The integrate-and-fire neuron models

By: Kamal Abu-Hassan

1. Introduction

This document provides a description of a computational implementation of integrate-and-fire neuron models in C++ programming language and an example simulation of the QIF model.

2. Computer programs of the model

The C++ project for simulating the model is provided in the below Figures 2 - 6 (with inline comments).

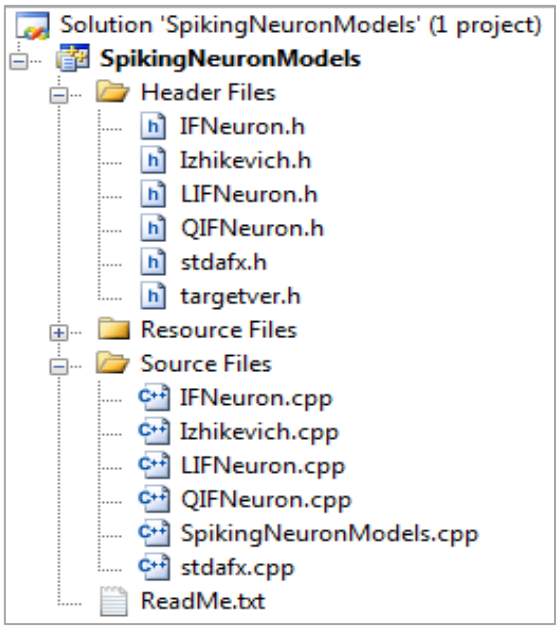

Figure 1. The structure of the C++ project.

```
#include "stdafx.h"
#include "IFNeuron.h"
#include "LIFNeuron.h"
#include "QIFNeuron.h"
#include "Izhikevich.h"

void main()
{
	/*
	simulating QIF neuron model using system-defined parameters
	(see QIFNeuron.cpp & QIFNeuron.h)
	*/
	QIFNeuron();

	/*
	simulating QIF neuron model using user-defined parameters
	*/
	// the below variables are explained in the header file (QIFNeuron.h)

	float C=1, k=0.02,  V_peak=10, dt = 1,
		V_th=-40, V_init=0, V_reset=-80;
	int tstop=100;
	char typeofCurrent='f';
	float fixedCurrent = 10;

	QIFNeuron( tstop,  dt,  V_th, V_init,  V_reset,
		&fixedCurrent,  typeofCurrent,  C, k,
		V_peak,V_reset);

}
```

Figure 2. The C++ application file 'SpikingNeuronModels.cpp' is the first class to be invoked by the runtime environment.

```
#include "stdafx.h"
#include <stdio.h>
#include <stdlib.h>
#include <iostream>
using namespace std;

class QIFNeuron
{
public:
	QIFNeuron(void);//default constructor
	QIFNeuron::QIFNeuron(
		int tstop, // model time in milliseconds
		float dt, // time step
		float V_th, // firing threshold
		float V_init, // initial value of the voltage
		float V_reset, // resting voltage
		float* I1, // input current (nA)
		char typeofCurrent, // type of current;
				//'f' for fixed or 'v' for variant
		float C,//capacitance
		float k,//k >0 is a parameter
		float V_peak,//To avoid simulating 'infinity', the voltage is
			//clipped at some sufficiently large value, V_peak
		float c //after-spike resetting potential
		);
	~QIFNeuron(void);//destructor
};
```

Figure 3. The C++ header file 'QIFNeuron.h' holds declarations for the C++ code file 'QIFNeuron.cpp'

```
#include "QIFNeuron.h"

QIFNeuron::QIFNeuron(void)
{
	float C=1, k=0.02,  V_peak=10, dt = 1,
		V_th=-40, V_init=0, V_reset=-80;
	int tstop=100;
	char typeofCurrent='f';
	float fixedCurrent = 10;

	QIFNeuron( tstop,  dt,  V_th, V_init,  V_reset,
		&fixedCurrent,  typeofCurrent,  C, k,
		V_peak,V_reset);

}

QIFNeuron::QIFNeuron(int tstop, float dt,float V_th, float V_init,float V_reset,float* I1,
                                    char typeofCurrent, float C,float k,float V_peak, float c){
	FILE   *fs= fopen("QIF_voltages.dat","w");
	int t;
	float tsteps=0;//the integration steps
	float v=V_init;
	float * I2; // the content of I1 will be copied to this variable
	I2 = new float [tstop/dt];// dynamic memory allocation
	//check if variant current; then the address of I1 is assigned to I2
	if (typeofCurrent=='v')
		I2 = I1;
	//otherwise; I1 is a pointer to one value (dereference: *I1)
	//that represents the fixed current
	else for (t=0;t<tstop/dt;t++) I2[t]=*I1;

	//write the integration steps to the output file
	//for plotting purposes
	for (tsteps=0;tsteps<tstop;tsteps=tsteps+dt)
		fprintf(fs, "%lf  ", tsteps);

	fprintf(fs, "\n");

	// (tstop/dt)= the total number of integration steps
	for (t=0;t<(tstop/dt);t++)
	{
		if (v>=V_peak)                                    // PREVENT INFINITY
		{
			v = c;                                 // voltage reset
		}

		//implementation of the voltage equation
		 v += dt*(k*(v-V_reset)*(v-V_th)+I2[t]) /C;

		//reset to the peak value (V_peak)
		if (v>=V_peak) v=V_peak;
		fprintf(fs, "%lf  ", v);

	}
	fclose(fs);
}

QIFNeuron::~QIFNeuron(void)
{
}
```

Figure 4. The default and parameterized constructors in the C++ code file 'QIFNeuron.cpp'. The modeller can invoke the default constructor to get an example simulation of the model.

```
f='C:\VCProjects\SpikingNeuronModels\SpikingNeuronModels\QIF_voltages.dat';
load(f)
%QIF_voltages(1,:) represents time in milliseconds
%QIF_voltages(2,:) represents the output voltage trace
plot(QIF_voltages(1,:),QIF_voltages(2,:))
xlabel('Time (ms)')
ylabel('Membrane voltage (mV)')
```

Figure 5. MATLAB script 'plotCplusresults.m' to plot the output of the C++ program.

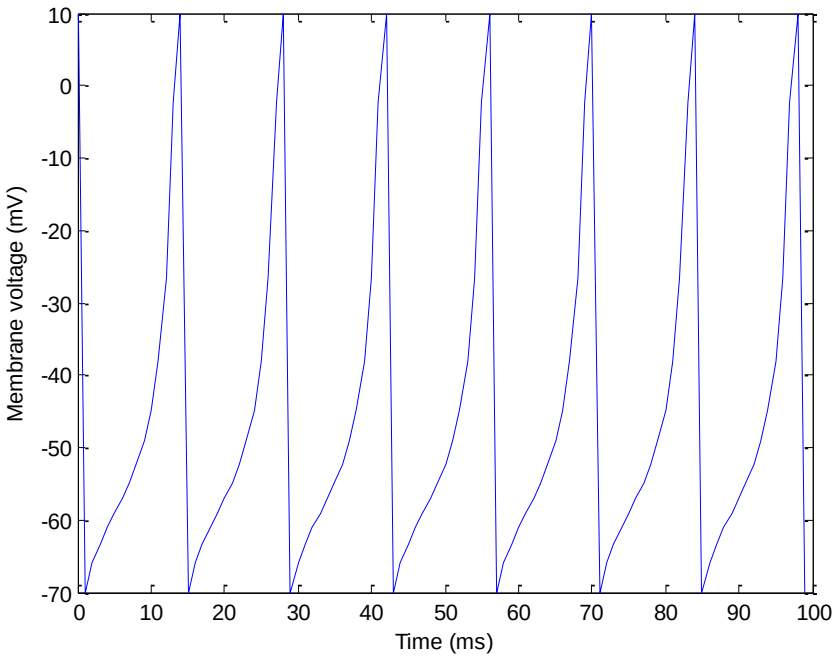

Figure 6. Voltage trace of the simulated model with a fixed input current. The plot can be reproduced using the MATLAB code ‘plotCplusresults.m’