Supplementary Material: Learning Structured Gaussians to Approximate Deep Ensembles

Ivor J.A. Simpson
University of Sussex, UK
i.simpson@sussex.ac.uk

Sara Vicente
Niantic, UK
svicente@nianticlabs.com

Neill D.F. Campbell
University of Bath, UK
n.campbell@bath.ac.uk

1. Sample and covariance videos

To demonstrate the variability we capture in our predicted covariance matrix, we provide videos containing the 8 samples from the ensembles (on repeat) and 100 samples from our approach. As can be seen, we capture similar modes of variability, but with the added benefit of being able to generate more samples.

To enable detailed illustration of the covariance between pixels, we provide videos showing the covariance between highlighted pixels, and all other pixels in the image. These videos allow for a broader view of the patterns of correlation that are learned by our model.

2. Network Architecture

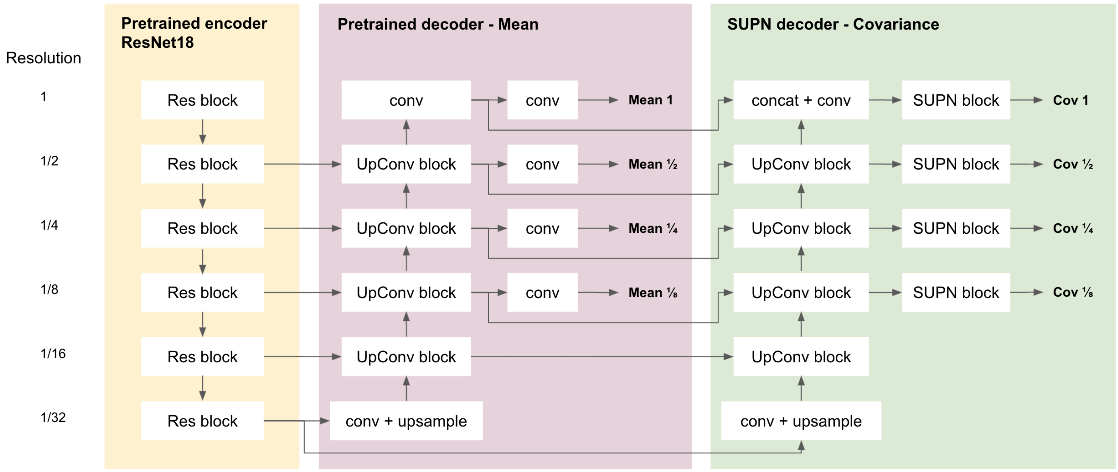

Figure 1. **Network architecture**

Illustrations of our network architecture are provided in figures 1, 2 and 3.

As shown in Fig. 3, our SUPN block predicts not only the diagonal and off-diagonal maps, but also scaling factors for those. While the scaling factors for the diagonal terms are image dependent, the scaling factors for off-diagonal elements are shared between all images.

Diagonal scaling The final diagonal value is given by: $\exp(\mathrm{D}) \times \exp(a) + \exp(b)$, where D is the log-diagonal output of the network and a and b are the diagonal multipliers.

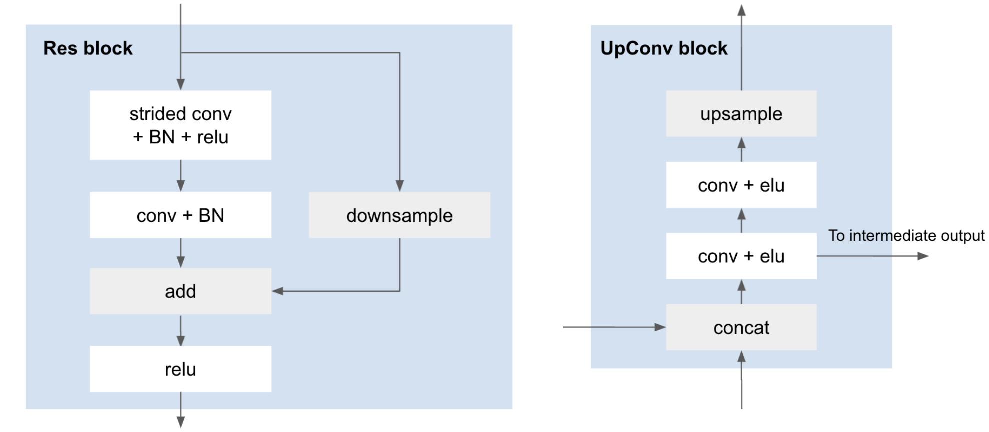

Figure 2. **Detail view of Res block and UpConv block**

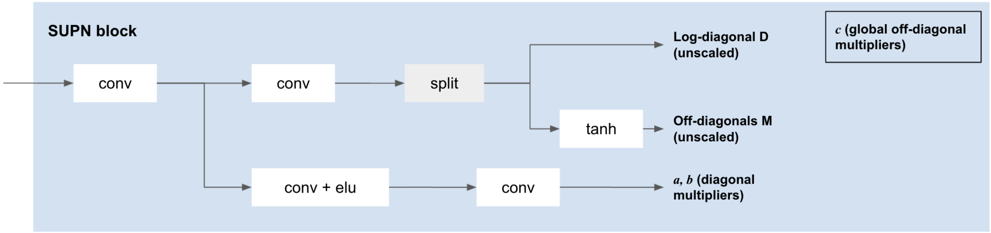
SUPN block
conv
conv
split
Log-diagonal D (unscaled)
tanh
Off-diagonals M (unscaled)
conv + elu
conv
a, b (diagonal multipliers)
c (global off-diagonal multipliers)

Figure 3. **Detail view of SUPN block**

Off-diagonal scaling The final value for the off-diagonals is given by: M × c, where M is the output of the network after the non-linearity $tanh$ and c takes a different value for each of the off-diagonal maps corresponding to a different neighbour.

3. Ablation Experiments

The following tables extend those in the main submission by removing aspects of the model architecture, or testing model variants. All SUPN variants were trained using the Monodepth2 Boot+Log ensembles from [4], except SUPN Boot+self. The various SUPN variants are given in Table 1.

Accuracy measures for different variants and baselines are given in Table 2 and uncertainty measures are given in Table 3. Key observation include:

- Removing the off-diagonal scaling mechanism described above leads to better log-likelihoods, but substantially worse samples and uncertainty metrics. We believe this is because our formulation provides an initial bias towards preferring smaller off-diagonal values, although the magnitude of these can grow this has to be sufficiently supported by an increase in likelihood.

- Using a 3×3 neighbourhood for the Cholesky of the precision matrix prediction leads to a small reduction in performance.

- The use of the additional diagonal scaling branch and concatenated pixel maps lead to reasonably small improvements in mean accuracy and uncertainty metrics, and a reduction in std-deviation for some metrics.

Table 1. SUPN variants for ablation study

Model suffix	Difference
3×3	Uses a 3×3 pixel neighbourhood rather than 5×5
- ODS	No off-diagonal scaling function
- PM	No concatenated pixel coordinate map
- DS	No extra diagonal scaling branch

Table 2. **Accuracy comparison**: Quantitative comparison of depth quality on three commonly used depth metrics. For the “Best” metrics, we sample 40 different depth map predictions for our model, and from the 8 ensembles for the baseline, and choose the best of them according to each metric. Standard deviations are given in brackets.

Model name	AbsRel Mean ↓	AbsRel Best ↓	RMSE Mean ↓	RMSE Best ↓	A1 Mean ↑	A1 Best ↑
MD2 Boot+Log	0.092 (0.035)	0.084 (0.031)	3.850 (1.370)	3.600 (1.260)	0.911 (0.064)	0.923 (0.055)
MD2 Boot+Self	**0.088** (0.034)	**0.083** (0.031)	**3.795** (1.397)	**3.574** (1.323)	**0.918** (0.060)	**0.929** (0.051)
Diagonal	0.101 (0.44)	0.103 (0.43)	4.00 (1.457)	4.02 (1.444)	0.896 (0.076)	0.894 (0.074)
SUPN 3×3	0.104 (0.047)	0.100 (0.044)	4.088 (1.510)	3.876 (1.422)	0.893 (0.079)	0.902 (0.073)
SUPN - ODS	0.107 (0.048)	0.103 (0.048)	4.148 (1.514)	3.965 (1.625)	0.887 (0.081)	0.899 (0.074)
SUPN - PM	0.104 (0.046)	0.096 (0.039)	4.077 (1.491)	3.663 (1.261)	0.892 (0.079)	0.908 (0.069)
SUPN - DS	0.104 (0.046)	0.097 (0.041)	4.072 (1.502)	3.700 (1.304)	0.892 (0.078)	0.908 (0.069)
SUPN Boot+Log	0.104 (0.047)	0.095 (0.039)	4.071 (1.489)	3.577 (1.191)	0.892 (0.080)	0.909 (0.069)
SUPN Boot+Self	0.103 (0.049)	0.096 (0.046)	4.091 (1.442)	3.800 (1.396)	0.894 (0.078)	0.906 (0.073)

Table 3. **Pixelwise uncertainty metrics**: AUSE (area under the sparsification error), lower is better. AURG (area under the random gain), higher is better. Uncertainy for SUPN estimated from std-deviation of 10 samples. Results marked with a * differ from the published work by [4], as to make it comparable we do not use the Monodepth 1 post-processing. LL (Log-Likelihood) columns provide the log-likelihood of samples from the respective ensembles under the diagonal (baseline) and SUPN models. Standard deviations are given in brackets.

Model name	AbsRel AUSE ↓	AbsRel AURG ↑	RMSE AUSE ↓	RMSE AURG ↑	A1 AUSE ↓	A1 AURG ↑	LL Boot+Log $\times 10^5$↑	LL Boot+Self $\times 10^5$↑
MD2 Boot+Log	0.038 (0.020)	0.021 (0.019)	2.449 (0.877)	0.820 (0.929)	0.046 (0.048)	0.037 (0.040)		
MD2 Boot+Self	**0.029** (0.018)	0.028 (0.019)	1.924 (1.006)	1.316 (1.000)	**0.028** (0.041)	0.049 (0.037)		
MD2 Boot+Log*	0.041 (0.019)	0.018 (0.020)	2.927 (1.327)	0.324 (1.019)	0.050 (0.049)	0.032 (0.037)		
MD2 Boot+Self*	0.040 (0.021)	0.017 (0.018)	2.906 (1.458)	0.331 (1.08)	0.045 (0.045)	0.031 (0.035)		
Diagonal	0.085 (0.050)	-0.020 (0.030)	5.075 (1.924)	-1.697 (0.799)	0.138 (0.083)	-0.440 (0.053)	1.77 (11.48)	1.15 (12.78)
SUPN 3×3	0.037 (0.028)	**0.030** (0.027)	1.922 (1.472)	1.525 (1.450)	0.041 (0.062)	0.056 (0.049)	38.12 (1.85)	35.95 (2.61)
SUPN - ODS	0.060 (0.040)	0.009 (0.027)	3.359 (1.788)	0.130 (1.249)	0.086 (0.081)	0.016 (0.053)	**41.92** (2.40)	38.28 (3.93)
SUPN - PM	0.039 (0.026)	0.028 (0.025)	1.853 (1.426)	1.583 (1.401)	0.044 (0.062)	0.053 (0.048)	40.54 (1.39)	38.18 (2.27)
SUPN - DS	0.039 (0.027)	0.027 (0.026)	1.785 (1.570)	1.648 (1.572)	0.045 (0.064)	0.053 (0.050)	40.54 (1.26)	38.14 (2.21)
SUPN Boot+Log	0.037 (0.027)	**0.030** (0.025)	**1.555** (1.307)	**1.856** (1.355)	0.040 (0.063)	**0.058** (0.047)	40.60 (1.35)	38.18 (2.93)
SUPN Boot+Self	0.050 (0.037)	0.017 (0.028)	2.786 (1.796)	0.674 (1.544)	0.062 (0.074)	0.034 (0.055)	36.51 (2.31)	**38.87** (1.63)

4. Sparsity pattern of the Cholesky decomposition

We include an illustration of the sparsity pattern of the Cholesky decomposition explained in section 2.2 of the paper.

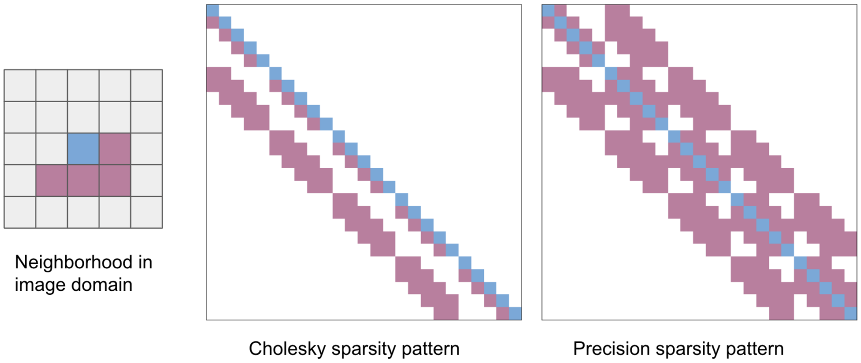

Figure 4. **Sparsity pattern for the Cholesky decomposition.** (Left) A 3×3 neighborhood around a central pixel in blue. Only neighbours in pink are considered in the Cholesky matrix, which ensures that the matrix is sparse and lower triangular. (Center) The sparsity pattern of the Cholesky matrix $\mathbf{L}_\Lambda$; only colored pixels have non-zero values. Elements in blue correspond to (positive) diagonal terms, while the elements in pink, correspond to off-diagonal values. (Right) The corresponding sparsity pattern of the precision matrix $\mathbf{\Lambda}$, where $\mathbf{\Lambda} = \mathbf{L}_\Lambda \mathbf{L}_\Lambda^\top$.

5. Training details

We use the Adam [3] optimiser with an initial learning rate of $1e^{-4}$, which we halve after the first, fifth and 15th epoch. We train for 20 epochs in total. A batch size of 16 images was used for all experiments. Our initial encoder and (mean) decoder are pre-trained models from the ensemble, we do not use samples from that model as observations during the training process.

6. Pixelwise summaries

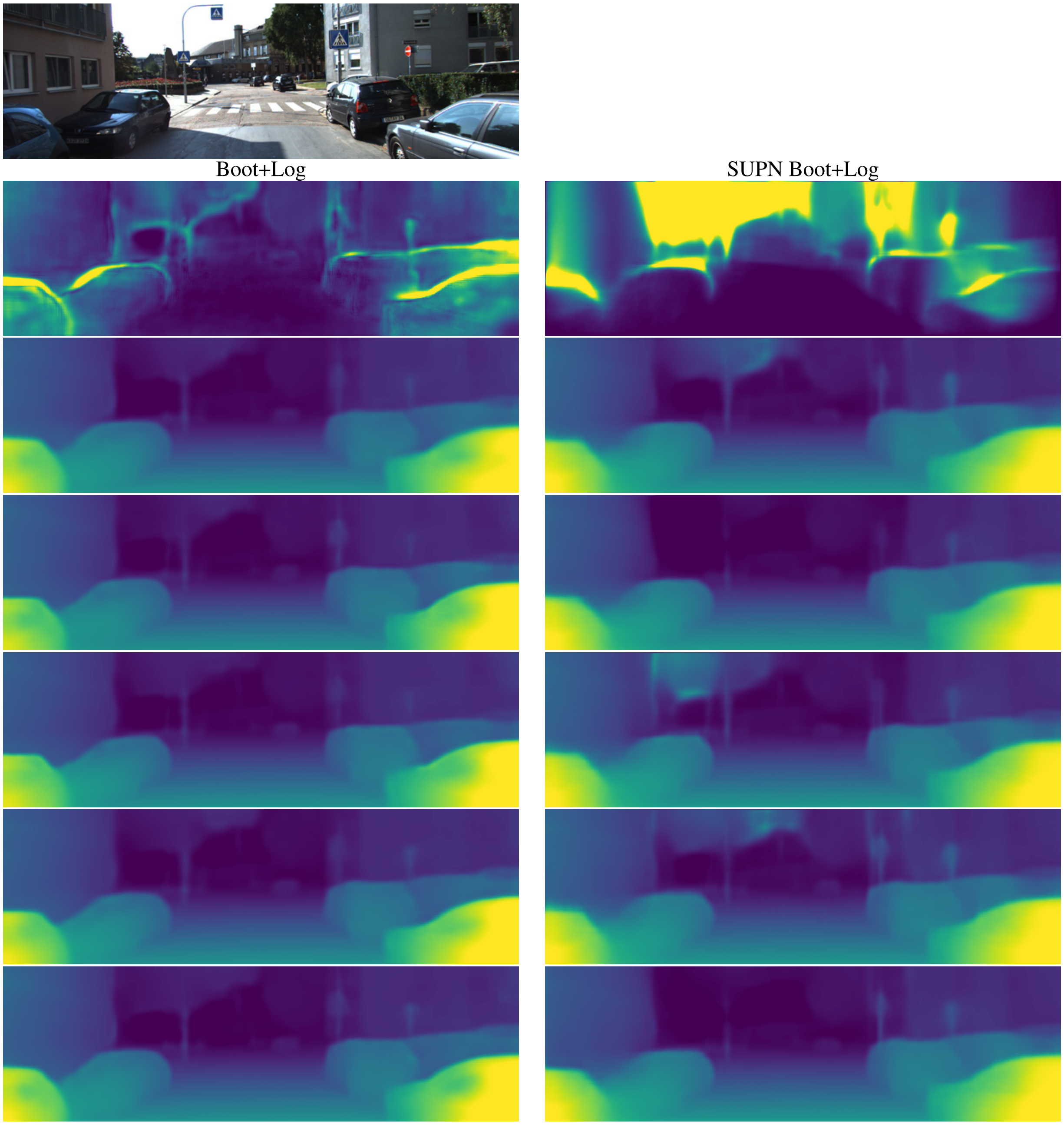

Figure 5. Visualisation of standard deviations (2nd row) and disparity samples (3rd row and below) for a given image (top row). Both of the standard deviations, and all of the samples are normalised to be in the same range. All of the samples are spatially smooth and we note that our predicted standard deviations show similar structure, although the map is smoother than that derived from the ensemble. We also observe substantially more uncertainty about the sky pixels with our approach. We speculate this is due to the inherent uncertainty in the monocular depth prediction task, which induces large variability between the ensemble models across images that the SUPN model has tried to faithfully capture.

7. Dataset and code

KITTI dataset [1] Available from `http://www.cvlibs.net/datasets/kitti/`. Available for non-commercial use only. License: Creative Commons Attribution-NonCommercial-ShareAlike 3.0

Monodepth2 repository [2] Available from `https://github.com/nianticlabs/monodepth2`. Available for non-commercial use only. License: `https://github.com/nianticlabs/monodepth2/blob/master/LICENSE`

Poggi et al repository [4] Available from: `https://github.com/mattpoggi/mono-uncertainty`. License: MIT License

Pytorch Available from: `https://pytorch.org/` License: `https://github.com/pytorch/pytorch/blob/master/LICENSE`

SuiteSparse Available from: `https://github.com/DrTimothyAldenDavis/SuiteSparse`. License: LGPL-2.1

torch-sparse-solve 0.0.5 Available from: `https://pypi.org/project/torch-sparse-solve/` License: LGPL-2.1

References

[1] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013. 7

[2] Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. Unsupervised monocular depth estimation with left-right consistency. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 270–279, 2017. 7

[3] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 5

[4] Matteo Poggi, Filippo Aleotti, Fabio Tosi, and Stefano Mattoccia. On the uncertainty of self-supervised monocular depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3227–3237, 2020. 2, 4, 7