Gaussian processes

MIPT

2024

Gaussian process, definition (wiki)

- A random process f_t with continuous time is gaussian if and only if for each finite set of indices $t_1, \ldots, t_k$: $f_{t_1}, \ldots, f_{t_k}$ is a mutlivariative Gaussian variable.
- Each linear combination $f_{t_1}, \ldots, f_{t_k}$ is a univariative Gaussian.

Definition (simplified)

Define a Gaussian process $\mathcal{GP}(\mathbf{m(x)}, \mathbf{k(x, x')}$ to be a distribution on the set of functions that for each $\mathbf{x}, \mathbf{x'}$: $\mathcal{GP}(\mathbf{m(x)}, \mathbf{k(x, x')}$ is a Gaussian distribution.

Example: regression

$$\mathbf{f} \sim \mathcal{N}(\mathbf{0}, \mathbf{K}),$$

where $\mathbf{K}$ is a covariance matrix for $\mathbf{X}$.

$$y \sim f + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma^2).$$

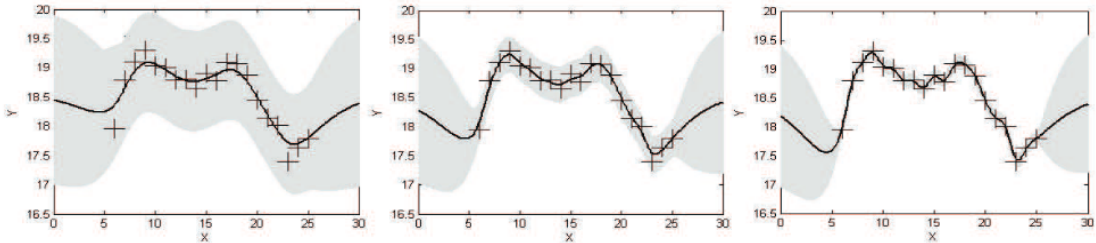
Then $\mathbf{y} \sim \mathcal{N}(\mathbf{0}, \mathbf{K} + \sigma^2\mathbf{I})$ is a prediction for new objects $\hat{\mathbf{y}}$:

$$\hat{\mathbf{y}} \sim \mathcal{N}(\mathbf{K'}^{\mathrm{T}}(\mathbf{K} + \sigma^2\mathbf{I})^{-1}\mathbf{y}, \mathbf{K''} - \mathbf{K'}^{\mathrm{T}}(\mathbf{K} + \sigma^2\mathbf{I})^{-1}\mathbf{K'}.$$

Gaussian process: main features

- Nonparametric
  - defined via covariance function and noise level
  - optimization: via MLE
- Prior is defined for the function, not for the parameters
  - we can set prior on the parameters of Gaussian process, then we get GP with degenerate covariance matrix
- Prediction complexity: $O(N^3)$.

σ^2 and prediction performance



(McDuff, 2010)

Covariance functions

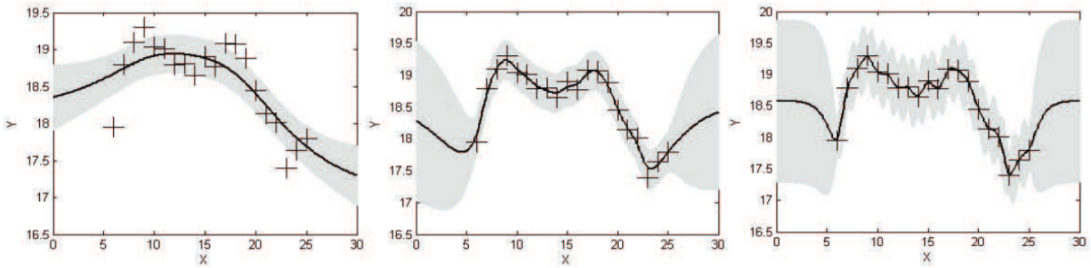
Requirements and properties:

- Symmetry: $\mathbf{K}(\mathbf{x}, \mathbf{x}') = \mathbf{K}(\mathbf{x}, \mathbf{x}')$;
- Positive semidefiniteness: $\mathbf{v}^T \mathbf{K} \mathbf{v} \geq 0$;
- Stationarity: $\mathbf{K}(\mathbf{x}, \mathbf{x}') = \mathbf{K}(\mathbf{x} + \mathbf{a}, \mathbf{x}' + \mathbf{a})$;
- Isotropy: dependence only on $||\mathbf{x} - \mathbf{x}'||$.

Covariance functions: examples

- Exponential: $\mathbf{K} = \sigma_0^2 \exp\left(\frac{-(x-x')^2}{2\lambda}\right)$
- Linear: $\sigma_0 + xx'$
- Brownian: $\min(x, x')$
- Periodic: $\exp\left(\frac{-2\sin^2\left(\frac{x-x'}{2}\right)}{\lambda^2}\right)$
- Defined using NN

λ and prediction performance



(McDuff, 2010)

Matérn covariance

$$\frac{2^{1-\nu}}{\Gamma(\nu)}\left(\frac{\sqrt{2\nu}|x-x'|}{\lambda}\right)^{\nu} K_\nu\left(\frac{\sqrt{2\nu}|x-x'|}{\lambda}\right)$$

- K_ν is a modified Bessel function;
- Stationary and isotropic;
- With $\nu \to \infty$: exponential
- With finite ν: less smooth

Hyperparameter optimization

(Snoek et al., 2012)

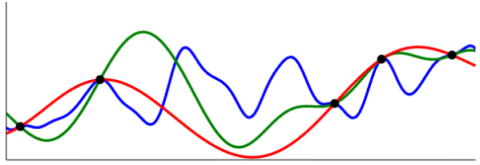
$$\mathbf{f}(\mathbf{x}, \mathbf{w}, \mathbf{h}) = \mathbf{f}(\mathbf{w}(\mathbf{h}), \mathbf{h}|\mathbf{x})$$

The model $\mathbf{f}$ is a function from hyperparameters:

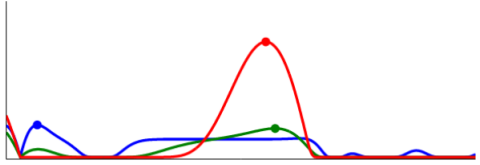
$$\mathbf{f} \sim \mathcal{GP}, \mathbf{y} \sim \mathbf{f} + \varepsilon$$

Using Matérn covariance with $\nu = 2.5$.

λ and hyperparameter optimization

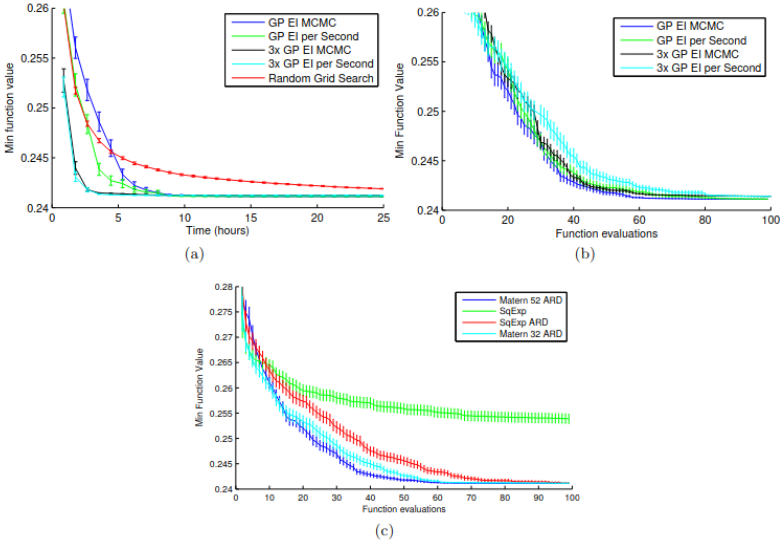


(a) Posterior samples under varying hyperparameters

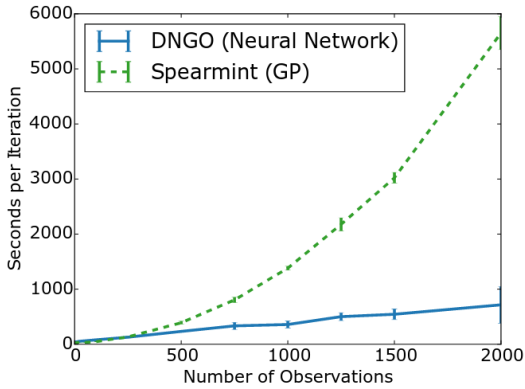


(b) Expected improvement under varying hyperparameters

Hyperparameter optimization



GP: complexity challenge



(Snoek, 2015)

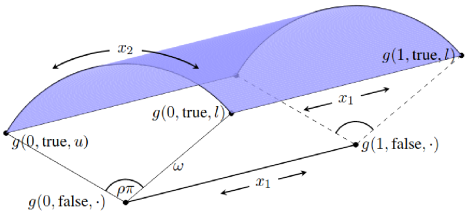
Covariance function for conditional parameters

NN selection problem:

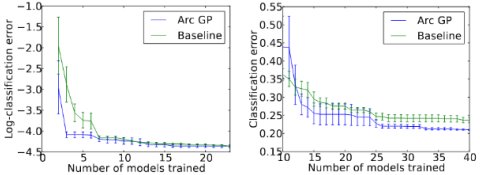
- Hidden layer neuron numbers: ok, we can set as a real-valued hyperparameter
- Layer number: ok, we can set as a real-valued hyperparameter
- **How to compare two architectures with different layer number?**
- **How to compare models with [100, 100] and [100] neurons?**

(Swersky et al., 2014): use a special covariance function!

Covariance function for conditional parameters: idea

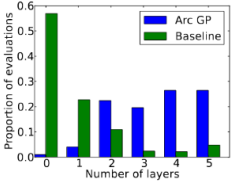


Covariance function for conditional parameters: results



(a) MNIST (b) CIFAR-10

Figure 2: Bayesian optimization results using the arc kernel.



Covariance function for conditional parameters: scheme

- If we are comparing two points for which the same parameters are relevant, the value of any unused parameters shouldn't matter,

$$k\big((x_1, \text{false}, x_2), (x_1', \text{false}, x_2')\big) = k\big((x_1, \text{false}, x_2''), (x_1', \text{false}, x_2''')\big),\ \forall x_2, x_2', x_2'', x_2'''; \quad (1)$$

- The covariance between a point using both parameters and a point using only one should again only depend on their shared parameters,

$$k\big((x_1, \text{false}, x_2), (x_1', \text{true}, x_2')\big) = k\big((x_1, \text{false}, x_2''), (x_1', \text{true}, x_2''')\big),\ \forall x_2, x_2', x_2'', x_2'''. \quad (2)$$

State-space models

An n-th order linear State space model (SSM) has the form:

$$\mathbf{x}(t) = \mathbf{A}\mathbf{z}(t) + \mathbf{B}\mathbf{u}(t),$$

$$\nabla\mathbf{z}(t) = \mathbf{C}\mathbf{z}(t) + \mathbf{D}\mathbf{u}(t),$$

here $\mathbf{z} \in \mathbb{R}^n$ is a state, $\mathbf{u}$ is a control variable.
Why this model is important?

- Can describe the dynamics of n-th order linear ODE (!)
- Allows us to use methods from DE in usual machine learning way.

Kalman filter

$$\mathbf{x}_t = \mathbf{A}\mathbf{z}_t + \varepsilon_x,$$
$$\mathbf{z}(t) = \mathbf{C}\mathbf{z}_{t-1} + \varepsilon_z,$$

where ε is Gaussian noise.
To compute posterior use Bayes rule:

$$p(\mathbf{z}_t|\mathbf{x}_t) \propto p(\mathbf{x}_t|\mathbf{z}_t, \mathbf{x}_{t-1})p(\mathbf{z}_t|\mathbf{x}_{t-1}).$$

Model parameter optimization can be done in different ways (the most simple: using EM algorithm).

Kalman filter: similar models and extensions

$$\mathbf{x}_t = \mathbf{A}\mathbf{z}_t + \varepsilon_x,$$
$$\mathbf{z}(t) = \mathbf{C}\mathbf{z}_{t-1} + \varepsilon_z,$$

- Linear RNN (or just SSM): Kalman filter without noise;
- HMM: discrete state.
- Extended Kalman filter: nonlinear KF using Taylor series.
- Also: we can approximate posterior with NN to make the model non-linear.
- **KF is a special case of Gaussian process**

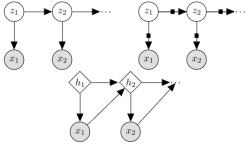


Figure 1: **Generative Models of Sequential Data:** (**Top Left**) Hidden Markov Model (HMM), (**Top Right**) Deep Markov Model (DMM) ■ denotes the neural networks used in DMMs for the emission and transition functions. (**Bottom**) Recurrent Neural Network (RNN), ◇ denotes a deterministic intermediate representation. Code for learning DMMs and reproducing our results may be found at: `github.com/clinicalml/structuredinference`

(Krishnan et al., 2016)

Deep Gaussian processes, Duvenaud et al, 2014[1]

Define a deep GP as a composition of functions, each is independently generated by GP:

$$\mathbf{f} = \mathbf{f}_1 \circ \mathbf{f}_2 \circ \ldots \mathbf{f}_n, \mathbf{f}_i \sim \mathcal{GP}.$$

This composition somewhat similar to MLP: let's say that deep GP can mimic the performance of deep NN.

[1] See the talk of Olga Grebenkova, 2022

Formalizing pathologies

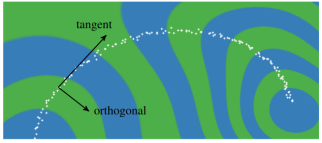


Figure 4: Representing a 1-D data manifold. Colors are a function of the computed representation of the input space. The representation (blue & green) changes little in directions orthogonal to the manifold (white), making it robust to noise in those directions. The representation also varies in directions tangent to the data manifold, preserving information for later layers.

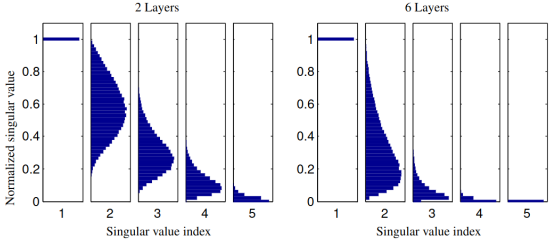
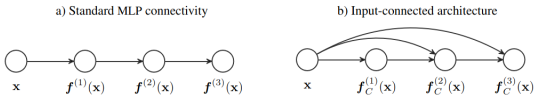


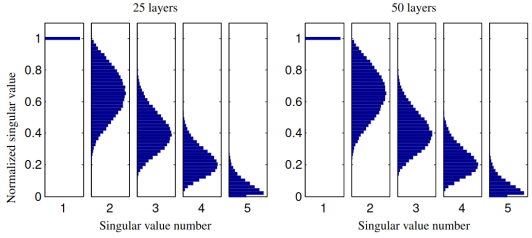
Figure 5: The distribution of normalized singular values of the Jacobian of a function drawn from a 5-dimensional deep GP prior 2 layers deep (*Left*) and 6 layers deep (*Right*). As nets get deeper, the largest singular value tends to become much larger than the others. This implies that with high probability, these functions vary little in all directions but one, making them unsuitable for computing representations of manifolds of more than one dimension.

How to deal with pathologies

The authors propose skip-connections, since they imrpove the Jacobian singular values for deep GP.

Note: ResNet, a model popularizing skip-connections, was proposed in 2015.





Reference

- Bishop C. M. Pattern recognition //Machine learning. – 2006. – T. 128. – No. 9.
- Daniel McDuff, Gaussian Processes, https://courses.media.mit.edu/2010fall/mas622j/ProblemSets/slidesGP.pdf
- Chris Williams, Gaussian Processes for Machine Learning, https://www.newton.ac.uk/files/seminar/20070809140015001-150844.pdf
- Ed Snelson, utorial: Gaussian process models for machine learning, https://mlg.eng.cam.ac.uk/tutorials/06/es.pdf
- Snoek J., Larochelle H., Adams R. P. Practical bayesian optimization of machine learning algorithms //Advances in neural information processing systems. – 2012. – T. 25.
- Swersky K. et al. Raiders of the lost architecture: Kernels for Bayesian optimization in conditional parameter spaces //arXiv preprint arXiv:1409.4011. – 2014.
- Snoek J. et al. Scalable bayesian optimization using deep neural networks //International conference on machine learning. – PMLR, 2015. – C. 2171-2180.
- https://courses.engr.illinois.edu/ece486/fa2021/documentation/lectures/slides/Lecture5C_StateSpaceModels.pdf
- Krishnan, Rahul, Uri Shalit, and David Sontag. "Structured inference networks for nonlinear state space models."Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 31. No. 1. 2017.
- Duvenaud, David, et al. "Avoiding pathologies in very deep networks."Artificial Intelligence and Statistics. PMLR, 2014.