tu dortmund university

computational statistics

Simulation-Based Prior Knowledge Elicitation for Parametric Bayesian Models

Florence Bockting
Stefan T. Radev
Paul-Christian Bürkner

Gerd Altmann/geralt

Bayes@Lund 2024

The Challenges of Prior Specification

Translating Expert Knowledge into Priors

- Method development in the area of prior elicitation with the goal to translate expert knowledge into corresponding (valid) prior distributions

florence.bockting@tu-dortmund.de

The Challenges of Prior Specification

Translating Expert Knowledge into Priors

- Method development in the area of prior elicitation with the goal to translate expert knowledge into corresponding (valid) prior distributions
- Is such a method really necessary?
  - Model parameters lack intuitive meaning for domain experts (Albert et al., 2012)
  - Relationship between priors and expert knowledge not apparent (da Silva et al., 2019)
  - Large number of model parameters makes prior construction inefficient (Mikkola et al., 2023)

florence.bockting@tu-dortmund.de

Related Work

Recent Methods for prior elicitation

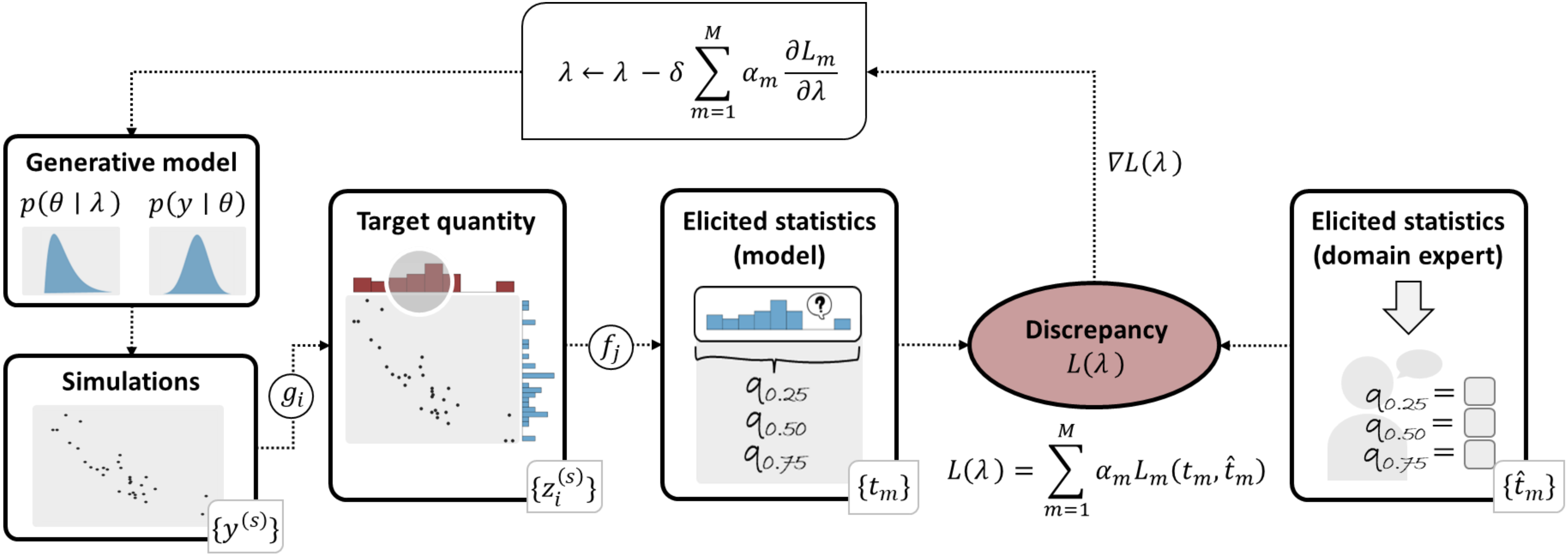
- Reviews: Garthwaite et al. (2005), O'Hagan et al. (2006), Mikkola et al., (2023)
- Historically, methods focused on model parameters
- Recent shift to methods that focus on prior predictive distribution, particularly
  - Da Silva et al. (2019); Hartmann et al. (2020); Manderson & Goudie (2023)

query information from expert	translate →	prior distribution $\boldsymbol{\beta} \sim p_\lambda$
Parameter space		Parameter space
Observable space		Parameter space
Parameter/ Observable space		Parameter space (our method)

florence.bockting@tu-dortmund.de

Prior Elicitation Method with SGD

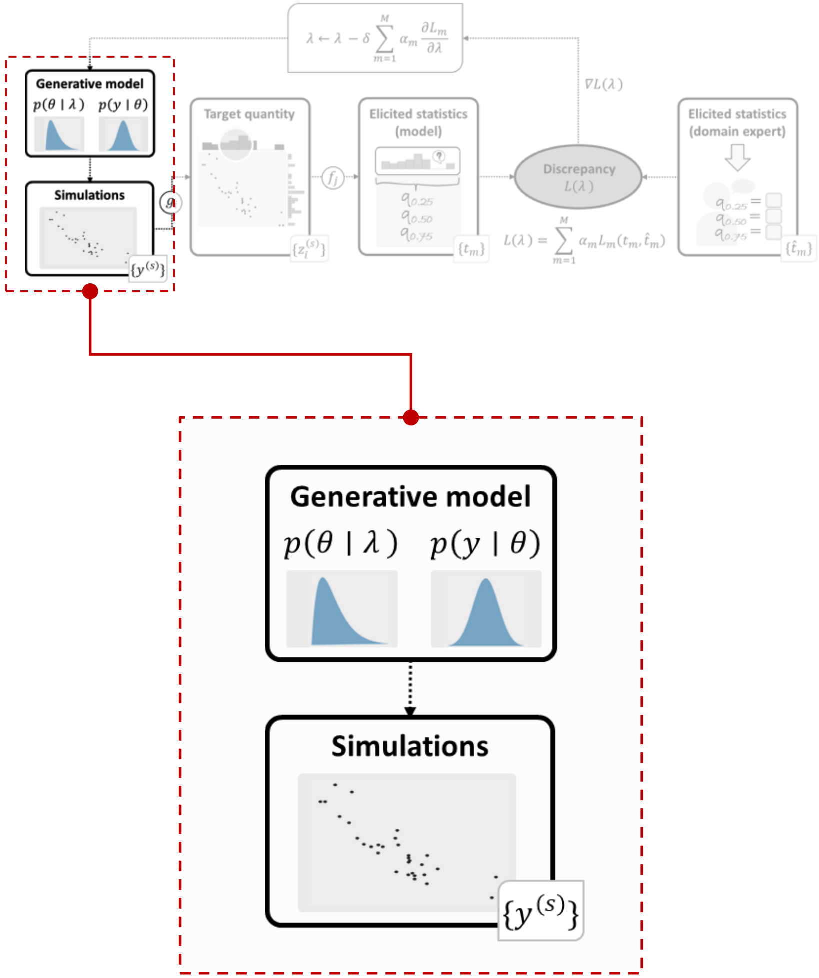
An overview



florence.bockting@tu-dortmund.de

Prior Elicitation Method with SGD

Simulate prior predictions from generative model



Prior Elicitation Method with SGD

Simulate prior predictions from generative model

to be learned

random initialization of hyperparameters

$$\text{RNG} \mapsto \lambda = (\mu_0, \sigma_0, \mu_1, \sigma_1, \mu_2, \sigma_2, \nu)$$

sampling from prior distributions

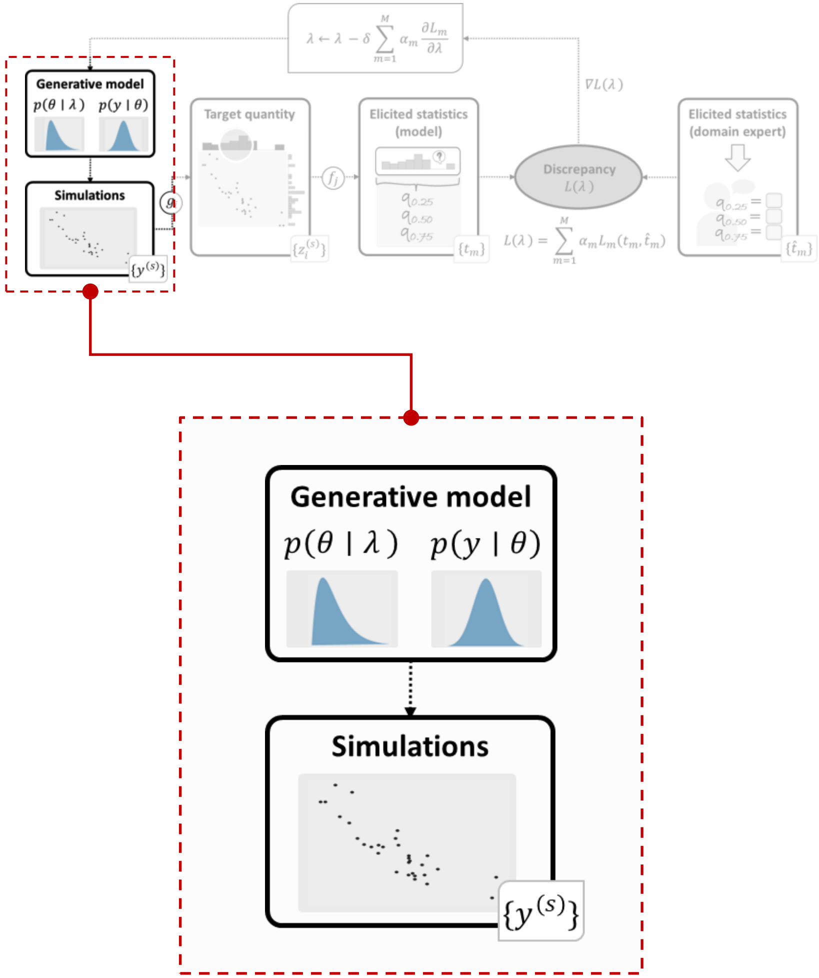
$$\beta_0^{(s)} \sim \text{Normal}(\mu_0, \sigma_0)$$
$$\beta_1^{(s)} \sim \text{Normal}(\mu_1, \sigma_1)$$
$$\beta_2^{(s)} \sim \text{Normal}(\mu_2, \sigma_2)$$
$$s^{(s)} \sim \text{Exponential}(\nu)$$

linear predictor with identity link

$$\theta_i^{(s)} = \beta_0^{(s)} + \beta_1^{(s)} x_{1,i} + \beta_2^{(s)} x_{2,i}$$

sample prior predictions from data model

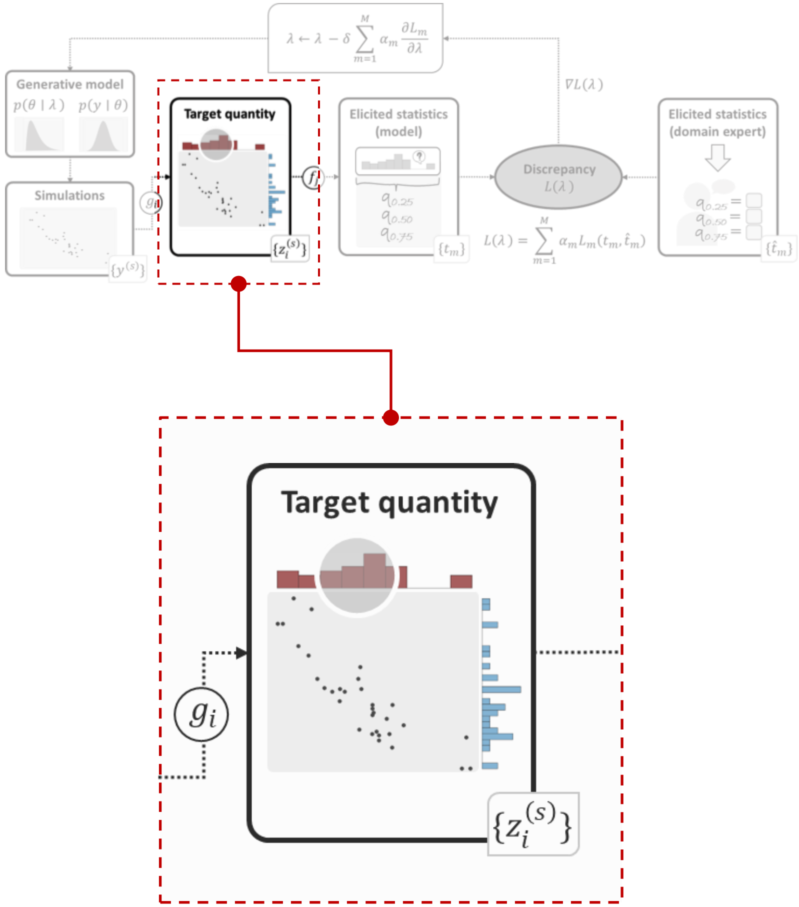
$$y_i^{(s)} \sim \text{Normal}\left(\theta_i^{(s)}, s^{(s)}\right)$$



florence.bockting@tu-dortmund.de

Prior Elicitation Method with SGD

Compute target quantities



Prior Elicitation Method with SGD

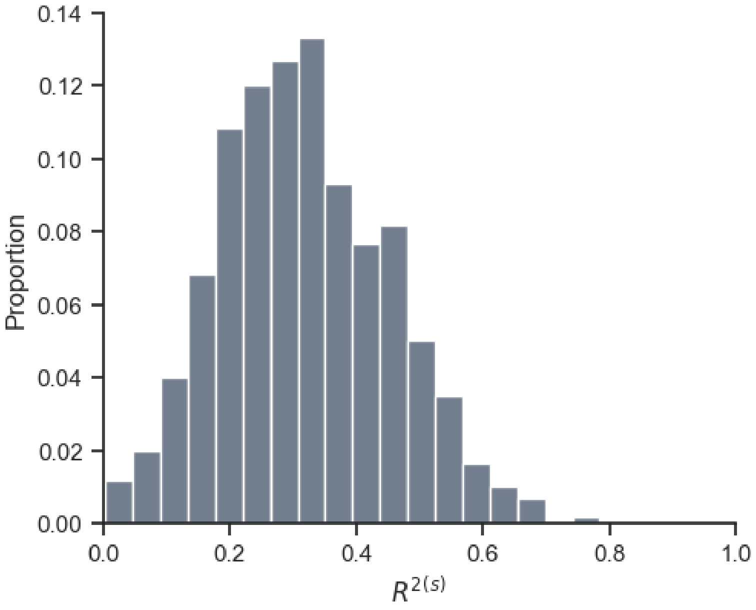
Compute target quantities

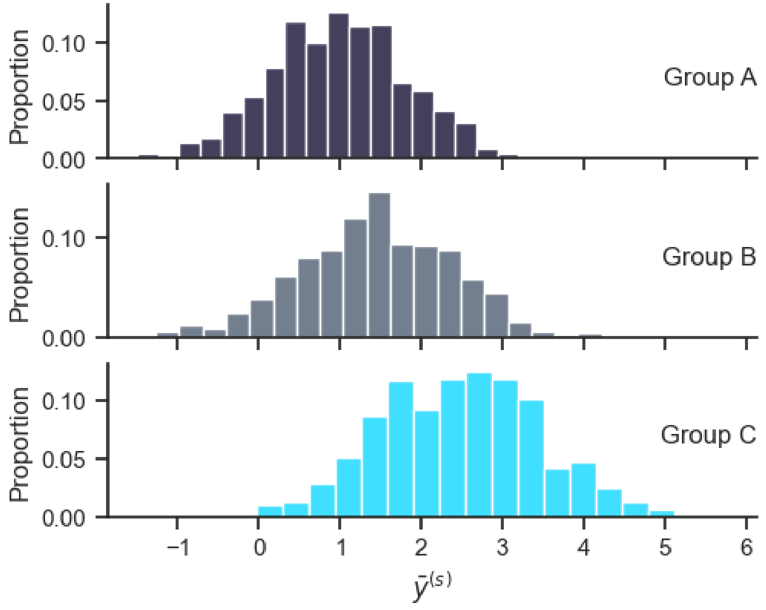
Compute R^2 (here: Gelman et al., 2019)

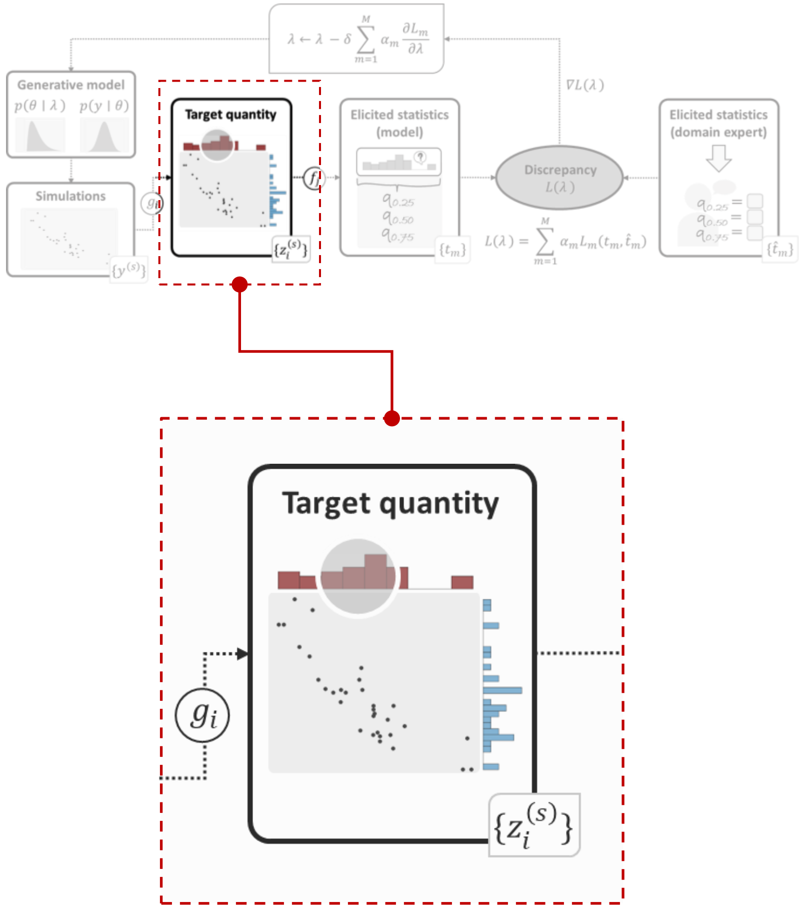
$$R^{2\,(s)} = \frac{\mathrm{Var}\left(\theta_i^{(s)}\right)}{\mathrm{Var}\left(y_i^{(s)}\right)}$$

Compute mean for group A, B, & C

$$\bar{y}_G^{\,(s)} = \frac{1}{|G|}\sum_{i\in G} \mathbb{1}_G\left(y_i^{(s)}\right) \text{ with } G = A, B, C$$

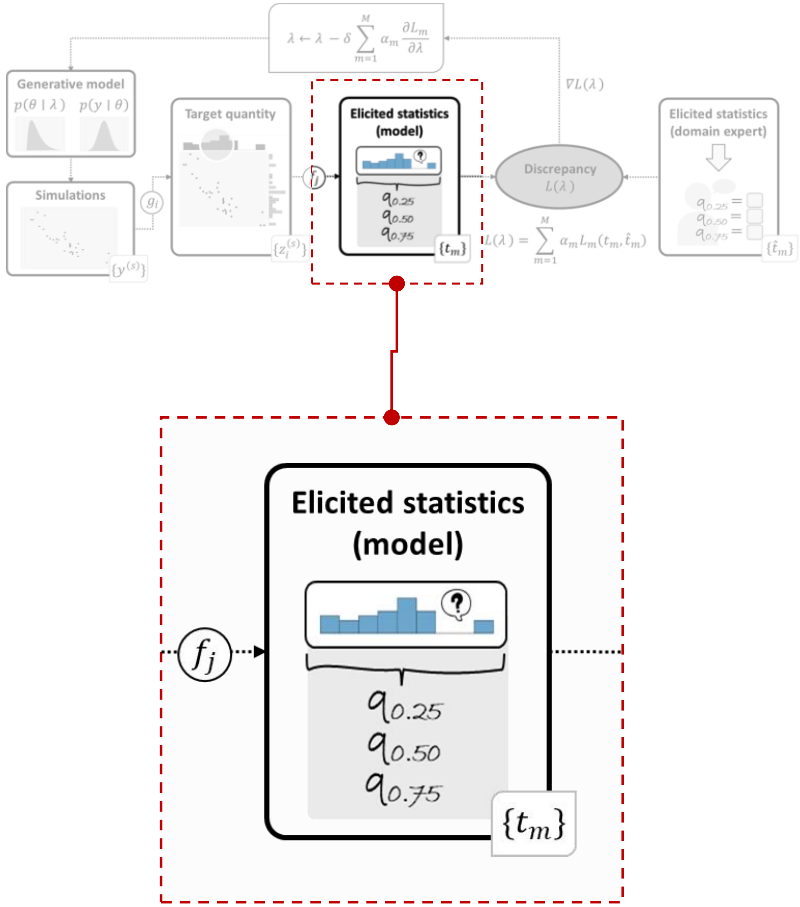






Prior Elicitation Method with SGD

Compute elicited statistics w.r.t. an elicitation technique



Prior Elicitation Method with SGD

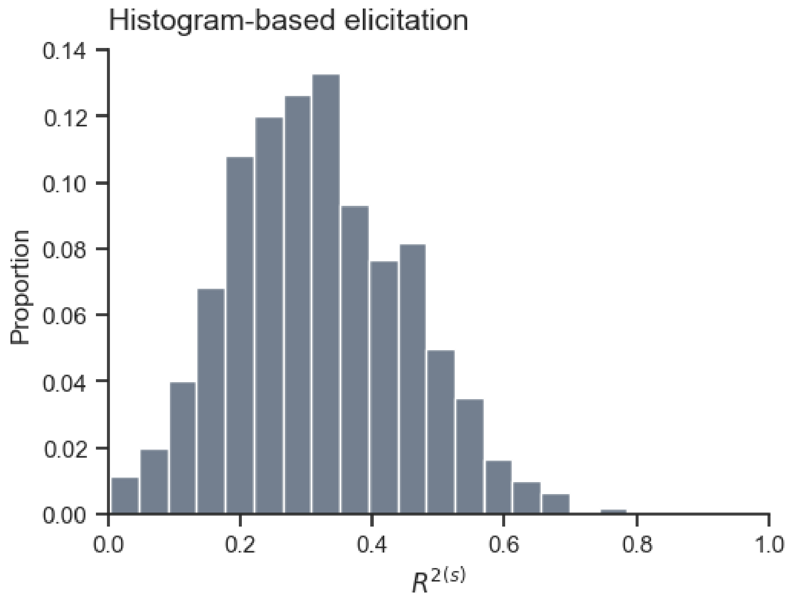
Compute elicited statistics w.r.t. an elicitation technique

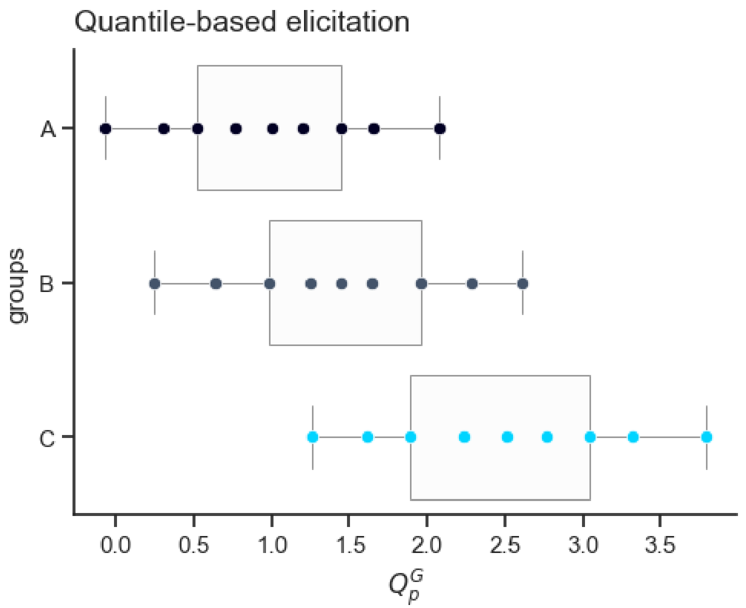
R^2: Histogram-based elicitation

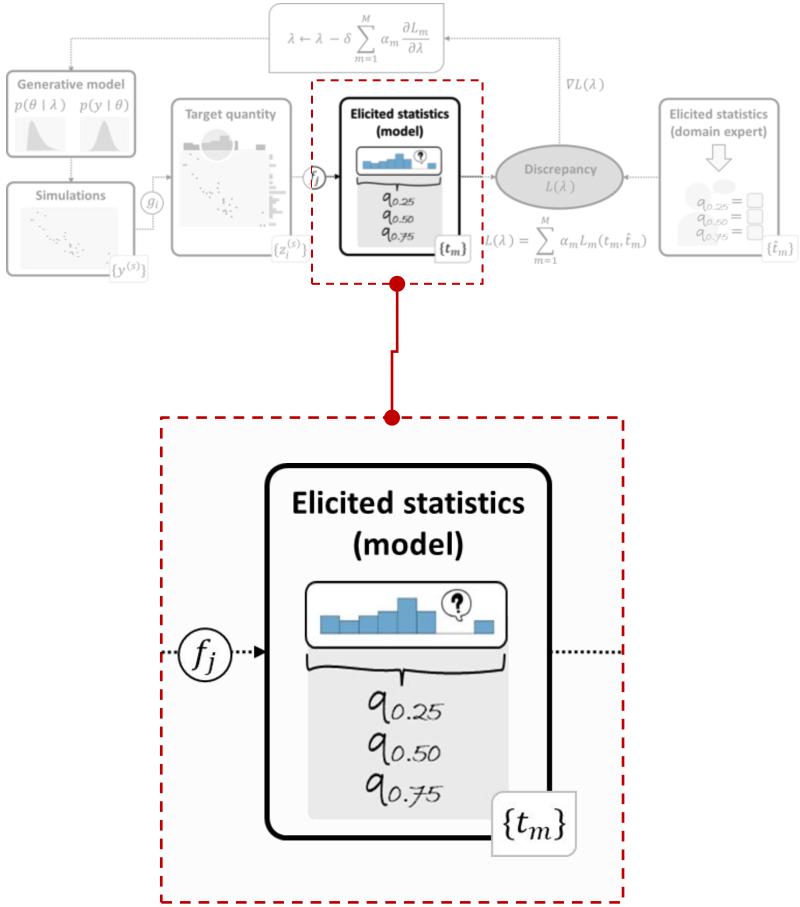
$R^{2^{(s)}}$

Groups A, B, & C: Quantile-based elicitation

$Q_p^G\left(\bar{y}_G^{(s)}\right) = q_{0.1}^G, \ldots, q_{0.9}^G$ with $G = A, B, C$



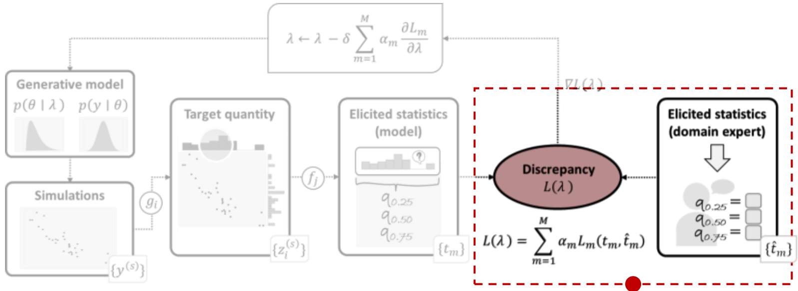


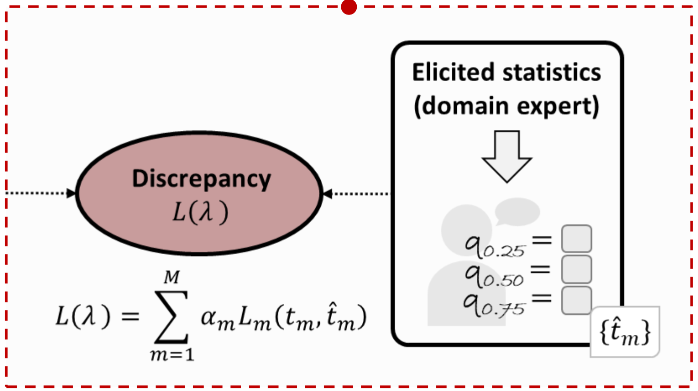


florence.bockting@tu-dortmund.de

Prior Elicitation Method with SGD

Discrepancy between model-implied and expert elicited statistics





Prior Elicitation Method with SGD

Discrepancy between model-implied and expert elicited statistics

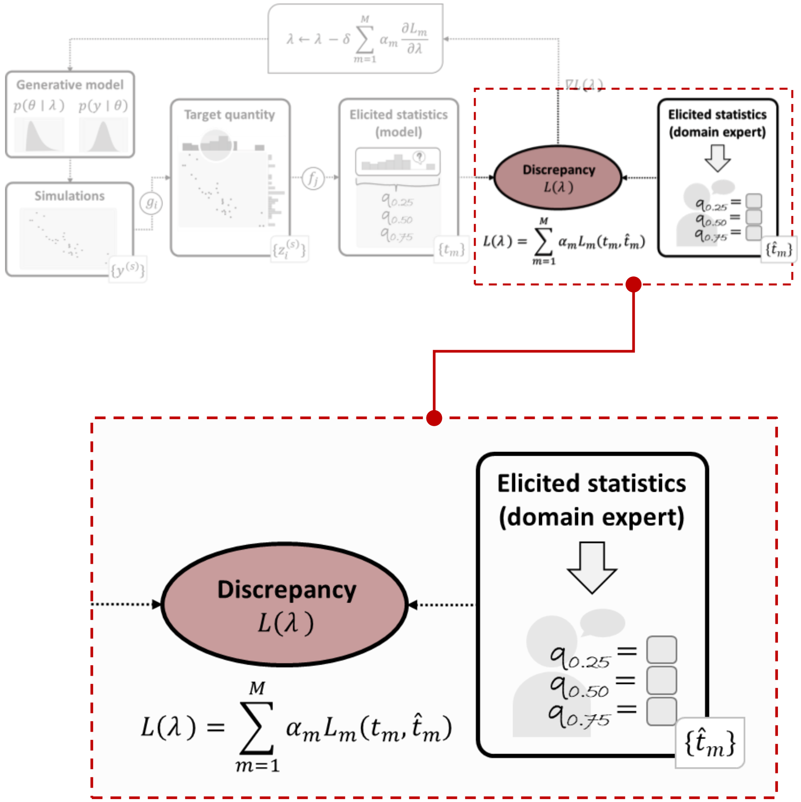
dortmund university

computational statistics

Ideal expert (or „oracle“):

$$\beta_0^{(s)} \sim \text{Normal}(1.0, 0.8)$$
$$\beta_1^{(s)} \sim \text{Normal}(0.5, 0.5)$$
$$\beta_2^{(s)} \sim \text{Normal}(1.5, 0.5)$$
$$s^{(s)} \sim \text{Exponential}(1.0)$$

$$\lambda^* = (1.0, 0.8, 0.5, 0.5, 1.5, 0.5, 1.0)$$



Prior Elicitation Method with SGD

Discrepancy between model-implied and expert elicited statistics

Ideal expert (or „oracle“):

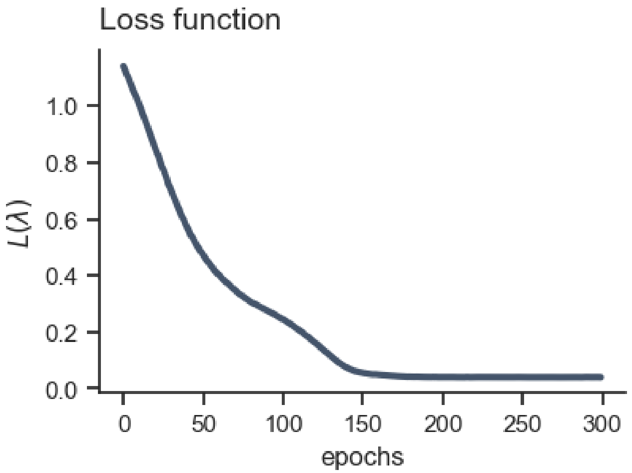
$$\beta_0^{(s)} \sim \text{Normal}(1.0, 0.8)$$
$$\beta_1^{(s)} \sim \text{Normal}(0.5, 0.5)$$
$$\beta_2^{(s)} \sim \text{Normal}(1.5, 0.5)$$
$$s^{(s)} \sim \text{Exponential}(1.0)$$

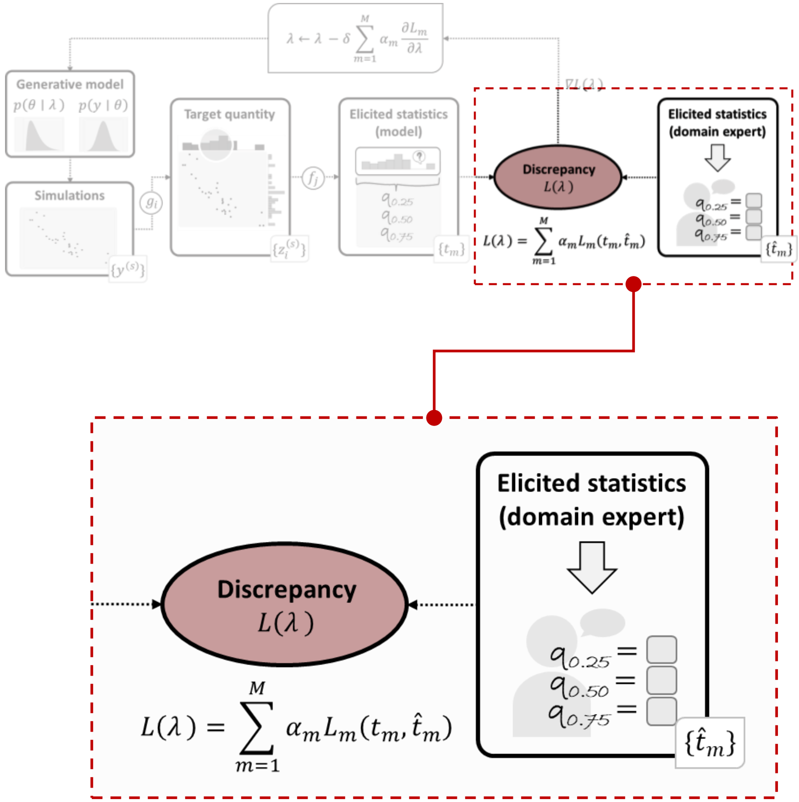
$$\lambda^* = (1.0, 0.8, 0.5, 0.5, 1.5, 0.5, 1.0)$$

Loss function:

model-implied — expert elicited

$$L(\lambda) = \alpha_1 L_1\left({R^2}^{(s)}, {R^2}^{(s)}\right) + \alpha_2 L_2\left(q^A_{0.1,0.2,\ldots,0.9}, q^A_{0.1,0.2,\ldots,0.9}\right) + \alpha_3 L_3\left(q^B_{0.1,0.2,\ldots,0.9}, q^B_{0.1,0.2,\ldots,0.9}\right)$$

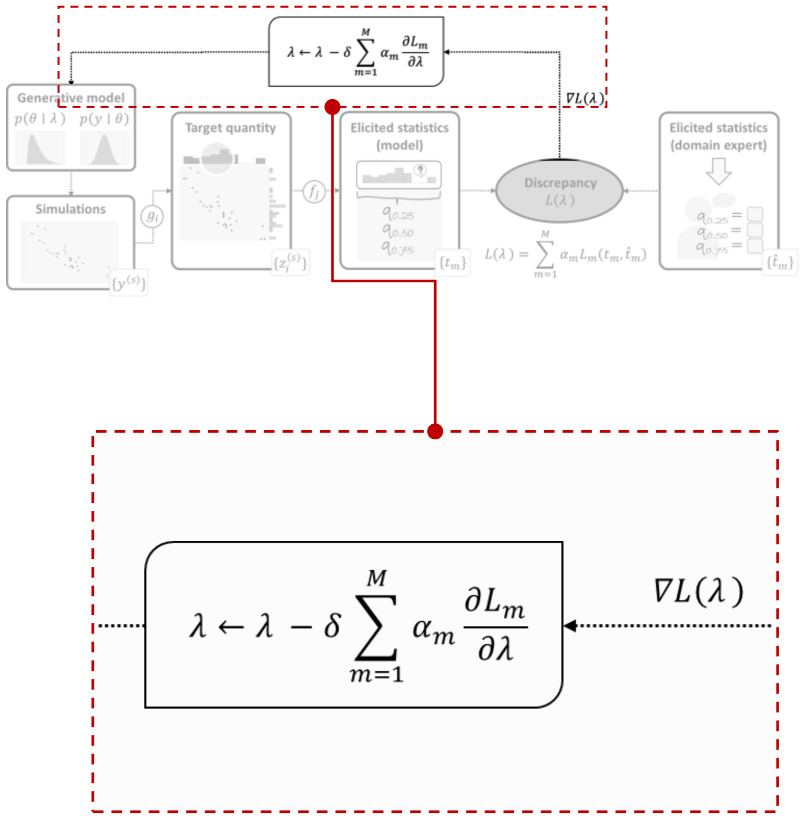




florence.bockting@tu-dortmund.de

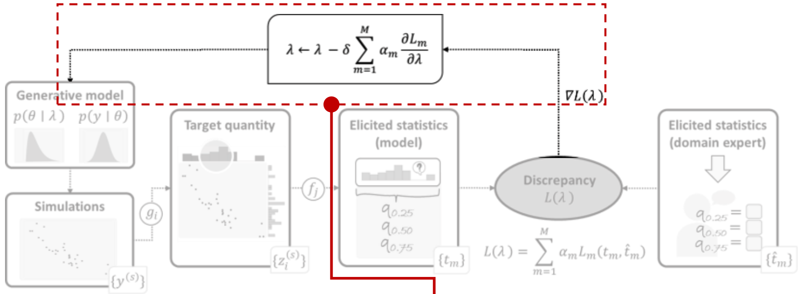
Prior Elicitation Method with SGD

Learn hyperparameter values of prior distributions



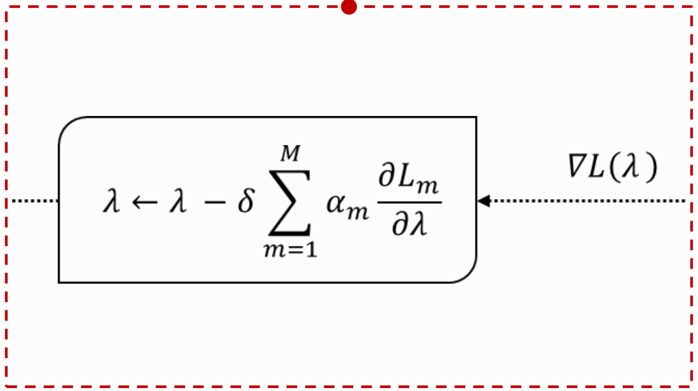
Prior Elicitation Method with SGD

Learn hyperparameter values of prior distributions

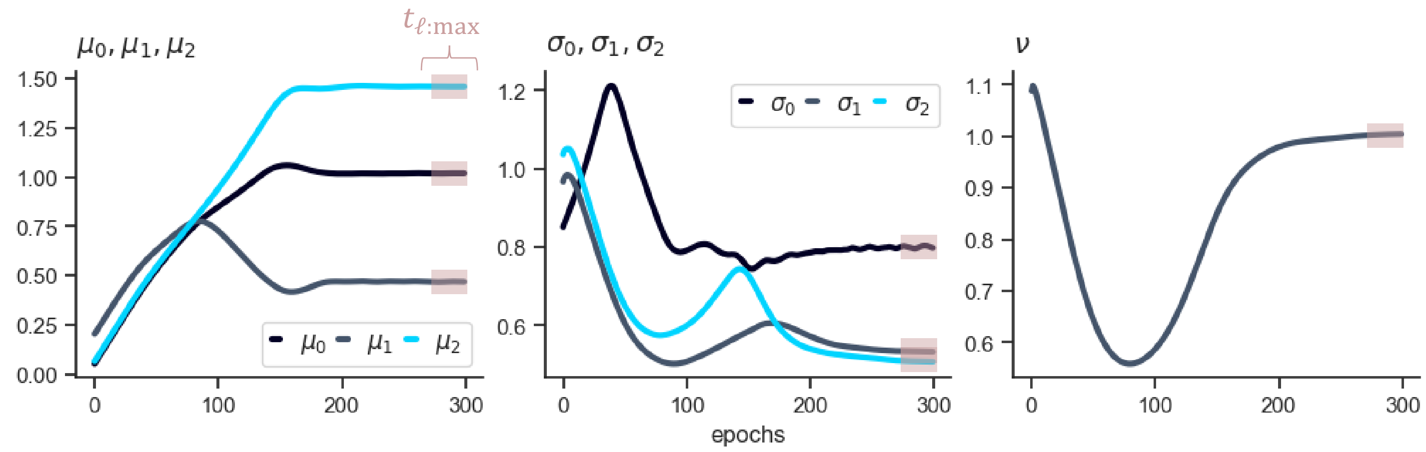


Learning hyperparameter values:

$$\text{update}(\lambda^{t_0}) \mapsto \lambda^{t_1}$$
$$\text{update}(\lambda^{t_1}) \mapsto \lambda^{t_2}$$
$$\dots$$
$$\text{update}(\lambda^{t_{\max}}) \mapsto \lambda^{t_{\max}}$$



Final learned hyperparameters: $\tilde{\lambda} = \frac{1}{t_{\max} - t_\ell} \sum_{i=t_\ell}^{t_{\max}} (\lambda^{t_i})$



florence.bockting@tu-dortmund.de

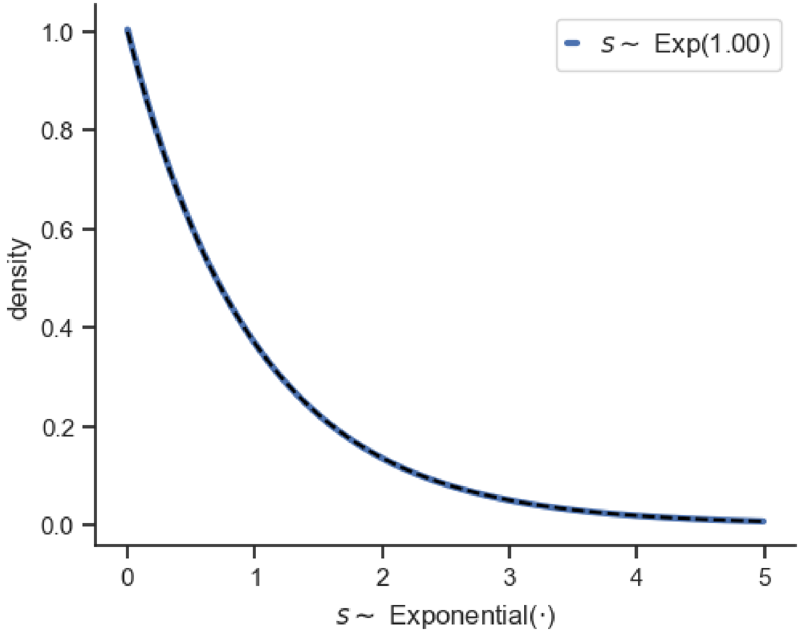
Prior Elicitation Method with SGD

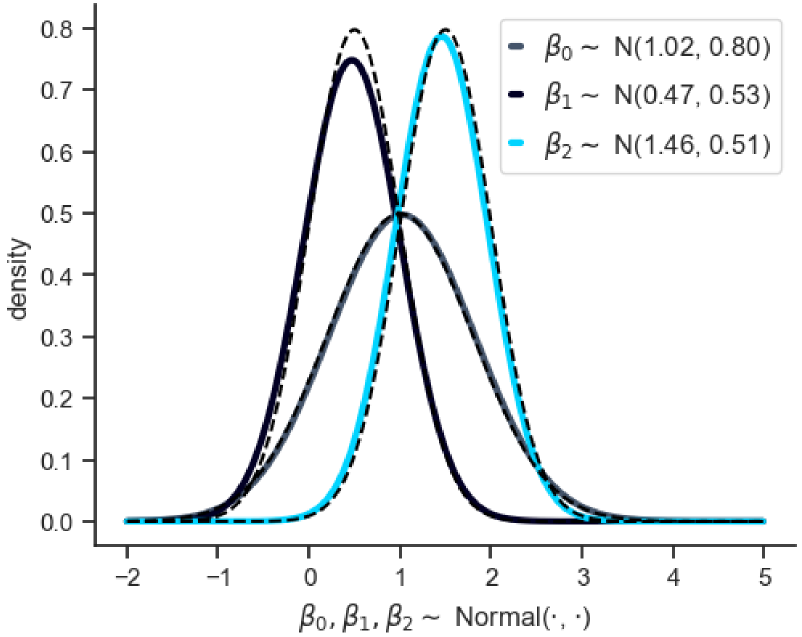
Learn hyperparameter values of prior distributions

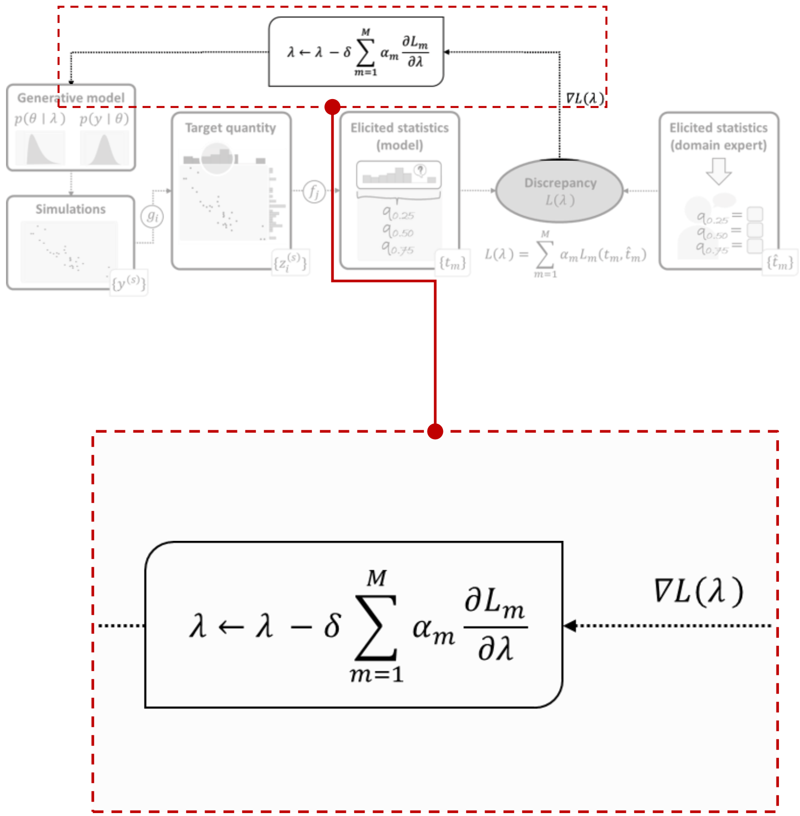
True hyperparameters vs. learned prior distributions:

$\lambda^* = (1.00, 0.80, 0.50, 0.50, 1.50, 0.50, 1.00)$

$\tilde{\lambda} = (1.02, 0.80, 0.47, 0.53, 1.46, 0.51, 1.00)$







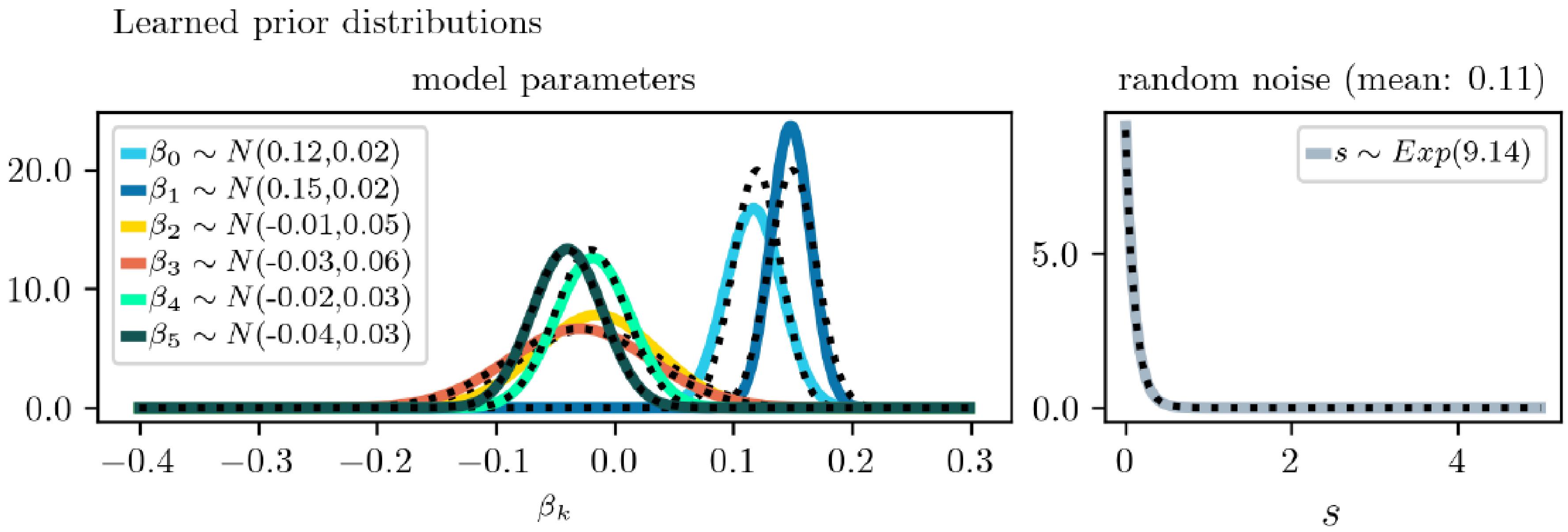
florence.bockting@tu-dortmund.de

tu dortmund university
computational statistics

Results

Learn hyperparameter values of prior distributions

$y_i \sim \text{Normal}(\theta_i, s)$
$\theta_i = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_4 + \beta_5 x_5$
$\beta_k \sim \text{Normal}(\mu_k, \sigma_k)$ for $k = 0, \dots, 5$
$s \sim \text{Exponential}(\nu)$

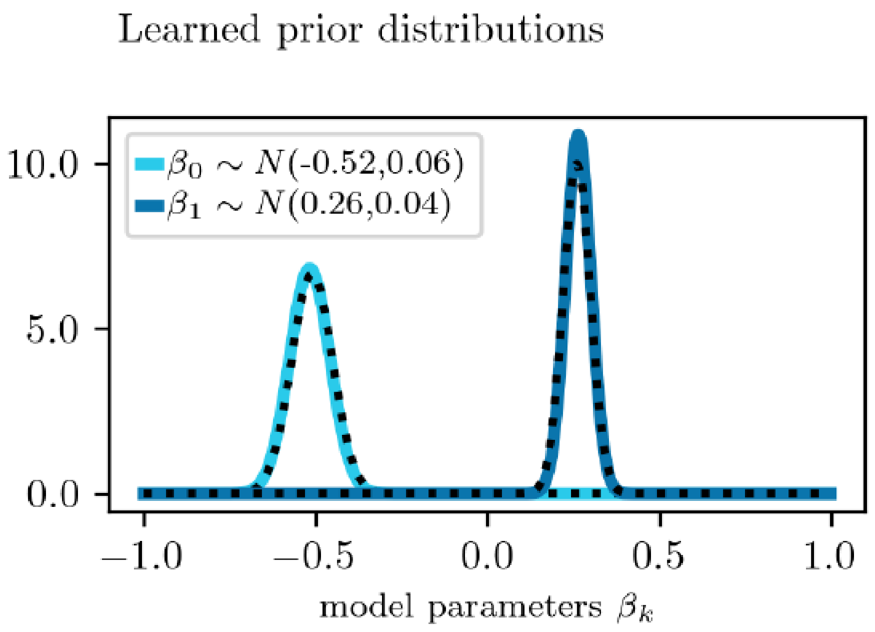


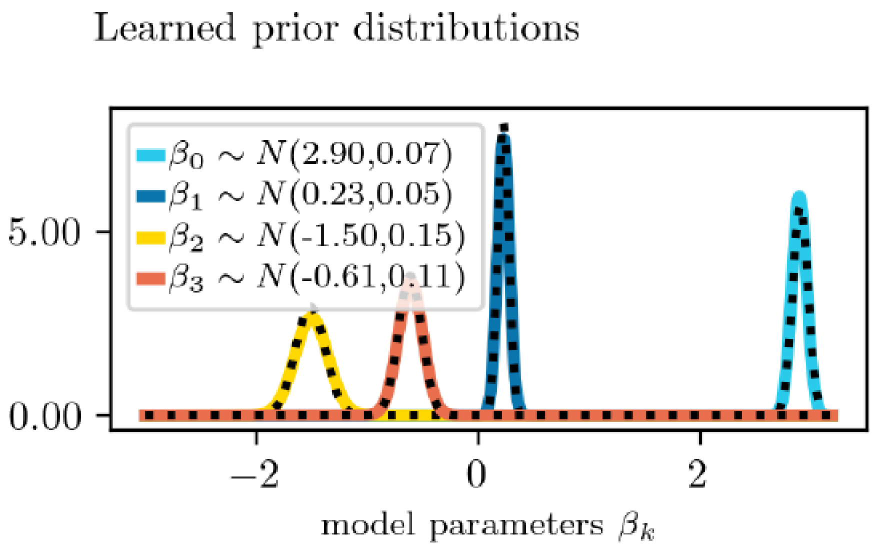
Results

Learn hyperparameter values of prior distributions

$y_i \sim \text{Binomial}(T, \theta_i)$
$\text{logit}(\theta_i) = \beta_0 + \beta_1 x_i$
$\beta_k \sim \text{Normal}(\mu_k, \sigma_k)$ for $k = 0, 1$

$y_i \sim \text{Poisson}(\theta_i)$
$\log(\theta_i) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3$
$\beta_k \sim \text{Normal}(\mu_k, \sigma_k)$ for $k = 0, \ldots, 3$





florence.bockting@tu-dortmund.de

Results

Learn hyperparameter values of prior distributions

$$y_{ij} \sim \text{Normal}(\theta_{ij}, s)$$

$$\theta_{ij} = \beta_0 + u_{0,j} + (\beta_1 + u_{1,j})x_{ij}$$

$$(u_{0,j}, u_{1,j}) \sim \text{MvNormal}(\mathbf{0}, \Sigma_u)$$

$$\Sigma_u = \begin{pmatrix} \tau_0^2 & \rho_{01}\tau_0\tau_1 \\ \rho_{01}\tau_0\tau_1 & \tau_1^2 \end{pmatrix}$$

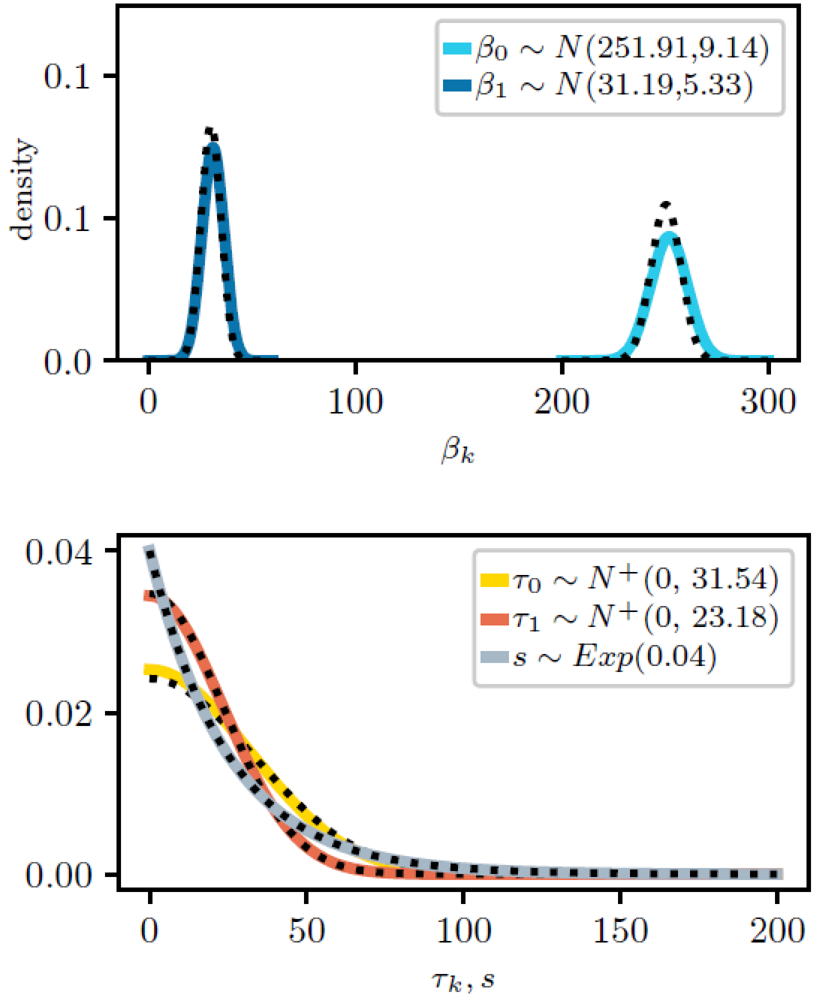
$$\beta_k \sim \text{Normal}(\mu_k, \sigma_k) \text{ for } k = 0, 1$$

$$\tau_k \sim \text{TruncatedNormal}(0, \omega_k) \text{ for } k = 0,1$$

$$\rho_{01} \sim \text{LKJ}(\alpha_{\text{LKJ}})$$

$$s \sim \text{Exponential}(\nu)$$

Learned prior distributions



Outlook

Where to go from here?

- **Conceptual level:**
  - Omit the necessity to pre-specify prior distribution families for model parameters
  - Learn joint prior distribution for all model parameters
  - Include multiple experts in analysis
  - Provide informative diagnostics for users (e.g., wrt model identification)

florence.bockting@tu-dortmund.de

Outlook

Where to go from here?

- **Conceptual level:**
  - Omit the necessity to pre-specify prior distribution families for model parameters
  - Learn joint prior distribution for all model parameters
  - Include multiple experts in analysis
  - Provide informative diagnostics for users (e.g., wrt model identification)

- **Implementational level**
  - User-friendly Python package
  - Interface to R and Stan
  - Provide useful default settings to minimize the requirement for extensive tuning

florence.bockting@tu-dortmund.de

tu dortmund university

computational statistics

Thank you for your attention.

Contact:

Florence Bockting
TU Dortmund University, GER

florence.bockting@tu-dortmund.de

Stefan T. Radev
Rensselaer Polytechnic Institute, NY, USA

radevs@rpi.edu

Paul-Christian Bürkner
TU Dortmund University, GER

https://paul-buerkner.github.io/

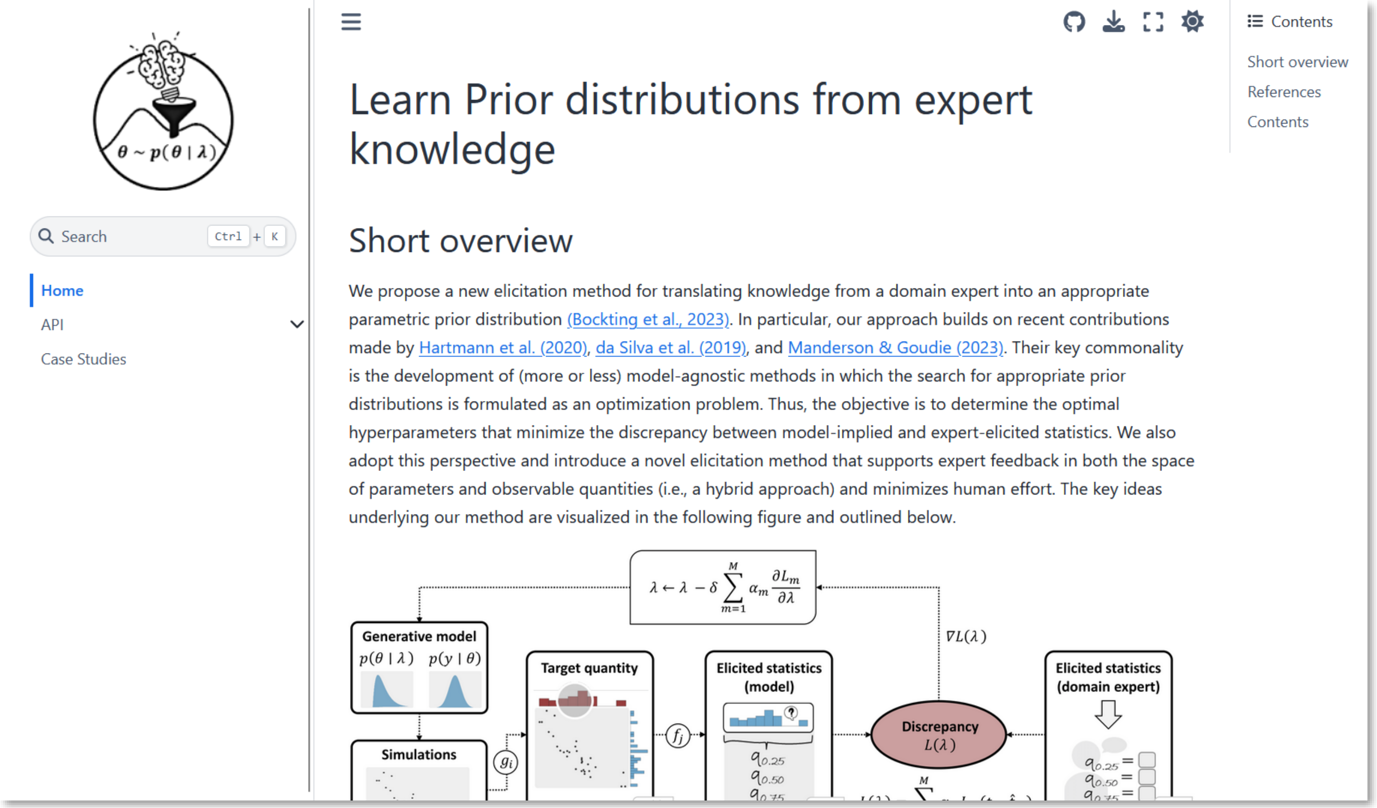
dortmund university

computational statistics

Thank you for your attention.

Project website: (under construction)

https://florence-bockting.github.io/PriorLearning/index.html



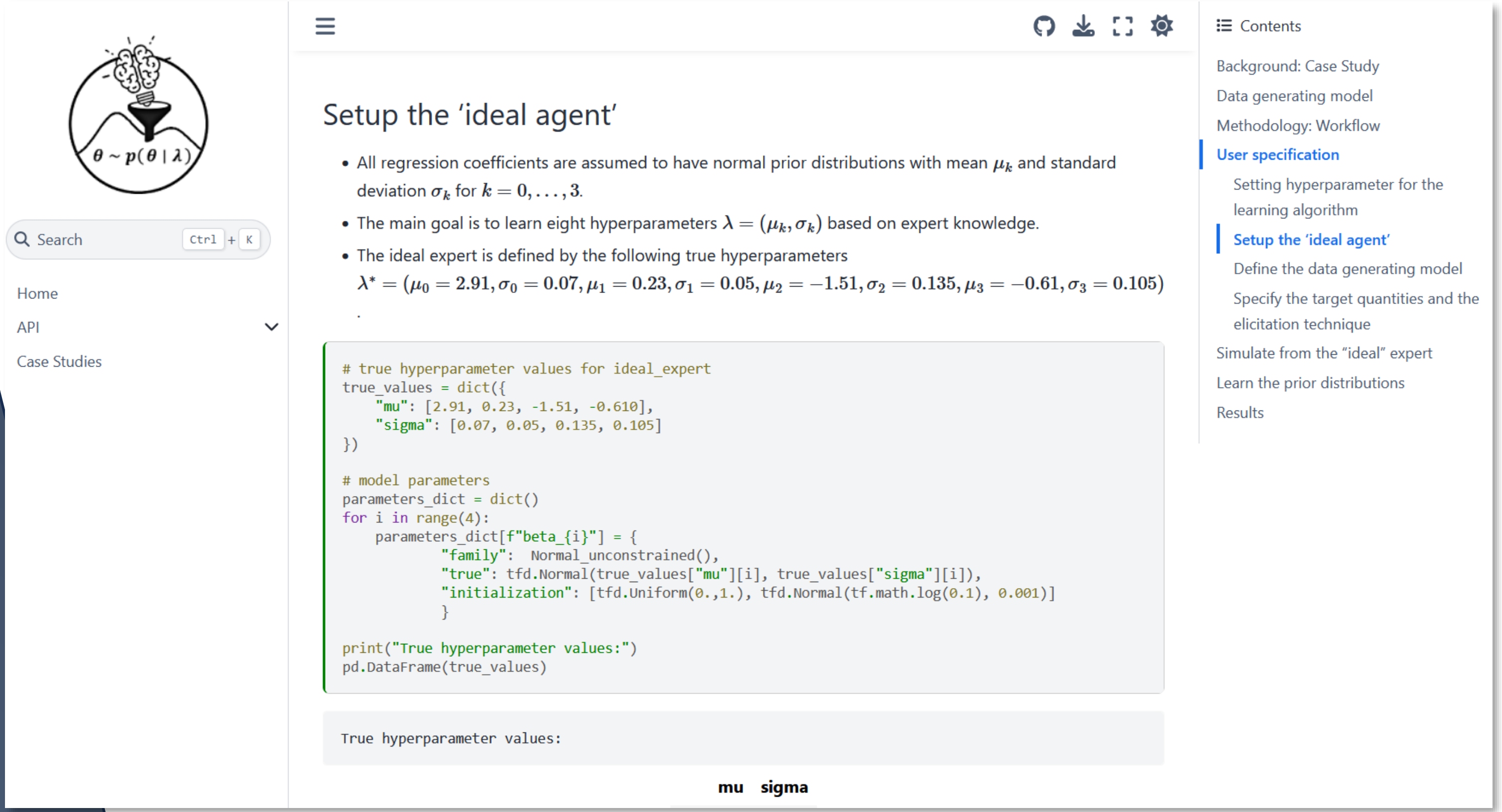
tu dortmund university

computational statistics

Thank you for your attention.

Project website: (under construction)

https://florence-bockting.github.io/PriorLearning/index.html



$\theta \sim p(\theta \mid \lambda)$

Search Ctrl + K

Home

API

Case Studies

Contents

Setup the 'ideal agent'

- All regression coefficients are assumed to have normal prior distributions with mean μ_k and standard deviation σ_k for $k = 0, \ldots, 3$.
- The main goal is to learn eight hyperparameters $\lambda = (\mu_k, \sigma_k)$ based on expert knowledge.
- The ideal expert is defined by the following true hyperparameters $\lambda^* = (\mu_0 = 2.91, \sigma_0 = 0.07, \mu_1 = 0.23, \sigma_1 = 0.05, \mu_2 = -1.51, \sigma_2 = 0.135, \mu_3 = -0.61, \sigma_3 = 0.105)$.

```
# true hyperparameter values for ideal_expert
true_values = dict({
    "mu": [2.91, 0.23, -1.51, -0.610],
    "sigma": [0.07, 0.05, 0.135, 0.105]
})

# model parameters
parameters_dict = dict()
for i in range(4):
    parameters_dict[f"beta_{i}"] = {
            "family":  Normal_unconstrained(),
            "true": tfd.Normal(true_values["mu"][i], true_values["sigma"][i]),
            "initialization": [tfd.Uniform(0.,1.), tfd.Normal(tf.math.log(0.1), 0.001)]
            }

print("True hyperparameter values:")
pd.DataFrame(true_values)
```

```
True hyperparameter values:
```

mu sigma

References

- Albert, I., Donnet, S., Guihenneuc-Jouyaux, C., Low-Choy, S., Mengersen, K., & Rousseau, J. (2012). Combining Expert Opinions in Prior Elicitation. *Bayesian Analysis*, 7(3), 503-532.
- da Silva, E. D. S., Kuśmierczyk, T., Hartmann, M., & Klami, A. (2023). Prior Specification for Bayesian Matrix Factorization via Prior Predictive Matching. *Journal of Machine Learning Research*, 24(67), 1-51.
- Hartmann, M., Agiashvili, G., Bürkner, P., & Klami, A. (2020). Flexible prior elicitation via the prior predictive distribution. In Conference on Uncertainty in Artificial Intelligence (pp. 1129-1138). PMLR.
- Manderson, A. A., & Goudie, R. J. (2023). Translating predictive distributions into informative priors. ArXiv preprint.
- Mikkola, P., Martin, O. A., Chandramouli, S., Hartmann, M., Pla, O. A., Thomas, O., … & Klami, A. (2021). Prior knowledge elicitation: The past, present, and future. ArXiv preprint.

Multi-Objective Optimization Problem

Weighted sum of loss functions

- Weighted sum of loss functions
- How to choose weights α_m?
  - Custom choice according to importance of loss component
  - Apply methods dealing with task balancing problems

$$\lambda^* = \underset{\lambda}{\operatorname{argmin}}\, L(\lambda) = \underset{\lambda}{\operatorname{argmin}} \sum_{m=1}^{M} \alpha_m L_m\big(t_m(\lambda), \hat{\lambda}_m\big)$$

florence.bockting@tu-dortmund.de

- Weights are determined based on the learning speed of each component

$$\alpha_m^t = \frac{M \cdot \exp(\gamma_m^{t-1}/a)}{\sum^M \exp(\gamma_{m'}^{t-1}/a)} \text{ with } \gamma_m^{t-1} = \frac{L_m^{t-1}}{L_m^{t_0}}$$

- High learning speed: small
- No learning: unity

- M: total number of loss components
- t-1, t_0: indexes the previous/initial iteration step
- γ_m: relative rate of descent
- a: temperature controls softness (for $a \rightarrow \infty$ the weights approach unity)

Liu, S., Johns, E., & Davison, A. J. (2019). End-to-end multi-task learning with attention. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 1871-1880).
Liu, S., Liang, Y., & Gitter, A. (2019). Loss-balanced task weighting to reduce negative transfer in multi-task learning. In *Proceedings of the AAAI conference on artificial intelligence* (Vol. 33, No. 01, pp. 9977-9978).

Softmax-Gumbel Trick
(Maddison et al., 2017; Jang et al., 2017)

- Approximation of a categorical with a continuous distribution
- Sample from a categorical distribution

$$x_i = \frac{\exp((\log \pi_i + g_i)/\tau)}{\sum_j \exp((\log \pi_j + g_j)/\tau)} \text{ with } g_i \sim_{iid} \text{Gumbel}(0,1)$$

- π_i: probability of category i among n categories
- τ: temperature parameter (higher values increase smoothness)
  (for $\tau \to 0$ the Gumbel-Softmax distr. is equiv. to the categorical distr.)

- Generalized Softmax-Gumbel trick for distributions that are not double-bounded by introducing a truncation threshold (Joo et al., 2020)

Maddison, C., Mnih, A., & Teh, Y. (2017, April). The concrete distribution: A continuous relaxation of discrete random variables. In Proceedings of the international conference on learning Representations.
Jang, E., Gu, S., & Poole, B. (2016, November). Categorical Reparameterization with Gumbel-Softmax. In International Conference on Learning Representations.
Joo, W., Kim, D., Shin, S., & Moon, I. C. (2020). Generalized Gumbel-Softmax Gradient Estimator for Generic Discrete Random Variables.

florence.bockting@tu-dortmund.de

Maximum Mean Discrepancy (Gretton et al., 2008)

- Assume: $x_i \sim p$ and $y_j \sim q$ for $i = 1, \dots, m$ and $j = 1, \dots, n$
- biased empirical estimate of the squared MMD:

$$\mathrm{MMD}_b^2 = \frac{1}{m^2}\sum_{i,j=1}^{m} k(x_i, x_j) - \frac{2}{mn}\sum_{i,j=1}^{m,n} k(x_i, y_j) + \frac{1}{n^2}\sum_{i,j=1}^{n} k(y_i, y_j)$$

- $k(\cdot,\cdot)$: continuous and characteristic kernel
- MMD is small if $p \approx q$ and large if the distributions are far apart
- Energy distance kernel (Feydy et al., 2019):

$$k(x, y) = -||x - y||$$

Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., & Smola, A. (2008). A Kernel Method for the Two-Sample Problem. Journal of Machine Learning Research, 1, 1-10.
Feydy, J., Séjourné, T., Vialard, F. X., Amari, S. I., Trouvé, A., & Peyré, G. (2019). Interpolating between optimal transport and mmd using sinkhorn divergences. In *The 22nd International Conference on Artificial Intelligence and Statistics* (pp. 2681-2690). PMLR.