Robotics for Future Industrial Applications

Learning Global and Local Dynamic Models

Philipp Ennen, M.Sc.
Learning a Policy

\[
p(x_{t+1}|x_t, u_t) = \mathcal{N}(f(x_t, u_t), \Sigma)
\]

\[f(x_t, u_t) \approx A_t x_t + B_t u_t\]

\[A_t = \frac{\partial f}{\partial x_t} \quad B_t = \frac{\partial f}{\partial u_t}\]
What kind of models can we use?

**Gaussian process**

GP with input \((x, u)\) and output \(x'\)

Pro: very data-efficient

Con: not great with non-smooth dynamics

Con: very slow when dataset is big

**Neural Network**

Input is \((x, u)\), output ist \(x'\)

Pro: very expressive, can use lots of data

Con: not so great in low data regimes

**Gaussian Mixture Model**

GMM over \((x, u, x')\) tuples

Train on \((x, u, x')\), condition to get \(p(x'|x, u)\)

For i'th mixture element, \(p_i(x, u)\) gives region where the mode \(p_i(x'|x, u)\) holds

Pro: very expressive, if the dynamics can be assumed as piecewise linear

This week's focus!
Content

I. Gaussians
   I. Univariate Gaussian
   II. Multivariate Gaussian
   III. Conditioning (Bayes’ rule)

II. Gaussian Mixture Model

III. Learning Local Dynamic Models

Disclaimer: lots of linear algebra now. In fact, pretty much all computations with Gaussians will be reduced to linear algebra!
Gaussian

Univariate Gaussian

• Gaussian distribution with mean \( \mu \), and standard deviation \( \sigma \):

\[
X \sim \mathcal{N}(\mu, \sigma^2) \\
p(x; \mu, \sigma^2) = \frac{1}{\sigma \sqrt{2\pi}} \exp\left(-\frac{(x - \mu)^2}{2\sigma^2}\right)
\]
Gaussian

Properties of Gaussians

• Densities integrate to one:

\[
\int_{-\infty}^{\infty} p(x; \mu, \sigma^2) \, dx = \int_{-\infty}^{\infty} \frac{1}{\sigma \sqrt{2\pi}} \exp\left(-\frac{(x - \mu)^2}{2\sigma^2}\right) \, dx = 1
\]

• Mean:

\[
E_X[X] = \int_{-\infty}^{\infty} xp(x; \mu, \sigma^2) \, dx
= \int_{-\infty}^{\infty} x \frac{1}{\sigma \sqrt{2\pi}} \exp\left(-\frac{(x - \mu)^2}{2\sigma^2}\right) \, dx
= \mu
\]

• Variance:

\[
E_X[(X - \mu)^2] = \int_{-\infty}^{\infty} (x - \mu)^2 p(x; \mu, \sigma^2) \, dx
= \int_{-\infty}^{\infty} (x - \mu)^2 \frac{1}{\sigma \sqrt{2\pi}} \exp\left(-\frac{(x - \mu)^2}{2\sigma^2}\right) \, dx
= \sigma^2
\]
Gaussian

Multivariate Gaussian

\[
p(x; \mu, \Sigma) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp \left( -\frac{1}{2} (x - \mu)^\top \Sigma^{-1} (x - \mu) \right)
\]

\[
\int \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp \left( -\frac{1}{2} (x - \mu)^\top \Sigma^{-1} (x - \mu) \right) dx = 1
\]

- Remember: For a matrix \( A \in \mathbb{R}^{n \times n} \), \(|A|\) denotes the determinant of \( A \)
- Remember: For a matrix \( A \in \mathbb{R}^{n \times n} \), \( A^{-1} \) denotes the inverse of \( A \)
  - Rule: \( A^{-1}A = I = AA^{-1} \)
  - \( I \in \mathbb{R}^{n \times n} \) is the identity matrix with all diagonal entries equal to one, and all off-diagonal entries equal to zero
Gaussian

Multivariate Gaussian

• Mean:
  \[ E_X[X_i] = \int x_i p(x; \mu, \Sigma) dx = \mu_i \]
  \[ E_X[X] = \int x p(x; \mu, \Sigma) dx = \mu \]
  (integral of vector = vector of integrals of each entry)

• Covariance:
  \[ E_X[(X_i - \mu_i)(X_j - \mu_j)] = \int (x_i - \mu_i)(x_j - \mu_j)p(x; \mu, \Sigma) dx = \Sigma_{ij} \]
  \[ E_X[(X - \mu)(X - \mu)^\top] = \int [(X - \mu)(X - \mu)^\top p(x; \mu, \Sigma) dx = \Sigma \]
  (integral of matrix = matrix of integrals of each entry)
Multivariate Gaussian: examples

- $\mu = [1; 0]$
  $\Sigma = [1 \ 0; 0 \ 1]$

- $\mu = [-0.5; 0]$
  $\Sigma = [1 \ 0; 0 \ 1]$

- $\mu = [-1; -1.5]$
  $\Sigma = [1 \ 0; 0 \ 1]$
Gaussian

Multivariate Gaussian: examples

- $\mu = [0; 0]$
- $\Sigma = [1 \ 0 \ ; \ 0 \ 1]$
- $\mu = [0; 0]$
- $\Sigma = [0.6 \ 0 \ ; \ 0.6]$
- $\mu = [0; 0]$
- $\Sigma = [2 \ 0 \ ; \ 0 \ 2]$
Multivariate Gaussian: examples

- $\mu = [0; 0]$
- $\Sigma = [1 \ 0; 0 \ 1]$
- $\mu = [0; 0]$
- $\Sigma = [1 \ 0.5; 0.5 \ 1]$
- $\mu = [0; 0]$
- $\Sigma = [1 \ 0.8; 0.8 \ 1]$
Multivariate Gaussian: examples

- $\mu = [0; 0]$
- $\Sigma = [1 \ 0; 0 \ 1]$
- $\mu = [0; 0]$
- $\Sigma = [1 \ 0.5; 0.5 \ 1]$
- $\mu = [0; 0]$
- $\Sigma = [1 \ 0.8; 0.8 \ 1]$
Multivariate Gaussian: examples

- $\mu = [0; 0]$  
  $\Sigma = [1 \ -0.5 \ ; \ -0.5 \ 1]$

- $\mu = [0; 0]$  
  $\Sigma = [1 \ -0.8 \ ; \ -0.8 \ 1]$

- $\mu = [0; 0]$  
  $\Sigma = [3 \ 0.8 \ ; \ 0.8 \ 1]$
Conditioning a multivariate Gaussian

• Consider a multi-variate Gaussian

\[ \mathcal{N}(\mu, \Sigma) = \mathcal{N}\left(\begin{bmatrix} \mu_X \\ \mu_Y \end{bmatrix}, \begin{bmatrix} \Sigma_{XX} & \Sigma_{XY} \\ \Sigma_{YX} & \Sigma_{YY} \end{bmatrix}\right) \]

• And the precision matrix

\[ \Gamma = \Sigma^{-1} = \begin{bmatrix} \Sigma_{XX} & \Sigma_{XY} \\ \Sigma_{YX} & \Sigma_{YY} \end{bmatrix}^{-1} = \begin{bmatrix} \Gamma_{XX} & \Gamma_{XY} \\ \Gamma_{YX} & \Gamma_{YY} \end{bmatrix} \]

• And the partitioned multi-variate Gaussian:

\[
p(x; \mu, \Sigma) = \frac{1}{(2\pi)^{n/2} |\Sigma|^{1/2}} \exp \left( -\frac{1}{2} \left( \begin{bmatrix} x \\ y \end{bmatrix} - \begin{bmatrix} \mu_X \\ \mu_Y \end{bmatrix} \right)^T \begin{bmatrix} \Gamma_{XX} & \Gamma_{XY} \\ \Gamma_{YX} & \Gamma_{YY} \end{bmatrix} \left( \begin{bmatrix} x \\ y \end{bmatrix} - \begin{bmatrix} \mu_X \\ \mu_Y \end{bmatrix} \right) \right)
\]

• We have

\[
p(x|y_0) \propto p(x; \mu, \Sigma) \\
\propto \exp \left( -\frac{1}{2} \left( x - \mu_X \right)^T \Gamma_{XX} (x - \mu_X) - (x - \mu_X)^T \Gamma_{XY} (y_0 - \mu_Y) - \frac{1}{2} (y_0 - \mu_Y)^T \Gamma_{YY} (y_0 - \mu_Y) \right)
\]

\[
\propto \exp \left( -\frac{1}{2} \left( x - \mu_X \right)^T \Gamma_{XX} (x - \mu_X) - \Gamma_{XX} \Gamma_{XY}^{-1} \Gamma_{XY} (y_0 - \mu_Y) \right)
\]

\[
= \exp \left( -\frac{1}{2} \left( x - \mu_X + \Gamma_{XY}^{-1} \Gamma_{XY} (y_0 - \mu_Y) \right)^T \Gamma_{XX} (x - \mu_X + \Gamma_{XY}^{-1} \Gamma_{XY} (y_0 - \mu_Y)) \right)
\]

\[
\propto \exp \left( -\frac{1}{2} \left( x - \mu_X + \Gamma_{XX}^{-1} \Gamma_{XY} (y_0 - \mu_Y) \right)^T \Gamma_{XX} (x - \mu_X + \Gamma_{XX}^{-1} \Gamma_{XY} (y_0 - \mu_Y)) \right)
\]
Gaussian

Conditioning a multivariate Gaussian

• If

\[(X, Y) \sim \mathcal{N}(\mu, \Sigma) = \mathcal{N}\left(\begin{bmatrix} \mu_X \\ \mu_Y \end{bmatrix}, \begin{bmatrix} \Sigma_{XX} & \Sigma_{XY} \\ \Sigma_{YX} & \Sigma_{YY} \end{bmatrix}\right)\]

• Then:

\[X | Y = y_0 \sim \mathcal{N}(\mu_X - \Gamma_{XX}^{-1} \Gamma_{XY} (y_0 - \mu_Y), \Gamma_{XX})\]

\[= \mathcal{N}(\mu_X + \Sigma_{XY} \Sigma_{YY}^{-1} (y_0 - \mu_Y), \Sigma_{XX} - \Sigma_{XY} \Sigma_{YY}^{-1} \Sigma_{YX})\]

– Mean moved according to correlation and variance on measurement
– Covariance \(\Sigma_{XX | Y = y_0}\) does not depend on \(y_0\)
Content

I. Gaussians
   I. Univariate Gaussian
   II. Multivariate Gaussian
   III. Conditioning (Bayes’ rule)

II. Gaussian Mixture Model

III. Learning Local Dynamic Models

Disclaimer: lots of linear algebra now. In fact, pretty much all computations with Gaussians will be reduced to linear algebra!
Gaussian Mixture Model

Clustering

• Unsupervised Learning
• Detect patterns in unlabelled data
• Useful if you do not know what to look for
• Requires data, but no labels
Gaussian Mixture Model

Clustering

• Fundamental approach: Group similar instances

• Example: 2D pattern

• What does similarity mean?
Gaussian Mixture Models

What does similarity mean?

• Similarity is hard to define
  – But we know it if we see it

• The true meaning of similarity is a philosophical question. We will therefore choose a more pragmatic approach: we think of **distances** (rather than similarities) between vectors or correlations between random variables
Gaussian Mixture Models

Hard Clustering with K-Means

K-Means Algorithm

1. Choose $K$ random points as cluster centers
2. Assign datapoint to the nearest cluster center
3. Adjust cluster center so that it get’s the mean value of the associated points

Problem: correct assignment is hard!
- Sometimes distances can be deceiving!
- Cluster may overlap!

What about probabilistic clustering?
Gaussian Mixture Models

The General Gaussian Mixture Model Assumption

• “The probabilistic version of K-Means”

• Each cluster has not only mean $\mu_i$ but also associated covariance matrix $\Sigma_i$

• GMM is a linear combination of $K$ Gaussians given by

$$ p(x) = \sum_{k=1}^{K} p(x|k) P(k) $$

where $P(k)$ is mixture weight subject to constraints

$$ 0 \leq P(k) \leq 1 \quad \text{and} \quad \sum_{k=1}^{K} P(k) = 1 $$

and $p(x|k)$ is height of $k^{th}$ Gaussian at datapoint $x$
The General Gaussian Mixture Model Assumption

- Gaussian Mixture Model
  - $P(Y)$ is multinomial
  - $p(x|k)$ is a multivariate Gaussian Distribution

\[
P(X = x_j | Y = i) = \frac{1}{(2\pi)^{m/2} |\Sigma_i|^{1/2}} \exp\left[-\frac{1}{2} (x_j - \mu_i)^T \Sigma_i^{-1} (x_j - \mu_i)\right]
\]

Single Gaussian

Mixture of two Gaussians
Gaussian Mixture Models

Combine simple models into a complex model

\[ p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x | \mu_k, \Sigma_k) \]

Component
Mixing coefficient

\[ \forall k : \pi_k \geq 0 \quad \sum_{k=1}^{K} \pi_k = 1 \]
Gaussian Mixture Models

Combine simple models into a complex model: Example

(a) Gaussian Mixture Models

(b) Gaussian Mixture Models

(c) Gaussian Mixture Models
Gaussian Mixture Models

Soft-Clustering with Expectation Maximization and GMM

Expectation-Maximization Algorithm

1. Choose K random mean and covariances as clusters
2. E-Step: Calculate, for each point, the probabilities of it belonging to each of the clusters
3. M-Step: recalculate mean and covariance of each cluster, using the probability of belonging to each cluster
Learning a Policy

\[ p(x_{t+1}|x_t, u_t) = \mathcal{N}(f(x_t, u_t), \Sigma) \]

\[ f(x_t, u_t) \approx A_t x_t + B_t u_t \]

\[ A_t = \frac{\partial f}{\partial x_t} \quad B_t = \frac{\partial f}{\partial u_t} \]
Linear Gaussian Dynamics are defined as

\[ p(x_{t+1}|x_t, u_t) = \mathcal{N}(f_{xt}x_t + f_{ut}u_t + f_{ct}, F_t) \]

Local models

Linearized local dynamics

Goal: get the system dynamics \( p(x_{t+1}|x_t, u_t) \) for each timestep \( t \)
Data: samples generated by the previous controller \( \tilde{p}_i(u_t|x_t) \rightarrow \{(x_t, u_t, x_{t+1})_i\} \)

How can we determine linear Gaussian dynamics from few samples?
Learning Local and Global Models

Train GMM: **Global Dynamic Model**
- Uses data from nearby timesteps
- Uses data from prior iterations

Linearize: **Local Dynamic Model**
- Uses prior from local dynamic model
- Uses data from last iteration at timestep $t$
- Condition on given $(x_t, u_t)$

Train GMM on all $\{\tau_i\} = \{(x_t, u_t, x_{t+1})\}$

Obtain prior parameters from GMM (mean $\mu_0$, covariance $\Phi$, degree of freedom $n_0$, $m$ (number of datapoints))

Get normal-inverse-Wishart ($\mathcal{NIW}$) prior from GMM for a given tuple $(x_t, u_t, x_{t+1})$

Calculate posterior covariance using $\mathcal{NIW}$-Prior parameters
Calculate posterior mean using sampled data

Condition on given $(x_t, u_t)$

$p(x_{t+1}|x_t, u_t)$ Local linear Gaussian dynamics

Data contains samples from all previous iteration

Run $p(u_t|x_t)$ on robot, collect $\mathcal{D} = \{\tau_i\}$
Its your turn!

Visit the website and implement it!
Introduction to the tasks

Tasks for today and tomorrow

• Task 1:
  – Implement an LQR Backward and Forward pass
  – Try to understand it!
  – Test it with our test method

• Task 2:
  – Implement linearization of the dynamic model
  – Try to understand it!
  – Test it with our test-method
  – Test it on the Box2D Scenario

• Task 3:
  – Test it with Kinova Jaco 2 in simulation
  – Adjust cost function

• Task 4:
  – Test it with real Kinova Jaco 2
  – Adjust cost function
Task 2 – Installation procedure

Download source code (do it in your home directory: cd ~):
git clone https://github.com/philippente/ss2017_task2_dynamics.git

Edit .bashrc to set environment variables:
gedit ~/.bashrc

At the end of file, the lines should look like this:
source /opt/ros/indigo/setup.bash
source /home/useradmin/catkin_ws/devel/setup.bash
export
ROS_PACKAGE_PATH=$ROS_PACKAGE_PATH:/opt/ros/indigo/share:/opt/ros/indigo/stacks:/home/useradmin/ss2017_task2_dynamics:/home/useradmin/ss2017_task2_dynamics/src/gps_agent_pkg

Check if the blue part of the source folder and ROS_PACKAGE_PATH is correct!

Then save it and close it. Source the .bashrc (load the environment variables):
source ~/.bashrc

Now, compile some stuff:
cd ss2017_task2_dynamics
sh compile_proto.sh
cd /src/gps_agent_pkg
cmake .
make -j
Task 2 – Installation procedure

- Open PyCharm
- Import the folder ss2017_task2_dynamics as a new project
- Open within PyCharm: python/gps/algorithmdynamics/dynamics_lr_prior.py
- Task: Implement dynamics learning! Look at the website for advices: https://goo.gl/X5twgi

- You can test your implementation with a little test program
  - using a terminal, open the directory ss2017_task2_dynamics
  - Start the program with: python python/gps/dynamics_test.py
  - Was it successful?

- If it was successful, lets look at the Box2D scenario!
  - using a terminal, open the directory ss2017_task2_dynamics
  - Start it with python python/gps/gps_main.py box2d_arm_example
  - How is the performance?