(Non-)Convergence Results for Predictive Coding Networks

Simon Frieder [1] **Thomas Lukasiewicz** [2 1]

Abstract

Predictive coding networks (PCNs) are (un)supervised learning models, coming from neuroscience, that approximate how the brain works. One major open problem around PCNs is their convergence behavior. In this paper, we use dynamical systems theory to formally investigate the convergence of PCNs as they are used in machine learning. Doing so, we put their theory on a firm, rigorous basis, by developing a precise mathematical framework for PCN and show that for sufficiently small weights and initializations, PCNs converge for any input. Thereby, we provide the theoretical assurance that previous implementations, whose convergence was assessed solely by numerical experiments, can indeed capture the correct behavior of PCNs. Outside of the identified regime of small weights and small initializations, we show via a counterexample that PCNs can diverge, countering common beliefs held in the community. This is achieved by identifying a Neimark-Sacker bifurcation in a PCN of small size, which gives rise to an unstable fixed point and an invariant curve around it.

1. Introduction

Predictive coding networks (PCNs) (Rao & Ballard, 1999; Friston, 2003) have been proposed as unsupervised learning models, approximating to how the brain works. Recent interest in these models has led to an increased number of publications, which can be split into two main lines of investigation: One line is concerned with neuroscience aspects and variations of PCN models, and how they can be embedded in a unified "theory of the brain", e.g., (Kadmon et al., 2020; Friston, 2018; 2010); the other line is concerned with integrating PCNs tightly into mainstream machine learning by building models that can be trained and used in practice; see, e.g., (Bartunov et al., 2018; Kriegeskorte, 2015; Hassabis et al., 2017; Song et al., 2020).

Within this latter context, supervised learning variants of PCN models have been proposed recently whose training is closely aligned with that of fully-connected feedforward networks (FCFNs), which use stochastic gradient descent (SGD) and backpropagation (Whittington & Bogacz, 2017; Bogacz, 2017; Song et al., 2020; Salvatori et al., 2021b). The PCN models outlined in these articles are our objects of study. They are, similarly to FCFNs, specified by a fully-connected architecture with weights but, crucially, their training (learning) stage *as well as* their prediction stage, uses a gradient-based iteration procedure. By prediction stage, we mean those computational steps that need to be performed to obtain an output for a network whose weights are (e.g., by training) already fixed; see Figure 4. For (trained) FCFNs, in contrast, one obtains the output after a finite number of steps.

While the PCNs outlined in the four articles referenced above are conceptually highly similar, they are not identical on a formal level, and multiple variants of PCNs are discussed in (Song et al., 2020). Appendix B contains a short comparison of the models across these articles. Because most of the differences between the models pertain to their training stage, and our results that are focused on the prediction stage the persist across all model variations. To have a fixed reference, we describe a specific PCN (Sections 2 and Appendix B, respectively) relative to which we prove our results of (non-)convergence; our description of PCNs is different from the referenced articles, as we introduce new mathematical objects and new notation for various parts of PCNs to achieve a fully rigorous setup that is also notationally easier to handle.

PCNs can be trained in a way that not only approximates backpropagation but provides an exact realization of it (Lillicrap et al., 2020; Whittington & Bogacz, 2017; Song et al., 2020). This has the advantage that certain biological principles of how the brain works are satisfied during the training of PCNs—unlike for FCFNs trained with SGD and backpropagation. For example, PCNs (even beyond the

[1]Department of Computer Science, University of Oxford, UK. [2]Institute of Logic and Computation, TU Wien, Austria. Correspondence to: Simon Frieder <simon.frieder@cs.ac.ox.uk>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022.

particular sense in which we consider them) are deemed to respect local plasticity (Lillicrap et al., 2020; Whittington & Bogacz, 2019): During training, the rule to update the weights of a node only uses, in one time step, the weights of those nodes that are adjacent to it.

This, together with the fact that their performance has been reported to match FCFNs (Song et al., 2020; Salvatori et al., 2021a), makes them interesting in their own right.

We motivate our main question regarding the (non-)convergence of PCNs by noting that the version of PCNs under consideration is derived by a forward Euler discretization procedure of a system of ordinary differential equations (Whittington & Bogacz, 2017; Bogacz, 2017). Such discretization procedures are not generally asymptotically faithful to the original system; cf. (Fiedler & Scheurle, 1996), Chapter 1: "*Over fixed finite time intervals, the analogy* [between discrete and continuous time dynamical systems] *is well understood in terms of discretization errors and sophisticated discretization schemes. Over large or infinite time intervals, this analogy is not so clear, because discretization errors tend to accumulate exponentially with time*".

One could alternatively see that convergence might be problematic by noting the connection to gradient descent, for which it should be clear that convergence cannot be assumed from the beginning, in particular, when time steps are constant: For an arbitrary, sufficiently smooth function h, discretizing the initial value problem

$$\begin{cases} \dot{x} = -\nabla h(x), \\ x(0) = \bar{x} \end{cases}$$

leads to standard gradient descent

$$x_{t+1} = x_t - \gamma \nabla h(x_t), \quad x_0 \coloneqq \bar{x}, \quad \gamma > 0,$$

which, if h represents a PCN, coincides with the definition of its computation rule (see Definition B.3).

Hence, summarizing, even if the original (continuous) system has special structure that may give the hope that convergence is possible, the long-term dynamics of its discretized version have to be analyzed on a case-by-case basis; and it will indeed turn out that convergence is not always possible (see Section 5).

In this regard, one important issue in the context of PCNs is the *fixed prediction assumption*. This expression has been coined by (Rosenbaum, 2021), though only for the training stage of PCNs, and is used implicitly in (Song et al., 2020; Whittington & Bogacz, 2017), in both stages. It means, in dynamical system terminology, that in both the prediction and the training regime, there exists a fixed point to which the iterations of the PCN converge; more precisely, it means that the fixed point is *asymptotically stable*, and all ω-limit sets are singletons (we uncover the dynamical systems nature of PCNs in Section 2 and Appendix B; see Appendix A for a short introduction to dynamical systems).

There is reason to be skeptical about the general validity of the fixed prediction assumption: PCNs are gradient systems, but there are examples of gradient systems, even with continuous time, whose orbits have a non-trivial behavior, where the ω-limit set of certain points is not a singleton, but the unit sphere S^1 (Palis & De Melo, 2012). The *potential* for failure of convergence for PCNs has been noted by (Rosenbaum, 2021), though no concrete example is given and no in-depth analysis of failure modes is carried out. We note that that article is solely concerned with the training stage and different training methodologies, whereas our convergence results pertain to the prediction stage (which is independent of *any* training methodology) as well as the training stage (for a specific example); the non-convergence results pertains only to the prediction stage.

Concluding, the main contributions of this paper are to:

- carry out the first mathematically rigorous treatment of PCNs in the context of supervised machine learning methods. In this regard, we emphasize the close relationship between PCNs and discrete dynamical systems, which has so far not been explored in the literature. We provide an explicit example of how convergence can fail (which shows that forward Euler discretization can lead to a problematic long-term behavior, even if the original system has a special structure). Such negative examples have not been reported yet for the type of PCNs that we are investigating; furthermore, empirical results have rather suggested the opposite, that convergence is mostly possible; see, e.g., (Song et al., 2020);
- develop a unified mathematical framework for prediction-stage PCNs from the viewpoint of machine learning and dynamical systems, within which the aforementioned results can be placed. New propositions (e.g., Proposition B.8) are proved along the way, and known results (e.g., Proposition C.5) are recapitulated and presented in a unified manner;
- prove general convergence theorems for the prediction that provide sufficient conditions when the fixed prediction assumption is fulfilled, i.e., when the model converges. For the training stage, whose formal setup parallels the prediction stage, we have omitted a general exposition in the appendices and directly proved a convergence theorem for the specific, running example from the main body of this paper. These results partially vindicate why previous experimental results have worked.

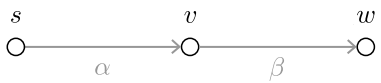

Figure 1. The simplest possible PCN that is not yet trivial. Removing the intermediate node would reduce the PCN to a linear system, when formulated as a discrete dynamical system, which would make it trivial.

Because fully formalizing PCNs leads to complex mathematical objects with difficult notation that potentially obscures the exposition, we have structured the article in such a way that all general results and definitions are relegated to the appendices. This also keeps the exposition short.

Thus, in the main body of this paper in Sections 2 to 5, a concrete example is used to illustrate the general results and proofs from the appendices in a simple and easy-to-read manner; then, a counterexample to the claim of unconditional convergence is provided (Section 5). Corresponding to Sections 2 and 3, which treat the concrete case of the prediction stage, are Appendices B and C, where the general theory in the prediction stage is presented. While the main body of this paper is self-contained, we have carefully outlined in Sections 2 and 3 how each of the concrete prediction-stage results generalize to the corresponding results from the appendices. Section 4 and 5 have no corresponding section in the appendices. The main part of the paper is written in an informal style, while the appendices are written in a definition-theorem-proof style, in order to support distinguishing different mathematical objects necessary to formalize PCNs.

2. Dynamical Systems Perspective

In this section, we illustrate via a concrete example how to interpret PCNs as dynamical systems. For the general case, see Appendix B. We focus on the prediction stage first; information about notation is given in Appendix A.

PCNs resemble FCFNs; see Figure 1. To obtain an output (on a PCN with given weights, obtained, e.g., by training), an infinite sequence of gradient-based computation steps is carried out during which the values of non-input nodes are successively computed. We consider here the simplest PCN that is not yet completely trivial, namely, one that consists of the nodes as in Figure 1.

At the end of this section, we provide an explanation how this formal model can be interpreted from a neuroscience point of view. The *architecture* of the PCN is completely determined by this diagram, but several components to complete the formalization are still missing: Let $f : \mathbb{R} \to \mathbb{R}$ be a (typically nonlinear) *activation function*, and $\alpha, \beta \in \mathbb{R}$ be fixed *weights*. Suppose that $s \in \mathbb{R}$ is some *input*, and $\hat{\eta}, \bar{\eta} \in \mathbb{R}$ are some *initializations* (which we also call *seeds*).

Adapting the general equations from (Song et al., 2020) to this example, consider the following system of recurrence equations, where at time $t = 0$, we set

$$\begin{cases} x_0 = s \\ v_0 = \hat{\eta} \\ w_0 = \bar{\eta} \end{cases} \tag{1}$$

and for all $t = 1, 2, \ldots$, with $t \in \mathbb{N}_{\geqslant 1}$, we set

$$\begin{cases} x_{t+1} = x_t \\ v_{t+1} = v_t - \gamma\, \partial_1 \mathrm{F}(v_t, w_t, \alpha, \beta) \\ w_{t+1} = w_t - \gamma\, \partial_2 \mathrm{F}(v_t, w_t, \alpha, \beta), \end{cases} \tag{2}$$

where $\mathrm{F} : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}$,

$$\mathrm{F}(y, z, \xi, \tau) \coloneqq \frac{1}{2}(y - \xi f(s))^2 + \frac{1}{2}(z - \tau f(y))^2.$$

System (2) determines a sequence of tuples $(v_t, w_t)_{t \geqslant 0}$; the sequence $(x_t)_{t \geqslant 0}$ is constant and equals the input s. Determining the convergence behavior of $(v_t, w_t)_{t \geqslant 0}$ is what is sought. If this system converges, we call

$$(v_\infty, w_\infty) \coloneqq \lim_{t \to \infty} (v_t, w_t)$$

the *output* of the PCN. (The general definition of a PCN is given in Definitions B.1 to B.3 in Appendix B.)

Notice that a simpler PCN than the above one consisting of only of two nodes is trivial, because such a system essentially consists of a single sequence analogous to the sequence $(w_t)_{t \geqslant 0}$, and this equation can be explicitly solved, as it is a linear recurrence equation.

It is worthwhile to interpret the sequence $(v_t, w_t)_{t \geqslant 0}$ as a (two-dimensional) discrete dynamical system on the phase space $\mathbb{R}^2$. Thus, we introduce the following notation. Let

$$\Psi_{s,\alpha,\beta}(y, z) \coloneqq (y - \gamma\, \partial_1 \mathrm{F}(y, z, \alpha, \beta),\, z - \gamma\, \partial_2 \mathrm{F}(y, z, \alpha, \beta)),$$

which we call a *PCN map*. Thus, obviously,

$$(v_t, w_t)_{t \geqslant 1} = (\Psi^t_{s,\alpha,\beta}(\hat{\eta}, \bar{\eta}))_{t \geqslant 1}, \tag{3}$$

where $(v_t, w_t)_{t \geqslant 0}$ is determined by (1) and (2). (Definition B.5 indicates how an analogous operator needs to be defined for a general PCN, and Proposition B.8 shows that an analogous statement holds, indicating how a general PCN can be represented as discrete dynamical system.)

Computing the derivatives of F, we explicitly obtain:

$$\begin{cases} \partial_1 \mathrm{F}(y, z, \xi, \tau) = y - \xi f(s) - f'(y)\tau[z - \tau f(y)], \\ \partial_2 \mathrm{F}(y, z, \xi, \tau) = z - \tau f(y). \end{cases}$$

Thus, the sequences $(v_t, w_t)_{t\geqslant 0}$ from system (2) are determined by the equations

$$\begin{cases} v_{t+1}=v_t-\gamma\{v_t-\alpha f(s)-f'(v_t)\beta[w_t-\beta f(v_t)]\}, \\ w_{t+1}=w_t-\gamma\{w_t-\beta f(v_t)\}, \end{cases} \tag{4}$$

with, by (3),

$$\Psi_{s,\alpha,\beta}(y,z) = (y-\gamma\{y-\alpha f(s)-f'(y)\beta[z-\beta f(y)]\}, \\ z-\gamma\{z-\beta f(y)\}). \tag{5}$$

(The general formula for the derivative is provided in Proposition B.9, and Remark B.10 shows what the analogous general form of (2) as well as of $\Psi_{s,\alpha,\beta}$ is.)

Remark 2.1. Because the gradient of F is essential for defining the dynamics of a PCN, it is helpful to introduce a number of further mathematical objects that make up F, in order to be able to relate the gradient dynamics to what happens in the network.

Thus, we define the *belief functions*

$$\hat{\mu}(y,z,\xi,\tau) \coloneqq \xi f(s), \quad \bar{\mu}(y,z,\xi,\tau) \coloneqq \tau f(y)$$

and *error functions*

$$\hat{\varepsilon}(y,z,\xi,\tau) \coloneqq y-\hat{\mu}(y,z,\xi,\tau), \\ \bar{\varepsilon}(y,z,\xi,\tau) \coloneqq z-\bar{\mu}(y,z,\xi,\tau).$$

With these, let

$$\begin{aligned} \hat{\mathrm{F}}(y,z,\xi,\tau) &\coloneqq \frac{1}{2}(y-\hat{\mu}(y,z,\xi,\tau))^2 \\ &= \frac{\hat{\varepsilon}(y,z,\xi,\tau)^2}{2} \\ \bar{\mathrm{F}}(y,z,\xi,\tau) &\coloneqq \frac{1}{2}(z-\bar{\mu}(y,z,\xi,\tau))^2, \\ &= \frac{\bar{\varepsilon}(y,z,\xi,\tau)^2}{2}, \end{aligned} \tag{6}$$

so F can be written as $\mathrm{F}(y,z,\xi,\tau) = \hat{\mathrm{F}}(y,z,\xi,\tau) + \bar{\mathrm{F}}(y,z,\xi,\tau)$. In Appendix B, Definition B.2, the abstract formulation of these objects is given.

Using these objects, the interpretation of F is the following: After each non-input node (i.e., hidden node or output node) is initialized, the PCN sequences $(\hat{\mu}(x_t))_{t\geqslant 0}$ and $(\bar{\mu}(x_t))_{t\geqslant 0}$ are thought to represent an initial guess that the network makes about the node values; this can be represented as the secondary "belief node", i.e., the value of an original node as given by the belief function. In time, these guesses are updated until an equilibrium is reached. To contrast the belief nodes from other nodes, we call the original nodes s, v_t, w_t "value nodes". A new representation of the network that accounts for these nodes looks as in Figure 2. Thus, these newly introduced objects, while not strictly necessary for the derivation of our results, are important in order attain an intuitive understanding of PCNs, as it was layed out in (Whittington & Bogacz, 2017; Bogacz, 2017; Song et al., 2020; Salvatori et al., 2021b).

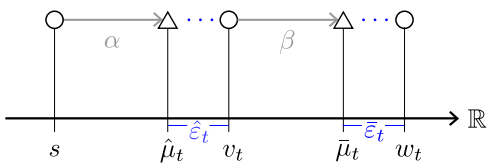

Figure 2. An illustration of how the value nodes (circles), the belief nodes (triangles), and the errors (in blue) change with time, via (4). While the input s stays fixed, the other nodes move around with time. We used the short-hand notation $\hat{\mu}_t \equiv \hat{\mu}(v_t, w_t, \alpha, \beta)$, $\bar{\mu}_t \equiv \bar{\mu}(v_t, w_t, \alpha, \beta)$, $\hat{\varepsilon}_t \equiv \hat{\varepsilon}(v_t, w_t, \alpha, \beta)$, and $\bar{\varepsilon}_t \equiv \bar{\varepsilon}(v_t, w_t, \alpha, \beta)$. Notice that $\hat{\varepsilon}_t$ and $\bar{\varepsilon}_t$ denote signed differences.

3. Phase Space Analysis and Convergence for Prediction

In this section, we prove sufficient conditions for convergence of the concrete example exhibited before. For the general case, we refer to Appendix C.

The dynamical system $(\Psi_{s,\alpha,\beta}, \mathbb{R}^2)$ that determines the iterations (4) has a single (unique) fixed point

$$\begin{cases} v^\star = \alpha f(s), \\ w^\star = \beta f(v^\star) = \beta f(\alpha f(s)). \end{cases} \tag{7}$$

Thus, if f is C^1, i.e., continuously differentiable, and therefore $\Psi_{s,\alpha,\beta}$ a continuous map, the output of the PCN, (v_∞, w_∞), if it exists, must be equal to the fixed point $(v^\star, w^\star)$. (This is a standard textbook argument. We include it here nonetheless for a general audience not accustomed to dynamical systems: If $(v_t, w_t)_{t\geqslant 0}$ has the limit (v_∞, w_∞), then $(\Psi_{s,\alpha,\beta}(v_t, w_t))_{t\geqslant 0}$ has the limit $\Psi_{s,\alpha,\beta}(v_\infty, w_\infty)$ by the continuity of $\Psi_{s,\alpha,\beta}$; but $(v_t, w_t)_{t\geqslant 0}$ and $(\Psi_{s,\alpha,\beta}(v_t, w_t))_{t\geqslant 0}$ differ by a single shift, so must have the same limit. Hence, $\Psi_{s,\alpha,\beta}(v_\infty, w_\infty) = (v_\infty, w_\infty)$, i.e., $(v_\infty, w_\infty) = (v^\star, w^\star)$, as the fixed point $(v^\star, w^\star)$ is unique. For the general version see Proposition C.2.)

Therefore, our qualitative understanding of PCNs at this point is described by Figure 3. We depicted the fixed point to be stable in this figure, but this does not need to be the case. Furthermore, other objects, that are not locally visible neighborhoods of the fixed point may exist; Section 5 exhibits an example were both of these situations occur simultaneously. Though the absence of further fixed points limits the complexity of the phase portrait to some degree.

Consider now an FCFN consisting of the same data (architecture, activation function, and weights—initializations and step size are not needed), i.e., an FCFN that is also represented by the diagram from Figure 1. For the same input $s \in \mathbb{R}$, the output of such an FCFN is given by the formula $\beta f(\alpha f(s))$, obtained by successively following

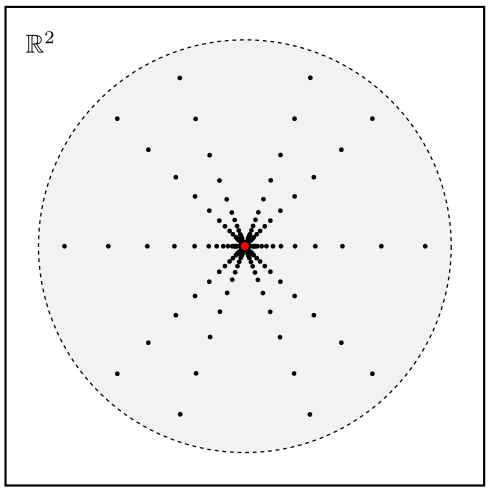

Figure 3. A hypothetical local phase portrait, around the fixed point $(v^\star, w^\star)$ of the discrete dynamical system $(\Psi_{s,\alpha,\beta}, \mathbb{R}^2)$, indicated by the red dot. This figure is purely illustrative. The dashed circle represent a neighborhood around the fixed point. The dotted lines indicate a few representative orbits that converge to the fixed point. Note that it is not always the case for the orbits to converge to the fixed point, i.e., it is not always the case for the fixed point to be stable, as highlighted in Section 5. Furthermore, even if the fixed point is stable, there are other types of stability than the radial stability depicted here.

how the input s changes layer by layer.

Thus, if the PCN converges, its value is given by the corresponding FCFN made up of the same data. (Notice that we assume that FCFNs do not apply the activation function after the last layer, which in some parts of the literature is done. The general definition of an FCFN is recapitulated in Definition C.3, while Proposition C.5 states in general that the fixed point of a PCN is equal to the output of the corresponding FCFN.)

A comparison between FCFNs and PCNs in terms of their evolution in time can be seen in Figure 4. We chose a different example for this illustration than our running example from Figure 1, to better indicate the complicated dependencies that can arise in time. For the running example the dependencies would be almost trivial (yet it still has interesting dynamics, illustrating the complexity of these models). This particular example is subsumed by the general setup of PCNs from Appendix A.

From the perspective of statistical learning theory, there is therefore no need to leave the hypothesis class of FCFNs, since PCNs do not increase this space; rather, their importance lies in their biological faithfulness, in particular, their training procedure and its connections to backpropagation, which, as mentioned in the introduction, has already been analyzed to a certain degree.

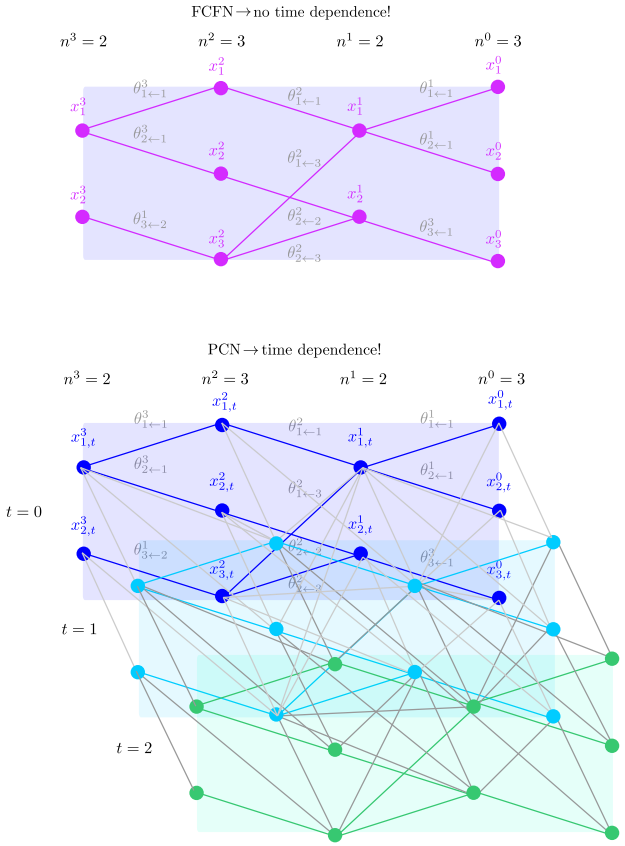

Figure 4. A comparison between the dynamics of a FCFN vs. a PCN in the prediction stage. The input is on the left and the output is on the right For a PCN three timesteps are depicted. The gray arrows indicate the dependencies of each node on the nodes from the previous time step. If a PCN converges to an output it means that right-most nodes do not change significantly anymore between different "time-slices" for a sufficiently large values of t.

This issue has a further dramatic consequence: *If one starts the iteration on the fixed point, one (obviously) has convergence; but in that case, there is no point in using PCNs, as the computations are identical to the ones produced by an FCFN, so one looses biological faithfulness in this case.*

If f is furthermore C^2, so that $\Psi_{s,\alpha,\beta}$ is C^1 by (5), one can apply a non-bifurcation theorem to $\Psi_{s,\alpha,\beta}$ (see Theorem D.1), which gives conditions when fixed points of a discrete dynamical system, as well as their stability, persist under small perturbation of parameters of the system. The proof idea is to notice that for a good choice of weights (which are the parameters in the sense of the mentioned theorem) the system becomes linear, where it can be fully understood; the above theorem then show that the results persist under small perturbations of the parameters. (A general version of this theorem is formulated in Theorem C.7).

Theorem 3.1 (Prediction-stage convergence criteria for a PCN)**.** *If f is C^2 and $\hat{\eta}$, $\bar{\eta}$, α, and β are sufficiently small, and the step size is in $\gamma \in (0,1)$, then the iterations of (2) converge (to the fixed point that coincides with the output of the corresponding FCFN as shown previously).*

Proof. By (3), we consider $\Psi_{s,0,0}$, i.e., $\alpha = 0$, $\beta = 0$. It follows from (5) that $\Psi_{s,0,0}$ is a linear map, $\Psi_{s,0,0}(\hat{\eta}, \bar{\eta}) = ((1-\gamma)\hat{\eta}, (1-\gamma)\bar{\eta})$, and the system has a (unique) fixed point $(0,0) \in \mathbb{R}^2$. Furthermore it has a single eigenvalue $1-\gamma$, which lies in the interval $(0,1)$. Hence for any initial value, all trajectories converge to the fixed point $(0,0)$, which is asymptotically stable. Since f is C^2 and thus the PCN map is C^1, we apply the non-bifurcation Theorem D.1 to $h(\alpha, \beta, \bar{\eta}, \hat{\eta}) \coloneqq \Psi_{s,\alpha,\beta}(\bar{\eta}, \hat{\eta})$ (thus $p = 2$ and $n = 2$ in Theorem D.1) to conclude the proof. □

4. Training stage

We first fix a training regime: For PCNs, multiple possibilities are mentioned in (Song et al., 2020), with varying degrees of biological faithfulness. We use a generalization (allowing arbitrary initializations and step sizes) of the training procedure that is called *Fa-Z-IL* (**f**ully **a**utonomous **z**ero **i**nference **l**earning) in (Song et al., 2020), because it is the most biologically faithful among the presented variants, while still recovering backpropagation exactly.

The development parallels the previous two sections: First, a general gradient system is considered, then its dynamical systems nature is discussed, and convergence is established. Thus, we proceed at a faster pace in this section. Furthermore, it is possible to generalize these results as well, but we have omitted such an undertaking. For simplicity, convergence for a dataset consisting of a single training example is discussed.

To train such a PCN via Fa-Z-IL, all the mathematical objects from the prediction stage are employed, except the initializations $\bar{\eta}$ for the node w, as this node is constant to the output $q \in \mathbb{R}$ in this stage. Additionally, a second step size $\bar{\gamma} > 0$ is given, and instead of fixed weights α, β, we now have entire sequences of weights $(\alpha_t)_{t\geqslant 0}$, $(\beta_t)_{t\geqslant 0}$ as well as weight initializations $\hat{\rho}, \bar{\rho} \in \mathbb{R}$ in addition to the node initialization. Thus, at time $t = 0$, we set

$$\begin{cases} x_0 = s \\ v_0 = \hat{\eta}, \\ w_0 = q \\ \alpha_0 = \hat{\rho} \\ \beta_0 = \bar{\rho} \end{cases} \tag{8}$$

and for all $t = 1, 2, \ldots$, we set

$$\begin{cases} x_{t+1} = x_t \\ v_{t+1} = v_t - \gamma\, \partial_1 \mathrm{F}(v_t, w_t, \alpha_t, \beta_t), \\ w_{t+1} = w_t, \\ \alpha_{t+1} = \alpha_t - \bar{\gamma}\, \partial_3 \mathrm{F}(v_t, w_t, \alpha_t, \beta_t), \\ \beta_{t+1} = \beta_t - \bar{\gamma}\, \partial_4 \mathrm{F}(v_t, w_t, \alpha_t, \beta_t). \end{cases} \tag{9}$$

Obviously, one can eliminate the constant sequence $(x_t)_{t\geqslant 0}$ like before to arrive at a dynamical system $\Psi_{s,p} : \mathbb{R}^3 \to \mathbb{R}^3$, which now has both input and output as parameters, whereas weights are allowed to vary; though for brevity we skip the discussion of a PCN map for the training stage entirely in this section. Computing the derivatives of F also for the weights, one obtains explicitly:

$$\begin{aligned} v_{t+1} &= v_t - \gamma \left\{ v_t - \alpha_t f(s) - f'(v_t)\beta_t \left[q - \beta_t f(v_t) \right] \right\}, \\ \alpha_{t+1} &= \alpha_t + \bar{\gamma} f(s)(v_t - \alpha_t f(s)), \\ \beta_{t+1} &= \beta_t + \bar{\gamma} f(v_t)(q - \beta_t f(v_t)), \end{aligned} \tag{10}$$

which determines a sequence $(v_t, \alpha_t, \beta_t)_{t\geqslant 0}$.

If the sequence $(v_t, \alpha_t, \beta_t)_{t\geqslant 0}$ converges to some fixed point, then, by definition, the pair (s, q) has been *learned*. This definition is meaningful, because (as it will turn out a few paragraphs later) the weights set in his way lead to an FCFN that outputs q when given s as input.

In contrast to (4), there is no single expression anymore of the fixed point $(v^\star, \alpha^\star, \beta^\star) \in \mathbb{R}^3$ of (10), as to solve

$$\begin{cases} v^\star - \alpha^\star f(s) - f'(v^\star)\beta^\star [q - \beta^\star f(v^\star)] = 0, \\ f(s)(v^\star - \alpha^\star f(s)) = 0, \\ f(v^\star)(q - \beta^\star f(v^\star)) = 0, \end{cases} \tag{11}$$

we have to make a case distinction whether $f(s), f(v^\star)$ are zero or not:

- If $f(s) \neq 0$, $f(v^\star) \neq 0$, then the fixed point fulfills the equations
$$\begin{cases} q = \beta^\star f(v^\star)), \\ v^\star = \alpha^\star f(s) \end{cases} \tag{12}$$
and is analogous to the fixed points of the system (2), given in (7) (with $w^\star$ denoted by q, and the weights starred).

- If $f(s) \neq 0$, $f(v^\star) = 0$, then the fixed point elements fulfill the equations
$$\begin{cases} v^\star = \alpha^\star f(s) \\ f'(v^\star)\beta^\star q = 0. \end{cases}$$

- If $f(s) = 0$, $f(v^\star) \neq 0$, then the fixed point fulfills the equations
$$\begin{cases} v^\star = 0 \\ q = \beta^\star f(v^\star). \end{cases}$$

- If $f(s) = 0$, $f(v^\star) = 0$, then the fixed point fulfills the equation
$$v^\star = f'(v^\star)\beta^\star q.$$

If the learning process converges such that $f(v^\star) \neq 0$ and $f(s) \neq 0$ hold, then we are in the case of (12). This then implies that if the network converges in the prediction stage, for an input s, we obtain an output q. This shows why assuming convergence of (10) is the correct concept for training for PCNs and extends the understanding of fixed points from (Whittington & Bogacz, 2017; Song et al., 2020; Rosenbaum, 2021).

Analogously, we can prove that convergence is possible in the training stage as well, by again appealing to the non-bifurcation Theorem D.1, as long as the initializations are sufficiently small.

Theorem 4.1 (Training-stage convergence criteria for a PCN)**.** *If f is C^2 and $\hat{\eta}$, $\bar{\eta}$, $\hat{\rho}$, and $\bar{\rho}$ are sufficiently small, the step sizes are in $\gamma, \bar{\gamma} \in (0, 1)$, and the conditions for the correct fixed-point case are met, then the iterations of (9) converge (to the fixed point (12)).*

5. Non-Convergence

While the results from the previous sections (in particular pertaining to the training stage) were also true for PCNs defined on arbitrary architectures, as we indicated by giving pointers to the general theorem from the appendices, in this section, we continue with a refined analysis for the prediction stage of the example that we have so far considered. We highlight parameter regimes and show the existence of step sizes where convergence cannot be obtained.

The fixed points whose eigenvalues do not lie on the unit circle, i.e., for which the previous non-bifurcation Theorem D.1 holds, are called *hyperbolic*. This is a term from dynamical system theory to denote fixed points with convenient theoretical properties: The local behavior around hyperbolic fixed points can be well described by general theorems, as no *center manifolds* exist, which potentially complicate the analysis.

Since we already showed in the previous section that, at least for sufficiently small weights and initializations, all trajectories converge to the fixed point, in this section, we must obviously investigate how hyperbolicity can be violated if we seek to exhibit a pathological behavior.

Suppose that the activation function f is C^2. Then, the linearization $J := D(\Psi_{s,\alpha,\beta})(v^\star, w^\star)$ at the fixed point $(v^\star, w^\star) \in \mathbb{R}^2$ exists and is given by:
$$J = \begin{bmatrix} A & \gamma\beta f'(v^\star) \\ \gamma\beta f'(v^\star) & 1-\gamma \end{bmatrix},$$
where
$$A := 1 - \gamma(1 + 2\beta w^\star - \beta^2[f''(v^\star)f(v^\star) - f'(v^\star)^2]).$$

Notice that the linearization depends only implicitly on the weight α and the input s, as the values of the fixed points are determined by these. We can thus (from a bifurcation point of view) consider β (on which $w^\star$ also depends) and γ to be the only parameters. We now investigate these *bifurcations*; these are parameter region which, when the parameter varies within them, induce qualitative changes in the phase portrait.

If $\beta = 0$, we obtain a double eigenvalue $1 - \gamma$. Thus, if $\gamma \in (0, 2)$, the system converges to the fixed point, since the (absolute value of the real part of the) eigenvalue is less than 1. For $\gamma = 0$ and $\gamma = 2$, one obtains (double) unital eigenvalues, and the fixed point become non-hyperbolic. These are called *fold* and *period-doubling bifurcations*, respectively. Though we do not pursue those bifurcation points in more detail here.

If $\beta \neq 0$, one can similarly find a pair $(\bar{\beta}, \bar{\gamma})$ of parameters and values for the fixed point, such that we obtain a complex conjugated pair of unital eigenvalues, and the fixed point again becomes non-hyperbolic.

These situations are the only ones that can occur *generically*, see Figure 6. (The term "generic" has a well-defined formal meaning within the context of dynamical systems theory; we refer the reader to (Ruelle, 1989), Chapter 8.7, for a detailed discussion of the topological aspects, as well as how this concept interacts with probabilistic meanings. Informally, it can be thought to mean "typically".)

We now consider $\bar{\beta}$ to be fixed, so that we have a single parameter γ left that is allowed to vary and induce bifurcations. Because the fixed point is now non-hyperbolic, and thus the non-bifurcation theorem mentioned above breaks down, bifurcations can appear. Informally, this means that, as the parameters change, the phase portrait can radically change (and, in particular, the stability of the fixed point to which the trajectory must converge, if they converge at all, can change and become unstable). Bifurcations of the type identified in this case are called *(generic) Neimark-Sacker bifurcations*. Note that because this type of bifurcation requires a complex conjugated pair of unital eigenvalues, it can only appear in phase spaces of dimension at least 2. Specific known behavior is associated with any type of Neimark-Sacker bifurcation points:

Theorem 5.1 (System behavior at a Neimark-Sacker bifurcation)**.** *For any generic two-dimensional, one-parameter system $(v, w) \mapsto h(v, w, \gamma)$, having at parameter $\gamma^\star$ a fixed point $(v^\star, w^\star)$ with complex conjugated unital eigenvalues $e^{\pm\theta i}$, there is a neighborhood of $(v^\star, w^\star)$ in which a unique closed invariant curve bifurcates from $(v^\star, w^\star)$ as γ passes through $\bar{\gamma}$.*

A statement and proof can be found in, e.g., (Kuznetsov, 2013), Theorem 4.6. We now prove one of the main theorems of this article:

Theorem 5.2 (System behavior at a Neimark-Sacker bifurcation)**.** *There exist PCNs (and PCN maps) that have points in their phase space that do not converge, as the fixed point is not stable and there exists a closed invariant curve around the fixed point, containing no recurrent points.*

Proof. Having found an input, weights, and step sizes such that the Jacobian has a unital pair of eigenvalues, we apply the theorem above, where everything is fixed except the step size. The existence of an invariant curve precludes global stability of the fixed point: If one would start the iterations outside the invariant curve, there is no possibility anymore to reach the fixed point. This establishes non-convergence. (Another line of arguing would be to assume that we start on the curve itself. Because it is invariant, we can not escape it and hence also not converge.)

To see that orbits on the curve have a more complicated behavior, i.e., are non-recurrent, we use topological methods and theorems from (Hale, 2004), Section 6. By direct computation (or by Remark B.6), one can see that system (5) is itself a gradient system. Because it has a single fixed point, it is a *strongly gradient system*, which means the set of recurrent points equals the set of its fixed points. But since there is just one fixed point, which does not lie on the curve, this means that the curve contains only non-recurrent points. (This also shows that finding an initial value that lies on the curve and tracking it with finite numerical precision might be a very difficult task, which makes it unsuitable for an explicit non-convergence example). □

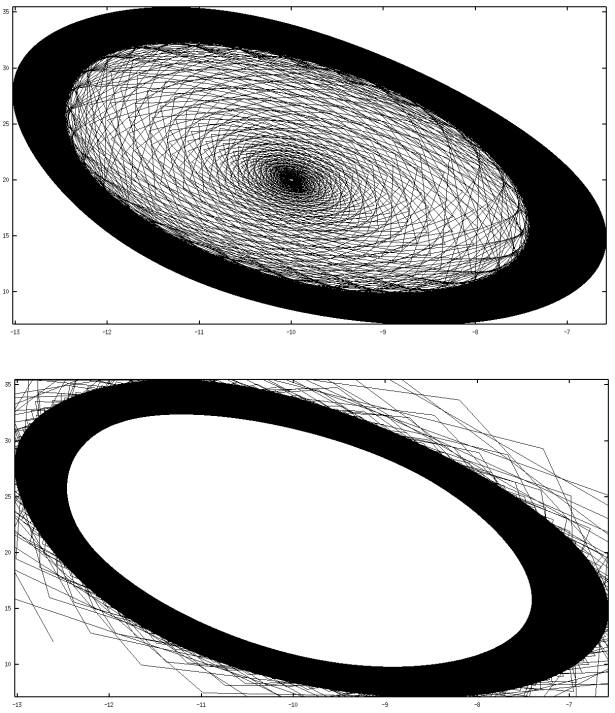

Figure 5. Various trajectories in $\mathbb{R}^2$ of $\Psi_{s,\alpha,\beta}$ for $s = 1$, $\alpha = -10$, $\beta = 0.2$, $\gamma = 1.2$, and $f(x) \coloneqq x^2$. In the diagram above, all initial values lie within the region enclosed by the invariant curve and have been chosen close to the unstable fixed point, which appears as a small white point in the center. The direction of the trajectories is not immediately clear from the finished diagram, but plotting it step-by-step indicates that the trajectories move away from the fixed point. In the diagram below, all initial values lie outside the invariant curve, approaching it as time increases.

In fact, it is possible to explicitly find examples of non-convergence, even beyond the discussed setting. Let, e.g., $s = 1$, $\alpha = -10$, $\beta = 0.2$, $\gamma = 1.2$, and $f(x) \coloneqq x^2$, which is C^2, and consider the trajectories for various initializations of the PCN, i.e., the initial value of $\Psi_{s,\alpha,\beta}$. We can see in Figure 5 that the iterations approach a curve, both when starting within the region that the curve encloses, as well as when starting outside. Theorem 5.2 provides the formal underpinning of this theorem. In particular, because $\Psi_{s,\alpha,\beta}$ is injective and continuous, it follows that the iterations cannot "jump" over the invariant curve and are forever confined within it.

6. Conclusion

The overarching take-away message for this investigation is that while neuroscience can offer important impetus, it is

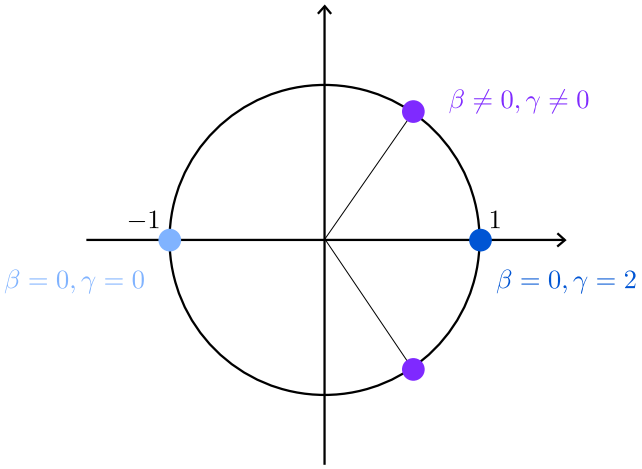

Figure 6. The possible unital eigenvalues of the linearization of $\Psi_{s,\alpha,\beta}$ at the fixed point. The violet dots indicate a family of pairs of complex conjugated eigenvalues, one for each fixed β. Cf. (Kuznetsov, 2013).

equally important to be aware of mathematical properties and pathologies that can arise within the proposed models and that may invalidate some of the aspirations that these models are deemed to fulfill.

The fixed prediction assumption, implicitly used in the previous works on PCNs, is an example of this, as we have shown that specific instances of PCNs exist that have a pathological non-convergence behavior for certain parameter regimes.

Could such counterexamples be found for other, more complicated architectures? By having a single parameter γ that we allow to vary in the discussed counterexample, we are in the case of a co-dimension 1 bifurcation. Informally, this is the number of independent conditions determining the bifurcation, see (Kuznetsov, 2013), Chapter 2.3. As the parameters increase, the co-dimensions increases, and it becomes increasingly harder and more technically involved to carry out a bifurcation analysis; problems of co-dimension 3 and higher are currently still open problems, see (Kuznetsov, 2013), Chapter 2.4. This precludes doing the same analysis as in this section for PCNs with larger architecture, as we would encounter such bifurcations of higher co-dimension, whose behavior is not yet fully understood. For other counterexamples, it is therefore likely that other techniques need to be employed, which call for an analysis of other bifurcation points.

The existence of a counterexample to convergence triggers the necessity of providing theoretical guarantees for convergence, as one can not take the fixed prediction assumption for granted. Employing a novel view on PCNs by casting them as dynamical systems, we have provided basic conditions that are sufficient for convergence (see the appendices for general formulations of such theorems), both in the prediction and the training stage. These also provide a justification why some of the observed experiments in the literature can be expected to succeed from an optimization point of view: If the weights and the initial values are small enough, everything works out. Intriguingly, the regimes for which convergence is proved assume that the step size is less than 1, which is consistent with what one would expect for an Euler discretization to converge by keeping local errors small. Yet, equivalence to backpropagation assumes a step size of exactly 1, see (Song et al., 2020). Further investigation is necessary to reconcile these findings.

Furthermore, the negative example concerning non-convergence can be extended to larger PCNs with more nodes. We hypothesize that it is possible to prove the existence of similar examples for the training stage as well. The general setup that we developed for the prediction stage can be carried out for the training stage as well, including a generalization of Theorem 4.1. It is possible to use quantitative versions of the implicit function theorem to obtain explicit guarantees on the sizes of the weight and initializations the are sufficient for convergence. This is ongoing work.

Acknowledgements

We thank the anonymous reviewers for helpful suggestions. This work was partially supported by the Alan Turing Institute under the EPSRC grant EP/N510129/1, the AXA Research Fund, and the EPSRC grant EP/R013667/1.

References

Bartunov, S., Santoro, A., Richards, B., Marris, L., Hinton, G. E., and Lillicrap, T. Assessing the scalability of biologically-motivated deep learning algorithms and architectures. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Bogacz, R. A tutorial on the free-energy framework for modelling perception and learning. *Journal of Mathematical Psychology*, 76:198–211, 2017.

Broer, H. and Takens, F. *Dynamical Systems and Chaos*, volume 172. Springer Science & Business Media, 2010.

Fiedler, B. and Scheurle, J. *Discretization of Homoclinic Orbits, Rapid Forcing and "Invisible" Chaos*, volume 570. American Mathematical Society, 1996.

Friston, K. Learning and inference in the brain. *Neural Networks*, 16(9):1325–1352, 2003.

Friston, K. The free-energy principle: A unified brain

theory? *Nature Reviews Neuroscience*, 11(2):127–138, 2010.

Friston, K. Does predictive coding have a future? *Nature Neuroscience*, 21(8):1019–1021, 2018.

Hale, J. K. Stability and gradient dynamical systems. *Revista Matemática Complutense*, 17(1):7–57, 2004.

Hassabis, D., Kumaran, D., Summerfield, C., and Botvinick, M. Neuroscience-inspired artificial intelligence. *Neuron*, 95(2):245–258, 2017.

Kadmon, J., Timcheck, J., and Ganguli, S. Predictive coding in balanced neural networks with noise, chaos and delays. In *Advances in Neural Information Processing Systems*, volume 33, pp. 16677–16688, 2020.

Krabs, W. *Dynamical Systems: Stability, Controllability and Chaotic Behavior*. Springer Science & Business Media, 2010.

Kriegeskorte, N. Deep neural networks: A new framework for modeling biological vision and brain information processing. *Annual Review of Vision Science*, 1:417–446, 2015.

Kuznetsov, Y. A. *Elements of Applied Bifurcation Theory*, volume 112. Springer Science & Business Media, 2013.

Lillicrap, T. P., Santoro, A., Marris, L., Akerman, C. J., and Hinton, G. Backpropagation and the brain. *Nature Reviews Neuroscience*, 21(6):335–346, 2020.

Liverani, C. Implicit function theorem (a quantitative version), accessed 20 December 2021. [https://www.mat.uniroma2.it/∼liverani/Calcolo1-2016/implicit.pdf].

Palis, J. J. and De Melo, W. *Geometric Theory of Dynamical Systems: An Introduction*. Springer Science & Business Media, 2012.

Rao, R. P. N. and Ballard, D. H. Predictive coding in the visual cortex: A functional interpretation of some extra-classical receptive-field effects. *Nature Neuroscience*, 2 (1):79–87, 1999.

Rosenbaum, R. On the relationship between predictive coding and backpropagation. *arXiv preprint arXiv:2106.13082*, 2021.

Ruelle, D. *Elements of Differentiable Dynamics and Bifurcation Theory*. Academic Press, 1989.

Salvatori, T., Song, Y., Hong, Y., Sha, L., Frieder, S., Xu, Z., Bogacz, R., and Lukasiewicz, T. Associative memories via predictive coding. In *Advances in Neural Information Processing Systems*, volume 34, 2021a.

Salvatori, T., Song, Y., Lukasiewicz, T., Bogacz, R., and Xu, Z. Predictive coding can do exact backpropagation on convolutional and recurrent neural networks. *arXiv preprint arXiv:2103.03725*, 2021b.

Song, Y., Lukasiewicz, T., Xu, Z., and Bogacz, R. Can the brain do backpropagation?—Exact implementation of backpropagation in predictive coding networks. In *Advances in Neural Information Processing Systems*, volume 33, pp. 22566–22579, 2020.

Whittington, J. C. R. and Bogacz, R. An approximation of the error backpropagation algorithm in a predictive coding network with local Hebbian synaptic plasticity. *Neural Computation*, 29(5):1229–1262, 2017.

Whittington, J. C. R. and Bogacz, R. Theories of error back-propagation in the brain. *Trends in Cognitive Sciences*, 23(3):235–250, 2019.

Appendices

A. Notation and Dynamical Systems Terminology

In our general setup of PCNs, we often use two and three indices to index vectors, where one index counts the layers, and the other one the nodes (within a certain layer) or a pair of nodes, respectively. We call a multi-indexed vector a *family of vectors*, since it generalizes notation for matrices (two indices with the same number of nodes in each layer) or finite sequences of matrices (three indices), respectively. Thus, such families of vectors offer a unified way to handle various multi-indexed vectors.

The notation that we employ for a family of vectors, of length $q := |Q|$, with two indices is of the form $(z_n^k)_{(k,n)\in\mathcal{Q}}$. Here, Q is a finite set of index tuples, which is not necessarily of "rectangular" form, since $(z_n^k)_{(k,n)\in\mathcal{Q}}$ in general is not a matrix. Because we do not use any of the matrix operations on such families of vectors, we treat them throughout the text simply as vectors, e.g. $(z_n^k)_{(k,n)\in\mathcal{Q}} \in \mathbb{R}^q$, where the (implicit) ordering on Q determines the ordering of $(z_n^k)_{(k,n)\in\mathcal{Q}}$ as a vector in $\mathbb{R}^q$. For three indices, $(z^k_{(n,m)})_{(k,n,m)\in\mathcal{P}}$, everything is analogous. An example of a family of vectors (with three indices) is, e.g., (14).

An (infinite) sequence of a family of vectors is denoted by $\big((z_{n,t}^k)_{(k,n)\in\mathcal{Q}}\big)_{t\in\mathbb{N}}$, where $\mathbb{N} := \{0,1,2,\ldots\}$. Furthermore, let $\mathbb{N}_{\geqslant 1} := \{1,2,\ldots\}$, and $[m] := \{1,\ldots,m\}$ for some $m \in \mathbb{N}_{\geqslant 1}$. The symbol ":=", as used just now, means definitional equality.

We next review basic notation from the theory of discrete dynamical systems. Within our context, a *discrete dynamical systems* is a tuple $(\mathbb{R}^d, \varphi)$, where $\varphi : \mathbb{R}^d \to \mathbb{R}^d$ is a continuous map. The dynamics that this tuple generates consists of the sequence

$$(x_0, \varphi(x_0), \varphi^2(x_0), \ldots) \tag{13}$$

for some starting value $x_0 \in \mathbb{R}^d$, which is captured by the equations $\varphi_{n+1} = \varphi(x_n)$, $n \in \mathbb{N}$. This sequence is called *orbit through* x_0. Understanding and classifying the behavior of (13) for $n \to \infty$, as x_0 takes different values, is what is sought. For $x_0 \in \mathbb{R}^d$, the *ω-limit set* is denoted by $\omega(x_0)$ and consists of all accumulation points of the sequence (13). An *accumulation point* is a point $y \in \mathbb{R}^d$ such that there exists a subsequence of (13) that converges to y as $n \to \infty$. A fixed point $\check{x}$ is a point such that $\varphi(\check{x}) = \check{x}$, i.e., a point such that if it appears in sequence (13), the sequence is constant equal to it. A fixed point $\check{x}$ is *asymptotically stable*, if there exists an open set $O \subseteq \mathbb{R}^d$ containing $\check{x}$, such that $\lim_{n\to\infty} \varphi(x) = \check{x}$ for any $x \in O$. The *phase space* is the domain $\mathbb{R}^d$ that consists of the objects of interest, e.g., points that are asymptotically stable fixed points, or *invariant* sets, i.e., subsets $M \subseteq \mathbb{R}^d$ such that $\varphi(M) \subseteq M$. A point z is *recurrent* if there exists an $m \geqslant 1$ such that $\varphi^m(z) = z$ and $\varphi^n(z) \neq z$ for $n = 1,\ldots,m-1$ in case $m \geqslant 2$,.

B. The General Dynamical Systems Perspective

Here, we show in full generality how one arrives at a dynamical system, given a PCN with fixed weights (i.e., no training is involved) as introduced in (Song et al., 2020). Because we need to reason about PCNs as being mathematical objects in their own right, we formalize them below, by collecting in a tuple the formal pieces of data that make up a PCN and specifying the rules of computation.

The setup in (Whittington & Bogacz, 2017; Bogacz, 2017) differs slightly from the setup in (Song et al., 2020; Salvatori et al., 2021b); for example, in the former, the seed (see Definition B.3) is chosen randomly (see Remark B.4 for a short explanation), while in the latter, no particular choice of the seed of the non-input nodes is specified. In (Song et al., 2020), three gradient descent training methodologies are presented, whereas (Salvatori et al., 2021b) only one of these training methodologies is used.

We follow the established convention from the cited papers of numbering the layers in descending order, the 0-th layer being the output layer. Because the training stage parallels the theory for the prediction stage, we only develop the theory for the latter here (see also Section 6 for more information on the training stage).

Because the entire definition of a PCN is cumbersome to wield, we brake it down in smaller pieces. Because we use superscript indices, there is a danger of conflating this notation with the operation of exponentiation. We therefore establish the convention that, unless explicitly stated, a superscript that appears directly on a single symbol is an index, not an exponent, as is, e.g., the case for n^L below. We number the layers of the network in descending numbering, in order to be consistent with (Song et al., 2020).

Definition B.1 (PCN specification)**.** Let

- $L \geqslant 1$, $n^L, \ldots, n^0 \in \mathbb{N}_{\geqslant 1}$ be the *neural architecture;*
- $f : \mathbb{R} \to \mathbb{R}$ differentiable be the *activation function;*
- $\gamma \in \mathbb{R}$ be the *PCN step size;*

which form the model hyperparameters and

- $\forall \ell \in \{L-1,\ldots,0\}\, \forall j \in [n^{\ell+1}]\, \forall i \in [n^\ell] : \theta^\ell_{i,j} \in \mathbb{R}$ be the *weight* from the j-th node in layer $\ell+1$ to the

i-th node in layer ℓ, and we collectively denote them by the family of vectors

$$\begin{aligned}\theta &:= (\theta^\ell_{i,j})_{\ell\in\{L-1,\dots,\},i\in[n^\ell],j\in[n^{\ell+1}]} \\ &:= (\theta^\ell_{i\leftarrow j})_{\ell\in\{L-1,\dots,0\},i\in[n^\ell],j\in[n^{\ell+1}]},\end{aligned} \tag{14}$$

where the notation $\theta^\ell_{i\leftarrow j}$ emphasizes which node pair this weight connects;

which form the model parameters (and can by changed by training). We call a tuple

$$(L, n^L, \dots, n^0, f, \theta, \gamma)$$

a *predictive coding network (PCN)*.

This definition collects all the pieces of information that are needed to specify (but not compute) a specific PCN, e.g., the concrete example introduced in (2) is the PCN given by

$$(3, 1, 1, 1, f, (\alpha, \beta), \gamma),$$

where, according to the definition above, 3 denotes the number of layers, the 3-tuple $(1, 1, 1)$ the number of nodes in each layer, $f : \mathbb{R} \to \mathbb{R}$ the (not further specified) activation function, (α, γ) the weights, and γ the step size. A (gradient-based) computation rule that specifies how PCNs deliver an output is still missing at this stage. In order to define it in a concise manner, two functions need to be introduced on which the gradient computation depends.

Definition B.2 (Auxiliary functions)**.** Let $(L, n^L, \dots, n^0)$ be a neural architecture and f an activation function. Consider the index sets

$$\mathcal{U} := \{(\ell, i) : \ell \in \{L, \dots, 0\}, i \in [n^\ell]\}, \tag{15}$$

$$\mathcal{U}_\star := \{(\ell, i) : \ell \in \{L-1, \dots, 0\}, i \in [n^\ell]\}, \tag{16}$$

and

$$\mathcal{W} := \{(\ell, i, j) : \ell \in \{L-1, \dots, 0\}, j \in [n^{\ell+1}], i \in [n^\ell]\}, \tag{17}$$

where $u := |\mathcal{U}|$, $u_\star := |\mathcal{U}_\star|$, and $w := |\mathcal{W}|$. The first set is used to index all the nodes, the second set indexes all nodes except the ones from the first (input) layer, and the third set is used to index all the weights between pairs of nodes.

Consider the families of vectors, indexed by $\mathcal{U}$ and $\mathcal{W}$,

$$y := (y^\ell_i)_{(\ell,i)\in\mathcal{U}} \in \mathbb{R}^u,$$

and

$$\xi := (\xi^\ell_{i,j})_{(\ell,i,j)\in\mathcal{W}} := (\xi^\ell_{i\leftarrow j})_{(\ell,i,j)\in\mathcal{W}} \in \mathbb{R}^w,$$

where we use the notation $\xi_{i\leftarrow j}$, concordant with the previous definition, to emphasize that this expression denotes the weight from node j to node i in a PCN context.

On these families of vectors, we define for each $(\ell, i) \in \mathcal{U}_\star$ the functions

$$\mu^\ell_i : \mathbb{R}^u \times \mathbb{R}^w \to \mathbb{R},\ \mu^\ell_i(y, \xi) := \sum_{j=1}^{n^{\ell+1}} \xi^{\ell+1}_{i\leftarrow j} f(y^{\ell+1}_j), \tag{18}$$

called *belief functions*, and

$$\mathrm{F} : \mathbb{R}^u \times \mathbb{R}^w \to \mathbb{R},\ \mathrm{F}(y, \xi) := \sum_{\ell=L-1}^{0} \sum_{j=1}^{n^\ell} \varepsilon^\ell_i(y, \xi), \tag{19}$$

called *energy functions*, where

$$\varepsilon^\ell_i : \mathbb{R}^u \times \mathbb{R}^w \to \mathbb{R},\ \varepsilon^\ell_i(y, \xi) := \frac{1}{2}(y^\ell_i - \mu^\ell_i(y, \xi))^2, \tag{20}$$

are the *error functions*. An interpretation of all these objects is given in Remark 2.1. We denote the derivative of F with respect to the $\mathcal{U}_\star \ni (\ell, i)$-th entry of the y variables (the "node derivative") with $\partial^\ell_i \mathrm{F}$.

Notice that $y^0_1, \dots, y^0_{n^0}$, the elements of y that are indexed by the nodes from the last layer $\ell = 0$, are not used in (18), only in (19) and (20). We keep these nodes nonetheless in the definition of μ^ℓ_i for notational convenience, in order for μ^ℓ_i, F and ε^ℓ_i to have the same set of arguments. The elements of y that are indexed by the first layer, $\ell = L$, are used in all three functions.

Definition B.3 (Computation rule)**.** Let $(L, n^L, \dots, n^0, f, \theta, \gamma)$ be a PCN. For a given input $s := (s_{n^L}, \dots, s_1) \in \mathbb{R}^{n^L}$ and given *initializations* (also called *seeds*) for all non-input nodes,

$$\eta := (\eta^\ell_i)_{(\ell,i)\in\mathcal{U}_\star} \in \mathbb{R}^{u_\star},$$

the sequence of all *value nodes*, which is a sequence of vectors within $\mathbb{R}^u$,

$$\big((x^\ell_{i,t})_{(\ell,i)\in\mathcal{U}}\big)_{t\in\mathbb{N}},$$

is defined in the following way:

If $t = 0$, we set:

- $\forall (\ell, i) \in \mathcal{U} \setminus \mathcal{U}_\star :\ x^\ell_{i,0} = s_i$,
- $\forall (\ell, i) \in \mathcal{U}_\star :\ x^\ell_{i,0} = \eta^\ell_i$,

and if $t = 1, 2, \dots$, we set:

- $\forall (\ell, i) \in \mathcal{U} \setminus \mathcal{U}_\star :\ x^\ell_{i,t+1} = x^\ell_{i,t}$,
- $\forall (\ell, i) \in \mathcal{U}_\star :\ x^\ell_{i,t+1} = x^\ell_{i,t} - \gamma\, \partial^\ell_i \mathrm{F}(x_t, \theta)$.

(To exclude a pathological behavior, we assume that the node derivative of F exists at all points (x_t, θ).)

Denote by $x_\infty \coloneqq (x^\ell_{i,\infty})_{(\ell,i)\in\mathcal{U}_\star}$ the limiting values of the vector all non-input value nodes,

$$\lim_{t\to\infty} (x^\ell_{i,t})_{(\ell,i)\in\mathcal{U}_\star}, \tag{21}$$

if all these limits exists. The subvector x^0_∞ of x_∞, that corresponds to the limiting nodes from the last layer, $(x^0_{i,\infty})_{i\in[n^0]}$, is called the ($\eta$-)*output* of the PCN, as the computations depend a priori on the initializations (it will turn out, see Propositions C.1 and C.2, that if an output exists, it is unique, hence we can speak simply of the *output*). Furthermore, let

$$(x_t)_{t\in\mathbb{N}} \coloneqq \big((x^\ell_{i,t})_{(\ell,i)\in\mathcal{U}_\star}\big)_{t\in\mathbb{N}},$$

as we will use this notation a number of times in the sequel (e.g.,limit (21) from above becomes $\lim_{t\to\infty} x_t$).

In the definition of the computation rule above, the expression "$\forall(\ell,i)\in\mathcal{U}\setminus\mathcal{U}_\star$" explicitly means "$\forall i\in[n^L]$", hence the nodes at the input layer take for all time steps the constant input value.

In (Song et al., 2020), instead of a single sequence

$$\big((x^\ell_{i,t})_{(\ell,i)\in\mathcal{U}}\big)_{t\in\mathbb{N}}, \tag{22}$$

there are two more sequences that are directly defined,

$$\big((\mu^\ell_{i,t})_{(\ell,i)\in\mathcal{U}_\star}\big)_{t\in\mathbb{N}} \qquad \text{and} \qquad \big((\varepsilon^\ell_{i,t})_{(\ell,i)\in\mathcal{U}_\star}\big)_{t\in\mathbb{N}}.$$

In our setup, these can be seen to arise by feeding the functions μ^ℓ_i and ε^ℓ_i from Definition B.2, defined for $(\ell,i)\in\mathcal{U}_\star$, with the sequence (22):

$$\big((\mu^\ell_{i,t})_{(\ell,i)\in\mathcal{U}_\star}\big)_{t\in\mathbb{N}} \coloneqq \big((\mu^\ell_i(x^\kappa_{j,t}))_{(\kappa,j)\in\mathcal{U}}\big)_{t\in\mathbb{N}},$$
$$\big((\varepsilon^\ell_{i,t})_{(\ell,i)\in\mathcal{U}_\star}\big)_{t\in\mathbb{N}} \coloneqq \big((\varepsilon^\ell_i(x^\kappa_{j,t}))_{(\kappa,j)\in\mathcal{U}}\big)_{t\in\mathbb{N}},$$

which makes our setup more lightweight, in comparison.

Remark B.4 (Deterministic initializations). The initializations in (Whittington & Bogacz, 2017) are chosen to be normally distributed with constant variance and mean depending on the value from the previous layer (which in turn depends on a random choice from the layer before), i.e., $\eta^\ell_i \sim \mathcal{N}(\mu^\ell_{i,0}, 1)$, where $\mu^\ell_{i,0}$ depends on $\eta^{\ell+1}_i$, for all $(\ell,i)\in\mathcal{U}_\star$; only for $\ell = L-1$, the mean of the normal distribution is itself deterministic and depends on the input. Here, in contrast, we make no assumption about the distribution from which the initial conditions might be drawn, since the focus lies on investigating whether the initial conditions affect the convergence of the network, regardless of which initial conditions might be more or less likely.

Definition B.5 (Dynamical system)**.** Let $(L, n^L,\dots,n^0,f,\theta,\gamma)$ be a PCN with input $s\in\mathbb{R}^{n^L}$. The mapping $\Psi_{s,\theta}:\mathbb{R}^{u_\star}\to\mathbb{R}^{u_\star}$, given by

$$\Psi_{s,\theta}(z) \coloneqq (z^\ell_i - \gamma\partial^\ell_i \mathrm{F}(s,z,\theta))_{(\ell,i)\in\mathcal{U}_\star}, \tag{23}$$

that maps $z\coloneqq(z^\ell_i)_{(\ell,i)\in\mathcal{U}_\star}$ to the vector that is obtained by carrying out one time step of the computation rule for all non-input nodes, is called *PCN map*. A PCN map captures the essential, non-trivial parts of a PCN.

Remark B.6. Note that PCN maps are discrete gradient systems, by virtue of

$$G(z) \coloneqq \sum_{(\ell,i)\in\mathcal{U}_\star} \frac{(z^\ell_i)^2}{2} - \gamma\,\mathrm{F}(s,z,\theta),$$

since then $\nabla G(z) = (z^\ell_i - \gamma\partial^\ell_i \mathrm{F}(s,z,\theta))_{(\ell,i)\in\mathcal{U}_\star}$.

Remark B.7 (Design decision). One could alternatively have used a different definition of PCNs (and in particular of its computation rule and of its auxiliary functions $\mu^\ell_i, \mathrm{F}, \varepsilon^\ell_i$), such that a PCN and all its associated quantities are only defined on those nodes that change with time, i.e., on all nodes except the x^L_i's, for $i\in[n^L]$. In that case, the connection with a dynamical system would have been more direct. We settled on the definitions as above in order to stay as faithful as possible to (Song et al., 2020) and show exactly how their model, when explicitly written down, gives rise to a dynamical system.

Proposition B.8 (Prediction-stage equivalence between PCNs and dynamical systems)**.** *Let $(L,n^L,\dots,n^0,f,\theta,\gamma)$ be a PCN with input $s\in\mathbb{R}^{n^L}$, initializations $\eta\coloneqq(\eta^\ell_i)_{(\ell,i)\in\mathcal{U}_\star}\in\mathbb{R}^{u_\star}$, and value nodes sequence $(x_t)_{t\in\mathbb{N}}$ as defined in the computation rule.*

Then, we have

$$(x_t)_{t\in\mathbb{N}_{\geqslant 1}} = \big(\Psi^t_{s,\theta}(\eta)\big)_{t\in\mathbb{N}_{\geqslant 1}}, \tag{24}$$

where the "t" superscript on the right-hand side denotes the t-th composition of $\Psi_{s,\theta}$ with itself.

In particular, supposing that all value nodes converge to a finite limit, we have

$$x^0_\infty = \mathrm{proj}_{n^0}\left[\lim_{t\to\infty}\Psi^t_{s,\theta}(\eta)\right],$$

i.e., the limit of the output layer vector equals last n^0-many entries of the limit of the vector $\Psi^t_{s,\theta}(\eta)$, where $\mathrm{proj}_{n^0}[\,\cdot\,]$ denotes the projection *of a vector down to the mentioned subset of its entries.*

Proof. Writing (24) out at $t=1$, the equivalence is obvious. By induction, equality follows for all $t=2,3,\dots$ From this, the claim about the equality in the limit is immediate. □

Therefore, studying a PCN is equivalent to studying a discrete dynamical system $(\mathbb{R}^{u_\star},\Psi_{s,\theta})$ given by the iterations of the PCN map, $t\mapsto\Psi^t_{s,\theta}$. Notice: *The input to the PCN becomes a parameter of the dynamical system, and the initializations of the PCN become the input for the dynamical system.*

The reason why we exclude the input-nodes in Definition B.5 is to allow the possibility of its linearization around a fixed point, which we will later specify, to have no unital eigenvalues. If an eigenvalue has value 1, a number of standard theorems cannot be applied, as the information that the linearization contains is less indicative for the behavior of the original system.

We now compute explicitly the node derivatives $\partial_i^\ell F$, recovering the prediction part of formula (7) from (Song et al., 2020).

Proposition B.9 (Expressing the derivative explicitly)**.** *Let $(L, n^L, \dots, n^0)$ be a neural architecture and f a differentiable activation function. Then, we have*

$$\begin{cases} \partial_i^\ell \mathrm{F}(y,\xi) = \varepsilon_i^\ell(y,\xi) - f'(y_i^\ell) \sum_{j=1}^{n^{\ell-1}} \theta_{j\leftarrow i}^\ell \, \varepsilon_i^\ell(y,\xi), & \ell \neq 0, \\ \partial_i^\ell \mathrm{F}(y,\xi) = \varepsilon_i^\ell(y,\xi), & \ell = 0, \end{cases}$$

for all $y = (y_i^\ell)_{(\ell,i)\in\mathcal{U}} \in \mathbb{R}^u$, $\xi = (\xi_{i,j}^\ell)_{(\ell,i,j)\in\mathcal{W}} \in \mathbb{R}^w$, and $(\ell, i) \in \mathcal{U}_\star$.

Proof. The proof consists of trivial, but tedious computations. Notice that, if $L = 1$, we only need to prove the formula for the case $\ell = 0$, as no terms with a higher ℓ value are present. Since those computations are a subset of the ones that we need to carry out in the case $L \geqslant 2$, we first assume that this holds and consider the case $L = 1$ at the end. We thus have

$$\partial_i^\ell \mathrm{F}(y,\xi) = \begin{cases} \partial_i^\ell [\varepsilon_i^\ell(y,\xi) + \sum_{j=1}^{n^{\ell-1}} \varepsilon_j^{\ell-1}(y,\xi)], & \ell \neq 0 \\ \partial_i^0 \, \varepsilon_i^0(y,\xi), & \ell = 0, \end{cases} \tag{25}$$

since

$$\begin{aligned} \mathrm{F}(y,\xi) = &\sum_{j=1}^{n^{L-1}} \varepsilon_j^{L-1}(y,\xi) + \cdots + \sum_{j=1}^{n^{\ell+1}} \varepsilon_j^{\ell+1}(y,\xi) + \\ &+ \sum_{j=1}^{n^\ell} \varepsilon_j^\ell(y,\xi) + \sum_{j=1}^{n^\ell} \varepsilon_j^\ell(y,\xi) + \cdots \\ &\cdots + \sum_{j=1}^{n^0} \varepsilon_j^0(y,\xi), \end{aligned}$$

which further equals

$$\begin{aligned} &= \sum_{\ell=L-1}^{0} \sum_{j=1}^{n^\ell} \varepsilon_j^\ell(y,\xi) \\ &= \sum_{\ell=L-1}^{0} \sum_{j=1}^{n^\ell} \frac{1}{2}(y_j^\ell - \mu_j^\ell(y,\xi))^2 \\ &= \sum_{\ell=L-1}^{0} \sum_{j=1}^{n^\ell} \frac{1}{2}(y_j^\ell - \sum_{k=1}^{n^{\ell+1}} \xi_{j\leftarrow k}^\ell f(y_k^{\ell+1}))^2, \end{aligned}$$

and the term y_i^ℓ, for $\ell \neq 0$,

- does not appear within the sum $\sum_{j=1}^{n^{\ell+1}} \varepsilon_j^\ell(y,\xi)$ nor all the previous sums;
- appears only within the i-th summand of the sum $\sum_{j=1}^{n^\ell} \varepsilon_j^\ell(y,\xi)$;
- appears within each summand of the sum $\sum_{j=1}^{n^{\ell-1}} \varepsilon_j^\ell(y,\xi)$;
- does not appear within any summand of any later sum;

and for $\ell = 0$, only $\varepsilon_i^0(y,\xi)$ contains the term y_i^0.

We first compute $\partial_i^\ell [\varepsilon_i^\ell(y,\xi) + \sum_{j=1}^{n^{\ell-1}} \varepsilon_j^{\ell-1}(y,\xi)]$ from (25), as computing $\partial_i^0 \, \varepsilon_i^0(y,\xi)$ amount to carrying out only a subset of the same computations. We have

$$\begin{aligned} &\partial_i^\ell [\, \varepsilon_i^\ell(y,\xi) + \sum_{j=1}^{n^{\ell-1}} \varepsilon_j^{\ell-1}(y,\xi)] = \partial_i^\ell [\frac{1}{2}(y_i^\ell - \mu_i^\ell(y,\xi))^2] + \\ &\quad + \sum_{j=1}^{n^{\ell-1}} \partial_i^\ell \, \varepsilon_j^{\ell-1}(y,\xi) \\ &\quad = \partial_i^\ell [\frac{1}{2}(y_i^\ell - \sum_{j=1}^{n^{\ell+1}} \theta_{i\leftarrow j}^{\ell+1} f(y_j^{\ell+1}))^2] + \\ &\qquad + \sum_{j=1}^{n^{\ell-1}} \partial_i^\ell [\frac{1}{2}(y_j^{\ell-1} - (\sum_{k=1,\, k\neq i}^{n^\ell} \theta_{j\leftarrow k}^\ell f(y_k^\ell)) - \theta_{j\leftarrow i}^\ell f(y_i^\ell))^2] \\ &\quad = (y_i^\ell - \sum_{j=1}^{n^{\ell+1}} \theta_{i\leftarrow j}^{\ell+1} f(y_j^{\ell+1})) + \\ &\qquad + \sum_{j=1}^{n^{\ell-1}} \Big((y_j^{\ell-1} - \sum_{k=1}^{n^\ell} \theta_{j\leftarrow k}^\ell f(y_k^\ell))) \cdot \\ &\qquad \cdot \partial_i^\ell [y_j^{\ell-1} - (\sum_{k=1,k\neq i}^{n^\ell} \theta_{j\leftarrow k}^\ell f(y_k^\ell)) - \theta_{j\leftarrow i}^\ell f(y_i^\ell)] \Big). \end{aligned}$$

Now

$$\partial_i^\ell[y_j^{\ell-1} - (\sum_{k=1,k\neq i}^{n^\ell} \theta^\ell_{j\leftarrow k} f(y_k^\ell)) - \theta^\ell_{j\leftarrow i} f(y_i^\ell)] = \\ = -\theta^\ell_{j\leftarrow i} f'(y_i^\ell), \quad (26)$$

so we obtain, inserting (26) into the previous expression,

$$\partial_i^\ell[\varepsilon_i^\ell(y,\xi) + \sum_{j=1}^{n^{\ell-1}} \varepsilon_j^{\ell-1}(y,\xi)] = \\ = y_i^\ell - \sum_{j=1}^{n^{\ell+1}} \theta^{\ell+1}_{i\leftarrow j} f(y_j^{\ell+1}) - f'(y_i^\ell) \cdot \\ \cdot \sum_{j=1}^{n^{\ell-1}} (\theta^\ell_{j\leftarrow i}(y_j^{\ell-1} - \sum_{k=1}^{n^\ell} \theta^\ell_{j\leftarrow k} f(y_k^\ell))).$$

Using the $\mu_i^\ell(y,\xi)$ and $\varepsilon_i^\ell(y,\xi)$ terms, we can rewrite this as

$$\partial_i^\ell[\varepsilon_i^\ell(y,\xi) + \sum_{j=1}^{n^{\ell-1}} \varepsilon_j^{\ell-1}(y,\xi)] = \\ = y_i^\ell - \mu_i^\ell(y,\xi) - f'(y_i^\ell) \sum_{j=1}^{n^{\ell-1}} \theta^\ell_{j\leftarrow i}(y_j^{\ell-1} - \mu_j^{\ell-1}(y,\xi))$$

and as

$$\partial_i^\ell[\varepsilon_i^\ell(y,\xi) + \sum_{j=1}^{n^{\ell-1}} \varepsilon_j^{\ell-1}(y,\xi)] = \\ \varepsilon_i^\ell(y,\xi) - f'(y_i^\ell) \sum_{j=1}^{n^{\ell-1}} \theta^\ell_{j\leftarrow i}(\varepsilon_j^{\ell-1}(y,\xi)),$$

respectively. To evaluate $\partial_i^0[\varepsilon_i^0](y,\xi)$, we notice that the term

$$\text{“}f'(y_i^\ell) \sum_{j=1}^{n^{\ell-1}} \theta^\ell_{j\leftarrow i}\, \varepsilon_j^{\ell-1}(y,\xi)\text{”}$$

appears only if $\ell \geqslant 1$, so thus, going through the previous computations, vanishes if $\ell = 0$, and we arrive at

$$\partial_i^0[\varepsilon_i^0](y,\xi) = \varepsilon_i^\ell(y,\xi). \quad (27)$$

If $L = 1$, the whole proof is reduced to showing $\partial_i^0\, \varepsilon_i^0(y,\xi)$, which we already did, see (27). □

Remark B.10 (Explicit computation rule). It follows that, explicitly, the computation rule in case of $L \geqslant 2$, for $\ell \neq 0$, becomes

$$x_{i,t+1}^\ell = x_{i,t}^\ell - \gamma\Big\{x_{i,t}^\ell - \mu_i^\ell(x,\theta) - f'(x_{i,t}^\ell) \cdot \\ \cdot \sum_{j=1}^{n^{\ell-1}} \theta^\ell_{j\leftarrow i}(x_{j,t}^{\ell-1} - \mu_j^{\ell-1}(x,\theta))\Big\},$$

while for $\ell = 0$, it becomes

$$x_{i,t+1}^0 = x_{i,t}^0 - \gamma\left\{x_{i,t}^0 - \mu_i^0(x,\theta)\right\} \quad (28)$$

and for $L = 1$ consists only of (28). It depends in both cases ultimately also on the values of the nodes in layer L, i.e., the input values, since these values appear within the "x" terms that is contain in the "$\mu_i^j(x,\theta)$" term.

Furthermore, for $L \geqslant 2$, the right-hand side of mapping $\Psi_{s,\theta} : \mathbb{R}^{u_\star} \to \mathbb{R}^{u_\star}$ then becomes

$$\begin{pmatrix} z_1^{L-1} - \gamma\{\varepsilon_1^{L-1}(z,\xi) - f'(z_1^{L-1}) \sum_{j=1}^{n^{L-2}} \theta^{L-1}_{j\leftarrow 1}(\varepsilon_j^{L-2}(s,z,\xi))\} \\ \vdots \\ z_{n^1}^1 - \gamma\{\varepsilon_{n^1}^1(s,z,\xi) - f'(z_{n^1}^1) \sum_{j=1}^{n^0} \theta^1_{j\leftarrow n^1}(\varepsilon_j^0(s,z,\xi))\} \\ z_1^0 - \gamma\, \varepsilon_1^0(s,z,\xi) \\ \vdots \\ z_{n^0}^0 - \gamma\, \varepsilon_{n^0}^0(z,\xi) \end{pmatrix} \quad (29)$$

while for $L = 1$ it becomes the subvector

$$\begin{pmatrix} z_1^0 - \gamma\, \varepsilon_1^0(s,z,\xi) \\ \vdots \\ z_{n^0}^0 - \gamma\, \varepsilon_{n^0}^0(s,z,\xi) \end{pmatrix}, \quad (30)$$

where again the right-hand side depends in each case the input values and $z = (z_i^\ell)_{(\ell,i)\in\mathcal{U}_\star} \in \mathbb{R}^{u_\star}$.

C. General Phase Space Analysis and Convergence for Prediction

We now prove basic theorems about the phase portrait to which PCNs give rise.

Proposition C.1 (Fixed point formula)**.** *Let $(L, n^L, \dots, n^0, f, \theta, \gamma)$ be a PCN with input $s := (s_1, \dots, s_{n^L}) \in \mathbb{R}^{n^L}$ and arbitrary initializations. The associated PCN map $\Psi_{s,\theta} : \mathbb{R}^{u_\star} \to \mathbb{R}^{u_\star}$ then has a unique fixed point*

$$\check{z} := (\check{z}_1^{L-1}, \dots, \check{z}_{n^0}^0) \in \mathbb{R}^{u_\star}$$

given by

$$\check{z}_1^{L-1} = \sum_{j_L=1}^{n^L} \theta^L_{1\leftarrow j_L} f(s_{j_L})$$

$$\vdots$$

$$\check{z}_{n^{L-1}}^{L-1} = \sum_{j_L=1}^{n^L} \theta^L_{n^{L-1}\leftarrow j_L} f(s_{j_L})$$

$$\check{z}_1^{L-2} = \sum_{j_{L-1}=1}^{n^{L-1}} \theta^{L-1}_{1\leftarrow j_{L-1}} f(\sum_{j_L=1}^{n^L} \theta^L_{j_{L-1}\leftarrow j_L} f(s_{j_L}))$$

$$\vdots$$

$$\vdots$$

$$\check{z}^1_{n^1} = \sum_{j_2=1}^{n^2} \theta^2_{n^1 \leftarrow j_2} f\Big(\cdots f(\sum_{j_L=1}^{n^L} \theta^L_{j_{L-1} \leftarrow j_L} f(s_{j_L})\Big)\cdots\Big)$$

$$\check{z}^0_1 = \sum_{j_1=1}^{n^1} \theta^1_{1 \leftarrow j_1} f\Bigg(\sum_{j_2=1}^{n^2} \theta^2_{j_1 \leftarrow j_2} f\Big(\cdots f(A)\cdots\Big)\Bigg)$$

$$\vdots$$

$$\check{z}^0_{n^0} = \sum_{j_1=1}^{n^1} \theta^1_{n^0 \leftarrow j_1} f\Bigg(\sum_{j_2=1}^{n^2} \theta^2_{j_1 \leftarrow j_2} f\Big(\cdots f(A)\cdots\Big)\Bigg).$$

where A denotes the term

$$A \coloneqq \sum_{j_L=1}^{n^L} \theta^L_{j_{L-1} \leftarrow j_L} f(s_{j_L}).$$

Proof. Because hierarchical dependencies of the entries of the vector $z \in \mathbb{R}^{u_\star}$, see Remark B.10, the proof consists of solving the equations given by the fixed point, starting from the equations corresponding to the layer $\ell = 0$ first and continuing up to layer $\ell = L-1$; then and substituting the obtained values back all the way down to layer $\ell = 0$.

Explicitly this is done in the following way: Let $\Psi_{s,\theta}(\check{z}) = \check{z}$, for $\check{z} = (\check{z}^{L-1}_1, \dots, \check{z}^{L-1}_{n^{L-1}}, \dots, \check{z}^0_1, \dots, \check{z}^0_{n^0}) \in \mathbb{R}^{u_\star}$, i.e., for all $\ell \in \{L-1, \dots, 1\}$ and all $i \in [n^\ell]$, we have

$$\check{z}^\ell_i = \check{z}^\ell_i - \gamma\Big(\check{z}^\ell_i - \sum_{j=1}^{n^{\ell+1}} \theta^{\ell+1}_{i \leftarrow j} f(z^{\ell+1}_j) + \\ - f'(\check{z}^\ell_i) \sum_{j=1}^{n^{\ell-1}} \theta^\ell_{j \leftarrow i} (\check{z}^{\ell-1}_j - \sum_{k=1}^{n^\ell} \theta^\ell_{j \leftarrow k} f(\check{z}^\ell_k))\Big), \tag{31}$$

and for $\ell = 0$ and $i \in [n^0]$, we have

$$\check{z}^0_i = \check{z}^0_i - \gamma\Big(\check{z}^0_i - \sum_{j=1}^{n^1} \theta^1_{i \leftarrow j} f(\check{z}^1_j)\Big), \tag{32}$$

respectively. Solving all the equations (32), we obtain for all $i \in [n^0]$ that

$$\check{z}^0_i = \sum_{j_1=1}^{n^1} \theta^1_{i \leftarrow j_1} f(\check{z}^1_{j_1}),$$

where we changed the name of the index variable, in order to better reflect the layer of nodes over which we are summing.

Plugging these into the equations (31), for $\ell = 1$, we obtain for all $i \in [n^1]$, since all terms containing $f'(\check{z}^0_i)$ vanish, that

$$\check{z}^1_i = \sum_{j_2=1}^{n^2} \theta^2_{i \leftarrow j_2} f(z^2_{j_2}),$$

where analogously as before we changed the name of the index variable.

Continuing in this manner until we arrive at $\ell = L-1$, where we obtain for all $i \in [n^{L-1}]$

$$\check{z}^{L-1}_i = \sum_{j_L=1}^{n^L} \theta^L_{i \leftarrow j_L} f(s_j).$$

Thus, all $\check{z}^{L-1}_i$'s are uniquely determined by the values of the input s. Now substituting these values backwards, we obtain that all $\check{z}^{L-2}_i$'s are uniquely determined by the values of s, namely,

$$\check{z}^{L-2}_i = \sum_{j_{L-1}=1}^{n^{L-1}} \theta^{L-1}_{i \leftarrow j_{L-1}} f(z^{L-1}_{j_{L-1}}) = \\ \sum_{j_{L-1}=1}^{n^{L-1}} \theta^{L-1}_{i \leftarrow j_{L-1}} f(\sum_{j_L=1}^{n^L} \theta^L_{j_{L-1} \leftarrow j_L} f(s_{j_L})).$$

Continuing in this manner now going backwards through all layers, we obtain that all $\check{z}^1_i$'s and $\check{z}^0_i$'s are also uniquely determined by the values of s, so we obtain for $i \in [n^1]$:

$$\check{z}^1_i = \sum_{j_2=1}^{n^2} \theta^2_{i \leftarrow j_2} f\Bigg(\sum_{j_3=1}^{n^3} \theta^3_{j_2 \leftarrow j_3} f\Big(\cdots f(A)\Big)\cdots\Bigg),$$

and, for $i \in [n^0]$:

$$\check{z}^0_i = \sum_{j_1=1}^{n^1} \theta^1_{i \leftarrow j_1} f\Bigg(\sum_{j_2=1}^{n^2} \theta^2_{j_1 \leftarrow j_2} f\Big(\cdots f(A)\Big)\cdots\Bigg),$$

where A denotes the term

$$A \coloneqq \sum_{j_L=1}^{n^L} \theta^L_{j_{L-1} \leftarrow j_L} f(s_{j_L}).$$ □

A standard textbook argument shows that x_∞, if it exists (see Definition B.3), equals $\check{z}$. We repeat the argument in order to cater to a general audience that may not be familiar with dynamical systems:

Proposition C.2. *Let $(L, n^L, \dots, n^0, f, \theta, \gamma)$ be a PCN with input $s \coloneqq (s_1, \dots, s_{n^L}) \in \mathbb{R}^{n^L}$ and arbitrary initializations and assume the limiting values of all non-input value nodes exist, i.e. x_∞ exists (see Definition B.3). Then, $x_\infty = \check{z}$, where $\check{z}$ is the unique fixed point $\check{z}$ of the associated PCN map $\Psi_{s,\theta}$ (as given by Proposition C.1).*

Proof. If $(x_t)_{t\in\mathbb{N}}$ has limit x_∞, then $(\Psi_{s,\theta}(x_t))_{t\geqslant 0}$ has limit $\Psi_{s,\theta}(x_\infty)$ by continuity of $\Psi_{s,\theta}$; but $(x_t)_{t\geqslant 0}$ and $(\Psi_{s,\theta}(x_t))_{t\geqslant 0}$ differ by a single shift, so must have the same limit. It follows that $\Psi_{s,\theta}(x_\infty) = x_\infty$, i.e. $x_\infty = \check{z}$ as the fixed point $\check{z}$ is unique. □

An analogous figure to the qualitative Figure 4 from Section 3 holds true for the general case.

Already from the definition of a PCN, it is apparent that excepting the step size, which does not exist for fully-connected feedforward networks (FCFNs), all the other pieces of information of PCNs are the same as for those for FCFNs. We use the same encoding of information (omitting only the step size) for FCFNs as we did for PCNs.

Definition C.3 (FCFN definition and computation rule)**.** Under the same terminology as in Definitions B.1 and B.2, we call a tuple $(L, n^L, \dots, n^0, f, \theta)$ a *fully-connected feedforward network (FCFN)*. For a given input s, the vector within $\mathbb{R}^u$ of value nodes

$$x \coloneqq (x_i^\ell)_{(\ell,i)\in\mathcal{U}},$$

is defined in the following way:

- $\forall (\ell, i) \in \mathcal{U} \setminus \mathcal{U}_\star : \; x_i^\ell \coloneqq s_i,$
- $\forall (\ell, i) \in \mathcal{U}_\star \setminus \{(\ell, i) : \ell = 0\} :$
$$x_i^\ell \coloneqq f\Big(\sum_{j=1}^{n^{\ell+1}} \theta_{i\leftarrow j}\, x_j^{\ell+1}\Big),$$
- $\forall (\ell, i) \in \mathcal{U}_\star,\; \ell = 0 : x_i^\ell \coloneqq \sum_{j=1}^{n^{\ell+1}} \theta_{i\leftarrow j}\, x_j^{\ell+1}$

The subvector of x of nodes from the last layer, $(x_i^0)_{i\in[n^0]}$, is called *output* of the FCFN.

Remark C.4 (FCFN and PCN prediction-equivalence by design). The way in which FCFNs are set up, we do not apply the activation function f at the last layer. This is necessary to ensure that the proposition below holds. Both variants of FCFNs (with and without applying the activation function at the last layer) are met in the literature and which variant one uses does not impact the major properties of FCFNs (such as universal approximation, to name the most basic one) in a significant way.

On a computational level, the major distinction between a PCN and a FCFN is that the latter needs to converge to have an output, while the former always has an output (computed after a finite number of steps).

The next proposition shows that if a PCN converges, it converges to the fixed point found above, which at the same time is equal to the output of the FCFN with the same architecture and weights.

Proposition C.5 (Prediction-stage FCFN and PCN equivalence on the fixed point)**.** *Let $(L, n^L, \dots, n^0, f, \theta, \gamma)$ be a PCN and f be C^1, i.e., continuously differentiable, and $s \coloneqq (s_1, \dots, s_{n^L}) \in \mathbb{R}^{n^L}$ an input. Suppose the PCN has an output (for some initializations η). The output is then given by the values of the fixed point projected on to the 0-th layer (and thus is independent of the initializations) and is also identical to the output of the FCFN with corresponding parameters $(L, n^L, \dots, n^0, f, \theta)$.*

Proof. The PCN map $\Psi_{s,\theta}$ is continuous, since f is C^1. Hence, if the sequence of value nodes converges for input s, or if $\Psi_{s,\theta}$ converges by Proposition B.8, they must converge to a fixed point (continuity is essential for this). By Proposition C.1, the fixed point is unique. Therefore, the value nodes from the last layer must converge to the corresponding values of the fixed point from the last layer, $(\check{z}_i^0)_{i\in[n^0]}$. But if one tracks the computation rule for FCFNs, given input s, through each layer, one arrives at the same formulas as given above for the fixed point and thus also at $(\check{z}_i^0)_{i\in[n^0]}$. □

Now, we analyze the stability of the fixed point: If each weight is sufficiently close (in absolute value) to the origin of the phase space, and the step size is sufficiently small, then every PCN is convergent to its fixed point for any (possibly non-linear) activation function f.

The proof is a direct application of the non-bifurcation Theorem D.1, which shows that for certain fixed points, these persists for small changes of the parameters:

Remark C.6 (Alternative proof strategy). One could have also based the proof of the theorem below on a different proof that relies on bounding the eigenvalues of the Jacobian for the original system, see (Krabs, 2010), Chapter 1.3. But since bounding the eigenvalues may be tedious if we have new information about the activation function f, and because our aim is to make this article self-contained to not rely on the continuous-time formalism (which, as we argued in the introduction, is in general not too informative of the discrete-time version), we do not use this approach.

Theorem C.7 (Prediction-stage convergence criteria for PCNs)**.** *Let $(L, n^L, \dots, n^0, f, \theta, \gamma)$ be a PCN, f be C^2, and $s \in \mathbb{R}^{n^L}$ an input. Then, there exists an open set open set $V \subseteq \mathbb{R}^{u_\star}$ around $0 \in \mathbb{R}^{u_\star}$, an open set $U \subseteq \mathbb{R}^w$ around $0 \in \mathbb{R}^w$, such that if we initialize the PCN with $\eta \in V$, pick the weights to be $\theta \in U$, and let the step size be $\gamma \in (0, 1)$, then the PCN converges (to the unique fixed point that coincides with the output of the corresponding FCFN as shown previously).*

Proof. We first analyze the behavior of $\Psi_{s,\theta}$, obtained by Proposition B.8, for $\theta = 0 \in \mathbb{R}^w$, i.e., all weights are set

to zero. In that case, by (29) and (30), $\Psi_{s,\theta}$ is a linear map given by

$$\Psi_{s,0}(\eta) = (1-\gamma)\eta,$$

and the dynamical system $(\mathbb{R}^{u_\star}, \Psi_{s,0})$ has as (unique) fixed point $0 \in \mathbb{R}^{u_\star}$. Furthermore, it has a single eigenvalue $1-\gamma \in (0,1)$, hence for any initial value, all trajectories converge to the fixed point 0, which is stable.

Now, if we set $\mu \coloneqq \theta'$ for some $\theta' \in \mathbb{R}^w$, $(\mu_0, x_0) \coloneqq (0,0) \in \mathbb{R}^{u_\star} \times \mathbb{R}^w$, and $h(\mu, \eta) \coloneqq \Psi_{s,\mu}(\eta)$, h is C^1 (since the activation function is C^2, but only its derivative appears within the defining formula, cf. Proposition B.9). Thus, we can apply Theorem D.1 to obtain the existence of an open set $U \subseteq \mathbb{R}^w$ that contains the zero weight vector and an open set $V \subseteq \mathbb{R}^{u_\star}$ in the parameter space that contains the zero fixed point, such that if we pick $\theta' \in U$ and $\eta \in V$, then $\Psi_{s,\theta'}$ also has a stable fixed point, which means

$$\lim_{t\to\infty} \Psi^t_{s,\theta'}(\eta)$$

exists and is finite. Reverting back to the PCN, by Proposition B.8, we conclude the proof. □

Remark C.8 (Quantitative convergence guarantees). Replacing the implicit function theorem with a quantitative version (see, e.g., (Liverani, accessed 20 December 2021)), it is possible to obtain a concrete estimate of the radiuses r_1, r_2 of open balls $B_{r_1}(0) \subseteq V \subseteq \mathbb{R}^{u_\star}$ and $B_{r_2}(0) \subseteq U \subseteq \mathbb{R}^w$ for which the above holds.

The example that is investigated in Section 5 from the main body of the article precisely is concerned with what happens with $\Psi_{s,\theta}(\eta)$ when (η, θ) are *outside* of the set $U \times V$.

D. A Theorem from Dynamical Systems Theory

In this section, we present a theorem from dynamical systems theory, that is relevant both in the main body of the article that illustrates the theory in a readable manner by a concrete example, as well as for the corresponding general theory from the appendices (see Appendices B and C).

Theorem D.1 (A non-bifurcation theorem)**.** *Let $h : \mathbb{R}^p \times \mathbb{R}^n \to \mathbb{R}^n$ be a family of C^1 maps, depending on a parameter $\mathbb{R}^p$. Assume that there exists a $(\mu_0, x_0) \in \mathbb{R}^p \times \mathbb{R}^n$ such that x_0 that is a fixed point (i.e., $h(\mu_0, x_0) = x_0$) and that 1 is not an eigenvalue of the Jacobian of $h(\mu_0, \cdot) : \mathbb{R}^n \to \mathbb{R}^n$ at x_0.*

Then, there exists

- *an open set $V \subseteq \mathbb{R}^n$ containing x_0,*
- *an open set $U \subseteq \mathbb{R}^p$ containing μ_0,*
- *and a C^1 "fixed point providing" function $r : U \to V$ such that*
  - *$r(\mu_0) = x_0$ and $h(\mu, r(\mu)) = r(\mu)$ for all $\mu \in U$;*
  - *for any fixed $\mu \in U$, the map $h(\mu, \cdot)$ has no other fixed point in V other than $r(\mu)$;*
  - *and for each $\mu \in U$, the stability of $r(\mu)$ is the same as that of x_0.*

A basic version of the proof can be found in, e.g., (Broer & Takens, 2010), Theorem 3.5, p. 116, and is in turn a direct application of the implicit function theorem for functions with domain $\mathbb{R}^p \times \mathbb{R}^n$.