J Comput Neurosci (2007) 22:81–100
DOI 10.1007/s10827-006-8526-7

Fast numerical methods for simulating large-scale integrate-and-fire neuronal networks

Aaditya V. Rangan · David Cai

Received: 29 October 2005 / Revised: 25 March 2006 / Accepted: 28 March 2006 / Published online: 28 July 2006

Abstract We discuss numerical methods for simulating large-scale, integrate-and-fire (I&F) neuronal networks. Important elements in our numerical methods are (i) a neurophysiologically inspired integrating factor which casts the solution as a numerically tractable integral equation, and allows us to obtain stable and accurate individual neuronal trajectories (i.e., voltage and conductance time-courses) even when the I&F neuronal equations are stiff, such as in strongly fluctuating, high-conductance states; (ii) an iterated process of spike-spike corrections within groups of strongly coupled neurons to account for spike-spike interactions within a single large numerical time-step; and (iii) a clustering procedure of firing events in the network to take advantage of localized architectures, such as spatial scales of strong local interactions, which are often present in large-scale computational models—for example, those of the primary visual cortex. (We note that the spike-spike corrections in our methods are more involved than the correction of single neuron spike-time via a polynomial interpolation as in the modified Runge-Kutta methods commonly used in simulations of I&F neuronal networks.) Our methods can evolve networks with relatively strong local interactions in an asymptotically optimal way such that each neuron fires approximately once in $\mathcal{O}(N)$ operations, where N is the number of neurons in the system. We note that quantifications used in computational modeling are often statistical, since measurements in a real experiment to characterize physiological systems are typically statistical, such as firing rate, interspike interval distributions, and spike-triggered voltage distributions. We emphasize that it takes much less computational effort to resolve *statistical* properties of certain I&F neuronal networks than to fully resolve trajectories of each and every neuron within the system. For networks operating in realistic dynamical regimes, such as strongly fluctuating, high-conductance states, our methods are designed to achieve *statistical accuracy* when very large time-steps are used. Moreover, our methods can also achieve *trajectory-wise accuracy* when small time-steps are used.

Keywords Numerical algorithm · Network architecture

Action Editor: Nicolas Brunel

A. V. Rangan (✉) · D. Cai
Courant Institute of Mathematical Sciences, New York University, New York, NY 10012, USA
e-mail: rangan@cims.nyu.edu

1. Introduction

Systems of conductance-based integrate-and-fire (I&F) neurons are often used to study the behavior of large neuronal networks (Somers et al., 1995; Troyer et al., 1998; McLaughlin et al., 2000), since they are far simpler to evolve computationally than networks of other more complicated point neuron models (such as Hodgkin-Huxley type (Koch, 1999)). However, in numerical simulations of I&F neuronal networks, one theoretical point is often not sufficiently emphasized, namely, the architecture and dynamic regimes of the simulated network, as well as the modelling goals, often determine which numerical methods can be used effectively. For example, if we wish to measure the mean steady state firing rate of an all-to-all weakly coupled excitatory network, driven by high rate excitatory Poisson input with weak input synaptic strength (i.e., a mean-driven regime (Cai et al., 2004), we can use most standard numerical schemes (e.g., explicit Euler method). This is because the equations of the I&F neuronal network are not stiff in this dynamic regime, and the cortico-cortical interactions only weakly perturb the dynamics.[1] Any errors made in computing the spike-time or trajectory (*i.e., voltage and conductance time course*) of an individual neuron will barely affect the trajectories of the other neurons in the network. On the other hand, for a neuronal circuit with very specific wiring and strong connection strengths, it may happen that subcomponents of this network may have high conductances (i.e., the equations of the I&F neuronal network may be stiff). And it is also possible that slight changes in the initial conditions or tiny errors in the spike-times may give rise to drastically different network responses. If we want to numerically study the input-output properties of these networks, care must be taken because standard numerical methods may have difficulties in resolving the neuronal trajectories accurately in these cases.

In this paper, we focus on numerical methods for simulating physiologically plausible I&F neuronal networks arising from modeling the primary visual cortex (V1) (McLaughlin et al., 2000; Tao et al., 2003). These networks are somewhere in between the two extremes mentioned above. The choice of I&F neurons (instead of more detailed neuronal models) is dictated by the computational limit imposed by the spatiotemporal scales of the dynamics associated with physiological phenomena we are interested in, which often span $\sim 10\,\text{mm}^2 - 100\,\text{mm}^2$ with $\sim 10^5 - 10^6$ neurons. The I&F model point neuron is very idealized, therefore, a simulation of I&F neuronal networks can (usually) only offer a statistical understanding of a real physiological network, rather than a detailed neuron-by-neuron account of that network (Rauch et al., 2003; Fourcaud-Trocme et al., 2003; Geisler et al., 2005). Hence, our goal is to obtain accurate statistical properties of the simulated network—such as voltage and spike-time distributions, rather than accurate individual intracellular membrane potential time courses for each and every neuron in the system over a long time.

One way to accurately simulate an I&F neuronal network is to numerically evolve each individual point I&F neuron, and obtain an accurate estimate of each neuronal trajectory. In general, we cannot take large time-steps with Runge-Kutta based methods if the I&F neuronal equations are stiff. More precisely, if the exact solution of the I&F neuronal equations over the desired numerical time-step is not accurately approximated by a low (usually 2nd) order polynomial, then the explicit Runge-Kutta schemes do not perform well. This can happen if the conductances are high. One immediate consequence is that the time-step is forced to be relatively small in order to evolve a system with relative strong cortico-cortical connection strengths. If the system is large (i.e., a large number of neurons) and a long-time simulation (say, seconds) is required, then the overall computational work can be large. In particular, we note that the real cortex may often operate in a high conductance state (Borg-Graham et al., 1996, 1998; Pare et al., 1998; Shadlen et al., 1998; Destexhe et al., 2003; Rudolph and Destexhe, 2003a, 2003b, 2003c, 2003d). This makes the problem especially acute in simulating realistic cortical dynamics. Incidentally, we note that the commonly used numerical methods (Hansel et al., 1998; Shelley and Tao, 2001), with the spike-time of a single neuron trace within a single numerical time-step approximated by means of polynomial interpolation, are of the explicit Runge-Kutta kind. Therefore, if the I&F neuronal network equations are stiff, the same numerical issue would arise to resolve the I&F dynamics when using these modified Runge-Kutta methods (Hansel et al., 1998; Shelley and Tao, 2001). The modified Runge-Kutta methods take relatively small time-steps (usually 0.02 ms or smaller, where $\tau \approx 2$ ms usually is the shortest time-scale imbedded into the I&F neuronal equations) in order to calculate each neuronal trajectory accurately.

If we desire a numerical method for an I&F neuronal network that can achieve statistical accuracy when using very large time-steps, as well as trajectory-wise accuracy when using small time-steps, then we need to address issues arising from large time-steps, such as time-steps comparable to the conductance decay time-scale (say, ~ 2 ms), in order to achieve statistical accuracy. In particular, we need to take into account the causality of spiking events within a single time-step via *spike-spike interactions*.[2] Usually, in the modified Runge-Kutta methods, at the beginning of a time-step, the state of the network at the end of the step is not known, thus, only the spikes of the feedforward input to the network within that time step can be used to attempt to evolve the system. This first approximation may indicate that, driven by the feedforward input spikes, many neurons in the network fire. However, this conclusion may be incorrect because the first few of these spikes induced within a *large* time-step may substantially affect the rest of the network via *spike-spike interactions* in such a way that the rest of the spikes *within* the time-step are spurious—For example, this happens when a large time-step is used in these methods to evolve a network with strong recurrent inhibition. We note that the modified Runge-Kutta methods do not take into account spike-spike interactions within a single large numerical time-step. As a consequence, when used to evolve a system with strong cortico-cortical coupling strengths, the modified Runge-Kutta methods with interpolated spikes need to take sufficiently small time-steps to have only a few spikes in the

[1] Note that we use *cortico-cortical* interaction to refer the interaction between neurons in a model network, in contrast to the driving of the external (i.e., feedforward) input to the model network.

[2] That is, the first few approximate spikes within the interval (as determined via feedforward input) interact, via cortico-cortical couplings, and affect the rest of the approximate spikes within the interval. We call this process spike-spike interactions.

entire system within a single time-step. Again, if the system involves a large number of neurons, the computational cost can become prohibitive.

Experimental data indicate that the V1 network architecture may include strong relatively nonspecific local recurrent connections (Fitzpatrick et al., 1985; Lund, 1987; Callaway and Wiser, 1996; Callaway, 1998; Marino et al., 2005) and modulating, orientation-specific, long-range connections, while the dynamics of the system may stabilize in a fluctuation-driven regime with moderate firing rate (Anderson et al., 2000; Stern et al., 1997; Cai et al., 2004, 2005). Therefore, each V1 neuron is only strongly connected to its nearby spatial neighbors within a local interaction spatial scale. The effect of each spike in V1 is thus relatively local, and each neuron in the system can fire once with only $\mathcal{O}(N)$ essential (strong) neuron-neuron interactions, where N is the number of neurons in the system. In order to explicitly take advantage of the specialized V1 cortical architecture, we need to address related numerical issues. Usually, when a general I&F neuronal solver is used for evolving each I&F neuron in a large network, the effect of a neuronal spike on each other neuron in the network is treated computationally independently and equally. Thus, if each of the N neurons in the system fires once, the number of computed neuron-neuron interactions grows like $\mathcal{O}(N^2)$. This very general computational structure does not reflect the V1 network architecture. Therefore, for networks which span a relatively large patch of cortical space with a V1-like architecture, the standard methods are inefficient, spending a lot of work resolving minuscule effects coming from distant neurons.

In this paper, we propose a numerical method which addresses the issues raised above, namely, issues related to (*i*) stiffness in high-conductance states of neuronal networks, (*ii*) causality of spiking events induced by spike-spike interactions with a single large numerical time-step in order to achieve statistical accuracy, and (*iii*) strong spatially local interactions and modulating long-range interactions in V1-like architectures. The basic ideas underlying our method are:

1. We choose an integrating factor, which takes advantage of the neurophysiological fact that the membrane potential of a neuron is nearly slaved to the effective reversal (slaving) potential in a high conductance state, to allow us to write the solution as a numerically tractable integral equation. This enables us to stably and accurately evolve individual neuronal trajectories, even when the I&F equations are stiff (i.e., large time-steps can be taken even for high conductance states).
2. We sort the approximated spike-times within each numerical time-step and apply an iterated correction procedure to account for the effects of each spike on all future spikes *within* the numerical time-step. Thus, when applied to networks of I&F neurons, our method can account for the spike-spike interactions within a single *large* (1 ms–2 ms) numerical time-step. We emphasize that these spike-spike corrections for *a group of neurons* should not be confused with the correction of spike-time by polynomial interpolation for a *single neuron* trajectory in the modified Runge-Kutta methods (Hansel et al., 1998; Shelley and Tao, 2001).
3. We sort the neurons into local clusters approximately as large as the local interaction spatial scale, and consider only $\mathcal{O}(N^0)$ nearby cortical neurons per spike (or equivalently, each time-step is chosen such that it only involves $\mathcal{O}(N^0)$ spikes affecting each neuron). This allows us to evolve V1-like model networks (with N neurons and spatially local interaction kernels) in an asymptotically optimal way such that each neuron fires approximately once in $\mathcal{O}(N^1)$ operations.

In addition to achieving a trajectory-wise accuracy when small time-steps are used in our method, one of the primary advantages of our method is that it can take very large time-steps (even in a high conductance state, for instance), typically 1–2 ms—close to the smallest time-scale in I&F neuronal networks, while still obtaining a good statistically quantitative picture of the V1-like model network behavior. Specifically, statistical information about the network (firing rates, interspike interval distributions, conductance and voltage distributions, spatiotemporal conductance or voltage patterns) can be obtained much more easily and accurately than with the standard modified Runge-Kutta methods. We have implemented our method in a large-scale computational model of a V1 patch ($\sim$25 mm^2 with $\sim 10^6$ neurons) to address spatiotemporal dynamics of V1 (Cai et al., 2005; Rangan et al., 2005), such as coherent, spontaneous ongoing spatiotemporal activity observed by *in vivo* imaging based on voltage-sensitive dyes (Tsodyks et al., 1999; Kenet et al., 2003; Grinvald and Heildesheim, 2004), and V1 spatiotemporal patterns of cortical activity under the line-motion illusion stimulus paradigm (Jancke et al., 2004; Grinvald and Heildesheim, 2004). Incidentally, we comment that, to model these cortical spatiotemporal activities, purely event-driven algorithms (for example, see Makino, 2003; Rochel and Martinez, 2003; Lytton and Hines, 2005; Geisler, 2005) are not suitable, since this type of modeling requires information about continuous internal states described by neurons' voltages and conductances. Event-driven algorithms usually are restricted to very special classes of model neurons and are interested mainly in spike dynamics (Morrison et al., 2005). Generally, it is difficult to use event-driven algorithms to simulate networks with relatively realistic physiological features. Finally, we point out that the applicability of our numerical methods is not limited to V1 modeling since the issues that our methods address are quite general and our methods can be easily extended to model other neuronal networks arising from computational neuroscience in general.

The paper is organized as follows: In Section 2, a brief description of the I&F equations is presented; In Section 3, we describe how a single test neuron can be accurately integrated within a larger network; In Section 4, we describe how to take into account the spike-spike interactions among a group of neurons within a single time step. In Section 5, we describe how to structure the algorithm to take account of the local architecture present in large-scale computational V1 models; Finally, in Section 6, we provide numerical examples which illustrate the power of our method.

2. Integrate-and-fire neurons

For reference, we briefly describe the standard conductance-based integrate-and-fire (I&F) point neuron (Koch, 1999). Given a network of N I&F neurons with label i, each with voltage V_i and conductances G_i^Q, its dynamics is governed by

$$\frac{d}{dt}V_i(t) = -G^L(V_i(t)-\epsilon^L) - \sum_Q G_i^Q(t)(V_i(t)-\epsilon^Q) \tag{1a}$$

$$\frac{d}{dt}G_i^Q(t) = -\frac{G_i^Q(t)}{\sigma^Q} + \sum_j\sum_k S_{i,j}^Q\delta(t-T_{j,k}) + \sum_k F_i^Q\delta\big(t-T_{i,k}^F\big). \tag{1b}$$

The voltage $V_i(t)$ evolves continuously until it reaches a threshold $V_i(t)=\epsilon^T$. At this point in time the ith neuron produces a spike (the kth spike of neuron i is recorded as $T_{i,k}$), and the voltage V_i is reset to ϵ^R, and held there for an absolute refractory period of τ_{ref} ms. Here, G^L is the leak conductance and ϵ^L is leakage potential. The index Q runs over the types of conductances used. These conductances are characterized by their different decay time scales σ^Q and reversal potentials ϵ^Q. Each spike from the ith neuron gives rise to an instantaneous increase in the Q-type conductance of neuron j of magnitude $S_{i,j}^Q$. These coupling strengths $S_{i,j}^Q$ can encode many different types of network architecture (e.g. inhibitory and excitatory type neurons with sparse isotropic local connections). The system is also driven by feedforward input. The kth input spike from the feedforward input to the ith neuron is denoted by $T_{i,k}^F$, and instantaneously increases that neuron's Q-type conductance by magnitude F_i^Q. Typical parameters (in our reduced-dimensional units (McLaughlin et al., 2000), in which only time retains dimension, with units of conductance being [ms $^{-1}$]) are: $G^L=0.05, \epsilon^L=0$, $\epsilon^T=1, \epsilon^R=0, \tau_{ref}=2$ ms. Each neuron is either excitatory or inhibitory, as indicated by its type $\mathcal{L}_i\in\{E,I\}$. There are two conductance types $Q\in\{E,I\}$ also labelling excitation and inhibition. The excitatory (inhibitory) conductance G^E (G^I) of any neuron is increased when that neuron receives a spike from another excitatory (inhibitory) neuron within the network. This is achieved as follows: The coupling strengths $S_{i,j}^E$ are zero whenever $\mathcal{L}_j=I$, and similarly $S_{i,j}^I$ is zero whenever $\mathcal{L}_j=E$. The conductance time-scales are $\sigma^E=2$ ms, $\sigma^I=7$ ms and the reversal potentials are $\epsilon^E=14/3, \epsilon^I=-2/3$. Note that the corresponding physiological values are $\epsilon^E=0\,\text{mV}$, $\epsilon^I=-80\,\text{mV}$, $\epsilon^L=-70\,\text{mV}$, $\epsilon^T=-55\,\text{mV}$ and $G^L=50\times10^{-6}\Omega^{-1}\text{cm}^{-2}$.

We note that, for clarity of presentation, we describe our numerical method below for the case in which there is only one time-scale σ^Q associated with each type of conductance as in Eq. (1b). However, our method can be readily extended to the case, where, for example, the conductances are of the form of an α-function with both rise and decay time-scales.

3. Single 'test' neuron

As a first step towards a stable and accurate method for general neuronal networks, we consider a special case of Eq. (1). We assume that there are $N+1$ neurons within a network such that for a fixed i, $S_{j,i}^Q=0, \forall j$—that is, the ith neuron does not connect to any of the other N neurons in the system (i.e., a 'test' neuron within a network, which is an analog of test particle, commonly used in physics). We further assume that the behavior of the rest of the system is given—every presynaptic spike $T_{j,k}$ for $j\neq i$ and $T_{i,k}^F$ is given as input to the computation. We discuss accuracy and stability issues arising from numerical methods that approximate the voltage, conductance and spike-times for the ith neuron (the test neuron).

Note that Eq. (1) can be rewritten in terms of the total conductance

$$G_i^S(t) = G^L + \sum_Q G_i^Q(t), \tag{2}$$

and effective reversal potential

$$V_i^S(t) = \frac{1}{G_i^S(t)}\left[G^L\epsilon^L + \sum_Q G_i^Q(t)\epsilon^Q\right], \tag{3}$$

as

$$\frac{d}{dt}V_i(t) = -G_i^S(t)\big[V_i - V_i^S(t)\big] \tag{4a}$$

$$\frac{d}{dt}G_i^Q(t) = -\frac{1}{\sigma^Q}G_i^Q(t) + \sum_j\sum_k S_{i,j}^Q\delta(t-T_{j,k}) + \sum_k F_i^Q\delta\big(t-T_{i,k}^F\big). \tag{4b}$$

Classical explicit Runge-Kutta and multistep methods have finite domains of stability (Gear, 1971), and will have stability problems solving Eq. (4) if the conductances G_i^Q are high and the stepsize Δt is large. Standard linearly stable schemes like implicit Euler method tend to be of low order accuracy when applied to smooth ODEs (Gear, 1971), and may not be very accurate if Δt is large. We propose a specialized scheme based on the analytical formula for the exact solution of Eq. (4). This scheme can be easily and quickly applied with high accuracy and no stability constraints. Given an initial time t_0 and a later time $t = t_0 + \Delta t$, we may write the solution $V_i(t), G_i^Q(t)$ as a function of the known input spike-times $T_{j,k}, T_{i,k}^F$:

$$G_i^Q(t) = G_i^Q(t_0)\exp\left(-\frac{t}{\sigma^Q}\right) \tag{5a}$$

$$+\sum_{j\neq i}\sum_{k|T_{j,k}\le t} S_{i,j}^Q \exp\left(-\frac{t-T_{j,k}}{\sigma^Q}\right) \tag{5b}$$

$$+\sum_{k|T_{i,k}^F\le t} F_i^Q\exp\left(-\frac{t-T_{i,k}^F}{\sigma^Q}\right) \tag{5c}$$

$$V_i(t) = V_i^S(t) + \exp\left(-\int_{t_0}^t G_i^S(s)ds\right)\Big(V_i(t_0) - V_i^S(t_0)\Big) \tag{6a}$$

$$-\int_{t_0}^t \exp\left(-\int_s^t G_i^S(r)dr\right)\frac{d}{ds}V_i^S(s)ds. \tag{6b}$$

Note that the voltage representation of Eq. (6) takes advantage of the fact that $V_i(t)$ decays to $V_i^S(t)$ with time-scale given by the value of G_i^S, and is substantially different from the straightforward standard integrating factor representation:

$$V_i(t) = V_i(t_0)\exp\left(-\int_{t_0}^t G_i^S(s)ds\right) + \int_{t_0}^t \exp\left(-\int_s^t G_i^S(r)dr\right)G_i^S(s)V_i^S(s)ds. \tag{7}$$

If the conductances are large, the quantity $G_i^S(s)V_i^S(s)$ in the integrand of Eq. (7) may change very rapidly over the course of a large time-step, so that it may be difficult to evaluate the integral numerically (quadrature requires that the integrand be well approximated by a polynomial). On the other hand, the integrand of Eq. (6) contains the term

$$\frac{d}{dt}V_i^S(t) = \frac{1}{G_i^S(t)}\left[\frac{1}{G_i^S(t)}\left(G^L\epsilon^L + \sum_Q G_i^Q(t)\epsilon^Q\right)\times\left(\sum_Q\frac{G_i^Q(t)}{\sigma^Q}\right) - \sum_Q G_i^Q(t)\frac{\epsilon^Q}{\sigma^Q}\right], \tag{8}$$

which is bounded and well behaved as the conductance values $G_i^Q(t_0)\to\infty$. Thus, the integrand of Eq. (6) is well approximated with a standard low (e.g., 2–3) order quadrature scheme even when Δt is large. The solution given by Eq. (5) for G_i^Q is valid for all time, but Eq. (6) for $V_i(t)$ is only valid as long as $V_i(t)\le\epsilon^T$. That is, as long as the ith neuron does not spike. If the ith neuron does spike at time $T_{i,k}$ (see below for how to determine $T_{i,k}$ numerically), then $V_i(t) := \epsilon^R$ throughout the interval $T_{i,k}\le t\le T_{i,k}+\tau_{ref}$ and for times t after $T_{i,k}+\tau_{ref}$ up until the ith neuron spikes again, say at time $T_{i,k+1}$, we must modify Eq. (6) to read

$$V_i(t) = V_i^S(t) + \exp\left(-\int_{T_{i,k}+\tau_{ref}}^t G_i^S(s)ds\right)\times(\epsilon^R - V_i^S(T_{i,k}+\tau_{ref})) \tag{9a}$$

$$-\int_{T_{i,k}+\tau_{ref}}^t \exp\left(-\int_s^t G_i^S(r)dr\right)\frac{d}{ds}V_i^S(s)ds. \tag{9b}$$

Now we outline the numerical scheme we use to approximate solutions to Eqs. (5) and (6). First the incoming spikes are sorted into an increasing sequence of spike-times. This allows us to use Eq. (5) to compute the conductances $G_i^Q(t)$ exactly for all required times. Then we use these conductance values to exactly compute the quantities $G_i^S(t)$, $V_i^S(t)$, $\frac{d}{dt}V_i^S(t)$ using Eqs. (2), (3) and (8), respectively; Since $G_i^S(t)$ is a sum of exponential functions, $\int_s^t G_i^S(r)dr$ can also be computed exactly. Next, using these computed values, we apply a quadrature scheme to approximate the integrand in Eq. (6)—$\int_{t_0}^t \exp(-\int_s^t G_i^S(r)dr)\frac{d}{ds}V_i^S(s)ds$—which, in turn, allows us to approximate $V_i(t)$. If our approximation to V_i crosses ϵ^T at some time $T_{i,k} < t_0 + \Delta t$, we use polynomial interpolation to determine the spike-time (Hansel et al., 1998; Shelley and Tao, 2001), reset the voltage appropriately, and recompute future values of $V_i(t)$ using Eq. (9). In Appendix A, we detail this numerical scheme—Algorithm 8.1.

This procedure (Algorithm 8.1) has nice properties: (i) If the spike-times are exact, then the conductances are computed exactly, regardless of the time-step; (ii) the conductances will always be positive (in contrast to those computed using a standard explicit Runge-Kutta scheme with too large a time-step).

This method allows us to take very large time-steps while maintaining accuracy. It requires $\mathcal{O}(KP)$ operations per time-step, where P is the quadrature order used in Algorithm 8.1. Clearly, one way to make this method faster and suitable for larger systems is by limiting P (the quadrature order), or K (the number of spikes considered in a time-step).

It is possible to incorporate other simple network mechanisms into Algorithm 8.1. For example, synaptic depression and spike frequency adaptation can be modeled by associating various dynamic relaxation variables with each neuron, which control the neuron's postsynaptic or presynaptic coupling strengths. These dynamic variables exponentially relax to a rest value and jump whenever the neuron spikes (or receives a spike). The equations for these parameters can be evolved in a manner similar to Eq. (5). Synaptic failure can be modeled by associating a Bernoulli random variable with each cortical spike. Axonal delay times can be modeled by associating with each presynaptic cortical spike from the jth neuron $T_{j,k}^{\text{source}} := T_{j,k}$ a list of postsynaptic impinging spike-times $T_{i,j,k}^{\text{target}}$ which record the times at which the original spike reaches and affects the ith neuron (by incorporating the appropriate delays). These impinging spike-times can then be sorted and corrected in a manner similar to the treatment of the $T_{j,k}$ above.

4. General coupling

Now we consider the general case of Eq. (1), where the intercortical connection strengths $S_{i,j}^{Q}$ are in general nonzero. We assume that all the feedforward input spike-times $T_{i,k}^{F}$ $\forall i, k$ are given, but that the cortical spike-times are yet to be determined.[3] We describe a numerical method which approximates the conductances, voltages and spike-times of all the neurons in the network, and allows us to collect accurate statistical information with a low computational cost. A particularly noteworthy feature of this method is that, with relatively large time-steps, it still accurately evolves the network.

One simple approach is to evolve the trajectories of each neuron in the network from t_0 to $t_0 + \Delta t$ by using Algorithm 8.1, while only considering the feedforward spikes within the time interval $[t_0, t_0 + \Delta t]$, and applying the effects of the cortical spikes within the time-interval $[t_0, t_0 + \Delta t]$ at the end of the step. As mentioned in the Introduction, this naive approach will work well for certain systems, such as an all-to-all mean-driven weakly coupled excitatory network, since the contribution of each spike to the overall conductance of this system is negligible, and the errors introduced at each step do not substantially alter the network dynamics. However, this naive approach may fail disastrously when applied to a strongly recurrent system. For example, if we use this simple method with large time-steps ($\Delta t \approx \sigma^{I}$) to evolve a very strongly inhibitory all-to-all autapse-free ($S_{i,i}^{Q} := 0 \; \forall \; i$) network, driven by excitatory feedforward input, we will likely surmise that the inhibitory neurons fire in clustered groups. This conclusion arises because the naive method ignores the cortical spike-spike interactions. One large time-step allows for the excitatory feedforward input to initiate many neuronal spikes, but the simple scheme does not take into account the fact that the first among these spikes may influence the trajectories of the other spiking neurons in the network. A more accurate method may reveal such a scenario, e.g., the first inhibitory neuron to fire may suppress all other network activity, and thus this single neuron may continue to fire alone, driven by the feedforward input.

We briefly outline a variation of Algorithm 8.1 which accounts for spike-spike interactions, and can accurately evolve recurrent networks governed by Eq. (1). (A detailed description of this numerical scheme is also included as part of Algorithm 5.1.)

Algorithm 4.1

1. *Given initial values $V_i(t)$, $G_i^{Q}(t)$ and feedforward input spike-times $T_{i,k}^{F}$ for all i, we can apply Algorithm* 8.1 *with a quadrature order of* 1 *or* 2 *and time-step Δt to obtain an estimate of neuronal trajectories on the interval* $[t_0, t_0 + \Delta t]$.
2. *We use this rough estimate to classify the neurons into two groups $\mathcal{A}^{spike}$, $\mathcal{A}^{\text{quiet}}$—those that are estimated to fire within $[t_0, t_0 + \Delta t]$ and those that are not.*
3. *We sort the approximate spike-times of neurons within $\mathcal{A}^{spike}$ into a list $\mathcal{T}$ with associated coupling strengths $\mathcal{S}$.*
4. *We use the feedforward spikes $T_{i,k}^{F}$ as well as the approximate spike-times $\mathcal{T}$ and Algorithm* 8.1 *with a quadrature of higher order P, (e.g., $P = 3, 4$), and the same time-step Δt to correct the neuronal trajectories of the subnetwork $\mathcal{A}^{spike}$ over the interval $[t_0, t_0 + \Delta t]$, and obtain a more accurate approximation to the spike-times $\mathcal{T}$. This spike-spike correction procedure is equivalent to stepping through the list $\mathcal{T}$ and computing the effect of each spike on all future spikes. We step through this correction process until the neuronal trajectories and spike-time of neurons in $\mathcal{A}^{spike}$ converge. (see Appendix B for further details.)*
5. *Finally we use the corrected estimates of the spike-times $\mathcal{T}$ as well as the feedforward spikes $T_{i,k}^{F}$ and Algorithm* 8.1 *with high quadrature order P and time-step Δt to evolve the remainder of the neurons $\mathcal{A}^{\text{quiet}}$ from t_0 to $t_0 + \Delta t$.*

[3] Note that we use *cortical spikes* to refer to the spikes from the neurons in a model network, in contrast to the spikes from the feedforward input.

This procedure works well most of the time, but it fails if the final evolution step using the corrected spike-times $\mathcal{T}$ causes a neuron from group $\mathcal{A}^{\text{quiet}}$ to fire. If this is the case, we redefine the list $\mathcal{A}^{spike}$ to include this extra neuron, and go back to step 3 and re-correct the spike-times $\mathcal{T}$ starting with the recently discovered spike. Obviously, a missed spike is costly, and slows down the computation. In order to minimize the frequency of these events, it is best to originally define $\mathcal{A}^{spike}$ (within step 2) to include not only the neurons which are estimated to spike due to the feedforward inputs, but also those neurons that have voltage trajectories or effective reversal potentials $V^S(t)$ 'close' to ϵ^T. The exact criteria of 'closeness' used in the algorithm may depend on the network being modeled as well as the dynamical regime. One method of defining $\mathcal{A}^{spike}$ is the '*one away*' approach: After using the known feedforward spikes to determine approximate neuronal trajectories, we classify a neuron as 'close to spiking' if a single extra nearby excitatory cortical spike at the start of the time-step would cause the neuron's effective reversal potential $V^S(t)$ to cross threshold. Another method of defining $\mathcal{A}^{spike}$ is the '*m away*' approach: A neuron is deemed to be 'close to spiking' if the feedforward input combined with the expected cortico-cortical excitatory conductance contribution (calculated using the the average excitatory firing rate m from the previous time-step) brings the effective reversal potential $V^S(t)$ of that neuron over threshold. A combination of these two approaches (detailed later in step 7 of Algorithm 5.1) is shown to work well for the V1 networks studied in the following sections. In practice, this combined procedure only misses about 0.1%–1% of the spiking neurons.

The procedure above also slows down if the set $\mathcal{A}^{spike}$ is very large. A simple strategy to overcome this is to adaptively choose Δt so that there are a manageable number of spikes within each time-step.

5. Arbored coupling

We consider a special case of Eq. (1) which is applicable to models of V1 (Somers et al., 1995; Troyer et al., 1998; McLaughlin et al., 2000; Cai et al., 2005; Rangan et al., 2005), as well as many other cortical areas. For ease of discussion, we make the following assumptions, many of which can later be relaxed:

1. The neurons in the system are spatially organized on a two-dimensional lattice. The spatial density of lattice points is 1 lattice point per unit area. The position of the ith neuron is given by its integer cortical coordinates $\mathbf{x}_i$.
2. There are two types of conductances $Q \in \{E, I\}$, corresponding to excitation and inhibition. Each neuron in the system has a type $\mathcal{L} = \{E, I\}$ labelling either excitatory or inhibitory. A spike from an excitatory (inhibitory) neuron only increases the excitatory (inhibitory) conductance of its postsynaptic neurons. These two types of neurons are uniformly randomly distributed with density ρ^E, ρ^I respectively.
3. The neurons are isotropically randomly connected to other nearby neurons, with interaction strengths $S^Q_{i,j} = \delta_{Q,\mathcal{L}_j}\Delta_{i,j}\bar{S}^Q_{\mathcal{L}_i}K^Q(|\mathbf{x}_i - \mathbf{x}_j|)$. The Kronecker $\delta_{Q,\mathcal{L}_j}$ reflects that a spiking neuron can only increase the conductance associated with its type. The random connectivity matrix $\Delta_{i,j}$ indicates whether neuron j is connected to neuron i. The diagonals $\Delta_{i,i}$ are all zero, and the percentage of nonzero entries (sparsity coefficient) is $\bar{\Delta}$. The maximum strength $\bar{S}^Q_{\mathcal{L}_i}$ only depends on the type of conductance and the type of the postsynaptic neuron. The normalized spatial kernel $K^Q(\mathbf{r})$ decays faster than $1/r^2$ (in three dimensions we would require the decay to be faster than $1/r^3$). A commonly used kernel is the Gaussian spatial kernel $K^Q(\mathbf{r}) = \bar{K}^Q \exp(-r^2/(D^Q)^2)$, which decays with spatial scale D^Q.

We denote the average Q-type conductance of the $\mathcal{L}$-type neurons within the system by $\bar{G}^Q_{\mathcal{L}}$, and the average instantaneous firing rate of the $\mathcal{L}$-type neurons by $m^{\mathcal{L}}$.

Given the initial conditions $V_i(0)$, $G^Q_i(0)$ and feedforward input spikes $T^F_{i,k}$ to the system, we want to approximate the individual neuronal trajectories and obtain accurate statistical information about neuronal ensembles. The method we propose performs very well when the network is in a fluctuation-driven, high conductance regime with low to moderate firing rate (Cai et al., 2004, 2005), and can resolve statistical features of the network behavior much more easily than standard numerical methods.

The basic idea behind our method is to divide the neurons into local groups (arbors) with diameter roughly equal to the largest interaction length scale D^Q. We adaptively choose the time-step Δt so that an initial estimate of neuronal trajectories (using only the feedforward spikes) predicts only a few neurons in each arbor with the potential to fire during the time-step Δt. We group the potential firing events into local clusters such that the firing events in each cluster are spatially well separated from those in other clusters. Then we focus only on the strong close range connections and correct our estimates for the spike times within each cluster. We use these locally corrected spike-times as well as the feedforward spikes to evolve the non-spiking neurons within each cluster. Finally, we account for the weak long distance connections by tracking the average firing rate of each arbor and updating the conductances of non-adjacent arbors with a low order (mean firing rate) method (see Fig. 1). We note in passing that similar ideas have been explored in molecular dynamics simulations—weaker or slowly changing far field

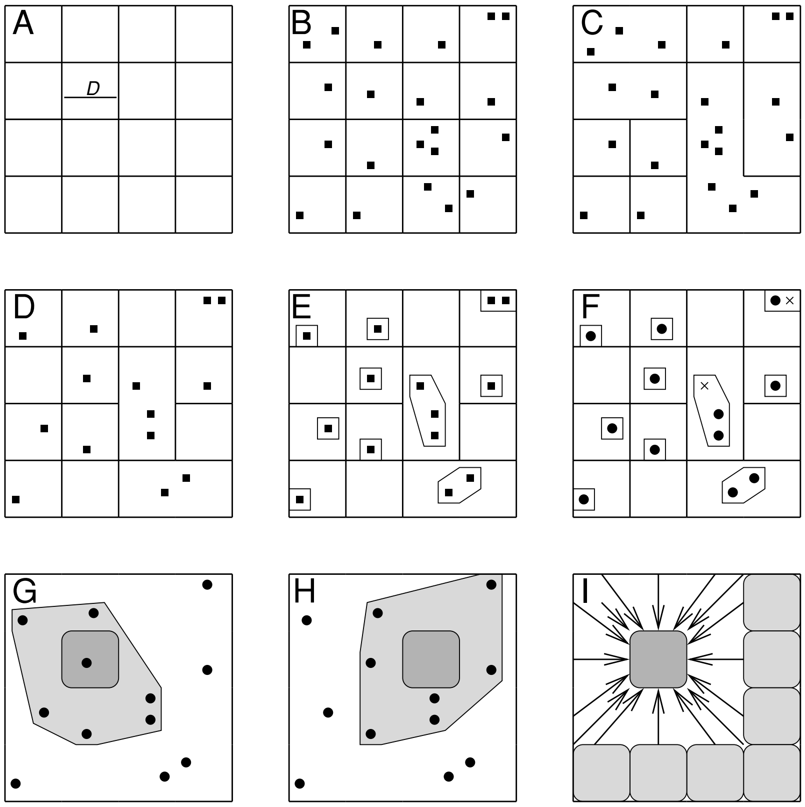

Fig. 1 *Illustration of Algorithm* 5.1. (A) First, space is divided into local regions (arbors), each of a fixed diameter D. D is specifically chosen such that the expected conductance contribution to any arbor from spikes occurring within non-adjacent arbors is small. Here the large square represents a region of cortex, and the smaller squares represent arbors of diameter D. (B) In a single time-step from T_0 to $T_0 + \Delta t$, the information of the feedforward input spike-times $T^F_{i,k}$ is used to estimate (predict) which neurons are likely to fire within the present time-interval. Here, the neurons at locations marked by "■" are classified as 'close to spiking' within the time-interval $[T_0, T_0 + \Delta t]$. (C) The neurons that are 'close to spiking' are grouped into local clusters. Each local cluster contains potentially firing neurons which are spatially sufficiently close to one another that they might strongly influence each others' trajectories within the time-interval $[T_0, T_0 + \Delta t]$. Here each black bordered region corresponds to a different cluster of potentially spiking neurons. Note that, as illustrated here, one cluster has 7 potentially spiking neurons. Suppose that 7 is higher than the upper cluster limit L, say 5. Therefore, the seven-neuron-cluster contains too many potentially interacting spiking neurons for us to consider. (D) If there are too many potentially interacting spiking neurons, the computation is restarted from the initial time T_0 and a smaller time-step $\widetilde{\Delta t} = \Delta t/2$ is taken. With this smaller time-step, the estimate of which neurons are likely to fire within the (smaller) time-interval $\left[T_0, T_0 + \widetilde{\Delta t}\right]$ is again performed using only feedforward spikes. These potentially firing neurons are grouped into clusters. This iteration proceeds until there are not many neurons in each cluster (For example, the largest cluster in Panel D has 3 neurons, which is smaller than the aforementioned threshold L). (E) Now we restrict our attention to each cluster of potentially spiking neurons, and ignore all the other neurons within each arbor that are not 'close to spiking'. Here the thinly bordered regions indicate the spatially well separated (non-interacting) groups of potentially spiking neurons. (F) In turn, for each cluster of potentially spiking neurons, the estimated spike-times within the time-interval $\left[T_0, T_0 + \widetilde{\Delta t}\right]$ are corrected by using a high-order scheme. These spike-spike corrections may reveal that some of the neurons originally estimated to fire during the time-interval $\left[T_0, T_0 + \widetilde{\Delta t}\right]$ actually do not spike within this interval (as indicated by '$\times$') while other potentially firing neurons do still fire within the interval, albeit with slightly different spike-times (as indicated by '$\bullet$'). We emphasize that this spike-spike correction procedure provides a much more accurate picture of spiking activity $T_{j,k}$ within the time-interval $\left[T_0, T_0 + \widetilde{\Delta t}\right]$ than the original set of spike-times computed using only the feedforward input spike-times $T^F_{i,k}$. (G) Once accurate estimates of all the neuronal spike-times $T_{j,k}$ within the time-interval $\left[T_0, T_0 + \widetilde{\Delta t}\right]$ are obtained, the trajectories of the non-spiking neurons within an arbor (indicated by the dark grey region) can be accurately evolved by taking into account the spiking neurons located in the adjacent arbors (indicated by the light grey region) as well as the feedforward input. (H) For each arbor in turn, all the non-spiking neurons are updated. Here the dark gray region indicates the arbor of non-spiking neurons that are being updated, and the light grey region indicates the cortical spikes which can strongly affect those neurons. (I) Finally, after all the non-spiking neurons in the network are updated, the far field conductance contribution to each neuron can be approximated using the mean firing rate within each non-adjacent arbor (the conductance contribution from adjacent arbors has already been considered since the nearby spiking neurons in steps G–H are specifically accounted for). Here the dark grey region indicates the arbor that is being updated, and the light gray regions indicates the (other) arbors (not adjacent to the dark grey region) which are accounted for with mean firing rates and average connectivity strengths.

effects are treated differently from stronger quickly changing local effects (Frenkel and Smit, 1996).

What follows is a detailed description (See Fig. 1):

Algorithm 5.1. Arbored Coupling Method

1. *Input: an initial time t_0, a maximum time-step $\Delta t_{\max}$, an upper limit L on the number of interacting spikes we wish to consider, and quadrature order P. Typical values are $\Delta t_{\max} \approx \tau_{ref}$ or $\Delta t_{\max} \approx \max_Q \sigma^Q$, $L \approx 20$, $P = 2$*
2. *Set $\Delta t := \Delta t_{\max}$.*
3. *Use the estimates of average conductance and firing rate $\bar{G}^Q_{\mathcal{L}}, m^{\mathcal{L}}$ to define an arbor diameter D such that, on average, the spikes outside of a neuron's local arbor do not affect its conductance very much. This can be accomplished by choosing D such that $\Delta t m^Q \rho^Q \bar{\Delta} \bar{S}^Q_{\mathcal{L}} \int\int_{B^c} K^Q(\mathbf{r})d\mathbf{r} \leq \Delta t^P_{\max} \bar{G}^Q_{\mathcal{L}}$ for all $Q, \mathcal{L}$, where the area B^c is defined as the complement to the square of side length D centered at the origin. Let $N' = D^2$ denote the number of neurons within B (a square of side length D centered at the origin).*
4. *Partition the lattice of neurons into $M = \lceil N/N' \rceil$ local square arbors $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_M$ each of diameter D. (We abuse the notation $\mathcal{A}_l$ to refer to both a square in space as well as the list of neurons within that square).*
5. *With a time-step of Δt, use Algorithm 8.1 with only the feedforward spikes $T^F_{i,k}$ and a low order quadrature to obtain a rough estimate of neuronal trajectories within each arbor on the interval $[t_0, t_0 + \Delta t]$. (This step takes $\mathcal{O}(N)$ operations).*
6. *Estimate the firing rate $m^{\mathcal{L}}_1, \ldots, m^{\mathcal{L}}_M$ of the neurons within each arbor by tallying the total number of predicted spikes over the time-step Δt, or by using the firing rates from the previous step. (This step takes $\mathcal{O}(N)$ operations).*
7. *Given the approximate neuronal trajectories and approximate firing rates, sort the neurons in each arbor $\mathcal{A}_l$ into two lists: $\mathcal{A}^{spike}_l$ and $\mathcal{A}^{quiet}_l$. Neurons that have an approximate trajectory that crosses or comes close to ϵ^T are put in list $\mathcal{A}^{spike}_l$, and the other neurons are put into list $\mathcal{A}^{quiet}_l$. A neuron (say, with index j) in arbor $\mathcal{A}_l$ is classified as 'close to' ϵ^T if, on average, the local excitatory firing rate estimate m^E_l within that arbor could be expected to drive that neuron to fire. More precisely, if an addition of $\max(1, m^E_l \Delta t \rho^E \bar{\Delta}) \bar{S}^E_{\mathcal{L}_j} \int\int_B K^E(\mathbf{r})d\mathbf{r}$ to the conductance G^E_j causes the effective reversal potential V^S_j to cross ϵ^T, then the jth neuron is likely to fire during the time-step Δt. (This a combination of the 'one spike away' and 'm spikes away' scenarios). The list $\mathcal{A}^{spike}_l$ ($\mathcal{A}^{quiet}_l$) contains the potentially spiking (nonspiking) neurons within the spatial arbor $\mathcal{A}_l$. (This step takes $\mathcal{O}(N)$ operations).*
8. *Define the clusters $\bar{\mathcal{A}}^{spike}_l$ as the set of potentially spiking neurons within arbor $\mathcal{A}_l$. Originally this cluster is simply $\bar{\mathcal{A}}^{spike}_l := \mathcal{A}^{spike}_l$, but we will later concatenate the $\bar{\mathcal{A}}^{spike}_l$ so that they hold disjoint groups of possibly interacting potentially spiking neurons. We define two clusters $\bar{\mathcal{A}}^{spike}_l, \bar{\mathcal{A}}^{spike}_{l'}$ as spatially adjacent if there exist neurons $n_i \in \bar{\mathcal{A}}^{spike}_l$ and $n_{i'} \in \bar{\mathcal{A}}^{spike}_{l'}$ that belong to arbors $\mathcal{A}_k \ni n_i$, $\mathcal{A}_{k'} \ni n_{i'}$ respectively such that $\mathcal{A}_k$ and $\mathcal{A}_{k'}$ are adjacent in cortical space.*
9. *Run through all spatially adjacent pairs of clusters $\bar{\mathcal{A}}^{spike}_l, \bar{\mathcal{A}}^{spike}_{l'}$. If two adjacent clusters contain neurons that are within D of one another, concatenate them into one cluster. (This step takes $\mathcal{O}(N + LM \log M)$ operations in the absolute worst case, but in our experience this step only takes $\mathcal{O}(N)$ operations).*
10. *If any cluster $\bar{\mathcal{A}}^{spike}_l$ has more than L neurons, there are too many spikes which might possibly interact. Halve Δt and go back to step 5. Otherwise continue.*
11. *Construct lists $\bar{\mathcal{T}}_l$ corresponding to the approximate spike-times of the neurons within each cluster $\bar{\mathcal{A}}^{spike}_l$.*
12. *Use Algorithm 8.1 with the feedforward spikes $T^F_{i,k}$ as well as the (less than L) spike-times $\bar{\mathcal{T}}_l$ and quadrature order P to correct the neuronal trajectories for each cluster $\bar{\mathcal{A}}^{spike}_l$ over the time interval $[t_0, t_0 + \Delta t]$. Update the lists $\bar{\mathcal{T}}_l$, and repeat this correction step until the approximate trajectories and spike-times converge (this typically takes 2–3 iterations). (If we limit ourselves to a fixed number of iterations, this step takes $\mathcal{O}(ML^2P)$ operations). (see Appendix B for further details.)*
13. *Construct lists $\mathcal{T}_l$ corresponding to the accurate spike-times of neurons within each arbor $\mathcal{A}_l$. (This step takes $\mathcal{O}(ML)$ operations).*
14. *For each arbor $\mathcal{A}_l$, in addition to the feedforward spikes, consider also the spikes $\mathcal{T}_l$ within that arbor as well as the spikes $\mathcal{T}_{l'}$ corresponding to adjacent arbors, and use Algorithm 8.1 to accurately evolve the nonspiking neurons in each cluster $\mathcal{A}^{quiet}_l$. (There are only 9 adjacent arbors associated with each cluster $\mathcal{A}^{quiet}_l$, and thus, maximally, only 9L cortical spikes can affect neurons within $\mathcal{A}_l$. Therefore this step takes $\mathcal{O}(NLP)$ operations).*
15. *If a neuron originally classified into one of the $\mathcal{A}^{quiet}_l$ spikes over the interval $[t_0, t_0 + \Delta t]$, add this neuron to the appropriate list $\bar{\mathcal{A}}^{spike}_l$ and go back to step 9. If the total number of neurons within the $\mathcal{A}^{quiet}_l$ which fire over the time interval is comparable to the total number of neurons within the $\bar{\mathcal{A}}^{spike}_l$, halve Δt and go back to step 5.*
16. *Calculate the average conductances $\bar{G}^Q_{\mathcal{L}}$ and firing rates $m^{\mathcal{L}}$ of the system, as well as the firing rates $m^{\mathcal{L}}_l$ of each arbor $\mathcal{A}_l$. (This step takes $\mathcal{O}(N)$ operations).*

17. *Use a mean rate method to account for the effect of distant spikes. This can be accomplished by adding* $\tilde{\sum}_l \Delta t m_l^Q \rho^Q \bar{\Delta}\bar{S}_{\mathcal{L}}^Q \int\int_{\mathcal{A}_l} K^Q(\mathbf{r}-\bar{\mathbf{x}}_{l'})d\vec{r}$ *to the Q-type conductance of each $\mathcal{L}$-type neuron within arbor $\mathcal{A}_{l'}$. Here the point $\mathbf{x}_{l'}$ refers to the center of arbor $\mathcal{A}_{l'}$, and the sum $\tilde{\sum}_l$ is over all arbors $\mathcal{A}_l$ not adjacent to $\mathcal{A}_{l'}$. Note that due to the construction of D in step 3, this mean rate update should generally produce a relative error of $\mathcal{O}(\Delta t_{\max}^P)$ in the conductance trajectories. (This can be done with only $\mathcal{O}(N)$ operations by using the fast fourier transform).*
18. *Go back to step 2.*

We note that in practice, each step of Algorithm 5.1 requires $\mathcal{O}(NLP)$ work and calculates $\mathcal{O}(LN/D_{\max}^2)$ spike-times, where $D_{\max}$ is the maximum arbor diameter allowed. If the network fires roughly uniformly, we can compute $\mathcal{O}(N)$ spikes in $\mathcal{O}(D_{\max}^2/L)$ steps, which requires a total of $\mathcal{O}(NPD_{\max}^2)$ operations. This procedure is much faster than the $\mathcal{O}(N^3)$ operations required if we were to explicitly consider the $\mathcal{O}(N)$ far field interactions associated with each spike.

We also note that this method is rather flexible, and can easily be generalized to accommodate different types of network architecture. For example, different types of interaction kernels (with radial decay no slower than exponential or $1/r^n$, with $n > 2$) can be considered. Other neuronal subpopulations can be added, each with their own associated set of interaction strengths.

As experimental observations suggest (Gilbert and Wiesel, 1983; Bosking et al., 1997; Sincich and Blasdel, 2001; Angelucci et al., 2002), regions of V1 which share similar orientation preference (to visual input) but are spatially separated by ≈ 1 mm share no isotropic local connections, but are weakly coupled via long range lateral (LR) connections. One practical application of Algorithm 5.1 involves adjusting the cortical architecture to include these LR connections. For example, we can assume that our network has local connections as given above (in assumption 3) and LR connections given by $S_{i,j}^{LR,Q} = \delta_{Q,\mathcal{L}_j}\bar{S}_{\mathcal{L}_i}^{LR,Q} K^{LR,Q}(|\mathbf{x}_i - \mathbf{x}_j|)$, with maximum coupling strengths $\bar{S}_{\mathcal{L}_i}^{LR,Q}$ and long range coupling kernels $K^{LR,Q}$. There is no need to consider the LR spike-spike interactions if, for example, the long range coupling strengths are small relative to the local cortical strengths, and the number of LR connections per neuron is large. In such a scenario it is reasonable to account for the LR connections by using the mean firing rate within each arbor (or small cortical region) to update the conductances of the other arbors within the network. This LR correction can be taken into account at the same time as the low order mean rate correction in step 16 of Algorithm 5.1.

We remark that the spatial divide-and-conquer strategy implemented in Algorithm 5.1 can be structured to take advantage of multiple processors. More precisely, Steps 5 through 16 of Algorithm 5.1 all involve multiple tasks which each require local information (i.e., estimates of the neuronal trajectories within a few arbors of the entire system), and thus can be computed in parallel. However, in our implementation of this algorithm we only use a single processor, which already allows us to simulate model networks of 10^{5-6} neurons for realistic modeling purposes (see Section 6.4 for details).

6. Numerical results

6.1. Test neuron

We construct a numerical experiment to emphasize the stability and accuracy of Algorithm 8.1. As stated earlier, modified Runge-Kutta methods may work well for mean-driven networks with weak cortical coupling strengths. These same Runge-Kutta methods will fail due to stability constraints if the network is in a fluctuation-driven regime with high conductance states, giving rise to stiff I&F neuronal equations. To emphasize the practicable nature of our algorithm and contrast it with the modified Runge-Kutta schemes, we will consider a single test neuron system, with $S_{i,j}^E$, $i \neq j$, given by a constant S^E when neuron j is excitatory, and with $S_{i,j}^I, i \neq j$, given by constant S^I when neuron j is inhibitory. We use strong cortical strengths $S^E \approx 1$–10, $S^I \approx 10$–40 and a fixed input spike train sampled from a Poisson process with rate $\approx 500\, spikes/s$. This system operates in a fluctuation-driven regime (Cai et al., 2004, 2005) with its neurons firing at a moderate rate ($\approx 50\, spikes/s$). The typical conductances of the neuron are sufficiently high ($G^S \approx 10$–50) that the time-scale $\frac{1}{G^S} \ll \sigma^Q$, therefore, the I&F neuronal equations are stiff. We approximate the exact solution by using a variety of numerical methods (Algorithm 8.1, modified explicit Runge-Kutta, modified implicit Euler methods) with a time-step $\Delta t \approx 2^{-16} \approx 1.5 \times 10^{-5}$ms that is sufficiently small that all of these solutions produce the same convergent answer. We take these convergent solutions as a representation of the exact solution—voltage trajectory $V^{exact}(t)$ and number of spikes K^{exact} over a 1024 ms time interval. We compare this exact solution with the trajectories $V^{\Delta t}(t)$ and total number of spikes $K^{\Delta t}$ calculated using Algorithm 8.1 with fixed quadrature order $p = 2$ and larger time-steps $\Delta t \approx 2^{-3} \to 2^2$ (0.125 ms $\to$ 4 ms). We measure:

1. The relative average error in the voltage

$$\bar{E}^V = \left(\frac{1}{t_F - t_0} \int_{t_0}^{t_F} \left| V^{exact}(t) - V^{\Delta t}(t) \right| dt \right) \Big/ \bar{V}, \quad (10)$$

2. The relative error in the average voltage

$$E^{\bar{V}} = \left| \frac{1}{t_F - t_0} \int_{t_0}^{t_F} V^{exact}(t)dt - \frac{1}{t_F - t_0} \int_{t_0}^{t_F} V^{\Delta t}(t)dt \right| \Big/ \bar{V}, \qquad (11)$$

3. The relative error in the total number of firing events

$$E^K = (K^{exact} - K^{\Delta t})/K^{exact}. \qquad (12)$$

As seen in Fig. 2, the relative errors $\bar{E}^V$ and $E^{\bar{V}}$ obtained using Algorithm 8.1 are both small, even for large time-steps (for example, $\Delta t \approx 1$ ms for two digit accuracy). When the time-step is sufficiently small ($\Delta t \approx 0.25$ ms), Algorithm 8.1 correctly calculates K. However, due to the stiffness of the equation, the spikes are not all exactly at the right times, namely, there are $\mathcal{O}(\Delta t)$ errors in spike-time. Therefore, $\bar{E}^V$ and $E^{\bar{V}}$ are both $\mathcal{O}(\Delta t)$. The errors of Algorithm 8.1 are compared with the errors obtained using modified implicit Euler and modified Runge-Kutta methods of order 2 and 4 with large time-steps. As expected, implicit Euler method is stable, but not very accurate. The modified Runge-Kutta methods suffer from stability constraints, and are useless for large time-steps ($\Delta t \geq 2^{-3} \approx 0.125$ ms). These methods perform well for sufficiently small time-steps ($\Delta t \leq 2^{-4} \approx 0.06$ ms), but require 16 to 32 times as many steps to achieve the same accuracy as Algorithm 8.1 with the appropriate quadrature order $p = 2, 4$.

In summary, Algorithm 8.1 is practicable, and performs well for test neurons within a wide range of networks. This method is free to choose relatively large timesteps ($\Delta t \approx 0.1 - 1$ ms) without restrictions due to stability, therefore, it can achieve a relative good statistical result when large time-steps are used. We emphasize that in addition to statistical convergence using large time-steps, the method can also achieve trajectory-wise accuracy when small time-steps are used.

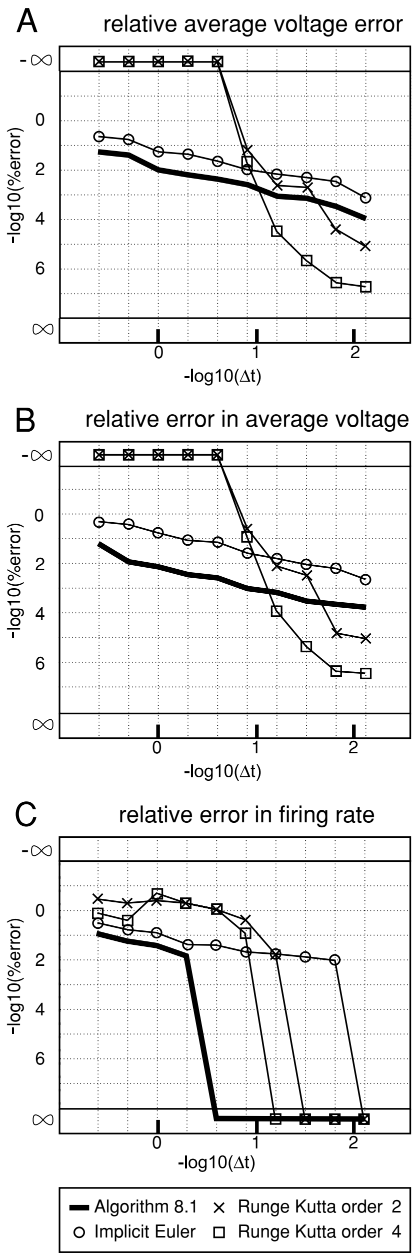

Fig. 2 *Accuracy of Algorithm 8.1.* Error graphs for the single test neuron solver Algorithm 8.1 (with 2nd order quadrature) when tested on a single test neuron with feedforward input spike drawn from a Poisson process with rate ≈ 500 spikes/s, and input strengths $S^E \approx 1 - 10$, $S^I \approx 10 - 40$. The same scale is used for all three sets of axes. The data points (on the vertical axis) at $\pm\infty$ indicate no error or infinite error, respectively. The thick black line corresponds to errors generated by Algorithm 8.1 (for various time-steps, as indicated in the horizontal axis). The '⊙',' × ' and '□' correspond to errors generated by the modified implicit Euler method, the modified Runge-Kutta method of order 2 and the modified Runge-Kutta method of order 4 respectively. (A) The relative average error in the voltage trace of the test neuron (Eq. (10)). Note that Algorithm 8.1 achieves 2 digit accuracy for large time-steps ($\Delta t \approx 1$ ms), whereas implicit Euler achieves 2 digit accuracy only for small time-steps ($\Delta t \approx 0.125$ ms), and the modified Runge-Kutta methods can achieve this only with *very* small time-steps ($\Delta t \leq 0.06$ ms). (B) The relative error in the average voltage (Eq. (11)) of the test neuron over a 1024 ms period. Note that again Algorithm 8.1 outperforms the implicit Euler method, and the modified Runge-Kutta methods can achieve the same accuracy only with very small time-steps. (C) The relative error in the firing rate (the total number of spikes of the test neuron over a 1024 ms period) (Eq. (12)). Algorithm 8.1 calculates the correct number of firing events for a somewhat large time-step ($\Delta t \approx 0.25$ ms), whereas the other methods need a smaller time-step to obtain the same accuracy in firing rate ($\Delta t \approx 0.03$ ms for Runge-Kutta of order 2, and $\Delta t \approx 0.008$ ms for the implicit Euler Method)

6.2. All-to-all coupling

Now we demonstrate the accuracy of Algorithm 4.1 by applying it to a small (≈ 100 neurons) all-to-all coupled test network with physiologically plausible parameters. In particular, we point out that the spike-spike corrections (step 4 in Algorithm 4.1 or step 12 in Algorithm 5.1) can be crucial to accurately capturing the dynamics arising from cortico-cortical coupling within such a network. The network we consider has moderately strong uniform cortico-cortical inhibition $S^I \approx 15$ and moderately weak excitation $S^E \approx 1.5$. We drive the excitatory neurons in the system with excitatory feedforward input spikes, sampled from a Poisson process with sinusoidal input rate proportional to $\sin(2\pi t/10 \text{ ms})$. We use Algorithm 4.1 with a sufficiently small time-step ($\Delta t \leq 2^{-16} \approx 1.5 \times 10^{-5}$ms) to obtain a convergent solution, which we then take as a representation of the exact solution. Computed sample trajectories are displayed in Fig. 3.

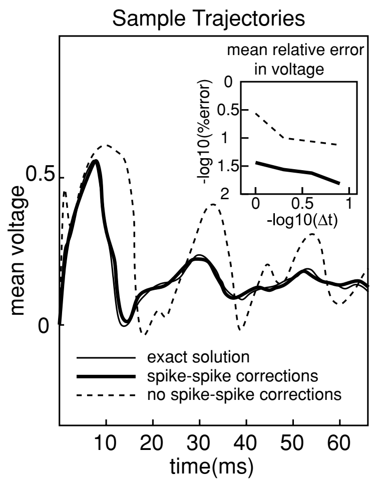

Fig. 3 *Importance of spike-spike corrections for evolution of networks.* Displayed are sample trajectories and error graphs for the general network solver (Algorithm 4.1 with and without spike-spike corrections) when tested on a small all-to-all connected network of ∼25 inhibitory neurons and ∼75 excitatory neurons ($S^I \approx 15$, and $S^E \approx 1.5$). Neurons within this network are driven by input spike-trains which are independent realizations of a Poisson process with sinusoidally varying rate proportional to $\sin(2\pi t/10 \text{ ms})$, (the input is turned on at 0 ms). The average voltage trajectories of the system over a 64 ms interval after stimulus onset are displayed. The thin black line corresponds to the exact solution (as approximated with very small time-step), the thick black line corresponds to the voltage computed with a large time-step ($\Delta t = 1$ ms) by using Algorithm 4.1 (by default, with spike-spike corrections), and the dashed line corresponds to the voltage computed (for the same large time-step $\Delta t = 1$ ms) by using Algorithm 4.1 *except without* spike-spike corrections. Note that, without spike-spike corrections, the general network solver does not accurately resolve the network dynamics. Inset: Average error in voltage for the system for various time-steps. Clearly, the general network solver computes a more accurate set of voltage trajectories with spike-spike corrections than without

This system exhibits behavior which is strongly dependent on the sequence of cortical spikes. The upstroke of the first period of feedforward excitatory input causes a surge of excitatory spikes, which drives a few inhibitory neurons to fire, which then strongly suppress the response of the rest of the system. The time-scale of the inhibitory conductance $\sigma^I \approx 7$ ms is not much shorter than the the period of the sinusoidal input (10 ms), and thus the system is still somewhat inhibited by the time when the second upstroke of input arrives. Therefore, the second (and subsequent) input upstrokes need only trigger a few excitatory spikes (which, in turn, trigger a few inhibitory spikes) before the response of the system is quickly suppressed again. Clearly, the cortical interactions are critically important to the dynamics of the system. For example, if there were a 5 ms axonal delay between the time of each spike and its effect on its postsynaptic neurons, the dynamics of the system could change drastically, since the inhibitory neurons could no longer prevent proximal excitatory spikes. The spike-spike corrections in Algorithm 4.1 allows us to address these situations with large time-steps (with or without delays)(Morrison et al., 2005).

We use Algorithm 4.1 with large fixed time-steps $\Delta t \approx 2^{-3} \to 2^0$ (0.125 ms−1 ms) to simulate this system. The error of computation is quantified by the relative average error in the voltage $\sum_i |V_i^{exact} - V_i^{\Delta t}|/N\bar{V}$, where N is the number of neurons in the system (see the inset in Fig. 3). Even for large time-steps ($\Delta t = 1$ ms), Algorithm 4.1 (by default, with spike-spike corrections) accurately evolves the system as the voltage trajectories for the system closely follow the exact solution. These accurate solutions capture the essential behavior of the network: each input upstroke triggers a few excitatory spikes which recruit a few inhibitory neurons, thus suppressing the rest of the neurons. We compare this method with a simpler, naive method that does not perform spike-spike corrections (i.e., Algorithm 4.1 without spike-spike corrections in step 4). As shown in Fig. 3, this naive method is not accurate, even for small time-steps ($\Delta t \approx 2^{-3} \approx 0.125$ ms), and the voltage trajectories for the system differ significantly from the exact solution. Since this naive method never corrects the spike-times due to cortico-cortical interaction within a time-interval, it overestimates, by a significant amount, the number of firing events that take place within a single time-step during an input upstroke. This results in incorrect voltage trajectories which are far more oscillatory than the actual network (see Fig. 3). This naive method is only accurate when it uses a time-step that is sufficiently small that spike-spike corrections are no longer required within a single time-step to accurately resolve the dynamics. In other words, the naive method can only evolve the system accurately if there are only 1–2 estimated spikes in the *entire* system per time step. Even though the firing rate for individual neurons within this system is ∼60 spikes/*s*, the individual neuron peak firing rates are ∼200 spikes/*s*.

In addition, the firing events within the system are highly synchronized, and often occur within 10^{-1}ms of one another. As a result, in order for the naive method to accurately evolve this test network, the time-step needs to be very small, namely $\Delta t \leq 2^{-7} \approx 8 \times 10^{-3}$ ms.

We emphasize that for this particular system, the error in the naive algorithm does not arise from the stability problem (e.g., stiffness due to high conductance). If we replace the individual neuron solver, Algorithm 8.1, in Algorithm 4.1 with either the modified Runge-Kutta method or the modified implicit Euler method, we obtain similar results, that is, without spike-spike corrections very small time-steps are required to resolve the dynamics. Regardless of the order or stability of a method for the single test neuron case, if the method does not take into account the spike-spike interactions, it will suffer in accuracy when applied to a strongly interacting network, unless the time-step is prohibitively small. As is demonstrated above, with a numerical method that accounts for the spike-spike corrections, one can take relatively large time-steps while still obtaining an accurate answer.

We reiterate that not every system requires these considerations, and significantly simpler networks (with, say, only weak excitatory coupling) can easily be investigated using standard methods. However, Algorithm 4.1 is highly advantageous for examining systems with strong cortical coupling strengths.

6.3. Arbored coupling

To test the efficiency of Algorithm 5.1, we construct the following test network to mimic real physiological architecture in V1.[4] We construct a square lattice of 64×64 neurons, each with integer lattice position $\mathbf{x}_i$ and randomly assigned type $\mathcal{T} = E, I$, corresponding to excitation or inhibition (75% excitatory and 25% inhibitory). In the network, there are three types of conductances, $Q \in \{\text{AMPA,NMDA,GABA}\}$. Both AMPA and NMDA conductances are associated with spikes of excitatory neurons, and have reversal potentials $\epsilon^{\text{AMPA}} = \epsilon^{\text{NMDA}} = 14/3$ and decay constants $\sigma^{\text{AMPA}} = 2$ ms, $\sigma^{\text{NMDA}} = 80$ ms respectively (Wang, 1999; Compte et al., 2003). Inhibitory conductance mediated by GABA is associated with spikes of inhibitory neurons, and has the reversal potential $\epsilon^{\text{GABA}} = -2/3$ and decay constant $\sigma^{\text{GABA}} = 7$ ms. Each neuron is coupled to its lattice neighbors with interaction strength $S^Q_{i,j} = S^Q K^Q(\|\mathbf{x}_i - \mathbf{x}_j\|)$. The three constants S^Q indicate the maximum connection strengths. The normalized Gaussian spatial kernel K^Q only depends on the distance

[4] Here, the test network is designed only for the purpose of illustrating essential computational elements involved in the primary visual cortex modeling. We have implemented our numerical method in more physiologically realistic computational models of primary visual cortex (see Refs. Cai et al., 2005; Rangan et al., 2005 for details).

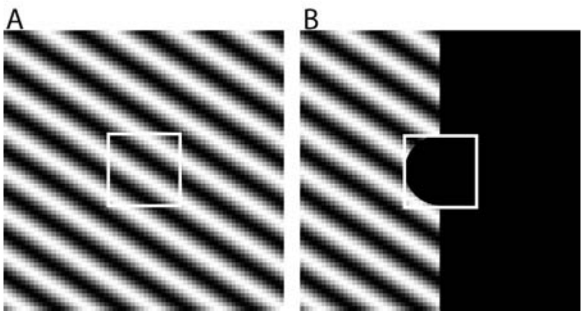

Fig. 4 *Inputs to arbored test system.* Snapshot of drifting grating input and obscured grating input to the test network described in Fig. 5. (A) The large box shown corresponds to the 64×64 neuron system and the small white bordered region corresponds to the 16×16 subset of neurons recorded. The intensity at each square corresponds to the drifting grating input rate to neurons at that cortical point, for a specific time—the grating drifts with a frequency $\omega = 2\pi/64$ ms. (B) The intensity shown indicates the obscured grating input rate to neurons at each cortical point. The center and right half of the cortex is obscured (neurons receive no feedforward input). Neurons inside the measured region receive almost all their input from the rest of the cortex outside the obscured region

between neurons and has spatial scale D^Q. We choose the strengths $S^{\text{AMPA}} = 3$, $S^{\text{NMDA}} = 0.05$, $S^{\text{GABA}} = 7$, and the interaction spatial scales $D^{\text{AMPA}} = 24$, $D^{\text{NMDA}} = 8$, $D^{\text{GABA}} = 16$ for the purpose of illustrating the numerical methods only. The system is driven by independent feedforward excitatory input spike trains to each neuron, which are each specific realizations of a Poisson process with rates $\lambda^{\text{LGN}}(\mathbf{x}_i, t)$ (to the ith neuron). The strengths associated with these feedforward input spikes are $F^{\text{AMPA}} = 0.05$, $F^{\text{NMDA}} = 0$. We design three different stimulus paradigms to mimic the real LGN drive:

1. Uniform input: The feedforward input rate to each neuron is uniform, with

$$\lambda^{\text{LGN}}(\mathbf{x}_i, t) = \bar{\lambda} := 0.05. \tag{13}$$

2. Drifting grating: The input rate

$$\lambda^{\text{LGN}}(\mathbf{x}_i, t) = \tilde{\lambda}^{\text{LGN}}(\mathbf{x}_i, t) := \bar{\lambda}\,(1 + \cos(\mathbf{k} \cdot \mathbf{x} + \omega t)) \tag{14}$$

   mimics a sinusoidal grating drifting across the cortex. The wavenumber $\mathbf{k}$ has magnitude $2\pi/12.8$, and the frequency $\omega = 2\pi/64$ ms. See Fig. 4(A).
3. Obscured grating: The input rate

$$\lambda^{\text{LGN}}(\mathbf{x}_i, t) = \Delta(\mathbf{x}_i)\tilde{\lambda}^{\text{LGN}}(\mathbf{x}_i, t), \tag{15}$$

where the mask $\Delta(\mathbf{x})$ obscures the right half and central region of the numerical cortex. See Fig. 4(B). This input provides a particularly strong test of Algorithm 5.1, since the obscured region is driven primarily by weak distant cortical firing events.

For each of these stimulus paradigms, our method (Algorithm 5.1) is capable of accurately resolving every spike, and computing conductances and voltages accurately in the textbook sense of numerical solutions for ODEs. However, if we insist on 1% accuracy in the voltage V_i of all 64×64 neurons over the time-course of 1024 ms, we are forced to take very small time-steps (typically $\Delta t_{\max} = 2^{-14} \approx 6 \times 10^{-5}$ ms or smaller) in order to accurately resolve the entire system. However, in many applications, it is not necessary to resolve every single trajectory of all neurons in the system. For example, in traditional approaches, many real experiments (Koch, 1999) only record the statistical properties of subsets of neurons (or the entire system), such as firing rate, ISI distributions, and correlations. An experiment involving firing rate statistics may only be concerned with the ISI distribution aggregated for many neurons in the system over a long time. To numerically reproduce these experiments we only need to obtain an accurate ISI distribution, etc. In modern experimental approaches, such as *in vivo* optical imaging based on voltage sensitive dyes (VSD), which has proven to be a powerful tool to investigate cortical spatiotemporal dynamics (Grinvald and Heildesheim, 2004), statistical information further includes spatiotemporal voltage patterns and their corresponding distributions in time and space. The VSD optical imaging has a spatial resolution typically covering ~50–200 neurons, and a sampling time of about 1– 10 ms. Therefore, when attempting to numerically model physiological phenomena, such as observed by VSD optical imaging, we may require only an accurate mean voltage for ensembles of ~50 neurons over time-intervals of, say, 8 ms. In both cases, the *statistical* properties of the network can be accurately resolved with much less computational effort than it would take to accurately resolve each and every neuronal trajectory. If we only require this type of statistical accuracy for the system above, we can take much larger time-steps (typically $\Delta t_{\max} = 2$ ms, with $\Delta t \approx 1$ ms for each step on average). Figure 5 shows the convergence results for statistical properties.

In Section 6.2, we discussed the importance of spike-spike corrections for the time-course of networks. Here, we discuss the importance of the spike-spike correction even for obtaining statistical information of networks. For each stimulus paradigm, we evolve the test system above for 1024 ms using Algorithm 5.1 with varying $\Delta t_{\max}$. For comparison, we also evolve the system using the naive algorithm—Algorithm 5.1 *without* spike-spike corrections. We record the following statistical quantities:

1. The Inter-Spike-Interval histogram ($\mathrm{ISI}^{\Delta t}$) for the entire system recorded over 1024 ms (cf. Fig. 5(D)). Every time a neuron in the system fires, we record the time lapsed since that same neuron last fired. The data is binned into 2 ms time bins.
2. The Accumulated Voltage histogram ($\mathrm{AV}^{\Delta t}$) for a patch of 16×16 neurons accumulated over the final 32 ms of a 1024 ms run (cf. Fig. 5(H)). We record the instantaneous voltage of every neuron in the region of interest throughout the desired time interval, and bin the data into 16 equispaced voltage bins.
3. The Spike-triggered Voltage histogram ($\mathrm{STV}^{\Delta t}$) for a patch of 16×16 neurons recorded over 1024 ms (cf. Fig. 5(L)). We record the instantaneous voltage of every neuron in the region of interest whenever any neuron in that region fires, and bin the data into 16 equispaced voltage bins.

As a measure of error, we use the relative max-norm—the largest difference in histogram bins divided by the largest bin, i.e.,

$$E^{ISI} = \left[\max_{i=1\ldots16} \left|\mathrm{ISI}^{exact} - \mathrm{ISI}^{\Delta t}\right|\right] \Big/ \left[\max_{i=1\ldots16} \left|\mathrm{ISI}^{exact}\right|\right], \tag{16}$$

$$E^{AV} = \left[\max_{i=1\ldots16} \left|\mathrm{AV}^{exact} - \mathrm{AV}^{\Delta t}\right|\right] \Big/ \left[\max_{i=1\ldots16} \left|\mathrm{AV}^{exact}\right|\right], \tag{17}$$

$$E^{STV} = \left[\max_{i=1\ldots16} \left|\mathrm{STV}^{exact} - \mathrm{STV}^{\Delta t}\right|\right] \Big/ \left[\max_{i=1\ldots16} \left|\mathrm{STV}^{exact}\right|\right], \tag{18}$$

where the exact histograms of ISI^{exact}, AV^{exact}, and STV^{exact}, such as shown in Figs. 5(D), (H) and (L), are estimated by using very small $\Delta t_{\max} = 2^{-16} \approx 1.5 \times 10^{-5}$ ms with Algorithm 5.1. We compute the errors in the histograms for many different Poisson spike trains, each of which is used for evolving the test system by using Algorithm 5.1 with and without the spike-spike corrections. The average errors over these trials are shown in Fig. 5. Clearly, it is demonstrated in Fig. 5 that, as the time-step Δt becomes smaller, the approximate histograms of $\mathrm{ISI}^{\Delta t}$, $\mathrm{AV}^{\Delta t}$, and $\mathrm{STV}^{\Delta t}$, converge to the exact histograms of ISI^{exact}, AV^{exact}, and STV^{exact}, respectively. As seen in Fig. 5, the histograms generated by the naive algorithm (i.e., without spike-spike corrections) do not capture the statistical behavior of the system well, with a significant difference from the histograms generated by Algorithm 5.1 (with spike-spike corrections) for the same time step Δt. As we mentioned above, for sufficiently small time-steps, of course, our methods can resolve the entire system accurately in the traditional sense.

6.4. Large systems

To further emphasize the efficiency of Algorithm 5.1, we construct the following test network which mimics the

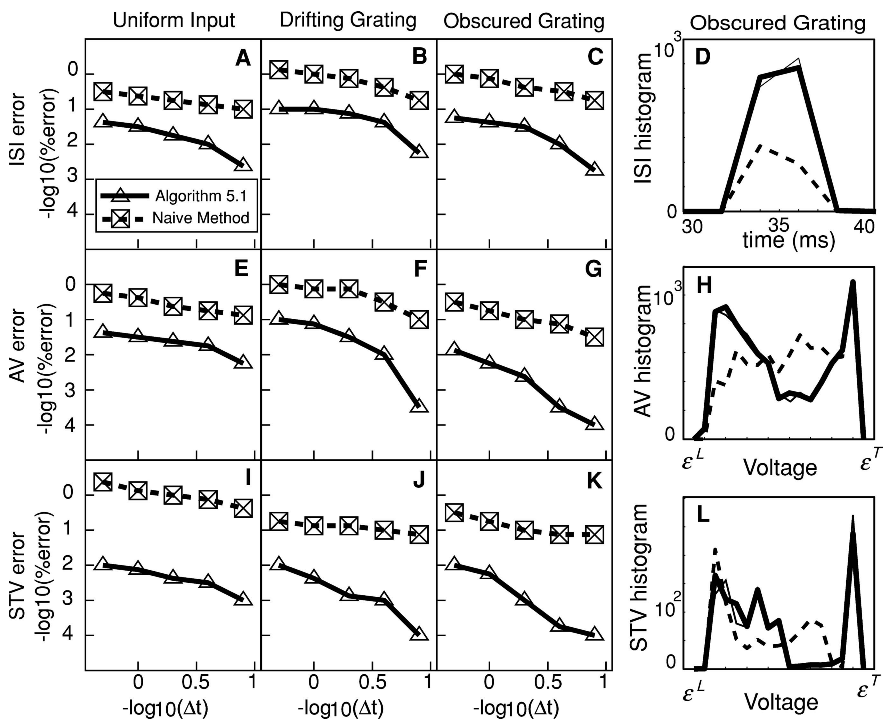

Fig. 5 *Importance of spike-spike corrections for statistical information.* Displayed are error graphs for Algorithm 5.1 applied to a test network under a variety of different stimuli. The test network consists of a square lattice of 64×64 neurons (75% excitatory and 25% inhibitory), of which each neuron has three types of conductance excitatory G^{AMPA} and G^{NMDA} (with $\epsilon^{\text{AMPA}} = \epsilon^{\text{NMDA}} = 14/3$, $\sigma^{\text{AMPA}} = 2$ ms and $\sigma^{\text{NMDA}} = 80$ ms) and inhibitory G^{GABA} (with $\epsilon^{\text{GABA}} = -2/3$ and $\sigma^{\text{GABA}} = 7$ ms). Each neuron is connected to its lattice neighbors with strength $S^Q_{i,j} = S^Q K^Q(|\vec{x}_i - \vec{x}_j|)$, where index Q runs over the types of conductance (with maximum strengths $S^{\text{AMPA}} = 3$, $S^{\text{NMDA}} = 0.05$, $S^{\text{GABA}} = 7$), and K^Q is a normalized Gaussian spatial kernel (with interaction scales $D^{\text{AMPA}} = 24$, $D^{\text{NMDA}} = 8$, $D^{\text{GABA}} = 16$). The three types of stimuli we consider for this test network are (i) homogeneous input (Eq. (13)), (ii) drifting grating input (Eq. (14))—see Fig. 4(A) and (iii) obscured grating input (Eq. (15))—See Fig. 4(B). Panels ABC,EFG,IJK all share the same axis scales. The thick black line (with data points marked by '△') corresponds to Algorithm 5.1 (by default, with spike-spike corrections). The dashed line (with data points marked by '(⊠)' corresponds to the naive algorithm (i.e., Algorithm 5.1 *without* spike-spike corrections). In panels D,H,L, the thin black line indicates the exact histogram (approximated by using Algorithm 5.1 with a sufficiently small time-step); the thick black line corresponds to Algorithm 5.1 and the dashed line corresponds to the naive algorithm. (A,B,C) Errors for the interspike-interval (ISI) histogram (Eq. (16)), as computed when the network is subject to homogeneous background input (A), drifting grating input (B) and obscured grating input (C). (D) The ISI histograms computed with $\Delta t = 1$ ms and compared with the exact solution (thin black line). (E,F,G) Errors for the instantaneous voltage (AV) histogram (Eq. (17)), as computed when the network is subject to homogeneous background input (E), drifting grating input (F) and obscured grating input (G). (H) The AV histograms computed with $\Delta t = 1$ ms. (I,J,K) Errors for the spike-triggered voltage (STV) histogram (Eq. (18)), as computed when the network is subject to homogeneous background input (I), drifting grating input (J) and obscured grating input (K). (L) The STV histograms computed with $\Delta t = 1$ ms. Note that Algorithm 5.1 is much better than the naive algorithm at capturing the statistical features of the network, especially for large time-steps

large-scale network architecture found within V1. We construct a square lattice of N neurons, similar to the network described in Section 6.3, except that the interaction strengths are given by $S^Q_{i,j} = S^Q_{\mathcal{T}_i\mathcal{T}_j} K^Q(\|\mathbf{x}_i - \mathbf{x}_j\|)$, where the 6 constants $S^Q_{\mathcal{T}_i\mathcal{T}_j}$ determine the maximum connection strengths. We choose $S^{\text{AMPA}}_{EE} = 5$, $S^{\text{AMPA}}_{IE} = 2$, $S^{\text{NMDA}}_{EE} = S^{\text{NMDA}}_{IE} = 10^{-5}$, $S^{\text{GABA}}_{EI} = 15$, $S^{\text{GABA}}_{II} = 5$. (We remark that even though Algorithm 5.1 has no problems resolving the system over a wide range of coupling parameters, the modified Runge-Kutta methods, which we use as a benchmark, estimate too many NMDA driven spiking event for larger values of S^{NMDA}_{EE} and S^{NMDA}_{IE}, and thus, given a fixed error tolerance, they require so small time-steps that the computation is intractable). For the purpose of illustration, we use the instantaneous value of G^{NMDA} as a measure of aggregate excitatory activity within the system. Each neuron in the system is driven by excitatory input spike trains, which are each independent realizations of a Poisson process with rate $\lambda(\vec{x}) = \bar{\lambda} := 0.15$. For a fixed system size N, we can evolve the system for 128 ms using both Algorithm 5.1, and the modified Runge-Kutta methods (Hansel et al., 1998; Shelley and Tao, 2001), which use interpolation to determine individual spike times, but do not incorporate spike-spike corrections, and do not use a spatial divide-and-conquer scheme to take advantage of the spatial architecture of the network. We vary the maximum timestep Δt and measure the instantaneous G^{NMDA} profiles of a 64×64 neuron subregion at the 128 ms time point. We denote by NP

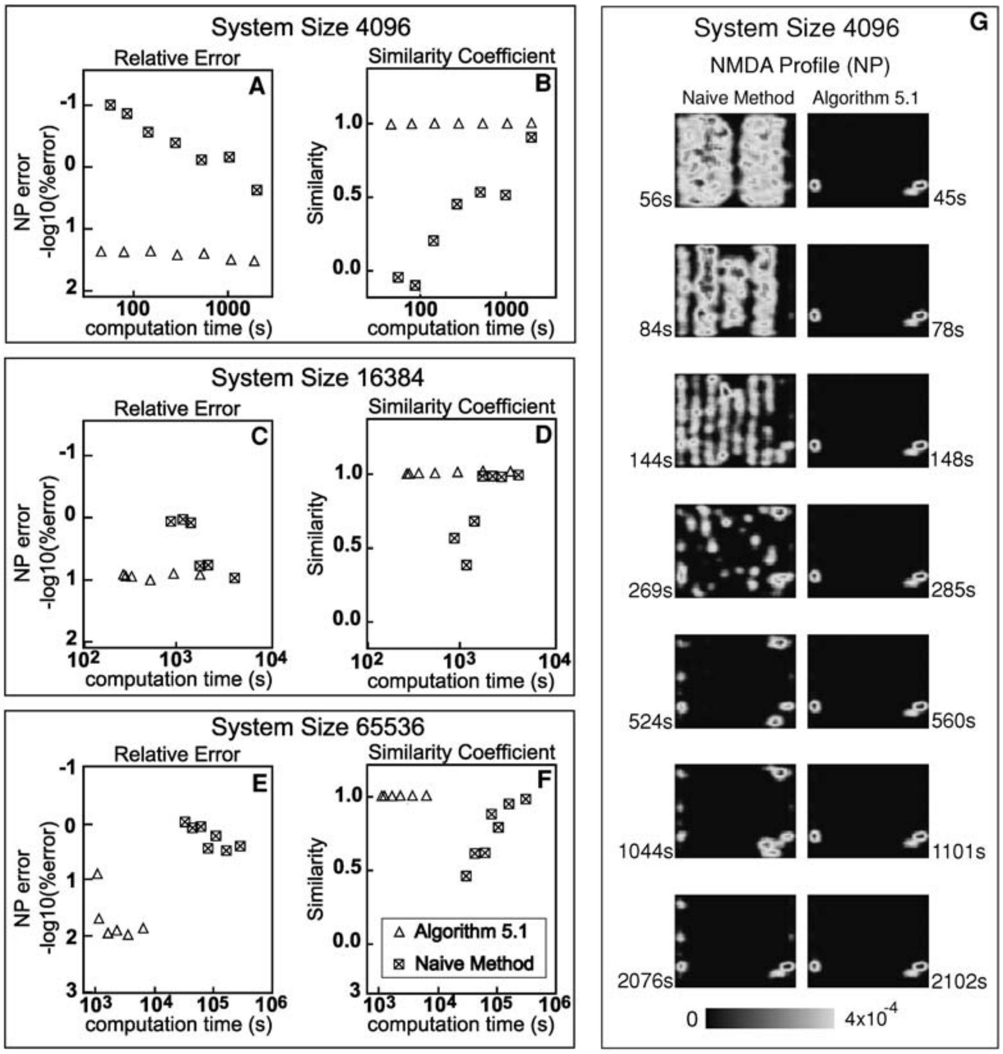

Fig. 6 *Efficiency of Algorithm* 5.1. Displayed are graphs of relative error and similarity coefficients for Algorithm 5.1 as well as the modified Runge-Kutta methods (Hansel et al., 1998; Shelley and Tao, 2001) when applied to a test network under homogeneous input (with rate $\bar{\lambda} = 0.15$ spikes/s). The test network consists of a square lattice of N neurons (75% excitatory and 25% inhibitory), of which each neuron has three types of conductance—excitatory G^{AMPA} and G^{NMDA} (with $\epsilon^{\text{AMPA}} = \epsilon^{\text{NMDA}} = 14/3$, $\sigma^{\text{AMPA}} = 2$ ms and $\sigma^{\text{NMDA}} = 80$ ms) and inhibitory G^{GABA} (with $\epsilon^{\text{GABA}} = -2/3$ and $\sigma^{\text{GABA}} = 7$ ms). Each neuron is connected to its lattice neighbors with interaction strengths given by $S^Q_{i,j} = S^Q_{\mathcal{T}_i\mathcal{T}_j} K^Q(\|\mathbf{x}_i - \mathbf{x}_j\|)$, where the 6 maximum connection strengths $S^Q_{\mathcal{T}_i\mathcal{T}_j}$ are $S^{\text{AMPA}}_{EE} = 5$, $S^{\text{AMPA}}_{IE} = 2$, $S^{\text{NMDA}}_{EE} = S^{\text{NMDA}}_{IE} = 10^{-5}$, $S^{\text{GABA}}_{EI} = 15$, $S^{\text{GABA}}_{II} = 5$, and K^Q is a normalized Gaussian spatial kernel (with interaction scales $D^{\text{AMPA}} = 24$, $D^{\text{NMDA}} = 8$, $D^{\text{GABA}} = 16$). We evolve the system and measure the instantaneous value of G^{NMDA} of a 64×64 neuron subregion at the 128 ms time point. We denote by NP the convolution of this G^{NMDA} profile with a two-dimensional Gaussian kernel with a radius of 2 neuron lattice spacings. We vary the time-step and measure the accuracy (Eq. (19)) and similarity of NP as a function of the overall computation time, where similarity is the spatial correlation coefficient between $\text{NP}^{\Delta t}(\vec{x})$ and $\text{NP}^{exact}(\vec{x})$ obtained from the convergent solution computed using a sufficiently small time-step) Panels AB,CD,EF display error and similarity graphs for system size 4096, 16384, 65536 respectively. Data points marked by '△' correspond to Algorithm 5.1. Data points marked by '⊠' correspond to the modified Runge-Kutta methods. (A,B) Relative errors (A) and similarity index (B) obtained by evolving a system of size $N_1 = 4096$. (C,D) Relative errors (C) and similarity index (D) obtained by evolving a system of size $N_2 = 16384$. (E,F) Relative errors (E) and similarity index (F) obtained by evolving a system of size $N_3 = 65536$. (G) Snapshots of NP for the system of size $N_1 = 4096$ shown in panels (A,B)—the left column illustrates the NP pattern obtained with the modified Runge-Kutta methods (with computation times listed at the left), and the right column illustrates the NP pattern obtained with Algorithm 5.1 (with computation times listed at the right). Note that the time associated with each panel is *not* the evolution time, but rather the computation time required to evolve the system to the $t = 128$ ms time point. All panels use the color bar at the bottom. Note that Algorithm 5.1 can resolve the statistical features of the network within a fraction of the time that the modified Runge-Kutta method requires

$^{\Delta t}(\vec{x})$ the linear convolution (spatial smearing) of the G^{NMDA} profile with a two dimensional Gaussian kernel with a radius of 2 neuron lattice spacings. We compute the relative error

$$E^{NP}(\Delta t) = \left\| \text{NP}^{exact}(\vec{x}) - \text{NP}^{\Delta t}(\vec{x}) \right\|_2 / \| \text{NP}^{exact}(\vec{x}) \|_2, \tag{19}$$

where the exact solution $\text{NP}^{exact}(\vec{x})$ is approximated by using a very small timestep with Algorithm 5.1. With this data, we can measure the computation time necessary to achieve a given statistical accuracy. The results for systems of size $N_1 \approx 4.1 \times 10^3$, $N_2 = 4N_1 \approx 1.6 \times 10^4$ and $N_3 = 4N_2 \approx 6.6 \times 10^4$ are shown in Fig. 6 for $E^{NP}(\Delta t)$ as well as for the similarity, which is the spatial correlation coefficient between $\text{NP}^{\Delta t}(\vec{x})$ and $\text{NP}^{exact}(\vec{x})$ at $t = 128$ ms It can be seen that, for these systems, Algorithm 5.1 can calculate an accurate G^{NMDA} profile quickly, whereas the modified Rung-Kutta methods require at least $50 - 100$ times as much computation time to obtain the same degree of accuracy. Figure 6(G) shows an example of the G^{NMDA} profile at $t = 128$ ms obtained by applying both methods to the smallest system (of size $N_1 \approx 4.1 \times 10^3$). Note that the times marked are not the evolution time, but rather the total computation time (in seconds) that each algorithm takes to evolve the system to the $t = 128$ ms time point (we run the code on a Linux platform using a Intel Pentium IV 2.3 MHz processor). It can be seen that Algorithm 5.1 obtains a convergent numerical solution for *NP* within 1 minute, whereas the modified Runge-Kutta methods require nearly an hour to resolve the computation.

As the system size increases, the disparity in performance between Algorithm 5.1 and the modified Runge-Kutta methods becomes even more apparent. For example, we can try to evolve a similar system of size $N_4 = 4N_3 \approx 2.6 \times 10^5$, which is the order of magnitude needed to simulate many V1 cortical phenomena (Cai et al., 2005; Rangan et al., 2005). Algorithm 5.1 can resolve this system within 1 hour, whereas the modified methods require nearly 6 months (as a conservative estimate) to calculate a reasonable *NP* profile. One reason for this disparity is that the spatial divide-and-conquer scheme within Algorithm 5.1 focuses computational effort towards the resolution of the $\approx ND^2$ relevant (strong) synapses within the network. The modified Runge-Kutta methods, on the other hand, spend an equal amount of time resolving all N^2 ordered pairs of neuronal interactions.

7. Conclusions

Algorithm 5.1 is a fast and accurate method for evolving very large integrate-and-fire neuronal networks with local coupling. There are three reasons underlying the efficiency of the method:

1. The neurophysiologically inspired integrating factor in Algorithm 8.1 allows us to evolve individual neuronal trajectories without time-step restrictions due to stability (i.e., even if the conductances are high and the I&F neuronal equations are stiff).
2. The spike-spike corrections (step 12 of Algorithm 5.1) allow us to accurately estimate spiking sequences for groups of strongly coupled neurons without the usual time-step restriction imposed by cortico-cortical interactions in modified Runge-Kutta methods.
3. Given a system of N neurons, the neuron-neuron interactions are grouped into local clusters. This clustering procedure allows Algorithm 5.1 to consider $\mathcal{O}(N)$ spiking events with only $\mathcal{O}(N)$ operations. Therefore, the entire system can be evolved in such a way that every neuron fires approximately once in $\mathcal{O}(N)$ operations, which is asymptotically optimal.

Finally, we emphasize that this method is also practicable. We have used Algorithm 5.1 to study spontaneous and evoked activity in V1 through the investigation of the associated spatiotemporal dynamics using large-scale, spatially extended neuronal networks of model cortex (Cai et al., 2005; Rangan et al., 2005).

8. Appendix A

We provide psuedocode for our single test neuron method. Given an initial time t_0 and time-step Δt, initial values $V_i(t_0), G_i^Q(t_0)$, and spike times $T_{i,k}^F, T_{j\neq i,k}$ from the rest of the network, this method calculates a numerical solution $V_i(t_0 + \Delta t), G_{Q,i}(t_0 + \Delta t)$ as well as the intervening spike-times $T_{i,k}$ for the ith neuron (i.e., the test neuron).

Algorithm 8.1. Test Neuron Method

```
1. Input: an initial time $t_0$, a time-step $\Delta t$, a set of spike-times $T_{j\neq i,k}$, $T_{i,k}^F$ and associated strengths $S_{i,j}^Q$, $F_i^Q$, and a quadrature order $P$.
2. Consider the time interval $[t_0, t_0 + \Delta t]$.
3. Define $K$ to be the total number of impinging presynaptic inter-cortical and feedforward spikes within the interval $[t_0, t_0 + \Delta t]$. Sort these spikes into an increasing list of $K$ spike-times $T_k^{sorted}$ with corresponding spike strengths $S_k^{sorted,Q}$. In addition, we extend this notation such that $T_0^{sorted} := t_0$, $T_{K+1}^{sorted} := t_0 + \Delta t$ and $S_0^{sorted,Q} = S_{K+1}^{sorted,Q} := 0$.
4. for $k = 1 \ldots K+1$
      Choose a set of $P$ quadrature nodes $\tau_1 < \tau_2 < \cdots < \tau_n < \cdots < \tau_P$ with corresponding weights
```

$\{w_1, \ldots, w_P\}$ in the interval $[T_{k-1}^{sorted}, T_k^{sorted}]$ (Fornberg, 1998).

Extend this notation such that $\tau_0 := T_{k-1}^{sorted}$ and $\tau_{P+1} := T_k^{sorted}$.

Evaluate the quantities $G_i^Q(\tau_n)$ and $G_i^S(\tau_n)$ exactly using Eqs. (5) and (2).

Evaluate the quantity $\exp(\int_{\tau_0}^{\tau_n} G_i^S(r)dr)$ exactly using Eqs. (5) and (2).

Evaluate $V_i^S(\tau_n)$ and $\frac{d}{dt}V_i^S(\tau_n)$ exactly using Eqs. (3) and (8).

Use Gaussian quadrature to approximate $V_i(\tau_{P+1})$ using Eq. (6).

Update the conductances $G_i^Q(\tau_{P+1}) = G_i^Q(T_k^{sorted})$ by adding the appropriate strengths $S_k{}^{sorted,Q}$. (see Eq. (5)).

5. *If the calculated values for $V_i(T_k^{sorted})$ are each less than ϵ^T, then we accept the solution. We update $t_0 \leftarrow t_0 + \Delta t$ and return to step 2 and continue.*
6. *Otherwise, let $V_i(T_k^{sorted})$ be the first calculated voltage greater than ϵ^T. We know that the ith neuron spiked somewhere during the interval $[T_{k-1}^{sorted}, T_k^{sorted}]$.*
7. *In this case we construct a Pth order polynomial interpolant of $V_i(t) - \epsilon^T$ in the interval $[T_{k-1}^{sorted}, T_k^{sorted}]$ and search for a root. For example, when $P = 3$, we can use the numerical values of $V_i(T_{k-1}^{sorted})$, $V_i(T_k^{sorted})$, $\frac{d}{dt}V_i(T_{k-1}^{sorted})$, $\frac{d}{dt}V_i(T_k^{sorted})$ to form a cubic polynomial. We record the root r as an approximation to the spike time and set $V_i(t) := \epsilon^R$ for the next τ_{ref} ms. We update $t_0 \leftarrow min(r + \tau_{ref}, t + \Delta t)$ and return to step 2 and continue.*

9. Appendix B

We provide a detailed description of the algorithm used for spike-spike corrections. Given an initial time t_0 and time-step Δt, a K-element list of spiking neurons $A = \{\mathcal{N}_1, \ldots, \mathcal{N}_K\}$ with initial conditions $V_i(t_0)$, $G_i^Q(t_0)$, as well as feedforward input spiketimes $T_{i,k}^F$, this method calculates accurate trajectories and spiketimes for the neuron list A over the time interval $[t_0, t_0 + \Delta t]$.

Algorithm 9.1 *Spike-Spike Corrections*

1. *Input: an initial time t_0, a timestep Δt, a set of K neurons $A^{[0]} := A = \{\mathcal{N}_1, \ldots, \mathcal{N}_K\}$, feeforward input spiketimes $T_{i,k}^F$ to those neurons, associated coupling strengths $S_{i,j}^Q$, F_i^Q, and a quadrature order P.*
2. *Considering the time interval $I^{[0]} = [t_0, t_0 + \Delta t]$, as well as the spikes $T_{i,k}^F$, use Algorithm 8.1 with quadrature order P to approximate the neuronal trajectories of the neurons within $A^{[0]}$. Use P^{th} order polynomial interpolation to approximate the spiketime $T_i^{[1]}$ of each neuron. Sort the list $\{T_1^{[1]}, \ldots, T_K^{[1]}\}$ of neuronal spiketimes into the increasing list $\mathcal{T}^{[1]} = \{T_{i_1^1}^{[1]}, \ldots, T_{i_K^1}^{[1]}\}$ such that $T_{i_j^1}^{[1]} < T_{i_{j+1}^1}^{[1]}$.*
3. *Note that, because each cortical spike only affects future neuronal trajectories, none of the cortical spikes within the interval $I^{[0]}$ can affect the spiking event $T_{i_1^1}^{[1]}$ of neuron $\mathcal{N}_{i_1^1}$. Therefore we can assume that the neuronal trajectories computed up to this spike, as well as the spiketime $T_{i_1^1}^{[1]}$ itself, are accurate. Construct the list $A^{[1]}$ consisting of all the neurons in $A^{[0]}$ except for $\mathcal{N}_{i_1^1}$. Construct the interval $I^{[1]} = [T_{i_1^1}^{[1]}, t_0 + \Delta t]$.*
4. *Using the sorted list $\mathcal{T}^{[1]}$ of neuronal spiketimes as well as the spikes $T_{i,k}^F$, use Algorithm 8.1 with quadrature order P to re-approximate the neuronal trajectories of the neurons within $A^{[1]}$ on the interval $I^{[1]}$. Again use P^{th} order polynomial interpolation to re-approximate the spiketimes $T_i^{[2]}$ of each neuron. Sort the list $\{T_1^{[2]}, \ldots, T_{i_1^1-1}^{[2]}, T_{i_1^1+1}^{[2]}, \ldots, T_K^{[2]}\}$ of neuronal spiketimes into the increasing list $\mathcal{T}^{[2]} = \{T_{i_2^2}^{[2]}, \ldots, T_{i_K^2}^{[2]}\}$ such that $T_{i_j^2}^{[2]} < T_{i_{j+1}^2}^{[2]}$. Now we can assume that both the spiketime $T_{i_1^1}^{[1]}$ as well as the spiketime $T_{i_2^2}^{[2]}$ are accurate. Construct the set $A^{[2]} = A^{[1]} - \{\mathcal{N}_{i_2^2}\}$, and the interval $I^{[2]} = [T_{i_2^2}^{[2]}, t_0 + \Delta t]$.*
5. *Continue in this manner, using the sorted list $\mathcal{T}^{[2]}$ as well as the spikes $T_{i,k}^F$, $\{T_{i_1^1}^{[1]}\}$ and Algorithm 8.1 with quadrature order P to re-approximate the neuronal trajectories on the interval $I^{[2]}$. Use polynomial interpolation to find spiketimes, and construct the sorted list $\mathcal{T}^{[3]} = \{T_{i_3^3}^{[3]}, \ldots, T_{i_K^3}^{[3]}\}$. Now we can assume that the spiketimes $\{T_{i_1^1}^{[1]}, T_{i_2^2}^{[2]}, T_{i_3^3}^{[3]}\}$ are all accurate. Construct the set $A^{[3]} = A^{[2]} - \{\mathcal{N}_{i_3^3}\}$, and the interval $I^{[3]} = [T_{i_3^3}^{[3]}, t_0 + \Delta t]$.*
6. *Continue in this manner, at each step recording the spiketime $T_{i_j^j}^{[j]}$ of the first neuron $\mathcal{N}_{i_j^j}$ which fires within the interval $I^{[j-1]} = [T_{i_{j-1}^{j-1}}^{[j-1]}, t_0 + \Delta t]$ under the influence of the feedforward input spikes $T_{i,k}^F$ as well as the accurate spiketimes $\{T_{i_1^1}^{[1]}, \ldots, T_{i_{j-1}^{j-1}}^{[j-1]}\}$.*
7. *Finally, after K steps (and $O(K^2)$ operations), we have K accurate spike-times $\{T_{i_1^1}^{[1]}, \ldots, T_{i_K^K}^{[K]}\}$. Use these spiketimes as well as the feedforward spiketimes $T_{i,k}^F$ and Algorithm 8.1 with quadrature order P to perform a final correction of the neuronal trajectories of all neurons in $A^{[0]}$ on the interval $I^{[0]} = [t_0, t_0 + \Delta t]$.*

References

Anderson J, Lampl I, Gillespie D, Ferster D (2000) The contribution of noise to contrast invariance of orientation tuning in cat visual cortex. Science, 290: 1968–1972

Angelucci A, Levitt JB, Walton EJ, Hupe JM, Bullier J, Lund JS (2002) Circuits for local and global signal integration in primary visual cortex. J. Neurosci., 22: 8633–8646

Borg-Graham L, Monier C, Fregnac Y (1996) Voltage-clamp measurement of visually-evoked conductances with whole-cell patch recordings in primary visual cortex. J Physiol Paris, 90(3-4): 185–188

Borg-Graham LJ, Monier C, Fregnac Y (1998) Visual input evokes transient and strong shunting inhibition in visual cortical neurons. Nature, 393(6683): 369–373

Bosking WH, Zhang Y, Schofield B, Fitzpatrick D (1997) Orientation selectivity and the arrangement of horizontal connections in tree shrew striate cortex. J. Neurosci., 17: 2112–2127

Brette R (to appear) Exact simulation of integrate-and-fire models with synaptic conductances. Neural Comput

Cai D, Rangan AV, McLaughlin DW (2005) Architectural and synaptic mechanisms underlying coherent spontaneous activity in v1. Proc. Nat'l Acad. Sci (USA), 102: 5868–5873

Cai D, Tao L, Shelley M, Mclaughlin DW (2004) An effective representation of fluctuation-driven neuronal networks with application to simple & complex cells in visual cortex. Pro. Nat. Acad. Sci. (USA), 101: 7757–7762

Callaway E (1998) Local circuits in primary visual cortex of the macaque monkey. Ann. Rev. Neurosci., 21: 47–74

Callaway E, Wiser A (1996) Contributions of individual layer 2 to 5 spiny neurons to local circuits in macaque primary visual cortex. Visual Neuroscience, 13: 907–922

Compte A, Sanchez-Vives MV, McCormick DA, Wang X-J (2003) Cellular and network mechanisms of slow oscillatory activity ($<$ 1hz) and wave propagations in a cortical network model. J Neurophysiol., 89: 2707–2725

Destexhe A, Rudolph M, Pare D (2003) The high-conductance state of neocortical nurons in vivo. Nat. Rev., Neurosci, 4: 730–751

Fitzpatrick D, Lund J, Blasdel G (1985) Intrinsic connections of macaque striate cortex Afferent and efferent connections of lamina 4C. Journal of Neuroscience, 5: 3329–3349

Fornberg B (1998) A Practical Guide to Pseudospectral Methods. Cambridge University Press, New York.

Fourcaud-Trocme N, Hansel D, van Vreeswijk C, Brunel N (2003) How spike generation mechanisms determine the neuronal response to fluctuating inputs. J. Neurosci., 23

Frenkel D, Smit B (1996) Understanding Molecular Simulation. Academic Press, New York.

Gear CW (1971) Numerical Initial Value Problems in Ordinary Differential Equations. Prentice Hall, Englewood Cliffs, NJ.

Geisler C, Brunel N, Wang X-J (2005) Contributions of intrinsic membrane dynamics to fast network oscillations with irregular neuronal discharges. J. Neurophysiol., page in press.

Gilbert CD, Wiesel TN (1983) Clustered intrinsic connections in cat visual cortex. J. Neurosci., 3: 1116–1133

Grinvald A, Heildesheim R (2004) VSDI: a new era in functional imaging of cortical dynamics. Nat. Rev. Neurosci., 5: 874–885

Hansel D, Mato G, Meunier C, Neltner L (1998) On numerical simulations of integrate-and-fire neural networks. Neural Comput., 10

Jancke D, Chavance F, Naaman S, Grinvald A (2004) Imaging cortical correlates of illusion in early visual cortex. Nature, 428: 423–426

Kenet T, Bibitchkov D, Tsodyks M, Grinvald A, Arieli A (2003) spontaneously emerging cortical representations of visual attributes. Nature, 425: 954–956

Koch C (1999) Biophysics of Computation. Oxford University Press, Oxford.

Lytton W, Hines ML (2005) Independent variable time-step integration of individual neurons for network simulations. Neural Comput, 17: 903–921

Lund JS (1987) Local circuit neurons of macaque monkey striate cortex: Neurons of laminae 4C and 5{A}. Journal of Comparative Neurology, 257: 60–92

Makino T (2003) A discrete-even neural network simulator for general neuron models. Neural Comput. and Appl., 11: 210–223

Marino J, Schummers J, Lyon DC, Schwabe L, Beck O, Wiesing P, Obermayer K, Sur M (2005) Invariant computations in local cortical networks with balanced excitation and inhibition. Nat. Neurosci., 8

McLaughlin D, Shapley R, Shelley M, Wielaard J (2000) A neuronal network model of macaque primary visual cortex ({V1}): {O}rientation selectivity and dynamics in the input layer 4Cα. Proc. Natl. Acad. Sci. USA, 97: 8087–8092

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

Pare D, Shink E, Gaudreau H, Destexhe A, Lang EJ (1998) Impact of spontaneous synaptic activity on the resting properties of cat neocortical pyramidal neurons in vivo. J Neurophysiol, 79: 1450–1460

Rangan AV, Cai D, McLaughlin DW (2005) Modeling the spatiotemporal cortical activity associated with the line-motion illusion in primary visual cortex. Proc. Nat'l Acad. Sci (USA), 102(52): 18793–18800

Rauch A, LaCamera G, Luscher HR, Senn W, Fusi S (2003) Neocortical pyramidal cells respond as integrate-and-fire neurons to in vivo-like input currents. J. Neurophysiol., 90

Rochel O, Martinez D (2003) An event-driven framework for the simulation of networks of spiking neurons. In Proc. 11th European Symposium on Artificial Neural Networks d-side publications, 295–300

Rudolph M, Destexhe A. (2003a) A fast-conducting, stochastic integrative mode for neocortical neurons in vivo. J Neurosci, 23(6): 2466–2476

Rudolph M, Destexhe A (2003b) Characterization of subthreshold voltage fluctuations in neuronal membranes. Neural Comput, 15(11): 2577–2618

Rudolph M, Destexhe A (2003c) The discharge variability of neocortical neurons during high-conductance states. Neuroscience, 119(3): 855–873

Rudolph M, Destexhe A (2003d) Tuning neocortical pyramidal neurons between integrators and coincidence detectors. J Comput Neurosci, 14(3): 239–251

Shadlen MN, Newsome WT (1998) The variable discharge of cortical neurons: Implications for connectivity, computation and information coding. J Neurosci, 18: 3870–3896

Shelley MJ, Tao L (2001) Efficient and accurate time-stepping schemes for integrate-and-fire neuronal networks. J. Comput. Neurosci., 11

Sincich L, Blasdel G (2001) Oriented axon projections in primary visual cortex of the monkey. J. Neurosci., 21: 4416–4426

Somers D, Nelson S, Sur M (1995) An emergent model of orientation selectivity in cat visual cortical simple cells. J. of Neurosci., 15: 5448–5465

Stern EA, Kincaid AE, Wilson CJ (1997) Spontaneous subthreshold membrane potential fluctuations and action potential variability of rat corticostriatal and striatal neurons in vivo. J. Neurophysiol., 77: 1697–1715

Tao L, Shelley M, McLaughlin D, Shapley R (2003) An egalitarian network model for the emergence of simple and complex cells in visual cortex. PNAS.

Troyer T, Krukowski A, Priebe N, Miller K (1998) Contrast invariant orientation tuning in cat visual cortex with feedforward tuning and correlation based intracortical connectivity. J. Neurosci., 18: 5908–5927

Tsodyks M,Kenet T, Grinvald A, Arieli A (1999) Linking spontaneous activity of single cortical neurons and the underlying functional architecture. Science, 286: 1943–1946

Wang X-J (1999) Synaptic basis of cortical persistent activity: The importance of NMDA receptors to working memory. J Neurosci., 19: 9587–9603