frontiers in Neuroinformatics

ORIGINAL RESEARCH
published: 24 May 2017
doi: 10.3389/fninf.2017.00034

# Integration of Continuous-Time Dynamics in a Spiking Neural Network Simulator

*Jan Hahne[1]*†, David Dahmen[2]†, Jannis Schuecker[2]†, Andreas Frommer[1], Matthias Bolten[1], Moritz Helias[2,3] and Markus Diesmann[2,3,4]*

[1] School of Mathematics and Natural Sciences, Bergische Universität Wuppertal, Wuppertal, Germany, [2] Institute of Neuroscience and Medicine (INM-6), Institute for Advanced Simulation (IAS-6), JARA BRAIN Institute I, Jülich Research Centre, Jülich, Germany, [3] Department of Physics, Faculty 1, RWTH Aachen University, Aachen, Germany, [4] Department of Psychiatry, Psychotherapy and Psychosomatics, Medical Faculty, RWTH Aachen University, Aachen, Germany

Contemporary modeling approaches to the dynamics of neural networks include two important classes of models: biologically grounded spiking neuron models and functionally inspired rate-based units. We present a unified simulation framework that supports the combination of the two for multi-scale modeling, enables the quantitative validation of mean-field approaches by spiking network simulations, and provides an increase in reliability by usage of the same simulation code and the same network model specifications for both model classes. While most spiking simulations rely on the communication of discrete events, rate models require time-continuous interactions between neurons. Exploiting the conceptual similarity to the inclusion of gap junctions in spiking network simulations, we arrive at a reference implementation of instantaneous and delayed interactions between rate-based models in a spiking network simulator. The separation of rate dynamics from the general connection and communication infrastructure ensures flexibility of the framework. In addition to the standard implementation we present an iterative approach based on waveform-relaxation techniques to reduce communication and increase performance for large-scale simulations of rate-based models with instantaneous interactions. Finally we demonstrate the broad applicability of the framework by considering various examples from the literature, ranging from random networks to neural-field models. The study provides the prerequisite for interactions between rate-based and spiking models in a joint simulation.

**Keywords: rate models, spiking neural network simulator, stochastic (delay) differential equations, waveform relaxation, parallelization, supercomputing**

OPEN ACCESS

***Edited by:***
*Daniel Gardner,*
*Weill Cornell Medical College,*
*United States*

***Reviewed by:***
*Marc De Kamps,*
*University of Leeds, United Kingdom*
*Nicholas Cain,*
*Allen Institute for Brain Science,*
*United States*

***Correspondence:**
*Jan Hahne*
*hahne@math.uni-wuppertal.de*

*† These authors have contributed equally to this work.*

***Received:*** *22 November 2016*
***Accepted:*** *01 May 2017*
***Published:*** *24 May 2017*

***Citation:***
*Hahne J, Dahmen D, Schuecker J, Frommer A, Bolten M, Helias M and Diesmann M (2017) Integration of Continuous-Time Dynamics in a Spiking Neural Network Simulator. Front. Neuroinform. 11:34. doi: 10.3389/fninf.2017.00034*

## 1. INTRODUCTION

Over the past decades, multiple strategies of neural network modeling have emerged in computational neuroscience. Functionally inspired top-down approaches that aim to understand computation in neural networks typically describe neurons or neuronal populations in terms of continuous variables, e.g., firing rates (Hertz et al., 1991; Schöner et al., 2015). Rate-based models originate from the seminal works by Wilson and Cowan (1972) and Amari (1977) and were introduced as a coarse-grained description of the overall activity of large-scale neuronal

networks. Being amenable to mathematical analysis and exhibiting rich dynamics such as multistability, oscillations, traveling waves, and spatial patterns (see e.g., Roxin et al., 2005), rate-based models have fostered progress in the understanding of memory, sensory and motor processes including visuospatial working memory, decision making, perceptual rivalry, geometric visual hallucination patterns, ocular dominance and orientation selectivity, spatial navigation, and movement preparation (reviewed in Coombes, 2005; Bressloff, 2012; Kilpatrick, 2015). On the brain scale, rate models have been used to study resting-state activity (Deco et al., 2011) and hierarchies of time scales (Chaudhuri et al., 2015). Ideas from functional network models have further inspired the field of artificial neuronal networks in the domain of engineering (Haykin, 2009).

In contrast, bottom-up approaches are motivated by the microscopic dynamics of individual neurons. Biophysically grounded spiking neuron models that simulate the time points of action potentials can explain a variety of salient features of microscopic neural activity observed *in vivo*, such as spike-train irregularity (Softky and Koch, 1993; van Vreeswijk and Sompolinsky, 1996; Amit and Brunel, 1997; Shadlen and Newsome, 1998), membrane-potential fluctuations (Destexhe and Paré, 1999), asynchronous firing (Brunel, 2000; Ecker et al., 2010; Renart et al., 2010; Ostojic, 2014), correlations in neural activity (Gentet et al., 2010; Okun and Lampl, 2008; Helias et al., 2013), self-sustained activity (Ohbayashi et al., 2003; Kriener et al., 2014), rate distributions across neurons (Griffith and Horn, 1966; Koch and Fuster, 1989; Roxin et al., 2011) and across laminar populations (Potjans and Diesmann, 2014), as well as resting state activity (Deco and Jirsa, 2012). Furthermore, in population-density approaches, statistical descriptions of neuronal populations neglect the identities of individual neurons and describe the dynamics of homogeneous populations in terms of probability densities (reviewed e.g., Deco et al., 2008). These approaches capture the time-dependent population activity enabling the investigation of phenomena like desynchronization (de Kamps, 2013) and computational properties of cortical circuits (Cain et al., 2016).

Simulation of rate-based models goes back to the works by Grossberg (1973), McClelland and Rumelhart (1981), Feldman and Ballard (1982), and the PDP group (Rumelhart et al., 1986). Various specialized tools have developed since then (O'Reilly, 2014), such as *PDP++* (McClelland and Rumelhart, 1989; O'Reilly et al., 2000), the *Neural Simulation Language* (Weitzenfeld et al., 2002), *emergent* (O'Reilly et al., 2012), the simulation platform *DANA* (Rougier and Fix, 2012), *TheVirtualBrain* (Sanz Leon et al., 2013), *Topographica* (Bednar, 2009) and the *Neural Field Simulator* (Nichols and Hutt, 2015). Similarly, efficient simulators for population-density approaches (*MIIND*: de Kamps et al., 2008, *DiPDE*: Cain et al., 2016) as well as spiking neural networks (see Brette et al., 2007 for a review) have evolved. The foci of the latter range from detailed neuron morphology (*NEURON*: Carnevale and Hines, 2006, *GENESIS*: Bower and Beeman, 2007) to an abstraction of neurons without spatial extent (*NEST*: Bos et al., 2015, *BRIAN*: Goodman and Brette, 2013). Such open-source software, combined with interfaces and simulator-independent languages (Davison et al., 2008; Djurfeldt et al., 2010, 2014), supports maintainability, reproducibility, and exchangeability of models and code, as well as community driven development. However, these tools are restricted to either rate-based or spike-based models only.

The situation underlines that bottom-up and top-down strategies are still mostly disjoint and a major challenge in neuroscience is to form a bridge between the spike- and rate-based models (Abbott et al., 2016), and, more generally, between the fields of computational neuroscience and cognitive science. From a practical point of view, a common simulation framework would allow the exchange and the combination of concepts and code between the two descriptions and trigger interaction between the corresponding communities. This is in particular important since recent advances in simulation (Djurfeldt et al., 2008; Hines et al., 2008; Kumar et al., 2010; Hines et al., 2011; Helias et al., 2012; Kunkel et al., 2014) and computing technology (Jülich Supercomputing Centre, 2015; Miyazaki et al., 2012) enable full-density bottom-up models of complete circuits (Potjans and Diesmann, 2014; Markram et al., 2015). In particular, it has become feasible to build spiking models (Schmidt et al., 2016) that describe the same macroscopic system as rate-based descriptions (Chaudhuri et al., 2015).

The relation between the different model classes is one focus of theoretical neuroscience. Assuming homogeneity across neurons, population-density methods reformulate the spiking dynamics as a dynamical equation for the probability density that captures the time evolution of the population activity (Knight, 1972; Gerstner, 1995, 2000). Under certain assumptions allowing the neglect of fluctuations in the input to neurons, a set of coupled differential equations for the population-averaged firing rate and membrane potential can be derived (Montbrió et al., 2015). For asynchronous irregular activity, input fluctuations can be taken into account in a diffusion approximation which leads to Fokker-Planck mean-field theory that can be used to determine homogeneous stationary state activities of spiking networks (Siegert, 1951; Brunel, 2000). The Fokker-Planck ansatz is, however, not limited to the population level, but can yield an heterogeneous stationary state firing rate across individual neurons in the network (Sadeh and Rotter, 2015). The dynamics of rate fluctuations around the background activity can be obtained using linear response theory on the population level (Brunel and Hakim, 1999) or the level of individual neurons (Lindner et al., 2005; Ostojic and Brunel, 2011; Trousdale et al., 2012; Grytskyy et al., 2013; Schuecker et al., 2015) yielding effective rate models on the population or single-neuron level. An alternative to linear response theory is given by moment expansions for mode decompositions of the Fokker-Planck operator (Mattia and Del Guidice, 2002, 2004; Deco et al., 2008).

An alternative derivation of rate-based dynamics aims at a closure of equations for synaptic currents of spiking networks in a coarse-graining limit by replacing spiking input with the instantaneous firing rate (Bressloff, 2012). Using field-theoretical methods (Buice and Chow, 2013) that were originally developed for Markovian network dynamics (Buice and Cowan,

2007; Buice et al., 2010) allows a generalization of this approach to fluctuations in the input (Bressloff, 2015).

In any case, the cascade of simplifications from the original spiking network to the rate-based model involves a combination of approximations which are routinely benchmarked in comparative simulations of the two models. A unified code base that features both models would highly simplify these validations rendering duplication of code obsolete.

In many cases rate models represent populations of spiking neurons. Thus, a hybrid model, employing both types of models in a multi-scale modeling approach, would contain a relatively large number of spiking neurons compared to the number of rate units. Despite the large size of the spiking network, the dynamics still features finite-size fluctuations (Ginzburg and Sompolinsky, 1994; Meyer and van Vreeswijk, 2002; Mattia and Del Guidice, 2004; Helias et al., 2013; Schwalger et al., 2016), and a downscaling of the network can generally not be performed without changing correlations (van Albada et al., 2015). Thus, it is crucial that a common simulation framework is able to handle real-sized spiking networks. In addition, the employed mean-field theories exploit the large number of neurons in biological networks. In fact, they are strictly valid only in the thermodynamic limit $N \to \infty$ (Helias et al., 2014). Therefore, in the above mentioned validation studies, the spiking networks are typically large. Thus, a common simulation framework should be optimized for spiking neurons rather than rate-based models.

Current spiking network simulators solve the neuronal dynamics in a distributed and parallel manner. They exploit the point-event like nature of the spike interaction between neurons, for example in event-based simulation schemes. Here, the term event-based denotes the update scheme of synapses. In contrast, for the neuron dynamics a globally time-driven update scheme is more beneficial due to the large total number of incoming events per neuron (Morrison et al., 2005). Moreover, a purely event-driven scheme cannot be efficiently distributed since it requires a central event queue (Hanuschkin et al., 2010). Spiking point-neuron models furthermore interact in a delayed fashion. The delays mimic the synaptic transmission and the propagation times along axons and dendrites. For the duration of the minimal delay $d_{\min}$ in a network, the dynamics of all neurons is decoupled. Hence, during $d_{\min}$, the neurons can be updated independently without requiring information from other neurons. Distributed processes therefore need to communicate spikes only after this period (Morrison et al., 2005). Due to considerable latencies associated with each communication, this scheme significantly improves performance and scalability of current simulators. Although rate-based models require communication of continuous state variables, the $d_{\min}$-communication scheme can be used if these interactions have a delay. However, many rate based-models consider instantaneous interactions between units (see Bressloff, 2012, and references therein), typically for analytical convenience in quasi-static situations where delays do not matter. A priori, these interactions require communication between units at each time step.

The present study provides the concepts and a reference implementation for the embedding of continuous-time dynamics in a spiking network simulator. To exploit existing functionality we choose as a platform the open source simulation code NEST (Gewaltig and Diesmann, 2007; Bos et al., 2015) which is a scalable software used on machines ranging from laptops to supercomputers. The software is used by a considerable user community and equipped with a Python interface, supports the construction of complex networks, and shields the neuroscientist from the difficulties of handling a model description, potentially including stochastic components, in a distributed setting (Morrison et al., 2005; Plesser et al., 2015). Within this framework we introduce an iterative numerical solution scheme that reduces communication between compute nodes. The scheme builds on the waveform-relaxation technique (Lelarasmee, 1982) already employed for gap-junction interactions (Hahne et al., 2015).

Our study begins with a brief review of numerical solution schemes for ordinary and stochastic (delay) differential equations in Section 2 and their application to neural networks in Section 2.2. Subsequently, we develop the concepts for embedding rate-based network models into a simulation code for spiking networks, adapt the waveform-relaxation scheme, and detail an extendable implementation framework for rate models in terms of templates (Section 2.3). In Section 3, different numerical schemes are evaluated as well as the scalability of our reference implementation. We illustrate the applicability of the framework to a broad class of network models on the examples of a linear network model (Grytskyy et al., 2013), a nonlinear network model (Sompolinsky et al., 1988; Goedeke et al., 2016), a neural field model (Roxin et al., 2005), and a mean-field description (Wong and Wang, 2006) of the stationary activity in a model of the cortical microcircuit (Potjans and Diesmann, 2014; Schuecker et al., 2017). Straight-forward generalizations are briefly mentioned at the end of the Results section, before the work concludes with the Discussion in Section 4. The technology described in the present article will be made available with one of the next major releases of the simulation software NEST as open source. The conceptual and algorithmic work is a module in our long-term collaborative project to provide the technology for neural systems simulations (Gewaltig and Diesmann, 2007).

## 2. MATERIALS AND METHODS

Rate-based single neuron and population models are described in terms of differential equations that often include delays and stochastic elements. Before we turn to the implementation of such models in computer code (Section 2.3) we review how such systems are mathematically solved and in particular how the stochastic elements are commonly interpreted with the aim to avoid an *ad-hoc* design. A stochastic differential equation (SDE) is defined by the corresponding stochastic integral equation. Let $W(t)$ denote a Wiener process, also called Standard Brownian motion. For the initial condition $X(t_0) = X_0$ an Itô-SDE in its most general form satisfies

$$X(t) = X_0 + \int_{t_0}^{t} a(s, X(s))\, ds + \int_{t_0}^{t} b(s, X(s))\, dW(s)\,, \qquad (1)$$

where the second integral is an Itô integral

$$\int_{t_0}^{t} Y(s)\, dW(s) := \lim_{n\to\infty} \sum_{i=1}^{n} Y_{i-1} \cdot (W_i - W_{i-1})$$

with $Y_i = Y(t_0 + i \cdot \frac{t-t_0}{n})$ and $W_i = W(t_0 + i \cdot \frac{t-t_0}{n})$. Alternatively, the second integral can be chosen as a Stratonovich integral, indicated by the symbol $\circ$,

$$\int_{t_0}^{t} Y(s) \circ dW(s) := \lim_{n\to\infty} \sum_{i=1}^{n} \frac{Y_{i-1} + Y_i}{2} (W_i - W_{i-1})$$

which approximates $Y(s)$ with the mid-point rule. In this case, the corresponding SDE is called a Stratonovich-SDE. We refer to Kloeden and Platen (1992) and Gardiner (2004) for a derivation and a deeper discussion on the differences between the two types of stochastic integrals. In the case of additive noise ($b(t, X(t)) = b(t)$) the Itô and Stratonovich integrals coincide. If furthermore the noise is constant ($b(t, X(t)) = \sigma = \text{const.}$) the integrals can be solved analytically

$$\begin{aligned}\int_{t_0}^{t} \sigma\, dW(s) &= \int_{t_0}^{t} \sigma \circ dW(s) = \lim_{n\to\infty} \sigma \cdot \sum_{i=1}^{n} (W_i - W_{i-1}) \\ &= \sigma \cdot (W(t) - W(t_0))\end{aligned}$$

with $W(t) - W(t_0) \sim \mathcal{N}(0, t - t_0)$. In the following, we focus on Itô-SDEs only.

The differential notation corresponding to Equation (1) reads

$$dX(t) = a(t, X(t))\, dt + b(t, X(t))\, dW(t) \tag{2}$$

and denotes an informal way of expressing the integral equation. Another widely used differential notation, called the Langevin form of the SDE, is mostly employed in physics. It reads

$$\frac{dX(t)}{dt} = a(t, X(t)) + b(t, X(t))\, \xi(t)\,, \tag{3}$$

where $\xi(t)$ is a Gaussian white noise with $\langle \xi(t) \rangle = 0$ and $\langle \xi(t)\xi(t') \rangle = \delta(t - t')$. Using the Fokker-Planck equation one obtains

$$\int_0^t \xi(t')dt' = W(t)\,,$$

which is a paradox, as one can also show that $W(t)$ is not differentiable (Gardiner, 2004, Chapter 4). Mathematically speaking this means that Equation (3) is not strictly well-defined. The corresponding stochastic integral equation

$$X(t) = X_0 + \int_{t_0}^{t} a(s, X(s))\, ds + \int_{t_0}^{t} b(s, X(s))\, \xi(s)\, ds\,,$$

however, can be interpreted consistently with Equation (1) as $dW(t) \equiv \xi(t)dt$.

## 2.1. Approximate Numerical Solution of SDEs

Similar to ordinary differential equations most stochastic differential equations cannot be solved analytically. Neuroscience therefore relies on approximate numerical schemes to obtain the solution of a given SDE. This section presents some basic numerical methods. Let $\Delta t$ denote the fixed step size, $t_k = t_0 + k\Delta t$ the grid points of the discretization for $k = 0, \ldots, n$, and $X_k$ the approximation for $X(t_k)$ obtained by the numerical method, at which $X_0$ is the given initial value. We consider systems of $N$ stochastic differential equations:

$$dX(t) = a(t, X(t))\, dt + b(t, X(t))\, dW(t) \tag{4}$$

with initial condition $X(t_0) = X_0$. Here, $X(t) = (X^1(t), \ldots, X^N(t))$ and $W(t) = (W^1(t), \ldots, W^N(t))$ denote $N$-dimensional vectors and $a : \mathbb{R}^N \to \mathbb{R}^N$ and $b : \mathbb{R}^N \to \mathbb{R}^N$ are $N$-dimensional functions. $W(t)$ is an $N$-dimensional Wiener process, i.e., the components $W^i(t)$ are independent and identically distributed.

### 2.1.1. Euler-Maruyama

The Euler-Maruyama method is a generalization of the forward Euler method for ordinary differential equations (ODE). Accordingly, it approximates the integrands in Equation (1) with their left-sided values. The update formula reads

$$X_{k+1} = X_k + a(t_k, X_k) \cdot \Delta t + b(t_k, X_k) \cdot \Delta W_k \tag{5}$$

with $\Delta W_k = W(t_{k+1}) - W(t_k) \sim \mathcal{N}(0, \Delta t)$ for $k = 0, \ldots, n-1$.

### 2.1.2. Semi-Implicit Euler

The (semi-)implicit Euler method is a generalization of the backwards Euler method for ODEs. The update formula reads

$$X_{k+1} = X_k + a(t_{k+1}, X_{k+1}) \cdot \Delta t + b(t_k, X_k) \cdot \Delta W_k. \tag{6}$$

The resulting scheme requires the solution of a system of nonlinear algebraic equations. Standard techniques for the solution of the system are Newton iteration and fixed-point iteration (Kelley, 1995). The method is sometimes called semi-implicit, because the function $b$ is still evaluated at $(t_k, X_k)$ instead of $(t_{k+1}, X_{k+1})$. However, a fully implicit Euler scheme for SDEs is not practicable (see Kloeden and Platen, 1992, Chapter 9.8) and thus the term implicit Euler usually refers to the semi-implicit method and is used in the following.

### 2.1.3. Exponential Euler

The exponential Euler method relies on the assumption that $a(t, X(t))$ consists of a linear part and a nonlinear remainder, i.e.,

$$a(t, X(t)) = A \cdot X(t) + f(t, X(t))$$

with $A \in \mathbb{R}^{N\times N}$. The idea is to solve the linear part exactly and to approximate the integral of the nonlinear remainder and the Itô

integral with an Euler-like approach. Variation of constants for Equation (4) yields

$$X(t) = e^{A(t-t_0)}X_0 + \int_{t_0}^{t} e^{A(t-s)} f(s, X(s))\, ds + \int_{t_0}^{t} e^{A(t-s)} b(s, X(s))\, dW(s)\,.$$

There are several versions of stochastic exponential Euler methods that differ in the approximation of the integrals. Unfortunately a standardized nomenclature to distinguish the methods is so far missing. The simplest approach, sometimes named stochastic Lawson-Euler scheme (e.g., Komori and Burrage, 2014), approximates the integrands with their left-sided values

$$X_{k+1} = e^{A\Delta t}X_k + e^{A\Delta t} f(t_k, X_k)\cdot\Delta t + e^{A\Delta t} b(t_k, X_k)\cdot\Delta W_k\,.$$

More advanced schemes approximate the nonlinear part by keeping $f(s, X(s))$ constant for $[t_0, t)$ and solving the remaining integral analytically

$$\int_{t_0}^{t} e^{A(t-s)} f(s, X(s))\, ds \approx \int_{t_0}^{t} e^{A(t-s)} f(t_0, X(t_0))\, ds = A^{-1}(e^{A(t-t_0)} - I)\cdot f(t_0, X(t_0))\,.$$

Here $I$ denotes the $N \times N$ identity matrix. The same technique can be used for the Itô integral

$$\int_{t_0}^{t} e^{A(t-s)} b(s, X(s))\, dW(s) \approx \int_{t_0}^{t} e^{A(t-s)} b(t_0, X(t_0))\, dW(s)\,. \tag{7}$$

For a single SDE, Shoji (2011) proposed a method where the remaining integral $\int_{t_0}^{t} e^{a(t-s)}\, dW(s)$ with $a \in \mathbb{R}$ is approximated by $\int_{t_0}^{t} \alpha\, dW(s)$, such that $\alpha \in \mathbb{R}$ is chosen to minimize the mean-square error. This results in a similar approximation as for the nonlinear part. Komori and Burrage (2014) adapted this approach for systems of SDEs. The scheme reads

$$X_{k+1} = e^{A\Delta t}X_k + A^{-1}(e^{A\Delta t} - I)\cdot f(t_k, X(t_k)) + \frac{1}{\Delta t}\cdot A^{-1}(e^{A\Delta t} - I)\cdot b(t_k, X_k)\cdot\Delta W_k\,.$$

Alternatively, calculating the variance of $X(t)$ within the approximation (7), amounts to (Adamu, 2011)

$$\mathrm{Var}\left(X(t)\right) = b(t_0, X(t_0))^2 \cdot \mathrm{Var}\left(\int_{t_0}^{t} e^{A(t-s)}\, dW(s)\right) = b(t_0, X(t_0))^2 \cdot A^{-1}\left(\frac{e^{2A(t-t_0)} - I}{2}\right).$$

The corresponding scheme reads

$$X_{k+1} = e^{A\Delta t}X_k + A^{-1}(e^{A\Delta t} - I)\cdot f(t_k, X(t_k)) + \sqrt{A^{-1}\left(\frac{e^{2A\Delta t} - I}{2}\right)}\cdot b(t_k, X_k)\cdot\eta_k \tag{8}$$

with $\eta_k \sim \mathcal{N}(0,1)$ and yields the exact solution of the system if $a(t, X(t)) = A\cdot X(t)$ and $b(t, X(t)) = \text{const.}$, since $X(t)$ has Gaussian statistics in this case (Risken, 1996). Therefore, in the following we exclusively employ (Equation 8) and just refer to it as the stochastic exponential Euler scheme. For more detailed reviews on the different stochastic exponential Euler methods we refer to Adamu (2011) and Komori and Burrage (2014).

## 2.2. Network of Rate Models

We now consider networks of $N$ rate-based units where each unit receives recurrent input from the network. The system fulfills the Itô-SDEs

$$\tau^i dX^i(t) = \left[-X^i(t) + \mu^i + \phi\left(\sum_{j=1}^{N} w^{ij}\psi\left(X^j(t - d^{ij})\right)\right)\right] dt + \sqrt{\tau^i}\sigma^i\, dW^i(t) \qquad i = 1, \ldots, N \tag{9}$$

with possibly nonlinear input-functions $\phi(x)$ and $\psi(x)$, connection weights $w^{ij}$, mean input $\mu^i$, and optional delays $d^{ij} \geq 0$. The corresponding Fokker-Planck equation shows that the parameter $\sigma^i \geq 0$ controls the variance of $X^i(t)$ and the time constant $\tau^i > 0$ its temporal evolution. For readability, from here on we omit unit indices for $\sigma, \tau, \mu$, and $d$. The considered class of rate models only contains additive noise. Therefore, as noted above, the system (Equation 9) can be written as Stratonovich-SDEs without the need for change in the employed numerical methods. For an illustrative purpose we explicitly state the different explicit solution schemes for the network dynamics (Equation 9) with $d = 0$. The Euler-Maruyama update step reads

$$X^i_{k+1} = X^i_k + \left[-X^i_k + \mu + \phi\left(\sum_{j=1}^{N} w^{ij}\psi\left(X^j_k\right)\right)\right]\frac{1}{\tau}\Delta t + \frac{1}{\sqrt{\tau}}\sigma\,\Delta W^i_k\,. \tag{10}$$

The implicit Euler update formula evaluates $X$ at $k+1$ instead of $k$ within the square brackets. This turns Equation 10 into a system of nonlinear algebraic equations for which the fixed-point iteration

$$X^{i,m+1}_{k+1} = \Phi(X^{1,m}_{k+1}, \ldots, X^{N,m}_{k+1}) \tag{11}$$

with initial value $X^{i,0}_{k+1} = X^i_k$ and the choice of

$$\Phi = \frac{X^i_k + \left[\mu + \phi\left(\sum_{j=1}^{N} w^{ij}\psi\left(X^{j,m}_{k+1}\right)\right)\right]\frac{1}{\tau}\Delta t + \frac{1}{\sqrt{\tau}}\sigma\,\Delta W^i_k}{1 + \Delta t/\tau} \tag{12}$$

yields the solution.

For nonlinear $\phi(x)$ or $\psi(x)$ the exponential Euler update step is

$$X^i_{k+1} = e^{-\Delta t/\tau}X^i_k + \left(1 - e^{-\Delta t/\tau}\right)\left[\mu + \phi\left(\sum_{j=1}^{N} w^{ij}\psi\left(X^j_k\right)\right)\right] + \sqrt{\frac{1}{2}(1 - e^{-2\Delta t/\tau})}\,\sigma\,\eta^i_k \tag{13}$$

with $\eta_k^i \sim \mathcal{N}(0,1)$. As $A = -I$ is a diagonal matrix, the exponential Euler scheme does not rely on a matrix exponential, but decomposes into $N$ equations with scalar exponential functions. Note that with a linear choice, $\phi(x) = \psi(x) = x$, the system of SDEs can be written in matrix notation

$$\tau dX(t) = \left[A \cdot X(t) + \mu\right] dt + \sqrt{\tau}\sigma dW(t) \qquad i = 1, \ldots, N \tag{14}$$

with $A = -I + W$ and $W = (w^{ij})_{N\times N}$. Here the stochastic exponential Euler scheme (Equation 8) yields the exact solution of the system.

The numerical schemes presented in Section 2.1 are developed for SDEs ($d = 0$), but can analogously be used for stochastic delay differential equations (SDDEs) ($d > 0$), if the delay $d$ is a multiple of the step size $\Delta t$. For the calculation of the approximation $X_{k+1}^i$ in time step $k + 1$ the recurrent input is then evaluated from $\frac{d}{\Delta t}$ steps earlier, i.e., from $X^j_{k-\frac{d}{\Delta t}}$ for the explicit methods.

## 2.3. Implementation in Spiking Network Simulation Code

This section describes the embedding of rate-based models (Section 2.2) in a simulation code for spiking neuronal networks. The Appendix (Section A.2) illustrates how to create, connect and record activity from rate models in our reference implementation.

The software architecture for rate models is based on existing concepts: Morrison et al. (2005) describe distributed buffers for the storage of delayed interactions and the technique to consistently generate random numbers in a distributed setting, and Hahne et al. (2015) introduce so called secondary events, that allow the communication of continuous state variables, like membrane potentials or rates, between neurons or rate units respectively. Events provide an abstraction layer on top of the MPI communication which allows the implementation of neuron models without explicit reference to MPI calls. Unlike primary events which are used to transmit the occurrence of spikes at discrete points in time, secondary events occur on a regular time grid. These concepts are designed to be compatible with the parallel and distributed operation of a simulation kernel for spiking neuronal networks, ensuring an efficient use of clusters and supercomputers (Helias et al., 2012). This allows researchers to easily scale up network sizes to more realistic number of neurons. The highly parallelizable structure of modern simulation codes for spiking neuronal networks, however, also poses restrictions on the utilizable numerical methods.

### 2.3.1. Parallelization and Restrictions

Parallelization for spiking neuronal networks is achieved by distributing neurons over compute nodes. Since the dynamics of spiking neurons (in the absence of gap junctions) is decoupled for the duration of the minimal synaptic delay $d_{\min}$ of the connections in the network, the states of the neurons can be propagated independently for this time interval. Thus, it is sufficient to specify solvers on the single-neuron level. The spike times, i.e., the mediators of interaction between neurons, are then communicated in steps of $d_{\min}$.

As a result of this structure the global connectivity of the network is unknown to the single neuron. The neuron object sends and receives events handled by an object on the compute node harboring the neuron termed network manager. However, the network manager only knows the incoming connections of the neurons on the compute node.

This poses restrictions on the use of implicit schemes. It is impossible to employ the implicit Euler scheme (Equation 6) with Newton iteration, which would require the simultaneous solution of a system of nonlinear algebraic equations with information distributed over all compute nodes. The use of the implicit Euler scheme with fixed-point iteration is however compatible with this structure. To this end, the scheme (Equation 6) needs to be formulated as a fixed-point iteration on the single-unit level (see Section 2.2) and the updated rates need to be communicated to the connected units after every iteration until some convergence criterion is met. The convergence of the fixed-point iteration is however only guaranteed if the scheme $\Phi$ is contractive (see e.g., Kelley, 1995, their Section 4.2), which poses restrictions on the employed step size $\Delta t$. Section 3.1 investigates if the implementation can gain stability or accuracy from using the implicit Euler method with fixed-point iteration and if the payoff is large enough to justify the additional effort of an iterative solution scheme.

The restricted knowledge of connectivity also limits the usage of the exponential Euler method. In the case of a linear rate model, we are unable to add the influence from all other rate units to the matrix $A$ in Equation (14), because most of these connections are unknown at the single-unit level. Therefore, we use the exponential Euler method with $A = -I$ resulting in the update formula (13). This also has the benefit of avoiding the need to numerically evaluate a general matrix exponential as $A$ is a diagonal matrix (see Section 2.2 for details).

### 2.3.2. Implementation

This section describes the additional data structure required for the implementation of rate-based models. While the naming convention refers to our reference implementation in the simulation software NEST, the employed algorithms and concepts are portable to other parallel spiking network simulators. As a result of the previous section and our analysis of the numerical schemes below (see Section 3.1) we restrict the discussion and our reference implementation to the exponential Euler method where we assume $A = -I$ and identify $\Delta t = h$ with $h$ denoting the global computation step size (Morrison et al., 2005). We have to distinguish the cases of connections with delay ($d > 0$) and connections without delay ($d = 0$). The former case is similar to spiking interaction: assuming a connection from unit $i$ to unit $j$, the rate of unit $i$ needs to be available at unit $j$ after $\frac{d}{h}$ additional time steps. This can be ensured if the delay of the connection is considered in the calculation of the minimal delay $d_{\min}$ that determines the communication interval. After communication the rate values are stored in a ring buffer of unit $j$ until they are due (Morrison and Diesmann, 2008). In the case of an instantaneous connection, the rate of unit $i$ at time $t_0$ needs to be known at time $t_0$ at the process which updates unit $j$ from $t_0$ to $t_0 + h$. Therefore, communication in every step is required for instantaneous rate connections, i.e., setting $d_{\min} = h$.

Due to the conceptual differences between instantaneous and delayed interactions (for the conceptual difference in the case of spiking interaction see Morrison and Diesmann, 2008) we employ two different connection types (`delay_rate_connection`, `rate_connection`) and associated secondary events (`DelayRateNeuronEvent`, `RateNeuronEvent`). This subdivision simplifies the discrimination of connections on the single-unit level, while still allowing for simultaneous use of instantaneous and delayed connections in the same rate model.

The large diversity of rate models (Equation 9) imposes a challenge for code maintenance and efficiency: Each combination of nonlinearities $\phi(x)$ and $\psi(x)$ constitutes its own model. All of these models can be implemented in the exact same way, except for the evaluation of the nonlinearities. A template class (`rate_neuron_ipn`) providing a base implementation for rate models of category (9) avoids code duplication. Nevertheless, we restrict the reference implementation to one nonlinearity per model. This keeps the zoo of rate models small and to our knowledge covers the majority of rate models.

The template rate model class is instantiated with an object that represents the nonlinearity. Being instantiated at compile time, this template solution does not incur additional overhead at run time compared to a solution using polymorphy (inheritance). A boolean class member `linear_summation` determines if the nonlinearity should be interpreted as $\phi(x)$ (true, default value) or $\psi(x)$ (false). The respective other function is assumed to be the identity function. The boolean parameter is evaluated in every update step of each unit. Deciding upon the type of nonlinearity at compile time would improve efficiency. In the present architecture this would, however, result in twice as many template instances for a given set of gain functions. With the future capabilities of code generation (Plotnikov et al., 2016) in mind it might be beneficial to elevate the constant boolean member object to a constant template parameter to allow compilers efficient preprocessing and at the same time profit from the code reliability achievable by modern C++ syntax. The present base implementation reduces the effort of creating a specific rate model of category (9) to the specification of an instance of the template class. Afterwards an actual rate model can be instantiated in a single line of code.

**Table 1** gives an overview of template-derived rate models of the reference implementation. These models serve as examples for customized rate models. Activity of rate units can be recorded using the `multimeter` and the recordable `rate`.

**TABLE 1 | Template-derived rate-based models.**

| Gain model | $\phi(x)$ or $\psi(x)$ |
|---|---|
| `lin_rate` | $g \cdot x$ with $g \in \mathbb{R}$ |
| `tanh_rate` | $\tanh(g \cdot x)$ with $g \in \mathbb{R}$ |
| `thresholdlin_rate` | $g \cdot (x-\theta) \cdot H(x-\theta)$ with $g, \theta \in \mathbb{R}$ |

*Gain functions of the rate-based models available in the NEST reference implementation. The name of a particular rate model is formed by* `<gain model>_ipn`. *The ending* `ipn` *indicates input noise, as the noise directly enters the r.h.s. of Equation (9). H denotes the Heaviside function.*

In addition to these template-derived models of category (9) the reference implementation also contains a rate model called `siegert_neuron`. This model, described by (30) in Section 3.3.4 is used for mean-field analysis of complex networks and constitutes a special case with respect to the recurrent input from the network. Firstly, it requires a numerically stable implementation of the Siegert formula (see Section A.1). Secondly, Equations (28) and (29) demonstrate that for this model the input rates are weighted by separate factors. Thus, for connections between instances of this model two different weights need to be specified and the rate model must be able to handle this anomaly. Therefore, the `siegert_neuron` does not derive from our base class `rate_neuron_ipn`, but constitutes an independent class. It comes with the connection type `diffusion_connection` providing the weight parameters. Section A.2 motivates the parameter names and shows the usage of the model in the reference implementation.

### 2.3.3. Reduction of Communication Using Waveform-Relaxation Techniques

Instantaneous connections between rate-based models require communication after every time step, thus in intervals of the global computation step size $h$. This requires setting $d_{\min} = h$ and impairs the performance and scalability, especially on supercomputers where communication is particularly expensive, because it is associated with a considerable latency. Therefore, for simulations with instantaneous connections we additionally study an iterative approach based on waveform-relaxation techniques that arrives, after convergence, at the same results as the standard approach, but allows us to use communication on a coarser time grid. Originally waveform-relaxation methods were developed (Lelarasmee, 1982) and investigated (see e.g., Miekkala and Nevanlinna, 1987) for ODEs. More recently they have also been analyzed for SDEs (Schurz and Schneider, 2005) and successfully applied to large systems of SDEs (Fan, 2013). With respect to simulations of rate-based models with instantaneous connections the basic idea is to solve the dynamics of the single units independently for the duration of $T = \alpha h$ with $\alpha \geq 2$ by treating the influence of the other units as known over this period of time. The solution of the original SDEs is determined by iteratively solving the decoupled SDEs:

$$\begin{aligned} dX^{1,m} &= a(t, X^{1,m}, Z^{1,m-1})\,dt + b(t, X^{1,m}, Z^{1,m-1})\,dW, \\ &\vdots \\ dX^{i,m} &= a(t, X^{i,m}, Z^{i,m-1})\,dt + b(t, X^{i,m}, Z^{i,m-1})\,dW, \\ &\vdots \\ dX^{N,m} &= a(t, X^{N,m}, Z^{N,m-1})\,dt + b(t, X^{N,m}, Z^{N,m-1})\,dW, \end{aligned} \tag{15}$$

where in the $m$-th iteration $Z^i = (X^1, \ldots, X^{i-1}, X^{i+1}, \ldots, X^N)$ is based on the solution of the $m-1$-th iteration and hence acts as a given input to the $i$-th system. The solutions improve step by step with each iteration. Schurz and Schneider (2005) demonstrate the convergence of the method for SDEs under mild assumptions on $X$, $a$ and $b$. For our specific application, systems of rate-based models with $b = \text{const.}$,

**TABLE 2 | Parameters of the waveform-relaxation algorithm.**

| Parameter name | Type | Default | Description |
|---|---|---|---|
| `use_wfr` | `bool` | `true` | Boolean parameter to enable (`true`) or disable (`false`) the use of the waveform-relaxation technique. If disabled and any rate-based units (or neurons supporting gap junctions) are present, communication in every step is automatically activated ($d_{\min} = h$). |
| `wfr_comm_interval` | `double` | 1.0 ms | Instantaneous rate connections (and gap junctions) contribute to the calculation of the minimal network delay with $\min(d_{\min}, \texttt{wfr\_comm\_interval})$. This way the length of the iteration interval of the waveform relaxation can be regulated. |
| `wfr_tol` | `double` | $10^{-4}$ | Convergence criterion for waveform relaxation. The iteration is stopped if the rates of all units change less than `wfr_tol` from one iteration to the next. |
| `wfr_max_iterations` | `int` | 15 | Maximum number of iterations performed in one application of the waveform relaxation. If the maximum number of iterations has been carried out without reaching the accuracy goal the algorithm advances system time and the reference implementation issues a warning. Additional speed-up in the simulation of rate-based units can only be achieved by `wfr_max_iterations` $< d_{\min}/h$. |
| `wfr_interpolation_order` | `int` | 3 | This parameter is exclusively used for gap junctions (see Hahne et al., 2015, their Section 2.1.2) and has no influence on the simulation of rate-based models. |

*The different parameters of the waveform-relaxation algorithm together with their C++ data-type, default value, and a brief description.*

the influence of the stochastic part on the convergence is even less critical. Based on our previous results in the neuroscience context (Hahne et al., 2015), the method converges within only a couple of iterations, due to the weak coupling of the SDEs. For more details on waveform-relaxation methods and their application in the neuroscience context we refer to Hahne et al. (2015).

As outlined above, in a simulator for spiking neuronal networks the minimal delay $d_{\min}$ in the network defines the communication interval. By employing the waveform-relaxation method with $T = d_{\min}$ we retain this communication interval for simulations with instantaneous rate connections. To control the iteration interval $T$ of the waveform-relaxation method, instantaneous connections contribute to the calculation of the minimal delay with an arbitrary user specified value given by the parameter `wfr_comm_interval` (see **Table 2**). Consequently the actual communication interval for waveform relaxation then is $T = \min\left(d_{\min}, \texttt{wfr\_comm\_interval}\right)$.

**Figure 1B** illustrates the concept of the iterative approach in contrast to the standard procedure in panel A. The iterative approach requires the repeated solution of all time steps in the communication interval and converges to the solution obtained with the standard approach (**Figure 1A**). The iteration terminates when a user chosen convergence tolerance `wfr_tol` (see **Table 2**) is met. If the method needs less than $T/h$ iterations, the approach reduces the overall number of communications required to obtain the solution. In conclusion, the avoidance of communication in every step comes for the price of additional computational load.

The coupling of neurons via gap junctions is instantaneous and continuous in time and thus constitutes a very similar problem to the rate dynamics. In order to combine gap junctions with spiking dynamics Hahne et al. (2015) already devised an iterative technique based on waveform-relaxation techniques and described a suitable framework. This framework can also

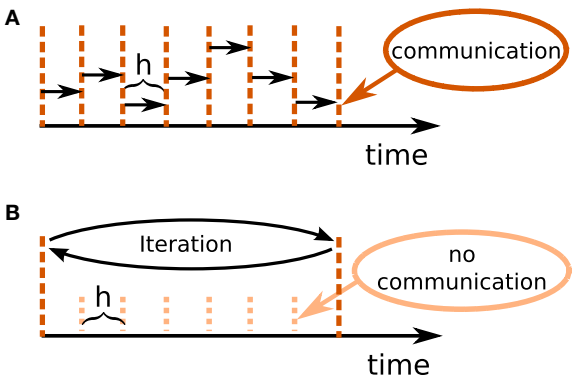


**FIGURE 1 | Different communication strategies for distributed simulations.** Distance between neighboring dotted orange lines indicates computation time step of size $h$. Distance between neighboring dashed red lines symbolizes one communication interval where rates (and other events like spike events) are communicated at the end of the interval. **(A)** Standard solution for rate-based models: rates are communicated in every time step. **(B)** Iterative approach using waveform relaxation: rates are communicated only after $\frac{T}{h}$ steps and the entire interval is solved repeatedly.

be employed for the simulation of rate-based models with instantaneous connections. The dynamics of a neuron model supporting gap junctions is solved with an adaptive step-size ODE-solver, routinely carrying out several steps of the employed numerical method within one global computation time step $h$. The communication of a cubic interpolation of the membrane potential provides the solver with additional information, resulting in a more accurate solution than the one obtained from the standard approach. For rate-based models this additional benefit cannot be gained: The combination of an iterative method with an adaptive step-size solver is not applicable to SDEs, where the noise in each time step constitutes a random number. However, an iterative approach with fixed

step size $\Delta t = h$ is applicable, as long as we ensure that the random noise applied to the units remains the same in every iteration. In Section 3.2 we investigate the performance of the iterative (**Figure 1B**) and the standard approach (**Figure 1A**) with a focus on large network simulations on supercomputers. In our reference implementation waveform relaxation can be enabled or disabled by a parameter `use_wfr`. Note that in the traditional communication scheme for spiking neuronal networks (Morrison et al., 2005) the first communication occurs earliest at the end of the first update step. Therefore, in the absence of waveform relaxation, the initial input to units from the network is omitted.

**Table 2** summarizes the parameters of our reference implementation of the waveform-relaxation technique. A subset (`wfr_interpolation_order`, `wfr_max_iterations`, `wfr_tol`) was previously introduced by Hahne et al. (2015), but we rename them here to arrive at more descriptive names. The remaining parameters (`use_wfr`, `wfr_comm_interval`) result from the generalization to rate-based models.

## 3. RESULTS

In the following, we assess the accuracy and stability of the different numerical solution schemes and benchmark the performance of the reference implementation on large-scale machines, with special focus on scalability and the comparison between the standard solution and the iterative approach using waveform relaxation for simulations with instantaneous connections. The iterative approach is only discussed with respect to efficiency, as the iteration always converges against the results of the standard approach within only a couple of iterations (for details see Section 2.3.3). The remainder of the section illustrates the application of the simulation framework to a selection of prominent problems in the neuroscientific literature.

### 3.1. Stability and Accuracy of Integration Methods

In this section we investigate numerical methods (see Section 2.1) for the solution of SDEs that can be employed to solve the dynamics of rate-based units. We analyze the accuracy and stability of the different numerical methods to choose the best-suited method for application in a distributed simulation scheme of a spiking network simulation code. The analysis only covers methods compatible with a spiking neural network simulator, namely (i) the Euler-Maruyama method, (ii) the implicit Euler method solved with a parallelizable fixed-point iteration, and (iii) the exponential Euler method where the linear part is restricted to $-I$, in the following called scalar exponential Euler. The distributed representation of the global connectivity of the network rules out both the regular exponential Euler method with a nondiagonal matrix $A$ and the implicit Euler method solved with Newton iteration (see Section 2.3.1 for details).

Consider an exactly solvable network of $N$ linear rate units with $\mu = 0$ (see also Section 2.2):

$$\tau dX(t) = A \cdot X(t)\, dt + \sqrt{\tau}\sigma\, dW(t)\,. \tag{16}$$

The exact solution of this system of SDEs coincides with the regular exponential Euler scheme and involves a matrix exponential and a matrix square root (Equation 8). We analyze two test cases, i.e., two different choices of $A$, to demonstrate different stability constraints. First, an all-to-all connected network with inhibitory connections of weight $w^{ij} = \frac{-1}{\sqrt{N}}$ and hence $A = -I + \frac{-1}{\sqrt{N}} \cdot \mathbb{1}$, with $\mathbb{1}$ denoting a $N \times N$ all-ones matrix (Cichocki et al., 2009). Second, a sparse balanced excitatory-inhibitory network where the number of excitatory units is four times larger than the number of inhibitory units. In this network, each unit receives input from a fixed number of $0.8 \cdot p \cdot N$ excitatory and $0.2 \cdot p \cdot N$ inhibitory randomly chosen source units with connection probability $p$ and connection weights $\frac{1}{\sqrt{N}}$ and $\frac{-4}{\sqrt{N}}$, respectively. In the following we refer to the test cases as inhibitory all-to-all and sparse balanced e/i test case.

First we turn to the accuracy analysis. Although the exact solution of Equation (16) cannot be obtained with a distributed representation of $A$ we can compute it using methods for numerical matrix computations implemented in MATLAB or Python (both provide an implementation of the same state-of-the-art algorithms, see Al-Mohy and Higham, 2009; Deadman et al., 2012). This way we obtain the exact solution within the accuracy of floating point numerics. This is the reference for computing the root mean square error (RMSE) of the different approximate methods. To employ the root mean square error in the context of stochastic differential equations we determine the reference solution for every tested step size and use the same random numbers for both, the reference solution and the approximative schemes.

**Figure 2** shows the root mean square error of the different numerical schemes for the two test cases with $N = 400$ units. In both test cases all investigated methods with decreasing step size converge toward the exact solution with convergence order 1, which is consistent with the established theory for SDEs with additive noise (Kloeden and Platen, 1992). **Figure 2A** shows the results for the inhibitory all-to-all test case. Here all three methods require a step size $\Delta t \leq 0.1$ to deliver reliable results. The implicit Euler scheme solved with fixed-point iteration even requires a step size $\Delta t \leq 0.05$. Within the stable region $\Delta t \leq 0.1$ the scalar exponential Euler yields more accurate results than the two other methods. **Figure 2B** shows the results for the sparse balanced e/i test case. Here all three methods achieve almost identical accuracy for $\Delta t \leq 0.1$ and stability problems only occur for the Euler-Maruyama method for step sizes $\Delta t > 0.5$. For step sizes $\Delta t > 0.1$ the implicit Euler method is more accurate than the other two methods.

To understand the stability issues shown in **Figure 2** we turn to stability analysis. In the following we assume that $A$ is diagonalizable, i.e., $A = T^{-1}DT$ with $T = (t^{ij})_{N\times N} \in \mathbb{C}^{N\times N}$ and $D = \mathrm{diag}(\lambda_1, \ldots, \lambda_N)$, and transform the system of SDEs with $Z(t) = T\,X(t)$. It follows

$$\tau dZ(t) = D \cdot Z(t)\, dt + \sqrt{\tau}\sigma\, T\, dW(t)$$

and $Z(t_0) = T\,X_0$. The transformed system consists of $N$ equations of the form

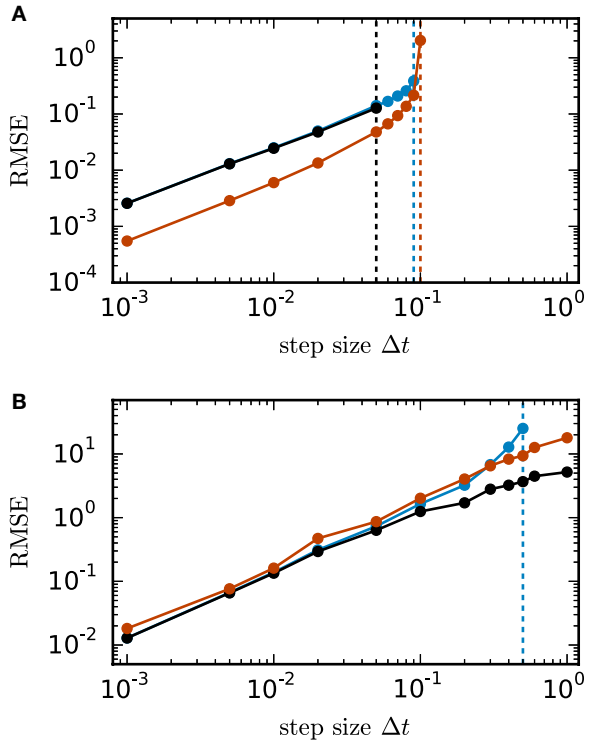


**FIGURE 2 | Accuracy of numerical methods for two networks of linear rate units.** RMSE $= \sqrt{\frac{1}{N(t_n - t_0)} \sum_{i=1}^{N} \sum_{j=1}^{n} (X_j^i - \hat{X}_j^i)^2}$ of the solution $X$ obtained by the approximate solvers (blue curve: Euler-Maruyama method, black curve: implicit Euler method solved with fixed-point iteration, red curve: scalar exponential Euler method) with respect to the reference solution $\hat{X}$ as a function of step size in double logarithmic representation. The respectively colored vertical lines mark the largest tested step size for which the corresponding methods deliver a solution with RMSE $\leq 10^{10}$. RMSE computed over 200.0 ms of biological time. **(A)** Inhibitory all-to-all test case. Network parameters: $N = 400$, $\mu = 0$, $\sigma = 10$ and $\tau = 1$ ms. **(B)** Sparse balanced e/i test case. Network parameters: $N = 400$, $p = 0.2$, $\mu = 0$, $\sigma = 10$ and $\tau = 0.5$ ms.

$$\tau\, dZ^i(t) = \lambda_i \cdot Z^i(t)\, dt + \sum_{j=1}^{N} \sqrt{\tau}\sigma t^{ij}\, dW^j(t) \qquad i = 1, \ldots, n \tag{17}$$

that depend on the eigenvalues $\lambda_i$ of $A$ and are pairwise independent except for the common contributions of the Wiener processes $W^j(t)$. In the following we only consider networks with bounded activity which requires eigenvalues $\lambda_i \in \mathbb{C}$ with negative real part $\mathrm{Re}(\lambda_i) < 0$. The solution of the $i$-th transformed equation then satisfies

$$|Z^i(t) - \tilde{Z}^i(t)| = e^{\lambda_i (t - t_0)/\tau} |Z_0^i - \tilde{Z}_0^i| < |Z_0^i - \tilde{Z}_0^i| \tag{18}$$

for two different initial values $Z_0^i$ and $\tilde{Z}_0^i$. It is a desirable stability criterion that a numerical method applied to Equation (16) conserves this property. This requirement is closely related to the concept of A-stability for SDEs (see Kloeden and Platen, 1992, Chapter 9.8) and A- respectively B-stability for ODEs (Hairer and Wanner, 1991). To derive stability conditions for the numerical schemes, we apply one update step ($t_0$ to $t_1 = t_0 + \Delta t$) of the methods to Equation (17) and investigate under which circumstances the property $|Z_1^i - \tilde{Z}_1^i| < |Z_0^i - \tilde{Z}_0^i|$ is conserved. Here $Z_1^i$ denotes the approximation to the exact solution $Z^i(t_1)$ obtained by the numerical scheme. A straight forward calculation shows that the Euler-Maruyama method retains the property if $|1 + \lambda_i \cdot \Delta t/\tau| < 1$ holds. We conclude that the Euler-Maruyama scheme is stable for $\max_{\lambda_i} \zeta_{\mathrm{EM}}(\lambda_i) < 1$ with $\zeta_{\mathrm{EM}}(\lambda_i) = |1 + \lambda_i \cdot \Delta t/\tau|$. The scalar exponential Euler method demands a splitting of $A = -I + W$ into $-I$ and $W$. Here the stability condition is derived from the modified transformed system

$$\tau\, dZ^i(t) = (-1 + \tilde{\lambda}_i) \cdot Z^i(t)\, dt + \sum_{j=1}^{N} \sqrt{\tau}\sigma t^{ij}\, dW^j(t) \quad i = 1, \ldots, n \tag{19}$$

where $\tilde{\lambda}_i$ are the eigenvalues of the matrix $W$. The scalar exponential Euler method conserves the property (Equation 18) if $|e^{-\Delta t/\tau} + \tilde{\lambda}_i \cdot (1 - e^{-\Delta t/\tau})| < 1$. We conclude that the scalar exponential Euler scheme is stable for $\max_{\lambda_i} \zeta_{\mathrm{EXP}}(\lambda_i) < 1$ with $\zeta_{\mathrm{EXP}}(\lambda_i) = |1 + \lambda_i \cdot (1 - e^{-\Delta t/\tau})|$.

The implicit Euler method solved with fixed-point iteration is stable, given the convergence of the fixed-point iteration. For the transformed system the employed fixed-point iteration on the single-unit level (Equation 12) reads

$$\Phi(Z_{k+1}^{i,m}) = \frac{1}{1 + \Delta t/\tau} (Z_k^i + \tilde{\lambda}_i Z_{k+1}^{i,m}\, \Delta t/\tau + \sum_{j=1}^{N} \frac{1}{\sqrt{\tau}} \sigma t^{ij} \Delta W_k^j).$$

It converges if the scheme $\Phi$ is contractive, i.e., if the inequality

$$|\Phi(Z_{k+1}^{i,0}) - \Phi(\tilde{Z}_{k+1}^{i,0})| \overset{!}{<} |(Z_{k+1}^{i,0} - \tilde{Z}_{k+1}^{i,0})|$$

holds for two different initial values $Z_{k+1}^{i,0}$ and $\tilde{Z}_{k+1}^{i,0}$. It follows that the fixed-point iteration converges if $|\frac{\Delta t/\tau}{1 + \Delta t/\tau} \tilde{\lambda}_i| < 1$. Thus, the implicit Euler method solved with fixed-point iteration is stable if $\max_{\lambda_i} \zeta_{\mathrm{IE}}(\lambda_i) < 1$ with $\zeta_{\mathrm{IE}}(\lambda_i) = \frac{\Delta t/\tau}{1 + \Delta t/\tau} |\lambda_i + 1|$. Hence for all investigated methods the stability depends on the eigenvalues of the matrix $A$, the time constant $\tau$ and the step size $\Delta t$. To conclude restrictions on the step size $\Delta t$ we analyze the eigenvalues of $A$ for our examples.

For the inhibitory all-to-all test case we determine the eigenvalues $\lambda_1 = -1 - \sqrt{N}$ and $\lambda_2 = \ldots = \lambda_N = -1$ of the matrix $A = -I + \frac{-1}{\sqrt{N}} \cdot \mathbb{1}$ analytically. It follows that the Euler-Maruyama scheme satisfies the stability criterion for $\Delta t \leq \frac{2\tau}{\sqrt{N} + 1}$, the scalar exponential Euler method demands $\Delta t \leq -\tau \cdot \ln(\frac{\sqrt{N} - 1}{\sqrt{N} + 1})$ and the implicit Euler method with fixed-point iteration requires $\Delta t \leq \frac{\tau}{\sqrt{N} - 1}$. For the example of 400 units with $\tau = 1$ in **Figure 2A** this yields step size restrictions of $\Delta t \leq \frac{2}{21}$ for the Euler-Maruyama method, $\Delta t \leq -\ln(\frac{19}{21}) \approx 0.1$ for the scalar exponential Euler method and $\Delta t \leq \frac{1}{19}$ for the implicit Euler method. This is consistent with the numerically obtained result (see vertical lines). For all methods the stability criterion implies that the step size $\Delta t$ needs to be reduced with increasing network size $N$ or decreasing time constant $\tau$.

This fully connected network, however, constitutes the worst case test for the class of rate-based models (Equation 9), as the absolute value of the negative eigenvalue quickly increases with the number of units $N$. Our second test case, the sparse balanced e/i network does not suffer from this problem (Rajan and Abbott, 2006), as it is a perfectly balanced network of excitatory and inhibitory units. In a scaling of the connection weights as $\frac{1}{\sqrt{N}}$, the spectral radius of $A$ and therefore the subsequent stability analysis is independent of $N$. Here the stability analysis is more complicated, as most of the eigenvalues of $A$ are complex and need to be computed numerically.

**Figure 3A** shows the eigenvalues of $A$ for a network of 2000 units. **Figure 3B** demonstrates that for this test case the scalar exponential Euler method and the implicit Euler method are stable regardless of the step size $\Delta t$. For the Euler-Maruyama the step size is restricted to $\Delta t < 1.2\tau$. This is again consistent with the results obtained in **Figure 2B**, where $\tau = 0.5$ and therefore the stability criterion of the Euler-Maruyama method yields $\Delta t < 0.6$.

Random inhibition-dominated networks exhibit characteristics of both examples. First the matrix $A$ contains a real eigenvalue $\lambda_1 = -1 - \alpha\sqrt{N}$ which scales with the network size, however, with a proportionality constant $0 < \alpha < 1$ which is reduced compared to the fully connected inhibitory network and determined by the sparseness and the imbalance between excitation and inhibition. Secondly, the matrix $A$ contains eigenvalues which constitute a cloud in the complex plane that is determined by the randomness of the connectivity. For these random networks $\lambda_{\max} = \text{argmax}_{\lambda_i}\, \zeta(\lambda_i)$ is a real eigenvalue. **Figure 4** shows the step size restrictions of the different numerical methods with respect to the absolute value of $\lambda_{\max}$. For $|\lambda_{\max}| < 2$ the scalar exponential Euler methods and the implicit Euler are stable regardless of the step size $\Delta t$. Starting at $|\lambda_{\max}| \geq 2.8$ the scalar exponential Euler is more stable than the implicit Euler method solved with fixed-point iteration. With increasing $|\lambda_{\max}|$ the step size restrictions of the scalar exponential Euler method converges against the step size restriction of the Euler-Maruyama method.

Based on the results in this section we employ the scalar exponential Euler to solve rate-based model dynamics (Equation 9) in our reference implementation. **Figure 4** demonstrates that it is the most stable algorithm compatible with the constraints of the distributed simulation scheme for spiking neural networks. Furthermore, the results in **Figure 2** indicate that it is the most accurate method in the case of an all-to-all connected network with inhibitory connections. For the sparse balanced excitatory-inhibitory network the analysis exhibits an accuracy similar to the implicit Euler method. However, the solution of the implicit Euler method with fixed-point iteration requires the application of an iterative scheme in each single time step with communication between the units after every iteration. This algorithm is therefore more time consuming than the scalar exponential Euler scheme. Besides the choice of method the analysis in this section indicates that numerical stability is an issue for all tested methods depending on step size $\Delta t$ and time constant $\tau$. Although the applications in Section 3.3 show that many practical examples do not suffer from stability issues, when

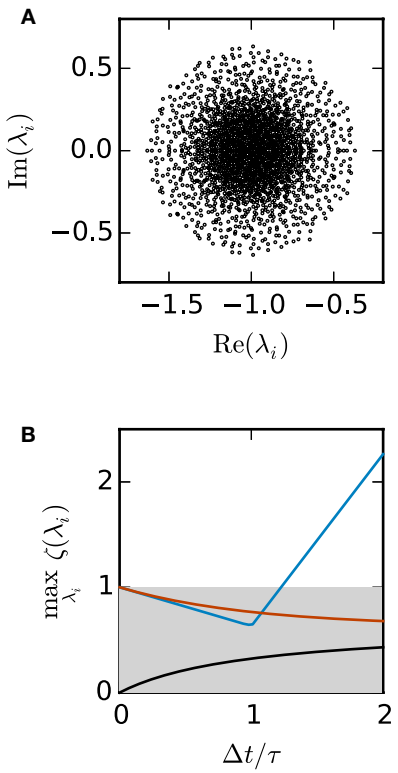


**FIGURE 3 | Stability analysis for the sparse balanced e/i test-case.** **(A)** Black circles show the eigenvalues $\lambda_i$ of the matrix $A$. Network parameters: $N = 2000$, $p = 0.2$. **(B)** The curves show the maximum of the stability function $\zeta(\lambda_i)$ over all eigenvalues $\lambda_i$ for the investigated methods ($\zeta_{\text{EM}}$: blue, $\zeta_{\text{IE}}$: black, $\zeta_{\text{EXP}}$: red) with respect to $\Delta t/\tau$. The gray area indicates the region where the stability criterion is met.

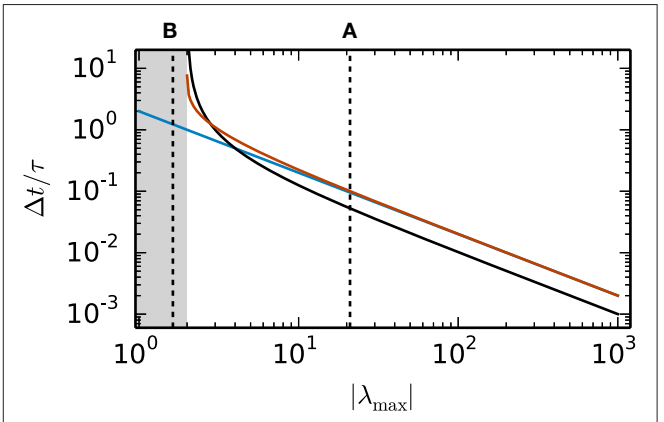


**FIGURE 4 | Step size restrictions of the numerical methods.** Largest ratio $\Delta t/\tau$ for which the different numerical methods (blue curve: Euler-Maruyama method, black curve: implicit Euler method solved with fixed-point iteration, red curve: scalar exponential Euler method) are stable shown against the absolute value of $\lambda_{\max} = \text{argmax}_{\lambda_i}\, \zeta(\lambda_i) \in \mathbb{R}$. The gray area indicates the region where the scalar exponential Euler method and the implicit Euler method are stable without any restrictions on $\Delta t/\tau$. The dotted vertical lines correspond to the examples presented in the panels of **Figure 2**.

a commonly used simulation step size is employed, the inevitable restrictions on the step size $\Delta t$ should be taken into account in simulations of rate-model networks. For simulations of linear rate models an appropriate step size can be determined by an analysis of the eigenvalues of $A$.

## 3.2. Performance of the Reference Implementation

This section investigates the performance of the rate model reference implementation. We are interested in i) the scalability of the rate model framework and ii) the comparison between the standard implementation with communication in every computation time step and the iterative approach using waveform relaxation (see Section 2.3.3 for details). We perform the simulations on the JUQUEEN BlueGene/Q supercomputer (Jülich Supercomputing Centre, 2015) at the Jülich Research Centre in Germany. It comprises 28,672 compute nodes, each with a 16-core IBM PowerPC A2 processor running at 1.6 GHz. For our benchmarks we use 8 OpenMP threads per JUQUEEN compute node and denote by VP $= 8 \cdot$ #nodes the total number of virtual processes employed.

As a test case we employ the sparse balanced e/i test case of linear rate units ($\phi(x) = \psi(x) = x$) introduced in Section 3.1, but with a fixed number of 2000 inputs independent of the number of units to allow for an unbiased weak scaling.

A weak scaling (**Figure 5A**) shows that the scalability of the standard implementation is impaired by the massive amount of communication. While for perfect scaling the simulation time should be constant over the number of virtual processes, the actual simulation time is increased by 15–25% when the number of virtual processes is doubled for VP $<$ 256 and even up to 83% from 8,192 to 16,384 virtual processes. For the iterative method, the scaling behavior is close to constant up to 1,024 virtual processes. When more processes are employed, the simulation time is increasing. However, the iterative method shows a better scaling behavior as the increase is weaker compared to the standard computation due to the lower total number of communication steps. Due to the higher computational load of the iterative method (see Section 2.3.3) the simulation time is larger compared to the straight forward approach for a small number of VP, where communication is not that crucial. For VP $\geq$ 1024, the iterative approach is superior with a speed up factor close to three for 16,384 virtual processes (1,209 s vs. 3,231 s).

The strong scaling scenario with a fixed total number of $N = 51,200$ units in **Figure 5B** constitutes a similar result. The iterative approach is beneficial for more than 1,024 virtual processes and the scaling behavior of the iterative method outperforms that of the standard computation. Starting at 4,096 virtual processes the savings in computation time decrease, which is explained by the very low workload of each single compute node. Again, for a smaller number of virtual processes the amount of additional computations is too high to outperform the standard implementation.

Despite the overall good scaling behavior, the performance in terms of absolute compute time is inferior to a simulator specifically designed for rate-based models alone (not shown). In the latter case it increases performance to collect the states of all units in one vector. If further the connectivity is available in form of a matrix and the delays are zero or homogeneous, the network can be efficiently updated with a single matrix-vector multiplication. Thus, the increased functionality and flexibility of having rate- and spiking models unified in one simulator comes for the price of a loss of performance for the rate-based models. However, as noted in the introduction, the number of units in rate-based network models is usually small and therefore performance is not as critical as for spiking network models.

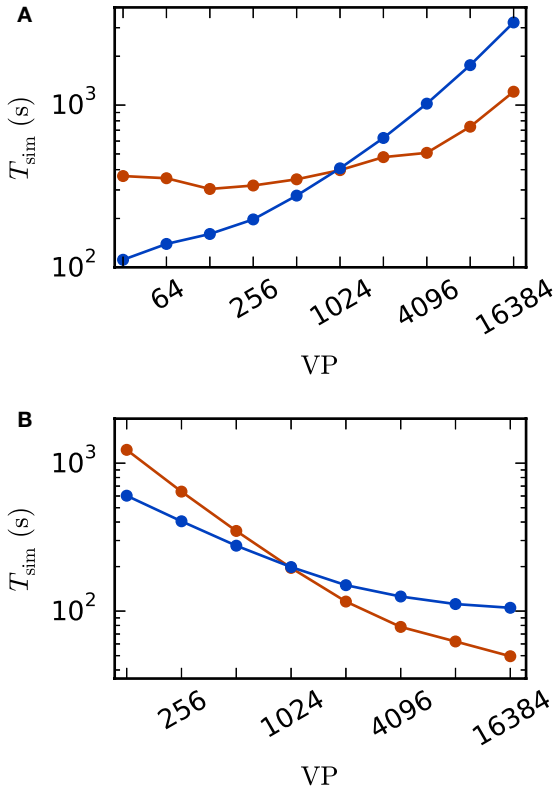


**FIGURE 5 | Scaling behavior of an excitatory-inhibitory network.** Simulation time with waveform relaxation (red curves, `wfr_comm_interval`: 1.0 ms, `wfr_tol`: $10^{-4}$) and without waveform relaxation (blue curves) as a function of the number of virtual processes in double logarithmic representation. The simulations span $T = 100$ ms of biological time at a computation step size of $h = 0.1$ ms. Sparse balanced e/i test case but with a fixed number of 2,000 inputs per unit. Other parameters: $\mu = 0$, $\sigma = 1$ and $\tau = 10$ ms **(A)** Weak scaling with 100 units per virtual process VP. **(B)** Strong scaling with a total number of $N = 51,200$ units.

## 3.3. Applications

First, we discuss a random inhibition-dominated network of linear rate units, then include nonlinear rate dynamics in a random network, and spatially structured connectivity in a functional neural-field model. In each case, simulation results are compared to analytical predictions. Furthermore, we simulate a mean-field model of a spiking model of a cortical microcircuit and discuss possible generalizations.

### 3.3.1. Linear Model

In the asynchronous irregular regime which resembles cortical activity, the dominant contribution to correlations in networks of nonlinear units is given by effective interactions between linear response modes (Pernice et al., 2011; Trousdale et al., 2012; Grytskyy et al., 2013; Dahmen et al., 2016). Networks of such noisy linear rate models have been investigated to explain features such as oscillations (Bos et al., 2016) or the smallness of average correlations (Tetzlaff et al., 2012; Helias et al., 2013). We here consider a prototypical network model of excitatory and inhibitory units following the linear dynamics given by Equation 9 with $\phi(x) = \psi(x) = x$, $\mu = 0$, and noise amplitude $\sigma$,

$$\tau dX^i(t) = \left( -X^i + \sum_{j=1}^{N} w^{ij} X^j(t) \right) dt + \sqrt{\tau}\sigma dW^i(t)\,. \qquad (20)$$

Due to the linearity of the model, the cross-covariance between units $i$ and $j$ can be calculated analytically and is given by Ginzburg and Sompolinsky (1994); Risken (1996); Gardiner (2004); Dahmen et al. (2016):

$$c(t) = \sum_{i,j} \frac{v^{i\mathrm{T}} \sigma^2 v^j}{\lambda_i + \lambda_j} u^i u^{j\mathrm{T}} \left( H(t) \frac{1}{\tau} e^{-\lambda_i \frac{t}{\tau}} + H(-t) \frac{1}{\tau} e^{\lambda_j \frac{t}{\tau}} \right), \qquad (21)$$

where $H$ denotes the Heaviside function. The $\lambda_i$ indicate the eigenvalues of the matrix $\mathbb{1} - W$ corresponding to the $i$-th left and right eigenvectors $v^i$ and $u^i$ respectively. Nonzero delays yield more complex analytical expressions for cross-correlations. In the population-averaged case, theoretical predictions are still analytically tractable (Equation 18 in Grytskyy et al., 2013). **Figure 6** shows the cross-covariance functions for pairs of instantaneously coupled units in a large network, as well as population-averaged covariance functions in a network of excitatory and inhibitory units with delayed interactions. In both cases, simulations are in good agreement with the theoretical predictions.

### 3.3.2. Non-linear Model

So far we considered a network with linear couplings between the units. Qualitatively new features appear in the presence of nonlinearities. One of the most prominent examples is the emergence of chaotic dynamics (Sompolinsky et al., 1988) in a network of nonlinearly coupled rate units. The original model is deterministic and has been recently extended to stochastic dynamics (Goedeke et al., 2016). The model definition follows from Equation (9) with $\mu = 0$, $\phi(x) = \mathrm{x}$, $\psi(x) = \tanh(x)$, i.e.

$$\tau dX^i(t) = \left( -X^i(t) + \sum_{j=1}^{N} w^{ij} \tanh\left(X^j(t)\right) \right) dt + \sqrt{\tau}\sigma\, dW^i(t)\,, \qquad (22)$$

where $w^{ij} \approx \mathcal{N}(0, g^2/N)$ are Gaussian random couplings. In the thermodynamic limit $N \to \infty$, the population averaged autocorrelation function $c(t)$ can be determined within dynamic mean-field theory (Sompolinsky et al., 1988; Goedeke et al., 2016; Schuecker et al., 2016). Comparing $c(t)$ obtained by simulation of a network (Equation 22) with the analytical result (Goedeke et al., 2016, their Equations 6 and 8) demonstrates excellent agreement (**Figure 7**). The simulated network is two orders of magnitude smaller than the cortical microcircuit, illustrating that in this context finite-size effects are already negligible at this scale.

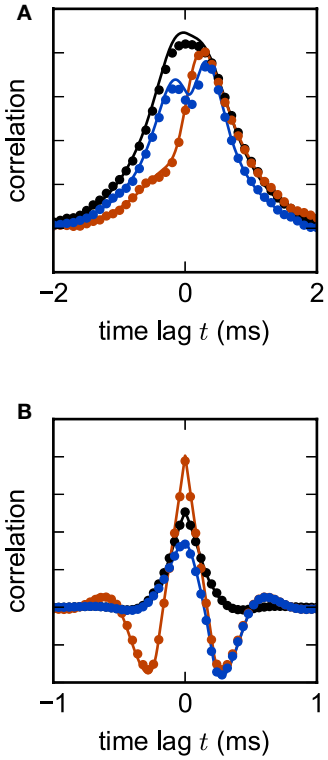


**FIGURE 6 | Linear rate model of a random excitatory-inhibitory network. (A)** Cross-correlation functions of two pairs of excitatory units (black, red) and an excitatory-inhibitory unit pair (blue) in a network without delay. The variability across correlation functions arises from heterogeneity in network connections (difference between black and red curves) and from different combinations of cell types (e.g., difference between black and blue curves). **(B)** Population-averaged autocorrelation function for excitatory (black) and inhibitory units (red), and cross-correlation function between excitatory and inhibitory units (blue) in a network with delay $d = 2$ ms. Symbols denote simulation results, curves show theoretical predictions. Parameters: $N_E = 80$ excitatory and $N_I = 20$ inhibitory units, random connections with fixed out-degree, connection probability $p = 0.1$, excitatory weight $w_E = 1/\sqrt{N_E + N_I}$, inhibitory weight $w_I = -6\,w_E$, $\tau = 1$ ms, $\mu = 0$, $\sigma = 1$. Step size $h = 0.1$ ms.

### 3.3.3. Functional Model

Complex dynamics not only arises from nonlinear single-unit dynamics, but also from structured network connectivity (Yger et al., 2011). One important nonrandom feature of brain connectivity is the spatial organization of connections (Malach et al., 1993; Voges et al., 2010). In spatially structured networks, delays play an essential role in shaping the collective dynamics (Roxin et al., 2005; Voges and Perrinet, 2012). Patterns of activity in such networks are routinely investigated using neural-field models. In contrast to the models discussed above,

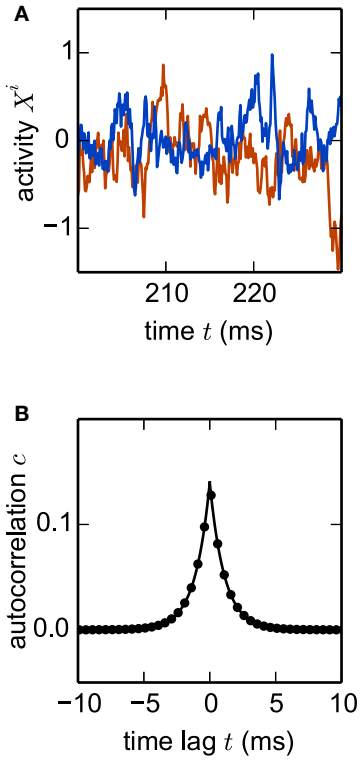


**FIGURE 7 | Nonlinear network model.** Simulation of the network specified by Equation (22) with $N = 1000$ units. **(A)** Noisy example trajectories of two units. **(B)** Autocorrelation function obtained by simulation averaged over all units (dots) and theory (solid curve). Other parameters: $\tau = 1$ ms, $\sigma = 0.5$, $g = 0.5$. Step size $h = 0.1$ ms.

field models require a discretization of space for numerical simulation. Such discretization can be done in the real space, leading effectively to a network of units at discrete positions in space, or alternatively, for particular symmetries in the couplings, in k-space (Roxin et al., 2006). Here, we follow the more general approach of discretization in real space.

A prototypical model of a spatial network is given by Roxin et al. (2005), where the authors consider the neural-field model

$$\tau dX(\varphi, t) = \left(-X(\varphi, t) + \phi\left[I_{\text{ext}} + \int_{-\pi}^{\pi} d\varphi'\, w(|\varphi - \varphi'|)X(\varphi', t - d)\right]\right) dt \tag{23}$$

with delayed (delay $d$) interactions, constant input $I_{\text{ext}}$, threshold-linear activation function $\phi = x \cdot H(x)$ and periodic Mexican-hat shaped connectivity

$$w(|\varphi - \varphi'|) = w_0 + w_1 \cos(\varphi - \varphi'). \tag{24}$$

The spatial variable $\varphi$ can also be interpreted as the preferred orientation of a set of units, thus rendering Equation (23) a model in feature space (Hansel and Sompolinsky, 1998). Discretizing space into $N$ segments yields the following set of coupled ODEs:

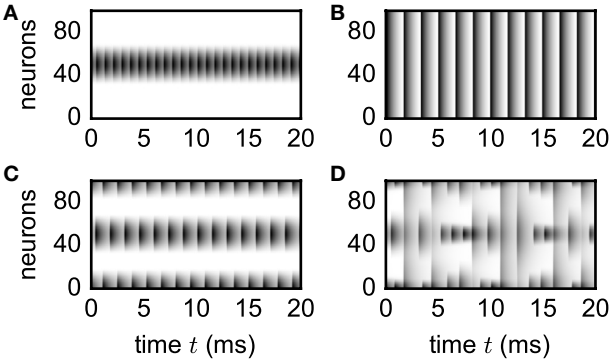


**FIGURE 8 | Spatial patterns in functional neural-field model.** Vertical axis shows unit indices organized according to ascending angle $\varphi \in [-\pi, \pi)$. Activity $X_i(t) = X(\varphi_i, t)$ encoded by gray scale with white denoting no activity. Initial transients not shown. Patterns reproduce the analytically derived phase diagram in the original study by Roxin et al. (2005). Parameters: $N = 100$, $d = 0.1$ ms, $\tau = 1$ ms, $I_{\text{ext}} = 1$, $w_0 = -80$, $w_1 = 15$ **(A)**, $w_1 = 5$ **(B)**, $w_1 = -46$ **(C)**, $w_1 = -86$ **(D)**. Initial condition: $X_i(0) = X(\varphi_i, 0) = \pi^2 - \varphi_i^2$. Step size $h = 0.01$ ms.

$$\tau dX^i = \left(-X^i + \phi\left[I_{\text{ext}} + \sum_{j=1}^{N} w^{ij} X^j(t - d)\right]\right) dt \tag{25}$$

with connectivity $w^{ij} = \frac{2\pi}{N} w(|\varphi^i - \varphi^j|)$, $\varphi_i = -\pi + \frac{2\pi}{N} \cdot i$ for $i \in [1, N]$ and discretization factor $\frac{2\pi}{N}$ that scales the space constants $w_0$ and $w_1$ with the neuron density. The spatial connectivity together with a delay in the interaction introduce various spatial activity patterns depending on the shape of the Mexican-hat connectivity.

To illustrate applicability of the simulation framework to neural-field models, we reproduce various patterns (**Figure 8**) observed by Roxin et al. (2005). Although the discrete and continuous networks strictly coincide only in the thermodynamic limit $N \to \infty$, numerically obtained patterns shown in **Figure 8** well agree with the analytically derived phase diagram of the continuous model (Roxin et al., 2005) already for network sizes of only $N = 100$ units.

### 3.3.4. Mean-Field Analysis of Complex Networks

A network of spiking neurons constitutes a high dimensional and complex system. To investigate its stationary state, one can describe the activity in terms of averages across neurons and time, leading to population averaged stationary firing rates (Brunel, 2000). Here, the spatial average collapses a large number of neurons into a single population, which is interpreted as a single rate unit. The ability to represent spiking as well as rate dynamics by the same simulation framework allows a straight-forward analysis of the spiking network by replacing the spiking neuron populations by single rate-based units.

In more formal terms, we now consider networks of neurons structured into $N$ interconnected populations. A neuron in population $\alpha$ receives $K_{\alpha\beta}$ incoming connections from neurons in population $\beta$, each with synaptic efficacy $w_{\alpha\beta}$. Additionally, each neuron in population $\alpha$ is driven by $K_{\alpha,\text{ext}}$ Poisson sources

with rate $X_{\text{ext}}$ and synaptic efficacy $w_{\text{ext}}$. We assume leaky integrate-and-fire model neurons with exponentially decaying post-synaptic currents. The dynamics of membrane potential $V$ and synaptic current $I_s$ is (Fourcaud and Brunel, 2002)

$$\begin{aligned}\tau_m \frac{dV^i}{dt} &= -V^i + I_s^i \\ \tau_s \frac{dI_s^i}{dt} &= -I_s^i + \tau_m \sum_{j=1}^{N} w^{ij} \sum_k \delta(t - t_k^j - d),\end{aligned} \tag{26}$$

where $t_k^j$ denotes the $k$-th spike-time of neuron $j$, and $\tau_m$ and $\tau_s$ are the time constants of membrane and synapse, respectively. The membrane resistance has been absorbed in the definition of the current. Whenever the membrane potential $V$ crosses the threshold $\theta$, the neuron emits a spike and $V$ is reset to the potential $V_r$, where it is clamped for a period of length $\tau_r$. Given that all neurons have identical parameters, a diffusion approximation, assuming asynchronous and Poissonian spiking statistics as well as small synaptic couplings, leads to the population-averaged firing rates $X_\alpha$ (Fourcaud and Brunel, 2002)

$$\begin{aligned}\frac{1}{X_\alpha} &= \tau_r + \tau_m \sqrt{\pi} \int_{(V_r-\mu_\alpha)/\sigma_\alpha + \gamma\sqrt{\tau_s/\tau_m}}^{(\theta-\mu_\alpha)/\sigma_\alpha + \gamma\sqrt{\tau_s/\tau_m}} e^{u^2} \left(1 + \text{erf}(u)\right) \mathrm{d}u \\ &=: 1/\Phi_\alpha(X)\end{aligned} \tag{27}$$

$$\mu_\alpha = \tau_m \sum_\beta K_{\alpha\beta} w_{\alpha\beta} X_\beta + \tau_m K_{\alpha,\text{ext}} w_{\text{ext}} X_{\text{ext}} \tag{28}$$

$$\sigma_\alpha^2 = \tau_m \sum_\beta K_{\alpha\beta} w_{\alpha\beta}^2 X_\beta + \tau_m K_{\alpha,\text{ext}} w_{\text{ext}}^2 X_{\text{ext}}. \tag{29}$$

Here, $\gamma = |\zeta(1/2)|/\sqrt{2}$, with $\zeta$ denoting the Riemann zeta function (Abramowitz and Stegun, 1974). We find the fixed points of Equation (27) by solving the first-order differential equation (Wong and Wang, 2006; Schuecker et al., 2017)

$$\tau \frac{\mathrm{d}X_\alpha}{dt} = -X_\alpha + \Phi_\alpha(X), \tag{30}$$

which constitutes a network of rate units with the dimension equal to the number of populations $N$.

Next we apply this framework to a cortical microcircuit model (Potjans and Diesmann, 2014) constituting roughly 80,000 spiking neurons structured into 8 populations across 4 layers [$L23, L4, L5, L6$], with one excitatory and one inhibitory cell type each (**Figure 9**). The model exhibits irregular and stationary spiking activity (**Figure 9C**). Replacing each population by a single rate unit (**Figure 9B**) results in an eight-dimensional rate network (Equation 30) which converges to a fixed point corresponding to the population-averaged firing rates obtained from direct simulation of the spiking model (**Figure 9D**).

The analysis only considers the stationary state of the microcircuit, which can as well be determined using the population-density approach (Cain et al., 2016). While the mean-field approach presented is strictly valid only in the thermodynamic limit, finite-size fluctuations around this state are accessible using the noisy linear-rate model (Section 3.3.1) as elaborated by Bos et al. (2016) or within the population-density approach (Schwalger et al., 2016).

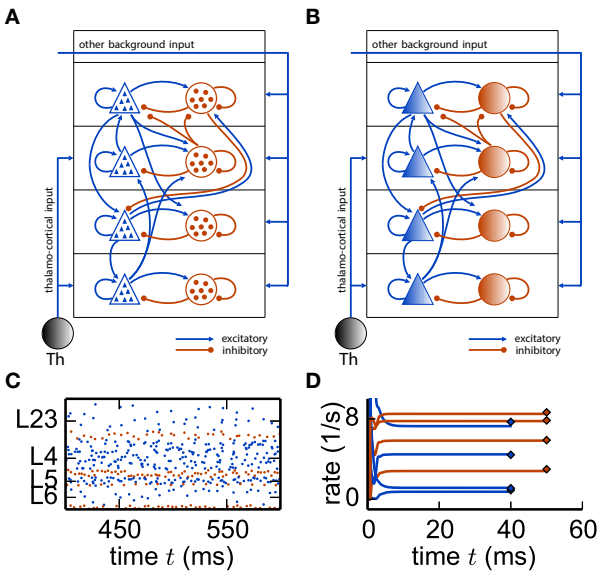


**FIGURE 9 | Reduction of spiking microcircuit model to rate dynamics.** **(A)** Sketch of a microcircuit model (Potjans and Diesmann, 2014) with excitatory (blue triangles) and inhibitory (red disks) neuron populations, each consisting of a large number of neurons indicated by the small triangles and disks respectively. Arrows between populations indicate the in-degree $K$. **(B)** Sketch of the corresponding reduced model where each population is replaced by a single rate unit. **(C)** Spiking activity in the different layers. **(D)** Dynamics of the eight units of the rate network (Equation 30) (curves) and comparison to population-averaged firing rates obtained from direct simulations of the spiking network (diamonds).

### 3.3.5. Further Rate Models

#### *3.3.5.1. Non-linear dynamics*

A characteristic feature of rate models considered so far is the leaky dynamics, i.e., the linear term $-X^i(t)$ in Equation (9). However, the presented framework can be extended to nonlinear dynamics as used for example by Stern et al. (2014). In a more general form Equation (9) reads

$$\begin{aligned}\tau dX^i(t) = &\left[ a(X^i(t)) + \phi\left( \sum_{j=1}^{N} w^{ij} \psi\left(X^j(t-d)\right) \right) \right] dt \\ &+ \sqrt{\tau}\sigma dW^i(t)\end{aligned} \tag{31}$$

where $a$ characterizes the intrinsic rate dynamics. If $a$ does not contain a linear part, the Euler-Maruyama scheme can be used for the update, i.e.,

$$\begin{aligned}X_{k+1}^i = X_k^i + &\left[ a(X_k^i) + \phi\left( \sum_{j=1}^{N} w^{ij} \psi\left(X^j_{k-\frac{d}{\Delta t}}\right) \right) \right] \frac{1}{\tau}\Delta t \\ &+ \frac{1}{\sqrt{\tau}}\sigma \Delta W_k^i.\end{aligned} \tag{32}$$

If $a$ also contains a linear part, so that $a(X^i) = -X^i + f(X^i)$, one can use an exponential Euler update approximating the nonlinear part as constant during the update. This leads to

$$X^i_{k+1} = e^{-\Delta t/\tau} X^i_k + \left(1 - e^{-\Delta t/\tau}\right) \left[ f(X^i_k) + \phi\left( \sum_{j=1}^{N} w^{ij} \psi\left( X^j_{k-\frac{d}{\Delta t}} \right) \right) \right] + \sqrt{\frac{1}{2}(1 - e^{-2\Delta t/\tau})}\, \sigma\, \eta^i_k, \quad (33)$$

with $\eta^i_k \sim \mathcal{N}(0,1)$.

#### 3.3.5.2. Multiplicative coupling

Another possible extension is a multiplicative coupling between units as for example employed in Gancarz and Grossberg (1998) or the original works of Wilson and Cowan (1972, 1973). In the most general form, this amounts to

$$\tau\, dX^i(t) = \left[ -X^i(t) + H(X^i) \cdot \phi\left( \sum_{j=1}^{N} w^{ij} \psi\left( X^j(t-d) \right) \right) \right] dt + \sqrt{\tau}\sigma\, dW^i(t)\,, \quad (34)$$

which, again assuming the coupling term to be constant during the update, can be solved using the exponential Euler update

$$X^i_{k+1} = e^{-\Delta t/\tau} X^i_k + \left(1 - e^{-\Delta t/\tau}\right) \left[ H(X^i_k) \cdot \phi\left( \sum_{j=1}^{N} w^{ij} \psi\left( X^j_{k-\frac{d}{\Delta t}} \right) \right) \right] + \sqrt{\frac{1}{2}(1 - e^{-2\Delta t/\tau})}\, \sigma\, \eta^i_k, \quad (35)$$

with $\eta^i_k \sim \mathcal{N}(0,1)$.

#### 3.3.5.3. Multiplicative noise

So far, we have considered rate models subject to additive noise corresponding to $b(t, x(t)) = b(t)$ in Equation (4). The linear rate model considered in Section 3.3.1 describes the dynamics around a stationary state and due to the stationary baseline, the noise amplitude is constant. However, one might relax the stationarity assumption which would render the noise amplitude proportional to the time dependent rate, i.e., a multiplicative noise amplitude. The presented framework covers the latter since the exponential Euler update is also valid for multiplicative noise (Equation 8).

#### 3.3.5.4. Output noise

Grytskyy et al. (2013) show that there is a mapping between a network of leaky integrate-and-fire models and a network of linear rate models with so-called output noise. Here the noise is added to the output rate of the afferent units

$$\tau \frac{dX^i(t)}{dt} = -X^i(t) + \mu + \phi\left( \sum_{j=1}^{N} w^{ij} \psi\left( X^j(t-d) + \sqrt{\tau}\sigma\, \xi^j(t) \right) \right) \quad i = 1, \ldots, N \quad (36)$$

and we cannot write the system as a SDE of type (2), as the nonlinearities $\phi(x)$ and $\psi(x)$ are also applied to the white noise $\xi^j$. In addition to the implementation `rate_neuron_ipn` for the rate-based models (Equation 9) discussed in the present work, our reference implementation also contains a base implementation `rate_neuron_opn` for models with output noise. For these models, the stochastic exponential Euler method can not be employed. Instead the solver assumes the noise $\xi^j$ to be constant over the update interval which leads to the update formula

$$X^i_{k+1} = e^{-\Delta t/\tau} X^i_k + \left(1 - e^{-\Delta t/\tau}\right) \left[ \mu + \phi\left( \sum_{j=1}^{N} w^{ij} \psi\left( X^j_k + \sqrt{\frac{\tau}{\Delta t}}\sigma\, \eta^j_k \right) \right) \right]. \quad (37)$$

The term $X^j_k + \sqrt{\frac{\tau}{\Delta t}}\sigma\, \eta^j_k$ with $\eta^j_k \sim \mathcal{N}(0,1)$ is calculated beforehand in the sending unit $j$, which results in the same amount of communicated data as in the case of models with input noise.

## 4. DISCUSSION

This work presents an efficient way to integrate rate-based models in a neuronal network simulator that is originally designed for models with delayed spike-based interactions. The advantage of the latter is a decoupling of neuron dynamics between spike events. This is used by current parallel simulators for large-scale networks of spiking neurons to reduce communication between simulation processes and significantly increase performance and scaling capabilities up to supercomputers (Morrison et al., 2005). In contrast, rate-based models interact in a continuous way. For delayed interactions, rate dynamics are still decoupled for the minimal delay of the network such that information can be exchanged on a coarse time-grid. For instantaneous coupling, communication in every time step is required. This is feasible for small networks that can be simulated on small machines and thus require only a small amount of communication. For improved efficiency of simulations of large networks on supercomputers, we implement an iterative numerical solution scheme (Lelarasmee, 1982). Furthermore, we investigate several standard methods for the solution of rate model equations and demonstrate that the scalar exponential Euler method is the best choice in the context of a neuronal network simulator that is originally designed for models with delayed spike-based interactions. Afterwards, we show the applicability of the numerical implementation to a variety of well-known and widely-used rate models and illustrate possible generalizations to other categories of rate-based models.

The current reference implementation uses an exponential Euler scheme (Adamu, 2011; Komori and Burrage, 2014) with a diagonal matrix $A$ (scalar exponential Euler): The additive noise as well as the leaky dynamics of single neurons are exactly integrated while the network input to the rate units is approximated as piecewise constant. The analysis in Section 3.1 demonstrates that the scalar exponential Euler is the most

accurate, stable and efficient standard-method for SDEs that is applicable to a distributed spiking simulator. In particular for all-to-all connected networks of linear rate units the distributed design renders implicit methods less feasible, as the convergence of the involved fixed-point iteration requires small time-steps. For all methods the computation step size needs to be compared against the time constant $\tau$. Therefore, stable solutions for small values $\tau \ll 1$ may require to decrease the step size below a default value.

The reference implementation provides an optional iterative method, the waveform relaxation (Lelarasmee, 1982), for networks with instantaneous rate connections. This method obtains the same results as the standard approach, but improves scalability by reducing communication at the cost of additional computations. As a consequence, the optimal method (standard vs. iterative) depends on the numbers of compute nodes and virtual processes. In our test case the use of the waveform-relaxation technique is beneficial for 1024 or more virtual processes. It is therefore recommended to employ the iterative scheme for large-scale simulations on supercomputers, but to disable it for smaller rate-model simulations on local workstations or laptops. This can easily be achieved by the parameter `use_wfr` (see Section 2.3.3 for details) of the algorithm. In general, the scalability for simulations of rate models is worse than for spiking network simulations (Kunkel et al., 2014) and comparable to simulations with gap junctions (Hahne et al., 2015). This is expected since for rate connections as well as for gap junctions a large amount of data needs to be communicated compared to a spiking simulation. Future work should assess whether this bottleneck can be overcome by a further optimized communication scheme.

While our reference implementation uses the simulation software NEST as a platform, the employed algorithms can be ported to other parallel spiking network simulators. Furthermore, the implementation of the example rate models as templates allows customization to arbitrary gain functions. Researchers can create additional models, without in-depth knowledge of simulator specific data structures or numerical methods. In addition, the infrastructure is sufficiently general to allow for extensions to other categories of rate models as shown explicitly for nonlinear dynamics, multiplicative coupling, and other types of noise. This design enables the usage of the framework for a large body of rate-based network models. Furthermore, the generality of the model equations supports applications beyond neuronal networks, such as in computational gliascience (Amiri et al., 2012) or artificial intelligence (Haykin, 2009).

Some design decisions for the reference implementation come with an up- and a downside and may at the present state of knowledge and experience constitute judgment calls: The choice to determine the type of nonlinearity of the recurrent network input with a boolean parameter is based on the assumption that this implementation covers the majority of rate models used in neuroscience today. The solution has an advantage in maintainability as it results in half as many template instances for a given set of gain functions than the alternative solution discussed above. It also avoids the introduction of potentially confusing names of rate models encoding the nature of the nonlinearity. On the downside models that do actually employ both nonlinearities at once cannot be expressed. Furthermore, a decision that can already be made at the time when the model instance is created, is delayed to the simulation phase. The decision to create a separate connection type for mean-field models of the `siegert` type is led by the ambition to avoid memory overhead. This comes at the price that units of this type cannot be connected to instances of rate models using the generic rate connection. Adapter elements like the `parrot_neuron` (see Kunkel et al., 2011, for a recent application) are one way to overcome this problem. Only the experience of researchers with the present implementation will inform us on whether characteristics and user interface serve the purpose of the community or if particular decisions need revision.

Mean-field theory has built a bridge between networks of spiking neurons and rate-based units that either represent single neurons or populations (Buice and Chow, 2007; Buice et al., 2010; Ostojic and Brunel, 2011; Bressloff, 2012; Grytskyy et al., 2013). In the latter case, the rate-based approach comes along with a considerable reduction of dimensionality (Section 3.3.4). Due to a possibly large number of populations, the fixed-point solution of the stationary activity can generally not be determined analytically, but still be found by evolving a pseudo-time dynamics. Within the presented framework, this approach is much faster than the spiking counter-part and thus facilitates the calibration of large-scale spiking network models (Schuecker et al., 2017).

Our unifying framework allows researchers to easily switch between rate-based and spiking models in a particular network model requiring only minimal changes to the simulation script. This facilitates an evaluation of the different model types against each other and increases reproducibility in the validation of reductions of spiking networks to rate-based models. Furthermore, it is instructive to study whether and how the network dynamics changes with the neuron model (Brette, 2015). In particular, functional networks being able to perform a given task are typically designed with rate-based units. Their validity can now be evaluated by going from a more abstract rate-based model to a biologically more realistic spiking neuron model. The present reference implementation does not allow for interactions between spiking and rate-based units. While this is technically trivial to implement, the proper conversion from spikes to rates and vice versa is a conceptual issue that has to be explored further by theoretical neuroscience.

The presented joined platform for spike-based and rate-based models hopefully triggers new research questions by facilitating collaboration and translation of ideas between scientists working in the two fields. This work therefore contributes to the unification of both modeling routes in multi-scale approaches combining large-scale spiking networks with functionally inspired rate-based elements to decipher the dynamics of the brain.

## AUTHOR CONTRIBUTIONS

Under the supervision of MH and MD, the authors JH, DD, and JS jointly worked on all parts of the above publication. DD and JS thereby focused on the neuroscientific applications

and the implementation of neuron models. JH focused on the NEST infrastructure, numerical schemes and performance benchmarks. All authors contributed to the writing of the manuscript.

## ACKNOWLEDGMENTS

The authors gratefully acknowledge the computing time on the supercomputer JUQUEEN (Jülich Supercomputing Centre, 2015) at Forschungszentrum Jülich granted by firstly the JARA-HPC Vergabegremium (provided on the JARA-HPC partition, jinb33) and secondly by the Gauss Centre for Supercomputing (GCS) (provided by the John von Neumann Institute for Computing (NIC) on the GCS share, hwu12). This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 720270 (HBP SGA1). Partly supported by Helmholtz Portfolio Supercomputing and Modeling for the Human Brain (SMHB), the Initiative and Networking Fund of the Helmholtz Association, and the Helmholtz young investigator group VH-NG-1028. All network simulations carried out with NEST (http://www.nest-simulator.org).

## REFERENCES

Abbott, L., DePasquale, B., and Memmesheimer, R.-M. (2016). Building functional networks of spiking model neurons. *Nat. Neurosci.* 19, 350–355. doi: 10.1038/nn.4241

Abramowitz, M., and Stegun, I. A. (1974). *Handbook of Mathematical Functions: With Formulas, Graphs, and Mathematical Tables*. New York, NY: Dover Publications.

Adamu, I. A. (2011). *Numerical Approximation of SDEs and Stochastic Swift-Hohenberg Equation*. Ph.D. thesis, Heriot-Watt University.

Al-Mohy, A. H., and Higham, N. J. (2009). A new scaling and squaring algorithm for the matrix exponential. *SIAM J. Matrix Anal. Appl.* 31, 970–989. doi: 10.1137/09074721X

Amari, S.-I. (1977). Dynamics of pattern formation in lateral-inhibition type neural fields. *Biol. Cybern.* 27, 77–87. doi: 10.1007/BF00337259

Amiri, M., Bahrami, F., and Janahmadi, M. (2012). Functional contributions of astrocytes in synchronization of a neuronal network model. *J. Theor. Biol.* 292, 60–70. doi: 10.1016/j.jtbi.2011.09.013

Amit, D. J., and Brunel, N. (1997). Model of global spontaneous activity and local structured activity during delay periods in the cerebral cortex. *Cereb. Cortex* 7, 237–252. doi: 10.1093/cercor/7.3.237

Bednar, J. A. (2009). Topographica: Building and analyzing map-level simulations from python, c/c++, matlab, nest, or neuron components. *Front. Neuroinform.* 24:8. doi: 10.3389/neuro.11.008.2009

Bos, H., Diesmann, M., and Helias, M. (2016). Identifying anatomical origins of coexisting oscillations in the cortical microcircuit. *PLoS Comput. Biol.* 12:e1005132. doi: 10.1371/journal.pcbi.1005132

Bos, H., Morrison, A., Peyser, A., Hahne, J., Helias, M., Kunkel, S., et al. (2015). NEST 2.10.0. *Zenodo* doi: 10.5281/zenodo.44222

Bower, J. M., and Beeman, D. (2007). GENESIS (simulation environment). *Scholarpedia* 2:1383. doi: 10.4249/scholarpedia.1383

Bressloff, P. C. (2012). Spatiotemporal dynamics of continuum neural fields. *J. Phys. A Math. Theor.* 45:033001. doi: 10.1088/1751-8113/45/3/033001

Bressloff, P. C. (2015). Path-integral methods for analyzing the effects of fluctuations in stochastic hybrid neural networks. *J. Math. Neurosci.* 5:4. doi: 10.1186/s13408-014-0016-z

Brette, R. (2015). Philosophy of the spike: rate-based vs. spike-based theories of the brain. *Front. Syst. Neurosci.* 9:151. doi: 10.3389/fnsys.2015.00151

Brette, R., Rudolph, M., Carnevale, T., Hines, M., Beeman, D., Bower, J. M., et al. (2007). Simulation of networks of spiking neurons: a review of tools and strategies. *J. Comput. Neurosci.* 23, 349–398. doi: 10.1007/s10827-007-0038-6

Brunel, N. (2000). Dynamics of sparsely connected networks of excitatory and inhibitory spiking neurons. *J. Comput. Neurosci.* 83, 183–208. doi: 10.1023/A:1008925309027

Brunel, N., and Hakim, V. (1999). Fast global oscillations in networks of integrate-and-fire neurons with low firing rates. *Neural Comput.* 11, 1621–1671. doi: 10.1162/089976699300016179

Buice, M. A., and Chow, C. C. (2007). Correlations, fluctuations, and stability of a finite-size network of coupled oscillators. *Phys. Rev. E* 76:031118. doi: 10.1103/physreve.76.031118

Buice, M. A., and Chow, C. C. (2013). Dynamic finite size effects in spiking neural networks. *PLoS Comput. Biol.* 9:e1002872. doi: 10.1371/journal.pcbi.1002872

Buice, M. A., and Cowan, J. D. (2007). Field-theoretic approach to fluctuation effects in neural networks. *Phys. Rev. E* 75:051919. doi: 10.1103/physreve.75.051919

Buice, M. A., Cowan, J. D., and Chow, C. C. (2010). Systematic fluctuation expansion for neural network activity equations. *Neural Comput.* 22, 377–426. doi: 10.1162/neco.2009.02-09-960

Cain, N., Iyer, R., Koch, C., and Mihalas, S. (2016). The computational properties of a simplified cortical column model. *PLoS Comput. Biol.* 12:e1005045. doi: 10.1371/journal.pcbi.1005045

Carnevale, T., and Hines, M. (2006). *The NEURON Book*. Cambridge: Cambridge University Press.

Chaudhuri, R., Knoblauch, K., Gariel, M.-A., Kennedy, H., and Wang, X.-J. (2015). A large-scale circuit mechanism for hierarchical dynamical processing in the primate cortex. *Neuron* 88, 419–431. doi: 10.1016/j.neuron.2015.09.008

Cichocki, A., Zdunek, R., Phan, A. H., and Amari, S.-i. (2009). *Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-Way Ddata Analysis and Blind Source Separation*. Chichester: John Wiley and Sons.

Coombes, S. (2005). Waves, bumps, and patterns in neural field theories. *Biol. Cybern.* 93, 91–108. doi: 10.1007/s00422-005-0574-y

Dahmen, D., Bos, H., and Helias, M. (2016). Correlated fluctuations in strongly coupled binary networks beyond equilibrium. *Phys. Rev. X* 6:031024. doi: 10.1103/physrevx.6.031024

Davison, A., Brüderle, D., Eppler, J., Kremkow, J., Muller, E., Pecevski, D., et al. (2008). PyNN: a common interface for neuronal network simulators. *Front. Neuroinformatics* 2:11. doi: 10.3389/neuro.11.011.2008

de Kamps, M. (2013). A generic approach to solving jump diffusion equations with applications to neural populations. *ArXiv e-prints* 1309.1654v2 [q-bio.NC].

de Kamps, M., Baier, V., Drever, J., Dietz, M., Mösenlechner, L., and van der Felde, F. (2008). The state of miind. *Neural Netw.* 21, 1164–1181. doi: 10.1016/j.neunet.2008.07.006

Deadman, E., Higham, N. J., and Ralha, R. (2012). "Blocked schur algorithms for computing the matrix square root," in *PARA, Vol. 7782 of Lecture Notes in Computer Science*, eds P. Manninen and P. Öster (Heidelberg: Springer), 171–182.

Deco, G., and Jirsa, V. K. (2012). Ongoing cortical activity at rest: criticality, multistability, and ghost attractors. *J. Neurosci.* 2, 3366–3375. doi: 10.1523/JNEUROSCI.2523-11.2012

Deco, G., Jirsa, V. K., and McIntosh, A. R. (2011). Emerging concepts for the dynamical organization of resting-state activity in the brain. *Nat. Rev. Neurosci.* 12, 43–56. doi: 10.1038/nrn2961

Deco, G., Jirsa, V. K., Robinson, P. A., Breakspear, M., and Friston, K. (2008). The dynamic brain: from spiking neurons to neural masses and cortical fields. *PLoS Comput. Biol.* 4:e1000092. doi: 10.1371/journal.pcbi.1000092

Destexhe, A., and Paré, D. (1999). Impact of network activity on the integrative properties of neocortical pyramidal neurons *in vivo*. *J. Neurophysiol.* 81, 1531–1547.

Djurfeldt, M., Davison, A. P., and Eppler, J. M. (2014). Efficient generation of connectivity in neuronal networks from simulator-independent descriptions. *Front. Neuroinformatics* 8:43. doi: 10.3389/fninf.2014.00043

Djurfeldt, M., Hjorth, J., Eppler, J. M., Dudani, N., Helias, M., Potjans, T. C., et al. (2010). Run-time interoperability between neuronal network simulators based on the MUSIC framework. *Neuroinformatics* 8, 43–60. doi: 10.1007/s12021-010-9064-z

Djurfeldt, M., Lundqvist, M., Johansson, C., Rehn, M., Ekeberg, O., and Lansner, A. (2008). Brain-scale simulation of the neocortex on the IBM Blue Gene/L supercomputer. *IBM J. Res. Dev.* 52, 31–41. doi: 10.1147/rd.521.0031

Ecker, A. S., Berens, P., Keliris, G. A., Bethge, M., and Logothetis, N. K. (2010). Decorrelated neuronal firing in cortical microcircuits. *Science* 327, 584–587. doi: 10.1126/science.1179867

Eppler, J. M., Helias, M., Muller, E., Diesmann, M., and Gewaltig, M. (2009). PyNEST: a convenient interface to the NEST simulator. *Front. Neuroinformatics* 2:12. doi: 10.3389/neuro.11.012.2008

Fan, Z. (2013). Sor waveform relaxation methods for stochastic differential equations. *Appl. Math. Comput.* 219, 4992–5003. doi: 10.1016/j.amc.2012.11.055

Feldman, J. A., and Ballard, D. H. (1982). Connectionist models and their properties. *Cogn. Sci.* 6, 205–254. doi: 10.1207/s15516709cog0603_1

Fourcaud, N., and Brunel, N. (2002). Dynamics of the firing probability of noisy integrate-and-fire neurons. *Neural Comput.* 14, 2057–2110. doi: 10.1162/089976602320264015

Galassi, M., Davies, J., Theiler, J., Gough, B., Jungman, G., Booth, M., et al. (2006). *GNU Scientific Library Reference Manual, 2nd Edn.* Network Theory Limited.

Gancarz, G., and Grossberg, S. (1998). A neural model of the saccade generator in the reticular formation. *IEEE Trans. Neural Netw.* 11, 1159–1174. doi: 10.1016/S0893-6080(98)00096-3

Gardiner, C. W. (2004). *Handbook of Stochastic Methods for Physics, Chemistry and the Natural Sciences, 3rd Edn.* Springer Series in Synergetics. Berlin: Springer.

Gentet, L., Avermann, M., Matyas, F., Staiger, J. F., and Petersen, C. C. (2010). Membrane potential dynamics of GABAergic neurons in the barrel cortex of behaving mice. *Neuron* 65, 422–435. doi: 10.1016/j.neuron.2010.01.006

Gerstner, W. (1995). Time structure of the activity in neural network models. *Phys. Rev. E* 51, 738–758. doi: 10.1103/PhysRevE.51.738

Gerstner, W. (2000). Population dynamics of spiking neurons: fast transients, asynchronous states, and locking. *Neural Comput.* 12, 43–89. doi: 10.1162/089976600300015899

Gerstner, W., Kistler, W. M., Naud, R., and Paninski, L. (2014). *Neuronal Dynamics. From single Neurons to Networks and Models of Cognition*. Cambridge: Cambridge University Press.

Gewaltig, M.-O., and Diesmann, M. (2007). NEST (NEural Simulation Tool). *Scholarpedia* 2:1430. doi: 10.4249/scholarpedia.1430

Ginzburg, I., and Sompolinsky, H. (1994). Theory of correlations in stochastic neural networks. *Phys. Rev. E* 50, 3171–3191.

Goedeke, S., Schuecker, J., and Helias, M. (2016). Noise dynamically suppresses chaos in neural networks. *arXiv*. 1603.01880v1 [q-bio.NC].

Goodman, D., and Brette, R. (2013). Brian simulator. *Scholarpedia* 8:10883. doi: 10.4249/scholarpedia.10883

Griffith, J. S., and Horn, G. (1966). An analysis of spontaneous impulse activity of units in the striate cortex of unrestrained cats. *J. Physiol.* 186, 516–534. doi: 10.1113/jphysiol.1966.sp008053

Grossberg, S. (1973). Contour enhancement, short term memory, and constancies in reverberating neural networks. *Stud. Appl. Math.* 52, 213–257. doi: 10.1002/sapm1973523213

Grytskyy, D., Tetzlaff, T., Diesmann, M., and Helias, M. (2013). A unified view on weakly correlated recurrent networks. *Front. Comput. Neurosci.* 7:131. doi: 10.3389/fncom.2013.00131

Hahne, J., Helias, M., Kunkel, S., Igarashi, J., Bolten, M., Frommer, A., et al. (2015). A unified framework for spiking and gap-junction interactions in distributed neuronal network simulations. *Front. Neuroinformatics* 9:22. doi: 10.3389/fninf.2015.00022

Hairer, E., and Wanner, G. (1991). *Solving Ordinary Differential Equations II*. Berlin: Springer.

Hansel, D., and Sompolinsky, H. (1998). "Modeling feature selectivity in local cortical circuits," in *Methods in Neuronal Modeling, 2nd Edn.,* eds C. Koch and I. Segev (Cambridge, MA: MIT Press), 499–567.

Hanuschkin, A., Kunkel, S., Helias, M., Morrison, A., and Diesmann, M. (2010). A general and efficient method for incorporating precise spike times in globally time-driven simulations. *Front. Neuroinform.* 4:113. doi: 10.3389/fninf.2010.00113

Haykin, S. S. (2009). *Neural Networks and Learning Machines, 3rd Edn.* New York, NY: Prentice Hall.

Helias, M., Kunkel, S., Masumoto, G., Igarashi, J., Eppler, J. M., Ishii, S., et al. (2012). Supercomputers ready for use as discovery machines for neuroscience. *Front. Neuroinform.* 6:26. doi: 10.3389/fninf.2012.00026

Helias, M., Rotter, S., Gewaltig, M., and Diesmann, M. (2008). Structural plasticity controlled by calcium based correlation detection. *Front. Comput. Neurosci.* 2:7. doi: 10.3389/neuro.10.007.2008

Helias, M., Tetzlaff, T., and Diesmann, M. (2013). Echoes in correlated neural systems. *New J. Phys.* 15:023002. doi: 10.1088/1367-2630/15/2/023002

Helias, M., Tetzlaff, T., and Diesmann, M. (2014). The correlation structure of local cortical networks intrinsically results from recurrent dynamics. *PLoS Comput. Biol.* 10:e1003428. doi: 10.1371/journal.pcbi.1003428

Hertz, J., Krogh, A., and Palmer, R. G. (1991). *Introduction to the Theory of Neural Computation*. Westview Press.

Hines, M., Eichner, H., and Schürmann, F. (2008). Neuron splitting in compute-bound parallel network simulations enables runtime scaling with twice as many processors. *J. Comput. Neurosci.* 25, 203–210. doi: 10.1007/s10827-007-0073-3

Hines, M., Kumar, S., and Schürmann, F. (2011). Comparison of neuronal spike exchange methods on a Blue Gene/P supercomputer. *Front. Comput. Neurosci.* 5:49. doi: 10.3389/fncom.2011.00049

Jülich Supercomputing Centre (2015). JUQUEEN: IBM Blue Gene/Q® supercomputer system at the Jülich Supercomputing Centre. *J. Large-Scale Res. Facil.* 1:A1. doi: 10.17815/jlsrf-1-18

Kelley, C. T. (1995). Iterative methods for linear and nonlinear equations. *Front. Appl. Math*. 156. doi: 10.1137/1.9781611970944

Kilpatrick, Z. P. (2015). "Wilson-Cowan model," in *Encyclopedia of Computational Neuroscience*, eds D. Jaeger and R. Jung (New York, NY: Springer), 3159–3163.

Kloeden, P. E., and Platen, E. (1992). *Numerical Solution of Stochastic Differential Equations*. Berlin: Springer.

Knight, B. W. (1972). Dynamics of encoding in a population of neurons. *J. Gen. Physiol.* 59, 734–766. doi: 10.1085/jgp.59.6.734

Koch, K. W., and Fuster, J. M. (1989). Unit activity in monkey parietal cortex related to haptic perception and temporary memory. *Exp. Brain Res.* 76, 292–306. doi: 10.1007/BF00247889

Komori, Y., and Burrage, K. (2014). A stochastic exponential Euler scheme for simulation of stiff biochemical reaction systems. *BIT Numer. Math.* 54, 1067–1085. doi: 10.1007/s10543-014-0485-1

Kriener, B., Enger, H., Tetzlaff, T., Plesser, H. E., Gewaltig, M.-O., and Einevoll, G. T. (2014). Dynamics of self-sustained asynchronous-irregular activity in random networks of spiking neurons with strong synapses. *Front. Comput. Neurosci.* 8:136. doi: 10.3389/fncom.2014.00136

Kumar, S., Heidelberger, P., Chen, D., and Hines, M. (2010). Optimization of applications with non-blocking neighborhood collectives via multisends on the blue gene/p supercomputer. *IPDPD*. doi: 10.1109/ipdps.2010.5470407

Kunkel, S., Diesmann, M., and Morrison, A. (2011). Limits to the development of feed-forward structures in large recurrent neuronal networks. *Front. Comput. Neurosci.* 4:160. doi: 10.3389/fncom.2010.00160

Kunkel, S., Schmidt, M., Eppler, J. M., Masumoto, G., Igarashi, J., Ishii, S., et al. (2014). Spiking network simulation code for petascale computers. *Front. Neuroinformatics* 8:78. doi: 10.3389/fninf.2014.00078

Lelarasmee, E. (1982). The waveform relaxation method for time domain analysis of large scale integrated circuits: theory and applications. *Memorandum*, No. UCB/ERL M82/40. doi: 10.1109/tcad.1982.1270004

Lindner, B., Doiron, B., and Longtin, A. (2005). Theory of oscillatory firing induced by spatially correlated noise and delayed inhibitory feedback. *Phys. Rev. E* 72:061919. doi: 10.1103/physreve.72.061919

Malach, R., Amir, Y., Harel, M., and Grinvald, A. (1993). Relationship between intrinsic connections and functional architecture revealed by optical imaging and *in vivo* targeted biocytin injections in primate striate cortex. *Proc. Natl. Acad. Sci. U.S.A.* 90, 10469–10473. doi: 10.1073/pnas.90.22.10469

Markram, H., Muller, E., Ramaswamy, S., Reimann, M. W., Abdellah, M., Sanchez, C. A., et al. (2015). Reconstruction and simulation of neocortical microcircuitry. *Cell* 163, 456–492. doi: 10.1016/j.cell.2015.09.029

Mattia, M., and Del Guidice, P. (2002). Population dynamics of interacting spiking neurons. *Phys. Rev. E* 66:051917. doi: 10.1103/physreve.66.051917

Mattia, M., and Del Guidice, P. (2004). Finite-size dynamics of inhibitory and excitatory interacting spiking neurons. *Phys. Rev. E* 70:052903. doi: 10.1103/physreve.70.052903

McClelland, J. L., and Rumelhart, D. E. (1981). An interactive activation model of context effects in letter perception: I. An account of basic findings. *Psychol. Rev.* 88:375. doi: 10.1037/0033-295X.88.5.375

McClelland, J. L., and Rumelhart, D. E. (1989). *Explorations in Parallel Distributed Processing: A Handbook of Models, Programs, and Exercises*. Cambridge: MIT Press.

Meyer, C., and van Vreeswijk, C. (2002). Temporal correlations in stochastic networks of spiking neurons. *Neural Comput.* 14, 369–404. doi: 10.1162/08997660252741167

Miekkala, U., and Nevanlinna, O. (1987). Convergence of dynamic iteration methods for initial value problems. *SIAM J. Sci. and Stat. Comput.* 8, 459–482. doi: 10.1137/0908046

Miyazaki, H., Kusano, Y., Shinjou, N., Fumiyoshi, S., Yokokawa, M., and Watanabe, T. (2012). Overview of the K computer System. *Fujitsu Sci. Tech. J.* 48, 255–265.

Montbrió, E., Pazó, D., and Roxin, A. (2015). Macroscopic description for networks of spiking neurons. *Phys. Rev. X* 5:021028. doi: 10.1103/physrevx.5.021028

Morrison, A., and Diesmann, M. (2008). "Maintaining causality in discrete time neuronal network simulations," in *Lectures in Supercomputational Neuroscience: Dynamics in Complex Brain Networks*, Understanding Complex Systems, eds P. beim Graben, C. Zhou, M. Thiel, and J. Kurths (Berlin: Springer), 267–278.

Morrison, A., Mehring, C., Geisel, T., Aertsen, A., and Diesmann, M. (2005). Advancing the boundaries of high connectivity network simulation with distributed computing. *Neural Comput.* 17, 1776–1801. doi: 10.1162/0899766054026648

Nichols, E. J., and Hutt, A. (2015). Neural field simulator: two-dimensional spatio-temporal dynamics involving finite transmission speed. *Front. Neuroinform.* 9:25. doi: 10.3389/fninf.2015.00025

Ohbayashi, M., Ohki, K., and Miyashita, Y. (2003). Conversion of working memory to motor sequence in the monkey premotor cortex. *Science* 301, 233–236. doi: 10.1126/science.1084884

Okun, M., and Lampl, I. (2008). Instantaneous correlation of excitation and inhibition during ongoing and sensory-evoked activities. *Nat. Neurosci.* 11, 535–537. doi: 10.1038/nn.2105

O'Reilly, R. C. (2014). *Comparison of Neural Network Simulators*. Available online at: https://grey.colorado.edu/emergent/index.php/Comparison_of_Neural_Network_Simulators. [Accessed: 2016-10-14].

O'Reilly, R. C., Munakata, Y., Frank, M. J., Hazy, T. E., and Contributors (2012). *Computational Cognitive Neuroscience, 1st Edn*. Wiki Book. Available online at: http://ccnbook.colorado.edu

O'Reilly, R. C., Munakata, Y., and McClelland, J. L. (2000). *Computational Explorations in Cognitive Neuroscience: Understanding the Mind by Simulating the Brain, 1st Edn*. Cambridge: MIT Press.

Ostojic, S. (2014). Two types of asynchronous activity in networks of excitatory and inhibitory spiking neurons. *Nat. Neurosci.* 17, 594–600. doi: 10.1038/nn.3658

Ostojic, S., and Brunel, N. (2011). From spiking neuron models to linear-nonlinear models. *PLoS Comput. Biol.* 7:e1001056. doi: 10.1371/journal.pcbi.1001056

Pernice, V., Staude, B., Cardanobile, S., and Rotter, S. (2011). How structure determines correlations in neuronal networks. *PLoS Comput. Biol.* 7:e1002059. doi: 10.1371/journal.pcbi.1002059

Plesser, H., Diesmann, M., Gewaltig, M.-O., and Morrison, A. (2015). "Nest: the neural simulation tool," in *Encyclopedia of Computational Neuroscience*, eds D. Jaeger and R. Jung (New York, NY: Springer), 1849–1852.

Plotnikov, D., Blundell, I., Ippen, T., Eppler, J. M., Morrison, A., and Rumpe, B. (2016). "NESTML: a modeling language for spiking neurons," in *Modellierung 2016 Conference*, *Vol. 254 LNI* (Bonn: Bonner Köllen Verlag), 93–108.

Potjans, T. C., and Diesmann, M. (2014). The cell-type specific cortical microcircuit: relating structure and activity in a full-scale spiking network model. *Cereb. Cortex* 24, 785–806. doi: 10.1093/cercor/bhs358

Rajan, K., and Abbott, L. F. (2006). Eigenvalue spectra of random matrices for neural networks. *Phys. Rev. Lett.* 97:188104. doi: 10.1103/PhysRevLett.97.188104

Renart, A., De La Rocha, J., Bartho, P., Hollender, L., Parga, N., Reyes, A., et al. (2010). The asynchronous state in cortical circuits. *Science* 327, 587–590. doi: 10.1126/science.1179850

Risken, H. (1996). *The Fokker-Planck Equation*. Berlin; Heidelberg: Springer Verlag.

Rougier, N. P., and Fix, J. (2012). Dana: distributed numerical and adaptive modelling framework. *Netw. Comput. Neural Syst.* 23, 237–253. doi: 10.3109/0954898X.2012.721573

Roxin, A., Brunel, N., and Hansel, D. (2005). The role of delays in shaping spatio-temporal dynamics of neuronal activity in large networks. *Phys. Rev. Lett.* 94:238103. doi: 10.1103/PhysRevLett.94.238103

Roxin, A., Brunel, N., and Hansel, D. (2006). Rate models with delays and the dynamics of large networks of spiking neurons. *Prog. Theor. Phys. Suppl.* 161, 68–85. doi: 10.1143/PTPS.161.68

Roxin, A., Brunel, N., Hansel, D., Mongillo, G., and van Vreeswijk, C. (2011). On the distribution of firing rates in networks of cortical neurons. *J. Neurosci.* 31, 16217–16226. doi: 10.1523/JNEUROSCI.1677-11.2011

Rumelhart, D. E., McClelland, J. L., and the PDP Research Group (1986). *Parallel Distributed Processing, Explorations in the Microstructure of Cognition: Foundations*, Vol. 1. Cambridge, MA: MIT Press.

Sadeh, S., and Rotter, S. (2015). Orientation selectivity in inhibition-dominated networks of spiking neurons: effect of single neuron properties and network dynamics. *PLoS Comput. Biol.* 11:e1004045. doi: 10.1371/journal.pcbi.1004045

Sanz Leon, P., Knock, S., Woodman, M., Domide, L., Mersmann, J., McIntosh, A., et al. (2013). The virtual brain: a simulator of primate brain network dynamics. *Front. Neuroinform.* 7:10. doi: 10.3389/fninf.2013.00010

Schmidt, M., Bakker, R., Diesmann, M., and van Albada, S. J. (2016). Full-density multi-scale account of structure and dynamics of macaque visual cortex. *arXiv preprint* arXiv:1511.09364v3.

Schöner, G., Spencer, J., and Group, D. (2015). *Dynamic Thinking: A Primer on Dynamic Field Theory*. Oxford Series in Developmental Cognitive Neuroscience. Oxford: Oxford University Press.

Schuecker, J., Diesmann, M., and Helias, M. (2015). Modulated escape from a metastable state driven by colored noise. *Phys. Rev. E* 92:052119. doi: 10.1103/physreve.92.052119

Schuecker, J., Goedeke, S., Dahmen, D., and Helias, M. (2016). Functional methods for disordered neural networks. *arXiv*. 1605.06758 [cond-mat.dis-nn].

Schuecker, J., Schmidt, M., van Albada, S. J., Diesmann, M., and Helias, M. (2017). Fundamental activity constraints lead to specific interpretations of the connectome. *PLoS Comput. Biol.* 13:e1005179. doi: 10.1371/journal.pcbi.1005179

Schurz, H., and Schneider, K. R. (2005). Waveform relaxation methods for stochastic differential equations. *Int. J. Numer. Anal. Model.* 3, 232–254.

Schwalger, T., Deger, M., and Gerstner, W. (2016). Towards a theory of cortical columns: from spiking neurons to interacting neural populations of finite size. *arXiv* 1611.00294.

Shadlen, M. N., and Newsome, W. T. (1998). The variable discharge of cortical neurons: implications for connectivity, computation, and information coding. *J. Neurosci.* 18, 3870–3896.

Shoji, I. (2011). A note on convergence rate of a linearization method for the discretization of stochastic differential equations. *Commun. Nonlinear Sci. Numer. Simul.* 16, 2667–2671. doi: 10.1016/j.cnsns.2010.09.008

Siegert, A. J. (1951). On the first passage time probability problem. *Phys. Rev.* 81, 617–623. doi: 10.1103/PhysRev.81.617

Softky, W. R., and Koch, C. (1993). The highly irregular firing of cortical cells is inconsistent with temporal integration of random EPSPs. *J. Neurosci.* 13, 334–350.

Sompolinsky, H., Crisanti, A., and Sommers, H. J. (1988). Chaos in random neural networks. *Phys. Rev. Lett.* 61, 259–262. doi: 10.1103/PhysRevLett.61.259

Stern, M., Sompolinsky, H., and Abbott, L. F. (2014). Dynamics of random neural networks with bistable units. *Phys. Rev. E* 90:062710. doi: 10.1103/physreve.90.062710

Tetzlaff, T., Helias, M., Einevoll, G., and Diesmann, M. (2012). Decorrelation of neural-network activity by inhibitory feedback. *PLoS Comput. Biol.* 8:e1002596. doi: 10.1371/journal.pcbi.1002596

Trousdale, J., Hu, Y., Shea-Brown, E., and Josic, K. (2012). Impact of network structure and cellular response on spike time correlations. *PLoS Comput. Biol.* 8:e1002408. doi: 10.1371/journal.pcbi.1002408

van Albada, S. J., Helias, M., and Diesmann, M. (2015). Scalability of asynchronous networks is limited by one-to-one mapping between

effective connectivity and correlations. *PLoS Comput. Biol.* 11:e1004490. doi: 10.1371/journal.pcbi.1004490

van Vreeswijk, C., and Sompolinsky, H. (1996). Chaos in neuronal networks with balanced excitatory and inhibitory activity. *Science* 274, 1724–1726. doi: 10.1126/science.274.5293.1724

Voges, N., and Perrinet, L. U. (2012). Complex dynamics in recurrent cortical networks based on spatially realistic connectivities. *Front. Comput. Neurosci.* 6:41. doi: 10.3389/fncom.2012.00041

Voges, N., Schüz, A., Aertsen, A., and Rotter, S. (2010). A modeler's view on the spatial structure of intrinsic horizontal connectivity in the neocortex. *Prog. Neurobiol.* 92, 277–292. doi: 10.1016/j.pneurobio.2010.05.001

Weitzenfeld, A., Arbib, M. A., and Alexander, A. (2002). *The Neural Simulation Language: A System for Brain Modeling*. Cambridge: MIT Press.

Wilson, H. R., and Cowan, J. D. (1972). Excitatory and inhibitory interactions in localized populations of model neurons. *Biophys. J.* 12, 1–24. doi: 10.1016/S0006-3495(72)86068-5

Wilson, H. R., and Cowan, J. D. (1973). A mathematical theory of the functional dynamics of cortical and thalamic nervous tissue. *Kybernetik* 13, 55–80. doi: 10.1007/BF00288786

Wong, K.-F., and Wang, X.-J. (2006). A recurrent network mechanism of time integration in perceptual decisions. *J. Neurosci.* 26, 1314–1328. doi: 10.1523/JNEUROSCI.3733-05.2006

Yger, P., El Boustani, S., Destexhe, A., and Frégnac, Y. (2011). Topologically invariant macroscopic statistics in balanced networks of conductance-based integrate-and-fire neurons. *J. Comput. Neurosci.* 31, 229–245. doi: 10.1007/s10827-010-0310-z

**Conflict of Interest Statement:** The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

# A. APPENDIX

## A.1. Numerical Evaluation of the Siegert Formula

We here describe how to numerically calculate Equation (27), frequently called Siegert formula in the literature (for a recent textbook see Gerstner et al., 2014), which is not straight forward due to numerical instabilities in the integral. First we introduce the abbreviations $y_\theta = (\theta - \mu)/\sigma + \gamma\sqrt{\tau_s/\tau_m}$ and $y_r = (V_r - \mu)/\sigma + \gamma\sqrt{\tau_s/\tau_m}$ and rewrite the integral as

$$\begin{aligned}
&\sqrt{\pi}\int_{y_r}^{y_\theta} e^{u^2}(1+\mathrm{erf}(u))du \\
=&2\int_{y_r}^{y_\theta} e^{u^2}\int_{-\infty}^{u} e^{-v^2}\,dv\,du \\
=&2\int_{y_r}^{y_\theta}\int_{-\infty}^{u} e^{(u+v)(u-v)}\,dv\,du\,.
\end{aligned}$$

Here, the numerical difficulty arises due to a multiplication of a divergent ($e^{u^2}$) and a convergent term ($1 + \mathrm{erf}(u)$) in the integrand. We therefore use the variable transform $w = v - u$ and obtain

$$\begin{aligned}
&= 2\int_{y_r}^{y_\theta}\int_{-\infty}^{u} e^{(u+v)(u-v)}\,dv\,du \\
=&2\int_{y_r}^{y_\theta}\int_{-\infty}^{0} e^{(2u+w)(-w)}\,dw\,du \\
=&\int_0^\infty e^{-w^2}\frac{e^{2y_\theta w} - e^{2y_r w}}{w}\,dw\,,
\end{aligned}$$

where we performed the integral over $u$ in the last line.

For $y_r, y_\theta < 0$ the integrand can be integrated straightforwardly as

$$\int_0^\infty \frac{e^{2y_\theta w - w^2} - e^{2y_r w - w^2}}{w}\,dw\,, \tag{A1}$$

where the two terms in the integrand converge separately and where, in approximation, the upper integration bound is chosen, such that the integrand has dropped to a sufficiently small value (here chosen to be $10^{-12}$). Here, for $w = 0$, the integrand has to be replaced by $\lim_{w\to 0}\frac{e^{2y_\theta w} - e^{2y_r w}}{w} = 2\left(y_\theta - y_r\right)$.

For $y_\theta > 0$ and $y_r < 0$ only the combination of the two terms in Equation (A1) converges. So we rewrite

$$\begin{aligned}
&\int_0^\infty e^{-w^2}\frac{e^{2y_\theta w} - e^{2y_r w}}{u}\,dw \\
=&\int_0^\infty e^{2y_\theta w - w^2}\frac{1 - e^{2(y_r - y_\theta)w}}{w}\,dw \\
=&\,e^{y_\theta^2}\int_0^\infty e^{-(w-y_\theta)^2}\frac{1 - e^{2(y_r - y_\theta)w}}{w}\,dw\,.
\end{aligned} \tag{A2}$$

The integrand has a peak near $y_\theta$. Therefore, in approximation, the lower and the upper boundary can be chosen to be left and right of $y_\theta$, respectively, such that the integrand has fallen to a sufficiently low value (here chosen to be $10^{-12}$). For $w = 0$ we replace the integrand by its limit, which is $\lim_{w\to 0} e^{-(w-y_\theta)^2}\frac{1-e^{2(y_r-y_\theta)w}}{w} = e^{-y_\theta^2}2\left(y_\theta - y_r\right)$.

We actually switch from Equations (A1) to (A2) when $y_\theta > 0.05|\tilde{V}_\theta|/\sigma_\alpha$ with $\tilde{V}_\theta = V_\theta + \gamma\sqrt{\tau_s/\tau_m}$. This provides a numerically stable solution in terms of a continuous transition between the two expressions. Our reference implementation numerically evaluates the integrals using the adaptive GSL implementation `gsl_integration_qags` (Galassi et al., 2006) of the Gauss-Kronrod 21-point integration rule.

## A.2. Usage of the NEST Reference Implementation

We here give a brief description of how to use our reference implementation of the continuous-time dynamics in the simulation code NEST with the syntax of the PyNEST interface (Eppler et al., 2009). Complete simulation scripts will be made available with one of the next major releases of NEST. The description focuses on rate-model specific aspects. A general introduction to PyNEST can be found on the simulator website (http://www.nest-simulator.org).

**Script 1** shows the creation of an excitatory-inhibitory network of linear rate units as used in our example in Section 3.3.1. Researchers already familiar with the PyNEST interface notice that there is no fundamental difference to scripts for the simulation of spiking neural networks. Line 4 illustrates how to disable the usage of the waveform-relaxation method. This is advisable for simulations on local workstations or clusters, where the waveform-relaxation method typically does not improve performance (see Section 3.2). The iterative method is enabled by default, as it is also employed for simulations with gap junctions, where it improves performance and even accuracy of the simulation, regardless of network size and parallelization (Hahne et al., 2015). Instances of the linear rate-based model are created by calling `nest.Create` in the usual way with model type `lin_rate_ipn`. The parameter `linear_summation` characterizes the type of nonlinearity ($\phi$ or $\psi$, see Section 2.3.2) of the rate model. In this particular example the explicit specification of the parameter is added for illustrative purposes only as i) the employed model is a linear rate model, where regardless of the choice of the parameter $\phi(x) = \psi(x) = x$ holds and ii) the default value is `True` anyway. Lines 13-14 and 31 demonstrate how to record the rate activity with the `multimeter`. The `record_from` parameter needs to be set to `rate` to pick up the corresponding state variable. As this particular network model includes delayed rate connections the synapse model `delay_rate_connection` is chosen (lines 17–20). In order to create instantaneous rate connections instead one changes the synapse model to `rate_connection` and removes the parameter `delay` from the synapse dictionary. For the simultaneous use of delayed and instantaneous connections one duplicates lines 17–28 and adapts the synapse and connection dictionaries of the copy according to the needs of the additional instantaneous connections.

**Script 1 | Simulation of an excitatory-inhibitory network of linear rate units.**

```
1 import nest
2
3 # Disable usage of waveform-relaxation method
4 nest.SetKernelStatus({'resolution': h, 'use_wfr': False})
5
6 # Create rate units and recording device
7 n_e = nest.Create('lin_rate_ipn', NE,
8                   params = {'linear_summation': True,
9                   'mean': mu, 'std': sigma, 'tau': tau})
10 n_i = nest.Create('lin_rate_ipn', NI,
11                   params = {'linear_summation': True,
12                   'mean': mu, 'std': sigma, 'tau': tau})
13 mm = nest.Create('multimeter', params = {'record_from': ['rate'],
14                  'interval': h, 'start': T_start})
15
16 # Specify synapse and connection dictionaries
17 syn_e = {'weight': w, 'delay': d,
18          'model': 'delay_rate_connection'}
19 syn_i = {'weight': -g * w, 'delay': d,
20          'model': 'delay_rate_connection'}
21 conn_e = {'rule': 'fixed_outdegree', 'outdegree': KE}
22 conn_i = {'rule': 'fixed_outdegree', 'outdegree': KI}
23
24 # Connect rate units
25 nest.Connect(n_e, n_e, conn_e, syn_e)
26 nest.Connect(n_i, n_i, conn_i, syn_i)
27 nest.Connect(n_e, n_i, conn_i, syn_e)
28 nest.Connect(n_i, n_e, conn_e, syn_i)
29
30 # Connect recording device to rate units
31 nest.Connect(mm, n_e + n_i)
32
33 # Start simulation
34 nest.Simulate(T)
```

*Here and in the following scripts we use the syntax of the PyNEST interface (Eppler et al., 2009) of the NEST simulation software as of version 2.10.0 (Bos et al., 2015). The script excludes the definitions of the parameters (*`h,NE,NI,mu,sigma,tau,T_start,w,d,g,KE,KI,T`*).*

**Script 2** shows a code snippet from a simulation script employing model (30) used for mean-field analysis of complex networks in Section 3.3.4. Here single rate units of type `siegert_neuron` represent an entire population of spiking neurons (lines 3-7). The units are coupled by connections of type `diffusion_connection`. This connection type is identical to type `rate_connection` for instantaneous rate connections except for the two parameters `drift_factor` and `diffusion_factor` substituting the parameter `weight` (lines 11–13). These two parameters reflect the prefactors in front of the rate variable in Equations (28) and (29). In general the prefactors differ from these well known forms; for example in case of distributed connection weights (see Helias et al., 2008, their Equation 33). Therefore, we prefer a generic parameterization over more specific alternatives like the pair `weight` and `convergence`.

**Script 2 | Connecting units of type `siegert_neuron`.**

```
import nest

# Create one siegert_neuron for the excitatory population
s_ex = nest.Create('siegert_neuron', 1)
# Create one siegert_neuron for the inhibitory population
s_in = nest.Create('siegert_neuron', 1)

[...]

# Create connections originating from the excitatory unit
syn_e = {'drift_factor': tau_m * K * w,
         'diffusion_factor': tau_m * K * w * w,
         'model': 'diffusion_connection'}
nest.Connect(s_ex, s_ex + s_in, 'all_to_all', syn_e)

[...]
```

*The script shows how connections between units of type (30) are created in PyNEST. Again the code snippet does not contain the definitions of the parameters (`tau_m`, `K`, `w`) for brevity.*