Where is This Leading Me

Treur, Jan

published in
Mental Models and their Dynamics, Adaptation and Control
2022

DOI (link to publisher)
10.1007/978-3-030-85821-6_18

document version
Publisher's PDF, also known as Version of record

document license
Article 25fa Dutch Copyright Act

citation for published version (APA)
Treur, J. (2022). Where is This Leading Me: Stationary Point and Equilibrium Analysis for Self-Modeling Network Models. In J. Treur, & L. Van Ments (Eds.), *Mental Models and their Dynamics, Adaptation and Control: A Self-Modeling Network Modeling Approach* (pp. 491-535). (Studies in Systems, Decision and Control; Vol. 394). Springer Nature Switzerland AG. Advance online publication. https://doi.org/10.1007/978-3-030-85821-6_18

Chapter 18
Where is This Leading Me: Stationary Point and Equilibrium Analysis for Self-Modeling Network Models

Jan Treur

Abstract In this chapter, analysis methods for the dynamics of self-modeling network models in relation to their network structure are presented. In particular, stationary points and equilibria are addressed and related to the network structure. It is shown how such analyses can be used for verification purposes: to verify whether an implemented network model used for simulation is correct with respect to the design description of the network's structure. An always applicable method is presented first. It is based on substitution of state values from simulations in stationary point or equilibrium equations, which can always be done. In addition, methods are presented that are applicable for certain groups of network models, where the aggregation is specified by combination functions for which equilibrium equations can be solved symbolically. As shown, these methods cover cases of self-model states for adaptation principles such as Hebbian learning for mental networks and Bonding based on homophily for social networks. In addition, such methods are shown to cover cases where the combination functions for aggregation satisfy certain properties such as being monotonically increasing, scalar-free, and normalised. The analysis for this class of functions used for aggregation also takes into account the network's connectivity in terms of its strongly connected components. This provides a class of functions which includes nonlinear functions but in contrast to often held beliefs, still enables analysis of the emerging network dynamics as well as linear functions do. Within this class, two specific subclasses of nonlinear functions (weighted Euclidean functions and weighted geometric functions) are addressed. Focusing on them in particular, it is illustrated in detail how methods for equilibrium analysis as normally only used for linear functions (based on a symbolic linear equation solver), can be applied to predict the state values in equilibria for such nonlinear cases as well. Finally, it shown how a stratified form of the condensation graph based on a network's strongly connected components can be used in equilibrium analysis.

Keywords Stationary point · Equilibrium analysis · Self-modeling network models · Strongly connected components

J. Treur (✉)
Social AI Group, Department of Computer Science, Vrije Universiteit Amsterdam, Amsterdam, The Netherlands
e-mail: j.treur@vu.nl

J. Treur and L. Van Ments (eds.), *Mental Models and Their Dynamics, Adaptation, and Control*, Studies in Systems, Decision and Control 394,
https://doi.org/10.1007/978-3-030-85821-6_18

18.1 Introduction

Self-modeling network models can show complex dynamics. Usually, emerging dynamic properties of dynamical models can be analysed by conducting simulation experiments. But some specific types of properties can also be found by mathematical analysis. Examples of properties that can be analyzed in such a manner are:

- Properties describing whether for some values for the states no change occurs (stationary points or equilibria), and how such values may depend on the values of the characteristics of the network model and/or the initial values for the states
- Properties describing whether certain states in the model converge to some joint limit value and how this may depend on the values of the characteristics of the network model and/or the initial values for the states
- Properties describing whether some state within the network model will show monotonically increasing or decreasing values over time (monotonicity)
- Properties describing situations in which no convergence takes place but in the end a specific sequence of values is repeated all the time (limit cycle).

Three types of emerging dynamics are often distinguished:

- **Reaching an equilibrium**
  In this case a socalled *equilibrium state* is reached, in which for all states the values do not change anymore. This often happens; for example, see Fig. 18.4.
- **Ending up in a limit cycle**
  The behaviour ends up in a regular repeating pattern of values (a periodic pattern) for the states; this is called a *limit cycle*. In Fig. 18.2 an example of this is shown, taken from (Treur 2016).
- **Chaotic behaviour**
  The behaviour is usually (loosely) called *chaotic* if there is no observed regularity in it like for the first two types: no equilibrium is reached and also no periodic pattern as a limit cycle. Lorenz (1963) used as title for his paper on chaotic behaviour 'Deterministic Nonperiodic Flow'. In Mathematics, the area of Chaos Theory has developed more specific definitions for chaotic behaviour, usually involving that the outcome is very sensitive for the values of the initial settings; e.g., (Lorenz 1963): the present determines the future but the approximate present does not approximately determine the future. An often cited example or metaphor is that a butterfly at one place in the world can cause a tornado somewhere else (the butterfly effect).

An example (seemingly) showing the third type of emerging behaviour may be found in (Treur 2020a), Ch. 6. Note that a pattern can initially look like this last type, but later on may still turn out to end up into one of the other two types.

Such properties of a network model's dynamics as found and analysed can be used for verification of the network model by checking them against the values observed in simulation experiments. Typically such properties take the form of equations for values of one state in relation to values of connected states. If one of these equations is not (approximately) fulfilled by the values found in a simulation (and the mathematical analysis was done in a correct manner), then this reveals that there is some error in the implementation of the model in comparison to its design description. In some cases, but certainly not always, such equations can also be solved in an analytical manner in the sense that equilibrium state values are expressed by a symbolic expression (a formula) in terms of the network characteristics. However, for the purpose of verification, solving equilibrium equations is not required as the equations can also be checked by substitution of simulation values in them.

In this chapter some methods to analyse dynamics in network models will be described in particular in the setting of self-modeling network models with self-model states for adaptation principles such as Hebbian learning for mental networks and bonding based on homophily for social networks.

As another case, social dynamics described by dynamics of node states (for example, for the individuals' opinions, intentions, emotions, beliefs, …) within social network models also depend on a number of network characteristics such as the weights of the connections and the aggregation of impacts from different nodes. While for networks usually there is much attention for the structure of nodes and connections, the role of the aggregation characteristics is often neglected. Nevertheless, these aggregation characteristics also play an important role in the dynamics within a network; for example, whether or not within a well-connected group in the end a common opinion, intention, emotion or belief is reached (a joint value for all node states) also depends on the aggregation characteristics. Often, only silent assumptions are made about these aggregation characteristics. For social network models usually linear forms of aggregation are applied. Indeed, when using linear aggregation theorems exist specifying conditions under which all node states converge to the same value, in particular when the network is strongly connected in the sense that from every node there is a path to every other node. In contrast, for neural network models traditionally often some type of logistic sum format is applied and for such functions analysis is indeed much harder than for linear functions.

The often occurring use of linear functions for aggregation for social network models may be based on a more general belief that dynamical system models can be analysed better for linear functions than for nonlinear functions. Although there may be some truth in this if specifically logistic nonlinear functions are compared to linear functions, in the current chapter it is shown that such a belief is not correct in general. It is shown that also classes of nonlinear functions exist that enable good analysis possibilities when it comes to the emerging dynamics within a network model. Such classes and the dynamics they entail are analysed here in some depth, thereby also not using any conditions on the connectivity but instead exploiting for any network its structure of strongly connected components. Among others, following (Treur 2020a) in the current chapter theorems are discussed specifying conditions under which all node states converge to the same value (for example, achieving a common decision or belief within a group). These theorems do not impose any conditions on connectivity and for aggregation apply to such classes of nonlinear functions as well as they apply to the class of linear functions. Moreover, for some (but not all) of these classes of 'well-behaving' nonlinear functions it is found out that they can be (indirectly) related to linear functions by some form of function transformation, which then enables application of linear analysis methods such as symbolically solving sets of linear equations including parameters.

In this chapter, first in Sect. 18.2 for some aspects of dynamics (in particular, stationary points and equilibria) of network models criteria are introduced in terms of the network structure characteristics such as aggregation and connectivity characteristics. These criteria can be used to verify (the implementation of) a network model against its design description based on inspection of stationary points or equilibria in three different manners:

- by *substitution* of observed simulation values in the equations (addressed in Sect. 18.3)
- by *symbolically solving* these equations to obtain symbolic expressions in terms of network characteristics and comparing these expressions to observed simulation values (addressed for different cases in Sects. 18.4–18.8)
- by deriving *general theorems* from them of the form 'network structure properties imply network behaviour properties' and comparing their conclusions to simulation values (addressed in some depth in Sects. 18.9–18.12)

18.2 Modeling and Analysis of Dynamics within Network Models

In this section, the underlying network-oriented modelling approach used is briefly discussed. Following (Treur 2020b), a temporal-causal network model is specified by the following types of network characteristics (here X and Y denote nodes of the network, also called states, which have state values $X(t)$ and $Y(t)$ over time t):

- **Connectivity Characteristics**

Connections from a state X to a state Y and their *weights* $\boldsymbol{\omega}_{X,Y}$

- **Aggregation Characteristics**

For any state Y, some *combination function* $\mathbf{c}_Y(V_1, \ldots, V_k)$ defines the aggregation that is applied to the *single impacts* $V_i = \boldsymbol{\omega}_{X_i,Y} X_i(t)$ on Y from its incoming connections from states $X_1, \ldots, X_k$.

- **Timing Characteristics**

Each state Y has a *speed factor* $\boldsymbol{\eta}_Y$ defining how fast it changes for given impact.

The following generic (canonical) difference equation used for simulation and analysis purposes incorporates these network characteristics $\boldsymbol{\omega}_{X,Y}$, $\mathbf{c}_Y$, $\boldsymbol{\eta}_Y$ in a numerical format:

$$Y(t + \Delta t) = Y(t) + \boldsymbol{\eta}_Y[\mathbf{aggimpact}_Y(t) - Y(t)]\Delta t \tag{18.1}$$

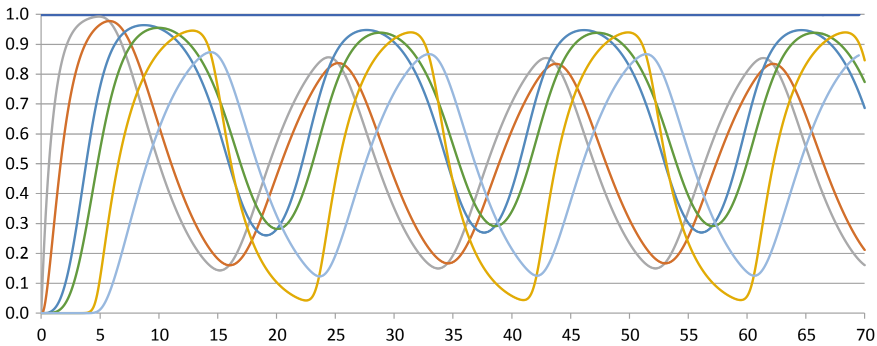


Fig. 18.1 Example network simulation ending up in a limit cycle; adopted from (Treur 2016)

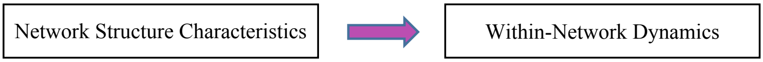


Fig. 18.2 The general principle that a network's structure implies the within-network dynamics

where

$$\mathbf{aggimpact}_Y(t) = \mathbf{c}_Y(\boldsymbol{\omega}_{X_1,Y} X_1(t), \ldots, \boldsymbol{\omega}_{X_k,Y} X_k(t))$$

for any state Y and where X_1 to X_k are the states from which Y gets its incoming connections.

This is a specific way of expressing the general principle that within-network dynamics is implied (or entailed) by the network's structure characteristics; see also Fig. 18.1.

The timing characteristics specified by speed factors $\boldsymbol{\eta}_Y$ enable to model more realistic processes for which not all states change in a synchronous manner. Network models that do not possess this option are less flexible as they silently impose synchronous processing as an artefact. The aggregation characteristics specified by the choice of combination functions $\mathbf{c}_Y$ and their parameters provide another form of flexibility to fit better to specific realistic applications. Also in this case, network models that do not possess such an option are less flexible and also silently impose artefacts that may make them fit less to specific applications. For example, for aggregation in social networks often linear functions are used for aggregation, which sometimes may be considered more like a tradition or custom than a deliberate choice.

The following types of properties are often considered to analyse the behaviour of dynamical systems in general.

Definition (stationary point, increasing, decreasing, equilibrium) Let Y be a network state

- Y has a *stationary point* at t if $\mathbf{d}Y(t)/\mathbf{d}t = 0$
- Y is *increasing* at t if $\mathbf{d}Y(t)/\mathbf{d}t > 0$
- Y is *decreasing* at t if $\mathbf{d}Y(t)/\mathbf{d}t < 0$

- The network model is in *equilibrium* a t if every state Y of the model has a stationary point at t.

For the specific case of network models, the following criteria in terms of the network characteristics $\boldsymbol{\omega}_{X,Y}$, $\mathbf{c}_Y$, $\boldsymbol{\eta}_Y$ can be derived from the generic difference Eq. (18.1); see also (Treur 2016, 2018).

Criteria for Within-Network Dynamics

Let Y be a state and X_1, …, X_k the states connected toward Y. For nonzero speed factors $\boldsymbol{\eta}_Y$ the following criterion (18.2) for a stationary point can be directly derived from the standard canonical Eq. (18.1):

$$\mathbf{c}_Y(\boldsymbol{\omega}_{X_1,Y}X_1(t), \ldots, \boldsymbol{\omega}_{X_k,Y}X_k(t)) = Y(t) \tag{18.2}$$

As can be noted, the above criterion for a network having a stationary point (or being in an equilibrium) shows how such a property of network dynamics depends on the aggregation characteristics specified by the combination functions $\mathbf{c}_Y$ and the connectivity characteristics specified by the connection weights $\boldsymbol{\omega}_{X_i,Y}$. Moreover, note that the equation is not a dynamic but a static equation for the values at the same time point t. For the sake of simplicity, if no confusion is expected often the variables in such an equation are named after the related state names (thereby accepting overloading of these names); then the equation simply is:

$$\mathbf{c}_Y(\boldsymbol{\omega}_{X_1,Y}X_1, \ldots, \boldsymbol{\omega}_{X_k,Y}X_k) = Y \tag{18.3}$$

where now X_1, …, X_k, Y are variables indicating numbers. This static equation in terms of the network structure characteristics is called a *stationary point equation*, and if an equilibrium is considered it is called an *equilibrium equation*. Again, note that although this equation is used to analyse the network's behaviour, this essentially is an equation expressed in terms of the network's structure. It also reflects the above-mentioned general principle that a network's behaviour is implied (or entailed) by the network's structure, as depicted in Fig. 18.2. The criteria for the different cases are summarised in Table 18.1.

Table 18.1 Criteria for types of dynamics in terms of network characteristics (assuming nonzero speed factors)

Dynamics	Criterion in terms of network characteristics	
Y has a stationary point at t	$\mathbf{aggimpact}_Y(t) = Y(t)$	$\mathbf{c}_Y(\boldsymbol{\omega}_{X_1,Y}X_1, \ldots, \boldsymbol{\omega}_{X_k,Y}X_k) = Y$
Y is increasing at t	$\mathbf{aggimpact}_Y(t) > Y(t)$	$\mathbf{c}_Y(\boldsymbol{\omega}_{X_1,Y}X_1, \ldots, \boldsymbol{\omega}_{X_k,Y}X_k) > Y$
Y is decreasing at t	$\mathbf{aggimpact}_Y(t) < Y(t)$	$\mathbf{c}_Y(\boldsymbol{\omega}_{X_1,Y}X_1, \ldots, \boldsymbol{\omega}_{X_k,Y}X_k) < Y$
The network model is in equilibrium a t	$\mathbf{aggimpact}_Y(t) = Y(t)$ for every state Y	$\mathbf{c}_Y(\boldsymbol{\omega}_{X_1,Y}X_1, \ldots, \boldsymbol{\omega}_{X_k,Y}X_k) = Y$ for every state Y

Table 18.2 Overview of deviations for stationary point or equilibrium equations

State nr	State name Y	Time point t	$\mathbf{aggimpact}_Y(t)$	$Y(t)$	Deviation
X_4	ps_a	35	0.999315	0.99903	0.00029
X_5	srs_e	35	…	…	…

18.3 Verification of a Network Model via Checking the Stationary Point Equations

As already previewed in the introduction Sect. 18.1, the criteria described in Sect. 18.2 can be used to verify (the implementation of) a network model based on inspection of stationary points or equilibria in three different manners:

- by *substitution* of observed simulation values in the equations (addressed in the current section)
- by *symbolically solving* these equations to obtain symbolic expressions in terms of network characteristics and comparing these expressions to observed simulation values (addressed for different cases in Sects. 18.4–18.8)
- by deriving *general theorems* from them of the form 'network structure properties imply network behaviour properties' and comparing their conclusions to simulation values (addressed in some depth in Sects. 18.9–18.12).

Note that in a given simulation the stationary points that are identified are usually approximately stationary; how closely they are approximated depends on different aspects, for example on the step size, or on how long the simulation is done.

1. Generate a simulation
2. For a number of states Y identify stationary points with their time points t and state values $Y(t)$
3. For each of these stationary points for a state Y at time t identify the values $X_1(t)$**, …,** $X_k(t)$ at that time of the states X_1, …, X_k connected toward Y
4. Substitute all these values $Y(t)$ and $X_1(t)$**, …,** $X_k(t)$ in the criterion $\mathbf{c}_Y(\boldsymbol{\omega}_{X_1,Y} X_1, \ldots, \boldsymbol{\omega}_{X_k,Y} X_k) = Y$
5. If the equation holds (for example, with an accuracy < 10^{-2}), then this test succeeds, otherwise it fails
6. If this test fails, then it has to be explored were the error can be found.

To illustrate these notions and analysis method, consider the example with connectivity depicted in Fig. 18.3, and an example simulation shown in Fig. 18.4. This is a mental network model for how a person is sensing (sensor state ss_s) a stimulus s in the world (word state ws_s), represents this (representation state srs_s), and is triggered to prepare (preparation state ps_a) and perform (execution state es_a) action a, after evaluation of the predicted (predicted effect representation state srs_e) effect e of this action. In simulations it can be seen that as a result of a constant value a of stimulus ws_s all state values are increasing until they reach an equilibrium value a as well. The question then is whether these observations based on one or more

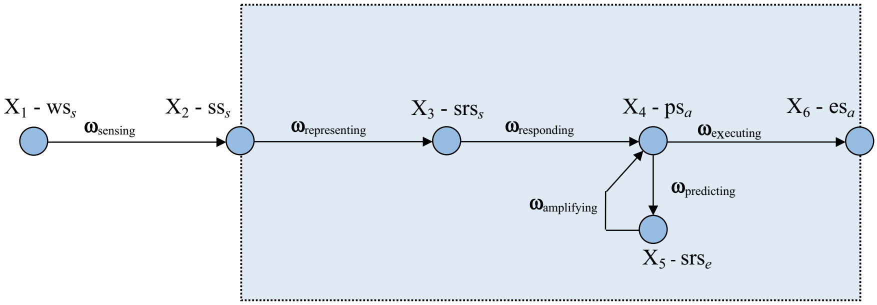


Fig. 18.3 Connectivity of the example model

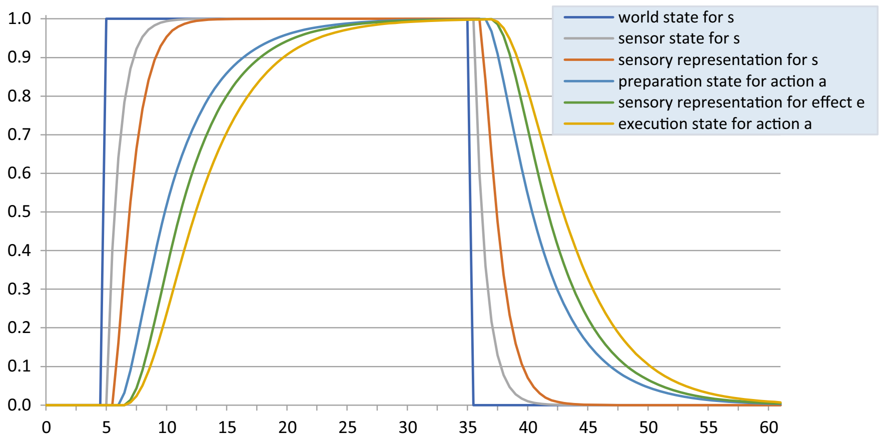


Fig. 18.4 Simulation example for the model depicted in Fig. 18.1 using identity and sum combination functions

simulation experiments are in agreement with a mathematical analysis. Combination functions are here the sum function and the identity function, and all connections have weight 1, except the connections to ps_a, which have weight 0.5. Later on the type of combination function will be varied.

In Fig. 18.4 it can be seen that as a result of the stimulus all states are increasing until time point 35, after which they start to decrease as the stimulus disappears. Just before time point 35 all states are almost stationary. If the stimulus is not taken away after this time point this trend is continued, and an equilibrium state is approximated. The question then is whether these observations based on one or more simulation experiments are in agreement with a mathematical analysis. If it is found out that they are in agreement with the mathematical analysis, then this provides some extent of evidence that the implemented model is correct in comparison to the design description. If they turn out not to be in agreement with the mathematical analysis, then this indicates that probably there is something wrong, and further inspection and correction is needed.

The analysis method described above can be illustrated for this example of Figs. 18.3 and 18.4 as follows. For example, consider state ps_a (which is X_4). According to the criterion in Sect. 18.2, Table 18.1 the equation expressing that state ps_a is stationary at time t is

$$\boldsymbol{\omega}_{\text{responding}} X_3(t) + \boldsymbol{\omega}_{\text{amplifying}} X_5(t) = X_4(t) \tag{18.4}$$

Now time t is left out of the equation by using variables X_i for the values $X_3 = X_3(t)$, $X_4 = X_4(t)$, and $X_5 = X_5(t)$. Then this becomes the following (static) equation in variables X_i, called the *stationary point equation*:

$$\boldsymbol{\omega}_{\text{responding}} X_3 + \boldsymbol{\omega}_{\text{amplifying}} X_5 = X_4 \tag{18.5}$$

At time point $t = 35$ (where all states are close to being stationary) the following simulation values occur:

$$\begin{aligned} X_3 &= \text{srs}_s(35) &= 1.00000 \\ X_4 &= \text{ps}_a(35) &= 0.99903 \\ X_5 &= \text{srs}_e(35) &= 0.99863 \end{aligned}$$

Moreover, in the simulation $\boldsymbol{\omega}_{\text{responding}} = \boldsymbol{\omega}_{\text{amplifying}} = 0.5$ was used. All these values can be substituted in the above equation:

$$\begin{aligned} &0.5^* 1.00000 + 0.5^* 0.99863 = 0.99903 \\ &0.999315 = 0.99903 \end{aligned}$$

It turns out that the equation is fulfilled with a very small deviation $< 10^{-3}$. This gives a piece of evidence that the network model as implemented indeed does what it was meant to do according to the design description. The step size Δt for the simulation here was 0.5, which is not even so small. For still more accurate results it is advisable to choose a smaller step size. So, having the equations for stationary points for all states provides a means to verify the implemented model in comparison to the model's design description. The equations for stationary points themselves can easily be obtained from the design description in a systematic manner according to the criteria in Sect. 18.2.

Note that this method works without having to solve the equations, only substitution takes place; therefore it works for any choice of combination function. Moreover, note that the method also works when there is no equilibrium but the values of the states fluctuate all the time, according to a recurring pattern (a limit cycle), like in Fig. 18.1. In such cases for each state there are maxima (peaks) and minima (dips) which also are stationary just for an instant. The method can be applied to such a type of stationary points as well; here it is still more important to choose a small step size as each stationary point occurs at just one time point.

18.4 Verification of a Network Model via Solving Equilibrium Equations

There is still another method possible that is sometimes proposed; this method is applied for the case of an equilibrium (where all states have a stationary point simultaneously), and is based on solving the equations for the equilibrium values first. This can provide explicit expressions for equilibrium values in terms of the parameters of the model. Such expressions can be used to predict equilibrium values for specific simulations, based on the choice of parameter values. This method provides more than the previous method, but it should be kept in mind that some types of equations cannot be solved symbolically (but still numerically). For example, when logistic combination functions are used the equations cannot be solved symbolically. The method in general works best for equilibria, so it is described for that case; it goes as follows.

1. Consider the equilibrium equations for all states Y:

$$\mathbf{c}_Y(\boldsymbol{\omega}_{X_1,Y} X_1(t),\ \ldots,\ \boldsymbol{\omega}_{X_k,Y} X_k(t)) = Y(t) \tag{18.6}$$

2. Leave the t out and as in Sect. 18.2 denote the values $X_i(t)$ and $Y(t)$ by variables X_i and Y

$$\mathbf{c}_Y(\boldsymbol{\omega}_{X_1,Y} X_1, \ldots, \boldsymbol{\omega}_{X_k,Y} X_k) = Y \tag{18.7}$$

   For the n states $X_1, \ldots, X_n$ of the model, equilibria for the network are described by solutions $\mathbf{X_1}, \ldots, \mathbf{X_n}$ of the following set of n equilibrium equations:

$$\begin{aligned} &\mathbf{c}_{X_1}(\boldsymbol{\omega}_{X_1,X_1} X_1,\ \ldots,\ \boldsymbol{\omega}_{X_n,X_1} X_n) = X_1 \\ &\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ &\mathbf{c}_{X_n}(\boldsymbol{\omega}_{X_1,X_n} X_1,\ \ldots,\ \boldsymbol{\omega}_{X_n,X_n} X_n) = X_n \end{aligned} \tag{18.8}$$

3. If possible, solve these equations mathematically in an explicit analytical form: for each state X_i a mathematical formula $\boldsymbol{X_i} = \ldots$ in terms of the network characteristics of the model (connection weights and parameters in the combination function c_{X_i}**(..)**, such as scaling factors $\boldsymbol{\lambda}$ or the steepness $\boldsymbol{\sigma}$ and threshold $\boldsymbol{\tau}$ in a logistic sum combination function); more than one solution is possible. If that's not possible symbolically, use a numerical solver to find a solution.
4. Generate a simulation
5. Identify equilibrium values in this simulation

6. If for all states Y the predicted value Y from a solution of the equilibrium equations equals the value for Y obtained from the simulation (for example, with a deviation < 10^{-2}), then this test succeeds, otherwise it fails
7. If this test fails, then it has to be explored where the error can be found.

As an illustration, for the example shown in Fig. 18.3 (using the sum function), the equations for all states found in step 2 are as shown in Table 18.3. In Sect. 18.5 it is shown how they can be solved easily by using a symbolic linear solver.

Note that sometimes the method can also be applied to part of a network that concerns states that all have a stationary point simultaneously. In subsequent sections, it will be illustrated how this method based on solving the equilibrium equations works for a number of cases, not only for cases with linear combinations functions, but also for cases with nonlinear combination functions. In particular, the following examples of this are addressed in subsequent sections:

- for linear combination functions
  - a (scaled) sum combination function (Sect. 18.5)
- for nonlinear combination functions
  - for Euclidean combination functions (Sect. 18.6)
  - for geometric combination functions (Sect. 18.7)
  - for self-model states for the Hebbian learning principle (Sect. 18.8)
  - for self-model states for the bonding by homophily principle (Sect. 18.8).

For the cases described in Sects. 18.5–18.8 below, explicit symbolic expressions are found for the (predicted) equilibrium values in terms of the network characteristics such as connection weights $\boldsymbol{\omega}$ and combination function parameters as scaling factors $\boldsymbol{\lambda}$, persistence factors $\boldsymbol{\mu}$ and tipping points $\boldsymbol{\tau}$ (so, for these no specific simulation values are needed at forehand).

However, note that there are also cases in which explicit symbolic solutions cannot be determined, for example, when logistic combination functions are used. In such cases an explicit analytical solution by a more generic expression which depends, as a function, on the network characteristics (as enabled by method in the current section) cannot be obtained. In these cases equilibria can still be determined for specific cases either by numerically solving the equations by some numerical approximation method, or by the substitution method discussed in Sect. 18.3. In addition, for some classes of (linear and nonlinear) combination functions, general theorems can be found relating equilibria to network characteristics that can be applied (as will be shown in Sects. 18.9–18.12).

18.5 Using a Linear Solver to Symbolically Solve Linear Equilibrium Equations

In this section it will be illustrated how the analysis method described in Sect. 18.4 can be used for linear combination functions thereby using a symbolic linear equation solver. Linear equations with parameters can be solved symbolically in an automated manner by such a symbolic linear solver, for example, the WIMS Linear Solver freely available online at URL.

https://wims.univ-cotedazur.fr/wims/en_tool~linear~linsolver.en.html

Recall the linear equations from Sect. 18.4, Table 18.3. When the $\boldsymbol{\omega}$ is replaced by **w**, they can directly be copied as input for the WIMS Linear Solver, resulting in the upper picture in Fig. 18.5. Note that the network characteristics $\boldsymbol{\omega}_{\text{sensing}}$ and so on and the constant stimulus value a have to be entered in the slot for equation parameters below the main slot for the equations; this prevents the solver to mistakenly consider them as variables instead of parameters of the equations.

Summarizing, by this symbolic solver the equilibrium values for all states have been expressed in terms of the value a of the external state ws_s and the connection weights (apparently assuming $\boldsymbol{\omega}_{\text{amplifying}}\ \boldsymbol{\omega}_{\text{predicting}} \neq 1$, but as $\boldsymbol{\omega}_{\text{amplifying}} = 0.5$ and $\boldsymbol{\omega}_{\text{predicting}} = 1$ in the example simulation this is fulfilled) a shown in Table 18.4. For example, if the external stimulus has level $a = 1$ this becomes:

$$X_1 = 1$$
$$X_2 = \boldsymbol{\omega}_{\text{sensing}}$$
$$X_3 = \boldsymbol{\omega}_{\text{representing}}\boldsymbol{\omega}_{\text{sensing}}$$
$$X_4 = \boldsymbol{\omega}_{\text{responding}}\boldsymbol{\omega}_{\text{representing}}\boldsymbol{\omega}_{\text{sensing}}/(1 - \boldsymbol{\omega}_{\text{amplifying}}\boldsymbol{\omega}_{\text{predicting}})$$
$$X_5 = \boldsymbol{\omega}_{\text{predicting}}\boldsymbol{\omega}_{\text{responding}}\boldsymbol{\omega}_{\text{representing}}\boldsymbol{\omega}_{\text{sensing}}/(1 - \boldsymbol{\omega}_{\text{amplifying}}\boldsymbol{\omega}_{\text{predicting}})$$
$$X_6 = \boldsymbol{\omega}_{\text{executing}}\boldsymbol{\omega}_{\text{predicting}}\boldsymbol{\omega}_{\text{responding}}\boldsymbol{\omega}_{\text{representing}}\boldsymbol{\omega}_{\text{sensing}}/(1 - \boldsymbol{\omega}_{\text{amplifying}}\boldsymbol{\omega}_{\text{predicting}})$$

Moreover, if all connection weights are 1, except that $\boldsymbol{\omega}_{\text{responding}} = 0.5$ and $\boldsymbol{\omega}_{\text{amplifying}} = 0.5$, as in the example simulation shown in Fig. 18.4, the values all become 1. Indeed, in the example simulation in Fig. 18.4 it can be seen that in the time period that the world state has value $a = 1$ all values go to 1. The solution of the equilibrium equations in terms of the connection weights can be used to predict that when

Table 18.3 Overview of linear equilibrium equations for the example of Figs. 18.3 and 18.4

State **nr**	State name	Equilibrium equation
X_1	ws_s	$X_1 = a$
X_2	ss_s	$\boldsymbol{\omega}_{\text{sensing}}\ X_1 = X_2$
X_3	srs_s	$\boldsymbol{\omega}_{\text{representing}}\ X_2 = X_3$
X_4	ps_a	$\boldsymbol{\omega}_{\text{responding}}\ X_3 + \boldsymbol{\omega}_{\text{amplifying}}\ X_5 = X_4$
X_5	srs_e	$\boldsymbol{\omega}_{\text{predicting}}\ X_4 = X_5$
X_6	es_a	$\boldsymbol{\omega}_{\text{executing}}\ X_5 = X_6$

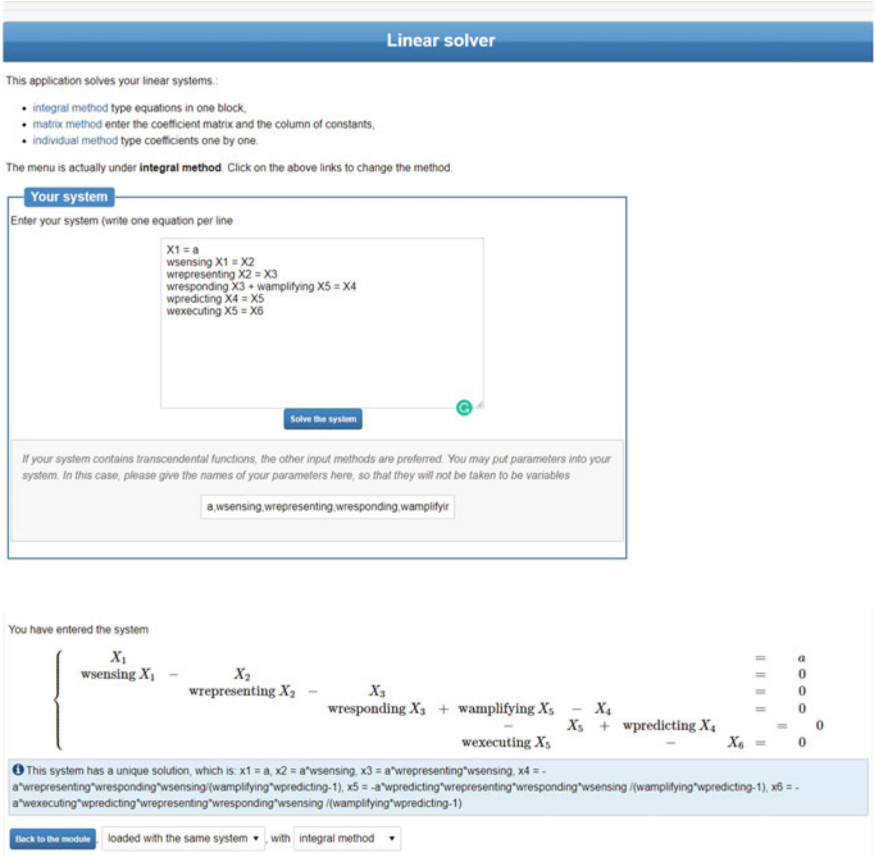


Fig. 18.5 Upper picture: the input interface of the Linear Solver after copying the linear equations in it. Lower picture, the provided output after running the solver; the solutions are in the shaded area

Table 18.4 Overview of the solutions for the linear equilibrium equations for the example of Figs. 18.3 and 18.4

State nr	State name	Solutions
X_1	ws_s	$X_1 = a$
X_2	ss_s	$X_2 = \boldsymbol{\omega}_{\text{sensing}}\ a$
X_3	srs_s	$X_3 = \boldsymbol{\omega}_{\text{representing}}\ \boldsymbol{\omega}_{\text{sensing}}\ a$
X_4	ps_a	$X_4 = \boldsymbol{\omega}_{\text{responding}}\ \boldsymbol{\omega}_{\text{representing}}\ \boldsymbol{\omega}_{\text{sensing}} a/(1 - \boldsymbol{\omega}_{\text{amplifying}}\ \boldsymbol{\omega}_{\text{predicting}})$
X_5	srs_e	$X_5 = \boldsymbol{\omega}_{\text{predicting}}\ \boldsymbol{\omega}_{\text{responding}}\ \boldsymbol{\omega}_{\text{representing}}\ \boldsymbol{\omega}_{\text{sensing}}\ a/(1 - \boldsymbol{\omega}_{\text{amplifying}}\ \boldsymbol{\omega}_{\text{predicting}})$
X_6	es_a	$X_6 = \boldsymbol{\omega}_{\text{executing}}\ \boldsymbol{\omega}_{\text{predicting}}\ \boldsymbol{\omega}_{\text{responding}}\ \boldsymbol{\omega}_{\text{representing}}\ \boldsymbol{\omega}_{\text{sensing}}\ a/(1 - \boldsymbol{\omega}_{\text{amplifying}}\ \boldsymbol{\omega}_{\text{predicting}})$

the connection weights have different values, also these equilibrium values will be different. Similarly, it can be seen that within the time period that the world state has value $a = 0$ all values go to 0, which is also indicated by the above expressions for the solutions when $a = 0$ is substituted.

18.6 Solving Nonlinear Equilibrium Equations for Euclidean Functions

It turns out that in certain cases a linear solver also can be used to solve nonlinear equilibrium equations. The idea is then to transform nonlinear equations in some way into linear equations that can be solved by a linear solver, after which the found solutions are transformed back. This will be illustrated in particular for equilibrium equations involving the euclidean combination function and the geometric combination function. In the current section, the euclidean function $\mathbf{eucl}_{\boldsymbol{n},\boldsymbol{\lambda}}$ is addressed, which is defined by

$$\mathbf{eucl}_{\boldsymbol{n},\boldsymbol{\lambda}}(V_1, \ldots, V_k) = \sqrt[n]{\frac{V_1^n + \ldots + V_k^n}{\boldsymbol{\lambda}}} \tag{18.9}$$

An equilibrium equation involving this combination function for a state $Y = X_j$ typically is of the form

$$X_j = \sqrt[n]{\frac{V_1^n + \ldots + V_k^n}{\boldsymbol{\lambda}}} \tag{18.10}$$

where $V_i = \boldsymbol{\omega}_{X_i,X_j} X_i(t)$ are single impacts
Assume all values are nonnegative. By applying the $\boldsymbol{n}^{th}$ power, this can be rewritten into

$$X_j^n = (\boldsymbol{\omega}_{X_1,X_j}^n X_1^n + \ldots + \boldsymbol{\omega}_{X_k,X_j}^n X_k^n / \boldsymbol{\lambda} \tag{18.11}$$

Now take $Y_i = X_i^n$ (with inverse relation $X_i = \sqrt[n]{Y_i}$) and rewrite the above equation into a linear equation in Y_i; this obtains

$$Y_j = (\boldsymbol{\omega}_{X_1,X_j}^n Y_1 + \ldots + \boldsymbol{\omega}_{X_k,X_j}^n Y_k)/\boldsymbol{\lambda}$$

$$\boldsymbol{\lambda} Y_j = \boldsymbol{\omega}_{X_1,X_j}^n Y_1 + \ldots + \boldsymbol{\omega}_{X_k,X_j}^n Y_k \tag{18.12}$$

For the example network model for $n = 2$; for state X_4 this is the following equation

$$\sqrt{((\boldsymbol{\omega}_{\text{responding}} X_3)^2 + (\boldsymbol{\omega}_{\text{amplifying}} X_5)^2)/\boldsymbol{\lambda}} = X_4 \tag{18.13}$$

By applying squares this can be rewritten as

$$((\boldsymbol{\omega}_{\text{responding}} X_3)^2 + (\boldsymbol{\omega}_{\text{amplifying}} X_5)^2)/\boldsymbol{\lambda} = X_4^2 \tag{18.14}$$

Then using variable $Y_i = {X_i}^2$ this becomes a linear equation in the Y_i:

$$(\boldsymbol{\omega}^2_{\text{responding}} Y_3 + \boldsymbol{\omega}^2_{\text{amplifying}} Y_5)/\boldsymbol{\lambda} = Y_4 \tag{18.15}$$

Similarly, the set of all equations becomes as shown in Table 18.5. This transforms the quadratic equations in the X_i into linear equations in Y_i. As in Sect. 18.5, these linear equations can be solved symbolically by the WIMS Linear Solver. In Fig. 18.6 it is shown how this set of equations is entered in this Linear Solver and (in the shaded lower area) what solutions are found. These solutions are (note that a is used as a parameter for an assumed stimulus level represented by X_1) translated back from the Y_i to the solutions in terms of the X_i as follows:

$$X_1^2 = a^2$$
$$X_2^2 = \boldsymbol{\omega}^2_{\text{sensing}} a^2$$
$$X_3^2 = \boldsymbol{\omega}^2_{\text{representing}} \boldsymbol{\omega}^2_{\text{sensing}} a^2$$
$$X_4^2 = \boldsymbol{\omega}^2_{\text{responding}} \boldsymbol{\omega}^2_{\text{representing}} \boldsymbol{\omega}^2_{\text{sensing}} a^2/(\boldsymbol{\lambda} - \boldsymbol{\omega}^2_{\text{amplifying}} \boldsymbol{\omega}^2_{\text{predicting}})$$
$$X_5^2 = \boldsymbol{\omega}^2_{\text{predicting}} \boldsymbol{\omega}^2_{\text{responding}} \boldsymbol{\omega}^2_{\text{representing}} \boldsymbol{\omega}^2_{\text{sensing}} a^2/(\boldsymbol{\lambda} - \boldsymbol{\omega}^2_{\text{amplifying}} \boldsymbol{\omega}^2_{\text{predicting}})$$
$$X_6^2 = \boldsymbol{\omega}^2_{\text{executing}} \boldsymbol{\omega}^2_{\text{predicting}} \boldsymbol{\omega}^2_{\text{responding}} \boldsymbol{\omega}^2_{\text{representing}} \boldsymbol{\omega}^2_{\text{sensing}} a^2/(\boldsymbol{\lambda} - \boldsymbol{\omega}^2_{\text{amplifying}} \boldsymbol{\omega}^2_{\text{predicting}})$$

Therefore, the solutions are obtained by applying the square root on these expressions (all is assumed nonnegative here); see Table 18.6. This provides explicit predictions for the equilibrium values that are reached.

In particular, for all connection weights 1 except $\boldsymbol{\omega}_{\text{responding}}$ and $\boldsymbol{\omega}_{\text{amplifying}}$ which are 0.5 and $\boldsymbol{\lambda} = 0.5$, the predicted values are $X_i = a$ for all i, which is confirmed by example simulations performed.

Table 18.5 Overview of linear equilibrium equations for the example of Fig. 18.3 with euclidean combination functions

State nr	State name	Equilibrium equation
Y_1	ws_s	$Y_1 = Y_1$
Y_2	ss_s	${\boldsymbol{\omega}_{\text{sensing}}}^2\ Y_1 = Y_2$
Y_3	srs_s	${\boldsymbol{\omega}_{\text{representing}}}^2\ Y_2 = Y_3$
Y_4	ps_a	${\boldsymbol{\omega}_{\text{responding}}}^2 Y_3 + {\boldsymbol{\omega}_{\text{amplifying}}}^2\ Y_5 = \boldsymbol{\lambda}\ Y_4$
Y_5	srs_e	${\boldsymbol{\omega}_{\text{predicting}}}^2\ Y_4 = Y_5$
Y_6	es_a	${\boldsymbol{\omega}_{\text{executing}}}^2\ Y_5 = Y_6$

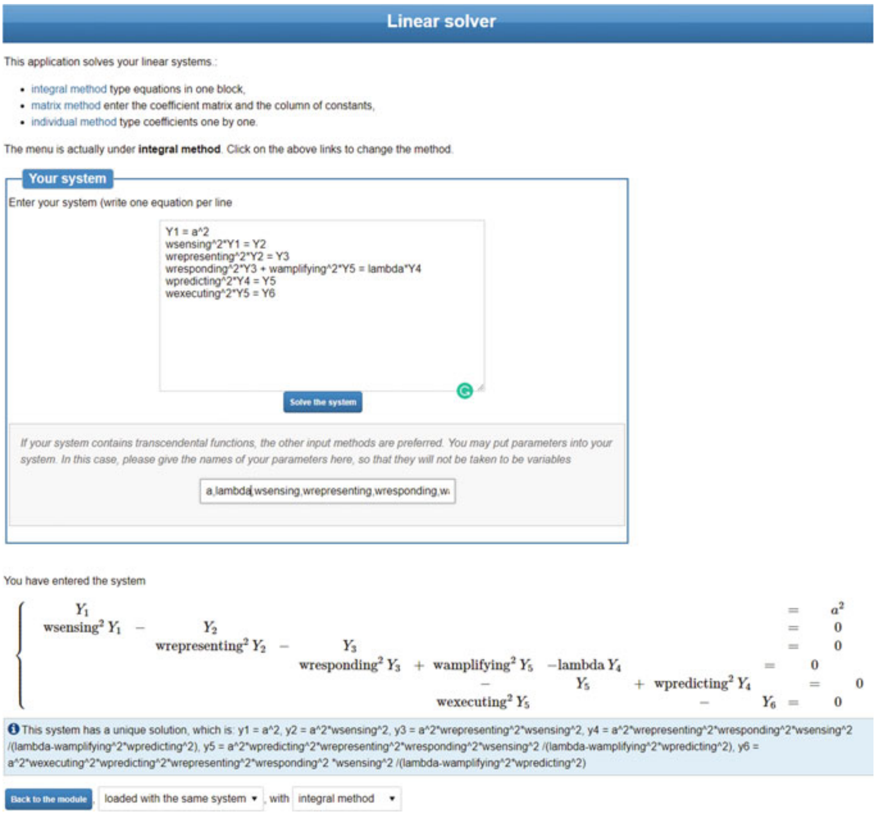


Fig. 18.6 Using the WIMS Linear Solver to solve the nonlinear equilibrium equations for Euclidean functions used for aggregation within the example network

Table 18.6 Overview of the solutions for the Euclidean equilibrium equations for the example of Figs. 18.3 and 18.4

State nr	State name	Solutions
X_1	ws_s	$X_1 = a$
X_2	ss_s	$X_2 = \omega_{\text{sensing}}\ a$
X_3	srs_s	$X_3 = \omega_{\text{representing}}\ \omega_{\text{sensing}}\ a$
X_4	ps_a	$X_4 = \omega_{\text{responding}}\ \omega_{\text{representing}}\ \omega_{\text{sensing}}\ a/\surd(\lambda - \omega_{\text{amplifying}}{}^2\ \omega_{\text{predicting}}{}^2)$
X_5	srs_e	$X_5 = \omega_{\text{predicting}}\ \omega_{\text{responding}}\ \omega_{\text{representing}}\ \omega_{\text{sensing}}\ a/\surd(\lambda - \omega^2{}_{\text{amplifying}}\ \omega^2{}_{\text{predicting}})$
X_6	es_a	$X_6 = \omega_{\text{executing}}\ \omega_{\text{predicting}}\ \omega_{\text{responding}}\ \omega_{\text{representing}}\ \omega_{\text{sensing}}\ a/\surd(\lambda - \omega_{\text{amplifying}}{}^2\ \omega_{\text{predicting}}{}^2)$

18.7 Solving Nonlinear Equilibrium Equations for Geometric Functions

In this section the geometric combination function **sgeomean**$_\lambda$ is addressed:

$$\mathbf{sgeomean}_{\boldsymbol{\lambda}}(V_1, \ldots, V_k) = \sqrt[k]{\frac{V_1 \ldots V_k}{\boldsymbol{\lambda}}} \tag{18.16}$$

An equilibrium equation involving this geometric combination function for state $Y = X_j$ typically is of the form

$$X_j = \sqrt[k]{\frac{V_1 \ldots V_k}{\boldsymbol{\lambda}}} \tag{18.17}$$

where $V_i = \boldsymbol{\omega}_{X_i,X_j} X_i(t)$ are single impacts. Assume all values are positive. By applying the k^{th} power and the natural log function, this can be rewritten into

$$\begin{aligned}
&X_j^k = (\boldsymbol{\omega}_{X_1,X_j} X_1 \ldots \boldsymbol{\omega}_{X_k,X_j} X_k)/\boldsymbol{\lambda}\\
&\boldsymbol{\lambda} X_j^k = \boldsymbol{\omega}_{X_1,X_j} X_1 \ldots \boldsymbol{\omega}_{X_k,X_j} X_k\\
&\log(\boldsymbol{\lambda} X_j^k) = \log(\boldsymbol{\omega}_{X_1,X_j} X_1 \ldots \boldsymbol{\omega}_{X_k,X_j} X_k)\\
&\log(\boldsymbol{\lambda}) + k\log(X_j) = \log(\boldsymbol{\omega}_{X_1,X_j}) + \log(X_1) + \ldots + \log(\boldsymbol{\omega}_{X_k,X_j}) + \log(X_k)
\end{aligned} \tag{18.18}$$

Now taking $Y_i = \log(X_i)$ (with inverse relation $X_j = \exp(Y_j)$) this can be rewritten into a linear equation in Y_i as follows

$$\begin{aligned}
\log(\boldsymbol{\lambda}) + kY_j &= \log(\boldsymbol{\omega}_{X_1,X_j}) + Y_1 + \ldots + \log(\boldsymbol{\omega}_{X_k,X_j}) + Y_k\\
Y_1 + \ldots + Y_k - kY_j &= \log(\boldsymbol{\lambda}) - (\log(\boldsymbol{\omega}_{X_1,X_j}) + \ldots + \log(\boldsymbol{\omega}_{X_k,X_j}))\\
Y_1 + \ldots + Y_k - kY_j &= \log(\boldsymbol{\lambda}/(\boldsymbol{\omega}_{X_1,X_j} \ldots \boldsymbol{\omega}_{X_k,X_j}))
\end{aligned} \tag{18.19}$$

As an illustration, assume in the example of Fig. 18.3 for ps$_a$ (which is X_4) the combination function **sgeomean**$_\lambda$ is used, with $k = 2$ and $\boldsymbol{\lambda} = 0.5$ and for the other states X_2, X_3, X_5, X_6 the function $\mathbf{sgeomean}_{\boldsymbol{\lambda}}(V_1)$ is used, with $\boldsymbol{\lambda} = 1$, which is the identity function. Then the equation for X_4 becomes

$$Y_3 + Y_5 - 2Y_4 = \log(\boldsymbol{\lambda}/(\boldsymbol{\omega}_{\text{responding}}\boldsymbol{\omega}_{\text{amplifying}})) \tag{18.20}$$

Using $Y_i = \log(X_i)$ for all states all equations are transformed into the set of linear equations shown in Table 18.7. Again applying the Linear Solver to them, as shown in Fig. 18.7, provides the following solutions:

Table 18.7 Overview of linear equilibrium equations for the example of Fig. 18.3 with geometric combination functions

State nr	State name	Equilibrium equation
Y_1	ws_s	$Y_1 = Y_1$
Y_2	ss_s	$\log(\boldsymbol{\omega}_{\text{sensing}}) + Y_1 = Y_2$
Y_3	srs_s	$\log(\boldsymbol{\omega}_{\text{representing}}) + Y_2 = Y_3$
Y_4	ps_a	$Y_3 + Y_5 - 2Y_4 = \log(\boldsymbol{\lambda}/(\boldsymbol{\omega}_{\text{responding}}\boldsymbol{\omega}_{\text{amplifying}}))$
Y_5	srs_e	$\log(\boldsymbol{\omega}_{\text{predicting}}) + Y_4 = Y_5$
Y_6	es_a	$\log(\boldsymbol{\omega}_{\text{executing}}) + Y_5 = Y_6$

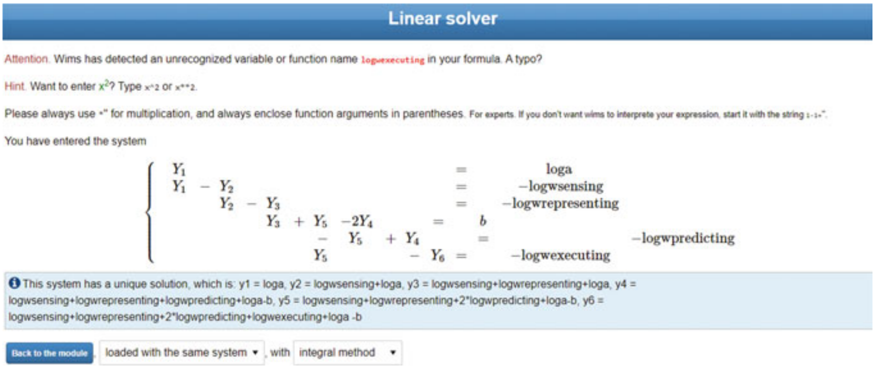


Fig. 18.7 Using the WIMS Linear Solver to solve the nonlinear equilibrium equations for weighted geometric functions used for aggregation within the example network

$$
\begin{aligned}
b &= \log(\boldsymbol{\lambda}/(\boldsymbol{\omega}_{\text{responding}}\boldsymbol{\omega}_{\text{amplifying}}))\\
\log(X_1) &= \log(a)\\
\log(X_2) &= \log(\boldsymbol{\omega}_{\text{sensing}}) + \log(a)\\
\log(X_3) &= \log(\boldsymbol{\omega}_{\text{representing}}) + \log(\boldsymbol{\omega}_{\text{sensing}}) + \log(a)\\
\log(X_4) &= \log(\boldsymbol{\omega}_{\text{responding}}) + \log(\boldsymbol{\omega}_{\text{representing}}) + \log(\boldsymbol{\omega}_{\text{sensing}}) + \log(a) - b\\
\log(X_5) &= 2\log(\boldsymbol{\omega}_{\text{predicting}}) + \log(\boldsymbol{\omega}_{\text{representing}}) + \log(\boldsymbol{\omega}_{\text{sensing}}) + \log(a) - b\\
\log(X_6) &= \log(\boldsymbol{\omega}_{\text{executing}}) + 2\log(\boldsymbol{\omega}_{\text{predicting}}) + \log(\boldsymbol{\omega}_{\text{representing}}) + \log(\boldsymbol{\omega}_{\text{sensing}})\\
&\quad + \log(a) - b
\end{aligned}
$$

Therefore, the solutions for the X_i are obtained by substituting

$$\log(\boldsymbol{\lambda}/(\boldsymbol{\omega}_{\text{responding}}\boldsymbol{\omega}_{\text{amplifying}}))$$

For b and applying the standard exponential function; see Table 18.8.

Table 18.8 Overview of the solutions for the geometric equilibrium equations for the example of Figs. 18.3 and 18.4

State nr	State name	Solutions
X_1	ws$_s$	$X_1 = a$
X_2	ss$_s$	$X_2 = \boldsymbol{\omega}_{\text{sensing}}\ a$
X_3	srs$_s$	$X_3 = \boldsymbol{\omega}_{\text{representing}}\boldsymbol{\omega}_{\text{sensing}}a$
X_4	ps$_a$	$X_4 = \boldsymbol{\omega}_{\text{responding}}\boldsymbol{\omega}_{\text{representing}}\boldsymbol{\omega}_{\text{sensing}}\boldsymbol{\omega}_{\text{responding}}\boldsymbol{\omega}_{\text{amplifying}}\ a/\boldsymbol{\lambda}$
X_5	srs$_e$	$X_5 = \boldsymbol{\omega}_{\text{predicting}}{}^2\ \boldsymbol{\omega}_{\text{representing}}\boldsymbol{\omega}_{\text{sensing}}\ \boldsymbol{\omega}_{\text{responding}}\boldsymbol{\omega}_{\text{amplifying}}\ a/\boldsymbol{\lambda}$
X_6	es$_a$	$X_6 = \boldsymbol{\omega}_{\text{executing}}\boldsymbol{\omega}_{\text{predicting}}{}^2\ \boldsymbol{\omega}_{\text{representing}}\boldsymbol{\omega}_{\text{sensing}}\ \boldsymbol{\omega}_{\text{responding}}\boldsymbol{\omega}_{\text{amplifying}}\ a/\boldsymbol{\lambda}$

Substituting 0.5 for $\boldsymbol{\omega}_{\text{responding}}$, $\boldsymbol{\omega}_{\text{amplifying}}$, and $\boldsymbol{\lambda}$, and 1 for the other connection weights again provides $X_i = a$ for all i, which again is confirmed by example simulations.

18.8 Solving Nonlinear Equilibrium Equations for Examples of Self-Model States

Note that as self-modeling networks are networks, in principle the methods and concepts from this entire chapter apply to them too. This will be illustrated in the current section, in particular for the method of solving stationary point or equilibrium equations applied to self-model states. In Sects. 18.6 and 18.7 examples of nonlinear functions were addressed by transforming their equilibrium equations into linear equations. Such a transformation is not always possible. Nevertheless, in some cases nonlinear equations can still be solved without being able to transform them into linear equations. In this section two of such cases are discussed, applied to self-model states for adaptive connections.

18.8.1 Solving Nonlinear Equations for Self-Model States for Hebbian Learning

Hebbian learning is often summarised in a simplified form as.

> Neurons that fire together, wire together.
>
> (Hebb 1949; Keysers and Gazzola 2014; Shatz 1992)

In relation to this, for Hebbian learning the combination function **hebb**$_{\boldsymbol{\mu}}$**(..)** is often used; it is defined by

$$\mathbf{hebb}_{\boldsymbol{\mu}}(V_1, V_2, W) = V_1 V_2 (1 - W) + \boldsymbol{\mu} W \qquad (18.21)$$

Here $\boldsymbol{\mu}$ is the persistence parameter, V_1 stands for $X_i(t)$, V_2 for $Y(t)$ and W for the self-model state $\mathbf{W}_{X_i,Y}(t)$ for connection weight $\boldsymbol{\omega}_{X_i,Y}(t)$. The parameter persistence parameter describes in how far a learnt connection persists over time. Full persistence is indicated by $\boldsymbol{\mu} = 1$ and full extinction takes place for $\boldsymbol{\mu} = 0$. If $0 < \boldsymbol{\mu} < 1$, then a fraction $\boldsymbol{\mu}$ of what was learned persists and a fraction 1-$\boldsymbol{\mu}$ of extinction takes place per time unit. For example, if $\boldsymbol{\mu} = 0.95$, then 5% of the learned value is lost per time unit. In the first part of the formula, the expression V_1V_2 models the condition 'neurons that fire together', and the factor (1-W) takes care that the connection weight stays in the [0, 1] interval.

In Box 18.1 it is shown how a stationary point equation for this combination function can be analysed by expressing W in terms of V_1, V_2 and $\boldsymbol{\mu}$. For example, it is found that when $\boldsymbol{\mu} < 1$, in an equilibrium state the value of W is always < 1. For more options for Hebbian learning functions and their analysis, see (Treur 2020a), Chap. 14.

Box 18.1 Analysis of the stationary point equation for a Hebbian learning function.

For a stationary point, applying criterion (2) provides the following stationary point equation for the above Hebbian learning combination function:

$$W = \mathbf{hebb}_{\boldsymbol{\mu}}(V_1, V_2, W) = V_1V_2(1 - W) + \boldsymbol{\mu}W \quad \Leftrightarrow$$
$$W = V_1V_2 - V_1V_2W + \boldsymbol{\mu}W \quad \Leftrightarrow$$
$$W(1 + V_1V_2 - \boldsymbol{\mu}) = V_1V_2 \quad \Leftrightarrow$$
$$W = \frac{V_1V_2}{1 - \boldsymbol{\mu} + V_1V_2}$$

For example

- When $\boldsymbol{\mu} = 1$ (no extinction) and V_1 and V_2 are nonzero, then $W = 1$
- When both V_1 and V_2 have value 1, then $W = \frac{1}{2-\boldsymbol{\mu}}$.

18.8.2 Solving the Nonlinear Equations for Self-Model States for Bonding by Homophily

Bonding by homophily is often summarised as

> Birds of a feather flock together.
>
> (McPherson et al. 2001)

For the bonding by homophily principle from (Treur 2016), Chapter 11, an option for the combination function is the *simple linear homophily function* $\mathbf{slhomo}_{\boldsymbol{\sigma},\boldsymbol{\tau}}(..)$:

$$\mathbf{slhomo}_{\boldsymbol{\sigma},\boldsymbol{\tau}}(V_1, V_2, W) = W + \boldsymbol{\alpha}(\boldsymbol{\tau} - |V_1 - V_2|)(1 - W)W \tag{18.22}$$

Here $\boldsymbol{\alpha}$ is the homophily modulation factor, and $\boldsymbol{\tau}$ the tipping point. The part $(\boldsymbol{\tau} - |V_1 - V_2|)$ models the condition 'birds of a feather': this part is positive if the difference between V_1 and V_2 is less than the tipping point $\boldsymbol{\tau}$ ('birds of a feather' holds) and negative when this difference is more than $\boldsymbol{\tau}$ ('birds of a feather' does not hold). The factor $(1 - W)W$ takes care that W stays within the [0, 1] interval. As long as W is not 0 or 1, in the first case by the combination function a positive term is added to W, so the connection weight will increase; in the second case a negative term is added, so the connection weight will decrease.

In Box 18.2 it is shown how a stationary point equation for this combination function can be analysed. It is found that usually a form of clustering takes place. In (Treur 2020a), Chapter 13 more options for functions for bonding by homophily are analysed.

Box 18.2 Analysis of the stationary point equation for a simple linear function for bonding by homophily.

For the combination function for bonding based on homophily case, for a stationary point applying the criterion (2) provides the following stationary point equation (assuming $\boldsymbol{\alpha} > 0$):

$$
\begin{aligned}
&W = \mathbf{slhomo}_{\sigma,\tau}(V_1, V_2, W) = W + \boldsymbol{\alpha}(\boldsymbol{\tau} - |V_1 - V_2|)(1 - W)W \quad \Leftrightarrow \\
&\boldsymbol{\alpha}(\boldsymbol{\tau} - |V_1 - V_2|)(1 - W)W = 0 \quad \Leftrightarrow \\
&W = 0 \quad \text{or} \quad W = 1 \quad \text{or} \quad |V_1 - V_2| = \boldsymbol{\tau}
\end{aligned}
$$

As in simulations the third option here often turns out to be not attracting, this shows that in an equilibrium a form of clustering is achieved with connection weights 1 between states within one cluster and connection weights 0 between states in different clusters.

18.9 General Equilibrium Analysis for a Class of Nonlinear Functions

In this section the analysis is addressed not at the level of specific network structures and implied within-network dynamics but at a more abstract level of properties of network structures and properties of within-network dynamics implied by them; see Fig. 18.8.

In Sects. 18.7 and 18.8 a specific approach was followed that for specific combination functions chosen, obtains detailed formulae for the predicted equilibrium values in terms of the network characteristics. In contrast, in the current section a general perspective is followed and theorems are discussed for a large class of nonlinear and

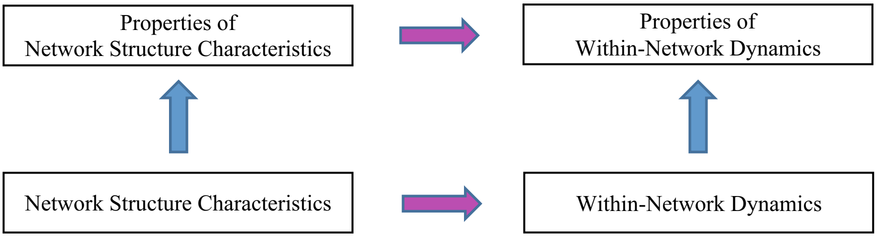
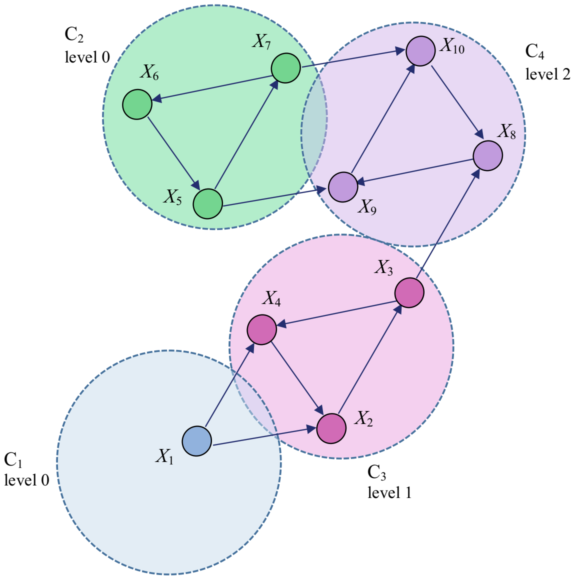


Fig. 18.8 Relating properties of within-network dynamics to properties of network structure

linear functions that have been found (Treur 2020b). This general perspective makes use of the notion of *(strongly connected) component* of a network; this is a maximal subnetwork *C* such that every node within *C* can be reached from every other node via a path following the direction of the connections; e.g., (Bloem et al. 2006; Harary et al. 1965; Łacki 2013; Wijs et al. 2016). These components form a partition of the set of nodes of the network. This is illustrated in Fig. 18.9 by an example network borrowed from (Treur 2020a), where the components are C_1 to C_4.

As another illustration, consider the example of a mental network model with connectivity depicted in Fig. 18.3. In Fig. 18.10, the components of this network are shown: from left to right components C_1 to C_5. In (Treur 2020b) the notion of *stratification* was introduced for such a partition of a network so that each component gets a level (or stratum) assigned. In this case, following the components in the direction of the connections the levels are 0 to 4 as indicated in Fig. 18.10 from left to right. Note that in the general case, by the stratification multiple components can

Fig. 18.9 Strongly connected components for the example network model from (Treur 2020a) and their stratification levels



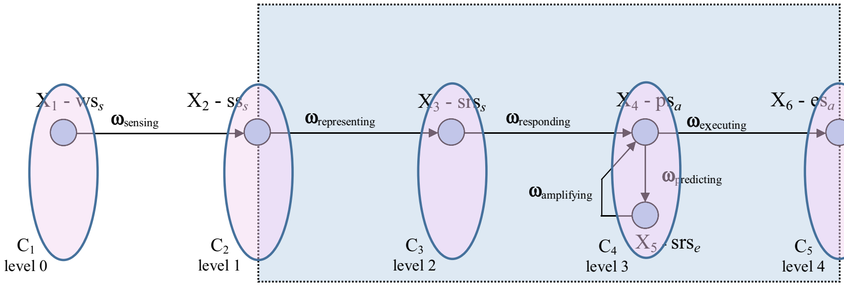


Fig. 18.10 Strongly connected components for the example network model of Fig. 18.3 and their stratification levels

get the same level. For example, in Fig. 18.9 there are two different components C_1 and C_2 that each has no incoming connection, then both components are at level 0; see also (Treur 2020b). In Fig. 18.9, component C_3 is at level 1 and component C_4 at level 2 (because C_4 has an incoming connection from a level 1 component, namely C_3).

Based on the levels defined by this notion of stratification, a number of general theorems and corollaria have been found and proven as presented in (Treur 2020a); see also (Treur 2020b), Chaps. 12 and 15. For aggregation, these results are not limited to linear functions and for connectivity no condition at all is demanded; some of the main results are the following. The definitions are as follows.

Definition (weakly scalar-free and scalar-free functions) Consider functions f: $R^k \to \mathbb{R}$ and θ: $R \to \mathbb{R}$ for some subset $R \subseteq \mathbb{R}$ which is $\mathbb{R}$ or $\mathbb{R}_{>0}$.

(a) A function f: $R^k \to \mathbb{R}$ is called *weakly scalar-free* for function θ if for all V_1, …, $V_k \in R$ and all $\alpha \in R$ it holds

$$f(\alpha V_1, \ldots, \alpha V_k) = \theta(\alpha) f(V_1, \ldots, V_k)$$

(b) A function f: $R^k \to \mathbb{R}$ is called *scalar-free* if for all V_1, …, $V_k \in R$ and all $\alpha \in R$ it holds

$$f(\alpha V_1, \ldots, \alpha V_k) = \alpha f(V_1, \ldots, V_k)$$

(c) A function f: $R^k \to \mathbb{R}$ is called *strictly (monotonically) increasing* if for all U_1, …, U_k, V_1, …, $V_k \in R$ such that $U_i \leq V_i$ for all i and $U_j < V_j$ for at least one j it holds

$$f(U_1, \ldots, U_k) < f(V_1, \ldots, V_k)$$

(d) A function f: $R^k \to \mathbb{R}$ used as combination function for a state Y in a network is called *normalised* if $f(\omega_1, \ldots, \omega_k) = 1$, where the ω's are the weights of the incoming connections of Y in the network.

Theorem 1 (Equilibrium values related to strongly connected components) If the following aggregation conditions are fulfilled

- The combination functions are normalised, scalar-free and strictly increasing

then in an achieved equilibrium:

(a) In any level 0 component C

- All states have the same equilibrium value V
- This V is between the highest and lowest initial value of states within C

(b) If for any level $i > 0$ component C the components C_1,.., C_k are the strongly connected components from which C gets an incoming connection, then

- If all states in C_1,.., C_k have the same equilibrium value V, then also all states in C have this same equilibrium value V
- The equilibrium values of the states in C are between the highest and lowest equilibrium values of the states in C_1,.., C_k

The following corollary can be immediately derived from Theorem 1.

Corollary 1 (dependence of all equilibrium values on the values in level 0 components)
If the following aggregation conditions are fulfilled

- The combination functions are normalised, scalar-free and strictly increasing

then in an achieved equilibrium:

(a) If all states in all level 0 components C have the same equilibrium value V, then all states of the whole network have that same equilibrium value V
(b) The equilibrium values of all states in the network

- are between the highest and lowest equilibrium values of the states in the level 0 components
- are between the highest and lowest initial values of the states in the level 0 components

For a strongly connected network (consisting of only one component, which then is a level 0 component), the following is obtained as a special case of the above:

Corollary 2 (strongly connected networks)
If the following connectivity and aggregation conditions are fulfilled

- The network is strongly connected
- The combination functions are normalised, scalar-free and strictly increasing

then in an achieved equilibrium:

- All states have the same equilibrium value V

- This equilibrium value V is between the highest and lowest initial values of the states

Given that in the example network model there is only one level 0 component with constant value a, by Theorem 1 or Corollary 1 above it can be concluded that all states will have equilibrium value a, as long as the aggregation conditions are fulfilled.

In this section a specific class of linear and nonlinear functions has been identified (strictly increasing, scalar-free, and normalised) which all share a similar mathematical analysis for equilibria of network models using such functions. It is sometimes believed that for dynamical models the borderline between linear and nonlinear functions is also the borderline between well-analyzable behavior and less well-analyzable behavior. In contrast to this, for contagion in social networks it has been found here that this borderline between well-analyzable behavior and less well-analyzable behavior lies somewhere within the domain of nonlinear functions: it is not between linear and nonlinear functions but between one class (of strictly increasing, scalar-free, normalised functions) covering both linear and nonlinear functions and another subclass of the class of nonlinear functions not satisfying these conditions. In the next two sections this class of functions is explored in some more mathematical detail.

18.10 Additive, Multiplicative, Log-like and Exp-like Functions

In this section a few basic types of functions needed in Sect. 18.11 and further are briefly reviewed. Proofs can be found in the Appendix. Below, the subset $R \subseteq \mathbb{R}$ used as domain for the considered functions θ in principle will be $\mathbb{R}$ or an interval within $\mathbb{R}$ of the form $\mathbb{R}_{>0} = (0, \infty)$, although in some cases also other intervals may be considered. Note that the symbol $\circ$ is used to denote function composition ($g \circ f$ is read for functions f and g as 'g over f' or 'g on f'). Sometimes it is left out: gf means $g \circ f$. The domain of a function f is denoted by Dom(f) and the range f(Dom(f)) by Range(f).

Definition (additive, multiplicative, log-like, exp-like)

(a) A function θ: $R \to \mathbb{R}$ is called *additive* if $\theta(\alpha + \beta) = \theta(\alpha) + \theta(\beta)$ for all $\alpha, \beta \in R$
(b) A function θ: $R \to \mathbb{R}$ is called *multiplicative* if $\theta(\alpha\beta) = \theta(\alpha)\theta(\beta)$ for all $\alpha, \beta \in R$
(c) A function θ: $R \to \mathbb{R}$ is called *log-like* if $\theta(\alpha\beta) = \theta(\alpha)+\theta(\beta)$ for all $\alpha, \beta \in R$
(d) A function θ: $R \to \mathbb{R}$ is called *exp-like* if $\theta(\alpha+\beta) = \theta(\alpha)\theta(\beta)$ for all $\alpha, \beta \in R$
(e) The standard (natural, based on the number e) exponential and logarithmic functions will be denoted by exp and log, respectively.

Note that multiplicative and log-like functions are typically used for domains R that are closed under multiplication and division such as $R = \mathbb{R}_{>0}$, whereas additive and exp-like functions are typically used for domains R that are closed under addition and subtraction such as $R = \mathbb{R}$.

Proposition 1 (relating additive, multiplicative, log-like, and exp-like functions) Let θ: $R \rightarrow S$ be any function for a finite or infinite interval R in $\mathbb{R}$, then the following hold:

(a) If θ is multiplicative and $S \subseteq \mathbb{R}_{>0}$, then log $\circ$ θ is log-like.
(b) If $R \subseteq \mathbb{R}_{>0}$ and θ is log-like, then θ $\circ$ exp is additive.
(c) If θ is exp-like, then log $\circ$ θ is additive.
(d) If θ is multiplicative and $S \subseteq \mathbb{R}_{>0}$, then log $\circ$ θ $\circ$ exp is additive.
(e) For any multiplicative function such that $\theta(\alpha) = 0$ for some $\alpha \neq 0$, it holds that $\theta(\alpha) = 0$ for all α. For any nonzero multiplicative function θ it holds $\theta(1) = 1$ and $\theta(\alpha^{-1}) = \theta(\alpha)^{-1}$ for all α.
(f) If a multiplicative θ is injective on Dom(θ), then it has an inverse θ^{-1} with Dom(θ^{-1}) = Range(θ) and Range(θ^{-1}) = Dom(θ); this inverse θ^{-1} is also multiplicative.

The following theorem provides simple characterisations of the different types of functions defined above.

Theorem 2 (characterisation of additive, multiplicative, log-like and exp-like) Let θ: R $\rightarrow \mathbb{R}$ be continuous. Then the following hold.

(a) Assume $R \subseteq \mathbb{R}$ is closed under addition and subtraction with $1 \in R$, then it holds:
θ is additive $\Leftrightarrow$ for some c $\in \mathbb{R}$ for all X it holds $\theta(X) = c\,X$.

(b) Assume $R \subseteq \mathbb{R}_{>0}$ is closed under multiplication and division with e $\in R$, then it holds:
θ is multiplicative $\Leftrightarrow$ for some c $\in \mathbb{R}$ for all X it holds $\theta(X) = X^c$.

(c) Assume $R \subseteq \mathbb{R}_{>0}$ is closed under multiplication and division with e $\in R$, then it holds:
θ is log-like $\Leftrightarrow$ for some c $\in \mathbb{R}$ for all X it holds $\theta(X) = c \log(X)$.

(d) Assume $R = \mathbb{R}$ is closed under addition and subtraction with $1 \in R$, then it holds:
θ is exp-like $\Leftrightarrow$for some c $\in \mathbb{R}$ for all X it holds $\theta(X) = \exp(cX)$.

18.11 Weakly Scalar-Free and Scalar-Free Functions

Whether or not combination functions are scalar-free is an important factor determining whether or not by social contagion all members of a well-connected social network converge to the same level of emotion, opinion, information, belief, intention, or any other mental or physical state; e.g., (Treur 2020a) and (Treur 2020b), Chaps. 11 and 12. The class of scalar-free functions includes all linear functions but goes far beyond the class of linear functions: a number of types of nonlinear functions are known that are scalar-free, such as the Euclidean functions and geometric mean functions (as will be defined below). However, it is not exactly clear how far exactly this class of scalar-free functions reaches. To get some more insight in this, in this section some further analysis is made of scalar-free functions, thereby also using a weakened variant of them called weakly scalar-free functions.

The following basic properties can easily be verified:

Proposition 2 (scalar-free and strictly increasing functions)

(a) Any function composition of scalar-free functions is scalar-free
(b) Any function composition of strictly increasing functions is strictly increasing
(c) All linear functions with positive coefficients are scalar-free and strictly increasing
(d) Any scalar-free function f is weakly scalar-free for $\theta = \mathrm{id}$, the identity function.

Examples (weakly scalar-free functions) There are many examples of weakly scalar-free functions. For example, the following functions on proper domains are weakly scalar-free

$$\begin{aligned} f(V) &= V^k \\ f(V_1, \ldots, V_k) &= V_1 * \ldots * V_k \end{aligned} \tag{18.23}$$

These are both weakly scalar-free with function

$$\theta(\alpha) = \alpha^k \tag{18.24}$$

The example

$$f(V_1, V_2, V_3) = w_1 V_1 V_2 + w_2 V_2 V_3 + w_3 V_3 V_1 \tag{18.25}$$

is weakly scalar-free with function

$$\theta(\alpha) = \alpha^2 \tag{18.26}$$

Definition (Cartesian product function) For functions $\theta_1,..,\theta_k: R \to R$ their *cartesian product function.*

$\mathrm{X}_{i=1}^{k}\theta_i : R^k \to R^k$ is defined by

$$\mathrm{X}_{i=1}^{k}\theta_i(V_1, \ldots, V_k) = (\theta_1(V_1), \ldots, \theta_k(V_k))$$

When all θ_i are equal to one θ, this cartesian product function $\mathrm{X}_{i=1}{}^{k}\theta_i$ is also denoted by $\mathrm{X}^k\theta$, and then is also called a *cartesian power function* of θ.

The following theorem describes some properties of scalar-free and weakly scalar-free functions. Again, proofs can be found in the Appendix.

Theorem 3 (relating weakly scalar-free and scalar-free functions)
Consider functions f: $R^k \to \mathbb{R}$ and θ: $R \to \mathbb{R}$ for some subset $R \subseteq \mathbb{R}$ which is $\mathbb{R}$ or $\mathbb{R}_{>0}$.

(a) If a nonzero function f is weakly scalar-free for function θ, then θ is multiplicative.

If, moreover, f is (strictly) monotonically increasing and has at least one positive value, then θ is also (strictly) monotonically increasing.

Therefore for the strict monotonically increasing case, θ is injective and has an inverse θ^{-1} on Range(θ), which is also multiplicative.

(b) Any nonzero multiplicative function θ is weakly scalar-free for itself.
(c) For any weakly scalar-free function f for θ the following are equivalent:
   (i) Range(f) $\subseteq$ Range(θ)
   (ii) For all $V_1, \ldots, V_k$ an α exists such that $f(\alpha V_1, \ldots, \alpha V_k) = 1$
(d) For each weakly scalar-free function f: $R^k \to \mathbb{R}$ for any injective θ, the function g: Range$(\theta)^k \to \mathbb{R}$ defined by $g = f\, \mathrm{X}^k\theta^{-1}$ is scalar-free. If, moreover, Range(f) $\subseteq$ Range(θ), then also the function h: $R^k \to \mathbb{R}$ defined by $h = \theta^{-1}f$ is scalar-free. For strictly increasing f and θ, these functions g,h are strictly increasing too.
(e) For each set of strictly increasing and weakly scalar-free functions f_i: $R^k \to \mathbb{R}_{\geq 0}$ for the same strictly increasing θ, for any linear combination f of the f_i with positive coefficients, the function g: $R^k \to \mathbb{R}$ defined by $g = f\, \mathrm{X}^k\theta^{-1}$ is strictly increasing and scalar-free. If, moreover, Range(f) $\subseteq$ Range(θ), then also the function h: $R^k \to \mathbb{R}$ defined by $h = \theta^{-1}f$ is strictly increasing and scalar-free.
(f) If f: $R^k \to \mathbb{R}$ is scalar-free, θ: $R \to R$ is multiplicative and $g = f \circ \mathrm{X}^k\theta$: $R^k \to \mathbb{R}$, then g is weakly scalar-free for θ. This holds in particular if f is linear.

Examples (from weakly scalar-free to scalar-free functions) From Theorem 3d it follows that for the function f of (18.25) and θ of (18.26) the function

$$g(V_1, V_2, V_3) = \theta^{-1} f(V_1, V_2, V_3) = \sqrt{w_1 V_1 * V_2 + w_2 V_2 * V_3 + w_3 V_3 * V_1} \tag{18.27}$$

is scalar-free. Also, by Theorem 3e the function

$$h(V_1, V_2, V_3) = \left[V_1 + V_2 + V_3 + \sqrt{(w_1 V_1 * V_2 + w_2 V_2 * V_3 + w_3 V_3 * V_1)}\right]/\boldsymbol{\lambda} \tag{18.28}$$

is scalar-free.

18.12 Scalar-Free Functions based on Function Conjugates

In this section it is analysed how from given scalar-free functions other scalar-free functions can be obtained by applying some transformation. The type of transformation applied can be interpreted as a form of scale transformation or coordinate transformation. It is done by generating conjugates of scalar-free functions defined as follows. This perspective provides an explanation of what was achieved in Sects. 18.6 and 18.7 where nonlinear equations were transformed into linear equations.

Definition (function conjugates). Let subsets R, $S \subseteq \mathbb{R}$ be given. The function g: $S^k \rightarrow S$ is a *(function) conjugate* of f: $R^k \rightarrow R$ by θ if θ: $S \rightarrow R$ is a bijective function and

$$g = \theta^{-1} \circ f \circ \mathrm{X}^k\theta$$

The following proposition describes in some more depth how a function f and its conjugates g relate to each other.

Proposition 3 (function conjugate operator) Let subsets R, $S \subseteq \mathbb{R}$ be given, and functions g: $S^k \rightarrow S$, f: $R^k \rightarrow R$, and bijective θ: $S \rightarrow R$. Then the following hold:

(a) Then the following are equivalent:

  (i) g is a function conjugate of f by θ
  (ii) The following commutation rules for θ, f and g hold:

$$\theta g = f\mathrm{X}^k\theta$$
$$\theta^{-1} f = g\mathrm{X}^k\theta^{-1}$$

(b) If a)(i) and (ii) hold, then for any g such an f is unique and can be denoted by $f = S_\theta(g)$ for a *function conjugate operator* S_θ; similarly, $g = S_{\theta^{-1}}(f)$ for function conjugate operator $S_{\theta^{-1}}$, so it holds:

$$\theta g = S_\theta(g)\mathrm{X}^k\theta$$
$$\theta^{-1} f = S_{\theta^{-1}}(f)\mathrm{X}^k\theta^{-1}$$

These operators S_θ and $S_{\theta^{-1}}$ are each other's inverse and they preserve function addition and composition: for all functions f, g, f_1, f_2, g_1 and g_2 of proper types it holds

$$S_{\theta^{-1}}S_\theta(g) = g$$
$$S_\theta S_{\theta^{-1}}(f) = f$$
$$S_\theta(g_1 + g_2) = S_\theta(g_1) + S_\theta(g_2)$$
$$S_\theta(g_1 \circ g_2) = S_\theta(g_1) \circ S_\theta(g_2)$$
$$S_{\theta^{-1}}(f_1 + f_2) = S_{\theta^{-1}}(f_1) + S_{\theta^{-1}}(f_2)$$
$$S_{\theta^{-1}}(f_1 \circ f_2) = S_{\theta^{-1}}(f_1) \circ S_{\theta^{-1}}(f_2)$$

Moreover, when conjugate operators S_{θ_1} and S_{θ_2} for θ_1 and θ_2 are applied in turn, it holds

$S_{\theta_1\theta_2}(g) = S_{\theta_1}S_{\theta_2}(g)S_{\theta_1}(\mathrm{X}^k\theta_2)\mathrm{X}^k\theta_1\mathrm{X}^k\theta_2^{-1}\mathrm{X}^k\theta_1^{-1}$ for all g

If in addition, θ_1 and θ_2 commute (i.e., $\theta_1\theta_2 = \theta_2\theta_1$), then $S_{\theta_1}S_{\theta_2} = S_{\theta_1\theta_2}$:

$S_{\theta_1}S_{\theta_2}(g) = S_{\theta_1\theta_2}(g)$ for all g

The condition that θ_1 and θ_2 commute is always fullfilled when θ_1 and θ_2 are both multiplicative, both additive, or both log-like or exp-like.

Definition (weighted Euclidean and geometric functions)

(a) A function g is a *weighted euclidean function* of order n if

$$g(V_1, \ldots, V_k) = \sqrt[n]{w_1 V_1^n + \ldots + w_k V_k^n}$$

For some weights w_1,.., w_k. A weighted euclidean function is *normalised* if *g(V, ..., V)* = *V* for all *V*, i.e., if the sum of its weights is 1, in which case it is called a *weighted euclidean average function.* A weighted euclidean function of order $n = 1$ is called a *linear function.*

(b) A function g is a *weighted geometric function* if

$$g(V_1, \ldots, V_k) = V_1^{w_1} \ldots V_k^{w_k}$$

For some weights w_1,.., w_k. A weighted geometric function is *normalised* if *g(V, ..., V)* = *V* for all *V*, i.e., if the sum of its weights is 1, in which case it is called a *weighted geometric mean function.*

In this section, it is established that the above-defined nonlinear functions are scalar-free. Note that the scaled euclidean function $\mathbf{eucl}_{n,\lambda}$ of order n is a special case of a weighted euclidean function. Moreover, the scaled geometric function $\mathbf{sgeomean}_1$ is a special case of a weighted geometric function with all weights 1, whereas $\mathbf{sgeomean}_\lambda$ for $\lambda \neq 1$ is not a weighted geometric function itself but $\mathbf{sgeomean}_\lambda$ is a constant factor $c = 1/\sqrt[k]{\lambda}$ times the weighted geometric function $\mathbf{sgeomean}_1$.

First, in a more general setting this will be addressed for weighted euclidean functions. Moreover, it is analysed how weighted euclidean functions can be related to linear functions: it turns out that they can be interpreted as conjugates of linear functions via some multiplicative function θ. This is explained by the following theorem.

Theorem 4 (from scalar-free functions to scalar-free conjugates by multiplicative θ)

(a) For any scalar-free function f: $R^k \rightarrow R$ with $R = \mathbb{R}_{\geq 0}$, all of its conjugates $\theta^{-1} \circ f \circ \mathrm{X}^k\theta$ by a multiplicative θ: $R \rightarrow R$ are also scalar-free.

(b) More specifically, for any scalar-free function f, for any positive real number n the function g defined by

$$g(V_1, \ldots, V_k) = \sqrt[n]{f(V_1^n, \ldots, V_k^n)}$$

is a conjugate $\theta^{-1} \circ f \circ \mathrm{X}^k\theta$ of f by the multiplicative function θ: $X \rightarrow X^n$ and therefore is also scalar-free.

(c) All weighted euclidean functions are conjugates of linear functions by a multiplicative function θ and therefore are scalar-free. In particular, this holds for all functions $\mathbf{eucl}_{n,\boldsymbol{\lambda}}$.

As an illustration of Theorem 4, the function $\mathbf{eucl}_{n,\boldsymbol{\lambda}}$ can be written as a conjugate function of a linear function as follows:

$$\mathbf{eucl}_{n,\boldsymbol{\lambda}} = \theta^{-1} \circ f \circ \mathrm{X}^k\theta \tag{18.29}$$

where $\theta(X) = X^n$ and

$$f(V_1, \ldots, V_k) = (V_1 + \ldots + V_k)/\boldsymbol{\lambda} \tag{18.30}$$

This can be verified as follows:

$$\begin{aligned}\theta^{-1} \circ f \circ \mathrm{X}^k\theta(V_1, \ldots, V_k) &= \theta^{-1} \circ f(V_1^n, \ldots, V_k^n) \\ &= \theta^{-1}((V_1^n + \ldots + V_k^n)/\boldsymbol{\lambda}) \\ &= \sqrt[n]{(V_1^n + \ldots + V_k^n)/\boldsymbol{\lambda}}\end{aligned}$$

This describes the function transformation underlying the equation transformation that has been used in Sect. 18.6 to transform the nonlinear euclidean equilibrium equations into linear equations.

Next, in a more general setting in Theorem 5 it is established that weighted geometric functions are scalar-free and how they can be related to linear functions. Again, it turns out that they can be considered conjugates of linear functions, this time not via a multiplicative function but via a log-like function θ. This is explained by the following:

Theorem 5 (from linear to scalar-free conjugates by log-like θ)

(a) For any linear function with sum of coefficients 1 all of its conjugates $\theta^{-1} \circ f \circ \mathrm{X}^k\theta$ by a log-like θ are scalar-free.

(b) More specifically, for any linear function f with sum of coefficients 1, the function g defined by

$$g(V_1, \ldots, V_k) = \exp(f(\log(V_1), \ldots, \log(V_k)))$$

is a conjugate $\theta^{-1} \circ f \circ \mathrm{X}^k\theta$ of a linear function by the standard log-like function $\theta = \log$ and therefore is scalar-free.

(c) All weighted geometric functions are conjugates of a linear function by a log-like function θ and therefore are scalar-free. In particular, this also holds for all functions $\mathbf{sgeomean}_{\boldsymbol{\lambda}}$.

As an illustration of Theorem 4, the function $\mathbf{sgeomean}_1$ can be written as a conjugate function of a linear function as follows:

$$\mathbf{sgeomean}_1 = \theta^{-1} \circ f \circ \mathrm{X}^k\theta \tag{18.31}$$

Holds for $\theta = \log$ and

$$f(V_1, \ldots, V_k) = (V_1 + \ldots + V_k)/k \tag{18.32}$$

This can be verified as follows:

$$\begin{aligned}
\theta^{-1} \circ f \circ \mathrm{X}^k\theta(V_1, \ldots, V_k) &= \theta^{-1} \circ f(\log(V_1), \ldots, \log(V_k)) \\
&= \theta^{-1}((\log(V_1) + \ldots + \log(V_k))/k) \\
&= \exp((\log(V_1) + \ldots + \log(V_k))/k) \\
&= \exp(\log(V_1 \ldots V_k)/k) \\
&= \exp(\log(V_1 \ldots V_k))^{1/k} \\
&= (V_1 \ldots V_k)^{1/k} \\
&= \sqrt[k]{V_1 \ldots V_k} \\
&= \mathbf{sgeomean}_1(V_1, \ldots, V_k)
\end{aligned}$$

This shows in Theorem 4c) why $\mathbf{sgeomean}_1$ is a conjugate of a linear function and therefore is scalar-free. Given this, the scaled geometric mean function $\mathbf{sgeomean}_\lambda$ for any λ can be written as

$$\mathbf{sgeomean}_\lambda = \mathrm{c}\ \mathbf{sgeomean}_1$$

with a constant factor $c = 1/\sqrt[k]{\boldsymbol{\lambda}}$ and therefore

$$\mathbf{sgeomean}_\lambda = \mathrm{c}\theta^{-1} \circ f \circ \mathrm{X}^k\theta$$

Therefore, also $\mathbf{sgeomean}_\lambda$ is scalar-free. This describes the function transformation underlying the equation transformation that has been used in Sect. 18.7 to transform the nonlinear geometric equilibrium equations into linear equations.

18.13 Appendix: Proofs

In this section proofs for Proposition 1 and Theorems 2 to 5 from Sects. 18.10–18.12 can be found.

18.13.1 Additive, Multiplicative, Log-Like and Exp-Like Functions

Proposition 1 (relating additive, multiplicative, log-like, and exp-like functions) Let θ: $R \to \mathbb{R}$ be any function for a finite or infinite interval R in $\mathbb{R}$, then the following hold:

(a) If θ is multiplicative, then $\log \circ\, \theta$ is log-like.
(b) If θ is log-like, then $\theta \circ \exp$ is additive.
(c) If θ is exp-like, then $\log \circ\, \theta$ is additive.
(d) If θ is multiplicative, then $\log \circ\, \theta \circ \exp$ is additive.
(e) For any multiplicative function such that $\theta(\alpha) = 0$ for some $\alpha \neq 0$, it holds that $\theta(\alpha) = 0$ for all α. For any nonzero multiplicative function θ it holds $\theta(1) = 1$ and $\theta(\alpha^{-1}) = \theta(\alpha)^{-1}$ for all α.
(f) If θ is multiplicative then $\theta(1) = 1$ and $\theta(\alpha^{-1}) = \theta(\alpha)^{-1}$. If a multiplicative θ is injective on Dom(θ), then it has an inverse θ^{-1} with Dom(θ^{-1}) = Range(θ) and Range(θ^{-1}) = Dom(θ); this inverse θ^{-1} is also multiplicative.

Proof

(a) This follows from

$$\log(\theta(\alpha\beta)) = \log(\theta(\alpha)\theta(\beta)) = \log(\theta(\alpha)) + \log(\theta(\beta))$$

(b) This follows from

$$\theta(\exp(\alpha + \beta)) = \theta(\exp(\alpha)\exp(\beta)) = \theta(\exp(\alpha)) + \theta(\exp(\beta))$$

(c) This follows from

$$\log(\theta(\alpha + \beta)) = \log(\theta(\alpha)\theta(\beta)) = \log(\theta(\alpha)) + \log(\theta(\beta))$$

(d) This immediately follows from a) and b).
(e) Suppose $\theta(\alpha) = 0$ for some $\alpha \neq 0$, then for any β it holds $\theta(\beta) = \theta(\alpha\beta\alpha^{-1}) = \theta(\alpha)\theta(\beta\alpha^{-1}) = 0$. Next, for any nonzero θ it holds $\theta(1) = \theta(1^2) = \theta(1)^2$; as it cannot be 0 from this it follows that $\theta(1) = 1$. The last part follows from $\theta(\alpha)\theta(\alpha^{-1}) = \theta(\alpha\alpha^{-1}) = \theta(1) = 1$.
(f) Choose any α', $\beta' \in$ Range(θ), then $\alpha' = \theta(\alpha)$ and $\beta' = \theta(\beta)$ for some $\alpha,\beta \in$ Dom(θ). Then this follows from

$$\theta^{-1}(\alpha'\beta') = \theta^{-1}(\theta(\alpha)\theta(\beta)) = \theta^{-1}(\theta(\alpha\beta)) = \alpha\beta = \theta^{-1}(\alpha')\theta^{-1}(\beta')$$

■

Theorem 2 (characterisation of additive, multiplicative, log-like and exp-like functions) *Let θ: $R \to \mathbb{R}$ be continuous. Then the following hold.*

(a) Assume $R \subseteq \mathbb{R}$ is closed under addition and subtraction with $1 \in R$, then it holds.
θ is additive $\Leftrightarrow$ for some $c \in \mathbb{R}$ for all X it holds $\theta(X) = c\,X$.
(b) Assume $R \subseteq \mathbb{R}_{>0}$ is closed under multiplication and division with $e \in R$, then it holds.
θ is multiplicative $\Leftrightarrow$ for some $c \in \mathbb{R}$ for all X it holds $\theta(X) = X^c$.
(c) Assume $R \subseteq \mathbb{R}_{>0}$ is closed under multiplication and division with $e \in R$, then it holds.
θ is log-like $\Leftrightarrow$ for some $c \in \mathbb{R}$ for all X it holds $\theta(X) = c \log(X)$.
(d) Assume $R = \mathbb{R}$ is closed under addition and subtraction with $1 \in R$, then it holds.
θ is exp-like $\Leftrightarrow$ for some $c \in \mathbb{R}$ for all X it holds $\theta(X) = \exp(cX)$.

Proof Note that all implications from right to left are easy to verify. The opposite implications can be found as follows.

(a) Note that $0 = 1 - 1 \in R$ and $\theta(0) = 0$ as from additivity it follows

$$\theta(0) = \theta(0 + 0) = 2\theta(0)$$

Therefore for any $c \in \mathbb{R}$ it holds $\theta(X) = cX$ for $X = 0$. Now, first for positive rational numbers $X = p/q \in R$ with $p, q \in \mathbb{N}$ with $p, q > 0$, from additivity it follows

$$q\theta(X) = \theta(qp/q) = \theta(p) = p\theta(1)$$

And therefore

$$\theta(X) = cX$$

where $c = \theta(1)$. Moreover, for any negative rational number $X = -p/q \in R$ with p, $q > 0$ it holds

$$\theta(-p/q) + \theta(p/q) = \theta(0) = 0$$

And therefore

$$\theta(X) = \theta(-p/q) = -\theta(p/q) = -\mathrm{c}p/q = cX$$

This proves that $\theta(X) = \mathrm{c}X$ for all rational numbers X.

Next, as any real number X is the limit of a sequence r_n, $n \in \mathbb{N}$ of rational numbers and both θ and the function $X \to \mathrm{c}X$ are continuous it holds

$$\theta(X) = \theta(\lim_{n\to\infty} r_n) = \lim_{n\to\infty} \theta(r_n) = \lim_{n\to\infty} c\, r_n = c \lim_{n\to\infty} r_n = cX$$

(b) Note that $R' = \log(R)$ is closed under addition and subtraction and $1 = \log(\mathrm{e}) \in R'$. By Proposition 1d) the function $\log \circ \theta \circ \exp$ on R' is additive. Therefore, by a) it follows that there is a $\mathrm{c} \in \mathbb{R}$ such that for any $X \in R$ for $Y = \log(X)$ it holds

$$\log \circ \theta \circ \exp(Y) = \mathrm{c}Y$$

From this it follows

$$\begin{aligned}
&\exp(\log \circ \theta \circ \exp(Y)) = \exp(cY) \\
&\theta \circ \exp(Y) = \exp(cY) \\
&\theta \circ \exp(Y) = \exp(Y)^c \\
&\theta(X) = X^c
\end{aligned}$$

(c) Note that $R' = \log(R)$ is closed under addition and subtraction and $1 = \log(\mathrm{e}) \in R'$. By Proposition 1b) the function $\theta \circ \exp$ is additive on R'. Therefore, by a) it follows that there is a $\mathrm{c} \in \mathbb{R}$ such that for any $X \in R$ for $Y = \log(X)$ it holds

$$\begin{aligned}
&\theta \circ \exp(Y) = cY \\
&\theta \circ \exp(\log(X)) = c\log(X) \\
&\theta(X) = c\log(X)
\end{aligned}$$

(d) By Proposition 1c) the function $\log \circ \theta$ is additive. Therefore, by a) it follows that there is a $\mathrm{c} \in \mathbb{R}$ such that for all $X \in R$ it holds

$$\begin{aligned}
&\log \circ \theta(X) = \mathrm{c}X \\
&\exp(\log \circ \theta(X)) = \exp(cX) \\
&\theta(X) = \exp(cX)
\end{aligned}$$

■

18.13.2 Weakly Scalar-Free And Scalar-Free Functions

Theorem 3 (relating weakly scalar-free and scalar-free functions)
Consider functions f: $R^k \to \mathbb{R}$ and θ: $R \to \mathbb{R}$ for some subset $R \subseteq \mathbb{R}$ which is $\mathbb{R}$ or $\mathbb{R}_{>0}$.

(a) If a nonzero function f is weakly scalar-free for function θ, then θ is multiplicative. If, moreover, f is (strictly) monotonically increasing and has at least one positive value, then θ is also (strictly) monotonically increasing. Therefore for the strict monotonically increasing case, θ is injective and has an inverse θ^{-1} on Range(θ), which is also multiplicative.

(b) Any nonzero multiplicative function θ is weakly scalar-free for itself.
(c) For any weakly scalar-free function f for θ the following are equivalent:

(iii) Range(f) $\subseteq$ Range(θ)
(iv) For all $V_1, \ldots, V_k$ an α exists such that $f(\alpha V_1, \ldots, \alpha V_k) = 1$

(d) For each weakly scalar-free function f: $R^k \to \mathbb{R}$ for any injective θ, the function g: Range$(\theta)^k \to \mathbb{R}$ defined by $g = f\ \mathrm{X}^k\theta^{-1}$ is scalar-free. If, moreover, Range(f) $\subseteq$ Range(θ), then also the function h: $R^k \to \mathbb{R}$ defined by $h = \theta^{-1}f$ is scalar-free. For strictly increasing f and θ, these functions g,h are strictly increasing too.
(e) For each set of strictly increasing and weakly scalar-free functions f_i: $R^k \to \mathbb{R}_{\geq 0}$ for the same strictly increasing θ, for any linear combination f of the f_i with positive coefficients, the function g: $R^k \to \mathbb{R}$ defined by $g = f\ \mathrm{X}^k\theta^{-1}$ is strictly increasing and scalar-free. If, moreover, Range(f) $\subseteq$ Range(θ), then also the function h: $R^k \to \mathbb{R}$ defined by $h = \theta^{-1}f$ is strictly increasing and scalar-free.
(f) If f: $R^k \to \mathbb{R}$ is scalar-free, θ: $R \to R$ is multiplicative and $g = f \text{ o } \mathrm{X}^k\theta$: $R^k \to \mathbb{R}$, then g is weakly scalar-free forθ. This holds in particular if f is linear.

Proof

(a) Suppose $f(V_1, \ldots, V_k) \neq 0$, then from

$\theta(\alpha\beta) f(V_1, \ldots, V_k) = f(\alpha\beta V_1, \ldots, \alpha\beta V_k) = \theta(\alpha) f(\beta V_1, \ldots, \beta V_k) = \theta(\alpha)\theta(\beta)\, f(V_1, \ldots, V_k)$
it follows that θ is multiplicative.

Suppose, moreover, f is (strictly) monotonically increasing and positive for at least one point $f(V_1, \ldots, V_k) > 0$ and $\alpha \leq \beta$ then from

$$\theta(\alpha) f(V_1, \ldots, V_k) = f(\alpha V_1, \ldots, \alpha V_k) \leq f(\beta V_1, \ldots, \beta V_k) = \theta(\beta)\, f(V_1, \ldots, V_k)$$

it follows that $\theta(\alpha) \leq \theta(\beta)$; it works similarly for the strict condition.

(b) This follows from $\theta(\alpha\beta) = \theta(\alpha)\theta(\beta)$.

(c) (i) $\Rightarrow$ (ii) Conversely suppose Range(f) $\subseteq$ Range(θ), then for any $V_1, \ldots, V_k$ it holds.

$f(V_1, \ldots, V_k) \in$ Range(θ).
$f(V_1, \ldots, V_k) = \theta(\beta)$ for some $\beta \in$ Dom(θ).
Then

$$\theta(\beta)^{-1} f(V_1, \ldots, V_k) = 1$$

Now pick $\alpha = \beta^{-1}$, then it follows.
$f(\alpha V_1, \ldots, \alpha V_k) = \theta(\alpha) f(V_1, \ldots, V_k) = \theta(\beta^{-1}) f(V_1, \ldots, V_k) = \theta(\beta)^{-1} f(V_1, \ldots, V_k) = 1$
(ii) $\Rightarrow$ (i) Suppose for any given $V_1, \ldots, V_k$ an α exists such that $f(\alpha V_1, \ldots, \alpha V_k) = 1$, then:

$$\begin{aligned} f(V_1, \ldots, V_k) &= f(\alpha^{-1}\alpha V_1, \ldots, \alpha^{-1}\alpha V_k) = \theta(\alpha^{-1}) f(\alpha V_1, \ldots, \alpha V_k) \\ &= \theta(\alpha^{-1}) \in \text{Range}(\theta) \end{aligned}$$

(d) For g the first part follows from

$$\begin{aligned} g(\alpha V_1, \ldots, \alpha V_k) &= f(\theta^{-1}(\alpha V_1), \ldots, \theta^{-1}(\alpha V_k)) \\ &= f(\theta^{-1}(\alpha)\theta^{-1}(V_1), \ldots, \theta^{-1}(\alpha)\theta^{-1}(V_k)) \\ &= \theta\theta^{-1}(\alpha) f(\theta^{-1}(V_1), \ldots, \theta^{-1}(V_k)) \\ &= \alpha f(\theta^{-1}(V_1), \ldots, \theta^{-1}(V_k)) \\ &= \alpha g(V_1, \ldots, V_k) \end{aligned}$$

And for h from

$$\begin{aligned} h(\alpha V_1, \ldots, \alpha V_k) &= \theta^{-1}(f(\alpha V_1, \ldots, \alpha V_k)) \\ &= \theta^{-1}(\theta(\alpha) f(V_1, \ldots, V_k)) \\ &= \theta^{-1}\theta(\alpha)\theta^{-1}(f(V_1, \ldots, V_k)) \\ &= \alpha\theta^{-1}(f(V_1, \ldots, V_k)) \\ &= \alpha h(V_1, \ldots, V_k) \end{aligned}$$

The second part follows from (a).

(e) This follows from (d) and (a).
(f) This follows from.

$$\begin{aligned} g(\alpha V_1, \ldots, \alpha V_k) &= f(\theta(\alpha V_1), \ldots, \theta(\alpha V_k)) \\ &= f(\theta(\alpha)\theta(V_1), \ldots, \theta(\alpha)\theta(V_k)) \\ &= \theta(\alpha) f(\theta(V_1), \ldots, \theta(V_k)) \\ &= \theta(\alpha) g(V_1, \ldots, V_k) \end{aligned}$$

■

18.13.3 Creating Scalar-Free Functions Based on Conjugates

Proposition 3 (function conjugate operator) Let subsets R, $S \subseteq \mathbb{R}$ be given, and functions g: $S^k \to S$, f: $R^k \to R$, and bijective θ: $S \to R$. Then the following hold:

(a) Then the following are equivalent:

(i) g is a function conjugate of f by θ

(ii) The following commutation rules for θ, f and g hold:

$$\theta g = f \mathrm{X}^k \theta$$
$$\theta^{-1} f = g \mathrm{X}^k \theta^{-1}$$

(b) If a)(i) and (ii) hold, then for any g such an f is unique and can be denoted by $f = S_\theta(g)$ for a *function conjugate operator* S_θ; similarly, $g = S_{\theta^{-1}}(f)$ for function conjugate operator $S_{\theta^{-1}}$, so it holds:

$$\theta g = S_\theta(g) \mathrm{X}^k \theta$$
$$\theta^{-1} f = S_{\theta^{-1}}(f) \mathrm{X}^k \theta^{-1}$$

These operators S_θ and $S_{\theta^{-1}}$ are each other's inverse and they preserve function addition and composition: for all f, g, f_1, f_2, g_1 and g_2 of proper types it holds

$$S_{\theta^{-1}} S_\theta(g) = g$$
$$S_\theta S_{\theta^{-1}}(f) = f$$
$$S_\theta(g_1 + g_2) = S_\theta(g_1) + S_\theta(g_2)$$
$$S_\theta(g_1 \circ g_2) = S_\theta(g_1) \circ S_\theta(g_2)$$
$$S_{\theta^{-1}}(f_1 + f_2) = S_{\theta^{-1}}(f_1) + S_{\theta^{-1}}(f_2)$$
$$S_{\theta^{-1}}(f_1 \circ f_2) = S_{\theta^{-1}}(f_1) \circ S_{\theta^{-1}}(f_2)$$

Moreover, when conjugate operators S_{θ_1} and S_{θ_1} for θ_1 and θ_2 are applied in turn, it holds.

$S_{\theta_1\theta_2}(g) = S_{\theta_1} S_{\theta_2}(g) S_{\theta_1}(\mathrm{X}^k\theta_2)\mathrm{X}^k\theta_1\mathrm{X}^k\theta_2^{-1}\mathrm{X}^k\theta_1^{-1}$ for all g.

If in addition, θ_1 and θ_2 commute (i.e., $\theta_1\theta_2 = \theta_2\theta_1$), then $S_{\theta_1} S_{\theta_2} = S_{\theta_1\theta_2}$:

$S_{\theta_1} S_{\theta_2}(g) = S_{\theta_1\theta_2}(g)$ for all g.

The condition that θ_1 and θ_2 commute is always fullfilled when θ_1 and θ_2 are both multiplicative, both additive, or both log-like or exp-like.

Proof

(a) (i) $\Rightarrow$ (ii) This follows from.

$$\theta g = \theta\theta^{-1} \circ f \circ \mathrm{X}^k \theta = f \circ \mathrm{X}^k \theta$$

and

$$\theta^{-1} f = \theta^{-1} f \mathrm{X}^k \theta \mathrm{X}^k \theta^{-1} = g \mathrm{X}^k \theta^{-1}$$

(ii) ⇒ (i) This follows from

$$g = \theta^{-1} \theta g = \theta^{-1} \circ f \circ \mathrm{X}^k \theta$$

(b) First, suppose $\theta g = f_1 \mathrm{X}^k \theta = f_2 \mathrm{X}^k \theta$, then from θ bijective it follows $f_1 = f_2$. Then an operator S_θ exists and it holds

$$\begin{aligned}\theta g &= S_\theta(g) \mathrm{X}^k \theta \\ \theta^{-1} f &= S_{\theta^{-1}}(f) \mathrm{X}^k \theta^{-1}\end{aligned}$$

Furthermore, consider

$$\begin{aligned}&\theta(g_1 + g_2) = S_\theta(g_1 + g_2) \mathrm{X}^k \theta \\ &\theta(g_1 + g_2) = \theta g_1 + \theta g_2 = S_\theta(g_1) \mathrm{X}^k \theta + S_\theta(g_2) \mathrm{X}^k \theta = (S_\theta(g_1) + S_\theta(g_2)) \mathrm{X}^k \theta\end{aligned}$$

Then,

$$\begin{aligned}S_\theta(g_1 + g_2) \mathrm{X}^k \theta &= (S_\theta(g_1) + S_\theta(g_2)) \mathrm{X}^k \theta \\ S_\theta(g_1 + g_2) &= S_\theta(g_1) + S_\theta(g_2)\end{aligned}$$

Also,

$$\begin{aligned}\theta(g_1 \circ g_2) &= S_\theta(g_1 \circ g_2) \mathrm{X}^k \theta \\ (\theta g_1) \circ g_2 &= S_\theta(g_1) \mathrm{X}^{k_1} \theta g_2 = S_\theta(g_1) S_\theta(g_2) \mathrm{X}^{k_1 k_2} \theta\end{aligned}$$

Here $k_1 k_2 = k$; therefore,

$$S_\theta(g_1 \circ g_2) = S_\theta(g_1) \circ S_\theta(g_2)$$

And the same applies to θ^{-1}.

When conjugate operators S_{θ_1} and S_{θ_2} for θ_1 and θ_2 are applied in turn, it holds

$$\begin{aligned}&(\theta_1 \theta_2) g = S_{\theta_1 \theta_2}(g) \mathrm{X}^k \theta_1 \theta_2 = S_{\theta_1 \theta_2}(g) \mathrm{X}^k \theta_1 \mathrm{X}^k \theta_2 \\ &\theta_1(\theta_2 g) = \theta_1 S_{\theta_2}(g) \mathrm{X}^k \theta_2 = S_{\theta_1}(S_{\theta_2}(g) \mathrm{X}^k \theta_2) \mathrm{X}^k \theta_1 = S_{\theta_1} S_{\theta_2}(g) S_{\theta_1}(\mathrm{X}^k \theta_2) \mathrm{X}^k \theta_1\end{aligned}$$

Therefore

$$S_{\theta_1 \theta_2}(g) \mathrm{X}^k \theta_1 \mathrm{X}^k \theta_2 = S_{\theta_1} S_{\theta_2}(g) S_{\theta_1}(\mathrm{X}^k \theta_2) \mathrm{X}^k \theta_1$$

$$S_{\theta_1\theta_2}(g)\mathrm{X}^k\theta_1 = S_{\theta_1}S_{\theta_2}(g)S_{\theta_1}(\mathrm{X}^k\theta_2)\mathrm{X}^k\theta_1\mathrm{X}^k\theta_2^{-1}$$
$$S_{\theta_1\theta_2}(g) = S_{\theta_1}S_{\theta_2}(g)S_{\theta_1}(\mathrm{X}^k\theta_2)\mathrm{X}^k\theta_1\mathrm{X}^k\theta_2^{-1}\mathrm{X}^k\theta_1^{-1}$$

If θ_1 and θ_2 commute, then $S_{\theta_1}(\mathrm{X}^k\theta_2) = \mathrm{X}^k\theta_2$ and therefore, this becomes

$$S_{\theta_1\theta_2}(g) = S_{\theta_1}S_{\theta_2}(g)\mathrm{X}^k\theta_2\mathrm{X}^k\theta_1\mathrm{X}^k\theta_2^{-1}\mathrm{X}^k\theta_1^{-1}$$
$$S_{\theta_1\theta_2}(g) = S_{\theta_1}S_{\theta_2}(g)$$

From Theorem 2 it follows that the condition that θ_1 and θ_2 commute is always fullfilled when θ_1 and θ_2 are both multiplicative, both additive, or both log-like or exp-like. By applying this to $\theta_1 = \theta$ and $\theta_2 = \theta^{-1}$ which also commute, it follows that S_θ and $S_{\theta^{-1}}$ are inverses of each other (here id indicates the identity mapping):

$$S_{\theta^{-1}}S_\theta(g) = S_{\theta^{-1}\theta}(g) = S_{\mathrm{id}}(g) = \mathrm{id}(g)$$
$$S_\theta S_{\theta^{-1}}(f) = S_{\theta\theta^{-1}}(f) = S_{\mathrm{id}}(f) = \mathrm{id}(f)$$

■

Theorem 4 (from scalar-free functions to scalar-free conjugates by multiplicative θ)

(a) For any scalar-free function f: $R^k \to R$ with $R = \mathbb{R}_{\geq 0}$, all of its conjugates by a multiplicative θ: $R \to R$ are also scalar-free.

(b) More specifically, for any scalar-free function f, for any positive real number n the function g defined by

$$g(V_1, \ldots, V_k) = \sqrt[n]{f(V_1^n, \ldots, V_k^n)}$$

Is a conjugate of f by the multiplicative function θ: $X \to X^n$ and therefore is also scalar-free.

(c) All weighted euclidean functions are conjugates of linear functions by a multiplicative function θ and therefore are scalar-free. In particular, this holds for all functions $\mathbf{eucl}_{n,\lambda}$**(..)**.

Proof

(a) If $f(V_1, \ldots, V_k)$ scalar-free and θ multiplicative and $g = \theta^{-1} \circ f \circ \mathrm{X}^k\theta$, i.e.,

$$g(V_1, \ldots, V_k) = \theta^{-1}(f(\theta(V_1), \ldots, \theta(V_k))$$

Then:

$$\begin{aligned} g(\alpha V_1, \ldots, \alpha V_k) &= \theta^{-1}(f(\theta(\alpha V_1), \ldots, \theta(\alpha V_k)) \\ &= \theta^{-1}(f(\theta(\alpha)\theta(V_1), \ldots, \theta(\alpha)\theta(V_k)) \end{aligned}$$

$$
\begin{aligned}
&= \theta^{-1}(\theta(\alpha) f(\theta(V_1), \ldots, \theta(V_k)) \\
&= \theta^{-1}(\theta(\alpha))\theta^{-1}(f(\theta(V_1), \ldots, \theta(V_k))) \\
&= \alpha g(V_1, \ldots, V_k)
\end{aligned}
$$

Therefore g is scalar-free.

(b) Substitute $\theta(X) = X^n$ and $\theta^{-1}(X) = X^{(1/n)}$, then

$$
g(V_1, \ldots, V_k) = \theta^{-1}(f(\theta(V_1), \ldots, \theta(V_k)) = \sqrt[n]{f(V_1^n, \ldots, V_k^n)}
$$

Then by (a) this function is scalar-free.

(c) When starting with a linear function in b), you get the general format of a weighted Euclidean function g.

$$
\begin{aligned}
g(V_1, \ldots, V_k) &= \theta^{-1}(f(\theta(V_1), \ldots, \theta(V_k)) \\
&= \theta^{-1}(f(V_1^n, \ldots, V_k^n)) \\
&= \theta^{-1}(w_1 V_1^n + \ldots + w_k V_k^n) \\
&= \sqrt[n]{w_1 V_1^n + \ldots + w_k V_k^n}
\end{aligned}
$$

■

Theorem 5 (from linear to scalar-free conjugates by log-like θ)

(a) For any linear function with sum of coefficients 1 all of its conjugates by a log-like θ are scalar-free.

(b) More specifically, for any linear function f with sum of coefficients 1 the function g defined by

$$
g(V_1, \ldots, V_k) = \exp(f(\log(V_1), \ldots, \log(V_k)))
$$

is a conjugate of a linear function by the standard log-like function $\theta = \log$ and therefore is scalar-free.

(c) All weighted geometric functions are conjugates of a linear function by a log-like function θ and therefore are scalar-free. In particular, this also holds for all functions **sgeomean**$_\lambda$**(..)**.

Proof

(a) If $f(V_1, \ldots, V_k)$ linear with sum of coefficients 1 and θ log-like and $g = \theta^{-1} \circ f \circ X^k\theta$, i.e.,

$$
g(V_1, \ldots, V_k) = \theta^{-1}(f(\theta(V_1), \ldots, \theta(V_k))
$$

Then g is scalar-free:

$$\begin{aligned} g(\alpha V_1, \ldots, \alpha V_k) &= \theta^{-1}(f(\theta(\alpha V_1), \ldots, \theta(\alpha V_k)) \\ &= \theta^{-1}(f(\theta(\alpha) + \theta(V_1), \ldots, \theta(\alpha) + \theta(V_k)) \\ &= \theta^{-1}(f(\theta(\alpha), \ldots, \theta(\alpha)) + f(\theta(V_1), \ldots, \theta(V_k))) \\ &= \theta^{-1}(f(\theta(\alpha), \ldots, \theta(\alpha))) * \theta^{-1}(f(\theta(V_1), \ldots, \theta(V_k))) \\ &= \theta^{-1}(\theta(\alpha)) * g(V_1, \ldots, V_k) \\ &= \alpha g(V_1, \ldots, V_k) \end{aligned}$$

(b) If

$$f(V_1, \ldots, V_k) = w_1 V_1 + \ldots + w_k V_k$$

Then

$$\begin{aligned} g(V_1, \ldots, V_k) &= \exp(w_1 \log(V_1) + \ldots + w_k \log(V_k)) \\ &= \exp(\log(V_1^{w_1}) + \ldots + \log(V_k^{w_k})) \\ &= \exp(\log(V_1^{w_1} \ldots V_k^{w_k})) \\ &= V_1^{w_1} \ldots V_k^{w_k} \end{aligned}$$

(c) This immediately follows from b). As **sgeomean**$_\lambda$**(..)** is the weighted geometric function **sgeomean**$_1$**(..)** times a constant factor, it is also scalar-free. ■

18.14 Discussion

The contents of this chapter are based on parts of Treur (2016, 2018, 2021) and Treur (2020b), Ch 12. In this chapter it was discussed how mathematical analysis can be used to find out some properties of the dynamics of a network model. By comparing such properties found by mathematical analysis and properties observed in simulation experiments, verification can be done of whether an implemented network model is correct with respect to its design specification. If the mathematical analysis expects a certain property but an example simulation does not satisfy this property, this should be a reason to inspect the implementation of the model and correct errors (and/or check whether the mathematical analysis is correct). This option for evaluation and feedback can be very useful during a development process of a network model.

Useful mathematical techniques for such types of analysis in general have been around already for quite some time; e.g., (Brauer and Nohel 1969; Lotka 1924; Picard 1891; Poincaré 1881, 1892). Mathematical analysis may not always be easy. The more easy options are when linear functions (for example, scaled sum combination functions) are used so that linear equilibrium equations occur that by using a linear solver in principle can be solved symbolically, thereby obtaining expressions for equilibrium values in terms of the characteristics of the network model. On the other hand, equilibrium equations involving logistic functions cannot be solved symbolically in that way. Nevertheless, for such cases still specific instances can be addressed: as shown in Sect. 18.2, verification of a network model does not depend exclusively on finding explicit symbolic solutions of the equilibrium equations. For verification purposes, it is sufficient if the equilibrium equations have been identified, which is always possible using the criterion based on the standard difference or differential equation used for network models here. Then, for any simulation, observed equilibrium values can be substituted in these equations and by this it is checked whether they satisfy the equations.

To analyse and predict at forehand from its structure what behaviour a given network model will eventually show can in general be a more challenging issue. For example, do all states in a social network model converge to the same value? Some earlier results address the case of acyclic, fully connected or strongly connected networks and for linear combination functions only; e.g., (Bosse et al. 2015). It is often believed that when nonlinear functions are used, such results cannot be found anymore. Also, networks that are not strongly connected are usually not addressed by analysis as they are assumed to be more difficult to handle. This chapter discussed more recently found much more general results showing what is still possible beyond the case of linear combination functions (for aggregation) and also beyond the case of strongly connected networks (for connectivity). These general results relate network behaviour to the network structure characteristics of two types in particular:

- Network *connectivity characteristics* in terms of the network's strongly connected components and their mutual connections and the stratification they induce
- Network *aggregation characteristics* in terms of the combination functions used to aggregate multiple incoming connections (in particular, monotonicity, scalar-freeness and normalisation).

The first item makes the network analysis approach applicable to any type of network connectivity, thus going beyond the limitation to strongly connected networks or acyclic networks. The second item makes the network analysis approach applicable to a wider class of combination functions (most of which are nonlinear) going beyond the limitation to linear functions. For some specific types of nonlinear functions in this class (weighted Euclidean functions and weighted geometric functions), it has been shown how they can be transformed (by conjugates) into linear functions and how after such a transformation linear equilibrium equations are obtained that can be solved easily. Also, for some other types of nonlinear functions it was shown how they can be handled: functions for Hebbian learning and functions for bonding by homophily as used for connectivity self-model states in self-modeling networks.

So, it was shown how, in contrast to often held beliefs, certain classes of nonlinear functions used for aggregation in network models enable analysis of the emerging within-network dynamics as easily as linear functions do. The chapter uses elements from Treur (2020a) and (2020b), Chapters 11 and 12, but also introduces a number of new concepts and methods for this type of network analysis, such as weakly scalar-free function, conjugate functions and the use of a linear solver to solve nonlinear equations. These new concepts and methods enable to get more insight in some of the types of nonlinear functions for which network analysis by solving equilibrium equations is well-feasible.

From scalar-free functions in a combinatorial manner new scalar-free functions can be generated easily, using (1) linear combinations of them, (2) function compositions of them, and (3) conjugates of them. By iteratively combining these three methods, scalar-free functions can be built of arbitrarily high complexity. This shows that there is a very large space of such nonlinear functions, which still are well-suitable for analysis. However, no complete classification of all possible types of nonlinear functions that are easy to handle (e.g., because of being scalar-free) has been obtained yet; there are still some remaining challenges for this.

References

Bosse, T., Duell, R., Memon, Z.A., Treur, J., van der Wal, C.N.: Agent-based modelling of emotion contagion in groups. Cogn. Comp. **7**(1), 111–136 (2015)

Bloem, R., Gabow, H.N., Somenzi, F.: An algorithm for strongly connected component analysis in n log n Symbolic Steps. Form. Meth. Syst. Des. **28**, 37–56 (2006)

Brauer, F., Nohel, J.A.: Qualitative Theory of Ordinary Differential Equations. Benjamin (1969)

Harary, F., Norman, R.Z., Cartwright, D.: Structural Models: An Introduction to the Theory of Directed Graphs. Wiley, New York (1965)

Hebb, D.O.: The Organization of Behavior: A Neuropsychological Theory. Wiley (1949)

Keysers, C., Gazzola, V.: Hebbian learning and predictive mirror neurons for actions, sensations and emotions. Philos. Trans. r. Soc. Lond. b: Biol. Sci. **369**, 20130175 (2014)

Łacki, J.: Improved deterministic algorithms for decremental reachability and strongly connected components. ACM Trans. Algorithms *9*(3), Article 27 (2013)

Lotka, A.J.: Elements of Physical Biology. Williams and Wilkins Co. (1924), Dover Publications, 2nd ed. (1956)

McPherson, M., Smith-Lovin, L., Cook, J.M.: Birds of a feather: homophily in social networks. Annu. Rev. Sociol. **27**, 415–444 (2001)

Picard, E.: Traité d'Analyse, vol. 1 (1891), vol. 2 (1893)

Poincaré, H.: Mémoire sur les courbes défine par une équation différentielle (On curves defined by differential equations) (1881–1882)

Poincaré, H.: New Methods of Celestial Mechanics, 3 vols. (1892–1899) English translation, (1967)

Shatz, C.J.: The developing brain. Sci. Am. **267**, 60–67 (1992). https://doi.org/10.1038/scientificamerican0992-60

Treur, J.: Verification of temporal-causal network models by mathematical analysis. Vietnam J. Comput. Sci. **3**, 207–221 (2016)

Treur, J.: Relating emerging network behaviour to network structure. In: Proceedings of the 7th International Conference on Complex Networks and their Applications, ComplexNetworks'18, vol. 1. Studies in Computational Intelligence, vol. 812, pp. 619–634. Springer Publishers (2018)

Treur, J.: Analysis of a network's asymptotic behaviour via its structure involving its strongly connected components. Netw. Sci. **8**(S1), S82–S109 (2020a)

Treur, J.: Network-oriented modeling for adaptive networks: designing higher-order adaptive biological, mental and social network models. Springer Nature Publishers (2020b)

Treur, J.: Equilibrium analysis for within-network dynamics: from linear to nonlinear aggregation. In: Nguyen, N.T., et al. (eds.), Proceedings of the 13th International Conference on Computational Collective Intelligence, ICCCI'21. Lecture Notes in AI, vol. 12876, pp. 94-110. Springer Nature (2021)

Wijs, A., Katoen, J.P., Bošnacki, D.: Efficient GPU algorithms for parallel decomposition of graphs into strongly connected and maximal end components. Formal Methods Syst. Des. **48**, 274–300 (2016)