Data in Brief 54 (2024) 110345

Contents lists available at ScienceDirect

Data in Brief

journal homepage: www.elsevier.com/locate/dib

Data Article

Simulated dataset on coordinated reset stimulation of homogeneous and inhomogeneous networks of excitatory leaky integrate-and-fire neurons with spike-timing-dependent plasticity

Justus A. Kromer*, Peter A. Tass

Department of Neurosurgery, Stanford University, Stanford, CA, United States of America

ARTICLE INFO

Article history:
Received 8 February 2024
Accepted 12 March 2024
Available online 20 March 2024

Dataset link: Simulated dataset on coordinated reset stimulation of homogeneous and inhomogeneous networks of excitatory leaky integrate-and-fire neurons with spike-timing-dependent plasticity (Original data)

Keywords:
Synchronization
Multistable networks
Leaky integrate-and-fire neurons
Spike-timing-dependent plasticity
Stimulation
Coordinated reset

ABSTRACT

We present simulated data on coordinated reset stimulation (CRS) of plastic neuronal networks. The neuronal network consists of excitatory leaky integrate-and-fire neurons and plasticity is implemented as spike-timing-dependent plasticity (STDP). A synchronized state with strong synaptic connectivity and a desynchronized state with weak synaptic connectivity coexist. CRS may drive the network from the synchronized state into a desynchronized state inducing long-lasting desynchronization effects that persist after cessation of stimulation. This is used to model brain stimulation-induced transitions between a pathological state, with abnormally strong neuronal synchrony, and a physiological state, e.g., in Parkinson's disease. During CRS, a sequence of stimuli is delivered to multiple stimulation sites – called CR sequence. We present simulated data for the analysis of long-lasting desynchronization effects of CRS with shuffled CR sequences versus non-shuffled CR sequences in which the order of stimulus deliveries to the sites remains unchanged throughout the entire stimulation period. Such data are presented for networks with homogeneous synaptic

DOI of original article: 10.1016/j.brs.2024.02.004
* Corresponding author.
E-mail address: jkromer@stanford.edu (J.A. Kromer).
Social media: @peter_tass (P.A. Tass)

https://doi.org/10.1016/j.dib.2024.110345

connectivity and networks with inhomogeneous synaptic connectivity. Homogeneous synaptic connectivity refers to a network in which the probability of a synaptic connection does not depend on the pre- and postsynaptic neurons' locations. In contrast, inhomogeneous synaptic connectivity refers to a network in which the probability of a synaptic connection depends on the neurons' locations. The presented neuronal network model was used to analyse the impact of the CR sequences and their shuffling on the long-lasting effects of CRS [1].

Specifications Table

Subject	Statistical and Nonlinear Physics, Mathematical Modelling.
Specific subject area	Computational modelling of long-lasting aftereffects of stimulation of plastic neuronal networks
Type of data	Methods, Simulated Data, Code, Figures
Data collection	Numerical simulations of the described neuronal network model were performed on Stanford's Sherlock Computing cluster.
Data source location	Data were collected at the Department of Neurosurgery, Stanford University, Stanford, California, United States of America
Data accessibility	Data are stored in a public repository on *Mendeley Data*. Repository name: Simulated dataset on coordinated reset stimulation of homogeneous and inhomogeneous networks of excitatory leaky integrate-and-fire neurons with spike-timing-dependent plasticity Data identification number: DOI: 10.17632/fmmr595pps.1 Direct URL to data: https://data.mendeley.com/datasets/fmmr595pps/1 Instructions for accessing these data: Data can be accessed using the provided URL.
Related research article	J.A. Kromer, P.A. Tass, Sequences and their shuffling may crucially impact coordinated reset stimulation – A theoretical study, Brain Stimul. 17 (2024) P194–P196.

1. Value of the Data

- These data present valuable additional information to support the findings presented in Ref. [1].
- The code provided in the repository https://data.mendeley.com/datasets/fmmr595pps/1 can be used to simulate long-term aftereffects of CRS in networks of excitatory leaky integrate-and-fire neurons with STDP with different types of synaptic connectivity. Long-lasting effects of CRS with different stimulation parameters can be simulated.
- The provided data can be used and enable to reproduce the simulated data shown in Ref. [1].

2. Background

Excessive neuronal synchrony accompanies several neurological disorders, including Parkinson's disease [2]. CRS was computationally developed to counteract abnormal neuronal synchrony [3]. CRS is a multisite stimulation technique in which phase-shifted stimuli are delivered according to a CR sequence, which characterizes in what order stimuli are delivered to the different stimulation contacts. Computational studies in plastic neuronal networks observed that CRS entailed long-lasting desynchronization after cessation of stimulation [4]. Corresponding long-lasting therapeutic aftereffects were later observed in preclinical and clinical studies in Parkinson's patients [5–9].

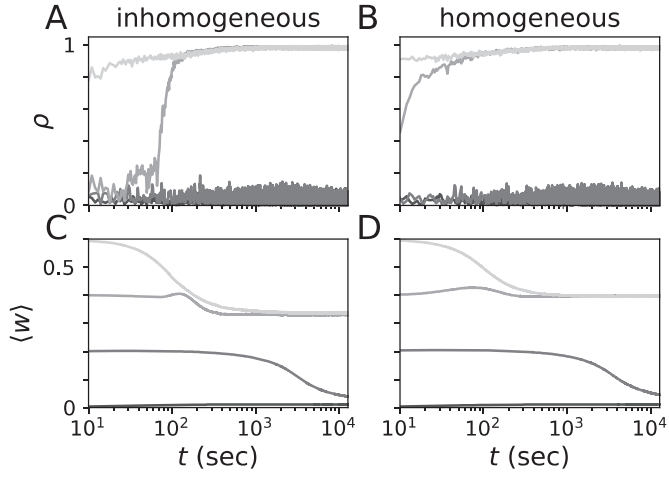


Fig. 1. Coexistence of a strongly connected synchronized state (SCSS) and a weakly connected desynchronized state (WCDS). Trajectories of the Kuramoto order parameters, ρ, according to Eq. (9) (A,B) and corresponding mean synaptic weights, $\langle \mathbf{w} \rangle$, for four initial mean synaptic weights ($\langle w(t=0) \rangle = 0$, 0.2, 0.4, 0.6 (from dark to light gray)) (C,D). Results for an inhomogeneous network are shown on the left and result for a homogeneous network on the right. The same realization of synaptic connections was used in all simulations for each network type. The corresponding initial, individual synaptic weights were distributed according to a bimodal distribution, i.e. $w_{i \to j}(t=0) \in \{0, 1\}$, such that the respective mean synaptic weights were realized. Note the logarithmic time axes.

In a recent letter, we presented results from computational studies that suggest that selecting certain favourable CR sequences may significantly improve the long-lasting desynchronization effects of CRS in networks with spatially dependent synaptic connectivity [1]. Furthermore, shuffling of the CR sequences after short shuffling periods provided robust long-lasting desynchronization effects without the need to select favourable CR sequences [1]. This data article provides a description of the neuronal network model, the code for running the simulations, and additional analyses of the dependence of long-lasting desynchronization effects on the shuffling period and network structure.

3. Data Description

The code to generate the simulated data is available in the public repository available at https://data.mendeley.com/datasets/fmmr595pps/1.

The repository contains the following **folders** and *files*:

- *main.ipynb* ... Detailed description of the code and how to run simulations.
- **CRS**... Contains code for simulations of CRS.
  - *CRS_non_shuffled.py*... Code for simulations of non-shuffled CRS.
  - *CRS_Tshuffle.py*... Code for simulations of shuffled CRS with different shuffle periods.
- **figures**... Contains code for data analyses and generation of the figures presented below.
  - **Fig1**... Code and data for Fig. 1 (see below).
    - **data**... Data files used to generate the figure.
      - *KuramotoOrderParameter_NETWORK_TYPE_mw_MWINIT_seed_SEED.npy*... Trajectory of Kuramoto order parameter for NETWORK_TYPE = "inhomogeneous_network" or "homogeneous_network," initial mean synaptic weights MWINIT = 0.0, 0.2, 0.4, 0.6, and SEED =12.
      - *NETWORK_TYPE_mwTrajectory_w_MWINIT_seed_SEED.npy*... Corresponding trajectories of mean synaptic weight.

- **NETWORK_TYPE_seed_SEED_0_*sec*...** Backups containing output of simulation at $t = 0$ s.
  - *cMatrix.npz*... Sparse matrix containing values of all synaptic weights.
  - *synConnections.npz*... This corresponds to the adjacency matrix containing 1 for excitatory connections and -1 for inhibitory connections.
  - *delayedSpikingNeurons.npy*... List of spikes that are traveling to postsynaptic neurons.
  - *lastSpikeTimeStep.npy*... Last spike times of neurons (imported to accurately continue simulating synaptic weight dynamics from backup file).
  - *evenPriorLastSpikeTimeStep.npy*... Second-to-last spike times of neurons (imported to accurately continue simulating synaptic weight dynamics from backup file).
  - *STNCenter.npy*... Center coordinates of simulated neurons.
  - *GPeCenter.npy*... Center coordinates of inhibitory (GPe) neurons. No GPe neurons were used in the presented simulated data as only excitatory neurons were used for the presented results. In the code, a small number of inhibitory neurons was simulated but all synaptic connections to the excitatory population were removed.
  - *npRandomState.pickle*... State of pseudo random number generator at backup time.
  - *systemState.npy*... Contains state of state variables at backup times.

- *generate_data_from_simulation_output.py*... Python script to obtain data for Fig. 1 from simulation output.
- *generate_Fig1.py*... Python script to generate Fig. 1 from the data obtained using *generate_data_from_simulation_output.py*.
- *Figure_1.png*... Png file of Fig. 1 generated using *generate_Fig1.py*.

- **Fig2** ... Contains code and data for Fig. 2.
  - **data**... Contains data files used to generate Fig. 2.
    - *dic_mean_Weights_STIM_TYPE_NETWORK_TYPE.pickle* ... Contains lists of mean synaptic weights for simulated network and CR sequence realizations for STIM_TYPE=init, nonShuffled, or Tshuffle_X and NETWORK_TYPE=homogeneous or inhomogeneous. X indicates the shuffle period and attains values 01 (100 ms), 10 (10 s), or 1800 (30 min).
    - **cMatrix_conMatrix** ... This is where weight matrices and adjacency matrices from simulation data are saved when running "python generate_data_from_simulation_output.py 2_calculate_average_mean_weight."
  - *generate_data_from_simulation_output.py* ... Python script to get data for Fig2 from simulation output.
  - *generate_Fig2.py*... Python script to generate data for Fig. 2 from simulation output.
  - *Figure_2.png*... Png file of Fig. 2 generated using *generate_Fig2.py*.

- **functions** ... Contains Python scripts with functions used during simulations.
  - *functions_genNetwork.py*... Contains functions for the generation of the different networks of synaptic connections.
  - *functions_pars.py*... Contains the function "gen_Parameter_set_Sequence_Paper(initialSeed)," which generates a dictionary with the system's parameters used in all simulations.
  - *functions_sim.py* ... Contains a variety of functions that are called when running simulations.

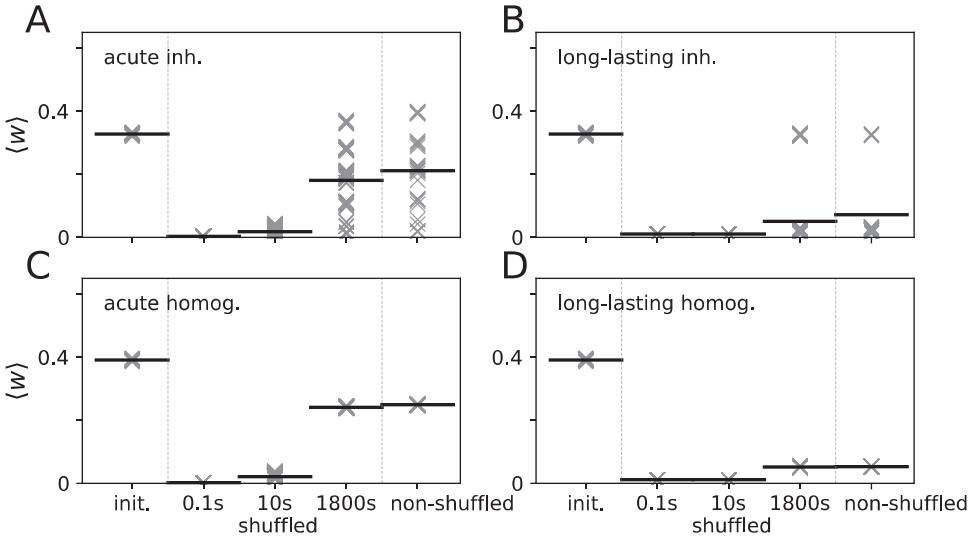


Fig. 2. Statistical analysis of mean synaptic weight before, during, and after stimulation. Mean synaptic weight shortly before cessation of CRS (acute) and ≈ 3 hours after cessation of CRS (long-lasting) are compared to its values before stimulation (init.). Results are shown for inhomogeneous networks (A,B) and homogeneous networks (C,D), and prior to stimulation (init.); at the end of a 2 h session of CRS for non-shuffled CRS (non-shuffled) and for shuffled CRS (shuffled) with shuffle periods 0.1 s, 10s, and 1800 s (A and C), and ≈ 3 hours after cessation of stimulation for the same setups (B and D). Results are shown for five network realization (all) and for the CR sequences I-II-III-IV, I-II-IV-III, I-III-II-IV, I-III-IV-II, I-IV-II-III, I-IV-III-II for non-shuffled CRS and for 30 realizations of the CR sequence for each shuffle period and network realization. Horizontal bars mark averages over individual sequence and network realizations. Symbols show results for individual simulations.

  - *functions_stim.py* … Contains function specifying the CR sequence, stimulus wave form, and other aspects of CRS.
  - *spatial_stimulus_profile.py* … Contains the functions "getWarray(sites, positions, stim_par)," which calculates the relative strength by which neurons at "positions" experience a stimulus delivered to "sites".
- **run_sim** … Python scripts in this folder generate shell commands to run simulations.
  - *1_run.py*… Generates shell commands to run simulations for the synchronized states.
  - *2_run.py* … Generates shell commands to run long simulations for different initial mean synaptic weights. Corresponding trajectories are plotted in Fig. 1.
  - *3_run.py* … Generates shell commands to run simulation on shuffled and non-shuffled CRS for homogeneous and inhomogeneous networks of synaptic connections.
  - *4_run.py* … Generates shell commands to run simulations on shuffled CRS for intermediate networks.
- **synch_states**… Python scripts for simulating networks in the synchronized state.
  - *get_stationary_states_arange_x_inhomogeneous_network.py* … Simulations of synchronized states for inhomogeneous networks.
  - *get_stationary_states_arange_x_homogeneous_network.py* … Simulations for homogeneous networks.
  - *get_stationary_states_arange_x_intermediate_network.py* … Simulations of intermediate networks.
- **multistability** … Contains Python scripts for simulations of the trajectories in Fig. 1.
  - *get_multistability_arange_x_inhomogeneous_network.py* … Simulations of long trajectories for different initial mean synaptic weights for inhomogeneous networks for Fig. 1.

- *get_multistability_arange_x_homogeneous_network.py* .... Simulations of long trajectories for different initial mean synaptic weights for homogeneous networks shown in Fig. 1.

- **images**... Contains images used to explain parts of the code in *main.ipynb*.
  - *typical_output.png*
- **relaxation_after_stimulation** .... Contains code for simulating the network after cessation of stimulation.
  - *less_relaxation.py* ... To perform simulations that start at a backup and simulate network dynamics in the absence of stimulation.

We present the figures contained in the dataset. A detailed description of the figures and how to obtain them using the code is given in section EXPERIMENTAL DESIGN, MATERIALS, AND METHODS.

4. Experimental Design, Materials and Methods

The dataset contains code for simulations of a model of excitatory leaky integrate-and-fire neurons with STDP. In this section, we first describe the neuronal network model and the different networks of synaptic connections for which simulated data are presented in Figs. 1-2 as well as in Ref. [1]. These figures present additional analyses supporting the results of Ref. [1]. Afterwards, we give a detailed explanation of the different simulations performed and describe how they can be reproduced using the Python code contained in the dataset.

4.1. Neuronal network model

Simulations were performed using the model network of excitatory leaky integrate-and-fire (LIF) neurons with STDP from Ref. [10]. A total of $N = 1000$ neurons were simulated. Neuron center coordinates were randomly distributed in the interval $[-L/2, L/2]$ according to a uniform distribution. L is the system's length scale and set to 5mm in simulations. We considered different topologies of synaptic connections, which are described in detail below.

The dynamic equations for the membrane potential and synaptic interaction were taken from Ref. [10]. The dynamics of the ith neuron's subthreshold membrane potential, $V_i(t)$, was given by

$$C_i \frac{dV_i}{dt} = g_{leak}(V_{rest} - V_i) + g_{syn,i}(t)\left(V_{syn} - V_i\right) + I_{stim,i}(t) + I_{noise,i}(t). \tag{1}$$

C_i is the membrane capacitance. Terms on the right-hand side represent: the leak current, with leakage conductance g_{leak} and resting potential V_{rest}; the excitatory synaptic input current, with synaptic conductance $g_{syn,i}(t)$ and reversal potential V_{syn}; the stimulation current $I_{stim}(t)$; and the noisy input current $I_{noise,i}(t)$, modelling input from other brain regions. Neuron I was defined to fire a spike whenever its membrane potential crossed the dynamic threshold potential $V_{th,i}(t)$,

$$\tau_{th} \frac{dV_{th,i}}{dt} = -\left(V_{th,i} - V_{th,rest}\right). \tag{2}$$

Rectangular spikes were implemented by setting $V_i(t)$ to V_{spike} for a duration of τ_{spike}. Afterwards, we performed an instantaneous reset: $V_{th,i}(t) \to V_{th,spike}$ and $V_i(t) \to V_{reset}$.

Synaptic input was modelled by considering the following dynamics of the synaptic conductances $g_{syn,i}(t)$:

$$\tau_{syn} \frac{dg_{syn,i}}{dt} = -g_{syn,i} + \kappa \frac{\tau_{syn}}{N} \sum_{j \in G_i} w_{j \to i}(t) \sum_{l^j} \delta\left(t - t_{l^j}^j - t_a\right). \tag{3}$$

Here, τ_{syn} is the synaptic timescale, t_{lj}^{j} the timing of the l^jth spike of neuron j, and t_a is the synaptic transmission delay. The outer sum runs over the set of presynaptic neurons G_i of neuron i. The strength of synaptic coupling was given by the maximum coupling strength κ. The strengths of individual synapses were scaled by the time-dependent synaptic weights $w_{j\to i}(t)$. Here, j is the index of the presynaptic and i the index of the postsynaptic neuron.

Independent Poisson input was delivered to the neurons modelling noisy input from other brain areas. To this end, Poisson spike trains with mean firing rate f_{noise} were fed into the neurons through excitatory synapses. The resulting input currents, $I_{noise,i}(t)$, were given by

$$I_{noise,i}(t) = g_{noise,i}(t)\left(V_{syn} - V_i\right) \tag{4}$$

with dynamic noise conductances $g_{noise,i}(t)$ that obeyed

$$\tau_{syn}\frac{dg_{noise,i}}{dt} = -g_{noise,i} + \kappa_{noise}\tau_{syn}\sum_{k_i}\delta\left(t_{k_i} - t\right). \tag{5}$$

The noise intensity was set by κ_{noise}. t_{k_i} is the timing of the k_ith spike in the Poisson spike train fed into neuron i.

We consider low noise intensities to reproduce the situation where neurons spike in response to synaptic input and stimuli as observed experimentally, e.g., in the STN and its projection targets in response to deep brain stimulation [11,12]. As shown computationally, in spatially homogeneous networks, intermediate noise and stimulation amplitude levels may increase the robustness of CR stimulation even more [13]. For inhomogeneous networks, this remains to be studied. In contrast, for higher noise levels, the neuronal firing will, ultimately, be dominated and primarily governed by noise, rendering multistable dynamical regimes unstable and counteracting the dedicated stimulus effects studied in Ref. [1].

We chose parameters as in Ref. [10]: $g_{leak} = 0.02$ mS/cm^2, $V_{rest} = -38$ mV, $V_{reset} = -67$ mV, $V_{th,spike} = 0$ mV, $V_{th,rest} = -40$ mV, $\tau_{th} = 5$ ms, $V_{syn} = 0$ mV, $\tau_{syn} = 1$ ms, $t_d = 3$ ms, $\kappa = 8$ mS/cm^2, $\kappa_{noise} = 0.026$ mS/cm^2, and $f_{noise} = 20$ Hz. The membrane capacitances C_i were Gaussian distributed ($N(\mu_C, \sigma_C)$, $\mu_C = 3$ $\mu F/cm^2$ and $\sigma_C = 0.05$ μ_C). For these parameters, the frequency and the range of resulting membrane potential oscillations matched recordings of periodically spiking neurons in the rat subthalamic nucleus [14].

The dynamics of the synaptic weights, $w_{i\to j}(t)$, were determined by STDP. We considered a nearest-neighbor STDP scheme in which weight updates were performed whenever either a postsynaptic or a presynaptic spike arrived at a synapse [15]. The synaptic weight updates, $w_{i\to j} \to w_{i\to j} + W((t_j + t_d) - (t_i + t_a))$, were given by the STDP function

$$W(\Delta t) = \eta\begin{cases} e^{-\frac{|\Delta t|}{\tau_+}}, & \Delta t > 0 \\ 0, & \Delta t = 0 \\ -\frac{\beta}{\tau_R}e^{-\frac{|\Delta t|}{\tau_-}}, & \Delta t < 0 \end{cases}. \tag{6}$$

If the update was triggered by the arrival of a backpropagating postsynaptic spike at the synapse, $\Delta t = (t_j + t_d) - (t_i + t_a)$ is the time lag between the arrival of the backpropagating postsynaptic spike at the synapse, $t_j + t_d$, and the latest presynaptic spike arrival time, $t_i + t_a$. In contrast, if the update was triggered by the arrival of a presynaptic spike at the synapse, Δt is the time lag between the latest postsynaptic spike arrival time, $t_j + t_d$, and the current presynaptic spike arrival time, $t_i + t_a$ [16]. Here, t_d and t_a are the dendritic and the axonal delays, respectively. We considered only an axonal delay of $t_a = 3$ ms and set the dendritic delay to zero.

STDP parameters were chosen according to Ref. [10] such that a stable strongly connected synchronized state (SCSS) and a stable weakly connected desynchronized state (WCDS) coexisted for each network considered here and in Ref. [1] (illustrations of these states can be found in Fig. 1). The impact of STDP parameters on the dynamics of the mean synaptic weight was analyzed in Ref. [17]. The parameter $\eta = 0.01$ scales the weight update per spike. $\tau_R = 4$ causes an asymmetry in the STDP decay times for positive time lags, $\tau_+ = 10$ ms, and for negative time

lags, $\tau_- = \tau_+ \tau_R$. $\beta = 1.4$ scales the ratio of overall long-term depression, given by the integral over the STDP function (Eq. (6)) over all time lags with negative $W(\Delta t)$, to overall long-term potentiation, given by the integral over $W(\Delta t)$, over all time lags with negative $W(\Delta t)$.

To change the model or STDP parameters, the file *functions_pars.py* in folder " **functions**" can be modified.

4.2. Network topologies

We considered three types of networks of synaptic connections: homogeneous networks (Fig. 1A' in Ref. [1]), inhomogeneous networks (Fig. 1A in Ref. [1]), and intermediate networks (Figs. 1L-O in Ref. [1]).

In the **homogeneous networks**, the probability for implementing a synaptic connection between any two different neurons was set to $P_{hom} = 7\%$. The realization of the resulting synaptic connections that was used in the simulations for Ref. [1] is shown in Fig. 1A' in Ref. [1].

In the **inhomogeneous networks**, we first sorted the neurons according to their center coordinates, X. Then, the neurons were separated into four subpopulations such that the first 250 neurons were part of subpopulation one, the second 250 neurons were part of subpopulation two, and so on. Afterwards, synaptic connections were implemented with a probability of $P_{inh} = 14\%$ between pairs of neurons in which the presynaptic neuron was part of subpopulation k and the postsynaptic neuron was part of subpopulation l for $(k, l) \in \{(1,1), (1,4), (2,1), (2,2), (3,1), (3,2), (3,3), (4,2), (4,3), (4,4)\}$. No self-connections were implemented. The realization of the resulting synaptic connections that was used in the Ref. [1] is shown in Fig. 1A in Ref. [1].

In the **intermediate networks**, we introduced a heterogeneity parameter H that scales between homogeneous and inhomogeneous networks. This was done by separating the neurons into subpopulations as in inhomogeneous networks and implementing synaptic connections with probability $HP_{inh}+$ (1- H) P_{hom} for synaptic connections in which the presynaptic neuron was part of subpopulation k and the postsynaptic neuron was part of subpopulation l for $(k, l) \in \{(1,1), (1,4), (2,1), (2,2), (3,1), (3,2), (3,3), (4,2), (4,3), (4,4)\}$ and with probability (1- H) P_{hom} for other synaptic connections. No self-connections were implemented. This way the case $H = 0$ corresponds to the homogeneous network and the case $H = 1$ to the inhomogeneous network.

4.3. Coordinated reset stimulation

We delivered CRS to the networks of LIF neurons using different CR sequences. CRS was delivered to four stimulation sites [3]. The four stimulation sites were located at $X_I = -3L/8$, $X_{II} = -L/8$, $X_{III} = L/8$, and $X_{IV} = 3L/8$. The mean frequency at which each site received stimuli was set to $f_{CR} = 10\ Hz$. We considered the following stimulation patterns:

- **non-shuffled CRS.** During non-shuffled CRS each stimulation site received stimuli periodically with frequency f_{CR}. We characterized the CR sequence at which individual sites receive stimuli using roman letters, e.g., the sequence at which stimuli were administered to site one first, site two second, site three third, and site four last was denoted as I-II-III-IV (Fig. 1F in Ref. [1]).
- **shuffled CRS.** During shuffled CRS a new CR sequence was randomly selected every shuffle period, $T_{shuffle}$. We used a uniform distribution among all 4! possible CR sequences. An exemplary realization of a corresponding CR pattern for $T_{shuffle} = 1/f_{CR}$ is shown in Fig. 1J in Ref. [1]. Note that the special case $T_{shuffle} = 1/f_{CR}$, i.e., shuffling after each CR cycle, is often referred to as CR with rapidly varying sequence (RVS CR) [4,6].

Individual stimuli were charge-balanced and consisted of an excitatory rectangular pulse of duration $T_p = 0.4\ ms$ that was followed by an inhibitory one of duration $2T_p$. The amplitudes

of the excitatory and inhibitory rectangular pulses were $A_{stim}\mu/T_p$ and $A_{stim}\mu/2T_p$, respectively, with $\mu = (V_{th,spike} - V_{reset})/\langle C_i \rangle$.

The stimulation currents each neuron experienced was given by

$$I_{stim,i}(t) = A(t) \sum_j S_j(X_i). \tag{7}$$

Here, $A(t)$ is the amplitude of the stimulus as described in the previous paragraph. $S_j(X_i)$ is the spatial stimulus profile for stimulation site j. Specifically, a neuron with center coordinated X_i was affected by a stimulus delivered to site j at a strength of

$$S_j(X_i) = \left(1 + \left(\frac{X_i - X_j}{\sigma_j}\right)^2\right)^{-1}. \tag{8}$$

σ_j scales the width of the stimulus profile.

We set $A_{stim} = 1$ for which stimuli are strong enough to elevate the membrane potential of neurons that are close to the stimulation site from its reset value to the spiking threshold during the excitatory rectangular pulse part of a stimulus. The width of the stimulus profile was set to $\sigma_j = L/8\pi$ (see Fig. 1K in Ref [1]).

Parameters related to CRS can be changed in *functions_stim.py* and *spatial_stimulus_profile.py* in folder **functions** or by modifying the parameters past to the simulation scripts (see *3_run.py* and *4_run.py*).

4.4. Simulation details

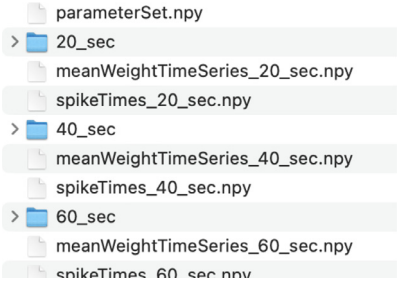
Simulations were performed as follows: First, a random network of synaptic connections for the considered network type was generated according to the procedure described above. Initial membrane potentials were distributed uniformly between V_{reset} and $V_{th,rest}$, and all synaptic conductances were set to zero (initial conditions can be modified in function "set_initial_conditions_for_nodes(system_parameters)" in file *functions_sim.py* in folder **functions**). Then, the initial synaptic weights were randomly generated such that individual weights either attained zero or one and a predefined mean synaptic weight was realized. For simulations of the synchronized state, the predefined mean synaptic weight was 0.5. Numerical integration was performed using the Euler method with an integration time step of 0.1 ms.

4.4.1. Simulations of synchronized states

We simulated the network of LIF neurons in the synchronized state. Corresponding shell commands for inhomogeneous and homogeneous networks, as well as intermediate networks can be generated using the Python script *1_run.py* in folder **run_sim**. Simulations were performed for different seeds of NumPy's pseudorandom number generator.

To get the shell commands for running simulations run "python 1_run.py NETWORK_TYPE", where NETWORK_TYPE = inhomogeneous_network, homogeneous_network, or intermediate_network. We submitted these commands to Stanford's computing cluster.

Simulation scripts generate output every 20 secs including an array of spike times for each spiking neuron *spikeTimes_T_sec.npy*, a trajectory of the mean synaptic weight (averaged over all possible connections, NOT all implemented connections) *meanWeightTimeSeries_T_sec.npy*, and a backup directory**.../T_sec** from which other simulations can be started. Here, T is an integer specifying the time in seconds. Spike times and mean weight trajectories contain data for the 20 s prior to T. In both files, time is measured in time steps (currently 0.1 ms). Typical output looks as follows:



The file *parameterSet.npy* contains system parameters. Additionally, backup folder names and times will be saved in *listOfBackupTimeSteps.txt*. The final output will have the identifier "Final-Backup" instead of "T_*sec*".

4.4.2. Simulations to get trajectories shown in Fig. 1

We simulated the LIF neurons for different predefined mean synaptic weights at $t = 0$. Corresponding commands for inhomogeneous and homogeneous networks can be generated using the script *2_run.py* in folder **run_sim**. Simulations were performed for different seeds used to set NumPy's pseudorandom number generator and different mean synaptic weights. The goal was to see whether different initial conditions (realizations of mean synaptic weights at the beginning of the simulation) lead to different degrees of neuronal synchrony as quantified by the Kuramoto order parameter (see below).

To run simulations, we submitted the commands that were output of "python 2_run.py NETWORK_TYPE", where multistability_NETWORK_TYPE = multistability_inhomogeneous or multistability_homogeneous, to Stanford's computing cluster. The output directory is specified as dataDirectory + '/homogeneous_network/multistability' by default and can be changed by modifying the file path "dataDirectory".

Simulations for homogenous and inhomogeneous networks and different initial mean synaptic weights are shown in Fig. 1. For both inhomogeneous and homogeneous networks, a stable SCSS and a stable WCDS coexist (see Fig. 1). We measured the degree of neuronal synchrony using the Kuramoto order parameter [18]

$$\rho(t) = \left| \frac{1}{N} \sum_{k=0}^{N-1} e^{2\pi \, I \, \psi_k(t)} \right|. \tag{9}$$

N is the number of neurons, I the imaginary unit, and $\psi_k(t)$ is a phase function associated with the inter-spike intervals of neuron k. $\psi_k(t)$ attains subsequent integer values at the subsequent spike times of neuron k and increases linearly during inter-spike intervals [19]. $\rho(t) \approx 1$ corresponds to synchronized spiking and $\rho(t) \approx 0$ indicates a lack of synchronized spiking.

To load the trajectories shown in Fig. 1 from simulation output, we used the code in the Python script *generate_data_from_simulation_output.py* in folder **figures/Fig1**. First, the trajectories of the mean synaptic weight and the spike trains were loaded ("python generate_data_from_simulation_output.py get_traj_meanWeight_spikeTrain_multistability NETWORK_TYPE"). This generates the files *NETWORK_TYPE_mwTrajectory_w_MWINIT_seed_12.npy* and *NETWORK_TYPE_spikeTrain_w_MWINIT_seed_12.npy* in the folder **figures/Fig1/data**. MWINIT is the initial mean weight at $t = 0$ and can be specified in *run_sim/2_run.py*.

Once these files are generated, the trajectory of the Kuramoto parameter can be generated using "python generate_data_from_simulation_output.py calculate_time_trace_of_Kuramoto_parameter_multistability_mw". This will generate the files *KuramotoOrderParameter_NETWORK_TYPE_mw_MWINIT_seed_12.npy* in **figures/Fig1/data**. Then, Fig. 1 can be generated using the python script *generate_Fig1.py*.

In Fig. 1, trajectories starting with high initial mean synaptic weight (light gray) approach a state with synchronized spiking ($\rho \approx 1$) and a mean synaptic weight between ≈ 0.3 and ≈ 0.4,

depending on the type of network, corresponding to an SCSS. In contrast, trajectories with low initial mean synaptic weight approach a state with low degree of neuronal synchrony ($\rho \approx 0$), corresponding to a WCSS. This shows the coexistence of a stable SCSS, used to model pathological neuronal synchrony, e.g., in Parkinson's disease, in Ref. [1], and a stable WCSS, used to model physiological neuronal activity in Ref. [1].

4.4.3. Simulations of CRS

To study, the effect of CRS on networks in the SCSS, we chose a large initial mean synaptic weight (half of the synaptic weights were set to one and the other half to zero) and simulated the network for 3000s, such that the network approached the SCSS. In the SCSS, the mean synaptic weight was between 0.3 and 0.4 for the inhomogeneous networks (Fig. 1B-E in Ref. [1] and Figs. 2A and B, init.) and ≈ 0.4 for the homogeneous networks (Fig. 1B'–1E' in Ref. [1] and Figs. 2C and 2D, init.). After 3000 s, CRS was delivered for 2 h. After cessation of stimulation, we simulated the network until the mean synaptic weight approached a stationary value. Resulting trajectories of the mean synaptic weights for various CR patterns and inhomogeneous and homogeneous networks are shown in Fig. 1 in Ref. [1]. A more detailed statistical analysis of the mean synaptic weight before, during, and after stimulation is presented in the next section.

The corresponding simulations start from the backup files for networks in the synchronized state from section 4.4.1) "Simulations of synchronized states" at 3000 s. We delivered CRS to four stimulation sites.

Simulations on CRS for homogeneous and inhomogeneous networks can be started using the shell commands generated by *3_run.py* in folder **run_sim**. To this end, run "python 3_run.py STIM_MODE". STIM_MODE describes what kind of scenario is simulated.

- Use STIM_MODE = CRS_Tshuffle to simulate shuffled CR with shuffling periods 0.1 s, 10.0 s, and 1800.0 s. These simulations start from the backups generated in section 4.4.1) "Simulations of synchronized states". Make sure that the latter simulations have finished and adjust the file paths in *3_run.py* if necessary. Backups are only generated after 1 hour and after 2hours of stimulation.
- Use STIM_MODE = CRS_non_shuffled to simulation non-shuffled CR with sequences: 'I_II_III_IV','I_II_IV_III','I_III_II_IV','I_III_IV_II','I_IV_II_III','I_IV_III_II'. These simulations also start from the backups as described for STIM_MODE = CRS_Tshuffle.
- Use STIM_MODE = relaxation_after_shuffled_CRS to simulate the relaxation after shuffled CR. These simulations start from the backup generated by the case of STIM_MODE = CRS_Tshuffle after 10,200 s. Adjust the paths in *3_run.py* if necessary. Backups are generated less often to save memory (see "TBackupSteps" in *less_relaxation.py* in folder **relaxation_after_stimulation**).
- Use STIM_MODE = relaxation_after_non_shuffled_CR to simulation the relaxation after non-shuffled CR. These simulations start from the backup generated by the case of STIM_MODE = CRS_non_shuffled after 10,200 s. Adjust the paths in *3_run.py* if necessary. Backups are generated less often to save memory (see "TBackupSteps" in *less_relaxation.py* in folder **relaxation_after_stimulation**).

Simulations for intermediate networks can be started using the shell commands generated by *4_run.py*. To this end, type "python 4_run.py STIM_MODE".

- Use "STIM_MODE = CRS_Tshuffle_intermediate_networks" to simulate shuffled CR with shuffling periods 0.1 s, 10.0 s, and 1800.0 s for heterogeneity parameters $H = 0, 0.2, 0.4, 0.6, 0.8, 1.0$. These simulations start at the backups generated in section 4.4.1) "Simulations of synchronized states". Make sure that the latter simulations have finished and adjust the file paths in *4_run.py* if necessary.
- Use "STIM_MODE = relaxation_after_shuffled_CRS_intermediate_networks" to simulate the relaxation after shuffled CR. These simulations start from the backup generated in the case of "STIM_MODE = CRS_Tshuffle_intermediate_networks" after 10,200 s. Adjust the paths in *4_run.py* if necessary.

4.5. Mean synaptic weight before, during, and after CRS

We analyzed the mean synaptic weight of inhomogeneous and homogeneous networks before, during, and after CRS (Fig. 2). Corresponding exemplary trajectories of the mean synaptic weight are shown in Fig. 1 in Ref. [1]. Additionally, for both inhomogeneous and homogeneous networks, we performed simulations for five network realizations and recorded the mean synaptic weights after 3000 s of simulation. Results are labeled "init." in Fig. 2. Then, we simulated a 2 *h* session of either non-shuffled CRS or shuffled CRS with different shuffle periods (Fig. 2). At the end of the CRS session, we recorded snapshots of the mean synaptic weight (acute) for different realizations of the CR sequences.

Once simulations of CRS are done, the simulated data shown in Fig. 2 can be reproduced using the python script *generate_data_from_simulation_output.py* in folder **figures/Fig2**. This will copy weight and adjacency matrices from simulations to folder **figures/Fig2/data/ cMatrix_conMatrix** ("python generate_data_from_simulation_output.py 1_copy_weight_and_adjacency_matrix"). Then, dictionaries containing averages over different network and CR sequence realizations were generated by running "python generate_data_from_simulation_output.py 2_calculate_average_mean_weight". The dictionaries can be found in folder **figures/Fig2/data**. Fig. 2 can be generated using these data with the script *generate_Fig2.py* in folder **figures/Fig2**.

In the inhomogeneous networks the recorded mean synaptic weights spread out over a wide range with an average of approximately ≈ 0.25 for non-shuffled CR and of ≈ 0.2 for shuffled CR with a shuffle period of 1800 *s* (Fig. 2A). In contrast, for short shuffle periods (0.1 *s* and 10 *s*) the mean synaptic weights were close to zero and similar across network realizations and realizations of the CR sequences (Fig. 2A). In the homogeneous networks, the mean synaptic weights were close to zero after 2 *h* of CRS for short shuffle periods; however, shuffled CRS with long shuffle periods or non-shuffled CR led to non-zero mean synaptic weights (Fig. 2C).

After the stimulation session, we simulated the system for another 3 h (see previous section for corresponding code). During this time the mean synaptic weights approached either their value in an SCSS or their value in a WCDS. Values of the mean synaptic weights recorded 3 h after different types of CRS are shown in Fig. 2B for the inhomogeneous and in Fig. 2D for the homogeneous networks. In the inhomogeneous network, after non-shuffled CR and shuffled CR with a shuffle period of 1800 *s* the mean synaptic weight approached either large values or small values depending on the network realization and the realization of the CR sequence (Fig. 2B). In contrast, after shuffled CR with short shuffle periods the system remained in the WCDS after cessation of stimulation, corresponding to long-lasting desynchronization effects of CRS. However, in the homogeneous network all network and sequence realizations led to similar mean synaptic weights at the end of the CRS session (Fig. 2C) and 3 h after cessation of stimulation (Fig. 2D). Here, the results did neither depend on the network realization nor on the realization of the CR sequence.

The Python scripts in the dataset can also be used to generate the simulated data shown in Fig. 1 in Ref. [1]. To this end, we performed simulations for a homogeneous network and an inhomogeneous network using seed=12 in *1_run.py* and simulations of non-shuffled and shuffled CRS and relaxation after cessation of simulation for these networks.

The results shown in the panels of Fig. 1 in Ref. [1] for intermediate networks with different degrees of heterogeneity, can be obtained from simulation data (see previous section).

5. Limitations

The presented network model allows for performing long stimulations for many different parameter combinations, i.e., CR sequences and networks of synaptic connections. Long simulation times are needed to simulate relaxation after stimulation as relaxation, especially, towards the desynchronized state with weak synaptic connections is rather slow.

We considered data for five different realizations of synaptic connections for homogeneous and inhomogeneous networks. The corresponding simulated mean synaptic weights prior to stimulation were close to each other (see Fig. 2, init.), suggesting that the major part of the variability in the mean synaptic weights after stimulation resulted from different CR sequence realizations. Given the large number of possible CR sequence realizations, the considered 30 CR sequence realizations may not fully capture the possible outcomes.

The LIF model only models subthreshold dynamics and the effects of stimulation on the spike shape are neglected.

Knowledge of synaptic connectivity in target brain regions of deep brain stimulation, e.g., in PD, is limited. The presented dataset provides valuable inside into the potential effects of inhomogeneous network structure on the long-lasting outcome of stimulation.

The model considered long-term plasticity in the form of STDP. Short-term plasticity, as well as structural plasticity is not considered.

Ethics Statement

The authors have read and followed the ethical requirements for publication in Data in Brief and confirm that the current work does not involve human subjects, animal experiments, or any data collected from social media platforms.

Data Availability

Simulated dataset on coordinated reset stimulation of homogeneous and inhomogeneous networks of excitatory leaky integrate-and-fire neurons with spike-timing-dependent plasticity (Original data) (Mendeley Data)

CRediT Author Statement

Justus A. Kromer: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Data curation, Writing – original draft, Writing – review & editing, Visualization; **Peter A. Tass:** Conceptualization, Methodology, Validation, Investigation, Resources, Writing – review & editing, Visualization, Supervision, Project administration, Funding acquisition.

Acknowledgements

PAT gratefully acknowledges support by the John A. Blum Foundation, the Alda Parkinson's Research Fund, and the Vaughn Bryson Research Fund. We are grateful to Stanford University and Stanford's Sherlock Computing cluster for computational resources and support that contributed to these research results.

Declaration of Competing Interest

JAK and PAT filed a Stanford-owned provisional patent related to the presented results.

References

[1] J.A. Kromer, P.A. Tass, Sequences and their shuffling may crucially impact coordinated reset stimulation – A theoretical study, Brain Stimul. 17 (2024) P194–P196.
[2] C. Hammond, H. Bergman, P. Brown, Pathological synchronization in Parkinson's disease: networks, models and treatments, Trends Neurosci. 30 (2007) P357–P364.

[3] P.A. Tass, A model of desynchronizing deep brain stimulation with a demand-controlled coordinated reset of neural subpopulations, Biol. Cybern 89 (2003) 81–88.
[4] P.A. Tass, M. Majtanik, Long-term anti-kindling effects of desynchronizing brain stimulation: a theoretical study, Biol. Cybern. 94 (2006) 58–66.
[5] P.A. Tass, L. Qin, C. Hauptmann, S. Dovero, E. Bezard, T. Boroud, W.G. Meissner, Coordinated reset has sustained aftereffects in parkinsonian monkeys, Ann. Neurol. 72 (2012) 816–820.
[6] I. Adamchic, C. Hauptmann, U.B. Barnikol, N. Pawelczyk, O. Popovych, et al., Coordinated reset neuromodulation for Parkinson's disease: proof-of-concept study, Mov. Disord. 29 (2014) 1679–1684.
[7] J. Wang, S. Nebeck, A. Muralidharan, M.D. Johnson, J.L. Vitek, K.B. Baker, Coordinated reset deep brain stimulation of subthalamic nucleus produces long-lasting, dose-dependent motor improvements in the 1-methyl-4-phenyl-1, 2, 3, 6-tetrahydropyridine non-human primate model of parkinsonism, Brain Stimul. 9 (2016) P609–P617.
[8] J. Wang, S.P. Fergus, L.A. Johnson, S.D. Nebeck, J. Zhang, et al., Shuffling improves the acute and carryover effect of subthalamic coordinated reset deep brain stimulation, Front. Neurol. 13 (2022) 716046.
[9] J.C. Bore, B.A. Campbell, H. Cho, F. Pucci, R. Gopalakrishnan, et al., Long-lasting effects of subthalamic nucleus coordinated reset deep brain stimulation in the non-human primate model of parkinsonism: a case report, Brain Stimul. 15 (2022) 598–600.
[10] J.A. Kromer, A. Khaledi-Nasab, P.A. Tass, Impact of number of stimulation sites on long-lasting desynchronization effects of coordinated reset stimulation, Chaos 30 (2020) 083134.
[11] L. Garcia, J. Audin, G. D'Alessandro, B. Bioulac, C. Hammond, Dual effect of high-frequency stimulation on subthalamic neuron activity, J. Neurosci. 23 (2003) 8743–8751.
[12] F. Agnesi, A. Muralidharan, K.B. Baker, J.L. Vitek, M.D Johnson, Fidelity of frequency and phase entrainment of circuit-level spike activity during DBS, J. Neurophysiol. 114 (2015) 825–834.
[13] A. Khaledi-Nasab, J.A. Kromer, P.A. Tass, Long-lasting desynchronization of plastic neuronal networks by double-random coordinated reset stimulation, Front. Netw. Physiol. 2 (2022) 864859.
[14] M.D. Bevan, C.J. Wilson, Mechanisms underlying spontaneous oscillation and rhythmic firing in rat subthalamic neurons, J. Neurosci. 19 (1999) 7617–7628.
[15] A. Morrison, M. Diesmann, W. Gerstner, Phenomenological models of synaptic plasticity based on spike timing, Biol. Cybern. 98 (2008) 459–478.
[16] M. Madadi Asl, A. Valizadeh, P.A Tass, Dendritic and axonal propagation delays determine emergent structures of neuronal networks with plastic synapses, Sci. Rep. 7 (2017) 39682.
[17] G.K. Ocker, A. Litwin-Kumar, B. Doiron, Self-organization of microcircuits in networks of spiking neurons with plastic synapses, PLoS Comput. Biol. 11 (2015) e1004458.
[18] Yoshiki Kuramoto, Chemical Oscillations, Waves, and Turbulence, Springer, Berlin, 1984.
[19] M. Rosenblum, A. Pikovsky, J. Kurths, C. Schäfer, P.A. Tass, Phase synchronization: from theory to data analysis, in: F. Moss, S. Gielen (Eds.), Handbook of Biological Physics, vol. 4, Elsevier, Amsterdam, 2001, pp. 279–321.