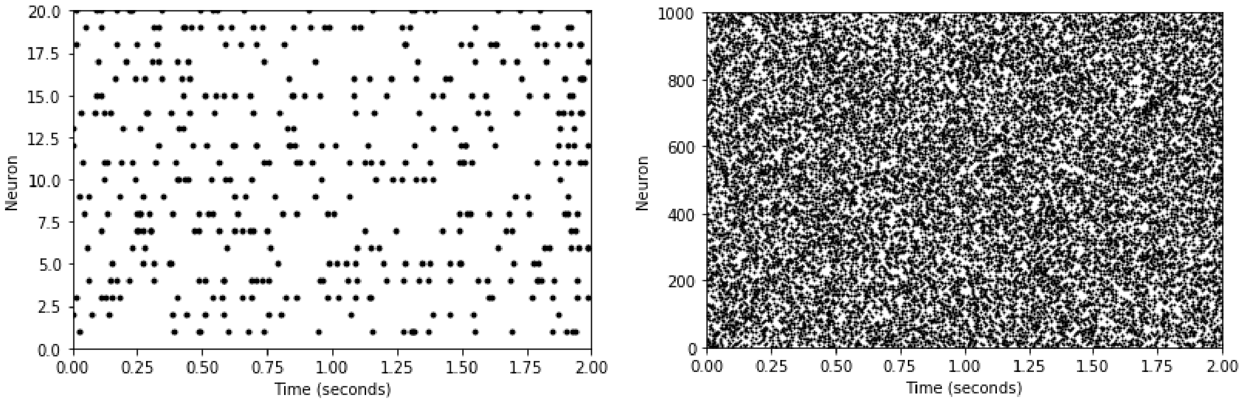
4G3 Coursework 1

1: Generating Poisson spike trains

The Poisson process was modeled as a set of Bernoulli trials over small time intervals, δt. Using this model, the probability of success of each trial is $r_x \delta_t$, where r_X was the mean of the Poisson distribution. The activity of 1000 Poisson-generated spike trains for 2 seconds is shown in the raster plot in figure 1b. The activity of each individual neuron can be seen more clearly in figure 1a, where the activity of only 20 neurons is shown. The results of figure 1 were verified by comparing observed spike counts to the theoretically expected value of 20 for the 2-second period ($2r_X$). Over 10000 neurons, the spike count over the 2 second period was observed to have a mean of 19.9845 and a variance of 0.0256. The observed mean matches the theoretically expected value to 3 significant figures, with a very low variance. The observation matches the theory.



(a) Activity of 20 neurons

(b) Activity of 1000 neurons

Figure 1: Raster plots showing the activity of 20 and 1000 neurons firing according to a Poisson distribution with a constant rate, $r_X = 20$.

```python
dur = 2.0
x_activity = generate_poisson_spikes(dur, N)
plot_spike_trains(x_activity, dur, N)

def generate_poisson_spikes(dur, num, delta = dt, rate = rX):
    return 1/dt * binomial(1, rX * delta, (num, int(dur/dt)))
```

Listing 1: Python excerpt to generate N independent Poisson trains

2: Single LIF neuron with one input spike train

The dynamics of a leaky-integrate-and-fire (LIF) neuron with a single Poisson spike train input was investigated in this part. The rate of the Poisson input was taken to be r_X, just as in Section 1. The dynamics of the neuron were governed by the following equation:

$$\frac{dV_i}{dt} = \frac{-V_i(t)}{\tau} + wS_j(t) \tag{1}$$

where the synaptic weight, w, was set to 0.9.

Upon discretization and the use of forward Euler integration, the membrane potential, V, at a time-step k, was modelled by the following:

$$\widetilde{V}(0) \leftarrow 0 \quad \text{and} \quad \widetilde{V}(k) \leftarrow \widetilde{V}(k-1) + \delta_t \left[\frac{\widetilde{V}(k-1)}{\tau} + w\widetilde{S}_j(k-1) \right] \tag{2}$$

The equation above was applied in a Python script to give the membrane potential shown at the middle of figure 2. The top and bottom plots of figure 2 show the input and output spike trains to and from the neuron respectively. It was observed that rises in the membrane potential coincide with peaks in the input spike train. Between peaks in the input spike train, the membrane potential relaxes exponentially to the baseline. When many peaks in the input spike train occur within a small time interval (rapid 'bursts') a more dramatic rise is noted in the membrane potential, since it has less time to relax to the baseline before another peak is seen in the input. This causes the membrane potential to cross the threshold $V_{th} = 1$ resulting in an action potential at the output of the neuron. It is observed that 'bursts' in the input correspond to spikes at the output.

```python
w = 0.9
Sj = generate_poisson_spikes(dur, 1)[0]
Si, V = LIF_single_input(w, Sj, dur)

def LIF_single_input(w, Sj, dur, spike_and_reset = True, delta = dt):
    V = np.zeros(int(dur/delta) + 1)
    S_out = np.zeros(len(V))
    if spike_and_reset:
        for k in range(1, len(V)):
            V[k] = V[k-1] + delta * (-V[k-1]/T + w * Sj[k-1])
            if V[k-1] > Vth:
                V[k] = 0
                S_out[k] = 1
    else:
        for k in range(1, len(V)):
            V[k] = V[k-1] + delta * (-V[k-1]/T + w * Sj[k-1])
    return S_out, V
```

Listing 2: Python excerpt to model leaky-integrate-and-fire dynamics

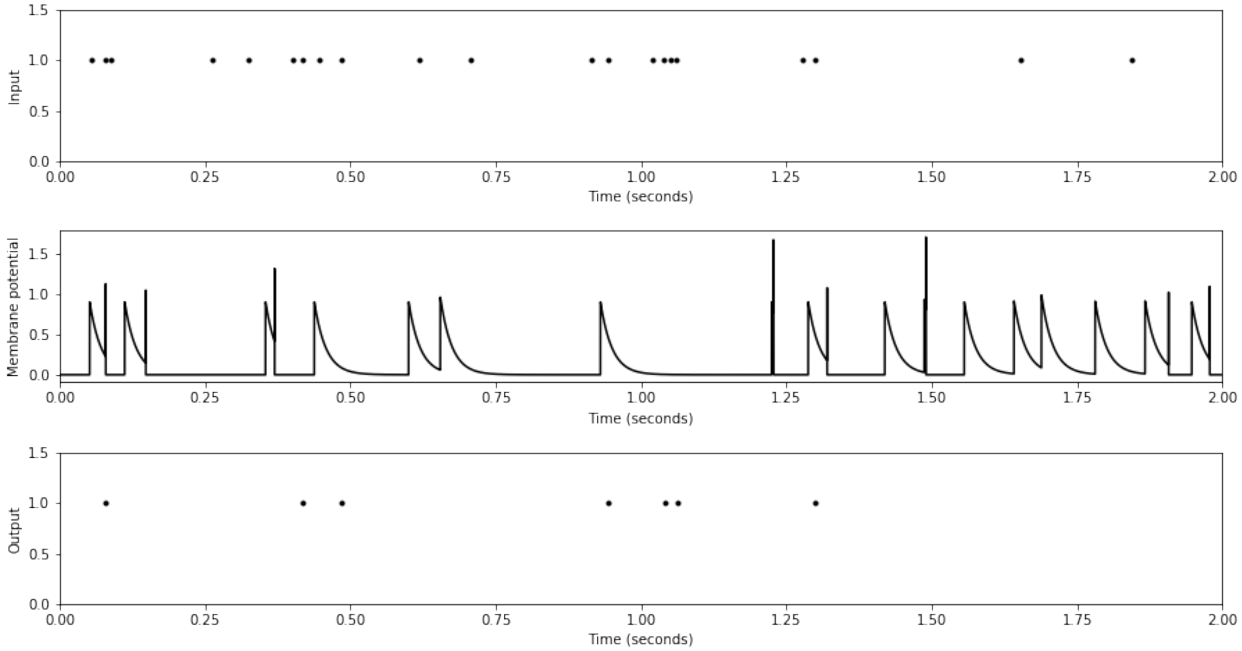


Figure 2: Activity at input and output of leaky-integrate-and-fire (LIF) neuron with a single Poisson-governed input. It is observed that the spikes at the output of the neuron coincide with rapid 'bursts' at the input of the neuron.

3: Single LIF neuron with many input spike trains

The single LIF neuron from section 2 was now adapted to have K independent Poisson inputs instead of just one. Again each input, S_j had a constant firing rate of r_X with the same weight, w. The dynamics of the neuron were therefore described by the following equation:

$$\frac{dV_i(t)}{dt} = -\frac{V_i(t)}{\tau} + \underbrace{\frac{w}{K}\sum_{j=1}^{K} S_j(t)}_{h(t)} \tag{3}$$

When discretized using forward Euler integration with a timestep of δ_t, the dynamics may also be described by:

$$\widetilde{V}(0) \leftarrow 0 \;\text{ and }\; \widetilde{V}(k) \leftarrow \widetilde{V}(k-1) + \delta_t \left[\frac{\widetilde{V}(k-1)}{\tau} + \frac{w}{K}\sum_{j=1}^{K} \widetilde{S}_j(k-1)\right] \tag{4}$$

Part A

The weight w was set to 1, and the spike-and-reset mechanism was disabled to give the plot in figure 3 for a 2-second run.

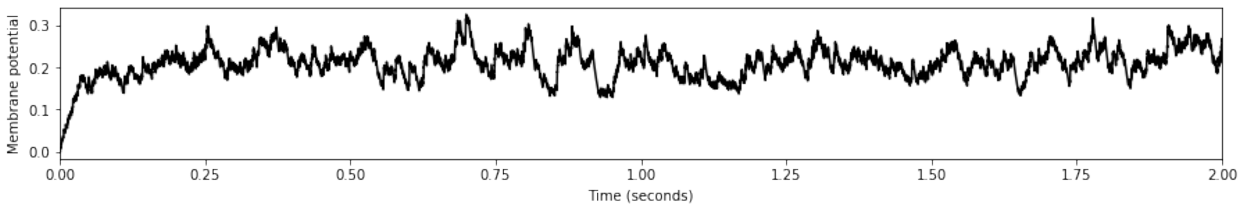


Figure 3: Membrane potential of multi-Poisson-input LIF neuron with spike-and-reset mechanism disabled. It is observed that the membrane potential does not decay to 0 though after about 100ms it fluctuates about a stationary value with mean $\tau w r_X$ where the membrane time constant, $\tau = 20$ms.

Without the spike-and-reset mechanism it is observed that the membrane potential rises but does not then decay back to the baseline. Instead it fluctuates around a mean value of $\tau w r_X$ with a variance of $\frac{w^2 r_X \tau}{2K}$ as shown in section 3 part B.

Part B

It was observed that the membrane potential in figure 3 fluctuated around a stationary value. The mean and variance of these fluctuations are derived in this section. First we consider the mean and variance of the *fluctuations* in $h(t)$ as defined in equation (3).

$$\begin{aligned}
\mathbb{E}\left[\widetilde{h}(k)\right] = \mathbb{E}\left[\frac{w}{K}\sum_{j=1}^{K}\widetilde{S}_j(k)\right] &= \frac{w}{K}\,\mathbb{E}\left[\sum_{j=1}^{K}\widetilde{S}_j(k)\right] \\
&= \frac{w}{K}\sum_{j=1}^{K}\mathbb{E}\left[\widetilde{S}_j(k)\right] \\
&= \frac{w}{K}\sum_{j=1}^{K}\frac{1}{\delta_t} r_X \delta_t \\
&= \frac{w}{K}\sum_{j=1}^{K} r_X \\
&= w r_X
\end{aligned} \tag{5}$$

where we have used the fact that the spike-train can be modelled as a Bernoulli distribution with success probability $r_X \delta_t$ at any given time-step k. Likewise:

$$\mathbf{Var}[\widetilde{h}(k)] = \frac{w^2}{K^2}\,\mathbf{Var}\left[\sum_{i=1}^{K}\widetilde{S}_j(k)\right] = \frac{w^2}{K^2}\sum_{i=1}^{K}\mathbf{Var}\left[\widetilde{S}_j(k)\right] \tag{6}$$

By using the same Bernoulli argument, the following expresions may be obtained:

$$\mathbb{E}\left[\widetilde{S}_i(k)\widetilde{S}_j(k')\right] = \begin{cases} (1/\delta_t)^2 \cdot (r_X\delta_t)^2 = r_X^2 & i = j \text{ and } k \neq k' \\ (1/\delta_t)^2 \cdot (r_X\delta_t) = r_X/\delta_t & i = j \text{ and } k = k' \\ (1/\delta_t)^2 \cdot (r_X\delta_t)^2 = r_X^2 & i \neq j \text{ and } k \neq k' \\ (1/\delta_t)^2 \cdot (r_X\delta_t)^2 = r_X^2 & i \neq j \text{ and } k = k' \end{cases} \tag{7}$$

For $\mathbf{Var}[\widetilde{h}(k)]$ we are interested in the case, $i = j$ and $k = k'$ i.e.

$$\mathbb{E}\left[\widetilde{S}_i(k)^2\right] = \frac{r_X}{\delta_t} \tag{8}$$

Hence

$$\begin{aligned} \mathbf{Var}\left[\widetilde{S}_i(k)\right] &= \mathbb{E}\left[\widetilde{S}_i(k)^2\right] - \mathbb{E}\left[\widetilde{S}_j(k)\right]^2 \\ &= r_X\left(\frac{1}{\delta_t} - r_X\right) \end{aligned} \tag{9}$$

Thus

$$\mathbf{Var}\left[\widetilde{h}(k)\right] = \frac{w^2 r_X}{K}\left(\frac{1}{\delta_t} - r_X\right) \tag{10}$$

Using equation (4) and noting that the mean is constant at stationarity:

$$\begin{aligned} \mu &= \mathbb{E}\left[V(k)\right] = \mathbb{E}\left[\widetilde{V}(k-1)\right] \\ \mu_h &= \mathbb{E}\left[h(k)\right] = \mathbb{E}\left[\widetilde{h}(k-1)\right] \end{aligned} \tag{11}$$

Therefore:

$$\begin{aligned} \mu &= \mu + \delta t\left[-\frac{\mu}{\tau} + \mu_h\right] \\ \mu &= \tau\mu_h = \tau w r_X \end{aligned} \tag{12}$$

Using equation (4) and noting that the variance is constant at stationarity, the variance can be calculated similarly:

$$\begin{aligned}
\mathbf{Var}\left[\widetilde{V}(k)\right] &= \mathbf{Var}\left[\widetilde{V}(k-1) + \delta_t\left[-\frac{\widetilde{V}(k-1)}{\tau} + \frac{w}{K}\sum_{j=1}^{K}\widetilde{S}_j(k-1)\right]\right] \\
\sigma^2 = \mathbf{Var}\left[\widetilde{V}(k)\right] &= \left[1-\frac{\delta_t}{\tau}\right]^2 \mathbf{Var}\left[\widetilde{V}(k-1)\right] + \delta_t^2\,\mathbf{Var}[h] \\
\sigma^2 &= \left[1-\frac{\delta_t}{\tau}\right]^2 \sigma^2 + \delta_t^2\sigma_h^2 \\
\sigma^2 &= \frac{\tau^2\delta_t\sigma_h^2}{2\tau-\delta_t} \\
&= \frac{\tau^2\delta_t}{2\tau-\delta_t}\frac{w^2r_X}{K\delta_t}\left(1-r_X\delta_t\right) \\
&= \frac{\tau^2}{2\tau-\delta_t}\frac{w^2r_X}{K}\left(1-r_X\delta_t\right)
\end{aligned} \tag{13}$$

Hence

$$\lim_{\delta_t\to 0}\sigma^2 = \frac{\tau w^2 r_X}{2K} \tag{14}$$

For further analysis, it may also be beneficial to derive the covariance function of h(k) for a discrete time-step k i.e. C(k,k'). It is given by:

$$\begin{aligned}
\mathbb{E}\left[\widetilde{h}(k)-\mu_h)(\widetilde{h}(k')-\mu_h)\right] &= \mathbb{E}\left[\widetilde{h}(k)\widetilde{h}(k')\right] - \mu_h^2 \\
&= \mathbb{E}\left[\widetilde{h}(k)\widetilde{h}(k')\right] - (wr_X)^2
\end{aligned} \tag{15}$$

Using the results of equation (7) it can be seen that

$$\mathbb{E}\left[\widetilde{h}(k)\widetilde{h}(k')\right] = \begin{cases} \frac{w^2}{K}\left[\frac{r_X}{\delta_t} + (K-1)r_X^2\right] & k = k' \\ w^2r_X^2 & k \neq k' \end{cases} \tag{16}$$

Therefore:

$$C_h(k,k') = \begin{cases} \frac{w^2r_X}{K}\left[\frac{1}{\delta_t} - r_X\right] & k = k' \\ 0 & k \neq k' \end{cases}$$

Part C

Using the expressions calculated in equations (12) and (13), μ and σ^2 were plotted against K as shown in figure 4.

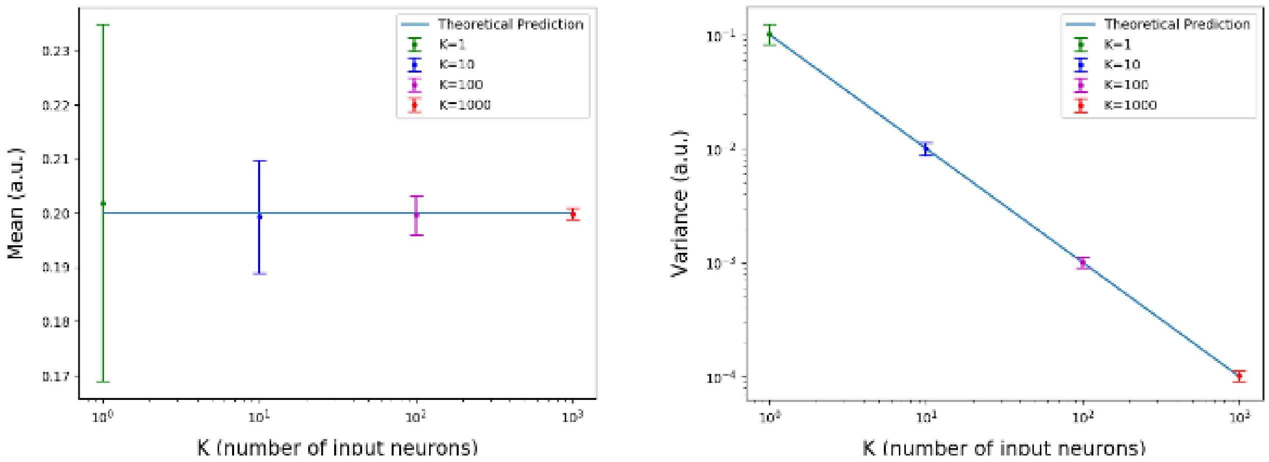


Figure 4: Change in expectation and variance of $\widetilde{V}(k)$ with K. Plots were taken from Liebana-Garcia, S. C. [1]. The light blue line shows theoretical predictions while the dots with error bars show experimental observations. The plots show agreement between experimental and theoretical values. The experimental values were obtained by average the results over 50, 10s long, experimental runs, after transients were removed.

Part D

For the mean potential μ to be equal to the threshold potential V_th regardless of the number of inputs K the following relationship must be satisfied:

$$\begin{aligned} \mu = \tau w r_X &= V_{th} \\ \therefore w &= \frac{V_{th}}{\tau r_X} \end{aligned} \tag{17}$$

With V_{th}, τ and r_X taking the default values of 1, 20ms an 10Hz respectively, this gives $w = 5$. This agrees with experimental values as figure 5 shows the membrane potential to fluctuate around 1, when w is set to 5.

Part E

The spike-and-reset mechanism was enabled, and w was varied by trial-and-error to give an output spike-rate of 10Hz i.e. 20 counts over a 2s run. It was found that $w = 4.25$ produced the desired results. Using this value of w and a counting window of 100ms, the Fano factor was found to vary considerably from one run to the next. However on multiple runs a Fano factor less than 1 was achieved. The human cerebral cortex is typically observed to have a

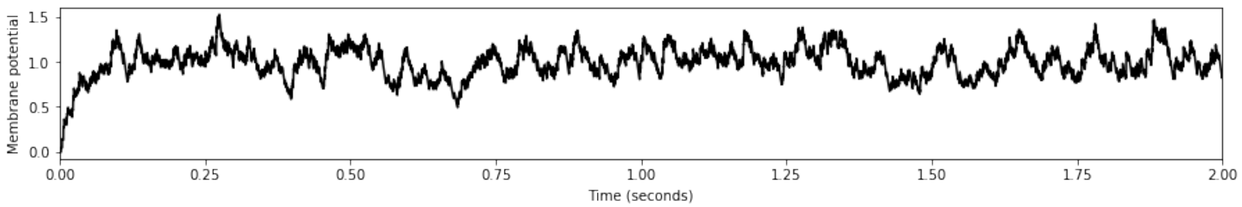


Figure 5: Membrane potential of multi-Poisson-input LIF neuron with synaptic weight w = 5. After transients have passed, it can be seen that the voltage fluctuates about a value of 1. This agrees with experimental predictions calculated in equation (17).

Fano factor greater than 1, thus this model as it currently stands is not sufficient to model the human brain.

```python
def fano(Si, window, delta = dt):
    dur = delta * len(Si)
    numWindows = int(dur / window)
    spikesInWindow = np.zeros(numWindows)
    numIndicesPerWin = int(window / delta)

    if window >= dur:
        mean = np.sum(Si)
    else:
        for i in range(numWindows):
            spikesInWindow[i] = np.sum(Si[i*numIndicesPerWin:(i+1)*numIndicesPerWin])
        mean = np.average(spikesInWindow)

    variance = np.average((spikesInWindow - mean)**2)
    return variance / mean
```

Listing 3: Python excerpt to calculate the Fano factor for spike-train S_i

4: Single LIF neuron with many E and I Poisson inputs

In this section, the LIF neuron in section 3 was extended to take K inhibitory Poisson inputs in addition to the K excitatory Poisson inputs. The synaptic weights for each connection was taken to be $w/\sqrt{K}$ and $-w/\sqrt{K}$ for excitatory and inhibitory connections respectively.

Part A

In a similar fashion to section 3 part B, the mean of the membrane potential, under these new conditions, can be derived as follows:

$$\begin{aligned}\mathbb{E}\left[\widetilde{V}(k)\right] &= \mathbb{E}\left[\left(1-\frac{\delta_t}{\tau}\right)\widetilde{V}(k-1)+\delta_t\sqrt{K}\left(\widetilde{h}^E(k-1)-\widetilde{h}^I(k-1)\right)\right] \\ &= \left(1-\frac{\delta_t}{\tau}\right)\mathbb{E}\left[\widetilde{V}(k-1)\right]+\delta_t\sqrt{K}\,\mathbb{E}\left[\widetilde{h}^E(k)-\widetilde{h}^I(k)\right] \\ &= \left(1-\frac{\delta_t}{\tau}\right)\mathbb{E}\left[\widetilde{V}(k)\right]+\delta_t\sqrt{K}\left(\mathbb{E}\left[\widetilde{h}^E(k)\right]-\mathbb{E}\left[\widetilde{h}^I(k)\right]\right)\end{aligned} \tag{18}$$

Then, noting that the excitatory and inhibitory inputs have the same mean, we get the expression:

$$\mathbb{E}\left[\widetilde{V}(k)\right] = \left(1-\frac{\delta_t}{\tau}\right)\mathbb{E}\left[\widetilde{V}(k)\right] \tag{19}$$

Thus $\mu = 0$ is the only value that satisfies this equation. The variance of the membrane potential can now be derived as follows:

$$\begin{aligned}\mathbf{Var}\left[\widetilde{V}(k)\right] &= \mathbf{Var}\left[\left(1-\frac{\delta_t}{\tau}\right)\widetilde{V}(k-1)+\delta_t\sqrt{K}\left(\widetilde{h}^E(k-1)-\widetilde{h}^I(k-1)\right)\right] \\ &= \left(1-\frac{\delta_t}{\tau}\right)^2\mathbf{Var}\left[\widetilde{V}(k-1)\right]+\delta_t^2K\mathbf{Var}\left[\widetilde{h}^E(k)-\widetilde{h}^I(k)\right]\end{aligned} \tag{20}$$

where we have used the fact that

$$\mathbf{Var}\left[X+Y\right] = \mathbf{Var}\left[X\right]+\mathbf{Var}\left[Y\right]+2\mathbf{cov}\left[X,Y\right] \tag{21}$$

where

$$\begin{aligned}X &= \left(1-\frac{\delta_t}{\tau}\right)\widetilde{V}(k-1) \\ Y &= \delta_t\sqrt{K}\left(\widetilde{h}^E(k)-\widetilde{h}^I(k)\right)\end{aligned} \tag{22}$$

giving

$$\begin{aligned}\mathbf{Var}\left[X\right] &= \left(1-\frac{\delta_t}{\tau}\right)^2 \mathbf{Var}\left[\widetilde{V}(k-1)\right] \\ \mathbf{Var}\left[Y\right] &= \delta_t^2 K \mathbf{Var}\left[\widetilde{h}^E(k)-\widetilde{h}^I(k)\right] \\ \mathbf{cov}\left[X,Y\right] &= 0\end{aligned} \tag{23}$$

where the last equality was obtained by noting that the membrane potential at any given time-step k is independent of the random Poisson inputs at that same time-step. Using forward Euler integration, the inputs at any given time-step only influence the membrane potential at subsequent time-steps.

The covariance function of h, $C_h(k,k')$, was calculated at the end of section 3 part B. Noting that $\widetilde{h}^E(k)$ is independent of $\widetilde{h}^I(k)$ and using equation (21) we may assert that

$$\mathbf{Var}\left[\widetilde{h}^E(k)-\widetilde{h}^I(k)\right] = 2\frac{w^2 r_X}{K}\left(\frac{1}{\delta_t}-r_X\right) \tag{24}$$

Hence at stationarity

$$\mathbf{Var}\left[\widetilde{V}(k)\right] = \left(1-\frac{\delta_t}{\tau}\right)^2 \mathbf{Var}\left[\widetilde{V}(k)\right] + \delta_t^2 K\left[2\frac{w^2 r_X}{K}\left(\frac{1}{\delta_t}-r_X\right)\right]$$

$$\begin{aligned}\left(1-\left(1-\frac{\delta_t}{\tau}\right)^2\right)\mathbf{Var}\left[\widetilde{V}(k)\right] &= 2\delta_t^2 w^2 r_X\left(\frac{1}{\delta_t}-r_X\right) \\ \mathbf{Var}\left[\widetilde{V}(k)\right] &= \frac{2\delta_t^2 w^2 r_X\left(\frac{1}{\delta_t}-r_X\right)}{\left(1-\left(1-\frac{\delta_t}{\tau}\right)^2\right)} \\ &= \frac{2\delta_t w^2 r_X(1-r_X\delta_t)}{1-\left(1+\frac{\delta_t^2}{\tau^2}-2\frac{\delta_t}{\tau}\right)} \\ &= \frac{2w^2\tau^2 r_X(1-r_X\delta_t)}{2\tau-\delta_t}\end{aligned} \tag{25}$$

Therefore

$$\begin{aligned}\lim_{\delta_t\to 0}\left[\mathbf{Var}\left[\widetilde{V}(k)\right]\right] &= \frac{2w^2\tau^2 r_X}{2\tau} \\ &= w^2\tau r_X\end{aligned} \tag{26}$$

Equation (26) shows that the variance does not decrease with K as it did in section 3.

Part B

Through trial-and-error, w was varied such that the neuron fired at a rate of 10Hz on average. This value of w was found to be c. 4.25. Using this value of w and a counting window of 100ms (just as in section 3 part E) the Fano factor was found to be greater than 1 on multiple runs. Therefore the introduction of inhibitory neurons allowed the LIF model to more closely approximate the behaviour of the human cerebral cortex.

Why should the introduction of inhibitory neurons give a more human-like (higher) Fano factor? We saw that at stationarity, the mean of the fluctuations with and without the inhibitory neurons were 0 and 42.5 ($wr_X = 4.25 * 10$) respectively. We also saw that the variances were $2.25 * 10^{-3}$ ($\frac{w^2 \tau r_X}{2K} = \frac{0.020*1.5*10}{2*100}$) and 0.45 ($w^2 \tau r_X = 1.5^2 * 0.020 * 10$). In the no-inhibitory-neuron case, the mean membrane potential at stationarity is non-zero, therefore after a spike-and-reset there is a delay before the potential can return to stationarity. This delay was found to be of the order of 100ms. Note also how small the variance is at stationarity i.e. the fluctuations in V around the trend-line are very small. Therefore the mechanism that allows V to cross the threshold is its rising mean before stationarity is reached. The chances that 2 action potentials fire in any given 100ms time-window are slim as that would happen only if one action potential fires at the start of the 100ms time-window and another fires towards the end of the 100ms window. Therefore it is expected that only 1 action potential fires on average in each window, and the deviation from this window-spike-count of 1 is slim i.e. small variance in spike-count. The Fano factor is therefore $\frac{x}{1}$ where x is some small number less than 1. In the inhibitory case, because the membrane potential is 0 at stationarity, there is no delay after the spike-and-reset for V to return to stationary levels. We also note that the variance in V is significantly higher (of the order of 200 times) than that of the no-inhibitory-neuron case. This fact, in conjunction with the zero mean, leads us to assert that the mechanism that allows V to rise above the threshold is the large fluctuations around the baseline. Therefore the probability that a spike occurs in a 100ms time-frame is the same as the probability that the voltage fluctuates beyond some fixed fraction of its variance. It can therefore be inferred that in the inhibitory-neuron case there is a more variable nature to the number of action potentials fired in any given 100ms time-window. Thus, the Fano factor is $\frac{y}{1}$ where y is some value greater than x. This is why the Fano factor is seen to be greater upon the introduction of inhibitory neurons. Hence this Fano factor more closely approximates values seen in the human cerebral cortex.

5: Full network

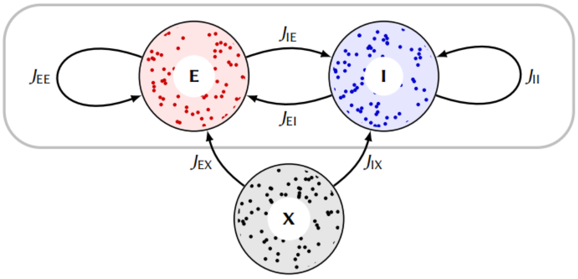


Figure 6: Full network; taken from Coursework handout [2]. The network is composed of 3 populations each with N neurons. The neurons in population X fire randomly according to a Poisson distribution with the same rate r_X. The neurons in populations E and I are leaky-integrate-and-fire neurons, which fire according to the dynamics described in equation (27). The synaptic weights at the outputs of neurons in populations E and X are positive (excitatory) while those at the outputs of neurons in population I are negative (inhibitory).

In this section, the full network was assembled as shown in figure 6. The dynamics of the network are governed by equation (27). Each neuron was chosen to take inputs from K randomly selected neurons in each of the 3 populations.

$$\frac{dV_i^\alpha}{dt} = -\frac{V_i^\alpha(t)}{\tau} + \sum_{\beta \in \{E,I,X\}} \frac{J_{\alpha\beta}}{\sqrt{K}} \sum_{j \in C_i^{\alpha\beta}} S_j(t) \tag{27}$$

Part A

The mean firing rates would satisfy J_{EE}

In lectures, it was desired that the expected value of the sum of the inputs to each neuron would finite i.e. it would not vary with the number of inputs to each neuron (K). This can be interpreted to mean that the magnitude of the input to any given neuron should not vary as new material is learned by the brain or new experiences are had such that new connections are formed. Another interpretation is that the number of inputs to each neuron is very large, and can be assumed to be infinite, so a theoretical measure must be taken to constrain the sum of the inputs to a finite value. This argument yields the following equation:

The theoretical arguments developed in the lecture state that in the case of finite mean inputs and fixed non-degenerate weights

$$\begin{cases} \sqrt{K}(J_{EE}r_E + J_{EI}r_I + J_{EX}r_X) = \mathcal{O}(1) \\ \sqrt{K}(J_{IE}r_E + J_{II}r_I + J_{IX}r_X) = \mathcal{O}(1) \end{cases} \tag{28}$$

Therefore,

$$\begin{cases} J_{EE}r_E + J_{EI}r_I + J_{EX}r_X = \mathcal{O}(1/\sqrt{K}) \to 0 \\ J_{IE}r_E + J_{II}r_I + J_{IX}r_X = \mathcal{O}(1/\sqrt{K}) \to 0 \end{cases} \tag{29}$$

as $K \to \infty$. Hence if the external neurons fire with rate $r_X = 10$ and the synaptic weight parameters are taken to have the following values:

$$\begin{aligned} J_{EE} = 1,\ J_{EI} = -2,\ J_{EX} = 1, \\ J_{IE} = 1,\ J_{II} = -1.8,\ J_{IX} = 0.8 \end{aligned} \tag{30}$$

The equations governing the firing rates of the excitatory population r_E and the inhibitory population r_I can be reduced to

$$\begin{cases} r_E - 2r_I + r_X = 0 \\ r_E - 1.8r_I + 0.8r_X = 0 \end{cases} \tag{31}$$

giving

$$r_E = r_I = r_X \tag{32}$$

Part B

The full network was simulated in Python according to figure 6 using the default parameter values and the synaptic weights in equation (30). The average spike rates observed in the excitatory and inhibitory populations were $r_E = 12.89$Hz and $r_I = 11.58$Hz. This agrees fairly well with theoretical predictions in equation 32 as the orders of magnitude are similar.

Part C

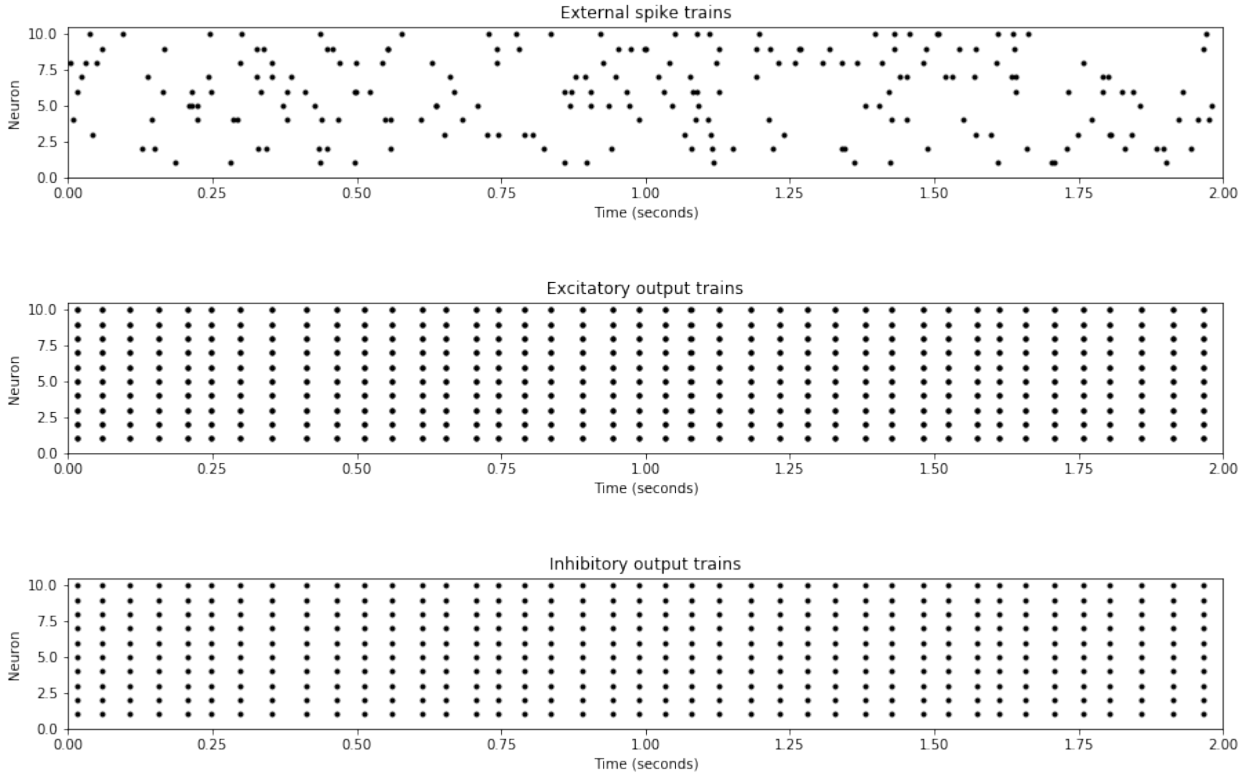
According to the theory in equation (30), r_E and r_I should match r_X as it increases. These predictions were hence put to the test by running the simulation several times, changing the value of r_X each time:

r_X	r_E	r_I
5Hz	7.05Hz	5.85Hz
10Hz	12.89Hz	11.58Hz
15Hz	18.54Hz	17.00Hz
20Hz	24.09Hz	22.39Hz

Table 1: Dependence of r_E and r_I on r_X [3]. Table created by Liebana-Garcia, S. C. It can be seen that the values of r_E and r_I are of the same order of r_X as r_X is varied. Therefore observations agree with theoretical predictions in equation (32)

.

Part D



External spike trains

Excitatory output trains

Inhibitory output trains

Figure 7: Raster plots showing the activity of 10 neurons from the external, excitatory and inhibitory populations in a fully-connected network. It can be seen that the excitatory and inhibitory neurons show the same activity at their outputs. This was expected as each neuron in the excitatory and inhibitory neurons has N of the same connections from either population i.e. each neuron has identical connections therefore yielding identical activity.

The number of neurons in each population N was set equal to the number of connections between populations K. This means the network is fully-connected i.e. each neuron takes all other neurons as its input. This means that every neuron in the network has the same connections. Consequently while the neurons in the X population fire randomly according to a Poisson distribution, the LIF neurons in the E and I populations are expected to exhibit the same activity at their outputs. This is exactly what is observed in the simulations in figure 7.

This behaviour clearly does not accurately model that seen in the human cerebral cortex. What assumptions have been made in developing the theory that are now violated? The network simulated in figure 7 is fully connected. However, the theory assumes that the network should not be fully connected. Though K is large, it should still be significantly smaller than N. This would allow each neuron to have different connections, thus yielding different output.

```python
Wee = get_weights(N, K, w=1.0)
Wie = get_weights(N, K, w=1.0)
Wei = get_weights(N, K, w=-2.0)
Wii = get_weights(N, K, w=-1.8)
Wex = get_weights(N, K, w=1.0)
Wix = get_weights(N, K, w=0.8)

def get_weights(N, K, w):
    W_unshuff = np.concatenate([np.ones([N, K])*w/np.sqrt(K), np.zeros([N, N-K])], 1)
    W = np.zeros([N, N])
    for i in range(N):
        W[i, :] = np.random.permutation(W_unshuff[i,:])
    return W
```

Listing 4: Python excerpt to make random network connections

References

1. Liebana-Garcia, S. C. (scl63@cam.ac.uk). Variation in mean and variance of membrane potential of a multi-Poisson-input LIF neuron.

2. 4G3 Coursework 1 2020 handout. https://www.vle.cam.ac.uk/pluginfile.php/11820311/mod_folder/content/0/cw01.pdf?forcedownload=1

3. Liebana-Garcia, S. C. (scl63@cam.ac.uk). Dependence of r_E and r_I on r_X.