Spatiotemporal pattern coding using Neural Fields: Optimal parameter estimation

Mauricio Cerda, Bernard Girau

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Spatiotemporal pattern coding using Neural Fields: Optimal parameter estimation

Mauricio Cerda — Bernard Girau

N° 7543

February 2010

Computational Medicine and Neurosciences

ISSN 0249-6399 ISRN INRIA/RR--7543--FR+ENG

INSTITUT NATIONAL
DE RECHERCHE
EN INFORMATIQUE
ET EN AUTOMATIQUE

INRIA

centre de recherche
NANCY - GRAND EST

Spatiotemporal pattern coding using Neural Fields: Optimal parameter estimation

Mauricio Cerda*, Bernard Girau*

Theme : Computational Medicine and Neurosciences
Computational Sciences for Biology, Medicine and the Environment
Équipe-Projet Cortex

Rapport de recherche n° 7543 — February 2010 — 24 pages

Abstract: In this work we present the parameter optimization of a distributed model for the classification of temporal sequences. The model we present codes information with a population of units, and it can be applied to discriminate between several spatiotemporal sequences. The implementation of this classification technique strongly depends on how the the connection weights are set according to the sequence we want to code. This "learning" phase depends on several parameters, for which we present a detailed analysis in this work. In the first part, we give some examples and we retrieve the model parameters in *1D* by extending other existing studies. Later on we extend the analysis into *2D*, proposing at the end of the report a strategy to code any given spatiotemporal sequence that can be described as a set local spatiotemporal trajectories. The derivation of parameters is performed analytically and numerical simulations are performed to verify our results.

Key-words: neural fields, pattern classification, dynamical systems, computational neuroscience, spatiotemporal sequences.

* Université Henri Poincaré (UHP), LORIA, Cortex Project, Nancy, France; **Firstname.Lastname@loria.fr**.

Centre de recherche INRIA Nancy – Grand Est
LORIA, Technopôle de Nancy-Brabois, Campus scientifique,
615, rue du Jardin Botanique, BP 101, 54602 Villers-Lès-Nancy
Téléphone : +33 3 83 59 30 00 — Télécopie : +33 3 83 27 83 19

Codage de motifs spatiotemporels par Champs neuronaux: Estimation optimale des paramètres

Résumé : Dans ce rapport nous présentons l'optimisation du paramétrage d'un modéle distribué de discrimination de séquences temporelles. Le modèle que nous présentons code l'information au moyen d'une population d'unités, et il peut être appliqué à la discrimination de plusieurs séquences spatio-temporelles. L'implantation de cette technique de discrimination dépend fortement de la façon dont les poids de connexion sont fixés en fonction de la séquence que nous voulons coder. Cette phase d'apprentissage dépend de plusieurs paramètres, pour lesquels ce rapport présente une analyse détaillée. Dans la première partie, nous donnons quelques exemples et nous montrons comment déduire les paramètres du modèle dans sa forme *1D* à partir d'autres travaux existants. Ensuite nous étendons l'analyse au cas *2D*, en proposant à la fin du rapport une stratégie pour coder n'importe quelle séquence temporelle pouvant être décrite comme un ensemble de trajectoires spatio-temporelles locales. Les paramètres sont déterminés analytiquement et des simultations numériques sont réalisées pour vérifier nos résultats.

Mots-clés : champs neuronaux, reconnaissance de motifs, systémes dynamique, neuroscience computationnelle, séquences spatio-temporelles

Contents

1 Introduction **3**

2 Spatiotemporal patterns **4**
2.1 An example pattern: video sequences 4
2.2 Coding with snapshots 4
2.2.1 Complete Snapshots coding (1D ACNFT) 5
2.2.2 Complete Snapshots coding, an example 7
2.3 Coding with local features 7
2.3.1 Sparse local feature coding (*2D* ACNFT) 8
2.3.2 Sparse local feature coding, an example 9

3 Parameter tuning **10**
3.1 *1D* ACNFT 10
3.2 *2D* ACNFT 12
3.3 Generic *2D* trajectories 14

4 Conclusions **15**

Appendices **16**

A Analytical expression of $r_0(t)$ **16**

B Optimization of $v(\beta)$ **18**

C Analytical expression of $r_0(t)$, *2D* **19**

D Optimization of $v(\beta)$, *2D* **22**

E Numerical Simulations **23**

1 Introduction

In this report we present the details of a bio-inspired model for the coding and classification of spatiotemporal patterns based on the Continuous Neural Field Theory (CNFT). In particular we report, the way in which this model can be tuned in order to obtain coding and classification properties.

A spatiotemporal pattern can be understood as a volume in space and time, where space is a generic dimension depending on the problem (position, frequency or features in general). In computer vision, action recognition from video sequences is an example of classification of spatiotemporal patterns. Similarly, speech recognition in the audio processing field is another example of spatiotemporal pattern classification. Both problems have in common that is not just to recognize (and to code), the right picture or phoneme, but also to verify the right temporal sequence at a specific speed of presentation.

In the Continuous Neural Field Theory (or CNFT), we code space whatever its nature, as a population of units that represent it locally (close units code for close stimuli in space) and we study the temporal activity of these neurons with no transmission delay among units [1] and a linear dependency with the

input activity. The advantage of this kind of model is a distributed representation of space suitable to be deployed in parallel implementations and even tolerant to failures: for example if some of the units that represent the space "fail". The CNFT provides also a macroscopic description for the behavior of massive populations of units presenting an interesting bridge of discussion with neuroscience, from a theoretical point of view.

This work presents a variant of the CNFT theory able to code spatiotemporal patterns, considering them as a temporal sequence of spatial patterns. For this model that we are introducing, we show both analytically and experimentally that it can be effectively tuned to code and retrieve simple, yet generic, temporal sequences.

2 Spatiotemporal patterns

Spatiotemporal patterns can be found in different contexts, as we have mentioned. The main property of these patterns is the temporal dependency (there is a sequence), while the spatial dimension may change depending on the problem: for video processing it is usually the 2D space, for audio processing it is usually the Fourier domain. We will begin the presentation of our work with an illustrative example from the image processing domain for the coding and classification of spatiotemporal patterns. After this example, we will focus on a special kind of pattern, suitable to be described as a set of local spatiotemporal features. Later, we will show how it can be modeled with the help of the CNFT, and we derive the precise parameters required for it.

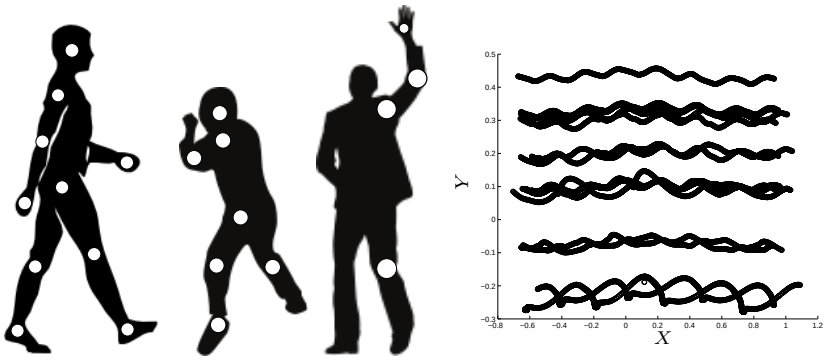
2.1 An example pattern: video sequences

In image processing, any video sequence can be seen as a temporal stack of images, where each image is a static pattern. Once we have seen a given sequence, we can ask, given a new sequence, if it corresponds to one of the sequences we have already seen. If we consider a special kind of videos: human motion, we can think of sequences such as to walk, to box or to wave, see Figure 1, and we could ask for a new video sequence, which of the actions we have already seen is the most similar. This kind of classification task must deal with the generalization/over-fitting problem [2].

2.2 Coding with snapshots

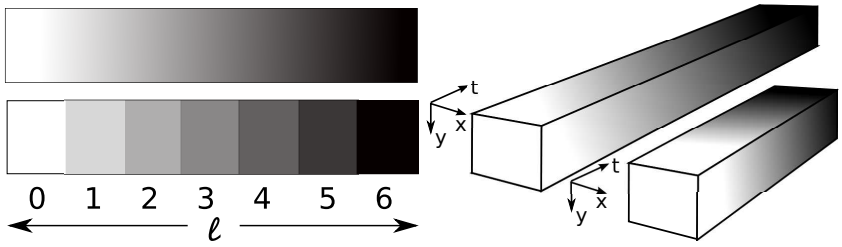
In general, a given spatiotemporal pattern will not have any structure, and we will be forced to completely record each "2D" spatial pattern (or snapshot, in the video example). Also, considering cyclic movements, the phase is not necessarily 0, *i.e.* the sequence does not necessarily start at the same point in time.

Suppose we want to code a simple on-to-off sequence as in Figure 2(a). The spatial dimension we are considering is light intensity (ℓ), a continuous variable that for the on-to-off sequence goes from 0 to 6. If after black (no light) it comes white again and between each light intensity there is a fix time, we can consider it as a cyclic sequence. We want to answer the question: is a given series of n images with speed of presentation v, in the on-to-off sequence?, see



(a) (b) (c) (d)

Figure 1: Human movements (a) to walk, (b) to fight and (c) to wave. In white we mark more relevant locations in terms of the information they can contribute to be differentiated from other sequences. (d) The set of trajectories in time for the joints in the walking sequence.



(a) (b)

Figure 2: Continuous and discrete light intensities. Also slow and fast discrete on-to-off sequence.

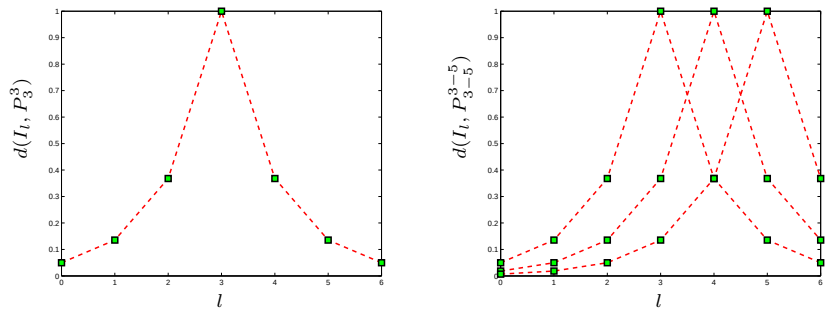
Figure 2(b) for different speeds of presentation. Additionally, we require answers with partial inputs ($n < 7$) and we will suppose that sequences do not always start at the same point in time (different phase).

Let us consider that the solution to this problem should first verify for each image of the set n, if it matches some image in the sequence. Only after this, we verify the temporal sequentiality. We will study this kind of solution, and propose a practical implementation using a variant of the CNFT.

2.2.1 Complete Snapshots coding (1D ACNFT)

The classification problem can be solved with several methods [2] from statistical decision theory such as LDA and Fisher's LDA, regression based techniques as MLP, RBF and SOM, nearest-neighbor or prototypes methods like k-nearest-neighbor and LVQ, without forgetting SVM's. Yet, if we require the solution to give partial answers while the input is displayed and also to answer in case of time distortions[1] (see Figure 2(b)), then the solution becomes less evident, even more if we want a robust distributed representation. In this case, an interesting

[1] usually called time-warping



(a) (b)

Figure 3: The similarity function for $P3_3$, and the sequence of similarities that corresponds to a part of the on-to-off sequence. Here we plot $d(I_l, P_k^k)$ for several P_k^k, using $d(I_l, P_k^k) = \exp(-(I_l - P_k^k)^2)$

approach to consider is by Giese et al. [3], where they consider a similarity function d between an input image (I) and a prototype image (P_k): in our discrete on-to-off example, we need 7 light intensity prototypes P_k (see 2(a) below). Note that in general the input (I) and prototypes (P_k) are images and as in the on-to-off sequence, images are single colored, we note I_l and P_l^k for the light intensity l of: the input image and the kth prototype image respectively. Also, as we can choose freely where to start to count the prototypes we impose P_k^k.

For each input image at any time we will have an answer from all prototypes, as the pattern may start at any light intensity, in our example. We can see the profile of responses for three prototypes with respect to light intensity of different inputs (I^l), in Figure 3(b). The shape of the profiles in Figure 3 will be given by the similarity function, but we still need to verify that images appears at precisely the right order and speed. To illustrate how the mechanism work we will explain it recursively. Suppose that we are verifying that image I at time t (or $I(t)$) belongs to the sequence, we know that until time $t-1$ the sequence is correct and we also know that next image in the sequence must be P_n. In this case we need to verify that image $I(t)$ corresponds to the prototype P_n. If both elements: history and current image are correct, we can stand that the image $I(t)$ correspond to the sequence.

In the CNFT, we represent the spatial dimension (light intensity) as units, where we describe the variation of the activity for each unit as a differential equation with a decay term (for the on-to-off sequence there are 7 units), see the left part of Eq. 1. In this specific case we consider the input as the similarity function that we mention, and we require the interaction between units to account for the verification of the sequence. All of which can be summarized in Eq. 1.

$$\tau \frac{\partial m(k,t)}{\partial t} + m(k,t) = \left[\sum_{k'} w(k'-k) m(k',t) + D(I(t), P^k) \right]^+ \qquad (1)$$

where we note $D(I, P^k) = d(I_l, P^k_k)$ to simplify the notation, $[\ ]^+$ is the non-linear rectification equivalent to $max(\ ,0)$. The function $w(k'-k)$ codes the sequence and the speed the light should change. Here, w must be asymmetric with respect to $k'-k$, to account for the sequentiality. The precise degree of asymmetry (the parameters of w) codes for the speed and it requires to determined, as it will be explained further in Section 3.1. To the the variation of the CNFT with an asymmetric kernel (w) we call it Asymmetric CNFT, or ACNFT.

To answer if an input image sequence is in the on-to-off sequence with a given speed, we look at the value of,

$$A(t) = \sum_{k,t} m(k,t) = 2\pi \sum_{t} r_0(t) \tag{2}$$

which should be maximal for the on-to-off sequence at the speed for which we tune the system. In Eq. 2, $r_0(t)$ accounts for the total instantaneous activity over space. Any other spatiotemporal sequence either with different spatial patterns or temporal order should deliver a lower value of A. This model allows to correctly verify a given sequence, and to answer as input images become available, not necessarily with the complete sequence.

There are several metrics we can use as d function, in general any RBF as other authors have proposed [4]. However, for the exact shape of w in terms of the input speed v there is no analytical result to our knowledge. The authors of [4] report manual tuning by using data directly. In the next section we will show how to effectively tune the model for a given input speed v, in such a way to guarantee a maximal value of A for the right sequence.

2.2.2 Complete Snapshots coding, an example

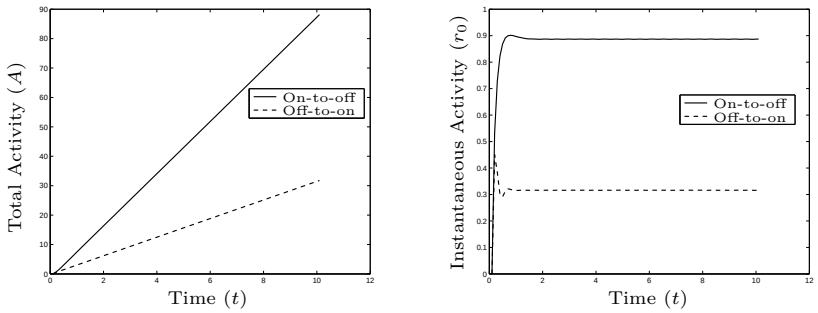
Following the on-to-off example, we show now how Eq. 1 provides an effective way to code and classify spatiotemporal patterns.

To check the sequentiality, we will consider two possible input sequences: the on-to-off and the exact inverse off-to-on sequence and only one population of units coding the on-to-off sequence. The values of A as function of time t, with A defined as in Eq. 2, can be seen in Figure 4(a), where it can be observed that the ACNFT is able to distinguish the on-to-off sequence before the complete pattern is presented (at $t = 6$) as the value of $A(t)$ is higher with the on-to-off sequence as input than with the off-to-on sequence, converging quickly to the stable state of the system as can be seen in Figure 4(b),

Until now, we have introduced a variant of the CNFT, able to code spatiotemporal patterns, mainly developed by [4, 5, 6]. In Section 3.1 we present the theoretical analysis for the optimization of parameters in the *1D* case, to make the model viable to be implemented as a classification system. Details about the parameters and methods of the numerical simulation, can be found in the Appendix E.

2.3 Coding with local features

As we have seen, we can code an arbitrary spatiotemporal sequence as a series of spatial patterns keeping full record of each spatial pattern, and for the temporal



(a) (b)

Figure 4: Temporal evolution of: (a) the total activity $A(t)$ and (b) the instantaneous activity $r_0(t)$ for the *1D* example.

part, we can use a differential equation to represent the speed the sequence should appear. To perform this we require a distance function d, to compare the input pattern with the prototypes that compose the sequence stored in advance (P_k).

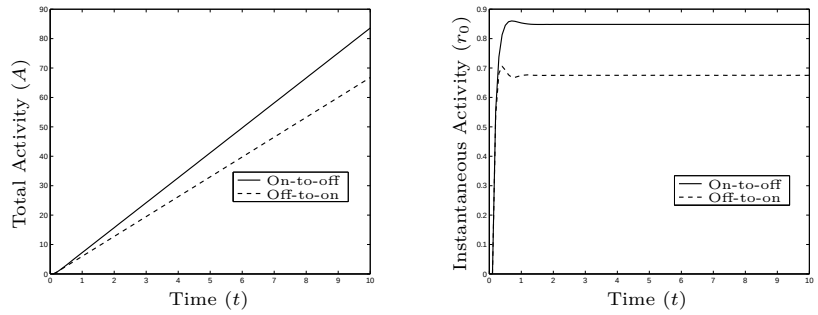
In fact, we do not need the metric d and we do not need to store the complete spatial patterns if the spatiotemporal pattern can be described as a sparse set of local features in the spatiotemporal volume. Many sequences can be decomposed as a set of "trajectories" in time: in human movements if we enlight the joints, the sequences can be described with a few trajectories, see Figure 1. The sparse local feature coding that we propose in *2D* using the ACNFT, takes advantage of this idea.

2.3.1 Sparse local feature coding (*2D* ACNFT)

The idea to code a spatiotemporal volume using local features has been addressed before with the local-feature description, specially in the image processing domain [7]. In these methods common issues are: how they respond to variations in speed (time compression/dilation), the lack of response for partial patterns and phase-dependent recognition. For those reasons, a model that combines both elements (the advantages of local feature detection with a distributed mechanism to compare patterns improving these issues) appears as an interesting and appealing idea.

To code spatiotemporal patterns in *2D*, we consider only those that can be described as a set of trajectories (joints trajectories for human movement in Figure 1). The main restriction for the local-features is that they have to be "close" in time, forming continuous trajectories in the space-time volume. We apply the ACNFT model to the set of trajectories, but without loss of generality we first focus on a single trajectory.

The *2D* ACNFT directly uses a local feature detection g over the image (see Eq. 3). The difference with the metric d in *1D* is the absence of prototypes, performing an isotropic feature detection in g, in other words an implicit representation of the prototypes. Furthermore, we modify the kernel w, to code the



(a) (b)

Figure 5: Temporal evolution of: (a) the total activity $A(t)$ and (b) the instantaneous activity $r_0(t)$ for the *2D* example.

location where the trajectory goes through and its corresponding speed. This can be summarized as,

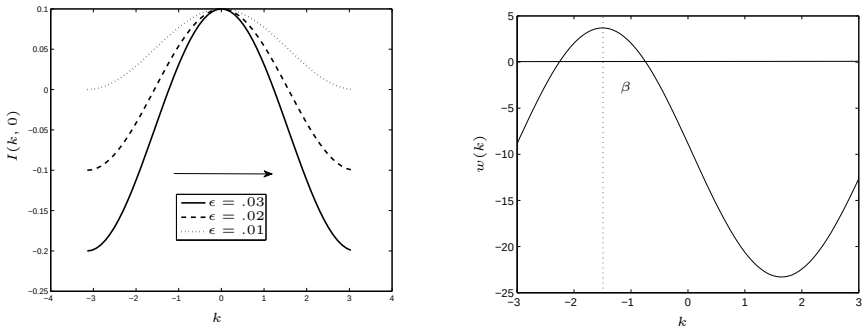
$$\tau \frac{\partial m(\vec{x},t)}{\partial t} + m(\vec{x},t) = \left[\int_{\Omega} w(\vec{x}',\vec{x}) m(\vec{x}',t) d\vec{x}' + g(I(\vec{x},t)) \right]^{+} \tag{3}$$

Eq. 3 differs from Eq. 1 by the spatial variable $\vec{x}$ of the pattern and by the input defined as $g(I(\vec{x},t))$, where $g(\cdot)$ is transformation independent from the sequence we want to code (local descriptors [7], see also [8] for a discussion of existing pattern coding schemes). Also, the w function no longer depends only on the distance, but also on the position of the unit (because we code the precise trajectories). To answer if an input image sequence belongs to a previously coded sequence with a given speed, we look at the total value of Eq. 2 as in the *1D* case.

2.3.2 Sparse local feature coding, an example

To visualize the kind of pattern we could potentially code, we consider for example several moving trajectories that correspond to the positions of the joints for a moving person in a video sequence, see Figure 1. The first example we will consider is the simple case of one left-to-right and one right-to-left trajectory moving at the same speed, where we use the same ACNFT population to discriminate between both.

We can see the results of the *2D* ACNFT in Figure 5(a), where the value $A(t)$ is larger for the speed the system codes for, and the convergence to the stable state is fast, see Figure 5(b). In this example, the input speeds where $\{5, -5\}$ in rad/frames (same conditions as in the *1D* example), and the system had the same parameters for both inputs, see Appendix E for more details on the numerical simulation. In this particular example, we use a version of the *2D* ACNFT detailed in Section 3.2 able to code only one trajectory at a time.



(a) (b)

Figure 6: (a) Input function, at difference contrast levels. (b) Asymmetric kernel function for $\beta = 1.5$, considering a neuron at position 0, only neurons at position -1.5 or close will have positive weights.

3 Parameter tuning

The model presented in the first part of the previous section and introduced by [3, 4, 5], is able to code any video sequence by saving snapshots (complete pictures) at each time instant (the *1D ACNFT*). In the second part, we introduced a new model able to code any *2D* sequence that can be decomposed into a set of local features (the *2D ACNFT*). However, for both cases, in order to implement them, we need to precise the shape of w and link it to the speed of the input sequence v. The following analysis has been inspired by the work of [5] in *1D* and extended to the *2D* case.

3.1 *1D* ACNFT

To determine the shape of w, we will use the form proposed by [6] see Figure 6(b),

$$w(k' - k) = J_0 + J_1 \cos(k' - k + \beta) \tag{4}$$

where k is the spatial variable. The w function should be asymmetric, periodic and continuous at least for the analysis, but this precise form only simplifies the analysis. This function has its highest value at $k' - k = -\beta$. This implies $k > k'$ if $\beta > 0$, or for the on-to-off sequence that this function gives a maximal positive value to units placed at distance β to the left of the unit k (*i.e.* before in terms of time of the sequence).

As we have mentioned, to code different speeds of appearance for the stack of images, the β parameter in Eq. 4 should be tuned accordingly to the input speed v. To derive $\beta(v)$, this should maximize A as define in Eq. 2 when the input is at speed v. To start the analysis we rewrite A in the continuous case as $\int \left(\int m(k,t)dk' \right) dt = 2\pi \int r_o(t)dt$, and we impose also a system with stable response, *i.e.* $dr_0/dt = 0$, allowing us to write $A = (t_f - t_0)r_0$.

In fact, r_0 is the first Fourier component of m; to simplify the calculations we consider k to be an angular variable, to account for cyclic patterns and to avoid border conditions,

$$r_0(t) = \int_{-\pi}^{\pi} m(k,t)(2\pi)^{-1} dk \tag{5}$$

r_0 is calculated in [5, 6] for the *1D* system assuming that the expected activity in $m(k,t)$ evolve as one single "bump" moving at input speed v and that this front has width $2k_c(t)$, where $k_c(t)$ is an unknown function of time. Using $k_c(t)$, we can deal with the non-linearity $[\ ]^+$ in the right side of Eq. 1, which is non-zero only in the interval $[\phi(t) - k_c(t), \phi(t) + k_c(t)]$), where $\phi(t)$ is the position for the center of the bump.

Considering the single bump assumption, introducing $k_c(t)$ and imposing simultaneously that $dr_0/dt = 0$ and $dr_1/dt = 0$, r_0 can be written as (see details in Appendix A),

$$r_0(t) = \frac{S}{-J_0 - \cos(k_c)/f_0(k_c)} \tag{6}$$

where f_0 is a increasing function of k_c as defined in [5, 6], S is the contrast of the sequence parameter (or the rate between the minimum and maximum values, see Figure 6(a)). As we want $\beta(v)$ to maximize r_0, in Eq. 6 this is equivalent to minimize k_c ($J_0 < 0$ and $S > 0$). To minimize k_c, we use the constraint of a single bump of activity in Eq. 7, first introduced in [6] and detailed in Appendix A,

$$S' = \frac{J_0 f_0(k_c) + \cos(k_c)}{\sqrt{J_1^2 f_1^2 \cos(\Delta)^2 - 2J_1 f_1 \cos(\Delta)\cos(\Delta+\beta) + 1}} \tag{7}$$

here S' is a parameter that depends on the shape of the input sequence, f_1 is a function of k_c and $\Delta = \arctan(\tau v)$. Deriving Eq. 7 with respect to v and asserting $dk_c/dv|_{v=v_m} = 0$ (see details in Appendix B), we obtain the following expression for the optimal speed (v_m) as function of β (here our analysis goes beyond [5]):

$$v_m = \frac{J_1 f_1(k_c)\tau - 2\tau\cos(\beta) - \sqrt{J_1^2 f_1^2(k_c)\tau^2 - 4\tau^2 J_1 f_1(k_c)\cos(\beta) + 4\tau^2}}{2\tau^2 \sin(\beta)} \tag{8}$$

all the terms in Eq. 8 are known except for $k_c(t)$, but the solution we look for must be independent of the input if we want to apply it to other sequences and link v_m with β. Considering the limit when the input width tends towards a localized input, this implies k_c tends towards zero, simplifying v_m,

$$v_m(\beta) = \frac{1 - \cos(\beta)}{\tau \sin(\beta)} \tag{9}$$

To verify this result, we took the known no-contrast limit of [5] (or width of the input $\to \infty$) and our high contrast limit at Eq. 9 (or width of the input

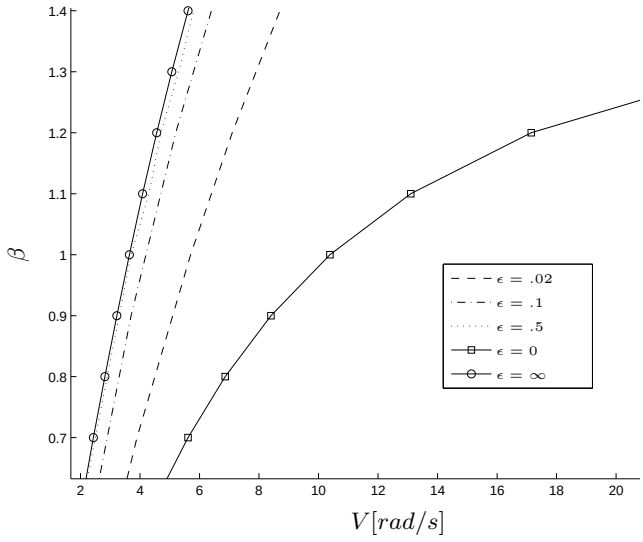


Figure 7: The $\beta(v)$ function for the maximal mean activity (r_0) at different values of ϵ. Continuous lines correspond to theoretical results in low ($\epsilon = 0$) and high contrast limits ($\epsilon = \infty$), the others tree dotted curves to simulations.

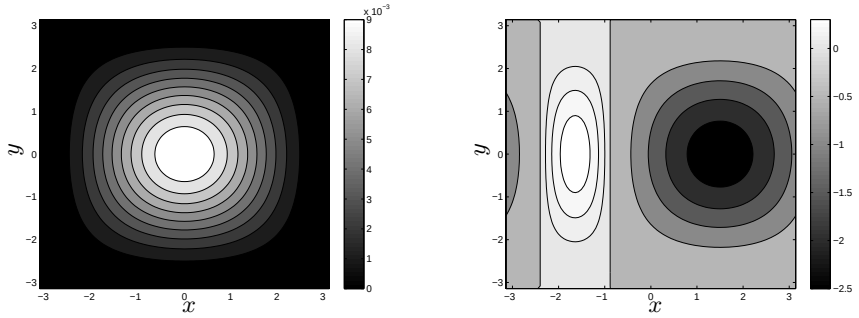
$\rightarrow 0$), and we ran simulations to check that inputs with medium contrast are better represented by our high contrast limit, see Figure 7. In Figure 7 we see that as the input contrast (ϵ) is higher our approximation of $v_m(\beta)$ is better, or in other terms, for a given input speed we can derive more precisely the right β so that the mean activity r_0 is maximal. By consequence, we maximize A as function of β. We can also notice that in the high contrast limit the relation between the optimal β and the input speed v is almost linear.

3.2 *2D* ACNFT

In *2D*, the ACNFT codes a set of trajectories in space with the same population, where each unit has a local kernel w. We will consider in this analysis the case where w has the size of the input image, *i.e* we can code only one trajectory with our system (one joint at a time, in Figure 1). If features are sparse "enough" this will be the case in general.

$$w(\vec{x}, \vec{x}') = p(y, y')q(x, x') = A\left(1 + \cos(y' - y)\right)\left(J_0 + J_1 \cos\left(x' - x + \beta\right)\right) \quad (10)$$

where $\vec{x} = (x, y)$ and $\vec{x}' = (x', y')$. In Eq. 10 we impose $w(\vec{x}, \vec{x}') = p(y, y')q(x, x')$ where the asymmetry will be only in the q function as in the *1D* case, and the $p(y, y')$ function is a RBF or something close (cosine in our analysis). The separability of w helps with the analysis, since as along the x-axis we have a system similar to the *1D* case, see Figure 8(b). The input we consider is again a spatial bump, moving in the x-direction without loss of generality, see Figure 8(a).



(a) (b)

Figure 8: (a) Input function, the deplacement is only in the x-axe. (b) Asymmetric kernel function for $\beta = 1.5$ for a neuron at $(0, 0)$, only neurons at $x = -1.5$ or close will have positive weights.

The first Fourier component of Eq. 3 needs to be computed as in the *1D* case,

$$r_0(t) = \int_{-\pi}^{\pi} \int_{-\pi}^{\pi} m(\vec{x}, t)(2\pi)^{-2} d\vec{x} \tag{11}$$

where the spatial variable y is also considered as angular and $(x, y) \in [-\pi, \pi] \times [-\pi, \pi]$. Using the separability of variables of w and the single bump restriction as for the *1D* case, we can derive an expression for r_0 (see Appendix C for details).

$$r_0(t) = \frac{(C + D^2)}{-AJ_0 - (2\cos(x_c) + \cos(x_c)^2)/g_0(x_c)} \tag{12}$$

The main difference is that now the first Fourier component (r_0) or the total activity depends on the size of the bump of activity in both axes. The size can be expressed as x_c for the width in the x-axis and y_c for the height in the y-axis.

As in the *1D* case, we want to tune parameter β of this system so as, to maximize r_0, see Eq. 12. The value of r_0 depends on $g_0(x_c)$ which is an increasing function of x_c and y_c (see details in Appendix D), which is equivalent to say that r_0 is a decreasing function of x_c and y_c. The next step is to use the single bump constraint of the solution, and to make $dx_c/dv|_{v=v_m} = 0$ and $dy_c/dv|_{v=v_m} = 0$. We thus obtain (see details in Appendix D),

$$v_m = \frac{AJ_1 g_1(x_c, y_c) - 2\cos(\beta) - \sqrt{(AJ_1)^2 g_1^2(x_c, y_c) - 4AJ_1 g_1(x_c, y_c)\cos(\beta) + 4}}{2\tau \sin(\beta)} \tag{13}$$

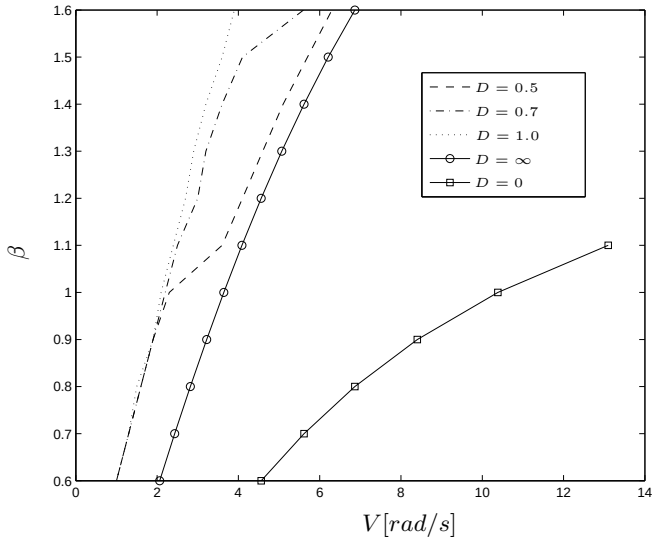


Figure 9: The $\beta(v)$ function for the maximal mean activity (r_0) at different values of the contrast. Continuous lines correspond to theoretical results in low and high contrast limits, the others tree dotted curves to simulations. Note that we use $C = 4D^2 - 1$, to have an amplitude of 1 in all the cases.

v_m is the magnitude of the speed, as the input is defined to be moving only in the x direction. The difference with the *1D* case is the g_1 function, defined in Appendix 13. For v_m we can again use the high contrast limit ($g_1(x_c, y_c) \to 0$),

$$v_m(\beta) = \frac{1 - \cos(\beta)}{\tau \sin(\beta)} \tag{14}$$

which is the same results as in Eq. 9. In a similar way for the the low contrast limit $v_m = \frac{\tan(\beta)}{\tau}$ can be found in the *2D* case, as obtained by [5] in the *1D* case. To verify both limits in the *1D* case, we perform simulations, see Figure 9, where it can be seen that at different input contrast levels (in the *2D* case, the contrast is defined by C and D) the optimal value of β is close to the high contrast limit we obtained, yet is not as precise as the *1D* case, probably due a to several constraints we assume that do not completely hold in general, see the end of Appendix C for details.

3.3 Generic *2D* trajectories

Until now we have considered straight line trajectories in *2D*, moving along a single axis with constant speed. Yet, curved trajectories can still be coded considering them locally as rectilinear ones. Concerning trajectories where $dv/dt \neq 0$, they can still be coded with our model if it is possible to consider them as piece-wise functions with constant speed.

In the general case, if we want to code a given trajectory we have each position available in space $\vec{x}^i = (x^i, y^i)$ in pixels and its derivative $\vec{v}^i = (v_x^i, v_y^i)$,

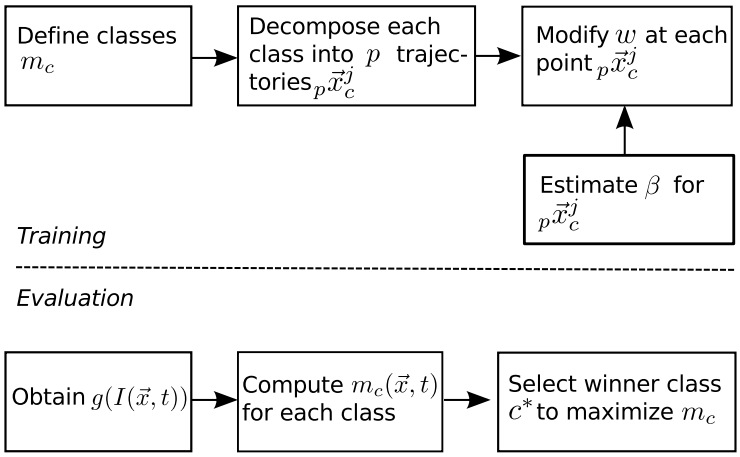


Figure 10: Schematic view illustrating training and evaluation stages for the classification task with our proposed method.

in pixels per frame. Due to implementation issues the size of the kernel should be much smaller than the total size of the field $\sigma \ll \Sigma$, and obviously this size limits the maximal speed the local kernel will be able to code as $|\beta| < \sigma/2$. On the other hand, the minimal speed is constrained to the equivalent of 1 pixel/frame, or $|\beta| > 2\pi/\sigma$.

The position in pixels within the trajectories identifies the unit which kernel should be modified directly as $\vec{x}^i = (x^i, y^i)$. Then, the direction of movement θ can be calculated as $\theta = \arctan(v_y^i/v_x^i)$, this angle is the rotation necessary to put the x-axis in the direction of motion. The magnitude of the speed in pixels $|\vec{v}^i|$ must be transformed into radians per frame with $v_r^i = |\vec{v}^i|2\pi/\sigma$, then used so as to calculate the asymmetry in the axis of movement to obtain β^i.

$$x_\theta = (x - x^i)\cos(\theta) - (y - y^i)\sin(\theta) \tag{15}$$

$$y_\theta = (x - x^i)\sin(\theta) + (y - y^i)\cos(\theta) \tag{16}$$

$$w(\vec{x}_\theta) = \exp\left(-\frac{1}{2\sigma^2}\left((x_\theta - \beta^i)^2 + y_\theta^2\right)\right)\cos\left(\frac{2\pi x_\theta}{\sigma} + \beta^i\right) \tag{17}$$

4 Conclusions

In this work we have presented an analytic optimization of the parameters in the asymmetric CNFT, where we assume a precise form of the input and the kernel. More precisely, we maximize the amplitude of the activity, for the stable state as function of the asymmetric parameter β in both, *1D* and *2D* scenarios. The optimization in *1D* and the complete analysis in *2D* are our main contributions, and allow us to propose a new classification model based in the decomposition of spatiotemporal volume into set of local trajectories, see Figure 10.

The results we have shown indicate that as the input stimuli gets more and more localized in space (less noisy), the relation between the asymmetry

parameter (β) and input speed (v) becomes almost linear, far from the intrinsic velocity of the system in the absence of stimuli. A simple relation like this for the asymmetric CNFT is interesting: as other inputs (with different shapes) get more localized, they will have a function $\beta(v)$ close to linear so as, to maximize the amplitude. Following this idea this we propose how to generalize the ACNFT for generic trajectories, see Figure 10, in order to build a classification model based in our ideas.

The kind of analysis we have presented is different to some other approaches, for example numerical ones [9, 10], where no assumptions about the input are performed. Yet, as we have shown in this work, even if we use a specific input function the results may hold in general cases, for example taking the appropriates limits. Nevertheless, it will be interesting to compare numerical approaches as the ones proposed by [9, 10], to see how the solutions differ (or not).

Appendices

A Analytical expression of $r_0(t)$

This first analysis has been performed elsewhere (see [5, 6]), and we only make explicit and clarify some of the steps. Given Eq. 1, we first look for a closed form for the mean activity (first Fourier component), as defined in Eq. 5. To achieve this we use the same input as in [5, 6],

$$I(k,t) = C\left[1 - \epsilon + \epsilon\cos(k - vt)\right] - T \tag{18}$$

at $t = 0$ this represents a bump around $k = 0$, shifting towards the right at speed v, where the parameters C, ϵ and T control the ratio between maximal $(C - T)$ and minimal $(C - T - 2\epsilon)$ activity. We also need the dynamics of the second Fourier component $r_1(t)$ defined in Eq. 19, because $r_0(t)$ depends on it, as we will see later on.

$$r_1(t) = \int_{-\pi}^{\pi} m(k',t)(2\pi)^{-1} e^{i(k' - \psi(t))} dk' \tag{19}$$

In Eq. 19 $\psi(t)$ is an unknown function of time, that makes $r_1(t)$ a real and positive number, in other words for a single bump solution, this is the peak of the bump position (or phase in the complex plane). Introducing Eq. 18 and Eq. 4 into Eq. 1 and identifying r_0 as defined in Eq. 5 we can rewrite Eq. 1 as,

$$\tau\frac{\partial m(k,t)}{\partial t} + m(k,t) = \Bigg[J_0 r_0(t) + C(1-\epsilon) - T + C\epsilon\cos(k - vt) + J_1\int_{-\pi}^{\pi} m(k,t)(2\pi)^{-1}\cos(k' - k + \beta)dk'\Bigg]^{+} \tag{20}$$

here the point is to simplify as much as possible any dependency in the space, and to impose a single bump solution. To do this, we can rewrite the right-side

of Eq. 20 in the complex plane to simplify calculations (considering only the real part) and then using Eq. 19,

$$
\begin{aligned}
&J_0 r_0 + C(1-\epsilon) - T + C\epsilon e^{-i(k-vt)} + J_1 \int_{-\pi}^{\pi} m(k,t)(2\pi)^{-1} e^{i(k'-k-\beta)} dk' \\
&J_0 r_0 + C(1-\epsilon) - T + C\epsilon e^{-i(k-vt)} + J_1 \int_{-\pi}^{\pi} m(k,t)(2\pi)^{-1} e^{i(k'-k-\beta+\psi-\psi)} dk' \\
&J_0 r_0 + C(1-\epsilon) - T + C\epsilon e^{-i(k-vt)} + J_1 \int_{-\pi}^{\pi} m(k,t)(2\pi)^{-1} e^{i(k'-\psi)} e^{i(-k-\beta+\psi)} dk' \\
&J_0 r_0 + C(1-\epsilon) - T + C\epsilon e^{-i(k-vt)} + J_1 r_1(t) e^{i(-k-\beta+\psi)} \qquad (21)
\end{aligned}
$$

To ensure a bump solution, we assume a single cosine shape as total input (all the terms inside $[\ ]^+$) introducing another variable: $\phi(t)$, which represents the spatial position of the total input center in Eq. 21 (not in m as $\psi(t)$). Taking only the real part of Eq. 21, and introducing $\phi - \phi$ we obtain,

$$
\begin{aligned}
I_0(t) + \cos(\phi - k)(J_1 r_1 \cos(\psi - \beta - \phi) + C\epsilon \cos(vt - \phi)) + \\
\sin(\phi - k)(J_1 r_1 \sin(\psi - \beta - \phi) + C\epsilon \sin(vt - \phi)) \qquad (22)
\end{aligned}
$$

where $I_0(t) = J_0 r_0(t) + C(1-\epsilon) - T$. The single bump shape for the solution translates then into imposing Eq. 23.

$$
J_1 r_1 \sin(\psi - \beta - \phi) + C\epsilon \sin(vt - \phi) = 0 \qquad (23)
$$

Using Eq. 22, the condition in Eq. 23 and the auxiliary variable $I_1(t) = J_1 r_1 \cos(\psi - \beta - \phi) + C\epsilon \cos(vt - \phi)$, we can rewrite Eq. 1 as:

$$
\tau \frac{\partial m(k,t)}{\partial t} + m(k,t) = [I_0(t) + \cos(k - \phi(t)) I_1(t)]^+ \qquad (24)
$$

Until now, we have imposed a shape for the solution (single bump) and thus we have obtained a restriction (Eq. 23). Rewriting the system, we have derived Eq. 24. The next step is to deal with the non-linearity $[\quad]^+$ and to show the existence and the stability of the solution, *i.e.* $dr_0(t)/dt = 0$ and $dr_1(t)/dt = 0$. In order to simplify the non-linearity we use variable $k_c(t)$ (the unknown width of the total input bump). When $k_c = k - \phi$, the right side of Eq. 24 is zero and we obtain $I_0(t) = -I_1(t)\cos(k_c)$, this allows us to write,

$$
\tau \frac{\partial m(k,t)}{\partial t} + m(k,t) = [I_1(t)\,(\cos(k - \phi(t)) - \cos(k_c))]^+ \qquad (25)
$$

Before using $dr_0(t)/dt = 0$ and $dr_1(t)/dt = 0$, the last step is to transform Eq. 25 into the Fourier domain by integrating each term in Eq. 25 with $\int_{-\pi}^{\pi}(2\pi)^{-1}dk$ for the first Fourier component, and with $\int_{-\pi}^{\pi}(2\pi)^{-1}e^{ik}dk$ for the second component. The second component derives into two equations since it is complex, describing the dynamics of the system with a total of 3 coupled equations in the Fourier domain, see Eqs. 26, 27 and 28.

$$\tau \frac{\partial r_0(t)}{\partial t} + r_0(t) = I_1(t) f_0(k_c) \tag{26}$$

$$\tau \frac{\partial r_1(t)}{\partial t} + r_1(t) = I_1(t) f_1(k_c) \cos(\phi - \psi) \tag{27}$$

$$\tau r_1(t) \frac{\partial \psi(t)}{\partial t} = I_1(t) f_1(k_c) \sin(\phi - \psi) \tag{28}$$

f_0 and f_1 are increasing functions of k_c defined as in [5, 6]. The system is coupled because $I_1(t)$ depends on $r_1(t)$. Now, we can impose $dr_0(t)/dt = 0$, $dr_1(t)/dt = 0$ and $d\psi(t)/dt = v$ (input and solution for $m(k,t)$ move at the same speed), to finally obtain the closed form of r_0 and r_1,

$$r_0(k_c) = f_0(k_c) \frac{S}{-J_0 f_0(k_c) - \cos(k_c)} \tag{29}$$

$$r_1(k_c) = f_1(k_c) \frac{S}{-J_0 f_0(k_c) - \cos(k_c)} \cos(\Delta) \tag{30}$$

where $S = (C(1-\epsilon) - T)$ and $\Delta = \arctan(\tau v)$. This solution exists (and if it exists we know it is stable) if the condition in Eq. 23 can be achieved. To check this we inject Eqs. 29 and 30 into Eq. 23, obtaining:

$$S' = \frac{J_0 f_0(k_c) + \cos(k_c)}{\sqrt{J_1^2 f_1^2 \cos(\Delta)^2 - 2 J_1 f_1 \cos(\Delta) \cos(\Delta + \beta) + 1}} \tag{31}$$

where $S' = (1 - (C\epsilon/C - T)^{-1})^2$. Then the system has a solution, which is stable, only if Eq. 31 can be verified for a given set of parameters. This verification cannot be performed analytically (we do not know k_c), but numerically changing v and β and fixing the other parameters there is a range of parameters where Eq. 31 can be verified.

B Optimization of $v(\beta)$

The analysis detailed now is original from this work and it differs from [6] in the asymmetry and in the results we obtain and the limits we verify from [5].

Once the shape of $r_0(t)$ has been determined, see Appendix A for details, we want to optimize it as a function of the input speed v. In other words, we want $\beta(v)$ to maximize r_0 for stable solutions, *i.e.* that verify Eq. 31.

First, it can be noticed from Eq. 32 that maximizing $r_0(k_c)$ is equivalent to minimizing k_c in Eq. 29 as f_0 is an increasing function of k_c, $J_0 < 0$ and $S > 0$.

$$r_0(k_c) = \frac{S}{-J_0 - \cos(k_c)/f_0(k_c)} \tag{32}$$

In the other hand, Eq. 31 can be derived in terms of v and setting the extrema at zero *i.e.* $\partial k_c/\partial v|_{v=v_m} = 0$, we can obtain the minimum k_c in terms of v, or v_m. If we apply the operator $\frac{\partial}{\partial v}$ in Eq. 31, we obtain two solutions for v_m:

$$v_m = \frac{J_1 f_1(k_c)\tau - 2\tau\cos(\beta) \pm \sqrt{J_1^2 f_1^2(k_c)\tau^2 - 4\tau^2 J_1 f_1(k_c)\cos(\beta) + 4\tau^2}}{2\tau^2 \sin(\beta)} \quad (33)$$

In order to choose between both solutions, we verify that at the limit $J_1 f_1 \to \frac{1}{cos(k_c)}$, the solution must verify the only admissible solution $v = \tan(\beta)/\tau$ to the system, see details on how to obtain this limit in [5]. This makes us choose the "$-$" solution. It can be noticed that as the optimization starts with a stable solutions, the solution we found belongs to this regime. This remark is important, as this kind of system usually gets unstable as the input speed is more distance from the intrinsic speed of the system (usually as lurching waves, see [11]). To determine the range of operation (where activity is stable), and to obtain the small perturbations analysis a linearization of the system must be performed to obtain its eigenvalues, for more details check the chapter on small perturbations in [12], and the application to this case in [11, 6, 5].

C Analytical expression of $r_0(t)$, *2D*

The following *2D* analysis has not been performed elsewhere to our knowledge. The idea is to follow the *1D* idea, and to find an expression for the first Fourier component, or the mean total activity and 3 other Fourier components needed to express the dynamics of the system. The first step is to assume an input function,

$$g(I(\vec{x},t)) = E(\vec{x},t) = D^2\left[1+\cos(y)\right]\left[1+\cos(x-vt)\right] + C \quad (34)$$

where D and C are parameters controlling the bump highest activity and the spontaneous activity level. If $x \in [-\pi,\pi]$ and $y \in [-\pi,\pi]$, then $E(\vec{x},t)$ defines a single bump moving from left to right ($v > 0$). For the kernel function we consider the same equation as in Eq. 10,

$$w(\vec{x},\vec{x}') = A\left[1+\cos(y'-y)\right]\left[J_0 + J_1\cos\left(x'-x+\beta\right)\right] \quad (35)$$

where the asymmetry is only in the x-axis, therefore we are considering only sequences moving on that axis. If $y \in [-\pi,\pi]$, then w is a decreasing function of y. As in *1D* we start by writing the dynamics of $m(\vec{x},t)$ in terms of quantities that are space independent (Fourier components). We know the definition of $r_0(t)$ in *2D* from Eq. 11, but we require also higher Fourier terms:

$$r_1(t) = \int_{-\pi}^{\pi}\int_{-\pi}^{\pi} m(\vec{x},t)e^{i(x-\psi(t))}(2\pi)^{-2}d\vec{x} \quad (36)$$

$$r_2(t) = \int_{-\pi}^{\pi}\int_{-\pi}^{\pi} m(\vec{x},t)e^{i(y-\Omega(t))}(2\pi)^{-2}d\vec{x} \quad (37)$$

$$r_3(t) = \int_{-\pi}^{\pi}\int_{-\pi}^{\pi} m(\vec{x},t)e^{i(x+y-\lambda_1(t))}(2\pi)^{-2}d\vec{x} \quad (38)$$

$$r_4(t) = \int_{-\pi}^{\pi}\int_{-\pi}^{\pi} m(\vec{x},t)e^{i(x-y-\lambda_2(t))}(2\pi)^{-2}d\vec{x} \quad (39)$$

where the unknown functions of time $\psi(t)$, $\Omega(t)$, $\lambda_1(t)$ and $\lambda_2(t)$ are such that they make each quantity real and positive. Before using the different Fourier components, we express the dynamics of m using Eq. 35,

$$\tau\frac{\partial m(\vec{x},t)}{\partial t}+m(\vec{x},t)=\Big[E(\vec{x},t)+AJ_0r_0(t)+\frac{AJ_0}{4\pi^2}\int\int cos(y'-y)m(\vec{x}',t)d\vec{x}'+ \frac{AJ_1}{4\pi^2}\int\int\cos(x'-x+\beta)m(\vec{x}',t)d\vec{x}'+\frac{AJ_1}{4\pi^2}\int\int cos(y'-y) \cos(x-x'+\beta)m(\vec{x}',t)d\vec{x}'\Big]^+ \tag{40}$$

All the right side of Eq. 40 is the total input, and the $[\quad]^+$ operator must be applied to it. Now, we rewrite the total input in the complex plane to simplify,

$$=\Big[E(\vec{x},t)+AJ_0r_0(t)+\frac{AJ_0e^{i(y-\Omega)}}{4\pi^2}\int\int e^{-i(y'-\Omega)}m(\vec{x}',t)\vec{x}'+\frac{AJ_1e^{-i(x-\beta-\psi)}}{4\pi^2} \int\int e^{i(x'-\psi)}m(\vec{x}',t)\vec{x}'+\frac{AJ_1}{2}\Big\{e^{i(x+y-\lambda_1+\beta)}\int\int\frac{e^{i(x'+y'-\lambda_1)}}{4\pi^2}m(\vec{x}',t)\vec{x}'+ e^{i(x-y-\lambda_2+\beta)}\int\int\frac{e^{i(x'-y'-\lambda_2)}}{4\pi^2}m(\vec{x}',t)\vec{x}'\Big\}\Big]^+ \tag{41}$$

where we omit the limits of the integrals as they are all either π or $-\pi$. Introducing now Eqs. 36, 37, 38 and 39 we can write more compactly Eq. 41 as,

$$=\Big[E(\vec{x},t)+AJ_0r_0(t)+AJ_0e^{-i(y-\Omega)}r_2(t)+AJ_1e^{-i(x-\beta-\psi)}r_1(t)+ \frac{AJ_1}{2}\Big\{e^{i(x+y-\lambda_1+\beta)}r_3(t)+e^{i(x-y-\lambda_2+\beta)}r_4(t)\Big\}\Big]^+ \tag{42}$$

In the complex plane, we will impose a total input with the shape of a single bump moving along the x direction. This implies that the modes associated to $x+y$ and $x-y$ must have the same coefficients, *i.e.* $r_3=r_4$ and $\lambda=\lambda_1=\lambda_2$. Using this and Eq. 34, the total input can be written as:

$$=\Big[I_0(t)+e^{i(x-\phi)}I_1(t)+e^{-i(y-\Omega)}I_2(t)+I_3(t)e^{-i\alpha}(e^{i(x+y)}+e^{i(x+y)})\Big]^+ \tag{43}$$

where $I_0(t)=AJ_0r_0(t)+C+D^2$, $I_1(t)=AJ_1e^{i(\beta-\psi+\phi)}r_1(t)+D^2e^{i(\phi-vt)}$, $AJ_0e^{-i\Omega}r_2(t)+D^2$ and $I_3(t)=(AJ_1e^{i(\beta-\lambda+\alpha)}r_3(t)+D^2e^{i(\alpha-vt)})/2$. In Eq. 43 we have also introduced two variables $\phi(t)$ and $\alpha(t)$, that are used later to ensure a single bump. Eq. 43 can be simplified into Eq. 44 using the identity $e^{i(x+y)}+e^{i(x-y)}=2\cos(y)e^{ix}$.

$$=\Big[I_0(t)+e^{i(x-\phi)}I_1(t)+e^{-i(y-\Omega)}I_2(t)+I_3(t)2\cos(y)e^{i(x-\alpha)}\Big]^+ \tag{44}$$

As in the *1D* case, we want a single bump for the total input and we impose $I_1(t)$ to be real and to ensure that we introduce $\phi(t)$ similarly for $I_3(t)$ and $\alpha(t)$. Also in *2D*, we want $\Omega(t) = 0$ as the total input should remain static in the y-axis. Furthermore, the total input expressed in Eq. 44 requires $\psi(t) = \lambda(t)$ and $\phi(t) = \alpha(t)$ to be a single bump, which implies $r_1(t) = r_3(t)$, otherwise two bumps could exist. Using these assumptions, and taking back Eq. 44 into the Real domain, we obtain,

$$= [I_0(t) + I_1(t)\cos(x-\phi) + I_2(t)\cos(y) + I_1(t)\cos(y)\cos(x-\phi)]^+ \tag{45}$$

There is one aspect of Eq. 45 that requires to be addressed; to have a symmetric single bump we must have $I_1(t) = I_2(t)$ or, using both definitions, $A(J_0 r_2 - J_1 r_1 \cos(\beta+\psi+\phi)) = D^2(\cos(\phi - vt) - 1)$, in other words $r_1(t) \propto r_2(t)$. This property will help to obtain $r_0(t)$. Using this assumption Eq. 45 can be simplified into Eq. 46

$$= [I_0(t) + I_1(t)\cos(x-\phi) + I_1(t)\cos(y) + I_1(t)\cos(y)\cos(x-\phi)]^+ \tag{46}$$

The total input is a single bump in *2D* of width x_c and height y_c, using the symmetry of the total input we get $x_c = y_c$. At point (x_c, x_c) we know that the total input is zero or $I_0 = -I_1(2\cos(x_c) + \cos(x_c)^2)$, allowing us to write:

$$= I_1(t)\left[\cos(x-\phi) + \cos(y) + \cos(y)\cos(x-\phi) - 2\cos(x_c) - \cos(x_c)\cos(x_c)\right]^+ \tag{47}$$

Eq. 47, implies that we can describe the dynamics of m in *2D*, using two Fourier components: $r_0(t)$ and $r_1(t)$ as in the *1D* case, and integrating Eq. 47 by $\int\int d\vec{x}'/(2\pi)^2$ and $\int\int e^{ix} d\vec{x}'/(2\pi)^2$:

$$\tau\frac{\partial r_0(t)}{\partial t} + r_0(t) = I_1(t) g_0(x_c) \tag{48}$$

$$\tau\frac{\partial r_1(t)}{\partial t} + r_1(t) = I_1(t) g_1(x_c)\cos(\phi - \psi) \tag{49}$$

$$\tau r_1(t)\frac{\partial \psi(t)}{\partial t} = I_1(t) g_1(x_c)\sin(\phi - \psi) \tag{50}$$

where the only differences with the *1D* case are functions g_0 and g_1 that we define in Eq. 51 and Eq. 52.

$$g_0(x_c) = \frac{1}{\pi^2}\int_0^{x_c}\int_0^{l(x_c)} (\cos(x) + \cos(y) + \cos(y)\cos(x) - K(x_c))d\vec{x} \tag{51}$$

$$g_1(x_c) = \frac{1}{\pi^2}\int_0^{x_c}\int_0^{l(x_c)} (\cos(x) + \cos(y) + \cos(y)\cos(x) - K(x_c))\cos(x)d\vec{x} \tag{52}$$

using $K(x_c) = 2\cos(x_c) + \cos(x_c)^2$. The function to be integrated in Eq. 51 represents a bump, with the same form in the four quadrants (symmetry). More precisely $l(x_c) = \arccos[(K(x_c) - \cos(y))/(1 + \cos(y)]$.

The set of equations 48, 49 and 50 describes the dynamics of $m(\vec{x},t)$ and as we want a stable activity, we impose $dr_0/dt = 0$, $dr_1/dt = 0$ and $d\psi(t)/dt = v$ and use $I_0 = -I_1(2\cos(x_c) + \cos(x_c)\cos(x_c))$ to obtain,

$$r_0(x_c) = \frac{(C + D^2)}{-AJ_0 - K(x_c)/g_0(x_c)} \tag{53}$$

in this expression the main difference with the *1D* case is the term with g_0, yet $r_0(t)$ is still a decreasing function of x_c as g_0 is an increasing function of x_c and $K(x_c)$ is a decreasing function ($C, D, A > 0$ and $J_0 < 0$), making the term $K(x_c)/g_0(x_c)$ a decreasing function of x_c.

The final expression we have obtained in Eq. 53 for r_0 holds if several conditions can be verified, in particular if the total activity is one symmetric bump. We have not verified the symmetry condition, but numerically the shape of the bump seems symmetric. However, in the *2D* case is common to observe small asymmetries in the shape of the bump.

D Optimization of $v(\beta)$, *2D*

In this appendix we derive the expression of the asymmetry parameter β in terms of the input speed v, to maximize the total activity r_0. Considering the expression for the total activity for the population r_0 of units m obtained in Appendix C for the *2D* case, this is equivalent to minimize x_c (the size of the activity bump). To impose this we use the constraint of a single bump for the total input of activity,

$$AJ_1 \sin(\beta - \psi + \phi)r_1 + D^2 \sin(\phi - vt) = 0 \tag{54}$$

Using the expression we derived for r_1 in the *2D* case (analogous to r_0 in Eq. 53) in Eq. 54, we can derive and expression for the existence of the stable solution,

$$S' = \frac{J_0 g_0(x_c) + K(x_c)}{\sqrt{J_1^2 g_1^2 \cos(\Delta)^2 - 2J_1 g_1 \cos(\Delta)\cos(\Delta + \beta) + 1}} \tag{55}$$

As in *1D* this expression can be checked numerically for a given set of parameters as we do not know x_c. Yet, deriving by $\partial/\partial v$ and imposing $\partial x_c/\partial v|_{v=v_m} = 0$ we can find an expression for $v_m(\beta)$,

$$v_m = \frac{J_1 g_1(x_c) - 2\cos(\beta) \pm \sqrt{J_1^2 g_1^2(x_c) - 4J_1 g_1(x_c)\cos(\beta) + 4}}{2\tau \sin(\beta)} \tag{56}$$

where there is still a dependency with x_c expressed by $g_1(x_c)$. Checking the expression and choosing the appropriate solution $(-)$, we obtain the low $x_c \to \infty$ (or $J_1 g_1 \to \frac{1}{cos(\beta)}$) and high contrast $x_c \to 0$ (or $g_1 \to 0$) limits, re-obtaining the known solution from the literature in the *1D* for the first case, and the expression we obtained in Appendix B for the high contrast expression,

$$v_m(\beta) = \frac{1 - \cos(\beta)}{\tau \sin(\beta)} \tag{57}$$

E Numerical Simulations

The numerical simulation of Eq. 1 and Eq. 3 have been performed with the RK4 method (Runge-Kutta 4th order). Here we will give the parameters and discretization of the simulation in the *1D* case, an only the extra parameters for the *2D* case. First, we rewrite Eq. 1 as,

$$\frac{\partial m(k,t)}{\partial t} = \frac{1}{\tau}(-m(k,t)+\left[\sum_{k'} w(k'-k)m(k',t)dk + C\left[1-\epsilon+\epsilon\cos(k-vt)\right] - T\right]^{+}) \tag{58}$$

where we applied directly the RK4 method, see [13], to discretize the integral we use a trapezoidal rule. Here we use the parameters: $dt = 0.1$, $dk = 2\pi/60$, $\tau = .15$, $J_0 = -9.8$, $J_1 = 13.5$, $C = 5$, $T = 4.9$ and the total activity was computed over 1000 iterations. For the *2D* case the parameters where: $A = .2$, $dk = 2\pi/21$ and the total activity was computed over 100 iterations. Numerically, the system is quite robust to simulate and we where able to obtain similar results using the Euler method. Both implementations where performed in Matlab and are available from the first author website.

References

[1] J. G. Taylor. Neural "bubble" dynamics in two dimensions: foundations. *Biological Cybernetics*, 80(6):393–409, 1999.

[2] Vladimir S. Cherkassky and Filip Mulier. *Learning from Data: Concepts, Theory, and Methods*. John Wiley & Sons, Inc., New York, NY, USA, 1998.

[3] Martin A. Giese and Tomaso Poggio. Neural mechanisms for the recognition of biological movements. *Nature Reviews Neuroscience*, 4:179–192, 2003.

[4] Antonino Casile and Martin A. Giese. Critical features for the recognition of biological motion. *J. Vis.*, 5(4):348–360, 4 2005.

[5] Xiaohui Xie and Martin A. Giese. Nonlinear dynamics of direction-selective recurrent neural media. *Phys Rev E Stat Nonlin Soft Matter Phys*, 65:051904, May 2002.

[6] David Hansel and Haim Sompolinsky. *Methods in Neuronal Modeling: From synapses to networks*, chapter Modeling Feature Selectivity in Local Cortical Circuits. MIT Press, Cambridge, MA, USA, 1998.

[7] Ivan Laptev. On space-time interest points. *Int. J. Comput. Vision*, 64:107–123, September 2005.

[8] Shimon Edelman. *Representation and recognition in vision*. MIT Press, Cambridge, MA, USA, 1999.

[9] Frédéric Alexandre, Jérémy Fix, Axel Hutt, Nicolas P. Rougier, and Thierry Viéville. On practical neural field parameters adjustment. In *Deuxième conférence française de Neurosciences Computationnelles, "Neurocomp08"*, Marseille France, 10 2008. ISBN : 978-2-9532965-0-1.

[10] Lucian Alecu and Hervé Frezza-Buet. Application-driven parameter tuning methodology for dynamic neural field equations. In *Neural Information Processing, ICONIP'09 Proceedings, Part I*, volume 5863/2009 of *Lecture Notes in Computer Science*, pages 135–142, Bangkok (Thailand), 2009. Springer Berlin / Heidelberg.

[11] Stefanos E. Folias and Paul C. Bressloff. Stimulus-locked traveling waves and breathers in an excitatory neural network. *SIAM Journal of Applied Mathematics*, 65(6):2067–2092, 2005.

[12] Herbert Goldstein and Charles P. Poole. *Classical Mechanics*. Addison Wesley, June 2001.

[13] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical recipes in C (2nd ed.): the art of scientific computing*. Cambridge University Press, New York, NY, USA, 1992.

Centre de recherche INRIA Nancy – Grand Est
LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)

Centre de recherche INRIA Bordeaux – Sud Ouest : Domaine Universitaire - 351, cours de la Libération - 33405 Talence Cedex
Centre de recherche INRIA Grenoble – Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier
Centre de recherche INRIA Lille – Nord Europe : Parc Scientifique de la Haute Borne - 40, avenue Halley - 59650 Villeneuve d'Ascq
Centre de recherche INRIA Paris – Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex
Centre de recherche INRIA Rennes – Bretagne Atlantique : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex
Centre de recherche INRIA Saclay – Île-de-France : Parc Orsay Université - ZAC des Vignes : 4, rue Jacques Monod - 91893 Orsay Cedex
Centre de recherche INRIA Sophia Antipolis – Méditerranée : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399