NNetSimu

A simulation of natural neural networks

User documentation

P. Kraus

14. 05.2020

**Changes in this release:**

1. Error #41ifaulted: Cancelling the Analyzefunctions with ESC did not work.
2. Error #4i 2: If the Analyzefunctions did notfind anything, the model moved outside the visible range.
3. Error #43 fixed: The Outgoing „Dendrite command was not executable in certain situations.
4. Various errors that can lead to crashes have been fixed.
5. Issue #36 has beenimplemented: Second window with a thumbnail of the entire model, see chapter 4.7 .

# Introduction

NNetSimu is a simulation of natural neural networks, such as the ­ human brain. The program allows to create a model consisting of neurons and ­ links between these neurons and to observe the dynamic behavior of the model. NNetSimu must necessarily greatly simplify the complexity of biological systems (see Chapter 4.6 ).

This document applies only to the elements implemented in NNetSimu. Statements of the species "there are three types of ..." are therefore not to be understood as claims about biological systems (where the diversity is usually higher), but only as descriptions of the functionality currently implemented in the NNetSimu program.

# The Elements of the models

## Static and dynamic model

NNetSimu displays a model of a neural network that contains static and dynamic ­ aspecs. The static model describes a number of differentneurons with their respectivepositions on the work surface as well ­ as connections between the neurons.

The **static model** can be changed by the user v,e.g. by adding other ­ neurons or changing their position,seeKapitel 4.3 .

The **dynamic model** describes the distribution and change of electrical voltage in the elements of the static model over time. Input neurons give in a ­ tuned clock electrical impulses that run through the connections to neurons. The ­ incoming impulses can cause a neuron to emit impulses itself, etc. These dynamic ­ operations are performed automatically in NNetSimu, but they canbe influenced by certain ­ parameters.

There are three types of neuronsthat differ visually by their outer shape (rectangle, circle with or without bulge). . In the following illustrations, in addition to the shape of the various ­ types of neurons to recognize other elements such as colors, and labels that have meaning for the dynamic model. .

## Input-Neuronen

Inputneuronsare represented by rectangles that are opened in one direction. They haveben no incoming dendrites and an ­ exit, the axon.

Input neurons are virtually the input into the neural network. They are triggered by external events, usually sensory stimuli, such as incident light at the eye or acoustic signals in the neurons in the inner ear.

The dynamic behavior of an input neuron is determined by the individually adjustable (see 4.5.3 ) pulse frequency and the trigger voltage identical for all neurons,, "threshold" (see 3.2 ). The electrical potential increases linearly until the trigger voltage is reached. The speed of the voltageriseis such that thetrigger ingenuation voltage is reached exactly after a pulse duration. Then an impulse is triggered in the axon,whose amplitude is determined by the parameter "peak voltage". The potential in the input neuron is reset to 0 volts. Now the process starts from the beginning, the potential increases linearly, etc.

The current state of an input neuron is visualized by the color of the interior, which changes with increasing electrical potential from black to a brighterenhue (currently ­ red).

## "Normale" neurons

"Normal" neurons are represented by a circle that has a bulge at one point. You have one or more incoming connectionsand an ­ output, the axon. The axon originates from the ausbook ­ also called Axon Hill. The incoming ­ bindings are Dendri ­ emanating from other neurons ausgehenvia variousbranches.

The dynamic behavior of a "normal" neuron is determined by the potential ­ tial histories in the incoming connections and the trigger voltage ("threshold") that is identical for all neurons. bestimmt. At each bar (currently 100 microseconds) ­ the upcoming voltage is queried in all incoming dendrites and sumd. When the trigger voltage is reached, an impulse with the amplitude "peakvoltage" is triggered in the axon. In contrast to the input neuron, the time to trigger is not constant, but depends on the number of incoming dendrites as well as on the course of the incoming pulses (frequency, phase).

As with the input neurons, the current electrical potential of neurons is visualized in ­ color. In addition, the potential is presented as a numerical display as a percentage of the trigger ­ voltage.

## Output neurons

Output neurons are represented by a circle without ­ bulge. You have one or more inbound but no ­ outbound connections. On the initial side, output neurons are not connected to other neurons, but control ­ actuators, e.g. muscle cells.

****When an axon is attached to an output neuron, it automatically transforms into a normal neuron. Conversely, a normal neuron automatically becomes an output neuron when the axon is removed.

The dynamic behavior of output neurons is basically the same as that of normal neurons. If the sum of the voltage potentials in the incoming connections exceeds the trigger voltage, the output neuron fires. However, since it does not have an axon, the impulse is notpassed on to other ­ neurons. Instead, an actuator is controlled, but thisis not shown in NNetSimu atthe ­ moment.

As with normal neurons, the current electrical potential of output neurons is visualized in color and by a numerical display.

## Connections

The connections between neurons are called axons or dendrites. Axons are the outputs of input neurons and normal neurons. If they continue to branch out, they are called Dendrites. In the context of this document, the generic term "connections" is usually used, if one does not specifically refer to the properties of e.g. axons.

The connections between the neurons are basically directed, i.e. they have defined start and end points. For visualization, direction arrows can be displayed, as shown in the figure for chapter 2.7 . The direction arrows canbe switched on and off via the menu bar ("View" - "Arrows on" or"Arrows off ")as well as via the context menu(see Kapitel 4.5.3 ) ).

## Branches

Branches in the static model of NNetSimu can be done e.g. by neurons that have multiple inputs. However, branches of dendrites are also possible. The ­ figure below shows different types ofwdigestion. Multiple incoming ­ connectors can converge to an outbound connection (top left) or an ­ incoming connection can split into multiple outgoing connections (right).

Dynamic behavior of branches:

In the case of converging branches, the incoming signals are added together and the ­ summation signal is sent to the outgoing branch.

In the case of splitting branches, the incoming signal is passed unchanged to all outgoing branches.

## Loops

An NNetSimu model can also contain loops. Electrical impulses then run in a circle across several neurons, so to speak.

The figure on the side shows a simple case of such a loop. The output of the lower neuron runs via the right-handbranch into ­ both the output neuron at the top right and into the neuron in the middle and from there reaches the lower neuron again.

# Aspects of the dynamicn models

## Pulse form

In the description of the different types of neurons, each of the "triggering" is the re ­ de. This means that the neuron generates an impulse on its axon, which then travels through the dendrite tree to further neurons.

In biological systems, according to Wikipedia, the impulse has the e form shownin the adjacent figure. .

Currently, NNetSimu uses a very rough approximation of this signal, which is represented by an easy-to-calculate second-degree parable. This function is determined by two parameters:

1. The amplitude, in the program "peak voltage" – the difference between the maximum ­ span and the rest potential.
2. The pulse width "pulse width" – the time ­ diffeence between leaving the rest potenti ­ as and the return to the rest potential at the end of the pulse.

The end of the pulse is followed by a phase (called "refractory period" in the program)in which no new triggering is possible, even if the incoming signals exceed the threshold voltage.

The program abstracts from some properties of biological systems:

1. The fact that the resting potential in biological systems does not have 0, but a negative stability ­ value is not taken into account in the simulation
2. The hyperpolarization during the rest ingescy phase is not taken into account..
3. In some biologineurons, a triggering währeisnot completely excluded fromthe resting phase, but the threshold voltage for a re-triggering is only greatly increased. This means that very strong impulses can prevail even during the resting durchsetzenphase.

Note: In the figure above, the amplitude of the pulse is significantly greater than the threshold voltage. If this pulse runs without attenuation to subsequent neurons, it would in any case have to cause immediate triggering there as well. The overall system would therefore always fire as soon as even a single triggering impulse occurs. To ensure a meaningful simulation, the amplitude of the pulse should be chosen lower than the threshold voltage, as opposed to the figure. At present, the contradiction between the image in Wikipedia and the requirements of the simulation is still unclear.

## Adjustable parameters

The global parameters of the dynamicen Dialog model can be changed via the "Global parameters" dialog. These are:

* Peak voltage: The amplitude of signals in mV
* Threshold: The threshold voltage for triggering a pulse in mV
* Pulse width: Pulse width in milliseconds
* Refractory period: The duration of the restephaseafter anem pulsein milliseconds
* Time resolution:The time resolution at which the model is calculated
* Pulse speed: The propagation speed of the pulses in the connections in the

Changes in the fields of the parameter dialog take effect when the "Apply" buttonis ­ applied.

As the term **global** parameters suggests, these values apply to all elements of the model. Individual settings for individual neurons or connections are currently not possible. In contrast, the pulse rate of each input neuron can be set ­ separately, see 2.2 .

If the parameter dialog interferes, it can be closed with the cross at the top right. If necessary, the dialog with theitem"View" - "Windows" - "Showparameter window" canbe reopened.

## Visual and acoustic feedback

The reproduction of the signals in the dendrites is visualized by the color change of the sections of the dendrite. In an unexcited state, the interior of a dendrite is black. The higher the voltage potential, the stronger the red portion of the section. If these sections are relatively short, the movement is fluid and even. If they are relatively long, then gradations can be seen and the movement appears angular and uneven. The length of the sections depends on the pulse speed(Pulse speed) and the time resolution, see also resolution chap. 3.2 ., as well as chap. 4.8 Point 3.

The triggering of a neuron can be signaled both visually and acoustically.

1. As with the dendrites, the interior of neurons is represented with increasing voltage potential with an ever-increasing red content.
2. In addition, the ratio of the current voltage potential to the trigger voltage (threshold) is displayed as a percentage value.
3. When triggered, the word "TRIGGER" is emitted instead of the 100% percentage.
4. In addition, an acoustic signal can be individually linked to the trigger event for each neuron. For this purpose, the item "Trigger Sound" is selected with the right mouse button on the neuron in the context menu. A small dialog box opens in which the frequency and duration of the acoustic signal can be entered.

# User interaction

The basic possibilities of userinteraction are:

* The menu bar at the top
* The status line below
* Actions with the mouse in the main area
* Keystrokes

## Menu

At the top level, the men's room offers the areas ü

* File saving and loading models, exiting the program
* Action different functions,e.g. "Analyze", "Center model", see below
* View functions related to the presentation of the program
* Options other options, e.g. to turn the acoustic feedback on and off
* Help information about the program

Under the "View" menu, the "Window refresh rate" option is included. It opens a dialog with which the refresh rate can be set. This does not mean the hardware refresh rate of the monitor, but the frequency at which the software renews the display of the network. Normally, it is not necessary to change the default value.

In the "Options" menu, the following properties of the program can be turned on and off:

* Arrows: Displaying the direction arrows on connections
* Sound: Acoustic signals when docking or deleting objects in theeditor, as well as TriggerSounds
* AutoOpen: When this option is active, the most recently edited model, zoom level and position is saved on the desktop and automatically restored the next time you start the program. When the option is disabled, the program starts with a minimal model consisting of two neurons.

### "Action" - "Analyze" - "Find loops"

This function examines the model and searches for loops (see 2.7 ). Small loops, in which only a few neurons are involved, can be created accidentally during the creation of a model and are often difficult to detect with the naked eye. However, they can affect the behavior of the model in an undesirable way.

In addition to the computational effort of larger models, this recognition function is primarily a problem of interaction with the user.

How many elements can a loop, that is captured by the analysis, have to be as large as possible?? Very small loops, which consist of only a few neurons, are particularly interesting, since these loops are most likely unintentional. Consciously set feedback loops that run over several fiber bundles, consist of a larger number of neurons. But where should the limit be set?? A user setting would be possible, but cumbersome. The function has now been implemented in such a way that small loops are searched for first. If nothing isfound, the search starts again with a slightly larger upper limit and so on, until finally searching for loops that contain all neurons in the model. The first runs go very fast. As the upper limit increases, the search requires more and more computing time. In the message area of the status line (see chap. 4.1.2 ) shows which loop size is currently being searched. The user can cancel the search at any time using the ESC key.

If a loop is found, the question arises as to how to display it to the user. In regions with a large number of neurons and dendrites, which are close to each other, sometimes even covering each other, it is difficult to detect a small loop. In addition, the entire model may not appear in the visible part of the window, and the loop may be just outside the window. The following procedure is therefore used to display a found loop:

* First, the "Center Model" action (see chap. 4.1.2 ) so that the entire model is visible and is approximately in the middle of the window,,
* At the same time, the neurons and dendrites not involved in the loop are mimicked in color, making them less noticeable. The elements involved in the loop are selected (see 4.7 ) and thus highlighted..
* Then the loop is zoomed in again so that the found loop is still completely visible and displayed as large as possible in the middle of the window.

The user can now resolve the loop, e.g. by deleting neurons or dissolving connections.

To return to normal mode, where the model is calculated and dynamically visualized, use the "Run" function in the status line.

Afterwards, the "Analyze" function should be used to test whether there are any other loops in the model.

### "Action" - "Analyze" - "Find anomalies"

In addition to loops, there are other anomalies that can inadvertently creep into amodel. These include "degenerate"“ dendrites,inwhich, for example, two sections lead into a crease, but none leads out, or vice versa. While such impurities do not affect the behavior of the model, as they can never flow impulses, they cause unnecessary computational effort and make the model confusing. The function searches for such degenerate dendrite fragments and visually highlights them just like "find loops"“ optisch hervorbyselecting them. Usually you will delete these pieces, which is quite easy in one step, since they are ready selected (context menu "Remove selected objects"). After correcting such an error, the function should be called again until no further anomaly is found.

### "Action" - "Center model"

If the model is completely outside the visible window, which can occur due to accidental movement or an error reading stored data, it is difficult to find the model again. The Center model function brings the model to the center of the window and adjusts the zoom level so that the model is fully and sufficiently large.

## Status bar



The status line offers from left to right

* A display of elapsed time in the model. Depending on the setr slow-motion-level, this time runs considerably slowly than the real time.
* Buttons to run the model in single steps (clock rate of model calculation), start the model and stop the model
* The display of the current slow-motion level.
* Buttons to make the calculation slower or faster.
* An area where messages can be displayed.

## Keystrokes

If the mouse cursor is over an input neuron, the pulse rate of the input neuron can be increased or decreased with the "+" and "-" keys in increments of 1/100 Hertz. This function has an immediate effect on the model and is a faster alternative to the dialog that is called from the context menu.

## Mouse actions in the main area

To be used

* the left mouse button
* the right mouse button
* the mouse wheel (if available)
* and of course the movement of the mouse cursor

To apply an operation, such as moving or calling the context menu (see 4.5.3 ),to apply it to a­subject, it must be selected. The selection is made automatically assoon as the mouse cursor is over the­object. The selected object is visualized by a changed color of the object border, see also chapter 4.6 .

## Functions of the editor

The editor allows the user to interactively modify the static model. The­other tool is context menus, which are called via the right mouse button,­see 4.5.3. and drag with the mouse (see chapter 4.5.1 ).

Available features:

* Create a new (input) neuron that initially has no connections
* Attach an inbound or outbound connection to a neuron
* Branching an inbound or outbound connection from a connection
* Insert a neuron into a connection
* Delete an (input) neuron or a connection.

The program ensures that the model is always in a consistent state. This means that

* Input neurons have no incoming and output neurons have no outgoing connections (the latter automatically become normal neurons when they get an output).
* neuron has more than one immediate exit. The only output, the axon, can branch, but from the neuron itself only the axon itself immediately emerges.

These conditions do not always provide all conceivable operations for all objects. For example, for a neuron that already has an output, the option "Add outgoing dendrite" is not offered at all.

The elements of the static model can in principle be arranged on a two-dimensional plane. It is usually useful to concentrate the input neurons at one or ­ more places in the periphery of the model,but the program itself does not impose any restrictions on the ­ subject. The same applies to the output neurons.

### Move objects with the linken mouse button

The left mouse button can be used to move the entire model or individual neurons: verschoben werden:

To move the model as a whole:

* move the mouse cursor to a location where no objecte (neuron, e etc. ) befin ­ det
* press and hold the left mouse button
* move the mouse cursor. The entire model moves with the mouse cursor.
* When the desired position is reached, release the left mouse button

Move individual neurons or dendrite branches:

* Move the mouse cursor to the object to be moved
* when the object is detected, its color changes as optical feedback
* press and hold the left mouse button
* move the mouse cursor. The selected object moves with the mouse cursor. Objects that are directly connected to the moving object are dragged along. Dendrite branches usually change their direction and length..
* When the desired position is reached, release the left mouse button

Move a group of objects at the same time:

* Define the desired group of objects as a selection (see 4.7 )
* move the mouse cursor to a location where no objects (neuron, etc.)­are located
* press and hold the left mouse button
* move the mouse cursor. The selected objecte moveen with the mouse cursor.
* When the desired position is reached, release the left mouse button

### Left mouse button (double click)

Double-clicking on the left mouse button selects an object (neuron or dendrite section)or adds it to an existing selection, see 4.7 .

### Right mouse button

The right mouse button calls the **context:ü** which offers different functions depending on which object the mouse cursor is currently, located.

For example, you can use the contextü

* input-neurons adjust the pulse rate of this neuron
* add a neuron to this new dendrite
* insert a neuron from a connection. The original compound is split into two ­ bonds, one of which flows into the newly generated neuron, while the other flows from the neuron.
* neuron whose connections release "Disconnect",
* convert an input neuron into a normal neuron and vice versa. However, the conversion of a normal neuron into an input neuron is only possible if the neuron does not have any incoming dendrites.
* of a neuron define an acoustic signal that is triggered with when the neuron is triggered.
* Etc.

Objects of the model can also be removed from the context menu ("Remove".

The figure on the side shows the context ­ menu of the connectionbetween two ­ bindung zwischen zwei Neuro ­ neurons.

The "Add output neuron" option creates a point of engagement at the point where the mouse cursor is gera ­ de befin ­ det and attaches an outgoing dend ­ rite.

The "Add input neuron" optiongeneratesan eoutgoing branch in the sameway.

"Insert neuron" creates a normal ­ neu ron, whichis inserted into the binding at the cursor ­ position.

### Wheel

You can zoom in with the mouse wheel.

* Move the mouse wheel forward to enlarge the model
* Move the mouse wheel backwards to shrink the model.

When zooming, the position atwhich the mouse cursor points, remains unchanged. This allows you to intuitively look at an object that you are interested in, pointing ­ the mouse cursor at the object and moving the mouse wheel forward.

## Optical and acoustic feedback

The NNetSimu editor uses visual and acousticapplicationsekte, to help the user find orientation and provide feedback on operations performed.

As visual feedback, the color of the selected object that the mouse cursor is pointing to changes. The contextual thatis activated with the right mouse button (see chapter 4.5.3 ) always refers to this object. There are situations, such as close-to-one or even superimposed objects, where itwould be difficult to uniquely recognize the ­ selected object without this visual feedback.

If objects are to beBconnected to each other, such asconnecting the loose end of a dendrite to a neuron, two different color markers are required. The object moved by the mouse cursor, in this case the end of the dendrite, isr ­ color-coded as described in the paragraph. Asblong as this object isover anotherobject to which it ­ can dock, the target object is highlighted in a different, more conspicuous color. This target is only marked if docking is really possible. This allows the user to detect early, when he is about to perform an inadmissible operation, e.g. to connect an incoming dendrite to an input neuron.

Further visual feedback by blinking occurs when selecting a set of objects, see chapter 4.7

Acoustic feedback is provided for some operations, such as docking objects, disconnecting, and deleting objects. The acoustic signal canü be switched on and off via the main "Options" - "Sound on" or "Sound off".

## The miniature window

A small "Mini Window" window displays the entire model in a thumbnail view at any time. Even if only part of the model is visible in the main window by zooming in, the mini-window displays the entire model in a correspondingly reduced size.

The area of the model visible in the main window is indicated by a transparent colored rectangle in the mini window.

To quickly navigate the model, the transparent rectangle in the mini window can be moved with the left mouse button. The snippet visible in the main window changes accordingly.

The mini window itself can be freely positioned on the screen. The SystemMenu in the upper left corner can also be used to resize it.

## Selection of objects (Select/Deselect)

### Definition "Selection"

For various operations, it is necessary to first select the object or select the objects on which the operation is to be performed. The term selection (or select) is used in this document for different operations, each meaning is derived from the context. However, when we talk about selection (select), this specifically refers to the meaning described in this chapter.

Merk rule: Selection is what flashes! (Declaration at the end of this chapter).

In the simplest case, an operation refers to how to move, delete, etc. to a single object (neuron, dendrite section, crease). In this case, the selection is done simply by placing the cursor over the object. The correct positioning is indicated by a color change of the object in question (see 4.6 ). This type of selection exists only as long as the cursor is over the object and is automatically removed when the cursor leaves the object. By calling the context menu, various operations can be applied to the selected object.

For an operation to be appliedto several objects at the same time, this volatile typeof selection, is not sufficient. The user must be able to expand or restrict the selection according to his or her wishes. Ifthe desired set of objects is defined,, it triggers the operation. Such a selection, which persists even if the cursor is moved further, is called **selection**.

### Operations on selections

Various operations can be appliedto an existingselection:

* Marking (see 4.8 )
* Move: The move operation with the leftn mouse button (see 4.5.1 ) on the background automatically affects only the selection if one is defined. Normal move operations for individual objects work as usual if the cursor is over the object.
* Delete ("Remove selected objects"). This function should not be confused with"Deselect all"! The latter removes the Selected property, but leaves the objects in the model. Remove selected objects,on the other hand, removes the selected objects completely from the model.
* Copy selection“): Each selected object (neurons and dendrites) creates a duplicate located in the same place as its original, i.e. the copies are exactly above the originals and obscure them. The originals are no longer selected, but the copies are now selected. This allows you to move them directly to another location with the left mouse button.   
  The copies are connected to each other, as are the originals. However, the connections to the environment are not present, the totality of the copies is isolated from the rest of the model. Connections must be made by hand.   
  The copies inherit all static properties from their originals, e.g. the pulse rate of input neurons, acoustic trigger signals or markers (see 4.8 ).

After the operation is complete, the selection is still available. The user can apply another operation to the selection, or remove the selection.

### Create, expand and remove selections

For the creation and manipulation of a selection a whole range of tools are available,depending on whether one prefers towork with thekeyboard, with menus or only with the mouse::

1. Select all objects of the model

* Keyboard: Hold down CTRL-A (the fastest method) key "CTRL" or "CTRL" on German keyboards and press the key "A"
* Use the rectangle function (see point 5) to draw up a large rectangle that contains the entire model
* Main menu: "Edit" - "Select all"

1. Remove selection completely

* Main menu: "Edit" -"Deselect all"
* Keyboard: ESC key (the fastest method)

1. Selectindividual objects /remove them from selection

* Context menu "Select"or "Deselect"
* Double-click with the left mouse button (the fastest method)

1. Select subtrees (one neuron and all outgoing dendrites)

* Context menu of a neuron: "Selectsubtree"

1. Select rectangular neckline

* Hold down and drag right mouse button

1. Selection by analysis function

* The analysis functions (see chapters 4.1.1 and 4.1.2 ) also generate a selection as a result

### Visual marking of a selection

In order to make the selected objects clearly distinguishable from the other objects,they change their color with a frequency of 5 hertz. This flashing is clearly visible, but has no interaction with the simple selection of individual objects. This means that even while a group of objects is selected and flashing, the model can be used as normal. For example, einzelne objects can be moved or deleted, etc.

## Marking of objekten

The user has the possibility to permanently mark neurons and dendrites with a color mark.

This type of marking must be distinguished from the selection of objects (see chapter 4.7 ). The latter is a tool for determining the scope of an operation. After the operation is complete, the selection is removed. In contrast, the marking of apermanent marking of a group of objects(neurons, dendrites) that the user wants to keep in mind.

In order to create a marker, however, you need a selection!

One possible course of action is::

* Select the objects **to**select. All methods described in 4.7.3 can be used for this purpose.
* Select Right Mouse Button and Mark selection
* Right mouse button and "Deselect all" (or use the ESC key instead)

When removing a mark, you proceed accordingly:

* Select all selected objects to remove the selection. E.g. by right-clicking a rectangle
* Right mouse button and"Unmark selection" select
* Right mouse button and "Deselect all" or ESC button again

# Limitations and limitations of the model

NNetSimu cannot represent the artificialnervous system in itsfull complexity. ­ Restrictions are necessary in several respects:

1. Restrictions in the level of detail: For example, no synapses are currently ­ modeled, but the connections between neurons are treated as if the dendrites were immediately entering a subsequent neuron. Details of the axons, such as the myelin sheath or the Ranvier lace-up rings, are also abstracted.
2. Quantitative limitations: While the human brain consists of about 86 billion ­ neurons, the capacity of NNetSimu must be drastically lower. At current development, it is not yet possible to say what the maximum possible number can be ­ made, but it will probably be smaller than a million neurons.
3. Limitations in temporal resolution: NNetSimu must calculate the dynamic processes in the network sequentially in a certain time cycle. The shorter this time ­ is clock, the greater the computational effort to run the model with a certain ­ speed.. In principle, it is possible to calculate in any small increment, but the model will only run in extreme slow motion. Conversely, too long is ­ selected clock unfavorable for the representation of fast-moving processes, e.g. increase and decrease of the action potential when triggering a neuron (see Nyquist-Shannon sampling theorem). At present, a clock of 100 microseconds is g voreinpre-set. The value can be changed in the parameter window, see chapter 3.2
4. Limitations in the variety of neurons, dendrites, etc. NNetSimu ­ currently contain only a small number of elementary types of neurons described in Chapter 2 " The Elements of model s". If necessary, additional special cases such as thosefound in the registry can be ­ added.

Limits to the size of the model that can be represented are due to the limited computing capacity of the computer being used. In particular, calculating the model for each time cycle is a limit to the size of the model. Influencing factors for this are:

* The computing power of the CPU
* The number of objects of the model. In this sense, each neuron or Input neuron is an object, but also every point of condensation or kink point of dendrites and each dendrite section between such points or neurons. The figure for chapters 2.7 For example, the model shown consists of 10 objects: an input neuron, two normal neurons, an output neuron, a point of conmitting and 5 dendrite sections.
* The time resolution of the model.
* The set slow motion

Acomputer with Intel Core i5 4670 CPU achieves the following results:

The test object was a model with a total of 187 objects, including 30 neurons. In order to push the computer to its limits, the timeresolution parameter has been set to 2seconds (factor 50 compared to the default value). The highest possible speed was a slow motion factor of about 10 (factor 10 faster than the default value). This means that there are still reserves of about a factor of 500.

In other words, if one takes the numberof neurons as a yardstick, a model with about 30 \* 500 = 15000 neurons on this computer would have to be possible, correspondingly less on a slower computer.

# Change history

**Changes in version 2020-04-28:**

1. New function: Select all neurons with trigger sound
2. New function: Remove all trigger sounds / Remove selected trigger sounds
3. Some minor optimizations
4. Additional information in the performance window

**Changes in version 2020-04-22:**

1. New function: Delete all selected objects, see 4.7.2
2. New function: Copy all selected objects, see 4.7.2

**Änderungen in der Version 2020-04-17:**

1. A visual signal when a neuron is triggered (short flashing) has been implemented. If an acoustic signal is connected to the neuron, then acoustic and visual signaling occurs simultaneously. See Github Issue #27. The previous output "TRIGGER" has been removed. Instead, the current potential is always displayed in percent, even if it is above 100%.
2. New Function Move All Selected Objects, see GitHub Issue #30. The Move operation (holding down and dragging left mouse button) on the background automatically affects the selection only if one is defined. Normal move operations for individual objects work as usual if the cursor is over the object.
3. This document: The description of selection 4.7 and marker 4.8 has been extended and clarified. Previously, the term "selection" had not been used consistently, which could lead to misunderstandings   **changes in version 2020-04-16:**
4. Fixed an error resetting the model. This error has resulted in the model not being restarted in sync after changing the pulse rate of an input neuron in some situations.
5. A new Center model feature is available, see 4.1.3 . It is virtually a waste product of the analysis functions. The meaning of this feature is described in GitHub #17.
6. New functions for converting neurons into input neurons and vice versa have been implemented. They can be selected from the context menu, see 4.5.3 . The meaning of this feature is described in GitHub #21.
7. Another function to analyze the model that finds "degenerate" dendrites has been implemented, see 4.1.2 and GitHub #24
8. The editor has been given different methods to select multiple objects. The possibility to select a lot of objects (neurons, dendrite fragments) is a prerequisite for manipulating such a selection, e.g. color marking.
9. Marking of neurons and dendrites has been implemented, see 4.8 and GitHub #23. The recommended way of working for marking "regressing neurons" is described in Github #23.

The Refresh rate function, which can be used to set the refresh rate of the graphics window, has been removed from the context menu and can now be reached from the main options menu. This function is rarely needed and the context menus become a little clearer.   **Changes in version 2020-03-23:**

1. The Open Points chapter, which included possible extensions, questions, and known bugs, has been removed from the document. The open points are now managed exclusively in GitHub.
2. At the end of the document, the change history documents the descriptions of previous versions.
3. GitHub Issue #17 Center model has been implemented, see chapter 4.1.2 . I suspect that when i automatically open the most recently saved model (see Issue #2), it is not visible because for some reason it lands outside the window. If this presumption is true, the model can be made visible again with this function. Regardless of this error analysis, the function can also be useful if you have accidentally moved the model to "Nirwana" and are struggling to find it again.
4. To get to the bottom of the issue with Issue #2 "Open last saved model automatically" additional error queries have been built in. A message window may appear when the program is started. If necessary, please report me.
5. Each time the pulse rate of an input neuron is changed, the entire model is reset, i.e. the voltage levels of all neurons and dendrites are set to 0. This means that the model always starts with reproducible starting conditions. See Issue #14 "Docking of Areas (Coherent or Different) - Natural Frequency"

**Changes in version 2020-03-20:**

1. GitHub Issue #5: Starting with this release, a readme.txt file will no longer be included, but all new features will be described in the documentation (this Word file).
2. GitHub Issue #15: The pulse rate of input neurons can now be changed very easily and in small increments by bringing the mouse cursor over the input neuron and pressing the +/- keys of the keyboard. See Chapter 4.3 .
3. GitHub Issue #1: A new feature has been implemented that scans the model for (unintentional loops). See chapter 4.1.1