HAL open science

ENAS: A new software for spike train analysis and simulation

Bruno Cessac, Pierre Kornprobst, S Kraria, H Nasser, Daniela Pamplona, Geoffrey Portelli, T Viéville

To cite this version:

Bruno Cessac, Pierre Kornprobst, S Kraria, H Nasser, Daniela Pamplona, et al.. ENAS: A new software for spike train analysis and simulation. [Research Report] RR-8958, Inria Sophia Antipolis; Inria Bordeaux Sud-Ouest. 2016. hal-01377307v1

HAL Id: hal-01377307

https://inria.hal.science/hal-01377307v1

Submitted on 6 Oct 2016 (v1), last revised 24 Aug 2017 (v2)

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.

ENAS: A new software for spike train analysis and simulation

B. Cessac, P. Kornprobst, S. Kraria, H. Nasser, D. Pamplona, G. Portelli, T. Viéville

Authors are presented using alphabetic order

RESEARCH REPORT

N° 8958

October 2016

Project-Teams Biovision and Mnemosyne

ISSN 0249-6399 ISRN INRIA/RR--8958--FR+ENG

ENAS: A new software for spike train analysis and simulation

B. Cessac*, P. Kornprobst*, S. Kraria*, H. Nasser*, D. Pamplona*, G. Portelli*, T. Viéville†

Authors are presented using alphabetic order

Project-Teams Biovision and Mnemosyne

Research Report n° 8958 — October 2016 — 21 pages

* Université Côte d'Azur, Inria, Biovision team, France
† Inria, Mnemosyne project team, Bordeaux, France

RESEARCH CENTRE
SOPHIA ANTIPOLIS – MÉDITERRANÉE

2004 route des Lucioles - BP 93
06902 Sophia Antipolis Cedex

Abstract: As one gains more intuitions and results on the importance of concerted activity in spike trains, models are developed to extract potential canonical principles underlying spike coding. These methods shed a new light on spike train dynamics. However, they require time and expertise to be implemented efficiently, making them hard to use in a daily basis by neuroscientists or modelers. To bridge this gap, we developed the license free multiplatform software ENAS (https://enas.inria.fr) integrating tools for individual and collective spike analysis and simulation, with some specificities devoted to the retina. It contains a toolbox allowing to handle Multi-Electrode Array (MEA) data: MEA grid geometry, spike analysis, and simultaneous visualisation of the stimulus. It allows also to compute receptive fields via Spike-Triggered Average (STA) and to represent them in 3 dimension (space and time). The core of ENAS is the statistical analysis of population codes. One of its main strength is to provide statistical analysis of spike trains using Maximum Entropy-Gibbs distributions taking into account both *spatial and temporal correlations* as constraints, allowing to introduce causality and memory in statistics. Finally, ENAS also generates simulated spike trains. On one hand, one can draw a population raster from an user-specified Gibbs distribution. On the other hand, we have integrated in ENAS our retina simulator Virtual Retina [53], extended here to include lateral connections in the IPL. The methods used in this software have already been applied and validated to the analysis of retina data [52, 17, 18]. All these tools are accessible through a friendly Graphical User Interface that avoids any scripting or writing code from user. Most of them have been implemented to run in parallel to reduce the time and memory consumption. We hope that ENAS will become a useful tool for neuroscientists to analyse spike trains and we hope to improve it thanks to user feedback. Our goal is to progressively enrich it with the latest research results, in order to facilitate transfer of new methods to the community.

Key-words:
Multi-electrode array recordings, spike train statistics, population coding, maximum entropy, Gibbs distributions, large scale spiking activity, spike train generation, retina simulator, Virtual Retina

ENAS: Un nouveau logiciel pour l'analyse et la simulation de train de potentiels d'action

Résumé : Alors que nous avons plus d'intuitions et de résultats sur l'importance de l'activité concertée dans les trains d'impulsion, des modèles sont conçus pour extraire les principes canoniques potentiels sous-jacent du code impulsionnel. Ces méthodes apportent un nouvel éclairage sur la dynamique des trains d'impulsion. Cependant, elles necessitent du temps et de l'expertise pour etre implémentées de manière efficace, ce qui les rend difficiles à utiliser par des neuroscientifiques ou des modélisateurs. Pour combler cette lacune, nous avons développé le logiciel libre multiplateforme ENAS (https://enas.inria.fr) intégrant des outils pour l'analyse individuelle ou en population des trains d'impulsion, avec des spécificités liées à la rétine. Il contient une boite à outils permettant de traiter des données venant de grille d'électrodes (*Multi-Electrode Arrays*, MEA): géométrie de la grille d'électrodes, analyse de ces trains d'impulsions et visualisation simultannée avec le stimulus. Il permet aussi d'estimer les champs recepteurs avec la méthode de *Spike-Triggered Average* (STA) et des les représenter en 3 dimensions (espace et temps). Le cœur d'ENAS est l'analyse statistique des codes en populations. L'une des principale force d'ENAS est de fournir une analyse statistique avec des distributions de type Maximum Entropy-Gibbs en tenant compte des corrélations spatiales et temporelles comme contraintes, ce qui permet d'introduire de la causalité et de la mémoire dans les statistiques. Finallement, ENAS permet également de générer des trasin d'impusion simulés. D'une part, on peut générer des trains d'impulsion à partir d'une distribution de Gibbs définie par l'utilisateur. D'autre part, nous avons intégré dans ENAS le simulateur de rétine Virtual Retina [53] que nous avons étendu en ajoutant la possiblité d'inclure des connexions latérales dans l'IPL. Les méthodes dévelopées dans ce logiciel ont déjà été appliqués à l'analyse des données de rétine [52, 17, 18]. Elles sont accessibles via une interface utilisateur graphique conviviale qui évite aux utilisateurs d'écrire des scripts ou du code. La plupart des ces méthodes ont été implémentées pour fonctionner avec des architectures parallèles afin de réduire le temps et la consommation de mémoire. Nous espérons qu'ENAS deviendra un outil utile pour les neuroscientifiques leur permettant d'analyser des trains d'impulsion, et nous espérons l'améliorer grâce aux retours des utilisateurs. Notre objectif est de l'enrichir progressivement avec les derniers résultats de recherche, en vue de faciliter le transfert de nouvelles méthodes pour la communauté.

Mots-clés :

Multi-electrode array recordings, spike train statistics, population coding, maximum entropy, Gibbs distributions, large scale spiking activity, spike train generation, retina simulator, Virtual Retina

1 Introduction

Despite it has passed more than one century since Cajal's neuron doctrine, how information is encoded and transmitted by neurons is not fully understood. Especially, the role of spatio-temporal correlations in population coding raises up deep theoretical and practical questions far from being answered [35, 4].

With the advent of new techniques, the recording of the simultaneous activity of groups of neurons provides a critical database to unravel the role of specific neural assemblies in spike coding. The acquisition capacity of Multi-Electrode Arrays (MEA) has been exponentially increasing over years [44]. Systems like the 256–MEA has become a standard although new high density MEA are now available such as the APS CMOS 4096–electrodes [12]. Using these systems, one can record from hundreds to thousands of neurons simultaneously, in vivo or in vitro, from cultures of neurons, in the retina or in other brain areas. In parallel, biologically inspired spiking neurons simulators have been developed such as [32, 53, 22, 24], generating large spike trains too.

As one gains more intuitions and results on the importance of concerted activity in spike trains, models are developed to extract, from real recordings, potential canonical principles underlying spike coding. This is an important challenge with many potential outcomes. Beyond the dream that population coding is ruled by a few first principles to be discovered in a near future, several potential applications could emerge such as the development of artificial systems having similar levels of performances as biological systems. This requires anyway accurate tools to analyze and compare spike trains, real or artificial ones.

Progress has been made recently to analyze spike trains [39, 40, 23, 48, 47, 38, 37, 36]. Especially, the spatio-temporal aspects (memory) and causality have been shown to be relevant for exploring neural activity [4]. For instance, [52] and [46] demonstrated the importance of temporal statistics and [30, 29] developed new tools to analyze spatio-temporal activity for large scale spiking networks. These methods shed a new light on spike train analysis.

However, they require time and expertise to be implemented efficiently, making them hard to use. The idea of developing a new software came from our motivation to share these recent developments with the neuroscience community in a broad sense. So we developed ENAS focusing on the usability of the software without the need of programming language knowledge. ENAS was initially devoted to spike trains statistics analysis. But, as we were developing and using this tool in close collaborations with neurobiologists working on the retina [51, 17, 18] (see acknowledgment section), we have designed specific tools dedicated to retina data analysis. At some point, we also made the decision to merge ENAS with the Virtual Retina simulator developed at Inria [53]. We have therefore progressively designed a user-friendly tool dedicated to neuroscientist community, presented in this paper.

The paper is organized as follows. In Sec. 2, we give an overview of ENAS and compare it with a selection of other tools for spike train analysis to highlight its main strengths. In Sec. 3, we present the main features of the software. Illustrations and step by step procedure are given in several cases allowing readers to reproduce them. In Sec. 4, we discuss future developments.

2 General presentation

ENAS targets a wide community of scientists interested in exploring spike trains. It provides new tools for analyzing spike trains at population level and several methods to generate them, either with a prescribed statistics (including spatio-temporal correlations) or by emulating the retinal response to visual scene. This is done through a friendly Graphical User Interface (GUI). ENAS is freeware and it can be download from *https://enas.inria.fr* where documentation, tutorial and

samples of spike trains are available. It runs on multiple platforms (Linux, Mac OS, Windows) and it supports parallel architectures.

The GUI of ENAS (version 1.0.0) has three main panels, as shown in Fig. 1:

- The function panel (FP) provides processings grouped in three sections: `Data`, `Analysis` and `Simulation` (see Sect. 3 for more details). Note that in the `Help` menu from the top, user can activate the `Show bubble help` giving information for each function.
- The parameter panel (PP) allows one to set parameters related to the chosen function from the function panel.
- The results panel (RP) contains the results. Several results can be displayed simultaneously in different subpanels and each result can be exported as a figure in a variety of formats (e.g., PDF, PNG, JPG). User can change the `Number of views` and which result should be displayed in each subpanel. In most subpanels, there are icons one the left-hand side to open, save, export or change plot settings (see Fig. 1(c)). Another example is shown in Fig. 4(a) (`Stimulus view`) where user can export, show/hide grid and spikes and select neurons graphically.

Finally, another convenient feature of the software is that user can save what has done in term of analysis in a HDF5 file [54] (`File>Save`) and load it later to continue the work.

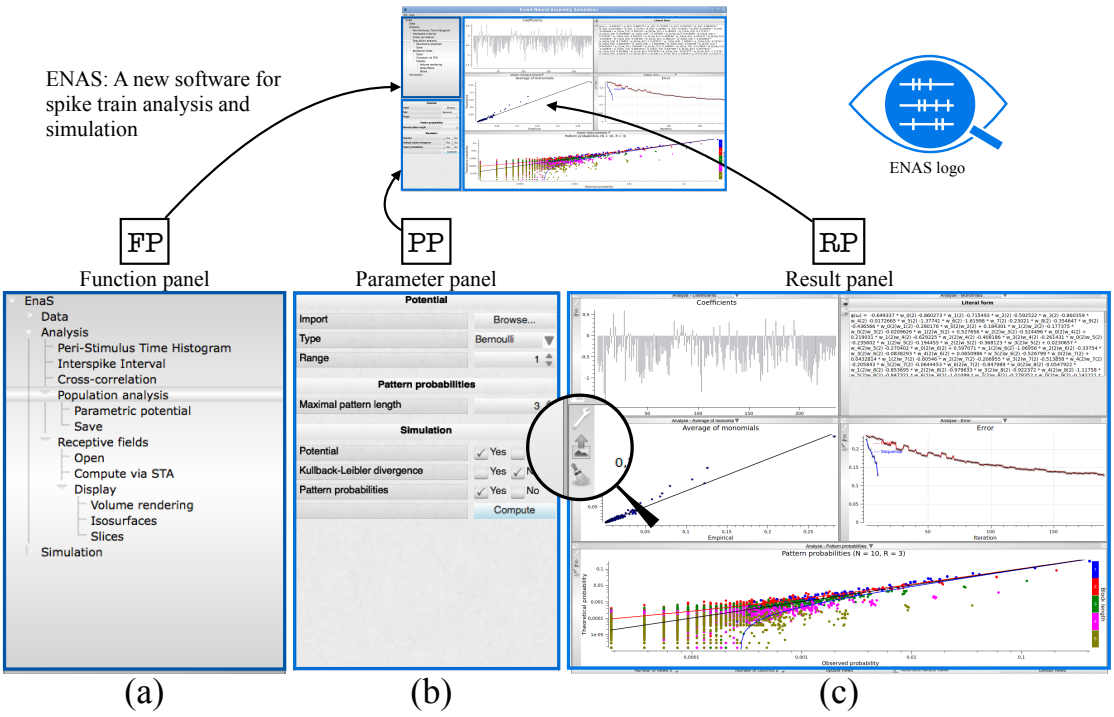

Figure 1: Presentation of ENAS GUI panels. (a) Function panel (FP) is the main panel where user chooses which processings to make. (b) Depending on the processing selected, the parameter panel (PP) gives the opportunity to set all parameters. (c) Results panel (RP) can show all results user want. The number of sub-panels and the windows to show can be changed. (d) For each result shown in a sub-panel, user can customize the display or export it in different formats.

ENAS is implemented in C++. It has its own dedicated libraries. It also uses license free libraries for storage (HDF5), analysis (GSL, SFMT), display and GUI (Qt4, Qwt, VTK, CImg). Some parts of the toolbox could be run on multiple processors thanks to OpenMP framework. Parallelization allows to boost heavy computations and save processing time and memory. Thanks to OpenMP, ENAS parallel processes run on a personal computer equipped with multi-core processors as well as on clusters.

There are several software or libraries for spike train analysis. Table 1 provides a qualitative comparison between ENAS features and a selection of existing ones. As can be seen from this comparison, up to our knowledge, ENAS is the only one allowing spatio-temporal Maximum Entropy analysis, joint spike/stimulus visualization, receptive field computation and synthetic raster generation using either statistics. It also includes a virtual retina simulator. One has to note that the implementation in C++/Qt4 makes this tool available without any license contrarily to, e.g., the FIND toolbox, one of the most ambitious project based on community contribution, which needs a Matlab license, for instance.

	Chronux	FIND	nSTAT	sigTOOL	NeuroPy	OpenElectrophy	STAR	NeuroExplorer	ENAS
Language	Matlab	Matlab	Matlab	Matlab	Python	Python	R	VB	C++
Reference	[19]	[33]	[15]	[21]	[43]	[13]	[27]		
GUI	●	●		●		●		●	●
Scripting	●	●	●	●	●	●	●	●	
Free binary	●	●	●	●	●	●	●		●
Files formats (Nex)		●		●		●		●	●
Files formats (HDF5)		●							●
Rate histogram PSTH				●		●		●	●
Population PSTH	●	●		●				●	●
ISI	●	●		●			●	●	●
Cross-correlograms	●	●		●			●	●	●
Joint spike and stimulus visu.									●
STA					●			●	●
PCA								●	
Spectral analysis	●							●	
MaxtEnt (x / x-t)			GLM						●
Raster generation									●
... with Poisson / Non Poisson									●
... with Virtual Retina									●

Table 1: ENAS compared to the state-of-the-art. Note that features selected in this table have essentially been chosen according to what ENAS does. Software we mention can have additional features not commented herein.

3 ENAS main functions

3.1 Data

In section `Data`, user can dowload spikes. Spikes may come either from real cells recordings or from simulated spiking neural network. They can be imported from different formats such as simple text file (`.txt`), DAT file for Windows (`.dat`), NEXUS file (`.nex`) or HDF5 file format (`.hdf5`, see [54]) allowing to store not only the spikes but also other information related to experimental protocole (e.g., time stamps, MEA characteristics).

Depending on how the spikes were generated, user can also download other information such as:

- The MEA configuration for real cell recordings, allowing the spiking activity to be displayed relative to the grid of electrode.
- The sequence of images in case of a visual experiment, allowing the spiking activity to be displayed together with the stimulus but also to emulate a retinal response (see Sect. 3.3.2).
- The time stamps in case of a visual real experiment, i.e., the precise time at which each image was presented, allowing to perform Spike-Triggered-Average (STA) [7].

3.2 Analysis

3.2.1 Classical analysis

Once the spikes are imported, different analysis can be performed starting with basic visualizations and classical analysis such as peri-stimulus histogram, interspike interval and cross-correlation (see Fig. 2 and Example 1). The number of neurons used for these analysis can be of several thousands, although user can select subsets of neurons according to different criteria, e.g., individual activity or receptive field localization (if applicable, i.e., if data allows to compute receptive fields).

Example 1 How to obtain the peri-stimulus time histogram (PSTH) on a selection of neurons having the highest spike rates?

1. FP `Data>Spikes>Open`
2. PP Browse and select the file of spikes.
3. RP In the sub-panel showing the list of neurons, click on `Spike rate` to sort neurons with respect to their averaged spike rate, select the set of neurons with higher spike rate and click on `Keep selected`.
4. FP `Analysis>Peri-stimulus time histogram`.
5. PP Choose the `bin size` (time step) in milliseconds and click on `Compute`.

3.2.2 Population analysis

With the current evolution of multi-electrode recordings, it is possible to record from tens to thousands of neurons [12], opening up the possibility of analysing the correlations at the population level. The structure of these correlations is imposed, on one hand, by the collective dynamics shaped by individual neurons characteristics and connectivity, and, on the other hand, by the entries the network receives (e.g., stimulus for sensory organs). Characterizing spatio-temporal correlations is a fundamental step toward extracting generic population coding principles in neuronal networks. In particular, there are more and more evidences that Poisson modelling is somewhat a bit too rough. For example, it has been observed in the concerted activity of retinal ganglion cells, in several vertebrate species,[14, 16, 39, 41]), that, even if (pairwise) correlations are *weak*, they are not *negligible* and they are necessary to explain the statistics of spikes [39, 40].

Analyzing spatio-temporal correlations is however a notoriously hard problem from a statistical perspective. The point is not only to measure pairwise or higher order correlations. It is to address properly the question of their significance in neuronal coding taking into account such constraints as finite statistical sample, reliablity of statistical tests, hypotheses underlying

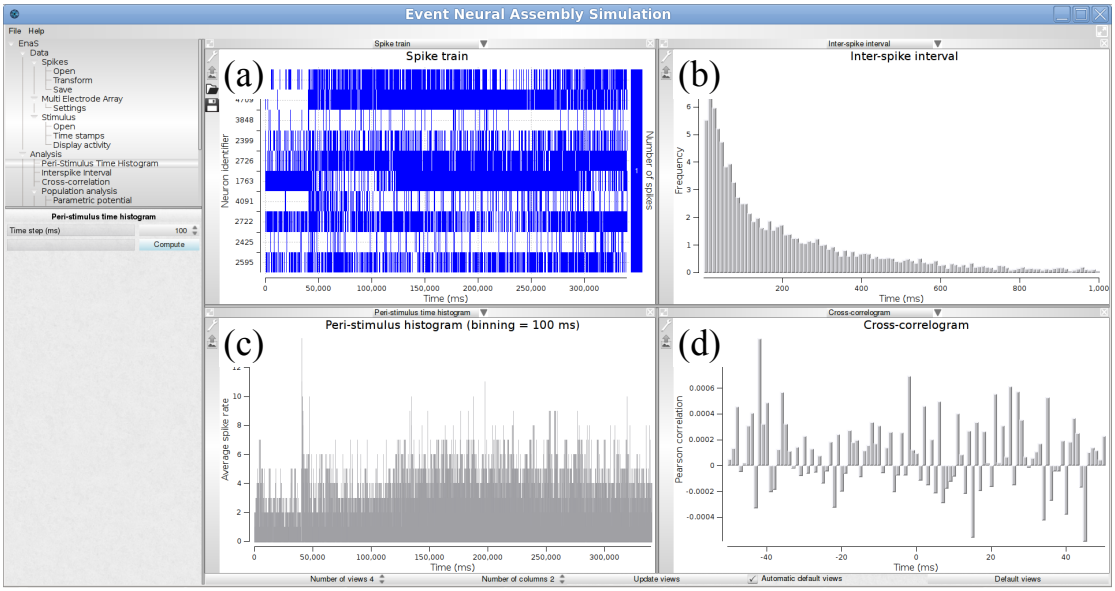

Figure 2: Example of classical analysis for a spike train shown in sub-panel (a). The following estimations are shown: (b) Inter-spike interval (ISI), (b) Peri-stimulus time histogram (PSTH) and (c) cross-correlogram

statistical models, risks of overfitting. We have proposed a framework to handle this problem using Gibbs distributions [5, 51, 8, 31, 30, 29, 9]. This is a general class of probability distributions, initially introduced in statistical physics by Boltzmann and Gibbs [20] but currently used in a more general context, especially neural population analysis. They constitute a canonical paradigm to explain the spatio-temporal statistics reproducing as close as possible empirical correlations, without adding additional unnecessary assumptions. Although most studies in neural population analysis are based on instantaneous pairwise correlations they can be extended to general spatio-temporal correlations at the algorithmic level [52, 31, 29].

This is what proposes ENAS: a method to compute a probability distribution from data, reproducing as close as possible empirical statistics, without adding additional unecessary assumptions. ENAS has already been used for the analysis of retina data in [52, 17, 34]. More generally, ENAS attempts to numerically answer the following problem: constructing a probability distribution μ approaching at best the empirical data, and especially reproducing spatial as well as *time* correlations. This is a prominent characteristic of ENAS. In its present version 1.0.0, ENAS focuses on *stationary distributions, i.e., the statistics is invariant under time-translation.* Extension to non stationary probability distributions are in progress.

The basic object defining the probability also called "potential" (or energy function) is a linear combination of terms called *monomials*, each of them corresponding to specific events in the raster. These events have the general form *"neuron i_{k_1} is firing at time t_{k_1}", ..., "neuron i_{k_n} is firing at time t_{k_n}"* , for some integer n. For example, $n = 1$ corresponds to the event "one neuron is spiking at a given time", $n = 2$, "two neurons are spiking at two (possibly distinct) times", and so on. The set of these events defines the statistical-model type. The simplest case corresponds to the so-called Bernoulli model: the events are of type "neuron i is firing at time t". The goal is to find a probability μ which gives to the event "neuron i is firing at time t" the same value as the empirical probability of this event. Modulo these constraints, μ must be

"as random as" possible (maximizing entropy). This is a convex problem and there is a unique answer: μ is here a Bernoulli distribution [4] where the probability that a neuron fires is the empirical firing rate. As we assumed stationarity, rates do not depend on time.

More generally, the times t_k defining a monomial are chosen in the interval $[0, D]$, where $D \geq 0$ is an integer called *memory depth*. The number $R = D + 1$ is called the model *range*.

Obviously, as there is an exponential number of monomials there are pre-defined model choices, although user can also define his own model (the procedure is explained below). The main predefined models are:

- `Bernoulli model`. It takes into account individual neurons activity where all neurons are independent (see [39] for an example). This model has range 1.
- `Ising model`. This paradigmatic model from statistical physics has been used in neuroscience in several papers such as [39, 40, 49, 50]. The events are of type "neuron i is firing at time t" and "neuron i and neuron j are simultaneously firing at time t". Thus events are instantaneous and this model does not involve memory and causality (successive times are independent under μ). This model has range 1.
- `Pairwise model`. This is an extension of Ising model where monomials are of type "neuron i is firing at time t" (single events) and "neuron i is firing at time t and neuron j is firing at time $t + k$", $0 \leq k \leq D$ (pairwise events). This model integrates memory and causality via time dependent pairwise interactions.
- `User`. Here the user defines his own potential of the general form:

$$\mathcal{H} = \sum_{l}^{L} h_l m_l(\omega), \tag{1}$$

  where each m_l corresponds to a monomial. For the user defined potential, ENAS allows to load a text file containing the potential. The written potential has to fit with ENAS language, which corresponds to:

  (i) A potential is represented by the addition of a finite number of monomials;

  (ii) A spike event is represented by the character chain $w_i(j)$. i corresponds to the neuron index and j for the spiking time;

  (iii) A monomial is represented by the multiplication or several single spike events.

  For example, the potential $w_0(1)+w_3(0)w_4(0)+w_2(0)+w_3(0)w_2(0)w_0(1)$, is represented by four monomials. The first, second, third and fourth monomials are respectively represented by one event (single neuron activity), two events (pairwise), one event and finally 3 events (triplets with memory). The parameters h_ls are self-tuned by the software to approach as best the empirical distribution. One can either start from $h_l = 0$ or from a fixed one. This last case is especially useful when one wants to converge to the optimal solutions in several steps. Indeed, for a given set of h_ls the method generates a spike train of length T distributed according to the corresponding Gibbs distribution, computes the empirical average of m_l from this raster and then tunes h_ls to minimize the distance between theoretical and empirical average. As the method requires usually many steps to converge, one has to recompute the raster at (almost[1]) each step. Since empirical averages are subject

[1] In fact, when the error is small the updated average are computed using a linear response formula, avoiding the recomputation of the raster [29].

to (Central Limit) errors with amplitude $\propto \frac{1}{\sqrt{T}}$ it is better to start with small Ts in the beginning of the computation and to increase it as error decreases. The algorithm is described in details in [29]. At each step, one uses the previously computed values of h_ls to start with.

Example 2 shows how to compute potential and μ of a spike train.

Example 2 How to calculate the potential and μ of a spike train?

1. FP `Data>Spikes>Open`
2. PP Browse and select the file of spikes.
3. RP In the sub-panel showing the list of neurons, select a subset of neurons if wanted and click on `Keep selected`.
4. FP `Analysis>Population analysis`.
5. PP Define the `Potential`, either from a file or by setting the type (e.g., `pairwise`) and the range (e.g., R=3). The product $N \times R$ somewhat defines the complexity of the model. The larger $N \times R$ the longer the simulation and the memory load. We have been able to run up to $N = 100$, $R = 2$ in reasonable times. Note however that the statistical estimation of a so-huge number of neurons raises the classical problems (*not inherent to our method*) of statistical estimation on a sample, which, for the retina rarely exceeds a few millions of spikes. Set the `Maximal pattern length`, i.e., the time-length of the spike blocks you want to test the probabilities (e.g., 5). In `Simulation`, choose what you want to compute. Possible choices are:
   (i) `Potential`: Fits the coefficients of the potential.
   (ii) `Kullback-Leibler divergence`: an estimation of Kullback-Leibler divergence derived from the entropy estimation in [45] and the classical relations between K-L divergence, μ average of $\mathcal{H}$ and entropy [4].
   (iii) `Pattern probabilities`: The confidence plot.

   Note that you can select only `Potential` for the first steps, to speed up computations.
6. FP `Analysis>Population analysis>Parametric potential`
7. PP Choose `Initial coefficients`: possible choices are `Current` (in the case when you have made a run before and you want to keep h_ls) or `0`. For the first run you can choose both as the initial parameters are anyway set to 0 (this corresponds to take a Bernoulli models where spikes are independent with probability of occurrence $\frac{1}{2}$). Set parameters of the Monte-Carlo method to speed up the process (see text for the role of each parameter).
8. FP `Population analysis`
9. RP `Compute` to do the estimation. A red bar at the bottom indicates that computation is in progress.

Note that in ENAS user can fine-tune the probability estimation and speed up the computation in the first steps of the Monte-Carlo method by acting on the following parameters (in FP `Analysis>Population analysis>Parametric potential`):

- `Convergence bound`: The lower bound of error you require. Simulations stops when the code reaches this value.

- `Tolerance Coefficient`: Allows to filter monomials to compute h_ls. For example, if the event m_l appears only, say 3 times in the raster you may consider that it is not significant and must be eliminated from the potential (1). In this case you set `Tolerance Coefficient` to 3.
- `Raster length`: The length of the Monte Carlo raster used to compute the average of monomials for the current values of h_ls. This number must increase as you get closer to the solution (typically when you don't get any improvement in the error).
- `Spike flips coefficients`: This defines the number of flips per neuron and per iteration in Monte Carlo methods [29].
- `Delta`: If the distance between predicted and empirical distribution is smaller than `Delta` the Monte Carlo raster is not recomputed. Instead the average value of monomials is computed via linear response [29]. The number must be decreased as you get closer to the solution (typically when you don't get any improvement in the error).
- `Number of parallel iterations`: We use two kinds of updating, based upon [11]: parallel or sequential. Parallel update sets all h_ls in one step. `Number of parallel iterations` tells how many parallel updating of h_ls are done before the program stops (and e.g. plots the confidence plot, when selected).
- `Number of sequential iterations`: Sequential update. It computes only one h_l at each step. It is better to use it at the end of the computation, to fine-tune coefficients.

Figure 3 shows different visualizations of the results. In (a), we show a graphical view of the potential's coefficients. In (b), we show the literal form of the potential $\phi(\omega)$. In this example, it is

$$\begin{aligned}\phi(\omega) = &- 0.64937\ w_0(2) - 0.86073\ w_1(2) + \cdots - 0.436566\ w_0(2)w_1(2) \\ &+ \cdots - 0.120281\ w_0(2)w_3(2)w_5(3) + \ldots,\end{aligned} \tag{2}$$

where, e.g., $w_0(2)w_1(2)$ corresponds to the event "neuron 0 is firing at time 2 and neuron 1 is firing at time 2, whereas -0.436566 is the corresponding coefficient of this term in the potential. In (c), we show a plot of events (monomials) average value, with empirical average on abscissa and theoretical average (predicted by μ) on ordinate. In (d), we show the evolution of the Hellinger distance between the model predictions and empirical data, versus the number of iterations. Finally, in (e) we show a *confidence plot*. This confidence plot consists of plotting, in log scale, in abscissa the empirical probability of observed spike blocks and in ordinate the expected probability of those blocks for the model μ. If the matching were perfect ($\mu = \pi$) one should have all points aligning on the diagonal. This would be true, however, only if the spike train were infinite. For finite spike trains fluctuations about the exact probability are observed, ruled by Central Limit Theorem. Thus, we draw, around the diagonal error bounds corresponding to 3σ where σ is an estimation of the Central Limit standard deviation. This defines a confidence region: if μ was a perfect estimation of π then points in the confidence plot would be distributed inside the confidence region with a probability of 99.7% of the expected averages.

3.2.3 Receptive fields estimation

Beyond tools to analyze spike trains at single cell and population levels, ENAS provides specific tools in the case when spike trains come from visual stimulation experiments. One example is

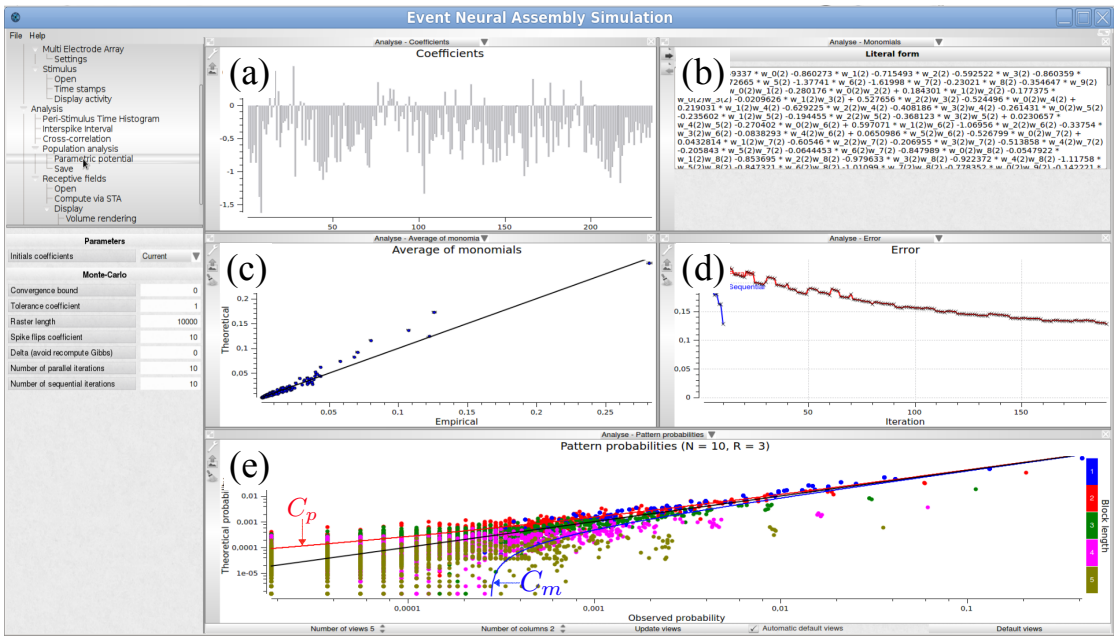

Figure 3: Population analysis view with $N = 10, R = 2$. (a) Graphic view of the potential's coefficients values; the abscissa corresponds to an index associated to a prescribed order on monomials: the N first coefficients corresponds to single events (hence are related to firing rate); by clicking on the histogram bar one makes appear the coefficient value and the corresponding monomial. Ordinate give the value of the coefficient attached to the corresponding monomial. (b) Literal form of the potential. (c) Theoretical average value of monomials versus their empirical average value. To which extents the model predicts the empirical data is drawn with this representation. This allows checking that constraints on empirical averages are respected by the model. (d) Decay of estimation errors versus the number of iterations allowing us to graphically check the convergence of the estimation algorithm (red curve). In the figure several successive runs have been performed. The blue curve shows the evolution of error after each of these trials. (e) Confidence plot. C_p, C_m corresponds respectively to the upper and lower 3σ bound associated with Central Limit Theorem (see text).

retinal recordings with photo-stimulation protocole. User can upload MEA characteristics, the sequences of images of the stimulus and the time stamps. Given this information, ENAS provides a method to estimate receptive fields which is a classical step to make in retinal experiments. The method available in version 1.0.0 is Spike Triggered Average (STA) [6]. STA can be computed for a single neuron, a subset of neurons or the entire population. Example 3 describes the general procedure.

Example 3 How to estimate receptive fields with STA method?[2]

1. FP `Data>Spikes>Open`
2. PP Browse and select the file of spikes. Note that user can select a subset of neurons as in Example 1.
3. FP `Data>Multi Electrode Array>Settings`
4. PP Define the MEA configuration, so that, if neurons have a position relative to the electrodes, spike activity can be displayed at the correct location.
5. RP `Stimulus view` shows the activity of neurons w.r.t. MEA array. User may choose a region of interest to select a subset of neurons Note that if you have a TEXT file for spikes, no position is available, and we chose to arrange neurons following their order, by rows and columns.
6. FP `Data>Stimulus>Open`
7. PP Browse to choose the directory containing the sequence of white noise images. Note that all images should be in this directory and ordered by name. All standard formats are accepted (e.g., PNG, JPG, TIFF).
8. FP `Data>Stimulus>Time stamps`: Time stamps correspond to the exact presentation time of each image during the experiment.
9. PP Browse and select the file. Time stamps can be imported from a `.txt` (row n contains the starting time of image n) or HDF5 [54].
10. FP `Analysis>Receptive fields>Compute via STA`
11. RP `Output directory`: Choose the directory where the receptive field will be saved, as a sequence of PNG files and in HDF5 format. Choose the number of images to be averaged before each spike with `Time depth (number of frames)`. Then click on `Compute`.
12. FP `Analysis>Receptive fields>Display` allows four types of visualization.
13. RP User can display the receptive field of interest by selecting the neuron in `Receptive Felds` subpanel and `Update views`.

The receptive field we obtain is a spatio-temporal volume. Several visualizations (2D and 3D) are proposed, as illustrated in Fig. 4.

[2]To test this example, one can use sample data available on ENAS website: recording called mouse-P39_17_06_14 (We are indebted to Evelyne Sernagor and Gerrit Hilgen, Newcastle University, who provided us these data and authorized us to use them as a basis for examples.)

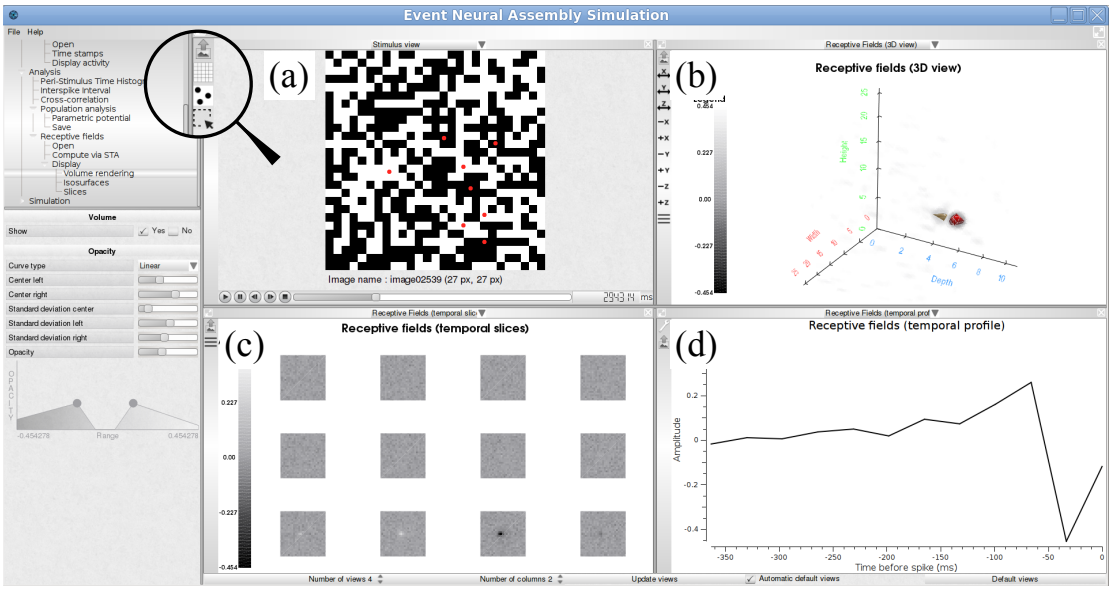

Figure 4: Receptive field estimation using the STA approach. (a) White noise image sequence with spiking activity superimposed as red dots. (b) 3D volume visualization of the receptive field of one neuron. (c) Temporal slices of the receptive field. (d) Temporal profile passing through the peak activity location of the spatio-temporal volume.

3.3 Simulation

3.3.1 From statistics

ENAS gives the possibility to generate artificial spike trains with a known probability distributions. There are two possibilities:

1. FP `Simulation>ComputeSpikes>GibbsRaster>current`: if distribution comes from a population analysis (see Sec. 3.2.2).

2. FP `Simulation>ComputeSpikes>GibbsRaster>File`: if distribution has been defined by the user and stored in a file containing the Gibbs potential form.

The probability distributions are indeed Gibbs probabilities. A Gibbs probability distribution (defined by Gibbs potential of the form (1)) is naturally associated with a Markov chain, with memory $D = R - 1$, whose transition probabilities can be computed from $\mathcal{H}$ [52, 4, 9]. The invariant distribution μ of this chain is the Gibbs distribution associated to $\mathcal{H}$. Therefore, given a potential of the form (1) it is easy to generate a sample raster distributed according to μ using Monte Carlo method [30]. ENAS affords this functionality allowing to generate rasters with spatio-temporal correlations tuned by the parameters h_l in $\mathcal{H}$.

3.3.2 The retina simulator

Another strength of ENAS is to provide a way to generate spike trains using a virtual retina simulator. To do so, ENAS has integrated and extended the open source simulator called Virtual Retina [53]. Virtual Retina allows large-scale simulations of biologically-plausible

retinas, with customizable parameters. It has been shown to reproduce a wide range of experimental data at single cell level, from salamander, cat and primate retinas, and has been used in several theoretical studies [25, 26, 28, 1]. The underlying model includes a non-separable spatio-temporal linear model of filtering in the Outer Plexiform Layer, a shunting feedback at the level of bipolar cells, and a spike generation process using an network generalization of the noisy leaky integrate-and-fire neurons to model RGCs.

The Inner Plexiform Layer (IPL) in the initial version of Virtual Retina was rudimentary. Ganglion cells (GCells), modelled by a Leaky-Integrate and Fire model, were receiving a bipolar current, but IPL neither had amacrine cells (ACells) nor lateral connectivity (we mean here the connectivity between GCells due to intermediate ACells). As a consequence correlation in GCells spikes were only due to shared stimulus. We wanted to implement an effective lateral connectivity in the new version so as to explore its effects on retinal responses to stimuli. So, although, the new version of Virtual Retina does not implement yet a thorough modelling of ACells and GCells, we have introduced the possibility to connect the GCells.

More precisely, the IPL is now composed by discrete time Leaky-Integrate and Fire neurons introduced in [42] and studied mathematically in [2, 3]. We made this choice of because this model has a fast response and its collective dynamics is well known. In a nutshell, the model comes from the time discretisation of the Leaky-Integrate and Fire model whose sub-threshold dynamics reads:

$$C\frac{dV}{dt} = -g_L(V - V_L) + I(t) + \sigma_B B(t), \tag{3}$$

for $V < \theta$, where θ is the firing threshold. Here C is the membrane capacity, g_L the leak conductance, V_L is the leak reversal potential, $B(t)$ is a Gaussian noise (mean zero and variance 1) and σ_B controls the amplitude of this noise. Finally, $I(t)$ is an external current. In our case, this is the bipolar current coming from the OPL (see [53] for details).

Let $\tau_L = \frac{C}{g_L}$ be the leak characteristic time. We consider now a time discretisation with a time step dt (typically 1 ms) and we set $\gamma = \left[1 - \frac{dt}{\tau_L}\right]$. Note that dt has to be quite smaller than τ_L in order to have a reasonable description of the biophysics. Therefore $\gamma \in [0, 1]$. Then, the dynamics of the membrane potential V_i of each neuron i is given by:

$$V_i(t+1) = \gamma V_i(t)(1 - Z_i) + \frac{I_i(t)}{C} + \frac{\sigma_B}{C} B(t),$$

where Z is the binary spike label

$$Z_i = \begin{cases} 1 & \text{if } V_i \geq \theta \\ 0 & \text{otherwise,} \end{cases}$$

which is 1 whenever neuron i spikes. In this equation neurons are not coupled. We add then a connectivity through synaptic weights W_{ij} where neuron j (pre synaptic) acts on neuron i (post synaptic) upon spiking. We have now:

$$V_i(t+1) = \gamma V_i(t)(1 - Z_i) + \frac{I_i(t)}{C} + \sum_{j=1}^{N} W_{ij} Z_j + \frac{\sigma_B}{C} B(t). \tag{4}$$

For a complete description see [2, 3]. The simulator allows to tune the parameters leak ($\gamma \in [0, 1]$); neuronal noise ($\sigma_B > 0$); membrane capacity (C); threshold (θ).

The connectivity matrix can be random or uploaded by a file offering the possibility to explore its impact on spike statistics.

Example 4 shows how to generates spikes using the retina simulator and Fig. 5 illustrates an example of simulation.

Example 4 Running the retina simulator.

1. FP `Data>Stimulus>Open`
2. PP Browse to choose the directory containing the sequence.
3. RP `Stimulus sequences` subpanel: The sequence of images is loaded with a default display duration of 100ms per image (`Display time` column). This value can be manually set or defined from time stamps file (see Example 3).
4. FP `Simulation>Virtual Retina>Retinal configuration>Open`
5. PP Browse and select the XML files containing the parameters of the retina. [3]
6. FP `Simulation>Virtual Retina>Retinal configuration>Customize`: Once loaded, user can still change parameters of the retina.
7. FP `Simulation>Virtual Retina>Compute spike`.
8. PP Set `Time step` Then `Compute` to get the spikes.

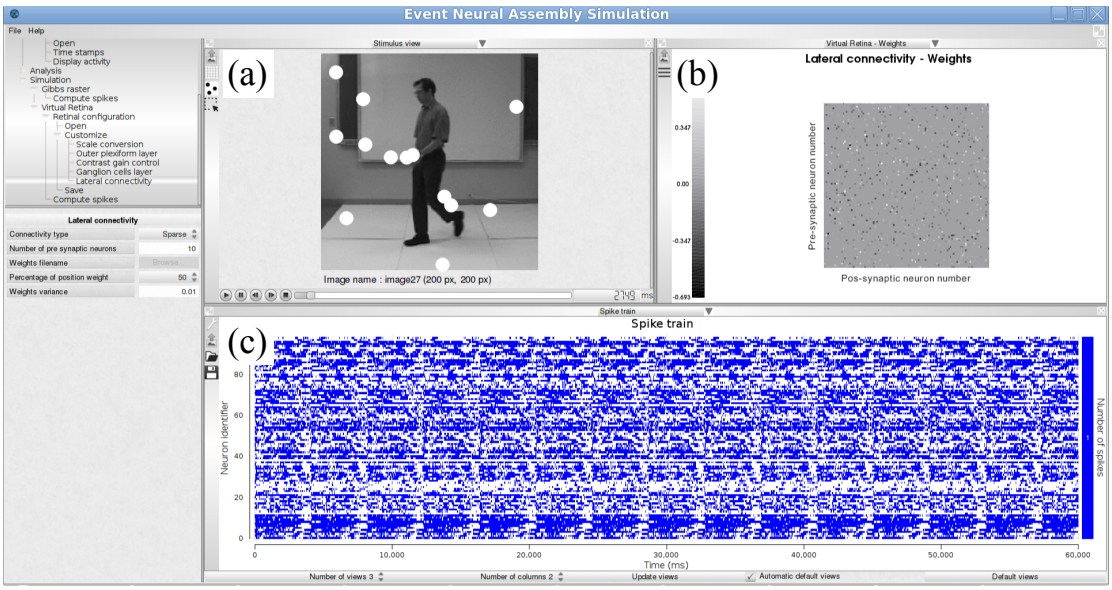

Figure 5: Spike generation from a virtual retina model. (a) Input image sequence with spiking activity super-imposed (white blobs). (b) Connectivity matrix chosen for the simulation, here random. (c) Simulated spike train.

[3] To test this example, one can use sample data available on ENAS website: retina configurations `linear_noSpikes.xml`, `cat_X_cell.xml` or `primate_ParvoMagno.xml`.

4 Conclusion

In this paper we have introduced ENAS, a new free software to analyze spike trains, coming from experiments or simulations. Here, we presented it's main features and we refer the interested reader to its website (*https://enas.inria.fr*) for downloads, more information (see, e.g., tutorials to get started) and a forum to exchange about your experience with other users. As explained, ENAS (version 1.0.0) includes a range of original methods from the state-of-the-art [4, 53] which are now available to the neuroscience community. To do so, we have focused on the following aspects:

- The user experience: We optimized the ergonomics of interactions between the user and the interface. No scripting is needed to analyze the spike trains.
- The richness of analysis: We developed a variety of analysis ranging from classical ones to the most recent analysis such as population analysis.
- The computational performance: We parallelized a selection of functions to make it possible to handle large populations of neurones faster.

We hope that ENAS will become a useful tool for neuroscientist to analyse spike trains and we expect to improve it thanks to user's feedbacks. Our goal is to progressively enrich ENAS with our latest research results, in order to facilitate transfer of new methods to the community. Future work will focus on improving several existing functions in terms of efficiency in time and memory consumptions. We will include methods for population analysis in the non-stationary case as well as other statistical analysis models. We are working on better methods for receptive fields estimation [10]. We are also developing extensions of the Virtual Retina model so that it could serve as a satisfactory model at a population level. Finally, as a general objective, we also plan to further improve the computational power of ENAS to handle even larger networks.

Acknowledgements

This work was supported by the French ministry of Research and University of Nice (ED-STIC), INRIA ADT, ERC-NERVI number 227747, KEOPS ANR-CONICYT, the European Union Project Renvision #600847 and by the ECOS-Sud program (grant C13E06). We warmly acknowledge Michael J. Berry, Maria-Jose Escobar, Matthias Hennig, Ruben Herzog, Gerrit Hillgen, Gaia Lombardi, Olivier Marre, Thierry Mora, Adrian Palacios, Cesar Ravello, Horacio Rostro, Vivien Robinet, Evelyne Sernagor, Gasper Tkacik, Juan-Carlos Vasquez for their invaluable help in designing Enas.

References

[1] G. Basalyga, M. A. Montemurro, and T. Wennekers. Information coding in a laminar computational model of cat primary visual cortex. J. Comput. Neurosci., 34:273–83, 2013.

[2] B. Cessac. A discrete time neural network model with spiking neurons. rigorous results on the spontaneous dynamics. J. Math. Biol., 56:311–345, 2008.

[3] B. Cessac. A discrete time neural network model with spiking neurons ii. dynamics with noise. J. Math. Biol., 62:863–900, 2011.

[4] B. Cessac and A. Palacios. Spike train statistics from empirical facts to theory: the case of the retina. In F. Cazals and P. Kornprobst, editors, Modeling in Computational Biology and Biomedicine: A Multidisciplinary Endeavor, Lectures Notes in Mathematical and Computational Biology (LNMCB). Springer-Verlag, 2012.

[5] B. Cessac, H. Rostro-Gonzalez, J.-C. Vasquez, and T. Viéville. How gibbs distribution may naturally arise from synaptic adaptation mechanisms: a model based argumentation. J. Stat. Phys, 136(3):565–602, August 2009.

[6] E. J. Chichilnisky. A simple white noise analysis of neuronal light responses. Network (Bristol, England), 12:199–213, 2001.

[7] E. J. Chichilnisky. A simple white noise analysis of neuronal light responses. Network: Comput. Neural Syst., 12:199–213, 2001.

[8] R. Cofré and B. Cessac. Dynamics and spike trains statistics in conductance-based integrate-and-fire neural networks with chemical and electric synapses. Chaos, Solitons & Fractals, 2013.

[9] R. Cofre and B. Cessac. Exact computation of the maximum-entropy potential of spiking neural-network models. Phys. Rev. E, 89(052117), 2014.

[10] A. Drogoul, G. Aubert, B. Cessac, and P. Kornprobst. A new nonconvex variational approach for sensory neurons receptive field estimation. In 6th International Workshop on New Computational Methods for Inverse Problems (NCMIP), 2016.

[11] M. Dudík, S. Phillips, and R. Schapire. Performance guarantees for regularized maximum entropy density estimation. In Proceedings of the 17th Annual Conference on Computational Learning Theory, 2004.

[12] E. Ferrea, A. Maccione, L. Medrihan, T. Nieus, D. Ghezzi, P. Baldelli, F. Benfenati, and L. Berdondini. Large-scale, high-resolution electrophysiological imaging of field potentials in brain slices with microelectronic multielectrode arrays. Front. Neural Circuits, 6(November):80, 2012.

[13] S. Garcia and N. Fourcaud-Trocmé. Openelectrophy: an electrophysiological data-and analysis-sharing framework. Frontiers in neuroinformatics, 3, 2009.

[14] K. S. Ginsburg, J. A. Johnsen, and L. Michael W. Common noise in the firing og neighbouring ganglion cells in goldfish retina. J. Physiol., page 433–450, 1984.

[15] D. H. Goldberg, J. D. Victor, E. P. Gardner, and D. Gardner. Spike train analysis toolkit: enabling wider application of information-theoretic techniques to neurophysiology. Neuroinformatics, 7(3):165–178, 2009.

[16] M. Greschner, J. Shlens, C. Bakolitsa, G. D. Field, J. L. Gauthier, L. H. Jepson, A. Sher, A. M. Litke, and E. J. Chichilnisky. Correlated firing among major ganglion cell types in primate retina. J. Physiol., 589(Pt 1):75–86, Jan. 2011.

[17] R. Herzog, J. Araya, M. Pizarro, B. Cessac, C. Ravello, M.-J. Escobar, and A. Palacios. From Habitat to Retina: Neural Population Coding using Natural Movies. Bernstein Conference 2015, Sept. 2014. Poster.

[18] G. Hilgen, S. Pirmoradian, D. Pamplona, P. Kornprobst, B. Cessac, M. H. Hennig, and E. Sernagor. Pan-retinal characterization of light responses from ganglion cells in the developing mouse retina. bioRxiv, 2016.

[19] R. A. Ince, A. Mazzoni, R. S. Petersen, and S. Panzeri. Open source tools for the information theoretic analysis of neural data. Frontiers in neuroscience, 4, 2010.

[20] E. Jaynes. Information theory and statistical mechanics. Phys. Rev., 106:620, 1957.

[21] M. Lidierth. sigtool: a matlab-based environment for sharing laboratory-developed software to analyze biological signals. Journal of neuroscience methods, 178(1):188–196, 2009.

[22] H. Lorach, R. Benosman, O. Marre, S.-H. Ieng, J. A. Sahel, and S. Picaud. Artificial retina: the multichannel processing of the mammalian retina achieved with a neuromorphic asynchronous light acquisition device. Journal of Neural Engineering, 9(6):066004, Oct. 2012.

[23] O. Marre, S. El Boustani, Y. Frégnac, and A. Destexhe. Prediction of spatiotemporal patterns of neural activity from pairwise correlations. Physical review letters, 102(13), Apr. 2009.

[24] A. Martinez-Alvarez, A. Olmedo-Payá, S. Cuenca-Asensi, J. M. Ferrandez, and E. Fernandez. RetinaStudio: A bioinspired framework to encode visual information. Neurocomputing, 114:45–53, Aug. 2013.

[25] K. Masmoudi, M. Antonini, and P. Kornprobst. Another look at the retina as an image scalar quantizer. In Proceedings of the International Symposium on Circuits and Systems (ISCAS), 2010.

[26] T. Masquelier. Relative spike time coding and stdp-based orientation selectivity in the early visual system in natural continuous and saccadic vision: a computational model. Journal of Computational Neuroscience, 32(3):425–441, 2012.

[27] R. Meier, U. Egert, A. Aertsen, and M. P. Nawrot. {FIND} — a unified framework for neural data analysis. Neural Networks, 21(8):1085 – 1093, 2008. Neuroinformatics Neuroinformatics.

[28] A. Mohemmed, G. Lu, and N. Kasabov. Evaluating SPAN Incremental Learning for Handwritten Digit Recognition. In Neural Information Processing, page 670–677. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[29] H. Nasser and B. Cessac. Parameters estimation for spatio-temporal maximum entropy distributions: application to neural spike trains. Entropy, 16(4):2244–2277, 2014.

[30] H. Nasser, O. Marre, and B. Cessac. Spatio-temporal spike train analysis for large scale networks using the maximum entropy principle and montecarlo method. Journal of Statistical Mechanics: Theory and Experiment, 2013(03):P03006, 2013.

[31] H. Nasser, O. Marre, and B. Cessac. Spatio-temporal spike trains analysis for large scale networks using maximum entropy principle and monte-carlo method. Journal Of Statistical Mechanics, 2013.

[32] F. J. Pelayo, S. Romero, C. A. Morillas, A. Martinez, E. Ros, and E. Fernandez. Translating image sequences into spike patterns for cortical neuro-stimulation. Neurocomputing, 58–60:885–892, 2004.

[33] C. Pouzat and A. Chaffiol. Automatic spike train analysis and report generation. an implementation with r, r2html and star. Journal of neuroscience methods, 181(1):119–144, 2009.

[34] C. Ravello, R. Herzog, B. Cessac, M.-J. Escobar, and A. Palacios. Spectral dimension reduction on parametric models for spike train statistics. 12e Colloque de la Société des Neurosciences , May 2015. Poster.

[35] F. Rieke, D. Warland, R. de Ruyter van Steveninck, and W. Bialek. Spikes, Exploring the Neural Code. The M.I.T. Press, 1996.

[36] Y. Roudi and J. Hertz. Mean field theory for non-equilibrium network reconstruction. Phys. Rev. Lett., 106(048702), 2011.

[37] Y. Roudi, S. Nirenberg, and P. Latham. Pairwise maximum entropy models for studying large biological systems: when they can work and when they can't. PLOS Computational Biology, 5(5), 2009.

[38] Y. Roudi, J. Tyrcha, and J. A. Hertz. Ising model for neural data: Model quality and approximate methods for extracting functional connectivity. Physical Review E, page 051915, 2009.

[39] E. Schneidman, M. Berry, R. Segev, and W. Bialek. Weak pairwise correlations imply strongly correlated network states in a neural population. Nature, 440(7087):1007–1012, 2006.

[40] J. Shlens, G. Field, J. Gauthier, M. Grivich, D. Petrusca, A. Sher, A. Litke, and E. Chichilnisky. The structure of multi-neuron firing patterns in primate retina. Journal of Neuroscience, 26(32):8254, 2006.

[41] K. D. Simmons, J. S. Prentice, G. Tkacik, J. Homann, H. K. Yee, S. E. Palmer, P. C. Nelson, and V. Balasubramanian. Transformation of stimulus correlations by the retina. PLoS Comput. Biol., 9(12):e1003344, Dec. 2013.

[42] H. Soula, G. Beslon, and O. Mazet. Spontaneous dynamics of asymmetric random recurrent spiking neural networks. Neural Computation, 18(1), 2006.

[43] M. Spacek, T. Blanche, and N. Swindale. Python for large-scale electrophysiology. Frontiers in neuroinformatics, 2, 2008.

[44] I. H. Stevenson and K. P. Kording. How advances in neural recording affect data analysis. Nature Neuroscience, 14(2):139–142, Jan. 2011.

[45] S. Strong, R. Koberle, R. de Ruyter van Steveninck, and W. Bialek. Entropy and information in neural spike trains. Phys. Rev. Let, 80(1):197–200, 1998.

[46] A. Tang, D. Jackson, J. Hobbs, W. Chen, J. L. Smith, H. Patel, A. Prieto, D. Petrusca, M. I. Grivich, A. Sher, P. Hottowy, W. Dabrowski, A. M. Litke, and J. M. Beggs. A maximum entropy model applied to spatial and temporal correlations from cortical networks *In Vitro*. The Journal of Neuroscience, 28(2):505–518, January 2008.

[47] G. Tkacik, O. Marre, T. Mora, D. Amodei, M. B. 2nd, and W. Bialek. The simplest maximum entropy model for collective behavior in a neural network. J Stat Mech, page P03011, 2013.

[48] G. Tkacik, J. S. Prentice, V. Balasubramanian, and E. Schneidman. Optimal population coding by noisy spiking neurons. PNAS, 107(32):14419–14424, August 2010.

[49] G. Tkačik, E. Schneidman, M. Berry II, and W. Bialek. Ising models for networks of real neurons. arXiv q-bio/0611072, 2006.

[50] G. Tkačik, E. Schneidman, M. J. Berry II, and W. Bialek. Spin glass models for a network of real neurons. arXiv preprint arXiv:0912.5409, 2009.

[51] J.-C. Vasquez, B. Cessac, and T. Viéville. Entropy-based parametric estimation of spike train statistics. In Statistical Mechanics of Learning and Inference, Stockholm-Mariehanm, May 2010.

[52] J. C. Vasquez, O. Marre, A. G. Palacios, M. J. Berry, and B. Cessac. Gibbs distribution analysis of temporal correlation structure on multicell spike trains from retina ganglion cells. J. Physiol. Paris, 106(3-4):120–127, May 2012.

[53] A. Wohrer and P. Kornprobst. Virtual retina: a biological retina model and simulator, with contrast gain control. Journal of Computational Neuroscience, 26(2):219–249, 2009.

[54] S. Zordan, M. Zanotto, T. Niels, S. Di Marco, H. Amin, A. Maccione, and L. Berdondini. A scalable high performance client/server framework to manage and analyze high dimensional datasets recorded by 4096 CMOS-MEAs. In 7th International IEEE/EMBS Conference on Neural Engineering (NER), 2015.

Contents

1 Introduction **4**

2 General presentation **4**

3 ENAS main functions **6**
3.1 Data 6
3.2 Analysis 7
3.2.1 Classical analysis 7
3.2.2 Population analysis 7
3.2.3 Receptive fields estimation 11
3.3 Simulation 14
3.3.1 From statistics 14
3.3.2 The retina simulator 14

4 Conclusion **17**

RESEARCH CENTRE
SOPHIA ANTIPOLIS – MÉDITERRANÉE

2004 route des Lucioles - BP 93
06902 Sophia Antipolis Cedex

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399