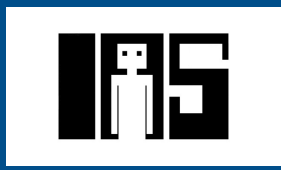
Wasserstein-Optimal Bayesian System Identification for Domain Randomization

Wasserstein-Optimale Bayessische Systemidentifikation für Domänen Randomisierung
Master thesis in the field of study “Computational Engineering” by Theo Gruner
Date of submission: October 29, 2021

1. Review: Fabio Muratore
2. Review: Boris Belousov
3. Review: Prof. Jan Peters
Darmstadt

TECHNISCHE
UNIVERSITÄT
DARMSTADT



Erklärung zur Abschlussarbeit gemäß §22 Abs. 7 APB TU Darmstadt

Hiermit versichere ich, Theo Gruner, die vorliegende Masterarbeit gemäß §22 Abs. 7 APB der TU Darmstadt ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die Quellen entnommen wurden, sind als solche kenntlich gemacht worden. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Mir ist bekannt, dass im Falle eines Plagiats (§38 Abs. 2 APB) ein Täuschungsversuch vorliegt, der dazu führt, dass die Arbeit mit 5,0 bewertet und damit ein Prüfungsversuch verbraucht wird. Abschlussarbeiten dürfen nur einmal wiederholt werden.

Bei einer Thesis des Fachbereichs Architektur entspricht die eingereichte elektronische Fassung dem vorgestellten Modell und den vorgelegten Plänen.

Darmstadt, October 29, 2021

T. Gruner

Abstract

Domain Randomization (DR) has recently proven to be an effective approach for learning control policies from randomized simulations to increase robustness against the model-plant mismatch. In particular, adaptive methods that update domain parameter distributions using Likelihood-Free Inference (LFI) were shown to be efficient at sim-to-real transfer. However, current LFI methods either require manually designed distance metrics to distinguish simulated and real trajectories or learn a costly global surrogate inverse model of the simulator to enable amortized inference. In this thesis, we propose an algorithm termed Wasserstein-optimal Likelihood-Free Inference, that estimates the intractable likelihood of a simulator using Energy-Based Models (EBMs) in which the energy functional is represented by the Wasserstein distance over trajectories, thereby addressing the challenges of the current LFI approaches. For the proposed method, we develop Bayesian inference routines based on Sequential Monte Carlo Approximate Bayesian Computation (SMC-ABC) and Relative Entropy Policy Search (REPS) algorithms. We establish a theoretical connection between SMC-ABC and REPS using maximum entropy principles, demonstrating that both approaches optimize the same objective function. This connection can be leveraged to apply REPS to typical LFI problems, and in particular, Bayesian system identification. The proposed method is validated on sim-to-sim and sim-to-real tasks and are compared against SNPE-C, a state-of-the-art LFI algorithm that sequentially updates a neural model of the posterior. The sim-to-sim experiments confirm that both SNPE-C and REPS are able to capture the domain parameter correlations of the posterior. The sim-to-real experiments on the cart-pole and the Furuta pendulum swing-up tasks demonstrate that the proposed Wasserstein-optimal LFI algorithm finds the parameter distributions that match the real trajectories better than SNPE-C in terms of the Dynamic Time Warping (DTW) discrepancy and the Mean Squared Error (MSE) between the real and simulated trajectories.

Zusammenfassung

Die Domänen Randomisierung (DR) hat sich in jüngster Zeit als wirksamer Ansatz für das Erlernen von Kontrollstrategien aus randomisierten Simulationen erwiesen, um die Robustheit gegenüber Modellfehlern zu erhöhen. Insbesondere adaptive Methoden, die Domänenparameterverteilungen mit Hilfe von Likelihood-Freier Inferenz (LFI) verbessern, haben sich als effizient bei der Übertragung von Simulationen in die Realität erwiesen. Aktuelle LFI Methoden erfordern jedoch entweder manuell entwickelte Abstandsmetriken, um simulierte und reale Trajektorien zu unterscheiden, oder sie lernen ein ineffizientes inverses Ersatzmodell des Simulators, um eine amortisierte Inferenz zu ermöglichen. In dieser Arbeit schlagen wir einen Algorithmus mit dem Namen Wasserstein-Optimale Likelihood-Freie Inferenz vor, der den schwer lösbaren Likelihood eines Simulators mit Hilfe von Energie-Basierten Modellen (EBMs) schätzt, wobei das Energiefunktional durch die Wasserstein-Distanz zwischen Trajektorien repräsentiert wird, wodurch die Herausforderungen der aktuellen LFI Ansätze addressiert werden. Für die vorgeschlagene Methode entwickeln wir Bayessche Inferenzroutinen, die auf den Algorithmen Sequential Monte Carlo Approximate Bayesian Computation (SMC-ABC) und Relative Entropy Policy Search (REPS) basieren. Wir stellen einen theoretische Zusammenhang zwischen SMC-ABC und REPS her, indem wir Maximale-Entropie-Prinzipien verwenden und zeigen, dass beide Ansätze dieselbe Zielfunktion optimieren. Die vorgeschlagenen Methoden werden in Sim-zu-Sim- und Sim-zu-Real Anwendungen validiert und mit SNPE-C verglichen, einem modernen LFI Algorithmus, der ein neuronales Modell des Posteriors fortlaufend aktualisiert. Die Sim-zu-Sim Experimente bestätigen, dass sowohl SNPE-C als auch REPS in der Lage sind, die Korrelationen zwischen den Domänenparametern des Posteriors zu erfassen. Die Sim-zu-Real Aufschwung-Experimente an einem inversen Pendel und dem Furuta Pendel zeigen, dass der Wasserstein-Optimale LFI Algorithmus diejenige Parameterverteilungen findet, die die realen Trajektorien besser darstellen können als SNPE-C mit Bezug auf die Diskrepanz der dynamischen Zeitnormierung (DTW) und den mittleren quadratischen Fehlern (MSE) zwischen den realen und simulierten Trajektorien.

Acknowledgments

I would like to thank my two supervisors, Fabio Muratore and Boris Belousov, for the intensive discussions, your insightful feedback, and great guidance throughout my thesis.

Contents

List of Acronyms

i.i.d.	independently and identically distributed
s.o.t.a	state-of-the-art
ABC	Approximate Bayesian Computation
APMC	Adaptive Population Monte Carlo
EBM	Energy-Based Model
DP	Domain Parameter
DR	Domain Randomization
DTW	Dynamic Time Warping
ESS	Effective Sample Size
FNN	Feed-Forward Neural Net
KL	Kullback-Leibler
LFI	Likelihood-Free Inference
MCMC	Markov Chain Monte Carlo
ML	Machine Learning
MAF	Masked Autoregressive Flow
MDP	Markov Decision Process
MSE	Mean Squared Error
PoWER	Policy Learning by Weighting Exploration with the Returns
PPO	Proximal Policy Optimization
RBF	Radial Basis Function
REPS	Relative Entropy Policy Search
RL	Reinforcement Learning
SBI	Simulation-Based Inference
SMC	Sequential Monte Carlo
SNDE	Sequential Neural Density Estimation
SNLE	Sequential Neural Likelihood Estimation
SNPE	Sequential Neural Posterior Estimation
SNRE	Sequential Neural Ratio Estimation

1. Introduction

Reinforcement Learning (RL) is a promising approach to solving complex robotic problems. Nevertheless, there are still very few physical systems that directly utilize RL. On one hand, the RL algorithms do not yet have sufficient capacity to scale to realistic, high-dimensional environments. On the other hand, policies trained solely in simulation are usually not applicable on physical systems. Such discrepancy is due to the simulation not being capable of grasping all the influences in the physical world. The mismatch between simulation and reality is commonly referred to as the 'reality gap'.

Applying policy optimization algorithms directly on the real device is time-consuming, error-prone and may lead to the harm of the robot, or even worse its environment. Ideally, policy optimization should be carried out in simulation and should only be transferred to the real system once it is safe. Therefore, algorithms for sim-to-real transfer have been proposed to make the transfer as robust as possible. One of these approaches is Domain Randomization (DR). In DR, policy optimization is carried out on randomized domains such that the trained policy is robust to environmental changes. Applications of DR in various robotic environments [1, 2, 3, 4, 5, 6] show that DR can bridge the sim-to-real gap.

Developed in the field of evolutionary biology, Likelihood-Free Inference (LFI) is a collective term for inference approaches for which it is impossible to compute the likelihood. Given a black-box simulator, LFI only requires reference data to infer the most likely parameters responsible for the observation. Recently, adaptive DR approaches [4, 6] have successfully integrated LFI as a Bayesian system identification routine into the DR framework.

Another approach towards Bayesian system identification is Relative Entropy Policy Search (REPS) [7]. Formally known from policy search, REPS presents an empirical inference method to infer the parameters of a parameterized distribution in a gradient free way [3].

In this thesis, we consider Bayesian system identification approaches which are based on LFI and the Wasserstein distance between trajectories. We show that a subclass of LFI

algorithms, Approximate Bayesian Computation (ABC), and REPS can be interpreted as inference algorithms which approximate the intractable likelihood with an EBM where the energy functional is the Wasserstein distance between observed and real trajectories. Therefore, we refer to these approaches as *Wasserstein-optimal Likelihood-Free Inference*.

We contribute to the state-of-the-art by (*i*) formulating the Wasserstein distance between trajectory distributions, (*ii*) presenting a unifying perspective on REPS and ABC as inference based on EBMs, and (*iii*) validating and carrying out an extensive ablation study for Wasserstein-optimal LFI.

The thesis is structured as follows: Chapter 2 outlines related work connected to Wasserstein-optimal LFI. In Chapter 3, the foundations are presented and subsequently the methodology is described. Afterwards, the experimental set-up is introduced in Chapter 4 and is followed by the experimental results and a final discussion in Chapter 5. Chapter 6 summarizes the findings and outlines future work.

2. Related Work

In this thesis, we employ Bayesian inference to infer the parameters of a dynamical system. The goal is to use the inferred system parameters for domain randomization. Putting our Bayesian system identification approach into a broader perspective, Section 2.1 introduces various domain randomization approaches. Section 2.2 summarizes a representative selection of recent LFI approaches. Section 2.3 highlights metrics for time series data and Section 2.4 presents statistical distances in Machine Learning (ML).

2.1. Domain Randomization (DR)

RL (see Section 3.1) trains policies to maximize the expected cumulative reward. In classical RL, a fixed simulator is used during the training procedure. It has been observed that policies tend to exploit this fact by finding solutions which explicitly fit to the specified domain [8]. In order to train policies which generalize to variations of the underlying dynamics, domain randomization aims at solving the problem by training on randomized domains

$$J(\boldsymbol{\theta}) = \mathop{\mathbb{E}}_{\boldsymbol{\xi}\sim p(\boldsymbol{\xi})} \left[\mathop{\mathbb{E}}_{\boldsymbol{\tau}\sim p_{\boldsymbol{\theta},\boldsymbol{\xi}}(\boldsymbol{\tau})} [R(\boldsymbol{\tau})] \right]. \tag{2.1}$$

As stated in Eq. (2.1), domain randomization extends the standard RL objective by an additional expectation over domain parameters. In general, a variety of domain parameters can be randomized, however, they must be manually selected for each task. Randomizing the friction coefficients, motor-specific parameters, or the geometric properties of the robot has shown to provide robust policies on underactuated swing-up tasks [5, 6] as well as on pushing and reaching tasks [2]. Domain parameters can also reflect other influences such as uncertainties of visual inputs. Grasping and pushing tasks [9] or drone control [1] are examples where the observed scene will change with each experiment as possible

disturbances can occur due to changes of the position and orientation of the camera or reference objects, or due to lighting changes.

Early approaches [9, 1] sample the parameters from a predefined uniform distribution. Although capable of bridging the reality gap, these approaches require fine tuned prior distributions and might lead to overly conservative policies. Therefore, recent approaches adapt the domain parameter distribution to represent the underlying parameter distribution of the target system.

SimOpt [3] is a DR approach which leverages a Bayesian system identification subroutine. Here, a loss between the simulated and the real-world reference data is defined. The domain parameter distribution of the real system is estimated by minimizing the loss w.r.t. the parameters $\boldsymbol{\phi}$ of $p_{\boldsymbol{\phi}}(\boldsymbol{\xi})$. To overcome the non-differentiability of the system dynamics, a gradient-free optimization algorithm is applied. The authors proposes to use episodic REPS [7]. The distance is chosen as a weighted l_1 and l_2 norm between the observations in addition with a Gaussian filter to account for misalignments. The parameter distribution is estimated by a parameterized multivariate Gaussian $p_{\phi}(\boldsymbol{\xi}) = \mathcal{N}(\boldsymbol{\xi}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ which can account for correlations between the domain parameters. SimOpt intertwines policy optimization with the Bayesian system identification method and apply their approach on a grasping and pulling task and a peg-in-a-hole task which moves away from the typical rigid motion assumption.

Muratore *et al.* [5] formulate a bi-level optimization problem of Eq. (2.1) which subsequently updates the parameters of the policy and of the domain parameter distribution. The domain parameter distribution is updated using sample efficient Bayesian optimization which is suited for black-box functions with low dimensional input space. The domain parameter distribution is fitted such that the expected real world return is maximized. The authors show promising sim-to-real results on a swing-up and stabilization task and in a ball-in-a-cup task where the proposed approach employs PPO [10] and Policy Learning by Weighting Exploration with the Returns (PoWER) [11] as their respective policy optimization subroutines.

Given recorded data from the reference system, Bayesian system identification can be applied to obtain the most likely domain parameters for a given set of observations. BayesSim [4] and its online variant [12] are two domain randomization algorithms that leverage LFI. The most recent LFI algorithms train neural posterior models based on pairs of domain parameters and simulated data. The inferred domain parameters can be obtained by conditioning the trained posterior on the observed data. Evaluated on various continuous tasks, BayesSim shows that the posterior obtained from LFI makes the policy optimization routine more robust. Online BayesSim builds upon this framework

by adding model predictive control to update the controller's parameters and show the results on a skid-steer robot. NPDR [6] extends the ideas introduced in BayesSim and learns a neural model of the posterior based on a normalizing flow. Normalizing flows are flexible density estimators that can represent complex distributions [13] and hence, can overcome the restriction for a specific type of parameterized distribution. Moreover, careful tuning of summary statistics of the trajectories can be circumvented as they are learned jointly with the neural model. The ability to depict multi-modal distributions is shown in a minigolf experiment. Moreover, the LFI subroutine is combined with PoWER on the Furuta pendulum to generate robust swing-up and stabilization policies.

2.2. Likelihood-Free Inference (LFI)

The previous section shows that Bayesian system identification is a promising subroutine for DR approaches. Specifically, we want to present LFI algorithms as a particular case of Bayesian inference. Inference routines from this family are specifically designed for problems where drawing samples from a stochastic model is possible but the likelihood is intractable. The survey of LFI approaches [14] offers an in-depth look into the LFI approaches described below.

2.2.1. Approximate Bayesian Computation (ABC)

ABC is one of the earliest and well known LFI approaches and is solely based on samples of the model parameters. In its simplest form, domain parameters are drawn from a prior distribution and are accepted if the simulated data is close to the observed data. For discrete data, it has been shown that the accepted domain parameter samples are drawn from the true posterior if the acceptance threshold goes to zero [15]. Several approaches have been proposed to improve on the vanilla rejection ABC method. Sampling strategies, such as Markov Chain Monte Carlo (MCMC) [16] and Sequential Monte Carlo (SMC) [17, 18, 19, 20, 21, 22], update the sampling distribution from which newly proposed samples are drawn iteratively.

2.2.2. Sequential Neural Density Estimation (SNDE)

Recently, neural density estimators have been proposed to approximate the posterior distribution. The simulator is used to generate a synthetic data set from which the neural model is trained. During the data generation process, parameter samples are first drawn from a sampling distribution and fed through the simulator to yield pairs of parameters and simulations. These samples are drawn from the joint distribution which is proportional to the posterior. The posterior can thus be learned from the samples by maximizing the log-probability of the density estimator. The recent success of neural density estimators in LFI can be attributed to the idea of sequentially updating the sampling distribution, originally proposed in [23]. This idea significantly improved the sample efficiency of the algorithms and made it possible to scale to higher dimensions and to solve more complex tasks.

Papamakarios and Murray [23] first proposed to approximate the posterior with a neural model. The algorithm, dubbed Sequential Neural Posterior Estimation (SNPE)-A, relies on an analytical solution using Gaussian conjugate priors which is why the neural model is restricted to mixture of Gaussians. The algorithms SNPE-B [24] and APT/SNPE-C [25] improve upon their predecessor such that any density estimator, e.g., Masked Autoregressive Flows (MAFs)[13], can be utilized.

Alternatively, Sequential Neural Likelihood Estimation (SNLE) [26] learns a neural model of the likelihood. With a tractable likelihood at hand, sampling strategies such as MCMC sampling can be applied to draw samples from the posterior distribution.

Sequential Neural Ratio Estimation (SNRE) approaches [27] estimate the likelihood-ratio $r(\boldsymbol{x}, \boldsymbol{\xi}) = p(\boldsymbol{x}|\boldsymbol{\xi})/p(\boldsymbol{x}) = p(\boldsymbol{\xi}|\boldsymbol{x})p(\boldsymbol{x})/p(\boldsymbol{\xi})p(\boldsymbol{x})$. The idea is to connect posterior estimation with contrastive learning which learns a binary classifier $r(\boldsymbol{x}, \boldsymbol{\xi})$ to discriminate positive and negative samples. In the likelihood-ratio, positive samples are considered to be sampled from the joint distribution $p(\boldsymbol{\xi}, \boldsymbol{x})$ and negative samples from $p(\boldsymbol{\xi})p(\boldsymbol{x})$. The paper sets the classifier to be the log of the likelihood-ratio $\log r(\boldsymbol{x}, \boldsymbol{\xi})$. The authors show that the binary classifier can be updated sequentially to sample in domain parameter regions which are likely to reproduce the observed data. Furthermore, they connect SNPE-C with SNRE, presenting a more general view on contrastive learning, and show that the likelihood-ratio can be recovered by SNPE-C.

2.3. Metrics for Time Series Data

Time series data is ubiquitous in the physical world. In robotics, time series most often appear as trajectories which comprise the information a robot gathers while moving through its environment. In nonlinear systems, two time series starting in close vicinity may diverge after few time-steps. Muskulus and Verduyn-Lunel [28] claim that comparing time series by pairwise comparisons of the time steps is often ineffective[28]. Therefore, the authors embed time series into a 'reconstruction space' in which each data point contains information about the correlations between time-steps over a fixed time horizon. Subsequently, statistical measures, such as the Wasserstein distance, are applied to compare two trajectories in their embedded representation.

Originally developed for speech recognition, DTW [29] finds the optimal alignments that minimize the distance between time steps of two time series. DTW is a dynamic programming algorithm which allows the comparison of time series of different length. The algorithm is especially interesting for time series which undergo different speed, e.g., comparing the gaits of two different persons. Soft-DTW [30] approximates an entropy-regularized version of the DTW objective which makes the discrepancy differentiable. The authors formulate the Soft-DTW discrepancy between time-series as a loss for classification and clustering tasks.

2.4. Statistical Distances in Machine Learning (ML)

The field of information geometry offers a set of tools for comparing probability distributions [31]. The Kullback-Leibler (KL) divergence, the Wasserstein distance, and the Jensen-Shannon divergence are examples of statistical distances employed in ML [31]. For example, the maximum likelihood approach for regression or classification, minimizes the KL divergence between a target and the approximate distribution. In this regard, statistical distances have been applied as a loss function for machine learning problems [32, 33] or can be applied as regularization terms in the loss function [7, 34].

The Wasserstein distance is defined as the solution of the optimal transport problem [35], which represents the minimal cost of moving mass from one probability measure to another. This optimal transport problem has gained attention in recent years due to its particularly favorable properties: (*i*) an optimal coupling between the two probability measures exists, (*ii*) the transport problem is convex, and (*iii*) the Wasserstein distance is symmetric [35].

With the goal to calculate the Wasserstein distance, numerical methods have been proposed. Cuturi [36] formulated a regularized optimal transport problem which can be solved using Sinkhorn's algorithm[37]. Sinkhorn iterations allow to parallelize the computations, resulting in significant speed-ups when using hardware accelerators such as GPUs.

3. Foundations and Methodology

In this chapter, an LFI approach to Bayesian system identification is presented that estimates a posterior based on the Wasserstein distance between ground truth data and simulated data. Section 3.1 introduces Markov Decision Processes (MDPs), as training data will be collected by rolling out trajectories in MDPs. Afterwards, Section 3.2 outlines an approach to calculate the discrepancy between two single trajectories using Dynamic Time Warping (DTW). To compare two empirical probability distributions over trajectories, Section 3.3 presents the Wasserstein distance as an example for measures of statistical distances. The proposed inference procedure leverages the Wasserstein distance over trajectories to carry out LFI.

3.1. Markov Decision Processes (MDPs)

Consider the tuple $(\mathcal{S}_{\boldsymbol{\xi}}, \mathcal{A}_{\boldsymbol{\xi}}, \mathcal{P}_{\boldsymbol{\xi}}, r, p_0(\boldsymbol{s}))$ forming a time-discrete Markov Decision Process [38] with continuous states $\boldsymbol{s} \in \mathcal{S}_{\boldsymbol{\xi}} \subset \mathbb{R}^n$ and continuous actions $\boldsymbol{a} \in \mathcal{A}_{\boldsymbol{\xi}} \subset \mathbb{R}^m$. The transition distribution $\mathcal{P} := p_{\boldsymbol{\xi}}(\boldsymbol{s}'|\boldsymbol{s}, \boldsymbol{a})$ is the probability moving from state $\boldsymbol{s}$ to state $\boldsymbol{s}'$ following action $\boldsymbol{a}$ for which a reward $r = r(\boldsymbol{s}, \boldsymbol{a})$ is collected. The system dynamics are defined by domain parameters which are assumed to be distributed according to $p(\boldsymbol{\xi})$. To complete the MDP, we denote the initial state distribution by $p_0(\boldsymbol{s}|\boldsymbol{\xi})$. In an MDP, an agent interacts with its environment following the policy $\pi_{\boldsymbol{\theta}}(\boldsymbol{a}|\boldsymbol{s})$ and collecting rewards. The goal of the agent is to maximize its discounted cumulative sum of rewards

$$J(\boldsymbol{\theta}) = \mathop{\mathbb{E}}_{\boldsymbol{\xi}\sim p(\boldsymbol{\xi})} \left[\mathop{\mathbb{E}}_{\boldsymbol{\tau}\sim p_{\pi_{\boldsymbol{\theta}}}(\boldsymbol{\tau})} [R(\boldsymbol{\tau})] \right]; \quad p_{\pi_{\boldsymbol{\theta}}}(\boldsymbol{\tau}) = p(\boldsymbol{s}_0|\boldsymbol{\xi}) \prod_{t=0}^{T-1} p_{\boldsymbol{\xi}}(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, \boldsymbol{a}_t) \pi_{\boldsymbol{\theta}}(\boldsymbol{a}_t|\boldsymbol{s}_t). \quad (3.1)$$

Trajectories $\boldsymbol{\tau} = \{\boldsymbol{s}_0, \boldsymbol{a}_t, r_t, \boldsymbol{s}_{t+1} | t = 0, ..., T-1\}$ comprise all states, actions, and rewards for all time steps which have been collected by an agent. The reward accumulated over the trajectory is $R(\boldsymbol{\tau}) = \sum_{t=0}^{T-1} \gamma^t r(\boldsymbol{s}_t, \boldsymbol{a}_t)$. Here, γ is a discount factor on returns.

3.2. Dynamic Time Warping (DTW)

Dynamic Time Warping (DTW) [29] is an algorithm to compare two time series of possibly different length. DTW finds the optimal alignments between the data points of two time series such that it minimizes the overall distance between these two time series. Here, we will follow the notation introduced in [29, 39]. Given two time sequences $\boldsymbol{\tau}_x = \{\boldsymbol{x}_0, \ldots, \boldsymbol{x}_M\}$ and $\boldsymbol{\tau}_y = \{\boldsymbol{y}_0, ..., \boldsymbol{y}_N\}$, we define the warping curve

$$\begin{aligned} \phi(t) &= (\phi_x(t), \phi_y(t)), \quad t = 1 \ldots T, \\ \phi_x(t) &\in \{0, ..., M\}, \\ \phi_y(t) &\in \{0, ..., N\}, \end{aligned}$$

where ϕ_x and ϕ_y contain the time steps of their respective time series. A warping curve $\phi(t)$ aligns the time steps of the data points $\boldsymbol{x}_{\phi_x(t)}$ and $\boldsymbol{y}_{\phi_y(t)}$ at time step t. The goal of DTW is to find the warping curve which minimizes the accumulated cost of all alignments

$$\rho(\boldsymbol{\tau}_x, \boldsymbol{\tau}_y) = \min_{\phi} \quad \frac{\sum_{t=1}^{T} d(\boldsymbol{x}_{\phi_x(t)}, \boldsymbol{y}_{\phi_y(t)})\, w(t)}{\sum_{t=1}^{T} w(t)}. \tag{3.2}$$

Here, $d(\boldsymbol{x}_m, \boldsymbol{y}_n) \geq 0$ is a dissimilarity function, e.g., the squared cost between single states of the two time series $\boldsymbol{\tau}_x$ and $\boldsymbol{\tau}_y$. $w(t)$ is a weighting coefficient which is normalized in order to compare time-series of different length. In order to only allow reasonable alignments between time steps, several constraints have been formulated. Monotonicity is insured by the following conditions on the warping curve:

$$\begin{aligned} \phi_x(k-1) &\leq \phi_x(t), \\ \phi_y(k-1) &\leq \phi_y(t) \end{aligned}$$

Continuity is insured by the following requirement:

$$\begin{aligned} \phi_x(t) - \phi_x(t-1) &\leq 1, \\ \phi_y(t) - \phi_y(t-1) &\leq 1 \end{aligned}$$

The alignments must meet the following boundary conditions on the starting and end position:

$$\begin{aligned} \phi_x(1) = 1,\, \phi_y(1) = 1, \\ \phi_x(T) = M,\, \phi_y(T) = N \end{aligned}$$

This Optimization Problem (3.2) can be solved using dynamic programming by making use of the following recursive formulation

$$g(t) = \min_{\phi(t-1)} \left[g(t-1) + d(\phi_x(t), \phi_y(t))\ w(t) \right], \tag{3.3}$$

$$\rho(\boldsymbol{\tau}_a, \boldsymbol{\tau}_b) = \frac{1}{\sum_{t=1}^{T} w(t)} g(T), \tag{3.4}$$

where $g(t)$ is the cost between $\boldsymbol{\tau}_x$ and $\boldsymbol{\tau}_y$ up to the t^{th} time step. The cost for time step t can be calculated from the cost up to $t-1$ and the pairwise comparison at t. Up to this point, the weighting coefficients and normalization have been neglected. Nevertheless, normalization and weighting is important in order to compare time-series with dissimilar warping curves. Step patterns [39] are a way to visualize the weightings in an elegant way, and furthermore, account for a warping slope which is neither too steep nor too flat. Step patterns define the possible transitions and the assigned weights from one pair matching $\phi(k)$ to the next one $\phi(k+1)$. A visualization of the step patterns is depicted in Fig. 3.1.

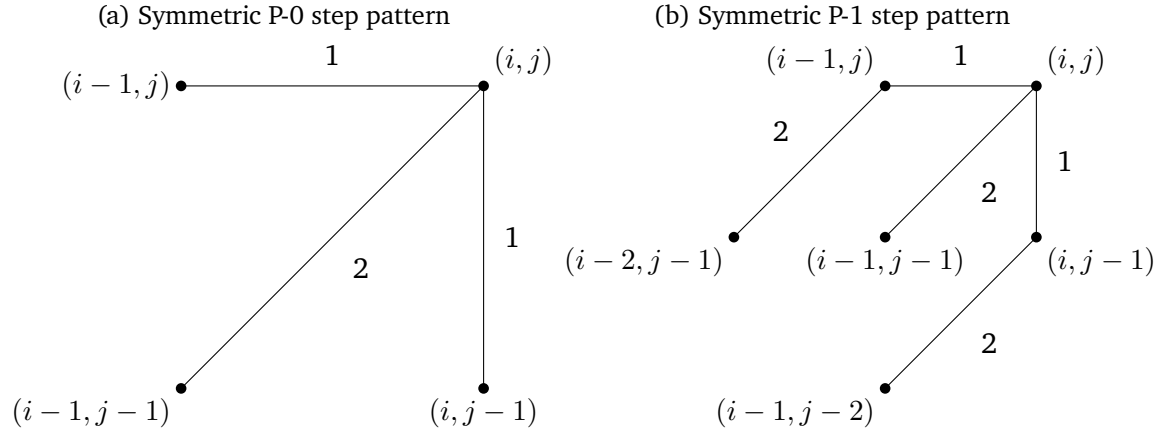


Figure 3.1.: Example step patterns for DTW. The weight along an edge denotes how much the distance between the pairs $\rho(\boldsymbol{x}_i, \boldsymbol{y}_j)$ is weighted. The recursion for the symmetric P-0 step pattern Eq. (3.5) gives further insights into this representation.

With these in mind, the dynamic programming recursion for the symmetric P-0 step

pattern is as follows:

$$g(i,j) = \min \left\{ \begin{array}{l} g(i,j-1) + d(i,j) \\ g(i-1,j-1) + 2d(i,j) \\ g(i-1,j) + d(i,j) \end{array} \right\} \tag{3.5}$$

Note here that the recursion is now over the pairs (i, j). A visualization of DTW with the prescribed step pattern is shown in Figure 3.3. The bandwidth b is a parameter which decides how far the pairwise comparison i and j should be skewed. If the time steps move too far away from each other, spatial correlation might not be given and hence, one should restrict the bandwidth.

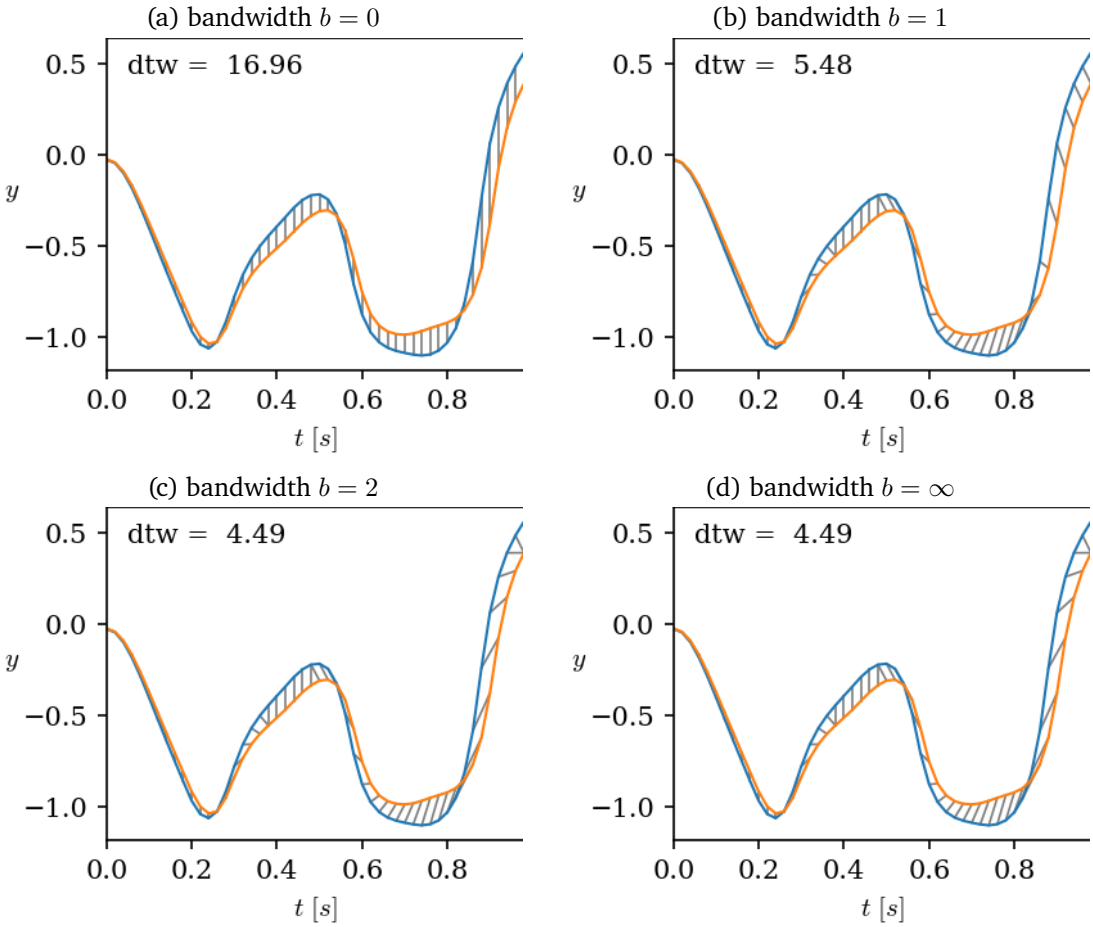


Figure 3.3.: Rollouts showing the DTW alignments for different bandwidths b. The bandwidth decides how far the elements $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ can be apart from each other, e.g., the bandwidth $b = 2$ says that $|i - j| <= 2$. Note, that for $b = 0$ DTW becomes the squared distance between single samples.

3.3. Wasserstein Distance

The Wasserstein distance is a statistical distance that can be used to compare two probability measures [35]. Assume two probability measures μ and ν defined on $\mathcal{X}$ and let

$(\mathcal{X}, \rho)$ be a polish space. Then, the p-Wasserstein distance between μ and ν is (see [35])

$$W_p(\mu, \nu) := \left(\inf_{\gamma \in \Gamma(\mu,\nu)} \int_{\mathcal{X}\times\mathcal{X}} \rho(\boldsymbol{x}, \boldsymbol{y})^p \mathrm{d}\gamma(\boldsymbol{x}, \boldsymbol{y}) \right)^{1/p}. \tag{3.6}$$

Here, $\gamma(\boldsymbol{x}, \boldsymbol{y})$ is a joint probability measure with marginals μ, ν and $\Gamma(\mu, \nu)$ denotes the set of all possible couplings of (μ, ν). The Wasserstein metric finds the optimal coupling which minimizes the cost $\rho(\boldsymbol{x}, \boldsymbol{y})$ of moving a particle $\boldsymbol{x}$ to $\boldsymbol{y}$.

In this thesis, the optimal transport problem for discrete measures is considered due to its numerical solutions presented in Section 3.3.1. Let us consider the discrete measures $\hat{\mu} = \sum_{i=1}^{M} \boldsymbol{\mu}_i \delta_{\boldsymbol{x}_i}$, $\hat{\nu} = \sum_{j=1}^{N} \boldsymbol{\nu}_j \delta_{\boldsymbol{x}_j}$, where δ denotes the Dirac delta distribution. The vectors $\boldsymbol{\mu}_i$ and $\boldsymbol{\nu}_j$ contain the probabilities assigned to the samples $\boldsymbol{x}_i$ and $\boldsymbol{y}_j$, respectively. As shown in [40], the optimal transport problem for the discrete case can be written as

$$W_p(\hat{\mu}, \hat{\nu}) = \left(\inf_{\gamma \in \Gamma(N,M)} \sum_{i=1}^{M} \sum_{j=1}^{N} \rho(\boldsymbol{x}_i, \boldsymbol{y}_j)^p \gamma_{i,j} \right)^{1/p}, \tag{3.7}$$

where $\Gamma(N, M)$ is the set of all coupling matrices $\gamma_{n,m}$, i.e., matrices whose rows sum up to μ and whose columns sum up to ν. If $N = M$, Eq. (3.7) can be reformulated as

$$W_p(\hat{\nu}, \hat{\mu}) = \left(\frac{1}{N} \inf_{\sigma \in \mathcal{S}_N} \sum_{i=1}^{N} \rho(\boldsymbol{x}_i, \boldsymbol{y}_{\sigma(i)})^p \right)^{1/p}. \tag{3.8}$$

Here, $\mathcal{S}_N$ denotes the set of all permutations of $\{1, ..., N\}$. This problem is known as the assignment problem and the solution to this problem is to find the optimal permutation σ^* to match the samples $\boldsymbol{x}_i$ to $\boldsymbol{y}_{\sigma^*(i)}$. In the following, we will only consider the case $p = 1$ which is known as the Monge-Kanterovic problem.

3.3.1. Numerical Solution to the Monge-Kanterovic Problem

The Assignment Problem (3.8) can be thought of as finding the permutation $\sigma(i)$ such that the distance between $\boldsymbol{x}_{\sigma(1:N)}$ and $\boldsymbol{y}_{1:N}$ is minimal. Therefore, the time complexity of the problem is $N \log N$ for one dimensional data and N^3 in multivariate settings [40].

The linear constraint program of the discrete optimal transport problem Eq. (3.7) is

$$\begin{aligned} \boldsymbol{L}_{\boldsymbol{C}}(\boldsymbol{\mu}, \boldsymbol{\nu}) &:= \min_{\boldsymbol{P} \in \boldsymbol{U}(\boldsymbol{\mu}, \boldsymbol{\nu})} \langle \boldsymbol{P}\boldsymbol{C} \rangle = \sum_{i,j} \boldsymbol{P}_{i,j} \boldsymbol{C}_{i,j}, \\ \text{s.t.} \quad \mu_i &= \sum_{i=1}^{n} \boldsymbol{P}_{i,j}, \\ \nu_j &= \sum_{j=1}^{m} \boldsymbol{P}_{i,j}, \end{aligned} \tag{3.9}$$

where $[\, \boldsymbol{C} \,]_{ij} = \rho(\boldsymbol{x}_i, \boldsymbol{y}_j)$ is the cost of moving $\boldsymbol{x}_i$ to $\boldsymbol{y}_j$ and $\boldsymbol{P}$ denotes a coupling matrix of $\boldsymbol{\mu}$ and ν. Again, the task is to find the optimal coupling which minimizes the total distance. The condition on the coupling matrix to be doubly stochastic are formalized as linear constraints. From linear optimization it is known that the given problem is convex and can be solved using standard simplex solvers for linear optimization.

Recently, effective approximations to the optimal control problem have been proposed by solving the regularized optimal transport problem [41]

$$\hat{\boldsymbol{L}}_{\boldsymbol{C}}(\boldsymbol{\mu}, \boldsymbol{\nu}) := \min_{\boldsymbol{P} \in \boldsymbol{U}(\boldsymbol{\mu}, \boldsymbol{\nu})} \langle \boldsymbol{P}\boldsymbol{C} \rangle - \varepsilon H(\boldsymbol{P}), \quad H(\boldsymbol{P}) := -\sum_{i,j} \boldsymbol{P}_{i,j} \left(\log \boldsymbol{P}_{i,j} - 1 \right). \tag{3.10}$$

Here, H is called the entropic regularization term of the coupling $\boldsymbol{P}$. The entropy is strictly concave which yields a strictly convex optimization problem with a unique optimal solution. The effect of the entropy is that for large ε the optimal solution favors maximum entropy solutions. In the extreme case, $\varepsilon \to \infty$, the optimal coupling $\boldsymbol{P}^*$ is the joint distribution between the two independent marginal distributions μ and ν. Contrary, the optimal solution to problem Eq. (3.10) converges to the optimal solution of the linear program Eq. (3.9) for $\varepsilon \to 0$ (see [41]).

It can be shown that the optimal coupling for the regularized optimal transport problem assumes the following form

$$\boldsymbol{P}_{i,j} = \boldsymbol{u}_i \underbrace{e^{-\frac{\boldsymbol{C}_{i,j}}{\varepsilon}}}_{\boldsymbol{K}_{i,j}} \boldsymbol{v}_j, \tag{3.11}$$

where $\boldsymbol{u}$ and $\boldsymbol{v}$ are two unknown vectors which have to be determined. $\boldsymbol{K}_{i,j}$ is the Gibbs kernel associated to the cost between the masses of μ and ν. It assigns large values towards couples which are close to each other. Fitting the vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ can be understood as finding the coupling $\boldsymbol{P}$ which projects the Gibbs kernel onto the space of doubly stochastic matrices (see constraints of Eq. (3.7)). Sinkhorn's algorithm [37] provides a solution to

iteratively updating the scaling vectors using only matrix multiplications. The vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ are updated one after another to yield the desired solution [41]

$$\boldsymbol{u}^{t+1} = \frac{\boldsymbol{\mu}}{\boldsymbol{K}\boldsymbol{v}^t}, \quad \boldsymbol{v}^{t+1} = \frac{\boldsymbol{\nu}}{\boldsymbol{K}\boldsymbol{u}^{t+1}}, \tag{3.12}$$

where division denotes pointwise division. This so called matrix scaling problem can be solved in $\mathcal{O}(n^2 \log n)$ operations. Sinkhorn iterations can be calculated using basic arithmetic operations which can be massively parallelized using GPUs. Furthermore, the gradient can be tracked such that the Wasserstein distance can be leveraged in gradient descent optimization problems.

3.4. Inference with Intractable Likelihoods

In the following, two inference algorithms, Approximate Bayesian Computation (ABC)[42] and Relative Entropy Policy Search (REPS) [7], are presented to obtain an approximation of the posterior $p(\boldsymbol{\xi}|\boldsymbol{\tau}_{1:N}^{\text{real}})$. The problem setting is that N observed data points $\boldsymbol{\tau}_{1:N}^{\text{real}}$ are available and the latent parameters $\boldsymbol{\xi}$, responsible for the observations, are to be inferred. Both algorithms face the problem that the likelihood is intractable, i.e., samples can be drawn from the likelihood $p(\boldsymbol{\tau}|\boldsymbol{\xi})$ but no probability can be assigned towards those. This case occurs with stochastic black-box simulators $g(\boldsymbol{\xi})$ whose output can be interpreted as samples from the likelihood. For conciseness, we establish the following notation: $p(\boldsymbol{\xi}|\boldsymbol{\tau}_{1:N})$ is the posterior, $p(\boldsymbol{\tau}|\boldsymbol{\xi})$ is the likelihood of a single trajectory, $p_0(\boldsymbol{\xi})$ is the prior, $p(\boldsymbol{\xi})$ is the sampling distribution, and $q(\boldsymbol{\xi}|\boldsymbol{\tau}_{1:N})$ denotes the approximate posterior which is connected to the sampling distribution via Bayes' rule

$$q(\boldsymbol{\xi}|\boldsymbol{\tau}_{1:N}) = \frac{\prod_{i=1}^{N} p(\boldsymbol{\tau}_i|\boldsymbol{\xi})p(\boldsymbol{\xi})}{\int \prod_{i=1}^{N} p(\boldsymbol{\tau}_i|\boldsymbol{\xi})p(\boldsymbol{\xi})\mathrm{d}\boldsymbol{\xi}}. \tag{3.13}$$

Here the data samples $\boldsymbol{\tau}_{1:N}$ are assumed to be i.i.d. random variables.

3.4.1. Approximate Bayesian Computation (ABC)

ABC uses sampling strategies to obtain samples from the posterior distribution. Several algorithms have been developed which use different sampling strategies to generate

Sampling strategy	Kernel	$K_\varepsilon\left(\rho\left(\boldsymbol{\tau}', \boldsymbol{\tau}\right)\right)$
Rejection, MCMC, SMC	Uniform	$\mathbb{1}_{(\rho(\boldsymbol{\tau}', \boldsymbol{\tau}) \leq \varepsilon)}$
MCMC, SMC	RBF	$\exp\left(-\frac{\rho(\boldsymbol{\tau}', \boldsymbol{\tau})}{2\varepsilon^2}\right)$

Table 3.1.: Kernels used in ABC to estimate the transition probability between the real and simulated data. ρ is any distance measure between the data $\boldsymbol{\tau}'$ and $\boldsymbol{\tau}$. The kernel is further parameterized by ε which serves as a bandwidth.

samples drawn from the underlying posterior. As shown in [42, 15], ABC approaches draw samples from the approximate posterior

$$q(\boldsymbol{\xi}|\boldsymbol{\tau}_{1:N}^{\text{real}}) \propto p(\boldsymbol{\xi}) \int K_\varepsilon\left(D(\boldsymbol{\tau}_{1:N}^{\text{real}}, \boldsymbol{\tau}_{1:M})\right) \prod_{m=1}^{M} p(\boldsymbol{\tau}_m|\boldsymbol{\xi}) \, \mathrm{d}\boldsymbol{\tau}_m. \tag{3.14}$$

A more intuitive understanding of the above expression can be gained by studying the general principle of ABC algorithms. The steps are: (i) sample from a sampling distribution $p(\boldsymbol{\xi})$, (ii) generate M data samples for each domain paramater from the generative model $g(\boldsymbol{\xi})$, and (iii) weigh the sample w.r.t. the observed data. These three steps are represented by the joint distribution over domain parameters and trajectories (see [42])

$$p(\boldsymbol{\xi}, \boldsymbol{\tau}_{1:M}|\boldsymbol{\tau}_{1:N}^{\text{real}}) \propto K_\varepsilon\left(D(\boldsymbol{\tau}_{1:N}^{\text{real}}, \boldsymbol{\tau}_{1:M})\right) p(\boldsymbol{\tau}_{1:M}|\boldsymbol{\xi}) p(\boldsymbol{\xi}). \tag{3.15}$$

We arrive at Eq. (3.14) by marginalizing Eq. (3.15) w.r.t. the M data samples and assuming i.i.d. data.

The kernel $K_\varepsilon\left(D(\boldsymbol{\tau}_{1:N}^{\text{real}}, \boldsymbol{\tau}_{1:M})\right)$ represents the transition probability for moving from the simulated data to the real data. It assigns high probability mass towards samples which are close to the real data and low probability mass towards samples which are not close to the real data. Typically, either a uniform or a Radial Basis Function (RBF) kernel is applied which are denoted in Table 3.1.

A straightforward sampling strategy is rejection ABC [43]. In rejection ABC, simulation parameters are drawn from a prior distribution and are accepted if the data, generated from the simulator, matches the real data up to a specified threshold ε. With respect to Eq. (3.14), rejection ABC uses a uniform kernel and the sampling distribution is always the prior distribution $p_0(\boldsymbol{\xi})$. The pseudocode is presented in Algorithm 1.

Algorithm 1: Rejection-ABC

Input: prior $p_0(\boldsymbol{\xi})$, generative model $g(\boldsymbol{\xi})$, distance $\rho(\boldsymbol{\tau}, \tilde{\boldsymbol{\tau}})$, acceptance threshold ε or quantile q, number of samples K, real world samples $\boldsymbol{\tau}^{\text{real}}$

Output: posterior samples $\{\boldsymbol{\xi}^i | \boldsymbol{\xi}^i \sim p(\cdot|\boldsymbol{\tau}^{\text{real}}), i = 1, ..., K\}$

for $i = 1, ..., K$ **do**
  repeat
    Sample candidate from prior $\boldsymbol{\xi}_0^i \sim p(\cdot)$
    Simulate $\boldsymbol{\tau}^i \sim g(\cdot|\boldsymbol{\xi}^i)$
  until $\rho(S(\boldsymbol{\tau}^{\text{real}}), S(\boldsymbol{\tau}^i)) < \varepsilon$

In general, rejection ABC does not scale well to high-dimensional problems as the prior will search the whole parameter space. Therefore, more sophisticated algorithms sequentially update the sampling distribution to be less diffusive each round.

In SMC-ABC [17, 44, 20], ideas from the particle filtering algorithm are applied, i.e., at each iteration a set of weighted particles $(\boldsymbol{\xi}^{(k)}, w^{(k)})$ is adapted such that it represents the current posterior distribution best. In order to sequentially move the sampling distribution from the prior towards the posterior, the epsilon threshold ε_t is adapted at each iteration step.

Given a prior set of particles drawn from the prior distribution $p_0(\boldsymbol{\xi})$, the SMC sampler follows five steps to improve upon the previous set of particles. First, the bandwidth ε is monotonically decreased every round. If a uniform kernel is assumed, this means that the upper threshold, for which data is accepted, is decreased every round. Afterwards, the particles are adapted by first, sampling from the previous weighted particle distribution. These samples are then perturbed using a transition kernel $G(\boldsymbol{\xi}_{t+1}, \boldsymbol{\xi}_t)$. This step moves the particle towards more likely parameter samples while maintaining the particle diversity. A renowned SMC-ABC algorithm [20] is shown in Algorithm 2.

Algorithm 2: Generalized SMC-ABC algorithm [20]

Input: prior $p_0(\boldsymbol{\xi})$, simulator $g(\boldsymbol{\xi})$, distance $D(\boldsymbol{\tau}'_{1:N}, \boldsymbol{\tau}_{1:M})$, number of samples K, number of samples per domain parameter L, number of simulated rollouts M, N real world samples $\boldsymbol{\tau}^{\text{real}}_{1:N}$, number of steps T, acceptance threshold ε, transition Kernel $G(\boldsymbol{\xi}_{t+1}, \boldsymbol{\xi}_t)$, distance Kernel $K_\varepsilon(\boldsymbol{\tau}_{t+1}, \boldsymbol{\tau}_t)$

Initialize: $t = 0$, $\varepsilon_0 = \infty$, $w_0^{(k)} = 1/K$

Sample from prior $\boldsymbol{\xi}_0^{(k)} \sim p_0(\boldsymbol{\xi}_k),\ (k = 1, ...K)$

Generate simulations $\boldsymbol{\tau}_0^{(k,l,m)} \sim g(\boldsymbol{\xi}_0^{(k)}),\ (l = 1, ..., L),\ (m = 1, ..., M)$

for $t = 1, ..., T$ **do**

  Determine ε_t by solving

$$\text{ESS}(\{w_t^{(k)}\}) = \alpha \text{ESS}(\{w_{t-1}^{(k)}\}) \tag{3.16}$$

$$\text{with} \quad w_t^{(k)} = w_{t-1}^{(k)} \frac{\sum_{l=1}^{L} K_{\varepsilon_t}\left(D(\boldsymbol{\tau}^{\text{real}}_{1:N}, \boldsymbol{\tau}_{t-1}^{(k,l,1:M)})\right)}{\sum_{l=1}^{L} K_{\varepsilon_{t-1}}\left(D(\boldsymbol{\tau}^{\text{real}}_{1:N}, \boldsymbol{\tau}_{t-1}^{(k,l,1:M)})\right)} \tag{3.17}$$

  Compute new particle weights $w_t^{(k)}$ using Eq. (3.17) and set particles $\boldsymbol{\xi}_t^{(k)} = \boldsymbol{\xi}_{t-1}^{(k)}$

  if $\text{ESS}(\{w_t^{(k)}\}) < N_t$ **then**

    Resample K new candidates $\boldsymbol{\xi}_t^{(k)} \sim \sum_{k=1}^{K} w_{t-1}^{(k)} \delta_{\boldsymbol{\xi}_{t-1}^{(k)}}$

    Set new weights $w_t^{(k)} = 1/K$

  for $k = 1, ..., K$ **do**

    Sample candidates $\tilde{\boldsymbol{\xi}}^{(k)} \sim G(\tilde{\boldsymbol{\xi}}^{(k)}, \boldsymbol{\xi}_t^{(k)}),\ (k = 1, ...K)$

    Generate candidate simulations $\tilde{\boldsymbol{\tau}}^{(k,l,m)} \sim g(\boldsymbol{\xi}^{(k)}),\ (l = 1, ..., L),\ (m = 1, ..., M)$

    Accept sample $\tilde{\boldsymbol{\xi}}^{(k)}$ with probability

$$\min\left\{1, \frac{\sum_{l=1}^{L} K_{\varepsilon_t}\left(D(\boldsymbol{\tau}^{\text{real}}_{1:N}, \boldsymbol{\tau}_{t-1}^{(k,l,1:M)})\right)\ p_0(\tilde{\boldsymbol{\xi}}^{(k)})\ G(\boldsymbol{\xi}_t^{(k)}, \tilde{\boldsymbol{\xi}}^{(k)})}{\sum_{l=1}^{L} K_{\varepsilon_t}(\rho(\boldsymbol{\tau}_t^{(k,n)}, \boldsymbol{\tau}^{\text{real}}))\ p_0(\boldsymbol{\xi}_t^{(k)})\ G(\tilde{\boldsymbol{\xi}}^{(k)}, \boldsymbol{\xi}_t^{(k)})}\right\} \tag{3.18}$$

    and set $\boldsymbol{\xi}_t^{(k)} = \tilde{\boldsymbol{\xi}}^{(k)}$, $\boldsymbol{\tau}_t^{(k,l,m)} = \tilde{\boldsymbol{\tau}}_t^{(k,l,m)}$

Output: posterior samples $\boldsymbol{\xi}_T^{(k)} \sim p(\cdot|\boldsymbol{\tau}^{\text{real}}_{1:N}),\ (k = 1, ..., K)$

In the context of SMC-ABC the Effective Sample Size (ESS)

$$\text{ESS}(\{w_t^{(k)}\}_{k=1}^K) = \left(\sum_{k=1}^K \left(w_t^{(k)}\right)^2\right)^{-1} \tag{3.19}$$

is used as a measure to compare the variability of the given weighted particles. The ESS takes values between zero and K. If all particles are evenly distributed, the ESS is K and gets smaller when the variability of the samples sinks. If the ESS is below a certain threshold, the SMC-ABC algorithm re-samples new particles to replenish the particles with non-negligible probability.

3.4.2. Relative Entropy Policy Search (REPS)

REPS [7] is an empirical inference algorithm, formally presented for policy search. Given a cost to discriminate between simulated and real data, the objective is to find the data distribution which minimizes the expected cost. For the joint distribution $q(\boldsymbol{\tau}_{1:M}, \boldsymbol{\xi}|\boldsymbol{\tau}_{1:N}^{\text{real}})$, the objective is as follows:

$$\begin{aligned} \min_{q(\boldsymbol{\tau}_{1:M},\boldsymbol{\xi})} \quad & \mathop{\mathbb{E}}_{\boldsymbol{\xi}\sim q(\boldsymbol{\xi})}\left[\mathop{\mathbb{E}}_{\boldsymbol{\tau}_{1:M}\sim p(\boldsymbol{\tau}_{1:M}|\boldsymbol{\xi})}\left[D\left(\boldsymbol{\tau}_{1:N}^{\text{real}}, \boldsymbol{\tau}_{1:M}\right)\right]\right], \\ \text{s.t.} \quad & \text{KL}\left(q(\boldsymbol{\tau}_{1:M}, \boldsymbol{\xi}) \parallel p(\boldsymbol{\tau}_{1:M}, \boldsymbol{\xi})\right) \leq \varepsilon, \\ & \iint q(\boldsymbol{\tau}_{1:M}|\boldsymbol{\xi}) p(\boldsymbol{\xi}) \, \mathrm{d}\boldsymbol{\tau}_{1:M} \, \mathrm{d}\boldsymbol{\xi} = 1. \end{aligned} \tag{3.20}$$

Note, for clarity of presentation and without loss of generality, the problem is formulated without conditioning on the real data $\boldsymbol{\tau}_{1:N}^{\text{real}}$. Following trust-region optimization approaches, the optimization of the joint distribution $q(\boldsymbol{\tau}_{1:M}, \boldsymbol{\xi})$ to the previously assumed posterior $p(\boldsymbol{\tau}_{1:M}, \boldsymbol{\xi})$ is restricted by the KL divergence between these distributions. The third term guarantees that $q(\boldsymbol{\xi})$ adheres to the family of probability distributions. The analytical solution to the optimization problem is

$$q^\star(\boldsymbol{\tau}_{1:M}, \boldsymbol{\xi}) \propto e^{-\eta^{-1} D(\boldsymbol{\tau}_{1:N}^{\text{real}}, \boldsymbol{\tau}_{1:M})} \; p(\boldsymbol{\tau}_{1:M}|\boldsymbol{\xi}) \; p(\boldsymbol{\xi}), \tag{3.21}$$

where η is a temperature parameter which can be obtained by maximizing the dual function of Eq. (3.20)

$$\max_{\eta>0} \quad g(\eta) = -\eta \left(\varepsilon + \log \mathop{\mathbb{E}}_{p(\boldsymbol{\xi})}\left[\mathop{\mathbb{E}}_{p(\boldsymbol{\tau}_{1:M}|\boldsymbol{\xi})}\left[e^{-\frac{1}{\eta} \, D(\boldsymbol{\tau}_{1:N}^{\text{real}}, \boldsymbol{\tau}_{1:M})}\right]\right]\right). \tag{3.22}$$

A derivation of the optimal domain parameter distribution $q^\star(\boldsymbol{\tau}_{1:M}, \boldsymbol{\xi})$ and the dual function using Lagrange multipliers can be found in Appendix A.1.

The optimal posterior can be recovered by marginalizing Eq. (3.21) w.r.t. the simulated trajectories $\boldsymbol{\tau}_{1:M}$

$$q^\star(\boldsymbol{\xi}|\boldsymbol{\tau}_{1:N}^{\text{real}}) \propto p(\boldsymbol{\xi}) \int e^{-\eta^{-1} D(\boldsymbol{\tau}_{1:N}^{\text{real}}, \boldsymbol{\tau}_{1:M})} \, p(\boldsymbol{\tau}_{1:M}|\boldsymbol{\xi}) \, \mathrm{d}\boldsymbol{\tau}_{1:M}. \tag{3.23}$$

From Eq. (3.23), a sequential algorithm can be derived which finds the optimal solution. In a first step, the dual Eq. (3.22) is maximized w.r.t. η. The expectation $\mathbb{E}_p(\boldsymbol{\xi})[\cdot]$ can be approximated using Monte Carlo samples. Standard non-linear optimization solvers can be applied to find η. After that, the parameter distribution can be fitted by minimizing the KL divergence between $q(\boldsymbol{\xi}|\boldsymbol{\tau}_{1:N}^{\text{real}})$ and $q^\star(\boldsymbol{\xi}|\boldsymbol{\tau}_{1:N}^{\text{real}})$. Optimizing the parameterized distribution $q_{\boldsymbol{\phi}}(\boldsymbol{\xi})$ w.r.t. the *m-projection* of the KL divergence is referred to as maximizing the weighted likelihood

$$\min_{\boldsymbol{\phi}} \quad \mathrm{KL}\left(q^\star \parallel q_{\boldsymbol{\phi}}\right) = \max_{\boldsymbol{\phi}} \quad \mathop{\mathbb{E}}_{q^\star}\left[\log q_{\boldsymbol{\phi}}(\boldsymbol{\xi})\right] \tag{3.24}$$

$$= \max_{\boldsymbol{\phi}} \quad \mathop{\mathbb{E}}_{p(\boldsymbol{\xi})}\left[\mathop{\mathbb{E}}_{p(\boldsymbol{\tau}_{1:M}|\boldsymbol{\xi})}\left[\frac{e^{-\eta^{-1} D(\boldsymbol{\tau}_{1:N}^{\text{real}}, \boldsymbol{\tau}_{1:M})}}{\mathbb{E}_{p(\boldsymbol{\xi})}\left[\mathbb{E}_{p(\boldsymbol{\tau}_{1:M}|\boldsymbol{\xi})}\left[e^{-\eta^{-1} D(\boldsymbol{\tau}_{1:N}^{\text{real}}, \boldsymbol{\tau}_{1:M})}\right]\right]} \log q_{\boldsymbol{\phi}}(\boldsymbol{\xi})\right]\right]. \tag{3.25}$$

Again, the expectations are approximated with Monte Carlo samples. REPS assumes a parameterized multivariate Gaussian posterior model while ABC relies on a set of particles to estimate the posterior. The pseudocode for REPS is shown in Algorithm 3.

3.5. Connecting Approximate Bayesian Computation and Relative Entropy Policy Search

The approximate posteriors of SMC-ABC and REPS in Sections 3.4.1 and 3.4.2 have a similar form and indeed, we will show that the two inference methods are closely related. In a first step, a generalized view on the two inference methods is established as a form of inference for Energy-Based Models (EBMs) [45]. In EBMs, parameters $\boldsymbol{z}$ and the corresponding data samples $\boldsymbol{x}$ are available from which the likelihood can be approximated using a Gibbs distribution

$$\tilde{p}(\boldsymbol{x}|\boldsymbol{z}) = e^{-\frac{1}{\beta} E(\boldsymbol{x}, \boldsymbol{z})}. \tag{3.29}$$

Algorithm 3: Pseudo code for episodic REPS[7] using the m-projection

Input: parameterized model $q_\phi(\boldsymbol{\xi})$, prior $p_0(\boldsymbol{\xi})$, generative model $g(\boldsymbol{\xi})$, cost function $D(\boldsymbol{\tau}_{1:N},\ \boldsymbol{\tau}_{1:M})$, number of samples K, number of samples per domain parameter M, N real world samples $\boldsymbol{\tau}^{\text{real}}_{1:N}$, number of steps T, acceptance threshold ε

Input: $t = 0$, Set prior as sampling distribution $p(\boldsymbol{\xi}) = p_0(\boldsymbol{\xi})$

for $t = 1, ..., T$ **do**

Sample parameters $\boldsymbol{\xi}_t^{(k)} \sim p(\boldsymbol{\xi}_k),\ (k = 1, ...K)$

Generate simulations $\boldsymbol{\tau}_t^{(k,l,m)} \sim g(\boldsymbol{\xi}_t^{(k)}),\ (l = 1, ..., L),\ (m = 1, ..., M)$

Calculate weights

$$w_k = \sum_{l=1}^{L} e^{-\eta^{-1}\ D\left(\boldsymbol{\tau}^{\text{real}}_{1:N},\ \boldsymbol{\tau}_t^{(k,l,1:M)}\right)} \tag{3.26}$$

Optimize η using non-linear optimization solvers (e.g. Newton CG)

$$\min_{\eta} \quad -g(\eta) \approx \eta\varepsilon + \eta \log\left(\frac{1}{K}\sum_{k=1}^{K} w_k\right) \tag{3.27}$$

Fit parameter distribution by maximizing the weighted log-likelihood

$$\max_{\phi} \quad \frac{\sum_{k=1}^{K} w_k \log q_\phi(\boldsymbol{\xi}^{(k)})}{\sum_{k=1}^{K} w_k} \tag{3.28}$$

Output: approximate posterior $q_\phi(\boldsymbol{\xi})$

Here β serves as a temperature parameter. The likelihood yields large probabilities if the energy functional $E(\boldsymbol{x}, \boldsymbol{z})$ is low and vice versa. Applying Bayes rule on the ABC posterior (Eq. (3.14)), the approximate likelihood reads

$$p(\boldsymbol{\tau}^{\text{real}}_{1:N}|\boldsymbol{\xi}) = \int K_\varepsilon\left(D(\boldsymbol{\tau}^{\text{real}}_{1:N}, \boldsymbol{\tau}_{1:M})\right) \prod_{m=1}^{M} p(\boldsymbol{\tau}_m|\boldsymbol{\xi})\ \mathrm{d}\boldsymbol{\tau}_m. \tag{3.30}$$

The likelihood $p(\boldsymbol{\tau}^{\text{real}}_{1:N}|\boldsymbol{\xi})$ of the real data can be approximated using Monte Carlo sampling from $p(\boldsymbol{\tau}_m|\boldsymbol{\xi})$ via the simulator $g(\boldsymbol{\xi})$. Assuming an RBF kernel (see Table 3.1), a closer look reveals the similarities between REPS, ABC and EBMs. For the RBF kernel,

$K_\varepsilon\left(D\left(\boldsymbol{\tau}_{1:N}^{\text{real}}, \boldsymbol{\tau}_{1:M}\right)\right)$ with standard deviation $\varepsilon = (\eta/2)^{-1/2}$, the approximate posteriors of SMC-ABC and REPS are the same and can be understood as being the posterior estimated from an energy based likelihood. Thus, the posterior is constructed from a sampling distribution and a likelihood which scores the drawn samples based on their distance to the ground truth.

A successful application of any ABC or REPS implementation relies on (*i*) a 'good' representation of the data via summary statistics and a 'reliable' distance measure and (*ii*), adapting the sampling distributions in a 'reasonable' way. The former point is addressed in various conducted research on summary statistics (see, e.g., [46, 42]). We will show in the next paragraphs how SMC-ABC and REPS address the sequential adaptation of the sampling distribution.

In SMC-ABC, the particle updates follow by sampling candidate samples from the previous particle distribution and accepting the candidate parameters with the acceptance probability shown in Eq. (3.18). Large kernel parameter values ε yield broader kernels K_ε which means that samples from a broad range get assigned equally likely probabilities. Therefore, the acceptance probabilities are higher for large ε for a broad range of parameters. The contrary is the case for small kernel parameters. Therefore, the choice of the hyperparameter α (see Eq. (3.16)) decides whether the sampling distribution adapts quickly or slowly.

REPS applies a trust-region via the KL divergence constraint to ensure that the update of the parameterized distribution is bounded by a threshold. The constraint on the distribution update is motivated to minimize the information loss[47, 7] while optimizing the main objective. In Eq.(3.23), the regularization constraint is represented by the exponential kernel. As can be seen in Eq. (3.25), the kernel weights the domain parameter samples $\boldsymbol{\xi}$. Thus, large sample weights move the optimization routine towards these samples. Similar to SMC-ABC, the kernel parameters decide how far the distribution can move away from the current distribution.

To complete the comparison of REPS and SMC-ABC, the kernel parameters η and ε are considered. In SMC-ABC, the kernel parameter ε is estimated from the ESS. As a measure for the variability of the particles, estimating ε from Eq.(3.16) finds the smallest kernel parameter ε which maintains a certain particle variability. In REPS, the kernel parameter η is obtained by maximizing the Dual (3.22). In practice, the Lagrange multiplier η and the rejection threshold ε are constantly updated within their respective inference routine. Figure 3.4 shows that the kernel parameters are small in the beginning and exponentially decrease to an equilibrium state. The kernels, depicted in the three remaining plots, show

that the kernels become less diffusive with every iteration. This means that the more diffusive the posterior is, the less confident the kernel is and vice versa.

In summary, we have shown that REPS and SMC-ABC approximate the intractable likelihood using a distance-based Gibbs distribution. Furthermore, both algorithms have a built-in regularization of the posterior update based on a kernel that represents the certainty with which samples and ground truth observations belong to each other. In experiments, the kernel bandwidths of REPS and SMC-ABC show similar behavior.

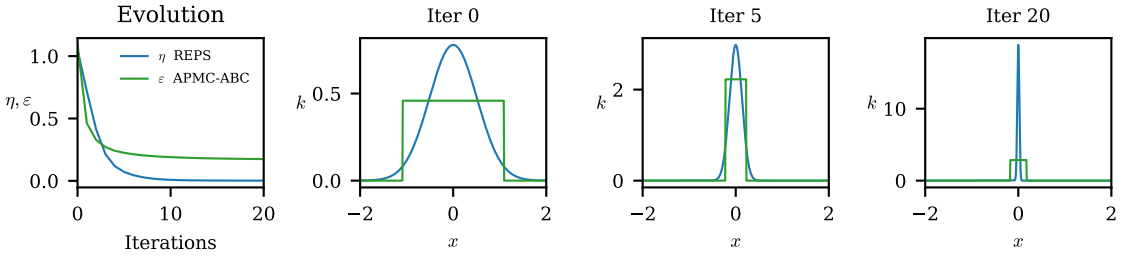


Figure 3.4.: Kernel parameters ε and η measured over one run of REPS and APMC-ABC on the Furuta pendulum. On the left, the exponential decline of the parameters is plotted over the algorithms iterations. The three remaining plots show how the kernels, associated to η and ε, develop. x can be interpreted as a distance representation between the reference and simulated data. The y axis depicts the kernel values, which represent an unnormalized probability. In APMC-ABC, a uniform kernel is used while REPS applies an RBF kernel.

3.6. Wasserstein-Optimal Bayesian System Identification

In this section, we introduce Wasserstein-optimal Bayesian system identification, an approach which combines the Wasserstein distance between trajectories with the LFI approaches SMC-ABC and REPS. First, the Wasserstein distance between two trajectory distributions is worked out. Then the Bayesian inference approaches SMC-ABC and REPS are combined with the Wasserstein distance.

Assume that we have collected trajectories from the real system $\{\boldsymbol{\tau}_i^{\text{real}}\}_{i=1}^N$. Given a behavioral policy, the goal is to infer the domain parameters which are responsible for the observed data. Therefore, data is collected in simulation $\{\boldsymbol{\tau}_j^{\boldsymbol{\xi}}\}_{j=1}^M$ to compare it to the

observed data. Let us assign an empirical probability measure for each dataset

$$p^{\text{real}}(\boldsymbol{\tau}) = \frac{1}{N} \sum_{i=1}^{N} \delta_{\boldsymbol{\tau}_i^{\text{real}}}(\boldsymbol{\tau}), \tag{3.31}$$

$$p_{\boldsymbol{\xi}}(\boldsymbol{\tau}) = \frac{1}{M} \sum_{j=1}^{M} \delta_{\boldsymbol{\tau}_j^{\boldsymbol{\xi}}}(\boldsymbol{\tau}); \quad \boldsymbol{\tau}_j^{\boldsymbol{\xi}} \sim p(\cdot|\boldsymbol{\xi}), \tag{3.32}$$

where each sample has the same probability. To calculate the statistical distance between the two probability measures, the Discrete Optimal Transport Problem Eq. (3.7) is solved. The cost matrix is the distance between the trajectories $\boldsymbol{\tau}_i^{\text{real}}$ and $\boldsymbol{\tau}_j^{\boldsymbol{\xi}}$

$$\boldsymbol{C}_{i,j} = \rho(\boldsymbol{\tau}_i^{\text{real}}, \boldsymbol{\tau}_j^{\boldsymbol{\xi}}). \tag{3.33}$$

Trajectories are represented as a set of vectors which contain the states and actions at each time step $\{(\boldsymbol{s}_t^T, \boldsymbol{a}_t^T)^T\}_{t=0}^{T-1}$. ρ can be any distance measure between two time series, but will be the DTW discrepancy or the MSE throughout this work. Note that the MSE is a special case of DTW assuming the bandwidth $b = 0$. Given the cost matrix, the optimal transportation cost can be calculated using the numerical methods described in Section 3.3.1.

$$D(\boldsymbol{\tau}_{1:N}^{\text{real}}, \boldsymbol{\tau}_{1:M}^{\boldsymbol{\xi}}) = \sum_{i,j} \boldsymbol{P}_{i,j}^* \boldsymbol{C}_{i,j}, \tag{3.34}$$

$$\boldsymbol{\tau}_{1:N}^{\text{real}} \sim p^{\text{real}}(\boldsymbol{\tau}), \quad \boldsymbol{\tau}_{1:M}^{\boldsymbol{\xi}} \sim p_{\boldsymbol{\xi}}(\boldsymbol{\tau}).$$

In the previous sections, the inference routines SMC-ABC and REPS have been formulated in a way such that the Wasserstein distance can directly be leveraged as the distance function for SMC-ABC and REPS. For Bayesian system identification, we choose REPS with m-projection updates of the posterior and a slightly improved SMC-ABC algorithm, namely (Adaptive Population Monte Carlo) APMC-ABC [21]. Note that in this thesis, the expectations over trajectories in Eq. (3.14) and Eq. (3.25) are estimated by a single sample. The validity of this approximation is left for future work.

4. Experiments

The proposed method, introduced in Section 3.6, is evaluated on three experiments. First, a toy experiment serves as a validation for the proposed method. Afterwards, Bayesian system identification is carried out for the swing-up tasks on the cart-pole and Furuta pendulum.

4.1. Environments

This section dives deeper into the specifics of each environment. For that, the equations of motion are outlined which can be derived by formulating the Euler-Lagrange equation for each environment and assuming rigid-body dynamics for the cart-pole and Furuta pendulum.

4.1.1. Damped Harmonic Oscillator

In this experiment, a point mass is attached to a spring which oscillates along the x-Axis around the resting position $x = 0$. The other end of the spring is attached to a fixed point which is assumed to be not moving. Furthermore, we assume that the system is damped. The ordinary differential equation for the damped harmonic oscillator is

$$f = m\ddot{x} + c\dot{x} + kx. \tag{4.1}$$

Here f denotes the external force applied to the point mass, m denotes the mass, c denotes the friction coefficient and k denotes the spring stiffness. In the experiments, the external force is set to zero, $f = 0$, which yields the analytical solution of the underdamped system

$$x(t) = x_0 \, e^{-\lambda t} \, \cos(\omega t + \phi). \tag{4.2}$$

The natural frequency ω and the damping ratio λ of Eq. (4.1) are characteristic constants to describe the system's behavior

$$\omega_0 = \sqrt{\frac{k}{m}}, \quad \lambda = \frac{c}{2m}, \quad \omega = \sqrt{\omega_0^2 - \lambda^2}. \tag{4.3}$$

The system oscillates around the equilibrium position $\boldsymbol{x}(t) \xrightarrow{t\to\infty} 0$ with the natural frequency ω. Assume that $\phi = 0$ and no initial velocity of the point mass, then the initial amplitude of the system is x_0 and the system slowly decays exponentially, based on the damping ratio λ, to its resting position.

4.1.2. Cart-Pole

The swing-up and stabilization task on the cart-pole is one of the common baselines to evaluate RL approaches [48]. A rod is mounted on an angular joint which is positioned on a cart which can drive horizontally along the x-direction. The swing-up can be achieved by driving the cart in both direction and thereby generating the kinetic energy needed for the rod to swing up. The generalized coordinates are the joint angle of the rod θ and the position of the cart x w.r.t. to its equilibrium position. The equations of motion for the cart-pole are given by

$$\begin{bmatrix} -d_c\dot{x} + f_c \\ -d_p\dot{\theta} \end{bmatrix} = \begin{bmatrix} m_c + m_p + I_{dc} & m_p l_p \cos\theta \\ m_p l_p \cos\theta & \frac{4}{3} m_p l_p^2 \end{bmatrix} \begin{bmatrix} \ddot{x} \\ \ddot{\theta} \end{bmatrix} + \begin{bmatrix} m_p l_p \sin\theta\, \dot{\theta}^2 \\ m_p l_p g \sin\theta \end{bmatrix}. \tag{4.4}$$

The left-hand side is the change of momentum due to the joint friction, defined by the damping coefficients d_c and d_p as well as the external force along the rails f_c applied by the servo motor. The servo motor can be actuated applying voltage in the interval $[V_{\text{min}}, V_{\text{max}}]$. The mass matrix contains the terms for the longitudinal and angular momentum of the cart and the pole as well as a term for the angular momentum of the DC motor I_{dc}. The non-linear terms are due to the Coriolis forces and the gravitation working on the rod.

4.1.3. Furuta Pendulum

The underactuated Furuta Pendulum is popular in control for its fast changing dynamics. In a Furuta pendulum, the end of the 'rotational' pole is attached on a rotational joint with a vertical rotation axis. The other end serves as the pendulum link, where the end of

a second pole is attached which can rotate around the longitudinal axis of the rotational rod. The system can be described by the angular deflection $[\theta_r, \theta_p]$ of the rods w.r.t. their equilibrium position $[0, 0]$. The equations of motion can be derived by formulating the Euler-Lagrange equation similar to [49]

$$\begin{bmatrix} -d_r\dot{\theta}_r + \tau \\ -d_p\dot{\theta}_p \end{bmatrix} = \begin{bmatrix} \frac{1}{12}m_r l_r^2 + m_p l_r^2 + \frac{1}{4}m_p l_p^2 \sin^2\theta_p & \frac{1}{2}m_p l_p l_r \cos\theta_p \\ \frac{1}{2}m_p l_p l_r \cos\theta_p & \frac{1}{3}m_p l_p^2 \end{bmatrix} \begin{bmatrix} \ddot{\theta}_r \\ \ddot{\theta}_p \end{bmatrix} + \begin{bmatrix} \frac{1}{4}m_p l_p^2 \sin 2\theta_p \, \dot{\theta}_r\dot{\theta}_p - \frac{1}{2}m_p l_p l_r \sin\theta_p \, \dot{\theta}_p^2 \\ -\frac{1}{8}m_p l_p^2 \sin 2\theta_p \, \dot{\theta}_r^2 + \frac{1}{2}m_p l_p g \sin\theta_p \end{bmatrix}. \tag{4.5}$$

Here, the assumption is made that the pole length is significantly greater than its diameter for which the moments of inertia of the poles around their pivot are $J_i = 1/3 \; m_i l_i^2, \; i \in \{r, p\}$. The mass matrix contains entries from the translatoric and rotational movement of the two poles. As the reference coordinate systems are constantly rotating w.r.t. the basis coordinate system, Coriolis forces dominate the non-linear term. They are complemented by the gravitation which works on the rotational pole. The left-hand side considers damping in the joints, represented by the damping coefficients d_r and d_p, and the torque τ which is applied from a servo motor

$$\tau = -\frac{k_m^2}{r_m}\dot{\theta}_p + \frac{k_m}{R_m}u, \tag{4.6}$$

where k_m is the motor resistance and r_m is the motor back-electromotive force. Here u is the applied voltage to control the robot.

4.2. Behavior Policies

A behavior policy is needed to complement the simulator for Bayesian system identification (see Figure 4.1).

The damped harmonic oscillator is not subject to any external influences and hence, the behavioral policy is an idle policy with zero return.

For the experiments, energy-based controllers are leveraged to solve the swing-up task on the given environment. Depending on its current state, the swing-up controller either

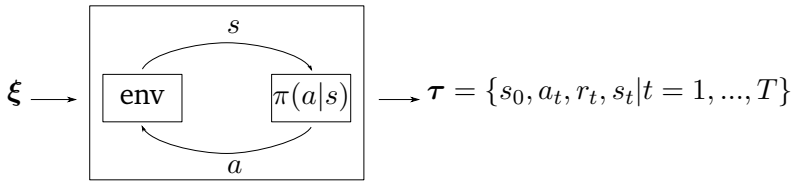


Figure 4.1.: Simulator used for Bayesian system identification. The domain parameters define the system dynamics. The simulator is an agent which chooses actions based on the policy $\pi(\boldsymbol{a}|\boldsymbol{s})$ within the environment and thereby creates a trajectory $\boldsymbol{\tau}$.

chooses its next action based on an energy generating controller or uses a PD controller to stabilize the swing-up. The deflection angle of the pendulum θ_p decides which controller to use. The decision boundary for the experiments are $|\pi - \theta_p| \leq 20\pi/360$. The energy controller generates the systems kinetic energy and compares it to a reference energy, which is a representation of the minimal energy required to do the swing-up. The missing kinetic energy will be provided by the controller.

4.3. Software Setup

The experiments are carried out within "SimuRLacra — a framework for reinforcement learning from randomized simulations" [50]. For the system identification baseline SNPE-C [25], the "sbi-toolbox" [51] is used which offers a variety of implementations for LFI algorithms. Finally, the python library "geomloss" [52] is used to calculate the Wasserstein distance between the two sets of trajectories.

5. Results

Wasserstein-optimal Bayesian system identification is carried out on the cart-pole and the Furuta pendulum. First, the proposed method is validated on sim-to-sim experiments to show that the approach converges towards the ground truth parameters. Afterwards, experiments using real world data are conducted on the swing-up tasks of the cart-pole and the Furuta pendulum. The experimental details are defined in Section 4. During all the experiments, we compare the Wasserstein-optimal LFI approaches, APMC-ABC and REPS, with the baseline SNPE-C.

5.1. Evaluation of Inference Methods

In sim-to-sim experiments, true domain parameters are defined from which N reference trajectories $\boldsymbol{\tau}_{1:N}^{\text{real}}$ are generated. Validation is carried out by finding the true domain parameters responsible for these reference rollouts. The sim-to-sim experiments are conducted on the damped harmonic oscillator (see Section 4.1.1) and the Furuta pendulum (see Section 4.1.3). The experimental specifications are shown in Tables A.1 and A.2. The sim-to-sim experiments are carried out for one reference rollout and are using the MSE between trajectories to compare the data. The evolution of the approximate posterior and the simulated rollouts of REPS on the damped harmonic oscillator are presented in Figure 5.1.

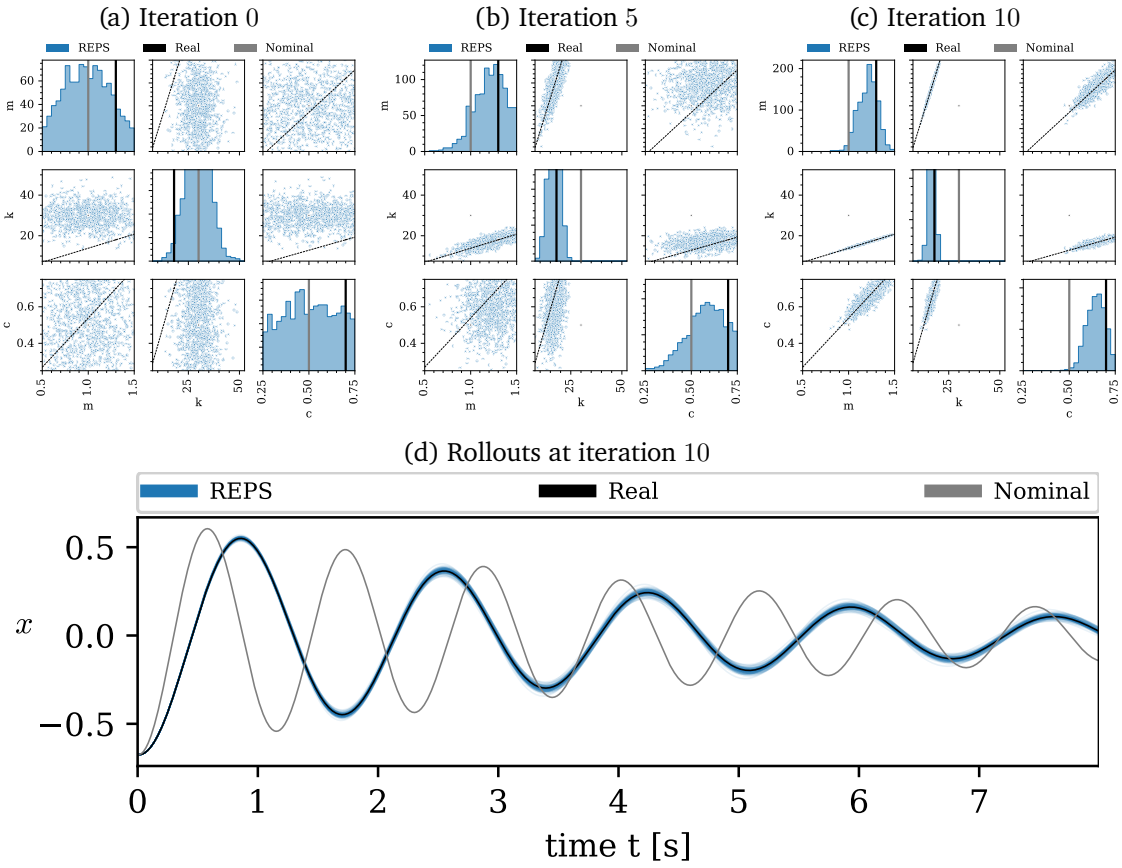


Figure 5.1.: Posterior evolution of REPS on the damped harmonic oscillator. The pair-plots (a)–(c) are plotted from 1000 samples and present the adaptation of the approximate posterior to the real data. REPS captures the pairwise linear correlations between the domain parameters highlighted in black dotted lines. 100 trajectories of the position of the mass x are plotted in (d) and show that the physical behavior is accurately presented.

The mode of the posterior moves from the prior, centered around the nominal parameters, towards the real parameters. Furthermore, the posterior samples show pairwise linear correlations between all domain parameters. In this experiment, the initial deflection and velocity is fixed such that the trajectories only deviate in the damping ratio λ and the natural frequency ω. These variables are completely described by the domain parameters (see Eq. (4.3)). Typically, ω_0 is a good approximation for ω and indeed, they differ

by $0.26\,\%$. Hence, ω_0 is modeled as the natural frequency from which the correlations between the parameters are

$$\omega_0^{\text{real}} = \frac{k}{m}, \qquad 2\lambda^{\text{real}} = \frac{c}{k}, \qquad \frac{2\lambda^{\text{real}}}{\omega_0^{\text{real}}} = \frac{c}{k}.$$

The linear correlations are highlighted in the plots by black dotted lines. The pair-plot shows that the approximate posterior can capture the underlying linear correlations. The trajectory plot of the inferred parameters in Figure 5.1d confirms that the found correlations reproduce the reference data. As can be seen, all simulations of the sampled parameter configurations reproduce the observed trajectories. APMC-ABC and the baseline SNPE-A reveal similar results and are therefore not further mentioned. Though, for completeness their results are shown in Table A.1.

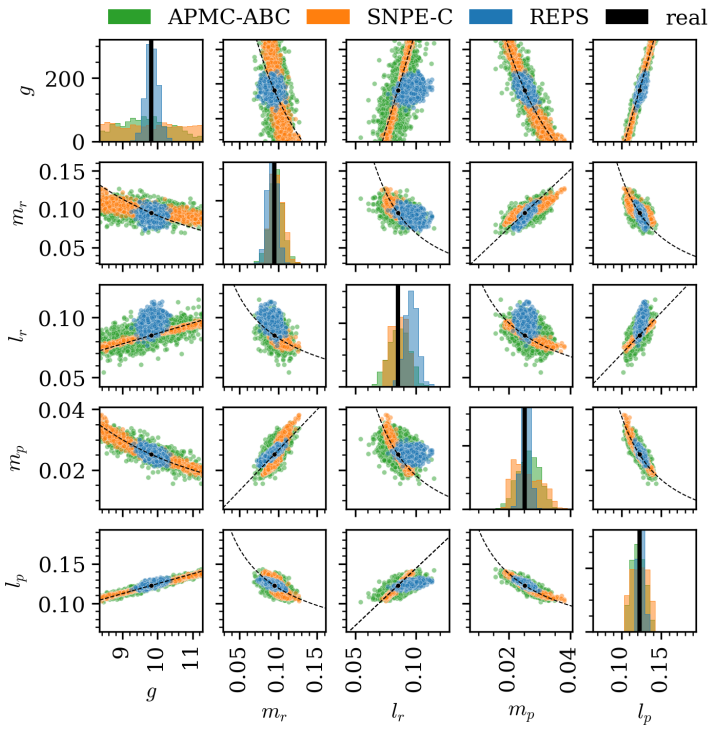


Figure 5.2.: Approximate posteriors of REPS, APMC-ABC and SNPE-C on the Furuta pendulum in a sim-to-sim experiment. The diagonal axis depicts the marginal histograms while the remaining subplots show scatter plots between two parameters. Each experiment is evaluated on 1000 samples. The domain parameter correlations, which yield the reference trajectory, are presented in black dotted lines.

The domain parameters for the Furuta pendulum are the gravitational constant g and the mass and length of the rotational rod m_r, l_r and of the pendulum rod m_p, l_p. Figure 5.2 presents the pair-plot of the trained approximate posteriors of the three inference methods

REPS, APMC-ABC and SNPE-C. The marginal histogram plots show that all inference methods are centered around the real domain parameters. It can be seen that SNPE-C has the least diffusive posterior while REPS has the most diffusive one. Next, we examine the scatter plots by comparing against the correlations plotted in black dotted lines. These correlations are the domain parameter configurations which yield the same reference rollout when fed through the simulator. A derivation of the correlations can be found in Appendix A.2. As shown in the previous experiment, the linear correlations are captured by all inference routines. The Furuta pendulum additionally features non-linear correlations, e.g., between m_r, m_p and l_r, l_p. REPS approximates the quadratic correlation with an elliptic point cloud, APMC-ABC shows more diffuse scatter plots when quadratic correlations are present while SNPE-C finds the correlations over a wide range of the prior region. This observation might be explained by the different models each algorithm assumes. Normalizing flows, leveraged by SNPE-C, are known to be flexible [13] while the Gaussian model, utilized by REPS, restricts the expressiveness of the density estimator.

5.2. Evaluation of Data Generation Processes on the Real System

In this section, an ablation study is carried out to investigate the influence of the data generation process for the proposed approaches. We will consider interfering the data generation process in following ways: (*i*) using recorded actions which will be replayed from the actions applied on the real system, (*ii*) partitioning the rollouts into five segments and resetting the rollouts to the states of the reference rollouts, and (*iii*) adding stochasticity to the environment by assuming Gaussian noise on the states

$$\tilde{\boldsymbol{s}} = \boldsymbol{s} + \boldsymbol{\varepsilon}, \quad \boldsymbol{\varepsilon} \sim \mathcal{N}(0, \text{diag}(\boldsymbol{\sigma}^2)).$$

We will refer to 'vanilla' rollout generation as not applying any of the aforementioned interferences. Experiments using APMC-ABC, REPS and the baseline SNPE-C are conducted on the Furuta pendulum and the cart pole using recorded data of the real system. The algorithmic specifications for each algorithm can be found in Tables A.2 and A.3 and the randomized domain parameters are shown in Table A.4.

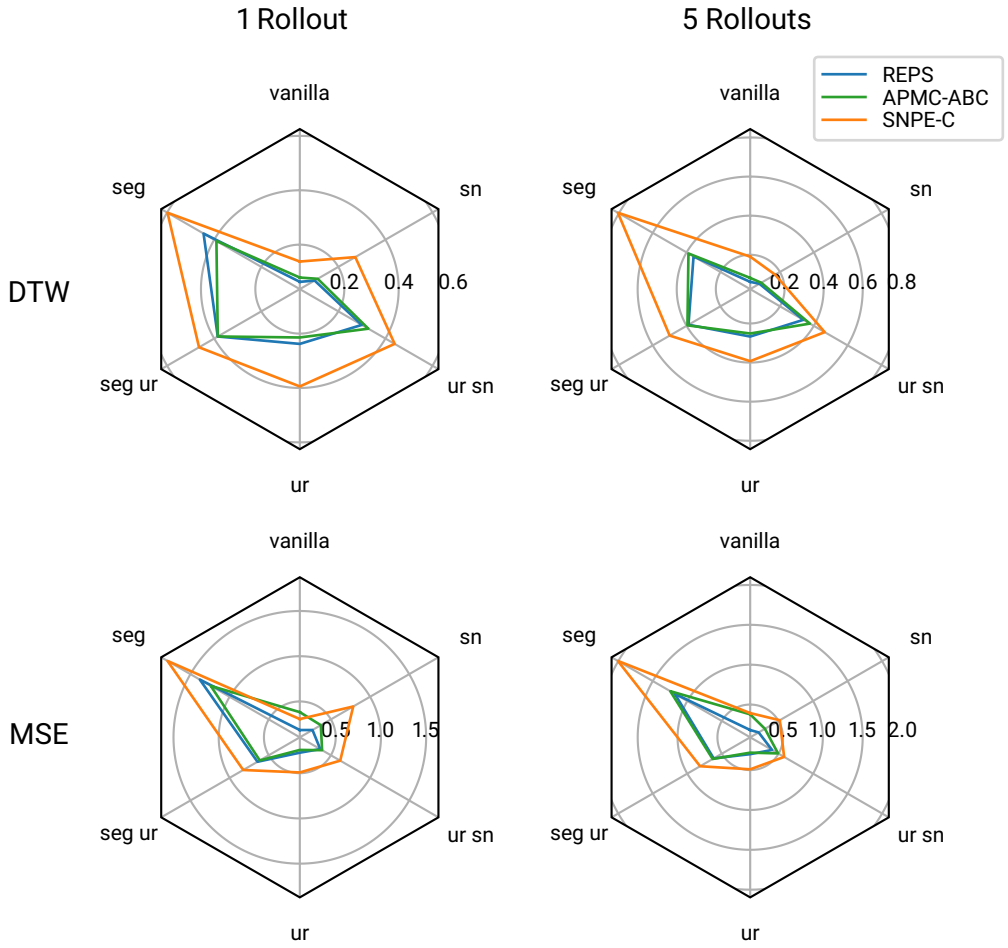


Figure 5.3.: Radar charts of the ablation study of the inference approaches using different trajectory representations on the Furuta pendulum. We report the mean DTW discrepancy and the mean MSE between the reference trajectories and the simulations generated from 100 domain parameters of the trained posterior. For five reference rollouts, five simulated rollouts are generated from a single domain parameter and compared via the Wasserstein distance. For each configuration 12 experiments have been conducted and trained until convergence (20 iterations); **rr**: number of reference rollouts; **sn**: state noise is applied; **ur**: the same actions are used on the physical and simulated system, **seg**: the rollouts are segmented into five trajectories of equal length and are reset to the states of the real trajectory; **vanilla**: neither of the described modifications is applied.

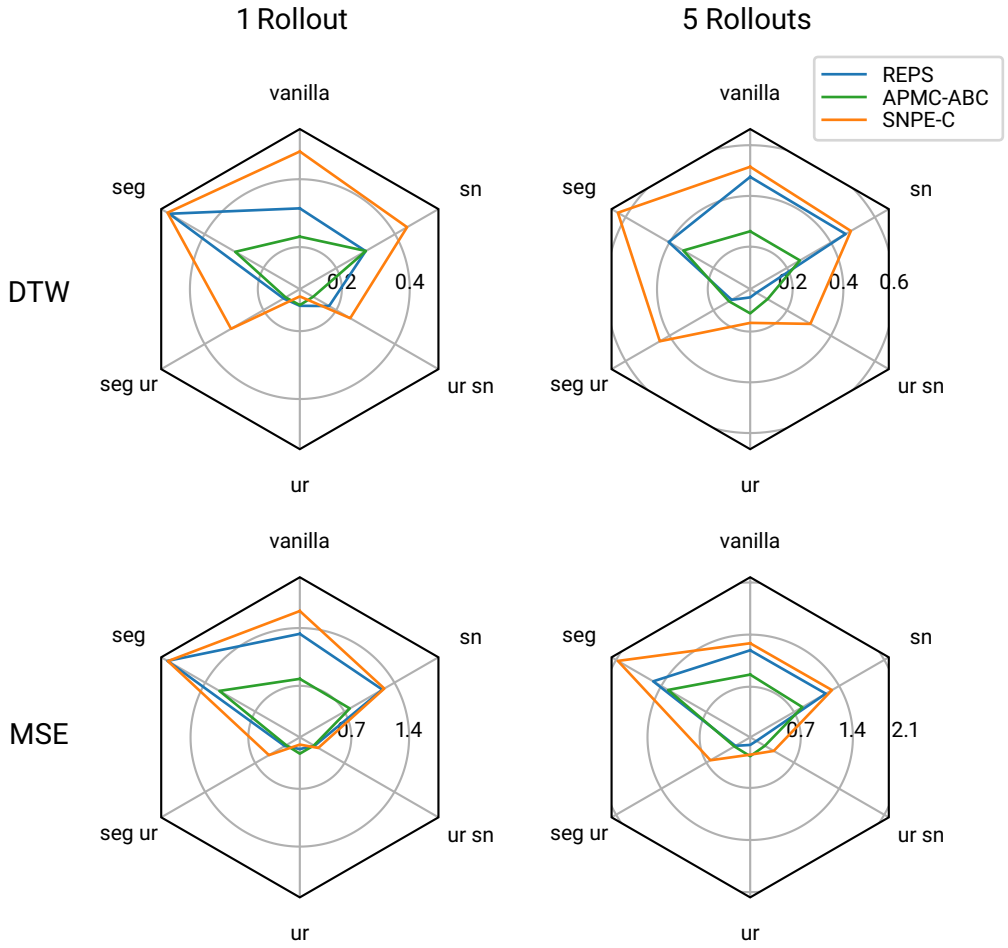


Figure 5.4.: Radar charts of the ablation study of the inference approaches using different trajectory representations on the cart-pole. We report the mean DTW discrepancy and the mean MSE between the reference trajectories and the simulations generated from 100 domain parameters of the trained posterior. For five reference rollouts, five simulated rollouts are generated from a single domain parameter and compared via the Wasserstein distance. For each configuration 12 experiments have been conducted and trained until convergence (20 iterations); **rr**: number of reference rollouts; **sn**: state noise is applied; **ur**: the same actions are used on the physical and simulated system, **seg**: the rollouts are segmented into five trajectories of equal length and are reset to the states of the real trajectory; **vanilla**: neither of the described modifications is applied.

Throughout the ablation study, Figures 5.3 and 5.4 present the mean performance of the inference routines for the different data generation processes on the Furuta pendulum and on the cart-pole. Furthermore, the radar charts are quantified by Tables A.5 and A.6 reporting the mean and standard deviation over 12 experiments. The ablation study is carried out using either one or five reference rollouts and is evaluated using the DTW discrepancy with infinite bandwidth b and the MSE between the reference data and the simulated data. Note that in all conducted experiments, the trajectories obtained from the inference methods yield better results than the nominal system parameters w.r.t. applied distance metric.

It can be seen that all algorithms adapt similarly to changes of the data generation processes. This observation indicates that the choice of the data representation is an important part of a successful LFI application. The mean performance on one and five reference rollouts is comparable. Though, the standard deviation of experiments conducted on five reference rollouts is in general smaller which is to be expected as the inference routine has to generalize on multiple rollouts. Overall, the two Wasserstein-optimal LFI methods perform best over all experimental configurations and metrics. Furthermore, the reference Tables A.5 and A.6 reveal that APMC-ABC and REPS are more robust over the series of conducted experiments. Note, while Wasserstein-optimal LFI is specifically trained to minimize the discrepancy between simulated and real trajectories, SNPE-C approximates the whole posterior and is conditioned on the real data only at the end.

5.2.1. Rollout Comparison

Rollout plots of the Furuta pendulum in Figure 5.5 and on the cart-pole in Figure 5.6 give further insight into the data representations. A single state is plotted for all six considered data generation processes. On the Furuta pendulum, we show the results of APMC-ABC for the pendulum angle θ_p. It can be seen that 'vanilla' data generation on the deterministic and noisy environment yield more diffuse rollouts compared to applying recorded actions. The plots suggest that the 'recorded actions' data fit the real data better, but this is indeed not the case, as the rotational angle θ_r (not shown) is not well presented. Therefore, the overall performance of the recorded actions is worse than 'vanilla' data generation.

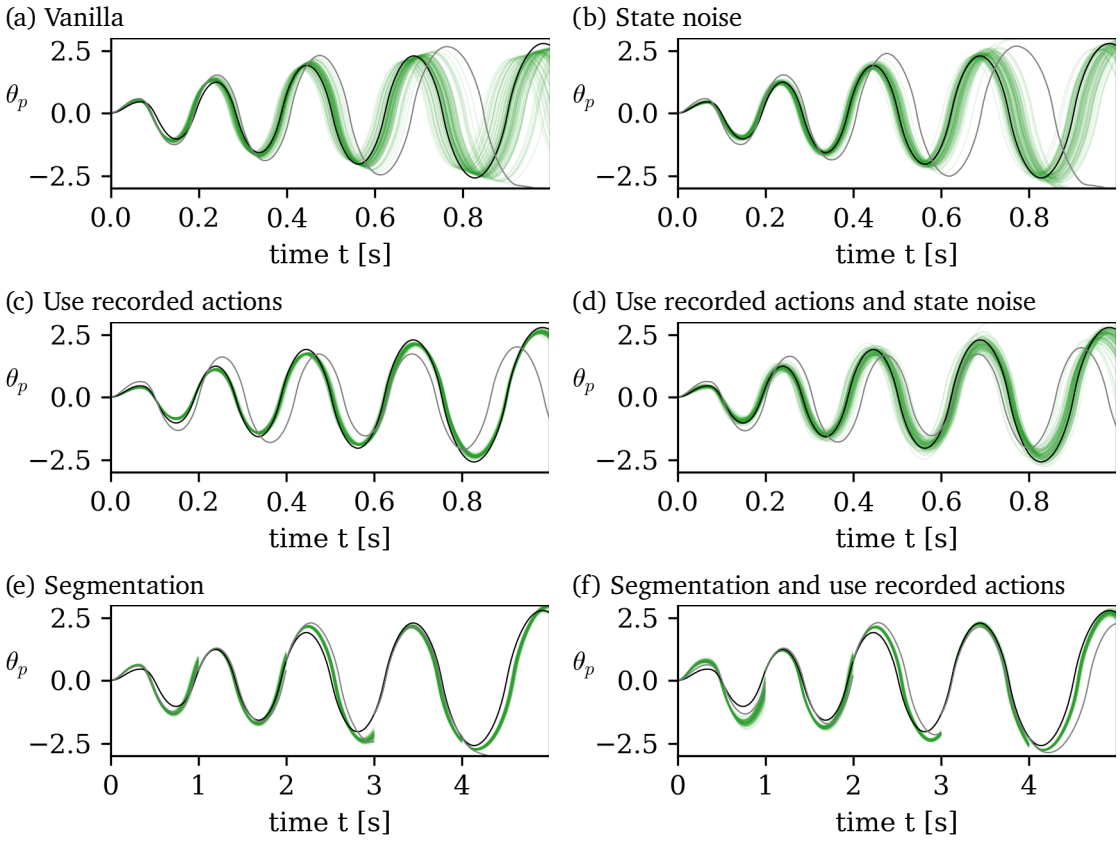


Figure 5.5.: 100 rollouts of APMC-ABC (green) on the Furuta pendulum. Here, the pendulum's angular change θ_p is plotted and compared with the reference rollout (black) and a nominal rollout (grey). The plots show the interferences into the data generation process by, (*i*) resetting the states after a certain segment length, (*ii*) applying recorded actions, or (*iii*) adding state noise to the environment.

The rollouts obtained from REPS on the cart-pole are shown in Figure 5.6 by means of the cart velocity $\dot{x}$. On the cart pole, the 'vanilla' rollouts suggests that the policy can not replicate the trajectory of the real system, but applying recorded actions yields trajectories close to the real data. These observations are further supported by the radar charts (Figures 5.3 and 5.4). On the Furuta pendulum, 'vanilla' rollout generation minimizes both metrics for one and five reference rollouts while using recorded actions performs best on the cart-pole. Interfering further into the data generation process by segmenting

the rollouts does not improve the performance, but rather yields larger metric values.

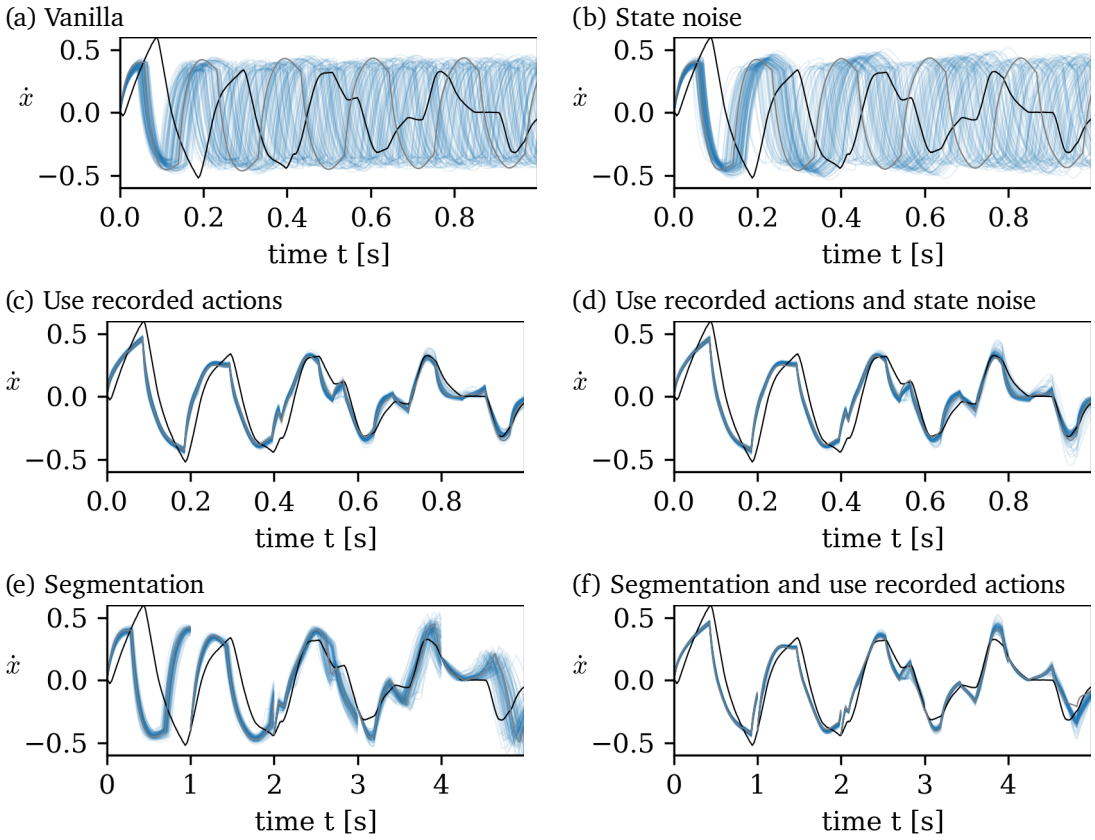


Figure 5.6.: 100 rollouts of REPS (blue) on the cart-pole. Here, the velocity of the cart $\dot{x}$ is plotted and compared with the reference rollout (black) and the nominal rollout (grey). The plots show the interferences into the data generation process by, (*i*) resetting the states after a certain segment length, (*ii*) applying recorded actions, or (*iii*) adding state noise to the environment.

5.2.2. Posterior Comparison

Next, the posterior samples obtained from the inference routines are compared with each other. Figure 5.7 shows the posteriors of APMC-ABC, REPS and SNPE-C trained on

five reference rollouts. Here, we only consider experiments trained with 'vanilla' data generation as it performed best on the Furuta pendulum. It is noticeable that the algorithms do not converge to a unique solution. While REPS is tightly centered, APMC-ABC is more diffuse and SNPE-C captures tight correlations over the whole prior region. As the sim-to-sim experiments have revealed the correlations between the domain parameters, these findings suggest that REPS concentrates around a single domain parameter configuration to reproduce the rollouts while Adaptive Population Monte Carlo (APMC)-ABC finds a larger solution space. In addition, the performance differences between SNPE-C and the Wasserstein-optimal LFI approaches might be explained by the visual shift of the domain parameters.

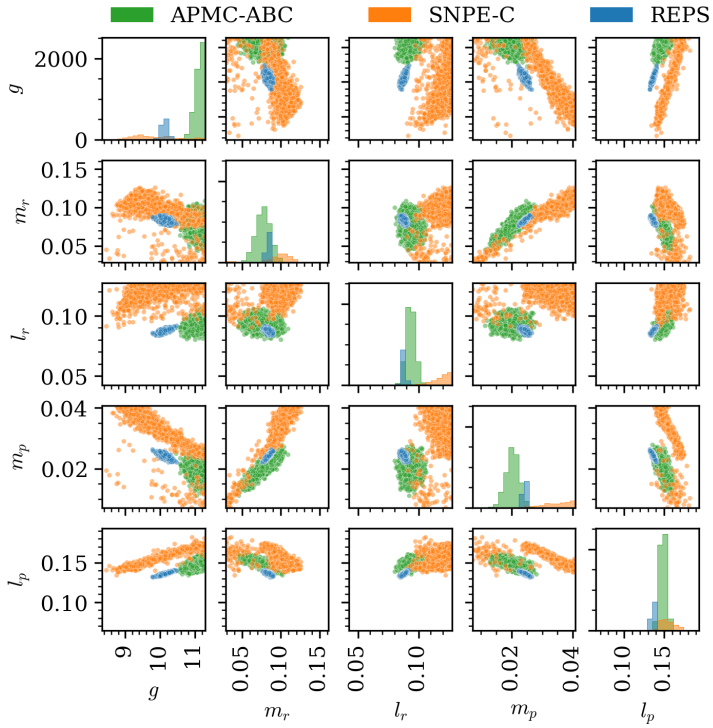


Figure 5.7.: Posterior of REPS, APMC-ABC and SNPE-C on the Furuta pendulum. The posterior, trained on five rollouts, which are generated without any interference into the data generating process, is displayed. For each distribution 1000 samples are drawn from the approximate posterior.

A similar behavior can be seen on the cart-pole which is presented in Figure 5.8. Here, the posteriors trained with recorded actions are shown as they perform best on the specified metrics. Again, the posterior samples cluster in different regions. While REPS was tightly clustered on the Furuta pendulum, REPS is spread over the whole prior region for the parameters m_c, m_p, V_{min} and V_{max}, suggesting that the domain parameters do not influence the inference procedure. The damping coefficient d_c and the length l_p are

similarly estimated by all inference routines.

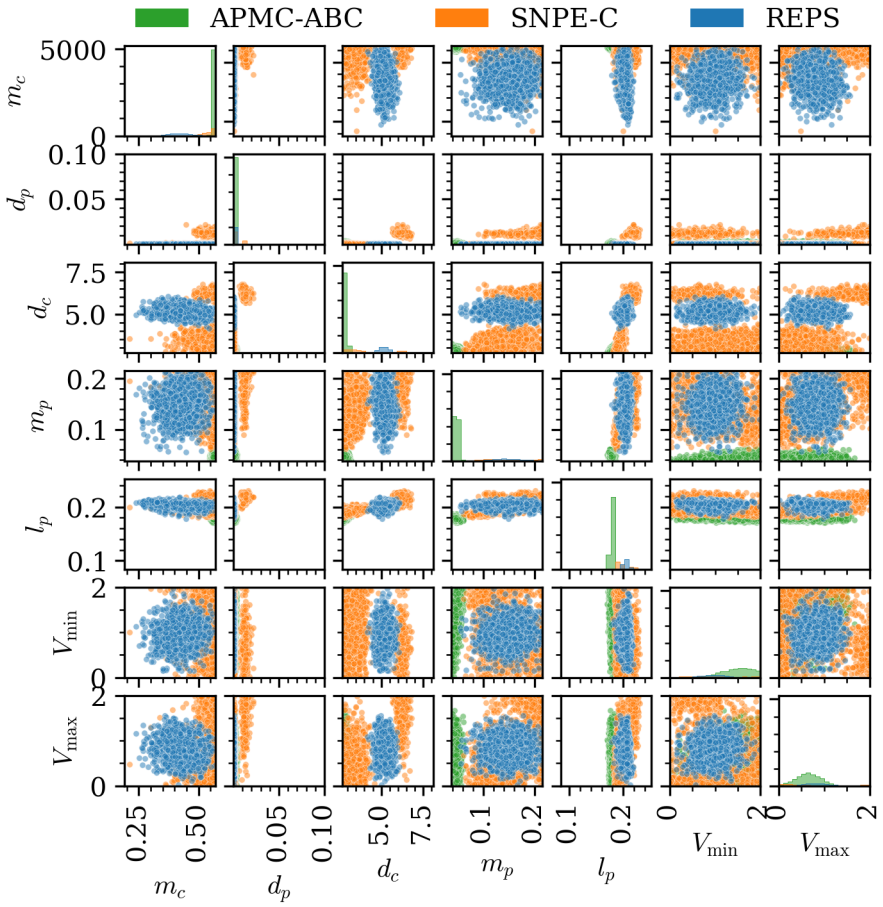


Figure 5.8.: Posterior of REPS, APMC-ABC and SNPE-C on the cart-pole.

Finally, the posterior samples of APMC-ABC are compared between three different data generation processes on the Furuta pendulum in Figure A.2. It can be seen that different data generation processes find their own clusters, but they seem to cluster along straights, see e.g., m_r, l_p. The according posterior plot of REPS on the cart-pole is shown in Figure A.3. Compared to the other data generation processes, segmenting the rollouts lead to more tight posteriors. Here, it is interesting to note that l_p which prior was uniquely estimated between the different inference methods, is widely spread for different data generation processes.

In summary, we compare different rollout representations for Bayesian system identification by interfering with the data generation process. The results of APMC-ABC, REPS and the baseline SNPE-C are evaluated using the DTW discrepancy and the MSE between the reference and the simulated rollouts. It can be seen that different data generation processes have a large impact on the performance of the algorithms. Not interfering into the data generation process at all yields the best results on the Furuta pendulum. In contrast, using recorded actions on the cart-pole is necessary to fit the real data. Samples from the posteriors show that local solutions are found by APMC-ABC and REPS. Slight shifts between the posteriors of the Wassestein-optimal LFI approaches and SNPE-C might explain the different approximate posteriors of the respective inference routine.

5.3. Choice of the Likelihood-Free Inference Algorithm

The results of the previous section show that Wasserstein-optimal LFI algorithms perform better than SNPE-C on the considered tasks. On the Furuta pendulum, it can be seen that the posterior samples of SNPE-C are shifted compared to the samples obtained from Wasserstein-optimal LFI. In this section, we discuss how the differences between Wasserstein-optimal LFI and the SNPE approaches might be explained by the different optimization objective. We further discuss the choice of the LFI algorithm based on the differences of the optimization objective, run time, and sample efficiency.

As the approximate posterior is solely trained on samples from the simulator in SNPE-C, the density estimator can be interpreted as an inverse model of the simulator. Only after training, the posterior is conditioned on the real data. In contrast, Wasserstein-optimal LFI estimates the parameter distribution based on the discrepancy between real and simulated data. Therefore, the parameter distribution is also trained on real data. The different approximate parameter distributions of SNPE and Wasserstein-optimal LFI might explain the observed performance differences and shifts in the posterior plots of the previous experiments. Directly training the parameter distribution of the real system with Wasserstein-optimal LFI might lead to overfitting on the training data yielding the better scores.

Table 5.1 presents the run-times of APMC-ABC, REPS and SNPE-C applied on the Furuta pendulum for five reference rollouts and 5000 domain parameter samples per round. For Wasserstein-optimal LFI, the computational cost is almost exclusively attributed to the simulation time. In SNPE-C, neural networks in form of normalizing flows are trained which takes up a significant part of the run time. Similar observations are made by

Lueckmann *et al.* [53] on a benchmarking study on Simulation-Based Inference (SBI) which shows that SMC-ABC has lower runtimes than SNPE-C on benchmarking simulators. On the other hand, SNPE-C can reuse the samples which it has been trained on for the next iteration. Thus, SNPE-C is in general more sample efficient than SMC-ABC [25]. For simulators, where simulation time exceeds the optimization time, SNPE-C might be the right choice.

Algorithm	Sim. time	Opt. time	Avg. run-time for one iteration
APMC-ABC	99.2 %	0.08 %	37 min
REPS	99.1 %	0.09 %	34 min
SNPE-C	66.7 %	33.3 %	52 min

Table 5.1.: Run-times of APMC-ABC, REPS and SNPE-C on the Furuta pendulum. In each iteration, a total of 25 000 simulations are carried out. Each experiment is multi-processed on 16 CPUs to speed up computations.

6. Conclusion

In this thesis, we presented Wasserstein-optimal Bayesian system identification, a Likelihood-Free Inference (LFI) algorithm which leverages the Wasserstein distance between trajectories to approximate the intractable likelihood. The proposed method was validated on sim-to-sim tasks and an extensive ablation study on data generation processes was carried out on sim-to-real swing-up tasks.

6.1. Summary

Sequential Monte Carlo Approximate Bayesian Computation (SMC-ABC) and Relative Entropy Policy Search (REPS) are two Bayesian inference approaches developed in two different areas of research. In this work we showed that both approaches approximate the same posterior by approximating the intractable likelihood with an Energy-Based Model (EBM) for which the energy functional is a distance between real and simulated data. By making the likelihood tractable, classical sampling strategies can be applied yielding the SMC-ABC algorithm. Using parameterized models, REPS carries out inference by weighting samples, drawn from the sampling distribution with the likelihood, and maximizing the Kullback-Leibler (KL) divergence between the true and approximate parameter distribution.

We applied the Wasserstein distance between trajectories as the energy functional for the LFI approaches SMC-ABC and REPS. The Wasserstein-optimal LFI algorithms were validated in sim-to-sim experiments on the damped harmonic oscillator and on the Furuta pendulum. Here, all inference algorithms could reproduce the linear correlations while REPS had problems representing the non-linear correlations of the Furuta pendulum due its Gaussian model assumptions.

Afterwards, an ablation study was carried out on the sim-to-real swing-up tasks of the cart-pole and the Furuta pendulum to investigate the effect of different data generation

processes, namely, using recorded actions from the real system, resetting the states to the real states segmentwise, or adding state noise. Using recorded actions was crucial for a successful application of LFI on the cart-pole while using the policy was beneficiary on the Furuta pendulum. Wasserstein-optimal LFI showed improved performance compared to the baseline SNPE-C on the specified metrics, Dynamic Time Warping (DTW) discrepancy and the MSE, between simulated and real trajectories. Posterior plots revealed that samples from SNPE-C were slightly shifted from the posterior samples of Wasserstein-optimal LFI which might have explained the performance differences.

6.2. Future Work

The presented Wasserstein-optimal Bayesian system identification approach can be straightforwardly integrated into Domain Randomization (DR) to enable sim-to-real transfer.

In Chapter 5, it was shown that Wasserstein-optimal LFI performs better than SNPE-C on the considered tasks. However, SNPE-C does not assume any form of the density estimator. The benefits of normalizing flows were visible on the sim-to-sim Furuta pendulum where SNPE-C was capable of estimating the nonlinear correlations. Adding more flexible density estimators could further improve the performance of Wasserstein-optimal Bayesian system identification. Gaussian mixture models have been successfully employed within REPS [54] and is one possibility to overcome the restrictions on the parameter distribution. Using differentiable simulators is another idea to sidestep the non-differentiability of the optimization objective in Eq. (3.20). This requirement on the simulator enables the training of neural models using gradient descent.

As discussed in Section 3.6, the inner expectation over trajectories in Eq. (3.14) was approximated by a single sample in this thesis. A further study is necessary to evaluate whether this assumption is valid.

In this thesis, changing the data generation process showed significant changes in the performance of the inference algorithms. Instead of further interfering with the data generation process, one approach is to consider step-wise data comparison instead of full trajectories. As time-series tend to diverge over long time horizons, comparing transitional changes could make the inference routine more robust. Furthermore, intertwining online Reinforcement Learning (RL) algorithms with an online update of the domain parameter distribution could improve the sample efficiency of the Domain Randomization (DR) method.

Bibliography

[1] F. Sadeghi and S. Levine, "CAD2RL: real single-image flight without a single real image," in *Robotics: Science and Systems XIII, MIT, Cambridge, Massachusetts, USA, July 12-16, 2017*, 2017.

[2] B. Mehta, M. Diaz, F. Golemo, C. J. Pal, and L. Paull, "Active domain randomization," in *CoRL, Osaka, Japan, October 30 - November 1*, vol. 100 of *PMLR*, pp. 1162–1176, 2019.

[3] Y. Chebotar *et al.*, "Closing the sim-to-real loop: Adapting simulation randomization with real world experience," in *ICRA, Montreal, QC, Canada, May 20-24*, pp. 8973–8979, IEEE, 2019.

[4] F. Ramos, R. Possas, and D. Fox, "BayesSim: Adaptive domain randomization via probabilistic inference for robotics simulators," in *Robotics: Science and Systems XV, Freiburg im Breisgau, Germany, June 22-26*, 2019.

[5] F. Muratore, C. Eilers, M. Gienger, and J. Peters, "Data-efficient domain randomization with bayesian optimization," *IEEE Robotics Autom. Lett.*, vol. 6, no. 2, pp. 911–918, 2021.

[6] F. Muratore, T. Gruner, F. Wiese, B. Belousov, M. Gienger, and J. Peters, "Neural posterior domain randomization," in *CoRL*, 2021.

[7] J. Peters, K. Mulling, and Y. Altun, "Relative entropy policy search," in *AAAI, Atlanta, Georgia, USA, July 11-15*, 2010.

[8] B. Baker *et al.*, "Emergent tool use from multi-agent autocurricula," in *ICLR, Addis Ababa, Ethiopia, April 26-30*, 2020.

[9] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *IROS, Vancouver, BC, Canada, September 24-28*, pp. 23–30, IEEE, 2017.

[10] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *CoRR*, vol. abs/1707.06347, 2017.

[11] J. Kober and J. Peters, "Policy search for motor primitives in robotics," *Mach. Learn.*, vol. 84, no. 1-2, pp. 171–203, 2011.

[12] R. Possas, L. Barcelos, R. Oliveira, D. Fox, and F. Ramos, "Online BayesSim for combined simulator parameter inference and policy improvement," in *IROS, Las Vegas, NV, USA, October 24, 2020 - January 24, 2021*, pp. 5445–5452, IEEE, 2020.

[13] G. Papamakarios, I. Murray, and T. Pavlakou, "Masked autoregressive flow for density estimation," in *NIPS, December 4-9, Long Beach, CA, USA*, pp. 2338–2347, 2017.

[14] K. Cranmer, J. Brehmer, and G. Louppe, "The frontier of simulation-based inference," *Proceedings of the National Academy of Sciences*, vol. 117, no. 48, pp. 30055–30062, 2020.

[15] G. Karabatsos and F. Leisen, "An approximate likelihood perspective on ABC methods," *Statistics Surveys*, vol. 12, no. none, pp. 66 – 104, 2018.

[16] P. Marjoram, J. Molitor, V. Plagnol, and S. Tavaré, "Markov chain monte carlo without likelihoods," *Proceedings of the National Academy of Sciences*, vol. 100, no. 26, pp. 15324–15328, 2003.

[17] S. A. Sisson, Y. Fan, and M. M. Tanaka, "Sequential monte carlo without likelihoods," *Proceedings of the National Academy of Sciences*, vol. 104, no. 6, pp. 1760–1765, 2007.

[18] M. A. Beaumont, "Approximate bayesian computation in evolution and ecology," *Annu. Rev. Ecol. Evol. Syst.*, vol. 41, no. 1, pp. 379–406, 2010.

[19] C. C. Drovandi and A. N. Pettitt, "Estimation of parameters for macroparasite population evolution using approximate bayesian computation," *Biometrics*, vol. 67, no. 1, pp. 225–233, 2011.

[20] P. D. Moral, A. Doucet, and A. Jasra, "An adaptive sequential monte carlo method for approximate bayesian computation," *Stat. Comput.*, vol. 22, no. 5, pp. 1009–1020, 2012.

[21] M. Lenormand, F. Jabot, and G. Deffuant, "Adaptive approximate bayesian computation for complex models," *Comput. Stat.*, vol. 28, no. 6, pp. 2777–2796, 2013.

[22] C. Albert, H. R. Künsch, and A. Scheidegger, "A simulated annealing approach to approximate bayes computations," *Stat. Comput.*, vol. 25, no. 6, pp. 1217–1232, 2015.

[23] G. Papamakarios and I. Murray, "Fast ϵ-free inference of simulation models with bayesian conditional density estimation," in *NIPS, December 5-10, 2016, Barcelona, Spain*, pp. 1028–1036, 2016.

[24] J. Lueckmann, P. J. Gonçalves, G. Bassetto, K. Öcal, M. Nonnenmacher, and J. H. Macke, "Flexible statistical inference for mechanistic models of neural dynamics," in *NIPS, December 4-9, Long Beach, CA, USA*, pp. 1289–1299, 2017.

[25] D. S. Greenberg, M. Nonnenmacher, and J. H. Macke, "Automatic posterior transformation for likelihood-free inference," in *ICML, 9-15 June, Long Beach, California, USA*, vol. 97, pp. 2404–2414, PMLR, 2019.

[26] G. Papamakarios, D. C. Sterratt, and I. Murray, "Sequential neural likelihood: Fast likelihood-free inference with autoregressive flows," in *AISTATS, 16-18 April, Naha, Okinawa, Japan*, vol. 89, pp. 837–848, PMLR, 2019.

[27] C. Durkan, I. Murray, and G. Papamakarios, "On contrastive learning for likelihood-free inference," in *ICML 2020, 13-18 July, Virtual Event*, vol. 119, pp. 2771–2781, PMLR, 2020.

[28] M. Muskulus and S. Verduyn-Lunel, "Wasserstein distances in the analysis of time series and dynamical systems," *Physica D: Nonlinear Phenomena*, vol. 240, no. 1, pp. 45–58, 2011.

[29] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE Trans. on Acoustics, Speech, and Sig. Proc.*, vol. 26, no. 1, pp. 43–49, 1978.

[30] M. Cuturi and M. Blondel, "Soft-dtw: a differentiable loss function for time-series," in *ICML, Sydney, NSW, Australia, 6-11 August*, vol. 70, pp. 894–903, PMLR, 2017.

[31] S.-i. Amari, *Information geometry and its applications*, vol. 194. Springer, 2016.

[32] I. J. Goodfellow *et al.*, "Generative adversarial nets," in *NIPS, December 8-13, Montreal, Quebec, Canada*, pp. 2672–2680, 2014.

[33] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein GAN," *CoRR*, vol. abs/1701.07875, 2017.

[34] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *Proceedings of the 32nd International Conference on Machine Learning* (F. Bach and D. Blei, eds.), vol. 37 of *Proceedings of Machine Learning Research*, (Lille, France), pp. 1889–1897, PMLR, 07–09 Jul 2015.

[35] C. Villani, *Optimal transport: old and new*, vol. 338. Springer, 2009.

[36] M. Cuturi, "Sinkhorn distances: Lightspeed computation of optimal transport," in *NIPS, Lake Tahoe, Nevada, United States, December 5-8*, pp. 2292–2300, 2013.

[37] R. Sinkhorn, "A Relationship Between Arbitrary Positive Matrices and Doubly Stochastic Matrices," vol. 35, no. 2, pp. 876 – 879, 1964.

[38] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley Series in Probability and Statistics, Wiley, 1994.

[39] T. Giorgino, "Computing and visualizing dynamic time warping alignments in r: The dtw package," *Journal of Statistical Software, Articles*, vol. 31, no. 7, pp. 1–24, 2009.

[40] E. Bernton, P. E. Jacob, M. Gerber, and C. P. Robert, "Approximate bayesian computation with the wasserstein distance," *J. R. Stat. Soc.: Series B (Statistical Methodology)*, vol. 81, no. 2, pp. 235–269, 2019.

[41] G. Peyré and M. Cuturi, "Computational optimal transport," *Found. Trends Mach. Learn.*, vol. 11, no. 5-6, pp. 355–607, 2019.

[42] S. A. Sisson, Y. Fan, and M. Beaumont, *Handbook of approximate Bayesian computation (1st ed.)*. Chapman and Hall/CRC, 2018.

[43] S. Tavaré, D. J. Balding, R. C. Griffiths, and P. Donnelly, "Inferring coalescence times from dna sequence data," vol. 145, no. 2, pp. 505–518, 1997.

[44] T. Toni, D. Welch, N. Strelkowa, A. Ipsen, and M. P. Stumpf, "Approximate bayesian computation scheme for parameter inference and model selection in dynamical systems," *J. R. Stat. Soc. Interface*, vol. 6, no. 31, pp. 187–202, 2009.

[45] Y. LeCun, S. Chopra, R. Hadsell, M. Ranzato, and F. Huang, "A tutorial on energy-based learning," *Predicting Structured Data*, vol. 1, no. 0, 2006.

[46] P. Fearnhead and D. Prangle, "Constructing summary statistics for approximate bayesian computation: semi-automatic approximate bayesian computation," *J. R. Stat. Soc.*, vol. 74, no. 3, pp. 419–474, 2012.

[47] S. M. Kakade, "A natural policy gradient," in *NIPS, December 3-8, Vancouver, British Columbia, Canada]* (T. G. Dietterich, S. Becker, and Z. Ghahramani, eds.), pp. 1531–1538, MIT Press, 2001.

[48] B. Belousov, H. Abdulsamad, P. Klink, S. Parisi, and J. Peters, *Reinforcement Learning Algorithms: Analysis and Applications*. Springer, 2021.

[49] K. Furuta, M. Yamakita, and S. Kobayashi, "Swing-up control of inverted pendulum using pseudo-state feedback," *Proc. Inst. Mech. Eng., Part I: Journal of Systems and Control Engineering*, vol. 206, no. 4, pp. 263–269, 1992.

[50] F. Muratore, "SimuRLacra — a framework for reinforcement learning from randomized simulations." `https://github.com/famura/SimuRLacra`, 2020.

[51] A. Tejero-Cantero *et al.*, "sbi: A toolkit for simulation-based inference," *J. Open Source Softw.*, vol. 5, no. 52, p. 2505, 2020.

[52] J. Feydy, T. Séjourné, F. Vialard, S. Amari, A. Trouvé, and G. Peyré, "Interpolating between optimal transport and MMD using sinkhorn divergences," in *AISTATS, 16-18 April, Naha, Okinawa, Japan*, vol. 89, pp. 2681–2690, PMLR, 2019.

[53] J. Lueckmann, J. Boelts, D. S. Greenberg, P. J. Gonçalves, and J. H. Macke, "Benchmarking simulation-based inference," in *AISTATS April 13-15, Virtual Event*, vol. 130, pp. 343–351, PMLR, 2021.

[54] O. Arenz, M. Zhong, and G. Neumann, "Trust-region variational inference with gaussian mixture models," *J. Mach. Learn. Res.*, vol. 21, pp. 163:1–163:60, 2020.

[55] M. P. Deisenroth, G. Neumann, and J. Peters, "A survey on policy search for robotics," *Found. Trends Robotics*, vol. 2, no. 1-2, pp. 1–142, 2013.

A. Appendix

A.1. Derivation of the Optimal Domain Parameter Distribution in REPS

The derivation of the optimal domain parameter distribution closely follows the derivation of the optimal policy in REPS [7]. A thorough insight into REPS and its derivation can also be found in [55]. The Lagrangian of the optimization problem in Eq. (3.20) is

$$\begin{aligned}\mathcal{L} = &\iint q(\boldsymbol{\tau}_{1:M}, \boldsymbol{\xi})\, D\left(\boldsymbol{\tau}_{1:N}^{\text{real}}, \boldsymbol{\tau}_{1:M}\right)\, \mathrm{d}\boldsymbol{\tau}_{1:M}\, \mathrm{d}\boldsymbol{\xi} \qquad \text{(A.1)}\\ &+ \eta\left(\iint q(\boldsymbol{\tau}_{1:M}, \boldsymbol{\xi})\, \log\frac{q(\boldsymbol{\tau}_{1:M}, \boldsymbol{\xi})}{p(\boldsymbol{\tau}_{1:M}, \boldsymbol{\xi})}\, \mathrm{d}\boldsymbol{\tau}_{1:M}\, \mathrm{d}\boldsymbol{\xi} - \varepsilon\right)\\ &+ \lambda\left(\iint q(\boldsymbol{\tau}_{1:M}, \boldsymbol{\xi})\, \mathrm{d}\boldsymbol{\tau}_{1:M}\, \mathrm{d}\boldsymbol{\xi} - 1\right)\\ = &\iint q(\boldsymbol{\tau}_{1:M}, \boldsymbol{\xi})\left[D\left(\boldsymbol{\tau}_{1:N}^{\text{real}}, \boldsymbol{\tau}_{1:M}\right) + \eta\log\frac{q(\boldsymbol{\tau}_{1:M}, \boldsymbol{\xi})}{q(\boldsymbol{\tau}_{1:M}, \boldsymbol{\xi})} + \lambda\right]\, \mathrm{d}\boldsymbol{\tau}_{1:M}\, \mathrm{d}\boldsymbol{\xi} - \eta\varepsilon - \lambda \qquad \text{(A.2)}\end{aligned}$$

The optimal domain parameter distribution is found where the derivative of the Lagrangian w.r.t. $q(\boldsymbol{\xi})$ vanishes

$$\left.\frac{\partial\mathcal{L}}{\partial q}\right|_{q=q^\star} = D\left(\boldsymbol{\tau}_{1:N}^{\text{real}}, \boldsymbol{\tau}_{1:M}\right) + \eta\left[\log\frac{q^\star(\boldsymbol{\tau}_{1:M}, \boldsymbol{\xi})}{p(\boldsymbol{\tau}_{1:M}, \boldsymbol{\xi})} + 1\right] + \lambda = 0. \qquad \text{(A.3)}$$

Reformulating the above expression yields the solution

$$q^\star(\boldsymbol{\tau}_{1:M}, \boldsymbol{\xi}) = p(\boldsymbol{\tau}_{1:M}, \boldsymbol{\xi})\, e^{-\frac{1}{\eta}D(\boldsymbol{\tau}_{1:N}^{\text{real}}, \boldsymbol{\tau}_{1:M})} e^{\frac{-\eta-\lambda}{\eta}} \qquad \text{(A.4)}$$

The last term can be substituted by

$$e^{\frac{\eta+\lambda}{\eta}} = \iint p(\boldsymbol{\tau}_{1:M}, \boldsymbol{\xi})\; e^{-\eta^{-1} D(\boldsymbol{\tau}_{1:N}^{\text{real}}, \boldsymbol{\tau}_{1:M})}\; \mathrm{d}\boldsymbol{\tau}_{1:M}\; \mathrm{d}\boldsymbol{\xi} \tag{A.5}$$

yielding the optimal solution. The last term is a result of the last constraint in (eq. (3.20)) making $q(\boldsymbol{\tau}_{1:M}, \boldsymbol{\xi})$ a probability density. Lastly, we can obtain the dual by plugging in the optimal domain parameter distribution into Eq. (A.2)

$$\begin{aligned} g(\eta) &= -\eta\varepsilon - \eta - \lambda \\ &= -\eta\varepsilon - \eta \log \left(\iint p(\boldsymbol{\tau}_{1:M}, \boldsymbol{\xi}) e^{-\frac{1}{\eta}\, D(\boldsymbol{\tau}_{1:N}^{\text{real}}, \boldsymbol{\tau}_{1:M})}\; \mathrm{d}\boldsymbol{\tau}_{1:M}\; \mathrm{d}\boldsymbol{\xi} \right) \\ &= -\eta\varepsilon - \eta \log \mathop{\mathbb{E}}_{p(\boldsymbol{\tau}_{1:M}, \boldsymbol{\xi})} \left[e^{-\frac{1}{\eta}\, D(\boldsymbol{\tau}_{1:N}^{\text{real}}, \boldsymbol{\tau}_{1:M})} \right]. \end{aligned} \tag{A.6}$$

A.2. Derivation of Domain Parameter Correlations for Furuta Pendulum

Muratore *et al.* [6] show that the equations of motion of the Furuta pendulum can be formulated as a linear equation w.r.t. the parameters $\boldsymbol{w}$

$$\begin{bmatrix} \theta_r & \ddot{\theta}_r \sin^2\theta_p + \dot{\theta}_r \theta_p \sin(2\theta_p) & \ddot{\theta}_p \cos\theta_p - \dot{\theta}_p^2 \sin\theta_p & 0 & \dot{\theta}_r & 0 \\ 0 & \frac{4}{3}\ddot{\theta}_p - \theta_r^2 \sin\theta_p \cos\theta_p & \ddot{\theta}_r \cos\theta_p & \sin\theta_p & 0 & \dot{\theta}_p \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \\ w_4 \\ w_5 \end{bmatrix} = \begin{bmatrix} \frac{k_m}{r_m} u \\ 0 \end{bmatrix}. \tag{A.7}$$

The parameters $\boldsymbol{w}$ are functions of the domain parameters which can be recovered by

following correlations:

$$\frac{l_r}{g} = \frac{w_2}{w_3}, \quad \frac{l_p}{g} = 2\frac{w_1}{w_3}, \quad \frac{l_r}{l_p} = \frac{w_2}{2w_1},$$
$$m_p g^2 = \frac{w_3^2}{w_1}, \quad m_p l_r^2 = \frac{w_2^2}{w_1}, \quad m_p l_p^2 = 4w_1, \quad \frac{m_r}{m_p} = 12\left(\frac{w_0 w_1}{w_2^2} - 1\right) \tag{A.8}$$
$$m_r g^2 = 12\frac{w_3^2}{w_1}\left(\frac{w_0 w_1}{w_2^2} - 1\right), \quad m_r l_r^2 = 12\,\frac{w_2^2}{w_1}\left(\frac{w_0 w_1}{w_2^2} - 1\right), \quad m_r l_p^2 = 48\,w_1\left(\frac{w_0 w_1}{w_2^2} - 1\right).$$

Let a rollout $\boldsymbol{\tau}$ be characterized by the generalized coordinates θ_p, θ_r and its derivatives $\dot{\theta}_p$, $\dot{\theta}_r$, $\ddot{\theta}_p$, $\ddot{\theta}_r$, and assume that a unique solution $\boldsymbol{w}^\star$ of Eq. (A.7) exists. Then, all domain parameter configurations that comply with the conditions in Eq. (A.8) yield the same rollout. Note that w_4 and w_5 are neglected as they only contain information about the motor constants and damping coefficients which are set to a constant value during the experiments. We refer to the supplementary materials of [6] for further information.

A.3. Algorithmic Configurations

Damped harmonic Oscillator

Parameter		Value
Sampling rate		250[Hz]
Downsampling factor		1
Rollout length		8 [s]
Number of DPs samples		1000
Number of rollouts per DPs		1
Dp	Prior range	Unit
m	$[0.5, 1.5]$	kg
c	$[0.25, 0.75]$	$\mathrm{N\,m^{-1}}$
k	$[7.5, 52.5]$	$\mathrm{N\,s\,m^{-1}}$

Table A.1.: Algorithmic and experimental configuration of the one-mass oscillator.

Furuta pendulum	
Parameter	Value
Sampling rate	500[Hz]
Downsampling factor	5
Rollout length	3[s]
Number of DPs samples	1000, 5000
Number of rollouts per DPs	1,5
Policy	Hybrid controller: energy-based + PD controller
REPS	
Number of iterations	20
Model	Multivariate Gaussian
KL bound ε	0.3
APMC-ABC	
Number of iterations	20
Model	Particles
Cov Factor	2
α	0.1
Acceptance cut-off	0.0
SNPE-C	
Model	MAF [13]
Embedding layer	FNN with 256 output neurons
Number of transformations	5
Number of features	50
Number of iterations	9

Table A.2.: Algorithmic configurations for Bayesian system identification on the Furuta pendulum.

Cart-pole	
Parameter	Value
Sampling rate	250[Hz]
Downsampling factor	1
Rollout length	4.5 [s]
Number of DPs samples	1000, 5000
Number of rollouts per DPs	1,5
Policy	Hybrid controller: energy-based + PD controller
REPS	
Number of iterations	20
Model	Multivariate Gaussian
KL bound ε	0.3
APMC-ABC	
Number of iterations	20
Model	Particles
Cov Factor	2
α	0.1
Acceptance cut-off	0.0
SNPE-C	
Model	MAF [13]
Embedding layer	FNN with 256 output neurons
Number of transformations	5
Number of features	50
Number of iterations	13

Table A.3.: Algorithmic configurations for Bayesian system identification on the cart-pole.

Furuta Pendulum			Cart Pole		
Dp	Prior range	Unit	Dp	Prior range	Unit
g	$[8.34, 11.28]$	$\mathrm{m\,s^{-2}}$	m_c	$[0.19, 0.57]$	kg
m_r	$[0.029, 0.162]$	kg	m_p	$[0.0381, 0.2159]$	kg
m_p	$[0.007, 0.041]$	kg	l_p	$[84.1, 252.4]$	mm
l_r	$[42.5, 127.5]$	mm	d_c	$[2.7, 8.1]$	$\mathrm{N\,s\,m^{-1}}$
l_p	$[64.5, 193.5]$	mm	d_p	$[0.00, 1.0]$	N s
			V_{min}	$[0.00, 2.0]$	V
			V_{max}	$-[0.00, 2.0]$	V

Table A.4.: Domain parameters of the Furuta pendulum and the cart-pole environment. The domain parameters are sampled within a fixed prior region to prohibit physically unplausible behavior.

A.4. Validation of APMC-ABC and SNPE-C on Damped Harmonic Oscillator

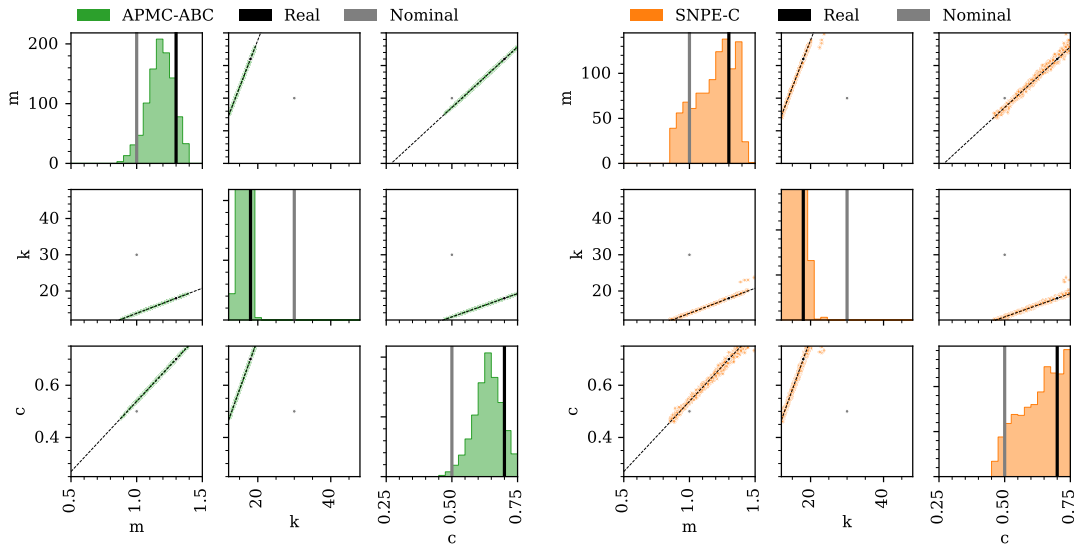


Figure A.1.: Pair-plots of the trained APMC-ABC and SNPE-C posteriors on the damped harmonic oscillator. We report the posterior after 10 iterations and 1000 samples.

A.5. Ablation Study of Data Generation Processes

	DTW			MSE		
	APMC-ABC	REPS	SNPE-C	APMC-ABC	REPS	SNPE-C
1 rr	(0.79±0.06)e−1	**(0.63±0.08)e−1**	(1.38±0.24)e−1	(3.82±0.27)e−1	**(1.84±0.16)e−1**	(3.05±0.42)e−1
1 rr seg	(3.91±0.20)e−1	(4.45±2.75)e−1	(5.98±2.82)e−1	(12.40±0.17)e−1	(13.80±5.94)e−1	(17.93±5.29)e−1
1 rr seg ur	(3.85±0.03)e−1	(3.85±0.39)e−1	(4.64±0.51)e−1	(6.14±0.02)e−1	(6.44±1.10)e−1	(8.26±1.07)e−1
1 rr sn	(1.12±0.16)e−1	(0.99±0.30)e−1	(2.72±0.26)e−1	(3.71±0.33)e−1	(2.65±0.80)e−1	(7.85±0.87)e−1
1 rr ur	(2.14±0.01)e−1	(2.38±0.19)e−1	(3.94±0.78)e−1	(2.43±0.01)e−1	(2.73±0.32)e−1	(4.92±1.23)e−1
1 rr ur sn	(3.27±0.09)e−1	(3.00±0.07)e−1	(4.39±1.52)e−1	(3.87±0.11)e−1	(3.64±0.10)e−1	(6.19±3.54)e−1
5 rr	(0.82±0.05)e−1	**0.61e−1**	(1.90±0.35)e−1	(3.77±0.29)e−1	**1.80e−1**	(3.93±0.96)e−1
5 rr seg	(3.89±0.15)e−1	(3.59±0.03)e−1	(8.06±2.99)e−1	(12.46±0.18)e−1	(11.81±0.13)e−1	(20.05±5.54)e−1
5 rr seg ur	(3.95±0.03)e−1	3.89e−1	(5.00±0.64)e−1	(6.33±0.02)e−1	6.18e−1	(8.14±0.49)e−1
5 rr sn	(0.90±0.04)e−1	(0.79±0.01)e−1	(1.72±0.60)e−1	(3.03±0.11)e−1	(2.12±0.03)e−1	(5.18±1.60)e−1
5 rr ur	2.51e−1	(2.67±0.07)e−1	(3.93±0.59)e−1	2.81e−1	(3.00±0.09)e−1	(4.90±1.08)e−1
5 rr ur sn	(3.77±0.13)e−1	(3.38±0.07)e−1	(4.64±0.45)e−1	(4.93±0.23)e−1	(4.06±0.09)e−1	(5.80±0.79)e−1

Table A.5.: Ablation study of the inference approaches using different trajectory representations on the Furuta pendulum. We report the mean DTW discrepancy and the mean MSE between the reference trajectories and the simulations generated from 100 domain parameters of the trained posterior. In case of 5 reference rollouts, 5 simulated rollouts are generated from a single domain parameter and compared via the Wasserstein distance. For each configuration 12 experiments have been conducted and trained until convergence (20 iterations); **rr**: number of reference rollouts; **sn**: state noise is applied; **ur**: the same actions are used on the physical and simulated system, **seg**: the rollouts are segmented into 5 trajectories of equal length and are reset to the states of the real trajectory; **vanilla**: neither of the described modifications is applied.

	DTW			MSE		
	APMC-ABC	REPS	SNPE-C	APMC-ABC	REPS	SNPE-C
1 rr	(2.31±0.41)e−1	(3.14±0.66)e−1	(4.82±2.07)e−1	(7.84±1.77)e−1	(13.30±3.25)e−1	(16.06±7.03)e−1
1 rr seg	(2.96±0.15)e−1	(5.20±0.62)e−1	(5.26±1.23)e−1	(12.01±0.48)e−1	(19.27±1.71)e−1	(19.22±3.91)e−1
1 rr seg ur	**(1.25±0.12)e−1**	**(1.31±0.32)e−1**	**(3.10±2.14)e−1**	**(2.66±0.20)e−1**	**(2.87±0.62)e−1**	**(5.08±2.50)e−1**
1 rr sn	(3.01±1.26)e−1	(3.00±0.71)e−1	(4.41±0.76)e−1	(7.75±1.93)e−1	(12.31±4.26)e−1	(12.58±1.02)e−1
1 rr ur	**(1.23±0.20)e−1**	**(1.25±0.53)e−1**	**(0.98±1.13)e−1**	**(2.74±0.40)e−1**	**(2.18±0.81)e−1**	**(1.64±1.26)e−1**
1 rr ur sn	**(1.21±0.15)e−1**	**(1.76±0.96)e−1**	**(2.48±1.73)e−1**	**(2.73±0.29)e−1**	**(2.81±1.25)e−1**	**(3.33±1.76)e−1**
5 rr	(2.61±0.16)e−1	(4.75±0.17)e−1	(5.15±0.99)e−1	(8.63±0.50)e−1	(11.90±0.19)e−1	(12.84±2.76)e−1
5 rr seg	(3.37±0.13)e−1	(4.04±0.19)e−1	(6.35±1.89)e−1	(12.93±0.31)e−1	(15.25±0.52)e−1	(20.73±3.16)e−1
5 rr seg ur	(1.29±0.05)e−1	(1.17±0.14)e−1	(4.44±1.24)e−1	(2.66±0.08)e−1	(2.49±0.26)e−1	(6.37±1.25)e−1
5 rr sn	(2.59±0.13)e−1	(4.67±0.11)e−1	(4.91±0.60)e−1	(8.31±0.33)e−1	(11.93±0.11)e−1	(12.89±0.68)e−1
5 rr ur	(1.28±0.11)e−1	**(0.65±0.10)e−1**	(1.66±1.04)e−1	(2.74±0.21)e−1	**(1.23±0.17)e−1**	(2.54±1.44)e−1
5 rr ur sn	(1.12±0.09)e−1	**(0.61±0.13)e−1**	(3.07±1.31)e−1	(2.49±0.18)e−1	**(1.19±0.29)e−1**	(3.86±1.31)e−1

Table A.6.: Ablation study of the inference approaches using different trajectory representations on the cart-pole. We report the mean DTW discrepancy and the mean MSE between the reference trajectories and the simulations generated from 100 domain parameters of the trained posterior. In case of 5 reference rollouts, 5 simulated rollouts are generated from a single domain parameter and compared via the Wasserstein distance. For each configuration 12 experiments have been conducted and trained until convergence (20 iterations); **rr**: number of reference rollouts; **sn**: state noise is applied; **ur**: the same actions are used on the physical and simulated system, **seg**: the rollouts are segmented into 5 trajectories of equal length and are reset to the states of the real trajectory; **vanilla**: neither of the described modifications is applied.

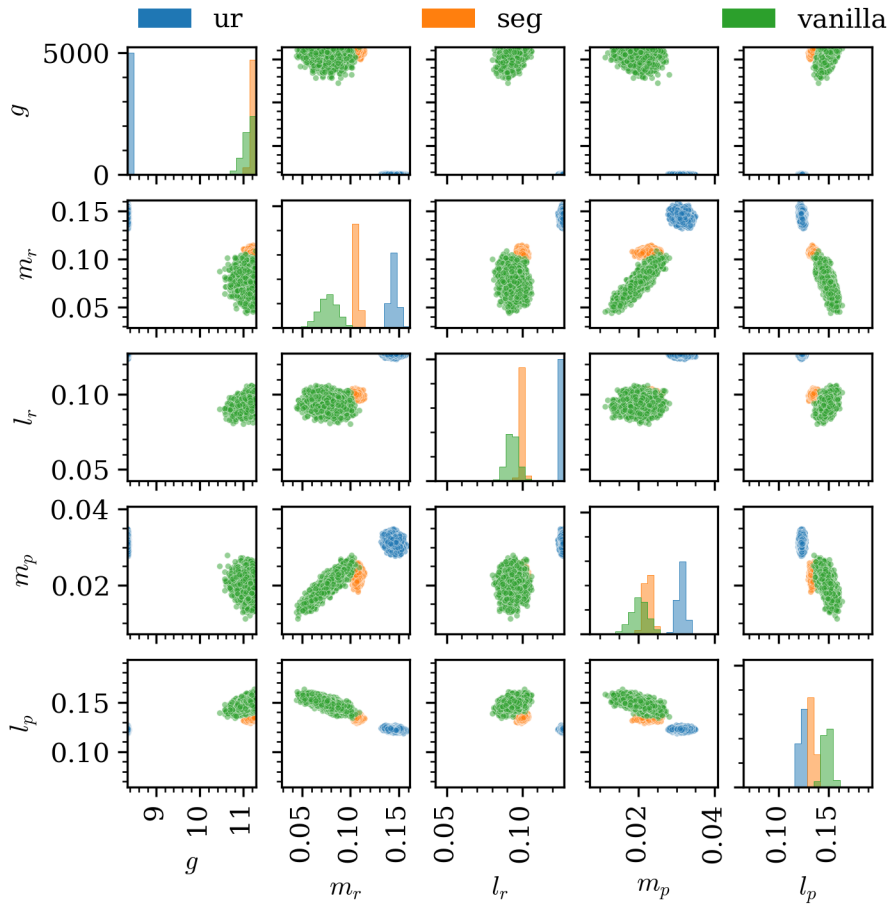


Figure A.2.: Posterior of APMC-ABC trained with 3 different data representations on the cart-pole; **ur**: the same actions are used on the physical and simulated system, **seg**: the rollouts are segmented into 5 trajectories of equal length and are reset to the states of the real trajectory; **vanilla**: neither of the described modifications is applied.

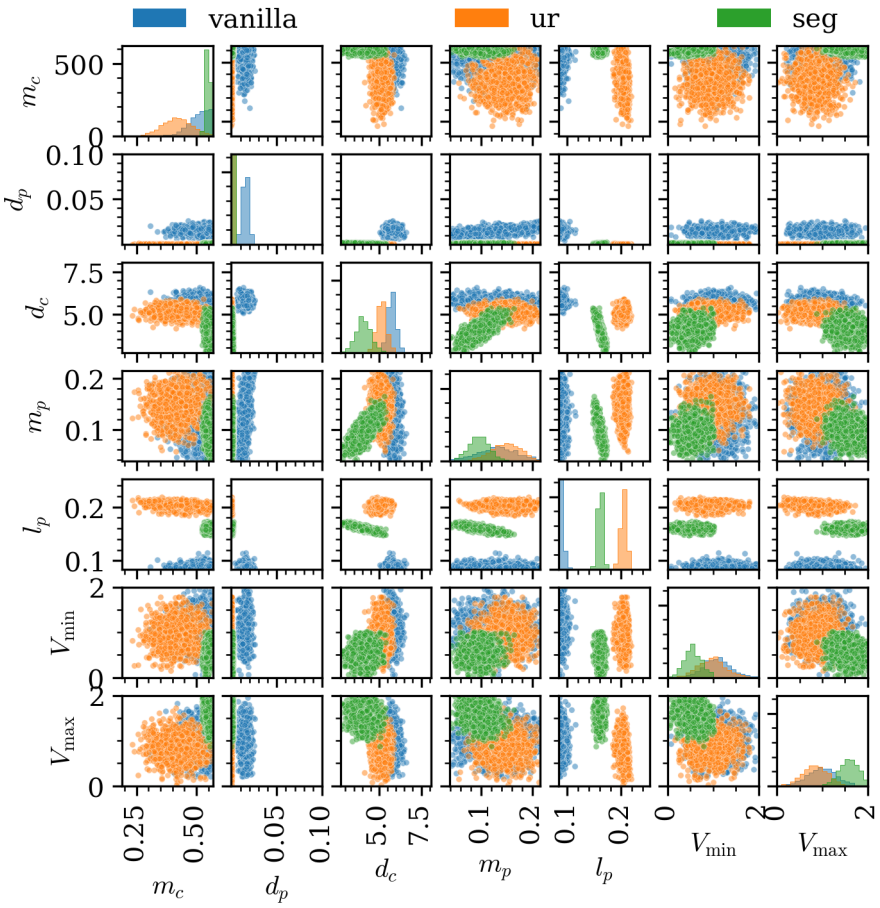


Figure A.3.: Posterior of REPS trained with 3 different data representations on the cart-pole; **ur**: the same actions are used on the physical and simulated system, **seg**: the rollouts are segmented into 5 trajectories of equal length and are reset to the states of the real trajectory; **vanilla**: neither of the described modifications is applied.

Figures and Tables

List of Figures

List of Tables