Learning Invariant Weights in Neural Networks (Supplementary material)

Tycho F.A. van der Ouderaa[1] **Mark van der Wilk**[1]

[1]Imperial College London, UK

APPENDIX A: DETAILED DERIVATION OF VARIATIONAL INVERENCE

Applying Variational Inference (VI) [?], we maximise the marginal likelihood w.r.t. parameters $\boldsymbol{\theta} = \text{vec}(\boldsymbol{W}_2)$ by minimizing the $D_{\text{KL}}(\cdot||\cdot)$-divergence between approximate posterior $q(\boldsymbol{W}_2|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ and true posterior distribution of weights $p(\boldsymbol{W}_2|\mathcal{D})$, equivalent to maximizing the evidence lower bound (ELBO) denoted by $\mathcal{L}$:

$$
\begin{aligned}
&\arg\min_{\boldsymbol{\mu},\boldsymbol{\Sigma}} D_{\text{KL}}(q(\boldsymbol{W}_2|\boldsymbol{\mu},\boldsymbol{\Sigma})||p(\boldsymbol{W}_2|\mathcal{D})) \\
&= \arg\min_{\boldsymbol{\mu},\boldsymbol{\Sigma}} \mathbb{E}_{q(\boldsymbol{W}_2|\boldsymbol{\mu},\boldsymbol{\Sigma})}\left[\log \frac{q(\boldsymbol{W}_2|\boldsymbol{\mu},\boldsymbol{\Sigma})}{p(\boldsymbol{W}_2|\mathcal{D})}\right] \\
&= \arg\min_{\boldsymbol{\mu},\boldsymbol{\Sigma}} \mathbb{E}_{q(\boldsymbol{W}_2|\boldsymbol{\mu},\boldsymbol{\Sigma})}\left[\log \frac{q(\boldsymbol{W}_2|\boldsymbol{\mu},\boldsymbol{\Sigma})}{p(\boldsymbol{W}_2)p(\mathcal{D}|\boldsymbol{W}_2)}\right] + \log p(\mathcal{D}) \\
&= \arg\min_{\boldsymbol{\mu},\boldsymbol{\Sigma}} \mathbb{E}_{q(\boldsymbol{W}_2|\boldsymbol{\mu},\boldsymbol{\Sigma})}\left[\log \frac{q(\boldsymbol{W}_2|\boldsymbol{\mu},\boldsymbol{\Sigma})}{p(\boldsymbol{W}_2)p(\mathcal{D}|\boldsymbol{W}_2)}\right] \\
&= \arg\min_{\boldsymbol{\mu},\boldsymbol{\Sigma}} \mathbb{E}_{q(\boldsymbol{W}_2|\boldsymbol{\mu},\boldsymbol{\Sigma})}\left[\log p(\boldsymbol{W}_2|\boldsymbol{\mu},\boldsymbol{\Sigma}) - \log p(\boldsymbol{W}_2) - \log p(\mathcal{D}|\boldsymbol{W}_2)\right] \\
&= \arg\min_{\boldsymbol{\mu},\boldsymbol{\Sigma}} \mathbb{E}_{q(\boldsymbol{W}_2|\boldsymbol{\mu},\boldsymbol{\Sigma})}\left[\log p(\boldsymbol{W}_2|\boldsymbol{\mu},\boldsymbol{\Sigma}) - \log p(\boldsymbol{W}_2)\right] - \mathbb{E}_{q(\boldsymbol{W}_2|\boldsymbol{\mu},\boldsymbol{\Sigma})}\left[\log p(\mathcal{D}|\boldsymbol{W}_2)\right] \\
&= \arg\min_{\boldsymbol{\mu},\boldsymbol{\Sigma}} D_{\text{KL}}(q(\boldsymbol{W}_2|\boldsymbol{\mu},\boldsymbol{\Sigma})||p(\boldsymbol{W}_2)) + \mathbb{E}_{q(\boldsymbol{W}_2|\boldsymbol{\mu},\boldsymbol{\Sigma})}[-\log p(\mathcal{D}|\boldsymbol{W}_2)] \\
&= \arg\max_{\boldsymbol{\mu},\boldsymbol{\Sigma}} \mathbb{E}_{q(\boldsymbol{W}_2|\boldsymbol{\mu},\boldsymbol{\Sigma})}[\log p(\mathcal{D}|\boldsymbol{W}_2)] - D_{\text{KL}}(q(\boldsymbol{W}_2|\boldsymbol{\mu},\boldsymbol{\Sigma})||p(\boldsymbol{W}_2)) \\
&= \arg\max_{\boldsymbol{\mu},\boldsymbol{\Sigma}} \mathcal{L}
\end{aligned}
$$

We independently model the weight $\boldsymbol{w}_2^c$ for each class c with a full co-variance multivariate Gaussian distribution $\mathcal{N}(\boldsymbol{w}_2^c|\boldsymbol{\mu}^c, \boldsymbol{\Sigma}^c)$, parameterised by mean vector $\boldsymbol{\mu}^c$ and lower-triangular (Cholesky) decomposition of the co-variance $(\boldsymbol{L}^c)^T\boldsymbol{L}^c = \boldsymbol{\Sigma}^c$ to avoid computational issues, following **?**. We can view the variational posterior $q(\boldsymbol{W}_2|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ as multivariate Gaussian over all classes with concatenated mean and block-diagonally stacked covariances from which we sample flattened matrix $\boldsymbol{W}_2$ in one go, or -equivalently- sample row vectors $\boldsymbol{w}_2^c$ for each class and concatenate them to obtain matrix $\boldsymbol{W}_2$. By sampling L times from variational approximation $\boldsymbol{W}_2^{(1)}, \boldsymbol{W}_2^{(2)} \ldots \boldsymbol{W}_2^{(L)} \sim q(\boldsymbol{W}_2|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ we obtain a Monte Carlo estimate of $\mathbb{E}_{\boldsymbol{W}} := \mathbb{E}_{\boldsymbol{W}_2 \sim q(\boldsymbol{W}_2|\boldsymbol{\mu},\boldsymbol{\Sigma})}$ required to compute the final ELBO or negative loss $\mathcal{L}(\boldsymbol{\theta}, \mathcal{D})$:

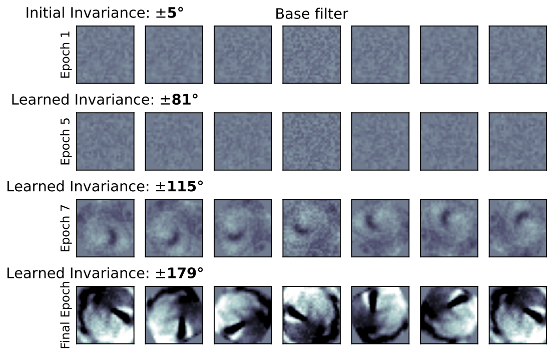
$$
\begin{aligned}
\mathcal{L}(\boldsymbol{\theta}, \mathcal{D}) &= \mathbb{E}_{q(\boldsymbol{W}_2|\boldsymbol{\mu},\boldsymbol{\Sigma})}[\log p(\mathcal{D}|\boldsymbol{W}_2)] - D_{\text{KL}}(p(\boldsymbol{W}_2|\boldsymbol{\mu},\boldsymbol{\Sigma})||p(\boldsymbol{W}_2)) \\
&= \mathbb{E}_{q(\boldsymbol{W}_2|\boldsymbol{\mu},\boldsymbol{\Sigma})}[\log p(\mathcal{D}|\boldsymbol{W}_2)] - \sum_c D_{\text{KL}}(\mathcal{N}(\boldsymbol{w}_2^c|\boldsymbol{\mu}^c,\boldsymbol{\Sigma}^c)||p(\boldsymbol{w}_2^c)) \\
&= \mathbb{E}_{q(\boldsymbol{W}_2|\boldsymbol{\mu},\boldsymbol{\Sigma}^c)}[\log p(\mathcal{D}|\boldsymbol{W}_2)] - \sum_c D_{\text{KL}}(\mathcal{N}(\boldsymbol{w}_2^c|\boldsymbol{\mu},\boldsymbol{\Sigma}^c)||\mathcal{N}(\boldsymbol{0};\boldsymbol{\Sigma}_p)) \\
&= -\overbrace{\sum_l^L \sum_i^N -\log \sigma_{y_c^{(i)}}\left(\mathbb{E}_{T\sim p_{\boldsymbol{\eta}}(T)}\left[\boldsymbol{W}_2 \circ \phi\left(\boldsymbol{W}_1 \circ T \circ \boldsymbol{x}^{(i)}\right)\right]\right)}^{\text{Regular Average Cross-entropy}} - \overbrace{\sum_c \frac{1}{2}\left[\log\frac{|\boldsymbol{\Sigma}^c|}{|\boldsymbol{\Sigma}_p|} - D + \text{tr}\left\{\boldsymbol{\Sigma}_p\boldsymbol{\Sigma}^c\right\} + \boldsymbol{\mu}^T\boldsymbol{\Sigma}_p^{-1}\boldsymbol{\mu}\right]}^{\text{Closed-form KL Regularizer}}
\end{aligned}
$$

for every input $\boldsymbol{x}^{(i)}$, log soft-argmax output σ_{y_c} for class of corresponding label $y_c^{(i)}$, fixed first layer weights $\boldsymbol{W}_1$, prior weights $\boldsymbol{\Sigma}_p = \boldsymbol{I}\alpha$, input dimensionality D, and trace tr$(\cdot)$. To allow for mini-batching, we use the Stochastic Variational Bayes Estimate (SGVB) from **?** of the ELBO or negative loss $\tilde{\mathcal{L}}(\boldsymbol{\theta}, \mathcal{D})$:

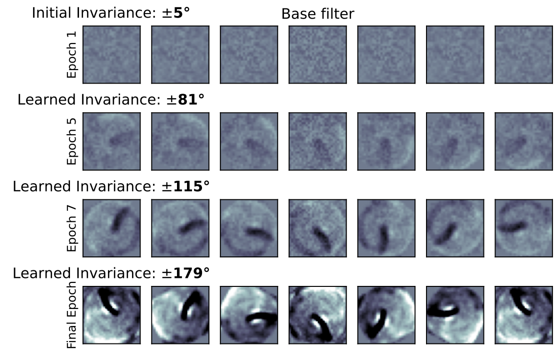
$$
\tilde{\mathcal{L}}(\boldsymbol{\theta}, \mathcal{D}) = -N\overbrace{\frac{1}{M}\sum_l^L \sum_i^M -\log \sigma_{y_c^{(i)}}\left(\mathbb{E}_{T\sim p_{\boldsymbol{\eta}}(T)}\left[\boldsymbol{W}_2 \circ \phi\left(\boldsymbol{W}_1 \circ T \circ \boldsymbol{x}^{(i)}\right)\right]\right)}^{\text{Regular Batch Averaged Cross-entropy}} - \overbrace{\sum_c \frac{1}{2}\left[\log\frac{|\boldsymbol{\Sigma}^c|}{|\boldsymbol{\Sigma}_p|} - D + \text{tr}\left\{\boldsymbol{\Sigma}_p\boldsymbol{\Sigma}^c\right\} + \boldsymbol{\mu}^T\boldsymbol{\Sigma}_p^{-1}\boldsymbol{\mu}\right]}^{\text{Closed-form KL Regularizer}}
$$

where we can choose $L = 1$ if we use a sufficiently large batch size.

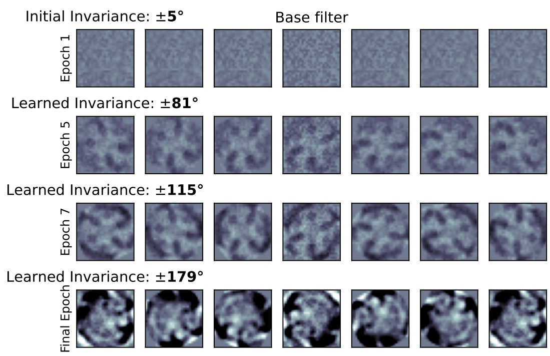
APPENDIX B: WEIGHT VISUALIZATIONS OF LEARNED ROTATIONAL INVARIANCE



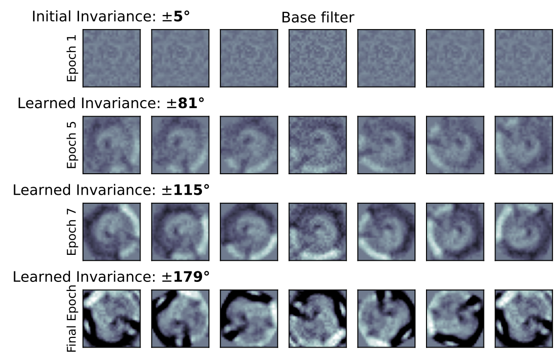
(a) Feature bank #1 over training iterations



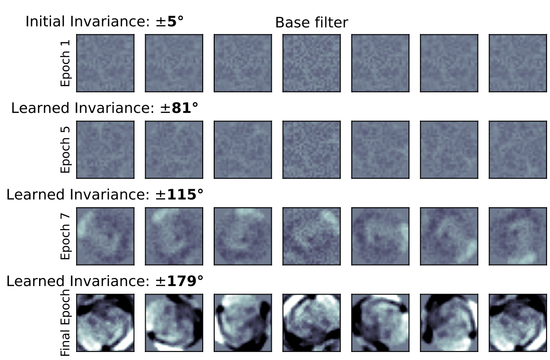
(b) Feature bank #2 over training iterations



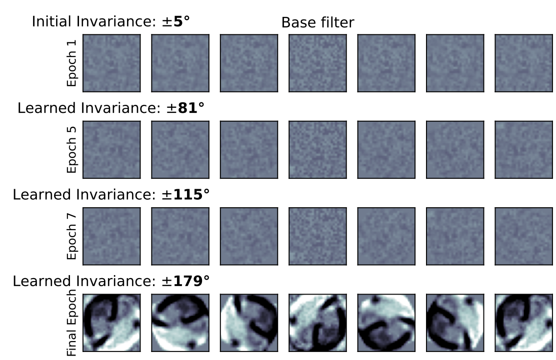
(c) Feature bank #3 over training iterations



(d) Feature bank #4 over training iterations



(e) Feature bank #5 over training iterations



(f) Feature bank #6 over training iterations

Figure 1: Illustration of the features banks over training iterations. Features are randomly initialised with almost no rotational invariance and converge to particular filters with full rotational invariance when trained on fully rotated MNIST data.

APPENDIX C.1: ROTATIONAL INVARIANCE IN RFF NEURAL NETWORK

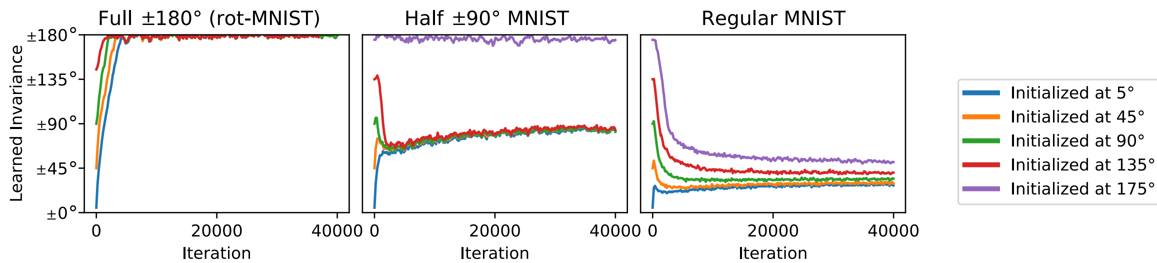


Figure 2: Predicted invariance over training iterations for different initial invariances for RFF neural network.

Model	Test Accuracy Fully rotated MNIST	 Partially rotated MNIST	 Regular MNIST	ELBO Fully rotated MNIST	 Partially rotated MNIST	 Regular MNIST
Fixed 5°	79.29	86.71	**96.00**	-1.07	-0.80	-0.36
Fixed 45°	87.35	91.13	95.93	-0.63	-0.49	**-0.26**
Fixed 90°	90.33	**91.69**	94.69	-0.52	**-0.44**	-0.30
Fixed 135°	91.19	91.04	92.13	-0.45	-0.45	-0.36
Fixed 175°	**91.57**	90.47	90.97	**-0.43**	-0.47	-0.45
Learned (5° Init)	**91.72**	**92.34**	**96.40**	**-0.43**	**-0.42**	**-0.26**
Learned (45° Init)	**91.65**	**92.31**	**96.42**	**-0.43**	**-0.42**	**-0.26**
Learned (90° Init)	**91.65**	**92.37**	**96.40**	**-0.43**	**-0.42**	**-0.26**
Learned (135° Init)	**91.66**	**92.37**	**96.10**	**-0.43**	**-0.42**	**-0.26**
Learned (175° Init)	**91.68**	**91.69**	95.64	**-0.43**	**-0.43**	**-0.26**

Table 1: Table containing Test Accuracy and ELBO scores after training for experiments with RFF network. In bold: the best scores for fixed invariance and, for learned invariances, all scores that surpass the best score using fixed invariance.

APPENDIX C.2: ROTATIONAL INVARIANCE IN RELU NEURAL NETWORK

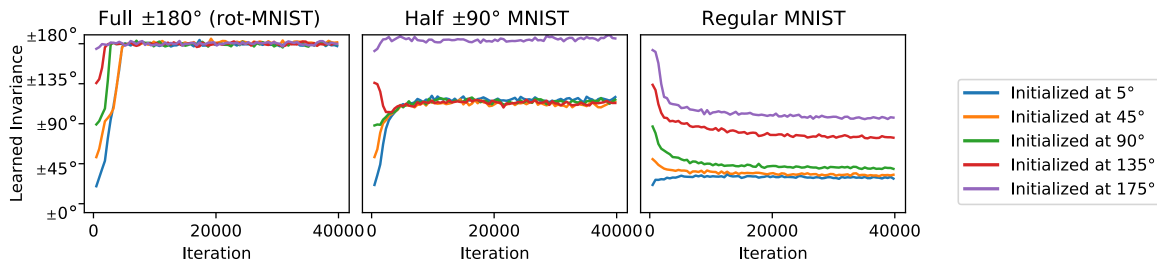


Figure 3: Predicted invariance over training iterations for different initial invariances of ReLU neural network with both input and output layer weights trained.

	Test Accuracy			ELBO		
Model	Fully rotated MNIST	Partially rotated MNIST	Regular MNIST	Fully rotated MNIST	Partially rotated MNIST	Regular MNIST
Fixed 5°	87.21	90.68	96.76	-0.28	-0.20	**-0.02**
Fixed 45°	95.24	96.46	98.13	-0.09	**-0.06**	-0.02
Fixed 90°	96.50	97.11	**98.14**	-0.07	-0.06	-0.03
Fixed 135°	97.15	**97.31**	97.79	**-0.06**	-0.06	-0.04
Fixed 175°	**97.53**	97.30	97.15	-0.07	-0.06	-0.06
Learned (0° Init)	97.34	97.13	**98.40**	-0.07	**-0.06**	**-0.02**
Learned (45° Init)	97.23	**97.36**	**98.27**	-0.07	**-0.05**	**-0.02**
Learned (90° Init)	97.28	**97.22**	**98.19**	-0.07	**-0.06**	**-0.02**
Learned (135° Init)	97.45	**97.29**	**98.33**	**-0.06**	**-0.05**	**-0.02**
Learned (175° Init)	97.23	**97.23**	**98.03**	**-0.06**	**-0.06**	**-0.03**

Table 2: Table containing Test Accuracy and ELBO scores after training for experiments of ReLU neural network with both input and output layer weights trained. In bold: the best scores for fixed invariance and, for learned invariances, all scores that surpass the best score using fixed invariance.

APPENDIX C.3: DIFFERENT TRANSFORMATIONS IN RFF NETWORK

Model	Test Accuracy				ELBO			
	Fully rotated MNIST	Translated MNIST	Scaled MNIST	Regular MNIST	Fully rotated MNIST	Translated MNIST	Scaled MNIST	Regular MNIST
Regular MLP	79.29	66.07	89.25	95.16	-1.14	-1.49	-0.69	-0.39
+ Rotation	**92.59**	75.06	88.66	96.59	**-0.43**	-1.08	-0.62	-0.26
+ Translation	83.66	**87.81**	86.15	96.78	-0.82	**-0.64**	-0.72	-0.24
+ Scale	82.77	75.48	**91.31**	96.52	-0.84	-1.08	**-0.49**	-0.26
+ Affine	**92.64**	**87.77**	**90.58**	**97.38**	**-0.43**	**-0.64**	**-0.54**	**-0.21**

Table 3: Test Accuracy and ELBO for learned invariance using different transformations in a shallow RFF neural network.

APPENDIX C.4: DIFFERENT TRANSFORMATION IN RELU NETWORK

Model	Test Accuracy				ELBO			
	Fully rotated MNIST	Translated MNIST	Scaled MNIST	Regular MNIST	Fully rotated MNIST	Translated MNIST	Scaled MNIST	Regular MNIST
Regular MLP	90.35	89.34	96.61	98.10	-0.06	-0.06	-0.03	-0.02
+ Rotation	**98.05**	94.08	97.62	98.64	**-0.05**	-0.06	-0.03	-0.02
+ Translation	93.59	**97.87**	97.98	98.76	-0.09	-0.06	-0.03	-0.02
+ Scale	93.80	94.30	**98.06**	98.35	-0.06	-0.06	-0.03	-0.02
+ Affine	**98.14**	**97.66**	**98.31**	**98.93**	**-0.05**	-0.06	-0.03	-0.02

Table 4: Test Accuracy and ELBO for learned invariance using different transformations in a shallow ReLU neural network.

APPENDIX C.4: DIFFERENT TRANSFORMATION IN RELU NETWORK ON DATASETS WITH COMBINATIONS OF TWO INVARIANCES.

Model	Test Accuracy				ELBO			
	Fully rotated + Translated MNIST	Fully rotated + Scaled MNIST	Translated + Scaled MNIST	Regular MNIST	Fully rotated + Translated MNIST	Fully rotated + Scaled MNIST	Translated + Scaled MNIST	Regular MNIST
Regular MLP	53.36	80.71	75.50	98.10	**-0.26**	**-0.10**	**-0.12**	**-0.02**
+ Rotation	**85.35**	**95.66**	85.42	98.64	-0.31	-0.10	-0.27	-0.02
+ Translation	**83.84**	83.40	**91.77**	98.76	-0.42	-0.16	-0.19	-0.02
+ Scale	55.63	**89.81**	**86.04**	98.35	-0.39	-0.12	-0.17	-0.02
+ Affine	**89.37**	**95.88**	**91.95**	**98.93**	-0.37	-0.09	-0.18	-0.02

Table 5: Test Accuracy and ELBO for learned invariance using different transformations in a shallow ReLU neural network on datasets augmented by two subsequent transformations (rotation+translation, rotation+scaling and translation+scaling). Surprisingly, the regular MLP ends up with the best ELBO in this experiment. We did not consistently observe the best ELBO for the regular MLP throughout optimization, and find that we can still use our method and the ELBO to learn invariances in this case. Again, we observe that models with learned invariances achieve the highest test accuracy.

APPENDIX D: DATASET DETAILS

All datasets have 60000 training examples and 10000 test examples and are created by taking regular MNIST or CIFAR-10 and applying random transformations:

Regular MNIST Dataset: MNIST handwritten digit database [**?**].
Regular CIFAR-10 Dataset: CIFAR-10 dataset with 10 classes [**?**].
Partially rotated dataset: Every sample rotated by radian angle θ, sampled from $\theta \sim U[-\frac{\pi}{2}, \frac{\pi}{2}]$.
Fully rotated dataset: Every sample rotated by radian angle θ, sampled from $\theta \sim U[-\pi, \pi]$.
Translated dataset: Translated samples relatively by dx and dy pixels, sampled from $dx, dy \sim U[-8, 8]$.
Scaled dataset: Every sample scaled around center with $\exp(s)$, sampled from $s \sim U[-\log(2), \log(2)]$.

APPENDIX E: LIE GROUP GENERATORS

We follow **?** and, similarly, utilise six matrix generators:

$$\boldsymbol{G}_{\text{transx}} = \boldsymbol{G}_1 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \qquad \boldsymbol{G}_{\text{transy}} = \boldsymbol{G}_2 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}, \qquad \boldsymbol{G}_{\text{rot}} = \boldsymbol{G}_3 = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$\boldsymbol{G}_{\text{scalex}} = \boldsymbol{G}_4 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \qquad \boldsymbol{G}_{\text{scaley}} = \boldsymbol{G}_5 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \qquad \boldsymbol{G}_{\text{shear}} = \boldsymbol{G}_6 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

To parameterise affine transformations we compute the following matrix exponential [**?**]:

$$T_{\boldsymbol{\epsilon}} = \exp\left(\sum_i \epsilon_i \eta_i \boldsymbol{G}_i\right), \qquad \boldsymbol{\epsilon} \sim U[-1, 1]^k \tag{1}$$

Optionally, the values of $\boldsymbol{\eta}$ can be constrained to a positive range by passing them through a 'softplus'-function, or in case of $\eta_3 = \eta_{\text{rot}}$ to $[-\pi, \pi]$ using a scaled 'tanh' function, preventing double coverage on the unit circle. In practice, however, we did not find such constraints necessary as long as η_{rot} is reasonably initialised (e.g. $\boldsymbol{\eta} = \boldsymbol{0}$).

By fixing certain η_i at 0, subsets of the generator matrices parameterise rotation, translation and scaling:

For rotation only:
Learn η_3.
Fix $\eta_i = 0$ for all $i \neq 3$.

$$\begin{aligned} T_{\boldsymbol{\epsilon}}^{(\text{rot})} &= \exp\left(\sum_i \epsilon_i \eta_i \boldsymbol{G}_i\right) \\ &= \exp\left(\epsilon_3 \eta_3 \boldsymbol{G}_3\right) \\ &= \exp\left(\begin{bmatrix} 0 & -\epsilon_3\eta_3 & 0 \\ \epsilon_3\eta_3 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}\right) \\ &= \begin{bmatrix} \cos(\epsilon_3\eta_3) & -\sin(\epsilon_3\eta_3) & 0 \\ \sin(\epsilon_3\eta_3) & \cos(\epsilon_3\eta_3) & 0 \\ 0 & 0 & 1 \end{bmatrix} \end{aligned}$$

For translation only:
Learn η_1 and η_2.
Fix $\eta_i = 0$ for all $i > 2$.

$$\begin{aligned} T_{\boldsymbol{\epsilon}}^{(\text{trans})} &= \exp\left(\sum_i \epsilon_i \eta_i \boldsymbol{G}_i\right) \\ &= \exp\left(\epsilon_1 \eta_1 \boldsymbol{G}_1 + \epsilon_2 \eta_2 \boldsymbol{G}_2\right) \\ &= \exp\left(\begin{bmatrix} 0 & 0 & \eta_1 \\ 0 & 0 & \eta_2 \\ 0 & 0 & 0 \end{bmatrix}\right) \\ &= \begin{bmatrix} 1 & 0 & \epsilon_1\eta_1 \\ 0 & 1 & \epsilon_2\eta_2 \\ 0 & 0 & 1 \end{bmatrix} \end{aligned}$$

For scaling only:
Learn η_4 and η_5.
Fix $\eta_i = 0$ for all $i \notin \{4, 5\}$.

$$\begin{aligned} T_{\boldsymbol{\epsilon}}^{(\text{scale})} &= \exp\left(\sum_i \epsilon_i \eta_i \boldsymbol{G}_i\right) \\ &= \exp\left(\epsilon_4 \eta_4 \boldsymbol{G}_4 + \epsilon_5 \eta_5 \boldsymbol{G}_5\right) \\ &= \exp\left(\begin{bmatrix} \eta_4 & 0 & 0 \\ 0 & \eta_5 & 0 \\ 0 & 0 & 0 \end{bmatrix}\right) \\ &= \begin{bmatrix} \exp(\epsilon_4\eta_4) & 0 & 0 \\ 0 & \exp(\epsilon_5\eta_5) & 0 \\ 0 & 0 & 1 \end{bmatrix} \end{aligned}$$