Thanks to all reviewers for their motivating comments. We appreciate the deep insights of the reviewers and their constructive suggestions which helped a lot to improve the paper.

Overall: **[new figure and animations → R1 10) & R2 2.1, 2.2, 5.1]:** We include a figure similar to Fig.2 for Atari game Venture and animations that show reward redistributions during the games.
[analyzing Atari results → R2 2.1, 2.2, 2.3, 5.1 & R3 failure]: For Atari games, we investigated failures and successes of RUDDER. We discuss those in the new version. RUDDER failures (LSTM problems): Breakout (slow learning), DoubleDunk (explaining away), Hero (spurious redistributed rewards), Qbert (overfits to first levels), MontezumaRevenge (exploration problem). RUDDER successes (redistributed rewards): Frostbite (moving to igloo), Seaquest (getting oxygen), Venture (moving to treasure), Phoenix (shooting at boss shield).
[immediate reward experiment → R3]: In grid worlds where the agent has to move to a target, no reward was redistributed with Q-value differences as immediate reward. In probabilistic environments the reward was larger near the target. For delayed reward, positive (negative) rewards are redistributed to steps toward (away from) the target. Rewards are redistributed to changes in reward expectations. For example, losing a queen in the game chess receives negative redistributed reward.
[heavy code → R3]: A lightweight RUDDER code for the tabular case (Q-table) was already included in the submission. We extended this code to PPO and used it for additional experiments on Artificial Task (III). All code will be in a github repository.
[motivation contribution analysis → R3]: We placed the motivation of contribution analysis in the appendix (lines 1506-1514). We now also do it in the main paper. Sensitivity analysis determines how infinitesimal changes of the input lead to infinitesimal changes at the output. We are not interested in infinitesimal changes of actions or inputs but in large changes. In [1] disadvantages of sensitivity analysis are pointed out, which are more severe for reward redistribution and lead to spurious rewards.

R1: **[6) and 9) RUDDER and GAE]:** GAE fits perfectly with RUDDER. For optimal reward redistribution GAE is justified by Theorem 3, which states that the advantage function does not change. For non-optimal reward redistribution, GAE redistributes the reward further back.
[7) and 9) GAE baseline]: For Atari, both RUDDER and the baseline already use GAE based on the OpenAI PPO implementation. We now move the introduction of GAE [2] and TRPO [3] to the main paper. We performed additional experiments on Artificial Task (III) but with function approximation to compare PPO+GAE, RUDDER-PPO, and RUDDER-PPO+GAE. Again RUDDER is exponentially faster than PPO+GAE. RUDDER-PPO+GAE is slightly better than RUDDER-PPO. We report these experiments in the new version.
[10) Fig.2]: Thanks, we will improve the figure accordingly.
[11) advantage and future zero reward]: For optimal reward redistribution, learning the advantage function a^{π} is simplified to estimating the mean of immediate rewards: $a^{\pi}(s_t, a_t) = r(s_t, a_t) - \sum_a \pi(a \mid s_t)\, r(s_t, a)$. Consequently, the response function $\chi(l; s_t, a_t)$ is zero except for $l = 0$. The future expected reward can be expressed as $\kappa = \mathrm{E}\left[A_t - R_{t+1} + v(s_t) \mid s_t, a_t\right]$. For GAE$(\gamma, \lambda)$, λ determines to what extend the value function replaces the sum of future rewards in A_t. $\lambda = 0$ gives $\kappa = \gamma \mathrm{E}\left[v(s_{t+1}) \mid s_t, a_t\right]$, while $\lambda = 1$ gives $\kappa = \gamma \mathrm{E}\left[\sum_{l=0}^{\infty} \gamma^l R_{t+l+2} \mid s_t, a_t\right]$.

R2: The interpretation, visualization, and analysis of the Atari games is of general interest, therefore we answered them in "Overall" above.
[RUDDER limitations]: We will mention that RUDDER is not effective without delayed rewards since LSTM learning takes extra time and has problems with very long sequences. Furthermore, reward redistribution may introduce disturbing spurious reward signals.

R3: **[bias-variance analysis]:** Absolutely. We will try to elaborate more on the topic.
[Fig. S6]: Probably space limits will not allow this.
[Q1 off-policy]: The lessons buffer is only used for LSTM training (reward redistribution) but not for PPO learning. PPO learning is justified with any reward redistribution but if it is on-policy, learning is more efficient. Still, off-policy lessons buffers are not critical since they contain episodes with low and high reward. Low reward is less dependent on the policy. High reward episodes are biased towards the current policy since it has learned to produce them.
[Q2 TD bias/ MC variance]: The estimates are unbiased if the future expected rewards are set to zero like for direct Q-value estimation. The variance of MC is in general smaller even if reward redistribution introduces variance.
[Tab. S6]: Sorry. We corrected the wrongly placed table headings.
[C1 not defined terms], **[C2 merging para.]**, **[C3 Eq.(2) in Th. 3]:** Done, thanks!
[ensure not to fail]: It is not obvious to us how to ensure that RUDDER improves the baseline.
Thank you very much for the encouraging words. Especially as we worked hard and for a long time on this project, this is very rewarding.

[1] G. Montavon et al. Methods for interpreting and understanding DNNs. *Digit Signal Process*, 73:1–15, 2017.
[2] J. Schulman et al. High-dim. continuous control using generalized advantage estimation. *ArXiv*, 2015.
[3] J. Schulman et al. Trust region policy optimization. *ArXiv*, 2015.