A dynamic scale-mixture model of motion in natural scenes

Jared M. Salisbury and Stephanie E. Palmer
Department of Organismal Biology and Anatomy & Department of Physics,
University of Chicago, Chicago, Illinois 60637, USA
(Dated: October 19, 2023)

Some of the most important tasks of visual and motor systems involve estimating the motion of objects and tracking them over time. Such systems evolved to meet the behavioral needs of the organism in its natural environment, and may therefore be adapted to the statistics of motion it is likely to encounter. By tracking the movement of individual points in videos of natural scenes, we begin to identify common properties of natural motion across scenes. As expected, objects in natural scenes move in a persistent fashion, with velocity correlations lasting hundreds of milliseconds. More subtly, we find that the observed velocity distributions are heavy-tailed and can be modeled as a Gaussian scale-mixture. Extending this model to the time domain leads to a dynamic scale-mixture model, consisting of a Gaussian process multiplied by a positive scalar quantity with its own independent dynamics. Dynamic scaling of velocity arises naturally as a consequence of changes in object distance from the observer, and may approximate the effects of changes in other parameters governing the motion in a given scene. This modeling and estimation framework has implications for the neurobiology of sensory and motor systems, which need to cope with these fluctuations in scale in order to represent motion efficiently and drive fast and accurate tracking behavior.

INTRODUCTION

One of the great triumphs of theoretical neuroscience has been the success of the efficient coding hypothesis [1], which posits that sensory neural systems are adapted to the statistics of the organism's natural environment. The importance of this hypothesis lies in its power to explain structural features of the nervous system, such as the shapes of nonlinear response functions [2] and receptive fields of sensory neurons [3–7], in terms of its function as an efficient information processing device. The success of this theory, particularly in vision, has sparked significant interest in measuring natural scene statistics (for a review, see [8]), and has yielded important results, like the ubiquity of non-Gaussian, heavy-tailed statistics and related nonlinear forms of dependency among scene features [9–12].

Beyond processing sensory stimuli, organisms must also produce adaptive behavior in a complex and dynamic natural environment, where tasks like capturing prey [13–16], fleeing predators [17–19], and navigating obstacles [20] are all critical to survival. These behaviors inevitably involve prediction [21–24] in order to compensate for substantial sensory and motor delays [25]. The basis for such predictive behavior must be statistical regularities in the environment, but little is known about the statistics of the inputs relevant to such behaviors.

As a step towards characterizing the statistics of behaviorally relevant quantities in natural scenes, we focus on one of the most important features therein: the motion of objects. Object motion relative to the observer drives oculomotor tracking [26, 27] and is an essential part of many crucial behaviors, like prey capture. Specialized circuitry as early as the retina distinguishes between object and background motion [28, 29], while entire brain regions in the visual cortex of primates specialize in processing motion [30] with increasing complexity along the dorsal stream [31].

While previous work has characterized motion in certain cases, often focusing on visual motion due to egomotion [20, 32, 33], little is known about the statistics of object trajectories in the natural world. To address this, we analyze videos from the Chicago Motion Database[34], which were shot and curated for the purposes of statistical analysis and for use as stimuli for neural recordings and visual psychophysics. Rather than trying to track discrete objects (which may be difficult even to define for some videos, like those of flowing water), we simplify the problem by tracking individual points using classic techniques from computer vision [35, 36].

Given a point trajectory, the velocity along that trajectory describes the object's local motion. We find that point velocity is highly correlated on the sub-second timescale we measured, and therefore point trajectories are highly predictable. More subtly, the velocity distributions we observe are jointly heavy-tailed, both across horizontal and vertical components and across time. This suggests the presence of an underlying *scale* variable, or local standard deviation, which allows the local velocity to be modeled as a Gaussian scale-mixture [37], as has been suggested previously in other contexts [10, 38]. We find that the scale fluctuates within individual trajectories on a relatively short timescale, so it is an essential part of our description of natural motion. Despite considerable differences in the velocity statistics across movies, the dynamic scale-mixture structure is remarkably consistent. This has important implications both for the efficient encoding of motion signals by neurons–which must adapt to the to the fluctuating scale to make full use of their limited dynamic range [39, 40]–and for behaviors relying on predictive tracking–which must take into account the highly non-Gaussian statistics of natural motion [41].

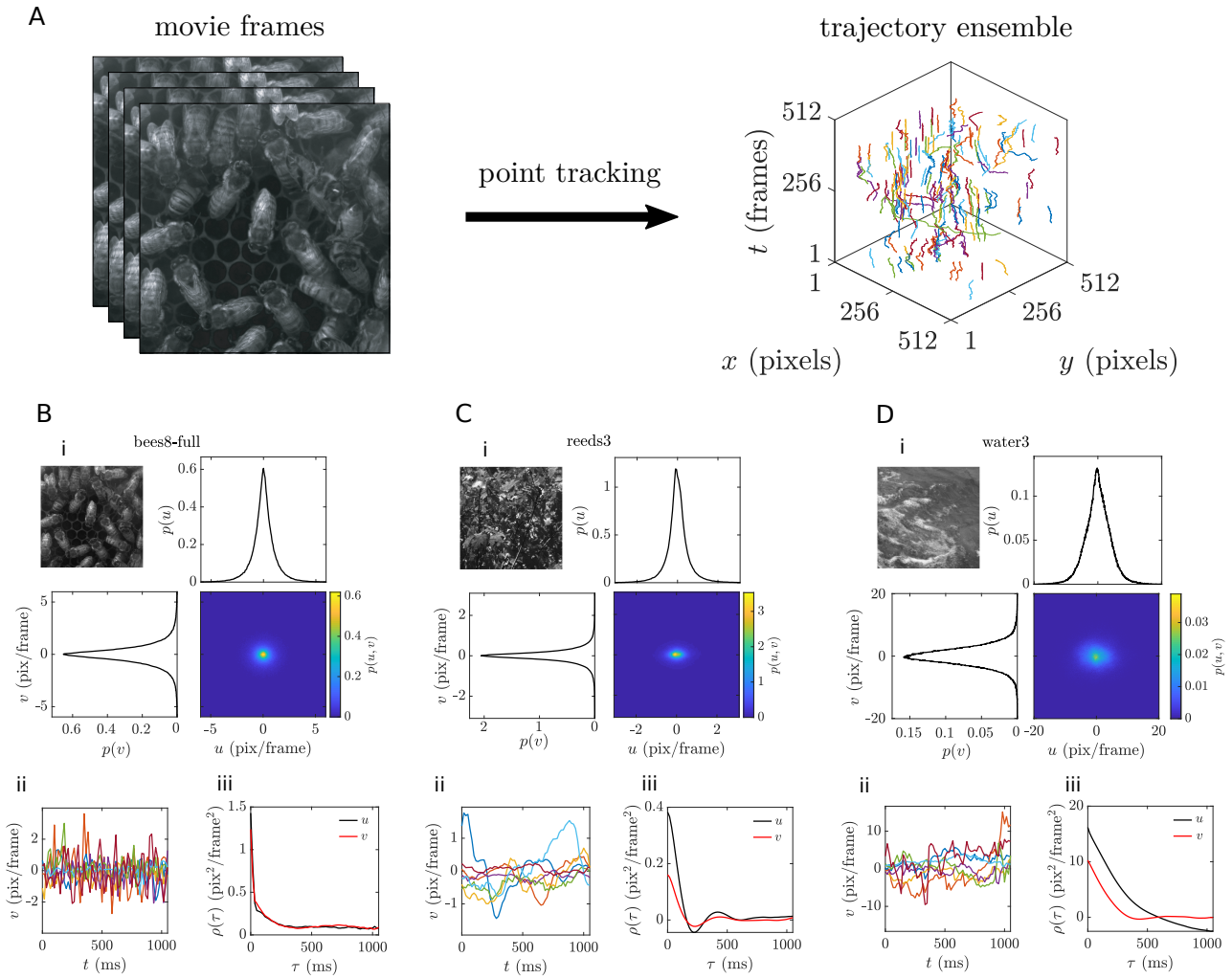

FIG. 1: **Automated point tracking reveals a diversity of motion statistics across natural scenes. A.** Natural video data analyzed via point tracking yields an ensemble of 1 s long point trajectories. **B-D.** Raw data summaries for three example movies, (**B**) `bees8-full`, (**C**) `trees14-1`, and (**D**) `water3`. **i.** Joint and marginal distributions for horizontal (u) and vertical (v) velocity components. **ii.** Seven example horizontal velocity component time series. **iii.** Horizontal and vertical velocity correlation functions.

RESULTS

In order to build up a statistical description of motion in natural scenes, we analyze natural movies from the Chicago Motion Database, which consists of a variety of natural scenes collected for this statistical analysis and for use as visual stimuli in experiments. All movies were recorded using a fixed camera, with scenes chosen to contain consistent motion within the field of view for minutes at a time. Scenes include flowing water, plants moving in the wind, and groups of animals such as insects and fish. We analyze a total of 15 scenes, with a resolution of 512×512 pixels, each $2^{14} = 16,384$ frames long at a frame rate of 60 Hz (~ 4.5 minutes).

From these natural movies, we quantify local motion within each scene using a standard point tracking algorithm [35, 36]. A set of tracking points are seeded randomly on each frame, then tracked both forward and backward in time to generate trajectories (see *Materials and Methods* for details). Early visual and sensorimotor systems operate on a timescale of tens to hundreds of milliseconds, so we restrict our analysis to short trajectories (64 frames, or ~ 1 s long) to reduce the amount of inevitable slippage from the point tracking algorithm. The resulting ensembles ($2^{13} = 8,192$ trajectories each) sparsely cover most of the moving objects in each movie (Figure 1A).

The focus of our analysis is the point velocity, or difference in position between subsequent frames, measured in raw units of pixels/frame (this is easily converted to

degrees of visual angle per unit of time, given a fixed viewing distance). The key advantage of this analysis is that the velocities are associated in time along a given point trajectory, which cannot be achieved by looking at the optical flow [42] or motion energy [43] alone. This analysis is also distinct from previous work examining the spatiotemporal power spectra of natural scenes [44], since power spectra measure only pairwise correlations between pixels, while the object motion involves coordinated changes among many pixels.

A motivating example: Brownian motion

Before presenting the results of our analysis, it will be useful to introduce a canonical example of natural motion to guide our intuition about departures from the simplest case: Brownian motion. Brown discovered the motion that bears his name by observing small particles released by a grain of pollen floating in water under a microscope [45]—a highly controlled setting, but similar in spirit to our own. Each particle is subjected to tiny molecular forces from individual water molecules colliding with it at random. As a result, it moves in a random fashion across the surface of the water, limited only by the boundary of the dish. Its motion is not completely unpredictable, however; since the particle has some mass, albeit small, it has some inertia, or tendency to continue moving with the same velocity. Statistically, this means the velocity is correlated in time.

Based on this observation, Einstein derived his famous diffusion equation, describing how a density of diffusing particles changes over time, by considering the case of particles with infinitesimal mass [46]. For describing the motion of an individual, massive particle, it is more useful to look to Langevin's description, an early application of stochastic differential equations [47]. Using Newton's law and assuming the force can be divided into the sum of a drag force, proportional to the velocity by a factor of $-\gamma$, and a randomly fluctuating force due to the collision of water molecules, $F(t)$, we have the following differential equation for the velocity, $V(t)$, of a particle with mass m:

$$m\frac{d}{dt}V(t) = -\gamma V(t) + F(t)\,.$$

When $F(t)$ is an uncorrelated Gaussian process, $V(t)$ is an Ornstein-Uhlenbeck process, which has an exponential correlation function (see [48, 49] for a pedagogical review). The variance of the velocity, σ^2, is related to the temperature, T, by $\sigma^2 = kT/m$, where k is Boltzmann's constant.

Numerically and experimentally, we must always discretize time. The discrete-time approximation to the Ornstein-Uhlenbeck process with time-step Δt is given by the difference equation

$$V_{t+\Delta t} = \phi V_t + \xi_t\,,$$
$$p(\xi_t) = \mathcal{N}(\xi_t; 0, \sigma_\xi^2)\,,$$

which is a first-order autoregressive, or $AR(1)$, process [50, 51]. Like the Ornstein-Uhlenbeck process, it has an exponential autocovariance function, given by

$$E\left[V_t V_{t+k\Delta t}\right] = \sigma^2 \exp\left(-\frac{k\Delta t}{\tau}\right)\,,$$

with variance

$$\sigma^2 = \frac{\sigma_\xi^2}{1-\phi^2}$$

and time constant

$$\tau = -\frac{\Delta t}{\ln\phi}\,.$$

In anticipation of our results, let us consider how to incorporate a fluctuating variance in the $AR(1)$ process. There are two possibilities: the variance of the driving noise ξ_t could fluctuate (as in Brownian motion with a fluctuating temperature), or the entire process could be scaled by a fluctuating positive variable (such as inverse viewing distance, if the velocity is an angular velocity relative to an observer). The former corresponds to

$$\bar{V}_{t+\Delta t} = \phi\bar{V}_t + S_t\xi_t\,;$$

the latter, to

$$\tilde{V}_t = S_t V_t\,,$$

where S_t is the fluctuating scale.

Rewriting

$$\tilde{V}_{t+\Delta t} = \frac{S_{t+\Delta t}}{S_t}\phi\tilde{V}_t + S_{t+\Delta t}\xi_t\,,$$

we see that $\bar{V}_t$ and $\tilde{V}_t$ are identical when S_t is constant. That is, scaling the driving noise by a random positive constant and scaling the entire process by the same are equivalent ways to generate an ensemble of time series where the variance fluctuates across realizations. However, they are not equivalent when S_t changes over time. In the following, we will adopt the multiplicative model, since it factors into a product of two independent stochastic processes (one for normalized velocity and one for scale), which simplifies the inference problem. This factorization strategy may have important ramifications for the neural processing of natural motion, as we will see.

Heavy-tailed statistics of natural motion

Natural scenes are, by definition, as richly varied as the natural world; each movie we analyze captures a small

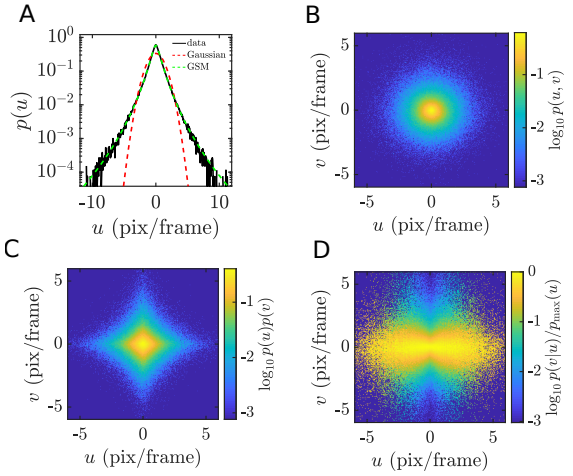

FIG. 2: **Velocity distributions are jointly heavy-tailed. A-D.** Velocity distributions for a single example video, `bees8-full`. The marginal distribution for horizontal velocity (**A**) has much heavier tails than a Gaussian with the same variance, and is well fit by a Gaussian scale-mixture model. The joint velocity distribution (**B**) is roughly radially symmetric, which differs substantially from the shuffled distribution (**C**) and indicates a nonlinear dependence between the two velocity components. This dependence is alternatively revealed by the conditional distribution of the vertical velocity given the horizontal velocity (**D**), showing a characteristic bow-tie shape.

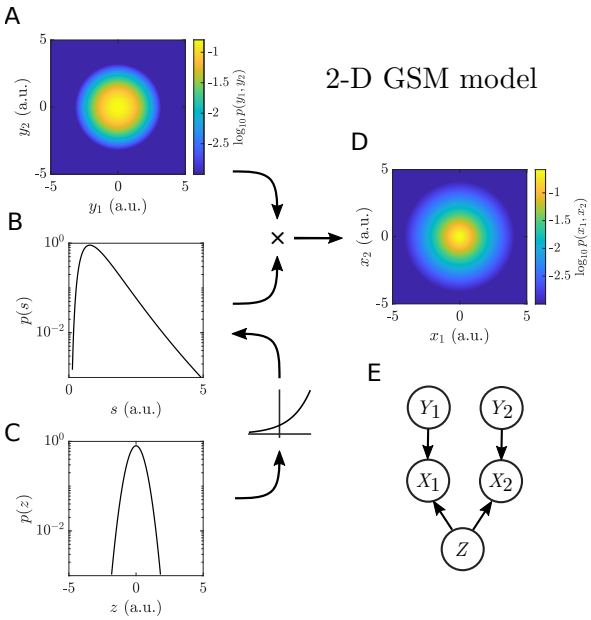

FIG. 3: **Schematic of the two-dimensional Gaussian scale-mixture model. A-D.** A Gaussian random variable Z (**C**) is passed through an exponential nonlinearity to yield a log-normal scale variable S (**B**). The scale multiplies both components of an underlying Gaussian distribution (**A**) to produce radially symmetric heavy tails (**D**). For the joint distributions, probabilities less than 10^{-3} were set to zero to facilitate comparison with empirical histograms. **E.** Dependency graph for the variables in the model.

slice of this immense diversity. Our selection can be divided into three broad categories–animals, plants (animated by wind), and water–and we present summaries of the raw data for a representative movie from each category in Figure 1B-D. In contrast to the Gaussian velocity distributions expected for Brownian motion, histograms of the raw velocity data tend to be sharply peaked with long tails. The velocity time series exhibit correlation functions with diverse shapes, rather than the simple exponential decay of the $AR(1)$ process described above.

To examine the heavy-tailed structure of the observed point-trajectory velocity distributions, we pool horizontal and vertical velocity components together for a single movie, `bees8-full`, and compare this histogram to a Gaussian distribution with the same variance (Figure 2A). Plotted on a log scale to emphasize the tail behavior, the empirical frequency falls off nearly linearly away from zero, while the Gaussian probability falls off quadratically. The same is true for the other movies in our dataset, pooled by category and all together (Figure 4A-C). Velocity distributions from animal and plant videos tend to have heavier tails, while those of water videos are closer to Gaussian.

Velocity components share a common scale variable

When multiple variables are involved, heavy tails may be associated with a nonlinear form of dependency, as observed in the spatial structure of natural images [38]. The same is true for the two velocity components in our data. We illustrate this for `bees8-full`, but results are similar for all other movies. In Figure 2B we show a heat map of the joint histogram of horizontal and vertical velocity, u and v. It is nearly radially symmetric. (For other movies with unequal variance in the two components, distributions are elliptic.) When we shuffle the data to break any association between u and v, the resulting histogram is no longer radially symmetric but is instead diamond-shaped (Figure 2C). In fact, the product of two marginal heavy-tailed distributions can never be radially symmetric or, more generally, elliptic, in contrast to the product of two marginal Gaussian distributions. We demonstrate this dependency more clearly by plotting the conditional distribution of v for each value of u, normalizing by the peak value at each u for visualization purposes (Figure 2D). The resulting "bow-tie" shape indicates that the variance of v conditioned on u

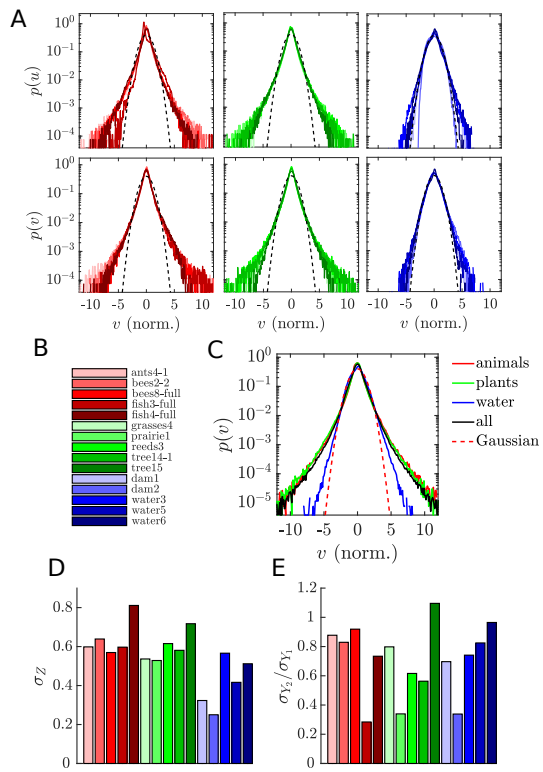

FIG. 4: **Quantifying heavy tails across scenes and categories A.** Marginal distributions for vertical and horizontal velocity components, grouped by category. **B.** Legend of individual movie names for **A** and all subsequent plots. **C.** Marginal distributions for the combined data across categories. Each velocity component of each movie was normalized by its standard deviation before combining. **D.** Estimated standard deviations for the scale generator variable, Z, varied across movies, corresponding to different amounts of kurtosis. **E.** The ratio of estimated standard deviations of the underlying Gaussian variables, Y_1 and Y_2, showing the the degree of anisotropy.

increases with the magnitude of u.

The form of the velocity distributions observed above suggests that they can be modeled as Gaussian scale-mixture (GSM) distributions. As the name suggests, a GSM distribution is obtained by combining (zero-mean) Gaussian distributions of different scales, parameterized by a positive scalar random variable S. Let Y be a Gaussian random variable with mean zero and variance σ_Y^2. It has distribution

$$p(y) = \mathcal{N}(y; 0, \sigma_Y^2)\,.$$

If S is a known quantity s, then $X = Ys$ is simply a Gaussian random variable with zero mean and variance $s^2\sigma_Y^2$. The conditional distribution is given by

$$p(x|s) = \mathcal{N}(x; 0, s^2\sigma_Y^2)\,.$$

When S is unknown, $X = YS$ follows a GSM distribution given by

$$p(x) = \int_0^\infty p(x|s)p(s)ds,$$

where $p(s)$ is a distribution with positive support. A convenient choice is to let $S = \exp(Z)$, where Z is Gaussian random variable, which we will refer to as the scale generator, with mean zero and variance σ_Z^2. Then S follows a log-normal distribution, which simplifies the inference problem significantly, despite the fact that the resulting GSM distribution does not have a closed form. See [38, 52] for a discussion of the GSM model in the context of wavelet analysis of natural images. Paremeters were estimated using a variant of the Expectation-Maximization (EM) algorithm [53] (see *Materials and Methods*).

For an individual velocity component as in Figure 2A, the GSM model captures the shape of the distribution well, with only two parameters: σ_Y, controlling the overall scale, and σ_Z, controlling the heaviness of the tail. To model the joint distribution, as in Figure 2B, clearly we cannot use independent GSM models for each component, since this corresponds to the shuffled distribution in Figure 2C. Instead, we consider a model in which two independent Gaussian random variables, Y_1 and Y_2, are multiplied by a shared scale variable S:

$$X_1 = Y_1 S\,,$$
$$X_2 = Y_2 S\,.$$

Note that we will maintain the general notation for the model for clarity. Applied to the velocity data, we have

$$(X_1, X_2) = (U, V)\,,$$

and (Y_1, Y_2) are the corresponding scale-normalized velocity components. The model is depicted in Figure 3, and it captures the radially symmetric (or more generally, when the variances are not equal, elliptic) shape of the joint distribution. This model has only three parameters: σ_{Y_1}, σ_{Y_2}, and σ_Z. A more complicated model could have two correlated scale variables with different standard deviations, σ_{Z_1} and σ_{Z_2}. This possibility is ruled out by the fact that scale generator standard deviations fit independently to each component are in nearly perfect agreement for most movies. On the other hand, σ_{Y_1} and σ_{Y_2} need not be similar, and many movies had a larger standard deviation of motion on the horizontal axis than vertical (the ratio $\sigma_{Y_2}/\sigma_{Y_1}$ tended to be less than one, Figure 4E). We observe a wide range of scale generator standard deviations σ_Z both within and across categories (Figure 4D). The trend across categories–namely that animal and plant movies tend to have higher scale standard deviations than water movies–agrees with the relative heaviness of the tails for the pooled data (Figure 4C).

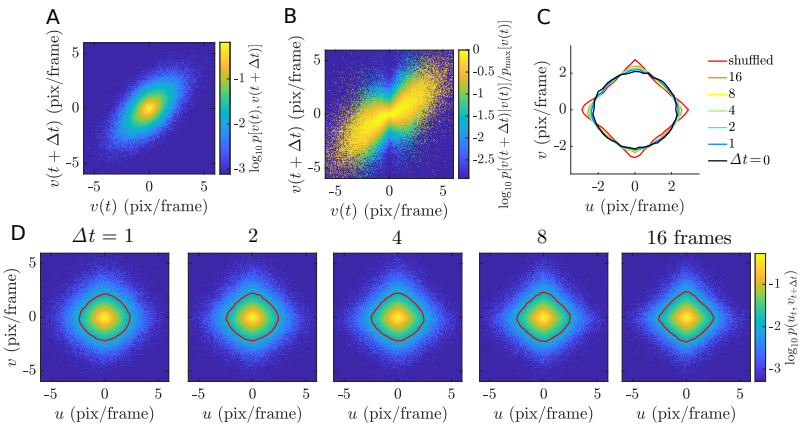

FIG. 5: **Temporal correlations in velocity and scale. A-B.** Joint (**A**) and conditional (**B**) histograms for horizontal velocity across two adjacent frames for an example movie (`bees8-full`). The tilt indicates a strong linear correlation, while the elliptic shape in (**A**) and bow-tie shape in (**B**) indicate the coexistence of a nonlinear dependence due to an underlying scale variable. **C-D.** Isoprobability contours at $p = 0.01$ of the joint distributions of the two components separated by τ frames (**D**) show a gradual transformation from the original circle (**Figure 2B**) towards the diamond shape of the shuffled distribution (**Figure 2C**), indicating that the nonlinear dependence decays slowly over time. Isoprobability contours are overlaid in **C** for clarity.

The dynamics of natural motion

While the time-independent statistics of velocity are important, a complete description of how objects move must include how the velocity evolves over time along object trajectories. This motivates our point tracking analysis, which provides information that cannot be gleaned from motion estimates at fixed locations alone. From the raw data we know the velocity is highly correlated at short time lags, but it is not clear how the scale variable enters into play. We again inspect the joint velocity distribution for an example movie, this time across neighboring time points for one velocity component (Figure 5A). The tilt indicates linear correlation, as expected, and we note that the overall shape is elliptic. In Figure 5B we condition on the velocity at one time-step, and observe the same bow-tie shape as in the horizontal-vertical joint distribution. Thus, two forms of dependence–linear correlation and the nonlinear dependence due to the scale variable–coexist.

We next ask whether this scale variable is constant or dynamic. If it is constant, the jointly heavy-tailed distribution of the two components will not depend on the alignment of the two components in time, so long as they are from the same trajectory. In Figure 5D, we examine these joint distributions after shifting one component relative to the other by a time lag, for a range of lags. The distributions gradually shift from the radially symmetric zero-lag distribution to a diamond shape similar to the shuffled distribution. This is most clearly seen by comparing the $p = 0.01$ isoprobability contours as the lag increases (Figure 5C). In other words, the nonlinear dependence induced by the shared scale variable decreases as the lag increases. We conclude that the underlying scale variable is dynamic. Notably, we can detect these changes within the ~ 1 s long trajectories to which we limit our analysis. This would not be the case if the scale were to change only on a very long timescale or only across different objects within a scene.

We want to capture this dynamic scale variable in our model of natural motion. It is straightforward to make the GSM model dynamic by replacing each Gaussian variable with an autoregressive Gaussian process, and we call this new model the ARGSM model. We illustrate it schematically in Figure 6 by generating example traces for one Gaussian velocity component Y and the scale generator Z. Given this model, we perform estimation of the parameters using a stochastic approximation variant of the expectation-maximization (EM) algorithm (see *Materials and Methods*). Example traces illustrating the results of this model are shown in Figure 7A-C. The estimated autoregression coefficients determine the correlation functions of the underlying Gaussian velocity and scale processes, which we plot for each movie in Figures 7D and E, respectively. Average correlation functions across categories are shown in Figure 7F-G. We also report the time to 0.5 correlation for each movie for the velocity in Figure 7H-I. In the context of visual tracking of moving objects, the timescales of these correlations functions are extremely important. On one hand, the velocity correlation time determines how far into the future motion can be extrapolated. On the other hand, the scale correlation time determines the timescale on which adaptation must take place in order to efficiently process motion signals with limited dynamic range.

Finally, we ask whether the full ARGSM model is necessary to carry out scale normalization in practice for our trajectory data. Our model fitting provides an estimate of the scale at each time point, which we use to normalize the raw data. To quantify normalization performance, we

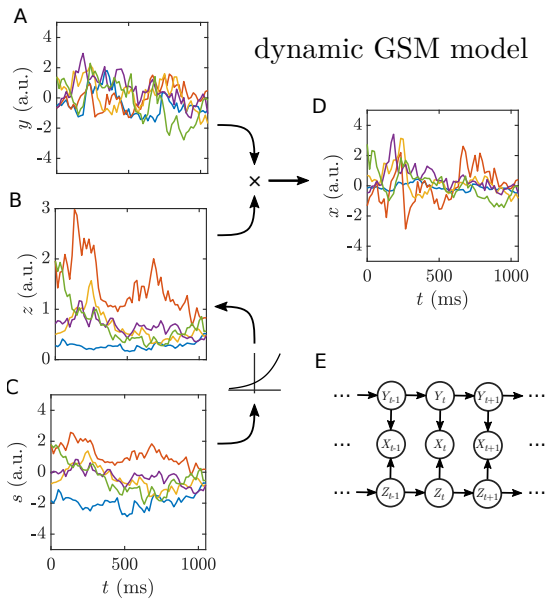

FIG. 6: **Schematic of the dynamic Gaussian scale-mixture model. A-D.** Both Y (**A**) and Z (**C**) are modeled by high-order autoregressive processes to capture arbitrary correlation functions. (Only $AR(1)$ processes are depicted graphically and used to simulate data.) The scale process S (**B**) is generated by passing Z through an element-wise exponential nonlinearity. It then multiplies the underlying Gaussian process Y element-wise to yield the observed process with fluctuating scale (**D**). Only one component is depicted. In the full model, two independent Gaussian processes share a common scale process. **E.** Dependency graph for the variables in the one-dimensional model.

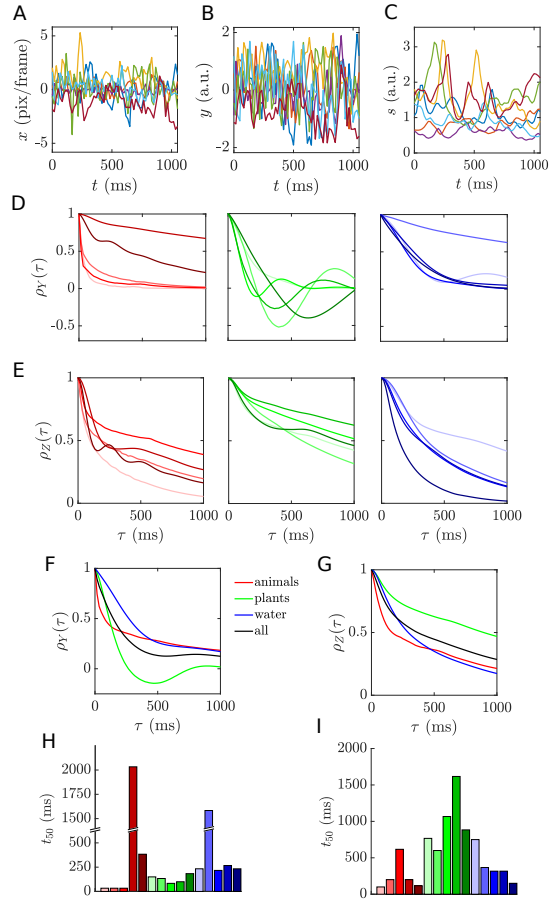

FIG. 7: **Quantifying velocity and scale correlations. A-C.** Example traces of the raw velocity (**A**), scale-normalized velocity (**B**), and estimated scale variable (**C**). **D.** Temporal correlation functions for the underlying Gaussian processes of each movie, grouped by category. Horizontal and vertical components were averaged before normalizing (equivalently, each component was weighted by its variance). **E.** As in **D**, for the scale-generating Gaussian process, Z. **F.** The Gaussian process correlation functions in **D** averaged within categories. **G.** As in **F**, for the scale-generating Gaussian process correlation functions in **E**. **H.** Lag time to reach a correlation of 0.5 for the underlying velocity Gaussian processes for each movie (components were weighted by variance as in **D**). **I.** As in **H**, for the scale-generating Gaussian process.

calculate the kurtosis, or fourth standardized moment, which measures how heavy-tailed a distribution is. The standard reference is a Gaussian random variable, which has a kurtosis of 3. In Figure 8 we compare the kurtosis of the velocity before and after dividing by a point estimate of the scale under three models of increasing complexity. If normalization is successful, the distribution of the resulting normalized velocity should be approximately Gaussian. Under the time-independent model (Figure 8A) the normalized velocity consistently has kurtosis less than 3, indicating that the scale tends to be over-estimated. In contrast, for a model with correlated velocity and constant scale for each trajectory (Figure 8B), the kurtosis is consistently larger than 3, indicating that the scale tends to be underestimated. Only the full model, with correlated velocity and a dynamic, correlated scale variable (Figure 8C) yields a kurtosis around 3 for each movie, even with highly kurtotic data.

This exercise of using the ARGSM model to estimate the scale at each time point, then dividing the velocity by this scale, serves as a proxy for what the nervous system can achieve through adaptation mechanisms. An important caveat is that the model has access to the full trajectory, while the nervous system must operate in an online, causal setting.

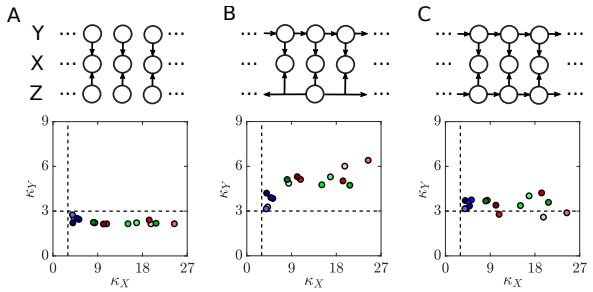

FIG. 8: **A dynamic scale-mixture model is necessary for effective normalization. A.** Kurtosis of the velocity before and after dividing by a point estimate of the scale (*bottom*) under the time-independent model (*top*). Kurtosis was computed by pooling the two components after normalizing by each standard deviation, so that differences in the variance across components do not contribute additional kurtosis. A Gaussian distribution has a kurtosis of 3 (dashed lines). **B.** As in **A**, but for a model with autocorrelated Gaussian processes and a constant scale for each trajectory. **C.** As in **A**, but for the fully dynamic model.

DISCUSSION

The pattern of heavy-tailed velocity distributions, with a scale parameter that is shared across velocity components and fluctuates in time, is remarkably consistent across scenes and categories, despite substantial variation in the content of those scenes, velocity correlation functions, and the overall velocity variance. Together with previous results showing similar statistics in natural images and sounds [10], this suggest that scale-mixing is a fundamental property of natural stimuli, with deep implications for both neural coding and behavior.

In the context of object motion, scale-mixing may arise from two distinct mechanisms, as outlined in our discussion of Brownian motion. First, objects may appear at a variety of distances from the observer, and those distances may change over time. The velocity of a point on an object as it appears to an image forming device, like a camera or eye, is an angular velocity, which can be calculated as the tangential component of the physical velocity divided by the distance. A fluctuating distance thus scales the overall angular velocity over time, analogously to changing the overall magnification in the Brownian motion experiment. Second, the scale of the driving forces may fluctuate over time. This is analogous to temperature fluctuations in the Brownian motion experiment. In our scenes, this corresponds to changes in the behavioral states of animals or to the turbulent nature of water and wind driving plant motion.

Regardless of the source of scale-mixing, strategies for encoding and behaviorally compensating for it should be similar. On the encoding side, the presence of local scale variables suggests that sensory systems should adapt their response properties to local statistics in order to maximize information transmission. Given a static distribution of some input stimulus, the optimal neural response function is the one produces a uniform distribution over the neuron's dynamic range [2]. We demonstrate that the scale of the distribution changes over time, so the gain of the response function should also change to match it [54–56]. Such adaptation or gain control is observed throughout the nervous system (see [57] for a recent review), including in systems relevant to object motion [39, 40, 58]. This adaptation could be the result of subcellular mechanisms, such as the molecular kinematics of synaptic vesicle release [59], or nonlinear circuit mechanisms [60, 61]. One of the contributions of our work is measuring the timescale on which the scale variable fluctuates, which determines the timescale on which adaptation mechanisms should operate in order to be effective.

Beyond single-cell adaptation, our results are also relevant to a population-level adaptation mechanisms known as divisive normalization [62, 63], in which neighboring neurons in a population are mutually inhibitory in a divisive fashion. In many systems, motion is represented by a local population of neurons, each tuned to a narrow band of directions. Our results show that the fluctuating scale is shared between horizontal and vertical velocity components, and, hence, adaptation should ideally be distributed throughout the local population. Divisive normalization is a prime candidate for the implementation of this population-level adaptation, as has been suggested for GSM models of filter activations [10, 64–68]. Such divisive normalizing could remove the nonlinear dependence of horizontal and vertical velocity components due to a shared scale variable. Most models of divisive normalization only capture steady-state responses to static or constant velocity stimuli, although some work has been done to describe its dynamics [69, 70]. Again, these dynamics should be tuned to the timescale of the scale variable fluctuations measured here.

These data suggest a previously unexplored challenge for adaptation mechanisms in the context of object motion: an object may travel an appreciable distance before local mechanisms have a chance to take effect. A solution is to pool from a larger neighborhood, or, more intriguingly, for a local population to receive an adaptation signal selectively from those neurons in nearby populations whose preferred directions point to it. To our knowledge, these hypotheses have not yet been explored, either theoretically or experimentally.

In terms of behavior, our results help refine our understanding of the object tracking problems animals must solve in natural environments, which are crucial to survival. A commonly invoked framework for tracking is sequential Bayesian inference under a state-space model [41]. In this framework, the brain has a probabilistic representation of the state of the object (that is, a probability distribution over its position and velocity). An

internal model of object motion is used to evolve this distribution forward in time, and this prediction is combined with incoming measurements to update the estimated state distribution. Under Gaussian assumptions this yields the famous Kalman filter solution [71]. Our work has two important implications for the state-space model framework of object tracking. First, the velocity distributions we observe are typically non-Gaussian, so the Kalman filter solution is not strictly applicable. Second, state-space models usually model the velocity as either an $AR(1)$ or (discrete) diffusion process (i.e., a nonstationary $AR(1)$ process with coefficient equal to one). The AR models we fit for the underlying Gaussian components generally have more than one large coefficient. The ARGSM model could naturally serve as a predictive state-space model that incorporates these empirical observations by including the recent history of the velocity and scale in the state description (note that the scale does not have a corresponding direct measurement, but it can be estimated the incoming velocity measurements). Flexible Bayesian methods like the particle filter [72] can be used to implement such a model. The merging of the sort of adaptation mechanisms described above with neuromorphic particle filtering [73] is an intriguing avenue for future research.

Finally, the significant diversity in velocity and scale correlation functions and variance has implications both for efficient coding and tracking. Namely, an encoder or tracker which is optimized for the statistics of one scene will be suboptimal for others. Indeed, there is a general trade-off in adaptation to global versus local statistics [74, 75]. The original efficient coding work posited adaptation on evolutionary timescales to natural scene statistics. Here, we emphasize the subsecond timescale of scale fluctuations in natural motion. Neural systems should also have the flexibility to adapt on intermediate timescales to changes in the environment or behavioral context [76].

MATERIALS AND METHODS

Point tracking

We compute short trajectories using the `PointTracker` function in Matlab's (MathWorks, Natick, MA) Computer Vision toolbox. The function employs a Kanade-Lucas-Tomasi [35, 36] feature tracking algorithm, which uses image registration under an affine motion model to track individual points from frame to frame. Our strategy is to collect as many high-quality, short trajectories (64 frames) as possible from each movie, then subsample these down to a reasonable number of trajectories (8,192) for statistical analysis. Initial points are seeded using the `detectMinEigenFeatures` function, which detects image features that can be tracked well under the motion model [77]. From the initial seeds, we run the tracking algorithm forward and backward for 32 frames each, rather than running it in one direction for 64 frames. This increases the chances capturing short-lived trajectories bounded by occlusion or image boundaries. Points are seeded on each frame, so the resulting set of trajectories is highly overlapping. On most movies we employ the built-in forward-backward error checking method [78], with a threshold of 0.5 pixels, to automatically detect tracking errors. The exceptions are three movies depicting water (`water3`, `water5`, and `water6`) where the small threshold leads to rejecting most trajectories. In these cases there are not well-defined objects, so relaxing this strict criterion is justified. The algorithm uses a multi-resolution pyramid and computes gradients within a neighborhood at each level. We use the default values of 3 pyramid levels and a neighborhood size of 31 by 31 pixels for all movies except the 3 water movies, where we find we can decrease the amount of erroneously large jumps in trajectories by increasing the neighborhood size to 127 by 127 pixels and using only 1 pyramid level (at a cost of greater computation time).

This method automatically tracks the stationary background points, which may be erroneously "picked up" by a moving object as it traverses that location. To ensure that the trajectories we analyze are full of motion, we define a speed threshold of 0.5 pix/frame, and discard trajectories that do not satisfy the following criteria: there are no runs of 7 or more frames with speed under this threshold; no more than half the total number of frames have speed under this threshold; the average speed for the whole trajectory is above this threshold; the total velocity variance within the trajectory is greater than 0.25; and the velocity correlation across one time step within the trajectory is greater than 0.3.

The velocity time series is simply the first difference of the point positions along each trajectory. Within each ensemble, we subtract the ensemble mean from each velocity (this is typically very close to zero, except for some water movies with a persistent flow). We then slightly rotate the horizontal and vertical velocity components to remove small correlations between them (these arise if, for example, objects tended to move along a slight diagonal relative to the camera's sensor). All visualizations and calculations are carried out after these minor preprocessing steps. Note that we do not subtract the average velocity within each trajectory, as this introduces an artificial anticorrelation at long lags.

Gaussian scale-mixture models

The basic one-dimensional Gaussian scale-mixture model is described in the main text. Note that for some choices of the distribution for S, the distribution for X has a closed-form solution. For example, the well-known Student's t-distribution is formed when S follows an inverse χ-distribution, and the Laplace distribution is formed when S follows a Rayleigh distribution. In this work, we assume S follows a log-normal

distribution, which does not yield a closed-form distribution for X. This choice makes modeling correlations straightforward, as will be made clear below. In practice, the lack of a closed-form $p(x)$ is not a drawback, since we do not need to normalize the posterior distribution, $p(s|x) \propto p(x|s)p(s)$, in order to sample from it.

When considering multiple variables, a shared scale variable introduces a nonlinear form of dependence between them. Suppose $X_1 = Y_1 S$ and $X_2 = Y_2 S$. If Y_1 and Y_2 are uncorrelated, then X_1 and X_2 are conditionally independent given S:

$$p(x_1, x_2|s) = p(x_1|s)p(x_2|s)\,.$$

However, X_1 and X_2 are not, in general, independent:

$$p(x_1, x_2) = \int_0^\infty p(x_1|s)p(x_2|s)p(s)ds \neq p(x_1)p(x_2)\,.$$

This nonlinear dependence manifests itself in the elliptic level sets of $p(x_1, x_2)$, in contrast to the diamond-shaped level sets of $p(x_1)p(x_2)$. Note that this nonlinear dependence can coincide with the usual linear dependence if Y_1 and Y_2 are correlated, and that a weaker form of nonlinear dependence may be present if $X_1 = Y_1 S_1$ and $X_2 = Y_2 S_2$, where S_1 and S_2 are not independent.

Autoregressive models

Autoregressive models are a well-established and flexible way to capture correlations in time series data by supposing a linear relationship between the current value of a random variable with its previous values. Given a time series, $\{X_1, \ldots, X_T\}$, the kth order autoregressive, or AR(k), model is given by

$$X_t = \sum_{i=1}^{k} \phi_i X_{t-i} + \Xi_t$$

where $\{\phi_1, \ldots, \phi_k\}$ are regression coefficients, Ξ_t is the innovation noise, and

$$p(\xi_t) = \mathcal{N}(\xi_t; 0, \sigma^2)$$

The order k is typically chosen by cross-validation to avoid over-fitting. This makes sense from the standpoint of finding a model that generalizes well to new data. However, our primary aim here is simply to measure the autocovariances of the hidden variables, since these timescales are relevant to prediction and adaptation in the nervous system. For this reason, we choose k to be as high as possible: $k = 31$ time steps, since k must be less than $T/2$.

Typically, the model parameters are fit by standard linear regression (after organizing the data appropriately) [79]. However, this method gives maximum likelihood estimates only if the initial k time steps are considered fixed. If the initial data are assumed to be drawn from the stationary distribution defined by the parameters, the problem becomes nonlinear. The EM algorithm described below requires parameter estimates to be maximum likelihood, and since we would like the initial k time steps (where k is large) to be modeled by the stationary distribution, we must pursue this more difficult course. Following [80], we consider the time series as a T-dimensional vector, $\mathbf{X} = [X_1 \ldots X_T]^\top$. Then

$$p(\mathbf{x}) = \mathcal{N}(\mathbf{x}; \mathbf{0}, \mathbf{\Sigma})\,,$$

where

$$\mathbf{\Sigma} = E[\mathbf{X}\mathbf{X}^\top] = \text{Toeplitz}(\gamma_0, \ldots, \gamma_{T-1})\,.$$

The autocovariance series $\{\gamma_0, \ldots, \gamma_{T-1}\}$ is related to the regression coefficients and innovation variance by

$$\gamma_0 = \sum_{i=1}^{k} \phi_i \gamma_i + \sigma^2$$
$$\gamma_j = \sum_{i=1}^{k} \phi_i \gamma_{i-j} \qquad j \geq 1\,.$$

Given a collection of N samples of length T from the time series, we calculate the matrix $\mathbf{R}$ defined by

$$R_{ij} = R_{ji} = \frac{1}{N} \sum_{n=1}^{N} \sum_{t=i+1}^{T-j} x_{n,t} x_{n,t+i-j} \qquad 0 \leq i \leq j \leq k\,,$$

where the index n ranges over samples and t ranges over time steps. Then the maximum likelihood estimate of the parameters satisfies a nonlinear system of $2(k+1)$ equations, consisting of the $k+1$ autocovariance relations above, up to γ_k, together with the following:

$$R_{00} = \sum_{i=1}^{k} \phi_i \left(R_{0i} + i\gamma_i\right) + T\sigma^2$$
$$R_{j0} = \sum_{i=1}^{k} \phi_i \left(R_{ij} + i\gamma_{i-j}\right) \qquad 1 \leq j \leq k\,.$$

We solve this system numerically using Matlab's `fsolve`.

While the covariance matrix is easily calculated from the AR model parameters, it is not well-appreciated that the inverse covariance matrix can be calculated exactly from the parameters as well [81]. This is useful because it is the inverse that is needed for the sampling methods described below, and inverting large, nearly singular matrices is numerically unstable. The inverse of $\mathbf{\Sigma}$ is given by

$$\mathbf{\Sigma}^{-1} = \frac{1}{\sigma^2}\left(\mathbf{I} + \sum_{i=1}^{k} \phi_i^2 \mathbf{E}_i - \sum_{i=1}^{k} \phi_i \mathbf{F}_i + \sum_{i=1}^{k-1} \sum_{j=1}^{p-i} \phi_i \phi_{i+j} \mathbf{G}_{j,i+j}\right)\,,$$

where $\mathbf{I}$ is the identity matrix, $\mathbf{E}_i$ is the identity matrix with the first and last i diagonal elements set to zero, $\mathbf{F}_i$

is a matrix with ones along the ith upper and lower diagonals and zeros elsewhere, and $\mathbf{G}_{j,i+j} = \mathbf{E}_i \mathbf{F}_j \mathbf{E}_i$ (note that we have corrected a typo in [81] in the indexing of the last term of the equation).

The ARGSM model

The dynamic scale-mixture model generalizes the two-dimensional, shared scale variable GSM model described above to time series, assuming the underlying Gaussian random variables, Y_1 and Y_2, and the generator, Z, of the scale variable are all $AR(k)$ processes. Specifically, let $X_{1,t} = Y_{1,t} S_t$ and $X_{2,t} = Y_{2,t} S_t$. Written as T-dimensional vectors, we have $\mathbf{X}_1 = \mathbf{Y}_1 \odot \mathbf{S}$ and $\mathbf{X}_2 = \mathbf{Y}_2 \odot \mathbf{S}$, where $\odot$ is element-wise multiplication. The AR process assumptions imply

$$\begin{aligned} p(\mathbf{y}_1) &= \mathcal{N}\left(\mathbf{y}_1; \mathbf{0}, \mathbf{\Sigma}_{Y_1}\right) \\ p(\mathbf{y}_2) &= \mathcal{N}\left(\mathbf{y}_2; \mathbf{0}, \mathbf{\Sigma}_{Y_2}\right) \\ p(\mathbf{z}) &= \mathcal{N}\left(\mathbf{z}; \mathbf{0}, \mathbf{\Sigma}_{Z}\right) \end{aligned}$$

where the covariance matrices are determined by the parameters of independent $AR(k)$ models as described above. $\mathbf{S}$ is related to $\mathbf{Z}$ by element-wise application of the exponential function, $\mathbf{S} = \exp(\mathbf{Z})$. When $\mathbf{Z}$ is known, we have

$$\begin{aligned} p(\mathbf{x}_1|\mathbf{z}) &= \mathcal{N}\left(\mathbf{x}_1; \mathbf{0}, \mathbf{D_s} \mathbf{\Sigma}_{Y_1} \mathbf{D_s}\right) \\ p(\mathbf{x}_2|\mathbf{z}) &= \mathcal{N}\left(\mathbf{x}_2; \mathbf{0}, \mathbf{D_s} \mathbf{\Sigma}_{Y_2} \mathbf{D_s}\right) \ , \end{aligned}$$

where $\mathbf{D_s}$ is the matrix with the elements of $\mathbf{s}$ along the diagonal and zeros elsewhere.

EM and stochastic approximation

The classic Expectation-Maximization, or EM, algorithm is a useful tool for finding (local) maximum likelihood estimates of parameters of hidden variable models [53]. Let $\theta = \left\{\boldsymbol{\phi}_{Y_1}, \boldsymbol{\phi}_{Y_2}, \boldsymbol{\phi}_Z, \sigma^2_{Y_1}, \sigma^2_{Y_2}, \sigma^2_Z\right\}$ be the collection of parameters of the ARGSM model, where each $\boldsymbol{\phi}$ is the vector of AR coefficients and each σ^2 is the innovation variance for each variable. The observed data, $\mathcal{D} = \{\mathbf{x}_{1,n}, \mathbf{x}_{2,n}\}, 1 \leq n \leq N$, are the N pairs of T-dimensional vectors corresponding here to the horizontal and vertical velocity along each trajectory. The hidden variables, $\mathcal{H} = \{\mathbf{z}_n\}, 1 \leq n \leq N$, are the Gaussian generators of the time-varying scale associated with each trajectory. The likelihood,

$$\begin{aligned} L &= \log p(\mathcal{D}|\theta) \\ &= \sum_{n=1}^{N} \log \int_{\mathbb{R}^T} p(\mathbf{x}_{1,n}, \mathbf{x}_{2,n}|\mathbf{z}, \theta) p(\mathbf{z}|\theta) D\mathbf{z} \end{aligned}$$

is intractable to maximize due to the high-dimensional integral. The EM algorithm finds a local maximum iteratively. Starting with an initial guess for the parameters, θ_0, at each step one computes the expectation with respect to the probability distribution of the hidden variables given the data and the current parameter estimate θ_t of the complete data log-likelihood,

$$Q(\theta|\theta_t) = E_{p(\mathcal{H}|\mathcal{D},\theta_t)}\left[\log p(\mathcal{D}, \mathcal{H}|\theta)\right] \ ,$$

then updates the parameters to maximize this function,

$$\theta_{t+1} = \arg\max_{\theta} Q(\theta|\theta_t) \, .$$

In our setting, we have

$$\begin{aligned} &Q(\theta|\theta_t) = \\ &-\frac{1}{2}\sum_{n=1}^{N} E\left[\mathbf{y}_{1,n}^\top \mathbf{\Sigma}_{Y_1}^{-1} \mathbf{y}_{1,n} + \mathbf{y}_{2,n}^\top \mathbf{\Sigma}_{Y_2}^{-1} \mathbf{y}_{2,n} + \mathbf{z}_n^\top \mathbf{\Sigma}_Z^{-1} \mathbf{z}_n\right] + K \\ &\qquad = -\frac{1}{2}\mathrm{tr}\left(\mathbf{C}_{Y_1}\mathbf{\Sigma}_{Y_1}^{-1} + \mathbf{C}_{Y_2}\mathbf{\Sigma}_{Y_2}^{-1} + \mathbf{C}_Z \mathbf{\Sigma}_Z^{-1}\right) + K \end{aligned}$$

where

$$\mathbf{C}_Z = \sum_{n=1}^{N} E\left[\mathbf{z}_n \mathbf{z}_n^\top\right]$$

and similarly for $\mathbf{C}_{Y_1}$ and $\mathbf{C}_{Y_2}$. Note that in this context, the $\mathbf{y}$ variables are merely shorthand for $\mathbf{y}_{1,n} = \mathbf{x}_{1,n} \oslash \mathbf{s}_n$ and $\mathbf{y}_{2,n} = \mathbf{x}_{2,n} \oslash \mathbf{s}_n$, where $\oslash$ is element-wise division. Given the $\mathbf{C}$ matrices, the corresponding $\mathbf{R}$ matrices defined above are easily computed, from which the maximum likelihood estimates of the AR parameters can be calculated.

Unfortunately, the expectation values in the $\mathbf{C}$ matrices are also intractable, but they can be approximated through sampling methods. A variant of the EM algorithm, called stochastic approximation EM, was developed to address this problem [82]. Given a sample from the distribution $p(\mathcal{H}|\mathcal{D}, \theta_t)$, one calculates the sample matrices $\hat{\mathbf{C}}$, then updates the stochastic approximations as

$$\mathbf{C}_t = \mathbf{C}_{t-1} + \eta_t \left(\hat{\mathbf{C}}_t - \mathbf{C}_{t-1}\right) \, .$$

The sequence of parameters η_t is given by

$$\eta_t = \begin{cases} 1 & 1 \leq t \leq \alpha \\ (t-\alpha)^{-\beta} & t > \alpha \end{cases} \, .$$

We choose $\alpha = 2500$ or 5000, so that the algorithm runs in a fully stochastic mode until the parameter estimates are nearly stationary, and $\beta = 1$, so that after this initial period, the algorithm converges by simply taking a running average of the samples of the $\hat{\mathbf{C}}$ matrices. Importantly, the samples do not need to be independent across iterations for the algorithm to converge [83]. This means that, when performing the Gibbs sampling described below, we only need to update each hidden variable element once for each iteration, rather than updating many

times and throwing out samples to achieve independence. Since each M-step (the AR model MLE algorithm described above) is much faster than each E-step (calculating the $\mathbf{C}$ matrices through sampling), this results in a more sample-efficient algorithm [84].

We also estimate the expectation of the hidden variables $\{\mathbf{z}_n\}$ in an identical fashion. This is equivalent to a Bayesian point estimate where the estimated parameters form a forward model and prior. These estimates are then used to remove the scale from the velocity, in order to examine the kurtosis under different model assumptions (Figure 4).

The EM algorithm, and its stochastic approximation variant, converges to a local maximum of the likelihood function that depends on the initial conditions. We find that, in practice, it is important to introduce the scale variable gradually to the model. We initialize the model with AR parameters fit to the raw data for the Y_1 and Y_2 components, and let Z be uncorrelated with very small variance (regression coefficients $\boldsymbol{\phi}_Z = \mathbf{0}$ and innovation variance $\sigma_Z^2 = 0.05^2(\gamma_{Y_1,0} + \gamma_{Y_2,0})/2$).

Sampling methods

We use a combination of Gibbs and rejection sampling to sample from the posterior of the hidden variables given the data and the current parameter estimates [85]. Gibbs sampling works by starting with an initial vector $\mathbf{z}$ and updating each element z_t (in a random order) conditioned on the remaining elements $\mathbf{z}_{\backslash t}$, where the notation $\backslash t$ indicates all indices in the set $\{1, \ldots, T\}\backslash t$. The posterior distribution is given by

$$\begin{aligned} p(z_t|\mathbf{x}_1, \mathbf{x}_2, \mathbf{z}_{\backslash t}) \propto \\ p(x_{1,t}, x_{2,t}|\mathbf{x}_{1,\backslash t}, \mathbf{x}_{2,\backslash t}, \mathbf{z}) p(z_t|\mathbf{x}_{1,\backslash t}, \mathbf{x}_{2,\backslash t}, \mathbf{z}_{\backslash t}) \\ = p(x_{1,t}|\mathbf{x}_{1,\backslash t}, \mathbf{z}) p(x_{2,t}|\mathbf{x}_{2,\backslash t}, \mathbf{z}) p(z_t|\mathbf{z}_{\backslash t}) \,. \end{aligned}$$

Each distribution in the product is a Gaussian given by

$$\begin{aligned} p(x_{1,t}|\mathbf{x}_{1,\backslash t}, \mathbf{z}) &= \mathcal{N}(x_{1,t}, \hat{\mu}_{Y_1} s_t, \hat{\sigma}_{Y_1}^2 s_t^2) \\ p(x_{2,t}|\mathbf{x}_{2,\backslash t}, \mathbf{z}) &= \mathcal{N}(x_{2,t}, \hat{\mu}_{Y_2} s_t, \hat{\sigma}_{Y_2}^2 s_t^2) \\ p(z_t|\mathbf{z}_{\backslash t}) &= \mathcal{N}(\hat{\mu}_Z, \hat{\sigma}_Z^2) \,, \end{aligned}$$

where

$$\begin{aligned} \hat{\sigma}_{Y_1}^2 &= \left(\boldsymbol{\Sigma}_{Y_1}^{-1}\right)_{tt}^{-1} \\ \hat{\sigma}_{Y_2}^2 &= \left(\boldsymbol{\Sigma}_{Y_2}^{-1}\right)_{tt}^{-1} \\ \hat{\sigma}_Z^2 &= \left(\boldsymbol{\Sigma}_Z^{-1}\right)_{tt}^{-1} \\ \hat{\mu}_{Y_1} &= -\hat{\sigma}_{Y_1}^2 \left(\boldsymbol{\Sigma}_{Y_1}^{-1}\right)_{t,\backslash t} \mathbf{y}_{1,\backslash t} \\ \hat{\mu}_{Y_2} &= -\hat{\sigma}_{Y_2}^2 \left(\boldsymbol{\Sigma}_{Y_2}^{-1}\right)_{t,\backslash t} \mathbf{y}_{2,\backslash t} \\ \hat{\mu}_Z &= -\hat{\sigma}_Z^2 \left(\boldsymbol{\Sigma}_Z^{-1}\right)_{t,\backslash t} \mathbf{z}_{\backslash t} \,. \end{aligned}$$

Rejection sampling is used to draw a sample of each individual z_t. In this framework, a proposal distribution that is easy to sample from, $q(z)$, is chosen as an envelope (after an appropriate scaling) of a target distribution, $\hat{p}(z)$, from which we would like to sample, that is, $\hat{p}(z) \leq Mq(z)$ for some positive scaling factor M. The proposal distribution is sampled from, followed by sampling another variable u from a uniform distribution. If $u \leq Mq(z)/\hat{p}(z)$, the sample is accepted; otherwise, it is rejected and the sampling is repeated. The method produces samples from the target distribution exactly, even if it is unnormalized, but requires the envelope to be tight to avoid rejecting too many samples. Here, the target distribution is the the unnormalized posterior (dropping time indices for clarity),

$$\hat{p}(z) = \mathcal{N}(x_1; \hat{\mu}_{Y_1} s, \hat{\sigma}_{Y_1}^2 s^2) \mathcal{N}(x_1; \hat{\mu}_{Y_2} s, \hat{\sigma}_{Y_2}^2 s^2) \mathcal{N}(z; \hat{\mu}_Z, \hat{\sigma}_Z^2) \,.$$

We let the proposal distribution be a Gaussian,

$$q(z) = \mathcal{N}(z; \mu_q, \sigma_q^2) \,,$$

whose mean and variance are optimized heuristically to form a tight envelope of the target distribution. We center this Gaussian over the peak of the target distribution,

$$\mu_q = \arg\max_z \hat{p}(z) \,.$$

and compute the scale factor M to match the two distributions at the peak, $M = \hat{p}(\mu_q)/q(\mu_q)$. To optimize the variance, note that we can write

$$\begin{aligned} &\log \hat{p}(z) = \\ &-\frac{1}{2}\frac{\left[x_1 \exp(-z) - \hat{\mu}_{Y_1}\right]^2}{\hat{\sigma}_{Y_1}^2} - \frac{1}{2}\frac{\left[x_2 \exp(-z) - \hat{\mu}_{Y_2}\right]^2}{\hat{\sigma}_{Y_2}^2} \\ &-\frac{1}{2}\frac{\left[z - \left(\hat{\mu}_Z - 2\hat{\sigma}_Z^2\right)\right]^2}{\hat{\sigma}_Z^2} - 2\left(\hat{\mu}_Z - \hat{\sigma}_Z^2\right) + K \,. \end{aligned}$$

The first two terms rapidly approach constants to the right and diverge negatively to the left. Thus, the right tail of $\hat{p}(z)$ behaves as $\mathcal{N}(z; \hat{\mu}_Z - 2\hat{\sigma}_Z^2, \hat{\sigma}_Z^2)$, and the left tail falls off extremely rapidly. A variance of $\sigma_q^2 = \hat{\sigma}_Z^2$ is necessary and sufficient to cover the two tails when the peak of the proposal distribution is to the right of the peak of this Gaussian distribution, i.e., when $\mu_q \geq \hat{\mu}_Z - 2\hat{\sigma}_Z^2$. A smaller variance will not fully envelop the right tail regardless of the value of μ_q. We require a slightly larger variance when $\mu_q < \hat{\mu}_Z - 2\hat{\sigma}_Z^2$, which we calculate by assuming $Mq(z)$ makes exactly one other point of contact with $\hat{p}(z)$. Let

$$f(z) = \log \hat{p}(z)$$

and

$$\begin{aligned} g(z) &= \log[Mq(z)] \\ &= -\frac{1}{2}\frac{(z - \mu_q)^2}{\sigma_q^2} + f(\mu_q) \end{aligned}$$

Let z_0 be the point of contact. Then $f(z_0) = g(z_0)$ implies

$$\sigma_q^2 = \frac{1}{2}\frac{(z_0-\mu_q)^2}{f(\mu_q)-f(z_0)}\,.$$

The two curves must also be tangent at z_0, $f'(z_0) = g'(z_0)$. We have

$$g'(z_0) = -\frac{z_0-\mu_q}{\sigma_q^2} = -2\frac{f(\mu_q)-f(z_0)}{z_0-\mu_q}\,.$$

The tangency condition implies z_0 is the solution to

$$f'(z_0) + 2\frac{f(\mu_q)-f(z_0)}{z_0-\mu_q} = 0\,,$$

which we solve numerically using Matlab's `fzero` function, and use this solution to calculate σ_q^2. Finally, if the above procedure fails due to numerical issues, we simply set $\sigma_q^2 = (1.1\hat{\sigma}_z)^2$, which is large enough to form an envelope in practice.

ACKNOWLEDGEMENTS

This work was supported in part by the National Science Foundation, through the Center for the Physics of Biological Function (PHY-1734030), and through a CAREER award to SEP (IIS-1652617); and by the National Institutes of Health BRAIN initiative (R01EB026943). We thank Siwei Wang Benjamin Hoshal for useful comments on the manuscript.

[1] H. B. Barlow, Possible principles underlying the transformations of sensory messages, in *Sensory Communication*, edited by W. A. Rosenblith (MIT Press, 1961) pp. 217–234.
[2] S. Laughlin, A simple coding procedure enhances a neuron's information capacity, Zeitschrift für Naturforschung c **36**, 910 (1981).
[3] M. V. Srinivasan, S. B. Laughlin, and A. Dubs, Predictive coding: A fresh view of inhibition in the retina, Proceedings of the Royal Society of London B: Biological Sciences **216**, 427 (1982).
[4] J. H. van Hateren, A theory of maximizing sensory information, Biological Cybernetics **68**, 23 (1992).
[5] Y. Dan, J. J. Atick, and R. C. Reid, Efficient coding of natural scenes in the lateral geniculate nucleus: Experimental test of a computational theory, Journal of Neuroscience **16**, 3351 (1996).
[6] C. K. Machens, M. S. Wehr, and A. M. Zador, Linearity of cortical receptive fields measured with natural sounds, Journal of Neuroscience **24**, 1089 (2004).
[7] E. Doi and M. S. Lewicki, A simple model of optimal population coding for sensory systems, PLoS Computational Biology **10**, e1003761 (2014).
[8] E. P. Simoncelli and B. A. Olshausen, Natural image statistics and neural representation, Annual Review of Neuroscience **24**, 1193 (2001).
[9] D. L. Ruderman and W. Bialek, Statistics of natural images: Scaling in the woods, Physical Review Letters **73**, 814 (1994).
[10] O. Schwartz and E. P. Simoncelli, Natural signal statistics and sensory gain control, Nature Neuroscience **4**, 819 (2001).
[11] X. Pitkow, Exact feature probabilities in images with occlusion, Journal of vision **10**, 42 (2010).
[12] D. Zoran and Y. Weiss, Natural images, gaussian mixtures and dead leaves, Advances in Neural Information Processing Systems **25** (2012).
[13] B. G. Borghuis and A. Leonardo, The role of motion extrapolation in amphibian prey capture, Journal of Neuroscience **35**, 15430 (2015).
[14] M. Mischiati, H.-T. Lin, P. Herold, E. Imler, R. Olberg, and A. Leonardo, Internal models direct dragonfly interception steering, Nature **517**, 333 (2015).
[15] S. B. M. Yoo, J. C. Tu, S. T. Piantadosi, and B. Y. Hayden, The neural basis of predictive pursuit, Nature Neuroscience **23**, 252 (2020).
[16] L. Shaw, K. H. Wang, and J. Mitchell, Fast prediction in marmoset reach-to-grasp movements for dynamic prey, Current Biology **33**, 2557 (2023).
[17] F. Gabbiani, H. G. Krapp, C. Koch, and G. Laurent, Multiplicative computation in a visual neuron sensitive to looming, Nature **420**, 320 (2002).
[18] G. Card and M. H. Dickinson, Visually mediated motor planning in the escape response of *Drosophila*, Current Biology **18**, 1300 (2008).
[19] F. T. Muijres, M. J. Elzinga, J. M. Melis, and M. H. Dickinson, Flies evade looming targets by executing rapid visually directed banked turns, Science **344**, 172 (2014).
[20] K. S. Muller, J. Matthis, K. Bonnen, L. K. Cormack, A. C. Huk, and M. Hayhoe, Retinal motion statistics during natural locomotion, eLife **12**, e82410 (2023).
[21] M. J. Berry, I. H. Brivanlou, T. A. Jordan, and M. Meister, Anticipation of moving stimuli by the retina, Nature **398**, 334 (1999).
[22] M. Spering, A. C. Schütz, D. I. Braun, and K. R. Gegenfurtner, Keep your eyes on the ball: Smooth pursuit eye movements enhance prediction of visual motion, Journal of Neurophysiology **105**, 1756 (2011).
[23] A. Ben-Simon, O. Ben-Shahar, G. Vasserman, and R. Segev, Predictive saccade in the absence of smooth pursuit: Interception of moving targets in the archer fish, Journal of Experimental Biology **215**, 4248 (2012).
[24] A. Leonardo and M. Meister, Nonlinear dynamics support a linear population code in a retinal target-tracking circuit, Journal of Neuroscience **33**, 16971 (2013).
[25] D. W. Franklin and D. M. Wolpert, Computational mechanisms of sensorimotor control, Neuron **72**, 425 (2011).
[26] R. J. Krauzlis and S. G. Lisberger, Temporal properties of visual motion signals for the initiation of smooth pursuit eye movements in monkeys, Journal of Neurophysiology **72**, 150 (1994).
[27] M. M. Hayhoe, T. McKinney, K. Chajka, and J. B. Pelz, Predictive eye movements in natural vision, Experimental Brain Research **217**, 125 (2012).

[28] B. P. Ölveczky, S. A. Baccus, and M. Meister, Segregation of object and background motion in the retina, Nature **423**, 401 (2003).

[29] S. A. Baccus, B. P. Ölveczky, M. Manu, and M. Meister, A retinal circuit that computes object motion, Journal of Neuroscience **28**, 6807 (2008).

[30] J. H. Maunsell and D. C. Van Essen, Functional properties of neurons in middle temporal visual area of the macaque monkey. I. Selectivity for stimulus direction, speed, and orientation, Journal of Neurophysiology **49**, 1127 (1983).

[31] P. J. Mineault, F. A. Khawaja, D. A. Butts, and C. C. Pack, Hierarchical processing of complex motion along the primate dorsal visual pathway, Proceedings of the National Academy of Sciences **109**, E972 (2012).

[32] D. Calow and M. Lappe, Local statistics of retinal optic flow for self-motion through natural sceneries, Network: Computation in Neural Systems **18**, 343 (2007).

[33] S. Roth and M. J. Black, On the spatial statistics of optical flow, International Journal of Computer Vision **74**, 33 (2007).

[34] Https://cmd.rcc.uchicago.edu/.

[35] B. D. Lucas and T. Kanade, An iterative image registration technique with an application to stereo vision, in *Proceedings of the 7th International Joint Conference on Artificial Intelligence-Volume 2* (1981) pp. 674–679.

[36] C. Tomasi and T. Kanade, *Detection and Tracking of Point Features*, Tech. Rep. (International Journal of Computer Vision, 1991).

[37] D. F. Andrews and C. L. Mallows, Scale mixtures of normal distributions, Journal of the Royal Statistical Society. Series B (Methodological) **36**, 99 (1974).

[38] M. J. Wainwright, E. P. Simoncelli, and A. S. Willsky, Random cascades on wavelet trees and their use in analyzing and modeling natural images, Applied and Computational Harmonic Analysis **11**, 89 (2001).

[39] B. P. Ölveczky, S. A. Baccus, and M. Meister, Retinal adaptation to object motion, Neuron **56**, 689 (2007).

[40] B. Liu, M. V. Macellaio, and L. C. Osborne, Efficient sensory cortical coding optimizes pursuit eye movements, Nature Communications **7** (2016).

[41] Y. Ho and R. Lee, A Bayesian approach to problems in stochastic estimation and control, IEEE Transactions on Automatic Control **9**, 333 (1964).

[42] B. K. Horn and B. G. Schunck, Determining optical flow, Artificial Intelligence **17**, 185 (1981).

[43] E. H. Adelson and J. R. Bergen, Spatiotemporal energy models for the perception of motion, Journal of the Optical Society of America A **2**, 284 (1985).

[44] D. W. Dong and J. J. Atick, Temporal decorrelation: A theory of lagged and nonlagged responses in the lateral geniculate nucleus, Network: Computation in Neural Systems **6**, 159 (1995).

[45] R. Brown, A brief account of microscopical observations made in the months of June, July and August, 1827, on the particles contained in the pollen of plants; and on the general existence of active molecules in organic and inorganic bodies, The Philosophical Magazine **4**, 161 (1828).

[46] A. Einstein, Über die von der molekularkinetischen theorie der wärme geforderte bewegung von in ruhenden flüssigkeiten suspendierten teilchen, Annalen der Physik **322**, 549 (1905).

[47] P. Langevin, Sur la théorie du mouvement brownien, C. R. Acad. Sci. (Paris) **146**, 530 (1908).

[48] D. T. Gillespie, Exact numerical simulation of the Ornstein-Uhlenbeck process and its integral, Physical Review E **54**, 2084 (1996).

[49] D. T. Gillespie, The mathematics of Brownian motion and Johnson noise, American Journal of Physics **64**, 225 (1996).

[50] G. U. Yule, On a method of investigating periodicities in disturbed series, with special reference to Wolfer's sunspot numbers, Philos. Trans. Royal Soc. A **226**, 267 (1927).

[51] G. T. Walker, On periodicity in series of related terms, Proc. R. Soc. Lond. A **131**, 518 (1931).

[52] M. J. Wainwright and E. P. Simoncelli, Scale mixtures of Gaussians and the statistics of natural images, in *Advances in Neural Information Processing Systems* (2000) p. 7.

[53] A. P. Dempster, N. M. Laird, and D. B. Rubin, Maximum likelihood from incomplete data via the EM algorithm, Journal of the Royal Statistical Society: Series B (Methodological) **39**, 1 (1977).

[54] M. J. Wainwright, Visual adaptation as optimal information transmission, Vision Research **39**, 3960 (1999).

[55] N. Brenner, W. Bialek, and R. d. R. Van Steveninck, Adaptive rescaling maximizes information transmission, Neuron **26**, 695 (2000).

[56] B. Wark, A. Fairhall, and F. Rieke, Timescales of inference in visual adaptation, Neuron **61**, 750 (2009).

[57] A. I. Weber, K. Krishnamurthy, and A. L. Fairhall, Coding principles in adaptation, Annual Review of Vision Science **5**, 427 (2019).

[58] W. Bair and J. A. Movshon, Adaptive temporal integration of motion in direction-selective neurons in macaque visual cortex, Journal of Neuroscience **24**, 7305 (2004).

[59] Y. Ozuysal and S. A. Baccus, Linking the computational structure of variance adaptation to biophysical mechanisms, Neuron **73**, 1002 (2012).

[60] A. Borst, V. L. Flanagin, and H. Sompolinsky, Adaptation without parameter change: Dynamic gain control in motion detection, Proceedings of the National Academy of Sciences **102**, 6172 (2005).

[61] A. Bharioke and D. B. Chklovskii, Automatic adaptation to fast input changes in a time-invariant neural circuit, PLoS Computational Biology **11**, e1004315 (2015).

[62] M. Carandini and D. J. Heeger, Summation and division by neurons in primate visual cortex, Science **264**, 1333 (1994).

[63] M. Carandini and D. J. Heeger, Normalization as a canonical neural computation, Nature Reviews Neuroscience **13**, 51 (2012).

[64] M. J. Wainwright, O. Schwartz, and E. P. Simoncelli, Natural image statistics and divisive normalization, in *Probabilistic Models of the Brain: Perception and Neural Function*, edited by R. Rao, B. Olshausen, and M. Lewicki (MIT Press, 2002).

[65] O. Schwartz, T. J. Sejnowski, and P. Dayan, Soft mixer assignment in a hierarchical generative model of natural scene statistics, Neural Computation **18**, 2680 (2006).

[66] R. Coen-Cagli, P. Dayan, and O. Schwartz, Cortical surround interactions and perceptual salience via natural scene statistics, PLoS Computational Biology **8**, e1002405 (2012).

[67] R. Coen-Cagli, A. Kohn, and O. Schwartz, Flexible gating of contextual influences in natural vision, Nature Neuroscience **18**, 1648 (2015).

[68] M. Snow, R. Coen-Cagli, and O. Schwartz, Specificity and timescales of cortical adaptation as inferences about natural movie statistics, Journal of vision **16** (2016).

[69] K. Louie, T. LoFaro, R. Webb, and P. W. Glimcher, Dynamic divisive normalization predicts time-varying value coding in decision-related circuits, Journal of Neuroscience **34**, 16046 (2014).

[70] U. A. Ernst, X. Chen, L. Bohnenkamp, F. O. Galashan, and D. Wegener, Dynamic divisive normalization circuits explain and predict change detection in monkey area mt, PLoS Computational Biology **17**, e1009595 (2021).

[71] R. E. Kalman, A new approach to linear filtering and prediction problems, Transactions of the ASME–Journal of Basic Engineering **82**, 35 (1960).

[72] P. Del Moral, Nonlinear filtering: Interacting particle resolution, Comptes Rendus de l'Académie des Sciences-Series I-Mathematics **325**, 653 (1997).

[73] A. Kutschireiter, S. C. Surace, H. Sprekeler, and J.-P. Pfister, Nonlinear Bayesian filtering and learning: A neuronal dynamics for perception, Scientific Reports **7**, 8722 (2017).

[74] W. F. Młynarski and A. M. Hermundstad, Adaptive coding for dynamic sensory inference, eLIFE , 43 (2018).

[75] W. Młynarski and A. M. Hermundstad, Efficient and adaptive sensory codes, bioRxiv , 669200 (2020).

[76] T. Teşileanu, S. Golkar, S. Nasiri, A. M. Sengupta, and D. B. Chklovskii, Neural circuits for dynamics-based segmentation of time series, Neural Computation **34**, 891 (2022).

[77] J. Shi and C. Tomasi, Good features to track, in *1994 Proceedings of IEEE Conference on Computer Vision and Pattern Recognition* (1994) pp. 593–600.

[78] Z. Kalal, K. Mikolajczyk, and J. Matas, Forward-backward error: Automatic detection of tracking failures, in *2010 20th International Conference on Pattern Recognition* (IEEE, 2010) pp. 2756–2759.

[79] J. D. Hamilton, *Time Series Analysis* (Princeton University Press, 2020).

[80] J. W. Miller, Exact maximum likelihood estimation in autoregressive processes, Journal of Time Series Analysis **16**, 607 (1995).

[81] A. Verbyla, A note on the inverse covariance matrix of the autoregressive process, Australian Journal of Statistics **27**, 221 (1985).

[82] B. Delyon, M. Lavielle, and E. Moulines, Convergence of a stochastic approximation version of the EM algorithm, Annals of Statistics , 94 (1999).

[83] E. Kuhn and M. Lavielle, Coupling a stochastic approximation version of EM with an MCMC procedure, ESAIM: Probability and Statistics **8**, 115 (2004).

[84] R. M. Neal and G. E. Hinton, A view of the EM algorithm that justifies incremental, sparse, and other variants, in *Learning in Graphical Models* (Springer, 1998) pp. 355–368.

[85] C. M. Bishop, *Pattern Recognition and Machine Learning* (Springer).