The Kalman Filter and Hidden Markov Models

tracking what you cannot see, in markets and elsewhere

Martin Burke (with Claude)

Draft

Abstract

A price on a screen is not the thing you want to trade on. It is a noisy, fleeting shadow of something you cannot see directly: a fair value, a hedge ratio, a market mood. This note is about a single, powerful idea for recovering that hidden thing as it changes—the *state-space model*, and the *filtering* recursions that estimate its hidden state one observation at a time. Two filters dominate practice and differ only in the *shape* they assume for the hidden state. The **Kalman filter** takes the state to be a continuous number (or a short vector of numbers) that drifts smoothly—a fair value, a trend, a regression coefficient—and, under linear-Gaussian assumptions, tracks it with nothing more than a running mean and variance updated by a beautifully simple rule. The **hidden Markov model** (HMM) takes the state to be a discrete *regime*—calm versus turbulent, trending versus ranging—that switches at random times, and infers from the data which regime you are probably in right now. We build both from the same Bayesian skeleton, derive their update rules from first principles (with the linear algebra quarantined to appendices), and show each at work on a canonical trading problem: the Kalman filter on a drifting pairs-trading hedge ratio, the HMM on volatility regimes. The pairs-trading example is a deliberate sequel: the companion note *A Practical Introduction to Time-Series Analysis for Pairs Trading and Cointegration* [1] builds the trade with a hedge ratio fitted *once* and held fixed; here we let that ratio become a hidden state that *drifts*, and track it in real time. We close with an candid account of where these models help, where they mislead, and how to tell the difference.

Contents

1 From a number you can see to a state you cannot

Look at a price chart and you are looking at the wrong thing—or rather, at the right thing wearing a disguise. The last traded price of an asset is a single number that jumps around for reasons that have little to do with what you care about: a large order clearing, the bid–ask bouncing, a stale quote, a fat finger. Underneath that jitter is usually something slower and more meaningful—a *fair value* that drifts as news and flows accumulate, a *relationship* between two assets that holds on average, a *mood* that tilts the odds toward calm or chaos. You never observe these directly. You observe their noisy shadow (Figure 1), and your job is to infer the thing casting it.

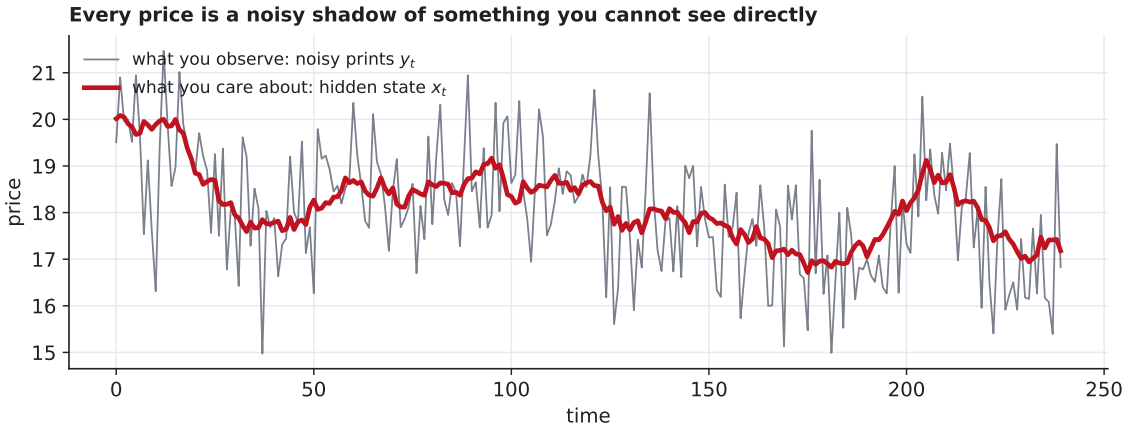

Figure 1: The whole problem in one picture. The grey line is what the market hands you—noisy prints y_t. The red line is what you actually want to act on—a hidden state x_t (here a slowly drifting fair value) that you never see cleanly. Estimating the red line from the grey, *as it arrives, without peeking at the future*, is the task this note is about.

The proper name for this task is *filtering*: separating signal from noise in a stream of measurements, in real time. "Real time" is the load-bearing word. It is easy to draw a smooth curve through a finished chart, because at the end you can see the whole series and average out the wiggles symmetrically. That is hindsight, and it is useless for trading, because to act at time t you may use only what you knew up to time t. A filter is a rule that, at each step, takes your previous best estimate, advances it, and nudges it toward the latest measurement—producing a new estimate that uses every observation so far and not one observation more.

Two questions organise everything that follows. First, *what kind of thing is the hidden state?* If it is a continuous quantity that drifts—a fair value, a slope, a hedge ratio—the natural tool is the **Kalman filter**. If it is a discrete label that switches—which of a few market regimes are we in—the natural tool is the **hidden Markov model**. Second, *how do we update our belief when a new number arrives?* Remarkably, both tools answer this the same way, with the same two-beat rhythm: predict, then correct. Section 2 builds that common skeleton. Part I puts continuous flesh on it (Kalman); Part II puts discrete flesh on it (HMM). A short synthesis section weighs one against the other, and a frank section on limitations keeps us grounded. The appendices hold the Gaussian algebra and the recursions in full, so the main text can stay about the ideas.

Who this is for. You are comfortable with probability, a little linear algebra, and the idea of a random process, but you are not an algorithmic-trading specialist. You do not need stochastic calculus. Where a derivation would pull us off course, it goes to an appendix and the main line carries the intuition and the result.

A follow-on, for those who have read it. This note is self-contained, but it is also a sequel to the companion note *Time-Series Analysis for Pairs Trading and Cointegration* [1]. That note built a market-neutral pairs trade end to end: it established when two prices are genuinely tethered (cointegration, the Augmented Dickey–Fuller test), fitted the hedge ratio that defines their spread, and turned the spread into a z-score signal with a mean-reversion half-life—all with the relationship estimated *once*, on a fixed sample. Its closing section warned that the tether can *break*, and that a live trade really wants the hedge ratio re-estimated as it drifts. This note answers that directly. The Kalman filter (§7) turns the fixed hedge ratio into a drifting hidden state tracked in real time; the hidden Markov model (§10) adds a regime layer that can flag the turbulent conditions in which tethers tend to snap. A reader who has not seen the companion note will follow everything here regardless—Appendix E recaps the one idea, stationarity, that the trading examples lean on.

2 The common skeleton: a hidden state and its noisy shadow

Strip both models down and the same three commitments remain. They are worth stating plainly, because once you accept them, the algorithms are essentially forced.

1. There is a hidden state that evolves on its own. At each time t there is a state x_t you cannot observe. It changes from one step to the next according to some rule—deterministic drift plus randomness. Crucially, that rule is *Markov*: where the state goes next depends only on where it is now, not on the entire path that brought it there. The state is a sufficient summary of the past. Formally, the dynamics are captured by a *transition distribution*

$$p(x_t \mid x_{t-1}),$$

read "given the state now, here is the distribution of the state next step." The Markov property is what makes real-time filtering possible at all: it means your current belief about x_t is all you need to carry forward—you never have to re-examine the full history.

2. Each observation is a noisy function of the current state. You do not see x_t; you see a measurement y_t that depends on x_t and nothing else, blurred by noise. This is the *emission* (or observation) distribution

$$p(y_t \mid x_t).$$

The state casts the shadow; the shadow is all you get. Given the state, successive observations are independent—all the dependence between observations flows *through* the hidden state, never directly.

3. Belief is a distribution, and we update it by Bayes' rule. Because we are uncertain about x_t, our knowledge of it is itself a probability distribution—a *belief*. Filtering is the art of maintaining one number's worth, or one distribution's worth, of belief and revising it as data arrives. These two commitments—a Markov hidden state, noisy emissions—define a *state-space model* (also called a hidden Markov process; Figure 2).

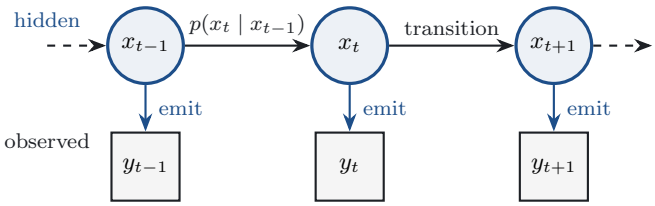

Figure 2: The state-space model, drawn as a graph. The hidden states (blue circles) form a Markov chain: each depends only on the one before. Each state emits an observation (grey square) and nothing else connects the observations to one another. The Kalman filter and the hidden Markov model are *the same diagram*—they differ only in whether the circles hold a continuous number or a discrete label, and in the shapes of the two distributions on the arrows.

2.1 The two-beat rhythm: predict, then update

Write $b_{t-1}(x) = p(x_{t-1} \mid y_{1:t-1})$ for our belief about the state after seeing the first $t-1$ observations. A new observation y_t arrives. How should the belief change? In exactly two moves (Figure 3).

Predict. First, advance the old belief through the dynamics, *before* looking at the new data. Every place the state might have been, it could have moved somewhere new according to $p(x_t \mid x_{t-1})$; averaging over the old belief gives the *prior* for x_t:

$$b_t^-(x_t) \;=\; \int p(x_t \mid x_{t-1})\, b_{t-1}(x_{t-1})\, \mathrm{d}x_{t-1}.$$

This step always makes you *less* certain: pushing a belief through a random transition spreads it out. The minus superscript marks "predicted, pre-measurement."

Update. Now fold in the measurement. By Bayes' rule, the new observation reweights the prior in proportion to how well each candidate state explains it:

$$b_t(x_t) \;=\; \frac{p(y_t \mid x_t)\, b_t^-(x_t)}{\displaystyle\int p(y_t \mid x_t)\, b_t^-(x_t)\, \mathrm{d}x_t}.$$

This step always makes you *more* certain: multiplying by the likelihood concentrates the belief on states consistent with what you just saw. The denominator is just the normalising constant that keeps the belief a valid distribution—and, usefully, it is the probability the model assigned to the observation y_t, a quantity we will reuse for fitting and for scoring how surprised the model is.

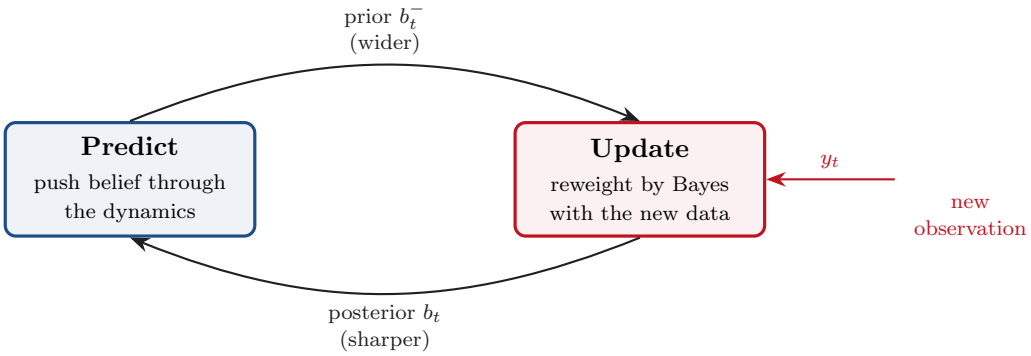

Figure 3: The filtering cycle. Each new observation drives one turn of a two-step loop: *predict* spreads the belief out by advancing it through the dynamics; *update* sharpens it again using Bayes' rule on the data. Belief flows around this loop forever, one lap per observation. The Kalman filter and the HMM are two ways of carrying out exactly these two steps.

That is the entire conceptual content of filtering. Everything specific to the Kalman filter or the HMM is just a choice of how to *represent* the belief b_t and how to *carry out* the two integrals (or, for discrete states, sums) above. The pleasant surprise is that for two important special cases those operations have closed forms and the belief never has to be stored as a general function:

- **Continuous state, linear dynamics, Gaussian noise** $\Rightarrow$ the belief stays Gaussian forever, so it is fully described by a *mean* and a *variance*. The two steps become simple matrix updates. This is the **Kalman filter** (Part I).
- **Finitely many discrete states** $\Rightarrow$ the belief is just a short vector of probabilities, one per state. The integrals become matrix–vector products. This is the **hidden Markov model** (Part II).

Both keep an exact, finite belief and update it in closed form. When neither assumption holds—nonlinear dynamics, non-Gaussian noise, a continuum of states with no linear structure—the belief has no finite summary and one must approximate it, for example by carrying a cloud of sample states (a *particle filter*, §11). But an astonishing amount of practical work lives inside the two tidy cases, and they are where we spend this note.

Part I — The Kalman filter: a continuous, drifting state

3 The linear-Gaussian bet

The Kalman filter makes two assumptions, and earns an enormous payoff for them.

The assumptions. *Linear dynamics, Gaussian noise.* The state is a real number (or a vector of them), it evolves by a linear map plus Gaussian noise, and each observation is a linear function of the state plus Gaussian noise. In the one-dimensional case, with state x_t and observation y_t,

$$\begin{aligned} x_t &= x_{t-1} + w_t, \quad w_t \sim \mathcal{N}(0,\, q) && \text{(state drifts a bit each step)},\\ y_t &= x_t + v_t, \qquad\; v_t \sim \mathcal{N}(0,\, r) && \text{(we measure it, with noise)}. \end{aligned}$$

This particular pair—a state that random-walks, observed through additive noise—is the *local level model*, the "hello world" of filtering. The process variance q says how fast the hidden level moves; the measurement variance r says how noisy each print is. Their ratio is the whole story, as we will see.

The payoff. Gaussians are closed under the two operations the filter needs. Push a Gaussian belief through a linear-Gaussian transition and you get another Gaussian (wider). Multiply a Gaussian prior by a Gaussian likelihood and you get another Gaussian (sharper). So if the belief starts Gaussian, it stays Gaussian for all time—and a Gaussian is pinned down by just two numbers, its mean and its variance. The filter never stores a distribution; it stores a running estimate $\hat{x}_t$ (the mean, our best guess) and an uncertainty P_t (the variance, how sure we are). Two numbers in, two numbers out, every step. Appendix A proves the two closure properties and is the algebraic engine behind everything in this part; Appendix B turns the crank to derive the update equations below.

Why Gaussian-in, Gaussian-out matters. A general belief over a continuous state is a function—infinitely many numbers. Carrying it forward exactly would be hopeless. The linear-Gaussian assumptions collapse that function to two numbers and the two filtering integrals to arithmetic. That collapse, not any single equation, is the reason the Kalman filter is fast enough to run on every tick.

4 The filter in one dimension

Take the local level model and walk through one cycle. Suppose after $t-1$ steps our belief about the level is Gaussian with mean $\hat{x}_{t-1}$ and variance P_{t-1}.

Predict. The level random-walks, so our best guess for it is unchanged but our uncertainty grows by the process variance:

$$\hat{x}_t^- = \hat{x}_{t-1},$$
$$P_t^- = P_{t-1} + q.$$

The prediction is vaguer than the posterior we started from—exactly the "predict spreads you out" effect from §2.1.

Update. Now the measurement y_t arrives. Two pieces of information must be combined: the prediction $\hat{x}_t^-$, with variance P_t^-, and the measurement y_t, with variance r. Combining two Gaussian estimates of the same quantity is a classic move—you weight each by its precision (inverse variance) and renormalise. The result, derived in Appendix B, is cleanest written through the *Kalman gain*

$$K_t = \frac{P_t^-}{P_t^- + r},$$

a number between 0 and 1, after which the update reads

$$\hat{x}_t = \hat{x}_t^- + K_t\left(y_t - \hat{x}_t^-\right),$$
$$P_t = \left(1 - K_t\right) P_t^-.$$

Read the first line slowly, because it is the soul of the filter. The quantity $y_t - \hat{x}_t^-$ is the *innovation*: how much the measurement surprised us relative to what we predicted. The new estimate is the old prediction plus a fraction K_t of that surprise. The filter does not lurch to the data and it does not ignore it; it steps *part of the way*, and the gain K_t sets how far.

The gain is a trust dial. Look at what sets K_t. If the measurement is clean (r small) relative to our predictive uncertainty (P_t^-), then $K_t \to 1$: trust the data, jump almost all the way to it. If the measurement is noisy (r large) or we were already confident (P_t^- small), then $K_t \to 0$: trust the model, barely move. Figure 5 plots this dial; Figure 4 shows one update as the multiplication of two Gaussians, with the posterior landing a fraction K_t of the way from prediction to measurement and *narrower than either*—the reward for combining two independent pieces of evidence.

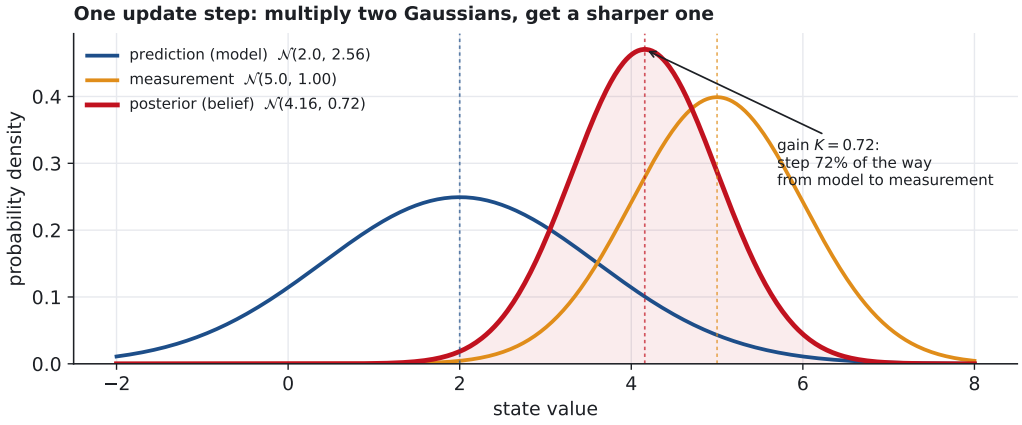

Figure 4: One update step is a product of two Gaussians. The prediction (blue, wide: the model is unsure) meets the measurement (amber, sharp: a clean reading). Their normalised product is the posterior (red), whose mean sits a fraction $K = 0.72$ of the way from prediction to measurement and whose spread is *smaller than either* input. Combining two noisy estimates of the same quantity always beats trusting either alone.

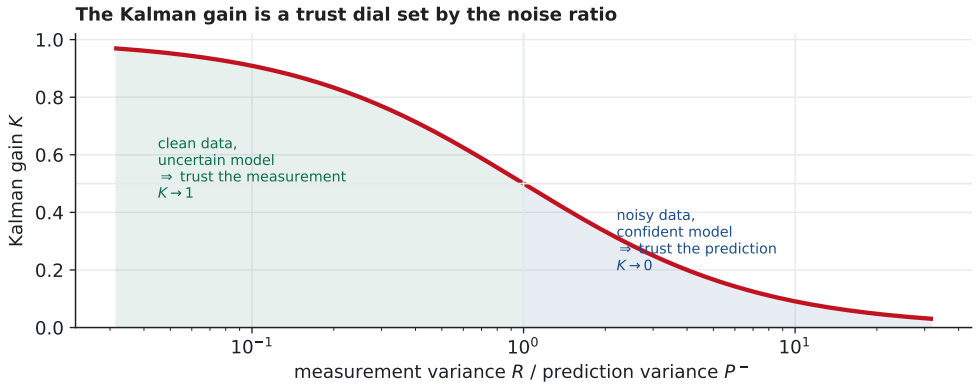

Figure 5: The Kalman gain as a trust dial. It depends only on the ratio of measurement noise r to predictive uncertainty P^-. When data is clean relative to the model's doubt (left), the gain approaches 1 and the filter follows the measurements. When data is noisy or the model is already confident (right), the gain approaches 0 and the filter holds its course. Every Kalman update is this dial, set afresh each step.

A self-tuning average. Notice what the filter is *not* doing: it has no window length, no smoothing parameter you tweak by hand. The effective amount of averaging emerges from q and r. If the level moves fast (q large), the filter keeps P_t^- high and stays responsive; if the level is nearly static (q small), P_t^- shrinks and the filter averages heavily, ignoring jitter. Over time P_t settles to a steady value and K_t to a steady gain, at which point the Kalman filter *is* an exponentially weighted moving average—but one whose decay rate was *derived* from your noise assumptions rather than guessed. That is the practical reason to reach for it: it replaces a tuning knob with a model.

Worked example: three steps by hand. Track a fair value with the scalar local level filter. Take process variance $q = 0.25$ and measurement variance $r = 1.0$, and start from a deliberately vague prior $\hat{x}_0 = 100.0$, $P_0 = 4.0$ ("I think it is near 100, but I am not sure"). Three prints arrive: $y = 101.3,\ 100.7,\ 101.9$. Each step is *predict* ($\hat{x}^- = \hat{x},\ P^- = P + q$) then *update* ($K = P^-/(P^- + r)$, $\hat{x} = \hat{x}^- + K(y - \hat{x}^-)$, $P = (1 - K)P^-$):

t	y_t	$\hat{x}_t^-$	P_t^-	innovation	K_t	$\hat{x}_t$	P_t
1	101.3	100.000	4.250	+1.300	0.810	101.052	0.810
2	100.7	101.052	1.060	−0.352	0.515	100.871	0.515
3	101.9	100.871	0.765	+1.029	0.433	101.317	0.433

Watch the gain fall. At $t = 1$ the vague prior makes the filter trust the data heavily ($K = 0.81$, it jumps most of the way to the first print); as P_t shrinks the filter grows confident and the gain settles, here toward its steady-state value $K_\infty = 0.39$ (the fixed point of $P^- = (1 - K)P^- + q$). From that point on the filter behaves as a fixed exponentially weighted moving average that keeps 39% of each new surprise—a decay rate *derived* from q and r, never tuned. Every line is arithmetic; no simulation is involved.

5 The filter in full

Real problems need more than one hidden number. A trend has a *level* and a *slope*; a yield curve has several factors; a regression has several coefficients. So let the state be a vector $\mathbf{x}_t \in \mathbb{R}^n$ and the observation a vector $\mathbf{y}_t \in \mathbb{R}^m$. The model becomes

$$\begin{aligned} \mathbf{x}_t &= F\,\mathbf{x}_{t-1} + \mathbf{w}_t, \quad \mathbf{w}_t \sim \mathcal{N}(\mathbf{0}, Q), \\ \mathbf{y}_t &= H\,\mathbf{x}_t + \mathbf{v}_t, \qquad \mathbf{v}_t \sim \mathcal{N}(\mathbf{0}, R), \end{aligned}$$

with four matrices that encode the modelling choices. F (the *transition matrix*) says how the state evolves on its own—e.g. "the new level is the old level plus the slope." Q (*process covariance*) says how much fresh randomness enters the state each step. H (the *observation matrix*) says how the state shows up in what you measure—e.g. "we observe the level but not the slope directly." R (*measurement covariance*) says how noisy the observations are. Choosing these four matrices *is* the act of model-building; the filter is then mechanical.

The scalar recursion generalises line for line, the only change being that scalars become matrices and division becomes matrix inversion:

The Kalman filter. Carry a state estimate $\hat{\mathbf{x}}_t$ and its covariance P_t. Each step:

$$\begin{aligned} &\textbf{Predict:} && \hat{\mathbf{x}}_t^- = F\,\hat{\mathbf{x}}_{t-1}, && P_t^- = F\,P_{t-1}\,F^\top + Q. \\ &\textbf{Update:} && \tilde{\mathbf{y}}_t = \mathbf{y}_t - H\,\hat{\mathbf{x}}_t^- && \text{(innovation)}, \\ && S_t = H\,P_t^-\,H^\top + R && \text{(innovation covariance)}, \\ && K_t = P_t^-\,H^\top S_t^{-1} && \text{(gain)}, \\ && \hat{\mathbf{x}}_t = \hat{\mathbf{x}}_t^- + K_t\,\tilde{\mathbf{y}}_t, && P_t = (I - K_t H)\,P_t^-. \end{aligned}$$

It is the same two beats. Predict advances the mean by F and inflates the covariance by Q. Update forms the innovation $\tilde{\mathbf{y}}_t$, scales the gain by the relative sizes of model and measurement uncertainty (S_t collects both), corrects the mean by a gained fraction of the innovation, and shrinks the covariance. You initialise with a prior $(\hat{\mathbf{x}}_0, P_0)$—often a vague one, a large P_0, meaning "I have little idea where the state starts, so let the first few observations dominate."

What you choose, and what the filter gives back. You supply F, Q, H, R and a prior. The filter returns, at every step, a full Gaussian belief: the estimate $\hat{\mathbf{x}}_t$ *and* the covariance P_t that plainly reports how unsure it is. That uncertainty is not decoration—it is what makes the

gain self-tune, and in trading it is what lets you size a position by conviction rather than treat every signal as equally trustworthy.

6 Worked example: estimating a level and a trend

Two concrete states make the machinery tangible. Both are computed by running the equations above on synthetic data.

A drifting level. The simplest useful filter is the scalar local level model of §4: $F = 1$, $H = 1$, and q, r chosen to match how fast the level wanders versus how noisy the prints are. Figure 6 runs it on observations of a slowly moving level that takes a sudden step partway through. The filter glides through the jitter and *adapts* at the break, where a fixed-window moving average must lag: the moving average cannot move faster than its window allows, while the filter's uncertainty P_t^- temporarily rises after the surprise, lifting the gain and letting it catch up. The shaded band is the filter's own $\pm 1\sigma$ uncertainty—wider in fast stretches, tighter when things settle.

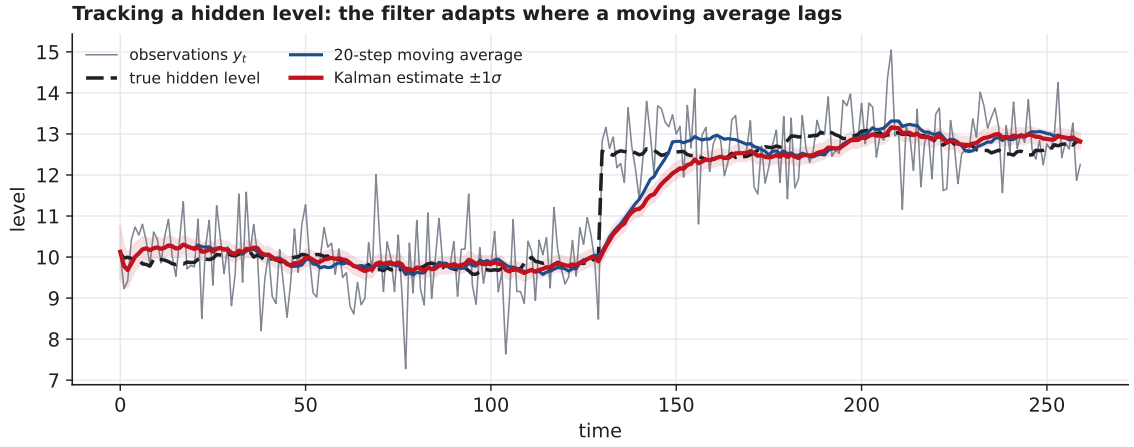

Figure 6: The local level filter at work. From noisy observations (grey) it recovers the hidden level (red, with its own $\pm 1\sigma$ band) and tracks a mid-series step that a fixed 20-step moving average (blue) can only follow with a lag. The filter has no window to choose: its responsiveness comes from the ratio q/r, and its uncertainty rises automatically after a surprise, then settles.

Adding a slope. Often you care not just where the level is but which way it is heading—the makings of a momentum signal. Promote the state to two components, $\mathbf{x}_t = (\text{level}_t,\ \text{slope}_t)$, with the dynamics "next level = level + slope, slope persists":

$$F = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, \qquad H = \begin{pmatrix} 1 & 0 \end{pmatrix},$$

so we observe the level but never the slope—the filter must *infer* the slope from how the level moves. This is the *local linear trend* model. Figure 7 shows it recovering both: the level tracks the series, and the slope (lower panel) reports the trend's sign and size, flipping as the series turns. Two things deserve note. First, the slope is inferred, never measured, yet the filter pins it down—a small lesson in how a state-space model can estimate quantities you cannot observe at all, just because they leave fingerprints in what you can. Second, the brief spike at the very start is the diffuse prior settling: with P_0 large the filter initially over-trusts the first observations, then calms. A trustworthy filter shows its warm-up.

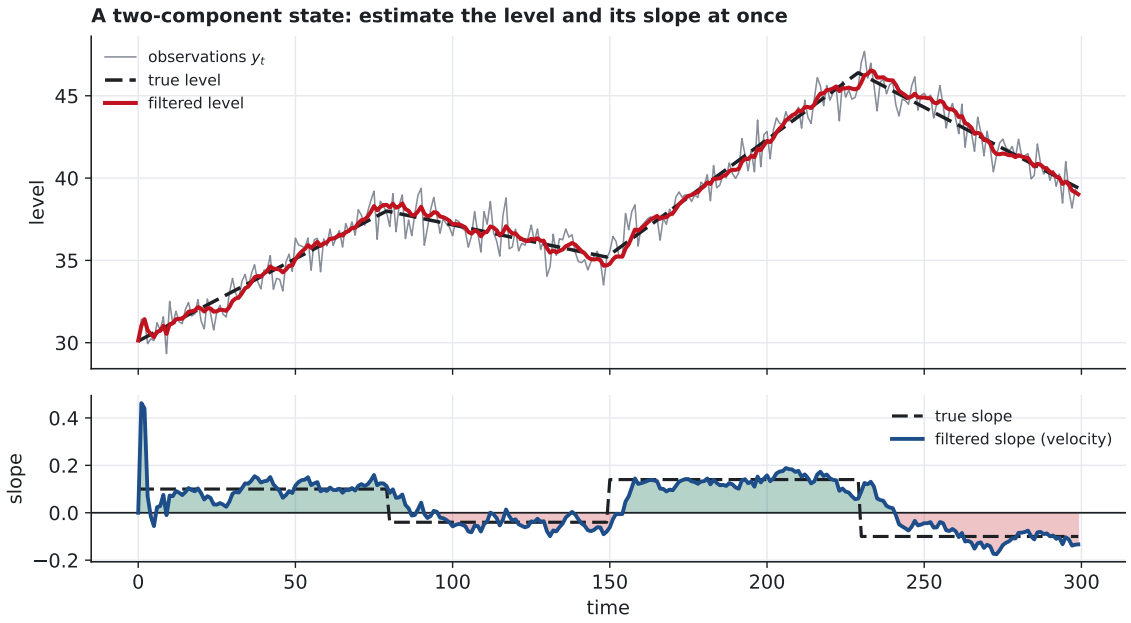

Figure 7: A two-component state. The filter estimates a hidden *level* (top) and a hidden *slope* (bottom) at once, observing only the level. The slope—an inferred velocity, never measured directly—reports the trend's direction (green up, red down) and turns as the series does. The opening transient is the vague prior settling in the first few steps.

7 In the markets: the dynamic hedge ratio

The canonical trading use of the Kalman filter is *pairs trading*, and it showcases a trick worth meeting: the filter can estimate a *regression coefficient that drifts over time*. Here the hidden state is not a price at all—it is a relationship. This section is the promised sequel to the companion note [1]: that note built the pairs trade with the hedge ratio fitted once; we now set the same ratio in motion.

The setup. Two assets, X and Y, tend to move together—two share classes, a stock and its sector ETF, two bond futures. Suppose a linear relationship holds on average,

$$y_t \;=\; \alpha_t + \beta_t\, x_t + (\text{small, mean-reverting spread}),$$

where β_t is the *hedge ratio*: how many units of X to short against one unit of Y so that the combined position is insulated from their common moves and exposed only to the residual *spread*. If that spread is stationary—it wanders but is tugged back to a mean (Appendix E recaps why this is the real precondition, called *cointegration*, and the companion note [1] develops it in full)—then you trade it: short the spread when it is unusually high, long when unusually low, and collect as it reverts. The companion note does exactly this for Goldman Sachs against Morgan Stanley, fitting a single hedge ratio $\beta \approx 3.30$ by the Engle–Granger regression and trading the resulting spread. Our quarrel is only with the word *single*.

Why the ratio cannot be a constant. A fixed β, fit once by least squares, ages badly: betas drift as fundamentals, liquidity and index weights change. The companion note is candid about this—it closes by recommending that a live trade re-estimate β on a rolling window and check that the spread still reverts out of sample. Rolling OLS is exactly that patch: refit β on the last N days. But it forces an awkward choice. A short window tracks change but is jumpy and noisy; a long window is smooth but stale. There is no good N, and the window throws away everything older than its edge as though it never happened.

The Kalman view. Make β_t (and the intercept α_t) the hidden state, and let it drift:

$$\mathbf{x}_t = \begin{pmatrix} \alpha_t \\ \beta_t \end{pmatrix}, \qquad F = I, \qquad H_t = \begin{pmatrix} 1 & x_t \end{pmatrix}, \qquad y_t = H_t\,\mathbf{x}_t + v_t.$$

Two features are new. The dynamics are a *random walk on the coefficients* ($F = I$, with a small process covariance Q that licenses slow drift)—this is how you say "the relationship is stable but not frozen." And the observation matrix H_t is *time-varying*: it carries the current regressor value x_t. With that, the very same filter equations from §5 run a regression whose coefficients evolve—an *online, adaptive least squares* with no window to pick. The process covariance Q plays the role the window length played, but continuously: larger Q lets β_t move faster.

Figure 8 shows the payoff. The hidden β_t drifts; the Kalman estimate tracks it almost exactly, while 60-step rolling OLS swings wildly around the truth—paying in variance for every bit of responsiveness. The lower panel turns the filter's residual into a tradable signal: the model-implied spread, standardised to a rolling z-score, with entry bands at ±2. This is *precisely* the signal the companion note trades—enter near ±2, exit at the mean—only now the spread it standardises comes from a hedge ratio that updates every step rather than one frozen at the start of the sample. The filter supplies the hedge ratio and the spread in one pass, updating both as each new pair of prices arrives, and its innovation $\tilde{y}_t = y_t - H_t\hat{\mathbf{x}}_t^-$ *is* the freshly de-trended spread—the same object the companion note built by hand, here produced as a by-product of the filter.

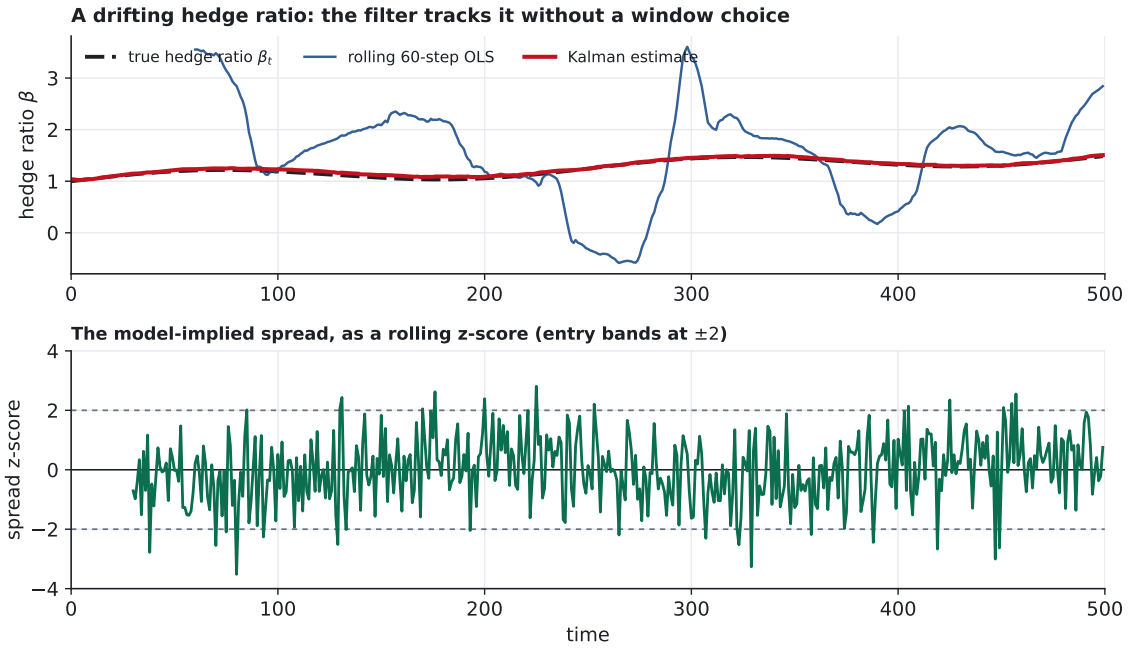

Figure 8: A drifting hedge ratio, tracked online. *Top:* the true time-varying β_t (dashed) is followed closely by the Kalman estimate (red), whereas 60-step rolling OLS (blue) is far noisier for the same responsiveness—the price of a fixed window. *Bottom:* the model's residual spread as a rolling z-score; crossings of the ±2 bands are entry signals, reversion to zero the exit. Hedge ratio and trading signal come from a single filter, updated each step with no window to choose.

The pattern to remember. Anything you would estimate by regression—a beta, a factor loading, a seasonal coefficient—can be made a *hidden state* and tracked through time by a Kalman filter, by letting its coefficients random-walk ($F = I$, small Q) and putting the regressors into a time-varying H_t. You trade a window length for a process variance, and gain a principled, real-time estimate with realistic error bars.

Part II — The hidden Markov model: a discrete, switching state

8 When the state is a regime, not a number

Sometimes the hidden thing is not a quantity that drifts but a *condition* that switches. Markets feel different in calm spells than in turbulent ones; trends give way to choppy ranges; liquidity is abundant until, abruptly, it is not. These are *regimes*: a small number of distinct states the world flips between, each with its own statistical character. The right hidden state here is not a real number but a *label*—which regime are we in?—and that calls for the second filter.

The model. A hidden Markov model keeps the same skeleton (Figure 2) but makes the state *discrete*: $x_t \in \{1, 2, \ldots, K\}$ for some small K. Two ingredients specify it.

Transitions. A $K \times K$ matrix A with $A_{ij} = p(x_t = j \mid x_{t-1} = i)$: the probability of moving from regime i to regime j next step. Its rows sum to one. Large diagonal entries mean *sticky* regimes that persist—turbulence, once it starts, tends to continue—which is exactly what makes regimes useful rather than noise.

Emissions. For each regime k, a distribution $p(y_t \mid x_t = k)$ for what you observe in that regime. For asset returns a natural choice is a Gaussian per regime: a calm regime emits small returns (low variance), a turbulent regime emits large ones (high variance), perhaps with a more negative mean. The regimes are never labelled in the data—you see only returns—but their differing fingerprints let the model tease them apart.

Figure 9 shows a synthetic return series generated from a two-regime model: long calm stretches punctuated by bursts of turbulence (shaded). The right panel shows the two emission distributions the model *recovered* from the returns alone—a narrow calm Gaussian and a fat turbulent one. Nobody told the model where the regimes were; it inferred both the regimes and their statistics, by a procedure we come to in §9.

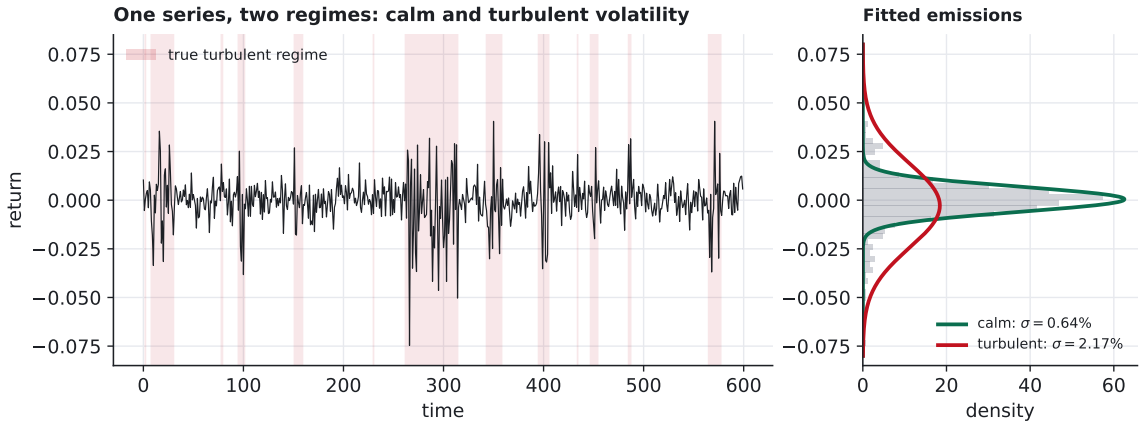

Figure 9: A two-regime world. *Left:* a single return series, with the true turbulent spells shaded; visually, volatility clusters rather than arriving at random. *Right:* the two Gaussian emission distributions recovered from the returns—a narrow "calm" and a fat "turbulent." The model is told neither where the regimes are nor their parameters; it learns both.

Why a Markov chain, and not just two buckets. You could ignore time and simply ask, return by return, "does this look calm or turbulent?" That throws away the most useful fact: regimes *persist*. The transition matrix encodes persistence, so the model resists flipping its mind

on a single odd return—if every recent return screamed calm, one large move is more likely noise than a regime change. Persistence is precisely the structure a plain classifier lacks and a Markov chain supplies. It is also what makes the filtered regime probability smooth enough to trade on, rather than a stream of jitter.

9 Three questions, three algorithms

Given an HMM, three questions arise in practice, and each has a clean, classic algorithm. All three are just the predict/update skeleton of §2.1 carried out with sums instead of integrals, because the state is discrete. The belief is now simply a vector of K probabilities.

9.1 Filtering: where am I now? (the forward algorithm)

This is the real-time question, and the direct analogue of the Kalman filter. Maintain a belief vector whose k-th entry is $p(x_t = k \mid y_{1:t})$, the probability of being in regime k given everything seen so far. The two beats:

Predict. Push the belief one step through the chain—multiply by the transition matrix A. This blends the regimes according to how likely each is to hand off to each other, spreading the belief out.

Update. Reweight each regime by how well it explains the new observation y_t—multiply, entrywise, by the emission likelihoods $p(y_t \mid x_t = k)$—then renormalise so the vector sums to one.

That is the *forward algorithm* (Appendix C writes the recursion in full, including the rescaling that keeps it numerically stable). Run online, it converts a stream of returns into a stream of regime probabilities. Figure 10 overlays the filtered probability of the turbulent regime on the true regimes: it rises within a step or two of trouble starting and falls when calm returns. It is occasionally fooled by a single large calm-regime return—fair enough, since in real time that is genuinely ambiguous—but the transition matrix keeps it from overreacting.

The figure also shows the *smoothed* probability $p(x_t = k \mid y_{1:T})$, which uses the *whole* series, future included, via a matching backward pass (the *forward–backward* algorithm). Smoothing is sharper and cleaner—but it peeks at the future, so it is for analysis and labelling history, never for live trading. The gap between the jumpy blue (filtered, tradable) and the crisp red (smoothed, hindsight) is a vivid picture of the cost of being candid about time.

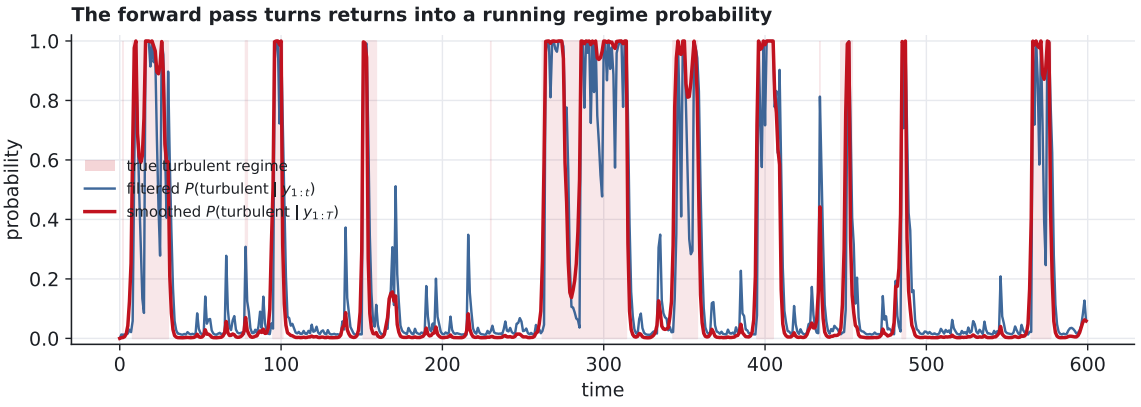

Figure 10: From returns to a regime probability. The forward pass (blue) reports, in real time, the probability of being in the turbulent regime; it tracks the true shaded spells with a step or two of lag and occasional understandable hesitation. The smoothed estimate (red) is cleaner but uses future data—fine for labelling history, off-limits for live trading.

9.2 Decoding: what was the most likely path? (Viterbi)

Filtering gives a probability for each regime at each time independently. Sometimes you want instead the single most likely *sequence* of regimes over the whole horizon—a clean segmentation of history into "calm here, turbulent there." Picking the most probable regime at each instant separately can produce a path the model considers jointly impossible (it might hop through a forbidden transition). The *Viterbi algorithm* finds the most likely whole path in one efficient dynamic-programming sweep, by carrying forward, for each regime, the best path that ends there, and tracing back at the end (Appendix D). Figure 11 shows its decoded path against the truth: on this series it recovers the regime correctly about 94% of the time. Viterbi, too, uses the full series, so it is a hindsight tool—for research, regime-labelling and diagnostics, not live signals.

9.3 Learning: where did the model come from? (Baum–Welch)

The forward and Viterbi passes assume you already know A and the emission parameters. Where do those come from? You *fit* them to data by maximum likelihood—but with a twist: you cannot count transitions directly, because you never observe the regimes. The way out is *expectation–maximisation* (EM), here called the *Baum–Welch algorithm*: guess the parameters; use forward–backward to compute soft, probabilistic regime assignments (the E-step); then re-estimate the parameters as if those soft assignments were data (the M-step); repeat. Each round provably increases the likelihood (Appendix C). The right panel of Figure 11 shows the likelihood climbing to a plateau over a few dozen iterations. Every HMM parameter in this note's figures was recovered this way, from returns alone—and it lands close to the values that generated the data, which is the reassurance you want before trusting it on data whose truth you do *not* know.

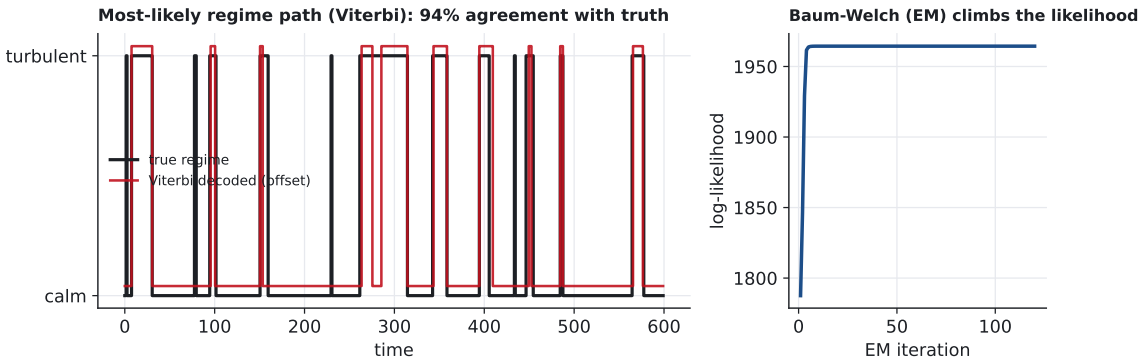

Figure 11: *Left:* the Viterbi most-likely path (red, offset for visibility) against the true regime sequence (black): about 94% agreement here. *Right:* Baum–Welch (EM) fitting the model from the returns alone—the log-likelihood rises monotonically to a plateau, and the recovered parameters land near the ones that generated the data.

Three questions, three tools. *Filtering* (forward) answers "which regime now?" in real time—the tradable one. *Decoding* (Viterbi) answers "what was the most likely path?" over history—a hindsight tool. *Learning* (Baum–Welch) fits the model to data when the regimes are unobserved. Only filtering is causal; the other two see the whole series and belong in research, not in a live signal.

10 In the markets: trading a regime signal

A filtered regime probability is a control dial for risk. The simplest, most defensible use is *defensive*: hold full exposure when the calm regime is likely, scale down as the turbulent regime's probability rises. Turbulent regimes combine larger swings with (often) worse average returns, so standing aside in them can both cut volatility and dodge drawdowns. Figure 12 illustrates this on the earlier series: exposure set to $1 - p(\text{turbulent})$, using the probability *lagged one step* so the decision uses only past information. The regime-scaled line is steadier than buy-and-hold, sidestepping the worst of the turbulent stretches.

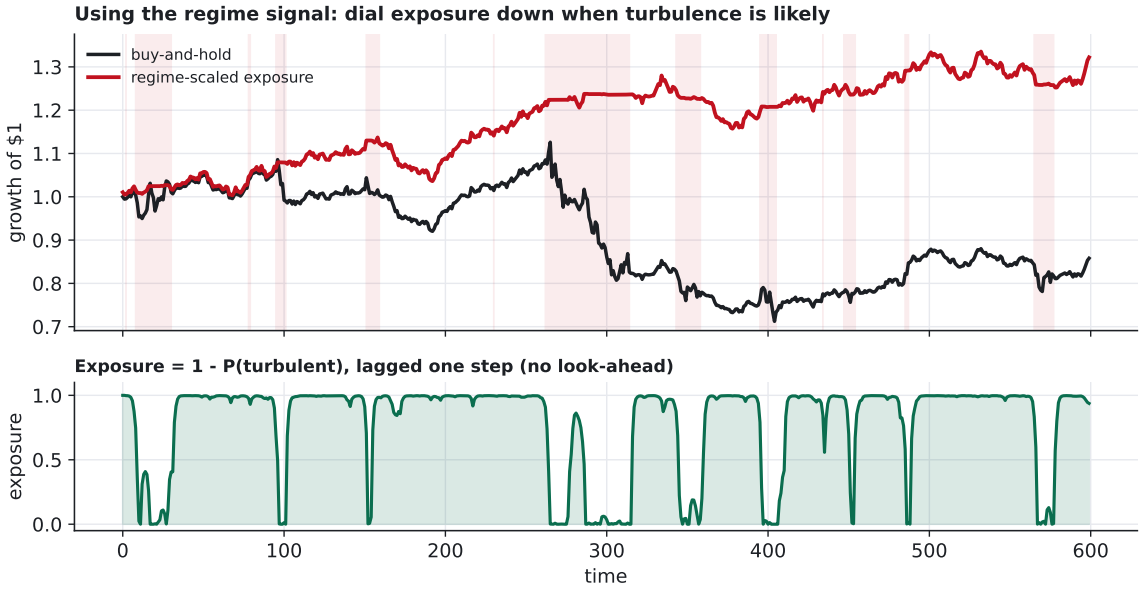

Figure 12: A regime signal as a risk dial. *Top:* exposure scaled by $1 - p(\text{turbulent})$ (lagged one step, no look-ahead) rides through the turbulent spells (shaded) more smoothly than buy-and-hold. *Bottom:* the exposure itself, retreating as turbulence becomes likely and returning as calm resumes. Illustrative on synthetic data—the caveats below are not optional reading.

Read this before believing the picture. Figure 12 is a clean result on *synthetic data generated by an HMM*, so of course an HMM does well—the model is true by construction. Real markets are not so kind, and the gap between this picture and a live strategy is where most of the difficulty lives. Section 12 is dedicated to it. The fair summary: regime models are most trustworthy as *risk overlays*—dialling exposure down in turbulence, where the statistical signal (volatility clustering) is real and well documented—and least trustworthy when asked to *time* regime changes for outright profit, where the lag and instability of the signal bite hardest.

Closing the loop with the pairs trade. A regime overlay is also a partial answer to the failure mode that haunts the companion note [1]: the structural break, where a cointegrated spread stops reverting and the strategy, still trusting the old relationship, rides a widening loss down. Such breaks rarely arrive in calm conditions; they cluster in the turbulent regime an HMM is built to flag. A regime signal cannot *predict* a break, but it can mark the conditions in which one is more likely and pull risk back accordingly—a statistical companion to the hard stop the other note recommends. And the two filters compose literally: a *switching Kalman filter* (§11) could run the drifting hedge ratio of §7 while a discrete regime governs how fast the ratio is allowed to move, or whether to trust the spread at all—the continuous tether and the discrete mood-of-the-market estimated together.

Synthesis

11 Choosing the shape of the hidden state

We have met two filters that share a skeleton and differ in the shape they give the hidden state. The choice between them is really a modelling question about the world (Figure 13, Table 1): *does the thing I cannot see drift continuously, or switch between conditions?*

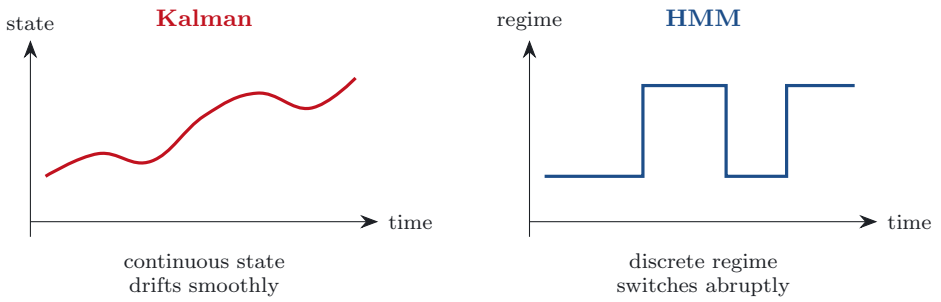

Figure 13: Two shapes of hidden state. The Kalman filter assumes a continuous state that *drifts* (left); the HMM assumes a discrete regime that *switches* (right). The filtering machinery—predict, then update—is identical; only the shape of the state, and therefore the form of the belief, differs.

	Kalman filter	**Hidden Markov model**
Hidden state	continuous vector $\mathbf{x}_t \in \mathbb{R}^n$	discrete label $x_t \in \{1, \dots, K\}$
Belief carried	mean $\hat{\mathbf{x}}_t$ + covariance P_t	probability vector (length K)
Dynamics	linear map F + Gaussian noise	transition matrix A
Observations	linear H + Gaussian noise	emission distribution per state
Real-time inference	the filter equations	forward algorithm
Full-series (hindsight)	RTS smoother	forward–backward / Viterbi
Fitting	MLE / EM for F, Q, H, R	Baum–Welch (EM)
Natural trading use	fair value, hedge ratio, trend	volatility / market regimes

Table 1: The two filters side by side. They are the same Bayesian recursion under two different assumptions about the hidden state.

When you need both, and beyond. The shapes combine. A *switching Kalman filter* (or regime-switching state-space model) lets a discrete regime choose *which* linear-Gaussian dynamics are running—a continuous fair value whose volatility, or whose drift, jumps with the regime. Exact filtering there is intractable (the number of possible regime histories explodes), so one approximates. And when the dynamics are genuinely nonlinear or the noise non-Gaussian, the closed forms fail entirely; the general fallback is a *particle filter*, which represents the belief by a cloud of weighted sample states and moves them through the same predict/update beats by simulation. These are the doors out of the two tidy rooms, opened when reality refuses to be linear, Gaussian, or finite-state—but the skeleton of §2 is unchanged, which is why it was worth building first.

12 Where these models break down

Both models are easy to make look brilliant in a figure and hard to make profitable in production. The difference is almost always one of the following, and naming them is the best defence.

The model is an assumption, not a fact. Linear-Gaussian dynamics and a handful of discrete regimes are convenient fictions. Real returns have fat tails, asymmetric moves, and volatility that varies continuously as much as it switches. When the world violates the model, the filter still produces confident-looking estimates—with credible-looking error bars that are nonetheless *wrong*, because they are computed under the false model. A tight P_t or a 99% regime probability is only as trustworthy as the assumptions beneath it.

Everything hinges on parameters you must estimate. The Kalman filter needs Q and R; the HMM needs A and the emissions. On synthetic data we knew or cleanly recovered them; on real data they are estimated noisily, drift over time, and the results can be sensitive to them. Set Q too high and the filter chases noise; too low and it lags reality. EM for an HMM can land in a poor local optimum, or split one true regime into two spurious ones. The estimates deserve as much scrutiny as the model.

Markets are non-stationary; the past mis-trains the future. These methods assume the process that generated history also generates the future. Markets regime-shift in ways no fixed transition matrix anticipates—a "crisis" regime fitted on one decade may not resemble the next. A model fit on calm years can be blindsided, and one fit through a crisis can be permanently spooked. Refitting helps and also introduces its own look-ahead and overfitting traps.

Regimes are seductive and easy to over-read. A two- or three-state HMM will *always* partition returns into "calm" and "turbulent" buckets, because volatility clustering is real—but that does not mean the regimes are causally meaningful, nor that you can *anticipate* the switches in time to profit. The robust, well-evidenced use is defensive risk scaling (§10); using regime calls to time directional bets demands far more skepticism, because the filtered signal lags real switches by exactly the delay that makes it causal.

In-sample beauty is not out-of-sample edge. The deepest trap, common to both: a smoothed regime path or a back-fit Kalman track laid over a finished chart looks clairvoyant—but smoothing and fitting both use the whole series. The only fair test is strict *walk-forward* evaluation: at each date, use only data available then, with parameters estimated only on the past, and never let a future observation touch a present decision. Most of the apparent magic of these filters evaporates under that discipline—and what survives it is the part actually worth trading.

The one-line verdict. The Kalman filter and the hidden Markov model are not crystal balls; they are disciplined, real-time *estimators of a hidden state* under explicit assumptions. Used where their assumptions roughly hold—a drifting relationship, a clustering volatility regime—and tested rigorously walk-forward, they turn noisy observations into usable, uncertainty-aware signals. Used as oracles, they will flatter you in-sample and disappoint you live. Respect the assumptions and the difference between filtering and hindsight, and they earn their keep.

13 A worked example: a pairs trade through a regime break

To finish, let us run the whole toolkit on one problem and watch each piece do the job it is built for. The problem is the one the companion note [1] opens with—a pairs trade—but carried

into the situation that note ends on: the tether *breaks*. Here all three methods earn their place. Time-series analysis *establishes* the tether and its hedge ratio; the Kalman filter *tracks* that ratio and re-learns it when it shifts; the hidden Markov model *detects* the regime in which the relationship is breaking. The punchline is that no one tool suffices—it is the combination that survives the break.

The setup. Two prices, X and Y, are cointegrated with hedge ratio $\beta = 1.30$ for the first stretch; then, at a structural break, the relationship shifts and β moves to 1.70 over a few weeks (a merger, an index reweighting, a change in business mix). The residual spread stays mean-reverting throughout—the pair is *not* permanently broken, it is re-tethered at a new ratio—which is exactly the case that punishes a hedge fitted once and never revisited. Everything below is computed on this synthetic series (Figure 14, top).

Step 1 — establish the tether (time-series analysis). Following the companion note, fit the hedge ratio by the Engle–Granger regression on the pre-break sample and test the residual for stationarity. The fit gives $\hat{\beta} = 1.27$; the Augmented Dickey–Fuller statistic on the residual is -5.04, well below the 5% critical value of -3.34, so the pair is comfortably cointegrated in-sample, with a mean-reversion half-life of about 7 days. We have a tether, a hedge ratio, and a spread to trade—the companion note's entire output, in two regressions.

Step 2 — track the tether (Kalman filter). Now make the hedge ratio a hidden state and run the filter of §7 over the full sample. Before the break the Kalman β_t sits right on the static $\hat{\beta}$—the filter *confirms* the stable tether (Figure 14, middle). After the break it does what the static fit cannot: it re-converges, climbing from 1.30 to about 1.68 within a few weeks and locking onto the new ratio. The consequence for the spread is decisive (Figure 15, top). The *static* spread $y - \hat{\beta}x$, computed with the frozen hedge ratio, runs away after the break to a dislocation of \$40 and more—it is no longer stationary, and the cointegration that licensed the trade is gone. The *Kalman* spread, computed with the re-learnt β_t, dips briefly during the transition and then reverts to zero: the filter has *repaired* the spread by re-learning the hedge. A mechanical pairs trader running the static spread is now trading a broken signal; the one running the Kalman spread has a tradeable signal again—once the dust settles.

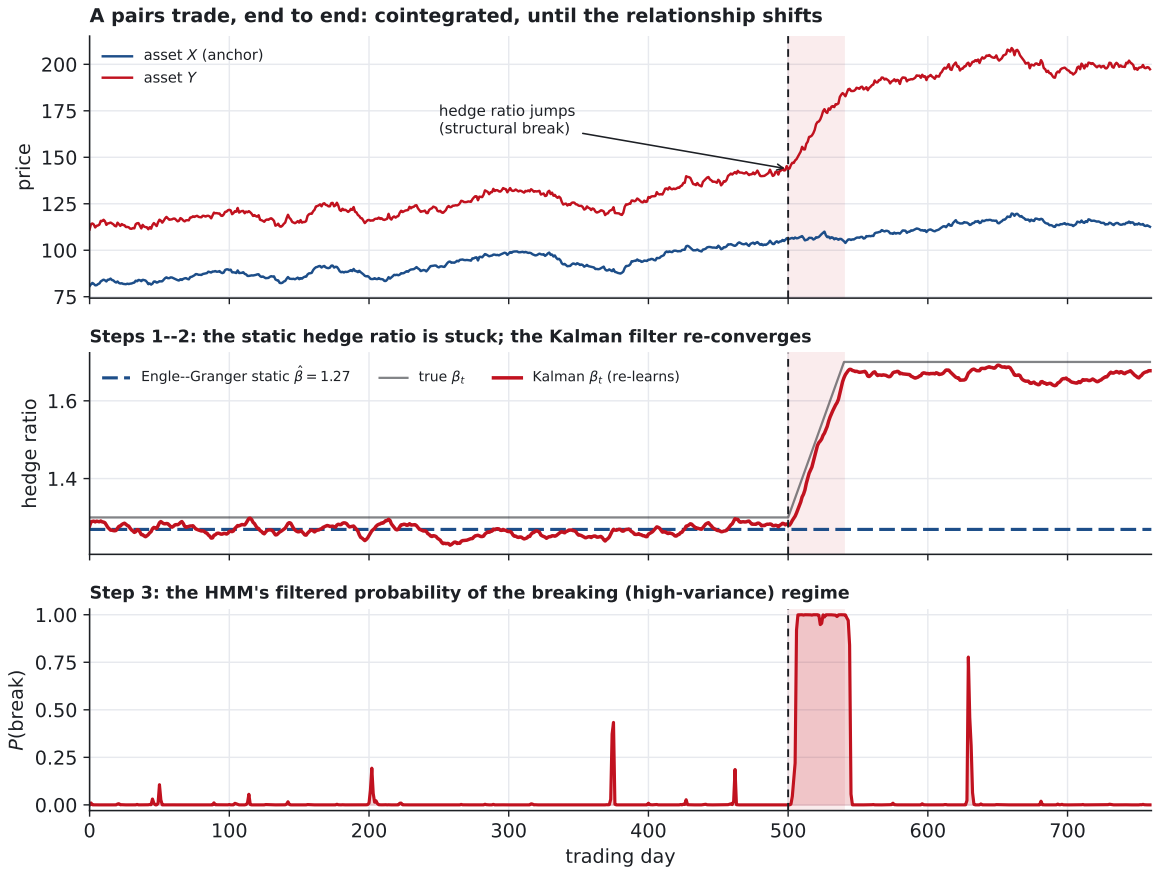

Figure 14: Diagnosing the break. *Top:* two cointegrated prices; at the dashed line the hedge ratio shifts (shaded transition). *Middle:* the Engle–Granger static $\hat{\beta} = 1.27$ (blue, dashed) is stuck, while the Kalman β_t (red) sits on it before the break and re-converges to the new ratio after. *Bottom:* the HMM's filtered probability of the breaking (high-variance) regime, computed from the Kalman filter's innovations—near zero in calm times, spiking to one exactly through the transition.

Step 3 — detect the break (hidden Markov model). Re-learning the hedge is not the same as knowing you are safe: *during* the transition, before β_t has caught up, the relationship is genuinely untradeable, and a filter quietly chasing the moving coefficient will not tell you so. This is where the regime layer comes in. Feed the Kalman filter's *innovations*—its one-step surprises—to a two-state HMM, fitted by Baum–Welch. One state is calm (small, zero-mean surprises: the model is fitting well); the other is the breaking regime (large surprises: the model is struggling). The filtered probability of the breaking regime (Figure 14, bottom) sits near zero through the calm stretch and spikes to one exactly across the transition, then subsides once the Kalman filter has re-locked. The HMM is watching the Kalman filter's own stream of errors and raising a flag the filter cannot raise about itself.

Step 4 — put it together, and trade. Now combine them. Trade the spread on the z-score rule from the companion note (enter at ± 2, exit at the mean), and compare three ways of running it (Figure 15, bottom):

- **Static Engle–Granger.** Trade the static spread. It does well until the break, then bleeds: the hedge is wrong, the spread trends away, and the rule keeps betting on a reversion that never comes. Final P&L: −$12.
- **Kalman, regime-blind.** Trade the Kalman spread, which re-learns the hedge—but trade *through* the break with no regime awareness. This is *worse*: −$26. Re-learning β is

necessary but not sufficient; while the filter is mid-transition the spread whipsaws violently, and a rule that keeps trading gets chopped up. Adapting the hedge does not save you from trading through the storm.

- **Kalman + HMM gate.** Trade the Kalman spread, but stand aside when the HMM's break probability is high (gate exposure by $1 - P(\text{break})$, lagged one step). This is the only one that comes through: +$11. It sits out the dangerous transition the HMM flagged, then re-enters once the filter has re-learnt the hedge and the spread is tradeable again.

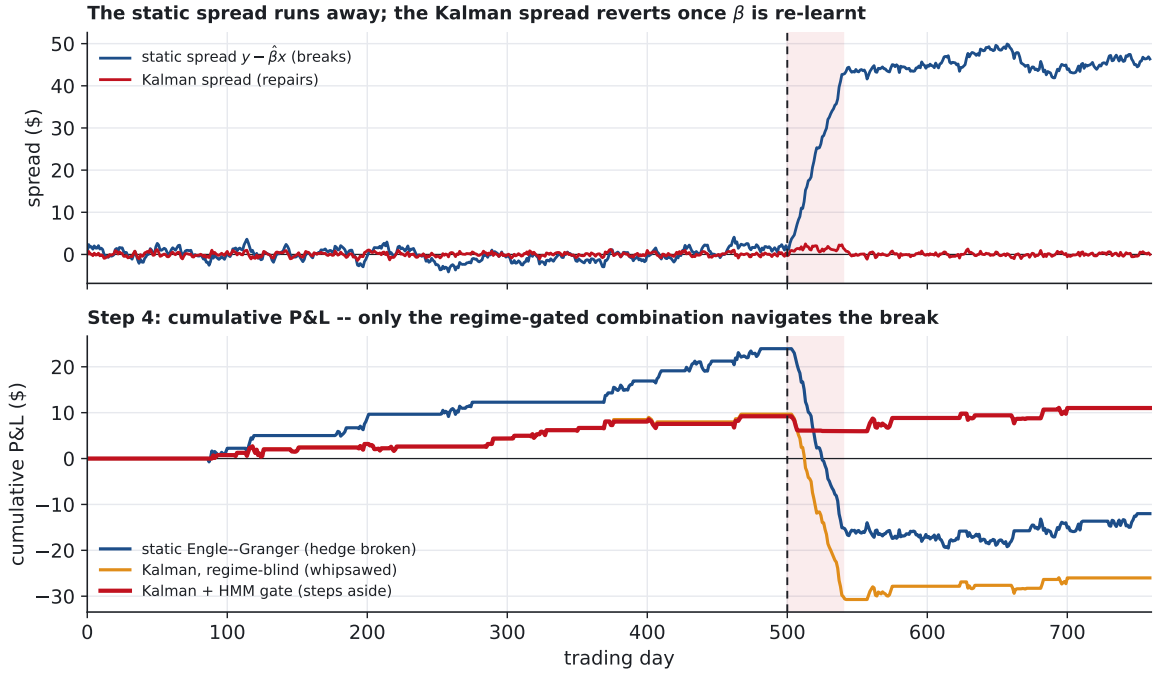

Figure 15: The consequence. *Top:* the static spread (blue) runs away after the break while the Kalman spread (red) reverts once β is re-learnt. *Bottom:* cumulative P&L of the three strategies. All three earn steadily before the break; at the break the static hedge bleeds and the regime-blind Kalman trade is whipsawed *worse*, while only the regime-gated combination steps aside and ends ahead. Synthetic data, gross of costs—read the caveat below before believing the level.

The pipeline, in one breath. *Time-series analysis* finds the tether and the hedge ratio and certifies the spread reverts (Engle–Granger, ADF). The *Kalman filter* tracks that hedge ratio online and re-learns it when it shifts, keeping the spread tradeable. The *HMM* watches the filter's innovations and flags the regime in which the relationship is breaking, so you can stand aside while it does. Each answers a question the others cannot: *is there a tether? what is its ratio right now? and is it breaking?*

Read this before believing the level. The same candour from §12 applies, doubled. This is synthetic data built to contain exactly one clean break, so of course the tools that model breaks do well; real breaks are messier, rarer, and not so obligingly followed by a clean re-tethering. The HMM's parameters were fitted in-sample (like the cointegration test itself), and the gating decision, though causal given those parameters, would in production demand walk-forward re-estimation. The regime flag *reacts* to the break, it does not predict it—notice the gated line still gives a little back at the onset before standing aside. And the P&L is gross of the transaction costs that, as the companion note's appendix stresses, decide whether a thin edge survives at all. What the example shows is not a strategy but a *division of labour*: three estimators of a hidden

state, each covering a blind spot of the others, and most trustworthy when their disagreements are respected rather than averaged away.

A The Gaussian toolkit

Everything in Part I rests on two facts about Gaussians: they are closed under the two operations filtering needs. Here they are, with just enough algebra to see why.

Fact 1: the product of two Gaussians is Gaussian. The Kalman update multiplies a Gaussian prior (the prediction) by a Gaussian likelihood (the measurement). In one dimension, if the prior is $\mathcal{N}(m_1, v_1)$ and the likelihood is $\mathcal{N}(m_2, v_2)$ in the same variable, their product is proportional to a Gaussian $\mathcal{N}(m, v)$ with

$$\frac{1}{v} = \frac{1}{v_1} + \frac{1}{v_2},$$
$$m = v\left(\frac{m_1}{v_1} + \frac{m_2}{v_2}\right).$$

Precisions (inverse variances) add; the mean is the precision-weighted average of the two means. Two consequences are worth internalising. The combined precision exceeds either input's, so $v < \min(v_1, v_2)$: combining evidence always sharpens the belief. And writing $K = v_1/(v_1 + v_2)$, the mean becomes $m = m_1 + K(m_2 - m_1)$—the prior nudged a fraction K toward the measurement. That K is the scalar Kalman gain of §4; the gain is nothing more exotic than the weight in a precision-weighted average. To prove the identities, add the two quadratic exponents and complete the square in the state variable—the cross terms assemble into exactly the (m, v) above.

Fact 2: conditioning a joint Gaussian gives a Gaussian. The Kalman update can also be read as: form the joint Gaussian of (hidden state, next observation), then condition on the observation you actually saw. For a bivariate Gaussian in (X, Y) with means (μ_X, μ_Y), variances (σ_X^2, σ_Y^2) and covariance σ_{XY}, the conditional of Y given $X = x$ is Gaussian with

$$\mathbb{E}[Y \mid X = x] = \mu_Y + \frac{\sigma_{XY}}{\sigma_X^2}(x - \mu_X),$$
$$\mathrm{Var}[Y \mid X = x] = \sigma_Y^2 - \frac{\sigma_{XY}^2}{\sigma_X^2}.$$

The conditional mean is linear in the observed x (that linear coefficient is, once more, a gain), and the conditional variance is *smaller* than the marginal σ_Y^2 by an amount set by how correlated the two are: a measurement informative about the state (σ_{XY} large) shrinks your uncertainty more. Figure 16 shows the geometry—slicing a tilted joint Gaussian at the observed value yields a narrower Gaussian along the hidden axis, recentred toward the data. The full Kalman filter is this picture in n dimensions, with the matrices of §5 in place of the scalars.

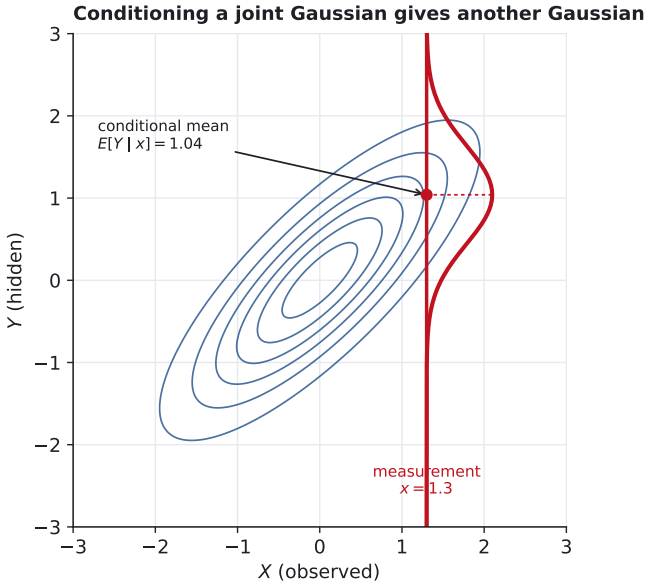

Figure 16: Conditioning a joint Gaussian. The blue contours are the joint distribution of an observed X and a hidden Y. Slicing at the measured value $x = 1.3$ (red line) gives the conditional distribution of Y (red curve)—Gaussian, recentred toward the data, and narrower than the unconditioned spread. This is one Kalman update, in two dimensions.

B Deriving the Kalman filter from Bayes

We derive the scalar local level update; the vector case is identical with matrices and is quoted in §5. Carry the belief after $t-1$ steps as $\mathcal{N}(\hat{x}_{t-1}, P_{t-1})$.

Predict. The state evolves as $x_t = x_{t-1} + w_t$ with $w_t \sim \mathcal{N}(0, q)$, independent of the past. The sum of independent Gaussians is Gaussian, with means and variances adding:

$$\begin{aligned} \hat{x}_t^- &= \mathbb{E}[x_{t-1}] = \hat{x}_{t-1}, \\ P_t^- &= \mathrm{Var}[x_{t-1}] + \mathrm{Var}[w_t] = P_{t-1} + q. \end{aligned}$$

Update. The measurement is $y_t = x_t + v_t$ with $v_t \sim \mathcal{N}(0, r)$. Given the predicted belief $x_t \sim \mathcal{N}(\hat{x}_t^-, P_t^-)$, the likelihood of the observation is $\mathcal{N}(y_t; x_t, r)$, viewed as a function of x_t a Gaussian in x_t with mean y_t and variance r. By Fact 1 of Appendix A, multiply prior and likelihood: precisions add and the mean is the precision-weighted average. With prior $(m_1, v_1) = (\hat{x}_t^-, P_t^-)$ and likelihood $(m_2, v_2) = (y_t, r)$,

$$\begin{aligned} P_t &= \left(\frac{1}{P_t^-} + \frac{1}{r} \right)^{-1} = \frac{P_t^- r}{P_t^- + r}, \\ \hat{x}_t &= P_t \left(\frac{\hat{x}_t^-}{P_t^-} + \frac{y_t}{r} \right). \end{aligned}$$

Now define the gain $K_t = P_t^- / (P_t^- + r)$ and the algebra tidies into the familiar form. For the mean,

$$\hat{x}_t = \hat{x}_t^- + K_t \, (y_t - \hat{x}_t^-),$$

and for the variance, $P_t = (1 - K_t) \, P_t^-$. The innovation $y_t - \hat{x}_t^-$ and the gain K_t are not ad hoc—they fall out of Bayes' rule and the Gaussian product. That is the whole filter: predict by adding variances, update by completing the square.

C The forward, backward, and Baum–Welch recursions

Let A be the $K \times K$ transition matrix, π the initial state distribution, and $b_k(y_t) = p(y_t \mid x_t = k)$ the emission likelihoods.

Forward (filtering). Define $\alpha_t(k) = p(x_t = k,\, y_{1:t})$. Then

$$\alpha_1(k) = \pi_k \, b_k(y_1),$$
$$\alpha_t(k) = b_k(y_t) \sum_{i=1}^{K} \alpha_{t-1}(i)\, A_{ik}.$$

The sum is the *predict* step (push through A); the multiply by $b_k(y_t)$ is the *update*. Normalising α_t to sum to one gives the filtered belief $p(x_t = k \mid y_{1:t})$, and the running normaliser at each step is $p(y_t \mid y_{1:t-1})$, whose logs sum to the total log-likelihood—the quantity Baum–Welch maximises and the number used in practice to avoid underflow (rescale α_t each step, accumulate the log of the scale).

Backward and smoothing. Define $\beta_t(k) = p(y_{t+1:T} \mid x_t = k)$, computed right-to-left:

$$\beta_T(k) = 1, \qquad \beta_t(k) = \sum_{j=1}^{K} A_{kj}\, b_j(y_{t+1})\, \beta_{t+1}(j).$$

Combining the two passes gives the smoothed posterior $\gamma_t(k) = p(x_t = k \mid y_{1:T}) \propto \alpha_t(k)\, \beta_t(k)$. This uses the whole series—hence hindsight only.

Baum–Welch (EM). With the regimes hidden, fit $(A, \pi, \text{emissions})$ by EM. The *E-step* computes, from current parameters, the smoothed state probabilities $\gamma_t(k)$ and the pairwise probabilities $\xi_t(i,j) = p(x_t = i, x_{t+1} = j \mid y_{1:T}) \propto \alpha_t(i)\, A_{ij}\, b_j(y_{t+1})\, \beta_{t+1}(j)$. The *M-step* re-estimates parameters as expected counts under those probabilities:

$$\hat{A}_{ij} = \frac{\sum_t \xi_t(i,j)}{\sum_t \gamma_t(i)}, \qquad \hat{\pi}_k = \gamma_1(k),$$
$$\hat{\mu}_k = \frac{\sum_t \gamma_t(k)\, y_t}{\sum_t \gamma_t(k)}, \qquad \hat{\sigma}_k^2 = \frac{\sum_t \gamma_t(k)\, (y_t - \hat{\mu}_k)^2}{\sum_t \gamma_t(k)},$$

the last two for Gaussian emissions. Iterating E and M never decreases the likelihood and converges to a local optimum—which is why initialisation matters and why a sane starting guess (emissions seeded from data quantiles, sticky transitions) earns its keep. This is exactly the procedure that produced the fitted parameters in Figures 9–11.

D Viterbi as dynamic programming

Viterbi finds $\arg\max_{x_{1:T}} p(x_{1:T} \mid y_{1:T})$—the single most likely regime path. The trick is that the best path ending in regime k at time t must extend a best path ending in *some* regime at $t-1$, so we can build it forward and need never enumerate the K^T possible paths. Work in logs to turn products into sums and avoid underflow. Let $\delta_t(k)$ be the log-probability of the best path ending in state k at time t:

$$\delta_1(k) = \log \pi_k + \log b_k(y_1),$$
$$\delta_t(k) = \log b_k(y_t) + \max_i \big[\delta_{t-1}(i) + \log A_{ik}\big],$$

recording at each step the maximising predecessor $\psi_t(k) = \arg\max_i[\dots]$. At the end, the best final state is $\arg\max_k \delta_T(k)$, and following the stored ψ pointers backward reconstructs the whole optimal path. The structure mirrors the forward algorithm with "max" in place of "$\sum$"—the difference between *the most likely single path* and *the total probability summed over all paths.*

E Why the spread is tradable: stationarity and cointegration

The pairs trade of §7 works only if the residual spread is *stationary*: it wanders but is repeatedly pulled back toward a stable mean, so that an unusually large value is a bet that it will revert. Two asset prices are typically each *non*-stationary—they random-walk, drifting without a fixed level, so their difference normally wanders off too. The special case where some linear combination $y_t - \beta x_t$ *is* stationary even though x_t and y_t individually are not is called *cointegration*: the two prices share a common stochastic trend that the combination cancels, leaving a mean-reverting residual. That residual is the spread you trade.

Two cautions follow. First, cointegration is a property to *test* (with, e.g., an Engle–Granger or Johansen test), not assume—many pairs that look related are merely both trending, and their spread will not revert. Second, the relationship can *break*: a structural change severs the cointegration and the spread trends away instead of reverting, which is the pairs trader's classic blow-up. The Kalman filter's drifting β_t accommodates *gentle* change in the relationship, but it cannot save you from a genuine break—and may even mask it for a while by chasing the moving coefficient. A widening, non-reverting spread and a β the filter keeps revising in one direction are warnings to heed, not noise to filter out.

This appendix is only the one-paragraph version of the precondition. The full apparatus—how to *test* for a unit root (the Augmented Dickey–Fuller test and its non-standard critical values), how to estimate β and certify the pair in one stroke (the Engle–Granger two-step), how the mean-reversion *half-life* sizes the holding period, how the error-correction model explains *why* a tethered spread must revert, and how to scale from a single pair to a whole basket (vector autoregressions and the Johansen procedure)—is the subject of the companion note [1]. This note takes the existence of a tradeable spread as given and asks the next question: how to track its hedge ratio as it moves, and how to read the market mood around it.

References

[1] M. Burke. *Time-Series Analysis for Pairs Trading and Cointegration.* Companion note, 2026. (Random walks and unit roots, the Augmented Dickey–Fuller test, the Engle–Granger two-step, mean-reversion half-life, the error-correction model, and the Johansen procedure.)

[2] R. E. Kalman. "A New Approach to Linear Filtering and Prediction Problems." *Journal of Basic Engineering*, 82(1):35–45, 1960.

[3] L. R. Rabiner. "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition." *Proceedings of the IEEE*, 77(2):257–286, 1989.

[4] J. D. Hamilton. "A New Approach to the Economic Analysis of Nonstationary Time Series and the Business Cycle." *Econometrica*, 57(2):357–384, 1989.

[5] A. C. Harvey. *Forecasting, Structural Time Series Models and the Kalman Filter.* Cambridge University Press, 1989.

[6] J. Durbin and S. J. Koopman. *Time Series Analysis by State Space Methods.* 2nd ed., Oxford University Press, 2012.

[7] C. M. Bishop. *Pattern Recognition and Machine Learning.* Springer, 2006. (Chapter 13, sequential data.)

[8] K. P. Murphy. *Machine Learning: A Probabilistic Perspective.* MIT Press, 2012. (State-space models and HMMs.)

[9] M. West and J. Harrison. *Bayesian Forecasting and Dynamic Models.* 2nd ed., Springer, 1997.

[10] E. P. Chan. *Algorithmic Trading: Winning Strategies and Their Rationale.* Wiley, 2013. (Kalman-filtered pairs trading.)