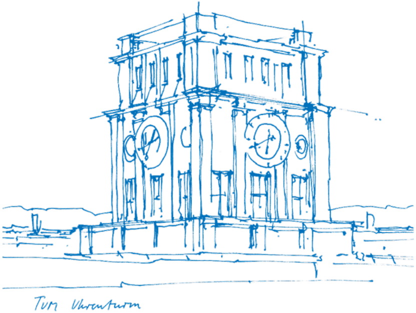
# Multi-Compartment Model

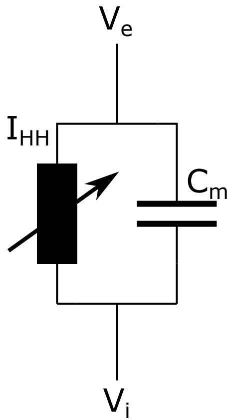
Neuroprothetics WS 2021/22

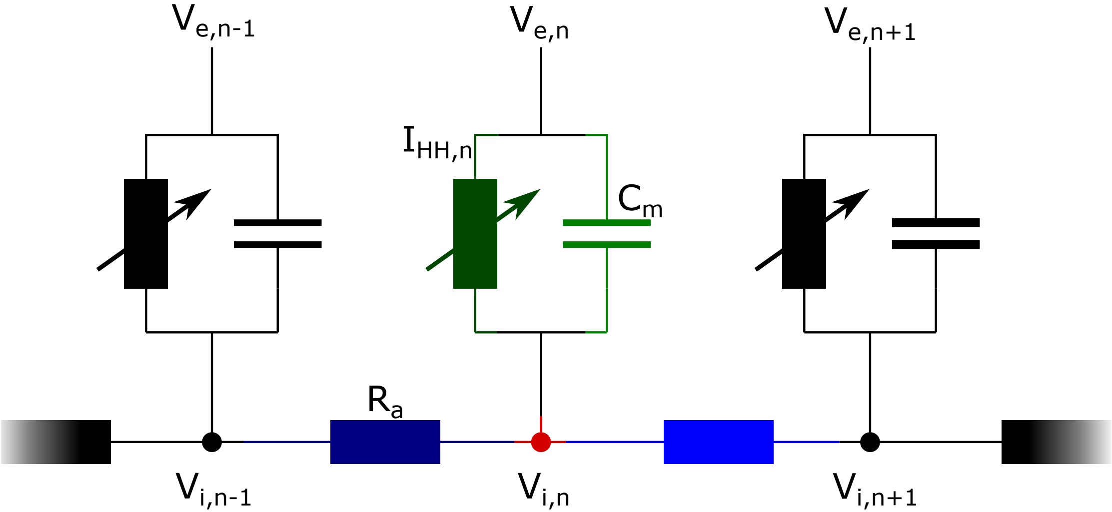
Albert Croner

**Prof. Werner Hemmert**
Technical University of Munich
Bio-Inspired Information Processing



- HH model is a single compartment model.
- To simulate multiple compartments, connect single compartments with $R_a$.
- $R_a$ is the axoplasmatic resistance (internal resistance of an axon).
- If a myelinated axon is modeled, $R_a$ can model internodes.
- In this case, internodes are assumed to be perfect insulators





$V_{e,n-1}$
$V_{e,n}$
$V_{e,n+1}$
$I_{HH,n}$
$C_m$
$R_a$
$V_{i,n-1}$
$V_{i,n}$
$V_{i,n+1}$

With the multicompartment equivalent circuit, ODEs can be derived. The membrane potential is defined as $V_m = V_i - V_e$. Kirchhoff's current law (node) for the compartment n results in the following ODEs:

$$0 = C_m \frac{dV_{m,n}}{dt} + I_{HH,n} + \frac{V_{i,n} - V_{i,n-1}}{R_a} + \frac{V_{i,n} - V_{i,n+1}}{R_a}$$

$$\frac{dV_{m,n}}{dt} = -\frac{1}{C_m} I_{HH,n} + \frac{1}{C_m} \frac{V_{i,n-1} - 2V_{i,n} + V_{i,n+1}}{R_a}$$

Adding a stimulating current:

$$\frac{dV_{m,n}}{dt} = \frac{1}{C_m}(-I_{HH,n} + I_{stim,n}) + \frac{1}{C_m} \frac{V_{i,n-1} - 2V_{i,n} + V_{i,n+1}}{R_a}$$

The system can be described for all indices of n using vectors:

$$\frac{d}{dt}\vec{V}_m = \frac{1}{C_m}(-\vec{I}_{HH} + \vec{I}_{stim}) + \frac{1}{C_m R_a}\boldsymbol{C}\vec{V}_i$$

With $\vec{V}_m = \begin{pmatrix} V_{m,1} \\ V_{m,2} \\ \vdots \end{pmatrix}$ and $\boldsymbol{C} = \begin{pmatrix} -1 & 1 & & & 0 \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ 0 & & & 1 & -1 \end{pmatrix}$

$\vec{V}_i, \vec{I}_{HH}, \vec{I}_{stim}$ are vectors similar to $\vec{V}_m$.

**$\boldsymbol{C}$ is not $C_m$!**

The membrane potential $V_m$ of a neuron can be written as
$\vec{V}_m = \vec{V}_i - \vec{V}_e \Rightarrow \vec{V}_i = \vec{V}_m + \vec{V}_e$ .

$$\frac{d}{dt}\vec{V}_m = \frac{1}{C_m}(-\vec{I}_{HH} + \vec{I}_{stim}) + \frac{1}{C_m R_a}\boldsymbol{C}\vec{V}_m + \frac{1}{C_m R_a}\boldsymbol{C}\vec{V}_e$$

In exercise 5: $\vec{V}_e = 0$ (no external potential gradient)

# Implicit Euler Method

This ODE system will be solved using the Implicit Euler Method:

$$\frac{dV}{dt} = f(t)$$

$$V(t + \Delta t) = V(t) + \Delta t \cdot f(t + \Delta t)$$

$$\vec{V}_m(t+\Delta t) = \vec{V}_m(t) + \frac{\Delta t}{C_m}(-\vec{I}_{HH}(t+\Delta t) + \vec{I}_{stim}(t+\Delta t)) + \frac{\Delta t}{C_m R_a}\boldsymbol{C}\vec{V}_m(t+\Delta t)$$

$$\underbrace{\left(\boldsymbol{I} - \frac{\Delta t}{C_m R_a}\boldsymbol{C}\right)}_{\boldsymbol{A}} \cdot \underbrace{\vec{V}_m(t+\Delta t)}_{\vec{x}} = \underbrace{\vec{V}_m(t) + \frac{\Delta t}{C_m}(-\vec{I}_{HH}(t+\Delta t) + \vec{I}_{stim}(t+\Delta t))}_{\vec{b}}$$

This leads to a simple linear system of equations:

$$\boldsymbol{A} \cdot \vec{x} = \vec{b}$$

A system of equations like $\boldsymbol{A} \cdot \vec{x} = \vec{b}$ can be solved easily using a computer:

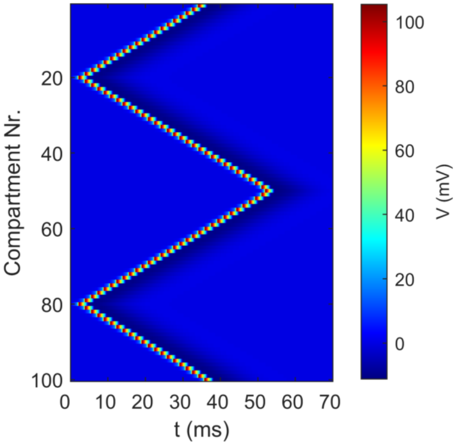
Matlab: $\mathrm{x} = \mathrm{A} \setminus \mathrm{b}$

Python: x = solve(A, b)
The solve function is provided by the Numpy package numpy.linalg

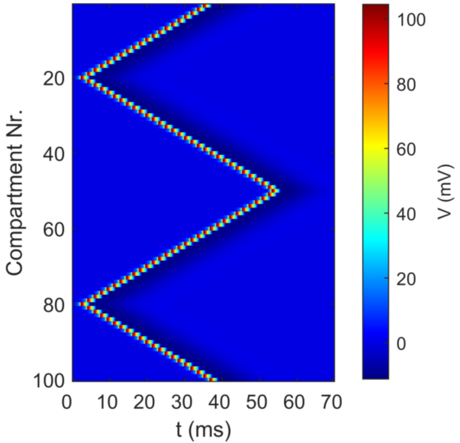
# Time index for $I_{HH}$

The presented equation has $I_{HH}(t + \Delta t)$ on the right hand side. To compute this correctly, $V(t + \Delta t)$ would be required.
This would lead to further restructuring of the equation, and more complicated code. For this reason, and as the difference in the result is neglegible, you may compute $I_{HH}(t + \Delta t)$ using $V(t)$, e.g.:

$$I_{Na}(t + \Delta t) = \bar{g}_{Na} \cdot m(t + \Delta t)^3 \cdot h(t + \Delta t) \cdot (V(t) - V_{Na})$$




Wrong time index for V




Correct time index for V