

# A simulation tool for modeling the influence of anatomy on information flow using discrete integrate and fire neurons

Maya Maimon · Larry Manevitz

Published online: 4 October 2007  
© Springer Science+Business Media, LLC 2007

**Abstract** There are theories on brain functionality that can only be tested in very large models. In this work, a simulation model appropriate for working with large number of neurons was developed, and Information Theory measuring tools were designed to monitor the flow of information in such large networks. The model's simulator can handle up to one million neurons in its current implementation by using a discretized version of the Lopicque integrate and fire neuron instead of interacting differential equations. A modular structure facilitates the setting of parameters of the neurons, networks, time and most importantly, architectural changes.

Applications of this research are demonstrated by testing architectures in terms of mutual information. We present some preliminary architectural results showing that adding a virtual analogue to white matter called “jumps” to a simple representation of cortex results in: (1) an increase in the rate of mutual information flow, corresponding to the “bias” or “priming” hypothesis; thereby giving a possible explanation of the high speed response to stimuli in complex networks. (2) An increase in the stability of response of the network; i.e. a system with “jumps” is a more reliable machine. This also has an effect on the potential speed of response.

**Keywords** Large scale neural simulator · Temporal discrete integrate and fire · Information theory · Bias or priming hypothesis

## 1 Introduction

It is well known that the human cortex is extremely large in terms of the number of neurons (Braitenberg and Schüz 1998). As a result, there are theories about its

---

M. Maimon · L. Manevitz (✉)  
Department of Computer Science, University of Haifa, Haifa, Israel  
e-mail: manevitz@cs.haifa.ac.il

M. Maimon  
e-mail: mayam@cs.haifa.ac.il

function which can only be tested in very large models. One standard approach is to use “in vivo” models, i.e. animal models. Another approach is to use “ex vivo” models (Marom and Shahaf 2002), i.e. cortical neurons that are grown in a petri dish and electronically monitored and stimulated. An advantage of these models is that they seem to have the same neuronal functionality as the human cortex; a disadvantage is that they are not subject to complete control and thus are not suitable, for example, to explore variants of architectural structure.

Thus, there is an important place for formal models of neurons and neural networks. These models, although necessarily highly caricatured and simplified provide a way to understand, both in theoretical results and in simulations, information processing principles in the brain. In fact, its very simplicity, rather than a defect, can in certain instances be thought of as an *abstraction* of natural networks. Thus, if a certain part of processing can be understood at this level of abstraction, it shows that further details are unnecessary to explain the natural phenomenon.

In addition, of course, such computational models allow the study of different aspects; e.g. of conduction or architecture independently of the myriad other parameters. Such models have been used by many researchers, such as Rolls et al, Deco, Panzeri, Abbott and Marom (Abbott 1999; Panzeri et al. 1999; Rolls et al. 2003; Marom and Shahaf 2002; Treves et al. 1997; Trappenberg 2002). Despite their relative simplicity to natural models, the interactions of such artificial neurons can be very complex; and researchers are forced to make simplifying choices in their modeling tools.

It is well established that the simplest of such models, networks based on the McCullough–Pitts model of the neuron do not contain sufficient information. While they can be used to good effect to study certain aspects of connectivity and so on, they lack any aspect of temporal effects and relationships.

The most famous “realistic” models of the neuron are the primitive Lapicque Integrate and Fire neuron (Lapicque 1907) and its successors and the more descriptive models based on the famous Hodgkin–Huxley model (Hodgkin and Huxley 1952). The latter, being a complex interaction of several differential equations is mostly thought to be too complex to use in large scale models; and the typical application tends to use a version of the integrate and fire model. Nonetheless, the Lapicque model itself involves differential equations, and modeling large scale requires the interactions between a huge number of such equations. Such a system can not be solved analytically (thereby, removing one of the compelling reasons to model with such equations); but must be solved numerically. This is possible, but involves numerous questions of numerical analysis.

In this work, we make the simplifying assumption to work with a discretized version of the integrate and fire neuron; thus our model of the network can be thought of as a version of a complex “cellular automaton”; where each cell is one of our neurons; but the connections are more complex than are usually modeled in such automata (Wolfram 2002).

This simplified model maintains at the level of the neurons the temporal effect, and we find that working with such a model is simpler both logically and computationally. If it is accepted that the important information from a computational perspective is represented in such temporally discrete neurons, the computational challenges are mitigated and there is a huge computational reduction. Thus experimentation with

different parameters, architectures and encodings can be accomplished in a reasonable time.

This idea is demonstrated here with the development of a version of a computational tool. Changing, for example, the architecture of the network (which generates the spike history of the neurons) can be done by simply changing parameters during the processing of the model.

The current implementation (the second such) works on a dual 64 bit-processor 2.4 MHz AMD with 16 Gigabytes of internal memory. On this machine, in principle 50 million neurons can be implemented. We actually experimented with as many as one million neurons; although most of our results were done with “small” networks of 10,000 to 100,000 neurons.

Monitoring such a large network carries its own problems. A central tool we decided to implement<sup>1</sup> is to use information theoretic measures, in particular the rate of flow of mutual information.

This is accomplished by implementing the ideas of Treves, Panzeri, Shultz, Treves and Rolls (Panzeri et al. 1999) which we describe in Sect. 3.

In our preliminary experiments we investigated the effect of changing the architecture of a rectangular piece of “cortex” allowing or disallowing “jumps” which, in some sense correspond to an interpretation of white matter.

The results show: (1) more connections (“dendrites”) results in faster mutual information, meaning the mutual information is higher earlier in time. Firing rates are higher and faster. That is, the spikes occur earlier in time, producing the information faster. The additional connections increase the chance to emit spikes, resulting in a higher firing rate and faster rate of mutual information. (2) The use of “jumps” verifies the “bias” hypothesis; i.e. a small number of forward jumps, as opposed to “percolations” of information, causes the mutual information curve to rise much faster. This helps to explain how responses can be so fast in such complex networks. (3) The use of “jumps” also results in a more stable and predictive network which has many possible consequences; including allowing more reliable guesses of the stimulus. This implies that it is possible for the cortex to function by making early guesses as to the total incoming signal; only correcting when necessary.

## 2 The modeled neuron and networks

### 2.1 The neuron

The main functionality of the neuron that we are interested in was modeled by Lapique in the integrate and fire neuron (I&F) (Abbott 1999) using a simple capacitor-resistor circuit. The capacitance and leakage resistance of the cell membrane is represented there by a parallel capacitor and resistor. The generation of the action potential corresponds to a discharge of the capacitor. Charging the capacitor to a specific threshold potential causes the generation of a spike and resets the capacitor, i.e. the membrane potential.

---

<sup>1</sup>Under the influence of Leonardo Franco—personal conversations with LM and (Rolls et al. 2003).

The more sophisticated Hodgkin and Huxley model (HH) (Hodgkin and Huxley 1952) describes the change of the membrane potential with time in details of the voltage dependent ionic current. HH quantifies the process of spike generation with a set of four coupled differential equations.

The I&F model, also models the membrane potential with differential equations. It simplifies the HH model by abstracting away some of the underlying processes. This model can be used to clarify the properties of neural networks and the implications of synaptic connections in such networks. (See simulators like, The Genesis simulator 1994–2006, for comparison.) On the other hand, close modeling of the spike generation process allows us to be concrete on other issues that are likely to be relevant (Abbott 1999).

Unfortunately, such coupled differential equations can not be solved analytically. Although they can be solved numerically, there are many details to be considered for efficient numerical integration, including the stability of solutions and a better utilization use of computational resources. The difficulties involved impede simulating neuron ensembles and in many cases are numerically too intensive to be used in very large network simulations.

In this work, we use a temporally discretized integrate and fire model, which is simpler both logically and computationally. The idea is to pursue the I&F approach further by abstracting more details by removing the differential equations. This transforms the neuron into a simple kind of automata.

Each neuron has a fixed number of inputs corresponding to dendrites. This is a parameter of the model; but we typically used a fan-in of 21. Each dendrite has a particular weight, a real number from 0 to 1; we typically chose random weights between 0 and 0.5. There was no learning implemented in the model as yet; although the design allows it.

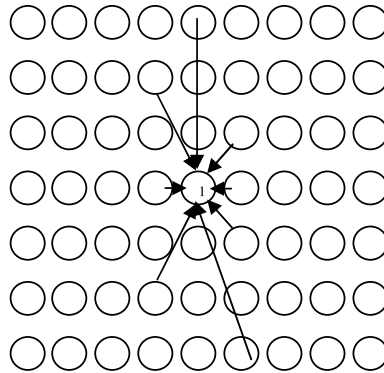
In addition the neuron has a register containing the current “voltage” or “activation” level. At each time slice the neuron updates this register level by adding the weighted sum of inputs from the dendrites (a linear combination). Each neuron has also a “leak” parameter and a “decay” parameter. Since in our implementation, we assumed each time slot was a milli-second, we chose the leak parameter as 0.2 and the decay as 1 following the physiological information as reported in (Panzeri et al. 1999). (Note, that all parameters can be set at run-time.) There is also a “noise” parameter which adds a random value to the voltage. The quantity of the noise is also a run-time parameter.

The rules of the automata indicate that the neuron “fires” (outputs 1) if the register passes the threshold. In this case the voltage is reset to zero and no firing is allowed prior to the refraction time. In our model, the refraction time was set as 2 ms.

Thus, each time cycle the neuron fires if its potential passes a threshold and has not fired for at least a refractory period. Thus the decision whether to emit a spike is done by calculating its new potential  $potential_{i+1} = potential_i * leak - decay + \sum_{j=1}^{inputs} w_j x_j$  where  $w_j$  is the weight of an input  $j$ , and  $x_j$  indicates that “dendrite”  $j$  has received a pulse at time  $i$ .

This combination reflects the original intuition behind the Lapique neuron. However, unlike differential equations, there is no obvious way to predict results without simulations. Note, that when one passes to networks, one can not solve the coupled equations analytically in any case.

**Fig. 1** The incoming dendrites are limited on the horizontal direction



## 2.2 The network

A network of such neurons is created by assigning dendrites of one neuron to “axons” (i.e. outputs) of other neurons. The specific assignments result in different architectural arrangements of the network.

In the current implementation, we assume that the neurons are located in a rectangular array; but that their connections are chosen randomly according to a chosen distribution. The model can be expanded to handle other geometries. In the basic “percolation” model, all connected neurons are within a given (small) horizontal radius. The neighborhood of a neuron is thus limited to a rectangle of neurons that have similar values in the horizontal direction. The specific connections are chosen randomly by a uniform distribution given this constraint. (See Fig. 1.) In the “jump” model (see below), a small percentage of neurons also “jump” a long distance in the horizontal dimension.

We also implemented models with “barriers” to percolation and other minor variants. All random numbers were chosen using the MZran algorithm for random number generation (Numerical Recipes in Fortran 90 1996) slightly adapted to support 64 bit architecture.

The result of this arrangement is that the entire network can be thought of as a complicated cellular automata (Wolfram 2002) although the connectivity is somewhat more complex than typically appears in, e.g. (Wolfram 2002).

The advantage of this arrangement is that, assuming this cellular automata captures the information processing intuition underlying the I&F neuron, the computational challenges are mitigated, the calculations are exact and there are no numerical analysis problems that arise from differential equations.

The use of such a model allows us to try different parameters, different architecture and various encoding techniques in a reasonable time. We are thus able to test theories and ideas about the importance of architectural elements in the cortex, in terms of information flow in the brain.

### 3 Information theory methods

Assuming the use of such neurons in a large model, the next step is analyzing the results of such large networks. In our work, we have implemented a tool that tracks the mutual information. This tool works according to the method that was described in Rolls et al. (2003), and was based on the approach taken by Panzeri et al. (1999); Rolls and Deco (2002). The main details and insights from this reference are presented below.

There are some reservations in the literature about the quality and applicability of this method in some circumstances; The mean number of spikes in the considered time window should be small (Panzeri et al. 1999; Bezzi et al. 2002). However, our testing, indicates that this method is reliable for our purposes. More specifically, Panzeri et al. (1999) indicates that tracking many neurons reduces the period of time the formulas remain accurate. Panzeri et al. (1999) showed that one can use the formulas on about 10–15 neurons for the time periods of interest. Accordingly, although we track 50 neurons, we used the formulas for subsets of 7 neurons and reported the average of ten such sets. This is described in Sect. 5.

Information theory measures provide a way to estimate how many stimuli could be encoded as a function of the responses. For a neuronal population  $C$  and a discrete set of stimuli  $S$ , a neuronal response  $n$  for each stimulus is defined as a vector consisting of the number of spikes emitted by each cell during a time window  $t$ . The rate response  $r$  is the vector  $n$  divided by  $t$ .  $r_i$ ,  $n_i$  refer to the value at the  $i^{\text{th}}$  location. A trial is a simulation of the ensemble for a specific stimulus. Multiple trials are performed for each stimulus.

The response set  $R$  consists of all the responses for all the stimuli of  $S$  in all the trials. The Shannon mutual information formula evaluates how much information is shared between  $S$  and  $R$ . Alternatively, it can be viewed as the amount of information that was captured in the responses. The formula uses the probabilities of  $S$  and  $R$  and the joint probability  $P(S, R)$ .

$$I(t) = \sum_{s \in S} \sum_r P(s, r) \log_2 \frac{P(s, r)}{P(s)P(r)}. \quad (1)$$

To use this formula empirically, sufficient trials are needed to estimate the correct probabilities. Unfortunately, even for a few cells, there can not be enough samples. This causes various difficulties including biased probabilities calculation, see Panzeri et al. (1999). Moreover, the sampling error increases exponentially with the number of spikes. That is, with more spikes comes more options for responses, and thus, more trials are needed to assess the probability for each response.

Panzeri et al. (1999) tackled this under-sampling problem by making a Taylor series approximation. Taylor series is adequate for calculating the function value in small neighborhoods. They calculate  $I(t)$ , the mutual information (MI) for time interval  $t$  around 0, using only the first two time derivatives  $I_t$  and  $I_{tt}$ . The refractory period limits the number of emitted spikes. Thus, in a small window, the expected values of  $n$  are usually 0–2 spikes. According to Rolls et al. (2003) the approximation by using the Taylor series up to the second derivatives is accurate in this range where each neuron fires 1–2 spikes. (Note that, for our purposes,  $I(0)$  is 0, and higher derivatives are negligible in a small population with such small values.) This short time

limitation is reasonable since the relevant sensory information is transmitted in a short time (Rolls et al. 2003). Hence the mutual information  $I(t)$  available is calculated by

$$I(t) = t(I_t) + \frac{t^2}{2}(I_{tt}). \tag{2}$$

Following Rolls et al. (2003) (and using their notation), the instantaneous mutual information rate  $I_t$  is a summation of the instantaneous information rate of each single cell from ensemble C cells. The  $\langle \rangle_s$  stands for an average across stimuli.

$$I_t = \sum_{i=1}^C \left\langle \bar{r}_i(s) \log_2 \frac{\bar{r}_i(s)}{\langle \bar{r}_i(s') \rangle'_s} \right\rangle_s. \tag{3}$$

This term evaluates how the average of the responses (across trials) of a cell are similar to his mean response across stimuli.

The second time derivative of the information accounts for the pairwise correlations. The expression for the second time derivative of the mutual information  $I_{tt}$  breaks up into three terms:  $I_{tta}$ ,  $I_{ttb}$  and  $I_{ttc}$ . According to Rolls et al. (2003),  $I_{tta}$  is a negative value which reduces the amount of redundancy produced by concurrent firings. The second term,  $I_{ttb}$ , adds the actual information that results from the correlation i.e. synergy. The last term,  $I_{ttc}$ , stands for stimulus dependent correlations.

$$I_{tt} = I_{tta} + I_{ttb} + I_{ttc}, \tag{4}$$

$$I_{tta} = \frac{1}{\ln 2} \sum_{i=1}^C \sum_{j=1}^C \langle \bar{r}_i(s) \rangle_s \langle \bar{r}_j(s) \rangle_s \left[ v_{ij} + (1 + v_{ij}) \ln \frac{1}{1 + v_{ij}} \right], \tag{5}$$

$$I_{ttb} = \sum_{i=1}^C \sum_{j=1}^C [\langle \bar{r}_i(s) \bar{r}_j(s) \gamma_{ij}(s) \rangle_s] \log_2 \frac{1}{1 + v_{ij}}, \tag{6}$$

$$I_{ttc} = \sum_{i=1}^C \sum_{j=1}^C \left\langle \bar{r}_i(s) \bar{r}_j(s) (1 + \gamma_{ij}(s)) \log_2 \left[ \frac{(1 + \gamma_{ij}(s)) \langle \bar{r}_i(s') \bar{r}_j(s') \rangle'_s}{\langle \bar{r}_i(s') \bar{r}_j(s') (1 + \gamma_{ij}(s')) \rangle'_s} \right] \right\rangle_s. \tag{7}$$

The shuffling method (Rolls et al. 2003) for estimating the  $I_{ttc}$  validity was implemented as well. The Monte Carlo procedure mainly consists of reevaluating the term for the data when the trials are shuffled within a stimulus. The “confidence limit” was kept as 2 std from the average.

The above formulas are defined in terms of two types of correlations:

$$\gamma_{ij}(s) = \frac{\overline{n_i(s)n_j(s)}}{\bar{n}_i(s)\bar{n}_j(s)} - 1, \tag{8}$$

$$\gamma_{ii}(s) = \frac{\overline{n_i(s)n_j(s)} - \bar{n}_i(s)}{\bar{n}_i(s)\bar{n}_j(s)} - 1, \tag{9}$$

$$v_{ij} = \frac{\langle \bar{n}_i(s) \bar{n}_j(s) \rangle_s}{\langle \bar{n}_i(s) \rangle_s \langle \bar{n}_j(s) \rangle_s} - 1 = \frac{\langle \bar{r}_i(s) \bar{r}_j(s) \rangle_s}{\langle \bar{r}_i(s) \rangle_s \langle \bar{r}_j(s) \rangle_s} - 1. \quad (10)$$

The mutual information (MI) formulas uses only the spike counts to express the amount of synergy and redundancy of the introduced correlations. Thus, the computations are done relative to a discrete set of stimuli and a set of possible responses, aka trial results, for each stimuli. This relative nature of the measure is useful when comparing different networks. The result of these formulas describes how many bits of information are present in each architecture.

The different components of the formulas relate to the encoding methods. The two main ways to encode information are (a) the firing rates and the spikes counts and (b) the correlations. Correlations occur when the firings of neurons are related for only some of the stimuli. For example, if the firing of two cells is concurrent for only one of the stimuli, it would be easier to know which stimulus was applied. One can see that in MI terms, this increases the ability to discriminate between the stimuli. The different parts of the formulas  $I(t)$  correspond to different kind of encodings (Rolls et al. 2003). This is very advantageous when we compare architectures. Thus, the formulas allow us to understand how different encoding methods are affected in different architectures as well as the rate and the flow of information in the brain through time.

Since the formulas require a distribution, hundreds of trials are needed for each stimulus in each test. Due to this time consuming limitation, most of the experiments in this paper are demonstrated for neuronal populations of about 10,000. This exhibits the same behavior as the populations of 100,000 which we tried as well. To be clear, the limitation in computation time is due to the information theoretic requirements, not to the simulation of the neurons themselves.

The different trials are produced in our tool by the addition of Gaussian noise. Without noise, the trials would be identical for each stimulus, and no distribution would be produced. To compute the correct probability, enough trials are required to cover all the possibilities that occur. To verify that the sampling error is decreasing we checked the information for varying numbers of trials. This gives some estimation on the decrement of the error. Since additional trials produces a more exact distribution, this can be used to give an indication of the variation on *an individual* trial; i.e. how robust a system is to the addition of noise. In other words, *the number of trials* needed to give a reliable estimate of mutual information gives an indication of the variance in the results of computation under noise which thus indicates how *reliable* the system is. The methodology is described in Sect. 5 where it is used to show that some architectures may be more reliable than others.

As explained above, the formula is valid for short time intervals. In addition, when more cells are checked and higher number of spikes are produced the time validity is reduced even further (Panzeri et al. 1999; Bezzi et al. 2002). Previous tests (Rolls et al. 2003) were done for fewer cells. In the tests below, we checked the information for 50 representative neurons. To overcome this obstacle, we computed the information 10 times, for different randomly selected 7 neurons, out of the 50. The total information reported is the average of all the groups of 7 representatives. The number of groups and selected neurons can be determined as external parameters in the implemented system.



## 4 Tool capacity and capability

Two software modules were developed, the simulator and the information component.

The simulator is an implementation of the model, which produces a spike train history of a selected subset of neurons. The usual procedure chooses neurons randomly, from specific locations, or by other criteria. In the runs discussed in this paper, the number of selected neurons was usually set to 50.

The spike train of these neurons is then used for further analysis by the information module. The spike train history is also saved and is available for further analysis. The simulator, in its current implementation, allows trials to be run in parallel, but the neurons are processed sequentially. However, the essential mechanism is well suited to future parallelization, and this was kept in mind during the design of the data structures. The program is implemented in the programming language C. The current experiments used a 64-bit machine with dual processors and 16 Gigabytes of memory. This is sufficient to, in principle, simulate up to 50 million neurons with a fan in of about 21.

The choice of topology is flexible, allowing the potential implementation of relatively realistic anatomy. The connectivity of the neurons can be chosen either randomly or specifically. In addition, it is easy to implement specific constraints, e.g. that connectivity is within a certain radius of a neuron.

There are many different external parameters that can be assigned, such as decay rate, leak rate, threshold, refractory period, size of connectivity fan-in and the number of neurons whose spike history will be recorded. The design of the simulator is by simple building blocks, so that new knowledge and requirements can be easily incorporated into the program.

In addition, the size of the time discretization is a parameter; both the total number of seconds and the number of parts in a second.

There is also the option of adding an external input to the neurons. This can be tuned by setting the time and location at which the external input will be applied.

An efficient algorithm (Numerical Recipes in Fortran 90 1996), modified to support 64-bit architecture machines, for random number generator is used; this is crucial because of the huge number of random numbers needed. The random generator seeds can also be changed.

Information theoretic techniques as described in the previous section were implemented in MATLAB. This requires multiple trials for each stimulus. Variation in the responses is obtained by the addition of a noise factor, generated by a Gaussian distribution, that is added to the potential of the neurons. The polarity of the noise (inhibitory, excitatory) is an external parameter. Other parameters of the noise (e.g. distribution parameters) are also external parameters.

In the tests reported in this paper, we simulated 100 trials for each stimulus. For comparison, in Rolls et al. (2003) 20 trials for 8 cells were run. We used the methodology described below in Sect. 3, to verify that 100 trials suffice.

Our machine has 16 gigabyte memory with a 2.4 gigahertz processor. This allows us to store as many as 50M neurons with 21 inputs to each one, which can be thought of as a bi-dimensional piece of cortex of  $1000 \times 50,000$ . The largest model we have

actually run is 1M neurons with 21 inputs. Thus a run of 3000 logical time steps corresponds to simulating 3 seconds in the real biological world.

Under our implementation it takes about 1500 seconds to simulate 1 trial on the average (across stimuli and trials) for 1M neurons. A simulation of one trial for  $10^4$  neurons takes 1.84 seconds and  $2 \times 10^4$  neurons takes 4.4 seconds. In each of our tests we used 21 inputs for each neuron. This number is chosen according to the proportion of neurons and connections as in Marom and Shahaf (2002). The complexity upper bound is linear on the number of connections and inputs as can be deduced from the above model. The actual running time depends on the total neurons and connectivity as well as processes memory.

## 5 Experiments and results

Having developed the simulator and information tools; we ran the following classes of experiments:

(i) Preliminary experiments. These show that the results themselves are consistent with the literature and that simple changes in parameters are in consonance with common intuition.

We reran the tests with different parameters, e.g. different size networks, providing a way to assess the inner consistency of new results. The parameters were varied for the topology and number of neurons, connectivity, integrate and fire parameters and time parameters. We ran the tests on different stimuli, different discretization time bins, and we chose different locations for applying the stimulus or checking the MI.

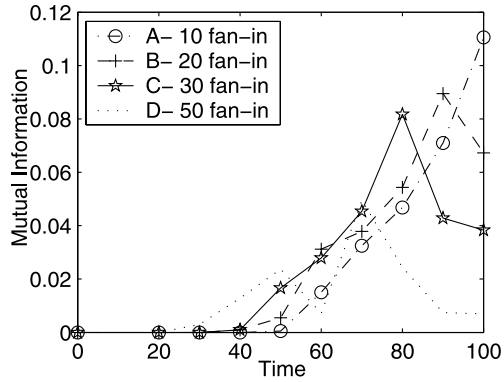
We tested the network with parameters that seem common in the literature (Panzeri et al. 1999; Schüz 1998): total time simulation of about 100 ms; time discretization slices of about 1 ms; and refractory periods of 2 ms. The choice of fan-in; i.e. number of “dendrites” to the neuron was chosen as 21. Non-determinism in the network is established by adding noise parameters to the neuronal responses. The most basic networks were rectangular with local connectivity (i.e. neurons could only connect to other neurons within a certain horizontal distance). See Fig. 1.

For example, we tested whether the approximation of the mutual information as given by formula (2), is valid for the consequences of sending no information to the network. No information is transmitted when there is no difference in response as the stimulus is varied, even if there is a distribution of response for a given stimulus. This was done in different ways, e.g., by using an architecture with a “barrier” between the stimulus and response. In this case, the formulas do in fact result in extremely small mutual information as anticipated.

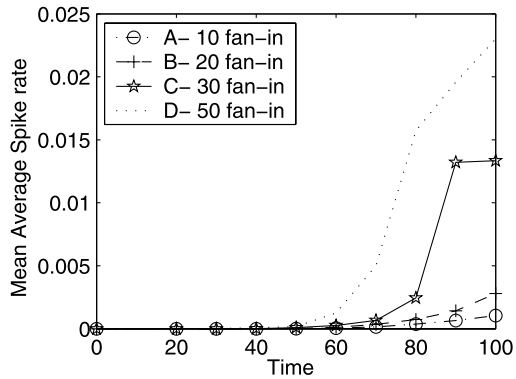
(ii) Experiments testing the relationship of connectivity to the rate of growth of mutual information and to the total amount of mutual information.

We varied the connectivity; i.e. fan-in to each neuron for a network of 10,000 neurons. This network was presented as a two-dimensional array of  $50 \times 200$  neurons. Each neuron was randomly connected to the outputs (“axons”) of neurons within a specified radius of the receiving neuron. Initialization of the weights were uniformly distributed real values between 0 and 0.5. 50 neurons were randomly chosen to be tracked for the information measures. The results of these operations for 10, 20, 30 and 50 dendrites are exhibited in Figs. 2, 3, 4 and 5.

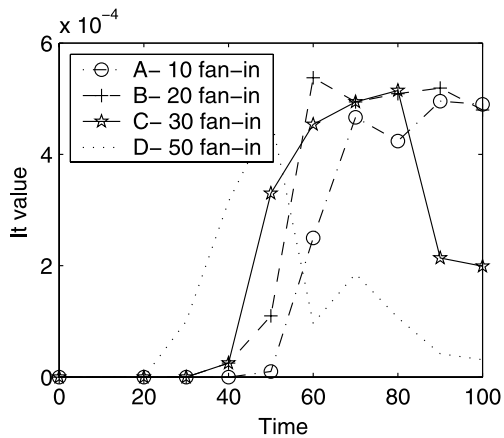
**Fig. 2** MI for different fan in



**Fig. 3** Mean average of spikes across trials and stimuli for different fan in

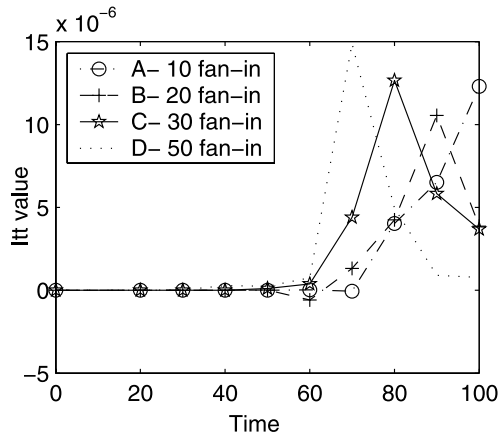


**Fig. 4** First derivative ( $I_t$ ) of MI for different fan in



As expected, increasing the number of dendrites causes both the spike rate (Fig. 3) and the mutual information (Fig. 2) to rise earlier. This was observed also in variations of this test. We present here typical results of this class of tests. Figure 3 shows that the mean averages of firing across both trials and stimuli is increasing with the fan-

**Fig. 5** Second derivative ( $Itt$ ) of MI for different fan in



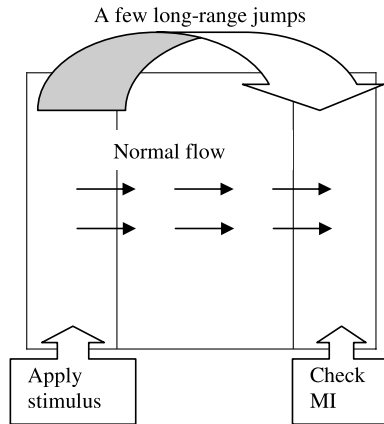
in. In fact, the rate of increase also grows in time. The four lines which represents the four different architectures do not intersect in this figure. This increase in the mutual information can be seen from formula (2) as arising from the first term which is related to the firing rate.

The first derivative,  $It$ , depicted in Fig. 4, and given by formula (3) increases over time until some bound before decreasing. The formula (3) means that when the firing of each cell on the average (over trials) are less different from the mean response for the different stimuli, the cell yields less information to discriminate between the stimuli. This implies that the observed phenomena could be expected, since, until a limit is reached, increase in the firing rate would allow a greater variance in the responses (which are directly related to the firing rate) but eventually less variance (since most of the cells would need to be firing most of the time). Thus, we see that the extent at which additional fan-in contributes to the first derivative is limited; a fan-in of 50 decreases  $It$  and hence less information in Fig. 2.

The second derivative,  $Itt$ , contributes positive values to the overall information when the correlations exist only for some of the stimuli. Once again, this helps in discrimination between the stimuli yielding more mutual information. The second term will contribute negative values when the correlation terms in (2) have more redundancy. That is, since the number of firings is large enough so that there are correlations regardless of stimulus. Mathematically, this is reflected in the  $Ittb$  term being less positive than the negative value contributed by  $Itta$  in formula (2). The maximum value of  $Itt$ , in Fig. 5, for the cases of increased fan-in, is larger and produces large values earlier.

Figure 2 shows that the measurable mutual information increases earlier as the number of dendrites per neuron is increased. One must take into account that the number of firing is limited due to the refractory period. For a very noisy network the information is low, as well as in a very quiet network. The architecture of 50 dendrites produces information early at the time 30. At time 30, the MI follows the order of the fan-in, and at time 100 it is reversed. Moreover, the maximum value of MI is achieved faster with more dendrites (10, 20 or 30) as expected, but this is not the case for 50 dendrites. These results are robust in all of our experiments. To conclude,

**Fig. 6** A few long-range jumps connects the neurons in the first column to the output column



the addition of additional dendrites may contribute to the total  $It$  and total mutual information to some extent. More dendrites results in a faster increase in  $It$ ,  $Itt$  and mutual information. When too many dendrites exist, the efficiency of the network is deteriorating since less information can be encoded in such a system. In the “network spike” (Eytan and Marom 2006) that occurs when all the network is firing it may be the way that the neurons are resetting before the next event.

(iii) Experiments related to architecture and the “bias” or “priming” hypothesis.

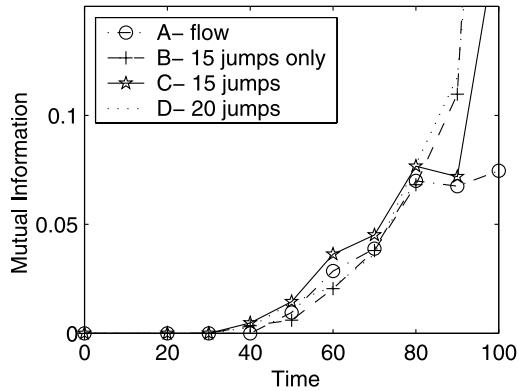
Psychophysical experiments show that certain types of visual information processing, such as face recognition, may be extremely rapid. In a few milliseconds a neuron may yield most of the information about a visual stimulus. This information can be extracted from the response of that neuron Treves et al. (1997). This is so fast, comparable to the speed of feed-forward networks; that a naive view would conclude that there is no time for attractor networks to converge (Fellman and Van Essen 2001; Panzeri et al. 2001).

However, the above work showed that, against intuition, the information flow in attractor networks can reach these speeds. The explanation put forward is that there are many neurons involved, and if they have a statistical bias toward the right solution, sufficient numbers of neurons will in fact reach the states that allow one to extract substantial information quickly.

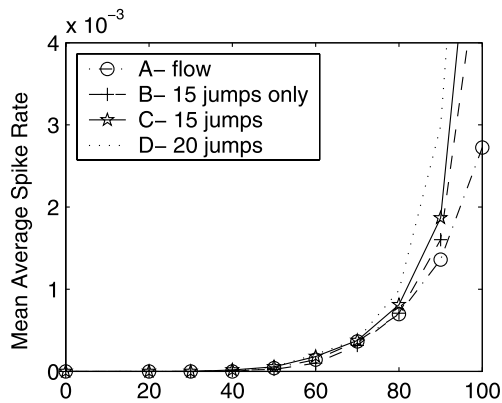
What can cause such a bias? One hypothesis might be that previously, other neurons have sent some spikes to the current column of neurons. In other words, some jumps from the current column are connected more directly to the later column. These should be thought of as early signals that introduce a bias. The percolation of the flow also contributes to the activity in the latter column of neurons. If the input contains information that is consistent with the bias, the system converges faster. If not, it might take more time to reach a different attractor state. The architecture is depicted in Fig. 6.

This idea has some features in common with some suggestions of Eytan and Marom (2006) on the use of hubs in ensembles; which is also put forward as a way to explain the speed of information flow. The short time is explained by the existence of hubs in sub-networks and short connections between the hubs. Marom et al. explain

**Fig. 7** MI of the “jump architecture”



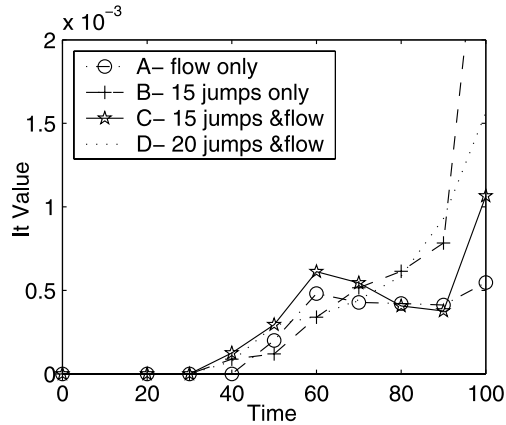
**Fig. 8** Mean average of spikes across trials and stimuli of the “jump architecture”



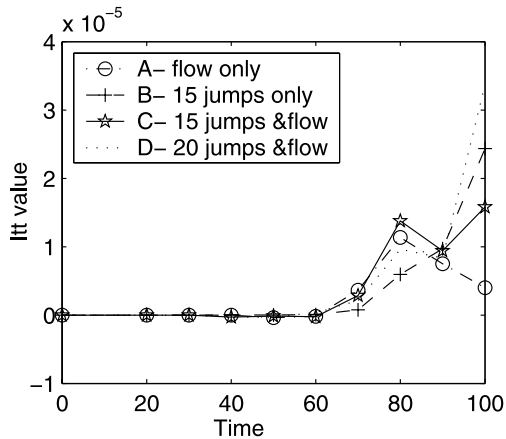
the short time by larger connectivity, in our case the short time is produced by the jumps.

We ran experiments with 0, 15 and 20 long-range connections. The four cases are presented in Fig. 7. The first case is the MI for the flow only without any jumps. Cases 3 and 4 show the mutual information produced for 15 and 20 long-range jumps together with the flow. The second case describes the MI produced for the 15 jumps when the flow of information is blocked. The actual value, in this case, includes also MI that arises from the internal noise in the last column. The long-range connections were added to 30% of the neurons in the first column of  $40 \times 50$  neurons. The mutual information arises earlier with more jumps. This is a relatively small effect, but is consistent over many experiments. 15 long-range jumps produce a somewhat faster rise in MI than 20 long-range jumps similarly to the more connected networks in the previous experiment. As was seen in the connectivity results, the mean firing rate increases by adding these long-range jumps, as opposed to flow-only. See Fig. 8. Adding jumps increase the first derivative as can be seen in Fig. 9. However, when there are too many jumps the trials tend to be similar, thus reducing the amount of information produced by this term. One can also see that the jumps enhance the degree of correlations as in Fig. 10.

**Fig. 9** First derivative ( $I_t$ ) of MI of the “jump architecture”



**Fig. 10** Second derivative ( $I_{tt}$ ) of MI of the “jump architecture”



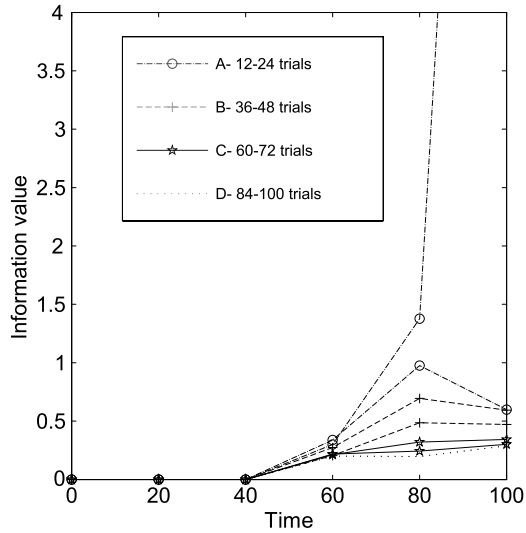
5.1 Precision and architecture

We also discovered that the jump architecture is more precise. To test this, we computed the MI for different numbers of trials. The test presented here shows the MI for 12–100 trials for each stimuli. The difference between the lines is around 12 additional trials for each stimulus. Case A is for 12 trials and case B is for 24 and so on, up to case G which is for 100 trials. When the lines converge, the final MI has been calculated. Figures 11 and 13 show that less trials are needed when an additional 30% long range connections (“jumps”) are added to the architecture of “flow” only.

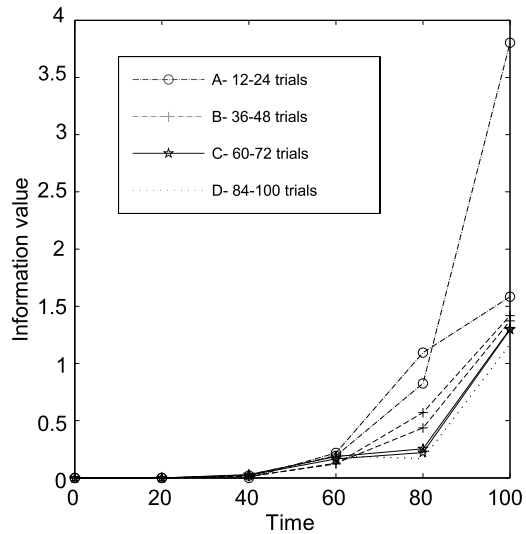
Figure 12 illustrates the case when the flow of information is blocked before the last column, and 30% of the neurons from the first column are attached with jumps to the last column. The information results from the jumps only (and internal noise), and the system is extremely reliable.

Less than 16 trials are enough to compute the correct MI as can be seen in Fig. 12. If one connects *all* the neurons in the first column with long-range jumps, then the architecture is extremely reliable, much more reliable than flow only. Interestingly,

**Fig. 11** MI for flow only for increasing number of trials



**Fig. 12** MI for 15 jumps only for increasing number of trials



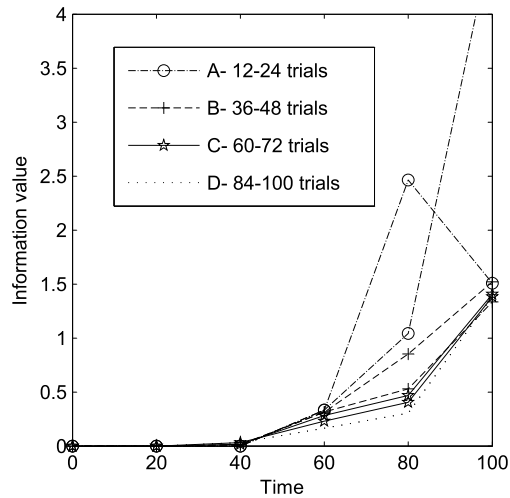
jumps-and-flow is more reliable than jumps only even in this case, although by a somewhat smaller amount.

### 6 Conclusions

We have implemented a tool intended to allow convenient modeling of architectural elements of a large scale cortex. An underlying simplification is in the replacement of differential equations with discretized time as the direct model of the component neurons. We implemented an information measuring tool to monitor the functioning of the models.



**Fig. 13** MI 15 jumps with the percolating flow for increasing number of trials



The novel features of the modeling tool are: (i) The use of a discrete version of integrate-and-fire neurons. This bypasses various computational and numerical analysis issues that arise when using the original Lopicque model and its derivatives. (ii) It is flexible and quite general. The model allows ease of changes in the structure of a particular instance of the model. Thus, for example, one can judge the computational effect of changes in the architecture, reflecting different ideas on the computational importance of anatomical considerations. (iii) Scalability—we have successfully implemented and run models with up to 1 million neurons. The computational facility at our disposal can, in principle, handle up to 50 million neurons on our machine. The model is also directly parallelizable. (iv) The implementation of tools to measure the information flow.

Some new methodologies were also presented to evaluate the credibility of results from our model and to compute information to more than a few neurons. We used the tool to perform simulations that show that some hypotheses are reasonable in such large networks.

The experiments with the tool shows that more dendrites as connections or long-range jumps results in faster arrival of mutual information. This gives an explanation for the bias hypothesis which accounts for the feed forward speed of computation. We showed that it is reasonable in such large networks and surprisingly produces a more efficient system. The architecture with the further jumps increases the speed and the amount of MI with a decrement of the expected error.

## 7 Future work

First, no notion of learning was implemented in the network. However, the design has been specifically chosen to allow such learning (e.g. by a version of Hebbian learning) or by other such methods based on considerations of specific architectures.

Second, it is natural to consider sequences of such networks linked and reacting to the early “guesses” we have isolated, and to verify that successful computational processing can be carried out by such guesses.

Third, the advantage of additional jumps and dendrites is confined to a limited amount of MI, speed and accuracy of computation. The amount of such jumps thus depends on the number of neurons, connectivity and the internal features of such neurons. One may try to assess the optimal jump structure for an architecture.

The tool is now ready for use in more physiologically specific architectures.

**Acknowledgements** This work was conceived when LM was on a sabbatical visit to Edmund Rolls laboratory in Oxford, UK; and an early version of the simulator was written there. LM thanks Prof. Rolls for his hospitality. Leonardo Franco is particularly thanked for explicating the use of information theory in this context and Paul Gabbott is thanked for painstakingly describing various aspects of physiology. This work is part of the M.Sc. thesis of MM at the U. Haifa where she was supported by the Caesarea Rothschild Center and the Neurocomputation Laboratory. MM thanks them, and LM for his guidance.

## References

- Abbott LF (1999) Lapicque’s introduction of the integrate-and-fire model neuron (1907). *Brain Res Bull* 50:303–304
- Bezzi M, Diamond ME, Treves A (2002) Redundancy and synergy arising from pairwise correlations in neuronal ensembles. *J Comp Neurosci* 12:165–174
- Braitenberg V, Schüz A (1998) *Cortex: statistics and geometry of neuronal connectivity*. Springer, Berlin (Revised edition of *Anatomy of the cortex, statistics and geometry*, 1991)
- Eytan D, Marom S (2006) The network spike: a basic mode of synchronization within and between neuronal assemblies. PhD thesis, Faculty of Medicine, Technion
- Fellman DJ, Van Essen DC (2001) Distributed hierarchical processing in the primate visual cortex. *Cereb Cortex* 11:1–47
- Hodgkin AL, Huxley AF (1952) A quantitative description of membrane current and its application to conduction and excitation in nerve. *J Physiol* 117:500–544
- Lapicque L (1907) Recherches quantitatives sur l’excitation lectrique des nerfs traite comme une polarisation. *Physiol Pathol Gen* 9:620–635
- Marom S, Shahaf G (2002) Development, learning and memory in large random networks of cortical neurons: lesson beyond anatomy. *Q Rev Biophys* 35:63–87
- Numerical recipes in Fortran 90 (1996) *The art of scientific parallel computing, scientific computing*. Cambridge University Press, Cambridge
- Panzeri S, Schultz SR, Treves A, Rolls ET (1999) Correlations and encoding of information in the nervous system. *Proc R Soc Lond Ser B: Biol Sci* 266:1001–1012
- Panzeri S, Rolls ET, Battaglia F, Lavis R (2001) Speed of feed forward and recurrent processing in multi-layer networks of integrate-and-fire neurons. *Network* 12(4):423–440
- Rolls ET, Deco G (2002) *Computational neuroscience of vision*. Oxford University Press, Oxford
- Rolls ET, Franco L, Aggelpoulos NC, Reece S (2003) An information theoretic approach to the contributions of the firing rates and the correlations between the firing of neurons. *Neurophysiol* 89:2810–2822
- Schüz A (1998) Neuroanatomy in a computational perspective. In: Arbib MA (ed) *Handbook of brain theory and neural networks*. MIT Press, Cambridge
- The genesis simulator (1994–2006) <http://www.genesis-sim.org/GENESIS>
- Trappenberg TP (2002) *Fundamental of Computational Neuroscience*. Oxford University Press, Oxford
- Treves A, Rolls ET, Simmen M (1997) Time for retrieval in recurrent associative memories. *Physica D* 107:392–400
- Wolfram S (2002) *A new kind of science*. Wolfram Media, Inc