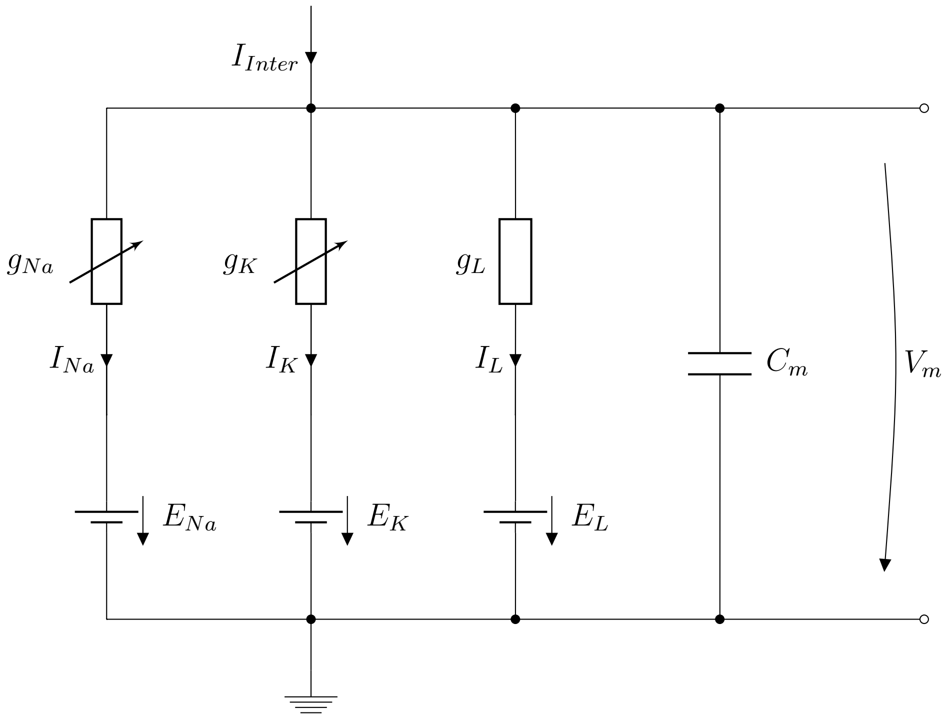
Workshop 1

Bioelektrische Signale - Axel Loewe

24.05.2019

1 Aufgabe 1



$$\alpha_n = 0.01 \left(-\frac{V_m + 55}{\exp\left(-\frac{V_m+55}{10}\right) - 1} \right)$$

$$\beta_n = 0.125 \exp\left(-\frac{V_m + 65}{80} \right)$$

$$\alpha_m = 0.1 \left(-\frac{V_m + 40}{\exp(-\frac{V_m+40}{10}) - 1} \right)$$

$$\beta_m = 4 \exp\left(-\frac{V_m + 65}{18} \right)$$

$$\alpha_h = 0.07 \exp\left(-\frac{V_m + 65}{20} \right)$$

$$\beta_h = \frac{1}{\exp\left(-\frac{V_m+35}{10}\right) + 1}$$

$$V_m = \begin{bmatrix} -100 & -90 & \dots & -40 & \dots & 30 & 40 \end{bmatrix}$$

$$\alpha_m(V_m)|_{V_m=-40\,\mathrm{mV}} = \text{undef.}$$

$$\alpha_m(-40\,\mathrm{mV}) := \lim_{V_m \to -40\,\mathrm{mV}} = 1$$

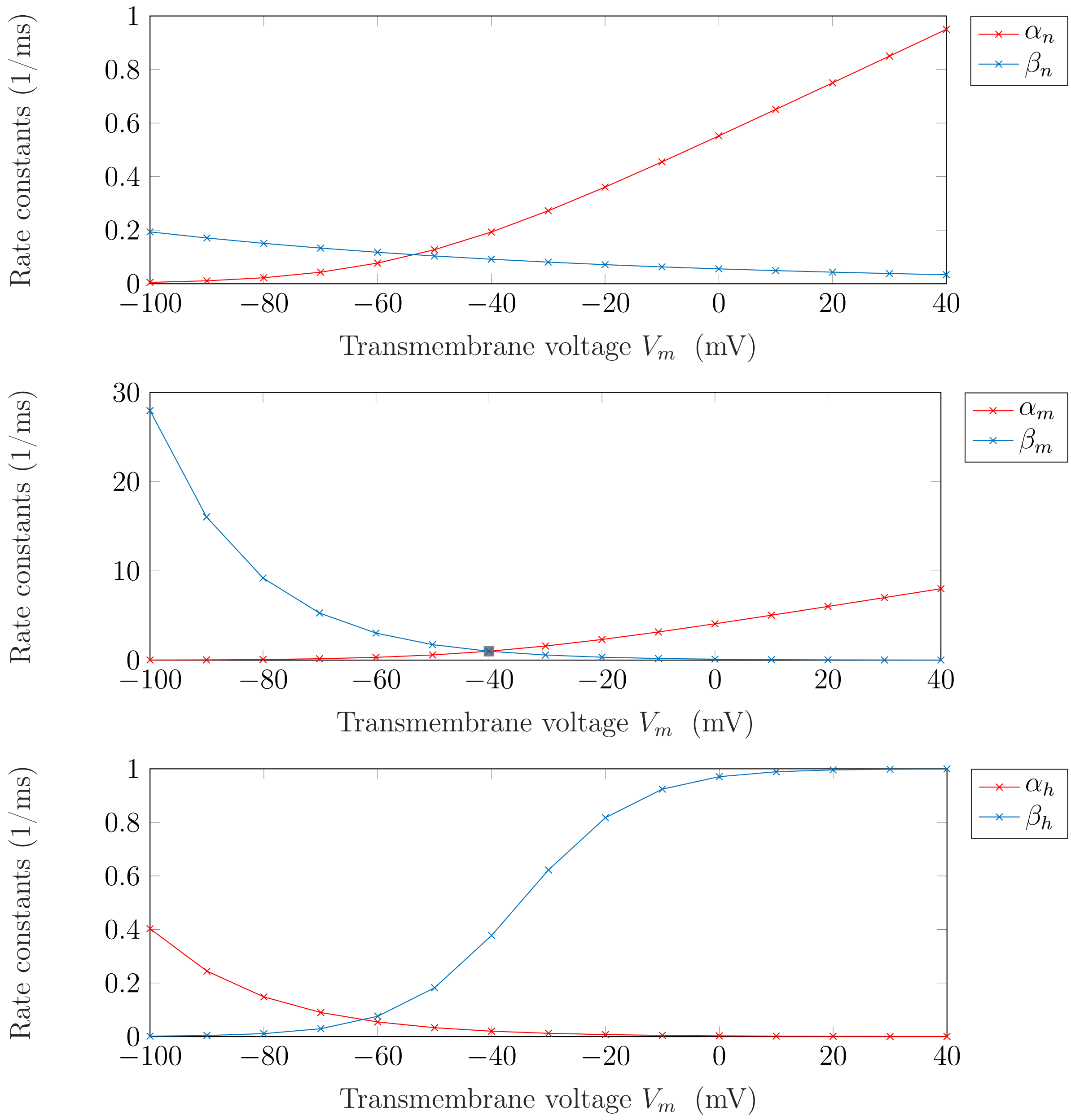


Abbildung 1: Übergangsraten α und β für n, m und h

```
clampVoltages = linspace(-100,40,15);

nVoltages = length(clampVoltages); % Anzahl an Clampspannungen
alpha_n   = zeros(nVoltages, 1);
beta_n    = zeros(nVoltages, 1);
alpha_m   = zeros(nVoltages, 1);
beta_m    = zeros(nVoltages, 1);
alpha_h   = zeros(nVoltages, 1);
beta_h    = zeros(nVoltages, 1);
```

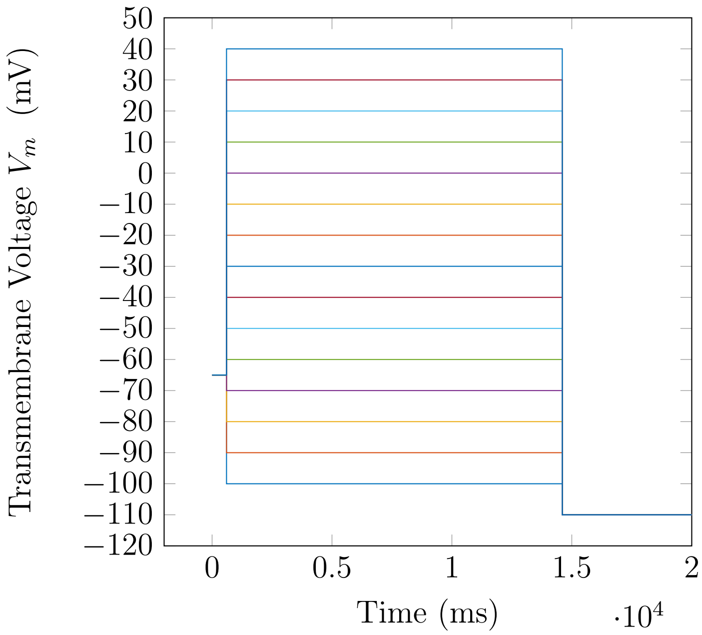
```
for iVoltage = 1:nVoltages
  alpha_n(iVoltage) = 0.01*(-(clampVoltages(iVoltage)+55)/(exp(-(
      clampVoltages(iVoltage)+55)/(10))-1));
  beta_n(iVoltage)  = 0.125*exp(-(clampVoltages(iVoltage)+65)/(80));

  alpha_m(iVoltage) = 0.1*(-(clampVoltages(iVoltage)+40)/(exp(-(
      clampVoltages(iVoltage)+40)/(10))-1));
  beta_m(iVoltage)  = 4*exp(-(clampVoltages(iVoltage)+65)/(18));

  alpha_h(iVoltage) = 0.07*exp(-(clampVoltages(iVoltage)+65)/(20));
  beta_h(iVoltage)  = 1/(exp(-(clampVoltages(iVoltage)+35)/(10))+1);
end
alpha_n(isnan(alpha_n))=1;
beta_n(isnan(beta_n))  =1;
alpha_m(isnan(alpha_m))=1;
beta_m(isnan(beta_m))  =1;
alpha_h(isnan(alpha_h))=1;
beta_h(isnan(beta_h))  =1;
```

2 Aufgabe 2

$$
\begin{aligned}
I_K &= \bar{G}_K \cdot n^4 \, (V_m - E_K) \\
I_{Na} &= \bar{G}_{Na} \cdot m^3 \, h \, (V_m - E_{Na})
\end{aligned}
$$

$$
\begin{aligned}
\dot{n} &= \alpha_n \, (1-n) \; - \beta_n \cdot n \\
\dot{m} &= \alpha_m \, (1-m) - \beta_m \cdot m \\
\dot{h} &= \alpha_h \, (1-h) \; - \beta_h \cdot h
\end{aligned}
$$

$$
\begin{aligned}
n_{i+1} &= n_i + \Delta t \cdot \dot{n}_i \\
m_{i+1} &= m_i + \Delta t \cdot \dot{m}_i \\
h_{i+1} &= h_i + \Delta t \cdot \dot{h}_i
\end{aligned}
$$



50
40
30
20
10
0
−10
−20
−30
−40
−50
−60
−70
−80
−90
−100
−110
−120
Transmembrane Voltage V_m (mV)
0
0.5
1
1.5
2
Time (ms)
$\cdot 10^4$

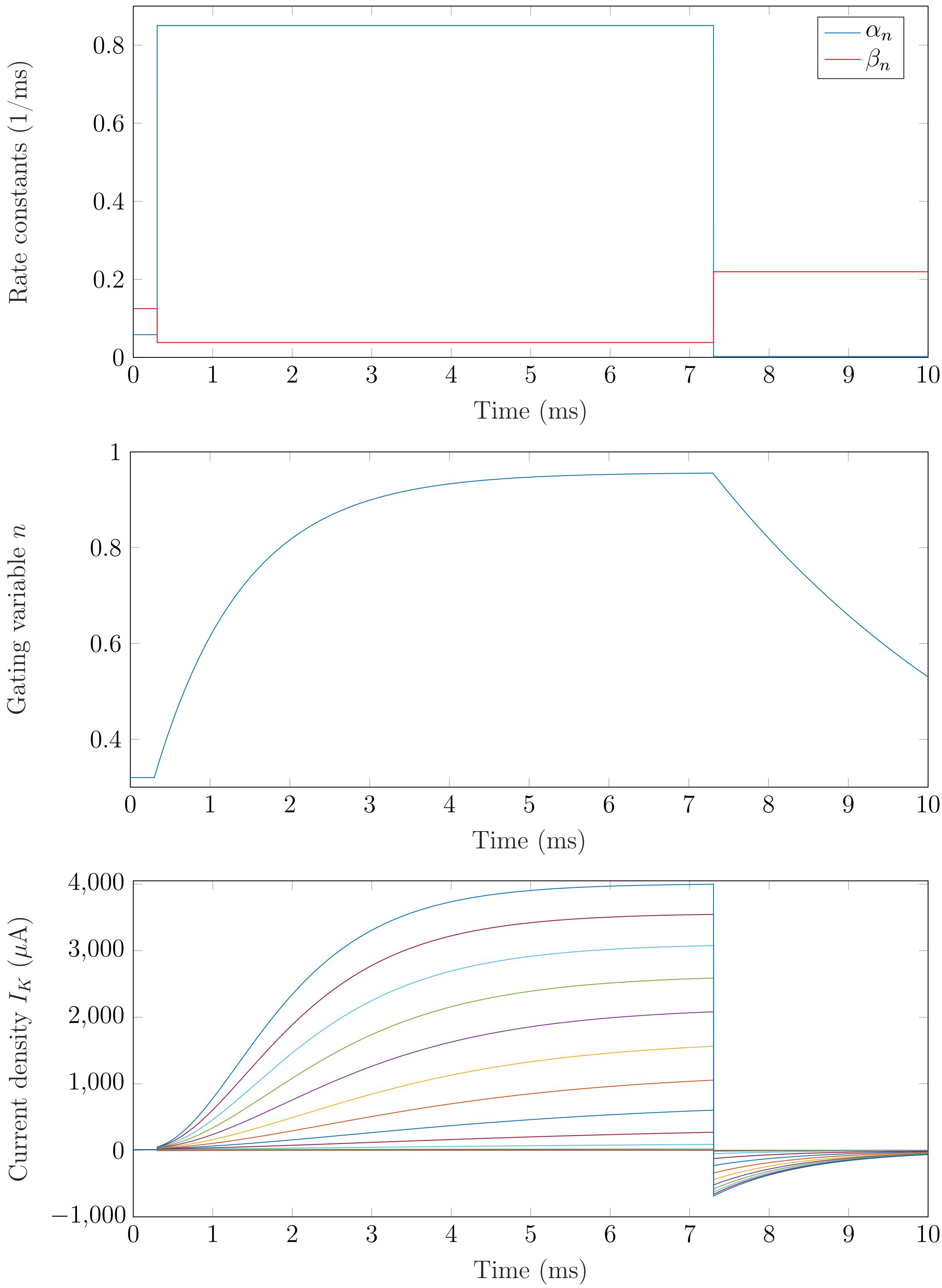


Abbildung 2: Kalium

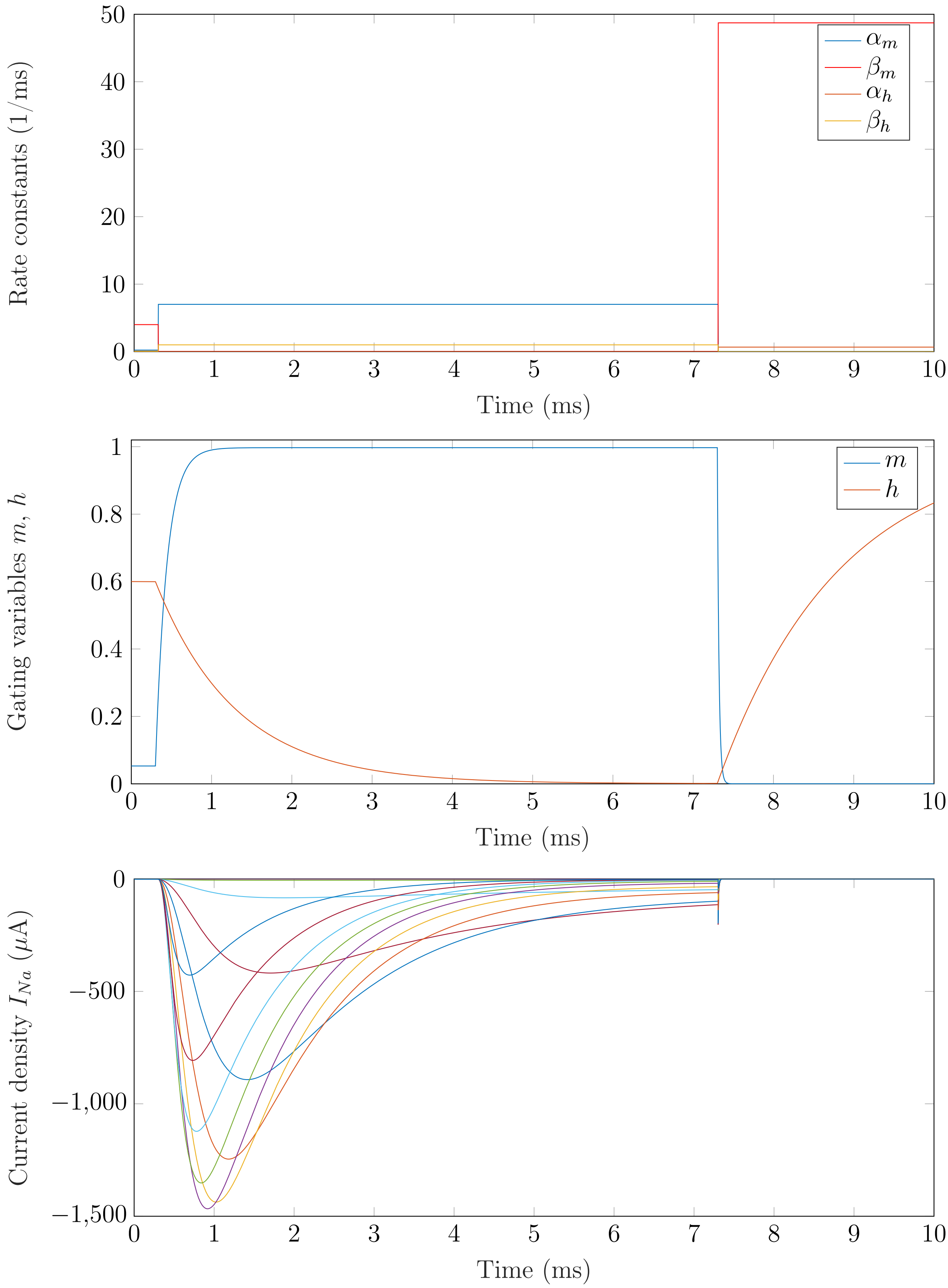


Abbildung 3: Natrium

```
% Konstanten
G_K_max     = 36;      % Maximale Kalium-Leitfaehigkeit (mS)
G_Na_max    = 120;     % Maximale Natrium-Leitfaehigkeit (mS)

% Spannungen (Nernstspannungen koennen als konstant betrachtet werden)
E_K         =  -88;    % Nernstsspannung fÃ¼r Kalium (mV)
E_Na        =   50;    % Nernstsspannung fÃ¼r Natrium (mV)
VpreStep    =  -65;    % Spannung vor Sprung (mV)
VpostStep   = -110;    % Spannung nach Sprung (mV)

clampVoltages = -100:10:40; % Protokoll fuer Voltageclamp (mV)

% Initialisiserung
n_0         = 0.32;
m_0         = 0.053;
h_0         = 0.6;

% Zeit
deltat      = 0.0005; %Zeitschritt (ms)
tend        = 10;      %Ende der Berechnung (ms)
timesteps   = 0:deltat:tend;

% Preallokation der Matrix fuer die Raten, Gates, Stroeme, und Spannung
nVoltages   = length(clampVoltages);
nTimesteps  = length(timesteps);
alpha_n     = zeros(nVoltages, nTimesteps);
beta_n      = zeros(nVoltages, nTimesteps);
n           = zeros(nVoltages, nTimesteps);
ndot        = zeros(nVoltages, nTimesteps);
I_K         = zeros(nVoltages, nTimesteps);
alpha_m     = zeros(nVoltages, nTimesteps);
beta_m      = zeros(nVoltages, nTimesteps);
m           = zeros(nVoltages, nTimesteps);
mdot        = zeros(nVoltages, nTimesteps);
alpha_h     = zeros(nVoltages, nTimesteps);
beta_h      = zeros(nVoltages, nTimesteps);
h           = zeros(nVoltages, nTimesteps);
hdot        = zeros(nVoltages, nTimesteps);
I_Na        = zeros(nVoltages, nTimesteps);
%Startwerte
n(:,1)      = n_0*ones(1,nVoltages);
m(:,1)      = m_0*ones(1,nVoltages);
h(:,1)      = h_0*ones(1,nVoltages);
```

```
%% Simulation
% Berechne Raten, Gates (Euler-1-Schritt), I_K und I_Na
for iVoltage=1:nVoltages
  for iTimestep=1:nTimesteps
    %Vm
    if (timesteps(iTimestep)<=0.3)
      Vm_current=VpreStep;
    elseif (timesteps(iTimestep)>0.3 && timesteps(iTimestep)<=7.3)
      Vm_current=clampVoltages(iVoltage);
    elseif (timesteps(iTimestep)>7.3 && timesteps(iTimestep)<=10)
      Vm_current=VpostStep;
    else
      Vm_current=NaN;
    end

    % Kalium
    I_K(iVoltage,iTimestep) = G_K_max * n(iVoltage,iTimestep).^4 * (Vm_current-E_K);

    % Natrium
    I_Na(iVoltage,iTimestep) = G_Na_max * m(iVoltage,iTimestep).^3 * h(iVoltage,iTimestep) * (Vm_current-E_Na);

    %Uebergangsraten
    alpha_n(iVoltage,iTimestep) = 0.01*(-(Vm_current+55)/(exp(-(Vm_current+55)/(10))-1));
    if(isnan(alpha_n(iVoltage,iTimestep))) alpha_n(iVoltage,iTimestep) = 1; end
    beta_n(iVoltage,iTimestep) = 0.125*exp(-(Vm_current+65)/(80));
    if(isnan(beta_n(iVoltage,iTimestep))) beta_n(iVoltage,iTimestep) = 1; end

    alpha_m(iVoltage,iTimestep) = 0.1*(-(Vm_current+40)/(exp(-(Vm_current+40)/(10))-1));
    if(isnan(alpha_m(iVoltage,iTimestep))) alpha_m(iVoltage,iTimestep) = 1; end
    beta_m(iVoltage,iTimestep) = 4*exp(-(Vm_current+65)/(18));
    if(isnan(beta_m(iVoltage,iTimestep))) beta_m(iVoltage,iTimestep) = 1; end

    alpha_h(iVoltage,iTimestep) = 0.07*exp(-(Vm_current+65)/(20));
    if(isnan(alpha_h(iVoltage,iTimestep))) alpha_h(iVoltage,iTimestep) = 1; end
    beta_h(iVoltage,iTimestep) = 1/(exp(-(Vm_current+35)/(10))+1);
    if(isnan(beta_h(iVoltage,iTimestep))) beta_h(iVoltage,iTimestep) = 1; end
```

```

    %nmh dots
    ndot(iVoltage,iTimestep) = alpha_n(iVoltage,iTimestep) .* (1-n(
        iVoltage,iTimestep)) - beta_n(iVoltage,iTimestep) .* n(
        iVoltage,iTimestep);
    mdot(iVoltage,iTimestep) = alpha_m(iVoltage,iTimestep) .* (1-m(
        iVoltage,iTimestep)) - beta_m(iVoltage,iTimestep) .* m(
        iVoltage,iTimestep);
    hdot(iVoltage,iTimestep) = alpha_h(iVoltage,iTimestep) .* (1-h(
        iVoltage,iTimestep)) - beta_h(iVoltage,iTimestep) .* h(
        iVoltage,iTimestep);

    if(iTimestep < nTimesteps)
      n(iVoltage,iTimestep+1) = deltat * ndot(iVoltage,iTimestep) +
          n(iVoltage,iTimestep);
      m(iVoltage,iTimestep+1) = deltat * mdot(iVoltage,iTimestep) +
          m(iVoltage,iTimestep);
      h(iVoltage,iTimestep+1) = deltat * hdot(iVoltage,iTimestep) +
          h(iVoltage,iTimestep);
    end
  end
end
```