Machine Learning for Systems and Control

5SC28

Lecture 8: Model Internalization-based Reinforcement Learning

dr. ir. Maarten Schoukens & dr. ir. Roland Tóth

Control Systems Group

Department of Electrical Engineering

Eindhoven University of Technology

Academic Year: 2020-2021 (version 1.1)

TU/e EINDHOVEN UNIVERSITY OF TECHNOLOGY

A Philosophical Point of View

- How humans learn to bike:

Humans build models of the environment
from samples they can observe

Generalization
capability of the model

A Philosophical Point of View

- How humans learn to bike:

Humans build models of the environment from samples they can observe

Generalization
capability of the model

Good regulator Theorem (Conant & Ashby)

Successful agent
iff internalization of a model of the environment

A Philosophical Point of View

- How humans learn to bike:

Humans build models of the environment from samples they can observe

Generalization
capability of the model

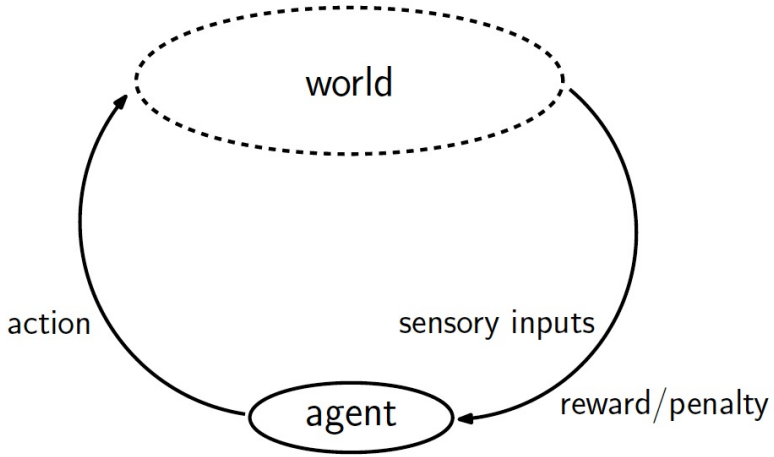
Good regulator Theorem (Conant & Ashby)

Successful agent
iff internalization of a model of the environment

Humans consider uncertainty of their models when making decisions

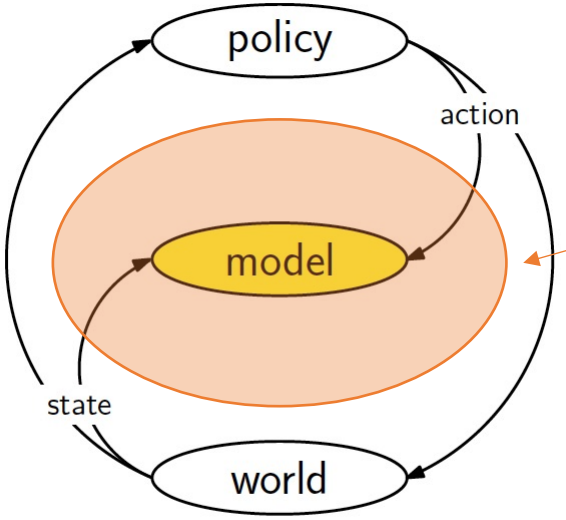
Efficient handling of uncertainty of the model

A Philosophical Point of View



Model free RL

A Philosophical Point of View



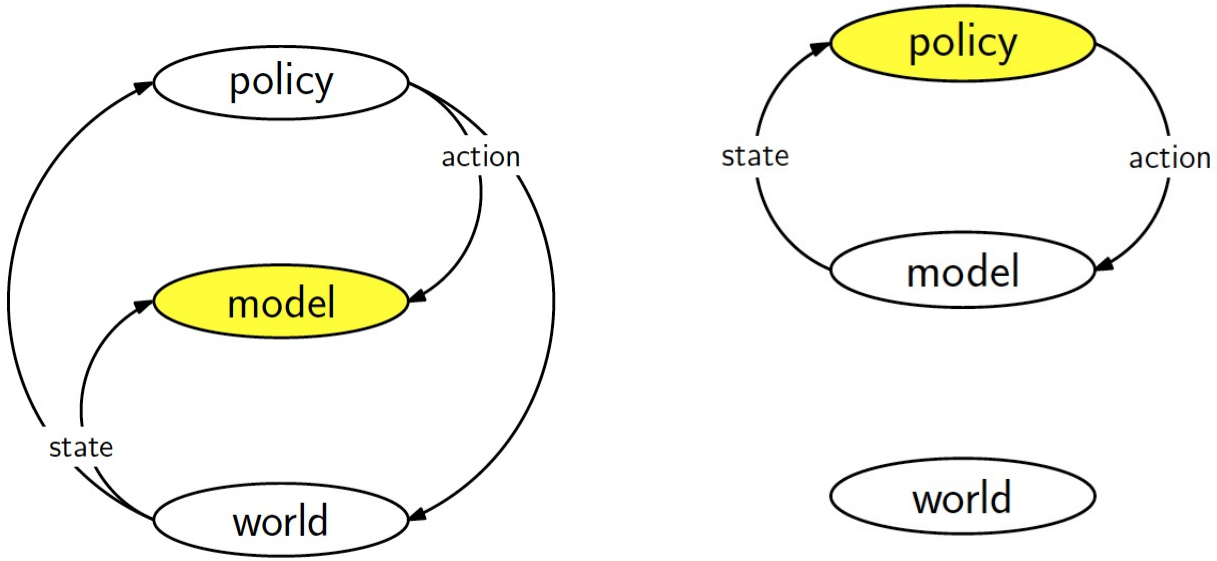
interaction

We captured this as the Q-function

Model free RL

- Estimation of the optimal policy directly (via interaction)
- Statistically inefficient

A Philosophical Point of View



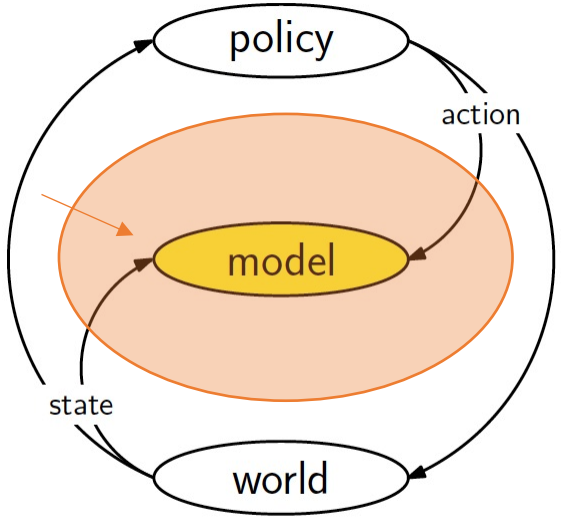
interaction *simulation*

Model learning RL

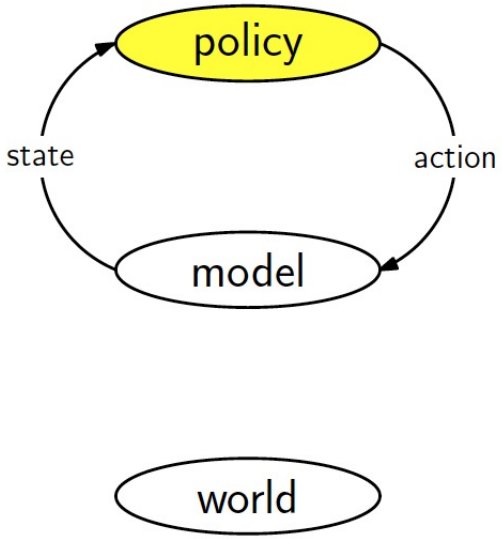
- In addition to interaction, simulation!

A Philosophical Point of View

Now we want to have a model of the dynamics



interaction



simulation

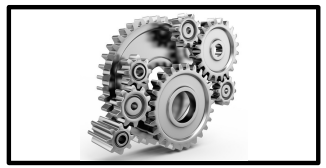
Model learning RL

- In addition to interaction, simulation!
- Policy is optimized based on simulations (efficiency)
- Model is learnt separately (identification)
- Model uncertainty (how to handle it?)

Contents

Problem setting

- Environment (world)
  - System represented as a Markov decision process



System

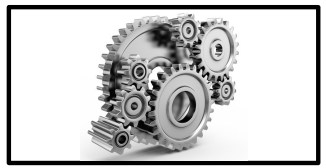
Deterministic function

$$x_t = f(x_{t-1}, u_{t-1})$$

$$x_t \in X \subseteq \mathbb{R}^{n_x}$$
$$u_t \in U \subseteq \mathbb{R}^{n_u}$$

Problem setting

- Environment (world)
  - System represented as a Markov decision process



System

Deterministic function

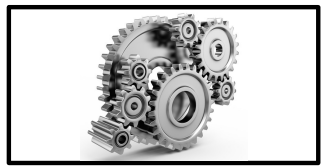
$$x_t = f(x_{t-1}, u_{t-1})$$

$$x_t \in X \subseteq \mathbb{R}^{n_x}$$
$$u_t \in U \subseteq \mathbb{R}^{n_u}$$

  - State constraints: no explicit constraints (possible to include them)

Problem setting

- Environment (world)
  - System represented as a Markov decision process



System

Deterministic function

$$x_t = f(x_{t-1}, u_{t-1})$$

$$x_t \in X \subseteq \mathbb{R}^{n_x}$$
$$u_t \in U \subseteq \mathbb{R}^{n_u}$$

- State constraints: no explicit constraints (possible to include them)
- Input constraints: $u_t \in [u_{\min}, u_{\max}]$

Problem setting

- Environment (world)
  - Reward function (rollout based):

Reward function

$$\sum_{t=0}^{T} r_t$$

Quadratic (MPC): $\sum_{t=0}^{T} \underbrace{(x_t - x_{\text{ref}})^\top Q (x_t - x_{\text{ref}})}_{c(x_t)}$ (cost)

Problem setting

- Environment (world)
  - Reward function (rollout based):

Reward function

$$\sum_{t=0}^{T} r_t$$

(cost)

Quadratic (MPC): $$\sum_{t=0}^{T} \underbrace{(x_t - x_{\text{ref}})^\top Q (x_t - x_{\text{ref}})}_{c(x_t)}$$

Saturated SE: $$\sum_{t=0}^{T} \underbrace{1 - \exp\left(-\frac{1}{2a}\|x_t - x_{\text{ref}}\|_2^2\right)}_{c(x_t)}$$

Problem setting

- Environment (world)
  - Reward function (rollout based):

Reward function

$$\sum_{t=0}^{T} r_t$$

(cost)

Quadratic (MPC): $$\sum_{t=0}^{T} \underbrace{(x_t - x_{\text{ref}})^\top Q (x_t - x_{\text{ref}})}_{c(x_t)}$$

Saturated SE: $$\sum_{t=0}^{T} \underbrace{1 - \exp\left(-\frac{1}{2a}\|x_t - x_{\text{ref}}\|_2^2\right)}_{c(x_t)}$$

...

Reward-cost relationship:

$$r_t = 1 - c(x_t)$$

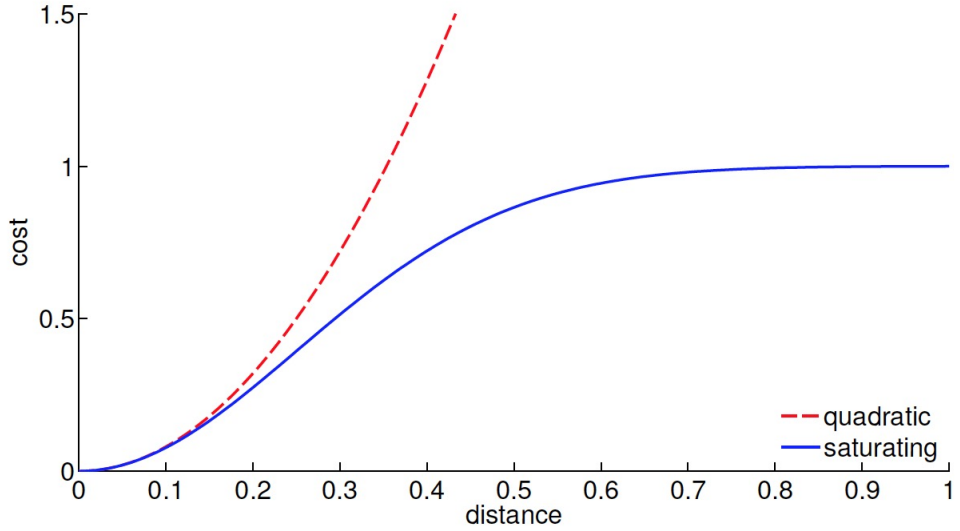
Problem setting

- Environment (world)
  - Comparison of reward functions:

Reward function

$$c(x) = 1 - \exp\left(-\frac{1}{2a}\|x - x_{\text{ref}}\|_2^2\right)$$

← Simple mean and partial derivative



Promotes early exploration

Problem setting

- Learning objective
  - Find a parametrized policy (state-feedback)

$$\pi(x_t, \theta) \quad \Rightarrow \quad x_t = f(x_{t-1}, \pi(x_{t-1}, \theta))$$

Problem setting

- Learning objective
  - Find a parametrized policy (state-feedback)

$$\pi(x_t, \theta) \quad \Rightarrow \quad x_t = f(x_{t-1}, \pi(x_{t-1}, \theta))$$

  - That minimizes the value function (negative reward)

Episode length

$$V_\pi(x_0) = \mathbb{E}_\tau \left\{ \sum_{t=0}^{T} c(x_t) \right\} = \sum_{t=0}^{T} \mathbb{E}_{x_t} \{c(x_t)\}$$

State trajectory $(x_0 \; x_1 \; \cdots \; x_T)$

Contents

- A Philosophical Point of View
- Problem Setting
- Bayesian Concept on RL
- Model Learning
- Value Function Calculation
- Policy Parametrization
- Policy Optimization
- Examples

Bayesian Concept on Learning

- Idea:
  - Apply a probabilistic modelling framework (uncertainty of the model)
  - Use the Bayesian framework to incorporate uncertainty in planning and decision making

Bayesian Concept on Learning

- Idea:
  - Apply a probabilistic modelling framework (uncertainty of the model)
  - Use the Bayesian framework to incorporate uncertainty in planning and decision making
- RL divided into 3 hierarchical problems (3 entities)

Targets

- Model: learning transition dynamics (probabilistic) f

- Value: predicting effects of decisions by inference V_π

- Policy: optimization of the policy π_*

Bayesian Concept on Learning

- Idea:
  - Apply a probabilistic modelling framework (uncertainty of the model)
  - Use the Bayesian framework to incorporate uncertainty in planning and decision making
- RL divided into 3 hierarchical problems (3 entities)
  - Model: learning transition dynamics (probabilistic)
  - Value: predicting effects of decisions by inference
  - Policy: optimization of the policy

Targets

f

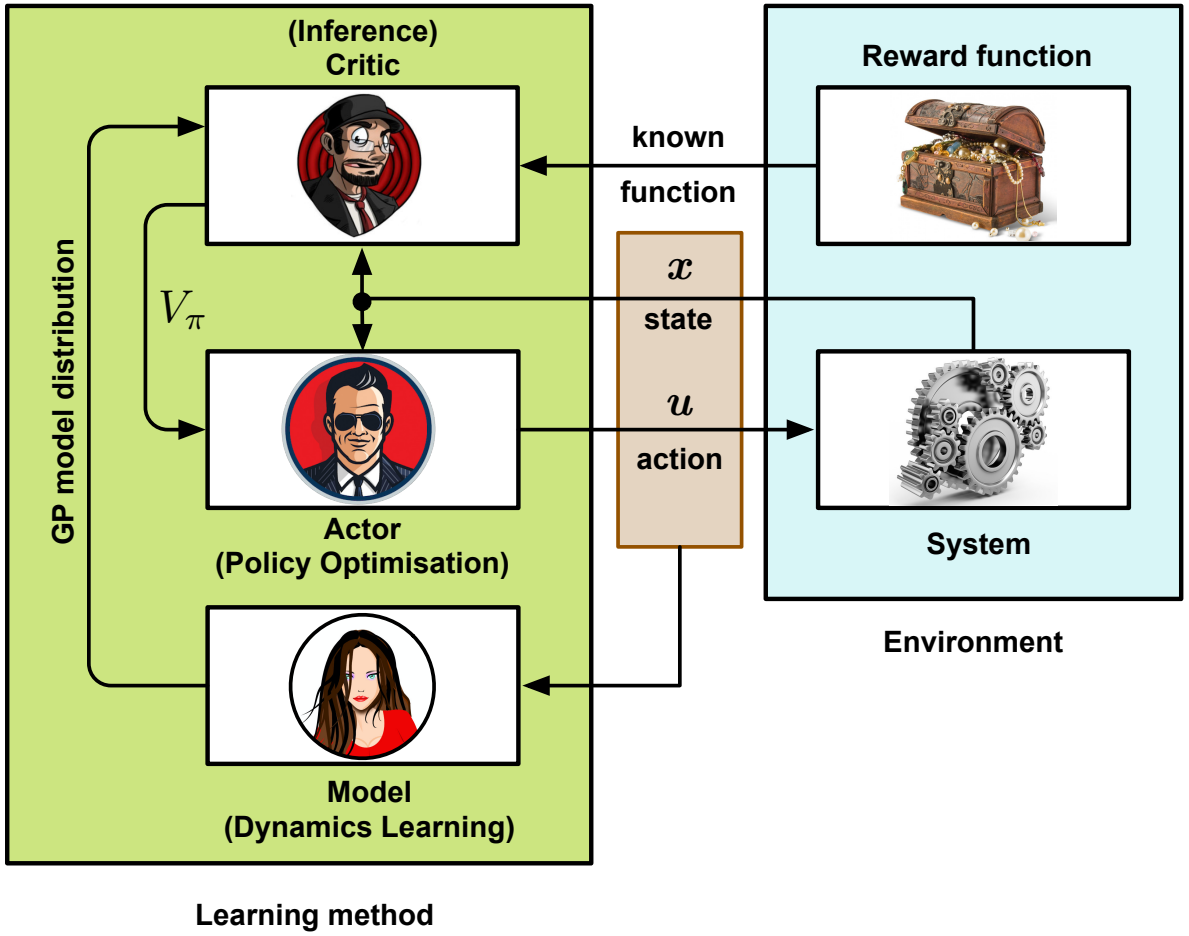
V_π

π_*

Probabilistic Inference & Learning Control

Bayesian Concept on Learning

- Concept:



Bayesian Concept on Learning

- PILCO algorithm:

Algorithm 1 PILCO

1: set policy to random ▷ policy initialization
2: **loop**
3: execute policy ▷ interaction
4: record collected experience
5: learn probabilistic dynamics model ▷ bottom layer
6: **loop** ▷ policy search
7: simulate system with policy π ▷ intermediate layer
8: compute expected long-term cost V^{π} ▷ policy evaluation
9: improve policy ▷ top layer
10: **end loop**
11: **end loop**

Contents

- A Philosophical Point of View
- Problem Setting
- Bayesian Concept on RL
- Model Learning
- Value Function Calculation
- Policy Parametrization
- Policy Optimization
- Examples

Model Learning

- GP based estimation (NARX)
  - Incremental modeling

$$\Delta x_{i,d} = f_d(\underbrace{x_i, u_i}) - x_{i,d}$$

(x_i, u_i)

i^{th}-state-action pair

d^{th}-element

Model Learning

- GP based estimation (NARX)
  - Incremental modeling

$$\Delta x_{i,d} = f_d(\underbrace{x_i, u_i}) - x_{i,d}$$

(x_i, u_i): i[th]-state-action pair

d[th]-element

  - GP probabilistic model: $p(f(x_i, u_i) \mid x_i, u_i) =$

$$\mathcal{N}\left(\begin{bmatrix} \mathbb{E}_f\{f_1(x_i,u_i) \mid x_i,u_i\} \\ \vdots \\ \mathbb{E}_f\{f_{n_x}(x_i,u_i) \mid x_i,u_i\} \end{bmatrix}, \begin{bmatrix} \mathrm{Var}_f\{f_1(x_i,u_i) \mid x_i,u_i\} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \mathrm{Var}_f\{f_{n_x}(x_i,u_i) \mid x_i,u_i\} \end{bmatrix}\right)$$

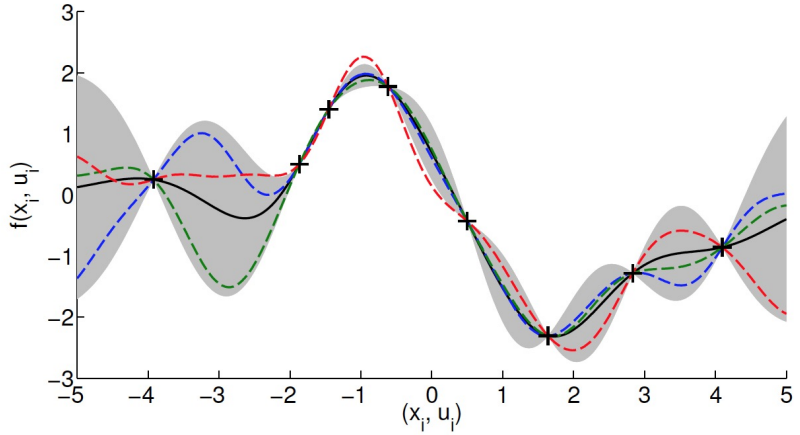
Model Learning

- GP based estimation (NARX)
  - Incremental modeling

$$\Delta x_{i,d} = f_d(\underbrace{x_i, u_i}) - x_{i,d}$$

(x_i, u_i): i[th]-state-action pair

$x_{i,d}$: d[th]-element



Gaussian process posterior as a distribution over transition functions

  - GP probabilistic model: $p(f(x_i, u_i) \mid x_i, u_i) =$

$$\mathcal{N}\left(\begin{bmatrix} \mathbb{E}_f\{f_1(x_i,u_i) \mid x_i,u_i\} \\ \vdots \\ \mathbb{E}_f\{f_{n_x}(x_i,u_i) \mid x_i,u_i\} \end{bmatrix}, \begin{bmatrix} \mathrm{Var}_f\{f_1(x_i,u_i) \mid x_i,u_i\} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \mathrm{Var}_f\{f_{n_x}(x_i,u_i) \mid x_i,u_i\} \end{bmatrix}\right)$$

Model Learning

- GP based estimation (NARX)
  - Prior:

$$p(\Delta x_d | x, u) = \mathcal{N}(m_{\Delta,d}, k_{\Delta,d} \mid (x, u))$$

pdf

Model Learning

- GP based estimation (NARX)
  - Prior:

$$p(\Delta x_d | x, u) = \mathcal{N}(m_{\Delta,d}, k_{\Delta,d} \mid (x, u))$$

$m_{\Delta,d} \equiv 0$

pdf

Model Learning

- GP based estimation (NARX)
  - Prior:

$$p(\Delta x_d | x, u) = \mathcal{N}(m_{\Delta,d}, k_{\Delta,d} \mid (x, u))$$

$m_{\Delta,d} \equiv 0$

pdf

$$k_{\Delta,d}(z, \tilde{z}) = \alpha^2 \exp\left(-\frac{1}{2}(z - \tilde{z})^\top \Lambda^{-1}(z - \tilde{z})\right)$$

diagonal, positive def.

SE kernel

Model Learning

- GP based estimation (NARX)
  - Prior:

$$p(\Delta x_d | x, u) = \mathcal{N}(m_{\Delta,d}, k_{\Delta,d} \mid (x, u))$$

$m_{\Delta,d} \equiv 0$

pdf

diagonal, positive def.

$$k_{\Delta,d}(z, \tilde{z}) = \alpha^2 \exp\left(-\frac{1}{2}(z - \tilde{z})^\top \Lambda^{-1}(z - \tilde{z})\right)$$

SE kernel (example)

  - Training a standard GP-estimator (Lecture 2)
    - Based on state-action data:

$$y_t = f(x_t, u_t) - x_t + e_t \quad \Rightarrow \quad \mathcal{D} = \{(y_t, x_t, u_t)\}_{t=0}^{T}$$

Model Learning

- GP based estimation (NARX)
  - Prior:

$$p(\Delta x_d | x, u) = \mathcal{N}(m_{\Delta,d}, k_{\Delta,d} \mid (x, u))$$

$m_{\Delta,d} \equiv 0$

pdf

diagonal, positive def.

$$k_{\Delta,d}(z, \tilde{z}) = \alpha^2 \exp\left(-\frac{1}{2}(z - \tilde{z})^\top \Lambda^{-1}(z - \tilde{z})\right)$$

SE kernel (example)

  - Training a standard GP-estimator (Lecture 2)
    - Based on state-action data:

$$y_t = f(x_t, u_t) - x_t + e_t \quad \Rightarrow \quad \mathcal{D} = \{(y_t, x_t, u_t)\}_{t=0}^{T}$$

    - Hyper parameters are tuned via the marginalized likelihood
    - GP models are often sparsified for data-management

Contents

- A Philosophical Point of View
- Problem Setting
- Bayesian Concept on RL
- Model Learning
- Value Function Calculation
- Policy Parametrization
- Policy Optimization
- Examples

Value Function Calculation

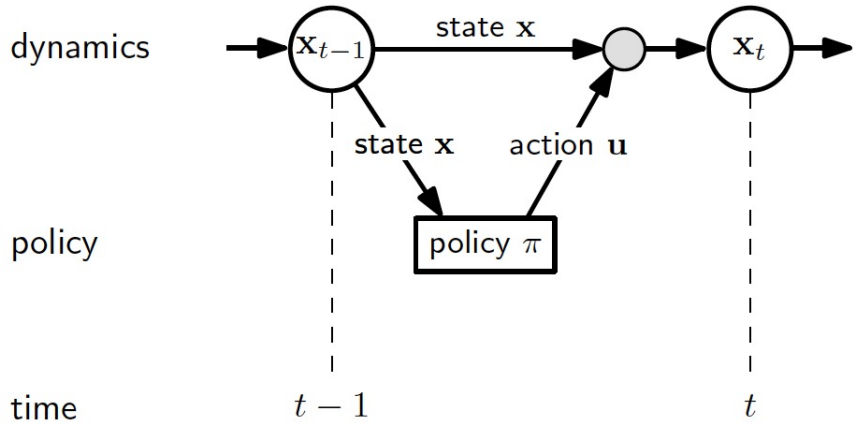
- Computation of V_π (via predictive inference)
  - We require to compute the distributions (probabilistic tube)

$$x_0 \rightarrow p(x_1) \rightarrow p(x_2) \rightarrow \ldots \rightarrow p(x_T)$$

Value Function Calculation

- Computation of V_π (via predictive inference)
  - We require to compute the distributions (probabilistic tube)

$$x_0 \rightarrow p(x_1) \rightarrow p(x_2) \rightarrow \ldots \rightarrow p(x_T)$$

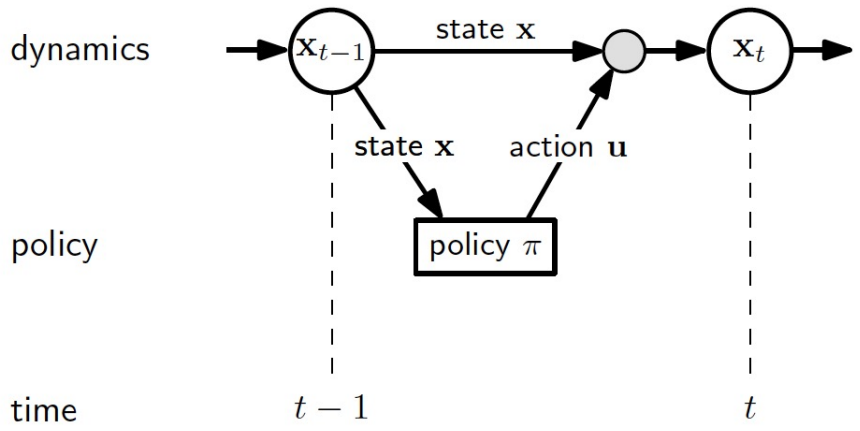


Similar to the prediction model in MPC, but now in a robust sense with a probabilistic model

Value Function Calculation

- Computation of V_π (via predictive inference)
  - We require to compute the distributions (probabilistic tube)

$$x_0 \rightarrow p(x_1) \rightarrow p(x_2) \rightarrow \ldots \rightarrow p(x_T)$$

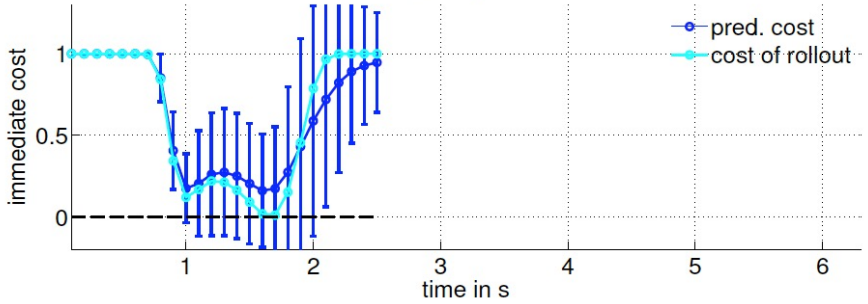


we can apply the cost function

$$V_\pi(x_0) = \mathbb{E}\left\{\sum_{t=0}^{T} c(x_t)\right\}$$

Similar to the prediction model in MPC, but now in a robust sense with a probabilistic model

Value Function Calculation

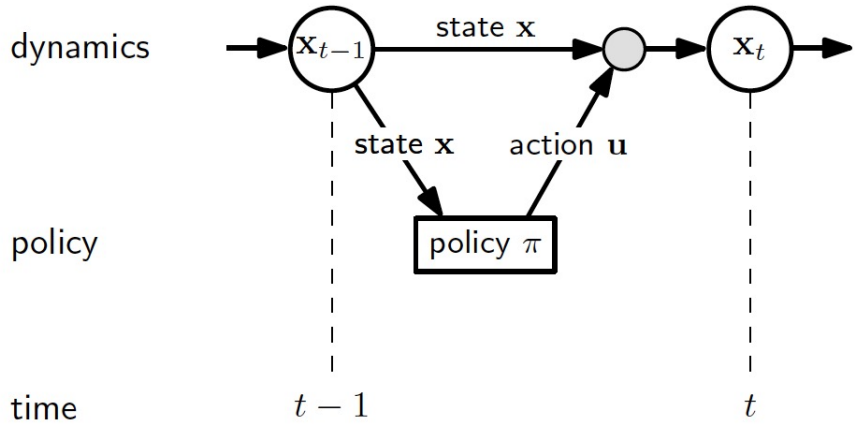


Probabilistic cost tube (95% confidence)

- Computation of V_π (via predictive inference)
  - We require to compute the distributions (probabilistic tube)

$$x_0 \rightarrow p(x_1) \rightarrow p(x_2) \rightarrow \ldots \rightarrow p(x_T) \longrightarrow$$ we can apply the cost function

$$V_\pi(x_0) = \mathbb{E}\left\{\sum_{t=0}^{T} c(x_t)\right\}$$



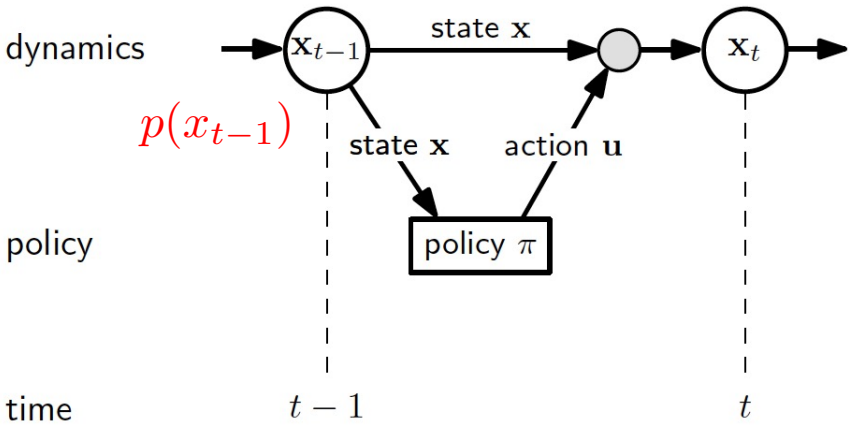
Similar to the prediction model in MPC, but now in a robust sense with a probabilistic model

Value Function Calculation

- Computation of V_π
  - Propagation step

Computing all these steps is required for policy iteration. How to do these efficiently?

$$p(x_t) = \int \int \underbrace{p(x_t \mid x_{t-1}, u_{t-1})}_{\text{GP model}} \cdot \underbrace{p(u_{t-1} \mid x_{t-1})}_{\pi \text{ policy}} \cdot \underbrace{p(x_{t-1})}_{\text{Previous step}} \; dx_{t-1} du_{t-1}$$



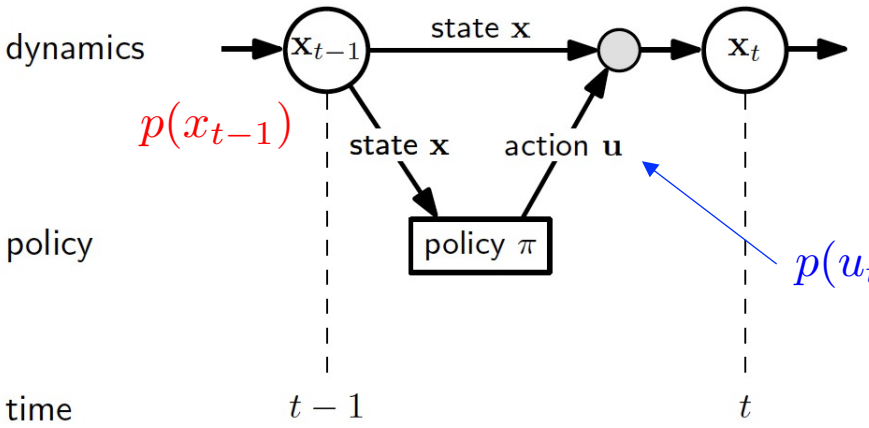
Joint Gaussian distribution

Value Function Calculation

- Computation of V_π
  - Propagation step

Computing all these steps is required for policy iteration. How to do these efficiently?

$$p(x_t) = \int \int \underbrace{p(x_t \mid x_{t-1}, u_{t-1})}_{\text{GP model}} \cdot \underbrace{p(u_{t-1} \mid x_{t-1})}_{\pi \text{ policy}} \cdot \underbrace{p(x_{t-1})}_{\text{Previous step}} \; dx_{t-1} du_{t-1}$$



Joint Gaussian distribution

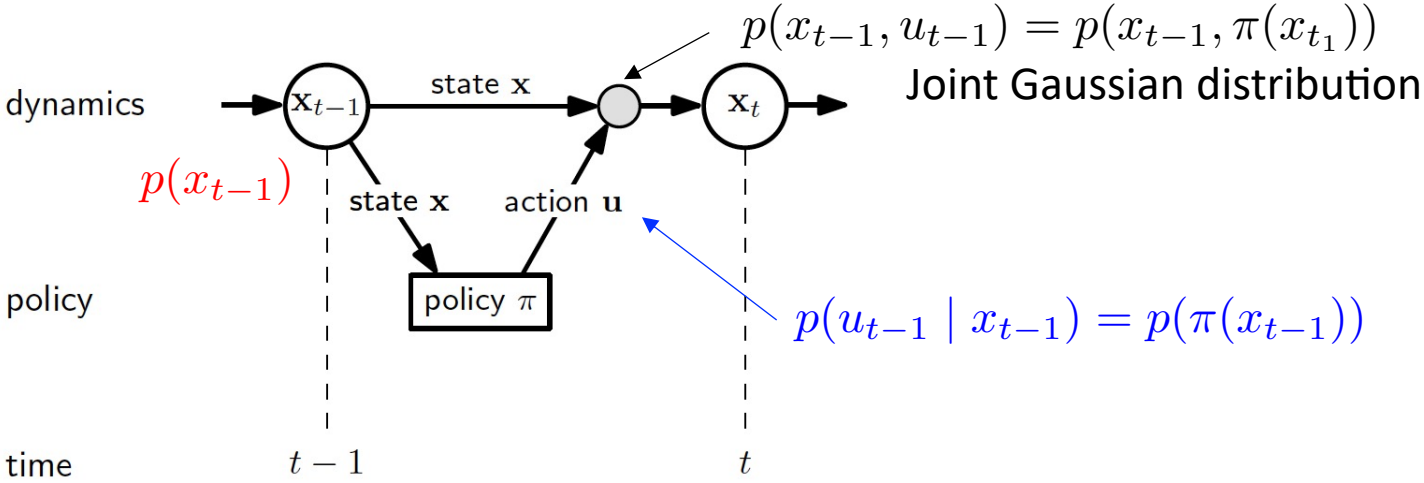
$$p(u_{t-1} \mid x_{t-1}) = p(\pi(x_{t-1}))$$

Value Function Calculation

- Computation of V_π
  - Propagation step

Computing all these steps is required for policy iteration. How to do these efficiently?

$$p(x_t) = \int \int \underbrace{p(x_t \mid x_{t-1}, u_{t-1})}_{\text{GP model}} \cdot \underbrace{p(u_{t-1} \mid x_{t-1})}_{\pi \text{ policy}} \cdot \underbrace{p(x_{t-1})}_{\text{Previous step}} \; dx_{t-1} du_{t-1}$$

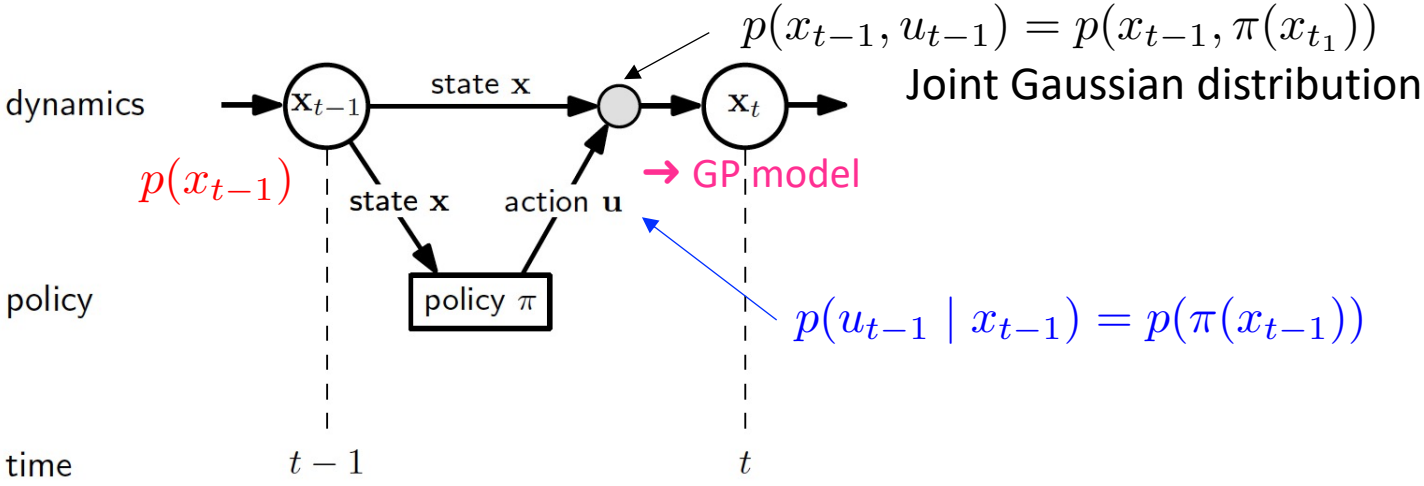


Value Function Learning

- Computation of V_π
  - Propagation step

Computing all these steps is required for policy iteration. How to do these efficiently?

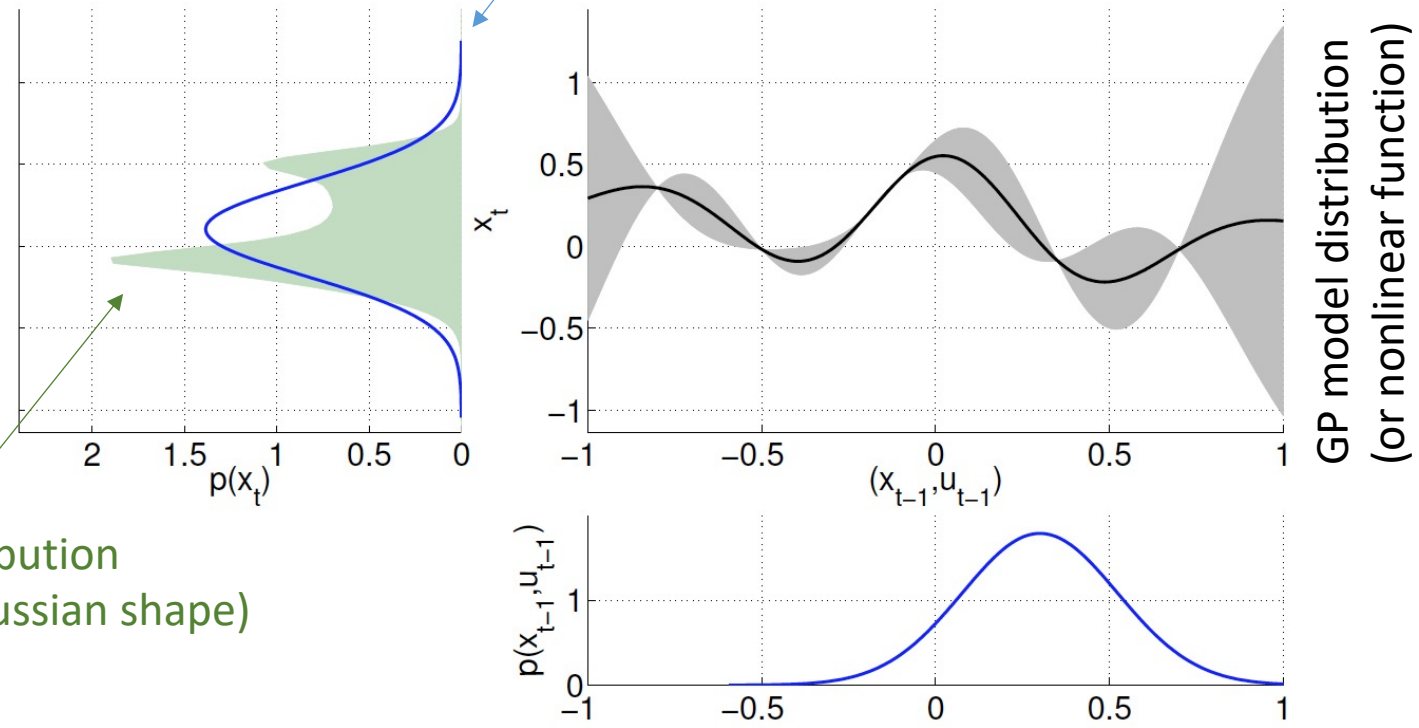
$$p(x_t) = \int \int \underbrace{p(x_t \mid x_{t-1}, u_{t-1})}_{\text{GP model}} \cdot \underbrace{p(u_{t-1} \mid x_{t-1})}_{\pi \text{ policy}} \cdot \underbrace{p(x_{t-1})}_{\text{Previous step}} \; dx_{t-1} du_{t-1}$$



Value Function Calculation

- Moment Matching

Analytical form of the mean and covariance
Idea: Match these moments with a Gaussian distribution



"output" distribution
(complicated non-Gaussian shape)

"input" distribution
(Gaussian)

Intermezzo

- Moment Matching (concept)
  - Consider the GP modelling problem

$$\mathbf{y}_t = \mathbf{f}(\mathbf{x}_t) + \mathbf{e}_t$$
$$\mathbf{f} \sim \mathcal{GP}(0, k)$$
$$\mathbf{e}_t \sim \mathcal{N}(0, \sigma_{\mathrm{e}}^2)$$

  - With a predictive distribution

$$\mu(x^*) = \mathbb{E}\{\mathbf{f}(x_*) \mid \mathcal{D}_N, x_*\} = K_x^\top(x_*)\overbrace{\left(K_{xx} + \sigma_{\mathrm{e}}^2 I_N\right)}^{\mathcal{K}}{}^{-1} Y$$
$$\sigma^2(x^*) = \mathrm{Var}\{\mathbf{f}(x_*) \mid \mathcal{D}_N, x_*\} = k(x_*, x_*) - K_x^\top(x_*)\mathcal{K}^{-1}K_x(x_*)$$

- What if $\mathbf{x}^* \sim \mathcal{N}(\nu, \lambda^2)$?

$$\mathbb{E}\{\mathbf{f}(\mathbf{x}^*) \mid \mathcal{D}_N, \mathbf{x}^*\} = ?$$
$$\mathrm{Var}\{\mathbf{f}(\mathbf{x}^*) \mid \mathcal{D}_N, \mathbf{x}^*\} = ?$$

Intermezzo

- Moment Matching (concept)

$$\mathbf{x}^* \sim \mathcal{N}(\nu, \lambda^2)$$

  - Mean: $\mathbb{E}_{x^*}\{\mathbb{E}_{f^*}\{\mathbf{f}^* \mid \mathbf{x}^*\}\} = \mathbb{E}_{x^*}\{\mu(\mathbf{x}^*)\}$

  - Variance: $\mathbb{E}_{x^*}\{\mathrm{Var}_{f^*}\{\mathbf{f}^* \mid \mathbf{x}^*\}\} + \mathrm{Var}_{x^*}\{\mathbb{E}_{f^*}\{\mathbf{f}^* \mid \mathbf{x}^*\}\} = \mathbb{E}_{x^*}\{\sigma^2(\mathbf{x}^*)\} + \mathrm{Var}_{x^*}\{\mu(\mathbf{x}^*)\}$

In case of the SE kernel these can be analytically computed (see [1]).

[1] J. Quionero-Candela, A. Girard, and C. Rasmussen, “Prediction at an uncertain input for Gaussian processes and relevance vector machines - application to multiple-step ahead time-series forecasting,” Max Planck Institute for Biological Cybernetics, Tubingen, Germany, Tech. Rep. IMM-2003-18, 2003.

Value Function Calculation

- Computation sequence of the state tube

$p(\mathbf{x}_{t-1})$ state distribution at time $t-1$ (given Gaussian)

1. ↓

$p(\mathbf{u}_{t-1})$ control distribution at time $t-1$ (analytic comp. of mean and variance) (moment matching ⇒ Gaussian)

2. ↓

$p(\mathbf{x}_{t-1}, \mathbf{u}_{t-1})$ joint distribution of state and control at time $t-1$ (analytic comp of covariance) (moment matching ⇒ Gaussian)

3. ↓

$p(\Delta\mathbf{x}_{t-1})$ predictive distribution of the change in state (analytic comp. of mean and variance) (moment matching ⇒ Gaussian)

4. ↓

$p(\mathbf{x}_t)$ state distribution at time t (via dynamics GP) (direct addition)

Value Function Calculation

- Computation sequence of the state tube

$p(\mathbf{x}_{t-1})$ state distribution at time $t-1$ (given Gaussian)

1. ↓

$p(\mathbf{u}_{t-1})$ control distribution at time $t-1$ (analytic comp. of mean and variance) (moment matching ⇒ Gaussian)

2. ↓

$p(\mathbf{x}_{t-1}, \mathbf{u}_{t-1})$ joint distribution of state and control at time $t-1$ (analytic comp of covariance) (moment matching ⇒ Gaussian)

3. ↓

$p(\Delta \mathbf{x}_{t-1})$ predictive distribution of the change in state (analytic comp. of mean and variance) (moment matching ⇒ Gaussian)

4. ↓

$p(\mathbf{x}_t)$ state distribution at time t (via dynamics GP) (direct addition)

Value Function Calculation

- Computation sequence of the state tube

$p(\mathbf{x}_{t-1})$ state distribution at time $t-1$ (given Gaussian)

1. ↓

$p(\mathbf{u}_{t-1})$ control distribution at time $t-1$ (analytic comp. of mean and variance) (moment matching ⇒ Gaussian)

2. ↓

$p(\mathbf{x}_{t-1}, \mathbf{u}_{t-1})$ joint distribution of state and control at time $t-1$ (analytic comp of covariance) (moment matching ⇒ Gaussian)

3. ↓

$p(\Delta \mathbf{x}_{t-1})$ predictive distribution of the change in state (analytic comp. of mean and variance) (moment matching ⇒ Gaussian)

4. ↓

$p(\mathbf{x}_t)$ state distribution at time t (via dynamics GP) (direct addition)

Value Function Calculation

- Computation sequence of the state tube

$p(\mathbf{x}_{t-1})$ — state distribution at time $t-1$ (given Gaussian)

1. ↓

$p(\mathbf{u}_{t-1})$ — control distribution at time $t-1$ (analytic comp. of mean and variance) (moment matching ⇒ Gaussian)

2. ↓

$p(\mathbf{x}_{t-1}, \mathbf{u}_{t-1})$ — joint distribution of state and control at time $t-1$ (analytic comp of covariance) (moment matching ⇒ Gaussian)

3. ↓

$p(\Delta\mathbf{x}_{t-1})$ — predictive distribution of the change in state (analytic comp. of mean and variance) (moment matching ⇒ Gaussian)

4. ↓

$p(\mathbf{x}_t)$ — state distribution at time t (via dynamics GP) (direct addition)

Value Function Calculation

- Computation sequence of the state tube

$p(\mathbf{x}_{t-1})$ — state distribution at time $t-1$ (given Gaussian)

1. ↓

$p(\mathbf{u}_{t-1})$ — control distribution at time $t-1$ (analytic comp. of mean and variance) (moment matching ⇒ Gaussian)

2. ↓

$p(\mathbf{x}_{t-1}, \mathbf{u}_{t-1})$ — joint distribution of state and control at time $t-1$ (analytic comp of covariance) (moment matching ⇒ Gaussian)

3. ↓

$p(\Delta\mathbf{x}_{t-1})$ — predictive distribution of the change in state (analytic comp. of mean and variance) (moment matching ⇒ Gaussian)

4. ↓

$p(\mathbf{x}_t)$ — state distribution at time t (via dynamics GP) (direct addition)

Value Function Calculation

- Computation sequence of the state tube

$p(\mathbf{x}_{t-1})$ — state distribution at time $t-1$ (given Gaussian)

1. ↓

$p(\mathbf{u}_{t-1})$ — control distribution at time $t-1$ (analytic comp. of mean and variance) (moment matching ⇒ Gaussian)

2. ↓

$p(\mathbf{x}_{t-1}, \mathbf{u}_{t-1})$ — joint distribution of state and control at time $t-1$ (analytic comp of covariance) (moment matching ⇒ Gaussian)

3. ↓

$p(\Delta\mathbf{x}_{t-1})$ — predictive distribution of the change in state (analytic comp. of mean and variance) (moment matching ⇒ Gaussian)

4. ↓

$p(\mathbf{x}_t)$ — state distribution at time t (via dynamics GP) (direct addition)

The whole sequence of computations correspond to simple matrix projections (can be implemented efficiently)

Value Function Calculation

- Computation of V_π (via predictive inference)

$$V_\pi(x_0) = \mathbb{E}_\tau \left\{ \sum_{t=0}^{T} c(x_t) \right\} = \sum_{t=0}^{T} \mathbb{E}_{x_t} \left\{ c(x_t) \right\}$$

$$\mathbb{E}_{x_t} \left\{ c(x_t) \right\} = \int c(x_t) p(x_t) dx_t$$

Based on the choice of c,
efficient analytic form

Contents

- A Philosophical Point of View
- Problem Setting
- Bayesian Concept on RL
- Model Learning
- Value Function Calculation
- Policy Parametrization
- Policy Optimization
- Examples

Policy Parametrization

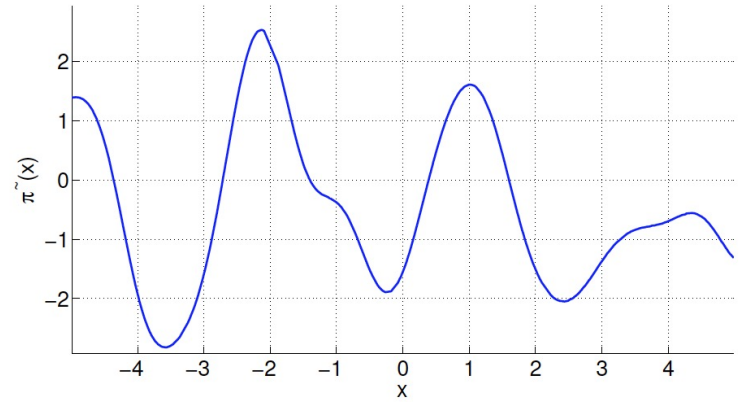
- How to parametrize the policy for efficient computation?
  - Input constrain satisfaction

Preliminary policy

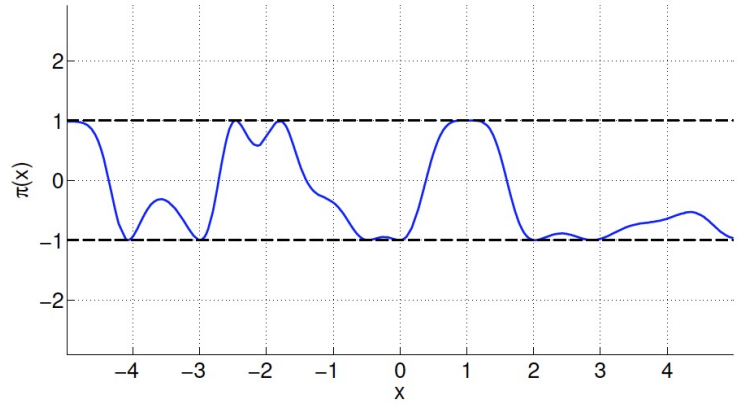
$$\pi(x) = w_{\mathrm{c}} + w_{\mathrm{s}} \sin(\tilde{\pi}(x)) \in [u_{\min}, u_{\max}]$$

$$w_{\mathrm{c}} = \frac{u_{\max} + u_{\min}}{2}$$

$$w_{\mathrm{s}} = \frac{u_{\max} - u_{\min}}{2}$$



(a) Preliminary policy $\tilde{\pi}$ as a function of the state.



(b) Policy $\pi = \sin(\tilde{\pi}(x))$ as a function of the state.

Policy Parametrization

- How to parametrize the policy for efficient computation?
  - Input constrain satisfaction

Preliminary policy

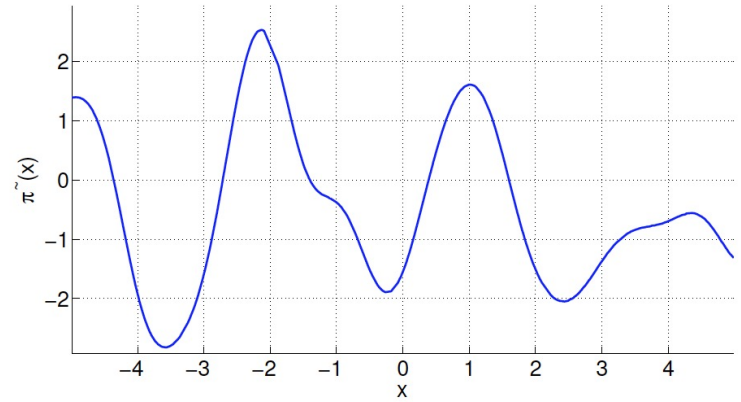
$$\pi(x) = w_{\mathrm{c}} + w_{\mathrm{s}} \sin(\tilde{\pi}(x)) \in [u_{\min}, u_{\max}]$$

$$w_{\mathrm{c}} = \frac{u_{\max} + u_{\min}}{2}$$

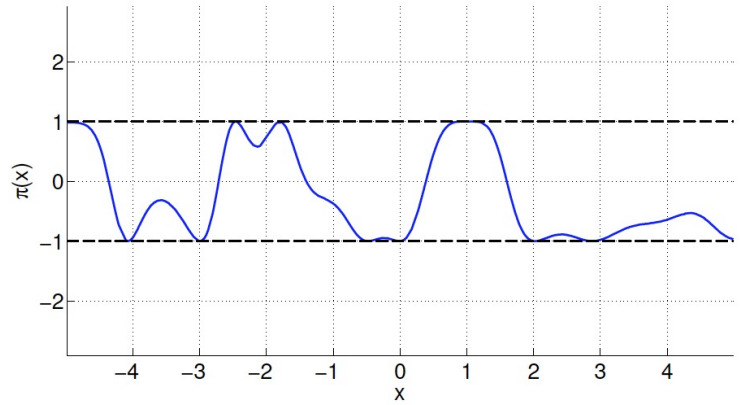
$$w_{\mathrm{s}} = \frac{u_{\max} - u_{\min}}{2}$$

Advantages:

- Analytic computations
- Attains maximum for finite values



(a) Preliminary policy $\tilde{\pi}$ as a function of the state.



(b) Policy $\pi = \sin(\tilde{\pi}(x))$ as a function of the state.

Policy Parametrization

- How to parametrize the policy for efficient computation?
  - Preliminary policy parametrization (linear)

Parameters θ

$$\tilde{\pi}(x) = \eta x + u_0$$

Number of parameters:

$$n_{\mathrm{u}}(n_{\mathrm{x}} + 1)$$

Policy Parametrization

- How to parametrize the policy for efficient computation?
  - Preliminary policy parametrization (linear)

Parameters θ

$$\tilde{\pi}(x) = \eta x + u_0$$

Number of parameters:

$$n_{\mathrm{u}}(n_{\mathrm{x}} + 1)$$

  - Predictive distribution for $p(x) = \mathcal{N}(\mu, \Sigma)$

$$\mathbb{E}_x\{\tilde{\pi}(x)\} = \eta\mu + u_0$$

$$\mathrm{Var}_x\{\tilde{\pi}(x)\} = \eta\Sigma\eta^\top$$

Policy Parametrization

- How to parametrize the policy for efficient computation?
  - Preliminary policy parametrization (RBF)

Coeff. $\beta_\pi = (K_\pi + \sigma_\pi^2 I)^{-1} y_\pi$

$$\tilde{\pi}(x) = \sum_{s=1}^{N} \beta_s k_\pi(x_s, x) = \beta_\pi^\top k_\pi(X_\pi, x)$$

Support vectors

Policy Parametrization

- How to parametrize the policy for efficient computation?
  - Preliminary policy parametrization (RBF)

Coeff. $\beta_\pi = (K_\pi + \sigma_\pi^2 I)^{-1} y_\pi$

$$\tilde{\pi}(x) = \sum_{s=1}^{N} \beta_s k_\pi(x_s, x) = \beta_\pi^\top k_\pi(X_\pi, x)$$

Support vectors

  - Predictive distribution for $p(x) = \mathcal{N}(\mu, \Sigma)$

GP trained with (X_π, y_π) ➡ $\tilde{\pi}(x)$ deterministic map ➡ analytical form

Support targets: $y_\pi = \tilde{\pi}(X_\pi) + \epsilon_\pi, \quad \epsilon_\pi \sim \mathcal{N}(0, \sigma_\pi^2 I)$

Policy Parametrization

Parameters: $\theta = \mathrm{vec}(X_\pi, y_\pi, \Lambda_\pi, \sigma_\pi)$

$n_\mathrm{u}(N(n_\mathrm{x}+1)+(n_\mathrm{x}+2))$

- How to parametrize the policy for efficient computation?
  - Preliminary policy parametrization (RBF)

Coeff. $\beta_\pi = (K_\pi + \sigma_\pi^2 I)^{-1} y_\pi$

$$\tilde{\pi}(x) = \sum_{s=1}^{N} \beta_\mathrm{s} k_\pi(x_\mathrm{s}, x) = \beta_\pi^\top k_\pi(X_\pi, x)$$

Support vectors

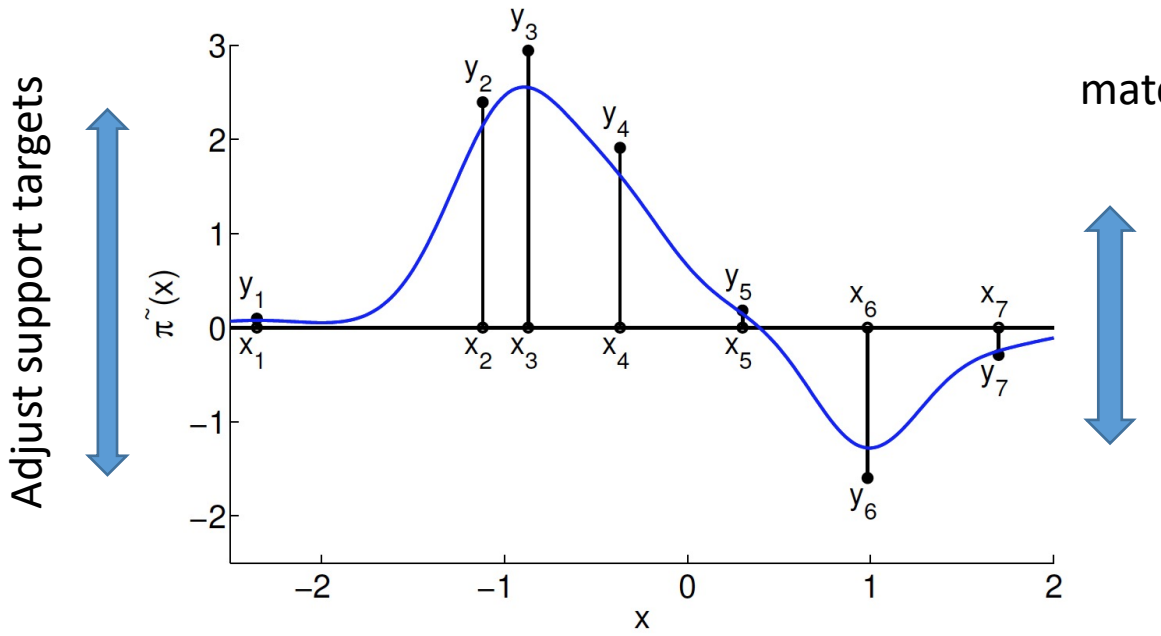
  - Predictive distribution for $p(x) = \mathcal{N}(\mu, \Sigma)$

GP trained with (X_π, y_π) ➡ $\tilde{\pi}(x)$ deterministic map ➡ analytical form

Support targets: $y_\pi = \tilde{\pi}(X_\pi) + \epsilon_\pi, \quad \epsilon_\pi \sim \mathcal{N}(0, \sigma_\pi^2 I)$

Policy Parametrization

- How to parametrize the policy for efficient computation?
  - Efficient storing of policy parametrization (RBF)



Such that the interpolation via
$\beta_\pi^\top k_\pi(X_\pi, \cdot)$
matches the desired shape

Adjust the hyper-parameters

Contents

- A Philosophical Point of View
- Problem Setting
- Bayesian Concept on RL
- Model Learning
- Value Function Calculation
- Policy Parametrization
- Policy Optimization
- Examples

Policy Optimization

- Gradient based policy search

$$\pi_* \in \arg\min_{\pi} V_\pi(x_0) \quad \Rightarrow \quad \theta_* \in \arg\min_{\theta} V_\pi(x_0)$$

  - Note that the predicted probabilistic state tube is described by matrix equations in the mean and variance that depend on θ !
  - Can be formulated for multiple initial states (replay).

Policy Optimization

- Gradient based policy search

$$\pi_* \in \arg\min_{\pi} V_\pi(x_0) \quad \Rightarrow \quad \theta_* \in \arg\min_{\theta} V_\pi(x_0)$$

  - Note that the predicted probabilistic state tube is described by matrix equations in the mean and variance that depend on θ !
  - Can be formulated for multiple initial states (replay).
  - Gradient of the predicted cost function:

$$\frac{d}{d\theta} V_\pi(x_0) = \sum_{t=0}^{T} \frac{d}{d\theta} \mathbb{E}_{x_t}\{c(x_t) \mid \pi_\theta\}$$

Policy Optimization

- Gradient propagation
  - Due to moment matching in inference:

$$p(x_t) \approx \mathcal{N}(\mu_t, \Sigma_t)$$

Propagated mean

Propagated covariance

Policy Optimization

- Gradient propagation
  - Due to moment matching in inference:

$$p(x_t) \approx \mathcal{N}(\mu_t, \Sigma_t)$$

Propagated mean

Propagated covariance

functions of $(\mu_{t-1}, \Sigma_{t-1})$

Policy Optimization

- Gradient propagation
  - Due to moment matching in inference:

$$p(x_t) \approx \mathcal{N}(\mu_t, \Sigma_t)$$

Propagated mean

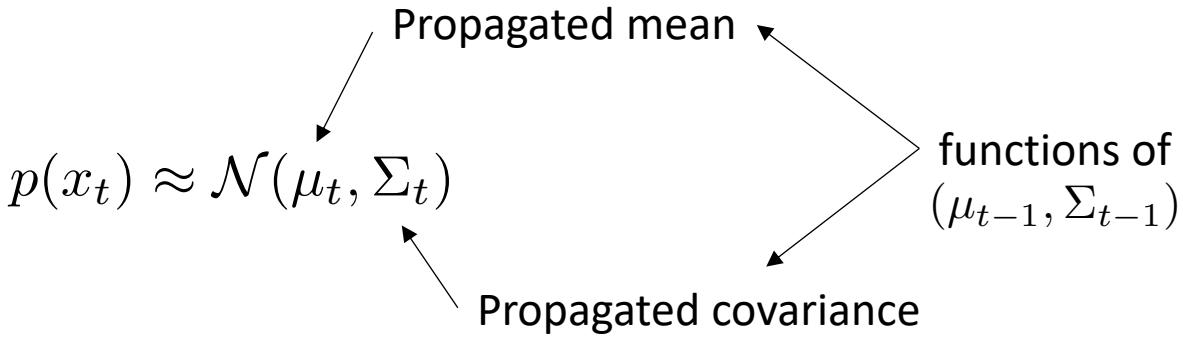
Propagated covariance

functions of $(\mu_{t-1}, \Sigma_{t-1})$

Propagation depends on the policy hence θ

Policy Optimization

- Gradient propagation
  - Due to moment matching in inference:



Propagation depends on the policy hence θ

  - Calculation:
    - Forward differentiation (Matlab)

$$\frac{d\mu_t}{d\theta} = \frac{d\mu_t}{d\mu_{t-1}} \frac{d\mu_{t-1}}{d\theta}$$

    - Backpropagation (Autograd Python)

$$\frac{dc_t}{d\mu_{t-1}} = \frac{dc_t}{d\mu_t} \frac{d\mu_t}{d\mu_{t-1}}$$

Policy Optimization

- Example: Forward Propagation

Only depends on c

$$\frac{d}{d\theta}\mathbb{E}_{x_t}\{c(x_t) \mid \pi_\theta\} = \left(\frac{\partial}{\partial \mu_t}\mathbb{E}_{x_t}\{c(x_t) \mid \pi_\theta\}\right)\frac{d\mu_t}{d\theta} + \left(\frac{\partial}{\partial \Sigma_t}\mathbb{E}_{x_t}\{c(x_t) \mid \pi_\theta\}\right)\frac{d\Sigma_t}{d\theta}$$

Only depends on c

Policy Optimization

$$\frac{d\mu_t}{d\theta} = \frac{\partial \mu_t}{\partial \mu_{t-1}} \frac{d\mu_{t-1}}{d\theta} + \frac{\partial \mu_t}{\partial \Sigma_{t-1}} \frac{d\Sigma_{t-1}}{d\theta} + \frac{\partial \mu_t}{\partial \theta}$$

- Example: Forward Propagation

Only depends on c

$$\frac{d}{d\theta}\mathbb{E}_{x_t}\{c(x_t) \mid \pi_\theta\} = \left(\frac{\partial}{\partial \mu_t}\mathbb{E}_{x_t}\{c(x_t) \mid \pi_\theta\}\right) \frac{d\mu_t}{d\theta} + \left(\frac{\partial}{\partial \Sigma_t}\mathbb{E}_{x_t}\{c(x_t) \mid \pi_\theta\}\right) \frac{d\Sigma_t}{d\theta}$$

Only depends on c

Policy Optimization

Analytic form
(simple matrix cal.)

$$\frac{d\mu_t}{d\theta} = \frac{\partial \mu_t}{\partial \mu_{t-1}} \frac{d\mu_{t-1}}{d\theta} + \frac{\partial \mu_t}{\partial \Sigma_{t-1}} \frac{d\Sigma_{t-1}}{d\theta} + \frac{\partial \mu_t}{\partial \theta}$$

- Example: Forward Propagation

Only depends on c

$$\frac{d}{d\theta}\mathbb{E}_{x_t}\{c(x_t) \mid \pi_\theta\} = \left(\frac{\partial}{\partial \mu_t}\mathbb{E}_{x_t}\{c(x_t) \mid \pi_\theta\}\right) \frac{d\mu_t}{d\theta} + \left(\frac{\partial}{\partial \Sigma_t}\mathbb{E}_{x_t}\{c(x_t) \mid \pi_\theta\}\right) \frac{d\Sigma_t}{d\theta}$$

Only depends on c

Policy Optimization

Analytic form
(simple matrix cal.)

$$\frac{d\mu_t}{d\theta} = \frac{\partial \mu_t}{\partial \mu_{t-1}} \frac{d\mu_{t-1}}{d\theta} + \frac{\partial \mu_t}{\partial \Sigma_{t-1}} \frac{d\Sigma_{t-1}}{d\theta} + \frac{\partial \mu_t}{\partial \theta}$$

Previous stage
(propagation)

- Example: Forward Propagation

Only depends on c

$$\frac{d}{d\theta}\mathbb{E}_{x_t}\{c(x_t) \mid \pi_\theta\} = \left(\frac{\partial}{\partial \mu_t}\mathbb{E}_{x_t}\{c(x_t) \mid \pi_\theta\}\right) \frac{d\mu_t}{d\theta} + \left(\frac{\partial}{\partial \Sigma_t}\mathbb{E}_{x_t}\{c(x_t) \mid \pi_\theta\}\right) \frac{d\Sigma_t}{d\theta}$$

Only depends on c

Policy Optimization

- Example: Forward Propagation

Analytic form (simple matrix cal.)

Stage cost (analytic form)

$$\frac{d\mu_t}{d\theta} = \frac{\partial \mu_t}{\partial \mu_{t-1}} \frac{d\mu_{t-1}}{d\theta} + \frac{\partial \mu_t}{\partial \Sigma_{t-1}} \frac{d\Sigma_{t-1}}{d\theta} + \frac{\partial \mu_t}{\partial \theta}$$

Previous stage (propagation)

Only depends on c

$$\frac{d}{d\theta}\mathbb{E}_{x_t}\{c(x_t) \mid \pi_\theta\} = \left(\frac{\partial}{\partial \mu_t}\mathbb{E}_{x_t}\{c(x_t) \mid \pi_\theta\}\right)\frac{d\mu_t}{d\theta} + \left(\frac{\partial}{\partial \Sigma_t}\mathbb{E}_{x_t}\{c(x_t) \mid \pi_\theta\}\right)\frac{d\Sigma_t}{d\theta}$$

Only depends on c

Policy Optimization

- Example: Forward Propagation

Analytic form (simple matrix cal.)

Stage cost (analytic form)

$$\frac{d\mu_t}{d\theta} = \frac{\partial \mu_t}{\partial \mu_{t-1}} \frac{d\mu_{t-1}}{d\theta} + \frac{\partial \mu_t}{\partial \Sigma_{t-1}} \frac{d\Sigma_{t-1}}{d\theta} + \frac{\partial \mu_t}{\partial \theta}$$

Previous stage (propagation)

Only depends on c

$$\frac{d}{d\theta}\mathbb{E}_{x_t}\{c(x_t) \mid \pi_\theta\} = \left(\frac{\partial}{\partial \mu_t}\mathbb{E}_{x_t}\{c(x_t) \mid \pi_\theta\}\right)\frac{d\mu_t}{d\theta} + \left(\frac{\partial}{\partial \Sigma_t}\mathbb{E}_{x_t}\{c(x_t) \mid \pi_\theta\}\right)\frac{d\Sigma_t}{d\theta}$$

Only depends on c

Analytic form (simple matrix cal.)

$$\frac{\partial \Sigma_t}{\partial \mu_{t-1}} \frac{d\mu_{t-1}}{d\theta} + \frac{\partial \Sigma_t}{\partial \Sigma_{t-1}} \frac{d\Sigma_{t-1}}{d\theta} + \frac{\partial \Sigma_t}{\partial \theta} = \frac{d\Sigma_t}{d\theta}$$

Previous stage (propagation)

Stage cost (analytic form)

Any gradient search method, like BFGS can be applied

The PILCO Approach

- Overview
  - On-policy, episodic (off-line), model learning approach
  - In principle it is an explicit, adaptive, robust, nonlinear, stochastic MPC
  - Highly efficient computational structure
  - No convergence or stability proof (due to moment-matching, gradient search)
  - Reference tracking is not trivial, but by appending the state, multiple set-points can be learnt.
  - Works remarkably well in practice!

Contents

- A Philosophical Point of View
- Problem Setting
- Bayesian Concept on RL
- Model Learning
- Value Function Calculation
- Policy Parametrization
- Policy Optimization
- Examples

https://youtu.be/oWKmgxZC5vk

Unicycle

- Challenging nonlinear system
- Starting from random initial conditions
- Objective:
  - Stabilize upright position
  - (x,y) position at the origin

https://youtu.be/XiigTGKZfks

Pendulum

- Physical experiments
- Need to learn complete swing-up motion (NL)
- Pose is encoded via

$$\sin(\theta), \cos(\theta)$$

- Objective:
  - keep upright position
  - saturated cost function

Conclusions

- Outlooks
  - Model internalization methods are the state-of-the-art of RL and the basis of
    - Learning based MPC (e.g. GP-MPC)
    - Learning methods with guaranteed safety (Safe Learning)
  - New research points towards
    - Data-based characterization of performance and stability certificates
    - Interpretability of ML (e.g., physical, human) and model/controller augmentation
    - Transparency / efficiency of learning
  - What we did not cover:
    - Generative Adversarial Learning (maybe next year)
    - Physics informed/driven DNNs