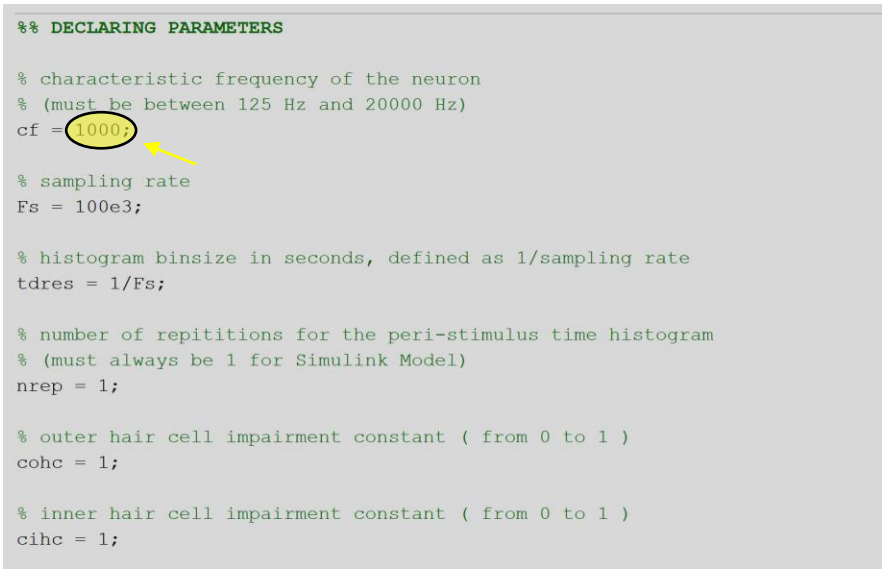
Auditory Nerve Simulink Model – Modify Model Instructions

Description: This set of instructions outlines various ways in which the user can modify the functionality of the model, including instructions on editing model parameters, input signals, model verification, and output display settings.

- Edit the Characteristic Frequency of the Neuron:
  The default is set to model a neuron with a characteristic frequency of 1000 Hz. If the user desires to change this or another model setting, open the 'AN_Set_Model_Parameters' MATLAB script. Then simply adjust the characteristic frequency to the desired value, as shown below.



```
%% DECLARING PARAMETERS

% characteristic frequency of the neuron
% (must be between 125 Hz and 20000 Hz)
cf = 1000;

% sampling rate
Fs = 100e3;

% histogram binsize in seconds, defined as 1/sampling rate
tdres = 1/Fs;

% number of repititions for the peri-stimulus time histogram
% (must always be 1 for Simulink Model)
nrep = 1;

% outer hair cell impairment constant ( from 0 to 1 )
cohc = 1;

% inner hair cell impairment constant ( from 0 to 1 )
cihc = 1;
```

If the user wishes to change some of the other overall model input parameters like impairment constants or refractory time constants, simply change them in the 'AN_Set_Model_Parameters' script as well. It is important to keep in mind that changing the parameters in this script may have a dramatic effect on model functionality and performance.

- Edit Other Model Parameters:
  If the user wishes to change calculations for function block-specific parameters, changes can be made to the 'AN_Get_Model_Parameters' script, which calculates all parameter values not declared in the overall model input. An example of some function block-specific parameters is shown below.

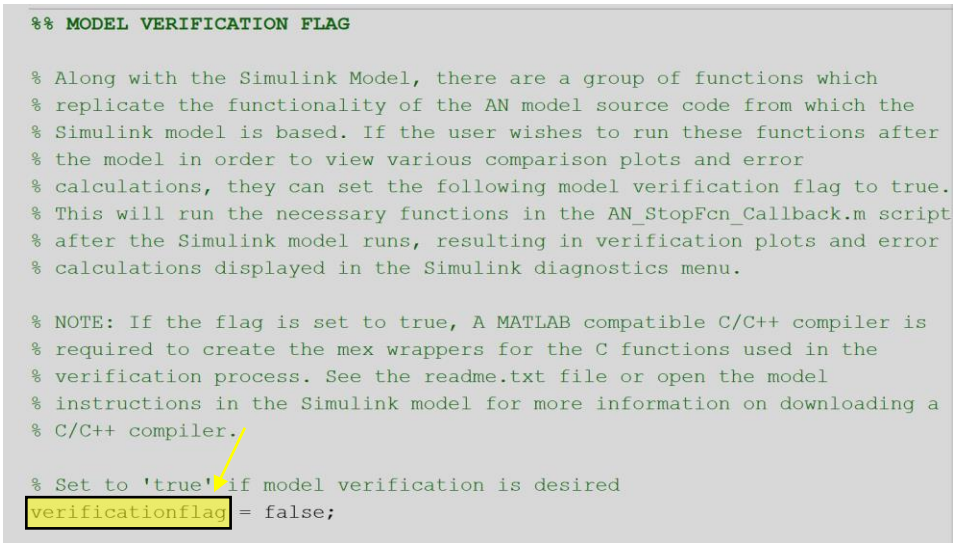
```
%% IHC LP FILTER PARAMETERS

% Function Inputs:
% (Hard-coded in C source code, but set as parameters for model)
Fcihc = 3000;
gainihc = 1.0;
orderihc = 7;

% Calculated Constants:
Cihc = 2.0/tdres;
c1LPihc = ( Cihc - 2*pi*Fcihc ) / ( Cihc + 2*pi*Fcihc );
c2LPihc = 2*pi*Fcihc / (2*pi*Fcihc + Cihc);

%% IHC NL LOGARITHMIC FUNCTION PARAMETERS

% Function Inputs:
% (Hard-coded in C source code, but set as parameters for model)
C1slope = 0.1;
C2slope = 0.2;
corner = 80;
strength = (20.0e6)/(10^(corner/20));

%% C2 FILTER PARAMETERS

% Calculate C2 filter coefficients and normalizing gain by calling MATLAB
% function
[C2coeffs, norm_gainc2] = C2Coefficients( tdres, cf, bmTaumax, 1/ratiobm );

%% C1 FILTER PARAMETERS

% Calculate the constant parameters for the C1 filter, including initial
% pole/zero locations, shifting constants, and gain constants
```

Again, note that these parameter calculations were implemented to replicate those found in the AN model source code. Thus, changing them may significantly affect model functionality and performance.

- Change the Input Signal:
  The default input signal is set to replicate the signal defined in the AN model source code (a short-duration stimulus with the same frequency as the characteristic frequency of the neuron). In order to change the input signal, visit the bottom of the 'AN_Get_Model_Parameters' MATLAB script. The input signal is defined as the vector 'RxSignal' as shown below.

```
%% TEST SIGNAL

% Closely Matching Source Code:

% stimulus parameters
stimdb = 60; % stimulus intensity in dB SPL
F0 = cf;     % stimulus frequency in Hz
T  = 50e-3;  % stimulus duration in seconds
rt = 2.5e-3; % rise/fall time in seconds
ondelay = 10e-3;

% PSTH parameters
psthbinwidth = 1e-4; % binwidth in seconds;
psthbins = round(psthbinwidth*Fs);  % number of psth bins per psth bin

t = 0:1/Fs:T-1/Fs; % time vector
mxpts = length(t);
irpts = rt*Fs;
onbin = round(ondelay*Fs);

pin = zeros(1,onbin+mxpts);

pin(onbin+1:onbin+mxpts) = sqrt(2)*20e-6*10^(stimdb/20)*sin(2*pi*F0*t); % unramped stimulus
pin(onbin+1:onbin+irpts)= pin(onbin+1:onbin+irpts).*(0:(irpts-1))/irpts;
pin(onbin+(mxpts-irpts):onbin+mxpts)=pin(onbin+(mxpts-irpts):onbin+mxpts).*(irpts:-1:0)/irpts;

% Assign to simulink input signal
RxSignal = zeros(1,10/3*length(pin));
RxSignal(1:length(pin)) = pin;

% Simulink simulation test time
testtime = length(RxSignal)/Fs;
```

- Verify the Model Against Source Code Functions:
  If the user changes the Simulink Model, or simply desires to compare the Simulink Model outputs to outputs given from a string of isolated source code functions, enable the verification flag in the 'AN_Set_Model_Parameters' script. This is done by simply setting the 'verificationflag' Boolean to 'true'. This will cause a variety of C and MATLAB functions to run after the Simulink model. Multiple comparison plots will be displayed, along with error calculations found in the Simulink diagnostics menu.



```
%% MODEL VERIFICATION FLAG

% Along with the Simulink Model, there are a group of functions which
% replicate the functionality of the AN model source code from which the
% Simulink model is based. If the user wishes to run these functions after
% the model in order to view various comparison plots and error
% calculations, they can set the following model verification flag to true.
% This will run the necessary functions in the AN_StopFcn_Callback.m script
% after the Simulink model runs, resulting in verification plots and error
% calculations displayed in the Simulink diagnostics menu.

% NOTE: If the flag is set to true, A MATLAB compatible C/C++ compiler is
% required to create the mex wrappers for the C functions used in the
% verification process. See the readme.txt file or open the model
% instructions in the Simulink model for more information on downloading a
% C/C++ compiler.

% Set to 'true' if model verification is desired
verificationflag = false;
```

- Edit Model Output Plots:
  Model output plots can be changed by editing the 'AN_StopFcn_Callback' MATLAB script, as shown below.

```
%% SIMULINK OUTPUT PLOTS

if verificationflag == true % only execute verification is flag is true

    figure('units','normalized','outerposition',[0 0 0.5 1])

    subplot(3,1,1)
    stem(out.sptime(1:end-1));
    title('Spike Train Output (sptime) - Simulink Model');
    ylabel('Spike Indicator');
    xlabel('Sample Number');

    subplot(3,1,2)
    plot(out.trd_vector(1:end-1));
    title('Mean Synaptic Redocking Time - Simulink Model');
    ylabel('Redocking Time [sec]');
    xlabel('Sample Number');

    subplot(3,1,3)
    plot(out.meanrate(1:end-1));
    title('Mean of Spike Rate - Simulink Model');
    ylabel('Mean Rate [/s]');
    xlabel('Sample Number');

else

    figure('units','normalized','outerposition',[0 0 1 1])

    subplot(3,2,1)
    plot(out.meout(1:end-1))
    title('Middle Ear Filter Pressure Output (meout) - Simulink Model')
    ylabel('Pressure [Pa]');
    xlabel('Sample Number');
```