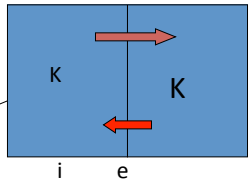
What are the flows (j's) if V_{rest}=-80mV

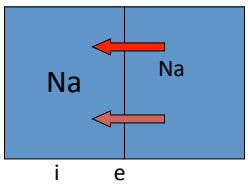


E_K=-104mV

net outflux

$$j_{diff} > j_{Efield}$$

net influx



E_{Na}=+85.3mV

$$j_{diff} \;\&\; j_{Efield}$$

point inward

Neither Nernst Potential is in equilibrium with the rest potential.

We can have another kind of equilibrium

$$\sum_p j_p = 0$$

The Goldman Equation was developed to consider this type of equilibrium. It is based on the Nernst Planck Equation, but under certain conditions.

Let's look at the total current

$$J = J_K + J_{Na} + J_{Cl}$$

From Goldman eqn we can write the following

$$J = \frac{V_m F^2 P_K}{RT}\left(\frac{w - ye^{V_m F/RT}}{1 - e^{V_m F/RT}}\right)$$

where

$$w = \left[K\right]_e + \frac{P_{Na}}{P_K}\left[Na\right]_e + \frac{P_{Cl}}{P_K}\left[Cl\right]_i$$

$$y = \left[K\right]_i + \frac{P_{Na}}{P_K}\left[Na\right]_i + \frac{P_{Cl}}{P_K}\left[Cl\right]_e$$

At Equilibrium: net current J=0

$$J = \frac{V_m F^2 P_K}{RT}\left(\frac{w - ye^{V_m F/RT}}{1 - e^{V_m F/RT}}\right) = 0$$

or

$$w - ye^{V_m F/RT} = 0 \Rightarrow e^{V_m F/RT} = \frac{w}{y}$$

$$V_m = \frac{RT}{F}\ln\frac{w}{y}$$

Goldman Eqn for rest (equil) potential

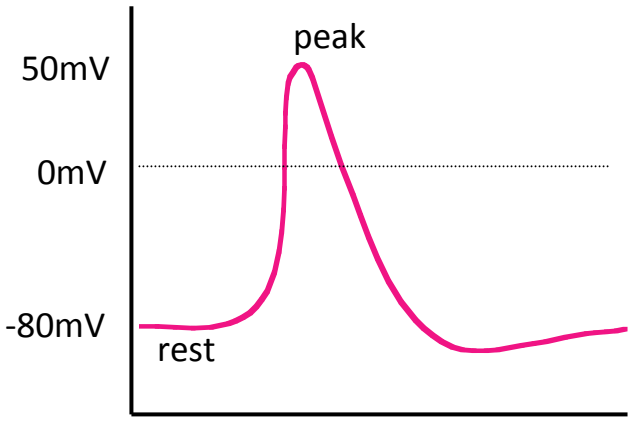
From Goldman Eqn- Equilbrium is a weighted average of all the Nernst Potentials. The weights are the permeabilities.

Under rest conditions, what permeability is greatest? Na? K?

$$P_K >> P_{Na}$$

How do we know this?

$$V_{m\_rest} \approx E_K$$



What's true about permeabilities at peak?

$$P_{Na} >> P_K$$

since

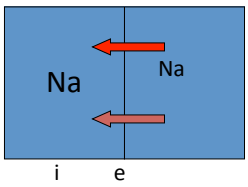
$$V_{m\_peak} \approx E_{Na}$$

Recall that at rest (V_m=-80mV)

net influx

j_{diff} & j_{Efield}

point inward



E_{Na}=+50.2mV

Everything is primed for Na to enter the cell at rest!

Let's define $\left(V_m - E_{Na}\right)$ as the driving force for Na

At rest, the driving force is negative signifying an inward current

To get the current density, $J_{Na,}$ we need multiply driving force by?

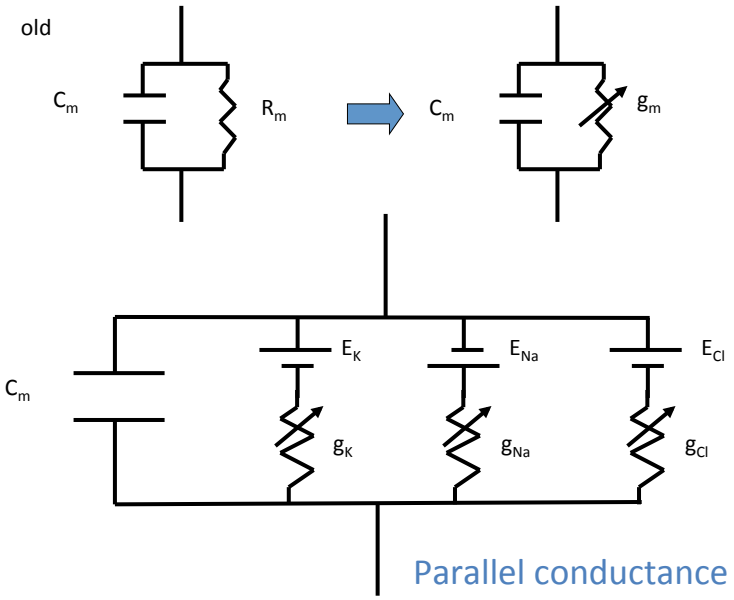
g_{Na} Conductance per unit area

Units?

g_{Na} has units S/cm^2 ,is positive, and varies during action potential

$$J_{Na} = g_{Na}\left(V_m - E_{Na}\right)$$

Improved model of excitable membrane



Parallel conductance model

Total membrane current given by

$$J_m = g_K(V_m - E_K) + g_{Na}(V_m - E_{Na}) + g_{Cl}(V_m - E_{Cl}) + C_m \frac{dV_m}{dt}$$

capacitive current

At rest, $J_m=0$, $dV_m/dt = 0$

$$V_m = \frac{g_k E_K + g_{Na} E_{Na} + g_{Cl} E_{Cl}}{g_k + g_{Na} + g_{Cl}}$$

How can the ionic conductances be determined as a function of V_m?

If V_m can be held constant, then since driving force (V_m-E_{Na}) is constant. As a result,

$$I_{Na} \propto g_{Na} \qquad g_{Na} = I_{Na} / (V_m - E_{Na})$$

Might be able to measure total membrane current. How can the individual components be separated?

First build a system to hold V_m constant.

From eqn. (6) this may be transformed into a form suitable for comparison with the experimental results, i.e.

$$g_K = \{(g_{K\infty})^{\frac{1}{4}} - [(g_{K\infty})^{\frac{1}{4}} - (g_{K0})^{\frac{1}{4}}] \exp(-t/\tau_n)\}^4, \qquad (11)$$

where $g_{K\infty}$ is the value which the conductance finally attains and g_{K0} is the conductance at $t=0$. The smooth curves in Fig. 3 were calculated from

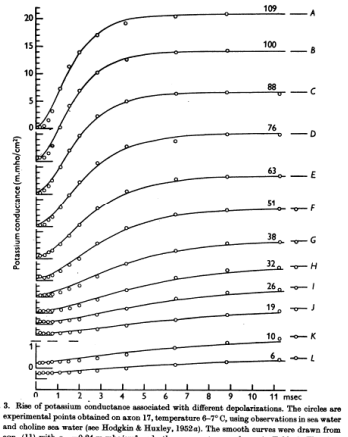
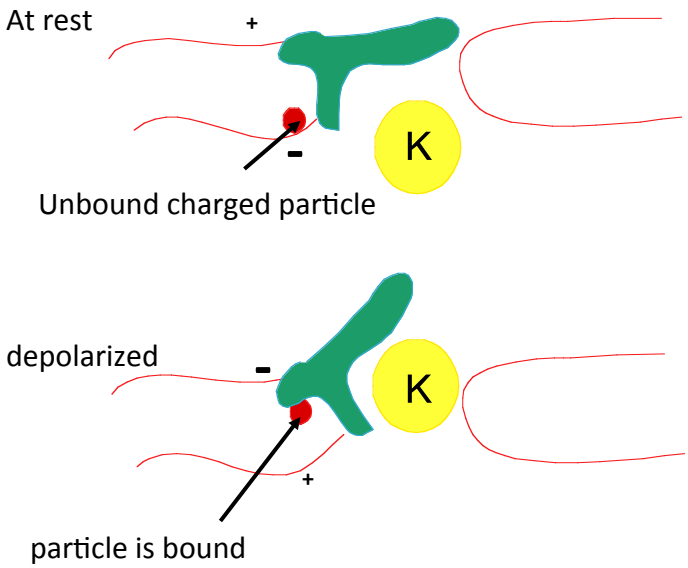


Fig. 3. Rise of potassium conductance associated with different depolarizations. The circles are experimental points obtained on axon 17, temperature 6-7° C, using observations in sea water and choline sea water (see Hodgkin & Huxley, 1952a). The smooth curves were drawn from eqn. (11) with $g_{K0}=0{\cdot}24$ m.mho/cm² and other parameters as shown in Table 1. The time scale applies to all records. The ordinate scale is the same in the upper ten curves (*A* to *J*) and is increased fourfold in the lower two curves (*K* and *L*). The number on each curve gives the depolarization in mV.

Voltage sensitive K gating: Simple Model



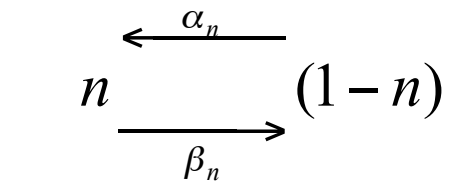
HH Gating Model

- Movement of gating particle is rate dependent
- At any given voltage, some particles are inside and some are outside. They move at some rate, depending on the membrane potential
- Binding can be explained with first order kinetics.

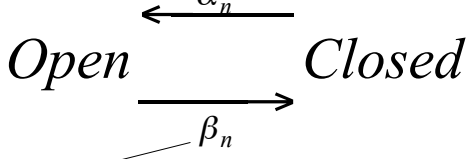
HH gating Model for K channel

- n - fraction of particles that are bound to the outside
- (1-n) - fraction of particles that are unbound on inside
- β_n - rate constant for going from outside to inside. Open to closed state. Rate cnst of closing
- α_n - rate constant for going from inside to outside. Closed to open. Rate cnst of opening

Binding Kinetics can be written as:



Rate of opening



Rate of closing

The net rate of change of the fraction of particles that become bound (net rate of opening) is

$$\frac{dn}{dt} = \alpha_n(1-n) - \beta_n n$$

Net rate of change depends on the difference of the rate from going from closed to open and the rate going from open to closed.

At any given voltage clamp, alphas and betas are constant. Equation for n is a first order ODE with constant coefficients.

The analytical solution is

$$n(t) = n_\infty - (n_\infty - n_0)e^{-t/\tau_n}$$

n_0 is the fraction of particles that are bound before voltage clamp is applied.

n_∞ is fraction of particles that are bound at steady state after clamp is applied

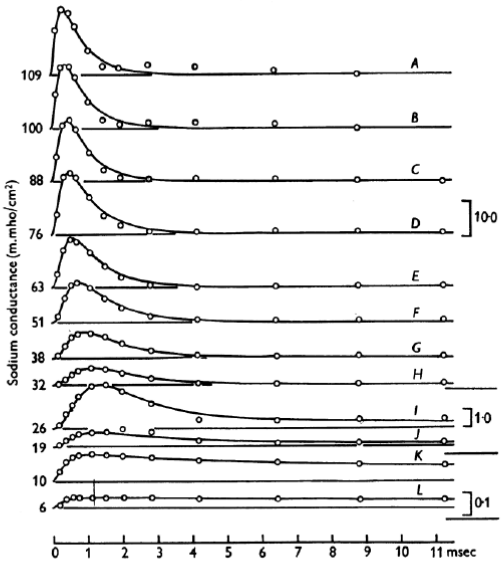
Both are obtained by setting dn/dt =0!

$$\frac{dn}{dt} = 0 = \alpha_n(1-n) - \beta_n n$$

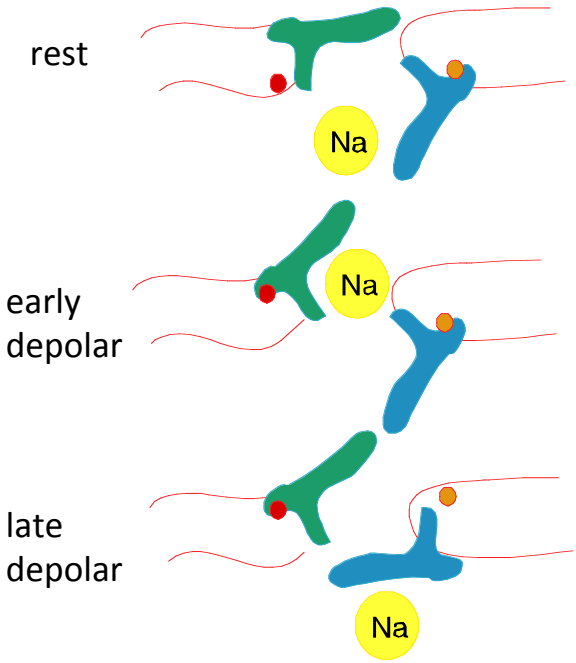
Where n was $$n_0 = \frac{\alpha_n(V_0)}{\alpha_n(V_0) + \beta_n(V_0)}$$

Where n wants to be $$n_\infty = \frac{\alpha_n(V_{Clamp})}{\alpha_n(V_{Clamp}) + \beta_n(V_{Clamp})}$$

How fast n wants to get there $$\tau_n = \frac{1}{\alpha_n(V_{Clamp}) + \beta_n(V_{Clamp})}$$



from Hodgkin & Huxley, J. Physiol. (1952)



Same binding Kinetics as used for K:

$$m \underset{\beta_m}{\overset{\alpha_m}{\rightleftarrows}} (1-m)$$

$$h \underset{\beta_h}{\overset{\alpha_h}{\rightleftarrows}} (1-h)$$

The net rate of change of the fraction of particles that become bound is

$$\frac{dm}{dt} = \alpha_m(1-m) - \beta_m m$$

$$\frac{dh}{dt} = \alpha_h(1-h) - \beta_h h$$

For activation and inactivation to work, alpha_m and alpha_h must have opposite behavior as a function of V_m

The analytical solutions are

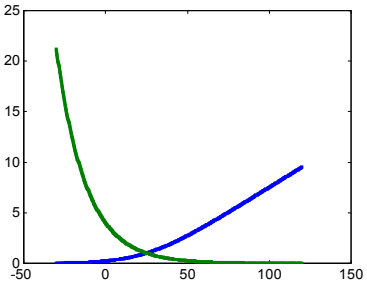
$$m(t) = m_\infty - (m_\infty - m_0)e^{-t/\tau_m}$$

$$h(t) = h_\infty - (h_\infty - h_0)e^{-t/\tau_h}$$

Parameters have analogous forms to those obtained for n.

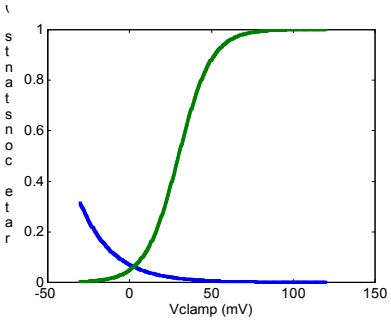
$$\alpha_m = 0.1\frac{25 - V_m}{\exp\left(\frac{25 - V_m}{10}\right) - 1}$$

$$\beta_m = 4.0\exp\left(\frac{-V_m}{18.0}\right)$$

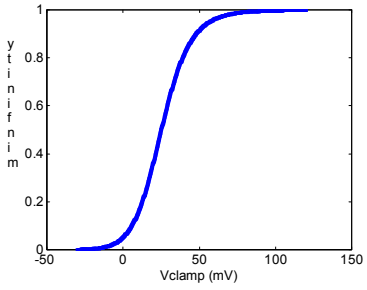


$$\alpha_h = 0.07\exp\left(\frac{-V_m}{20.0}\right)$$

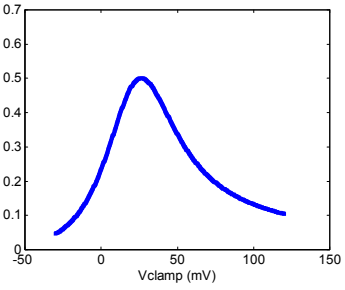
$$\beta_h = \frac{1}{\exp\left(\frac{30 - V_m}{10}\right) + 1}$$



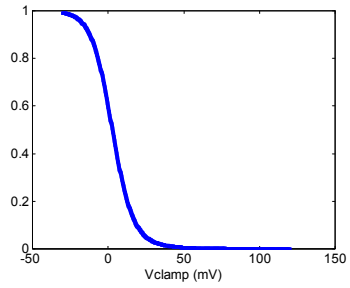
m_∞



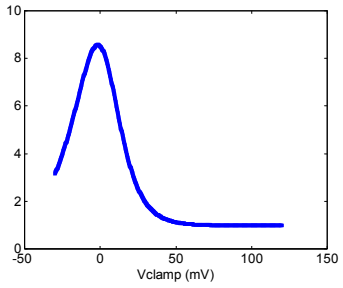
τ_m



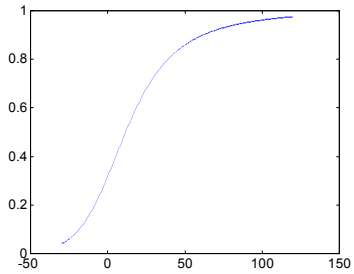
h_∞



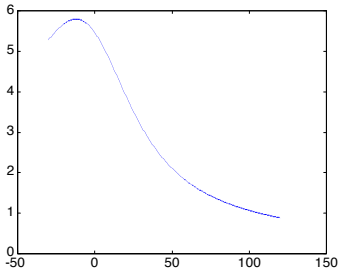
τ_h



n_∞



τ_n



Putting it all together

$$\frac{dv_m}{dt} = -\frac{1}{C_m}\left(I_{Na} + I_K + I_L\right) \qquad I_{Na} = \overline{g}_{Na} m^3 h (v_m - e_{Na})$$

$$\frac{dm}{dt} = \alpha_m (1-m) - \beta_m m \qquad I_K = \overline{g}_K n^4 (v_m - e_K)$$

$$\frac{dh}{dt} = \alpha_h (1-h) - \beta_h h \qquad I_L = \overline{g}_L (v_m - e_L)$$

$$\frac{dn}{dt} = \alpha_n (1-n) - \beta_n n$$

No analytical solution! Must solve numerically.

Alphas and Betas

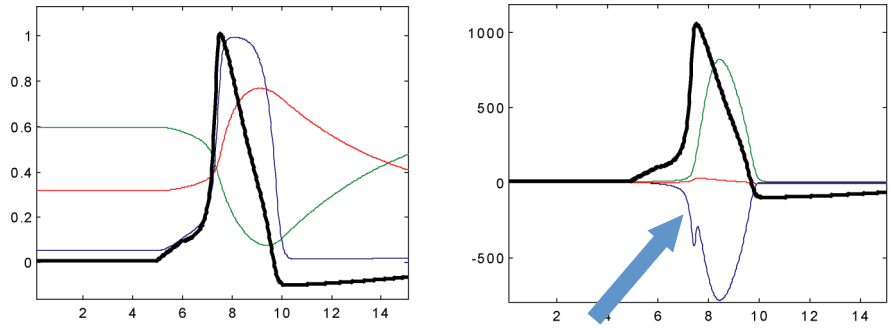
$$\alpha_m(V_m) = 0.1\frac{25 - V_m}{\exp\left(\frac{25 - V_m}{10}\right) - 1},$$
$$\beta_m(V_m) = 4\exp\left(-\frac{V_m}{18}\right);$$

$$\alpha_h(V_m) = 0.07\exp\left(\frac{-V_m}{20}\right),$$
$$\beta_h(V_m) = \frac{1}{\exp\left(\frac{30 - V_m}{10}\right) + 1};$$

$$\alpha_n(V_m) = 0.01\frac{10 - V_m}{\exp\left(\frac{10 - V_m}{10}\right) - 1},$$
$$\beta_n(V_m) = 0.125\exp\left(-\frac{V_m}{80}\right);$$

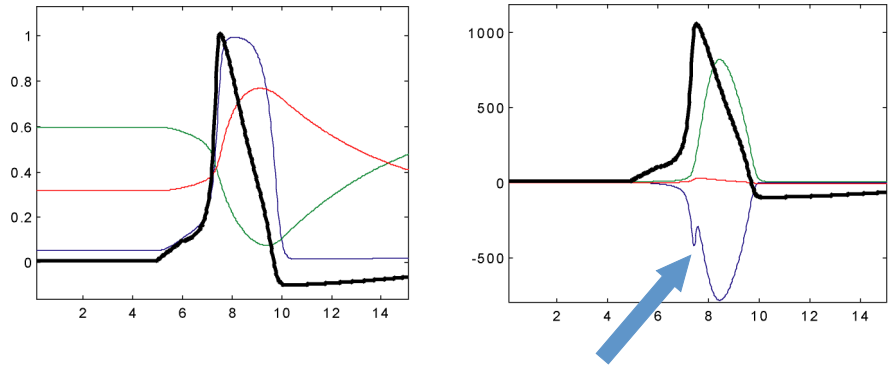
Qualitative view of an AP

- Stimulus causes membrane potential to rise.
- m increases, n increases and h decreases as V_m increases
- Number of bound m particles increases faster than number of bound n and unbound h since tau_m<tau_n and tau_h
- I_{Na} increases since more channels are open and this acts to further increase V_m



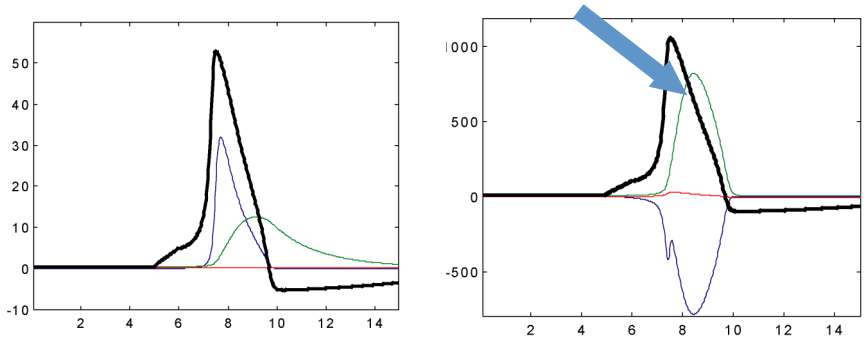
Qualitative view of an AP

- As V_m increases, the driving force (d.f.) for Na decreases. For I_{Na} to increase, the increase in g_{Na} must dominate the decrease in d.f.
- Eventually decreasing d.f. overwhelms increase in g_{Na} and I_{Na} decreases. The h particles are also tending to unbind as V_m increases but time constant is relatively slow.



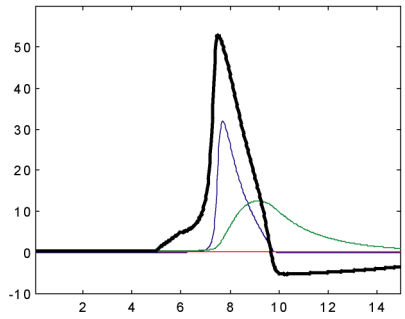
Qualitative view of an AP

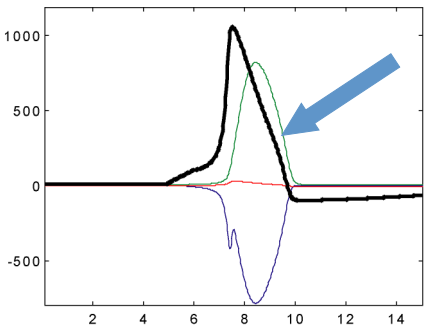
- As V_m increases driving force (d.f.) for K increases. g_K increases slowly due to n^4 and large tau_n. I_K increases acting to push V_m toward E_K.
- At peak of A.P., I_{ion}=0 since no capacitive current at peak since dV_m/dt=0.



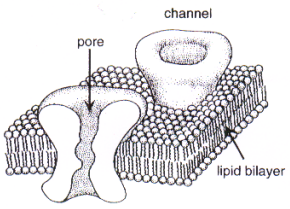
Qualitative view of an AP

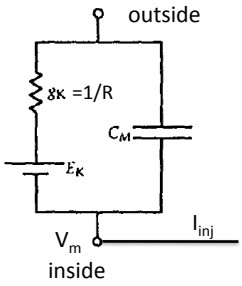
- After peak, gNa decreases due to inactivation (h unbinding). As Vm moves toward E_K, d.f. for Na increases but decrease in g_{Na} overwhelms and I_{Na} decreases.
- Increases in g_K overwhelms decrease in d.f. for K. Increase I_K dominates I_{Na} and V_m returns toward E_K and eventually equilibrates to V_{rest}.





The membrane equation



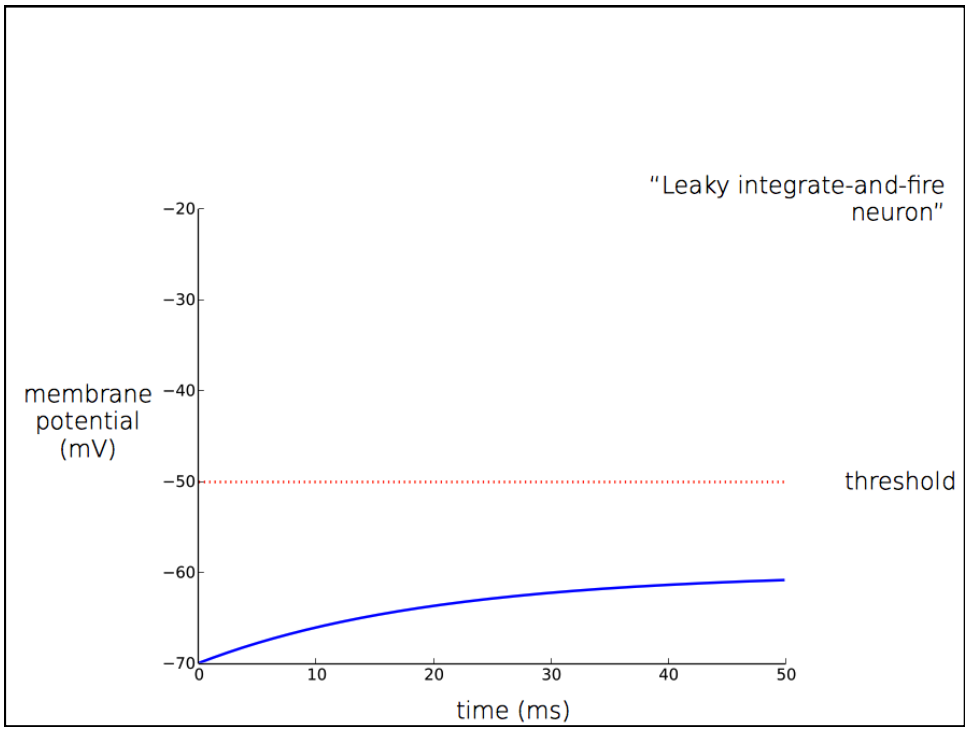


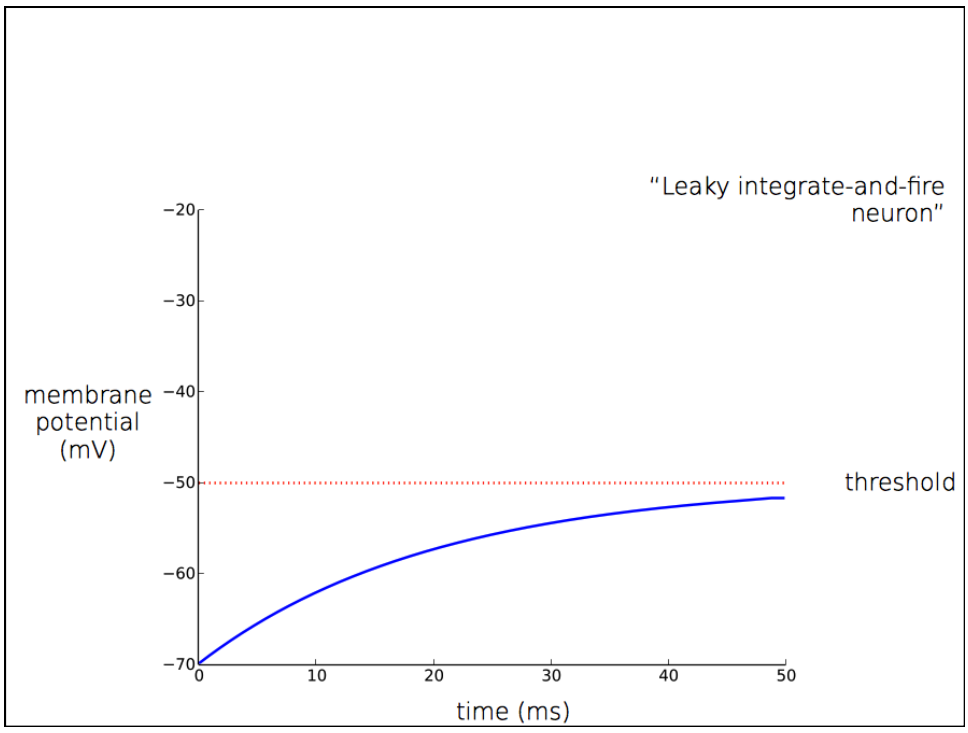
$$C\frac{dV_m}{dt} + \frac{(V_m - E_K)}{R} = I_{inj}$$

$$\tau\frac{dV_m}{dt} = E_K - V_m + RI_{inj}$$

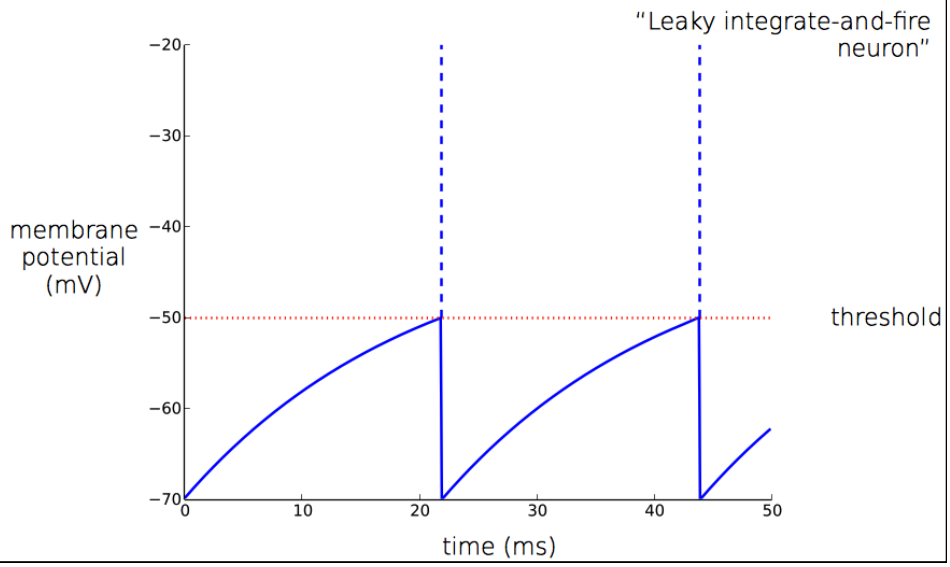
$$\tau = RC$$

membrane time constant (typically 3-10 ms)

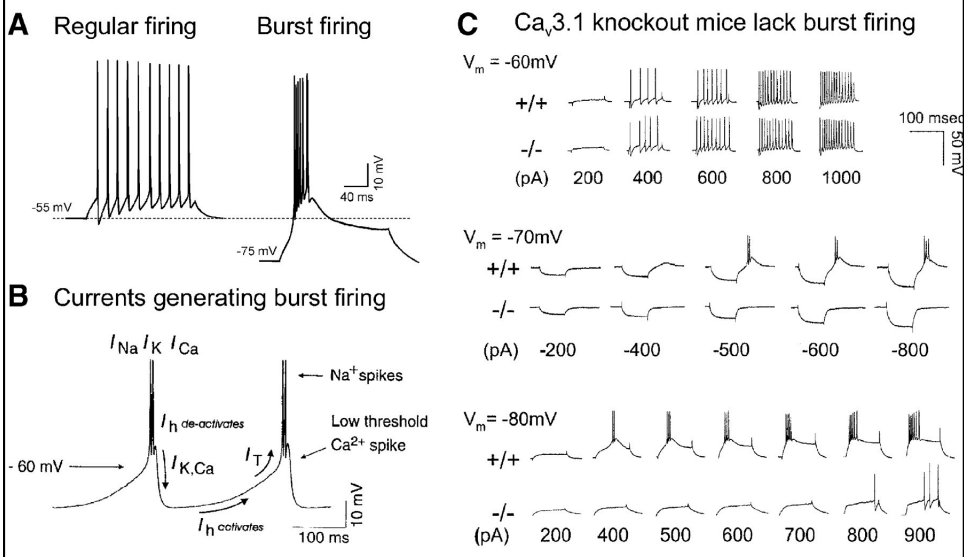

"Leaky integrate-and-fire neuron"
membrane potential (mV)
−20
−30
−40
−50
−60
−70
threshold
0
10
20
30
40
50
time (ms)


"Leaky integrate-and-fire neuron"
membrane potential (mV)
−20
−30
−40
−50
−60
−70
threshold
0
10
20
30
40
50
time (ms)

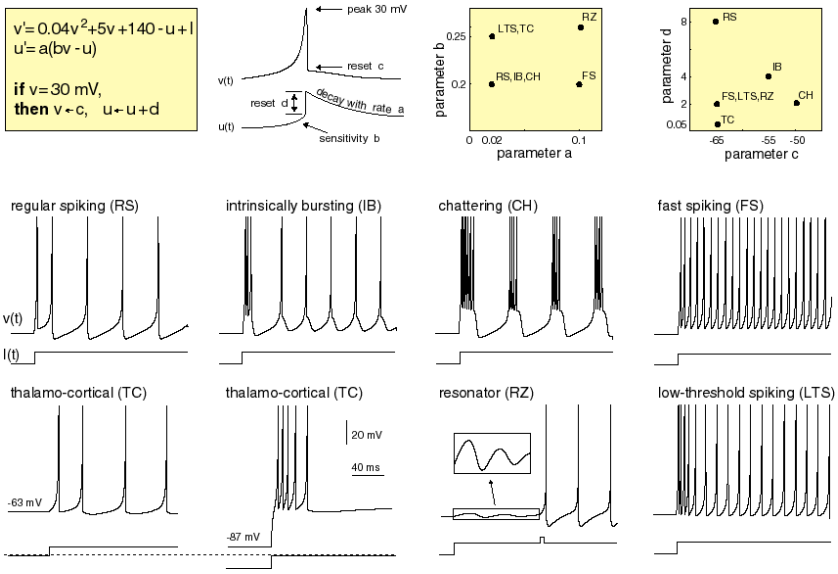
Spiking:
If $V = V_t$ (threshold)
then: neuron spikes and $V \rightarrow V_r$ (reset)



Different Firing modes for Thalamic Neurons



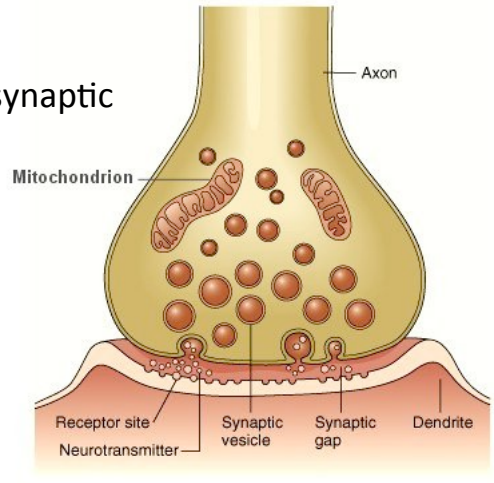
Edward Perez-Reyes, Physiological Reviews, 83:1, 2003



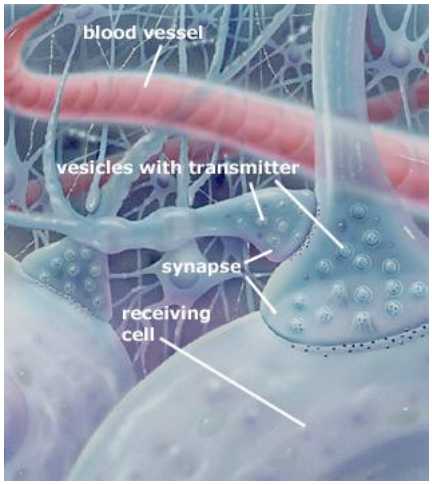
http://www.izhikevich.org/publications/spikes.htm

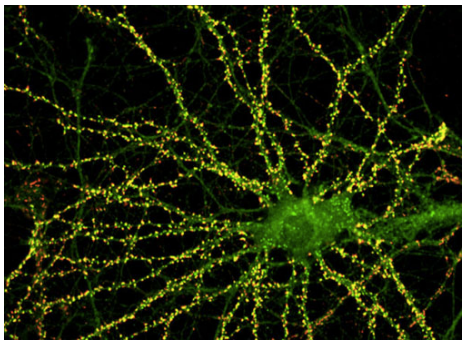
Synapse

Pre-synaptic



Post-synaptic





Synapse

- Functional chemical connection between neurons through close contact (20-50 nm)
- to dendrite: to soma; to hillock : to dendrite.
- Neurons may have 1000's of synapses
- vesicles containing neurotransmitter 40 nm

Synapse- Transmission

- Action potential reaches pre-synaptic terminal
- pre-synaptic potential opens voltage dependent Ca channels
- Ca causes vesicle migration, fusion to pre-synaptic membrane
- transmitter release via exocytosis - release in quantized amounts
- transmitter substance diffuses across cleft toward post-synaptic membrane

Synapse- Transmission (cont)

- Transmitter binds to receptor sites, causing membrane permeability change to ions
- post-synaptic potential is generated; can be EPSP (excitatory) or IPSP (inhibitory), depending on synapse type
- if excitatory and EPSP exceeds threshold, action potential is generated
- transmitter is broken down by enzyme in cleft
- products of hydrolysis are taken up by presynaptic site

Synapse - characteristics

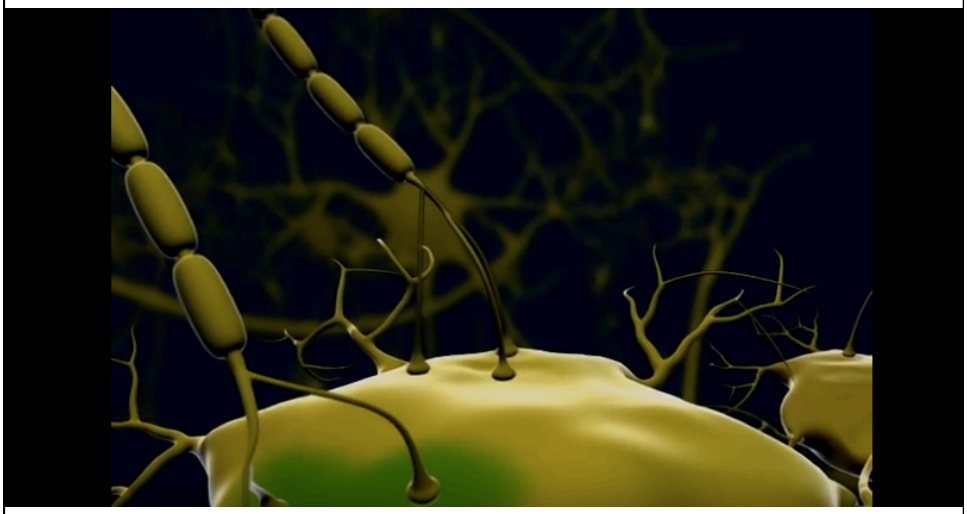
- Delay time for vesicle release, diffusion and binding (approx 0.5 msec)
- unidirectional action- propagation is from pre to post-synaptic fiber
- response is graded- proportional to rate of release of neurotransmitter, firing rate of neurons
- shows summation: temporal (same synapse) or spatial (different synapses at same time)
- shows fatigue: depletion of neurotransmitter

Synapse - neurotransmitters

- Glutamate- excitatory -
  - AMPA/kainate (fast, 1msec rise, 5 msec fall) receptors
  - NMDA (slow, 20 msec rise and 25-125 msec fall) receptors
- γ- aminobutyric acid (GABA) is primary inhibitory neurotransmitter
  - $GABA_A$ - fast inhibition receptors
  - $GABA_B$ - uses second messenger (slow) receptors

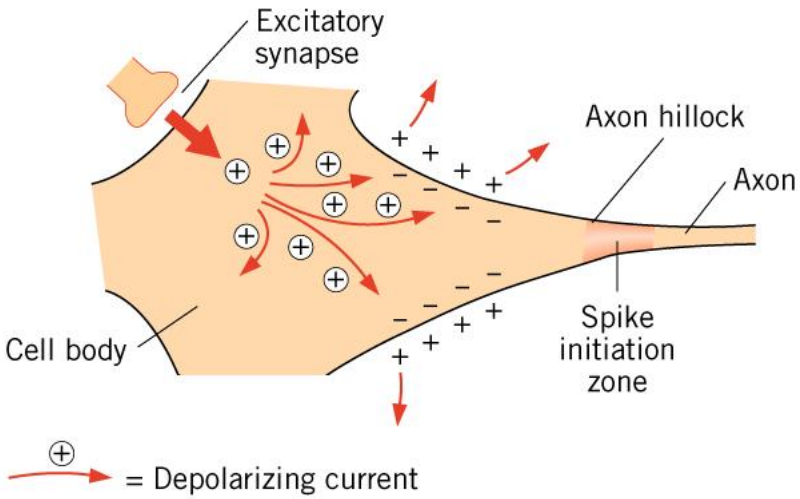
Synapse - neuromodulators

- neuromodulatory transmitters are secreted by a small group of neurons and diffuse through large areas of the nervous system, having an effect on multiple neurons. Examples of neuromodulators include dopamine, serotonin, acetylcholine, histamine and others.

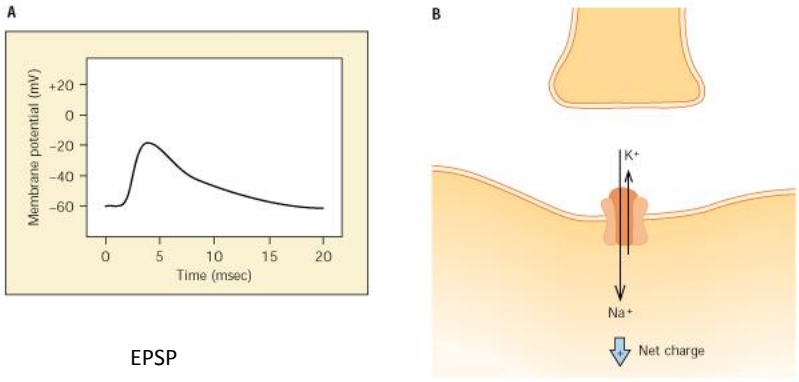


https://www.youtube.com/watch?v=lbzfwtdtong

Synaptic currents drive spike generation

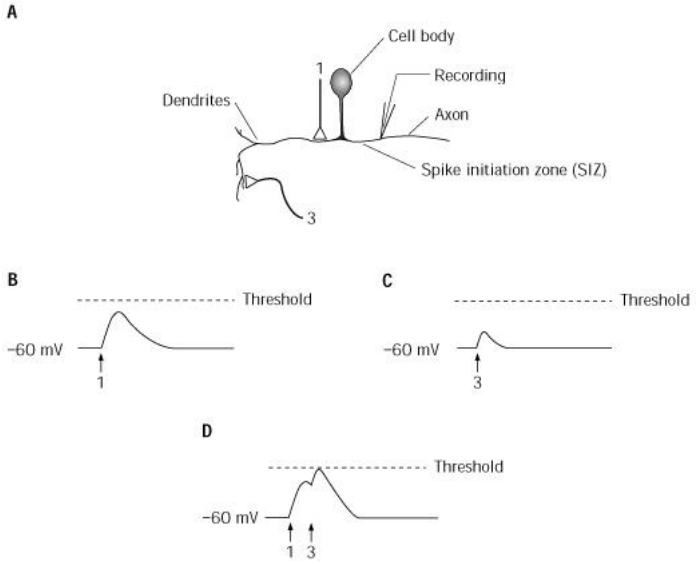


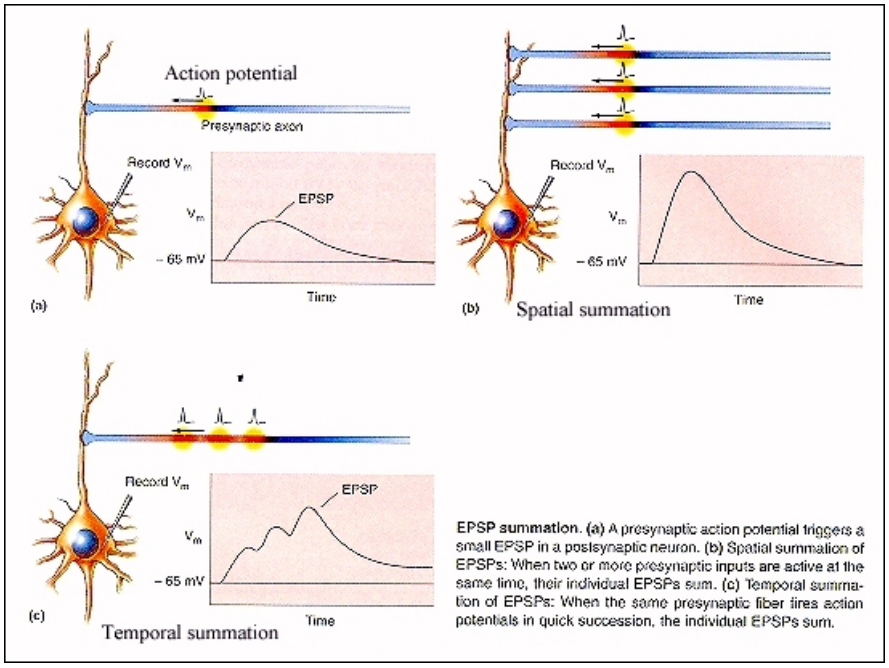
Synaptic transmission



EPSP

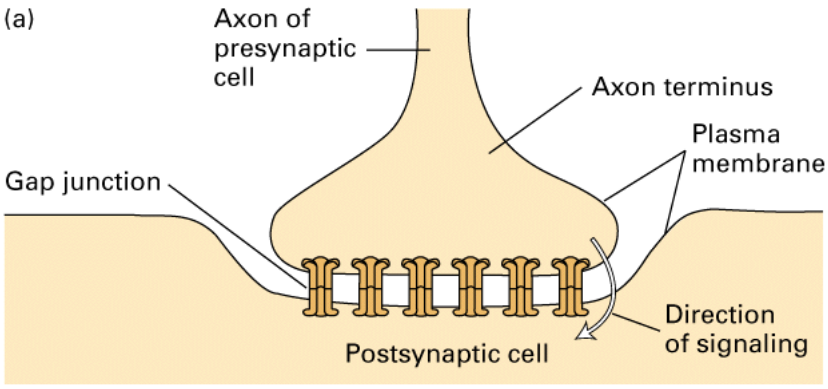
Summation





EPSP summation. (a) A presynaptic action potential triggers a small EPSP in a postsynaptic neuron. (b) Spatial summation of EPSPs: When two or more presynaptic inputs are active at the same time, their individual EPSPs sum. (c) Temporal summation of EPSPs: When the same presynaptic fiber fires action potentials in quick succession, the individual EPSPs sum.

Electrical Connections- Gap Junctions



Neuroplasticity

- Neuroplasticity (sometimes called brain plasticity, cortical plasticity or cortical re-mapping) refers to the changes that occur in the organization of the brain as a result of experience or learning. Can be a change in the wiring. Can be a change in the strength of connections.

CNS Tutorial T3

Modelling of spiking neural networks with the Brian simulator

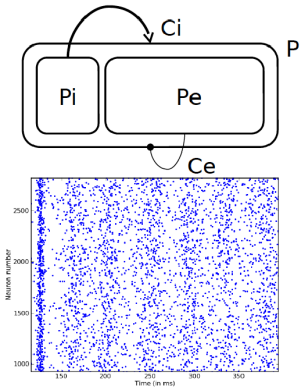
Marcel Stimberg Dan Goodman

An example

```
from brian2 import *

tau_m = 20*ms
tau_e = 5*ms
tau_i = 10*ms
V_th = -50*mV
E_L = -60*mV
I_const = 11*mV
eqs = '''
dv/dt = (-(v-E_L) + I_const + g_e + g_i)/tau_m : volt
dg_e/dt = -g_e/tau_e : volt
dg_i/dt = -g_i/tau_i : volt
'''
P = NeuronGroup(4000, model=eqs,
                threshold='v>V_th', reset='v=E_L')
P.v = 'E_L+10*mV*rand()'
Pe = P[:3200]
Pi = P[3200:]

w_e = 1.62*mV
w_i = 9*mV
Ce = Synapses(Pe, P, pre='g_e+=w_e')
Ci = Synapses(Pi, P, pre='g_i-=w_i')
Ce.connect(True, p=0.02)
Ci.connect(True, p=0.02)

M = SpikeMonitor(P)

run(1*second)
plot(M.t/ms, M.i, '.')
xlabel('Time (in ms)'); ylabel('Neuron number')
show()
```

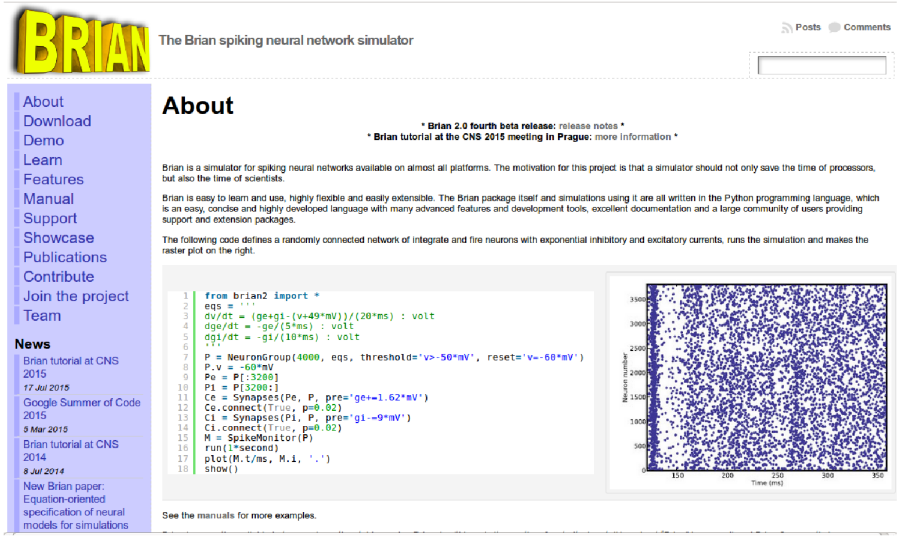
Stimberg M, Goodman DFM, Benichoux V, Brette R (2014).Equation-oriented specification of neural models for simulations. Frontiers Neuroinf, doi: 10.3389/fninf.2014.00006.

$$\tau_m \frac{\mathrm{d}V}{\mathrm{d}t} = -(V - E_L) + g_e + g_i$$
$$\tau_e \frac{\mathrm{d}g_e}{\mathrm{d}t} = -g_e$$
$$\tau_i \frac{\mathrm{d}g_i}{\mathrm{d}t} = -g_i$$



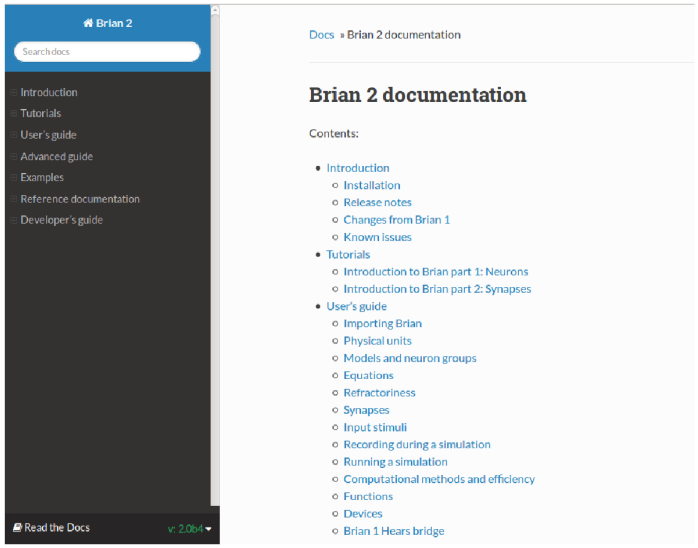
Website

briansimulator.org



Documentation

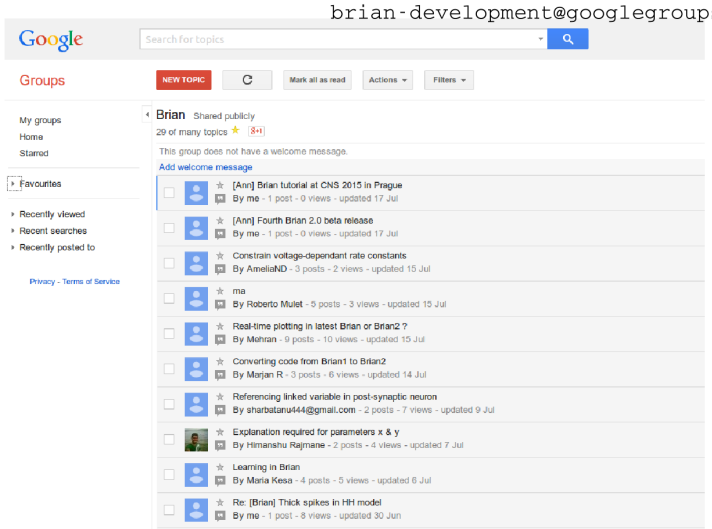
`brian2.readthedocs.org`



Mailing lists

`briansupport@googlegroups.com`

`brian-development@googlegroups.com`



Anatomy of a Brian script

```
from brian2 import *

tau_m = 20*ms
tau_e = 5*ms
tau_i = 10*ms
V_th = -50*mV
E_L = -60*mV
I_const = 11*mV
eqs = '''
dv/dt = (-(v-E_L) + I_const + g_e + g_i)/tau_m : volt
dg_e/dt = -g_e/tau_e : volt
dg_i/dt = -g_i/tau_i : volt
'''
P = NeuronGroup(4000, model=eqs,
                threshold='v>V_th', reset='v=E_L')
P.v = 'E_L+10*mV*rand()'
Pe = P[:3200]
Pi = P[3200:]

w_e = 1.62*mV
w_i = 9*mV
Ce = Synapses(Pe, P, pre='g_e+=w_e')
Ci = Synapses(Pi, P, pre='g_i-=w_i')
Ce.connect(True, p=0.02)
Ci.connect(True, p=0.02)

M = SpikeMonitor(P)

run(1*second)
plot(M.t/ms, M.i, '.')
xlabel('Time (in ms)'); ylabel('Neuron number')
show()
```

Import Brian library

Set constants

Specify the neuronal model

Initialise state variables

Specify subgroups of neurons

Specify synaptic model

Connect neurons

Record activity

Run the simulation

Plot the activity

Constants

- Constants defined outside of strings can be used inside strings (only scalar values):

```
tau = 10*ms
G = NeuronGroup(1, 'dv/dt = -v / tau : volt')
```

- Values of constants are resolved at the **run** call:

```
tau = 10*ms
G = NeuronGroup(1, 'dv/dt = -v / tau : volt')
run(100*ms)
tau = 20*ms
run(100*ms)
```

will be used for second run

Variables and equations

Equations define *state variables* of an object:

```
tau = 10*ms
G = NeuronGroup(5, '''dv/dt = (-v + inp) / tau                : volt (unless refractory)
                      inp = a * clip(sin(2*pi*freq*t), 0, inf) : volt
                      freq                                     : Hz   (constant)
                      a                                        : volt (shared)
                   ''', threshold='v>20*mV', reset='v = 0*mV',
                   refractory=2*ms)
G.freq = [100, 200, 300, 400, 500] * Hz
G.a = 100*mV
G.v = 10*mV
print G.inp  # not a state variable, but can be accessed like one
```

Additional variables are defined automatically

```
>>> print(G.variables.keys())
['a', '_spikespace', 'i', 'inp', 'N', 't', 'v', 'dt', 'lastspike', 'freq', 'not_refractory']
```

neuron index · number of neurons · time step · time of last spike · refractory?

Variables and equations

A Hodgkin-Huxley equations

```
G = NeuronGroup(number_of_neurons,
                '''dv/dt = (I_l+I_Na+I_K)/C_m   : volt # membrane potential
                   I_l = g_l*(-v+E_l)           : amp  # passive current
                   I_Na = g_Na*(m**3)*h*(-v+E_Na) : amp  # sodium current
                   I_K = g_K*(n**4)*(-v+E_K)    : amp  # potassium current
                   ...# equations for n, m, h''')
```

B Noisy membrane

```
G = NeuronGroup(number_of_neurons,
                'dv/dt = -(v-v_0)/tau_m +
                        tau_m**-0.5*3*mV*xi : volt # membrane potential')
```

special variable provided for noise

C Leaky integrate-and-fire neuron

```
G = NeuronGroup(number_of_neurons,
                'dv/dt = -(v-v_0)/tau_m : volt # membrane potential',
                threshold='v > v_th', reset='v = v_0')
```

D Leaky integrate-and-fire neuron with adaptive threshold

```
G = NeuronGroup(number_of_neurons,
                '''dv/dt = -(v-v_0)/tau_m          : volt # membrane potential
                   dv_th/dt = -(v_th-v_th0)/tau_th : volt # threshold''',
                threshold='v > v_th', reset='''v = v_0
                                              v_th += 3*mV''')
```

Expressions

- Can be used to
  - Specify conditions (threshold, refractoriness, synaptic connection)

```
G = NeuronGroup(10, 'dv/dt = -v / tau : 1 (unless refractory)',
                threshold='v > 1', refractory='(t-lastspike) < 3*ms')
```

  - Index and set state variables

```
min_freq = 100*Hz
print G.v['freq > min_freq']
G.v = '0*mV + rand()*10*mV'
```

- Can refer to state variables, constants, units

Brian's unit system

- Brian allows to use units for scalars and vectors

```
>>> E_L = -70*mV
>>> print E_L
-70.0 mV
>>> freqs = [100, 200, 300] * Hz
>>> print freqs
[ 100.  200.  300.] Hz
```

- Most numpy functions work correctly with units (make sure to not import from numpy directly)

```
>>> mean(freqs)
200.0 * hertz
>>> diff(freqs)
array([ 100.,  100.]) * hertz
```

Brian's unit system

- To remove units, use `numpy.asarray` or divide by the unit

```
>>> print freqs/Hz
[ 100.  200.  300.]
>>> print asarray(freqs)
[ 100.  200.  300.]
```

- For state variables: adding an underscore returns unitless value

```
>>> print G.v
<neurongroup.v: array([-70., -70., -70., -70., -70.]) * mvolt>
>>> print G.v_
<neurongroup_1.v: array([-0.07, -0.07, -0.07, -0.07, -0.07])>
```

- Unit consistency is also checked in equations, expressions and abstract code statements