



A Long Journey into Reproducible Computational Neuroscience

Meropi Topalidou, Arthur Leblois, Thomas Boraud, Nicolas P. Rougier

► To cite this version:

Meropi Topalidou, Arthur Leblois, Thomas Boraud, Nicolas P. Rougier. A Long Journey into Reproducible Computational Neuroscience. 2014. hal-01109483

HAL Id: hal-01109483

<https://inria.hal.science/hal-01109483>

Preprint submitted on 26 Jan 2015

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.

A Long Journey into Reproducible Computational Neuroscience

Meropi Topalidou^{1,2,3,4}, Arthur Leblois⁵, Thomas Boraud^{2,3}, and Nicolas P. Rougier^{1,2,3,4,*}

¹INRIA Bordeaux Sud-Ouest, Bordeaux, France

²Institut des Maladies Neurodégénératives,
Université Bordeaux-Segalen, UMR 5293, Bordeaux, France

³Institut des Maladies Neurodégénératives,
Centre National de la Recherche Scientifique, UMR 5293, Bordeaux, France

⁴LaBRI, Université de Bordeaux, Institut Polytechnique de Bordeaux,
Centre National de la Recherche Scientifique, UMR 5800, Talence, France

⁵Centre de Neurophysique, Physiologie et Pathologies, Université Paris Descartes,
Centre National de la Recherche Scientifique, UMR 8119, Paris, France

*Corresponding author: Nicolas.Rougier@inria.fr

Abstract Computational neuroscience, a relatively recent field, has gained fast ground and modelling is now widely used to better understand individual and collective neuronal dynamics, and to propose new functions relying on neural substrate. While the development of a model is initially tightly linked to the specific question asked by a given group of researchers, further development of the model in different contexts is often possible and desired. When such further development relies on new players, the continuity of the work requires proper validation, reproduction and sharing of the models original equations and code. However, as they are published today, computational neuroscience papers rarely include the sufficient material for the reproduction and sharing of the underlying model as a whole. Here, we aim at showing the full extent of the problem, as well as state-of-the-art solutions, through the detailed story of the reproduction of a computational modelling study by [Guthrie et al. \(2013\)](#) investigating the dynamics of basal ganglia circuits and their function in multiple level action selection. In collaboration with the authors of the original work, we first explain the difficulties encountered during the reproduction and validation of the initial model and results. These difficulties led us to completely rewrite the model enforcing best software practices, relying on previous attempts to provide a common framework for reproducible computational science and software sustainability. We hereby detail these practices in the face of our practical example: the reproduction of the results from [Guthrie et al. \(2013\)](#). In particular, these practices include: (i) a template for formal description of the model in a single table, (ii) a public repository for shared software a proper version control, and (iii) an easy interface to run the underlying code and reproduce figures. We finally propose new formats for communicating results allowing the replay of a code while reading a computational study, in order to get a deeper understanding of the concepts being introduced.

Author Summary Computational sciences, such as bioinformatics, computational biology or computational neuroscience, are gaining fast ground in modern Life Sciences and new discoveries can be made thanks to computational models or numerical simulations and analysis. The picture is not all bright however. The *computational* part in computational sciences implies the use of computers, operating systems, tools, frameworks, libraries and data. This leads to such a large number of combinations (taking into account the version for each components) that the chances to have the exact same configuration as one of your colleague are nearly zero. This draws consequences in our respective computational approaches in order to make sure models and simulations can be actually and faithfully reproduced. If reproducibility is the hallmark of Science, computational sciences seems to be still in their infancy in this domain, even though things have started to improve. This article highlights the extent of the problem based on the reproduction of a model in computational neuroscience and show how to circumvent it using recommended practices.

Introduction

During the last few years, there has been an increased interest in reproducible computational science as shown by the flourishing literature on the topic (Peng, 2011; Sandve et al., 2013; Osborne et al., 2014; Stodden et al., 2014), on good software practices (Nordlie et al., 2009; Donoho, 2011; Delescluse et al., 2012; Wilson et al., 2014; Goble, 2014) and the newly created websites to ensure software sustainability (recomputation.org, runmycode.org, etc.). These both illustrate the extent of the problem and the paradoxical nature of the situation. While computer science offers a large set of tools for prototyping, writing, running, testing, validating, sharing and reproducing results, computational neuroscience still lags behind. In the best case, authors may provide sources of their model as a compressed archive and feel confident they contributed to reproducible computational neuroscience. But this is not exactly true. Buckheit and Donoho (1995) explained almost 20 years ago that, *an article about computational result is advertising, not scholarship. The actual scholarship is the full software environment, code and data, that produced the result.* Anyone that ever open such compressed archive will plainly agree with this quote. How things went so bad? Part of the answer may be the ever growing place of software in modern Science, up to the point where new research domains have emerged, mirroring their traditional counterpart with the computational adjective (computational biology, computational neuroscience, computational geometry, etc.). Does this mean all of a sudden everybody can write software? Unfortunately no. If more than 50% of scientists admit to write software according to a survey by Hannay et al. (2009), it is very unlikely that all of these people received proper training to do so. This results in software of very different quality and accessibility, but more importantly, this may also result in incorrect software. And if your software is incorrect, so will be your science (Merali, 2010). This article aims at showing the full extent of the problem through the detailed story of the reproduction of a computational model. It has been written in collaboration with the authors of the original work, that allowed a full access to their archives and code and provides all the missing information that grounded the research.

In a previous modeling study, Leblois et al. (2006) demonstrated an action selection mechanism in cortico-basal ganglia loops based on a competition between the positive feedback, direct pathway through the striatum and the negative feedback, hyperdirect pathway through the subthalamic nucleus. In Guthrie et al. (2013), authors investigated further how multiple level action selection could be performed by the basal ganglia, and the model has been extended in a manner consistent with known anatomy and electro-physiology of the basal ganglia in the monkey. The model is quite complex, but such is the basal ganglia. Unfortunately, the information provided by the latter article was not sufficient to allow for the direct reproduction of the model. We explained in this article the precise difficulties we encountered and the reasons that lead us to a complete rewrite of the model, trying to enforce best software practices. Following good software practices, we were ultimately able to reproduce original results, confirming the correctness of the original implementation of the model. But we lost nearly three months in the process, while this should have been a routine task.

Models

Original model

The initial version of the model (Leblois et al., 2006) was studied through analytical calculation (in a reduced model of the network) and through numerical simulations (for the complete model including random connectivity, heterogeneities and synaptic noise). The simulation programs were written in C code (by A. Leblois), and, for some exploratory simulations, in Fortran (by D. Hansel). All the code used to run simulations for the results and figures of (Leblois et al., 2006) was written in C. At that stage, analytical calculation and simulations of the model could be reproduced simply from the equation and parameters provided in the paper. The task would be relatively easy for the reduced model, but may take quite some time for the extended model given the large number of parameters required to define network connectivity,

neuronal properties and inputs. The C code underlying the simulations of this complete model was not written to be easily shared (no versioning, no public repository), and providing the raw code would therefore be of partial help only. The later version of the model Guthrie et al. (2013) was developed in a collaboration between authors of the original version (A. Leblois and T. Boraud) and a new post-doc (M. Guthrie). The latter decided to use a graphical user interface (Delphi) to develop further the model, most likely due to his specific coding history. This methodological decision had a rather negative impact on the further development of the model: it was not possible for A. Leblois, who developed the initial version of the model, to run the code and therefore he had limited access to the behavior of the new model as the architecture complexity was incremented. It now turns out that this approach also impedes the reproducibility of the model by other scientist. This was precisely the case for M. Topalidou and N. Rougier when they decided to reproduce results of the article and soon realized that the information provided by the article was not sufficient to reproduce the results.

Literal description

The literal description of the model is primarily addressed to a neuroscientist audience (it was published in Journal of Neurophysiology) and assumes implicitly that readers are familiar with both neurophysiology and neuroanatomy of the basal ganglia. For example, the detailed description of the nuclei is absent and well know facts, such as the nature (excitatory or inhibitory) of this or that projection, are not mentioned. While this might be obvious for experts, such absence makes the reproduction task more difficult for non experts, even if it is possible to somehow *deduce* this knowledge from various hints in the article. Overall, the model is described quite precisely, with a lot of important information, but still, some information are ambiguous or erroneous and some others are just missing. We report in this section the description of the model as it has been proposed in the original article. We use gray boxes to indicate *in extenso* citation of the original material.

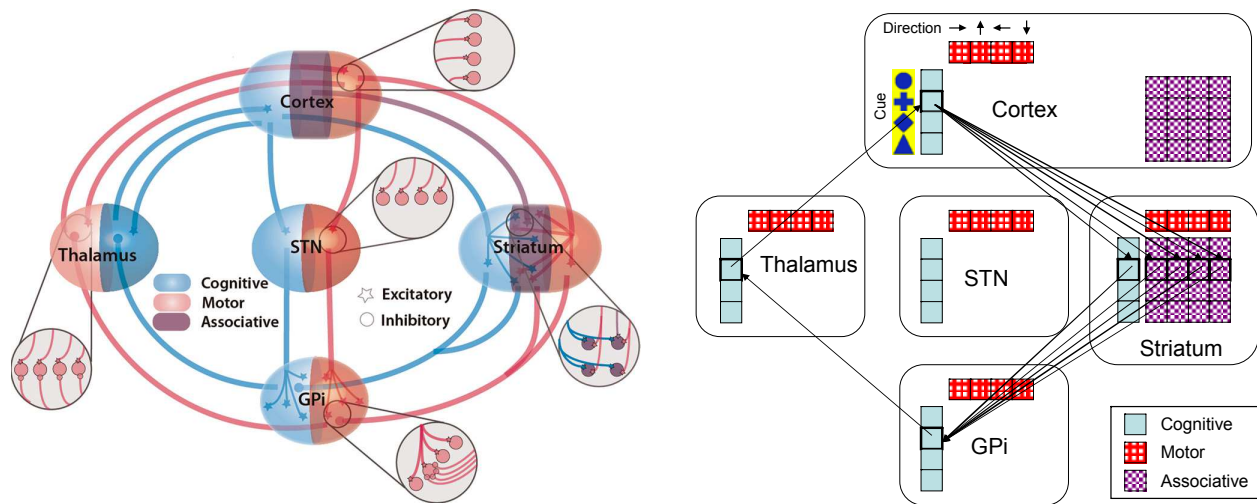


Figure 1: Architecture of basal ganglia model (left) and partial architecture of action selection model (right). Reproduced with authors permission.

Ambiguous and erroneous information Throughout the whole article, the different parts of the model are described using *neuron*, *ensemble* or *unit*. This ambiguity is related to the fact that there are two possible interpretations of the rate model. Either it is used to represent a neuronal population, each firing rate variable being associated to the average activity over a large pool of neurons. Or it can be used to represent the output (formally, the slow synaptic output) of a given neuron. In the case where connectivity between populations is all-to-all, all neurons behave the same in one population, and the single variable represent

either the average firing of the population or the activity of any of these neurons. In the original article, ensemble is defined as a set of co-activated neurons but units seems to be reserved for *striatal units* which suggest striatal elements might be represented by a single neuron while other elements may need several neurons. From the various figure describing the architecture of the model, it happens eventually an ensemble is represented by a single equation.

No units are given for the different constants and reader has to guess the unit from the nature of the variable. While this poses no problem for most variables and constants, there are some ambiguous situations. For example, time constant *tau* has been set to a value of 10. Since this is a time constant used for synaptic decay, we can deduce the actual value is 10 *milliseconds* or 0.01 *second*. However, this is far from evident by looking at the sources because the actual *dt* variable has been set to 1*ms* and removed from equations (implicit use of the forward Euler method).

The initialization of weights are defined in two different parts of the paper. First on page 3030 (second column):

Weights were initialized to a Gaussian distribution with a mean of 0.5 and a SD of 0.005 at the start of each simulation...

then on page 3031 in the caption of figure 4.

All synaptic weights were initialized to 0.5.

In fact, both definitions are right but do not address the same projections. Cortico-striatal synaptic weights use Gaussian distribution while all other weights are set to 1.0.

Finally, the Boltzmann equation according to the paper is:

$$m_{out} = V_{min} \cdot \left(\frac{V_{max} - V_{min}}{1 + e^{\frac{V_h - m_{in}}{V_c}}} \right)$$

This definition is wrong according to the proper Boltzmann equation that use a $+$ instead of a \cdot . The problem is even worse, because the article equation, when used with parameters given in table 3 of the original article, is quite similar to the actual Boltzmann equation. This makes even more difficult to spot the error and only the change of parameters clearly indicate the equation is wrong.

Undisclosed information As explained previously, the model refers to two pathways, one direct and one hyperdirect, but authors never explained what nuclei are involved or how they interact. For example, it is not mentioned if GPi sends inhibitory or excitatory inputs to thalamus. The only reference exists in the caption of the first figure. Furthermore, the last sentence of the same caption and table 2 indicate that cortex sends input to the thalamus but the gains used in the source code, are different from the article.

The learning rule includes a V_i term which refers to *the value of cue i*. Knowing that each cue is associated with a reward probability, it is legitimate at this stage to assimilate the two of them. However, it appears this value is actually referring to the state value as defined in reinforcement learning [Sutton and Barto \(1998\)](#). This was far from obvious since these values were not defined beforehand and their initial state is reported nowhere. Only the knowledge of reinforcement learning methods helped us setting their initial state to 0.5.

Source description

Because we were unable to reproduce the model using article description, we proceeded with studying the sources of the model that we requested from the authors. These sources are not available elsewhere and we were lucky enough that authors still have a copy around. However, studying these sources, to try to understand how the model works, has been a very demanding task. The main reason is that the main project file is 6000 lines long (using Pascal programming language) and mix the graphical user code with the actual model simulation. Furthermore, in order to compile the model, we needed a Windows system, the Delphi 7 personal edition compiler suite (which is no more available from the editor) and various extra components that were used for the graphical user interface. Even if we were able to gather most of these components, we were not able to find all of them and consequently, we did not manage to compile the model. However, having these sources was still very useful because they allowed us to check for the implementation of this or that equation. This is, for example, how we spotted the error on the Boltzmann equation and the initialization of weights. It also helped us to find out which set of parameters were used since the sources are accompanied by several configuration files (quite ill-named `default.ini`, `default2.ini`, etc.).

Executable

We had to turn to the actual executable and run it on a combination of Virtual Box and a 32 bits Windows 7 even though the original model has been developed using Windows XP, but we did not encounter any compatibility issue. This binary executable came as a great help in understanding how the model actually works (see figure 2) and allowed us to check for the activity of other structures and to compare the respective implementations (original and our version). Unfortunately, such executable is quite limited in what you can actually experiment, because it is dedicated to the main experimental paradigm. If we were able to tweak the various parameters to check for the activity of this or that structure, we could not, for example, experience lesion or new connectivity patterns. In other words, we could not escape the main experimental paradigm that is *hard-coded* through the interface.

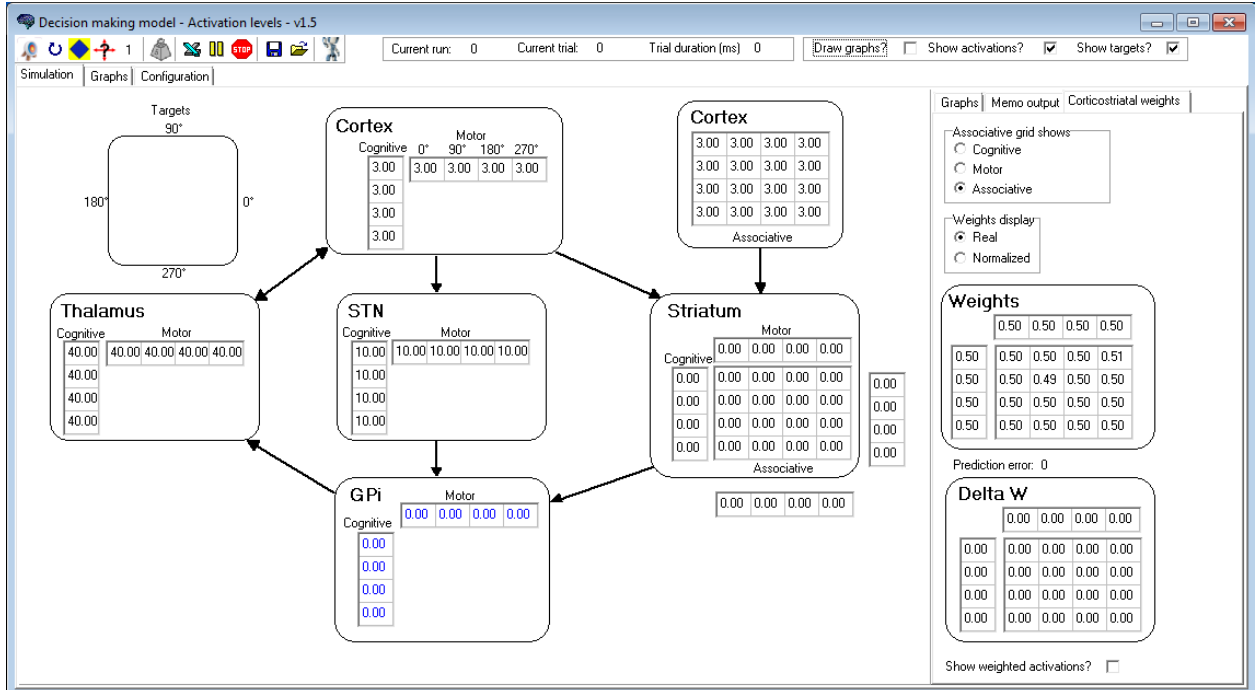


Figure 2: The executable of the original model provides a graphical user interface that allow to re-run the article experiment and allows to tweak different parameters.

Author’s interview

Ultimately, we were able to reach one of the original author (T. Boraud) in order to talk with him about the model. This took the form of an interview and we had prepared a series of questions to be answered such that we could reproduce the model. This was the final stage of our journey. We finally had enough material to start the reproduction of the model.

Revamped model

Before diving in the redesign of the model, we needed first to gather relevant information into a formal description following guidance of Nordlie et al. (2009) and to redesign figures of the model following rules described in Rougier et al. (2014). All the sources (python scripts and notebooks) are available from <https://github.com/rougier/Neurosciences/tree/master/basal-ganglia/guthrie-et-al-2013>.

Description

The complete description of the model (architecture, connectivity and neuron model) is spread throughout the whole article in a variety of forms (prose, equations, and figure). This has been identified as a source of problems by Nordlie et al. (2009) and *detrimental to the computational neuroscience field*. Authors suggests instead good model description practices in order for a model to be reproduced easily. Most notably, they propose a general guideline and checklist for a model description, but also propose templates for tables describing the model. These tables allow both a precise and concise description of the model that are really useful if one wants to reproduce the model or check the actual implementation (provided model sources has been given) of the model. We gathered all available information (article & source) into a tabular description given in appendix A.

Figures

There are mainly two figures describing the model in the original article. One is offering a global view of the model, emphasizing the cognitive and motor loops with an artistic style (see left part of figure 1). The other one rather aims at specifying precisely what are the projections between the different nuclei and makes use of the model graphical user interface to do so. Since there is a lot of projections, this figure is actually split into 6 sub-figures using a semantic that is not described but can be guessed (parallel/divergent/convergent connections). We took a different approach for this latter figure and used standard boxes and arrows (circle/red/inhibitory, arrow/blue/excitatory) to indicate the existence and the nature of a projection while the exact parameters of each projection are given in the appendix A.

Language

The first step in the redesign of a model is to choose the proper modeling approach. Namely, one can use a simulator (e.g. Neuron Carnevale and Hines (2006), Emergent O’Reilly and Y.Munakata (2000)), a toolbox (e.g. Matlab, Mathematica), a library (e.g. Brian Goodman and Brette (2008), DANA Rougier and Fix (2012)) or a programming language (e.g. C, Python). Each approach has its pros and cons and the choice depends mainly on the expertise of the modeler and the modeling history of the team/lab. While a simulator offers many advantages (rapid prototyping, correctness of the model), a new model might not fit exactly the modeling paradigm. Toolboxes are more flexibles since they offer basic components that can be assembled together to make the model, but correctness of the model is no longer guaranteed and depends on the interaction of the different components. At a lower level, modeling libraries are very similar to toolboxes but offer a unified approach that might save a lot of troubles. For example, the unit checking system of Brian ensures that any equation is physically menaningful and hence saves a lot of time during the design of new models. Last, but not least, one can choose to directly program the model from scratch, benefiting

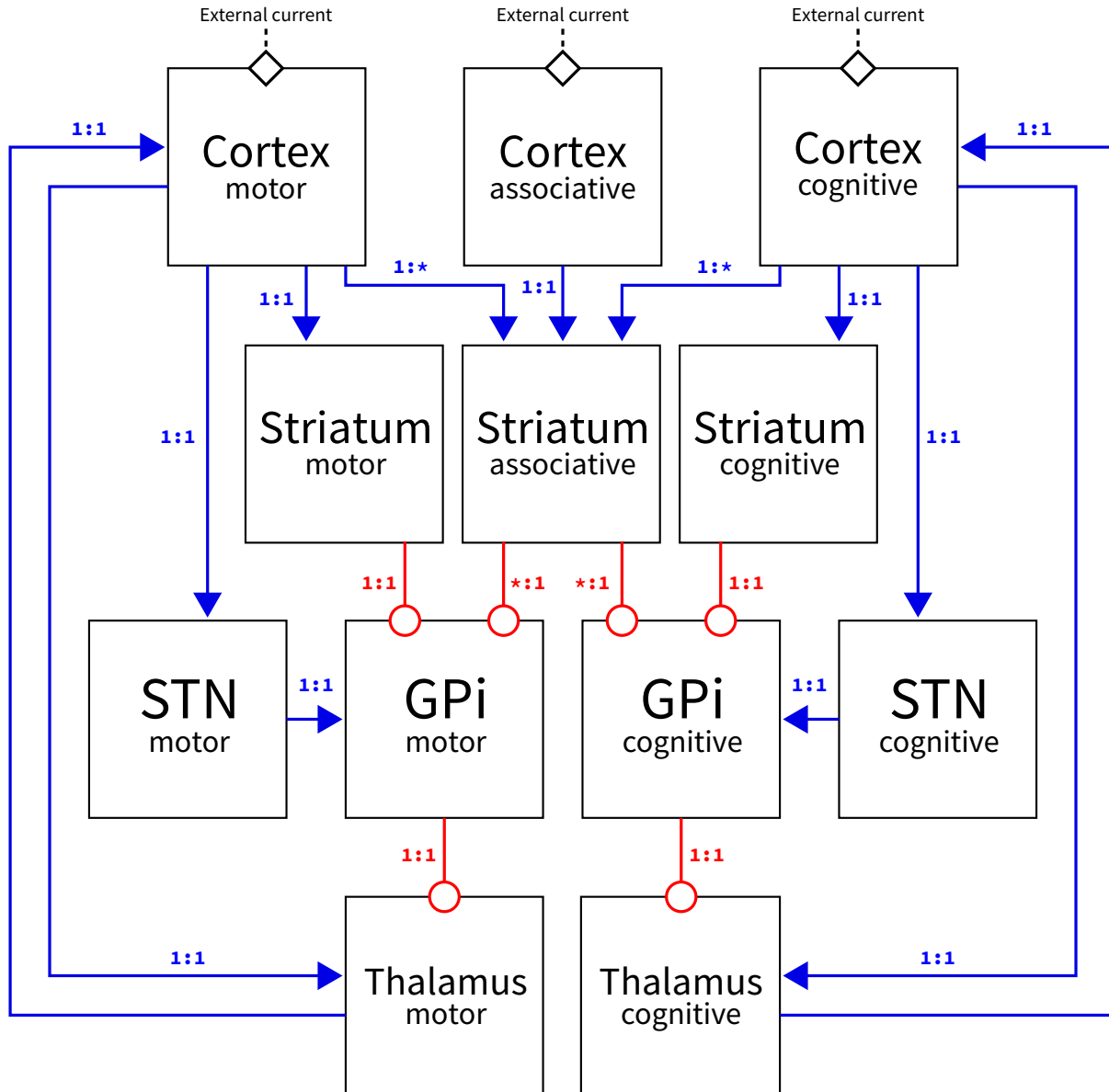


Figure 3: The figure describing the model architecture has been redesigned from the ground up in order to have a unique figure showing all excitatory and inhibitory connections at once as well as the external current insertion points. The color of a connection indicates its excitatory (blue) or inhibitory (red) nature and a label indicates if it is a parallel (1:1), convergent (1:*) or divergent (*:1). All these informations are also given in the tabular description of the model (Appendix A).

from a total freedom. Depending on the level of expertise of the designer, this can be a viable approach but is prone to many errors (e.g. inconsistent variable type, use of the default random generator, improper integration scheme, etc.). For the redesign of the model, we choose the DANA library (Rougier and Fix (2012), <https://github.com/rougier/dana>) which is a Python library heavily inspired from Brian but slanted towards mean-rate models. The main reason for such a choice is that last author is an expert in

Python and developed the DANA library which is open source, documented, tested and maintained. The use of this library allowed us to reduce the model to a 200 lines Python script.

Version control

In any experimental laboratory, notebooks help to keep track of the daily activity. Such notebooks are typically permanently bound (no flying pages) with page numbered (one can notice if a page is missing) and entries are timestamped and signed and such that an entry cannot be physically deleted. Even if such notebook are scarcely shared, they are used by the experimentalist who wrote them to get back some info they forgot. Legally, the notebook may attest that you did the experiment first, and it is used to attest that you use the right medications (for animale welfare). The digital equivalent for computational neuroscience is version control using tools such as subversion (<http://subversion.apache.org>) or git (<http://www.github.com>). These tools allow to keep track of changes, to know who did what/when, and to test new hypothesis very easily by, for example, creating branches. In the end, one can read the logs to know what were the different steps to build the model or what were the errors. However, such tools are restricted to sources. If your model is based on an external simulation software, you'll dependent on whether the software offers version control or not.

Public repository

As explained earlier, we managed to contact the original authors and they were able to send us the original source code. This was somehow expected since the model is barely two years old. However, for older models, it might be very unlikely that authors can still be contacted or that they still have a copy of the model, if this one has not be made public in the first place. There are many ways to do that, a simple compressed archive bundled with the supplementary materials on the journal website, a dedicated repository like, for example, the Neuron database available at <http://senselab.med.yale.edu/modeldb/>. For our revamped model, we decided from the start to make it available on github which offer a web-based framework for collaboration, code review, and code management and yields several advantages. First, all the repositories are public by default and anyone can look directly at a specific project as well as its commit history (however, this public nature is not always desirable). Furthermore, anyone can fork the project by hitting a single button and starts working on his own personal version of the model. Also, if some changes or bug fixed are worth to be added to the main repository, it is possible to issue a pull request that tell the original authors what is the purpose of this pull request and what would be changed (see figure 4). At this point, it is possible to discuss the pros and cons of this or that change before merging it into the main repository. Once this is done, the contribution will be recorded inside logs.


Interface




The IPython notebook (Pérez and Granger, 2007) is a web-based interactive computational environment where you can combine code execution, text, mathematics, plots and rich media into a single document. This allows easy collaboration with colleagues since the notebook can be exported to various formats (HTML, pdf, etc.) or directly shared online using a static version. One of the strength of the notebook is that it allows to mix code, figures and elaborated comments into a single document. For example, the equation governing a potential can be inserted in extenso (using L^AT_EX language) and the result is a self-contained document with code, results and explanations. The model has been made also available as a notebook such that it is a self-contained interactive document, mixing model definition and explanation.


Results

The final step in our journey has been to check if results could be reproduced faithfully. Due to the difficulty in the redesign of the model, this has been split in two phases. First, we checked if a single trial could be qualitatively reproduced in terms of dynamic (oscillations during settling) and decision (does a decision

< Learning #1


 **Open** mtopalid wants to merge 12 commits into rougier:master from mtopalid:learning

 Conversation 11  Commits 12  Files changed 2





mtopalid commented on Jun 2


Learning is working but the cognitive and motor decision is not always the same, even after 100 trials. Also, for 3 trials the weights for shape 4 goes to 0 and then up again if you put 123 as an input to the random function.



mtopalid added some commits on May 20


 Guthrie-model-learning.py is working on learning but the cognitive an... 0075176

 Delete model-learning.py 8e62516





rougier commented on Jun 2


If this is an IPython notebook, the file extension should be .ipynb




mtopalid added some commits on Jun 3

 Add ipython notebook for learning model c972595

 Merge branch 'learning' of https://github.com/mtopalid/Neurosciences ... 1c19399

 Removing some comments f659889



mtopalid commented on Jun 3

iPython Notebook for learning model added

Figure 4: Screenshot of the github interface showing a discussion on a pull request (<https://github.com/rougier/Neurosciences/pull/1>). Before merging modifications into the model, it is possible to discuss and comment to improve the proposed modification.

actually occurs in the cognitive and motor structures). We then proceeded to the learning task in order to reach asymptotic performances.

Single trial

A trial lasts for exactly three seconds, including a settling phase of 500ms with no external input. During this settling phase, there appear some characteristic damped oscillations that stabilizes just before the actual start of the trial (see figure 5). At time $t=500\text{ms}$, cues are presented to the model in the form of external activity that feed the cognitive and motor areas. The activity of the associative cortex receives proper external activity in order to disambiguate correspondence between shape and position (no binding problem).

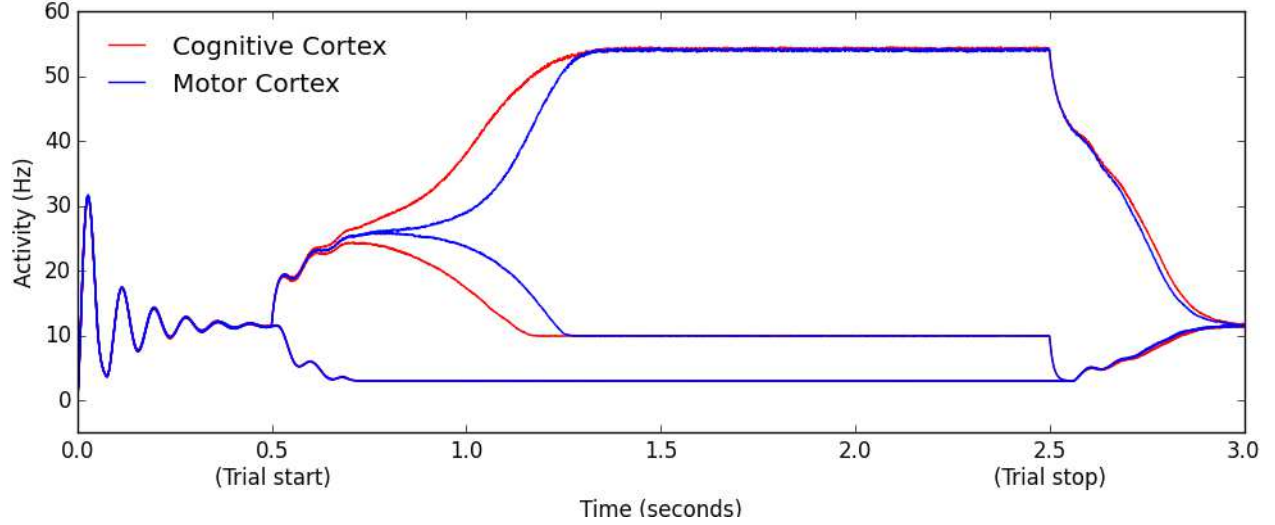


Figure 5: Activity in the cortical populations (blue for motor, red for cognitive) during a single trial.

Cognitive and motor neurons that are not fed by the external input decrease their activity to the resting potential while the fed ones increase their activity. After 750ms, the activity of the active neurons starts to diverge in both cognitive and motor groups. For a decision to be considered to have occurred, there must be a significant difference (40Hz) which happens around 1250ms for both cognitive and motor groups. The trial stops at 2500ms and the model can return to the resting state in 500 ms. The original article provided a single figure with the activity of both cortical cognitive and motor structures, although it was essential for us to check the activity of other structures during a single trial as shown on figure 7 in the appendix.

Learning

According to the original protocol, learning has been tested over 120 consecutive trials. However, there was missing information about the exact protocol and we assumed the following elements:

- We assumed an exact uniform target distribution (each target appears exactly 60 times). This has been implemented by shuffling an array holding all the targets pairs.
- We implemented an explicit resetting phase between trials to ensure we have no trailing activities from previous trial.
- Learning occurs at the end of the trial, if and only if a motor decision has been made by the model.
- Trials where no motor decision has occurred (difference of activity is less than 40Hz) have been discarded.

The learning protocol has been averaged over 250 simulations of 120 trials. Performance has been calculated as the ratio of correct motor decisions, independently of the cognitive decision. This means that if the model chose the wrong shape (cognitive decision) but the right move (motor decision), this results in reaching the right target. Since there is no way to disambiguate such double errors in the monkey, we assumed this should be counted as a correct answer. The asymptotic performance of the revamped model is approximately 97%, to be compared with the 96% performance of the original model.

Discussion

Computational neuroscience is a powerful ally in our quest to understand the brain. Even the most simple model can shed light on the role of this or that structure and propose new hypothesis concerning the overall brain organization. The model we studied and reproduced in this paper, while simple, is able to explain to some extents how the motor and the cognitive cortico-basal ganglia loops interact during decision making. However, any model in Science is doomed to be proved wrong or incomplete and replaced by a more accurate one (Box and Draper, 1987). In the meantime, for such replacement to happen, we have first to make sure it is actually reproducible such that it can be tested, evaluated, criticized and ultimately modified, replaced or even rejected. This is where the shoe pinches. If we cannot reproduce a model in the first place, we're doomed to re-invent the wheel again and again and we won't be able to build an incremental computational knowledge of the brain. In this short article, we tried to show the extent of the problem and the precise difficulties that arise when one want to reproduce a computational model using a singular example. However, from our own experience, this is not an isolated case, even though there are few cases where the model can be reproduced in a straightforward way as we did for Gurney et al. (2001) and Girard et al. (2008). We believe the revamped model we proposed allows any researcher in computational neuroscience to run it and obtain the exact same results as the ones introduced in this article. Some colleagues already have shown interest in the revamped model because they failed at reproducing the original one. Furthermore, the new description, as well as the new figures, allow anyone to rewrite the model using different language, tools or software. To obtain such reproducible results, we merely applied precepts that are now widely available in the literature Nordlie et al. (2009); Peng (2011); Osborne et al. (2014); Delescluse et al. (2012); Stodden et al. (2014).

From a broader perspective, this singular experience into reproducible neuroscience raises some questions about the whole publication process. While it has been true for a long time that article were actually printed, it is more and more common to download and read articles in electronic form only. In such context, does it really make sense any more to have supplementary materials as a separate document while they may carry critical information for the reproduction of the result ? We think instead, any supplementary materials carrying important information for the reproduction of the model should be attached with the main document. Furthermore, for the computational part, it is quite surprising that there is still no official journal source repository. A simple dedicated account on github (or any similar website such as sourceforge, gitlab, etc.) would be an extremely valuable resources for researchers as it would provide a unique entry point for any published model. It could even be hosted on the journal website since most repository offers deployment of enterprise edition. The next step in that direction would be to have online repository of virtual machines where model could be directly ran and modified from within a web browser. This is what is currently experienced on the recomputation.org website where the first recomputable experiments from ECAI 2014 have been made available using a combination of downloadable virtual machines (www.virtualbox.org) and vagrant environments (www.vagrantup.com). Of course, such virtual machines solution would prohibit the use of non free software and this could penalize a lot of authors. But such non-free software equally penalizes reviewers and readers, when it is necessary, for example, to buy several 500\$ toolboxes to run a single model.

Finally and given the quality of the new tools available today, it may be time to envisage new formats for communicating results. For example, interactive documents could allow to replay a simulation and modify parameters while reading the description of a model or simulation. The IPython notebook is a serious candidate in that direction and could soon become a new way to exchange knowledge. It has been recently highlighted on Nature (Shen, 2014) and it is already widely used for teaching. Furthermore, interactive books are now available at nbviewer.ipython.org that allow to tweak the parameters of a simulation in order to get a deeper understanding of the concepts being introduced. It is probably a matter of months before no one can tell the difference between a regular document (PDF) and an interactive one living in the browser.

Such new formats would definitely help authors, reviewers, readers and ultimately, Science as a whole.

A Long Journey into Reproducible Computational Neurosciences Research

Meropi Topalidou^{1,2,3}, Arthur Leblois⁴, Thomas Boraud³ and Nicolas P. Rougier^{1,2,3,*}

¹ INRIA Bordeaux Sud-Ouest, France

² LaBRI, UMR 5800 CNRS, Talence, France

³ Institute of Neurodegenerative Diseases, UMR 5293, Bordeaux, France

⁴ Laboratoire de Neurophysique et Physiologie, UMR 8119, Paris, France

* Corresponding author (Nicolas.Rougier@inria.fr)

Abstract In a previous modelling study, Leblois et al. (2006) demonstrated an action selection mechanism in cortico-basal ganglia loops based on competition between the positive feedback, direct pathway through the striatum and the negative feedback, hyperdirect pathway through the subthalamic nucleus. In Guthrie et al. (2013), authors investigated how multiple level action selection could be performed by the basal ganglia. To do this, the model has been extended in a manner consistent with known anatomy and electro-physiology in three main areas. Unfortunately, the information provided by the article were not sufficient to reproduce the model. If reproducibility is the hallmark of Science, non-reproducibility seems to be the hallmark of Computational Neurosciences. In that respect, Guthrie et al. (2013) is a prototypic case of such non-reproducible computational neuroscience research even though the proposed model gives a fair account of decision making in the basal ganglia complex. While trying to replicate results starting from the article description, we soon realised some information were undisclosed, some other were ambiguous and there were also some factual errors. Even after accessing the original sources (more than 6000 lines of Pascal), we were still unable to understand how the model worked. In the end, only the original material (a binary executable) and a direct contact with the authors allowed us to access the whole picture. After two months of intensive refactoring, we were finally able to replicate results using only 200 lines of Python. From this experience, which is unfortunately not an isolated case, we would like to share a simple message with the computational neuroscience community: designing computational models is not all about writing & running programs. If a model is to be reviewed, understood, used, replicated and integrated, it requires a minimal amount of coordinated efforts. Or the model will be soon forgotten, even by their own original designers.

Packages import

```
In [1]: %matplotlib inline
        from dana import *
        import matplotlib.pyplot as plt
```

Simulation parameters

```
In [2]: # Population size
        n = 4

        # Default trial duration
        duration = 3.0*second

        # Default Time resolution
        dt = 1.0*millisecond

        # Initialization of the random generator (reproductibility !)
        np.random.seed(1)
```

```
In [3]: # Threshold
        Cortex_h = -2.0
        Striatum_h = 0.0
        STN_h = -10.0
        GPi_h = 10.0
        Thalamus_h = -40.0
```

Figure 6: Screenshot of the IPython notebook implementing the model. Title and abstract are part of the notebook and have been written using markdown cell type.

Supporting Information

Model description

Notebook

Additional results

References

G.E.P. Box and N.R. Draper. *Empirical Model Building and Response Surfaces*. John Wiley & Sons, New York, 1987.

A Model Summary	
Populations	Twelve: Cortex (motor, associative & cognitive), Striatum (motor, associative & cognitive), GPi (motor & cognitive), STN (motor & cognitive), Thalamus (motor & cognitive)
Topology	–
Connectivity	one to one, one to many (divergent), many to one (convergent)
Neuron model	Dynamic rate model
Channel model	–
Synapse model	Linear synapse
Plasticity	Reinforcement learning rule
Input	External current in cortical areas (motor, associative & cognitive)
Measurements	Firing rate

B Populations					
Name	Elements	Size	Threshold (h)	Noise	Initial state
Cortex motor	Linear neuron	1×4	-3	1.0%	0.0
Cortex cognitive	Linear neuron	4×1	-3	1.0%	0.0
Cortex associative	Linear neuron	4×4	-3	1.0%	0.0
Striatum motor	Sigmoidal neuron	1×4	0	0.1%	0.0
Striatum cognitive	Sigmoidal neuron	4×1	0	0.1%	0.0
Striatum associative	Sigmoidal neuron	4×4	0	0.1%	0.0
GPi motor	Linear neuron	1×4	+10	3.0%	0.0
GPi cognitive	Linear neuron	4×1	+10	3.0%	0.0
STN motor	Linear neuron	1×4	-10	0.1%	0.0
STN cognitive	Linear neuron	4×1	-10	0.1%	0.0
Thalamus motor	Linear neuron	1×4	-40	0.1%	0.0
Thalamus cognitive	Linear neuron	4×1	-40	0.1%	0.0
Values (V_i)	Scalar	4	–	–	0.5

C Connectivity					
Source	Target	Pattern	Weight (W)	Gain (G)	Plastic
Cortex motor	Thalamus motor	$(1, i) \rightarrow (1, i)$	1.0	0.4	No
Cortex cognitive	Thalamus cognitive	$(i, 1) \rightarrow (i, 1)$	1.0	0.4	No
Cortex motor	STN motor	$(1, i) \rightarrow (1, i)$	1.0	1.0	No
Cortex cognitive	STN cognitive	$(i, 1) \rightarrow (i, 1)$	1.0	1.0	No
Cortex motor	Striatum motor	$(1, i) \rightarrow (1, i)$	$\mathcal{N}(0.5, 0.005)$	1.0	No
Cortex cognitive	Striatum cognitive	$(i, 1) \rightarrow (i, 1)$	$\mathcal{N}(0.5, 0.005)$	1.0	Yes
Cortex motor	Striatum associative	$(1, i) \rightarrow (*, i)$	$\mathcal{N}(0.5, 0.005)$	0.2	No
Cortex cognitive	Striatum associative	$(i, 1) \rightarrow (i, *)$	$\mathcal{N}(0.5, 0.005)$	0.2	No
Cortex associative	Striatum associative	$(i, j) \rightarrow (i, j)$	$\mathcal{N}(0.5, 0.005)$	1.0	No
Thalamus motor	Cortex motor	$(1, i) \rightarrow (1, i)$	1.0	1.0	No
Thalamus cognitive	Cortex cognitive	$(i, 1) \rightarrow (i, 1)$	1.0	1.0	No
GPi motor	Thalamus motor	$(1, i) \rightarrow (1, i)$	1.0	-0.5	No
GPi cognitive	Thalamus cognitive	$(i, 1) \rightarrow (i, 1)$	1.0	-0.5	No
STN motor	GPi motor	$(1, i) \rightarrow (1, i)$	1.0	1.0	No
STN cognitive	GPi cognitive	$(i, 1) \rightarrow (i, 1)$	1.0	1.0	No
Striatum cognitive	GPi cognitive	$(i, 1) \rightarrow (i, 1)$	1.0	-2.0	No
Striatum motor	GPi motor	$(i, 1) \rightarrow (i, 1)$	1.0	-2.0	No
Striatum associative	GPi motor	$(*, i) \rightarrow (1, i)$	1.0	-2.0	No
Striatum associative	GPi cognitive	$(i, *) \rightarrow (i, 1)$	1.0	-2.0	No

D1 Neuron Model	
Name	Linear neuron
Type	Rate model
Membrane Potential	$\tau dV/dt = -V + I_{syn} + I_{ext} - h$ $U = max(V, 0)$
D2 Neuron Model	
Name	Sigmoidal neuron
Type	Rate model
Membrane Potential	$\tau dV/dt = -V + I_{syn} + I_{ext} - h$ $U = V_{min} - (V_{max} - V_{min}) / \left(1 + e^{\frac{V_h - V}{V_c}}\right)$
E Synapse	
Name	Linear synapse
Type	Weighted sum
Output	$I_{syn}^B = \sum_{A \in sources} (G_{A \rightarrow B} W_{A \rightarrow B} U_A)$
F Plasticity	
Name	Reinforcement learning
Type	Delta rule
Delta	$\Delta W_{A \rightarrow B} = \alpha \times PE \times U_B$ $PE = Reward - V_i$ $\alpha = 0.01$ if $PE < 0$ (LTD), $\alpha = 0.02$ if $PE > 0$ (LTP)
G Input	
Type	Cortical input
Description	A trial is preceded by a settling period (500ms) and followed by a reset period. At time $t = 0$, two shapes are presented in cortical cognitive area ($I_{ext} = 7$ at $\{i_1, i_2\}$) at two different locations in cortical motor area ($I_{ext} = 7$ at $\{j_1, j_2\}$) and the cortical associate area is updated accordingly ($I_{ext} = 7$ at $\{i_1, i_2\} \times \{j_1, j_2\}$).
Timing	<div><div>Trial start</div><div>Stimulus onset</div><div>Stimulus offset</div><div>Reset</div><div>-500ms</div><div>0</div><div>2500 ms</div><div>3000 ms</div></div>
H Measurements	
Site	Cortical areas
Data	Activity in cognitive and motor cortex Cortico-striatal weights
I Environment	
OS	OSX 10.9 (maverick)
Language	Python 2.7.6 (brew installation)
Libraries	Numpy 1.8.1 (pip installation) SciPy 0.13.3 (pip installation) IPython 1.2.1 (pip installation) Matplotlib 1.3.0 (pip installation) DANA 0.5.1 (pip installation)
Tools	Safari browser (native)

Table 1: A tabular description of the model following the prescription of Nordlie et al. (2009).

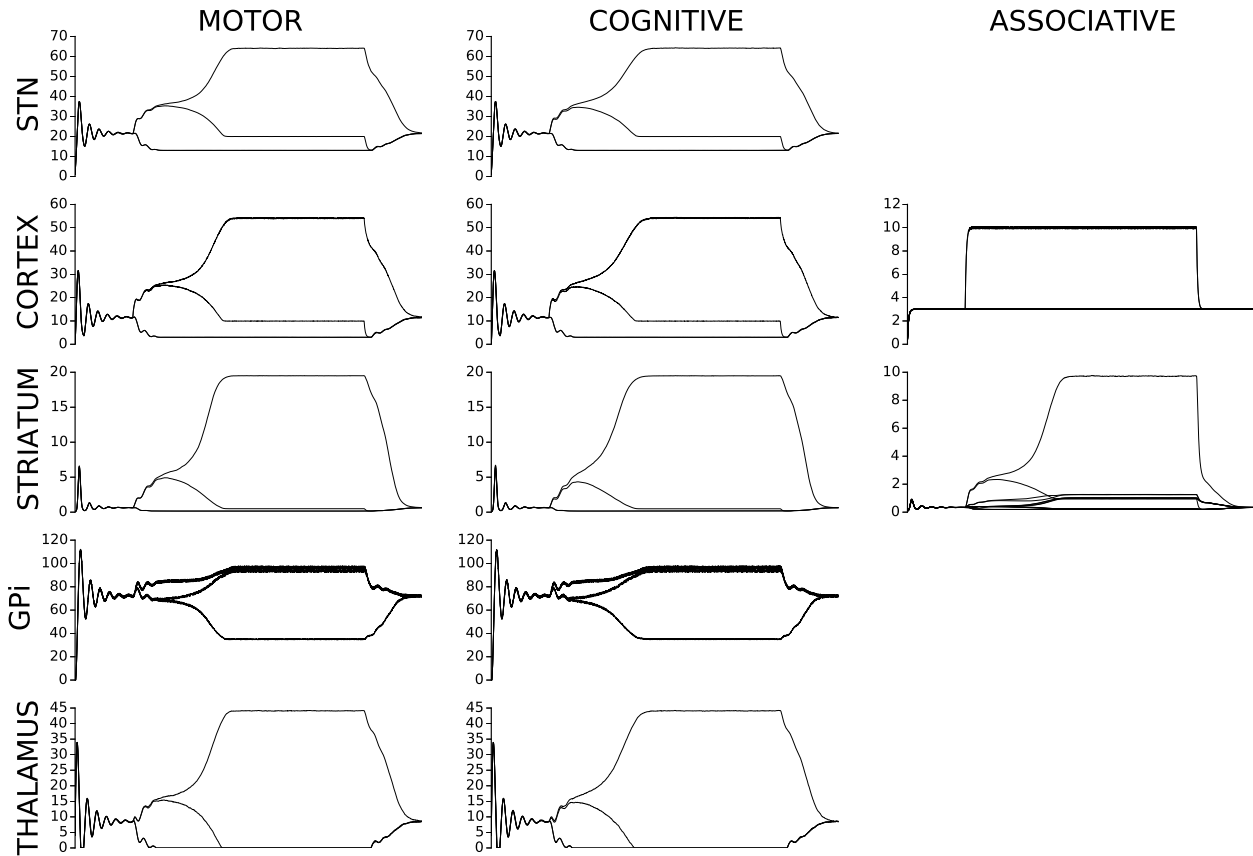


Figure 7: The activities of the non-cortical groups during a single trial were not provided in the original manuscript. However, we found these activities to be extremely useful for understanding the dynamic of the model. Knowing the overall shape of the different activities in the model, even without precise details, allows to quickly spot problems. For example, using these activities, we know what are the bounds for the activities in each area and as soon as one of our activity is out of bounds, we'll know something is wrong.

J. Buckheit and D.L. Donoho. *Wavelets and Statistics*, chapter Wavelab and reproducible research. Antoniadis, A., 1995.

N.T. Carnevale and M.L. Hines. *The NEURON Book*. Cambridge University Press, 2006.

Matthieu Delescluse, Romain Franconville, Sébastien Joucla, Tiffany Lieury, and Christophe Pouzat. Making neurophysiological data analysis reproducible. why and how ? *Journal of Physiology (Paris)*, 106(3–4):159–170, 2012.

D.L. Donoho. An invitation to reproducible computational research. *Biostatistics*, 11(3):385–388, 2011. doi: 10.1093/biostatistics/kxq028.

B. Girard, N. Tabareau, Q. Pham, A. Berthoz, and J. Slotine. Where neuroscience and dynamic system theory meet autonomous robotics : a contracting basal ganglia model for action selection. *Neural Networks*, 21(4):628–641, 2008.

C. Goble. Better software, better research. *IEEE Internet Computing*, 2014.

D. Goodman and R. Brette. Brian: a simulator for spiking neural networks in python. *Frontiers in Neuroinformatics*, 2(5):1–10, 2008. ISSN 1662-5196.

K. Gurney, T. Prescott, and P. Redgrave. A computational model of action selection in the basal ganglia. i. a new functional anatomy. *Biological cybernetics*, 84(6):401–410, 2001.

- M. Guthrie, A. Leblois, A. Garenne, and T. Boraud. Interaction between cognitive and motor cortico-basal ganglia loops during decision making: a computational study. *Journal of Neurophysiology*, 109:3025–3040, 2013.
- J.E. Hannay, H.P. Langtangen, C. MacLeod, D. Pfahl, J. Singer, and G. Wilson. How do scientists develop and use scientific software? In MIT Press, editor, *Software Engineering for Computational Science and Engineering*, 2009.
- A. Leblois, T. Boraud, W. Meissner, H. Bergman, and D. Hansel. Competition between feedback loops underlies normal and pathological dynamics in the basal ganglia. *Journal of Neurosciences*, 26:3567–3583, 2006.
- Z. Merali. Computational science: ... error ... why scientific programming does not compute. *Nature*, 467:775–777, 2010.
- E. Nordlie, M. Gewaltig, and H.E. Plesser. Towards reproducible descriptions of neuronal network models. *PLoS Computational Biology*, 5(8):e1000456, 2009. doi: 10.1371/journal.pcbi.1000456.
- R.C. O'Reilly and Y. Munakata. *Computational Explorations in Cognitive Neuroscience: Understanding the Mind by Simulating the Brain*. MIT Press, Cambridge, MA, USA, 2000.
- J.M. Osborne, M.O. Bernabeu, M. Bruna, B. Calderhead, J. Cooper, N. Dalchau, S. Dunn, A.G. Fletcher, R. Freeman, D. Groen, B. Knapp, G.J. McInerny, G.R. Mirams, J. Pitt-Francis, B. Sengupta, D.W. Wright, C.A. Yates, D.J. Gavaghan, S. Emmott, and C. Deane. Ten simple rules for effective computational research. *PLoS Computational Biology*, 10(3):e1003506, 2014. doi: 10.1371/journal.pcbi.1003506.
- R.D. Peng. Reproducible research in computational science. *Science*, 334(6060):1226–1227, 2011.
- Fernando Pérez and Brian E. Granger. IPython: a system for interactive scientific computing. *Computing in Science and Engineering*, 9(3):21–29, 2007. ISSN 1521-9615. doi: 10.1109/MCSE.2007.53.
- N.P. Rougier and J. Fix. Dana: Distributed numerical and adaptive modelling framework. *Network: Computation in Neural Systems*, 23(4), 2012.
- N.P. Rougier, M. Droettboom, and P.E. Bourne. Ten simple rules for better figure. *PLOS Computational Biology*, 10(9), 2014.
- G.K. Sandve, A. Nekrutenko, J. Taylor, and E. Hovig. Ten simple rules for reproducible computational research. *PLoS Computational Biology*, 9(10):e1003285, 2013. doi: doi:10.1371/journal.pcbi.1003285.
- Helen Shen. Interactive notebooks: Sharing the code. *Nature*, (515):151–152, 2014.
- V. Stodden, F. Leisch, and R.D. Peng, editors. *Implementing Reproducible Research*. Chapman & Hall/CRC, 2014.
- Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998.
- G. Wilson, D. A. Aruliah, C.T. Brown, N.P. Chue Hong, M. Davis, R.T. Guy, S.H.D. Haddock, K.D. Huff, I.M. Mitchell, M.D. Plumbley, B. Waugh, E.P. White, and P. Wilson. Best practices for scientific computing. *PLoS Biology*, 12(1), 2014.