Numerical integration of ordinary differential equations

a. Euler Method

So far we have studied the RC differential equation, a first order linear differential equation which has the form:

$$\tau \frac{dV(t)}{dt} = V_{\infty}(t) - V(t), \quad \tau = RC \tag{1}$$

$$V_{\infty}(t) = V_{rest} + R\, I(t) \tag{2}$$

We are interested in solving for *V(t)* . To do this, we will numerically integrate the equation using the Euler Method. We iterate through time, using our estimated *V(t)* and the derivative of *V* at time *t* to estimate *V(t+dt)*.

The formal definition of the derivative is:

$$\frac{dV(t)}{dt} = \lim_{dt \to 0} \frac{V(t+dt) - V(t)}{dt}.$$

In reality we cannot numerically evaluate this ratio as *dt* goes to zero. However, we can pick a finite but sufficiently small time interval Δt where we can approximate the derivative as a difference equation:

$$\begin{aligned} \frac{dV(t)}{dt} &\approx \frac{\Delta V(t)}{\Delta t} \\ \frac{\Delta V(t)}{\Delta t} &= \frac{V(t+\Delta t) - V(t)}{\Delta t}. \end{aligned} \tag{3}$$

We combine equations 1 and 3 to obtain:

$$\frac{\Delta V(t)}{\Delta t} = \frac{V(t+\Delta t) - V(t)}{\Delta t} \approx \frac{1}{\tau}(V_{\infty}(t) - V(t)).$$

So far, we have re-written the RC equation as a difference equation. Our objective is to find an iterative algorithm that can compute $V(t+\Delta t)$ as a function of *V(t)* and *I(t)*. If we rearrange the above equation we obtain:

$$V(t+\Delta t) \approx V(t) + \frac{\Delta t}{\tau}\left(V_{\infty}(t) - V(t)\right).$$

Before we start numerically integrating, we need to set Δt and the membrane voltage at time t_0=0. This voltage is denoted $V_0=V(t_0)$ and called initial conditions. From the initial conditions, we numerically integrate $V(t)$ in time steps of size Δt starting from t_0. This constitutes the **Euler method**, which can be implemented using a *for-loop* in MATLAB.

Note that the Euler approximation for the RC equation can be written in terms of an index n that counts multiples of the time-step Δt (i.e. $t = n\Delta t$):

$$V_{n+1} = V_n + \frac{\Delta t}{\tau}\left(V_{\infty,n} - V_n\right).$$

This notation makes the implementation of the algorithm in MATLAB simpler.

In essence this method linearly extrapolates the derivative in the time interval t to $t + \Delta t$ in each iteration. This scheme works well for first order linear ordinary differential equations (like the RC system) provided that the time-step Δt is much smaller than the time constant τ. This will ensure that difference equation approximation is accurate. Otherwise, the method would fail us miserably.

It is worth noticing that the Euler algorithm is very inefficient if the input current is fluctuating (changing) at a temporal scale that is much slower than the membrane time constant τ. In this case, we will be using a very small Δt when the input is almost constant during that period of time. Therefore, we will be wasting a lot of unnecessary numerical computations (steps in the *for-loop*) to find $V(t)$.

The Euler method is important for pedagogical reasons, as it is conceptually easy to understand. The following section explains the Exponential Euler Method, which is more efficient for slowly varying inputs.

b. The Exponential Euler Method.

The Exponential Euler algorithm takes advantage of the analytic solution for a constant input current:

$$V(t) = V_\infty + (V_0 - V_\infty)\exp\left(-\frac{t}{\tau}\right)$$

Now, let's assume that we have a current input that is not constant, but changes slowly. That is, we can approximate the input current as constant within time windows of some size Δt. With this approximation, we can derive an iterative method to obtain the solution:

$$V_{n+1} = V_{\infty,n} + (V_n - V_{\infty,n})\exp\left(-\frac{\Delta t}{\tau}\right)$$

where we use the same notation as before for the index *n*. This is the **Exponential Euler Method**. Notice that this approximation is very different than in the regular Euler method. Here, we are exploiting the fact that for a constant input we know the exact solution of the differential equation. And indeed this method would be exact for a constant applied current. You can check that if you pick too large a Δt, the worst that would happen is that $V_{n+1} = V_{\infty,n}$. What is the worst case with the regular Euler Method?

In practice, we will pick a time-step Δt for the exponential Euler method that is still smaller than the membrane time constant τ. However, this method will allow us to pick a larger time-step compared to the Euler method without wasting computational resources or making inaccurate estimates.

The Exponential Euler Method is easily implemented in MATLAB using a *for-loop*.

This is the method we will be using in the problem set. It **only** works for differential equations of the form:

$$\frac{dy}{dt} = -\frac{y}{\tau_y} + a(t)$$

Which can always be written in the form:

$$\tau_y \frac{dy}{dt} = y_\infty - y$$

This is the case for the RC equation, the integrate and fire model, the Hodgkin-Huxley model, and most conductance based neurophysiological models.

To conclude, is important to know which approximations each method makes, and what can limit their applicability. In the case of the Euler method, the principal flaw comes from the fact that the derivative is evaluated at the beginning of the interval and it is assumed to have the same value across it. In cases where the input fluctuates at very short time scales or where the differential equation is highly non-linear, Euler based methods are not the best choice. In those cases, higher order methods such as the Runge-Kutta will overcome these problems and provide efficient and stable algorithms to find numerical solutions. References to application of these methods in MATLAB and to neuroscience problems are given at the end of this document.

References:

1. Press, WH; Teukolsky, SA; Vetterling, WT and Flannery, BP. *Numerical recipes: the art of scientific computing.* Cambridge University Press, 2007. 3rd edition.

(Great general reference for numerical methods. A bit out-dated in terms of programming as original routines written in FORTRAN and C)
2. Chapra, SC. *Applied Numerical Methods with MATLAB for engineers and scientists*. McGraw-Hill, 2010. 3rd edition. (Numerical methods with applications in MATLAB. Well-documented and easy to read.)
3. Ermentrout, GB and Terman, DH. *Mathematical Foundations of Neuroscience.* Springer, 2010. (Great and rigorous book in computational neuroscience, explaining some numerical methods with applicability.)

9.40 Introduction to Neural Computation
Spring 2018