Solving the Sudoku problem with spiking neural networks

David Samu
09/10/2017

Contents

- Solving Sudoku puzzles (forward pass / inference)
- Generating Sudoku puzzles (backward pass / spontaneous state settling)
- Sampling from compatible solutions (solving the under-defined problem)
- Learning the connectivity by examples

Example puzzles

3	4	1	2
4	2	3	1

3	4	1	2
2	1	4	3
1	3	2	4
4	2	3	1

Example puzzles

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

5	3	4	6	7	8	9	1	2
6	7	2	1	9	5	3	4	8
1	9	8	3	4	2	5	6	7
8	5	9	7	6	1	4	2	3
4	2	6	8	5	3	7	9	1
7	1	3	9	2	4	8	5	6
9	6	1	5	3	7	2	8	4
2	8	7	4	1	9	6	3	5
3	4	5	2	8	6	1	7	9

Example puzzles

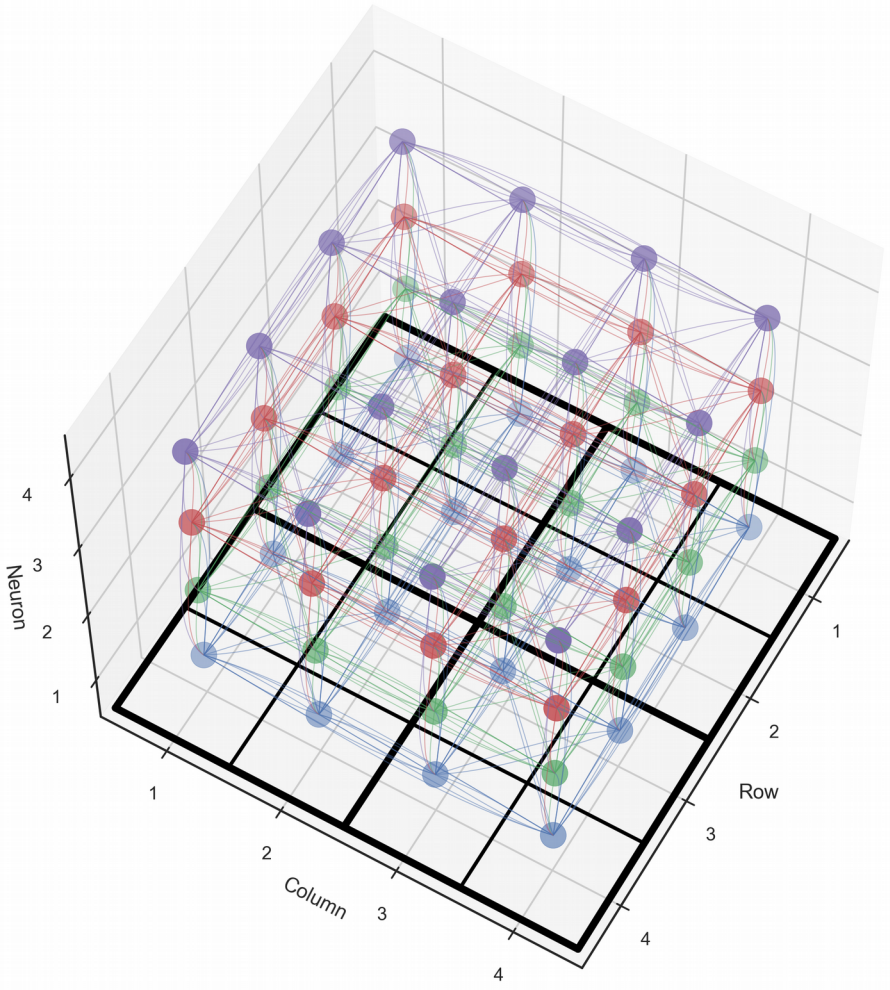
1			2	3	4			12		6				7	
		8				7			3			9	10	6	11
	12			10			1		13		11			14	
3			15	2			14				9			12	
13				8			10		12	2		1	15		
	11	7	6				16				15			5	13
			10		5	15			4		8			11	
16			5	9	12			1						8	
	2						13			12	5	8			3
	13			15		3			14	8		16			
5	8			1				2				13	9	15	
		12	4		6	16		13			7				5
	3			12				6			4	11			16
	7			16		5		14			1			2	
11	1	15	9			13			2				14		
	14				11		2			13	3	5			12

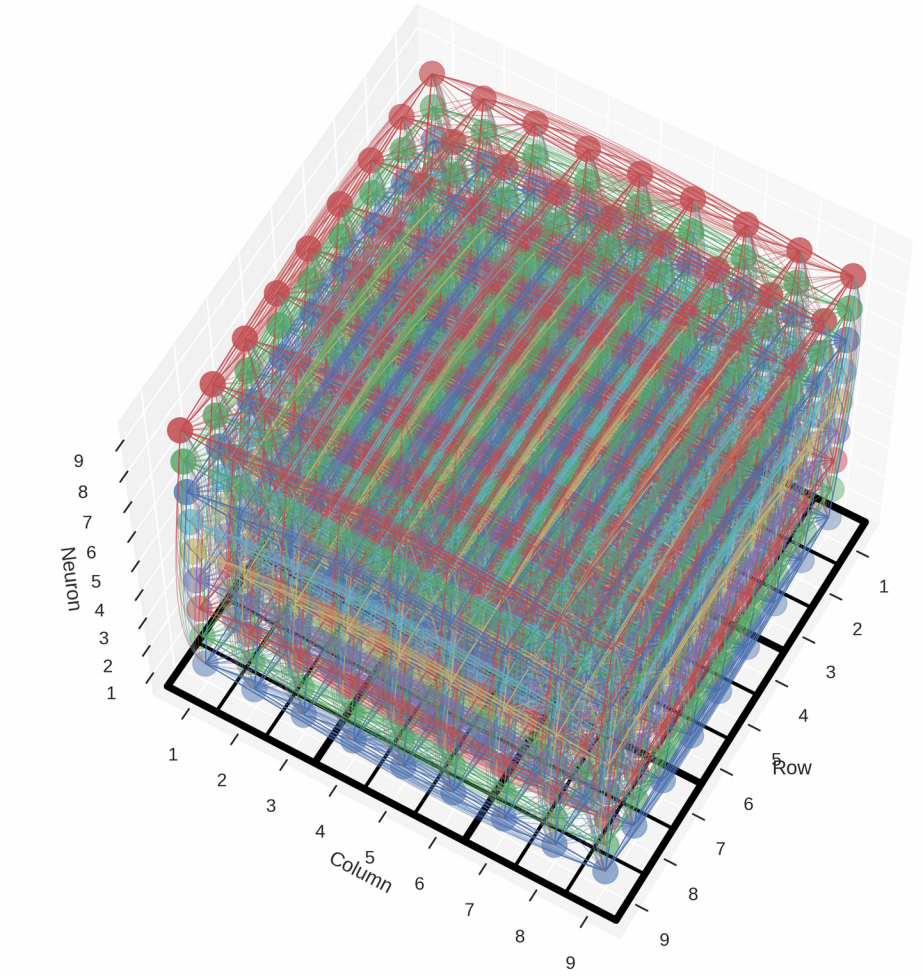
1	5	10	2	3	4	9	11	12	16	6	14	15	13	7	8
14	16	8	13	5	15	7	12	4	3	1	2	9	10	6	11
9	12	4	7	10	16	6	1	8	13	15	11	3	5	14	2
3	6	11	15	2	13	8	14	7	5	10	9	4	16	12	1
13	4	14	3	8	7	11	10	5	12	2	6	1	15	16	9
8	11	7	6	4	1	2	16	9	10	14	15	12	3	5	13
12	9	1	10	13	5	15	6	3	4	16	8	14	2	11	7
16	15	2	5	9	12	14	3	1	11	7	13	10	6	8	4
6	2	16	14	11	9	4	13	15	1	12	5	8	7	10	3
7	13	9	1	15	2	3	5	11	14	8	10	16	12	4	6
5	8	3	11	1	10	12	7	2	6	4	16	13	9	15	14
15	10	12	4	14	6	16	8	13	9	3	7	2	11	1	5
2	3	5	8	12	14	10	15	6	7	9	4	11	1	13	16
10	7	13	12	16	3	5	9	14	8	11	1	6	4	2	15
11	1	15	9	6	8	13	4	16	2	5	12	7	14	3	10
4	14	6	16	7	11	1	2	10	15	13	3	5	8	9	12

The (mechanistic) model

- Spiking neural network
- Units: I&F neurons, each representing a number
  - e.g. in an N-by-N table, a number between 1 and N
- Network structure: N x N x N units, organized in a 3-D grid
- Connectivity: encodes the rules of Sudoku via WTA circuits (all-to-all inhibition) at
  - each position of the table (only one value is possible at each position)
  - each row, column and subsquare among units encoding the same number (numbers cannot repeat along rows, etc)
- Input: delivered as Poisson processes
- Output: active unit at each table cell (decoded by rate or spiking coding)
- Critical parameters: input strength – to – lateral inhibition ratio

Network connectivity

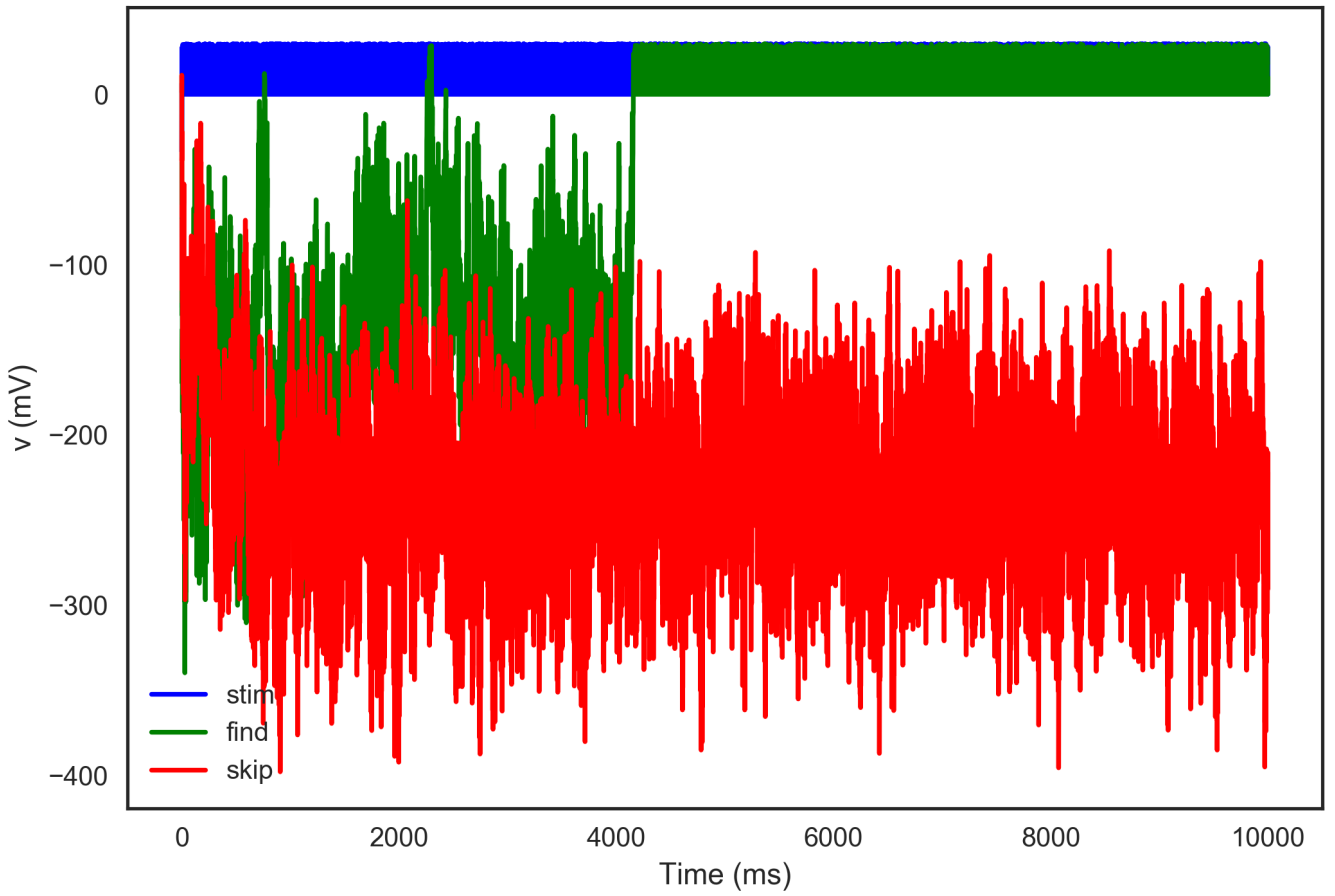




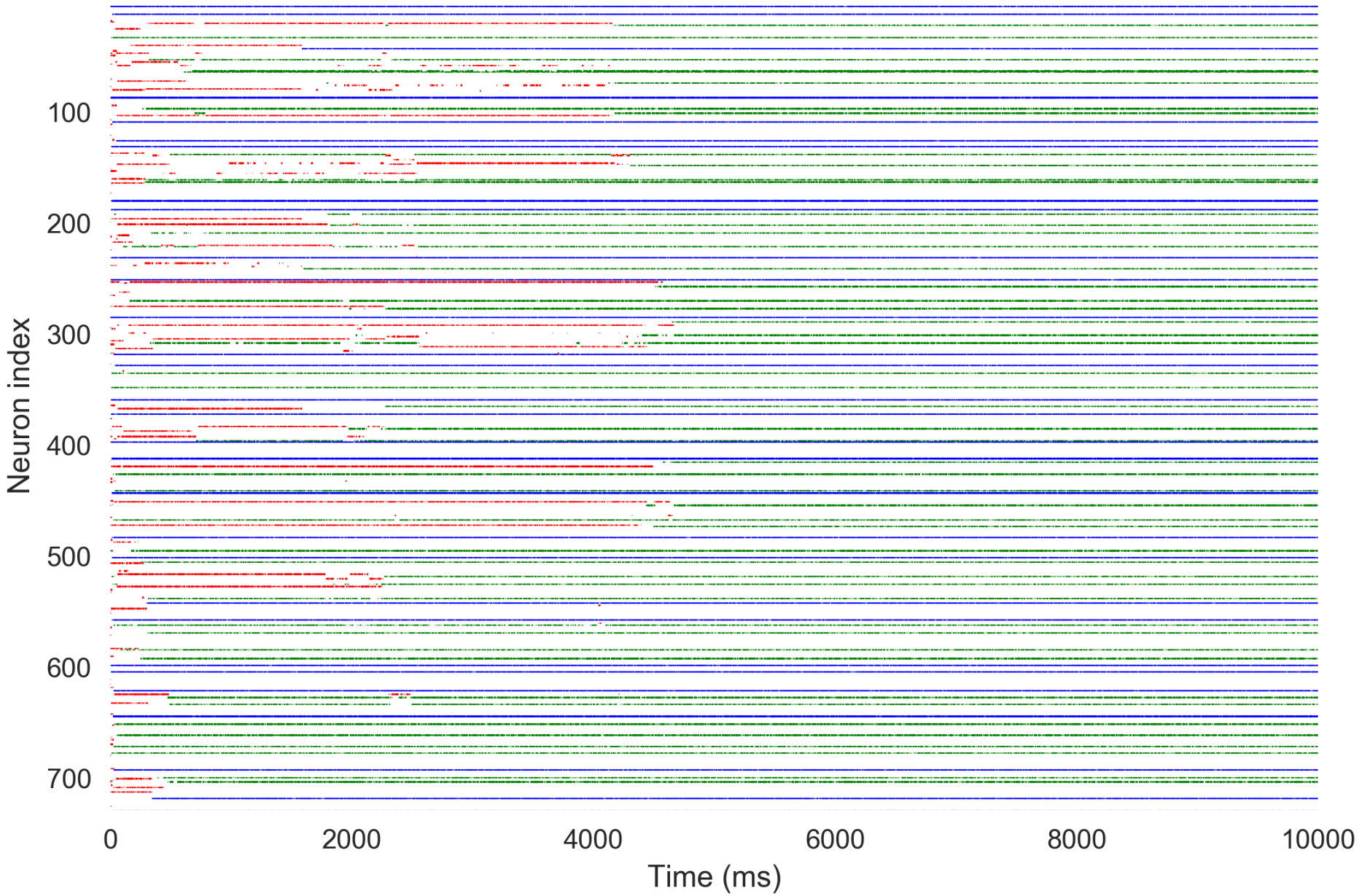
Perception – Sudoku model correspondence

- **puzzle**: available incomplete/ambiguous sensory information
  - received as input
- **solution**: complete state of the world
  - represented in the activity of units
- **solving the puzzle**: inferring the complete state of the world
  - arrived at by the network through inference
- **multiple solutions of an underdefined problem**: alternative, but “correct” interpretations of the sensory information (Necker cube)
  - the network randomly transitions between the corresponding states, according to the neural sampling hypothesis
- **rules of Sudoku**: relationships among features or causes of sensory data
  - encoded by network connectivity

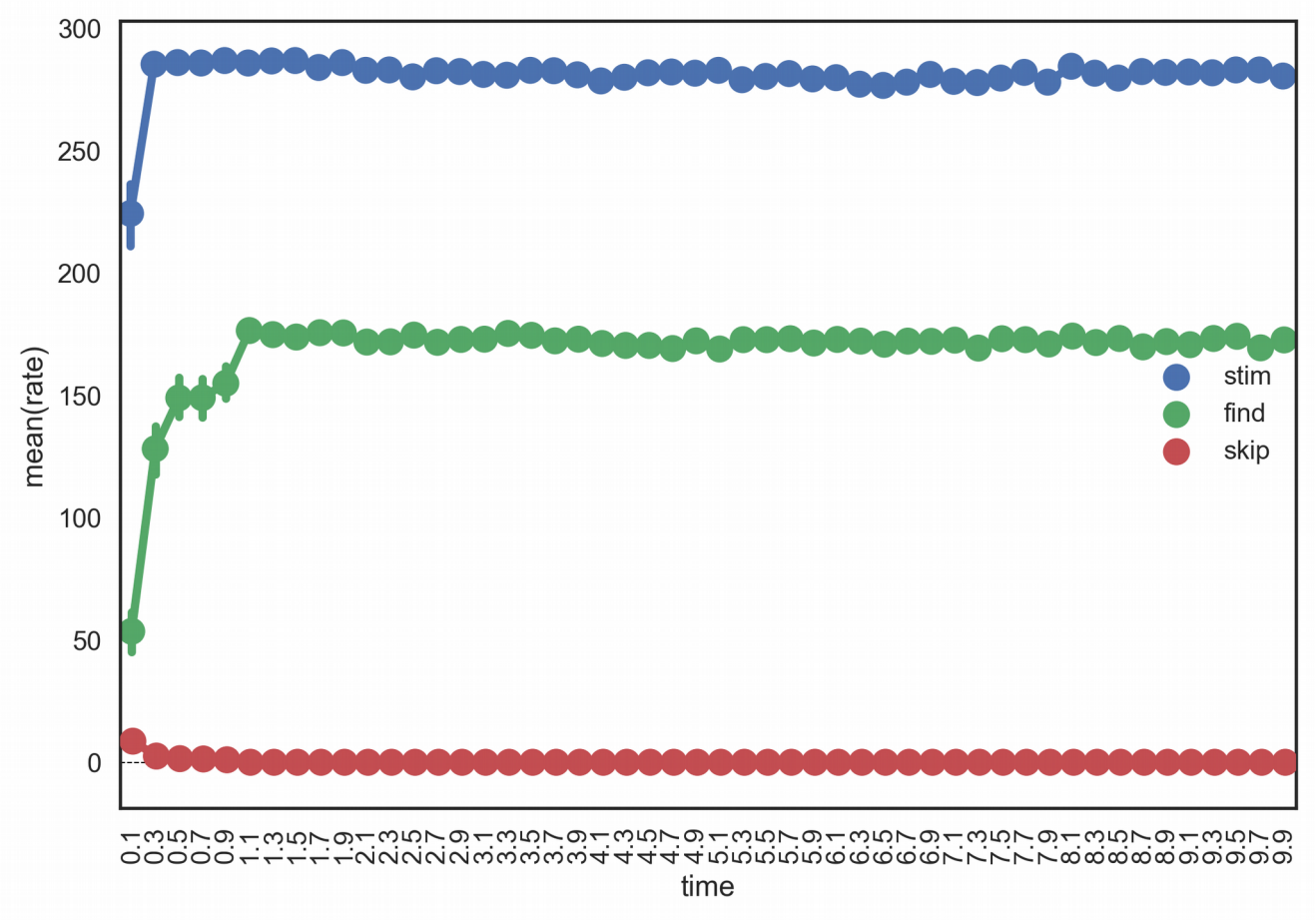
Solving the puzzle
Example voltage traces



Raster plot


Neuron index
100
200
300
400
500
600
700
0
2000
4000
6000
8000
10000
Time (ms)

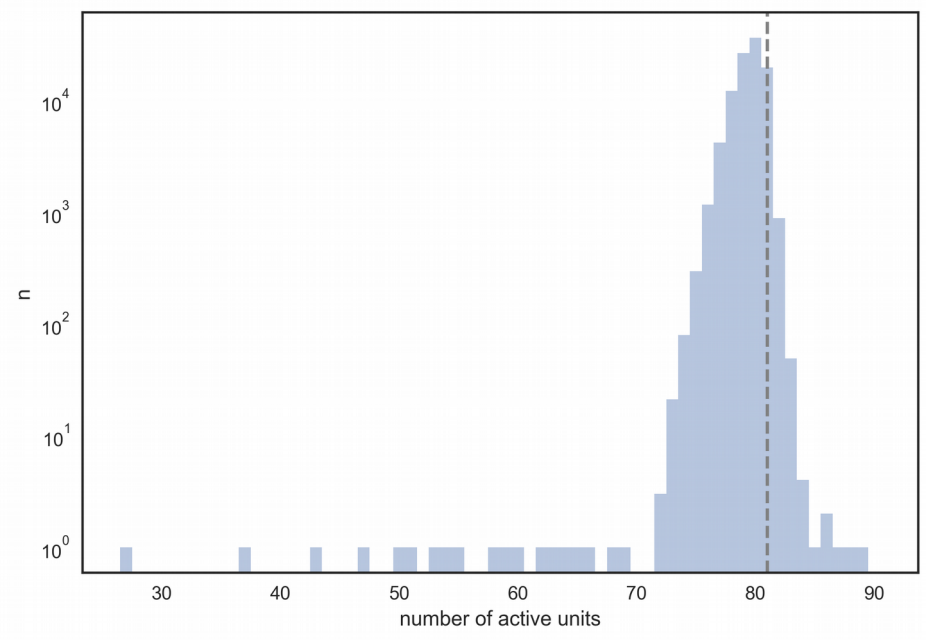
Mean rates per group

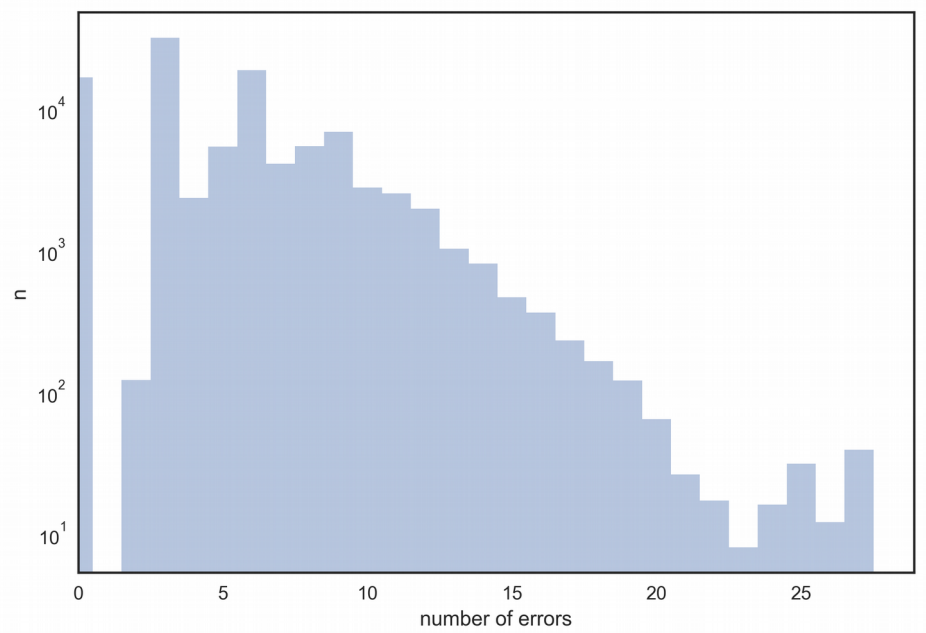


Generating new configurations

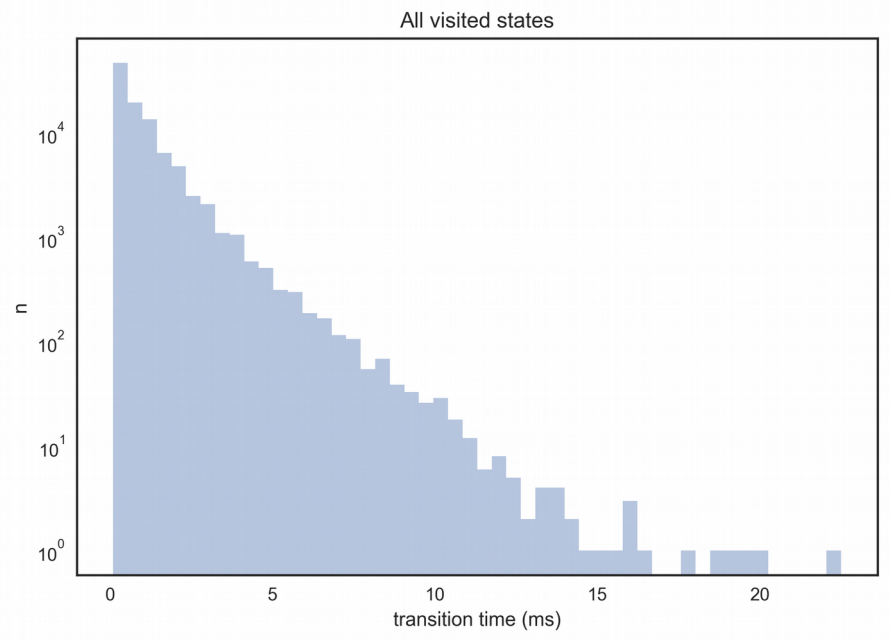
- Equivalent to: Dreaming valid/coherent states of the world (correct puzzles)
  - in the probabilistic context: probable states, given the distribution of the data
  - incorrect Sudoku tables are improbable/impossible states, therefor shouldn’t be generated (sampled from)
- Starting from random initial network states, 100 / 100 tables generated were correct Sudoku configurations.

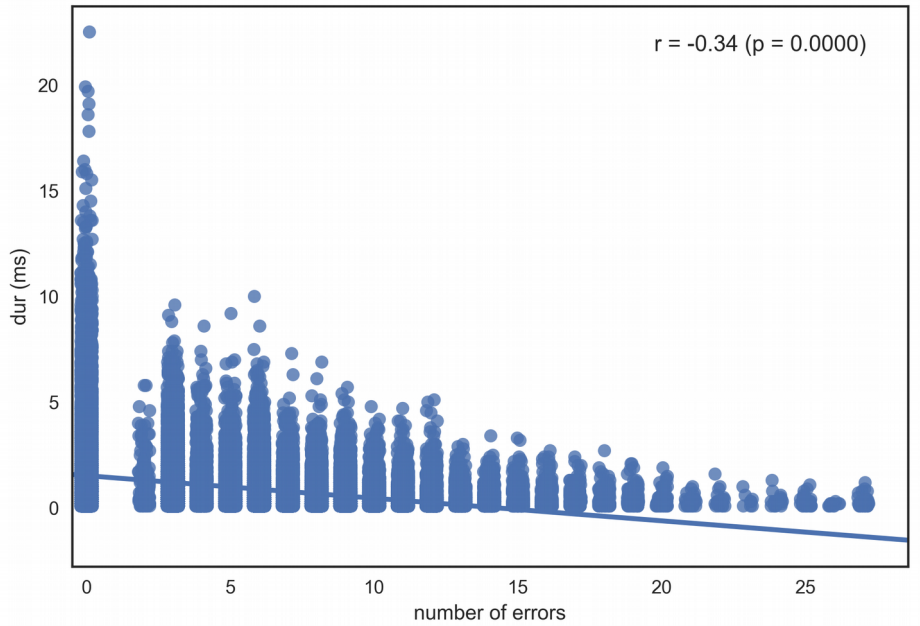
Sampling from alternative solutions in a heavily underdefined puzzle





Sampling from alternative solutions in a heavily underdefined puzzle





Learning the weights

1) ‘Supervised’ learning: presenting correct solutions as input and let the evoked activity adjust the weights by STDP

2) Reinforcement learning? Provide positive reward signal when the network happens upon a correct solution in spontaneous activity (no/restricted input).