A Designing total variation priors

To develop a probabilistic DIP, we describe first how to design a tractable TV prior. We reinterpret the TV regulariser eq. (2) as a prior over images, favouring those with low ℓ^1 norm gradients

$$p(x) = Z_\lambda^{-1} \exp(-\lambda \mathrm{TV}(x)), \tag{23}$$

where $Z_\lambda = \int \exp(-\lambda \mathrm{TV}(x))\, \mathrm{d}x$. This prior is intractable because Z_λ does not admit a closed form; thus approximations are necessary. We now explore alternatives without this limitation.

A.1 Further discussion on the TV regulariser as a prior

It is tempting to think that we do not need the PredCP machinery in section 5.3 to translate the TV regulariser into the parameter space. Indeed, the Laplace method simply involves a quadratic approximation around a mode of the log posterior, without placing any requirements on the prior used to induce said posterior. Hence, we can decompose the Hessian of the log posterior $\log p(\theta|y)$ into the contributions from the likelihood and the prior as

$$\frac{\partial}{\partial \theta^2} \left(\log p(y|\mathrm{A}x(\theta)) + \log p(x(\theta)) \right) |_{\theta=\hat{\theta}}$$

and realise that the log of the anisotropic TV prior $p(x) \propto \exp(-\lambda \mathrm{TV}(x))$ as in eq. (23) is only once differentiable. Ignoring the origin (where the absolute value function is non-differentiable), we obtain:

$$\frac{\partial}{\partial \theta^2} \log p(x(\theta))|_{\theta=\hat{\theta}} \propto -\frac{\partial}{\partial \theta^2} \mathrm{TV}(x(\theta))|_{\theta=\hat{\theta}} = 0.$$

Thus, a naive application of the Laplace approximation would eliminate the effect of the prior, leaving the posterior ill defined. In practice, one may smooth the non-smooth region around the origin, but the amount of smoothing can significantly influence the behaviour of the Hessian approximation.

A.2 Further discussion on inducing TV-smoothness with Gaussian priors

A standard alternative to enforce local smoothness in an image is to adopt a Gaussian prior $p(x) = \mathcal{N}(x; \mu, \Sigma_{xx})$ with covariance $\Sigma_{xx} \in \mathbb{R}^{d_x \times d_x}$ given by

$$[\Sigma_{xx}]_{ij,i'j'} = \sigma^2 \exp\left(\frac{-\mathrm{d}(i-i', j-j')}{\ell} \right), \tag{24}$$

where i, j index the spatial locations of pixels of x, as in eq. (2), and $\mathrm{d}(a, b) = \sqrt{a^2 + b^2}$ denotes the Euclidean vector norm. Equation (24) is also known as the Matern-$1/2$ kernel and matches the covariance of Brownian motion (Guttorp & Gneiting, 2005). The hyperparameter $\sigma^2 \in \mathbb{R}^+$ informs the pixel amplitude while the lengthscale parameter $\ell \in \mathbb{R}^+$ determines the correlation strength between nearby pixels. The TV in eq. (2) only depends on pixel pairs separated by one pixel ($\mathrm{d} = 1$), allowing analytical computation of the expected TV associated with the Gaussian prior

$$\kappa := \mathbb{E}_{x \sim \mathcal{N}(\mu, \Sigma_{xx})}[\mathrm{TV}(x)] = c\sqrt{\sigma^2(1-\rho)}, \tag{25}$$

with the correlation coefficient $\rho = \exp(-\ell^{-1}) \in (0, 1)$ and $c = {}^{4\sqrt{d_x}(\sqrt{d_x}-1)}/_{\sqrt{\pi}}$ for square images. See appendix A.3 for derivations. Increasing ℓ (for a fixed σ^2) favours x with low TV on average, resulting in smoother images. The prior $\mathcal{N}(x; \mu, \Sigma_{xx})$ is conjugate to the likelihood implied by the least-square fidelity $\mathcal{N}(y; \mathrm{A}x, \sigma_y^2 \mathrm{I})$, leading to a closed form posterior predictive distribution and marginal likelihood objective with costs $\mathcal{O}(d_y^3)$ and $\mathcal{O}(d_y^2 d_x)$, respectively.

A.3 Derivation of the identity eq. (9)

The identity follows from the following result (appendix, (McGraw & Wong, 1994)). The short proof is recalled for the convenience of the reader.

Lemma A.1. *Let X and Y be normal random variables with mean μ, variance σ^2 and correlation coefficient ρ. Let $Z = |X - Y|$. Then*

$$\mathbb{E}[Z] = \frac{2}{\sqrt{\pi}}\sqrt{\sigma^2(1-\rho)}.$$

Proof. Clearly, $X - Y$ follows a Gaussian distribution with mean 0 and variance $2\sigma^2(1-\rho)$. Then the random variable

$$W = \frac{Z^2}{2\sigma^2(1-\rho)} = \Big(\frac{X-Y}{\sqrt{2\sigma^2(1-\rho)}}\Big)^2$$

follows χ_1^2 distribution. Then

$$\mathbb{E}[\sqrt{W}] = \int_0^\infty W^{\frac{1}{2}} \frac{1}{\Gamma(\frac{1}{2})\sqrt{2}} W^{\frac{1}{2}-1} e^{-\frac{W}{2}} \mathrm{d}W = \frac{\sqrt{2}}{\Gamma(\frac{1}{2})} = \frac{\sqrt{2}}{\sqrt{\pi}},$$

where $\Gamma(z)$ denotes the Euler's Gamma function, with $\Gamma(\frac{1}{2}) = \sqrt{\pi}$. Then it follows that

$$\mathbb{E}[Z] = \sqrt{2\sigma^2(1-\rho)}\mathbb{E}[\sqrt{W}] = \frac{2}{\sqrt{\pi}}\sqrt{\sigma^2(1-\rho)}.$$

This shows the assertion in the lemma. □

Now by the marginalisation property of multivariate Gaussians, any two neighbouring pixels of x for $x \sim \mathcal{N}(\mu, \Sigma_{xx})$ satisfy the conditions of Lemma A.1, with $\rho = \exp(-\ell^{-1}) \in (0, 1)$. Thus Lemma A.1 and the trivial fact $d_x = h \times w$ imply

$$\kappa_d = \mathbb{E}_{\mathcal{N}(x;\mu,\Sigma_{xx})}[\mathrm{TV}(x)] = \frac{2[2hw - h - w]}{\sqrt{\pi}}\sqrt{\sigma^2(1-\rho)}.$$

In particular, for a square image, $h = w = \sqrt{d_x}$, we obtain the desired identity in eq. (9).

B Derivation of the linearised deep image prior

B.1 Posterior predictive covariance

We provide an alternative derivation of the posterior predictive covariance of the linearised DIP by reasoning in the parameter space. First we have linearised the neural network $x(\theta)$, turning it into a Bayesian basis function linear model (Khan et al., 2019). The probabilistic model in eq. (12) is thus:

$$y|\theta \sim \mathcal{N}(\mathrm{A}h(\theta), \sigma_y^2\mathrm{I}), \quad \theta|\ell \sim \mathcal{N}\left(0, \Sigma_{\theta\theta}\right),$$

and the linearised Laplace approximate posterior distribution over weights is given by Immer et al. (2021b)

$$p(\theta|y) \approx \mathcal{N}(\theta;\, \hat{\theta}, \Sigma_{\theta|y}) \quad \text{with} \quad \Sigma_{\theta|y} = \left(\sigma_y^{-2} J^\top \mathrm{A}^\top \mathrm{A} J + \Sigma_{\theta\theta}^{-1}\right)^{-1}. \tag{26}$$

In this work we exploit the equivalence between basis function linear models and Gaussian Processes (GP), and perform inference using the dual GP formulation. This is advantageous due to its lower computational cost when $d_\theta >> d_y$, which is common in tomographic reconstruction.

We switch to the dual formulation using the SMW matrix inversion identity, we have

$$\Sigma_{\theta|y} = \left(\sigma_y^{-2} J^\top \mathrm{A}^\top \mathrm{A} J + \Sigma_{\theta\theta}^{-1}\right)^{-1} = \Sigma_{\theta\theta} - \Sigma_{\theta\theta} J^\top \mathrm{A}^\top (\sigma_y^2 \mathrm{I} + \mathrm{A} J \Sigma_{\theta\theta} J^\top \mathrm{A}^\top)^{-1} \mathrm{A} J \Sigma_{\theta\theta} \tag{27}$$

The predictive distribution over images can be built by marginalising the NN parameters in the conditional likelihood $p(x|y) = \int p(x|\theta)p(\theta|y)\,\mathrm{d}\theta$. Since $h(\cdot)$ is a deterministic function, we have $p(x|\theta) = \delta(x - h(\theta))$ and

$$\int p(x|\theta)p(\theta|y)\,\mathrm{d}\theta = \int \delta(x - h(\theta))\mathcal{N}(\theta; \hat{\theta}, \Sigma_{\theta|y})\,\mathrm{d}\theta = \mathcal{N}(x; \hat{x}, J\Sigma_{\theta|y}J^\top).$$

Note that this assumes $\hat{\theta}$ to be a mode of the DIP training loss eq. (5). In practise, this will not be satisfied and thus the posterior mean of the linear model $\hat{\theta}_h$, which is given as the minima of the linear model's loss introduced in section 5.4, will not match that of the NN, that is, $\hat{\theta}$. Using the linear model's exact mode is only necessary for the purpose of constructing the marginal likelihood objective (Antorán et al., 2022; Antorán et al., 2022) (see also appendix B.2). However, for the purpose of making predictions, assuming $\hat{\theta}$ to be the mode allows us to keep the DIP reconstruction $\hat{x}$ as the predictive mean.

B.2 Laplace marginal likelihood and Type-II MAP in eq. (17)

For the purpose of uncertainty estimation, we tune the hyperparameters of our linear model using the marginal likelihood of the conditional-on-ℓ Gaussian-linear model introduced in eq. (6). The posterior mode of the TV-regularised linearised model is given by $\hat{\theta}_h = \text{argmin}_{\theta_h} \sigma_y^{-2} \|\mathrm{A}h(\theta_h) - y\| + \lambda \mathrm{TV}(h(\theta_h))$. However, we substitute the TV with a multivariate Gaussian surrogate $p(\theta|\ell)$. Now we derive the marginal log-likelihood (MLL) for the linearised model conditional on ℓ following Antorán et al. (2022). In Bayes rule

$$\log p(\theta|y, \ell; \sigma_y^2, \sigma^2) = \log p(y|\theta; \sigma_y^2) + \log p(\theta|\ell; \sigma^2) - \log\, p(y|\ell; \sigma_y^2, \sigma^2),$$

we isolate the MLL $\log\, p(y|\ell; \sigma_y^2, \sigma^2)$, evaluate at the linear model's posterior mode $\theta = \hat{\theta}_h$ and obtain

$$\log p(y|\ell; \sigma_y^2, \sigma^2) = \log p(y|\theta{=}\hat{\theta}_h; \sigma_y^2) + \log p(\theta{=}\hat{\theta}_h|\ell; \sigma^2) - \log p(\theta{=}\hat{\theta}_h|y, \ell; \sigma_y^2, \sigma^2). \tag{28}$$

The log-density $\log p(y|\theta = \hat{\theta}_h; \sigma_y^2)$ quantifies the quality of the model's fit to the data y, and is given by

$$\log p(y|\theta = \hat{\theta}_h; \sigma_y^2) = -\frac{d_y}{2} \log(2\pi) - \frac{1}{2} \log |\sigma_y^2 \mathrm{I}| - \frac{1}{2\sigma_y^2} \|y - Ah(\hat{\theta}_h)\|_2^2.$$

However, since our predictive mode is given by the DIP reconstruction and not the linear model's reconstruction, we depart from the exact expression for the linear model's MLL and use $-\frac{d_y}{2} \log(2\pi) - \frac{1}{2} \log |\sigma_y^2 \mathrm{I}| - \frac{1}{2\sigma_y^2} \|y - \mathrm{A}x(\hat{\theta})\|_2^2$ as the data fit term instead. The weight-mode log prior density $\log p(\theta{=}\hat{\theta}_h|\ell, \sigma^2)$ is given by

$$\log p(\theta{=}\hat{\theta}_h|\ell, \sigma^2) = -\frac{d_\theta}{2} \log(2\pi) - \frac{1}{2} \log |\Sigma_{\theta\theta}| - \frac{1}{2} \hat{\theta}_h^\top \Sigma_{\theta\theta}^{-1} \hat{\theta}_h.$$

Evaluating the Gaussian posterior log density over θ at its mode $\hat{\theta}_h$ cancels the exponent of the Gaussian and leaves us with just the normalising constant

$$\log p(\theta{=}\hat{\theta}_h|y, \ell; \sigma_y^2, \sigma^2) = -\frac{1}{2} \log |\Sigma_{\theta|y}| - \frac{d_\theta}{2} \log(2\pi)$$

By the matrix determinant lemma, the determinant $|\Sigma_{\theta|y}|$ is given by

$$|\Sigma_{\theta|y}| = |\sigma_y^{-2} J^\top \mathrm{A}^\top \mathrm{A} J + \Sigma_{\theta\theta}^{-1}|^{-1} = |\mathrm{A} J \Sigma_{\theta\theta} J^\top \mathrm{A}^\top + \sigma_y^2 \mathrm{I}|^{-1} |\Sigma_{\theta\theta}| |\sigma_y^2 \mathrm{I}|. \tag{29}$$

Thus, the linearised Laplace marginal likelihood is given by

$$\begin{aligned}
\log p(y|\ell; \sigma_y^2, \sigma^2) = &-\frac{1}{2} \log |\sigma_y^2 \mathrm{I}| - \frac{1}{2\sigma_y^2} \|y - \mathrm{A}x(\hat{\theta})\|_2^2 - \frac{1}{2} \log |\Sigma_{\theta\theta}| - \frac{1}{2} \hat{\theta}_h^\top \Sigma_{\theta\theta}^{-1} \hat{\theta}_h \\
&- \frac{1}{2} \log |\mathrm{A} J \Sigma_{\theta\theta} J^\top \mathrm{A}^\top + \sigma_y^2 \mathrm{I}| + \frac{1}{2} \log |\Sigma_{\theta\theta}| + \frac{1}{2} \log |\sigma_y^2 \mathrm{I}| + C \\
= &-\frac{1}{2\sigma_y^2} ||y - \mathrm{A}x(\hat{\theta})||_2^2 - \frac{1}{2} \hat{\theta}_h^\top \Sigma_{\theta\theta}^{-1} \hat{\theta}_h - \frac{1}{2} \log |\mathrm{A} J \Sigma_{\theta\theta} J^\top \mathrm{A}^\top + \sigma_y^2 \mathrm{I}| + C
\end{aligned} \tag{30}$$

where C captures all terms constant in $(\sigma_y^2, \ell, \sigma^2)$. Recall that $\Sigma_{yy} = \mathrm{A} J \Sigma_{\theta\theta} J^\top \mathrm{A}^\top + \sigma_y^2 \mathrm{I}$. Next we turn to the TV-PredCP prior over ℓ

$$\log p(\ell; \sigma^2) = -\sum_{d=1}^{D} \kappa_d + \log \left| \frac{\partial \kappa_d}{\partial \ell_d} \right|, \text{ with } \kappa_d := \mathbb{E}_{\mathcal{N}(\hat{\theta}_d, \Sigma_{\theta_d \theta_d})} \prod_{i=1, i \neq d}^{D} \delta(\theta_i - \hat{\theta}_i) \left[\lambda \mathrm{TV}(h(\theta)) \right].$$

Hence we obtain the following Type-II maximum a posteriori (MAP)-style objective:

$$\begin{aligned}&\log p(y, \ell; \sigma_y^2, \sigma^2) \approx \log \mathcal{N}(y; 0, \Sigma_{yy}) + \log p(\ell; \sigma^2) \\ =&\frac{1}{2}\left(-\sigma_y^{-2}||y - \mathrm{A}x(\hat{\theta})||_2^2 - \hat{\theta}_h^\top \Sigma_{\theta\theta}^{-1}\hat{\theta}_h - \log|\Sigma_{yy}|\right) - \sum_{d=1}^{D}\kappa_d + \log\left|\frac{\partial \kappa_d}{\partial \ell_d}\right| + C.\end{aligned}$$

C Additional details on our TV-PredCP

C.1 Correspondence to the formulation of Nalisnick et al. (2021)

The original formulation of the TV-PredCP (Nalisnick et al., 2021) defines a base model $q(x) = p(x|a = a_0)$ and an extended model $p(x) = p(x|a = \tau)$. The (hyper)parameter τ determines how much the predictions of the two models vary. A divergence $\mathcal{D}(p(x|a = a_0)||p(x|a = \tau))$ is placed between the two distributions and a prior placed over the divergence. This divergence is mapped back to the parameter τ using the change of variables formula. To see how our approach eq. (10) falls within this setup, take $p(x|a = \tau)$ to be $p(x) = \mathcal{N}(x; \mu, \Sigma_{xx}(\sigma^2, \ell))$, where the lengthscale ℓ takes the place of τ. The base model sets the lengthscale to be infinite, or equivalently the correlation coefficient ρ to be 1, $q(x) = \mathcal{N}(x; \mu, \Sigma_{xx}(\sigma_x^2, \infty))$. As a divergence, we choose $\mathcal{D}(p, q) = \mathbb{E}_p[\mathrm{TV}(x)] - \mathbb{E}_q[\mathrm{TV}(x)]$. We have defined our base model to be one in which all pixels are perfectly correlated and thus have the same value. This results in the expected TV for this distribution taking a value of 0. We end up with our divergence simply matching the expected TV under the extended model $\mathbb{E}_{\mathcal{N}(\mu,\Sigma_{xx})}[\mathrm{TV}(x)]$. Even when an expected TV of 0 is not attainable for any value of ℓ, as is the case when using the DIP eq. (15), there still exists a base model which will be constant with respect to our parameters of interest and can be safely ignored.

C.2 An upper bound on the expected TV

To ensure dimensionality preservation, we define our prior over ℓ in eq. (15) as a product of TV-PredCP priors, one defined for every convolutional block in the CNN, indexed by d,

$$p(\ell) = p(\ell_1)p(\ell_2)\ldots p(\ell_D) = \prod_{d=1}^{D}\pi(\kappa_d)\left|\frac{\partial \kappa_d}{\partial \ell_d}\right|, \text{with } \kappa_d := \mathbb{E}_{\mathcal{N}(\hat{\theta}_d, \Sigma_{\theta_d\theta_d})}\prod_{i=1, i\neq d}^{D}\delta(\theta_i - \hat{\theta}_i)\left[\mathrm{TV}(h(\theta))\right].$$

This formula differs from the expected TV in eq. (9), which doesn't discriminate by blocks $\kappa := \mathbb{E}_{\mathcal{N}(\hat{\theta}, \Sigma_{\theta\theta})}[\mathrm{TV}(h(\theta))]$. By the triangle inequality, $\sum_d \kappa_d$ upper bounds the expectation under $\mathcal{N}(\hat{\theta}, \Sigma_{\theta\theta})$:

$$\mathbb{E}_{\mathcal{N}(\hat{\theta}, \Sigma_{\theta\theta})}\left[\mathrm{TV}(h(\theta))\right] = \sum_{(i,j)\in\mathcal{S}}\mathbb{E}_{\mathcal{N}(\hat{\theta}, \Sigma_{\theta\theta})}\left[|(J_i\theta - J_j\theta)|\right] = \sum_{(i,j)\in\mathcal{S}}\mathbb{E}_{\mathcal{N}(\hat{\theta}, \Sigma_{\theta\theta})}\left[|\sum_d (J_{id} - J_{jd})\theta_d)|\right]$$

$$\leq \sum_{(i,j)\in\mathcal{S}}\sum_d \mathbb{E}_{\mathcal{N}(\hat{\theta}_d, \Sigma_{\theta_d\theta_d})}\left[|(J_{id} - J_{jd})\theta_d|\right] = \sum_d \mathbb{E}_{\mathcal{N}(\hat{\theta}_d, \Sigma_{\theta_d\theta_d})}\prod_{c=1, c\neq d}^{D}\delta(\theta_c - \hat{\theta}_c)\left[\sum_{(i,j)\in\mathcal{S}}|(J_i - J_j)\theta|\right] = \sum_d \kappa_d,$$

where $\mathcal{S}$ is the set of all adjacent pixel pairs. Thus, the separable form of the TV prior as a regulariser for MAP ensures that the expectated TV under the joint distribution of parameters is also regularised.

C.3 Discussing monotonicity of the TV in the prior lengthscales

In order to apply the change of variables formula in eq. (15), we require bijectivity between ℓ_d and κ_d. In the simplest setting, both variables are one-dimensional, making this constraint easier to satisfy. In fact, it suffices to show monotonicity between the two.

In practice, we use the linearised model in eq. (6) for inference. In fig. 8, we show very compelling numerical evidence for the monotonicity. We observe that κ increases in ℓ since large values for ℓ lead to an increased marginal variance σ^2 over images. After fixing the marginal variance to 1, the lengthscales have a monotonically

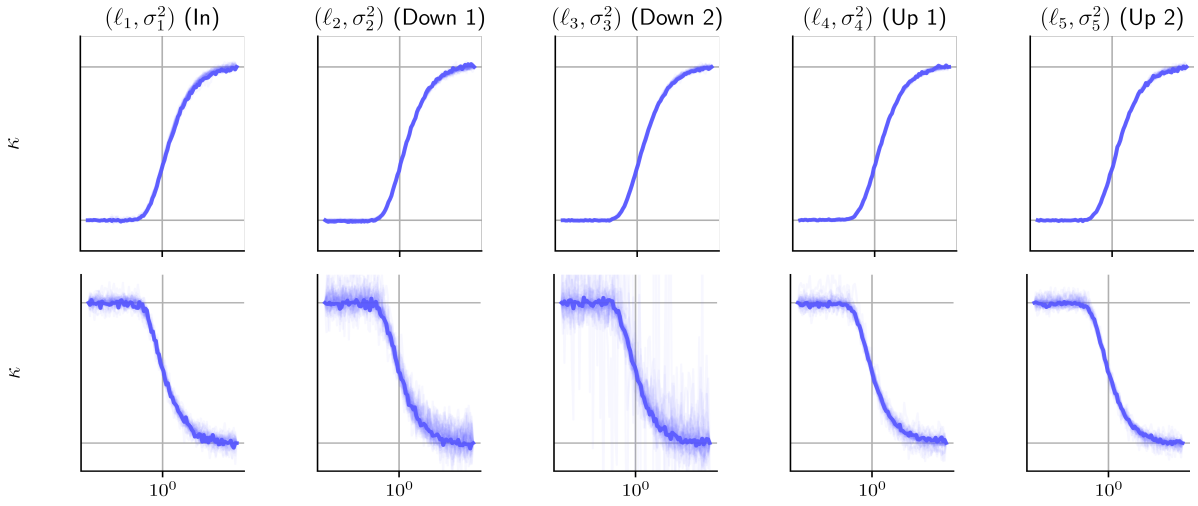

Figure 8: Experimental evidence of monotonicity computed over 50 KMNIST test images for the linearised network used in the KMNIST experiments. Horizontal axis represents lengthscale $\ell \in [0.01, 100]$. κ is estimated with 10k Monte Carlo samples. In the bottom row we fix the marginal variances of $J\Sigma_{yy}J^\top$ in image space to be 1. This allows us to observe the smoothing effect from ℓ. We use the first and last value to normalise over different KMNIST sample. The monotonicity implies the desired invertibility of the mappings ℓ and κ. We draw 500 samples to estimate k.

decreasing relationship with the expected TV. However, analytically studying the monotonicity is delicate. We investigate the issue in the linear setting to she insights (which also matches our experimental setup):

$$\kappa_d = \mathbb{E}_{\mathcal{N}(\hat{\theta}, \Sigma_{\theta\theta})\prod_{j=1, j\neq d}^{D}\delta(\theta_j - \hat{\theta}_j)}[\mathrm{TV}(h(\theta))] = \mathbb{E}_{\mathcal{N}(\hat{\theta}, \Sigma_{\theta\theta})\prod_{j=1, j\neq d}^{D}\delta(\theta_j - \hat{\theta}_j)}\Big[\sum_i |h(\theta)_i - h(\theta)_{i+1}|\Big], \tag{31}$$

assuming that the output is a 1D signal so there is only one derivative to simplify the discussion. First we derive the distribution of $h(\theta)_i - h(\theta)_{i+1}$. Note that $h(\theta)$ can be written as $h(\theta) = h_0 + J(\theta - \hat{\theta})$, by slightly abusing the notation h_0 to denote the vectors constant with respect to ℓ_d and i indices an entry of the vector $(J\theta) \in \mathbb{R}^{d_x}$. Note that the constant vector h_0 depends on the choice of the based point $\theta = 0$ (or equally plausible $\theta = \hat{\theta}$), but it does not play a role in $\mathrm{TV}(h(\theta))$, since it cancels out from the definition of $\mathrm{TV}(h(\theta))$. Then, we can rewrite it as an inner product between two vectors

$$h(\theta)_i - h(\theta)_{i+1} = (J\theta)_i - (J\theta)_{i+1} = (J_i - J_{i+1})\theta_d = v_i\theta_d,$$

where $J_i \in \mathbb{R}^{1\times d_{\theta_d}}$ denotes our NN's Jacobian for a single output pixel i (i.e. the ith row of the Jacobian matrix J, corresponding to the block parameters $\theta_d \in \mathbb{R}^{1\times d_{\theta_d}}$) and $v_i = J_i - J_{i+1} \in \mathbb{R}^{1\times d_{\theta_d}}$, $i = 1, \ldots, d_x - 1$. Now, the block parameters θ_d is distributed as $\theta_d \sim \mathcal{N}(0, \Sigma_{\theta_d\theta_d})$, in the expectation in eq. (31), whereas the remaining parameters are fixed at the mode $\hat{\theta}_j$, $j \neq d$, i.e. $\prod_{j=1, j\neq d}^{D}\delta(\theta_j - \hat{\theta}_j)$. Let $V_d \in \mathbb{R}^{(d_x-1)\times d_{\theta_d}}$ correspond to the stacking of the vectors $v_i \in \mathbb{R}^{1\times d_{\theta_d}}$, i.e. the Jacobian of the network output with respect to the weights in convolutional group d. Since the affine transformation of a Gaussian distribution remains Gaussian, $V_d\theta_d$ is distributed according to $V_d\theta_d \sim \mathcal{N}(0, V_d\Sigma_{\theta_d\theta_d}V_d^\top)$. Note that the matrix $V_d\Sigma_{\theta_d\theta_d}V_d^\top$ is not necessarily invertible, and if not, as usual, the inverse covariance should be interpreted in the sense of pseudo-inverse. Let $a =: V_d\theta_d \in \mathbb{R}^{d_x-1}$. Then

$$\kappa_d = \mathbb{E}_{a\sim\mathcal{N}(0, V_d\Sigma_{\theta_d\theta_d}V_d^\top)}\Big[\sum_i |a_i|\Big] = \sum_i \mathbb{E}_{a_i\sim\mathcal{N}(0, v_i\Sigma_{\theta_d\theta_d}v_i^\top)}[|a_i|].$$

The distribution of $|a_i|$ follows a half-normal distribution, and there holds (cf. eq. (3) of Leone et al. (1961))

$$\mathbb{E}_{a_i\sim\mathcal{N}(0, v_i\Sigma_{\theta_d\theta_d}v_i^\top)}[|a_i|] = \sqrt{\frac{2}{\pi}}(v_i\Sigma_{\theta_d\theta_d}v_i^\top)^{\frac{1}{2}}.$$

Consequently,

$$\kappa_d = \sqrt{\frac{2}{\pi}} \sum_i (v_i \Sigma_{\theta_d \theta_d} v_i^\top)^{\frac{1}{2}} \quad \text{and} \quad \frac{\partial \kappa_d}{\partial \ell_d} = \sqrt{\frac{1}{2\pi}} \sum_i (v_i \Sigma_{\theta_d \theta_d} v_i^\top)^{-\frac{1}{2}} v_i \frac{\partial}{\partial \ell_d} \Sigma_{\theta_d \theta_d} v_i^\top. \tag{32}$$

It remains to examine the monotonicity of $v_i \Sigma_{\theta_d \theta_d} v_i^\top$ in ℓ_d. Indeed, by the definition of Σ_d, we have

$$\frac{\partial}{\partial \ell_d} [\Sigma_{\theta_d \theta_d}(\ell_d)]_{j,j'} = \frac{\partial}{\partial \ell_d} \sigma_d^2 \exp\Big(-\frac{\mathrm{d}(j,j')}{\ell_d}\Big) = \frac{\sigma_d^2 \mathrm{d}(j,j')}{\ell_d^2} \exp\Big(-\frac{\mathrm{d}(j,j')}{\ell_d}\Big),$$

and thus

$$\frac{\partial}{\partial \ell_d} v_i \Sigma_{\theta_d \theta_d} v_i^\top = \frac{\sigma_d^2}{\ell_d^2} \sum_j \sum_{j'} v_{i,j} \mathrm{d}(j,j') \exp\Big(-\frac{\mathrm{d}(j,j')}{\ell_d}\Big) v_{i,j'}.$$

Then it follows that if the vectors v_i were arbitrary, the monotonicity issue would rest on the positive definiteness of the associated derivative kernel. For example, for a Gaussian kernel $e^{-\frac{(x-y)^2}{\ell_d}}$ (i.e. d is the squared Euclidean distance), the associated kernel $k(x,y)$ is given by $(x-y)^2 e^{-\frac{(x-y)^2}{\ell_d}}$. This issue seems generally challenging to verify directly, since $(x-y)^2$ is not a positive semidefinite kernel by itself on $\mathbb{R}$, even though the Gaussian kernel $e^{-\frac{(x-y)^2}{\ell_d}}$ is indeed positive semidefinite. Thus, one cannot use the standard Schur product theorem to conclude the monotonicity. Alternatively, one can also compute the Fourier transform of the kernel $k(x) = x^2 e^{-x^2}$ directly, which is given by

$$\mathcal{F}[k(x)](\omega) = \frac{2-\omega^2}{4} \frac{1}{\sqrt{2}} e^{-\frac{\omega^2}{4}}.$$

see the proposition below for the detailed derivation. Clearly, the Fourier transform of the kernel $x^2 e^{-x^2}$ is not positive over the whole real line $\mathbb{R}$. By Bochner's theorem (see e.g. p. 19 of Rudin (1990)), this kernel is actually not positive. The fact that the kernel is no longer positive definite makes the analytical analysis challenging. This observation holds also for the Matern-1/2 kernel, see the proposition below. These observations clearly indicate the risk for a potential non-monotonicity in ℓ. Nonetheless, we emphasise that this condition is only sufficient, but not necessary, since the kernel is only evaluated at lattice points (instead of arbitrary scattered points). We leave a full investigation of the monotonicity to a future work, given the compelling empirical evidence for monotonicity in both the NN and linearised settings.

Now we give Fourier transforms of the associated kernel for the Gaussian and Matern-1/2 kernels.

Proposition 1. *The Fourier transforms of the functions $x^2 e^{-x^2}$ and $|x| e^{-|x|}$ are given by*

$$\mathcal{F}[x^2 e^{-x^2}](\omega) = \frac{2-\omega^2}{4\sqrt{2}} e^{-\frac{x^2}{4}} \quad \text{and} \quad \mathcal{F}[|x| e^{-|x|}](\omega) = \frac{2(1-\omega^2)}{\sqrt{2\pi}(1+\omega^2)^2}.$$

Proof. Recall that the Fourier transform $\mathcal{F}[e^{-x^2}]$ of the Gaussian kernel e^{-x^2} is given by

$$\mathcal{F}[e^{-x^2}](\omega) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} e^{-x^2} e^{-\mathrm{i}\omega x} \mathrm{d}x = \frac{1}{\sqrt{2}} e^{-\frac{\omega^2}{4}}.$$

Direct computation shows

$$k''(x) = 4x^2 e^{-x^2} - 2e^{-x^2} = 4x^2 e^{-x^2} - 2k(x).$$

Taking Fourier transform on both sides and using the identity $\mathcal{F}[k''(x)](\omega) = -\omega^2 \mathcal{F}[f(x)](\omega)$, we obtain

$$-\omega^2 \mathcal{F}[f(x)](\omega) = 4\mathcal{F}[x^2 e^{-x^2}](\omega) - 2\mathcal{F}[f(x)](\omega),$$

which upon rearrangement gives the desired expression for $\mathcal{F}[x^2 f(x)]$. Next we compute $\mathcal{F}[|x| e^{-|x|}](\omega)$:

$$\begin{aligned} \mathcal{F}[|x| e^{-|x|}](\omega) &= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} |x| e^{-|x|} e^{-\mathrm{i}\omega x} \mathrm{d}x \\ &= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} |x| e^{-|x|} (\cos \omega x - \mathrm{i} \sin \omega x) \mathrm{d}x = \frac{2}{\sqrt{2\pi}} \int_0^{\infty} x e^{-x} \cos \omega x \mathrm{d}x, \end{aligned}$$

since $\sin \omega x$ is odd and the corresponding integral vanishes. Integration by parts twice gives

$$\begin{aligned}\int_0^\infty xe^{-x}\cos\omega x\mathrm{d}x &= -xe^{-x}\cos\omega x|_{x=0}^\infty + \int_0^\infty e^{-x}(\cos\omega x - \omega x\sin\omega x)\mathrm{d}x \\ &= \int_0^\infty e^{-x}\cos\omega x\mathrm{d}x - \int_0^\infty \omega x e^{-x}\sin\omega x\mathrm{d}x \\ &= \int_0^\infty e^{-x}\cos\omega x\mathrm{d}x + \omega x e^{-x}\sin\omega x|_{x=0}^\infty - \int_0^\infty e^{-x}(\omega\sin\omega x + \omega^2 x\cos\omega x)\mathrm{d}x.\end{aligned}$$

Rearranging the identity gives

$$\int_0^\infty xe^{-x}\cos\omega x\mathrm{d}x = \frac{1}{\omega^2+1}\int_0^\infty e^{-x}\cos\omega x\mathrm{d}x - \frac{\omega}{\omega^2+1}\int_0^\infty e^{-x}\sin\omega x\mathrm{d}x$$

This and the identities

$$\int_0^\infty e^{-x}\cos\omega x\mathrm{d}x = \frac{1}{1+\omega^2} \quad \text{and} \quad \int_0^\infty e^{-x}\sin\omega x\mathrm{d}x = \frac{\omega}{1+\omega^2},$$

immediately imply

$$\mathcal{F}[|x|e^{-|x|}](\omega) = \frac{2}{\sqrt{2\pi}}\int_0^\infty xe^{-x}\cos\omega x\mathrm{d}x = \frac{2(1-\omega^2)}{\sqrt{2\pi}(1+\omega^2)^2}.$$

This shows the second identity. □

D Additional experimental discussion

In this section, we provide additional empirical evaluation of the uncertainty estimates obtained with the linearised DIP. Validating the accuracy of the uncertainty estimates is crucial for their reliable integration into downstream tasks and computer human interaction workflows, as discussed by Antorán et al. (2021), Bhatt et al. (2021), and Barbano et al. (2021).

D.1 Evaluating approximate computations

We validate the accuracy of our approximate computation presented in section 6 on the KMNIST dataset. KMNIST is the perfect ground for this evaluation due to the fact that the low-dimensionality of d_x and d_y guarantees computational tractability of the inference problem, allowing us to benchmark the approximations we introduce in section 6, against exact computation. In this section, if not stated otherwise, we carry out our investigations with the setting where the forward operator A, comprises 20 angles, and we add 5% noise to Ax. We repeat the analysis on 10 characters taken from the test set of the KMNIST dataset. We assess the suitability of the Hutchinson trace estimator for the gradient of the log-determinant (section 6.1), and the ancestral sampling for the TV-PredCP gradients (section 6.2). Figure 9 and fig. 10 show hyperparameter optimisation $(\sigma_y^2, \sigma^2, \ell)$ using exact and estimated gradients. The hyperparameters trajectories match closely; we only observe tiny oscillations when using estimated gradients. The log-determinant gradients $\frac{\partial \log|\Sigma_{yy}|}{\partial \phi}$ are estimated using 10 samples, $v \sim \mathcal{N}(0, P)$. The PCG for solving $v^\top \Sigma_{yy}^{-1}$ uses a maximum of 50 iterations (with a early stopping criterion in place if a tolerance of 1.0 is met). We use a randomised SVD-based preconditioner P (cf. 6.1), where the rank, r, is chosen to be 200, and P is updated every 100 steps. The TV-PredCP gradients are estimated using 500 samples.

We assess the approximations introduced in section 6.3; the accuracy of the estimation of the posterior covariance matrix, but most importantly, the estimation of the test log-likelihood. For large image sizes (e.g. the Walnut cf. section 7.2), it is infeasible to store the posterior predictive covariance matrix $\Sigma_{x|y} \in \mathbb{R}^{d_x \times d_x}$, which in single precision would require 250 GB of memory. However, it can be made computationally cheaper if we consider smaller image patches of pixels, neglecting the inter-patch-dependencies. This assumes the covariance matrix $\Sigma_{x|y}$ to be block diagonal. Figure 11 shows the effect of neglecting inter-patch-dependencies. The log-likelihood increases with increasing patch-size (i.e. with more inter-dependencies being taken into

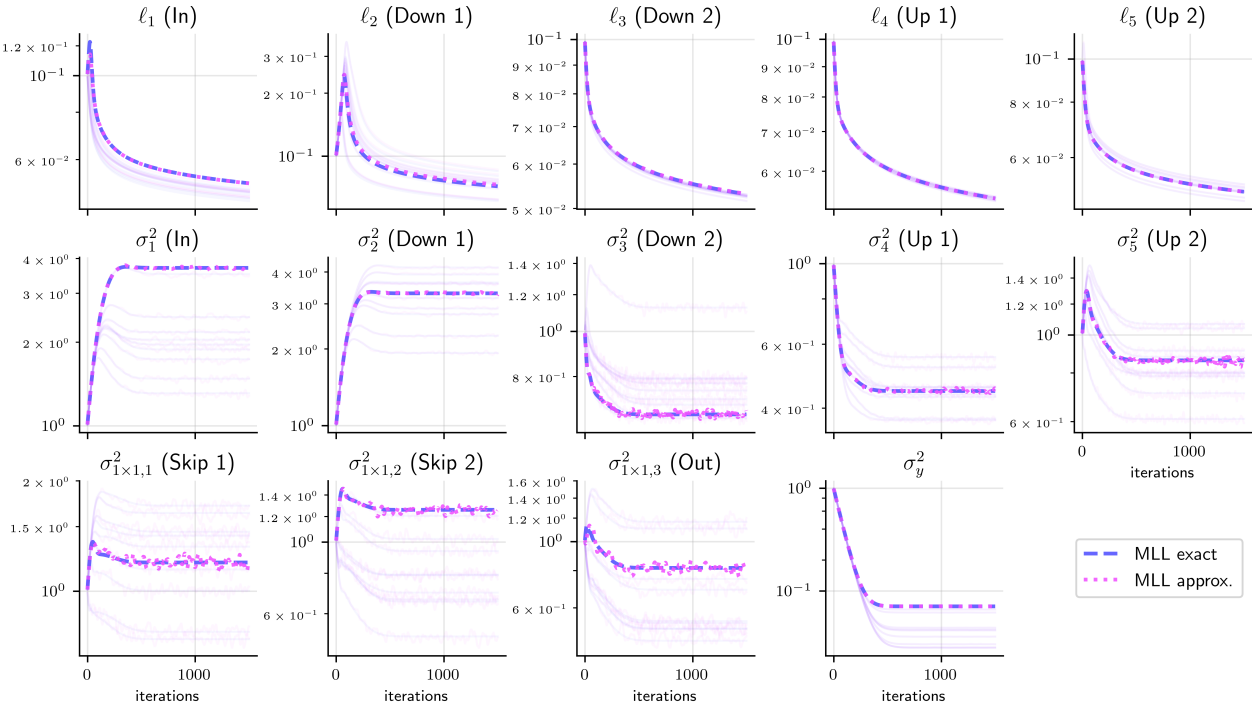

Figure 9: Hyperparameters' optimisation in eq. (17) for lin.-DIP excluding PredCP (MLL), computing exact gradients as well as resorting to the approximate numerical methods discussed in section 6.1 (i.e. PCG-based log-determinant gradients) on 10 KMNIST images.

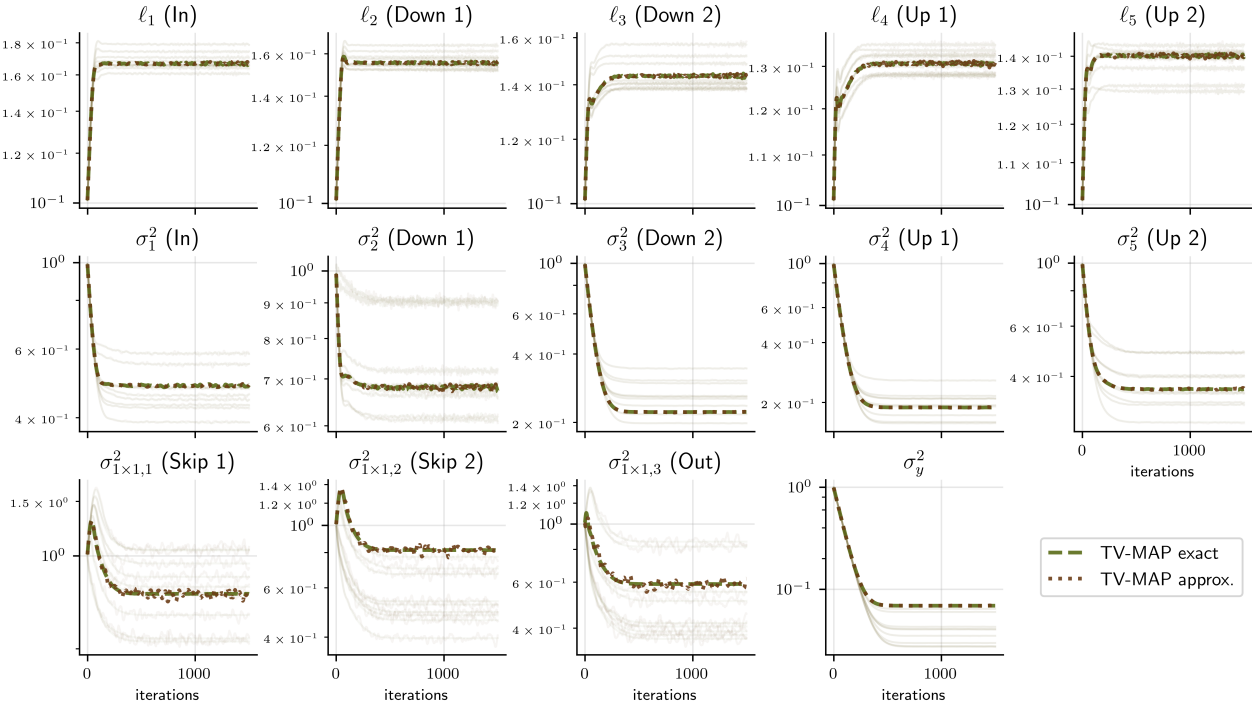

Figure 10: Hyperparameters' optimisation in eq. (17) for lin.-DIP including TV-PredCP (TV-MAP), computing exact gradients as well as resorting to the approximate numerical methods discussed in section 6.1 (i.e. PCG-based log-determinant gradients) and section 6.2 (i.e. ancestral sampling for TV-PredCP term) on 10 KMNIST images.

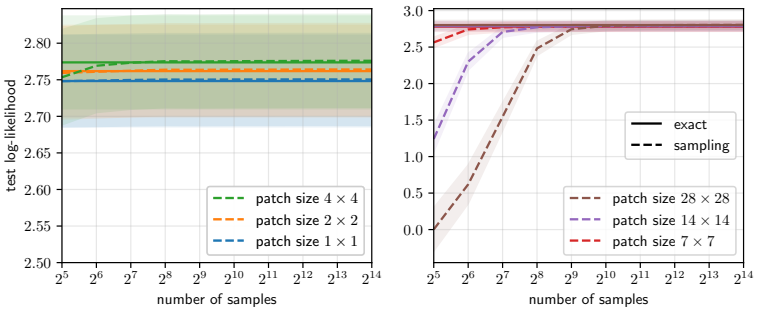

Figure 11: Test log-likelihood computed with posterior predictive covariance matrices estimated via eq. (22), and compared to the one obtained with exact methods (i.e. using exact posterior predictive covariance matrices via eq. (14)). The log-likelihood is overall well approximated. As we would expect, we observe that larger patches require more samples. We conduct our investigation over 10 KMNIST images, and show mean and standard error. $(\sigma_y^2, \sigma^2, \ell)$ are obtained using MLL.

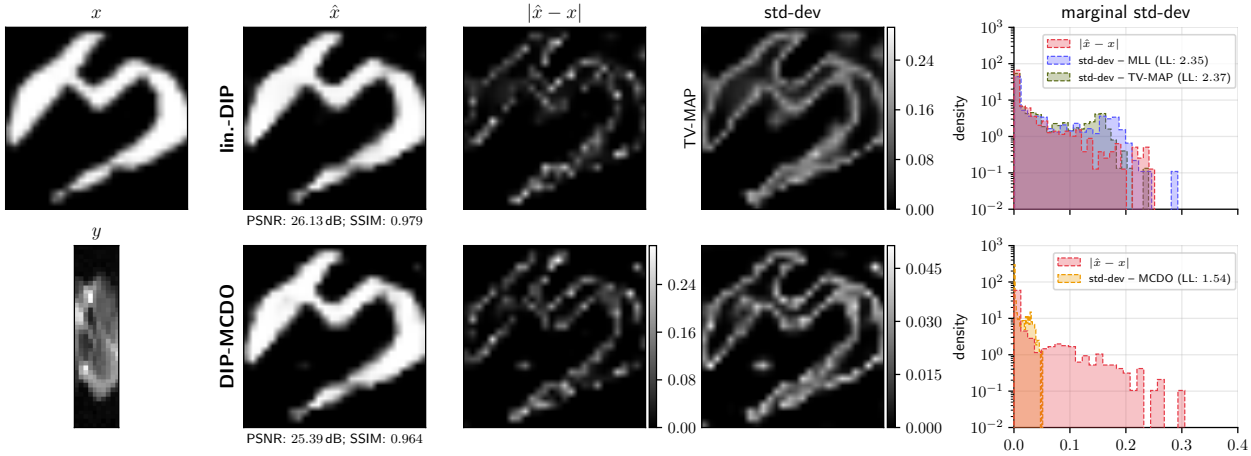

Figure 12: KMNIST character recovered from a simulated observation y (using 10 angles and $\eta(5\%)$) with lin.-DIP, DIP-MCDO and along with their uncertainty estimates and histogram plots.

account). Figure 11 shows how well the test log-likelihood is approximated when resorting to posterior predictive covariance matrices estimated via sampling using eq. (22), while sweeping across different numbers of samples and patch-sizes. As expected, estimating the log-likelihood for larger patch-sizes requires more samples. On KMNIST, 1024 samples are sufficient for almost perfect approximation of the test log-likelihood, when approximating the posterior predictive covariance matrix with patch-size of 28×28. Note that a patch-size of 28×28 on KMNIST implies that no inter-patch-dependencies are neglected.

D.2 Further discussion on KMNIST

We include additional experimental figures to support the discussion about the experiments in section 7.1.2. Figure 12, fig. 13, fig. 14, and fig. 15 are analogous to fig. 4, yet show a KMNIST character for four different problem settings: 10 angles and 20 angles, and the two noise regimes.

Figure 16 and fig. 17 show the hyperparameters' optimisation via Type-II MAP and MLL outlined in section 5.4. The use of our TV-PredCP prior leads to smaller marginal variances and larger lengthscales. This restricts our prior over reconstructions to smooth functions. The TV-PredCP introduces additional constraints into the model by encouraging the prior to contract (stronger parameter correlations and smaller posterior predictive marginal variances. In turn, this results in a more contracted posterior, which we observe as a larger Hessian determinant.

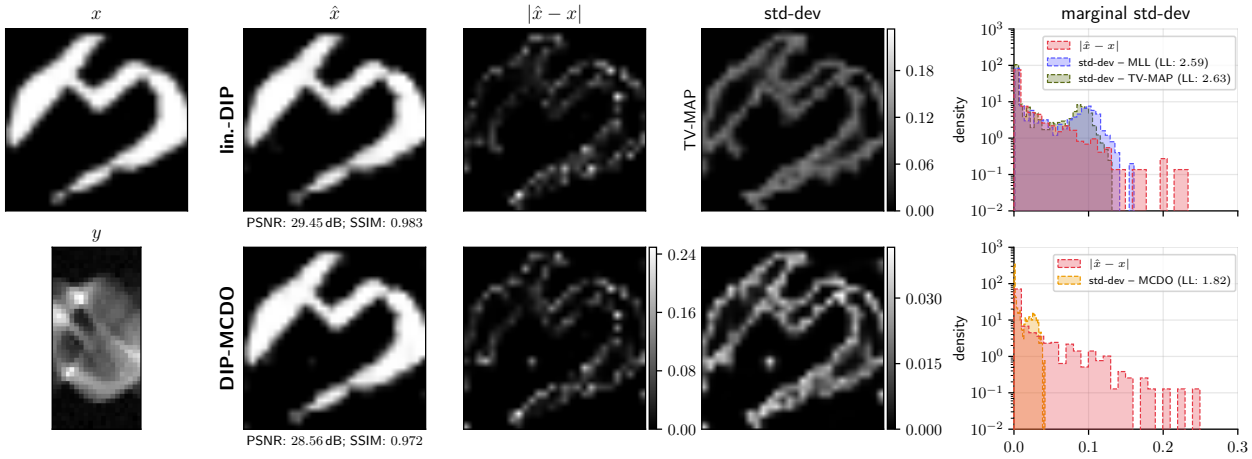

Figure 13: KMNIST character recovered from a simulated observation y (using 20 angles and $\eta(5\%)$) with lin.-DIP, DIP-MCDO along with their uncertainty estimates and histogram plots.

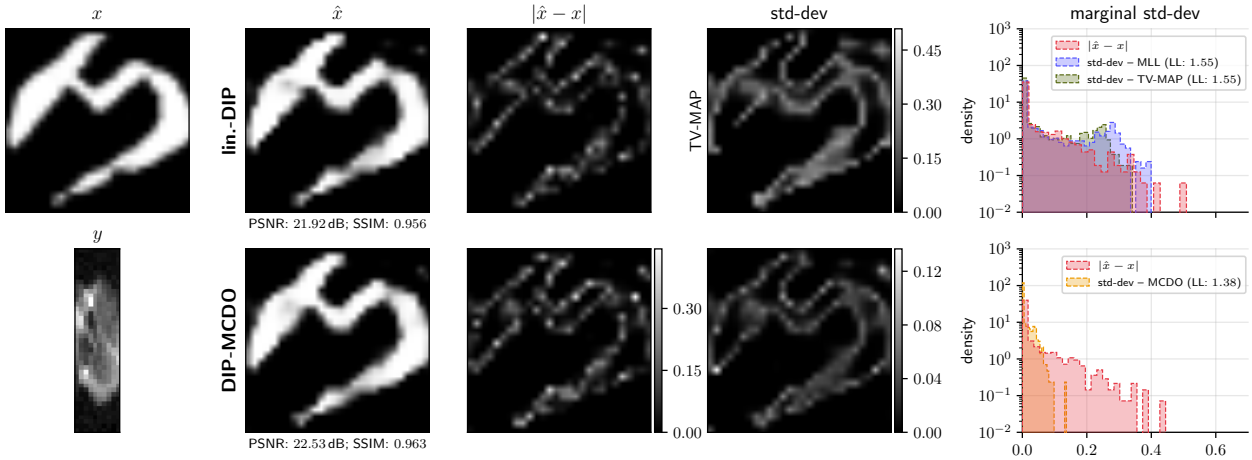

Figure 14: KMNIST character recovered from a simulated observation y (using 10 angles and $\eta(10\%)$) with lin.-DIP, DIP-MCDO along with their uncertainty estimates and histogram plots.

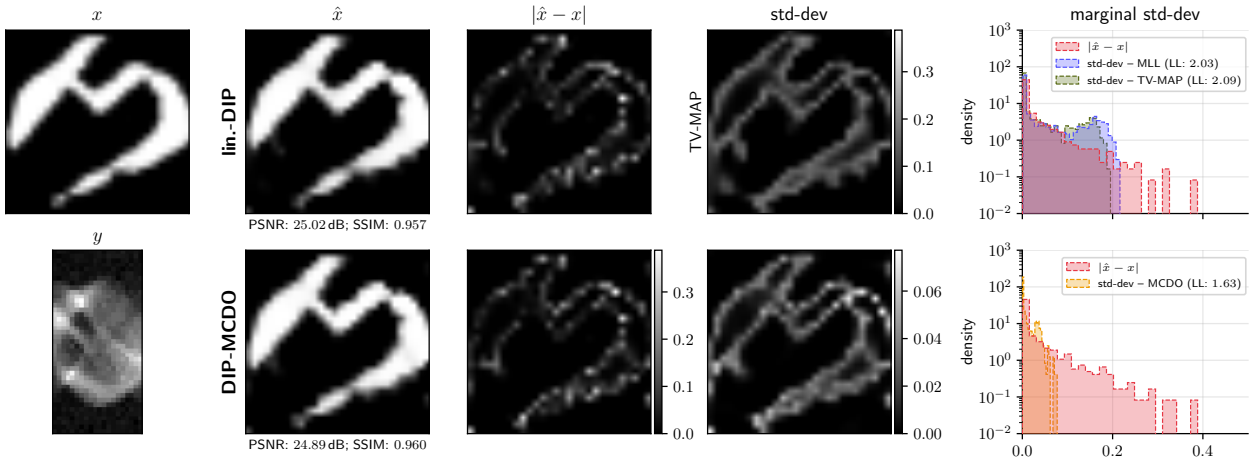

Figure 15: KMNIST character recovered from a simulated observation y (using 20 angles and $\eta(10\%)$) with lin.-DIP, DIP-MCDO along with their uncertainty estimates and calibration plots.

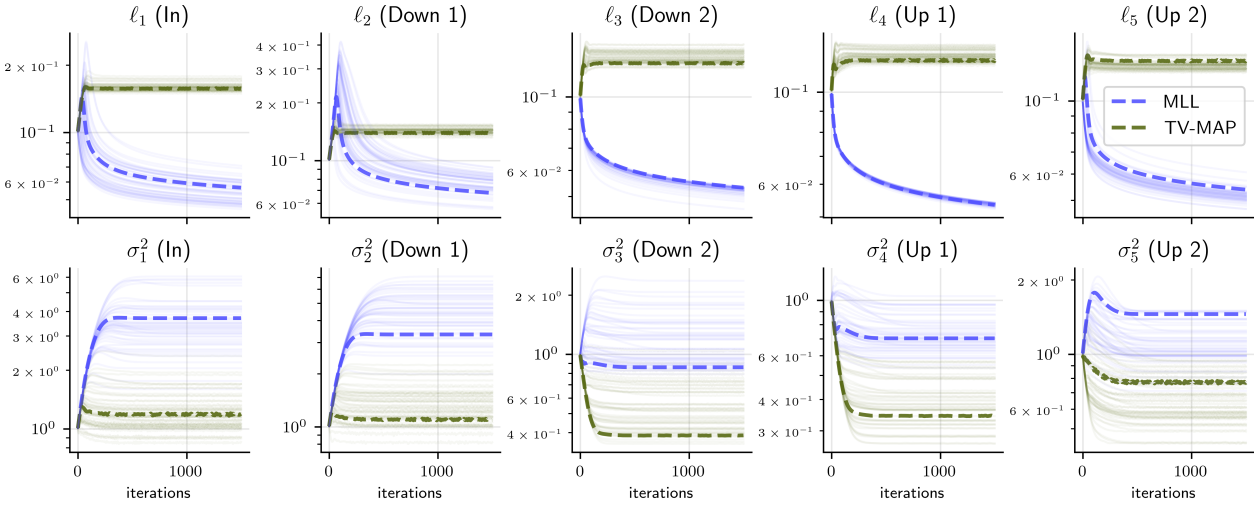

Figure 16: Optimisation of (ℓ, σ^2) via MLL and Type-II MAP for 3×3 convolution layers belonging to the small U-Net used for KMNIST. Thicker dotted lines refer to the optimisation of the exemplary reconstruction shown in fig. 4 while transparent lines correspond to other KMNIST images. The TV-PredCP leads to larger prior lengthscales ℓ and lower variances σ^2.

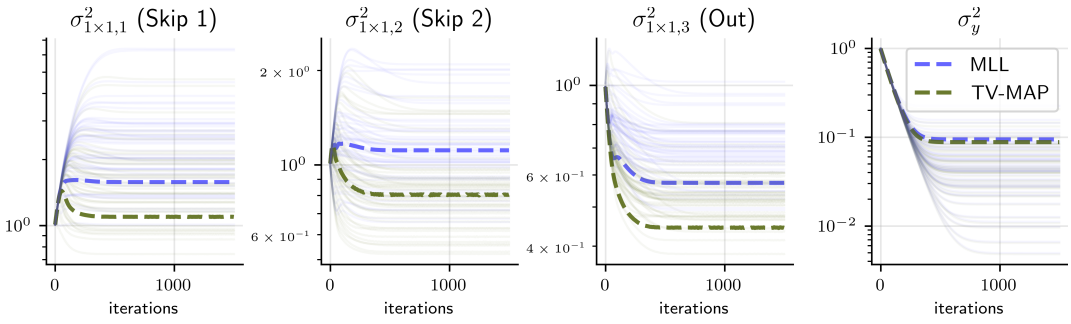

Figure 17: Hyperparameters' optimisation via MLL and Type-II MAP for 1×1 convolution layers belonging to the small U-Net used for KMNIST, along with σ_y^2. Thicker dotted lines refer to the optimisation of the KMNIST image shown in fig. 4, while transparent lines correspond to other KMNIST images.

For the KMIST dataset, one may question whether TV is an ideal regulariser. The TV regulariser enforces sparsity in the local image gradients. A TV regulariser is highly recommended when we observe sparsity in the edges present in an image, especially when the edges constitute a small fraction of the overall image pixels. That is often the case in high-resolution medical images or natural images. Intuitively, the higher the resolution of the image is, the higher the sparsity level of the edges is. However, in the KMIST dataset, due to the low resolution of the images, the edges constitute a considerable fraction of the total pixels. Therefore, a TV regulariser could be sub-optimal. In the KMNIST dataset, it is difficult to distinguish (in TV sense) what is part of the image structure and what is part of the background. The stroke is only a few pixels wide, and ground-truth pixel values are generated through interpolation (Clanuwat et al., 2018). Indeed we observe a larger gain from selecting hyperparameters using Type-II MAP (instead of MLL) for the real-measured high-resolution Walnut data than for KMNIST.

Furthermore, some KMNIST images present spurious high valued pixels away from the region containing the handwritten character. This contradicts the modelling assumption in eq. (1) which assumes x is noiseless. Our likelihood function from eq. (12) is defined over the space of observations y and thus can not account for noise in x. We translate the uncertainty induced by the observation noise to the space of images by computing the conditional log-likelihood Hessian with respect to x: $-\frac{\partial^2 \log p(y|x)}{\partial x^2} = \sigma_y^{-2} \mathrm{A}^\top \mathrm{A} \in \mathbb{R}^{d_x \times d_x}$. This matrix is of rank at most d_y, which potentially can be much smaller than d_x due to the ill-conditioning of the reconstruction problem, and therefore cannot act as a proper Gaussian precision matrix on its own. We incorporate the noise uncertainty from the observation subspace into the image space by adding the mean of the diagonal of the pseudoinverse $\sigma_y^2(\mathrm{A}^\top \mathrm{A})^\dagger$ to the marginal variances of the predictive distribution. This can also be seen as placing a Gaussian likelihood over reconstruction space, which can be marginalised to recover the predictive distribution $p(x|y) = \int \mathcal{N}(x; \hat{x}, \sigma_y^2 \mathrm{Tr}((\mathrm{A}^\top \mathrm{A})^\dagger) d_x^{-1} \mathrm{I}) \mathcal{N}(\theta; \hat{\theta}, \Sigma_{\theta|y})\, \mathrm{d}\theta = \mathcal{N}(x; \hat{x}, J \Sigma_{\theta|y} J^\top + \sigma_y^2 \mathrm{Tr}((\mathrm{A}^\top \mathrm{A})^\dagger) d_x^{-1} \mathrm{I})$.

D.3 Further discussions on Walnut data

We include additional figures to support the discussion in section 7.2. We evaluate the effect of the TV-PredCP prior for hyperparameter optimisation. We observe that this prior leads to a slightly less heavy tailed standard deviation histogram. It presents slightly better agreement with the empirical reconstruction error, resulting in a lager log-likelihood. Figure 18 and fig. 19 show the optimisation of the hyperparameters (σ_y^2, ℓ, $\boldsymbol{\sigma}_\theta^2$) using the method in section 5.4 and approximate computations in section 6. For both MLL and Type2-MAP learning, the marginal variance for all CNN blocks except the two closest to the output goes to $\approx$0. This is due to the representations from these last layer being able to explain the data well on their own. The our hyperparameter objectives are thus able to eliminate previous layers from our probabilistic model, simplifying it without sacrificing reconstruction quality. We did not observe this for KMNIST data, possibly because of our use of a smaller, less overparametrised network without any spare capacity.

E Additional experimental setup details

E.1 Setup for KMNIST experiments

We use a down-sized version of U-Net (Ronneberger et al., 2015), cf. fig. 20, as the reduced output dimension d_x and the simplicity of the problem allow us to employ a shallow architecture without compromising the reconstruction quality. This problem is computationally tractable removing the need for the approximations described in section 6. We reduce the U-Net architecture in fig. 3 to 3 scales and 32 channels at each scale, remove group-normalisation layers and use a sigmoid activation for the output. A filtered back-projection reconstruction from y is used as the network input.

Table 7 lists the hyperparameters of DIP optimisation for each setting. These values were found by grid-search on 50 KMNIST training images. The dropout rate p of DIP-MCDO is set to 0.05.

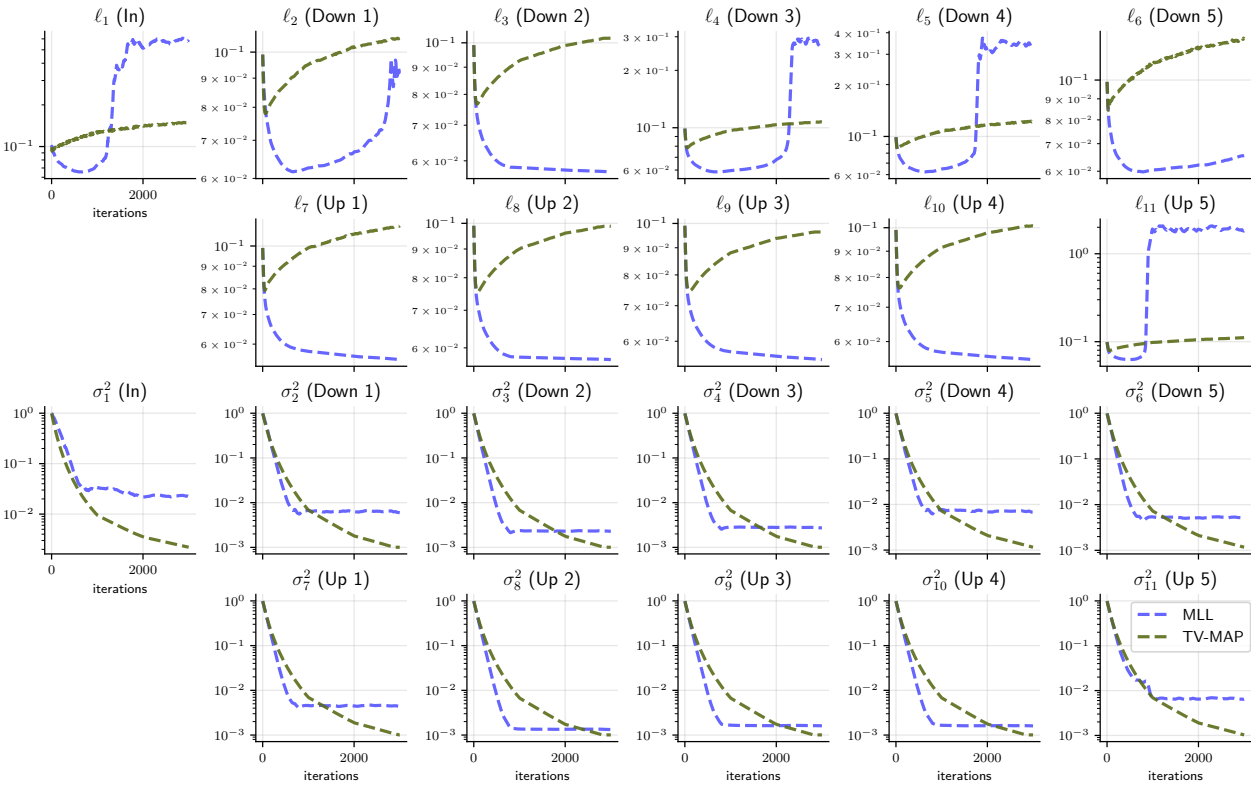

Figure 18: Optimisation of (ℓ, σ^2) via MLL and Type-II MAP for 3×3 convolution layers for the Walnut data described in section 7.2. As in fig. 16, the TV-PredCP leads to larger prior lengthscales ℓ and lower variances σ^2.

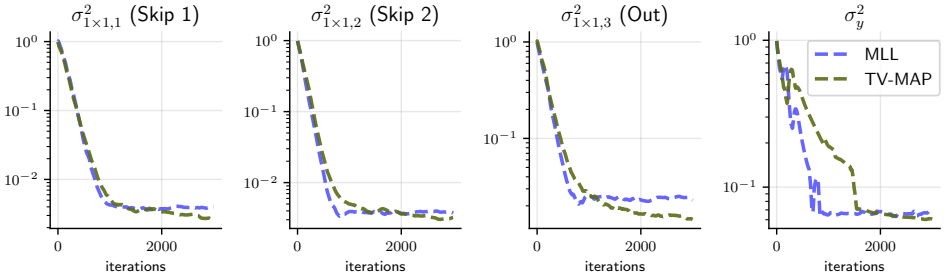

Figure 19: Optimisation of (σ_y^2, σ^2) via MLL and Type-II MAP for 1×1 convolutions.

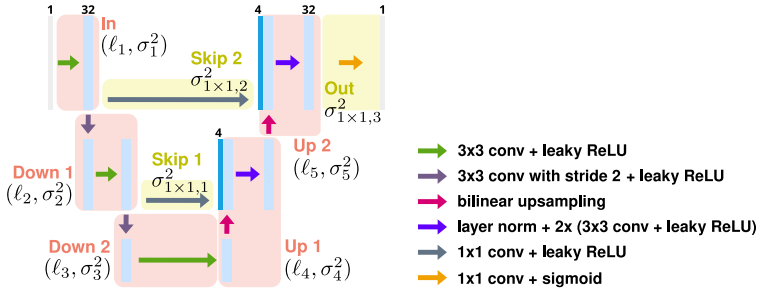

Figure 20: A schematic illustration of the reduced U-Net architecture used in the numerical experiments on KMNIST data. It has 3 scales and does not include group norm layers. Each light-blue rectangle corresponds to a multi-channel feature map. We highlight the architectural components corresponding to each block for which a separate prior is defined with red boxes.

Table 7: Hyperparameters of DIP optimisation selected using 50 randomly chosen images from the KMNIST training set. The λ values refer to our implementation of eq. (5) in which $\|\cdot\|^2$ is replaced with mean squared error (or the regularisation term is up-scaled by d_x).

	5% noise				10% noise			
#angles	5	10	20	30	5	10	20	30
TV scaling for DIP: λ	1e−5	3e−5	1e−4	1e−4	3e−5	1e−4	3e−4	3e−4
DIP iterations	14 000	29 000	41 000	50 000	7400	13 000	17 000	22 000

E.2 Computing the preconditioner for conjugate gradients

For our preconditioner P, we approximate $\mathrm{A}J\Sigma_{\theta\theta}J^\top\mathrm{A}^\top$ —for simplicity denoted as $H \in \mathbb{R}^{d_y\times d_y}$— as $\tilde{U}\tilde{\Lambda}\tilde{U}^\top$, using a randomised eigendecomposition algorithm (Halko et al., 2011; Martinsson & Tropp, 2020) with $\tilde{U} \in \mathbb{R}^{d_y\times r}$ and $r \ll d_y$. The approach first computes an orthonormal basis capturing the space spanned by H's columns. The idea is to obtain a matrix Q with r orthonormal columns, that approximates the range of H. This is done by constructing a standard normal test matrix $\Omega \in \mathbb{R}^{d_y\times r}$, and computing the (thin) QR decomposition of $H\Omega$. Once Q is computed, we solve for a symmetric matrix $B \in \mathbb{R}^{r\times r}$ (much smaller than H) such that B approximately satisfies $B(Q^\top\Omega) \approx Q^\top H\Omega$. We then compute the eigendecomposition of B, $V\Lambda V^\top$, and recover $\tilde{U} = QV$. This method requires $\mathcal{O}(r)$ matvecs resembling Hv to construct not only an approximate basis but also its complete factorisation. Finally, the preconditioner P is defined as $\tilde{U}\tilde{\Lambda}\tilde{U}^\top + \sigma_y^2\mathrm{I}$. To compute $P^{-1}v$ efficiently, we make use of the Woodbury identity.

E.3 Setup for X-ray Walnut data experiments

In (Der Sarkissian et al., 2019) projection data sets obtained with three different source positions are provided for 42 walnuts, as well as high-quality reconstructions of size 501^3 px^3 obtained via iterative reconstruction using the measurements from all three source positions. We consider the task of reconstructing a single slice of size 501^2 of the first walnut from a sub-sampled set of measurements using the second source position, which corresponds to a sparse fan-beam-like geometry. From the original 1200 projections (equally distributed over 360°) of size 972×768 we first select the appropriate detector row matching the slice position (which varies for different detector columns and angles due to a tilt in the setup), yielding measurement data of size $1200 \cdot 768$. We then sub-sample in both angle and column dimensions by factors of 20 and 6, respectively, leaving $d_y = 60 \cdot 128 = 7680$ measurements. For evaluation metrics, we take the corresponding slice from the provided high-quality reconstruction as the reference ground truth image x. The sparse operator matrix A is assembled by calling the forward projection routine of the `ASTRA` toolbox (van Aarle et al., 2015) for every standard basis vector, $\mathrm{A} = \mathrm{A}[e_1, e_2, \ldots e_{d_x}]$. While especially for large data dimensions it would be favourable to directly use the matrix-free implementations from the toolbox, we also need to evaluate the transposed operation $v_y^\top\mathrm{A}$, which would be only approximately matched by the back-projection routine (especially for the tilted 2D sub-geometry, which would require padding). Therefore, we resort to the sparse matrix multiplication via PyTorch.

The network architecture is shown in fig. 3. Following Barbano et al. (2022c), we pretrain the network to perform post-processing of filtered back-projection (FBP) reconstructions on synthetic data. The dataset consists of pairs of images containing random ellipses, and corresponding FBPs from observations simulated according to eq. (1) with 5% noise. The supervised pretraining accelerates the convergence of the subsequent unsupervised DIP reconstruction from y. In the DIP phase, the FBP of y is used as the network input. Table 8 lists the hyperparameters of DIP optimisation. The dropout rate p of DIP-MCDO is set to 0.05.

After DIP optimisation, following Antorán et al. (2022) the network weights are refined for the linearised model (eq. (6)). We optimise the same loss function as for DIP, but with the linear model eq. (6) instead of the network model, for 1000 steps. This yields network weights that fit better the subsequent MLL / Type-II MAP optimisation eq. (17), which employs the linear model.

Table 8: Hyperparameters of DIP optimisation used for the walnut data. The λ value refers to our implementation of eq. (5) in which $\|\cdot\|^2$ is replaced with mean squared error (or the regularisation term is upscaled by d_x).

TV scaling for DIP: λ	6.5e−6
DIP iterations (after pretraining)	1500

In MLL / Type-II MAP optimisation eq. (17), we use 10 probes to estimate the gradients of the log-determinant $\log|\Sigma_{yy}|$ eq. (19), employing the PCG method for solving $v^\top \Sigma_{yy}^{-1}$ using a maximum of 50 steps with a randomised SVD-based preconditioner P of rank 200 that is updated every 100 steps. The TV-PredCP gradients eqs. (20) and (21) are estimated using 20 samples. The MLL / Type-II MAP optimisation is run for 3000 iterations.

The posterior predictive covariance matrices for all methods are estimated by drawing 4096 zero-mean samples and computing empirical posterior predictive covariance matrix. The latter is done for patch-sizes from 1×1 up to 10×10 image patches. We use a stabilising heuristic for the estimated covariance matrices, inspired by Maddox et al. (2019): by letting $\tilde{\Sigma}_{x|y} \leftarrow \alpha\Sigma_{x|y} + (1-\alpha)\mathrm{diag}(\mathrm{diag}(\Sigma_{x|y}))$, $\alpha = \frac{1}{2}$, the impact of the off-diagonal entries is reduced. Note that our Gaussian assumption is correct in the case of linearised DIP but not for MCDO. However, MCDO does not provide a closed form density over the reconstructed image, only samples. The dimensionality of the reconstruction is too large for exact density estimation on real-measured data. We thus compute the log-likelihood in the same way as for the linearised DIP, i.e. via a Gaussian distribution with mean and posterior predictive covariance matrices estimated from samples. The accelerated sampling method via $\tilde{J}$ & PCG uses a randomised SVD-based 500-rank approximation $\tilde{J}$ of the Jacobian, and PCG for solving $v^\top \Sigma_{yy}^{-1}$ with a maximum of 50 steps along with a randomised SVD-based preconditioner of rank 400. This sampling variant can be performed in single precision (32 bit floating point). Thus constructing $\tilde{J}$ is actually much faster than reported in table 1 (0.5 min instead of 0.2 h).