
Hodgkin Huxley LEMS Tutorial Documentation

Release 0.1

Joe Bowen

December 21, 2014

CONTENTS

1	Hodgkin Huxley NeuroML/LEMS Neuron Model Tutorial	3
1.1	Membrane Capacitance	3
1.2	Sodium (Na) Ion Channel Variables	3
1.3	Potassium (K) Ion Channel Variables	3
1.4	Passive Leak Channel Variables	4
1.5	Time of Simulation	4
1.6	Input Current / Input Current Density	4
1.7	Channel Gating Kinetics for Sodium (Na) Channel m	5
1.8	Channel Gating Kinetics for Sodium (Na) Channel h	5
1.9	Channel Gating Kinetics for Potassium (K) channel n	6
1.10	Initial Values	6
1.11	Plots	6
2	Hodgkin Huxley Sources	9
2.1	Hodgkin Huxley.py	9
2.2	NML2_SingleCompHHCell.nml	11
2.3	LEMS_NML2_Ex5_DetCell.xml	12
2.4	run.sh	13
3	Indices and tables	15
	Python Module Index	17
	Index	19

Overview:

HODGKIN HUXLEY NEUROML/LEMS NEURON MODEL TUTORIAL

1.1 Membrane Capacitance

This variable from HodgkinHuxley.py:

```
C_m = 1.0
"""membrane capacitance, in uF/cm^2"""
```

Is used in this line in NML2_SingleCompHHCell.nml:

```
<specificCapacitance value="1.0 uF_per_cm2"/>
```

1.2 Sodium (Na) Ion Channel Variables

These variables from HodgkinHuxley.py:

```
g_Na = 120.0
"""Sodium (Na) maximum conducances, in mS/cm^2"""

E_Na = 50.0
"""Sodium (Na) Nernst reversal potentials, in mV"""
```

Is used in this line in NML2_SingleCompHHCell.nml:

```
<channelDensity id="naChans" ionChannel="naChan" condDensity="120.0 mS_per_cm2" erev="50.0"/>
```

1.3 Potassium (K) Ion Channel Variables

These variables from HodgkinHuxley.py:

```
g_K = 36.0
"""Potassium (K) maximum conducances, in mS/cm^2"""

E_K = -77.0
"""Potassium (K) Nernst reversal potentials, in mV"""
```

Is used in this line in NML2_SingleCompHHCell.nml:

```
<channelDensity id="kChans" ionChannel="kChan" condDensity="36 mS_per_cm2" erev="-77.0"/>
```

1.4 Passive Leak Channel Variables

These variables from HodgkinHuxley.py:

```
g_L = 0.3
"""Leak maximum conducances, in mS/cm^2"""

E_L = -54.387
"""Leak Nernst reversal potentials, in mV"""
```

Is used in this line in NML2_SingleCompHHCell.nml:

```
<channelDensity id="leak" ionChannel="passiveChan" condDensity="0.3 mS_per_cm2" erev=
```

1.5 Time of Simulation

This variable from HodgkinHuxley.py:

```
t = sp.arange(0.0, 450.0, 0.01)
""" The time to integrate over """
```

Is used in this line in LEMS_NML2_Ex5_DetCell.xml:

```
<Simulation id="sim1" length="450ms" step="0.01ms" target="net1">
```

1.6 Input Current / Input Current Density

The method from HodgkinHuxley.py takes the input in as a current density in the form of uA/cm². NeuroML/LEMS uses an input current in the form of nA, which requires a conversion in the input values.

This method from HodgkinHuxley.py:

```
1 def I_inj(self, t):
2     """
3     External Current
4
5     | :param t: time
6     | :return: step up to 10 uA/cm^2 at t>100
7     |           step down to 0 uA/cm^2 at t>200
8     |           step up to 35 uA/cm^2 at t>300
9     |           step down to 0 uA/cm^2 at t>400
10    """
11    return 10*(t>100) - 10*(t>200) + 35*(t>300) - 35*(t>400)
```

By using a given surface area of 1000.0 um² in the cell, it makes the conversion from uA/cm² to nA easier.

$$SurfaceArea = 4 * \pi * (radius)^2 = 4 * \pi * (diameter/2)^2 = 4 * \pi * (17.841242/2)^2 = 4 * \pi * (8.920621)^2 = 1000um^2$$

```
<segment id="0" name="soma">
  <proximal x="0" y="0" z="0" diameter="17.841242"/> <!--Gives a convenient surface area-->
  <distal x="0" y="0" z="0" diameter="17.841242"/>
</segment>
```


Given a surface area of 1000.0 μm^2 in the cell the following equation is used to convert from X $\mu\text{A}/\text{cm}^2$ to Y nA:

$$(XuA/cm^2) * (1000.0\mu m^2) * (1000nA/uA)/(1 * 10^8\mu m^2/cm^2) = YnA$$

Line 11 can then be translated into the delay, duration and amplitude of the two pulseGenerator objects in NML2_SingleCompHHCell.nml:

```
<pulseGenerator id="pulseGen1" delay="100ms" duration="100ms" amplitude="0.10nA"/>
<pulseGenerator id="pulseGen2" delay="300ms" duration="100ms" amplitude="0.35nA"/>
```

1.7 Channel Gating Kinetics for Sodium (Na) Channel m

Functions of membrane voltage

These methods from HodgkinHuxley.py:

```
1 def alpha_m(self, V):
2     """Channel gating kinetics. Functions of membrane voltage"""
3     return 0.1*(V+40.0)/(1.0 - sp.exp(-(V+40.0) / 10.0))

1 def beta_m(self, V):
2     """Channel gating kinetics. Functions of membrane voltage"""
3     return 4.0*sp.exp(-(V+65.0) / 18.0)
```

Are used in these lines in NML2_SingleCompHHCell.nml:

```
<gateHHrates id="m" instances="3">
  <forwardRate type="HHExpLinearRate" rate="1per_ms" midpoint="-40mV" scale="10mV"/>
  <reverseRate type="HHExpRate" rate="4per_ms" midpoint="-65mV" scale="-18mV"/>
</gateHHrates>
```

1.8 Channel Gating Kinetics for Sodium (Na) Channel h

Functions of membrane voltage

These methods from HodgkinHuxley.py:

```
1 def alpha_h(self, V):
2     """Channel gating kinetics. Functions of membrane voltage"""
3     return 0.07*sp.exp(-(V+65.0) / 20.0)

1 def beta_h(self, V):
2     """Channel gating kinetics. Functions of membrane voltage"""
3     return 1.0/(1.0 + sp.exp(-(V+35.0) / 10.0))
```

Are used in these lines in NML2_SingleCompHHCell.nml:

```
<gateHHrates id="h" instances="1">
  <forwardRate type="HHExpRate" rate="0.07per_ms" midpoint="-65mV" scale="-20mV"/>
  <reverseRate type="HHSigmoidRate" rate="1per_ms" midpoint="-35mV" scale="10mV"/>
</gateHHrates>
```

1.9 Channel Gating Kinetics for Potassium (K) channel n

Functions of membrane voltage

These methods from HodgkinHuxley.py:

```
1 def alpha_n(self, V):
2     """Channel gating kinetics. Functions of membrane voltage"""
3     return 0.01*(V+55.0)/(1.0 - sp.exp(-(V+55.0) / 10.0))

1 def beta_n(self, V):
2     """Channel gating kinetics. Functions of membrane voltage"""
3     return 0.125*sp.exp(-(V+65) / 80.0)
```

Are used in these lines in NML2_SingleCompHHCell.nml:

```
<gateHHrates id="n" instances="4">
  <forwardRate type="HHExpLinearRate" rate="0.1per_ms" midpoint="-55mV" scale="10mV"/>
  <reverseRate type="HHExpRate" rate="0.125per_ms" midpoint="-65mV" scale="-80mV"/>
</gateHHrates>
```

1.10 Initial Values

This line from HodgkinHuxley.py:

```
X = odeint(self.dALLdt, [-65, 0.05, 0.6, 0.32], self.t, args=(self,))
```

Is used to define the initial values for the model in NML2_SingleCompHHCell.nml:

```
<initMembPotential value="-65mV"/>
```

Where do the rest of these initial values from HodgkinHuxley.py fit into the NeuroML/LEMS Model?

1.11 Plots

This line in HodgkinHuxley.py:

```
plt.subplot(4,1,1)
plt.title('Hodgkin-Huxley Neuron')
plt.plot(self.t, V, 'k')
plt.ylabel('V (mV)')
```

Is used in these lines in LEMS_NML2_Ex5_DetCell.xml:

```
<Display id="d1" title="Hodgkin-Huxley Neuron: V (mV)" timeScale="1ms" xmin="0" xmax="450" y
  <Line id="v" quantity="hhpop[0]/v" scale="1mV" color="#ffffff" timeScale="1ms"/>
</Display>
```

This line in HodgkinHuxley.py:

```
plt.subplot(4,1,2)
plt.plot(self.t, ina, 'c', label='$I_{Na}$')
plt.plot(self.t, ik, 'y', label='$I_{K}$')
plt.plot(self.t, il, 'm', label='$I_{L}$')
plt.ylabel('Current')
plt.legend()
```

Is used in these lines in LEMS_NML2_Ex5_DetCell.xml:

```
<Display id="d3" title="Hodgkin-Huxley Neuron: Current" timeScale="1ms" xmin="0" xmax="450" y
  <Line id="I_na" quantity="hhpop[0]/bioPhys1/membraneProperties/naChans/iDensity" scale="1"
  <Line id="I_k" quantity="hhpop[0]/bioPhys1/membraneProperties/kChans/iDensity" scale="1"
  <Line id="I_l" quantity="hhpop[0]/bioPhys1/membraneProperties/leak/iDensity" scale="1"
</Display>
```

This line in HodgkinHuxley.py:

```
plt.subplot(4,1,3)
plt.plot(self.t, m, 'r', label='m')
plt.plot(self.t, h, 'g', label='h')
plt.plot(self.t, n, 'b', label='n')
plt.ylabel('Gating Value')
plt.legend()
```

Is used in these lines in LEMS_NML2_Ex5_DetCell.xml:

```
<Display id="d2" title="Hodgkin-Huxley Neuron: Gating Value" timeScale="1ms" xmin="0" xmax="4
  <Line id="m" quantity="hhpop[0]/bioPhys1/membraneProperties/naChans/naChan/m/q" scale="1"
  <Line id="h" quantity="hhpop[0]/bioPhys1/membraneProperties/naChans/naChan/h/q" scale="1"
  <Line id="n" quantity="hhpop[0]/bioPhys1/membraneProperties/kChans/kChan/n/q" scale="1"
</Display>
```

This line in HodgkinHuxley.py:

```
plt.subplot(4,1,4)
plt.plot(self.t, self.I_inj(self.t), 'k')
plt.xlabel('t (ms)')
plt.ylabel('$I_{inj}$ ($\mu A/cm^2$)')
plt.ylim(-1, 40)
```

Is used in these lines in LEMS_NML2_Ex5_DetCell.xml:

```
<Display id="d4" title="Hodgkin-Huxley Neuron: I_inj ( nA )" timeScale="1ms" xmin="0" xmax="4
  <Line id="I_inj1" quantity="hhpop[0]/pulseGen1/i" scale="1nA" color="#004040" timeScale=
  <Line id="I_inj2" quantity="hhpop[0]/pulseGen2/i" scale="1nA" color="#004040" timeScale=
</Display>
```


HODGKIN HUXLEY SOURCES

2.1 Hodgkin Huxley.py

Source Package

```

class Source.HodgkinHuxley.HodgkinHuxley
    Full Hodgkin-Huxley Model (copied from Computational Lab 2)

    C_m = 1.0
        membrane capacitance, in uF/cm^2

    g_Na = 120.0
        Sodium (Na) maximum conducances, in mS/cm^2

    g_K = 36.0
        Potassium (K) maximum conducances, in mS/cm^2

    g_L = 0.3
        Leak maximum conducances, in mS/cm^2

    E_Na = 50.0
        Sodium (Na) Nernst reversal potentials, in mV

    E_K = -77.0
        Potassium (K) Nernst reversal potentials, in mV

    E_L = -54.387
        Leak Nernst reversal potentials, in mV

    t = array([ 0.00000000e+00, 1.00000000e-02, 2.00000000e-02, ..., 4.49970000e+02, 4.49980000e+02, 4.49990000e+02])
        The time to integrate over

    alpha_m(V)
        Channel gating kinetics. Functions of membrane voltage

    beta_m(V)
        Channel gating kinetics. Functions of membrane voltage

    alpha_h(V)
        Channel gating kinetics. Functions of membrane voltage

    beta_h(V)
        Channel gating kinetics. Functions of membrane voltage

    alpha_n(V)
        Channel gating kinetics. Functions of membrane voltage

    beta_n(V)
        Channel gating kinetics. Functions of membrane voltage

```

I_Na (*V, m, h*)

Membrane current (in uA/cm²) Sodium (Na = element name)

:param V:

:param m:

:param h:

:return:

I_K (*V, n*)

Membrane current (in uA/cm²) Potassium (K = element name)

:param V:

:param h:

:return:

I_L (*V*)

Membrane current (in uA/cm²) Leak

:param V:

:param h:

:return:

I_inj (*t*)

External Current

:param t: time

:return: step up to 10 uA/cm² at t>100

step down to 0 uA/cm² at t>200

step up to 35 uA/cm² at t>300

step down to 0 uA/cm² at t>400

static dALLdt (*X, t, self*)

Integrate

:param X:

:param t:

:return: calculate membrane potential & activation variables

Main ()

Main demo for the Hodgkin Huxley neuron model

2.2 NML2_SingleCompHHCell.nml

```
<?xml version="1.0" encoding="UTF-8"?>

<neuroml xmlns="http://www.neuroml.org/schema/neuroml2"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.neuroml.org/schema/neuroml2 ../Schemas/NeuroML2/NeuroML_v2bet
  id="NML2_SingleCompHHCell">

  <!-- Single compartment cell with HH channels -->

  <!-- This is a "pure" NeuroML 2 file. It can be included in a LEMS file for use in a simulation
  by the LEMS interpreter, see LEMS_NML2_Ex5_DetCell.xml -->

  <!-- This is a modified version to duplicate the values and charts used in
  https://gist.github.com/slarson/37463b35ef8606629d2e#file-hodgkin-huxley-py -->

  <ionChannelHH id="passiveChan" conductance="10pS">
    <notes>Leak conductance</notes>
  </ionChannelHH>

  <ionChannelHH id="naChan" conductance="10pS" species="na">
    <notes>Na channel</notes>

    <gateHHrates id="m" instances="3">
      <forwardRate type="HHExpLinearRate" rate="1per_ms" midpoint="-40mV" scale="10mV"/>
      <reverseRate type="HHExpRate" rate="4per_ms" midpoint="-65mV" scale="-18mV"/>
    </gateHHrates>

    <gateHHrates id="h" instances="1">
      <forwardRate type="HHExpRate" rate="0.07per_ms" midpoint="-65mV" scale="-20mV"/>
      <reverseRate type="HHSigmoidRate" rate="1per_ms" midpoint="-35mV" scale="10mV"/>
    </gateHHrates>
  </ionChannelHH>

  <ionChannelHH id="kChan" conductance="10pS" species="k">

    <gateHHrates id="n" instances="4">
      <forwardRate type="HHExpLinearRate" rate="0.1per_ms" midpoint="-55mV" scale="10mV"/>
      <reverseRate type="HHExpRate" rate="0.125per_ms" midpoint="-65mV" scale="-80mV"/>
    </gateHHrates>
  </ionChannelHH>

  <cell id="hhcell">

    <morphology id="morph1">
      <segment id="0" name="soma">
        <proximal x="0" y="0" z="0" diameter="17.841242"/> <!--Gives a convenient surface ar
        <distal x="0" y="0" z="0" diameter="17.841242"/>
      </segment>

      <segmentGroup id="soma_group">
```

```
<member segment="0"/>
</segmentGroup>

</morphology>

<biophysicalProperties id="bioPhys1">

  <membraneProperties>

    <channelDensity id="leak" ionChannel="passiveChan" condDensity="0.3 mS_per_cm2" erev="0"/>
    <channelDensity id="naChans" ionChannel="naChan" condDensity="120.0 mS_per_cm2" erev="55"/>
    <channelDensity id="kChans" ionChannel="kChan" condDensity="36 mS_per_cm2" erev="-77"/>

    <spikeThresh value="-20mV"/>
    <specificCapacitance value="1.0 uF_per_cm2"/>
    <initMembPotential value="-65mV"/>

  </membraneProperties>

  <intracellularProperties>
    <resistivity value="0.03 kohm_cm"/>    <!-- Note: not used in single compartment simulation -->
  </intracellularProperties>

</biophysicalProperties>

</cell>

<pulseGenerator id="pulseGen1" delay="100ms" duration="100ms" amplitude="0.10nA"/>
<pulseGenerator id="pulseGen2" delay="300ms" duration="100ms" amplitude="0.35nA"/>

<network id="net1">
  <population id="hhpop" component="hhcell" size="1"/>
  <explicitInput target="hhpop[0]" input="pulseGen1"/>
  <explicitInput target="hhpop[0]" input="pulseGen2"/>
</network>

</neuroml>
```

2.3 LEMS_NML2_Ex5_DetCell.xml

```
<Lems>

  <!-- Example with Simple Hodgkin-Huxley cell specifying segment details-->

  <!-- This is a file which can be read and executed by the LEMS Interpreter.
       It imports the LEMS definitions of the core NeuroML 2 Components,
       imports in "pure" NeuroML 2 and contains some LEMS elements for running
       a simulation -->

  <!-- This is a modified version to duplicate the values and charts used in
       https://gist.github.com/slarson/37463b35ef8606629d2e#file-hodgkin-huxley-py -->

  <Target component="sim1"/>

  <Include file="Cells.xml"/>
  <Include file="Networks.xml"/>
```



```

<Include file="Simulation.xml"/>

<!-- Including file with a <neuroml> root, a "real" NeuroML 2 file -->
<Include file="./NML2_SingleCompHHCell.nml"/>

<Simulation id="sim1" length="450ms" step="0.01ms" target="net1">

  <Display id="d1" title="Hodgkin-Huxley Neuron: V (mV)" timeScale="1ms" xmin="0" xmax="450" y
    <Line id="v" quantity="hhpop[0]/v" scale="1mV" color="#ffffff" timeScale="1ms"/>
  </Display>

  <Display id="d2" title="Hodgkin-Huxley Neuron: Gating Value" timeScale="1ms" xmin="0" xmax="
    <Line id="m" quantity="hhpop[0]/bioPhys1/membraneProperties/naChans/naChan/m/q" scale="1
    <Line id="h" quantity="hhpop[0]/bioPhys1/membraneProperties/naChans/naChan/h/q" scale="1
    <Line id="n" quantity="hhpop[0]/bioPhys1/membraneProperties/kChans/kChan/n/q" scale="1"
  </Display>

  <Display id="d3" title="Hodgkin-Huxley Neuron: Current" timeScale="1ms" xmin="0" xmax="450" y
    <Line id="I_na" quantity="hhpop[0]/bioPhys1/membraneProperties/naChans/iDensity" scale="1
    <Line id="I_k" quantity="hhpop[0]/bioPhys1/membraneProperties/kChans/iDensity" scale="1"
    <Line id="I_l" quantity="hhpop[0]/bioPhys1/membraneProperties/leak/iDensity" scale="1"
  </Display>

  <Display id="d4" title="Hodgkin-Huxley Neuron: I_inj ( nA )" timeScale="1ms" xmin="0" xmax="
    <Line id="I_inj1" quantity="hhpop[0]/pulseGen1/i" scale="1nA" color="#004040" timeScale=
    <Line id="I_inj2" quantity="hhpop[0]/pulseGen2/i" scale="1nA" color="#004040" timeScale=
  </Display>

</Simulation>

</Lems>

```

2.4 run.sh

```

#!/bin/bash
#####
# LEMS Hodgkin Huxley Neuron Model
#
# Command to run LEMS_NML2_Ex5_DetCell.xml script
#
# Usage: ./run.sh
#
#####

set -e
jnml LEMS_NML2_Ex5_DetCell.xml

```


INDICES AND TABLES

- *genindex*
- *modindex*
- *search*

S

`Source.HodgkinHuxley`, 9

A

alpha_h() (Source.HodgkinHuxley.HodgkinHuxley method), 9
 alpha_m() (Source.HodgkinHuxley.HodgkinHuxley method), 9
 alpha_n() (Source.HodgkinHuxley.HodgkinHuxley method), 9

B

beta_h() (Source.HodgkinHuxley.HodgkinHuxley method), 9
 beta_m() (Source.HodgkinHuxley.HodgkinHuxley method), 9
 beta_n() (Source.HodgkinHuxley.HodgkinHuxley method), 9

C

C_m (Source.HodgkinHuxley.HodgkinHuxley attribute), 9

D

dALLdt() (Source.HodgkinHuxley.HodgkinHuxley static method), 10

E

E_K (Source.HodgkinHuxley.HodgkinHuxley attribute), 9
 E_L (Source.HodgkinHuxley.HodgkinHuxley attribute), 9
 E_Na (Source.HodgkinHuxley.HodgkinHuxley attribute), 9

G

g_K (Source.HodgkinHuxley.HodgkinHuxley attribute), 9
 g_L (Source.HodgkinHuxley.HodgkinHuxley attribute), 9
 g_Na (Source.HodgkinHuxley.HodgkinHuxley attribute), 9

H

HodgkinHuxley (class in Source.HodgkinHuxley), 9

I

I_inj() (Source.HodgkinHuxley.HodgkinHuxley method), 10
 I_K() (Source.HodgkinHuxley.HodgkinHuxley method), 10
 I_L() (Source.HodgkinHuxley.HodgkinHuxley method), 10
 I_Na() (Source.HodgkinHuxley.HodgkinHuxley method), 10

M

Main() (Source.HodgkinHuxley.HodgkinHuxley method), 10

S

Source.HodgkinHuxley (module), 9

T

t (Source.HodgkinHuxley.HodgkinHuxley attribute), 9