An Evolving Gradient Resampling Method for Machine Learning

Jorge Nocedal

Northwestern University

NIPS, Montreal 2015

Collaborators

Figen Oztoprak Stefan Solntsev

Richard Byrd

Outline

1. How to improve upon the stochastic gradient method for risk minimization

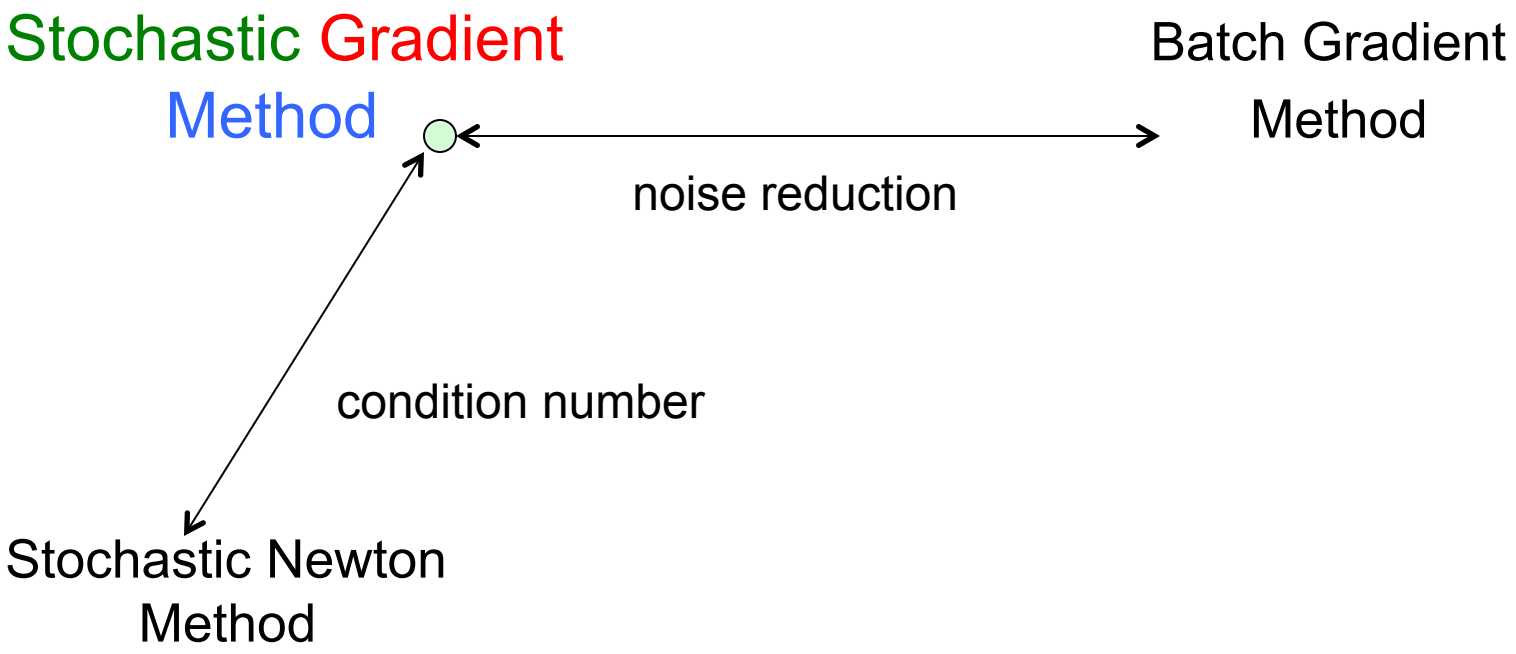
2. Noise reduction methods

   Dynamic Sampling (batching)
   Aggregated Gradient methods (SAG, SVRG, etc)

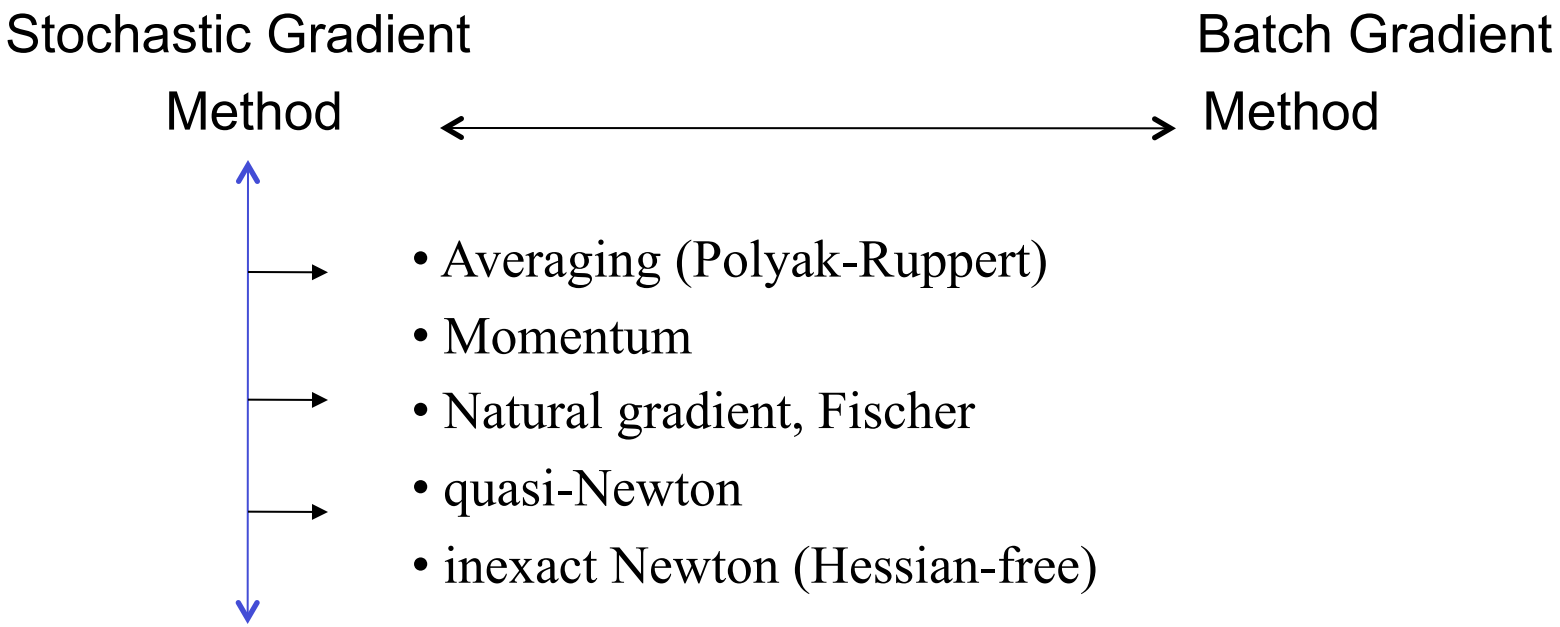
3. Second order methods

4. Propose a noise reduction method that re-uses old gradients *and also* employs dynamic sampling

Organization of optimization methods

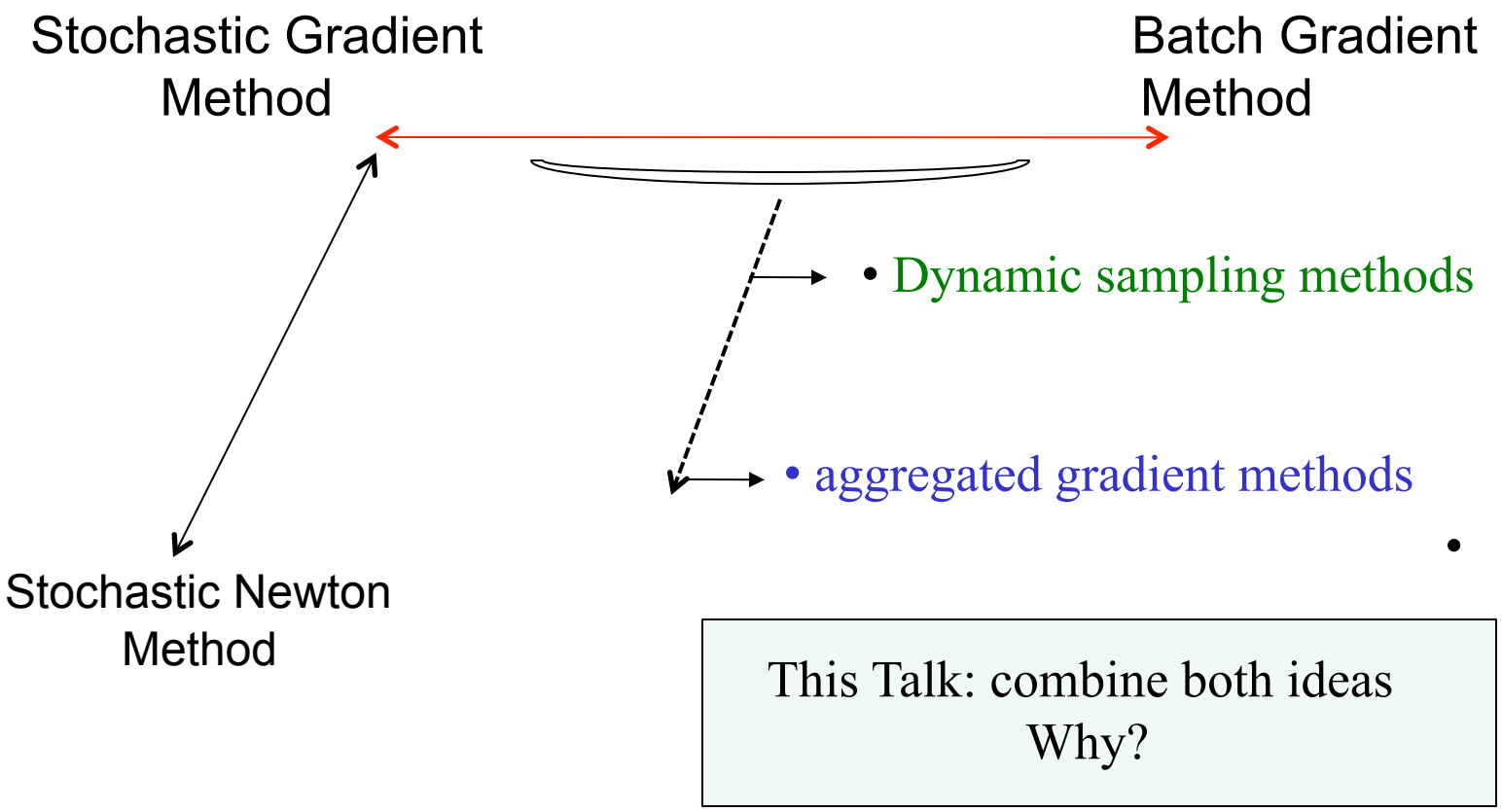


Second-order methods



Stochastic Newton

Noise reducing methods



Objective Function

$$\min_w F(w) = \mathbb{E}[f(w;\xi)]$$

$\xi = (x, y)$ random variable with distribution P

$f(\cdot;\xi)$ composition of loss ℓ and prediction h

$$w_{k+1} = w_k - \alpha_k \nabla f(w_k;\xi_k)$$

stochastic gradient method (SG)

Sample gradient approximation - batch (or mini-batch)

$$F_s(w) = \frac{1}{|S|}\sum_{i\in S} f(w;\xi_i)$$

$$w_{k+1} = w_k - \alpha_k \nabla F_S(w_k)$$

batch (mini) method

Transient behavior of SG

Expected function decrease

$$\mathbb{E}[F(w_{k+1}) - F(w_k)] \leq -\alpha_k \| \nabla F(w_k) \|_2^2 + \alpha_k^2 \mathbb{E} \| \nabla f(w_k, \xi_k) \|^2$$

Initially, gradient decrease dominates; then variance in gradient hinters progress (area of confusion)

To ensure convergence $\alpha_k \to 0$ in SG method to control variance.
Steplength selected to achieve fast initial progress, but this will slow progress in later stages

Dynamic sampling methods reduce gradient variance by increasing batch.
What is the right rate?

Geometric noise reduction

Consider stochastic gradient method with fixed steplength

$$w_{k+1} = w_k - \alpha g(w_k, \xi_k)$$

If the variance of stochastic gradient decreases geometrically, the method yields linear convergence

Lemma If $\exists M > 0, \quad \zeta \in (0,1)$ s.t.

$$\mathbb{E}[\| g(w_k, \xi_k) \|_2^2] - \| \nabla F(w_k) \|_2^2 \leq M \zeta^{k-1}$$

Then

$$\mathbb{E}[F(w_k) - F_*] \leq \nu \rho^{k-1}$$

Schmidt et al

Pasupathy et al

Extension of classical convergence result for gradient method where error in gradient estimates decreases sufficiently rapidly to preserve linear convergence

Optimal work complexity

We can ensure variance condition

$$\mathbb{E}[\,\|\, g(w_k,\xi_k)\,\|_2^2\,] - \|\,\nabla F(w_k)\,\|_2^2 \le M\,\zeta^{k-1}$$

by letting $\boxed{|\,S_k\,| = a^{k-1}}$ $\quad a>1$ $\quad \nabla F_s(w) = \dfrac{1}{|\,S\,|}\sum_{i\in S}\nabla f(w;\xi_i)$

Moreover, we obtain optimal complexity bounds

The total number of stochastic gradient evaluations to achieve

$$\mathbb{E}[F(w_k) - F_*] \le \epsilon \qquad \text{is} \quad O(1/\epsilon)$$

with favorable constants $\quad a \in [1, 1 - \dfrac{\beta c \mu}{2}]^{-1}$

Pasupathy, Glynn et al 2014
Homem-de-Mello, Shapiro 2012

Friedlander, Schmidt 2012
Byrd, Chin, N., Wu 2013

Theorem: Suppose F is strongly convex. Consider

$$w_{k+1} = w_k - (1/L) g_k$$

where S_k is chosen so that variance condition holds and

$$|S_k| \geq \gamma^k \quad \text{for } \gamma > 1.$$

Then

- $$\mathbb{E}[F(w_k) - F(w_*)] \leq C\rho^k \qquad \rho < 1$$
- the number of gradient samples to achieve ε accuracy is

$$O\left(\frac{\kappa}{\varepsilon}\frac{\omega d}{\lambda}\right) \qquad d = \text{no. of variables}$$

$\kappa =$ condition number, $\lambda =$ smallest eigenv of Hessian

$$\| \mathrm{Var} \nabla \ell(w_k; i) \|_1 \leq \omega \qquad \text{(population)}$$

Dynamic sampling (batching)

At every iteration, choose a subset S of {1,…,n} and apply one step of an optimization algorithm to the function

$$F_S(w_k) = \frac{1}{|S|}\sum_{i\in\mathcal{S}} f(w;\xi_i),$$

At the start, a small sample size $|S|$ is chosen

- If optimization step is likely to reduce *F(w)*, sample size is kept unchanged; new sample *S* is chosen; next optimization step taken
- Else, a larger sample size is chosen, a new random sample *S* is selected, a new iterate computed

Many optimization methods can be used. This approach creates the opportunity of employing second order methods

How to implement this in practice?

1. Predetermine a geometric increase, tuning parameter

$$| S_k | = a^{k-1} \qquad a > 1$$

2. Use angle (i.e. variance test)

Ensure bound is satisfied in expectation

$$\| g(w_k) - \nabla F(w_k) \| \leq \theta \| g_k \| \qquad \theta < 1$$

Popular: combination of these two strategies.

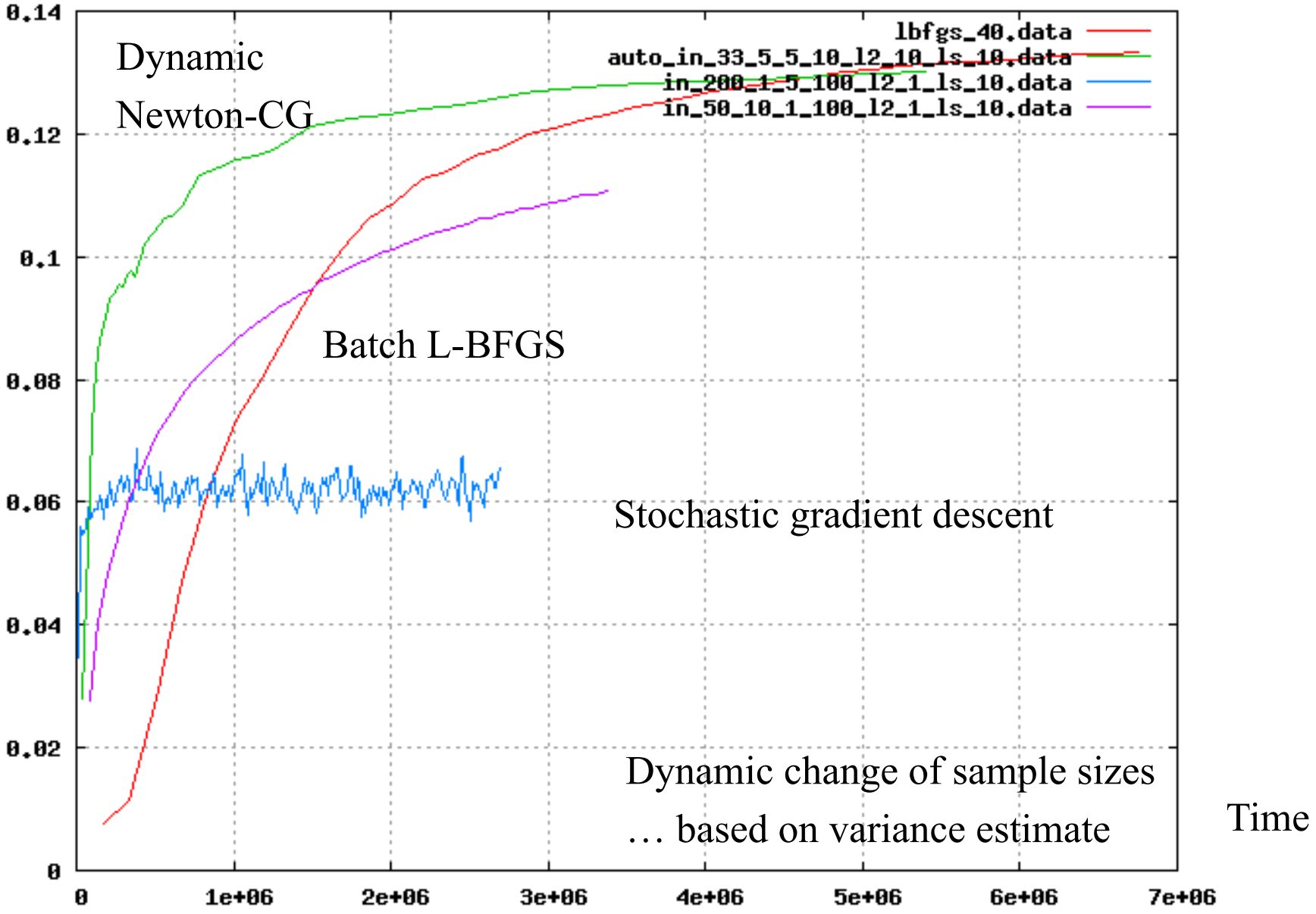
Numerical test

Newton-CG method with dynamic sampling, Armijo line search

$$w_{k+1} = w_k - \alpha_k \nabla^2 F_{H_k}(w_k)^{-1} g_k \qquad \alpha_k \approx 1$$

Test Problem

- From Google VoiceSearch
- 191,607 training points
- 129 classes; 235 features
- 30,315 parameters (variables)
- Small version of production problem
- Multi-class logistic regression
- Initial batch size: 1%; Hessian sample 10%


Dynamic
Newton-CG
Batch L-BFGS
Stochastic gradient descent
Dynamic change of sample sizes
… based on variance estimate
Time
lbfgs_40.data
auto_in_33_5_5_10_l2_10_ls_10.data
in_200_1_5_100_l2_1_ls_10.data
in_50_10_1_100_l2_1_ls_10.data
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
0
1e+06
2e+06
3e+06
4e+06
5e+06
6e+06
7e+06

However, not completely satisfactory

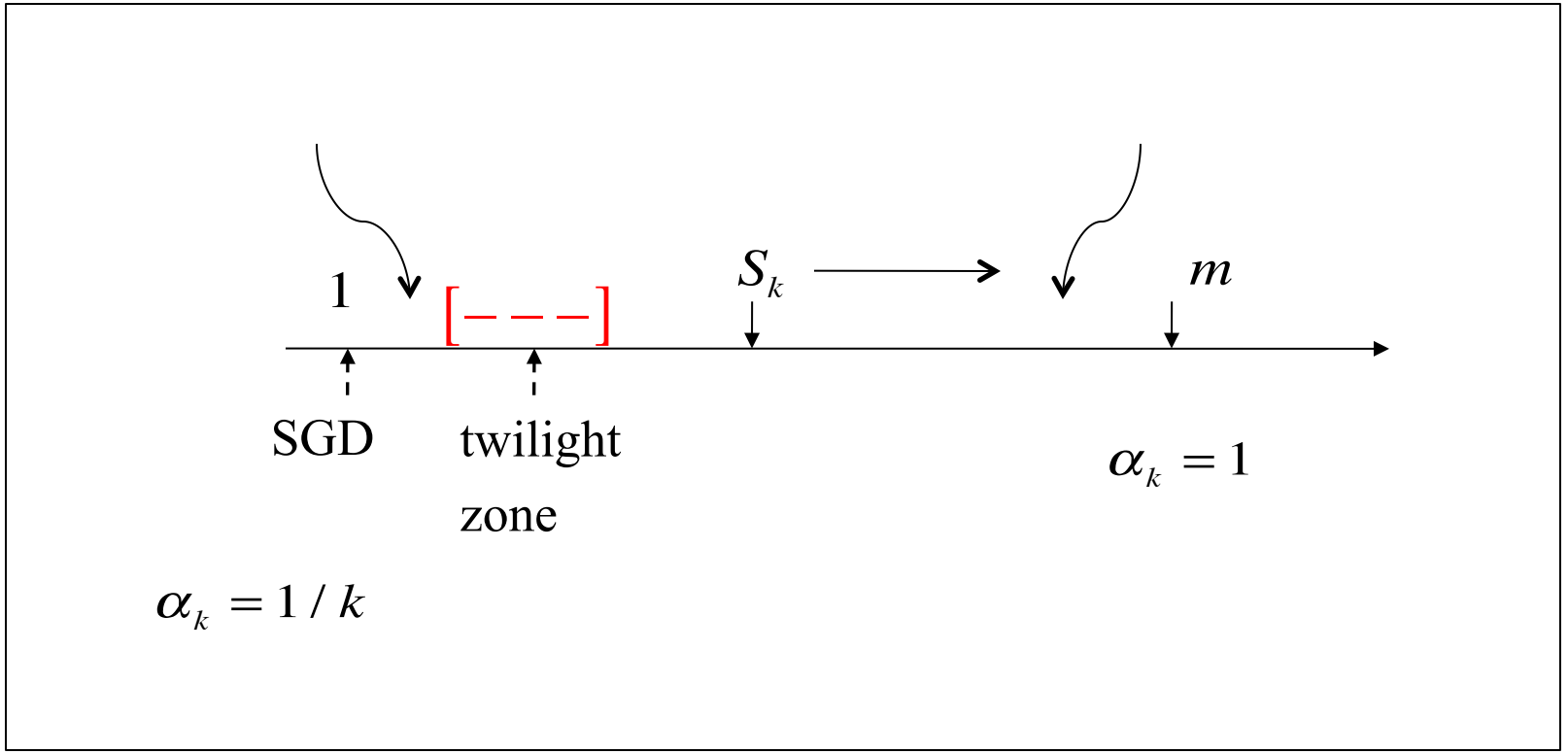
More investigation is needed ….

Particularly:

- Transition between stochastic and batch regimes
- Coordination between step size and batch size
- Use of second order information (one stochastic gradient is not too noisy)
- Can the idea of re-using gradients in a gradient aggregation approach help?

Transition from stochastic to batch regimes

Stochastic process gradient methods



Gradient aggregation could smooth transition

Randomized Aggregated Gradient Methods (for empirical risk min)

Expected Risk: $F(w) = \mathbb{E}[f(w;\xi)]$

Empirical Risk: $F_m(w) = \frac{1}{m}\sum_{i=1}^{n} f(w;\xi_i) = \frac{1}{m}\sum_{i=1}^{n} f_i(w)$

SAG, SAGA, SVRG, etc focus on minimizing empirical risk

Iteration:

$$w_{k+1} = w_k - \alpha\ y_k$$

y_k combination of gradients ∇f_i evaluated at previous iterates ϕ_j

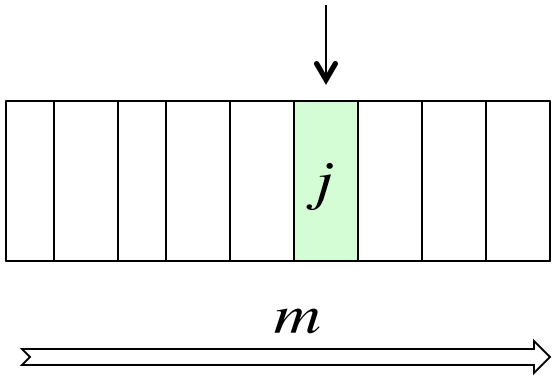
$$y_k = \frac{1}{m}[\nabla f_j(w_k) - \nabla f_j(\phi_{k-1}^j) + \frac{1}{m}\sum_{i=1}^{m} \nabla f_i(\phi_{k-1}^i)]$$

Choose j at random

SAG

Example of Gradient Aggregation Methods

$$y_k = \frac{1}{m}[\nabla f_j(w_k) - \nabla f_j(\phi_{k-1}^j) + \frac{1}{m}\sum_{i=1}^{m} \nabla f_i(\phi_{k-1}^i)]$$



SAG

SAG, SAGA, SVRG

$$F_m(w) = \frac{1}{m}\sum_{i=1} f(w;\xi_i) = \frac{1}{m}\sum_{i=1} f_i(w)$$

Achieve linear rate of convergence in expectation (after a full initialization pass)

EGR Method

The Evolving Gradient Resampling Method
for Expected Risk Minimization

Proposed algorithm

1. Minimizes expected risk (not training error)
2. Stores previous gradients and updates several (s_k) at each
3. iteration
4. Additional (u_k) gradients are computed at current iterate
5. Total amount of stored gradients increases monotonically
6. Shares properties with dynamic sampling and gradient aggregation methods

Goal: analyze an algorithm of this generality (interesting in its own right)
Finding right balance between re-using old information and batching can result in efficient method.

The EGR method

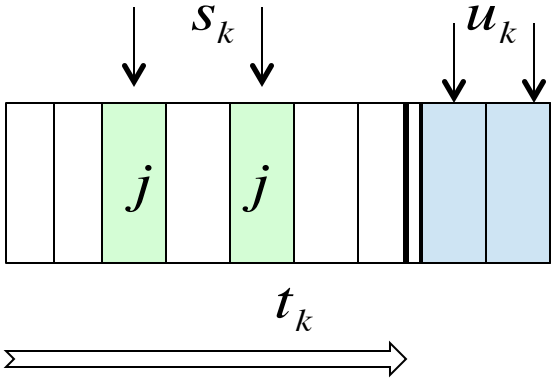
$$y_k = \frac{1}{t_k + u_k}\Big(\sum_{j \in S_k} [\nabla f_j(w_k) - \nabla f_j(\phi_{k-1}^j)] + \sum_{i=1}^{t_k} \nabla f_i(\phi_{k-1}^i) + \sum_{j \in U_k} \nabla f_j(w_k) \Big)$$

t_k number of gradients in storage at start of iteration k

U_k indices of new gradients sampled at w_k $\quad u_k = |U_k|$

S_k indices of previously computed gradients that are updated

$s_k = |S_k|$



Evaluated at w_k

How should s_k and u_k be controled?

Related work: Frostig et al 2014, Babanezhad et al 2015

Algorithms included in framework

stochastic gradient method: $s_k = 0, u_k = 1$

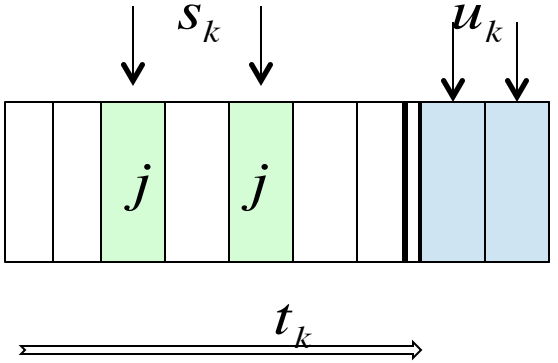
dynamic sampling method: $s_k = 0, u_k = \text{function}(k)$

aggregated gradient: $s_k = \text{constant}, u_k = 0$

EGR lin: $s_k = 0, u_k = r$

EGR quad: $s_k = rk, \; u_k = r$

EGR exp: $s_k = u_k \approx a^k$



Assumptions:

1) $s_k = u_k = a^k \quad a \in \mathbb{N}^+$ geometric growth

2) F strongly convex, f_i Lipschitz continuous gradients

3) $\text{tr}\,(\text{var}\,[\nabla f_i(w)]) \leq v^2 \quad \forall w$

Lemma:

$$\begin{pmatrix} \mathrm{E}[\mathrm{E}_k[e_k]] \\ \mathrm{E}[\,\|\, w_k - w_* \,\|\,] \\ \sigma_k \end{pmatrix} \leq M \begin{pmatrix} \mathrm{E}[\mathrm{E}_k[e_{k-1}]] \\ \mathrm{E}[\,\|\, w_{k-1} - w_* \,\|\,] \\ \sigma_{k-1} \end{pmatrix}$$

Lyapunov function

$$e_k = \frac{1}{t_{k+1}} \sum_{i=1}^{t_{k+1}} \| \nabla f_i(\phi_k^i) - \nabla f_i(w_k) \| \qquad \sigma_k = \sqrt{v^2 / t_{k+1}}$$

$$M = \begin{pmatrix} \frac{1-\eta}{1+\eta}(1+\alpha L) & \frac{1-\eta}{1+\eta}\alpha L & \frac{1-\eta}{1+\eta}\alpha \\ \alpha L & 1-\alpha\mu & \alpha \\ 0 & 0 & \sqrt{\frac{1}{1+\eta}} \end{pmatrix}$$

η : probability that an old gradient is recomputed

α : steplength

L: Lipschitz constant

μ : strong convexity parameter

Lemma. For sufficiently small α the spectral radius of M satisfies

$$\rho_M < 1$$

Theorem: If α_k is chosen small enough

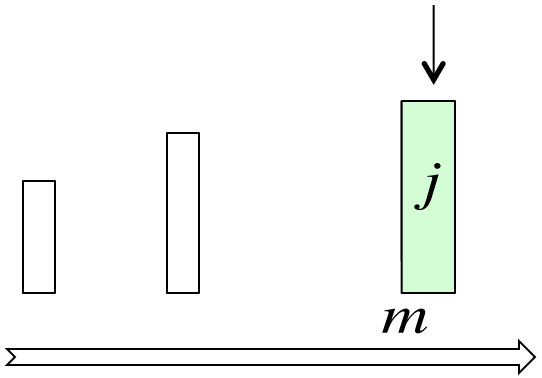
$$\mathbb{E}\,\|w_k - w_*\| \leq c^k \beta$$ R-linear convergence

SAG special case: $t_k = m,\ u_k = 0,\ s_k =$ constant

Simple proof of R-linear convergence of SAG but with a larger constant

Related Methods

- Streaming SVRG (Frostig et al 2014)
- Stop wasting gradients (Babanezhad et al 2015)



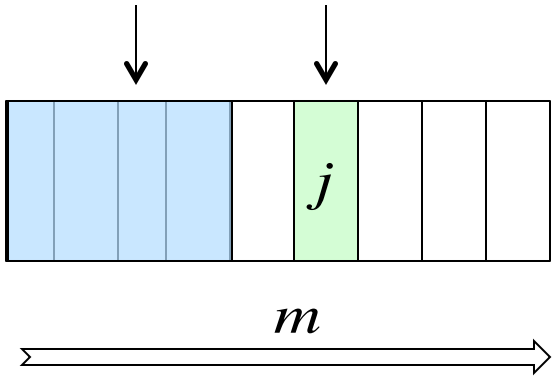
SVRG

$$y_k = \frac{1}{m}[\nabla f_j(w_k) - \nabla f_j(\phi_{k-1}^j) + \frac{1}{m}\sum_{i=1}^{m} \nabla f_i(\phi_{k-1}^i)]$$

SAG(A) initialization phase

1. Sample $j \in \{1,\ldots,m\}$ at random
2. Compute $\nabla f^j(w_k)$
3. If j has been sampled earlier, replace old gradient
4. Else add new $\nabla f^j(w_k)$ to aggregated gradient

(memory grows)



SAG

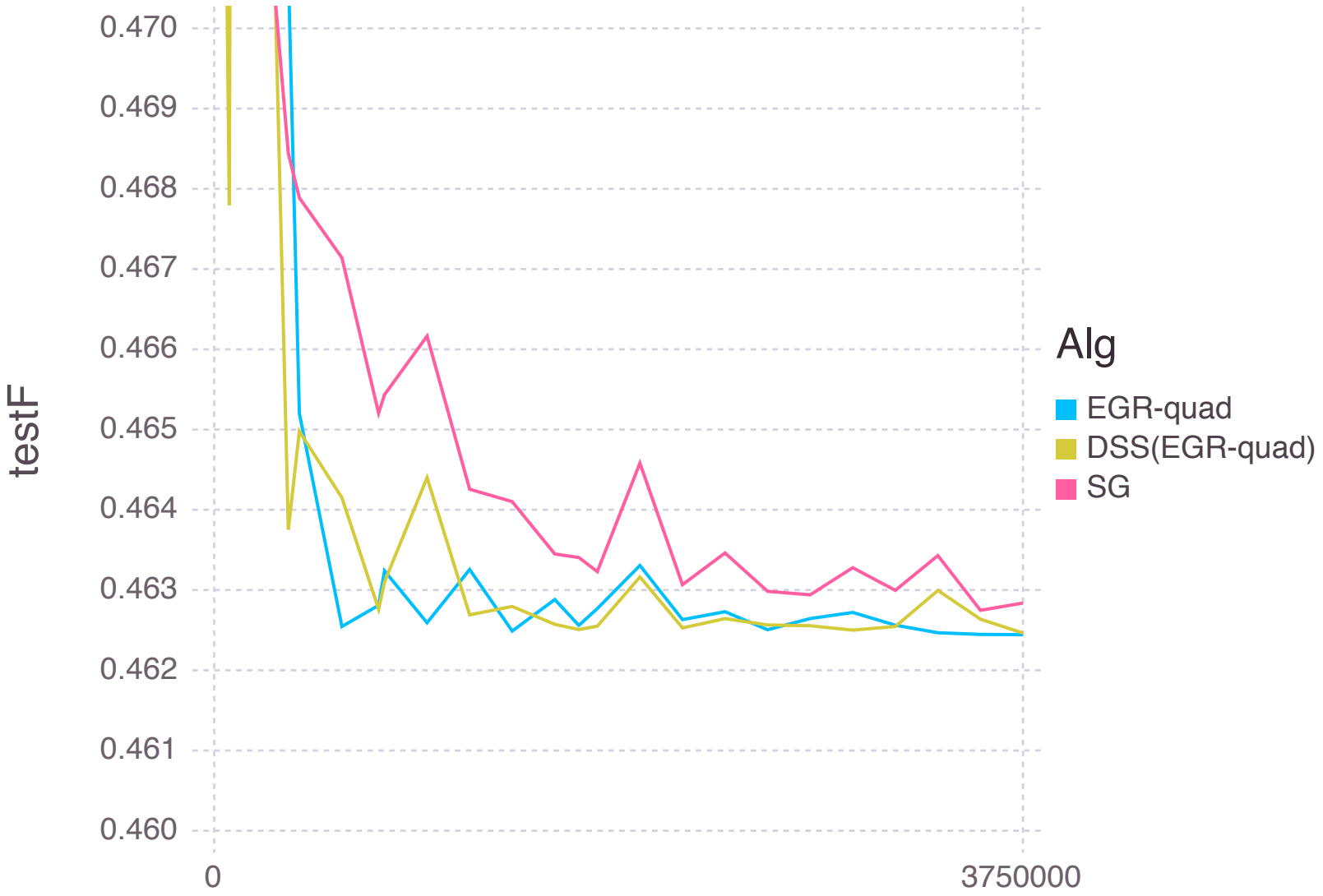
$$y_k = \frac{1}{m}[\nabla f_j(w_k) - \nabla f_j(\phi_{k-1}^j) + \frac{1}{m}\sum_{i=1}^{m} \nabla f_i(\phi_{k-1}^i)]$$

Numerical Result

1. Comparison of EGR with various growth rates
2. Comparison with SGD
3. Comparison with SAG-init and SAGA-init

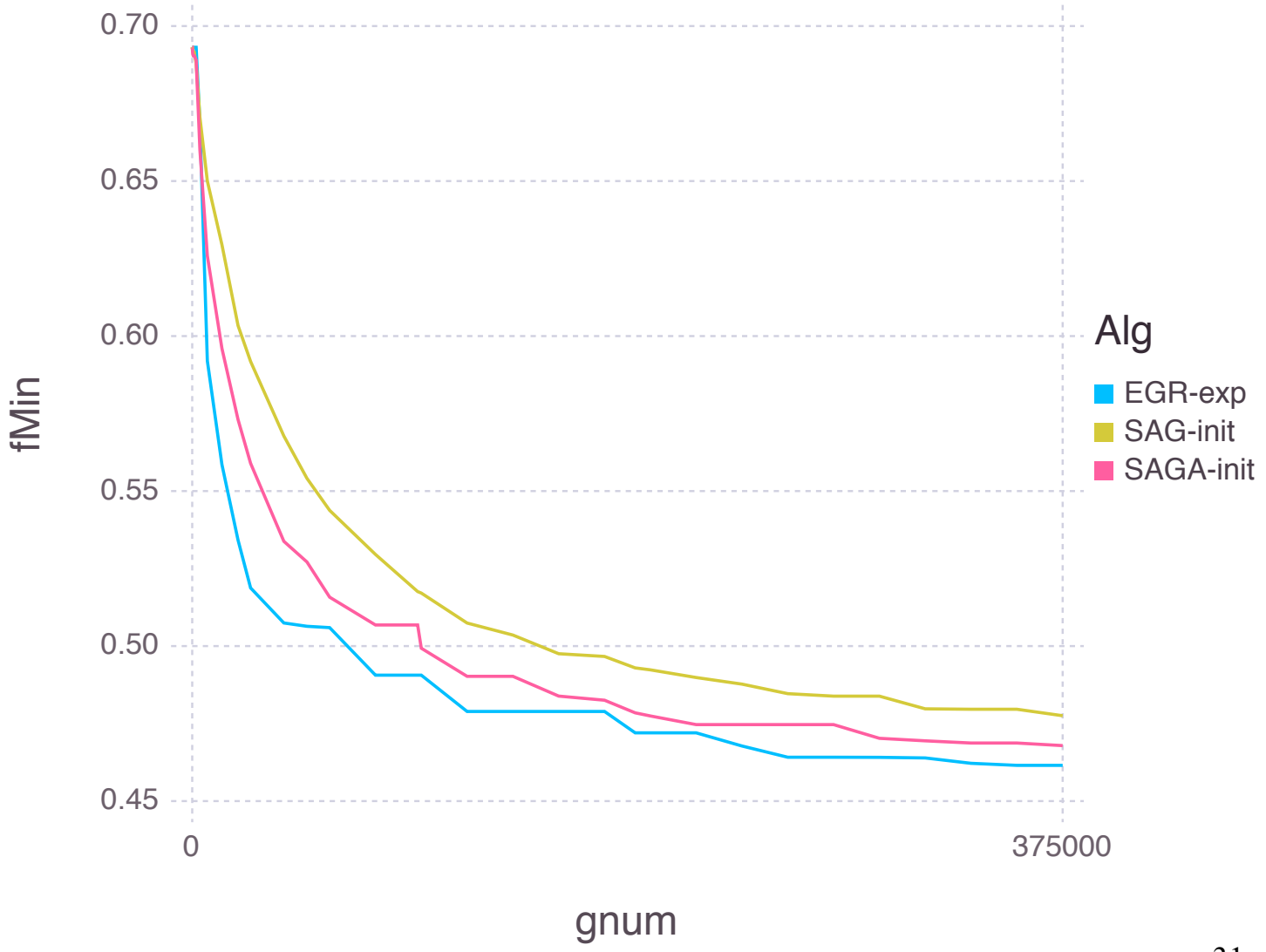
Goal: analyze an algorithm of this generality (interesting in its own right)
Finding right balance between re-using old information and batching can result in efficient method.

Susy EGR vs SG vs Dynamic Sampling

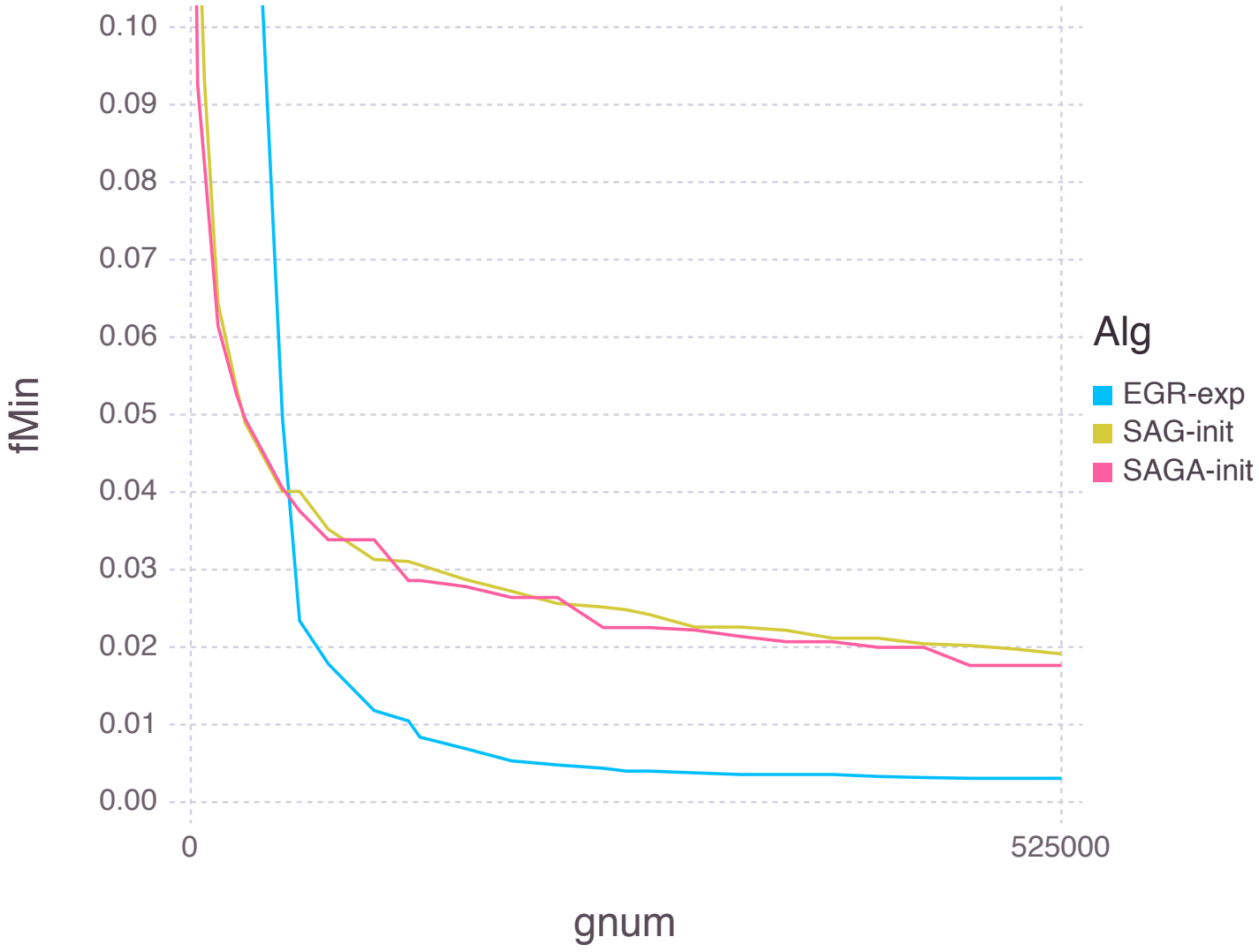


Random

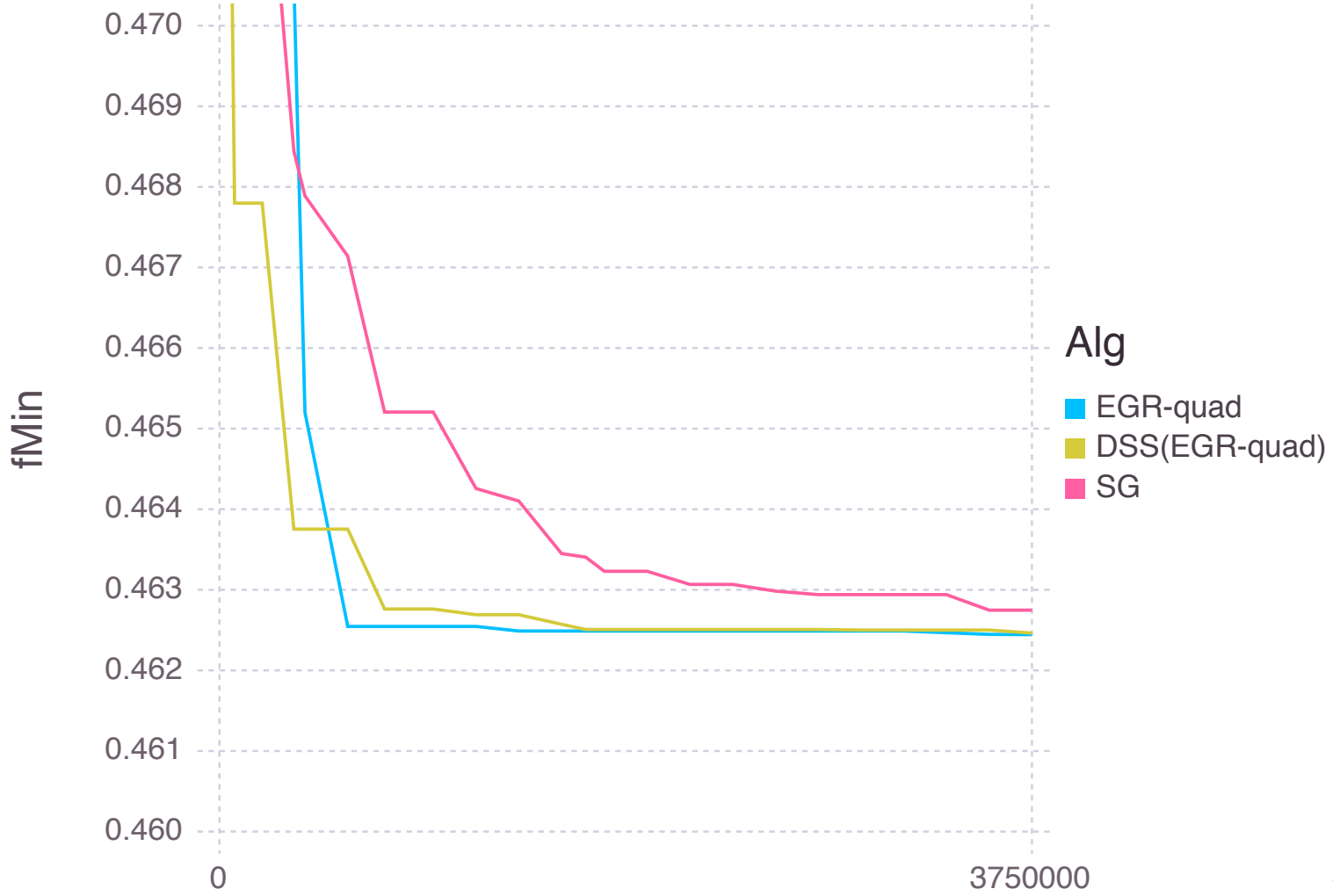
Comparing with SAG initialization



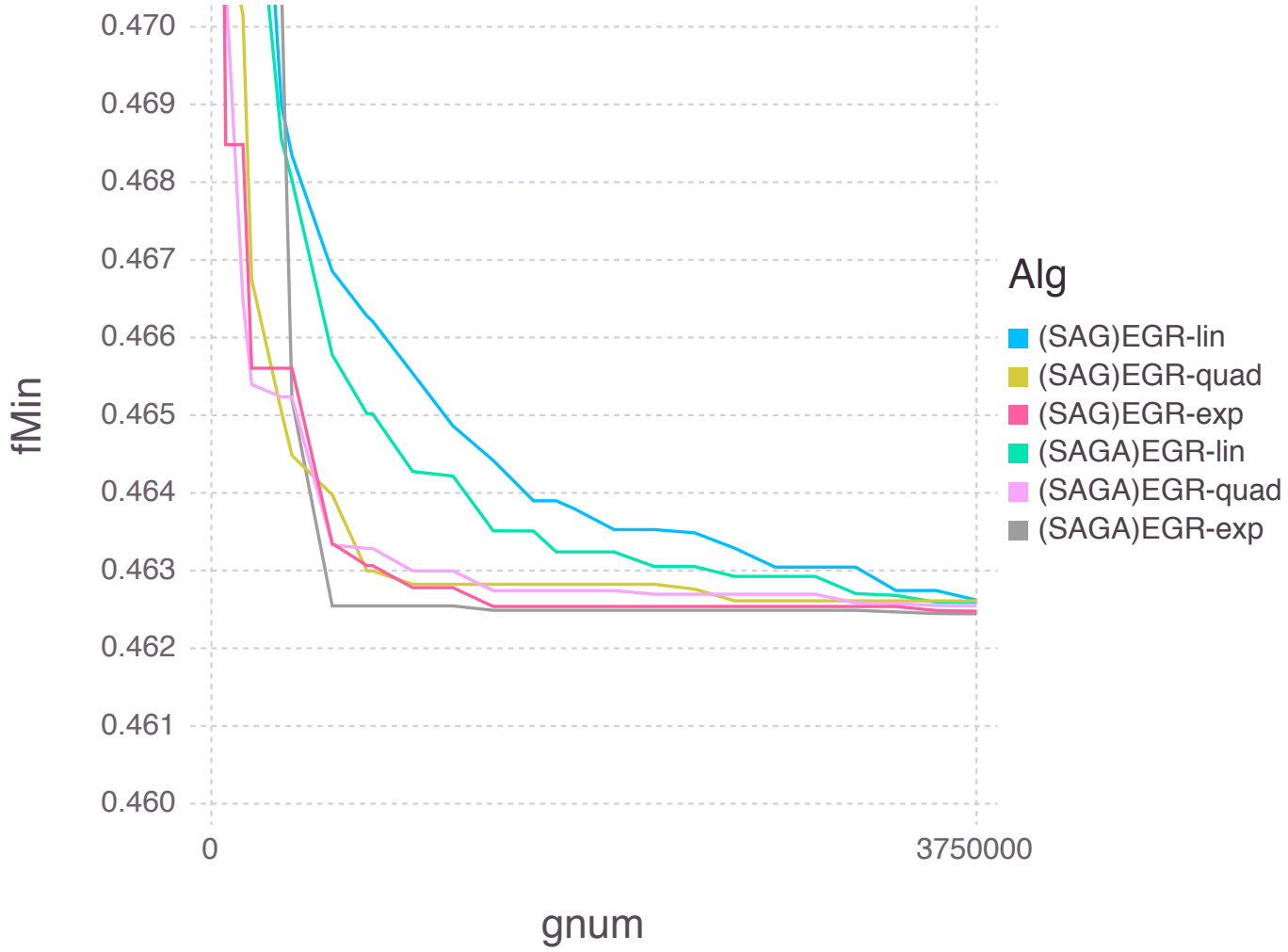
Random Larger initial batch for EGR



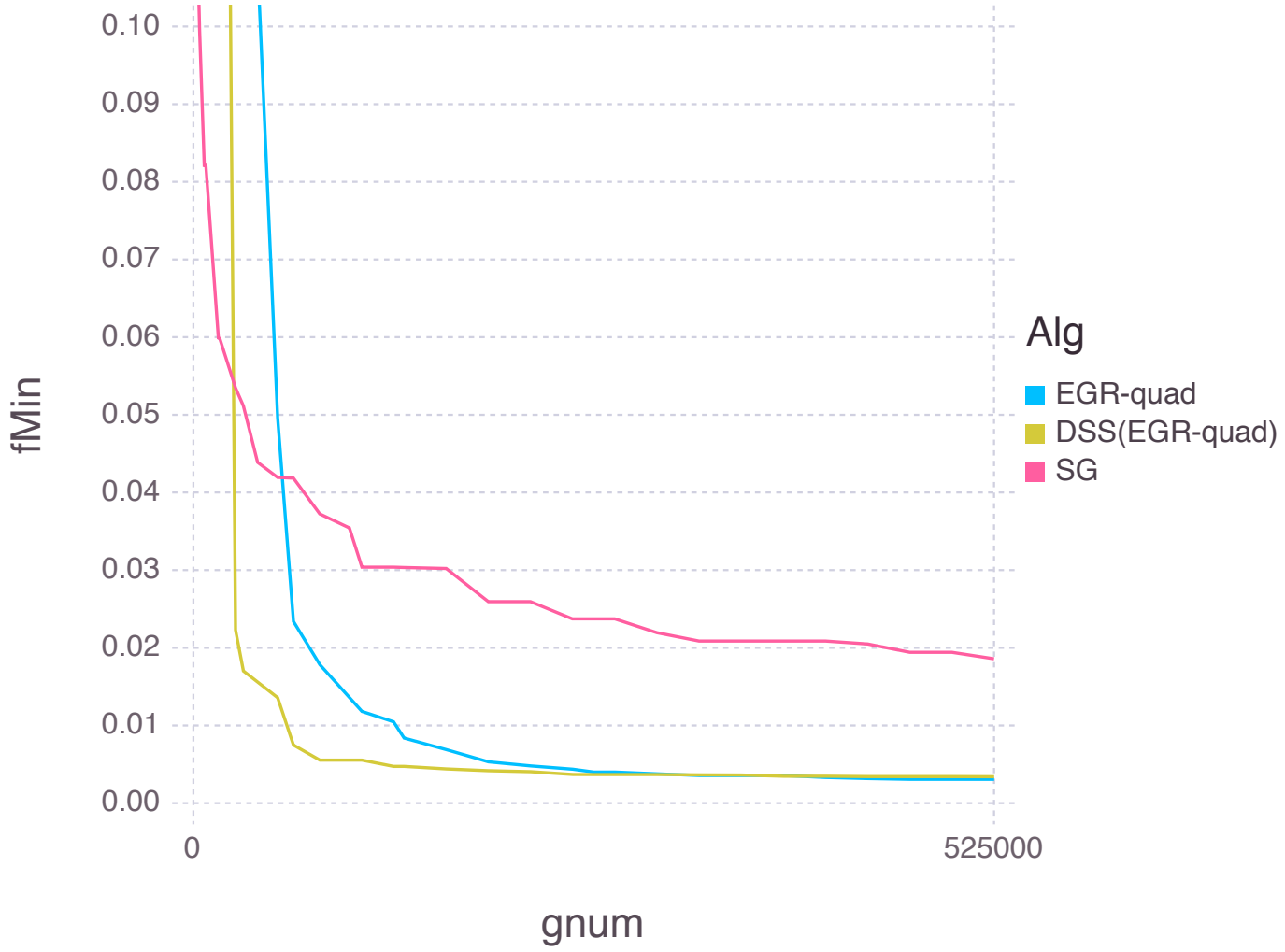
Alpha


0.470
0.469
0.468
0.467
0.466
0.465
0.464
0.463
0.462
0.461
0.460
fMin
0
3750000
Alg
EGR-quad
DSS(EGR-quad)
SG

EGR with various growth rates (Alpha)



Random


0.10
0.09
0.08
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0.00
fMin
0
525000
gnum
Alg
EGR-quad
DSS(EGR-quad)
SG

The End