Appendix
Sparse Spectrum Warped Input Measures for Nonstationary Kernel Learning

Anonymous Author(s)
Affiliation
Address
email

A Mathematical derivations

Lemma 1. *Let* $\mathbf{v} \sim \mathcal{N}(\hat{\mathbf{v}}, \mathbf{\Sigma_v})$ *denote a Gaussian random vector and* $\mathbf{x} \in \mathbb{R}^D$ *an arbitrary point. Then* $\mathbf{z} := \mathbf{v} \odot \mathbf{x}$ *is Gaussian,* $\mathbf{z} \sim \mathcal{N}(\hat{\mathbf{z}}, \mathbf{\Sigma_z})$*, with mean and covariance matrix given by:*

$$\hat{\mathbf{z}} = \hat{\mathbf{v}} \odot \mathbf{x} \tag{1}$$

$$\mathbf{\Sigma_z} = \mathbf{x}\mathbf{1}^\mathsf{T} \odot \mathbf{\Sigma_v} \odot \mathbf{1}\mathbf{x}^\mathsf{T} \tag{2}$$

Proof. The element-wise vector product, which is the Hadamard product for single-column matrices, is symmetric and linear, i.e. $\mathbf{a} \odot \mathbf{b} = \mathbf{b} \odot \mathbf{a}$ and $\mathbf{a} \odot (\mathbf{b} + \mathbf{c}) = \mathbf{a} \odot \mathbf{b} + \mathbf{a} \odot \mathbf{c}$, for any $\mathbf{a}, \mathbf{b}, \mathbf{c} \in \mathbb{R}^D$. By the linearity of the expectation, we then have $\mathbb{E}[\mathbf{z}] = \mathbb{E}[\mathbf{v} \odot \mathbf{x}] = \mathbb{E}[\mathbf{v}] \odot \mathbf{x}$. From the properties of the Hadamard product, one can also show that $\mathbf{a}(\mathbf{b} \odot \mathbf{c})^\mathsf{T} = \mathbf{a}\mathbf{b}^\mathsf{T} \odot \mathbf{1}\mathbf{c}^\mathsf{T}$, where $\mathbf{1}$ is a vector of 1's. Therefore, the covariance matrix $\mathbf{\Sigma_z} := \mathbb{V}[\mathbf{z}] = \mathbb{E}[(\mathbf{z} - \hat{\mathbf{z}})(\mathbf{z} - \hat{\mathbf{z}})^\mathsf{T}]$, is given by:

$$\begin{aligned} \mathbb{V}[\mathbf{z}] &= \mathbb{E}[((\mathbf{v} - \hat{\mathbf{v}}) \odot \mathbf{x})((\mathbf{v} - \hat{\mathbf{v}}) \odot \mathbf{x})^\mathsf{T}] \\ &= \mathbf{x}\mathbf{1}^\mathsf{T} \odot \mathbb{E}[(\mathbf{v} - \hat{\mathbf{v}})(\mathbf{v} - \hat{\mathbf{v}})^\mathsf{T}] \odot \mathbf{1}\mathbf{x}^\mathsf{T} \\ &= \mathbf{x}\mathbf{1}^\mathsf{T} \odot \mathbb{V}[\mathbf{v}] \odot \mathbf{1}\mathbf{x}^\mathsf{T}\,, \end{aligned} \tag{3}$$

which concludes the proof. □

B Relationship to output warped GPs

A body of work has previously been developed under the title of *warped* Gaussian processes [2]. As noted, this contrasts from our modelling problem because while we proceed to expand the GP's capabilities to warp the *inputs*, the WGP and extensions warps explicitly the *output* distribution of the Gaussian process. We now juxtapose the efficacy of our input warping formulation with WGP by applying SSWIM to the three challenging datasets *abalone*, *creep*, and *ailerons* experimented upon in [2, 1].

Our results in Table Table 1 suggest that with SSWIM we are able to improve upon both WGP and BWGP in MSE: marginal improvements for *abalone* and a signficant improvement for *creep* with comparable performance in *ailerons*. Contrasting this, the output warping methods outperform unanimously on the MNLP metric. This is expected because output warping may allow one to capture non-gaussian conditional distributions which an input warping formulation cannot with a standard Gaussian process. The discussion we wish to raise here is that both aspects of manipulating the inputs and outputs of a GP can result in major improvements respectively across different metrics.

Submitted to 34th Conference on Neural Information Processing Systems (NeurIPS 2020). Do not distribute.

Table 1: MSE and MNLP metrics for comparison with Warped and Bayesian Warped GPs [1]. MSE results for *ailerons* are $\times 10^{-8}$.

	Method	abalone	creep	ailerons
MSE	GP	4.55 ± 0.14	584.9 ± 71.2	2.95 ± 0.16
	BWGP	4.55 ± 0.11	491.8 ± 36.2	2.91 ± 0.14
	MLWGP3	4.54 ± 0.10	502.3 ± 43.3	$\mathbf{2.80 \pm 0.11}$
	MLWGP20	4.59 ± 0.32	506.3 ± 46.1	3.42 ± 2.87
	$SSWIM_1$	4.64 ± 0.13	483.69 ± 64.12	2.96 ± 0.08
	$SSWIM_2$	$\mathbf{4.50 \pm 0.11}$	$\mathbf{279.86 \pm 31.88}$	2.83 ± 0.06
MNLP	GP	2.17 ± 0.01	4.46 ± 0.03	-7.30 ± 0.01
	BWGP	1.99 ± 0.01	4.31 ± 0.04	-7.38 ± 0.02
	MLWGP3	$\mathbf{1.97 \pm 0.02}$	$\mathbf{4.21 \pm 0.03}$	-7.44 ± 0.01
	MLWGP20	1.99 ± 0.05	$\mathbf{4.21 \pm 0.08}$	$\mathbf{-7.45 \pm 0.08}$
	$SSWIM_1$	2.18 ± 0.01	4.45 ± 0.03	-7.24 ± 0.01
	$SSWIM_2$	2.17 ± 0.02	4.27 ± 0.03	-7.00 ± 0.02

C Additional Experiments

C.1 Increasing number of pseudo-training points

For the "increasing number of pseudo-training points" experiment we used 1 layer of warping with 256 features for both the warping and top-level predictive functions.

C.2 Increasing warping depth

We used 256 features and 1280 pseudo-training points for all of the experiments.

C.3 Complete real-dataset experiments table

Table 2 contains additional real-world experiments to extend the majore experimental results from the main paper.

Table 2: RMSE and MNLP metrics for various real world datasets.

	(D, N)	$(18, 8751)$	$(5, 1503)$	$(8, 1030)$	$(16, 5875)$	$(15, 17379)$	$(379, 53500)$	$(81, 21263)$	$(9, 45730)$	$(77, 583250)$	$(90, 515345)$
	Method	elevators	airfoil	concrete	parkinsons	bikeshare	ct slice	supercond	protein	buzz	song
RMSE ($\times 10^{-1}$)	$SSWIM_1$	2.83 ± 0.08	2.38 ± 0.26	3.05 ± 0.26	7.63 ± 0.20	0.13 ± 0.04	0.46 ± 0.02	3.44 ± 0.14	5.91 ± 0.07	2.98 ± 0.04	8.12 ± 0.05
	$SSWIM_2$	$\mathbf{2.74 \pm 0.08}$	$\mathbf{2.35 \pm 0.22}$	$\mathbf{3.01 \pm 0.31}$	$\mathbf{7.55 \pm 0.15}$	0.11 ± 0.03	$\mathbf{0.23 \pm 0.01}$	$\mathbf{3.02 \pm 0.04}$	$\mathbf{5.80 \pm 0.08}$	$\mathbf{2.40 \pm 0.01}$	$\mathbf{7.97 \pm 0.03}$
	DSDGP	$\mathbf{2.74 \pm 0.06}$	4.30 ± 0.19	5.88 ± 1.24	7.94 ± 0.20	0.33 ± 0.55	4.81 ± 1.18	5.10 ± 0.84	5.96 ± 0.06	3.65 ± 0.75	8.46 ± 0.03
	DKL	3.06 ± 0.29	3.19 ± 0.37	3.18 ± 0.38	8.84 ± 0.74	0.24 ± 0.03	0.52 ± 0.08	3.46 ± 0.18	7.15 ± 1.10	4.11 ± 3.33	16.66 ± 8.14
	RFFNS	2.83 ± 0.07	3.31 ± 0.37	3.46 ± 0.24	8.15 ± 0.15	0.05 ± 0.01	4.39 ± 0.27	3.85 ± 0.05	6.87 ± 0.06	5.70 ± 0.84	8.35 ± 0.03
	SVGP	2.88 ± 0.10	2.70 ± 0.15	3.32 ± 0.26	8.14 ± 0.12	0.06 ± 0.03	1.16 ± 0.02	4.06 ± 0.05	7.32 ± 0.08	9.98 ± 0.02	12.19 ± 0.18
	SGPR	4.96 ± 2.02	4.24 ± 0.40	5.55 ± 0.58	7.86 ± 0.22	0.67 ± 0.18	1.79 ± 0.04	4.27 ± 0.06	6.45 ± 0.07	2.89 ± 0.02	8.40 ± 0.04
	RFFS	2.87 ± 0.10	3.28 ± 0.24	3.33 ± 0.30	8.24 ± 0.17	$\mathbf{0.03 \pm 0.00}$	2.34 ± 0.05	3.89 ± 0.06	6.91 ± 0.07	3.78 ± 0.14	8.36 ± 0.04
MNLP ($\times 10^{-1}$)	$SSWIM_1$	1.81 ± 0.55	1.25 ± 2.50	10.22 ± 4.15	11.95 ± 0.47	-11.89 ± 0.15	-11.24 ± 0.05	3.55 ± 0.32	8.95 ± 0.12	2.03 ± 0.13	12.08 ± 0.05
	$SSWIM_2$	1.65 ± 0.63	$\mathbf{1.05 \pm 1.74}$	5.19 ± 2.59	12.50 ± 0.44	-11.78 ± 0.07	$\mathbf{-11.79 \pm 0.02}$	$\mathbf{2.82 \pm 0.29}$	$\mathbf{8.82 \pm 0.15}$	$\mathbf{-0.09 \pm 0.04}$	$\mathbf{11.93 \pm 0.04}$
	DSDGP	$\mathbf{1.16 \pm 0.21}$	9.19 ± 0.20	11.02 ± 1.06	$\mathbf{11.91 \pm 0.24}$	-23.28 ± 8.29	6.62 ± 2.61	7.36 ± 1.62	9.04 ± 0.10	3.80 ± 2.02	12.52 ± 0.04
	DKL	7.57 ± 0.15	7.70 ± 0.17	7.69 ± 0.20	13.17 ± 1.12	6.82 ± 0.01	6.83 ± 0.01	7.76 ± 0.10	11.02 ± 1.53	9.01 ± 4.65	42.64 ± 44.77
	RFFNS	1.44 ± 0.24	2.74 ± 1.20	3.31 ± 0.45	12.18 ± 0.18	-11.97 ± 0.00	5.95 ± 0.66	4.66 ± 0.12	10.39 ± 0.08	8.78 ± 1.87	12.39 ± 0.04
	SVGP	1.78 ± 0.30	1.19 ± 0.33	$\mathbf{2.83 \pm 0.56}$	12.21 ± 0.14	$\mathbf{-27.70 \pm 1.24}$	-5.98 ± 0.13	5.32 ± 0.12	11.10 ± 0.09	63.31 ± 3.44	18.02 ± 0.09
	SGPR	11.59 ± 22.38	6.45 ± 0.47	8.48 ± 2.10	12.39 ± 0.30	-13.67 ± 0.98	-3.14 ± 0.26	5.58 ± 0.10	10.05 ± 0.13	1.11 ± 0.12	11.97 ± 0.07
	RFFS	1.59 ± 0.26	2.72 ± 0.69	3.05 ± 0.96	12.29 ± 0.20	-11.98 ± 0.00	-0.33 ± 0.22	4.79 ± 0.13	10.45 ± 0.09	4.41 ± 0.37	12.40 ± 0.05

C.4 Extended discussion

It is imperative to note here our aim is not to demand any algorithmic dominance when comparing methods. Firstly, this is a fruitless pursuit due to the diversity and dependence of all advanced GP methods on hyperparameter optimization, and secondly it is not a constructive approach to the communal development of new methodologies to claim some benchmark task superiority. Rather, encouraged by the results of this paper, we invite the discussion to move beyond stationary kernels and inquire upon and interpret the effectiveness of new Gaussian process methodologies through the

more general perspective of input space warping for unearthing hidden nonstationary patterns within data.

C.5 Overfitting analysis

We ran an overfitting analysis of $SWIM_1$ to observe the effect of over-optimising with respect to the marginal likelihood. We ran with 256 features, 1280 pseudo-training points, for 150 steps, with 10 repeats, and evaluated the test RMSE and test MNLP on the test set for every single epoch of optimisation. This test bench is provided in the supplementary code. The results are averaged with mean and standard deviations of the training curves displayed in Figure 1. We can see that we are quite resistant to overfitting except for RMSE in the *elevators* dataset and the MNLP in the *concrete* and *parkinsons* datasets. The causes of this could be explained by the underlying flexibility of the proposed method which allows the model to become overconfident in what it has learned with respect to the data it has observed. In fact, we observed similar overfitting behaviour for similar optimisation periods with DKL and DSDGP.

This analysis leads to some interesting observations and recommendations for future algorithm development in more expressive GP methodologies: 1. the marginal likelihood is no panacea although it is easy to think it is, and 2. other loss functions and training schemes, such as leave-one-out cross validation. Actually, these results corroborate long known discussions from [3] about the risk of overfitting from trusting the marginal likelihood with standard optimisation procedures, however their importance seems to have been largely ignored in evaluation of recent methodology innovations in the GP literature. We believe that a more open discussion should be on the table for analysing the interplay between model expressiveness and the effect this has on overfitting; this is especially pertinent to the GP literature which has placed a large emphasis on the importance of the marginal likelihood has a valid hyperparameter optimisation loss.

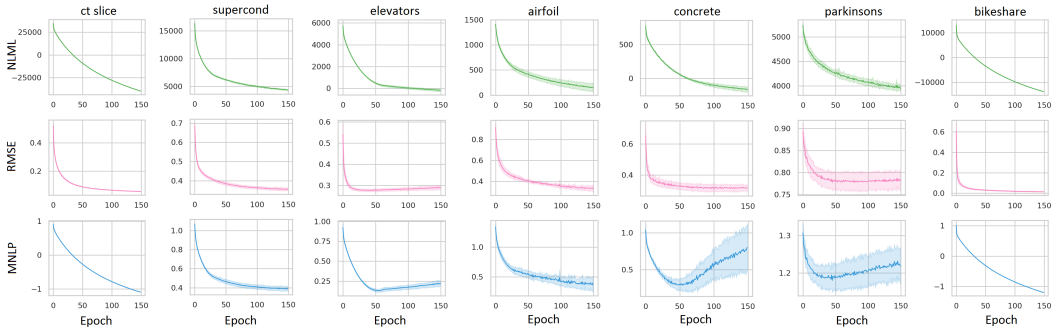

Figure 1: Empirical analysis of overfitting behaviour in SSWIM

C.6 Pseudo-training points in $2D$

Figure 2 contains an interpretation of input warping and the pseudo-training points in higher dimensions. We use the exponential 2D function from [4] as a case where it is intuitive to observe how SSWIM reacts to topological differences in the underlying function. The function consists of a mostly flat surface with abrupt spiking occurring at a corner of the domain as seen in Figure 2 (b). If we were to assume a homogeneous domain, and model our data with a standard SSGP with stationary RBF kernel, the kernel's hyperparameters would be optimised to provide a homogeneous representation resulting in conflict between the spiked area in the corner and flat areas elsewhere. By directly manipulating the input domain with our warping, we are able to transform the input domain with a continuous warping that consequently allows accurate representation as seen in Figure 2 (c).

The intuition and utility of our proposed *pseudo-training* as virtual training points is visualised from a birds-eye view in Figure 2 (d). Before optimisation, we initialise these points uniformly across the domain of the training space. It is clear that the learned positions of the pseudo-training data have transformed their spatial locations away from uniform. At the bottom left they appear to have clustered near the discontinuity of the test function while in the remaining corners of the space the

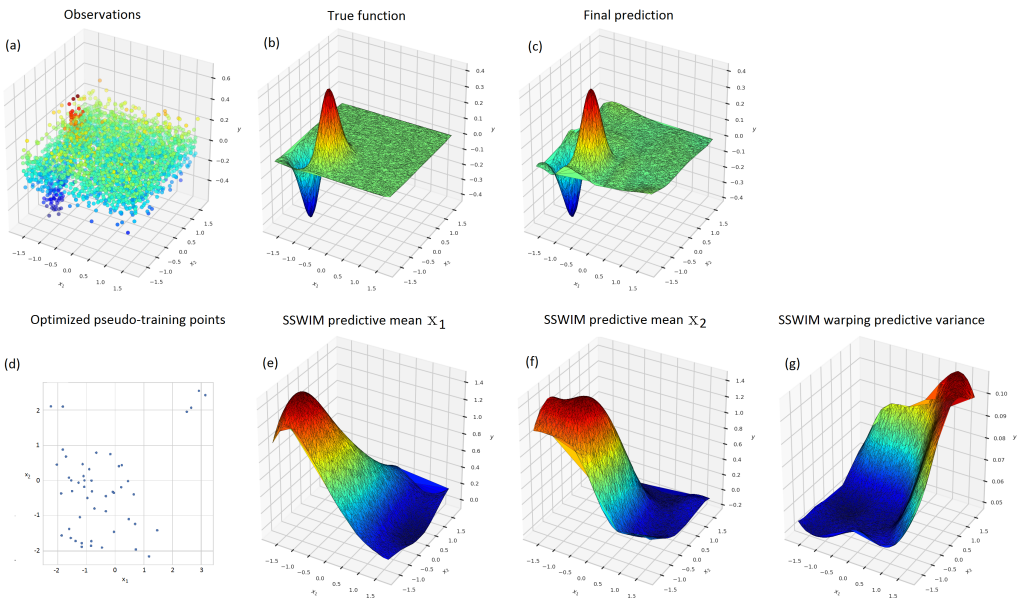

Figure 2: Visualisation of learning spatial positioning of pseudo training points in 2D. We demonstrate spatial nonstationarity with the exponential 2D function from [4]. (a) Noisy training data, (b) True function surface, (c) SSWIM prediction conditioned on training data, (d) Learned pseudo-training point positions, (e) Learned warping predictive mean for x_1, (f) Learned warping predictive mean for x_2, (g) Learned warping predictive variance.

points have spread away. The predictive mean of the learned warping function is thus visualised across the domain, for both x_1 and x_2, in Figure 2 (e) and (f). Furthermore, Figure 2 (g) shows the predictive variance of the warping function and we can see how a lower amount of spatial uncertainty arises both from the noisy pseudo-targets as well as the pseudo-inputs from (d).

D Datasets and Experimental Conditions

For completeness, we specify for each dataset used, the data dimensionality and sample size, the raw data source, the modelling objective (i.e. the target) as defined for the original problem, and any target variable pre-processing excluding standardisation. The dimensionality D is reported for the inputs X (i.e. excluding the target variable y). All problems are single output regression tasks. Number of samples N is reported for the entire dataset before train/test splitting is applied. Note that we do not alter the raw data files and pre-processing is applied through code exactly as in the provided supplementary code. *dataloader.py*.

elevators

$D : 18$
$N : 8751$
Source: `https://web.archive.org/web/*/http://www.liacc.up.pt/~ltorgo/Regression/*`
Preprocessing: None
Target: goal

airfoil

$D : 5$
$N : 1503$
Source: `https://archive.ics.uci.edu/ml/datasets/Airfoil+Self-Noise`

Preprocessing: None
Target: Sound pressure in decibels

concrete

$D : 8$
$N : 1030$
Source: `https://archive.ics.uci.edu/ml/datasets/concrete+compressive+strength`
Preprocessing: None
Target: Compressive Strength

parkinsons

$D : 16$
$N : 5875$
Source: `https://archive.ics.uci.edu/ml/datasets/Parkinsons+Telemonitoring`
Preprocessing: Drop the first 5 columns as they are not used in the original problem
Target: Total udpr

bikeshare

$D : 15$
$N : 17379$
Source: `https://archive.ics.uci.edu/ml/datasets/Bike+Sharing+Dataset`
Preprocessing: None
Target: Number of bike shares per hour

ct slice

$D : 379$
$N : 53500$
Source: `https://archive.ics.uci.edu/ml/datasets/Relative+location+of+CT+slices+on+axial+axis`
Preprocessing: Drop Patient ID. Drop columns which have constant value throughout entire dataset.
Target: Reference (relative location)

supercond

$D : 81$
$N : 21263$
Source: `https://archive.ics.uci.edu/ml/datasets/Superconductivty+Data`
Preprocessing: None
Target: Critical Temperature

protein

$D : 9$
$N : 45730$
Source: `https://archive.ics.uci.edu/ml/datasets/Physicochemical+Properties+of+Protein+Tertiary+Structure`
Preprocessing: $\log(1 + y)$ transform for target y
Target: RMSD

buzz

$D : 77$
$N : 583250$
Source: `https://archive.ics.uci.edu/ml/datasets/Buzz+in+social+media+#`
Preprocessing: $\log(1 + y)$ transform of target y
Target: Mean Number of Active Discussion (NAD)

song

D : 90
N : 515345
Source: `https://archive.ics.uci.edu/ml/datasets/Buzz+in+social+media+#`
Preprocessing: None
Target: Year of song release

abalone

D : 9
N : 4177
Source: `https://web.archive.org/web/*/http://www.liacc.up.pt/~ltorgo/Regression/*`
Preprocessing: None
Target: Number of Rings

creep

D : 30
N : 2066
Source: `http://www.phase-trans.msm.cam.ac.uk/map/data/materials/creeprupt-b.html#down`
Preprocessing: None
Target: Rupture stress

ailerons

D : 40
N : 7154
Source: `https://web.archive.org/web/*/http://www.liacc.up.pt/~ltorgo/Regression/*`
Preprocessing: None
Target: goal

References

[1] M. Lázaro-Gredilla, "Bayesian warped Gaussian processes," in *Advances in Neural Information Processing Systems (NIPS)*, 2012.

[2] E. Snelson, Z. Ghahramani, and C. E. Rasmussen, "Warped Gaussian processes," in *Advances in Neural Information Processing Systems (NIPS)*, 2004.

[3] C. E. Rasmussen, "Gaussian processes in machine learning," in *Advanced lectures on machine learning*, Springer, 2004.

[4] R. B. Gramacy, *Bayesian treed Gaussian process models*. PhD dissertation, 2005.