Deep Gaussian Markov Random Fields

Per Sidén[1] **Fredrik Lindsten**[1]

Abstract

Gaussian Markov random fields (GMRFs) are probabilistic graphical models widely used in spatial statistics and related fields to model dependencies over spatial structures. We establish a formal connection between GMRFs and convolutional neural networks (CNNs). Common GMRFs are special cases of a generative model where the inverse mapping from data to latent variables is given by a 1-layer linear CNN. This connection allows us to generalize GMRFs to multi-layer CNN architectures, effectively increasing the order of the corresponding GMRF in a way which has favorable computational scaling. We describe how well-established tools, such as autodiff and variational inference, can be used for simple and efficient inference and learning of the deep GMRF. We demonstrate the flexibility of the proposed model and show that it outperforms the state-of-the-art on a dataset of satellite temperatures, in terms of prediction and predictive uncertainty.

1. Introduction

Convolutional neural networks (CNNs) are the *de facto* standard model in computer vision when training on a large set of images. Images are lattice-based data with local dependencies and thus have clear connections with spatial statistics. However, many spatial problems lack the abundance of data common to computer vision applications, and often we need to build a model based on a single "image", i.e., data recorded over some spatial field. Models such as deep image prior (Ulyanov et al., 2018) have shown that CNN architectures can encode useful spatial priors even in such situations, but the dominant approach is still to model the spatial dependencies explicitly using, e.g., Gaussian processes (GPs) (Williams & Rasmussen, 2006) or Gaussian

[1]Division of Statistics and Machine Learning, Department of Computer and Information Science, Linköping University, Linköping, Sweden. Correspondence to: Per Sidén <per.siden@liu.se>.

Markov random fields (GMRFs) (Rue & Held, 2005).

In this paper we show a formal connection between GMRFs applied to lattice data and CNNs. Common GMRFs based on nearest neighbour interactions can be viewed as special cases of a generative model where the inverse mapping from the spatial field $\mathbf{x}$ to a latent variable $\mathbf{z}$ is given by a 1-layer linear CNN. Since common GP models have previously been shown to be tightly linked with GMRFs (Lindgren et al., 2011), this connection applies to certain GPs as well.

Modeling the inverse mapping ($\mathbf{x} \rightarrow \mathbf{z}$) using a CNN results in an auto-regressive (AR) spatial model, whereas using a CNN for the forward mapping ($\mathbf{z} \rightarrow \mathbf{x}$) would correspond to a moving average (MA) model (see, e.g., (Ljung, 1999) for a discussion on AR and MA models in a time series context). This has the important implication that we obtain an infinite receptive field (i.e., infinite range on the spatial dependencies in $\mathbf{x}$) even with a 1-layer model. Indeed, this is a well known property of GMRFs.

The interpretation of a GMRF as a 1-layer CNN opens up for a straightforward generalization to multi-layer architectures, resulting in *deep GMRFs* (DGMRFs). Even when all layers are linear this has important practical implications: adding layers corresponds to increasing the auto-regressive order of the GMRF, i.e., edges are added to the GMRF graph which improves its expressivity. For conventional GMRF algorithms, simply adding more edges can have have a significant impact on the computational complexity due to a reduced degree of sparsity of the resulting precision matrix. For a DGMRF, on the other hand, the structure imposed by a multi-layer CNN architecture results in a favorable computational scaling. Indeed, using variational inference for the latent variables, we obtain a learning algorithm that scales linearly both with the number of dimensions in the spatial field ("pixels") and the number of layers (AR order). Furthermore, viewing GMRFs through the lens of deep learning allows us to use well-established toolboxes for, e.g., automatic differentiation and GPU training, to facilitate simple learning of DGMRFs.

After a review of GMRFs in Section 2, we introduce the DGMRF model in Section 3. Section 4 describes how to efficiently train the model, and how to compute the posterior predictive distribution, including uncertainty. We discuss related work in Section 5. The results in Section 6 illus-

trate how DGMRFs are adaptive models, with outstanding predictive ability. Section 7 concludes.

2. Background

2.1. Gaussian Markov Random Fields

A Gaussian Markov random field (GMRF) $\mathbf{x}$ is an N-dimensional Gaussian vector with mean $\boldsymbol{\mu}$ and precision (inverse covariance) matrix $\mathbf{Q}$, so that $\mathbf{x} \sim \mathcal{N}\left(\boldsymbol{\mu}, \mathbf{Q}^{-1}\right)$. For each GMRF there exists a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, with vertices $\mathcal{V}$ that correspond to the elements in $\mathbf{x}$, and edges $\mathcal{E}$ that define their conditional independencies. For example, the vertices could represent the pixels in an image, with the edges connecting neighboring pixels. Formally,

$$\{i, j\} \notin \mathcal{E} \iff x_i \perp x_j | \mathbf{x}_{-ij}, \quad \text{for all } i \neq j,$$

where $\mathbf{x}_{-ij}$ refers to all elements except i and j, meaning that two elements x_i and x_j, that are not neighbors in the graph, are conditionally independent given the rest. Importantly, the edges $\mathcal{E}$ also determine the zero-pattern in the precision matrix $\mathbf{Q}$, as every GMRF has the property

$$\{i, j\} \in \mathcal{E} \iff Q_{ij} \neq 0, \quad \text{for all } i \neq j.$$

This means that a sparsely connected graph $\mathcal{G}$ results in a sparse precision matrix $\mathbf{Q}$, which gives great computational gains compared to working with the dense covariance matrix in many large-scale applications.

2.2. Example: GMRF Defined Using Convolution

As an example, consider the second-order intrinsic GMRF or thin-plate spline model (Rue & Held, 2005), which can be defined by $\mathbf{x} \sim \mathcal{N}\left(\mathbf{0}, \left(\mathbf{G}^\top \mathbf{G}\right)^{-1}\right)$, with

$$G_{ij} = \begin{cases} 4 & \text{, for } i = j \\ -1 & \text{, for } i \sim j \\ 0 & \text{, otherwise} \end{cases}, \quad (1)$$

where $i \sim j$ denotes adjacency[1]. Imputing missing pixel values conditioned on its second-order neighborhood using this prior is equivalent to bicubic interpolation. The non-zero elements of each row i of $\mathbf{G}$ and $\mathbf{Q}$, with respect to neighboring pixels in 2D, can be compactly represented through the stencils

$$\mathbf{w_G} : \begin{bmatrix} & -1 & \\ -1 & 4 & -1 \\ & -1 & \end{bmatrix} \qquad \mathbf{w_Q} : \begin{bmatrix} & & 1 & & \\ & 2 & -8 & 2 & \\ 1 & -8 & 20 & -8 & 1 \\ & 2 & -8 & 2 & \\ & & 1 & & \end{bmatrix}. \quad (2)$$

---

[1]It is perhaps more standard to define G_{ii} equal to the number of neighbors of pixel i, which makes a difference in the image border. Our definition is convenient here as it makes $\mathbf{G}$ invertible.

An equivalent definition of this model is to first define $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and then $\mathbf{x}$ through the inverse transform

$$\mathbf{z} = \mathbf{Gx}. \quad (3)$$

It is trivial that $\mathbf{x}$ is Gaussian with mean $\mathbf{0}$ and it can be readily verified that $\text{Cov}(\mathbf{x}) = \mathbf{G}^{-1}\mathbf{I}\mathbf{G}^{-\top} = \left(\mathbf{G}^\top\mathbf{G}\right)^{-1}$. In a third, equivalent definition the inverse transform $\mathbf{z} = \mathbf{Gx}$ is instead written as a convolution

$$\mathbf{Z} = \text{conv}\left(\mathbf{X}, \mathbf{w_G}\right),$$

where $\mathbf{Z}$ and $\mathbf{X}$ are image representations of the vectors $\mathbf{z}$ and $\mathbf{x}$. The stencil $\mathbf{w_G}$ in Eq. (2) is here used as a filter and conv() denotes *same* convolution (`padding="SAME"`), for the equivalence to hold. This observation, that a certain kind of GMRF can be defined with the convolution operator, a filter and an auxiliary variable is a key observation for DGMRFs, which are defined in Section 3.

2.3. Link Between GMRFs and Gaussian Processes

Another instructive view of GMRFs is as a represetation of a Gaussian processes (GP) with Matérn kernel (Lindgren et al., 2011). This result comes from a stochastic partial differential equation (SPDE) of the form

$$\left(\kappa^2 - \Delta\right)^\gamma \tau x(\mathbf{s}) = W(\mathbf{s}),$$

which can be shown to have a GP solution $x(\mathbf{s})$ with Matérn covariance function (Whittle, 1954; 1963). Here, $W(\mathbf{s})$ is Gaussian white noise in a continuous coordinate $\mathbf{s}$, Δ is the Laplacian operator, and κ, τ and γ are hyperparameters that appear in the Matérn kernel. In particular, γ controls the smoothness of the GP. Moreover, for positive integer values of γ, a numerical finite difference approximation to the SPDE, on the integer lattice, is given by

$$\tau\left(\kappa^2\mathbf{I} + \mathbf{G}\right)^\gamma \mathbf{x} = \mathbf{z}, \quad (4)$$

with $\mathbf{G}$ defined as in Eq. (1). As in Eq. (3), this inverse transform describes a GMRF, here with precision matrix $\mathbf{Q} = \tau^2((\kappa^2\mathbf{I} + \mathbf{G})^\gamma)^\top(\kappa^2\mathbf{I} + \mathbf{G})^\gamma$. Firstly, this provides a sparse representation of a GP as a GMRF, with a discrepancy that can be reduced by making the discretization of the SPDE finer. Secondly, it gives an interpretation of GMRFs as models with similar properties as GPs.

2.4. GMRFs as Spatial Priors

GMRFs are commonly used as spatial priors for latent variables, which is common in spatial statistics and image analysis. In the simplest case, the data $\mathbf{y}$ is modeled as Gaussian, and conditionally independent given $\mathbf{x}$

$$p(\mathbf{y}|\mathbf{x}) = \prod_{i \in \mathcal{M}} p(y_i|x_i), \quad y_i|x_i \sim \mathcal{N}\left(y_i|x_i, \sigma^2\right),$$

where $\mathcal{M} \subseteq \{1, \ldots, N\}$ is the set of observed pixels with $M = |\mathcal{M}|$. Typical problems include inpainting ($M < N$), and denoising ($\sigma^2 > 0$), where the target is to reconstruct the latent $\mathbf{x}$. Conveniently, in this situation the GMRF prior is conjugate, so the posterior is also a GMRF

$$\mathbf{x}|\mathbf{y} \sim \mathcal{N}(\tilde{\boldsymbol{\mu}}, \tilde{\mathbf{Q}}^{-1}), \quad \text{with}$$
$$\tilde{\mathbf{Q}} = \mathbf{Q} + \frac{1}{\sigma^2}\mathbf{I_m}, \quad \tilde{\boldsymbol{\mu}} = \tilde{\mathbf{Q}}^{-1}\left(\mathbf{Q}\boldsymbol{\mu} + \frac{1}{\sigma^2}\mathbf{y}\right). \quad (5)$$

Here, the observation mask $\mathbf{m}$ has value 0 for missing pixels and 1 elsewhere, $\mathbf{y}$ are the observations with value 0 at missing pixels and $\mathbf{I_m}$ is the identity matrix with diagonal element 0 at missing pixels. Although the posterior is on closed form, the computational cost associated with $\tilde{\mathbf{Q}}^{-1}$, needed for the posterior mean and marginal standard deviations, can be high. This is discussed more in Section 4.

3. Deep Gaussian Markov Random Fields

3.1. Linear DGMRFs

We define a linear DGMRF $\mathbf{x}$ using an auxiliary standard Gaussian vector $\mathbf{z}$ and a bijective function $\mathbf{g}_{\boldsymbol{\theta}} : \mathbb{R}^N \to \mathbb{R}^N$,

$$\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \qquad \mathbf{z} = \mathbf{g}_{\boldsymbol{\theta}}(\mathbf{x}).$$

In other words, we define $\mathbf{x}$ through the inverse transform $\mathbf{g}_{\boldsymbol{\theta}}^{-1}$. The function $\mathbf{g}_{\boldsymbol{\theta}}(\mathbf{x})$ is for now assumed to be linear, so we can write $\mathbf{g}_{\boldsymbol{\theta}}(\mathbf{x}) = \mathbf{G}_{\boldsymbol{\theta}}\mathbf{x} + \mathbf{b}_{\boldsymbol{\theta}}$, where $\mathbf{G}_{\boldsymbol{\theta}}$ is an invertible square matrix. The non-linear case is considered in Section 3.4. We will specify $\mathbf{g}_{\boldsymbol{\theta}}(\mathbf{x})$ using a CNN with L layers. Let $\mathbf{Z}_l$ be a tensor of dimension $H \times W \times C$, with height H, width W and C channels, and let $\mathbf{z}_l = \text{vec}(\mathbf{Z}_l)$ be its vectorized version, with length $N = HWC$. The output of layer l is defined as

$$\mathbf{Z}_l = \text{conv}(\mathbf{Z}_{l-1}, \mathbf{w}_l) + \mathbf{b}_l, \quad (6)$$

where $\mathbf{w}_l$ is a 4D-tensor containing $C \times C$ 2D-filters and $\mathbf{b}_l$ is a set of C biases. Here, conv() refers to multichannel same convolution, more details are given in Section 3.3. In particular, we define $\mathbf{Z}_L \triangleq \mathbf{Z}$ and $\mathbf{Z}_0 \triangleq \mathbf{X}$. An illustration of the model is given in Figure 1. The model parameters $\boldsymbol{\theta} = \{(\mathbf{w}_l, \mathbf{b}_l) : l = 1, \ldots, L\}$ will be omitted in the following for brevity.

Just as for normalizing flows (Dinh et al., 2014; Rezende & Mohamed, 2015), $\mathbf{g}$ can be seen as a sequence of bijections $\mathbf{g} = \mathbf{g}_L \circ \mathbf{g}_{L-1} \circ \cdots \circ \mathbf{g}_1$, each with corresponding transform matrix $\mathbf{G}_l$. Since $\mathbf{g}$ is linear, $\mathbf{x}$ is a GMRF with density

$$p(\mathbf{x}) = \frac{|\det(\mathbf{G})|}{(2\pi)^{N/2}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^\top \mathbf{G}^\top \mathbf{G}(\mathbf{x} - \boldsymbol{\mu})\right), \quad (7)$$

with $\mathbf{G} = \mathbf{G}_L\mathbf{G}_{L-1}\cdots\mathbf{G}_1$ and the mean $\boldsymbol{\mu} = -\mathbf{G}^{-1}\mathbf{b}$ where $\mathbf{b}$ can be computed as $\mathbf{b} = \mathbf{g}(\mathbf{0})$. The determinant
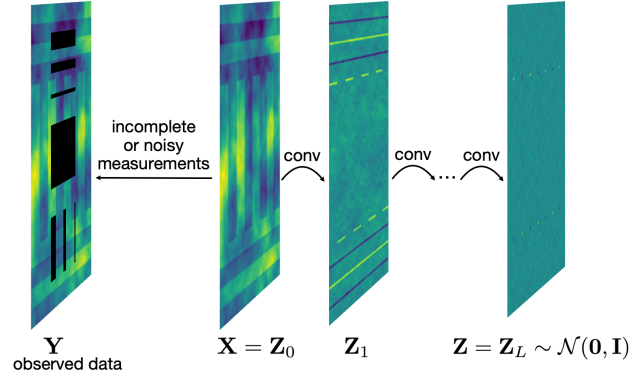

Figure 1. Illustration of the deep GMRF. The observed data $\mathbf{Y}$ are incomplete measurements of the latent DGMRF $\mathbf{X}$. The prior distribution of $\mathbf{X}$ is defined by a CNN that transforms the input $\mathbf{X}$ to the output image $\mathbf{Z}$ which has iid. standard normal pixel values.

$\det(\mathbf{G})$ can be computed as

$$\det(\mathbf{G}) = \prod_{l=1}^{L} \det(\mathbf{G}_l) \quad (8)$$

and we address the problem of making this computation fast below in Section 3.2. This GMRF has precision matrix $\mathbf{Q} = \mathbf{G}^\top\mathbf{G}$, that is guaranteed to be positive (semi-)definite for all $\boldsymbol{\theta}$, which gives an advantage compared to modeling $\mathbf{Q}$ directly. The reason for defining $\mathbf{x}$ through an inverse transform $\mathbf{z} = \mathbf{g}(\mathbf{x})$, rather than a forward transform, is twofold. Firstly, it establishes a formal connection to traditional GMRFs, see Proposition 1 below. Secondly, it gives an AR model with infinite receptive field, meaning that the output prediction at each pixel depends on all the pixels, rather than just on nearby pixels, even for a one-layer model. Compared to dilated CNNs (e.g. Yu & Koltun, 2015), which achieve long (yet finite) range dependencies through several layers of dilated filters, this is a simpler construction.

3.2. Computationally Cheap Convolutional Filters

In this paper, we consider two special forms of convolutional filters: plus ($+$) filters and sequential (seq) filters, with forms

$$+ : \begin{bmatrix} & a_3 & \\ a_2 & a_1 & a_4 \\ & a_5 & \end{bmatrix} \qquad \text{seq} : \begin{bmatrix} & a_1 & a_2 \\ a_3 & a_4 & a_5 \end{bmatrix} \quad (9)$$

where $a_1, \ldots, a_5 \in \mathbb{R}$ are parameters to be learned, and the empty positions are fixed to zero. The benefit of filters with these special designs is that they correspond to transforms for which $\det(\mathbf{G}_l)$ in Eq. (8) can be cheaply computed.

Defining a linear DGMRF through a sequence of convolutions using different small filters is of course equivalent to defining it using a single convolution with a larger filter, apart from at the boundary. The equivalent larger filter can be obtained by sequentially convolving the smaller filters with themselves. The main motivation for using the deep

definition is that it has cheap determinant calculations, using Eq. (8) and the $+$- and seq-filters, which is not the case for a general larger filter. Also, a deep definition results in fewer parameters to learn. For example, four full 3×3-filters has 36 parameters compared to 81 parameters for the corresponding 9×9-filter. Finally, the deep convolutional architecture has proven successful in countless applications of neural networks, which makes it plausible that also GM-RFs should benefit from the same inductive biases. The addition of non-linearities between the layers is discussed in Section 3.4. We now discuss the two filter types and their determinant computation for the singlechannel case and discuss the multichannel construction in Section 3.3.

3.2.1. $+$-FILTERS

We begin with two propositions, the first connecting linear DGMRFs with $+$-filters to the GMRFs in Section 2, and the second giving the cheap determinant computation.

Proposition 1. *The second-order intrinsic GMRF, as well as the GMRF approximation of a Matérn GP, are special cases of the linear DGMRF model with $+$-filters.*

Proof. The non-zero pattern of the $+$-filter is the same as that of $\mathbf{w_g}$ in Eq. (2), meaning that a one-layer linear DGMRF with the same filter weights is equivalent to the second-order intrinsic GMRF. Similarly, Eq. (4) can be written as a linear DGMRF with L layers of $+$-filters, for $L \geq \gamma$. This requires that γ of the layers have the same filter with $a_1 = 4 + \kappa^2$ and $a_2 = \cdots = a_5 = -1$, and the other $L - \gamma$ layers to be identity functions. $\qquad\square$

Proposition 2. *The linear transform matrix $\mathbf{G}_+$ defined by singlechannel same convolution of an $H \times W$ image with the $+$-filter defined in Eq. (9), has determinant*

$$\det\left(\mathbf{G}_+\right) = \prod_{i=1}^{H}\prod_{j=1}^{W}\left[a_1 + 2\sqrt{a_3 a_5}\cos\left(\pi\frac{i}{H+1}\right) + 2\sqrt{a_2 a_4}\cos\left(\pi\frac{j}{W+1}\right)\right],$$

where $\sqrt{-1}$ is treated as imaginary. Computing the determinant thus has complexity $\mathcal{O}(N)$.

The proof, given in detail in the supplement, is to show that the factors of this product are indentical to the eigenvalues of $\mathbf{G}_+$, which can be done by writing $\mathbf{G}_+$ as a Kronecker sum of tridiagonal Toeplitz matrices. Proposition 2 provides a fast method for computing the determinant, which would have complexity $\mathcal{O}(N^3)$ for a general square matrix, or $\mathcal{O}(N^{3/2})$ if based on sparse Cholesky factorization (Rue & Held, 2005). In practice, we reparameterize $a_1, \ldots, a_5$ to ensure that all the eigenvalues are real and positive, see details in the supplement. Without this constraint, we have

observed unstable, oscillating solutions in some situations, and the constraint also ensures that the bijective assumption is valid.

3.2.2. SEQ-FILTERS

For some ordering of the image pixels, the output pixels for a convolution with a seq-filter only depend on previous input pixels. This is equivalent to saying that the corresponding transform matrix $\mathbf{G}_{\text{seq}}$ yields a lower triangular matrix $\mathbf{P}^\top \mathbf{G}_{\text{seq}}\mathbf{P}$ for some permutation matrix $\mathbf{P}$, which implies that $\det(\mathbf{G}_{\text{seq}}) = a_1^N$. Seq-filters are therefore extremely cheap, but somewhat restricted compared to $+$-filters, due to the inability to model non-sequential patterns. However, the seq-filter in Eq. (9) can be rotated/mirrored in eight different ways, each encoding a different pixel ordering. Thus different sequential dependencies can be modelled in different layers. Also seq-filters can trivially be extended to 5×5- or 7×7-filters, still without changing the determinant. Connections to auto-regressive models, for example PixelCNN (Van den Oord et al., 2016b), are discussed in Section 5.

3.3. Multichannel Construction

When $C > 1$, Eq. (6) can be written on vector form as

$$\mathbf{z}_l = \mathbf{G}_l \mathbf{z}_{l-1} + \mathbf{b}_l$$
$$= \begin{bmatrix} \mathbf{G}_{l,1,1} & \cdots & \mathbf{G}_{l,1,C} \\ \vdots & \ddots & \vdots \\ \mathbf{G}_{l,C,1} & \cdots & \mathbf{G}_{l,C,C} \end{bmatrix} \mathbf{z}_{l-1} + \begin{bmatrix} b_{l,1}\mathbf{1} \\ \vdots \\ b_{l,C}\mathbf{1} \end{bmatrix},$$

where $\mathbf{G}_{l,i,j}$ is the transition matrix of a single convolution from input channel j to output channel i, and $b_{l,j}$ is the bias of output channel j. In order to make $\det(\mathbf{G}_l)$ computationally tractable, we set $\mathbf{G}_{l,i,j} = \mathbf{0}$ for $i < j$, making $\mathbf{G}_l$ lower block triangular and

$$\det\left(\mathbf{G}_l\right) = \prod_{c=1}^{C}\det\left(\mathbf{G}_{l,c,c}\right).$$

The ordering of channels could vary between different layers to allow information to flow back and forth between all channels. One could also add invertible 1×1 convolutions (Kingma & Dhariwal, 2018) between layers for this ordering to be dynamic and learnable. A multichannel DGMRF could learn more interesting representations by storing different features in different channels of the hidden layers. Even with singlechannel data, the hidden layers can be multichannel by using a multiscale architecture (Dinh et al., 2017).

3.4. Non-Linear Extension

The linear DGMRFs can be extended by adding non-linear activation functions between layers of the neural network $\mathbf{g}_{\boldsymbol{\theta}}(\mathbf{x})$. Formally, Eq. (6) is replaced by

$$\mathbf{Z}_l = \psi_l\left(\text{conv}\left(\mathbf{Z}_{l-1}, \mathbf{w}_l\right) + \mathbf{b}_l\right),$$

where ψ_l is a non-linear scalar function that operates element-wise. We restrict ψ_l to be strictly increasing, to ensure that $\mathbf{g}_{\boldsymbol{\theta}}(\mathbf{x})$ is a bijection.

The distribution of $\mathbf{x}$ can now be computed by the change of variable rule

$$\log p(\mathbf{x}) = \log p(\mathbf{z}) + \log |\det (d\mathbf{z}/d\mathbf{x})|$$

$$= \log p(\mathbf{z}) + \sum_{l=1}^{L} \log |\det (\mathbf{G}_l)| + \sum_{l=1}^{L} \sum_{i=1}^{N} \log |\psi_l'(h_{l,i})|,$$

where $\mathbf{h}_l = \text{vec}(\text{conv}(\mathbf{Z}_{l-1}, \mathbf{w}_l) + \mathbf{b}_l)$ is the input to ψ_l and ψ_l' is the derivative. Just as in the linear case, the computational cost of this density is linear.

Per default we assume that ψ_l are Parametric Rectified Linear Units (PReLUs), defined by

$$\psi_l(h) = \begin{cases} h, & \text{for } h \geq 0 \\ \alpha_l h, & \text{for } h < 0 \end{cases},$$

where α_l are learnable parameters with $\alpha_l > 0$. We can now add $\alpha_1, \ldots, \alpha_L$ to the parameters $\boldsymbol{\theta}$ and optimize them jointly.

4. Learning and Inference

There exist two kinds of unknown variables that we wish to infer: the latent field $\mathbf{x}$ and the model parameters $\boldsymbol{\theta}$. Since we are not interested in the posterior uncertainty of $\boldsymbol{\theta}$ directly, we take a practical course of action and optimize these, using a variational lower bound on the marginal likelihood $p(\mathbf{y}|\boldsymbol{\theta})$. Given the optimal value $\hat{\boldsymbol{\theta}}$, for the linear model, we make a fully Bayesian treatment of the posterior $p(\mathbf{x}|\hat{\boldsymbol{\theta}}, \mathbf{y})$.

4.1. Optimization of Parameters

For the linear DGMRF, the marginal likelihood can be written on closed form as

$$p(\mathbf{y}|\boldsymbol{\theta}) = \left. \frac{p(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta}) p(\mathbf{x}|\boldsymbol{\theta})}{p(\mathbf{x}|\mathbf{y}, \boldsymbol{\theta})} \right|_{\mathbf{x}=\mathbf{x}^*}$$

for arbitrary value of $\mathbf{x}^*$. Unfortunately, this expression requires the determinant of the posterior precision matrix $\det(\mathbf{G}^\top \mathbf{G} + \sigma^{-2} \mathbf{I_m})$, which is computationally infeasible for large N. Instead, we focus on the variational evidence lower bound (ELBO) $\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}, \mathbf{y}) \leq \log p(\mathbf{y}|\boldsymbol{\theta})$ which can be written as

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}, \mathbf{y}) = \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{x})} [-\log q_{\boldsymbol{\phi}}(\mathbf{x}) + \log p(\mathbf{y}, \mathbf{x}|\boldsymbol{\theta})],$$

where $q_{\boldsymbol{\phi}}(\mathbf{x})$ is the variational posterior approximation, which depends on variational parameters $\boldsymbol{\phi}$. We here only intend to use $q_{\boldsymbol{\phi}}(\mathbf{x})$ as a means for optimizing $\boldsymbol{\theta}$, and not

for example to make posterior predictions. For simplicity, we choose $q_{\boldsymbol{\phi}}(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\nu}, \mathbf{S})$ with diagonal covariance matrix, and $\boldsymbol{\phi} = \{\boldsymbol{\nu}, \mathbf{S}\}$. After inserting the variational and model densities and simplifying, we can write the ELBO as

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}, \mathbf{y}) = \frac{1}{2} \log |\det (\mathbf{S}_{\boldsymbol{\phi}})| - M \log \sigma_{\boldsymbol{\theta}} + \log |\det (\mathbf{G}_{\boldsymbol{\theta}})|$$

$$- \frac{1}{2} \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{x})} \left[\mathbf{g}_{\boldsymbol{\theta}}(\mathbf{x})^\top \mathbf{g}_{\boldsymbol{\theta}}(\mathbf{x}) + \frac{1}{\sigma_{\boldsymbol{\theta}}^2} (\mathbf{y} - \mathbf{x})^\top \mathbf{I_m} (\mathbf{y} - \mathbf{x}) \right], \tag{10}$$

where constant terms have been omitted, and all parameters have been subscripted with $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ to clarify whether they are model or variational parameters. Additionally, we use the reparameterization trick (Kingma & Welling, 2013) replacing the last expectation with a sum over N_q standard random samples $\boldsymbol{\varepsilon}_1, \ldots, \boldsymbol{\varepsilon}_{N_q}$, and set $\mathbf{x}_i = \boldsymbol{\nu}_{\boldsymbol{\phi}} + \mathbf{S}_{\boldsymbol{\phi}}^{1/2} \boldsymbol{\varepsilon}_i$ in the sum. This gives an unbiased estimator of the ELBO, which has low variance. Moreover, this estimator is differentiable with respect to $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ and can be used for stochastic gradient optimization with autodiff and backprop. Parameter learning will be fast, with a time complexity that is $\mathcal{O}(N)$ for a fixed number of iterations of optimization, since backprop in a CNN is linear and so are the determinant computations described in Section 3.2. This can be compared with $\mathcal{O}(N^3)$ for standard GP and $\mathcal{O}(N^{3/2})$ for standard GMRF inference in 2D problems, based on the Cholesky-decomposition. The ELBO can be trivially extended to the non-linear model by replacing $\log |\det(\mathbf{G}_{\boldsymbol{\theta}})|$ in Eq. (10).

4.2. Exact Inference for the Latent Field

For a linear DGMRF, the conditional posterior $p(\mathbf{x}|\hat{\boldsymbol{\theta}}, \mathbf{y})$ is also a GMRF, see Eq. (5). Even though computing this density is too costly in general, it is possible to compute the posterior mean and to draw samples from the posterior, which can be used for making predictions. Both require solving linear equation systems $\tilde{\mathbf{Q}}\mathbf{x} = \mathbf{c}$ involving the posterior precision matrix $\tilde{\mathbf{Q}} = \mathbf{G}^\top \mathbf{G} + \sigma^{-2} \mathbf{I_m}$. For this we use the conjugate gradient (CG) method (see e.g. Barrett et al., 1994), which is an iterative method that, rather than exactly computing $\mathbf{x} = \tilde{\mathbf{Q}}^{-1} \mathbf{c}$, iteratively minimizes the relative residual $\|\tilde{\mathbf{Q}}\mathbf{x} - \mathbf{c}\|/\|\mathbf{c}\|$ until it falls below some threshold δ, that can be set arbitrarily low. In practice, $\delta = 10^{-7}$ gives solutions that visually appear to be identical to the exact solution. CG only requires matrix-vector-multiplications, which means that the multiplications with $\mathbf{G}$ and $\mathbf{G}^\top$ can be implemented in a matrix-free fashion using convolution.

Posterior sampling for $\mathbf{x}$ can be performed using the method of Papandreou & Yuille (2010) by computing

$$\mathbf{x}_s = \tilde{\mathbf{Q}}^{-1} \left(\mathbf{G}^\top (\mathbf{u}_1 - \mathbf{b}) + \frac{1}{\sigma^2} (\mathbf{y} + \sigma \mathbf{I_m} \mathbf{u}_2) \right),$$

where $\mathbf{u}_1$ and $\mathbf{u}_2$ are standard Gaussian random vectors. It

Algorithm 1 Inference algorithm

Input: data $\mathbf{y}$, model structure, learning rate, N_q, etc.

Initialize values for the model parameters $\boldsymbol{\theta} \subseteq \{(\mathbf{w}_l, \mathbf{b}_l, \alpha_l)_{l=1,\ldots,L}, \sigma^2\}$ and the variational parameters $\boldsymbol{\phi} = \{\boldsymbol{\nu}, \mathbf{S}\}$

Optimize the ELBO (Eq. 10) wrt. $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ using Adam to obtain $\hat{\boldsymbol{\theta}}$ and $\hat{\boldsymbol{\phi}}$

if the model is linear **then**
    Compute the mean and marginal variances of the posterior $p(\mathbf{x}|\mathbf{y}, \hat{\boldsymbol{\theta}})$ using CG and simple RBMC
else
    Approximate the mean and marginal variances of the posterior $p(\mathbf{x}|\mathbf{y}, \hat{\boldsymbol{\theta}})$ with $\hat{\boldsymbol{\nu}}$ and $\mathrm{diag}(\hat{\mathbf{S}})$
end if

Compute the predictive means $\mathbb{E}(y_i^*|\mathbf{y}, \hat{\boldsymbol{\theta}}) = \mathbb{E}(x_i|\mathbf{y}, \hat{\boldsymbol{\theta}})$ and variances $\mathrm{Var}(y_i^*|\mathbf{y}, \hat{\boldsymbol{\theta}}) = \mathrm{Var}(x_i|\mathbf{y}, \hat{\boldsymbol{\theta}}) + \sigma^2$

---

can be easily verified that $\mathbf{x}_s$ is Gaussian, with the same mean and covariance as the posterior.

Given a number of posterior samples, the posterior marginal variances $\mathrm{Var}(x_i|\mathbf{y}, \hat{\boldsymbol{\theta}})$ can be naively approximated using Monte Carlo estimation, but more efficiently approximated using the simple Rao-Blackwellized Monte Carlo (simple RBMC) method by Sidén et al. (2018). This provides a way to compute the posterior predictive uncertainty as

$$\mathrm{Var}(y_i^*|\mathbf{y}, \hat{\boldsymbol{\theta}}) = \mathrm{Var}(x_i|\mathbf{y}, \hat{\boldsymbol{\theta}}) + \sigma^2.$$

The time complexity is here mainly decided by the complexity of the CG method, which can be described as $\mathcal{O}(N\sqrt{\kappa})$, where κ is the condition number of $\tilde{\mathbf{Q}}$ (Shewchuk, 1994). It is difficult to say how κ depends on N for a general $\tilde{\mathbf{Q}}$, but for a two-dimensional, second-order elliptic boundary value problems, for example the SPDE approach with $\gamma = 1$, κ is linear in N so the method is $\mathcal{O}(N^{3/2})$. Even though this is the same as for the sparse Cholesky method, we have observed CG to be a lot faster in practice. Also, the storage requirements for CG is $\mathcal{O}(N)$, versus $\mathcal{O}(N \log N)$ for Cholesky.

A drawback with non-linear DGMRFs is that they do not give a simple posterior $p(\mathbf{x}|\hat{\boldsymbol{\theta}}, \mathbf{y})$. We could use the variational approximation $q_{\boldsymbol{\phi}}(\mathbf{x})$ instead, but the proposed Gaussian independent variational family is probably too simple to approximate the true posterior in a satisfying way. We discuss possible solutions to this limitation in Section 7.

The inference algorithm is summarized in Algorithm 1.

5. Related Work

5.1. GMRFs

GMRFs have a long history in spatial and image analysis (see e.g. Woods, 1972; Besag, 1974 and Rue & Held, 2005). The mentioned SPDE approach has been extended in numerous ways leading to flexible GMRF models, with for example non-stationarity (Fuglstad et al., 2015), oscillation (Lindgren et al., 2011) and non-Gaussianity (Bolin, 2014). A nested (deep) SPDE is introduced by Bolin & Lindgren (2011), which results in a model similar to ours, but which lacks the connection to CNNs and some of the computational advantages that we provide.

Papandreou & Yuille (2010) use GMRFs for inpainting in natural images, in combination with CG to obtain posterior samples. They do not learn the form of the prior precision matrix, but instead rewrite the model as a product of Gaussian experts, and learn the mean and variance of the Gaussian factors. Papandreou & Yuille (2011) assume a heavy-tailed non-Gaussian prior for the latent pixel values, and use a variational GMRF approximation for the posterior.

5.2. GPs

GPs (Stein, 1999; Williams & Rasmussen, 2006) are commonplace for modelling spatially dependent data, not necessarily restricted to observations on the grid. The GP covariance kernel, which encodes the spatial dependencies, can be made flexible (Wilson & Adams, 2013) and deep (Dunlop et al., 2018; Roininen et al., 2019). However, standard GPs are limited by $\mathcal{O}(N^3)$ computational complexity (assuming the number of measurements $M \sim \mathcal{O}(N)$).

Inducing point methods (Quiñonero-Candela & Rasmussen, 2005; Snelson & Ghahramani, 2006) can reduce the complexity to $\mathcal{O}(P^2N + P^3)$ or even $\mathcal{O}(P^3)$ (Hensman et al., 2013), where P is the number of inducing points. However, for grid data, these methods tend to over-smooth the data (Wilson & Nickisch, 2015) unless P is chosen in the same order of magnitude as N. When the data are on a regular grid and fully observed, Kronecker and Toeplitz methods can be used for fast computation (e.g. Saatçi, 2011; Wilson, 2014). Under certain assumptions about the interactions across the input dimensions, such as additivity and separability, this can reduce the complexity to $\mathcal{O}(N \log N)$. These methods can also be extended to when the data are not on the grid (Wilson & Nickisch, 2015), or when the grid is incomplete (Stroud et al., 2016).

5.3. Deep Generative Models

Deep generative models are normally trained on large datasets, and can then be used to generate new samples from the data distribution, or to explore learned latent representa-

tions. For example, generative adversarial networks (GANs, Goodfellow et al., 2014) have been used for inpainting problems in natural images (see e.g. Yu et al., 2018; Liu et al., 2018) with impressive results. Our model is mainly targeted at smaller datasets, as single images, where too complex models are likely to overfit. Instead, we achieve an infinite receptive field through the inverse definition $\mathbf{z} = \mathbf{g}(\mathbf{x})$, which require few parameters. Also, linear DGMRFs can readily provide uncertainty estimates, which is prohibitive for most deep generative models. Overall, deep generative models are difficult to use for typical spatial problems, but these models share ideas with our proposed model as we discuss below.

Flow-based or invertible generative models, e.g., NICE (Dinh et al., 2014), normalizing flows (Rezende & Mohamed, 2015), IAF (Kingma et al., 2016), MAF (Papamakarios et al., 2017), real NVP (Dinh et al., 2017), Glow (Kingma & Dhariwal, 2018) and i-ResNet (Behrmann et al., 2019), model $\mathbf{x}$ as a bijective function of a latent variable from a simple base distribution, just as our model. The likelihood can be optimized directly for parameter learning, but generally require all pixels to be non-missing, that is, these methods are not designed to handle the situation with incomplete or noisy observations that we are considering. One connection to our work is that a linear DGMRF with seq-filters can be seen as a fully linear MAF with masked convolutional MADE-layers. Dinh et al. (2014) use a learned model for inpainting with projected gradient ascent, which gives the posterior mode but no uncertainty quantification. Lu & Huang (2019) provide a conditional version of Glow for inpainting, but this requires missing pixels to be known during training, and the predictive distribution is modeled directly and not through a Bayesian inversion of the model.

Auto-regressive models, e.g. PixelRNN (Van den Oord et al., 2016a), PixelCNN (Van den Oord et al., 2016b) and PixelCNN++ (Salimans et al., 2017), model the pixel values sequentially similarly to our proposed seq-filters, but there are some major differences. The seq-filters use masked convolutions to obtain cheap determinant computations and the ordering of pixels can change between layers, whereas auto-regressive models have no latent variables and are trained using the same image as input and output. (Van den Oord et al., 2016a) consider image completion, but are limited to the e.g. the case where the bottom half of the image is missing, and can't handle a general mask.

Variational autoencoders (Kingma & Welling, 2013) differ from our model in that they use the deep transform of the latent variables for modeling the parameters of the distribution of the data, rather than for the data directly. This makes the recovery of the latent variables intractable, which is not the case for our model, however, we still use the same variational optimization algorithm for speedups.

Deep image prior (DIP, Ulyanov et al., 2018) uses CNNs, trained on single images, for denoising and inpainting, with promising results. However, this is not truly a probabilistic generative model, since the latent variables are not inferred, but fixed to some initial random values. Thus, it would be difficult to output predictive uncertainty from such a model.

6. Experiments

We have implemented DGMRF in TensorFlow (Abadi et al., 2016), taking advantage of autodiff and GPU computations. We train the parameters, and compute the predictive distribution using the CG algorithm. To avoid boundary effects, the images are extended with a 10-pixel wide frame of missing values at all sides. Additional details about the implementation can be found in the supplement. Code for our methods and experiments are available at `https://bitbucket.org/psiden/deepgmrf`.

6.1. Toy Data

We demonstrate the behaviour of our model for an inpainting problem on the two toy datasets in Figure 2, which have size 160×120 pixels. The data in the first row are generated as a random sample from the Matérn GMRF in Eq. (4) with $\gamma = 1$, $\tau = 1$, and $\kappa^2 = 8/50^2$ corresponding to a spatial correlation range of 50 pixels. The data in the second row are the same, but we have added horizontal and vertical edges, to investigate the model's ability to capture such patterns. Column 2 shows the posterior mean for $\mathbf{x}$ when the Matérn model with the same hyperparameters is used for prediction, which gives optimal results for the first row, but which over-smooths the edges in the second row. The corresponding results for different instances of the linear DGMRF, are shown in column 3-6. They all perform well in the simple case without edges, in which the $+$-filter models contain the true model according to Proposition 1. In the case with edges, the model with depth $L = 1$ which corresponds to a standard GMRF is too simple to handle the more complex structure in the data, but for $L = 3$, all the models give reasonable results, that preserve the edges.

Figure 3 displays the learned $+$-filters of the 3-layer model, and the values of valid pixels of the hidden layers, when original data, including missing pixels, are used as input. The first two filters learns differentiation in the vertical and horizontal direction, and the third filter is close to an identity function. Most spatial structure is removed in layer 3, that is assumed to be standard normal by the model.

6.2. Satellite Data

We compare our method against some popular methods for large data sets in spatial statistics, by considering the satel-
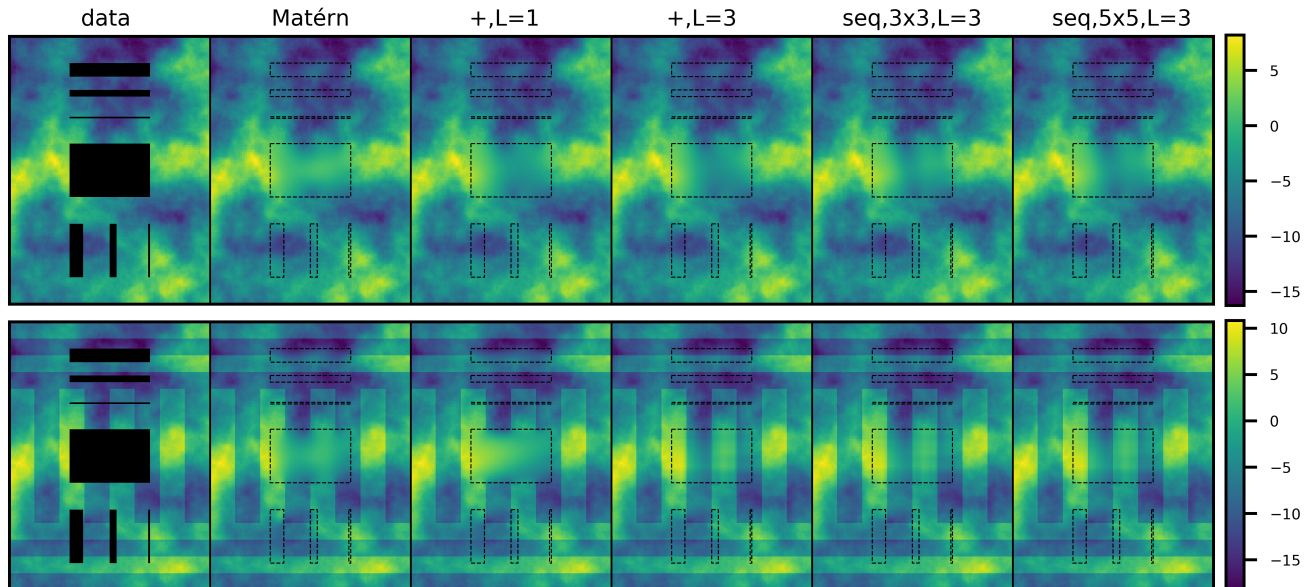
Figure 2. Posterior mean for inpainting the 160×120 pixel toy data without edges (top) and with edges (bottom). The second column shows inpainting by the same Matérn GMRF model and hyperparameters that were used to generate the data without edges.
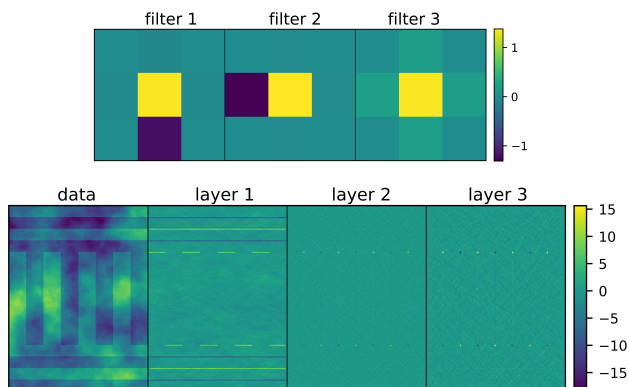

Figure 3. A linear deep GMRF with 3 layers of learned +-filters for the toy data with edges. The filters collaborate to remove spatial structures to the final layer.

lite data of daytime land surface temperatures, used in the competition by Heaton et al. (2018). The data are on a 500×300 grid, with 105,569 non-missing observations as training set. The test set consists of 42,740 observations and have been selected as the pixels that were missing due to cloud cover on a different date. The dataset and the missing pixels are shown in the supplement, together with the posterior output from our model. The data and code for some of the methods can be found at https://github.com/finnlindgren/heatoncomparison. The participants of the competition were instructed to use exponential correlation if applicable, and to add a constant mean and linear effects of the latitude and longitude. For this rea-

son, we extend the measurement equation to include linear trends, so that

$$y_i|x_i \sim \mathcal{N}\left(y_i|x_i + \mathbf{F}_{i,\cdot}\boldsymbol{\beta}, \sigma^2\right), \qquad (11)$$

where $\mathbf{F}$ is a spatial covariate matrix with columns corresponding to (constant, longitude, latitude), and $\boldsymbol{\beta}$ is a 3-dimensional regression coefficient vector, which can be integrated out jointly with $\mathbf{x}$ for the predictions, see details in the supplement.

Table 1 compares different instances of our model with the methods in the competition, which are briefly described in the supplement, and in more detail in Heaton et al. (2018). One of the top competitors, SPDE, is essentially the same method as described in Section 2.3. For comparison with another deep method, we have also included results for DIP using the authors own implementation[2]. For DIP, removal of linear trends and normalization to [0, 1] were done in preprocessing, and these steps were inverted before evaluation. As DIP does not give uncertainty estimates, we also compare with an ensemble of 10 DIP models, using the ensemble mean and standard deviation as predictive distribution. The scores used are mean absolute error (MAE), root-mean-squared-error (RMSE), mean continuous rank probability score (CRPS), mean interval score (INT), and prediction interval coverage (CVG). CRPS and INT are proper scoring rules, that also account for the predictive uncertainty (Gneiting & Raftery, 2007). These are the same scores that were used in the competition.

---

[2]https://github.com/DmitryUlyanov/deep-image-prior

Table 1. Prediction scores on the satellite data. The scores of the methods in the upper pane come from Table 3 in Heaton et al. (2018). Our models are presented in the lower pane. Lower scores are better, except from CVG, for which 0.95 is optimal. The results for our models are averages computed across five random seeds. Standard deviations across seeds are shown in parenthesis for $\text{seq}_{5 \times 5, L=5}$, and for the other models in the supplement.

Method	MAE	RMSE	CRPS	INT	CVG
FRK	1.96	2.44	1.44	14.08	0.79
Gapfill	1.33	1.86	1.17	34.78	0.36
LatticeKrig	1.22	1.68	0.87	7.55	0.96
LAGP	1.65	2.08	1.17	10.81	0.83
MetaKriging	2.08	2.50	1.44	10.77	0.89
MRA	1.33	1.85	0.94	8.00	0.92
NNGP	1.21	1.64	0.85	7.57	**0.95**
Partition	1.41	1.80	1.02	10.49	0.86
Pred. Proc.	2.15	2.64	1.55	15.51	0.83
SPDE	1.10	1.53	0.83	8.85	0.97
Tapering	1.87	2.45	1.32	10.31	0.93
Peri. Embe.	1.29	1.79	0.91	7.44	0.93
DIP	1.53	2.06	-	-	-
DIP ensemble	1.30	1.67	0.96	11.82	0.72
DGMRF (our)					
$\text{seq}_{5 \times 5, L=1}$	1.06	1.42	0.76	7.21	0.97
$\text{seq}_{5 \times 5, L=3}$	0.95	1.3	0.75	8.29	0.97
$\text{seq}_{5 \times 5, L=5}$	**0.93**	**1.25**	**0.74**	8.14	0.97
	(.037)	(.051)	(.012)	(.461)	(.001)
$\text{seq}_{3 \times 3, L=5}$	1.16	1.57	0.81	**6.98**	0.97
$+_{L=5}$	1.09	1.47	0.78	7.63	0.97
$\text{seq}_{5 \times 5, L=5, \text{NL}}$	1.37	1.87	-	-	-

For the DGMRFs, based on the first three scores, the seq-filters of size 5×5 perform better compared to those of size 3×3 and the +-filters, which may be due to the increased flexibility of larger filters. Moreover, deeper models tend to give better results. For the non-linear (NL) model, CG cannot be used to compute posterior mean and uncertainty, and the first two scores are instead computed based on the mean of the variational approximation. We note that the NL model performs worse than the linear. Our primary explanation for this disappointing result is that the variational posterior is insufficient in approximating the true posterior. An independent distribution naturally has limited ability to approximate a spatially dependent posterior, and empirically we have also seen that the variational approximation for the linear model performs much worse than the true posterior (results not shown). DGMRF outperforms all the methods from the competition on all criteria except CVG, where it is slightly worse than NNGP. In terms of MAE, RMSE and CRPS the improvement is substantial. It is difficult to compare computing times due to different hardware, but our method takes roughly 2.5h for the $\text{seq}_{5 \times 5, L=5}$ model using a Tesla K40 GPU, out of which roughly 95% of the time is for parameter learning.

7. Conclusions and Future Work

We have proposed deep GMRFs which enable us to view (high-order) GMRF models as CNNs. We have focused our attention on lattice-based graphs to clearly show the connection to conventional CNNs, however, the proposed method can likely be generalized to arbitrary graphs via graph convolutions (see, e.g., Xu et al., 2019), and to continuously referenced data, similar to Bolin & Lindgren (2011). We have also primarily considered *linear* CNNs, resulting in a model that is indeed equivalent to a GMRF. The DGMRF nevertheless has favorable properties compared to conventional algorithms for GMRFs: the CNN architecture opens up for simple and efficient learning, even though the corresponding graph will have a high connectivity when using multiple layers of the CNN. Empirically we have found that multi-layer architectures are indeed very useful even in the linear case, enabling the model to capture complex data dependencies such as distinct edges, and result in state-of-the-art performance on a spatial benchmark problem.

Using a CNN for the inverse mapping when defining the DGMRF results in a spatial AR model. This gives rise to an infinite receptive field, but can also result in instability of the learned prior. We have constrained the filter parameters to yield real eigenvalues, and under this constraint we have not seen any issues with instability. A more principled way of addressing this potential issue is left for future work.

The CNN-based interpretation offers a natural extension, namely to introduce non-linearities to enable modeling of more complex distributions for the prior $p(\mathbf{x})$. A related extension would be to replace $p(\mathbf{x})$ with one of the flow-based or auto-regressive generative models mentioned in Section 5.3, but appropriately restricted to avoid overfitting on a single image. Further exploring these extensions requires future work. In particular we believe that the independent variational approximation is insufficient to accurately approximate the posterior distribution over latent variables in non-linear DGMRFs. One way to address this limitation is to parameterize the square-root of the covariance matrix $\mathbf{S}_\phi^{1/2}$ of the variational approximation q_ϕ directly as a lower triangular matrix. Another approach is to model q_ϕ as a GMRF with the same graph structure as the original model.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pp. 265–283, 2016.

Barrett, R., Berry, M. W., Chan, T. F., Demmel, J., Donato, J., Dongarra, J., Eijkhout, V., Pozo, R., Romine, C., and

Van der Vorst, H. *Templates for the solution of linear systems: building blocks for iterative methods*, volume 43. Siam, Philadelphia, PA, 1994.

Behrmann, J., Grathwohl, W., Chen, R. T. Q., Duvenaud, D., and Jacobsen, J.-H. Invertible residual networks. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 573–582, Long Beach, California, USA, 09–15 Jun 2019. PMLR. URL http://proceedings.mlr.press/v97/behrmann19a.html.

Besag, J. Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, 36(2):192–236, 1974.

Bolin, D. Spatial matérn fields driven by non-gaussian noise. *Scandinavian Journal of Statistics*, 41(3):557–579, 2014.

Bolin, D. and Lindgren, F. Spatial models generated by nested stochastic partial differential equations, with an application to global ozone mapping. *Annals of Applied Statistics*, 5(1):523–550, 2011.

Dinh, L., Krueger, D., and Bengio, Y. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

Dinh, L., Sohl-Dickstein, J., and Bengio, S. Density estimation using real NVP. *arXiv preprint arXiv:1605.08803*, 2017.

Dunlop, M. M., Girolami, M. A., and Stuart, A. M. How Deep Are Deep Gaussian Processes? *Journal of Machine Learning Research*, 19:1–46, 2018.

Fuglstad, G.-A., Lindgren, F., Simpson, D., and Rue, H. Exploring a new class of non-stationary spatial Gaussian random fields with varying local anisotropy. *Statistica Sinica*, 25(1):115–133, 2015.

Gneiting, T. and Raftery, A. E. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378, 2007.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Guinness, J. and Fuentes, M. Circulant embedding of approximate covariances for inference from gaussian data on large lattices. *Journal of computational and Graphical Statistics*, 26(1):88–97, 2017.

Heaton, M. J., Datta, A., Finley, A. O., Furrer, R., Guinness, J., Guhaniyogi, R., Gerber, F., Gramacy, R. B.,

Hammerling, D., Katzfuss, M., Lindgren, F., Nychka, D. W., Sun, F., and Zammit-mangion, A. A Case Study Competition Among Methods for Analyzing Large Spatial Data. *Journal of Agricultural, Biological and Environmental Statistics*, 24(3):398–425, 2018. doi: 10.1007/s13253-018-00348-w.

Hensman, J., Fusi, N., and Lawrence, N. D. Gaussian Processes for Big Data. In *Uncertainty in Artificial Intelligence (UAI)*, 2013.

Kingma, D. P. and Dhariwal, P. Glow : Generative Flow with Invertible 1x1 Convolutions. In *Advances in Neural Information Processing Systems*, pp. 10215–10224, 2018.

Kingma, D. P. and Welling, M. Auto-Encoding Variational Bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Kingma, D. P., Salimans, T., Jozefowicz, R., Chen, X., Sutskever, I., and Welling, M. Improved variational inference with inverse autoregressive flow. In *Advances in neural information processing systems*, pp. 4743–4751, 2016.

Lindgren, F., Rue, H., and Lindström, J. An explicit link between Gaussian fields and Gaussian Markov random fields: The SPDE approach. *Journal of the Royal Statistical Society Series B*, 73(4):423–498, 2011.

Liu, G., Reda, F. A., Shih, K. J., Wang, T.-C., Tao, A., and Catanzaro, B. Image inpainting for irregular holes using partial convolutions. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 85–100, 2018.

Ljung, L. *System identification, Theory for the user*. System sciences series. Prentice Hall, Upper Saddle River, NJ, USA, second edition, 1999.

Lu, Y. and Huang, B. Structured Output Learning with Conditional Generative Flows. *arXiv preprint arXiv:1905.13288*, 2019.

Papamakarios, G., Pavlakou, T., and Murray, I. Masked autoregressive flow for density estimation. In *Advances in Neural Information Processing Systems*, pp. 2338–2347, 2017.

Papandreou, G. and Yuille, A. Gaussian sampling by local perturbations. *Advances in Neural Information Processing Systems 23*, 90(8):1858–1866, 2010.

Papandreou, G. and Yuille, A. L. Efficient variational inference in large-scale Bayesian compressed sensing. *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pp. 1332–1339, 2011.

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

Rezende, D. and Mohamed, S. Variational inference with normalizing flows. In *Proceedings of the 32nd International Conference on Machine Learning*, pp. 1530–1538, 2015.

Roininen, L., Girolami, M., Lasanen, S., and Markkanen, M. Hyperpriors for Matérn fields with applications in Bayesian inversion. *Inverse Problems & Imaging*, 13:1–29, 2019. URL http://aimsciences.org//article/id/d17bde6b-3e5f-438d-af0a-b712cf433748.

Rue, H. and Held, L. *Gaussian Markov random fields: theory and applications*. CRC Press, 2005.

Saatçi, Y. *Scalable Inference for Structured Gaussian Process Models*. PhD thesis, University of Cambridge, 2011.

Salimans, T., Karpathy, A., Chen, X., and Kingma, D. P. Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications. *arXiv preprint arXiv:1701.05517*, 2017.

Shewchuk, J. R. An introduction to the conjugate gradient method without the agonizing pain. *Technical Report CS-94-125, Carnegie Mellon University*, 1994.

Sidén, P., Lindgren, F., Bolin, D., and Villani, M. Efficient covariance approximations for large sparse precision matrices. *Journal of Computational and Graphical Statistics*, 27(4):898–909, 2018.

Snelson, E. and Ghahramani, Z. Sparse gaussian processes using pseudo-inputs. In *Advances in neural information processing systems*, pp. 1257–1264, 2006.

Stein, M. L. Interpolation of spatial data: some theory for kriging. *Springer*, 1999.

Stroud, J. R., Stein, M. L., and Lysen, S. Bayesian and Maximum Likelihood Estimation for Gaussian Processes on an Incomplete Lattice. *Journal of Computational and Graphical Statistics*, 26(1):108–120, 2016.

Ulyanov, D., Vedaldi, A., and Lempitsky, V. Deep image prior. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9446–9454, 2018.

Van den Oord, A., Kalchbrenner, N., and Kavukcuoglu, K. Pixel Recurrent Neural Networks. *arXiv preprint arXiv:1601.06759*, 2016a.

Van den Oord, A., Kalchbrenner, N., Vinyals, O., Espeholt, L., Graves, A., and Kavukcuoglu, K. Conditional Image Generation with PixelCNN Decoders. In *Advances in neural information processing systems*, 2016b.

Whittle, P. On stationary processes in the plane. *Biometrika*, 41:434–449, 1954.

Whittle, P. Stochastic processes in several dimensions. *Bulletin of the International Statistical Institute*, 40(2):974–994, 1963.

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

Wilson, A. and Adams, R. Gaussian process kernels for pattern discovery and extrapolation. In *International Conference on Machine Learning*, pp. 1067–1075, 2013.

Wilson, A. and Nickisch, H. Kernel interpolation for scalable structured gaussian processes (kiss-gp). In *International Conference on Machine Learning*, pp. 1775–1784, 2015.

Wilson, A. G. *Covariance kernels for fast automatic pattern discovery and extrapolation with Gaussian processes*. PhD thesis, University of Cambridge, 2014.

Woods, J. W. Two-Dimensional Discrete Markovian Fields. *IEEE Transactions on Information Theory*, 18(2):232–240, 1972.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=ryGs6iA5Km.

Yu, F. and Koltun, V. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015.

Yu, J., Lin, Z., Yang, J., Shen, X., Lu, X., and Huang, T. S. Generative image inpainting with contextual attention. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5505–5514, 2018.