Mean Field Conditions explained by General Linear System Morphisms: Application to Neuronal Network Connections and Computational Approximation Perspective

Alexandre Muzy, Bernard P Zeigler

Mean Field Conditions explained by General Linear System Morphisms: Application to Neuronal Network Connections and Computational Approximation Perspective

Alexandre Muzy*, Bernard P. Zeigler†

January 28, 2020

1 Introduction

Mean field theory is used widely to abstract analytically a large pool of neurons (Faugeras, Touboul, and Cessac 2009; Nykamp et al. 2017; Ostojic 2014; El Boustani and Destexhe 2009). Considering homogeneous behaviors (all neurons have the same state transition functions) and all-to-all or randomly uniform couplings between neurons, particular equations of neurons are derived until the average firing rate of the pool. Usually, the connections between the neurons are taken to be inversely proportional to an infinite number of neurons leading to a weak coupling between neurons. The results obtained are specific to the neuronal equations derived. Both structure and dynamics of the pool of neurons constitute a rough abstraction of the complexity of an actual neuronal pool but this abstraction is worth to infer more knowledge about the global behavior resulting from the interactions between neurons.

On the other hand, linear systems (Zadeh and Desoer 1963) constitute a general analytical tool. Based on the linear properties of system dynamics, the set of parameters of the corresponding equations is usually studied through diagram phases and un/stability of the dynamics. Including non/linear systems, general system theory (Arbib 1972; Klir 1985; Mesarovic and Takahara 1989; Mesarovic and Takahara 1975; Wymore 1967; Arnold 1994; Harrison 1969; Ho 1992) has been developed to reason very generally over abstract states and system dynamics. This abstraction level is worth for manipulating and reasoning over system structures and behaviors. The intertwined structures and dynamics can be studied analytically inferring general properties thus providing more knowledge on the systems before simulating them.

*Université Côte d'Azur, I3S CNRS, France, Email: alexandre.muzy@cnrs.fr.

†Chief Scientist, RTSync Corp, 530 Bartow Drive Suite A Sierra Vista, AZ 85635, United-States of America.

This article presents the implementation of an abstraction of linear systems using a system morphism representation and mean field conditions. Computational modeling is done using the system specification formalism (Zeigler, Muzy, and Kofman 2018). Base networks of linear systems are abstracted into lumped networks. The lumping is detailed based on the base network. Mean field conditions ensure the preservation of the average activity in the network. Also, the coupling of networks is studied to be able to construct networks of networks while still getting a good idea of the dynamics of the whole system. Finally, the mathematical framework proposed allows differentiating between the convergence of the dynamics of networks of systems and the computational error introduced. Although usual mean field conditions lead to no error dynamics, these conditions can be relaxed (with a finite number of systems and non uniform coulings between the systems) identifying the frontier between analytical analysis and the necessity of simulation to better understand the dynamics of the overall network. Finally, all the results obtained are discussed in the context of neuronal network modeling.

In Section 2, mathematical system and mean field theories are introduced. In Section 3, the mathematical framework of linear time invariant systems with inputs/outputs is defined. In Section 4, the mean field abstraction is clearly defined for linear systems using computational morphisms. Section 6 presents the abstraction of network dynamics based on the connections in the network. Section 7 discusses the results obtained for linear systems in the context of neuron models. Finally, in Section 8 a conclusion is provided.

2 Mathematical general system and mean field theories

2.1 Mathematical system theory

Mathematical general systems consist of state-based systems with inputs and outputs. These systems can be linear or non-linear, with few hypotheses about their structure (as time invariance described in the mathematical framework section), making these structures very abstract. Input/Output (I/O) interactions of systems make them very realistic but require adding particular mathematical properties to derive theorems about the expected behavior of these systems. In the theory of modeling and simulation (Zeigler, Muzy, and Kofman 2018), a computational specification of general systems has been proposed. The computational systems considered here consist of linear systems.

2.2 Mean field theory

Many references using the mean field hypotheses in the context of neural networks could be cited here. In Table 1, we focus on the main usual hypotheses and breakthrough results with respect to neural network structures (Nykamp et al. 2017; El Boustani and Destexhe 2009; Ostojic 2014) (allowing neuronal

publication	non linear state correlation between neurons	all-to-all coupling btwn networks	many-to-many coupling btwn networks	infinite nb of neurons	non infinite nb of neurons	weights inv. prop. to nb of neurons	arbitrary degree distribution in a network
(Faugeras, Touboul, and Cessac 2009)	x	x		x		x	
(Nykamp et al. 2017)	x			x			x
(El Boustani and Destexhe 2009)				x			x
(Ostojic 2014)	?			x			x
us		x	x	x	x		x

Table 1: Mean field conditions in neural networks.

networks with arbitrary random degree distributions) and behavior (Faugeras, Touboul, and Cessac 2009) (exploring the state correlation between neurons). For mathematical convergence to a fixed point, usual hypotheses consist of all-to-all couplings between networks, an infinite number of neurons, weights inversely proportional to the number of neurons. We will show here that the all-to-all couplings between networks is sufficient but not necessary and that many-to-many coupling is necessary. The same way, we will show that the an infinite number of neurons in a network is sufficient but not necessary and that a finite number of neurons is possible.

3 Mathematical framework

3.1 I/O general state-based systems

Definition 1. A *deterministic general I/O system is a structure* (cf. behavior introduced in Figure 1)

$$SYS = (\delta, \lambda)$$

Where

$\delta : Q \times \Omega \to Q$ is the *transition function*, with Q the *set of states*, Ω the *set of (piecewise continuous) input segments* $\omega :< t, t_2 >\to X$[1], with $< t_1, t_2 >$ the *interval of the segment,* signs '$<$' and '$>$' correspond either to a square brackets '[' or a square bracket ']', and X the *set of input values.* The sets of input values X and states Q are arbitrary.

$\lambda : Q \to Y$ is the *output function*, which can be considered as a (partial) observation of the state of the system.

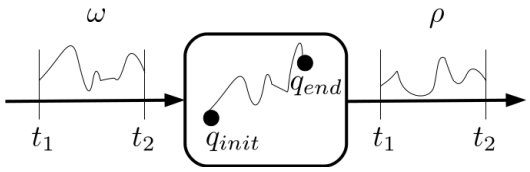

Figure 1: General I/O system dynamics: When receiving an input segment $\omega \in \Omega$, the system achieves a transition from initial state $q_{init} \in Q$ to final state $q_{end} \in Q$ and returns an output segment $\rho \in$ P.

For one input segment $\omega \in \Omega$ defined over an interval $< t_1, t_2 >$, with t_1 and t_2 not fixed[2], the system goes continuously from one initial state $q_{init} \in Q$ to one final state $q_{end} \in Q$ by its transition function: $q_{end} = \delta(q_{init}, \omega)$. To do so, intermediate states are computed for particular (allowed) time breakpoints $t \in< t_1, t_2 >$ based on the *composition property of the transition function* (cf. Figure 2): $\delta(q, \omega) = \delta(\delta(q, \omega_{t>}), \omega_{<t})$, with $\omega_{t>} = \omega|_{<t_1,t>}$ and $\omega_{t>} = \omega|_{<t,t_2>}$ being respectively the *left sub-segment* and the *right sub-segment of* ω. Finally, the system generates an *output segment* $\rho \in$ P such that $\rho :< t_1, t_2 >\to Y$ and $\rho_{t>} = \lambda(\delta(q, \omega_{t>}))$. The *set of input segments*, Ω, is the union of all input segments $\omega_{t>}$ and $\omega_{<t}$ and the *set of output segments*, P, is the union of all output segments $\rho_{t>}$ and $\rho_{<t}$.

[1] A piecewise continuous input segment is a map from each time point $t \in< t_1, t_2 >$ (with t_1 and t_2 not fixed) to a corresponding input value $x \in X$.

[2] Segments can be also defined as starting from time 0 showing then that they can be translated, this is the *time invariance property of systems* (Zeigler, Muzy, and Kofman 2018).

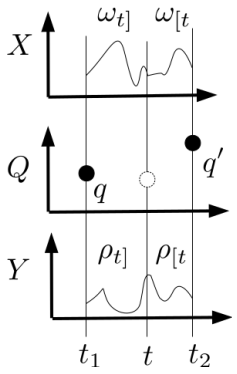

Figure 2: Composition of segments.

The *current state* is the minimal information to deterministically compute the *next state* in a very large state space. The system is markovian. However, notice that a current state can be seen as the result of previous input-state transitions (Zadeh and Desoer 1963). Then, the state of the system can be considered at a higher dependence order, a state being the result of several previous state transitions. Notice also that the system holds inputs and outputs, which is a more general and convenient principle for modeling complex systems, although it makes these systems more unpredictable. In (Ivanov 2013), it is proven also that previous inputs can be stored in states showing the equivalence of both closed and open system structures.

Systems are very abstract and general structures that proved to map all usual modeling formalisms (Zeigler, Muzy, and Kofman 2018). They allow integrating and comparing these formalisms. However, abstract does not mean trivial in the sense that the properties shown for arbitrary inputs, states and outputs can be shown to hold at a lower specification level, i.e., for specific inputs, states and outputs.

Systems are *time invariant*, i.e., any input segment $\omega :< t_1, t_2 >\rightarrow X$, applied at time t_1 can be applied at a time t_3, leading to the same state and output transitions. Defining a translation operator for each time $t \in T$, as $TRANS_\tau : \Omega \rightarrow \Omega$, for an input segment ω, $\omega' = TRANS(\omega)$, with $\omega'(t+\tau) = \omega(t)$ for all $t \in < t_1, t_2 >$. Then, a system $SYS = (\delta, \lambda)$ is time invariant for all input segments $\omega \in \Omega$ and all times $\tau \in T$, if:

1. Ω is *closed under translation*: for $\omega \in \Omega \Rightarrow TRANS_\tau(\omega) \in \Omega$.
2. δ is time invariant: for all states $q \in Q$, $\delta(q, \omega) = \delta(q, TRANS_\tau(\omega))$.

3.2 Linear time invariant systems

Definition 2. A Linear time invariant System (LSYS) is a structure

$$LSYS = (\delta, \lambda)$$

Where

X, Q, Y are finite dimensional vector spaces over $\mathbb{R}$,
$\delta(q,\omega) = qe^{Al(\omega)} + \int_0^{l(\omega)} e^{A(l(\omega)-\tau)}B\omega(\tau)d\tau$ is the transition function[3],
$\lambda(q) = CQ$ is the output function,
$A : Q \to Q$, $B : X \to Q$ and $C : Q \to Y$ are linear operators.

Definition 3. A matrix representation $\begin{cases} \frac{dq(t)}{dt} = Aq(t) + Bx(t) \\ y(t) = Cq(t) \end{cases}$ is represented by a Linear Time Invariant System as $\begin{cases} \delta(q,\omega) = qe^{At} + \int_0^t e^{A(t-\tau)}B\omega_{\tau>}d\tau \\ \lambda(q) = CQ \end{cases}$.

Definition 4. A linear time invariant system consists of linear transition and output functions with additive and distributive properties:

1. $\delta(q_1 + q_2, \omega_{1,t>} + \omega_{2,t>}) = \delta(q_1, \omega_{1,t>}) + \delta(q_2, \omega_{2,t>})$
2. $\delta(aq, a\omega_{t>}) = a\delta(q, \omega_{t>})$
3. $\lambda(\delta(q_1 + q_2, \omega_{1,t>} + \omega_{2,t>})) = \lambda(\delta(q_1, \omega_{1,t>})) + \lambda(\delta(q_2, \omega_{2,t>}))$

Let us prove these properties.

Proposition 1. $\delta(q_1 + q_2, \omega_{1,t>} + \omega_{2,t>}) = \delta(q_1, \omega_{1,t>}) + \delta(q_2, \omega_{2,t>})$

Proof. Based on matrix representation,

$$\begin{aligned} \delta(q_1 + q_2, \omega_{1,t>} + \omega_{2,t>}) &= e^{At}q_1 + \int e^{A(t-\tau)}B\omega_{1,\tau>}d\tau + e^{At}q_2 + \int e^{A(t-\tau)}B\omega_{2,\tau>}d\tau \\ &= e^{At}q_1 + e^{At}q_2 + \int e^{A(t-\tau)}B\omega_{1,\tau>}d\tau + \int e^{A(t-\tau)}B\omega_{2,\tau>}d\tau \\ &= e^{At}(q_1 + q_2) + \int e^{A(t-\tau)}B(\omega_{1,\tau>} + \omega_{2,\tau>})d\tau \\ &= \delta(q_1, \omega_{1,t>}) + \delta(q_2, \omega_{2,t>}) \end{aligned}$$

□

Proposition 2. $\delta(aq, a\omega_{t>}) = a\delta(q, \omega_{t>})$

Proof. Similar to Proposition 1. □

Proposition 3. $\lambda(\delta(q_1 + q_2, \omega_{1,t>} + \omega_{2,t>})) = \lambda(\delta(q_1, \omega_{1,t>})) + \lambda(\delta(q_2, \omega_{2,t>}))$

Proof. Starting from additive properties,

$$\begin{aligned} \lambda(\delta(q_1 + q_2, \omega_{1,t>} + \omega_{2,t>})) &= \lambda(\delta(q_1, \omega_{1,t>}) + \delta(q_2, \omega_{2,t>})) && \text{by Proposition 1} \\ &= C(\delta(q_1, \omega_{1,t>}) + \delta(q_2, \omega_{2,t>})) && \text{by Definition 2} \\ &= C(\delta(q_1, \omega_{1,t>}) + \delta(q_2, \omega_{2,t>})) && \text{by linearity of } C \\ &= \lambda(\delta(q_1, \omega_{1,t>})) + \lambda(\delta(q_2, \omega_{2,t>})) \end{aligned}$$

□

In the sequel we will use linear time invariant systems called linear systems for short.

[3] Notice that all segments are translated to 0, for simplicity.

3.3 General system morphisms

I/O general systems can be considered as abstract machines achieving *temporal computations (or executions of system (output) transitions functions*). A temporal computation relies on a delay (possibly zero) between inputs and outputs. Computations take time. The number of the computations should be finite to guarantee that the simulation ends.

Simulation and computers consist more and more of a huge number of components interacting together. A fundamental modeling challenge remains the development of a guiding mathematical framework to constructively set and analyze the behavior of networks of components at both local and global levels. The difficulty of developing such modeling structures is due to the number of local state computations and to the interactions between the components (the temporal coordination of the distributed computations). To abstract local system behaviors into network ones, relying on local (temporal) state computations, *system morphisms* can be used.

Definition 5. A *system morphism* or generalized homomorphism[4], between a detailed system SYS (or base model) and another abstract system SYS' (or lumped model), is a pair (g, h) such that (cf. Figure 3):

1. $g : \Omega \to \Omega'$, is the input mapping,
2. $h : \overline{Q} \to^{onto} Q'$, where $\overline{Q} \subseteq Q'$, is the state mapping,
3. for all $q \in \overline{Q}$, $\omega' \in \Omega'$,$h(\delta(q, g(\omega))) = \delta'(h(q), \omega')$ (transition function preservation)
4. $k : P \to P'$, is the output mapping.

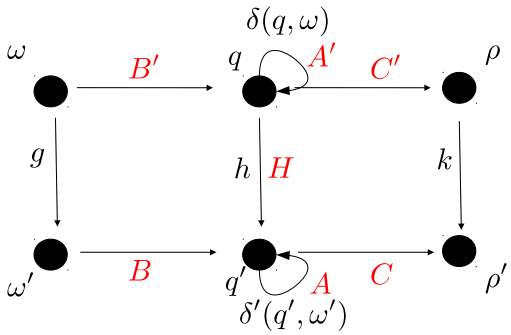

Figure 3: Commutative diagram of morphism mappings from a small system SYS to a big system SYS'. In red are indicated the matrix representations.

[4]We will use the term "homomorphism" in the sequel for state-to-state mapping in a system is considered and "morphism" when input/output systems are considered.

4 Network lumping based on mean field conditions

After having informally used networks of linear systems let's define more formally such networks. Figure 4 shows the morphism between a base network and a lumped network.

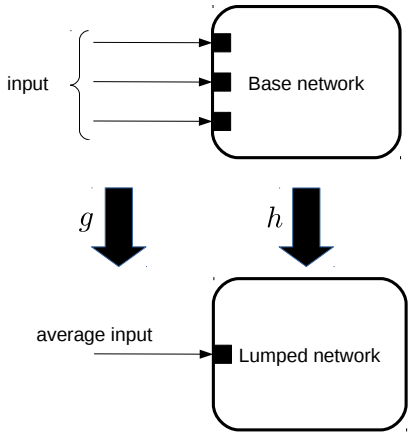

Figure 4: Morphism between base and lumped networks. An example of input average is provided here.

The morphism of networks can be achieved based on mean field conditions.

Definition 6. Usual mean field conditions can be summarized into only *two sufficient and necessary conditions* for abstracting linear networks:

1. Homogeneity: All the components of the network have the same dynamics (or state transition function) and the same structure (same transition/output functions and receive the same inputs),

2. Stability: The transition function of the network resultant system admits a fixed point, the system having input to output feedback.

Mean field morphism between a base network model and a lumped network model is presented in Figure 5.

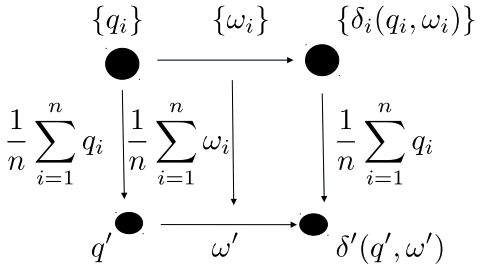

Figure 5: Commutative diagram of mean field network morphism.

Example 1. Let's detail the structure lumping achieved at coupling level between the components of a network and a simple *homomorphism* between the states of base and lumped networks. Based on homogeneity condition, the transitions of a base network embedding two pools of interacting components and a lumped network with two corresponding single states is shown in Figure 6. Following this commutative diagrams, the homomorphism requirement, for all states (q_A, q_B) in $Q_A \times Q_B$ and all states $(q_{A'}, q_{B'})$ in $Q_{A'} \times Q_{B'}$, is $\delta'(h(q_A, q_B)) = h(\delta(q_A, q_B))$, for $h(q_A, q_B) = (\sum_{i=1}^{n_A} q_{A_i}, \sum_{i=1}^{n_B} q_{B_i})$, i.e., taking the sum of the states of the n_A (resp. n_B) components in pool A (resp. B). At each time, the states of A' and B', i.e., (q_A, q_B) in $Q_{A'} \times Q_{B'}$ must turn out to be the same whether you compute them by:

1. First computing the base model transition and the projecting down using h, or
2. First projecting down using h and then computing the lumped model transition.

Besides, structural homogeneity requires the number senders in pool A being the same for each receiver in pool B.

Remark 1. Usual all-to-all coupling of components in the base network taken when mean field theory is applied to neural networks (Cessac 2019; Faugeras, Touboul, and Cessac 2009) is sufficient but not necessary. It can be seen that homomorphism requirement holds for either all-to-all, one-to-one, many-to-one couplings. Also, in the computational context, the number n of components needs not to be infinite.

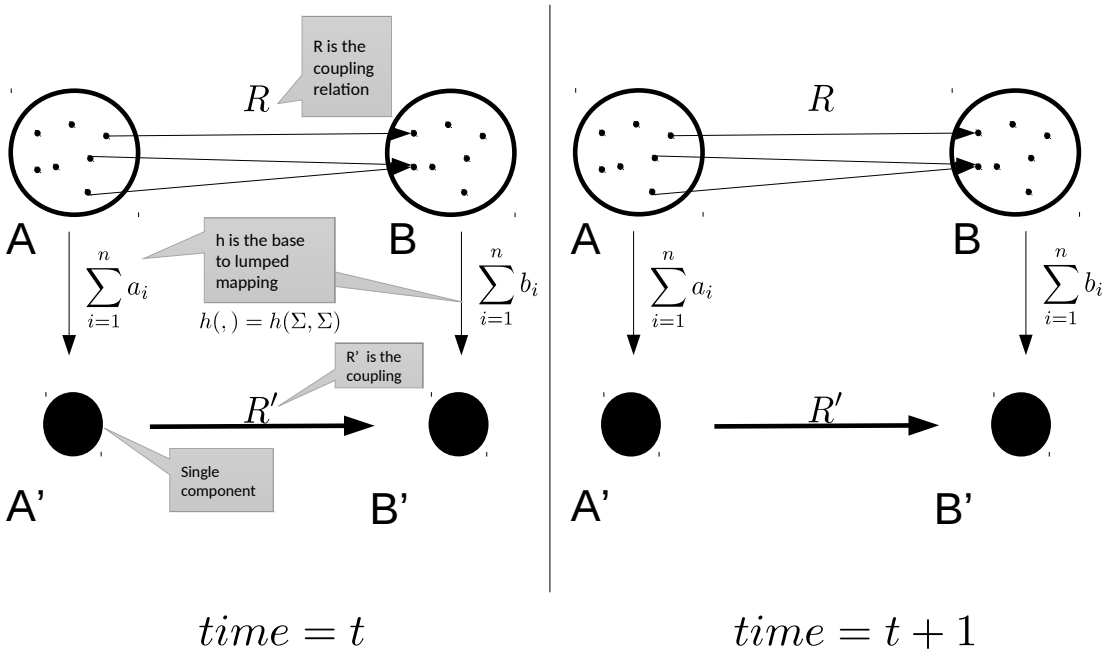

Figure 6: Base and lumped model transitions at times t and $t+1$ based on a simple homomorphism h between a base network and a lumped network. In the base network (resp. lumped network), components in pool A (resp. A') influence components in pool B (resp. B').

Both homogeneity and stability assumptions lead, for example, to the network structure of Figure 7. At structural level, each receiver gets the same values and has the same number of senders. The feedback consists of independent self couplings.

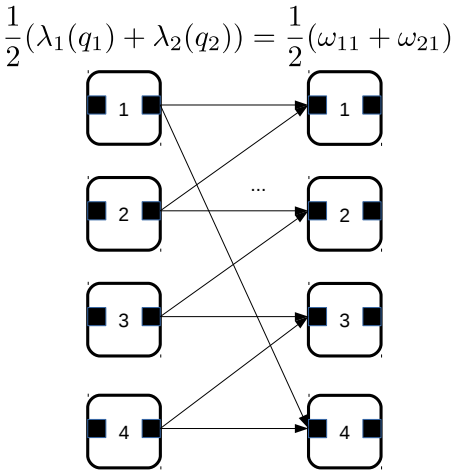

Figure 7: Example of two-to-one and self internal coupling in a mean field (lumped) network. The senders are represented on the first column and the receivers on the second. For the sake of simplicity, the same component is represented twice. The homomorphism here is the sum of component states.

Let's define now more precisely the base and lumped model structures.

Definition 7. A *base network of linear systems* consists of

$$\eta_{base} = (\delta, \lambda, \{S_i\})$$

Where

$\delta(q, \omega) = \times_i \delta_i(q_i, \omega_i)$, for $i = 1, ..., n$, is the *base network transition function* based on the *component transition functions* δ_i,which depend on: (i) the *base network state*, $q = (..., q_i, ...)$, with q_i the *component states* and on the *base network external input* $\omega = (..., \omega_i, ...)$, with ω_i the *component inputs*. Based on dynamics homogeneity mean field condition of Definition 6, all transition functions have the same behavior, i.e., $\delta_i \stackrel{def}{=} \delta_j$, for all components i, j of the network.

$\lambda(q) = \times_i \lambda_i(q_i)$ is the *base network output function* based on the *component output functions* λ_i. Based on dynamics homogeneity mean field condition of Definition 6, all output functions have the same behavior, i.e., $\lambda_i \stackrel{def}{=} \lambda_j$, for all components i, j of the network.

$\{S_i\} = \{S_i^{ext} \cup S_i^{int}\}$ is the *set of senders to the network* (cf. Example 1) such that $S_i = \{j \mid \omega_i = \lambda_j(\delta_j(q_j, \omega_j)\}$ is the *set of external or internal sending components* j (as pool A in Figure 6) connected to the input, $\omega_i \in \Omega_i$, of a *a receiving* component i. Structural homogeneity requires the number (external or internal) senders in pool A being the same for each receiver in

pool B and couplings between senders and receivers being either all-to-all (or one-to-all), one-to-one or many-to-one.

Definition 8. A *lumped network of linear systems* consist of

$$\eta_{lumped} = (\delta', \lambda', \{S'_i\})$$

Where

$\delta'(q', \omega') = \delta'(h(\times_i q), g(\times_i \omega_i))$ is the *transition function of the network*, where (for all $q \in \overline{Q}$, $\omega' \in \Omega'$), $\times_i \omega_i$, $\times_i q_i$ and δ represent respectively the component inputs, the component states, and the transition function of the base network η_{base}; and $g(\omega_1, \omega_2, ..., \omega_n)$ is the input mapping, considering an external coupling relation from external components or networks to the input of the network (as described in Example 1). Besides based on mean field conditions on dynamics homogeneity (cf. Definition 6), $\delta' \stackrel{def}{=} \delta_i$, where δ_i are the transition functions in the base network.

$\lambda'(\delta'(q'))$ is the *output transition function of the network*. Besides based on mean field conditions on dynamics homogeneity (cf. Definition 6), $\lambda' \stackrel{def}{=} \lambda_i$, where λ_i are the output functions in the base network and considering internal coupling relation from the component outputs of the network to the output of the network (as described in Example 1).

$\{S'_i\}$ is the *set of senders to the network* such that $S'_i = \{j \mid \omega'_i = \lambda'_j(\delta'_j(q'_j, \omega'_j)\}$.

Remark 2. In a *base network*, a *pool of states* of the components consists of *equivalent states* mapped into a single state of the lumped network. At dynamics level, for each transition, each pool of states in the base network matches a single state in corresponding lumped network. The mapping (or homomorphism) between state pools to single states can be generalized (instead of a simple average or summation) as a *census* of states in a pool, i.e., as counting the number of components in a particular state in the pool. Also, notice that the states in a pool are permutable (either using a summation, an average or a census mapping).

5 Mean field conditions explained by structure morphisms

5.1 Homogeneity and stability condition

When applied to a network, a system morphism is called a *structure morphism.* Let's take a simple example of a network of linear systems to expose the notion of structure morphism with respect to mean field conditions.

Example 2. A network of two 1-D linear components with identical structure (following homogeneity condition of mean field theory (cf. Definition 6)) consists of:

$$\begin{cases} q_1\text{'} = aq_1 \ + bx_1 \\ y_1 = cq_1 \end{cases} \quad \text{and} \quad \begin{cases} q_2\text{'} = aq_2 \ + bx_2 \\ y_2 = cq_2 \end{cases}$$

The stability condition of mean field theory (cf. Definition 6), based on feedback, can be represented by different homomorphisms and structures of the network as long as each component receives the same values, i.e., they have the same number of influencers of the same kind (or dynamic transition) through different feedback loops:

1. A network with feedback being the (same) average output to each component: $x_1 = x_2 = \frac{y_1 + y_2}{2} = \frac{cq_1 + cq_2}{2} = \frac{c(q_1 + q_2)}{2}$.
2. A network where components have self and independent feedback loops: $x_1 = \frac{y_1}{2}$ and $x_2 = \frac{y_2}{2}$, also leads to $x_1 = x_2 = \frac{c(q_1 + q_2)}{2}$.
3. A network where components have cross feedback loops: $x_1 = y_2$ and $x_2 = y_1$, also leads to $x_1 = x_2 = \frac{c(q_1 + q_2)}{2}$.

In all feedback loop cases, taking the homomorphism $h(q_1, q_2) = q_1 + q_2$, the state of the lumped network is given by: $q_1' + q_2' = a(q_1 + q_2) + b(x_1 + x_2) = a(q_1 + q_2) + 2bx_1 = a(q_1 + q_2) + bc(q_1 + q_2)$, thus leading to equation:

$$Q' = (a + bc)Q \tag{1}$$

With $Q = q_1 + q_2$.

In conclusion, based on different homogeneity and stability conditions, different structures lead to the same lumped network dynamics.

Remark 3. This simple example shows that the stability of the lumped network dynamics depends on the sign of parameters $a + bc$: if $a + bc > 0$, the system dynamics is exponentially growing, the system thus being unstable while for $a + bc < 0$, the system dynamics is stable.

Following previous assumptions, the relationship between the base and lumped model networks consists of:

Theorem 1. *If there exists a fixed point in the base network of linear systems, $\eta_{base} = (\delta, \lambda, \{S_i\})$, for a particular input segment $\omega \in \Omega$, there exists a fixed point in the lumped network of linear systems, $\eta_{lumped} = (\delta', \lambda', \{S_i'\})$, for the lumped input segment $\omega' \in \Omega'$ corresponding to $\omega \in \Omega$.*

Proof. For the sake of simplicity, let's take an all-to-all coupling. The proof can be easily extended to all other cases of Example 1, by considering the set of senders. The fixed point in the base network consists of $\omega = \lambda(\delta(q, \omega))$. In a lumped network averaging both states and inputs, the latter equation can be developed as follows:

$$\begin{array}{rcll} \omega & = & \lambda(\delta(q, \omega)) & \\ & = & \lambda(..., \delta_i(q_i, \omega_i), ...) & \text{based on base network structure (cf. Definition 7)} \\ & = & \lambda'(\frac{1}{n}\sum_{i=1}^{n} \delta_i(q_i, \omega_i)) & \text{based on lumped network structure (cf. Definition 8)} \\ & = & \lambda'(\delta_i(\frac{1}{n}\sum_{i=1}^{n} q_i, \frac{1}{n}\sum_{i=1}^{n} \omega_i)) & \text{based on linear system properties (cf. Definition 4)} \\ & = & \lambda'(\delta'(q', \omega')) & \end{array}$$

□

Remark 4. As shown by this theorem, the usual mean field assumption requiring the number of components n to be infinite is sufficient but not necessary.

5.2 Matrix expression of homomorphisms

For linear systems, the matrix expression of homomorphisms is a simple and constructive way to implement general systems morphisms. We provide here the conditions to fulfill for such homomorphism to hold.

Theorem 2. *From a matrix representation point of view, let $H : Q \rightarrow^{onto} Q$ be a linear mapping, then H is a homomorphism from a big system SYS to a small system SYS', if the following conditions hold (cf. Figure 3):*

1. *$H'A = AH$, being the state-to-state linear mapping (or state mapping for short) with A "large" compared to corresponding parameter A' of the small system being "small",*
2. *$HB = B'$, being the input-to-state linear mapping (or input mapping for short) with B "large" compared to corresponding parameter B' of the small system being "small",*
3. *$C'H = C$, being the state-to-output linear mapping (or output mapping for short) with C "large" compared to corresponding parameter C' of the small system being "small".*

Proof. Let us take the simple example of the network of two 1-D linear components (cf. Example 2). The matrix form of the linear operators consists of $A = \begin{bmatrix} a & 0 \\ 0 & a \end{bmatrix}$, $Q = \begin{bmatrix} q_1 \\ q_2 \end{bmatrix}$, $B = \begin{bmatrix} b & b \end{bmatrix}$, $X = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$, $C = \begin{bmatrix} c \\ c \end{bmatrix}$ and $Y = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$. Taking homomorphism $H \begin{bmatrix} 1 & 1 \end{bmatrix}$ and applying state linear mapping condition, we obtain $A'H = a \begin{bmatrix} 1 & 1 \end{bmatrix} = \begin{bmatrix} a & a \end{bmatrix}$ and $HA = \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} a & 0 \\ 0 & a \end{bmatrix} = \begin{bmatrix} a & a \end{bmatrix}$. So $A'H = HA$. Applying input linear mapping, $HB = \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} b & b \end{bmatrix} = [2b] = B'$. Applying output linear mapping, $C'H = [c] \begin{bmatrix} 1 & 1 \end{bmatrix} = \begin{bmatrix} c & c \end{bmatrix} = C$. Hence, the parameters of the lumped system consist of: $A' = [a]$, $B' = [2b]$, and $C' = [c]$, with $Q' = (a + bc)Q$, $Y' = Y = [y_1]$, and $X' = X = [x_1]$ through identity mapping $Id = [1]$. □

6 Lumping connected networks

Figure 8 shows the morphism between two base networks and two lumped networks.

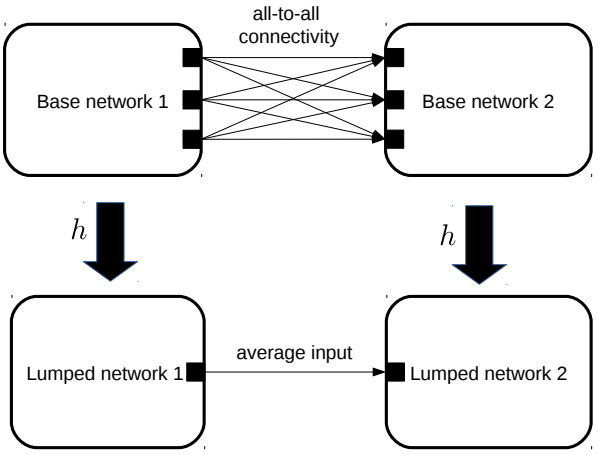

Figure 8: Morphism between two base networks and two lumped networks, an example of all-to-all coupling between the base networks.

The set of coupled base networks is closed under morphism, meaning that coupling base networks, a morphism holds when coupling corresponding lumped networks.

Theorem 3. *The dynamics of base linear networks is closed under morphism.*

Proof. Figure 9 shows the commutative diagram of the morphism between two base networks and two lumped networks, with the same number of components n. We prove here that the state transition of the target lumped network 2 depends on the output transition of the source network 1:

$$\begin{aligned}
\delta'(q'_1), \delta'(q'_2, \omega'_2) &= \delta'(\tfrac{1}{n}\textstyle\sum_{k=1}^{n} q_k^1), \delta'(\tfrac{1}{n}\sum_{k=1}^{n} q_k^2, \tfrac{1}{n}\sum_{k=1}^{n}\omega_k) \\
&= \delta'(\tfrac{1}{n}\textstyle\sum_{k=1}^{n} q_k^1), \delta'(\tfrac{1}{n}\sum_{k=1}^{n} q_k^2, \tfrac{1}{n}\sum_{k=1}^{n}\sum_{k\in S_i}\lambda_k(q_k)) \\
&= \delta'(\tfrac{1}{n}\textstyle\sum_{k=1}^{n} q_k^1), \delta'(\tfrac{1}{n}\sum_{k=1}^{n} q_k^2, \lambda'(\tfrac{2}{n}\sum_{k=1}^{n} q_k)) \qquad if\ all\ |S_i| = 2n \\
&= \delta'(q'_1), \delta'(q'_2, 2\lambda'(q')) \qquad \forall q' \in Q_1 \cup Q_2
\end{aligned}$$

□

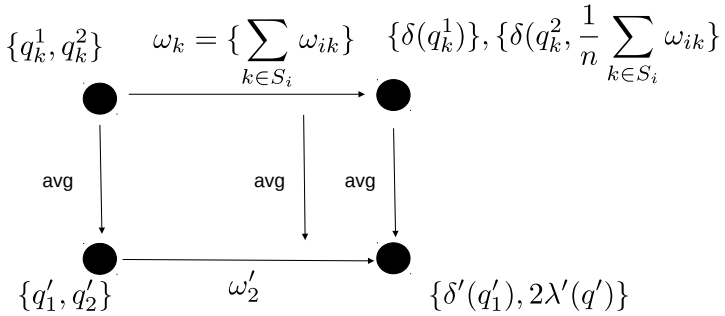

Figure 9: Commutative diagram of the morphism between two base networks and two lumped networks.

Figure 10 summarizes the large to small correspondence between the structures of base and lumped networks of networks.

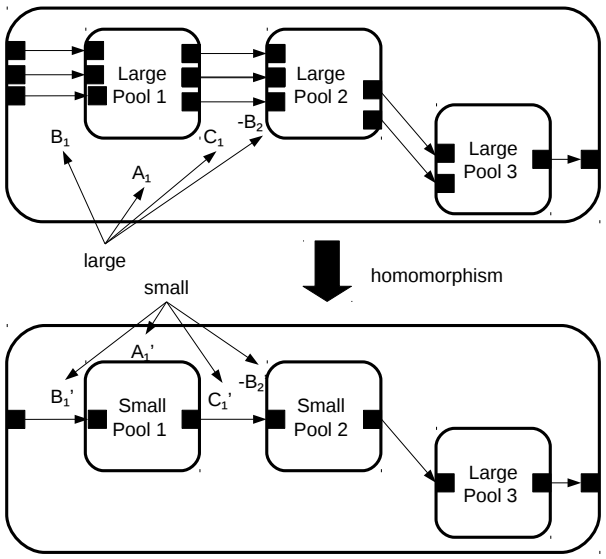

Figure 10: Couplings and lumping of a network of 3 networks. In the base network, the number of inputs (B), components and states in the state space (A), and outputs (C) is *large*. In the lumped network, corresponding numbers of lumped parameters $\{A', B', C'\}$ are *small*.

Notice that if the inputs to a linear system in a network consist of the sum of the output from its influencers, $\sum_{k \in S_i} \lambda_k(q_k)$, with linear and homogeneity properties it is obtained: $\lambda_k(\sum_{k \in S_i} q_k) = \lambda(q')$. Also, input is strictly positive if and only if $\sum_{k \in S_i} \lambda_k(q_k) > 0$, i.e., sensitive only to whether or not influencers send positive outputs $\lambda_k(\sum_{k \in S_i} q_k) = \lambda(q') > 0$ and if and only if $\sum_{k \in S_i} q_k > 0$, for state range strictly positive, and λ monotonically increasing. We will see hereafter that these conditions can be used to model network model connections.

6.1 Simple discrete formulation

Figure 11 describes the lumping of two connected pools of components with no internal couplings. The discrete time state dynamics of a neuron $i = 1, ..., n$ consists of

$$q_i(t+1) = q_i(t) + I_i(t)$$

Where I_i is the *input of component* i.

For a lumped network, the *lumped state* at time t, $Q(t)$, consists of the sum of the component states in corresponding base network, $Q(t) = \sum_{i=1}^{n} q_i(t)$, and the

lumped input at time t, $I(t)$, is the average of external inputs, $I(t) = \sum_{i=1}^{n} \frac{I_i(t)}{n}$. So the state of a lumped of:

$$Q(t+1) = Q(t) + I(t)$$

With $I^1(t)$ as output of the first pool and input to the second pool. Then,

$$Q^2(t+1) = Q^2(t) + Q^1(t)$$

Feeding back the output of a pool to itself, it is obtained for each component, $I_i(t) = \sum_{i=1}^{n} \frac{Q_i(t)}{n}$, so $I(t) = \sum_{i=1}^{n} \sum_{i=1}^{n} \frac{q_i(t)}{n} = Q(t)$. So, $Q(t+1) = Q(t)+Q(t)$ and

$$Q(t+1) = 2Q(t)$$

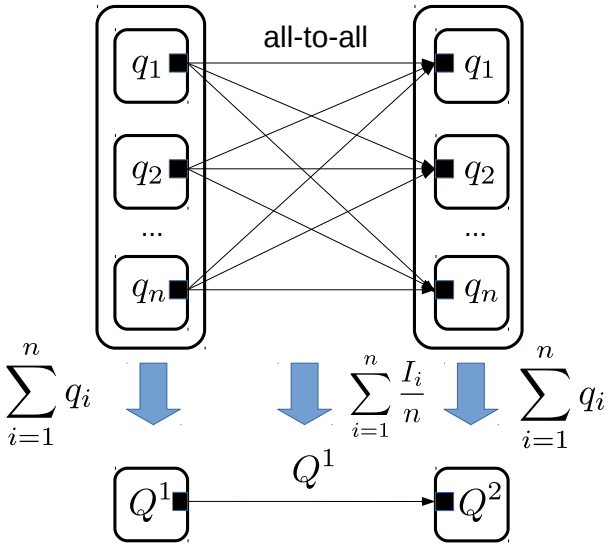

Figure 11: Simple discrete formulation of the lumping of two fully coupled networks.

This illustrates a *homomorphism* $h(\{q_i\}) = \sum_{i=1}^{n} q_i$. Starting with $q_i(t)$ the transition to the next state is $q_i(t+1) = q_i(t) + I_i(t) = q_i(t) + \sum_{i=1}^{n} \frac{Q_i(t)}{n}$. Applying h to both $\{q_i(t)\}$ and $\{q_i(t+1)\}$, we have $Q(t+1)$ and $Q(t) + \sum_{i=1}^{n} \sum_{i=1}^{n} \frac{Q_i(t)}{n} = 2Q(t)$ which match the lumped model transition $Q(t+1) = 2Q(t)$.

The solution of the lumped model is

$$Q(t) = Q(0)2^t$$

Remark 5. In neuronal pools, averaging the inputs can be conceived as averaging the firing rates of neurons when taking their inputs as firing rates.

6.2 Computational approximation

Based on previous simple discrete result, the *exact lumped state* can be exactly computed as $Q(t) = Q(0)2^t$ with uniformly connected pools and an large number of components. However, real network simulation frequently consists of non-uniformly coupled networks, of finite size, leading to an approximation of the dynamics of the lumped network. Figure 12 shows the simulation lumping approximation, $\frac{Q_{sim}(t)}{Q(t)}$, where $Q_{sim}(t)$ is the *simulated lumped state*. It can be seen that for a finite number of components, $n = 100$, the approximation of the lumped state is acceptable. The error increases after for smaller probabilities of connection. In a fully connected network a component samples all the states of other components. With, e.g., a probability $p = 30\%$, a component samples randomly 30% of all the components, at each step.

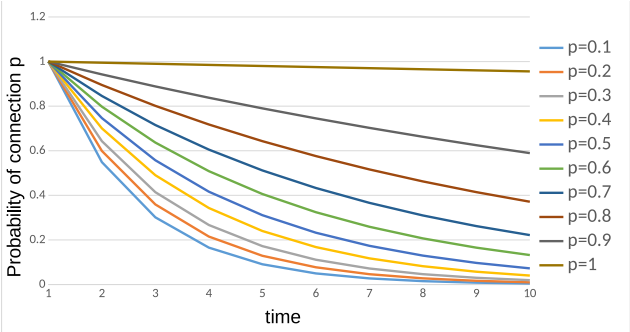

Figure 12: Lumping approximation as the ratio $\frac{Q_{sim}(t)}{Q(t)}$. Simulation are obtained according to the probability of connection p between the component pools, and with a number of components $n = 100$. Parameters of Equation 1 are $a = 1$ and $bc = -0.1$, so the exact state consists of $Q(t) = Q(0)0.9^t$, thus exhibiting a decreasing stable behavior.

We will see in the neural network application that stability and dynamics error accumulated at each state transition should not be confused.

7 Application to neuronal networks

7.1 Motivation

Figure 13 describes the lumping application onto models of brain regions. The base model consists of many details preventing simulation under reasonable execution times, while the lumped model abstracting the base model details aims to be simulatable.

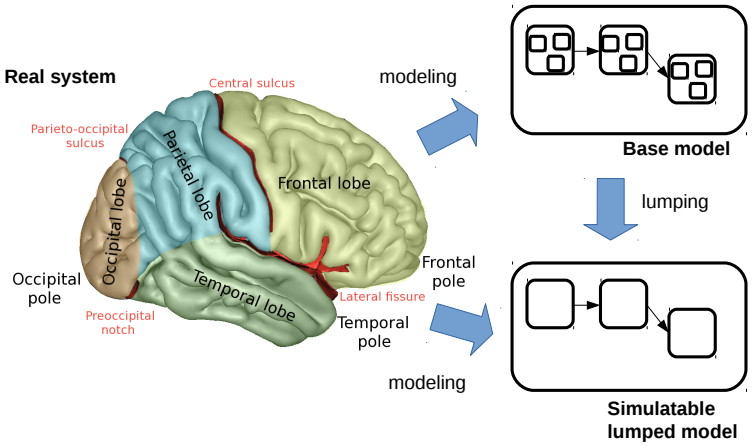

Figure 13: Modeling of the brain regions and model lumping for simulatability. Free picture of the brain lobes from Wikipedia.

Figure 14 presents a coupling between two brain regions that can be considered as coupling two neuronal networks with inhibitory external synaptic connections coming from another network (or brain region). Inside the networks, synaptic connections are considered to be excitatory. This assumption follows the one of balanced networks (Brunel 2000).

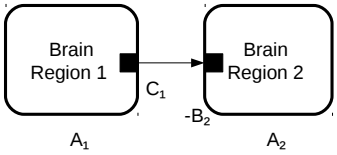

Figure 14: Couplings between two brain regions. The synaptic inputs received by Brain region 2 from Brain region 1 are inhibitory thus leading to matrix $C_1 - B_2$ for output-to-input mapping.

7.2 Neuron dynamics model

A well known model of neuron dynamics is the the Amari-Wilson-Cowan model (Amari 1972; Wilson and Cowan 1972; Wilson and Cowan 1973), slightly modified by taking external input current to the network as external inputs from another network:

$$\frac{dq_i}{dt} = aq_i + \sum_{j \in S_i} b_{ij} f(x_j) \tag{2}$$

Where q_i represents the *voltage of the neuron* i, $a \leq 0$ is the *leak parameter*, b_{ij} is a *synaptic weight* (with $b_{ij} < 0$ an *inhibitory synapse* and $b_{ij} > 0$ an *excitatory synapse*), x_j is the input voltage received from an influencing (external or internal to the network) neuron j in S_i, and f is a typical non-linear sigmoid function $f(x) = \frac{1}{2}(1 + tanh(gx))$, g a gain parameter (cf. Figure 15).

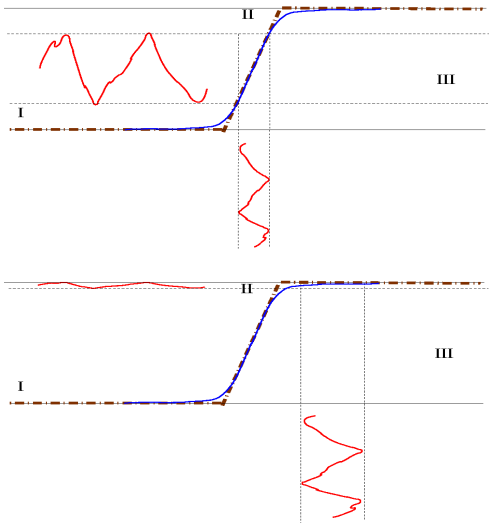

Figure 15: The sigmoidal shape of the function f and its effects on voltage fluctuations. Top: When neuron's voltage fluctuates around the inflection point of the sigmoid, and if the gain g is large enough fluctuations are amplified. Bottom: When the neuron's voltage fluctuates in the flat parts of the sigmoid (here, the saturated region) fluctuations are damped. Dashed red lines correspond to a piecewise linear approximation of the sigmoid, allowing to delimit regions I, II, III (legend and figure from (Cessac 2019)).

Lumping neurons consists of using the linear part of the sigmoid, with parameters $bcg = b$ for simplification, where c is the *signal attenuation on the axon*, b is the *pulse attenuation on synapses and dendrites.*

7.3 Computational approximation, dynamics and stability

Figure 16 describes the error dynamics in the lumping model. In stable linear systems, any two states converge to one state unless there is a break of the convergence because of :

1. The non linearities of the sigmoid,

2. Positive synaptic weights, b_{ji},

3. Random sampling couplings instead of a uniform ones in brain regions. Then, the dynamics could differ just a little bit from the average.

In all cases, lumping analytic results are not possible anymore and simulation is required to study the dynamics between brain regions.

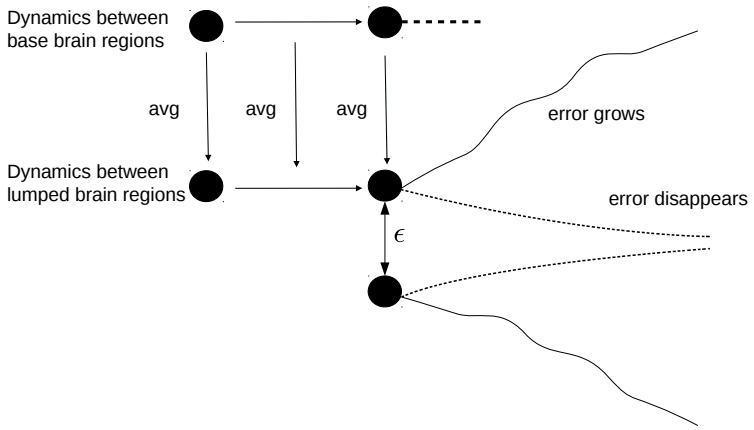

Figure 16: The error depends on the dynamics between base and lumped brain regions: Usually, the brain is a stable system where the error of the lumped model used for study should disappear. Sometimes, the brain turns unstable (as during epileptic crises).

8 Conclusion

Sufficient and necessary conditions have been proposed in Definition 6 for I/O general linear systems. These conditions proved to be implemented by system morphisms leading to fixed points in networks (cf. Theorem 1) and allowing lumping coupled networks (cf. Theorem 6). Compared to usual mean field conditions shown in Table 1, computational assumptions allowed a non infinite number of components in networks as well as different couplings (cf. Remarks 4 and **??**).

References

Amari, Shun-Ichi (1972). "Characteristics of random nets of analog neuron-like elements". In: *IEEE Transactions on Systems, Man, and Cybernetics* 5, pp. 643–657.

Arbib, Michael A (1972). *Theories of abstract automata*.

Arnold, André (1994). *Finite Transition Systems. International Series in Computer Science*.

Brunel, Nicolas (2000). "Dynamics of sparsely connected networks of excitatory and inhibitory spiking neurons". In: *Journal of computational neuroscience* 8.3, pp. 183–208.

Cessac, Bruno (Oct. 2019). "Linear response in neuronal networks: from neurons dynamics to collective response". In: *Chaos* 29.103105. DOI: 10.1063/1.5111803. URL: https://hal.inria.fr/hal-02280089.

El Boustani, Sami and Alain Destexhe (2009). "A master equation formalism for macroscopic modeling of asynchronous irregular activity states". In: *Neural computation* 21.1, pp. 46–100.

Faugeras, Olivier D, Jonathan D Touboul, and Bruno Cessac (2009). "A constructive mean-field analysis of multi population neural networks with random synaptic weights and stochastic inputs". In: *Frontiers in Computational Neuroscience* 3, pp. 1–28. ISSN: 1662-5188. DOI: 10.3389/neuro.10.001.2009. URL: http://dx.doi.org/10.3389/neuro.10.001.2009.

Harrison, Michael A (1969). *Lectures on linear sequential machines*. Tech. rep.

Ho, Yu-Chi (1992). *Discrete event dynamic systems: analyzing complexity and performance in the modern world*. IEEE.

Ivanov, E. (2013). *Investigation of abstract systems with inputs and outputs as partial functions of time*. Phd dissertation. Taras Shevchenko National University of Kyiv, Ukrain.

Klir, George J (1985). *Architecture of systems complexity*. Saunders, New York.

Mesarovic, M.D. and Y. Takahara (1975). *General Systems Theory: Mathematical Foundations*. LNCIS 116. Academic Press.

— (1989). *Abstract Systems Theory*. LNCIS 116. Springer.

Nykamp, Duane Q et al. (2017). "Mean-field equations for neuronal networks with arbitrary degree distributions". In: *Physical Review E* 95.4, p. 042323.

Ostojic, Srdjan (2014). "Two types of asynchronous activity in networks of excitatory and inhibitory spiking neurons". In: *Nature neuroscience* 17.4, p. 594.

Wilson, Hugh R and Jack D Cowan (1972). "Excitatory and inhibitory interactions in localized populations of model neurons". In: *Biophysical journal* 12.1, pp. 1–24.

— (1973). "A mathematical theory of the functional dynamics of cortical and thalamic nervous tissue". In: *Kybernetik* 13.2, pp. 55–80.

Wymore, Wayne (1967). *A mathematical theory of systems engineering*. Wiley.

Zadeh, L.A. and C.A. Desoer (1963). *Linear system theory: the state space approach*. McGraw-Hill series in system science. McGraw-Hill.

Zeigler, Bernard P, Alexandre Muzy, and Ernesto Kofman (2018). *Theory of Modeling and Simulation: Discrete Event & Iterative System Computational Foundations*. Academic press.