A class of noise-tolerant algorithms

Serge Gratton with S. Jerad and Ph.L. Toint

University of Toulouse - IRIT - ANITI
serge.gratton@toulouse-inp.fr

Bath-RAL Day, April 2023

Outline for section 1

The problem (again)

We consider the unconstrained nonlinear programming problem:

$$\text{minimize} \quad F(x)$$

for $x \in \mathbb{R}^n$ and $F : \mathbb{R}^n \to \mathbb{R}$ smooth, with Lipschitz continuous (exact) gradient $G(x) = \nabla F(x)$.

In the Big Data Era we often encounter

$$\text{minimize} \quad f(x) = \frac{1}{N}\sum_{j=1}^{N} \ell(a_j, y_j; x) \ \text{(sample mean)}$$

In ML, e.g.,

$$\ell(a_j, y_j; x) = (a_j^\top x - y_j)^2 \text{ or } \ell(a_j, y_j; x) = \log(1 + e^{-y_j(a_j^\top x - b)})$$

and sampling can be very aggressive
For now, focus on the

unconstrained case

The problem (again)

We consider algorithms for noisy problems

- that use derivatives for the step computation
- do not rely on function evaluations for the step size control

with Lipschitz continuous (exact) gradient $G(x) = \nabla F(x)$.

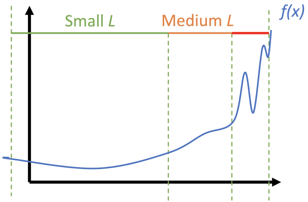
Hence, we consider now

gradient based methods for noisy problems

Stepsize adaptivity

The Lipschitz constant L in the step-size $1/L$

- is very hard to compute. Often trial and error.
- is too global to be locally efficient



Adaptively tune the step size: trust-region idea

Compute

$$\rho = \frac{\text{True decrease}}{\text{First order decrease}}$$

ρ	action
$\geq \eta_2$	increase α
$\in]\eta_1, \eta_2]$	keep α
$\leq \eta_1$	decrease α

$\Longrightarrow$

convergent algorithm ☺

complexity in $O(\epsilon^{-2})$ ☺

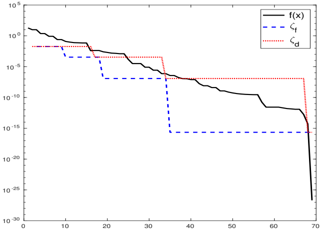
$d = -\nabla f(x)$ and $f(x)$ both needed ☹

Drama: effect of noise

In ML, severe sampling in the data results in noise in f and in ∇f.

Convergence typically provable provided

$$\text{accuracy}(f) \approx \text{accuracy}(\nabla f)^2 \qquad \text{(i.e. high sensitivity to noise in } f\text{)}$$



$\Rightarrow$ very inconvenient when inexactness results from sampling!

Can one dispense with ~~evaluating~~ using f altogether???

Objective-Function Free Optimization (OFFO)

Objective Function Free Optimization

1. Minimization algorithms when objective function and gradient are noisy have motivated many papers over the years
2. In the convergence theory, the noise in the function has to be smaller than that on the gradient. See literature on TR, and regularization algorithms
3. Stochastic methods have been developped in Machine Learning such as Adagrad (adaptative gradient algorithm) for finite sum minimization
4. Convergence theory exists in, e.g., [Défossez, Bottou, Bach, Usunier'2020], with complexity in expected square norm of the gradient: $O(N^{-\frac{1}{2}})\ln(N)$
5. See recent work, e.g., by G. Grapiglia, and F. Curtis, D. Robinson and co-authors.
6. In what follows, $g_k = g(x_k)$ is a stochastic gradient of F at x_k

Outline for section 2

The algorithm

Algorithm 2.1: The ASGRAD framework

Step 0: Initialization. Define x_0, $k = 0$, and $\gamma_{\rm low} \in (0, 1]$.

Step 1: Step computation. Evaluate g_k and set

$$s_k = \gamma_k s_k^L \quad \text{and} \quad s_{i,k}^L = -\frac{g_{i,k}}{w_{i,k}}$$

for a stepsize $\gamma_k \in [\gamma_{\rm low}, 1]$ and positive scaling factors $w_{i,k}$.
[ADAGRAD: $v_{i,k} = v_{i,k-1} + (\nabla_i f(x_k))^2$ and $w_{i,k} = \sqrt{\epsilon + v_{i,k}}$]

Step 2: New iterate. Define

$$x_{k+1} = x_k + s_k,$$

increment k by one and return to Step 1.

One may then wonder. . .

Is it possible improve the complexity bound of [Défossez et al.]???

Is is possible to derive OFFO variants that do better than ADAGRAD complexity wise ???

How about the numerical performance of such variants ???

A stochastic process

1. The source of randomness is the approximate gradient g_k
2. It generates a stochastic process

$$\{x_k, g_k, \gamma_k, s_k^L, s_k\}$$

3. $\mathbb{E}_k[\cdot]$ will stand for the conditional expectation knowing $\{g_0, \dots, g_{k-1}\}$

Assumption 1 :
We have that, for all $k \geq 0$, $\mathbb{E}_k[g_k] = G(x_k)$.
Moreover, there exists a constant $\kappa_g \geq 1$ such that $\|g_k\|_\infty \leq \kappa_g$ for all $k \geq 0$

The scaling factors $w_{i,k}$ are left unspecified:
ASGRAD is an algorithmic framework

Some assumptions on $w_{i,k}$

1. There exist a constant $\varsigma_i > 0$ and a random variable $v_{i,k}$ such that $v_{i,k} \geq \varsigma_i$ and $w_{i,k} = (v_{i,k})^\mu$ for some $\mu \in (0,1)$
2. A variance condition,

$$|\mathbb{E}_k[v_{i,k}] - v_{i,k}| \leq \kappa_v (\mathbb{E}_k[g_{i,k}^2] + g_{i,k}^2)$$

3. In addition, $g_{i,k}^2 \leq v_{i,k}$

ADGRAD is covered with $\mu = \frac{1}{2}$ and $v_{i,k} = \varsigma + \sum_{\ell=0}^{k} g_{i,\ell}^2$.

1. $v_{i,k} \geq \min_{i \in \{1,\dots,n\}} \varsigma_i \stackrel{\text{def}}{=} \varsigma_{\min}$
2. $\mathbb{E}_k[g_{i,k}^2] \leq \mathbb{E}_k[v_{i,k}]$

A decrease lemma

Generalizing a technique from [Défossez et al. 20, Ward 19], we derive a parametric bound on the decrease obtained with step s_k

Let G_j be the true gradient of F at x_j. Then, there exists $\kappa_\Delta > 0$ such that, for all $i \in \{1, \dots, n\}$,

$$\mathbb{E}_j\left[\gamma_j G_{i,j} s^L_{i,j}\right] \leq -\left(1 - \frac{\mu}{2}\right) \frac{\gamma_{\text{low}} G^2_{i,j}}{\left(\mathbb{E}_j\left[v_{i,j}\right]\right)^\mu} + 2\kappa_\Delta \mathbb{E}_j\left[\frac{g^2_{i,j}}{w^2_{i,j}}\right].$$

Remember $w_{i,k} = (v_{i,k})^\mu$.
This shows that s^L provides a descent direction on the true F as long as the square of the true gradient's norm remains large compared with the stepsizes.

Convergence of ASGRAD (I)

It is clear from

$$w_{i,k} = \left(\varsigma + \sum_{\ell=0}^{k} g_{i,\ell}^2 \right)^{\mu}$$

that $w_{i,k} \geq \varsigma^{\mu}$.

Moreover, if we define $v_{i,k} \stackrel{\text{def}}{=} \varsigma + \sum_{\ell=0}^{k} g_{i,\ell}^2$, then

$$w_{i,k} = v_{i,k}^{\mu} \text{ and } v_{i,k} \geq g_{i,k}^2$$

and

$$|\mathbb{E}_k [v_{i,k}] - v_{i,k}| = |\mathbb{E}_k [g_{i,k}^2] - g_{i,k}^2| \leq \mathbb{E}_k [g_{i,k}^2] + g_{i,k}^2.$$

Thus the proposed scaling factors verify our Assumptions with $\kappa_v = 1$.

Convergence of ASGRAD (I)

Starting from the Taylor bound

$$\mathbb{E}_j\big[F(x_{j+1})\big] \leq F(x_j) + \sum_{i=1}^{n} \mathbb{E}_j\big[\gamma_j G_{i,j} s_{i,j}^L\big] + \frac{L}{2}\mathbb{E}_j\big[\|s_j^L\|^2\big],$$

and using the descent direction Lemma, we obtain that

$$\mathbb{E}_j\big[F(x_{j+1})\big] \leq F(x_j) - (1 - \frac{\mu}{2})\gamma_{\text{low}} \frac{\|G_j\|^2}{\kappa_g^{2\mu}(k+2)^{\mu}} + \left(\frac{L}{2} + 2\kappa_\Delta\right)\mathbb{E}_j\big[\|s_j^L\|^2\big].$$

By summing up and taking full expectation,

$$\mathbb{E}\big[F(x_{k+1})\big] \leq F(x_0) - (1 - \frac{\mu}{2})\frac{\gamma_{\text{low}}}{\kappa_g^{2\mu}(k+2)^{\mu}} \sum_{j=0}^{k} \mathbb{E}\big[\|G_j\|^2\big]$$
$$+ \left(\frac{L}{2} + 2\kappa_\Delta\right)\sum_{i=1}^{n}\sum_{j=0}^{k} \mathbb{E}\big[(s_{i,j}^L)^2\big].$$

Convergence of ASGRAD (II)

Within our assumptions, consider : $w_{i,k} = \left(\varsigma + \sum_{\ell=0}^{k} g_{i,\ell}^2\right)^{\mu}$

The second order terms can be expanded as

$$\sum_{j=0}^{k} (s_{i,j}^L)^2 = \sum_{j=0}^{k} \frac{g_{i,j}^2}{(\varsigma + \sum_{j=0}^{k} g_{i,j}^2)^{2\mu}},$$

One has the technical result on non-negative sequences

Set $b_k = \sum_{j=0}^{k} a_j$.

1. If $\alpha \neq 1$, $\sum_{j=0}^{k} \frac{a_j}{(\varsigma+b_j)^{\alpha}} \leq \frac{1}{(1-\alpha)}((\varsigma + b_k)^{1-\alpha} - \varsigma^{1-\alpha})$.
2. If $\alpha = 1$, $\sum_{j=0}^{k} \frac{a_j}{\varsigma+b_j} \leq \log\left(\frac{\varsigma+b_k}{\varsigma}\right)$.

Convergence of ASGRAD (III)

For $w_{i,k} = \left(\varsigma + \sum_{\ell=0}^{k} g_{i,\ell}^2\right)^{\mu}$ we get

$$\mathbb{E}\left[\underset{j\in\{0,\ldots,k\}}{\text{average}} \|G_j\|\right] \leq \begin{cases} \mathcal{O}\left(\dfrac{1}{(k+1)^{\frac{1}{2}\mu}}\right) & (\mu \in (0,\tfrac{1}{2})), \\ \mathcal{O}\left(\dfrac{\sqrt{\log(k+1)}}{(k+1)^{\frac{1}{4}}}\right) & (\mu = \tfrac{1}{2}), \\ \mathcal{O}\left(\dfrac{1}{(k+1)^{\frac{1}{2}(1-\mu)}}\right) & (\mu \in (\tfrac{1}{2},1)). \end{cases}$$

1. This proves the convergence of the algorithm for $\mu \in (0,1)$
2. Recover complexity obtained for the standard Adagrad algorithm

Is this optimal ???

Convergence of ASGRAD (III)

For $w_{i,k} = \left(\varsigma + \sum_{\ell=0}^{k} g_{i,\ell}^2\right)^{\mu}$, suppose the variance condition

$$\mathrm{Var}_k\left[g_{i,k}\right] = \mathbb{E}_k\left[g_{i,k}^2 - G_{i,k}^2\right] \leq \kappa_{\text{var}} G_{i,k}^2$$

holds. Then there exists j_θ, implicitly defined, such that

$$\mathbb{E}\left[\underset{j\in\{j_\theta+1,\ldots,k\}}{\text{average}} \|G_j\|\right] = \mathcal{O}\left(\frac{1}{(k+1)^{\frac{1}{2}(1-\mu)}}\right)$$

.

- The index j_θ depends on the particular realization considered
- Better bound than the existing ones for Adagrad (no log term)
- For small μ this result is close to bounds obtained by standard algorithms that do evaluate F (TR, LS)

Divergent weights

Let $\nu \in (0,1)$ and $\mu \in [\nu, \max(1, \frac{4}{3}\nu))$, $\rho_{i,k}$ and $\xi_{i,k}$ be uniformy bounded random variables. Take $\rho_{i,k}(k+1)^{\nu} \le w_{i,k} \le \xi_{i,k}(k+1)^{\mu}$, with $\varsigma \le \rho_{i,k}$ and $\xi_{i,k} \le \kappa_\xi$ for some constants $0 < \varsigma \le \kappa_\xi$.
Then, for any $\theta \in (0, \frac{\gamma_{\text{low}}}{\kappa_\xi}))$,

$$\mathbb{E}\left[\underset{j \in \{j_\theta+1,\dots,k\}}{\text{average}} \|G_j\|\right] = \mathcal{O}\left(\frac{1}{(k+1)^{\frac{1}{2}(1-\mu)}}\right).$$

This hold with

$$j_\theta \overset{\text{def}}{=} \left\lfloor \left(\frac{L\kappa_\xi^3(1+\kappa_{\text{var}})}{2^{1-\mu}\varsigma^4(\gamma_{\text{low}} - \theta\kappa_\xi)}\right)^{\frac{1}{4\nu-3\mu}} \right\rfloor + 1.$$

This results are identical to the Adagrad family, with now an explicit formula for j_θ.

Numerical experiments: weights

Take fix learning rates $\gamma = 5e-5$ or $5e-4$.
The following strategies satisfy our assumptions:

1. the μ–strategy:

$$w_{i,k} = \left(\varsigma + \sum_{\ell=0}^{k} g_{i,\ell}^2\right)^{\mu},$$

2. the *maxgi* strategy:

$$\xi_k = \max(\varsigma, \xi_{k-1}, |g_k|) \text{ and } w_{i,k} = \xi_k (k+1)^{\nu},$$

3. the *avrgi* strategy:

$$w_{i,k} = \max(\varsigma, \frac{1}{k+1} \sum_{j=0}^{k} |g_{i,k}|)(k+1)^{\nu}.$$

Remember $\rho_{i,k}(k+1)^{\nu} \leq w_{i,k} \leq \xi_{i,k}(k+1)^{\mu}$ for the *maxgi* and *avrgi* strategies.

We use $\mu \in \{0.1, 0.5, 0.9\}$, $\nu = 0.1$ and $\varsigma = 0.01$.

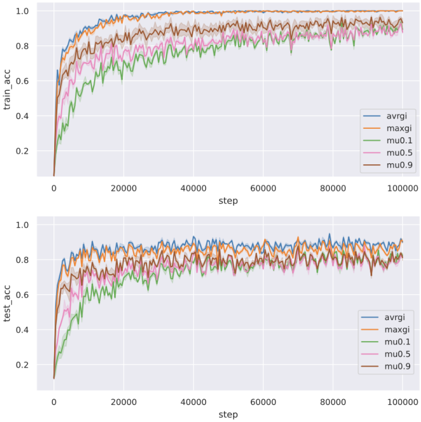
Numerical experiments: data bases, architectures, software

- Two network architectures: `cifar-nv` convolutional network of [Gitman, Ginsburg'17] and a small `resnet18` model [He et al.'15]
- Four standard datasets of 32×32 images: CIFAR10 and CIFAR100[(1)], SVHN[(2)] and FMNIST [Xiao et al.'17]
- We used haiku [Henn et al.'20] and optax [Hess et al.'20], two JAX [Brad et al.'18] based libraries
- A workstation with four GTX 1080TI

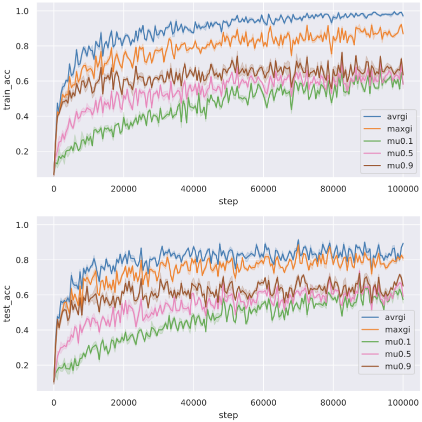
(1) https://www.cs.toronto.edu/~kriz/cifar.html

(2) http://ufldl.stanford.edu/housenumbers

CIFAR10 - `cifar-nv`

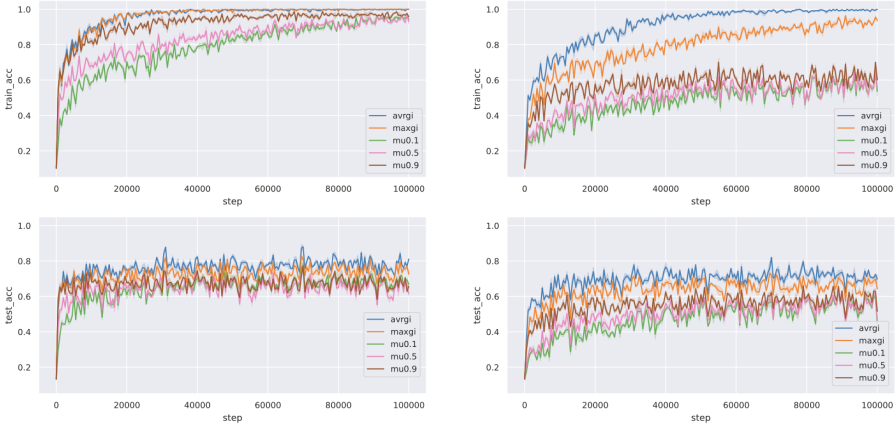


$\gamma = 5.10^{-4}$

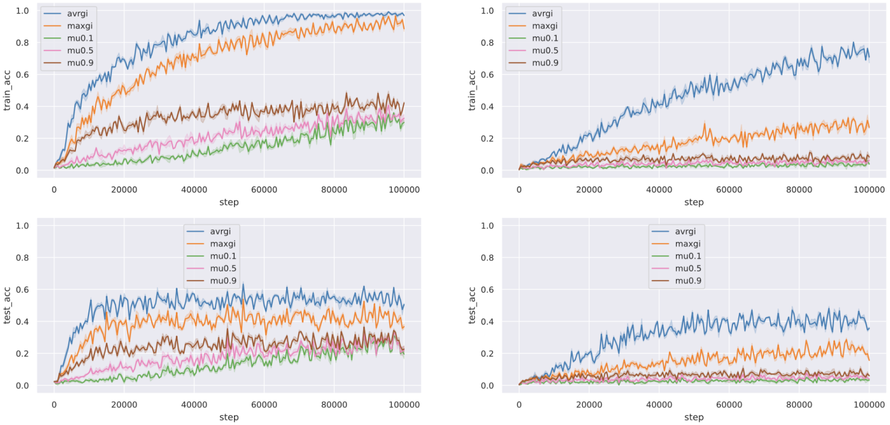


$\gamma = 5.10^{-5}$

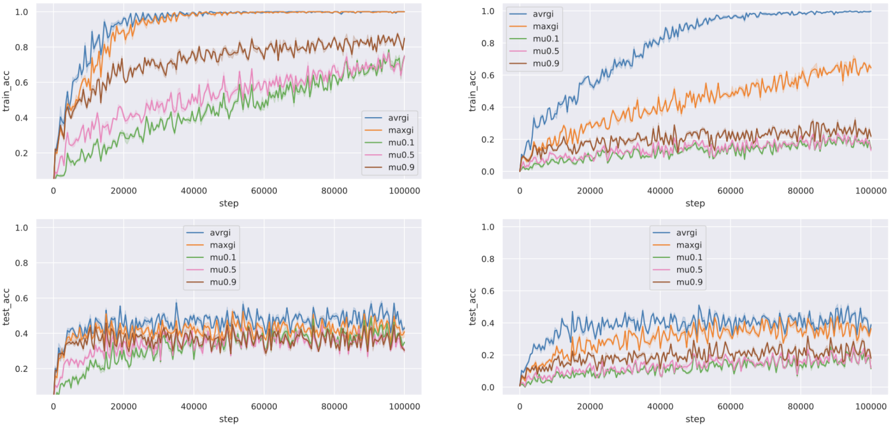
CIFAR10 - resnet18



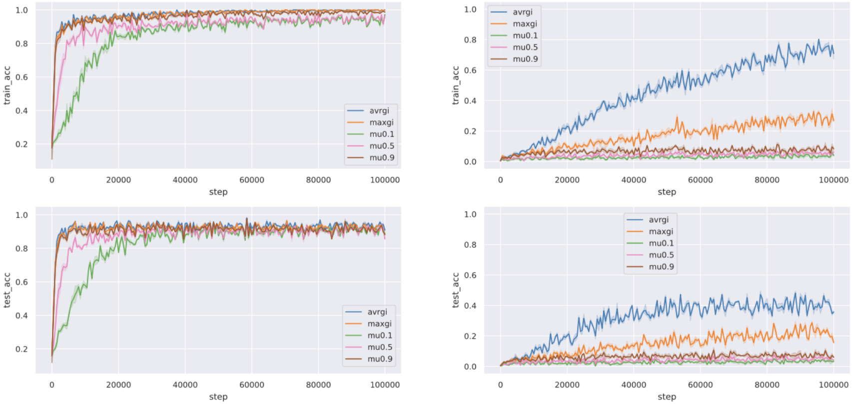
CIFAR100 - cifar-nv



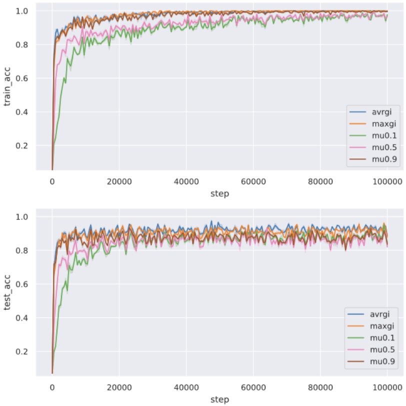
CIFAR100 - resnet18



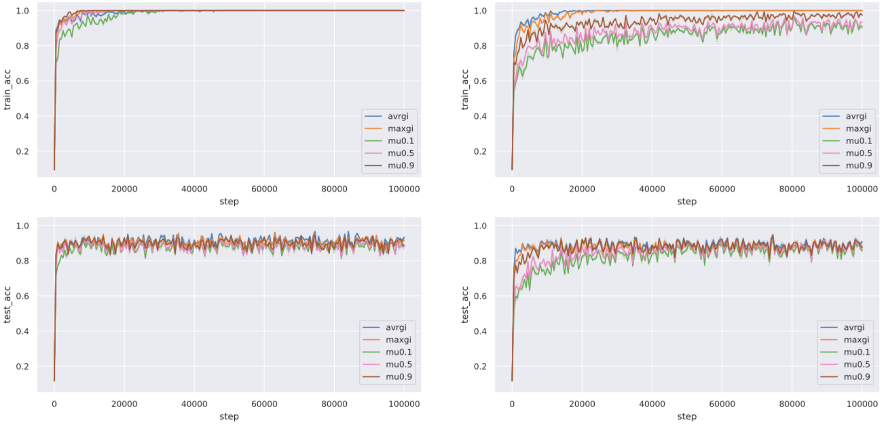
SVSH - cifar-nv



SVSH - resnet18



FMNIST - resnet18



Outline for section 3

Second order models

We allow the use of second-order information by defining a quadratic model

$$g_k^T s + \tfrac{1}{2} s^T B_k s$$

where B_k can of course be chosen as the true second-derivative matrix of f at x_k or an approximation. Choosing $B_k = 0$ results in a purely first-order algorithm.
For given $\varsigma \in (0,1]$, $\vartheta \in (0,1]$ and $\mu \in (0,1)$, define, for all $i \in \{1,\ldots,n\}$ and for all $k \geq 0$,

$$w_{i,k} \in \left[\sqrt{\vartheta}\, v_{i,k}, v_{i,k}\right] \quad \text{where} \quad v_{i,k} \stackrel{\text{def}}{=} \left(\varsigma + \sum_{\ell=0}^{k} g_{i,\ell}^2\right)^{\mu}. \tag{3.1}$$

Clearly, the Adagrad scaling factors are recovered by $\mu = \frac{1}{2}$, and $B_k = 0$ is the (deterministic) Adagrad method.

The algorithm

Algorithm 3.1: ASTR1

Step 0: Initialization. x_0, $\kappa_B \geq 1$ and $\tau \in (0,1]$ given. Let $k = 0$.

Step 1: Define the TR. Compute $g_k = g(x_k)$ and define $\Delta_{i,k} = \frac{|g_{i,k}|}{w_{i,k}}$

Step 2: Hessian approximation. Select a symmetric Hessian approximation B_k such that $\|B_k\| \leq \kappa_B$.

Step 3: GCP. Compute a step s_k such that $|s_{i,k}| \leq \Delta_{i,k}$, and
$g_k^T s_k + \frac{1}{2} s_k^T B_k s_k \leq \tau \left(g_k^T s_k^Q + \frac{1}{2} (s_k^Q)^T B_k s_k^Q \right)$, where
$s_{i,k}^L = -\mathrm{sgn}(g_{i,k}) \Delta_{i,k}$, $s_k^Q = \gamma_k s_k^L$, with

$$\gamma_k = \begin{cases} \min\left[1, \dfrac{|g_k^T s_k^L|}{(s_k^L)^T B_k s_k^L}\right] & \text{if } (s_k^L)^T B_k s_k^L > 0, \\ 1 & \text{otherwise.} \end{cases}$$

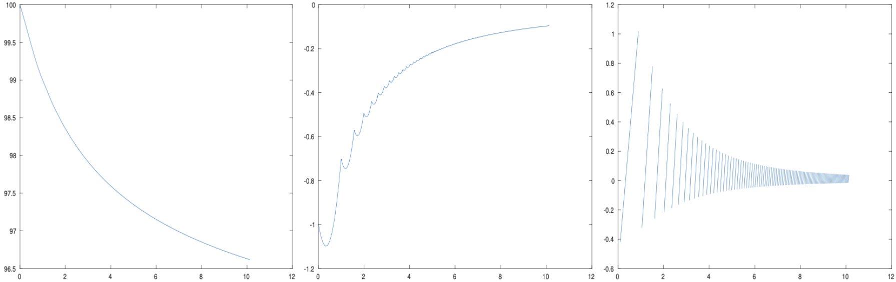
Step 4: New iterate. $x_{k+1} = x_k + s_k$

The algorithm

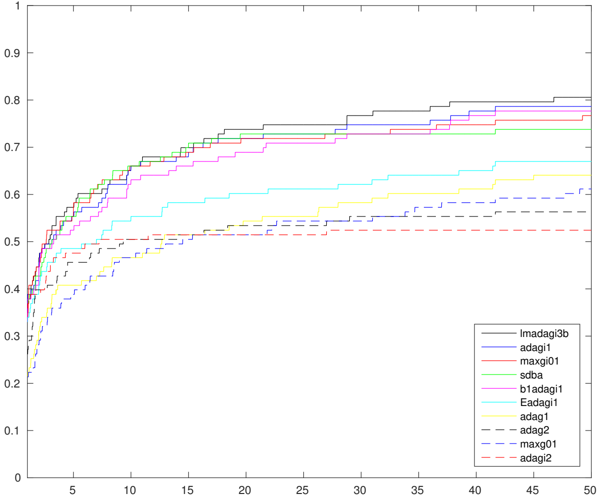
For ASTR1 algorithm we have for all $\mu \in (0,1)$

$$\min_{j\in\{0,\ldots,k\}} \|g_j\| \leq \frac{\kappa_\circ}{\sqrt{k+1}}$$

- No assumption on the gradient boundedness
- This complexity bound can be reached



Some results on small OPM problems



Regularization method

- Compute $H = \nabla_x^2 f(x_k)$ and consider

$$f(x+s) \sim m(s) = f(x) + \alpha \nabla f(x)^\top s + \frac{1}{2} s^\top H s + \frac{1}{6}\sigma \|s\|^3$$

- Approximately minimize m to get s such that

$$\nabla f(x)^\top s + \frac{1}{2} s^T H s + \frac{1}{6}\sigma \|s\|^3 < 0 \quad \text{and} \quad \|g + Hs\| \leq \sigma \|s\|^2$$

- Take σ_k essentially equal to $\prod_{i<k}(1 + \|s_i\|^3)$

Suppose that f has a Lipschitz continuous Hessian. Our algorithm requires at most

$$\mathcal{O}\left(\epsilon^{-3/2}\right)$$

iterations to produce an iterate with $\|g_k\| \leq \epsilon$.

Some numerics with OFFAR2: the framework

Does this work in practice?

Some numerical experiments with

- AR2 (the standard adaptive regularization method using second-order models) and an instance of OFFAR2
- a set of 117 small-dimensional CUTEst problems (as available in Matlab in the OPM collection)
- increasing levels of relative Gaussian noise (both in function values and derivatives): 0%, 5%, 15%, 25%, 50%
- search for an approximate first-order point ($\epsilon = 10^{-6}$)

Reporting:

- a performance measure: $\pi_{\texttt{algo}}$ (see paper for details)
- a reliability ratio: $\rho_{\texttt{algo}}$

Enhanced robustness of $\epsilon^{-3/2}$ smethods

	$\pi_{\texttt{algo}}$	$\rho_{\texttt{algo}}$
with f	0.99	97.48
OFFO	0.83	88.24

No obvious reason to use new method in the absence of noise. . .

$\rho_{\texttt{algo}}$	5%	15%	25%	50%
with f	40.67	30.84	24.54	6.81
OFFO	85.97	80.67	72.69	47.98

. . . but the picture is very different when noise is present (e.g. in ML)!

Conclusions and perspectives

Summary:

- The methods *maxgi* and *avrgi* seem to produce relatively good results. They often outperform the Adagrad-like variants
- The relative behaviour of all variants is not significantly affected by the network architectures. Same for learning rate
- Among Adagrad-like variants of the first class, those with a larger μ handle smaller learning rates better
- Still some gaps between theory and experiments to be filled-in

Perspectives:

- Deterministic and stochastic OFFO methods of higher degree (cubic?) for a better complexity and better performance ??
- The usual: constraints, infinite dimension, multilevel
- More numerical results

Thank you for your attention!

Reference on OFFO

- S. Gratton, S. Jerad, Ph. L. Toint,
  Convergence properties of an Objective-Function-Free Optimization regularization algorithm including an $O(\epsilon^{-3/2})$ SIOPT , 2022
  hal-03718813
- S. Gratton, S. Jerad, Ph. L. Toint,
  First-Order Objective-Function-Free Optimization Algorithms and Their Complexity, 2022
  hal-03718811
- S. Gratton, S. Jerad, Ph. L. Toint,
  Parametric complexity analysis for a class of first-order Adagrad-like algorithms, 2022
  hal-03718810