The integrate and fire model

The integrate and fire model is a simple model of the voltage dynamics of a neuron. According to the model, the voltage V satisfies:

$$\tau_m \frac{dV}{dt} = E_L - V + R_m I \tag{1}$$

I might end up being synaptic input, but traditionally we write the equation to match the *in vivo* experiment where I is an injected current from an electrode, so we write I_e, 'e' for electrode. τ_m is a time constant which physiologically depends on the membrane resistance and capacitance.

The equation on its own above leaves out the possibility that there are other non-linear changes in the currents through the membrane as V changes. This is a problem since there are other non-linear changes in the currents through the membrane as V changes. The equation above leaves these out, in fact, the non-linear effects are strongest for values of V near where a spike is produced, so one approach is to use the linear equation unless V reaches a threshold value and then add a spike 'by hand'. This has the effect of changing the voltage to a reset value, this mimics what happens in the neuron, or in the Hodgkin Huxley model which includes the full non-linear dynamics which makes the spike. Anyway, in summary

- V satisfies
$$\tau_m \frac{dV}{dt} = E_L - V + R_m I_e \tag{2}$$
- If $V \geq V_T$ a spike is recorded and the voltage is set to a reset value V_R.

The reset value, the voltage after the spike is often set equal to the leak potential. This is the **leaky integrate and fire model**. It lacks lots of the details important in the dynamics of neurons, but is useful and often used for modelling the behaviour of large neuronal networks or for exploring ideas about neuronal computation in a relatively straight-forward setting.

Numerical integration

In some circumstances it is possible to solve the integrate-and-fire model analytically, that is, it is possible to write down a solution. However, it is often necessary to solve it approximately using a computer. The idea with

that is that time is sliced up into small time steps and the value at each successive time step is calculated approximately from the previous time step using the differential equation. This relies on the Taylor expansion:

$$V(t+\delta t) = V(t) + \delta t \frac{dV}{dt} + \frac{1}{2}(\delta t)^2 \frac{d^2V}{dt^2} + \dots \tag{3}$$

where the derivatives are evaluated at t, not $t+\delta t$. The '...' stands for more terms involving higher and higher powers of δt. δt in turn is the small time step, which means that higher and higher powers of δt are smaller and smaller still. The hope is that the derivatives of V don't get bigger and bigger so that we can approximate

$$V(t+\delta t) \approx V(t) + \delta t \frac{dV}{dt} \tag{4}$$

This is the Euler approximation, other approximations include more of the terms, but you can see that it allows you to calculate a value for $V(t+\delta t)$ from the value at $V(t)$ and the differential equation.

To say this again, but in a way more directly related to how it would work on a computer, let $t_n = n\delta t$ and $V_n = V(t_n)$ and imagine the differential equation says

$$\frac{dV}{dt} = f(V) \tag{5}$$

so in our case $f(V) = [E_l - V + R_m I_e]/\tau_m$. Now the Euler approximation says that:

$$V_{n+1} = V_n + \delta t f(V_n) \tag{6}$$

It is easy to see how that might be calculated as part of a Python programme. The thing that is missing is the reset; that is easy to add though, all that is needed is an `if` statement to check whether the new value of V exceeds the threshold value V_T.

Programming challenge

Simulate an integrate and fire model with the following parameters for one second: $\tau_m = 10$ms, $E_L = V_r = -70$ mV, $V_t = -40$ mV, $R_m = 10$ MΩ, $I_e = 3.1$ nA. Use Euler's method with time-step $\delta t = 1$ ms. Here E_L is the leak potential, V_r is the reset voltage, V_t is the threshold, R_m is the membrane

resistance, that is one over the conductance, and τ_m is the membrane time constant. Plot the voltage as a function of time. For simplicity assume that the neuron does not have a refractory period after producing a spike. You do not need to plot spikes - once membrane potential exceeds threshold, simply set the membrane potential to V_r.

We haven't spoke about plotting; this is done using `matplotlib`; rather than try to describe it there is a sample programme available in the `github` folder that plots a sine function.