Phase control optimization in Coherent Beam Combining (CBC) systems using SPGD with different optimizers

Samudrala Hareesh

July 10,2024

Guide: Dr. Balaji Srinivasan
Professor
Department of Electrical Engineering
IIT Madras

Contents

1 INTRODUCTION

Attaining active phase control among multiple sources is a critical implementation challenge for Coherent Beam Combining (CBC). The optical beams intended for combination experience perturbations caused by numerous factors, impeding their ability to interfere in-phase and thereby diminishing the beam combining efficiency. Active phase control allows for real-time adjustment of phase changes, improving combining efficiency.

The active feedback strategy employed in CBC commonly adopts a Master Oscillator Power Amplifier (MOPA) architecture. In this, light from a master oscillator is typically split into multiple arms, amplified independently, precisely controlled and interfered coherently to achieve high power levels. This is achieved while preserving a high level of beam quality.

The beam combining approaches in CBC may be either a filled aperture or a tiled aperture configuration. Here we are using tiled aperture configuration.

When developing a phase synchronization method, the following aspects are to be considered: 1) Scalability
2) Maintaining the locked state
3) Real-time implementation.

Active phase control approaches fall into two types based on their sensing methods: 1) phase sensing
2) metric sensing

In metric sensing, Stochastic Parallel Gradient Descent (SPGD) algorithm is often used for the phase synchronization. The SPGD is relatively simple and scalable.

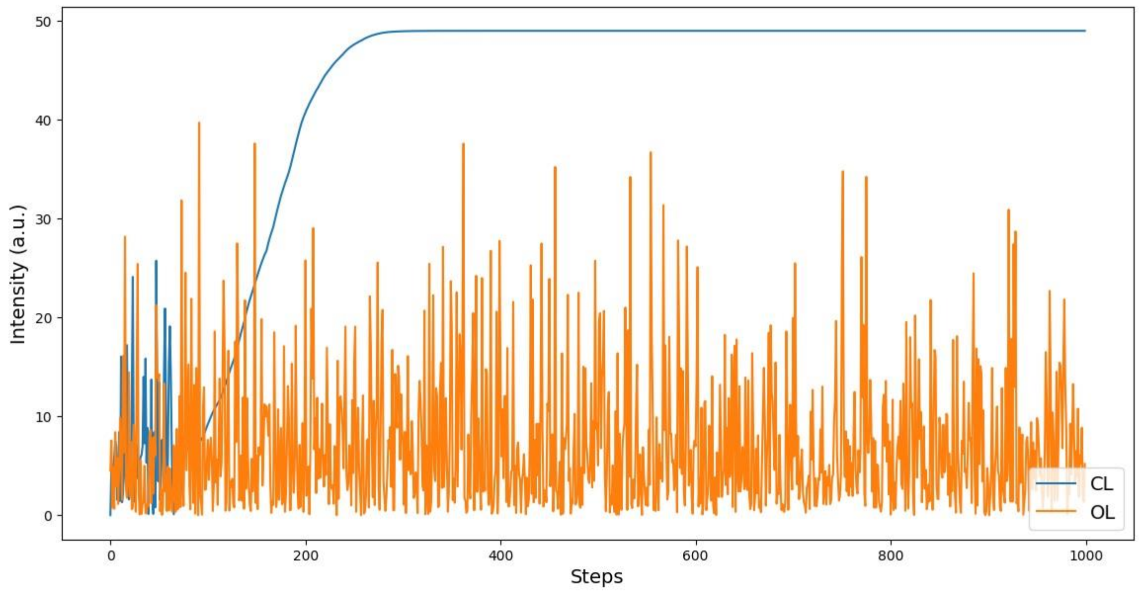
2 METHODOLOGY

The choice and application of optimizers within the Stochastic Parallel Gradient Descent (SPGD) framework can significantly affect the beam combination efficiency in coherent beam combining (CBC).

The following are phase locking simulations with number of channels as 7 and noise used is uniform distribution:

1)ADAM (2014):

Adam is known for its efficiency in terms of computation and memory. It combines the advantages of two other extensions of stochastic gradient descent, specifically Momentum and Decaying learning rate. Adam maintains separate learning rates for different parameters (adaptive learning rates) and also keeps an exponentially decaying average of past gradients and squared gradients.



Using Adam

The mathematical expression for adam is as follows: (Kingma & Ba, 2014)

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t}$$

$$\hat{v}_t = \frac{v_t}{1-\beta_2^t}$$

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2$$

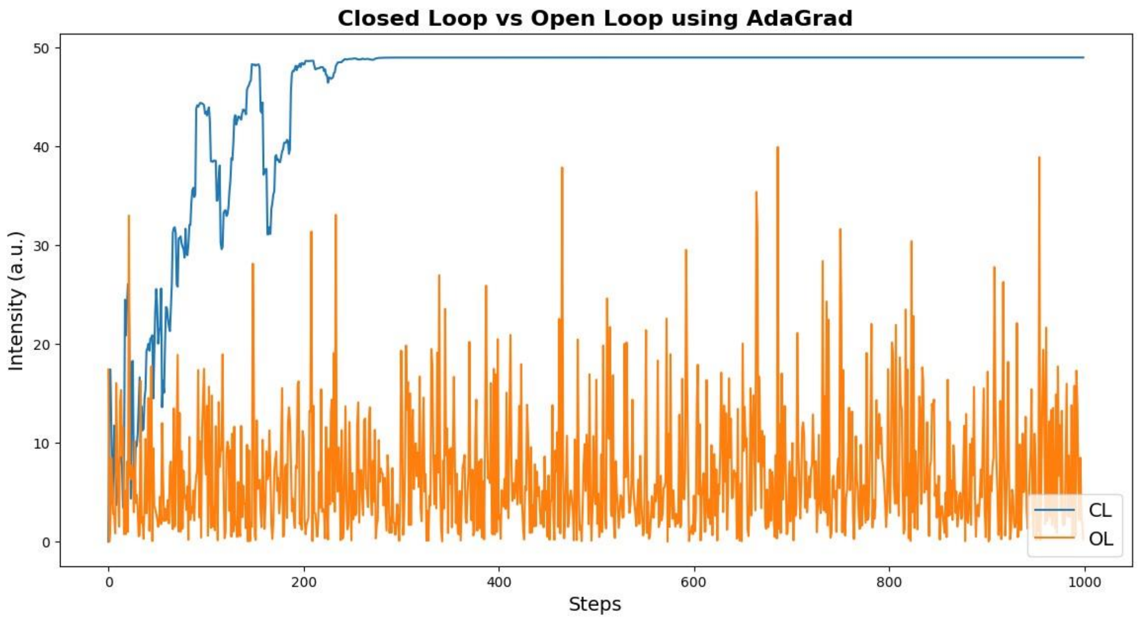
$$\theta_t = \theta_{t-1} - \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t}+\epsilon}$$

2)ADAGRAD (2011):

AdaGrad adapts the learning rate based on the history of gradients, which makes it suitable for dealing with sparse data and features. It maintains a separate learning rate for each parameter, which allows it to perform well in cases where different parameters require different magnitudes of updates. AdaGrad accumulates the sum of the squares of the gradients for each parameter. This accumulation ensures that the learning rate decreases over time, especially for parameters associated with larger gradients.

(Duchi JDUCHI & Singer, 2011)

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\eta}{\sqrt{G_{t,ii} + \epsilon}} g_{t,i}$$



3)ADADELTA: (2012)

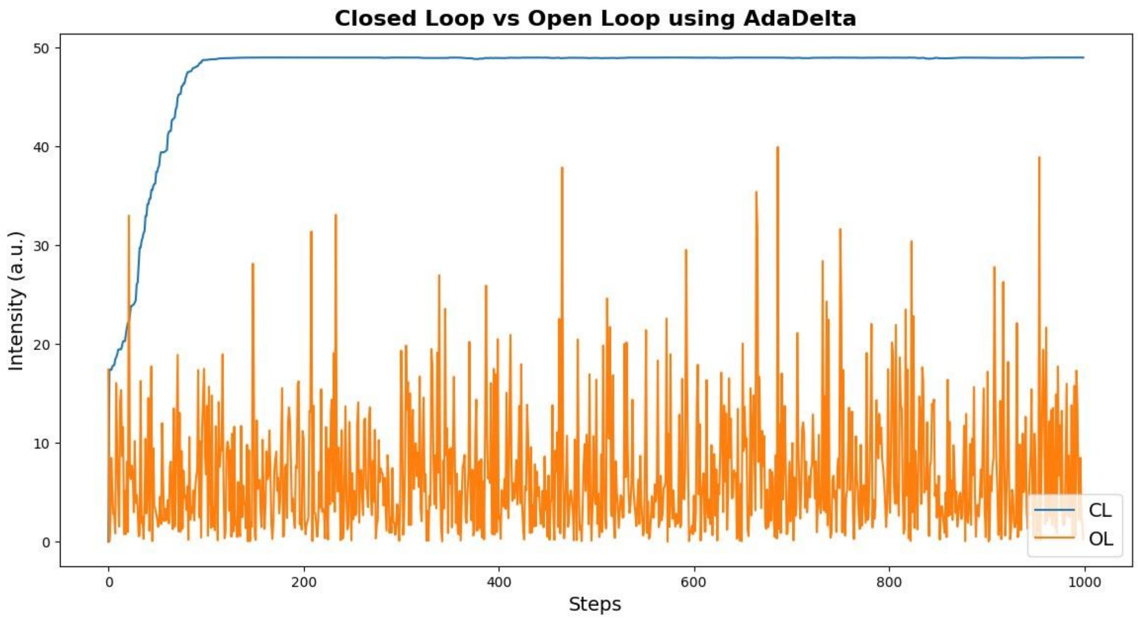
Adadelta is an extension of AdaGrad that aims to reduce its aggressive, monotonically decreasing learning rate. That is it avoids the problem of learning rates becoming too small (AdaGrad). It restricts the window of accumulated past gradients to a fixed size.

(Zeiler, 2012)

$$\Delta\theta_t = -\eta \cdot g_{t,i}$$

$$\theta_{t+1} = \theta_t + \Delta\theta_t$$

$$\Delta\theta_t = -\frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} g_t$$



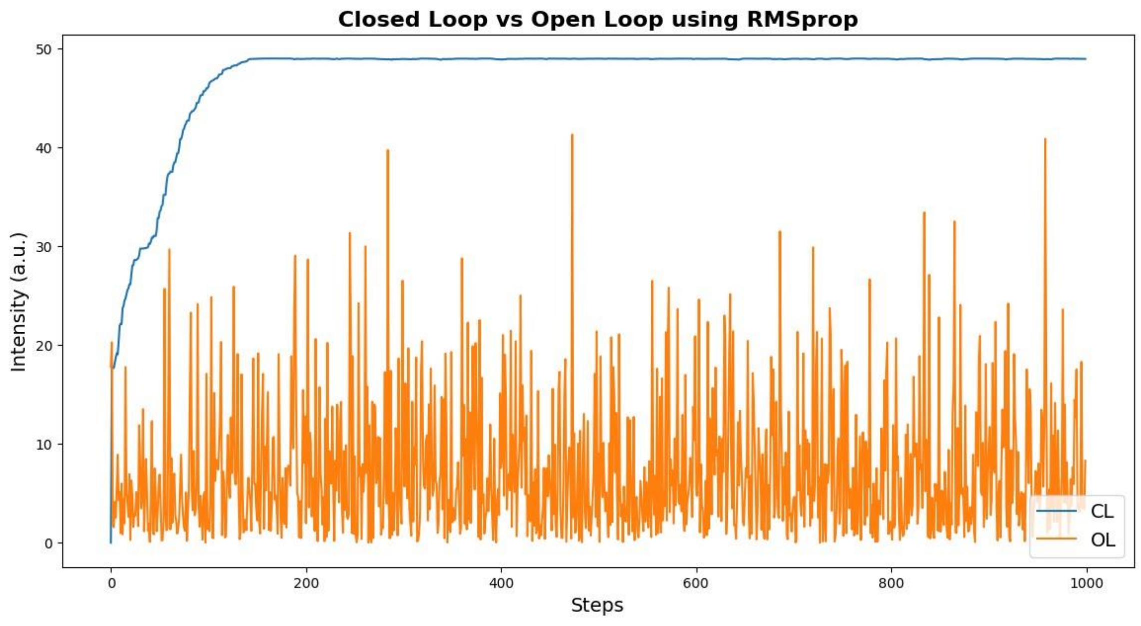
4)RMSPROP: (2012)

RMSProp divides the learning rate by an exponentially decaying average of squared gradients. It helps in dealing with the diminishing learning rates problem of AdaGrad.

The mathematical expression for rmsprop is as follows: (Ruder, 2016)

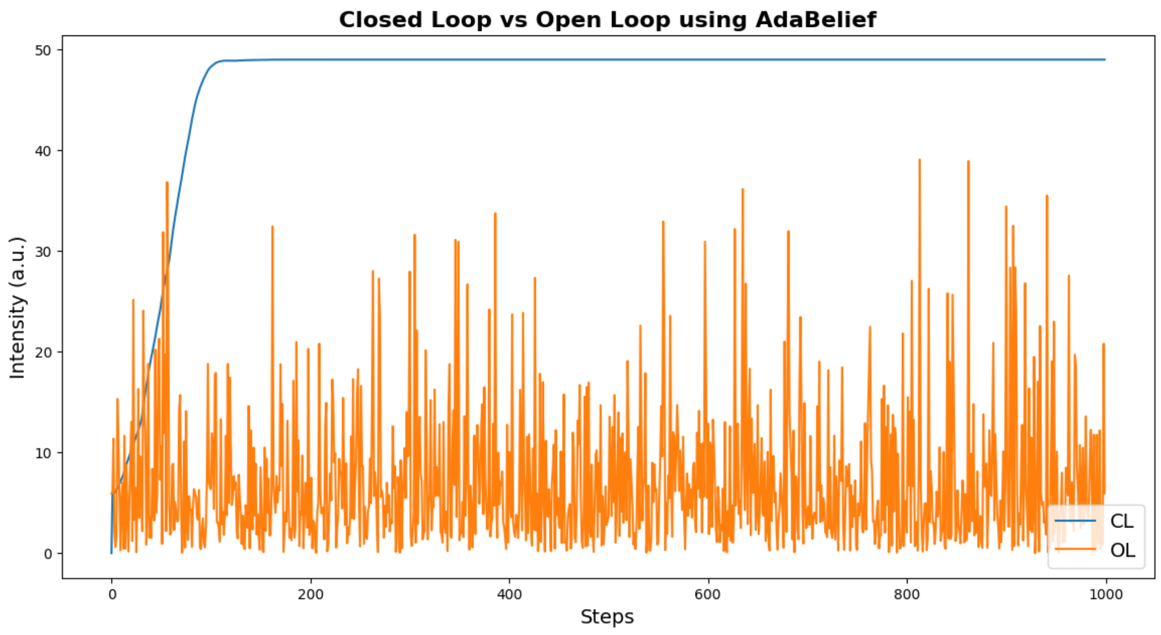
$$E[g^2]_t = \gamma E[g^2]_{t-1} + (1-\gamma) g_t^2$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} g_t$$



5) AdaBelief: (2020)

AdaBelief modifies the update rule of Adam by considering the "belief" in the observed gradient, aiming to improve generalization and training performance. AdaBelief extends Adam by employing adaptive moments with bias correction, akin to AMSGrad, and incorporating the variance of gradients to stabilize training.



The mathematical expression for adabelief is as follows: (Zhuang et al., 2020)

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$$

$$s_t = \beta_2 s_{t-1} + (1 - \beta_2)(g_t - m_t)^2$$

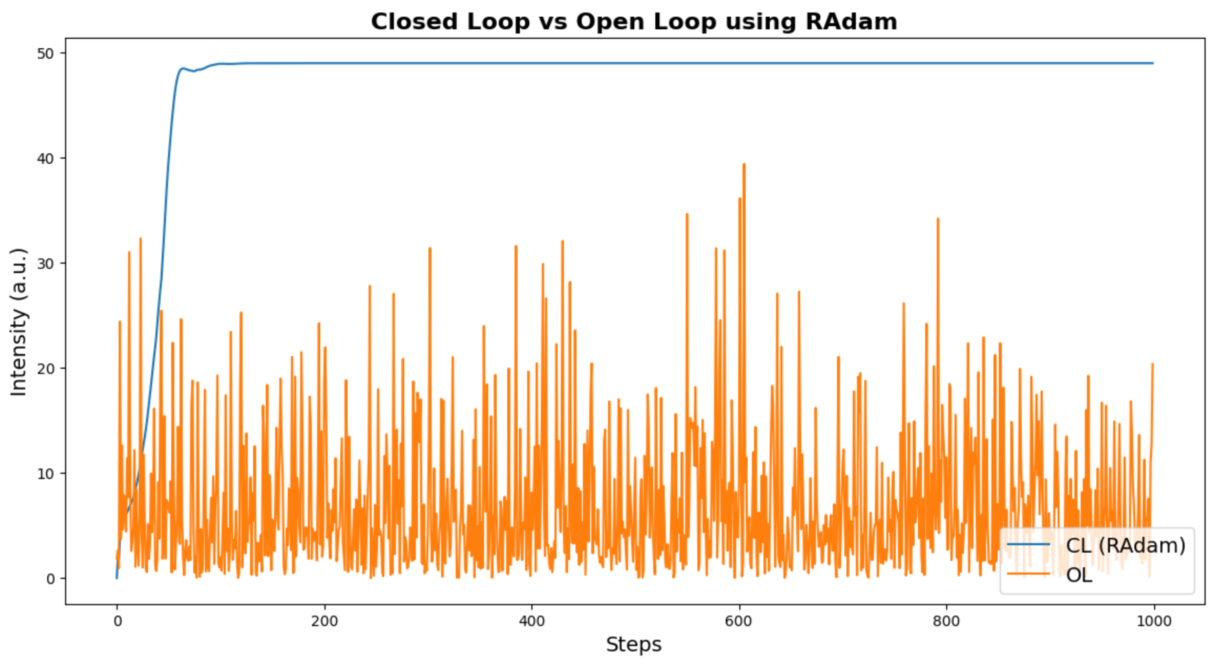
$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}$$

$$\hat{s}_t = \frac{s_t}{1 - \beta_2^t}$$

$$\theta_{t+1} = \theta_t - \alpha \frac{\hat{m}_t}{\sqrt{\hat{s}_t} + \epsilon}$$

6) RAdam (Rectified Adam): (2019)

RAdam rectifies the variance of the adaptive learning rate, aiming to stabilize the training process. RAdam transitions smoothly from SGD-like behaviour in the early stages of training to Adam-like behaviour in later stages. This is achieved through the rectification term, which gradually decreases the variance of the learning rate as training progresses.



The mathematical expression for radam is as follows: (Liu et al., 2019)

$$v_t = 1/\beta_2 v_{t-1} + (1-\beta_2)g_t^2$$

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)g_t$$

$$\hat{m}_t = m_t/(1-\beta_1^t)$$

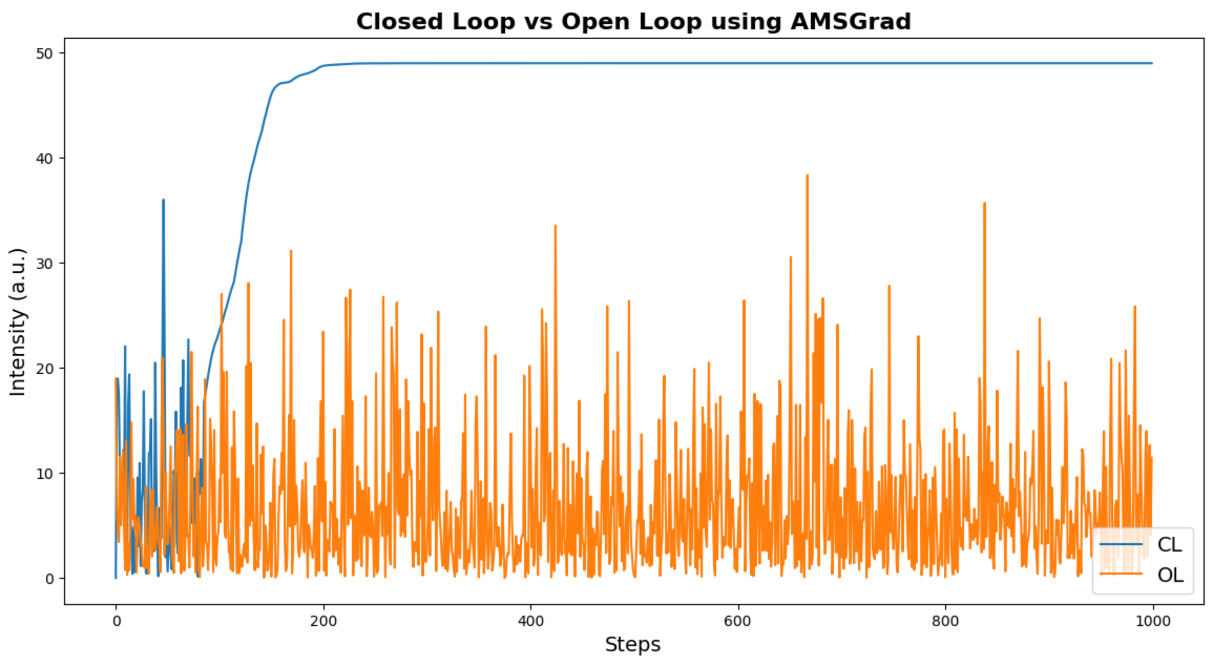
$$\rho_t = \rho_\infty - 2t\beta_2^t/(1-\beta_2^t)$$

$$\rho_\infty = \frac{2}{1-\beta_2} - 1$$

$$\theta_t = \theta_{t-1} - \alpha_t \hat{m}_t$$

7) AMSGrad: (2018)

AMSGrad is a variant of Adam that maintains the maximum of past squared gradients instead of the exponential average to address the convergence issues of Adam. By ensuring that the second moment estimate does not decrease, AMSGrad addresses the potential non-convergent behaviour of Adam, providing better theoretical convergence guarantees.



The mathematical expression for AMSGrad is as follows: (Reddi et al., 2019)

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$$

$$\hat{v}_t = \max(\hat{v}_{t-1}, v_t)$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} m_t$$

These simulation plots assist us to proceed with the actual experiment. They offer an understanding of whether each optimizer is feasible or not.

2.1 ANALYSIS OF CLOSED LOOP RESPONSE

The plots were taken after locking the phase using the different optimizers having the noise emulation turned on (with x-axis as time in seconds and y-axis as %efficiency):

ADAM-18% efficiency

ADADELTA-18% efficiency

RMSPROP-19% efficiency

ADABELIEF-19% efficiency

RADAM-19% efficiency

AMSGRAD-19% efficiency

The efficiencies mentioned above are the mean values.

The variances of these plots are:

ADAM- 3.449974×10^{-3}

ADADELTA- 4.121861×10^{-3}

RMSPROP- 1.763538×10^{-4}

ADABELIEF-2.552055×10^{-4}

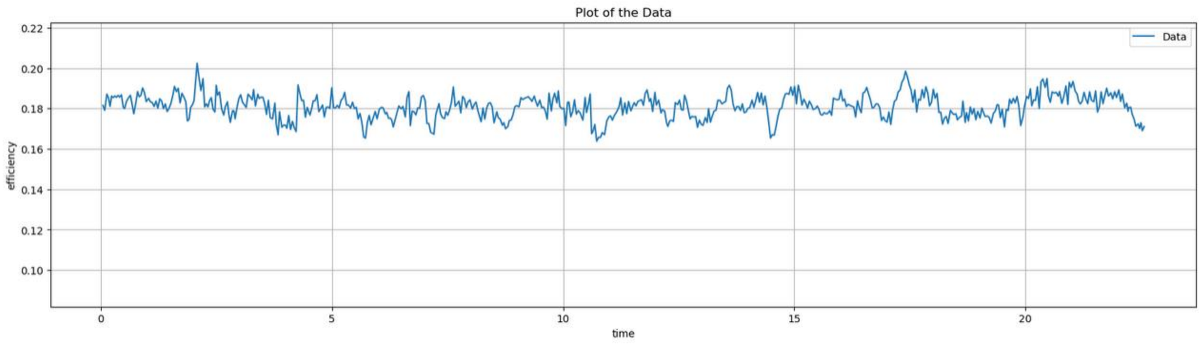
RADAM-2.515491×10^{-4}

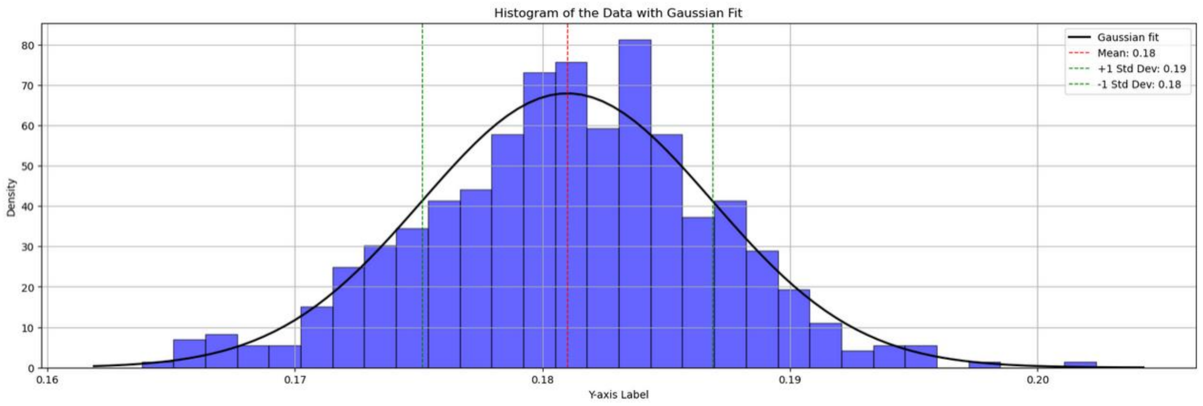
AMSGRAD-5.0016267×10^{-4}

By observing the variance, it is clear that RMSPROP is more stable.
While performing experiment, the environmental conditions are as follows:

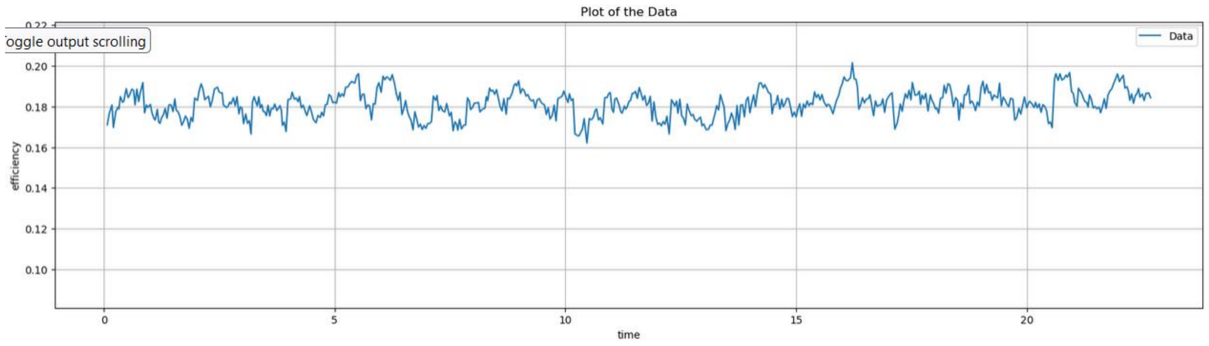
Temperature-24^{o} C
Humidity- 60%

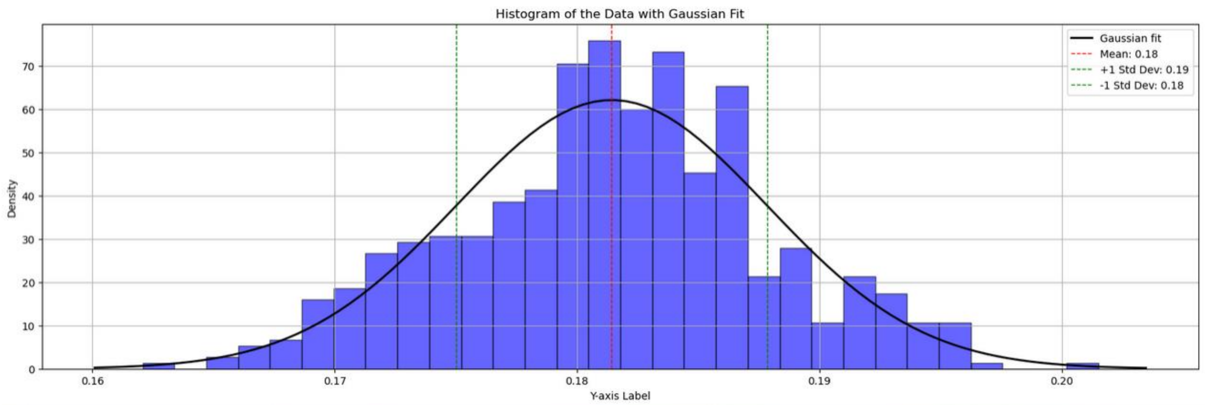
Adam:



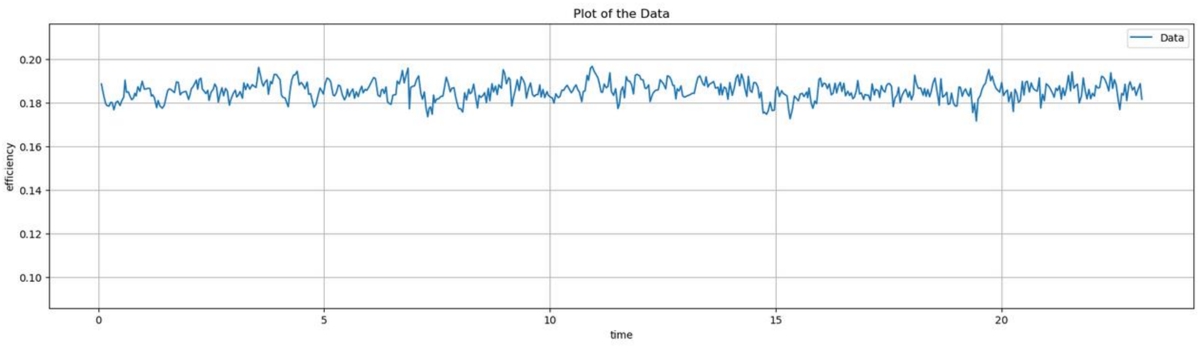


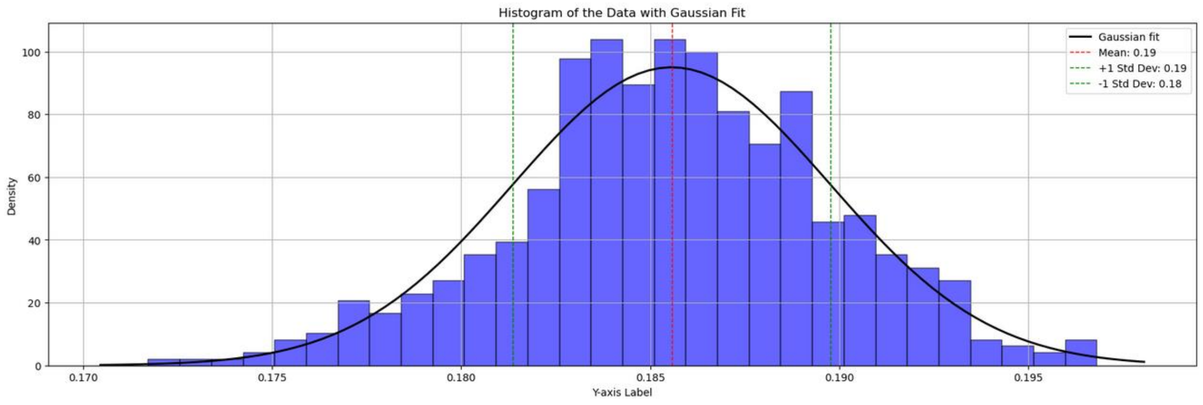
AdaDelta:



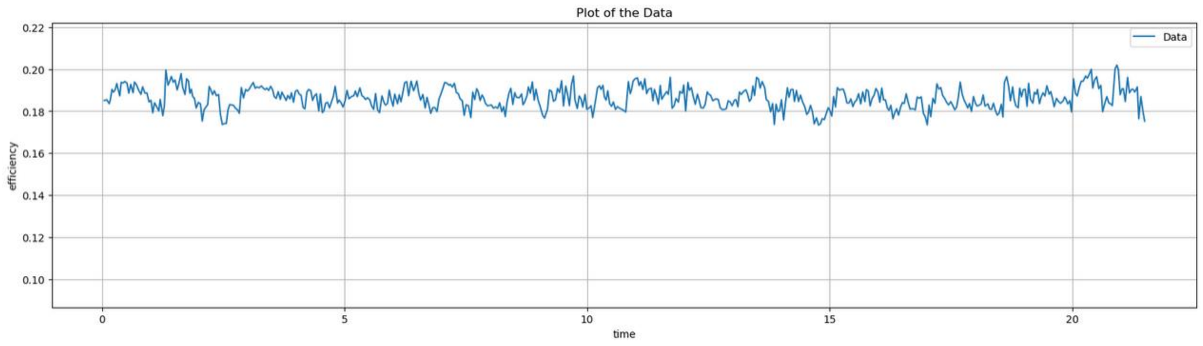


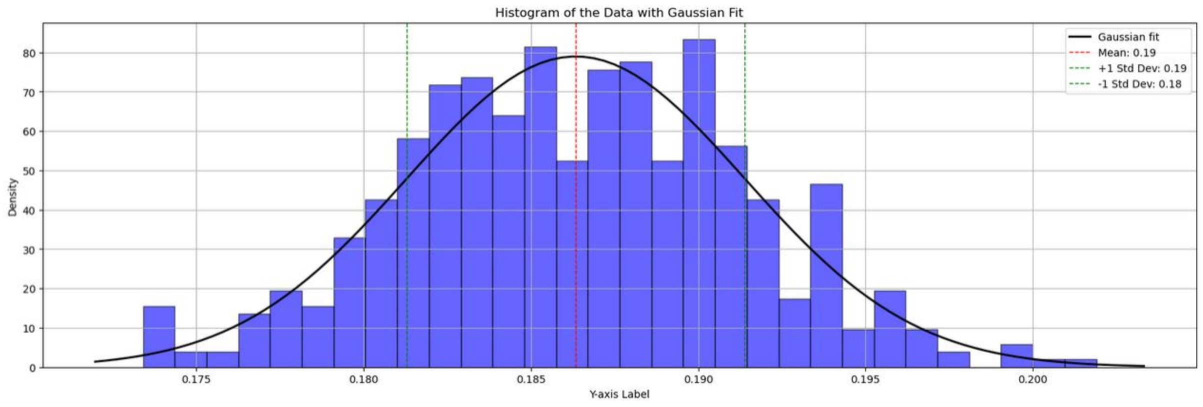
Rmsprop:



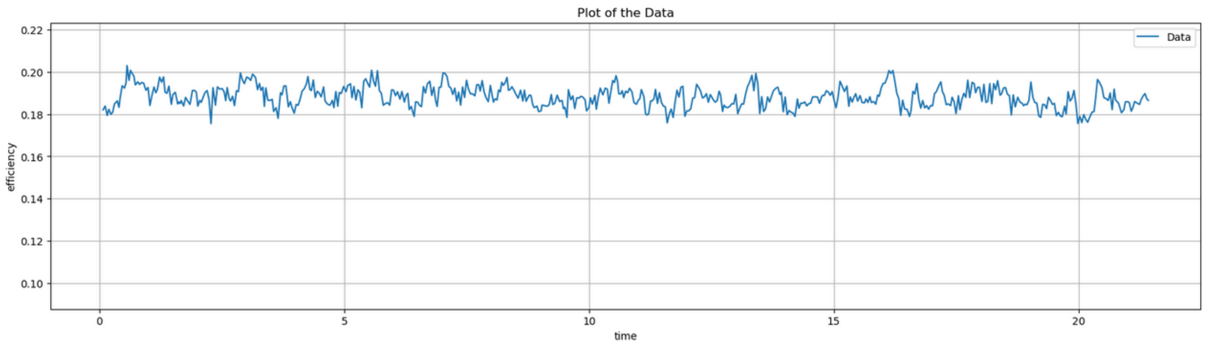


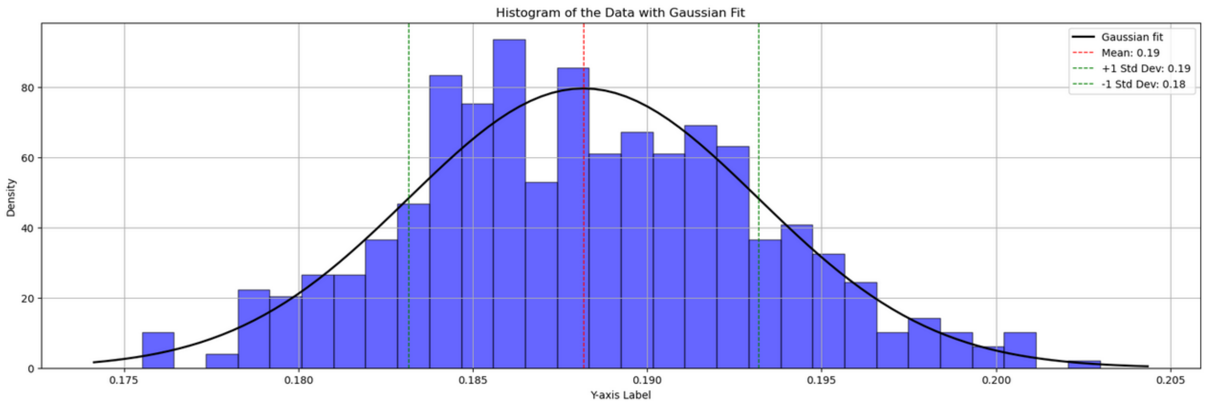
AdaBelief:



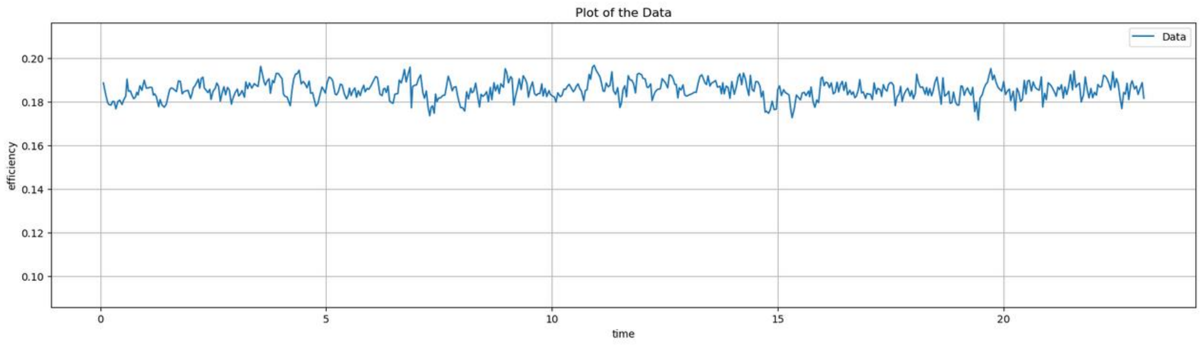


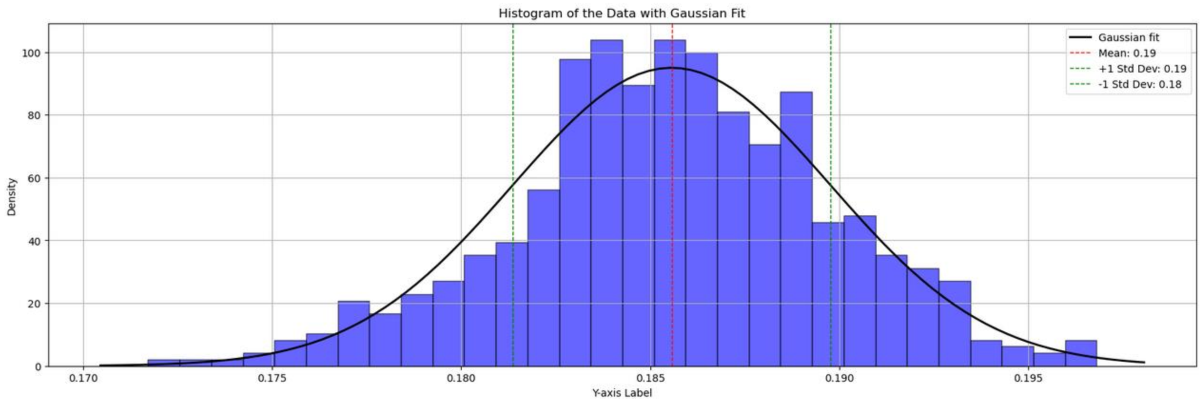
Radam:





AMSGrad:

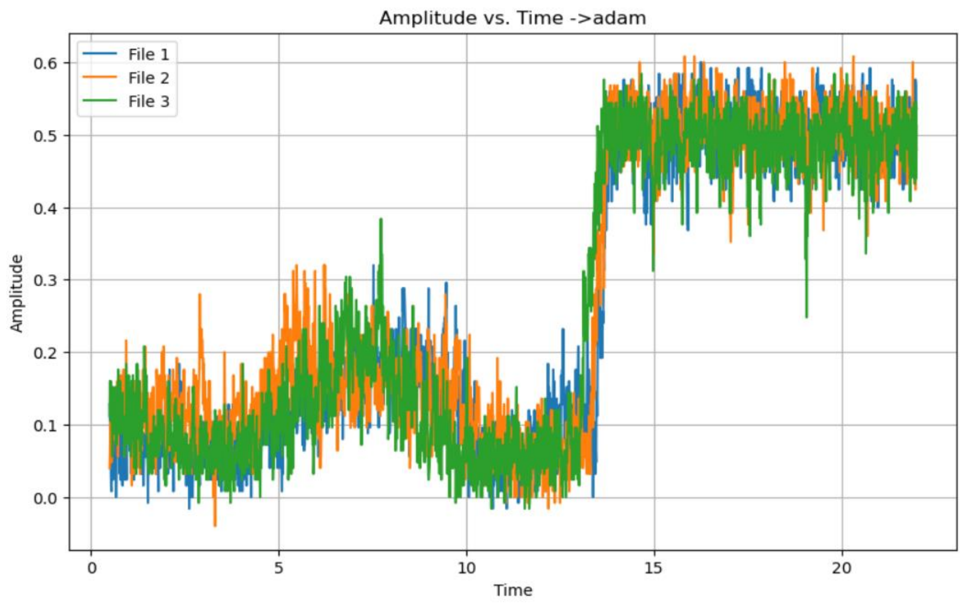




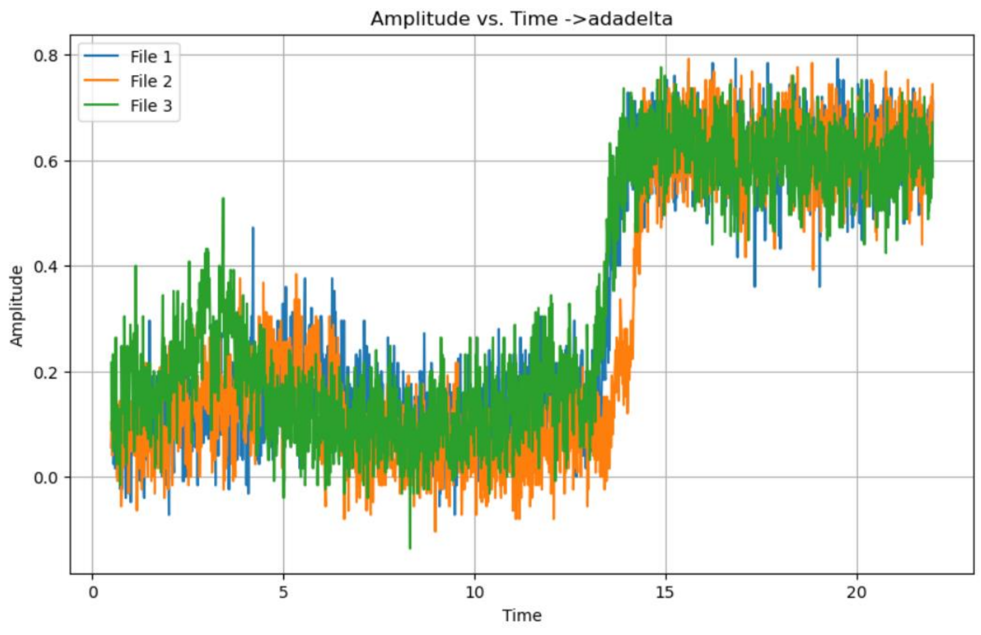
2.2 ANALYSIS OF THE OVERALL LOOP RESPONSE

To analyse the overall loop response using different optimizers, I performed phase locking experiments three times using each optimizer and plotted the corresponding graphs. (with x-axis as time in seconds and y-axis as voltage in volts). While conducting the experiment, power level of each beam is 1 watt and the total power level is 7 watts (as seven beams were used). With noise emulation turned on.

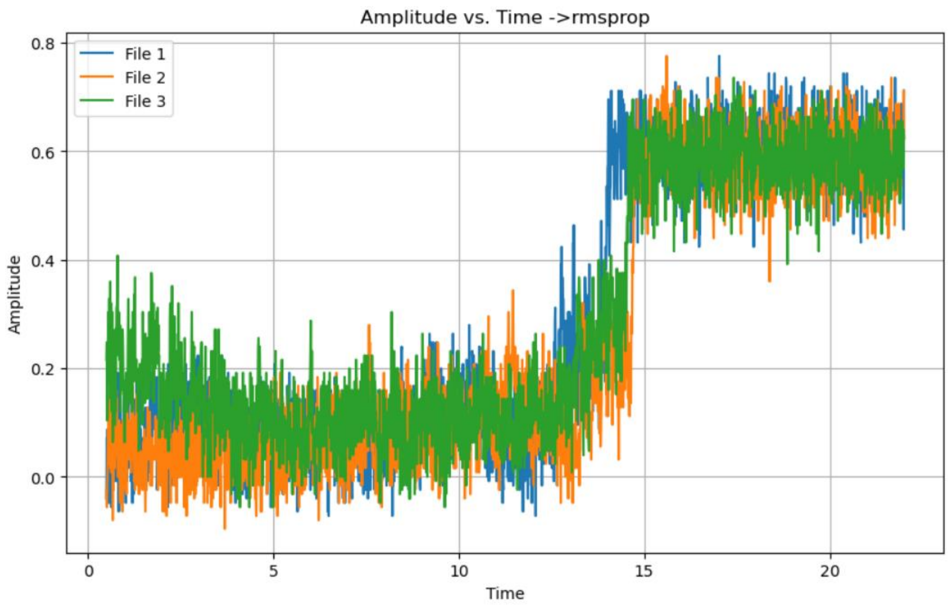
Adam:



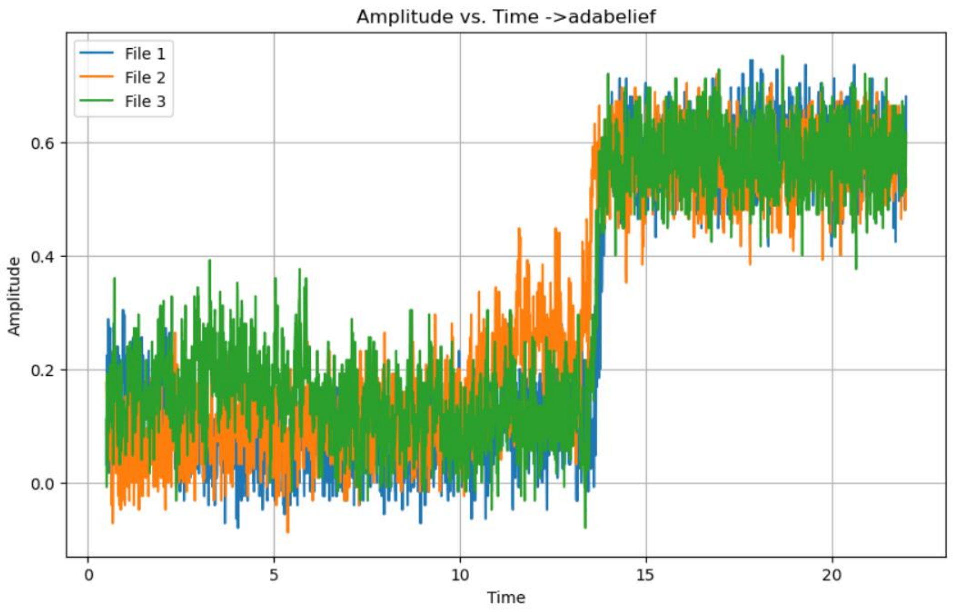
Adadelta:



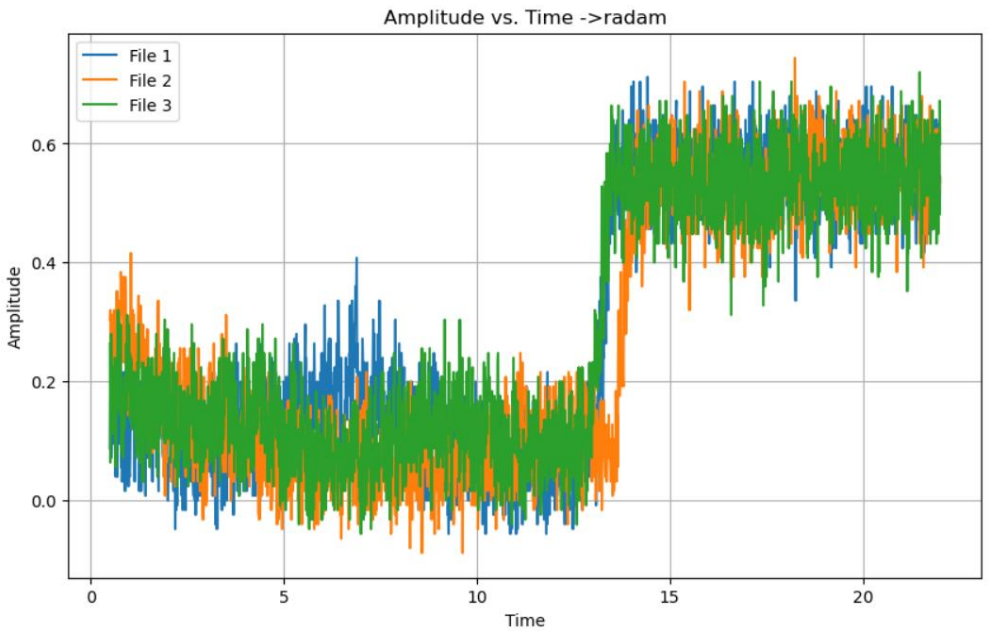
Rmsprop:



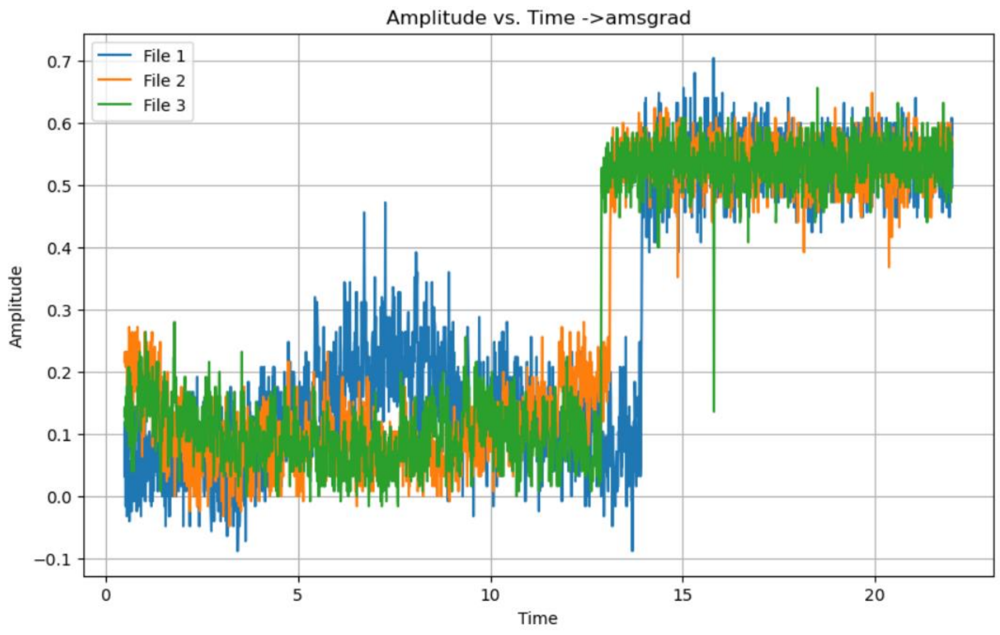
AdaBelief:



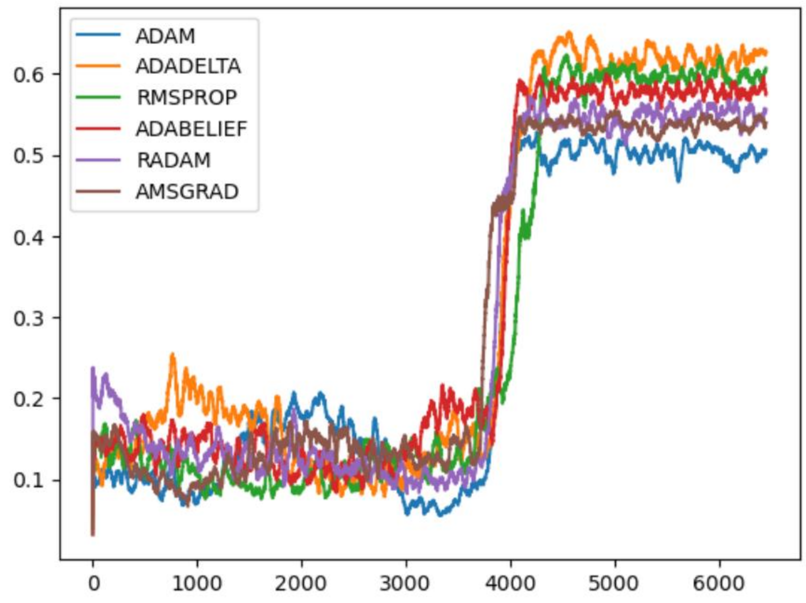
Radam:



AMSGrad:



As previously stated, phase locking experiments were conducted three times using each optimizer. The three cases' rms values were then plotted for each optimizer.



The plot clearly shows that the ADADelta optimizer achieves the highest power level, followed by the RMSprop. However, ADADelta is less stable than the other optimizers, as evidenced by the closed loop response. The plot shows that there is a step for AMSGrad and RMSprop, which implies that the optimizers first prefer to converge to the local maxima. Because of the random perturbations, they were able to surpass the local maxima and reach the global maxima.

At the of start the experiment the environmental conditions are as follows:

Temperature-24^{o} C

Humidity- 60%

At the end of the experiment the environmental conditions are as follows:

Temperature-26^{o}C

Humidity-55%

The phase locking using all the optimizers was done in a single session so that the environmental circumstances and noise variables remained comparable throughout the experiment.

3 OBSERVATION AND FINDINGS:

A brief understanding of the optimizers used so far:

1) <u>ADAM</u>

Phase locking: Adam quickly adjusts the phase differences due to its adaptive learning rate, making it suitable for systems where the phase differences change rapidly. However, there are some optimizers in terms of stability and ease of convergence.

2) <u>ADAGRAD</u>

Phase locking: AdaGrad is less effective in phase locking because it reduces the learning rate too aggressively over time, making it difficult to achieve fine-tuned phase adjustments.

3) <u>ADADELTA</u>

Phase Locking: Adadelta handles the dynamic adjustments required for phase locking more efficiently than AdaGrad, but it still faces challenges in highly dynamic phase environments due to its restricted window of gradient accumulation.

4) <u>RMSPROP</u>

Phase Locking: RMSProp is well-suited for phase locking in CBC as it adapts quickly to changes while avoiding the diminishing learning rate problem, providing a balance between stability and adaptability.

5) <u>ADABELIEF</u>

Phase Locking: AdaBelief focus on better generalization and stability, which helps in achieving robust phase locking by reducing the risk of overshooting and providing more reliable adjustments.

6) <u>RADAM</u>

Phase Locking: RAdam offers more stable phase locking performance by rectifying the adaptive learning rate, thus providing consistent adjustments without the initial instability that occurs in other adaptive optimizers.

7) <u>AMSGRAD</u>

Phase Locking: AMSGrad improves the reliability of phase locking by addressing the convergence issues of Adam, ensuring that the adjustments remain consistent and do not suffer from the diminishing variance problem.

4 CONCLUSION

The comparison of various optimization techniques for phase locking reveals distinct strengths and weaknesses. Adam is effective for rapid phase changes; however, it is less stable than some other optimizers. AdaGrad issues with phase locking since it aggressively reduces the learning rate. Adadelta outperforms AdaGrad but suffers difficulties in highly dynamic situations. RMSProp provides an excellent blend of stability and agility, making it ideal for phase locking. AdaBelief focuses on generalization and stability, which aids in durable phase transitions. RAdam maintains consistent performance by adjusting the adaptive learning rate. AMSGrad enhances dependability by addressing Adam's convergence concerns, resulting in consistent modifications without reducing variation.

5 REFERENCES

[1] Kingma, D. P., & Ba, J. (2014). *Adam: A Method for Stochastic Optimization*. http://arxiv.org/abs/1412.6980

[2] Duchi JDUCHI, J., & Singer, Y. (2011). Adaptive Subgradient Methods for Online Learning and Stochastic Optimization * Elad Hazan. In *Journal of Machine Learning Research* (Vol. 12).

[3] Zeiler, M. D. (2012). *ADADELTA: An Adaptive Learning Rate Method*. http://arxiv.org/abs/1212.5701

[4] Ruder, S. (2016). *An overview of gradient descent optimization algorithms*. http://arxiv.org/abs/1609.04747

[5] Zhuang, J., Tang, T., Ding, Y., Tatikonda, S., Dvornek, N., Papademetris, X., & Duncan, J. S. (2020). *AdaBelief Optimizer: Adapting Stepsizes by the Belief in Observed Gradients*. http://arxiv.org/abs/2010.07468

[6] Liu, L., Jiang, H., He, P., Chen, W., Liu, X., Gao, J., & Han, J. (2019). *On the Variance of the Adaptive Learning Rate and Beyond*. http://arxiv.org/abs/1908.03265

[7] Reddi, S. J., Kale, S., & Kumar, S. (2019). *On the Convergence of Adam and Beyond*. http://arxiv.org/abs/1904.09237