* 1. The power of the brain

1. Average energy consumption in human brain per action potential:
2. Average energy consumption in human brain per synaptic event:
3. Energy consumption per spike and synaptic event in simulation:
4. Scale up of simulations:
5. Power after speed up the K computer:

The nuclear power plant Sedai has two reactors, each with 890 MW output.

* 1. Stimulus Currents

Simplified equivalent circuit:

1. Dirac delta function:

Look for solution of homogenous equation:

:

Putting everything together yields:

1. Step current:

The homogenous solution is the same as before, so:

Putting everything together yields:

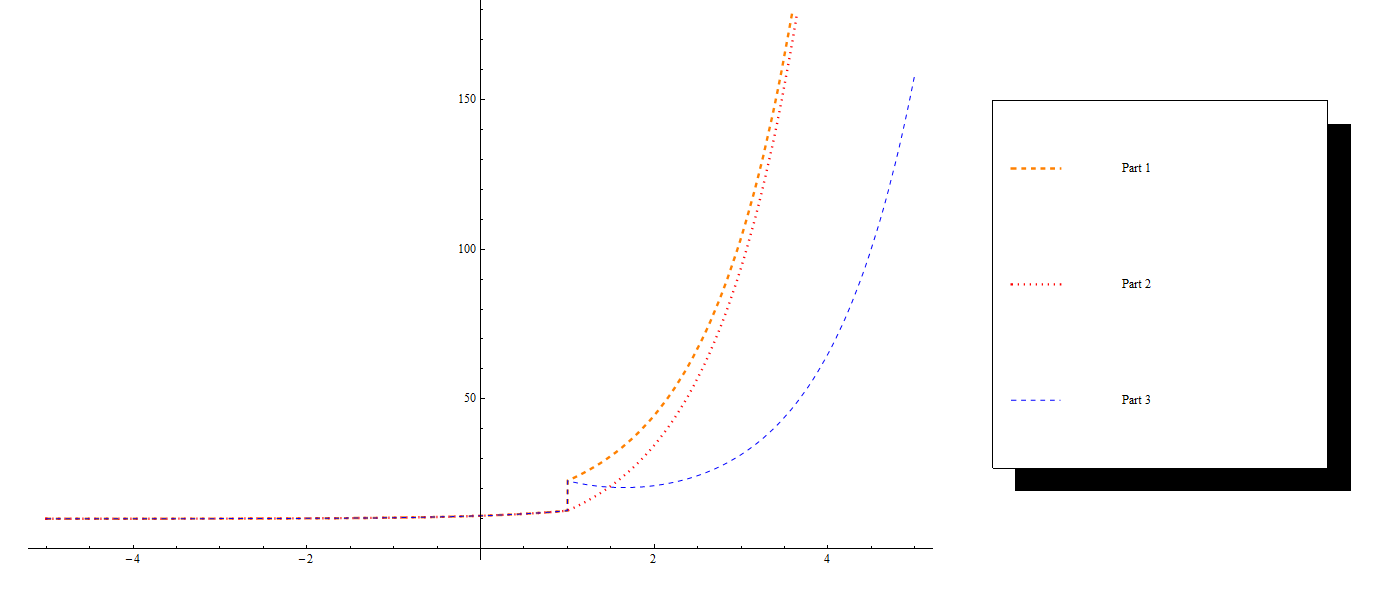


1. Exponential current:

Using the same as before:

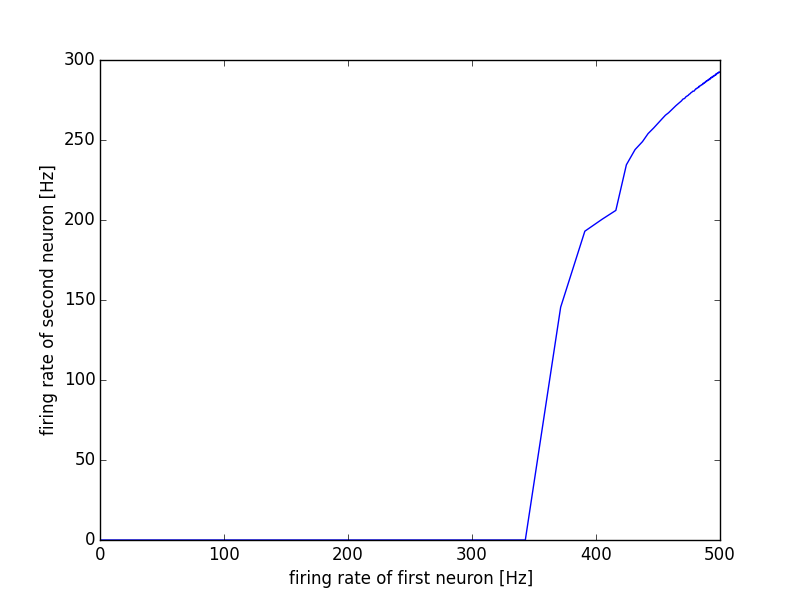
Putting everything together:

1. In the limit of very short synaptic time constants we get:

1.3 Facets Live!

Two-neuron demo: output firing rate of first neuron as a function if its input current:

Firing rate of second neuron as a function of its input firing rate:



#!/usr/bin/env python2

# encoding: utf-8

import pyNN.nest as sim

def two\_neuron\_example(

current=1000.0,

time\_simulation=2000.,

weight=0.4,

neuron\_parameters={

'v\_rest' : -65.0,

'cm' : 0.1,

'tau\_m' : 1.0,

'tau\_refrac' : 2.0,

'tau\_syn\_E' : 10.0,

'tau\_syn\_I' : 10.0,

'i\_offset' : 0.0,

'v\_reset' : -65.0,

'v\_thresh' : -50.0,

},

):

"""

Connects to neurons with corresponding parameters.

The first is stimulated via current injection while the second receives

the other one's spikes.

"""

sim.setup(timestep=0.1, min\_delay=0.1)

pulse = sim.DCSource(amplitude=current, start=0.0, stop=time\_simulation)

pre = sim.Population(1, sim.IF\_curr\_exp(\*\*neuron\_parameters))

post = sim.Population(1, sim.IF\_curr\_exp(\*\*neuron\_parameters))

pre.record('spikes')

post.record('spikes')

sim.Projection(pre, post, connector=sim.OneToOneConnector(),

synapse\_type=sim.StaticSynapse(weight=weight),

receptor\_type='excitatory')

pulse.inject\_into(pre)

sim.run(time\_simulation)

# rates in Hz

rate\_pre = len(pre.get\_data('spikes').segments[0].spiketrains[0])\

/ time\_simulation \* 1000.

rate\_post = len(post.get\_data('spikes').segments[0].spiketrains[0])\

/ time\_simulation \* 1000.

sim.end()

return rate\_pre, rate\_post

import matplotlib.pyplot as plt

import numpy as np

rate\_pre = []

rate\_post = []

current = []

for curr in np.linspace(0,500,1000):

pre, post = two\_neuron\_example(current=curr)

rate\_pre.append(pre)

rate\_post.append(post)

current.append(curr)

# Plot results

plt.figure(1)

plt.plot(current,rate\_pre)

plt.xlabel('input current')

plt.ylabel('firing rate of first neuron [Hz]')

plt.savefig('figure1.png')

plt.figure(2)

plt.plot(rate\_pre,rate\_post)

plt.xlabel('firing rate of first neuron [Hz]')

plt.ylabel('firing rate of second neuron [Hz]')

plt.savefig('figure2.png')