01_exercises_katja

May 16, 2022

1 Problem Set 1: The Hodgkin-Huxley Model

1.1 Problem 1: Spiking threshold

[1]:
```
# NEST code used for simulation (copied from NEST desktop)
import nest
import numpy

nest.ResetKernel()


# Simulation kernel
nest.local_num_threads = 1
nest.resolution = 0.1
nest.rng_seed = 1


# Create nodes
dc1 = nest.Create("dc_generator", 1, params={
  "amplitude": 224
})
n1 = nest.Create("hh_psc_alpha", 1, params={
  "g_K": 3600,
  "g_Na": 12000
})
vm1 = nest.Create("voltmeter", params={
  "interval": 0.1
})
mm1 = nest.Create("multimeter", params={
  "interval": 0.1,
  "record_from": ["Act_m","Act_n","Inact_h"]
})


# Connect nodes
nest.Connect(dc1, n1)
nest.Connect(vm1, n1)
nest.Connect(mm1, n1)
```

```
# Run simulation
nest.Simulate(1000)


# Get IDs of recorded node
def getNodeIds(node):
  if node.model == "spike_recorder":
    return list(nest.GetConnections(None, node).sources())
  else:
    return list(nest.GetConnections(node).targets())


# Collect response
response = {
  "kernel": {
    "biological_time": nest.biological_time
  },
  "activities": [
    {"events": vm1.events, "nodeIds": getNodeIds(vm1)},
    {"events": mm1.events, "nodeIds": getNodeIds(mm1)}
  ]
}
```

```
             -- N E S T --
  Copyright (C) 2004 The NEST Initiative

 Version: 3.3
 Built: Mar 23 2022 13:33:55

 This program is provided AS IS and comes with
 NO WARRANTY. See the file LICENSE for details.

 Problems or suggestions?
   Visit https://www.nest-simulator.org

 Type 'nest.help()' to find out more about NEST.


May 10 16:54:14 SimulationManager::set_status [Info]:
    Temporal resolution changed from 0.1 to 0.1 ms.

May 10 16:54:14 NodeManager::prepare_nodes [Info]:
    Preparing 4 nodes for simulation.
```

```
May 10 16:54:14 SimulationManager::start_updating_ [Info]:
    Number of local nodes: 4
    Simulation time (ms): 1000
    Number of OpenMP threads: 1
    Not using MPI

May 10 16:54:14 SimulationManager::run [Info]:
    Simulation finished.
```

Answers: - The spiking threshold is at 225 pA (fires not before 224 pA [see figure 1.1: 224 pA with not spike] but for 225 pA [see figure 1.2: 225 pA with spike]). - The start and stop time of the input (duration) and the population size of the input also affect it's impact. - For the 225 pA input, there is a steep rise of the sodium channel activation gating variable m which allows the fast depolarization of the membrane at the beginning of the action potential through influx of sodium into the cell. This does not occur for the 224pA input and therefore no spike is elicited.

Figure 1.1: 224 pA with no spike

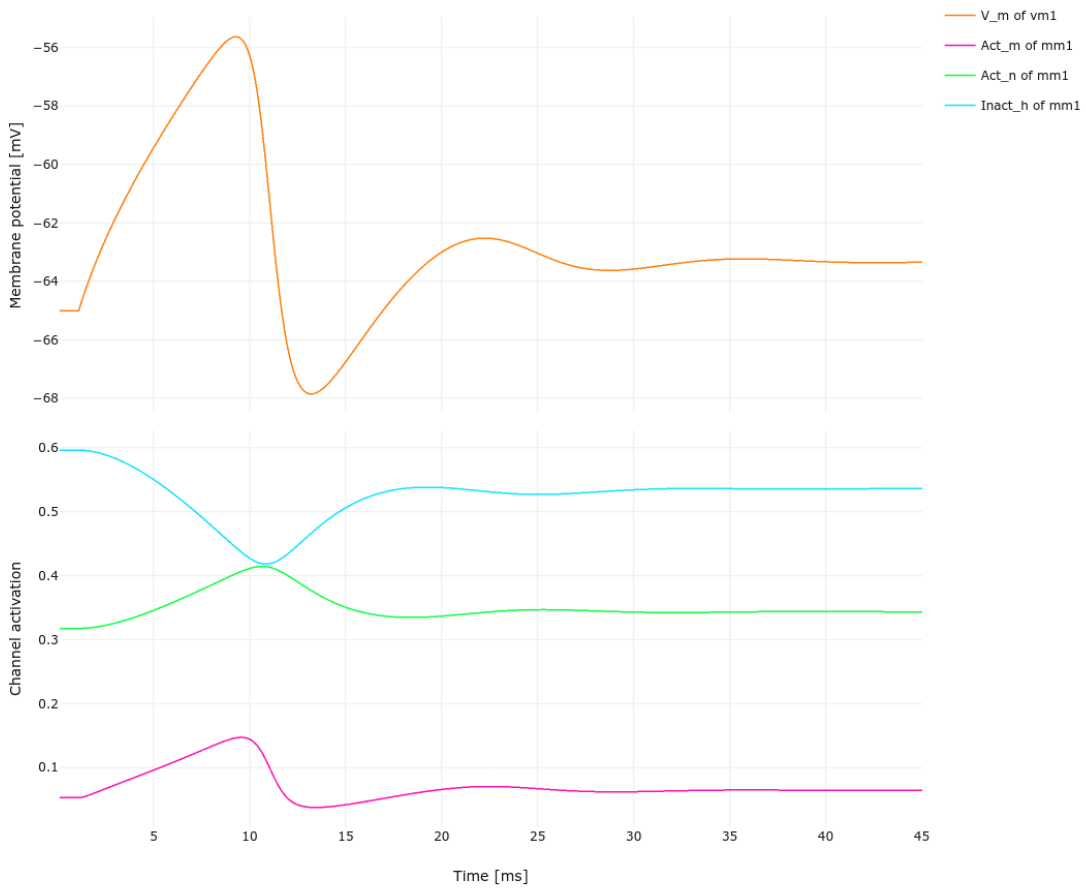
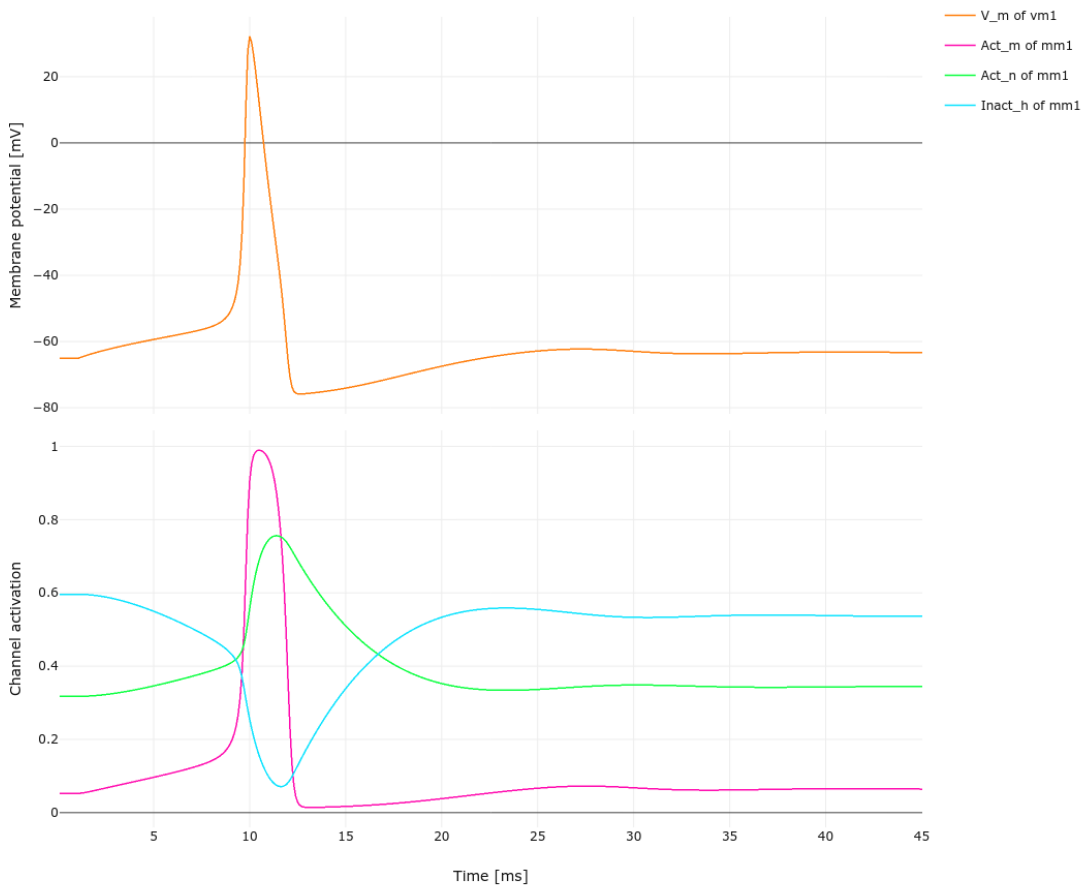


Figure 1.2: 225 pA with spike



1.2 Problem 2: Refractory period of the neuron

[2]:
```
# NEST code used for simulation (copied from NEST desktop)

import nest
import numpy

nest.ResetKernel()


# Simulation kernel
nest.local_num_threads = 1
nest.resolution = 0.1
nest.rng_seed = 1


# Create nodes
dc1 = nest.Create("dc_generator", 1, params={
  "amplitude": 1000
})
```

```
n1 = nest.Create("hh_psc_alpha", 1, params={
  "g_K": 3600,
  "g_Na": 12000
})
vm1 = nest.Create("voltmeter", params={
  "interval": 0.1
})
mm1 = nest.Create("multimeter", params={
  "interval": 0.1,
  "record_from": ["Act_m","Act_n","Inact_h"]
})
sr1 = nest.Create("spike_recorder")


# Connect nodes
nest.Connect(dc1, n1)
nest.Connect(vm1, n1)
nest.Connect(mm1, n1)
nest.Connect(n1, sr1)


# Run simulation
nest.Simulate(1000)


# Get IDs of recorded node
def getNodeIds(node):
  if node.model == "spike_recorder":
    return list(nest.GetConnections(None, node).sources())
  else:
    return list(nest.GetConnections(node).targets())


# Collect response
response = {
  "kernel": {
    "biological_time": nest.biological_time
  },
  "activities": [
    {"events": vm1.events, "nodeIds": getNodeIds(vm1)},
    {"events": mm1.events, "nodeIds": getNodeIds(mm1)},
    {"events": sr1.events, "nodeIds": getNodeIds(sr1)}
  ]
}
```

```
May 10 16:54:14 SimulationManager::set_status [Info]:
```

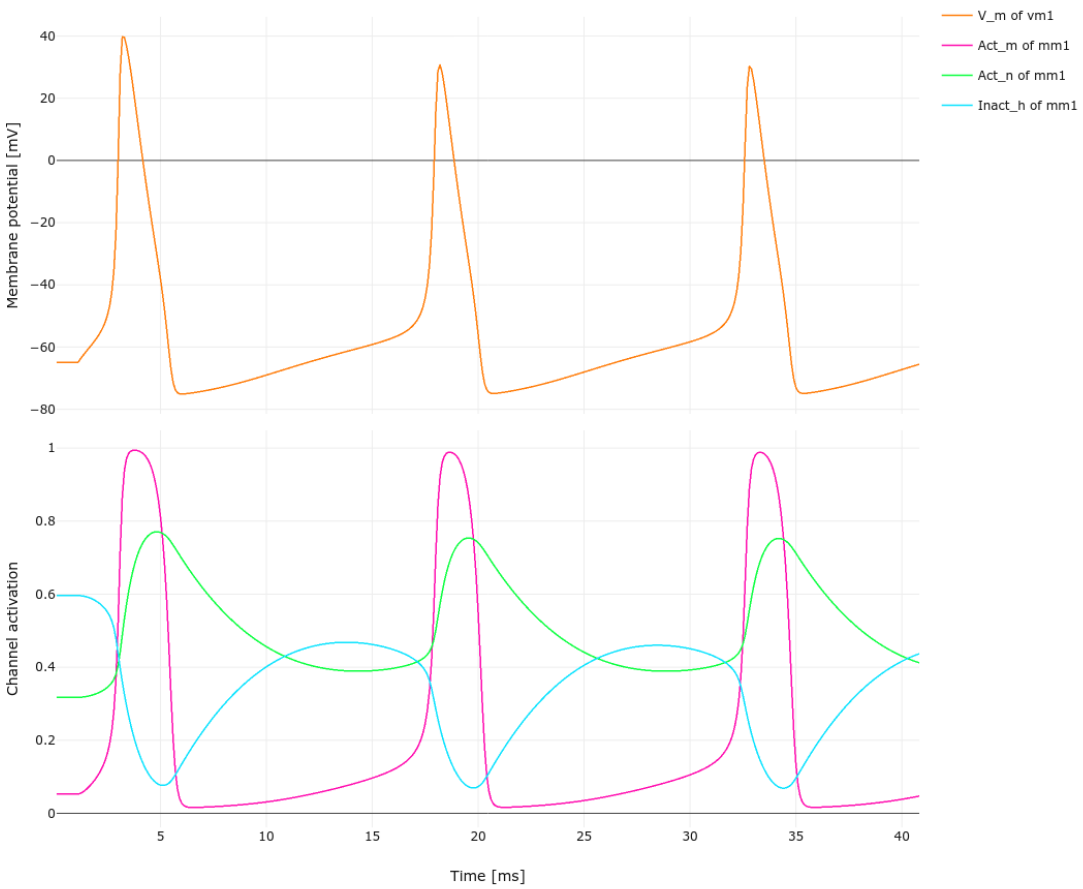
```
    Temporal resolution changed from 0.1 to 0.1 ms.

May 10 16:54:14 NodeManager::prepare_nodes [Info]:
    Preparing 5 nodes for simulation.

May 10 16:54:14 SimulationManager::start_updating_ [Info]:
    Number of local nodes: 5
    Simulation time (ms): 1000
    Number of OpenMP threads: 1
    Not using MPI

May 10 16:54:14 SimulationManager::run [Info]:
    Simulation finished.
```

[3]:
```
# calculate average ISI = refractory period
numpy.diff(response['activities'][2]['events']['times']).mean()
```

[3]: 14.644117647058824

Answers: - Figure: see figure 2.1: refractory period - The refractory period is mainly determined by the sodium channel inactivation gating variable h. As long as this inactivation inactvation is very pronounced, the neuron cannot fire (absolute refractory period). Furthermore, the prolonged potassium channel activation (gating variable n) channel inactivation works (relative refractory period, also still some inactivation of sodium channels). However, the crucial factor is the sodium channel inactivation. - The refractory period is on average 15 ms (average of inter-spike-intervals) for a direct current input of 1000 pA.

Figure 2.1: refractory period



1.3 Problem 3: Current input

Answers: - For the input range of [225 pA, 597 pA], only a single spike with no further spike is elicited (see figure 3.1: 225 pA and figure 3.2: 597 pA). - Looking at the gating variables reveals that at the border of 597 pA and 598 pA, for 597 pA, the sodium channel activation is not strong enough, but at 598 pA it exhibits a steep rise again for the second spike (see figure 3.3: 598 pA). Again, at the border of 224 pA and 225 pA, the sodium channel activation is also not strong enough (see figure 3.4: 224 pA). - Below the input range ($<$ 224 pA) there are not spikes at all. Above the input range, at least one further spike emerges. At 627 pA, a continuous spike train emerges and the neuron does not go to rest after a couple of spikes anymore (see figure 3.4: 627 pA).

Figure 3.1: 225 pA

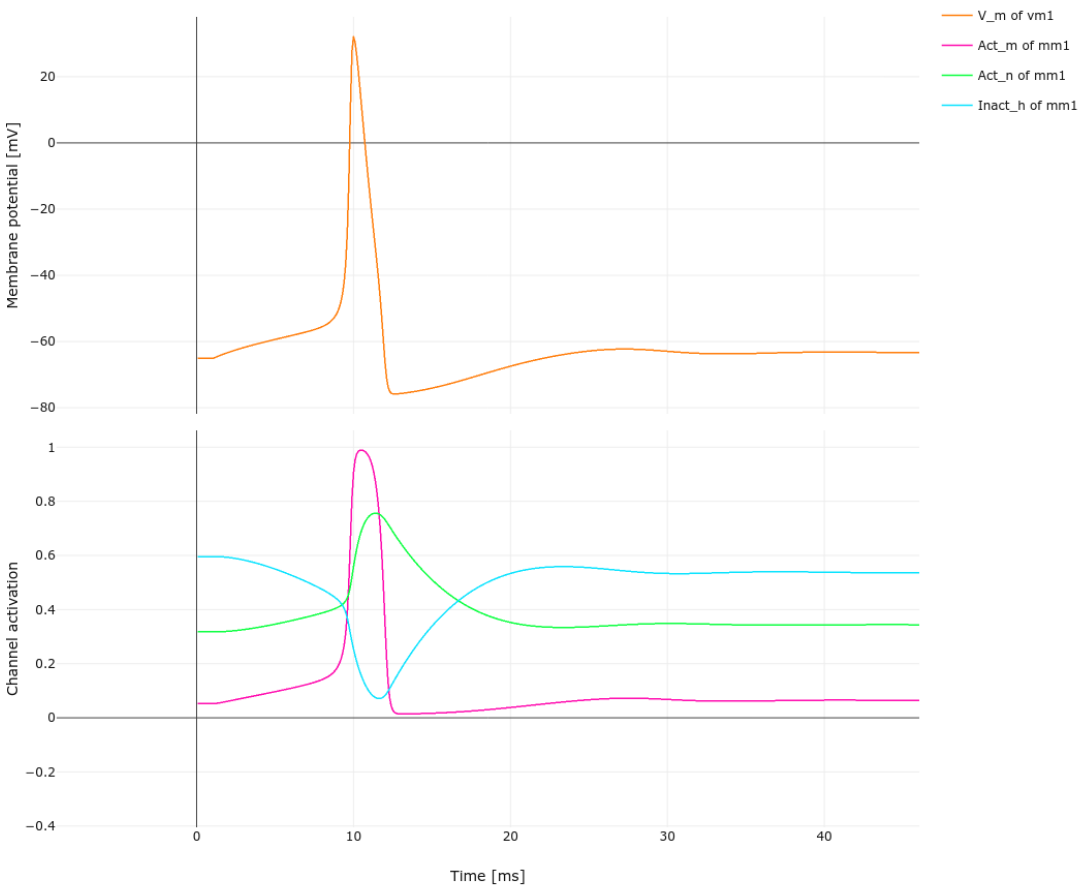


Figure 3.2: 597 pA

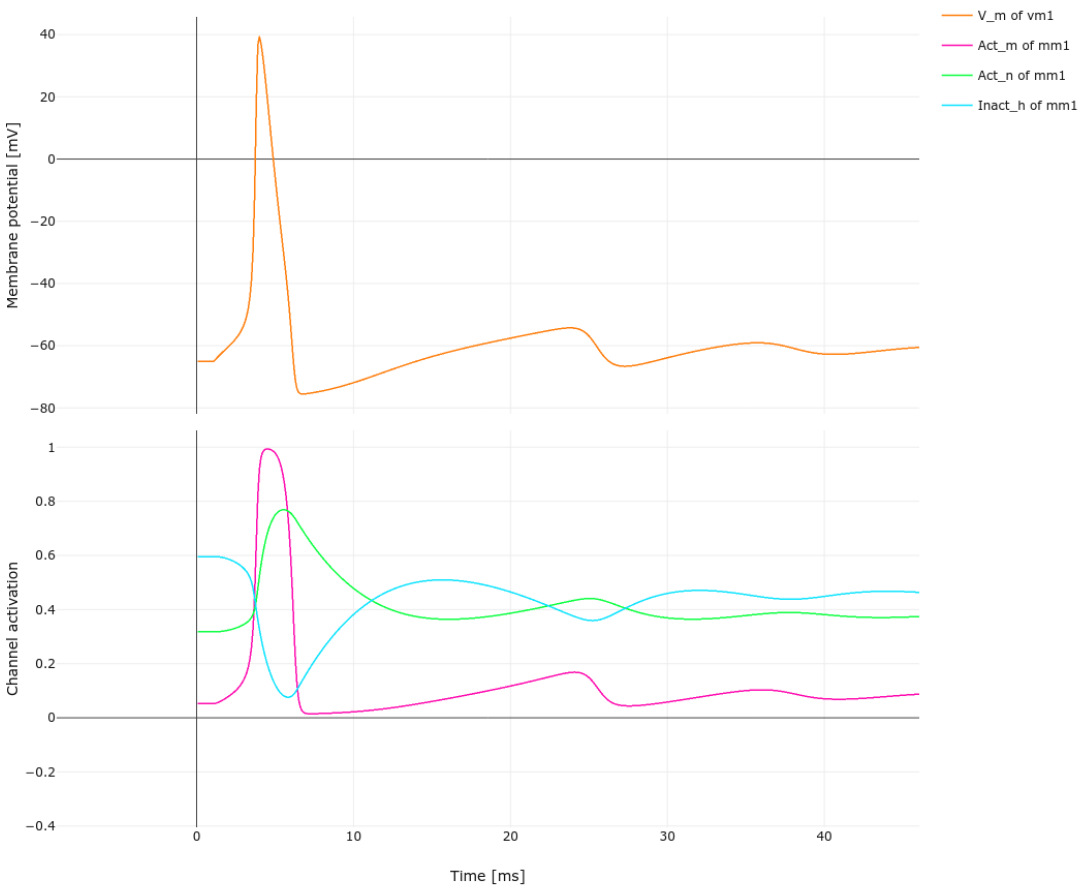


Figure 3.3: 598 pA

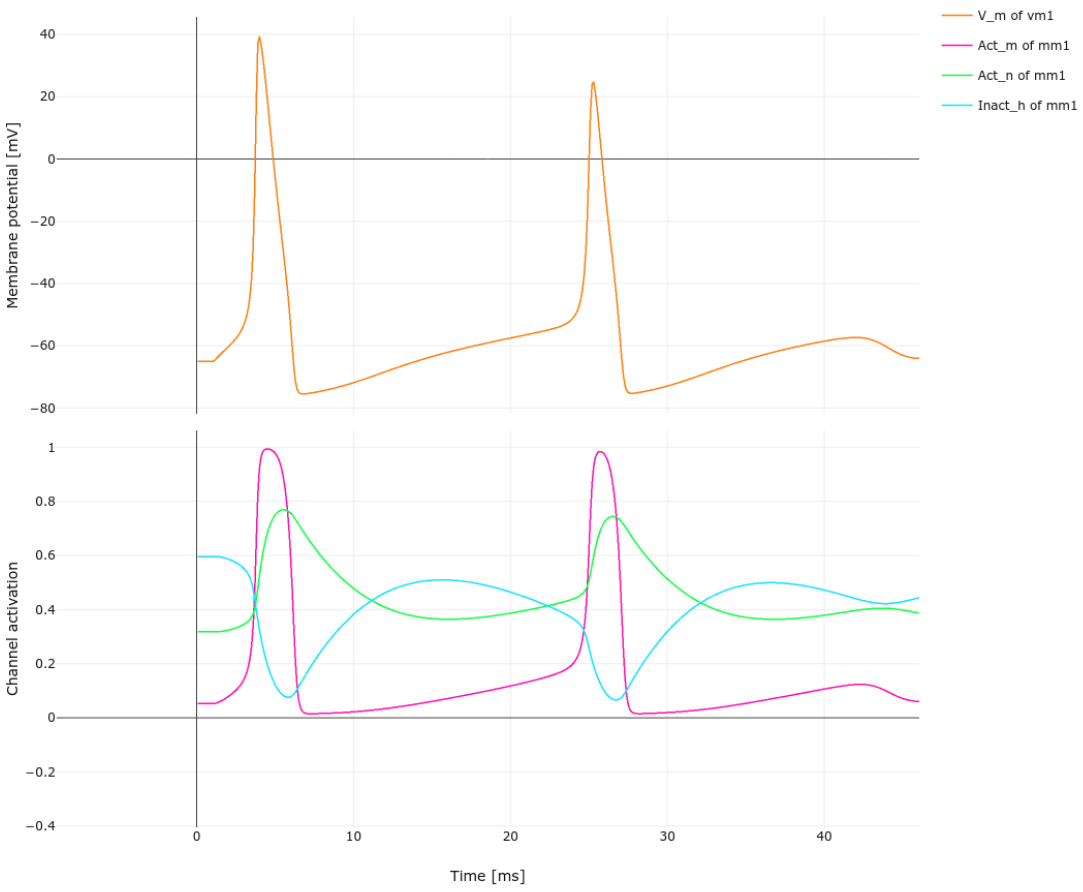


Figure 3.4: 224 pA

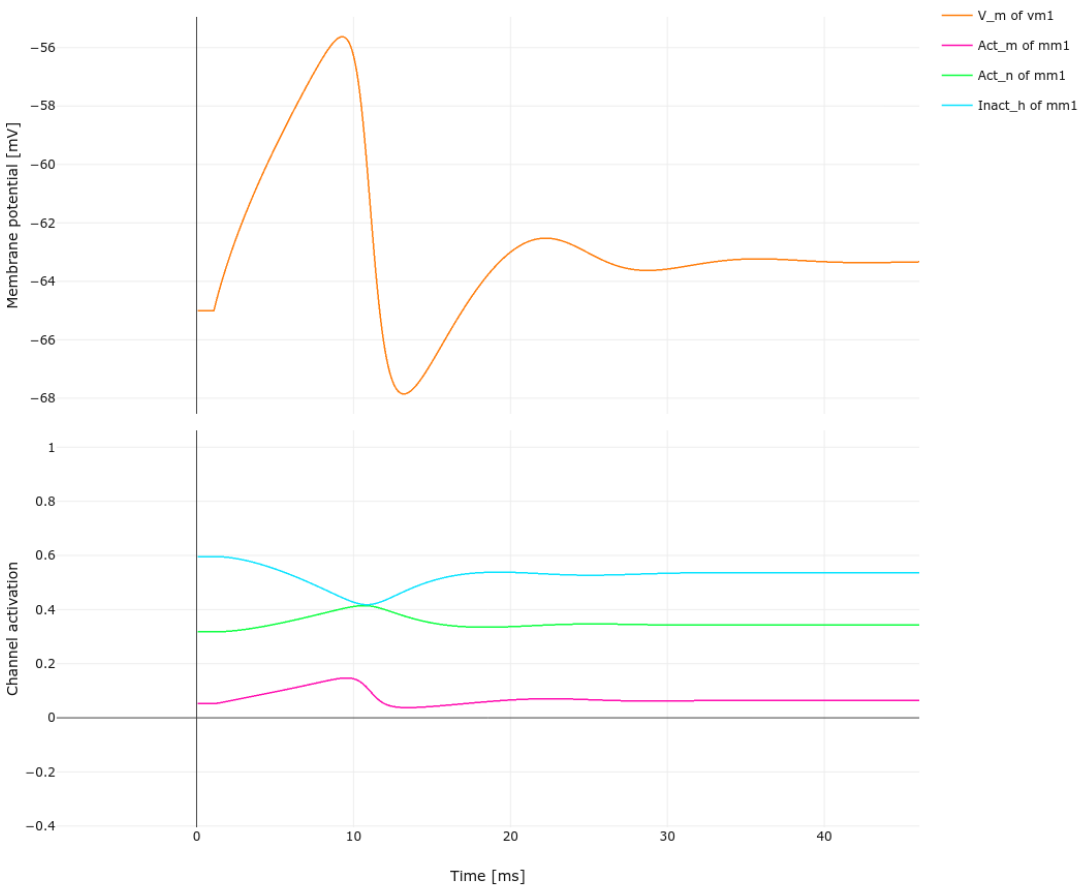
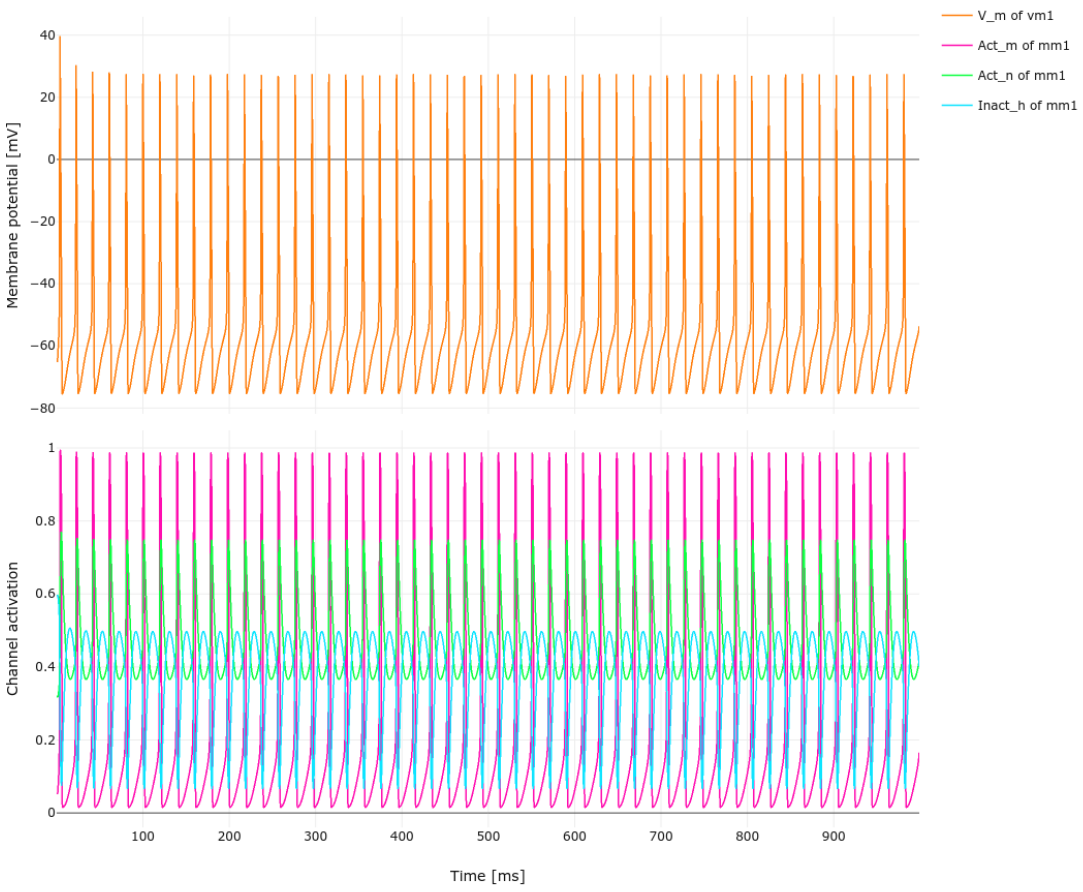


Figure 3.5: 627 pA


V_m of vm1
Act_m of mm1
Act_n of mm1
Inact_h of mm1
Membrane potential [mV]
40
20
0
−20
−40
−60
−80
Channel activation
1
0.8
0.6
0.4
0.2
0
100
200
300
400
500
600
700
800
900
Time [ms]