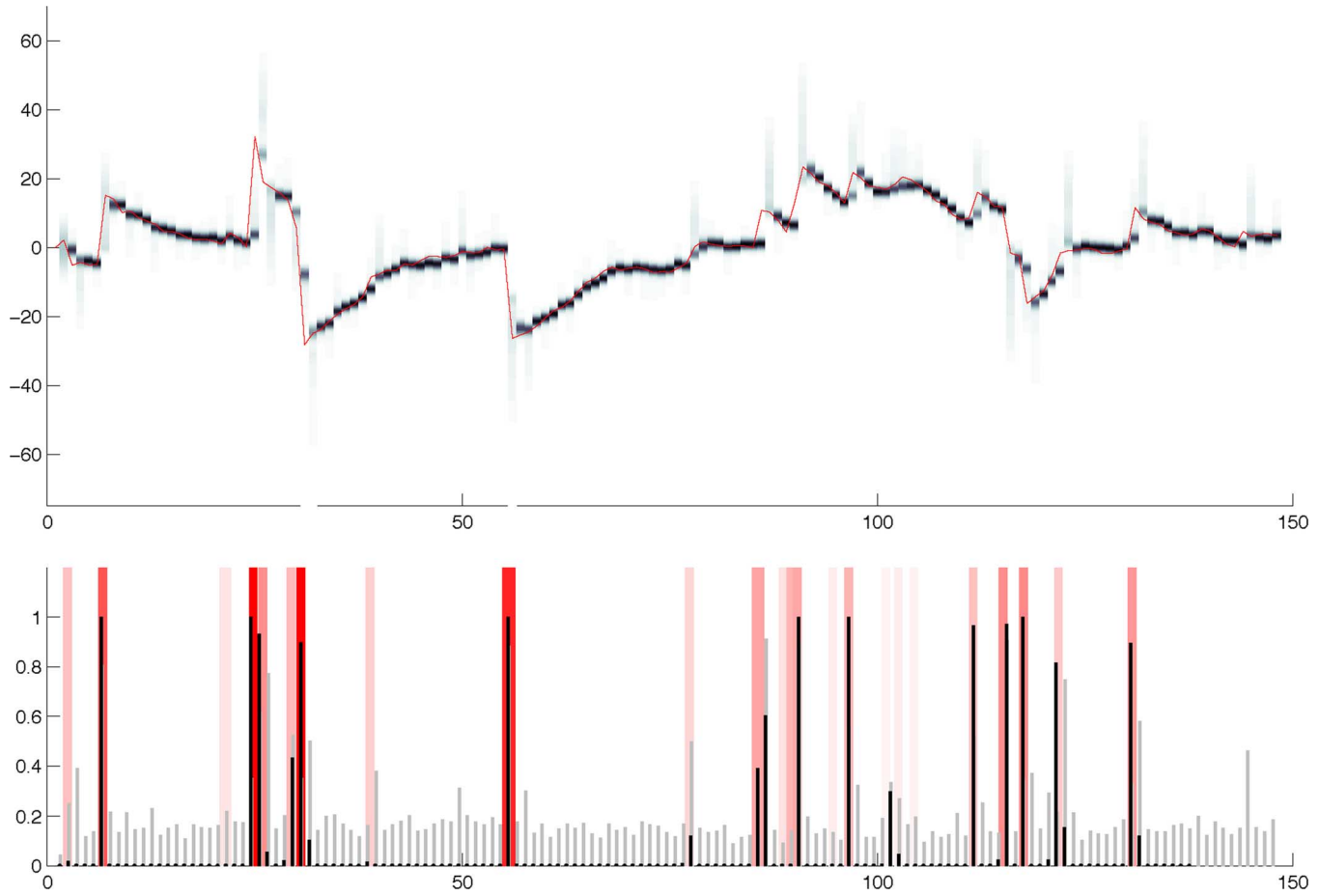

Fig. 3. Jump time detection in synthetic data. The upper graph shows the trend process (red line) and the filter distribution obtained (grey shading) using around 400 particles. The lower graph shows the proportion of particle weight held by jumping particles at each observation time. The light gray bars show the filter weight (no observation lag) and the black bars show the weight at a lag of ten observations. Background shading (red) indicates the presence of a jump with the magnitude of a jump being indicated by the intensity of the shading, with larger jumps being darker.

lower frequency performance baseline to compare with results for 15 minute (*min15*), 5 minute (*min5*), and 1 minute (*min1*), each on smaller subsets of the total data set. This is done because, while we would like to present long-run simulation results for all discretization rates, we are restricted by lack of computational power (though we note that particle filtering is well suited to parallel processing, and indeed each of the 75 contract streams may be filtered independently on different nodes). Once on a discrete time grid the data observation times are defined as $t_i = ih$, where h is the regular time difference between observations (e.g., for *min15* $h = 15 \times 60$s, etc.).

Three other data sets are also required for this study: static data for the futures contracts, costs data, and spot Foreign Exchange rate (FX) data. We define static data as data that is specific to a particular contract time series and is required to effectively trade that contract—for example contract size, currency of the contract, contract margin. Static data is required to move from our trading signal to the number of contracts to buy/sell. Costs data has multiple components including technicalities such as bid/ask spread and slippage and is essential in order to model trading in a realistic way. Spot FX data is required to carry out any currency conversion to the USD base currency of the fund as we have implemented it.

C. Implementation of the Particle Filter

Here we describe the parameter settings used for the Langevin model and the particle filter. For each contract an initial state prior and covariance μ_0 and C_0 must be assigned, see Section III. In order not to bias the filter unduly, these are initialized as $\mu_0 = [y_1, 0]^T$ and $C_0 = k\mathbf{I}$, where y_1 is the first observed price for each contract and k is a number significantly larger than the typical price variation.

The scale of the transition and observation process must resemble the scale of the observed real life price process. To this end some of our model parameters are dependent on the scale of the price process. These values are set at run-time and we define them in our parameter table as a percentage of their initial price ("SF()'), since absolute security prices vary widely and this is a convenient approximation of their scale. Process scale will also vary between sectors, with equities, for example, being more volatile than interest rates. Since the scale factors are only approximate and the price level can vary much more widely than volatilities, we do use the price level to define the scale factors, not the volatilities.

The scale factors used for parameter values were optimized manually, based on measurements of the Sharpe ratio on earlier

TABLE I
ALGORITHM PARAMETER VALUES

Parameter	J on	J off	Description
μ_J	0.0	0.0	Jump mean of x_2 process
$SF(\sigma_J)$	6.0%	0.0%	Jump volatility of x_2 process
λ_J	5.0	0.0	Jump rate of x_2 process
θ	-0.2	-0.2	Mean reversion coefficient of x_2
$SF(\sigma)$	0.35%	3.5%	Diffusion volatility for x_2
$SF(\sigma_{obs})$	35.0%	35.0%	Observation noise

time periods of data of data. The Sharpe ratio is the ratio of returns to standard deviation [53] and so is a measure of risk-adjusted return and is the dominant performance metric used in the industry. We calculate the ratio as $S = (\mu - r)/(\sigma)$, where μ is the mean strategy return, σ is the strategy return volatility and r is the risk free rate.

Table I presents the values chosen, based on data from all 75 contracts and for daily data. We do this for jumps turned on ("J on") and for jumps turned off ("J off"). Further optimization at sector-specific level is likely to improve performance in future work. Parameter values and thus performance might be improved by more sophisticated estimation methods (see [54], [55]).

D. Signal Generation

Sections II and III describe in detail how the proposed Langevin model is defined, and how to carry out Bayesian sequential inference in that model. The output of that sequential inference is a Monte Carlo estimate of the posterior density and posterior means for the hidden state X_t given all of the observed data up to the current observation time. The flexible form of the Bayesian Monte Carlo scheme allows many other posterior inferences to be made, including predictions of future price and smoothed state estimates. These all come in the form of probability densities and can in principle be carried forward into optimal decision-making processes about whether to buy or sell particular commodities ("contracts"). This will be a complex process, however, for the large numbers of contracts that we are considering simultaneously within our evaluations. Hence we adopt a somewhat simplistic approach in which a particle filter is run for each of 75 contract time series, split into eight different market sectors. We then generate buy and sell decisions (or "signals" in trading terminology) using the posterior mean outputs $\hat{X}_{t_i}$ from each of the 75 particle filters [see (44)].

A number of processing steps are carried out in order to generate a reliable set of 'signals' across the portfolio. For reasons of computational simplicity as before we use the expected value of the price as a point estimate for 'signal' generation, i.e., we use the Monte Carlo estimate of $\hat{x}_{1,t} = \mathbb{E}(x_{1,t} \mid y_{1:t})$, as presented in Section III. Other forms of output from the particle filter were experimented with, but this was found to be simple and reliable when combined with the signal generation process below. Full details of the signal processing operations required in order to go from a set of particle filter estimates $\hat{x}_{1,t}$ (one for each contract) to a set of buy or sell "signals" for each contract are given in the supporting document [51], since these are fairly standard operations in many trading systems. Briefly, though, the steps are as follows:

1) **Differencing and Smoothing**. For each contract we use the sign of the time-differenced price estimate $\Delta_t = \mathrm{sgn}(\hat{x}_{1,t} - \hat{x}_{1,t-1})$ as an instantaneous estimate of the predicted price change. This binary signal is then smoothed with a four-step FIR smoothing filter in order to measure the persistence of any momentum effect, leading to a continuous-valued momentum indicator Δ_t^s.
2) **Transfer Function**. An empirical nonlinear function is applied to each Δ_t^s as follows: $[Z_t \propto (\Delta_t^s)/(\Delta_t^{s2} + {\sigma_\Delta}^2)]$ where σ_Δ is the measured sample standard deviation of Δ_t^s from the recent past [56]. In the case of a momentum-based model, such a transfer function has an understandable economic basis, implying that momentum has a linear effect on price around the origin (for small values of Δ_t^s) but at high signal levels the effect tails off, since mean-reversion effects tend to kick-in when trends become too severely up or down.
3) **Signal Volatility Scaling**. The volatility for each contract time series is determined using an IGARCH(1,1) model [57] and the Z_t rescaled for each contract by dividing by its estimated volatility. This attempts to normalize the volatility of the signals within each of the eight market sectors [58].

The volatility-scaled signals for each contract are then used to determine a desired position for each contract, i.e., how many "lots" (items) of each contract should be present in the portfolio, and hence how many need to be bought or sold at the current time step. This includes targeting a desired value at risk (VaR) level by gearing the fund accordingly [59]. In dynamically controlling risk in this way, by being able to take a view on where we are in the volatility space, we are able to optimally lever the fund and increase its Sharpe ratio [60]. The effect of such a step is that as market volatility rises, the position sizes taken by the algorithm get smaller, and so the impact of the volatility is muted, while the realized volatility is constant [61].

In summary, the size of the position held in any single given security is primarily a function of the value of the signal for that security and the correlation matrix of the portfolio returns.

E. Simulation Methodology

In this section, we describe how the performance of our complete trading algorithm is tested on a large set of past data—the process known as *backtesting*. Our system attempts to keep the process as simple as possible while at the same time capturing real-life features.

As for signal generation, more complete details of the simulation methodology from this section are covered in the supporting document [51].

In order to backtest the algorithm, a notional fund of $1 billion is constructed, having eight equally weighted sectors, and with the securities in each sector being also equally weighted. This results in $12.5M being placed in each of the 75 securities, which is consistent with the large average daily transaction volumes of these liquid contracts. Evaluation is carried out in terms of the profit and loss (PnL) across the entire fund, calculated on a daily basis at 1800 h GMT, and then using the FX spot x-rates to convert to U.S. Dollars (USD), which is then aggregated to give annualized figures.

Realistic representation of execution (the actual process of buying or selling a contract, often done by placing an "order" in a simple queuing system (the "order book") is perhaps the most difficult part of HF simulation, since it is likely that large trades would have significantly impacted the market (a feedback effect), an effect that cannot be replicated without real trading in a real market. At low trading frequencies, for liquid securities, it is quite accurate to assume a zero feedback model, whereby transactions do not significantly affect the rest of the system; at high frequency, however, this is not necessarily the case, since any trades will cause the system to change significantly, an effect that should ideally be accounted for in simulation. This is very hard to do given a fixed set of historical data and presents an additional inference problem [62]. For the sake of simplicity, we assume a static slippage model where the slippage incurred is a fixed percentage of the dollar size of the transaction.

Costs (i.e., those in addition to notional value of a contract) are an important part of trading and thus we incorporate a three-part cost model comprising slippage (as above), transaction costs and bid-ask spread. The transaction costs we use are typical figures from a prime broker futures direct market access (DMA) desk and cover the cost of exchange fees and brokerage fees etc. The bid-ask spread is charged twice for every round trip (buy and sell) trade. Exact values used are specified in [51].

The full trading system is then run, generating particle filter estimates for all contracts, trading signals and updated portfolio positions at each time. The overall PnL values can then be computed at the end of each day, including realistic cost estimates as above.

In order to evaluate these PnL values we focus on the Sharpe ratio. As we are trading futures, interest is earned at the risk free rate on the short positions, which means we do not need to include a risk free rate when calculating Sharpe. Interest paying long positions are balanced with interest receiving short positions, as the signal is zero mean for each asset.

While no performance metric is able to tell the whole story, the Sharpe ratio avoids some of the serious drawbacks associated with metrics such as performance relative to a benchmark (such as "long-only" or "constant proportion rebalancing") as a benchmark must consist of comparable risk factors (a variable associated with an increased risk of losses) and many risk factors from hedge fund style strategies are very hard to replicate in benchmark form [58].

F. Real Data Results

First, we illustrate that the signals generated by the proposed method are correlated with the actual returns, and hence that the method has some predictive power. This is carried out by looking at the signals generated by step 1. of the signal generation procedure (i.e., just prior to application of the nonlinear transfer function). Actual realized returns for the contract are plotted in risk-adjusted form, i.e., by normalizing for volatility (estimated as before using an IGARCH(1,1) model): such an adjustment allows clear representation of returns independent of market volatility. The result of a bootstrapping procedure is shown, in which the mean and 95% confidence intervals are shown for expected returns corresponding to particular values of our proposed signal. Bootstrapping is a resampling method,

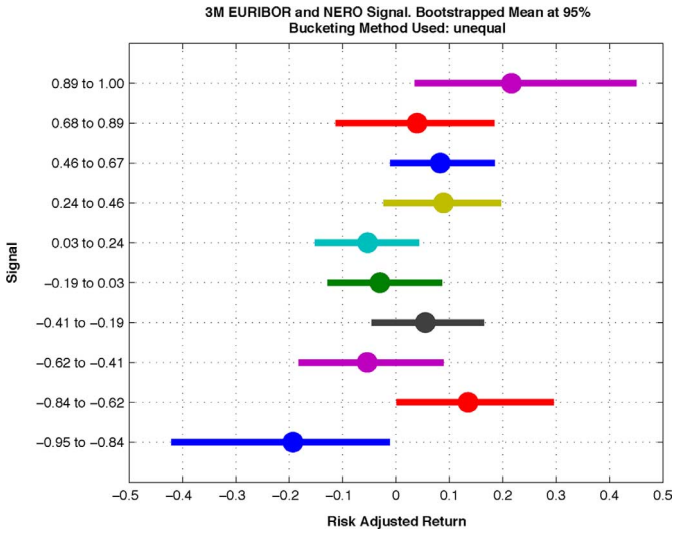

Fig. 4. Testing for statistical significance by bootstrapping. For the short term interest rate future, EURIBOR, we see our algorithm is able to separate between bands of returns. The "mid" of each bar is the mean with the "spread" representing the 95th percentiles. (NERO: Nonlinear Evolving Rao–Blackwellized Online algorithm).

which allows the calculation of the sample distribution of almost any statistic [11], [63]. We use bootstrapping to inspect the relationship of our predictive signal to the risk-adjusted market returns by bootstrapping the mean at the 95th percentile. We implement the procedure by histograming the data and then sampling with replacement 1000 times and calculating the mean and the confidence intervals of the distribution of the means. For signals which have clear predictive power over the market returns we would expect to see a high degree of correlation between signal and returns, and also good separation between means for different signal values. Fig. 4 demonstrates for a typical contract that the signal has the predictive power we are looking for, while supporting the hypothesis of the nonlinearity of strategy returns.

Having illustrated that there is a strong relationship between the market returns and the proposed signal, we look to see if we can profitably exploit it, taking costs into account. We start with a simulation at daily frequency, using all 75 contracts over the period 1st January 1996 to 1st January 2011. While we do not consider this to be "high-frequency," as noted in Section IV-B, we do not have the computational power available to us to run such a full simulation with high-frequency data. Additionally, knowing the performance at a daily frequency will then also allow us to compare performance at higher frequencies.

The *fundamental law of forecasting* [64], [65] describes an *ex-ante* relationship between expected performance and the assumed information coefficient, IC, of the forecasting process. Expected performance is measured by the ex-ante information ratio IR (equivalent to Sharpe ratio in our case)

$$\mathrm{IR} = \mathrm{IC}\sqrt{N} \tag{45}$$

where N is the number of independent "bets" in the portfolio. The law tells us that Sharpe is proportional to the breadth of the application, as well as the quality of the signal. Grinold states the law "gives us only an upper bound on the value we can add"

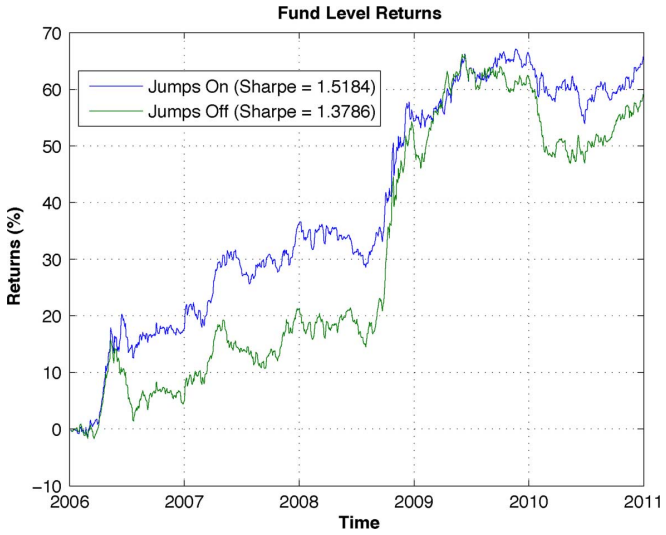

Fig. 5. Long run fund performance (75 futures contracts) using daily frequency data (post-cost). The use of daily frequency data allows a comparison of the algorithm to commercial hedge fund products [67].

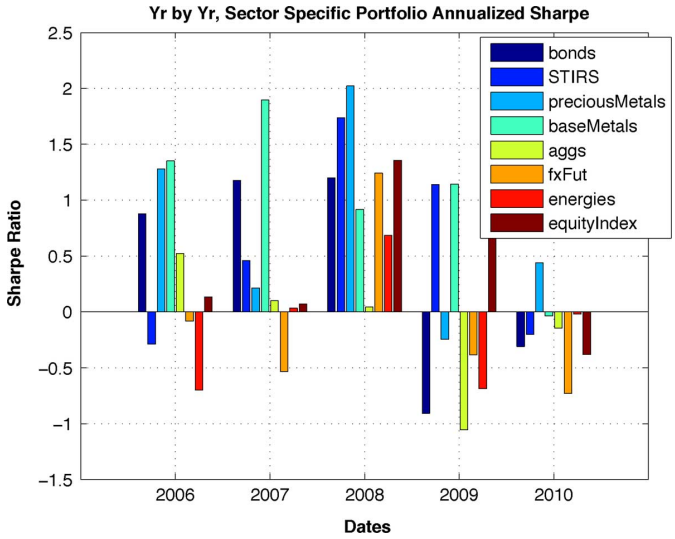

Fig. 6. Sector performance in the fund, annualized (Sharpe). When the performance is broken down over time and sector, we see there is a large amount of variation in each sector. This undesirable characteristic is negated by trading a large, diversified portfolio.

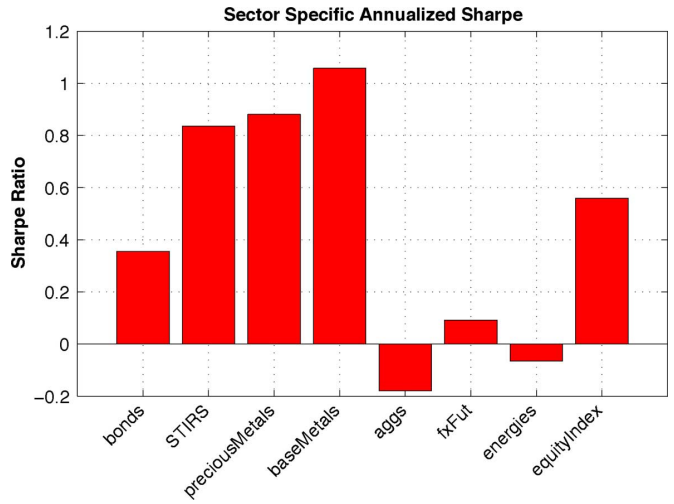

Fig. 7. Sector performance in the fund, overall (Sharpe). By breaking down the fund performance, we can see some sectors tend to perform better than others. This may be partially explained by the poor performing sectors (such as agriculturals) having higher transaction costs. Note that the fund Sharpe is greater than the individual sector Sharpe's, explained by the effect of diversification on the portfolio.

and thus while theoretically Sharpe scales proportionally to frequency, this does not account for issues such as transaction cost and so we do not expect to see a direct scaling between frequency and Sharpe.

Equation (45) is not saying that the size of the expected returns and their volatility remains constant despite changing the holding period, it saying that unless you have a good reason to believe otherwise, the information content of a certain effect will be constant across the frequency spectrum.

The results of this daily simulation give an annualized post-cost Sharpe ratio of ~ 1.52, while the cumulative returns are shown in Fig. 5. When compared to major hedge funds which also trade momentum strategies using daily frequency data, we see our algorithm's performance is similar, with losses in early 2010, which suggests to us that we are to some degree successfully extracting momentum information from the market [66], [67].

Looking at the breakdown of the returns by year and sector (Figs. 6 and 7), we see that the sectors with the highest transaction costs (as per [51]) have the worst performance and that for any given sector, the annualized Sharpe ratio varies widely over time. These observations would suggest that portfolio construction could be improved by a constrained mean-variance process (as opposed to the current equal weighting scheme) allowing more weight to be given to better performing sectors, conditional on transaction costs [58], but this is not explored here.

We next turn the jump estimation part of the algorithm off (as per Table I) and reevaluate performance. With jumps turned off (and hence estimation is done solely using the Kalman filter), we see that there is a 25% drop in the Sharpe ratio to about 1.37 on daily data. When analyzed using bootstrap statistics [68], [69], at the 95th percentile, the decrease in performance is shown to be statistically significant (Fig. 8). This result confirms our hypothesis that the underlying returns process is better modeled by jump-diffusion than just a pure diffusion process.

At the higher frequencies of fifteen minutes ($N = 48$, Fig. 9) and one minute ($N = 720$, Fig. 10), evaluated over a much shorter period of one calendar month, for computational reasons, we find that the results are in agreement with those using daily data in terms of absolute returns, but show a significant post-cost Sharpe ratio greater than that for the daily data, which is in agreement with (45).

We are not surprised that the higher frequency results seem to mimic the effect seen at lower frequency, albeit with better performance. If an effect exists in the financial markets, we would expect it to exist throughout the frequency spectrum. For example, momentum effects exist in the market due to the actions of different classes of traders. A low-frequency trader, such as a pension fund, moves money over a long period of time and has a long holding period. At the other end of the spectrum are high-frequency algorithmic hedge funds, who move very quickly. In between them is a near-continuous spectrum of classes of investors, who operate at different frequencies. If

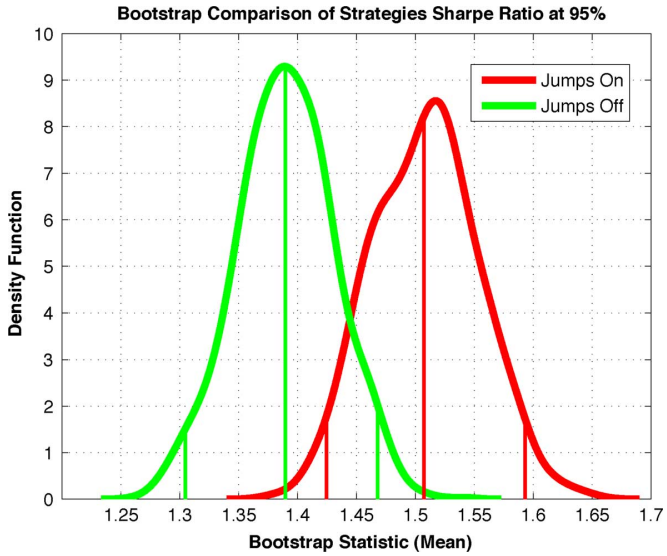

Fig. 8. Algorithm with jumps versus no jumps. Bootstrapping allows comparison of the two strategies [69]. The presence of jump detection improves the algorithms performance, suggesting the futures markets are well modeled by a jump-diffusion process.

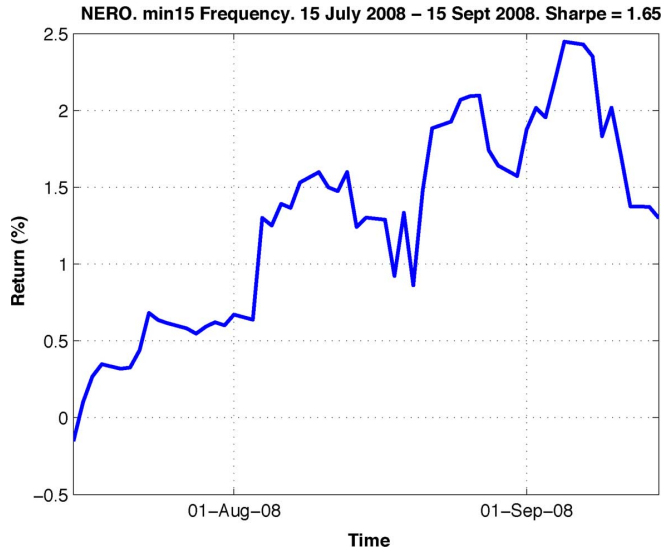

Fig. 9. Fifteen-minute frequency. 15 July 2008–15 Sept. 2008. Whole fund (75 futures contracts). The annualized Sharpe is greater than the lower frequency daily data by nearly 10%.

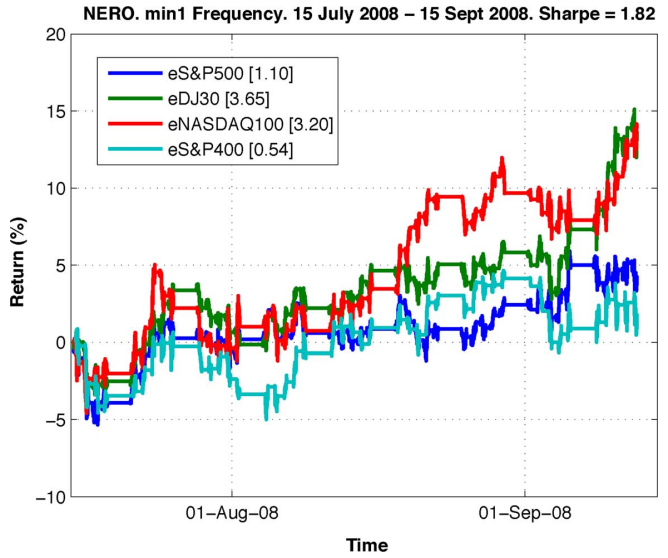

Fig. 10. One minute frequency. 15 July 2008–15 Sept. 2008. Four equity index contracts only. We estimate this to be the maximum frequency at which the algorithm in its current form can operate profitably, due to the fact the market orders it places need to cross the bid-ask spread.

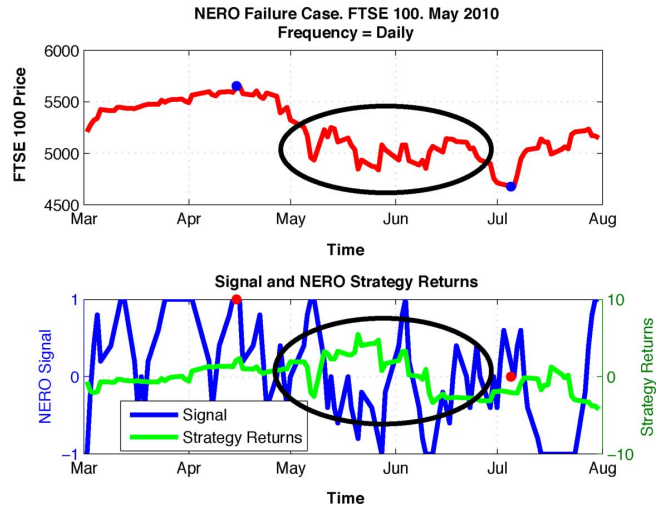

Fig. 11. Failure Case: cyclic behavior causes negative performance. What appears to be a trend at a high frequency will be cyclic at another, lower, frequency. The inherent lag in the filter (for a given set of filter parameters) means that the change points cannot be detected across all frequencies.

other traders exploit momentum more at some frequencies than others, then we would not expect to see our algorithm performance scale exactly as per the fundamental law of forecasting, as the effect will have been arbitraged away.

G. Failure Case

For any single trading strategy, there will be market conditions under which the strategy fails to perform well. We see three principal failure cases for the algorithm. These are seasonal variation, high-frequency noise and cyclic (non-trending) behavior. We briefly look at seasonality and cyclic behavior, while high-frequency noise is generically accounted for by our Gaussian observation noise model (albeit with a performance degradation when high-frequency noise is predominant).

Fig. 11 shows price for the FTSE100 and the strategy return in mid 2010 and is an example of losses related to non-trending behavior. The FTSE reaches a local maximum on 15th April 2010 and moves to a local minimum on 5th July 2010, but in between displays cyclic behavior. For this period, the algorithm performs negatively because the filter can take some time to react to turning points in the data, leading to a forecast signal with the wrong sign and in the worst case, being in anti-phase with the price. This behavior could be improved upon by an improved signal generation mechanism in which the particle filter's output is not smoothed so extensively, though this could be at the cost of stability in more persistently trending markets.

The existence of seasonality is well established in financial data, and occurs on the intraday timescale too [70]. Such patterns occur over the space of the trading day, for example on morning-afternoon and inter-hour timescales. The presence of seasonalities may act to obscure the underlying low-frequency dynamics by decreasing the autocorrelation function [71]. In

order to prevent such cyclic patterns masking the momentum effect we could attempt to preprocess our data by removing these harmful frequencies [72] while leaving the underlying inherent nonseasonal structure intact. Once the core momentum signal has been calculated on the preprocessed data, the seasonality time series could be added back into the signal prior to the transform function—alternatively a more elaborate state-space model that includes intra-day seasonal components as well as our Langevin dynamics could help here.

V. Conclusion and Further Work

In this paper, we have proposed a specialized jump-diffusion model of asset prices and shown how we can use this to extract a price trend from high-frequency asset data. By conditioning on jump times we were able to derive closed-form expressions for the transition densities in our model and this has allowed us to apply an efficient Rao–Blackwellized variable rate particle filter algorithm in order to extract the state of the underlying model, including timings of jumps. Results have been evaluated in quite a realistic way by constructing a signal generation system around the particle filter outputs and a performance metric based on simulated PnL and Sharpe ratio.

The results seem quite successful when applied to an extensive and recent data set of the world's most liquid futures contracts. The daily frequency momentum signal generated by our algorithm has a high correlation to the returns generated by major trend-following hedge funds, suggesting that we are indeed capturing momentum. With the average major momentum hedge fund returning a Sharpe ratio of around 1.0 [73], our model seems to compare well in terms of absolute performance. We have found experimental evidence that suggests the underlying price dynamics of futures contracts are better modeled a jump-diffusion process than by a pure diffusion process.

Whilst the level of sophistication of algorithms in the financial services industry is still quite basic, this work suggests that more sophisticated approaches might find real application in this area, even given its high computational complexity, which presents new software and hardware challenges.

Our current work in the area is seeking accurate and adaptive parameter estimation techniques for these classes of model. We are also constructing optimal decision-theoretic methods which incorporate the full Bayesian state distribution (rather than simple point estimators) and appropriate utility functions that account for costs and returns in a realistic way. In future work we also hope to be able to perform out of sample testing by *paper trading* the algorithm via an API such as that provided by www.interactivebrokers.co.uk. Such paper trading is identical in every detail to real trading, except that the orders are not executed on the exchange and simulated fills are provided instead. In this way we hope to achieve greater realism in the evaluation of the algorithms.

Acknowledgment

The authors would like to thank both www.tickdata.com and www.financialcalendar.com for allowing them to use their data sets in this research.

References

[1] P. Zubulake, "High frequency trading in the futures markets," Aite Group Rep., May 2010.
[2] B. Malkiel, "The efficient market hypothesis and its critics," *J. Econ. Perspect.*, vol. 17, no. 1, pp. 59–82, 2003.
[3] E. Fama, "Efficient capital markets: Ii," *J. Finance*, pp. 1575–1617, 1991.
[4] N. Jegadeesh and S. Titman, "Returns to buying winners and selling losers: Implications for stock market efficiency," *J. Finance*, pp. 65–91, 1993.
[5] D. Vayanos and P. Woolley, "An institutional theory of momentum and reversal," NBER Working Papers, 2008.
[6] D. Lesmond, M. Schill, and C. Zhou, "The illusory nature of momentum profits," *J. Financial Econ.*, vol. 71, no. 2, pp. 349–380, 2004.
[7] E. Fama, "Efficient capital markets: A review of theory and empirical work," *J. Finance*, vol. 25, no. 2, pp. 383–417, 1970.
[8] L. Menkhoff and M. P. Taylor, "The obstinate passion of foreign exchange professionals: Technical analysis," *J. Econ. Lit.*, vol. 45, no. 4, pp. 936–972, 2007.
[9] A. Lo and A. MacKinlay, *A Non-Random Walk Down Wall Street*. Princeton, NJ: Princeton Univ. Press, 2001.
[10] A. Lo, H. Mamaysky, and J. Wang, "Foundations of technical analysis: Computational algorithms, statistical inference, and empirical implementation," *J. Finance*, pp. 1705–1765, 2000.
[11] R. Levich *et al.*, "The significance of technical trading-rule profits in the foreign exchange market: A bootstrap approach," *J. Int. Money Finance*, vol. 12, no. 5, pp. 451–474, 1993.
[12] H. Hong, T. Lim, and J. Stein, "Bad news travels slowly: Size, analyst coverage, and the profitability of momentum strategies," *J. Finance*, pp. 265–295, 2000.
[13] N. Jegadeesh and S. Titman, "Profitability of momentum strategies: An evaluation of alternative explanations," *J. Finance*, vol. 56, no. 2, pp. 699–720, 2001.
[14] K. Chan, A. Hameed, and W. Tong, "Profitability of momentum strategies in the international equity markets," *J. Financial Quantit. Anal.*, vol. 35, no. 2, pp. 153–172, 2000.
[15] J. Okunev and D. White, "Do momentum-based strategies still work in foreign currency markets?," *J. Financial Quantit. Anal.*, vol. 38, no. 02, pp. 425–447, 2003.
[16] W. Fung and D. Hsieh, "The risk in hedge fund strategies: Theory and evidence from trend followers," *Rev. Financial Studies*, vol. 14, pp. 313–341, 2001.
[17] J. Miffre and G. Rallis, "Momentum strategies in commodity futures markets," *J. Banking Finance*, vol. 31, pp. 1863–1886, 2007.
[18] R. Shiller, *Irrational Exuberance*. Princeton, NJ: Princeton Univ. Press, 2005.
[19] T. Johnson, "Rational momentum effects," *J. Finance*, pp. 585–608, 2002.
[20] S. Jones, *AHL Strength a Positive for Man Group*. London, U.K.: Financial Times, Sep. 2011.
[21] S. Schulmeister, "Profitability of technical stock trading: Has it moved from daily to intraday data?," *Rev. Financial Econ.*, vol. 18, no. 4, pp. 190–201, 2009.
[22] A. Gerald, *Technical Analysis Power Tools for Active Investors*. Upper Saddle River, NJ: Financial Times Prentice-Hall, 1999.
[23] P. Craigmile and D. Percival, *Wavelet-Based Trend Detection and Estimation*. New York: Encyclopedia of Environmetrics, Wiley, 2002.
[24] X. R. Li and V. Jilkov, "Survey of maneuvering target tracking. Part I. dynamic models," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 39, no. 4, pp. 1333–1364, Oct. 2003.
[25] E. Fama, "The behavior of stock-market prices," *J. Business*, vol. 38, no. 1, pp. 34–105, 1965.
[26] C. Granger and Z. Ding, Stylized facts on the temporal and distributional properties of daily data from speculative markets, Dept. of Economics, Univ. of California San Diego. La Jolla, 1994.
[27] D. M. Guillaume, M. M. Dacorogna, R. R. Dave, U. A. Muller, R. B. Olsen, and O. V. Pictet, "From the bird's eye to the microscope: A survey of new stylized facts of the intra-daily foreign exchange markets," *Finance Stochast.*, vol. 1, pp. 95–129, 1997.
[28] R. Cont, "Empirical properties of asset returns: Stylized facts and statistical issues," *Quantit. Finance*, vol. 1, no. 2, pp. 223–236, 2001.
[29] R. F. Engle, "The econometrics of ultra-high-frequency data," *Econometrica*, vol. 68, no. 1, pp. 1–22, 2000.
[30] O. Jorda and M. Marcellino, "Modeling high frequency FX data dynamics, 2002.
[31] P. A. Mykland and L. Zhang, The econometrics of high frequency data, 2009.
[32] T. Bollerslev, "Generalized autoregressive conditional heteroskedasticity," *J. Econometrics*, vol. 31, pp. 307–327, 1986.

[33] F. Black and M. Scholes, "The pricing of options and corporate liabilities," *J. Political Econ.*, vol. 81, no. 3, pp. 637–654, 1973.
[34] S. Heston, "A closed-form solution for options with stochastic volatility with applications to bond and currency options," *Rev. Financial Studies*, vol. 6, pp. 327–343, 1993.
[35] S. Kou, "A jump-diffusion model for option pricing," *Manage. Sci.*, vol. 48, pp. 1086–1101, 2002.
[36] O. E. Barndorff-Nielsen and N. Shepherd, *Modelling by Levy Processes for Financial Econometrics (in Levy Processes: Theory and Applications)*. New York: Springer/Birkhauser, 2001, pp. 283–318, Applied Probability and Statistics.
[37] S. Godsill, "Particle filters for continuous-time jump models in tracking applications," in *Proc. ESAIM*, 2007, vol. 19, pp. 39–52.
[38] L. Sun and C. Stivers, "Cross-sectional return dispersion and time variation in value and momentum premium," *J. Financial Quantit. Anal.*, vol. 45, pp. 987–1014, 2010.
[39] S. Godsill and J. Vermaak, "Variable rate particle filters for tracking applications," in *Proc. IEEE/SP 13th Statist. Signal Process. Workshop*, 2005, pp. 1280–1285.
[40] S. Godsill, J. Vermaak, W. Ng, and J. Li, "Models and algorithms for tracking of maneuvering objects using variable rate particle filters," *Proc. IEEE*, pp. 925–952, 2007.
[41] P. Kloeden and E. Platen, *Numerical Solution of Stochastic Differential Equations*. New York: Springer, 1992.
[42] S. Sarkka, "Recursive Bayesian inference on stochastic differential equations," Doctoral dissertation, Lab. of Comput. Eng., Helsinki Univ. of Technol., Espoo, Finland, 2006.
[43] P. Langevin, "On the theory of brownian motion," *C. R. Acad. Sci. (Paris)*, vol. 146, pp. 530–533, 1908.
[44] A. Doucet, S. Godsill, and C. Andrieu, "On sequential Monte Carlo sampling methods for Bayesian filtering," *Statist. Comput.*, vol. 10, no. 3, pp. 197–208, 2000.
[45] O. Cappe, S. Godsill, and E. Moulines, "An overview of existing methods and recent advances in sequential Monte Carlo," *Proc. IEEE*, vol. 95, no. 5, pp. 899–924, May 2007.
[46] R. Kalman, "A new approach to linear filtering and prediction problems," *J. Basic Eng.*, vol. 82, pp. 35–45, 1960.
[47] Y. Ho and R. Lee, "A Bayesian approach to problems in stochastic estimation and control," *IEEE Trans. Autom. Control*, vol. AC-9, no. 4, pp. 333–339, Oct. 1964.
[48] A. Harvey, *Forecasting, Structual Timeseries Models and the Kalman Filter*. Cambridge, U.K.: Cambridge Univ. Press, 1990.
[49] M. Grewal and A. Andrews, *Kalman Filtering: Theory and Practice Using Matlab*. New York: Wiley-IEEE Press, 2008.
[50] R. Douc, O. Cappe, and E. Moulines, "Comparison of resampling schemes for particle filtering," in *Proc. 4th Int. Symp. Image Signal Process. Anal.*, 2005, pp. 64–69.
[51] H. Christensen, S. Godsill, and J. Murphy, Supplementary material to forecasting high-frequency futures returns using online Langevin dynamics, [Online]. Available: http://www-sigproc.eng.cam.ac.uk/~hlc54/pubs/2012/supp.PDF
[52] J. Field and J. Large, Pro-rata matching and the order book dynamics of fixed income futures Univ. of Oxford, Oxford, U.K., Tech. Rep., 2007.
[53] W. Sharpe, "The Sharpe ratio," *J. Portfolio Manage.*, vol. 21, pp. 49–58, 1994.
[54] M. Johannes and N. Polson, "MCMC methods for financial econometrics," in *Handbook of Financial Econometrics*. Amsterdam, The Netherlands: North Holland, 2002.
[55] J. Murphy, "Bayesian methods for high frequency financial time series analysis," Ph.D. First Year Report, Cambridge Univ. Dept. of Eng., Cambridge, U.K., 2010.
[56] W. Ferson and A. Siegel, "The efficient use of conditioning information in portfolios," *J. Finance*, vol. LVI, no. 3, pp. 967–982, 2001.
[57] P. Zaffaroni, "Large-scale volatility models: Theoretical appraisal of professionals practice," *J. Time Series Anal.*, vol. 29, no. 3, pp. 581–599, May 2008.
[58] R. Kahn and R. Grinold, *Active Portfolio Mangement*. New York: McGraw-Hill, 1999.
[59] P. Jorion, *Value at Risk: The New Benchmark for Managing Financial Risk*. New York: McGraw-Hill, 2006.
[60] N. Papageorgiou, A. Hocquard, and S. Ng, "A constant volatility framework for managing tail risk," Brockhouse Copper White Paper, Sep. 2010.
[61] "Interview with Dr. Ewan Kirk, CIO of cantab capital partners LLP," Eurekahedge, Nov. 2009.
[62] M. Amos, "Market data-driven simulation of order book mechanics," U.S. patent publication number: US 2008/0243572 A1, 2008.
[63] D. H. A. Davison, *Bootstrap Methods and Their Application*, ser. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge, U.K.: Cambridge Univ. Press, 1997.
[64] R. Grinold, "The fundamental law of active management," *J. Portfolio Manage.*, vol. 15, no. 3, pp. 30–37, 1989.
[65] R. Grinold, "Alpha is volatility times IC times score, or real alphas don't get eaten," *J. Portfolio Manage.*, vol. 20, no. 4, pp. 9–16, 1994.
[66] T. Cahill and A. Xydias, "Futures funds fall most since 1987 as AHL, Henry miss," *Bloomberg*, Jan. 2010 [Online]. Available: http://www.bloomberg.com/apps/news?pid=newsarchive&sid=aMMX8YwsyFnM
[67] Man Group PLC, [Online]. Available: www.mangroupplc.com/assets/excel/analyst-workbook.xls AHL monthly NAV.
[68] A. Zoubir and B. Boashash, "The bootstrap and its application in signal processing," *IEEE Signal Process. Mag.*, vol. 15, no. 1, pp. 55–76, Jan. 1998.
[69] O. Ledoit and M. Wolf, "Robust performance hypothesis testing with the Sharpe ratio," *J. Empir. Finance*, vol. 15, pp. 850–859, 2008.
[70] R. Huptas, "Intraday seasonality in analysis of UHF financial data: Models and their empirical verification," *Dynam. Econom. Mod.*, vol. 9, pp. 1–10, 2009.
[71] R. Gencay, F. Selcuk, and B. Whitcher, *An Introduction to Wavelets and Other Filtering Methods in Finance and Economics*. Amsterdam, The Netherlands: Elsevier, 2002.
[72] J. Murphy, "The seasonality of risk and return on agricultural futures positions," *Amer. J. Agricult. Econ.*, vol. 69, pp. 639–646, 1987.
[73] Barclay CTA Index, [Online]. Available: www.barclayhedge.com

Hugh L. Christensen received the undergraduate and M.Sc. degrees from Oxford University, Oxford, U.K. He is currently pursuing the Ph.D. degree with the Signal Processing Laboratory, Cambridge University, Cambridge, U.K.

He has worked as a Researcher for the U.K. Government and subsequently in a number of quantitative hedge funds and banks designing high-frequency algorithmic trading platforms and strategies. His research interests include online methods for time series prediction and signal combination.

James Murphy received the undergraduate degree from Cambridge University, Cambridge, U.K., and the M.Sc. degree in mathematical modeling from Oxford University, Oxford, U.K. He is currently pursuing the Ph.D. degree in the Signal Processing Group, Engineering Department, Cambridge University.

Before this, he worked as a Senior Analyst with a quantitative finance consulting firm. His research interests include Monte Carlo methods for inference in univariate and multivariate time series problems.

Simon J. Godsill (M'93)

He is the Professor of Statistical Signal Processing in the Engineering Department, Cambridge University, Cambridge, U.K. He has research interests in Bayesian and statistical methods for signal processing, Monte Carlo algorithms for Bayesian problems, modeling and enhancement of audio and musical signals, tracking, and high-frequency financial data. He has published extensively in journals, books and conferences. He is currently co-organizing a year-long program on Sequential Monte Carlo Methods at the SAMSI Institute in North Carolina.

Prof. Godsill has acted as an Associate Editor for the IEEE TRANSACTIONS ON SIGNAL PROCESSING and the *Journal Bayesian Analysis*, and as a member of IEEE Signal Processing Theory and Methods Committee. He has coedited in 2002 a special issue of the IEEE TRANSACTIONS ON SIGNAL PROCESSING on Monte Carlo Methods in Signal Processing and organized many conference sessions on related themes.