Project S5 - Building a single cell model from Allen Database using BMTK

Goals

1. Continuing to learn to use APIs – You have used the following APIs so far: `CellTypesApi` and `RmaApi` . In this project you will use the `BiophysicalApi` to download a biophysical NEURON model from Allen database.
2. Learning how to model single neurons using BMTK - Build a single cell model in BMTK, simulate current clamp experiment, and compare electrophysiology features with experimental data.

Introduction

The Allen Cell Types Database contains biophysical models that characterize the firing behavior of neurons measured in slices through current injection by a somatic whole-cell patch clamp electrode.
The biophysical models are run with the NEURON simulation environment. The Allen SDK package contains libraries that assist in downloading and setting up the models available on the Allen Institute web site for users to run using NEURON. The examples and scripts provided run **on Linux** using the bash shell.

Some useful links to the Allen website:
  Introduction to Biophysical Models (https://allensdk.readthedocs.io/en/latest/biophysical_models.html)
  Cell Type Database (http://celltypes.brain-map.org/)
  Example jupyter notebook - Stimulating a biophysical model with a square pulse (https://allensdk.readthedocs.io/en/latest/_static/examples/nb/pulse_stimulus.html)

Procedure

1. Download a biophysical NEURON model.

There are two ways to download files necessary to run a biophysical model. The first way is to visit **Cell Type Database (http://celltypes.brain-map.org/)** and find cells that have biophysical models available for download (Select from the Venn diagrams). The electrophysiology page of a cell has a neuronal model download link. Specifically:

  Click 'Electrophysiology' of a cell
  Click 'Select neuronal model'
  Check Models -> 'Biophysical - perisomatic' or 'Biophysical - all active'
  Scroll down and click the 'Biophysical - perisomatic' or 'Biophysical - all active' 'Download model' link.

The second way is to programmatically download it using API. The neuronal model id can be found to the left of the corresponding 'Biophysical - perisomatic' or 'Biophysical - all active' 'Download model' link.
We will adopt the second way to download the files using the following codes adapted from the example juptyer notebook given in the links.

Download model files according to the **model id** which is not the same as **specimen id**,

In [1]:
```
from allensdk.api.queries.biophysical_api import BiophysicalApi

neuronal_model_id = 472451419    # get this from the web site
model_directory = './source/'    # the files will be downloaded to the 'source'

bp = BiophysicalApi('http://api.brain-map.org')
bp.cache_stimulus = False # don't want to download the large stimulus NWB file
bp.cache_data(neuronal_model_id, working_directory=model_directory)
```

```
2020-05-05 03:22:15,719 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/49145917
3 (http://api.brain-map.org/api/v2/well_known_file_download/491459173)
2020-05-05 03:22:16,003 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/49660710
3 (http://api.brain-map.org/api/v2/well_known_file_download/496607103)
2020-05-05 03:22:16,103 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/39533729
3 (http://api.brain-map.org/api/v2/well_known_file_download/395337293)
2020-05-05 03:22:16,208 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/39533705
4 (http://api.brain-map.org/api/v2/well_known_file_download/395337054)
2020-05-05 03:22:16,308 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/39533722
5 (http://api.brain-map.org/api/v2/well_known_file_download/395337225)
2020-05-05 03:22:16,410 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/39533701
9 (http://api.brain-map.org/api/v2/well_known_file_download/395337019)
2020-05-05 03:22:16,513 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/39533700
3 (http://api.brain-map.org/api/v2/well_known_file_download/395337003)
2020-05-05 03:22:16,614 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/39533705
0 (http://api.brain-map.org/api/v2/well_known_file_download/395337050)
2020-05-05 03:22:16,716 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/39533704
2 (http://api.brain-map.org/api/v2/well_known_file_download/395337042)
2020-05-05 03:22:16,816 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/39533701
1 (http://api.brain-map.org/api/v2/well_known_file_download/395337011)
2020-05-05 03:22:16,917 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/39533704
6 (http://api.brain-map.org/api/v2/well_known_file_download/395337046)
2020-05-05 03:22:17,019 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/39533701
5 (http://api.brain-map.org/api/v2/well_known_file_download/395337015)
2020-05-05 03:22:17,119 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/39533706
6 (http://api.brain-map.org/api/v2/well_known_file_download/395337066)
2020-05-05 03:22:17,219 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/46413809
6 (http://api.brain-map.org/api/v2/well_known_file_download/464138096)
2020-05-05 03:22:17,320 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/39533700
7 (http://api.brain-map.org/api/v2/well_known_file_download/395337007)
2020-05-05 03:22:17,421 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/39533706
2 (http://api.brain-map.org/api/v2/well_known_file_download/395337062)
2020-05-05 03:22:17,523 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/49111342
5 (http://api.brain-map.org/api/v2/well_known_file_download/491113425)
2020-05-05 03:22:17,625 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/39533707
0 (http://api.brain-map.org/api/v2/well_known_file_download/395337070)
2020-05-05 03:22:17,728 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/49723580
5 (http://api.brain-map.org/api/v2/well_known_file_download/497235805)
```

2. Check cell information and electrophysiology features.

After downloading is done, the **specimen id** can be found in the file name of xxxxxx_fit.json which contains biophysical model parameters.
For this example, it is 386049446. Use the API to access the Cell Type Database to see the cell information and its electrophysiology features from biological recordings as you did in Project S4-2.

In [2]:
```python
from allensdk.api.queries.cell_types_api import CellTypesApi
cta = CellTypesApi() # the CellTypesApi instance
cell = cta.get_cell(386049446)
cell
```

Out[2]:
```
{'cell_reporter_status': 'positive',
 'csl__normalized_depth': 0.369535013823174,
 'csl__x': 8627.12987123298,
 'csl__y': 857.28871673753,
 'csl__z': 7777.58376005663,
 'donor__age': '',
 'donor__disease_state': '',
 'donor__id': 339692362,
 'donor__name': 'Nr5a1-Cre;Ai14(IVSCC)-177334',
 'donor__race': '',
 'donor__sex': '',
 'donor__species': 'Mus musculus',
 'donor__years_of_seizure_history': '',
 'ef__adaptation': 0.0450018072271697,
 'ef__avg_firing_rate': 11.0791048083315,
 'ef__avg_isi': 90.26,
 'ef__f_i_curve_slope': 0.172321437682424,
 'ef__fast_trough_v_long_square': -53.03125,
 'ef__peak_t_ramp': 6.25953666666667,
 'ef__ri': 123.281250672546,
```

In [3]:
```
from allensdk.api.queries.rma_api import RmaApi
rma = RmaApi() # the RmaApi instance
data = rma.model_query(model='EphysFeature',criteria='[specimen_id$eq386049446]
data
```

Out[3]:
```
{'adaptation': 0.0450018072271697,
 'avg_isi': 90.26,
 'electrode_0_pa': 22.9562497483515,
 'f_i_curve_slope': 0.172321437682424,
 'fast_trough_t_long_square': 1.12969,
 'fast_trough_t_ramp': 6.261465,
 'fast_trough_t_short_square': 1.02522375,
 'fast_trough_v_long_square': -53.03125,
 'fast_trough_v_ramp': -52.0625012715658,
 'fast_trough_v_short_square': -52.9687519073486,
 'has_burst': False,
 'has_delay': False,
 'has_pause': False,
 'id': 396494722,
 'input_resistance_mohm': 176.9292,
 'latency': 0.04582,
 'peak_t_long_square': 1.12778,
 'peak_t_ramp': 6.25953666666667,
 'peak_t_short_square': 1.02349125,
 'peak_v_long_square': 33.15625,
 'peak_v_ramp': 41.9583358764648,
 'peak_v_short_square': 35.9218759536743,
 'rheobase_sweep_id': 396429054,
 'rheobase_sweep_number': 44,
 'ri': 123.281359672546,
 'sag': 0.0316939763724804,
 'seal_gohm': 3.10554368,
 'slow_trough_t_long_square': 1.175875,
 'slow_trough_t_ramp': 6.29017,
 'slow_trough_t_short_square': 1.72551,
 'slow_trough_v_long_square': -59.09375,
 'slow_trough_v_ramp': -58.1354179382324,
 'slow_trough_v_short_square': -84.7109375,
 'specimen_id': 386049446,
 'tau': 12.7026528308998,
 'threshold_i_long_square': 110.0,
 'threshold_i_ramp': 130.958333333333,
 'threshold_i_short_square': 840.0,
 'threshold_t_long_square': 1.12727,
 'threshold_t_ramp': 6.25906,
 'threshold_t_short_square': 1.0230625,
 'threshold_v_long_square': -45.28125,
 'threshold_v_ramp': -40.1458333333333,
 'threshold_v_short_square': -46.2031259536743,
 'thumbnail_sweep_id': 396429034,
 'trough_t_long_square': 1.175875,
 'trough_t_ramp': 6.29017,
 'trough_t_short_square': 1.72551,
 'trough_v_long_square': -59.09375,
 'trough_v_ramp': -58.1354179382324,
 'trough_v_short_square': -84.7109375,
 'upstroke_downstroke_ratio_long_square': 3.52442479462201,
 'upstroke_downstroke_ratio_ramp': 3.70285736190798,
 'upstroke_downstroke_ratio_short_square': 3.54903700402711,
 'vm_for_sag': -99.6562576293945,
 'vrest': -85.1570739746094}
```

3. Simulate in NEURON (from the example notebook)

The following part runs a simulation with a current clamp using NEURON.

In [4]:
```
# based on allensdk.model.biophysical.biophysical_perisomatic.runner

# These will be useful for accessing and configuring the downloaded model
from allensdk.model.biophys_sim.config import Config
from allensdk.model.biophysical.utils import Utils

# not using NwbDataSet
# from allensdk.core.nwb_data_set import NwbDataSet

# We'll save results to a simple text file instead
from allensdk.core.dat_utilities import DatUtilities

import os
cwd = os.getcwd()
```

Compile modfiles in 'source' folder. If this does not work in Windows, manually compile the 'modfiles' folder and move the 'nrnmech.dll' file to the 'source' folder.

In [5]:
```
os.chdir('source')
print(os.system('nrnivmodl modfiles')) # compile modfiles. Return 0 for success
os.chdir(cwd)
```
```
0
```

Set up model configurations.

In [6]:
```
os.chdir('source')

description = Config().load('manifest.json')
utils = Utils(description)
h = utils.h # NEURON handle

# configure model
manifest = description.manifest
morphology_path = description.manifest.get_path('MORPHOLOGY')
utils.generate_morphology(morphology_path.encode('ascii', 'ignore'))
utils.load_cell_parameters()

os.chdir(cwd)
```

At this point the cell model has been fully set up in NEURON.

Configure a simple current-clamp stimulus to generate some spikes.

In [7]:
```
stim = h.IClamp(h.soma[0](0.5))
stim.amp = 0.18  # nA
stim.delay = 500.0
stim.dur = 1000.0

h.tstop = 2000.0

vec = utils.record_values()
```

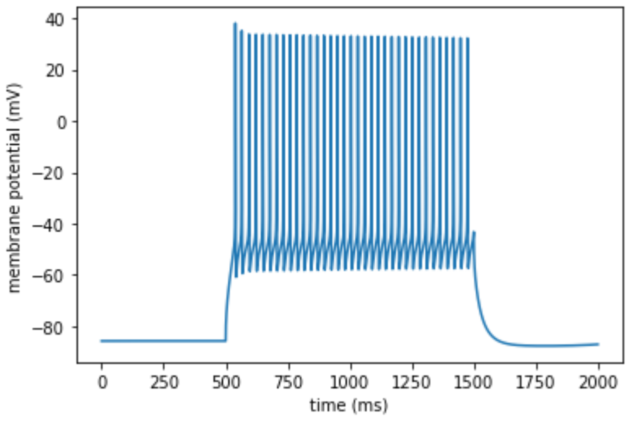
In [8]:
```
h.finitialize()
h.run()
```
Out[8]:
```
0.0
```

Save the result to a simple time and voltage space-separated text file.

In [9]:
```
import numpy

output_path = './source/output_voltage.dat'

junction_potential = description.data['fitting'][0]['junction_potential']
mV = 1.0e-3
ms = 1.0e-3

output_data = (numpy.array(vec['v']) - junction_potential) * mV
output_times = numpy.array(vec['t']) * ms

data = numpy.transpose(numpy.vstack((output_times, output_data)))
with open (output_path, "w") as f:
    numpy.savetxt(f, data)
```

Plot membrane voltage trace.

In [10]:
```
%matplotlib inline
import matplotlib.pyplot as plt
plt.plot(vec['t'], numpy.array(vec['v']) - junction_potential)
plt.xlabel('time (ms)')
plt.ylabel('membrane potential (mV)')
plt.show()
```



4. Build a single cell model in BMTK using downloaded files.

Follow the BMTK tutorial 01 and use the files downloaded in the previous notebook to build a single cell model. If you need more instructions on how to build a model, try out the notebook `01_single_cell_clamped_S5.ipynb` adapted from the BMTK tutorial 01 for building the model of the example cell downloaded in this notebook.

Details of Tasks

1. Download a perisomatic model with **model_id = 485591806** or choose another cell that has biophysical model available from the Allen Cell Type database website using this notebook as example.
2. Then build the single cell model with current clamp using BMTK.
3. Simulate with different current amplitudes using the model you build and find out the threshold for the neuron to fire. Compare the threshold you find with `threshold_i_long_square` (unit: pA) given in the electrophysiology features from the biological recordings.
4. You can also compare other features if you are interested (Optional). Check **electrophysiology overview technical whitepaper (http://help.brain-map.org/download/attachments/8323525/CellTypes_Ephys_Overview.pdf)** to see how the electrophysiology features are quantified.
5. Finally, in your own words, describe what you learned from this project. Also, comment on what can be improved in the project.

In [1]:
```
from allensdk.api.queries.biophysical_api import BiophysicalApi

neuronal_model_id = 485591806    # get this from the web site
model_directory = './source/'    # the files will be downloaded to the 'source'

bp = BiophysicalApi('http://api.brain-map.org')
bp.cache_stimulus = False # don't want to download the large stimulus NWB file
bp.cache_data(neuronal_model_id, working_directory=model_directory)
```

```
2020-05-05 19:49:11,522 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/49210109
5 (http://api.brain-map.org/api/v2/well_known_file_download/492101095)
2020-05-05 19:49:11,825 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/49660515
1 (http://api.brain-map.org/api/v2/well_known_file_download/496605151)
2020-05-05 19:49:11,928 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/39533729
3 (http://api.brain-map.org/api/v2/well_known_file_download/395337293)
2020-05-05 19:49:12,048 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/39533705
4 (http://api.brain-map.org/api/v2/well_known_file_download/395337054)
2020-05-05 19:49:12,151 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/39533722
5 (http://api.brain-map.org/api/v2/well_known_file_download/395337225)
2020-05-05 19:49:12,251 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/39533701
9 (http://api.brain-map.org/api/v2/well_known_file_download/395337019)
2020-05-05 19:49:12,353 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/39533700
3 (http://api.brain-map.org/api/v2/well_known_file_download/395337003)
2020-05-05 19:49:12,454 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/39533705
0 (http://api.brain-map.org/api/v2/well_known_file_download/395337050)
2020-05-05 19:49:12,555 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/39533704
2 (http://api.brain-map.org/api/v2/well_known_file_download/395337042)
2020-05-05 19:49:12,655 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/39533701
1 (http://api.brain-map.org/api/v2/well_known_file_download/395337011)
2020-05-05 19:49:12,756 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/39533704
6 (http://api.brain-map.org/api/v2/well_known_file_download/395337046)
2020-05-05 19:49:12,857 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/39533701
5 (http://api.brain-map.org/api/v2/well_known_file_download/395337015)
2020-05-05 19:49:12,958 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/39533706
6 (http://api.brain-map.org/api/v2/well_known_file_download/395337066)
2020-05-05 19:49:13,114 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/46413809
6 (http://api.brain-map.org/api/v2/well_known_file_download/464138096)
2020-05-05 19:49:13,217 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/39533700
7 (http://api.brain-map.org/api/v2/well_known_file_download/395337007)
2020-05-05 19:49:13,318 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/39533706
2 (http://api.brain-map.org/api/v2/well_known_file_download/395337062)
2020-05-05 19:49:13,420 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/49111342
5 (http://api.brain-map.org/api/v2/well_known_file_download/491113425)
2020-05-05 19:49:13,523 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/39533707
0 (http://api.brain-map.org/api/v2/well_known_file_download/395337070)
2020-05-05 19:49:13,625 allensdk.api.api.retrieve_file_over_http INFO     Down
loading URL: http://api.brain-map.org/api/v2/well_known_file_download/49723712
6 (http://api.brain-map.org/api/v2/well_known_file_download/497237126)
```

In [2]:
```
from bmtk.builder.networks import NetworkBuilder

net = NetworkBuilder('Nr5a1-Cre')
net.add_nodes(cell_name='484559000',
              potental='exc',
              model_type='biophysical',
              model_template='ctdb:Biophys1.hoc',
              model_processing='aibs_perisomatic',
              dynamics_params='484559000_fit.json',
              morphology='Nr5a1-Cre_Ai14-177334.05.01.01_491459171_m.swc')
```

In [3]:
```
net.build()
net.save_nodes(output_dir='network')
```

In [4]:
```
for node in net.nodes():
    print(node)
```

```
{'cell_name': '484559000', 'potental': 'exc', 'model_type': 'biophysical', 'mo
del_template': 'ctdb:Biophys1.hoc', 'model_processing': 'aibs_perisomatic', 'd
ynamics_params': '484559000_fit.json', 'morphology': 'Nr5a1-Cre_Ai14-177334.0
5.01.01_491459171_m.swc', 'node_type_id': 100, 'node_id': 0}
```

In [5]:
```
from bmtk.utils.sim_setup import build_env_bionet

build_env_bionet(base_dir='Proj_S5',      # Where to save the scripts and confi
                 network_dir='network',    # Location of directory containing n
                 tstop=2000.0, dt=0.025,     # Run a simulation for 2000 ms at
                 report_vars=['v', 'cai'], # Tells simulator we want to record
                 current_clamp={           # Creates a step current from 500.ms
                     'amp': 0.090,
                     'delay': 500.0,
                     'duration': 1000.0
                 },
                 include_examples=True,    # Copies components files
                 compile_mechanisms=True   # Will try to compile NEURON mechani
                )
```

In [6]:
```
%%bash
cp source/484559000_fit.json Proj_S5/components/biophysical_neuron_models
cp source/Nr5a1-Cre_Ai14-177334.05.01.01_491459171_m.swc Proj_S5/components/mor
```

In [7]:
```python
from bmtk.simulator import bionet

conf = bionet.Config.from_json('Proj_S5/simulation_config.json')
conf.build_env()
net = bionet.BioNetwork.from_config(conf)
sim = bionet.BioSimulator.from_config(conf, network=net)
sim.run()
```

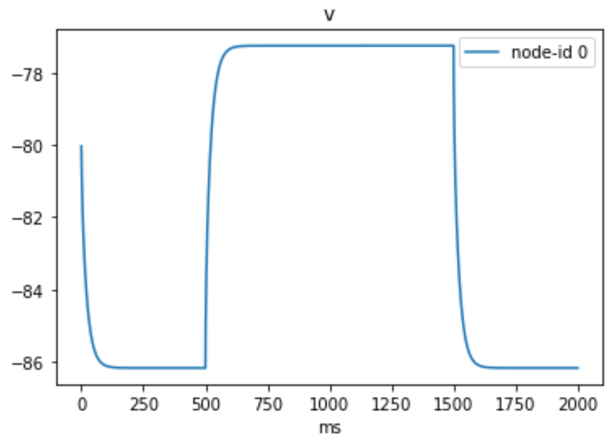
```
2020-05-05 19:49:15,661 [INFO] Created log file

INFO:NEURONIOUtils:Created log file

2020-05-05 19:49:15,729 [INFO] Building cells.

INFO:NEURONIOUtils:Building cells.

2020-05-05 19:49:15,875 [INFO] Building recurrent connections

INFO:NEURONIOUtils:Building recurrent connections

2020-05-05 19:49:15,889 [INFO] Running simulation for 2000.000 ms with the tim
e step 0.025 ms

INFO:NEURONIOUtils:Running simulation for 2000.000 ms with the time step 0.025
ms

2020-05-05 19:49:15,891 [INFO] Starting timestep: 0 at t_sim: 0.000 ms

INFO:NEURONIOUtils:Starting timestep: 0 at t_sim: 0.000 ms

2020-05-05 19:49:15,892 [INFO] Block save every 5000 steps

INFO:NEURONIOUtils:Block save every 5000 steps

2020-05-05 19:49:16,181 [INFO]     step:5000 t_sim:125.00 ms

INFO:NEURONIOUtils:    step:5000 t_sim:125.00 ms

2020-05-05 19:49:16,459 [INFO]     step:10000 t_sim:250.00 ms

INFO:NEURONIOUtils:    step:10000 t_sim:250.00 ms

2020-05-05 19:49:16,747 [INFO]     step:15000 t_sim:375.00 ms

INFO:NEURONIOUtils:    step:15000 t_sim:375.00 ms

2020-05-05 19:49:17,056 [INFO]     step:20000 t_sim:500.00 ms

INFO:NEURONIOUtils:    step:20000 t_sim:500.00 ms

2020-05-05 19:49:17,323 [INFO]     step:25000 t_sim:625.00 ms

INFO:NEURONIOUtils:    step:25000 t_sim:625.00 ms

2020-05-05 19:49:17,591 [INFO]     step:30000 t_sim:750.00 ms

INFO:NEURONIOUtils:    step:30000 t_sim:750.00 ms

2020-05-05 19:49:17,859 [INFO]     step:35000 t_sim:875.00 ms

INFO:NEURONIOUtils:    step:35000 t_sim:875.00 ms

2020-05-05 19:49:18,135 [INFO]     step:40000 t_sim:1000.00 ms

INFO:NEURONIOUtils:    step:40000 t_sim:1000.00 ms

2020-05-05 19:49:18,421 [INFO]     step:45000 t_sim:1125.00 ms

INFO:NEURONIOUtils:    step:45000 t_sim:1125.00 ms

2020-05-05 19:49:18,698 [INFO]     step:50000 t_sim:1250.00 ms

INFO:NEURONIOUtils:    step:50000 t_sim:1250.00 ms

2020-05-05 19:49:18,949 [INFO]     step:55000 t_sim:1375.00 ms
```

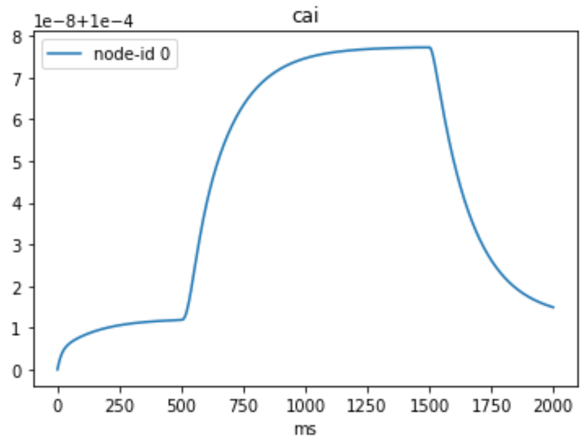
In [8]:
```
from bmtk.analyzer.spike_trains import to_dataframe
to_dataframe(config_file='Proj_S5/simulation_config.json')
```

Out[8]:

timestamps	population	node_ids

In [9]:
```
%matplotlib inline
from bmtk.analyzer.cell_vars import plot_report

plot_report(config_file='Proj_S5/simulation_config.json')
```

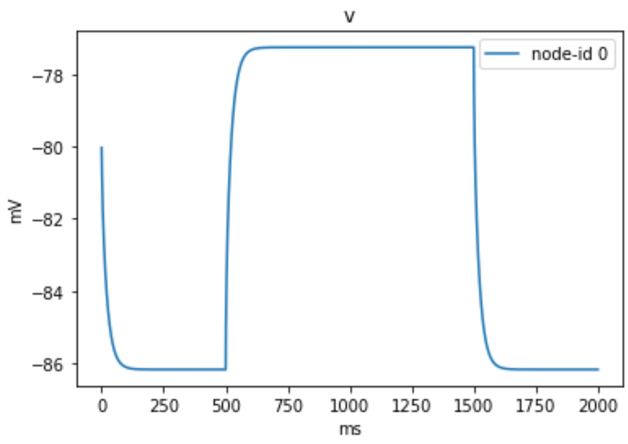




In [10]:
```
import h5py
import numpy as np

# get recorded variable from h5 file into numpy array
f = h5py.File('Proj_S5/output/v_report.h5','r') # load data file of membrane vo
f.visit(print) # check groups

print(f['report/Nr5a1-Cre/mapping/time'][()]) # time (start, stop, step size)
t = np.arange(*f['report/Nr5a1-Cre/mapping/time']) # create a numpy array of ti
v = f['report/Nr5a1-Cre/data'][()] # get membrane voltage data to a numpy array
node_id = f['report/Nr5a1-Cre/mapping/node_ids'][0] # node id (cell id is 0 sin
```

```
report
report/Nr5a1-Cre
report/Nr5a1-Cre/data
report/Nr5a1-Cre/mapping
report/Nr5a1-Cre/mapping/element_ids
report/Nr5a1-Cre/mapping/element_pos
report/Nr5a1-Cre/mapping/index_pointer
report/Nr5a1-Cre/mapping/node_ids
report/Nr5a1-Cre/mapping/time
[0.0e+00 2.0e+03 2.5e-02]
```

In [11]:
```
%matplotlib inline
import matplotlib.pyplot as plt

# plot the membrane voltage
plt.figure()
plt.plot(t,v)
plt.xlabel('ms')
plt.ylabel('mV')
plt.title('v')
plt.legend(['node-id '+str(node_id)])
plt.show()
```



In [12]:
```
import pandas as pd
pd.read_csv('network/Nr5a1-Cre_node_types.csv', sep=' ')
```

Out[12]:

	node_type_id	cell_name	morphology	model_type	model_tem
0	100	484559000	Nr5a1-Cre_Ai14-177334.05.01.01_491459171_m.swc	biophysical	ctdb:Biophys

In [13]:
```
from allensdk.api.queries.rma_api import RmaApi
rma = RmaApi() # the RmaApi instance
data = rma.model_query(model='EphysFeature',criteria='[specimen_id$eq484559000]
data
```

Out[13]:
```
{'adaptation': 0.0081539780749065,
 'avg_isi': 78.9129166666667,
 'electrode_0_pa': 21.9081246199959,
 'f_i_curve_slope': 0.148720608859549,
 'fast_trough_t_long_square': 1.109125,
 'fast_trough_t_ramp': 4.86082833333333,
 'fast_trough_t_short_square': 1.024905,
 'fast_trough_v_long_square': -45.9062538146973,
 'fast_trough_v_ramp': -49.7500025431315,
 'fast_trough_v_short_square': -53.3062522888184,
 'has_burst': False,
 'has_delay': False,
 'has_pause': False,
 'id': 484579736,
 'input_resistance_mohm': 113.40336,
 'latency': 0.0446200000000001,
 'peak_t_long_square': 1.107065,
 'peak_t_ramp': 4.85904666666667,
 'peak_t_short_square': 1.023385,
 'peak_v_long_square': 29.5000019073486,
 'peak_v_ramp': 31.9687506357829,
 'peak_v_short_square': 32.2625011444092,
 'rheobase_sweep_id': 484564045,
 'rheobase_sweep_number': 42,
 'ri': 118.43753606081,
 'sag': 0.103550091385841,
 'seal_gohm': 2.254352896,
 'slow_trough_t_long_square': 1.13875,
 'slow_trough_t_ramp': 4.896285,
 'slow_trough_t_short_square': 1.194107,
 'slow_trough_v_long_square': -55.7812538146973,
 'slow_trough_v_ramp': -58.4479192097982,
 'slow_trough_v_short_square': -72.9062545776367,
 'specimen_id': 484559000,
 'tau': 12.5565277603897,
 'threshold_i_long_square': 170.0,
 'threshold_i_ramp': 96.0,
 'threshold_i_short_square': 949.999938964844,
 'threshold_t_long_square': 1.10666,
 'threshold_t_ramp': 4.85863166666667,
 'threshold_t_short_square': 1.023036,
 'threshold_v_long_square': -40.0625038146973,
 'threshold_v_ramp': -41.0833358764648,
 'threshold_v_short_square': -52.2500022888184,
 'thumbnail_sweep_id': 484564049,
 'trough_t_long_square': 1.13875,
 'trough_t_ramp': 4.896285,
 'trough_t_short_square': 1.194107,
 'trough_v_long_square': -55.7812538146973,
 'trough_v_ramp': -58.4479192097982,
 'trough_v_short_square': -72.9062545776367,
 'upstroke_downstroke_ratio_long_square': 3.05927547343154,
 'upstroke_downstroke_ratio_ramp': 2.88852425505902,
 'upstroke_downstroke_ratio_short_square': 2.95132283616187,
 'vm_for_sag': -83.21875,
 'vrest': -71.9118957519531}
```

In [14]:

In [ ]:

```
In [ ]:
```