Self-Tuning Stochastic Optimization with Curvature-Aware Gradient Filtering

UNIVERSITY OF TORONTO

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

Ricky T. Q. Chen*[,1], Dami Choi*[,1], Lukas Balles*[,2], David Duvenaud[1], Philipp Hennig[2]

*Equal contribution. [1]University of Toronto, Vector Institute. [2]Max Planck Institute for Intelligent Systems, Tübingen.

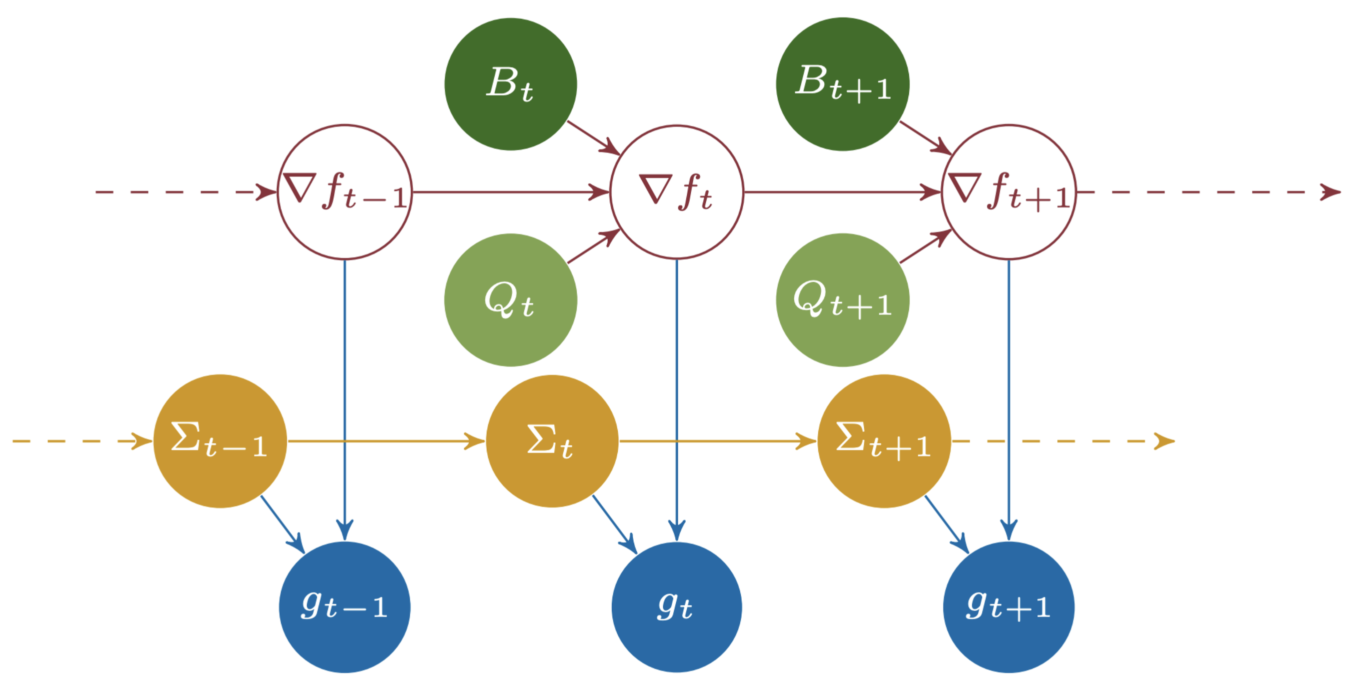
Research Question

Can we create a self-tuning optimizer by *simply tracking more quantities during optimization*, such as **curvature** and **variance**? (Instead of just minibatch gradient.)

Our approach:

- Build **gradient dynamics model**, with quantities estimated using automatic differentiation.
- Posterior inference provides low-variance gradient estimator.
- Adaptive momentum-like parameter.
- Uncertainty-aware adaptive step sizes.

Gradient Dynamics Model



Let $\delta_{t-1} = \theta_t - \theta_{t-1}$, then based on Taylor expansion:

$$\nabla f_t | \nabla f_{t-1} \sim \mathcal{N}(\nabla f_{t-1} + B_t\delta_{t-1}, Q_t) \quad (1)$$
$$g_t | \nabla f_t \sim \mathcal{N}(\nabla f_t | \Sigma_t)$$

∇f_t - expected / full batch gradient

g_t - minibatch gradient

Σ_t - minibatch gradient variance

$B_t\delta_{t-1}$ - minibatch Hessian-vector product

Q_t - minibatch Hessian-vector product variance

Inference is Online Variance Reduction

With the gradient dynamics model, posterior inference

$$p(\nabla f_t | g_1, \dots, g_t) \quad (2)$$

is equivalent to Kalman filtering.

Let $f_t | g_1, \dots, g_t \sim \mathcal{N}(m_t, P_t)$, then m_t and P_t are iteratively

$$m_t^- = m_{t-1} + B_t\delta_{t-1} \quad (3)$$
$$P_t^- = P_{t-1} + Q_{t-1} \quad (4)$$
$$K_t = P_t^-(P_t^- + \Sigma_t)^{-1} \quad (5)$$
$$m_t = (I - K_t)m_t^- + K_t g_t \quad (6)$$
$$P_t = (I - K_t)P_t^-(I - K_t)^T + K_t\Sigma_t K_t^T \quad (7)$$

Intuition regarding gradient update:

Curvature-corrected momentum-like update.

More weight on new gradient observation if its variance is relatively smaller.

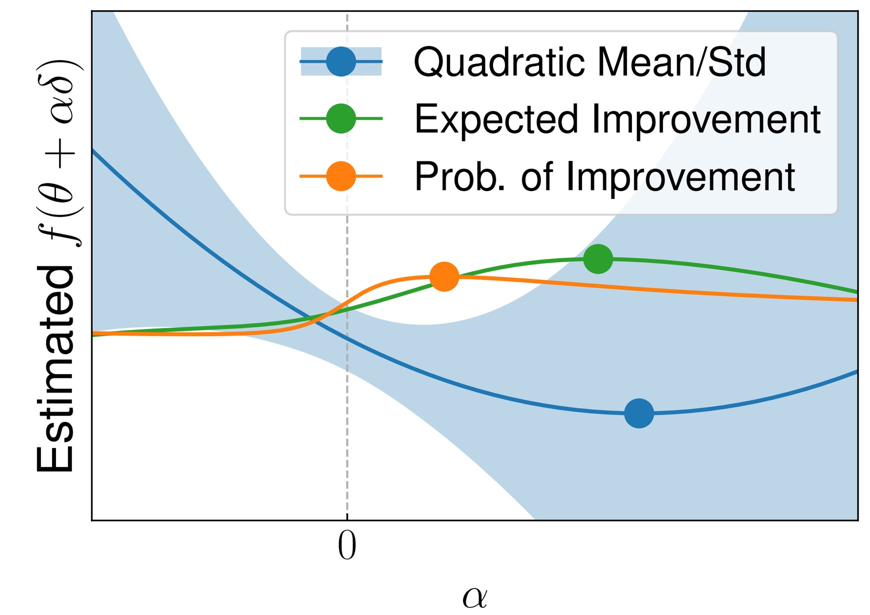
m_t is a variance-reduced gradient estimator.

Automatic Step Size Selection

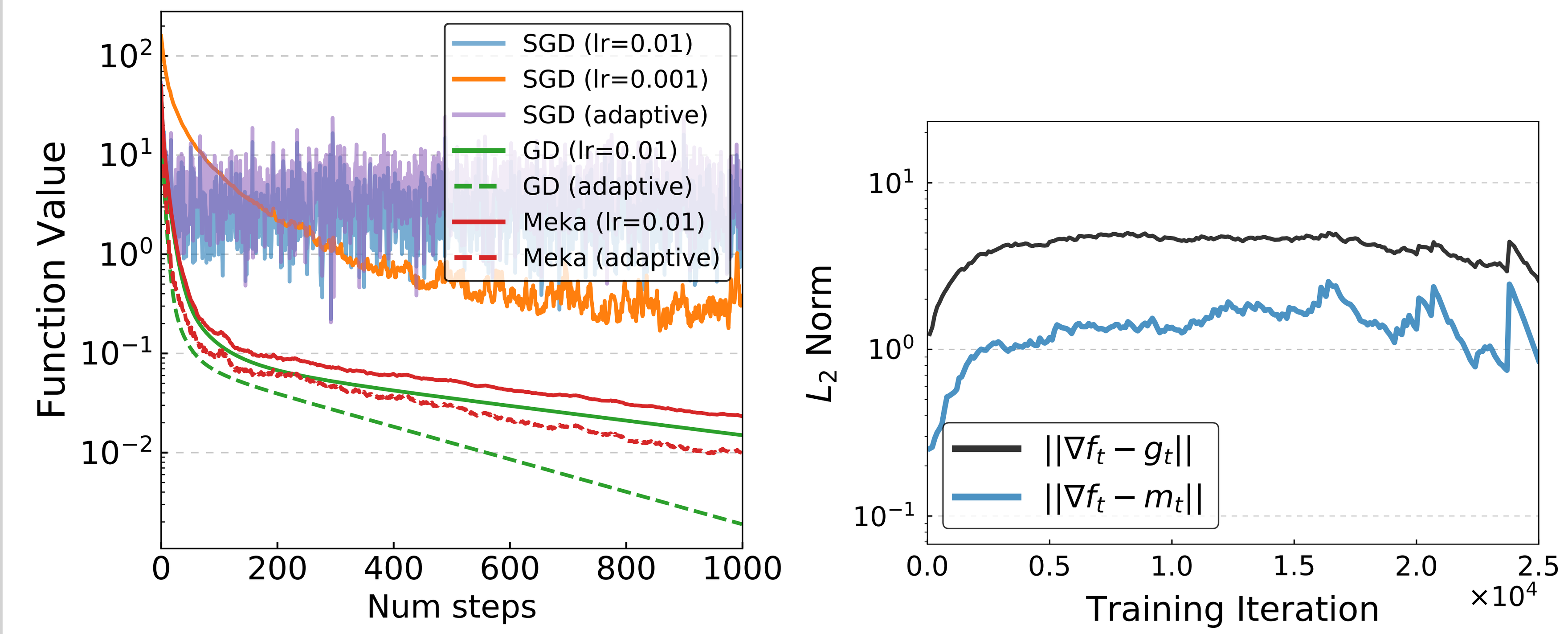
Construct 1-D Gaussian process (in the direction of δ_t):

$$\underbrace{f_{t+1} - f_t \mid y_{1:t}, g_{1:t}, \delta_{1:t}}_{\text{posterior belief of loss landscape}} \sim \mathcal{N}\Bigg(\underbrace{\alpha_t\delta_t^T m_t + \frac{\alpha_t^2}{2}\delta_t^T B_t \delta_t}_{\text{quadratic approximation}},\ \underbrace{\alpha_t^2\delta_t^T P_t\delta_t + \frac{\alpha_t^4}{4}\delta_t^T Q_t\delta_t}_{\text{posterior variance of approximation}}\Bigg)$$

Trade-off between minimization and uncertainty by choice of *acquisition function*.



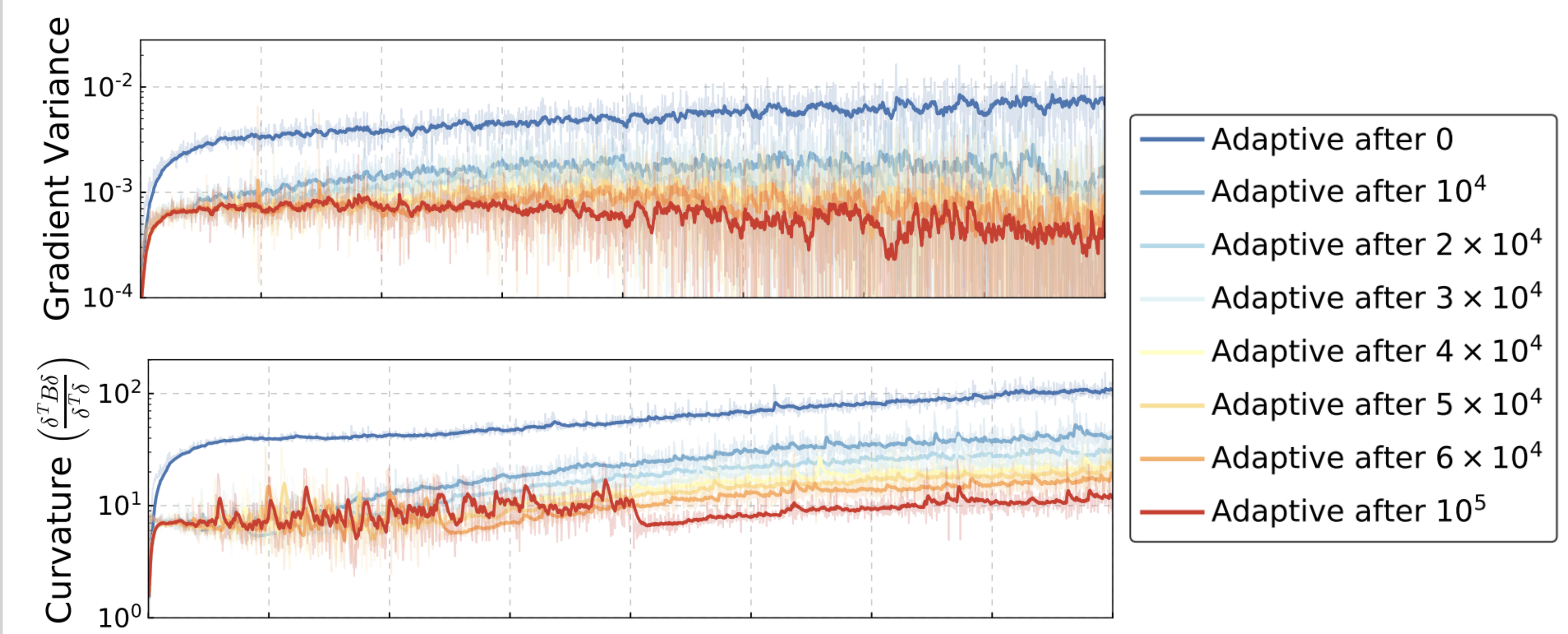
Unit Tests



(*left*) Convergence guaranteed in noisy quadratic setting.
(*right*) m_t is closer to true gradient than g_t on CIFAR-10.

Dives into High-variance High-curvature

Extra quantities can be used to diagnose training:



Adaptive step sizes allow us to dive into high-variance high-curvature regions. It works, but not ideal for deep learning.

Main issues are:

- Stochastic model parameters (B_t, Q_t, and Σ_t).
- Local 1-D Gaussian process has short-horizon bias.

Fixes (*maybe*): better dynamics models, and planning.