Online Resource 5 - Summary of the code used to simulate an individual MCN.

The code has to be launched with the Scilab software (version 4.1.2).

```
// ****************************************
// Vasopressinergic Magnocellular neuron
// Louis Nadeau - Louis.Nadeau.3@ulaval.ca
// Initially based on Magno-0.45.sce but
// modified to include vasopressin secretion
// ****************************************

function Simulate(Osmo,Time,Injected_I,Ratio_D,Ratio_PSP_Freq,Ratio_PSP_Amp,Seed,Path_to_File)
ieee(2);            //Added so that log10(0) returns -inf instead of an error of singularity
lines(0);           //Added so that its not necessary to press y or n after each page of output from scilab
stacksize(40000000);    //Increase the memory of scilab
rand("seed",Seed);      //Replace the random seed by a fixed one in order to obtain the exac same
                        simulation each time, enable database reus

//Parameters of the simulation related to time (in milisecond)
Resolution=Time*240;            //Number of calculated point in the simulation
delta_t=Time/Resolution;        //Calculation of the time step of the simulation
delta_t_half=delta_t/2;         //To save computation time later in the code delta_t/2 is done only once
delta_t_sixth=delta_t/6;        //To save computation time later in the code delta_t/6 is done only once

// Initial conditions and initialization of the simulation
V=-60;                          //Initial membrane potential
SpikeRecord=[];                 //Record each spike position
last_V=V;                       //The value of V in the last time step
Nb_Of_Currents=9;               //Number of currents
D=9;                            //Setting D to its initial concentration so it does not take forever to stabilize
Instant_Freq=0;                 //Instantaneous frequency based on the two last spikes
I=zeros(Nb_Of_Currents,1);      //Initialize the currents vector

//Concentration initialization
Na_Internal=21.3;                       //Internal Concentration of sodium (from richard1995)
//Na_Internal=3; //Oliet1993
Na_External=135;                        //Zhang2007
//Na_External=138.9;//Richard1995)//Na_External=120;//Oliet1993
K_Internal=135;                         //Internal Concentration of potassium (from richard1995)
//K_Internal=150; //Oliet1993
K_External=3;                           //Zhang2007 //K_External=3;//External Concentration of
potassium (from richard1995)//K_External=3; //Oliet1993
Ca_External=4000000;                    //External concentration of calcium in nM (Roper2004)
Ca_Internal=113;                        //Internal concentration of calcium in nM (Roper2004)
Ca=[Ca_Internal;Ca_Internal;Ca_Internal];//Setting Ca to its resting concentration
Delta_Sugar=Osmo;                       //Change in the osmolality cause by an increase in sugar (mannitol)
ENa=26.4*log(Na_External/Na_Internal);  //Reversal potential of sodium current computed with the Nerst
Equation at 33 celcius (average papers)
EK=26.4*log(K_External/K_Internal);     //Reversal potential of potassium current computed with the Nerst
Equation at 33 celcius
```

```
PNaKSic=0.182+(Na_External-105)*0.0021333;//Calculation of the atypical permeability ratio of the SIC channels
//E is set to its steady-state parameters
E=[ENa;12.5*log(Ca_External/Ca(1));EK;EK;EK;EK;ENa;EK;26.4*log((PNaKSic*Na_External+K_External)/(PNaKSic*Na_Internal+K_Internal))];
```

//Synaptic initialization

```
EPSP_Frequency=3.53*Ratio_PSP_Freq;//The average excitatory synaptic input is 3.53Hz in control condition
IPSP_Frequency=4.00*Ratio_PSP_Freq;//The average inhibitory synaptic input 4hz is in control condition
EPSP_Multiplicator=2.39E-22*(289+Delta_Sugar)^8.79;//When the osmolarity increases, the number of EPSP increases based on Richard 1997
IPSP_Multiplicator=1;//The IPSP frequency may not change at the same speed as the EPSP frequency
EPSP_Probability=EPSP_Frequency*EPSP_Multiplicator*Time/1000/Resolution;//Probability of receiving an excitatory input
IPSP_Probability=IPSP_Frequency*IPSP_Multiplicator*Time/1000/Resolution;//Probability of receiving an inhibitory input
ISynaptic=0;                    //Set the synaptic input to 0 at the beginning
```

//Current initialization

```
Delta_Osmo=Delta_Sugar;
if Delta_Osmo<-25 then Delta_Osmo=-25; end; //added because there is a saturation below 270 mmol/L starting from 295 mmol/L in the experimental data
//for the main loop, m_infinity is compute in two lines to improve speed. Current 1,2,3,4 and 9 have the same form and the use of vectorization save a lot of computing time
m_infinity(1:4)=(1)./(1+exp((last_V+[38;-10;-2;45])./[-4;-7;-11;-11]));
m_infinity(5:9)=[1/(1+exp(-1.120-2.508*log10((Ca(3)-Ca_Internal)/1000)));1/((1+470/Ca(2)^2.38)*(1+exp((-last_V-140*log10(Ca(2))+370)/7.4)));1;tanh((Ca(1)-Ca_Internal-20*D)/55);Delta_Osmo*0.0297+1];
//Here t_m and t_h are initialize and all their elements are set to their steady-state. In the main loop, only the non-static elements are updated.
t_m=[%inf;1/(0.19*(-last_V+19.88)/(exp((-last_V+19.88)/10)-1)+0.046*exp(-last_V/20.73));8/(exp((last_V+40)/19)+exp((-last_V-40)/20))+1.8;35/(1.1*exp((last_V+43)/17)+exp((-last_V-20)/13));%inf;1.22;%inf;75;35000];
t_h=[20/((1+exp((last_V-1)/14))*(1+exp((last_V+30)/34)))-0.075;%inf;%inf;5+120/(exp((last_V+75)/30)+exp((-last_V-45)/18));%inf;%inf;%inf;%inf;%inf];
//Since many elements of h_infinity have a value of 1, the fast "ones()" function is used to set the correct size of h_infinity, afterward (and in the main loop) only the 3 currents with a dynamic inactivating parameters are updated. Here we set those 3 currents to their steady-state
h_infinity=ones(9,1);
h_infinity([1;4;8])=(1)./(1+exp((last_V+[45;80;70])./[2;6.5;-7]));
//Sets the m and h parameters to their steady-state value
m=m_infinity;
h=h_infinity;
```

//Main loop, this loop and its functions do not contain comments since they are evaluated by scilab each time and its a waste of time

```
for i=2:Resolution
  m_infinity(1:4)=(1)./(1+exp((last_V+[38;-10;-2;45])./[-4;-7;-11;-11]));
  m_infinity(5:9)=[1/(1+exp(-1.120-2.508*log10((Ca(3)-Ca_Internal)/1000)));1/((1+470/Ca(2)^2.38)*(1+exp((-last_V-140*log10(Ca(2))+370)/7.4)));1;tanh((Ca(1)-Ca_Internal-20*D)/55);Delta_Osmo*0.0297+1];
  t_m(2:4)=[1/(0.19*(-last_V+19.88)/(exp((-last_V+19.88)/10)-1)+0.046*exp(-last_V/20.73));8/(exp((last_V+40)/19)+exp((-last_V-40)/20))+1.8;35/(1.1*exp((last_V+43)/17)+exp((-last_V-20)/13))];
```

```
 h_infinity([1;4;8])=(1)./(1+exp((last_V+[45;80;70])./[2;6.5;-7]));
 t_h(1:4)=[20/((1+exp((last_V-1)/14))*(1+exp((last_V+30)/34)))-
0.075;%inf;%inf;5+120/(exp((last_V+75)/30)+exp((-last_V-45)/18))];
 E(2)=12.5*log(Ca_External/Ca(1));
 E(9)=26.4*log((PNaKSic*Na_External+K_External)/(PNaKSic*Na_Internal+K_Internal));
 Ca=Ca+delta_t*(([-0.9;-100;-1.6]*I(2))-(Ca-Ca_Internal)./[2330;1;656]);
 I=[14;0.1;14;14;0.19;1;0.0169095;0;0.023546].*(m.^[3;2;3;4;2;1;1;1;1]).* h .*(last_V-E);
 I(8)=0.0745*(0.6+0.4*(1-m(8)*h_infinity(8)))*(last_V-E(8));
 m=m+delta_t*(m_infinity-m)./t_m;m([1;5;7])=m_infinity([1;5;7]);
 h([1;4])=h([1;4])+delta_t*(h_infinity([1;4])-h([1;4]))./t_h([1;4]);
 V=last_V-delta_t*(Injected_I+ISynaptic+sum(I));
 if (V>0 & last_V<0 & (V-last_V)>0) then
  SpikeRecord($+1)=i;
  if length(SpikeRecord)>1 then
   Instant_Freq=1000*Resolution/(Time*(i-SpikeRecord($-1)));
   if Instant_Freq>15 then
        Freq_Effec=1.053-((1)./(1+exp((Instant_Freq-22.5)/-2.5)));
   else
        Freq_Effect=1;
   end;
  else
  Freq_Effect=1;
  end;
  D=D+Ratio_D*Freq_Effect*(0.001+D*0.06);
 end;
 D=D*(1-delta_t/5000);
 last_V=V;
 ISynaptic=ISynaptic-delta_t*ISynaptic/7.5;
 if rand()<EPSP_Probability then ISynaptic=ISynaptic+Ratio_PSP_Amp*0.08*(last_V+38); end;
 if rand()<IPSP_Probability then ISynaptic=ISynaptic+Ratio_PSP_Amp*0.04*(last_V+72); end;
end;
//Show the results
Freq=length(SpikeRecord)/Time*1000;
SpikeRecord=SpikeRecord*delta_t;
save(Path_to_File,SpikeRecord,Freq);
endfunction;
```

//Excute the argument when loaded through the scilab bacth mode on the command line

```
arg=sciargs();
Simulate(evstr(arg(6)),evstr(arg(7)),evstr(arg(8)),evstr(arg(9)),evstr(arg(10)),evstr(arg(11)),evstr(arg(12)),
arg(13));
quit;
```