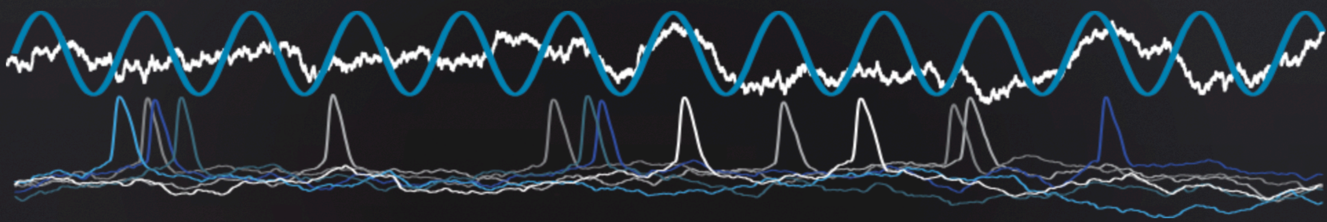
PRINCIPLES OF COMPUTATIONAL NEUROSCIENCE



AN INTRODUCTORY COURSE OFFERED TO
(UNDER)GRAD NEUROSCIENCES STUDENTS AT UNITS AND SISSA, TRIESTE (ITALY).

Michele GIUGLIANO

Hands-on session: assignment 1

Numerical solutions of an o.d.e.

forward Euler's method

x

- The independent variable is *discretised* $\quad x_0, x_1, x_2, \ldots, x_{k-1}, x_k, x_{k+1}, \ldots$
  - e.g. uniformly $\quad x_k = k\ \Delta x$
- Derivatives are *approximated* $\quad \dfrac{df}{dx} \approx \dfrac{f(k\Delta x) - f((k-1)\Delta x)}{\Delta x}$

$$\frac{df}{dx} = -30\ f(x) \quad \longrightarrow \quad f_k \approx f_{k-1} - 30\ \Delta x\ f_{k-1}$$

o.d.e. $\qquad$ algebraic iterative equation

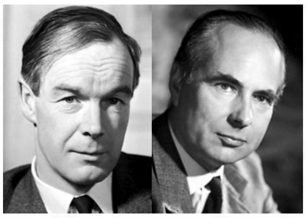
```
for k=2:N
    f[k] = f[k-1] - 30 * Δx * f[k-1]
end
```

Charge-balance equation

$$C\frac{d}{dt}V = (i_1 + i_2 + i_3 + ...)$$

$$C\frac{d}{dt}V = [G_{Na}(E_{Na} - V) + G_K(E_K - V) + G_{Ca}(E_{Ca} - V) + ...]$$

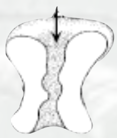
O.D.E., first-order, non-homogeneous, non-linear, time-varying: Numerical methods and *in silico* studies



HH described (mesoscopic) membrane permeability by (phenomenological) kinetic schemes

state 1 state 2

$$\mathit{closed} \underset{\beta(V)}{\overset{\alpha(V)}{\rightleftarrows}} \mathit{open}$$

$$\frac{dn}{dt} = -(\alpha + \beta)\,n + \alpha$$

Joint exercise (was “Assignment 1”)

$$\frac{dV}{dt} = 0.15\ (-70 - V(t)) + sin(2\ \pi\ F\ 0.001\ t)\ +\ 1$$

$$V(0) = -70$$

$$F = 2 \quad or \quad F = 200$$

$$\frac{dV}{dx} \approx \frac{V(k\Delta t) - V((k-1)\Delta t)}{\Delta t} \qquad t \rightarrow (k-1)\Delta t$$

... $\longrightarrow$ ...

o.d.e. algebraic iterative equation

- **write down on paper** the o.d.e. as a discrete-time (Euler’s) approxim.
- by a new *Julia-Jupyter Notebook* (inspired from the one provided)
  - **solve** the discrete-time numerical approximation of such an o.d.e.
  - **plot both the graphs of the function V(t) and of the function** $u(t) = sin(2\ \pi\ F\ 0.001\ t)\ +\ 1$
  - **plot also the graphs of *(V(t)+70)* and of *(u(t)-1)***
  - **describe** in words how the solution looks like
  - **document your entire work in Markdown**

- **beware** of properly defining u(t) (Δt and *F*)
- **comment and explain** every line of the code
- perform/explore **longer simulations**, until a state of *steadyness* is reached
- zoom on the last “cycles” of the V(t) in your (long) simulation
  - extract its (relative) **peak amplitude**
- plot the **peak amplitude** for distinct values of F (e.g. 2, 5, 8, 10, 20, 50, 80, 100, 200, 500, 1000, 2000, 5000)
- can you explain what happens to the ***phase* of V(t)** with respect to the *phase* of u(t) ?
- can you ***comment “functionally”* the input(u)-output(V) transformation?**