Common Issues

Controlled vs naturalistic data

Lack of trials

Nonstationarity and long timescales

Trials/Sessions/Animals/Conditions

Binning

Missing data

Too much data

Multicollinearity

Gotcha: Normalization and correlations

Gotcha: Correlated training/test data

Gotcha: What is N?

Interpreting analysis results

Common Issues - Controlled vs naturalistic data

Controlled

Many trials/repetitions

Simplified task/stimuli

Usually uncorrelated stimuli

Experiment designed as simplest way to address specific scientific question

Can use “traditional” statistics

Unclear whether/how results generalize to natural settings

Naturalistic

Closer to natural setting

Often no trials or exact repetitions

Complex stimuli

Stimuli often correlated (see “multicollinearity”)

Experiment designed to evoke naturalistic behavior with little interference

Traditional statistics often insufficient

Common Issues - Lack of Trials

Issue:

- Traditional experiments often have many repeated trials of the same type to gain statistical power
  - Trials assumed to be independent (*this could be a strong/unfair assumption)
- More naturalistic experiments or observations may not have obvious trial structure
- E.g. video + brain recording of animal engaged in free behavior
  - Unlikely to be organized into clean trials unless animal is doing very repetitive behavior
- Typical methods, e.g. trial-averaging don’t immediately apply

Solutions:

- Create artificial trials by segmenting data
  - If specific event (e.g. locomotion onset) frequently repeats, let signal in window around event time correspond to “trial”. Then perform trial-based analysis as usual. Crucial: Be clear about how independent you expect “trials” to be.
- Apply non-trial-based analysis methods
  - Model comparison is fairly universal and applies to many models
  - Perform statistical testing by comparing measured statistic to shuffled/null data

Common Issues - Trials/Sessions/Animals/Conditions

Issue:

- Lots of nested structure inside a dataset
- Unclear what constitutes “independent” sample: Trial? Session (trial block)? Animal?
- Many decisions to make about how to average, normalize etc.
    - E.g. if interested in change in spike rate in response to some variable, do you: Subtract baseline rate before each trial? Normalize firing rates across sessions? Normalize firing rates across animals? Etc.

Solutions:

- Think very carefully about what scientific question or hypothesis you are addressing.
    - Pre-processing methods can strongly affect downstream results
- Run all analyses on control/artificial datasets of exact same structure as real data
    - I.e. same number of artificial animals, sessions, trials
    - Positive control = artificial dataset where phenomenon of interest is built in.
    - Negative control = artificial dataset explicitly excluding phenomenon of interest
        - Try to retain as much structure from the real data as possible and remove only phenomenon of interest. Rule of thumb: perform simplest shuffling of data dimensions that removes hypothesized phenomenon

Common Issues - Nonstationarity and long timescales

Issue:

- Neural and behavioral data often very different from classical signal processing signals
- Classical signals, e.g. in telecom engineering, often assumed to be stationary w short, well-defined autocorrelation time
  - E.g. autocorr function well approximated by exponential w short timescale tau
  - Simple-ish to segment into individual samples separated by >> tau
- Neural/behavioral data, however, are often nonstationary, e.g. part of signal changes on same timescale as length of experiment.
  - Can't easily segment into indiv trials. Classical signal processing analyses will be confounded, since these often assume stationarity and short timescales

Solutions:

- If you can justify that slow component of data is not important to research question, remove/subtract from data before proceeding (e.g. detrend, high-pass filter, etc).
- If long timescales are important and cannot be removed, avoid metrics like test set accuracy, since test set will be correlated with training set. Compare analysis outputs on real data to same analyses applied to thoughtful control datasets instead.
  - E.g. generate control datasets with long timescales but with and without hypothesized phenomenon present.

Common Issues - Binning

Issue:

- Many analysis require binning data, but what bin size should you use?
- E.g. when making a PSTH (peristimulus time histogram) from spike trains
  - Extremely small bins will contain at most 1 spike each, making the PSTH not very helpful
  - One large bin will obscure all finer resolution structure, also not very helpful

Solutions:

- Choose bin size large enough so that there is reasonable variation in how much each bin is filled
- If possible, test analysis results using different bin sizes to see whether they hold up
- Use kernel density estimators (convolve “kernel” function with spike train)
  - Although have similar problem: need to specify a “width”, analogous to bin size
- As usual, test analyses on positive+negative control datasets using same binning procedure as your real data.

Common Issues - Missing data

Issue:

- Experiments are not perfect, sometimes data gets lost (or doesn't exist in the first place)
  - E.g. camera drops out, animal is occluded, animal state changes (e.g. if want to ignore neural activity when animal is sleeping)

Solutions:

1. Ignore missing data. A lot of analyses can simply ignore missing data, e.g. when estimating autocovariance, simply ignore timepoint pairs where either are both NaN. Make sure to adjust number of samples accordingly.
   a. Or e.g. exclude missing data points from your likelihood function if fitting a max-likelihood model.
2. Interpolate missing data. Some common analysis methods, e.g. Fourier transform*, don't play well with NaNs. To apply them will need to fill-in missing data.
   a. Need to be very careful about how to do this. Filling in missing data in the wrong way can introduce spurious signals that will turn up as positive analysis results.
   b. As usual, test interpolation method on control datasets first.

Common Issues - Too much data

Issue:

- Experiments are very high-power and produce many Gb, Tb, or Pb of data.
- Makes working with data feel like driving a semi-truck.
- Everything is more complicated, requires more compute, more money, more time.
- Despite common idea that collecting more data can't hurt, especially if the collection process is cheap (e.g. natural behavior movies), overload of data can very much slow down practical time required to transform data into scientific result.

Solutions:

- Before anything else, **make sure there is interesting signal in the data**
- Look at a lot of raw and lightly pre-processed data by eye
- Compute all the basic stats to make sure they pass sanity checks
- Then, put science first:
  - What do you really want to know about data?
  - Easy to spend way too much time on "engineering" around data munging/processing, only to realize much of that wasn't necessary for science Q
- Create the simplest processed version of your data possible that will let you address your scientific question. This makes working with it much more flexible and iterative.

Common Issues - Multicollinearity

Multicollinearity: highly correlated inputs

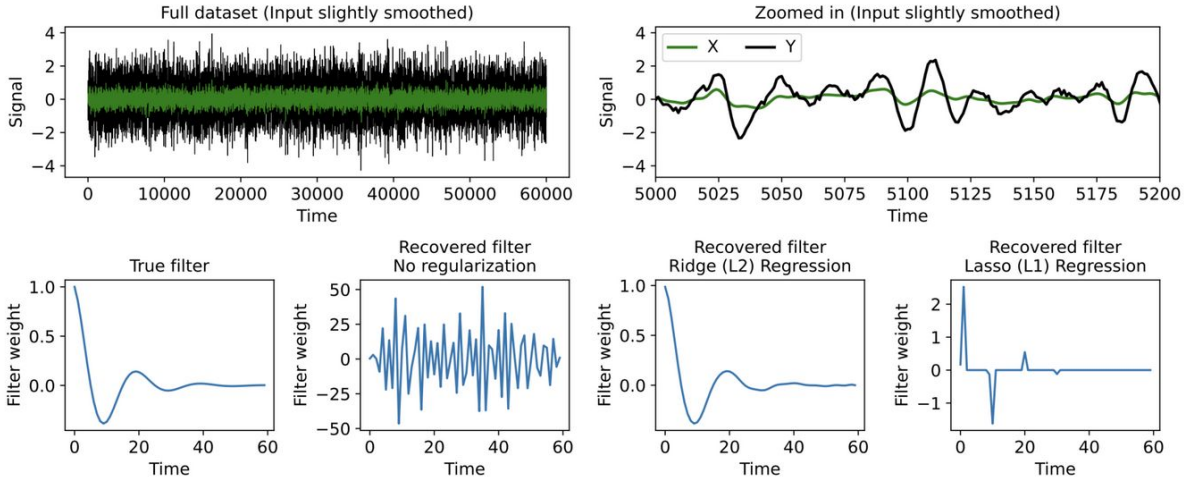
- Example: any time-series in which input timepoints are not i.i.d.
- *Extremely* when “input” is measured rather than controlled

Problem: ambiguity in which inputs to weight as predictors

- Example: x(t) and x(t+1) highly correlated → 2x(t) + 2x(t+1) ≈ x(t) + 3x(t+1) ≈ 4x(t) ≈ 4x(t+1)
- Many different solutions for how to weight inputs to predict output y(t)
- Unregularized regression techniques → weights highly sensitive to noise… often get e.g. 1000x(t) - 996x(t+1)

Solution: need “rule” for choosing which inputs to weight

- Ridge regression (L2 penalty): similar inputs → similar weights
- Lasso (L1 penalty): sparse set of nonzero weights on input
- Represent D-dim weight vector (filter) as combo of smaller number of weight vectors



Common Issues - Gotchas: normalization

Issue:

- Normalizing data, as innocent as it sounds, can introduce artificial correlations that get picked up by downstream analyses and can accidentally be interpreted as positive results.
- This is because normalization, i.e. dividing each data point by sum of all data points, creates interactions between resulting normalized data points.
    - Suppose that prior to normalization all N data points were independent.
    - After normalization, knowing N-1 data points gives you complete knowledge of the final data point.

Solutions:

- Run some simulations of normalization procedure to get an idea for how strong normalization-based interactions are. (Sometimes it’s no problem).
- If in doubt: run your whole analysis pipeline, including normalization step, on control datasets.

Common Issues - Gotchas: correlated training/test data

Issue:

- When using metrics like validation or test set prediction accuracy, important that test/validation data are approximately independent from training data
- In time-series data, timepoint-by-timepoint random splits of data will lead to data leakage ie correlated training/test data. This is because some training timepoints will be adjacent to test timepoints, and for many (most?) real-world neural/behavior time-series neighboring timepoints will be highly correlated.

Solutions:

- If data comes in trials, run training/test split across trials instead of timepoints, if you feel it is fair to treat trials as independent
- If data all in one or a few trials, check if data has reasonable autocorrelation time.
    - E.g. can autocorrelation function be stably estimated and fit with something like an exponential with a timescale tau much shorter than the time-series length?
    - If so, just make sure test and training data are separated by much more than tau
- If data is too nonstationary/no short autocorrelation time, avoid test-set prediction metrics.

Common Issues - Gotchas: what is N?

Issue:

- In traditional statistical testing, N is simply the number of independent samples, a key quantity used to determine statistical significance.
- But what is N when working with time-series?
- E.g. suppose you are estimating the autocorrelation function of a 1-D time-series and want to put error bars around reflecting your uncertainty of the estimate

Solutions:

- Except in extreme cases (pure white noise, etc), N is NOT the number of timepoints in a time-series dataset. This because timepoints are usually correlated and not independent.
- If dataset appears to have a short autocorrelation time tau, N can be approximated as the length of the time-series divided by tau (or some >1 multiple of tau to ensure good separation).
- If the dataset does not have a short autocorrelation time (e.g. it changes in a meaningful way on timescales on the order of the experiment length), run statistical testing by comparing quantities computed on the data to those on artificial control data. In this case, the relevant N is the number of random instantiations of the control data.

Common Issues - Interpreting data-driven analyses (the “Twilight Zone”)

Issue:

- Many methods now to “learn” directly from data, e.g.
  - Dimensionality reduction, clustering, fitting deep neural networks
- Often unclear what do once you’ve applied method X to your dataset
  - Also easily to spend months/years on technical details of method without knowing what you will do with the results in the end
- Tempting to think: “We should wait and see what the results look like before we decide how to interpret them. That way we won’t bias ourselves”. But:
  - Results, even when method is applied correctly, are often messy
  - Just applying method to data makes it very hard to know whether results are statistically significant or not
  - Due to nuances in many models/methods, easy to mistake property of method for property of data! E.g. clustering method with hidden “resolution” parameter.

Solutions:

- Before starting w technical details, predict at least TWO meaningful outcomes—PLOT + short written description of what plot means. This makes it much easier to interpret real data results, as long as there is sufficient signal in data.
- As always: test method on thoughtfully constructed control datasets (positive + negative) to estimate how probable it was that your result arose by chance