FEEDFORWARD NETWORKS: ADAPTATION, FEEDBACK, AND SYNCHRONY

MANUELA AGUIAR, ANA DIAS, AND MICHAEL FIELD

Abstract. We investigate the effect on synchrony of adding feedback loops and adaptation to a large class of feedforward networks. We obtain relatively complete results on synchrony for identical cell networks with additive input structure and feedback from the final to the initial layer of the network. These results extend previous work on synchrony in feedforward networks by Aguiar, Dias and Ferreira [2]. We also describe additive and multiplicative adaptation schemes that are synchrony preserving and briefly comment on dynamical protocols for running the feedforward network that relate to unsupervised learning in neural nets and neuroscience.

Contents

Date: September 29, 2018.

1. Introduction

In this paper, and its companion [3], we study the effect of adding feedback loops to adaptive feedforward networks. Specifically, the implications for synchrony, dynamics, and bifurcation. The theoretical results in this article will focus on synchrony and be mainly algebraic (combinatorial) in character. Although we present a few results and examples on bifurcation and dynamics from [3] in this introduction, the theoretical development is left to the companion article.

Aside from reviewing background on feedforward networks—mainly coming from neural nets and learning in neuroscience—our aim in the introduction is to describe our approach and give several motivational examples that hint at some of the rich and tractable dynamical structure present in the setting of feedforward networks with feedback. We also compare and contrast our work with prior work on synchrony (for example, [6, 29, 1, 2]) and asynchronous networks [7, 8].

1.1. **Background on feedforward networks.** Dynamicists typically regard a network of dynamical systems as modelled by a graph with vertices or nodes representing individual dynamical systems, and edges (usually directed) codifying interactions between nodes. Usually, evolution is governed by a system of ordinary differential equations (ODEs) with each variable tied to a node of the graph. Examples include the ubiquitous Kuramoto phase oscillator network, which models weak coupling between nonlinear oscillators [31, 26], and coupled cell systems as formalized by Golubitsky, Stewart *et al.* [46, 22, 21].

Feedforward networks play a well-known and important role in network theory and appear in many applications ranging from synchronization in feed-forward neuronal networks [18, 43], to the modelling of learning and computation—data processing (see below). Yet feedforward networks often do not fit smoothly into the dynamicists lexicon for networks. Feedforward networks, such as artificial neural nets (ANNs) and network models for visualization and learning in the brain, usually process input data *sequentially* and not synchronously as is the case in a dynamical network. More precisely, a feedforward network is divided into layers—the (hidden) layers of an ANN—and processing proceeds layer-by-layer rather than simultaneously across all layers as happens with networks modelled by systems of differential equations. The way in which data is processed—synchronously or sequentially—can have a major impact on both dynamics and output (see Example 1.2 below). An additional feature of many feedforward networks is that they have a *function*, represented by going from the input layer to the output layer. Optimization of network function typically requires the network to be adaptive.

Example 1.1 (Artificial Neural Nets & Supervised Learning)**.** The interest in ANNs lies in their potential for approximating or representing highly complex and essentially unknown functions. For example, a map from a large set of facial images to a set of individuals with those facial images (facial recognition) or from a large set of drawn or printed characters to the actual characters (handwriting recognition). An approximation to the required function is obtained by a process of training and adaptation. No attempt is made to derive an "analytic" form for the function. We sketch only the simplest model and refer the reader to the extensive literature for more details, greater generality and related methods [10, 24, 25, 44, 23].

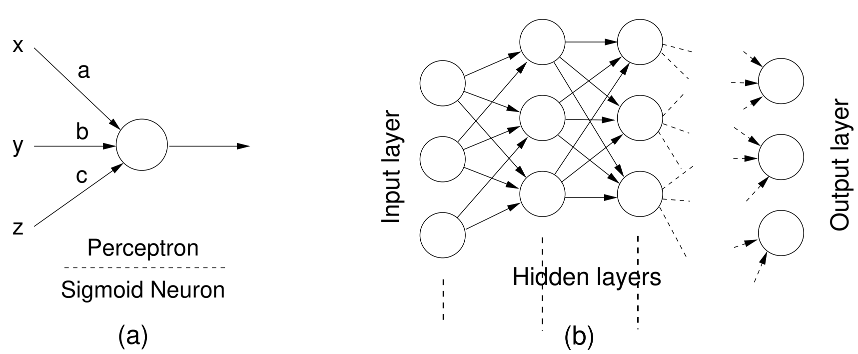

FIGURE 1. (a) Perceptron or Sigmoid neuron, (b) Model for Artificial Neural Net.

In Figure 1(a), we show the abstract building block for an ANN—the *perceptron*, first described by Rosenblatt [42] in 1958. There is no limit on the number of inputs—in the figure we show a perceptron with three inputs denoted x, y, z. Each input connection has a *weight*, denoted a, b, c in the figure. Here we assume inputs and weights are positive real numbers though negative numbers can be allowed. Assume a threshold $T > 0$. If $ax + by + cz \geq T$, the output is 1, else it is 0.

In Figure 1(b), we show an ANN built from perceptrons. Data from the input layer is successively processed by the hidden layers according to the perceptron rules to yield a data set in the output layer. In supervised learning—training of the network— data sets are repeatedly processed by the network and the output data set is compared with the true output set. For example, inputs might be facial images and the true output data set would be the actual individuals with the given facial image. Algorithms based on gradient descent and back propagation are used to adjust weights to reduce the error between the computed output data and true output data. For this to work, the model has to be smoothed so that gradient methods can work. To this end, the perceptron is replaced by a *sigmoid neuron*. The output of the sigmoid neuron in response to the input $ax + by + cz$ will be of the form $1/(1 + \exp(-\sigma(ax + by + cz))) \in (0, 1)$, where $\sigma > 0$ (for large σ the output of the sigmoid neuron closely approximates that of the perceptron). Apart from allowing the use of gradient methods, the output of the sigmoid neuron, unlike that of the perceptron, will depend continuously on the inputs. Adaptation is crucial for the performance of artificial neural networks: Minsky and Papertin showed in their 1969 book [37] that, without adaptation, ANNs based on the perceptron could not perform some basic pattern recognition operations.

From a dynamical point of view, an ANN can be regarded as a composite of maps—one for each layer—followed by a map acting on weight space. Note that the processing is layer-by-layer and not synchronous. In particular, an artificial neural net is not modelled by a discrete dynamical system—at least in the conventional sense.

The inspiration for artificial neural nets comes from neuroscience and learning. In particular, the perceptron is a model for a spiking neuron. The adaptation, although inspired by ideas from neuroscience and Hebbian learning, is global in character and does not have an obvious counterpart in neuroscience.

There does not seem to be a natural or productive way to replace the nodes in an ANN with continuous dynamical systems—at least within the framework of supervised learning. Indeed, the supervised learning model attempts to construct a good approximation to a function that acts on data sets. This is already a hard problem and it is not clear why one would want to develop the framework to handle data sets parametrised by time[1]. However, matters change when one considers unsupervised learning. In this case there are models from neuroscience involving synaptic plasticity, such as Spike-Timing Dependent Plasticity (STDP), which involve dynamics and asynchronous or sequential

[1] In dynamical systems theory there are methods based on the Takens embedding theorem [47] that allow reconstruction of complex dynamical systems from time series data. However, these techniques seem not to be useful in data processing.

computation. In the case of STDP, learning and adaptation are controlled by relative timings of spike firing (we refer to [16, 38, 11] for more details, examples and references and note that adaptive rules for a weight typically depend only on states of neurons at either end of the connection—axon). It has been observed that adaptive models using STDP are capable of pattern recognition in noisy data streams (for example, [34, 35, 36]).

As a possible application of this viewpoint, one can envisage data processing of a continuous data stream by an adaptive feedforward network comprised of layers consisting of continuous dynamical units. Here the output would reflect dynamical structure in the data stream—for example, periodicity or quantifiable chaotic behaviour. The processing could be viewed as a dynamic filter and the approach can be contrasted with reconstruction techniques based on the Takens embedding theorem.

1.2. **Dynamical feedforward networks.** We view dynamical feedforward networks as 'primitive' network objects in the sense that it is often possible to give and/or compute a simple description of the dynamics by layer. Unlike what generally happens in all-to-all or strongly connected networks, the dynamics of an individual layer in a feedforward network often has a direct and quantifiable impact on the dynamics of the network—reductive methods can often be used [7, §1.3]. For example, the dynamics of layer 1 is independent of the dynamics of subsequent layers. In the simplest cases, running layer 1 (but not subsequent layers) will result in dynamics of each node in layer 1 converging to an equilibrium or periodic solution. If we then switch on the second layer then, after a transient, the dynamics on each node will again often converge to either an equilibrium or periodic orbit. In this way, we can evolve to a final state for the network by successively switching on subsequent layers after the transient for the previously layer has decayed. We call this *sequential computation* of dynamics (as opposed to synchronous computation) and note the parallel with neural nets (see Example 1.2 below for an example). In the event that each layer evolves to an equilibrium state, we only need evolve each layer in turn until the transient has decayed. An advantage of sequential computation is that transients are not amplified through the network. This is relevant for production and transport networks (cf. the discussion in Bick & Field [7, §1.4]).

If we regard dynamical feedforward networks as primitive dynamical building blocks, then it is natural to investigate the effect of adding feedback loops to a feedforward network. The addition of a feedback loop can sometimes be viewed as an evolutionary adaptation with the potential to optimize a network function (see the discussions in [8, §6], [7, §1.6]). We assume throughout that nodes have an additive input structure—this allows for the natural addition or deletion of connections (see Section 2.1). From the structural point of view, adding a feedback loop from the final to the first layer of a feedforward network results in a new dynamical object and can sometimes lead to dramatic bifurcation in the synchrony structure of the network as well as interesting and rich dynamics as we illustrate in Examples 1.2, 1.3 below. We remark that if we add a feedback loop from the last layer to an intermediate layer A ($A > 1$), then the resulting object can be regarded as a concatenation of a feedforward network with $A - 1$ layers and a feedforward network with feedback from the final to the initial layer (see [8, §4] for the concatenation of functional asynchronous networks). Similarly comments hold if we take feedback from an intermediate layer rather than the final layer. Our focus is on feedback from the final to the initial layer of a feedforward network so as to emphasize indecomposable network objects.

1.3. **Objectives and Motivation.** In this paper we show how the addition of feedback loops to a feedforward network can enrich the synchrony structure and result in, for example, periodic synchrony patterns that do not occur for feedforward networks without feedback. The existence of a rich synchrony structure is an indicator for the potential of the network in learning applications (for example, pattern recognition). In this case, rather than internal synchronization between nodes, the objective is to synchronize *some* nodes with components of the incoming data stream[2]. As an illustration of this phenomenon, we cite the mechanism of STDP which can lead to synchronization of a small subset of nodes with a repeating component in a noisy data stream [34] or, via similar mechanisms, lead to direction location [16]. Our objective here is more modest and directed towards gaining a better understanding

[2]For effective and efficient implementation of this approach, it is expedient to introduce some *intra-layer* inhibitory structure (for example [35, 33]).

of how the synchrony structure of a feedforward network changes when feedback is added to the network. Since we are interested in adaptive networks, we also need to identify types of adaptation that preserve synchrony. It turns out that the requirement of synchrony preservation in the presence of adaptation forces a small but significant change in the definition of synchrony that we use.

There is an extensive literature on algorithms for determining synchrony in networks. For example, the combinatorial algorithms developed by Belykh & Hasler for finding synchronous clusters [6] and Kamei & Cock [29] for the determination of all synchrony in a coupled cell network. The more recent work of Aguiar and Dias [1] gives a characterization of all the synchrony patterns for general coupled cell networks in terms of the eigenvalue and eigenvector structures of the network adjacency matrix or matrices and, based on that, develops an algorithm to find those synchrony patterns. In the context of feedforward networks, Aguiar *et al.* determine the structure of synchrony in general feedforward networks [2]. Although the general methods of [6, 29, 1] can be applied to feedforward networks, such an approach takes no advantage of the feedforward structure of the networks. Our emphasis in this work is on structure and we describe structural properties of the synchrony patterns that are specific to the class of feedforward networks with feedback structures. In particular, our methods differ from those developed in [6, 29, 1].

Although the theoretical focus of this article is on synchrony, questions of dynamics and bifurcation addressed in [3] very much motivate the paper. For this reason, we give some examples of numerical simulations that illustrate both synchrony and the rich dynamics that can occur in feedforward networks with feedback. Before doing this, we briefly mention two issues related to the effects on dynamics and bifurcation when we add feedback loops.

First, it is well-known that the addition of feedback loops may have dramatic and unintended effects on the dynamics and functionality of a feedforward network—for example, the *Bullwhip effect* in stock-inventory control systems [32], and delays in transport networks containing loops overlapping other routes (the *Circle* line in the London underground system is a classic example [40]). There is also the issue of Hopf bifurcation that can occur with the addition of feedback loops. We address this in [3] and give quantitative results—the bifurcation parameter will quantify the strength of the feedback loops and the Hopf bifurcation will preserve synchrony (none of these bifurcations appear to be related to the *amplified* Hopf bifurcation [41, 19] that can occur in feedforward networks).

1.4. **Examples: Dynamics, Synchrony and Bifurcation.** We use a theta-neuron model for simulations and we briefly recall definitions from Ermentrout & Kopell [14].

The theta-neuron model. The dynamics of a *theta neuron* is given by

$$\theta' = (1 - \cos(2\pi\theta)) + \eta(1 + \cos(2\pi\theta)),\ \theta \in \mathbb{R}/\mathbb{Z}.$$

If $\eta > 0$ (excitable case), dynamics is periodic; if $\eta < 0$, there are two equilibria, one of which is attracting.

Following Chandra *et al.* [12], we start by considering a network $\mathcal{N}$ of 50 theta neurons with dynamics of node i, $1 \leq i \leq 50$, given by

$$\theta_i' = (1 - \cos(2\pi\theta_i)) + (1 + \cos(2\pi\theta_i))(\eta_i + I_i), \tag{1.1}$$

$$I_i = s_i \sum_{j \in \mathbf{50}} w_{ij} P(\theta_j), \tag{1.2}$$

where $P(\theta) = \frac{2^6(6!)^2}{12!}(1 - \cos(2\pi\theta))^6$ is a 'bump function', $w_{ij} \in \mathbb{R}$ are weights, and s_i is a scaling constant defined below. Suppose that $\mathcal{N}$ is a feedforward network with 4 layers consisting of 10, 15, 10 and 15 nodes respectively. We assume there is all to all coupling from layer j to layer $j+1$, for $j = 1, 2, 3$, and no-self loops ($w_{ii} = 0$, for all nodes i). The scaling constants s_i depend only on the layer and, apart from layer 1, are the reciprocals of the in-degrees of nodes in the layer: $s_1 = 1$, $s_2 = s_4 = 1/10$, $s_3 = 1/15$. The constants η_i will all be chosen equal to -0.1 (non excitable case).

We initialize the network in the following way. Initial states are chosen randomly and uniformly on the circle; initial weights are chosen randomly and uniformly in $[0.3, 0.8]$ (the initialization data for the two examples is available on the authors' webpages). Weights are assumed positive and constrained to lie in $[0, 2]$. Adaptation is multiplicative (see Section 3) and, for $w_{ij} \in [0, 2)$, weight dynamics is given by

$$w_{ij}' = w_{ij}(0.3 - 0.75\rho(\theta_i, \theta_j)), \tag{1.3}$$

where $\rho(\theta, \phi) = \min\{|\theta - \phi|, 1 - |\theta - \phi|\}$ is arc length on the circle $\mathbb{R}/\mathbb{Z}$. Taking account of the constraint, $w_{ij} = \max\{0, \min\{2, w_{ij}\}\}$.

Example 1.2 (A feedforward network of theta neurons)**.** When the network $\mathcal{N}$ is run, with or without adaptation, dynamics converges to a fully synchronous equilibrium (individual nodes have the approximate state 0.9025... and, with adaptation, non-zero weights all saturate at 2). This is not surprising since the individual neurons are not excitable and the basin of attraction of the fully synchronized state is open and dense in the phase space. All this may easily be established by direct computation (cf. Section 1.2).

Matters become more interesting with the addition of feedback loops. We add loops from the first node of layer 4 (node 36 of the network) back to all nodes in layer 1. This leads to new equations

$$\theta_i' = (1 - \cos(2\pi\theta_i)) - (0.1 + 5.7P(\theta_{36}))(1 + \cos(2\pi\theta_i)),\ i \in \mathbf{10}, \tag{1.4}$$

for nodes in layer 1 (we continue to assume the constants $\eta_i = -0.1$). The weight -5.7 is fixed in what follows and is not subject to adaptation.

In Figure 2 we show numerical simulations of the dynamics of the feedforward system with feedback, governed by (1.1—1.4). The simulation shows 2.754 seconds of time evolution and uses 4th order Runge-Kutta with time step $\Delta t = 0.00085$ (the adaptation uses Euler with the same time step). All computations shown here were done in long double precision—about 18 decimal places of accuracy. For this example, the same initialization of weights and states is used throughout (different initializations may and do result in different dynamics—notably

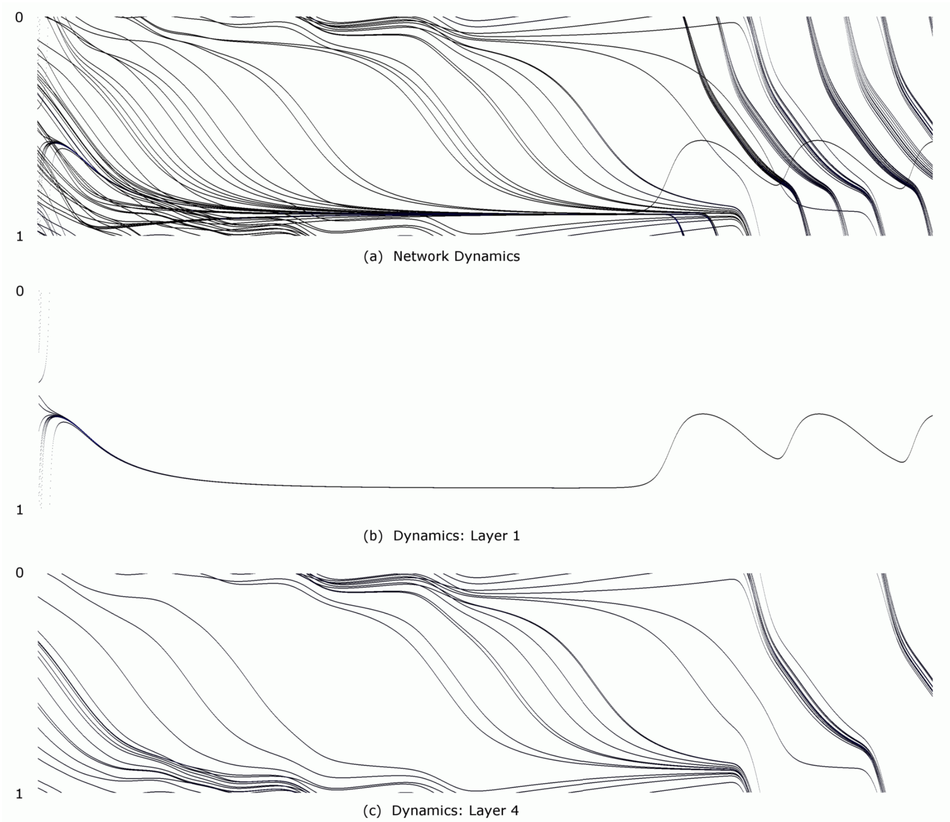

FIGURE 2. Dynamics in an adaptive 50 node feedforward network with 4 layers and feedback. (a) Network dynamics: showing the dynamics of all nodes. (b) Dynamics of nodes in layer 1. (c) Dynamics of nodes in layer 4. For each panel, we show 2.754 seconds of time evolution using fourth order Runge-Kutta with $\Delta t = 0.00085$.

convergence to a fully synchronous equilibrium, as in the case without feedback).

In Figure 2(a), we show dynamics for the complete network (with adaptation governed by (1.3)). Figure 2(b) shows the appearance of a threshold oscillation in layer 1 after about 1.7 seconds of simulation.

By contrast, Figure 2(c) suggests that the dynamics on layer 4 is converging to a phase oscillation (similar behaviour is seen in layers 2 and 3 and is not shown).

In Figure 3 we show the asymptotic behaviour of the dynamics of the network shown in Figure 2: 1.0175 seconds of simulation are displayed after 42.735 seconds of evolution of the network. All nodes are synchronized within layers. Layers 2, 3 and 4 exhibit synchronized phase oscillations while the nodes of layer 1 display synchronized threshold oscillations. The oscillations are all frequency synchronized. However, they are not phase synchronized—although the phase relationships are periodic between layers. The oscillatory behaviour is not the result of a Hopf bifurcation from a fully synchronous equilibrium state. Indeed, the mechanisms leading to the appearance of the threshold oscillation—which is very robust—appear subtle.

Choosing a different initialization of weights and states can lead to convergence to a fully synchronous equilibrium. In every case, if weights are all initialized to be non-zero, weights eventually all saturate to the maximum allowed value of 2.0. It turns out that the fully synchronous equilibrium and the synchronized periodic state shown in Figure 3 are both asymptotically stable attractors. Moreover, the synchronized oscillatory state has a large basin of attraction. In particular, it is hard to perturb the oscillatory solution so that it converges to the fully synchronous equilibrium. This is so even though most initializations of the original network (weights initialized uniformly in $[0.3, 0.8]$) appear to evolve to the fully synchronous equilibrium. It is natural to think of the threshold oscillation as pathological and reminiscent of the bullwhip effect [32].

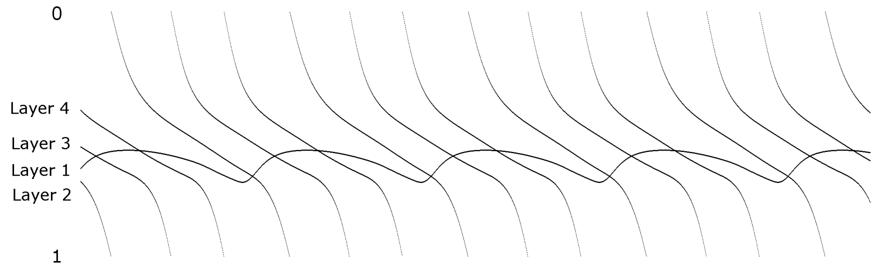

FIGURE 3. Periodicity and exact synchronization within layers for adaptive dynamics shown in Figure 2(a): 1.0175 seconds of time evolution shown after 42.735 seconds of time evolution. Fourth order Runge-Kutta with $\Delta t = 0.0005$.

We may evolve the network using sequential computation: initially evolve layer 1, then switch on layer 2 and evolve both layers 1 and 2, and so on until all 4 layers are evolving (synchronous computation). We show the result for layer 1 in Figure 4 over a period of 17.09 seconds of simulation. In this case, layer 2 is switched on after 4.27 seconds, layer 3 after 8.54 seconds and all layers after 12.82 seconds. Until layer 4 switched on, nodes in layer 1 converge to a synchronized equilibrium at $\theta = 0.564....$ After layer 4 is switched on, the nodes in layer 1 start to oscillate synchronously (as shown in Figure 2(b)) but the oscillation decays after a few seconds and all 50 nodes converge to a fully synchronous asymptotically stable equilibrium state at $\theta = 0.9025....$ Of course, using sequential computation avoids the propagation of large transients through the network. Indeed, if we increase the time between switching on layers, then the oscillatory transient decays faster.

Increasing the time intervals by a factor of two or more leads to a single oscillation followed by decay to the fully synchronous state. Decreasing the time between switching initially lengthens the time of the oscillatory transient. However, if the time interval is small enough (for example, at most 1 second), there is no oscillatory transient and dynamics either converges to the fully synchronous equilibrium or to the threshold oscillation.

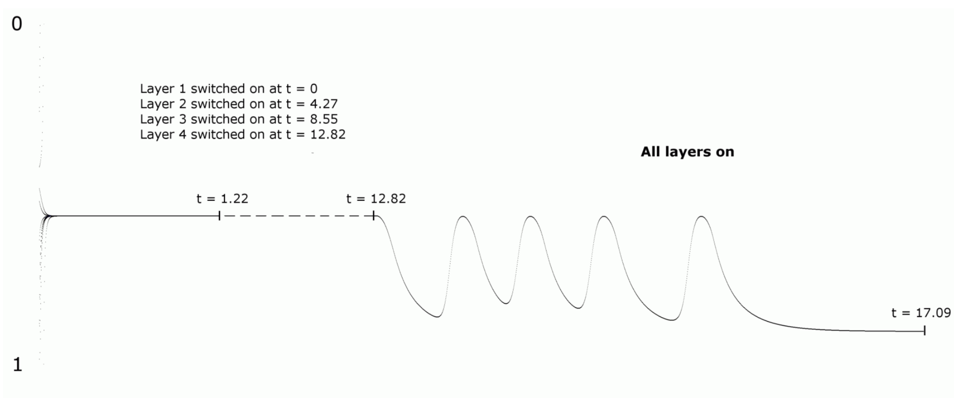

Figure 4. Dynamics of layer 1 under sequential computation. Layer 2 is switched on at $t = 4.27$, layer 3 at $t = 8.54$, and all layers at $t = 12.82$. Dynamics over time interval $[1.22, 12.82]$ is a fixed equilibrium and not shown. Fourth order Runge-Kutta with $\Delta t = 0.0021$.

In the companion paper [3], we examine this class of examples further and show that in parameter ranges where exact synchronization is not obtained, long-term dynamics can be reminiscent of Chimeras [39]. We also address the issue of long term weight dynamics—which is subtle—and the use of other adaptation schemes that do not necessarily lead to saturation of weights.

Example 1.3 (Layer periodic synchrony and stability)**.** The main focus of this paper is on the classification of synchrony in feedforward networks with feedback. We give an example that illustrates our results on synchrony as well as the stability of synchrony clusters. We continue to use the theta-neuron model described above. We assume 20 nodes and 6 layers connected as shown in Figure 5. Weights for each connection are shown in the figure. We assume no adaptation, and so dynamics is governed by the obvious variation of (1.1,1.2). For the simulations, we take $\eta_i = -0.715$ for all nodes (similar dynamics are exhibited for $\eta_i < 0$ and close enough to -0.715).

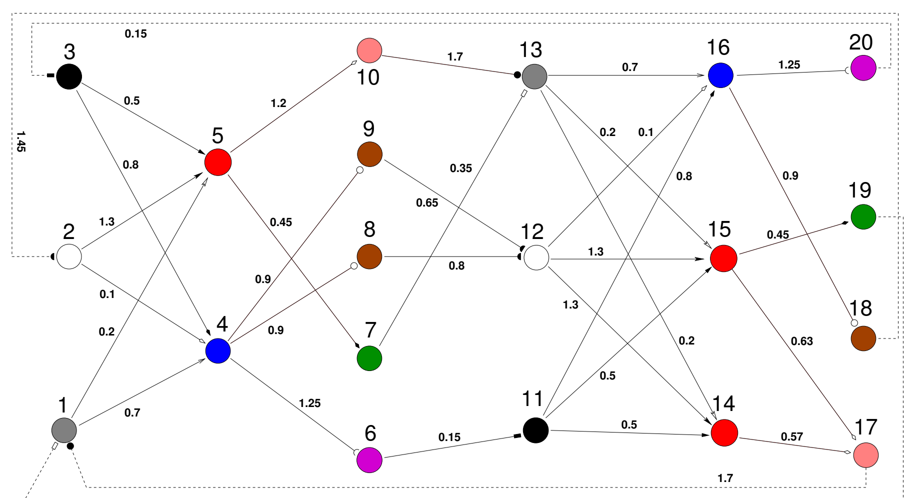

FIGURE 5. 20 node network of theta-neurons. Nodes are numbered from 1 to 20 and weights are shown for each connection—for example $w_{17,15} = 0.63$. The partition $\{\{1,13\},\{2,12\},\{3,11\},\{4,16\},\{5,14,15\},\{6,20\},\{7,19\},\{8,9,18\},\{10.17\}\}$ is a synchrony partition of **20** and is layer 3-periodic: for all $i \in \mathbf{6}$, nodes in layers i and j can be synchronous only if $j \equiv i \bmod 3$ and every node in layer i is synchronous with at least one node in layer $i+3 \pmod 6$.

If nodes are initialized to lie within the synchrony subspace defined by the synchrony partition of Figure 5, we find that dynamics is asymptotic to the periodic motion shown in Figure 6.

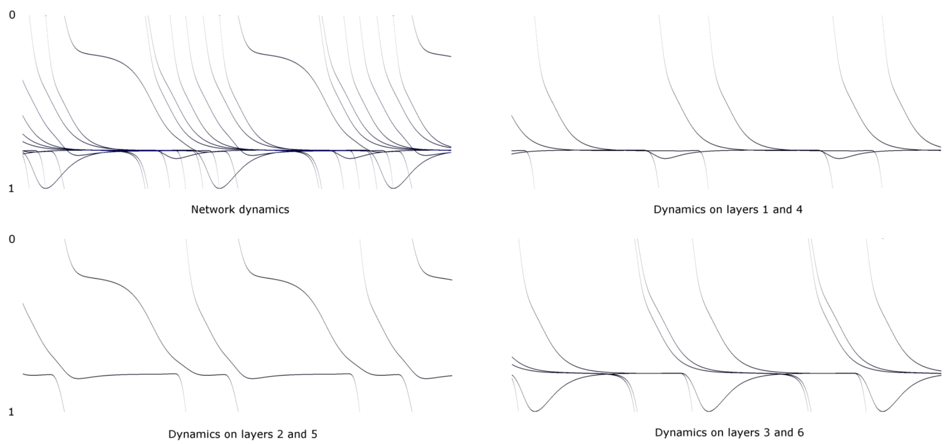

FIGURE 6. Dynamics of 20 node network showing layer periodic synchrony. Simulation of 3.256 seconds using fourth order Runge-Kutta with $\Delta t = 0.0002$.

Referring to the figure, we show network dynamics in the top left-hand panel and, working clockwise, the dynamics in Layers 1 & 4, 2 & 5, and 3 & 6. As can be seen from the figure, dynamics in each layer is periodic; the period is approximately 1 second.

Using numerical methods, it is not hard to show that the periodic solution shown in Figure 6 is not asymptotically stable. Indeed, if we either run the simulation for sufficiently long (about 25 seconds) or make small perturbations of the trajectory, we leave the synchrony subspace. Further evolution leads to the dynamics shown in Figure 7.

A casual glance at the network dynamics shown in the left hand panel of Figure 7 suggests there may have been a frequency doubling bifurcation. However, examining the dynamics of layer 2, shown in the

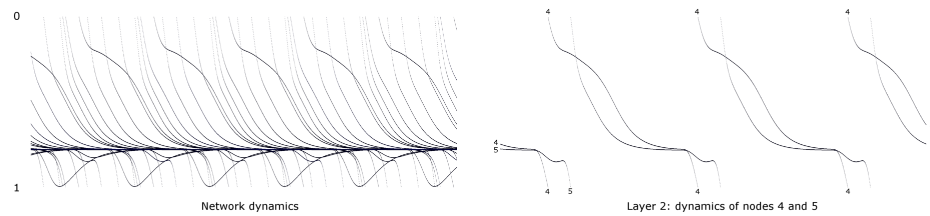

FIGURE 7. Dynamics of 20 node network showing breakdown of layer 3-periodic synchrony. Fourth order Runge-Kutta with $\Delta t = 0.0002$.

right hand panel, it is clear that the motion is still periodic and the period is close to 0.9 seconds. The true situation is revealed by Figure 8 where it is seen that dynamics in layers i and $i+3 \pmod 6$ are identical up to a half-period phase shift. This is a common phenomenon seen in both equivariant and network dynamics ([5], [20, Chapters VII, VIII], [21]). We remark that (a) synchrony within layers is preserved, and (b) the periodic solution is an asymptotically stable attractor.

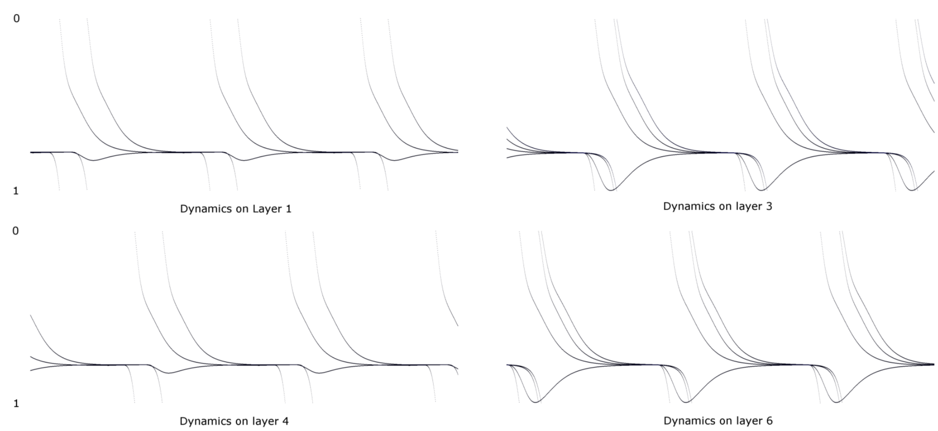

FIGURE 8. Dynamics of 20 node network showing antiphase synchrony between layers 1 and 4 & 3 and 6. Simulation of 3.256 seconds using fourth order Runge-Kutta with $\Delta t = 0.0002$.

1.5. **Main results and outline of paper.** In Section 2, we start with preliminairies on notation followed by a description of the class of dynamical networks with *additive input structure*. We then define the subclass of *weighted networks* that we study in the remainder of the paper. We conclude by reviewing the notions of synchrony and synchrony subspace. Our presentation is brief and self-contained and does not require significant background or familiarity with the theory or formalism of coupled cell systems [21]. In Section 3, we give basic definitions and results needed for defining adaptation on the class of weighted networks. We define adaptation of additive, multiplicative and mixed type and indicate the relationship to the learning rules used in neuroscience. Theorem 3.3 gives conditions for synchrony preservation in weighted networks. In Section 4 we introduce the concept of *network compatibility*. Network compatibility is a condition on the topology of a weighted network—but not the weights—that leads to a natural definition of synchrony on weighted networks and avoids degenerate synchrony. In Section 5, we review the definition of layered structure and feedforward networks [2] and define a *feedback structure* on a feedforward network—for us, this will almost always be a set of connections from the last layer to the first layer of the network though we briefly discuss other possilities in Section 5.2. In Section 6, we classify possible synchrony for a feedforward network with feedback structure from the final to first layer (Theorem 6.4). This is done under the assumption that there are no self-loops (the feedforward network is not recurrent—an FFNN in the terminology of [2]). A consequence of this result is that it is possible for synchrony to have a periodic structure across layers; this cannot happen if there is no feedback [2]. We also prove a classification result (Theorem 7.5) in Section 7 for feedforward networks which may have self-loops in the first layer (an AFFNN [2]). We add a few concluding remarks in Section 8.

2. A class of networks with additive input structure

In this section we briefly review definitions and concepts about synchrony in networks with additive input structure as well as fix the notations used throughout the paper.

2.1. **Weighted networks.** We consider a class of dynamical networks $\mathcal{N}$ consisting of k interacting dynamical systems, where $k \geq 2$. We label the individual dynamical systems, or nodes, in $\mathcal{N}$, by $\mathbf{k} \stackrel{\text{def}}{=} \{1, \ldots, k\}$. Thus $i \in \mathbf{k}$ will denote the ith node of the network. We assume that the uncoupled nodes have identical dynamics and phase space. Specifically, each node will have phase space M (a differential manifold, possibly with boundary), and there will exist a C^1 vector field f on M such that the *intrinsic dynamics* on node i is given by

$$\dot{\mathbf{x}}_i = f(\mathbf{x}_i), \quad i \in \mathbf{k}.$$

Note our convention that the state of node i is denoted by $\mathbf{x}_i$. In our examples, M will be $[0,1]$, $\mathbb{R}$, $\mathbb{T} = \mathbb{R}/\mathbb{Z}$, or $S^1 = \mathbb{R}/2\pi\mathbb{Z}$ (unit circle). This gives the simplification that we can regard the dynamics and coupling as being given by real valued functions since in these cases the tangent bundle is trivial: $TM = M \times \mathbb{R}$.

Associated to the network $\mathcal{N}$ there will be a $k \times k$ *adjacency matrix* $A = [A_{ij}]$. Each $A_{ij} \in \{0,1\}$ and the matrix A defines a unique directed graph Γ on the nodes $\mathbf{k}$ according to the rule that $j \to i$ is a connection from j to i if and only if $A_{ij} = 1$. If $i \neq j$ and $A_{ij} = 1$, i, j are *adjacent* nodes. We always assume that Γ is *connected* in the sense that we cannot write $\Gamma = \Gamma_1 \cup \Gamma_2$ where Γ_1, Γ_2 are graphs on complementary proper subsets of $\mathbf{k}$. We say that $\mathcal{N}$ has no self-loops if $A_{ii} = 0$ for all $i \in \mathbf{k}$. If $A_{ii} = 1$, then node i has a *self-loop*.

The space of $k \times k$ real matrices may be identified with $\mathbb{R}^{k^2}$—map $[m_{ij}]$ to $(m_{ij}) = (m_{11}, m_{12}, \ldots, m_{1k}, m_{21}, \ldots)$. Using this identification, the adjacency matrix A naturally defines a subspace

$$W = \{\mathbf{w} = (w_{ij}) \mid w_{ij} = 0 \text{ if } A_{ij} = 0\}$$

of $\mathbb{R}^{k^2}$ of dimension $\sum_{(i,j) \in \mathbf{k}^2} A_{ij}$. We refer to W as the *weight space* for the adjacency matrix A. Note that w_{ij} may be zero if $A_{ij} = 1$ but that w_{ij} is always zero if $A_{ij} = 0$.

Fix a C^1 *coupling function* $\phi : M^2 \to TM$ satisfying $\phi(\mathbf{x}, \mathbf{y}) \in T_{\mathbf{y}}M$ for all $\mathbf{x}, \mathbf{y} \in M$. Note that if M is a subset of $\mathbb{R}^m$ or $\mathbb{T}^m$, we may assume $\phi : M^2 \to \mathbb{R}^m$.

Under the assumption of constant weights, dynamics on $\mathcal{N}$ will be defined by the system

$$\dot{\mathbf{x}}_i = f(\mathbf{x}_i) + \sum_{j=1}^{k} w_{ij} \phi(\mathbf{x}_j, \mathbf{x}_i), \quad i \in \mathbf{k}, \tag{2.5}$$

where $\mathbf{w}$ belongs to the weight space W for A. We call networks governed by dynamics of this type *weighted networks*.

Remark 2.1. System (2.5) has an *additive input structure* [15, 7, 2]. This is an important assumption that allows for the changing of the network topology (by adding or removing connections) or for adaptation (dynamically changing weights). In either case, the underlying functional structure is not changed. The assumption is needed for the reduction of weakly coupled non-linear oscillators to the Kuramoto phase oscillator equations [31, 26]. Without the assumption, the reduced model is not usually a network of phase oscillators with diffusive coupling (for example [4]). The assumption is not always satisfied: an example is given by the phenomenon of spike amplification for pyramidal neurons in the hypocampus [17].

2.2. **Synchrony subspaces.** Let $\mathcal{P} = \{P_a\}_{a\in\mathbf{s}}$ be a partition of $\mathbf{k}$. We refer to the subsets P_a as *parts* of $\mathcal{P}$. Let p_a denote the cardinality of P_a, $a \in \mathbf{s}$. After a relabelling of nodes, we may assume that $P_1 = \{1, \ldots, p_1\}$, $P_2 = \{1 + p_1, \ldots, p_1 + p_2\}$ and so on up to $P_s = \{1 + \sum_{i=1}^{s-1} p_i, \ldots, k = \sum_{i=1}^{s} p_i\}$. We often make this assumption in proofs.

Consider a partition $\mathcal{P}$ and a cell phase space M. Take the subspace $\mathbf{\Delta}_{\mathcal{P}}$ of M^k given by

$$\mathbf{\Delta}_{\mathcal{P}} = \{(\mathbf{x}_1, \ldots, \mathbf{x}_k) \mid \mathbf{x}_i = \mathbf{x}_j \text{ if } i, j \in P_a, \text{ some } a \in \mathbf{s}\}.$$

In the coupled cell network literature [46, 22, 21], $\mathbf{\Delta}_{\mathcal{P}}$ is usually called a *polydiagonal* subspace of M^k. Polydiagonal subspaces are the natural class of subspaces to consider for the study of exact synchronization. Specifically, if $\mathbf{\Delta}_{\mathcal{P}}$ is an invariant subspace for the dynamics of (2.5), then every solution $\mathbf{X}(t) = (\mathbf{x}_1(t), \dots, \mathbf{x}_k(t))$ of (2.5) with initial condition in $\mathbf{\Delta}_{\mathcal{P}}$, will consist of s groups of synchronized trajectories: for all $a \in \mathbf{s}$, the trajectories $\mathbf{x}_i(t)$, $i \in P_a$, will be identical. After relabelling of nodes (see above), we may write $\mathbf{X} = (\mathbf{x}_1^{p_1}, \dots, \mathbf{x}_s^{p_s})$, where $\mathbf{x}^p$ is shorthand for $\mathbf{x}$ repeated p times.

If, given $\mathbf{w} \in W$, $\mathbf{\Delta}_{\mathcal{P}}$ is an invariant subspace for all choices of the cell phase spaces M, and all f, ϕ in (2.5), we call $\mathcal{P}$ a *synchrony partition* of $\mathcal{N}$ and $\mathbf{\Delta}_{\mathcal{P}}$ a *synchrony subspace* (of M^k for every M). We emphasise that we do not vary the weights (yet).

Remark 2.2. We caution that our definition of synchrony partition is provisional. It turns out for weighted networks it is natural to add a further structural condition on the network topology. This we shall do in Section 4 after we have addressed the issue of adaptation in weighted networks.

If $s = k$, we refer to $\mathcal{P}$ as the *asynchronous* partition—all parts of $\mathcal{P}$ are singletons—and denote the partition by $\mathcal{A}$. If $\mathcal{P}$ is not asynchronous, then $p_a \geq 1$ for all $a \in \mathbf{s}$, and $s < k$ (so that at least one part contains more than one element). If $s = 1$ then $\mathcal{P} = \{\mathbf{k}\}$ is the *fully synchronous* partition.

Remark 2.3. In the coupled cell literature, it is common to regard each part of a synchrony partition as being associated to a colour. With this convention, nodes are synchronized if and only if they have the same colour, that is belong to the same part. The convention in this work is that nodes lie in the same part if and only if they are *synchronous*;

nodes that are not synchronous are *asynchronous*.

We want to give a necessary and sufficient condition for a partition to be a synchrony partition.

Definition 2.4. Given a network $\mathcal{N}$ with adjacency matrix A and a fixed weight vector W, let $\mathcal{P} = \{P_a\}_{a\in\mathbf{s}}$ be a partition of the network set of cells $\mathbf{k}$. For $a, b \in \mathbf{s}$ define the *local valency function* $\nu_{a,b} : P_a \to \mathbb{R}$ and *local in-degree function* $\rho_{a,b} : P_a \to \mathbb{N}$ by

$$\nu_{a,b}(i) = \sum_{j\in P_b} w_{ij}, \quad \rho_{a,b}(i) = \sum_{j\in P_b} A_{ij},\ i \in P_a.$$

If $s = 1$ set $\nu_{1,1} = \nu : \mathbf{k}\to\mathbb{R}$, $\rho_{1,1} = \rho : \mathbf{k}\to\mathbb{Z}_0^+$ and refer to ν and ρ as the *valency* and *in-degree*.

The following proposition corresponds to Theorem 2.4 of [2] which is a generalization of Theorem 6.5 of [46] to weighted networks. We use our notation and present a different proof.

Proposition 2.5. *(Notation and assumptions as above.) Given* $\mathbf{w} \in W$*,* $\mathcal{P} = \{P_a\}_{a\in\mathbf{s}}$ *is a synchrony partition of* $\mathcal{N}$ *iff each local valency function* $\nu_{a,b}$ *is constant.*

Proof. Sufficiency. Let $t_{a,b}$ denote the constant value of $\nu_{a,b}$. Consider the network with s nodes and dynamics given by

$$\dot{\mathbf{y}}_a = f(\mathbf{y}_a) + \sum_{b\in\mathbf{s}} t_{a,b}\phi(\mathbf{y}_b, \mathbf{y}_a),\ a \in \mathbf{s}, \tag{2.6}$$

where each node has state space M (as in (2.5)). Clearly every solution of (2.6) determines a solution to (2.5) lying in $\mathbf{\Delta}_{\mathcal{P}}$ and with initial condition $(\mathbf{y}_1^{p_1}(0), \ldots, \mathbf{y}_s^{p_s}(0)) \in \mathbf{\Delta}_{\mathcal{P}}$. It follows by uniqueness of solutions that every solution $\mathbf{X}(t)$ of (2.5) with initial condition $\mathbf{X}(0) \in \mathbf{\Delta}_{\mathcal{P}}$ is of this form and so $\mathbf{X}(t) \in \mathbf{\Delta}_{\mathcal{P}}$ for all t.

Necessity. Suppose that $\nu_{\alpha,\beta}$ is not constant for some pair $(\alpha, \beta) \in \mathbf{s}^2$. Necessarily $p_\alpha > 1$. It suffices to find a specific equation of the form (2.5) for which $\mathbf{\Delta}_{\mathcal{P}}$ is not an invariant subspace. For this, take $M = \mathbb{R}$, $f \equiv 0$. Taking $x_a = a$, $a \in \mathbf{s}$, choose any smooth $\phi : \mathbb{R}^2 \to \mathbb{R}$ such that $\phi(x, y) = 1$, for (x, y) near (α, β), and $\phi(x, y) = 0$ for values of (x, y) near $(a, b) \neq (\alpha, \beta)$. Pick $i, j \in P_\alpha$ such that $\nu_{\alpha,\beta}(i) \neq \nu_{\alpha,\beta}(j)$. Suppose $\mathbf{x}_i(0) = \mathbf{x}_j(0) = \alpha$. The equations for $\mathbf{x}_i, \mathbf{x}_j$ near $t = 0$ are

$$\dot{\mathbf{x}}_i = \nu_{\alpha,\beta}(i), \quad \dot{\mathbf{x}}_j = \nu_{\alpha,\beta}(j).$$

It follows from our assumptions on ϕ and choice of i, j that $\mathbf{x}_i(t) \neq \mathbf{x}_j(t)$ for t close to zero, $t \neq 0$. Hence $\mathcal{P}$ cannot be a synchrony partition. □

Remark 2.6. In the coupled cell literature [22, 21], a synchrony partition corresponds to a balanced equivalence relation and (2.6) are the equations of the *quotient network* giving the dynamics of (2.5) on the synchrony subspace.

3. Adaptation and Weight Dynamics

We use an adaptive scheme for weight dynamics which is natural for the analysis of synchronization. We refer the reader to the remarks at the end of this section for connections with learning in neuroscience and limitations on the model.

First, assume weights and dynamics evolve according to

$$\dot{\mathbf{x}}_i = f(\mathbf{x}_i) + \sum_{j=1}^{k} w_{ij}\phi(\mathbf{x}_j, \mathbf{x}_i), \quad i \in \mathbf{k}, \tag{3.7}$$

$$\dot{w}_{ij} = \varphi(w_{ij}, \mathbf{x}_i, \mathbf{x}_j), \quad (i,j) \in \mathbf{N}, \tag{3.8}$$

where $\mathbf{N} = \{(i,j) \in \mathbf{k}^2 \mid A_{ij} = 1\}$, (3.7) satisfies the conditions for (2.5), and $\varphi : \mathbb{R} \times M^2 \to \mathbb{R}$ is C^1. This model for dynamics and adaptation assumes that the evolution of the weight w_{ij} depends only on w_{ij} and the states of the nodes i and j.

In what follows, we assume for simplicity that solutions of (3.7,3.8) are defined for all $t \geq 0$.

Definition 3.1. Let $\mathcal{N}$ be a network, $\overline{\mathbf{w}} \in W$ a weight vector and dynamics for $\mathcal{N}$ be given by equations (3.7). The system (3.7,3.8) *preserves synchrony* if for every synchrony partition $\mathcal{P}$ of $\mathcal{N}$ for the weight $\overline{\mathbf{w}}$, the subspace $\Delta_{\mathcal{P}}$ is forward invariant by the flow of (3.7) taking the initial condition $\mathbf{w}(0) = \overline{\mathbf{w}}$ for (3.8).

Of course, without further conditions, (3.7,3.8) will *not* preserve synchrony.

Definition 3.2. (1) Adaptation is *multiplicative* if there is a C^1 map $\Phi : M^2 \to \mathbb{R}$ such that

$$\varphi(w, \mathbf{x}, \mathbf{y}) = w\Phi(x, y), \ (w, (\mathbf{x}, \mathbf{y})) \in \mathbb{R} \times M^2.$$

(2) Adaptation is *additive* if there is a C^1 map $\Phi : M^2 \to \mathbb{R}$ such that

$$\varphi(w, \mathbf{x}, \mathbf{y}) = \Phi(x, y),\ (w, (\mathbf{x}, \mathbf{y})) \in \mathbb{R} \times M^2.$$

(3) Adaptation is of *mixed type* if there are distinct C^1 maps $\Phi, \Psi : M^2 \to \mathbb{R}$ and $C \neq 0$ such that

$$\varphi(w, \mathbf{x}, \mathbf{y}) = w\Phi(\mathbf{x}, \mathbf{y}) + (C - w)\Psi(\mathbf{x}, \mathbf{y}),\ (w, (\mathbf{x}, \mathbf{y})) \in \mathbb{R} \times M^2.$$

Theorem 3.3. *(Notation and assumptions as above.)*

(1) *If adaptation is multiplicative, then (3.7,3.8) preserves synchrony.*
(2) *If adaptation is additive or of mixed type, then (3.7,3.8) preserves a synchrony partition* $\{P_a\}_{a\in\mathbf{s}}$ *provided that the local in-degree functions* $\rho_{a,b}$ *are constant for all* $a, b \in \mathbf{s}$.

Proof. The proof is similar to that of Proposition 2.5. We give details for (1). Suppose that for the weight vector $\mathbf{w}^0$, $\mathcal{P}$ is a synchrony partition for (3.7). Necessarily $\nu_{a,b}$ will be constant functions for all $a, b \in \mathbf{s}$ for (3.7) (no adaptation). Fix $\mathbf{x}_0 \in \mathbf{\Delta}_{\mathcal{P}}$.

Initialize (3.7,3.8) at $\mathbf{w}^0$ and $\mathbf{x}_0 \in \mathbf{\Delta}_{\mathcal{P}}$.

Consider the ‘quotient’ system

$$\dot{\mathbf{y}}_a = f(\mathbf{y}_a) + \sum\nolimits_{b\in\mathbf{s}} V_{a,b}\phi(\mathbf{y}_b, \mathbf{y}_a), \quad a \in \mathbf{s}, \tag{3.9}$$

$$\dot{V}_{a,b} = V_{a,b}\Phi(\mathbf{y}_b, \mathbf{y}_a), \quad a, b \in \mathbf{s}, \tag{3.10}$$

$$\dot{w}_{ij} = w_{ij}\Phi(\mathbf{y}_b, \mathbf{y}_a), \quad a, b \in \mathbf{s},\ i \in P_a,\ j \in P_b, \tag{3.11}$$

where $\mathbf{y}_a \in M$, $a \in \mathbf{s}$, and $V_{a,b} : \mathbb{R} \to \mathbb{R}$, $a, b \in \mathbf{s}$. Observe that if we initialize weights with $\mathbf{w}^0$, and set $V_{a,b}(0) = \nu_{a,b} = \sum_{j \in P_b} w_{ij}^0$, where $i \in P_a$, $a, b \in \mathbf{s}$, then the solution to (3.10) is given by $V_{a,b}(t) = \sum_{j \in P_b} w_{ij}(t)$, $a, b \in \mathbf{s}$, any $i \in P_a$.

Suppose $\mathbf{x}_0 = (\tilde{\mathbf{x}}_1^{p_1}, \ldots, \tilde{\mathbf{x}}_s^{p_s}) \in \mathbf{\Delta}_{\mathcal{P}}$. Initialize (3.9,3.10,3.11) at $\mathbf{y}_0 = (\tilde{\mathbf{x}}_1, \ldots, \tilde{\mathbf{x}}_s) \in M^s$, $\mathbf{w}^0$, and $V_{a,b}(0) = \nu_{a,b}$, $a, b \in \mathbf{s}$. Then $\mathbf{x}(t) = (\mathbf{y}_1^{p_1}(t), \ldots, \mathbf{y}_s^{p_s}(t))$, $(w_{ij}(t))$ will solve

$$\dot{\mathbf{x}}_i = f(\mathbf{x}_i) + \sum_{b \in \mathbf{s}} (\sum_{j \in b} w_{ij} \phi(\mathbf{x}_j, \mathbf{x}_i)), \quad i \in \mathbf{k}, \tag{3.12}$$

$$\dot{w}_{ij} = w_{ij} \Phi(\mathbf{x}_i, \mathbf{x}_j), \quad (i, j) \in \mathbf{N}. \tag{3.13}$$

We showed above that the solution to (3.10) is given by $V_{a,b}(t) = \sum_{j \in P_b} w_{ij}(t)$, $a, b \in \mathbf{s}$, for any $i \in P_a$ and so, by Proposition 2.5, $\sum_{j \in b} w_{ij} = V_{a,b}$ is independent of $i \in P_a$ for all $a, b \in \mathbf{s}$. Hence synchrony is preserved. □

Remarks 3.4. (1) The models we have used for weight dynamics are partly motivated by models for (unsupervised) learning in neuroscience —most notably *Hebbian learning* rules [11, 10]: "neurons that fire together wire together"—and related models for synaptic plasticity such as *Spike-Timing Dependent Plasticity (STDP)* [16, 11, 38]. These models are local in the sense that the dynamics of a weight depends only on the weight and the nodes at the end of the associated connection and do not optimise or constrain a 'global' quantity such as $\sum_{ij} w_{ij}$ (as is done, for example, in the work of Ito & Kaneko [30, 27, 28]). In [3] we consider the dynamical implications of various choices of weight dynamics related to the relative timing model of STDP.

(2) In practice, it is customary to assume weights are positive and so weight dynamics will be constrained to the positive orthant $\mathbb{R}_+^{\mathfrak{a}}$. This is no problem for adaptation which is multiplicative or of mixed type (with appropriate conditions). However, for additive adaptation, hard lower and upper bounds are typically required. If weights saturate, synchrony is usually lost. If we restrict to positive weights, then there are no issues with spurious synchrony [2]—see also Remark 4.5 and the discussion in Section 4.

4. Network compatibility

In a coupled identical cell network, synchrony depends only on the network topology. In particular, synchrony is independent of the functional structure of the network: synchrony measures the minimal number of synchrony subspaces that a network with given topology must have—of course, a specific choice of network dynamics can lead to more synchrony subspaces. As we have defined it, synchrony for weighted networks depends on the network topology and the weights but not on the intrinsic dynamics or coupling function (f and ϕ in (2.5)). As was observed in Aguiar *et al.* [2], this definition is too general as it allows for degenerate or *spurious* [2, Definition 2.9] synchrony partitions.

Example 4.1. Consider the network and synchrony partition $\mathcal{P} = \{\{A, B\}, \{C, D\}, \{E, F\}\}$ shown in Figure 9. Following [2], $\mathcal{P}$ is spurious: there are no connections from nodes C, D to E and the sum of

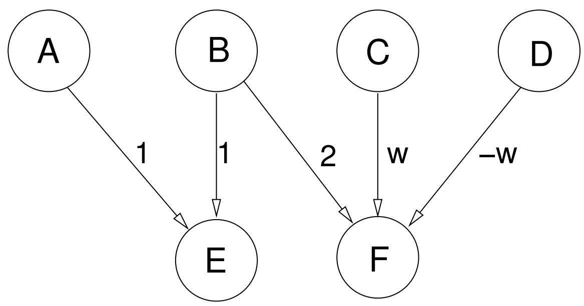

FIGURE 9. A six-node network and a spurious synchrony partition $\mathcal{P} = \{\{A, B\}, \{C, D\}, \{E, F\}\}$.

the weights of the connections from C, D to F is zero. Note that it is not possible to choose weights from cells C, D to F which do not sum to zero and preserve the synchrony partition.

In an adaptive network, it is possible for the weights to evolve so that synchrony is preserved but may change from spurious to non-spurious or vice versa. We now modify our definition of synchrony partition so as to give a natural definition of synchrony in the presence of adaptation and at the same time avoid the pathology present in example 4.1.

4.1. **Network compatibility partition.**

Definition 4.2. The partition $\mathcal{P}$ is *network compatible* if for all $a, b \in \mathbf{s}$, either $\rho_{a,b} \equiv 0$ or $\rho_{a,b}$ is non-vanishing on P_a.

Remark 4.3. Note that network compatibility depends on the network topology but not on the weights.

Example 4.4. The partition given in example 4.1 is not network compatible since $\rho_{\{E,F\},\{C,D\}}(E) = 0 \neq \rho_{\{E,F\},\{C,D\}}(F)$. On the other hand, if there is at least one connection from $\{C, D\}$ to E, then $\mathcal{P}$ is network compatible. Moreover, there will be an open dense subset of weights for which $\mathcal{P}$ will be a non-spurious synchrony partition in the sense of [2, Definition 2.9]. In Theorem 4.12 we prove that for any network compatible synchrony partition, there is a dense subset of weights for which $\mathcal{P}$ is a non-spurious synchrony partition.

Henceforth, we always assume synchrony partitions are network compatible. Moreover, if $\mathcal{P}$ is a synchrony partition in the sense defined in Section 2.2, we call the synchrony partition *spurious* if and only if it is not network compatible. Note that this is a less restrictive (weaker) notion of spurious than that given in [2]. In particular, it is *independent* of weights. However, it is the natural definition to use in the presence of adaptation and the one we adopt for the remainder of this work.

Remarks 4.5. (1) If we assume multiplicative adaptation (Definition 3.2), then synchrony and spurious synchrony are dynamically invariant. The same result holds for adaptation of mixed type or additive adaptation provided that the local in-degrees are all constant with the same value (see Theorem 3.3(2)). Since weight dynamics, with multiplicative or additive adaptation, often leads, in the limit, to zero weights, and hence zero local valencies, this is the main reason why we prefer not to impose restrictions on spurious synchrony other than to require partitions are network compatible.
(2) If $\mathcal{P} = \{P_a\}_{a\in\mathbf{s}}$ is a synchrony partition (in the sense of Definition 4.2), then two nodes can be synchronous *only if* for every $b \in \mathbf{s}$ *both* nodes receive inputs from nodes in P_b or *neither* node receives

an input from a node in P_b. This result underpins the combinatorial arguments we use for our classification of synchrony and fails if we drop the network compatibility requirement: it fails for spurious synchrony.

The asynchronous partition is *always* network compatible. In fact, it is the finest network compatible partition. The next lemma shows that there is a coarsest partition in the set of all network compatible partitions for $\mathcal{N}$. This partition gives the maximally synchronous subspace that can be defined by a network compatible partition.

Proposition 4.6. *The partition associated to the polydiagonal subspace*

$$\bigcap_{\mathcal{P}} \Delta_{\mathcal{P}},$$

where $\mathcal{P}$ runs through the network compatible partitions for $\mathcal{N}$, is network compatible for $\mathcal{N}$.

Proof. Let $\mathcal{T} = \{T_a\}_{a\in\mathbf{s}}$. Then $c, d \in T_a$ if and only if we can find a sequence $c = c_0, c_1, \ldots, c_r = d$ such that for each $i \in \mathbf{r}$, there exist a network compatible partition $\mathcal{P}$ and $P \in \mathcal{P}$, such that $c_{i-1}, c_i \in P$. Since each partition $\mathcal{P}$ is network compatible, it follows easily from this characterisation of the parts of $\mathcal{T}$, that $\mathcal{T}$ is network compatible. □

Remark 4.7. In more abstract terms, Proposition 4.6 follows from the existence of a complete lattice structure on the set of network compatible partitions for $\mathcal{N}$. See Stewart [45] for the lattice structure on synchrony partitions in coupled cell systems and Davey and Priestley [13] for background on lattices. In terms of the lattice structure, $\mathcal{T}$ is the top (maximal) element and the asynchronous partition is the bottom (minimal) element. In our context, it straightforward to define the join operation in terms of operations on partitions (what is used in the proof of Proposition 4.6 to obtain the top element) and we do not have to be concerned about the definition of the meet operation which does not generally correspond to the intersection operation on partitions.

Given a network $\mathcal{N}$ and a weight vector $\mathbf{w} \in W$, denote by $\mathrm{sync}(\mathcal{N}, \mathbf{w})$ the set of the network compatible synchrony partitions. Note that this set contains the asynchronous partition and it is in general a proper subset of the set of all the network compatible partitions.

Lemma 4.8. *Given* $\mathbf{w} \in W$*,*

$$V(\mathbf{w}) = \{\mathbf{u} \in W \mid sync(\mathcal{N}, \mathbf{u}) \supseteq sync(\mathcal{N}, \mathbf{w})\},$$

is a vector subspace of W*.*

Proof. Obvious. □

Remark 4.9. If $\mathbf{u} \in V(\mathbf{w})$, then we may have $\mathrm{sync}(\mathcal{N}, \mathbf{u}) \supsetneq \mathrm{sync}(\mathcal{N}, \mathbf{w})$. For example, if $\mathbf{u} = \mathbf{0}$. On the other hand, $\mathrm{sync}(\mathcal{N}, \mathbf{u}) = \mathrm{sync}(\mathcal{N}, \mathbf{w})$ for $\mathbf{u}$ in an open dense subset of $V(\mathbf{w})$.

Lemma 4.10. *Let* $\mathcal{P} = \{P_a\}_{a \in \mathbf{s}}$ *be a network compatible partition. For all* $a, b \in \mathbf{s}$*,* $t \in \mathbb{R}$*, there exists* $\mathbf{w} \in W$ *such that* $\nu_{a,b} \equiv t$*.*

Proof. The network compatibility condition on $\mathcal{P}$ implies that if the local in-degree $\rho_{a,b} \not\equiv 0$, then for all $t \in \mathbb{R}$, $i \in P_a$, the equality $\nu_{a,b}(i) = \sum_{j \in P_b} w_{ij} = t$ has solutions. □

Definition 4.11. Let $\mathcal{P} = \{P_a\}_{a\in\mathbf{s}} \in \text{sync}(\mathcal{N}, \mathbf{w})$. The local valencies $\nu_{a,b}$ are *non-degenerate* if $\nu_{a,b}$ is non-vanishing whenever $\rho_{a,b}$ is not identically zero.

Theorem 4.12. *Let $\varepsilon > 0$ and $\mathbf{w} \in W$ be a weight vector for $\mathcal{N}$.*

(1) *If $\mathcal{P} = \{P_a\}_{a\in\mathbf{s}} \in \text{sync}(\mathcal{N}, \mathbf{w})$ and it is not the asynchronous partition, then we can choose weight vectors $\mathbf{w}', \mathbf{w}''$ such that*
   (a) *$\|\mathbf{w} - \mathbf{w}'\| < \varepsilon$, the local valencies $\nu_{a,b}$ for $\mathbf{w}'$ are non-degenerate, and $\mathcal{P} \in \text{sync}(\mathcal{N}, \mathbf{w}')$.*
   (b) *All weights w''_{ij}, with $A_{ij} = 1$, are strictly positive and $\mathcal{P} \in \text{sync}(\mathcal{N}, \mathbf{w}'')$.*

(2) *We may choose weight vectors $\mathbf{w}'$, $\mathbf{w}''$ such that*
   (a) *$\text{sync}(\mathcal{N}, \mathbf{w}') = \text{sync}(\mathcal{N}, \mathbf{w})$, $\|\mathbf{w} - \mathbf{w}'\| < \varepsilon$, and local valencies for $\mathbf{w}'$ are non-degenerate.*
   (b) *$\text{sync}(\mathcal{N}, \mathbf{w}'') = \text{sync}(\mathcal{N}, \mathbf{w})$, and $\mathbf{w}''$ is strictly positive ($w''_{ij} > 0$, if $A_{ij} = 1$).*

Remark 4.13. Theorem 4.12 shows that for network compatible partitions $\mathcal{P}$, we can always perturb the weights so that $\mathcal{P}$ is a non-spurious synchrony partition in the sense of Aguiar *et al.* [2]—the local valencies are non-degenerate. Note that if the weight vector is strictly positive, as in (1,2)(b), then the non-identically zero local valencies are strictly positive and automatically non-degenerate.

Proof of Theorem 4.12 (1) Both statements follow easily from Lemma 4.10. We indicate a direct proof of (1b). For each $a, b \in \mathbf{s}$,

$i \in \mathbf{a}$, $j \in \mathbf{b}$, define

$$w''_{ij} = \begin{cases} 0 \ \text{ if } \rho_{a,b} \equiv 0 \\ \frac{1}{\rho_{a,b}(i)} \ \text{ otherwise.} \end{cases}$$

For this choice of $\mathbf{w}''$, the non-identically zero local valencies $\nu^{\mathcal{P}}_{a,b}$ are all constant, equal to 1.

(2) Since not being a specific synchrony partition is an open property on the set of weights, we may choose an open neighbourhood U of $\mathbf{w}$ such that for all $\mathbf{u} \in U$, $\text{sync}(\mathcal{N}, \mathbf{u}) \subseteq \text{sync}(\mathcal{N}, \mathbf{w})$ (see Lemma 4.8 and note that if $\mathbf{u} \in V(\mathbf{w}) \cap U$, then $\text{sync}(\mathcal{N}, \mathbf{u}) = \text{sync}(\mathcal{N}, \mathbf{w})$).

Let $\mathcal{T}$ be the network compatible partition given by Proposition 4.6. Applying the argument of the proof of (1b) with $\mathcal{P} = \mathcal{T}$, choose a strictly positive weight vector $\mathbf{w}^\star$ such that the local valencies $\nu_{a,b}$ are all non-degenerate. Since every network compatible partition $\mathcal{P}$ is a refinement of $\mathcal{T}$, the local valencies $\nu_{a,b}$ for $\mathbf{w}^\star$ are non-degenerate for all $\mathcal{P} \in \text{sync}(\mathcal{N}, \mathbf{w})$. Consider the weight vector $\mathbf{w}^\star_\lambda = \mathbf{w}^\star + \lambda \mathbf{w}$, $\lambda \in \mathbb{R}$. By Lemma 4.8, $\text{sync}(\mathcal{N}, \mathbf{w}^\star_\lambda) \supseteq \text{sync}(\mathcal{N}, \mathbf{w})$, for all $\lambda \in \mathbb{R}$. For sufficiently large λ, $\text{sync}(\mathcal{N}, \mathbf{w}^\star_\lambda) = \text{sync}(\mathcal{N}, \mathbf{w})$ (since $\lambda^{-1}\mathbf{w}^\star + \mathbf{w} \in U$). Consequently, $\text{sync}(\mathcal{N}, \mathbf{w}^\star_\lambda) = \text{sync}(\mathcal{N}, \mathbf{w})$, $\lambda \neq 0$. Hence we can choose $\lambda_0 \in \mathbb{R}$ so that $\text{sync}(\mathcal{N}, \mathbf{w}^\star_{\lambda_0}) = \text{sync}(\mathcal{N}, \mathbf{w})$, local valencies are non-degenerate and $\mathbf{w}^\star_{\lambda_0}$ is strictly positive. Take $\mathbf{w}'' = \mathbf{w}^\star_{\lambda_0}$ to complete the proof of (2b). For (2a), choose $\mu_0 \in [0, \varepsilon/\|\mathbf{w}^\star\|)$ so that $\mathbf{w}' = \mathbf{w} + \mu_0 \mathbf{w}^\star \in U$ and local valencies are non-degenerate. □

Example 4.14. If $\mathcal{N}$ is a network with an all-to-all coupling adjacency matrix (no self-loops), then all partitions are network compatible. Here it is easy to see that if all weights are equal, then all partitions are synchrony partitions.

Remarks 4.15. (1) For network compatible partitions, we allow zero local valency when the local in-degree is non-zero. It follows from Theorem 4.12(2) that network compatibility allows us to choose a synchrony preserving perturbation of the weight vectors making all local valencies non-degenerate.
(2) Suppose that row i of the adjacency matrix of $\mathcal{N}$ is zero and that row j is non-zero. Then for any network compatible partition the nodes i and j are asynchronous. In particular, it is not possible for all the nodes in $\mathcal{N}$ to be synchronous ($\{\mathbf{k}\}$ is not a synchrony partition).

We have the following restatement of Proposition 2.5

Proposition 4.16. *Let $\mathcal{P} = \{P_a\}_{a\in\mathbf{s}}$ be a network compatible partition and $\mathbf{w} \in W$. Then $\mathcal{P}$ is a synchrony partition of $\mathcal{N}$, with weight vector $\mathbf{w}$, iff for all $a, b \in \mathbf{s}$, there exist $t_{a,b} \in \mathbb{R}$ such that $\nu_{a,b} \equiv t_{a,b}$.*

Remark 4.17. As a simple corollary of Proposition 4.16, we have:
(1) There is an open and dense subset of W for which $\mathcal{N}$ has only the asynchronous synchrony partition.
(2) If $\mathcal{P} = \{P_a\}_{a\in\mathbf{s}}$ is a partition of $\mathbf{k}$, then the set of weight vectors $\mathbf{w} \in W$ for which $\mathcal{P}$ is a synchrony partition is a vector subspace of W of codimension $\sum_{a\in\mathbf{s}} \delta_a(p_a - 1)$ where δ_a is the cardinality of the set $\{b \in \mathbf{s} : \rho_{a,b} \not\equiv 0\}$.
(3) If $A_{ij} = 1$ for all i, j such that $i \neq j$, (all-to-all coupling), and no nodes have self loops, then all partitions are synchrony partitions

iff the non-diagonal weights w_{ij} are all equal. If there are self-loops, the same result holds provided that the diagonal weights w_{ii} are all equal. Conversely, if $\mathcal{N}$ has no self-loops and does not have all-to-all coupling, then there exists at least one partition which is not a synchrony partition.

5. Layered structure and feed forward networks

We continue to follow the notational conventions and terminology developed in section 2. Thus $\mathcal{N}$ will be a connected network consisting of k nodes, an adjacency matrix A, an associated connected network graph Γ and weight vector $\mathbf{w} \in W$. Dynamics will be given according to (2.5). Recall that we only consider network compatible partitions.

Definition 5.1 ([2, Definition 3.1])**.** The network $\mathcal{N}$ has a *layered* structure $\mathcal{L} = \{\mathcal{L}_t\}_{t\in\boldsymbol{\ell}}$ if we can partition $\mathbf{k}$ as $\mathbf{k} = \cup_{t=1}^{\ell}\mathcal{L}_t$, where

(a) $\ell > 1$.
(b) The only connections to nodes in $\mathcal{L}_1$ are self-loops.
(c) If $i \in \mathcal{L}_t$, $t > 1$, then $A_{iu} = 1$ only if $u \in \mathcal{L}_{t-1}$. In particular, no node receives a connection from a node in $\mathcal{L}_\ell$.
(d) Every node in $\cup_{t=2}^{\ell}\mathcal{L}_t$ receives at least one input.
(e) Every node in $\cup_{t=1}^{\ell-1}\mathcal{L}_t$ has at least one output.

We refer to $\mathcal{L}_t$ as the tth *layer* of $\mathcal{N}$.

Suppose that the network $\mathcal{N}$ has a layered structure with layers $\mathcal{L}_1, \dots, \mathcal{L}_\ell$. Following [2], $\mathcal{N}$ is a *Feed-Forward Neural Network*—FFNN for short—if nodes in $\mathcal{L}_1$ have no self-loops, and $\mathcal{N}$ is an *Auto-regulation*

Feed-Forward Neural Network—AFFNN for short—if at least one node in $\mathcal{L}_1$ has a self-loop.

5.1. **Notation and assumptions.** Throughout this section, $\mathcal{N}$ will denote a network with layered structure $\mathcal{L} = \{\mathcal{L}_t\}_{t\in\boldsymbol{\ell}}$.

Definition 5.2. A *transversal* (for $\mathcal{N}$) is a path $n_1 \to n_2 \to \ldots \to n_\ell$ in the network graph Γ such that $n_t \in \mathcal{L}_t$, $t \in \boldsymbol{\ell}$. The network $\mathcal{N}$ is *feedforward connected* if there is a transversal joining any node in $\mathcal{L}_1$ to any node in $\mathcal{L}_\ell$.

Remark 5.3. Since every node in $\mathcal{N} \smallsetminus \mathcal{L}_1$ has at least one input (Definition 5.1(d)), it follows (Definition 5.1(c,e)) that if $j \in \mathcal{N}$, then j lies on at least one transversal through $\mathcal{N}$.

5.2. **Feedback structures on an (A)FFNN.** Henceforth assume that $\mathcal{N}$ is a feedforward connected (A)FFNN.

Definition 5.4. Let $\mathcal{N}$ have layers $\{\mathcal{L}_t\}_{t\in\boldsymbol{\ell}}$ and J be a non-empty subset of $\{1, \ldots, \ell - 1\}$. A *J-feedback structure* $\mathcal{F}$ on $\mathcal{N}$ consists of a non-empty set of connections from nodes in $\mathcal{L}_\ell$ to nodes in each $\mathcal{L}_t$, for $t \in J$, together with a corresponding weight vector $\mathbf{u}$ lying in the associated weight space U for the feedback loops.

A description of synchrony partitions for (A)FFNNs, with examples, is given in [2, §§3,4]. In particular, it is proved that if $\mathcal{N}$ is an FFNN, synchronization can occur within but not between layers. Nevertheless, if $\mathcal{N}$ is an AFFNN, synchronization can occur between layers. The same results hold for J-feedback structures on (A)FFNNs, where $1 \notin J$, and we include them for completeness.

Proposition 5.5. *If $\mathcal{N}$ is a FFNN with a J-feedback structure where $1 \notin J$, with synchrony partition $\{P_a\}_{a\in\mathbf{s}}$, then each P_a is contained in a single layer: nodes in different layers are not synchronous.*

Proposition 5.6. *Let $\mathcal{N}$ be an AFFNN with a J-feedback structure where $1 \notin J$, with synchrony partition $\{P_a\}_{a\in\mathbf{s}}$. Along any transversal, there are the following possibilities:*

(1) *All nodes are synchronous.*
(2) *An initial segment of the transversal is synchronous, the remaining nodes are asynchronous.*
(3) *All nodes are asynchronous.*

The straightforward proofs of the above two propositions use ideas from [2, Theorem 3.4 & Lemmas 4.7, 4.8] and are omitted.

Here we focus on $\{1\}$-feedback structures and henceforth refer to a $\{1\}$-feedback structure as a *feedback structure*.

Definition 5.7. Let $\mathcal{F}$ be a feedback structure on $\mathcal{N}$.

(1) $\mathcal{F}$ is of type A if every node in $\mathcal{L}_1$ receives at least one connection from a node in $\mathcal{L}_\ell$.
(2) $\mathcal{F}$ is of type B if every node in $\mathcal{L}_\ell$ is connected to at least one node in $\mathcal{L}_1$.
(3) $\mathcal{F}$ is of type C if it is of type A and B.

If $\mathcal{F}$ is a feedback structure on $\mathcal{N}$, let $\mathcal{N}^\star$ denote the associated network. Note that $\mathcal{N}$ and $\mathcal{N}^\star$ have the same node set. If $\mathcal{F}$ is of type A, we say $\mathcal{N}^\star$ is of type A. Similarly for types B and C.

Lemma 5.8. *Let $\mathcal{F}$ be a feedback structure on $\mathcal{N}$. There exists a maximal feedforward connected subnetwork $\mathcal{N}_c$ of $\mathcal{N}$ such that*

(1) *$\mathcal{F}$ is a feedback structure of type B on $\mathcal{N}_c$.*
(2) *$i \in \mathbf{k}$ is a node of $\mathcal{N}_c$ iff there is a transversal (in $\mathcal{N}$) containing i and ending at a node in $\mathcal{L}_\ell$ connected to a node in $\mathcal{L}_1$.*
(3) *$i \to j$ is a connection for $\mathcal{N}_c$ iff $i \to j$ is a segment of a transversal (in $\mathcal{N}$) containing i, j and ending at a node in $\mathcal{L}_\ell$ connected to a node in $\mathcal{L}_1$.*
(4) *If $\mathcal{N}^\star$ is of type A, then $\mathcal{N}_c^\star$ will be of type C and the node set of $\mathcal{N}_c$ contains all nodes in $\mathcal{L}_1$.*

Proof. Define the network graph of $\mathcal{N}_c \subset \mathcal{N}$ to be the union of all transversals joining nodes in $\mathcal{L}_1$ to nodes in $\mathcal{L}_\ell$ which connect to nodes in $\mathcal{L}_1$. Obviously, $\mathcal{N}_c$ is feedforward connected, satisfies (2,3), and $\mathcal{F}$ defines a feedback structure of type B on $\mathcal{N}_c$. □

Remark 5.9. For FFNNs (as opposed to AFFNNs), we usually assume feedback structures are of type A. It follows from Lemma 5.8, that for the study of feedback induced synchrony on networks $\mathcal{N}^\star$ of type A, it is no loss of generality to assume $\mathcal{N}^\star$ of type C. Indeed, once we have synchrony for $\mathcal{N}_c^\star$, it is easy to extend to $\mathcal{N}^\star$ as the extension will not be constrained by the feedback structure.

Lemma 5.10. *(Notation and assumptions as above.) Suppose $\mathcal{F}$ is a feedback structure of type C on $\mathcal{N}$. Given $i \in \mathcal{L}_t$, $j \in \mathcal{L}_u$, there exists a path $\gamma : i = i_0 \to i_1 \to \ldots \to i_p = j$ in the network graph of $\mathcal{N}^\star$. The minimal length p of γ is $d\ell + u - t$, where $d \in \{0, 1, 2\}$, if $u \geq t$, and $d \in \{1, 2\}$ otherwise.*

Proof. A straightforward computation using feedforward connectedness and Remark 5.3. ☐

5.3. **Adaptation on feed forward networks.**

Proposition 5.11. *(Notation and assumptions as above.) Suppose that $\mathcal{N}$ is an adaptive (A)FFNN with layers $\mathcal{L}_1, \ldots, \mathcal{L}_\ell$ and that the partition $\mathcal{P}$ is a synchrony partition for an initial weight vector. Set $\mathcal{P}_t = \mathcal{L}_t \cap \mathcal{P}$, $t \in \boldsymbol{\ell}$.*

(1) *If adaptation is multiplicative, then the synchrony will be preserved within layers. That is, the induced partitions $\mathcal{P}_t$ are preserved for all $t \in \boldsymbol{\ell}$.*
(2) *If adaptation is of mixed or additive type and the local in-degrees $\rho_{a,b}$ are constant on layers, then synchrony will be preserved within layers.*

Proof. (1) follows from Theorem 3.3. (2) uses Theorem 3.3 and an easy induction on layers. ☐

6. Synchrony for FFNNs with feedback structure.

We continue with the assumptions and notation of the previous section and emphasize that $\mathcal{N}$ is always assumed feedforward connected.

Definition 6.1. Let $\mathcal{P} = \{P_a\}_{a\in\mathbf{s}}$ be a synchrony partition for $\mathcal{N}^\star$ and suppose $d \in [1, \ell - 1]$ is a divisor of ℓ and $\mathcal{P} \neq \{\mathbf{k}\}$—the fully synchronous partition.

(1) $\mathcal{P}$ is *layer d-periodic* (or layer periodic, period d) if, for all $a \in \mathbf{s}$, and $t, u \in \boldsymbol{\ell}$.

$$P_a \cap \mathcal{L}_t \neq \emptyset \Longrightarrow P_a \cap \mathcal{L}_u \neq \emptyset,\ t \equiv u,\ \bmod d.$$

(d is assumed minimal for this property.)

(2) If $\mathcal{P}$ is layer 1-periodic, $\mathcal{P}$ is *layer complete.*

Remark 6.2. If $\mathcal{P}$ is layer periodic, then each node in $\mathcal{L}_t$ will be synchronized with nodes in other layers. If $\mathcal{P}$ is layer complete, then each node in $\mathcal{L}_t$ will be synchronized with nodes in every other layer. In particular, since a layer complete partition is not the fully synchronous partition, each layer of $\mathcal{N}$ contains at least two nodes.

Examples 6.3. (1) In Figure 10 we show two examples of layer periodic synchrony partitions for feedforward connected FFNNs with feedback structure. Connections are labelled with weights and weights are arbitrary real numbers with the proviso that weights with the same symbol must have the same value.

In Figure 10(b), if we move the outputs from the top node in $\mathcal{L}_4$ labelled **A** to the other node labelled **A** in $\mathcal{L}_4$, $\mathcal{P}$ is still layer complete. However, the feedback structure is no longer of type B. As in Lemma 5.8, we can remove the node without outputs and the two nodes labelled **B** in the first row, together with associated 6 connections, to

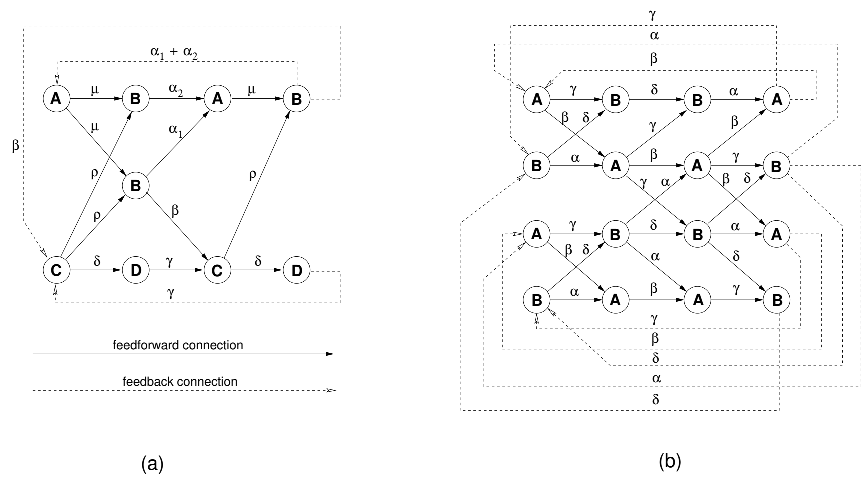

FIGURE 10. Both networks shown are feedforward connected FFNNs with feedback structure of type C. Weights are denoted $\alpha, \beta, \ldots \in \mathbb{R}$ and nodes with same letter are synchronized. (a) Layer 2-periodic network synchrony partition. (b) Layer complete network synchrony partition.

arrive at a 9-node network. The resulting feedback structure is of type C and the network is layer complete.

(2) Figure 11(a) gives an example of layer complete synchrony $\mathcal{P}$ such that there are no adjacent synchronous nodes: if the weight sums $a+e$, $b+d$ and $c+f$ are distinct, then nodes labelled A, B, C are pairwise asynchronous and so there are no edges between synchronous nodes. Figure 11(b) gives an example of layer 2-periodic synchrony such that no transversal consists of synchronous nodes.

Theorem 6.4. *(Notation and assumptions as above.) Suppose $\mathcal{N}$ has feedback structure $\mathcal{F}$ of type B. If $\mathcal{P}$ is a synchrony partition for $\mathcal{N}^\star$ such that there exists $P \in \mathcal{P}$ containing nodes from different layers, then*

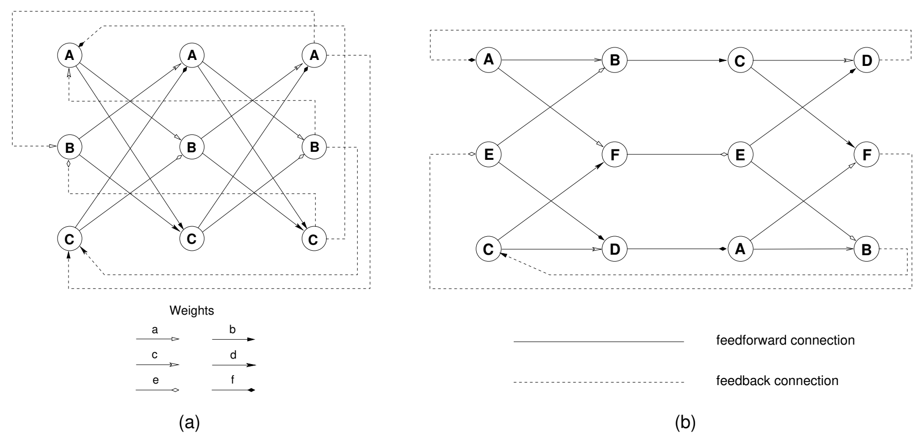

FIGURE 11. Feedforward connected FFNNs with feedback structure of type C. Synchronous nodes are labelled with same letter and connections with the same weight have the same arrowhead. (a) Layer complete synchrony containing no transversal with adjacent synchronous nodes. (b) Layer 2-periodic synchrony with all transversals consisting of asynchronous nodes.

(1) *If P contains nodes i, j from adjacent layers, $\mathcal{P}$ is layer complete or the fully synchronous partition. If, in addition, $\mathcal{N}^\star$ contains an edge $i \to j$, then P contains a transversal.*

(2) *If P only contains nodes from non-adjacent layers, then $\mathcal{P}$ is layer d-periodic, $d > 1$.*

(3) *$\mathcal{F}$ is of type A.*

Conversely, if $\mathcal{P}$ is a synchrony partition then either (a) $\mathcal{P}$ is layer periodic, or (b) only nodes in the same layer can synchronize, or (c) all nodes are synchronous, or (d) all nodes are asynchronous. In cases (a,c), $\mathcal{F}$ must be of type A; in cases (b,d) $\mathcal{F}$ may or may not be of type A.

The proof of Theorem 6.4 depends on a number of subsidiary results of interest in their own right.

Lemma 6.5. *Let $\mathcal{N}$ have feedback structure $\mathcal{F}$ and $\mathcal{P}$ be a synchrony partition for $\mathcal{N}^\star$. If there exists $P \in \mathcal{P}$ which contains nodes i, j, with $i \to j$, then there exists a transversal consisting entirely of nodes in P. We may require that the transversal ends at a node in $\mathcal{L}_\ell$ connected to a node in $\mathcal{L}_1$. (The transversal may, or may not, contain i, j.)*

Proof. Suppose $i \in \mathcal{L}_t$, $j \in \mathcal{L}_{t+1}$. Since $i \to j$ and i, j are synchronous, i must receive an input from a node $i' \in \mathcal{L}_{t-1} \cap P$ (if $t = 1$, $i' \in \mathcal{L}_\ell$). Proceeding by backwards iteration, we obtain a path

$$i_1 \to i_2 \to \ldots \to i_\ell \to \ldots \to i \to j$$

in P with $i_1 \in \mathcal{L}_1$ and $i_\ell \in \mathcal{L}_\ell$. The required transversal path is $i_1 \to i_2 \to \ldots \to i_\ell$. ☐

Remark 6.6. We often use the "backward iteration" technique of the proof of Lemma 6.5. This method may fail if there are nodes with self-loops but no other inputs. In particular, no edge in a path should be a self-loop. This will be important later when we consider AFFNNs with feedback structures.

Next a useful definition and result.

Definition 6.7. Let $\mathcal{N}$ have feedback structure $\mathcal{F}$ and $\mathcal{P} = \{P_a\}_{a \in \mathbf{s}}$ be a synchrony partition. Let γ be a path $i_0 \to i_1 \to \ldots \to i_L$ of length L in $\mathcal{N}^\star$ and suppose that $i_u \in P_{a_u}$, $u = 0, \ldots, L$. A *synchrony translate* of γ is a path $j_0 \to j_1 \to \ldots \to j_L$ such that $j_u \in P_{a_u}$, $u = 0, \ldots, L$.

Lemma 6.8. *Let $\mathcal{N}$ have feedback structure $\mathcal{F}$ and $\mathcal{P} = \{P_a\}_{a\in\mathbf{s}}$ be a synchrony partition. Let γ be a path $i_0 \to i_1 \to \ldots \to i_L$ in $\mathcal{N}^\star$ with $i_p \in P_{a_p}$, $0 \leq p \leq L$.*

1. *If $j_L \in P_{a_L}$, there is a synchrony translate $j_0 \to j_1 \to \ldots \to j_L$ with $j_p \in P_{a_p}$, $0 \leq p \leq L$.*
2. *If $a_0 = a_L$, there is a synchrony translate $j_0 \to j_1 \to \ldots \to j_L$ of γ with $j_L = i_0$. Necessarily, $j_0 \in P_{a_0}$.*

Proof. (1) follows using the standard synchrony based backward iteration argument. Statement (2) is a special case of (1). □

Proposition 6.9. *Let $\mathcal{N}$ have feedback structure $\mathcal{F}$ of type B and $\mathcal{P} = \{P_a\}_{a\in\mathbf{s}}$ be a synchrony partition for $\mathcal{N}^\star$ with $s > 1$. Suppose there exist $a \in \mathbf{s}$ and nodes $i, j \in P_a$ lying in adjacent layers. Then:*

1. *If $i \to j$ then P_a contains a transversal.*
2. *$\mathcal{F}$ is of type A.*
3. *$\mathcal{P}$ is layer complete or the fully synchronous partition.*

Proof. (1) By Lemma 6.5, P_a contains a transversal γ.
(2) $\mathcal{F}$ is of type B and feedforward connected. Suppose that $i \in \mathcal{L}_p$, $j \in \mathcal{L}_{p+1}$, where $i, j \in P_a$. Take a transversal containing i and let $i' \in \mathcal{L}_l \cap P_b$ denote the end node of the transversal. Take a synchrony translate of this transversal through node j and note that there is a node in $j' \in P_b \cap \mathcal{L}_1$ belonging to this translate. Suppose there is one node $k \in \mathcal{L}_1$ not receiving at least one connection from a node of $\mathcal{L}_l$. Take a transversal from k to i'. The synchrony partition of node k should be different from any synchrony partition of the other nodes in

that transversal. Indeed, since k has no inputs, the synchrony partition of k can only occur in the first layer. Take a synchrony translate of this transversal leading to j'. Then there is a node in $\mathcal{L}_2$ in the same synchrony partition as k, a contradiction. Thus $\mathcal{F}$ is of type A.

(3) Since $\mathcal{F}$ is of type C, it follows from Lemma 5.10 that there is a path from i to j. A synchrony translate of this path, starting at node j, ends at a node in $P_a \cap \mathcal{L}_{p+2}$. Iterating this argument, we conclude that there is at least one node from each layer in P_a. If we take any node $q \in P_d$, with $d \neq a$, then we have paths from q to any of the nodes in P_a in each of the layers. Taking synchrony translates of these paths, we conclude that P_d contains nodes from every layer and so $\mathcal{P}$ is layer complete or the fully synchronous partition. □

Lemma 6.10. *Let $\mathcal{N}$ have feedback structure $\mathcal{F}$ which is not of type A. If $\mathcal{P} = \{P_a\}_{a\in\mathbf{s}}$ is a synchrony partition, then each $P_a \in \mathcal{P}$ is contained in a unique layer $\mathcal{L}_{i(a)}$ of $\mathcal{L}$.*

Proof. Suppose the contrary. Then, for some $a \in \mathbf{s}$, there exist $i_0, j_0 \in P_a$ with $i_0 \in \mathcal{L}_t$, $j_0 \in \mathcal{L}_u$, where $t < u$. Note that if $t = 1$ there is a connection from $\mathcal{L}_\ell$ to i_0—since i_0, j_0 are synchronous and $u > 1$. Since $\mathcal{N}$ is feedforward connected, there is a path $\tau : i_p \to i_{t-1} \to \ldots \to i_0$, of length either $t-1$ or $\ell+t-1$, where $i_p \in \mathcal{L}_1$ has no connections from $\mathcal{L}_\ell$. By Lemma 6.8(1), there is a synchrony translate $j_p \to j_{p-1} \to \ldots \to j_0$ of τ. But $j_p \notin \mathcal{L}_1$ and so has inputs. Contradiction since i_p receives no inputs and so cannot be synchronous with j_p. □

Before proving the final result needed for the proof of Theorem 6.4, we need a definition.

Definition 6.11. Let $\mathcal{P} = \{P_a\}_{a\in\mathbf{s}}$ be a synchrony partition for $\mathcal{N}^\star$ and $a \in \mathbf{s}$. If P_a only contains nodes in one layer, set $\delta(a) = 0$, else define

$$\delta(a) = \min\{|i-j| \mid i,j \in P_a, \text{ where } i,j \text{ lie in different layers}\}.$$

We refer to $\delta(a)$ as the *synchronization distance* for P_a.

Proposition 6.12. *Let $\mathcal{N}$ have feedback structure $\mathcal{F}$ of type B. If $\mathcal{P} = \{P_a\}_{a\in\mathbf{s}}$ is a synchrony partition for $\mathcal{N}^\star$ and for some $a \in \mathbf{s}$, $\delta(a) = d > 0$, then*

(1) *$\mathcal{F}$ is of type A.*
(2) *$d|\ell$.*
(3) *$\mathcal{P}$ is layer d-periodic and $\delta : \mathbf{s}\to\mathbb{N}$ is constant, equal to d.*

In particular, if $d = 1$, $\mathcal{P}$ is either layer complete or the fully synchronous partition.

Proof. Property (1) holds by Lemma 6.10. Hence, by Lemma 5.10, given any two nodes in $\mathcal{N}^\star$ there is a path connecting them. Suppose $m_1, n_1 \in P_a$, where $m_1 \in \mathcal{L}_u$, $n_1 \in \mathcal{L}_{u+d}$, and $u \geq 1$ is minimal for this property. By Lemma 5.10, we may choose a path γ_1 of shortest length joining m_1 to n_1. If the length of γ_1 is L, then $L = p\ell + d$, where $0 \leq p \leq 2$. By Lemma 6.8(2), we may choose a sequence γ_j of synchrony translates of γ_1 such that γ_j will connect m_j to n_j, where $n_j = m_{j-1}$, $j > 1$. If $d \not| \ell$, then for some $j > 1$ either $m_j \in \mathcal{L}_v$, for $v < u$, or $u < v < u + d$. In the first case, we contradict the minimality of u; in the second case we contradict the definition of synchronization distance. Hence $d|\ell$, proving (2).

For (3), suppose that $a \in \mathbf{s}$ is chosen so that $\delta(a)$ is minimal. Suppose $b \in \mathbf{s}$, $b \neq a$, and choose the minimal $v \geq 1$ such that $\mathcal{L}_v \cap P_b \neq \emptyset$. Pick $x \in \mathcal{L}_v \cap P_b$ and path from x to $m_1 \in \mathcal{L}_u$. Then choose a synchrony translate of the path to connect some $y \in \mathcal{L}_{v+d} \cap P_b$ to $n_1 \in P_a$. Just as above, we show that $\mathcal{L}_{v+jd} \cap P_b \neq \emptyset$ for all $j \geq 0$. If there were nodes in other layers, this would contradict the minimality of $\delta(a)$. □

Remark 6.13. Let $\mathcal{N}^\star$ be of type B. If there is a synchrony partition $\mathcal{P}$ for which there exists $P_a \in \mathcal{P}$ with $\delta(a) \geq 2$, then it follows from Proposition 6.12 that $\ell \geq 4$ and is not prime.

Proof of Theorem 6.4. (1) follows from Propositions 6.12(3), Proposition 6.9, and Lemma 6.5. (2) follows from Proposition 6.12(3), (3) follows from Proposition 6.12(1). The converse statements follow from Lemma 6.10, Proposition 6.12 and Examples 6.3. □

7. Synchrony for AFFNNs with feedback structure

Throughout this section $\mathcal{N}$ will be an AFFNN with layer structure $\mathcal{L} = \{\mathcal{L}_i\}_{i \in \mathbf{s}}$ and $\mathcal{F}$ will be a feedback structure on $\mathcal{N}$. Let $\mathcal{N}^\star$ denote the associated network. We always assume

(1) $\mathcal{N}$ is feedforward connected.

(2) $\mathcal{N}^\star$ is of type B.

Regarding (2), note that by Lemma 5.8 there is a maximal feedforward connected subnetwork $\mathcal{N}_c$ of $\mathcal{N}$ on which $\mathcal{F}$ defines a connection structure of type B. Noting Remark 5.9, it is no loss of generality to assume $\mathcal{N}^\star$ is of type B.

Type A has the meaning previously given—every node in layer 1 receives a feedback loop.

We define $\mathcal{F}$ (or $\mathcal{N}^\star$) to be of type D if (a) there is a node in layer 1 which does not receive a feedback loop, and (b) every node in layer 1 which does not receive a feedback loop has a self-loop. If (a) is true but (b) fails, $\mathcal{F}$ is of type D*. With these conventions, an AFFNN with feedback structure will be precisely one of types A, D or D*. We emphasize that there will always be at least one feedback loop and one self loop but that for type D* networks there may be nodes with neither a feedback loop nor a self-loop.

Let $\mathcal{F}$ be a feedback structure of type D or D*. For $t \in \boldsymbol{\ell}$, define subsets D_t of $\mathcal{L}_t$ recursively by:

(1) D_1 is the subset of $\mathcal{L}_1$ consisting of nodes which receive no feedback loop.
(2) D_t is the subset of $\mathcal{L}_t$ consisting of nodes which only receive connections from nodes in D_{t-1}, $t \geq 2$.

Let $\mathcal{N}_D$ be the subnetwork of $\mathcal{N}$ with node set $N_D = \cup_{t\geq 1} D_t$ and all connections $i \to j \in \mathcal{N}$, where $i, j \in N_D$.

Lemma 7.1. *(Notation and assumptions as above.)*

(1) *There exists $p < \ell$ such that $D_j = \emptyset$, $j > p$.*

(2) *For $t > 1$, every node in D_t receives a connection from a node in D_{t-1}. Moreover, no node in $\mathcal{N}_D$ receives a connection from a node not in $\mathcal{N}_D$.*

(3) *Feedforward connected components of $\mathcal{N}_D$ are either of type D or type $D^\star$. If $\mathcal{N}_D$ is of type D, then all the feedforward components will be of type D.*

Proof. Immediate by feedforward connectedness and the definition of $\mathcal{N}_D$. □

Example 7.2. In Figure 12 we show subsets A, B, C of $\mathcal{N}_D \subset \mathcal{N}^\star$. Observe that no node in $A \cup B \cup C$ receives an input from a node outside of $A \cup B \cup C$ (or $\mathcal{N}_D$). For this example, the groups A, B

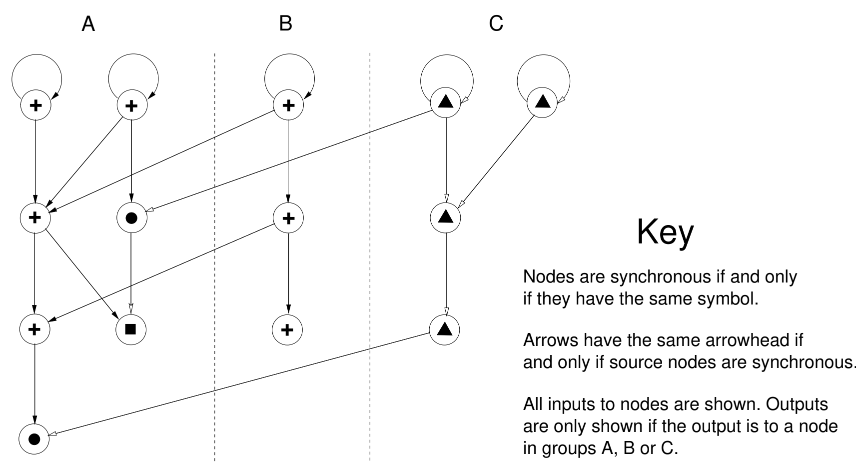

FIGURE 12. Subsets of $\mathcal{N}_D$ and synchrony across layers.

and C are feedforward connected if we drop inputs from the outside the group. Of course, $A \cup B \cup C$ is not feedforward connected as, for example, there are no connections from A to $B \cup C$. Observe that it is possible for nodes in different layers to be synchronous. As we shall see, this is essentially the only way a proper subset of nodes in different layers of a network of type D or D* can be synchronous. In particular, if no nodes in $\mathcal{N}_D$ are synchronous, there can be no nodes in different layers of $\mathcal{N}^\star$ that are synchronous. We refer to [2] for general results on synchrony for feedforward connected AFFNN networks.

We define a second subnetwork of $\mathcal{N}^\star$ which is the maximal subnetwork of type A. Let $F_1 \subset \mathcal{L}_1$ be the set of all nodes in $\mathcal{L}_1$ that receive a feedback loop. We define the subnetwork $\mathcal{N}_A$ to consist of all nodes and edges that belong to transversals from nodes in F_1. The feedback structure $\mathcal{F}$ induces a feedback structure on $\mathcal{N}_A$ with associated network $\mathcal{N}_A^\star$. Denote the node set of $\mathcal{N}_A^\star$ by N_A.

Lemma 7.3. *(Notation and assumptions as above.)*

(1) *N_A contains all the nodes in $\mathcal{L}_\ell$.*
(2) *$\mathcal{N}_A^\star$ is of type A.*
(3) *The node sets of $\mathcal{N}_D$ and $\mathcal{N}_A^\star$ (or $\mathcal{N}_A$) are disjoint and complimentary.*
(4) *If $\mathcal{P}$ is a synchrony partition for $\mathcal{N}^\star$, then $\mathcal{P}_D = \mathcal{P} \cap N_D$ will be a synchrony partition for $\mathcal{N}_D$. Conversely, every synchrony partition of $\mathcal{N}_D$ extends to a synchrony partition of $\mathcal{N}^\star$.*

Proof. For (3), observe that $\mathcal{N}_D$ receives no inputs from $\mathcal{N}_A^\star$. All the remaining statements are routine and we omit proofs. □

Remark 7.4. If $\mathcal{N}$ is not of type A, then it is possible that no node in layer 1 of $\mathcal{N}_A$ has a self-loop. In this case, possible synchrony for $\mathcal{N}_A$ is constrained by Theorem 6.4. Moreover, for this case, we shall show that nodes in $\mathcal{N}$, which lie in $\mathcal{N}_A$, can only be synchronous if they lie in the same layer. In particular, $\mathcal{N}$ cannot be layer periodic or fully synchronous.

Theorem 7.5. *(Assumptions and notation as above.) Let $\mathcal{F}$ be a feedback structure on the AFFNN $\mathcal{N}$. If $\mathcal{P} = \{P_a\}_{a\in\mathbf{s}}$ is a synchrony partition for $\mathcal{N}^\star$ then one (at least) of the following possibilities hold.*

(1) *$\mathcal{P}$ is layer complete and $\mathcal{F}$ is of type A.*

(2) *All nodes are synchronous and $\mathcal{F}$ is not of type $D^\star$.*

(3) *All nodes are asynchronous and $\mathcal{F}$ is of type A, D or $D^\star$.*

(4) *There exists $P \in \mathcal{P}$ such that P is contained in a layer and is not a singleton. $\mathcal{F}$ can be of type A, D or $D^\star$.*

(5) *If $P \neq \{\mathbf{k}\}$, $\mathcal{F}$ is of type D or $D^\star$, and there exist synchronous nodes i, j in different layers,*
   (a) *$i, j \in N_D$ and are synchronous in $\mathcal{P}_D = \mathcal{P} \cap N_D$.*
   (b) *If $\mathcal{F}$ is not of type $D^\star$, the partition $\mathcal{P}_D$ may be the fully synchronous partition of $\mathcal{N}_D$.*
   (c) *No node in $\mathcal{N}_D$ can be synchronous with a node in $\mathcal{N}_A^\star$ and there are no synchronous nodes in different layers of $\mathcal{N}_A^\star$.*

Remarks 7.6. (1) One only of (1–3) of Theorem 7.5 can occur and then (4,5) do not occur. On the other hand, (4) and (5) may both occur multiple times for the same synchrony partition. Note that if $\mathcal{N}^\star$ is of type A, then $\mathcal{N}_D = \emptyset$.

(2) If no node in layer 1 of $\mathcal{N}_A$ has a self loop, it is easy to find examples where every layer of $\mathcal{N}^\star$ contains at least two synchronous nodes.

(3) Synchrony for feedforward connected AFFNNs is classified in [2,

§4] and these results can be used to enumerate synchrony partitions for a specific network $\mathcal{N}_D$.

The following results are corollaries of Theorem 7.5.

Corollary 7.7. *Let $\mathcal{F}$ be a feedback structure on the AFFNN $\mathcal{N}$ such that at least one node in the first layer does not receive a feedback loop. Given a synchrony partition for $\mathcal{N}^\star$, we have precisely one of the following possibilities:*

(a) *All nodes are synchronous.*
(b) *Only nodes in the same layer can be synchronous.*
(c) *All nodes are asynchronous.*
(d) *There is a transversal γ with proper initial segment $\gamma_i \subset \mathcal{N}_D$ consisting of synchronous nodes with the remaining nodes of γ being asynchronous. Any synchronous nodes lying in different layers of $\mathcal{N}$ lie in $\mathcal{N}_D$.*

Corollary 7.8. *Let $\mathcal{F}$ be a feedback structure on the AFFNN $\mathcal{N}$ of type A. Consider a synchrony partition for $\mathcal{N}^\star$. We have precisely one of the following possibilities:*

(a) *The synchrony partition is layer complete. In particular, all nodes can be synchronous.*
(b) *Only nodes in the same layer can synchronize.*
(c) *All nodes are asynchronous.*

Examples 7.9. In Figure 13 we show two examples of AFFNNs with feedback. Network (a) is of type D, network (b) is of type A.

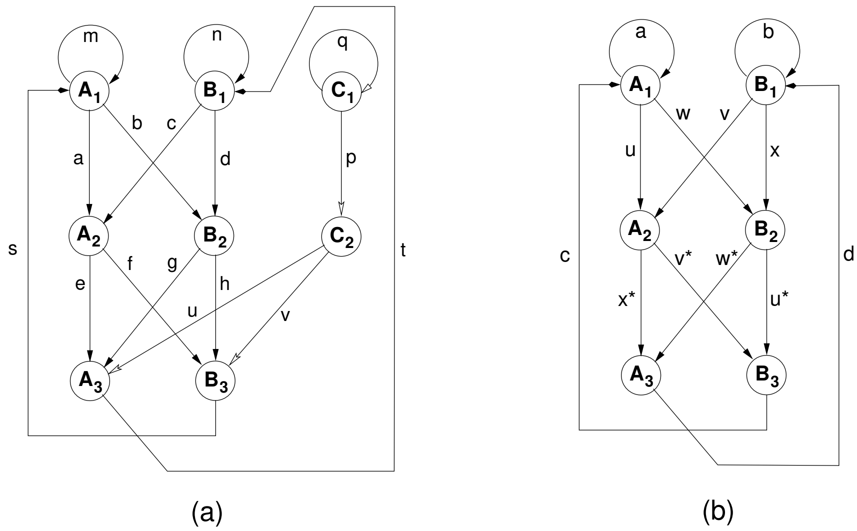

FIGURE 13. Both networks shown are feedforward connected AFFNNs with feedback structure. The network shown in (a) is of type D, or D* if we remove the self-loop from C_1. The network in (b) is of type A. For appropriate choice of weights, all types of synchrony given by Theorem 7.5 can be exhibited using these networks.

(a) If $a+c = b+d$, $e+g = f+h$, $s = t$, $m = n$, $u = v$ and $p = q$, then the node pairs A_1, B_1, A_2, B_2, A_3, B_3 and C_1, C_2 may all be synchronous and $\mathcal{N}_D$ has nodes C_1, C_2, connection $C_1 \to C_2$ and self-loop on C_1. If the local valencies are all equal, then all nodes may be synchronous. Otherwise, the only synchrony across layers is that between C_1 and C_2. For an open and dense set of weights, all nodes are asynchronous. If we remove the self-loop from C_1, then the network is of type D*. In this case, we still can have the node pairs A_1, B_1, A_2, B_2, A_3, B_3 synchronous but C_1, C_2 cannot be synchronous and all nodes cannot be synchronous. This example illustrates parts (3–5) of Theorem 7.5.

(b) If $a = w = w^\star$, $b = v = v^\star$, $c = x = x^\star$, $d = u = u^\star$, then $\{\{A_1, B_2, A_3\}, \{B_1, A_2, B_3\}\}$ is a layer complete synchrony partition. If the valencies are constant on layers but differ between layers, then $\{\{A_i, B_i\} \mid i \in \mathbf{3}\}$ will be a synchrony partition and nodes can be only synchronous if they are in the same layer. If $x^\star = v^\star$, $u^\star = w^\star$, $u \neq w$, $a \neq d$, then nodes in layer 3 can be synchronous with all other nodes asynchronous. For an open and dense set of weights, all nodes will be asynchronous. This example illustrates (1–4) of Theorem 7.5 for networks of type A.

The proof of Theorem 7.5 depends on some preliminary results.

The definition of synchrony translate continues to hold for AFFNNs with a feedback structure and it is easy to see that Lemma 5.10 remains valid for AFFNNs with feedback structure of type A. Lemma 6.8 holds for AFFNNs with feedback structure of type A if, for example, adjacent nodes on the path are not synchronous.

Proposition 7.10. *Let $\mathcal{F}$ be a feedback structure of type A on the AFFNN $\mathcal{N}$. Suppose $\mathcal{P}$ is a synchrony partition for $\mathcal{N}^\star$. If there exist synchronous nodes lying in different layers then $\mathcal{P}$ is either layer complete or the fully synchronous partition. In either case there is a transversal consisting of synchronous nodes.*

Proof. Suppose first that there is no pair of adjacent synchronous nodes but there exists at least one pair of synchronous nodes lying in different layers. Lemma 6.8(1) applies and we may follow the proof of Proposition 6.12 to obtain an integer $d \in [1, \ell - 1]$, $d|\ell$, such that $\mathcal{P}$ is layer

d-periodic. Let $i \in \mathcal{L}_1$ have a self-loop (at least one such node exists since $\mathcal{N}$ is an AFFNN). It follows by d-periodicity that there is a node $j \in \mathcal{L}_{d+1}$ which is synchronous to i. Since $i \in \mathcal{L}_1$ has a self-loop, there must be a node $i' \in \mathcal{L}_d$ which is synchronous with j and adjacent to j. Contradiction.

It follows that if there exists a pair of synchronous nodes lying in different layers, then there must exist a pair i, j of adjacent synchronous nodes lying in adjacent layers. Suppose that $i \to j$ and $i \in \mathcal{L}_t$, $j \in \mathcal{L}_{t+1}$, where $t \in [1, \ell]$ ($t+1$ is computed mod ℓ). By backward iteration, we obtain a path of adjacent synchronous nodes $i_1 \to \ldots \to i_t = i \to i_{t+1} = j$ and so a synchronous transversal if $t \in \{\ell-1, \ell\}$. Suppose we cannot find adjacent i, j with $t \in \{\ell-1, \ell\}$. Let $T \in [2, \ell-2]$ be the maximum value of t for which there exists an adjacent pair of i, j of synchronous nodes with $i \in \mathcal{L}_t$, $j \in \mathcal{L}_{t+1}$. By Lemma 5.10, we may choose a path $\tau : j \to \ldots \to i$ of length $L = \ell - 1$, mod ℓ. By our maximality assumption, τ will contain no pairs a, b of adjacent synchronous nodes with $a \in \mathcal{L}_s$, $b \in \mathcal{L}_{s+1}$, $s \in [T+1, \ell]$. It follows that Lemma 6.8 applies to give a translate $j'_1 \to \ldots \to j'_L = j$ of τ. Hence there exists $j' = j'_1 \in \mathcal{L}_{T+2}$ which is synchronous to j. Therefore, by synchrony, there exists $i' \in \mathcal{L}_{T+1}$ which is synchronous to j' and adjacent to j', contradicting the maximality of T.

Our arguments show that if there exist synchronous nodes lying in different layers there is a transversal consisting of synchronous nodes. Now apply the argument of Proposition 6.9 to deduce that $\mathcal{P}$ is either layer complete or the fully synchronous partition. $\square$

Lemma 7.11. *Let $\mathcal{F}$ be a feedback structure on the AFFNN $\mathcal{N}$ of type D or D* and $\mathcal{P}$ be a synchrony partition for $\mathcal{N}^\star$.*

(1) *If $\mathcal{N}^\star$ is of type D and there is a node in $\mathcal{N}_A^\star$ synchronous with a node in $\mathcal{N}_D$, then all nodes are synchronous: $\mathcal{P} = \{\mathbf{k}\}$.*
(2) *If $\mathcal{N}^\star$ is of type D*, it is not possible for a node in $\mathcal{N}_A^\star$ to be synchronous with a node in $\mathcal{N}_D$.*

Proof. Suppose $i \in \mathcal{N}_A^\star$ and $j \in \mathcal{N}_D$ are synchronous. Choose a closed path $\gamma : i = i_0 \to i_1 \to \ldots \to i_L = i$ which necessarily is contained in $\mathcal{N}_A^\star$ since the nodes in layer 1 of $\mathcal{N}_D$ can only have a self-loop but no feedback loop. Since j is synchronous with i, we can lift the final segment of γ to a path $\tau = j_p \to \ldots \to j_0 = s$ where $j_r \in \mathcal{N}_D$ and is synchronous with i_{L-r}, $0 \leq r \leq p$, and $j_p \in \mathcal{L}_1$. Either j_p has no self-loop—contradicting the synchrony of j_p and i_{L-p} or j_p has a self loop. In the latter case all the nodes $i_0, \ldots, i_{L-p}$ all receive inputs but only from nodes synchronous to i. Hence γ consists of nodes synchronous to i. Since γ contains a transversal of synchronous nodes, it follows easily by feedforward connectedness that $\mathcal{P} = \{\mathbf{k}\}$. In particular, $\mathcal{N}^\star$ is of type D since a network of type D* has a node with no input. ☐

Lemma 7.12. *Let $\mathcal{F}$ be a feedback structure on the AFFNN $\mathcal{N}$ of type D or D* and $\mathcal{P}$ be a synchrony partition for $\mathcal{N}^\star$. If there is a pair of synchronous nodes in $\mathcal{N}_A^\star$ lying in different layers, then $\mathcal{P} = \{\mathbf{k}\}$ and $\mathcal{F}$ is of type D.*

Proof. From Proposition 7.10 we have that the synchrony partition for $\mathcal{N}_A^\star$ is either layer complete or the fully synchronous partition. From

Lemma 7.1(1) we have that the number of layers of $\mathcal{N}_D$ (or $\mathcal{N}_D^\star$) is less than ℓ. It follows that there are at least two synchronous nodes i, j in $\mathcal{N}_A^\star$ such that one receives an input from a node d in $\mathcal{N}_D$ (or $\mathcal{N}_D^\star$) and the other does not receive any input from $\mathcal{N}_D$ (or $\mathcal{N}_D^\star$). Thus d must be synchronous with i, j. The result follows by Lemma 7.11. □

Proof of Theorem 7.5. Statement (1) follows from Proposition 7.10. Statement (2) is obvious since a network of type D* always contains a pair of asynchronous nodes. Statement (3) is clear since given the adjacency matrix, it is possible (and easy) to choose weights so that all nodes are asynchronous. Statement (5) follows from Lemmas 7.11, 7.12 and this leaves (4) as the only other possibility. □

8. Concluding remarks

Definitions for weight dynamics and examples of adaptation rules respecting synchrony were given in Section 3. All of this applies to feedforward networks with feedback. However, our interest lies rather in allowing the weights on a feedforward network to evolve to their final stable state—if that exists—and then investigating how the addition of fixed feedback loops can affect dynamics and structure of the resulting "optimized" network. In particular, quantifying bifurcations that can occur when feedback is added, and understanding the extent to which a judicious choice of feedback can optimize the function of the network [7, 8, 9] (this is part of the subject of [3]). In Sections 6, 7, we have seen how the addition of feedback can enrich the possible synchrony that occur in the network. In terms of unsupervised learning, this suggests enhancement of the potential for unsupervised learning even in a context where we do not add inhibition within layers of the network.

9. Acknowledgments

The authors acknowledge and appreciate the detailed comments of the referees which have helped improve the exposition and readability of the paper. The authors were partially supported by CMUP (UID/MAT/00144/2013), which is funded by FCT (Portugal) with national (MEC) and European structural funds through the programs FEDER, under the partnership agreement PT2020.

References

[1] M A D Aguiar and A Dias. 'The Lattice of Synchrony Subspaces of a Coupled Cell Network: Characterization and Computation Algorithm', *J. Nonlinear Sci.* **24** (6) (2014) 949–996.

[2] M A D Aguiar, A Dias, & F Ferreira. 'Patterns of synchrony for feed-forward and auto-regulation feed-forward neural networks', *Chaos* **27** (2017) 013103.

[3] M A D Aguiar, A Dias, & M J Field. 'Bifurcation, dynamics and feedback for adaptive feed-forward networks', in preparation.

[4] P Ashwin and A Rodrigues. 'Hopf normal form with S_N symmetry and reduction to systems of nonlinearly coupled phase oscillators', *Physica D: Nonlinear Phenomena* **325** (2016), 14–24.

[5] P Ashwin and J W Swift. 'The dynamics of n weakly coupled identical oscillators', *J. nonlinear Sci.* **2**(1) (1992), 69–108.

[6] V Belykh and M Hasler. 'Mesoscale and clusters of synchrony in networks of bursting neurons', *Chaos* **18** (2011), 016106.

[7] C Bick and M J Field. 'Asynchronous Networks and Event Driven Dynamics', *Nonlinearity* **30**(2) (2017), 558–594.

[8] C Bick and M J Field. ‘Asynchronous Networks: Modularization of Dynamics Theorem’, *Nonlinearity* **30**(2) (2017), 595–621.

[9] C Bick and M J Field. ‘Functional Asynchronous Networks: Factorization of Dynamics and Function’, (MATEC Web of Conferences, **83** (2016), CSNDD 2016 - International Conference on Structural Nonlinear Dynamics and Diagnosis, Marrakech, May 23–25, 2016).

[10] C M Bishop. *Neural Networks for Pattern Recognition.* (Oxford: Oxford University Press, 1995).

[11] N Caporale and Y Dan. ‘Spike timing dependent plasticity: a Hebbian learning rule’ *Annu. Rev. Neurosci.* **31** (2008), 25–36.

[12] S Chandra, D Hathcock, K Crain, T M Antonsen, M Girvan, and E Ott. ‘Modeling the network dynamics of pulse-coupled neurons’, *Chaos* **27** (2017).

[13] B. A. Davey and H. A. Priestley. *Introduction to Lattices and Order* (Cambridge University Press, 1990).

[14] G B Ermentrout and N Kopell. ‘Parabolic bursting in an excitable system coupled with a slow oscillation,’ *SIAM J. Appl. Math.* **46** (1986), 233-253.

[15] M J Field. ‘Heteroclinic Networks in Homogeneous and Heterogeneous Identical Cell Systems’, *J. Nonlinear Science* **25**(3) (2015), 779–813.

[16] W Gerstner, R Kempter, L J van Hemmen & H Wagner. ‘A neuronal learning rule for sub-millisecond temporal coding’, *Nature* **383** (1996), 76–78.

[17] D Fricker & R Miles. ‘EPSP Amplification and the precision of Spike Timing in Hippocampal Neurons’, *Neuron* **28** (2000), 559–569.

[18] S Goedeke and M Diesmann. ‘The mechanism of synchronization in feed-forward neuronal networks’, *New J. Phys.* **10** (2008).

[19] M Golubitsky and C Postlethwaite. ‘Feed-forward networks, center manifolds, and forcing’, *Disc. Cont. Dyn. Sys. Ser. A* **14**(2) (2012), 2913–2935.

[20] M Golubitsky, D G Schaeffer & I N Stewart. *Singularities and Groups in Bifurcation Theory, Vol. II*, (Appl. Math. Sci. **69**, Springer-Verlag, New York, 1988).

[21] M Golubitsky and I Stewart. 'Nonlinear dynamics of networks: the groupoid formalism'. *Bull. Amer. Math. Soc.* **43** (2006), 305–364.

[22] M Golubitsky, I Stewart, & A. Török. 'Patterns of synchrony in coupled cell networks with multiple arrows', *SIAM J. on Appl. Dyn. Sys.* **4**(1) (2005), 78–100.

[23] I Goodfellow, Y Bengio, and A Courville. *Deep Learning* (MIT Press, 2017).

[24] B Hadley. *Pattern Recognition and Neural Networks* (Cambridge University Press, 2007).

[25] M H Hassoun. *Fundamentals of Artificial Neural Networks* (MIT Press, Cambridge Mass. 1995).

[26] F C Hoppensteadt and E M Izhikevich. *Weakly Connected Neural Networks* (Applied Mathematical Science vol. 126, Berlin: Springer, 1997).

[27] J Ito and K Kaneko. 'Self-organized hierarchical structure in a plastic network of chaotic units', *Neural Networks* **13**(3) (2000), 275–281.

[28] J Ito and K Kaneko. 'Spontaneous Structure Formation in a Network of Chaotic Units with Variable Connection Strengths', *Phys. Rev. Lett* **88**(2) (2002),

[29] H Kamei and P J A Cock. 'Computation of balanced equivalence relations and their lattice for a coupled cell network', *SIAM J. Appl. Dyn. Sys.* **12** (2013), 352–382.

[30] K Kaneko. 'Relevance of dynamical clustering to biological networks', *Physica D* **75** (1994), 55–73.

[31] Y Kuramoto. *Chemical Oscillations, Waves, and Turbulence* (Berlin: Springer, 1984).

[32] H L Lee, V Padmanabhan, & S Whang. 'Information Distortion in a Supply Chain: The Bullwhip Effect', *Management Science* **43**(4) (1997), 546–558.

[33] K K Lin, E Shea-Brown, & L-S Young. 'Spike-Time Reliability of Layered Neural Oscillator Networks', *J. Comp. Neurosci.* **27** (2009), 135–160.

[34] T Masquelier, R Guyonneau, & S J Thorpe. 'Spike Timing Dependent Plasticity Finds the Start of Repeating Patterns in Continuous Spike Trains', *PLoS ONE* **3**(1) (2008), e1377.

[35] T Masquelier, R Guyonneau, & S J Thorpe. 'Competitive STDP-Based Spike Pattern Learning', *Neural Computation* **21**(5) (2009), 1259–1276

[36] T Masquelier, E Hugues, G Deco & S J Thorpe. 'Oscillations, Phase-of-Firing-Coding, and Spike Timing-Dependent Plasticity: An Efficieint Learning Scheme', *J. of Neuroscience* **29**(43) (2009), 13484–13493.

[37] M Minsky and S Papert. *Perceptrons: An introduction to Computational Geometry* (MIT press, 1969).

[38] A Morrison, M Diesmann, & W Gerstner. 'Phenomenological models of synaptic plasticity based on spike timing', *Biol. Cybern.* (2008), 459-478.

[39] M J Panaggio and D M Abrams. 'Chimera states: coexistence of coherence and incoherence in networks of coupled oscillators', *Nonlinearity* **28**(3) (2015), R67.

[40] Pedantic of Purley. 'Uncircling the Circle: Part I'. (On line article https://www.londonreconnections.com/2013/uncircling-circle-part-1/, in *London Reconnections*, October 3, 2013.)

[41] B Rink and J Sanders. 'Amplified Hopf bifurcations in feed-forward networks', *SIAM J. Appl. Dyn. Sys.* **12**(2) (2013), 1135–1157.

[42] F Rosenblatt. 'The Perceptron: A Probabilistic Model For Information Storage And Organization In The Brain'. *Psych. Rev.* **65**(6) (1958), 386–408.

[43] P Sailamul, J Jang, & S-B Paik. 'Synaptic convergence regulates synchronization-dependent spike transfer in feedforward neural networks', *J. Comput. Neurosci.* **43** (2017), 189–202.

[44] J Schmidhuber. 'Deep Learning in Neural Networks: An Overview'. *Neural Networks* **61** (2015), 85-117.

[45] I Stewart. 'The lattice of balanced equivalence relations of a coupled cell network', *Math. Proc. Cambridge Philos. Soc.* **143** (1) (2007), 165–183.

[46] I Stewart, M Golubitsky, & M. Pivato. 'Symmetry groupoids and patterns of synchrony in coupled cell networks', *SIAM J. on Appl. Dyn. Sys.* **2**(4) (2003), 609–646.

[47] F Takens. *Detecting strange attractors in turbulence* (Lecture Notes in Mathematics **898**, Berlin:Springer-Verlag, 1981).

Manuela Aguiar, Faculdade de Economia, Centro de Matemática, Universidade do Porto, Rua Dr Roberto Frias, 4200-464 Porto, Portugal.

E-mail address: maguiar@fep.up.pt

Ana Dias, Departamento de Matemática, Centro de Matemática, Universidade do Porto, Rua do Campo Alegre, 687, 4169-007 Porto, Portugal

E-mail address: apdias@fc.up.pt

Michael Field, Department of Mathematics, Imperial College, South Kensington Campus, London SW7 2AZ, UK

Current address: Department of Mathematics, Rice University MS-136, 6100 Main St., Houston, TX 77005, USA

E-mail address: mikefield@gmail.com