infoTheory toolbox

A toolbox for benchmarking information theoretic metrics

This toolbox allows users to generate simulated data with a known, ground truth mutual information between the firing of a neuron and a continuous parameter (e.g. location on a track). This firing pattern can be convolved with a florescence indicator kernel to generate a simulated florescence trace. Information theoretic measures can then be applied to the spiking or florescence data. The function randomCell contains all of the methods used to generate a cell, but each step is also modularly designed.

The steps in generating a random cell are

1. Load behavior
2. Create a rate map with known information
3. Use these to create a spike train
4. Use the spike train to generate a florescence trace
5. Analyze the data using information theory tools

# 1. Loading behavior

The loadBehaviorT function is the tool for interacting with a dataset of 574 behavioral traces recorded in the Dombeck lab of mice traversing a virtual reality, linear track, sampled at 1 kHz. The behavioral traces can be downloaded here: <https://doi.org/10.7910/DVN/SCQYKR>

## loadBehaviorT

### INPUTS

**time** - The duration of the session in minutes

### OPTIONAL PARAMETERS

**behaviorIs** ([]) - The indices of the behaviors chosen. If left empty, uses randomly selected sessions

**behaviorPath** ([]) - The location of the behavior file. If empty, is chosen via a user interface

### RETURNS

**x** - The position of the animal, sampled at 1 kH

**behaviorIs** - The indices of the behavior traces used, used to recreate the behavior

# 2. Creating a rate map with **known** information

A central tool for these methods is the generation of rate maps with a known, ground truth information (). These rate maps take the form of exponentiated splines, defined by a series of control points. The central functions are:

## genExpSpline

Generate a map that matches a Skaggs spike information rate ()

### REQUIRED INPUT

**I** - The target bit-rate (bits/AP) ()

### RETURNS

**X** - The control points across the space

**Y** - The control points in rate

**Ihat** - The actual theoretical value for the map defined by X and Y

## expSpline

Evaluates at x the exponentiated spline, normalized by the mean

### INPUTS

**x** - The values to evaluate the spline

**X** - The sorted control points for the spline, bounded by [0 1]

**Y** - The locations for the control points in log space

### OPTIONAL PARAMETERS

**trackLength** (0) – If ‘auto’ calculates from range(x). If set to some other value, normalizes to that value. If 0, does not normalize.

**trackStart** (0) – If ‘auto’ calculates from min(x).

### RETURNS

**lambda** - The values of the exponentiated spline at x ()

**d** - The normalization factor

# 3. Use (1) and (2) to generate a spike train

The rate map and behavior can be used to generate action potential firing as an inhomogenous Poisson process.

## makeSpikeTrain

Generates random spiking following a exponentiated spline rate map.

### REQUIRED INPUTS

**x** - The position of the animal

**X** - The x position of nodes in the rate map

**Y** - The y position of nodes in the rate map

### OPTIONAL PARAMETERS

**trackLength** (300 cm) - The length of the track

**goodEpochs** ('longRuns') - Can be a n x 2 matrix of start and stop times. If set to the string 'longRuns'

finds the long running epochs.

**Fs** (1e3 Hz) - The sample frequency

**FsVid** (30 Hz) - The sample frequency of the output

**meanRate** (1 Hz) - The mean firing rate during the good epochs

### RETURNS

**spk** - The spike counts sampled at Fs

**spkTs** - The times of the APs

# 4. Generate a florescence trace

The simulated florescence trace is created by convolving the firing raster with a double exponential kernel.

## spk2F

Generates a simulated florescence trace ()

### INPUTS

spk - A time trace of bined spike counts

### PARAMETERS

a - The a parameter for doubleExp. Defaults to gCamp6f kernel.

b - The b parameter for doubleExp. Defaults to gCamp6f kernel.

riseTau - The rise time for the kernel (sec). Defaults to gCamp6f kernel.

fallTau - The fall time for the kernel (sec). Defaults to gCamp6f kernel.

Fs (1000) - The sampling frequency for the spike time trace

FsVid (30) - The sampling frequency for the video

transientHeight (0.19) - The average height of transients

transientStd (0.028) - The variance of the height of the transients

noiseAmount (0.03) - The size of Gaussian shot noise

### RETURNS

Fc - The simulated florescence trace (F/F)

## florescentKernel

Solve for the double-exponential based on rise and fall times

### INPUT

riseTau - The time to the peak of the kernel (sec) ()

fallTau - The half fall time of the kernel (sec) ()

### OPTIONAL PARAMETERS

**asab** (false) - If set to true, the syntax is florescentKernel(a,b,true) , rise and fall times are estimated

### RETURNS

**a** - The a parameter for doubleExp ()

**b** - The b parameter for doubleExp ()

**riseTauHat** - The numerically solved resulting rise time ()

**fallTauHat** - The numerically solved resulting fall time ()

## doubleExp

Evaluates the double exponential at t. The double exponential is , which has a peak of 1.

INPUTS

**a** – A positive number ()

**b** – A positive number ()

**t** – The time points at which to evaluate the kernel ()

### RETURNS

**z** – The time trace ()

# 5. Analyze the signals using information theory tools

## SMGM information

SMGMMI The Skaggs, McNaughton Gothard mutual information estimate

[I,lambda\_] = SMGMMIRATE(X,Y,...) Returns the Skaggs, McNaughton, Gothard and Markus information rate of Y across X. X and Y are N long vectors, where N is the number of samples. I is the information rate in bits per second. lambda\_ is the average Y. To convert to bits per action potential, use I/lambda\_.

OPTIONAL PARAMETERS

**x\_min** (min(X)) - The lower bounds for the binning of X

**x\_max** (max(X)) - The upper bounds for the binning of X

**n\_bin** (60) - The number of bins across X

**Fs** - The sample frequency, used only for spiking data

# 6. Other tools

In addition to the SMGM method, we provide a similar means to generate cells with ground truth information using the binned estimator (Timme and Lapish, 2018).

## genBinnedDistribution

Generate a binned probablility map that matches a spike information and mean firing rate.

### REQUIRED INPUT

**I0** - The target bit-rate (bits/second)

**lambda\_** - The mean firing rate assuming uniform occupancy

### OPTIONAL INPUT

**Z0** ([]) - The initial guess for [X Y]. If empty is random.

**npoints** (5) - The number of control points for the continuous initial guess.

**n** (60) - The number of spatial bins

**m** (10) - The maximim number of spikes per frame

**Fs** (1e3) - The sampling rate

### RETURNS

**z** - The log probability distribution

**Ihat** - The actual theorhetical value for the map defined by X and Y

## makeSpikeTrainBinned

Generates a spike train from a 2D distribution

### REQUIRED INPUTS

**x** - The position of the animal

**z** - The log probability map

### OPTIONAL PARAMETERS

**trackLength** (300 cm) - The length of the track

**Fs** (1e3 Hz) - The sample frequency

**FsVid** (30 Hz) - The sample frequency of the output

### RETURNS

**spk** - The spike histogram, sampled at Fs

**spkTs** - The spike times