Principled Weight Initialisation for Input-Convex Neural Networks

Pieter-Jan Hoedt & Günter Klambauer
LIT AI Lab & ELLIS Unit Linz
Institute for Machine Learning
Johannes Kepler University, Linz, Austria
{hoedt, klambauer}@ml.jku.at

Abstract

Input-Convex Neural Networks (ICNNs) are networks that guarantee convexity in their input-output mapping. These networks have been successfully applied for energy-based modelling, optimal transport problems and learning invariances. The convexity of ICNNs is achieved by using non-decreasing convex activation functions and non-negative weights. Because of these peculiarities, previous initialisation strategies, which implicitly assume centred weights, are not effective for ICNNs. By studying signal propagation through layers with non-negative weights, we are able to derive a principled weight initialisation for ICNNs. Concretely, we generalise signal propagation theory by removing the assumption that weights are sampled from a centred distribution. In a set of experiments, we demonstrate that our principled initialisation effectively accelerates learning in ICNNs and leads to better generalisation. Moreover, we find that, in contrast to common belief, ICNNs can be trained without skip-connections when initialised correctly. Finally, we apply ICNNs to a real-world drug discovery task and show that they allow for more effective molecular latent space exploration.

1 Introduction

Input-Convex Networks. Input-Convex Neural Networks (ICNNs) are networks for which each output neuron is convex with respect to the inputs. The convexity is a result of using non-decreasing convex activation functions and weight matrices with non-negative entries (Amos et al., 2017). ICNNs were originally introduced in the context of energy modelling (Amos et al., 2017). Also in other contexts, ICNNs have proven to be useful. E.g. Sivaprasad et al. (2021) show that ICNNs can be used as regular classification models, Makkuva et al. (2020) rely on the convexity to model optimal transport mappings, and Nesterov et al. (2022) illustrate how ICNNs can be used to learn invariances by simplifying the search for level sets — i.e. inputs for which the output prediction remains unchanged. ICNNs have also been used in model predictive control, where recently an input-convex variant of LSTMs was proposed to reduce convergence time while ensuring closed-loop stability (Wang and Wu, 2023). Despite their successful application for various tasks, convergence can be notably slow (Sivaprasad et al., 2021, Fig. 1 (d)). We hypothesize that this slow training is the result of poor initialisation of the positive weights in ICNNs. This poor initialisation leads to distribution shifts that make learning harder, as illustrated in Figure 1. Therefore, we propose a principled initialisation strategy for layers with non-negative weight matrices (cf. Chang et al., 2020).

Importance of initialisation strategies. Initialisation strategies have enabled faster and more stable learning in deep networks (LeCun et al., 1998; Glorot and Bengio, 2010; He et al., 2015). The goal of a good initialisation strategy is to produce similar statistics in every layer. This can be done in the forward pass (LeCun et al., 1998; Mishkin and Matas, 2016; Klambauer et al., 2017; Chang et al.,

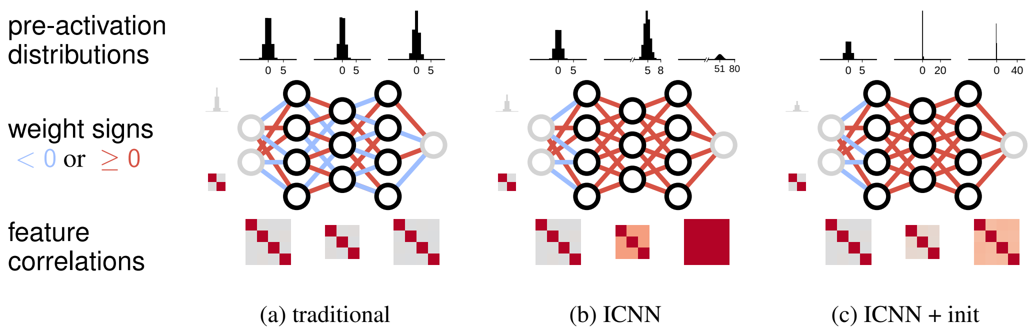


Figure 1: Illustration of the effects due to good signal propagation in hidden layers. Blue and red connections depict negative and positive weights, respectively. The top row shows histograms of pre-activations in each hidden layer. The bottom row displays the feature correlation matrices for these layers. Small visualisations on the left depict the input distribution.

2020) or during back-propagation (Glorot and Bengio, 2010; Hoedt et al., 2018; Defazio and Bottou, 2021). It is also important to account for the effects due to non-linearities in the network (Saxe et al., 2014; He et al., 2015; Klambauer et al., 2017; Hoedt et al., 2018). However, there are no initialisation strategies for non-negative weight matrices, which are a key component of ICNNs (Amos et al., 2017). We derive an initialisation strategy that accounts for the non-negative weights in ICNNs by generalising the signal propagation principles that underlie modern initialisation strategies.

Signal propagation. The derivation of initialisation strategies typically builds on the signal propagation framework introduced by Neal (1995). This signal propagation theory has been used and expanded in various ways (Saxe et al., 2014; Poole et al., 2016; Klambauer et al., 2017; Martens et al., 2021). One critical assumption in this traditional signal propagation theory is that weights are sampled from a centred distribution, i.e. with zero mean. In ICNNs, this is not possible because some weight matrices are constrained to be non-negative (Amos et al., 2017). Therefore, we generalise the traditional signal propagation theory to allow for non-centred distributions.

Skip-connections in ICNNs. Architectures of ICNNs typically include skip-connections (Amos et al., 2017; Sivaprasad et al., 2021; Makkuva et al., 2020; Nesterov et al., 2022). The skip-connections in ICNNs were introduced to increase their representational power (Amos et al., 2017). Although it is possible to study signal propagation with skip-connections (e.g. Yang and Schoenholz, 2017; Brock et al., 2021; Hoedt et al., 2022), they are typically built on top of existing results for plain networks. Therefore, we limit our theoretical results to ICNNs without skip-connections. However, we find that ICNNs without skip-connections can be successfully trained, indicating that skip-connections might not be necessary for representational power. We show that with our initialisation, we are able to train an ICNN to the same performance as a non-convex baseline without skip-connections. This confirms the hypothesis that good signal propagation can replace skip-connections (Martens et al., 2021; Zhang et al., 2022).

Contributions. Our contributions[1] can be summarised as follows:

- We generalise signal propagation theory to include weights without zero mean (Section 2).
- We derive a principled initialisation strategy for ICNNs from our new theory (Section 3).
- We empirically demonstrate the effectiveness of our initialisation strategy (Section 5.1).
- We apply ICNNs in a real-world drug-discovery setting (Sections 5.2 and 5.3).

2 Generalising Signal Propagation

In this section, we revisit the traditional signal propagation theory (Neal, 1995), which assumes centred weights. This provides us with a framework to study signal propagation in standard fully-connected networks. We then expand this framework to enable the study of networks where weights do not have zero mean to derive a weight initialisation strategy for ICNNs (Section 3).

[1]Code for figures and experiments can be found at `https://github.com/ml-jku/convex-init`

2.1 Background: Traditional Signal Propagation

When studying signal propagation in a network, the effect of a neural network layer on characteristics of the signal, such as its mean or variance, are investigated. The analysis of the vanishing gradient problem (Hochreiter, 1991) can also be considered as signal propagation theory where the norm of the delta-errors is tracked throughout layers in backpropagation. Typically, the network is assumed to be a repetitive composition of the same layers (e.g. fully-connected or convolutional). This allows to reduce the analysis of an entire network to a single layer and enables fixed-point analyses (e.g. Klambauer et al., 2017; Schoenholz et al., 2017). In our work, we focus on fully-connected networks, $f : \mathbb{R}^N \to \mathbb{R}^M$, with some activation function $\phi : \mathbb{R} \to \mathbb{R}$ that is applied element-wise to vectors. We study the propagation of pre-activations[2] (cf. Saxe et al., 2014; He et al., 2015; Poole et al., 2016) throughout the network:

$$\boldsymbol{s} = \boldsymbol{W}\phi(\boldsymbol{s}^-) + \boldsymbol{b}. \tag{1}$$

Here, $\boldsymbol{W} \in \mathbb{R}^{M\times N}$ and $\boldsymbol{b} \in \mathbb{R}^M$ are the weight matrix and bias vector, respectively. The notation $\boldsymbol{s}^-$ is used to indicate the pre-activations from the preceding layer, such that $\boldsymbol{s}^- \in \mathbb{R}^N$.

The first two moments of the signal can be expressed in terms of the randomness arising from the parameters. At initialisation time, the weight parameters are considered identically and independently distributed (i.i.d.) random variables $w_{ij} \sim \mathcal{D}_w$. The weight distribution is often assumed to be uniform or Gaussian, but the exact shape does not matter in wide networks (Golikov and Yang, 2022, Principle 2). The bias parameters, on the other hand, are commonly ignored in the analysis, which is justified when $\forall i : b_i = 0$. The pre-activations are also random variables due to the randomness of the parameters. If we assume the weights to be centred, such that $\mathbb{E}[w_{ij}] = 0$, and $\mathrm{Var}[w_{ij}] = \sigma_w^2$, the first two moments of the pre-activations are given by

$$\mathbb{E}[s_i] = \mathbb{E}\left[\sum_k w_{ik}\phi(s_k^-) + b_i\right] = 0 \tag{2}$$

$$\mathbb{E}\big[s_i^2\big] = \mathbb{E}\left[\left(\sum_k w_{ik}\phi(s_k^-) + b_i\right)^2\right] = N\sigma_w^2\,\mathbb{E}\big[\phi(s_1^-)^2\big]. \tag{3}$$

Note that the variance is independent of the index, i, in the pre-activation vector and thus $\forall k : \mathbb{E}\big[\phi(s_k^-)^2\big] = \mathbb{E}\big[\phi(s_1^-)^2\big]$. Moreover, it can be shown (see Appendix A) that $\forall i \neq j : \mathbb{E}[s_i s_j] = 0$, i.e. features within a pre-activation vector are uncorrelated in expectation.

Signal propagation theory has been used to derive initialisation and normalisation methods (LeCun et al., 1998; Glorot and Bengio, 2010; He et al., 2015; Klambauer et al., 2017). Initialisation methods often aim at having pre-activations with the same mean and variance in every layer of the network. Because the mean is expected to be zero in every layer, the focus lies on keeping the variance of the pre-activations constant throughout the network, i.e. $\forall i, j : \mathbb{E}\left[s_i^2\right] = \mathbb{E}\left[{s_j^-}^2\right] = \sigma_*^2$. Plugging this into Eq. (3), we obtain the fixed-point equation

$$\sigma_*^2 = N\sigma_w^2\,\mathrm{varprop}_\phi(\sigma_*^2). \tag{4}$$

Here, $\mathrm{varprop}_\phi$ is a function that models the propagation of variance through the activation function, ϕ, assuming zero mean inputs. E.g. $\mathrm{varprop}_{\mathrm{ReLU}}(\sigma^2) = \frac{1}{2}\sigma^2$ (He et al., 2015) or $\mathrm{varprop}_{\tanh}(\sigma^2) \approx \sigma^2$ (LeCun et al., 1998; Glorot and Bengio, 2010). We refer to Appendix A.3 for a more detailed discussion on propagation through activation functions.

This shows how modern initialisation methods are a solution to a fixed-point equation of the variance propagation. Our goal is to apply the same principles to find an initialisation strategy for the non-negative weights in ICNNs. However, Eq. (4) heavily relies on the assumption that the weights are centred, i.e. $\mu_w = 0$. This assumption is impossible to satisfy for the non-negative weights in ICNNs, unless all weights are set to zero. If $\mu_w \neq 0$, the mean of the pre-activations is no longer zero by default (as in Eq. 2). Furthermore, also the propagation of variance and covariance is affected. Therefore, we extend the signal propagation theory to accurately describe the effects due to non-centred weights.

[2]a common alternative is to consider the (post-)activations (see Hoedt et al., 2018)

2.2 Generalised Signal Propagation

We generalise the traditional signal propagation theory by lifting a few assumptions from the traditional approach. Most notably, the weights are no longer drawn from a zero-mean distribution, such that $\mathbb{E}[w_{ij}] = \mu_w \neq 0$. Additionally, we include the effects due to bias parameters, which will give us extra options to control the signal propagation. Similar to the weights, we assume bias parameters to be i.i.d. samples from some distribution with mean $\mathbb{E}[b_i] = \mu_b$ and variance $\mathrm{Var}[b_i] = \sigma_b^2$. By re-evaluating the expectations in Eq. (2) and Eq.(3), we arrive at the following results:

$$\mathbb{E}[s_i] = N\mu_w \, \mathbb{E}[\phi(s_1^-)] + \mu_b \tag{5}$$

$$\mathbb{E}[s_i s_j] = \delta_{ij} \left(N\sigma_w^2 \, \mathbb{E}\left[\phi(s_1^-)^2\right] + \sigma_b^2 \right) + \mu_w^2 \sum_{k,k'} \mathrm{Cov}[\phi(s_k^-), \phi(s_{k'}^-)] + \mathbb{E}[s_i] \, \mathbb{E}[s_j], \tag{6}$$

where δ_{ij} is the Kronecker delta. We refer to Appendix A.2 for a full derivation. Note that mean and variance are still independent of the index, i. Therefore, we can continue to assume $\forall k : \mathbb{E}\left[\phi(s_k^-)^n\right] = \mathbb{E}\left[\phi(s_1^-)^n\right]$.

There are a few crucial differences between the traditional approach and our generalisation. First, the expected value of the pre-activations is no longer zero by default. This also means that the second moment, $\mathbb{E}\left[s_i^2\right]$, does not directly model the variance of the pre-activations in our setting. Secondly, the propagation of the second moment has an additional term that incorporates effects due to the covariance structure of pre-activations in the previous layer. Finally, the off-diagonal elements of the feature covariance matrix, $\mathrm{Cov}[s_i, s_j] = \mathbb{E}_{i \neq j}[s_i s_j] - \mathbb{E}[s_i] \, \mathbb{E}[s_j]$, are not necessarily zero and also depend on the variance. This interaction between on- and off-diagonal elements in the feature covariance matrix makes it harder to ensure stable propagation.

Similar to Eq. (3), Eq. (6) depends on the variance propagation through the activation function. In addition, our generalised propagation theory involves the covariance propagation through the activation function. The covariance propagation through the ReLU function has been studied in prior work (Cho and Saul, 2009; Daniely et al., 2016) and can be specified as follows:

$$\mathbb{E}[\mathrm{ReLU}(s_1) \, \mathrm{ReLU}(s_2)] = \frac{\sigma^2}{2\pi} \left(\sqrt{1 - \rho^2} + \rho \arccos(-\rho) \right), \tag{7}$$

with $(s_1, s_2) \sim \mathcal{N}\left((0, 0), \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix} \sigma^2 \right)$ and correlation ρ. Note that this expression implicitly describes the propagation of the squared mean ($\rho = 0$) and the second raw moment ($\rho = 1$). We will assume $\phi = \mathrm{ReLU}$ for the remainder of our analysis. A derivation for the propagation through the leaky ReLU activation function (Maas et al., 2013) can be found in Appendix A.3.

With our generalised propagation theory, the effects due to $\mu_w \neq 0$ become apparent. As expected, the pre-activations no longer have zero mean by default. Furthermore, the covariance plays a significant role in the signal propagation. This shows that we also have to stabilise mean and covariance on top of the variance propagation to derive our principled initialisation.

3 Principled Initialisation of ICNNs

With our generalised signal propagation theory, we will now derive a principled initialisation strategy for ICNNs. First, we set up fixed point equations for mean, variance and correlation using our generalised framework. By solving these fixed point equations for μ_w, σ_w^2, μ_b and σ_b^2 we obtain our principled initialisation.

3.1 Background: Input-Convex Neural Networks

Neural networks for which all input-output mappings are convex, are called Input-Convex Neural Networks (ICNNs) (Amos et al., 2017). In general, neural networks are functions that are constructed by composing one or more simpler layers or components. A function composition $f \circ g$ is convex if both f and g are convex, where f additionally has to be non-decreasing in every output. Therefore, ICNNs are created by enforcing that every layer is *convex* and *non-decreasing* (Amos et al., 2017). A direct consequence is that only convex, non-decreasing activation functions can be used in ICNNs — e.g. softplus (Dugas et al., 2001), ReLU (Nair and Hinton, 2010), LReLU (Maas et al., 2013), ELU (Clevert et al., 2016).

Fully-connected and convolutional layers are affine transformations, which are trivially convex. To make these layers non-decreasing for building ICNNs, their weights have to be non-negative (Amos et al., 2017). This can be done by replacing negative values with zero after every update. The non-negativity constraint does not apply to bias parameters. Another notable exception are direct connections from the input layer (Amos et al., 2017). This means that the first layer and any skip-connections from the input can have unconstrained weights. Amos et al. (2017) argue that these skip-connections are necessary for representational power. We limit our analysis to networks without skip-connections. Our experiments (see Section 5) show that ICNNs with our proposed initialisation do not require skip-connections for efficient training.

3.2 (Co-)Variance Fixed Points

To set up the fixed-point equation for the second moments (Eq. 6), we need to be able to propagate through the ReLU non-linearity. However, the analytical results from Eq. (7) only hold if the pre-activations have zero mean. Therefore, we look for a configuration that makes the mean (Eq. 5) of the pre-activations zero. Because we cannot enforce $\mu_w = 0$, we make use of the bias parameters to obtain centred pre-activations:

$$\mu_b = -N\mu_w \, \mathbb{E}[\phi(s_1^-)]. \tag{8}$$

This also allows us to refer to the second moments as the (co-)variance. Note that we approximate the pre-activations with Gaussians to use Eq. (7) at this point and for the remainder of the analysis. Figure 1c suggests that this Gaussian assumption generally does not hold in ICNNs (see also Appendix A.3). Nevertheless, our experiments (see Section 5) suggest that the approximation is sufficiently good to derive an improved initialisation strategy for ICNNs. Furthermore, Figure 6b indicates that the use of random initial bias parameters (cf. Appendix C.3) causes pre-activations to be (slightly) more Gaussian.

The propagation of the second moment described by Eq. (6) consists of two parts. The first part describes the off-diagonal entries, for which the dynamics are given by

$$\mathrm{Cov}_{i\neq j}[s_i, s_j] = \mu_w^2 \sum_{k,k'} \mathrm{Cov}[\phi(s_k^-), \phi(s_{k'}^-)] = N\mu_w^2 \Big(\mathrm{Var}[\phi(s_1^-)] + (N-1)\, \mathrm{Cov}[\phi(s_1^-), \phi(s_2^-)] \Big).$$

Here, we rewrite the sum assuming that the pre-activations are identically distributed. This is possible because the covariance does not depend on the indices i or j. For further details, we refer to Appendix A.2 The second part models the on-diagonal entries, i.e. the variance, which can be simplified in a similar way

$$\mathrm{Var}[s_i] = N\sigma_w^2 \, \mathbb{E}[\phi(s_1^-)] + \sigma_b^2 + \mathrm{Cov}[s_1, s_2].$$

By plugging the ReLU moments from Eq. (7) into these results in Appendix B.2, we obtain the desired fixed-point equations in terms of variance and correlation:

$$\rho_* = \mu_w^2 \frac{1}{2\pi} N \Big(\pi - N + (N-1)\big(\sqrt{1-\rho_*^2} + \rho_* \arccos(-\rho_*)\big)\Big) \tag{9}$$

$$\sigma_*^2 = N\sigma_w^2 \frac{1}{2}\sigma_*^2 + \sigma_b^2 + \rho_*\sigma_*^2. \tag{10}$$

with $\rho_* = \frac{1}{\sigma_*^2}\,\mathrm{Cov}[s_1^-, s_2^-] = \frac{1}{\sigma_*^2}\,\mathrm{Cov}_{i\neq j}[s_i, s_j]$ and $\sigma_*^2 = \mathrm{Var}[s_1^-] = \mathrm{Var}[s_i]$.

3.3 Weight Distribution for ICNNs

To obtain the distribution parameters for the initial weights, we solve the fixed point equations in Eq. (9) and Eq.(10) for σ_b^2, σ_w^2 and μ_w^2. Because this system is over-parameterised, we choose to set $\sigma_b^2 = 0$. An analysis for $\sigma_b^2 > 0$ can be found in Appendix C.3. This leads to the following initialisation parameters for ICNNs:

$$\sigma_w^2 = \frac{2}{N}(1-\rho_*) \tag{11}$$

$$\mu_w^2 = \frac{2\pi}{N}\rho_* \Big(\pi - N + (N-1)\big(\sqrt{1-\rho_*^2} + \rho_* \arccos(-\rho_*)\big)\Big)^{-1}. \tag{12}$$

Note that these solutions only depend on the correlation, ρ_*, and not on the variance, σ_*^2. For a stability analysis of these fixed points, we refer to Appendix B.4.

Although our initialisation (Eq. 12 and 11) is entirely defined by the correlation between features, $\rho = \frac{\mathrm{Cov}[s_1, s_2]}{\mathrm{Var}[s_1]}$, not all solutions are admissible. For example, uncorrelated features ($\rho = 0$) are only possible if $\mu_w = 0$. This means that features in ICNNs must have non-zero correlation. On the other hand, perfect correlation ($\rho = 1$) would mean that all features point in the same direction, which is not desirable. Therefore, we choose $\rho_* = \frac{1}{2}$ as a compromise to obtain the following initialisation parameters for our experiments in Section 5:

$$\mu_w = \sqrt{\frac{6\pi}{N\big(6(\pi - 1) + (N - 1)(3\sqrt{3} + 2\pi - 6)\big)}} \qquad \sigma_w^2 = \frac{1}{N}$$

$$\mu_b = \sqrt{\frac{3N}{6(\pi - 1) + (N - 1)(3\sqrt{3} + 2\pi - 6)}} \qquad \sigma_b^2 = 0.$$

We refer to appendix C.4 for a discussion on different choices for ρ_*.

For the first layer, which can also have negative weights, we use LeCun et al. (1998) initialisation. For the non-negative weights in an ICNNs, we need to sample from a distribution with non-negative support. The lower bound of a uniform distribution with our suggested mean and variance is negative for any $N > 1$ and thus not useful. Also, sampling from a Gaussian distribution is not practical because its support includes negative values. Eventually, we propose to sample from a log-normal distribution with parameters

$$\tilde{\mu_w} = \ln(\mu_w^2) - \frac{1}{2}\ln(\sigma_w^2 + \mu_w^2) \qquad \tilde{\sigma_w}^2 = \ln(\sigma_w^2 + \mu_w^2) - \ln(\mu_w^2).$$

This ensures that sampled weights are non-negative and have the desired mean and variance. Note that other positive distributions with sufficient degrees of freedom should also work.

One advantage of the log-normal distribution is that sampling is simple and efficient. It suffices to sample $\tilde{w}_{ij} \sim \mathcal{N}(\tilde{\mu_w}, \tilde{\sigma_w}^2)$ to compute $w_{ij} = \exp\big(\tilde{w}_{ij}\big)$. In the original ICNN (Amos et al., 2017), the exponential function is only used to make the initial weights positive. However, if weights are re-parameterised using the exponential function, initial weights can be directly sampled from a Gaussian distribution. This setting is studied in Appendix C.5.

4 Related Work

Input-convex neural networks. Input-Convex Neural Network (ICNN) were originally designed in the context of energy models (Amos et al., 2017). We focus on the fully convex ICNNs variant and use gradient descent for optimisation instead of the proposed bundle-entropy method. In their implementation, Amos et al. (2017) use projection methods to keep weights positive, i.e. negative values are set to zero after every update. However, also other projection methods can be used to keep the weights positive (e.g. Sivaprasad et al., 2021). Instead of projecting the weights onto the non-negative reals after every update, it is also possible to use a reparameterisation of the weights. E.g. Nesterov et al. (2022) square the weights in the forward pass. Note that a reparameterisation can have an effect on the learning dynamics because it is an inherent part of the forward pass. Another alternative is to use regularisation to impose a *soft* non-negativity constraint on the weights (Makkuva et al., 2020). Similar to (Sivaprasad et al., 2021), we directly train an ICNN, but we do not observe the same generalisation benefits. On the other hand, Sankaranarayanan and Rengaswamy (2022) point out that restricting the network to be convex decreases their capacity. They propose to use the difference of two ICNNs to allow modelling non-convex functions (c.f. Yuille and Rangarajan, 2001). However, we did not find the convexity restriction to cause any problems in our experiments. Nesterov et al. (2022) use ICNNs to simplify the computation of level sets to learn invariances. We adopt this experiment setting to highlight the potential of ICNNs. *Optimization.* While training the parameters of an ICNN is unrelated to convex optimisation, Bengio et al. (2005) showed how training a neural network together with the number of hidden neurons can be seen as a convex optimisation problem. This work was continued by Bach (2017), who established a connection between this convex optimisation and automatic feature selection. *Signal propagation theory.* The study of signal propagation is concerned with tracking statistics of the data throughout multiple layers of the network. In his thesis, Neal (1995) computed how the first two moments propagate through a two-layer network to study networks as Gaussian processes. This is similar to how modern initialisation strategies have been derived (e.g. LeCun et al., 1998; He et al., 2015). A similar approach has been taken to incorporate the propagation of gradients for initialisation (Glorot and Bengio, 2010). The idea to explicitly account for the effects of activation functions can be attributed to (Saxe et al., 2014). All of these analyses assume that weights are initialised from a zero-mean distribution, which is not possible in ICNNs. Mishkin and Matas (2016) empirically compute the variance for the weights. This approach is more robust to deviations, but nevertheless requires adaptations to account for weights with non-zero mean. There is also a significant body of work that studies signal propagation in terms of mean field theory (Poole et al., 2016). In these works, the correlation between samples is included in the analysis (Poole et al., 2016). In our work, we find that the correlation between features plays an important role. Schoenholz et al. (2017) applied the mean field analysis to the backward dynamics and introduced the concept of *depth scales* to quantify how deep signals can propagate. The mean field theory was also applied to convolutional networks to derive the delta-orthogonal initialisation (Xiao et al., 2018). We refer to (Martens et al., 2021) for a general overview of signal propagation.

5 Experiments

We evaluate our principled initialisation strategy by training ICNNs on three sets of experiments. In our first experiments, we investigate the effect of initialisation on learning dynamics and generalisation in ICNNs on multiple permuted image datasets. We also include non-convex networks in these experiments to illustrate that ICNNs with our principled initialisation can be trained as well as regular networks. However, we would like to stress that non-convex networks are expected to outperform ICNNs because they are not constrained to convex decision boundaries. Moreover, we assess whether ICNNs using our initialisation are able to match the performance of regular fully-connected networks in toxicity prediction without the need for skip-connections. Finally, we explore ICNNs as a tool for more controlled latent space exploration of a molecule auto-encoder system. This illustrates a setting where regular networks can no longer be used. Details on the computing budget are provided in the Appendix.

5.1 Computer vision benchmarking datasets

As an illustrative first set of experiments, we use computer vision benchmarks to assess the effects of our principled initialisation strategy. Concretely, we trained fully-connected ICNNs on MNIST (Bottou et al., 1994), CIFAR10 and CIFAR100 (Krizhevsky, 2009), to which we refer as permuted image benchmarks (cf. Goodfellow et al., 2014).

In our comparison, we consider four different types of methods: (1) a classical non-convex network without skip-connections and default initialisation (cf. He et al., 2015). (2) an ICNN, "ICNN", without skip connections and default initialisation (3) an ICNN, "ICNN + skip", with skip connections and default initialisation (4) an ICNN, "ICNN + init", without skip connections and our proposed initialisation. All networks use ReLU activation functions. The non-negativity constraint in ICNNs is implemented by setting negative weights to zero after every update (cf. Amos et al., 2017). Note that skip-connections in ICNNs introduce additional parameters to allow for additional representational power (cf. Amos et al., 2017).

Training dynamics. *Settings.* In the first experiment, we analyze the training dynamics during the first ten epochs of training. Similar to Chang et al. (2020), we fixed the number of neurons in each layer to the input size, and the number of hidden layers to five. We refer to Appendix C.2 for results with different depths. The learning rate for the Adam optimiser was obtained by manually tuning on the non-convex baseline. *Results.* We found that ICNNs with our principled initialisation exhibit similar learning behaviour as the non-convex baseline, while ICNNs without our initialisation strategy can not decrease the loss to the level of the baseline on CIFAR10 and CIFAR100. Figure 2 shows the learning curves for our different methods on three permuted image benchmarks.

Generalisation. *Settings.* In our next experiment, the effects of our principled initialisation on the generalisation capabilities of ICNNs is investigated. To this end, we train our three ICNN variants and the non-convex baseline on the permuted image benchmarks again, but focus on the test performance this time. To this end we perform a grid-search to find the hyper-parameters that attain the

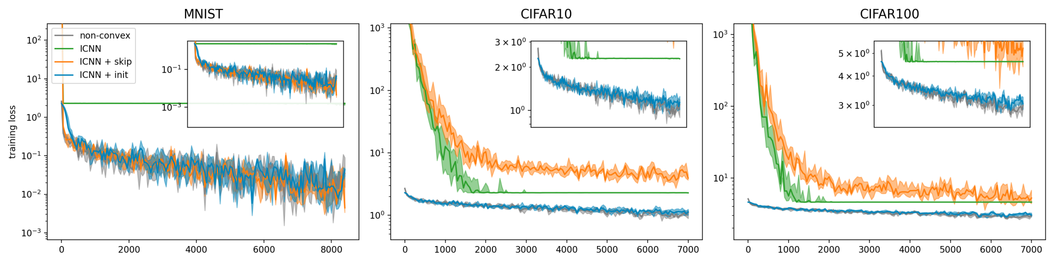


Figure 2: Training loss curves of ICNN variants with the same architecture on the MNIST, CIFAR10 and CIFAR100 datasets. "ICNN" input-convex network with default initialisation. "ICNN + skip": same settings, but with skip connections. "ICNN + init" our principled initialisation for ICNNs w/o skip connections. "non-convex": a regular non-convex network. Each curve represents the median performance over ten runs and shaded regions indicate the inter-quartile range. The inset figures provide a view of the loss curves zoomed in. Note that ICNN losses do decrease before the plateau.

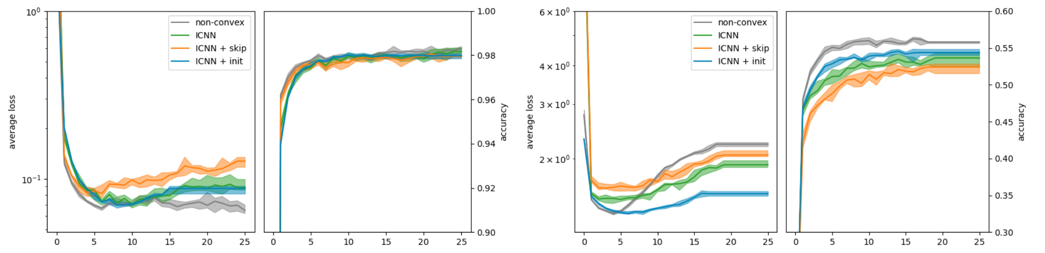


Figure 3: Test set metrics of compared methods on the (a) MNIST and (b) CIFAR10 datasets. Each curve displays the median performance over ten runs. Shaded regions represent the inter-quartile range over the ten runs. On MNIST, all methods can be successfully trained and exhibit similar performance. On CIFAR 10, ICNNs with our proposed initialisation outperform other ICNNs variants.

best accuracy after 25 epochs of training on a random validation split, for each of the four compared methods. The grid consists of multiple architectures with at least one hidden layer, learning rate for Adam, L_2-regularisation and whitening pre-processing transforms (details in Appendix C.2). *Results.* On the MNIST dataset, all variants reach similar accuracy values, which we attribute to the simplicity of the prediction problem. However, on CIFAR10 the initialisation strategy leads to better generalisation already in early epochs and — compared to the respective ICNNs variants without our initialisation — improves generalisation overall (see Figure 3). The ICNN with our initialisation almost matches the accuracy values of non-convex nets.

5.2 Toxicity Prediction

To test ICNNs in a real-world setting, in a different application domain, we consider the binary multi-task problem of toxicity prediction in drug discovery. More specifically, we train ICNNs on the Tox21 challenge data (Huang et al., 2016; Mayr et al., 2016; Klambauer et al., 2017). The input data consists of so-called SMILES representations of small molecules. The target variables are binary labels for twelve different measurements or assays indicating whether a molecule induces a particular toxic effect on human cells. During pre-processing, SMILES strings were converted to Continuous and Data-Driven Descriptors (CDDDs) (Winter et al., 2019), which are 512-dimensional numerical representations of the molecules. The choice of these particular descriptors is motivated by CDDDs properties of being decodeable into valid molecules (see Section 5.3). Hyper-parameters were selected by a manual search on the non-convex baseline. We ended up using a fully-connected network with two hidden layers of 128 neurons and ReLU activations. The network was regularised with fifty percent dropout after every hidden layer, as well as seventy percent dropout of the inputs. The results in Table 1 show that our initialisation significantly outperforms ICNNs with standard initialization (p-value 2.6e-13, binomial test) and ICNNs with skip connections (p-value 5.5e-14). Furthermore, results are close to the performance of traditional, non-convex networks.

Table 1: Area under the ROC curve on the test set for each of the twelve tasks in the Tox21 data. Each value represents the median performance over 10 runs and error bars is the maximum distance to the boundary of the interval defined by the (0.05, 0.95) quantiles. Of all variants of ICNNs, the variant with our proposed initialisation, “ICNN+init (ours)”, performs best matching almost the predictive quality of traditional non-convex neural networks.

	NR.AhR	NR.AR	NR.AR.LBD	NR.Aromatase	NR.ER	NR.ER.LBD	
ICNN+init (ours)	$\mathbf{91.29} \pm 0.58\%$	$\mathbf{81.84} \pm 2.43\%$	$\mathit{81.36} \pm 3.58\%$	$\mathit{82.40} \pm 1.09\%$	$\mathbf{78.37} \pm 0.80\%$	$77.85 \pm 0.90\%$	
ICNN	$90.56 \pm 0.63\%$	$\mathit{81.28} \pm 4.82\%$	$78.14 \pm 3.15\%$	$80.46 \pm 1.74\%$	$77.30 \pm 0.78\%$	$\mathit{78.15} \pm 1.02\%$	
ICNN+skip	$89.83 \pm 0.21\%$	$68.12 \pm 1.74\%$	$74.17 \pm 2.20\%$	$78.95 \pm 0.45\%$	$76.95 \pm 0.48\%$	$\mathbf{81.92} \pm 1.16\%$	
non-convex	$\mathit{91.01} \pm 0.65\%$	$79.21 \pm 1.73\%$	$\mathbf{86.19} \pm 2.50\%$	$\mathbf{83.63} \pm 0.34\%$	$\mathit{77.88} \pm 1.13\%$	$74.32 \pm 1.59\%$	
	SR.ATAD5	SR.HSE	SR.MMP	SR.p53	NR.PPARγ	SR.ARE	AVG
ICNN+init (ours)	$77.01 \pm 1.67\%$	$\mathit{80.05} \pm 1.36\%$	$\mathit{93.69} \pm 0.27\%$	$81.10 \pm 0.46\%$	$77.46 \pm 2.50\%$	$76.80 \pm 0.33\%$	$\mathit{80.57} \pm 11.84\%$
ICNN	$74.25 \pm 3.26\%$	$78.64 \pm 1.42\%$	$93.19 \pm 0.59\%$	$\mathit{81.32} \pm 0.43\%$	$75.00 \pm 1.73\%$	$76.11 \pm 0.82\%$	$78.33 \pm 13.42\%$
ICNN+skip	$\mathit{75.93} \pm 1.26\%$	$78.23 \pm 0.50\%$	$75.40 \pm 0.88\%$	$78.25 \pm 0.93\%$	$\mathbf{92.09} \pm 0.41\%$	$\mathbf{79.32} \pm 0.97\%$	$78.29 \pm 12.56\%$
non-convex	$\mathbf{78.93} \pm 1.68\%$	$\mathbf{80.12} \pm 2.57\%$	$\mathbf{93.99} \pm 0.39\%$	$\mathbf{82.17} \pm 0.96\%$	$\mathit{82.70} \pm 1.21\%$	$\mathit{78.30} \pm 0.90\%$	$\mathbf{81.31} \pm 11.03\%$

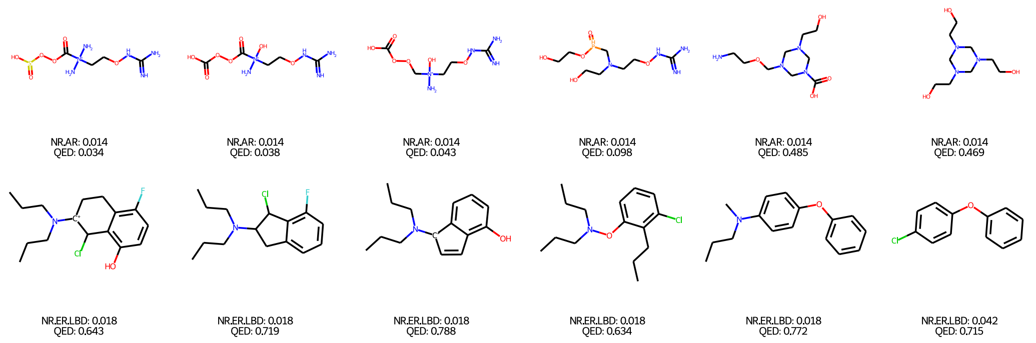


Figure 4: Example of two level set trajectories of a Tox21 model. The leftmost molecule represents the starting molecule with low toxicity (top: “NR.AR”, bottom: “NR.ER.LBD”). The rightmost molecule represents the target molecule. All intermediary molecules are samples on the level set between the starting molecule and the target molecule and exhibit different drug-likeness (“QED”). In this way, one predicted molecular property can be kept fixed while another property is optimized.

5.3 Latent space exploration of molecular spaces

In this experiment, we exploit the intrinsic property of ICNNs that level sets, i.e. the set of all inputs that map to the same output, can be parameterised (Nesterov et al., 2022). These level sets provide an opportunity in drug discovery to keep one molecular property, such as low toxicity, fixed while optimizing another property, such as drug-likeness. To demonstrate this application, we randomly chose reference molecules with low toxicity and followed the according level set in the direction of another randomly chosen target molecules. Since the input space of the ICNN that predicts toxicity (see above) possesses a decoder model from (Winter et al., 2019), the numeric representations of the molecules on the level set can be decoded into molecular structures. The obtained trajectories in Figure 4 demonstrate that the predicted toxicity of the molecules remain constant, while different QED scores, which measure drug-likeness, are obtained by traversing the level set from the reference molecule to the target molecule.

6 Conclusion and Discussion

We demonstrated how signal propagation theory can be generalised to include weights without zero mean. By controlling the propagation of correlation through the network, we derived an initialisation strategy that promotes a stable propagation of signals. We empirically verified that this steady propagation leads to improved learning. Finally, we showed that ICNNs can be trained to a comparable level as regular networks on the Tox21 data using CDDDs.

Our initialisation approximates the distribution of pre-activations with Gaussians, which falls short of agreement with the observed data, but has been sufficient to improve the initialisation of ICNNs. A possible way forward is to use the central limit theorem for weakly dependent variables. In Appendix C.3, we also observed that random bias initialisation leads to pre-activations with a more Gaussian distribution. Appendix A.5 also includes an analysis for convolutional layers, but proficient empirical results with input-convex convolutional networks are still to be made. We conjecture that this is due to the more complicated covariance structure in convolutional layers. We also included a signal propagation analysis for back-propagation in Appendix A.4. However, incorporating insights from this analysis into an initialisation method is also left for future work. We envision that our initialisation strategy will make it easier to incorporate ICNNs or other networks with non-negative weights.

Acknowledgments and Disclosure of Funding

We thank Sepp Hochreiter for pointing us in the direction of input-convex networks and their interesting properties. Special thanks go to Niklas Schmidinger for his help with experiments on convolutional networks.

The ELLIS Unit Linz, the LIT AI Lab, the Institute for Machine Learning, are supported by the Federal State Upper Austria. We thank the projects AI-MOTION (LIT-2018-6-YOU-212), DeepFlood (LIT-2019-8-YOU-213), Medical Cognitive Computing Center (MC3), INCONTROL-RL (FFG-881064), PRIMAL (FFG-873979), S3AI (FFG-872172), DL for GranularFlow (FFG-871302), EPILEPSIA (FFG-892171), AIRI FG 9-N (FWF-36284, FWF-36235), AI4GreenHeatingGrids(FFG- 899943), INTEGRATE (FFG-892418), ELISE (H2020-ICT-2019-3 ID: 951847), Stars4Waters (HORIZON-CL6-2021-CLIMATE-01-01). We thank Audi.JKU Deep Learning Center, TGW LOGISTICS GROUP GMBH, Silicon Austria Labs (SAL), FILL Gesellschaft mbH, Anyline GmbH, Google, ZF Friedrichshafen AG, Robert Bosch GmbH, UCB Biopharma SRL, Merck Healthcare KGaA, Verbund AG, GLS (Univ. Waterloo) Software Competence Center Hagenberg GmbH, TÜV Austria, Frauscher Sensonic, TRUMPF and the NVIDIA Corporation.

References

Amos, B., Xu, L., and Kolter, J. Z. (2017). Input convex neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pages 146–155. PMLR.

Bach, F. (2017). Breaking the curse of dimensionality with convex neural networks. *Journal of Machine Learning Research*, 18(19):1–53.

Bengio, Y., Roux, N., Vincent, P., Delalleau, O., and Marcotte, P. (2005). Convex neural networks. In *Advances in Neural Information Processing Systems*, volume 18, pages 123–130. MIT Press.

Bottou, L., Cortes, C., Denker, J., Drucker, H., Guyon, I., Jackel, L., LeCun, Y., Muller, U., Sackinger, E., Simard, P., and Vapnik, V. (1994). Comparison of classifier methods: a case study in handwritten digit recognition. In *Proceedings of the 12th IAPR International Conference on Pattern Recognition*, volume 2, pages 77–82, Jerusalem, Israel. IEEE.

Brock, A., De, S., and Smith, S. L. (2021). Characterizing signal propagation to close the performance gap in unnormalized ResNets. In *International Conference on Learning Representations*, volume 9, virtual. arXiv: 2101.08692.

Chang, O., Flokas, L., and Lipson, H. (2020). Principled weight initialization for hypernetworks. In *International Conference on Learning Representations*, volume 8.

Cho, Y. and Saul, L. (2009). Kernel methods for deep learning. *Advances in neural information processing systems*, 22.

Clevert, D.-A., Unterthiner, T., and Hochreiter, S. (2016). Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs). In *International Conference on Learning Representations*, volume 4, San Juan, Puerto Rico. arXiv: 1511.07289.

Daniely, A., Frostig, R., and Singer, Y. (2016). Toward Deeper Understanding of Neural Networks: The Power of Initialization and a Dual View on Expressivity. In *Advances in Neural Information Processing Systems*, volume 29, pages 2253–2261, Barcelona, Spain. Curran Associates, Inc.

Defazio, A. and Bottou, L. (2021). Beyond Folklore: A Scaling Calculus for the Design and Initialization of ReLU Networks. arXiv: 1906.04267.

Dugas, C., Bengio, Y., Bélisle, F., Nadeau, C., and Garcia, R. (2001). Incorporating Second-Order Functional Knowledge for Better Option Pricing. In *Advances in Neural Information Processing Systems*, volume 13, pages 472–478, Vancouver, BC, Canada. MIT Press.

Glorot, X. and Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9, pages 249–256.

Golikov, E. and Yang, G. (2022). Non-gaussian tensor programs. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K., editors, *Advances in Neural Information Processing Systems*, volume 35.

Goodfellow, I. J., Mirza, M., Da, X., Courville, A. C., and Bengio, Y. (2014). An Empirical Investigation of Catastrophic Forgeting in Gradient-Based Neural Networks. In *International Conference on Learning Representations*, volume 2.

He, K., Zhang, X., Ren, S., and Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034.

Hochreiter, S. (1991). Untersuchungen zu dynamischen neuronalen Netzen. Master's thesis, Technical University Munich, Munich, Germany.

Hoedt, P.-J., Hochreiter, S., and Klambauer, G. (2018). Characterising activation functions by their backward dynamics around forward fixed points. NeurIPS workshop on Critiquing and Correcting Trends in Machine Learning.

Hoedt, P.-J., Hochreiter, S., and Klambauer, G. (2022). Normalization is dead, long live normalization! In *ICLR Blog Track*. https://iclr-blog-track.github.io/2022/03/25/unnormalized-resnets/.

Huang, R., Xia, M., Nguyen, D.-T., Zhao, T., Sakamuru, S., Zhao, J., Shahane, S. A., Rossoshek, A., and Simeonov, A. (2016). Tox21Challenge to Build Predictive Models of Nuclear Receptor and Stress Response Pathways as Mediated by Exposure to Environmental Chemicals and Drugs. *Frontiers in Environmental Science*, 3.

Klambauer, G., Unterthiner, T., Mayr, A., and Hochreiter, S. (2017). Self-normalizing neural networks. *Advances in neural information processing systems*, 30.

Krizhevsky, A. (2009). Learning Multiple Layers of Features from Tiny Images. Technical report, Canadian Institute for Advanced Research.

LeCun, Y., Bottou, L., Orr, G. B., and Müller, K. R. (1998). Efficient BackProp. In Orr, G. B. and Müller, K.-R., editors, *Neural Networks: Tricks of the Trade*, Lecture Notes in Computer Science, pages 9–50. Springer, Berlin, Heidelberg, 1 edition.

Maas, A. L., Hannun, A. Y., and Ng, A. Y. (2013). Rectifier Nonlinearities Improve Neural Network Acoustic Models. In *Workhop for Deep Learning for Audio, Speech and Language Processing at ICML*, Atlanta, GA, USA.

Makkuva, A., Taghvaei, A., Oh, S., and Lee, J. (2020). Optimal transport mapping via input convex neural networks. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pages 6672–6681. PMLR.

Martens, J., Ballard, A., Desjardins, G., Swirszcz, G., Dalibard, V., Sohl-Dickstein, J., and Schoenholz, S. S. (2021). Rapid training of deep neural networks without skip connections or normalization layers using Deep Kernel Shaping. arXiv: 2110.01765.

Mayr, A., Klambauer, G., Unterthiner, T., and Hochreiter, S. (2016). Deeptox: toxicity prediction using deep learning. *Frontiers in Environmental Science*, 3:80.

Mishkin, D. and Matas, J. (2016). All you need is a good init. In *International Conference on Learning Representations*, volume 4.

Nair, V. and Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning*, pages 807–814, Madison, WI, USA. Omnipress.

Neal, R. M. (1995). *Bayesian Learning for Neural Networks*. PhD thesis, University of Toronto.

Nesterov, V., Torres, F. A., Nagy-Huber, M., Samarin, M., and Roth, V. (2022). Learning invariances with generalised input-convex neural networks.

Poole, B., Lahiri, S., Raghu, M., Sohl-Dickstein, J., and Ganguli, S. (2016). Exponential expressivity in deep neural networks through transient chaos. In *Advances in Neural Information Processing Systems*, volume 29, pages 3360–3368.

Sankaranarayanan, P. and Rengaswamy, R. (2022). CDiNN – convex difference neural networks. *Neurocomputing*, 495:153–168.

Saxe, A., McClelland, J., and Ganguli, S. (2014). Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. In *International Conference on Learning Representations*, volume 2.

Schoenholz, S. S., Gilmer, J., Ganguli, S., and Sohl-Dickstein, J. (2017). Deep information propagation. In *International Conference on Learning Representations*.

Sivaprasad, S., Singh, A., Manwani, N., and Gandhi, V. (2021). The curious case of convex neural networks. In Oliver, N., Pérez-Cruz, F., Kramer, S., Read, J., and Lozano, J. A., editors, *Machine Learning and Knowledge Discovery in Databases. Research Track*, Lecture Notes in Computer Science, pages 738–754. Springer International Publishing.

Wang, Z. and Wu, Z. (2023). Input convex lstm: A convex approach for fast Lyapunov-based model predictive control. *arXiv preprint arXiv:2311.07202*.

Winter, R., Montanari, F., Noé, F., and Clevert, D.-A. (2019). Learning continuous and data-driven molecular descriptors by translating equivalent chemical representations. *Chemical Science*, 10(6):1692–1701. Publisher: The Royal Society of Chemistry.

Xiao, L., Bahri, Y., Sohl-Dickstein, J., Schoenholz, S., and Pennington, J. (2018). Dynamical isometry and a mean field theory of cnns: How to train 10,000-layer vanilla convolutional neural networks. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pages 5393–5402. PMLR.

Yang, G. and Schoenholz, S. (2017). Mean field residual networks: On the edge of chaos. *Advances in neural information processing systems*, 30.

Yuille, A. L. and Rangarajan, A. (2001). The concave-convex procedure (cccp). In *Advances in Neural Information Processing Systems*, volume 13, pages 1033–1040.

Zhang, G., Botev, A., and Martens, J. (2022). Deep learning without shortcuts: Shaping the kernel with tailored rectifiers. In *International Conference on Learning Representations*, volume 10, virtual.

Contents of the Appendix

A Generalisation of Signal Propagation

This section provides the derivations for the generalised signal propagation.

A.1 Traditional Signal Propagation

One of the key assumptions in traditional signal propagation is that the weights are drawn from a zero-mean distribution with some variance σ_w^2. E.g. $w \sim \mathcal{N}(0, \sigma_w^2)$ or $w \sim \mathcal{U}\Big(-\sqrt{3\sigma_w^2}, \sqrt{3\sigma_w^2}\Big)$ are typical distributions for sampling initial weights. Also, bias parameters are typically assumed to be initialised with zeros, such that mean and variance are $\mu_b = 0$ and $\sigma_b^2 = 0$, respectively. Since the initial weights are drawn from i.i.d. samples, the first two (raw) moments of the pre-activations from eq. (1) be directly computed as follows:

$$\begin{aligned}
\mathbb{E}[s_i] &= \mathbb{E}\Big[b_i + \sum_k w_{ik}\phi(s_k^-)\Big] \\
&= \mathbb{E}[b_i] + \sum_k \mathbb{E}[w_{ik}]\,\mathbb{E}[\phi(s_k^-)] \\
&= \mu_b + \sum_k \mu_w\,\mathbb{E}[\phi(s_k^-)] = 0 \qquad (13) \\
\mathbb{E}\big[s_i^2\big] &= \mathbb{E}\bigg[\Big(b_i + \sum_k w_{ik}\phi(s_k^-)\Big)^2\bigg] \\
&= \mathbb{E}\big[b_i^2\big] + 2\,\mathbb{E}[b_i]\sum_k \mathbb{E}[w_{ik}]\,\mathbb{E}[\phi(s_k^-)] + \mathbb{E}\bigg[\Big(\sum_k w_{ik}\phi(s_k^-)\Big)^2\bigg] \\
&= (\sigma_b^2 + \mu_b^2) + 2\mu_b\sum_k \mu_w\,\mathbb{E}[\phi(s_k^-)] + \mathbb{E}\Big[\sum_{k,k'} w_{ik}w_{ik'}\phi(s_k^-)\phi(s_{k'}^-)\Big] \\
&= \mathbb{E}\Big[\sum_{k,k'} w_{ik}w_{ik'}\phi(s_k^-)\phi(s_{k'}^-)\Big] \\
&= \sum_k \mathbb{E}\big[w_{ik}^2\big]\,\mathbb{E}\big[\phi(s_k^-)^2\big] + \sum_{k,k'\neq k}\mathbb{E}[w_{ik}]\,\mathbb{E}[w_{ik'}]\,\mathbb{E}[\phi(s_k^-)\phi(s_{k'}^-)] \\
&= \sum_k(\sigma_w^2 + \mu_w^2)\,\mathbb{E}\big[\phi(s_k^-)^2\big] + \sum_{k,k'\neq k}\mu_w^2\,\mathbb{E}[\phi(s_k^-)\phi(s_{k'}^-)] \\
&= \underbrace{\sigma_w^2\sum_k \mathbb{E}\big[\phi(s_k^-)^2\big]}_{\mathrm{Var}[s_i]}. \qquad (14)
\end{aligned}$$

If we additionally assume that the pre-activations from the previous layer are identically distributed, such that

$$\forall k : \mathbb{E}\big[\phi(s_k^-)^2\big] = \mathbb{E}\big[\phi(s_1^-)^2\big], \qquad (15)$$

we retrieve the traditional signal propagation formulas (equations 2 and 3):

$$\begin{aligned}
\mathbb{E}[s_i] &= 0 \\
\mathbb{E}\big[s_i^2\big] &= N\sigma_w^2\,\mathbb{E}\big[\phi(s_1^-)^2\big],
\end{aligned}$$

where N is the number of incoming connections, such that $\boldsymbol{s}^- \in \mathbb{R}^N$.

For the first layer, the recursion formulas in this section can obviously not be used. The moments of the pre-activations in the first layer can be computed using a very similar derivation, however:

$$\begin{aligned}
\mathbb{E}[s_i] &= \mathbb{E}\Big[b_i + \sum_k w_{ik}x_k\Big] & \mathbb{E}\big[s_i^2\big] &= \mathbb{E}\bigg[\Big(b_i + \sum_k w_{ik}x_k\Big)^2\bigg] \\
&= \mu_b + \mu_w\sum_k \mathbb{E}[x_k] = 0 & &= \sigma_w^2\sum_k \mathbb{E}\big[x_k^2\big].
\end{aligned}$$

There are two possible ways to interpret the inputs. If we wish to treat inputs as random variables, we would also need x_k to be identically distributed to obtain

$$\mathbb{E}[s_i] = 0 \qquad\qquad \mathbb{E}\big[s_i^2\big] = N\sigma_w^2\,\mathbb{E}\big[x_1^2\big]$$

Alternatively, we can just use the input data as constants, such that $\mathbb{E}\big[x_k^2\big] = x_k^2$. In this scenario, the pre-activation moments are

$$\mathbb{E}[s_i] = 0 \qquad\qquad \mathbb{E}\big[s_i^2\big] = \sigma_w^2\|\boldsymbol{x}\|.$$

A.2 Generalised Signal Propagation

In ICNNs, the zero-mean assumption on the weights does not hold. Therefore, we apply the signal propagation analysis assuming that weights have mean μ_w and variance σ_w^2. Furthermore, we also account for non-zero bias initialisation by not making any assumptions on the mean and variance of the bias parameters. In this setting, the first two (raw) moments of the pre-activations from eq. (1) are:

$$\begin{aligned}
\mathbb{E}[s_i] &= \mathbb{E}\Big[b_i + \sum_k w_{ik}\phi(s_k^-)\Big] \\
&= \mu_b + \mu_w \sum_k \mathbb{E}[\phi(s_k^-)] && (16) \\
\mathbb{E}\big[s_i^2\big] &= \mathbb{E}\Big[\Big(b_i + \sum_k w_{ik}\phi(s_k^-)\Big)^2\Big] \\
&= \sigma_b^2 + \mu_b^2 + 2\mu_b\mu_w \sum_k \mathbb{E}[\phi(s_k^-)] + \mathbb{E}\Big[\sum_{k,k'} w_{ik}w_{ik'}\phi(s_k^-)\phi(s_{k'}^-)\Big] \\
&= \sigma_b^2 + \mu_b^2 + 2\mu_b\mu_w \sum_k \mathbb{E}[\phi(s_k^-)] + (\sigma_w^2 + \mu_w^2)\sum_k \mathbb{E}\big[\phi(s_k^-)^2\big] + \mu_w^2 \sum_{k,k'\neq k} \mathbb{E}[\phi(s_k^-)\phi(s_{k'}^-)] \\
&= \sigma_b^2 + \mu_b^2 + 2\mu_b\mu_w \sum_k \mathbb{E}[\phi(s_k^-)] + \sigma_w^2 \sum_k \mathbb{E}\big[\phi(s_k^-)^2\big] + \mu_w^2 \sum_{k,k'} \mathbb{E}[\phi(s_k^-)\phi(s_{k'}^-)] \\
&= \sigma_b^2 + \sigma_w^2 \sum_k \mathbb{E}\big[\phi(s_k^-)^2\big] + \mu_w^2 \sum_{k,k'} \mathbb{E}[\phi(s_k^-)\phi(s_{k'}^-)] \\
&\qquad - \mu_w^2 \sum_{k,k'} \mathbb{E}[\phi(s_k^-)]\,\mathbb{E}[\phi(s_{k'}^-)] + \Big(\mu_b + \mu_w \sum_k \mathbb{E}[\phi(s_k^-)]\Big)^2 \\
&= \underbrace{\sigma_b^2 + \sigma_w^2 \sum_k \mathbb{E}\big[\phi(s_k^-)^2\big] + \mu_w^2 \sum_{k,k'} \mathrm{Cov}[\phi(s_k^-), \phi(s_{k'}^-)]}_{\mathrm{Var}[s_i]} + \mathbb{E}[s_i]^2, && (17)
\end{aligned}$$

where

$$\mathrm{Cov}[\phi(s_k^-), \phi(s_{k'}^-)] = \mathbb{E}[\phi(s_k^-)\phi(s_{k'}^-)] - \mathbb{E}[\phi(s_k^-)]\,\mathbb{E}[\phi(s_{k'}^-)]$$

is the covariance.

Unlike in the traditional setting in section A.1, the pre-activations are not uncorrelated. This means that we also need to include the propagation of covariance in our generalised signal propagation theory. We do this by considering the second mixed moment of the pre-activations:

$$\begin{aligned}
\mathbb{E}_{i \neq j}[s_i s_j] &= \mathbb{E}\Big[\Big(b_i + \sum_k w_{ik}\phi(s_k^-)\Big)\Big(b_j + \sum_k w_{jk}\phi(s_k^-)\Big)\Big] \\
&= \mathbb{E}[b_i b_j] + 2\,\mathbb{E}\Big[b_i \sum_k w_{jk}\phi(s_k^-)\Big] + \mathbb{E}\Big[\sum_{k,k'} w_{ik} w_{jk'} \phi(s_k^-)\phi(s_{k'}^-)\Big] \\
&= \mu_b^2 + 2\mu_b\mu_w \sum_k \mathbb{E}[\phi(s_k^-)] + \mu_w^2 \sum_{k,k'} \mathbb{E}[\phi(s_k^-)\phi(s_{k'}^-)] \\
&= \Big(\mu_b + \mu_w \sum_k \mathbb{E}[\phi(s_k^-)]\Big)^2 + \mu_w^2 \sum_{k,k'} \mathrm{Cov}[\phi(s_k^-), \phi(s_{k'}^-)] \\
&= \underbrace{\mu_w^2 \sum_{k,k'} \mathrm{Cov}[\phi(s_k^-), \phi(s_{k'}^-)]}_{\mathrm{Cov}_{i \neq j}[s_i, s_j]} + \mathbb{E}[s_i]\,\mathbb{E}[s_j]. \qquad (18)
\end{aligned}$$

Note that this expression also appears in the variance formula (eq. 17).

We can again assume that pre-activations are (approximately) identically distributed to conclude that

$$\forall n \in \{1, 2\} : \forall k : \mathbb{E}\big[\phi(s_k^-)^n\big] = \mathbb{E}\big[\phi(s_1^-)^n\big].$$

Putting everything together, we arrive at our generalised signal propagation results from equations (5) and (6):

$$\begin{aligned}
\mathbb{E}[s_i] &= N\mu_w\,\mathbb{E}[\phi(s_1^-)] + \mu_b \\
\mathbb{E}[s_i s_j] &= \underbrace{\delta_{ij}\,\big(N\sigma_w^2\,\mathbb{E}\left[\phi(s_1^-)^2\right] + \sigma_b^2\big) + \mu_w^2 \sum_{k,k'} \mathrm{Cov}[\phi(s_k^-), \phi(s_{k'}^-)]}_{\mathrm{Cov}[s_i, s_j]} + \mathbb{E}[s_i]\,\mathbb{E}[s_j].
\end{aligned}$$

Here, we combined equations (17) and (18) into a single expression using the Kronecker delta, δ_{ij}.

The identical distribution of the pre-activations also induces a particular structure in the covariance matrix. Because the diagonal entries represent the variance, they must have the same value. The off-diagonal entries happen to be the same — but typically different from the variance — as well. After all, eq. (18) is also independent of indices i and j. As a result, the sum over entries in the covariance matrix can be written in terms of these two values:

$$\begin{aligned}
\sum_{k,k'} \mathrm{Cov}[\phi(s_k^-), \phi(s_{k'}^-)] &= \sum_k \Big(\mathrm{Var}[\phi(s_k^-)] + \sum_{k' \neq k} \mathrm{Cov}[\phi(s_k^-), \phi(s_{k'}^-)]\Big) \\
&= N\big(\mathrm{Var}[\phi(s_1^-)] + (N-1)\,\mathrm{Cov}[\phi(s_1^-), \phi(s_2^-)]\big). \qquad (19)
\end{aligned}$$

A.3 Activation Function Kernels

Activation functions are not directly affected by the positive weights in ICNNs. Nevertheless, the propagation through non-linear activation functions also has to be reconsidered for our generalised theory. The main reason is the difference in covariance structure between the traditional and our generalised signal propagation.

When considering the propagation through non-linearities, the pre-activations are assumed to be Gaussian random variables. This is typically justified by the central limit theorem, which applies to sums of (weakly) independent random variables. In the traditional theory (sec. A.1), where $\mu_w = \mu_b = 0$, it can be verified that pre-activations are uncorrelated:

$$\begin{aligned}
\mathbb{E}_{i \neq j}[s_i s_j] &= \mathbb{E}\Big[\Big(b_i + \sum_k w_{ik}\phi(s_k^-)\Big)\Big(b_j + \sum_k w_{jk}\phi(s_k^-)\Big)\Big] \\
&= \mathbb{E}[b_i b_j] + 2\,\mathbb{E}\Big[b_i \sum_k w_{jk}\phi(s_k^-)\Big] + \mathbb{E}\Big[\sum_{k,k'} w_{ik} w_{jk'} \phi(s_k^-)\phi(s_{k'}^-)\Big] \\
&= \mu_b^2 + 2\mu_b\mu_w \sum_k \mathbb{E}[\phi(s_k^-)] + \mu_w^2 \sum_{k,k'} \mathbb{E}[\phi(s_k^-)\phi(s_{k'}^-)] \\
&= 0. \qquad (20)
\end{aligned}$$

Although this does not imply independence, it is often sufficient to claim weak independence. Also, empirical investigations typically confirm these assumptions (e.g. figure 1a).

In contrast, the results of our generalised theory (sec. A.2) suggest that pre-activations have non-zero correlation. Therefore, the Gaussian assumption on the pre-activations is hard to justify. Moreover, this assumption also clearly does not match empirical observations (e.g. figure 1c). Nevertheless, we adopt the Gaussian assumption for the generalised theory due to a lack of better options.

Consider some (non-linear) activation function, $\phi : \mathbb{R} \to \mathbb{R}$. The effects of ϕ on signal propagation can be captured by the matrix of mixed moments,

$$\mathbb{E}_{s_1, s_2 \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})}[\phi(s_1)\phi(s_2)], \tag{21}$$

where $\boldsymbol{\Sigma} = \left(\begin{smallmatrix} 1 & \rho \\ \rho & 1 \end{smallmatrix}\right)\sigma^2$. Here, ρ is the pair-wise correlation and σ^2 the variance of pre-activations. This matrix of mixed moments can also be used to define kernels (see e.g. Cho and Saul, 2009). Therefore, we will refer to this matrix as the *kernel* of the activation function. Note that for traditional signal propagation theory, only the diagonal is relevant.

For the sake of example, we show how to compute the kernel for the leaky ReLU function (Maas et al., 2013),

$$\mathrm{LReLU} : \mathbb{R} \to \mathbb{R} : x \mapsto \mathrm{LReLU}(x\,;\alpha) \begin{cases} x & x \geq 0 \\ \alpha x & x < 0 \end{cases}.$$

We choose the leaky ReLU function because it allows for analytical solutions. Throughout these computations, we assume pre-activations to have zero mean. Moreover, the covariance between any two features is assumed to be $\boldsymbol{\Sigma} = \left(\begin{smallmatrix} 1 & \rho \\ \rho & 1 \end{smallmatrix}\right)\sigma^2$. Here ρ is the correlation and σ^2 is the variance of pre-activations.

Using the kernel for ReLU from (Daniely et al., 2016; Cho and Saul, 2009), the covariance for leaky ReLU is given by

$$\begin{aligned}
&\mathbb{E}[\,\mathrm{LReLU}(s_1\,;\alpha)\,\mathrm{LReLU}(s_2\,;\alpha)] \\
&\quad= \int_{-\infty}^{0}\int_{-\infty}^{0} \alpha^2 s_1 s_2\, p_{\mathcal{N}}(s_1, s_2\,; \mathbf{0}, \boldsymbol{\Sigma})\,\mathrm{d}s_1\,\mathrm{d}s_2 + \int_{-\infty}^{0}\int_{0}^{\infty} \alpha s_1 s_2\, p_{\mathcal{N}}(s_1, s_2\,; \mathbf{0}, \boldsymbol{\Sigma})\,\mathrm{d}s_1\,\mathrm{d}s_2 \\
&\qquad\quad + \int_{0}^{\infty}\int_{-\infty}^{0} \alpha s_1 s_2\, p_{\mathcal{N}}(s_1, s_2\,; \mathbf{0}, \boldsymbol{\Sigma})\,\mathrm{d}s_1\,\mathrm{d}s_2 + \int_{0}^{\infty}\int_{0}^{\infty} s_1\, s_2\, p_{\mathcal{N}}(s_1, s_2\,; \mathbf{0}, \boldsymbol{\Sigma})\,\mathrm{d}s_1\,\mathrm{d}s_2 \\
&\quad= (1 + \alpha^2)\,\mathbb{E}_{\mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})}[\mathrm{ReLU}(s_1)\,\mathrm{ReLU}(s_2)] - 2\alpha\,\mathbb{E}_{\mathcal{N}(\mathbf{0}, \bar{\boldsymbol{\Sigma}})}[\mathrm{ReLU}(s_1)\,\mathrm{ReLU}(s_2)] \\
&\quad= (1 + \alpha^2)\frac{\sigma^2}{2\pi}\Big(\sqrt{1 - \rho^2} + \rho \arccos(-\rho)\Big) - 2\alpha\frac{\sigma^2}{2\pi}\Big(\sqrt{1 - \rho^2} - \rho \arccos(\rho)\Big) && (22) \\
&\quad= (1 + \alpha^2)\frac{\sigma^2}{2\pi}\Big(\sqrt{1 - \rho^2} + \rho \arccos(-\rho)\Big) - 2\alpha\frac{\sigma^2}{2\pi}\Big(\sqrt{1 - \rho^2} - \rho\pi + \rho \arccos(-\rho)\Big) \\
&\quad= (1 - \alpha)^2\,\frac{\sigma^2}{2\pi}\Big(\sqrt{1 - \rho^2} + \rho \arccos(-\rho)\Big) + \alpha\sigma^2\rho, && (23)
\end{aligned}$$

where $\bar{\boldsymbol{\Sigma}} = \left(\begin{smallmatrix} 1 & -\rho \\ -\rho & 1 \end{smallmatrix}\right)\sigma^2$ appears as a side-effect of flipping the integration boundaries. Note that this expression also captures the squared mean ($\rho = 0$) and second raw moment ($\rho = 1$).

A.4 Generalised Backward Propagation

The generalisation to non-zero mean also applies to backpropagation, where signals are the so-called deltas, the derivatives of the loss, L, w.r.t. the pre-activations. These deltas can be defined using the $\mathbb{R}^M \to \mathbb{R}^N$ mapping:

$$\boldsymbol{\delta}^- = \frac{\partial L}{\partial \boldsymbol{s}^-} = \phi'\big(\boldsymbol{s}^-\big)\boldsymbol{W}^\mathsf{T}\boldsymbol{\delta}. \tag{24}$$

Here, $\boldsymbol{W}$ are the weights of the layer that takes $\phi\big(\boldsymbol{s}^-\big)$ as inputs. Note that

The first two (raw) moments of the deltas from eq. (24) are:

$$\begin{aligned}
\mathbb{E}[\delta_j^-] &= \mathbb{E}\Big[\phi'(s_j^-)\sum_k \delta_k w_{kj}\Big] \\
&= \mu_w\,\mathbb{E}[\phi'(s_j^-)]\sum_k \mathbb{E}[\delta_k] \qquad (25)\\
\mathbb{E}\big[(\delta_j^-)^2\big] &= \mathbb{E}\bigg[\Big(\phi'(s_j^-)\sum_k \delta_k w_{kj}\Big)^2\bigg] \\
&= \mathbb{E}\Big[\phi'(s_j^-)^2\sum_{k,k'} \delta_k\delta_{k'} w_{kj}w_{k'j}\Big] \\
&= (\sigma_w^2+\mu_w^2)\,\mathbb{E}\big[\phi'(s_j^-)^2\big]\sum_k \mathbb{E}\big[\delta_k^2\big] + \mu_w^2\,\mathbb{E}\big[\phi'(s_j^-)^2\big]\sum_{k,k'\neq k}\mathbb{E}[\delta_k\delta_{k'}] \\
&= \sigma_w^2\,\mathbb{E}\big[\phi'(s_j^-)^2\big]\sum_k \mathbb{E}\big[\delta_k^2\big] + \mu_w^2\,\mathbb{E}\big[\phi'(s_j^-)^2\big]\sum_{k,k'}\mathbb{E}[\delta_k\delta_{k'}]. \qquad (26)
\end{aligned}$$

The second mixed moment can be derived in a similar way:

$$\begin{aligned}
\mathbb{E}_{i\neq j}[\delta_i^-\delta_j^-] &= \mathbb{E}\bigg[\Big(\phi'(s_i^-)\sum_k \delta_k w_{ki}\Big)\Big(\phi'(s_j^-)\sum_k \delta_k w_{kj}\Big)\bigg] \\
&= \mathbb{E}\Big[\phi'(s_i^-)\phi'(s_j^-)\sum_{k,k'} w_{ki}w_{k'j}\delta_k\delta_{k'}\Big] \\
&= \mu_w^2\,\mathbb{E}[\phi'(s_i^-)\phi'(s_j^-)]\sum_{k,k'}\mathbb{E}[\delta_k\delta_{k'}]. \qquad (27)
\end{aligned}$$

From the forward pass (see section A.2), we know that the moments of pre-activations are independent of their index, i.e. $\forall k : \mathbb{E}\big[(s_k^-)^n\big] = \mathbb{E}\big[(s_1^-)^n\big]$. As a result, we also have $\mathbb{E}\big[\phi'(s_k^-)^n\big] = \mathbb{E}\big[\phi'(s_1^-)^n\big]$ and therefore

$$\forall n \in \{1,2\} : \forall k : \mathbb{E}\big[(\delta_k^-)^n\big] = \mathbb{E}\big[(\delta_1^-)^n\big].$$

Putting everything together, we obtain the generalised signal propagation for the deltas during back-propagation:

$$\begin{aligned}
\mathbb{E}[\delta_j^-] &= M\mu_w\,\mathbb{E}[\phi'(s_1^-)]\,\mathbb{E}[\delta_1] \\
\mathbb{E}[\delta_i^-\delta_j^-] &= \delta_{ij}\,M\sigma_w^2\,\mathbb{E}\big[\phi'(s_1^-)^2\big]\,\mathbb{E}\big[\delta_1^2\big] + \mu_w^2\,\mathbb{E}[\phi'(s_i^-)\phi'(s_j^-)]\sum_{k,k'}\mathbb{E}[\delta_k\delta_{k'}],
\end{aligned}$$

where δ_{ij} is the Kronecker delta.

The covariance for the derivative of LReLU under the same assumptions as in section A.3 would be given by

$$\begin{aligned}
&\mathbb{E}[\,\mathrm{LReLU}'(s_1\,;\alpha)\,\mathrm{LReLU}'(s_2\,;\alpha)] \\
&\quad= \int_{-\infty}^{0}\int_{-\infty}^{0}\alpha^2 p_{\mathcal{N}}(s_1,s_2\,;\mathbf{0},\mathbf{\Sigma})\,\mathrm{d}s_1\,\mathrm{d}s_2 + \int_{-\infty}^{0}\int_{0}^{\infty}\alpha\, p_{\mathcal{N}}(s_1,s_2\,;\mathbf{0},\mathbf{\Sigma})\,\mathrm{d}s_1\,\mathrm{d}s_2 \\
&\qquad+ \int_{0}^{\infty}\int_{-\infty}^{0}\alpha\, p_{\mathcal{N}}(s_1,s_2\,;\mathbf{0},\mathbf{\Sigma})\,\mathrm{d}s_1\,\mathrm{d}s_2 + \int_{0}^{\infty}\int_{0}^{\infty} p_{\mathcal{N}}(s_1,s_2\,;\mathbf{0},\mathbf{\Sigma})\,\mathrm{d}s_1\,\mathrm{d}s_2 \\
&\quad= \big(1+\alpha^2\big)\frac{1}{2\pi}\arccos(-\rho) + 2\alpha\frac{1}{2\pi}\arccos(\rho) \\
&\quad= \big(1+\alpha^2\big)\frac{1}{2\pi}\arccos(-\rho) + 2\alpha\frac{1}{2\pi}\big(\pi-\arccos(-\rho)\big) \\
&\quad= (1-\alpha)^2\frac{1}{2\pi}\arccos(-\rho) + \alpha \qquad (28)
\end{aligned}$$

A.5 Convolutional Layers

Our generalised signal propagation (see section A.2) is also applicable to convolutional layers. This section provides the derivations for a one-dimensional cross-correlation,

$$s_{ia} = b_i + \sum_{c,k} w_{ick} \phi(s^-_{c(a+k)}),$$

but can be generalised in a similar way to higher dimensional operations. The first two moments (cf. equations 16 and 17) are given by

$$\begin{aligned}
\mathbb{E}[s_{ia}] &= \mathbb{E}\Big[b_i + \sum_{c,k} w_{ick} \phi(s^-_{c(a+k)})\Big] \\
&= \mu_b + \mu_w \sum_{c,k} \mathbb{E}\big[\phi(s^-_{c(a+k)})\big] \qquad (29) \\
\mathbb{E}\big[s_{ia}^2\big] &= \mathbb{E}\Big[\Big(b_i + \sum_{c,k} w_{ick} \phi(s^-_{c(a+k)})\Big)^2\Big] \\
&= \sigma_b^2 + \mu_b^2 + 2\mu_b\mu_w \sum_{c,k} \mathbb{E}\big[\phi(s^-_{c(a+k)})\big] + \sigma_w^2 \sum_{c,k} \mathbb{E}\Big[\phi(s^-_{c(a+k)})^2\Big] + \mu_w^2 \sum_{c,k}\sum_{c',k'} \mathbb{E}\Big[\phi(s^-_{c(a+k)})\phi(s^-_{c'(a+k')})\Big] \\
&= \underbrace{\sigma_b^2 + \sigma_w^2 \sum_{c,k} \mathbb{E}\Big[\phi(s^-_{c(a+k)})^2\Big] + \mu_w^2 \sum_{c,k}\sum_{c',k'} \mathrm{Cov}\Big[\phi(s^-_{c(a+k)}), \phi(s^-_{c'(a+k')})\Big]}_{\mathrm{Var}[s_{ia}]} + \mathbb{E}[s_{ia}]^2. \qquad (30)
\end{aligned}$$

For the second mixed moment we find (cf. eq 18)

$$\begin{aligned}
\mathbb{E}_{i\neq j}\big[s_{ia}s_{j(a+d)}\big] &= \mathbb{E}\Big[\Big(b_i + \sum_{c,k} w_{ick}\phi(s^-_{c(a+k)})\Big)\Big(b_j + \sum_{c,k} w_{jck}\phi(s^-_{c(a+d+k)})\Big)\Big] \\
&= \mu_b^2 + \mu_b\mu_w \sum_{c,k} \Big(\mathbb{E}\big[\phi(s^-_{c(a+k)})\big] + \mathbb{E}\big[\phi(s^-_{c(a+d+k)})\big]\Big) \\
&\quad + \mu_w^2 \sum_{c,k}\sum_{c',k'} \mathbb{E}\big[\phi(s^-_{c(a+k)})\phi(s^-_{c'(a+d+k')})\big] \\
&= \underbrace{\mu_w^2 \sum_{c,k}\sum_{c',k'} \mathrm{Cov}\big[\phi(s^-_{c(a+k)}), \phi(s^-_{c'(a+d+k')})\big]}_{\mathrm{Cov}[s_{ia}, s_{j(a+d)}]} + \mathbb{E}[s_{ia}]\,\mathbb{E}[s_{j(a+d)}]. \qquad (31)
\end{aligned}$$

However, this mixed moment does not capture the covariance between elements in the same channel, but on different positions. Therefore, we additionally include the following result:

$$\begin{aligned}
\mathbb{E}_{d\neq 0}\big[s_{ia}s_{i(a+d)}\big] &= \mathbb{E}\Big[\Big(b_i + \sum_{c,k} w_{ick}\phi(s^-_{c(a+k)})\Big)\Big(b_i + \sum_{c,k} w_{ick}\phi(s^-_{c(a+d+k)})\Big)\Big] \\
&= \sigma_b^2 + \mu_b^2 + \mu_b\mu_w \sum_{c,k} \Big(\mathbb{E}\big[\phi(s^-_{c(a+k)})\big] + \mathbb{E}\big[\phi(s^-_{c(a+d+k)})\big]\Big) \\
&\quad + \sigma_w^2 \sum_{c,k} \mathbb{E}\big[\phi(s^-_{c(a+k)})\phi(s^-_{c(a+d+k)})\big] + \mu_w^2 \sum_{c,k}\sum_{c',k'} \mathbb{E}\big[\phi(s^-_{c(a+k)})\phi(s^-_{c'(a+d+k')})\big] \\
&= \sigma_b^2 + \sigma_w^2 \sum_{c,k} \mathbb{E}\big[\phi(s^-_{c(a+k)})\phi(s^-_{c(a+d+k)})\big] \\
&\quad + \mu_w^2 \sum_{c,k}\sum_{c',k'} \mathrm{Cov}\big[\phi(s^-_{c(a+k)}), \phi(s^-_{c'(a+d+k')})\big] + \mathbb{E}[s_{ia}]\,\mathbb{E}[s_{i(a+d)}] \qquad (32)
\end{aligned}$$

Putting these results together, we can summarise the signal propagation through a convolutional layer as follows:

$$\begin{aligned}\mathbb{E}[s_{ia}] &= \mu_b + \mu_w \sum_{c,k} \mathbb{E}\big[\phi\big(s^-_{c(a+k)}\big)\big] \\ \mathbb{E}[s_{ia} s_{j(a+d)}] &= \delta_{ij}\left(\sigma_b^2 + \sigma_w^2 \sum_{c,k} \mathbb{E}\big[\phi\big(s^-_{c(a+k)}\big)\phi\big(s^-_{c(a+d+k)}\big)\big]\right) \\ &\quad + \mu_w^2 \sum_{c,k}\sum_{c',k'} \operatorname{Cov}\big[\phi\big(s^-_{c(a+k)}\big), \phi\big(s^-_{c'(a+d+k')}\big)\big] + \mathbb{E}[s_{ia}]\,\mathbb{E}[s_{j(a+d)}]\end{aligned}$$

Here, we use the Kronecker delta to combine equations (30), (31) and (32) into one equation.

B Deriving the Initialisation

This section aims to show how the ICNN initialisation can be derived from our generalised signal propagation theory.

B.1 Zero Mean Pre-Activations

The goal of a good initialisation is to have pre-activations with similar statistics in every layer. In the traditional theory, mean and covariance are zero in every layer by default. Therefore, variance is the only statistic that needs to be controlled. This is why traditional initialisation methods (LeCun et al., 1998; Glorot and Bengio, 2010; He et al., 2015) typically focus on the standard deviation of weights.

Consider the equation for pre-activations from eq. (1). Assuming $\phi = \mathrm{LReLU}$, we can use the kernel from eq. (23). Plugging the result for $\rho = 1$ into eq. (3) we obtain the traditional variance propagation

$$\mathbb{E}\big[s_i^2\big] = N\sigma_w^2(1+\alpha^2)\frac{1}{2}\,\mathbb{E}\big[(s_1^-)^2\big].$$

By requiring that $\mathbb{E}\big[s_i^2\big] = \mathbb{E}\big[(s_1^-)^2\big] = \sigma_*^2 > 0$, we obtain the fixed point equation

$$\sigma_*^2 = N\sigma_w^2(1+\alpha^2)\frac{1}{2}\sigma_*^2,$$

which can be solved for σ_w^2 to obtain the initialisation variance for (regular) LReLU networks:

$$\sigma_w^2 = \frac{2}{1+\alpha^2}\frac{1}{N}.$$

Here, $2/(1+\alpha^2)$ is sometimes also referred to as the *gain* of the initialisation (cf. Saxe et al., 2014).

We adopt a similar approach for our generalised propagation results. In contrast to the traditional setting, however, we can not directly rely on the activation kernels. After all, the pre-activations in our general setting are not automatically zero.

Derivation of Bias Mean By setting eq. (5) to zero and solving for μ_b, we get

$$\mu_b = -N\mu_w\,\mathbb{E}[\phi(s_1^-)].$$

This also makes it easier to use the activation kernels, which assume zero mean inputs. Note, however, that the activation kernels theoretically do not apply to our setting (see sec. A.3). Nevertheless, we assume that the activation kernels approximate the actual dynamics well enough. Plugging in the square root of the LReLU kernel (eq. 23) with $\rho = 0$, we obtain the bias mean for $\phi = \mathrm{LReLU}$:

$$\begin{aligned}\mu_b &= -N\mu_w(1-\alpha)\sqrt{\frac{1}{2\pi}\Big(\mathbb{E}\big[(s_1^-)^2\big] - \mathbb{E}[s_1^-]^2\Big)} \\ &= -N\mu_w(1-\alpha)\sqrt{\frac{1}{2\pi}\,\mathbb{E}\big[(s_1^-)^2\big]}. \qquad (33)\end{aligned}$$

B.2 Variance and Covariance Fixed Points

To obtain fixed points for the variance and covariance, we start from equations (17) and (18), respectively. Assuming identically distributed pre-activations, we can use equations (15) and (19), to get expressions without sums:

$$\mathbb{E}\big[s_i^2\big] = \sigma_b^2 + \sigma_w^2 N\,\mathbb{E}\big[\phi(s_1^-)^2\big] + \mathbb{E}_{i\neq j}[s_i s_j]$$
$$\mathbb{E}_{i\neq j}[s_i s_j] = \mu_w^2 N\Big(\operatorname{Var}\big[\phi(s_1^-)^2\big] + (N-1)\operatorname{Cov}\big[\phi(s_1^-), \phi(s_2^-)\big]\Big)$$

For $\phi = \mathrm{LReLU}$, we can use the kernel from eq. (23) with $\rho = 1$ to work out the moments for the variance,

$$\mathbb{E}\big[s_i^2\big] = \sigma_b^2 + \sigma_w^2 N(1+\alpha^2)\frac{1}{2}\,\mathbb{E}\big[(s_1^-)^2\big] + \mathbb{E}_{i\neq j}[s_i s_j], \tag{34}$$

and with $\rho = c^- = \operatorname{Corr}[s_1^-, s_2^-] = \mathbb{E}[s_1^- s_2^-]/\,\mathbb{E}\big[(s_1^-)^2\big]$ to obtain the covariance

$$\begin{aligned}\mathbb{E}_{i\neq j}[s_i s_j] &= \mu_w^2 N\Big(\mathbb{E}\big[\mathrm{LReLU}(s_1^-)^2\big] + (N-1)\,\mathbb{E}\big[\mathrm{LReLU}(s_1^-)\,\mathrm{LReLU}(s_2^-)\big] - N\,\mathbb{E}\big[\mathrm{LReLU}(s_1^-)\big]^2\Big)\\
&= \mu_w^2 N\Bigg(\frac{1+\alpha^2}{2}\,\mathbb{E}\big[(s_1^-)^2\big] - N\frac{(1-\alpha)^2}{2\pi}\,\mathbb{E}\big[(s_1^-)^2\big]\\
&\qquad + (N-1)\bigg(\frac{(1-\alpha)^2}{2\pi}\,\mathbb{E}\big[(s_1^-)^2\big]\Big(\sqrt{1-(c^-)^2} + c^-\arccos\big(-c^-\big)\Big) + \alpha\,\mathbb{E}\big[(s_1^-)^2\big]c^-\bigg)\Bigg)\\
&= \mu_w^2\frac{N}{2\pi}\,\mathbb{E}\big[(s_1^-)^2\big]\Bigg((1+\alpha^2)\pi - N(1-\alpha)^2\\
&\qquad + (N-1)\bigg((1-\alpha)^2\Big(\sqrt{1-(c^-)^2} + c^-\arccos\big(-c^-\big)\Big) + 2\pi\alpha c^-\bigg)\Bigg)\end{aligned}$$

For the sake of readability, we introduce $f_\mathrm{c} : [-1, 1] \to \mathbb{R}$ to denote

$$f_\mathrm{c}(\rho) = \frac{N}{2\pi}\Bigg((1+\alpha^2)\pi - N(1-\alpha)^2 + (N-1)\bigg((1-\alpha)^2\Big(\sqrt{1-\rho^2} + \rho\arccos\big(-\rho\big)\Big) + 2\pi\alpha\rho\bigg)\Bigg), \tag{35}$$

such that the covariance can be written more compactly as

$$\mathbb{E}_{i\neq j}[s_i s_j] = \mu_w^2\,\mathbb{E}\big[(s_1^-)^2\big]f_\mathrm{c}\big(\operatorname{Corr}[s_1^-, s_2^-]\big) \tag{36}$$

Derivation of Bias and Weight Variances The fixed point equation for the variance (eq. 34) with $\mathbb{E}\big[s_i^2\big] = \mathbb{E}\big[(s_1^-)^2\big] = \sigma_*^2$ can be written as

$$\sigma_*^2 = \sigma_b^2 + \sigma_w^2 N(1+\alpha^2)\frac{1}{2}\sigma_*^2 + \sigma_*^2\mu_w^2 f_\mathrm{c}\big(\operatorname{Corr}[s_1^-, s_2^-]\big),$$

where f_c is taken from equation (35). Note that we retrieve eq. (10) in the main text from this result by setting $\alpha = 0$ and replacing $\mu_w^2 f_\mathrm{c}\big(\operatorname{Corr}[s_1^-, s_2^-]\big)$ by the correlation fixed point, ρ_*. We elaborate on the latter when discussing the fixed point equation for the correlation. Solving the fixed point equation for σ_w^2, we find

$$\sigma_w^2 = \frac{2}{1+\alpha^2}\frac{1}{N}\Big(1 - \frac{\sigma_b^2}{\sigma_*^2} - \mu_w^2 f_\mathrm{c}\big(\operatorname{Corr}[s_1^-, s_2^-]\big)\Big). \tag{37}$$

By initialising the bias vector with a constant value, such that $\sigma_b^2 = 0$, the expression above becomes independent of the fixed point σ_*^2:

$$\sigma_w^2 = \frac{2}{1+\alpha^2}\frac{1}{N}\Big(1 - \mu_w^2 f_\mathrm{c}\big(\operatorname{Corr}[s_1^-, s_2^-]\big)\Big). \tag{38}$$

Using similar substitutions as before, we obtain the result from the main text (eq. 11). A further simplification is possible by revisiting eq. (36) and observing that if $\mathbb{E}\big[s_i^2\big] = \mathbb{E}\big[(s_1^-)^2\big]$, we have $\operatorname{Corr}[s_i, s_j] = \mu_w^2 f_\mathrm{c}\big(\operatorname{Corr}[s_1^-, s_2^-]\big)$, such that

$$\sigma_w^2 = \frac{2}{1+\alpha^2}\frac{1}{N}\Big(1 - \operatorname{Corr}[s_i, s_j]\Big). \tag{39}$$

Derivation of Weight Mean Because eq. (36) is mainly a function of correlation, we consider a fixed point equation in terms of correlation rather than covariance. Concretely, we set $\mathrm{Corr}[s_i, s_j] = \mathrm{Corr}[s_1^-, s_2^-] = \rho_*$ to obtain the fixed point equation

$$\rho_* = \mu_w^2 \frac{\mathbb{E}\big[(s_1^-)^2\big]}{\mathbb{E}\big[s_1^2\big]} f_\mathrm{c}(\rho_*),$$

where we used the identical distribution assumption as follows

$$\mathrm{Corr}[s_i, s_j] = \frac{\mathbb{E}_{i \neq j}\big[s_i s_j\big]}{\sqrt{\mathbb{E}\big[s_i^2\big]\,\mathbb{E}\big[s_j^2\big]}} = \frac{\mathbb{E}_{i \neq j}\big[s_i s_j\big]}{\mathbb{E}\big[s_1^2\big]}.$$

Note that for $\alpha = 0$ and $\mathbb{E}\big[s_1^2\big] = \mathbb{E}\big[(s_1^-)^2\big]$, this fixed point equation corresponds to eq. (9) in the main text. Solving the fixed point equation for μ_w^2, we find

$$\mu_w^2 = \frac{\mathbb{E}\big[s_1^2\big]}{\mathbb{E}\big[(s_1^-)^2\big]} \rho_*\, f_\mathrm{c}(\rho_*)^{-1}. \tag{40}$$

Under similar conditions as previously stated, this gives us the result from the main text (eq. 12). We can additionally plug in eq. (34) into this result to get rid of $\mathbb{E}\big[s_1^2\big]$:

$$\mu_w^2 = \left(\frac{\sigma_b^2}{\mathbb{E}\big[(s_1^-)^2\big]} + \sigma_w^2 N \frac{1+\alpha^2}{2} + \rho_*\right) \rho_*\, f_\mathrm{c}(\rho_*)^{-1}.$$

Assuming $\sigma_b^2 = 0$ again, we obtain an expression that is independent of $\mathbb{E}\big[(s_1^-)^2\big]$:

$$\mu_w^2 = \left(\sigma_w^2 N \frac{1+\alpha^2}{2} + \rho_*\right) \rho_*\, f_\mathrm{c}(\rho_*)^{-1}. \tag{41}$$

B.3 Initialisation Parameters

Putting everything together from sections B.1 and B.2, we obtain an initialisation for fully-connected layers with features produced by a LReLU non-linearity. The initialisation parameters are (see equations 33, 38 and 41):

$$\mu_w = \pm\sqrt{\left(\sigma_w^2 N \frac{1+\alpha^2}{2} + \rho_*\right) \rho_*\, f_\mathrm{c}(\rho_*)^{-1}} \qquad \sigma_w^2 = \frac{2}{1+\alpha^2} \frac{1}{N}\Big(1 - \mu_w^2 f_\mathrm{c}\big(\mathrm{Corr}[s_1^-, s_2^-]\big)\Big)$$

$$\mu_b = -N\mu_w(1-\alpha)\sqrt{\frac{1}{2\pi}\,\mathbb{E}\big[(s_1^-)^2\big]} \qquad \sigma_b^2 = 0.$$

Note that the mean for the weights is the only result that directly depends on the fixed point. However, because these results are strongly connected, they have to be considered simultaneously. Practically, this means that we have to focus on the joint fixed point (σ_*, ρ_*) instead of the individual parts. Considering equations (39) and (40), it is clear that the joint fixed point can be obtained with the following formulations

$$\mu_w = \pm\sqrt{\rho_*\, f_\mathrm{c}(\rho_*)^{-1}} \qquad \sigma_w^2 = \frac{2}{1+\alpha^2}\frac{1}{N}(1-\rho_*) \tag{42}$$

$$\mu_b = -N\mu_w(1-\alpha)\sqrt{\frac{\sigma_*^2}{2\pi}} \qquad \sigma_b^2 = 0. \tag{43}$$

Our initialisation parameters still depend on the choice of the joint fixed point σ_*^2 and ρ_*. Normally, $\sigma_*^2 = 1$ is assumed because that is how the input data is typically normalised. In general, we can assume σ_*^2 to be the variance of the input data. The ideal scenario for correlation is to have $\rho_* = 0$, i.e. uncorrelated features. However, when plugging this into our initialisation parameters, we retrieve the traditional setting from section A.1, where $\mu_w = 0$. In other words, uncorrelated features are only possible if $\mu_w = 0$. Also, $|\rho_*| < 1$ is desirable, because $\rho = 1$ practically corresponds to all values being the same. Therefore, we (arbitrarily) choose to set $\rho_* = \frac{1}{2}$. One argument in favour of this choice is that eq. (35) has a relatively nice solution

$$\begin{aligned}
f_\mathrm{c}\Big(\frac{1}{2}\Big) &= \frac{N}{2\pi}\bigg((1+\alpha^2)\pi - N(1-\alpha)^2 + (N-1)\bigg((1-\alpha)^2\Big(\frac{\sqrt{3}}{2}+\frac{\pi}{3}\Big)+\pi\alpha\bigg)\bigg) \\
&= \frac{N}{12\pi}\Big(6\big(1+\alpha^2\big)\pi - 6N(1-\alpha)^2 + (N-1)\Big((1-\alpha)^2\big(3\sqrt{3}+2\pi\big)+6\pi\alpha\Big)\Big) \\
&= \frac{N}{12\pi}\Big(6(1-\alpha)^2\pi + 12\alpha\pi - 6N(1-\alpha)^2 + (N-1)(1-\alpha)^2\big(3\sqrt{3}+2\pi\big)+6(N-1)\pi\alpha\Big) \\
&= \frac{N}{12\pi}\Big((1-\alpha)^2\Big(6\pi - 6N + (N-1)\big(3\sqrt{3}+2\pi\big)\Big)+6(N+1)\pi\alpha\Big) \\
&= \frac{N}{12\pi}\Big((1-\alpha)^2\Big(6(\pi-1) + (N-1)\big(3\sqrt{3}+2\pi-6\big)\Big)+6(N+1)\pi\alpha\Big).
\end{aligned}$$

Filling out the fixed point $(\sigma_*^2, \rho_*) = (1, \frac{1}{2})$, we obtain the final initialisation parameters for ICNNs with LReLU activation functions:

$$\begin{aligned}
\mu_w &= \pm\sqrt{\frac{6\pi}{N\Big((1-\alpha)^2\Big(6\pi-6N+(N-1)\big(3\sqrt{3}+2\pi\big)\Big)+6(N+1)\pi\alpha\Big)}} & \sigma_w^2 &= \frac{1}{1+\alpha^2}\frac{1}{N} \\
\mu_b &= \mp\sqrt{\frac{3N(1-\alpha)^2}{(1-\alpha)^2\Big(6\pi-6N+(N-1)\big(3\sqrt{3}+2\pi\big)\Big)+6(N+1)\pi\alpha}} & \sigma_b^2 &= 0.
\end{aligned}$$

Note again that these results correspond to the results in the main paper where $\alpha = 0$.

B.4 Fixed Point Stability

The goal of this section is to establish the stability of the joint fixed point we used for our initialisation from section B.3. Therefore, we analyse the Jacobian of the mapping

$$f_* : \mathbb{R}^+ \times [-1,1] \to \mathbb{R}^+ \times [-1,1] : (\sigma^2, \rho) \mapsto \Big(\sigma_b^2 + \sigma_w^2 N\big(1+\alpha^2\big)\frac{1}{2}\sigma^2 + \sigma^2\mu_w^2 f_\mathrm{c}(\rho), \mu_w^2 f_\mathrm{c}(\rho)\Big),$$

which combines equations (34) and (36) with f_c from eq. (35). Filling out our initialisation parameters from equations (42) and (43), we obtain

$$f_*(\sigma^2, \rho) = \bigg(1 - \rho_* + \sigma^2\rho_*\frac{f_\mathrm{c}(\rho)}{f_\mathrm{c}(\rho_*)}, \rho_*\frac{f_\mathrm{c}(\rho)}{f_\mathrm{c}(\rho_*)}\bigg). \tag{44}$$

The stability of f_* can be analysed by computing its Jacobian

$$\boldsymbol{\mathcal{J}}_{f_*}(\sigma^2, \rho) = \begin{bmatrix} \rho_*\frac{f_\mathrm{c}(\rho)}{f_\mathrm{c}(\rho_*)} & \sigma^2\rho_*\frac{f_\mathrm{c}'(\rho)}{f_\mathrm{c}(\rho_*)} \\ 0 & \rho_*\frac{f_\mathrm{c}'(\rho)}{f_\mathrm{c}(\rho_*)} \end{bmatrix},$$

where

$$\begin{aligned}
f_\mathrm{c}'(\rho) &= \frac{N}{2\pi}(N-1)(1-\alpha)^2\frac{\partial}{\partial\rho}\Big(\sqrt{1-\rho^2} + \rho\arccos(-\rho)\Big) + \frac{N}{2\pi}(N-1)2\pi\alpha \\
&= \frac{N}{2\pi}(N-1)(1-\alpha)^2\left(\frac{-\rho}{\sqrt{1-\rho^2}} + \arccos(-\rho) + \rho\frac{1}{\sqrt{1-\rho^2}}\right) + N(N-1)\alpha \\
&= \frac{N}{2\pi}(N-1)(1-\alpha)^2\arccos(-\rho) + N(N-1)\alpha.
\end{aligned}$$

The eigenvalues of the Jacobian are the roots of the characteristic polynomial

$$|\lambda\boldsymbol{I} - \boldsymbol{\mathcal{J}}_{f_*}(\sigma^2, \rho)| = \bigg(\lambda - \rho_*\frac{f_\mathrm{c}(\rho)}{f_\mathrm{c}(\rho_*)}\bigg)\bigg(\lambda - \rho_*\frac{f_\mathrm{c}'(\rho)}{f_\mathrm{c}(\rho_*)}\bigg),$$

which happen to be the entries on the diagonal:

$$\lambda_1 = \rho_* \frac{f_c(\rho)}{f_c(\rho_*)} \qquad\qquad \lambda_2 = \rho_* \frac{f_c'(\rho)}{f_c(\rho_*)}.$$

Evaluating the eigenvalues of the Jacobian at the joint fixed point, (σ_*^2, ρ_*), we find

$$\lambda_1 = \rho_* \qquad\qquad \lambda_2 = \rho_* \frac{f_c'(\rho_*)}{f_c(\rho_*)}.$$

Setting $\rho_* = \frac{1}{2}$ and $\alpha = 0$, this becomes

$$\lambda_2 = \frac{(N-1)2\pi}{6\pi - 6 + (N-1)(3\sqrt{3} + 2\pi - 6)},$$

which is only less than one if

$$N < 1 + \frac{2\pi - 2}{2 - \sqrt{3}} \approx 17.$$

As a result, the fixed point $(\sigma_*^2, \rho_*) = (1, \frac{1}{2})$ is not stable in practical settings.

C Additional Experiments

This section presents experimental details and results that did not make it into the main paper.

C.1 Computing resources and budget

For our experiments, we had access to server infrastructure with multiple GPUs. We mainly used NVIDIA Titan V and NVIDIA RTX 2080Ti graphics cards with approximately 12 GB of memory for the computer vision experiments. Experiments were run in parallel on up to 8 GPUs. To maximise GPU usage, repetitions also ran in parallel on single cards. The Tox21 experiments were run on a desktop PC with one NVIDIA GTX 1070Ti, which has 8 GB of on-board memory.

A single run for the experiments in Fig. 2, Fig. 5, and Fig. 10 took approximately one minute and used no more than 2 GB of GPU memory on MNIST. For CIFAR10 and CIFAR100, a single run took up to four minutes and used up to 4 GB of GPU memory. As a result, an (over-)estimate for the compute time for Figure 2 is 6 hours or 18 hours for Figure 5. All the repetitions for the Tox21 results in Table 1, were obtained in 2 hours — i.e. 30 minutes per model — using a little over 500 MB of GPU memory.

The hyper-parameter search that lead to Table 3, required the most compute. For these estimates, we processed the timestamps for each run in our logs. Since we did not log memory consumption, we can not reliably report this data. The MNIST non-convex baseline model search required approximately 24 hours with a median run-time of less than four minutes. The search for the non-convex model on CIFAR10 required 9 hours with a median run-time of approximately two minutes. Note that five runs ran simultaneously (on the same card) for MNIST, but only three runs ran at the same time for CIFAR10. The search for "ICNN" took 11 hours for MNIST and 44 hours for CIFAR10. The search for "ICNN + skip" took 14 hours on both MNIST and CIFAR10. The search for "ICNN + init" took 45 hours on MNIST and 40 hours for CIFAR10. This gives a total of 201 hours wall-clock computation time for the hyper-parameter search. Training the resulting ten repetitions for the eight resulting models required an additional 2.5 hours for MNIST and 3.5 hours for CIFAR10 to obtain figure 3.

C.2 Computer Vision Benchmarking Datasets

In this section, we provide additional details on the experiments on computer vision datasets. Figure 5 extends Figure 2 from the main paper with learning curves for networks with three and seven hidden layers. Table 3 lists the best hyper-parameters for the search behind Figure 3. Table 2 lists the options for the different hyper-parameters that we used.

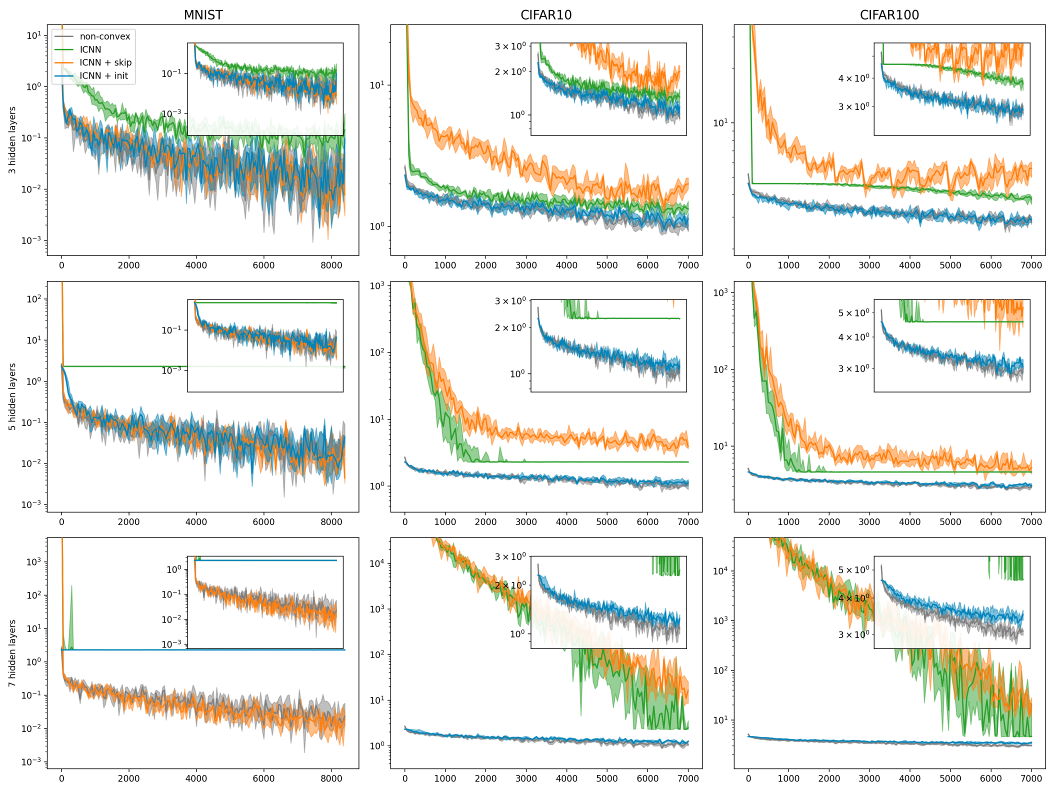


Figure 5: Training loss curves of ICNN variants with the same architecture with three, five and seven hidden layers on the MNIST, CIFAR10 and CIFAR100 datasets. "ICNN" is an Input-Convex Neural Network with He initialisation. "ICNN + skip": same settings but with skip-connections. "ICNN + init": our principled initialisation for ICNNs w/o skip-connections. "non-convex": a traditional non-convex network. The median performance over ten runs is displayed, shaded regions represent the inter-quartile range. The inset figures provide a view of the loss curves zoomed in.

Table 2: Hyper-parameter search space for results in Table 3

pre-processing		{none, PCA, ZCA}
learning rate		$\{10^{-2}, 10^{-3}, 10^{-4}\}$
L_2 regularisation		$\{0, 10^{-2}\}$
	1 hidden	{(1k), (10k)}
layer-widths	2 hidden	{(1k, 1k), (1k, 10k), (10k, 1k), (10k, 10k)}
	3 hidden	{(1k, 1k, 1k), (1k, 10k, 1k), (10k, 1k, 10k), (10k, 10k, 10k)}

Table 3: Optimal hyper-parameters for each configuration in Figure 3. The epoch column reports the epoch in which the validation accuracy was highest.

		architecture	η	L_2	pre-processing	epoch	accuracy
MNIST	non-convex	(784, 1 000, 1 000, 10)	1e-4	0.01	ZCA	25	98.62%
	ICNN	(784, 10 000, 10)	1e-4	0.01	PCA	24	98.28%
	ICNN + skip	(784, 10 000, 10)	1e-3	0.00	none	24	98.30%
	ICNN + init	(784, 10 000, 10)	1e-4	0.00	ZCA	15	98.27%
CIFAR10	non-convex	(3072, 10 000, 1 000, 10)	1e-4	0.01	ZCA	18	55.92%
	ICNN	(3072, 1 000, 10)	1e-4	0.00	none	19	54.74%
	ICNN + skip	(3072, 1 000, 10)	1e-4	0.01	none	19	52.47%
	ICNN + init	(3072, 1 000, 10)	1e-4	0.00	none	17	55.64%

C.3 Random Bias Initialisation

In section 3.3, we chose to set $\sigma_b^2 = 0$ for simplicity. However, it is also possible to derive an initialisation with $\sigma_b^2 > 0$. Practically, this means that we can initialise the biases with random samples from a Gaussian distribution with mean μ_b as in eq. (8) and $\sigma_b^2 > 0$. However, due to eq. (10) we know that this will also affect the initialisation of the synaptic weight parameters.

For the variance of the weight variance to be positive we must consider the following inequality (from eq. 37):

$$\begin{aligned}\sigma_w^2 = \frac{2}{1+\alpha^2}\frac{1}{N}\Big(1 - \frac{\sigma_b^2}{\sigma_*^2} - \mu_w^2 f_\mathrm{c}\big(\mathrm{Corr}[s_1^-, s_2^-]\big)\Big) > 0 \\ \Leftrightarrow 1 - \mu_w^2 f_\mathrm{c}\big(\mathrm{Corr}[s_1^-, s_2^-]\big) > \frac{\sigma_b^2}{\sigma_*^2} \\ \Leftrightarrow \sigma_*^2\Big(1 - \mu_w^2 f_\mathrm{c}\big(\mathrm{Corr}[s_1^-, s_2^-]\big)\Big) > \sigma_b^2.\end{aligned}$$

Assuming we are working in a fixed point regime, we obtain the following simplified constraint: $0 \leq \sigma_b^2 < \sigma_*^2(1-\rho_*)$. Based on this constraint, we propose to specify the bias variance using $\sigma_b^2 = \beta(1-\rho_*)\sigma_*^2, \beta \in [0,1)$. As a result, we obtain the following expression for the variance of the weights:

$$\sigma_w^2 = \frac{2}{1+\alpha^2}\frac{1}{N}\Big(1 - \beta(1-\rho^*) - \mu_w^2 f_c\big(\mathrm{Corr}[s_1^-, s_2^-]\big)\Big).$$

Combining this result with equations (33) and (40) under similar conditions as in section B.3, we obtain the following initialisation parameters:

$$\mu_w = \pm\sqrt{\rho_*\, f_\mathrm{c}(\rho_*)^{-1}} \qquad \sigma_w^2 = \frac{2}{1+\alpha^2}\frac{1}{N}(1-\rho_*)(1-\beta) \tag{45}$$

$$\mu_b = -N\mu_w(1-\alpha)\sqrt{\frac{\sigma_*^2}{2\pi}} \qquad \sigma_b^2 = \beta(1-\rho_*)\sigma_*^2. \tag{46}$$

Note that this introduces an additional hyper-parameter, $\beta \in [0,1)$.

To get an idea of the effects of setting $\beta > 0$, we ran an additional experiment with $\beta = \frac{1}{2}$. By introducing variation in the initial bias parameters, the pre-activations at initialisation can become negative as well. As a result, we suspect that the distribution of the pre-activations can become more Gaussian-like by increasing β. Figure 6 provides a comparison of the propagation with $\beta = 0$ and $\beta = \frac{1}{2}$, showing a subtle, yet visible, effect. Figure 7 shows learning curves for $\beta \in \{0, \frac{1}{2}\}$. Here, we observe that the random biases enables training of ICNNs in the 7-layer network on MNIST. This indicates that initialising the bias parameters with Gaussian random samples can further improve the model performance.

C.4 Ablation Studies

To better understand the effects of the various hyper-parameters of our proposed method, we ran a set of ablation experiments. One of the most important factors of our proposed method is the initialisation of the bias parameters. This raises the question whether it might suffice to initialise the bias parameters (cf. eq. 8) and ignore the weights entirely. Figure 8 compares networks trained with and without our full initialisation to those where only the bias parameters are initialised according to eq. (8). The weight parameters for the latter networks are initialised using strategies for regular networks. Our experiments show that only centring the pre-activations does not suffice to make these networks learn better. The networks converge faster, but they get stuck on the same plateau that ICNNs that do not use our principled initialisation get stuck on. For shallow networks, the bias shift even seems to prevent ICNNs from getting past this plateau, leading to inferior learning (first row of Figure 8).

The choice for $\rho_* = \frac{1}{2}$ in section 3 is rather arbitrary. The main motivation for this choice is that it leads to a closed-form expression for the ReLU kernel (see section A.3). However, other choices for ρ_* are still possible. Since the fixed point for ρ_* is not stable (see section B.4), the correlation still tends to drift towards one. As a result, choosing a lower value for ρ_* might allow to train even deeper networks. We verified this experimentally and the learning curves for these experiments can be found in Figure 9.

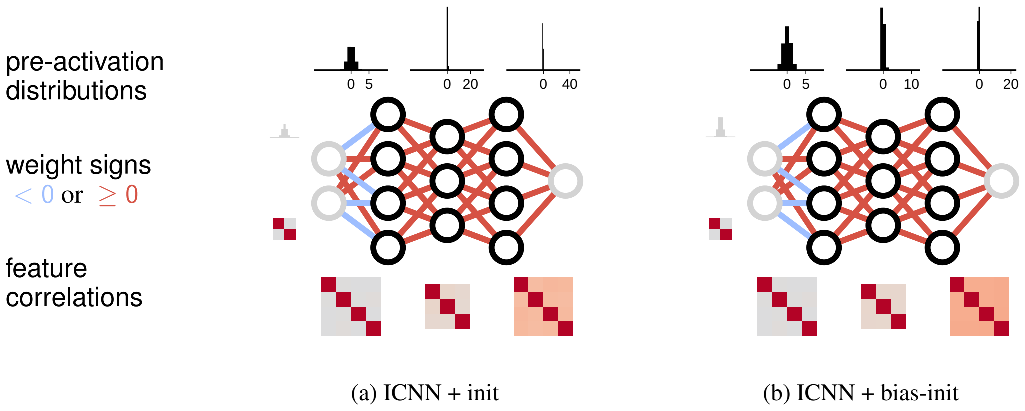


Figure 6: Illustration of the effects due to good signal propagation in hidden layers. Blue and red connections depict negative and positive weights, respectively. The distributions of pre-activations in each layer are shown on the top. On the bottom, the feature correlation matrices in each layer are displayed. The input distribution is depicted by the small elements on the left.

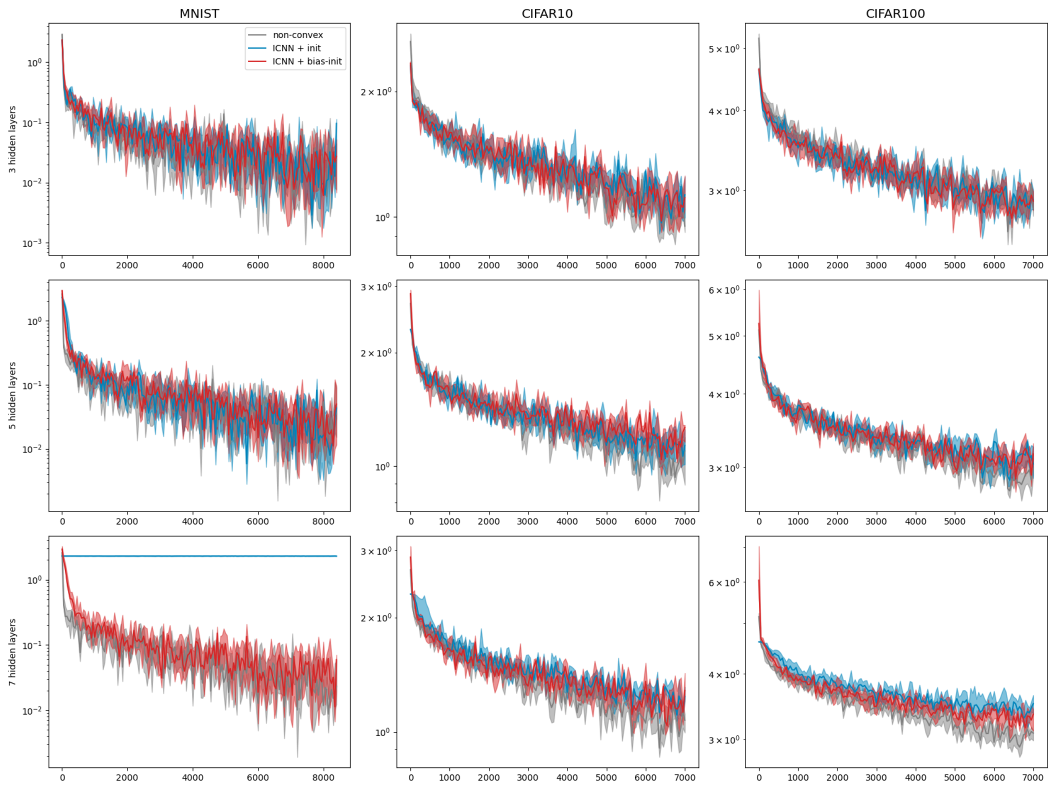


Figure 7: Training loss curves of ICNN variants with the same architecture with three, five and seven hidden layers on the MNIST, CIFAR10 and CIFAR100 datasets. "ICNN + init": our principled initialisation for ICNNs with $\sigma_b^2 = \beta = 0$. "ICNN + bias-init": our principled initialisation for ICNNs with $\sigma_b^2 = \beta = \frac{1}{2}$. "non-convex": a traditional non-convex network. The median performance over ten runs is displayed, shaded regions represent the inter-quartile range.

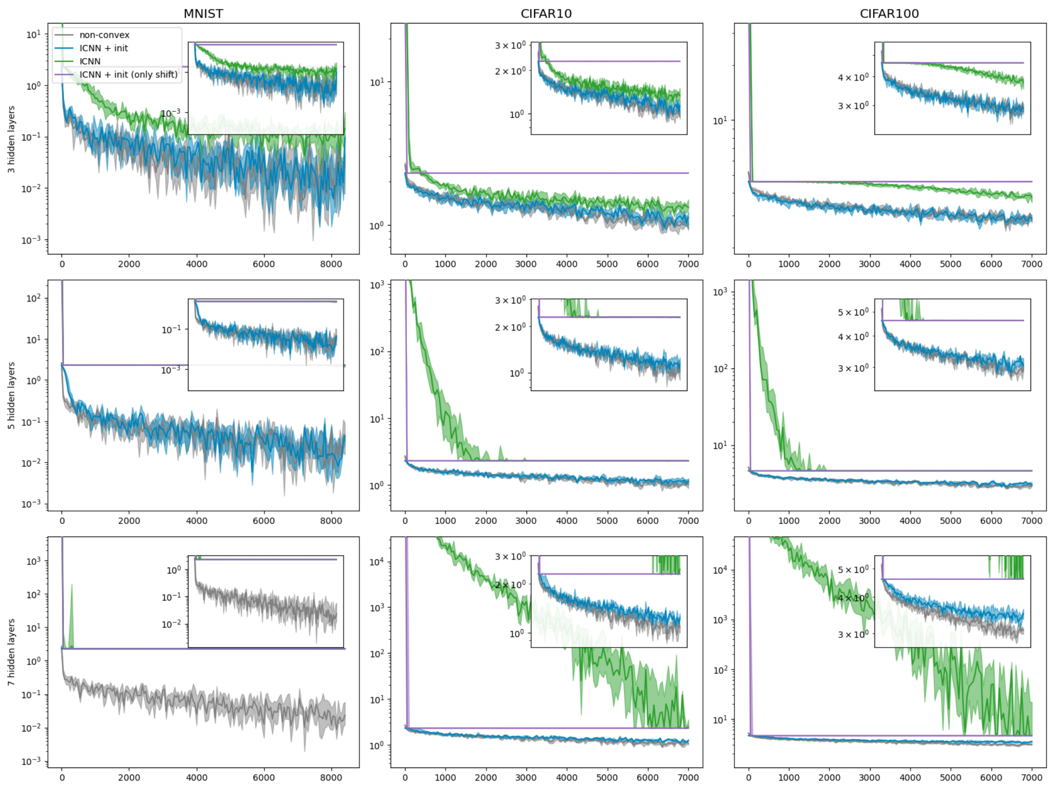


Figure 8: Training loss curves of ICNN variants with the same architecture with three, five and seven hidden layers on the MNIST, CIFAR10 and CIFAR100 datasets. “ICNN + init”: our principled initialisation for ICNNs w/o skip-connections. “ICNN” is an Input-Convex Neural Network with He initialisation. “ICNN + init (only shift)”: our principled initialisation for ICNNs for bias parameters only. “non-convex”: a traditional non-convex network. The median performance over ten runs is displayed, shaded regions represent the inter-quartile range. Note that learning curves for “ICNN” and “ICNN + init (only shift)” overlap on MNIST.

C.5 Different ICNN Implementations

The original ICNN uses a projection method after every update to keep the weights positive (Amos et al., 2017). An alternative method to keep weights positive is to re-parameterise the weights using a function with positive co-domain (cf. Nesterov et al., 2022). If weights are re-parameterised using the exponential function, we do not need to draw weights from a log-normal distribution. I.e. such that $\boldsymbol{W} = \exp(\tilde{\boldsymbol{W}})$, where $\exp(\cdot)$ is applied element-wise. Instead, the initial weights can be directly sampled from a Gaussian distribution. Figure 10 shows the result for the initialisation experiments from Figure 2 using the re-parameterisation instead of the projection method.

Figure 10 shows that our initialisation makes it possible to train ICNNs using the exponential re-parameterisation. This re-parameterisation greatly affects the learning dynamics of the network, however. Therefore, we find that this particular implementation of ICNNs generally does not perform quite as well as the original implementation. Figure 11 shows the learning curves for re-parameterised ICNNs using the hyper-parameters from table 4. The search space (in Table 2) is the same as for Figure 3 in the main paper.

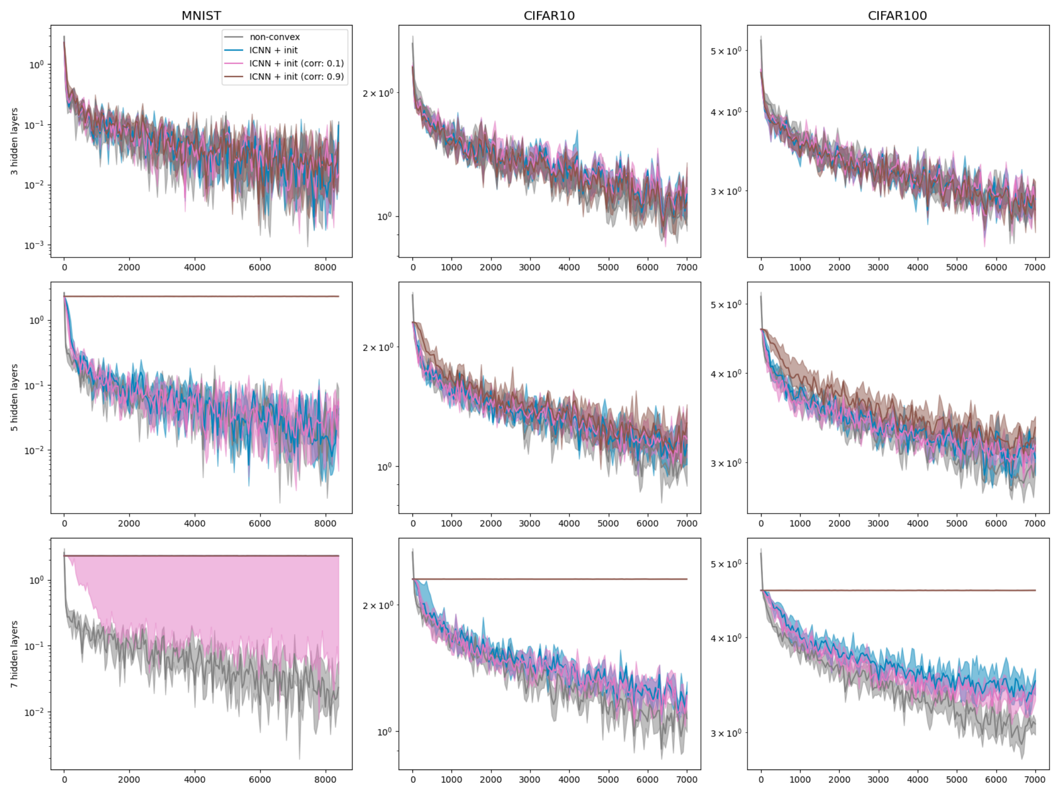


Figure 9: Training loss curves of ICNN variants with the same architecture with three, five and seven hidden layers on the MNIST, CIFAR10 and CIFAR100 datasets. "ICNN + init": our principled initialisation for ICNNs with $\rho_* = \frac{1}{2}$. "ICNN + init (corr: r)": our principled initialisation for ICNNs with $\rho_* = r$. "non-convex": a traditional non-convex network. The median performance over ten runs is displayed, shaded regions represent the inter-quartile range.

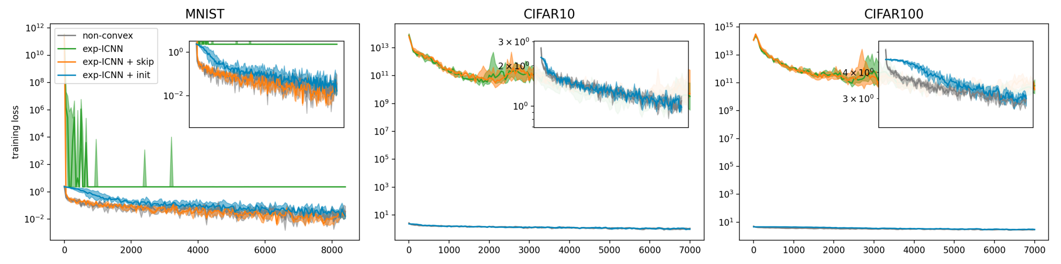
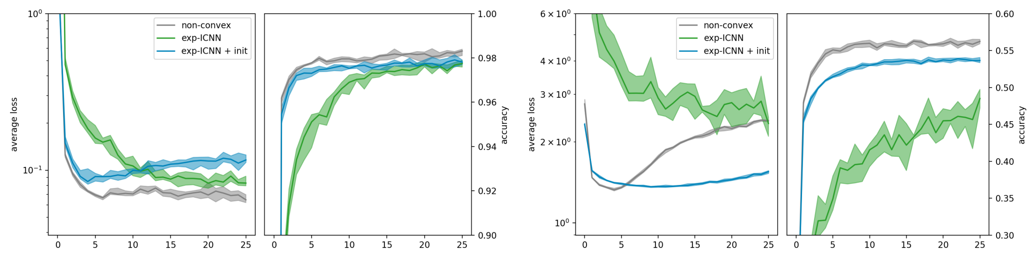


Figure 10: Training loss curves of ICNN variants using an exponential reparameterisation of weights. The reference exp-ICNN has no skip-connections and is initialised with the default He-initialisation. The exp-ICNN + skip model additionally includes skip-connections, but no principled initialisation. Our principled initialisation for exp-ICNNs without skip-connections is denoted by ICNN + init. The results for a non-convex network are included for reference. Each curve displays the median performance over ten runs. Shaded regions represent the region between the quartiles and dashed lines represent min- and maxima over the ten runs. The inset figures provide a view of the loss curves zoomed in on the reference non-convex network.



(a) MNIST (b) CIFAR10

Figure 11: Test set metrics of compared methods on the (a) MNIST and (b) CIFAR10 datasets. Each curve displays the median performance over ten runs. Shaded regions represent the interquartile range over the ten runs.

Table 4: Optimal hyper-parameters for each configuration in Figure 11. The epoch column reports the epoch in which the validation accuracy was highest.

		architecture	η	L_2	pre-processing	epoch	accuracy
MNIST	non-convex	(784, 1 000, 1 000, 10)	1e-4	0.01	ZCA	25	98.62%
	exp-ICNN	(784, 1 000, 10)	1e-4	0.01	none	25	98.02%
	exp-ICNN + init	(784, 1 000, 1 000, 10)	1e-2	0.00	ZCA	25	98.18%
CIFAR	non-convex	(3072, 10 000, 1 000, 10)	1e-4	0.01	ZCA	18	55.92%
	exp-ICNN	(3072, 1 000, 10)	1e-4	0.01	none	22	48.38%
	exp-ICNN + init	(3072, 10 000, 10)	1e-3	0.00	PCA	24	55.10%