Parameter Optimization and Nonlinear Fitting for Computational Models in Neuroscience on GPGPUs

Manjusha Nair [1,2], Krishna Subramanyan[1],
[1] Amrita School of Engineering,
Amrita Vishwa Vidyapeetham (Amrita University),
Amritapuri Campus, Clappana P O, Kollam, Kerala,
India- 690525.
manjushanair@am.amrita.edu

Manjusha Nair [1,2], Bipin Nair [2], Shyam Diwakar [2],
[2] Amrita School of Biotechnology, Amrita Vishwa Vidyapeetham (Amrita University), Amritapuri Campus, Clappana P O, Kollam, Kerala, India-690525.
manjushanair@am.amrita.edu,
shyam@amrita.edu

***Abstract–*One of the main challenges in computational modeling of neurons is to reproduce the realistic behaviour of the neurons of the brain under different behavioural conditions. Fitting electrophysiological data to computational models is required to validate model function and test predictions. Various tools and algorithms exist to fit the spike train recorded from neurons to computational models. All these require huge computational power and time to produce biologically feasible results. Large network models rely on the single neuron models to reproduce population activity. A stochastic optimization technique called Particle Swam Optimisation (PSO) was used here to fit spiking neuron model called Adaptive Exponential Leaky Integrate and Fire (AdEx) model to the firing patterns of different types of neurons in the granular layer of the cerebellum. Tuning a network of different types of spiking neurons is computationally intensive, and hence we used Graphic Processing Units (GPU) to run the parameter optimisation of AdEx using PSO. Using the basic principles of swam intelligence, we could optimize the n-dimensional space search of the parameters of the spiking neuron model. The results were significant and we observed a 15X performance in GPU when compared to CPU. We analysed the accuracy of the optimization process with the increase in width of the search space and tuned the PSO algorithm to suit the particular problem domain. This work has promising roles towards applied modeling and can be extended to many other disciplines of model-based predictions.**

***Keywords*- Model Fitting; Parameter Optimization; Adaptive Exponential Leaky Integrate and Fire Model; Particle Swam Optimization; Graphic Processing Units.**

I. INTRODUCTION

Computational neuroscience addresses the problem of describing the behavior of biological neurons using mathematical models over a wide range of details and accuracy[1]. Modeling in neuroscience requires parameter estimation via data fitting in order to reproduce spike patterns statistically significant with electrophysiological recordings. The parameter estimation problem was formulated here as an optimization problem including the efficient use of data which aids modeling. Different techniques such as Maximum Likelihood estimation, least–square fitting, regression analysis, clustering [2] etc exist to fit linear and non linear models to data. Due to the lack of predictability of nonlinear optimization techniques, different algorithms based on social behavior and evolutionary computing such as genetic algorithms [3], particle swam optimization algorithm etc. [4] also attracted considerable attention in this field [5]. It has been observed that the PSO algorithm is consistent in locating the area of the global optimum in all constrained non linear optimization problems[6].

The spiking neuron model used here for modeling large network of biological neurons-Adaptive exponential leaky integrate and fire model [7] is versatile and can quantitatively reproduce the firing and some of the bursting dynamics of different classes of neurons. Since our model spiking neuron included quadratic equations with exponential terms, the upswing of the voltage and the threshold reset conditions impose constraints in the problem area that we need to address and adjust the objective function accordingly. PSO has been used to produce the spiking models for the different neurons of the cerebellum [8]. We used a modified PSO algorithm since all the solutions of the search do not result in a feasible solution due to the dynamics of the AdEx equations.

PSO requires increasing the number of iterations and number of particles to increase the probability of getting more accurate results when we perform completely blind search in the search space such as tuning the model to spike trains recorded from neurons of unknown type. This increases the total execution time of the algorithm, when run in CPU. Hence parallel processing techniques based on Graphic Processing Units (GPU) [9] were used here to increase the efficiency while reducing the execution time.

II. METHODS

A. Particle Swam Optimization

PSO, which is a biologically motivated algorithm makes no assumptions about the model and is suitable for irregular and noisy problems even though it does not guarantee a solution always. It performs *n*- dimensional space calculations for *m* particles carried out over a series of *T*- time steps. Synchrony of movement among the particles is achieved through simple rules from social behavior and stochastic variables in the algorithm. It uses stochastic search patterns in the solution space based on global information as in population based search algorithms and has an iterative evolution where the position of the particles are influenced by its own experience and knowledge of its neighbors. The potential solutions or particles move through a hyper-dimensional search space during the different iterations of the algorithm and the change in position of each particle called velocity is stochastically accelerated towards its best position in the previous step and towards the best position of its neighbors. Velocities were adjusted according to their difference, per dimension from best locations. Using autobiographical memory (pBest) and publicized knowledge (gBest) results in reduced wandering of particles in the search space. Since the relative importance of these two factors can vary from one step to another, we apply random weights to each part. We modeled multi dimensional- non linear search to optimize the nine parameters of the spiking neuron model AdEx and use *M* x *N* matrices, where *M* is the number of particles and *N* is the number of dimensions. The algorithm can be summarized as follows.

```
For each particle $P_i$ at time t=0
{
    Initialize $\vec{X}_i(0)$ and $\vec{V}_i(0)$ with random values within the
    hyperspace such that the swam with K particles is P(0) = $\{P_1, P_2, \dots, P_k\}$
}
Do
{
    For each particle $P_i$ for each time step t,
    {
        Calculate the fitness value $F(\vec{X}_i(t))$ at time t.
        If $F(\vec{X}_i(t))$ better than $pbest(i)$, then
            $pbest(i) = F(\vec{X}_i(t))$
            $\vec{X}_{pbest(i)} = \vec{X}_i(t)$
        End
    }
    $gbest =$ particle with the best $F(\vec{X}_i(t))$ of all particles
        $\vec{X}_{gbest(i)} = \vec{X}_i(t)$

    For each particle $P_i$ for each time step t,
    {
        Update the velocity vector as per the velocity change equation.
            $\vec{V}_i(t+1) = \vec{V}_i(t) + \rho_1(\vec{X}_{pbest(i)} - X_i(t)) + \rho_2\left(\vec{X}_{gbest(i)} - X_i(t)\right)$
        Update particle to the new position as per position change equation.
            $\vec{X}_i(t+1) = \vec{X}_i(t) + \vec{V}_i(t+1)$
    }
}While(convergence is not reached)
```

B. Fitness Function

To measure the performance of each particle, there should be a method to find the closeness of each solution to the optimum solution. We used a single fitness value for each set of parameters which determined the optimality of the parameters. Our aim was to produce a model which generates a spike train which is identical or similar to the reference spike train. Since neuron convey information through spike train responses, we measured the similarity of spike trains using a coincidence factor г [10] [11] which is the difference between the number of coincidences and the number of coincidences by chance relative to the total number of spikes produces by the reference and model spike trains (1).

$$\Gamma = \frac{N_{coinc} - \langle N_{coinc} \rangle}{\frac{1}{2}(N1 + N2)}\left(\frac{1}{N}\right) \tag{1}$$

where *N1* and *N2* are the number of spikes in the model and reference spike trains, N_{coinc} is the number of coincidence within the time window δ, $< N_{coinc} >$ = $2v\delta N1$ is the expected number of coincidences generated by a homogeneous Poisson process with firing rate ν same as that of model spike train. Γ has expectation zero for a purely random spike train and has value 1 if there is a perfect coincidence between the model and reference spike trains.

C. Spiking Neuron Modeling: Adaptive Exponential Leaky Integrate and Fire Model

The elementary processing units of the nervous system are modeled using a phenomenological neuron model, Adaptive Exponential Leaky Integrate and Fire Model, a two-dimensional integrate and-fire model that combines an exponential spike mechanism with an adaptation equation. This model is able to correctly predict timing of 96% of the spikes (±2 ms) of a detailed conductance based model [12] since this model can capture the effects of a number of fast ionic currents in a single equation. The equations (2) and (3) of the model are able to generate different firing patterns and are found to be suitable for large network simulations [7].

$$\frac{dV}{dt} = \frac{gl * (V - El) + gl * delT * \exp\left(\frac{V - Vt}{delT}\right) - Isyn - w}{C} \tag{2}$$

$$\frac{dw}{dt} = a * (V - El) - w, \tag{3}$$

$$If V > 0\, mv, V = Vr \& w = w + b$$

where *C* is the membrane capacitance, *gl* and *El* are the leak conductance and resting potential, *Vr* represents the resting membrane potential, *Vt* denotes the threshold potential, *delT* is the slope factor, *a* and *b* are the parameters showing sub-threshold adaptation and spike-triggered adaptation. The first equation describes the dynamics of the action potential generation and the second equation describes adaptation in the firing rate of the neuron. The equation follows the dynamics of an RC circuit until *V* reaches *Vt.* The neuron is assumed to fire on crossing this threshold voltage and the downswing of the action potential is replaced by a reset of membrane potential *V* to a lower value *Vr.*

The PSO optimization algorithm was used in our study to mimic the behavior of two different types of neurons in the cerebellar granular layer: The granule neurons and Golgi neurons. The granule neurons form the main input layer of the cerebellum through which the mossy fiber inputs innervate the cerebellum. Golgi neurons showed spontaneous pace maker activity with a frequency between 1 and 8Hz at room temperature [13] while granule neurons showed no such spontaneous activity at rest but showed regular repetitive firing at current injection [14]. The scaling and bifurcation parameters of the spiking neuron model were fine tuned to match the electrophysiological recordings of the granule and Golgi neurons using current clamp protocol.

D. GPU Implementation

GPUs outperformed the CPUs with enormous arithmetic capability, streaming memory bandwidth and with a richer set of APIs. Since highly parallel programmable processors like GPUs deliver a high compute capacity at low cost, the scientific computing community is moving the computationally intensive parts of their software from

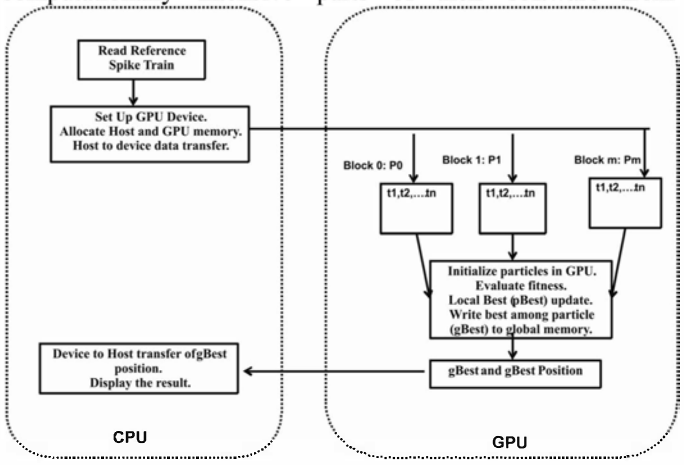
Read Reference Spike Train
Set Up GPU Device. Allocate Host and GPU memory. Host to device data transfer.
Block 0: P0
Block 1: P1
Block m: Pm
t1,t2,....tn
t1,t2,....tn
t1,t2,....tn
Initialize particles in GPU. Evaluate fitness. Local Best (pBest) update. Write best among particle (gBest) to global memory.
Device to Host transfer ofgBest position. Display the result.
gBest and gBest Position
CPU
GPU

Fig.1: CPU and GPU task subunits for the execution of the PSO algorithm. Each particle was assigned to one GPU block. Each thread in a block was mapped to a single dimension of the problem. The blocks shared the gBest values at the end of each iteration.

distributed computing clusters to GPUs. Since PSO can be formulated as an embarrassingly parallel algorithm with minimal exchange of information between particles, we used single instruction multiple data paradigm to get a parallel implementation of the algorithm. Each Individual particle was updated in parallel and updations were performed based on the same set of rules for all particles. Each updation depended on its own history (pBest) and knowledge of its neighbors (gBest). So the simulations were run independently for each particle with exchange of only gBest information between them. Multiple parallel blocks were allocated in the GPU and were run at the same time simulating one particle per block. The total number of threads per block allocated equals the number of dimensions of the problem (Fig. 1).

The essential sequential components of the optimization process such as reading external reference voltage and setting the default optimisation parameters etc. were performed in an Intel Xeon CPU with 8 cores running at 2.4 GHz. The AdEx model was then fitted to the reference data in parallel using NVIDIA GeForce GT 440 with 96 cores running at 1.64 GHz and 1 GB of DRAM**.** The global best response of the entire particles was then copied back to the CPU and were visualised.

III. RESULTS AND DISCUSSION

PSO algorithm was simple and versatile to solve optimization problems from varying domains. Yet, it remains as a challenging job for a researcher to modify and use the basic structure of the algorithm to suit the underlying properties of the domain under consideration. Here, the algorithm was well investigated and re-implemented to fit with the application domain and underlying hardware.

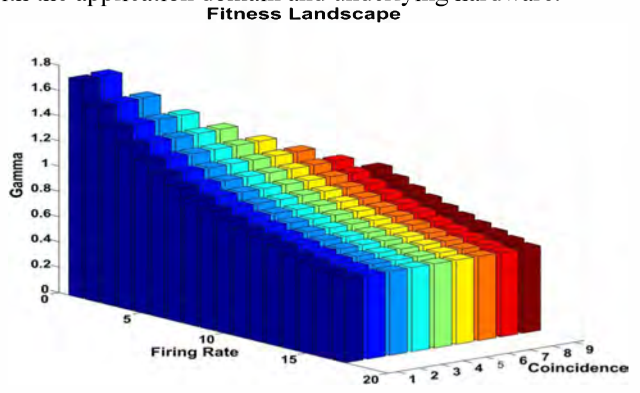

Fig. 2: The plot shows the gamma values for reference spike train with 9 spikes while firing rate of the model was varied from 1 to 18 and the gamma values were plotted for coincidences from 1 to 9. Coincidences were calculated in a 4ms time window. The optimum gamma value 1 was reached for an exact match of firing rate and coincidences.

A. Factors affecting fitness function

Since we used a discrete fitness parameter Γ for each iteration, the fitness landscape for the parameters was identified to evaluate the fitness function correctly. The fitness landscape plots the n-dimensional parameter space against the one-dimensional fitness value. Similarities of spike trains were identified based on the coincidence factor. The quality of Γ is reduced with the difference in firing rate between the model and reference spike trains and also when the numbers of coincident spikes were different in the time window considered. The variation in fitness factor, Γ with the change in firing rate and the number of coincidences is thoroughly investigated to identify the best fitness criteria (Fig. 2). It has been observed that the optimal value of 1 is achieved when both the firing rate and coincidences matched with the reference train.

The proximal values of Γ were affected by the number of iterations performed since we used a stochastic search in the search space (Fig. 3: A). Also, for a fixed set of iterations, as the range of available values for each of the parameters increased, the area of the search space increased and the probability of getting a feasible solution decreased (Fig. 3: B). In order to compensate these, we needed to perform more iterations exploring the complete possible range for the parameter values. GPUs seemed more efficient in this fitting process since GPU allowed us to perform more operations per unit time when compared to CPU.

Since our problem is a constrained optimization problem, whenever the algorithm encountered a partial solution that did not satisfy the model equations to produce voltage traces, the algorithm backtracked, instead of trying to extend that solution. Another possibility to improve the algorithm was to use variable boundary values for different parameters of the model. There were some variables in the model (such as Vr and $delT$) whose change adversely affected the resulting voltage traces while there exist feasible solutions for a wide range of other parameters. Such domain knowledge and expertise was incorporated during the fitting process. Hence the parameter Vr and $delT$ were fixed constant or were given a narrow range while for other parameters a wide range was chosen.

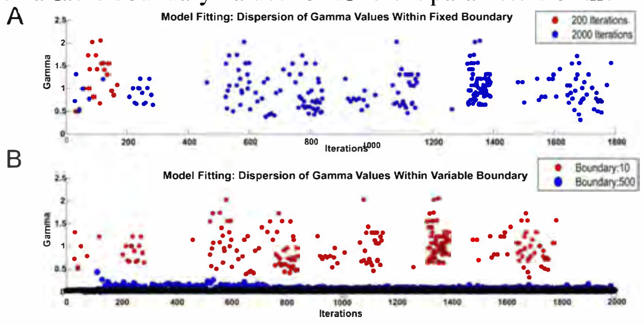

Fig. 3: A) Dispersion of Gamma for different number of iterations within a limited boundary for the search space of each particle dimension. B) Dispersion of Gamma for fixed number of iterations but with two different boundary values for the search space of each particle dimension. The probability of best fit was decreased as the area of search space was increased.

B. Estimating the Optimization Error

The mean absolute error (MAE) is calculated to measure how close the values of the fitted AdEx parameters were to the actual known parameter values.

$$MAE = \frac{1}{N}\sum_{i=1}^{n} fi - xi \quad (4)$$

Where fi are the predicted values and xi are the true values. The best parameter values identified so far are chosen as xi for error calculations. The error estimates helped us to modify the optimization parameters. For e.g., MAE is found to increase with the increase in boundary of the search space for fixed iterations (Fig. 4). Increasing the number of iterations and number of particles helped us to reduce the error.

C. GPU Simulations

The model fitting was performed in GPU where each iteration required the neuron model to be simulated for a fixed time period to calculate the fitness. The neurons were simulated for 100 ms time with discretisation time-step chosen as 0.02 ms. The coincidence of each model spike train against the reference train was calculated in a 4 ms time window. The reference voltages were recorded from the multi-compartmental neuronal model [15] and we were able to correctly reproduce both granule neuron and Golgi neuron firing patterns of the cerebellum with millisecond precision (Fig. 5 A and B). The optimization process was performed on CPU and GPU for a benchmark comparison of the runtimes

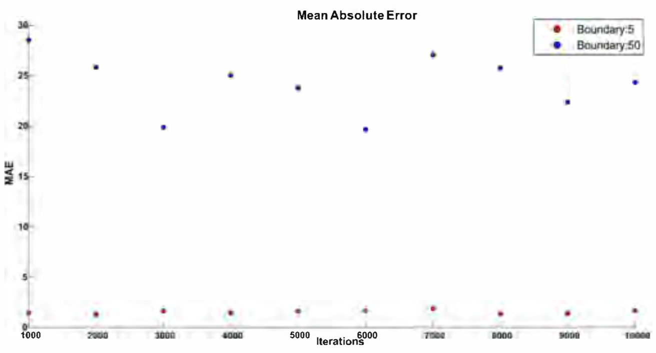

Fig.4: Mean Absolute Error was found to increase with the increase in area of the search space for fixed number of iterations.

and we obtained a 15-fold performance improvement in GPU than in CPU for runtime (Fig. 6).

IV. CONCLUSION

Parameter estimation and non linear fitting on parallel hardware enabled us to set various experimental boundary and initial conditions which was essential in fitting neuron models to experimental trains especially when the type of neuron and its biophysical characteristics are not known. The search process can be further improved by implementing both data parallel and task parallel computations where the entire search space boundary can be divided and distributed among different blocks of the same GPU. This implementation of search optimization is being extended and the scalability is being tested on our multi-GPU system with high-end GPU cards.

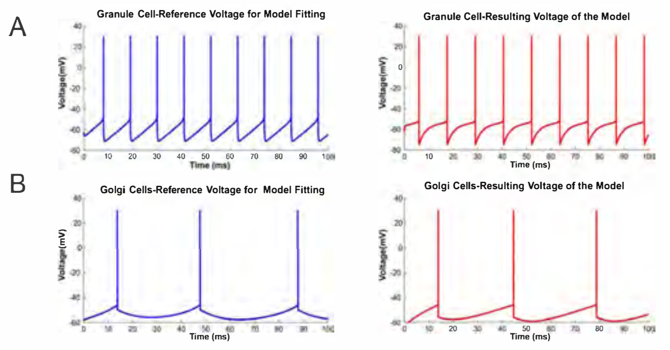

Fig.5: Voltage responses of models fitted to spike trains. A) AdEx model fitted to Granule cells of the cerebellum. B) AdEx model fitted to Golgi cells of the cerebellum.

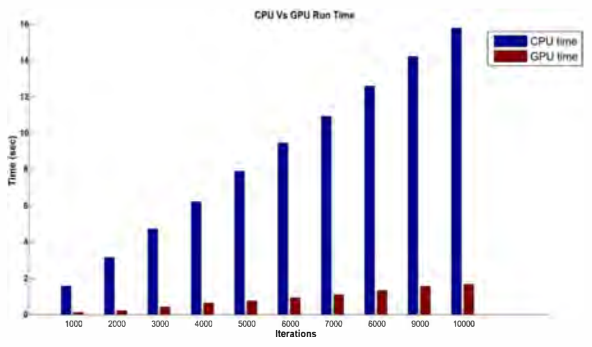

Fig.6: Comparison of CPU-GPU run time. GPU gave approximately 15-fold performance improvement.

Acknowledgment

This work derives direction and ideas from the chancellor of Amrita University, Sri Mata Amritanandamayi Devi. This work is supported by NVIDIA CTC grants 2012–13, 2013-14 and is partially supported by grants SR/CSI/49/2010 and SR/CSI/60/2011 and Indo-Italy POC 2012-13 from DST, BT/PR5142/MED/30/764/2012 from DBT, Government of India.

References

[1] D. Sterratt, B. Graham, A. Gillies, and D. Willshaw, *Principles of Computational Modelling in Neuroscience*. Cambridge University Press, 2010.

[2] Z. Zhang, "Parameter estimation techniques: A tutorial with application to conic fitting," *Image Vis. Comput.*, vol. 15, no. 1, pp. 59–76, 1997.

[3] C. M. Fonseca and P. J. Fleming, "Genetic Algorithms for Multi objective Optimization□: Formulation Discussion and Generalization.," *ICGA*, vol. 93, no. July, 1993.

[4] J. Kennedy and R. Eberhart, "Particle swarm optimization," *Proc. ICNN'95 - Int. Conf. Neural Networks*, vol. 4, pp. 1942–1948, 1995.

[5] Z. Michalewicz and M. Schoenauer, "Evolutionary Algorithms for Constrained Parameter Optimization Problems," *Evol. Comput.*, vol. 4, no. 1, pp. 1–32, 1996.

[6] X. Hu and R. Eberhart, "Solving Constrained Nonlinear Optimization Problems with Particle Swarm Optimization," in *Proceedings of the sixth world multiconference on systemics, cybernetics and informatics.*, 2002, pp. 203–205.

[7] R. Naud, N. Marcille, C. Clopath, and W. Gerstner, "Firing patterns in the adaptive exponential integrate-and-fire model.," *Biol. Cybern.*, vol. 99, no. 4–5, pp. 335–47, Nov. 2008.

[8] Chaitanya Medini, Asha Vijayan, E. D.Angelo, "Computationally Efficient Bio-realistic Reconstructions of Cerebellar Neuron Spiking Patterns," in *ICONIA*, 2014.

[9] C. Rossant, D. F. M. Goodman, B. Fontaine, J. Platkiewicz, A. K. Magnusson, and R. Brette, "Fitting neuron models to spike trains.," *Front. Neurosci.*, vol. 5, no. February, p. 9, Jan. 2011.

[10] R. Jolivet, T. J. Lewis, and W. Gerstner, "Generalized integrate-and-fire models of neuronal activity approximate spike trains of a detailed model to a high degree of accuracy.," *J. Neurophysiol.*, vol. 92, no. 2, pp. 959–76, Aug. 2004.

[11] W. Kistler, W. Gerstner, and J. L. Van Hemmen, "Reduction of the Hodgkin-Huxley Equations to a Single-Variable Threshold Model," *Neural Comput.*, vol. 9, pp. 1015–1045, 1997.

[12] R. Brette and W. Gerstner, "Adaptive exponential integrate-and-fire model as an effective description of neuronal activity.," *J. Neurophysiol.*, vol. 94, no. 5, pp. 3637–42, Nov. 2005.

[13] L. Forti, E. Cesana, J. Mapelli, and E. D'Angelo, "Ionic mechanisms of autorhythmic firing in rat cerebellar Golgi cells.," *J. Physiol.*, vol. 574, no. Pt 3, pp. 711–29, Aug. 2006.

[14] E. D'Angelo, et al.,"Ionic Mechanism of Electroresponsiveness in Cerebellar Granule Cells Implicates the Action of a Persistent Sodium Current Ionic Mechanism of Electroresponsiveness in Cerebellar Granule Cells Implicates the Action of a Persistent Sodium Current," *J. Neurophysiol.*, pp. 493–503, 1998.

[15] S. Diwakar, J. Magistretti, M. Goldfarb, G. Naldi, and E. D'Angelo, "Axonal Na+ channels ensure fast spike activation and back-propagation in cerebellar granule cells.," *J. Neurophysiol.*, vol. 101, no. 2, pp. 519–32, Mar. 2009.