GP World — Tutorial on Gaussian Processes and their Use in Reinforcement Learning

Marc P. Deisenroth

Max Planck Institute for Biological Cybernetics
Department of Empirical Inference for Machine Learning and Perception
Tübingen, Germany

Workshop on Analytical Challenges in Reinforcement Learning
Tübingen, Germany

August 03, 2007

MAX-PLANCK-GESELLSCHAFT

MPI FOR BIOLOGICAL CYBERNETICS

Outline

Gaussian distribution

$$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) := (2\pi)^{D/2}|\boldsymbol{\Sigma}|^{-0.5} \exp(-0.5(\mathbf{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}))$$

- fully specified by mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$
- nice property: conditionals and marginals of a joint Gaussian are again Gaussian

Definition

Definition (Åström, 2006)

A stochastic process is a function of two arguments $\{x(t,\omega), t \in T, \omega \in \Omega\}$, where T is a (not necessarily finite) time interval, and Ω is a sample space. For fixed $t \in T$, $x(t,\cdot)$ is thus a random variable and for fixed ω, $x(\cdot,\omega)$ is a function of time, a realization of the process[a].

[a]also called path or trajectory

- in Euclidean spaces probability distributions define a probability measure on subsets of Ω

Definition

Definition (Åström, 2006)

A stochastic process is a function of two arguments $\{x(t,\omega), t \in T, \omega \in \Omega\}$, where T is a (not necessarily finite) time interval, and Ω is a sample space. For fixed $t \in T$, $x(t,\cdot)$ is thus a random variable and for fixed ω, $x(\cdot,\omega)$ is a function of time, a realization of the process[a].

[a] also called path or trajectory

- in Euclidean spaces probability distributions define a probability measure on subsets of Ω
- consider a function as an infinitely long vector

Definition (Åström, 2006; Rasmussen and Williams, 2006)

A normal or Gaussian process is a collection of random variables, any finite number of which have consistent joint Gaussian distributions.

- GP as distribution over functions

Generalization of Gaussian distribution

recall:
Gaussian distribution is fully specified by mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$

here:
Gaussian process is fully specified by mean function $\mu(\mathbf{x})$ and covariance function $k(\mathbf{x}, \mathbf{x}')$

Prediction

joint Gaussian distribution of training set $\mathbf{X}$ and test set $\mathbf{X}_*$

$$p(\mathbf{f}, \mathbf{f}_*) = \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} k(\mathbf{X}, \mathbf{X}) & k(\mathbf{X}, \mathbf{X}_*) \\ k(\mathbf{X}_*, \mathbf{X}) & k(\mathbf{X}_*, \mathbf{X}_*) \end{bmatrix}\right)$$

then the conditional is given by

$$\begin{aligned} p(\mathbf{f}_* | \mathbf{f}) &= \mathcal{N}(\mathbf{m}, \mathbf{\Sigma}) \\ \mathbf{m} &= k(\mathbf{X}_*, \mathbf{X}) k(\mathbf{X}, \mathbf{X})^{-1} \mathbf{f} \\ \mathbf{\Sigma} &= k(\mathbf{X}_*, \mathbf{X}_*) - k(\mathbf{X}_*, \mathbf{X}) k(\mathbf{X}, \mathbf{X})^{-1} k(\mathbf{X}, \mathbf{X}_*) \end{aligned}$$

Prediction

joint Gaussian distribution of training set $\mathbf{X}$ and test set $\mathbf{X}_*$

$$p(\mathbf{f}, \mathbf{f}_*) = \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} k(\mathbf{X}, \mathbf{X}) & k(\mathbf{X}, \mathbf{X}_*) \\ k(\mathbf{X}_*, \mathbf{X}) & k(\mathbf{X}_*, \mathbf{X}_*) \end{bmatrix}\right)$$

then the conditional is given by

$$\begin{aligned} p(\mathbf{f}_* | \mathbf{f}) &= \mathcal{N}(\mathbf{m}, \mathbf{\Sigma}) \\ \mathbf{m} &= k(\mathbf{X}_*, \mathbf{X}) k(\mathbf{X}, \mathbf{X})^{-1} \mathbf{f} \\ \mathbf{\Sigma} &= k(\mathbf{X}_*, \mathbf{X}_*) - k(\mathbf{X}_*, \mathbf{X}) k(\mathbf{X}, \mathbf{X})^{-1} k(\mathbf{X}, \mathbf{X}_*) \end{aligned}$$

for only 1 test input $\mathbf{x}_*$

$$m(\mathbf{x}_*) = \sum_i \alpha_i k(\mathbf{x}_i, \mathbf{x}_*)$$

➡ predicted mean linear combination of cross-covariances

Extension to multiple test points

previous results extend to arbitrarily many test points
➡ construct posterior Gaussian process from a GP prior conditioned on observed data

Bayesian model selection

parametric Bayesian inference:

1. observe data
2. define a prior
3. determine posterior conditioned on data and prior

- data: $\mathbf{X}, \mathbf{y}$
- model: $y = f_{\mathbf{w}}(\mathbf{x}) + \varepsilon$

levels:

1. parameters $\mathbf{w}$ of f
2. hyperparameters $\boldsymbol{\theta}$ controlling the distribution of $\mathbf{w}$
3. set of possible models $\mathcal{H} = \{H_i : i \in \mathcal{I}\}$

Bayesian model selection

parametric Bayesian inference:

1. observe data
2. define a prior
3. determine posterior conditioned on data and prior

- data: $\mathbf{X}, \mathbf{y}$
- model: $y = f_{\mathbf{w}}(\mathbf{x}) + \varepsilon$

levels:

1. parameters $\mathbf{w}$ of f
2. hyperparameters $\boldsymbol{\theta}$ controlling the distribution of $\mathbf{w}$
3. set of possible models $\mathcal{H} = \{H_i : i \in \mathcal{I}\}$

nonparametric Bayesian inference:
parameters $\mathbf{w}$ are given by the function f itself
➡ embedding into GP framework

Marginal likelihood

end up with optimizing the (log-) marginal likelihood to get the best hyperparameters

$$\log p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta}, H_i) = \underbrace{-\frac{1}{2}\mathbf{y}^T \mathbf{K}_{\boldsymbol{\theta}}^{-1} \mathbf{y}}_{\text{data fit term}} \; \underbrace{-\frac{1}{2}\log|\mathbf{K}_{\boldsymbol{\theta}}|}_{\text{complexity penalty}} \; -\frac{n}{2}\log(2\pi)$$

➡ trade-off between data-fit and model complexity (Occam's razor)
➡ optimizing marginal likelihood automatically finds the least complex model that reasonably explains observed data

Wrap-up I

- GP is distribution in function space
- nonparametric Bayesian inference
- favors easy models
- restriction to finite sets
  ➡ properties of Gaussian distribution

Who am I?
What am I able to do?

recall:
GP is able to find a model of low complexity that explains data

Who am I?
What am I able to do?

recall:
GP is able to find a model of low complexity that explains data

➡ Why not identification of system dynamics to build a model based on finite training set?

simulations showed that this works well

$$f \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$$

➡ for each test input obtain predicted mean and confidence interval

What will happen next?
Predictions with GPs

1. 1-step prediction (near future)
2. k-step prediction (later)

idea: propagate not only mean, but also uncertainty information
➡ prediction for uncertain inputs needed (Girard et al., 2003)
problem:

$$p(f(\mathbf{x}_*)|\boldsymbol{\mu}_{x_*}, \boldsymbol{\Sigma}_{x_*}) = \int p(f(\mathbf{x}_*)|\mathbf{x}_*, \mathbf{X}, \mathbf{y})p(\mathbf{x}_*)\mathrm{d}\mathbf{x}_*$$

Gaussian approximation (moment matching).
Example:

$$m(\mathbf{x}_*) \approx \underset{\mathbf{x}_*}{\mathrm{E}}\left[\underset{\mathbf{f}(x_*)}{\mathrm{E}}[f(\mathbf{x}_*)|\mathbf{x}_*]\right] = \underset{\mathbf{x}_*}{\mathrm{E}}[\mu(\mathbf{x}_*)]$$

k-step ahead prediction

different methods

- learn GP for each time step doing a direct k-step prediction of the dynamics
- learn 1 GP for 1-step prediction and apply previous method recursively

How about others?

Embedding into RL framework

so far:

- model building via GPs
- approximate k-step ahead prediction

How about others?
Embedding into RL framework

so far:

- model building via GPs
- approximate k-step ahead prediction

what can we use it for?

- policy evaluation: solve

$$\mathop{\mathrm{E}}_{\pi}\left[\sum_{k}\gamma^{k}r(\mathbf{x}_k,\mathbf{u}_k)\right]$$

given by

$$\int\gamma^{N}r(\mathbf{x}_N)p(\mathbf{x}_N)\mathrm{d}\mathbf{x}_N+\sum_{k=0}^{N-1}\gamma^{k}\int\int p(\mathbf{x}_k,\mathbf{u}_k)r(\mathbf{x}_k,\mathbf{u}_k)\mathrm{d}\mathbf{x}_k\mathrm{d}\mathbf{u}_k\,.$$

How about others?
Embedding into RL framework

so far:

- model building via GPs
- approximate k-step ahead prediction

what can we use it for?

- policy evaluation: solve

$$\mathop{\mathrm{E}}_{\pi}\left[\sum_k \gamma^k r(\mathbf{x}_k, \mathbf{u}_k)\right]$$

  given by

$$\int \gamma^N r(\mathbf{x}_N) p(\mathbf{x}_N) \mathrm{d}\mathbf{x}_N + \sum_{k=0}^{N-1} \gamma^k \int\int p(\mathbf{x}_k, \mathbf{u}_k) r(\mathbf{x}_k, \mathbf{u}_k) \mathrm{d}\mathbf{x}_k \mathrm{d}\mathbf{u}_k \,.$$

- policy gradient
  derivative w.r.t. (hyper-)parameters is possible

How about others?
Embedding into RL framework

so far:

- model building via GPs
- approximate k-step ahead prediction

what can we use it for?

- policy evaluation: solve

$$\mathop{\mathrm{E}}_{\pi}\left[\sum_k \gamma^k r(\mathbf{x}_k, \mathbf{u}_k)\right]$$

  given by

$$\int \gamma^N r(\mathbf{x}_N) p(\mathbf{x}_N) \mathrm{d}\mathbf{x}_N + \sum_{k=0}^{N-1} \gamma^k \int\int p(\mathbf{x}_k, \mathbf{u}_k) r(\mathbf{x}_k, \mathbf{u}_k) \mathrm{d}\mathbf{x}_k \mathrm{d}\mathbf{u}_k \,.$$

- policy gradient
  derivative w.r.t. (hyper-)parameters is possible
- fitted value iteration (modeling V and Q function by GPs)

Almost forgotten

don't forget: GPs usually map into $\mathbb{R}$
➡ construct multi-dimensional output:

Almost forgotten

don't forget: GPs usually map into $\mathbb{R}$

➡ construct multi-dimensional output:

1. place GP on each output dimension (independent of other outputs) using the same inputs

$$\mathbf{f} \sim \begin{bmatrix} \mathcal{GP}_1(m_1, k_1) \\ \vdots \\ \mathcal{GP}_n(m_n, k_n) \end{bmatrix}$$

2. construct joint probability distribution for output

$$\mathcal{N}(\mathbf{m}, \mathbf{\Sigma})$$

$$\Sigma = \begin{bmatrix} \sigma_1^2 & \mathrm{Cov}(y_1, y_2) & \dots & \mathrm{Cov}(y_1, y_n) \\ \vdots & \ddots & \dots & \vdots \\ \mathrm{Cov}(y_n, y_1) & & \dots & \sigma_n^2 \end{bmatrix}$$

Current research with GPs in RL and control

some examples

- model building and optimal control (Kocijan et al., 2003; 2004; Grancharova et al., 2007)
- policy iteration with GP-models of dynamics and value function (Rasmussen and Kuss, 2004)
- value function approximation without model (Engel et al., 2003; 2005)
- Bayesian policy gradient with known model (Ghavamzadeh and Engel, 2007)

Wrap-up II

- GPs can be used in RL framework
- several difficulties occur
- GPs in standard RL methods are being investigated
- so far no approach to solve everything together only with GPs

References I

Karl J. Åström.
Introduction to Stochastic Control Theory.
Dover Publications, Inc., 31 East 2nd Street, Mineola, NY 11501, U.S.A., January 2006.

Yaakov Engel, Shie Mannor, and Ron Meir.
Bayes Meets Bellman: The Gaussian Process Approach to Temporal Difference Learning.
In *Proceedings of the 20th International Conference on Machine Learning (ICML-2003)*, volume 20, pages 154–161, Washington, DC, U.S.A., August 2003.

Yaakov Engel, Shie Mannor, and Ron Meir.
Reinforcement Learning with Gaussian Processes.
In *Proceedings of the 22nd International Conference on Machine Learning (ICML-2005)*, volume 22 of *201–208*, Bonn, Germany, August 2005.

Mohammad Ghavamzadeh and Yaakov Engel.
Bayesian Policy Gradient Algorithms.
In Bernhard Schölkopf, John Platt, and Thomas Hofman, editors, *Advances in Neural Information Processing Systems 19*. MIT Press, Cambridge, MA, U.S.A., 2007.

Alexandra Grancharova, Juš Kocijan, and Tor A. Johansen.
Explicit Stochstic Nonlinear Predictive Control Based on Gaussian Process Models.
In *Proceedings of the 9th European Control Conference 2007 (ECC 2007)*, pages 2340–2347, Kos, Greece, July 2007.

Agathe Girard, Carl E. Rasmussen, Joaquin Quiñonero Candela, and Roderick Murray-Smith.
Gaussian Process Priors With Uncertain Inputs — Application to Multiple-Step Ahead Time Series Forecasting.
In Suzanna Becker, Sebastian Thrun, and Klaus Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 529–536. The MIT Press, Cambridge, MA, U.S.A., 2003.

Juš Kocijan, Roderick Murray-Smith, Carl E. Rasmussen, and Agathe Girard.
Gaussian Process Model Based Predictive Control.
In *Proceedings of the 2004 American Control Conference (ACC 2004)*, pages 2214–2219, Boston, MA, U.S.A., June–July 2004.

References II

Juš Kocijan, Roderick Murray-Smith, Carl E. Rasmussen, and Bojan Likar.
Predictive Control with Gaussian Process Models.
In Baldomir Zajc and Marko Tkalčič, editors, *Proceedings of IEEE Region 8 Eurocon 2003: Computer as a Tool*, pages 352–356, Piscataway, NJ, U.S.A., September 2003.

Carl E. Rasmussen.
Advances in Gaussian Processes.
NIPS Tutorial, December 2006.
http://media.nips.cc/Conferences/2006/Tutorials/Slides/Rasmussen.pdf.

Carl E. Rasmussen and Malte Kuss.
Gaussian Processes in Reinforcement Learning.
In Sebastian Thrun, Lawrence K. Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems 16*, pages 751–759. The MIT Press, Cambridge, MA, U.S.A., June 2004.

Carl E. Rasmussen and Christopher K. I. Williams.
Gaussian Processes for Machine Learning.
Adaptive Computation and Machine Learning. The MIT Press, Cambridge, MA, U.S.A., 2006.