Longitudinal Deep Kernel Gaussian Process Regression

Paper #1894

Derivations for L-DKGPR

Model Inference. We start with the ELBO:

$$\mathcal{L} \triangleq \mathbb{E}_{q(\mathbf{f},\mathbf{u}|X,Z)}[\log p(\boldsymbol{y}|\mathbf{f})] - \mathrm{KL}[q(\mathbf{u}|X,Z)||p(\mathbf{u}|Z)] \quad (1)$$

where $q(\mathbf{f},\mathbf{u}|X,Z) = p(\mathbf{f}|\mathbf{u},X,Z)q(\mathbf{u}|X,Z)$. Following the DTC assumption (Liu et al. 2020), we substitute $p(\mathbf{f}|\mathbf{u},X,Z)$ with its deterministic form $\mathbf{f} = A\mathbf{u}$ with $A = K_{XZ}K_{ZZ}^{-1}$. Together with the reparameterization $q(\mathbf{u}|X,Z) = \boldsymbol{\mu}_q + L_q\boldsymbol{\epsilon}$ with $\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},I)$, we can rewrite the first term of (1) as:

$$\begin{aligned}
&\mathbb{E}_{q(\mathbf{f},\mathbf{u}|X,Z)}[\log p(\boldsymbol{y}|\mathbf{f})] \\
=& -N\log\sigma - \frac{1}{2\sigma^2}\mathbb{E}_{\boldsymbol{\epsilon}}[\|\boldsymbol{y} - A(\boldsymbol{\mu}_q + L_q\boldsymbol{\epsilon}))\|_2^2] \\
=& -N\log\sigma - \frac{1}{2\sigma^2}\left(\|\boldsymbol{y}\|_2^2 - 2\boldsymbol{y}^\top A\boldsymbol{\mu}_q + \|A\boldsymbol{\mu}_q\|_2^2 + \|AL_q\mathbf{1}\|_2^2\right)
\end{aligned} \quad (2)$$

Since the second term in (1) is the KL divergence between two multivariate Gaussian distributions, the analytical form can be obtained directly as

$$\begin{aligned}
2\mathrm{KL}(q(\mathbf{u}|X,Z)||p(\mathbf{u}|Z)) =& \log\frac{|K_{ZZ}|}{|L_q|^2} \\
&- M + \mathrm{tr}(K_{ZZ}^{-1}L_qL_q^\top) + \boldsymbol{\mu}_q^\top K_{ZZ}^{-1}\boldsymbol{\mu}_q
\end{aligned} \quad (3)$$

Combining (2) and (3), we therefore obtain:

$$\begin{aligned}
\mathcal{L} =& \underbrace{-N\log\sigma - \frac{1}{2\sigma^2}\left(\|\boldsymbol{y}\|_2^2 - 2\boldsymbol{y}^\top A\boldsymbol{\mu}_q + \|A\boldsymbol{\mu}_q\|_2^2 + \|AL_q\mathbf{1}\|_2^2\right)}_{\mathbb{E}_{q(\mathbf{f},\mathbf{u}|X,Z)}[\log p(\boldsymbol{y}|\mathbf{f})]} \\
&\underbrace{-\frac{1}{2}\left[\log\frac{|K_{ZZ}|}{|L_q|^2} - M + \mathrm{tr}(K_{ZZ}^{-1}L_qL_q^\top) + \boldsymbol{\mu}_q^\top K_{ZZ}^{-1}\boldsymbol{\mu}_q\right]}_{\mathrm{KL}[q(\mathbf{u}|X,Z)||p(\mathbf{u}|Z)]}
\end{aligned} \quad (4)$$

where $\mathbf{1}$ is a column vector of ones. We can then compute the partial derivatives of $\mathcal{L}$ w.r.t. the parameters of the proposal posterior $q(\mathbf{u}|X,Z)$ (*i.e.,* $\{\boldsymbol{\mu}_q, L_q\}$) and derive its optimal form, such that:

$$\frac{\partial\mathcal{L}}{\partial\boldsymbol{\mu}_q} = \frac{1}{\sigma^2}(-A^\top\boldsymbol{y} + A^\top A\boldsymbol{\mu}_q) + K_{ZZ}^{-1}\boldsymbol{\mu}_q = 0 \quad (5)$$

$$\frac{\partial\mathcal{L}}{\partial L_q} = \frac{1}{\sigma^2}A^\top AL_q\mathbf{1}\mathbf{1}^\top + (L_q^{-\top} + K_{ZZ}^{-1}L_q) = 0 \quad (6)$$

Solving the above equations gives:

$$\boldsymbol{\mu}_q = \sigma^{-2}K_{ZZ}BK_{XZ}^\top\boldsymbol{y} \quad (7)$$

$$L_q(\mathrm{I} + \mathbf{1}\mathbf{1}^\top) = K_{ZZ}BK_{ZZ} \quad (8)$$

with $B = (K_{ZZ} + \sigma^{-2}K_{XZ}^\top K_{XZ})^{-1}$. To solve the triangular matrix L_q from (8), we first compute the Cholesky decomposition of $\mathrm{I} + \mathbf{1}\mathbf{1}^\top = CC^\top$ and $K_{ZZ}BK_{ZZ} = UU^T$. We then simplify both side of (8) to $L_qC = U$. L_q can then be solved by exploiting the triangular structure on both side with

$$L_{i,i-k} = \frac{U_{i,i-k} - \sum_{j=0}^{k-1}L_{i,i-j}C_{i-j,i-k}}{C_{i-k,i-k}}, \quad k = 0,1,\cdots,i-1 \quad (9)$$

where $L_{i,j}$ is a short notation for $[L_q]_{i,j}$.

Prediction. A common approximation assumption associated with the inducing points idea is that the signals between training data and test data are conditionally independent given $\mathbf{u}$ (Quiñonero-Candela and Rasmussen 2005). This is particularly useful during the test phase. Given the covariate matrix X_* for the test data, the prediction distribution is given by:

$$\begin{aligned}
p(\mathbf{f}_*|X_*,X,y,Z) =& \int p(\mathbf{f}_*,\mathbf{f},\mathbf{u}|X_*,X,y,Z)d\mathbf{f}d\mathbf{u} \\
=& \int p(\mathbf{f}_*|\mathbf{u},X_*,Z)p(\mathbf{f},\mathbf{u}|X,y,Z)d\mathbf{f}d\mathbf{u} \\
\simeq& \mathbb{E}_{q(\mathbf{u}|X,Z)}[p(\mathbf{f}_*|\mathbf{u},X_*,Z)] \\
=& \mathcal{N}(K_{X_*Z}[K_{ZZ} + \sigma^2\mathrm{I}]^{-1}\boldsymbol{\mu}_q, \\
& K_{X_*X_*} - K_{X_*Z}[K_{ZZ} + \sigma^2\mathrm{I}]^{-1}K_{X_*Z}^\top)
\end{aligned} \quad (10)$$

We can then make prediction using the mode and evaluate the prediction uncertainty with the covariance matrix from (10).

Implementation Details and Parameter Setup

We implement L-DKGPR using PyTorch (Paszke et al. 2019). We formulate e_γ using a deep neural network (DNN) consisting of multiple fully connected layers. Specifically, the structure of e_γ is P-H-`CELU`-`D(0.2)`-H-`CELU`-`D(0.2)`-D_v, where H is the size of hidden units, `CELU` stands for Continuously Differentiable Exponential Linear Units (Barron 2017) and `D(0.2)` represents a dropout layer with 20% dropout rate. We set $H = 16$ for simulated data and $H = 32$ for real-life data. The latent dimension D_v is fixed at 10 for all experiments. Although we only use a simple fully connected structure throughout the experiment, the implementation is flexible enough to allow more advanced DNN structure such as CNN and RNN. The embedding function g_ϕ is a I-by-D_i parameter matrix. We set $D_i = D_v$. Though the full lower triangular matrix L_q can be computed using (9), we find that approximating L_q by using only its main diagonal components provides similar accuracy, but have substantially less computation and numerically stable. Therefore, in our implementation, $\tilde{L_q} = \text{diag}(U/C)$. We update $\Theta = \{\sigma^2, Z, \alpha^{(v)}, \alpha^{(i)}, \gamma, \phi\}$ using Adam optimizer. The learning rate for $\Theta - \{\phi\}$ is fixed at 0.001. To facilitate more effective learning on cluster correlation, we assign larger learning rate on $\{\phi\}$, which is fixed at 0.01. The training and testing batch sizes are set to 1024. The maximum training epoch of L-DKGPR is set to 300 for all data sets. We use early stopping if the R^2 evaluated on validation set decrease in two consecutive epochs. The number of Inducing points is fixed at 10 for all data sets. We initialize $\{\sigma^2, \alpha^{(v)}, \alpha^{(i)}\} = 1$, $Z \sim U[0,1)^{M\times(D_v+D_i)}$. γ, ϕ are initialized with the default initialization mechanism in PyTorch. To avoid numerical issue during Cholesky decomposition, we add a small factor $\Delta = \text{diag}(\mathbf{0.001})$ to the main diagonal of the correlation matrix.

As for the implementation of our baseline methods, we use the implementations of GLMM, GEE and LGPR available in the `lmer4`, `PGEE` and `lgpr` packages, respectively from CRAN.[1]. We use the LMLFM implementation from https://github.com/junjieliang672/LMLFM. Implementation of ODVGP and KISSGP can be found through Gpytorch (Gardner et al. 2018). For GLMM, we keep most hyperparameters to their default values but increase the maximum iteration to 200. In GEE, we use an first-order auto-regressive correlation structure. The maximum iteration is fixed at 200. For LGPR, results are averaged over 5 independent simulated chains. For each chain, we use 2000 iterations. The number of burn-in samples is fixed at 200. Performance of ODVGP seems to be sensitive to the the initialization of the inducing points. We find that using the cluster centers learned by a KMeans algorithm generally produce more stable results. Throughout all experiments, the number of inducing points for both mean and variance are fixed at 100. We use the same deep encoder as used in L-DKGPR for KISSGP. The number of inducing points for KISSGP is fixed at 32. Maximum iteration for both ODVGP and KISSGP is fixed at 200.

[1]https://cran.r-project.org/

All experiments are conducted on a desktop machine with Intel Core i7-7700K CPU, 32GB RAM and RTX 2070 super graphics card. Codes are available through https://anonymous.4open.science/r/cce1f2c6-29ff-4941-993d-d597a71ecc8c/.

Experimental Data Setup

Generating Simulated Data. We construct simulated longitudinal data sets that exhibit *i.e.,* longitudinal correlation (LC) and multilevel correlation (MC) as follows: The outcome is generated using $\boldsymbol{y} = f(X) + \boldsymbol{\epsilon}$ where $f(X)$ is a non-linear transformation based on the observed covariate matrix X and the residual $\boldsymbol{\epsilon} \sim N(\mathbf{0}, \Sigma)$. To simulate longitudinal correlation, we simply set Σ to a block diagonal matrix. For each individual, we use a first-order auto-regressive correlation structure (AR(1)) with decaying factor fixed at 0.9. To simulate a data set that exhibits multilevel correlation, we first split the individuals into C clusters. We then define the cluster correlation matrix by setting the correlation associated to data points in the same cluster to 1. Finally, we compute the multilevel correlation by summing up the longitudinal correlation and cluster correlation. Following (Cheng et al. 2019; Timonen et al. 2019), we simulate 40 individuals, 20 observations, and 30 covariates for each individual. To simulate correlation among the covariates, we first generate 10 base features independently from $[0, 1)$ uniform distribution, then the covariate matrix X is computed using an encoder network with architecture $10 - 100 - Tanh - Dropout(0.7) - BatchNorm - 30 - Tanh$. It therefore results in 30 covariates that are conditionally independent given encoder network and base features. We hold out both the base features and the encoder network to all comparing methods, thus leading to a covaraite matrix with non-linear correlation that is unknown to all methods. To generate $\boldsymbol{y}$, we use another nonlinear transformation $f(X)$, which is defined by a network with structure $30 - 100 - Tanh - 1$. In our experiment, We vary the number of clusters C from $[2, 5]$.

Pre-processing on SWAN data. Since CESD score is not contained from the original SWAN data, we manually compute the score based on its definition (Radloff 1977). To form the outcome label, we define an adjusted CESD score by $y = CESD - 15$, thus $y \geq 0$ indicates depression. We center $\boldsymbol{y}$ with $\boldsymbol{y} = \boldsymbol{y} - \text{mean}(\boldsymbol{y})$. After computing the label, we exclude all columns that are directly associated to computing the CESD score. We convert the categorical features using one-hot encoding and perform standard scaling on the continuous features.

Pre-processing on GSS data. Since the original data set contains repeated columns for the same survey question, we keep only one column for each survey question. We re-format all the answer codes associated to 'unknown' and 'missing' to 'unknown'. The outcome label is derived from the field 'General Happiness', we code the value 'pretty happen' and 'very happy' to 1 and the others to -1. As the other covaraites, We convert the categorical features using one-hot encoding and perform standard scaling on the continuous features.

Pre-processing on TADPOLE data. There are three data sets in the original files. We first combine the three data

sets and remove the repeated data points. Then, we convert the categorical features using one-hot encoding and perform standard scaling on the continuous features. The outcome label is defined by the value of 'ADAS13'. Similarly, we center the label with $\boldsymbol{y} = \boldsymbol{y} - \text{mean}(\boldsymbol{y})$.

Additional Experiment Results

Run time Comparison

The CPU run times and failure to complete execution on the real-world data sets are reported in 1. We see that LGPR, GLMM and GEE are exceptionally sensitive to the number of variables. Indeed, their computational complexity increases proportional to P^3 where P is the number of variables. In contrast, L-DKGPR, LMLFM and state-of-the-art GP baselines (KISSGP and ODVGP) scale gracefully with increasing number of data points and covariates.

Correction Structure in Simulated Data.

The outcome correlations estimated by all methods on the simulated data are shown in Figure 1. It is easy to see that KISSGP and ODVGP are incapable of recovering any correlation structure from the data. LGPR seems to be slightly better than KISSGP and ODVGP when MC is presented. However, we see that only one known cluster is correctly recovered when $C > 2$. The correlation estimation results also justify the inferior regression performance in terms of R^2 as they fail to learn the correlation structure. Moreover, we see that LMLFM, GLMM and GEE are only capable of recovering LC, but not MC. This fact is quite reasonable since by design LMLFM is only able to handle a special case of MC where cluster correlation exists for individuals observe at the same time. Both GLMM and GEE rely on a correct input of correlation structure which is assumed a priori unknown. We note that L-GKDPR is able to recover most of the correlation structure present in the data. We further note that L-DKGPR, despite being the best performer among the methods compared in this study, it tends to underestimate the number of clusters because the full data correlation is approximated by a low-rank matrix (see Eq. (10)) resulting in information loss.

References

Barron, J. T. 2017. Continuously differentiable exponential linear units. *arXiv preprint arXiv:1704.07483* .

Cheng, L.; Ramchandran, S.; Vatanen, T.; Lietzén, N.; Lahesmaa, R.; Vehtari, A.; and Lähdesmäki, H. 2019. An additive Gaussian process regression model for interpretable non-parametric analysis of longitudinal data. *Nature communications* 10(1): 1798.

Gardner, J.; Pleiss, G.; Weinberger, K. Q.; Bindel, D.; and Wilson, A. G. 2018. Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration. In *Advances in Neural Information Processing Systems*, 7576–7586.

Liu, H.; Ong, Y.-S.; Shen, X.; and Cai, J. 2020. When Gaussian process meets big data: A review of scalable GPs. *IEEE Transactions on Neural Networks and Learning Systems* .

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; Desmaison, A.; Kopf, A.; Yang, E.; DeVito, Z.; Raison, M.; Tejani, A.; Chilamkurthy, S.; Steiner, B.; Fang, L.; Bai, J.; and Chintala, S. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In Wallach, H.; Larochelle, H.; Beygelzimer, A.; d'Alché-Buc, F.; Fox, E.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 32*, 8024–8035. Curran Associates, Inc.

Quiñonero-Candela, J.; and Rasmussen, C. E. 2005. A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research* 6(Dec): 1939–1959.

Radloff, L. S. 1977. The CES-D scale: A self-report depression scale for research in the general population. *Applied psychological measurement* 1(3): 385–401.

Timonen, J.; Mannerström, H.; Vehtari, A.; and Lähdesmäki, H. 2019. An interpretable probabilistic machine learning method for heterogeneous longitudinal studies. *arXiv preprint arXiv:1912.03549* .

Table 1: Runtime (in second) comparison on real-world data sets. We use 'N/A' to denote execution error.

Data sets	N	I	P	L-DKGPR	KISSGP	ODVGP	LGPR	LMLFM	GLMM	GEE
TADPOLE	595	50	24	0.03	0.34	0.03	6.39	0.01	0.01	0.13
SWAN	550	50	137	0.03	0.29	0.04	26.1	0.02	0.06	0.59
GSS	1,500	50	1,553	0.12	0.09	0.11	N/A	0.30	N/A	30.1
TADPOLE	8,771	1,681	24	1.48	0.36	1.32	N/A	0.25	0.03	4.66
SWAN	28,405	3,300	137	4.48	1.21	2.81	N/A	1.74	N/A	N/A
GSS	59,599	4,510	1,553	5.31	2.01	4.65	N/A	24.35	N/A	N/A

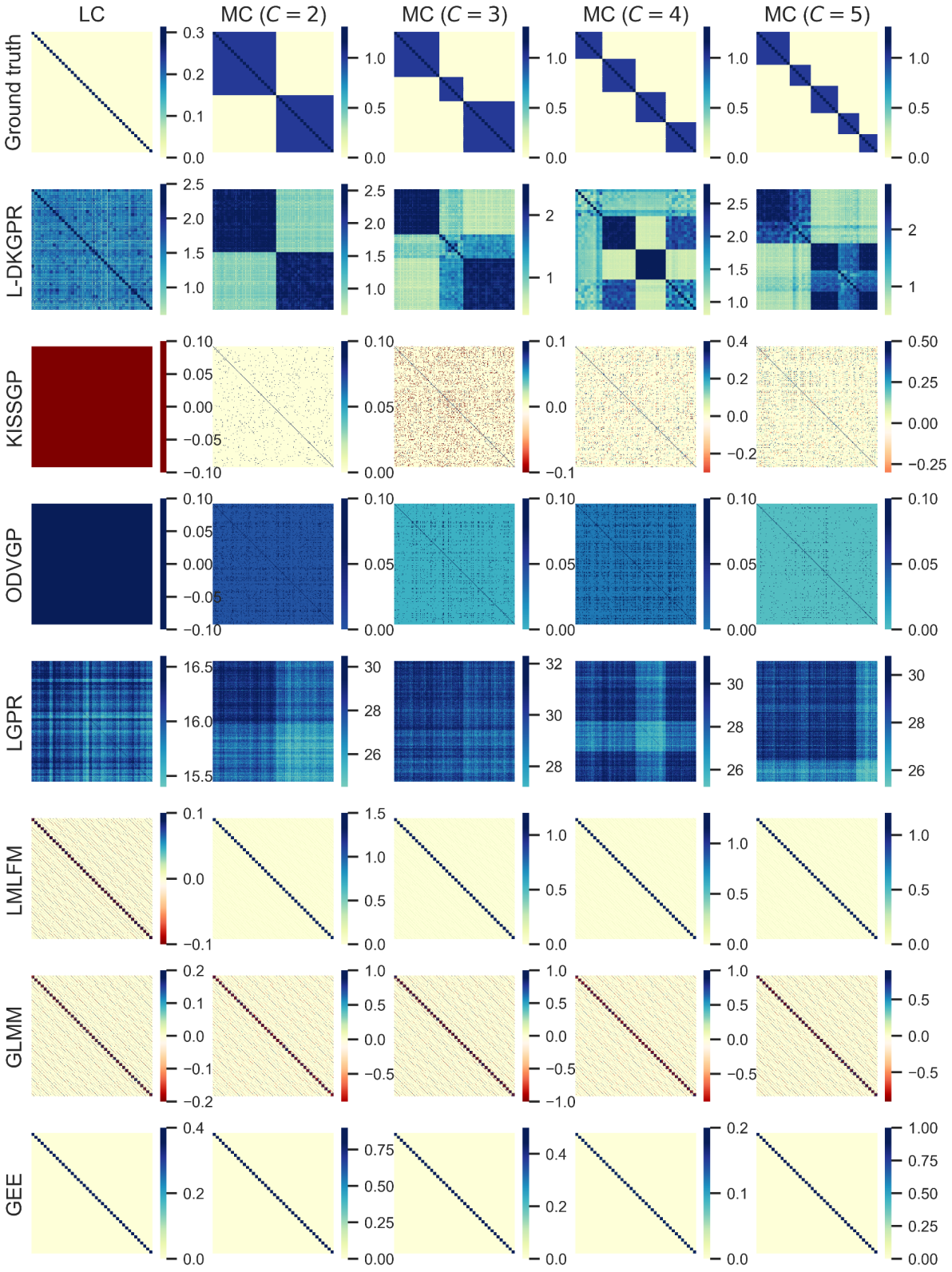

Figure 1: Outcome correlation estimated by all methods on simulated data.