Neural Network Online Training With Sensitivity to Multiscale Temporal Structure

Matt Jones, David Mayo, Tyler Scott, Mengye Ren,
Gamaleldin ElSayed, Katherine Hermann, Michael C. Mozer
Google Research, Brain Team
Mountain View, CA 94043
mcjones@google.com

Abstract

Many online-learning domains in artificial intelligence involve data with nonstationarities spanning a wide range of timescales. Heuristic approaches to combat nonstationarity include retraining models frequently with only the freshest data and using iterative gradient-based updating methods that implicitly discount older data. We propose an alternative approach based on Bayesian inference over $1/f$ noise. The method is cast as a Kalman filter that posits latent variables with various characteristic timescales and maintains a joint posterior over them. We also derive a variational approximation that tracks these variables independently. The variational method can be implemented as a drop-in optimizer for any neural network architecture, which works by decomposing each weight as a sum of subweights with different decay rates. We test these methods on two synthetic, online-learning tasks with environmental parameters varying across time according to $1/f$ noise. Baseline methods based on bounded memory show a nonmonotonic relationship between memory horizon and performance, a signature of data going "stale." The Bayesian and variational methods perform significantly better by leveraging all past data and performing appropriate inference at all timescales.

Many online tasks facing both biological and artificial intelligence systems involve changes in data distribution over time. In machine learning, this situation often manifests as poorer generalization performance on future data versus on held-out data from within the training interval. Common solutions are to train on a window of fixed length (to exclude older "stale" data) or stochastic gradient descent (SGD) with fixed learning rate and weight decay, so that older data are less influential [DRAP15]. Here we demonstrate that performance can be significantly improved by retaining all data and using a learning model that reflects the temporal structure of the environment.

Methods based on SGD or temporal-difference learning with constant learning rates implicitly assume a dynamic environment, but following a random walk or diffusion process that exhibits only short-range correlations (see Appendix B). In contrast, real environments, both natural and human-constructed, tend to have correlations at a wide range of temporal and spatial scales, as often modeled by the $1/f$ noise family of stochastic processes [Kes82]. We argue that such structure can be exploited by building it into existing learning methods.

We develop a family of Bayesian models in which the prior (as a stochastic process) is defined by $1/f$ noise, represented as a sum of diffusion processes at different timescales (Appendix A). This generative model reflects an assumption that natural environments comprise a variety of events with different magnitudes and durations. Exact Bayesian inference is possible using a Kalman filter (KF) that tracks the component processes jointly [KTS07]. When learning a single environmental parameter θ, such as mean reward for some action in a bandit task, this amounts to modeling $\theta = \sum_{i=1}^{n} z_i$, where each z_i is a stochastic process with a different characteristic timescale τ_i, and doing joint inference over $\boldsymbol{Z} = (z_1, \ldots, z_n)$.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

We then generalize this approach to an arbitrary statistical model, $h(\boldsymbol{x}, \boldsymbol{\theta})$, where $\boldsymbol{x}$ is the input and $\boldsymbol{\theta} \in \mathbb{R}^m$ is a latent parameter vector to be estimated. For instance, h might be a neural network (NN) with parameters $\boldsymbol{\theta}$. Our Bayesian model places a $1/f$ prior on $\boldsymbol{\theta}$ (as a stochastic process), by assuming $\boldsymbol{\theta}(t) = \sum_{i=1}^{n} \boldsymbol{z}_i(t)$ for diffusion processes $\boldsymbol{z}_i \in \mathbb{R}^m$ with respective characteristic timescales τ_i. We then do inference over the joint state $\boldsymbol{Z} = (\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n)$, by extending the exact Bayesian KF above to an extended Kalman filter (EKF) that linearizes h by calculating its Jacobian at each step [SW89, PF03]. Next, we derive a variational approximation to the EKF that constrains the covariance matrix to be diagonal. The variational approximation admits an interpretation wherein each weight estimate w_j, defined as the current prior mean of θ_j, is decomposed as a sum of subweights, $w_j = \sum_i \omega_{ij}$ (i indexing timescales and j indexing weights in the network). The subweights are learned independently, each with a different learning rate and decay rate determined by τ_i. This can be seen as a new variation of fast-weight methods [HP87, BHM+16], involving many timescales and grounded in a Bayesian interpretation as inference over $1/f$ noise. The result can serve as a drop-in replacement for other neural network optimizers, and is sensitive to $1/f$-type temporal structure.

We test the KF-based methods in synthetic online learning tasks (Sections 2 and 3), where the latent parameters vary over time according to $1/f$ noise. An important caveat is that the task domains are matched to the Bayesian model in that the data are generated using the same sum-of-diffusion process that the model assumes. Notwithstanding this, we test robustness by using a different set of timescales for sampling versus inference (see Appendix F) and, in Section 3, a sampling process that mismatches the NN architecture. Results show that the Bayesian methods (KF and EKF) outperform heuristic methods of windowing or online SGD, demonstrating the value of modeling temporal structure. We also find that the variational approximation performs nearly as well as the full model, which speaks to the feasibility of this approach in contemporary applications of large NNs.

1 Bayesian inference over $1/f$ noise

Let $z_i(t)$ be an Ornstein-Uhlenbeck process (i.e., diffusion with decay), with timescale or inverse decay rate τ_i and diffusion rate σ_i^2, defined by the following stochastic differential equation:

$$\mathrm{d}z_i = -\tau_i^{-1} z \,\mathrm{d}t + \sigma_i \,\mathrm{d}W. \tag{1}$$

Here $W(t)$ is a standard Wiener process (Brownian motion). For a chosen $\nu > 1$, define $\tau_i = \nu^i$ and $\sigma_i = \nu^{-i/2}$, and choose n so that τ_n is vey large. We show in Appendix A that the summed process,

$$\xi(t) = \sum_{i=1}^{n} z_i(t), \tag{2}$$

has a power spectrum that is well approximated by $1/f$, for frequencies $f \gg \tau_n^{-1}$. Moreover, m independent copies of this process constitute m-dimensional $1/f$ noise, due to the rotational invariance of multidimensional Ornstein-Uhlenbeck processes.

Note that $\boldsymbol{Z} = (z_1, \ldots, z_n)$ is a linear dynamic system, with linear observation model ξ. Thus if observations are made at discrete intervals, say $y(t) = \xi(t)$ for $t \in \mathbb{N}$, then optimal Bayesian online prediction of each $y(t)$ based on all preceding observations is implemented by a KF over $\boldsymbol{Z}$ [KTS07].

This framework suggests a method for statistical optimization, including training NNs, in nonstationary domains. Assume we receive observations $\boldsymbol{x}(t), y(t)$ that we wish to explain using a model h with latent parameters $\boldsymbol{\theta} \in \mathbb{R}^m$. For example, h may be a NN with weights $\boldsymbol{\theta}$, input $\boldsymbol{x}$, and output y:

$$y(t) = h(\boldsymbol{x}(t), \boldsymbol{\theta}(t)). \tag{3}$$

Building on h, we define a Bayesian model that makes the generative assumption $\boldsymbol{\theta} = \sum_{i=1}^{n} \boldsymbol{z}_i$, where each $\boldsymbol{z}_i$ is an Ornstein-Uhlenbeck process in $\mathbb{R}^m$ with timescale τ_i, so that $\boldsymbol{\theta}$ follows a $1/f$ process. We then define the latent state $\boldsymbol{Z} = (\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n) \in \mathbb{R}^{nm}$ and do inference over $\boldsymbol{Z}$.

When h is linear in $\boldsymbol{\theta}$ (and hence in $\boldsymbol{Z}$), as in the regression task and one-layer perceptron of Section 2, exact inference is possible with a standard KF (Appendix C). For a general h, such as a multilayer NN, we use an EKF (Appendix D). The EKF makes a local linear approximation of h based on its Jacobian, the matrix of gradients of predictions $\hat{\boldsymbol{y}}$ with respect to $\boldsymbol{\theta}$. We use Ollivier's generalization of the EKF [Oll18] that replaces Gaussian observation noise with any exponential family $p(y|\hat{\boldsymbol{y}})$, which is better suited for modeling discrete outcomes such as the classification task of Section 3.

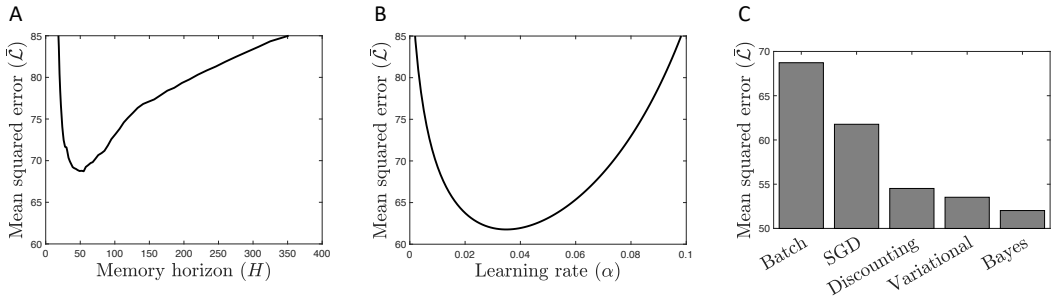

Figure 1: Regression task with $1/f$ dynamics. Loss is squared error. A: Batch learner. B: Stochastic gradient descent. C: Model comparison.

Finally, we derive a variational approximation of the EKF, suitable for efficient implementation in large NNs. Recall that a KF or EKF maintains an iterative prior based on all previous observations:

$$\boldsymbol{Z}(t)|\boldsymbol{x}_{<t}, \boldsymbol{y}_{<t} \sim \mathcal{N}(\boldsymbol{\omega}(t), \boldsymbol{S}(t)). \tag{4}$$

The mean, $\boldsymbol{\omega}(t)$, is the vector of current subweight estimates in the network, while $\boldsymbol{S}(t)$ captures their joint uncertainty and hence determines updates (as a preconditioner). We approximate this distribution by one in which $\boldsymbol{S}(t) \approx \tilde{\boldsymbol{S}}(t)$, where $\tilde{\boldsymbol{S}}(t)$ is constrained to be diagonal (Appendix E). This reduces the complexity from $\mathcal{O}(m^2n^2)$ to $\mathcal{O}(mnk)$ (the size of the Jacobian, where k is the size of the output layer). Under this approximation, each ω_{ij} is updated independently, based on the current loss gradient. Thus the variational method amounts to decomposing every weight in the network as a sum of subweights, $w_j = \sum_i \omega_{ij}$, with decay rates τ_i and learning rates coming from $\tilde{\boldsymbol{S}}(t)$. Note that this multiplexing is not expensive relative to current optimizers (e.g., Adam [KB15]), which also store multiple variables for each weight.

2 Regression task

As a simple demonstration, we created a linear regression task with 10 features (including a bias term), in which the true weights $\boldsymbol{\beta}$ varied over time according to $1/f$ noise. The outcome was generated as $y = \boldsymbol{x}^\top \boldsymbol{\beta}$ (no noise term was needed because $\boldsymbol{\beta}$ is inherently noisy). The corresponding predictive model is a one-layer perceptron, which we write as $\hat{y} = \boldsymbol{x}^\top \boldsymbol{w}$ to distinguish weight estimates from true parameters. We model the data using the perceptron and compare methods for optimization.

We tested two baseline training methods, representing common heuristic practices with nonstationary data [DRAP15, PKP$^+$19]. First, we tested a batch model that uses a fixed memory horizon H. To generate a prediction on step t, the batch learner fits the perceptron to trials $t - H$ through $t - 1$. Figure 1A shows performance is U-shaped: accuracy suffers with short horizons because of sampling error, but it also suffers from longer horizons because older observations are less valid. Second, we tested SGD, in which the weights are updated once after each step t, based on $\boldsymbol{x}(t)$ and $y(t)$. Figure 1B shows performance is best with an intermediate learning rate, which roughly corresponds to assuming the environment changes on a single characteristic timescale (see Appendix B).

As applied to the perceptron, the Bayesian $1/f$ model described in Section 1 decomposes the weight for each feature j into subweights, $w_j = \sum_i \omega_{ij}$, and tracks the ω_{ij} jointly with a KF (generalizing [DK00]). The subweights combine to predict the outcome on each step: $\hat{y}(t) = \sum_{ij} x_i \omega_{ij}$. Relative to blind prediction (guessing the overall mean of y on every trial), this exact Bayesian solution explains 54.0% more variance in the outcome than the best batch learner, and 25.7% more than the best parameterization of SGD (Figure 1C). Moreover, the variational model that constrains the KF covariance matrix to be diagonal performs nearly (96.9%) as well as the full Bayesian model.

Finally, we tested a discounting method, similar to the batch learner except all past trials were used for training, discounted by a function of lag. Because the $1/f$ environment has power-law correlations, we weighted each observation $t - k$ by k^{-a} and optimized a. This method also significantly outperforms windowed batch and SGD, showing that accounting for an environment's autocorrelation function can achieve much of the advantage of the Bayesian approach. Nevertheless, the full KF and variational methods still outperform discounting, by 5.5% and 2.2% respectively.

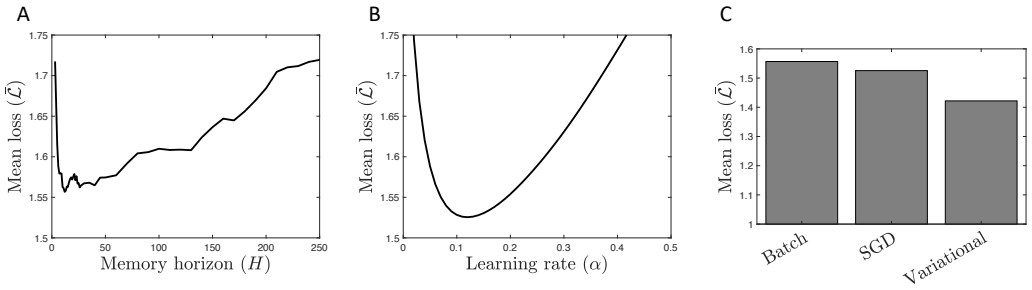

Figure 2: Classification task with $1/f$ dynamics. Loss is negative log-likelihood. A: Batch learner. B: Stochastic gradient descent. C: Model comparison.

3 Classification task

Next, we investigated a 10-way classification task with 10 features (including a bias term). The data were generated by first sampling the class, $y(t) \sim \mathrm{softmax}(\boldsymbol{e}(t))$, and then sampling the feature vector, $\boldsymbol{x}(t)|y(t) \sim \mathcal{N}(\boldsymbol{\mu}_{y(t)}, \boldsymbol{I})$. The logits e_j and the feature-class means μ_{ij} were independently sampled from $1/f$ processes, so that there was nonstationarity in both $p(y)$ and $p(\boldsymbol{x}|y)$. Scaling of $\boldsymbol{e}$ and $\boldsymbol{\mu}$ was chosen to equate maximum possible performance based on perfect knowledge of either one alone (both yielding average loss $\bar{\mathcal{L}} \approx 1$). For the predictive model, we used a perceptron with a softmax output layer: $\hat{\boldsymbol{y}} = \mathrm{softmax}(\boldsymbol{x}^\top \boldsymbol{W})$, where $\boldsymbol{W}$ is a matrix of learnable feature-class weights.

The batch method trained the network on trials $t-H$ through $t-1$ until convergence before predicting $y(t)$. Weight decay was included for regularization, with the decay rate optimized for each value of H. Figure 2A shows a U-shaped pattern of performance, reflecting the tradeoff between sampling error and stale data. We also used weight decay with SGD, optimized for each learning rate. Figure 2B again shows a U-shaped pattern of performance.

The variational EKF uses a local linear approximation of the network, $\frac{\partial \hat{y}}{\partial w}$, and the variance of $\boldsymbol{T}(y)$ conditioned on $\hat{\boldsymbol{y}}$ (where $\boldsymbol{T}(y)$ is a one-hot encoding) as inputs to the standard KF update equations (see Appendix E.3). We enforce the diagonal approximation of the covariance matrix when calculating the posterior after each observation. The optimal approximation uses the diagonal of the precision matrix but can be calculated without matrix inversion, which is relevant to scaling up to large networks. We also applied ℓ_2 regularization to the prior for the next timestep, on par with the SGD and batch methods. Figure 2C shows the variational method outperforms the other two.

4 Conclusions

Our results offer an initial demonstration of how online learning performance in nonstationary environments can be improved by incorporating a model of temporal structure. The Bayesian $1/f$ model amounts to distributing knowledge across multiple timescales, and the variational approximation enables implementation in a neural network using subweights with different learning and decay rates. The latter method is closely related to models of learning at multiple timescales that have been proposed in psychology and neuroscience [Jon22]. It is also similar to power-law discounting of past observations, in that a sum of subweights decaying at different exponential rates approximates power-law decay. Nevertheless, the variational method was seen to perform better, presumably because of the interaction among subweights (via learning from a shared gradient) that approximates Bayesian allocation of knowledge to different timescales. Another possible variational method is to assume a block-diagonal covariance matrix, in which covariance information is maintained between subweights (timescales) within each weight but not across weights, so that computational complexity still scales linearly with network size. Follow-up work applying the variational EKF to multilayer networks trained on real data [JSR+22] further supports the potential of these methods to scale up to large networks and complex online-learning domains.

References

[BHM+16] Jimmy Ba, Geoffrey E. Hinton, Volodymyr Mnih, Joel Z. Leibo, and Catalin Ionescu. Using fast weights to attend to the recent past. *Advances in neural information processing systems*, 29, 2016.

[DK00] Peter Dayan and Sham Kakade. Explaining away in weight space. *Advances in neural information processing systems*, 13, 2000.

[DRAP15] Gregory Ditzler, Manuel Roveri, Cesare Alippi, and Robi Polikar. Learning in nonstationary environments: A survey. *IEEE Computational Intelligence Magazine*, 10(4):12–25, 2015.

[HP87] Geoffrey E. Hinton and David C. Plaut. Using fast weights to deblur old memories. In *Proceedings of the 9th annual conference of the cognitive science society*, pages 177–186, 1987.

[Jon22] Matt Jones. Learning at multiple timescales. In *NeurIPS workshop on Memory in Artificial and Real Intelligence (MemARI)*, 2022.

[JSR+22] Matt Jones, Tyler Scott, Mengye Ren, Gamaleldin ElSayed, Katherine Hermann, David Mayo, and Michael C. Mozer. Learning in temporally structured environments. In *Submitted to Twelfth International Conference on Learning Representations (ICLR23)*, 2022.

[KB15] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

[Kes82] Marvin S. Keshner. 1/f noise. *Proceedings of the IEEE*, 70(3):212–218, 1982.

[KTS07] Konrad P. Kording, Joshua B. Tenenbaum, and Reza Shadmehr. The dynamics of memory as a consequence of optimal adaptation to a changing body. *Nature Neuroscience*, 10(6):779–786, 2007.

[Oll18] Yann Ollivier. Online natural gradient as a kalman filter. *Electronic Journal of Statistics*, 12(2):2930–2961, 2018.

[PF03] Gintaras V Puskorius and Lee A Feldkamp. Parameter-based kalman filter training: Theory and implementation. In Simon Haykin, editor, *Kalman Filtering and Neural Networks*, pages 23–67. John Wiley & Sons, Inc., 2003.

[PKP+19] German I. Parisi, Ronald Kemker, Jose L. Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019.

[Sut92] Richard S Sutton. Gain adaptation beats least squares. In *Proceedings of the 7th Yale workshop on adaptive and learning systems*, volume 161, page 166, 1992.

[SW89] Sharad Singhal and Lance Wu. Training multilayer perceptrons with the extended kalman algorithm. volume 1, 1989.

A Generative model for $1/f$ noise

Consider a single Ornstein-Uhlenbeck (OU) process $z(t)$ described by Equation 1 with $\sigma = 1$. The covariance function of z is given by

$$\mathbb{E}[z(t)z(t+s)] = \int_{-\infty}^{t} e^{-(t-t')/\tau} e^{-(t+s-t')/\tau} \mathrm{d}t' \tag{5}$$

$$= \frac{\tau}{2} e^{-s/\tau}. \tag{6}$$

Note that this expression decays exponentially with the lag s, yielding short-range autocorrelations. The power spectrum of z, as a functinon of frequency f, is the Fourier transform of the covariance function:

$$P_z(f) = \int_{\mathbb{R}} \frac{\tau}{2} e^{-|s|/\tau} e^{-2\pi i f s} \mathrm{d}s \tag{7}$$

$$= \frac{1}{\tau^{-2} + (2\pi f)^2} \tag{8}$$

where $2\pi f$ is angular frequency.

To define a generative model of $1/f$ noise, we define a continuous mixture model over a family of OU processes $(z_\tau)_{0<\tau\leq T}$ for some large T:

$$\xi(t) = \int_0^T 2\tau^{-1} z_\tau(t) \mathrm{d}\tau. \tag{9}$$

The power spectrum of y is then a mixture over the component spectra P_{z_τ}:

$$P_y(f) = \int_0^T 4\tau^{-2} P_{z_\tau}(f) \mathrm{d}\tau \tag{10}$$

$$= \int_0^T \frac{4\tau^{-2}}{\tau^{-2} + (2\pi f)^2} \mathrm{d}\tau \tag{11}$$

$$= \frac{2}{\pi f} \tan^{-1}(2\pi f T) \tag{12}$$

which is approximately $1/f$ for $f \gg 1/T$, i.e. for all but very low frequencies.

We next define a discrete approximation of the continuum mixture model in Equation 9,

$$\xi(t) = \sum_i z_i(t), \tag{13}$$

where z_i has timescale τ_i and scaling parameter σ_i. To approximate a $1/f$ spectrum, $\sigma_i^2(\tau_{i+1} - \tau_i)^{-1}$ should scale as $4\tau_i^{-2}$, the squared weight density in Equation 9 (because power is additive and proportional to σ^2). For example, the τ_i could be arithmetically spaced with $\sigma_i \propto \tau^{-1}$. Instead, we assume geometric spacing, with n components defined by $\tau_i = \nu^i$ and $\sigma_i = 2\rho\tau_i^{-1/2}$, where ρ^2 is a hyperparameter determining overall steady-state variance. Figure 3 illustrates this construction and exemplifies the accuracy of the discrete approximation.

To define a generative model of $1/f$ noise in $\mathbb{R}^m$, we assume an independent copy of this process for each dimension. The resulting joint process is rotationally symmetric, a property inherited from OU processes. That is, any linear combination $\sum_j c_j \xi_j$ is a 1d $1/f$ process.

B Kalman filter for a single timescale

Start again with a single OU process z with $\sigma = 1$, and assume it is observed at unit intervals $(y_t)_{t\in\mathbb{N}}$ with Gaussian observation noise of variance η^2. Assume a conjugate iterative prior:

$$z(t)|\boldsymbol{y}_{<t} \sim \mathcal{N}\left(w(t), s^2(t)\right). \tag{14}$$

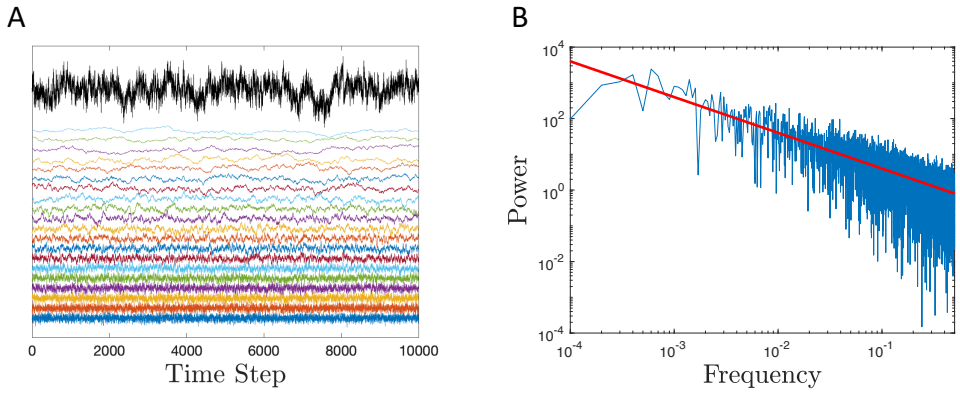

Figure 3: A: Construction of $1/f$ noise (black trajectory at top) as a sum of OU processes with different time constants (colored trajectories). Vertical offsets are applied as a visual aid for discriminating the curves. B: The power spectrum of the aggregate trajectory in log-log coordinates. The red line has slope 1.

The posterior after each observation is

$$z(t)|\boldsymbol{y}_{\leq t} \sim \mathcal{N}\left(\frac{\eta^2 w(t) + s^2(t)y(t)}{s^2(t)+\eta^2}, \frac{s^2(t)\eta^2}{s^2(t)+\eta^2}\right). \tag{15}$$

Evolving the process to time $t+1$ amounts to decay by $e^{-1/\tau}$ and accumulation of new noise. At each time $t' \in [t, t+1]$, variance from the noise appearing at time t' (i.e., from $\mathrm{d}W(t')$ in Equation 1) decays by a factor $e^{-2(t+1-t')/\tau}$ by time $t+1$. Therefore the total accumulated variance is:

$$\int_t^{t+1} e^{-2(t+1-t')/\tau}\mathrm{d}t' = \frac{\tau}{2}\left(1 - e^{-2/\tau}\right). \tag{16}$$

Therefore the prior for the next observation is

$$z(t+1)|\boldsymbol{y}_{\leq t} \sim \mathcal{N}\left(e^{-1/\tau}\frac{\eta^2 w(t) + s^2(t)y(t)}{s^2(t)+\eta^2}, e^{-2/\tau}\frac{s^2(t)\eta^2}{s^2(t)+\eta^2} + \frac{\tau}{2}\left(1 - e^{-2/\tau}\right)\right) \tag{17}$$

$$= \mathcal{N}\left(w(t+1), s^2(t+1)\right). \tag{18}$$

We thus obtain the recursion

$$w(t+1) = e^{-1/\tau}\frac{\eta^2 w(t) + s^2(t)y(t)}{s^2(t)+\eta^2} \tag{19}$$

$$= e^{-1/\tau}\left(w(t) + \alpha(t)(y(t) - w(t))\right). \tag{20}$$

This is a temporal-difference learning rule, or gradient descent on squared error $\frac{1}{2}(y(t) - w(t))$, with learning rate $\alpha(t) = \frac{s^2(t)}{s^2(t)+\eta^2}$. This connection between temporal-difference learning and the Kalman filter is well known [Sut92].

The steady state for s^2 is given by the solution to $s^2(t) = s^2(t+1)$, a quadratic with one positive root. If $s^2(0)$ is initialized to this value, then $\alpha(t)$ will be constant. Note that this algorithm (and the extension to $1/f$ noise in Appendix C) generalizes to irregularly spaced observations, in which case one can derive how the optimal learning rate should vary on each step.

C Kalman filter over $1/f$ noise

Assume we receive a sequence of observations at unit time intervals, $y(t)$ for $t \in \mathbb{N}$, and we want to do online prediction under the assumption that y is a $1/f$ noise process. Then we can make the generative assumption $y(t) = \xi(t)$ with ξ defined as at the end of Appendix A. For simplicity and in contrast to Appendix B, we assume no observation noise, because the shortest timescales (z_1, etc.) already play this role.

Because $y(t)$ is conditionally independent of the history given (and indeed is fully determined by) the joint state $\boldsymbol{Z}(t) = (z_1(t), \ldots, z_n(t))$, it suffices to compute the posterior for the latter. Write the iterative prior for $\boldsymbol{Z}$ as

$$\boldsymbol{Z}(t)|\boldsymbol{y}_{<t} \sim \mathcal{N}\left(\boldsymbol{\omega}(t), \boldsymbol{S}(t)\right), \tag{21}$$

implying an optimal (maximum-likelihood or least-squares) prediction of $\hat{y}(t) = \sum_i \omega_i(t)$. The posterior after observing $y(t)$ is the intersection of the prior with the hyperplane $\sum_i z_i(t) = y(t)$:

$$\boldsymbol{Z}(t)|\boldsymbol{y}_{\leq t} \sim \mathcal{N}\left(\boldsymbol{\omega}(t) + \frac{\boldsymbol{S}(t)\mathbf{1}}{\mathbf{1}^\top \boldsymbol{S}(t)\mathbf{1}}(y(t) - \hat{y}(t)), \left(\boldsymbol{P}\boldsymbol{S}(t)^{-1}\boldsymbol{P} + \mathbf{1}\mathbf{1}^\top\right)^{-1}\boldsymbol{P}\right). \tag{22}$$

Here, $\mathbf{1}$ is the vector with all elements equal to 1, and $\boldsymbol{P} = \boldsymbol{I} - \frac{1}{n}\mathbf{1}\mathbf{1}^\top$ is the orthogonal projector.

The prior for the next timestep is then obtained by applying decay and adding variance from the noise accumulated over the intervening interval. Define $\boldsymbol{D}$ as the diagonal matrix of decay factors, e^{-1/τ_i}, and $\boldsymbol{N}$ as the diagonal matrix of added noise $2\rho^2(1 - e^{-2/\tau_i})$, obtained by multiplying the RHS of Equation 16 by $\sigma_i^2 = 4\rho^2\tau_i^{-1}$ (from Appendix A). Then we have the update equations for the exact Bayesian model:

$$\boldsymbol{\omega}(t+1) = \boldsymbol{D}\left(\boldsymbol{\omega}(t) + \frac{\boldsymbol{S}(t)\mathbf{1}}{\mathbf{1}^\top \boldsymbol{S}(t)\mathbf{1}}(y(t) - \hat{y}(t))\right) \tag{23}$$

and

$$\boldsymbol{S}(t+1) = \boldsymbol{D}\left(\boldsymbol{P}\boldsymbol{S}(t)^{-1}\boldsymbol{P} + \mathbf{1}\mathbf{1}^\top\right)^{-1}\boldsymbol{P}\boldsymbol{D} + \boldsymbol{N}. \tag{24}$$

Next, consider the perceptron in Section 2, where $y = \boldsymbol{x}^\top\boldsymbol{\theta}$. The $1/f$ generative model assumes $\theta_j = \sum_i z_{ij}$ for each j, where i indexes timescales and j indexes features, and z_{ij} has timescale τ_i and scaling parameter σ_i defined as above. The latent state is described by $\boldsymbol{Z} = (z_{ij})_{ij}$. We treat ij as a single composite index, so that $\boldsymbol{Z}$ is a vector. As above, write the iterative prior as

$$\boldsymbol{Z}(t)|\boldsymbol{x}_{<t}, \boldsymbol{y}_{<t} \sim \mathcal{N}\left(\boldsymbol{\omega}(t), \boldsymbol{S}(t)\right). \tag{25}$$

Let $\boldsymbol{X}$ be a multiplexed copy of the input $\boldsymbol{x}$, so that $\boldsymbol{X}_{ij} = \boldsymbol{x}_j$. Assuming square loss, the optimal prediction for $y(t)$ is $\hat{y}(t) = \boldsymbol{X}(t)^\top\boldsymbol{\omega}(t)$. The posterior after observing $y(t)$ is the intersection of the prior with the hyperplane $\boldsymbol{X}(t)^\top\boldsymbol{Z}(t) = y(t)$:

$$\begin{aligned}\boldsymbol{Z}(t)|\boldsymbol{x}_{\leq t}, \boldsymbol{y}_{\leq t} \sim \mathcal{N}\Bigg(&\boldsymbol{\omega}(t) + \frac{\boldsymbol{S}(t)\boldsymbol{X}(t)}{\boldsymbol{X}(t)^\top\boldsymbol{S}(t)\boldsymbol{X}(t)}(y(t) - \hat{y}(t)),\\ &\left(\boldsymbol{P}_{\boldsymbol{X}(t)}\boldsymbol{S}(t)^{-1}\boldsymbol{P}_{\boldsymbol{X}(t)} + \boldsymbol{X}(t)\boldsymbol{X}(t)^\top\right)^{-1}\boldsymbol{P}_{\boldsymbol{X}(t)}\Bigg),\end{aligned} \tag{26}$$

where $\boldsymbol{P}_{\boldsymbol{X}} = \boldsymbol{I} - \boldsymbol{X}\boldsymbol{X}^\top/\boldsymbol{X}^\top\boldsymbol{X}$ is the projector orthogonal to $\boldsymbol{X}$.

Generalizing the definitions above, let $\boldsymbol{D}$ and $\boldsymbol{N}$ be the diagonal matrices of decay factors and noise accumulation, $D_{ij,ij} = e^{-1/\tau_i}$ and $N_{ij,ij} = 2\rho^2(1 - e^{-2/\tau_i})$. Applying these to Equation 26 to obtain the prior for the next time step yields the update equations for the exact Bayesian model of the regression task:

$$\boldsymbol{\omega}(t+1) = \boldsymbol{D}\left(\boldsymbol{\omega}(t) + \frac{\boldsymbol{S}(t)\boldsymbol{X}(t)}{\boldsymbol{X}(t)^\top\boldsymbol{S}(t)\boldsymbol{X}(t)}(y(t) - \hat{y}(t))\right) \tag{27}$$

$$\boldsymbol{S}(t+1) = \boldsymbol{D}\left(\boldsymbol{P}_{\boldsymbol{X}(t)}\boldsymbol{S}^{-1}(t)\boldsymbol{P}_{\boldsymbol{X}(t)} + \boldsymbol{X}(t)\boldsymbol{X}(t)^\top\right)^{-1}\boldsymbol{P}_{\boldsymbol{X}(t)}\boldsymbol{D} + \boldsymbol{N}. \tag{28}$$

Equation 27 exemplifies how the variance matrix, $\boldsymbol{S}(t)$, can be thought of as defining a preconditioner of the gradient, $\boldsymbol{X}(t)(y(t) - \hat{y}(t))$.

D Extended Kalman filter

Given a nonlinear model $h(\boldsymbol{x}, \boldsymbol{\theta})$, the $1/f$ EKF posits that the optimal parameters follow $1/f$ dynamics according to $\boldsymbol{\theta} = \sum_{i=1}^n \boldsymbol{z}_i$, with expanded latent state $\boldsymbol{Z} = (\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n)^\top$. It is convenient

to introduce the expanded model $\tilde{h}$ defined by $\tilde{h}(\boldsymbol{x}, \boldsymbol{Z}) = h(\boldsymbol{x}, \sum_i \boldsymbol{z}_i)$. The $1/f$ EKF maintains an iterative prior over $\boldsymbol{Z}$ as in Equation 25 and updates that prior by linearizing $\tilde{h}$ about $\boldsymbol{Z} = \boldsymbol{\omega}(t)$:

$$\tilde{h}(\boldsymbol{x}(t), \boldsymbol{Z}) \approx \tilde{h}(\boldsymbol{x}(t), \boldsymbol{\omega}(t)) + J_{\tilde{h}}(\boldsymbol{Z} - \boldsymbol{\omega}(t)). \tag{29}$$

Here, $J_{\tilde{h}} = \frac{\partial \hat{\boldsymbol{y}}}{\partial \boldsymbol{Z}}$ is the Jacobian matrix of $\tilde{h}$, evaluated at $\boldsymbol{Z} = \boldsymbol{\omega}(t)$. Note that $J_{\tilde{h}}$ is just n copies of J_h.

Following [Oll18], we assume the observation y is governed by some exponential family $P(y|\boldsymbol{\eta}(\hat{\boldsymbol{y}}))$, with vector of sufficient statistics $\boldsymbol{T}(y)$. The model's output $\hat{\boldsymbol{y}} = \tilde{h}(\boldsymbol{x}, \boldsymbol{\omega})$ is taken to encode the predicted mean parameter of that family: $\hat{\boldsymbol{y}} = \mathbb{E}_{y \sim P(\cdot|\boldsymbol{\eta}(\hat{\boldsymbol{y}}))}[\boldsymbol{T}(y)]$ (this can be read as a definition of the mapping $\hat{\boldsymbol{y}} \mapsto \boldsymbol{\eta}(\hat{\boldsymbol{y}})$). It then approximates the conditional distribution of the sufficient statistics as a Gaussian,

$$p(\boldsymbol{T}(y)|\hat{\boldsymbol{y}}) \approx \mathcal{N}(\hat{\boldsymbol{y}}, \boldsymbol{R}_{\hat{\boldsymbol{y}}}), \tag{30}$$

where $\boldsymbol{R}_{\hat{\boldsymbol{y}}} = \mathrm{Var}(\boldsymbol{T}(y)|\hat{\boldsymbol{y}})$ is the conditional variance.

For example, when h is the 2-layer (perceptron+softmax) network in Section 3, the output of the network is a vector $\hat{\boldsymbol{y}}$ of class probabilities, and the sufficient statistics $\boldsymbol{T}(y)$ are a one-hot vector. For numerical stability, we exclude the final element of $\hat{\boldsymbol{y}}$ and $\boldsymbol{T}(y)$, which are determined by the other elements. The conditional outcome variance is given by

$$[\boldsymbol{R}_{\hat{\boldsymbol{y}}}]_{i,j} = \begin{cases} \hat{y}_i(1 - \hat{y}_i) & i = j \\ -\hat{y}_i \hat{y}_j & i \neq j. \end{cases} \tag{31}$$

Under the approximations of Equations 29 and 30, the posterior is given by the standard KF formula:

$$\begin{aligned} \boldsymbol{Z}(t)|\boldsymbol{x}_{\leq t}, \boldsymbol{y}_{\leq t} \sim \mathcal{N}\Big(&\boldsymbol{\omega}(t) + \boldsymbol{S}(t)\boldsymbol{J}_{\tilde{h}}^\top \left(\boldsymbol{J}_{\tilde{h}}\boldsymbol{S}(t)\boldsymbol{J}_{\tilde{h}}^\top + \boldsymbol{R}_{\hat{\boldsymbol{y}}(t)}\right)^{-1} (\boldsymbol{T}(y(t)) - \hat{\boldsymbol{y}}(t)), \\ &\boldsymbol{S}(t) - \boldsymbol{S}(t)\boldsymbol{J}_{\tilde{h}}^\top \left(\boldsymbol{J}_{\tilde{h}}\boldsymbol{S}(t)\boldsymbol{J}_{\tilde{h}}^\top + \boldsymbol{R}_{\hat{\boldsymbol{y}}(t)}\right)^{-1} \boldsymbol{J}_{\tilde{h}}\boldsymbol{S}(t)\Big). \end{aligned} \tag{32}$$

Applying decay ($\boldsymbol{D}$) and accumulated noise ($\boldsymbol{N}$) as in Appendix C to obtain the prior for the next time step yields the update equations for the $1/f$ EKF:

$$\boldsymbol{\omega}(t+1) = \boldsymbol{D}\left(\boldsymbol{\omega}(t) + \boldsymbol{S}(t)\boldsymbol{J}_{\tilde{h}}^\top \left(\boldsymbol{J}_{\tilde{h}}\boldsymbol{S}(t)\boldsymbol{J}_{\tilde{h}}^\top + \boldsymbol{R}_{\hat{\boldsymbol{y}}(t)}\right)^{-1} (\boldsymbol{T}(y(t)) - \hat{\boldsymbol{y}}(t))\right) \tag{33}$$

$$\boldsymbol{S}(t+1) = \boldsymbol{D}\left(\boldsymbol{S}(t) - \boldsymbol{S}(t)\boldsymbol{J}_{\tilde{h}}^\top \left(\boldsymbol{J}_{\tilde{h}}\boldsymbol{S}(t)\boldsymbol{J}_{\tilde{h}}^\top + \boldsymbol{R}_{\hat{\boldsymbol{y}}(t)}\right)^{-1} \boldsymbol{J}_{\tilde{h}}\boldsymbol{S}(t)\right)\boldsymbol{D} + \boldsymbol{N}. \tag{34}$$

E Variational approximation

This section derives variational approximations of the KF and EKF models, with $\boldsymbol{S}(t) \approx \tilde{\boldsymbol{S}}(t)$ where $\tilde{\boldsymbol{S}}(t) := \mathrm{diag}(\boldsymbol{s}^2(t))$. Thus the resulting algorithms need only track the individual variance terms in $\boldsymbol{s}(t)$ rather than the full covariance matrix. Moreover, the update equations (41,42,46,47,52,55) all avoid matrix inversion—even though this might initially seem necessary from Equation 35—a property that may be relevant for efficient scaling.

E.1 Uncued inference

We begin with the simple Bayesian $1/f$ model in Equations 23 and 24, where there are no predictors and the model merely tracks an observable $y(t)$. Given an arbitrary Gaussian distribution, the variational approximation (in the sense of minimizing Kullback-Leibler divergence) by another Gaussian with diagonal covariance matrix is obtained by taking the diagonal of the original distribution's precision matrix. Thus in the present case we have

$$s_i^{-2}(t+1) = \left[\boldsymbol{S}^{-1}(t+1)\right]_{i,i} \tag{35}$$

where $\boldsymbol{S}(t+1)$ is given by Equation 24 with $\boldsymbol{S}^{-1}(t)$ replaced by $\tilde{\boldsymbol{S}}^{-1}(t)$ (the inductive assumption). To calculate $\boldsymbol{S}^{-1}(t+1)$ under this assumption, we first observe the identity

$$\left(\boldsymbol{P}\,\mathrm{diag}\left(\boldsymbol{s}^{-2}\right)\boldsymbol{P} + \boldsymbol{1}\boldsymbol{1}^\top\right)^{-1}\boldsymbol{P} = \mathrm{diag}\left(\boldsymbol{s}^2\right) - \frac{\boldsymbol{s}^2\boldsymbol{s}^{2\top}}{\sum_i s_i^2}. \tag{36}$$

Therefore Equation 24 becomes

$$\boldsymbol{S}(t+1) = \boldsymbol{D}\tilde{\boldsymbol{S}}(t)\boldsymbol{D} + \boldsymbol{N} - \frac{(\boldsymbol{D}\boldsymbol{s}^2(t))(\boldsymbol{D}\boldsymbol{s}^2(t))^\top}{\sum_i s_i^2(t)} \tag{37}$$

$$:= \mathrm{diag}(\boldsymbol{a}) - \frac{\boldsymbol{b}\boldsymbol{b}^\top}{c} \tag{38}$$

where $\mathrm{diag}(\boldsymbol{a}) = \boldsymbol{D}\tilde{\boldsymbol{S}}(t)\boldsymbol{D} + \boldsymbol{N}$, $\boldsymbol{b} = \boldsymbol{D}\boldsymbol{s}^2(t)$, and $c = \sum_i s_i^2(t)$. We next use the identity

$$\left(\mathrm{diag}\left(\boldsymbol{a}\right) - \frac{\boldsymbol{b}\boldsymbol{b}^\top}{c}\right)^{-1} = \mathrm{diag}\left(\frac{1}{\sqrt{\boldsymbol{a}}}\right)\left(\boldsymbol{I} + \frac{\left(\boldsymbol{a}^{-1/2}\circ\boldsymbol{b}\right)\left(\boldsymbol{a}^{-1/2}\circ\boldsymbol{b}\right)^\top}{c - \sum \frac{b_i^2}{a_i}}\right)\mathrm{diag}\left(\frac{1}{\sqrt{\boldsymbol{a}}}\right), \tag{39}$$

implying the diagonal elements are

$$\left[\left(\mathrm{diag}(\boldsymbol{a}) - \frac{\boldsymbol{b}\boldsymbol{b}^\top}{c}\right)^{-1}\right]_{i,i} = \frac{1}{a_i}\left(1 + \frac{\frac{b_i^2}{a_i}}{c - \sum_{i'}\frac{b_{i'}^2}{a_{i'}}}\right). \tag{40}$$

Combining Equations 35, 38, and 40 gives the final form of the variational update:

$$s_i^2(t+1) = \frac{\left(s_i^2(t)e^{-1/\tau_i} + 4\rho^2\sinh\frac{1}{\tau_i}\right)^2\Omega}{\left(s_i^2(t) + 4\rho^2 e^{1/\tau_i}\sinh\frac{1}{\tau_i}\right)\Omega + s_i^4(t)} \tag{41}$$

with

$$\Omega = \sum_i \frac{4\rho^2 s_i^2(t)\sinh\frac{1}{\tau_i}}{e^{-1/\tau_i}s_i^2(t) + 4\rho^2\sinh\frac{1}{\tau_i}}. \tag{42}$$

This update of the variance converges exponentially to a unique fixed point. Numerical simulations confirm that, in this limit, s_i^2 is larger for smaller τ_i, meaning faster learning rates for shorter timescales. If the prior variances are initialized at the fixed point then they are constant throughout learning.

By substituting the diagonal matrix $\tilde{\boldsymbol{S}}(t)$ for $\boldsymbol{S}(t)$, the update for the mean in Equation 23 simplifies to

$$\omega_i(t+1) = e^{-1/\tau_i}\omega_i(t) - \alpha_i(t)\left(\hat{y}(t) - y(t)\right) \tag{43}$$

where $\hat{y}(t) - y(t)$ is the loss gradient (assuming square loss), and the learning rates are given by

$$\alpha_i(t) = \frac{e^{-1/\tau_i}s_i^2(t)}{\sum_{i'} s_{i'}^2(t)}. \tag{44}$$

That is, the subweights learn independently according to their gradients, with different decay rates and learning rates.

E.2 Regression model

To derive the variational model for the regression task, we apply the analysis of Section E.1 to the KF update in Equations 27 and 28. Paralleling the derivation of Equation 37, the variance update in Equation 28 (i.e., before applying the diagonal variational approximation) can be written as

$$\boldsymbol{S}(t+1) = \boldsymbol{D}\tilde{\boldsymbol{S}}(t)\boldsymbol{D} + \boldsymbol{N} - \frac{(\boldsymbol{D}(\boldsymbol{x}(t)\circ\boldsymbol{s}^2(t)))(\boldsymbol{D}(\boldsymbol{x}(t)\circ\boldsymbol{s}^2(t)))^\top}{\sum_{ij} x_{ij}^2(t)s_{ij}^2(t)}. \tag{45}$$

Paralleling the derivation of Equations 41 and 42, the variational update comes out to be

$$s_{ij}^2(t+1) = \frac{\left(s_{ij}^2(t)e^{-1/\tau_i} + 4\rho^2\sinh\frac{1}{\tau_i}\right)^2\Omega}{\left(s_{ij}^2(t) + 4\rho^2 e^{1/\tau_i}\sinh\frac{1}{\tau_i}\right)\Omega + x_j^2(t)s_{ij}^4(t)} \tag{46}$$

with

$$\Omega = \sum_{ij} \frac{4\rho^2 x_j^2(t) s_{ij}^2(t) \sinh \frac{1}{\tau_i}}{e^{-1/\tau_i} s_{ij}^2(t) + 4\rho^2 \sinh \frac{1}{\tau_i}}. \tag{47}$$

By substituting the diagonal matrix $\tilde{\boldsymbol{S}}(t)$ for $\boldsymbol{S}(t)$, the mean update in Equation 27 simplifies to

$$\omega_{ij}(t+1) = e^{-1/\tau_i} \omega_{ij}(t) - \alpha_{ij} x_j(t) \left(\hat{y}(t) - y(t)\right) \tag{48}$$

where $x_j(t) \left(\hat{y}(t) - y(t)\right)$ is the loss gradient for ω_{ij}, and the learning rates are given by

$$\alpha_{ij} = \frac{e^{-1/\tau_i} s_{ij}^2(t)}{\sum_{i'j'} x_{j'}^2(t) s_{i'j'}^2(t)}. \tag{49}$$

Thus, as above, the subweights learn independently according to their gradients, with different decay rates and learning rates.

E.3 Classification model

To derive a closed-form variational approximation for the general EKF, such as for the classification model in Section 3, it turns out that we need to apply the variational approximation to the posterior in Equation 32, rather than to the iterative prior in Equation 34. Using Woodbury's identity, the posterior variance can be rewritten as

$$\boldsymbol{S}(t) - \boldsymbol{S}(t) \boldsymbol{J}_{\tilde{h}}^\top \left(\boldsymbol{J}_{\tilde{h}} \boldsymbol{S}(t) \boldsymbol{J}_{\tilde{h}}^\top + \boldsymbol{R}_{\hat{\boldsymbol{y}}(t)} \right)^{-1} \boldsymbol{J}_{\tilde{h}} \boldsymbol{S}(t) = (\boldsymbol{J}_{\tilde{h}}^\top \boldsymbol{R}_{\hat{\boldsymbol{y}}(t)}^{-1} \boldsymbol{J}_{\tilde{h}} + \boldsymbol{S}^{-1}(t))^{-1}. \tag{50}$$

The form on the RHS is convenient because it is in terms of precision, allowing us to read off the diagonal entries directly. That is, the variational approximation for the posterior variance is $\mathrm{diag}(\boldsymbol{s}'(t)^2)$, with

$$s_{ij}'^2(t) = \left(\left[\boldsymbol{J}_{\tilde{h}}^\top \boldsymbol{R}_{\hat{y}(t)}^{-1} \boldsymbol{J}_{\tilde{h}} \right]_{ij,ij} + s_{ij}^{-2}(t) \right)^{-1}. \tag{51}$$

Here we have used the inductive assumption $\boldsymbol{S}(t) \approx \mathrm{diag}(\boldsymbol{s}(t)^2)$. Applying the transition from posterior on step t to prior on step $t+1$, we obtain the variance update for the variational model, replacing Equation 34:

$$s_{ij}^2(t+1) = D_{ij,ij}^2 s_{ij}'^2(t) + N_{ij,ij}. \tag{52}$$

Woodbury's identity also enables the mean update from Equation 33 to be rewritten, as

$$\boldsymbol{\omega}(t+1) = \boldsymbol{D} \left(\boldsymbol{J}_{\tilde{h}}^\top \boldsymbol{R}_{\hat{\boldsymbol{y}}(t)}^{-1} \boldsymbol{J}_{\tilde{h}} + \boldsymbol{S}^{-1}(t) \right)^{-1} \left(\boldsymbol{S}^{-1}(t) \boldsymbol{\omega}(t) + \boldsymbol{J}_{\tilde{h}}^\top \boldsymbol{R}_{\hat{\boldsymbol{y}}(t)}^{-1} \left(\boldsymbol{T}(y(t)) - \hat{\boldsymbol{y}}(t) + \boldsymbol{J}_{\tilde{h}} \boldsymbol{\omega}(t) \right) \right). \tag{53}$$

Substituting the gradient of the EKF's approximate likelihood (denoted $\tilde{\mathcal{L}}$) from Equation 30,

$$\partial_{\boldsymbol{\omega}(t)} \tilde{\mathcal{L}} = \boldsymbol{J}_{\tilde{h}}^\top \boldsymbol{R}_{\hat{\boldsymbol{y}}(t)}^{-1} (\hat{\boldsymbol{y}}(t) - \boldsymbol{T}(y(t))), \tag{54}$$

yields

$$\boldsymbol{\omega}(t+1) = \boldsymbol{D} \left(\boldsymbol{\omega}(t) - \left(\boldsymbol{J}_{\tilde{h}}^\top \boldsymbol{R}_{\hat{\boldsymbol{y}}(t)}^{-1} \boldsymbol{J}_{\tilde{h}} + \boldsymbol{S}^{-1}(t) \right)^{-1} \partial_{\boldsymbol{\omega}(t)} \tilde{\mathcal{L}} \right). \tag{55}$$

The preconditioner on the gradient here is the posterior variance (see Equation 50), which we have approximated as $\mathrm{diag}(\boldsymbol{s}'^2(t))$. Although not necessarily entailed by the variational approximation, we can consider applying the same approximation in updating the mean. This yields

$$\omega_{ij}(t+1) = e^{-1/\tau_i} \omega_{ij}(t) - e^{-1/\tau_i} s_{ij}'^2(t) \partial_{w_j(t)} \tilde{\mathcal{L}}. \tag{56}$$

Thus, once again, the subweights learn independently according to their gradients, with different decay rates and learning rates.

F Implementation details

Latent parameters defining the task environments were sampled using the generative model in Appendix A (i.e., matching the Bayesian model's assumptions), using 20 timescales geometrically spaced from $\tau_1 = 1$ to $\tau_{20} = 1000$ (as in Figure 3A). The regression task was run for 10,000 trials, and the classification task for 1000 trials. Each component OU process was run for $10\tau_i$ burn-in steps. The $1/f$ power spectrum was confirmed with a log-log plot (see Figure 3B).

In the regression task, the first feature was a constant bias term, $x_1 \equiv 1$. The other 9 features were sampled as $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$ on each time step.

For the classification task, class logits were sampled from $1/f$ processes and then multiplied by 0.886 before entering into softmax to determine class probabilities. Feature-class means were fixed at 1 for the first feature (i.e., bias term) and were sampled from mutually independent $1/f$ processes for features 2-10, multiplied by 0.224. These scaling factors were chosen so that perfect knowledge of either the prior probabilities or the feature-class means on every trial would yield ideal-observer performance of $\bar{\mathcal{L}} \approx 1$. Perfect knowledge of both would yield $\bar{\mathcal{L}} \approx 0.35$. These were merely guidelines for equating prior and likelihood information, as perfect knowledge of either source of information is not possible even with an optimal model of the dynamics.

All Bayesian and variational models assumed 10 timescales, geometrically spaced from $\tau_1 = 2$ to $\tau_{10} = 800$. This deliberate deviation from the generating process enabled testing of robustness, specifically the hypothesis that it is the aggregate $1/f$ character of the environment and of the model that matters, not the choice of component timescales used to approximate that character.