Stochastic Analogues to Deterministic Optimizers

ISMP 2018 – Bordeaux, France

Vivak Patel
Presented by: Mihai Anitescu

July 6, 2018

Apology

I apologize for not being here to give this talk myself. I injured my left knee and recently had surgery, which requires me to spend a little bit more time at the airport. So, I had to leave a bit earlier today than I had initially planned.

Overview & Outline

Overview: This is early work that attempts to bridge the gap between deterministic optimizers and (unconstrained) stochastic optimization problems.

Outline:

1. What do we mean by a stochastic optimization problems?
2. What is the gap between gradient-based deterministic optimizers and stochastic optimization problems?
3. What are other solutions to address this gap?
4. What is our solution to this problem?
5. How well does it work?

What do we mean by a stochastic optimization problem?

(1 of 5)

Stochastic Optimization Problem

For our purposes, an unconstrained stochastic optimization problem is one that satisfies:

Stochastic Optimization Problem

For our purposes, an unconstrained stochastic optimization problem is one that satisfies:

(1) The optimization problem is of the form

Optimization Problem

$$\min_{\theta \in \mathbb{R}^p} \mathbb{E}_{\xi} \left[f(\theta, \xi) \right] \tag{1}$$

where ξ is a random variable and $\mathbb{E}$ is the expectation operator.

Stochastic Optimization Problem

For our purposes, an unconstrained stochastic optimization problem is one that satisfies:

(1) The optimization problem is of the form

Optimization Problem

$$\min_{\theta \in \mathbb{R}^p} \mathbb{E}_{\xi} \left[f(\theta, \xi) \right] \tag{1}$$

where ξ is a random variable and $\mathbb{E}$ is the expectation operator.

(2) Importantly, the objective function (and its derivatives) is **impossible or impractical** to evaluate. We can **sample** values of ξ to generate **unbiased** samples of the objective function (and its derivatives).

Stochastic Optimization Problem: Example

Example: Empirical Risk Minimization (ERM) or Maximum Likelihood Estimation (MLE)

$$\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} f(\theta, i) \tag{2}$$

where N is so large (relative to memory or communication costs)) that it becomes impractical to evaluate all $f(\cdot, i)$ at each iterate of a classical optimization algorithm.

Stochastic Optimization Problem: Example

Example: Empirical Risk Minimization (ERM) or Maximum Likelihood Estimation (MLE)

$$\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} f(\theta, i) \tag{2}$$

where N is so large (relative to memory or communication costs)) that it becomes impractical to evaluate all $f(\cdot, i)$ at each iterate of a classical optimization algorithm.

Rewrite: Let ξ be a random variable over $\{1, 2, \ldots, N\}$ with uniform distribution. Then, the objective function is the same as

$$\min_{\theta} \mathbb{E}\left[f(\theta, \xi)\right] \tag{3}$$

Stochastic Optimization Problem: Example

Example: Empirical Risk Minimization (ERM) or Maximum Likelihood Estimation (MLE)

$$\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} f(\theta, i) \tag{2}$$

where N is so large (relative to memory or communication costs)) that it becomes impractical to evaluate all $f(\cdot, i)$ at each iterate of a classical optimization algorithm.

Rewrite: Let ξ be a random variable over $\{1, 2, \ldots, N\}$ with uniform distribution. Then, the objective function is the same as

$$\min_{\theta} \mathbb{E}\left[f(\theta, \xi)\right] \tag{3}$$

While this is a common example, our approach also applies to the case where ξ is very high-dimensional with an infinite support.

Stochastic Optimization Problem: Further Qualification

In this talk we will be interested in gradient-based optimization routines. (Note: we can also consider the case were we have Hessian information without too much additional difficulty.)

Therefore, we will require that

1. $f(\theta, \xi)$ is differentiable with respect to θ with probability one.
2. f is sufficiently regular such that $\nabla \mathbb{E}\left[f(\theta, \xi)\right] = \mathbb{E}\left[\nabla f(\theta, \xi)\right]$.

What is the gap between gradient-based deterministic optimizers and stochastic optimization problems?

(2 of 5)

Gap: Typical Unconstrained, Gradient-based Optimizer Routine

(1) Choose an initial iterate θ_0.

(2) Repeat until a stopping criteria is satisfied.

(A) Build a local model (e.g., linear, quadratic) of the objective function using an evaluation of the objective function and its derivatives.

(B) Exactly or approximately solve the local model to propose a new iterate.

(C) Accept or reject the new iterate and modify the parameters based on some performance criteria.

(3) Return the final iterate, objective function value, and performance information.

Gap: Building a Local Model

In step **(A)**, we must build a local model using the objective function and its derivatives.

Gap: Building a Local Model

In step **(A)**, we must build a local model using the objective function and its derivatives.

Problem 1: Local Model

(Obviously) The objective function and its derivatives cannot be evaluated practically. Therefore, we cannot use this information to build a local model.

Gap: Performance Criteria

In step **(C)**, we either accept or reject the proposed iterate and modify the algorithm's parameters based on some performance criteria.

Some typical performance criteria for line search subproblems include Armijo's condition or Goldstein's conditions. A typical performance criteria for trust-region subproblems is to compare the faithfulness of the trust-region problem to the original problem at the proposed iterate.

Gap: Performance Criteria

In step **(C)**, we either accept or reject the proposed iterate and modify the algorithm's parameters based on some performance criteria.

Some typical performance criteria for line search subproblems include Armijo's condition or Goldstein's conditions. A typical performance criteria for trust-region subproblems is to compare the faithfulness of the trust-region problem to the original problem at the proposed iterate.

Problem 2: Performance Criteria

Performance criteria require evaluating the objective function. However, we cannot evaluate the objective function. Therefore, we cannot evaluate a proposed iterate.

What are current solutions to these problems?
(3 of 5)

Other Solutions: Bayesian Optimization

(1) In the **Bayesian Optimization** (BO) approach, the objective function (and its derivatives) are modeled nonparametrically using a Gaussian Process.

Other Solutions: Bayesian Optimization

(1) In the **Bayesian Optimization** (BO) approach, the objective function (and its derivatives) are modeled nonparametrically using a Gaussian Process.

(2) The nonparametric model is then updated using Bayes' Rule whenever a new observation of the objective function (and its derivative) is made.

Other Solutions: Bayesian Optimization

(1) In the **Bayesian Optimization** (BO) approach, the objective function (and its derivatives) are modeled nonparametrically using a Gaussian Process.

(2) The nonparametric model is then updated using Bayes' Rule whenever a new observation of the objective function (and its derivative) is made.

(3) The estimated objective function and gradient function can then be used to solve **Problem 1** and **Problem 2** for use in a typical optimization routine.

Other Solutions: Bayesian Optimization

(1) In the **Bayesian Optimization** (BO) approach, the objective function (and its derivatives) are modeled nonparametrically using a Gaussian Process.

(2) The nonparametric model is then updated using Bayes' Rule whenever a new observation of the objective function (and its derivative) is made.

(3) The estimated objective function and gradient function can then be used to solve **Problem 1** and **Problem 2** for use in a typical optimization routine.

(4) However, we must keep all the data points in order to have convergence, which requires storing and manipulating a matrix that grows linearly with the number of observed data points. **Prohibitively expensive for even moderately sized problems!** (e.g., dimension 10)

Other Solutions: Stochastic Optimizers

(1) Stochastic Optimizers (SO) can also be used. When the random variable has a finite support, we can use techniques such as SAG(A), (A)SVRG, etc. When the random variable has infinite support, we can use SGD, AdaGrad, Kalman-based SGD, Implicit SGD etc.

Other Solutions: Stochastic Optimizers

(1) Stochastic Optimizers (SO) can also be used. When the random variable has a finite support, we can use techniques such as SAG(A), (A)SVRG, etc. When the random variable has infinite support, we can use SGD, AdaGrad, Kalman-based SGD, Implicit SGD etc.

(2) The infinite support approaches generate a proposed iterate, θ_{i+1}, (i.e., solve **Problem 1**) by using a sampled information to generate a

$$(\text{Stochastic Local Model of } \theta) + \|\theta - \theta_i\|^2_{C_i^{-1}}, \tag{4}$$

where C_i is a matrix or scalar that decays over iterations either by a schedule or adaptively, which ensures that the iterates converge.

Other Solutions: Stochastic Optimizers

(3) However, these methods do not solve **Problem 2**. They usually accept whatever iterate is generated, and hope that the schedule for C_i is appropriately chosen. This can lead to well-known, catastrophic problems such as severe stagnation or exponential divergence.

(4) Note, there has been recent efforts to use hybrid Bayesian Optimization approaches to perform local line search for stochastic optimizers to improve such issues. While this is promising, it still requires generating many samples per iteration in order to evaluate a proposed iterate.

Other Solutions: Sample Average Approximation

(1) When the random variable has infinite support, the **Sample Average Approximation** (SAA) approach replaces the original objective function with the empirical average of a sample.

Other Solutions: Sample Average Approximation

(1) When the random variable has infinite support, the **Sample Average Approximation** (SAA) approach replaces the original objective function with the empirical average of a sample.

(2) The SAA results in a potentially tractable objective function, which can solve **Problems 1 & 2**. However, this is not always guaranteed even with moderate sample sizes (e.g., Kouri and Surowiec [2016] point out that 1000 samples for a random-parameter PDE constrained problem is dauntingly expensive to solve).

Other Solutions: Sample Average Approximation

(1) When the random variable has infinite support, the **Sample Average Approximation** (SAA) approach replaces the original objective function with the empirical average of a sample.

(2) The SAA results in a potentially tractable objective function, which can solve **Problems 1 & 2**. However, this is not always guaranteed even with moderate sample sizes (e.g., Kouri and Surowiec [2016] point out that 1000 samples for a random-parameter PDE constrained problem is dauntingly expensive to solve).

(3) Moreover, we are not solving the original problem anymore, and so our solutions will still have some statistical variability from the true solution.

What is our alternative approach?

(4 of 5)

Our Solution: Objective and Gradient Modeling

(1) Just as in Bayesian Optimization, we will statistically estimate the objective and gradient function (and Hessian), and update the estimate whenever a new sample becomes available.

Our Solution: Objective and Gradient Modeling

(1) Just as in Bayesian Optimization, we will statistically estimate the objective and gradient function (and Hessian), and update the estimate whenever a new sample becomes available.

(2) However, we avoid carrying around a growing matrix; instead, we need a matrix of fixed size (e.g., dimension of the gradient plus one) in order to estimate the objective function (and its derivatives).

Our Solution: Objective and Gradient Modeling

(1) Just as in Bayesian Optimization, we will statistically estimate the objective and gradient function (and Hessian), and update the estimate whenever a new sample becomes available.

(2) However, we avoid carrying around a growing matrix; instead, we need a matrix of fixed size (e.g., dimension of the gradient plus one) in order to estimate the objective function (and its derivatives).

(3) The insight is that the objective and gradient only need to be **locally** accurate to the current iterate in order to solve **Problems 1 & 2**.

Our Solution: Objective and Gradient Modeling

(1) Just as in Bayesian Optimization, we will statistically estimate the objective and gradient function (and Hessian), and update the estimate whenever a new sample becomes available.

(2) However, we avoid carrying around a growing matrix; instead, we need a matrix of fixed size (e.g., dimension of the gradient plus one) in order to estimate the objective function (and its derivatives).

(3) The insight is that the objective and gradient only need to be **locally** accurate to the current iterate in order to solve **Problems 1 & 2**.

(4) Thus, a great way to achieve this local accuracy to use **statistical filters** rather than a Gaussian process to estimate the objective and gradient function.

Our Solution: Review of Statistical Filters

(1) Statistical filters are used to track (A) dynamical systems that are (B) observed indirectly.

Our Solution: Review of Statistical Filters

(1) Statistical filters are used to track (A) dynamical systems that are (B) observed indirectly.

(A) Suppose our dynamical system has states $\{X_i : i = 0, 1, \ldots\}$ that are related by

$$X_{i+1} = f(X_i, i) + \epsilon_i, \tag{5}$$

where f is given by the dynamics of the system, and $\epsilon_i \sim \mathcal{N}(0, \Sigma_i)$ are independent noise terms.

Our Solution: Review of Statistical Filters

(B) Suppose our observations are given by $\{Y_i : i = 1, 2, \ldots\}$ that are related to the states by

$$Y_i = LX_i + \eta_i, \tag{6}$$

where L is a matrix (usually rank deficient), and $\eta_i \sim \mathcal{N}(0, \Gamma_i)$ are independent noise terms.

Our Solution: Review of Statistical Filters

(B) Suppose our observations are given by $\{Y_i : i = 1, 2, \ldots\}$ that are related to the states by

$$Y_i = LX_i + \eta_i, \tag{6}$$

where L is a matrix (usually rank deficient), and $\eta_i \sim \mathcal{N}(0, \Gamma_i)$ are independent noise terms.

(2) If X_i is available, then (by Bayes' rule) X_{i+1} given X_i and Y_{i+1} has distribution

$$\mathcal{N}(f(X_i, i) + \Sigma_i L'(\Gamma_{i+1} + L\Sigma_i L')^{-1}\left[Y_{i+1} - Lf(X_i, i)\right], (L\Gamma_{i+1}^{-1}L' + \Sigma_i^{-1})^{-1}) \tag{7}$$

Our Solution: Review of Statistical Filters

(B) Suppose our observations are given by $\{Y_i : i = 1, 2, \ldots\}$ that are related to the states by

$$Y_i = LX_i + \eta_i, \tag{6}$$

where L is a matrix (usually rank deficient), and $\eta_i \sim \mathcal{N}(0, \Gamma_i)$ are independent noise terms.

(2) If X_i is available, then (by Bayes' rule) X_{i+1} given X_i and Y_{i+1} has distribution

$$\mathcal{N}(f(X_i, i) + \Sigma_i L'(\Gamma_{i+1} + L\Sigma_i L')^{-1}\left[Y_{i+1} - Lf(X_i, i)\right], (L\Gamma_{i+1}^{-1}L' + \Sigma_i^{-1})^{-1}) \tag{7}$$

(3) However, only an estimate of X_i is available. A statistical filter propagates this uncertainty in the estimate of X_i into the estimate of X_{i+1}, which is done by replacing Σ_i with the appropriate variance term.

Our Solution: Statistical Filtering for Optimization

The key challenge for using statistical filtering for optimization is to appropriately formulate the state dynamics.

Our Solution: Statistical Filtering for Optimization

The key challenge for using statistical filtering for optimization is to appropriately formulate the state dynamics.

State Dynamics: Our state dynamics are given by a Taylor expansion

$$\begin{bmatrix} \mathbb{E}[f(\theta,\xi)] \\ \mathbb{E}[\nabla f(\theta,\xi)] \end{bmatrix} = \begin{bmatrix} \mathbb{E}[f(\theta_i,\xi)] + \mathbb{E}[\nabla f(\theta_i,\xi)]'(\theta-\theta_i) \\ \mathbb{E}[\nabla f(\theta_i,\xi)] \end{bmatrix} + \epsilon_i, \tag{8}$$

where the (systematic) error, ϵ_i, grows like $\max\{\|\theta-\theta_i\|, \|\theta-\theta_i\|^2\}$.

Note: Our state dynamics are only locally accurate. But this is okay! We only need our function and gradient estimate to be locally accurate.

Our Solution: Statistical Filtering for Optimization

The key challenge for using statistical filtering for optimization is to appropriately formulate the state dynamics.

State Dynamics: Our state dynamics are given by a Taylor expansion

$$\begin{bmatrix} \mathbb{E}[f(\theta,\xi)] \\ \mathbb{E}[\nabla f(\theta,\xi)] \end{bmatrix} = \begin{bmatrix} \mathbb{E}[f(\theta_i,\xi)] + \mathbb{E}[\nabla f(\theta_i,\xi)]'(\theta-\theta_i) \\ \mathbb{E}[\nabla f(\theta_i,\xi)] \end{bmatrix} + \epsilon_i, \tag{8}$$

where the (systematic) error, ϵ_i, grows like $\max\{\|\theta-\theta_i\|, \|\theta-\theta_i\|^2\}$.

Note: Our state dynamics are only locally accurate. But this is okay! We only need our function and gradient estimate to be locally accurate.

So how do we use these state dynamics for estimating the objective function?

Our Solution: Statistical Filtering for Optimization

Prior: Suppose at iterate θ_i, we have estimates F_i of $\mathbb{E}[f(\theta_i, \xi)]$ and G_i of $\mathbb{E}[\nabla f(\theta_i, \xi)]$ (F_0 and G_0 can be generated by an independent copy of ξ) with an estimated variance Σ_i.

Our Solution: Statistical Filtering for Optimization

Prior: Suppose at iterate θ_i, we have estimates F_i of $\mathbb{E}[f(\theta_i, \xi)]$ and G_i of $\mathbb{E}[\nabla f(\theta_i, \xi)]$ (F_0 and G_0 can be generated by an independent copy of ξ) with an estimated variance Σ_i.

Prediction: Using our state dynamics, our predicted state is then

$$\begin{bmatrix} \tilde{F}_{i+1}(\theta) \\ \tilde{G}_{i+1}(\theta) \end{bmatrix} = \begin{bmatrix} F_i + G_i'(\theta - \theta_i) \\ G_i \end{bmatrix}, \tag{9}$$

where we have excluded the systematic error, ϵ_i, because it is unknown. Somehow, we must account for this...

Our Solution: Statistical Filtering for Optimization

Correction: To account for the systematic error, we "inflate" the variance of the prediction to reduce our "certainty" about the prediction estimate. For example, we may use (with $c > 0$)

$$\bar{\Sigma}_{i+1}(\theta) = \begin{bmatrix} 1 & (\theta - \theta_i)' \\ 0 & I \end{bmatrix} \Sigma_i \begin{bmatrix} 1 & 0 \\ (\theta - \theta_i) & I \end{bmatrix} + c \begin{bmatrix} 1 & 0 \\ 0 & I \end{bmatrix}, \tag{10}$$

or

$$\bar{\Sigma}_{i+1}(\theta) = \begin{bmatrix} 1 & (\theta - \theta_i)' \\ 0 & I \end{bmatrix} \Sigma_i \begin{bmatrix} 1 & 0 \\ (\theta - \theta_i) & I \end{bmatrix} + c\|\theta - \theta_i\| \begin{bmatrix} 1 & 0 \\ 0 & I \end{bmatrix}, \tag{11}$$

which will result in different convergence properties as we will see later.

Our Solution: Statistical Filtering for Optimization

Observation: Given an independent sample of ξ, ξ_{i+1}, our observation is $f(\theta, \xi_{i+1})$ and $\nabla f(\theta, \xi_{i+1})$.

Our Solution: Statistical Filtering for Optimization

Observation: Given an independent sample of ξ, ξ_{i+1}, our observation is $f(\theta, \xi_{i+1})$ and $\nabla f(\theta, \xi_{i+1})$.

Filter: Our filtered estimate of the objective and gradient is then given by the Kalman Filter (just as above), which can be stated as

$$\begin{bmatrix} \hat{F}_{i+1}(\theta) \\ \hat{G}_{i+1}(\theta) \end{bmatrix} = \operatorname{argmin}_Z \left\| Z - \begin{bmatrix} f(\theta, \xi_{i+1}) \\ \nabla f(\theta, \xi_{i+1}) \end{bmatrix} \right\|^2_{\Gamma(\theta)^{-1}} + \left\| Z - \begin{bmatrix} \tilde{F}_{i+1}(\theta) \\ \tilde{G}_{i+1}(\theta) \end{bmatrix} \right\|^2_{\bar{\Sigma}_{i+1}(\theta)^{-1}} \tag{12}$$

where $\Gamma(\theta)$ is the (estimated) covariance of the observation, and $\bar{\Sigma}_{i+1}(\theta)$ is the (estimated) covariance of the prediction with an inflation (to correct for error in the dynamics).

Our Solution: Problems 1

Solution to Problem 1

The filtered estimates (e.g., $\hat{F}_{i+1}(\theta_i)$, $\hat{G}_{i+1}(\theta_i)$) can be used to construct local models to generate line-search subproblems or to generate a trust-region subproblem.

For example, in a gradient descent approach, we can use $-\hat{G}_{i+1}(\theta_i)$ as our search direction, and we can use backtracking to propose an iterate $\bar{\theta}_{i+1}$.

Note, we have not accepted $\bar{\theta}_{i+1}$. In typical backtracking, we must still evaluate the performance criteria (e.g., Armijo's condition) in order to accept $\bar{\theta}_{i+1}$

Our Solution: Problems 1

Solution to Problem 1

The filtered estimates (e.g., $\hat{F}_{i+1}(\theta_i)$, $\hat{G}_{i+1}(\theta_i)$) can be used to construct local models to generate line-search subproblems or to generate a trust-region subproblem.

For example, in a gradient descent approach, we can use $-\hat{G}_{i+1}(\theta_i)$ as our search direction, and we can use backtracking to propose an iterate $\bar{\theta}_{i+1}$.

Note, we have not accepted $\bar{\theta}_{i+1}$. In typical backtracking, we must still evaluate the performance criteria (e.g., Armijo's condition) in order to accept $\bar{\theta}_{i+1}$

This brings us to **Problem 2**...

Our Solution: Problem 2

Solution to Problem 2

The filtered estimates (e.g., $\hat{F}_{i+1}(\theta)$) can then be used to evaluate performance criteria in order to reject or accept solutions to these subproblems, and to adjust parameters of the algorithm.

That is, we use $\hat{F}_{i+1}(\theta)$ to evaluate our performance criteria (e.g., Armijo's condition). If we do not accept $\bar{\theta}_{i+1}$, we can simply backtrack as we normally would to propose a new iterate and again use our performance criteria to accept or reject the iterate.

Our Solution: Problem 2

Solution to Problem 2

The filtered estimates (e.g., $\hat{F}_{i+1}(\theta)$) can then be used to evaluate performance criteria in order to reject or accept solutions to these subproblems, and to adjust parameters of the algorithm.

That is, we use $\hat{F}_{i+1}(\theta)$ to evaluate our performance criteria (e.g., Armijo's condition). If we do not accept $\bar{\theta}_{i+1}$, we can simply backtrack as we normally would to propose a new iterate and again use our performance criteria to accept or reject the iterate.

Note, we must impose an additional performance criteria (added onto Armijo's condition) that limits the growth of the variance of the filtered objective at the proposal, $\hat{F}_{i+1}(\bar{\theta}_{i+1})$, in comparison to the filtered objective at the current iterate, $\hat{F}_{i+1}(\theta_i)$. This additional constraint **is always satisfied** within a neighborhood of θ_i.

Our Solution: What if we fail?

Because the search direction (filtered gradient) is random and the filtered function is random (and highly nonlinear owing to the statistical filter), it is not always guaranteed that the filtered gradient will produce a descent direction (especially in the first few iterations).

Our Solution: What if we fail?

Because the search direction (filtered gradient) is random and the filtered function is random (and highly nonlinear owing to the statistical filter), it is not always guaranteed that the filtered gradient will produce a descent direction (especially in the first few iterations).

However, if we fail to find a step of sufficient decay (by imposing a lower bound on the step size), then we simply set $\theta_{i+1} = \theta_i$, and our estimates $\hat{F}_{i+1}(\theta_{i+1})$ and $\hat{G}_{i+1}(\theta_{i+1})$ will be more accurate estimates of the objective and the gradient evaluated at θ_i.

Therefore, we will eventually find a descent direction.

Our Solution: Convergence Theory of Filtered Estimates

Theorem: Statistical Convergence

Let $\{\theta_i\}$ be a sequence converging to θ^*, and suppose we generate filtered objective function estimates $\{F_{i+1} = \hat{F}_{i+1}(\theta_{i+1})\}$ and filtered gradient function estimates $\{G_{i+1} = \hat{G}_{i+1}(\theta_{i+1})\}$. Under certain choices of variance inflation, (F_{i+1}, G_{i+1} either

1. Converge in expectation to $(\mathbb{E}[f(\theta^*, \xi)], \mathbb{E}[\nabla f(\theta^*, \xi)])$ with bounded variance, or
2. Have a variance that converges to zero, but have a bounded bias w.r.t $(\mathbb{E}[f(\theta^*, \xi)], \mathbb{E}[\nabla f(\theta^*, \xi)])$.

Our Solution: Convergence Theory of Filtered Estimates

While the previous result is promising and supplies valuable insights into the behavior of the filtered estimates it has several drawbacks.

1. The iterates are assumed to be generated apriori. This allows us to ignore the dependence of the iterates on the filtered estimates, but we will need to include this information in order to fully understand the filters.
2. The results suggests that there is a bias-variance trade-off. While this is true when the iterates are generated apriori, it seems that it can be overcome when we consider the feedback between the iterates and the filters, and modify our performance criteria for accepting or rejecting a proposed iterate.

How well does our solution work?
(5 of 5)

Examples Summary

To demonstrate the promise of our method, we consider examples of stochastic optimization problems coming from statistics, machine learning, and optimal control.

Statistical Problem: We will fit a generalized linear mixed effect model that relates patient outcomes to some demographic variables and a random physician effect (i.e., the random variable). We compare our gradient-based method against the gold standard `lme4` result in the `R` language.

Examples Summary

Machine Learning Problem: We solve an empirical risk minimization problem for training a two-layer neural network to discern between background particles and signal particles in a physics data set. We compare our gradient-based method and SGD, where both have the same step size schedule. (Note: Our method can integrate stochastic optimizers to provide more information).

Optimal Inventory Control: We solve an inventory control problem of a distribution center serving one product under a stochastic demand. We compare our gradient-based method against a SAA approximation.

Further details can be found in the short paper on my website or in my thesis, which will be available in a few months.

Color Coding

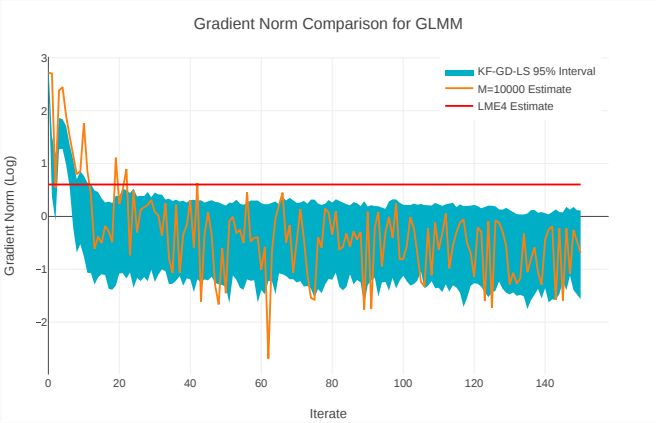
BLUE lines are the estimates generated by our algorithms during evaluation.

ORANGE lines are the high-fidelity estimates of the objective and gradient functions at the iterates generated by our algorithms, which are determined after our algorithm is finished running.

GREEN lines are the estimates (if available) generated by the competing algorithm during evaluation.

RED lines are the high-fidelity estimates of the objective and gradient functions at the iterates generated by the competing algorithms, which are determined after the algorithm is finished running.

Statistical Problem: Gradient Plot



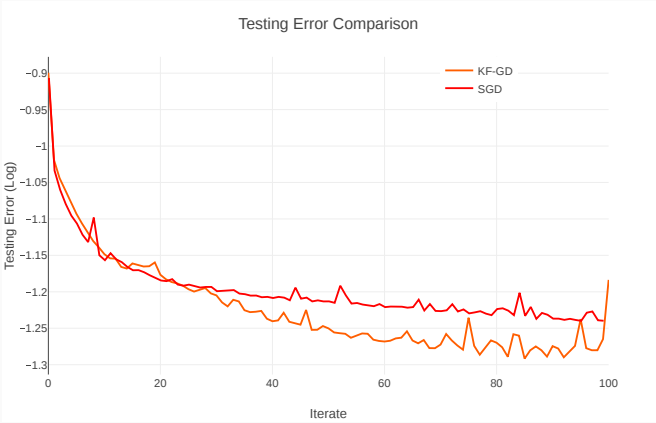
(1) The thick **blue** lines shows a 95% confidence interval that is generated by our method during evaluation.

(2) We see that our method is closer to a stationary point than the gold standard `lme4` package's estimate.

Machine Learning: Training Error Plot

(1) Except at the end, we see that our method results in a lower training error than the iterates generated by SGD.
(2) I am not entirely sure why at the end of this run, the error jumps up. This was the last run, so I kept the results, but I did not observe it in the others.

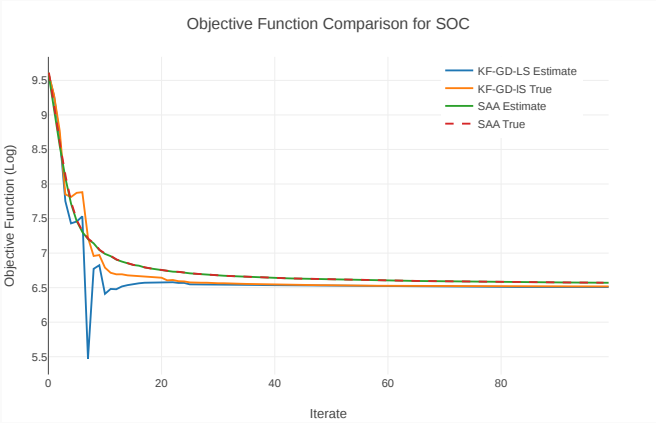
Machine Learning: Testing Error Plot



(1) Again, we outperform SGD for testing error as well, which is the metric that counts. (Except at the very end.)

(2) Note, our method takes a little bit longer than SGD because of the linear system solve that occurs at each iteration.

Optimal Control: Cost Function Plot



(1) Neither approach has solved the problem, but ours saves $\$430,000$ over SAA.
(2) Even though both methods use $50,000$ samples, our method is over $50\times$ faster as it only uses 1000 samples per iteration, whereas SAA (solved with gradient descent + back tracking) manipulates all $50,000$ at each iteration.

Summary

Summary

(1) We introduced a novel framework for overcoming the challenge of using deterministic optimizers on stochastic optimization problems (including those whose random variables have **infinite support**).

(2) The key ingredient of our framework over other approaches is that we use **statistical filters** to estimate the objective function (and its derivatives), which allows us to **efficiently** (storage and computationally) estimate the objective function and update the objective when new information is available.

(3) We showed some early theoretical results on the convergence of our framework, and we demonstrated the effectiveness of our framework on three toy problems.

Ongoing and Future Work

(1) A more complete theory to consider the dependence of the iterates on the filtered objective and gradient function estimates. This will include an extension to the case when Hessian information is used as well.

(2) We will then extend this framework to work for Quasi-Newton and Conjugated Gradient methods. These techniques introduce a more complex dependency process, even when the iterate dependence on the filtered estimates is ignored.

(3) Extensions to infinite dimensional problems, constrained problems, derivative-free paradigms, limited-memory approaches, etc.

Acknowledgements

(1) A very sincere thank you to Mihai Anitescu for presenting this talk on my behalf.

(2) NSF RTG Award 1547396 for supporting my research.

THANK YOU

www.vivakpatel.org