APPENDIX: APOLLO: AN ADAPTIVE PARAMETER-WISE DIAGONAL QUASI-NEWTON METHOD FOR NONCONVEX STOCHASTIC OPTIMIZATION

A COUPLED STEPSIZE AND CONVEXITY

Before proving Theorem 1, we first define the notations.

Let $\alpha = \frac{\eta'}{\eta} = \frac{\sigma'}{\sigma}$ be the ratio of the two sets of learning rates. Let $m'_t,\ d'_t$ and B'_t be the corresponding terms of parameter θ'_t at step t for (η', σ').

Proof of Theorem 1

Proof. Induction on the step of updates t, we attempt to prove that at each step t:

$$m_t = m'_t, \quad B'_t = \alpha B_t, \text{ and } \theta_t = \theta'_t, \ \forall t \tag{15}$$

Initial step: when $t = 1$, since $\theta_0 = \theta'_0$, we have $m_1 = m'_1$. With $d_0 = d'_0 = 0$ and (8), we have $B_1 = B'_1 = 0$ and:

$$D_1 = \text{rectify}(B_1, \sigma) = \sigma$$
$$D'_1 = \text{rectify}(B'_1, \sigma') = \sigma'$$

Then, $D'_1 = \alpha D_1$ and

$$\theta'_1 = \theta'_0 - \eta' D'^{-1}_1 m'_1 = \theta_0 - \eta\alpha(D_1^{-1}/\alpha)m_1 = \theta_0 - \eta D_1^{-1} m_1 = \theta_1.$$

Thus, the statement (15) is true.

Induction on step t: Suppose that the statement (15) is true for all the previous t steps. Now we prove the case $t + 1$. From the inductive assumption and (9), we have,

$$B'_t = \alpha B_t, \quad d'_t = \frac{1}{\alpha} d_t \text{ and } m_{t+1} = m'_{t+1}.$$

From (8),

$$\begin{aligned}
B'_{t+1} &= B'_t - \frac{d'^T_t y'_t + d'^T_t B'_t d'_t}{\|d'_t\|_4^4} \ \text{Diag}(d'^2_t) \\
&= \alpha B_t - \frac{(\frac{1}{\alpha} d_t)^T y_t + (\frac{1}{\alpha} d_t)^T (\alpha B_t)(\frac{1}{\alpha} d_t)}{\|(\frac{1}{\alpha} d_t)\|_4^4} \ \text{Diag}((\frac{1}{\alpha} d_t)^2) \\
&= \alpha B_t - \alpha \frac{d_t^T y_t + d_t^T B_t d_t}{\|d_t\|_4^4} \ \text{Diag}(d_t^2) \\
&= \alpha B_{t+1}.
\end{aligned}$$

Then,

$$\begin{aligned}
D'_{t+1} &= \text{rectify}(B'_{t+1}, \sigma') \\
&= \text{rectify}(\alpha B_{t+1}, \alpha\sigma) \\
&= \alpha \text{rectify}(B_{t+1}, \sigma) \\
&= \alpha D_{t+1}
\end{aligned}$$

and we have $\theta'_{t+1} = \theta_{t+1}$.

Finally, to sum up with the induction, we have proven Theorem 1. □

B CONVERGENCE ANALYSIS

B.1 CONVERGENCE ANALYSIS IN CONVEX OPTIMIZATION

Proof of Theorem 2

Proof. Let $\theta^* = \underset{\theta \in \mathcal{F}}{\operatorname{argmin}} \sum_{t=1}^{T} f_t(\theta)$, where $\mathcal{F}$ is the feasible set of θ. As $\theta_{t+1} - \theta^* = \theta_t - \theta^* - \eta_t D_t^{-1} m_t$ and $m_t = \beta_t m_{t-1} + (1-\beta_t) g_t$, we have the following:

$$\|D_t^{1/2}(\theta_{t+1} - \theta^*)\|_2^2 \leq \|D_t^{1/2}(\theta_t - \theta^*)\|_2^2 + \|\eta_t D_t^{-1/2} m_t\|_2^2 - 2\eta_t(\beta_t m_{t-1} + (1-\beta_t) g_t)^T(\theta_t - \theta^*)$$

Then, we have

$$\begin{aligned}
g_t^T(\theta_t - \theta^*) &\leq \frac{1}{2\eta_t(1-\beta_t)} \left[\|D_t^{1/2}(\theta_t - \theta^*)\|_2^2 - \|D_t^{1/2}(\theta_{t+1} - \theta^*)\|_2^2\right] \\
&\quad + \frac{\eta_t}{2(1-\beta_t)} \|D_t^{-1/2} m_t\|_2^2 - \frac{\beta_t}{1-\beta_t} m_{t-1}^T(\theta_t - \theta^*) \\
&\leq \frac{1}{2\eta_t(1-\beta_t)} \left[\|D_t^{1/2}(\theta_t - \theta^*)\|_2^2 - \|D_t^{1/2}(\theta_{t+1} - \theta^*)\|_2^2\right] \\
&\quad + \frac{\eta_t}{2(1-\beta_t)} \|D_t^{-1/2} m_t\|_2^2 + \frac{\eta_t}{2(1-\beta_t)} \|m_{t-1}\|_2^2 + \frac{\beta_t^2}{2\eta_t(1-\beta_t)} \|\theta_t - \theta^*\|_2^2
\end{aligned}$$

Using the standard approach of bounding the regret at each step with convexity of the functions $\{f_t\}_{t=1}^T$, we have the following bound of $R_T = \sum_{t=1}^{T} f_t(\theta_t) - f_t(\theta^*)$:

$$\begin{aligned}
\sum_{t=1}^{T} f_t(\theta_t) - f_t(\theta^*) &\leq \sum_{t=1}^{T} g_t^T(\theta_t - \theta^*) \\
&\leq \sum_{t=1}^{T} \frac{1}{2\eta_t(1-\beta_t)} \left[\|D_t^{1/2}(\theta_t - \theta^*)\|_2^2 - \|D_t^{1/2}(\theta_{t+1} - \theta^*)\|_2^2\right] \\
&\quad + \sum_{t=1}^{T} \frac{\eta_t}{2(1-\beta_t)} \|D_t^{-1/2} m_t\|_2^2 + \frac{\eta_t}{2(1-\beta_t)} \|m_{t-1}\|_2^2 \\
&\quad + \sum_{t=1}^{T} \frac{\beta_t^2}{2\eta_t(1-\beta_t)} \|\theta_t - \theta^*\|_2^2
\end{aligned} \tag{16}$$

As $\|\theta_t - \theta^*\|_2 \leq D$, $\beta_t < \beta < 1$ and $\|D_t\|_1/\eta_t \geq \|D_{t-1}\|_1/\eta_{t-1}$, we have

$$\begin{aligned}
&\sum_{t=1}^{T} \frac{1}{2\eta_t(1-\beta_t)} \left[\|D_t^{1/2}(\theta_t - \theta^*)\|_2^2 - \|D_t^{1/2}(\theta_{t+1} - \theta^*)\|_2^2\right] \\
=&\frac{\|D_1^{1/2}(\theta_1 - \theta^*)\|_2^2}{2\eta_1(1-\beta_1)} + \sum_{t=2}^{T} \left[\frac{\|D_t^{1/2}(\theta_t - \theta^*)\|_2^2}{2\eta_t(1-\beta_t)} - \frac{\|D_{t-1}^{1/2}(\theta_t - \theta^*)\|_2^2}{2\eta_{t-1}(1-\beta_{t-1})}\right] \\
\leq&\frac{\|D_1^{1/2}(\theta_1 - \theta^*)\|_2^2}{2\eta_1(1-\beta)} + \frac{1}{2(1-\beta)} \sum_{t=2}^{T} \left[\frac{\|D_t^{1/2}(\theta_t - \theta^*)\|_2^2}{\eta_t} - \frac{\|D_{t-1}^{1/2}(\theta_t - \theta^*)\|_2^2}{\eta_{t-1}}\right] \\
\leq&\frac{\|(\theta_1 - \theta^*)\|_2^2}{2\eta_1(1-\beta)} \|D_1^{1/2}\|_2^2 + \frac{1}{2(1-\beta)} \sum_{t=2}^{T} \|(\theta_t - \theta^*)\|_2^2 \left[\frac{\|D_t^{1/2}\|_2^2}{\eta_t} - \frac{\|D_{t-1}^{1/2}\|_2^2}{\eta_{t-1}}\right]
\end{aligned} \tag{17}$$

Since $\|D_t^{1/2}\|_2^2 = \|D_t\|_1$, we can rewrite the RHS of (17) as:

$$\begin{aligned}
&\frac{\|(\theta_1-\theta^*)\|_2^2}{2\eta_1(1-\beta)}\|D_1^{1/2}\|_2^2 + \frac{1}{2(1-\beta)}\sum_{t=2}^{T}\|(\theta_t-\theta^*)\|_2^2\left[\frac{\|D_t^{1/2}\|_2^2}{\eta_t} - \frac{\|D_{t-1}^{1/2}\|_2^2}{\eta_{t-1}}\right] \\
=&\frac{\|(\theta_1-\theta^*)\|_2^2}{2\eta_1(1-\beta)}\|D_1\|_1 + \frac{1}{2(1-\beta)}\sum_{t=2}^{T}\|(\theta_t-\theta^*)\|_2^2\left[\frac{\|D_t\|_1}{\eta_t} - \frac{\|D_{t-1}\|_1}{\eta_{t-1}}\right] \\
\leq&\frac{D^2}{2\eta_1(1-\beta)}\|D_1\|_1 + \frac{D^2}{2(1-\beta)}\sum_{t=2}^{T}\left[\frac{\|D_t\|_1}{\eta_t} - \frac{\|D_{t-1}\|_1}{\eta_{t-1}}\right] \\
=&\frac{D^2\|D_T\|_1}{2\eta_T(1-\beta)} = \frac{\sqrt{T}D^2\|D_T\|_1}{2\eta(1-\beta)}
\end{aligned} \tag{18}$$

To sum up with (16) and (18), we have

$$\begin{aligned}
R_T = \sum_{t=1}^{T} f_t(\theta_t) - f_t(\theta^*) \leq& \frac{\sqrt{T}D^2\|D_T\|_1}{2\eta(1-\beta)} \\
&+ \sum_{t=1}^{T}\frac{\eta_t}{2(1-\beta_t)}\|D_t^{-1/2}m_t\|_2^2 + \frac{\eta_t}{2(1-\beta_t)}\|m_{t-1}\|_2^2 \\
&+ \sum_{t=1}^{T}\frac{\beta_t^2}{2\eta_t(1-\beta_t)}\|\theta_t-\theta^*\|_2^2
\end{aligned} \tag{19}$$

Since the element of D_t is rectified by 1, i.e. $D_{t,i} \geq 1$, and $\|m_t\|_2 \leq G$, $\beta_t < \beta < 1$, we have

$$\begin{aligned}
\sum_{t=1}^{T}\frac{\eta_t}{2(1-\beta_t)}\|D_t^{-1/2}m_t\|_2^2 + \frac{\eta_t}{2(1-\beta_t)}\|m_{t-1}\|_2^2 &\leq \sum_{t=1}^{T}\frac{\eta_t}{2(1-\beta_t)}\|m_t\|_2^2 + \frac{\eta_t}{2(1-\beta_t)}\|m_{t-1}\|_2^2 \\
&\leq \frac{G^2}{1-\beta}\sum_{t=1}^{T}\eta_t = \frac{\eta G^2}{1-\beta}\sum_{t=1}^{T}\frac{1}{\sqrt{t}} \\
&\leq \frac{\eta G^2}{1-\beta}(2\sqrt{T}-1)
\end{aligned} \tag{20}$$

The last inequality is due to the following upper bound:

$$\sum_{t=1}^{T}\frac{1}{\sqrt{t}} \leq \int_{t=1}^{T}\frac{\mathrm{d}t}{\sqrt{t}} = 2\sqrt{T}-1$$

Again, as $\|\theta_t-\theta^*\|_2 \leq D$ and $\beta_t < \beta < 1$, we have

$$\sum_{t=1}^{T}\frac{\beta_t^2}{2\eta_t(1-\beta_t)}\|\theta_t-\theta^*\|_2^2 \leq \frac{D^2}{2(1-\beta)}\sum_{t=1}^{T}\frac{\beta_t^2}{\eta_t} \tag{21}$$

Finally, to sum up with (19), (20) and (21), we have

$$R_T \leq \frac{\sqrt{T}D^2\|D_T\|_1}{2\eta(1-\beta)} + \frac{\eta G^2}{1-\beta}(2\sqrt{T}-1) + \frac{D^2}{2(1-\beta)}\sum_{t=1}^{T}\frac{\beta_t^2}{\eta_t}$$

□

Proof of Corollary 2.1

Proof. Since $\beta_t = \beta\lambda^{t-1}$, by sum of arithmetico-geometric series we have

$$\sum_{t=1}^{T}\lambda^{2(t-1)}\sqrt{t} \leq \sum_{t=1}^{T}\lambda^{2(t-1)}t \leq \frac{1}{(1-\lambda^2)^2} \tag{22}$$

Plugging (22) into (21), we have

$$R_T \leq \frac{\sqrt{T}D^2\|D_T\|_1}{2\eta(1-\beta)} + \frac{\eta G^2}{1-\beta}(2\sqrt{T}-1) + \frac{D^2\beta^2}{2\eta(1-\beta)(1-\lambda^2)^2}.$$

□

B.2 Convergence Analysis in Nonconvex Stochastic Optimization

To prove Theorem 3 in §3.5, we first describe the Theorem 3.1 in (Chen et al., 2019):

Theorem 3.1 (Chen et al., 2019) For an Adam-type method under the assumptions:
• f is lower bounded and differentiable; $\|\nabla f(\theta) - \nabla f(\theta')\|_2 \leq L\|\theta - \theta'\|_2,\ \forall \theta,\ \theta'$.
• Both the true and stochastic gradient are bounded, i.e. $\|\nabla f(\theta_t)\|_2 \leq H,\ \|g_t\|_2 \leq H,\ \forall t$.
• Unbiased and independent noise in g_t, i.e. $g_t = \nabla f(\theta_t) + \zeta_t$, $\mathbb{E}[\zeta_t] = 0$, and $\zeta_i \perp \zeta_j,\ \forall i \neq j$.
Assume $\beta_t \leq \beta \leq 1$ in non-increasing, $\|\eta_t m_t/\sqrt{v_t}\|_2 \leq G$, then:

$$\begin{aligned}
&\mathbb{E}\left[\sum_{t=1}^{T} \eta_t \langle \nabla f(\theta_t), \nabla f(\theta_t)/\sqrt{v_t}\rangle\right] \\
\leq &\mathbb{E}\left[C_1 \sum_{t=1}^{T}\left\|\frac{\eta_t g_t}{\sqrt{v_t}}\right\|_2^2 + C_2 \sum_{t=2}^{T}\left\|\frac{\eta_t}{\sqrt{v_t}} - \frac{\eta_{t-1}}{\sqrt{v_{t-1}}}\right\|_1 + C_3 \sum_{t=2}^{T-1}\left\|\frac{\eta_t}{\sqrt{v_t}} - \frac{\eta_{t-1}}{\sqrt{v_{t-1}}}\right\|_2^2\right] + C_4
\end{aligned} \tag{23}$$

where C_1, C_2, C_3 are constants independent of d and T, C_4 is a constant independent of T, the expectation is taken w.r.t all the randomness corresponding to $\{g_t\}$.

Since APOLLO belongs to the family of *generalized Adam* (Chen et al., 2019) with $\sqrt{v_t}$ corresponding to D_t, we have

$$\begin{aligned}
&\mathbb{E}\left[\sum_{t=1}^{T} \eta_t \langle \nabla f(\theta_t), \nabla f(\theta_t)/D_t\rangle\right] \\
\leq &\mathbb{E}\left[C_1 \sum_{t=1}^{T}\left\|\frac{\eta_t g_t}{D_t}\right\|_2^2 + C_2 \sum_{t=2}^{T}\left\|\frac{\eta_t}{D_t} - \frac{\eta_{t-1}}{D_{t-1}}\right\|_1 + C_3 \sum_{t=2}^{T-1}\left\|\frac{\eta_t}{D_t} - \frac{\eta_{t-1}}{D_{t-1}}\right\|_2^2\right] + C_4
\end{aligned} \tag{24}$$

Note that APOLLO does not specify the bound of each update $\|\eta_t m_t/D_t\|_2$, because it is straightforward to derive the bound with conditions $\eta_t \leq \eta$, $\|g_t\|_2 \leq H$ and $D_t \geq 1$.

Proof of Theorem 3 With (24), we can prove our Theorem 3 with similar derivations in Chen et al. (2019).

Proof. We first bound non-constant terms in RHS of (24), which is given by

$$\mathbb{E}\left[C_1 \sum_{t=1}^{T}\left\|\frac{\eta_t g_t}{D_t}\right\|_2^2 + C_2 \sum_{t=2}^{T}\left\|\frac{\eta_t}{D_t} - \frac{\eta_{t-1}}{D_{t-1}}\right\|_1 + C_3 \sum_{t=2}^{T-1}\left\|\frac{\eta_t}{D_t} - \frac{\eta_{t-1}}{D_{t-1}}\right\|_2^2\right]$$

For the term with C_1, since $D_t \geq 1$, we have

$$\begin{aligned}
\mathbb{E}\left[\sum_{t=1}^{T}\left\|\frac{\eta_t g_t}{D_t}\right\|_2^2\right] &\leq \mathbb{E}\left[\sum_{t=1}^{T}\|\eta_t g_t\|_2^2\right] \\
&= \mathbb{E}\left[\sum_{t=1}^{T}\left\|\left(\frac{\eta}{\sqrt{t}}\right) g_t\right\|_2^2\right] \\
&\leq \eta^2 H^2 \sum_{t=1}^{T}\frac{1}{t} \leq \eta^2 H^2(1+\log T)
\end{aligned} \tag{25}$$

where the last inequality is due to $\sum_{t=1}^{T} 1/t \leq 1 + \log T$.
For the term with C_2, we have

$$\begin{aligned}\mathbb{E}\left[\sum_{t=2}^{T}\left\|\frac{\eta_t}{D_t}-\frac{\eta_{t-1}}{D_{t-1}}\right\|_1\right] &= \mathbb{E}\left[\sum_{j=1}^{d}\sum_{t=2}^{T}\left(\frac{\eta_{t-1}}{D_{t-1,j}}-\frac{\eta_t}{D_{t,j}}\right)\right] \\ &= \mathbb{E}\left[\sum_{j=1}^{d}\left(\frac{\eta_1}{D_{1,j}}-\frac{\eta_T}{D_{T,j}}\right)\right] = \mathbb{E}\left[\sum_{j=1}^{d}\frac{\eta_1}{D_{1,j}}\right] \leq d\eta\end{aligned} \tag{26}$$

where the first equality is due to $\frac{D_{t-1,j}}{\eta_{t-1}} \leq \frac{D_{t,j}}{\eta_t}$, $\forall t \in [T], j \in [d]$ and the second equality is due to telescope sum.
For the term with C_3, we have

$$\mathbb{E}\left[\sum_{t=2}^{T-1}\left\|\frac{\eta_t}{D_t}-\frac{\eta_{t-1}}{D_{t-1}}\right\|_2^2\right] \leq \mathbb{E}\left[\eta\sum_{t=2}^{T-1}\left\|\frac{\eta_t}{D_t}-\frac{\eta_{t-1}}{D_{t-1}}\right\|_1\right] \leq d\eta^2 \tag{27}$$

where the first inequality is due to $|\eta_{t-1}/D_{t-1,j} - \eta_t/D_{t,j}| \leq \eta$.
Then for APOLLO we have

$$\begin{aligned}&\mathbb{E}\left[C_1\sum_{t=1}^{T}\left\|\frac{\eta_t g_t}{D_t}\right\|_2^2 + C_2\sum_{t=2}^{T}\left\|\frac{\eta_t}{D_t}-\frac{\eta_{t-1}}{D_{t-1}}\right\|_1 + C_3\sum_{t=2}^{T-1}\left\|\frac{\eta_t}{D_t}-\frac{\eta_{t-1}}{D_{t-1}}\right\|_2^2\right] + C_4 \\ \leq& C_1\eta^2H^2(1+\log T) + C_2 d\eta + C_3 d\eta^2 + C_4\end{aligned} \tag{28}$$

Now we lower bound the LHS of (24). With the assumption $\|D_t\|_\infty \leq L$, we have

$$(\eta_t/D_t)_j \geq \frac{\eta}{L\sqrt{t}}$$

And thus

$$\mathbb{E}\left[\sum_{t=1}^{T}\eta_t\langle\nabla f(\theta_t), \nabla f(\theta_t)/D_t\rangle\right] \geq \mathbb{E}\left[\sum_{t=1}^{T}\frac{\eta}{L\sqrt{t}}\|\nabla f(\theta_t)\|_2^2\right] \geq \frac{\sqrt{T}}{L}\min_{t\in[T]}\mathbb{E}\left[\|\nabla f(\theta_t)\|_2^2\right] \tag{29}$$

Then, to sum up with (24), (28) and (29), we have

$$\frac{\sqrt{T}}{L}\min_{t\in[T]}\mathbb{E}\left[\|\nabla f(\theta_t)\|_2^2\right] \leq C_1\eta^2H^2(1+\log T) + C_2 d\eta + C_3 d\eta^2 + C_4$$

which is equivalent to

$$\begin{aligned}\min_{t\in[T]}\mathbb{E}\left[\|\nabla f(\theta_t)\|_2^2\right] &\leq \frac{L}{\sqrt{T}}(C_1\eta^2H^2(1+\log T) + C_2 d\eta + C_3 d\eta^2 + C_4) \\ &= \frac{1}{\sqrt{T}}(Q_1 + Q_2\log T)\end{aligned}$$

This completes the proof. □

C EXTENSIONS AND FUTURE WORK

Parameter-Wise Gradient Clipping. The standard gradient clipping method (Pascanu et al., 2013) is to clip the gradients based on the norm computed over gradients of all the parameters together. A modification of gradient clipping to properly apply it to APOLLO is to clip the gradient of each parameter individually based on its own norm. Preliminary results are provided in Appendix F.4.

Decoupled Weight Decay in APOLLO. (Loshchilov & Hutter, 2019) demonstrated that L_2 regularization is not identical to weight decay for adaptive gradient methods and proposed Adam with decoupled weight decay (AdamW). The application of decoupled weight decay to APOLLO is slightly different from AdamW as APOLLO memorizes the update direction of the last iteration d_t to update the diagonal Hessian. The algorithm of APOLLO with decoupled weight decay is in Appendix D.

Making APOLLO Scale-Invariant. An important advantage of adaptive optimization methods, including Adam and its variants, is that they are inherently scale-invariant — invariant with the scale of the objective function. The property of scale-invariance yields more consistent hyper-parameters of these adaptive methods than SGD across different machine learning tasks. Unfortunately, APOLLO does not hold the property of scale-invariance, and we need to ask if it is possible to make APOLLO scale-invariant. Interestingly, it is quite easy to develop a scale-invariant version of APOLLO by applying a simple modification. We provide more details about scale-invariant APOLLO in Appendix H.

D APOLLO WITH DECOUPLED WEIGHT DECAY

Algorithm 2: APOLLO with weight decay (L_2/Decoupled)

Initial: $m_0, d_0, B_0 \leftarrow 0, 0, 0$ // Initialize m_0, d_0, B_0 to zero
while $t \in \{0, \dots, T\}$ **do**
  for $\theta \in \{\theta^1, \dots, \theta^L\}$ **do**
    $g_{t+1} \leftarrow \nabla f_t(\theta_t) + \gamma\theta_t$ // Calculate gradient at step t
    $m_{t+1} \leftarrow \frac{\beta(1-\beta^t)}{1-\beta^{t+1}} m_t + \frac{1-\beta}{1-\beta^{t+1}} g_{t+1}$ // Update bias-corrected moving
    $\alpha \leftarrow \frac{d_t^T (m_{t+1} - m_t) + d_t^T B_t d_t}{(\|d_t\|_4 + \epsilon)^4}$ // Calculate coefficient of B update
    $B_{t+1} \leftarrow B_t - \alpha \cdot \mathrm{Diag}(d_t^2)$ // Update diagonal Hessian
    $D_{t+1} \leftarrow \mathrm{rectify}(B_{t+1}, 0.01)$ // Handle nonconvexity
    $d_{t+1} \leftarrow D_{t+1}^{-1} m_{t+1} + \gamma\theta_t$ // Calculate update direction
    $\theta_{t+1} \leftarrow \theta_t - \eta_{t+1} d_{t+1}$ // Update parameters
  end
end
return θ_T

Algorithm 2 illustrates the algorithm of APOLLO with the standard L_2 and the decoupled weight decay. As APOLLO memorizes the update direction of the last iteration d_t to update the diagonal Hessian B_{t+1}, the application of decoupled weight decay to APOLLO is slightly different from AdamW. The weight decay term is added to the update direction d_t, instead of directly to the update of parameters. We conducted experiments to evaluate APOLLO with decoupled weight decay on image classification tasks. The results are provided in Appendix F.

E EXPERIMENTAL DETAILS

E.1 IMAGE CLASSIFICATION

CIFAR-10 For CIFAR-10 dataset, we use the ResNet-110 architecture in the public implementation[5]. Note that ResNet-110 is a modified version of ResNet-18 (He et al., 2016) to adapt the small image size 32×32 in CIFAR-10, and is much smaller than standard ResNet-18. The number of parameters for ResNet-110 and ResNet-18 are 1.73 M and 11.69 M, respectively. The implementation of AdaHessian is based on the public implementation[6]. The training batch size is set to 128. For each optimizer, we used two learning rate decay strategies. First, we train the model on CIFAR-10 for 164 epochs and decay the learning rate at the end of 80-th and 120-th epochs by 0.1. Second, we also used the cosine annealing schedule (Loshchilov & Hutter, 2017). For the cosine annealing schedule, we train a CIFAR-10 model for 200 epochs.

For every optimizer, we comprehensively tuned its hyper-parameters and selected the set of hyper-parameters with the optimal classification accuracy. Concretely, for SGD, we fixed momentum at 0.9 and perform grid search of learning rate $\eta \in \{0.05,\ 0.1,\ 0.2,\ 0.5\}$, weight decay rate $\gamma \in [1e^{-4},\ 1e^{-3}]$ with step $1e^{-4}$. For Adam and RAdam, we fixed $\beta_1 = 0.9,\ \beta_2 = 0.999,\ \epsilon = 1e^{-8}$ and grid search learning rate $\eta \in \{1e^{-4},\ 5e^{-4},\ 1e^{-3},\ 5e^{-3},\ 1e^{-2}\}$, weight decay rate $\gamma \in [1e^{-2},\ 5e^{-1}]$ with step $1e^{-2}$. For AdaBelief, we fixed $\beta_1 = 0.9,\ \beta_2 = 0.999$, and grid search

[5] https://github.com/bearpaw/pytorch-classification
[6] https://github.com/davda54/ada-hessian

Table 4: Hyper-parameters of each optimizer on CIFAR-10 and ImageNet.

	CIFAR-10	**ImageNet**
SGD	$\eta = 0.1,\ \gamma = 5e^{-4},$	$\eta = 0.1,\ \gamma = 1e^{-4}$
Adam	$\eta = 0.001,\ \gamma = 2.5e^{-1},\ \epsilon = 1e^{-8}$	$\eta = 0.001,\ \gamma = 1e^{-1},\ \epsilon = 1e^{-8}$
RAdam	$\eta = 0.001,\ \gamma = 2.5e^{-1},\ \epsilon = 1e^{-8}$	$\eta = 0.001,\ \gamma = 1e^{-1},\ \epsilon = 1e^{-8}$
AdaBelief.	$\eta = 0.001,\ \gamma = 2.5e^{-1},\ \epsilon = 1e^{-8}$	$\eta = 0.001,\ \gamma = 1e^{-1},\ \epsilon = 1e^{-8}$
AdaHessian	$\eta = 0.15,\ \gamma = 1e^{-3},\ \epsilon = 1e^{-2}$	—
APOLLO	$\eta = 0.01,\ \gamma = 2.5e^{-4},\ \epsilon = 1e^{-4},$	$\eta = 0.01,\ \gamma = 1e^{-4},\ \epsilon = 1e^{-4}$
APOLLOW	$\eta = 0.01,\ \gamma = 2.5e^{-2},\ \epsilon = 1e^{-4},$	$\eta = 0.01,\ \gamma = 1e^{-2},\ \epsilon = 1e^{-4}$

learning rate $\eta \in \{1e^{-4},\ 5e^{-4},\ 1e^{-3},\ 5e^{-3},\ 1e^{-2}\}$, and $\epsilon \in \{1e^{-6},\ 1e^{-8},\ 1e^{-12}\}$. For weight decay, we tried both the standard L_2 and the decoupled version of weight decay. For L_2, we search weight decay rate $\gamma \in [1e^{-4},\ 1e^{-3}]$ with step $1e^{-4}$, and for decoupled version we search weight decay rate $\gamma \in [1e^{-2},\ 5e^{-1}]$ with step $1e^{-2}$. For AdaHessian, we fixed $\beta_1 = 0.9,\ \beta_2 = 0.999$ and grid search $\eta \in \{0.05,\ 0.1,\ 0.15,\ 0.2\}$, weight decay rate $\gamma \in [5e^{-4},\ 5e^{-3}]$ with step $5e^{-4}$ and $\epsilon \in \{1e^{-2},\ 1e^{-4},\ 1e^{-6}\}$. For APOLLO, we fixed $\beta = 0.9,\ \epsilon = 1e^{-4}$ and grid search learning rate $\eta \in \{0.001,\ 0.005,\ 0.01,\ 0.02\}$, weight decay rate $\gamma \in [5e^{-5},\ 1e^{-3}]$ with step $5e^{-5}$. We explored applying learning rate warmup to all the optimizers and found that APOLLO and AdaHessian significantly benefit from warmup. The impact of warmup on other optimizers is marginal. Thus, for APOLLO and AdaHessian, learning rates are warmed up linearly in the first 500 updates. The selected optimal hyper-parameters for each optimizer are summarized in Table 4 Random cropping and random horizontal flipping are applied to training data. For each experiment, we conduct training on one NVIDIA Tesla V100 GPU.

ImageNet For ImageNet, we used the neural architecture of ResNeXt-50 (Xie et al., 2017), with training batch size 256. For each optimizer, we also used the two learning rate decay strategies — milestone and cosine. For milestone decay, we train the model for 120 epochs and decay the learning rate at at the end of 40-th and 80-th epochs by 0.1. For cosine annealing, we also train each model for 120 epochs with the cosine annealing schedule. For each optimizer, we fixed all the hyper-parameters selected from CIFAR-10 experiments, except the rate of weight decay γ which is tuned on the classification accuracy. Random cropping and random horizontal flipping are applied to training data. For each experiment, we conduct training on eight NVIDIA Tesla V100 GPUs.

E.2 LANGUAGE MODELING

One Billion Words dataset (Chelba et al., 2013) is a publicly available benchmark for measuring progress of language modeling. It contains about 0.8 billion tokens with a vocabulary of 793,471 words, including sentence boundary markers. Different from Liu et al. (2020) which shrinks the vocabulary to about 0.64 million words, we used the standard vocabulary[7]. For the language model, we used two-layer LSTM with 2048 hidden states with adaptive softmax and 300-dimensional word embeddings as input. The cut-offs of the adaptive softmax are set to $[60000,\ 100000,\ 640000]$, which is different from Liu et al. (2020). Dropout (Srivastava et al., 2014) is applied to each layer with drop rate of 0.1. No weight decay is applied to these optimizers. Gradient clips with 1.0 are applied to all the optimization methods.

For each optimizer, we comprehensively tuned its learning rate. Concretely, for SGD, we searched the learning rate $\eta \in \{0.05,\ 0.1,\ 0.5,\ 1.0\}$ and $\eta = 0.5$ was selected. For Adam, RAdam and AdaBelief, we fixed $\beta_1 = 0.9,\ \beta_2 = 0.999$, and searched for $\eta \in \{5e^{-4},\ 1e^{-3},\ 2e^{-3},\ 5e^{-3}\}$, and finally $\eta = 1e^{-3}$ was selected. In addition, following Zhuang et al. (2020), we also tuned ϵ for AdaBelief (for Adam and RAdam, we fixed $\epsilon = 1e^{-8}$). We searched $\epsilon \in \{1e^{-8},\ 1e^{-12},\ 1e^{-16}\}$ and found that $\epsilon = 1e^{-12}$ worked best. It should be noticed that AdaBelief is very sensitive to the value of ϵ. The result in Table 2 is obtained using $\epsilon = 1e^{-12}$. With other values, e.g. $1e^{-8}$ or $1e^{-16}$, the PPL points of AdaBelief are even higher than Adam and RAdam. Thus, we suspected that the

[7] https://github.com/rafaljozefowicz/lm/blob/master/1b_word_vocab.txt

improvement of AdaBelief over Adam or RAdam on LSTM mainly comes from the fine-tuning of ϵ. Similar observations were also found in our experiments of image classification, and were reported in Yuan & Gao (2020). For APOLLO, we fixed $\beta = 0.9, \epsilon = 1e^{-4}$, and searched $\eta \in \{0.01,\ 0.05,\ 0.1\}$. Finally, $\eta = 0.1$ was selected. Each model is trained for 20 epochs, and the learning rate decays at the end of the 12-th and 18-th epochs by decay rate 0.1. LSTMs are unrolled for 20 steps without resetting the LSTM states and the batch size is set to 128. Every models is trained on one NVIDIA Titan RTX GPU.

E.3 NEURAL MACHINE TRANSLATION

Our experiments on WMT 2014 English-German are based on the Transformer-base model (Vaswani et al., 2017), with implementation from the FairSeq package (Ott et al., 2019). This dataset contains 4.5M parallel sentence pairs for training. We following the standard setting (Vaswani et al., 2017), using Newstest2013 as the validation set and Newstest2014 as the test set. The dataset is pre-processed following (Ma et al., 2019), using the scripts from FairSeq package[8]. Specifically, we use word embedding with 512 dimension and 6-layer encoder/decoder with 8 multi-head attention and 2048 feed-forward dimensions. We apply 0.1 label smoothing (Szegedy et al., 2016), and perform totally $500,000$ updates to train each model. For Adam, RAdam and AdaBelief, we use start learning rate 0.0005. For Adam we set $\beta = (0.9, 0.98)$, while for RAdam and AdaBelief we set $\beta = (0.9, 0.999)$. For SGD and APOLLO, the start learning rates is 0.1. The momentum of SGD is 0.9. For learning rate scheduling, we applied linear warm up the learning rate for SGD, Adam, AdaBelief, and APOLLO — 4000 updates for Adam and 1000 updates for SGD, AdaBelief and APOLLO. For RAdam, we did not apply warm up because RAdam is inherently designed to avoid it. After learning rate warming up, we applied the inverse square root decay (Vaswani et al., 2017) to Adam. For SGD, RAdam, AdaBelief and APOLLO, we decayed the learning rate at the $300,000$ and $450,000$ updates by decay rate 0.1. Gradient clips with 1.0 are applied to all the optimization methods, and the dropout ratio are set to 0.1. Weight decay rates are $1e^{-4}$ for Adam-type methods, $1e^{-6}$ for SGD, and $1e^{-8}$ for APOLLO. The decoding beam size is set to 5, and the checkpoints of the last 10 epochs are averaged before evaluation. For each experiment, we conducted distributed training across eight NVIDIA Tesla V100 GPUs with maximum batch size as 8192 tokens per GPU (totally 8192×8 tokens per batch).

E.4 THE CHOICE OF σ

In our final version, we change σ from 1.0 to 0.01 to make the learning rate η of APOLLO in a suitable range. Concretely, in the case of $\sigma = 1.0$, the optimal η for image classification, language modeling and machine translation are 1.0, 10.0 and 10.0, respectively. These values are very different from previous algorithms. After we changed $\sigma = 0.01$, the optimal η for the three tasks because 0.01, 0.1 and 0.1, which are in a more acceptable range. Note that we change $sigma = 0.01$ only for the consideration of the convenient application of APOLLO. It has no affect on the behavior of the algorithm.

F DETAILED EXPERIMENTAL RESULTS

In this section, we report the detailed experimental results in Section 4, and the results of investigation of the effect of weight decay.

F.1 DETAILED RESULTS ON IMAGE CLASSIFICATION

Figure 4 and Table 5 illustrate the details of the experimental results on Image Classification. For each experiment, we report the mean values with corresponding standard deviations over 5 runs. Though Loshchilov & Hutter (2019) claimed that the optimal settings of the learning rate and weight decay factor in Adam with decoupled weight decay is more independent than the original Adam, we observed that the strength of weight decay regularization is still co-related with the learning rate. To illustrate the significant effect of weight decay strength on both the performance of convergence and generalization, we also report the performance of Adam and RAdam with the same weight decay rate of SGD, named Adam* and RAdam*.

[8] https://github.com/pytorch/fairseq

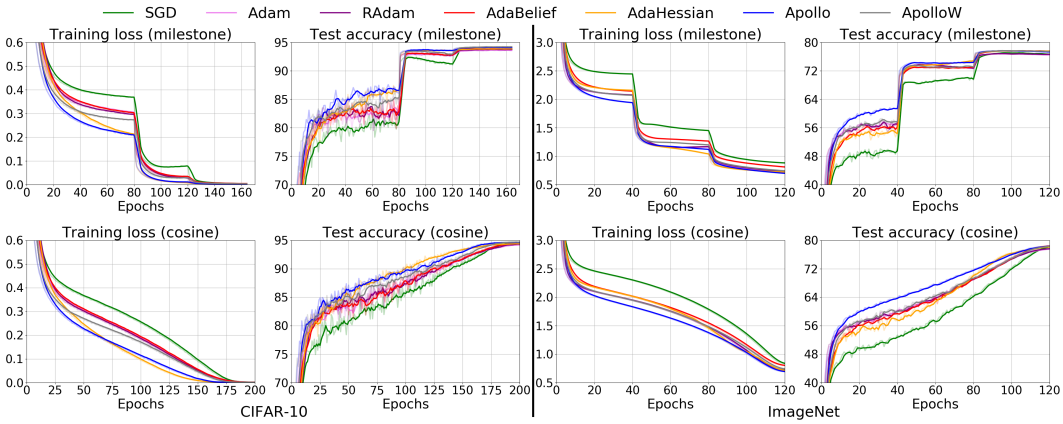

Figure 4: Training loss and test accuracy of ResNet-110 on CIFAR-10 and ResNeXt-50 on ImageNet, with two schedule strategies of learning rate decay.

Table 5: Classification accuracy on CIFAR-10 and ImageNet. For each experiment, we report the mean and standard variance over 5 runs.

Method	**CIFAR-10**		**ImageNet**	
	milestone decay	**cosine annealing**	**milestone decay**	**cosine annealing**
SGD	93.94±0.07	94.53±0.27	77.57±0.07	78.26±0.08
Adam*	91.41±0.30	91.56±0.19	71.72±0.13	71.19±0.10
RAdam*	91.80±0.04	91.88±0.15	72.37±0.08	71.64±0.14
Adam	93.74±0.15	94.24±0.09	76.86±0.06	77.54±0.16
RAdam	93.88±0.11	94.38±0.25	76.91±0.07	77.68±0.08
AdaBelief	94.03±0.11	94.51±0.07	77.55±0.08	78.22±0.11
AdaHessian	93.97±0.22	94.48±0.17	77.61±0.09	78.02±0.10
APOLLO	**94.21**±0.08	**94.64**±0.09	**77.85**±0.07	**78.45**±0.06
APOLLOW	**94.34**±0.12	**94.76**±0.07	**77.86**±0.09	**78.48**±0.07

From Figure 4 and Table 5, we see that Adam* and RAdam*, with the same weight decay rate of SGD, converge much faster than other optimization methods, while obtaining significantly worse classification accuracy. After adjusting the weight decay rates, the test accuracy of Adam and RAdam remarkably improves, with rapid decline of convergence speed. This suggests that the fast convergence speed of Adam and RAdam results from relatively weak regularization. Thus, the effect of regularization strength needs to be considered when we analyze the performance of different optimization methods.

In addition, we also report the results of APOLLO with decoupled weight decay, which is denoted as APOLLOW. The hyper-parameters of APOLLOW (see Table 4) are exactly the same of the optimal ones of APOLLO. From Figure 4 and Table 5, we see that APOLLO with the standard L_2 regularization achieves faster convergence speed, while APOLLOW with the decoupled weight decay achieves slightly better generalization accuracy. Importantly, comparing with Adam-type methods whose performance is significantly impacted by different weight decay implementations, APOLLO is much more consistent with the two implementations of weight decay.

F.2 EFFECT OF WEIGHT DECAY RATE ON OPTIMIZATION

To further investigate the effect of weight decay rate on converge speed and generalization performance for different optimization methods, we conduct experiments on CIFAR-10 of ResNet-110 with a range of weight decay rates. Concretely, we use the weight decay rates γ in Table 4 as the base, and explore different γ that are α times of the base weight decay rate, with $\alpha \in \{0.2, 0.6, 1.0, 1.4, 1.8\}$.

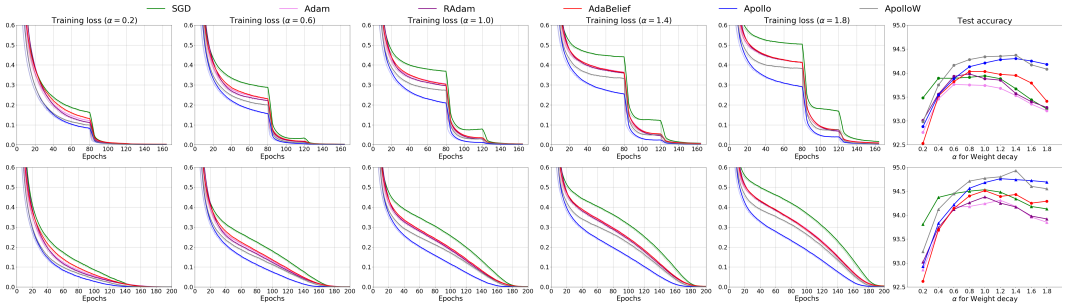

Figure 5: Training loss and test accuracy of ResNet-110 on CIFAR-10 with various rates of weight decay, with two schedule strategies of learning rate decay.

Figure 5 shows the convergence of different optimization methods with various rates of weight decay, together with the classification accuracy. APOLLO achieves improvements over all the four baselines on convergence speed with different rates of weight decay. For classification accuracy, APOLLO obtains the best accuracy when the weight decay rate ratio α is larger than 0.3. When the weight decay rate is decreasing, SGD obtains the best accuracy, while APOLLO achieves comparable performance.

F.3 COMPARISON ON TRAINING SPEED AND MEMORY COST

In this section, we compare the training speed and memory between SGD, Adam, AdaHessian and APOLLO. Table 6 summarizes the comparison of cost of a single iteration of update. Note that the cost measured in our experiments includes all aspects of model training, including the forward and backward pass of DNNs, not only that of updating parameters for an optimizer. For fair comparison, experiments of CIFAR-10 and One Billion Words are conduced on a single NVIDIA TITAN RTX GPU, while experiments of ImageNet and WMT are performed with distributed training on 8 NVIDIA Tesla V100 GPUs.

Table 6: Comparison between different optimization methods on training speed and memory cost. Cost includes all aspects of model training, not only that of an optimizer.

	CIFAR-10		**ImageNet**		**1BW**		**WMT-14**	
Cost ($\times$SGD)	Speed	Memory	Speed	Memory	Speed	Memory	Speed	Memory
SGD	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Adam	1.16	1.01	1.01	1.03	1.19	1.34	1.13	1.04
Apollo	1.42	1.01	1.23	1.05	1.49	1.62	1.19	1.06
AdaHessian	5.76	2.12	11.78	2.51	3.51	2.78	8.46	2.47

From Table 6, we see that the second-order AdaHessian requires much more computational resource than first-order methods on both time and memory. In addition, the slow-down of AdaHessian becomes more significant for larger-scale models with distributed training across multiple GPUS, such as ResNext-50 on ImageNet and Transformer on WMT.

F.4 EXPERIMENTS ON PARAMETER-WISE GRADIENT CLIPPING

In this section, we provide some preliminary results on parameter-wise gradient clipping, a modification of the standard gradient clipping that is inherently proper to APOLLO. Parameter-wise gradient clipping is to clip the gradient of each parameter individually based on its own norm. It can be regarded as a trade-off between gradient clipping by global norm and by each value.

We conducted two groups of experiments to compare with the standard gradient clipping method — language modeling and neural machine translation. The experimental settings for standard gradient clipping are exactly the same as in section 4, where we clipped the gradient by global norm 1.0 for each model. For parameter-wise gradient clipping, we clipped each parameter by 0.5 for the LSTM model in language modeling, and 0.1 for the Transformer-base model in NMT.

Table 7: Comparison between APOLLO with standard and parameter-wise gradient clipping on One Billion Words and WMT-14. We report the mean and standard variance over 5 runs.

	1BW	**WMT-14**
Standard	31.94±0.09	28.34±0.10
Parameter-wise	**31.75**±0.10	**28.39**±0.11

Table 7 lists the preliminary results. On both the two groups of experiments, parameter-wise gradient clipping slightly outperforms the standard one.

G EXPERIMENTS WITH SMALL TOY CNN MODELS

In this section, we provide the comparison between SdLBFGS (Wang et al., 2017) and APOLLO on CIFAR-10 dataset with a small toy CNN model[9]. The implementation of SdLBFGS is based on the public PyTorch release[10], which includes two important modifications to the original SdLBFGS algorithm: identity matrix initialization and direction normalization (Li & Liu, 2018). For each optimizer, we train the CNN model for 50 epochs with batch size equals to 64. After each epoch, the learning rate is decayed by the rate 0.95. For the start learning rate for each optimizer, we performed search in a wide range: $\eta \in \{0.2, 0.1, 0.05, 0.01, 0.005, 0.002, 0.001, 0.0005, 0.0002\}$, and select the one obtains the optimal performance. The final start learning rates for SdLBFGS and APOLLO are 0.1 and 0.001, respectively. Following Li & Liu (2018), the memory size of SdLBFGS is set to 100. For APOLLO, we linearly warmed up the learning rate from 0.01 in the first 10 updates. For other hyper-parameters of each optimizer, we choose the default value.

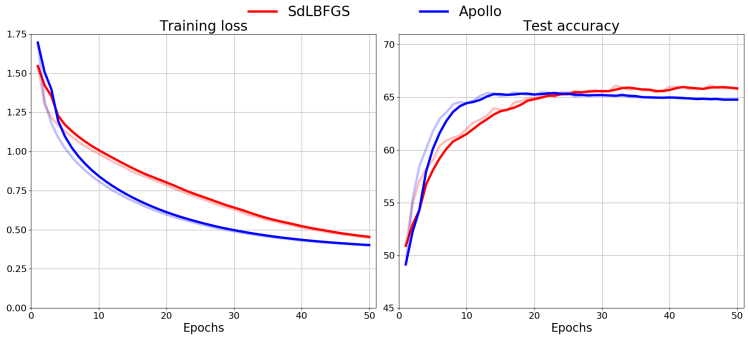

Figure 6: Training loss and test accuracy of SdLBFGS and APOLLO on CIFAR-10 with the small toy CNN model.

From Figure 6, we see that APOLLO convergences faster than SdLBFGS and obtains comparable test accuracy. Note that APOLLO is much faster (more than 10 times for one iteration) than SdLBFGS and consumes much less memory (SdLBFGS stores 100 previous update directions).

H SCALE-INVARIANT APOLLO

In (10), we rectify the absolute value of B_t with a convexity hyper-parameter σ:

$$D_t = \text{rectify}(B_t, \sigma) = \max(|B_t|, \sigma)$$

To make APOLLO scale-invariant, we modify this rectification operation by incorporating a term similar to the gradient "belief" (Zhuang et al., 2020):

$$D_t = \text{rectify}(B_t, \sigma) = \max(|B_t|, \sigma\|g_{t+1} - g_t\|_\infty) \tag{30}$$

[9] https://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial.html
[10] https://github.com/harryliew/SdLBFGS

It is not hard to prove that APOLLO with the rectification in (30) is scale-invariant. Importantly, after this modification, σ is still coupled with the stepsize η, and we can set $\sigma = 1$ in practice. Thus, we do not introduce new hyper-parameters.