GPdoemd: a python package for design of experiments for model discrimination using Gaussian process surrogates

Simon Olofsson SIMON.OLOFSSON15@IMPERIAL.AC.UK
Ruth Misener R.MISENER@IMPERIAL.AC.UK
Department of Computing, Imperial College, London, SW7 2AZ, United Kingdom

Abstract

GPdoemd is an open-source python package for design of experiments for model discrimination that uses Gaussian process surrogate models to approximate and maximise the divergence between marginal predictive distributions of rival mechanistic models. GPdoemd uses the divergence prediction to suggest a maximally informative next experiment.

Keywords: Software, Gaussian processes, design of experiments, model discrimination

Explicit parametric mechanistic models are common in science and engineering, e.g. economics (Black and Scholes, 1973), biology (Mehrian et al., 2018), and control theory (Bemporad et al., 2002). Researchers develop multiple rival mechanistic models, corresponding to different hypotheses about underlying system mechanisms, but typically lack sufficient data to discriminate between the models (Box and Hill, 1967; Buzzi-Ferraris and Forzatti, 1983). To discard inaccurate models, GPdoemd suggests gathering more data through additional experiments. To minimise cost, e.g. money and time, GPdoemd designs optimal new experiments, i.e. experiments yielding maximally informative results.

1. Background

GPdoemd assumes multiple rival mechanistic models f_i, e.g. systems of ODEs, which take as inputs some design variables $\boldsymbol{x}$, e.g. temperature, pressure, flow rate, and model parameters $\boldsymbol{\theta}_i$, e.g. chemical reaction rate. Typically, the model parameters $\boldsymbol{\theta}_i$ are tuned to make the model predictions fit the experimental data $\mathcal{D}$. However, there will be model parameter uncertainty, i.e. $\boldsymbol{\theta}_i \sim p(\boldsymbol{\theta}_i \,|\, \mathcal{D})$, which propagates through to uncertainty in the model predictions. To account for the model parameter uncertainty, we approximate the models' marginal predictive distributions

$$p(f_i(\boldsymbol{x}) \,|\, \mathcal{D}) = \int p(f_i(\boldsymbol{x}) \,|\, \boldsymbol{\theta}_i) p(\boldsymbol{\theta}_i \,|\, \mathcal{D}) \mathrm{d}\boldsymbol{\theta}_i \,. \tag{1}$$

The optimal next experiment $\boldsymbol{x}^*$ is found by maximising a design utility function

$$\boldsymbol{x}^* = \arg\max_{\boldsymbol{x}} U\left\{ f_i(\boldsymbol{x}) \,|\, \mathcal{D} \,;\; i = 1, \ldots, M \right\} \,, \tag{2}$$

where $U\{\cdot\}$ is a divergence measure between the rival models' marginal predictive distributions. The idea is to find the experiment $\boldsymbol{x}^*$ for which the model predictions differ the most, within the error margin imposed by the model parameter uncertainty and measurement noise variance. Equation (2) is typically intractable, so the solution must be approximated.

Typical methods for design of experiments for model discrimination can be divided into two different approaches (Olofsson et al., 2018). First, the analytical approach, e.g. Box and Hill (1967), Buzzi-Ferraris et al. (1990) and Michalik et al. (2010), assumes that the mechanistic models are (approximately) linear in the model parameters, and the model parameter posterior $p(\boldsymbol{\theta}_i \,|\, \mathcal{D})$ is a multivariate Gaussian distribution. The analytical approach is computationally efficient but requires gradient information that may not be readily available in complex models or legacy code. The second approach is the data-driven, i.e. Monte Carlo-based, approach, e.g. Vanlier et al. (2014) and Ryan et al. (2015), where samples are drawn from the model parameter posterior in order to compute the optimal next experiment. No gradient information is required, but the data-driven approach can be computationally very expensive due to the number of samples required for large $\boldsymbol{x}$ and $\boldsymbol{\theta}_i$ search spaces.

Olofsson et al. (2018) hybridise the analytical and data-driven approaches by training Gaussian process (GP) surrogate models on sampled data and using the surrogate models in an analytical fashion. The GPdoemd open-source software package implements this hybrid approach. The hybrid approach computes $p(f_i(\boldsymbol{x}) \,|\, \mathcal{D}) = \mathcal{N}(\breve{\boldsymbol{\mu}}, \breve{\boldsymbol{\Sigma}})$ by placing a GP prior $f_i \sim \mathcal{GP}(0, k_x k_\theta)$ on the original mechanistic models f_i, assuming a Gaussian model parameter distribution $\boldsymbol{\theta} \,|\, \mathcal{D} \sim \mathcal{N}(\boldsymbol{\theta}^*, \boldsymbol{\Sigma}_\theta)$, and then using closed-form approximations to compute the first two moments $\breve{\boldsymbol{\mu}}$ and $\breve{\boldsymbol{\Sigma}}$ of (1). Design utility functions $U\{\cdot\}$ from classical literature, which are closed-form for Gaussian distributions, find the optimal next experiment.

2. Implementation

GPdoemd, available from the GitHub repository `https://github.com/cog-imperial/GPdoemd`, is a python package implementing the Olofsson et al. (2018) hybrid approach. It uses functionality from the GPy (since 2012) python package for GP inference. The only other dependencies are on the standard numpy (v1.7+) and scipy (v0.17+) packages. GPdoemd is regularly tested for python v3.4 and up. On GitHub, we provide documentation for installing and using GPdoemd via a PDF and Jupyter notebook demonstrations.

The modular toolbox, illustrated in Figure 1, offers a choice between different GP kernel functions, inference methods, methods to approximate the marginal predictive distributions, design utility functions and model discrimination criteria. New modules can be easily implemented and added to the GPdoemd toolbox. The toolbox currently comes with the Table 1 case studies which are mostly from literature, although collaborators developed `mixing`. Researchers and engineers may try the Olofsson et al. (2018) hybrid approach and compare the performance to competing methods.

3. Syntax and Supported Features

Assuming the rival models have been proposed, GPdoemd assists in model discrimination.

Model type A model object is initialised using a python dictionary containing the model name (`name`), the model function $f(\boldsymbol{x}, \boldsymbol{\theta}_i)$ handle (`call`), the design variable and model parameter dimensions (`dim_x` and `dim_p`), the number of target dimensions (`num_outputs`), model parameter bounds (`p_bounds`), experimental measurement noise (co)variance (`meas_noise_var`), and a list of discrete design variable dimensions (`binary_variables`). This dictionary is passed to one of the implemented model types (Box 1 in Figure 1).

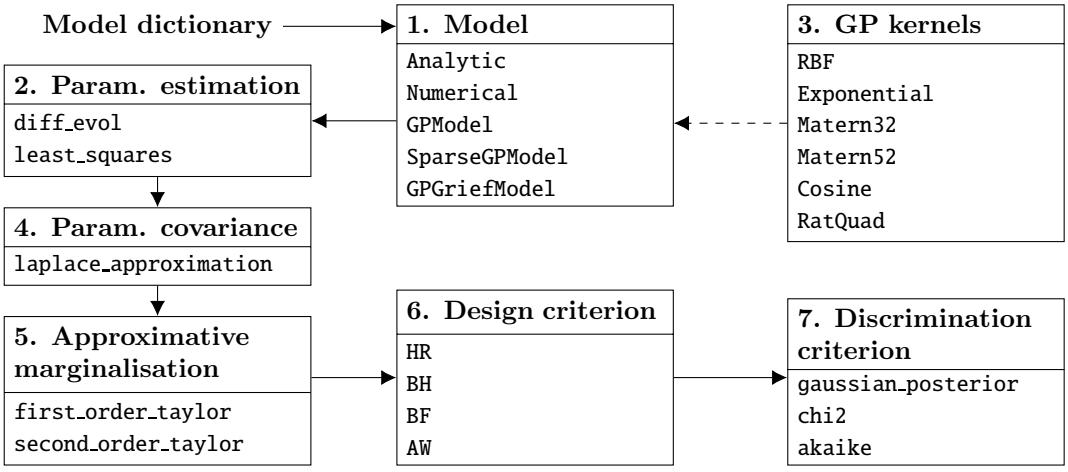

Figure 1: The modular structure of the GPdoemd open-source software package.

Table 1: GPdoemd case studies, with the number of design variables $|\boldsymbol{x}|$ (continuous or discrete), model parameters $|\boldsymbol{\theta}_i|$, target dimensions $|\boldsymbol{y}|$ and rival models M.

Name	Reference	$\lvert\boldsymbol{x}\rvert$, ($\in \mathbb{Z}$)	$\lvert\boldsymbol{\theta}_i\rvert$	$\lvert\boldsymbol{y}\rvert$	M	Type
`bff1983`	Buzzi-Ferraris and Forzatti (1983)	3, (0)	5	1	5	Analytic
`bffeh1984`	Buzzi-Ferraris et al. (1984)	2, (0)	4	2	4	Analytic
`bffc1990a`	Buzzi-Ferraris et al. (1990)	3, (0)	2–6	1	4	Analytic
`mixing`	-	3, (1)	1	1	5	Analytic
`msm2010`	Michalik et al. (2010)	3, (0)	1	1	10	Analytic
`vthr2014linear`	Vanlier et al. (2014)	1, (0)	2–4	1	4	Analytic
`vthr2014ode`	Vanlier et al. (2014)	3, (2)	14	1	4	Black-box
`tandogan2017`	Tandogan et al. (2017)	4, (0)	8–14	2	3	Black-box

Parameter estimation Given experimental data `Ydata` for designs `Xdata`, GPdoemd optimises the model parameter values $\boldsymbol{\theta}^*$ using a prediction error minimisation methods (Box 2 in Figure 1): differential evolution (`diff_evol`) or least squares with finite difference gradient approximation (`least_squares`). Both optimisation methods call functions in `scipy`.

GP kernels The GP surrogate models require a choice of GP kernel functions k_x and k_θ for the GP prior $\mathcal{GP}(0, k_x k_\theta)$. GPdoemd currently supports six different kernel functions (Box 3 in Figure 1) taken from the `GPy` package and extended with function calls for the second derivatives $\partial^2 k(r)/\partial r^2$ with respect to the distance measure r.

Model parameter covariance ($\boldsymbol{\Sigma}_\theta$) GPdoemd uses a Gaussian approximation $\mathcal{N}(\boldsymbol{\theta}^*, \boldsymbol{\Sigma}_\theta)$ of the model parameter distribution and implements a Laplacian approximation of $\boldsymbol{\Sigma}_\theta$.

Approximating marginal predictive distributions The Olofsson et al. (2018) hybrid approach approximates the marginal predictive distribution in (1) with a Gaussian distribution. The GPdoemd implements two different methods of computing the first two moments of (1): first- and second-order Taylor approximations (Box 5 of Figure 1), using the first and second derivatives of the GP surrogate's predictive mean and variance.

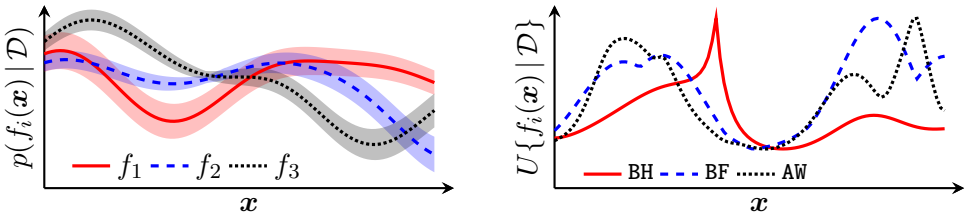

Figure 2: (Left) Model predictive distributions over designs $\boldsymbol{x}$ for rival models f_1, f_2, f_3. (Right) Three design utility functions for the model predictive distributions on the left. Note that the different utilities, i.e. Eq. (2), prefer different next experiments.

Design criterion GPdoemd provides five different utility functions (Box 6 in Figure 1) for designing the next experiment: `HR` (Hunter and Reiner, 1965), `BH` (Box and Hill, 1967), `BF` (Buzzi-Ferraris et al., 1990) and `AW` (Michalik et al., 2010). Figure 2 illustrates the difference between a few of these design criteria.

Discrimination criterion GPdoemd provides three different criteria (Box 7 in Figure 1) for model discrimination: normalised Gaussian posteriors (Box and Hill, 1967), χ^2 test (Buzzi-Ferraris and Forzatti, 1983), and the Akaike information criterion weights (Michalik et al., 2010). Olofsson et al. (2018) discuss trade-offs between the discrimination criteria.

3.1 Example

Assume a list `dlist` of model dictionaries, experimental data `Xdata`, `Ydata` with measurement noise variance `measvar`, and lists `X`, `P` and `Y` of surrogate model training data (design, model parameters and predictions, respectively) are given. We wish to select the optimal next experiment from candidates `Xnew`.

```
N = Xnew.shape[0]       # Number of test points
M = len( dlist )        # Number of rival models
E = Ydata.shape[1]      # Number of target dimensions
mu, s2 = np.zeros(( N, M, E )), np.zeros(( N, M, E, E ))
for i,d in enumerate( dlist ):
    m = GPdoemd.models.GPModel(d) # Initialise surrogate model
    # Estimate model parameter values
    m.param_estim(Xdata, Ydata, GPdoemd.param_estim.least_squares, m.p_bounds)
    # Set-up surrogate model
    RBF = GPdoemd.kernels.RBF
    m.gp_surrogate(Z=np.c_[X[i], P[i]], Y=Y[i], kern_x=RBF, kern_p=RBF)
    m.gp_optimise()
    # Approximate model parameter covariance
    m.Sigma = GPdoemd.param_covar.laplace_approximation( m, Xdata )
    # Approximate marginal predictive distribution at test points
    mu[:,i], s2[:,i] = GPdoemd.marginal.taylor_first_order( m, Xnew )
dc = GPdoemd.design_criteria.AW(mu, s2, measvar) # Design criterion at test points
xnext = Xnew[ np.argmax(dc) ]                    # Optimal next experiment
```

Acknowledgments

This work has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement no.675251, and an EPSRC Research Fellowship to R.M. (EP/P016871/1).

References

A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos. The explicit linear quadratic regulator for constrained systems. *Automatica*, 38:3–20, 2002.

F. Black and M. Scholes. The pricing of options and corporate liabilities. *J Political Econ*, 81(3):637–354, 1973.

G. E. P. Box and W. J. Hill. Discrimination among mechanistic models. *Technometrics*, 9 (1):57–71, 1967.

G. Buzzi-Ferraris and P. Forzatti. A new sequential experimental design procedure for discriminating among rival models. *Chem Eng Sci*, 38(2):225–232, 1983.

G. Buzzi-Ferraris, P. Forzatti, G. Emig, and H. Hofmann. Sequential experimental design for model discrimination in the case of multiple responses. *Chem Eng Sci*, 39(1):81–85, 1984.

G. Buzzi-Ferraris, P. Forzatti, and P. Canu. An improved version of a sequential design criterion for discriminating among rival multiresponse models. *Chem Eng Sci*, 45(2): 477–481, 1990.

GPy. GPy: A Gaussian process framework in python. `http://github.com/SheffieldML/GPy`, since 2012.

W. G. Hunter and A. M. Reiner. Designs for discriminating between two rival models. *Technometrics*, 7(3):307–323, 1965.

M. Mehrian, Y. Guyot, I. Papantoniou, S. Olofsson, M. Sonnaert, R. Misener, and L. Geris. Maximizing neotissue growth kinetics in a perfusion bioreactor: An in silico strategy using model reduction and Bayesian optimization. *Biotechnol Bioeng*, 115(3):617–629, 2018.

C. Michalik, M. Stuckert, and W. Marquardt. Optimal experimental design for discriminating numerous model candidates: The AWDC criterion. *Ind Eng Chem Res*, 49:913–919, 2010.

S. Olofsson, M. P. Deisenroth, and R. Misener. Design of experiments for model discrimination hybridising analytical and data-driven approaches. In *ICML '18: Proceedings of the International Conference on Machine Learning*, Stockholm, Sweden, PMLR 80, 2018.

E. G. Ryan, C. C. Drovandi, and A. N. Pettitt. Fully Bayesian experimental design for pharmacokinetic studies. *Entropy*, 17:1063–1089, 2015.

N. Tandogan, S. Garcia-Muñoz, M. Sen, T. M. Wilson, J. Y. Buser, S. P. Kolis, I. V. Borkar, and C. A. Alt. Use of model discrimination method in drug substance process developments. In *Proceedings of the AIChE Annual Meeting*, pages X–Y, 2017.

J. Vanlier, C. A. Tiemann, P. A. J. Hilbers, and N. A. W. van Riel. Optimal experiment design for model selection in biochemical networks. *BMC Syst Biol*, 8(20), 2014.