Nonsense correlations in neuroscience

Kenneth D. Harris

UCL Institute of Neurology, Queen Square, London WC1N 3BG, UK. Correspondence: kenneth.harris@ucl.ac.uk.

Most neurophysiological signals exhibit slow continuous trends over time. Because standard correlation analyses assume that all samples are independent, they can yield apparently significant "nonsense correlations" even for signals that are completely unrelated. Here we compare the performance of several methods for assessing correlations between timeseries, using simulated slowly drifting signals with and without genuine correlations. The best performance was obtained from a "pseudosession method", which relies on one of the signals being randomly generated by the experimenter, or a "session perturbation" method which requires multiple recordings under the same conditions. If neither of these is applicable, we find that a "linear shift method can work well, but only when one of the signals is stationary. Methods based on cross-validation, circular shifting, phase randomization, or detrending gave up to 100% false positive rates in our simulations. We conclude that analysis of neural timeseries is best performed when stationarity and randomization is built into the experimental design.

In neuroscience we often aim to correlate variables that depend on time. For example, we might correlate neuronal population activity on each trial of a task with behavioral variables such as choices. The statistical analysis of such data is difficult because the recorded variables often show slow changes in activity, which can lead to apparent correlations between them even if they are completely unrelated.

This phenomenon was given the memorable name of "nonsense correlation" by statistician G. Udny Yule (Yule, 1926). The problem of nonsense correlations has been discussed extensively in fields such as econometrics (Box, 2008; Granger and Newbold, 1974; Haugh, 1976; Phillips, 1986), but despite its importance to understanding neurophysiology data, has seen little discussion in this field (but see Elber-Dorozko and Loewenstein, 2018).

Here, we evaluate ten possible solutions to the problem, by applying them to simulated neural data. We find that two methods (the pseudosession and session permutation methods) do not produce nonsense correlations, they cannot be used in all situations. The remaining methods (linear shift, circular shift, phase and wavelet randomization, cross-validation) can all produce nonsense correlations, although linear shift appears to do so least often. We end with conclusions for how to design experiments that allow pseudosession and session permutation methods to be used.

What are nonsense correlations?

To illustrate the phenomenon of nonsense correlations, we consider a simulated experiment (**Figure 1**). Imagine we have recorded a population of $N = 10$ cells and computed their firing rate on $T = 200$ behavioral trials. To simulate the case that the neurons encode no information about behavior, we generate their rates randomly, independent of each other and of the simulated behavioral variables. We simulate slow rate drifts by summing logistic sigmoid functions centered on random times, together with pink noise (Methods; **Figure 1A1,1A2**).

We consider two types of simulated behavioral variable. First, we consider a binary "block" variable, which switches pseudo-randomly during the experiment; for example, this could indicate which of two stimuli or actions is most likely to reward (**Figure 1B1**). Even though this was generated independently of the neural

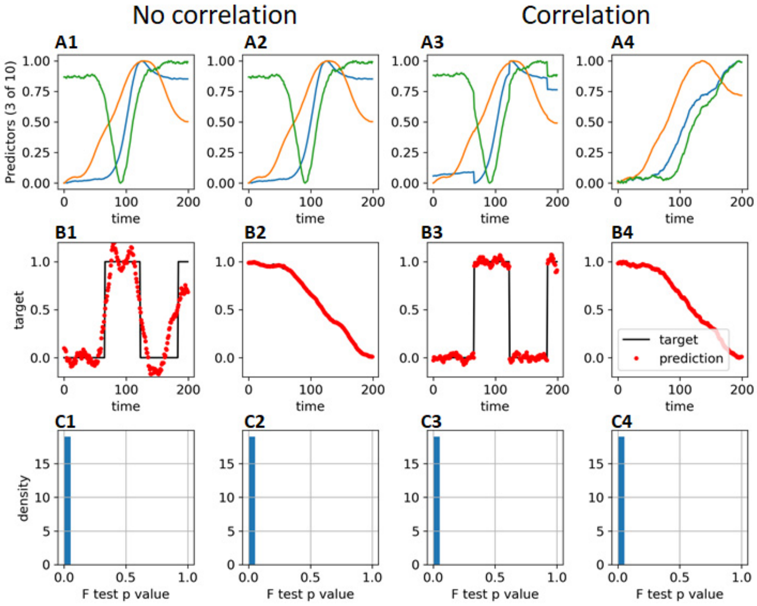

Figure 1 | Simulation of nonsense correlations. A1-A4: firing rates of 3 out of 10 simulated neurons as a function of trial number. In columns 1 and 2 the simulated firing rates are uncorrelated with the behavioral variable. In columns 3 and 4, a small randomly weighted copy of the behavioral variable has been added to the firing rates to produce a correlation. **B1-B4:** behavioral variable (black) and prediction of it from neural firing (red dots), using multiple linear regression with weights constant across the simulation. **C1-C4:** histogram of F-test p-values measuring significance of the linear regression, over 1000 simulations.

activity, it is possible to predict it accurately from neural activity, since by chance some of the neurons showed rate shifts at times close to the block switches (e.g. the green cell in **Figure 1A1**).

The second type of simulated behavioral variable was a continuous one, simulated the same way as the neural variables (**Figure 1B2)**; for example this could measure running speed on each trial. Again, this variable could be predicted almost perfectly from the simulated neural activity, even though it was generated independently.

To simulate a situation where neuronal firing rates do encode information about the behavioral variable, we added a small multiple of the behavioral variable to each neuron's firing rate, with a random weight (**Figure 1A3, 1A4**). Throughout the paper we therefore consider four scenarios: a binary or continuous behavioral variable, with or without genuine correlation to neural activity.

To measure how well the behavioral variable correlated with neural activity on each trial, we predicted it by multiple linear regression. (Other methods could be used but this would not change the basic results.) Applying the usual test of significance for multiple linear regression (the F-test), we find statistical significance in each of our four scenarios, for each of 1000 simulations (**Figure 1**).

The F-test thus always produced a false-positive error even when there was no genuine relationship between neural activity and the behavioral variable. This is because the F test assumes that the data on each timestep are statistically independent. However, both the predictor firing rates and the target behavioral variable are correlated across timesteps, and the test gives false significance.

Defining correlation between time series

Before considering potential solutions to the problem of nonsense correlations, we must first clearly define what we mean by a correlation between time series. To do so, we recall some basic concepts of probability theory, working here within the classical "frequentist" framework.

A fundamental concept in probability theory is the *sample space*. The sample space defines the set of all possible outcomes of an experiment, and a point in the sample space determines the value of everything measured in the experimental session. Throughout this paper we consider a simultaneous recording of N neurons and one behavioral variable, both measured on T trials. A point in the sample space is therefore defined by $(N+1)T$ numbers: the firing rate of each neuron and the behavioral variable on each trial. We will denote the firing rate of neuron n on trial t as x_{nt}, and gather them together into an NT-dimensional vector **x;** and we will denote the behavioral variable on trial t as y_t, gathered together into a T-dimensional vector **y**. Importantly, the sample space is defined by the entire history of these variables on all trials, not by their values on a single trial.

In the frequentist framework, we consider experiments to be repeatable, at least in principle: even if we only performed the experiment once, we consider it as part of an ensemble of repeats we could have performed. A probability distribution $\mathbb{P}(\mathbf{x},\mathbf{y})$ measures the frequency with which the experiment yields a particular outcome, over the infinite ensemble of possible repetitions of the experiment.

We say that neural activity is uncorrelated with behavior if the entire history of neural activity in an experiment (summarized by the vector **x**) is statistically independent of the history of behavioral variables summarized in **y,** i.e. if $\mathbb{P}(\mathbf{x},\mathbf{y}) = \mathbb{P}(\mathbf{x})\mathbb{P}(\mathbf{y})$.

Importantly, this definition allows neural activity to be *autocorrelated*: the firing rate of neuron n on trial t can be correlated with the firing rate of neuron m on trial u. Behavior can also be autocorrelated: the value of the behavioral variable at one time can be correlated with the value at another. Instead, independence requires that there be no *cross-correlation:* the activity of any neuron at any time is independent of behavior at any time. Thus, $\mathbb{P}(x_{nt}, y_u) = \mathbb{P}(x_{nt})\mathbb{P}(y_u)$, for any neuron n and any pair of times t and u.

A correlation between timeseries is therefore defined as a relationship that holds consistently across multiple repeats of the experiment, rather than across timepoints within a single experimental session. Predicting behavior from neural activity within a single session (Figure 1) does not show that neural activity is correlated with behavior. Instead, it must be possible to predict behavior from activity for all experimental sessions, using the same set of prediction weights for each session.

Does this mean that to show a correlation between neural activity and behavior one must record from the same neural population over multiple experimental sessions? Luckily, the answer is no, provided we make certain further assumptions. We next discuss how different assumptions allow different methods for detecting true correlations between time series. We focus on the simple case of testing whether neural and behavioral variables are correlated: more complex questions such as testing whether neural activity correlates with some behavioral variables after taking others into account, are discussed at the end of the manuscript.

Pseudosession method

The "pseudosession method" is simple, requires only a single experimental session, and is the only method we describe here that can show a causal relationship between two timeseries. However it has the strongest requirement: that one of the timeseries is randomly generated by the experimenter according to a known probability distribution. This method could be used for example to test whether neural activity differs between behavioral blocks, in an experiment where the block structure is generated randomly without dependence on the subject's choices.

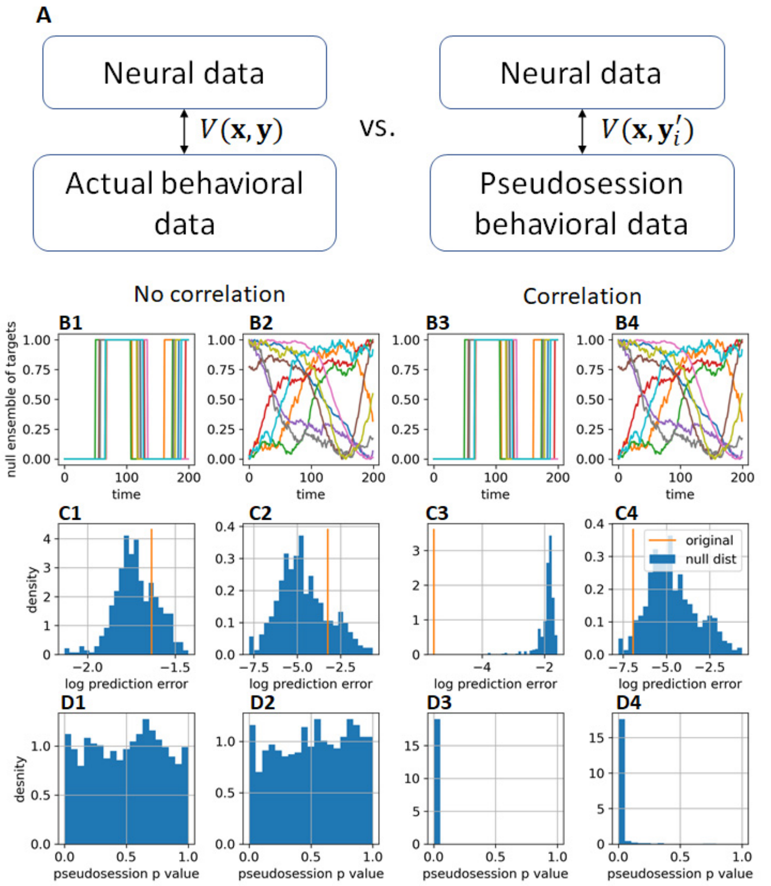

Figure 2 | Pseudosession method. A: a test statistic $V(\mathbf{x},\mathbf{y})$ is computed that measures the strength of relationship between the neural and behavioral data for an experiment. This is compared against a null distribution of $V(\mathbf{x},\mathbf{y}_i')$ obtained by repeatedly generating other behavior data $\mathbf{y}_i'$ drawn from the same distribution as $\mathbf{y}$. **B:** ensemble of behavioral variables for 10 pseudosessions, drawn from the same probability distribution as the original. **C**: histogram of log prediction error $\{V(\mathbf{x},\mathbf{y}_i')\}$ of null distribution (blue), with $V(\mathbf{x},\mathbf{y})$, the value for original data superimposed (orange). **D:** histogram of p-values obtained by comparing the predictability of the actual behavioral variable against a null ensemble of predictability of pseudosessions, from 1000 simulations.

Let $\mathbf{x}$ and $\mathbf{y}$ denote the history of neural activity and of the behavioral variable in a single session. The pseudosession method requires a "test statistic" $V(\mathbf{x},\mathbf{y})$, a single real number which quantifies the degree of association between $\mathbf{x}$ and $\mathbf{y}$ in that session. A good choice is the non-cross-validated error of a classifier trained to predict $\mathbf{y}$ from $\mathbf{x}$ or vice versa. Any choice of V gives a valid test; poor choices can only result in false-negative errors. In this paper we use the squared error of linear regression summed over time points. But any classifier can be used, including if it uses multiple timepoints of one series to predict individual timepoints of the other.

To apply the pseudosession method, we repeatedly generate random histories $\mathbf{y}'_i$ from the same probability distribution that generated $\mathbf{y}$, refit the prediction model to predict each $\mathbf{y}_i'$ from $\mathbf{x}$, and recompute the test statistic $V(\mathbf{x},\mathbf{y}'_i)$. We then define a p-value as the quantile of $V(\mathbf{x},\mathbf{y})$ relative to the null ensemble $\{V(\mathbf{x},\mathbf{y}'_i)\}$. This method therefore rejects the null hypothesis of no correlation if we can predict the actual behavioral history significantly better than we could predict a randomly generated one.

Applying the method to our four scenarios (**Figure 2**), we observe that p-values are evenly distributed when there is no true correlation but concentrated near zero when there is. We conclude that the pseudosession method works reliably when the behavioral variable is generated randomly from a known distribution.

Session permutation method

Because the pseudosession method requires the behavioral variable to have been randomly generated by the experimenter, it cannot be used to correlate neural activity with variables such as the subject's choices or running speed, which are not under the experimenter's control. The session permutation method can

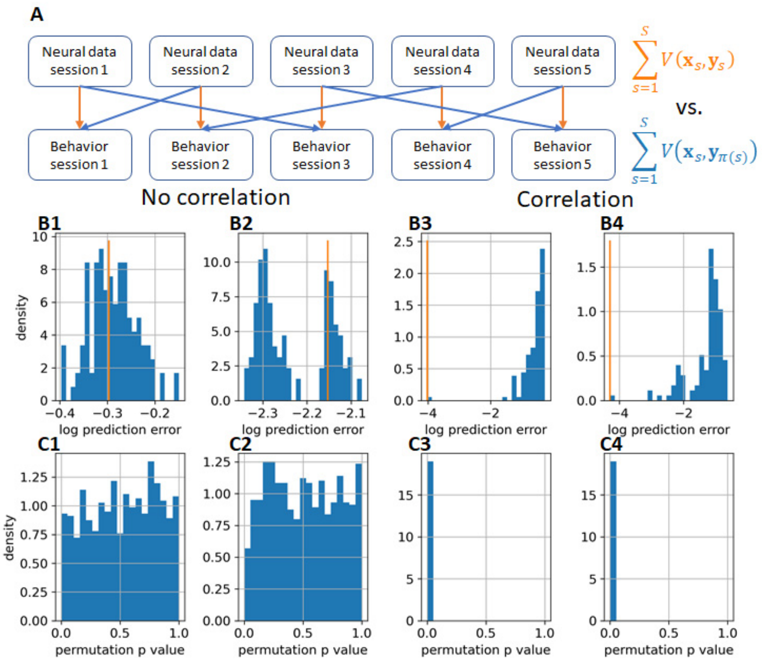

Figure 3 | Session permutation method. A: a test statistic is computed by summing the predictability of behavioral from neural data over each of S total sessions. This is compared to a null ensemble obtained from each of the $S!$ possible permutations of the sessions. **B:** histogram of log prediction error of permuted sessions (blue), with value for unpermuted data superimposed (orange). **C:** Histogram of p-values obtained with this method, over 1000 simulations. Columns 1-4 correspond to the same scenarios as in Figure 1.

analyze these cases but requires data from multiple sessions recorded under identical conditions.

The session permutation method asks whether neural activity predicts the behavioral variable on the same session more accurately than on other sessions. We denote the vectors containing the history neural activity and behavioral variables on the s^{th} session as $\mathbf{x}_s$ and $\mathbf{y}_s$. We sum the association measure over sessions to obtain a test statistic $V = \sum_{s=1}^{S} V(\mathbf{x}_s, \mathbf{y}_s)$. We compare this test statistic to a null ensemble in which the neural data of each session is compared to behavioral data from a randomly chosen session: $V'_\pi = \sum_{s=1}^{S} V\left(\mathbf{x}_s, \mathbf{y}_{\pi(s)}\right)$, where π runs over all of the $S!$ permutations of the S sessions. To obtain statistical significance needs at least 5 sessions (since $5! = 120$).

This method works for all our four scenarios, giving a flat distribution of p-values when the null hypothesis is true, and a sharp peak near 0 when the null is false (**Figure 3)**.

Session permutation does not require that the same neurons be recorded in each session, provided one can consider the neurons sampled on each session to be drawn randomly from the same population of potentially recordable neurons. The statistic $V(\mathbf{x}_s, \mathbf{y}_s)$ measures the degree to which the recorded population predicts behavior, and this can be computed using different neural populations on different sessions. If one cannot return to the same neurons on each session, however, it is not possible to say which neurons correlate with the behavioral variables; it is only possible to conclude that the population as a whole does.

Some caution is required in interpreting results of the session permutation method. Without a randomized experimental design, one cannot infer causality as there may be a third factor affecting the neural and behavioral recordings. For example, if the S sessions were recorded sequentially from the same subject, and consecutive experiments showed both a degradation in both the quality of neuronal recording and in behavioral performance, one might observe a correlation between neural activity and behavior simply for this reason.

Linear shift method

We have defined correlation between neural activity and behavior as a relationship holding

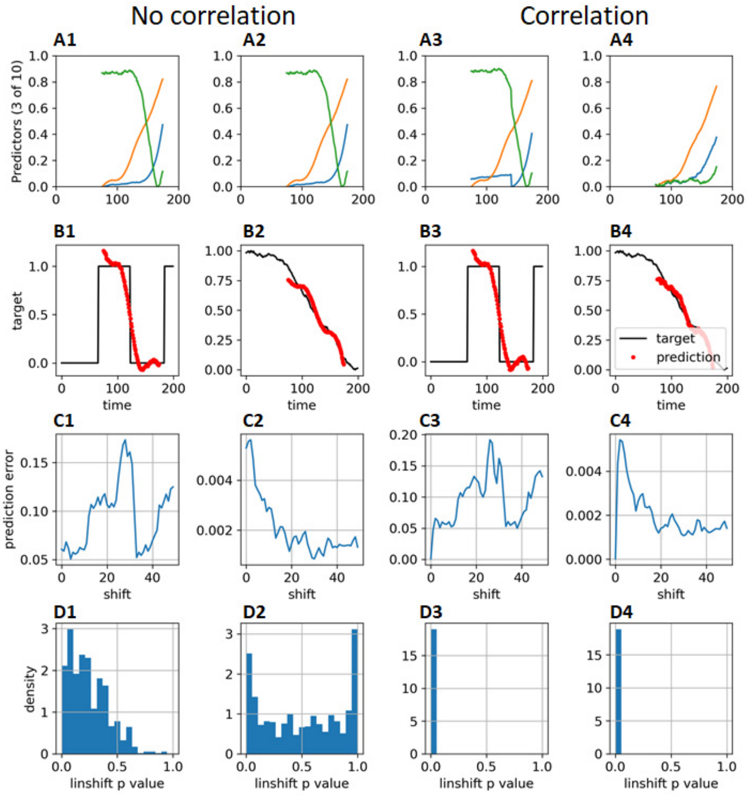

Figure 4 | Linear shift method. A,B: To create a null distribution, a segment of the neural data is time shifted and used to predict behavioral data starting at a different time. The example shows a segment of length 100 trials, linearly shifted by 75 trials. **C:** Prediction error as a function of time shift magnitude. **D:** histogram of p-values obtained for the method, over 1000 simulations. The four columns refer to the four scenarios of Figure 1.

consistently across sessions. Nevertheless, we can infer such a correlation from just one session if we make further assumptions. As we saw above, the pseudosession method makes an inference from one session, assuming a known probability distribution for the behavioral variable. If this distribution is unknown, we can still obtain an approximate measure of significance by making further assumptions of *stationarity* and *ergodicity*.

A probability distribution for time series is *stationary* if it is invariant to time shifting: for any τ, $t_1, t_2, \dots t_n$, $\mathbb{P}(x_{t_1+\tau}, x_{t_2+\tau}, \dots x_{t_n+\tau}) = \mathbb{P}(x_{t_1}, x_{t_2}, \dots x_{t_n})$. Stationarity is a property of the ensemble of all possible histories, rather than of any one session. Stationarity does not mean that the timeseries has a consistent character during a single experiment; rather, it means that absolute time irrelevant to the ensemble. For example, our ensemble of simulated block histories (**Figure 2B1**) is not stationary, since the first trial of any session is always in block 0. This distribution of block histories could be made stationary by first generating a long block history, then starting at a random point. Experimentally measured data such as behavioral and neural timeseries will be nonstationary if they show consistent trends between the beginning and end of each experiment. For example, if subjects typically respond faster at the beginning of a session experiment than at the end, then the timeseries of reaction times would be nonstationary.

A probability distribution over timeseries is *ergodic* if expectations over the ensemble of possible histories can be found by averaging over time: given any $t_1, t_2, \dots t_n$ and function ϕ, $\mathbb{E}[\phi(x_{t_1}, x_{t_2}, .. x_{t_n})] = \lim_{N\to\infty} \frac{1}{2N+1} \sum_{\tau=-N}^{N} \phi(x_{\tau+t_1}, x_{\tau+t_2}, .. x_{\tau+t_n})$. Ergodicity means that after enough time, anything that can happen, will happen. Stationarity does not imply ergodicity: an example of a time series that is stationary but not ergodic is one where every history is constant over time, but its constant value varies

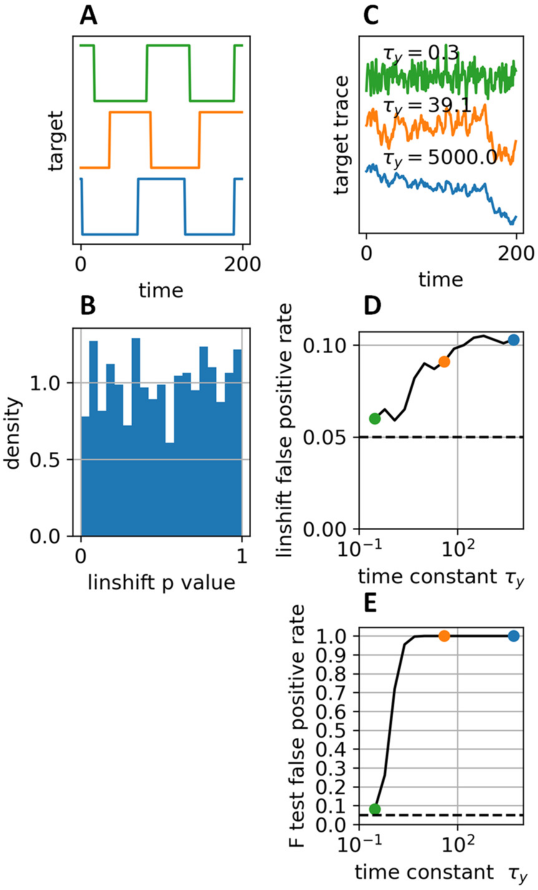

Figure 5 | Linear shift method: analysis of false positives. A: a stationary distribution for block structure was generated by starting at a random point of a long block sequence. Three examples are shown, vertically offset for clarity **B:** histogram of p-values for linear shift method, for neural series independent of these blocks. **C:** examples of pink noise target series with three different autocorrelation time constants. **D:** fraction of simulations for which the linear shift method gave p<.05 despite no genuine correlation, as a function of target autocorrelation time constant. Colored points reflect timescales shown in panel C. **E:** Similar plot for F-test of multiple linear regression. Note the different y-axis scale to panel D.

randomly between sessions. In practice, methods that rely on ergodicity will only work given enough data that the limit in the above equation has converged to good approximation.

The linear shift method tests the null hypothesis that time series are independent, and that one of them is stationary and ergodic (in this case $\mathbf{y}$). Denote by $\mathbf{x}[n{:}m]$ the $N(m-n)$-dimensional vector containing the firing rates of all N neurons between trials n and m. Given an integer parameter $s < T$, the linear shift method compares a test statistic $V(\mathbf{x}[0{:}s], \mathbf{y}[0{:}s])$ to a null distribution $\{V(\mathbf{x}[0{:}s], \mathbf{y}[\tau{:}\tau + s]) \mid \tau = 0 \dots T - s\}$. In other words, it asks whether neural activity predicts simultaneous behavior better than behavior taken from a different part of the experiment. If $\mathbf{x}$ and $\mathbf{y}$ are independent and $\mathbf{y}$ is stationary, the probability distribution of $V(\mathbf{x}[0{:}s], \mathbf{y}[\tau{:}\tau + s])$ does not depend on τ. Nevertheless, stationarity is not sufficient for the test to work: this empirical null distribution will only approximate the distribution over sessions if $\mathbf{y}$ is ergodic and $T - s$ is sufficiently large.

We evaluated the linear shift method on the same four scenarios as before (**Figure 4),** using $s = T/2$. The method gave few false negatives, but produced approximately double the rate of false positives than should occur. There are two possible explanations for this problem. First, although the scenario 2 target is stationary, the scenario 1 target is not. Second, while both types of behavioral data are ergodic, $T - s$ might not be large enough for accurate results.

To investigate which of these possibilities could be causing the false positives in scenario 1, we repeated the simulation but now using a block structure that was stationary by construction, by creating a long block sequence and starting at a random point within it (**Figure 5A**). After doing this, we found that the rate of false positives had returned to the correct value (**Figure 5B**).

The excess false positives in scenario 2 can only result from insufficient data, as the data are stationary and ergodic by construction. Indeed, examining the plot of prediction error vs. shift length (**Figure 4C2**) we see an approximately monotonic function, indicating that the session is not long enough for the ergodic limit to converge. If this curve were strictly monotonic, the fit quality for 0 shift would always be either the smallest or largest value possible, with 50% probability. Approximate monotonicity of this curve thus implies that the p-value histogram should show peaks at both 0 and 1, which is indeed seen (**Figure 4D2**), consistent with the hypothesis that the inflated false-positive rates in scenario 2 come from insufficiently long recording times.

The false positive error rate of the linear shift method depends on the temporal structure of the timeseries. To show this, we reran the simulations with the target variable given by pink noise of varying autocorrelation time constants (**Figure 5C**). As the time constant decreases, fast fluctuations come to dominate over slow trends (**Figure 5D**), and error rates decrease from ~10% to the correct level of 5%. In all cases, performance was still much better than the naïve F-test of multiple linear regression, which assumes independence, and produced a false positive in close to 100% if the autocorrelation time constant exceeded 1 trial.

Thus, the linear shift method will have inflated false positive errors on insufficiently long experiments containing slow drifts, but is still much better than traditional tests. In a case of insufficient data, the plot of fit quality vs. shift may be approximately monotonic (**Figure 4C2**).

Circular shift method

An alternative to the linear shift method is to generate a null ensemble by circularly shifting one of the timeseries: to replace $\mathbf{x}[0\colon T]$ with the concatenation of $\mathbf{x}[s\colon T]$ and $\mathbf{x}[0\colon s]$. This has the advantage of using all the data, unlike the linear shift method which discards some. However, in our simulations circular shifting showed much greater inflation of false-positive errors than linear shifting (**Figure 6**).

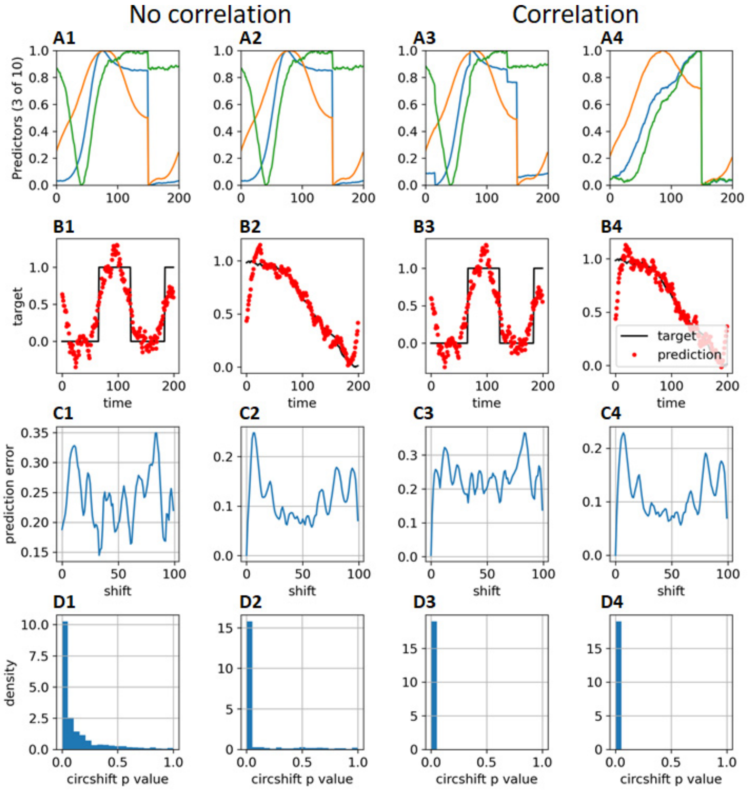

Figure 6 | Circular shift method. A,B: To create a null distribution, the neural data is circularly time shifted and used to predict behavioral data. The example shows a circular shift of 75 trials, where a discontinuity is visible. **C:** Prediction error as a function of circular shift magnitude. **D:** histogram of p-values obtained for the method, over 1000 simulations. Columns refer to the four scenarios of Figure 1.

The reason for this problem is that the circular shift method makes an assumption that is unlikely to hold. It tests the null hypothesis that not only are the series independent, but one of them is also cyclo-stationary: i.e. the probability of observing a particular history is the same as the probability of observing a cyclic shift of it. The reason this is unlikely to hold is that unless the start and end values of the timeseries are identical, cyclic shifting will introduce a

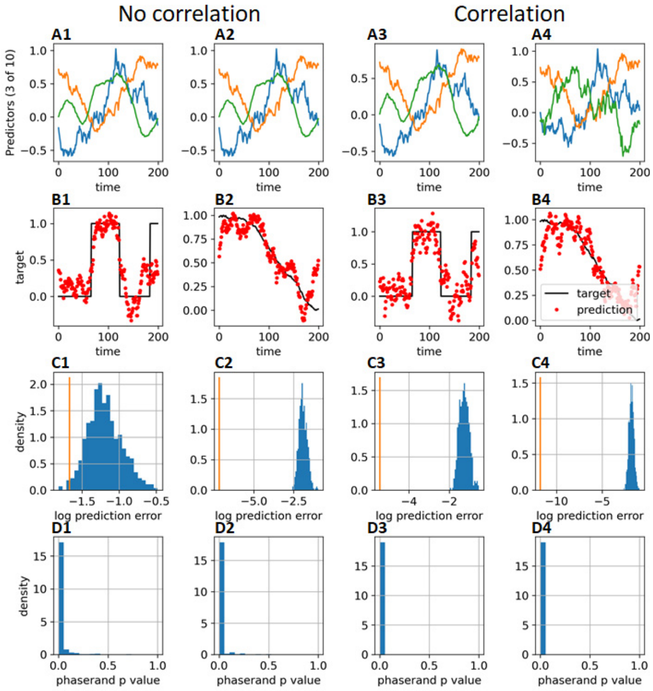

Figure 7 | Phase randomization method. A,B: To create a null distribution, the neural data is randomly phase-shifted to create a surrogate timeseries with the same frequency content, and used to predict behavioral data. **C:** histogram of p-values obtained for the method, over 1000 simulations. Columns refer to the four scenarios of Figure 1.

discontinuity (**Figure 6A**), which then renders the prediction of the behavioral series worse.

Phase/wavelet randomization

Another alternative to linear shifting, which has been suggested in the fMRI literature (Bullmore et al., 2001; Laird et al., 2004) is phase or wavelet randomization.

In the phase randomization method, a null distribution is obtained by applying a Fourier transform to one of the timeseries, multiplying each Fourier coefficient by a random phase and reverse transforming.

Our simulations suggested that this method inflates false-positive errors much more than the linear shift method (**Figure 7**). Whereas the circular shift method transforms continuous timeseries into discontinuous ones, phase randomization instead imposes cyclic continuity: after phase randomization the last sample is always close to the first. Furthermore, the phase-randomized signals have more high-frequency activity than the original, as the high-frequency energy resulting from the cyclic discontinuity in the original data has now been spread throughout time. As a result, the phase randomized data tends to predict the behavioral data worse, resulting in inflated false-positive errors.

An alternative method is wavelet randomization (Bullmore et al., 2001), which creates a null distribution by performing a wavelet transform on one of the timeseries, permuting the coefficients at each scale, and then inverse transforming. We found that

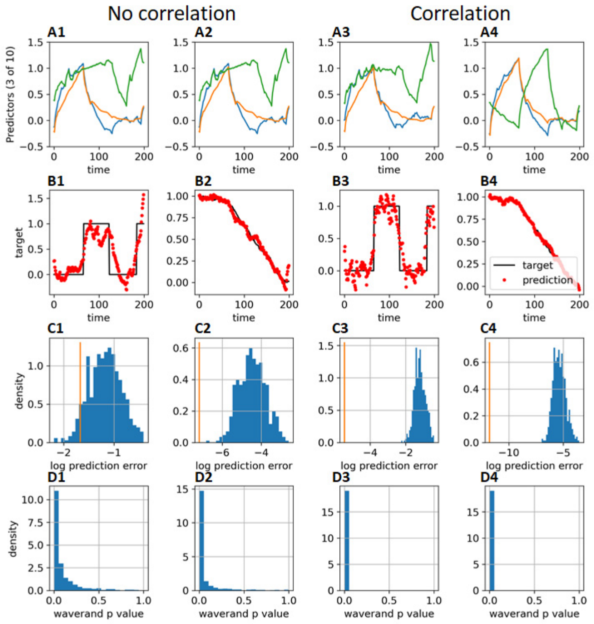

Figure 8 | Wavelet randomization method. A,B: To create a null distribution, a Daubechies(4) wavelet transform is applied to the neural data, the wavelet coefficients are permuted within each scale, and inverse transformed create a surrogate timeseries with the same frequency content, and used to predict behavioral data. **C:** histogram of p-values obtained for the method, over 1000 simulations. Columns refer to the four scenarios of Figure 1.

this method performed better than Fourier randomization, but still gave substantially more false positives than linear shift (**Figure 8).**

Cross validation

Cross validation does not solve nonsense correlations: slow autocorrelations mean that a predictor function learned on one part of the data will still be valid on another part of the data, even if these training and test sets are temporally segregated.

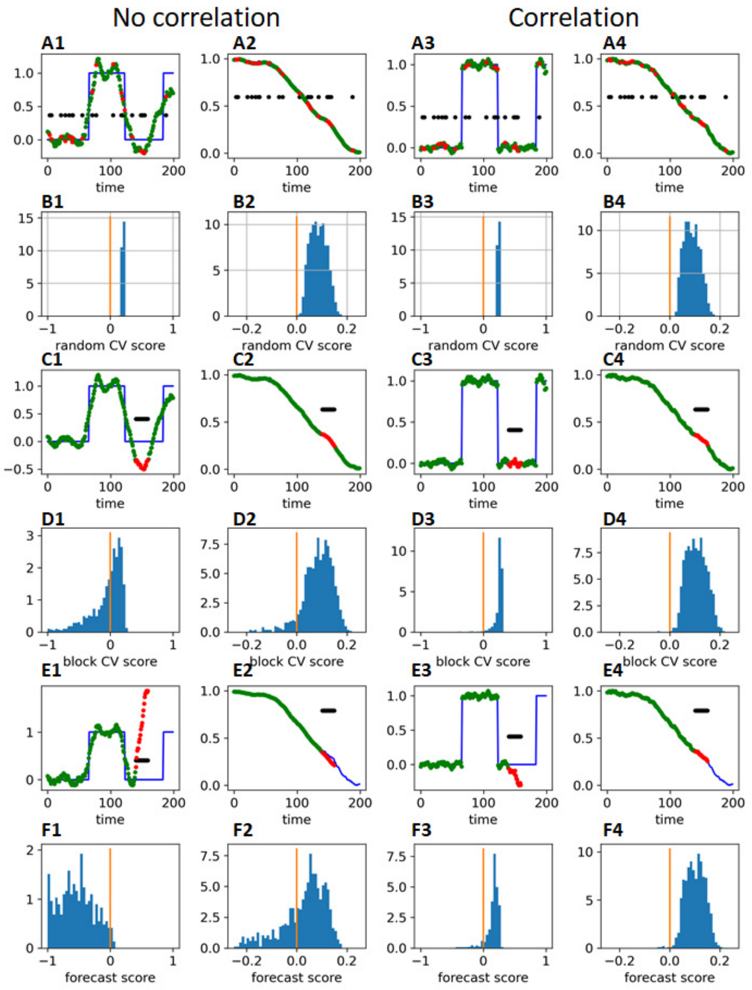

Figure 9 | Cross-validation method. A: Trials were divided into 10 random sets, with 9 used to train a classifier (green dots) and the last to test the prediction (red dots). Black dots indicate a null prediction made without access to the neural variables (the training set mean). **B**: histogram of predictability (mean error of prediction without neural variables minus mean error with them), over 1000 simulated sessions. **C,D**: same analysis, with blocks chosen to be temporally continuous. **E,F:** forecasting method, where the training set (green) is strictly before the test set (red). Columns correspond to the four scenarios in Figure 1.

To demonstrate this, we applied 10-fold cross-validation to our four scenarios (**Figure 9)**. When the training and test sets consisted of random time points, performance was abysmal: test-set predictions of the behavioral variable were more accurate than predictions made without access to the simulated neural variables in 100% of simulations, even when the neural and behavioral variables were unrelated (**Figure 9A,B**). When training and test sets consisted of blocks of sequential trials, false-positives errors were less common but still occurred in 54% of simulations of the block behavioral variable and 92% of simulations of the continuous variable (**Figure 9C,D**).

An alternative approach to time series cross-validation is forecasting (Tashman, 2000). In the approach, we predict the target timeseries in the n^{th} block using a predictor learned only from temporally prior blocks. As such predictions are extrapolation rather than an interpolation, one might expect false predictability to therefore be lower. This approach worked for the block variable, reducing the false positive rate to 1%; but it did not work for the continuous target, for which false-positives still occurred 61% of the time. (**Figure 9C,D**).

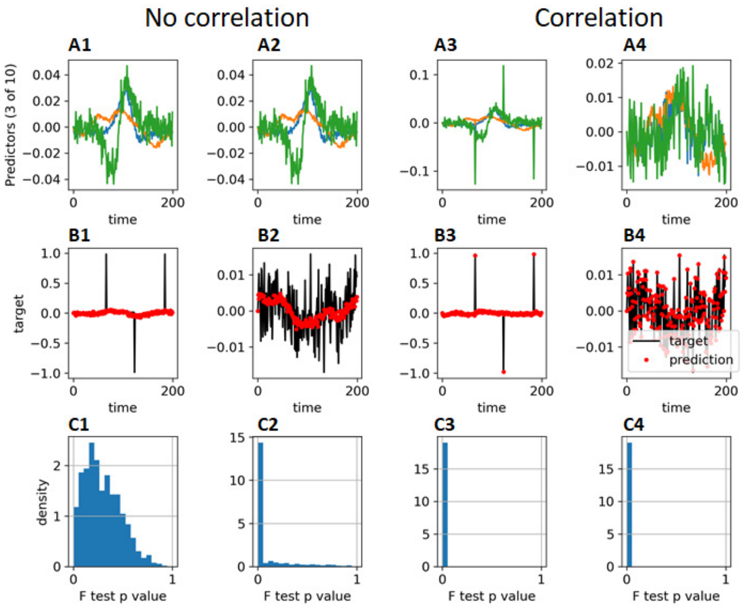

Figure 10 | Auto-decorrelation. A: the simulated neural data was preprocessed by fitting a first-order autoregressive model to each timeseries, and calculating a residual at each timestep. Due to the nonlinear nature of these timeseries, this does not result in white noise. **B:** the behavioral variables were preprocessed the same way (black) and predicted from the preprocessed neural data (red dots) using multiple linear regression. **C:** histogram of F-test p-values measuring significance of the linear regression, over 1000 simulations.

We conclude that cross-validation does not in general avoid nonsense correlations, although forecasting cross-validation can help in some circumstances. The use of cross-validation to avoid nonsense correlations must therefore be justified on a case-by-case basis.

Auto-decorrelation

A commonly-suggested approach to eliminate nonsense correlations is to preprocess the data to remove correlations within a single timeseries (Haugh, 1976). If we could remove these autocorrelations, then standard statistical tests that assume independent samples could safely be applied to the auto-decorrelated timeseries.

The usual way to perform auto-decorrelation is with an autoregressive model: one predicts the value of $\mathbf{x}_t$ by linear regression from previous values $\mathbf{x}_{t-n} \dots \mathbf{x}_{t-1}$, and performs all further analyses on this residual. Simpler approaches are to take the time derivative of each timeseries, or to detrend (for example by subtracting a best fit line).

While auto-decorrelation is in principle a solution to nonsense correlation, it comes with a major caveat: the auto-decorrelation algorithm must be extremely accurate. This cannot be guaranteed. For example, autoregressive models only exactly decorrelate linear timeseries (filtered white noise).

To evaluate auto-decorrelation, we fit a first-order autoregressive model to our simulated neural and behavioral variables, then applied a standard F-test to measure significance (**Figure 10**). Because these timeseries are nonlinear, however, the autoregressive model did not fully decorrelate the data: slow trends were still observed in the neural data (**Figure 10A,B)** although smaller than prior to preprocessing. For the binary block variable, auto-decorrelation replaced the step functions with impulses at the start and end of each block.

Even after this preprocessing, the F-test produced inflated false positive rates, although this was less bad for the block variable (**Figure 10C**).

Thus, it is not safe to apply statistical tests that assume independence even after auto-decorrelation, unless one has strong evidence that the auto-decorrelation method really worked to high accuracy. Nevertheless, auto-decorrelation could still be a useful tool used in conjunction with other approaches such as the linear shift method. Even if it only works partially, decorrelating the data cannot increase false positives found by the linear shift method, and may decrease them (Figure 5).

More complex analyses: partial correlation

So far, we have discussed the simple case of detecting a correlation between two timeseries, such as neural activity and a behavioral variable. One often wants to ask more complex questions. For example, if a subject's choices differ between behavioral blocks, one might ask whether the correlation of neural activity and behavioral block is stronger than would be predicted from an encoding of choice alone: in other words, is there a partial correlation of neural activity and block, after accounting for the common correlate of choice.

Many such questions can be transformed into questions about independence of timeseries. In the above example, we could predict neural activity from choice, and then test a null hypothesis that the residual of this correlation is independent of the behavior block. The approaches described above, therefore, can also be used to test more complex hypotheses such as partial correlation.

Implications for experimental design

This survey of methods for establishing genuine correlations between neural and behavioral timeseries yields a familiar lesson for experimental design: whenever possible, use a randomized experiment.

The power of randomized experiments to enable statistical analysis has long been recognized (Fisher, 1935). Of the methods described above, the one that is most reliable, powerful, and accurate is the pseudosession method, which can only be applied when one of the timeseries to be correlated is randomly controlled by the experimenter. Thus, whenever possible, experiments should be designed with randomized covariates. To test if neural population activity differs between behavioral blocks, the timing of these blocks should be randomized between sessions. Furthermore, because statistical power increases the more variable the block structure is across sessions, it may be worth using stationary block distributions (Figure 5A). To test if neural activity correlates with running, the best experimental design would be one that requires the subject to run at random times controlled by the experimental apparatus.

Summary

We have reviewed and simulated methods for detecting correlations between neural timeseries. Statistical tests that assume independence between timepoints result in "nonsense correlations" of erroneous statistical significance, due to autocorrelations within timeseries. The most reliable method for detecting genuine correlations, the pseudosession method, requires that one of the timeseries be randomly generated by the experimenter. The session permutation method requires the experiment be replicated at least 5 times under identical conditions, and can provide reliable results although correlations could reflect a common effect of session-to-session variability. Other methods reduce but not eliminate the risk of false positive errors; of these, the linear shift method appears to perform best. If possible, experiments should be designed so that time series of interest are randomized.

Methods

To generate a firing rate sequence, we added together a random number of logistic sigmoid functions. The center times $t_1, \ldots t_n$ of these functions were drawn from a homogeneous Poisson process of rate 1/100, so the mean number of sigmoids in T=200 trials was 2; their widths were always 10, and directions σ_i were random signs ± 1 with equal probability. A pink noise sequence p_t was added, generated by passing white Gaussian noise through an IIR filter with parameters $a = [1, -\sqrt{\alpha}], b = [\sqrt{1-\alpha}]$, where $\alpha = e^{-2/\tau}$ and $\tau = 5000$. The final sequence was

$$x_t = 0.1 p_t + \sum_{i=1}^{n} \frac{1}{1 + e^{(t-t_i)/10\sigma_i}}$$

To simulate behavioral binary blocks (column 1 of the figures), we generated alternating blocks of 0s and 1s, of lengths independently uniformly distributed between 50 and 70; the first 0 block always began at the first sample. To make a stationary block sequence (Figure 5), we generated a longer sequence (1000 blocks) and started it at a random time. To simulate continuous behavioral signal (column 2 of the figures), we generated another sequence from from the same distribution as the neural activity.

To simulate the case where the neurons encoded information about the behavioral variables (columns 3 and 4), the behavioral signal was added to each neuron's activity, with weight drawn from a Gaussian distribution of mean 0, SD 0.1.

Finally, each neuron's activity timeseries scaled between 0 and 1, although this will not have affected the linear regressions.

Code to perform the simulations is available at https://github.com/kdharris101/nonsense-correlations/ and can be run online at https://colab.research.google.com/github/kdharris101/nonsense-correlations/blob/main/nonsense.ipynb

Acknowledgements

I thank Sylvia Schroeder, Kevin Miller, Anna Lebedeva, Peter Dayan, and Matteo Carandini for conversations and comments on the manuscript. This work was supported by the Wellcome Trust (205093) and European Research Council (694401).

REFERENCES

Box, G.E.P. (2008). Time series analysis forecasting and control / George E.P. Box, Gwilym M. Jenkins, Gregory C. Reinsel. (Hoboken, N.J.: John Wiley).

Bullmore, E., Long, C., Suckling, J., Fadili, J., Calvert, G., Zelaya, F., Carpenter, T.A., and Brammer, M. (2001). Colored noise and computational inference in neurophysiological (fMRI) time series analysis: resampling methods in time and wavelet domains. Hum. Brain Mapp. *12*, 61–78.

Elber-Dorozko, L., and Loewenstein, Y. (2018). Striatal action-value neurons reconsidered. ELife *7*.

Fisher, R.A. (1935). The design of experiments (Edinburgh ... [etc: Oliver and Boyd).

Granger, C.W.J., and Newbold, P. (1974). Spurious regressions in econometrics. J. Econom. *2*, 111–120.

Haugh, L.D. (1976). Checking the independence of two covariance-stationary time series: a univariate residual cross-correlation approach. J. Am. Stat. Assoc. *71*, 378–385.

Laird, A.R., Rogers, B.P., and Meyerand, M.E. (2004). Comparison of Fourier and wavelet resampling methods. Magn. Reson. Med. *51*, 418–422.

Phillips, P.C.B. (1986). Understanding spurious regressions in econometrics. J. Econom. *33*, 311–340.

Tashman, L.J. (2000). Out-of-sample tests of forecasting accuracy: an analysis and review. Int. J. Forecast. *16*, 437–450.

Yule, G.U. (1926). Why do we Sometimes get Nonsense-Correlations between Time-Series?--A Study in Sampling and the Nature of Time-Series. J. R. Stat. Soc. *89*, 1–63.