

Neuron electrophysiology: From biology to modeling

Simula summer school

26.6.2018

Tuomo Mäki-Marttunen

Outline

- Part I: Neuron biology in brief
- Part II: Fundamentals of multicompartmental neuron modeling
- Part III: Modeling with NEURON software: Examples

Part I: Neuron biology in brief

Neuron electrophysiology

- Many shared features with cardiac cell electrophysiology
 - Expression of ion-channel encoding genes that allow electrogenesis
 - Cell hyperpolarization mediated by voltage-gated K⁺ currents
 - Ca²⁺ an Na⁺ currents contribute to depolarization
 - Communication with neighboring cells through gap junctions (electrical synapses)
- Many different features too
 - Action potentials in neurons are sharper than in cardiac cells, lasting typically 0.4-1.5 ms
 - Neurons express genes that
 - regulate the neurite growth and branching (e.g. neurotrophin, CDK5 and actin pathway genes)
 - contribute to synaptogenesis and encode synaptic ion channels

[Felfly, H., Xue, J., Zambon, A. C., Muotri, A., Zhou, D., & Haddad, G. G. (2011). Identification of a neuronal gene expression signature: Role of cell-cycle arrest in murine neuronal differentiation in vitro. *American Journal of Physiology-Regulatory, Integrative and Comparative Physiology.*]

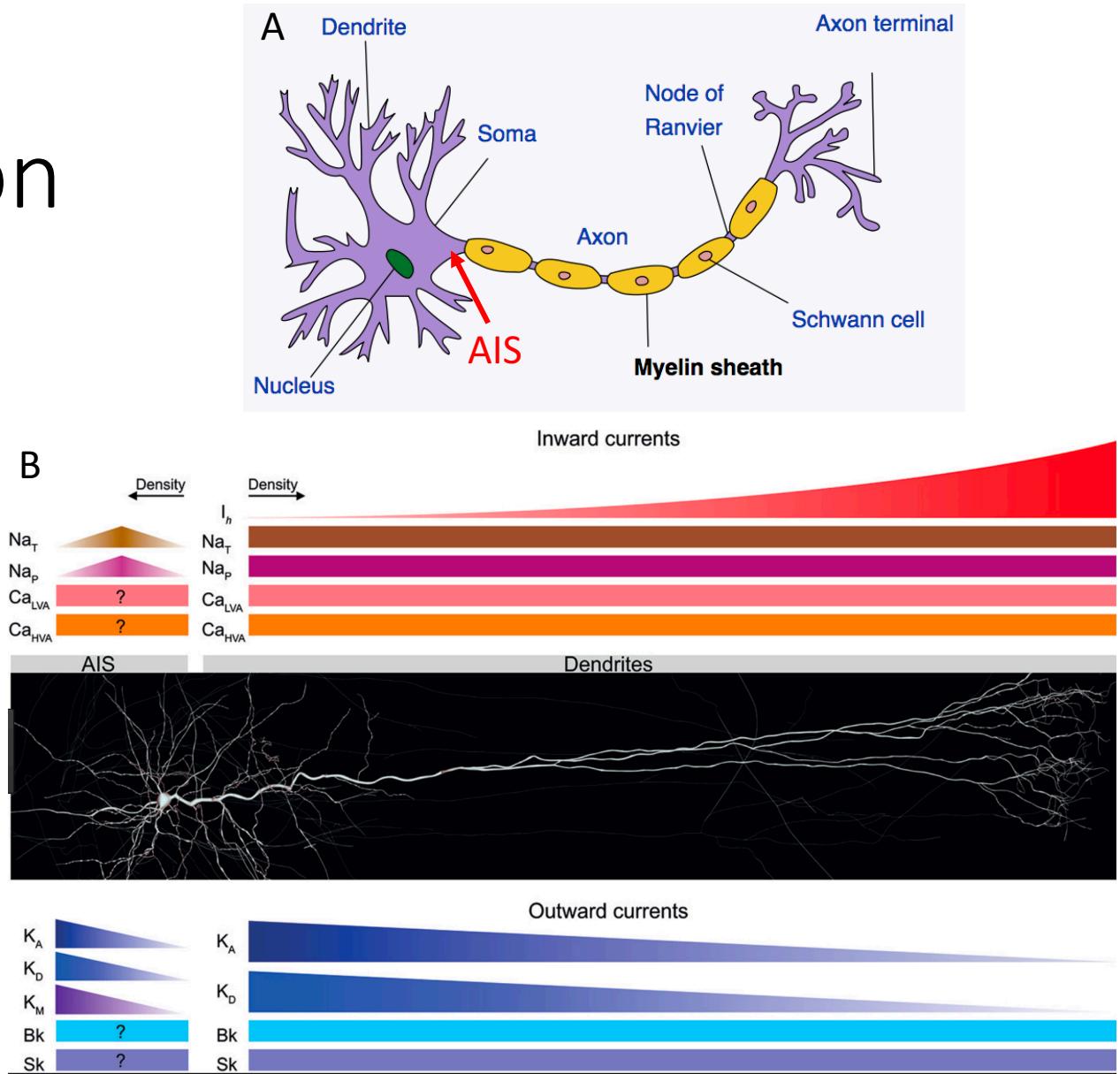
[Bullock, T., & Horridge, G. A. (1965). Structure and function in the nervous systems of invertebrates.]

Neuron electrophysiology

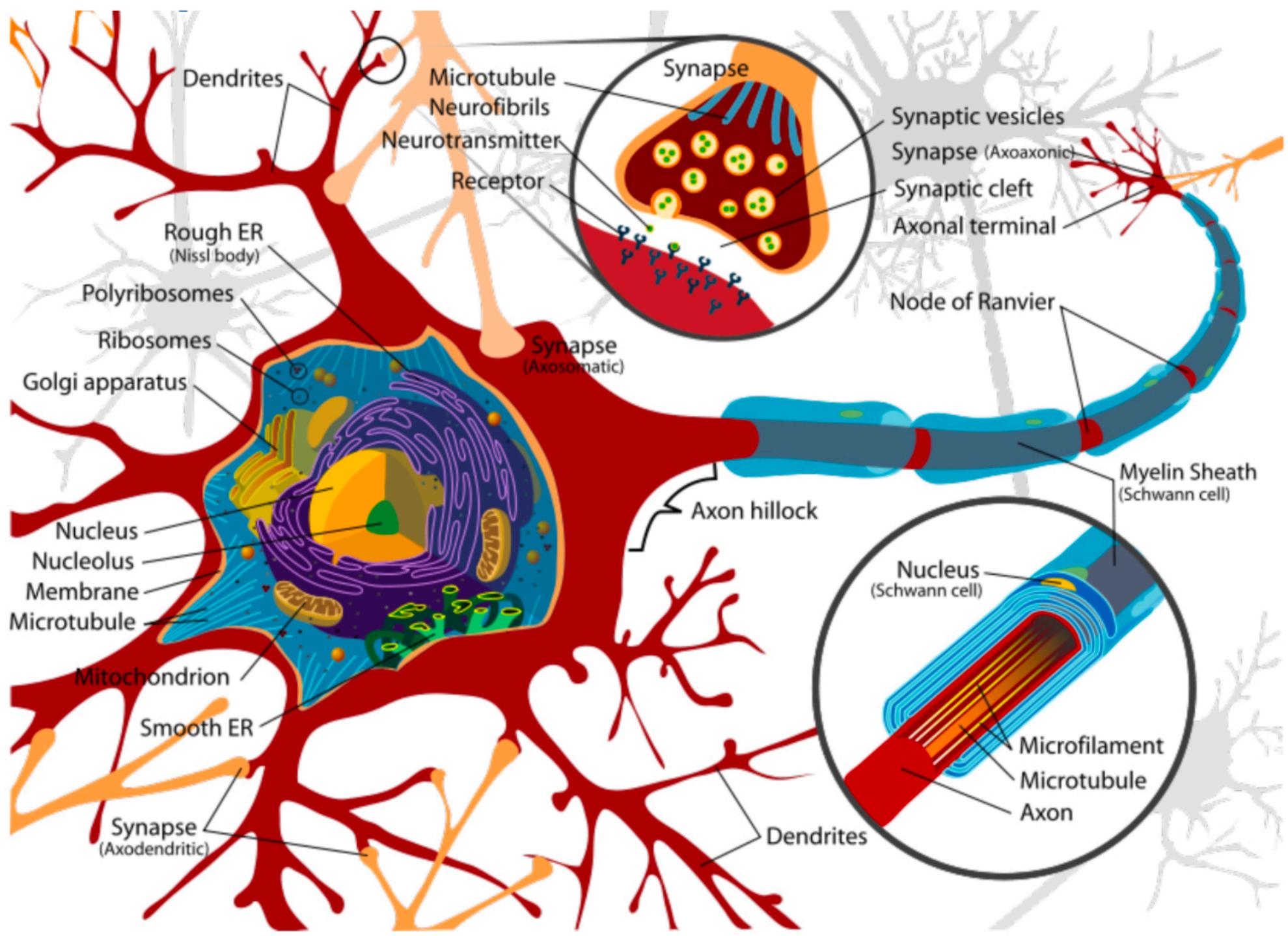
- Other distinct features
 - Instead of sarcoplasmic reticulum (SR), neurons have endoplasmic reticulum (ER)
 - Neuromodulators (messenger molecules released from neurons) such as dopamine, serotonin, norepinephrine, acetylcholine and histamine have an important role in regulating the neuron activity
 - Chemical synapses and synaptic plasticity
 - Non-uniform expression of ion channels across the cell morphology
 - Also synaptic inputs can be differentially localized

Ion channel distribution

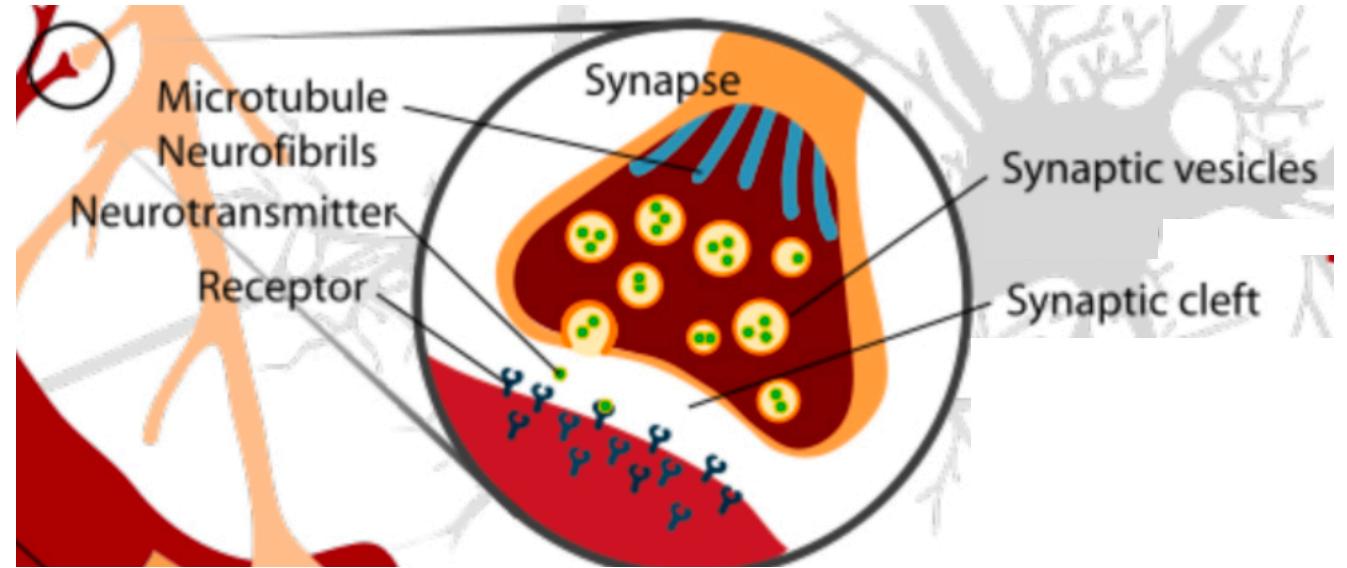
- Ion channels localized to specific parts of the neuron
- Action potentials are typically generated in the axon initial segment (AIS)
 - Characterized by high density of Na^+ and K^+ channels



[Ramaswamy, S., & Markram, H. (2015). Anatomy and physiology of the thick-tufted layer 5 pyramidal neuron. *Frontiers in cellular neuroscience*, 9, 233.]

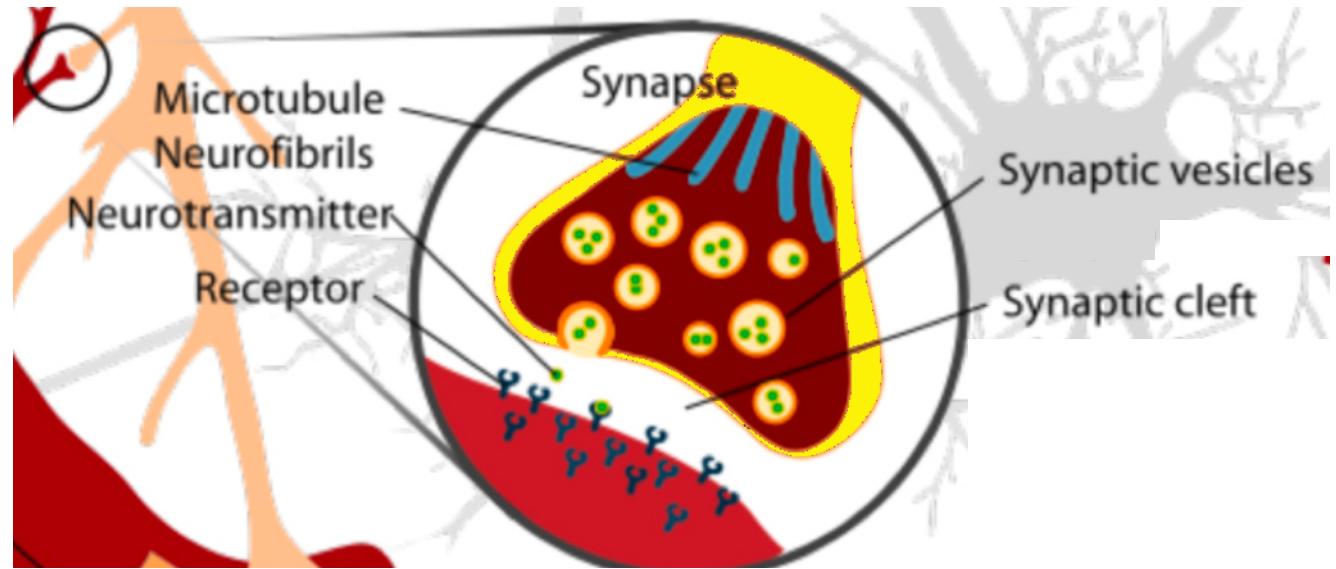


Chemical synapse function



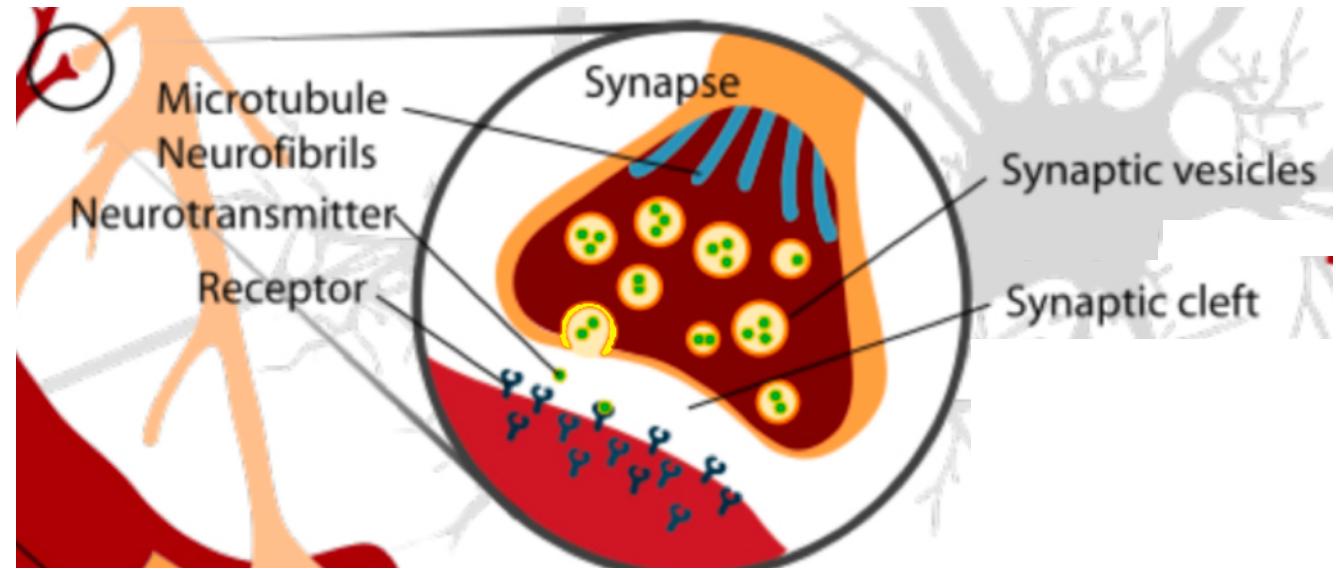
Chemical synapse function

- 1) Activation of the presynaptic axon



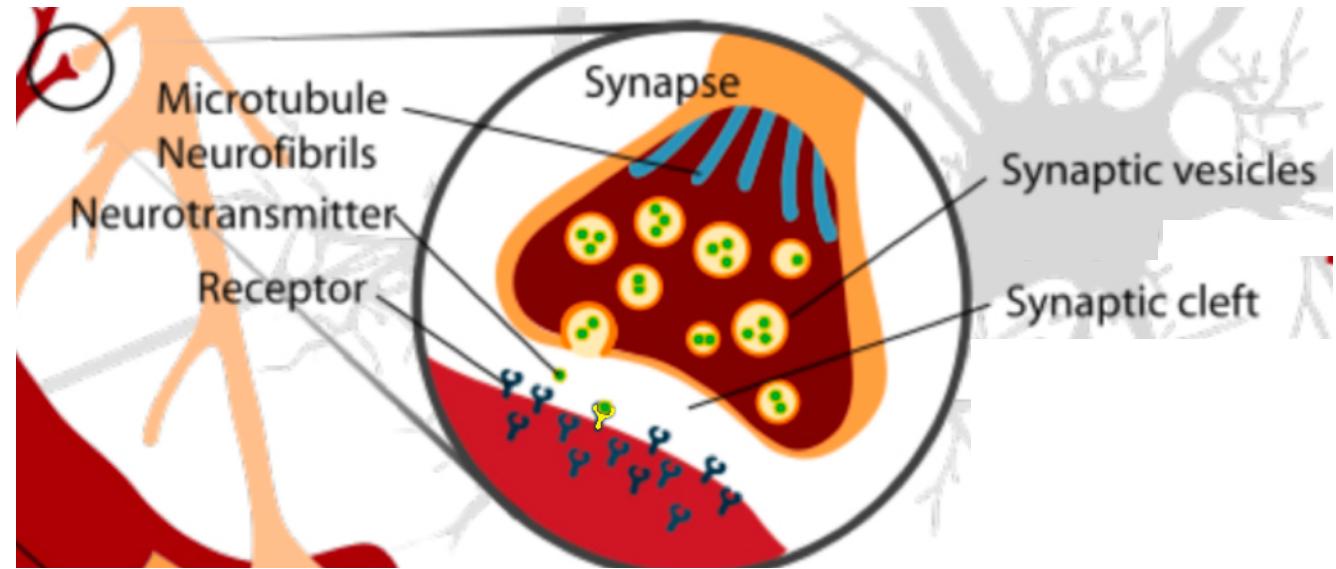
Chemical synapse function

- 1) Activation of the presynaptic axon
- 2) Fusion of neurotransmitter containing vesicles into the membrane and release of the neurotransmitters



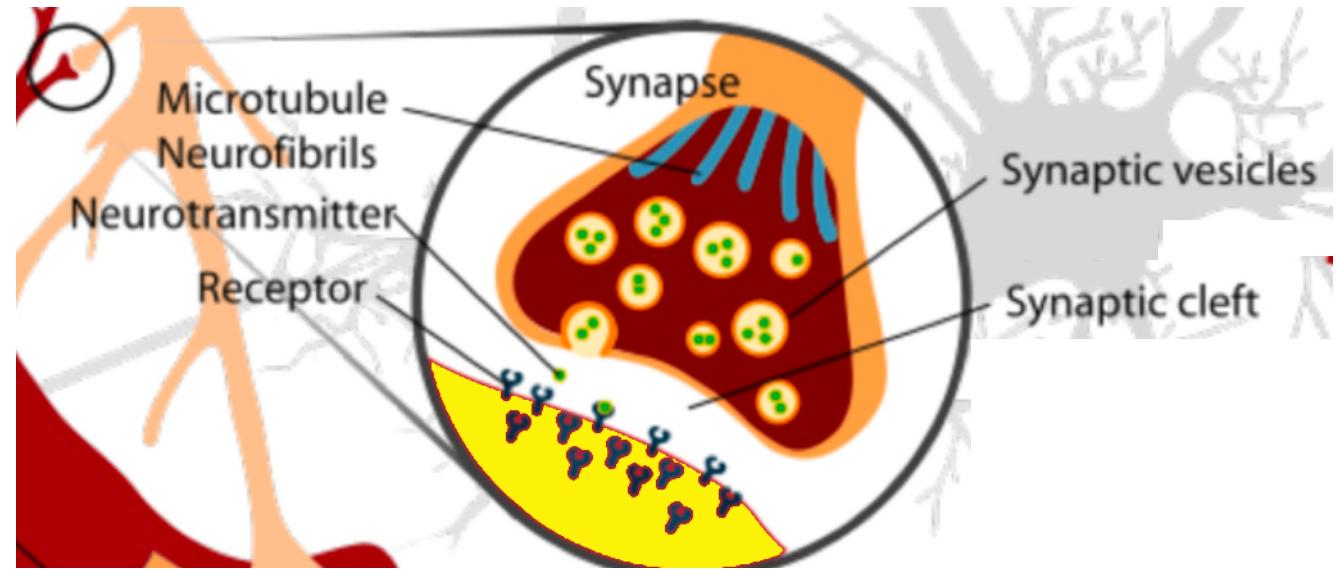
Chemical synapse function

- 1) Activation of the presynaptic axon
- 2) Fusion of neurotransmitter containing vesicles into the membrane and release of the neurotransmitters
- 3) Neurotransmitter binding and opening of the post-synaptic ion channels
 - Excitatory transmission: Glutamate binds to AMPA, NMDA or kainate receptors
 - Inhibitory transmission: GABA binds to GABA receptors



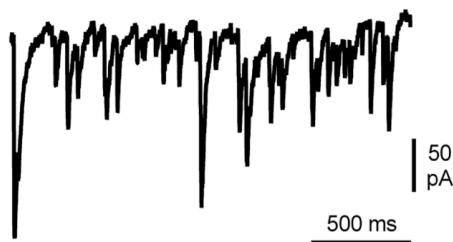
Chemical synapse function

- 1) Activation of the presynaptic axon
- 2) Fusion of neurotransmitter containing vesicles into the membrane and release of the neurotransmitters
- 3) Neurotransmitter binding and opening of the post-synaptic ion channels
 - Excitatory transmission: Glutamate binds to AMPA, NMDA or kainate receptors
 - Inhibitory transmission: GABA binds to GABA receptors
- 4) Depolarization of the post-synaptic membrane



Synaptic currents

A1 spontaneous inward PSCs at -60 mV



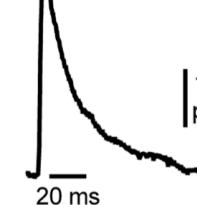
2 averaged inward PSC



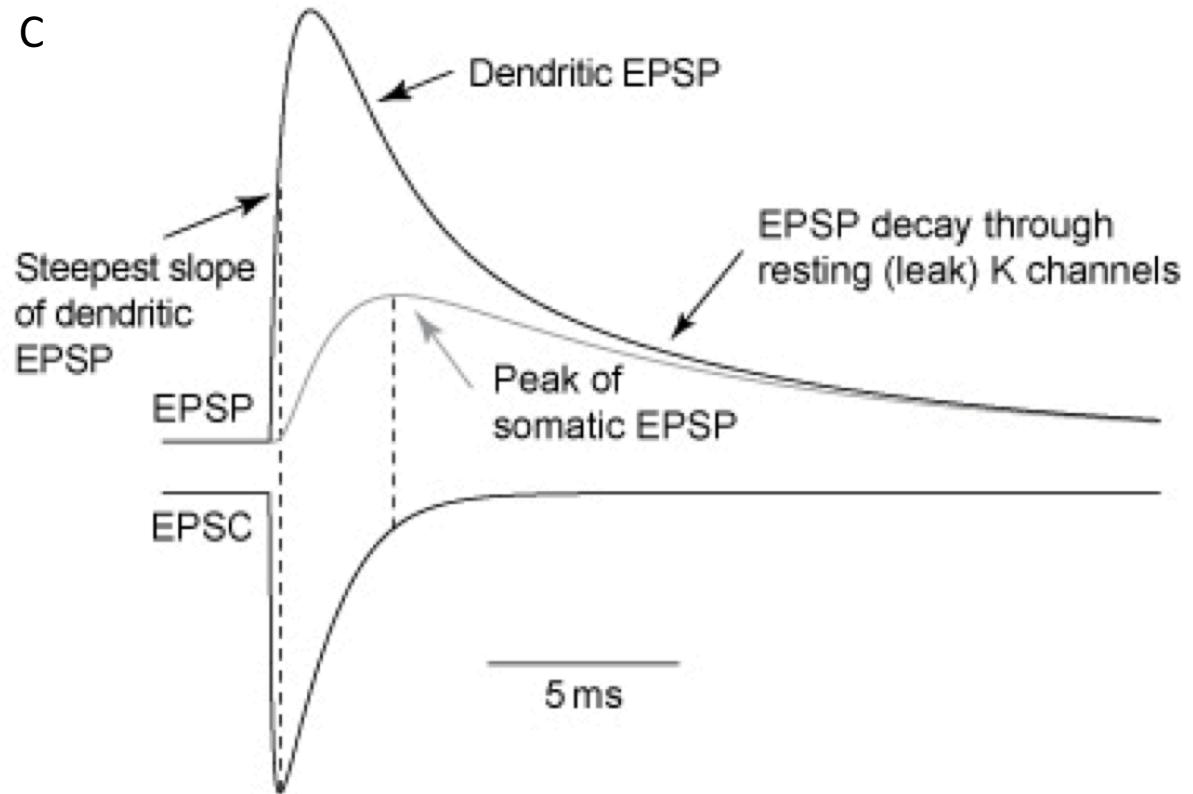
B1 spontaneous outward PSCs at 0 mV



2 averaged outward PSC



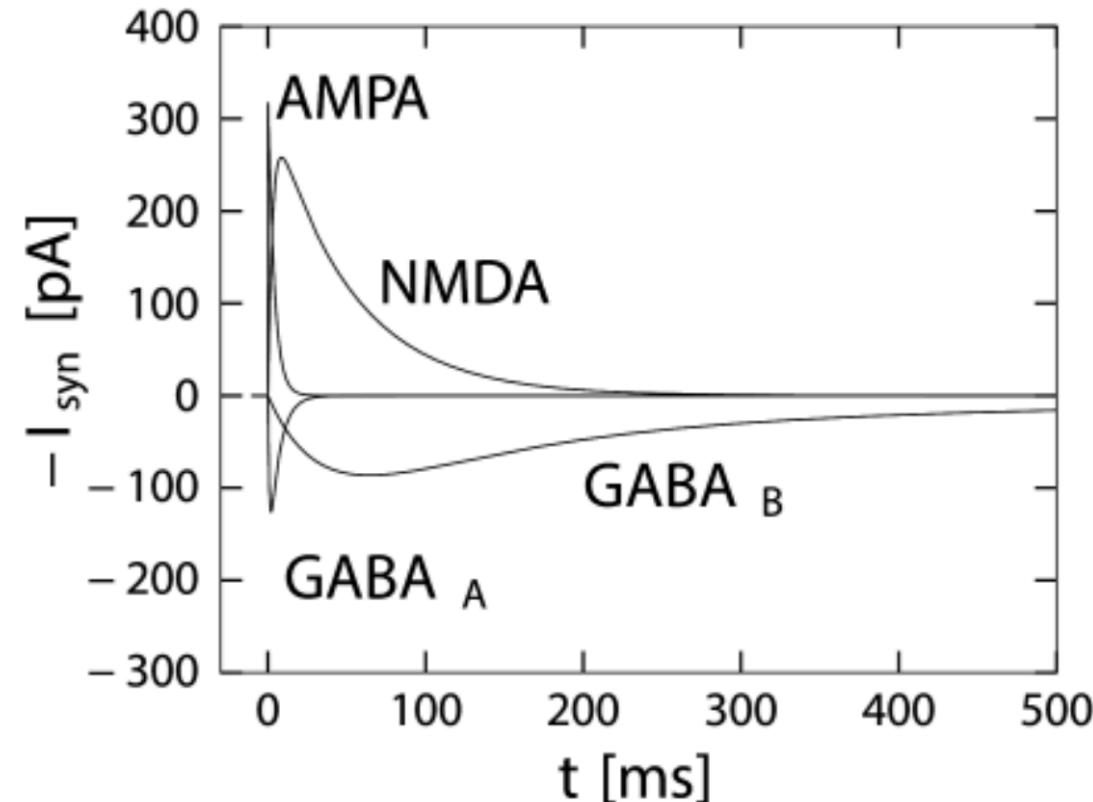
[Fogarty, M. J., Hammond, L. A., Kanjhan, R., Bellingham, M. C., & Noakes, P. G. (2013). A method for the three-dimensional reconstruction of Neurobiotin™-filled neurons and the location of their synaptic inputs. *Frontiers in neural circuits*, 7, 153.]



[Spruston, N., Häusser, M., & Stuart, G. (2012). Information processing in dendrites and spines. In: Squire, L., Berg, D., Bloom, F. E., Du Lac, S., Ghosh, A., & Spitzer, N. C. (Eds.). (2012). *Fundamental neuroscience*. Academic Press.]

Synaptic currents

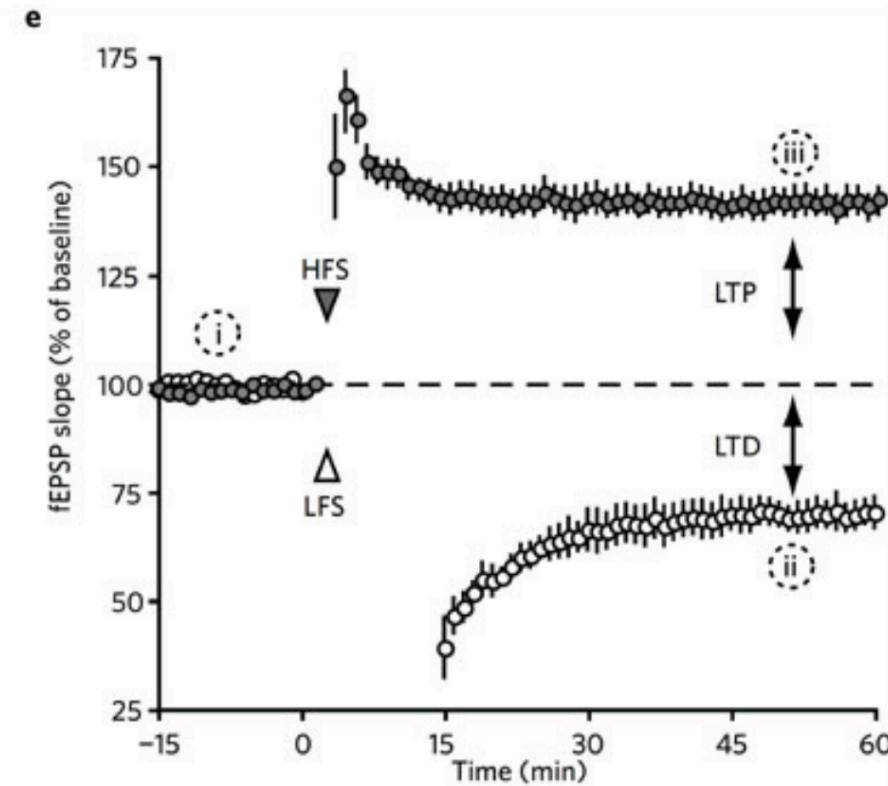
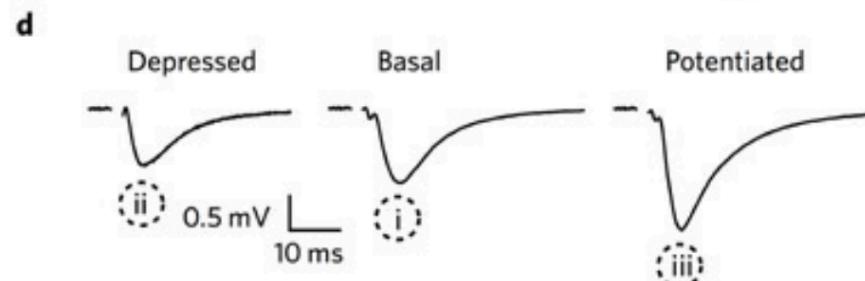
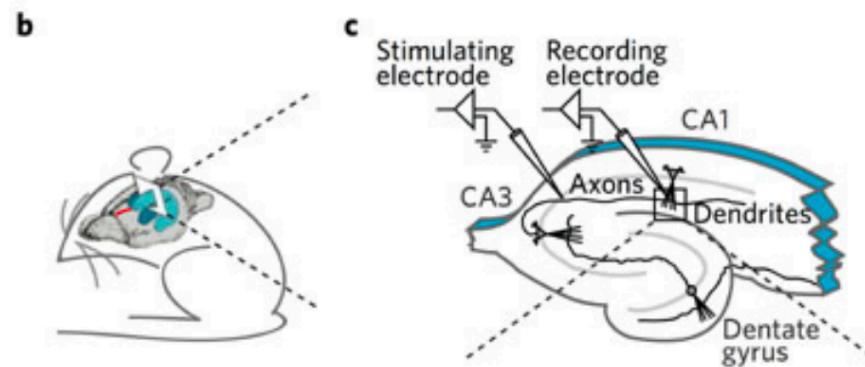
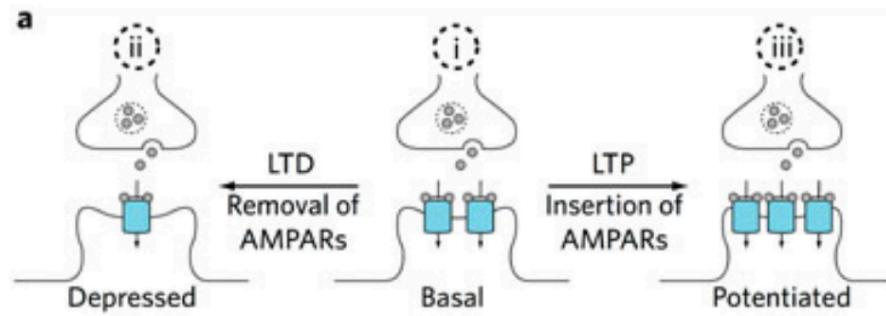
- AMPA and NMDA are agonists that mimic effects of neurotransmitter glutamate
- GABA is a neurotransmitter itself
- AMPA receptors and NMDA receptors conduct Ca^{2+} and other cationic currents
- GABA receptors conduct Cl^-
- AMPA and GABA_A -mediated currents are fast, while NMDA and GABA_B -mediated currents are much slower



Synaptic plasticity

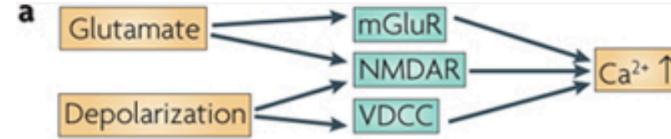
- Strengthening or weakening the input of a synapse
- Required for memory and learning
- Different time scales
 - Short-term plasticity (< 1s)
 - Enhanced fusion (short-term facilitation) or depletion (short-term depression) of the synaptic vesicles in the pre-synaptic terminal
 - Early long-term plasticity (< 1h)
 - Phosphorylation and trafficking of synaptic receptors
 - Late long-term plasticity ($\lesssim 1d$)
 - Transcription and protein synthesis
- May lead to structural changes in the synapse, i.e. change in size or complete removal of the synapse

Long-term synaptic plasticity

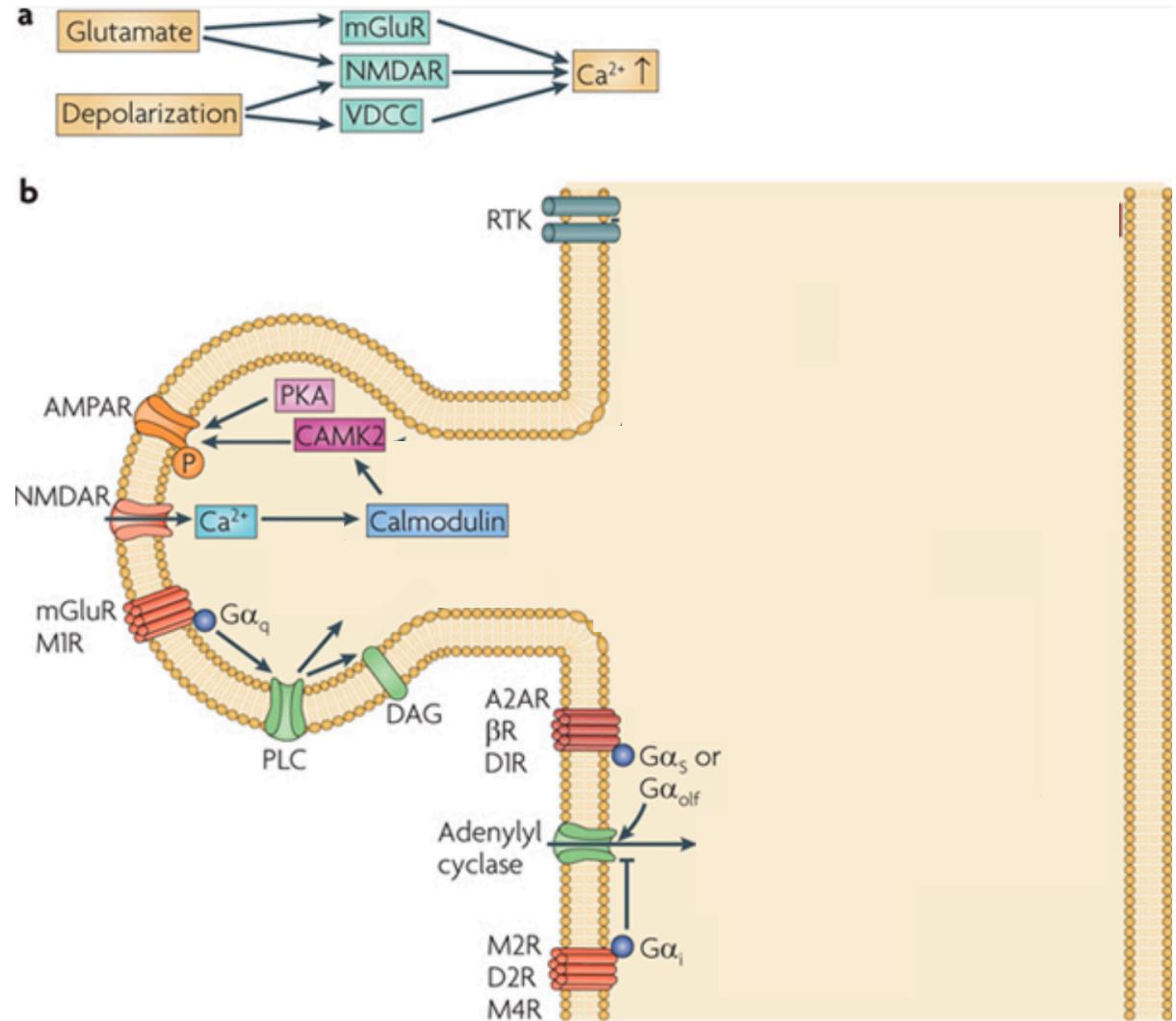


[Fleming, J. J., & England, P. M. (2010). AMPA receptors and synaptic plasticity: a chemist's perspective. *Nature chemical biology*, 6(2), 89-97.]

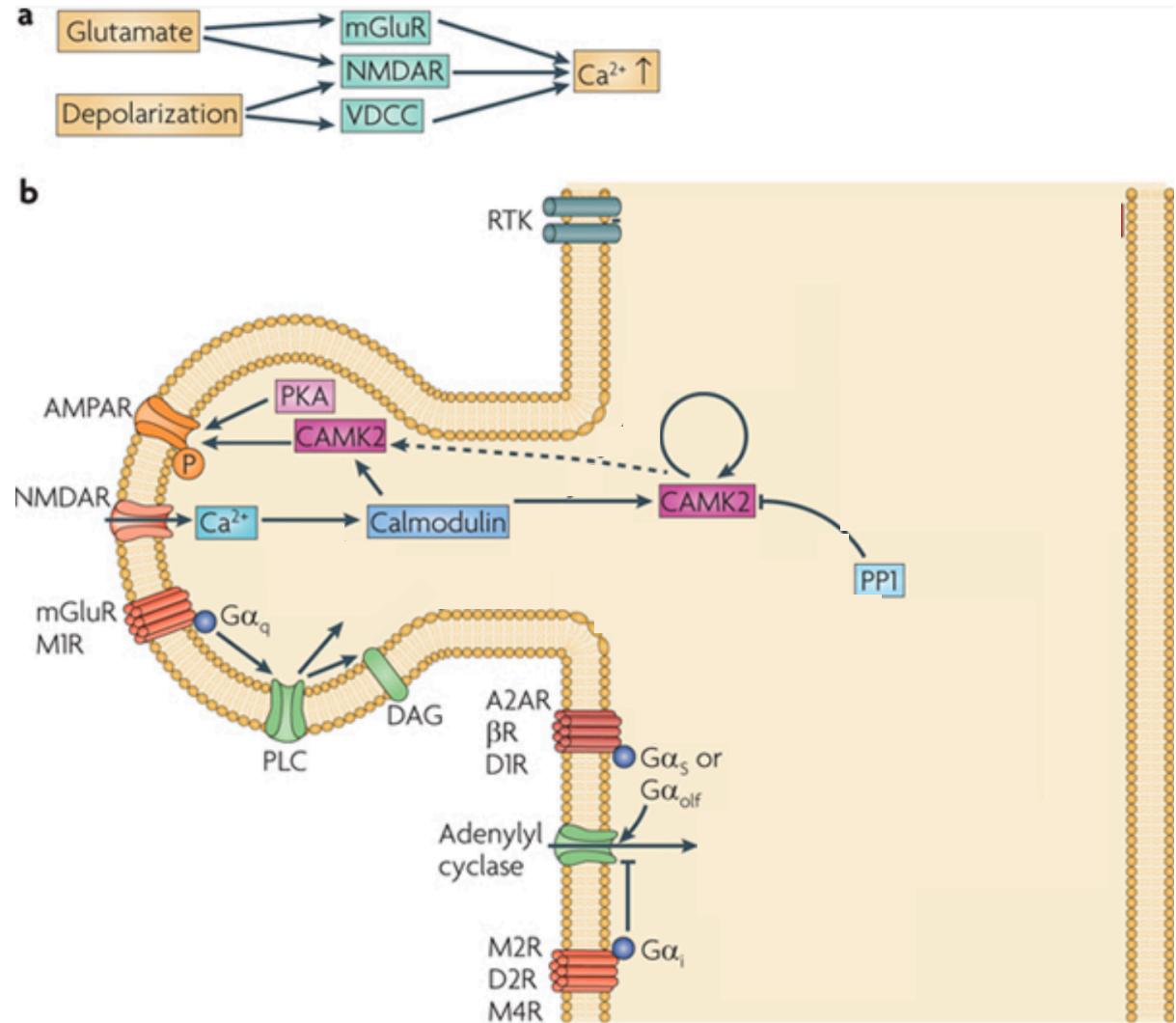
Long-term synaptic plasticity: cellular mechanisms



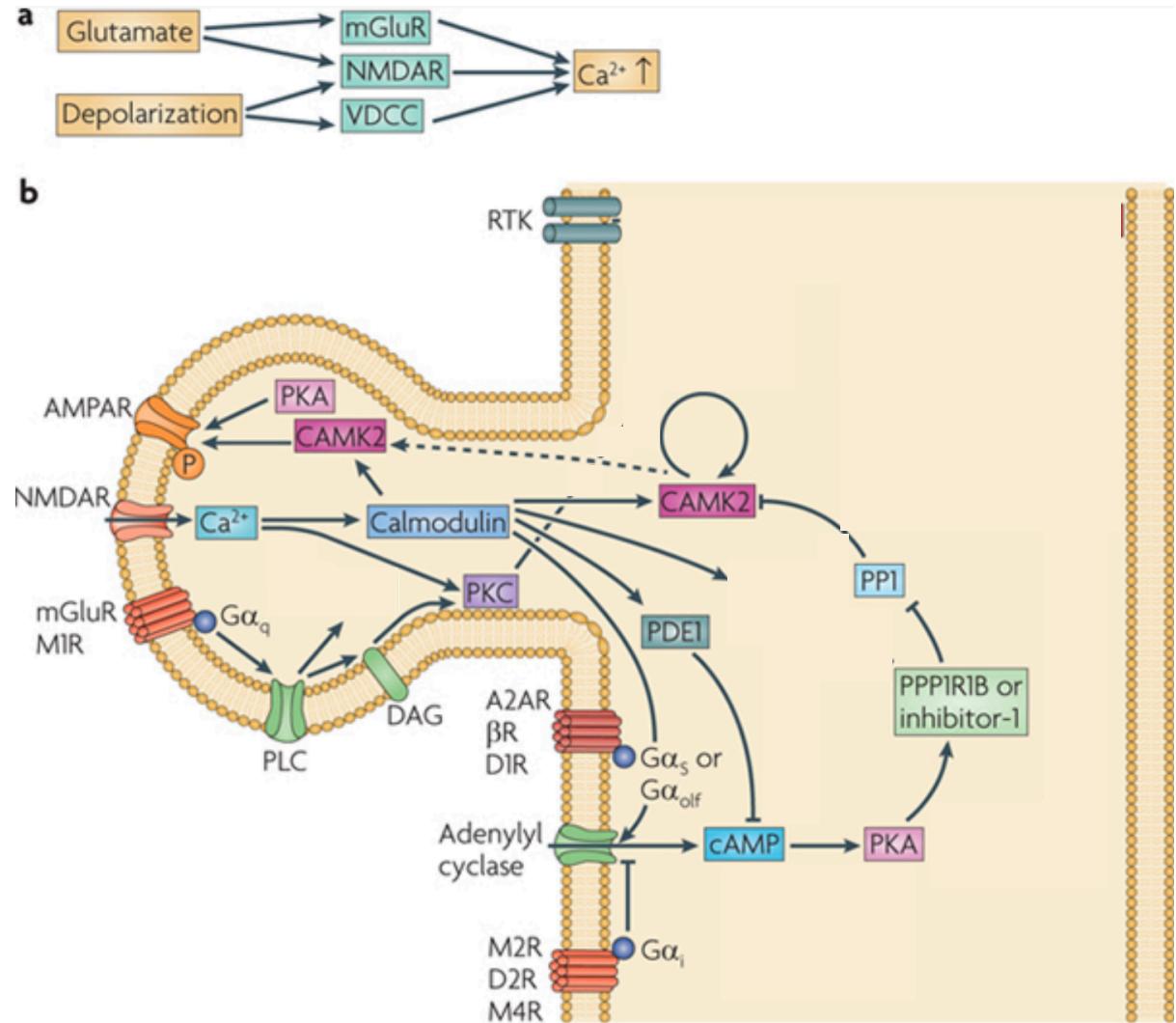
Long-term synaptic plasticity: cellular mechanisms



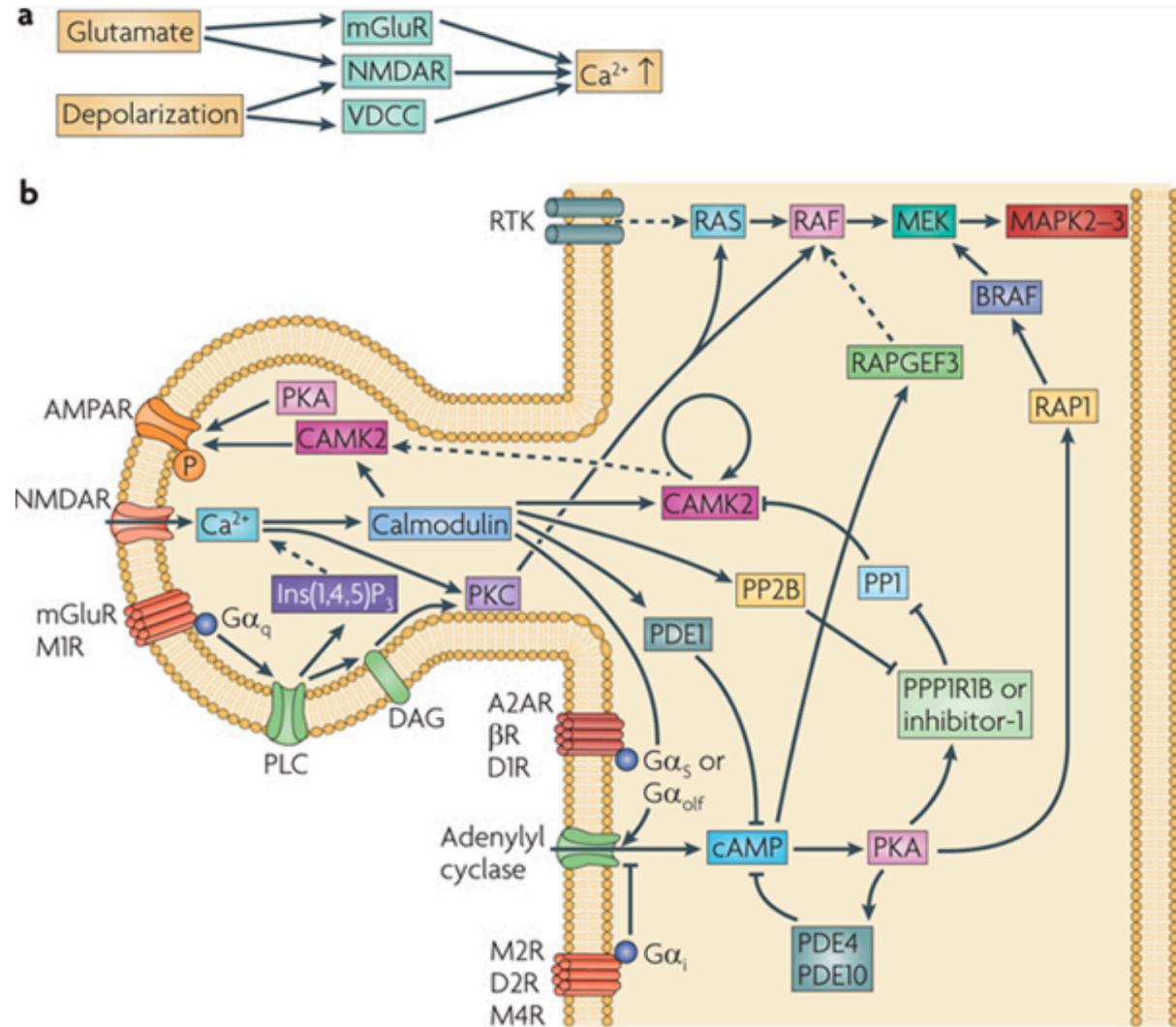
Long-term synaptic plasticity: cellular mechanisms



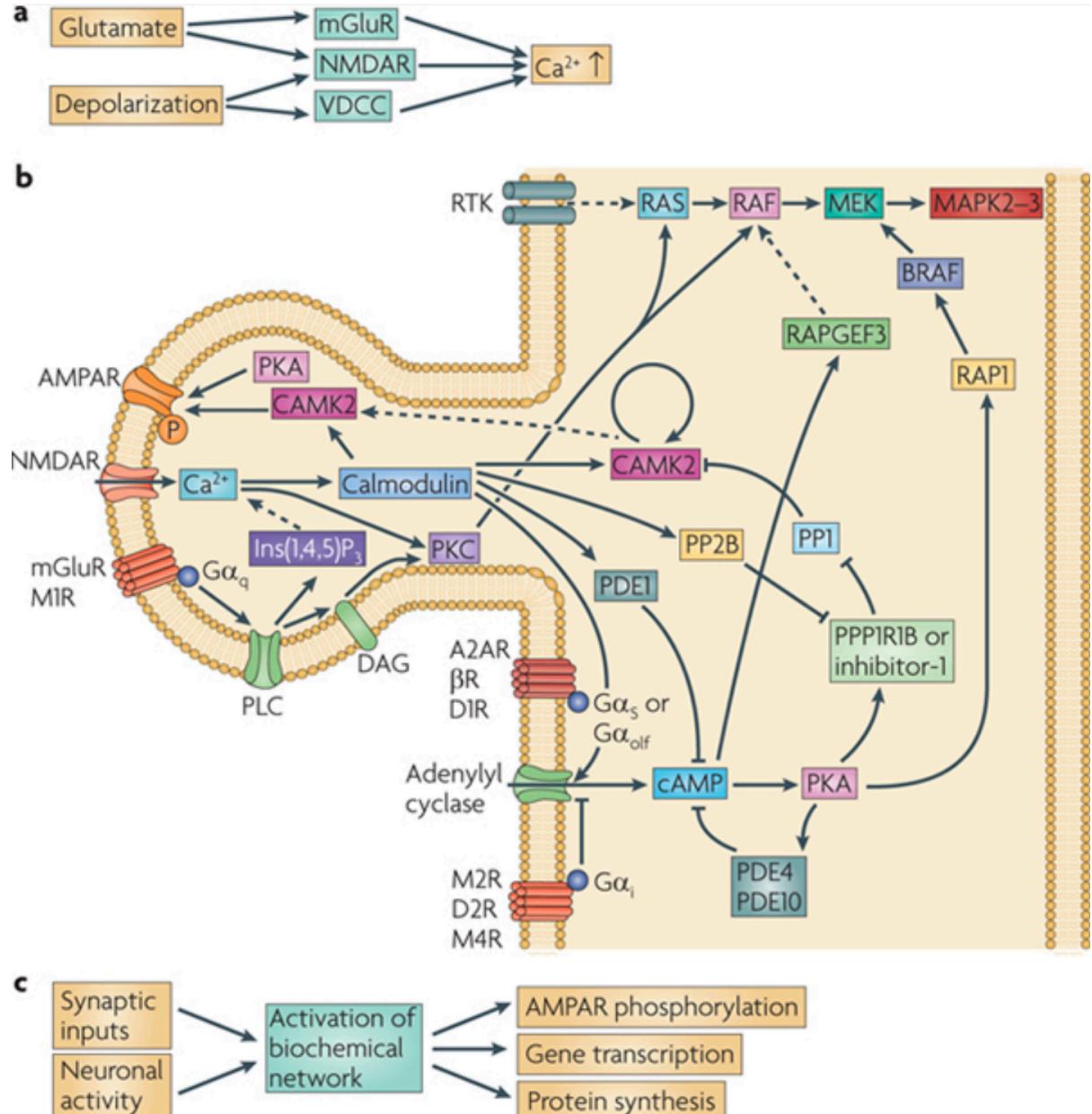
Long-term synaptic plasticity: cellular mechanisms



Long-term synaptic plasticity: cellular mechanisms



Long-term synaptic plasticity: cellular mechanisms



[Kotaleski, J. H., & Blackwell, K. T. (2010). Modelling the molecular mechanisms of synaptic plasticity using systems biology approaches. *Nature Reviews Neuroscience*, 11(4), 239-251.]

Other types of brain plasticity

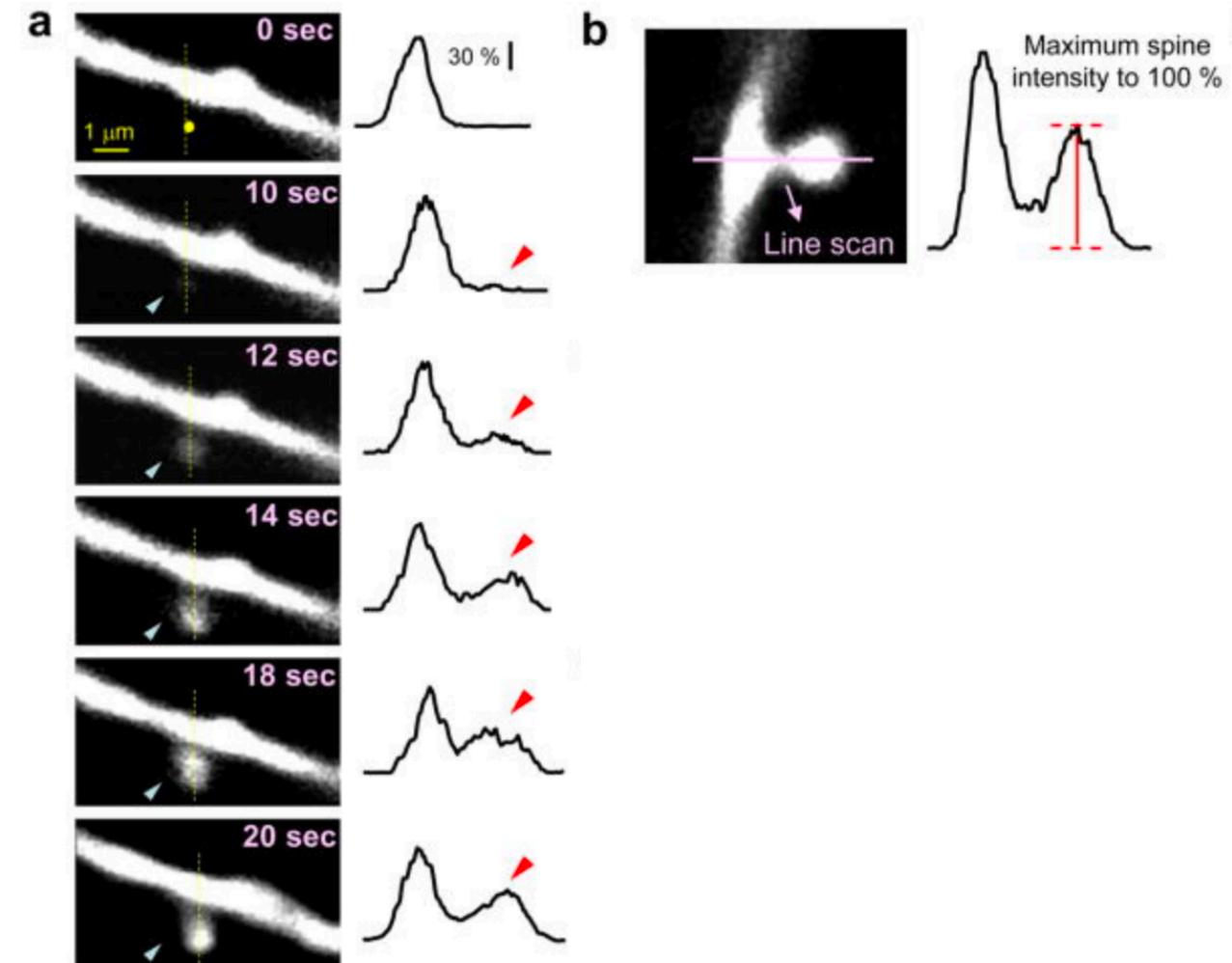
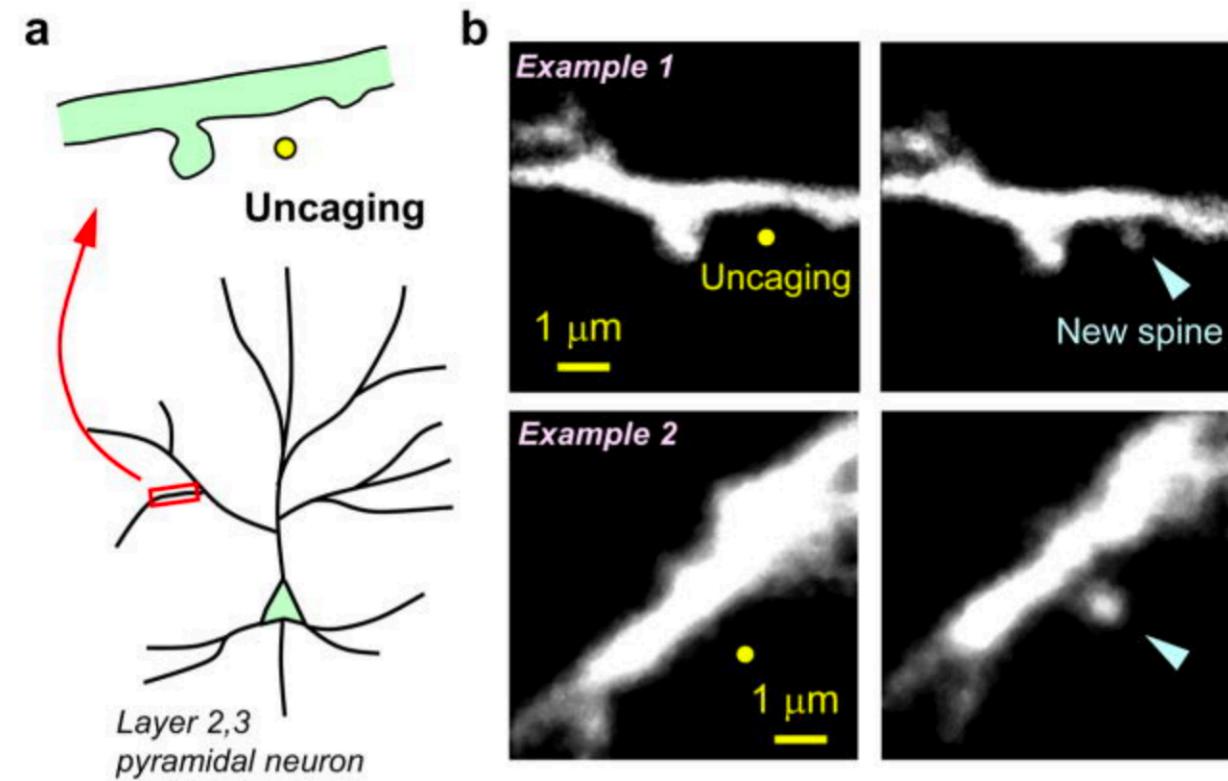
- Structural plasticity
 - Growth and retraction of neurites
 - Formation and pruning of synapses
- Adult neurogenesis
 - Formation of new neurons in dentate gyrus (in hippocampus)
- Homeostatic plasticity
 - Cellular and synaptic mechanisms that alter the excitability in case the neuron is firing too frequently or too seldom
- Trafficking of other (non-synaptic) ion channels to different locations in the neuron
- Perineuronal nets

[Kimpinski, K., Campenot, R. B., & Mearow, K. (1997). Effects of the neurotrophins nerve growth factor, neurotrophin-3, and brain-derived neurotrophic factor (BDNF) on neurite growth from adult sensory neurons in compartmented cultures. *Journal of neurobiology*, 33(4), 395-410.]

[Eriksson, P. S., Perfilieva, E., Björk-Eriksson, T., Alborn, A. M., Nordborg, C., Peterson, D. A., & Gage, F. H. (1998). Neurogenesis in the adult human hippocampus. *Nature medicine*, 4(11), 1313-1317.]

[Turrigiano, G. G., & Nelson, S. B. (2004). Homeostatic plasticity in the developing nervous system. *Nature Reviews Neuroscience*, 5(2), 97-107.]

Example: Growth of synaptic spine by glutamate exposure



[Kwon, H. B., & Sabatini, B. L. (2011). Glutamate induces de novo growth of functional spines in developing cortex. *Nature*, 474(7349), 100-104.]

[Richards, D. A., Mateos, J. M., Hugel, S., de Paola, V., Caroni, P., Gähwiler, B. H., & McKinney, R. A. (2005). Glutamate induces the rapid formation of spine head protrusions in hippocampal slice cultures. *Proceedings of the National Academy of Sciences*, 102(17), 6166-6171.]

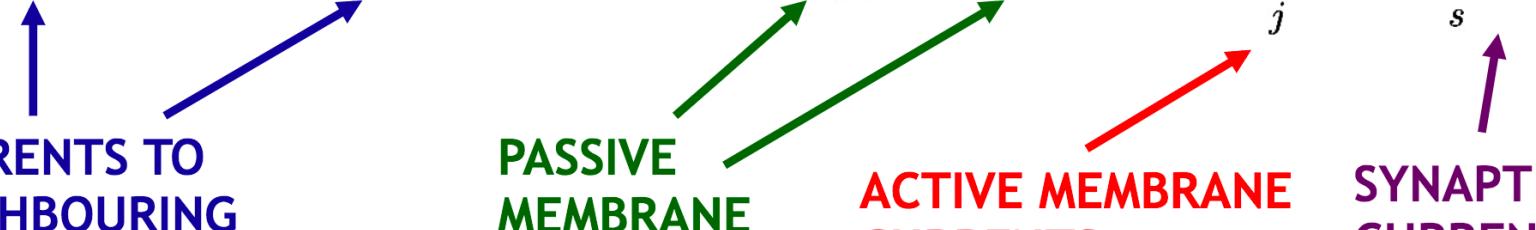
Part II: Fundamentals of multicompartmental neuron modeling

Modeling the membrane potential dynamics

- Current balance equation

$$g_{i,i+1}(V_{i+1} - V_i) - g_{i-1,i}(V_i - V_{i-1}) = c_i \frac{dV_i}{dt} + g_i^m(V_i - V_r) + \sum_j I_i^j + \sum_s I_i^s$$

CURRENTS TO NEIGHBOURING SEGMENTS **PASSIVE MEMBRANE CURRENT** **ACTIVE MEMBRANE CURRENTS** **SYNAPTIC CURRENTS**



Modeling the membrane potential dynamics

- Current balance equation

$$g_{i,i+1}(V_{i+1} - V_i) - g_{i-1,i}(V_i - V_{i-1}) = c_i \frac{dV_i}{dt} + g_i^m(V_i - V_r) + \sum_j I_i^j + \sum_s I_i^s$$

CURRENTS TO NEIGHBOURING SEGMENTS

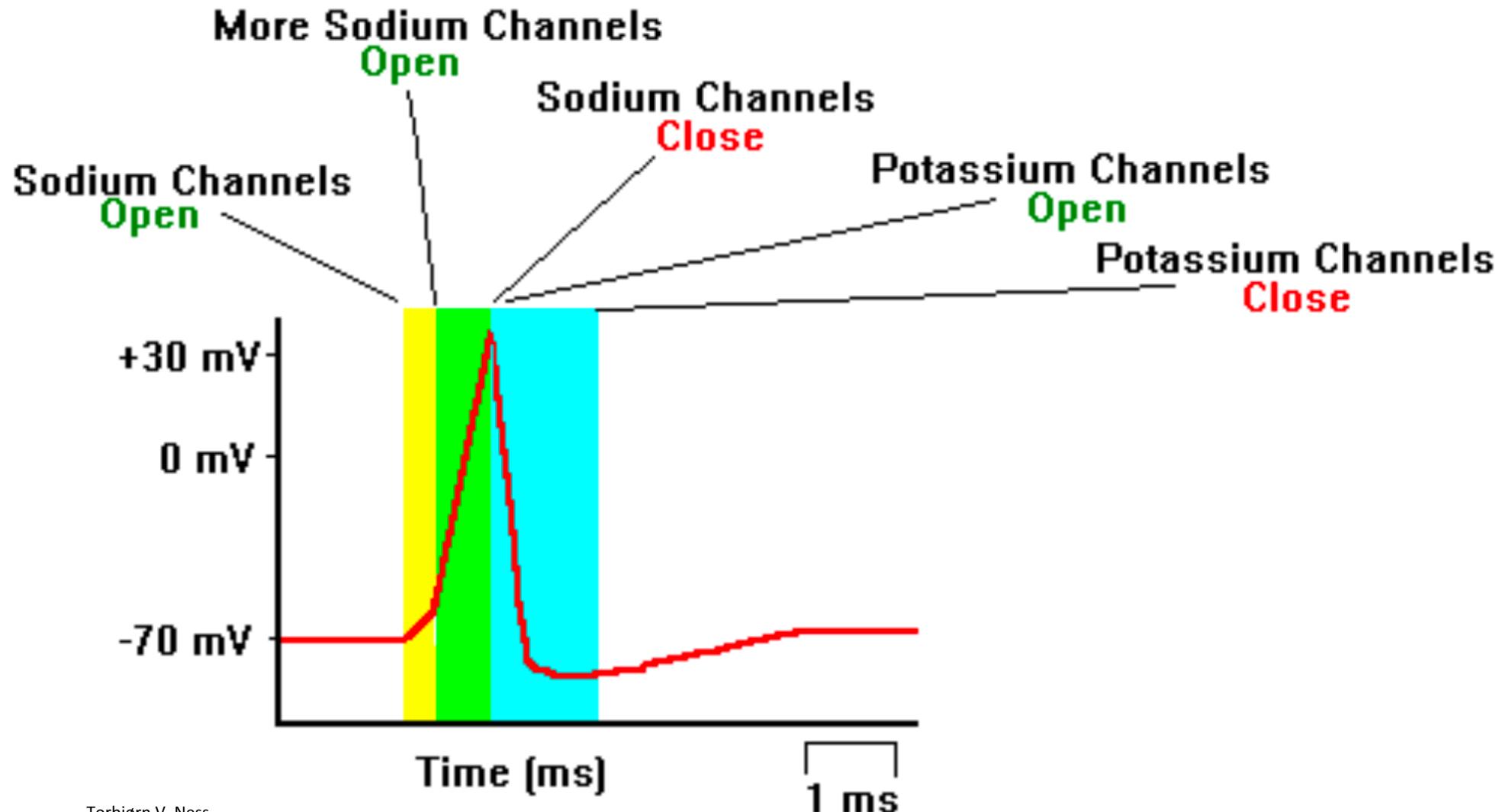
PASSIVE MEMBRANE CURRENT

ACTIVE MEMBRANE CURRENTS

SYNAPTIC CURRENTS

The diagram illustrates the components of the membrane potential dynamics equation. It shows a central term $c_i \frac{dV_i}{dt}$ with a green arrow pointing towards it from the left. Above it is a term $g_i^m(V_i - V_r)$ with a red arrow pointing towards it from the right. To the left of the central term is a term $g_{i,i+1}(V_{i+1} - V_i)$ with a blue arrow pointing upwards and to the right. To the right of the central term is a term $\sum_j I_i^j$ with a red arrow pointing downwards and to the left. To the far right is a term $\sum_s I_i^s$ with a purple arrow pointing upwards. A red square box encloses the terms $\sum_j I_i^j$ and $\sum_s I_i^s$.

Neuronal action potential



Modeling the membrane potential dynamics

- Current balance equation

$$g_{i,i+1}(V_{i+1} - V_i) - g_{i-1,i}(V_i - V_{i-1}) = c_i \frac{dV_i}{dt} + g_i^m(V_i - V_r) + \sum_j I_i^j + \sum_s I_i^s$$

CURRENTS TO NEIGHBOURING SEGMENTS **PASSIVE MEMBRANE CURRENT** **ACTIVE MEMBRANE CURRENTS** **SYNAPTIC CURRENTS**

The diagram illustrates the components of the membrane potential dynamics equation. It shows four types of currents represented by arrows pointing towards the central term: 1) 'CURRENTS TO NEIGHBOURING SEGMENTS' (blue arrow), 2) 'PASSIVE MEMBRANE CURRENT' (green arrow), 3) 'ACTIVE MEMBRANE CURRENTS' (red arrow), and 4) 'SYNAPTIC CURRENTS' (purple arrow). A red box highlights the sum of synaptic currents.

Modeling synaptic currents

$$I_{\text{syn}}(t) = g_{\text{syn}}(t) (u(t) - E_{\text{syn}})$$

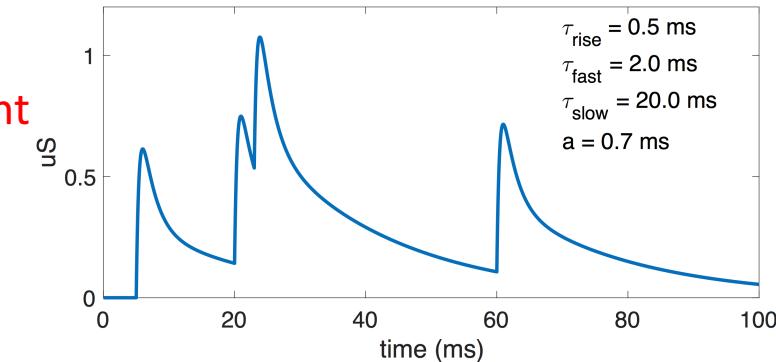
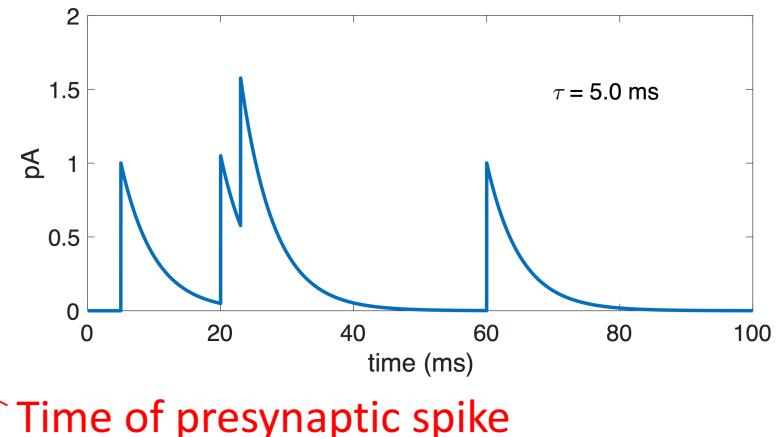
Synaptic current
 Synaptic conductance
 Membrane potential
 Reversal potential

$$g_{\text{syn}}(t) = \sum_f \bar{g}_{\text{syn}} e^{-(t-t^{(f)})/\tau} \Theta(t - t^{(f)})$$

Maximal conductance
 Decay constant
 Heaviside function
 Time of presynaptic spike

$$g_{\text{syn}}(t) = \sum_f \bar{g}_{\text{syn}} \left[1 - e^{-(t-t^{(f)})/\tau_{\text{rise}}} \right] \left[a e^{-(t-t^{(f)})/\tau_{\text{fast}}} + (1-a) e^{-(t-t^{(f)})/\tau_{\text{slow}}} \right] \Theta(t - t^{(f)})$$

Rise time constant
 Fast decay constant
 Slow decay constant
 Proportion of fast current



Modeling the membrane potential dynamics

- Current balance equation

$$g_{i,i+1}(V_{i+1}-V_i) - g_{i-1,i}(V_i-V_{i-1}) = c_i \frac{dV_i}{dt} + g_i^m(V_i-V_r) + \sum_j I_i^j + \sum_s I_i^s$$

CURRENTS TO NEIGHBOURING SEGMENTS

PASSIVE MEMBRANE CURRENT

ACTIVE MEMBRANE CURRENTS

SYNAPTIC CURRENTS

$$\frac{1}{2\pi a} \frac{\partial}{\partial x} \left(\frac{\pi a^2}{R_a} \frac{\partial V}{\partial x} \right) = C_m \frac{\partial V}{\partial t} + I_{\text{HH}}$$

[Hines (1984): Efficient computation of branched nerve equations. Int. J. Bio-Medical Computing 15: 69-76]

Solution method in the NEURON simulator

- Default integration method: Backward Euler
- Hines (1984): Efficient computation of branched nerve equations. Int. J. Bio-Medical Computing 15: 69-76

$$\frac{dy}{dt} = f(t, y)$$

$$y_{k+1} = y_k + h f(t_{k+1}, y_{k+1})$$

- Tree-like dependence of solved parameters from each other
- Efficient Gaussian elimination

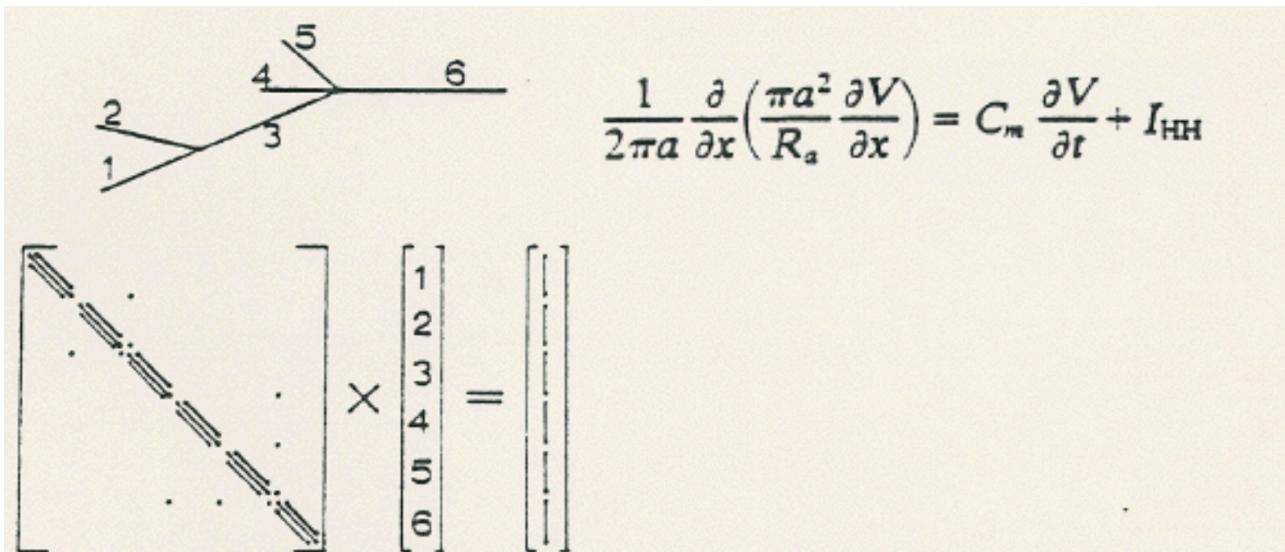


Fig. 1. An example of a branched cable and the form of its finite difference matrix equation. The position of the branches in the matrix equation is reflected in the membrane voltage vector, V . Most of the equations for a branch form a tridiagonal system but connected branches can generate non-zero far off diagonal elements.

Part III: Modeling with NEURON software: Examples

NEURON: What is it?

- A simulation environment for neuron modelling
- Based on an interpreted hoc language
- Efficient for simulating the cable equation in branching geometries
- Membrane mechanisms described using NMODL language, which allows the extension of
 - Stimulating currents
 - Ion channels
 - Synapses
- Supports use of reconstructed neuron morphologies

HOC language: Short tutorial

- The binary *nrniv* opens the hoc interpreter
- Things to do before running the simulation
 - Define the neuron morphology / morphologies and create the neuron(s)
 - Insert the membrane mechanisms and define the model parameters for each section
 - Define the numbers of segments, the time step size and the simulation end time
 - Introduce the stimuli
 - Define which model variables will be recorded
 - Initialize the variables
- Things to do after running the simulation
 - Save the recorded data into file

HOC language: Short tutorial

```
load_file("stdlib.hoc")
load_file("stdrun.hoc")

objref cvode
cvode = new CCode()
cvode.active(0)

create soma, dend[3]

connect dend[0](0), soma(1)
connect dend[1](0), soma(1)
connect dend[2](0), dend[0](1)

access soma

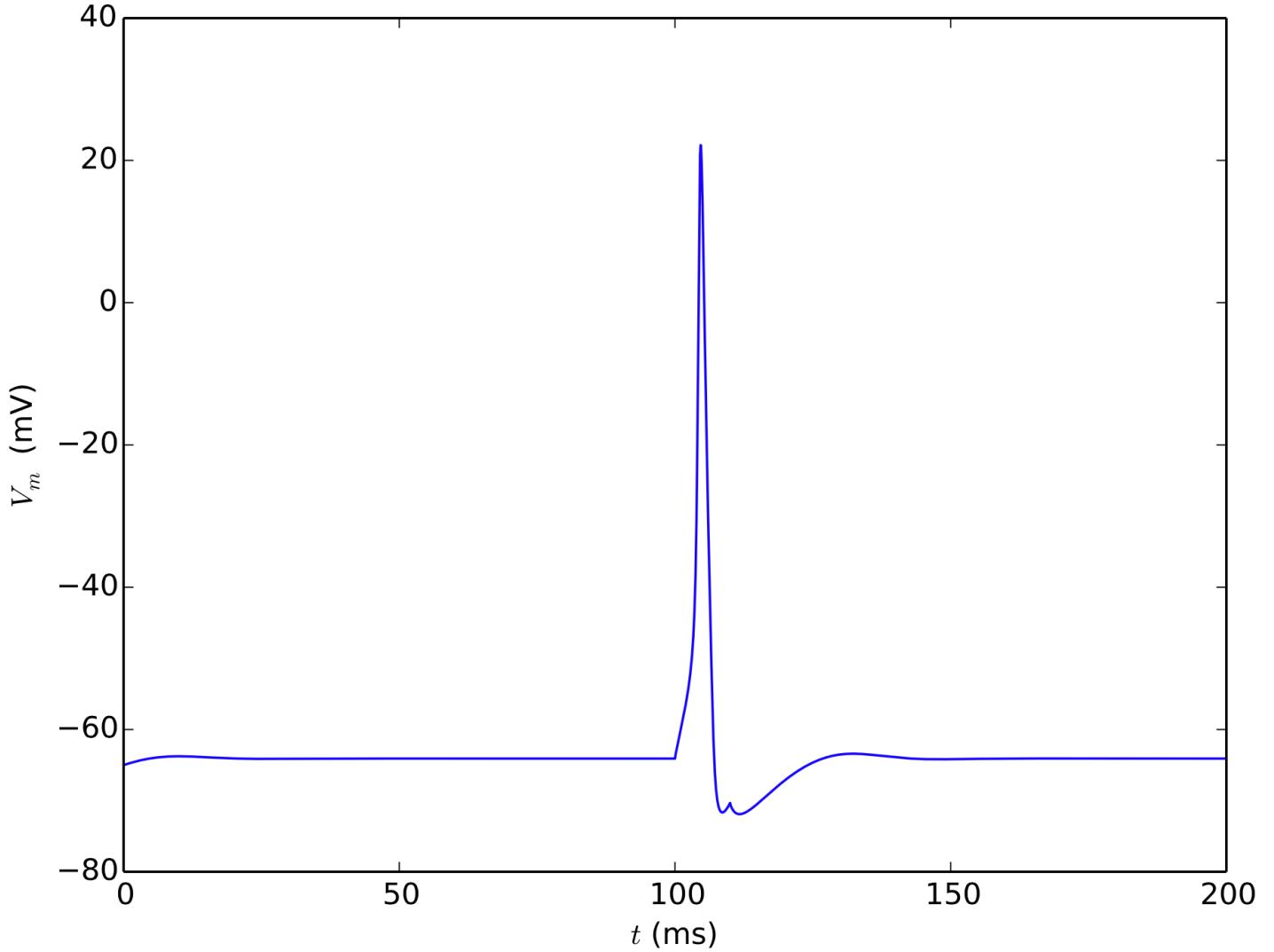
dend[0] insert pas
dend[1] insert pas
dend[2] insert pas
soma insert hh
soma {Ra = 100 cm = 1 diam = 10 L = 20}
dend[0] {Ra = 100 cm = 1 diam = 5 L = 200 e_pas = -54.3 g_pas = 0.00003}
dend[1] {Ra = 100 cm = 1 diam = 2 L = 100 e_pas = -54.3 g_pas = 0.00003}
dend[2] {Ra = 100 cm = 1 diam = 2 L = 160 e_pas = -54.3 g_pas = 0.00003}
forall nseg = 5dt = 0.025
tstop = 200

objref stim, Vrec

soma stim = new IClamp(0.5)
stim.del = 100
stim.dur = 10
stim.amp = 0.075

Vrec = new Vector()
Vrec.record(&v(0.5), dt)

init()
run()
objref myFile
myFile = new File()
myFile.wopen("run.dat")
for i=0,tstop/dt-1 {
    myFile.printf("%g %g\n", dt*i, Vrec.x(i))
}
myFile.close()
```



HOC language: Short tutorial

```
load_file("stdlib.hoc")
load_file("stdrun.hoc")
```

Load standard hoc functions that may be useful

```
objref cvode
cvode = new CVode()
cvode.active(0)
```

Define an object cvode

Initialize a variable time step integrator

But for now, use fixed time step method

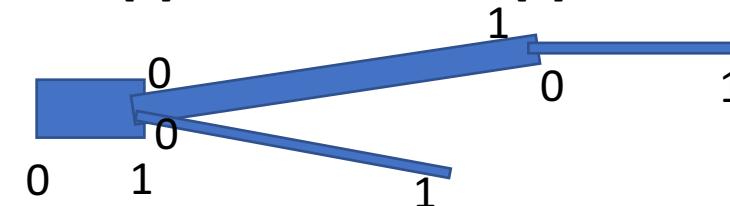
```
create soma, dend[3]
```

Create four segments called soma, dend[0], dend[1] and dend[2]

```
connect dend[0](0), soma(1)
connect dend[1](0), soma(1)
connect dend[2](0), dend[0](1)
```

Connect 0-ends of dend[0] and dend[1] to the 1-end of soma,
and 0-end of dend[2] to 1-end of dend[0]

```
access soma
```



Set soma as the “reference” section

HOC language: Short tutorial

```
dend[0] insert pas  
dend[1] insert pas  
dend[2] insert pas ]  
soma insert hh  
soma {Ra = 100 cm = 1 diam = 10 L = 20}  
dend[0] {Ra = 100 cm = 1 diam = 5 L = 100 e_pas = -54.3 g_pas = 0.00003}  
dend[1] {Ra = 100 cm = 1 diam = 2 L = 50 e_pas = -54.3 g_pas = 0.00003}  
dend[2] {Ra = 100 cm = 1 diam = 2 L = 80 e_pas = -54.3 g_pas = 0.00003}  
forall nseg = 5  
dt = 0.025  
tstop = 200  
  
objref stim, Vrec  
  
soma stim = new IClamp(0.5)  
stim.del = 100  
stim.dur = 10  
stim.amp = 0.12  
  
Vrec = new Vector()  
Vrec.record(&v(0.5),dt)
```

Insert passive membrane mechanism to all sections

Insert Hodgkin-Huxley currents (including leak) to soma section

Define the lengths and thicknesses and passive membrane parameters of each section

Define the number of compartments in each section

Define the time step size (only when cvode.active(0)), in milliseconds

Define the simulation end time in milliseconds

Define two objects, one for stimulus and one for recordings

Initialize object stim to be a current clamp stimulus

Set the delay (onset time) and duration of the current clamp in milliseconds and the amplitude in nanoamps

Initialize object Vrec to be a vector

Set Vrec to record membrane potential at the middle of soma

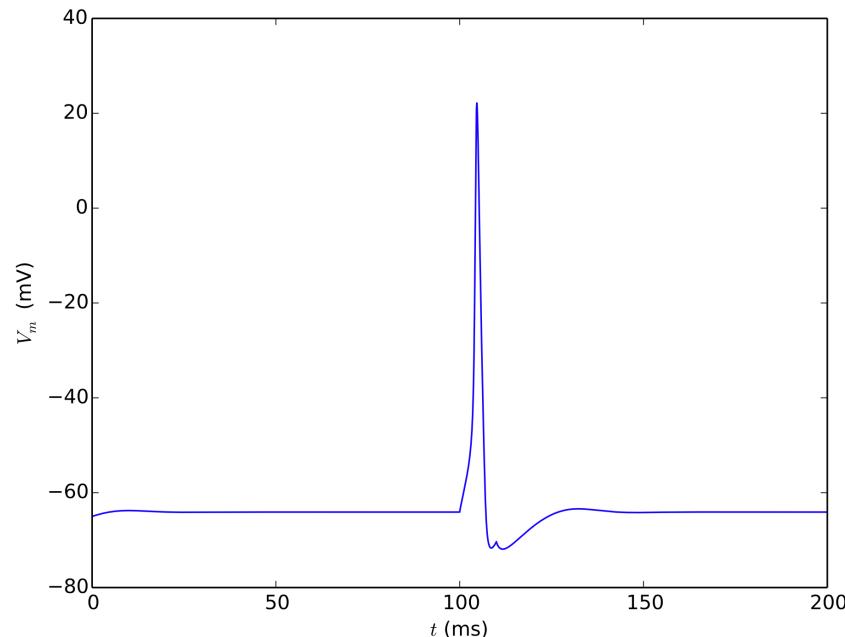
HOC language: Short tutorial

```
init() ←  
run() ←  
  
objref myFile  
myFile = new File()  
myFile.wopen("run.dat")  
for i=0,tstop/dt-1 {  
    myFile.printf("%g %g\n", dt*i, Vrec.x(i))  
}  
myFile.close()
```

Initialize all state variables to their initial values

Run the simulation

Save the simulation output (time, V) in a file “run.dat”



HOC language: Short tutorial

- Useful commands
 - psection(): show all mechanisms of the selected section
 - topology(): show all compartments and how they are connected

```
bash-3.2$ nrniv
NEURON -- VERSION 7.5 (1515;79f6026b82d7) 2017-02-09
Duke, Yale, and the BlueBrain Project -- Copyright 1984-2016
See http://neuron.yale.edu/neuron/credits

oc>load_file("myneuron.hoc")
    1
    1
    0
    1
    1
    0
    1
oc>soma psection()
soma { nseg=5 L=20 Ra=100
    /*location 0 attached to cell 0*/
    /* First segment only */
    insert hh { gbar_hh=0.12 gkbar_hh=0.036 gL_hh=0.0003 el_hh=-54.3}
    insert na_ion { ena=50}
    insert k_ion { ek=-77}
    insert capacitance { cm=1}
    insert morphology { diam=10}
}
oc>topology()

|-----|     soma(0-1)
`----|     dend[0](0-1)
      `---|     dend[2](0-1)
      `---|     dend[1](0-1)

1
oc>|
```

HOC language with Python interface

```
from neuron import h  
h('<hoc command>')  
  
Vrec = numpy.array(h.Vrec)
```

If NEURON is installed with a Python interface, the hoc interpreter can be loaded in a Python script

Any command in hoc language can then be run through Python

There exist built-in commands in Python for the hoc interpreter, e.g., for loading a hoc vector as a numpy array

NMODL language: Very short tutorial

- Mechanism files (.mod) can be compiled with the binary *nrnivmodl*
- NMODL code distributed to blocks:
 - NEURON: define the interface with NEURON
 - PARAMETER: define the model parameters
 - STATE: define the solved state variables
 - ASSIGNED: define other variables that will be handled by the mechanism
 - DERIVATIVE: define the differential equation for the state variables
 - BREAKPOINT: calculate the current terms based on the state variables
 - INITIAL: initialize the state variables

NMODL example: I_h channel

- <https://senselab.med.yale.edu/modeldb/showModel.cshtml?model=139653&file=/L5bPCmodelsEH/mod/Ih.mod>

$$I = \bar{g}m^{N_m}h^{N_h}(E - V) \quad \frac{dm}{dt} = \frac{m - m_\infty}{\tau_m} \quad \text{and} \quad \frac{dh}{dt} = \frac{h - h_\infty}{\tau_h}$$

$$\alpha_m = -\frac{1}{\tau_{ma}} \cdot \frac{V_{offma} - V}{\exp(-\frac{V_{offma} - V}{V_{sloma}}) - 1}$$

$$\beta_m = \frac{1}{\tau_{mb}} \exp\left(-\frac{V_{offmb} - V}{V_{slomb}}\right)$$

$$m_\infty = \frac{\alpha_m}{\alpha_m + \beta_m}$$

$$\tau_m = \frac{1}{\alpha_m + \beta_m}$$

```

NEURON {
  SUFFIX Ih
  NONSPECIFIC_CURRENT ihcn
  RANGE gIhbar, gIh, ihcn
}

UNITS {
  (S) = (siemens)
  (mV) = (millivolt)
  (mA) = (milliamp)
}

PARAMETER {
  gIhbar = 0.00001 (S/cm2)
  ehcn = -45.0 (mV)
}

ASSIGNED {
  v      (mV)
  ihcn  (mA/cm2)
  gIh    (S/cm2)
  mInf
  mTau
  mAlpha
  mBeta
}

STATE {
  m
}

BREAKPOINT {
  SOLVE states METHOD cnexp
  gIh = gIhbar*m
  ihcn = gIh*(v-ehcn)
}

DERIVATIVE states {
  rates()
  m' = (mInf-m)/mTau
}

INITIAL{
  rates()
  m = mInf
}

PROCEDURE rates(){
  UNITSOFF
  if(v == -154.9){
    v = v + 0.0001
  }
  mAlpha = 0.001*6.43*(v+154.9)/(exp((v+154.9)/11.9)-1)
  mBeta = 0.001*193*exp(v/33.1)
  mInf = mAlpha/(mAlpha + mBeta)
  mTau = 1/(mAlpha + mBeta)
}
UNITSON
}

```

NMODL example: I_h channel (Hay et al. 2011)

```
NEURON {
    SUFFIX Ih
    NONSPECIFIC_CURRENT ihcn
    RANGE gIhbar, gIh, ihcn
}

UNITS {
    (S) = (siemens)
    (mV) = (millivolt)
    (mA) = (milliamp)
}

PARAMETER {
    gIhbar = 0.00001 (S/cm2)
    ehcn = -45.0 (mV)
}

ASSIGNED
    v      (mV)
    ihcn  (mA/cm2)
    gIh   (S/cm2)
    mInf
    mTau
    mAlpha
    mBeta
}
```

Name of the mechanism

Name of the current

Variables that may be recorded in the hoc code

Units used

Model parameters and their default values

Variables that are in dependence with the state variables of the STATE block

NMODL example: I_h channel (Hay et al. 2011)

```
STATE {  
    m  
}  
  
BREAKPOINT {  
    SOLVE states METHOD cnexp  
    gIh = gIhbar*m  
    ihcn = gIh*(v-ehcn)  
}  
  
DERIVATIVE states {  
    rates()  
    m' = (mInf-m)/mTau  
}  
  
INITIAL{  
    rates()  
    m = mInf  
}  
  
PROCEDURE rates(){  
    UNITSOFF  
    if(v == -154.9){  
        v = v + 0.0001  
    }  
    mAlpha = 0.001*6.43*(v+154.9)/(exp((v+154.9)/11.9)-1)  
    mBeta = 0.001*193*exp(v/33.1)  
    mInf = mAlpha/(mAlpha + mBeta)  
    mTau = 1/(mAlpha + mBeta)  
    UNITSON  
}
```

The state variable

In the beginning of the BREAKPOINT block the program is told to solve the update the state variables using a solution method for linear ODEs

Update the RANGE variable g_{Ih}

Update the current term $ihcn$

Calculate m_{Inf} and m_{Tau} based on membrane potential v

Define the differential equation for m (linear w.r.t. m)

Initialize the variable m at a value which is stable if the membrane potential value is stable

Prevent an operation of dividing zero by zero

HH equations for the functions $m_{\text{Inf}}(v)$ and $m_{\text{Tau}}(v)$