Periodic Extrapolative Generalisation in Neural Networks

Peter Belcak
ETH Zürich
Zürich, Switzerland
belcak@ethz.ch

Roger Wattenhofer
ETH Zürich
Zürich, Switzerland
wattenhofer@ethz.ch

***Abstract*—The learning of the simplest possible computational pattern – periodicity – is an open problem in the research of strong generalisation in neural networks. We formalise the problem of extrapolative generalisation for periodic signals and systematically investigate the generalisation abilities of classical, population-based, and recently proposed periodic architectures on a set of benchmarking tasks. We find that periodic and "snake" activation functions consistently fail at periodic extrapolation, regardless of the trainability of their periodicity parameters. Further, our results show that traditional sequential models still outperform the novel architectures designed specifically for extrapolation, and that these are in turn trumped by population-based training. We make our benchmarking and evaluation toolkit, PerKit[1], available and easily accessible to facilitate future work in the area.**

***Index Terms*—neural networks, generalisation, extrapolation, periodicity**

I. Introduction

Neural networks have for long been hailed for their ability to construct representations capturing the prevailing patterns in data provided, and to meaningfully generalise to previously unseen data of similar nature. The ability of a network to perform *interpolating* generalisations is commonly associated with settling at loss surface minima that are surrounded by large areas of flatness [5]. This is also the mode of generalisation exploited by autoencoder [9], [11] and variational autoencoder [10] architectures. Generalising with neural networks by *extrapolation* is often impossible for non-linear tasks, and only sometimes possible if the appropriate non-linearities are encoded in the architecture and input representation [17], or if the task is fundamentally algorithmic and tailored to that end [1], [2].

We focus on the extrapolative generalisation and observe that the simplest possible patterns permitting extrapolation are periodic, requiring only a single-tape write-only Turing machine unconditionally outputting the pattern without halting. In the context of neural networks, we distinguish between three distinct periodicity-related learning tasks, namely the learning of a periodic signal

L1. without the need for extrapolation (e.g. for conditional generation),
L2. with the need for extrapolation but with the knowledge of the (approximate) period a priori, and
L3. with the need for extrapolation but with period being a learned parameter.

Naturally, even more complex tasks (more difficult than L3) can be considered, for example the learning to separate individual periodic components from which a class of signal is composed. These are beyond the scope of our work.

We show that models succeeding in L1 are prone to fail at L2, and that models proposed for L2 fail or struggle with L3. This leads to an establishment of an order of learning difficulty for neural networks. Given a downstream task, all models of higher levels generally succeed at lower levels of the hierarchy. We further find that the standard evaluation metrics used for regression tasks do not represent the relative successes of models learning periodic patterns well when training models for extrapolation. After classifying the common prediction faults for L2-L3 tasks, we therefore propose new metrics for this purpose.

Having noticed that the current literature lacks a unified set of criteria to evaluate the suitability of models for L1-L3 tasks, we develop a benchmarking dataset for periodic generalisation and produce a comprehensive comparative study of models proposed so far. On top of models that have been introduced specifically to tackle the problems of learning periodic functions and periodic generalisation, we also evaluate classical and evolutionary population-based training (PBT) methods on the same benchmarking dataset. These methods are consistent in their execution with their peers in literature and their evaluation results give a natural insight into what can be achieved by making repeated "informed guesses" about the periods of the signals being learned. Since PBT methods operate on a population of simple networks that are iteratively trained and adjusted for best fit with the target signal, their performance serves as an expectation baseline for future work aiming to address periodic generalisation in a more sophisticated manner.

Our contributions are therefore

- the formal specification of the problem of periodic extrapolative generalisation in neural networks and the establishment of difficulty hierarchy L1-L3 (Sect. III),
- a systematic evaluation of the behaviour of the models recently proposed for learning of periodic signals with

[1]PerKit: A toolkit for the study of periodicity in neural networks. Available at https://github.com/pbelcak/perkit.

neural networks (Sect. V),

- the establishment of an expectation baseline on the possible performance of neural models for the periodic extrapolation tasks by performing population-based training of simple neural networks (Sect. IV and Sect. V), and
- the introduction of a comprehensive, accessible benchmarking toolkit, consisting of a dataset and tasks tailored to meaningfully asses the ability of models to perform periodic extrapolative generalisations and of a unified software framework designed to accelerate further research in the field (Footnote 1).

II. Related Work

Trigonometric Activations. A feedforward neural architecture aiming to mimic the behaviour of Fourier series has first been proposed in [7], introducing the "cosine squasher" activation, very similar in shape to the now-familiar sigmoid but with domain directly adjusted to π-periodicity. A further attempt of similar nature has been made in [16], which introduced an activation featuring frequency-adjusted and frequency-offset cosines, aiming for "Fourier Neural Networks". More recently, [14] introduced an activation based on a linear combination of frequency-adjusted and frequency-offset sine and cosine. These trigonometric activations have, however, been singled out for causing problems in optimisation due to their non-monotonicity [15], [18]. These proposed approaches are included in our evaluation.

Monotonicitised Trigonometric Activations. To address what has been thought of as the main fault of the work on neural networks mimicking the behaviour of Fourier series, [19] proposed $x+\sin(x)$, $x+\cos(x)$, and $x+\sin^2(ax)$ activations and demonstrated that they possess some potential for generalisation beyond the training domain while still performing well on standard tasks defined on real-world data such as classifying the MNIST dataset. In later sections, we show that these activation functions consistently fail at periodic extrapolation, and that they offer little to no improvement over purely periodic activations.

Fourier-Based Decoder. [13] designed a VAE-based architecture giving coefficients of Fourier series as decoder output and evaluated it against both synthetic and ECG data, showing superior performance in comparison to methods common in time-series analysis. The authors, however, normalise all signals to have period 1 and do not explore extrapolative generalisation behaviour of their model.

Recurrent Architectures. Recurrent architectures are the canonical tool for time-series prediction but have been criticised in [13] for their shortcomings when the input samples are fed in irregular time intervals or contain noisy observations. We find that they are remarkably robust nevertheless.

Classical Approaches. In the context of structure discovery, several methods have been proposed for decomposing time series in an explainable fashion to arrive at a composition of patterns that permits straightforward extrapolation [3], [4]. While our work carries some resemblance to the motifs of this research, our goal is to investigate the extrapolative abilities of neural networks, rather than to propose neurosymbolic or evolutionary methods to further research in structure discovery.

III. Learning for Periodic Extrapolative Generalisation

A. Formal Specification

Let $\mathcal{D}_T \subseteq \mathbb{R}^{d_i} \times \mathbb{R}^{d_o}$ be a training domain (dataset) where d_i, d_o are the input and output dimensions, respectively. Let $\gamma : \mathbb{R}^{d_i} \to \mathbb{R}^{d_o}$ be a function such that for $\psi \in \mathbb{R}^{d_i}$

$$\gamma(\psi k + \phi) = \gamma(\psi(k+1) + \phi) \, \forall k \in \mathbb{Z}, \phi \in \mathbb{R}^{d_i}$$

and for $(x, y) \in \mathcal{D}_T$, $y = \gamma(x)$.

The problem of periodic generalisation in neural networks is the problem of designing a neural model $\mathcal{M}$ such that when trained on $\mathcal{D}_T$,

$$\mathcal{M}(x) = \gamma(x) \text{ for } x = \psi k_x + \phi_x \notin \mathcal{D}_T.$$

Let Γ be the set of all pairs (γ, ψ) satisfying the above requirement. Then the dimension of span$(\psi : (\gamma, \psi) \in \Gamma)$ is *the order of periodicity* of $\mathcal{D}_T$.

It is, however, difficult to find out whether a network outputs a particular value on infinitely many points. We therefore evaluate the ability of $\mathcal{M}$ to extrapolate periodically by testing its predictions $\mathcal{M}(x)$ against $\gamma(x)$ only for $x \in \mathcal{D}_E$, where $\mathcal{D}_E \subseteq \mathbb{R}^{d_i} \times \mathbb{R}^{d_o}$ with $\mathcal{D}_E \cap \mathcal{D}_T = \emptyset$ is the evaluation domain.

B. Difficulty of Periodicity-Learning Tasks

We illustrate the existence of the difficulty hierarchy L1-L3 of periodicity-learning tasks experimentally. For the direct GRU and snake experiments, we sample training data randomly from example periodic functions in the range $[-5\pi, 5\pi]$. In the case of the L1 task, we further add Gaussian noise randomly generated for every epoch of learning with mean 0 and variance 0.0225 to the original signal. For the L2 and L3 tasks we look at the model predictions on the wider range $[-8\pi, 8\pi]$ except for auto-regressive GRU, where we train on $[0\pi, 12\pi]$ and evaluate on $[12\pi, 23\pi]$. The results are shown in Fig. 1.

We see that the GRU recurrent network fed directly with signal time input succeeds in uncovering the original signal despite the presence of noise in training, but fails to generalise the learned signal beyond the training range even when trained on regular inputs without the presence of noise. We also observe that auto-regressive GRU recurrent networks fare fairly well on the L3 tasks but suffer from loss of information the further away from the training data they extrapolate. It was presented in [19] that a snake-activated feedforward neural network can learn to fit a periodic signal if the period is known beforehands. The bottom row of Fig. 1 shows that such networks, however, fail to learn periodic signals when the frequency parameter of the snake activation is made trainable, something we elaborate on in Sect. III-D. In our experience, models that succeed at L3 tasks do well on L2 tasks, and those which succeed at L2 tasks mostly do not struggle with L1 tasks either if the period is known. This illustration is further supported by our results in Sect. V.

C. Recurrent Extrapolation

It was previously demonstrated that the extrapolation behaviour of feedforward neural networks with ReLU and $\tanh$ activation functions is dictated by the analytical form of the activation function (ReLU diverges to $\pm\infty$, $\tanh$ tends towards a constant value), and that this result also holds for sigmoidal networks and the corresponding common variants [19]. This is despite the fact that feedforward neural networks regularly show excellent performance in approximating sampled functions on training intervals, even in the presence of balanced noise. We extended on this by looking at the extrapolative properties of recurrent architectures and noted that while recurrent networks in auto-regressive predictive setup succeed in extrapolating reasonably well beyond the training range, they do not do so if the immediate signal past does not serve as a reliable clue to the future. The results of an experiment showcasing the extrapolative behaviour of RNNs in auto-regressive configuration are depicted in Fig. 2

D. Snake Extrapolation

To address the shortcomings of standard activation functions in extrapolation, [19] proposed the family of "snake" activation functions. If the frequency a of the periodic signal to be approximated is known, a two-layer feedforward network with the corresponding snake activation $x + sin^2(ax)$ has been demonstrated to be able to approximately learn the amplitude of the signal. Authors further proposed to make the parameter a of the activation a learnable parameter and appealed to Fourier convergence for justification of general learnability properties. Our experiments following the setup and training from [19] show that regardless of the trainability of the frequency parameter, a snake-activated feedforward neural network often resorts to maximising the frequency and ignores the loss minima corresponding to the Fourier coefficients (Fig. 3). Further, we observed (Fig. 4) that when the frequency parameter is trainable, such feedforward networks even fail to learn the activation function itself.

E. Common Modes of Signal Degradation in Periodic Extrapolation

Three types information can be lost or slowly degrading with increasing distance from the training domain, namely the values of frequency and amplitude parameters, and the shape of the underlying periodic function (Fig. 5). In Fig. 2 we have observed on recurrent networks that these three modes of information loss do not necessarily have to happen at same the pace.

If the data in the evaluation domain $\mathcal{D}_E$ consists of only sampled points, it might be difficult or impossible to judge the quality of predictions when the signal shifts in "time" (the argument of γ), even if only slightly.

In our experimentation, we recognised three modes of model prediction degradation related to the periodicity parameters of the ground-truth γ:

- **Periodic Shift (SH).** The model clips away or fills in parts of the learned or previously predicted signal periodically in discrete increments, thus making the predictions for individual periods increasingly more shifted with every iteration.
- **Periodic Speedup (SP).** The model believes that the frequency of γ is a constant $f' \neq f$ and the predictions are continuously shifted due to the frequency mismatch.
- **Periodic Acceleration (AC).** The model relies on its previous predictions for the periodicity information and

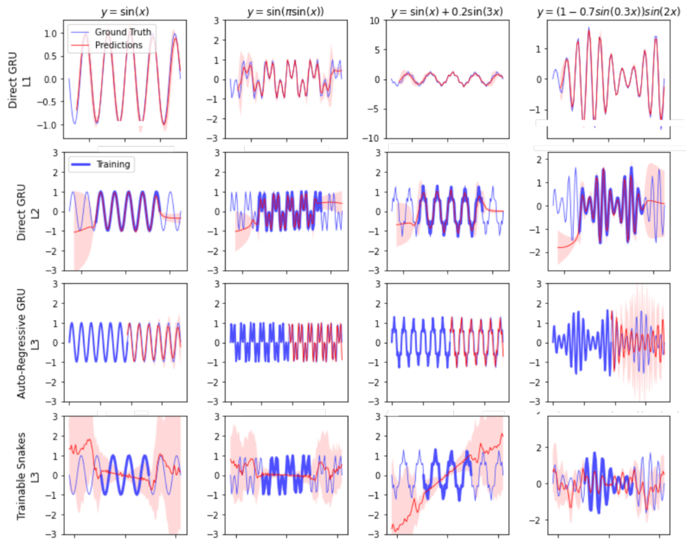

Fig. 1. An experimental analysis of how a selection of models learns and extrapolates various basic function types. The red curve represents the median model prediction and the shaded regions show the 90% credibility interval from 30 runs. GRU networks used 120 units aligned in a single layer, and the snake network was a two-layer feedforward network with 128 snake neurons in the hidden layer.

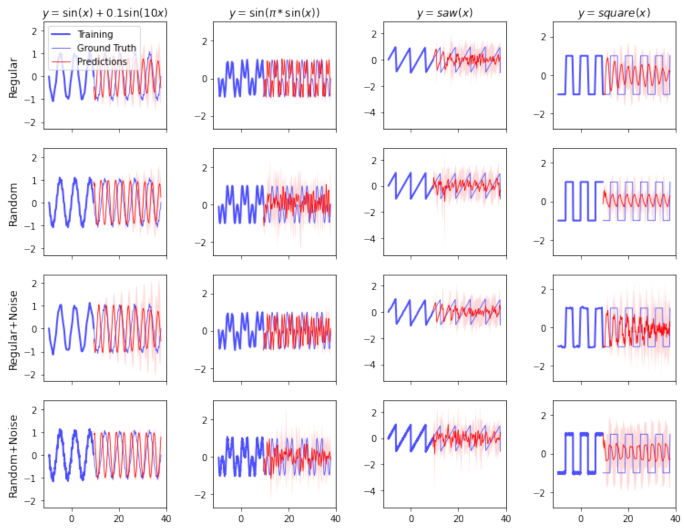

Fig. 2. The results of a systematic experiment on the ability of auto-regressive RNNs to generalise by extrapolation. Regular sampling means sampling training points at regular intervals, random sampling entails choosing a point at random from equally-spaced bins.

increases or decreases its belief f' about the ground-truth frequency f periodically in discrete increments.

We perform systematic evaluation with these defects in mind in Sect. V.

IV. Population-Based Training for Periodic Generalisation

To give a reasonable expectation on the possible performance of machine-learning models on the PerKit's benchmark datasets we also evaluate the extrapolation performance of populations of simple models trained with Bayesian and genetic approaches. In contrast to the recurrent networks and feedforward networks with periodic activations of Sect. III, populations of simple models benefit from being able to make multiple simultaneous guesses about the period of the signal being learned. Further, once a "good guess" has been identified, they can start to exploit the guess to train ever better-performing models – a behaviour hard to induce in traditional neural network training.

We consider three population-based models, Bayes, n-Fittest, and Pareto, for learning periodic signals with the objective to extrapolate. These are all parametrised the same and train populations of networks of the same architecture.

Bayes performs Bayesian optimisation [6], aiming to minimise individual final training losses. It does so by tuning its period guess by guided sampling in each iteration of the optimisation process. n-Fittest and Pareto follow a common skeleton evolutionary algorithm with parametric neighbourhood crossovers, differing only in the method by which they choose the subset of the population that is to reproduce. They too aim to minimise individual final training losses, but they do so by choosing the best-performing models to reproduce for each generation. For simplicity and consistency, we will refer to sampling in Bayes as reproduction and to iterations of Bayes as generations.

The inputs to each of the above models are the assumed signal master period range $[r_a, r_b]$, root (starting) population n_r, and minimum number of individuals to reproduce each generation $n_g \leq n_r$. Further, n-Fittest and Pareto also need a minimum number of descendants of different roots that the algorithm is to *enforce to be present* among the population that is to reproduce – $n_e \leq n_r$. The algorithms run until the maximum number of generations has been reached or until a fitness threshold has been crossed. Their output is a population $\mathcal{P}$, whose fittest individuals are those who are likely to achieve the best performance in terms of their ability to extrapolate pure periodic signals or periodic signals with trends.

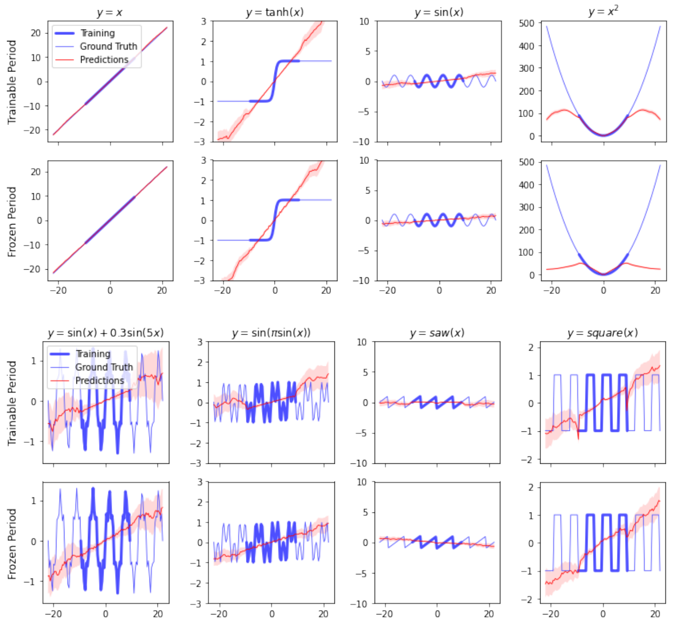

Fig. 3. A depiction of the tendency of snake neural networks to increase frequency and treat the snake activation as almost-linear component to interpolate the training data. *Top.* The results of 40 training instances of snake-powered two-layer feedforward neural networks on general signals. The hidden layer is 128 neurons wide and its initial frequencies are drawn according to a continuous uniform distribution on the closed interval $[1, 6]$. The shaded region represents the 90% confidence interval, and the median line is bold red. *Bottom.* As for the above but on periodic signals.

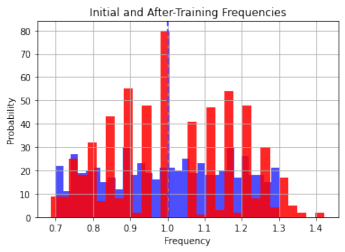

Fig. 4. The results of an experiment using a single-snake-neuron network to learn an underlying snake signal with parameter $a = 1$. The parameter a of the network attempting to learn the signal is trainable and initialised according to a uniform distribution on $[0.7, 1.3]$. The blue and red histograms chart the initial (pre-training) and post-training values of parameter a in the network. We observe that the trained parameters settle consistently around nearby local minima and only a fractional minority finds the true frequency of 1.

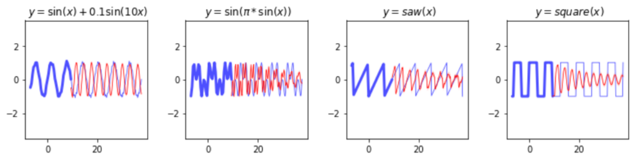

Fig. 5. An illustration of information loss by recurrent neural networks in extrapolative generalisation. The left-most sinusoidal signal is shifted and its frequency is higher than that of the ground truth (periodic speedup). Other three signals suffer from slow dissolution of shape and amplitude information. In the right-most plot, the recurrent network failed to learn the shape of the signal.

A. Individuals

The individuals of the population, or population *units*, are neural networks consisting of *trend*, *periodicity*, and *composer* sub-units (Fig. 6). In our experiments, we chose the trend sub-unit to be a linear feedforward network, periodicity sub-unit to be a feedforward network with ReLU-activated hidden layers and a linear output layer, and the composer to be a simple single-layer linear network. We have also successfully experimented with a polynomial neuron (such as the one seen in GMDH networks) as the trend unit and aim to report on our findings in our future work. Every individual a has an associated genetic parameter p_a representing its period estimate. The input to population units is the time-component x of a potentially periodic signal. x is wired directly to the trend sub-unit, $x \mod p_a$ (modulo taken in direction from $-\infty$ to ∞) is fed into the periodicity sub-unit, and the outputs of the periodicity and trend sub-units are then forwarded to the composer, which yields the signal estimate y_a of a. This configuration assumes that a new period of the target signal begins at the origin (0).

The initial population consists of n_r root individuals, with parameters $p_\bullet$ spaced evenly on $[r_a, r_b]$ including at the endpoints of the interval. In our experiments, the weights of roots' sub-units are initialised with the Glorot uniform distribution [8]. When individuals a_1, a_2 are chosen to reproduce, we designate the root ancestor of the fitter of the two roots the root ancestor of their offspring. The root ancestor of each root is the root itself.

We number generations as $g = 1, 2, \ldots$ (the zeroth generation is the root population). For each generation, we first train previously untrained units on 80% of the available training data with mean squared error loss, and use the remaining data for validation. We terminate training early if the validation loss stops improving for a number of epochs.

B. *Evolution of* BAYES

We simply follow the iterative procedure for Bayesian optimisation [6].

C. *Evolution of* PARETO, N-FITTEST

While it could be argued that the fitness should be assessed on the basis of an evaluation sample that is taken from outside the training range, it is precisely the point that our models learn to extrapolate periodically without any information about the signal besides what is available in training. For each individual a we keep the end validation loss ℓ_a (the "unfitness" of a).

Once the training phase has been completed, we proceed with selecting the set of "best" candidates for reproduction $\mathcal{B}$. For the N-FITTEST algorithm, we use $-\ell_a$ as the measure of fitness and take the n_g individuals with least $\ell_\bullet$ to be $\mathcal{B}$, choosing the younger individual in case of a tie. For the PARETO algorithm, we fit a Pareto distribution across the range of losses in $\mathcal{P}$ and calculate Pareto fitness scores $s_\bullet$ according to the formula

$$\mu_a = \frac{\ell_a - \min_{z \in \mathcal{P}} \ell_z}{\max_{z \in \mathcal{P}} \ell_z - \min_{z \in \mathcal{P}} \ell_z} \quad s_a \quad = \frac{S\sqrt{g}}{(1+\mu_a)^{1+S\sqrt{g}}},$$

where S is a score-scaling hyperparameter of the model controlling the exploration-exploitation balance and $\mu_\bullet$ is the validation loss normalised to the range seen in $\mathcal{P}$. n_g candidates for reproduction are then chosen to form $\mathcal{B}$ according to a multinoulli distribution where the probability of individual a being selected is proportional to s_a.

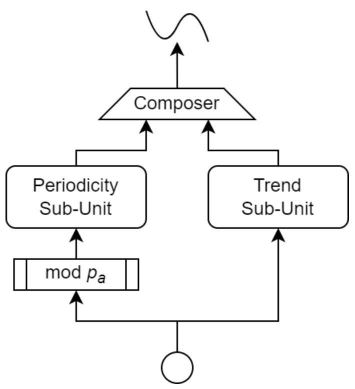

Fig. 6. The architecture of a single population unit a. p_a denotes the genetic parameter.

We then count the number of distinct root ancestors d_r among candidates in $\mathcal{B}$ and if it is less than n_e, we keep adding least unfit individuals not yet in $\mathcal{B}$ with root ancestors different from all the previous until $d_r = n_e$.

Finally, we perform the crossovers. For every $b_2 \in \mathcal{B}$, we choose the other parents b_1, b_3 such that

$$b_1 = \underset{z \text{ s.t. } p_z < p_{b_2}}{\operatorname{argmax}} \; p_z \quad \text{and} \quad b_3 = \underset{z \text{ s.t. } p_z > p_{b_2}}{\operatorname{argmin}} \; p_z.$$

We place parameter of the offspring individuals b_1, b_2 proportionally to the parents' fitness scores. Let b_i, b_j be parents such that $p_{b_i} < p_{b_j}$. We first calculate the pair-relative fitness scores,

$$\sigma_{b_i}^{\text{N-FITTEST}} = \frac{\ell_{b_i}}{\ell_{b_i} + \ell_{b_j}} \qquad \sigma_{b_j}^{\text{N-FITTEST}} = 1 - \sigma_{b_i}^{\text{N-FITTEST}}$$
$$\sigma_{b_i}^{\text{PARETO}} = \frac{s_{b_j}}{s_{b_i} + s_{b_j}} \qquad \sigma_{b_j}^{\text{PARETO}} = 1 - \sigma_{b_i}^{\text{PARETO}},$$

(note the contrasting roles of b_i and b_j per case), and then compute the new parameter,

$$p_c = p_{b_i} + \sigma_{b_i}^{\bullet}(p_{b_j} - p_{b_i})$$

We perform this procedure for the pairs b_1, b_2 and b_2, b_3 to get children parameters p_{c_1}, p_{c_2} respectively.

Instead of initialising units for children c_i afresh, we clone the trained state of b_2 (central parent) into c_i and begin the training from that state. Experimenting, we learned that while this approach does not lead to consistent improvements in terms of accuracy of our models' predictions when compared to starting from default weight initialisation, it significantly accelerates the training.

The offspring of $\mathcal{B}$ are then added to $\mathcal{P}$ and if the prespecified termination criteria such as accuracy or maximum number of generations are not met, the algorithms proceed with the next generation.

We illustrate the outputs of N-FITTEST in Fig. 7.

V. EVALUATION

A. *Method*

To systematically evaluate the ability of a neural model to generalise extrapolatively, we design a straightforward but

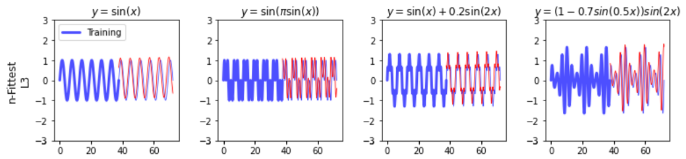

Fig. 7. An example of the training and predictions of the N-FITTEST model with configuration as in Fig. 1. $n_r = 8, n_g = 7, n_e = 3, r_a = 3\pi, r_b = 5\pi$.

rich collection of classes of continuous periodic signals with hierarchically increasing complexity.

We begin with generation of periodic forms. Recursively, a periodic form is either an elementary form (see Table I) or a sum, product, or chained application of a periodic form and an elementary form. More formally,

$$\begin{aligned} \text{FORM}_0 &:= \text{FORM}_E \\ \text{FORM}_i^{+\times\circ} &:= \{f + g, fg, f \circ g \\ &\quad : f \in \text{FORM}_E, g \in \text{FORM}_{i-1}\} \end{aligned}$$

where $i \geq 1$, FORM_E is the set of elementary forms, FORM_i^- the set of forms of order $i \geq 0$ arrived at by combining lower-order forms by operations in superscript.

On top of learning to generalise in periodicity, some models are also able to capture a general trend offsetting an otherwise periodic signal. To represent that in our signal collection, we consider polynomial and exponential trends. Denoting the trend forms (Table I) by FORM_T, a general form is a sum of a periodic form and a trend form.

B. *Experiments*

We conduct three experiments, assessing the ability of models to extrapolate (L2-L3): periodic signals, periodic signals with noisy training, and signals generated by offsetting a periodic base with a linear trend.

For the noiseless and noisy periodic signals we use forms in $\text{FORM}_2^{+\times}$, for signals with trend we use forms $\{f + g : f \in \text{FORM}_2^{+\times}, g \in \text{FORM}_T\}$. For each form we generate n_v variants, drawing every coefficient including frequencies uniformly at random from a fixed range. Then, we choose a master period τ from $[0.5, 1.0]$ uniformly at random and normalise individual frequencies so that the variant is a periodic signal with period τ. The training domain is $\mathcal{D}_T = (-n_T\tau, n_T\tau)$ and we evaluate on $\mathcal{D}_E = (-n_E\tau, n_E\tau) \backslash \mathcal{D}_T$, where n_T, n_E represent the numbers of periods to train/evaluate on. The signal is normalised by a constant so that all values on $\mathcal{D}_E$ lie between -1 and 1 inclusive. In the noisy scenario, we further add Gaussian noise with mean 0 and variance σ^2. The data is then sampled uniformly at random and used for training. We repeat the training and evaluation independently n_r times for each variant.

The literature on trigonometric activation functions [15], [18] cites problems with their optimisation due to the existence of a series of local minima. In an attempt to counter this, we ran all of the above models on the whole dataset 6 times – once for each optimiser among SGD, RMSprop, Adam, AdaMax,

TABLE I. An overview of forms used to evaluate extrapolative generalisation abilities. *square*, *saw*, and *poly* denote the unit square wave with 50% duty cycle, unit positive sawtooth wave, and symmetric bilateral polynomial wave of order n, respectively, each with period 1.

Name	Type	Analytic Form
square wave	elementary	$\text{square}(\frac{x}{T} + \phi)$
sawtooth wave	elementary	$\text{saw}(\frac{x}{T} + \phi)$
sinusoid	elementary	$\sin(2\pi\frac{x}{T} + 2\phi\pi)$
tangent	elementary	$\tan(\pi\frac{x}{T} + \phi\pi)$
polynomial wave of order n	elementary	$\text{poly}(\pi\frac{x}{T} + \phi\pi, n)$
polynomial trend of order n	trend	$c_0 + c_1x + \ldots + c_nx^n$
exponential trend	trend	$c_0e^{c_1x}$

AdaDelta, and Nadam. We compared the aggregate results for the whole benchmarking dataset and observed no significant differences between the results of different optimisers. Our final results, reported below, are thus computed on the collation of runs, disregarding the optimiser used.

We chose a uniform width of 64 hidden units for all feedforward networks, including those used by N-FITTEST and PARETO. In the case of snake activation functions and by the analysis of [19] this would correspond to a target of 32 (not necessarily consecutive) harmonics in the approximating Fourier series, while for $\sin + \cos$ [14] this corresponds to 64 harmonics. Individual experiments showed that adding more hidden neurons did not improve the extrapolative performance of feedforward networks. We also set the number of recurrent units to 64 for LSTM, GRU, and simple RNNs, but note that recurrent units are more complex in their structure than individual feedforward neurons. For the population units of neuroevolutionary models, we use a two-layer periodic unit 60 neurons wide with ReLU activation on the hidden layer, and a trend unit consisting of a single linear neuron.

C. *Metrics*

As was shown in Sect. III, neural models might exhibit a wide range of behaviours in periodic extrapolation scenarios, and straightforward per-point losses such as mean squared error (MSE) or mean absolute error (MAE) may not be representative of the models' ability to learn information pertaining to signal periodicity and to extrapolate beyond the training range.

We therefore use a range of metrics to capture and report on prevalence of phenomena seen in Sect. III such as periodic signal shift, signal frequency shift (signal "speed-up"), and signal frequency acceleration. Each of the evaluation metrics is based on a point-based metric m and is arrived at by minimising m across a parameter trying to correct the predictions. Let $x, y(x), y'(x)$ be the position, true, and predicted signal values respectively, e_T the nearest endpoint of the training domain, and τ a period of the signal. If the signal is offset by a trend or otherwise modified, τ is a period of the periodic component. We then define

$$\begin{aligned}\mathrm{SH}m &= \min_{w\in(-\epsilon,+\epsilon)} m\left(x, y(x), y'\left(x + w\left\lfloor\frac{x-e_T}{\tau}\right\rfloor\right)\right)\\ \mathrm{SP}m &= \min_{w\in(-\epsilon,+\epsilon)} m\left(x, y(x), y'\left(x(1+w)\right)\right)\\ \mathrm{AC}m &= \min_{w\in(-\epsilon,+\epsilon)} m\left(x, y(x), y'\left(x\left(1 + w\left\lfloor\frac{x-e_T}{\tau}\right)\right\rfloor\right)\right)\end{aligned}$$

The choice of the metrics specific to degradation phenomena can be justified as assuming that the corresponding prediction degradation phenomenon is present and finding the minimum value of point-based metric m for a parameter characterising the decay in predictive ability within some permissible range.

Further, in our experiments we observed that the tails of the extrapolative predictions (i.e. the segments of the signal farthest from the training domain) often disproportionally affected the summary metrics, and in response we chose to weigh the point contributions to the original metrics decreasingly with the increasing distance from the training domain. Let m be a point-based metric. The the distance-adjusted m is

$$\text{DA-}m(x, y, y') = m(y, y')\left(\frac{x-e_T}{d_T}\right)^{\alpha}$$

where d_T is the diameter of the training domain and $\alpha \geq 1$ is the weighing decay parameter. The comparison of the values for a per-point metric m and the distance-adjusted m allows us to quickly assess the extent to which the predictions of the model deteriorate with increasing distance from the training domain.

D. Results

The results of our experiments are shown in Tables II to IV. We performed the experiments with $n_v = 10, n_r = 5, n_T = 5, n_E = 10, \sigma^2 = 0, \alpha = 1$, sampling rate 100 per period, and recurrent window of length 7. PBT methods were run for exactly 10 generations with $n_r = 8, n_g = 7, n_e = 3, r_a = 0.5, r_b = 1.0$. Each experimental instance was repeated 6 times, once for each optimiser considered. For metric evaluation, $\epsilon = 0.05$, and 21 samples were taken over $[-\epsilon, \epsilon]$. **Emph.** and emph. denote the best performance per metric and model, respectively.

In the scenario with noiseless periodic signals (Table II), we observe that the snake activations outperform $x + \sin$ and $x + \cos$, and that the snake feedforward neural network with frozen frequency parameters further outperforms snake networks with trainable frequencies. All of the recurrent networks outperform the feedforward networks, and our genetic models N-FITTEST and PARETO further improve on the best-performing recurrent networks by 32-35% in MSE and 45-46% in SHDA-MSE.

There was a noticeable drop in performance of feedforward networks when noisy signals were considered (Table III). Both recurrent and PTB models remained largely unaffected.

On trend-offset periodic signals (Table IV), we observe that $x + \sin$ and $x + \cos$ show performance comparable to snake activations and that GRU networks outperform simple recurrent networks and LSTM architectures. It is surprising that GRU networks perform very well on the trend-offset data, tentatively suggesting that the gated recurrent unit possesses some ability to preserve shape while recognising the presence of a linear trend. The genetic models N-FITTEST and PARETO further improve on the best-performing recurrent networks by 77-79% in MSE and 85% in SHDA-MSE and SPDA-MSE.

Comparing MSE with DA-MSE allows us to judge whether there are large prediction deviations from the target signal at the points far from the training range. We observed that the

TABLE II. Results of the experiments on noiseless periodic signals.

	MSE	DA-	SHDA-	SPDA-	ACDA-
sin	0.209	0.149	0.148	0.147	0.147
cos [16]	0.245	0.174	0.172	0.171	0.171
sin + cos [14]	0.319	0.220	0.218	0.217	0.217
x + sin [19]	5.095	3.624	3.620	0.317	3.621
x + cos [19]	4.662	3.289	3.283	3.281	3.280
snake [19]	0.375	0.264	0.262	0.261	0.261
t-snake [19]	0.391	0.275	0.273	0.272	0.272
SRNN	0.081	0.811	0.074	0.070	0.070
GRU	0.071	0.072	0.065	0.061	0.062
LSTM	0.076	0.076	0.069	0.063	0.065
BAYES	0.050	0.038	0.037	0.037	0.049
N-FITTEST	0.048	0.037	0.036	0.036	**0.048**
PARETO	**0.046**	**0.036**	**0.035**	**0.035**	0.049

TABLE III. Results of the experiments on noisy periodic signals with $\sigma^2 = 0.15$.

	MSE	DA-	SHDA-	SPDA-	ACDA-
sin	0.338	0.232	0.231	0.228	0.274
cos [16]	0.373	0.274	0.273	0.269	0.322
sin + cos [14]	0.358	0.250	0.248	0.248	0.247
x + sin [19]	0.399	0.279	0.277	0.276	0.276
x + cos [19]	0.257	0.181	0.180	0.179	0.179
snake [19]	0.391	0.274	0.272	0.271	0.271
t-snake [19]	0.435	0.305	0.303	0.302	0.302
SRNN	0.075	0.075	0.069	0.065	0.065
GRU	0.072	0.072	0.065	0.059	0.061
LSTM	0.073	0.073	0.066	0.061	0.063
BAYES	0.051	0.039	0.038	0.038	0.049
N-FITTEST	**0.045**	0.035	**0.034**	**0.034**	**0.049**
PARETO	0.046	**0.035**	0.035	0.035	0.049

TABLE IV. Results of the experiment on trend-offset periodic signals without noise.

	MSE	DA-	SHDA-	SPDA-	ACDA-
sin	0.338	0.232	0.231	0.228	0.274
cos [16]	0.373	0.274	0.273	0.269	0.322
sin + cos [14]	0.520	0.371	0.370	0.367	0.431
x + sin [19]	0.486	0.336	0.334	0.331	0.432
x + cos [19]	0.342	0.237	0.236	0.235	0.346
snake [19]	0.441	0.307	0.305	0.303	0.381
t-snake [19]	0.498	0.345	0.344	0.342	0.451
SRNN	0.081	0.081	0.074	0.070	0.070
GRU	0.026	0.026	0.019	0.018	0.034
LSTM	0.366	0.366	0.355	0.351	0.378
BAYES	0.006	0.005	0.004	0.004	0.107
N-FITTEST	0.006	0.005	0.004	0.004	0.107
PARETO	**0.006**	**0.005**	**0.004**	**0.004**	**0.107**

gap (15-33%) between MSE and DA-MSE was particularly common for trigonometric and snake feedforward networks, irrespective of whether they used activations monotonicitised by linear offset. There was hardly any difference between MSE and DA-MSE for for recurrent and evolutionary models.

We see that the most common mode of extrapolative signal degradation is that of periodic speedup (cf. Sect. III-E), and that the difference is particularly noticeable (in terms of relative improvement over base metrics) in recurrent neural networks.

We attribute the generally comparable performance of N-FITTEST and PARETO to the simplicity of the search space and note that a significant difference is immediately seen if the assumed master period range $[r_a, r_b]$ is made larger and contains several multiples of the master period. We also note from the results that trigonometric activations that had been proposed for L2 tasks do not perform well on L3 tasks, consistently with our outline of the difficulty hierarchy for periodicity learning.

E. Training Resource Consumption of PBT Methods

Pupulation-based training and especially evolutionary methods are often associated with high computational cost and long program runtime.

We trained and evaluated our models on a single NVIDIA TITAN Xp GPU. For the performance comparison, we trained all our models with the Adam optimiser [12]. Thanks to the simplicity the genetic unit architecture, the full multi-generational evolutionary training and evaluation of BAYES, N-FITTEST, and PARETO took only 12-15% longer than that of snake feedforward neural networks with frozen frequencies and 40-44% shorter than snake networks with trainable frequency parameters, while outperforming both in terms of the metrics above in all scenarios.

This suggest that population-based training, albeit naive in its motivation, currently trumps the best available models tailored for periodic extrapolation in practice.

VI. CONCLUSION

We have identified periodic extrapolation as the computationally simplest mode of extrapolative generalisation. Our work systematically evaluates network architectures thought to generalise well beyond training domains and finds that traditional recurrent neural networks outperform all of the architectures proposed specifically to tackle extrapolation. Moreover, the latest invention proposed – snake activation – also consistently fails at periodic extrapolation and offers little to no improvement over purely periodic activations. We believe that these results suggest that there does not yet exist a universal architecture with plausible extrapolative properties, despite the claims of effectiveness founded on the existence of weights corresponding to the Fourier series of the target signal.

With the classical Bayesian and neuroevolutionary population-based training methods outperforming all other models while keeping their training time below that of the most recent architectures for periodic generalisation, we view their performance as natural baselines to be overcome by future work.

All our code is available as PERKIT[1] – a toolkit for the study of periodicity in neural networks. PERKIT has been designed specifically to allow making of future benchmarking extensions and additions of new models for evaluation with ease. We hope that together with our study it will help to facilitate research in extrapolative generalisation.

REFERENCES

[1] Peter Belcak, Ard Kastrati, Flavio Schenker, and Roger Wattenhofer. Fact: Learning governing abstractions behind integer sequences. *arXiv preprint arXiv:2209.09543*, 2022.

[2] Stéphane d'Ascoli, Pierre-Alexandre Kamienny, Guillaume Lample, and François Charton. Deep symbolic regression for recurrent sequences. *arXiv preprint arXiv:2201.04600*, 2022.

[3] David Duvenaud. *Automatic model construction with Gaussian processes*. PhD thesis, University of Cambridge, 2014.

[4] David Duvenaud, James Lloyd, Roger Grosse, Joshua Tenenbaum, and Ghahramani Zoubin. Structure discovery in nonparametric regression through compositional kernel search. In *International Conference on Machine Learning*, pages 1166–1174. PMLR, 2013.

[5] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. *arXiv preprint arXiv:2010.01412*, 2020.

[6] Peter I Frazier. A tutorial on bayesian optimization. *arXiv preprint arXiv:1807.02811*, 2018.

[7] A Ronald Gallant and Halbert White. There exists a neural network that does not make avoidable mistakes. In *ICNN*, pages 657–664, 1988.

[8] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings, 2010.

[9] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[10] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. 2016.

[11] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

[12] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[13] Jiyoung Lee, Wonjae Kim, Daehoon Gwak, and Edward Choi. Conditional generation of periodic signals with fourier-based decoder. *arXiv preprint arXiv:2110.12365*, 2021.

[14] Shuang Liu. Fourier neural network for machine learning. In *2013 International Conference on Machine Learning and Cybernetics*, volume 1, pages 285–290. IEEE, 2013.

[15] Giambattista Parascandolo, Heikki Huttunen, and Tuomas Virtanen. Taming the waves: sine as activation function in deep neural networks. 2016.

[16] Adrian Silvescu. Fourier neural networks. In *IJCNN'99. International Joint Conference on Neural Networks. Proceedings (Cat. No. 99CH36339)*, volume 1, pages 488–491. IEEE, 1999.

[17] Keyulu Xu, Mozhi Zhang, Jingling Li, Simon S Du, Ken-ichi Kawarabayashi, and Stefanie Jegelka. How neural networks extrapolate: From feedforward to graph neural networks. *arXiv preprint arXiv:2009.11848*, 2020.

[18] Abylay Zhumekenov, Malika Uteuliyeva, Olzhas Kabdolov, Rustem Takhanov, Zhenisbek Assylbekov, and Alejandro J Castro. Fourier neural networks: A comparative study. *arXiv preprint arXiv:1902.03011*, 2019.

[19] Liu Ziyin, Tilman Hartwig, and Masahito Ueda. Neural networks fail to learn periodic functions and how to fix it. *Advances in Neural Information Processing Systems*, 33:1583–1594, 2020.