

Scalable Gaussian Processes for Online Learning

A Feasibility Study - MATH 7270 Project

Maxwell Jenquin

May 2nd, 2019

Cornell University

A Brief Outline

Introduction:

- Gaussian processes
- Online learning and goals of the project
- Related work

Methods and Experiments:

- KISS-GP and fast Toeplitz methods
- Experiments and results
- Conclusions

Introduction

Gaussian Process (GP) Models

Gaussian Process:

A stochastic process such that any finite subset of its random variables has a joint multivariate normal distribution:

$$\mathbf{f} \sim \mathcal{GP} \Rightarrow \{f_{i_1}, \dots, f_{i_n}\} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}), \boldsymbol{\mu} \in \mathbb{R}^n, \boldsymbol{\Sigma} \in \mathbb{R}^{n \times n}$$

Gaussian Process (GP) Models

Gaussian Process:

A stochastic process such that any finite subset of its random variables has a joint multivariate normal distribution:

$$\mathbf{f} \sim \mathcal{GP} \Rightarrow \{f_{i_1}, \dots, f_{i_n}\} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}), \boldsymbol{\mu} \in \mathbb{R}^n, \boldsymbol{\Sigma} \in \mathbb{R}^{n \times n}$$

- Fully constrained by a mean function $\mu(\cdot)$ and covariance function $k(\cdot, \cdot)$ or “kernel”

Gaussian Process (GP) Models

Gaussian Process:

A stochastic process such that any finite subset of its random variables has a joint multivariate normal distribution:

$$\mathbf{f} \sim \mathcal{GP} \Rightarrow \{f_{i_1}, \dots, f_{i_n}\} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}), \boldsymbol{\mu} \in \mathbb{R}^n, \boldsymbol{\Sigma} \in \mathbb{R}^{n \times n}$$

- Fully constrained by a mean function $\mu(\cdot)$ and covariance function $k(\cdot, \cdot)$ or “kernel”
- Flexible Bayesian model for scalar functions

Gaussian Process (GP) Models

Gaussian Process:

A stochastic process such that any finite subset of its random variables has a joint multivariate normal distribution:

$$\mathbf{f} \sim \mathcal{GP} \Rightarrow \{f_{i_1}, \dots, f_{i_n}\} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}), \boldsymbol{\mu} \in \mathbb{R}^n, \boldsymbol{\Sigma} \in \mathbb{R}^{n \times n}$$

- Fully constrained by a mean function $\mu(\cdot)$ and covariance function $k(\cdot, \cdot)$ or “kernel”
- Flexible Bayesian model for scalar functions
- Analytic posterior distribution after training

Gaussian Process (GP) Models

Gaussian Process:

A stochastic process such that any finite subset of its random variables has a joint multivariate normal distribution:

$$\mathbf{f} \sim \mathcal{GP} \Rightarrow \{f_{i_1}, \dots, f_{i_n}\} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}), \boldsymbol{\mu} \in \mathbb{R}^n, \boldsymbol{\Sigma} \in \mathbb{R}^{n \times n}$$

- Fully constrained by a mean function $\mu(\cdot)$ and covariance function $k(\cdot, \cdot)$ or “kernel”
- Flexible Bayesian model for scalar functions
- Analytic posterior distribution after training
- Kernels intuitively encode inductive biases

Gaussian Process (GP) Models

Model:

$f \sim \mathcal{GP}(\mu(x), k(x, x'))$ represents a function with $\mathbb{E}[f(x)] = \mu(x)$ and $\text{cov}[f(x), f(x')] = k(x, x')$.

Gaussian Process (GP) Models

Model:

$f \sim \mathcal{GP}(\mu(x), k(x, x'))$ represents a function with $\mathbb{E}[f(x)] = \mu(x)$ and $\text{cov}[f(x), f(x')] = k(x, x')$.

Training:

Optimize the marginal log-likelihood \mathcal{L} with respect to noise estimate σ , and parameters of μ , k (collectively called “hyperparameters”, θ):

$$\mathcal{L}(\theta) = p(\mathbf{y}|\mathbf{x}, \theta) = -\frac{1}{2} \log |K + \sigma^2 I| - \frac{1}{2} \mathbf{y}^T (K + \sigma^2 I)^{-1} \mathbf{y} - \frac{n}{2} \log 2\pi.$$

Gaussian Process (GP) Models

Model:

$f \sim \mathcal{GP}(\mu(x), k(x, x'))$ represents a function with $\mathbb{E}[f(x)] = \mu(x)$ and $\text{cov}[f(x), f(x')] = k(x, x')$.

Training:

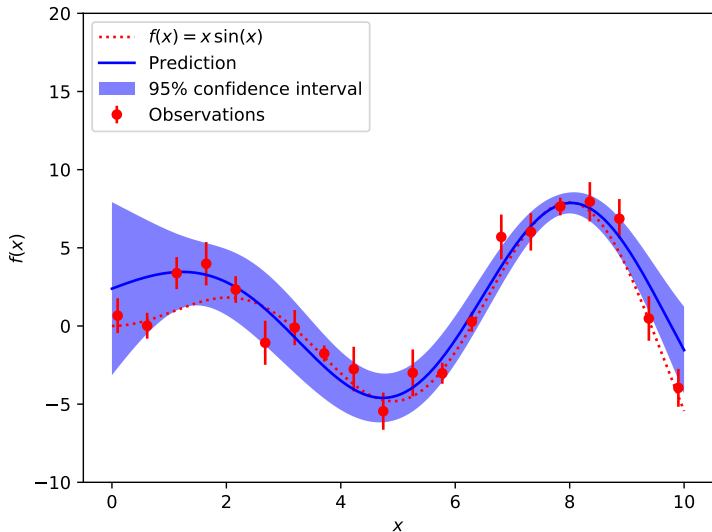
Optimize the marginal log-likelihood \mathcal{L} with respect to noise estimate σ , and parameters of μ , k (collectively called “hyperparameters”, θ):

$$\mathcal{L}(\theta) = p(\mathbf{y}|\mathbf{x}, \theta) = -\frac{1}{2} \log |K + \sigma^2 I| - \frac{1}{2} \mathbf{y}^T (K + \sigma^2 I)^{-1} \mathbf{y} - \frac{n}{2} \log 2\pi.$$

Prediction:

Prediction: For test inputs \mathbf{x}^* , $p(f(\mathbf{x}^*)|\mathbf{x}^*, \mathbf{y}, \mathbf{x})$ is an analytically available multivariate Gaussian distribution.

Gaussian Process (GP) Models



Real-time modeling of incoming data demands efficient, scalable models, and preferably models with constant-time updates (difficult for GPs, see Csató and Opper).

Existing online GP frameworks make serious assumptions about covariance structure. Modern scalable implementations may allow us to relax those assumptions and use a time-windowed approach:

Real-time modeling of incoming data demands efficient, scalable models, and preferably models with constant-time updates (difficult for GPs, see Csató and Opper).

Existing online GP frameworks make serious assumptions about covariance structure. Modern scalable implementations may allow us to relax those assumptions and use a time-windowed approach:

- Fit a GP model every n points, with some overlap between model domains

Real-time modeling of incoming data demands efficient, scalable models, and preferably models with constant-time updates (difficult for GPs, see Csató and Opper).

Existing online GP frameworks make serious assumptions about covariance structure. Modern scalable implementations may allow us to relax those assumptions and use a time-windowed approach:

- Fit a GP model every n points, with some overlap between model domains
- Fit a change surface between those two models on the overlap (linear combinations of GPs are GPs)

Two requirements for a time-windowed online GP model:

1. Sufficiently large window to capture dynamics
2. Ability to keep up with data (fast enough training process)

Two requirements for a time-windowed online GP model:

1. Sufficiently large window to capture dynamics
2. Ability to keep up with data (fast enough training process)

Goals of the Project:

- Choose optimization methodology for scalable KISS-GP implementation
- Determine maximum possible data arrival rate for this methodology

Related Work

- Csató and Opper: “Sparse” GPs with updates (2002)
- Leithead et al.: Quasi-Newton updates reduce $\mathcal{O}(n^3)$ tasks (2005)
- Nguyen et al.: “Local” GP models for adaptive control (2009)
- Ranganathan et al.: Sparse covariance matrix Cholesky factor (2011)
- Ambikasaran, Greengard et al.: Hierarchical matrix approximation (2014)
- Wilson, Nickisch: Structured Kernel Interpolation and KISS-GP (2015)

Methods and Experiments

For input data \mathbf{x}, \mathbf{y} , let U be a grid of size m covering the data domain. Then for some sparse weight matrix (e.g. local cubic interpolation) W , make the approximation

$$K_{\mathbf{x}, U} \approx WK_{U, U} \Rightarrow K_{\mathbf{x}, \mathbf{x}} \approx WK_{U, U}W^T$$

For input data \mathbf{x}, \mathbf{y} , let U be a grid of size m covering the data domain. Then for some sparse weight matrix (e.g. local cubic interpolation) W , make the approximation

$$K_{\mathbf{x}, U} \approx WK_{U, U} \Rightarrow K_{\mathbf{x}, \mathbf{x}} \approx WK_{U, U}W^T$$

Notice that if $k(\cdot, \cdot)$ is stationary, then $K_{U, U}$ has Toeplitz structure (in 1 dimension) or Kronecker structure (in more than 1 dimension).

For input data \mathbf{x}, \mathbf{y} , let U be a grid of size m covering the data domain. Then for some sparse weight matrix (e.g. local cubic interpolation) W , make the approximation

$$K_{\mathbf{x}, U} \approx WK_{U, U} \Rightarrow K_{\mathbf{x}, \mathbf{x}} \approx WK_{U, U} W^T$$

Notice that if $k(\cdot, \cdot)$ is stationary, then $K_{U, U}$ has Toeplitz structure (in 1 dimension) or Kronecker structure (in more than 1 dimension).

Toeplitz structure allows for fast matrix-vector products and linear solves via DFT: KISS-GP costs $\mathcal{O}(mn + m \log m)$ to train and can provide predictions in constant time (after precomputation) using Lanczos variance estimates

Fast Toeplitz Methods

For a Toeplitz matrix K of size $m \times m$, there exists a unique circulant matrix C of size $p \times p$, where $p = 2(m + 1)$ with the structure

$$C = \begin{bmatrix} K & S \\ S^T & A \end{bmatrix} \Rightarrow C \begin{bmatrix} \mathbf{w} \\ \mathbf{0} \end{bmatrix} = \begin{bmatrix} K\mathbf{w} \\ \mathbf{0} \end{bmatrix}$$

Fast Toeplitz Methods

For a Toeplitz matrix K of size $m \times m$, there exists a unique circulant matrix C of size $p \times p$, where $p = 2(m + 1)$ with the structure

$$C = \begin{bmatrix} K & S \\ S^T & A \end{bmatrix} \Rightarrow C \begin{bmatrix} \mathbf{w} \\ \mathbf{0} \end{bmatrix} = \begin{bmatrix} K\mathbf{w} \\ \mathbf{0} \end{bmatrix}$$

As discussed in class, circulant matrix-vector products can be computed in $\mathcal{O}(p \log p)$ operations with $\mathcal{O}(p)$ storage using DFT.

Fast Toeplitz Methods

For a Toeplitz matrix K of size $m \times m$, there exists a unique circulant matrix C of size $p \times p$, where $p = 2(m + 1)$ with the structure

$$C = \begin{bmatrix} K & S \\ S^T & A \end{bmatrix} \Rightarrow C \begin{bmatrix} \mathbf{w} \\ \mathbf{0} \end{bmatrix} = \begin{bmatrix} K\mathbf{w} \\ \mathbf{0} \end{bmatrix}$$

As discussed in class, circulant matrix-vector products can be computed in $\mathcal{O}(p \log p)$ operations with $\mathcal{O}(p)$ storage using DFT.

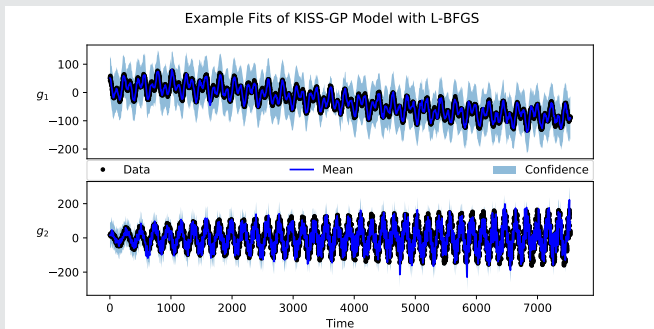
Linear solves may be computed using preconditioned conjugate gradients, which converge quickly and cost only matrix-vector products (making them the same complexity asymptotically). The log-determinant can similarly be accelerated.

Experiment Design

Data Generating Functions:

$$g_1(t) = 50 \cos(t/25) \cos(t/64) + \sqrt{t} \sin\left(\frac{\sqrt{t}}{2 \log t}\right), \quad t > 1$$

$$g_2(t) = 10t^{\frac{1}{4}} \cos\left(\frac{t^{1.1}}{75}\right) + 2t^{0.4} \sin(t/4) \quad t > 0$$



Underlying Model:

$\mu(x) = 0$ (standard practice to subtract the mean),

$k(x, x') = k_{\text{rbf}}(x, x') \cdot k_{\text{per}}(x, x')$ (smooth kernel times periodic kernel)

Experiment Design

Underlying Model:

$\mu(x) = 0$ (standard practice to subtract the mean),

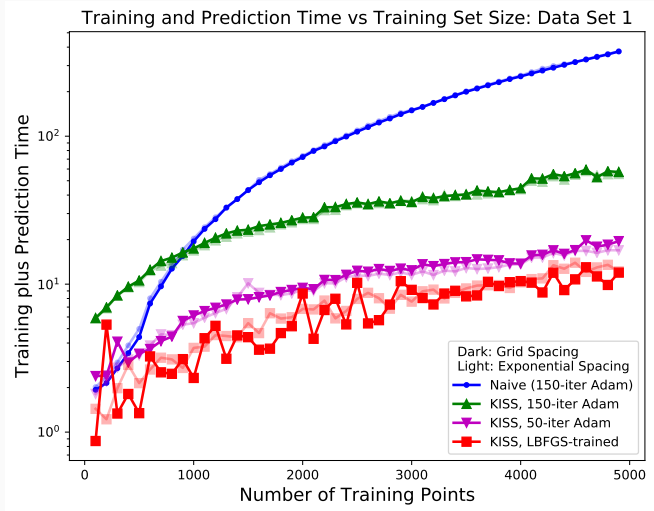
$k(x, x') = k_{\text{rbf}}(x, x') \cdot k_{\text{per}}(x, x')$ (smooth kernel times periodic kernel)

Data Sampling:

Phase 1: Zero noise. Grid-spaced data at intervals of 1, five trials of exponentially spaced data with mean interval 1, up to 5,000 data points

Phase 2: Gaussian noise with variance 3. 30 trials of exponentially spaced data with mean interval 1, up to 30,000 data points

Results: Phase 1



Post-2,000 data point slopes are approximately 10^{-4} . (naive: 5.5, KISS-150: 2.3, KISS-50: 2.0, KISS-LBFGS: 2.6).

Results: Phase 2











n (thousands)	Data from $g_1(t)$			Data from $g_2(t)$		
	time (s)	mean error	input rate	time (s)	mean error	input rate
1	2.398845	2.529934	396	2.095944	1.098058	453
5	5.140201	1.147658	924	6.75071	1.213397	704
10	11.08531	0.797631	857	13.65495	1.819256	696
15	15.59625	0.611085	914	19.11884	1.147181	745
20	28.34102	0.718122	670	35.42328	1.483744	536
25	34.61692	0.601228	686	44.02118	2.296331	540
29	40.5879	0.528249	679	46.90926	0.716015	587

Best-case measurement rate: approximately 900 samples per second without GPU acceleration!

Conclusions

- Early-exit optimizers such as L-BFGS scale well with data in the GP context, but are subject to higher training-time variance than standard choices
- Gaussian process models with generic (stationary) covariance structure can be efficiently implemented using KISS-GP in the 1-d case
- KISS-GP without parallelization has suitable speed for many large-windowed online context, up to 900 samples per second in numerical experiments
- GPU acceleration has the potential to make this a viable specialized online model, when implemented with care

References

-  S. AMBIKASARAN, D. FOREMAN-MACKEY, L. GREENGARD, D. W. HOGG, AND M. O'NEIL, *Fast direct methods for gaussian processes*, arXiv preprint arXiv:1403.6015, (2014).
-  J. R. GARDNER, G. PLEISS, D. BINDEL, K. Q. WEINBERGER, AND A. G. WILSON, *Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration*, in Advances in Neural Information Processing Systems, 2018.
-  D. P. KINGMA AND J. BA, *Adam: A method for stochastic optimization*, arXiv preprint arXiv:1412.6980, (2014).
-  W. LEITHEAD, Y. ZHANG, AND D. LEITH, *Efficient gaussian process based on bfgs updating and logdet approximation*, IFAC Proceedings Volumes, 38 (2005), pp. 1305–1310.
-  W. E. LEITHEAD AND Y. ZHANG, *$O(n^2)$ -operation approximation of covariance matrix inverse in gaussian process regression based on quasi-newton bfgs method*, Communications in StatisticsSimulation and Computation®, 36 (2007), pp. 367–380.
-  G. PLEISS, J. R. GARDNER, K. Q. WEINBERGER, AND A. G. WILSON, *Constant-time predictive distributions for gaussian processes*, arXiv preprint arXiv:1803.06058, (2018).
-  A. RANGANATHAN, M.-H. YANG, AND J. HO, *Online sparse gaussian process regression and its applications*, IEEE Transactions on Image Processing, 20 (2011), pp. 391–404.
-  C. K. WILLIAMS AND C. E. RASMUSSEN, *Gaussian processes for machine learning*, vol. 2, MIT Press Cambridge, MA, 2006.
-  A. WILSON AND H. NICKISCH, *Kernel interpolation for scalable structured gaussian processes (kiss-gp)*, in International Conference on Machine Learning, 2015, pp. 1775–1784.
-  A. G. WILSON, *Covariance kernels for fast automatic pattern discovery and extrapolation with Gaussian processes*, PhD thesis, University of Cambridge, 2014.