Probabilistic models for neural data: From single neurons to population dynamics

NEUROBIO 316QC

Jan Drugowitsch

jan_drugowitsch@hms.harvard.edu

Session 7: Kernel methods

Today

Q&A about previous session

Paper discussion (~1h)

Kernel methods and Gaussian Processes (~30min)

Overview

Kernel methods

Gaussian processes

- What makes them possible: marginalization of multivariate Gaussians
- Gaussian progress regression
- Learning hyperparameters, extensions, etc.

Overview

Kernel methods

Gaussian processes

- What makes them possible: marginalization of multivariate Gaussians
- Gaussian progress regression
- Learning hyperparameters, extensions, etc.

Linear regression by kernel methods

Assume $p(y_n|\boldsymbol{x}_n) = N(y_n|\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}_n), \sigma^2)$ and $p(\boldsymbol{w}) = (\boldsymbol{w}|\boldsymbol{0}, \sigma^2\lambda^{-1})$ (i.e., L2 regularization)

Negative log-posterior & MAP solution

$$J(\boldsymbol{w}) = \frac{1}{2\sigma^2}\|\boldsymbol{\Phi w} - \boldsymbol{y}\|^2 + \frac{\lambda}{2\sigma^2}\boldsymbol{w}^T\boldsymbol{w} + \text{const.} \longrightarrow \hat{\boldsymbol{w}}_{MAP} = -\frac{1}{\lambda}\boldsymbol{\Phi}^T(\boldsymbol{\Phi}\hat{\boldsymbol{w}}_{MAP} - \boldsymbol{y}) = \boldsymbol{\Phi}^T\boldsymbol{a}$$

Substitute $\hat{\boldsymbol{w}}_{MAP}$ into negative log-posterior, becomes function of $\boldsymbol{a}$

$$J(\boldsymbol{a}) = \frac{1}{2\sigma^2}\boldsymbol{a}^T\boldsymbol{K}\boldsymbol{K}\boldsymbol{a} - \frac{1}{\sigma^2}\boldsymbol{a}^T\boldsymbol{K}\boldsymbol{y} + \frac{1}{2\sigma^2}\boldsymbol{y}^T\boldsymbol{y} + \frac{\lambda}{2\sigma^2}\boldsymbol{a}^T\boldsymbol{K}\boldsymbol{a}$$

with *Gram* matrix $\boldsymbol{K} = \boldsymbol{\Phi}\boldsymbol{\Phi}^T$

$$\hat{\boldsymbol{a}}_{MAP} = (\boldsymbol{K} + \lambda\boldsymbol{I})^{-1}\boldsymbol{y}$$

$$K_{nm} = \underbrace{\boldsymbol{\phi}(\boldsymbol{x}_n)^T\boldsymbol{\phi}(\boldsymbol{x}_m)} = k(\boldsymbol{x}_n, \boldsymbol{x}_m)$$

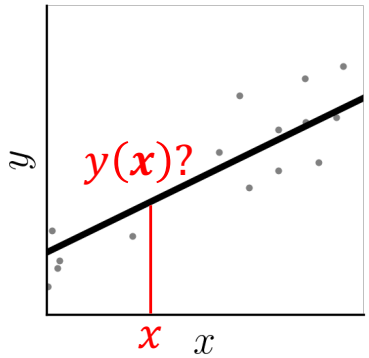
~similarity between $\boldsymbol{x}_n$ and $\boldsymbol{x}_m$ → kernel $k(\cdot,\cdot)$

Predict $\boldsymbol{y}$ for new $\boldsymbol{x}$

similarity of $\boldsymbol{x}$ to training "inputs"

$$y(\boldsymbol{x}) = \hat{\boldsymbol{w}}_{MAP}^T\boldsymbol{\phi}(\boldsymbol{x}) = \boldsymbol{\phi}^T(\boldsymbol{x})\boldsymbol{\Phi}^T\boldsymbol{a} = \underbrace{\boldsymbol{k}(\boldsymbol{x})^T(\boldsymbol{K} + \lambda\boldsymbol{I})^{-1}\boldsymbol{y}}$$

weighted combination of training "outputs"



Valid kernel functions & computational complexity

$K_{nm} = k(\boldsymbol{x}_n, \boldsymbol{x}_m) = \boldsymbol{\phi}(\boldsymbol{x}_n)^T \boldsymbol{\phi}(\boldsymbol{x}_m)$, but can specify $k(\boldsymbol{x}_n, \boldsymbol{x}_m)$ without $\boldsymbol{\phi}(\cdot)$

Gram matrix needs to be positive definite, i.e., $\boldsymbol{a}^T \boldsymbol{K} \boldsymbol{a} \geq 0$ for all $\boldsymbol{a}$

Examples

Polynomial kernel $k(\boldsymbol{x}, \boldsymbol{x}') = (\boldsymbol{x}^T \boldsymbol{x}' + c)^M$

"Gaussian" kernel $k(\boldsymbol{x}, \boldsymbol{x}') = \exp(-\|\boldsymbol{x} - \boldsymbol{x}'\|^2 / 2\sigma^2)$

Generalized "Gaussian" kernel $k(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\frac{1}{2\sigma^2}\left(\tilde{k}(\boldsymbol{x}, \boldsymbol{x}) + \tilde{k}(\boldsymbol{x}', \boldsymbol{x}') - 2\tilde{k}(\boldsymbol{x}, \boldsymbol{x}')\right)\right)$

Computational complexity

Assume N "inputs", $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$, D-dimensional basis function $\boldsymbol{\phi}(\cdot)$ and $\boldsymbol{w}$ (usually $N \gg D$)

"standard" linear regression $y(\boldsymbol{x}) = \boldsymbol{\phi}^T(\boldsymbol{x}) \widehat{\boldsymbol{w}}_{MAP} = \boldsymbol{\phi}^T(x) \underbrace{(\boldsymbol{\Phi}^T \boldsymbol{\Phi} + \lambda \boldsymbol{I})}_{D \times D}^{-1} \boldsymbol{\Phi}^{\mathrm{T}} \mathbf{y}$

Kernel regression $y(\boldsymbol{x}) = \boldsymbol{\phi}^T(\boldsymbol{x}) \boldsymbol{\Phi}^{\mathrm{T}} \widehat{\boldsymbol{a}}_{MAP} = \boldsymbol{k}(\boldsymbol{x})^T \underbrace{(\boldsymbol{\Phi} \boldsymbol{\Phi}^T + \lambda \boldsymbol{I})}_{N \times N}^{-1} \boldsymbol{y}$

Overview

Kernel methods

Gaussian processes

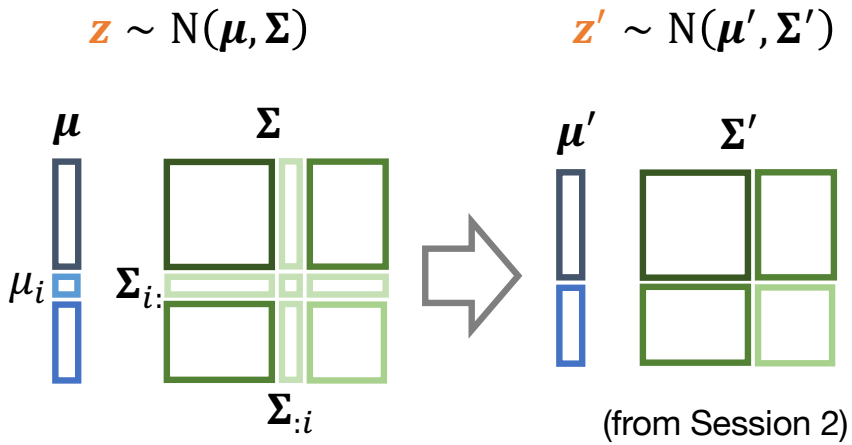
- What makes them possible: marginalization of multivariate Gaussians
- Gaussian progress regression
- Learning hyperparameters, extensions, etc.

Marginalizing multivariate Gaussians → Gaussian process

Marginalization: 'removing' z_i from $\mathbf{z}$ by

$$p(\mathbf{z}') = p(z_1, \dots, z_{i-1}, z_{i+1}, \dots, z_N) = \int p(\mathbf{z}) dz_i$$

does not change moments of $z_{j \neq i}$
(as moments of z_j only depend on μ_j and $\boldsymbol{a}_{j:}$)



(from Session 2)

Gaussian process $z(\boldsymbol{x})$ (continuous function)

Distribution over any $z(\boldsymbol{x}_1), \dots, z(\boldsymbol{x}_N)$ is multivariate Gaussian

$$\boldsymbol{z} \equiv z(\boldsymbol{x}_1), \dots, z(\boldsymbol{x}_N) \sim \mathrm{N}(0, \boldsymbol{K}) \qquad \text{with covariance } K_{nm} = k(\boldsymbol{x}_n, \boldsymbol{x}_m)$$

positive (semi-)definite — expected similarity of $z(\boldsymbol{x}_n)$ and $z(\boldsymbol{x}_m)$

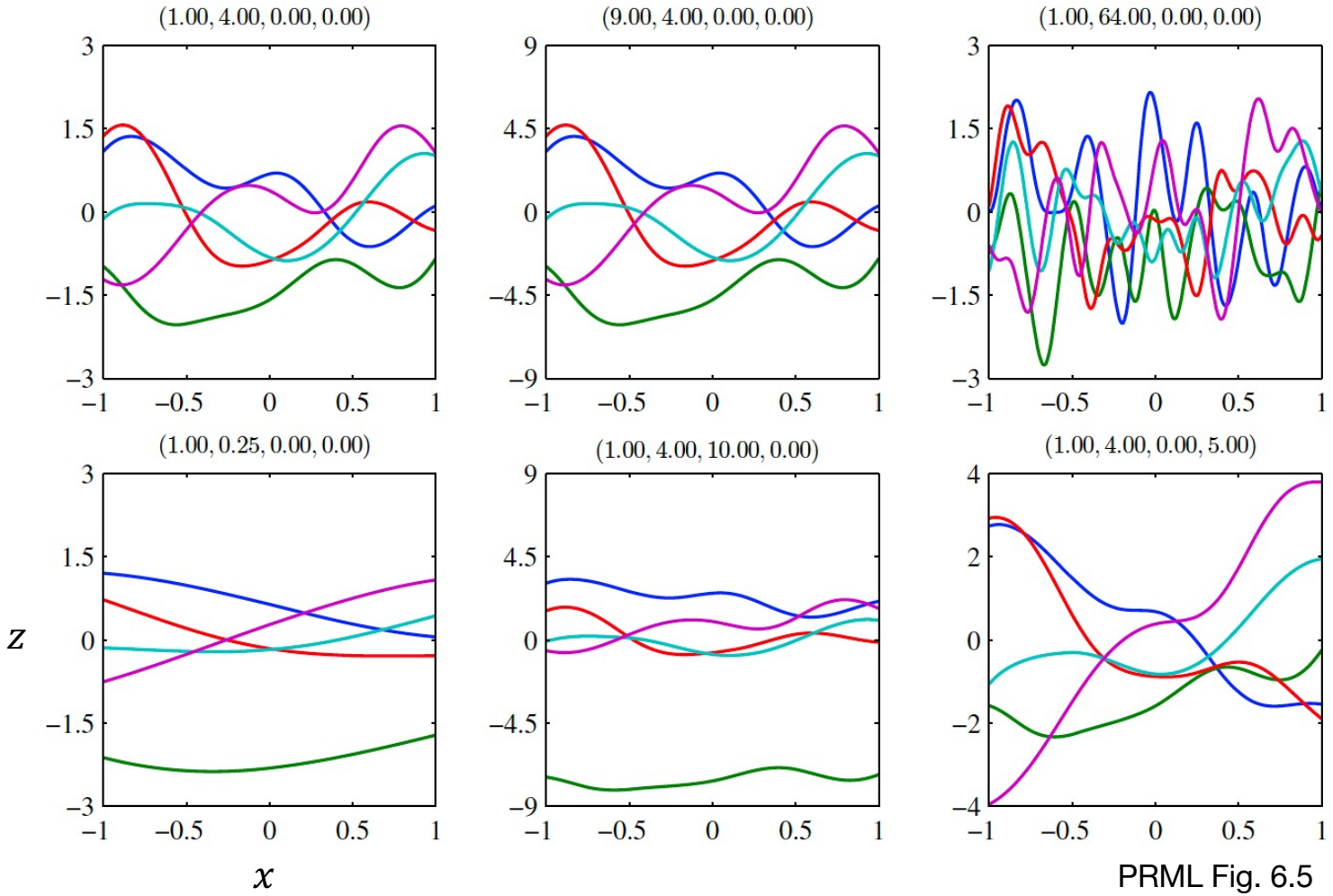
Construction possible as "marginalizing out" unobserved $z(\boldsymbol{x})$'s yields Gaussian

Also: adding another $z(\boldsymbol{x}_{N+1})$ again yields Gaussian

Draws from Gaussian processes

magnitude — length-scale — long-range correlations — linear in $\boldsymbol{x}$ and $\boldsymbol{x}'$

$$k(\boldsymbol{x}, \boldsymbol{x}') = \theta_0 \exp\left(-\frac{\theta_1}{2}\|\boldsymbol{x} - \boldsymbol{x}'\|^2\right) + \theta_2 + \theta_3 \boldsymbol{x}^T \boldsymbol{x}'$$



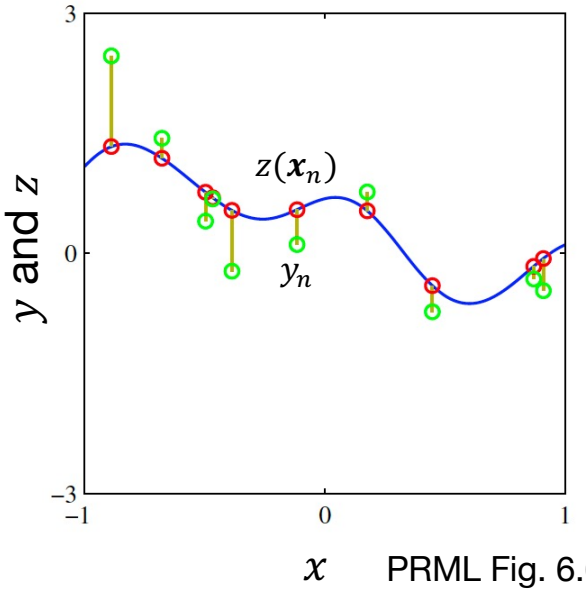
PRML Fig. 6.5

Gaussian progress regression

Model data as draws from GP + noise

GP $\quad \boldsymbol{z}|\boldsymbol{x}_{1:N} \sim \mathrm{N}(\boldsymbol{0}, \boldsymbol{K})$

noise $\quad y_n|z_n \sim \mathrm{N}(z_n, \lambda) \longrightarrow \boldsymbol{y}|\boldsymbol{z} \sim \mathrm{N}(\boldsymbol{y}|\boldsymbol{z}, \lambda\boldsymbol{I})$



PRML Fig. 6.6

Yields marginal $\boldsymbol{y}$

GP indep. noise

$$p(\boldsymbol{y}|\boldsymbol{x}_{1:N}) = \int p(\boldsymbol{y}|\boldsymbol{z})p(\boldsymbol{z}|\boldsymbol{x}_{1:N})\mathrm{d}\boldsymbol{z} = \mathrm{N}(\boldsymbol{y}|\boldsymbol{0}, \underbrace{\boldsymbol{K} + \lambda\boldsymbol{I}}_{\boldsymbol{C}_N})$$

Predict $y_{N+1:N+M}$ from $\boldsymbol{x}_{N+1:N+M}$

1. Joint distribution over $\boldsymbol{y}_{N+M} \equiv y_{1:N+M}$

GP cov. between $\boldsymbol{x}_{1:N}$ and $\boldsymbol{x}_{N+1:N+M}$

$$p(\boldsymbol{y}_{N+M}|\boldsymbol{x}_{1:N+M}) = \mathrm{N}(\boldsymbol{y}_{N+M}|\boldsymbol{0}, \boldsymbol{C}_{N+M}) \quad \text{with } \boldsymbol{C}_{N+M} = \begin{pmatrix} \boldsymbol{C}_N & \boldsymbol{k} \\ \boldsymbol{k}^T & \boldsymbol{c} \end{pmatrix}$$

$\boldsymbol{x}_{N+1:N+M}$ GP cov. + noise

2. Condition on $\boldsymbol{y} \equiv y_{1:N}$

$$p(y_{N+1:N+M}|\boldsymbol{y}, \boldsymbol{x}_{1:N+M}) = \mathrm{N}(y_{N+1:N+M}|\underbrace{\boldsymbol{k}^T\boldsymbol{C}_N^{-1}\boldsymbol{y}}_{= \boldsymbol{k}^T(\boldsymbol{K}+\lambda\boldsymbol{I})^{-1}y}, \boldsymbol{c} - \boldsymbol{k}^T\boldsymbol{C}_N^{-1}\boldsymbol{k})$$

as in Kernel regression

Learning the GP hyperparameters, extensions & complexity

Hyperparameters = parameters $\boldsymbol{\theta}$ of the GP kernel

Standard approach: maximum likelihood

$\boldsymbol{C}_N = \boldsymbol{K} + \lambda \boldsymbol{I}$, function of $\boldsymbol{\theta}$

$$\widehat{\boldsymbol{\theta}} = \text{argmax}_{\boldsymbol{\theta}} \log p(\boldsymbol{y}|\boldsymbol{x}_{1:N}, \boldsymbol{\theta}) = \text{argmax}_{\boldsymbol{\theta}} \left(-\frac{1}{2} \log|\boldsymbol{C}_N| - \frac{1}{2} \boldsymbol{y}^T \boldsymbol{C}_N^{-1} \boldsymbol{y} \right)$$

Challenge: potentially many local maxima

Extensions

GP kernels matching AR(n) process, LDS, OU process, etc. (see Rassumsen & Williams, 2006)

Kernel classification: non-conjugate likelihood, requires (Laplace) approximation

General: use of non-conjugate likelihood (e.g., Poisson spiking) requires approximations

Computational complexity

Use of GP requires $N \times N$ matrix inversion, $O(N^3)$ complexity

Low-dimensional approximation of covariance matrix K (e.g., Fourier decomposition)

Sparse GP: replace data with representative examples

Overview

Kernel methods

Gaussian processes

- What makes them possible: marginalization of multivariate Gaussians
- Gaussian progress regression
- Learning hyperparameters, extensions, etc.

Summary

Kernel methods: perform regression in (dual) space of kernels rather than basis functions

Can handle (effectively) infinitely-dimensional basis functions, at cost of increased complexity

Akin to interpolation: predictions by similarity-based averaging of training data

Gaussian processes: priors over (smooth) functions

Made possible due to nice marginalization properties of multivariate Gaussians

Gaussian process regression like kernel regression + measure of uncertainty

Non-conjugate likelihoods require approximations

Until next week

Read paper and prepare presentation (see notes for Session 7)

Read statistical methods section (no separate PRML reading, only session notes)

Next session

Q & A for previous sesstion

Paper discussions (~1h)

Brief introduction to variational autoencoders (~30min)

Note: one-week break, next session on March 23