AMP-IS Exploration

Things we're changing

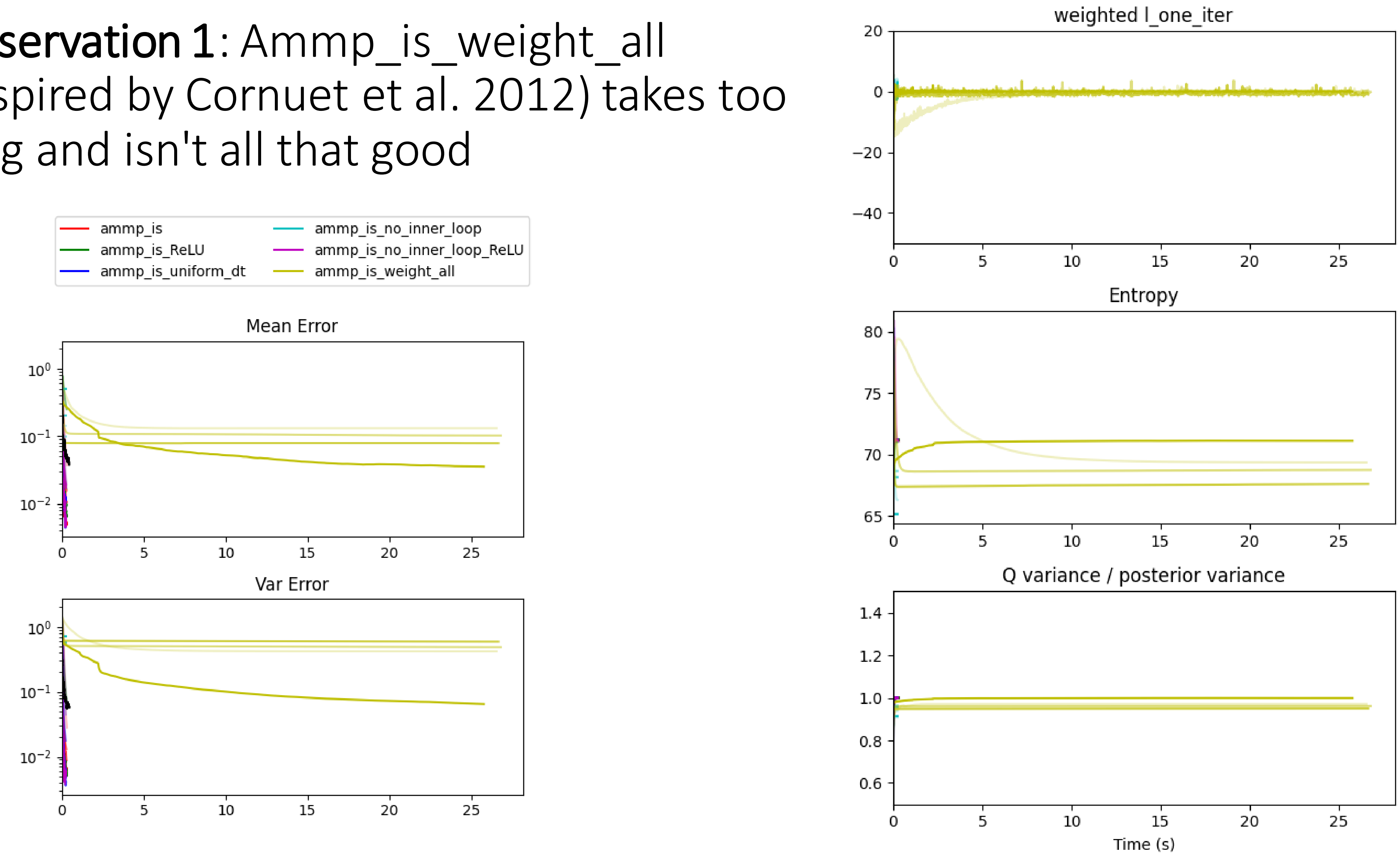
- Num_latents: 50, 500, 5000
- K: 3, 10, 30, 100
- Posterior width:
  - "WIDE": log(scale) ~ Normal(0,1)
    - Mean(scale) = 1.591, stddev(scale) = 1.854
  - "NARROW": log(scale) ~ Uniform(-6,-1)
    - Mean(scale) = 0.072, stddev(scale) = 0.087

- Learning rates (AMP-IS & natural RWS)
- Inner_loop_iters (AMP-IS)

Algorithms we're trying

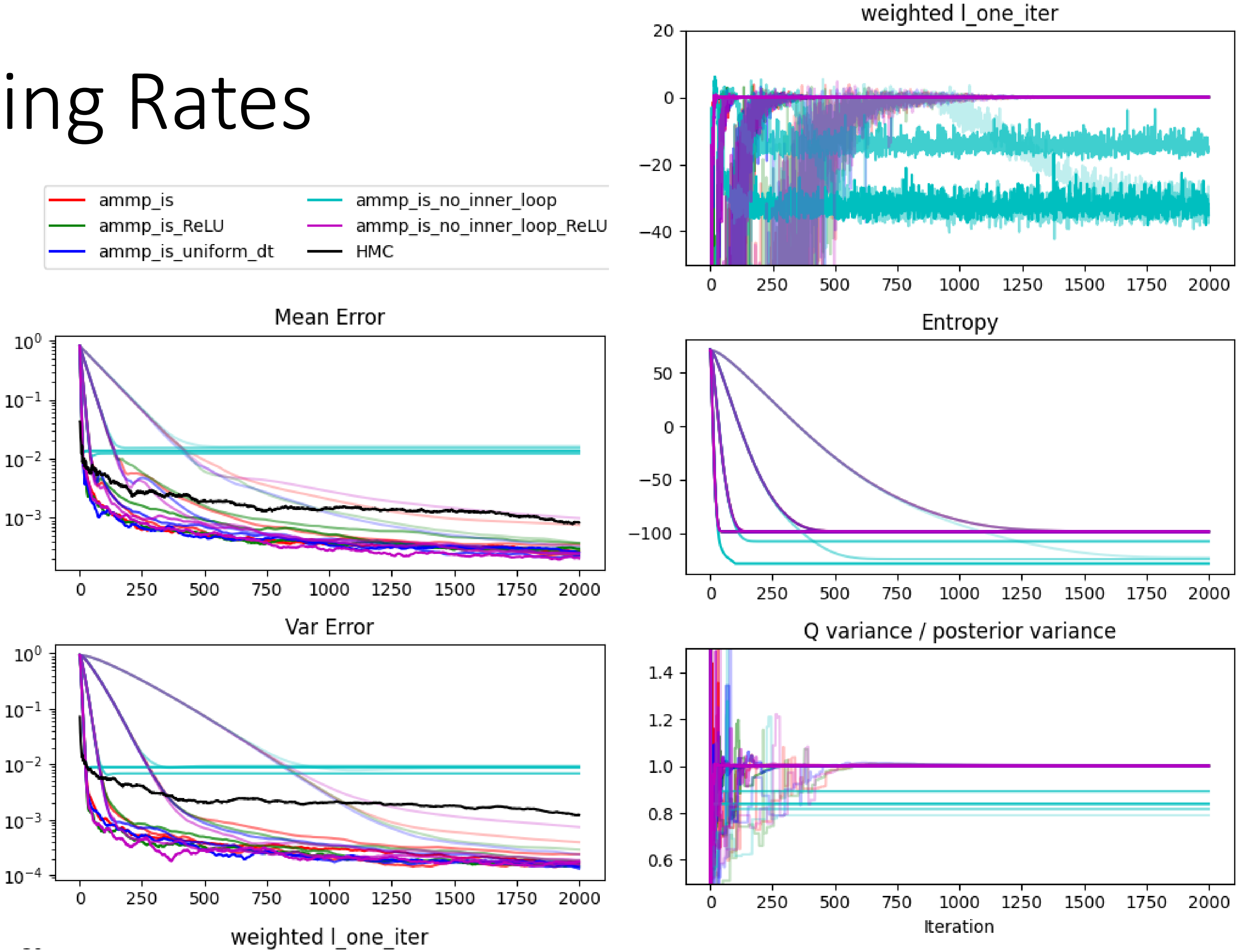
- AMP-IS:
  - Regular version with some inner loop iterations, with and without ReLU in d_t calculation
  - No_inner_loop (with and without ReLU in d_t calculation) -- this disregards w_{t-1}
  - Uniform_d_t
  - Weight_all (saving all samples into a big mixture a la Cornuet et al. 2012 (AMIS))
- Natural RWS:
  - Regular -- just moving average of MP-IS moments
  - Difference -- update with difference of sample moments and MP-IS moments
  - Standardised -- force sample (z) to have same mean and variance as Q
- HMC
- VI
- MCMC
- Langevin

Observation 1: Ammp_is_weight_all (inspired by Cornuet et al. 2012) takes too long and isn't all that good



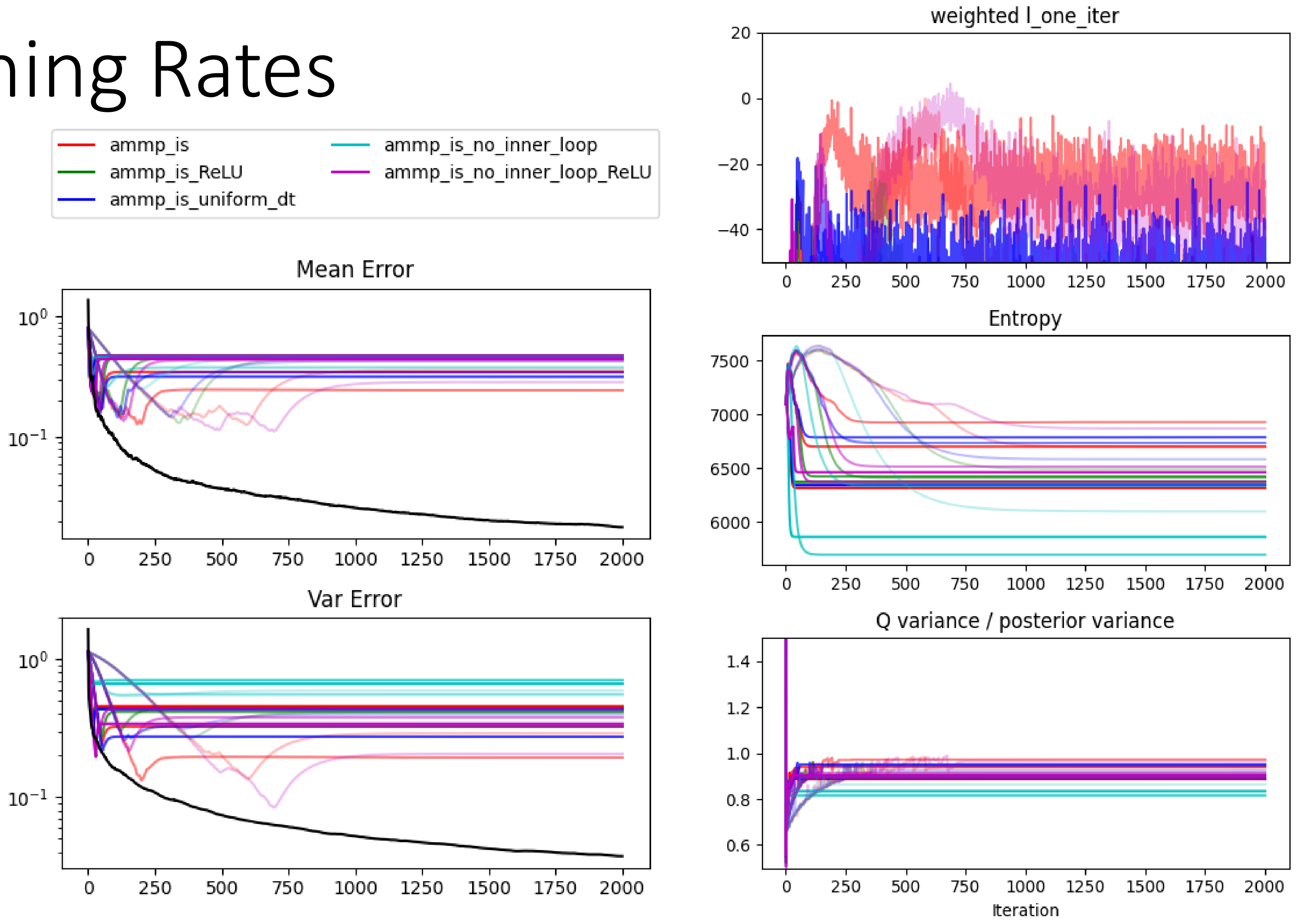
AMP-IS Learning Rates

- Higher opacity means higher learning rate.
- Lrs tried: 0.01, 0.03, 0.1, 0.3
- **Observation 2:** when AMP-IS does well, higher lrs are mostly better (or at least faster)
- **Observation 3:** no_inner_loop isn't performing very well
- (Right: N=50, K=30, NARROW posterior)



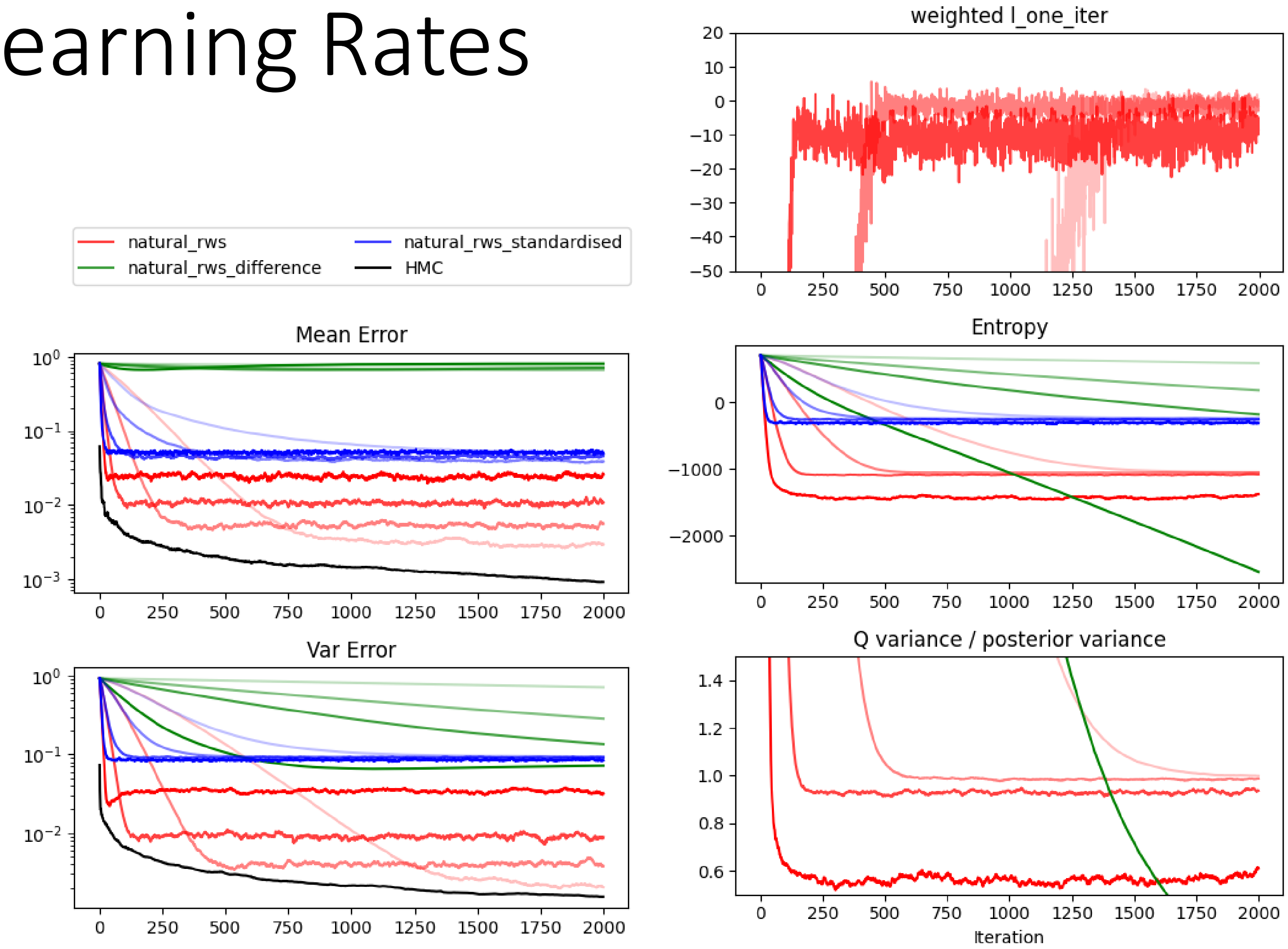
AMP-IS Learning Rates

- Higher opacity means higher learning rate.
- Lrs tried: 0.01, 0.03, 0.1, 0.3
- **Observation 4:** when AMP-IS fails, its weighted l_one_iter peaks and then decreases, occuring later on for smaller lrs
- (Right: N=5000, K=10, WIDE posterior)



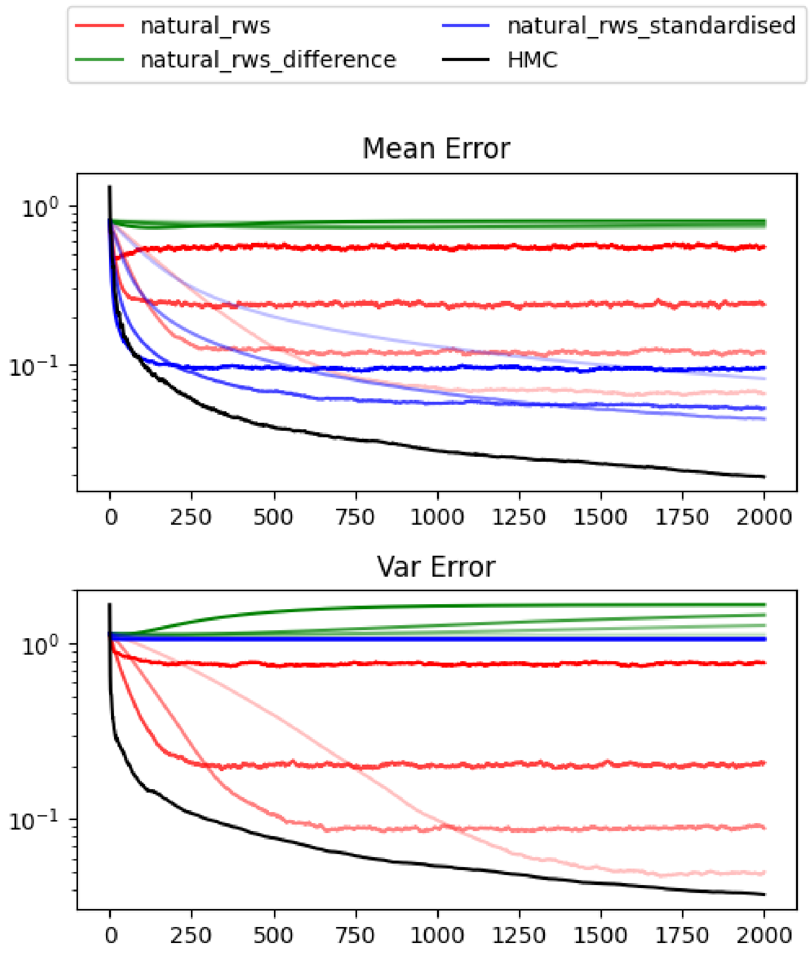
Natural RWS Learning Rates

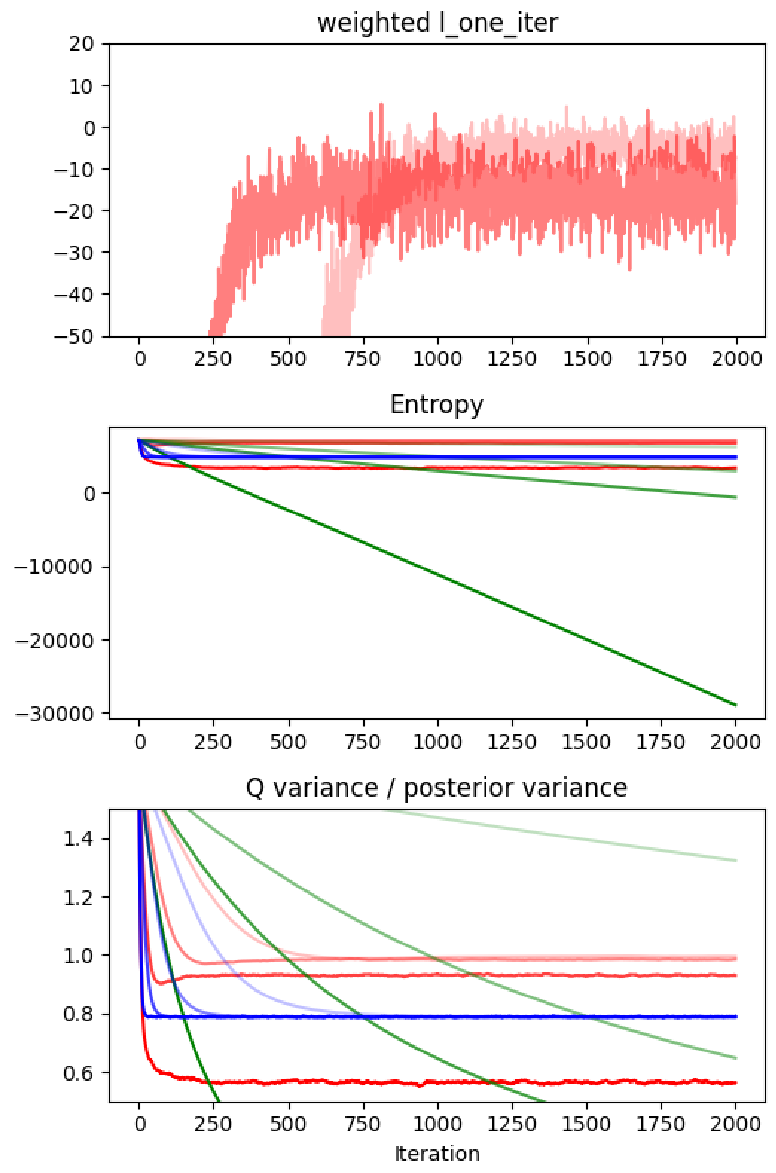
- Higher opacity means higher lr.
- Lrs tried:
  - 0.01, 0.03, 0.1, 0.3 (for regular and standardised)
  - 0.0001, 0.0005, 0.001, 0.005 (for difference, very unstable)
- **Observation 5:**
  - Regular > standardised > difference
  - (in terms of performance)
- **Observation 6:** For regular natural RWS, smaller lr means better final results but more time required to reach
- (Right: N=500, K=3, NARROW posterior)



Natural RWS Learning Rates

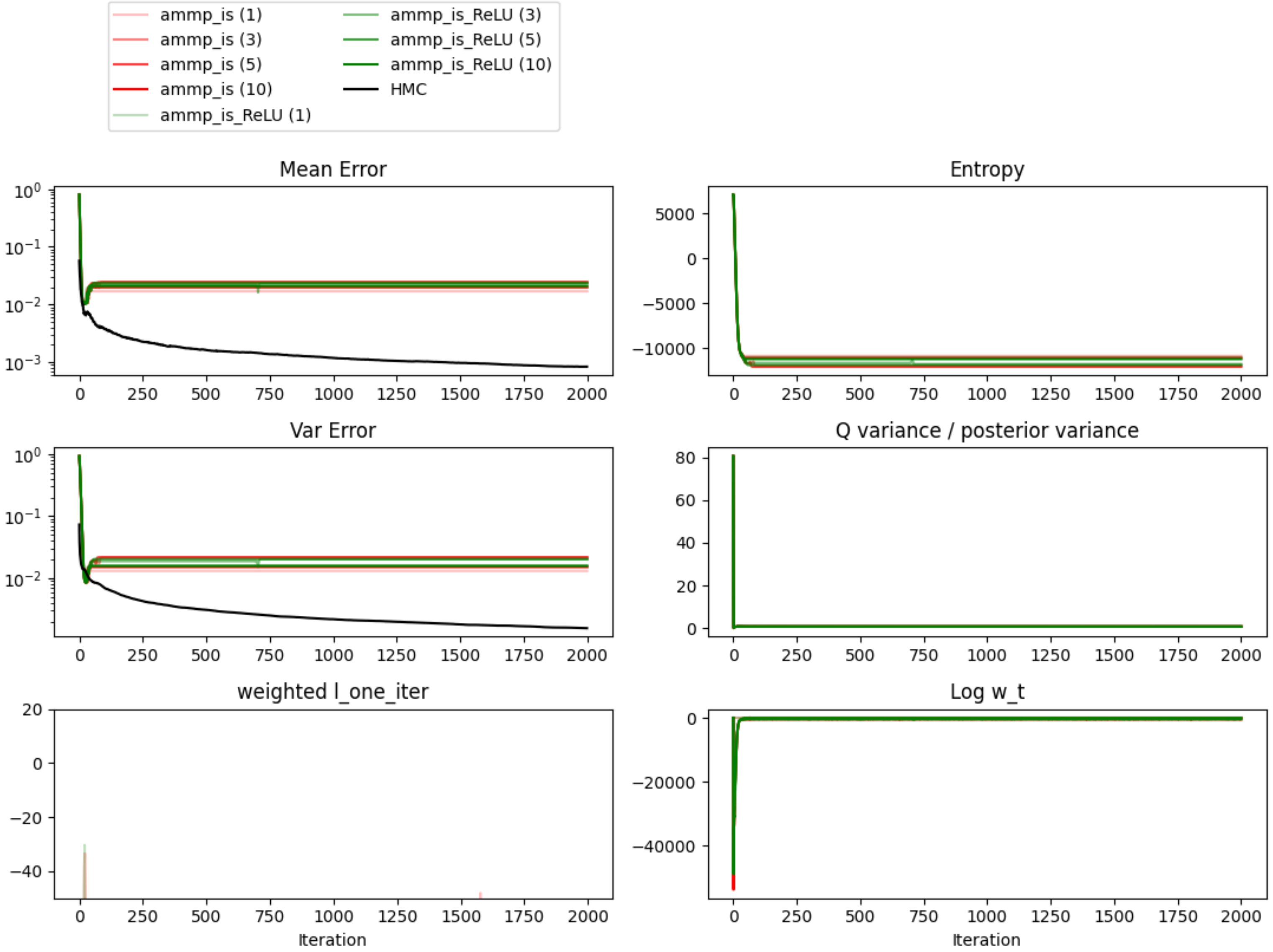
- Higher opacity means higher lr.
- Lrs tried:
  - 0.01, 0.03, 0.1, 0.3 (for regular and standardised)
  - 0.0001, 0.0005, 0.001, 0.005 (for difference, very unstable)
- **Observation 7:** sometimes standardised is better than regular RWS (for low K and usually only for the mean error)
- Showing the same learning rate behaviour as regular RWS (though perhaps to a lesser extent).
- (Right: N=5000, K=3, NARROW posterior)





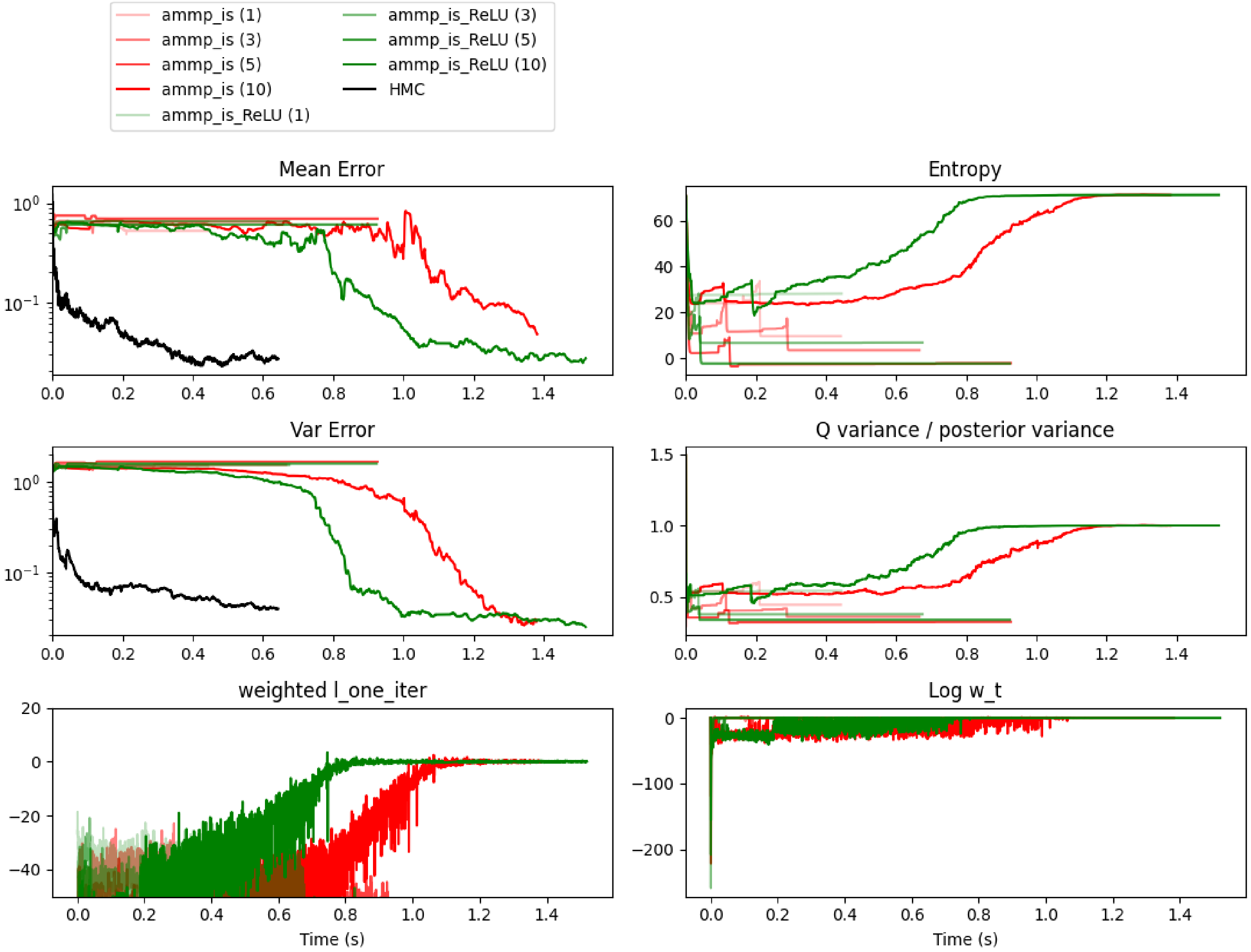
AMP-IS Inner Loop Iterations

- Higher opacity means more inner loop iters (from 1, 3, 5, 10)
- Tried AMP-IS with and without ReLU on entropy difference in d_t
- **Observation 8:** Often, the number of inner loop iterations doesn't make too much of a difference: if the posterior is very easy (low N, high K) or very hard (high N, low K).
- (Right: N=5000, K=10, NARROW posterior)



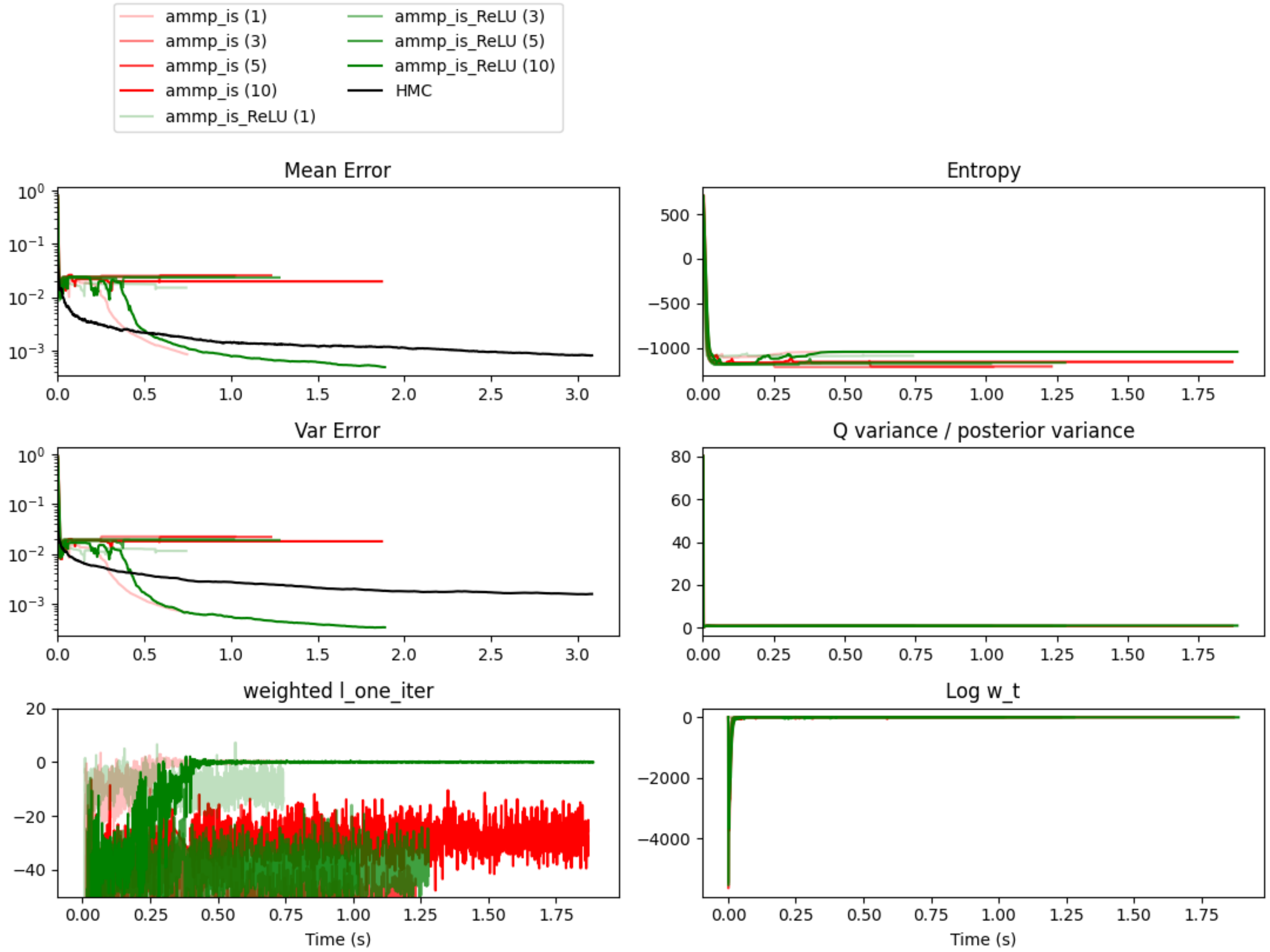
AMP-IS Inner Loop Iterations

- Higher opacity means more inner loop iters (from 1, 3, 5, 10)
- **Observation 9:** For low N, often more inner loops is helpful but expensive (takes longer than HMC)
- (Right: N=50, K=3, WIDE posterior)



AMP-IS Inner Loop Iterations

- Higher opacity means more inner loop iters (from 1, 3, 5, 10)
- **Observation 10:** For medium-large N, it seems optimal to use 1 iteration for non-ReLU and 10 for ReLU variants.
- **Observation 11:** Clearly the use of extra inner loops takes up more time for larger K and N.
- (Right: N=500, K=10, NARROW posterior)

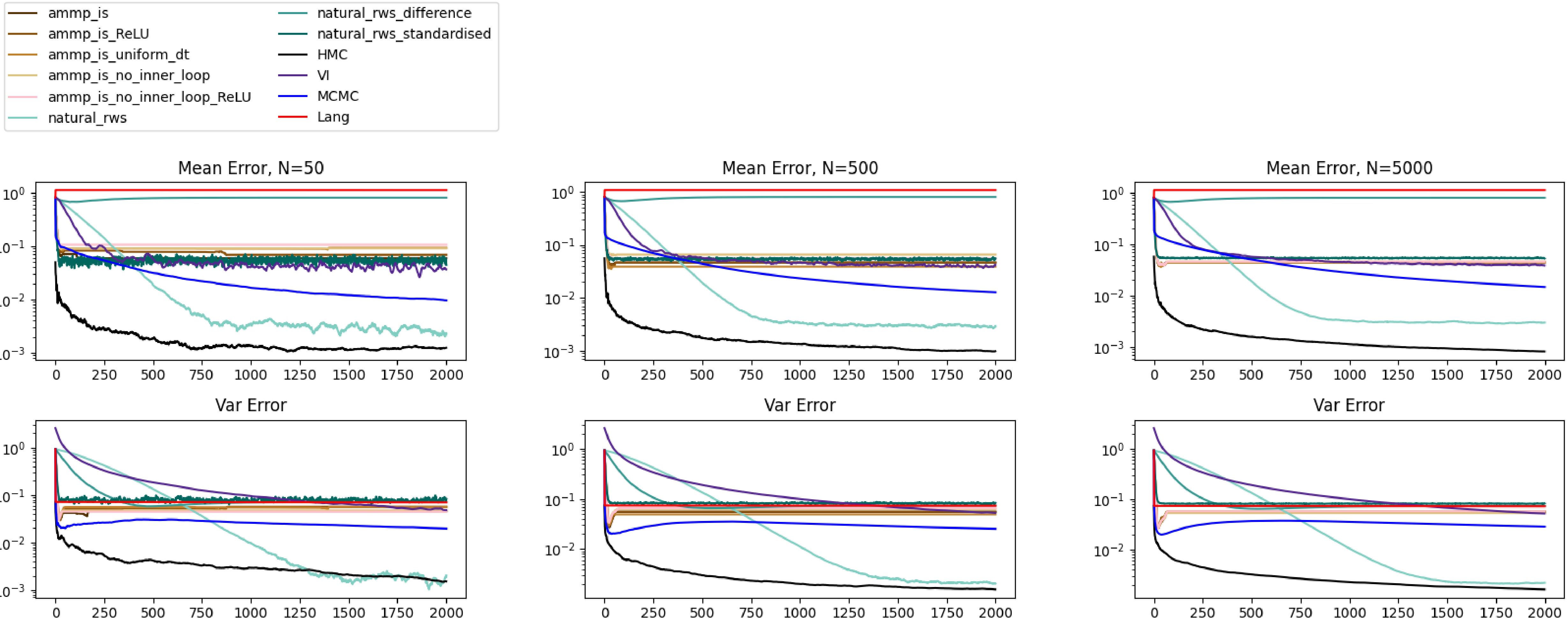


Ideal params

- AMP-IS
  - Use high learning rate (0.4)
  - Use 1 inner loop to minimise cost compared to RWS
- RWS:
  - Regular, lr = 0.01
  - Difference, lr = 0.01
  - Standardised, lr = 0.4
- VI: lr = 0.05
- MCMC and Lang tuned to (theoretically optimal) acceptance rates of 0.44 and 0.574 respectively.

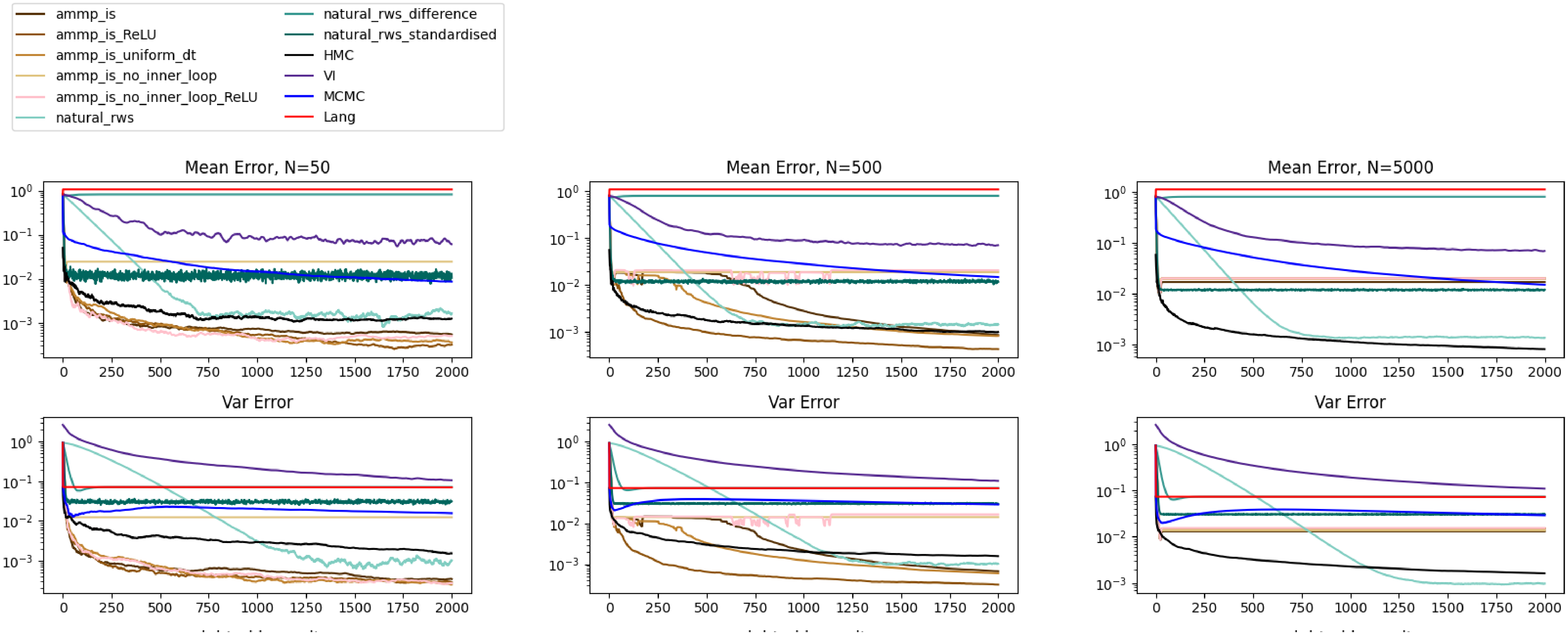
Main results, with K=10 (Below: NARROW posterior)

- For low K, natural_rws is best



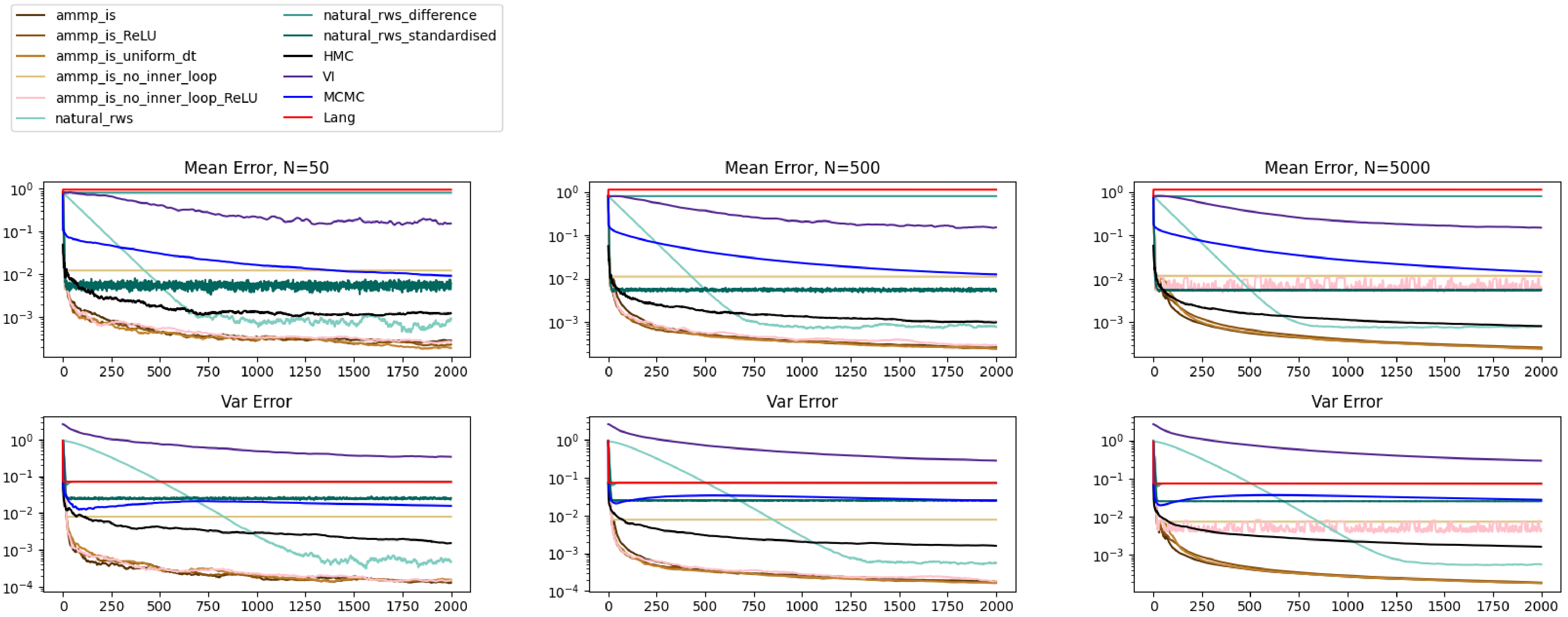
Main results, with K=10 (Below: NARROW posterior)

- For lowish K, amp-is (particularly vanilla version) is best, but large N requires natural_rws
- No_inner_loop is only good without ReLU



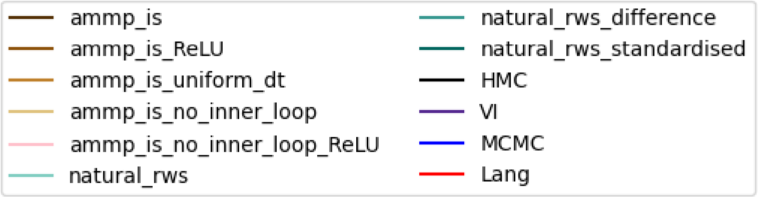
Main results, with K=30 (Below: NARROW posterior)

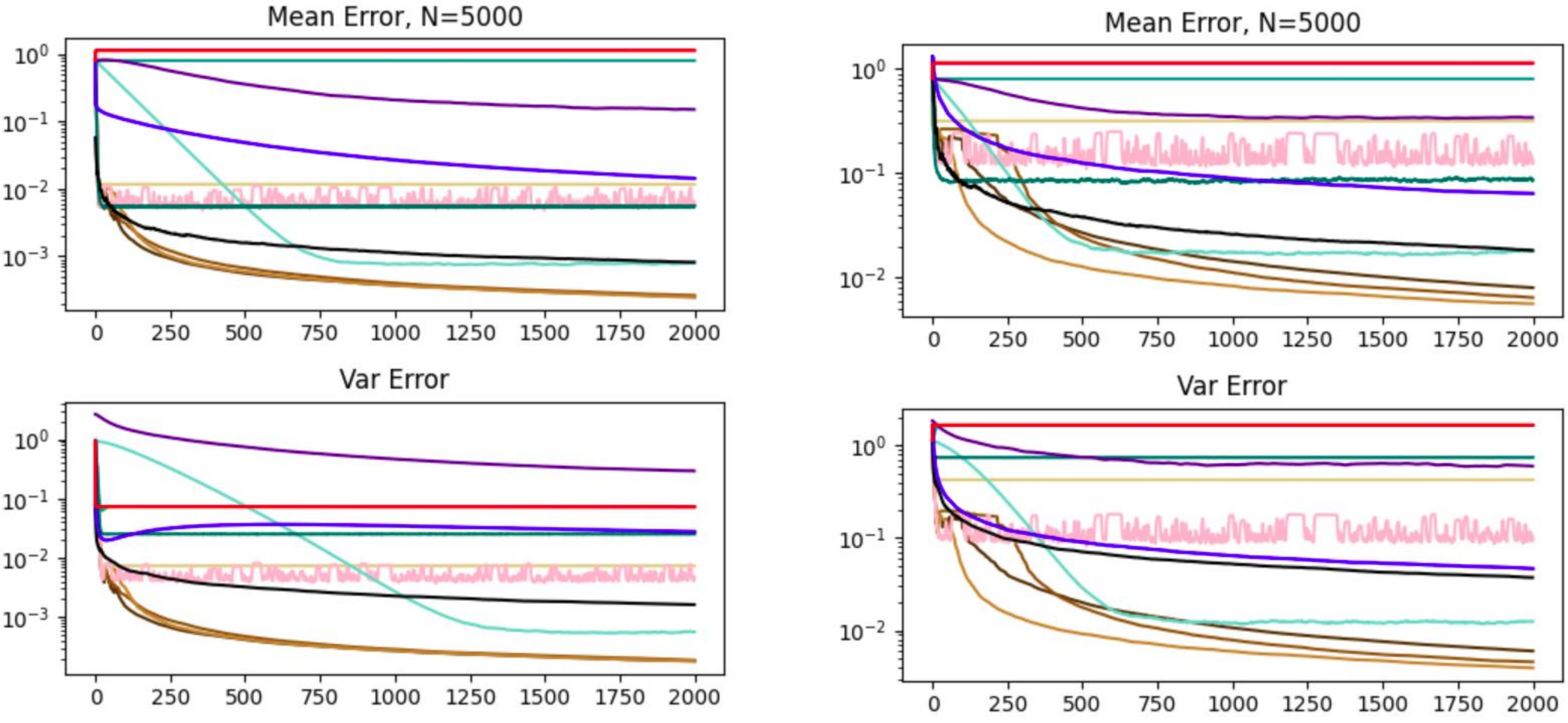
- If K is big enough (i.e. 30) then AMP-IS is best (including uniform_dt), but natural_rws still better than HMC
- No_inner_loop is only good without ReLU and for N=5000, we get weird results with ReLU



Main results, K=30 (left: NARROW, right: WIDE)

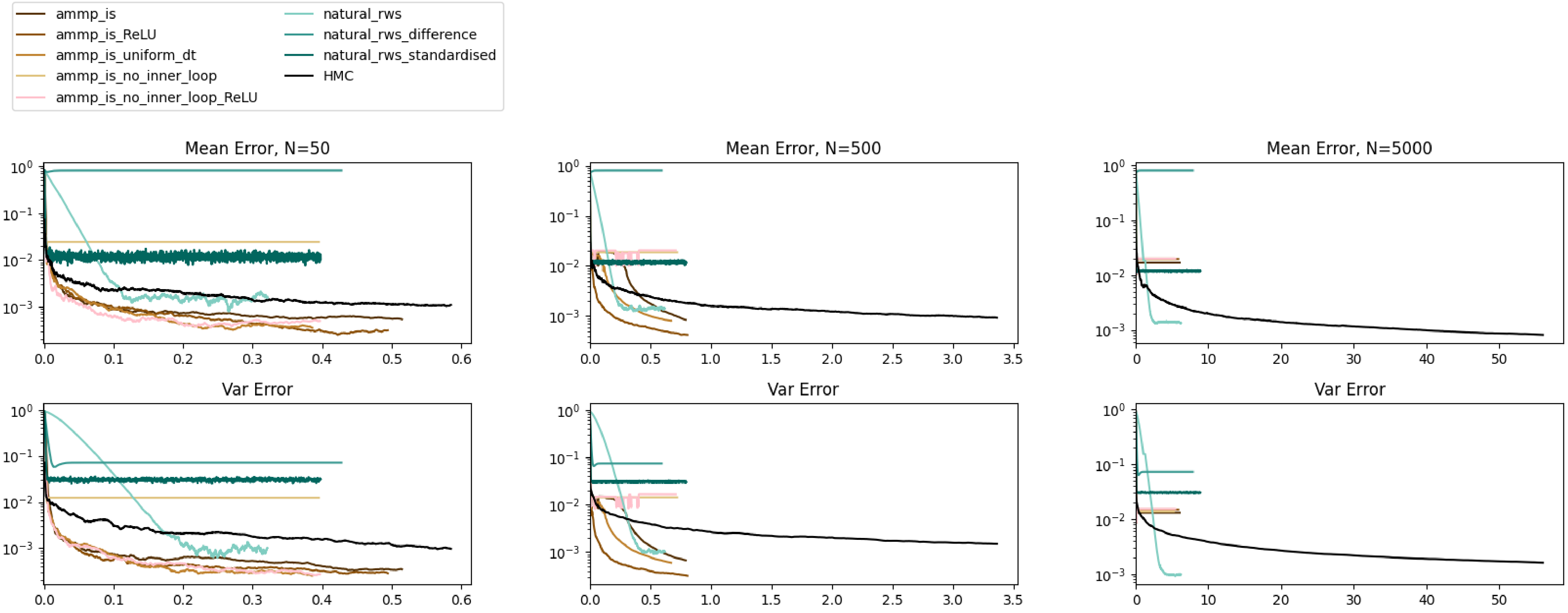
I want to say that our methods are more robust to narrow posteriors than HMC...





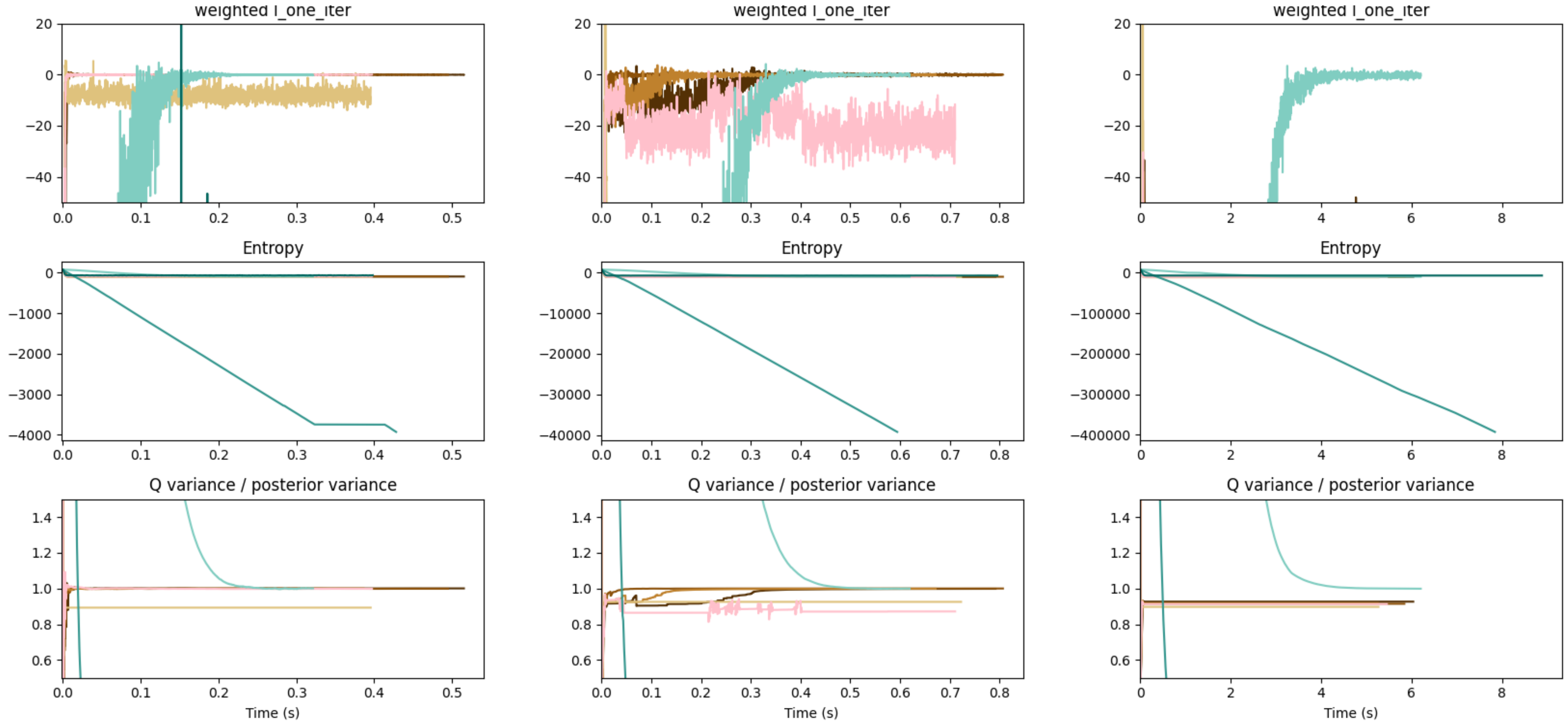
Main results: TIME (Below: K=10, NARROW posterior)

- Our methods are generally faster than HMC but this improvement is made more obvious as we increase N



Main results: TIME (Below: K=10, NARROW posterior)

- Our methods are generally faster than HMC but this improvement is made more obvious as we increase N



Appendix: AMP-IS (no_inner_loop == 1 inner_iter with d_t=log w_{t-1})

Algorithm 4 AMP-IS

$\hat{m}_0^{\mathrm{Q}}$ = Initial values. ▷ Initialize approximate posterior mean parameters.
$\hat{m}_0^{\mathrm{avg}} = 0$ ▷ Initialize importance sampled moment estimator
$\mathcal{L}_0^{\mathrm{tot}} = -10^{15}$ ▷ Initialize accumulator for denominator
for $t \in 1, 2, \ldots, T$ **do**
  $Q_t = \mathrm{fit\_approx\_post}(\hat{m}_{t-1}^{\mathrm{Q}})$
  $z \sim Q_t$
  $\hat{m}_t^{\text{one iter}}, \mathcal{L}_t^{\text{one iter}} = \mathrm{MPIW}(z, Q_t)$
  Iteratively update ML approx. post, Q_{temp} to find the right w_t
  $Q_{\mathrm{temp}} \leftarrow Q_t$
  for a couple of iterations or something **do**
    $\log w_t = -(H[Q_t] - H[Q_{\mathrm{temp}}])$ or $\log w_t = -\mathrm{ReLU}(H[Q_t] - H[Q_{\mathrm{temp}}])$
    $d_t = \log w_t - \log w_{t-1}.$
    $\mathcal{L}_t^{\mathrm{tot}} \leftarrow \mathcal{L}_{t-1}^{\mathrm{tot}} - d_t + \mathrm{softplus}\left(\mathcal{L}_t^{\text{one iter}} + d_t - \mathcal{L}_{t-1}^{\mathrm{tot}}\right)$
    $\eta_t \leftarrow \exp\left(\mathcal{L}_t^{\text{one iter}} - \mathcal{L}_t^{\mathrm{tot}}\right)$
    $\hat{m}_t^{\mathrm{avg}} \leftarrow \eta_t \hat{m}_t^{\text{one iter}} + (1 - \eta_t)\, \hat{m}_{t-1}^{\mathrm{avg}}$
    $Q_{\mathrm{temp}} = \mathrm{fit\_approx\_post}(\hat{m}_t^{\mathrm{avg}})$
  end for
  $\hat{m}_t^{\mathrm{Q}} = \lambda \hat{m}_t^{\mathrm{avg}} + (1 - \lambda)\, \hat{m}_{t-1}^{\mathrm{Q}}.$
end for

Appendix: Natural RWS

Algorithm 1 Natural RWS

$\hat{m}_0^{\mathrm{Q}}$ = Initial values. ▷ Initialize approximate posterior mean parameters.
for $t \in 1, 2, \ldots, T$ **do**
  $Q_t = \text{fit\_approx\_post}(\hat{m}_{t-1}^{\mathrm{Q}})$
  $z \sim Q_t$
  $\hat{m}_t^{\text{one iter}}, \mathcal{L}_t^{\text{one iter}} = \text{MPIW}(z, Q_t)$
  $\hat{m}_t^{\mathrm{Q}} = \lambda \hat{m}_t^{\text{one iter}} + (1 - \lambda)\, \hat{m}_{t-1}^{\mathrm{Q}}.$
end for

Appendix: Natural RWS (Difference)

Algorithm 2 Natural RWS (difference). Note that moments(z) is just the raw, uncorrect moments from the sample, z. This is nice because if the approximate posterior exactly matches the true posterior, then P=Q, and $0 = \hat{m}_t^{\text{one iter}} - \text{moments}(z)$, so there is no update, even if K is small, so there's lots of stochasticity in the sample.

$\hat{m}_0^{\text{Q}} = \text{Initial values.}$ ▷ Initialize approximate posterior mean parameters.

for $t \in 1, 2, \ldots, T$ **do**

$\quad Q_t = \text{fit\_approx\_post}(\hat{m}_{t-1}^{\text{Q}})$

$\quad z \sim Q_t$

$\quad \hat{m}_t^{\text{one iter}}, \mathcal{L}_t^{\text{one iter}} = \text{MPIW}(z, Q_t)$

$\quad \hat{m}_t^{\text{Q}} = \lambda(\hat{m}_t^{\text{one iter}} - \text{moments}(z)) + \hat{m}_{t-1}^{\text{Q}}.$

end for

Appendix: Natural RWS (Standardised)

Algorithm 3 Natural RWS (standardised). Forces the sample from z to have the same mean and variance as Q. Note that $\mathrm{E}[z]$ is the means of the sample, while $\mathrm{E}[Q]$ is the mean of the approximate posterior.

$\hat{m}_0^{\mathrm{Q}} =$ Initial values. ▷ Initialize approximate posterior mean parameters.
for $t \in 1, 2, \ldots, T$ **do**
  $Q_t = \text{fit\_approx\_post}(\hat{m}_{t-1}^{\mathrm{Q}})$
  $z \sim Q_t$
  $z \leftarrow \sqrt{\frac{\mathrm{Var}[Q]}{\mathrm{Var}[z]}}(z - \mathrm{E}[z]) + \mathrm{E}[Q]$
  $\hat{m}_t^{\text{one iter}}, \mathcal{L}_t^{\text{one iter}} = \mathrm{MPIW}(z, Q_t)$
  $\hat{m}_t^{\mathrm{Q}} = \lambda \hat{m}_t^{\text{one iter}} + (1 - \lambda)\, \hat{m}_{t-1}^{\mathrm{Q}}.$
end for