Bonsai.ML

Intelligent Experimental Control

Nicholas Guilbeault and Gonçalo Lopes and Joaquín Rapela

Gatsby Computational Neuroscience Unit
NeuroGEARS Ltd.

December 5, 2024

Outline

Introduction

Linear Regression

Bonsai-Python Integration

Linear Dynamical Systems

Hidden Markov Models

State-Space Decoders

Roadmap

Discussion

Outline

Introduction

Linear Regression

Bonsai-Python Integration

Linear Dynamical Systems

Hidden Markov Models

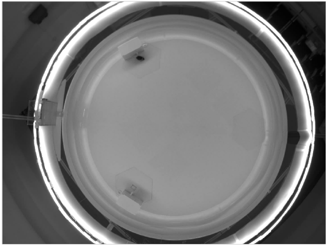
State-Space Decoders

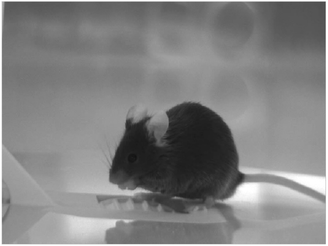
Roadmap

Discussion

AEON project

We are building a new type of experimentation: continual recording of behavioural and neural data of mice foraging in large arenas for weeks to months.





BBSRC grant: Machine learning for Neuroscience experimental control

Abstract

To understand the brain, scientists aim to explain how animal behaviour relates to neural activity. This requires the design and precise control of behavioural experiments, wherein animals perform particular tasks while experimenters either record or manipulate neural activity in specific neural circuits. Such experiments require data acquisition software that integrates and controls hardware from multiple recording devices (cameras, electrodes, sensors), and analysis tools that can interpret large and complex datasets. Progress is held back by the lack of standardised tools for design and implementation of experimental protocols, and the difficulty of integrating state-of-the-art data processing and neuroinformatics into custom experimental designs. The fields of behavioural and brain sciences have consequently suffered from both inefficiency and poor reproducibility, due to disparate data acquisition and analysis solutions created independently across laboratories. To address these challenges, we propose to extend, enhance, maintain and support **Bonsai**, a fully integrated software environment to enable cutting-edge reproducible systems neuroscience experiments using animal models, with a particular emphasis on machine-intelligence-enabled, real-time neuroinformatics methods. While Bonsai is already adopted by hundreds of scientists worldwide, we aim to extend Bonsai's functionality with a toolbox of online and offline Machine Intelligence tools for analysis of behavioural and neural data (video-based analysis of behavioural motifs, real-time and offline analysis of neural signals), and create an open-access platform for software sharing and integration with multiple programming languages. Enhancing Bonsai's ecosystem will be a game-changer for behavioural and brain science experiments by enabling new types of research, increasing and diversifying user base, and dramatically improve efficiency and reproducibility of research.

grant details

My experience in machine learning and Neuroscience

I have developed methods for:

- nonlinear regression methods to estimate receptive fields of visual cells (Rapela et al., 2006, 2010),
- Bayesian linear regression methods to understand the relation between phase concentration in the human EEG and attention (Rapela et al., 2012a,b, 2018),
- dynamical systems models to model the relation between ECoG measurements and speech production in humans (Rapela, 2016, 2017, 2018),
- unsupervised models to characterise epilepsy using Utah array recordings from humans (Rapela et al., 2019; Rapela and Todorov, 2019).

and I am the main developer of svGPFA, a method using variational inference on Gaussian processes to infer latent variables from Neuropixels population recordings.

What type of machine learning we want for Bonsai?

supervised, unsupervised and reinforcement methods

- **supervised methods** find mappings between inputs X and outputs y, like in curve fitting
- **unsupervised methods** discover structure in inputs X, without any output, like in clustering
- **reinforcement learning methods** learn relations between inputs X and actions a to maximise future rewards.

batch vs online processing

- **batch processing** all data is read (generally from files) and processed at the same time.
- **online processing** data is processes as it arrives and processed one at a time

What type of machine learning we want for Bonsai?

stationary vs non-stationary data

- stationary data has statistics (i.e., characteristics) that do not change with time.
- non-stationary data has time-varying statistics.

iid vs time-series datasets

- iid datasets contain samples that are unrelated (i.e., independent) from each other and all come from the same distribution. For example a dataset of coin tosses is iid.
- time-series datasets contain samples that are related to each other. For example a dataset of frames from a movie is a time-series one.

What type of machine learning we want for Bonsai?

probabilistic vs deterministic models

probabilistic models
: assume that variables of interest are random quantities, and seek to estimate their distribution.

deterministic models
: treat their variables of interest as deterministic quantities.

reactive vs non-reactive models

non-reactive inference models
: perform inference by following a pre-established sequence of steps, assuming availability of data before each step begins.

reactive inference models
: react to data availability performing a inference step only when data becomes available. Reactive inference models allow continuous inference in scenarios were data sources (e.g., cameras) can appear or disappear over time.

Machine learning models for Bonsai

For Bonsai we want ML models that:

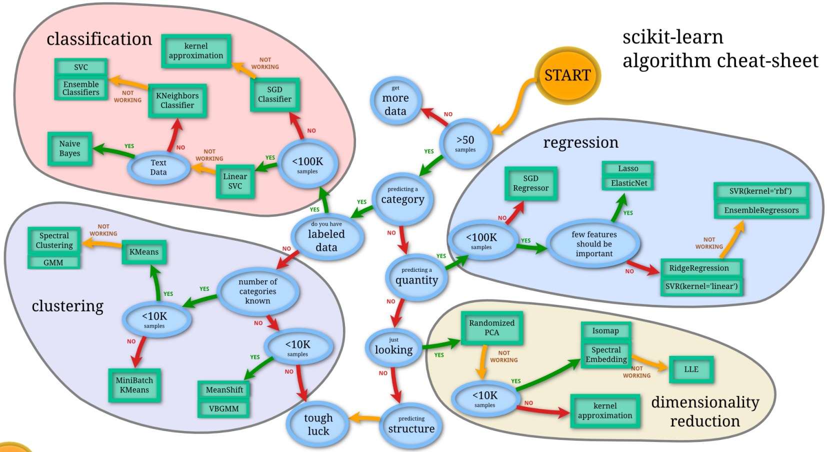
can process online data process one data item at a time, when they are produced (reactively), and can handle infinite data streams

are non-stationary can process data with time-varying statistics

can handle time-series datasets as most neuroscience datasets are time series (e.g., behavioural videos, neuron spike counts).

are reactive can continue doing inference when adding/removing data sources

Large variety of machine learning models



Back

Our choices

We decided to focus on:

- neuro applications,
- Bayesian probabilistic models.

Our choices

We decided to focus on:

- neuro applications,
- Bayesian probabilistic models.

Shall we distribute machine learning software:

- addressing a single need (e.g., deeplabcut for tracking body parts, or MOSEQ for inferring behavioural syllables), or
- generic ML software addressing multiple needs (e.g., linear dynamical systems that can be use to infer kinematics, or to discover neural latent variables from population recordings)?

We opted for the latter.

Current Bonsai.ML software

We built:

- linear dynamical systems to characterise kinematics of foraging mice,
- hidden Markov models to estimate discrete behavioural states (i.e., behavioural syllables) of foraging mice.

Current Bonsai.ML software

We built:

- linear dynamical systems to characterise kinematics of foraging mice,
- hidden Markov models to estimate discrete behavioural states (i.e., behavioural syllables) of foraging mice.

We are now working on:

- online Bayesian linear regression models to, for example, estimate receptive fields of visual neurons,
- mark-based hippocampal decoding methods to characterise replay in rodents, without spike sorting.

A new type of machine learning for Bonsai

We are excited about working on a new type of machine learning for Bonsai. One that:

- process datastreams online,
- allows non-stationary datastreams,
- is reactive.

We trust that this new type of machine learning will uncover new findings on behaviour and brain function.

Outline

Introduction

Linear Regression

Bonsai-Python Integration

Linear Dynamical Systems

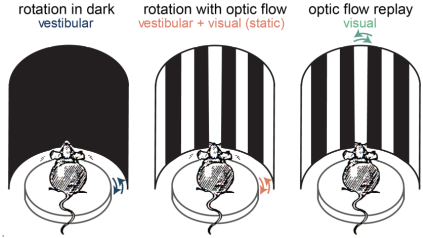
Hidden Markov Models

State-Space Decoders

Roadmap

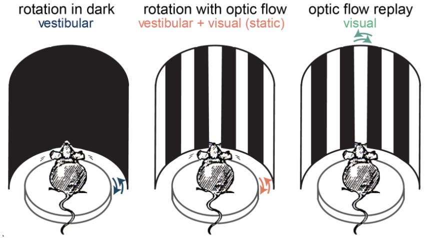
Discussion

Linear regression example

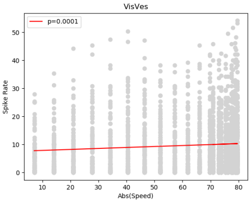


Keshavarzi et al., 2021

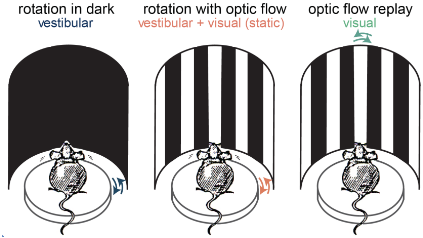
Linear regression example



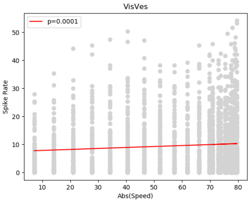
Keshavarzi et al., 2021



Linear regression example



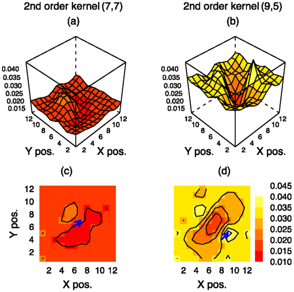
Keshavarzi et al., 2021



Is there a linear relation between the speed of rotation and the firing rate of visual cells?

Estimating nonlinear receptive fields from natural images

$$
\begin{aligned}
y(\mathbf{x}) &= k_0 + \sum_{i,j=1}^{N} k_1(i,j)x(i,j) \qquad (1)\\
&+ \sum_{i_1,j_1,i_2,j_2=1}^{N} k_2(i_1,j_1,i_2,j_2)x(i_1,j_1)x(i_2,j_2) + \dots\\
&+ \sum_{i_1,j_1,\dots,i_Q,j_Q=1}^{N} k_Q(i_1,j_1,\dots,i_Q,j_Q)x(i_1,j_1)\dots x(i_Q,j_Q) + \varepsilon\\
y(\mathbf{x}) &= \mathbf{A}\mathbf{q}(\mathbf{x}) + \varepsilon \qquad (2)
\end{aligned}
$$



Rapela et al., 2006.

Linear regression model

simple linear regression model

$$y(x_i, \mathbf{w}) = w_0 + w_1 x_i = [1, x_i] \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} = [\phi_0(x_i), \phi_1(x_i)] \begin{bmatrix} w_0 \\ w_1 \end{bmatrix}$$
$$= \phi(x_i)^\mathsf{T} \mathbf{w}$$

polynomial regression model

$$y(x_i, \mathbf{w}) = w_0 + w_1 x_i + w_2 x_i^2 + w_3 x_i^3 = [1, x_i, x_i^2, x_i^3] \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \end{bmatrix}$$
$$= [\phi_0(x_i), \phi_1(x_i), \phi_2(x_i), \phi_3(x_i)] \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \end{bmatrix} = \phi(x_i)^\mathsf{T} \mathbf{w}$$

basis functions linear regression model

$$y(x_i, \mathbf{w}) = \phi(x_i)^\mathsf{T} \mathbf{w} = \sum_{j=1}^{M} w_j \phi_j(x_i)$$

Linear regression model

$$
\begin{aligned}
\mathbf{y}(\mathbf{x},\mathbf{w}) &= \begin{bmatrix} y(x_1,\mathbf{w}) \\ y(x_2,\mathbf{w}) \\ \cdots \\ y(x_N,\mathbf{w}) \end{bmatrix} = \begin{bmatrix} \phi_1(x_1) & \phi_2(x_1) & \dots & \phi_M(x_1) \\ \phi_1(x_2) & \phi_2(x_2) & \dots & \phi_M(x_2) \\ \vdots & \vdots & \dots & \vdots \\ \phi_1(x_N) & \phi_2(x_N) & \dots & \phi_M(x_N) \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_M \end{bmatrix} \\
&= \mathbf{\Phi}\mathbf{w}
\end{aligned}
$$

where $\mathbf{y}(\mathbf{x},\mathbf{w}) \in \mathbb{R}^N, \mathbf{\Phi} \in \mathbb{R}^{N\times M}, \mathbf{w} \in \mathbb{R}^M$.

Notes

- We learned how to build a linear regression model.
- But, how can we learn the model parameters w?

Least-squares estimation of model parameters (Trefethen and Bau III, 1997)

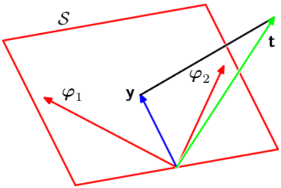
Definition 1 (Least-squares problem)

Given $\boldsymbol{\Phi} \in \mathbb{R}^{N \times M}, N \geq M, \mathbf{t} \in \mathbb{R}^{N}$*, find* $\mathbf{w} \in \mathbb{R}^{M}$ *such that* $E_{LS}(\mathbf{w}) = ||\mathbf{t} - \boldsymbol{\Phi}\mathbf{w}||_2$ *is minimised.*

Theorem 1 (Least-squares solution)

Let $\boldsymbol{\Phi} \in \mathbb{R}^{N \times M} (N \geq M)$ *and* $\mathbf{t} \in \mathbb{R}^{N}$ *be given. A vector* $\mathbf{w} \in \mathbb{R}^{M}$ *minimises* $||\mathbf{r}||_2 = ||\mathbf{t} - \boldsymbol{\Phi}\mathbf{w}||_2$*, thereby solving the least-squares problem, if and only if* $\mathbf{r} \perp \mathit{range}(\boldsymbol{\Phi})$*, that is,* $\boldsymbol{\Phi}^{\intercal}\mathbf{r} = 0$*, or equivalently,* $\boldsymbol{\Phi}^{\intercal}\boldsymbol{\Phi}\mathbf{w} = \boldsymbol{\Phi}^{\intercal}\mathbf{t}$*, or again equivalently,* $P\mathbf{t} = \boldsymbol{\Phi}\mathbf{w}$*, where* $P \in \mathbf{R}^{N \times N}$ *is the orthogonal projector onto* $\mathit{range}(A)$ *(i.e.,* $P = A\,(A^{\intercal}A)^{-1}A^{\intercal}$*).*

Figure 3.2 Geometrical interpretation of the least-squares solution, in an N-dimensional space whose axes are the values of $t_1, \ldots, t_N$. The least-squares regression function is obtained by finding the orthogonal projection of the data vector $\mathbf{t}$ onto the subspace spanned by the basis functions $\phi_j(\mathbf{x})$ in which each basis function is viewed as a vector $\boldsymbol{\varphi}_j$ of length N with elements $\phi_j(\mathbf{x}_n)$.



Bishop (2016)

Instruction to run notebooks in Google Colab

1. open a notebook from here
2. replace **github.com** by **githubtocolab.com** in the URL
3. insert a cell at the beginning of the notebook with the following content

```
!git clone https://github.com/joacorapela/gcnuBridging2023.git
%cd gcnuBridging2023
!pip install -e .
```

4. from the menu **Runtime** select **Run all**.

Code for least-squares estimation of model parameters

- overfitting
- cross validation
- larger datasets allow more complex models

Notes

- We learned how to estimate the parameters of a linear regression models by least squares.
- But, how to avoid overfitting in the estimation?

Regularised least-squares estimation of model parameters

To cope with the overfitting of least squares, we can add to the least squares optimisation criterion a term that enforces coefficients to be zero. The regularised least-squares optimisation criterion becomes:

$$E_{RLS}(\mathbf{w}) = ||\mathbf{t} - \boldsymbol{\Phi}\mathbf{w}||_2^2 + \lambda||\mathbf{w}||_2^2$$

where λ is the regularisation parameter that weights the strength of the regularisation.

Regularised least-squares estimation of model parameters

Claim 1 (Regularised least-squares estimate)

$$\mathbf{w}_{RLS} = \arg\min_{\mathbf{w}} E_{RLS}(\mathbf{w}) = \arg\min_{\mathbf{w}} ||\mathbf{t} - \mathbf{\Phi}\mathbf{w}||_2^2 + \lambda||\mathbf{w}||_2^2 = (\lambda\mathbf{I} + \mathbf{\Phi}^\intercal\mathbf{\Phi})^{-1}\mathbf{\Phi}^\intercal\mathbf{t}$$

Proof.

Since $E_{RLS}(\mathbf{w})$ is a polynomial of order two on the elements of $\mathbf{w}$ (i.e., a quadratic form), we can use the *Completing the Squares* technique below to find its minimum.

$$\begin{aligned}
\boldsymbol{\mu} &= \arg\max_{\mathbf{w}} \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \Sigma) = \arg\max_{\mathbf{w}} \log \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \Sigma) \\
&= \arg\max_{\mathbf{w}} \{K - \frac{1}{2}(-2\boldsymbol{\mu}^\intercal\Sigma^{-1}\mathbf{w} + \mathbf{w}\Sigma^{-1}\mathbf{w})\} && (3) \\
&= \arg\min_{\mathbf{w}} \{-K + \frac{1}{2}(-2\boldsymbol{\mu}^\intercal\Sigma^{-1}\mathbf{w} + \mathbf{w}\Sigma^{-1}\mathbf{w})\} \\
&= \arg\min_{\mathbf{w}} \{K_1 - 2\boldsymbol{\mu}^\intercal\Sigma^{-1}\mathbf{w} + \mathbf{w}\Sigma^{-1}\mathbf{w}\} && (4)
\end{aligned}$$

To find the minimum of a quadratic form, we write it in the form of the terms inside the curly brackets of Eq. 4, and the term corresponding to $\boldsymbol{\mu}$ will be the minimum.

Regularised least-squares estimation of model parameters

Proof.

Let's write E_{RLS} in the form of the terms inside the curly brackets of Eq. 4.

$$\begin{aligned} E_{RLS} &= ||\mathbf{t} - \mathbf{\Phi w}||_2^2 + \lambda||\mathbf{w}||_2^2 = (\mathbf{t} - \mathbf{\Phi w})^{\mathsf{T}}(\mathbf{t} - \mathbf{\Phi w}) + \lambda \mathbf{w}^{\mathsf{T}}\mathbf{w} \\ &= \mathbf{t}^{\mathsf{T}}\mathbf{t} - 2\mathbf{t}^{\mathsf{T}}\mathbf{\Phi w} + \mathbf{w}^{\mathsf{T}}\mathbf{\Phi}^{\mathsf{T}}\mathbf{\Phi w} + \lambda \mathbf{w}^{\mathsf{T}}\mathbf{w} \\ &= \mathbf{t}^{\mathsf{T}}\mathbf{t} - 2\mathbf{t}^{\mathsf{T}}\mathbf{\Phi w} + \mathbf{w}^{\mathsf{T}}(\mathbf{\Phi}^{\mathsf{T}}\mathbf{\Phi} + \lambda \mathbf{I}_M)\mathbf{w} \end{aligned}$$

Calling

$$\begin{aligned} \Sigma^{-1} &= \mathbf{\Phi}^{\mathsf{T}}\mathbf{\Phi} + \lambda \mathbf{I}_M \\ \boldsymbol{\mu}^{\mathsf{T}}\Sigma^{-1} &= \mathbf{t}^{\mathsf{T}}\mathbf{\Phi} \text{ or } \boldsymbol{\mu}^{\mathsf{T}} = \mathbf{t}^{\mathsf{T}}\mathbf{\Phi}\Sigma \text{ or } \boldsymbol{\mu} = \Sigma\mathbf{\Phi}^{\mathsf{T}}\mathbf{t} = \left(\mathbf{\Phi}^{\mathsf{T}}\mathbf{\Phi} + \lambda \mathbf{I}_M\right)^{-1}\mathbf{\Phi}^{\mathsf{T}}\mathbf{t} \end{aligned}$$

we can express

$$E_{RLS} = K + 2\boldsymbol{\mu}^{\mathsf{T}}\Sigma^{-1}\mathbf{w} + \mathbf{w}\Sigma^{-1}\mathbf{w}$$

Then

$$\mathbf{w}_{RLS} = \arg\min_{\mathbf{w}} E_{RLS}(\mathbf{w}) = \boldsymbol{\mu} = \left(\mathbf{\Phi}^{\mathsf{T}}\mathbf{\Phi} + \lambda \mathbf{I}_M\right)^{-1}\mathbf{\Phi}^{\mathsf{T}}\mathbf{t}$$

□

Code for regularised least-squares estimation of model parameters

- control of overfitting

Notes

- So far we have assummed deterministic parameters. But it is useful to treat them as random quantities.
- How can we build linear regression models for random parameters?

Maximum-likelihood estimation of model parameters

Definition 2 (Likelihood function)

For a statistical model characterised by a probability density function $p(\mathbf{x}|\theta)$ *(or probability mass function* $P_\theta(X = \mathbf{x})$*) the likelihood function is a function of the parameters* θ*,* $\mathcal{L}(\theta) = p(\mathbf{x}|\theta)$ *(or* $\mathcal{L}(\theta) = P_\theta(\mathbf{x})$*).*

Definition 3 (Maximum likelihood parameters estimates)

The maximum likelihood parameters estimates are the parameters that maximise the likelihood function.

$$\theta_{ML} = \arg\max_\theta \mathcal{L}(\theta)$$

Maximum-likelihood estimation for the basis function linear regression model

We seek the parameter $\mathbf{w}_{ML}$ and β_{ML} that maximised the following likelihood function

$$\mathcal{L}(\mathbf{w}, \beta) = p(\mathbf{t}|\mathbf{w}, \beta) = \mathcal{N}(\mathbf{t}|\mathbf{\Phi}\mathbf{w}, \beta^{-1} I_N) \tag{5}$$

They are

$$\mathbf{w}_{ML} = (\mathbf{\Phi}^\intercal \mathbf{\Phi})^{-1} \mathbf{\Phi}^\intercal \mathbf{t} \tag{6}$$

$$\frac{1}{\beta_{ML}} = \frac{1}{N} ||\mathbf{t} - \mathbf{\Phi}\mathbf{w}_{ML}||_2^2 \tag{7}$$

- first regression method that assumes random observations
- if the likelihood function is assumed to be Normal, maximum-likelihood and least-squares coefficients estimates are equal.

Maximum likelihood: exercise

Exercise 1

Derive the formulas for the maximum likelihood estimates of the coefficients, $\mathbf{w}$*, and noise precision,* β*, of the basis functions linear regression model given in Eqs. 6 and 7.*

Solution.

$$
\begin{aligned}
\mathcal{L}(\mathbf{w}, \beta) &= p(\mathbf{t}|\mathbf{w}, \beta) = \mathcal{N}(\mathbf{t}|\mathbf{\Phi}\mathbf{w}, \beta^{-1}\mathbf{I}) \\
&= \frac{1}{(2\pi)^{\frac{N}{2}} |\beta^{-1}\mathbf{I}|^{\frac{1}{2}}} \exp\left(-\frac{\beta}{2}||\mathbf{t} - \mathbf{\Phi}\mathbf{w}||_2^2\right) \\
\log \mathcal{L}(\mathbf{w}, \beta) &= -\frac{N}{2}\log 2\pi + \frac{N}{2}\log\beta - \frac{\beta}{2}||\mathbf{t} - \mathbf{\Phi}\mathbf{w}||_2^2 \\
\mathbf{w}_{ML} &= \arg\max_{\mathbf{w}} \log \mathcal{L}(\mathbf{w}, \beta) = \arg\min_{\mathbf{w}} ||\mathbf{t} - \mathbf{\Phi}\mathbf{w}||_2^2 = (\mathbf{\Phi}^{\mathsf{T}}\mathbf{\Phi})^{-1}\mathbf{\Phi}^{\mathsf{T}}\mathbf{t} \\
\frac{\partial}{\partial\beta}\log p(\mathbf{t}|\mathbf{w}_{ML}, \beta) &= \frac{N}{2}\frac{1}{\beta} - \frac{1}{2}||\mathbf{t} - \mathbf{\Phi}\mathbf{w}_{ML}||_2^2 \\
\frac{\partial}{\partial\beta}\log p(\mathbf{t}|\mathbf{w}_{ML}, \beta_{ML}) &= 0 \quad \text{iff} \quad \frac{1}{\beta_{ML}} = \frac{1}{N}||\mathbf{t} - \mathbf{\Phi}\mathbf{w}_{ML}||_2^2
\end{aligned}
$$

Notes

- We have learned how to estimate random parameters in linear regression models.
- How can we incorporate prior assumptions in this estimation?

Batch Bayesian linear regression: posterior distribution of parameters

In Bayesian linear regression we seek the posterior distribution of the weights of the linear regression model, $\mathbf{w}$, given the observations, which is proportional to the product of the likelihood function, $p(\mathbf{t}|\mathbf{w})$, and the prior, $p(\mathbf{w})$; i.e.,

$$p(\mathbf{w}|\mathbf{t}) \propto p(\mathbf{t}|\mathbf{w})p(\mathbf{w}) \tag{8}$$

To calculate this posterior below we use the likelihood function defined in Eq. 5 and the following prior

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{0}, \alpha^{-1}\mathbf{I})$$

Using the expression of the conditional of the Linear Gaussian model we obtain

$$\begin{aligned} p(\mathbf{w}|\mathbf{t}) &= \mathcal{N}(\mathbf{w}|\mathbf{m}_N, \mathbf{S}_N) \\ \mathbf{m}_N &= \beta \mathbf{S}_N \mathbf{\Phi}^\intercal \mathbf{t} && (9) \\ \mathbf{S}_N^{-1} &= \alpha \mathbf{I} + \beta \mathbf{\Phi}^\intercal \mathbf{\Phi} && (10) \end{aligned}$$

Batch Bayesian linear regression: exercise

Exercise 2

Derive the formulas for the Bayesian posterior mean (Eq. 9) and covariance (Eq. 10) of the basis function linear regression model.

Exercise 3

Show that

$$\log p(\mathbf{w}|\boldsymbol{t}) = K - \frac{\beta}{2}||\mathbf{t} - \boldsymbol{\Phi}\mathbf{w}||_2^2 - \frac{\alpha}{2}||\mathbf{w}||_2^2 \tag{11}$$

Therefore, the maximum-a-posteriori parameters of the basis function linear regression model are the solution of the regularised least-squares problem with $\lambda = \alpha/\beta$.
Note that, as we will show next, Bayesian linear regression uses the full posterior of the parameters to make predictions or to do model selection, and not just the maximum-a-posteriori parameters.

Batch Bayesian linear regression: demo code

Available here

Notes

- Now we know how to do batch Bayesian linear regression.
- However, in Bonsai we don't want to work with batch data. We want to do online processing of infinite data streams. How can we do this?

Recursive update of posterior distribution of the parameters for conditionally independent observations

Claim 2 (recursive update)

If the observations, $\{\mathbf{t}_1, \ldots, \mathbf{t}_n, \ldots\}$, are linearly independent when conditioned on the model parameters, $\boldsymbol{\theta}$, then for any $n \in \mathbb{N}$

$$p(\boldsymbol{\theta}|\mathbf{t}_1, \ldots, \mathbf{t}_n) = K \; p(\mathbf{t}_n|\boldsymbol{\theta}) p(\boldsymbol{\theta}|\mathbf{t}_1, \ldots, \mathbf{t}_{n-1}) \tag{12}$$

where K is a quantity that does not depend on $\boldsymbol{\theta}$.

Recursive update of posterior distribution of the parameters for conditionally independent observations

Proof.

By induction on $H_n : p(\boldsymbol{\theta}|\mathbf{t}_1, \ldots, \mathbf{t}_n) = K\ p(\mathbf{t}_n|\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathbf{t}_1, \ldots, \mathbf{t}_{n-1})$.

H_1

$$p(\boldsymbol{\theta}|\mathbf{t}_1) = \frac{p(\boldsymbol{\theta}, \mathbf{t}_1)}{p(\mathbf{t}_1)} = \frac{p(\mathbf{t}_1|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathbf{t}_1)} = K\ p(\mathbf{t}_1|\boldsymbol{\theta})p(\boldsymbol{\theta})$$

$H_n \to H_{n+1}$

$$\begin{aligned}
p(\boldsymbol{\theta}|\mathbf{t}_1, \ldots, \mathbf{t}_{n+1}) &= \frac{p(\boldsymbol{\theta}, \mathbf{t}_1, \ldots, \mathbf{t}_{n+1})}{p(\mathbf{t}_1, \ldots, \mathbf{t}_{n+1})} \\
&= \frac{p(\mathbf{t}_{n+1}|\boldsymbol{\theta}, \mathbf{t}_1, \ldots, \mathbf{t}_n)p(\boldsymbol{\theta}, \mathbf{t}_1, \ldots, \mathbf{t}_n)}{p(\mathbf{t}_1 \ldots, \mathbf{t}_{n+1})} \\
&= \frac{p(\mathbf{t}_{n+1}|\boldsymbol{\theta})p(\boldsymbol{\theta}, \mathbf{t}_1, \ldots, \mathbf{t}_n)}{p(\mathbf{t}_1 \ldots, \mathbf{t}_{n+1})} \\
&= \frac{p(\mathbf{t}_{n+1}|\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathbf{t}_1, \ldots, \mathbf{t}_n)p(\mathbf{t}_1, \ldots, \mathbf{t}_n)}{p(\mathbf{t}_1 \ldots, \mathbf{t}_{n+1})} \\
&= K\ p(\mathbf{t}_{n+1}|\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathbf{t}_1, \ldots, \mathbf{t}_n)
\end{aligned}$$

Note: the third equality above holds because the observations are assumed to be conditional independent given the parameters.

□

Recursive update of the posterior distribution of the parameters for a conjugate prior

Claim 3

If

$$P(\mathbf{w}|\mathbf{t}_1, \ldots, \mathbf{t}_n) = \mathcal{N}(\mathbf{w}|\mathbf{m}_n, \mathbf{S}_n) \tag{13}$$

$$P(\mathbf{t}_{n+1}|\mathbf{w}) = \mathcal{N}(\mathbf{t}_{n+1}|\mathbf{\Phi}\mathbf{w}, \beta^{-1}\mathbf{I}) \tag{14}$$

then

$$P(\mathbf{w}|\mathbf{t}_1, \ldots, \mathbf{t}_{n+1}) = \mathcal{N}(\mathbf{w}|\mathbf{m}_{n+1}, \mathbf{S}_{n+1})$$

with

$$S_{n+1} = S_n - (\beta^{-1} + \phi(\mathbf{x}_{n+1})^\intercal S_n \phi(\mathbf{x}_{n+1}))^{-1} S_n \phi(\mathbf{x}_{n+1}) \phi(\mathbf{x}_{n+1})^\intercal S_n \tag{15}$$

$$\begin{aligned} m_{n+1} &= \beta t_{n+1} S_{n+1} \phi(\mathbf{x}_{n+1}) + \mathbf{m}_n - \\ &\quad (\beta^{-1} + \phi(\mathbf{x}_{n+1})^\intercal S_n \phi(\mathbf{x}_{n+1}))^{-1} \phi(x_{n+1})^\intercal \mathbf{m}_n S_n \phi(x_{n+1}) \end{aligned} \tag{16}$$

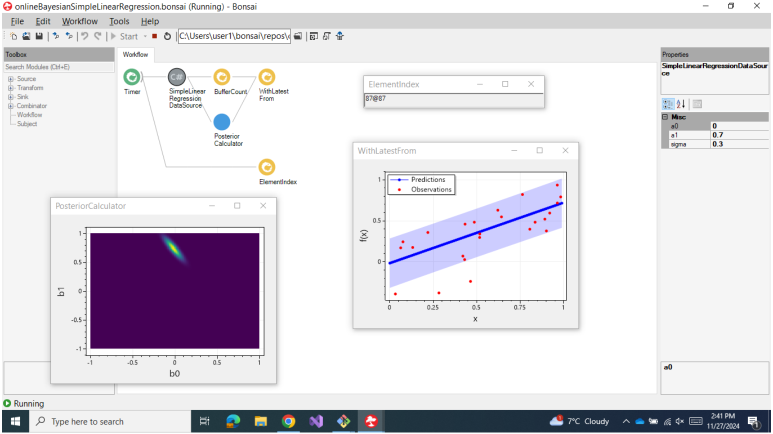
Python code for online Bayesian linear regression

Available here.

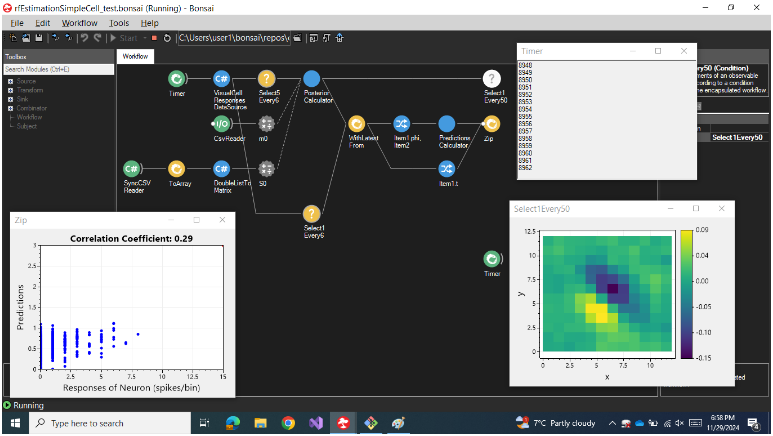
Notes

- We have learned how to do linear regression in Python.
- However, we are in the first Bonsai conference. Let's do online Bayesian linear regression in Bonsai.

Online Bayesian linear regression in Bonsai

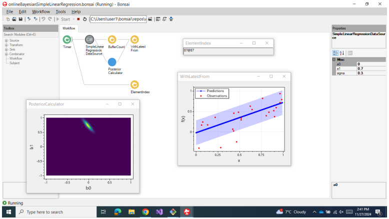


Estimating receptive fields of cortical visual neurons in Bonsai



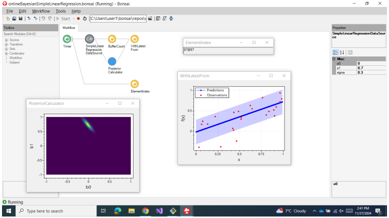
Is online Bayesian linear regression adaptive?

Let's try it:



Is online Bayesian linear regression adaptive?

Let's try it:

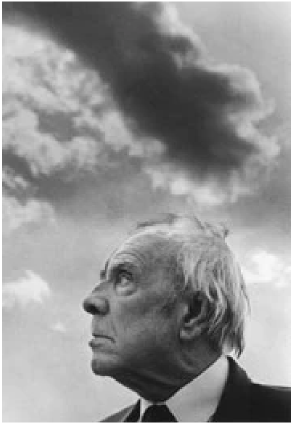


Above we saw that:

$$\log p(\mathbf{w}|\boldsymbol{t}) = K - \frac{\beta}{2}||\mathbf{t} - \boldsymbol{\Phi}\mathbf{w}||_2^2 - \frac{\alpha}{2}||\mathbf{w}||_2^2$$

So, online Bayesian linear regression should not be adaptive.

Funes, the memorious



Borges, J. L. (1962). Funes, the memorious (pp. 59-66).

Recursive least squares

$$\mathbf{w}_i = \arg\min_{\mathbf{w}} \left[\lambda^{(i+1)} ||\mathbf{w}||^2 + \sum_{j=0}^{i} \lambda^{i-j} |y(\mathbf{x}_j) - \phi^\intercal(\mathbf{x}_j)\mathbf{w}|^2 \right]$$

with

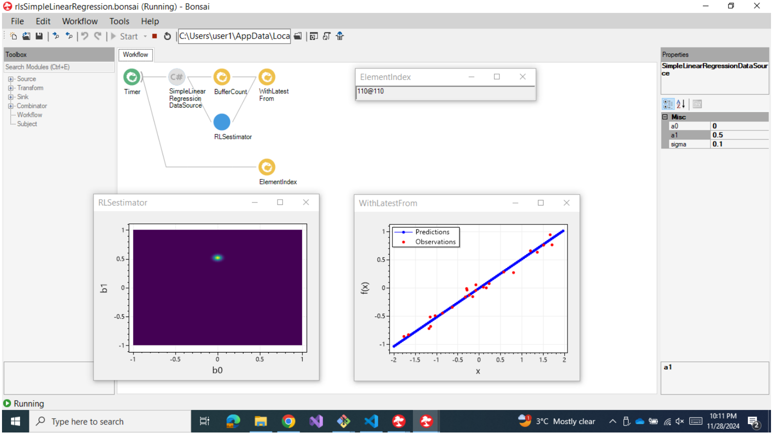
$$\mathbf{w}_i = \mathbf{w}_{i-1} + P_i\, \phi(\mathbf{x}_i) \left[y(\mathbf{x}_i) - \phi^\intercal(\mathbf{x}_i)\, \mathbf{w}_{i-1} \right]$$

$$P_i = \lambda^{-1} \left[P_{i-1} - \frac{\lambda^{-1} P_{i-1} \phi_i \phi_i^\intercal P_i^{-1}}{1 + \lambda^{-1} \phi(\mathbf{x}_i^\intercal P_{i-1} \phi(\mathbf{x}_i)} \right]$$

$$P_1 = \epsilon^{-1} I$$

$$0 \ll \lambda \leq 1$$

Recursive least squares in Bonsai



Outline

Introduction

Linear Regression

Bonsai-Python Integration

Linear Dynamical Systems

Hidden Markov Models

State-Space Decoders

Roadmap

Discussion

Outline

Introduction

Linear Regression

Bonsai-Python Integration

Linear Dynamical Systems

Hidden Markov Models

State-Space Decoders

Roadmap

Discussion

Outline

Introduction

Linear Regression

Bonsai-Python Integration

Linear Dynamical Systems

Hidden Markov Models

State-Space Decoders

Roadmap

Discussion

Outline

Introduction

Linear Regression

Bonsai-Python Integration

Linear Dynamical Systems

Hidden Markov Models

State-Space Decoders

Roadmap

Discussion

Outline

Introduction

Linear Regression

Bonsai-Python Integration

Linear Dynamical Systems

Hidden Markov Models

State-Space Decoders

Roadmap

Discussion

Roadmap

See here.

Outline

Introduction

Linear Regression

Bonsai-Python Integration

Linear Dynamical Systems

Hidden Markov Models

State-Space Decoders

Roadmap

Discussion

Dissemination

- we may add excellent machine learning functionality to Bonsai that nobody uses, since some knowledge of machine learning is needed to use generic machine learning models.

Dissemination

- we may add excellent machine learning functionality to Bonsai that nobody uses, since some knowledge of machine learning is needed to use generic machine learning models.
- reviewer comment: *by adding machine learning methods to Bonsai, you may be doing more harm than good*.

Dissemination

- we may add excellent machine learning functionality to Bonsai that nobody uses, since some knowledge of machine learning is needed to use generic machine learning models.
- reviewer comment: *by adding machine learning methods to Bonsai, you may be doing more harm than good*.
- focus on documentation, dissemination and early collaborations.

Dissemination

- we may add excellent machine learning functionality to Bonsai that nobody uses, since some knowledge of machine learning is needed to use generic machine learning models.
- reviewer comment: *by adding machine learning methods to Bonsai, you may be doing more harm than good*.
- focus on documentation, dissemination and early collaborations.
- problem driven or tool driven? I prefer to be problem driven, collaborate with experimental scientists using Bonsai.ML to solve specific problems, and then disseminate the Bonsai tool plus with the example problems we solved together.

Dissemination

- we may add excellent machine learning functionality to Bonsai that nobody uses, since some knowledge of machine learning is needed to use generic machine learning models.
- reviewer comment: *by adding machine learning methods to Bonsai, you may be doing more harm than good*.
- focus on documentation, dissemination and early collaborations.
- problem driven or tool driven? I prefer to be problem driven, collaborate with experimental scientists using Bonsai.ML to solve specific problems, and then disseminate the Bonsai tool plus with the example problems we solved together.
- we have the privilege of working on one of the most prestigious centres for experimental neuroscience. We want to apply our Bonsai.ML tools to help them process their data.

Dissemination

- we may add excellent machine learning functionality to Bonsai that nobody uses, since some knowledge of machine learning is needed to use generic machine learning models.
- reviewer comment: *by adding machine learning methods to Bonsai, you may be doing more harm than good*.
- focus on documentation, dissemination and early collaborations.
- problem driven or tool driven? I prefer to be problem driven, collaborate with experimental scientists using Bonsai.ML to solve specific problems, and then disseminate the Bonsai tool plus with the example problems we solved together.
- we have the privilege of working on one of the most prestigious centres for experimental neuroscience. We want to apply our Bonsai.ML tools to help them process their data.
- or perhaps we should be patient, add excellent ML methods to Bonsai, and maybe people will use them.

Other topics

- how to disseminate Bonsai.ML?
  - distribute methods, with high-quality code, documentation and examples.
  - publish papers.
  - collaborate with experimentalists to use Bonsai.ML to tackle interesting neuroscience problems.
  - train the trainers.
  - train Bonsai users on basic machine learning topics.
  - find a killer Bonsai.ML application.
- suggestions for Bonsai.ML roadmap?
- suggestions for:
  - new ML models to integrate into Bonsai
  - new applications areas to investigate with Bonsai.ML
    - testing causality of brain activity patterns on behaviour
    - online selection of data to save (e.g. cameras)
    - Bonsai for online data analysis (e.g., Terry Sejnowski: for very large datasets, retrieval could be very onerous, so you'd better analyse data online).

References

Bishop, C. M. (2016). *Pattern recognition and machine learning*. Springer-Verlag New York.

Rapela, J. (2016). Entrainment of traveling waves to rhythmic motor acts.

Rapela, J. (2017). Rhythmic production of consonant-vowel syllables synchronizes traveling waves in speech-processing brain regions.

Rapela, J. (2018). Traveling waves appear and disappear in unison with produced speech.

Rapela, J., Felsen, G., Touryan, J., Mendel, J. M., and Grzywacz, N. M. (2010). ePPR: a new startegy for the characterization of sensory cells from input/output data. *Network: Computation in Neural Systems*, 21(1-2):35–90.

Rapela, J., Gramann, K., Westerfield, M., Townsend, J., and Makeig, S. (2012a). Brain oscillations in switching vs. focusing audio-visual attention. In *Proc. of 2012 IEEE Engineering in Medicine and Biology Society (EMBC'12)*, San Diego, CA.

Rapela, J., Lin, T.-Y., Westerfield, M., Jung, T.-P., and Townsend, J. (2012b). Assisting autistic children with wireless EOG technology. In *Proc. of 2012 IEEE Engineering in Medicine and Biology Society (EMBC'12)*, San Diego, CA.

Rapela, J., Mendel, J., and Grzywacz, N. (2006). Estimating nonlinear receptive fields from natural images. *Journal of Vision*, 6(4):441–474.

Rapela, J., Proix, T., Todorov, D., and Truccolo, W. (2019). Uncovering low-dimensional structures of high-dimensional electrophysiological recordings in epilepsy. In *Engineering in Medicine and Biology Society (EMBC), 2019 41st Annual International Conference of the IEEE*, Berlin, Germany. IEEE.

Rapela, J. and Todorov, D. (2019). Hidden markov models applied to LFPs from layer two and three of human cortex reveal highly stereotypical discrete states in epileptic seizures separated by more than an hour. Accepted at the 41st Engineering in Medicine and Biology Conference of the IEEE; main text `https://sccn.ucsd.edu/~rapela/papers/EMBC19_Rapela_and_Todorov_EMBC_2019.pdf` and supplemental information `embc19_hmm_supplemental.pdf`.

Rapela, J., Westerfield, M., and Townsend, J. (2018). A new single-trial foreperiod effect on inter-trial phase coherence. Part I: existence and relevance. *Neural Computation*, 30(9):2348–2383.

Trefethen, L. n. and Bau III, D. (1997). Numerical linear algebra.