Advanced Techniques for Mobile Robotics

Gaussian Processes in Robotics

Wolfram Burgard, Cyrill Stachniss, Kai Arras, Maren Bennewitz
Overview

- Regression problem
- Gaussian process models
- Learning GPs
- Applications
- Summary
The Regression Problem

- Given \( n \) observed points

\[ \mathcal{X} = \{(x_1, t_1), \ldots, (x_n, t_n)\}, \quad x_i \in \mathbb{R}^p, \quad t_i \in \mathbb{R} \]

- Assuming the dependency

\[ t_i = f(x_i) + \epsilon_i, \quad \epsilon_i \sim \mathcal{N}(0, \sigma^2), \quad i.i.d \]

- How to predict new points

\[ p(t_{n+1} \mid x_{n+1}, \mathcal{X}) \]
The Regression Problem

- Given $n$ observed points
The Regression Problem

- Solution 1: Parametric models
  - Linear \( f(x_i) = c_0 + c_1 x_i + \epsilon_i \)
  - Quadratic \( f(x_i) = c_0 + c_1 x_i + c_2 x_i^2 + \epsilon_i \)
  - Higher order polynomials
  - ...

- Learning: optimizing the parameters
The Regression Problem

- Solution 1: Parametric models
The Regression Problem

- Solution 1: Parametric models
The Regression Problem

- Solution 2: Non-parametric models
  - Radial Basis functions
    \[ f(x_i) = \sum_j w_j k(\| x_i - c_j \|) \]
    \[ k(\| x - c \|) \propto e^{-\beta\|x-c\|^2} \]
  - Histograms, Splines, Support Vector Machines ...
- Learning: finding the structure of the model and optimize its parameters
The Regression Problem

- Given n observed points
The Regression Problem

- Solution 3: Express \( t_i = f(x_i) + \epsilon_i \) directly in terms of the data points

- Idea: Any finite set of values \( t_i \) sampled from \( (t_1, \ldots, t_n) \sim \mathcal{N}(0, K) \) has a joint Gaussian distribution with a covariance matrix \( K \) given by

\[
k_{ij} = \text{cov}(t_i, t_j) = \text{cov}(f(x_i), f(x_j)) =: c(x_i, x_j)
\]
Gaussian Process Models

- Then, the \( n+1 \) dimensional vector

\[
\begin{pmatrix}
    f(x_1), & \ldots, & f(x_n), & f(x_{n+1})
\end{pmatrix},
\]

which includes the new target to be predicted \( t_{n+1} = f(x_{n+1}) \), comes from an \( n+1 \) dimensional Gaussian

- The predictive distribution for the new target \( p(t_{n+1} \mid x_{n+1}, \mathcal{X}) \) is a 1-dimensional Gaussian
Gaussian Process Model

- Given the $n$ observed points
- **Squared exponential** covariance function

\[
c(x_i, x_j) = \sigma_f^2 \cdot \exp\left(-\frac{(x_i-x_j)^2}{\ell^2}\right) + \delta_{i=j}\sigma_n^2
\]

- with $\sigma_f = \frac{1}{6}$, $\ell = 5$,
- and a noise level $\sigma_n^2$
The Regression Problem

- Given \( n \) observed points
Gaussian Process Models

- GP model
Learning GPs

- The **squared exponential** covariance function:

\[
c(x_i, x_j) = \sigma_f^2 \cdot \exp \left( -\frac{(x_i - x_j)^2}{\ell^2} \right) + \delta_{i=j}\sigma_n^2
\]

- Easy to interpret parameters
Learning GPs

- Example: low noise
Learning GPs

- Example: medium noise
Learning GPs

- Example: high noise
Learning GPs

- Example: small lengthscale
Learning GPs

- Example: large lengthscale
Learning GPs

- Covariance function specifies the **prior**

![prior and posterior plots](image)

**prior**

**posterior**
Gaussian Process Models

- Recall, the $n+1$ dimensional vector
  \( (f(x_1), \ldots, f(x_n), f(x_{n+1})) \),
  comes from an $n+1$ dimensional normal distribution.

- The predictive distribution for the new target $p(t_{n+1} | x_{n+1}, \mathcal{X})$ is a 1-dimensional Gaussian.

- Why?
The Gaussian Distribution

- Recall the 2-dimensional joint Gaussian:

- The **conditionals** and the **marginals** are also Gaussians

Figure taken from Carl E. Rasmussen: NIPS 2006 Tutorial
The Gaussian Distribution

- Simple bivariate example:

\[ p(x, y) = \mathcal{N}(0, \Sigma) \]

\[ \Sigma = \begin{pmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{pmatrix} \]

\[ p(x, y) = \frac{1}{2\pi\sigma_x\sigma_y} \exp \left( -\frac{1}{2} \left( \frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2} \right) \right) \]

\[ p(x \mid y) = \]
The Gaussian Distribution

- Simple bivariate example:

\[ p(x, y) = \mathcal{N}(0, \Sigma) \]

\[ \Sigma = \begin{pmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{pmatrix} \]

\[ p(x, y) = \frac{1}{2\pi\sigma_x\sigma_y} \exp \left( -\frac{1}{2} \left( \frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2} \right) \right) \]

\[ p(x \mid y) = \frac{p(x, y)}{p(y)} \quad \text{joint} \]

\[ \text{conditional} \quad \text{marginal} \]
The Gaussian Distribution

- Marginalization:

\[ p(y) = \int p(x, y) \, dx \]
\[ = \int \frac{1}{2\pi\sigma_x\sigma_y} \exp \left( -\frac{1}{2} \left( \frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2} \right) \right) \, dx \]
\[ = \frac{1}{\sigma_y \sqrt{2\pi}} \exp \left( -\frac{1}{2} \frac{y^2}{\sigma_y^2} \right) \]

\[ \mathcal{N}(0, \sigma_y^2) \]
The Gaussian Distribution

- The conditional:

\[ p(x \mid y) = \frac{p(x, y)}{p(y)} = \frac{1}{2\pi \sigma_x \sigma_y} \exp \left( -\frac{1}{2} \left( \frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2} \right) \right) \cdot \frac{1}{\sigma_y \sqrt{2\pi}} \exp \left( -\frac{1}{2} \frac{y^2}{\sigma_y^2} \right)^{-1} \]

\[ = \frac{1}{\sigma_x \sqrt{2\pi}} \exp \left( -\frac{1}{2} \frac{x^2}{\sigma_x^2} \right) \]

\[ \mathcal{N}(0, \sigma_x^2) \]
The Gaussian Distribution

- Slightly more complicated in the general case:

- The **conditionals** and the **marginals** are also Gaussians

Figure taken from Carl E. Rasmussen: NIPS 2006 Tutorial
The Gaussian Distribution

- Conditioning the joint Gaussian in general

\[ p(x, y) = \mathcal{N}\left(\begin{pmatrix} a \\ b \end{pmatrix}, \begin{pmatrix} A & B \\ B^T & C \end{pmatrix}\right) \]

\[ p(x \mid y) = \mathcal{N}(a + BC^{-1}(y - b), A - BC^{-1}B^T) \]

- In case of zero mean:

\[ p(x, y) = \mathcal{N}\left(0, \begin{pmatrix} A & B \\ B^T & C \end{pmatrix}\right) \]

\[ p(x \mid y) = \mathcal{N}(BC^{-1}y, A - BC^{-1}B^T) \]
Gaussian Process Models

- Recall the GP assumption

\[
t = (t_1, \ldots, t_n)^T \\
t \sim \mathcal{N}(0, K) \\
\begin{pmatrix} t \\ t_{n+1} \end{pmatrix} \sim \mathcal{N}(0, \begin{pmatrix} K & k \\ k^T & \nu \end{pmatrix}) \\
t_{n+1} | t \sim \mathcal{N}(\mu^*, \sigma^*)
\]
Gaussian Process Models

- **Noise-free** mean and variance of the predictive distribution have the form

\[
\begin{align*}
\mu^* &= E(t_{n+1} \mid t_1, \ldots, t_n) = k^T K^{-1} t \\
\sigma^* &= V(t_{n+1} \mid t_1, \ldots, t_n) = v - k^T K^{-1} k
\end{align*}
\]

- with

\[
K = \begin{bmatrix} 
    c(x_1, x_1) & \cdots & c(x_1, x_n) \\
    \vdots & \ddots & \vdots \\
    \cdots & \cdots & c(x_n, x_n) 
\end{bmatrix} \quad k = \begin{bmatrix} 
    c(x_1, x_{n+1}) \\
    \vdots \\
    c(x_n, x_{n+1}) 
\end{bmatrix} \quad v = c(x_{n+1}, x_{n+1})
\]

\[
t = \begin{bmatrix} 
    t_1 \\
    \vdots \\
    t_n 
\end{bmatrix}
\]
Gaussian Process Models

- Mean and variance of the predictive distribution then lead to

\[
\mu^* = k^T (K + I\sigma_n^2)^{-1} t \\
\sigma^* = c(x_{n+1}, x_{n+1}) - k^T (K + I\sigma_n^2)^{-1} k
\]

- with

\[
K = \begin{bmatrix}
c(x_1, x_1) & \ldots & c(x_1, x_n) \\
\vdots & \ddots & \vdots \\
\vdots & \vdots & c(x_n, x_n)
\end{bmatrix} \quad k = \begin{bmatrix} c(x_1, x_{n+1}) \\ \vdots \\ c(x_n, x_{n+1}) \end{bmatrix}
\]
Learning GPs

- Learning a Gaussian process means
  - choosing a covariance function
  - finding its parameters and the noise level

- What is the objective?
Learning GPs

- The hyperparameters

$$\theta = \{ \sigma_f, (l_1, \ldots, l_n), \sigma_n^2 \}$$

can be found by maximizing the likelihood of the training data

$$\theta = \operatorname{argmax}_\theta \log p(t_1, \ldots, t_n \mid x_1, \ldots, x_n, \theta),$$

e.g., using gradient methods
Learning GPs

- **Objective:** high data likelihood

\[
\log p(t \mid x) = -\frac{1}{2} t^T (K + \sigma_n^2)^{-1} t - \frac{1}{2} \log |K + \sigma_n^2 I| - \frac{n}{2} \log 2\pi
\]

- Due to the Gaussian assumption, GPs have Occam’s razor built in
Occam’s Razor

- Use the simplest explanation that is needed to describe the data

- **Data-fit** favors overfitting
- **Complexity penalty** favors simplicity
Advanced Topics / Extensions

- Classification/non-Gaussian noise
- Sparse GPs: Approximations for large data sets
- Heteroscedastic GPs: Modeling non-constant noise
- Nonstationary GPs: Modeling varying smoothness (lengthscales)
- Mixtures of GPs
- Uncertain inputs
  
...
Further Reading

Rasmussen and Williams

Gaussian process web (code, papers, etc): http://www.GaussianProcess.org
Applications in Robotics

- Monocular range sensing
- Terrain modeling
- Learning sensor models
- Learning to control a blimp
- Localization in cellular networks
- Time-series forecasting
- ...
Applications in Robotics

- Monocular range sensing
- Terrain modeling
  - Learning sensor models
  - Learning to control a blimp
  - Localization in cellular networks
  - Time-series forecasting
- ...

Monocular Range Sensing

- Can we learn range from single, monocular camera images?
Training Setup

- Mobile robot + laser range finder
- Omni-directional monocular camera
Training Setup

DFKI Saarbrücken

University of Freiburg
Learning Range from Vision

- Associate (polar) pixel columns with ranges

\[ p_i \in \mathbb{R}^{420} \]

Extract features

\[ x_i = f(p_i) \]

Associate with ranges

\[ r_i = r(x_i) \in \mathbb{R} \]
Pre-processing

- Warp images into a panoramic view
- 120 pixels per column
- Transform to HSV -> 420 dimensions
Visual Features

- Two types of features
  1. No human engineering: Principle components analysis (PCA) on raw input
  2. Use of domain specific knowledge: Edge features that shall correspond to floor boundaries
Experiments

**Typical 180° scan**

- Ground Truth Distances (Laser)
- Predicted means (FeatureGP)
Online Prediction

Visual features

GP range prediction

Results: Range predictions from visual input
Mapping Results

Saarbrücken:

Laser-based

Vision-based

Freiburg:
GP-based Terrain Modeling

- **3D terrain models** are important in many tasks in outdoor robotics
Terrain Modeling

- **Given:** observations of the terrain surface
- **Task:** Learn a predictive model
- Classic Approach: Elevation grid maps
GP-Based Approach

- Generalize the grid-based model to **fully continuous spaces** by viewing the problem as function regression.

- Requirements
  - **Probabilistic** formulation to handle uncertainty
  - Ability to **adapt** to the spatial structures
Covariance Function

- Standard covariance function have limited flexibility to adapt to the local spatial structure.
Covariance Function

- What is **optimal** in this case?
Local Kernel Adaptation

- Adapt kernels based on the **terrain gradients**
- Covariance is adjusted according to the change in terrain elevation in the local neighborhood

\[
\Sigma_i = EST(x_i)^{-1} = (\nabla y(x_i))(\nabla y(x_i))^T
\]
Adapting to Local Structures

Ground truth

Stationary GP

Non-stationary GP
Adapting to Local Structure

- Model to deal with slowly changing characteristics and strong discontinuities
Experiments

standard

adaptive
Experiments

Observation (with white noise $\sigma=0.3$)

Kernels

Predicted Map

Local errors
Experiments – Stone Block
Experiments – Stone Block

Ground Truth

Observations
Experiments – Stone Block

Prediction
Experiments – Slope

Observations & model:

Uncertainties:
Summary

- GPs are a flexible and practical approach to Bayesian regression
- Prior knowledge is encoded in a human understandable way
- Learned models can be interpreted
- Efficiency mainly depends on the number of training points
- Real-world problem sizes require approximations/sparsity/...