Abstract:
The metabotropic glutamate receptor 7 (mGluR7) is a G-protein coupled receptor that activates the $G_{i/o}$ α-subunit. In addition to its neuroprotective roles and long-term depression mechanisms, this receptor has been shown to elicit temporal learning mechanisms in cerebellar Purkinje Cells (PCs). The basis for storage and utilization of temporal memories in the cerebellum occurs through PC firing patterns. PCs give inhibitory inputs to nucleus cells, which output to higher-order structures and thus act on temporally conditioned stimuli. mGluR7 has been shown to elicit an intracellular mechanism in PCs that encodes specific firing patterns, likely via microtubule dynamics. While temporal memory simulations emulating the cerebellum have been made using network dynamics, none have investigated how an intrinsic mechanism could affect this encoding. This paper investigates if adding an intrinsic mechanism to the PCs in a modern cerebellum simulation will allow for greater network capacity and a closer representation of the actual cerebellum.

Methods:
The code utilized for this study was developed in C++ in order to prevent runtime limitations and to perform efficient multithreading. The architecture for the firing patterns was based on a LIF model where each action potential sends a signal that a spike occurred to its connected outputs as the voltage returns to 0. *(10/2/22)*

For the purposes of the GitHub
The actual simulation is held within the NuevaTemporal2.cpp file. When you compile and run it, you will be required to input 5 variables. The following are the variables. Note this simulation is without the proposed Intrinsic mGluR mechanism

- Test Number → wherever you run this program it will save an output file of the firing rates of neuron "N" at the time "t". Additionally, the program will also create a file of the final weights for each neuron.

- Conductance Decay Constant → According to the following equation from (Medina et al 2000) representing the change of synaptic conductance:

$$\frac{dg\ \mathrm{syn}}{dt} = \sum_{i=0}^{\mathrm{inputs}} S_i \cdot w_i \cdot (1 - g\ \mathrm{syn}) - g\ \mathrm{syn}\ \tau\ \mathrm{syn}\ ,$$

  The decay constant is represented by $\boldsymbol{\tau_{syn}}$ (should be around 0.05 to 0.3)

- Threshold Decay Constant → $\boldsymbol{\tau_{thresh}}$ for the leaky integrate and fire system:

$$d(\mathrm{thresh})/dt = \mathrm{thresh} - \mathrm{thresh}*\tau_{thresh}$$

- Chloride Decay Constant → The current system in place involves a float representation of the chloride concentration in the learning neurons. This is represented in the same manner as the threshold decay constant:

$$d(\mathrm{Chlor})/dt = \mathrm{Chlor} - \mathrm{Chlor}*\tau_{Chlor}$$

- Voltage Leak → The following represents the value of the voltage leak per iteration of the simulation. Multiple papers suggest 0.07 for this value, but I am still running tests to see if other values may work better

The code for this input is displayed as such within NuevaTemporal2.cpp

```
cin >> testnumb;
cin >> DecCund;
cin >> DecThresh;
cin >> DecChlor;
cin >> VoltageLeak;
```

Optimizing this process is lengthy and requires running multiple simulations. I provide a very rough way to do this in the CereRunTrain.py file provided in GitHub. This program has not been optimized yet and is just for easy testing. *(6/30/23)*

References:

Javier F. Medina, Keith S. Garcia, William L. Nores, Nichole M. Taylor, Michael D. Mauk., (2000) Timing Mechanisms in the Cerebellum: Testing Predictions of a Large-Scale Computer Simulation, J of Neurosci, 20 (14) 5516-5525; DOI: 10.1523