On the Performance of Voltage Stepping for the Simulation of Adaptive, Nonlinear Integrate-and-Fire Neuronal Networks

Mohamed Ghaith Kaabi, Arnaud Tonnelier, Dominique Martinez

▶ To cite this version:

Mohamed Ghaith Kaabi, Arnaud Tonnelier, Dominique Martinez. On the Performance of Voltage Stepping for the Simulation of Adaptive, Nonlinear Integrate-and-Fire Neuronal Networks. Neural Computation, 2011, 23 (5), pp.1187-1204. 10.1162/NECO_a_00112 . inria-00611646

HAL Id: inria-00611646
https://inria.hal.science/inria-00611646

Submitted on 29 May 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

On the Performance of Voltage Stepping for the Simulation of Adaptive, Nonlinear Integrate-and-Fire Neuronal Networks

Mohamed Ghaith Kaabi
mohamedghaith.kaabi@loria.fr
Unité Mixte de Recherche 7503, LORIA, CNRS, 54506 Vandoeuvre-lès-Nancy, France

Arnaud Tonnelier
arnaud.tonnelier@inria.fr
INRIA, 38334 Saint Ismier, France

Dominique Martinez
dominique.martinez@loria.fr
Unité Mixte de Recherche 7503, LORIA, CNRS, 54506 Vandoeuvre-lès-Nancy, France, and Unité Mixte de Recherche 1272, Physiologie de l'Insecte Signalisation et Communication, Institut National de la Recherche Agronomique, 78026 Versailles, France

In traditional event-driven strategies, spike timings are analytically given or calculated with arbitrary precision (up to machine precision). Exact computation is possible only for simplified neuron models, mainly the leaky integrate-and-fire model. In a recent paper, Zheng, Tonnelier, and Martinez (2009) introduced an approximate event-driven strategy, named voltage stepping, that allows the generic simulation of nonlinear spiking neurons. Promising results were achieved in the simulation of single quadratic integrate-and-fire neurons. Here, we assess the performance of voltage stepping in network simulations by considering more complex neurons (quadratic integrate-and-fire neurons with adaptation) coupled with multiple synapses. To handle the discrete nature of synaptic interactions, we recast voltage stepping in a general framework, the discrete event system specification. The efficiency of the method is assessed through simulations and comparisons with a modified time-stepping scheme of the Runge-Kutta type. We demonstrated numerically that the original order of voltage stepping is preserved when simulating connected spiking neurons, independent of the network activity and connectivity.

1 Introduction

Computer simulations have become essential for understanding the complex dynamics of the brain. Recently, large-scale simulations of realistic cortical networks have been undertaken (Izhikevich & Edelman, 2008; Migliore, Cannia, Lytton, Markram, & Hines, 2006). However, biophysically detailed neuron models have a high computational cost, and the difficulty to tune their numerous parameters makes them inefficient for spike timing prediction (Gerstner & Naud, 2009). Simpler spiking neuron models of the integrate-and-fire type are more suitable for efficient large-scale neural network simulations. The leaky integrate-and-fire (LIF) is computationally attractive and amenable to mathematical analysis. Yet it is too simple, and two key elements have to be incorporated into the model to reproduce the large range of spiking dynamics exhibited by cortical neurons: nonlinear spike generating currents that allow an accurate description of the membrane potential near the threshold (Fourcaud-Trocme, Hansel, van Vreeswijk, & Brunel, 2003) and adaptive currents that keep track of the past activity of the neuron, yielding a considerable increase in neurocomputational properties (Izhikevich, 2003; Brette & Gerstner, 2005). Recently, a class of adaptive nonlinear spiking neuron models that captures both characteristics has been proposed (Izhikevich, 2003; Brette & Gerstner, 2005; Touboul, 2008). If v is the membrane potential and w an adaptation variable, then the time evolution of the neuron is given by

$$C\frac{dv}{dt} = f(v) - w + I + I_{syn}(t), \tag{1.1}$$

$$\frac{dw}{dt} = a\left(b(v - v_{rest}) - w\right), \tag{1.2}$$

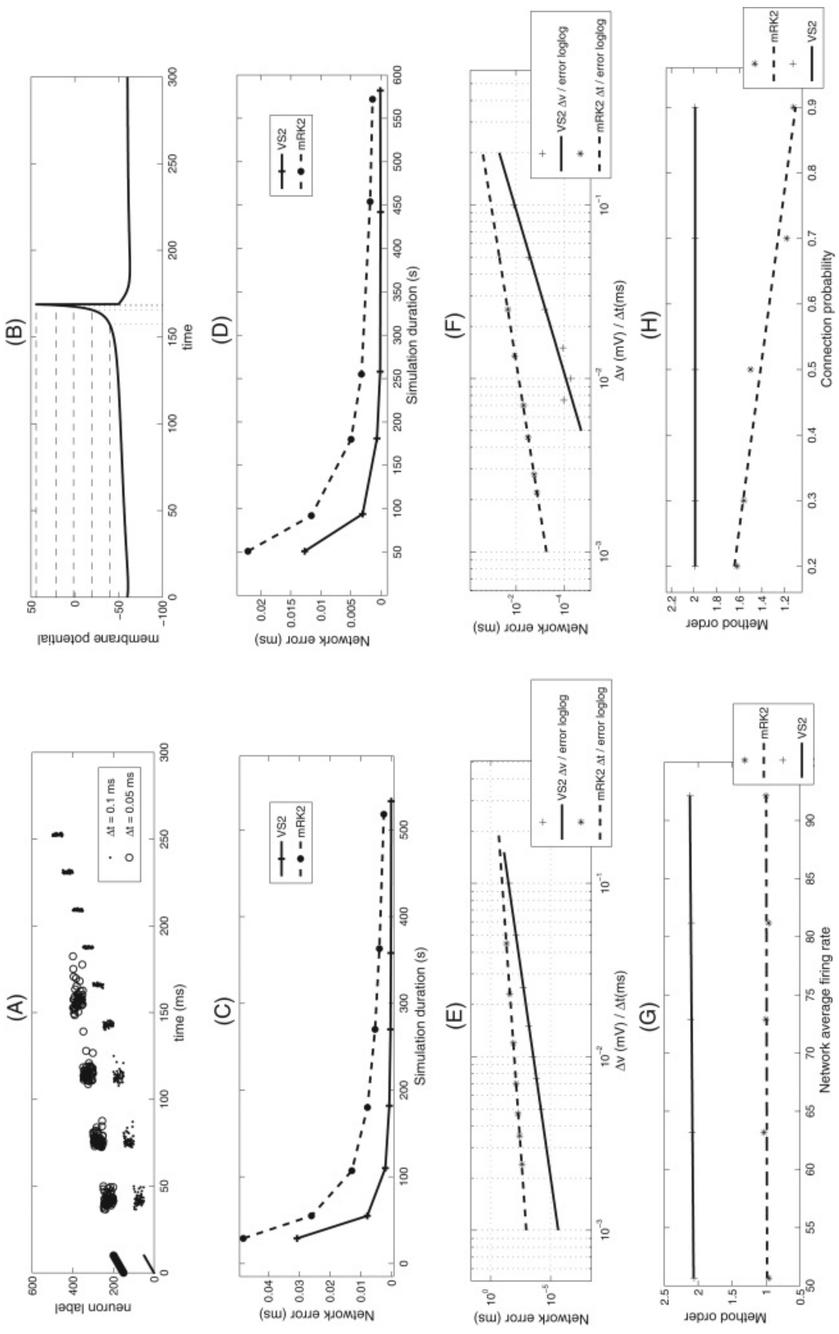
with the reset condition $v \leftarrow v_{reset}$ and the update rule $w \leftarrow w + d$ at firing times obtained when v reaches a threshold value v_{th}. The parameter $d > 0$ represents the spike-triggered adaptation. In equation 1.1, C is the membrane capacitance, f is a nonlinear current voltage function, I is an external constant input current, and I_{syn} is the synaptic current. In equation 1.2, parameter a describes the timescale of the adaptation current, b describes the sensitivity of w to the subthreshold membrane potential fluctuations, and v_{rest} is the resting potential. The LIF model is obtained using a relaxation function $f(v) = -v$. More realistic models include nonlinear spike-generating currents like the adaptive quadratic model (Ermentrout, 1996; Izhikevich, 2003) obtained for $f(v) = v^2$, the adaptive exponential model (Fourcaud-Trocme et al., 2003; Brette & Gerstner, 2005) for which $f(v) = -v + e^v$, or the quartic model (Touboul, 2008) obtained for $f(v) = v^4 + 2\alpha v$. The class of models described by equations 1.1 and 1.2 has attracted a lot of attention mainly because of its relative simplicity

(Izhikevich, 2004), its ability to reproduce a large variety of firing patterns (Izhikevich, 2003; Touboul, 2008), and its predictive abilities (Brette & Gerstner, 2005). However the relative computational efficiency of the model is affected by highly nonlinear dynamics. The numerical errors associated with the discontinuities of the membrane potential v at firing times can have severe consequences on the numerical integration, and spurious dynamics can be generated (Hansel, Mato, Meunier, & Neltner, 1998). At the the same time, the addition of an adaptation current can lead to a high sensitivity to the cutoff value v_{th} (Touboul, 2009). An accurate numerical scheme requires precise estimation of the membrane potential and the adaptation current at firing times. It is therefore necessary to develop specific integration schemes that treat correctly the numerical errors at the discontinuities without compromising the computational efficiency of the model.

Traditional time-stepping methods have to be modified to prevent the loss of accuracy at the firing times (Hansel et al., 1998; Shelley & Tao, 2001; Rangan & Cai, 2007). However, an efficient treatment at the firing times cannot avoid a fundamental limitation of generic time-stepping methods that is imposed by the smoothness of synaptic interactions (Shelley & Tao, 2001) and relatively small time steps, usually smaller than 0.01 ms, have to be used in numerical methods, limiting the size of the network that can be simulated accurately. To illustrate our purpose, we consider the commonly used exponential synaptic current $I_s^i = we^{-t/\tau_i}H(t)$, where w is the synaptic weight, τ_i the synaptic time constant, and H the Heaviside step function. Exponential synaptic currents can be reformulated to give an efficient implementation as

$$\tau_i \frac{dI_s^i}{dt} = -I_s^i \tag{1.3}$$

with the jump condition $I_s^i \leftarrow I_s^i + w$ whenever the neuron receives a spike. Due to the discontinuous nature of the exponential currents, a time-stepping method falls into an accuracy of $O(\Delta t)$, and therefore, a sufficiently small time step has to be used to correctly simulate the network. This drawback is illustrated in Figure 1A where a time step of $\Delta t = 0.1$ ms leads to a spurious synchronous propagation along a synfire chain of adaptive quadratic integrate-and-fire neurons. The linear interpolation, combined with the recalibration of the membrane potential, cannot restore the accuracy of the second-order Runge-Kutta methods (RK2) scheme. Artificial synchronizations are created, leading to an incorrect propagating activity. The correct behavior is, however, retrieved for $\Delta t = 0.05$ ms. The choice of a fixed time step for network simulations causes an antinomic problem. On the one hand, the time step is probably too small for slowly varying neurons. On the other hand, it may be too large for neurons that are sharply increasing their membrane potential near the threshold.


(A)
(B)
(C)
(D)
(E)
(F)
(G)
(H)
neuron label
time (ms)
Δt = 0.1 ms
Δt = 0.05 ms
membrane potential
time
Network error (ms)
Simulation duration (s)
VS2
mRK2
Δv (mV) / Δt(ms)
VS2 Δv / error loglog
mRK2 Δt / error loglog
Method order
Network average firing rate
Connection probability

To overcome the crucial problem of determining the correct time step, event-driven strategies have been proposed (Brette, 2006, 2007; Tonnelier, Belmabrouk, & Martinez, 2007). Event-driven algorithms naturally deal with the discontinuities in the dynamics and do not require smooth

---

Figure 1: Performance assessment of voltage stepping. (A) Spike raster plot of synfire chain activity. The network consists of 10 layers, each having 50 adaptive quadratic integrate-and-fire neurons (see equations 1.1 and 1.2), where $f(v) = k(v - v_r)(v - v_t)$ and $k = 0.7$, $v_r = -60$, $v_t = -40$, $v_{reset} = -50$, $v_{th} = 35$, $C = 100$, $I = 0$, $a = 0.03$, $b = -2$, $\tau = 5$, $d = 100$. Successive layers are fully connected with 80% excitatory (random weights in [13,15]) and 20% inhibitory (weights $= -5$) synaptic connections using exponential synaptic currents. Spike times are computed with a modified second-order Runge-Kutta (mRK2) method with $\Delta t = 0.05$ ms (plot at the top) and $\Delta t = 0.1$ ms (plot at the bottom). For $\Delta t = 0.1$ ms, the activity propagates along the chain. The correct behavior is found with $\Delta t = 0.05$ ms when the activity vanishes after the fifth layer. (B) Schematic view, at the neuron level, of the voltage-stepping technique. The voltage state-space is discretized with a voltage step Δv. The state of the neuron is updated each time that a voltage step is completed, producing adaptive time steps that follow efficiently the trajectory of the neuron: time steps decrease near the firing times and increase during slowly varying periods. (C) Network error (ms) versus time cost (s) for the numerical simulation of the inhibitory network with the VS2 ($\Delta v \in [0.05, 0.1]$ mV) scheme (plain curve) and mRK2 ($\Delta t \in [0.002, 0.05]$ ms) scheme (dashed curve). (D) Same legend as in panel C but for the excitatory network. (E) Log-log plot of the inhibitory network error as a function of the voltage step Δv and the time step Δt. The slopes of the regression lines indicate the orders of the methods. The slope for VS2 (plain line) is approximately 2, indicating that VS2 is a second-order method for network simulations. In contrast, the slope for mRK2 (dashed line) is approximately 1, revealing the loss of the original second-order mRK2 scheme when simulating a network with nonsmooth synaptic interactions. (F) Same legend as in panel E but for the excitatory network. (G) Orders of the VS2 and mRK2 methods as a function of the network activity (average firing rate). The network consists of $N = 100$ adaptive quadratic integrate-and-fire neurons with parameters given in section 3. The neurons are connected all-to-all, and the weights are randomly drawn from the interval [1, 2]. Simulations were performed for different input currents $I \in [60, 150]$ mA so that the network average firing rate is within the range [50, 90] Hz. For each network activity, the network error is plotted as a function of the voltage step for VS2 and the time step for mRK2. The order of the method is then estimated as the slope of the linear regression in the log-log error plot. The plain and dashed curve represents the order of VS2 and mRK2, respectively. (H) Orders of the VS2 and mRK2 methods as a function of the network connectivity (probability of connection). Same legend as in panel G except that the input current $I = 60$ and the synaptic weights $w = 2$. The connectivity is randomly generated with probability p. The plain and dashed curves represent the order of VS2 and mRK2, respectively.

postsynaptic changes. However, they are limited to a small class of spiking neuron models, mainly the LIF and some variants for which exact computation of spike timings is possible. In a recent paper, Zheng, Tonnelier, and Martinez (2009) proposed a new integration method, the voltage-stepping scheme, for the generic simulation of spiking neurons. The technique originates from a local approximation of the nonlinear neuron model with a LIF model that allows event-driven computation where the events adapt to the voltage variation of the membrane potential. Zheng et al. (2009) evaluated the efficiency of voltage stepping for simulations of single neurons. In this note, we extend this previous work by considering network simulations and more complex neurons (nonlinear integrate-and-neurons with adaptation). We propose a network implementation of voltage stepping using the discrete event system specification (DEVS) (Zeigler, Praehofer, & Kim, 2000) and assess its performance through simulations and comparisons with a modified time-stepping scheme of the Runge-Kutta type.

2 Voltage Stepping for Network Simulation

The basic idea of the voltage-stepping approach is to approximate locally the nonlinear part of the neuronal dynamics by a linear variation so that a local event-driven algorithm can be used. The approximation is achieved using a discretization of the voltage state-space with a fixed voltage step Δv. The numerical integration is performed through successive computations of timings at which a voltage step is achieved. As seen in Figure 1B, this approach induces local events that could be seen as implicit and adaptive time steps leading to a precise approximation of the firing time. The voltage-stepping scheme iterates between the following two steps: (1) approximate the original nonlinear neuron model near the current voltage v_0 with a LIF neuron and integrate formally the equations and (2) search the exit time at which the membrane potential reaches $v_0 \pm \Delta v$ and advance the simulation at that time. These steps are detailed below.

2.1 Voltage-Dependent LIF Approximation and Integration. We recall here the basic steps of the voltage-stepping method presented in Zheng et al. (2009) and extend the formalism for networks with multiple synapses. Let us consider that a given neuron in the network has a membrane potential, v, in the voltage interval $v \in]v_0 - \Delta v, v_0 + \Delta v[$ (for convenience we use a voltage step of $2\Delta v$). The original nonlinear function $f(v)$ is replaced by $\tilde{f}(v) = -g(v - E)$, a linear approximation of f on the interval $v_0 \pm \Delta v$ where g is a voltage-dependent conductance and E a local resting potential obtained from the linear interpolation (see section A.2). Replacing f by $\tilde{f}$ in equation 1.1 and using equation 1.3 lead to the linear differential system,

$$\frac{dX}{dt} = AX + B, \tag{2.1}$$

where $X = (X_{new}\ X_{syn})^t$ is the local state vector composed of two blocks

$$X_{new} = \begin{pmatrix} v \\ w \end{pmatrix} \quad \text{and} \quad X_{syn} = \begin{pmatrix} I_s^1 \\ I_s^2 \\ \vdots \\ I_s^n \end{pmatrix}$$

describing, respectively, the state of the neuron and the exponential currents with the different synaptic time constants, τ_i (note that a change of variable $I_s \leftarrow I_s/C$ has been made), and n is the number of synaptic inputs. The matrix A is a block matrix and B is a vector given by

$$A = \begin{pmatrix} A_{new} & A_{syn} \\ 0 & A_{exp} \end{pmatrix}, B = \begin{pmatrix} B_{new} \\ 0 \end{pmatrix}$$

where the index new is related to the neuron, syn to the inputs, and exp to the exponential synaptic currents. Here, A_{new} is a matrix of order 2, A_{syn} is a $2 \times n$ matrix, and A_{exp} a diagonal matrix of order n, given by

$$A_{new} = \begin{pmatrix} -g & -1/C \\ ab & -a \end{pmatrix}, A_{syn} = \begin{pmatrix} 1 & \dots & 1 \\ 0 & \dots & 0 \end{pmatrix}$$
$$A_{exp} = diag(-1/\tau_1, -1/\tau_2, \dots, -1/\tau_n)$$

and B_{new} is a vector with two entries

$$B_{new} = \begin{pmatrix} gE + I/C \\ -abv_r \end{pmatrix}.$$

The formal resolution of the differential system, equation 2.1, is detailed in section A.2 where the approximation of $v(t)$ and $w(t)$ are analytically given.

2.2 Computation of the Events. The second stage of voltage stepping is to compute the exit time t^*—the time at which the membrane potential trajectory reaches the threshold value $v_0 \pm \Delta v$ and enters a new voltage interval. The voltage-dependent approximation of the dynamics previously derived can be formally integrated using standard techniques of linear differential equations, and the threshold crossing is rewritten as a root-finding problem. The numerical integration is reduced to the computation of roots that define the local adaptive time steps of the neuron. However, a standard root-finding algorithm, of the Newton-Raphson type, for example, may not converge or may converge to a wrong value because of the nonmonotonicity of the membrane potential v on the considered interval. We use here a technique from event detection in hybrid systems (Girard, 2002a). The method is based on computing two sequences of lower approximation and upper approximation, $\underline{t}_i$ and $\overline{t}_i$, of t^*. The method is described in section A.3 and gives two sequences such that

$$\begin{cases} \underline{t}_i \leq t^* \leq \overline{t}_i \\ \overline{t}_i \rightarrow t^* \\ \underline{t}_i \rightarrow t^* \end{cases} . \tag{2.2}$$

It has been shown that the rate of convergence of the sequences is quadratic (Girard, 2002b), and therefore the computation is very efficient. If the neuron stays in the current voltage interval, the sequences diverge and there is no exit time, indicating that a steady state is reached.

2.3 Network Discrete Event Specification. Several frameworks have been applied to the design of event-driven algorithms for spiking neurons, among them, MVASpike (Rochel & Martinez, 2003), which is based on the DEVS formalism (Zeigler et al., 2000). The DEVS formalism describes the evolution of system components (here, neurons or populations) through three main functions:

- δ_{int}: An update function that returns a Boolean indicating if the state of the component has changed. For a neuron, it returns true when the membrane potential reaches the threshold value. δ_{int} takes the update time as a parameter and performs the vLIF integration. This function is used to update the state of the neuron after the reception of a spike or after a local event.
- $\delta_{ext}()$: An external function that manages the incoming external outputs (incoming spikes). δ_{ext} takes the incoming spike time and synaptic port as parameters. It updates the state of the neuron (by calling $\delta_{int}()$) before updating the synaptic currents.
- $ta()$: An advance function gives the time to the next internal transition. In the case of the local event-driven scheme, it performs the new vLIF approximation (see section A.2) and returns the next local event time (by taking voltage stepping into account).

An additional function, reset, is used to reset the state of the neuron after the emission of a spike ($v \leftarrow v_{reset}$, $u \leftarrow u + d$ for adaptive models).

MVASpike handles the events with an event-driven simulation algorithm that sorts the events, updates the neurons, and propagates the spikes. The general event-driven simulation is sketched in algorithm 1, where we also incorporate possible connection delays between neurons:

Algorithm 1: General event-driven algorithm for the simulation of networks of spiking neurons.

```
enumeration EventType {$\delta_{int}\_event$; $\delta_{ext}\_event$;};
structure Event {Neuron neuron; double time; EventType type;};
PriorityQueue pq ;
for each neuron n in the network do
    Event e(n, n.ta(), $\delta_{int}\_event$);
    pq.insert(e);
end
while (!pq.isEmpty()) do
    Event e1 = pq.dequeue() ;
    if (e1.type == $\delta_{int}\_event$)/*in the case of a local event*/ then
        if (e1.neuron.$\delta_{int}$(e1.time))/*check if $e1.neuron$ triggered a spike*/ then
            for each post-synaptic neuron n connected to e1.neuron do
                Event e2(n , e1.time + connection delay, $\delta_{ext}\_event$);
                pq.insert(e2);
            end
            reset (e1.neuron);
        end
    else
        $e1.neuron.\delta_{ext}$(e1.time, event port);
        /*reception of a spike*/
    end
    Event e3(e1.neuron.ta(), e1.neuron, $\delta_{int}\_event$);
    pq.insert(e3);
end
```

In traditional event-driven simulations of spiking neural networks, the events correspond to the reception or the emission of a spike (Mattia & Del Giudice, 2000; Brette, 2006, 2007; Tonnelier et al., 2007). In our scheme, the events are not only firing times or spike receptions but also the times t^* at which the voltage v reaches a new voltage interval $v_0 \pm \Delta v$. Therefore, the DEVS approach can be used to design spiking neural network simulators, and one has to distinguish between two types of events: global events associated with spike emission or reception and local events corresponding to a significant variation of the membrane potential of the neuron. Note that this event-based formalism easily handles the discrete nature of synaptic interactions and allows an efficient treatment of the discontinuities. The membrane fluctuations due to presynaptic spikes are directly treated within the current voltage interval unlike modified Runge-Kutta methods where spike-spike interactions are not considered. (In Runge-Kutta methods, the spikes produced within the current time step are taken into account at the beginning of the next step, thereby neglecting possible interactions.) As noted in (Rangan & Cai, 2007), the first spikes computed within a large time step may jeopardize the simulation via spike-spike interactions in a way that the remaining spikes within the time step are spurious.

3 Numerical Results

The local event-driven method resulting from a linear interpolation of the nonlinear spike-generating current at the boundaries of a voltage interval (see section 2.1) induces a second-order numerical scheme VS2 (Zheng et al., 2009). We compare the performance of VS2 to a standard fixed time-step integration scheme of order 2. The modified Runge-Kutta method with linear interpolation of the spike time and a recalibration of the membrane potential after the reset (mRK2 hereafter) is a second-order scheme for the simulation of neural networks with smooth synaptic interactions (Shelley & Tao, 2001; Hansel et al., 1998). The simulations are done with an extended version of the MVASpike software that incorporates the local-event-driven method using DEVS. (The MVASpike implementation of VS2 and mRK2 can be downloaded at http://webloria.loria.fr/~kaabimmo/index.php?n=Main.Software.)

We simulate two networks of $N = 101$ all-to-all coupled neurons. The inhibitory (resp. excitatory) network has inhibitory (resp. excitatory) connections randomly distributed in $[-2, 0[$ (resp. $]0, 2]$). The neurons are adaptive quadratic integrate-and-fire neurons given by equations 1.1 and 1.2 with $f(v) = k(v - v_r)(v - v_t)$ and the following parameter values: $k = 0.7$, $v_r = -60$, $v_t = -40$, $v_{reset} = -50$, $v_{th} = 35$, $C = 100$, $I = 70$, $a = 0.03$, $b = -2$, $d = 100$. The neurons are coupled with exponential synapses given by equation 1.3, each neuron receiving 50% of fast synapses, $\tau_1 = 5$ ms, and 50% of slow synapses, $\tau_2 = 30$ ms.

We evaluate the efficiency (i.e., accuracy versus time cost) of the algorithms as follows. The two networks are simulated for 2 seconds of biological time. The exact spike times are provided by a time-stepping simulation with a very small time step of 10^{-6} ms. We compute the error associated with the jth neuron as

$$E_j = \frac{1}{M_j} \sum_f |t_f - t_{ap}|,$$

where M_j is the number of spikes, t_f are the exact spike times, and t_{ap} are the approximate spike times obtained by the VS2 or the mRK2 schemes. In all the simulations, the time step or the voltage step is sufficiently small so that no spike is missed. We measure the network error of a simulation as the average error over the neurons:

$$E = \frac{1}{N} \sum_i E_i.$$

The network error for the two test networks as a function of the time cost of the algorithms (duration of the simulation) is shown in Figures 1C and 1D. It is clear that for a given simulation duration, the VS2 scheme gives a more accurate description of the spiking activity. Moreover, when high accuracy is required, VS2 significantly outperforms the mRK2 method. This is explained by the inherent properties of the event-driven scheme obtained from the voltage-stepping technique where the time step is implicitly adapted to the membrane potential fluctuations. The variable time step allows us to speed up simulation when the neurons are at rest or weakly active and to increase precision when the neuron presents strong variations. Moreover, the time steps are defined individually and independently for each neuron in the network, and therefore the quickest varying neurons do not slow down the simulation of the network.

In Figures 1E and 1F, the error is plotted as a function of the voltage step and the time step in a logarithmic scale. The orders of the methods are estimated as the slopes of the linear regressions in the log-log plots. We observe that VS2 maintains a second-order accuracy for the numerical simulation of connected networks. Numerically, the order is 2.06 for the inhibitory network and 2.44 for the excitatory network. In contrast, the error of the mRK2 scheme decreases linearly with the step size and, therefore mRK2 behaves like a Euler scheme (order of 1.003 for the inhibitory network and 1.166 for the excitatory network). The accuracy of the mRK2 scheme is degraded by the nonsmooth nature of the synaptic interactions. A classic second-order Runge-Kutta scheme produces an error having the following form:

$$e = \underbrace{0(\Delta t^2)}_{\text{integration error}} + \text{synaptic interaction error.}$$

The synaptic interaction error (denoted *sie* hereafter) is a first-order error composed of a spike detection error and a spike reception error. The latter is an error generated by the artificial delay due to the spike propagation introduced by the time-stepping scheme. Therefore,

$$sie = \underbrace{0(\Delta t)}_{\text{spike detection error}} + \underbrace{0(\Delta t)}_{\text{spike reception error}} .$$

The spike detection error can be decreased by one order by using a linear interpolation to estimate the correct spike time (Shelley & Tao, 2001; Hansel et al., 1998). In contrast, the spike reception error can be avoided by using synaptic delays (Morrison, Straube, Plesser, & Diesmann, 2007) or can be decreased by several orders using sufficiently smooth synaptic interactions (Shelley & Tao, 2001). Therefore, unlike voltage stepping, the error after one spike is of order Δt, and a smooth synaptic interaction or synaptic delay is required to restore the accuracy of the scheme (here, the existence of the derivative at $t = 0$ is necessary to obtain second-order accuracy). For these reasons, the mRK2, and higher-order schemes of Runge-Kutta type perform like a first-order method for the simulation of networks with exponential synaptic currents (see equation 1.3).

We then study whether the order of the methods is affected by the network activity (average firing rate) and connectivity (probability of connection). As shown in Figures 1G and 1H, the second order of the VS2 method does not depend on the network firing rate and the level of connectivity. The mRK2 method behaves like a first-order method independent of the network activity (see Figure 1H). We note, however, that the order of mRK2 increases for low probabilities p of connection (see Figure 1H), as mRK2 is a second-order method for uncoupled neurons. For a low level of connectivity, the number of synaptic events decreases, and the contribution of the first-order synaptic interaction error becomes less important. Nevertheless, mRK2 is not a second-order method for the simulation of connected networks.

4 Conclusion

Recent efforts have been made to develop efficient schemes for the numerical simulation of spiking neural networks. Exact methods (Brette, 2006, 2007; Tonnelier et al., 2007) where the spike timings are found exactly (up to machine precision) and fast methods (Rangan & Cai, 2007) have been proposed. On the one hand, these methods are based on some analytic expressions for the state variables and are therefore limited to simple models (leaky or quadratic integrate-and-fire). On the other hand, the accuracy of generic time-stepping methods (Hansel et al., 1998; Shelley & Tao, 2001) is severely limited by the smoothness of synaptic interactions and is not computationally efficient in network simulations due to unnecessary updates of inactive neurons.

An alternative technique, the voltage-stepping method, has been recently proposed for combining the generic nature of time-stepping schemes and the efficiency of event-driven approaches (Zheng et al., 2009). Promising results were achieved for the simulation of single neurons. Nevertheless, performance comparison on network simulations was still lacking. Here, we assessed the performance of voltage stepping by considering network simulations and more complex neurons (quadratic integrate-and-fire neurons with adaptation, coupled with multiple synapses). For network simulations, the discrete event system specification formalism was applied to voltage stepping. We demonstrated numerically that the method outperforms the time-stepping schemes of Runge-Kutta type in terms of speed and accuracy. The original order of voltage stepping is preserved in network simulations independent of the network activity and connectivity. This outcome results from the efficiency of the activity-dependent time discretization implicitly generated by the voltage-stepping scheme for each neuron in the network.

The numerical integration is reduced to root-finding methods for which efficient techniques exist. Although the complexity increases with the number of synaptic currents, it is still manageable in practice, as most neural network models do not use as many as four types of synaptic currents. Finally, our work can be extended to conductance-based synaptic currents that are more biologically plausible. This extension could be done through a complete linearization of the differential system and will require a conductance step in addition to the voltage step.

Appendix: Mathematical Details of Voltage Stepping

A.1 The vLIF Model. The linear interpolation of $f(v)/C$ at the boundaries of the voltage interval $v_0 - \Delta v$ and $v_0 + \Delta v$ is $\tilde{f}(v) = -g(v - E)$, where

$$g = -\frac{f(v_0 + \Delta v) - f(v_0 - \Delta v)}{2C\,\Delta v}$$

and

$$E = v_0 - \Delta v + \frac{f(v_0 - \Delta v)}{gC}.$$

A.2 Formal Integration of the vLIF Network. The solution of system 2.1 is of the following form:

$$X(t) = C_1^*(t)X_1^*(t) + C_2^*(t)X_2^*(t) + \sum_{i=1}^{n} C_i(t)X_i(t).$$

$X^{*}{}_1(t)$, $X^{*}{}_2(t)$, and the $X_i(t)$ are determined from the eigenvalues and eigenvectors of A.

The solution is obtained by solving $\frac{dX}{dt} = AX$ and applying the variation-of-constants method to solve $\frac{dX}{dt} = AX + B$ (if A is invertible).

Let $\Delta = (a - g)^2 - 4\frac{ab}{C}$, the eigenvalues of A , for $\Delta > 0$, are

$$\lambda^*_{1,2} = -\frac{a+g}{2} \pm \frac{\sqrt{\Delta}}{2}$$
$$\lambda_i = -\bar{\tau}_i \quad 1 \le i \le n$$

for $\Delta > 0$, where $\bar{\tau}_i = 1/\tau_i$. If $\Delta < 0$, λ^*_1 and λ^*_2 are complex conjugate that we write as $\lambda^*_{1,2} = \alpha \pm i\beta$. The eigenvectors of A are

$$e^*_1 = \begin{pmatrix} \gamma_1 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix};\ e^*_2 = \begin{pmatrix} \gamma_2 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix};\ e_i = \begin{pmatrix} \gamma^i_v \\ \gamma^i_w \\ 0 \\ \vdots \\ 1 \quad \text{(on the } (i+2)\text{th } element) \\ \vdots \\ 0 \end{pmatrix},$$

where

$$\gamma_i = \frac{a + \lambda^*_i}{ab}$$
$$\gamma^i_v = \frac{a - \bar{\tau}_i}{(a - \bar{\tau}_i)(g - \bar{\tau}_i) + \frac{ab}{C}}$$
$$\gamma^i_w = \frac{ab}{(a - \bar{\tau}_i)(g - \bar{\tau}_i) + \frac{ab}{C}}.$$

We define $I_0 = gE + \frac{I}{C}$ and $R = -abv_r$. The membrane potential $v(t)$ and the adaptation variable $w(t)$ are given by:

- If $\Delta > 0$,

$$v(t) = cst_v + \gamma_1 C^*_{10} e^{\lambda^*_1(t-t_0)} + \gamma_2 C^*_{20} e^{\lambda^*_2(t-t_0)} + \sum_{i=1}^{n} C_i \gamma^i_v e^{-\bar{\tau}_i(t-t_0)}$$

$$w(t) = cst_w + C^*_{10} e^{\lambda^*_1(t-t_0)} + C^*_{20} e^{\lambda^*_2(t-t_0)} + \sum_{i=1}^{n} C_i \gamma^i_w e^{-\bar{\tau}_i(t-t_0)}$$

where

$$\begin{cases} cst_v = \dfrac{\gamma_1(R\gamma_2 - I_0)}{\lambda_1(\gamma_1 - \gamma_2)} + \dfrac{\gamma_2(I_0 - R\gamma_1)}{\lambda_2(\gamma_1 - \gamma_2)} \\ cst_w = \dfrac{R\gamma_2 - I_0}{\lambda_1(\gamma_1 - \gamma_2)} + \dfrac{I_0 - R\gamma_1}{\lambda_2(\gamma_1 - \gamma_2)} \\ C_{10}^* = \dfrac{v(t_0) - cst_v - \sum_{i=1}^n C_i\gamma_v^i - \gamma_2(u(t_0) - cst_2 - \sum_{i=1}^n C_i\gamma_w^i)}{\gamma_1 - \gamma_2} \\ C_{20}^* = \dfrac{\gamma_1(w(t_0) - cst_w - \sum_{i=1}^n C_i\gamma_w^i) - v(t_0) + cst_1 + \sum_{i=1}^n C_i\gamma_v^i}{\gamma_1 - \gamma_2} \\ C_i = \dfrac{I_s^i(t_0)}{C}. \end{cases}$$

- If $\Delta < 0$:

$$\begin{aligned} v(t) = {} & cst_v + e^{\alpha(t-t_0)}\left[\left(C_{10}^*\frac{a+\alpha}{ab} + C_{20}^*\frac{\beta}{ab}\right)cos(\beta(t-t_0)) + \right. \\ & \left. + \left(C_{20}^*\frac{a+\alpha}{ab} - C_{10}^*\frac{\beta}{ab}\right)sin(\beta(t-t_0))\right] + \sum_{i=1}^n C_i\gamma_v^i e^{-\bar{\tau}_i(t-t_0)} \\ w(t) = {} & cst_w + e^{\alpha(t-t_0)}[C_{10}^*cos(\beta t) + C_{20}^*sin(\beta(t-t_0))] \\ & + \sum_{i=1}^n C_i\gamma_v^i e^{-\bar{\tau}_i(t-t_0)} \end{aligned}$$

where

$$\begin{cases} cst_v = v_r - \dfrac{\gamma\beta + R\alpha}{b(\alpha^2 + \beta^2)} \\ cst_w = -\dfrac{\gamma\beta + R\alpha}{\alpha^2 + \beta^2} \\ C_{10}^* = w(t_0) - cst_w - \displaystyle\sum_{i=1}^n C_i\gamma_w^i \\ C_{20}^* = \dfrac{ab}{\beta}\left(v(t_0) - cst_w - \displaystyle\sum_{i=1}^n C_i\gamma_v^i\right) \\ \qquad - \dfrac{a+\alpha}{\beta}\left(w(t_0) - cst_w - \displaystyle\sum_{i=1}^n C_i\gamma_w^i\right) \\ C_i = \dfrac{I_s^i(t_0)}{C}. \end{cases}$$

A.3 Computation of Local Events. Let $d(t)$ be the distance from the membrane potential $v(t)$ to the boundary of the voltage interval

$[v_0 - \Delta v, v_0 + \Delta v]$. We have $d(t) = \min(d_l(t), d_u(t))$ where

$$d_l(t) = v(t) - v_0 + \Delta v, \quad \text{and} \quad d_u(t) = v_0 + \Delta v - v(t).$$

A local event in our scheme corresponds to an exit time t^*, the smallest $t > 0$, so that $v(t^*) = v_0 \pm \Delta v$. The problem is equivalent to

$$\begin{cases} d(t) > 0 \ \ \forall t \in]t_0, t^*[\\ d(t^*) = 0 \end{cases} . \tag{A.1}$$

To solve this problem, successive upper approximations and lower approximations, $\underline{t}_i$ and $\overline{t}_i$, of t^* are constructed. We start with $\underline{t}_0 = t_0$ and $\overline{t}_0 = +\infty$. Sequences are derived from an upper and a lower approximation of $d_e(t)$ ($e = l, u$) on $[\underline{t}_i, t^*]$ defined by two quadratic polynomials:

$$d_e(t) \leq P_e(t - \underline{t}_i) = d_e(\underline{t}_i) + v'(\underline{t}_i)(t - \underline{t}_i) + \frac{(t - \underline{t}_i)^2}{2} M \tag{A.2}$$

$$d_e(t) \geq p_e(t - \underline{t}_i) = d_e(\underline{t}_i) + v'(\underline{t}_i)(t - \underline{t}_i) + \frac{(t - \underline{t}_i)^2}{2} m, \tag{A.3}$$

where

$$m \leq v''(t) \leq M \ \ \forall t \in [t_0, t^*].$$

We define the new values at the next iteration as

$$\begin{cases} \underline{t}_{i+1} \leftarrow \underline{t}_i + r_e^i \\ \overline{t}_{i+1} \leftarrow \overline{t}_i + R_e^i \end{cases} ,$$

where r_e^i and R_e^i are the smallest positive roots of the polynomials p_e and P_e, respectively. If P_e has no positive roots, we set $\overline{t}_{i+1} = +\infty$. Note that two lower and two upper approximations of the exit time are constructed (corresponding to a possible exit at $v_0 - \Delta$ and at $v_0 - \Delta$) and one sequence ($e = l$ or $e = u$) can be removed if its lower approximation is greater than the upper approximation of the other one. Moreover, after an iteration, one may keep only one value for the lower (upper) approximation of the exit time taking the minimum of the two values.

To compute the bounds, m and M, of $v''(t)$ one may observe that

$$v''(t) = cX''(t) = c(A^2 X(t) + AB)$$

where $c = (1, 0, \dots, 0)$. Let $\{x_1, x_2, .., x_p\}$ the 2^{n+2} vertices of the hyper-rectangle containing the state vector $X(t)$, we can write

$$X(t) = \sum_{i=1}^{i=p} \lambda_i(t) x_i, \quad \sum_{i=1}^{i=p} \lambda_i(t) = 1, \quad \forall i \lambda_i \geq 0.$$

Consequently, we have

$$\begin{cases} m = \min_{i=1}^{p}[c(A^2 x_i + AB)], \\ M = \max_{i=1}^{p}[c(A^2 x_i + AB)]. \end{cases}$$

The $\{x_i\}_{1 \leq i \leq 2^{n+2}}$ are obtained using

$$\begin{cases} v_0 - \Delta v \leq v(t) \leq v_0 + \Delta v \\ w_{inf} \leq w(t) \leq w_{sup} \\ \min(0, I_s^i(t_0)) \leq I_s^i(t) \leq \max(0, I_s^i(t_0)) \ \forall 1 \leq i \leq n. \end{cases}$$

We introduce an additional parameter Δw so that $w_{inf} \leq w(t) \leq w_{sup}$; ∀ t ∈ $[t_0, t^*]$ with

$$\begin{cases} w_{inf} = w_0 - \Delta w \\ w_{sup} = w_0 + \Delta w \end{cases}.$$

Even if the variation on w is not important, w_{inf} and w_{sup} are required to run the algorithm. In practice, one chooses a large Δw, for example, equal to 10.

References

Brette, R. (2006). Exact simulation of integrate-and-fire models with synaptic conductances. *Neural Comput.*, *18*, 2004–2027.

Brette, R. (2007). Exact simulation of integrate-and-fire models with exponential currents. *Neural Comput.*, *19*, 2604–2609.

Brette, R., & Gerstner, W. (2005). Adaptive exponential integrate-and-fire model as an effective description of neuronal activity. *J. Neurophysiol.*, *94*, 3637–3642.

Ermentrout, B. (1996). Type I membranes, phase resetting curves, and synchrony. *Neural Comput*, *8*, 979–1001.

Fourcaud-Trocme, N., Hansel, D., van Vreeswijk, C., & Brunel, N. (2003). How spike generation mechanisms determine the neuronal response to fluctuating inputs. *J. Neuroscience*, *23*, 11628–11640.

Gerstner, W., & Naud, R. (2009). How good are neuron models. *Science*, *326*, 379–380.

Girard, A. (2002a). Detection of event occurrence in piecewise linear hybrid systems. In *Proceedings of the 4th International Conference on Recent Advances in Soft Computing* (pp. 19–25). Munich: Technical University.

Girard, A. (2002b). Approximate solutions of ODEs using piecewise linear vector fields. In *5th International Workshop on Computer Algebra in Scientific Computing* (pp. 107–120). Munich: Technical University.

Hansel, D., Mato, G., Meunier, C., & Neltner, L. (1998). On Numerical simulations of integrate-and-fire neural networks. *Neural Comput.*, *10*, 467–483.

Izhikevich, E. M. (2003). Simple model of spiking neurons. *IEEE Transactions on Neural Networks*, *14*, 1569–1572.

Izhikevich, E. M. (2004). Which model to use for cortical spiking neurons? *IEEE Transactions on Neural Networks*, *15*, 1063–1070.

Izhikevich, E. M, & Edelman, G. M. (2008). Large-scale model of mammalian thalamocortical systems. *Proceedings of the National Academy of Sciences*, *105*, 3593–3598.

Mattia, M., & Del Giudice, P. (2000). Efficient event-driven simulation of large networks of spiking neurons and dynamical synapses. *Neural Comput.*, *12*, 2305–2329.

Migliore, M., Cannia, C., Lytton, W. W., Markram, H. & Hines, M. L. (2006). Parallel network simulations with NEURON. *J. Comput. Neurosci.*, *21*, 119–129.

Morrison, A., Straube, S., Plesser, H. E., & Diesmann, M. (2007). Exact subthreshold integration with continuous spike times in discrete-time neural network simulations. *Neural Comput.*, *19*, 47–79.

Rangan, A., & Cai, D. (2007). Fast numerical methods for simulating large-scale integrate-and-fire neuronal networks. *J. Comput. Neurosci.*, *22*, 81–100.

Rochel, O., & Martinez, D. (2003). An event-driven framework for the simulation of networks of spiking neurons. *In Proc. of the European Symposium on Artificial Neural Network*. Bruges: d-side.

Shelley, M. J., & Tao, L. (2001). Efficient and accurate time-stepping schemes for integrate-and-fire neuronal networks. *J. Comput. Neurosci.*, *11*, 111–119.

Tonnelier, A., Belmabrouk, H., & Martinez, D. (2007). Event-driven simulations of nonlinear integrate-and-fire neurons. *Neural Comput.*, *19*, 3226–3238.

Touboul, J. (2008). Bifurcation analysis of a general class of nonlinear integrate-and-fire neurons. *SIAM Journal on Applied Mathematics*, *68*, 1045–1097.

Touboul, J. (2009). Importance of the cutoff value in the quadratic adaptive integrate-and-fire model *Neural Computation*, *21*, 2114–2122.

Zeigler, B. P., Praehofer, H., & Kim, T. G. (2000). *Theory of modeling and simulation*. San Diego, CA: Academic Press.

Zheng, G., Tonnelier, A., & Martinez, D. (2009). Voltage-stepping schemes for the simulation of spiking neural networks. *J. Comput. Neurosci.*, *26*, 409–423.

Received April 16, 2010; accepted October 5, 2010.