Supplemental Modeling

Contents Page

1 Notation

Here we discuss the notation we use, along with a translation to dot product notation for those who prefer that.

We use bold for vectors and matrices, and non-bold for scalars and components of vectors and matrices. Two matrices next to each other corresponds to a dot product; if $\mathbf{A}$ and $\mathbf{B}$ are matrices, then the ij^{th} component of $\mathbf{AB}$ is given by

$$(\mathbf{AB})_{ij} = \sum_k A_{ik} B_{kj} \quad \leftrightarrow \quad (\mathbf{A} \cdot \mathbf{B})_{ij} \,. \tag{1}$$

A vector next to a matrix is a dot product, so long as the transpose symbol (T) is in the right place,

$$(\mathbf{Av})_i = \sum_k A_{ik} v_k \quad \leftrightarrow \quad (\mathbf{A} \cdot \mathbf{v})_i \tag{2a}$$

$$(\mathbf{v}^T\mathbf{A})_j = \sum_k v_k A_{kj} \quad \leftrightarrow \quad (\mathbf{v} \cdot \mathbf{A})_j \,. \tag{2b}$$

Finally, two vectors next to each other is either a dot product or an outer product, depending on whether the transpose is first or second: if $\mathbf{v}$ and $\mathbf{u}$ are vectors,

$$\mathbf{v}^T\mathbf{u} = \sum_i v_i u_i \quad \leftrightarrow \quad \mathbf{v} \cdot \mathbf{u} \tag{3a}$$

$$(\mathbf{vu}^T)_{ij} = v_i u_j \quad \leftrightarrow \quad (\mathbf{vu})_{ij} \,. \tag{3b}$$

You should not see $\mathbf{vA}$ or $\mathbf{Av}^T$ in this manuscript when $\mathbf{v}$ is a vector and $\mathbf{A}$ is a matrix; if you do, it's a typo.

2 A recurrent network of integrate and fire neurons

Our goal in this section is to determine how much information a network of leaky integrate and fire neurons contains about its input. The analysis for leaky integrate and fire neurons is, however, hard, so we consider a simpler neuron: one without a leak. For that we can compute information analytically, and we find that the information in large time windows is independent of the connectivity, and thus independent of the correlational structure. We apply our analysis to a network of leaky integrate and fire neurons, and we find that it works well, in the sense that information in long time windows is very weakly dependent on both connectivity and correlations.

2.1 The non-leaky integrate and fire neuron

Here we compute the amount of information in a network of N non-leaky integrate and fire neurons. Our starting point is the time evolution equation for the membrane potential, denoted V_i. For the non-leaky integrate and fire neuron, this is given by

$$\frac{dV_i}{dt} = \sum_j J_{ij} \sum_l h_{ij}(t - t_j^l) + g_i(s) + \sum_k M_{ik}\xi_k(t)\,. \tag{4}$$

Here J_{ij} is the connectivity matrix (so the sum over j runs from 1 to N), t_j^l is the time of the l^{th} spike on neuron j, $h_{ij}(t-t_j^l)$ is the synaptic response to that spike (its properties are given in Eq. (9) and preceding text, and the form for $h_{ij}(t)$ that we use in our simulations is given in Eqs. (8b) and (41)), $g_i(s)$ is the mean synaptic drive, which depends on a (potentially multi-dimensional, but time-independent) stimulus, s, and $\xi_k(t)$ is time dependent Gaussian noise,

$$\langle \xi_k(t)\xi_{k'}(t')\rangle = \delta_{kk'}C(t-t')\,. \tag{5}$$

The autocorrelation function, $C(\tau)$, is chosen to integrate to 1,

$$\int_{-\infty}^{\infty} d\tau\, C(\tau) = 1\,, \tag{6}$$

and $\delta_{kk'}$ is the Kronecker delta. The last term in Eq. (4) consists of both independent and

shared noise; the latter is taken into account by letting $M_{ik} = \sigma_{ind}\delta_{ik} + \sigma_s\delta_{k0}$. Consequently,

$$\sum_k M_{ik}\xi_k(t) = \sigma_{ind}\xi_i(t) + \sigma_s\xi_0(t)\,. \tag{7}$$

The neuron emits a spike when the voltage reaches a threshold, denoted θ_i for neuron i, after which the voltage is reset to V_r, which we take to be 0. To take care of the reset, we introduce a negative self-current,

$$J_{ii} = -\theta_i \tag{8a}$$

$$h_{ii}(t) = \delta(t) \tag{8b}$$

where $\delta(t)$ is the Dirac delta function.

When $i \neq j$, $h_{ij}(t)$ corresponds to a brief current pulse; it is zero when $t < 0$ and, for convenience we choose it so that it integrates to 1,

$$\int_0^\infty dt\, h_{ij}(t) = 1\,. \tag{9}$$

This gives $h_{ij}(t)$ units of inverse time, and so J_{ij} has units of voltage. For now we take J_{ij} to be an arbitrary matrix. In our simulations, however, we consider excitatory-inhibitory networks.

For this model, computing spike count in a window of size T is straightforward: simply integrate both sides of Eq. (4) from 0 to T. This gives

$$\Delta V_i(T) = \sum_j J_{ij}\int_0^T dt\, \sum_l h_{ij}(t - t_j^l) + Tg_i(s) + \sum_k M_{ik}\int_0^T dt\, \xi_k(t) \tag{10}$$

where $\Delta V_i(T) \equiv V_i(T) - V_i(0)$. The first integral is approximately the spike count at time T, denoted $n_j(T)$ for neuron j,

$$\int_0^T dt\, \sum_l h_{ij}(t - t_j^l) = n_j(T) + \delta n_j(T)\,. \tag{11}$$

The term $\delta n_j(t)$ comes from the fact that $h_{ij}(t)$ has finite width in time (if $i \neq j$). Note that δn_j is at most 1. The mean of the second integral is zero, and, using Eq. (5), its variance is given by

$$\mathrm{Var}\left[\int_0^T dt\, \xi_k(t)\right] = \int_0^T dt\, \int_0^T dt'\, C(t - t')\,. \tag{12}$$

If the correlation time is zero (corresponding to delta-correlated white noise), the double integral on the right is just T. However, for finite correlation time there is a small reduction due to edge effects. To take that into account, we write

$$\mathrm{Var}\left[\int_0^T dt\, \xi_k(t)\right] = T_c \tag{13}$$

where T_c is smaller than T by about the correlation time. Thus,

$$\int_0^T dt\, \xi_k(t) = T_c^{1/2}\eta_k \tag{14}$$

where the η_k are a set of uncorrelated, zero mean, unit variance Gaussian random variables,

$$\langle \eta_k \eta_{k'} \rangle = \delta_{kk'}\,. \tag{15}$$

Inserting Eqs. (11) and (14) into Eq. (10), we have

$$0 = \sum_j J_{ij} n_j(T) + T g_i(s) + T_c^{1/2} \sum_k M_{ik}\eta_k + \gamma_i(T) \tag{16}$$

and the noise term, γ_i, given by

$$\gamma_i(T) \equiv \sum_j J_{ij}\delta n_j(T) - \Delta V_i(T)\,. \tag{17}$$

How big is $\gamma_i(T)$? In the large T limit there are typically a large number of spikes, so δn_j is small compared to n_j. Moreover, ΔV_i is at most θ_i. Consequently, in this limit, $\gamma_i(T)$ is usually small compared to T. However, it isn't *always* small: although ΔV_i can be at most θ_i, it can be negative, and it can be large and negative. This happens whenever a neuron receives a consistently negative current, in which case $\Delta V_i(T)$ is proportional to $-T$. We can partially solve this problem by simply ignoring any neuron that doesn't spike. Unfortunately, this is only a partial solution, because it's possible for a neuron to fire once or twice due to noise, and then for the voltage to steadily decrease with time; this would again mean $\Delta V_i \propto -T$ in the large T limit. This can be taken care of by ignoring the transients, which we effectively do in our simulations: we run one very long simulation, divide them into intervals of either 2 or 10 s, and collect spike counts in those intervals.

Assuming the connectivity matrix, J_{ij}, is invertible, we can solve Eq. (16) directly for spike count. To simplify notation we switch to vectors and matrices, for which we use bold

font. Multiplying both sides of Eq. (16) by $\mathbf{J}^{-1}$, we have

$$\mathbf{n} = -T\mathbf{J}^{-1}\mathbf{g}(s) - \mathbf{J}^{-1}\left(\, T_c^{1/2}\mathbf{M}\boldsymbol{\eta} + \boldsymbol{\gamma}(T)\right) . \tag{18}$$

The mean and covariance of the spike count are, therefore, given by

$$\text{Mean}[\mathbf{n}] = -\mathbf{J}^{-1}\big[T\mathbf{g}(s) + \langle\boldsymbol{\gamma}\rangle\big] \tag{19a}$$

$$\text{Covar}[\mathbf{n}] = T_c\mathbf{J}^{-1}\left[\mathbf{M}\mathbf{M}^T + T_c^{-1/2}(\mathbf{M}\langle\boldsymbol{\eta}\boldsymbol{\gamma}^T\rangle + \langle\boldsymbol{\gamma}\boldsymbol{\eta}\rangle^T\mathbf{M}^T) + T_c^{-1}\langle\boldsymbol{\gamma}\boldsymbol{\gamma}^T\rangle\right]\mathbf{J}^{-T} \tag{19b}$$

where the superscript $-T$ denotes transpose and inverse, and, for clarity, we suppress the fact that $\boldsymbol{\eta}$ and $\boldsymbol{\gamma}$ depend on T.

2.2 Linear Fisher information

Our next step is to compute the linear Fisher information, both in the input and in the output. In general, for a random variable with stimulus-dependent mean $\mathbf{f}(s)$ and covariance matrix $\boldsymbol{\Sigma}(s)$, the linear Fisher information is given by [1]

$$I = \mathbf{f}'(s)^T\boldsymbol{\Sigma}^{-1}(s)\mathbf{f}'(s) \tag{20}$$

where a prime denotes a derivative with respect to s (if s were a vector, I would be the Fisher information matrix, but here we restrict ourselves to scalars). Using Eq. (19), and working to lowest non-vanishing order in $1/T^{1/2}$, we find that the linear Fisher information in the spike train, denoted I_{out}, is given by

$$I_{\text{out}} = T\ \mathbf{g}'(s)^T\left[\mathbf{M}\mathbf{M}^T + \mathcal{O}(T^{-1/2})\right]^{-1}\mathbf{g}'(s)\,. \tag{21}$$

How does this compare to the information in the input? The latter is the information available to an observer that has direct access to the current, $\mathbf{g}(s) + \mathbf{M}\boldsymbol{\xi}$. After observing this current for time T, the mean is $T\mathbf{g}(s)$ and, to lowest nonvanishing order in $1/T$, the variance is $T\mathbf{M}\mathbf{M}^T$. Thus, using Eq. (20), the Fisher information in the input, denoted I_{in}, is given by

$$I_{\text{in}} = T\ \mathbf{g}'(s)^T\left[\mathbf{M}\mathbf{M}^T\right]^{-1}\mathbf{g}'(s)\,. \tag{22}$$

This is almost the same as the Fisher information in the output, I_{out}. The difference is that there is extra noise, captured by the $\mathcal{O}(T^{-1/2})$ correction, associated with spikes. Essentially, for small times, spikes inject additional noise into the estimate of the mean, but as time increases, that noise diminishes. Note, though, that in general there is one more potential loss of information: if some neurons never spike, they need to be taken out of the network. See, for example, the comments following Eq. (26).

Here, and in the simulations, we consider a simple model in which the noise is given in Eq. (7), so that

$$\mathbf{M}\mathbf{M}^T = \sigma_{ind}^2 \mathbf{I} + \sigma_s^2 \mathbf{1}\mathbf{1}^T \tag{23}$$

where $\mathbf{I}$ is the identity matrix, $\mathbf{1}$ is a vector consisting of all 1's (so $\mathbf{1}\mathbf{1}^T$ is a matrix with all elements equal to 1), and the signal, $\mathbf{g}(s)$, is given by

$$\mathbf{g}(s) = s\mathbf{1}\,. \tag{24}$$

In this case, as is straightforward to show, the input information is given by

$$I_{in} = \frac{T}{\sigma_s^2 + \sigma_{ind}^2/N}\,. \tag{25}$$

The output information, I_{out}, is only slightly smaller,

$$I_{out} = \frac{T}{\sigma_s^2 + \sigma_{ind}^2/N + \mathcal{O}(T^{-1/2})}\,. \tag{26}$$

Note that here N should really be the number of neurons that fire. However, assuming that number is large, it doesn't really matter what it is – the Fisher information is determined mainly by the shared noise, σ_s^2

Equations (25) and (26) correspond to Eqs. (1) and (2) of the main text, except that in the main text we report information rates, so we divide by T.

2.3 Correlation coefficients

The output information, Eq. (26), is independent of the network parameters. But what about the correlation coefficients – do they depend on network parameters? If so, that would mean the information is independent of the correlational structure in the network. Typically it is

not easy to compute correlation coefficients analytically, but for the non-leaky integrate and fire neuron with the noise models given in (23), we can. Essentially, we compute the right hand side of Eq. (19b). We work to lowest nonvanishing order in T, so the quantity we are interested in is

$$\mathbf{\Sigma} \equiv \mathbf{J}^{-1} \left[\sigma_{ind}^2 \mathbf{I} + \sigma_s^2 \mathbf{1}\mathbf{1}^T \right] \mathbf{J}^{-T} \tag{27}$$

where we used Eq. (23) for $\mathbf{M}\mathbf{M}^T$, and $\mathbf{J}^{-T} \equiv (\mathbf{J}^{-1})^T$ (when T appears as a superscript, it means transpose, not observation time window).

For the weight matrix, $\mathbf{J}$, we use the homogeneous model given in Eq. (40) below (with $\theta_i = 1$; see Table 1, located at the end of this section), for which we may write

$$\mathbf{J} = -\left(1 + \frac{J_{EE}}{N_E - 1}\right) \mathbf{I}_E - \left(1 + \frac{J_{II}}{N_I - 1}\right) \mathbf{I}_I + \left(\begin{array}{c|c} \frac{J_{EE}}{N_E - 1} & \frac{J_{EI}}{N_I} \\ \hline \frac{J_{IE}}{N_E} & \frac{J_{II}}{N_I - 1} \end{array}\right) . \tag{28}$$

Here $\mathbf{I}_E$ and $\mathbf{I}_E$ are identity matrices in the excitatory and inhibitory subspaces, respectively,

$$(\mathbf{I}_E)_{ij} = \begin{cases} \delta_{ij} & 1 \le i, j \le N_E \\ 0 & \text{otherwise} \end{cases} \tag{29a}$$

$$(\mathbf{I}_I)_{ij} = \begin{cases} \delta_{ij} & N_E < i, j \le N \\ 0 & \text{otherwise}\,, \end{cases} \tag{29b}$$

and the last term in Eq. (28) – the matrix with the vertical and horizontal lines – is an $N \times N$ matrix consisting of four sub-matrices of sizes $N_E \times N_E$ (upper left), $N_E \times N_I$ (upper right), $N_I \times N_E$ (lower left), and $N_I \times N_I$ (lower right). All the elements of each of the sub-matrices are the same, and equal to the value specified in the corresponding quadrant.

The inverse of $\mathbf{J}$ is given by

$$\mathbf{J}^{-1} = -\frac{\mathbf{I}_E - \frac{\mathbf{1}_E \mathbf{1}_E^T}{N_E}}{1 + \frac{J_{EE}}{N_E - 1}} - \frac{\mathbf{I}_I - \frac{\mathbf{1}_I \mathbf{1}_I^T}{N_I}}{1 + \frac{J_{II}}{N_I - 1}} + \frac{1}{D} \left(\begin{array}{c|c} \frac{J_{II} - 1}{N_E} & -\frac{J_{EI}}{N_I} \\ \hline -\frac{J_{IE}}{N_E} & \frac{J_{EE} - 1}{N_I} \end{array}\right) \tag{30}$$

where D is closely related to the determinant of the last term in Eq. (28),

$$D \equiv (J_{EE} - 1)(J_{II} - 1) - J_{EI} J_{IE} \tag{31}$$

and $\mathbf{1}_E$ and $\mathbf{1}_I$ are vectors of 1s in the excitatory and inhibitory spaces, respectively,

$$(\mathbf{1}_E)_i = \begin{cases} 1 & 1 \le i \le N_E \\ 0 & \text{otherwise} \end{cases} \tag{32a}$$

$$(\mathbf{1}_I)_i = \begin{cases} 1 & N_E < i \le N \\ 0 & \text{otherwise}\,. \end{cases} \tag{32b}$$

It is relatively straightforward to show that $\mathbf{J}^{-1}$ really is the inverse of $\mathbf{J}$. For that it is helpful to notice that the first two terms on the right hand side of Eq. (30) are orthogonal to both $\mathbf{1}_E$ and $\mathbf{1}_I$, and that

$$D^{-1}\mathbf{J}\left(\begin{array}{c|c} \frac{J_{II}-1}{N_E} & -\frac{J_{EI}}{N_I} \\ \hline -\frac{J_{IE}}{N_E} & \frac{J_{EE}-1}{N_I} \end{array}\right) = \left(\begin{array}{c|c} \frac{1}{N_E} & 0 \\ \hline 0 & \frac{1}{N_I} \end{array}\right) = \frac{\mathbf{1}_E\mathbf{1}_E^T}{N_E} + \frac{\mathbf{1}_I\mathbf{1}_I^T}{N_I}\,. \tag{33}$$

We can now insert Eq. (30) into (27) to compute the covariance matrix, $\mathbf{\Sigma}$. There are two components to this matrix, $\sigma_{ind}^2\mathbf{J}^{-1}\mathbf{J}^{-T}$ and $\sigma_s^2\mathbf{J}^{-1}\mathbf{1}\mathbf{1}^T\mathbf{J}^{-T}$. We treat these one at a time, starting with the first,

$$\begin{aligned}\mathbf{J}^{-1}\mathbf{J}^{-T} = &\frac{\mathbf{I}_E - \frac{\mathbf{1}_E\mathbf{1}_E^T}{N_E}}{(1+\frac{J_{EE}}{N_E-1})^2} + \frac{\mathbf{I}_I - \frac{\mathbf{1}_I\mathbf{1}_I^T}{N_I}}{(1+\frac{J_{II}}{N_I-1})^2} \\ &+ \frac{1}{D^2}\left(\begin{array}{c|c} \frac{(J_{II}-1)^2}{N_E} + \frac{J_{EI}^2}{N_I} & -\frac{(J_{II}-1)J_{IE}}{N_E} - \frac{(J_{EE}-1)J_{EI}}{N_I} \\ \hline -\frac{(J_{II}-1)J_{IE}}{N_E} - \frac{(J_{EE}-1)J_{EI}}{N_I} & \frac{(J_{EE}-1)^2}{N_I} + \frac{J_{IE}^2}{N_E} \end{array}\right).\end{aligned} \tag{34}$$

In the large N limit, $\mathbf{J}^{-1}\mathbf{J}^{-T}$ simplifies considerably: it's just $\mathbf{I}_E + \mathbf{I}_I$, which in turn is the identity matrix, $\mathbf{I}$; we use that approximation here.

The second component of $\mathbf{\Sigma}$ is

$$\mathbf{J}^{-1}\mathbf{1}\mathbf{1}^T\mathbf{J}^{-T} = \left(\begin{array}{c|c} Q_{EE} & Q_{EI} \\ \hline Q_{IE} & Q_{II} \end{array}\right) \tag{35}$$

where

$$Q_{EE} \equiv \frac{(J_{II} - J_{EI} - 1)^2}{D^2} \tag{36a}$$

$$Q_{EI} = Q_{IE} \equiv \frac{(J_{II} - J_{EI} - 1)(J_{EE} - J_{IE} - 1)}{D^2} \tag{36b}$$

$$Q_{II} \equiv \frac{(J_{EE} - J_{IE} - 1)^2}{D^2}\,. \tag{36c}$$

With the approximation that $\mathbf{J}^{-1}\mathbf{J}^{-T} = \mathbf{I}$, $\mathbf{\Sigma}$ has a very simple form

$$\mathbf{\Sigma} = \sigma_{ind}^2 \mathbf{I} + \sigma_s^2 \left(\begin{array}{c|c} Q_{EE} & Q_{EI} \\ \hline Q_{IE} & Q_{II} \end{array} \right) . \tag{37}$$

An outcome of this analysis is that the covariance matrix, and thus the correlation coefficients, depend only on neuron type. Let us use ρ_{KL} to denote the correlation coefficient between neurons of type K and type L. Examining Eq. (37), we see that

$$\rho_{KL} = \frac{\sigma_s^2 Q_{KL}}{\left((\sigma_{ind}^2 + \sigma_s^2 Q_{KK})(\sigma_{ind}^2 + \sigma_s^2 Q_{LL})\right)^{1/2}} . \tag{38}$$

This equation is used to make the solid lines in Fig. 2b of the main text.

2.4 Simulations – recurrent network

Below we report the parameters we use in our simulations (summarized in Table 1 below), the method for estimating Fisher information, and the numerical details.

Simulations were performed with leaky, rather than non-leaky, integrate and fire neurons, as they are much more realistic. That introduces a modification to Eq. (4): a leak was added to the time evolution equation for the membrane potential,

$$\frac{dV_i}{dt} = -\frac{V_i}{\tau_m} + \sum_j J_{ij} \sum_l h_{ij}(t - t_j^l) + g_i(s) + \sum_k M_{ik} \xi_k(t) . \tag{39}$$

2.4.1 Parameters

For all simulation, 80% of the neurons were excitatory and 20% were inhibitory. Connectivity was all-to-all (except that there were no autapses), and connection strength depended only on neuron type. Using N_E and N_I for the number of excitatory and inhibitory neurons, respectively, the connection strengths between individual neurons (with $i \neq j$) were

$$J_{ij} = \begin{cases} J_{EE}/(N_E - 1) & 1 \leq i \leq N_E; \quad 1 \leq j \leq N_E \\ J_{EI}/N_I & 1 \leq i \leq N_E; \quad N_E < j \leq N \\ J_{IE}/N_E & N_E < i \leq N; \quad 1 \leq j \leq N_E \\ J_{II}/(N_I - 1) & N_E < i \leq N; \quad N_E < j \leq N . \end{cases} \tag{40}$$

If $i = j$, $J_{ii} = -\theta_i$, as in Eq. (8a). Note the slight abuse of notation: capital letter subscripts on J mean something different than lower case subscripts. For the current, $h_{ij}(t)$ with $i \neq j$, we used a decaying exponential,

$$h_{ij}(t) = \Theta(t)\,\frac{e^{-t/\tau_K}}{\tau_K} \tag{41}$$

where K is either E (excitatory) or I (inhibitory) and $\Theta(t)$ is the Heaviside step function. The above expression applies if $i \neq j$; if $i = j$, $h_{ij}(t)$ is a delta function. This implements the voltage reset when the voltage reaches threshold (see Eq. (8b)).

Parameter	Value	Description
N	75, 125, 250, 500	number of neurons
N_E	$0.8N$	number of excitatory neurons
N_I	$0.2N$	number of inhibitory neurons
T	2, 10 s	observation time window for leaky and non-leaky integrate and fire networks, respectively
τ_m	20 ms	membrane time constant
τ_E	2 ms	excitatory synaptic time constant
τ_I	3 ms	inhibitory synaptic time constant
θ_i	1	threshold
V_r	0	reset
σ^2_{ind}	76.5 s^{-1}	variance of the independent noise
σ^2_s	3.5 s^{-1}	variance of the shared noise
J_{EE}	6	$E \to E$ connection strength
J_{EI}	-9.5	$I \to E$ connection strengths
J_{IE}	(7.6, 5.9, 5.4)	$E \to I$ connection strengths (low, medium, high)
J_{II}	(-11.2, -9.4, -8.9)	$I \to I$ connection strength (low, medium, high)

Table 1. Simulation parameters for the recurrent network (parameters for the feedforward network are given in Sec. 3.3). None of the results depend critically on the values of the parameters.

For all simulations we used the input given in Eq. (24). The noise model was the one given in Eq. (7), and the noise was delta-correlated, so that $T_c = T$ (see Eq. (13)). We used three different sets of connection strengths (low, medium and high), designed to provide three different average correlation coefficients. They correspond to average correlation coefficients of 0.013, 0.047 and 0.108, respectively, when $N = 500$. All parameter are listed in Table 1.

2.4.2 Estimating linear Fisher information

To compare the information in the output of the network to that in the input, we need to estimate the Fisher information from the spike trains. For that we use a locally optimal linear estimator. In general, for a random variable $\mathbf{r}$ with stimulus-dependent mean $\mathbf{f}(s)$ and covariance $\mathbf{\Sigma}(s)$, that estimator, taken near a stimulus value of s_0, is given by [2]

$$\hat{s} = s_0 + \frac{\mathbf{f}'(s_0)^T \mathbf{\Sigma}^{-1}(s_0)\big(\mathbf{r} - \mathbf{f}(s_0)\big)}{\mathbf{f}'(s_0)^T \mathbf{\Sigma}^{-1}(s_0)\mathbf{f}'(s_0)}\,. \tag{42}$$

Applying this equation to our problem, for which the mean and covariance are given in Eq. (19), and ignoring corrections associated with finite observation time, T, we have

$$\hat{s} = s_0 + \frac{\mathbf{g}'(s_0)^T \big(\mathbf{M}\mathbf{M}^T\big)^{-1}\big[-\mathbf{J}\boldsymbol{\nu} - \mathbf{g}(s_0)\big]}{\mathbf{g}'(s_0)^T \big(\mathbf{M}\mathbf{M}^T\big)^{-1}\mathbf{g}'(s_0)} \tag{43}$$

where $\boldsymbol{\nu}$ is a vector of firing rates: $\boldsymbol{\nu} = (n_1/T, n_2/T, ..., n_N/T)$. Specializing to the form of $\mathbf{g}(s)$ given in Eq. (24), and using the noise model given in Eq. (7), for which $\mathbf{M}\mathbf{M}^T$ is given in Eq. (23), this simplifies to

$$\hat{s} = -\frac{1}{N}\sum_{ij} J_{ij}\nu_j\,. \tag{44}$$

Using Eq. (40), and taking into account the fact that $J_{ii} = -\theta_i$, it is easy to show that

$$\sum_i J_{ij} = -\theta_j + \begin{cases} J_{EE} + J_{IE} & 1 \le j \le N_E \\ J_{EI} + J_{II} & N_E < j \le N\,. \end{cases} \tag{45}$$

The sum over j is now straightforward, and we arrive at

$$\hat{s} = \frac{1}{N}\sum_i \theta_i\nu_i - \frac{N_E}{N}\big(J_{EE} + J_{IE}\big)\nu_E - \frac{N_I}{N}\big(J_{EI} + J_{II}\big)\nu_I \tag{46}$$

where ν_E and ν_I are the population averaged excitatory and inhibitory firing rates,

$$\nu_E \equiv \frac{1}{N_E}\sum_{j=1}^{N_E}\nu_j \tag{47a}$$

$$\nu_I \equiv \frac{1}{N_I}\sum_{j=N_E+1}^{N}\nu_j\,. \tag{47b}$$

Finally, taking into account the fact that we use $\theta_i = 1$ in our simulations, which implies that $N^{-1}\sum_i \theta_i\nu_i = N_E\nu_E/N + N_I\nu_I/N$, the expression for $\hat{s}$ reduces even further,

$$\hat{s} = \frac{N_E}{N}\left[1-\left(J_{EE}+J_{IE}\right)\right]\nu_E + \frac{N_I}{N}\left[1-\left(J_{EI}+J_{II}\right)\right]\nu_I\,. \tag{48}$$

This estimator applies to the non-leaky integrate and fire network. Because it is optimal and unbiased, we can take the Fisher information to be the inverse of the variance of $\hat{s}$.

For the leaky integrate and fire neuron, Eq. (48) is not necessarily the correct estimator. However, it is likely to have the correct form. We thus assume that the estimator for this neuron is linear in the average excitatory and inhibitory firing rates, but with unknown coefficients,

$$\hat{s}_{LIF}(w) = (1-w)\nu_E + w\nu_I\,. \tag{49}$$

Ultimately we will optimize this estimate with respect to the weight, w. First, though, we note that it is likely to be biased. However, we can correct for the bias [3], and at the same time compute the Fisher information, denoted $I(w)$, using the equation

$$I(w) = \frac{[d\langle\hat{s}_{LIF}(w)\rangle_s/ds]^2}{\mathrm{Var}[\hat{s}_{LIF}(w)]_s}\,, \tag{50}$$

where the angle brackets indicate an average over trials. The subscript s on both the angle brackets and the variance indicates that the presented stimulus was s (which should be near s_0, since this is a *locally* optimal linear estimator).

In our simulations, we estimated the derivative numerically by presenting two different, but nearby, values of s (denoted s_1 and s_2, whose average is s_0), and averaging the variance over those values. Thus, the Fisher information was estimated via

$$I(w) = \left[\frac{\langle\hat{s}_{LIF}(w)\rangle_{s_1} - \langle\hat{s}_{LIF}(w)\rangle_{s_2}}{s_1-s_2}\right]^2 \frac{2}{\mathrm{Var}[\hat{s}_{LIF}(w)]_{s_1} + \mathrm{Var}[\hat{s}_{LIF}(w)]_{s_2}}\,. \tag{51}$$

This estimate depends on the choice of the weight, w. To find the optimal weight, we used cross-validation: for each simulation, we created a training and a test set; the information, $I(w)$, was maximized on the training test, resulting in an optimal weight, w_{opt}; the reported information was then $I(w_{opt})$, evaluated on the test set.

2.5 Effect of observation time on information

In the main text we showed that information in the spike trains for the non-leaky integrate and fire network is almost identical to input information for all network sizes (see open circles in Fig. 2d, main text). This is expected to occur, however, only for large observation time, T. What happens when T isn't so large? To answer that, we plot, in Supplementary Fig. 1, information per unit time versus network size, N, for T ranging from 100 ms to 10 seconds. Not surprisingly, the shorter the time window, the smaller the information. However, the time window becomes less and less important as the number of neurons increases.

2.6 Numerical details

Simulations were performed using custom C code, with the parameters given in Table 1. For each set of parameters, simulations were run for 12000 seconds with a one-step Euler method and a time step of 0.01 ms.

To estimate the linear Fisher information for the leaky integrate and fire network, we used Eq. (51) with $s_0 = 182.5$ Hz, $s_1 = 180$ Hz and $s_2 = 185$ Hz. Specifically, we simulated the network for 12000 seconds with $s = s_1$ and 12000 seconds with $s = s_2$, and divided both runs into 6000 two second intervals (corresponding to an observation time window, T, of 2 seconds). The 6000 intervals were randomly split into two halves to create a training and a test set; this procedure was iterated ten times. For each iteration, information was optimized on the training test and estimated, using Eq. (51), on the test set. The mean values were plotted in Fig. 2d, with error bars given by the standard error of the mean.

To estimate the linear Fisher information for the non-leaky integrate and fire network, we use the inverse of variance of the estimator given in Eq. (48). Simulations were run for 12000 seconds and were divided into 1200 ten second intervals (corresponding to an observation

time window, T, of 10 seconds). The mean values were plotted in Fig. 2d of the main text, with error bars corresponding to the standard error of the mean computed from 20 random subgroups of intervals.

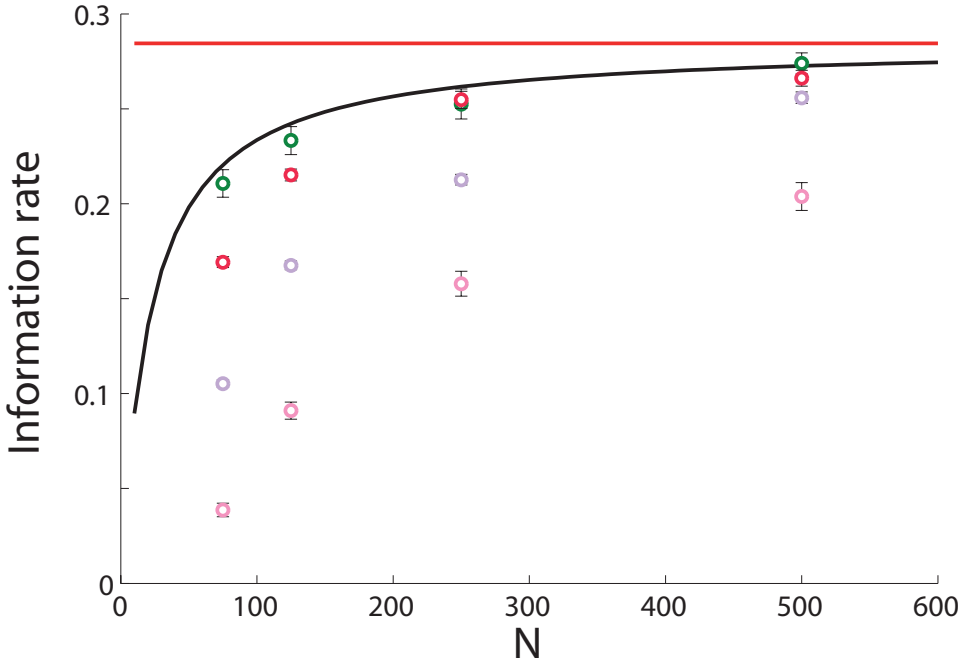

Figure 1: Effect of observation time. Output information for a non-leaky integrate and fire network as a function of network size, N for several values of the observation time, T: 100 ms (pink), 500 ms (purple), 2 seconds (red), and 10 seconds (green). The black solid line is the information in the input, Eq. (25). Parameters correspond to those of the network with high correlations (Table 1 with $J_{IE} = 5.4$ and $J_{II} = -8.9$); this is the same network we used to make Fig. 2d of the main text. The open green circles here correspond to the ones in Fig. 2d of the main text; slight differences between the open green circles in this figure and in Fig. 2d are because we used a different seed in the random number generator in the two sets of simulations; those differences are within error bars. Small observation times reduce information, but the effect diminishes as the number of neurons increases.

3 A feedforward network of non-leaky integrate and fire neurons

As in Sec. 2, we study a network of N non-leaky integrate and fire neurons with current based synapses, driven by a stimulus-dependent input corrupted by Gaussian noise. The main difference is that the network is purely feedforward rather than recurrent. Again, though, our goal is to compute how much information about the stimulus there is in the spike trains. We start by describing the network; we then compute the linear Fisher information. We end with details of the numerical simulations.

3.1 The model

The network consists of N input "neurons" (quotes because we modeled them as white noise) and N output neurons. The membrane potential of the i^{th} output neuron evolves according to

$$\frac{dV_i}{dt} = \frac{J_0}{N^{1/2}} \sum_{j=1}^{N} W_{ij}\big(s + \xi_j(t)\big) + g_{0i} \tag{52}$$

where s is the stimulus, g_{0i} is an offset drive to ensure that the mean output firing rate was 50 Hz, $\xi_j(t)$ is independent white noise (Eq. (5) with $C(t-t') = \delta(t-t')$; note that the noise is independent across neurons as well as time), and W_{ij} is a binary connectivity matrix,

$$W_{ij} = \begin{cases} 1 & \text{with probability } f \\ 0 & \text{with probability } 1-f\,. \end{cases} \tag{53}$$

Integrating Eq. (52) with respect to t from 0 to T (as in Eq. (10)), and keeping terms only up to $\mathcal{O}(T^{1/2})$, we can derive an equation for $\mathbf{n}$, the vector of spike counts,

$$\mathbf{n} = \frac{J_0}{N^{1/2}} \left[Ts\mathbf{W1} + T^{1/2}\mathbf{W}\boldsymbol{\eta}\right] + T\mathbf{g}_0 \tag{54}$$

where $\boldsymbol{\eta}$ is uncorrelated, zero mean, unit variance Gaussian noise, as in Eq. (15): $\langle\boldsymbol{\eta}\rangle = 0$ and $\langle\boldsymbol{\eta}\boldsymbol{\eta}^T\rangle = \mathbf{I}$. The mean and covariance of the spike count are, then, given by

$$\text{Mean}[\mathbf{n}] = \frac{J_0}{N^{1/2}} Ts\mathbf{W1} + T\mathbf{g}_0 \tag{55a}$$

$$\text{Covar}[\mathbf{n}] = \frac{J_0^2}{N}\, T\mathbf{W}\mathbf{W}^T\,. \tag{55b}$$

3.2 Linear Fisher information

Using Eq. (20) for the Fisher information, denoted I_{ff} (for feedforward information), we have

$$I_{ff} = \mathbf{1}^T \mathbf{W}^T (\mathbf{W}\mathbf{W}^T)^{-1} \mathbf{W} \mathbf{1} T \,. \tag{56}$$

When $\mathbf{W}$ is invertible, $I_{ff} = \mathbf{1}^T \mathbf{1} T = NT$, independent of the connectivity. This is exactly the information in the input, so for the invertible case there is no information loss – at least not in the large T limit.

To estimate the linear Fisher information numerically, we computed the variance of a locally optimal linear estimator, and took the linear Fisher information to be its inverse. The form of the locally optimal linear estimator is given in Eq. (42); using Eq. (55) for the mean value of $\mathbf{n}$ (from which we can get $\mathbf{f}'$) and the covariance, and performing a small amount of algebra, we see that the locally optimal linear estimate of the stimulus is

$$\hat{s} = \frac{N^{1/2}}{J_0} \frac{\mathbf{1}^T \mathbf{W}^T (\mathbf{W}\mathbf{W}^T)^{-1} (\boldsymbol{\nu} - \mathbf{g}_0)}{\mathbf{1}^T \mathbf{W}^T (\mathbf{W}\mathbf{W}^T)^{-1} \mathbf{W} \mathbf{1}} \tag{57}$$

where, as above, $\boldsymbol{\nu} \equiv \mathbf{n}/T$ is a vector of firing rates. When $\mathbf{W}$ is invertible, this simplifies to

$$\hat{s} = \frac{1}{J_0 N^{1/2}} \mathbf{1}^T \mathbf{W}^{-1} (\boldsymbol{\nu} - \mathbf{g}_0) \,. \tag{58}$$

In our numerical simulations we did not assume that $\mathbf{W}$ was invertible, and so we used Eq. (57) to estimate the stimulus.

3.3 Simulations – feedforward network

For each set of parameters, simulations were run for 12000 seconds with a one-step Euler method and a time step of 0.01 ms. Runs were broken into 6000 two second intervals (corresponding to an observation time window, T, of 2 seconds); those intervals were used to compute the Fano factors and information. We took information to be the inverse of the variance of $\hat{s}$, computed via Eq. (57) at $s = 0$. The model has two parameters, J_0 and f; these were set to 20 and 0.1, respectively. We used 40, 80, 160 and 500 neurons. Results are shown in Fig. 4 of the main text.

4 Theoretical analysis of differential correlations

In this section we leave simulations, and turn to theoretical analysis of differential correlations. In Sec. 4.1 we analyze information when there is a "pure" $\mathbf{f}'\mathbf{f}'^T$ component and, just as importantly, when there is a not so pure component. We show that in the former case information saturates with N; in the latter case it doesn't. We also show, somewhat surprisingly, that the optimal decoder doesn't need to know about the $\mathbf{f}'\mathbf{f}'^T$ component of the correlations. In Sec. 4.2 we provide further insight into differential correlations by expressing them in terms of the eigenvectors and eigenvalues of the covariance matrix. Finally, in Sec. 4.3, we use that analysis to understand why, and when, it's hard to accurately estimate Fisher information.

4.1 Information for "pure" (and not so pure) $\mathbf{f}'\mathbf{f}'^T$ correlations

Here we ask how the linear Fisher information scales with the number of neurons, N, when the covariance matrix contains a "pure" $\mathbf{f}'\mathbf{f}'^T$ component, (the second term in Eq. (61) below). Our starting point is a covariance matrix, $\mathbf{\Sigma}_0(s)$, that doesn't necessarily contain an $\mathbf{f}'\mathbf{f}'^T$ component. As in Eq. (3) of the main text, the (linear) Fisher information associated with $\mathbf{\Sigma}_0(s)$, denoted I_0, is given by

$$I_0 = \mathbf{f}'(s)^T \mathbf{\Sigma}_0^{-1}(s) \mathbf{f}'(s)\,, \tag{59}$$

where, as usual, $\mathbf{f}(s)$ is a vector of tuning curves,

$$\mathbf{f}(s) \equiv (f_1(s), f_2(s), ..., f_N(s))^T\,, \tag{60}$$

and a prime denotes a derivative with respect to s. Note that the information also depends on stimulus, s; we suppress that dependence for clarity. To add a pure $\mathbf{f}'\mathbf{f}'^T$ component, we define a new covariance matrix, $\mathbf{\Sigma}_\epsilon(s)$, via

$$\mathbf{\Sigma}_\epsilon(s) = \mathbf{\Sigma}_0(s) + \epsilon \mathbf{f}'(s)\mathbf{f}'^T(s)\,. \tag{61}$$

The new information, denoted I_ϵ, is given by

$$I_\epsilon = \mathbf{f}'(s)^T \mathbf{\Sigma}_\epsilon^{-1}(s) \mathbf{f}'(s)\,. \tag{62}$$

To compute I_ϵ, we need the inverse of $\mathbf{\Sigma}_\epsilon$. As is easy to verify, this inverse is given by

$$\mathbf{\Sigma}_\epsilon^{-1}(s) = \mathbf{\Sigma}_0^{-1}(s) - \frac{\epsilon}{1+\epsilon I_0}\mathbf{\Sigma}_0^{-1}(s)\mathbf{f}'(s)\mathbf{f}'^T(s)\mathbf{\Sigma}_0^{-1}(s)\,. \tag{63}$$

Inserting Eq. (63) into (62), we arrive at

$$I_\epsilon = I_0 - \frac{\epsilon I_0^2}{1+\epsilon I_0} = \frac{I_0}{1+\epsilon I_0}\,. \tag{64}$$

This is Eq. (5) of the main text.

Perhaps surprisingly, although $\mathbf{f}'\mathbf{f}'^T$ correlations play a critical role in determining information, they are irrelevant for decoding, in the sense that they have no effect on the locally optimal linear estimator. To see this explicitly, note first of all that the locally optimal linear estimator, denoted $\mathbf{w}^T$, generates an estimate of the stimulus near some particular value, s_0, by linearly operating on neural activity,

$$\hat{s} = s_0 + \mathbf{w}^T\big(\mathbf{r} - \mathbf{f}(s_0)\big)\,. \tag{65}$$

As in Eq. (42), in the presence of the covariance matrix given in Eq. (61), the optimal weight, $\mathbf{w}_{opt}^T$ is given by

$$\mathbf{w}_{opt}^T = \frac{\mathbf{f}'^T\big(\mathbf{\Sigma}_0 + \epsilon\mathbf{f}'\mathbf{f}'^T\big)^{-1}}{\mathbf{f}'^T\big(\mathbf{\Sigma}_0 + \epsilon\mathbf{f}'\mathbf{f}'^T\big)^{-1}\mathbf{f}'} \tag{66}$$

where we have dropped, for clarity, the explicit dependence on s_0. Using Eq. (63), this reduces to

$$\mathbf{w}_{opt}^T = \frac{\mathbf{f}'^T\mathbf{\Sigma}_0^{-1}}{\mathbf{f}'^T\mathbf{\Sigma}_0^{-1}\mathbf{f}'}\,. \tag{67}$$

Thus, the locally optimal linear decoder does not need to know the size of the $\mathbf{f}'\mathbf{f}'^T$ correlations.

In hindsight this makes sense: $\mathbf{f}'\mathbf{f}'^T$ correlations shift the hill of activity, and there is, quite literally, nothing any decoder can do about this. This suggests that these correlations are in some sense special. To determine just how special, we ask what happens when we add correlations in a different direction – say correlations of the form $\mathbf{u}\mathbf{u}^T$, where $\mathbf{u}$ is not parallel to $\mathbf{f}'$. In that case, the covariance matrix becomes (with a normalization added for convenience only)

$$\mathbf{\Sigma}_u(s) = \mathbf{\Sigma}_0(s) + \epsilon\,\frac{\mathbf{f}'(s)^T\mathbf{\Sigma}_0^{-1}(s)\mathbf{f}'(s)}{\mathbf{u}^T\mathbf{\Sigma}_0^{-1}(s)\mathbf{u}}\,\mathbf{u}\mathbf{u}^T\,. \tag{68}$$

Repeating the steps leading to Eq. (64), we find that

$$I_u \equiv \mathbf{f}'(s)^T \mathbf{\Sigma}_u^{-1}(s) \mathbf{f}'(s) = I_0 \sin^2\theta + \frac{I_0 \cos^2\theta}{1+\epsilon I_0}\,, \tag{69}$$

where I_0 is defined in Eq. (59) and

$$\cos\theta \equiv \frac{\mathbf{f}'(s)^T \mathbf{\Sigma}_0^{-1}(s) \mathbf{u}}{\left[\mathbf{f}'(s)^T \mathbf{\Sigma}_0^{-1}(s) \mathbf{f}'(s)\, \mathbf{u}^T \mathbf{\Sigma}_0^{-1}(s) \mathbf{u}\right]^{1/2}}\,. \tag{70}$$

Whenever $\theta \neq 0$ – meaning $\mathbf{u}$ is not parallel to $\mathbf{f}'(s)$ – information does not saturate as N goes to infinity. Thus, in the large N limit, $\mathbf{f}'(s)\mathbf{f}'(s)^T$ correlations are the only ones that cause saturation.

4.2 What does "Information limiting correlations" really mean?

To understand what it means for correlations to be information limiting, it is useful to express the information in terms of the eigenvectors and eigenvalues of the covariance matrix. These are denoted $\mathbf{v}_k$ and σ_k^2, respectively, and defined according to

$$\mathbf{\Sigma} \mathbf{v}_k = \sigma_k^2 \mathbf{v}_k\,. \tag{71}$$

(Here we suppress all s-dependence for clarity.) For convenience, we'll use an orthonormal basis: $\mathbf{v}_k^T \mathbf{v}_l = \delta_{kl}$. In this basis, the covariance matrix is given in terms of the $\mathbf{v}_k$ and σ_k^2 by

$$\mathbf{\Sigma} = \sum_k \sigma_k^2 \mathbf{v}_k \mathbf{v}_k^T\,, \tag{72}$$

and its inverse by

$$\mathbf{\Sigma}^{-1} = \sum_k \frac{\mathbf{v}_k \mathbf{v}_k^T}{\sigma_k^2}\,. \tag{73}$$

The square roots of the eigenvalues, σ_k, have a natural interpretation: they are the lengths of the principle axes of the covariance ellipse.

Using Eq. (73), the Fisher information may be written

$$I = \mathbf{f}'^T \mathbf{f}' \sum_k \frac{\cos^2\theta_k}{\sigma_k^2} \tag{74}$$

where

$$\cos^2\theta_k \equiv \frac{(\mathbf{v}_k^T\mathbf{f}')^2}{\mathbf{f}'^T\mathbf{f}'} \tag{75}$$

is the square of the cosine of the angle between $\mathbf{v}_k$ and $\mathbf{f}'$. Because of the completeness of $\mathbf{v}_k$ – that is, because $\sum_k \mathbf{v}_k\mathbf{v}_k^T = \mathbf{I}$, the identity matrix – it follows that

$$\sum_k \cos^2\theta_k = 1\,. \tag{76}$$

To make sense of Eq. (74), we need several facts about how the various terms depend on N. First, because $\mathbf{f}'$ has N components, $\mathbf{f}'^T\mathbf{f}'$ is proportional to N. This has an immediate and important consequence: if even one of the terms $\cos^2\theta_k/\sigma_k^2$ is $\mathcal{O}(1)$, then information is $\mathcal{O}(N)$. For information to saturate with N, the sum has to be $\mathcal{O}(1/N)$. To see how that could happen, we need to know how big the eigenvalues are. For that we use

$$\sum_k \sigma_k^2 = \mathrm{tr}\{\mathbf{\Sigma}\} = \mathcal{O}(N)\,. \tag{77}$$

Because the sum of the eigenvalues is $\mathcal{O}(N)$, some of them can be $\mathcal{O}(N)$. If information is to saturate, it is these $\mathcal{O}(N)$ eigenvalues that must dominate the sum in Eq. (74): at least one of them must have $\cos^2\theta_k \sim \mathcal{O}(1)$, and for the small (i.e., $\mathcal{O}(1)$) eigenvalues, $\cos^2\theta_k$ can be at most $\mathcal{O}(1/N)$. (In fact, because there are $\mathcal{O}(N)$ small eigenvalues, $\cos^2\theta_k$ must be $\mathcal{O}(1/N^2)$ for most of them.)

From a geometrical point of view, this means correlations are information limiting if $\mathbf{f}'$ points predominantly in the high-variance directions of the covariance ellipse; that is, it points in directions in which $\sigma_k^2 \sim \mathcal{O}(N)$. It can have small components in the low-variance directions, but they have to be very small: the eigenvectors must be almost perpendicular to $\mathbf{f}'$ – sufficiently perpendicular that $\cos^2\theta_k/\sigma_k^2$ is sometimes proportional to $1/N$ but usually proportional to $1/N^2$.

To visualize information limiting correlations, it is convenient to consider a case in which only one of the σ_k^2, say σ_1^2, is $\mathcal{O}(N)$ and the rest are $\mathcal{O}(1)$. Explicitly separating these two components, we have

$$I = \frac{\mathbf{f}'^T\mathbf{f}'}{\sigma_1^2}\cos^2\theta_1 + \mathbf{f}'^T\mathbf{f}'\sum_{k>1}\frac{\cos^2\theta_k}{\sigma_k^2}\,. \tag{78}$$

For information to be $\mathcal{O}(1)$, the sum on the right hand side must be $\mathcal{O}(1/N)$; this in turn implies that $\sum_{k>1} \cos^2 \theta_k \sim \mathcal{O}(1/N)$, which, via Eq. (76), implies that $\cos^2 \theta_1 = 1 - \mathcal{O}(1/N)$. Thus, for differential correlations to exist, θ_1 must be $\mathcal{O}(1/N^{1/2})$. In plain language: for correlations to be information limiting, the angle between $\mathbf{f}'$ and the $\mathcal{O}(N)$ direction (or, in the more general case, directions) of the covariance ellipse must be very small – it must scale as $1/N^{1/2}$. This is illustrated in Supplementary Fig. 2a. (We also show, mainly for completeness, the structure of the noise when we take into account the fact that $\mathbf{f}(s)$ is actually a curve, not a straight line; see Supplementary Fig. 2b.)

Of course, the eigenvalues don't typically neatly divide into $\mathcal{O}(1)$ and $\mathcal{O}(N)$ groups (although, because of Eq. (77), there are always a large number of $\mathcal{O}(1)$ eigenvalues). However, this doesn't change much the basic picture: the smaller σ_k^2 is, the closer $\mathbf{f}'$ must be to a direction perpendicular to $\mathbf{v}_k$, the corresponding eigenvector. However, it does introduce one new ingredient: the covariance matrix can have a large direction not parallel to $\mathbf{f}'$. For instance, if there are two large directions, say $k = 1$ and 2, then we would break the information up as

$$I = \frac{\mathbf{f}'^T \mathbf{f}'}{\sigma_1^2} \cos^2 \theta_1 + \frac{\mathbf{f}'^T \mathbf{f}'}{\sigma_2^2} \cos^2 \theta_2 + \mathbf{f}'^T \mathbf{f}' \sum_{k>2} \frac{\cos^2 \theta_k}{\sigma_k^2} \,. \tag{79}$$

Again we must have $\sum_{k>2} \cos^2 \sim \mathcal{O}(1/N)$, but now this means $\cos^2 \theta_1 + \cos^2 \theta_2 = 1 - \mathcal{O}(1/N)$. Thus, both θ_1 and θ_2 could be far from 0.

4.3 Implications for estimating Fisher information from data

This analysis tells us why computing information is so tricky in the presence of differential correlations: the error in the estimates of the direction of $\mathbf{f}'$ and the $\mathcal{O}(N)$ directions of the covariance ellipse must scale as $1/N^{1/2}$. Thus, if one estimated the covariance matrix and tuning curves form data (which is equivalent to computing the angles, θ_k, and eigenvalues, σ_k^2), and then used that to directly estimate information via Eq. (20), large errors would be likely. And, indeed, that is exactly what we find (see Sec 6.3 of this document, and Fig. 7b of the main text).

Note, though, that if differential correlations are present (an important "if," as we discuss shortly), estimating a lower bound on information using a linear decoder is relatively easy.

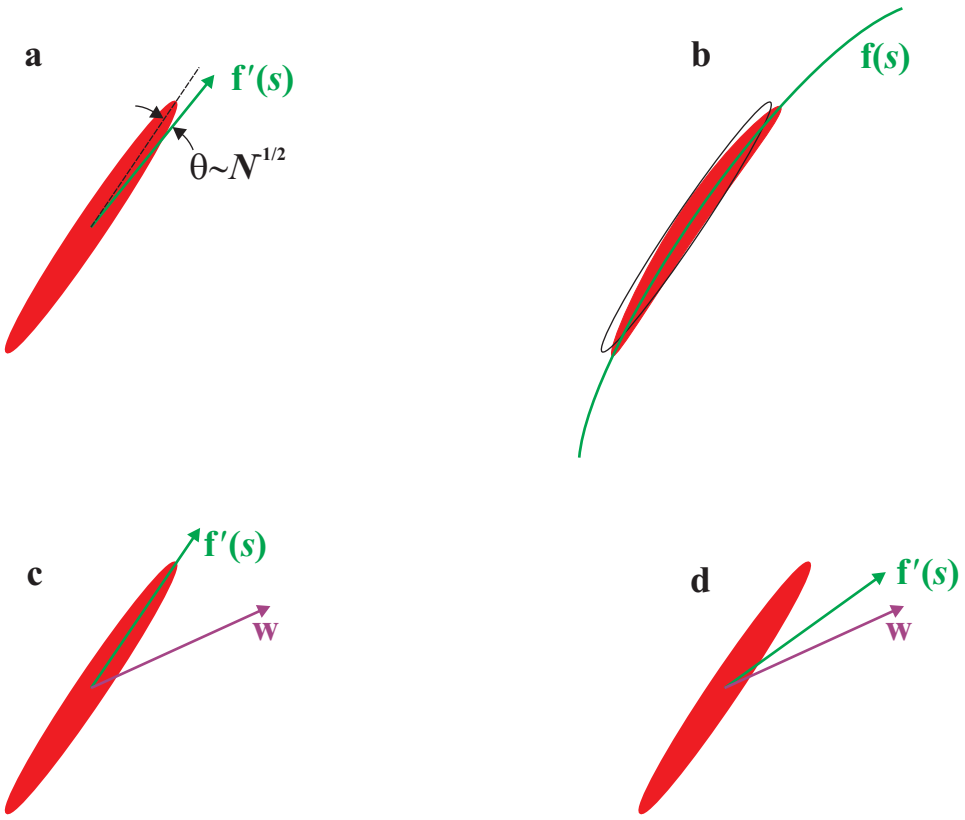

Figure 2: Visualizing information limiting correlations. All figures lie in firing rate space, here shown as two (out of N) dimensional. Red indicates the noise; in panels a, c and d, it denotes the noise covariance ellipse. We are assuming that one of the eigenvalues of the noise covariance ellipse is $\mathcal{O}(N)$ and the rest are $\mathcal{O}(1)$. **a**. When the angle between $\mathbf{f}'(s)$ and the long (i.e, $\mathcal{O}(N)$) direction of the noise covariance ellipse is $\mathcal{O}(1/N^{1/2})$, the correlations are information limiting. **b**. Realistic information limiting noise tracks $\mathbf{f}(s)$. The thin black curve is the covariance ellipse from panel a. **c**. When there are information limiting correlations, a suboptimal linear decoder, $\mathbf{w}$, predicts that information saturates with N even if the angle between $\mathbf{w}$ and the long direction of the noise covariance ellipse is $\mathcal{O}(1)$. **d**. When there are no information limiting correlations, so that information is $\mathcal{O}(N)$, the same linear decoder still predicts that information saturates with N. To observe the $\mathcal{O}(N)$ scaling, $\mathbf{w}$ would have to be nearly perpendicular to the long direction of the covariance ellipse.

To see why, consider a linear decoder, $\mathbf{w}$, chosen to be unbiased when s is near s_0,

$$\hat{s} = s_0 + \frac{\mathbf{w}^T\big(\mathbf{r} - \mathbf{f}(s_0)\big)}{\mathbf{w}^T\mathbf{f}'(s_0)}\,. \tag{80}$$

When $s = s_0$, the variance of $\hat{s}$ is given by

$$\mathrm{Var}[\hat{s}] = \frac{\mathbf{w}^T\mathbf{\Sigma}\mathbf{w}}{\big(\mathbf{w}^T\mathbf{f}'\big)^2} \tag{81}$$

where both $\mathbf{\Sigma}$ and $\mathbf{f}'$ are evaluated at $s = s_0$. Letting ψ be the angle between $\mathbf{w}$ and $\mathbf{f}'$, and using Eq. (72) for $\mathbf{\Sigma}$, this becomes

$$\mathrm{Var}[\hat{s}] = \frac{\sum_k \sigma_k^2 \cos^2\phi_k}{\mathbf{f}'^T\mathbf{f}'\cos^2\psi} \tag{82}$$

where ϕ_k is the angle between $\mathbf{w}$ and $\mathbf{v}_k$: $\mathbf{w}^T\mathbf{v}_k = |\mathbf{w}|\cos\phi_k$. Again, for the purpose of visualization, consider the case in which σ_1^2 is $\mathcal{O}(N)$ and the rest of the eigenvalues are $\mathcal{O}(1)$. Separating the two components gives us

$$\mathrm{Var}[\hat{s}] = \frac{\sigma_1^2\cos^2\phi_1}{\mathbf{f}'^T\mathbf{f}'\cos^2\psi} + \frac{\sum_{k>1}\sigma_k^2\cos^2\phi_k}{\mathbf{f}'^T\mathbf{f}'\cos^2\psi}\,. \tag{83}$$

Assuming that $\cos\psi$ is $\mathcal{O}(1)$, we see that the second term is $\mathcal{O}(1/N)$ (because the numerator is essentially an average of $\mathcal{O}(1)$ quantities and the denominator is $\mathcal{O}(N)$). The first term, on the other hand, is $\mathcal{O}(1)$ if $\cos^2\phi_1$ is $\mathcal{O}(1)$. Thus, so long as $\mathbf{w}$ is not almost perpendicular to either $\mathbf{f}'$ or the long direction (or, more generally directions) of the covariance ellipse, the variance of the optimal estimator is $\mathcal{O}(1)$ (see Supplementary Fig. 2c). If we use the inverse of that variance as an estimate of the Fisher information, then the Fisher information would saturate as N becomes large.

If we estimate $\mathbf{w}$ – by, say, minimizing decoding error, as we do in Sec. 6.4 below – we can get a relatively good estimate of the Fisher information (see also Fig. 7c of the main text). Importantly, $\mathbf{w}$ no longer needs to be estimated all that accurately. This is in sharp contrast to the direct approach, in which both $\mathbf{f}'$ and $\mathbf{\Sigma}$ have to be estimated very accurately.

This analysis applies, of course, only if differential correlations exist. If they don't, then we know $\mathrm{Var}[\hat{s}]$ is $\mathcal{O}(1/N)$. Consequently (assuming we're in the interesting case in which at least one eigenvalue of $\mathbf{\Sigma}$ is $\mathcal{O}(N)$), the first term in Eq. (83) must be $\mathcal{O}(1/N)$. For this to happen, the suboptimal estimator must be almost perfectly perpendicular to the long

direction (or, in the general case, the long directions) of the covariance ellipse. Again, then, we have a hard estimation problem, as illustrated in Supplementary Fig. 2d. So if we were to estimate a linear decoder, $\mathbf{w}$ from data, but didn't estimate it perfectly, we would find that information saturated when in fact it would not with a perfect linear decoder. This is exactly what happened when we used ridge-regularized early stopping to compute a decoder (third method Sec. 6.5): information associated with the decoder saturated even though the true information did not, as can be seen in the bottom left panel of Fig. 3. Thus, although one can use a suboptimal estimator if differential correlations exist, one cannot use a suboptimal estimator to determine *whether* they exist. Fortunately, this doesn't really matter, since differential correlations always exist in the large N limit.

5 Detecting differential correlations

As discussed in the main text, differential correlations may be hard to detect directly. This is because they may be masked by other correlations, as we show in two examples.

5.1 Differential correlations hidden by structured correlations

We first look at differential correlations that are large but hidden by structured correlations. As in the main text, we consider tuning curves and correlations of the form

$$\Sigma_{ij} = (1-c)\delta_{ij} + c\cos(s_i - s_j) \tag{84a}$$

$$f_i(s) = a + b\cos(s - s_i) \tag{84b}$$

where $a > b$, the preferred stimuli, s_i, are equally spaced, and there are N neurons. Although differential correlations are not immediately obvious, they do exist, as can be seen by expanding the cosine,

$$\Sigma_{ij} = (1-c)\delta_{ij} + c\big[\cos(s - s_i)\cos(s - s_j) + \sin(s - s_i)\sin(s - s_j)\big]\,. \tag{85}$$

The last term, $\sin(s - s_i)\sin(s - s_j)$, is proportional to the derivatives of the tuning curves.

To compute Fisher information, we need the inverse of the covariance matrix. Using the fact that

$$\sum_j \cos^2 s_j = \sum_j \sin^2 s_j = N/2\,, \tag{86}$$

it is straightforward to verify that

$$\Sigma_{ij}^{-1} = \frac{1}{1-c}\left[\delta_{ij} - \frac{c\cos(s_i - s_j)}{1-c+Nc/2}\right]\,. \tag{87}$$

Consequently, the Fisher information is given by

$$I = \frac{b^2}{1-c}\sum_{ij}\sin(s-s_i)\left[\delta_{ij} - \frac{c\cos(s_i-s_j)}{1-c+Nc/2}\right]\sin(s-s_j)\,. \tag{88}$$

Again using Eq. (86) to perform the sums over i and j, this reduces to

$$I = \frac{b^2}{1-c}\left[\frac{N}{2} - \frac{cN^2/4}{1-c+Nc/2}\right] = \frac{b^2}{c+2(1-c)/N}\,. \tag{89}$$

Thus, Fisher information saturates, as we would expect for differential correlations.

5.2 Differential correlations hidden by non-differential correlations

Differential correlations can also be hidden by non-differential correlations. To illustrate that, we consider a covariance matrix and tuning curves for which the information scales as N, add differential correlations as we did in Eq. (61), and show that the correlation coefficients with and without differential correlations are virtually indistinguishable – certainly indistinguishable with the amount of data that can be collected in realistic experiments.

We consider a model in which a population of N neurons has circular Gaussian tuning curves and, following Ecker and colleagues [4], random peak firing rates. Specifically, the mean firing rate of neuron i is given by

$$f_i(s) = A_i \exp\big((\cos(s-s_i)-1)\big) \tag{90}$$

where the preferred stimuli, s_i, are drawn from a uniform distribution on the interval $(-\pi, \pi]$, and the peak firing rates, A_i, are sampled from a Gamma distribution with a mean of 40 and a standard deviation of 20,

$$p(A) = \frac{A^3 e^{-A/10}}{10^4\,\Gamma(4)}\,. \tag{91}$$

A representative sample of these tuning curves is shown in Fig. 6a of the main text. In the absence of differential correlations, the covariance matrix, denoted $\mathbf{\Sigma}_0(s)$, is given by

$$\Sigma_{0,ij}(s) = c_{ij}\big(f_i(s)f_j(s)\big)^{1/2} \tag{92}$$

with correlation coefficients, c_{ij}, chosen to be

$$c_{ij} = (1-\rho)\delta_{ij} + \rho \exp\left(\kappa\left(\cos(s_i - s_j) - 1\right)\right). \tag{93}$$

We used $\rho = 0.2$ and $\kappa = 2$. As above, δ_{ij} is the Kronecker delta.

When differential correlations are present, the covariance matrix acquires an additional term, $\epsilon \mathbf{f}'\mathbf{f}'^T$, as in Eq. (61). This gives us a new covariance matrix, denoted $\mathbf{\Sigma}_\epsilon(s)$,

$$\Sigma_{\epsilon,ij}(s) = c_{ij}\big(f_i(s)f_j(s)\big)^{1/2} + \epsilon f_i'(s)f_j'(s)\,. \tag{94}$$

In our simulations we used $\epsilon = 0.002742$, chosen so that, in the presence of differential correlations, information saturates at a value that corresponds to a discrimination threshold of approximately 3 degrees.

To show how information scales with N, we computed information both with and without differential correlations. We started without differential correlations. To do that for a particular N, we sampled N random preferred orientations, s_i, and for each we sampled a random amplitude, A_i. We used those to construct tuning curves $f_i(s)$, via Eq. (90), and the covariance matrix $\mathbf{\Sigma}_0$, via Eq. (92). Information was then computed directly from Eq. (59). Next we computed information with differential correlations. For that we used Eq. (64), which tells us that $I_\epsilon = I_0/(1+\epsilon I_0)$. The results are shown in Fig. 6b of the main text, with the blue and red curves corresponding to information with and without differential correlations, respectively.

To visualize the correlation coefficients with and without differential correlations, we plot them versus difference in preferred stimuli (Fig. 6c, main text). Although we have analytic expressions for the correlation coefficients, we computed them by simulating data, as that gives a more realistic picture of what one might see in experiments. Specifically, we did the following: Two preferred orientations, s_i, and their associated amplitudes, A_i, were randomly sampled, as describe above. Using Eqs. (90-94), this fully specifies the mean, variance,

and covariance of the two neurons. We then generated 1000 samples from the associated two dimensional Gaussian random variable, both with ($\epsilon = 0.002742$) and without ($\epsilon = 0$) differential correlations. For all samples we set the stimulus, s, to 0. These 1000 samples were used to compute the correlation coefficient between the two randomly generated neurons. We repeated this for 400 pairs of neurons; the resulting 400 blue and 400 red dots are shown in Fig. 6c of the main text. Blue dots come from a population with differential correlations present; red dots from a population without differential correlations.

6 Empirically estimating information

Estimating Fisher information from data in the presence of differential correlations is non-trivial, mainly because of limited data and access to only a subset of of the neurons. (Note that neither of these are issues for simulated data.) Here we demonstrate that numerically. We estimate Fisher information four ways: 1) computing a subset of the elements of the covariance matrix and filling in the rest (Fig. 7a, main text), 2) computing all the elements of the covariance matrix and then directly using the expression for the Fisher information, Eq. (20) (Fig. 7b, main text), 3) using a locally near-optimal linear decoder, regularized by early stopping (Fig. 7c, main text), and 4) using ridge regularization (Supplementary Fig. 3). Only the third method, early stopping, worked well. This is not to say that this is the only decoder that can be used to identify the presence of differential correlations. Other methods, perhaps even some based on variations of ridge regression, may work. Our goal here was simply to point out the limitations of naive approaches while providing at least one method that works.

6.1 Simulation details

For all figures, we used data simulated from the model described in the previous section, Sec. 5.2; see in particular Eqs. (90-94). Although there were differences in how we computed information, simulations always proceeded in two steps: first we sampled a set of N neurons, then we simulated data from those neurons.

Step 1: Sampling neurons. Each neuron is fully described by two parameters: preferred stimulus, s_i, and amplitude, A_i, where i labels neuron. Thus "sampling neurons" corresponds to choosing a set of s_i and A_i, $i = 1, ..., N$. As discussed in the previous section, the former, s_i, was chosen from a uniform distribution between $-\pi$ and π, and the latter, A_i, from a Gamma distribution with a mean of 40 and a standard deviation of 20 (see Eq. (91)). Given these parameters, the covariance matrix for a given stimulus, s, is specified by Eq. (94).

Step 2: Simulating data. To simulate data, we drew samples from a Gaussian distribution with mean and covariance matrix given by Eqs. (90) and (94), respectively. Because we had to estimate the derivative of the tuning curve, we used two values for the stimulus: $s_+ = +\epsilon^{1/2}$ and $s_- = -\epsilon^{1/2}$. For all simulations in which differential correlations were present, we set ϵ to 0.002742 (the same value we used in the previous section). We drew M samples for $s = s_+$ and another M for $s = s_-$, for a total of $2M$ samples. Estimates of the covariance matrix and the derivative of the tuning curves were, then, given by

$$\hat{\Sigma}_{ij} = \frac{1}{2M} \sum_k \left(r_i^k(s_+) - \hat{f}_i\right)\left(r_j^k(s_+) - \hat{f}_j\right) + \left(r_i^k(s_-) - \hat{f}_i\right)\left(r_j^k(s_-) - \hat{f}_j\right) \tag{95a}$$

$$\hat{f}_i' = \frac{1}{M} \sum_k \frac{r_i^k(s_+) - r_i^k(s_-)}{s_+ - s_-} \tag{95b}$$

where k labels trial and $\hat{f}_i$ is the empirically estimated mean of the population response averaged over both conditions,

$$\hat{f}_i = \frac{1}{2M} \sum_k \left(r_i(s_+) + r_i(s_-)\right) . \tag{96}$$

Each point in Figs. 7a-c of the main text represents an estimate of information averaged over 20 independent simulated data sets, each consisting of M independent samples. The colored lines indicate the number of samples, $M = 400, 800, 1600, 2400$ and 4000. Each of these 20 data sets was generated from a different population of N neurons sampled in the manner described above. Error bars represent standard error of the mean.

6.2 Filling in the covariance matrix (Fig. 7a, main text)

It is common in experiments to record different subsets of neurons from the same animal but at different times (often on different days), and from different animals. The question we

address here is: can these subsets be pooled to estimate Fisher information? For illustrative purposes, we consider the case in which we have only pairs of neurons, although the results generalize to larger subsets.

For this analysis, we generated simulated samples of neural activity from 4000 pairs of neurons, as described in Sec. 6.1, and estimated the derivatives of the resulting tuning curves and 2×2 covariance matrix via Eq. (95). We then filled in the missing entries in the covariance matrix using a slightly modified version of an approach introduced by Shadlen and colleagues [5]. The idea is to construct an approximation to the square root of the correlation matrix by resampling from the correlation coefficients derived from our 4000 pairs of neurons. The method, which is relatively complicated, proceeded as follows.

We first divided stimulus space into 20 uniformly spaced bins (the first bin ranged from $-\pi$ to $-\pi + 2\pi/20$, the second from $-\pi + 2\pi/20$ to $-\pi + 4\pi/20$, and so on). We then placed each of our 8000 neurons into one of the bins, based on its preferred stimulus. Since order doesn't matter, this gave us 210 ($= 20 \times 21/2$) different categories; each category had, on average, 4000/210 ($\approx$19) distinct pairs. Associated with each pair was a correlation coefficient, so we had about 19 correlation coefficients for each category.

Given this simulated data, we then constructed the square root of the correlation matrix, denoted $\mathbf{Q}$. For that, we randomly selected N neurons (out of 8000) from our data set. For each of the $N(N-1)/2$ pairs of neurons (labeled by indices i and j), we determined what category the pair was in (based on the preferred stimuli), and randomly, and uniformly, selected a correlation coefficient, denoted r_{ij}, from that category. The diagonal elements of square root of the correlation matrix were then set to

$$Q_{ii} = \frac{1}{\bar{r}\sqrt{N}} \sqrt{\bar{r} + \frac{2}{N}\left(1 - \bar{r} - \sqrt{(1-\bar{r})(1-\bar{r}+\bar{r}N)}\right)} \left(1 + \sqrt{(1-\bar{r})(1-\bar{r}+\bar{r}N)}\right) \quad (97)$$

and the off-diagonal elements to

$$Q_{i\neq j} = \frac{1}{\sqrt{N}} \sqrt{r_{ij} + \frac{2}{N}\left(1 - r_{ij} - \sqrt{(1-r_{ij})(1-r_{ij}+r_{ij}N))}\right)} \quad (98)$$

where $\bar{r}$ is the average of the randomly selected correlation coefficients, r_{ij},

$$\bar{r} = \frac{2}{N(N-1)} \sum_{i<j} r_{ij} \,. \quad (99)$$

The matrix of correlation coefficients was obtained by setting $\mathbf{C}^* = \mathbf{Q}\mathbf{Q}^T$, and then setting the diagonal entries of $\mathbf{C}^*$ to one.

While the method looks obscure, it simplifies considerably in the large N limit. In that limit, $Q_{ii} \to 1$ and $Q_{i\neq j} \to (r_{ij}/N)^{1/2}$ and, as is relatively easy to show,

$$C^*_{i\neq j} \to \frac{1}{N-2} \sum_{k\neq i,j} (r_{ik} r_{jk})^{1/2} \,. \tag{100}$$

This tells us something reasonable: the correlation coefficient between neurons i and j is the (nonlinearly) weighted sum of the correlation coefficients from those neurons to the neurons that they both have in common. The downside is that this works only if all the correlation coefficients are positive (which they were for us, but that's not the case in general).

The correlation matrix $\mathbf{C}^*$ was turned into a covariance matrix by multiplying it on each side by a diagonal matrix which contains the square root of the empirically observed average variances of each of the selected neurons. Fisher information was then computed from Eq. (20). The results are shown in Fig. 7a of the main text. As can be seen in that figure, this approach greatly overestimates information.

6.3 Direct estimate (Fig. 7b, main text)

Given that filling in the missing elements doesn't work, we computed Fisher information from neurons that were simulated simultaneously, as described in Sec. (6.1). Again we used a direct estimate, Eq. (20). The results are shown in Fig. 7b of the main text. This method also tended to over-estimate information, although not as badly.

6.4 Decoding with early stopping (Fig. 7c, main text)

Next we consider a decoding method – we use a locally optimal linear decoder to estimate the stimulus, and we take the inverse of the variance of the estimate of the stimulus (with a slight modification to account for bias) to be the Fisher information. As discussed in Sec. 4.2, this should provide reasonably good results.

The linear decoder is a set of weights, which we denote $\mathbf{w}$. To find $\mathbf{w}$, we combined linear regression with early stopping, the latter important because it avoids overfitting. Specifically,

we divided our $2M$ samples into three data sets of approximately equal size: the training set (size M_{TR}), test set (size M_{TE}), and validation set (size M_{VAL}), with $M_{TR}+M_{TE}+M_{VAL} = 2M$. The squared error for the training and test data sets are

$$E^2_{TR} = \sum_{m\in TR} \left(s_m - \bar{s}_{TR} - \mathbf{w}^T\left(\mathbf{r}_m - \bar{\mathbf{r}}_{TR}\right)\right)^2 \tag{101a}$$

$$E^2_{TE} = \sum_{m\in TE} \left(s_m - \bar{s}_{TR} - \mathbf{w}^T\left(\mathbf{r}_m - \bar{\mathbf{r}}_{TR}\right)\right)^2 . \tag{101b}$$

Here the notation $\in TR$ and $\in TM$ means include data in the training and test sets, respectively, s_m can be either s_+ or s_- (taken as usual to be $\pm\epsilon^{1/2}$), and a bar indicates an empirical average,

$$\bar{s}_{TR} = \frac{1}{M_{TR}} \sum_{m\in TR} s_m \tag{102a}$$

$$\bar{\mathbf{r}}_{TR} = \frac{1}{M_{TR}} \sum_{m\in TR} \mathbf{r}_m . \tag{102b}$$

Note that these empirical averages are taken over the training set, not the test set.

The weights of the linear estimator were updated by gradient descent applied to the training error,

$$\frac{d\mathbf{w}}{dt} \propto -\frac{\partial E^2_{TR}}{\partial \mathbf{w}} . \tag{103}$$

Weights were initialized to random values, and gradient decent updates were stopped when the error on the test data started to increase; that is, when dE^2_{TE}/dt changed sign from negative to positive, where

$$\frac{dE^2_{TE}}{dt} = \left(\frac{\partial E^2_{TE}}{\partial \mathbf{w}}\right)^T \frac{d\mathbf{w}}{dt} \propto -\left(\frac{\partial E^2_{TE}}{\partial \mathbf{w}}\right)^T \frac{\partial E^2_{TR}}{\partial \mathbf{w}} . \tag{104}$$

(Note that $\partial_{\mathbf{w}}$ is a vector whose i^{th} component is ∂_{w_i}.)

To estimate the linear Fisher information given the decoder, $\mathbf{w}$, we need to account for bias, as in Eq. (51); approximating $\mathbf{f}'$ using the two values of the stimulus, s_+ and s_-, we have

$$I_{\text{Early Stopping}} = \left[\frac{\mathbf{w}^T\left(\bar{\mathbf{r}}_{VAL}(s_+) - \bar{\mathbf{r}}_{VAL}(s_-)\right)}{s_+ - s_-}\right]^2 \frac{2}{\mathbf{w}^T\bar{\mathbf{\Sigma}}_{VAL}(s_+)\mathbf{w} + \mathbf{w}^T\bar{\mathbf{\Sigma}}_{VAL}(s_-)\mathbf{w}} . \tag{105}$$

Here $\bar{\mathbf{r}}(s_\pm)_{VAL}$ is the mean activity and $\bar{\mathbf{\Sigma}}_{VAL}(s_\pm)$ is the covariance of the activity, computed at either $s = s_+$ or $s = s_-$, using the validation data set. This quantity is plotted in Fig. 7c of the main text.

6.5 Ridge regularization

In Sec. 6.3 we saw that the direct approach for estimating Fisher information did not work well. One of the reasons is that it required the inversion of a covariance matrix. This can cause problems if some of the eigenvalues are near zero. We thus used exactly the same method as for the direct approach, but we regularized the covariance matrix by adding an additional diagonal component to the estimated covariance matrix,

$$\Sigma_{ij} \to \Sigma_{ij} + \lambda \delta_{ij} \,. \tag{106}$$

The parameter λ was chosen using three methods. The first two are relatively standard; the last involves decoding.

In the first method, we simply use $\lambda = 0.1$ – large enough to ensure that the covariance matrix is not ill-conditioned, but not so large that it adds a great deal of bias. In the second, we selected λ so that the effective number of degrees of freedom of the covariance matrix was $0.9 * N$, where N is the number of neurons [6]. Specifically, λ was chosen by solving the equation

$$0.9 * N = \sum_k \frac{\sigma_k^2}{\sigma_k^2 + \lambda} \tag{107}$$

where σ_k^2 is the k^{th} eigenvalue of the empirically estimated covariance matrix. This also led to relatively small values ($\lambda \approx 0.25$).

For both methods we estimated the tuning curves and covariance matrix as described in Sec. 6.1, and then estimated information using Eq. (20) but with the ridge-regularized covariance matrix (Eq. (106)). The results are shown in the first and second rows of Supplementary Fig. 3. In the presence of differential correlations, there is improvement compared to the direct approach (compare the right column of Supplementary Fig. 3 with Fig. 7b of the main text). However, the improvement is not all that large.

In the third method, we optimize the ridge parameter, λ, using held out data [7]. Note, though, that this method is somewhat different than the previous two: rather than directly estimating the Fisher information with a regularized covariance matrix, as in Eq. (106), we decode the neural responses with a ridge-regularized decoder. We do not discuss this analysis

in the main text, as it is a decoding-based means of estimating information, one that doesn't work as well as the early stopping algorithm used in Sec. 6.4 above (compare the bottom row of Supplementary Fig. 3 to Fig. 7c of the main text).

The method follows closely the early stopping method presented in the previous section. The only real difference is that we restrict the readout weights; rather than letting **w** be

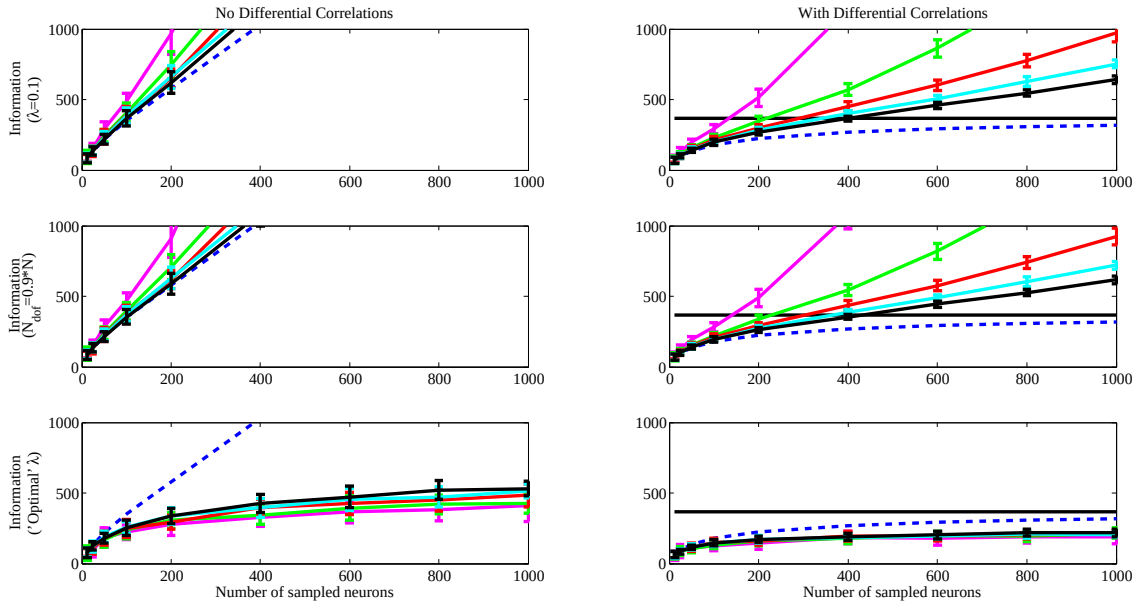

Figure 3: Empirically estimated information using ridge regularization. The black lines on the right show the upper bound on information given by $1/\epsilon$ (see Eq. (64)); the dashed blue lines are the true information. Different colors indicate different number of samples (M) used to estimate information: from top to bottom, M is 400, 800, 1600, 2400, 3200, and 4000. On the left, differential correlations are not present and information grows with the number of neurons. On the right, differential correlations are present and information saturates at the black line. **First row:** The ridge parameter, λ, is fixed at 0.1. **Second row:** The ridge parameter is chosen so that the effective number of degrees of freedom of the resulting covariance matrix is 90% of the maximum number of degrees of freedom possible (see Eq. (107)); typically $\lambda \approx 0.25$. **Third row:** The inverse of the variance of a decoder regularized using ridge regression and optimized by minimizing cross-validation error.

arbitrary, we use weights of the form

$$\mathbf{w}_{TR}(\lambda) = \frac{s_+ - s_-}{4} \left(\mathbf{\Sigma}_{TR} + \lambda \mathbf{I}\right)^{-1} \left(\mathbf{f}_{TR}(s_+) - \mathbf{f}_{TR}(s_-)\right). \tag{108}$$

(The reason for the prefactor, $(s_+ - s_-)/4$, will be discussed shortly.) As above, we randomly divided our $2M$ samples of population patterns of activity into three data sets of roughly equal size: a training set (of size M_{TR}), a test set (of size M_{TE}), and a validation set (of size M_{VAL}). The covariance matrix, $\mathbf{\Sigma}_{TR}$, was computed via

$$\mathbf{\Sigma}_{TR} = \frac{1}{M_{TR}} \sum_{m \in TR} (\mathbf{r}_m - \bar{\mathbf{r}}_{TR})(\mathbf{r}_m - \bar{\mathbf{r}}_{TR})^T \tag{109}$$

with $\bar{\mathbf{r}}_{TR}$ given in Eq. (102b), and the tuning curves, $\mathbf{f}_{TR}(s_\pm)$, were computed via

$$\mathbf{f}_{TR}(s_\pm) = \frac{1}{M_{TR}(s_\pm)} \sum_{m \in TR; s_m = s_\pm} \mathbf{r}_m \tag{110}$$

where $M_{TR}(s_\pm)$ is the number of training samples with $s_m = s_\pm$.

The ridge parameter, λ, was selected to minimize decoding error on the test set, E^2_{TE} in Eq. (101b) with $\mathbf{w}$ is replaced by $\mathbf{w}(\lambda)$. Information was then estimated using Eq. (105), again with $\mathbf{w}$ replaced by $\mathbf{w}(\lambda)$. Results are shown in the last row of Supplementary Fig. 3. The method worked reasonable well when information saturated (right plot). However, because the information is based on a suboptimal decoder, information computed this way saturated when the true information did not, as predicted in Sec. 4.3.

Properly normalizing the decoding weights is important. Here we show that the prefactor, $(s_+ - s_-)/4$, in Eq. (108) does provide the correct normalization, in the following sense: in the limit of infinite data, if s_+ is close to s_-, the correct weight vector is $\mathbf{w}_{TR}(0)$; i.e., in the limit of infinite date, the ridge parameter, λ, becomes zero. To do that we need to show that the correct weight vector in the infinite data limit is equal to $\mathbf{w}_{TR}(0)$.

We start with a generic regression problem, in which we want to minimize

$$\text{mean squared error} = \left\langle (\delta s - \mathbf{w}^T \delta \mathbf{r})^2 \right\rangle \tag{111}$$

with respect to $\mathbf{w}$, where

$$\delta s \equiv s - \overline{s} \tag{112a}$$

$$\delta \mathbf{r} \equiv \mathbf{f}(s) - \mathbf{f}(\overline{s}) + \boldsymbol{\xi}\,, \tag{112b}$$

$\boldsymbol{\xi}$ is zero mean noise with covariance $\boldsymbol{\Sigma}$, $\overline{s}$ is the mean value of s, and the angle brackets indicate an average over the true distributions of both s and $\boldsymbol{\xi}$. As usual, the weight vector that minimizes the mean squared error, denoted $\mathbf{w}_{opt}$, is

$$\mathbf{w}_{opt} = \langle \delta\mathbf{r}\delta\mathbf{r}^T \rangle^{-1} \langle \delta s \delta\mathbf{r} \rangle \,. \tag{113}$$

We'll consider a regime in which δs is small. In that case,

$$\delta\mathbf{r} = \delta s \mathbf{f}'(\overline{s}) + \boldsymbol{\xi} \,. \tag{114}$$

Consequently, assuming δs and $\boldsymbol{\xi}$ are independent,

$$\langle \delta\mathbf{r}\delta\mathbf{r}^T \rangle = \boldsymbol{\Sigma} + \sigma_s^2 \, \mathbf{f}'\mathbf{f}' \tag{115a}$$

$$\langle \delta s \delta\mathbf{r} \rangle = \sigma_s^2 \, \mathbf{f}' \tag{115b}$$

where $\mathbf{f}'$ is short for $\mathbf{f}'(\overline{s})$, and

$$\sigma_s^2 \equiv \langle \delta s^2 \rangle \tag{116}$$

(here the angle brackets indicate an average only over s). Inserting Eq. (115) into (113), we arrive at

$$\mathbf{w}_{opt} = \sigma_s^2 \left(\boldsymbol{\Sigma} + \sigma_s^2 \, \mathbf{f}'\mathbf{f}'^T \right)^{-1} \mathbf{f}' \,. \tag{117}$$

How does this compare to $\mathbf{w}_{TR}(0)$ when s_+ is close to s_-? To answer that, we first note that

$$\frac{s_+ - s_-}{4} \big(\mathbf{f}_{TR}(s_+) - \mathbf{f}_{TR}(s_-) \big) = \frac{(s_+ - s_-)^2}{4} \frac{\mathbf{f}_{TR}(s_+) - \mathbf{f}_{TR}(s_-)}{s_+ - s_-} \approx \frac{(s_+ - s_-)^2}{4} \mathbf{f}'(\overline{s}) \,. \tag{118}$$

Second, note that because we are using only two value of s, s_+ and s_-, it follows that

$$\frac{(s_+ - s_-)^2}{4} = \sigma_s^2 \,. \tag{119}$$

Finally, note, via Eqs. (109) and (114), that in the limit of infinite data (and, again, s_+ close to s_-),

$$\boldsymbol{\Sigma}_{TR} = \big\langle \big(\mathbf{r} - \mathbf{f}(\overline{s})\big) \big(\mathbf{r} - \mathbf{f}(\overline{s})\big)^T \big\rangle = \big\langle \big(\delta\mathbf{r} + \delta s \mathbf{f}'(\overline{s})\big) \big(\delta\mathbf{r} + \delta s \mathbf{f}'(\overline{s})\big)^T \big\rangle = \boldsymbol{\Sigma} + \sigma_s^2 \mathbf{f}'(\overline{s}) \mathbf{f}'(\overline{s})^T \,. \tag{120}$$

Combining this with Eqs. (118) and (119), we arrive at

$$\mathbf{w}_{TR}(\lambda) = \sigma_s^2 \big(\boldsymbol{\Sigma} + \sigma_s^2 \mathbf{f}'(\overline{s}) \mathbf{f}'(\overline{s})^T + \lambda \mathbf{I} \big)^{-1} \mathbf{f}'(\overline{s}) \,. \tag{121}$$

Comparing this to Eq. (117), we see that $\mathbf{w}_{TR}(0)$ is the correct weight vector.

6.6 Coarse discrimination (Fig. 6d, main text)

So far we have focused on Fisher information, which is reasonable for continuous-valued stimuli. However, our analysis applies also to coarse discrimination, in which the goal is to discriminate between two discrete stimuli. Here Fisher information cannot be used. We can, though, measure performance using percent correct.

To investigate how differential correlations affect decoding in a coarse discrimination task, we used the model described in Sec. 5.2, with some minor modifications: First, we allow the stimulus to take on only two values: s_+ and s_-, with $s_\pm = \pm 0.5236$ radians (corresponding to ± 30 degrees). Second, we chose $\epsilon = s_\pm^2 = 0.2742$ radians squared, so that performance is expected to saturate around 70%. And third, we reduced the average peak firing rate; we chose the amplitudes from a Gamma distribution with a mean and standard deviation of 4 and 2, respectively (rather than 40 and 20, as in our previous simulations), for which the distribution of amplitudes was

$$p(A) = \frac{A^3 e^{-A}}{\Gamma(4)} \,. \tag{122}$$

Covariance matrices were again computed using Eq. (94). Because of the Gaussian noise, the lower amplitude sometimes led to negative firing rates; those were set to zero. The number of samples was set to $M = 4000$.

When firing rates are this small, Fisher information no longer adequately captures the quality of the neural code. Therefore, we quantified performance using percent correct in a two alternative forced choice task. Weights of the separating hyperplane were computed by minimizing mean square error using the early stopping algorithm that is described in Sec. 6.4 above. Training, test, and validation sets were randomly selected and were of approximately equal size (about 1333 each, corresponding to a total of 4000 trials). Results are plotted in Fig. 6d of the main text. Once again, error bars represent standard error of the mean.

References

[1] J Beck, V R Bejjanki, and A Pouget. Insights from a simple expression for linear fisher information in a recurrently connected population of spiking neurons. *Neural Comput*, 23(6):1484–1502, 2011.

[2] P Seriès, P E Latham, and A Pouget. Tuning curve sharpening for orientation selectivity: coding efficiency and the impact of correlations. *Nat Neurosci*, 7(10):1129–1135, 2004.

[3] T.M. Cover and J.A. Thomas. *Elements of information theory*. John Wiley & Sons, New York, 1991.

[4] A S Ecker, P Berens, A S Tolias, and M Bethge. The effect of noise correlations in populations of diversely tuned neurons. *J. Neurosci.*, 31(40):14272–14283, 2011.

[5] Michael N. Shadlen, Kenneth H. Britten, William T. Newsome, and Anthony J. Movshon. A computational analysis of the relationship between neuronal and behavioral responses to visual motion. *Journal of Neuroscience*, 16(4):1486–1510, 1996.

[6] T Hastie, R Tibshirani, and J Friedman. *The Elements of Statistical Learning*. 2nd edition, 2009.

[7] GH Golub, M Heath, and G Wahba. Generalized cross-validation as a method for choosing a good ridge parameter. *Technometrics*, 21(2), 1979.