We thank all reviewers for comments. We are glad to see our work commented as "promising"(R3), "effective"(R6), supported with "strong experimental results" and "intuitive justification"(R1). We address their concerns below.

Response to R1

Q1. Writing We'll rephrase remarks, e.g."Examples give hints to local behavior of optimizers in deep learning".

Q2.a Assumptions We list assumptions (1)-(3) as below:

–(1) assume g_t is drawn from a stationary distribution, hence after bias correction, $\mathbb{E}v_t = (\mathbb{E}g_t)^2 + \mathbf{Var}g_t$.

–(2) low-noise assumption, $(\mathbb{E}g_t)^2 \gg \mathbf{Var}g_t$, hence we have $\mathbb{E}g_t/\sqrt{\mathbb{E}v_t} \approx \mathbb{E}g_t/\sqrt{(\mathbb{E}g_t)^2} = sign(\mathbb{E}g_t)$.

–(3) low-bias assumption, β_1^t (β_1 to the power of t) is small. m_t as an estimator of $\mathbb{E}g_t$ has bias $\beta_1^t \mathbb{E}g_t$, as in [1]. Numerically, we need a small β (e.g 0.3) or large t. We also tried default β with large t, results similar to Fig.3(d).

Q2.b Conclusion ("Parameter" refer to coordinates of g_t) Under above assumptions, Adam is close to sign-descent, which hurts performance , similar results explained in [3] (e.g. Lemma 2&3). We will rephrase as suggested.

Q2.c Analysis setting, line 107 By "run a long time", we refer to a large t, hence β_1^t is small, and assumption (a.3) is satisfied. m_t, v_t are calculated strictly following Adam and updated with iterates, NOT post-hoc analysis of SGD.

Q3.a Notations Our notations strictly follow the convention in [1,2]. We will add missing notations to Sec2.1 and 2.3. We use (β_{1t}, β_{2t}) to denote the momentum for m_t and v_t respectively at step t, and typically set as constant (e.g. $\beta_{1t} = \beta_1, \beta_{2t} = \beta_2, \forall t \in \{1, 2, ...T\}$, where T is the total number of steps). Note that $\beta_{1t} \neq \beta_1^t$, β_1^t is β_1 to the power t. As in Algo. 1 in Appendix A, we use $\widehat{s_t}$ and $\widehat{m_t}$ to denote the bias-corrected version of s_t and m_t respectively.

Q3.b Optimization problem Strictly following the convention in [1,2], for deterministic problems, the problem to be optimized is $\min_{\theta \in \mathcal{F}} f(\theta)$; for online optimization, the problem is $\min_{\theta \in \mathcal{F}} \sum_{t=1}^{T} f_t(\theta)$, where f_t can be interpreted as "loss of the model with the chosen parameters in t-th step" [2].

Q3.c Projection step A detailed version of our method with projection step is in Appendix A. Our proof already considers projection, see Lemma 0.1 and Formula.(1) in Appendix B.

Q3.d Corollary 2.1.1 **(1)** Similar to Theorem 4.1 in [1] and corollary 1 in [2], where the term $\sum_{i=1}^{d} v_{T,i}^{1/2}$ exists, we have $\sum_{i=1}^{d} s_{T,i}^{1/2}$. Without further assumption, $\sum_{i=1}^{d} s_{T,i}^{1/2} < dG_\infty$ since $||g_t - m_t||_\infty < G_\infty$ as assumed in Theorem 2.1, and dG_∞ is constant. **(2)** The literature [1,2,5] exerts a stronger assumption that $\sum_{i=1}^{d} T^{1/2} v_{T,i}^{1/2} \ll dG_\infty T^{1/2}$. Our assumption could be similar or weaker, because $\mathbb{E}s_t = \mathrm{Var}g_t \leq \mathbb{E}g_t^2 = \mathbb{E}v_t$, then get better regret than $O(T^{1/2})$.

Response to additional comments

(a) No, see response to Q2 of R6. **(b)** Yes. It's related to "cycle" in theory, and "mode collapse" in practice. **(e)** see response to R3 below. **(f)** We refer to all three optimizers. Fig2 is illustrative; rigorously, oscillation amplitude in y-axis decreases, but gradient is independent of the distance to axis for L1 loss, hence our analysis holds for both fixed-step-size and decreasing-step-size. **(g)** We absorb ϵ into s_t in theoretical analysis, in implementation we add ϵ to match assumption $s_t > c > 0$ in Theorem 2.1 ($c \geq \epsilon > 0$). AdaBelief is robust to ϵ, as Fig.4 in Appendix.

Response to R3 We only claim AdaBelief is related to Hessian but not necessarily a good approximation, mainly because: (1) in Newton method, the update is $H^{-1}\nabla f$, using $diag(H)^{-1}$ to approximate H^{-1} may cause problems. It might be better to directly approximate $H^{-1}\nabla f$ rather than approximating H as $diag(H)$. (2) omitting the effect of EMA, $g_t - g_{t-1} \approx H\Delta\theta_t$, where $\Delta\theta_t$ is the update of parameter; in other words, $g_t - g_{t-1}$ approximates the product of H with a direction $\Delta\theta_t$, rather than approximating $diag(H)$. (3) Adam-type methods use $1/\sqrt{v_t}$, which is approximation to $H^{-1/2}$ rather than H^{-1}. We'll work on a tighter bound from Hessian perspective in future work.

Response to R6

Q1. Simplicity Our method is "simple but effective" (R6). To our knowledge, it's novel and uninvestigated before.

Q2. Comparison with Adam We address R6's concern that the success of AdaBelief stems largely from an effectively larger stepsize. We argue that this is not the case. **(1)** As in Fig.5 in Appendix, for various learning rates, AdaBelief consistently outperforms the best choice of Adam, including when Adam uses a much larger lr than AdaBelief. Validating the performance improvement of Adabelief does not solely come from larger stepsize. **(2)** when $sign(g_t) \neq sign(m_t)$ (e.g. due to noise in g_t) hence $(g_t - m_t)^2 > g_t^2$, AdaBelief can take a smaller step than Adam.

Q3. Name of our method **(1)** We use the word "belief" in a colloquial sense to refer to the amount by which the observed gradient g_t deviates from its exponential moving average m_t(viewed as approximated expected gradient). Updates in AdaBelief are "per-gradient" and element-wise, similar to Adam[1] and AdaBayes[6] where they all depend on history gradients implicitly due to the momentum and iterative update. **(2)** R6 argues intuition for AdaBelief holds for Adam, which is not true. AdaBelief resembles Adam when $(\mathbb{E}g_t)^2 \ll \mathrm{Var}g_t$. When $(\mathbb{E}g_t)^2 \gg \mathrm{Var}g_t$ Adam is close to "sign-descent" and affects accuracy, explained in Sec.2.2 of our paper and [3]; while AdaBelief overcomes this.

Q4. Prior work **(1)** The denominator in [4] is $(v_t - m_t^2)^{1/2}$, could result in numerical errors (e.g. $v_t - m_t^2 < 0$), as the authors mentioned. AdaBelief uses $[EMA((g_t - m_t)^2)]^{1/2}$ as denominator, guaranteed to be valid operation, and trains LSTM successfully without numerical issues. Compared with [4], we provide extensive theoretical and experimental validations. **(2)** [6] is completely different, "AdaBelief has nothing to do with AdaBayes" (by R6).

Q5. In Fig.3, AdaBelief uses same hyperparameters as Adam thus have similar trajectories, but reaches optima faster.

References [1] Kingma et. al, Adam: A method for stochastic optimization [2] Reddi et. al, On the convergence of Adam and beyond. [3] Lukas et. al, Dissecting adam: The sign, magnitude and variance of stochastic gradients [4] Graves et. al, Generating sequences with recurrent neural networks [5] Duchi, Adaptive subgradient methods for online learning and stochastic optimization [6] Aitchison, Bayesian filtering unifies adaptive and non-adaptive neural network optimization methods