1st Symposium on Advances in Approximate Bayesian Inference, 2018 1–7

Learning Sparse Representative Subsets

Si Kai Lee* A.TECHNICOLOR.SKYE@GMAIL.COM

Mohammad Emtiyaz Khan EMTIYAZ.KHAN@RIKEN.JP
RIKEN AIP

1. Introduction

Gaussian processes (GPs) are a family of statistical models for inferring distributions over latent functions fs from data. GPs are widely used and can be applied to many machine learning-related tasks such as regression, classification and optimisation. The main drawback of GPs is that inference is performed over high dimensional correlated latent variables which results in complexity of $O(N^3)$ and storage of $O(N^2)$ for a dataset of size N. Since N in modern datasets can be very large, the cost of using GPs can be prohibitive.

There exist a plethora of methods proposed to address this pitfall, most notably sparse approximations methods based on pseudo-points or inducing points (see e.g. Snelson and Ghahramani, 2006; Titsias, 2009). Sparse GPs enables us to trade off accuracy against complexity by approximating the full N-rank posterior with a low M-rank posterior where M is a set of inducing points and $M \ll N$, which reduces complexity and storage to $O(M^2N)$ and $O(M^2)$ respectively. Hensman et al. (2013)'s sparse variational Gaussian processes (SVGP) extends Titsias (2009) to very large datasets by opting to not collapse the distribution of the inducing points, thus enabling stochastic optimisation to be used. However, we know of no existing approaches for learning sparse GPs given a fixed subset of points.

We propose a framework for learning a sparse representative set of points conditioned on a subset of existing points. Next, we present two methods derived from the framework that enables sparse GPs to be learned given a fixed subset of points. The first straddles the divide between GP regression on a subset and full GP regression while the second learns a transformation from the set of sampled point to a set of representative points. We also explore the use of different priors for our methods and SVGP.

2. Learning a Representative Subset

Consider the setting where we are given N input-output tuples of $\{\boldsymbol{X_i}, y_i\}$ with $\boldsymbol{X}_i \in \mathbb{R}^D$ and $y_i \in \mathbb{R}$. Our task is to learn a latent function $\boldsymbol{f}$ with parameters $\boldsymbol{\theta}$ that best maps inputs $\boldsymbol{X_i}$ to noisy outputs y_i with added independent noise drawn from some distribution. $\boldsymbol{f}$ is a high dimensional correlated latent function so computing its gradient w.r.t $\boldsymbol{\theta}$ is expensive.

* Work done at RIKEN AIP

A natural question to ask is: could we achieve comparable performance by choosing a small subset of m points to represent the full dataset?

We can frame the above problem as finding a subset of m points that maximises the log marginal likelihood $\log p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta})$ with the following factorisation:

$$\boldsymbol{y}_m := \arg\max_{\boldsymbol{y}_m} \log p(\boldsymbol{y}|\boldsymbol{y}_m, \boldsymbol{X}, \boldsymbol{\theta}) + \log p(\boldsymbol{y}_m|\boldsymbol{X}_m, \boldsymbol{\theta}). \tag{1}$$

From this point on, we omit $\boldsymbol{X}$, $\boldsymbol{X}_m$ and $\boldsymbol{\theta}$ when sensible to reduce clutter. Campbell and Broderick (2018), Bachem et al. (2018) and Paige et al. (2016) etc. also address the problem of learning representative subsets in different contexts and settings.

A randomly chosen subset would most probably not be representative of the full dataset and would result in a latent function that does not describe the dataset well. However, given the size of modern datasets, using brute force search to find the best subset is computationally infeasible. We present two different methods in Sections 2.1 and 2.2 that maximises the above objective, which is also the GP objective, in the context of sparse GPs given a fixed subset. To be succinct, we denote the approximating Gaussian distribution as q and parameterise it with mean $\boldsymbol{\mu}$ and variance $\boldsymbol{\Sigma}$.

2.1. Stacked Subset of Data

Since any randomly chosen subset contains points that could be used in constructing a representative subset, a possible way to obtain such a subset would be to learn an additional set of M inducing points $\boldsymbol{Z}$ that compensates for the deficiencies of the existing subset $\{\boldsymbol{X}_m, \boldsymbol{y}_m\}$ by conditioning $\boldsymbol{Z}$ on the subset and using $\boldsymbol{Z}$ in conjunction with the subset to make predictions. We term this method Stacked Subset of Data (SSoD).

We start by enforcing the structure defined by the GP prior on the first term of Equation (1), $\log p(\boldsymbol{y}|\boldsymbol{y}_m)$, resulting in the factorisation $\log \int p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{y}_m)d\boldsymbol{f}$. We incorporate the inducing points $\boldsymbol{Z}$ into the model and approximate $p(\boldsymbol{y}_u|\boldsymbol{y}_m)$ with $q(\boldsymbol{y}_u)$ where $\boldsymbol{y}_u$ are the outputs of $\boldsymbol{Z}$ that have been passed through the likelihood.

$$\begin{aligned}
\log p(\boldsymbol{y}|\boldsymbol{y}_m) + \log p(\boldsymbol{y}_m) &= \log \int p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{y}_u, \boldsymbol{y}_m)p(\boldsymbol{y}_u|\boldsymbol{y}_m)d\boldsymbol{y}_u \; d\boldsymbol{f} + \log p(\boldsymbol{y}_m) \\
&\geq \mathbb{E}_{q(\boldsymbol{y}_u}\left[\log \int p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{y}_u, \boldsymbol{y}_m)\frac{p(\boldsymbol{y}_u|\boldsymbol{y}_m)}{q(\boldsymbol{y}_u)}d\boldsymbol{f}\right] + \log p(\boldsymbol{y}_m) \\
&= \mathbb{E}_{q(\boldsymbol{y}_u)}\left[\log \int p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{y}_u, \boldsymbol{y}_m)d\boldsymbol{f}\right] - KL(q||p) + \log p(\boldsymbol{y}_m) \\
&\geq \mathbb{E}_{q(\boldsymbol{y}_u)}[\mathbb{E}_{p(\boldsymbol{f}|\boldsymbol{y}_u, \boldsymbol{y}_m)}[\log p(\boldsymbol{y}|\boldsymbol{f})]] - KL(q||p) + \log p(\boldsymbol{y}_m).
\end{aligned} \tag{2}$$

When we set the number of inducing points to 0, SSoD performs GP regression on the subset. As we increase the number of inducing points, performance monotonically improves and converges to full GP regression when the number of inducing points is equal to the remaining number of points in the dataset. Hence SSoD could be seen as a bridge between GP regression on the subset and the full GP solution as it enables us to find inducing points that best represent the remaining points not in the subset. See Appendix A for sparse GP regression with SSoD.

2.2. Transformed Subset of Data

Given that in expectation, a randomly selected subset is representative of the overall dataset, we could construct a representative subset from a sampled subset given the right amount of noise. This leads us to posit that for every randomly drawn subset, there exist some function g that takes the subset $\{\boldsymbol{X}_m, \boldsymbol{y}_m\}$ as input and transforms it into a representative subset $\{\tilde{\boldsymbol{X}}_m, \tilde{\boldsymbol{y}}_m\}$. Hence, we could learn g for a random subset that yields a representative subset. We call this method Transformed Subset of Data (TSoD).

To do so, we expand $\log p(\boldsymbol{y}|\boldsymbol{y}_m)$ to incorporate g which maps $\boldsymbol{X}_m$ to $\tilde{\boldsymbol{X}}_m$

$$\begin{aligned}
\log p(\boldsymbol{y}|\boldsymbol{y}_m) + \log p(\boldsymbol{y}_m) &= \log \int p(\boldsymbol{y}|\tilde{\boldsymbol{f}}_m, \boldsymbol{y}_m) p(\tilde{\boldsymbol{f}}_m|\boldsymbol{y}_m) d\tilde{\boldsymbol{y}}_m + \log p(\boldsymbol{y}_m) \\
&\geq \mathbb{E}_{q(\tilde{\boldsymbol{f}}_m)} \left[\log p(\boldsymbol{y}|\tilde{\boldsymbol{f}}_m, \boldsymbol{y}_m) + \frac{\log p(\tilde{\boldsymbol{f}}_m|\boldsymbol{y}_m)}{\log q(\tilde{\boldsymbol{f}}_m)} \right] + \log p(\boldsymbol{y}_m) \\
&= \mathbb{E}_{q(\tilde{\boldsymbol{f}}_m)}[\log p(\boldsymbol{y}|\tilde{\boldsymbol{f}}_m, \boldsymbol{y}_m)] - KL(q(\tilde{\boldsymbol{f}}_m)||p(\tilde{\boldsymbol{f}}_m|\boldsymbol{y}_m)) + \log p(\boldsymbol{y}_m) \\
&\geq \mathbb{E}_{q(\tilde{\boldsymbol{f}}_m)}[\mathbb{E}_{p(\boldsymbol{f}|\tilde{\boldsymbol{y}}_m)}[\log p(\boldsymbol{y}|\boldsymbol{f})]] - KL(q||p) + \log p(\boldsymbol{y}_m). \qquad (3)
\end{aligned}$$

TSoD learns a specific transformation g for the given subset of points while SVGP learns a set of inducing points. TSoD generalises SVGP: in the case where the subset $\boldsymbol{Z}$ is drawn from a random distribution, we learn inducing points i.e. $p(\tilde{\boldsymbol{f}}_m|\boldsymbol{y}_m) = p(\tilde{\boldsymbol{f}}_m) = \mathcal{N}(0, K_{ZZ})$ which yields the SVGP objective. See Appendix B for sparse GP regression with TSoD.

2.3. Choice of Priors

We made two modelling choices in Equations (2) and (3) by electing to use $p(\boldsymbol{y}|\boldsymbol{y}_u)$ and $p(\boldsymbol{y}|\boldsymbol{f})$ respectively. Here, we discuss why we made such choices. Incorporating the likelihood into the conditional prior allows the model to regularise accordingly given the point-wise uncertainty of predictions made on $\boldsymbol{Z}/\tilde{\boldsymbol{X}}_m$ and increases the modelling responsibility of the likelihood parameter σ. More formally, replacing $\boldsymbol{K}_{mm}^{-1}$ with $(\boldsymbol{K}_{mm} + \sigma^2\boldsymbol{I})^{-1}$ regularises predictions by reducing the impact of $\boldsymbol{K}_{mn}$ on the mean and variance terms. We chose to use $p(\boldsymbol{y}|\boldsymbol{y}_u)$ in SSoD as we wanted to prevent the model from overfitting on the subset. On the other hand, we used $p(\boldsymbol{y}|\boldsymbol{f})$ in TSoD to ensure maximum flexibility as we are representing the full dataset with fewer points compared to SSoD. Similarly, we can use either $p(\boldsymbol{f}|\boldsymbol{y}_u)$ or $p(\boldsymbol{f}|\boldsymbol{u})$ for SVGP and see similar effects to those described above.

3. Experiments

We compare SSoD and TSoD against SVGP on various benchmark UCI GP regression datasets and show the effect of different priors for SSoD and SVGP. We run both SVGP and SSoD with two priors $p(\boldsymbol{y}|\boldsymbol{f})$ (f) and $p(\boldsymbol{y}|\boldsymbol{y}_u)$ (y), and TSoD with only $p(\boldsymbol{y}|\boldsymbol{f})$ as the empirical results obtained with this prior are consistently better than the other. We use 3 different transformations for TSOD: learning $\tilde{\boldsymbol{X}}_m$ independently of $\boldsymbol{X}$, and performing element-wise addition (A) and performing element-wise multiplication (M) on $\boldsymbol{X}$. See Appendix C for experimental details.

Tables 1 and 2 compares SSoD with subset of 500 and 250 inducing points and TSoD with 500 transformed points against SVGP with 500 inducing points on test log likelihood (TLL) and test RMSE (RMSE) respectively.

	SSoD f	**SSoD y**	**SVGP f**	**SVGP y**	**TSoD f**	**TSoD f A**	**TSoD f M**
boston	-2.96 (0.32)	-2.88 (0.31)	-3.69 (0.19)	-3.69 (0.19)	**-2.51 (0.23)**	-3.57 (0.50)	-3.66 (0.67)
concrete	-7.80 (2.94)	-2.99 (0.22)	-4.20 (0.05)	-4.20 (0.05)	-3.02 (0.08)	**-2.97 (0.28)**	-2.98 (0.28)
energy	-1.09 (0.34)	-0.51 (0.24)	-3.71 (0.04)	-0.73 (0.09)	-0.70 (0.15)	-0.46 (0.19)	**-0.45 (0.19)**
wine	-11.05 (1.40)	-0.94 (0.07)	-1.03 (0.14)	-1.19 (0.07)	-0.93 (0.07)	**-0.92 (0.08)**	**-0.92 (0.08)**
kin8nm	1.01 (0.04)	1.11 (0.01)	0.69 (0.02)	0.53 (0.01)	**1.21 (0.01)**	**1.21 (0.01)**	**1.21 (0.01)**
power	-3.14 (0.06)	-2.81 (0.05)	-2.85 (0.05)	-2.87 (0.04)	-2.75 (0.06)	**-2.69 (0.06)**	-2.73 (0.10)

Table 1: Test Log Likelihood, 500 points. Entries are denoted as mean (variance).

	SSoD f	**SSoD y**	**SVGP f**	**SVGP y**	**TSoD f**	**TSoD f A**	**TSoD f M**
boston	4.39 (0.90)	3.53 (0.44)	9.43 (1.58)	9.43 (1.58)	**3.27 (0.79)**	3.77 (0.52)	3.67 (0.56)
concrete	13.80 (4.98)	5.04 (0.63)	16.15 (0.89)	16.15 (0.89)	5.04 (0.32)	**4.74 (0.69)**	4.76 (0.67)
energy	0.49 (0.06)	0.37 (0.06)	9.87 (0.41)	0.49 (0.06)	0.48 (0.07)	**0.36 (0.05)**	**0.36 (0.05)**
wine	1.52 (0.14)	0.62 (0.05)	0.68 (0.10)	0.79 (0.05)	**0.61 (0.05)**	0.62 (0.05)	**0.61 (0.05)**
kin8nm	0.08 (0.00)	0.08 (0.00)	0.12 (0.00)	0.14 (0.00)	**0.07 (0.00)**	**0.07 (0.00)**	**0.07 (0.00)**
power	5.58 (0.28)	4.02 (0.21)	4.15 (0.19)	4.27 (0.17)	3.77 (0.23)	**3.55 (0.23)**	3.60 (0.21)

Table 2: Test RMSE, 500 points. Entries are denoted as mean (variance).

From the above tables, we can see that the choice of prior does not really affect performance of SVGP with $p(\boldsymbol{f}|\boldsymbol{u})$ generally yielding better TLL compared to $p(\boldsymbol{f}|\boldsymbol{y_u})$. For SSoD, using $p(\boldsymbol{y}|\boldsymbol{y}_u)$ instead of $p(\boldsymbol{y}|\boldsymbol{f}_m)$ almost always results in higher TLL which we attribute to its regularising effect. We obtain the best TLL with element-wise transformations for TSoD across all the datasets apart from *boston*. Generally, element-wise transformations is superior to learning $\tilde{\boldsymbol{y}}_m$ directly. We postulate that this is due the implicit regularisation by element-wise transformations which are less flexible than free transforms and might make optimisation easier. Lastly, SSoD and TSoD are competitive with SVGP and usually outperforms it. However, the performance boost comes with a corresponding computational trade-off: using TSoD and SSoD require 3 matrix inversions while SVGP only requires 1, hence the two are around 2-3x slower per iteration as compared to SVGP. A set of smaller scale experimental results can be found in Appendix D.

SSoD should be used when we would like to retain a core set of observations, for example the evaluated points in Bayesian optimisation. TSoD provides a more principled way of performing sparse GP inference initialised by randomly drawn points from the dataset without having to run a clustering algorithm to select them beforehand.

Acknowledgments

We would like to thank Thang Bui and the anonymous reviewers for providing valuable comments and suggestions.

References

Olivier Bachem, Mario Lucic, and Silvio Lattanzi. One-shot coresets: The case of k-clustering. *AISTATS*, 2018.

Trevor Campbell and Tamara Broderick. Bayesian coreset construction via greedy iterative geodesic ascent. *ICML*, 2018.

James Hensman, Nicolo Fusi, and Neil D Lawrence. Gaussian processes for big data. *UAI*, 2013.

José Miguel Hernández-Lobato and Ryan Adams. Probabilistic backpropagation for scalable learning of Bayesian neural networks. *ICML*, 2015.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *ICLR*, 2015.

De G Matthews, G Alexander, Mark Van Der Wilk, Tom Nickson, Keisuke Fujii, Alexis Boukouvalas, Pablo León-Villagrá, Zoubin Ghahramani, and James Hensman. GPflow: A Gaussian process library using TensorFlow. *JMLR*, 2017.

Brooks Paige, Dino Sejdinovic, and Frank Wood. Super-sampling with a reservoir. *UAI*, 2016.

Edward Snelson and Zoubin Ghahramani. Sparse Gaussian processes using pseudo-inputs. *NIPS*, 2006.

Michalis Titsias. Variational learning of inducing variables in sparse Gaussian processes. *AISTATS*, 2009.

Appendix A. SSoD for sparse GPs

Since $\boldsymbol{u}$ is conditioned on $\boldsymbol{y}_m$, we can obtain the conditional prior $p(\boldsymbol{u}|\boldsymbol{y}_m)$ through vanilla GP regression. Assuming additive i.i.d. Gaussian noise $\mathcal{N}(\boldsymbol{0}, \sigma^2\boldsymbol{I})$, the joint distribution of $\boldsymbol{y}_m$ and $\boldsymbol{f}_u$ has the form:

$$\begin{bmatrix}\boldsymbol{y}_m \\ \boldsymbol{u}\end{bmatrix} \sim \mathcal{N}\left(\boldsymbol{0}, \begin{bmatrix}\boldsymbol{K}_{mm} + \sigma^2\boldsymbol{I} & \boldsymbol{K}_{mZ} \\ \boldsymbol{K}_{Zm} & \boldsymbol{K}_{ZZ}\end{bmatrix}\right).$$

The conditional latent prior $p(\boldsymbol{f}_u|\boldsymbol{y}_m)$ is then a Gaussian with mean $\boldsymbol{K}_{Zm}(\boldsymbol{K}_{mm} + \sigma^2\boldsymbol{I})^{-1}\boldsymbol{y}_m$ and variance $\boldsymbol{K}_{ZZ} - \boldsymbol{K}_{Zm}(\boldsymbol{K}_{mm} + \sigma^2\boldsymbol{I})^{-1}\boldsymbol{K}_{mZ}$. We get $p(\boldsymbol{y}_u|\boldsymbol{y}_m)$ by passing $p(\boldsymbol{f}_Z|\boldsymbol{y}_m)$ through the Gaussian likelihood which results in a Gaussian with the same mean

and variance $\boldsymbol{K}_{ZZ} - \boldsymbol{K}_{Zm}(\boldsymbol{K}_{mm} + \sigma^2\boldsymbol{I})^{-1}\boldsymbol{K}_{mZ} + \sigma^2\boldsymbol{I}$. Next, we concatenate Z with the existing subset to obtain the following quantities

$$\begin{aligned}\boldsymbol{v} &= [\boldsymbol{X}_m; \boldsymbol{Z}],\\ \boldsymbol{w} &= [\boldsymbol{y}_m; \boldsymbol{\mu}],\\ \boldsymbol{S} &= \begin{bmatrix}\boldsymbol{0} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{\Sigma}\end{bmatrix}.\end{aligned}$$

The expectation term in Eq. 2 is

$$\log \mathcal{N}(\boldsymbol{y}|\boldsymbol{A}^T\boldsymbol{w}, \sigma^2) + Tr\left(\frac{\boldsymbol{K}_{nn} + \boldsymbol{A}^T(\boldsymbol{S} - \boldsymbol{K}_{SSoD})\boldsymbol{A}}{\sigma^2}\right)$$

with $\boldsymbol{K}_{SSoD} = (\boldsymbol{K}_{vv} + \sigma^2\boldsymbol{I})$ and $\boldsymbol{A}^T = \boldsymbol{K}_{nv}\boldsymbol{K}_{SSoD}^{-1}$.

Appendix B. TSoD for sparse GPs

Like in Appendix A, we perform GP regression to obtain the conditional prior. We have the joint distribution of $\boldsymbol{y}_m$ and $\tilde{\boldsymbol{f}}_m$ as

$$\begin{bmatrix}\boldsymbol{y}_m\\ \tilde{\boldsymbol{f}}_m\end{bmatrix} \sim \mathcal{N}\left(\boldsymbol{0}, \begin{bmatrix}\boldsymbol{K}_{mm} + \sigma^2\boldsymbol{I} & \boldsymbol{K}_{m\tilde{m}}\\ \boldsymbol{K}_{\tilde{m}m} & \boldsymbol{K}_{\tilde{m}\tilde{m}}\end{bmatrix}\right). \tag{4}$$

The conditional prior is parameterised by mean $\boldsymbol{K}_{\tilde{m}m}(\boldsymbol{K}_{mm} + \sigma^2\boldsymbol{I})^{-1}\boldsymbol{y}_m$ and variance $\boldsymbol{K}_{\tilde{m}\tilde{m}} - \boldsymbol{K}_{\tilde{m}m}(\boldsymbol{K}_{mm} + \sigma^2\boldsymbol{I})^{-1}\boldsymbol{K}_{m\tilde{m}}$.

The expectation in Equation (3) is

$$\log \mathcal{N}(\boldsymbol{y}|\boldsymbol{A}^T\boldsymbol{\mu}, \sigma^2) + Tr\left(\frac{\boldsymbol{K}_{nn} + \boldsymbol{A}^T(\boldsymbol{S} - \boldsymbol{K}_{TSoD})\boldsymbol{A}}{\sigma^2}\right) \tag{5}$$

with $\boldsymbol{K}_{TSoD} = (\boldsymbol{K}_{\tilde{m}\tilde{m}} + \sigma^2\boldsymbol{I})$ and $\boldsymbol{A}^T = \boldsymbol{K}_{n\tilde{m}}\boldsymbol{K}_{TSoD}^{-1}$.

In our experiments, we have tried transformations of increasing complexity. The simplest option is to directly learn $\tilde{\boldsymbol{y}}_m$, $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ which has been shown to do well in SVGP. Next, we perform either element-wise addition or multiplication i.e. $\tilde{\boldsymbol{X}}_m = \boldsymbol{X}_m + t_X$ or $\tilde{\boldsymbol{X}}_m = \boldsymbol{X}_m \odot t_X$ respectively. $\boldsymbol{S}$ would then $\boldsymbol{\Sigma}$ in the former and $\tilde{\boldsymbol{y}}_m\tilde{\boldsymbol{y}}_m^T \odot \boldsymbol{\Sigma}$ in the latter.

Appendix C. Experimental Setup

All models were implemented in GPflow (Matthews et al., 2017) and we use the provided Adam optimiser (Kingma and Ba, 2015) throughout. For the optimiser, we have step size set to 0.01 and the remaining values set to their default values. We use the RBF-ARD kernel and Gaussian likelihood with their default GPflow initialisations, and initialise $\boldsymbol{\mu}$ from $\mathcal{N}(0, 0.05^2)$ and $\boldsymbol{\Sigma} = \boldsymbol{I}$

We base our implementation of SVGP on GPflow's to account for the two different priors $p(\boldsymbol{y}|\boldsymbol{y}_Z)$ and $p(\boldsymbol{y}|\boldsymbol{f})$, and follow GPflow's choice of initialising $\boldsymbol{\mu}$ from 0 and $\boldsymbol{\Sigma} = \boldsymbol{I}$.

Preprocessing is done by scaling the input and outputs to zero mean and unit standard deviation within the training set, and scaling the test set with the previously obtained

means and standard deviations. We restore output scalings during evaluation. Minibatch size is set to 10000 or the size of the remaining points outside the subset for datasets of size smaller than 10000.

Following (Hernández-Lobato and Adams, 2015), we used 10-fold (instead of the usual 20-fold due to time constraints) cross validation with a 10% randomly selected held out test set. All methods are run for 20000 iterations to ensure convergence.

Appendix D. Smaller Scale Results

For the smaller experiment, we ran SSoD with subset of 100 and 50 inducing points and TSoD with 100 transformed points and SVGP with 100 inducing points. On the smaller datasets, using 100 instead 500 inducing/transformed points yields better results which is not surprising as that constrains model capacity and thus prevents overfitting. TLL and RMSE for this set of experiments are shown in the following tables.

	SSoD f	**SSoD y**	**SVGP f**	**SVGP y**	**TSoD f**	**TSoD f A**	**TSoD f M**
boston	-5.46 (1.54)	-2.54 (0.14)	-2.88 (0.30)	-3.05 (0.39)	-2.49 (0.15)	**-2.45 (0.19)**	-2.49 (0.18)
concrete	-3.27 (0.29)	-3.19 (0.07)	-3.32 (0.06)	-3.36 (0.05)	-3.13 (0.07)	**-3.12 (0.07)**	**-3.12 (0.07)**
energy	-0.85 (0.17)	-0.71 (0.11)	**-0.70 (0.11)**	-0.75 (0.07)	-0.71 (0.12)	-0.71 (0.15)	-0.71 (0.15)
wine	-1.56 (0.27)	-0.94 (0.06)	-0.96 (0.06)	-0.97 (0.06)	**-0.93 (0.06)**	-0.94 (0.06)	**-0.93 (0.06)**
kin8nm	**1.07 (0.03)**	0.87 (0.01)	0.55 (0.02)	0.47 (0.01)	1.05 (0.01)	0.98 (0.08)	1.05 (0.01)
power	-3.05 (0.08)	-2.87 (0.04)	-2.85 (0.05)	-2.88 (0.04)	-2.84 (0.05)	**-2.80 (0.05)**	**-2.80 (0.05)**

Table 3: Test Log Likelihood, 100 points. Entries are denoted as mean (variance).

	SSoD f	**SSoD y**	**SVGP f**	**SVGP y**	**TSoD f**	**TSoD f A**	**TSoD f M**
boston	5.55 (1.97)	3.04 (0.37)	4.40 (1.64)	5.41 (2.43)	2.89 (0.37)	**2.81 (0.40)**	2.95 (0.45)
concrete	5.81 (1.03)	5.88 (0.47)	6.67 (0.39)	6.97 (0.37)	5.54 (0.37)	**5.51 (0.37)**	5.52 (0.36)
energy	0.57 (0.10)	**0.49 (0.06)**	**0.49 (0.06)**	**0.49 (0.06)**	**0.49 (0.06)**	**0.49 (0.06)**	**0.49 (0.06)**
wine	0.80 (0.08)	0.62 (0.04)	0.63 (0.04)	0.63 (0.05)	**0.61 (0.04)**	0.62 (0.04)	0.62 (0.04)
kin8nm	**0.08 (0.00)**	0.10 (0.00)	0.14 (0.00)	0.15 (0.00)	**0.08 (0.00)**	0.09 (0.00)	**0.08 (0.00)**
power	5.15 (0.44)	4.26 (0.17)	4.17 (0.18)	4.33 (0.17)	4.12 (0.18)	**3.99 (0.21)**	**3.99 (0.21)**

Table 4: Test RMSE, 100 points. Entries are denoted as mean (variance).