– Appendix –
Neural Posterior Domain Randomization

Fabio Muratore[1,2], **Theo Gruner**[1], **Florian Wiese**[1],
Boris Belousov[1], **Michael Gienger**[2], **Jan Peters**[1]
[1] Intelligent Autonomous Systems Group, Technical University Darmstadt, Germany
[2] Honda Research Institute Europe, Offenbach am Main, Germany
Correspondence to fabio@robot-learning.de

A Modeling Description for the Pendulum

The pendulum (Section 4.1) is modeled as a rigid rod, mounted to a frictionless rotational joint at the top. Its Equation of Motion (EoM) is modeled as

$$\ddot{\theta} = \frac{\tau - \frac{1}{2} g m_p l_p \sin(\theta)}{\frac{1}{3} m_p l_p^2},$$

where θ is the angle towards the vertical axis ($\theta = 0$ when hanging down), τ is the torque applied to the pendulum at the joint, m_p and l_p are the pendulum's mass and length, d_p is the viscous friction coefficient, and g is the gravitational acceleration constant.

B System Identification via Bayesian Linear Regression on the Furuta Pendulum

In general, rigid body dynamics can be written in the form

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{\tau},$$

where $\boldsymbol{M}$ is the mass/inertia matrix, $\boldsymbol{q}$ are generalized coordinates, $\boldsymbol{c}$ is a nonlinear term consisting of centrifugal, Coriolis, and gravitation components, and $\boldsymbol{\tau}$ are forces/torques. In case of the Furuta pendulum with generalized coordinates $\boldsymbol{q} = [\theta, \alpha]^\mathsf{T}$, with $\boldsymbol{q} = \boldsymbol{0}$ when handing down centered, the components of this equation are given by

$$\begin{aligned}
\boldsymbol{M}(\boldsymbol{q}) &= \begin{bmatrix} w_0 + w_1 \sin^2(\alpha) & w_2 \cos(\alpha) \\ w_2 \cos(\alpha) & \frac{4}{3} w_1 \end{bmatrix}, \\
\boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}}) &= \begin{bmatrix} w_1 \sin(2\alpha)\, \dot{\theta}\dot{\alpha} - w_2 \sin(\alpha)\, \dot{\alpha}^2 + w_4 \dot{\theta} \\ -\frac{1}{2} w_1 \sin(2\alpha)\, \dot{\theta}^2 + w_3 \sin(\alpha) + w_5 \dot{\alpha} \end{bmatrix}, \\
\boldsymbol{\tau} &= \begin{bmatrix} \gamma u \\ 0 \end{bmatrix},
\end{aligned} \tag{1}$$

with parameters $\boldsymbol{w}$ defined as follows

$$\begin{aligned}
w_0 &= \left(\frac{1}{12} M_r + M_p \right) L_r^2, \\
w_1 &= \frac{1}{4} M_p L_p^2, \\
w_2 &= \frac{1}{2} M_p L_p L_r, \\
w_3 &= \frac{1}{2} M_p L_p g, \\
w_4 &= \frac{k_m^2}{R_m} + D_r, \\
w_5 &= D_p.
\end{aligned} \tag{2}$$

5th Conference on Robot Learning (CoRL 2021), London, UK.

The parameter $\gamma = k_m/R_m$ in front of the control input u characterizes motor properties and $[k_m, R_m]$ are assumed to be known. Since the equations of motion are linear in the parameters, they can be written in the form

$$\begin{bmatrix} \ddot{\theta} & \ddot{\theta}\sin^2(\alpha) + \dot{\theta}\dot{\alpha}\sin(2\alpha) & \ddot{\alpha}\cos(\alpha) - \dot{\alpha}^2\sin(\alpha) & 0 & \dot{\theta} & 0 \\ 0 & \frac{4}{3}\ddot{\alpha} - \dot{\theta}^2\sin(\alpha)\cos(\alpha) & \ddot{\theta}\cos(\alpha) & \sin(\alpha) & 0 & \dot{\alpha} \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \\ w_4 \\ w_5 \end{bmatrix} = \begin{bmatrix} \gamma u \\ 0 \end{bmatrix}$$

and one can find vector w by linear regression, subsequently recovering the physics parameters as

$$L_r = \frac{w_2}{w_3} g, \quad L_p = 2\frac{w_1}{w_3} g, \quad M_p = \frac{w_3^2}{w_1 g^2}, \quad M_r = 12\frac{w_3^2}{w_1 g^2}\left(\frac{w_0 w_1}{w_2^2} - 1\right).$$

The gravitational acceleration g is a known physical constant here. Both friction coefficients can be recovered as $D_p = w_5$ and $D_r = w_4 - \gamma k_m$. After all parameters have been identified, system trajectories can be obtained by simulating the forward dynamics

$$\ddot{\boldsymbol{q}} = \boldsymbol{M}^{-1}(\boldsymbol{q})\left(\boldsymbol{\tau} - \boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}})\right).$$

If one finds a distribution over $\boldsymbol{w}$, e.g., by performing Bayesian ridge regression, then one can sample from this distribution and simulate an ensemble of trajectories. Exact analytic uncertainty propagation is not possible here even if the distribution is Gaussian because the forward dynamics are highly nonlinear in the parameters.

Note that if we would assume k_m or R_m to be unknown, the solution of (1) would not be unique, i.e., every coefficient in (2) could be scaled by γ which is a variable in the context of domain randomization.

C Overview of Likelihood-Free Inference Approaches

Table A1 summarizes state-of-the-art Likelihood-Free Inference (LFI) approaches with focus on novel methods which use neural density estimators (Section 5.3). For a comprehensive survey on LFI from simulations, we refer to [42].

Table A1: List of LFI approaches. Here, $p(\boldsymbol{\xi}|\boldsymbol{x})$ denotes the true posterior, $d(\boldsymbol{x}, \boldsymbol{x}^{\text{obs}})$ a distance measure between the query data $\boldsymbol{x}$ and the observed data $\boldsymbol{x}^{\text{obs}}$, $\tilde{p}(\boldsymbol{\xi}|\boldsymbol{x})$ the proposal posterior, $q(\boldsymbol{\xi}|\boldsymbol{x})$ the approximate posterior, and $p(\boldsymbol{x}|\boldsymbol{\xi})$ the likelihood. We use the acronyms Sequential Neural Posterior Estimation (SNPE), Sequential Neural Likelihood Estimation (SNLE), Sequential Neural Ratio Estimation (SNRE), Mixture Density Network (MDN), Mixture of Gaussians (MoG), and Masked Autoregressive Flow (MAF). The approach presented in this paper employs SNPE-C, but could have used any of the others, too.

Algorithm	**Estimated Density**	**Model**
ABC	$p(\boldsymbol{\xi}\|(d(\boldsymbol{x}, \boldsymbol{x}^{\text{obs}})) < \varepsilon)$	empirical (e.g. MCMC)
SNPE-A [8]	$\tilde{p}(\boldsymbol{\xi}\|\boldsymbol{x})$	MDN (e.g. MoG)
SNPE-B [9]	$q(\boldsymbol{\xi}\|\boldsymbol{x})$	MDN (e.g. MoG)
SNPE-C a.k.a. APT [10]	$q(\boldsymbol{\xi}\|\boldsymbol{x})$	MAF
SNLE [11]	$p(\boldsymbol{x}\|\boldsymbol{\xi})$	MAF
SNRE-A a.k.a. AALR-MCMC [12]	$p(\boldsymbol{\xi}\|\boldsymbol{x})/p(\boldsymbol{\xi}) = p(\boldsymbol{x}\|\boldsymbol{\xi})/p(\boldsymbol{x})$	classifier (e.g. ResNet)
SNRE-B a.k.a. SRE [13]	$p(\boldsymbol{\xi}\|\boldsymbol{x})/p(\boldsymbol{\xi}) = p(\boldsymbol{x}\|\boldsymbol{\xi})/p(\boldsymbol{x})$	classifier (e.g. ResNet)

D Additional Results

In the following, we present supplementary results for the swing-up and balance task on the Furuta pendulum (Section 4.3). We investigate samples from the learned posteriors, the importance of the density estimator model, and the number of rollouts for the inference subroutine in NPDR.

D.1 Domain Parameter Posteriors for the Sim-to-Real Experiment on the Furuta pendulum

Similar to the mini golf experiment, Figure A1 shows the approximated domain parameter posterior $\hat{p}\left(\boldsymbol{\xi}|\boldsymbol{x}=\boldsymbol{x}^{\text{obs}}\right)$ for the swing-up and balance task. The condition $\boldsymbol{x}^{\text{obs}}$ is a set of 5 common real-world trajectories, and the displayed posteriors are the ones reported in Table 2.

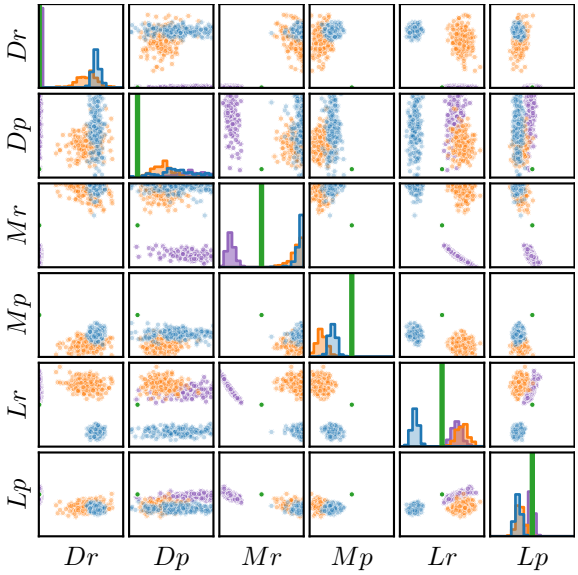

Figure A1: A 6-dimensional slice from the 9-dimensional posteriors learned with NPDR (blue), BayesSim (orange), and Bayesian linear regression (purple) in the Furuta pendulum experiment. The nominal values (green), were either determined by prior measurements or by coarse estimates. Every domain off-diagonal plot shows the same 200 samples for 2 dimensions, whereas the diagonal plots show the marginal distributions. The domain parameters k_m, R_m, and g were omitted since they can not be handled by Bayesian linear regression (Section B). Moreover, for this baseline the Mp and Mr samples are not visible since all posterior samples lie outside the prior range of NPDR and BayesSim. Details on the domain parameters ranges can be found in Table A5.

D.2 On the Influence of the Density Estimator Model for NPDR

We carried out an ablation study on a physical Furuta pendulum to assess the importance of the inference procedure and the density estimator model. For that, we repeat the Bayesian system identification from Section 4.3 using SNPE-C and MoGs with 10 mixture components. The results listed in Table A2 highlight that NPDR works well with either MAFs and MoGs which suggests that the inference procedure is pivotal for the performance on the swing-up task.

Table A2: Performances of the Bayesian system identification for different inference methods and density estimators. The metrics quantify how well each approach fits a common ground truth data set of trajectories recorded on a physical Furuta pendulum. For every configuration (column), we report the mean and standard deviation of 1000 domain parameter samples from 5 distinct experiments. Each domain parameter sample was evaluated with one (simulated) rollout.

Metric	NPDR (MAF)	NPDR (MoG)	BayesSim (MoG)
DTW dist.	**[1.07 ± 0.03]e+3**	**[1.11 ± 0.07]e+3**	[1.24 ± 0.06]e+3
RMSE	**[2.63 ± 0.04]e−1**	**[2.74 ± 0.21]e−1**	[2.95 ± 0.02]e−1

D.3 On the Influence of the Number of Target Domain Rollouts for NPDR

The influence of the number of target domain rollouts H on the domain parameter posterior is visualized in Figure A2. To study this effect, we chose the sim-to-sim variant of swing-up and balance task on the Furuta pendulum. By conducting this experiment in simulation we were able to minimize side-effects and to sample more target domain rollouts. Figure A2 shows that with an increasing number of trajectories, i.e., a higher dimensional context for the density estimator network, NPDR yields more accurate estimates. For sim-to-real experiments, we need to trade off the number of rollouts we are able or willing to record at every iteration against the accuracy which we demand from our posterior estimate.

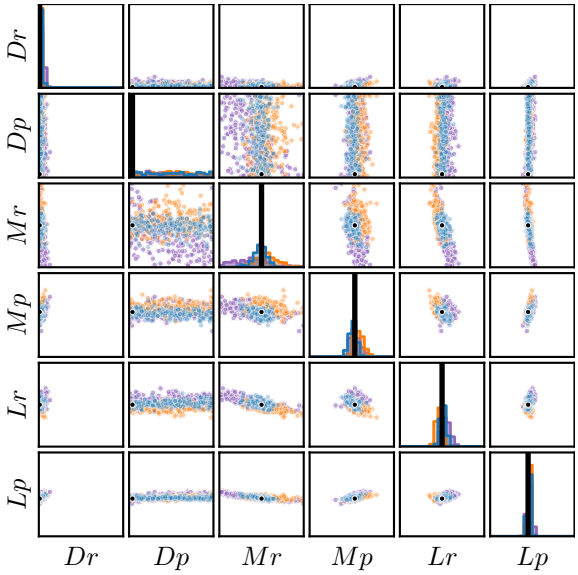

Figure A2: A 6-dimensional slice from the 9-dimensional posteriors learned with NPDR for $H = 1$ (purple), $H = 5$ (orange), and $H = 10$ (blue) target domain rollouts used for the LFI for a sim-to-sim Furuta pendulum experiment. The ground truth domain parameters (black) define a different dynamical system which lies within the prior but is unknown to the inference procedure. Every domain off-diagonal plot shows the same 200 samples for 2 dimensions, whereas the diagonal plots show the marginal distributions. The damping coefficient Dp was difficult to identify for all our experiments, which is most likely due its minor importance for the system dynamics (Dr affects the rotating pole).

E Empirical Discussion on the Computational Costs

The vast majority of the computational cost for NPDR as well as BayesSim originates from running the simulations. Once the simulations are done, the posteriors are fitted. Using the sbi toolbox [20], these weight updates take about 5–300 s for SNPE-C as well as SNPE-A. Importantly, SNPE-C is performing a substantially different multi-round inference than SNPE-A, meaning it updates the posterior multiple times on the same set of simulations (unlike SNPE-A). This leads to a notable reduction of the required number of simulations. Eventually, NPDR and BayesSim take approximately the same wall clock time, measured on a desktop PC: 20 min (pendulum) 3.5 h (mini golf), and 8 h (Furuta pendulum). Most of the required time for the experiment on the Furuta comes from the policy optimization and the interaction with the physical device.

F Parameter Values for the Experiments

The Tables A3 to A5 list the hyper-parameters for all training runs during the experiments in Section 4. Note that the physics engine multiplies the rolling friction parameter μ_b with the (local) curvature of the associated body's shape before applying it.

Table A3: Configuration for the simulated pendulum experiment in Section 4.1

Hyper-parameter	Value
common	
prior range m_p	[0.3, 1.7] kg
prior range l_p	[0.3, 1.7] m
behavioral policy π	$\pi(t) = 4.5\sin(2\pi t)$
num. iterations I	1
num. rounds R	3
learning rate inference	5e−4
num. simulations per round N	200
num. target domain rollouts H	5
num. segments	1
time series embedding f	BayesSim embedding [14]
NPDR specific	
density estimator	MAF [41]
num. features	20
num. transformations	2
BayesSim specific	
density estimator	MoG
num. mixture components	5
component perturbation	1e−2

Table A4: Configuration for the mini golf experiment in Section 4.2

Hyper-parameter	Value
common	
prior range r_b	[0.014, 0.026] m
prior range m_b	[2.5, 7.5] g
prior range e_b	[0, 1] 1
prior range μ_b	[0, 5e−4] m
prior range Δx_1	[−0.08, 0.08] m
prior range Δy_1	[−0.08, 0.08] m
prior range Δx_2	[−0.08, 0.08] m
prior range Δy_2	[−0.08, 0.08] m
prior range $\Delta\gamma_1$	(-π, π) rad
prior range $\Delta\gamma_2$	(-π, π) rad
behavioral policy π	$\pi(t) = \boldsymbol{q}_{\text{init}} + (\boldsymbol{q}_{\text{end}} - \boldsymbol{q}_{\text{init}}) \min(t/t_{\text{end}}, 1)$
num. iterations I	1
learning rate INFER	3e−4
num. target domain rollouts H	2
num. segments	1
NPDR specific	
num. rounds R	7
num. simulations per round N	4e+3
time series embedding f	linear layer with 128 neurons
density estimator	MAF [41]
num. features	100
num. transformations	10
BayesSim specific	
num. rounds R	1 (due to SNPE-A)
num. simulations per round N	2.8e+5
time series embedding f	BayesSim embedding [14]
density estimator	MoG
num. mixture components	20

Table A5: Configuration for the Furuta pendulum experiment in Section 4.3

Hyper-parameter	Value
common	
prior range D_r	[0.0, 1.0e−3] N m s/rad
prior range D_p	[0.0, 1.0e−5] N m s/rad
prior range R_m	[0.3, 1.7] Ω
prior range k_m	[0.3, 1.7] V s/rad
prior range M_r	[0.66e−1, 1.2e−1] kg
prior range M_p	[1.68e−2, 3.12e−2] kg
prior range L_r	[0.43e−1, 1.28e−1] m
prior range L_p	[0.65e−1, 1.94e−1] m
prior range g	[8.34, 11.28] m/s^2
policy $\pi_{\boldsymbol{\theta}}$	hybrid controller: energy-based + PD (7 parameters)
policy optimization POLOPT	PoWER
num. importance samples POLOPT	10
population size POLOPT	25
learning rate INFER	3e−4
num. target domain rollouts H	5
len. segments	200 steps
time series embedding f	linear layer with 256 neurons
NPDR specific	
density estimator	MAF [41]
num. rounds R	5
num. simulations per round N	1000
num. features	50
num. transformations	5
num. iterations POLOPT	20
num. iterations I	1
online BayesSim specific	
num. rounds R	1 (due to SNPE-A)
density estimator	MoG
num. mixture components	10
num. simulations per round N	5000
num. iterations POLOPT	1
num. iterations I	3