How to Sim2Real with Gaussian Processes: Prior Mean versus Kernels as Priors

Rika Antonova[‡], Akshara Rai[§], Danica Kragic[‡]
[‡]EECS, KTH, Stockholm, Sweden
[§]Facebook AI Research

***Abstract*—Gaussian Processes (GPs) have been widely used in robotics as models, and more recently as key structures in active learning algorithms, such as Bayesian optimization. GPs consist of two main components: the mean function and the kernel. Specifying a prior mean function has been a common way to incorporate prior knowledge. When a prior mean function could not be constructed manually, the next default has been to incorporate prior (simulated) observations into a GP as 'fake' data. Then, this GP would be used to further learn from true data on the target (real) domain. We argue that embedding prior knowledge into GP kernels instead provides a more flexible way to capture simulation-based information. We give examples of recent works that demonstrate the wide applicability of such kernel-centric treatment when using GPs as part of Bayesian optimization. We also provide discussion that helps to build intuition for why such 'kernels as priors' view is beneficial.**

I. Introduction

Gaussian Processes (GPs) have been utilized in a variety of robotics algorithms, e.g. motion planing [1], active perception [2], [3], manipulation [4], [5] and reinforcement learning for control [6], [7], [8]. GPs have also been the top choice for non-parametric models as part of active learning algorithms, such as Bayesian optimization (BO). BO allows executing a set of trials/trajectories and helps decide how to adjust control parameters to improve performance with respect to a given black-box cost. BO has been used for optimizing controllers for a variety of hardware tasks, such as locomotion for AIBO quadruped [9], snake [10], bipeds [11], as well as manipulation tasks like grasping [12], [13], pushing [14]. BO is particularly promising for Sim2Real, since it provides a data-efficient way to learn from hardware trials. However, early BO experiments on hardware mostly involved optimizing low-dimensional controllers. To scale up, BO needs to incorporate prior knowledge. We discuss two main paths for achieving this. One way is to use hand-constructed prior mean functions or add 'fake' observations from simulation to shape the prior mean. The other way is to build kernels from simulation that reshape the search space of BO. In the following sections we first give a brief explanation of the BO algorithm, then give examples and analysis of approaches that incorporate simulation in the mean- vs kernel-centric way. We conclude by giving intuition as to why re-shaping the search space helps BO for Sim2Real.

II. Background: Gaussian Processes in BO

In BO, the problem of optimizing controllers is viewed as finding controller parameters $\boldsymbol{x}^*$ that optimize some objective function $f(\boldsymbol{x})$: $f(\boldsymbol{x}^*) = \max_{\boldsymbol{x}} f(\boldsymbol{x})$. At each optimization trial BO optimizes an auxiliary 'acquisition' function to select the next promising $\boldsymbol{x}$ to evaluate. f is commonly modeled with a Gaussian process (GP): $f(\boldsymbol{x}) \sim \mathcal{GP}(m(\boldsymbol{x}), k(\boldsymbol{x}_i, \boldsymbol{x}_j))$. Model prior/posterior of f with a GP gives a way to compute posterior mean $\bar{f}(\boldsymbol{x})$ and variance/uncertainty $Var[f(\boldsymbol{x})]$ for each candidate test point $\boldsymbol{x}$. Hence, the acquisition function can select points to balance high mean (exploitation) and high uncertainty (exploration). The kernel function $k(\cdot,\cdot)$ encodes similarity between inputs. If $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is large for inputs $\boldsymbol{x}_i, \boldsymbol{x}_j$, then $f(\boldsymbol{x}_i)$ strongly influences $f(\boldsymbol{x}_j)$. One of the most widely used kernel functions is the Squared Exponential (SE) kernel: $k_{SE}(\boldsymbol{r} \equiv |\boldsymbol{x}_i - \boldsymbol{x}_j|) = \sigma_k^2 \exp\big(-\frac{1}{2}\boldsymbol{r}^T \operatorname{diag}(\boldsymbol{\ell})^{-2}\boldsymbol{r}\big)$, where σ_k^2, $\boldsymbol{\ell}$ are signal variance and a vector of length scales respectively. σ_k^2, $\boldsymbol{\ell}$ are called 'hyperparameters' and are optimized automatically by maximizing marginal likelihood. SE belongs to a broader class of Matérn kernels. These kernels are stationary, since they depend on $\boldsymbol{r} \equiv \boldsymbol{x}_i - \boldsymbol{x}_j \ \forall \boldsymbol{x}_{i,j}$, and not on individual $\boldsymbol{x}_i, \boldsymbol{x}_j$. See [15] for details. Stationarity allows avoiding commitment to domain-specific assumptions, which helps generality, but can be detrimental do data efficiency.

III. Informing Prior Mean vs Kernels

Informing Prior Mean: A classic book on GPs for machine learning [16] gives advice on shaping the prior mean function (Section 2.7). It shows that incorporating a fixed deterministic mean function is straightforward and also gives examples of how to express a prior mean as a linear combination of a given set of basis functions. This approach has been used as early as 1975, e.g. with polynomial features $h(\boldsymbol{x}) = (1, \boldsymbol{x}, \boldsymbol{x}^2, ...)$ [17].

Modern approaches seek more flexibility. One direction is to initialize GPs with points from simulated trials directly. This can be formulated as a multi-fidelity problem, with different fidelities for simulated vs real points [18], [19]. The main issue is that one needs to carefully weigh the contributions from simulated vs real trials, since 'fake' data from inaccurate simulations can overwhelm the effects of the real data. This can be done if simulation fidelity is known, but is more challenging otherwise. Another issue arises if simulation is cheap and the number of simulated/fake points is too large to be handled by exact GPs. Sparse GPs can be used in such cases ([20], [21] implement several versions), however this may cause loss in precision due to approximate inference. In [22, Section 5.3] we illustrate the effects of simulation fidelity on such 'cost prior' formed by adding 35K simulated points to a Sparse GP. We use a high-fidelity simulator of a bipedal robot as a surrogate

for reality, and show results of BO with 'cost priors' from 3 levels of fidelity. For high and medium simulator fidelity we observe significant gains over BO with zero-mean prior. However, for low fidelity the result is worse than baseline BO.

Informing Kernels: [23] proposed to combine a simple 'cost prior' with a kernel-centric method. They collected best performing points in simulation and searched among these points using a domain-specific behavior metric. Using the metric was akin to defining a custom function to express similarities between controllers i.e. a kernel function supported on a limited set of points. They showed excellent results on BO for hexapod recovering from hardware damage, but did not investigate the effects of simulation fidelity. We adapted [23] to bipedal locomotion and compared results when using 3 different simulation fidelities [22, Section 5.3] (using simulator with the highest fidelity as as surrogate for reality). For high and medium fidelities we saw significant gains both with the original ('cost prior'+kernel) method and a kernel-only variant. With low fidelity the gains were small. Moreover, the final performance of the kernel-only variant was similar to the original method, i.e. no further benefit from 'cost prior'. The kernel in [23] is constrained by the fact that only pre-selected points are included. We showed that it was possible to significantly strengthen a kernel-only approach. We achieved this by letting all simulated points influence kernel similarities instead of selecting an 'elite' subset, and by learning to dynamically adjust to simulation-hardware mismatch. Our further experiments in [22, Section 5.3] showed large improvements for BO even with a kernel constructed using the low-fidelity simulator. The benefits of kernel-based approaches can be extended even further by decoupling the effects of simulation-based and hardware-based kernels [24]. We investigated the effects of degrading the kernels, until the quality was bad enough to cause negative transfer. The approach in [24] was able to recover and significantly outperform baseline BO even in this case. These later experiments were conduced on hardware (ABB Yumi robot performing task-oriented grasping).

To summarize: kernel-based approaches can offer robustness against sim2real mismatch and can provide benefits even when low-fidelity simulators are used to construct them. Kernel-centric view is especially relevant for cross-task transfer and lifelong learning, since kernel-based approaches can avoid including any task- or cost-specific information. The learning community expressed interest in the kernel-centric view, giving significant attention to [25], [26]. However, originally these approaches did not include a data-efficient way to handle large sim2real mismatch. Our later work offered one solution with increased modularity and data efficiency [27]. We hope to motivate further interest in this area and inspire extensions to kernel-centric sim2real approaches in various areas of robotics.

IV. Parametric vs Intrinsic Dimensionality

We showed experimental evidence that kernel-centric approaches can be made data-efficient and robust to sim2real mismatch. However, it might still seem puzzling as to why shaping the search space with kernel-based methods can yield ultra data-efficient search even with higher-dimensional controllers (e.g. 30D+). Such puzzlement usually does not arise when we think of 'cost priors', since it is easy to imagine that we could sample a number of successful points in simulation. When these points are added as 'fake' data they very clearly re-shape the posterior mean, so we would likely sample close to these successful points in the first few hardware trials. But in the kernel-centric approaches it may seem that we are starting from scratch. Here, we aim to give intuition regarding where the benefits of kernel-centric approaches come from.

One easy case is a kernel that projects inputs $\boldsymbol{x} \in \mathbb{R}^N$ to a low-dimensional space e.g. $k(\phi(\boldsymbol{x}), \phi(\boldsymbol{x}')), \phi(\boldsymbol{x}) \in \mathbb{R}^n, n \ll N$. But what if we do not restrict dimensionality? To build intuition, let's look at a basic case without simulation or advanced kernels. Consider objective/reward functions that come from an arbitrary distribution (we maximize rewards instead of minimizing costs). For BO in 30D we expect to need at least 60 trials to start seeing benefits. However, our reward landscapes are not arbitrary: they come from real-world problems. While robotics problems have a clear parametric dimensionality, their intrinsic dimensionality is usually unknown and could be much lower. The vision community has a similar concept: 'a lower-dimensional manifold of real-world images'. Intrinsic dimensionality of vision problems could be orders of magnitude lower than parametric dimensionality expressed in pixel space.

Consider a 30D quadratic: $f(\boldsymbol{x}) = \sum_i (x_i+1)^2,\ \boldsymbol{x} \in \mathbb{R}^{30}, x_i \in [0,1]$. Even on this simple f BO with SE kernel gives only modest gains for the first 60 trials. Now consider $f_{sm}(\cdot)$ such that a large number of dimensions do not contribute significantly: $f_{sm}(\boldsymbol{x}) = \sum_{i=1}^{3} (x_i+1)^2 + 0.001 \sum_{i=4}^{30} x_i$. Fig. 1 shows that BO needs <15 trials.

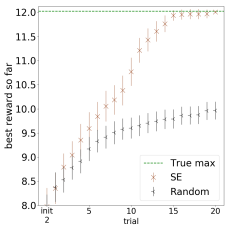

Fig. 1: BO in 30D when only 3 dimensions contribute significantly.

Now consider a class of simulation-informed kernels $k(\phi_{sim}(\boldsymbol{x}), \phi_{sim}(\boldsymbol{x}')), \phi_{sim}(\boldsymbol{x}) \in \mathbb{R}^d, d \approx N$ or even $d > N$. With this, kernel similarities will be computed in the space that only retains aspects relevant to simulation. The aspects of behavior caused by controller $\boldsymbol{x}$ that do not significantly influence ϕ_{sim} are discarded. We obtain a kind of 'compression' that discards information not relevant to simulation. Moreover, strong regularities might arise due to simplifications imposed by simulation modeling limitations. To view this as re-shaping of the search space: 'discarding' can be seen as shrinking of parts of the search space. Instead of using a small coefficient for irrelevant dimensions (e.g. as in f_{sm}), we take the perspective of shrinking irrelevant regions.

Overall, such cases can be viewed as potentially reducing intrinsic dimensionality or complexity without reducing explicit parametric dimensionality. This could also explain why we can obtain significant benefits from highly imprecise simulations. Imprecise simulations can point us in the right direction and reduce the number of samples needed to discover potentially good regions quickly. If care is taken to pay attention to sim2real mismatch: we can exploit this initial boost, then proceed further and rely more on the hardware data.

REFERENCES

[1] M. Mukadam, X. Yan, and B. Boots, "Gaussian process motion planning," in *2016 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2016, pp. 9–15.

[2] N. Jamali, C. Ciliberto, L. Rosasco, and L. Natale, "Active perception: Building objects' models using tactile exploration," in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2016, pp. 179–185.

[3] S. Caccamo, Y. Bekiroglu, C. H. Ek, and D. Kragic, "Active exploration using gaussian random fields and gaussian process implicit surfaces," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 582–589.

[4] S. Dragiev, M. Toussaint, and M. Gienger, "Gaussian process implicit surfaces for shape estimation and grasping," in *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 2845–2850.

[5] Z. Hu, P. Sun, and J. Pan, "Three-dimensional deformable object manipulation using fast online gaussian process regression," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 979–986, 2018.

[6] M. P. Deisenroth, D. Fox, and C. E. Rasmussen, "Gaussian processes for data-efficient learning in robotics and control," *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 2, pp. 408–423, 2013.

[7] J. Kober, J. A. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1238–1274, 2013.

[8] A. S. Polydoros and L. Nalpantidis, "Survey of model-based reinforcement learning: Applications on robotics," *Journal of Intelligent & Robotic Systems*, vol. 86, no. 2, pp. 153–173, 2017.

[9] D. J. Lizotte, T. Wang, M. H. Bowling, and D. Schuurmans, "Automatic Gait Optimization with Gaussian Process Regression." in *International Joint Conference on Artificial Intelligence (IJCAI)*, vol. 7, 2007, pp. 944–949.

[10] M. Tesch, J. Schneider, and H. Choset, "Using response surfaces and expected improvement to optimize snake robot gait parameters," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2011, pp. 1069–1074.

[11] R. Calandra, "Bayesian Modeling for Optimization and Control in Robotics," Ph.D. dissertation, Darmstadt University of Technology, Germany, 2017.

[12] O. Kroemer, R. Detry, J. Piater, and J. Peters, "Combining active learning and reactive control for robot grasping," *Robotics and Autonomous systems*, vol. 58, no. 9, pp. 1105–1116, 2010.

[13] L. Montesano and M. Lopes, "Active learning of visual descriptors for grasping using non-parametric smoothed beta distributions," *Robotics and Autonomous Systems*, vol. 60, no. 3, pp. 452–462, 2012.

[14] I. Arnekvist, D. Kragic, and J. A. Stork, "VPE: Variational Policy Embedding for Transfer Reinforcement Learning," in *2019 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2019.

[15] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas, "Taking the Human Out of the Loop: A Review of Bayesian Optimization," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 148–175, 2016.

[16] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005.

[17] B. Blight and L. Ott, "A bayesian approach to model inadequacy for polynomial regression," *Biometrika*, vol. 62, no. 1, pp. 79–88, 1975.

[18] K. Kandasamy, G. Dasarathy, J. Schneider, and B. Póczos, "Multi-fidelity Bayesian Optimisation with Continuous Approximations," in *International Conference on Machine Learning (ICML)*, 2017, pp. 1799–1808.

[19] A. Marco, F. Berkenkamp, P. Hennig, A. P. Schoellig, A. Krause, S. Schaal, and S. Trimpe, "Virtual vs. real: Trading off simulations and physical experiments in reinforcement learning with Bayesian optimization," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 1557–1563.

[20] C. E. Rasmussen and H. Nickisch, "Gaussian processes for machine learning (gpml) toolbox," *J. Mach. Learn. Res.*, vol. 11, pp. 3011–3015, Dec. 2010. [Online]. Available: http://dl.acm.org/citation.cfm?id=1756006.1953029

[21] J. R. Gardner, G. Pleiss, D. Bindel, K. Q. Weinberger, and A. G. Wilson, "Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration," in *Advances in Neural Information Processing Systems*, 2018.

[22] A. Rai, R. Antonova, F. Meier, and C. G. Atkeson, "Using simulation to improve sample-efficiency of bayesian optimization for bipedal robots." *Journal of Machine Learning Research (JMLR)*, vol. 20, no. 49, pp. 1–24, 2019.

[23] A. Cully, J. Clune, D. Tarapore, and J.-B. Mouret, "Robots that can adapt like animals," *Nature*, vol. 521, no. 7553, pp. 503–507, 2015.

[24] R. Antonova, M. Kokic, J. A. Stork, and D. Kragic, "Global Search with Bernoulli Alternation Kernel for Task-oriented Grasping Informed by Simulation," in Conference on Robot Learning (CoRL), 2018, vol. 87. PMLR, 2018 pp. 641–650. Experiments on recovery from negative transfer are described in the last part of CoRL18 talk: video.ethz.ch/events/2018/corl/cc7acaa8-0a91-40ce-a837-e75bbec4848b.html

[25] A. G. Wilson, Z. Hu, R. Salakhutdinov, and E. P. Xing, "Deep kernel learning," in *Artificial Intelligence and Statistics*, 2016, pp. 370–378.

[26] R. Calandra, J. Peters, C. E. Rasmussen, and M. P. Deisenroth, "Manifold gaussian processes for regression," in *2016 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2016, pp. 3338–3345.

[27] R. Antonova, A. Rai, T. Li, and D. Kragic, "Bayesian optimization in variational latent spaces with dynamic compression," in *Proceedings of the Conference on Robot Learning (CoRL), 2019*, vol. 100. PMLR, 2020, pp. 456–465.