ESTIMATION OF TIME-DEPENDENT MODELS

Samuel Monnier

Section de Mathématiques, Université de Genève

Geneva, Machine Learning Club, April 9, 2019

Inspired by *Quantitative Equity Investment Management with Time-Varying Factor Sensitivities* - Bentz

Outline

Outline

TIME SERIES DATA

In this talk we will be interested in time series data.

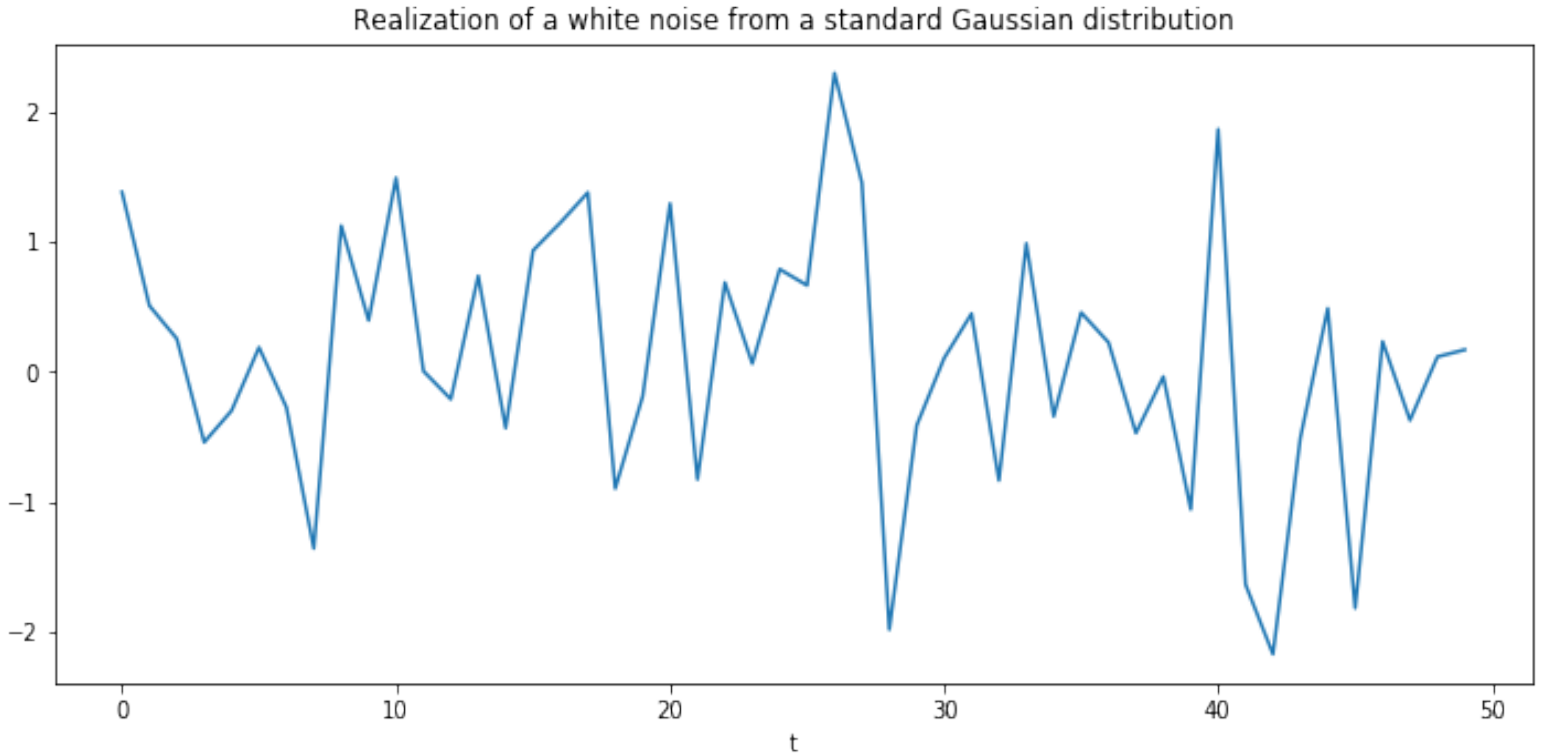
A *time series* is a set of values $\{x_t\}$ indexed by timestamps: $t \in T \subset \mathbb{R}$.

For simplicity, we will only consider discrete, evenly spaced time series, i.e. $T = \mathbb{Z}$.

A *stochastic time series* is a sequence of random variables $\{X_t\}$, $t \in \mathbb{Z}$.

TIME SERIES DATA

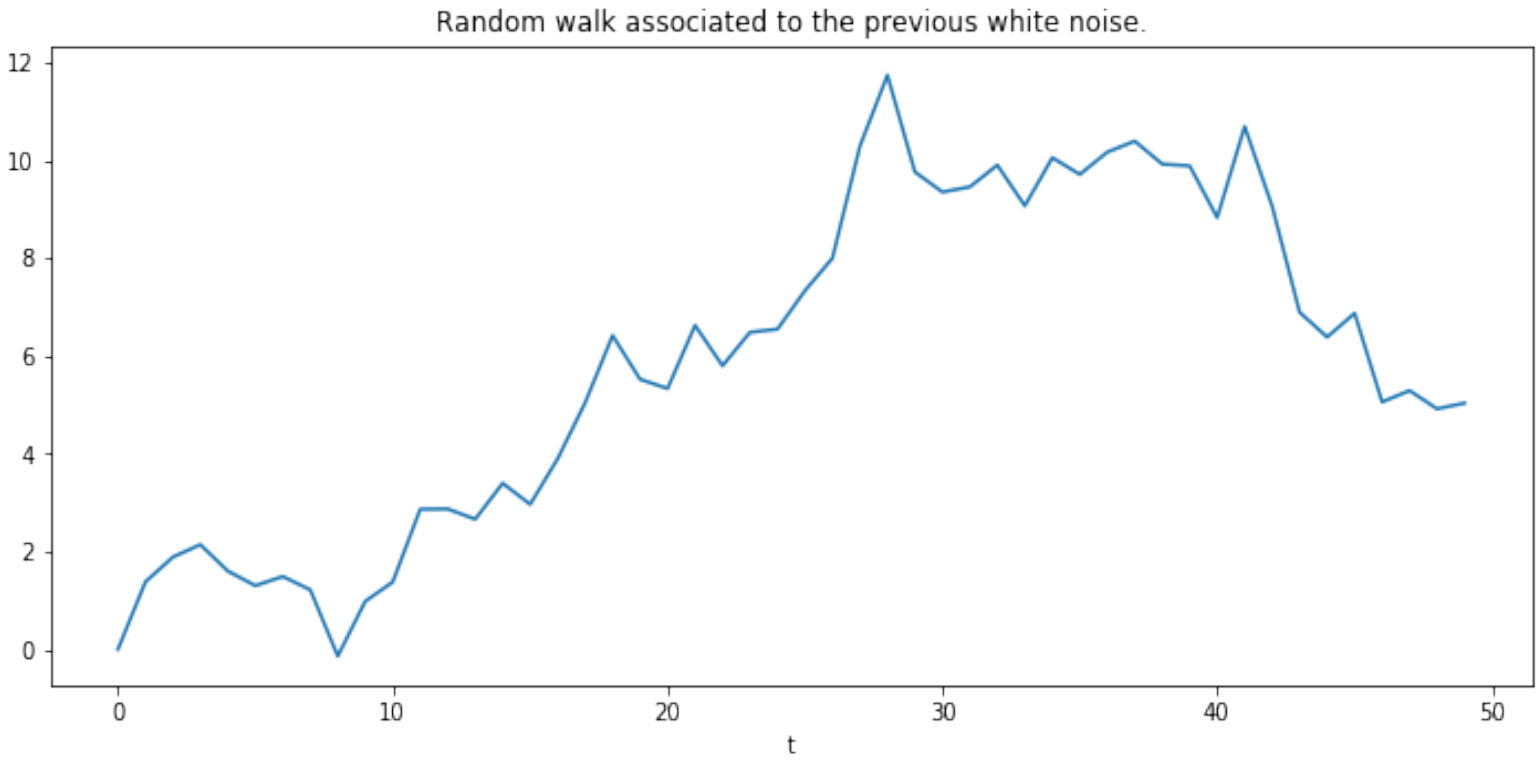
If the random variables $\{X_t\}$ are independent and identically distributed with zero mean, the time series is a *white noise*.



TIME SERIES DATA

A time series X can be *integrated* into a time series Y by defining $Y_t = X_t + Y_{t-1}$, $Y_0 = 0$.

A time series integrating a white noise is a random walk.



TIME SERIES DATA

Conversely, a time series Y_t can be differenced, by defining $X_t = Y_t - Y_{t-1}$.

A time series $\{X_t\}$ is *(weakly) stationary* if

1. The expectation $\mathbb{E}(X_t)$ is a constant μ as a function of t;
2. The j-lagged autocovariance $\mathbb{E}((X_t - \mu)(X_{t-j} - \mu))$ is constant as a function of t for all lags j.

A white noise is weakly stationary. A random walk is not.

A times series is *strongly stationary* if for any set of integers $(j_1, ..., j_n)$, the joint distribution of $(X_t, X_{t+j_1}, ..., X_{t+j_n})$ is independent of t.

MACHINE LEARNING WITH TIME SERIES DATA

Given time series realizations $\{x_t^1\}$, $\{x_t^2\}$, ..., $\{x_t^n\}$, $\{y_t\}$, one can consider the time index t as labelling samples and use machine learning techniques to try to predict y_t from $(x_t^1, x_t^2, ..., x_t^n)$.

Some of the features x_t^i can be lagged time series, representing values at $t' < t$.

The response y_t can be the future value of a quantity of interest to predict. Every time we try to make a sequence of forecasts, we have a problem of this form.

y_t can also simply be a quantity unobservable at the present time.

Machine learning with time series data

There are many pitfalls!

They are mainly due to the fact that time-ordered samples are generally not independent, but display autocorrelations in the time direction. This is a problem because:

1. It contradicts the usual assumptions made when studying machine learning models. For instance, the least square estimate of a linear regression may be biased when the samples are correlated.
2. It makes cross-validation rather subtle. We cannot randomly assign samples to the training and validation set: correlations make information in the validation set leak into the training set, leading to performance overestimation.
3. ...

Machine learning with time series data

We will not deal with any of the subtleties above in the present talk.

Rather we will try to address an even more fundamental problem: the fact that the relationship between the features and the response may not be constant in time.

This means that the parameters of the model are functions of time.

How can we estimate these functions efficiently with a limited amount of data?

We will answer this question for the simplest machine learning model there is: linear regression.

Outline

FACTOR MODELS

We will consider the problem of regressing a time series $\{y_t\}$ against another (possibly vectorial) time series $\{x_t\}$.

$$y_t = \beta x_t + \epsilon_t$$

where $\{\epsilon_t\}$ is the realization of a white noise and β is a suitable vector of coefficients.

Such linear regression models are often called *factor models*.

- The predictors $\{x_t\}$ are the *factors*.
- The regression coefficients β are called the *factor loadings*.

FACTOR MODELS

In order to avoid the problems mentioned previously arising from the correlation of the samples, we will assume that $\{x_t\}$ is a white noise realization.

This is typically the case when the time series are the logarithmic returns of stocks. Relations between such times series is what makes statistical arbitrage strategies possible.

We will now explain various techniques to estimate the coefficients of the factor model, as functions of time.

Outline

ROLLING REGRESSION

In our setup, the regression coefficients may depend on t:

$$y_t = \beta_t x_t + \epsilon_t$$

The mere fact that the samples come in a time-ordered way means that the model needs to be re-estimated regularly, making automatically the estimated coefficients $\hat{\beta}_t$ functions of time.

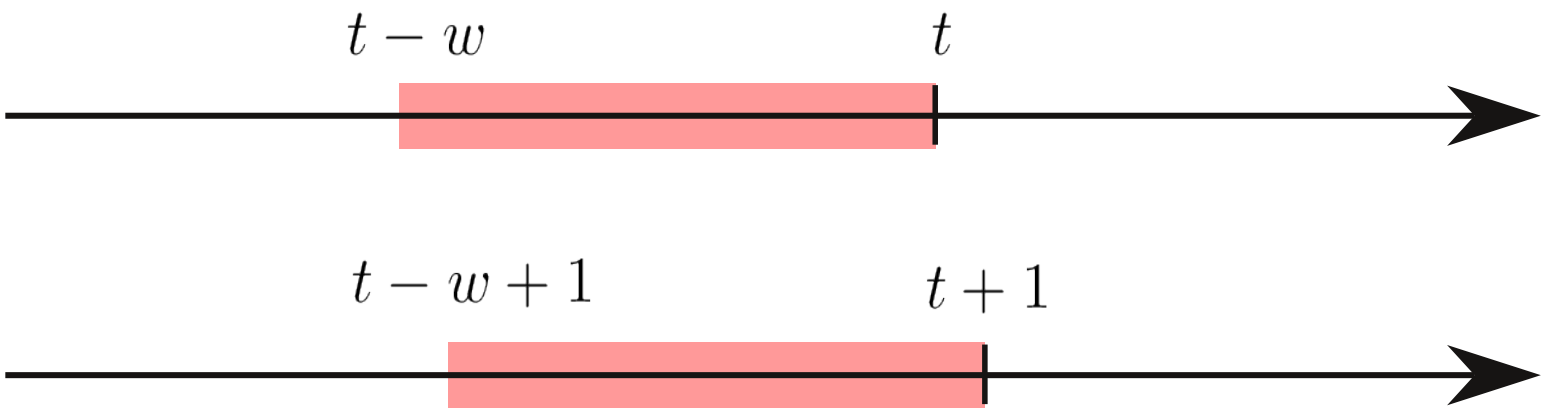
$\beta_t = \beta = \text{cst}$, our estimate $\hat{\beta}_t$ will converge toward the true value β as we collect more samples.

But β_t it is not constant, the samples at $t' \ll t$ are no longer relevant to estimate β_t.

ROLLING REGRESSION

Rolling regression solves the problem by specifying a rolling window of samples used to estimate β_t at time t.

Let w be a positive integer. $\hat{\beta}_t$ is obtained by performing an ordinary least square (OLS) regression using the samples between $t - w$ and t, and ignoring the other samples.



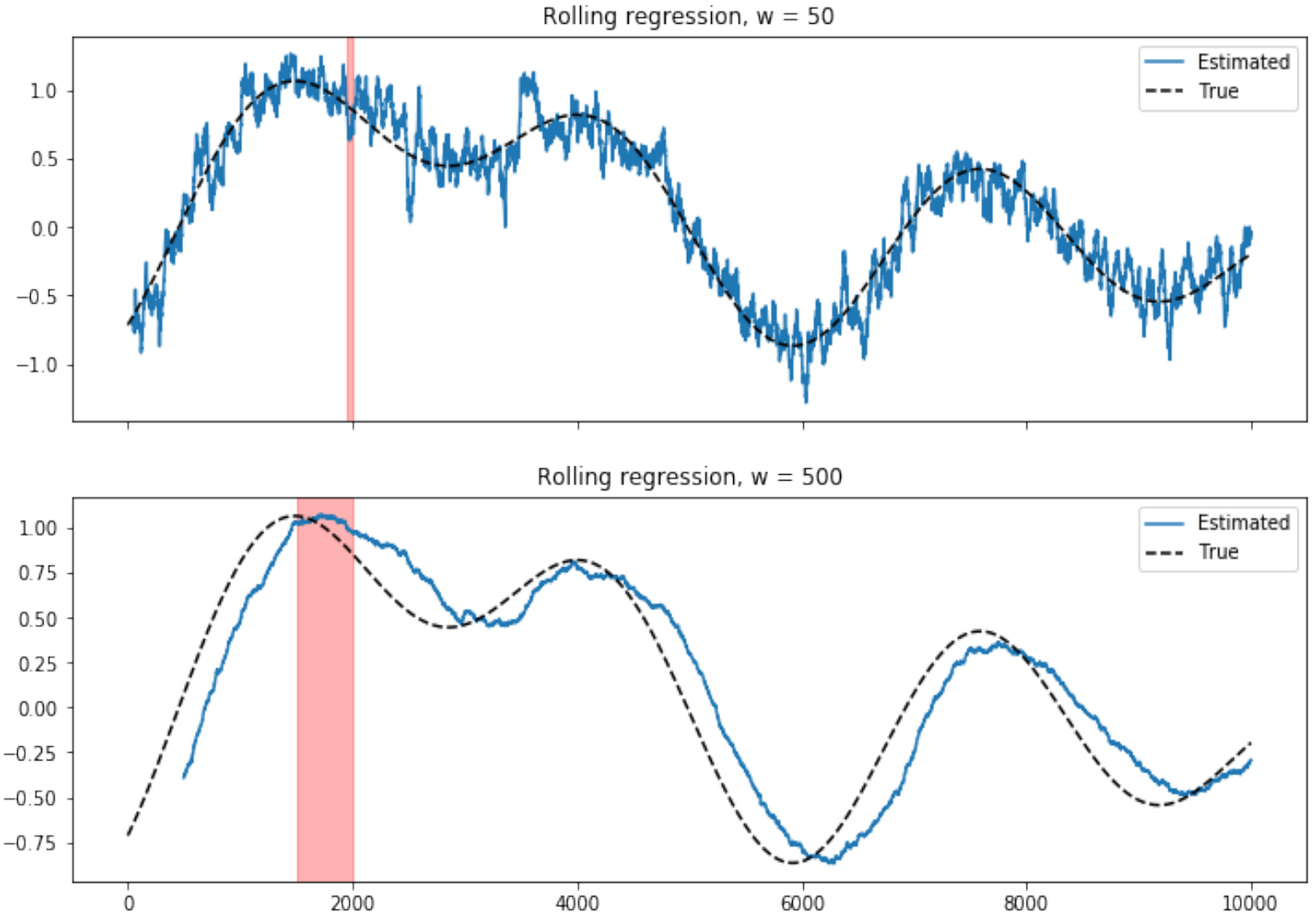
ROLLING REGRESSION

The size of the window w should be chosen depending on the speed at which the coefficients are expected to change, and on the variance of the error term.

This is a bias-variance tradeoff:

- A long window w ensures a small variance but can suffer from a large bias if the coefficients changed significantly during the time scale set by w.
- A short window ensures that only the most recent and relevant samples are used, but due to the small sample size, the variance of the estimate may be large.

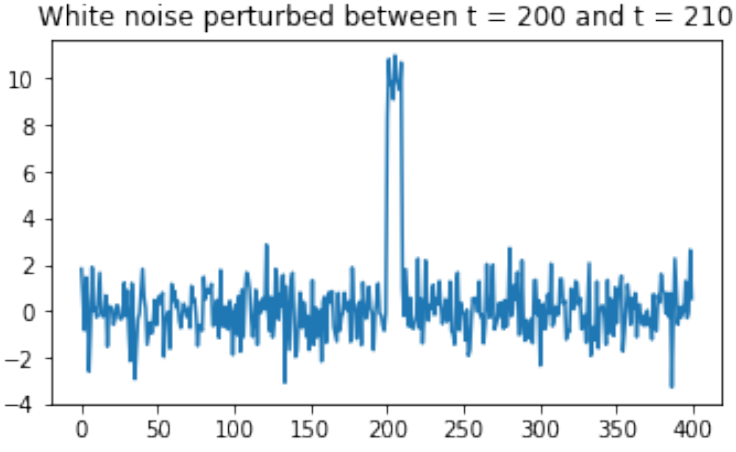
ROLLING REGRESSION

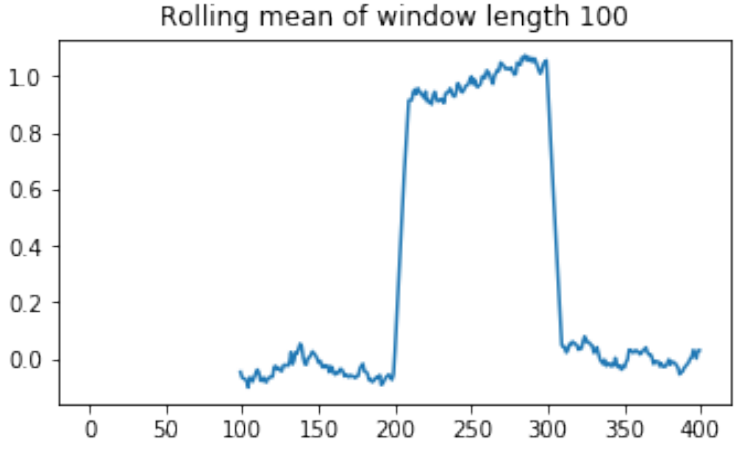


ROLLING REGRESSION

Rolling regression has a number of problems:

1. All the samples have equal weight. If we expect the most recent samples to be more relevant, they should have more weight.
2. The end of the window is arbitrary: at time t, the sample (x_{t-w}, y_{t-w}) carries $1/(w+1)$ of the total weight and at time $t+1$, its weight suddenly drops to zero. This means that the variance around time $t-w$ has an undue influence on the variance of the estimate around time t.

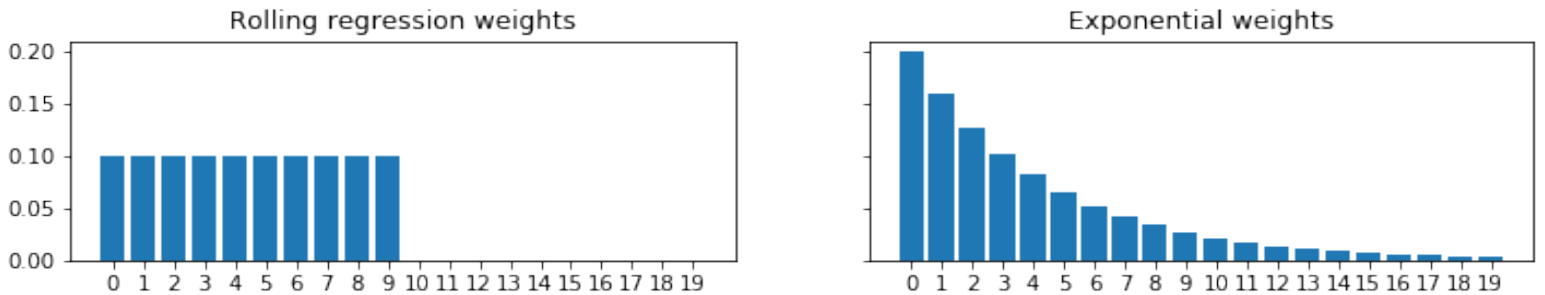




EXPONENTIALLY WEIGHTED MOVING REGRESSION

The two problems above are solved by exponentially weighted moving regression.

Instead of having a fixed window within which the samples are equally weighted and outside which they have weight zero, we will make the sample weight decay with time, so that the most recent samples have the highest weight.



Using an exponential decay is convenient, because it yield an *online* algorithm: we do not need to remember all the samples at previous times. We only remember quantities computed at the previous time.

EXPONENTIALLY WEIGHTED MOVING REGRESSION

Recall that the OLS estimate for β in $y = \beta x + \epsilon$ reads:

$$\hat{\beta} = \frac{\sum_i x_i y_i}{\sum_i x_i x_i}$$

The exponentially weighted regression works as follows:

1. Choose $0 < \lambda < 1$.
2. At time 0, let $S_{xy,0} = S_{xx,0} = 0$.
3. At time t, set
   - $S_{xy,t} = (1-\lambda) S_{xy,t-1} + \lambda x_t y_t$
   - $S_{xx,t} = (1-\lambda) S_{xx,t-1} + \lambda x_t x_t$
4. Estimate $\hat{\beta}_t = S_{xy,t} / S_{xx,t}$.

This is a weighted regression in which the sample x_{t-w} has weight $\sim (1-\lambda)^w$ in the estimate of $\hat{\beta}_t$

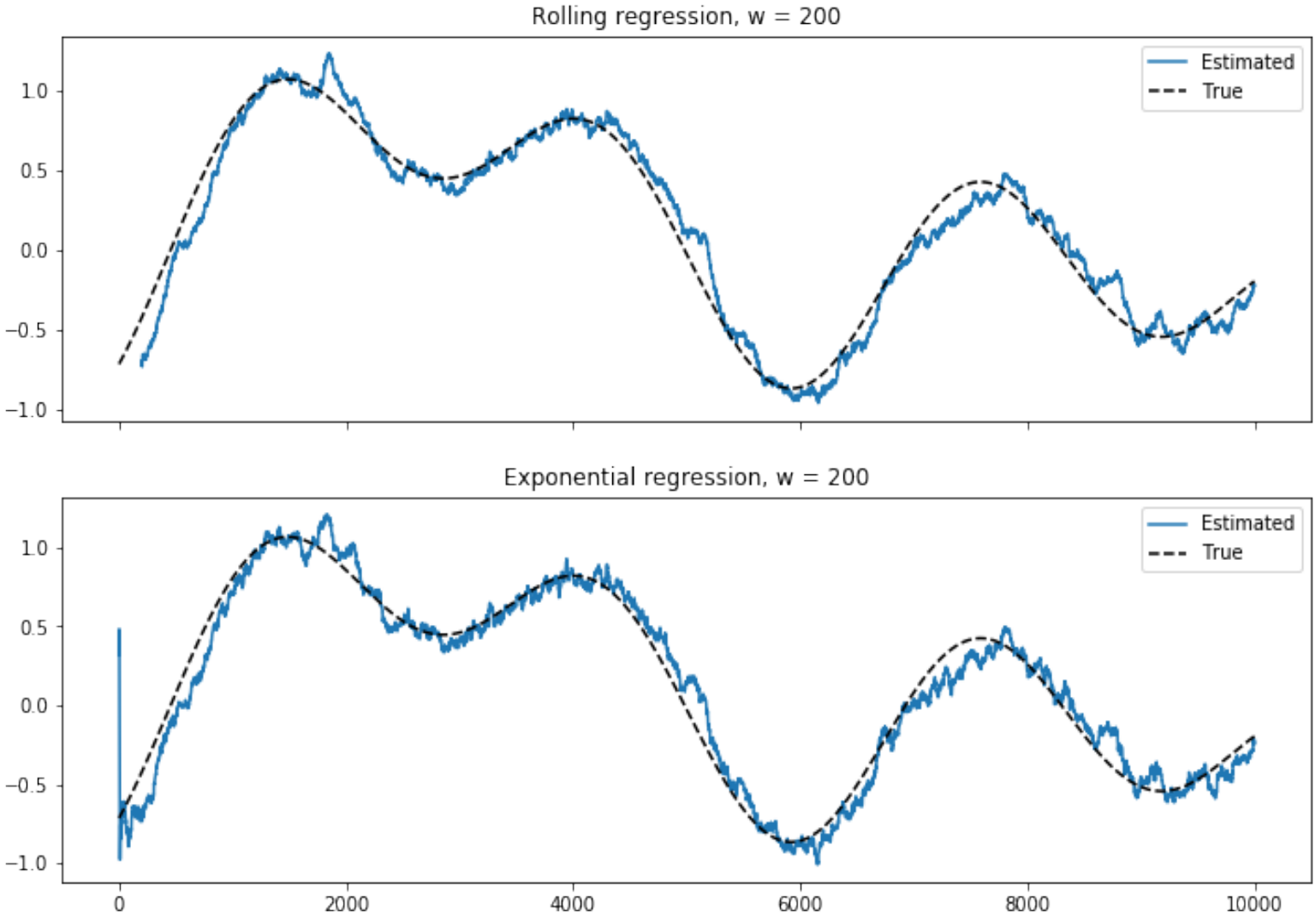
EXPONENTIALLY WEIGHTED MOVING REGRESSION

The parameter λ plays the same role as the window size. A large λ means a fast decay of the weights, analogous to a short window size, and conversely.

Equating the center of mass of the weights in the two cases, we get the relation:

$$\lambda = \frac{2}{w}$$

EXPONENTIALLY WEIGHTED MOVING REGRESSION



Outline

STOCHASTIC REGRESSION

The exponentially weighted regression has a heuristic motivation (recent observations are more relevant than old ones), but no real formal justification.

We can be more rigorous once we specify a model for the time evolution of the coefficients β_t.

As we do not know how the true sensitives evolve in time, we will choose a stochastic model. A simple choice is to assume that β_t follows a random walk:

$$\beta_{t+1} = \beta_t + \delta_t \ ,$$

where δ_t is a white noise independent of $\{\epsilon_t\}$.

We want then to use our observations (x_t, y_t) to extract information about the random walk $\{\beta_t\}$.

HIDDEN MARKOV STATE MODELS

We are now dealing with a *hidden Markov model*. A *hidden state model* is any statistical model that contains random variables (the "state"), whose realizations are not contained in the sample (i.e. they are "hidden").

In a *hidden Markov model*, the hidden variable evolutions in time form a Markov chain, i.e. their distribution at time $t + 1$ conditioned to their value at time t is independent of any previous value.

β_t is a (continuous) hidden state, and the observations

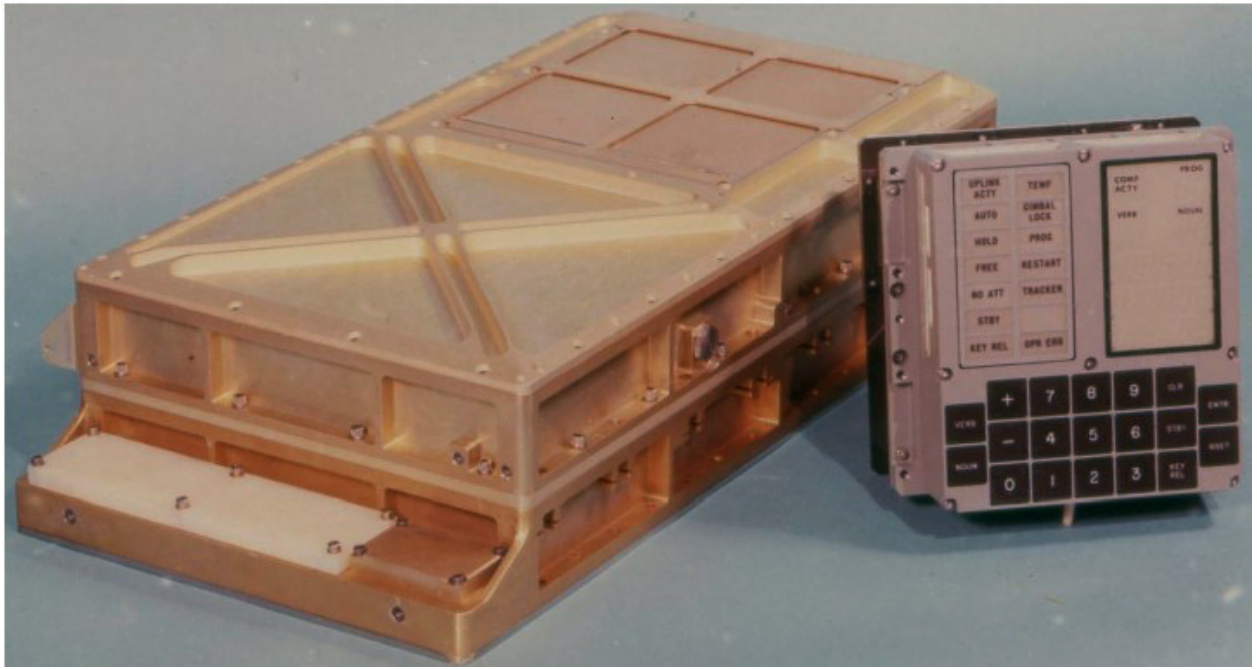
$$y_t = x_t \beta_t + \epsilon_t$$

depend linearly on this state.

The Kalman filter

There is a way to estimate linear Gaussian continuous hidden Markov models, known as the *Kalman filter*.

One of the first application of the Kalman filter was trajectory estimation in the Apollo navigation computer (4kB RAM, 72kB ROM).



The Kalman filter

$$\begin{aligned} &\text{Evolution of the state:} && s_{t+1} = E_t s_t + \epsilon_t^{\mathrm{e}} \\ &\text{Observation of the state:} && z_t = O_t s_t + \epsilon_t^{\mathrm{o}} \end{aligned}$$

- s_t is the state at time t.
- z_t is some observed quantity depending on the state.
- E_t is the evolution matrix.
- O_t is the observation matrix.
- The evolution and observation have stochastic independent Gaussian white noise terms ϵ_t^{e} and ϵ_t^{o}, with zero means and covariances V^{e} and V^{o}.

$\hat{X}_{|t}$ will denote our estimate of the random variable X based on our knowledge at time t.

The Kalman filter

$$\begin{aligned}
&\text{Evolution of the state:} && s_{t+1} = E_t s_t + \epsilon_t^{\mathrm{e}} \\
&\text{Observation of the state:} && z_t = O_t s_t + \epsilon_t^{\mathrm{o}}
\end{aligned}$$

$\hat{V}_t^{\mathrm{s}}$ will denote the estimated variance of our state estimate at time t.

Prediction: Based on our knowledge $\mathcal{I}_{t-1}$ of the system at $t-1$, we can make a prediction $\hat{s}_{t|t-1}$:

$$\begin{aligned}
\hat{s}_{t|t-1} &= \mathbb{E}(s_t|\mathcal{I}_{t-1}) = E_{t-1}\hat{s}_{t-1|t-1} \\
\hat{V}^{\mathrm{s}}_{t|t-1} &= \mathrm{Var}(s_t|\mathcal{I}_{t-1}) = E_{t-1}\hat{V}^{\mathrm{s}}_{t-1|t-1}E_{t-1}^T + V^{\mathrm{e}} \\
\hat{z}_{t|t-1} &= O_t\hat{s}_{t|t-1}
\end{aligned}$$

The Kalman filter

$$\begin{aligned}
&\text{Evolution of the state:} && s_{t+1} = E_t s_t + \epsilon_t^{\mathrm{e}} \\
&\text{Observation of the state:} && z_t = O_t s_t + \epsilon_t^{\mathrm{o}}
\end{aligned}$$

Observation: We now learn the measurement z_t at time t. The residual of our prediction and its covariance are

$$\begin{aligned}
r_t &= z_t - O_t \hat{s}_{t|t-1} \\
\hat{V}_t^{\mathrm{r}} &= O_{t-1} \hat{V}_{t|t-1}^{\mathrm{s}} O_{t-1}^T + V^{\mathrm{o}}
\end{aligned}$$

Update: We update our estimate of the state linearly:

$$\begin{aligned}
\hat{s}_{t|t} &= \hat{s}_{t|t-1} + K_t r_t = (1 - K_t O_t)\hat{s}_{t|t-1} + K_t z_t \\
\hat{V}_{t|t}^{\mathrm{s}} &= (1 - K_t O_t)\hat{V}_{t|t-1}^{\mathrm{s}}(1 - K_t O_t)^T + K_t V^{\mathrm{o}} K_t^T
\end{aligned}$$

THE KALMAN FILTER

$$\hat{s}_{t|t} = \hat{s}_{t|t-1} + K_t r_t = (1 - K_t O_t)\hat{s}_{t|t-1} + K_t z_t$$
$$\hat{V}^{\mathrm{s}}_{t|t} = (1 - K_t O_t)\hat{V}^{\mathrm{s}}_{t|t-1}(1 - K_t O_t)^T + K_t V^{\mathrm{o}} K_t^T$$

The matrix K_t is called the *gain*. The *Kalman gain* K_t^{K} is the optimal choice of gain, namely the one that minimizes the expected mean square error between s_t and $\hat{s}_{t|t}$:

$$K_t^{\mathrm{K}} = \mathrm{argmin}_{K_t} \mathbb{E}\left((s_t - \hat{s}_{t|t})^2 | \mathcal{I}_t\right) = \mathrm{argmin}_{K_t} \mathrm{Tr} \hat{V}^{\mathrm{s}}_{t|t}$$
$$= \hat{V}^{\mathrm{s}}_{t|t-1} O_t^T (\hat{V}^{\mathrm{r}}_t)^{-1} .$$

This tells us exactly how to estimate the state s_t given the observations z_t.

KALMAN FILTER FOR STOCHASTIC REGRESSION

Recall our stochastic regression model:

$$\beta_{t+1} = \beta_t + \delta_t$$
$$y_t = x_t \beta_t + \epsilon_t$$

It is clear how to identify the parameters of the Kalman filter that will estimate the coefficients β:

$$s_t = \beta_t \ , \quad E_t = \mathrm{Id} \ , \quad \epsilon_t^{\mathrm{e}} = \delta_t \ ,$$

$$z_t = y_t \ , \quad O_t = x_t \ , \quad \epsilon_t^{\mathrm{o}} = \epsilon_t$$

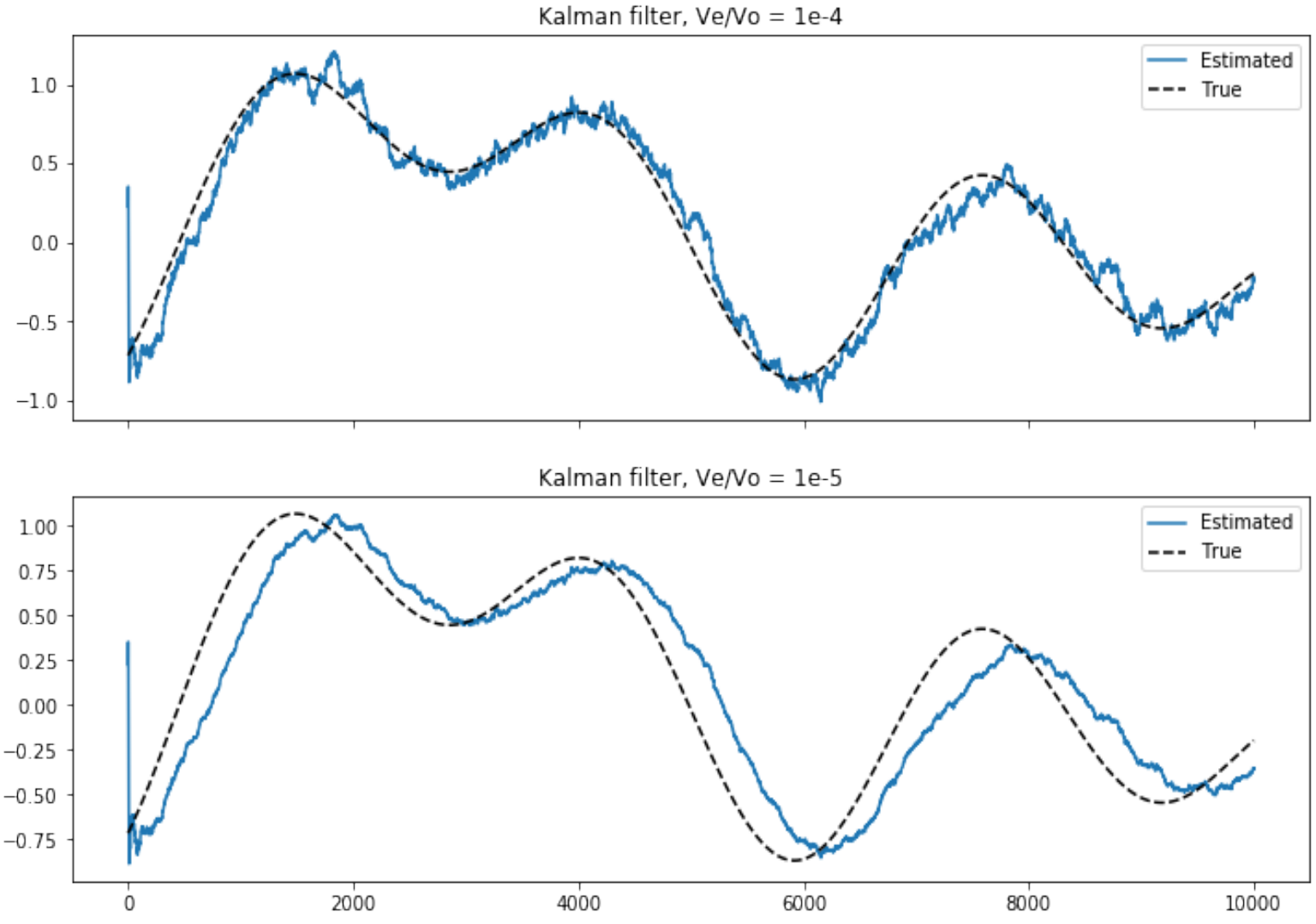
The problem is well-defined once we pick estimates for the covariances of δ_t and ϵ_t, yielding V^{e} and V^{o}.

KALMAN FILTER FOR STOCHASTIC REGRESSION

In fact, the ratio $V^{\mathrm{e}}/V^{\mathrm{o}}$ allows us to tune the bias/variance trade-off, just like the window parameters in the rolling regression:

- The larger V^{e}, the faster the coefficients are expected to change and the faster we have to change our estimate at the expense of increasing the variance of the estimate.
- The larger V^{o} is, the less we should trust any measurement to update our estimate of the coefficients.

KALMAN FILTER FOR STOCHASTIC REGRESSION



THE LOCAL TREND MODEL

Nothing prevents us from creating more fancy models.

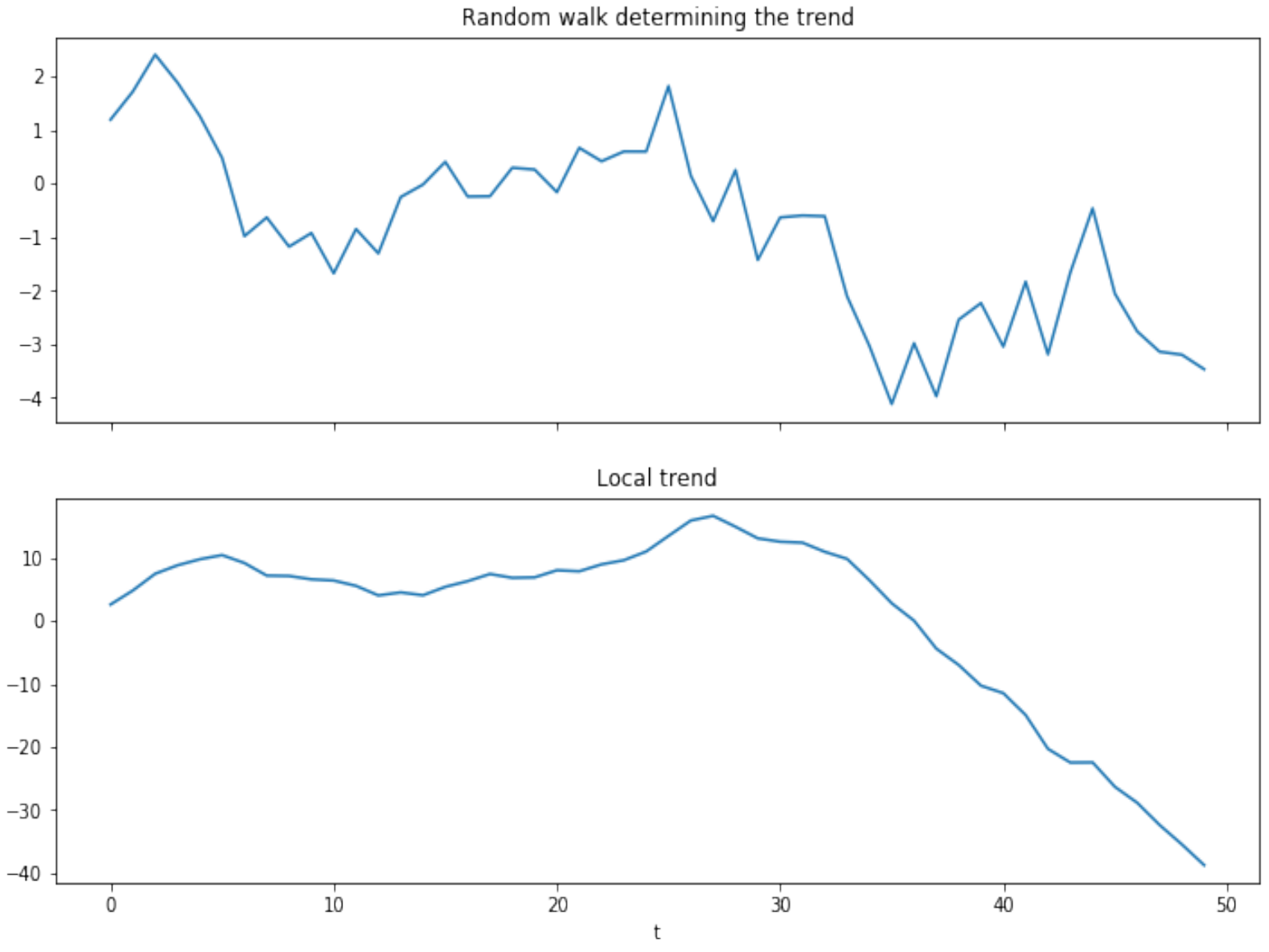
For instance, if the coefficients are smooth functions, a random walk is may not be the best model: if a coefficient has been increasing over the past periods, it is likely that it will continue to do so. A *local trend model* may be more appropriate:

$$\beta_{t+1} = \beta_t + \tau_t + \delta_t$$

$$\tau_{t+1} = \tau_t + \gamma_t$$

where γ_t and δ_t are as before independent Gaussian white noises. τ is a "trend", taking the form of a random walk. The variation of the coefficient β now also has a component given by the random walk τ_t. If $\tau_t > 0$, it is likely that $\tau_{t+1} > 0$ as well, so the variations of β tends to have the same sign, making β closer to a smooth function.

THE LOCAL TREND MODEL



KALMAN FILTER FOR THE LOCAL TREND MODEL

Recall the local trend model:

$$\begin{aligned}\beta_{t+1} &= \beta_t + \tau_t + \delta_t \\ \tau_{t+1} &= \tau_t + \gamma_t \\ y_t &= x_t \beta_t + \epsilon_t\end{aligned}$$

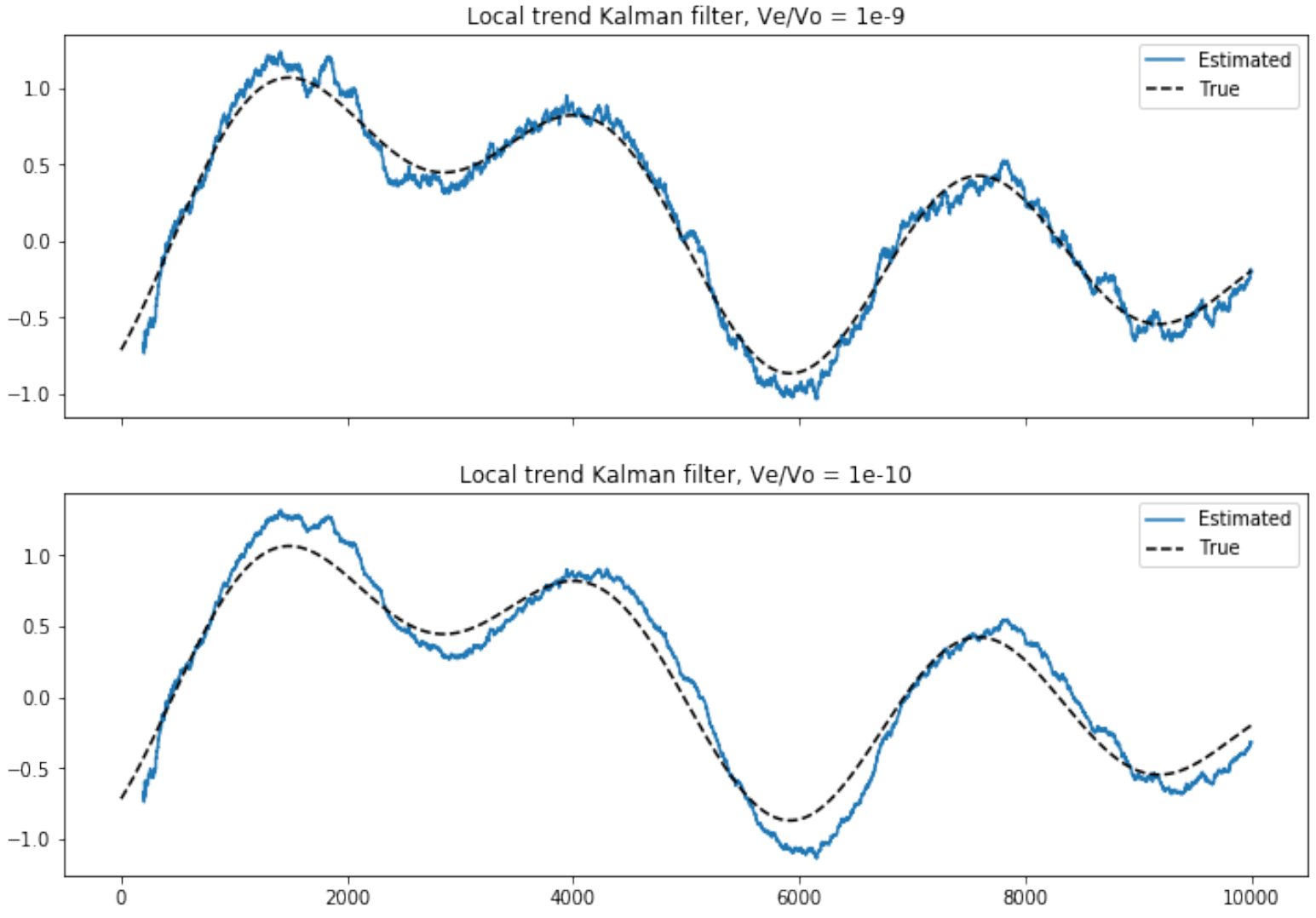
We can again reformulate it as a Kalman filter:

$$s_t = \begin{pmatrix} \beta_t \\ \tau_t \end{pmatrix}, \quad E_t = \begin{pmatrix} \mathrm{Id} & \mathrm{Id} \\ 0 & \mathrm{Id} \end{pmatrix}, \quad \epsilon_t^{\mathrm{e}} = \begin{pmatrix} \delta_t \\ \gamma_t \end{pmatrix},$$

$$z_t = y_t, \quad O_t = \begin{pmatrix} x_t & 0 \end{pmatrix}, \quad \epsilon_t^{\mathrm{o}} = \epsilon_t$$

The bias-variance tradeoff is again controlled by $V^{\mathrm{e}}/V^{\mathrm{o}}$.

KALMAN FILTER FOR THE LOCAL TREND MODEL



Outline

EXPERIMENT

It is hard to assess the models reviewed above visually. We need a concrete setup to compare them.

We will consider 3-factor models:

$$x_t = \begin{pmatrix} x_t^0 \\ x_t^1 \\ x_t^2 \end{pmatrix}, \quad \beta_t = \begin{pmatrix} \beta_t^0 & \beta_t^1 & \beta_t^2 \end{pmatrix},$$

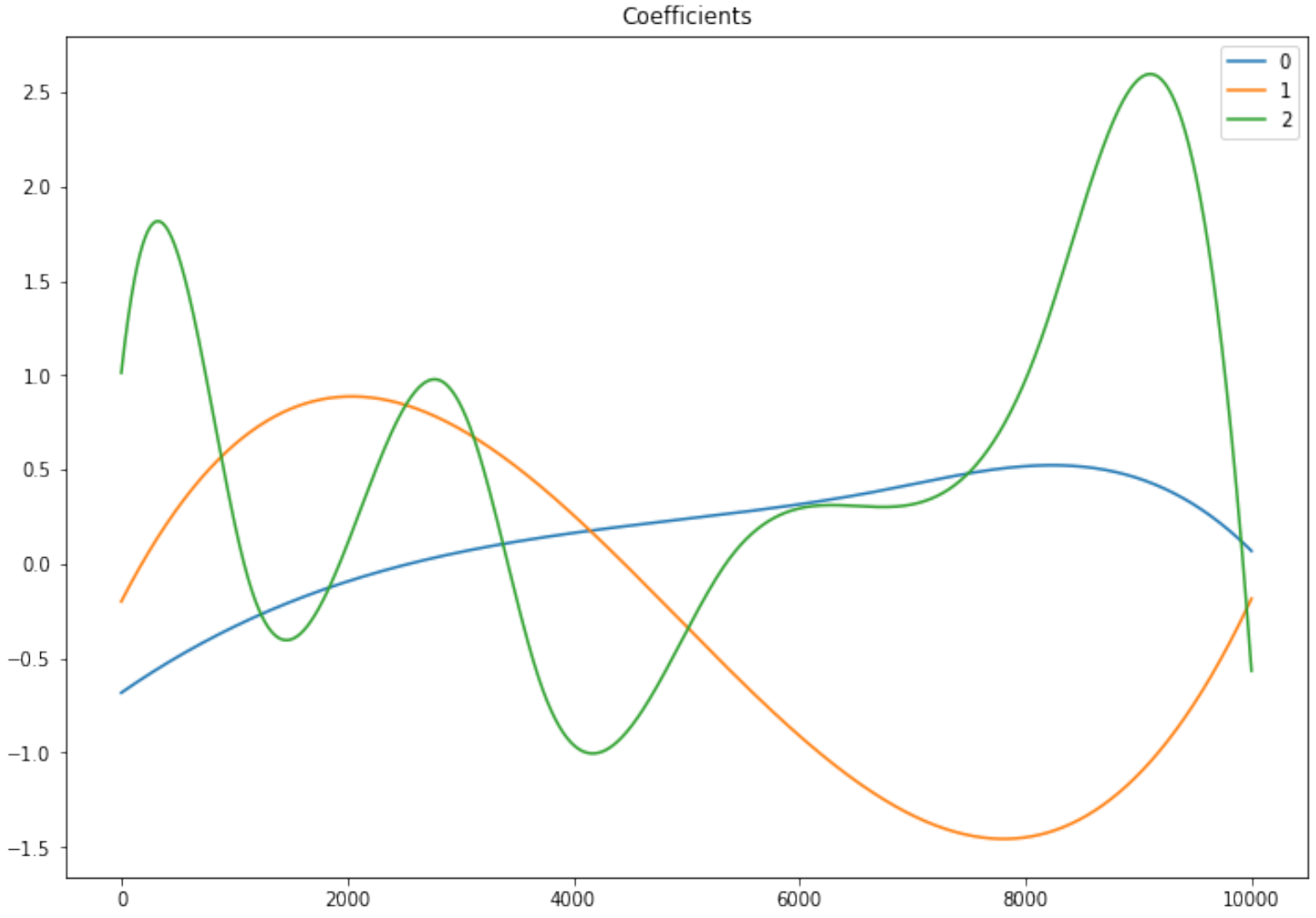
$$y_t = \beta_t x_t + \epsilon_t$$

where x_t^i, ϵ_t are independent Gaussian white noises.

We construct random smoothly varying coefficients β_t^i using splines.

We generate 10'000 datasets $(\beta_t, x_t, y_t, \epsilon_t)$ of 10'000 timesteps each.

EXPERIMENT


Coefficients
0
1
2
2.5
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
0
2000
4000
6000
8000
10000

EXPERIMENT

We evaluate the performance of the models as follows:

- We assume that at time t, the factors x_t are known, as well as $(y_s, x_s)_{s<t}$.
- We use our models to get an estimate $\hat{\beta}_t$, and then predict $\hat{y}_t = \hat{\beta}_t x_t$.
- We evaluate the performance of the model using the mean square error and the weighted accuracy

$$\frac{1}{T}\sum_t |y_t| * \Delta(\text{sgn}(y_t) = \text{sgn}(\hat{y}_t)) \ ,$$

where $\Delta(\text{True}) = 1$, $\Delta(\text{False}) = -1$.

EXPERIMENT

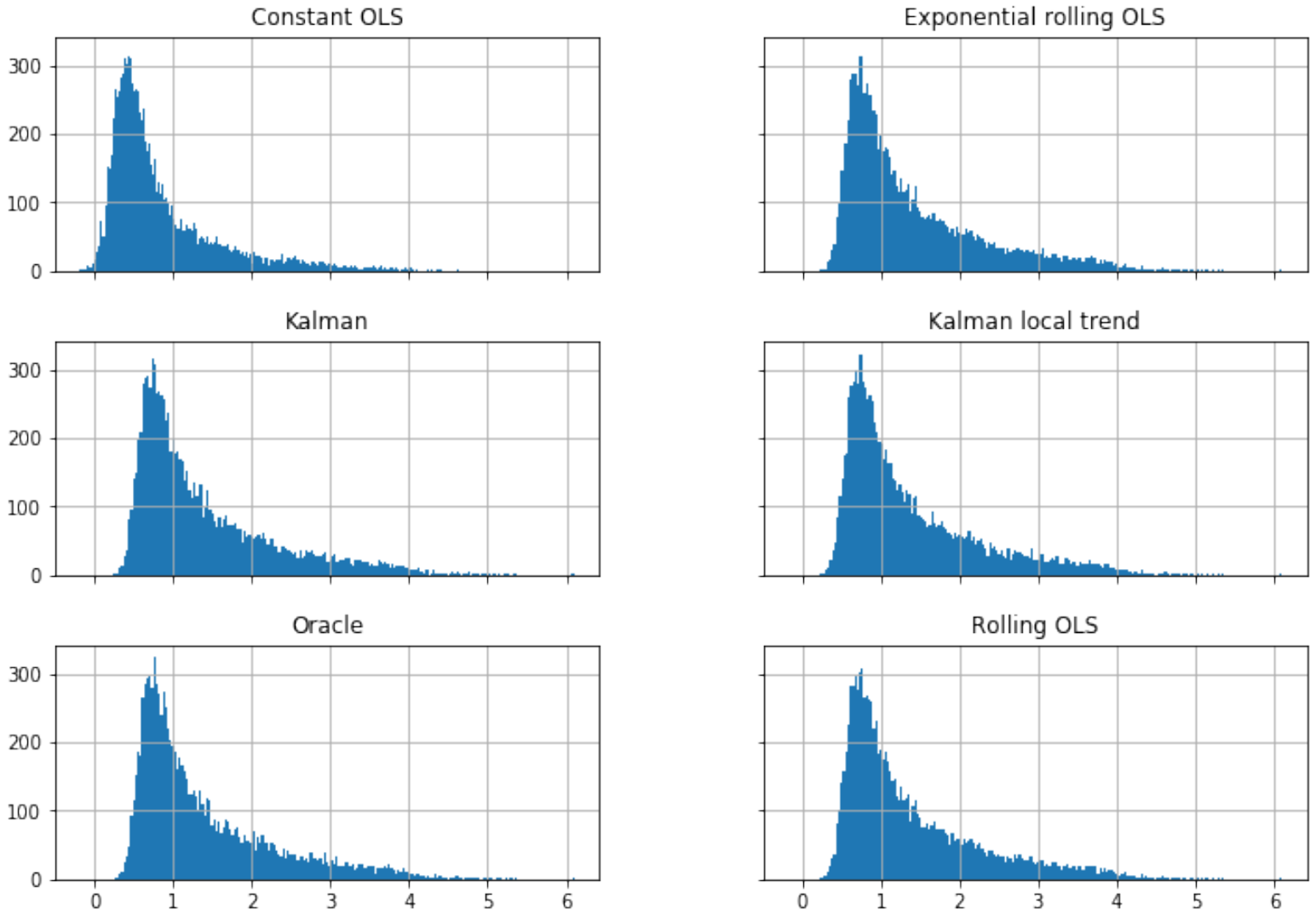
We compute the weighted accuracy and the mean square error for each of the following models on each of the datasets:

1. Ordinary least square regression. (Note that this model suffers from look-ahead bias.)
2. Rolling regression
3. Exponentially weighted rolling regression.
4. Kalman filter
5. Local trend Kalman filter
6. An "oracle" model that knows the true coefficients β_t.

We select the model parameters on an independent set of datasets. We do not try to tune them too finely by considering only values in $\{..., 0.1, 0.2, 0.5, 1, 2, 5, 10, 20, 50, ...\}$.

RESULTS

Weighted accuracy



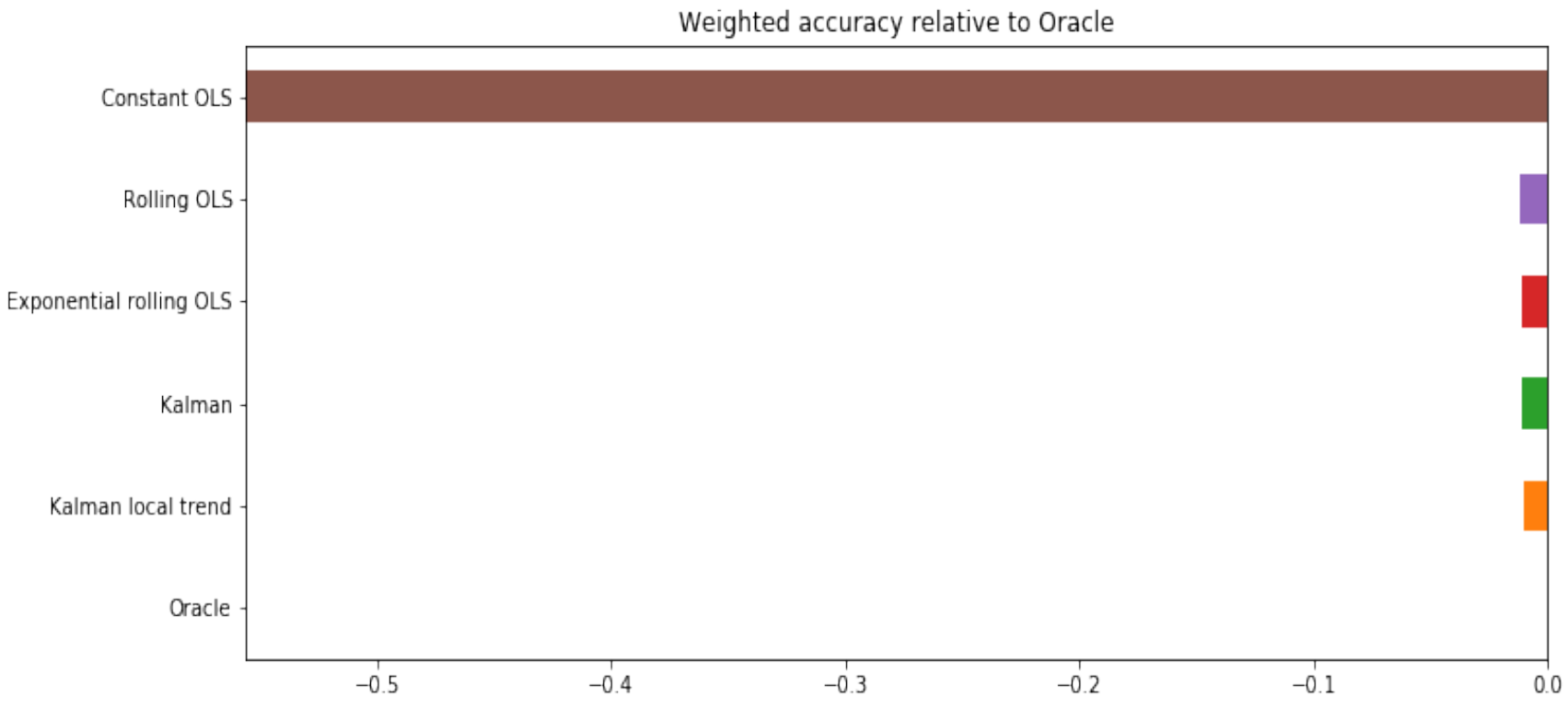
RESULTS

There is a lot of variability in accuracy.

However, as the accuracies of the models are highly correlated, we perform dependent t-tests and (non-parametric) Wilcoxon signed rank tests to test the differences in accuracies between models at successive ranks. All the differences are statistically significant:

Model	W. accuracy	t-test	Wilcoxon
Oracle	1.3984	0.00e+00	0.000000e+00
Kalman local trend	1.3885	6.02e-144	1.17e-128
Kalman	1.3873	1.04e-03	2.73e-73
Exponential OLS	1.3871	1.09e-41	8.48e-29
Rolling OLS	1.3866	0.00e+00	0.00e+00
Constant OLS	0.8423	-	-

Results

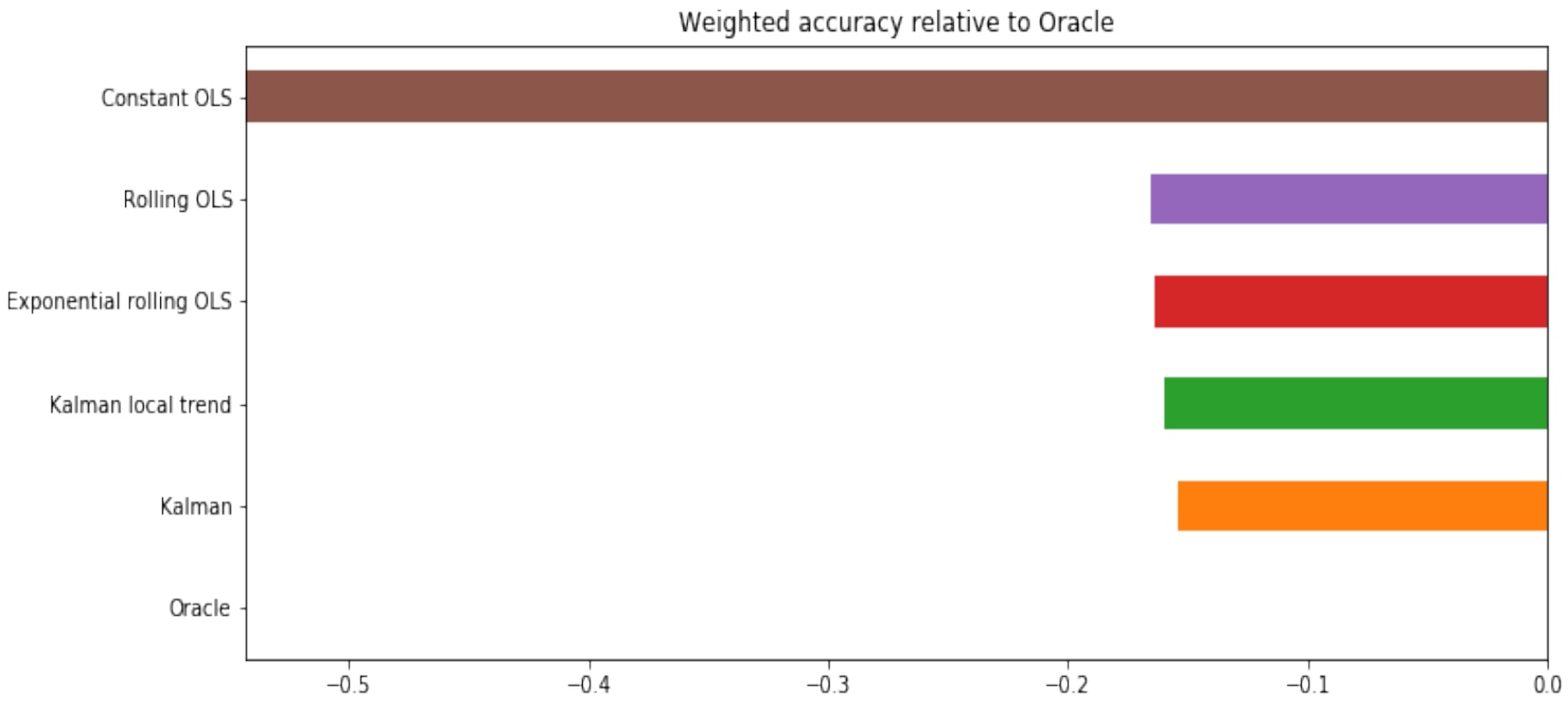

Weighted accuracy relative to Oracle
Constant OLS
Rolling OLS
Exponential rolling OLS
Kalman
Kalman local trend
Oracle
−0.5
−0.4
−0.3
−0.2
−0.1
0.0

Results

We can repeat the same experiment in a very noisy setting where the variance of the error term in the regression is 10 times the variance of the factors.

Model	W. accuracy	t-test	Wilcoxon
Oracle	1.4036	0.00e+00	0.00e+00
Kalman	1.2499	1.09e-18	4.15e-06
Kalman local trend	1.2438	2.06e-11	2.70e-11
Exponential OLS	1.2397	4.37e-02	2.99e-05
Rolling OLS	1.2382	0.00e+00	0.00e+00
Constant OLS	0.8607	NaN	NaN

Results

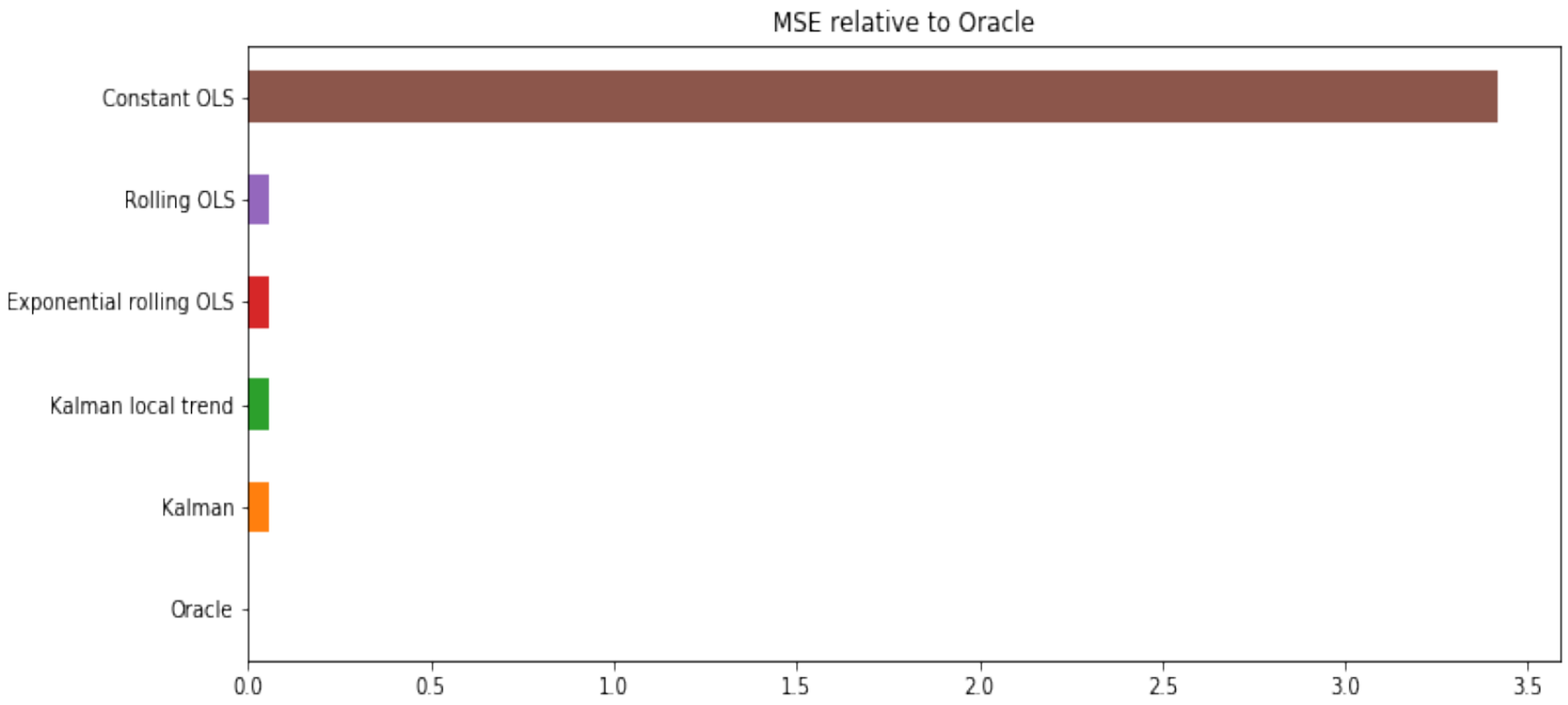

Weighted accuracy relative to Oracle
Constant OLS
Rolling OLS
Exponential rolling OLS
Kalman local trend
Kalman
Oracle
−0.5
−0.4
−0.3
−0.2
−0.1
0.0

Results

We can also evaluate the algorithms using simply the mean square error. Interestingly, the hyperparameters optimal for the weighted accuracy and for the MSE are different for the Kalman filters.

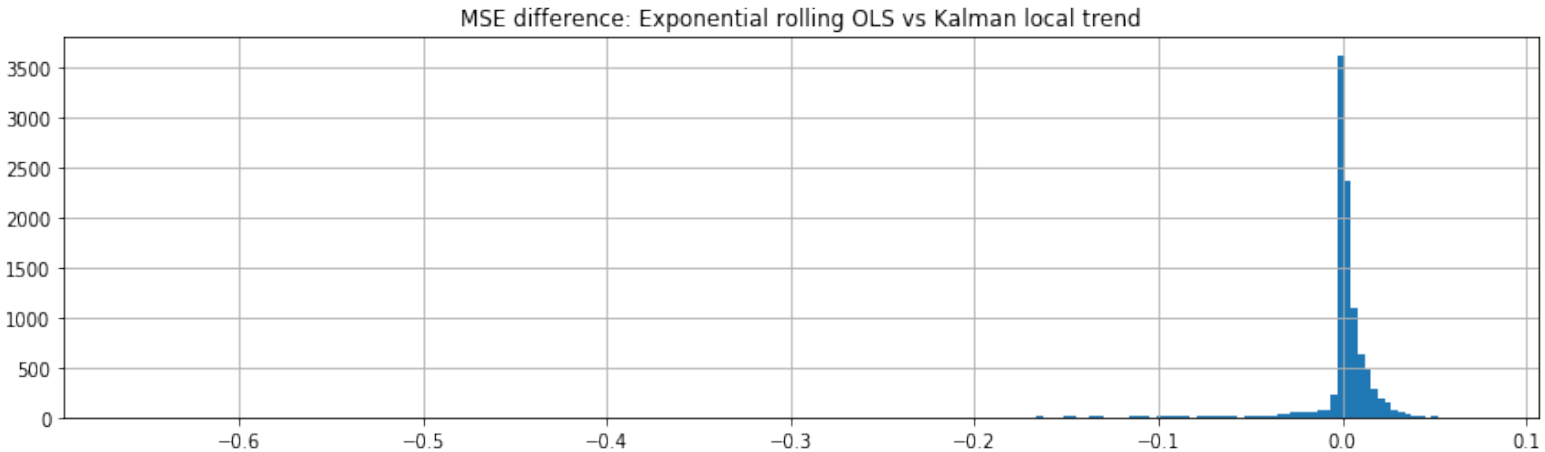
	MSE	t-test	Wilcoxon
Oracle	1.0006	0.00e+00	0.0
Kalman	1.0551	4.54e-01	0.0
Kalman local trend	1.0554	8.45e-01	0.0
Exponential OLS	1.0554	1.57e-126	0.0
Rolling OLS	1.0594	0.00e+00	0.0
Constant OLS	4.4189	NaN	NaN

Results


MSE relative to Oracle
Constant OLS
Rolling OLS
Exponential rolling OLS
Kalman local trend
Kalman
Oracle
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5

RESULTS

When looking at the distribution of the differences does not look very Gaussian and one may think the Wilcoxon p-value should be trusted more than the t-test p-value.



Results

But redoing the experiment confirms that the difference between the local trend model and the exponentially weighted regression is not significant.

It is not clear why the Wilcoxon test gives such a small p-value.

	MSE	t-test	Wilcoxon
Oracle	1.0007	0.00e+00	0.0
Kalman	1.0565	2.49e-11	0.0
Exponential OLS	1.0570	6.42e-02	0.0
Kalman local trend	1.0576	5.77e-95	0.0
Rolling OLS	1.0613	0.00e+00	0.0
Constant OLS	4.5279	NaN	NaN

Conclusions

1. There are better way of estimating time-dependent model parameters than a rolling window.
2. The Kalman filter lets us specify rather elaborate models for the time-dependent parameters and provides a sound method to estimate them.
3. In our factor model experiment, Kalman filters generally overperform other methods. The overperformance is statistically significant, but rather minimal.

To experiment for yourself, see:
https://github.com/sam31415/time-dependent-estimation