A variational approximate posterior for the deep Wishart process

Sebastian W. Ober [1] **Laurence Aitchison** [2]

Abstract

Recent work introduced deep kernel processes as an entirely kernel-based alternative to NNs (Aitchison et al. 2020). Deep kernel processes flexibly learn good top-layer representations by alternately sampling the kernel from a distribution over positive semi-definite matrices and performing nonlinear transformations. A particular deep kernel process, the deep Wishart process (DWP), is of particular interest because its prior is equivalent to deep Gaussian process (DGP) priors. However, inference in DWPs has not yet been possible due to the lack of sufficiently flexible distributions over positive semi-definite matrices. Here, we give a novel approach to obtaining flexible distributions over positive semi-definite matrices by generalising the Bartlett decomposition of the Wishart probability density. We use this new distribution to develop an approximate posterior for the DWP that includes dependency across layers. We develop a doubly-stochastic inducing-point inference scheme for the DWP and show experimentally that inference in the DWP gives improved performance over doing inference in a DGP with the equivalent prior.

1. Introduction

The successes of modern deep learning have highlighted that good performance on tasks such as image classification (Krizhevsky et al., 2012) requires deep models with lower layers that have the flexibility to learn good representations. Up until very recently, this was only possible in feature-based methods such as neural networks (NNs). Kernel methods did not have this flexibility because the kernel could be modified only using a few kernel hyperparameters. However, with the advent of deep kernel processes (DKPs; Aitchison et al., 2020), we now have deep kernel methods that offer neural-network like flexibility in the kernel / top-layer representation. DKPs introduce this flexibility by taking the kernel from the previous layer, then sampling from a Wishart or inverse Wishart centered on that kernel, followed by a nonlinear transformation. The sampling and nonlinear transformation steps are repeated multiple times to form a deep architecture. Remarkably, deep Gaussian processes (DGPs; Damianou & Lawrence, 2013; Salimbeni & Deisenroth, 2017), standard Bayesian NNs, infinite-width Bayesian NNs (neural network Gaussian processes or NNGPs; Lee et al., 2017; Matthews et al., 2018; Novak et al., 2018; Garriga-Alonso et al., 2018) and infinite NNs with finite width bottlenecks (Aitchison, 2019) can be written as DKPs (Aitchison et al., 2020). In e.g. DGPs the random variables used in variational inference are ultimately features, even though a kernel is computed as a function of the features. By contrast, in a DKP, there are no features *at all*: the only random variables are the positive semi-definite kernel matrices themselves, which are sampled directly from distributions over positive semi-definite matrices such as the (inverse) Wishart.

Aitchison et al. (2020) argued that DKPs should have considerable advantages over related feature-based models, because feature-based models have pervasive symmetries in the true posterior, which are difficult to capture in standard variational approximate posteriors. For instance, in a neural network, it is possible to permute rows and columns of weight matrices, such that the activations at a given layer are permuted, but the network's overall input-output function remains the same (MacKay, 1992; Sussmann, 1992; Bishop et al., 1995). These permutations result in networks with the same output probability density under the true posterior, but with different probability densities under standard variational approximate posteriors, which are generally unimodal. However, these issues do not arise with DKPs, because all permutations of the hidden units correspond to the same kernel (see App. D in Aitchison et al. (2020)).

Deep Wishart processes (DWPs) are the most important DKP because their prior is equivalent to the DGP prior. While Aitchison et al. (2020) showed this equivalence, they were not able to do inference in DWPs because they were not able to find a sufficiently flexible distribution over positive semi-definite matrices to form the basis of an approximate posterior. Instead, they were forced to work with a different DKP: the deep *inverse* Wishart processes

[1]Department of Engineering, University of Cambridge, Cambridge, United Kingdom [2]Department of Computer Science, University of Bristol, Bristol, United Kingdom. Correspondence to: Sebastian W. Ober <swo25@cam.ac.uk>.

Presented at the ICML 2021 Workshop on Uncertainty and Robustness in Deep Learning.,

(DIWPs), which was easier because the inverse Wishart itself forms a suitable approximate posterior. We show how to create a sufficiently flexible approximate posterior for DWPs, thereby enabling us to compare directly to the equivalent DGPs. In particular, our contributions are:

- We develop a new family of flexible distributions over positive semi-definite matrices by generalising the Bartlett decomposition.
- We use this distribution to develop an effective approximate posterior for the deep Wishart process which incorporates dependency across layers.
- We develop a doubly stochastic inducing-point inference scheme for the DWP. While the derivation mostly follows that for deep inverse Wishart processes (Aitchison et al., 2020), we need to give a novel scheme for sampling the test/training points conditioned on the inducing points, as this is very different in the DWP compared to the previous DIWP.
- We empirically compare DGP and DWP inference with the *exact same prior*. This was not possible in Aitchison et al. (2020) as they only derived an inference scheme for deep inverse Wishart processes, whose prior is not equivalent to a DGP prior. As expected, DWPs show strong benefits over DGPs.

2. Background

2.1. Wishart distribution

The Wishart, $\mathcal{W}(\mathbf{\Sigma}, \nu)$, is a distribution over positive semi-definite $P \times P$ matrices, $\mathbf{G}$, with positive definite scale parameter $\mathbf{\Sigma} \in \mathbb{R}^{P\times P}$ and a positive, integer-valued degrees-of-freedom parameter, ν. The Wishart distribution is defined by taking ν vectors $\mathbf{n}_\lambda \in \mathbf{R}^P$ sampled from a zero-mean Gaussian. These vectors can be generated from standard Gaussian vectors, $\boldsymbol{\xi}_\lambda$, by transforming them with the Cholesky, $\mathbf{L}$ of the scale parameter, $\mathbf{\Sigma} = \mathbf{L}\mathbf{L}^T$,

$$\mathbf{L}\boldsymbol{\xi}_\lambda = \mathbf{n}_\lambda \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}) \text{ where } \boldsymbol{\xi}_\lambda \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \quad (1)$$

Both $\mathbf{n}_\lambda$ and $\boldsymbol{\xi}_\lambda$ can be stacked to form $P \times \nu$ matrices, $\mathbf{N}$ and $\mathbf{\Xi}$,

$$\mathbf{N} = (\mathbf{n}_1 \quad \mathbf{n}_2 \quad \cdots \quad \mathbf{n}_\nu), \quad (2)$$
$$\mathbf{\Xi} = (\boldsymbol{\xi}_1 \quad \boldsymbol{\xi}_2 \quad \cdots \quad \boldsymbol{\xi}_\nu). \quad (3)$$

Wishart samples are defined by taking the sum of the outer products of the $\mathbf{n}_\lambda$'s, which can be written as a matrix multiplication:

$$\sum_{\lambda=1}^{\nu} \mathbf{n}_\lambda \mathbf{n}_\lambda^T = \mathbf{L}\mathbf{\Xi}\mathbf{\Xi}^T\mathbf{L} = \mathbf{L}\mathbf{Z}\mathbf{L}^T = \mathbf{G} \sim \mathcal{W}(\mathbf{\Sigma}, \nu), \quad (4)$$

where $\mathbf{Z} = \mathbf{\Xi}\mathbf{\Xi}^T$ is a sample from a standard Wishart (i.e. one with an identity scale parameter),

$$\sum_{\lambda=1}^{\nu} \boldsymbol{\xi}_\lambda \boldsymbol{\xi}_\lambda^T = \mathbf{\Xi}\mathbf{\Xi}^T = \mathbf{Z} \sim \mathcal{W}(\mathbf{I}, \nu). \quad (5)$$

2.2. Bartlett decomposition

However, sampling $\mathbf{\Xi}$ can be computationally expensive for very large values of ν. Instead, it is possible to sample a Wishart by writing down the distribution over the Cholesky of $\mathbf{Z}$, denoted $\mathbf{A}$ (Bartlett, 1934). Taking $\mathbf{Z} = \mathbf{A}\mathbf{A}^T$, the distribution over $\mathbf{A}$ is

$$\mathrm{P}\left(A_{jj}^2\right) = \text{Gamma}\left(A_{jj}^2; \alpha{=}\tfrac{\nu-j+1}{2}, \beta{=}\tfrac{1}{2}\right), \quad (6a)$$
$$\mathrm{P}\left(A_{j>k}\right) = \mathcal{N}\left(A_{jk}; 0, 1\right)., \quad (6b)$$

i.e. the square of the on-diagonal elements are gamma and the off-diagonal elements are IID standard Gaussian.

2.3. Deep Gaussian proceses (DGPs)

In a DGP, we sample features, $\mathbf{F}_\ell$, from a Gaussian process, conditioned on features from the previous layer,

$$\mathrm{P}\left(\mathbf{F}_\ell | \mathbf{F}_{\ell-1}\right) = \textstyle\prod_{\lambda=1}^{\nu_\ell} \mathcal{N}\left(\mathbf{f}_\lambda^\ell; \mathbf{0}, \mathbf{K}_{\text{features}}^\ell(\mathbf{F}_{\ell-1})\right), \quad (7a)$$
$$\mathrm{P}\left(\mathbf{Y} | \mathbf{F}_{L+1}\right) = \textstyle\prod_{\lambda=1}^{\nu_{L+1}} \mathcal{N}\left(\mathbf{y}_\lambda; \mathbf{f}_\lambda^{L+1}, \sigma^2\mathbf{I}\right), \quad (7b)$$

where $\mathbf{F}_0 = \mathbf{X} \in \mathbb{R}^{P\times\nu_0}$ is the input and $\mathbf{F}_\ell \in \mathbb{R}^{P\times\nu_\ell}$ are the features. We use P for the number of input points and ν_ℓ for the width of layer ℓ; thus ν_0 is the number of inputs and ν_{L+1} is the number of outputs. In addition, the features and targets can be written as a stack of vectors, $\mathbf{f}_\lambda^\ell \in \mathbb{R}^P$ and $\mathbf{y}_\lambda \in \mathbb{R}^P$, i.e.

$$\mathbf{F}_\ell = (\mathbf{f}_1^\ell \quad \mathbf{f}_2^\ell \quad \cdots \quad \mathbf{f}_{\nu_\ell}^\ell), \quad (8)$$
$$\mathbf{Y} = (\mathbf{y}_1 \quad \mathbf{y}_2 \quad \cdots \quad \mathbf{y}_{\nu_{L+1}}). \quad (9)$$

The function $\mathbf{K}_{\text{features}}^\ell(\mathbf{F}_{\ell-1})$ takes the features at the previous layer and returns the corresponding $P \times P$ kernel matrix. We mainly consider isotropic kernels such as the squared exponential, which can be written as a function of $R_{ij}^{\ell-1}$, the distance between input features i and j:

$$K_{\text{features};ij}^\ell = k_\ell(R_{ij}^{\ell-1}), \quad (10)$$
$$R_{ij}^{\ell-1} = \tfrac{1}{N_\ell}\textstyle\sum_{\lambda=1}^{N_\ell}\left(F_{i\lambda}^{\ell-1} - F_{j\lambda}^{\ell-1}\right)^2. \quad (11)$$

2.4. Deriving equivalent deep Wishart processes

Following Aitchison et al. (2020), we show how the DGP model of Eq. (7) can be expressed as a deep Wishart process. We first consider the $P \times P$ Gram matrices defined as

$$\mathbf{G}_\ell = \tfrac{1}{\nu_\ell}\mathbf{F}_\ell\mathbf{F}_\ell^T = \tfrac{1}{\nu_\ell}\sum_{\lambda=1}^{\nu_\ell}\mathbf{f}_\lambda^\ell(\mathbf{f}_\lambda^\ell)^T, \quad (12)$$

where $\mathbf{f}_\lambda^\ell$ are IID and multivariate-Gaussian distributed conditioned on the features at the previous layer (Eq. 7a). Thus, $\mathbf{G}_\ell$ follows the definition of the Wishart (Eq. 4), and we can sample $\mathbf{G}_\ell$ directly,

$$\mathrm{P}\left(\mathbf{G}_\ell|\mathbf{F}_{\ell-1}\right) = \mathcal{W}\left(\mathbf{G}_\ell; \tfrac{1}{\nu_\ell}\mathbf{K}^\ell_{\text{features}}(\mathbf{F}_{\ell-1}), \nu_\ell\right). \quad (13)$$

To work entirely with Gram matrices rather than features, we need to be able to compute the kernel, $\mathbf{K}^\ell_{\text{features}}(\mathbf{F}_{\ell-1})$, as a function of the Gram matrix at the previous layer, $\mathbf{G}_{\ell-1}$. Remarkably, for isotropic kernels it *is* possible to obtain $\mathbf{K}_\ell(\cdot)$, which takes the Gram matrix from the previous layer and returns the same kernel matrix as that returned by applying $\mathbf{K}_{\text{features}}$ to the features from the previous layer (see Aitchison et al. (2020) for details):

$$\mathbf{K}^\ell_{\text{features}}(\mathbf{F}_{\ell-1}) = \mathbf{K}_\ell(\mathbf{G}_{\ell-1}) = \mathbf{K}_\ell(\tfrac{1}{\nu_\ell}\mathbf{F}_{\ell-1}\mathbf{F}_{\ell-1}^T). \quad (14)$$

By using the equivalent kernel written as a function of the Gram matrix at the previous layer, we can entirely eliminate intermediate-layer features, giving a *deep Wishart process*,

$$\mathrm{P}\left(\mathbf{G}_\ell|\mathbf{G}_{\ell-1}\right) = \mathcal{W}\left(\mathbf{G}_\ell; \tfrac{1}{\nu_\ell}\mathbf{K}_\ell(\mathbf{G}_{\ell-1}), \nu_\ell\right), \quad (15a)$$

where we set $\mathbf{G}_0 = \frac{1}{\nu_0}\mathbf{X}\mathbf{\Omega}\mathbf{X}^T$, where $\mathbf{\Omega}$ is a learned diagonal matrix allowing for automatic relevance determination, and

$$\mathrm{P}\left(\mathbf{F}_{L+1}|\mathbf{G}_L\right) = \textstyle\prod_{\lambda=1}^{\nu_{L+1}} \mathcal{N}\left(\mathbf{f}_\lambda^{L+1}; \mathbf{0}, \mathbf{K}_\ell\left(\mathbf{G}_L\right)\right), \quad (15b)$$

$$\mathrm{P}\left(\mathbf{Y}|\mathbf{F}_{L+1}\right) = \textstyle\prod_{\lambda=1}^{\nu_{L+1}} \mathcal{N}\left(\mathbf{y}_\lambda; \mathbf{0}, \sigma^2\mathbf{I}\right). \quad (15c)$$

2.5. DWP formulation captures true-posterior symmetries while DGP does not

We now have two equivalent generative models: one phrased in terms of features, $\mathbf{F}_\ell$ and another phrased in terms of Gram matrices, $\mathbf{G}_\ell$. Is there any reason to prefer one over the other? It turns out that there is. In particular, consider a transformation of the features, $\mathbf{F}'_\ell = \mathbf{U}\mathbf{F}_\ell$ where $\mathbf{U}$ is a unitary matrix, such that $\mathbf{U}\mathbf{U}^T = \mathbf{I}$. Remarkably, the true posterior is symmetric under these transformations, in the sense that all unitary transformations of the underlying features have the exact same true-posterior probability density (see Aitchison et al., 2020, Appendix D.2), that is, $\mathrm{P}\left(\mathbf{F}'_1, \ldots, \mathbf{F}'_L, \mathbf{F}_{L+1}|\mathbf{X}, \mathbf{Y}\right) = \mathrm{P}\left(\mathbf{F}_1, \ldots, \mathbf{F}_L, \mathbf{F}_{L+1}|\mathbf{X}, \mathbf{Y}\right)$.

It would be desirable for variational approximate posteriors to capture these true posterior symmetries. However, the usual family of Gaussian approximate posteriors over features fails to capture these symmetries because they use non-zero means. Worryingly, the failure to capture these symmetries can bias variational inference to focus on low-mass areas of the true posterior (Moore, 2016; Pourzanjani et al., 2017).

In contrast, the deep Wishart process (DWP) sidesteps this issue by phrasing posteriors entirely in terms of Gram matrices, $\mathbf{G}_\ell = \frac{1}{\nu_\ell}\mathbf{F}_\ell\mathbf{F}_\ell^T$. Critically, the Gram matrix is invariant to unitary transformations of the features,

$$\mathbf{G}_\ell = \tfrac{1}{N_\ell}\mathbf{F}_\ell\mathbf{F}_\ell^T = \tfrac{1}{N_\ell}\mathbf{F}_\ell\mathbf{U}_\ell\mathbf{U}_\ell^T\mathbf{F}_\ell^T = \tfrac{1}{N_\ell}\mathbf{F}'_\ell\mathbf{F}'^T_\ell. \quad (16)$$

As such, DWP approximate posteriors in terms of $\mathbf{G}_\ell$ implicitly respect this unitary symmetry over the features.

3. Methods

As detailed in Aitchison et al. (2020), the key difficulty in obtaining a variational inference scheme for DWPs is the difficulty of providing a sufficiently flexible approximate posterior. In particular, as we are working with a probabilistic process, the number of input points, P, can be arbitrarily large, and thus there is always the possibility that $\nu < P$ and hence that our sampled Gram matrices are low-rank. We therefore need to form flexible variational approximate posteriors over rank ν Gram matrices. An obvious first choice is the Wishart distribution itself with degrees of freedom ν, so as to match the rank of matrices sampled from the prior. However, for fixed degrees of freedom the Wishart variance,

$$\mathbb{V}\left[G_{ij}\right] = \nu\left(\Sigma_{ij}^2 + \Sigma_{ii}\Sigma_{jj}\right) \quad (17)$$

cannot be specified independently of the mean, which is essential for a variational approximate posterior that can flexibly capture potentially narrow true posteriors. An alternative approach would be to work with a non-central Wishart, which is defined by taking $\mathbf{\Xi}$, which is IID standard Gaussian in the case of the Wishart, to have non-zero mean. However, the non-central Wishart has a probability density function that is too difficult to evaluate in the inner loop of a deep learning algorithm (Koev & Edelman, 2006). Instead, we develop a new Generalised Singular Wishart distribution, based on the Bartlett decomposition, which modifies the Wishart to give independent control over the mean and variance of sampled matrices.

Our Generalised Singular Wishart distribution takes additional parameters $\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\mu}$, and $\boldsymbol{\sigma}$, leading to a density

$$\mathrm{P}\left(\mathbf{G}\right) = \mathcal{W}\left(\mathbf{G}; \mathbf{\Sigma}, \nu, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\mu}, \boldsymbol{\sigma}\right). \quad (18)$$

Inspired by the form of the approximate posterior used for the deep *inverse* Wishart process from Aitchison et al. (2020), we create an approximate posterior for the DWP layers that introduces between-layer correlations:

$$\mathrm{Q}\left(\mathbf{G}_\ell|\mathbf{G}_{\ell-1}\right) = \mathcal{W}\big(\mathbf{G}_\ell; (1-p_\ell)\tfrac{1}{\nu_\ell}\mathbf{K}(\mathbf{G}_{\ell-1}) + \quad (19)$$

$$p_\ell\mathbf{V}_\ell\mathbf{V}_\ell^T, \nu_\ell, \boldsymbol{\alpha}_\ell, \boldsymbol{\beta}_\ell, \boldsymbol{\mu}_\ell, \boldsymbol{\sigma}_\ell\big). \quad (20)$$

The approximate posterior parameters are $\{\mathbf{V}_\ell, \boldsymbol{\alpha}_\ell, \boldsymbol{\beta}_\ell, \boldsymbol{\mu}_\ell, \boldsymbol{\sigma}_\ell, p_\ell\}_{\ell=1}^L$, where $0 < p_\ell < 1$ is a scalar, and $\mathbf{V}_\ell \in \mathbb{R}^{p\times p}$.

Table 1. ELBOs and log-likelihoods for UCI datasets from (Gal & Ghahramani, 2015) for a five-layer network. See Appendix H for other depths. Significantly better results are highlighted.

metric	boston	concrete	energy	kin8nm	naval	power	protein	wine	yacht
ELBO	**-0.38 ± 0.01**	**-0.49 ± 0.00**	**1.41 ± 0.00**	**-0.14 ± 0.00**	3.62 ± 0.07	**0.03 ± 0.00**	**-1.01 ± 0.00**	-1.19 ± 0.00	**1.63 ± 0.01**
	-0.46 ± 0.01	-0.55 ± 0.00	1.37 ± 0.00	-0.18 ± 0.00	**3.74 ± 0.08**	0.01 ± 0.00	-1.02 ± 0.00	-1.19 ± 0.00	1.30 ± 0.02
LL	**-2.39 ± 0.04**	**-3.13 ± 0.01**	-0.70 ± 0.03	**1.40 ± 0.01**	8.20 ± 0.04	**-2.77 ± 0.01**	-2.73 ± 0.00	-0.96 ± 0.01	**-0.46 ± 0.12**
	-2.48 ± 0.04	-3.18 ± 0.01	-0.73 ± 0.03	1.38 ± 0.01	8.15 ± 0.07	-2.79 ± 0.01	-2.74 ± 0.01	-0.96 ± 0.01	-0.77 ± 0.03

We provide additional details of the methods used, including the details of the Generalised Singular Wishart, as well as the doubly-stochastic inference scheme using it, in App. A.

4. Results

The DWP prior is equivalent to a DGP prior (Sec. 2.4) (Aitchison et al., 2020); in a DGP, we use features as the latent variables, whereas in the DWP we use Gram matrices. Using Gram matrices should be beneficial as the true posteriors are expected to be simpler than in the DGP (Sec. 2.5).

We trained a DWP and a DGP with the same generative models using squared exponential kernels. We trained both models for 20000 gradient steps using the Adam optimizer (Kingma & Ba, 2014); see Appendix G for details on the experimental setup. We report ELBOs and test log likelihoods for depth 5 in Table 1 with other depths in Appendix H. We found strong benefits for the DWP, which are especially evident if we look at the ELBOs and smaller datasets (boston, concrete, energy, wine and yacht). On larger datasets the benefits are smaller as accurate uncertainty modelling is less relevant. Note that we compared against the recently introduced DGP method based on global inducing points (Ober & Aitchison, 2020), which were particularly important in our setting because we use an architecture without skip connections for equivalence of the DGP and DWP. Standard DSVI has considerable difficulties with optimizing the approximate posterior, which was alleviated in Salimbeni & Deisenroth (2017) by modifying the prior to introduce skip connections.

5. Related Work

The DWP prior was introduced by Aitchison et al. (2020). However, they were not able to do variational inference with the DWP because they did not have a sufficiently flexible approximate posterior over positive semi-definite matrices. Instead, they were forced to work with a deep *inverse* Wishart process, which is easier because the inverse Wishart itself is a suitable approximate posterior. Here, we give a flexible generalised Wishart distribution over positive semi-definite matrices which is suitable for use as a variational approximate posterior in the DWP. As the DWP prior is equivalent to a DGP prior, we were able to directly compare DGP and DWP inference in models with the exact same prior. Such a comparison with equivalent priors was not possible in Aitchison et al. (2020), because their deep *inverse* Wishart process priors are not equivalent to DGP priors.

There is an alternative line of work using *generalised* Wishart processes (Wilson & Ghahramani, 2010, as opposed to our *deep* Wishart processes). A generalised Wishart process specifies a distribution over an infinite number of finite-dimensional Wishart-distributed matrices. These matrices might represent e.g. the noise covariance in a dynamical system, in which case there might be an infinite number of such matrices, one for each time or location in the state-space (Wilson & Ghahramani, 2010; Heaukulani & van der Wilk, 2019; Jorgensen et al., 2020). In contrast, the Wishart process (Dawid, 1981; Bru, 1991) describes finite dimensional marginals of a single, potentially infinite dimensional matrix. In our context, we stack (non-generalised) Wishart processes to form a deep Wishart process. These generalised Wishart priors do not have the flexibility to capture a DGP prior because the underlying features at all locations are jointly multivariate Gaussian (Sec. 4 in Wilson & Ghahramani, 2010) and therefore lack the required nonlinearities between layers. Further, inference is also radically different. In particular, work on the generalised Wishart performs inference on the underlying multivariate Gaussian feature vectors (e.g. Eq. 15-18 in Wilson & Ghahramani 2010). Unfortunately, variational approximate posteriors defined over multivariate Gaussian feature vectors fail to capture symmetries in the true posterior (Sec. 2.5). In contrast, we define approximate posteriors directly over the symmetric positive semi-definite Gram matrices themselves, which required us to develop new, more flexible distributions over these matrices.

6. Conclusions

We introduced a flexible distribution over positive semi-definite matrices which formed the basis of a variational approximate posterior for the deep Wishart process, which adapted the doubly stochastic variational inference scheme from Aitchison et al. (2020). Thus, we were able to directly compare the performance for inference in a DWP vs. DGP with exactly the same prior. We found considerable benefits, both in terms of predictive performance and the ELBO from doing inference in the DWP rather than the DGP.

References

Aitchison, L. Why bigger is not always better: on finite and infinite neural networks. *arXiv preprint arXiv:1910.08013*, 2019.

Aitchison, L., Yang, A. X., and Ober, S. W. Deep kernel processes. *arXiv preprint arXiv:2010.01590*, 2020.

Bartlett, M. S. On the theory of statistical regression. *Proceedings of the Royal Society of Edinburgh*, 53:260–283, 1934.

Bishop, C. M. et al. *Neural networks for pattern recognition*. Oxford university press, 1995.

Bru, M.-F. Wishart processes. *Journal of Theoretical Probability*, 4(4):725–751, 1991.

Coelho, L. P. Jug: Software for parallel reproducible computation in python. *Journal of Open Research Software*, 5(1), 2017.

Damianou, A. and Lawrence, N. Deep gaussian processes. In *Artificial Intelligence and Statistics*, pp. 207–215, 2013.

Dawid, A. P. Some matrix-variate distribution theory: notational considerations and a Bayesian application. *Biometrika*, 68(1):265–274, 1981.

Eaton, M. L. et al. The wishart distribution. In *Multivariate Statistics*, pp. 302–333. Institute of Mathematical Statistics, 2007.

Gal, Y. and Ghahramani, Z. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. *arXiv:1506.02142*, 2015.

Garriga-Alonso, A., Rasmussen, C. E., and Aitchison, L. Deep convolutional networks as shallow gaussian processes. *arXiv preprint arXiv:1808.05587*, 2018.

Heaukulani, C. and van der Wilk, M. Scalable bayesian dynamic covariance modeling with variational wishart and inverse wishart processes. In *Advances in Neural Information Processing Systems*, pp. 4582–4592, 2019.

Jorgensen, M., Deisenroth, M., and Salimbeni, H. Stochastic differential equations with variational wishart diffusions. In *International Conference on Machine Learning*, pp. 4974–4983. PMLR, 2020.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Koev, P. and Edelman, A. The efficient evaluation of the hypergeometric function of a matrix argument. *Mathematics of Computation*, 75(254):833–846, 2006.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105, 2012.

Lee, J., Bahri, Y., Novak, R., Schoenholz, S. S., Pennington, J., and Sohl-Dickstein, J. Deep neural networks as gaussian processes. *arXiv preprint arXiv:1711.00165*, 2017.

MacKay, D. J. A practical bayesian framework for backpropagation networks. *Neural computation*, 4(3):448–472, 1992.

Mathai, A. M. *Jacobians of matrix transformation and functions of matrix arguments*. World Scientific Publishing Company, 1997.

Mathai, A. M. and Haubold, H. J. *Special functions for applied scientists*. Springer, 2008.

Matthews, A. G. d. G., Rowland, M., Hron, J., Turner, R. E., and Ghahramani, Z. Gaussian process behaviour in wide deep neural networks. *arXiv preprint arXiv:1804.11271*, 2018.

Moore, D. A. Symmetrized variational inference. In *NIPS Workshop on Advances in Approximate Bayesian Inferece*, volume 4, pp. 31, 2016.

Novak, R., Xiao, L., Lee, J., Bahri, Y., Yang, G., Hron, J., Abolafia, D. A., Pennington, J., and Sohl-Dickstein, J. Bayesian deep convolutional networks with many channels are gaussian processes. *arXiv preprint arXiv:1810.05148*, 2018.

Ober, S. W. and Aitchison, L. Global inducing point variational posteriors for bayesian neural networks and deep gaussian processes. *arXiv preprint arXiv:2005.08140*, 2020.

Pourzanjani, A. A., Jiang, R. M., and Petzold, L. R. Improving the identifiability of neural networks for bayesian inference. In *NIPS Workshop on Bayesian Deep Learning*, volume 4, pp. 31, 2017.

Rezende, D. J., Mohamed, S., and Wierstra, D. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

Roeder, G., Wu, Y., and Duvenaud, D. Sticking the landing: Simple, lower-variance gradient estimators for variational inference. In *Advances in Neural Information Processing Systems*, 2017.

Salimbeni, H. and Deisenroth, M. Doubly stochastic variational inference for deep gaussian processes. In *Advances in Neural Information Processing Systems*, pp. 4588–4599, 2017.

Srivastava, M. S. et al. Singular wishart and multivariate beta distributions. *The Annals of Statistics*, 31(5):1537–1560, 2003.

Sussmann, H. J. Uniqueness of the weights for minimal feedforward nets with a given input-output map. *Neural networks*, 5(4):589–593, 1992.

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

Wilson, A. G. and Ghahramani, Z. Generalised Wishart processes. *arXiv preprint arXiv:1101.0240*, 2010.

A. Full Methods

Here we describe in full the methods introduced in this paper. We begin by discussing the singular Bartlett decomposition (App. A.1), followed by defining the Generalised Singular Wishart (App. A.2). We proceed by writing our approximate posterior (App. A.3), continuing by describing in detail the doubly-stochastic inducing point scheme we use to perform inference in the model, including an algorithm (App. A.4), and finally finishing by discussing the computational complexity of the method in App. A.5.

A.1. Singular Bartlett decomposition

To define the Generalised Singular Wishart distribution, we first need to generalise the Wishart construction to potentially singular matrices (i.e. those for which $\nu < P$). Remembering that $\mathbf{Z} = \mathbf{A}\mathbf{A}^T$, in the singular case $\mathbf{A}$ is given by

$$\mathbf{A} = \begin{pmatrix} A_{11} & \dots & 0 \\ \vdots & \ddots & \vdots \\ A_{\nu 1} & \dots & A_{\nu\nu} \\ \vdots & \vdots & \vdots \\ A_{P1} & \dots & A_{P\nu} \end{pmatrix}, \tag{21}$$

$$\mathrm{P}\left(A_{jj}^2\right) = \text{Gamma}\left(A_{jj}^2; \tfrac{\nu-j+1}{2}, \tfrac{1}{2}\right) \tag{22}$$

$$\mathrm{P}\left(A_{i>j}\right) = \mathcal{N}\left(A_{ij}; 0, 1\right). \tag{23}$$

The full probability density is

$$\mathrm{P}\left(\mathbf{A}\right) = \prod_{j=1}^{\min(p,\nu)} 2A_{jj}\text{Gamma}\left(A_{jj}^2; \tfrac{\nu-j+1}{2}, \tfrac{1}{2}\right) \prod_{i=j+1}^{p} \mathcal{N}\left(A_{ij}; 0, 1\right). \tag{24}$$

where the $2A_{jj}$ accounts for the Jacobian for the transformation from A_{jj}^2 to A_{jj}. Now we transform from $\mathbf{A}$ to $\mathbf{G}$ using the Jacobians in Appendix C and Appendix D (i.e. by first transforming $\mathbf{A} \to \mathbf{L}\mathbf{A}$ then transforming $\mathbf{L}\mathbf{A} \to \mathbf{L}\mathbf{A}\mathbf{A}^T\mathbf{L}^T$.)

$$\mathrm{P}\left(\mathbf{G}\right) = \left(\prod_{j=1}^{p} \frac{1}{L_{jj}^{\min(j,\nu)}}\right) \prod_{j=1}^{\min(p,\nu)} \frac{\text{Gamma}\left(A_{jj}^2; \frac{\nu-j+1}{2}, \frac{1}{2}\right)}{A_{jj}^{p-j} L_{jj}^{p-j+1}} \prod_{i=j+1}^{p} \mathcal{N}\left(A_{ij}; 0, 1\right). \tag{25}$$

In Appendix E we prove that this corresponds to the known full rank and singular Wishart distribution.

A.2. Generalised Singular Wishart distributions

Our goal is to develop a generalisation of the Wishart distribution, $\mathcal{W}\left(\mathbf{\Sigma}, \nu, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\mu}, \boldsymbol{\sigma}\right)$ based on the Bartlett decomposition, which will turn out to have additional parameters, $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ for the on-diagonal elements of $\mathbf{A}$ and $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ for the off-diagonal elements. As we will be using this distribution for the approximate posterior, we write densities under this distribution as $\mathrm{Q}\left(\cdot\right)$,

$$\mathrm{Q}\left(\mathbf{G}\right) = \mathcal{W}\left(\mathbf{G}; \mathbf{\Sigma}, \nu, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\mu}, \boldsymbol{\sigma}\right). \tag{26}$$

To specify this distribution, we generalise the Bartlett decomposition using

$$\mathrm{Q}\left(A_{jj}^2\right) = \text{Gamma}\left(A_{jj}^2; \alpha_j, \beta_j\right), \qquad \text{for } j \leq \nu \tag{27a}$$

$$\mathrm{Q}\left(A_{i>j}\right) = \mathcal{N}\left(A_{ij}; \mu_{ij}, \sigma_{ij}^2\right). \tag{27b}$$

Thus, the full probability density for $\mathbf{A}$ is,

$$\mathrm{Q}\left(\mathbf{A}\right) = \prod_{j=1}^{\nu} 2A_{jj}\text{Gamma}\left(A_{jj}^2; \alpha_j, \beta_j\right) \prod_{i=j+1}^{p} \mathcal{N}\left(A_{ij}; \mu_{ij}, \sigma_{ij}^2\right), \tag{28}$$

and applying the same transformations and Jacobians as in the previous section, this implies a distribution over $\mathbf{G}$ of,

$$\mathrm{Q}\left(\mathbf{G}\right) = \left(\prod_{j=1}^{p} \frac{1}{L_{jj}^{\min(j,\nu)}}\right) \prod_{j=1}^{\min(p,\nu)} \frac{1}{A_{jj}^{p-j} L_{jj}^{p-j+1}} \mathrm{Gamma}\left(A_{jj}^2; \alpha_j, \beta_j\right) \prod_{i=j+1}^{p} \mathcal{N}\left(A_{ij}; \mu_{ij}, \sigma_{ij}^2\right). \tag{29}$$

A.3. Full approximate posterior distribution

Inspired by the across-layer dependencies in Aitchison et al. (2020), we use an Generalised Wishart approximate posterior for $\mathbf{G}_\ell$ with dependencies across layers,

$$\mathrm{Q}\left(\mathbf{G}_\ell|\mathbf{G}_{\ell-1}\right) = \mathcal{W}\left(\mathbf{G}_\ell; (1-p_\ell)\tfrac{1}{\nu_\ell}\mathbf{K}(\mathbf{G}_{\ell-1}) + p_\ell \mathbf{V}_\ell \mathbf{V}_\ell^T,\, \nu_\ell,\, \boldsymbol{\alpha}_\ell,\, \boldsymbol{\beta}_\ell,\, \boldsymbol{\mu}_\ell,\, \boldsymbol{\sigma}_\ell\right), \tag{30a}$$

where the approximate posterior parameters are $\{\mathbf{V}_\ell, \boldsymbol{\alpha}_\ell, \boldsymbol{\beta}_\ell, \boldsymbol{\mu}_\ell, \boldsymbol{\sigma}_\ell, p_\ell\}_{\ell=1}^L$, where $0 < p_\ell < 1$ is a scalar, and $\mathbf{V}_\ell \in \mathbb{R}^{p\times p}$. Note that the exact form for the across layer dependencies in our approximate posterior is inspired by the approximate posterior in (Aitchison et al., 2020).

A.4. Doubly stochastic inducing-point variational inference in deep inverse Wishart processes

For efficient inference in high-dimensional problems, we take inspiration from the DGP literature (Salimbeni & Deisenroth, 2017) by considering doubly-stochastic inducing-point deep Wishart processes. We begin by decomposing all variables into inducing and training (or test) points $\mathbf{X}_\mathrm{i} \in \mathbb{R}^{P_\mathrm{i}\times N_0}$ and $\mathbf{X}_\mathrm{t} \in \mathbb{R}^{P_\mathrm{t}\times N_0}$ where P_i is the number of inducing points, and P_t is the number of testing/training points,

$$\mathbf{X} = \begin{pmatrix}\mathbf{X}_\mathrm{i}\\ \mathbf{X}_\mathrm{t}\end{pmatrix}, \qquad \mathbf{F}_{L+1} = \begin{pmatrix}\mathbf{F}_\mathrm{i}^{L+1}\\ \mathbf{F}_\mathrm{t}^{L+1}\end{pmatrix}, \qquad \mathbf{G}_\ell = \begin{pmatrix}\mathbf{G}_{\mathrm{ii}}^\ell & \mathbf{G}_{\mathrm{it}}^\ell\\ \mathbf{G}_{\mathrm{ti}}^\ell & \mathbf{G}_{\mathrm{tt}}^\ell\end{pmatrix}, \tag{31}$$

where e.g. $\mathbf{G}_{\mathrm{ii}}^\ell$ is $P_\mathrm{i}\times P_\mathrm{i}$ and $\mathbf{G}_{\mathrm{it}}^\ell$ is $P_\mathrm{i}\times P_\mathrm{t}$. The full ELBO including latent variables for all the inducing and training points is

$$\mathcal{L} = \mathbb{E}\left[\log \mathrm{P}\left(\mathbf{Y}|\mathbf{F}_{L+1}\right) + \log \frac{\mathrm{P}\left(\{\mathbf{G}_\ell\}_{\ell=1}^L, \mathbf{F}_{L+1}|\mathbf{X}\right)}{\mathrm{Q}\left(\{\mathbf{G}_\ell\}_{\ell=1}^L, \mathbf{F}_{L+1}|\mathbf{X}\right)}\right], \tag{32}$$

where the expectation is taken over $\mathrm{Q}\left(\{\mathbf{G}_\ell\}_{\ell=1}^L, \mathbf{F}_{L+1}|\mathbf{X}\right)$. The prior is given by combining all terms in Eq. (15) for both inducing and test/train inputs,

$$\mathrm{P}\left(\{\mathbf{G}_\ell\}_{\ell=1}^L, \mathbf{F}_{L+1}|\mathbf{X}\right) = \left[\prod_{\ell=1}^L \mathrm{P}\left(\mathbf{G}_\ell|\mathbf{G}_{\ell-1}\right)\right] \mathrm{P}\left(\mathbf{F}_{L+1}|\mathbf{G}_L\right), \tag{33}$$

where the $\mathbf{X}$-dependence enters on the right because we set $\mathbf{G}_0 = \frac{1}{\nu_0}\mathbf{X}\boldsymbol{\Omega}\mathbf{X}^T$. Taking inspiration from Salimbeni & Deisenroth (2017), the full approximate posterior is the product of an approximate posterior over inducing points and the conditional prior for train/test points,

$$\mathrm{Q}\left(\{\mathbf{G}_\ell\}_{\ell=1}^L, \mathbf{F}_{L+1}|\mathbf{X}\right) = \mathrm{Q}\left(\{\mathbf{G}_{\mathrm{ii}}^\ell\}_{\ell=1}^L, \mathbf{F}_\mathrm{i}^{L+1}|\mathbf{X}_\mathrm{i}\right) \mathrm{P}\left(\{\mathbf{G}_{\mathrm{it}}^\ell\}_{\ell=1}^L, \{\mathbf{G}_{\mathrm{tt}}^\ell\}_{\ell=1}^L, \mathbf{F}_\mathrm{t}^{L+1}|\{\mathbf{G}_{\mathrm{ii}}^\ell\}_{\ell=1}^L, \mathbf{F}_\mathrm{i}^{L+1}, \mathbf{X}\right). \tag{34}$$

And the prior can be written in the same form,

$$\mathrm{P}\left(\{\mathbf{G}_\ell\}_{\ell=1}^L, \mathbf{F}_{L+1}|\mathbf{X}\right) = \mathrm{P}\left(\{\mathbf{G}_{\mathrm{ii}}^\ell\}_{\ell=1}^L, \mathbf{F}_\mathrm{i}^{L+1}|\mathbf{X}_\mathrm{i}\right) \mathrm{P}\left(\{\mathbf{G}_{\mathrm{it}}^\ell\}_{\ell=1}^L, \{\mathbf{G}_{\mathrm{tt}}^\ell\}_{\ell=1}^L, \mathbf{F}_\mathrm{t}^{L+1}|\{\mathbf{G}_{\mathrm{ii}}^\ell\}_{\ell=1}^L, \mathbf{F}_\mathrm{i}^{L+1}, \mathbf{X}\right). \tag{35}$$

We discuss the second terms (the conditional prior) in Eq. (39). The first terms (the prior and approximate posteriors over inducing points), are given by combining terms in Eq. (15) and Eq. (30),

$$\mathrm{P}\left(\{\mathbf{G}_{\mathrm{ii}}^\ell\}_{\ell=1}^L, \mathbf{F}_\mathrm{i}^{L+1}|\mathbf{X}_\mathrm{i}\right) = \left[\prod_{\ell=1}^L \mathrm{P}\left(\mathbf{G}_{\mathrm{ii}}^\ell|\mathbf{G}_{\mathrm{ii}}^{\ell-1}\right)\right] \mathrm{P}\left(\mathbf{F}_\mathrm{i}^{L+1}|\mathbf{G}_{\mathrm{ii}}^L\right), \tag{36}$$

$$\mathrm{Q}\left(\{\mathbf{G}_{\mathrm{ii}}^\ell\}_{\ell=1}^L, \mathbf{F}_\mathrm{i}^{L+1}|\mathbf{X}_\mathrm{i}\right) = \left[\prod_{\ell=2}^L \mathrm{Q}\left(\mathbf{G}_{\mathrm{ii}}^\ell|\mathbf{G}_{\mathrm{ii}}^{\ell-1}\right)\right] \mathrm{Q}\left(\mathbf{F}_\mathrm{i}^{L+1}|\mathbf{G}_{\mathrm{ii}}^L\right). \tag{37}$$

Substituting Eqs. (34–37) into the ELBO (Eq. 32), the conditional prior cancels and we obtain,

$$\mathcal{L} = \mathbb{E}\left[\log \mathrm{P}\left(\mathbf{Y}|\mathbf{F}_{\mathrm{t}}^{L+1}\right) + \log \frac{\left[\prod_{\ell=1}^{L} \mathrm{Q}\left(\mathbf{G}_{\mathrm{ii}}^{\ell}|\mathbf{G}_{\mathrm{ii}}^{\ell-1}\right)\right] \mathrm{Q}\left(\mathbf{F}_{\mathrm{i}}^{L+1}|\mathbf{G}_{\mathrm{ii}}^{L}\right)}{\left[\prod_{\ell=1}^{L} \mathrm{P}\left(\mathbf{G}_{\mathrm{ii}}^{\ell}|\mathbf{G}_{\mathrm{ii}}^{\ell-1}\right)\right] \mathrm{P}\left(\mathbf{F}_{\mathrm{i}}^{L+1}|\mathbf{G}_{\mathrm{ii}}^{L}\right)}\right]. \tag{38}$$

The first term is a summation across test/train datapoints, and the second term depends only on the inducing points, so as in Salimbeni & Deisenroth (2017) we can compute unbiased estimates of the expectation by taking only a minibatch of datapoints. We also never need to compute the density of the conditional prior in Eq. (35), we only need to be able to sample from it,

$$\mathrm{P}\left(\{\mathbf{G}_{\mathrm{ti}}^{\ell}, \mathbf{G}_{\mathrm{tt}}^{\ell}\}_{\ell=1}^{L}, \mathbf{F}_{\mathrm{t}}^{L+1} | \{\mathbf{G}_{\mathrm{ii}}^{\ell}\}_{\ell=1}^{L}, \mathbf{F}_{\mathrm{i}}^{L+1}, \mathbf{X}\right) = \mathrm{P}\left(\mathbf{F}_{\mathrm{t}}^{L+1}|\mathbf{F}_{\mathrm{i}}^{L+1}, \mathbf{G}_{L}\right) \prod_{\ell=1}^{L} \mathrm{P}\left(\mathbf{G}_{\mathrm{ti}}^{\ell}, \mathbf{G}_{\mathrm{tt}}^{\ell}|\mathbf{G}_{\mathrm{ii}}^{\ell}, \mathbf{G}_{\ell-1}\right). \tag{39}$$

The first distribution, $\mathrm{P}\left(\mathbf{F}_{\mathrm{t}}^{L+1}|\mathbf{F}_{\mathrm{i}}^{L+1}, \mathbf{G}_{L}\right)$, is a multivariate Gaussian, and can be evaluated using methods from the GP literature (Williams & Rasmussen, 2006; Salimbeni & Deisenroth, 2017). Specifically, we use the global inducing point scheme from Ober & Aitchison (2020). The second distribution is more difficult to sample from. To address this issue, we introduce sampled features $\hat{\mathbf{F}}_{\ell}$ (not to be confused with the features $\mathbf{F}_{\ell}$ in the corresponding DGP) such that

$$\hat{\mathbf{F}}_{\ell}\hat{\mathbf{F}}_{\ell}^{T} = \mathbf{G}_{\ell} \sim \mathcal{W}\left(\mathbf{\Sigma}, \nu\right), \tag{40}$$

with

$$\hat{\mathbf{F}}_{\ell} = \begin{pmatrix} \hat{\mathbf{F}}_{\mathrm{i}}^{\ell} \\ \hat{\mathbf{F}}_{\mathrm{t}}^{\ell} \end{pmatrix}, \qquad \mathbf{\Sigma} = \begin{pmatrix} \mathbf{\Sigma}_{\mathrm{ii}} & \mathbf{\Sigma}_{\mathrm{ti}}^{T} \\ \mathbf{\Sigma}_{\mathrm{ti}} & \mathbf{\Sigma}_{\mathrm{tt}} \end{pmatrix} = \tfrac{1}{\nu}\mathbf{K}_{\ell}\left(\mathbf{G}_{\ell-1}\right), \tag{41}$$

where $\hat{\mathbf{F}}_{\ell} \in \mathbb{R}^{(P_{\mathrm{i}}+P_{\mathrm{t}})\times\nu_{\ell}}$, $\hat{\mathbf{F}}_{\mathrm{i}} \in \mathbb{R}^{P_{\mathrm{i}}\times\nu_{\ell}}$ and $\hat{\mathbf{F}}_{\mathrm{t}} \in \mathbb{R}^{P_{\mathrm{t}}\times\nu_{\ell}}$. Our goal is to sample $\mathbf{G}_{\mathrm{it}}^{\ell}$ and $\mathbf{G}_{\mathrm{tt}}^{\ell}$ given $\mathbf{G}_{\mathrm{ii}}^{\ell}$. Our approach is to note that, $\hat{\mathbf{F}}_{\mathrm{t}}$ conditioned on $\hat{\mathbf{F}}_{\mathrm{i}}$ is given by a matrix normal, (Eaton et al., 2007, page 310),

$$\mathrm{P}\left(\hat{\mathbf{F}}_{\mathrm{t}}^{\ell}\middle|\hat{\mathbf{F}}_{\mathrm{i}}^{\ell}\right) = \mathcal{MN}\left(\mathbf{\Sigma}_{\mathrm{ti}}^{T}\mathbf{\Sigma}_{\mathrm{ii}}^{-1}\hat{\mathbf{F}}_{\mathrm{i}}, \mathbf{\Sigma}_{\mathrm{tt\cdot i}}, \mathbf{I}\right), \tag{42}$$

where

$$\mathbf{\Sigma}_{\mathrm{tt\cdot i}} = \mathbf{\Sigma}_{\mathrm{tt}} - \mathbf{\Sigma}_{\mathrm{it}}^{T}\mathbf{\Sigma}_{\mathrm{ii}}^{-1}\mathbf{\Sigma}_{\mathrm{it}}. \tag{43}$$

Note that we sample each test/train point one-at-a-time/independently, in which case, $P_{\mathrm{t}} = 1$ and $\mathbf{\Sigma}_{22\cdot 1}$ is scalar.

Then $\mathbf{G}_{\ell}$, which includes $\mathbf{G}_{\mathrm{it}}^{\ell}$ and $\mathbf{G}_{\mathrm{tt}}^{\ell}$ is given by,

$$\mathbf{G}_{\ell} = \begin{pmatrix} \mathbf{G}_{\mathrm{ii}}^{\ell} & \mathbf{G}_{\mathrm{it}}^{\ell} \\ \mathbf{G}_{\mathrm{ti}}^{\ell} & \mathbf{G}_{\mathrm{tt}}^{\ell} \end{pmatrix} = \begin{pmatrix} \hat{\mathbf{F}}_{\mathrm{i}}^{\ell}\left(\hat{\mathbf{F}}_{\mathrm{i}}^{\ell}\right)^{T} & \hat{\mathbf{F}}_{\mathrm{i}}^{\ell}\left(\hat{\mathbf{F}}_{\mathrm{t}}^{\ell}\right)^{T} \\ \hat{\mathbf{F}}_{\mathrm{t}}^{\ell}\left(\hat{\mathbf{F}}_{\mathrm{i}}^{\ell}\right)^{T} & \hat{\mathbf{F}}_{\mathrm{t}}^{\ell}\left(\hat{\mathbf{F}}_{\mathrm{t}}^{\ell}\right)^{T} \end{pmatrix} = \begin{pmatrix} \hat{\mathbf{F}}_{\mathrm{i}}^{\ell} \\ \hat{\mathbf{F}}_{\mathrm{t}}^{\ell} \end{pmatrix}\left(\left(\hat{\mathbf{F}}_{\mathrm{i}}^{\ell}\right)^{T} \quad \left(\hat{\mathbf{F}}_{\mathrm{t}}^{\ell}\right)^{T}\right) = \hat{\mathbf{F}}_{\ell}\hat{\mathbf{F}}_{\ell}^{T} \tag{44}$$

For $\hat{\mathbf{F}}_{\mathrm{i}}$, we can use any value as long as $\mathbf{G}_{\mathrm{ii}}^{\ell} = \hat{\mathbf{F}}_{\mathrm{i}}^{\ell}\left(\hat{\mathbf{F}}_{\mathrm{i}}^{\ell}\right)^{T}$, as the resulting distribution over $\mathbf{G}_{\ell}$ arising from Eq. (40) does not depend on the specific choice of $\hat{\mathbf{F}}_{\mathrm{i}}$ (App. F). Remembering that to sample $\mathbf{G}_{\mathrm{ii}}$, we explicitly sample its potentially low-rank Cholesky, $\mathbf{L}_{\ell}\mathbf{A}_{\ell}$, we can directly use

$$\hat{\mathbf{F}}_{\mathrm{i}}^{\ell} = \mathbf{L}_{\ell}\mathbf{A}_{\ell} \tag{45}$$

However, this only works if $\nu \leq P_{\mathrm{i}}$, in which case, $\mathbf{L}_{\ell}\mathbf{A}_{\ell} \in \mathbb{R}^{P_{\mathrm{i}}\times\nu}$. In the unusual case where we have fewer inducing points than degrees of freedom, $P_{\mathrm{i}} < \nu$, then $\mathbf{L}_{\ell}\mathbf{A}_{\ell} \in \mathbb{R}^{P_{\mathrm{i}}\times P_{\mathrm{i}}}$, so we need to pad to achieve the required size of $P_{\mathrm{i}} \times \nu_{\ell}$,

$$\hat{\mathbf{F}}_{\mathrm{i}}^{\ell} = \begin{pmatrix} \mathbf{L}_{\ell}\mathbf{A}_{\ell} & \mathbf{0} \end{pmatrix}. \tag{46}$$

Finally, note that we can optimise all the variational parameters using standard reparameterised variational inference (Kingma & Welling, 2013; Rezende et al., 2014).

A.5. Computational complexity

Recalling that ν_ℓ is the width of the ℓth layer, P_i is the number of inducing points, and P_t is the number of train or test points, the computational complexity of one DWP layer is given by $O(P_i^3 + P_t P_i^2)$. This is a decrease of a factor of $\nu_{\ell+1}$ over the complexity for standard DGP inference, such as doubly stochastic variational inference (Salimbeni & Deisenroth, 2017), which has complexity $O(\nu_{\ell+1}(P_i^3 + P_t P_i^2))$. The difference arises from the fact that in a DGP, $\nu_{\ell+1}$ Gaussian processes are sampled in each layer, whereas for a DWP we sample a single Gram matrix.

B. Deriving Jacobians for matrix transformations

Following the approach in (Mathai, 1997; Mathai & Haubold, 2008), we define the Jacobian of a function from x to y as the ratio of volume elements,

$$\text{jacobian} = \frac{dy_1 dy_2 \cdots dy_N}{dx_1 dx_2 \cdots dx_N}. \tag{47}$$

Critically, dx_i and dy_i are basis-vectors, *not* scalars. As we are multiplying vectors, not scalars, we need to be careful about our choice of multiplication operation. The correct choice in our context is an anti-symmetric exterior product, representing a directed area or volume element, such that,

$$dx_i dx_j = -dx_j dx_i. \tag{48}$$

As the product is antisymmetric, the product of a basis-vector with itself is zero,

$$dx_i dx_i = -dx_i dx_i = 0, \tag{49}$$

which makes sense because the product represents an area, and the area is zero if the two vectors are aligned. To confirm that this matches usual expressions for Jacobians, consider a 2×2 matrix-vector multiplication, $\mathbf{y} = \mathbf{A}\mathbf{x}$:

$$\begin{pmatrix} dy_1 \\ dy_2 \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} dx_1 \\ dx_2 \end{pmatrix} = \begin{pmatrix} A_{11}dx_1 + A_{12}dx_2 \\ A_{21}dx_1 + A_{22}dx_2 \end{pmatrix}. \tag{50}$$

Thus,

$$dy_1 dy_2 = (A_{11}dx_1 + A_{12}dx_2)(A_{21}dx_1 + A_{22}dx_2) \tag{51}$$

$$dy_1 dy_2 = A_{11}A_{21}dx_1^2 + A_{11}A_{22}dx_1 dx_2 + A_{12}A_{21}dx_2 dx_1 + A_{12}A_{22}dx_2^2 \tag{52}$$

As $dx_1^2 = dx_2^2 = 0$, and $dx_1 dx_2 = -dx_1 dx_2$, we have,

$$dy_1 dy_2 = (A_{11}A_{22} - A_{12}A_{21})\, dx_1 dx_2 \tag{53}$$

$$dy_1 dy_2 = |\mathbf{A}|\, dx_1 dx_2, \tag{54}$$

so that the Jacobian computed using the determinant definition is equivalent to the expression for the determinant obtained by working with volume elements.

C. Jacobian for the product of a lower triangular matrix with itself

In this section, we compute the Jacobian for the transformation from $\mathbf{\Lambda} = \mathbf{L}\mathbf{A}$ to $\mathbf{G} = \mathbf{\Lambda}\mathbf{\Lambda}^T$. We begin by noting that the top-left block of the product of a lower-triangular matrix with itself is a product of smaller lower-triangular matrices:

$$\begin{pmatrix} \mathbf{\Lambda}_{:N,:N} & \mathbf{0} \\ \mathbf{\Lambda}_{N:,:N} & \mathbf{\Lambda}_{:N,:N} \end{pmatrix} \begin{pmatrix} \mathbf{\Lambda}_{:N,:N}^T & \mathbf{\Lambda}_{N:,:N}^T \\ \mathbf{0} & \mathbf{\Lambda}_{:N,:N}^T \end{pmatrix} = \begin{pmatrix} \mathbf{\Lambda}_{:N,:N}\mathbf{\Lambda}_{:N,:N}^T & \cdots \\ \vdots & \ddots \end{pmatrix}. \tag{55}$$

As such, we can incrementally compute the Jacobian for this transformation by starting with the top-left 1×1 matrix,

$$G_{11} = \Lambda_{11}^2 \tag{56}$$

$$dG_{11} = 2\Lambda_{11} d\Lambda_{11}. \tag{57}$$

Next, we consider the top-left 2×2 matrix,

$$\begin{pmatrix} G_{11} & G_{12} \\ G_{21} & G_{22} \end{pmatrix} = \begin{pmatrix} \Lambda_{11} & 0 \\ \Lambda_{21} & \Lambda_{22} \end{pmatrix} \begin{pmatrix} \Lambda_{11} & \Lambda_{21} \\ 0 & \Lambda_{22} \end{pmatrix} = \begin{pmatrix} \Lambda_{11}^2 & \Lambda_{21}\Lambda_{11} \\ \Lambda_{21}\Lambda_{11} & \Lambda_{21}^2 + \Lambda_{22}^2 \end{pmatrix}. \tag{58}$$

Thus

$$dG_{21} = \Lambda_{21} d\Lambda_{11} + \Lambda_{11} d\Lambda_{21} \tag{59}$$
$$dG_{22} = 2\Lambda_{22} d\Lambda_{22} + 2\Lambda_{21} d\Lambda_{21}. \tag{60}$$

Combining dG_{11} and dG_{21} gives

$$dG_{11} dG_{21} = (2\Lambda_{11} d\Lambda_{11}) (\Lambda_{21} d\Lambda_{11} + \Lambda_{11} d\Lambda_{21}) \tag{61}$$
$$dG_{11} dG_{21} = 2\Lambda_{11}^2 (d\Lambda_{11} d\Lambda_{21}), \tag{62}$$

and then combining $dG_{11}dG_{21}$ and dG_{22} gives

$$dG_{11} dG_{21} dG_{22} = \left(2\Lambda_{11}^2 (d\Lambda_{11} d\Lambda_{21})\right) (2\Lambda_{22} d\Lambda_{22} + 2\Lambda_{21} d\Lambda_{21}) \tag{63}$$
$$dG_{11} dG_{21} dG_{22} = 4\Lambda_{11}^2 \Lambda_{22} (d\Lambda_{11} d\Lambda_{21} d\Lambda_{22}). \tag{64}$$

Thus, we can prove by induction that the volume element for the top-left $p \times p$ block of $\mathbf{G}$, and in addition the first $K < p+1$ off-diagonal elements of the $p+1$th row is

$$\underbrace{\left(\prod_{i=1}^{p}\prod_{k=1}^{i} dG_{ik}\right)}_{\text{vol. elem. for } \mathbf{G}_{:p,:p}} \underbrace{\left(\prod_{k=1}^{K} dG_{p+1,k}\right)}_{\text{vol. elem. for } \mathbf{G}_{p+1,:K}} = 2^p \left(\prod_{i=1}^{p}\prod_{k=1}^{i} \Lambda_{kk} d\Lambda_{ik}\right) \left(\prod_{k=1}^{K} d\Lambda_{kk} \Lambda_{p+1,k}\right). \tag{65}$$

The proof consists of three parts: the base case, adding an off-diagonal element and adding an on-diagonal element. For the base-case, note that the expression is correct for $p = 1$ and $K = 0$ (Eq. 57). Next, we add an off-diagonal element, $G_{p+1,K+1}$, where $K + 1 < p + 1$. We begin by computing $dG_{p+1,K+1}$. Note that the sum only goes to $K + 1$, because $\Lambda_{K+1,j} = 0$ for $j > (K + 1)$:

$$G_{p+1,K+1} = \sum_{j=1}^{K+1} \Lambda_{p+1,j} \Lambda_{K+1,j}, \tag{66}$$
$$dG_{p+1,K+1} = \sum_{j=1}^{K+1} \left(\Lambda_{K+1,j} d\Lambda_{p+1,j} + \Lambda_{p+1,j} d\Lambda_{K+1,j}\right). \tag{67}$$

Remembering that $d\Lambda_{ij}^2 = 0$, the only term that does not cancel when we multiply by the volume element for the previous terms is that for $d\Lambda_{p+1,K+1}$:

$$\underbrace{\left(\prod_{i=1}^{p}\prod_{k=1}^{i} dG_{ik}\right)}_{\text{vol. elem. for } \mathbf{G}_{:p,:p}} \underbrace{\left(\prod_{k=1}^{K+1} dG_{p+1,k}\right)}_{\text{vol. elem. for } \mathbf{G}_{p+1,:K+1}} = \left(\prod_{i=1}^{p}\prod_{k=1}^{i} dG_{ik}\right) \left(\prod_{k=1}^{K} dG_{p+1,k}\right) dG_{p+1,K+1} \tag{68}$$
$$= 2^p \left(\prod_{i=1}^{p}\prod_{k=1}^{i} \Lambda_{kk} d\Lambda_{ik}\right) \left(\prod_{k=1}^{K} d\Lambda_{kk} \Lambda_{p+1,k}\right) dG_{p+1,K+1} \tag{69}$$
$$= 2^p \left(\prod_{i=1}^{p}\prod_{k=1}^{i} \Lambda_{kk} d\Lambda_{ik}\right) \left(\prod_{k=1}^{K} d\Lambda_{kk} \Lambda_{p+1,k}\right) (\Lambda_{K+1,K+1} d\Lambda_{p+1,K+1}) \tag{70}$$
$$= 2^p \left(\prod_{i=1}^{p}\prod_{k=1}^{i} \Lambda_{kk} d\Lambda_{ik}\right) \left(\prod_{k=1}^{K+1} d\Lambda_{kk} \Lambda_{p+1,k}\right). \tag{71}$$

So the expression is consistent when adding an on-diagonal element. Finally, the volume element for $G_{p+1,p+1}$ is,

$$G_{p+1,p+1} = \sum_{j=1}^{p+1} \Lambda_{p+1,j}^2 dG_{p+1,K+1} \qquad = \sum_{j=1}^{K+1} \Lambda_{p+1,j} d\Lambda_{p+1,j}. \tag{72}$$

Remembering that $d\Lambda_{ij}^2 = 0$, the only term that does not cancel when we multiply by the volume element for the previous terms is that for $d\Lambda_{p+1,p+1}$,

$$\underbrace{\left(\prod_{i=1}^{p+1}\prod_{k=1}^{i} dG_{ik}\right)}_{\text{vol. elem. for } \mathbf{G}_{:p+1,:p+1}} = \underbrace{\left(\prod_{i=1}^{p}\prod_{k=1}^{i} dG_{ik}\right)}_{\text{vol. elem. for } \mathbf{G}_{:p,:p}} \underbrace{\left(\prod_{k=1}^{p+1} dG_{p+1,k}\right)}_{\text{vol. elem. for } \mathbf{G}_{p+1,:p+1}} \tag{73}$$

$$= \left(\prod_{i=1}^{p}\prod_{k=1}^{i} dG_{ik}\right)\left(\prod_{k=1}^{p} dG_{p+1,k}\right) dG_{p+1,p+1} \tag{74}$$

$$= 2^p \left(\prod_{i=1}^{p}\prod_{k=1}^{i} \Lambda_{kk} d\Lambda_{ik}\right)\left(\prod_{k=1}^{p} d\Lambda_{kk}\Lambda_{p+1,k}\right) dG_{p+1,p+1} \tag{75}$$

$$= 2^p \left(\prod_{i=1}^{p}\prod_{k=1}^{i} \Lambda_{kk} d\Lambda_{ik}\right)\left(\prod_{k=1}^{p} d\Lambda_{kk}\Lambda_{p+1,k}\right) (2\Lambda_{p+1,p+1} d\Lambda_{p+1,p+1}) \tag{76}$$

$$= 2^{p+1} \left(\prod_{i=1}^{p}\prod_{k=1}^{i} \Lambda_{kk} d\Lambda_{ik}\right)\left(\prod_{k=1}^{p+1} d\Lambda_{kk}\Lambda_{p+1,k}\right) \tag{77}$$

$$= 2^{p+1} \left(\prod_{i=1}^{p+1}\prod_{k=1}^{i} \Lambda_{kk} d\Lambda_{ik}\right). \tag{78}$$

Thus, the final result is:

$$\left(\prod_{i=1}^{p}\prod_{k=1}^{i} dG_{ik}\right) = \left(2^p \prod_{i=1}^{\min(p,\nu)} \Lambda_{ii}^{p+1-i}\right)\left(\prod_{i=1}^{p}\prod_{k=1}^{i} d\Lambda_{ik}\right). \tag{79}$$

$$d\mathbf{G} = d\mathbf{\Lambda} \prod_{i=1}^{p} 2\Lambda_{ii}^{p+1-i}. \tag{80}$$

C.1. Singular matrices

The above derivation can be extended to the singular case, where $\mathbf{\Lambda}$ has a form mirroring that of $\mathbf{A}$ in Eq. (21):

$$\mathbf{\Lambda} = \begin{pmatrix} \Lambda_{11} & \dots & 0 \\ \vdots & \ddots & \vdots \\ \Lambda_{\nu 1} & \dots & \Lambda_{\nu\nu} \\ \vdots & \vdots & \vdots \\ \Lambda_{P1} & \dots & \Lambda_{P\nu} \end{pmatrix}. \tag{81}$$

To form a valid Jacobian, we need the same number of inputs as outputs. We therefore consider differences in only the corresponding part of $\mathbf{G}$ (i.e. $G_{i,j\leq\min(i,\nu)}$). The recursive expression is

$$\underbrace{\left(\prod_{i=1}^{p}\prod_{k=1}^{\min(i,\nu)} dG_{ik}\right)}_{\text{vol. elem. for } \mathbf{G}_{:p,:p}} \underbrace{\left(\prod_{k=1}^{K} dG_{p+1,k}\right)}_{\text{vol. elem. for } \mathbf{G}_{p+1,:K}} = 2^{\min(p,\nu)} \left(\prod_{i=1}^{p}\prod_{k=1}^{\min(i,\nu)} \Lambda_{kk} d\Lambda_{ik}\right)\left(\prod_{k=1}^{K} d\Lambda_{kk}\Lambda_{p+1,k}\right), \tag{82}$$

where $K < \min(p, \nu)$. For $p \leq \nu$, the recursion is exactly as in the full-rank case. For $p > \nu$, the key difference is that there are no longer any on-diagonal elements. As such, for $K = \nu$ we have

$$\left(\prod_{i=1}^{p+1}\prod_{k=1}^{\min(i,\nu)} dG_{ik}\right) = \underbrace{\left(\prod_{i=1}^{p}\prod_{k=1}^{\min(i,\nu)} dG_{ik}\right)}_{\text{vol. elem. for } \mathbf{G}_{:p,:p}} \underbrace{\left(\prod_{k=1}^{\nu} dG_{p+1,k}\right)}_{\text{vol. elem. for } \mathbf{G}_{p+1,:\nu}} \tag{83}$$

$$= 2^{\min(p,\nu)} \left(\prod_{i=1}^{p}\prod_{k=1}^{\min(i,\nu)} \Lambda_{kk} d\Lambda_{ik}\right) \left(\prod_{k=1}^{K} d\Lambda_{kk}\Lambda_{p+1,k}\right) \tag{84}$$

$$= 2^{\min(p,\nu)} \left(\prod_{i=1}^{p+1}\prod_{k=1}^{\min(i,\nu)} \Lambda_{kk} d\Lambda_{ik}\right). \tag{85}$$

The final expression, allowing for the possibility of singular and non-singular matrices, is thus

$$\left(\prod_{i=1}^{p}\prod_{k=1}^{\min(i,\nu)} dG_{ik}\right) = \left(\prod_{i=1}^{\min(p,\nu)} 2\Lambda_{ii}^{p+1-i}\right) \left(\prod_{i=1}^{p}\prod_{k=1}^{\min(i,\nu)} d\Lambda_{ik}\right) \tag{86}$$

$$d\mathbf{G} = d\mathbf{\Lambda} \prod_{i=1}^{\min(p,\nu)} 2\Lambda_{ii}^{p+1-i}. \tag{87}$$

D. Jacobian for product of two different lower triangular matrices

In this section, we compute the Jacobian for the transformation from $\mathbf{A}$ to $\mathbf{\Lambda} = \mathbf{LA}$. We begin by noting that $\mathbf{\Lambda}$ (Eq. 81) is a potentially rectangular lower-triangular matrix, with the same structure as $\mathbf{A}$. Writing this out,

$$\begin{pmatrix} \Lambda_{11} & 0 & 0 \\ \Lambda_{21} & \Lambda_{22} & 0 \\ \Lambda_{31} & \Lambda_{32} & \Lambda_{33} \\ \Lambda_{41} & \Lambda_{42} & \Lambda_{43} \\ \Lambda_{51} & \Lambda_{52} & \Lambda_{53} \end{pmatrix} = \begin{pmatrix} L_{11} & 0 & 0 & 0 & 0 \\ L_{21} & L_{22} & 0 & 0 & 0 \\ L_{31} & L_{32} & L_{33} & 0 & 0 \\ L_{41} & L_{42} & L_{43} & L_{44} & 0 \\ L_{51} & L_{52} & L_{53} & L_{54} & L_{55} \end{pmatrix} \begin{pmatrix} A_{11} & 0 & 0 \\ A_{21} & A_{22} & 0 \\ A_{31} & A_{32} & A_{33} \\ A_{41} & A_{42} & A_{43} \\ A_{51} & A_{52} & A_{53} \end{pmatrix}. \tag{88}$$

For the first column,

$$\begin{pmatrix} \Lambda_{11} \\ \Lambda_{21} \\ \Lambda_{31} \\ \Lambda_{41} \\ \Lambda_{51} \end{pmatrix} = \begin{pmatrix} L_{11} & 0 & 0 & 0 & 0 \\ L_{21} & L_{22} & 0 & 0 & 0 \\ L_{31} & L_{32} & L_{33} & 0 & 0 \\ L_{41} & L_{42} & L_{43} & L_{44} & 0 \\ L_{51} & L_{52} & L_{53} & L_{54} & L_{55} \end{pmatrix} \begin{pmatrix} A_{11} \\ A_{21} \\ A_{31} \\ A_{41} \\ A_{51} \end{pmatrix}, \tag{89}$$

i.e.

$$\mathbf{\Lambda}_{:,1} = \mathbf{LA}_{:,1}. \tag{90}$$

For the second column,

$$\begin{pmatrix} 0 \\ \Lambda_{22} \\ \Lambda_{32} \\ \Lambda_{42} \\ \Lambda_{52} \end{pmatrix} = \begin{pmatrix} L_{11} & 0 & 0 & 0 & 0 \\ L_{21} & L_{22} & 0 & 0 & 0 \\ L_{31} & L_{32} & L_{33} & 0 & 0 \\ L_{41} & L_{42} & L_{43} & L_{44} & 0 \\ L_{51} & L_{52} & L_{53} & L_{54} & L_{55} \end{pmatrix} \begin{pmatrix} 0 \\ A_{22} \\ A_{32} \\ A_{42} \\ A_{52} \end{pmatrix}. \tag{91}$$

We can eliminate the first row and column of $\mathbf{C}$,

$$\begin{pmatrix} \Lambda_{22} \\ \Lambda_{32} \\ \Lambda_{42} \\ \Lambda_{52} \end{pmatrix} = \begin{pmatrix} L_{22} & 0 & 0 & 0 \\ L_{32} & L_{33} & 0 & 0 \\ L_{42} & L_{43} & L_{44} & 0 \\ L_{52} & L_{53} & L_{54} & L_{55} \end{pmatrix} \begin{pmatrix} A_{22} \\ A_{32} \\ A_{42} \\ A_{52} \end{pmatrix}, \tag{92}$$

i.e.

$$\mathbf{\Lambda}_{2:,2} = \mathbf{L}_{2:,2:}\mathbf{A}_{2:,2}. \tag{93}$$

For the third column,

$$\begin{pmatrix} 0 \\ 0 \\ \Lambda_{33} \\ \Lambda_{43} \\ \Lambda_{53} \end{pmatrix} = \begin{pmatrix} L_{11} & 0 & 0 & 0 & 0 \\ L_{21} & L_{22} & 0 & 0 & 0 \\ L_{31} & L_{32} & L_{33} & 0 & 0 \\ L_{41} & L_{42} & L_{43} & L_{44} & 0 \\ L_{51} & L_{52} & L_{53} & L_{54} & L_{55} \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ A_{33} \\ A_{43} \\ A_{53} \end{pmatrix}, \tag{94}$$

so we can eliminate the first two rows and columns of $\mathbf{C}$:

$$\begin{pmatrix} \Lambda_{33} \\ \Lambda_{43} \\ \Lambda_{53} \end{pmatrix} = \begin{pmatrix} L_{33} & 0 & 0 \\ L_{43} & L_{44} & 0 \\ L_{53} & L_{54} & L_{55} \end{pmatrix} \begin{pmatrix} A_{33} \\ A_{43} \\ A_{53} \end{pmatrix}, \tag{95}$$

i.e.

$$\mathbf{\Lambda}_{3:,3} = \mathbf{L}_{3:,3:}\mathbf{A}_{3:,3}. \tag{96}$$

As such, the full computation $\mathbf{\Lambda} = \mathbf{LA}$ can be written as a matrix-vector multiplication by rearranging the columns of $\mathbf{\Lambda}$ and $\mathbf{A}$ into a single vector:

$$\begin{pmatrix} \mathbf{\Lambda}_{1:,1} \\ \mathbf{\Lambda}_{2:,2} \\ \vdots \\ \mathbf{\Lambda}_{\nu:,\nu} \end{pmatrix} = \begin{pmatrix} \mathbf{L}_{1:,1:} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{L}_{2:,2:} & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{L}_{\nu:,\nu:} \end{pmatrix} \begin{pmatrix} \mathbf{A}_{1:,1} \\ \mathbf{A}_{2:,2} \\ \vdots \\ \mathbf{A}_{\nu:,\nu} \end{pmatrix}. \tag{97}$$

The Jacobian is given by the determinant of the square matrix, and as the matrix is lower-triangular, the determinant can be written in terms of the diagonal elements of $\mathbf{L}$,

$$\left(\prod_{i=1}^{p} \prod_{k=1}^{\min(i,\nu)} d\Lambda_{ik} \right) = \left(\prod_{i=1}^{p} L_{ii}^{\min(i,\nu)} \right) \left(\prod_{i=1}^{p} \prod_{k=1}^{\min(i,\nu)} dA_{ik} \right). \tag{98}$$

E. Proving the singular Bartlett (above) corresponds to the Wishart

We need to change variables to A_{jj} rather than A_{jj}^2 [1]:

$$\begin{aligned} \mathrm{P}\left(A_{jj}\right) &= \mathrm{P}\left(A_{jj}^2\right) \left| \frac{\partial A_{jj}^2}{\partial A_{jj}} \right|, && (99) \\ &= \text{Gamma}\left(A_{jj}^2; \tfrac{\nu-j+1}{2}, \tfrac{1}{2}\right) 2A_{jj}, && (100) \\ &= \frac{\left(A_{jj}^2\right)^{(\nu-j+1)/2-1} e^{-A_{jj}^2/2}}{2^{(\nu-j+1)/2}\Gamma\left(\frac{\nu-j+1}{2}\right)} 2A_{jj}, && (101) \\ &= \frac{A_{jj}^{\nu-j} e^{-A_{jj}^2/2}}{2^{(\nu-j-1)/2}\Gamma\left(\frac{\nu-j+1}{2}\right)}. && (102) \end{aligned}$$

Thus, the probability density for $\mathbf{A}$ under the Bartlett sampling operation is

$$\mathrm{P}\left(\mathbf{A}\right) = \underbrace{\prod_{j=1}^{\tilde{\nu}} \frac{A_{jj}^{\nu-j} e^{-T_{jj}^2/2}}{2^{\frac{\nu-j-1}{2}}\Gamma\left(\frac{\nu-j+1}{2}\right)}}_{\text{on-diagonals}} \underbrace{\prod_{i=j+1}^{p} \frac{1}{\sqrt{2\pi}} e^{-A_{ij}^2/2}}_{\text{off-diagonals}}, \tag{103}$$

[1] djalil.chafai.net/blog/2015/10/20/bartlett-decomposition-and-other-factorizations/

where $\tilde{\nu} = \min(\nu, p)$. To convert this to a distribution on $\mathbf{G}$, we need the volume element for the transformation from $\mathbf{A}$ to $\mathbf{Z} = \mathbf{A}\mathbf{A}^T$, which is given in Appendix C (Eq. (80)):

$$\mathrm{d}\mathbf{Z} = \mathrm{d}\mathbf{A} \prod_{j=1}^{\tilde{\nu}} 2A_{jj}^{p-j+1}. \tag{104}$$

Thus

$$\mathrm{P}\left(\mathbf{Z}\right) = \mathrm{P}\left(\mathbf{A}\right) \left(\prod_{j=1}^{\tilde{\nu}} \tfrac{1}{2} A_{jj}^{-(p-j+1)} \right) \tag{105}$$

$$= \prod_{j=1}^{\tilde{\nu}} \frac{A_{jj}^{\nu-p-1} e^{-T_{jj}^2/2}}{2^{\frac{\nu-j+1}{2}} \Gamma\left(\frac{\nu-j+1}{2}\right)} \prod_{i=j+1}^{p} \frac{1}{\sqrt{2\pi}} e^{-A_{ij}^2/2}. \tag{106}$$

Now, we break this expression down into separate components and perform standard algebraic manipulations. First, we manipulate a product over the diagonal elements of $\mathbf{A}$ to obtain the determinant of $\mathbf{Z}$:

$$\prod_{j=1}^{\tilde{\nu}} A_{jj}^{\nu-p-1} = \left(\prod_{j=1}^{\tilde{\nu}} A_{jj} \right)^{\nu-p-1} = \left| \mathbf{A}_{:\tilde{\nu},:} \mathbf{A}_{:\tilde{\nu},:}^T \right|^{(\nu-p-1)/2} = \left| \mathbf{Z}_{:\tilde{\nu},:\tilde{\nu}} \right|^{(\nu-p-1)/2}. \tag{107}$$

Next, we manipulate the exponential terms to form an exponentiated trace. We start by combining on- and off-diagonal terms, and noting that $A_{ij} = 0$ for $i < j$, we can extend the sum

$$\prod_{j=1}^{\tilde{\nu}} e^{-A_{jj}^2/2} \prod_{i=j+1}^{p} e^{-A_{ij}^2/2} = \prod_{j=1}^{\tilde{\nu}} \prod_{i=j}^{p} e^{-A_{ij}^2/2} = \prod_{j=1}^{\tilde{\nu}} \prod_{i=1}^{p} e^{-A_{ij}^2/2}. \tag{108}$$

Then we take the product inside the exponential and note that as $\mathbf{Z} = \mathbf{A}\mathbf{A}^T$, we can write the sum as a trace of $\mathbf{Z}$,

$$= e^{\sum_{j=1}^{\tilde{\nu}} \sum_{i=1}^{p} -A_{ij}^2/2} = e^{-\operatorname{Tr}(\mathbf{Z})/2}. \tag{109}$$

Next, we consider powers of 2. We begin by computing the number of $1/\sqrt{2}$ terms, arising from the off-diagonal elements,

$$\prod_{j=1}^{\tilde{\nu}} \prod_{i=j+1}^{p} \tfrac{1}{\sqrt{2}} = \left(\tfrac{1}{\sqrt{2}}\right)^{\nu(p-\tilde{\nu})+\tilde{\nu}(\tilde{\nu}-1)/2}. \tag{110}$$

Note that the $\tilde{\nu}(\tilde{\nu}-1)$ term corresponds to the off-diagonal terms in the square block $\mathbf{A}_{:\tilde{\nu},:}$, and the $\nu(p-\tilde{\nu})$ term corresponds to the terms from $\mathbf{A}_{\tilde{\nu}:,:}$. Next we consider the on-diagonal terms,

$$\prod_{j=1}^{\tilde{\nu}} \frac{1}{2^{(\nu-j+1)/2}} = \left(\tfrac{1}{\sqrt{2}}\right)^{\tilde{\nu}(\nu+1)} \prod_{j=1}^{\tilde{\nu}} \left(\tfrac{1}{\sqrt{2}}\right)^{-j} = \left(\tfrac{1}{\sqrt{2}}\right)^{\tilde{\nu}(\nu+1)-\tilde{\nu}(\tilde{\nu}+1)/2}. \tag{111}$$

Combining the on and off-diagonal terms,

$$\prod_{j=1}^{\tilde{\nu}} \frac{1}{2^{(\nu-j+1)/2}} \prod_{i=j+1}^{p} \tfrac{1}{\sqrt{2}} = \left(\tfrac{1}{\sqrt{2}}\right)^{\nu(p-\tilde{\nu})+\tilde{\nu}(\tilde{\nu}-1)/2+\tilde{\nu}(\nu+1)-\tilde{\nu}(\tilde{\nu}+1)/2} \tag{112}$$

$$= \left(\tfrac{1}{\sqrt{2}}\right)^{(\nu p-\nu\tilde{\nu})+(\tilde{\nu}^2/2-\tilde{\nu}/2)+(\tilde{\nu}\nu+\tilde{\nu})+(-\tilde{\nu}^2/2-\tilde{\nu}/2)} \tag{113}$$

$$= \left(\tfrac{1}{\sqrt{2}}\right)^{\nu p}. \tag{114}$$

Finally, using the definition of the multivariate Gamma function,

$$\prod_{j=1}^{\tilde{\nu}} \Gamma\left(\tfrac{\nu-j+1}{2}\right) \prod_{i=j+1}^{p} \sqrt{\pi} = \pi^{\nu(p-\tilde{\nu})/2} \underbrace{\pi^{\tilde{\nu}(\tilde{\nu}-1)/4} \prod_{j=1}^{\tilde{\nu}} \Gamma\left(\tfrac{\nu-j+1}{2}\right)}_{=\Gamma_{\tilde{\nu}}\left(\frac{\nu}{2}\right)} \tag{115}$$

$$= \pi^{\nu(p-\tilde{\nu})/2} \Gamma_{\tilde{\nu}}\left(\tfrac{\nu}{2}\right). \tag{116}$$

We thereby re-obtain the probability density for the standard Wishart distribution,

$$\mathrm{P}\left(\mathbf{Z}\right) = \frac{\pi^{\nu(\tilde{\nu}-p)/2}}{2^{\nu p/2}\Gamma_{\tilde{\nu}}\left(\frac{\nu}{2}\right)} \left|\mathbf{Z}_{:\tilde{\nu},:\tilde{\nu}}\right|^{(\nu-p-1)/2} e^{-\operatorname{Tr}(\boldsymbol{\Sigma})/2}. \tag{117}$$

For $\tilde{\nu} = \nu$, this matches Eq. 3.2 in Srivastava et al. (2003), and for $\tilde{\nu} = p$ it matches the standard full-rank Wishart probability density function.

F. Choice of $\mathbf{F}_{\mathrm{i}}$

Here, we establish that the distribution over $\mathbf{F}_{\mathrm{t}}\mathbf{F}_{\mathrm{i}}^T$ and $\mathbf{F}_{\mathrm{t}}\mathbf{F}_{\mathrm{t}}^T$ does not depend on the choice of $\mathbf{F}_{\mathrm{i}}$. Due to the definition of $\mathbf{F}_{\mathrm{t}}$ (Eq. 42) we can write,

$$\mathbf{F}_t = \boldsymbol{\Sigma}_{\mathrm{ti}}^T \boldsymbol{\Sigma}_{\mathrm{ii}}^{-1} \mathbf{F}_{\mathrm{i}} + \boldsymbol{\Sigma}_{\mathrm{tt\cdot i}}^{1/2} \boldsymbol{\Xi}. \tag{118}$$

where $\boldsymbol{\Xi}$ is a matrix with IID standard Gaussian elements. Thus,

$$\mathbf{F}_{\mathrm{t}}\mathbf{F}_{\mathrm{i}}^T = \boldsymbol{\Sigma}_{\mathrm{ti}}^T \boldsymbol{\Sigma}_{\mathrm{ii}}^{-1} \mathbf{F}_{\mathrm{i}}\mathbf{F}_{\mathrm{i}}^T + \boldsymbol{\Sigma}_{\mathrm{tt\cdot i}}^{1/2} \boldsymbol{\Xi}\mathbf{F}_{\mathrm{i}}^T \tag{119}$$

$$\mathbf{F}_{\mathrm{t}}\mathbf{F}_{\mathrm{i}}^T \sim \mathcal{MN}\left(\boldsymbol{\Sigma}_{\mathrm{ti}}^T \boldsymbol{\Sigma}_{\mathrm{ii}}^{-1} \mathbf{G}_{\mathrm{ii}}, \boldsymbol{\Sigma}_{\mathrm{tt\cdot i}}, \mathbf{G}_{\mathrm{ii}}\right). \tag{120}$$

We can do the same for $\mathbf{F}_{\mathrm{t}}\mathbf{F}_{\mathrm{t}}^T$:

$$\begin{aligned}\mathbf{F}_{\mathrm{t}}\mathbf{F}_{\mathrm{t}}^T =& \boldsymbol{\Sigma}_{\mathrm{ti}}^T \boldsymbol{\Sigma}_{\mathrm{ii}}^{-1} \mathbf{F}_{\mathrm{i}}\mathbf{F}_{\mathrm{i}}^T \boldsymbol{\Sigma}_{\mathrm{ii}}^{-1} \boldsymbol{\Sigma}_{\mathrm{ti}} \\ &+ \boldsymbol{\Sigma}_{\mathrm{ti}}^T \boldsymbol{\Sigma}_{\mathrm{ii}}^{-1} \mathbf{F}_{\mathrm{i}} \boldsymbol{\Xi} \boldsymbol{\Sigma}_{\mathrm{tt\cdot i}}^{1/2} + \boldsymbol{\Sigma}_{\mathrm{tt\cdot i}}^{1/2} \boldsymbol{\Xi}\mathbf{F}_{\mathrm{i}}^T \boldsymbol{\Sigma}_{\mathrm{ii}}^{-1} \boldsymbol{\Sigma}_{\mathrm{ti}} \\ &+ \boldsymbol{\Sigma}_{\mathrm{tt\cdot i}}^{1/2} \boldsymbol{\Xi}\boldsymbol{\Xi} \boldsymbol{\Sigma}_{\mathrm{tt\cdot i}}^{1/2}.\end{aligned}$$

The first term is independent of the choice of of $\mathbf{F}_{\mathrm{i}}$ because $\mathbf{G}_{\mathrm{ii}} = \mathbf{F}_{\mathrm{i}}\mathbf{F}_{\mathrm{i}}^T$. The term on the last line does not depend on $\mathbf{F}_{\mathrm{i}}$ at all. Finally, the two terms in the middle are Gaussian with covariance that depends on $\mathbf{G}_{\mathrm{ii}}$ but not the specific choice of $\mathbf{F}_{\mathrm{i}}$:

$$\boldsymbol{\Sigma}_{\mathrm{ti}}^T \boldsymbol{\Sigma}_{\mathrm{ii}}^{-1} \mathbf{F}_{\mathrm{i}} \boldsymbol{\Xi} \boldsymbol{\Sigma}_{\mathrm{tt\cdot i}}^{1/2} \sim \mathcal{MN}\left(\mathbf{0}, \boldsymbol{\Sigma}_{\mathrm{ti}}^T \boldsymbol{\Sigma}_{\mathrm{ii}}^{-1} \mathbf{G}_{\mathrm{i}} \boldsymbol{\Sigma}_{\mathrm{ii}}^{-1} \boldsymbol{\Sigma}_{\mathrm{ti}}, \boldsymbol{\Sigma}_{\mathrm{tt\cdot i}}\right), \tag{121}$$

$$\boldsymbol{\Sigma}_{\mathrm{tt\cdot i}}^{1/2} \boldsymbol{\Xi}\mathbf{F}_{\mathrm{i}}^T \boldsymbol{\Sigma}_{\mathrm{ii}}^{-1} \boldsymbol{\Sigma}_{\mathrm{ti}} \sim \mathcal{MN}\left(\mathbf{0}, \boldsymbol{\Sigma}_{\mathrm{tt\cdot i}}, \boldsymbol{\Sigma}_{\mathrm{ti}}^T \boldsymbol{\Sigma}_{\mathrm{ii}}^{-1} \mathbf{G}_{\mathrm{i}} \boldsymbol{\Sigma}_{\mathrm{ii}}^{-1} \boldsymbol{\Sigma}_{\mathrm{ti}}\right). \tag{122}$$

Thus, the additional components of $\mathbf{G}$, $\mathbf{G}_{\mathrm{ti}} = \mathbf{F}_{\mathrm{t}}\mathbf{F}_{\mathrm{i}}$ and $\mathbf{G}_{\mathrm{tt}} = \mathbf{F}_{\mathrm{t}}\mathbf{F}_{\mathrm{t}}$ depend on $\mathbf{G}_{\mathrm{ii}}$ but not on the specific choice of $\mathbf{F}_{\mathrm{i}}$. Thus, any $\mathbf{F}_{\mathrm{i}}$ can be used as long as $\mathbf{G}_{\mathrm{ii}} = \mathbf{F}_{\mathrm{i}}\mathbf{F}_{\mathrm{i}}^T$.

G. Experimental details

Datasets All experiments were performed using the UCI splits from Gal & Ghahramani (2015), available at `https://github.com/yaringal/DropoutUncertaintyExps/tree/master/UCI_Datasets`. For each dataset there are twenty splits, with the exception of protein, which only has five. We report mean plus or minus one standard error over the splits.

Model details As standard, we set ν (the 'width' of each layer) to be equal to the dimensionality of the input space. We use the squared exponential kernel, with automatic relevance determination (ARD) in the first layer, but without for the intermediate layers as ARD relies on explicit features existing. However, we found in practice that using ARD for

intermediate layers in a DGP did not hugely affect the results, as each output GP in a layer shares the same prior and hence output prior variance. For the final, GP, layer of the DWP model we use a global inducing approximate posterior (Ober & Aitchison, 2020), as done in the DGP. We leave the particular implementation details for the code provided with the paper, but we note that we use the 'sticking the landing' gradient estimator (Roeder et al., 2017) for the $\{\boldsymbol{\alpha}_\ell, \boldsymbol{\beta}_\ell, \boldsymbol{\mu}_\ell, \boldsymbol{\sigma}_\ell\}_{\ell=1}^L$ approximate posterior parameters of the DWP (using it for the other parameters, as well as for the DGP parameters, is difficult as the parameters of one layer will affect the KL estimate of the following layers for global inducing posteriors).

Training details We train all models using the same training scheme. We use 20,000 gradient steps to train each model, using the Adam optimizer (Kingma & Ba, 2014) with an initial learning rate of 1e-2. We anneal the KL using a factor increasing linearly from 0 to 1 over the first 1,000 gradient steps, and step the learning rate down to 1e-3 after 10,000 gradient steps. We use 10 samples from the approximate posterior for training, and 100 for testing. Experiments were performed using an internal cluster of machines with NVIDIA GeForce 2080 Ti GPUs, although we used CPU (Intel Core i9-10900X) for the smaller datasets (boston, concrete, energy, wine, yacht). For easy parallelisation across machines and sessions, we used the Jug python package (Coelho, 2017).

H. Tables

Table 2. ELBOs per datapoint. We report mean plus or minus one standard error over the splits.

{dataset} - {depth}	DWP	DGP
boston - 2	**-0.33 ± 0.00**	-0.37 ± 0.00
3	**-0.34 ± 0.01**	-0.41 ± 0.00
4	**-0.36 ± 0.01**	-0.44 ± 0.01
5	**-0.38 ± 0.01**	-0.46 ± 0.01
concrete - 2	**-0.42 ± 0.00**	-0.45 ± 0.00
3	**-0.43 ± 0.00**	-0.48 ± 0.00
4	**-0.46 ± 0.00**	-0.52 ± 0.00
5	**-0.49 ± 0.00**	-0.55 ± 0.00
energy - 2	**1.46 ± 0.00**	1.43 ± 0.00
3	**1.44 ± 0.00**	1.41 ± 0.00
4	**1.42 ± 0.00**	1.39 ± 0.00
5	**1.41 ± 0.00**	1.37 ± 0.00
kin8nm - 2	-0.16 ± 0.00	-0.16 ± 0.00
3	**-0.15 ± 0.00**	-0.13 ± 0.00
4	**-0.14 ± 0.00**	-0.17 ± 0.01
5	**-0.14 ± 0.00**	-0.18 ± 0.00
naval - 2	3.79 ± 0.07	**3.93 ± 0.06**
3	3.75 ± 0.11	3.82 ± 0.07
4	3.72 ± 0.07	**3.95 ± 0.03**
5	3.62 ± 0.07	**3.74 ± 0.08**
power - 2	0.03 ± 0.00	0.03 ± 0.00
3	0.03 ± 0.00	0.03 ± 0.00
4	**0.03 ± 0.00**	0.02 ± 0.00
5	**0.03 ± 0.00**	0.01 ± 0.00
protein - 2	-1.07 ± 0.00	-1.07 ± 0.00
3	-1.04 ± 0.00	-1.04 ± 0.00
4	**-1.01 ± 0.00**	-1.02 ± 0.00
5	**-1.01 ± 0.00**	-1.02 ± 0.00
wine - 2	-1.18 ± 0.00	-1.18 ± 0.00
3	-1.18 ± 0.00	-1.18 ± 0.00
4	**-1.18 ± 0.00**	-1.19 ± 0.00
5	-1.19 ± 0.00	-1.19 ± 0.00
yacht - 2	**2.02 ± 0.01**	1.94 ± 0.01
3	**1.88 ± 0.02**	1.48 ± 0.01
4	**1.74 ± 0.02**	1.38 ± 0.01
5	**1.63 ± 0.01**	1.30 ± 0.02

Table 3. Average test log likelihoods. We report mean plus or minus one standard error over the splits.

{dataset} - {depth}	DWP	DGP
boston - 2	-2.40 ± 0.05	-2.42 ± 0.05
3	**-2.38 ± 0.04**	-2.46 ± 0.05
4	**-2.37 ± 0.04**	-2.47 ± 0.05
5	**-2.39 ± 0.04**	-2.48 ± 0.04
concrete - 2	-3.13 ± 0.02	-3.12 ± 0.02
3	-3.11 ± 0.02	-3.12 ± 0.02
4	**-3.12 ± 0.02**	-3.17 ± 0.01
5	**-3.13 ± 0.01**	-3.18 ± 0.01
energy - 2	-0.70 ± 0.03	-0.70 ± 0.03
3	-0.71 ± 0.03	-0.70 ± 0.03
4	-0.70 ± 0.03	-0.70 ± 0.03
5	-0.70 ± 0.03	-0.73 ± 0.03
kin8nm - 2	1.35 ± 0.00	1.35 ± 0.00
3	**1.37 ± 0.00**	1.40 ± 0.00
4	**1.40 ± 0.00**	1.38 ± 0.01
5	**1.40 ± 0.01**	1.38 ± 0.01
naval - 2	8.16 ± 0.10	**8.27 ± 0.05**
3	8.18 ± 0.08	8.13 ± 0.09
4	8.19 ± 0.06	**8.33 ± 0.03**
5	8.20 ± 0.04	8.15 ± 0.07
power - 2	-2.77 ± 0.01	-2.78 ± 0.01
3	-2.77 ± 0.01	-2.77 ± 0.01
4	-2.77 ± 0.01	-2.78 ± 0.01
5	**-2.77 ± 0.01**	-2.79 ± 0.01
protein - 2	**-2.81 ± 0.00**	-2.82 ± 0.00
3	**-2.78 ± 0.00**	-2.76 ± 0.00
4	**-2.73 ± 0.00**	-2.75 ± 0.01
5	-2.73 ± 0.00	-2.74 ± 0.01
wine - 2	-0.96 ± 0.01	-0.96 ± 0.01
3	-0.96 ± 0.01	-0.96 ± 0.01
4	-0.96 ± 0.01	-0.96 ± 0.01
5	-0.96 ± 0.01	-0.96 ± 0.01
yacht - 2	-0.03 ± 0.09	-0.05 ± 0.08
3	**-0.14 ± 0.07**	-0.68 ± 0.04
4	**-0.45 ± 0.11**	-0.72 ± 0.03
5	**-0.46 ± 0.12**	-0.77 ± 0.03