# Adaptive Inducing Points Selection for Gaussian Processes

**Théo Galy-Fajou** [1] **Manfred Opper** [1]

[1] Technical University of Berlin

## Abstract

Gaussian Processes (**GPs**) are flexible non-parametric models with strong probabilistic interpretation. While being a standard choice for performing inference on time series, GPs have little techniques to work in a streaming setting. (Bui et al., 2017) developed an efficient variational approach to train online GPs by using sparsity techniques: The whole set of observations is approximated by a smaller set of inducing points (**IPs**) and moved around with new data. Both the number and the locations of the IPs will affect greatly the performance of the algorithm. In addition to optimizing their locations we propose to adaptively add new points, based on the properties of the GP and the structure of the data.

## 1. Introduction

Gaussian Processes (**GPs**) are flexible non-parametric models with strong probabilistic interpretation. They are particularly fitted for time-series (Roberts et al., 2013) but one of their biggest limitations is that they scale cubically with the number of points (Williams & Rasmussen, 2006). Quinonero-Candela & Rasmussen (2005) introduced the notion of sparse GPs, models approximating the posterior by a smaller number $M$ of inducing points (**IPs**) and reducing the inference complexity from $\mathcal{O}(N^3)$ to $\mathcal{O}(M^3)$ where $M$ is the number of IPs. Titsias (2009) introduced them later in a variational setting, allowing to optimize their locations. Based on this idea, (Bui et al., 2017) introduced a variational streaming model relying on inducing points. One of their algorithm's features is that hyper-parameters can be optimized and more specifically the number of inducing can vary between batches of data. However in their work, the number of IPs is fixed and their locations are simply optimized against the variational bound of the marginal likelihood. Having a fixed number of IPs limits the model's scope if the total data size is unknown. A gradient based approach leads to two problems:
- IP's locations need to be optimized until convergence for every batch. Therefore batches need to be sufficiently large to get a meaningful improvement. If the new data comes in very far from the original positions of the IPs, the optimization will be extremely slow.
- The number of IPs being fixed, there is no way to know how many will be required to have a desired accuracy. Finding the optimal number of IPs is also not an option as it is an ill-posed problem: the objective will only decrease with more IPs, i.e. the optimum is obtained when every data point is an IP.

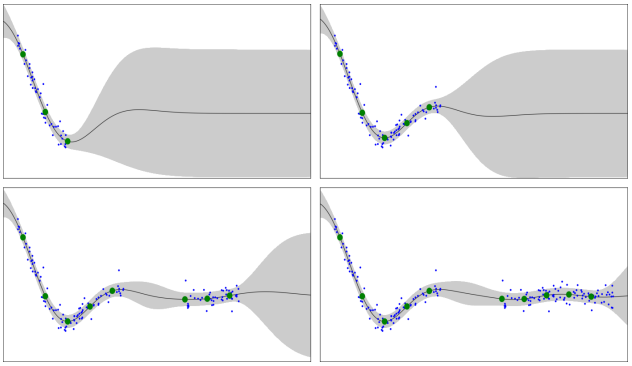

Figure 1: Illustration of the inducing point selection process. Blue points represent inducing points, green points data and the orange line represent the mean of the prediction from the GP model surrounded by one standard error. The dashed represent the space covered by the existing IPs, only points seen outside those areas are selected as new IPs.

We propose a different approach to this problem with a simple algorithm, **Online Inducing Points Selection** (**OIPS**), requiring only one parameter to select automatically both the number of inducing points and their location. OIPS naturally takes into account the structure of the data while the performance trade-off and the expected number of IPs can be inferred.

Our main contributions are as follow :
- We develop an efficient online algorithm to automatically select the number and location of inducing points for a streaming GP.
- We give theoretical guarantees on the expected number of inducing points and the performance of the GP.

In section 2 we present existing methods to select inducing

points, as well as an online inference for GPs. We present our algorithm and its theoretical guarantees in section 3. We show our experiments in comparison with popular inducing points selection methods in section 4. Finally we summarize our findings and explore outlooks in section 5.

## 2. Background

### 2.1. Sparse Variational Gaussian Processes

**Gaussian Processes:** Given some training data $\mathcal{D} = \{X, \boldsymbol{y}\}$ where $X = \{x_i\}_{i=1}^N$ are the inputs $x_i \in \mathbb{R}^D$ and $\boldsymbol{y} = \{y_i\}_{i=1}^N$ are the labels, we want to compute the predictive distribution $p(y^*|D, x^*)$ for new inputs $x^*$. In order to do this we try to find an optimal distribution over a latent function $f$. We set the latent vector $\boldsymbol{f}$ as the realization of $f(X)$, where $f_i = f(x_i)$, and put a GP prior $\mathcal{GP}(\mu_0, k)$ on $\boldsymbol{f}$, with $\mu_0$ the prior mean (set to 0 without loss of generality) and $k$ a kernel function. In this work we are going to use an isotropic squared exponential kernel (**SE kernel**) : $k(x, x') = \exp(-||x - x'||^2/l^2)$, but it is generally applicable to all translation-invariant kernels. We then compute the posterior:

$$p(\boldsymbol{f}|\mathcal{D}) = \frac{\prod_{i=1}^N p(y_i|f_i)p(\boldsymbol{f})}{p(\mathcal{D})} \tag{1}$$

Where $p(\boldsymbol{f}) \sim \mathcal{N}(0, K_{XX})$ and $K_{XX}$ is the kernel matrix evaluated on $X$ (in later notation we use $K_X$ instead of $K_{XX}$). For a Gaussian likelihood the posterior $p(\boldsymbol{f}|\mathcal{D})$ is known analytically in closed-form. Prediction and inference have nonetheless a complexity of $\mathcal{O}(N^3)$

**Sparse Variational Gaussian Processes:** When the likelihood is not Gaussian, there is no tractable solution for the posterior. One possible approximation is to use variational inference : a family of distributions over $\boldsymbol{f}$ is selected, e.g. the multivariate Gaussian $q(\boldsymbol{f}) = \mathcal{N}(\boldsymbol{m}, S)$, and one optimizes the variational parameters $\boldsymbol{m}$ and $S$ by minimizing the negative ELBO, a proxy for the KL divergence $\mathrm{KL}(q(\boldsymbol{f})||p(\boldsymbol{f}|\mathcal{D}))$. However the computational complexity still grows cubically with the number of samples, and is therefore inadequate to large datasets.

Quinonero-Candela & Rasmussen (2005) and Titsias (2009) introduced the notion of sparse variational GPs (**SVGP**). One adds inducing variables $\boldsymbol{u}$ and their inducing locations $Z = \{Z_i\}_{i=1}^M$ to the model. In this work we restrict $Z_i$ to be in the same domain as $X_i$ but inter-domain approaches also exist (Hensman et al., 2017). The relation between $\boldsymbol{u}$ and $\boldsymbol{f}$ is given by the distribution $p(\boldsymbol{f}, \boldsymbol{u}) = p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u})$ where

$$p(\boldsymbol{f}|\boldsymbol{u}) = \mathcal{N}(\boldsymbol{f}|K_{XZ}K_Z^{-1}\boldsymbol{u}, \widetilde{K}),\ p(\boldsymbol{u}) = \mathcal{N}(0, K_Z) \tag{2}$$

where $\widetilde{K} = K_X - K_{XZ}K_Z^{-1}K_{ZX}$

Then we approximate $p(\boldsymbol{f}, \boldsymbol{u})$ with the variational distribution $q(\boldsymbol{f}, \boldsymbol{u}) = p(\boldsymbol{f}|\boldsymbol{u})q(\boldsymbol{u})$ where $q(\boldsymbol{u}) = \mathcal{N}(\boldsymbol{\mu}, \Sigma)$ by optimizing $\mathrm{KL}(q(\boldsymbol{f}, \boldsymbol{u})||p(\boldsymbol{f}, \boldsymbol{u}|\mathcal{D}))$.

Note that if the likelihood is Gaussian, the optimal variational parameters $\boldsymbol{\mu}^*$ and $\Sigma^*$ are known in closed-form. The only parameters left to optimize are the kernel parameters as well as selecting the number and the location of the inducing variables.

### 2.2. Inducing points selection methods

Titsias (2009) initially proposed to select the points location via a greedy selection : A small batch of data is randomly sampled, each sample is successively tested by adding it to the set of inducing points and evaluating the improvement on the ELBO. The sample bringing the best performance is added to the set of inducing points and the operation is repeated until the desired number of inducing points is reached. This greedy approach has the advantage of selecting a set which is already close to the optimum set but is extremely expensive and is not applicable to non-conjugate likelihoods as it relies on estimating the optimal bound.

The most popular approach currently is to use the $k$-means++ algorithm (Arthur & Vassilvitskii, 2007) and take the optimized clusters centers as inducing points locations. The clustering nature of the algorithm allows to have good coverage of the whole dataset. However the $k$-means algorithm have a complexity of $\mathcal{O}(NMDT)$ on the whole dataset where $T$ is the number of $k$-means iterations. Another issue is that it might allocate multiple centers in a region of high density leading to very close inducing points and no significant performance improvement. It is also not applicable online and does not solve the problem of choosing the number of inducing points.

Another classical approach is to simply take a grid. For example Moreno-Muñoz et al. (2019) use a grid in an online setting by updating the bounds of a uniform grid. Using a grid is unfortunately limited a small number of dimensions and does not take into account the structure of the data.

### 2.3. Online Variational Gaussian Process Learning

(Bui et al., 2017) developed a streaming algorithm for GPs (**SSVGP**) based the inducing points approach of (Titsias, 2009). The method consists in recursively optimizing the variational distribution $q_t(\boldsymbol{u}_t, \boldsymbol{f})$ for each new batch of data $\mathcal{D}_t$ given the previous variational distribution $q_{t-1}(\boldsymbol{u}_{t-1}, \boldsymbol{f})$. $q_t$ initially approximates the posterior :

$$p(\boldsymbol{u}_t, \boldsymbol{f}|\mathcal{D}_{1:t}) = \frac{p(\mathcal{D}_t|\boldsymbol{f})p(\mathcal{D}_{1:(t-1)}|\boldsymbol{f})p(\boldsymbol{u}_t, \boldsymbol{f}|\theta_t)}{p(\mathcal{D}_{1:t})} \tag{3}$$

where $\theta_t$ are the set of hyper-parameters. Since $D_{1:(t-1)}$ is not accessible anymore, the likelihood on previously seen

data is approximated using the previous variational approximation $q_{t-1}(\boldsymbol{u}_{t-1})$ and the previous hyper-parameters $\theta_{t-1}$:

$$p(\mathcal{D}_{1:(t-1)}|\boldsymbol{f}) \approx \frac{q_{t-1}(\boldsymbol{u}_{t-1})p(\mathcal{D}_{1:(t-1)})}{p(\boldsymbol{u}_{t-1}|\theta_{t-1})}.$$

The distribution approximated by $q_t$ is in the end:

$$q_t(\boldsymbol{u}_t, \boldsymbol{f}|\mathcal{D}_{1:t}) \approx \frac{p(\mathcal{D}_t|\boldsymbol{f})q_{t-1}(\boldsymbol{u}_{t-1})p(\boldsymbol{u}_t, \boldsymbol{f}|\theta_t)}{p(\boldsymbol{u}_{t-1}|\theta_{t-1})} \frac{p(\mathcal{D}_{1:(t-1)})}{p(\mathcal{D}_{1:t})} \quad (4)$$

The optimization of the (bound on the) KL divergence between the two distributions for each new batch will preserve the information of $\mathcal{D}_{1:(t-1)}$ via $q_{t-1}$ and ensure a smooth transition of the hyper-parameters, including the number of inducing points. We give all technical details including the hyper-parameter derivatives and the ELBO in full form in appendix A.

## 3. Algorithm

The idea of our algorithm is that to give a good approximation, a large majority of the samples should be "close" (in the reproducing kernel Hilbert space (RKHS)) to the set $Z$ of IPs locations. Additionally, $Z$ should be as diverse as possible, since IP degeneracy will not improve the approximation. This intuition is supported by previous works:
- Bauer et al. (2016) showed that the most substantial improvement obtained by adding a new inducing point was through the reduction of the uncertainty of $q(\boldsymbol{f})$, which decreases quadratically with $K_{XZ}$.
- Burt et al. (2019) showed that the quality of the approximation made with inducing points is bounded by the norm of $Q_X = K_X - K_{XZ}K_Z^{-1}K_{ZX}$.
Therefore by ensuring that $K_{XZ}$ and $|K_Z|$ are sufficiently large, we can expect an improvement on the approximation of the non-sparse problem.

### 3.1. Adding New Inducing Points

A simple yet efficient strategy is to verify that for each new data point $x$ seen during training, there exists a close inducing point. We first compute $K_{xZ} = [k(x, Z_1), \dots, k(x, Z_M)]$. If the maximum value of $K_{xZ}$ is smaller than a threshold parameter $\rho$, the sample is added to the set of IPs $Z$. If not, the algorithm passes on to the next sample. We summarize all steps in Algorithm 1.

The streaming nature of the algorithm makes it perfectly suited for an online learning setting : it needs to see samples only once, whereas other algorithms like $k$-means need to parse all the data multiple times before converging. It is fully deterministic for a given sequence of samples and therefore convergence guarantees are given under some conditions. This approach was previously explored in a different context by Csató & Opper (2002), but was limited to small datasets.

**Algorithm 1** Online Inducing Point Selection (**OIPS**)

**Input:** sample $x$, set of inducing points $Z = \{Z_j\}_{j=1}^M$, acceptance threshold $0 < \rho < 1$, kernel function $k$
$d \leftarrow \max_j(k(x, Z_j))$
**if** $d < \rho$ **then**
  $\{Z_j\} \leftarrow \{Z_j\} \bigcup x$
  $M \leftarrow M + 1$
**end if**
**return** $\{Z_j\}$

The extra cost of the algorithm is virtually free since $K_{XZ}$ needs to be computed for the variational updates of the model.

One of our claims is that our algorithm is model and data agnostic. The reason is that as kernel hyper-parameters are optimized, the acceptance condition changes as well

Note that this method can be interpreted as a half-greedy approach of a sequential sampling of a determinantal point process (Kulesza & Taskar, 2012). In appendix B, we show that for the same number of points, the probability of our selected set is higher than the one of a k-DPP.

### 3.2. Theoretical guarantees

The final size of $Z$ is depending on many factors: the selected threshold $\rho$, the chosen kernel, the structure of the data (distribution, sparsity, etc) and the number of points seen. However by having some weak assumptions on the data we can prove a bound on the expected number of inducing points as well as on the quality of the variational approximation.

**Expected number of inducing points :** Since the selection process is directly depending on the data, it is impossible to give an arbitrary bound. However by adding assumptions on the distribution of $x$ one can

**Theorem 1.** *Given a dataset i.i.d and uniformly distributed, i.e.* $x \sim \mathcal{U}(0, a)^D$*, and a SE kernel with lengthscale* $l^D \ll 1$*, the expected number of selected inducing points* $M$ *after parsing* $N$ *points is*

$$\mathbb{E}[M|N] \leq \frac{a^D - (a^D - \alpha)^{N+1}}{\alpha}, \quad (5)$$

*where* $\alpha = \left(\frac{l\sqrt{-D\log\rho}}{2}\right)^D$.

The proof is given in the appendix C. As $N \to \infty$, this bound will converge to $a^D/\alpha$ which is the estimated number of overlapping hyper-spheres of radius $l\sqrt{-D\log\rho_{in}}$ to fill a hypercube of dimension $D$ with side length $a$. This can be used as an upper bound for any data lying in a compact domain. This confirms the intuition that the number

of selected inducing points will grow faster with larger dimensions and a larger $\rho$ and with smaller lengthscales.

**Expected performance on regression :** Burt et al. (2019) derived a convergence bound for the inducing points approach of (Titsias, 2009). Even if they show this bound in an offline setting, their bound is still relevant for online problems. They show that when $Z$ is sampled via a k-DPP process (Kulesza & Taskar, 2011), i.e. a determinantal point process conditioned on a fixed set size, the difference between the ELBO and the log evidence $\log p(\mathcal{D})$ is bounded by

$$\mathbb{E}_Z \left[ \|K_X - Q_X\| \right] \leq (M+1) \sum_{i=M+1}^{N} \lambda_i(K_X) \quad (6)$$

where $\lambda_i(K_X)$ is the $i$-th largest eigenvalue of $K_X$ and $Q_X = K_{XZ}K_Z^{-1}K_{ZX}$ is the Nyström approximation of $K_X$.

We derive a similar bound when using our algorithm instead of k-DPPsampling:

**Theorem 2.** *Let $Z$ be the set of inducing points locations of size $M$ selected via Algorithm 1 on the dataset $X$ of size $N$.*

$$\|K_X - Q_X\| \leq (N-M)\left(1 - \frac{\rho^2}{1 + M(M-1)\rho}\right) \quad (7)$$

*where $K_X$ is the kernel matrix on $X$ and $Q_X$ is the Nyström approximation of $K_X$ using the subset $Z$*

The proof and an empirical comparison are given in the appendix D.

## 4. Experiments

In this section we get a quick look on how our algorithm performs in different settings compared to approaches described in section 2.2. We compare the online model **SSVGP** described in section 2 with different IP selection techniques. We select from the first batch via k-means and then optimize them (**k-means/opt**), select them via our algorithm and optimize them (**OIPS/opt**), select them via our algorithm but don't optimize them (**OIPS**) and finally create a **Grid** that we adapt according to new bounds. We consider 3 different toy datasets, from which two are displayed in figure 2. The dataset A is a uniform time series and the output function is a noisy sinus. The dataset B is an irregular time-series, with a gap in the inputs. The output function is also a noisy sinus. Dataset C inputs are random samples from an isotropic multivariate 3D Gaussian and the output function is given by $\sin(||x||)/||x||$. All datasets contain 200 training points and 200 test points. For all experiments we use an isotropic SE kernel with fixed parameters. For datasets A and B, **Grid** and $k$**-means** has 25 IPs while **OIPS** converged to around 20 IPs. For dataset C, **Grid** has $10^3$ IPs, $k$**-means** 50, and both **OIPS** converged to 10 IPs Figure 2 shows the evolution on the average negative log likelihood on test data after every batch has been seen. On a uniform time-series context all methods are pretty much equivalent. The presence of a gap, blocks the optimization of IP locations and impede inference of future points. Whereas the grid suffers from being in high-dimensions and All details on the datasets, different training methods, hyper-parameters and optimization parameters used are to be found in appendix E.

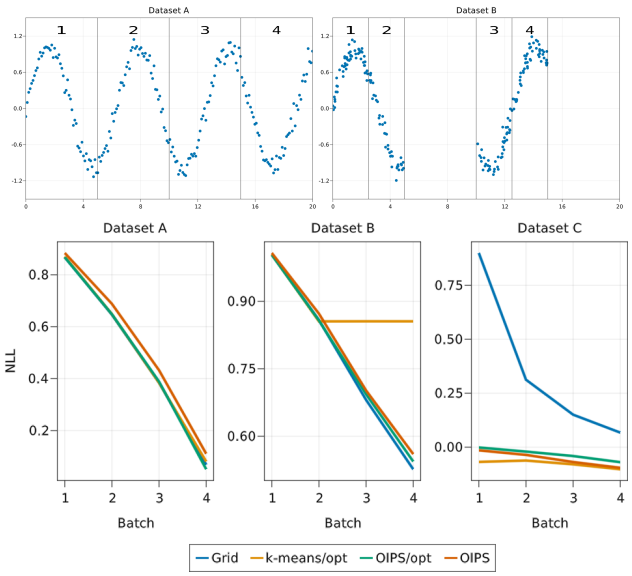


Figure 2: Toy datasets A and B, divived in 4 batches. Average Negative Test Log-Likelihood on a test set in function of number of batches seen. In a uniform streaming setting all methods perform similarly but having a gap blocks the convergence of a simple position optimization whereas in a non-compact situation the adaptive grid suffers in performance.

## 5. Conclusion

We presented a new algorithm, OIPS, able to select inducing points automatically for a GP in an online setting. The theoretical bounds derived outperforms the previous work based on DPPs. There is yet to improve the selection process to make it robust to outliers and to variations of the hyper-parameters. Using for instance a threshold on the median or a mean on the $k$-nearest IPs could help to avoid picking adversarial points such as outliers. We have only considered regression but our algorithm is also compatible with non-conjugate likelihoods. Using augmentations approaches (Wenzel et al., 2019; Galy-Fajou et al., 2019), same performance can be attained. Finally the most interesting improvement would be to use a non-stationary kernel (Remes et al., 2017) and be able to automatically adapt the number of inducing points across the dataset.

## References


Arthur, D. and Vassilvitskii, S. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pp. 1027–1035. Society for Industrial and Applied Mathematics, 2007.

Bauer, M., van der Wilk, M., and Rasmussen, C. E. Understanding probabilistic sparse gaussian process approximations. In *Advances in neural information processing systems*, pp. 1533–1541, 2016.

Belabbas, M.-A. and Wolfe, P. J. Spectral methods in machine learning and new strategies for very large datasets. *Proceedings of the National Academy of Sciences*, 106 (2):369–374, 2009.

Bui, T. D., Nguyen, C., and Turner, R. E. Streaming sparse gaussian process approximations. In *Advances in Neural Information Processing Systems*, pp. 3299–3307, 2017.

Burt, D., Rasmussen, C. E., and Van Der Wilk, M. Rates of convergence for sparse variational gaussian process regression. In *International Conference on Machine Learning*, pp. 862–871, 2019.

Csató, L. and Opper, M. Sparse on-line gaussian processes. *Neural computation*, 14(3):641–668, 2002.

Galy-Fajou, T., Wenzel, F., Donner, C., and Opper, M. Multi-class gaussian process classification made conjugate: Efficient inference via data augmentation. *arXiv preprint arXiv:1905.09670*, 2019.

Hensman, J., Durrande, N., and Solin, A. Variational fourier features for gaussian processes. *The Journal of Machine Learning Research*, 18(1):5537–5588, 2017.

Kulesza, A. and Taskar, B. k-dpps: Fixed-size determinantal point processes. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pp. 1193–1200, 2011.

Kulesza, A. and Taskar, B. Determinantal point processes for machine learning. pp. 1–120, 2012. ISSN 1935-8237. doi: 10.1561/2200000044. URL http://arxiv.org/abs/1207.6083%0Ahttp://dx.doi.org/10.1561/2200000044. ZSCC: 0000516 arXiv: 1207.6083 ISBN: 9781601986283.

Moreno-Muñoz, P., Artés-Rodríguez, A., and Álvarez, M. A. Continual multi-task gaussian processes. *arXiv preprint arXiv:1911.00002*, 2019.

Quinonero-Candela, J. and Rasmussen, C. E. A Unifying View of Sparse Approximate Gaussian Process Regression. *Journal of Machine Learning Research*, 6:1939–1959, 2005. ZSCC: NoCitationData[s0].

Remes, S., Heinonen, M., and Kaski, S. Non-stationary spectral kernels. In *Advances in Neural Information Processing Systems*, pp. 4642–4651, 2017.

Roberts, S., Osborne, M., Ebden, M., Reece, S., Gibson, N., and Aigrain, S. Gaussian processes for time-series modelling. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 371(1984):20110550, February 2013. ISSN 1364-503X, 1471-2962. doi: 10.1098/rsta.2011.0550. URL https://royalsocietypublishing.org/doi/10.1098/rsta.2011.0550.

Stewart, G. W. and guang Sun, J. *Matrix Perturbation Theory*. Academic Press, 1990.

Titsias, M. Variational learning of inducing variables in sparse gaussian processes. In *Artificial Intelligence and Statistics*, pp. 567–574, 2009.

Wenzel, F., Galy-Fajou, T., Donner, C., Kloft, M., and Opper, M. Efficient gaussian process classification using pòlya-gamma data augmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 5417–5424, 2019.

Williams, C. K. and Rasmussen, C. E. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.

# A. Derivations online GPs

## A.1. ELBO

Following Bui et al. (2017), the ELBO for variational inference is defined as :

$$\begin{aligned}\mathcal{L} = &- \mathrm{KL}\left(q_t(\boldsymbol{u}_t)||p(\boldsymbol{u}_t|\theta_t)\right) + \mathbb{E}_{q_t(\boldsymbol{u}_t,\boldsymbol{f}_t)}\left[\log p(y_t|\boldsymbol{f}_t)\right] \\ &- \mathrm{KL}(q_t(\boldsymbol{u}_{t-1})||q_{t-1}(\boldsymbol{u}_{t-1})) \\ &+ \mathrm{KL}(q_t(\boldsymbol{u}_{t-1})||p(\boldsymbol{u}_{t-1}|\theta_{t-1}))\end{aligned}$$

The terms of the first line correspond to a classical SVGP problem and the second line express the KL divergence with the previous variational posterior. The distributions are defined as :

$$\begin{aligned}q_t(\boldsymbol{u}_t) =& \mathcal{N}\left(\boldsymbol{\mu}_t, \Sigma_t\right) \\ p(\boldsymbol{u}_t|\theta_t) =& \mathcal{N}\left(0, K_{Z_t}\right) \\ q_t(\boldsymbol{u}_{t-1}) =& \int p(\boldsymbol{u}_{t-1}|\boldsymbol{u}_t) q_t(\boldsymbol{u}_t) d\boldsymbol{u}_t \\ =& \mathcal{N}\left(\kappa_{Z_{t-1}Z_t}\boldsymbol{\mu}_t, \widetilde{K}_{Z_{t-1}}\right) \\ \widetilde{K}_{Z_{t-1}} =& K_{Z_{t-1}} + \kappa_{Z_{t-1}Z_t}\Sigma_t\kappa^\top_{Z_{t-1}Z_t} \\ &- K_{Z_{t-1}Z_t}K^{-1}_{Z_t}K_{Z_tZ_{t-1}} \\ q_{t-1}(\boldsymbol{u}_{t-1}) =& \mathcal{N}\left(\boldsymbol{\mu}_{t-1}, \Sigma_{t-1}\right) \\ p(\boldsymbol{u}_{t-1}||\theta_{t-1}) =& \mathcal{N}(0, \underbrace{K'_{Z_{t-1}}}_{\text{Given } \theta_{t-1}})\end{aligned}$$

The first terms ares

$$\begin{aligned}&\mathrm{KL}(q_t(\boldsymbol{u}_t)||p(\boldsymbol{u}_t|\theta_t) = \\ &\frac{1}{2}\left(\log|K_{Z_t}| - \log|\Sigma_t| - M_t\right. \\ &\left. + \mathrm{tr}(K^{-1}_{Z_t}\Sigma_t) + \boldsymbol{\mu}_t^\top K^{-1}_{Z_t}\boldsymbol{\mu}_t\right)\end{aligned}$$

And for $p(\boldsymbol{y}_t|\boldsymbol{f}_t) = \prod_{i=1}^B \mathcal{N}(y_i|f_i, \sigma)$. The expected log-likelihood is given by L

$$\begin{aligned}\mathbb{E}_{q_t(\boldsymbol{u}_t,\boldsymbol{f}_t)}\left[\log p(\boldsymbol{y}_t|\boldsymbol{f}_t)\right] = &-\frac{B}{2}\log 2\pi\sigma^2 \\ &- \frac{1}{2\sigma^2}\sum_{i=1}^B (y_i - \kappa_{X_iZ_t}\mu_t)^2 + \widetilde{K} + \kappa_{X_iZ_t}\Sigma_t\kappa^\top_{X_iZ_t}\end{aligned}$$

Writing the second terms fully we get :

$$\begin{aligned}&\mathrm{KL}(q_t(\boldsymbol{u}_{t-1})||p(\boldsymbol{u}_{t-1}|\theta_{t-1})) = \\ &\frac{1}{2}\left(\log|K'_{Z_{t-1}}| - \log|\widetilde{K}_{t-1}| - M_{t-1}\right. \\ &+ \mathrm{tr}((K'_{Z_{t-1}})^{-1}\widetilde{K}_{Z_{t-1}}) \\ &\left.+(\kappa_{Z_{t-1}Z_t}\boldsymbol{\mu}_t)^\top(K'_{Z_{t-1}})^{-1}\kappa_{Z_tZ_{t-1}}\boldsymbol{\mu}_t\right) \\ &\mathrm{KL}(q_t(\boldsymbol{u}_{t-1})||q_{t-1}(\boldsymbol{u}_{t-1})) = \\ &\frac{1}{2}\left(\log|\Sigma_{t-1}| - \log|\widetilde{K}_{Z_{t-1}}| - M_{t-1}\right. \\ &+ \mathrm{tr}(\Sigma^{-1}_{t-1}\widetilde{K}_{Z_{t-1}}) \\ &\left.+(\boldsymbol{\mu}_{t-1} - \kappa_{Z_tZ_{t-1}}\boldsymbol{\mu}_t)^\top\Sigma^{-1}_{t-1}(\boldsymbol{\mu}_{t-1} - \kappa_{Z_tZ_{t-1}}\boldsymbol{\mu}_t)\right)\end{aligned}$$

Subtracting the second term to the first we get:

$$\begin{aligned}&\mathrm{KL}_{t:t-1} = \\ &\mathrm{KL}(q_t(\boldsymbol{u}_{t-1})||p(\boldsymbol{u}_{t-1}|\theta_{t-1})) - \mathrm{KL}(q_t(\boldsymbol{u}_t)||q_{t-1}(\boldsymbol{u}_{t-1})) \\ &= \frac{1}{2}\left(\log|K'_{Z_{t-1}}| - \log|\Sigma_{t-1}| - \mathrm{tr}((\Sigma^{-1}_{t-1} - (K'_{Z_{t-1}})^{-1})\widetilde{K}_{Z_{t-1}})\right. \\ &\quad - \boldsymbol{\mu}^\top_{t-1}\Sigma^{-1}_{t-1}\boldsymbol{\mu}_{t-1} + 2\boldsymbol{\mu}_{t-1}\Sigma^{-1}_{t-1}\kappa_{Z_{t-1}Z_t}\boldsymbol{\mu}_t \\ &\quad \left.-(\kappa_{Z_{t-1}Z_t}\boldsymbol{\mu}_t)^\top(\Sigma^{-1}_{t-1} - (K'_{Z_{t-1}})^{-1})(\kappa_{Z_{t-1}Z_t}\boldsymbol{\mu}_t)\right) \\ &= \frac{1}{2}\left(\log|K'_{Z_{t-1}}| - \log|\Sigma_{t-1}| - \mathrm{tr}(D^{-1}_{t-1}\widetilde{K}_{t-1})\right. \\ &\quad - \boldsymbol{\mu}^\top_{t-1}\Sigma^{-1}_{t-1}\boldsymbol{\mu}_{t-1} + 2\boldsymbol{\mu}_{t-1}\Sigma^{-1}_{t-1}\kappa_{Z_{t-1}Z_t}\boldsymbol{\mu}_t \\ &\quad \left.-(\kappa_{Z_{t-1}Z_t}\boldsymbol{\mu}_t)^\top D^{-1}_{t-1}(\kappa_{Z_{t-1}Z_t}\boldsymbol{\mu}_t)\right)\end{aligned}$$

Where $D_t = \left(\Sigma_t^{-1} - K_{Z_t}^{-1}\right)^{-1}$.

Taking the derivative of $\mathcal{L}$ given $\boldsymbol{\mu}_t$ and $\Sigma_t$ gives us directly the optimal solution for Gaussian regression:

$$\begin{aligned}\Sigma_t^* =& \left(\sigma^{-2}\kappa^\top_{X_tZ_t}\kappa_{X_tZ_t} + \kappa^\top_{Z_{t-1}Z_t}D^{-1}_{t-1}\kappa_{Z_{t-1}Z_t} + K^{-1}_{Z_t}\right)^{-1} \\ \boldsymbol{\mu}_t^* =& \Sigma_t\left(\kappa^\top_{X_tZ_t}\sigma^{-2}\boldsymbol{y}_t + \kappa^\top_{Z_{t-1}Z_t}\Sigma_{t-1}\boldsymbol{\mu}_{t-1}\right)\end{aligned}$$

Rewritten in natural parameters terms:

$$\begin{aligned}\eta_1^t =& \kappa^\top_{X_tZ_t}\sigma^{-2}\boldsymbol{y}_t + \kappa^\top_{Z_{t-1}Z_t}\eta_1^{t-1} \\ \eta_2^t =& -\frac{1}{2}\left(\kappa^\top_{X_tZ_t}\sigma^{-2}I\kappa_{X_tZ_t}\right. \\ &\left.+\kappa^\top_{Z_{t-1}Z_t}\left(-2\eta_2^{t-1} - K^{-1}_{Z_{t-1}}\right)\kappa_{Z_{t-1}Z_t} + K^{-1}_{Z_t}\right)\end{aligned}$$

### A.2. Hyper-parameter derivatives

Given $\theta$ a kernel hyperparameter and $J_{\square\square} = \frac{dK_{\square\square}}{d\theta}$ the derivatives are given by:

$$\begin{aligned}
\frac{d\mathrm{KL}_{t:t-1}}{d\theta_t} =& -\frac{1}{2}\mathrm{tr}\left(D_{t-1}^{-1}\frac{d\widetilde{K}_{Z_{t-1}}}{d\theta_t}\right) \\
&+ \boldsymbol{\mu}_{t-1}\Sigma_{t-1}^{-1}\frac{d\kappa_{Z_{t-1}Z_t}}{d\theta_t}\boldsymbol{\mu}_t \\
&- (\kappa_{Z_{t-1}Z_t}\boldsymbol{\mu}_t)^\top D_{t-1}^{-1}(\frac{d\kappa_{Z_{t-1}Z_t}}{d\theta_t}\boldsymbol{\mu}_t) \\
\frac{d\kappa_{Z_{t-1}Z_t}}{d\theta_t} =& \frac{dK_{Z_{t-1}Z_t}}{d\theta_t}K_{Z_t}^{-1} + K_{Z_tZ_{t-1}}\frac{dK_{Z_t}^{-1}}{d\theta_t} \\
=& (J_{Z_tZ_{t-1}} - \kappa_{Z_tZ_{t-1}}J_{Z_t})K_{Z_t}^{-1} = \iota_{Z_{t-1}Z_t} \\
\frac{d\widetilde{K}_{Z_{t-1}}}{d\theta_t} =& \frac{dK_{Z_{t-1}}}{d\theta_t} + 2\frac{d\kappa_{Z_{t-1}Z_t}}{d\theta_t}\Sigma_t\kappa_{Z_tZ_{t-1}}^\top \\
&- \frac{d\kappa_{Z_{t-1}Z_t}}{d\theta_t}K_{Z_tZ_{t-1}} - \kappa_{Z_{t-1}Z_t}\frac{dK_{Z_tZ_{t-1}}}{d\theta_t} \\
=& J_{Z_{t-1}} + 2\iota_{Z_{t-1}Z_t}\Sigma_t\kappa_{Z_{t-1}Z_t}^\top \\
&- \iota_{Z_{t-1}Z_t}K_{Z_tZ_{t-1}} - \kappa_{Z_{t-1}Z_t}J_{Z_tZ_{t-1}}
\end{aligned}$$

$$\frac{d\mathrm{KL}(q_t(\boldsymbol{u}_t)||p(\boldsymbol{u}_t|\theta_t)}{d\theta_t}$$

Special derivative given the variance :

$$\frac{dKL_a}{dv} = -\frac{1}{2}\left(\mathrm{tr}\left(D_a^{-1}\left[\frac{1}{v}(K_{aa} - K_{ab}K_{bb}^{-1}K_{ba})\right]\right)\right)$$

### A.3. Comparison with SVI

If we take the special case where inducing points do not change between iterations, then $\kappa_{Z_{t-1}Z_t} = I$ and $K_{Z_{t-1}} = K_{Z_t}$. The updates become

$$\begin{aligned}
\eta_1^t =& \kappa_{X_tZ_t}^\top\sigma^{-2}\boldsymbol{y}_t + \eta_1^{t-1} \\
\eta_2^t =& -\frac{1}{2}\left(\kappa_{X_tZ_t}^\top\sigma^{-2}\kappa_{X_tZ_t} + \left(-2\eta_2^{t-1} - K_{Z_t}^{-1}\right) + K_{Z_t}^{-1}\right) \\
=& -\frac{1}{2}\kappa_{X_tZ_t}^\top\sigma^{-2}\kappa_{X_tZ_t} + \eta_2^{t-1}
\end{aligned}$$

Compared to the SVI updates:

$$\begin{aligned}
\eta_1^t =& \eta_1^{t-1} + \rho\left(\frac{N}{|B|}\left(\kappa_{X_tZ_t}^\top\sigma^{-2}\boldsymbol{y}_t\right) - \eta_1^{t-1}\right) \\
\eta_2^t =& \eta_2^{t-1} + \rho\left(-\frac{1}{2}\left(\frac{N}{|B|}\kappa_{X_tZ_t}^\top\sigma^{-2}\kappa_{X_tZ_t} + K_{Z_t}^{-1}\right) - \eta_2^{t-1}\right)
\end{aligned}$$

If we ignore $\rho$ by setting it as 1:

$$\begin{aligned}
\eta_1^t =& \frac{N}{|B|}\left(\kappa_{X_tZ_t}^\top\sigma^{-2}\boldsymbol{y}_t\right) \\
\eta_2^t =& -\frac{1}{2}\left(\frac{N}{|B|}\kappa_{X_tZ_t}^\top\sigma^{-2}\kappa_{X_tZ_t} + K_{Z_t}^{-1}\right)
\end{aligned}$$

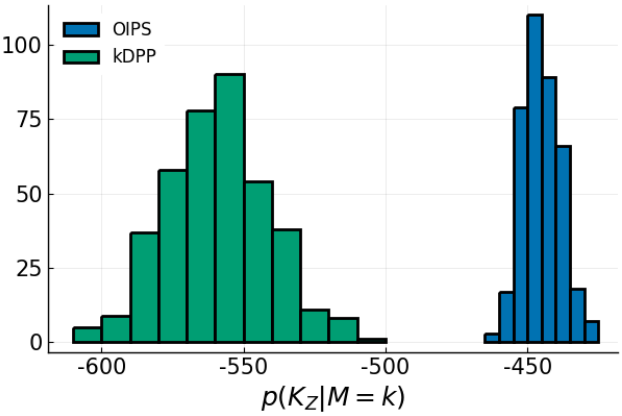


Figure 3: Histogram of $p(Z|k = M)$ for the OIPS algorithm and k-DPPsampling

We forget completely the previous $\eta_1$.
To make it directly comparable to streaming:

SVI

$$\begin{aligned}
\eta_1^{t+1} =& (1-\rho)\eta_1^t + \rho\left(\frac{N}{|B|}\left(\kappa_f^\top\sigma^{-2}y\right)\right) \\
\eta_2^{t+1} =& (1-\rho)\eta_2^t + -\frac{1}{2}\rho\left(\frac{N}{|B|}\kappa_f^\top\sigma^{-2}\kappa_f + K_{bb}^{-1}\right) \\
\eta_1^t =& (1-\rho)^t\eta_0 + \sum_{i=1}^{t}(1-\rho)^{i-1}\rho\frac{N}{|B|}\kappa_f^\top\sigma^{-2}y^i
\end{aligned}$$

Streaming

$$\begin{aligned}
\eta_1^{t+1} =& \eta_1^t + \kappa_f^\top\sigma^{-2}y \\
\eta_2^{t+1} =& \eta_2^t - \frac{1}{2}\kappa_f^\top\sigma^{-2}\kappa_f
\end{aligned}$$

## B. Deterministic algorithm as a DPP half-greedy sampling

We proceed to a simple experiment, where given a dataset, Abalone ($N = 4177, D = 7$), we repeatedly shuffle the data. We apply algorithm 1 parsing all the data to get the subset $Z_{OIPS}$. We use the resulting number of inducing points $k$ as a parameter to sample from a k-DPP and obtain $Z_{kDPP}$. We compute the probabilities of $\log p(Z_{OIPS}|M = k)$ and $\log p(Z_{kDPP}|M = k)$ and report the histogram of the probabilities on figure 3 One can observe that the probability given by the OIPS algorithm is consistently higher as well as more narrow then the sampling. This can be explained by the fact that we deterministically constrain all the points to have a certain distance from each other and therefore put a deterministic limit on the determinant of $K_Z$.

## C. Proof Theorem 1 : Bound on the number of points

Algorithm 1 can be interpreted as filling a domain with closed balls, where balls intersections are allowed but no center can be inside another ball. For a SE kernel we can compute the radius $r$ (in euclidean space) of these balls :

$$\begin{aligned} k(x,x') &= \rho_{in} \\ \exp\left(-\frac{||x-x'||^2}{h^2}\right) &= \rho_{in} \\ ||x-x'||^2 &= -h^2 \log \rho_{in} \\ r &= h\sqrt{-\log \rho_{in}} \end{aligned}$$

We can bound the volume of the union of the balls by the union of inscribed hypercubes. The length of an inscribed hypercube in an hypersphere of radius $r$ is $l = r\sqrt{D}/2$. Since the volume of the hypercube is defined to be smaller, this gives us an upper bound on the expected number of inducing points. Defining as $K_n$ the number of inducing points at time $n$, the probability of having a point outside of the union of all $k$ hypercubes is

$$\begin{aligned} p(K_{n+1} = k+1 | K_n = k) &= \max\left(a^D - \sum_{i=1}^{k} l^D\right) \\ &= \max\left(a^D - kl^D, 0\right) \\ p_k^+ &= \max\left(a^D - k\alpha, 0\right) \end{aligned}$$

Where $\alpha = \left(\frac{r\sqrt{D}}{2}\right)^D$, is the volume of one hypercube and therefore the probability of a new sample to appear in it.

The probability of keeping the same number of points is

$$\begin{aligned} p(K_{n+1} = k | K_n = k) &= \min\left(\sum_{i=1}^{k} l^D, 1\right) \\ p_k^= &= \min(k\alpha, 1) \end{aligned}$$

We now consider the problem as a Markov chain where the state $p$ is represented by a vector $\{p_i\}_{i=1}^N$ where $p_i = 1$ if there are $i$ inducing points. The transition matrix $P$ is given by :

$$P = \begin{pmatrix} p_1^= & 0 & 0 & 0 \\ p_1^+ & p_2^= & 0 & 0 \\ 0 & p_2^+ & \ddots & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & p_{N-1}^+ & p_N^= \end{pmatrix}$$

If we define that we start with inducing points the initial state is $p^1 = \{1, 0, \dots, 0\}^\top$, the probability of having $k$ balls after $n$ steps is $p(K_n = k|p^1) = \left(P^n p^1\right)_k$ while the expected number of pointsis given by $\sum_k k \cdot p(K_n = k|p^1)$.

These sequence can be complex to compute. Instead we can approximate the final expectation by recursively computing the update given the expectation at the previous step:

$$\begin{aligned} &\mathbb{E}_{p(K_{n+1}|K_n = \mathbb{E}[K_n])}\left[K_{n+1}\right] \\ &= \mathbb{E}\left[K_n\right]\mathbb{E}\left[K_n\right]\alpha + (\mathbb{E}\left[K_n\right] + 1)(a^D - \mathbb{E}\left[K_n\right]\alpha) \\ &= a^D\mathbb{E}\left[K_n\right] + a^D - \mathbb{E}\left[K_n\right]\alpha = a^D + \mathbb{E}\left[K_n\right](a^D - \alpha) \end{aligned}$$

This is an arithmetico-geometric suite and given the original condition $\mathbb{E}\left[K_0\right] = 1$ and since $\alpha < a^D$ we can get a closed form solution for $\mathbb{E}\left[K_n\right]$:

$$\begin{aligned} \mathbb{E}\left[K_n\right] =& (a^D - \alpha)^n\left(1 - \frac{a^D}{\alpha}\right) + \frac{a^D}{\alpha} \\ =& \frac{a^D - (a^D - \alpha)^{n+1}}{\alpha} \end{aligned}$$

### C.1. Empirical Comparison

We show the realization of this bound on uniform data with 3 dimensions, $\rho = 0.7$ and $l = 0.3$ on figure 4.

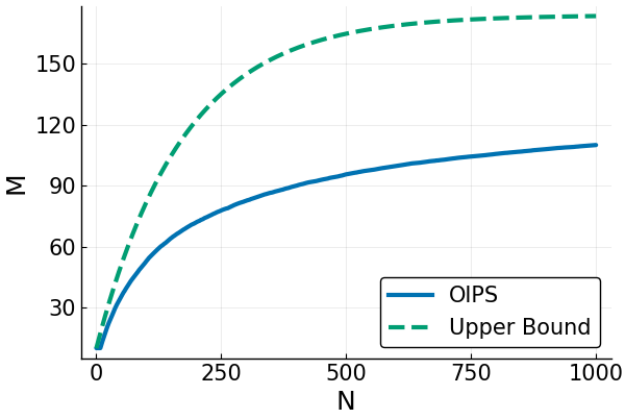


Figure 4: Bound on the number of inducing points accepted $M$ given the number of seen points $N$ vs the empirical estimation

## D. Proof theorem 2 : Bounding the ELBO

We follow the approach of Burt et al. (2019) and Belabbas & Wolfe (2009). Burt et al. (2019) showed that the error between the ELBO and the log evidence was bounded by $\|K_X - K_{XZ}K_Z^{-1}K_{ZX}\|$. Where $\|\cdot\|$ is the Froebius norm. Using a k-DPP sampling (Kulesza & Taskar, 2011), they were able to show a bound on the expectation of this norm. We follow similar calculations with our deterministic algorithm for fixed kernel parameters. Let be $K_X$ the kernel matrix of the full dataset and $K_Z$ the submatrix given

the set of points $\{Z_i\}_{i=1}^M$. The Schur complement of $K_{ZZ}$, $S_C(K_{ZZ})$ in $K_{XX}$ is given by $K_X - K_{XZ}K_Z^{-1}K_{ZX}$. Following a similar approach then Belabbas & Wolfe (2009) we bound the norm by the trace:

$$\|S_C(K_{ZZ})\| = \sqrt{\sum_{j=1}^{N-M} \overline{\lambda}_j} \leq \sum_{j=1}^{N-M} \overline{\lambda}_j = \text{tr}(S_C(K_{ZZ}))$$

Using the definiton of $S_C(K_{ZZ})$ we get :

$$\text{tr}(S_C(K_{ZZ})) = \sum_{i=1}^{N-M} K_{X_i} - K_{X_iZ}K_Z^{-1}K_{ZX_i}$$

where every element of the sum is a scalar. Taking $W^\top \overline{\Lambda} W$ the eigendecomposition of $K_Z^{-1}$, $w_i = WK_{X_iZ}$ and assuming a kernel variance $v$ of 1 (although generalizable to all variances) and a translation invariant kernel such that $k(x,x) = 1$ we get :

$$K_{X_i} - K_{X_iZ}K_Z^{-1}K_{ZX_i} = 1 - w_i^\top \Lambda w_i = 1 - \sum_{j=1}^{M} \overline{\lambda_j}(w_i)_j^2$$
$$\leq 1 - \overline{\lambda}_{\min}\|w_i\|^2 = 1 - \overline{\lambda}_{\min}\|K_{X_iZ}\|^2 \leq 1 - \overline{\lambda}_{\min}\rho^2$$

Where we used the fact that at least $X_i$ was close enough to at least one $Z_j$ such that $k(X_i, Z_j) > \rho$. For clarity we replace $\overline{\lambda}_{\min} = \lambda_{\max}^{-1}$ where $\lambda_{\max}$ is the largest eigenvalue of $K_Z$. When summing over the trace we get the final bound :

$$\|K_X - K_{XZ}K_Z^{-1}K_{ZX}\| \leq (N-M)\left(1 - \frac{\rho^2}{\lambda_{\max}}\right)$$

Now by construction all off-diagonal terms of $K_Z$ are smaller than $\rho$. Using the equality (Stewart & guang Sun, 1990)

$$|\lambda_i(A) - \lambda_i(B)| \leq \|A - B\|, \quad \forall i = 1, \ldots, N$$

We get that

$$|\lambda_{\max}(K_Z) - 1| \leq \|K_Z - I\|_2 = \sqrt{\sum_{i \neq j} (K_Z)_{ij}^2}$$
$$\leq M(M-1)\rho$$

Assuming $\lambda_{\max}(K_Z) \geq 1$, we get

$$\lambda_{\max}(K_Z) \leq 1 + M(M-1)\rho_{out}$$

Getting then the final bound :

$$\|K_X - Q_X\| \leq (N-M)\left(1 - \frac{\rho^2}{1 + M(M-1)\rho}\right)$$

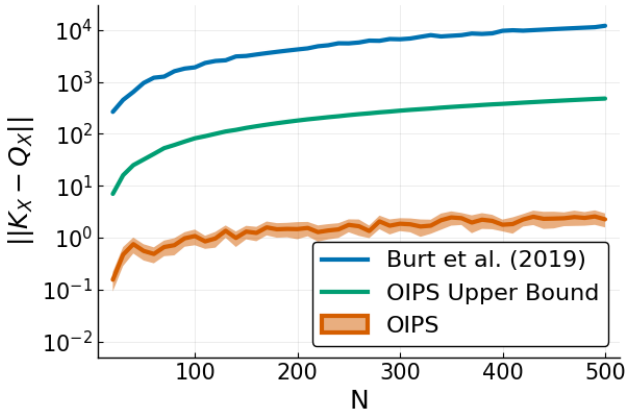


Figure 5: Evaluation of the $\|K_X - Q_X\|$ given the OIPS algorithm and computation of the bound from Burt et al. (2019) given in equation 6 and our bound given in equation 7

### D.1. Empirical Comparison

These bounds are difficult to compare due to the different parameters characterizing them. Nevertheless we give an example by comparing the bound and the empirical value on toy data drawn uniformly in 3 dimensions in figure 5. For each $N$ we ran our algorithm and input the required $M$ in the bounds as the resulting number of selected inducing points. We show in the section 4 the empirical effect on the accuracy and on the number of points given the choice of $\rho$.

## E. Experiments parameters

For every problem we use an isotropic Squared Exponential Kernel :

$$k(\mathbf{x}, \mathbf{x}') = v \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{h^2}\right)$$

Where $h$ is initialized by taking the median of the lower triangular part of the pairwise distance matrix of the first subset of points and fixed for the rest of the training. Future work will involve working with kernel parameter optimization as well. We fix the noise of the Gaussian likelihood to $\sigma^2 = 0.01$.

IPs were optimized via ADAM ($\alpha = 10^{-2}$).