Overview of the Stochastic Gradient Method

Lecture Outline

1. Stochastic Gradient Algorithm
2. Connexion with Stochastic Approximation
3. Asymptotic Efficiency and Averaging
4. Practical Considerations

Deterministic Constrained Optimization Problem

General optimization problem $(\mathcal{P})$

$$\min_{u \in U^{\mathrm{ad}} \subset \mathbb{U}} J(u)$$

- U^{ad} closed convex subset of an Hilbert space $\mathbb{U}$,
- J cost function $\mathbb{U} \longrightarrow \mathbb{R}$, satisfying some assumptions
  - convexity,
  - coercivity,
  - continuity,
  - differentiability.

Extension of Problem $(\mathcal{P})$ — Open-Loop Case (1)

Consider Problem $(\mathcal{P})$ and suppose that J is the expectation of a function j, depending on a random variable $\boldsymbol{W}$ defined on a probability space $(\Omega, \mathcal{A}, \mathbb{P})$ and valued on $(\mathbb{W}, \mathcal{W})$:

$$J(u) = \mathbb{E}\big(j(u, \boldsymbol{W})\big) \ .$$

Then the optimization problem writes

$$\min_{u \in U^{\mathrm{ad}}} \mathbb{E}\big(j(u, \boldsymbol{W})\big) \ .$$

Decision u is a deterministic variable. The available information is the probability law of $\boldsymbol{W}$ (no on-line observation of $\boldsymbol{W}$), that is, an open-loop situation. The information structure is trivial, but...

$\rightsquigarrow$ **main difficulty:** calculation of the expectation.

Extension of Problem $(\mathcal{P})$ — Open-Loop Case (1)

Consider Problem $(\mathcal{P})$ and suppose that J is the expectation of a function j, depending on a random variable $\boldsymbol{W}$ defined on a probability space $(\Omega, \mathcal{A}, \mathbb{P})$ and valued on $(\mathbb{W}, \mathcal{W})$:

$$J(u) = \mathbb{E}\big(j(u, \boldsymbol{W})\big) \ .$$

Then the optimization problem writes

$$\min_{u \in U^{\mathrm{ad}}} \mathbb{E}\big(j(u, \boldsymbol{W})\big) \ .$$

Decision u is a deterministic variable. The available information is the probability law of $\boldsymbol{W}$ (no on-line observation of $\boldsymbol{W}$), that is, an open-loop situation. The information structure is trivial, but...

$\rightsquigarrow$ **main difficulty:** calculation of the expectation.

Extension of Problem $(\mathcal{P})$ — Open-Loop Case (1)

Consider Problem $(\mathcal{P})$ and suppose that J is the expectation of a function j, depending on a random variable $\boldsymbol{W}$ defined on a probability space $(\Omega, \mathcal{A}, \mathbb{P})$ and valued on $(\mathbb{W}, \mathcal{W})$:

$$J(u) = \mathbb{E}\big(j(u, \boldsymbol{W})\big) \ .$$

Then the optimization problem writes

$$\min_{u \in U^{\mathrm{ad}}} \mathbb{E}\big(j(u, \boldsymbol{W})\big) \ .$$

Decision u is a deterministic variable. The available information is the probability law of $\boldsymbol{W}$ (no on-line observation of $\boldsymbol{W}$), that is, an open-loop situation. The information structure is trivial, but...

$\rightsquigarrow$ **main difficulty:** calculation of the expectation.

Extension of Problem $(\mathcal{P})$ — Open-Loop Case (1)

Consider Problem $(\mathcal{P})$ and suppose that J is the expectation of a function j, depending on a random variable $\boldsymbol{W}$ defined on a probability space $(\Omega, \mathcal{A}, \mathbb{P})$ and valued on $(\mathbb{W}, \mathcal{W})$:

$$J(u) = \mathbb{E}\big(j(u, \boldsymbol{W})\big) \ .$$

Then the optimization problem writes

$$\min_{u \in U^{\mathrm{ad}}} \mathbb{E}\big(j(u, \boldsymbol{W})\big) \ .$$

Decision u is a deterministic variable. The available information is the probability law of $\boldsymbol{W}$ (no on-line observation of $\boldsymbol{W}$), that is, an open-loop situation. The information structure is trivial, but...

$\rightsquigarrow$ **main difficulty:** calculation of the expectation.

Extension of Problem $(\mathcal{P})$ — Open-Loop Case (2)

Solution using Exact Quadrature

$$J(u) = \mathbb{E}\big(j(u, \boldsymbol{W})\big) \quad , \quad \nabla J(u) = \mathbb{E}\big(\nabla_u j(u, \boldsymbol{W})\big) \ .$$

Projected gradient algorithm:

$$u^{(k+1)} = \mathrm{proj}_{U^{\mathrm{ad}}}\left(u^{(k)} - \epsilon \nabla J\big(u^{(k)}\big)\right) \ .$$

Sample Average Approximation (SAA)

Obtain a realization $(w^{(1)}, \ldots, w^{(k)})$ of a k-sample of $\boldsymbol{W}$ and minimize the Monte Carlo approximation of J:

$$u^{(k)} \in \arg\min_{u \in U^{\mathrm{ad}}} \frac{1}{k} \sum_{l=1}^{k} j(u, w^{(l)}) \ .$$

Note that $u^{(k)}$ depends on the realization $(w^{(1)}, \ldots, w^{(k)})$!

Extension of Problem $(\mathcal{P})$ — Open-Loop Case (2)

Solution using Exact Quadrature

$$J(u) = \mathbb{E}\big(j(u, \boldsymbol{W})\big) \quad , \quad \nabla J(u) = \mathbb{E}\big(\nabla_u j(u, \boldsymbol{W})\big) \ .$$

Projected gradient algorithm:

$$u^{(k+1)} = \mathrm{proj}_{U^{\mathrm{ad}}}\left(u^{(k)} - \epsilon \nabla J\big(u^{(k)}\big)\right) \ .$$

Sample Average Approximation (SAA)

Obtain a realization $(w^{(1)}, \ldots, w^{(k)})$ of a k-sample of $\boldsymbol{W}$ and minimize the Monte Carlo approximation of J:

$$u^{(k)} \in \operatorname*{arg\,min}_{u \in U^{\mathrm{ad}}} \frac{1}{k} \sum_{l=1}^{k} j(u, w^{(l)}) \ .$$

Note that $u^{(k)}$ depends on the realization $(w^{(1)}, \ldots, w^{(k)})$!

Extension of Problem $(\mathcal{P})$ — Open-Loop Case (3)

Stochastic Gradient Method

Underlying ideas:

- incorporate the realizations $(w^{(1)}, \ldots, w^{(k)}, \ldots)$ of samples of $\boldsymbol{W}$ one by one into the algorithm.
- use an easily computable approximation of the gradient ∇J, e.g. replace $\nabla J(u^{(k)})$ by $\nabla_u j(u^{(k)}, w^{(k+1)})$,

These considerations lead to the following algorithm:

$$u^{(k+1)} = \mathrm{proj}_{\mathcal{U}_{\mathrm{ad}}}\left(u^{(k)} - \epsilon^{(k)} \nabla_u j(u^{(k)}, w^{(k+1)})\right)$$

Iterations of the gradient algorithm are used a) to move towards the solution *and* b) to refine the Monte-Carlo sampling process.

Extension of Problem $(\mathcal{P})$ — Open-Loop Case (3)

Stochastic Gradient Method

Underlying ideas:

- incorporate the realizations $(w^{(1)}, \ldots, w^{(k)}, \ldots)$ of samples of $\boldsymbol{W}$ one by one into the algorithm.
- use an easily computable approximation of the gradient ∇J, e.g. replace $\nabla J(u^{(k)})$ by $\nabla_u j(u^{(k)}, w^{(k+1)})$,

These considerations lead to the following algorithm:

$$u^{(k+1)} = \mathrm{proj}_{U^{\mathrm{ad}}} \left(u^{(k)} - \epsilon^{(k)} \nabla_u j(u^{(k)}, w^{(k+1)}) \right) .$$

Iterations of the gradient algorithm are used a) to move towards the solution *and* b) to refine the Monte-Carlo sampling process.

1 Stochastic Gradient Algorithm

2 Connexion with Stochastic Approximation

3 Asymptotic Efficiency and Averaging

4 Practical Considerations

Stochastic Gradient (SG) algorithm

Standard Stochastic Gradient Algorithm

$$\min_{u \in U^{\mathrm{ad}} \subset \mathbb{U}} \mathbb{E}\big(j(u, \boldsymbol{W})\big) \ . \qquad (\mathcal{P}_{\mathrm{ol}})$$

1. Let $u^{(0)} \in U^{\mathrm{ad}}$ and choose a positive real sequence $\{\epsilon^{(k)}\}_{k \in \mathbb{N}}$.
2. At iteration $(k+1)$, draw a realization $w^{(k+1)}$ of the r.v. $\boldsymbol{W}$.
3. Compute the gradient of j and update $u^{(k+1)}$ by the formula:

$$u^{(k+1)} = \mathrm{proj}_{U^{\mathrm{ad}}} \left(u^{(k)} - \epsilon^{(k)} \nabla_u j\big(u^{(k)}, w^{(k+1)}\big) \right) \ .$$

4. Set $k = k+1$ and go to step 2.

Note that $(w^{(1)}, \ldots, w^{(k)}, \ldots)$ is a realization of a ∞-sample of W

$\rightsquigarrow$ numerical implementation of the stochastic gradient method.

Stochastic Gradient (SG) algorithm

Standard Stochastic Gradient Algorithm

$$\min_{u \in U^{\mathrm{ad}} \subset \mathbb{U}} \mathbb{E}\big(j(u, \boldsymbol{W})\big) \; . \qquad (\mathcal{P}_{\mathrm{ol}})$$

1. Let $u^{(0)} \in U^{\mathrm{ad}}$ and choose a positive real sequence $\{\epsilon^{(k)}\}_{k \in \mathbb{N}}$.
2. At iteration $(k+1)$, draw a realization $w^{(k+1)}$ of the r.v. $\boldsymbol{W}$.
3. Compute the gradient of j and update $u^{(k+1)}$ by the formula:

$$u^{(k+1)} = \mathrm{proj}_{U^{\mathrm{ad}}} \left(u^{(k)} - \epsilon^{(k)} \nabla_u j\big(u^{(k)}, w^{(k+1)}\big) \right) \; .$$

4. Set $k = k+1$ and go to step 2.

Note that $(w^{(1)}, \ldots, w^{(k)}, \ldots)$ is a realization of a ∞-sample of $\boldsymbol{W}$

$\rightsquigarrow$ numerical implementation of the stochastic gradient method.

Probabilistic Considerations (1)

In order to study the convergence of the algorithm, it is necessary to cast it in the adequate probabilistic framework:

$$\boldsymbol{U}^{(k+1)} = \mathrm{proj}_{U^{\mathrm{ad}}} \left(\boldsymbol{U}^{(k)} - \epsilon^{(k)} \nabla_u j(\boldsymbol{U}^{(k)}, \boldsymbol{W}^{(k+1)}) \right) .$$

where $\{\boldsymbol{W}^{(k)}\}_{k \in \mathbb{N}}$ is a infinite-dimensional sample of $\boldsymbol{W}$.[3]

$\rightsquigarrow$ Iterative relation involving random variables.

- Convergence in law.
- Convergence in probability.
- Convergence in L^p norm.
- Almost sure convergence (the "intuitive" one).

[3] Note that $(\Omega, \mathcal{A}, \mathbb{P})$ has to be "big enough" to support such a sample.

Probabilistic Considerations (1)

In order to study the convergence of the algorithm, it is necessary to cast it in the adequate probabilistic framework:

$$\boldsymbol{U}^{(k+1)} = \mathrm{proj}_{U^{\mathrm{ad}}}\left(\boldsymbol{U}^{(k)} - \epsilon^{(k)} \nabla_u j(\boldsymbol{U}^{(k)}, \boldsymbol{W}^{(k+1)})\right) .$$

where $\{\boldsymbol{W}^{(k)}\}_{k\in\mathbb{N}}$ is a infinite-dimensional sample of $\boldsymbol{W}$.[3]

$\rightsquigarrow$ Iterative relation involving random variables.

- Convergence in law.
- Convergence in probability.
- Convergence in L^p norm.
- Almost sure convergence (the "intuitive" one).

[3] Note that $(\Omega, \mathcal{A}, \mathbb{P})$ has to be "big enough" to support such a sample.

Probabilistic Considerations (2)

An iteration of the algorithm is represented by the general relation:

$$\boldsymbol{U}^{(k+1)} = \mathcal{R}^{(k)}\left(\boldsymbol{U}^{(k)}, \boldsymbol{W}^{(k+1)}\right) .$$

Let $\mathcal{F}^{(k)}$ be the σ-field generated by $(\boldsymbol{W}^{(1)}, \ldots, \boldsymbol{W}^{(k)})$.

Probabilistic Considerations (2)

An iteration of the algorithm is represented by the general relation:

$$\boldsymbol{U}^{(k+1)} = \mathcal{R}^{(k)}\left(\boldsymbol{U}^{(k)}, \boldsymbol{W}^{(k+1)}\right) .$$

Let $\mathcal{F}^{(k)}$ be the σ-field generated by $(\boldsymbol{W}^{(1)}, \ldots, \boldsymbol{W}^{(k)})$.

- Since $\boldsymbol{U}^{(k)}$ is $\mathcal{F}^{(k)}$-mesurable for all k, we have

$$\mathbb{E}\left(\boldsymbol{U}^{(k)} \mid \mathcal{F}^{(k)}\right) = \boldsymbol{U}^{(k)} .$$

- Since $\boldsymbol{W}^{(k+1)}$ is independent of $\mathcal{F}^{(k)}$, we have (disintegration) that the conditional expectation of $\boldsymbol{U}^{(k+1)}$ w.r.t. $\mathcal{F}^{(k)}$ merely consists of a standard expectation:

$$\mathbb{E}\left(\boldsymbol{U}^{(k+1)} \mid \mathcal{F}^{(k)}\right)(\omega) = \int_{\Omega} \mathcal{R}^{(k)}\left(\boldsymbol{U}^{(k)}(\omega), \boldsymbol{W}(\omega')\right) \mathrm{d}\mathbb{P}(\omega') .$$

Probabilistic Considerations (2)

An iteration of the algorithm is represented by the general relation:

$$\boldsymbol{U}^{(k+1)} = \mathcal{R}^{(k)}\big(\boldsymbol{U}^{(k)}, \boldsymbol{W}^{(k+1)}\big) \,.$$

Let $\mathcal{F}^{(k)}$ be the σ-field generated by $(\boldsymbol{W}^{(1)}, \ldots, \boldsymbol{W}^{(k)})$.

- Since $\boldsymbol{U}^{(k)}$ is $\mathcal{F}^{(k)}$-mesurable for all k, we have

$$\mathbb{E}\big(\boldsymbol{U}^{(k)} \,\big|\, \mathcal{F}^{(k)}\big) = \boldsymbol{U}^{(k)} \,.$$

- Since $\boldsymbol{W}^{(k+1)}$ is independent of $\mathcal{F}^{(k)}$, we have (disintegration) that the conditional expectation of $\boldsymbol{U}^{(k+1)}$ w.r.t. $\mathcal{F}^{(k)}$ merely consists of a standard expectation:

$$\mathbb{E}\big(\boldsymbol{U}^{(k+1)} \,\big|\, \mathcal{F}^{(k)}\big)(\omega) = \int_{\Omega} \mathcal{R}^{(k)}\big(\boldsymbol{U}^{(k)}(\omega), \boldsymbol{W}(\omega')\big)\, \mathrm{d}\mathbb{P}(\omega') \,.$$

Example: Estimation of an Expectation (1)

Let $\boldsymbol{W}$ be a real-valued random variable defined on $(\Omega, \mathcal{A}, \mathbb{P})$.
We want to compute an estimate of its expectation

$$\mathbb{E}(\boldsymbol{W}) = \int_{\Omega} \boldsymbol{W}(\omega) \, \mathrm{d}\mathbb{P}(\omega) \, .$$

Monte Carlo method: obtain a k-sample $(\boldsymbol{W}^{(1)}, \ldots, \boldsymbol{W}^{(k)})$ of $\boldsymbol{W}$ and compute the associated arithmetic mean:

$$\boldsymbol{U}^{(k)} = \frac{1}{k} \sum_{l=1}^{k} \boldsymbol{W}^{(l)} \, .$$

By the Strong Law of Large Numbers (SLLN), the sequence of random variables $\left\{\boldsymbol{U}^{(k)}\right\}_{k \in \mathbb{N}}$ almost surely converges to $\mathbb{E}(\boldsymbol{W})$.

Example: Estimation of an Expectation (1)

Let $\boldsymbol{W}$ be a real-valued random variable defined on $(\Omega, \mathcal{A}, \mathbb{P})$.
We want to compute an estimate of its expectation

$$\mathbb{E}(\boldsymbol{W}) = \int_{\Omega} \boldsymbol{W}(\omega) \, \mathrm{d}\mathbb{P}(\omega) \, .$$

Monte Carlo method: obtain a k-sample $(\boldsymbol{W}^{(1)}, \ldots, \boldsymbol{W}^{(k)})$ of $\boldsymbol{W}$ and compute the associated arithmetic mean:

$$\boldsymbol{U}^{(k)} = \frac{1}{k} \sum_{l=1}^{k} \boldsymbol{W}^{(l)} \, .$$

By the Strong Law of Large Numbers (SLLN), the sequence of random variables $\{\boldsymbol{U}^{(k)}\}_{k \in \mathbb{N}}$ almost surely converges to $\mathbb{E}(\boldsymbol{W})$.

Example: Estimation of an Expectation (2)

A straightforward computation leads to

$$\boldsymbol{U}^{(k+1)} = \boldsymbol{U}^{(k)} - \frac{1}{k+1}\left(\boldsymbol{U}^{(k)} - \boldsymbol{W}^{(k+1)}\right) .$$

Using the notations $c^{(k)} = 1/(k+1)$ and $j(u,w) = (u-w)^2/2$, the last expression of $\boldsymbol{U}^{(k+1)}$ writes

$$\boldsymbol{U}^{(k+1)} = \boldsymbol{U}^{(k)} - c^{(k)} \nabla_u j\left(\boldsymbol{U}^{(k)}, \boldsymbol{W}^{(k+1)}\right) ,$$

which corresponds to the stochastic gradient algorithm applied to:[4]

$$\min_{u \in \mathbb{R}} \frac{1}{2}\mathbb{E}\left((u-\boldsymbol{W})^2\right) .$$

[4] Recall that $\mathbb{E}(\boldsymbol{W})$ is the point which minimizes the dispersion of $\boldsymbol{W}$.

Example: Estimation of an Expectation (2)

A straightforward computation leads to

$$\boldsymbol{U}^{(k+1)} = \boldsymbol{U}^{(k)} - \frac{1}{k+1}\left(\boldsymbol{U}^{(k)} - \boldsymbol{W}^{(k+1)}\right) .$$

Using the notations $\epsilon^{(k)} = 1/(k+1)$ and $j(u,w) = (u-w)^2/2$, the last expression of $\boldsymbol{U}^{(k+1)}$ writes

$$\boldsymbol{U}^{(k+1)} = \boldsymbol{U}^{(k)} - \epsilon^{(k)} \nabla_u j(\boldsymbol{U}^{(k)}, \boldsymbol{W}^{(k+1)}) ,$$

which corresponds to the stochastic gradient algorithm applied to:[4]

$$\min_{u \in \mathbb{R}} \frac{1}{2}\mathbb{E}\left((u - \boldsymbol{W})^2\right) .$$

[4] Recall that $\mathbb{E}(\boldsymbol{W})$ is the point which minimizes the dispersion of $\boldsymbol{W}$.

Example: Estimation of an Expectation (3)

This example makes it possible to enlighten some features of the stochastic gradient method.

Example: Estimation of an Expectation (3)

This example makes it possible to enlighten some features of the stochastic gradient method.

- The step size $\epsilon^{(k)} = 1/(k+1)$ goes to zero as k goes to $+\infty$. Note however that $\epsilon^{(k)}$ goes to zero "not too fast", that is,

$$\sum_{k \in \mathbb{N}} \epsilon^{(k)} = +\infty \, .$$

- It is reasonable to expect an almost sure convergence result for the stochastic gradient algorithm (rather than a weaker notion as convergence in distribution or convergence in probability).
- As the Central Limit Theorem (CLT) applies to this case, we may expect a similar result for the rate of convergence of the stochastic gradient algorithm.

Example: Estimation of an Expectation (3)

This example makes it possible to enlighten some features of the stochastic gradient method.

- The step size $\epsilon^{(k)} = 1/(k+1)$ goes to zero as k goes to $+\infty$. Note however that $\epsilon^{(k)}$ goes to zero "not too fast", that is,

$$\sum_{k \in \mathbb{N}} \epsilon^{(k)} = +\infty \, .$$

- It is reasonable to expect an almost sure convergence result for the stochastic gradient algorithm (rather than a weaker notion as convergence in distribution or convergence in probability).
- As the Central Limit Theorem (CLT) applies to this case, we may expect a similar result for the rate of convergence of the stochastic gradient algorithm.

Example: Estimation of an Expectation (3)

This example makes it possible to enlighten some features of the stochastic gradient method.

- The step size $\epsilon^{(k)} = 1/(k+1)$ goes to zero as k goes to $+\infty$. Note however that $\epsilon^{(k)}$ goes to zero "not too fast", that is,

$$\sum_{k \in \mathbb{N}} \epsilon^{(k)} = +\infty \, .$$

- It is reasonable to expect an almost sure convergence result for the stochastic gradient algorithm (rather than a weaker notion as convergence in distribution or convergence in probability).
- As the Central Limit Theorem (CLT) applies to this case, we may expect a similar result for the rate of convergence of the stochastic gradient algorithm.

1 Stochastic Gradient Algorithm

2 Connexion with Stochastic Approximation

3 Asymptotic Efficiency and Averaging

4 Practical Considerations

Stochastic Approximation (SA) Framework

A classical problem in Stochastic Approximation is to determine the zero of a function $h : \mathbb{U} \to \mathbb{U}$, with $\mathbb{U} = \mathbb{R}^n$, in case that the observation of $h(u)$ is perturbed by an additive random variable $\boldsymbol{\xi}$.

Given a random process $\{\boldsymbol{\xi}^{(k)}\}_{k\in\mathbb{N}}$ and a filtration $\{\mathcal{F}^{(k)}\}_{k\in\mathbb{N}}$, the standard SA algorithm consists in the following iteration:

$$\boldsymbol{U}^{(k+1)} = \boldsymbol{U}^{(k)} + \epsilon^{(k)} \left(h(\boldsymbol{U}^{(k)}) + \boldsymbol{\xi}^{(k+1)} \right) ,$$

Link with the stochastic gradient algorithm:

$$h(u) = -\nabla J(u) ,$$
$$\xi^{(k+1)} = \nabla J(\boldsymbol{U}^{(k)}) - \nabla_u j(\boldsymbol{U}^{(k)}, \boldsymbol{W}^{(k+1)}) .$$

$\rightsquigarrow$ Finding $u^\sharp$ s.t. $h(u^\sharp) = 0$ is equivalent to solving $\nabla J(u^\sharp) = 0$.

Stochastic Approximation (SA) Framework

A classical problem in Stochastic Approximation is to determine the zero of a function $h : \mathbb{U} \to \mathbb{U}$, with $\mathbb{U} = \mathbb{R}^n$, in case that the observation of $h(u)$ is perturbed by an additive random variable $\boldsymbol{\xi}$.

Given a random process $\{\boldsymbol{\xi}^{(k)}\}_{k\in\mathbb{N}}$ and a filtration $\{\mathcal{F}^{(k)}\}_{k\in\mathbb{N}}$, the standard SA algorithm consists in the following iteration:

$$\boldsymbol{U}^{(k+1)} = \boldsymbol{U}^{(k)} + \epsilon^{(k)} \left(h(\boldsymbol{U}^{(k)}) + \boldsymbol{\xi}^{(k+1)} \right) ,$$

Link with the stochastic gradient algorithm:

$$\begin{aligned} h(u) &= -\nabla J(u) , \\ \boldsymbol{\xi}^{(k+1)} &= \nabla J(\boldsymbol{U}^{(k)}) - \nabla_u j(\boldsymbol{U}^{(k)}, \boldsymbol{W}^{(k+1)}) . \end{aligned}$$

$\rightsquigarrow$ Finding $u^\sharp$ s.t. $h(u^\sharp) = 0$ is equivalent to solving $\nabla J(u^\sharp) = 0$.

Convergence Theorem (SA) (1)

Assumptions

1. The random variable $\boldsymbol{U}^{(0)}$ is $\mathcal{F}^{(0)}$-mesurable.
2. The mapping $h : \mathbb{U} \longrightarrow \mathbb{U}$ is continuous, such that
   - $\exists\, u^\sharp \in \mathbb{R}^n,\ h(u^\sharp) = 0$ and $\left\langle h(u)\,, u - u^\sharp \right\rangle < 0,\ \forall u \neq u^\sharp$;
   - $\exists\, a > 0,\ \forall u \in \mathbb{R}^n,\ \|h(u)\|^2 \leq a\big(1 + \|u\|^2\big)$.
3. The random variable $\boldsymbol{\xi}^{(k)}$ is $\mathcal{F}^{(k)}$-mesurable for all k, and
   - $\mathbb{E}\big(\boldsymbol{\xi}^{(k+1)} \,\big|\, \mathcal{F}^{(k)}\big) = 0$,
   - $\exists\, d > 0,\ \mathbb{E}\big(\|\boldsymbol{\xi}^{(k+1)}\|^2 \,\big|\, \mathcal{F}^{(k)}\big) \leq d\big(1 + \|\boldsymbol{U}^{(k)}\|^2\big)$.
4. The sequence $\{\epsilon^{(k)}\}_{k\in\mathbb{N}}$ is a σ-sequence, that is,

$$\sum_{k\in\mathbb{N}} \epsilon^{(k)} = +\infty\,,\ \sum_{k\in\mathbb{N}} \big(\epsilon^{(k)}\big)^2 < +\infty\,.$$

Convergence Theorem (SA) (2)

Robbins-Monro Theorem

Under the previous assumptions, the sequence $\{\boldsymbol{U}^{(k)}\}_{k\in\mathbb{N}}$ of random variables generated by the Stochastic Approximation algorithm almost surely converges to the solution $u^\sharp$ of $h(u) = 0$.

For a proof, see [Duflo, 1997, §1.4].

This theorem can be extended to more general situations.

- A projection operator can be added:

$$\boldsymbol{U}^{(k+1)} = \mathrm{proj}_{U^{\mathrm{ad}}}\left(\boldsymbol{U}^{(k)} + \epsilon^{(k)}\left(h(\boldsymbol{U}^{(k)}) + \xi^{(k+1)}\right)\right).$$

- A "small" additional term $R^{(k+1)}$ can be added:[5]

$$\boldsymbol{U}^{(k+1)} = \boldsymbol{U}^{(k)} + \epsilon^{(k)}\left(h(\boldsymbol{U}^{(k)}) + \xi^{(k+1)} + R^{(k+1)}\right).$$

[5] for example a bias on $h(u)$, as considered in the Kiefer-Wolfowitz algorithm

Convergence Theorem (SA) (2)

Robbins-Monro Theorem

Under the previous assumptions, the sequence $\{\boldsymbol{U}^{(k)}\}_{k\in\mathbb{N}}$ of random variables generated by the Stochastic Approximation algorithm almost surely converges to the solution $u^\sharp$ of $h(u) = 0$.

For a proof, see [Duflo, 1997, §1.4].

This theorem can be extended to more general situations.

- A projection operator can be added:

$$\boldsymbol{U}^{(k+1)} = \mathrm{proj}_{U^{\mathrm{ad}}}\left(\boldsymbol{U}^{(k)} + \epsilon^{(k)}\left(h(\boldsymbol{U}^{(k)}) + \boldsymbol{\xi}^{(k+1)}\right)\right) .$$

- A "small" additional term $\boldsymbol{R}^{(k+1)}$ can be added:[5]

$$\boldsymbol{U}^{(k+1)} = \boldsymbol{U}^{(k)} + \epsilon^{(k)}\left(h(\boldsymbol{U}^{(k)}) + \boldsymbol{\xi}^{(k+1)} + \boldsymbol{R}^{(k+1)}\right) .$$

[5] for example a bias on $h(u)$, as considered in the Kiefer-Wolfowitz algorithm

Rate of Convergence (SA) (1)

We recall a result about the asymptotic normality of the sequence $\{\boldsymbol{U}^{(k)}\}$ generated by the SA algorithm, that is, an estimation of its rate of convergence.

We first need to be more specific about the notion of σ-sequence.

Definition

A positive real sequence $\{\epsilon^{(k)}\}_{k\in\mathbb{N}}$ is a $\sigma(\alpha,\beta,\gamma)$-sequence if it is such that

$$\epsilon^{(k)} = \frac{\alpha}{k^\gamma + \beta},$$

with $\alpha > 0$, $\beta \geq 0$ and $1/2 < \gamma \leq 1$.

A consequence of this definition is that a $\sigma(\alpha,\beta,\gamma)$-sequence is also a σ-sequence.

Rate of Convergence (SA) (1)

We recall a result about the asymptotic normality of the sequence $\{\boldsymbol{U}^{(k)}\}$ generated by the SA algorithm, that is, an estimation of its rate of convergence.

We first need to be more specific about the notion of σ-sequence.

Definition

A positive real sequence $\{\epsilon^{(k)}\}_{k\in\mathbb{N}}$ is a $\sigma(\alpha,\beta,\gamma)$-sequence if it is such that

$$\epsilon^{(k)} = \frac{\alpha}{k^\gamma + \beta}\,,$$

with $\alpha > 0$, $\beta \geq 0$ and $1/2 < \gamma \leq 1$.

A consequence of this definition is that a $\sigma(\alpha,\beta,\gamma)$-sequence is also a σ-sequence.

Rate of Convergence (SA) (2)

Assumptions

1. h is continuously differentiable and, in a neighborhood of $u^\sharp$,
$$h(u) = -H(u - u^\sharp) + \mathrm{O}\big(\|u - u^\sharp\|^2\big) \,,$$
where H is a symmetric positive-definite matrix.
2. The sequence $\big\{\mathbb{E}\big(\boldsymbol{\xi}^{(k+1)}(\boldsymbol{\xi}^{(k+1)})^\top \mid \mathcal{F}^{(k)}\big)\big\}_{k\in\mathbb{N}}$ almost surely converges to a symmetric positive-definite matrix Γ.
3. $\exists\, \delta > 0$ such that $\sup_{k\in\mathbb{N}} \mathbb{E}\big(\|\boldsymbol{\xi}^{(k+1)}\|^{2+\delta} \mid \mathcal{F}^{(k)}\big) < +\infty$.
4. The sequence $\{\epsilon^{(k)}\}_{k\in\mathbb{N}}$ is a $\sigma(\alpha,\beta,\gamma)$-sequence.
5. The square matrix $(H - \lambda I)$ is positive-definite, with
$$\lambda = \begin{cases} 0 & \text{if } \gamma < 1 \,, \\ \dfrac{1}{2\alpha} & \text{if } \gamma = 1 \,. \end{cases}$$

Rate of Convergence (SA) (3)

We retain the assumptions ensuring the almost sure convergence.

Central Limit Theorem

Under all previous assumptions, the sequence of random variables $\left\{(1/\sqrt{\epsilon^{(k)}})(\boldsymbol{U}^{(k)} - u^\sharp)\right\}_{k\in\mathbb{N}}$ converges in law towards a centered gaussian distribution with covariance matrix Σ, that is,

$$\frac{1}{\sqrt{\epsilon^{(k)}}}\left(\boldsymbol{U}^{(k)} - u^\sharp\right) \xrightarrow{\mathcal{D}} \mathcal{N}\left(0, \Sigma\right) ,$$

in which Σ is the solution of the so-called Lyapunov equation

$$\left(H - \lambda I\right)\Sigma + \Sigma\left(H - \lambda I\right) = \Gamma .$$

For a proof, see [Duflo, 1996, Chapter 4].

Rate of Convergence (SA) (4)

- The result is valid only for unconstrained problems: $U^{\mathrm{ad}} = \mathbb{U}$.
- The result can be rephrased as

$$k^{\frac{\gamma}{2}} \left(\boldsymbol{U}^{(k)} - u^\sharp \right) \xrightarrow{\mathcal{D}} \mathcal{N}(0, a\Sigma) \,,$$

  so that β has in fact no influence on the convergence rate.
- The choice $\gamma = 1$ achieves the greatest convergence rate. We recover the rate $1/\sqrt{k}$ of a standard Monte Carlo estimator.
- If we refer back to the optimization problem $(\mathcal{P}_{\mathrm{ol}})$, that is, $h = -\nabla J$, we notice that H is the Hessian matrix of J at $u^\sharp$:

$$H = \nabla^2 J(u^\sharp) \,,$$

  and that Γ is the covariance matrix of $\nabla_u j$ evaluated at $u^\sharp$:

$$\Gamma = \mathbb{E}\left(\nabla_u j(u^\sharp, \boldsymbol{W}) (\nabla_u j(u^\sharp, \boldsymbol{W}))^{\mathrm{T}} \right) .$$

Rate of Convergence (SA) (4)

- The result is valid only for unconstrained problems: $U^{\mathrm{ad}} = \mathbb{U}$.
- The result can be rephrased as

$$k^{\frac{\gamma}{2}} \left(\boldsymbol{U}^{(k)} - u^\sharp \right) \xrightarrow{\mathcal{D}} \mathcal{N}\left(0, \alpha \Sigma\right) ,$$

  so that β has in fact no influence on the convergence rate.

- The choice $\gamma = 1$ achieves the greatest convergence rate. We recover the rate $1/\sqrt{k}$ of a standard Monte Carlo estimator.
- If we refer back to the optimization problem $(\mathcal{P}_{\mathrm{ol}})$, that is, $h = -\nabla J$, we notice that H is the Hessian matrix of J at $u^\sharp$:

$$H = \nabla^2 J(u^\sharp) ,$$

  and that Γ is the covariance matrix of $\nabla_u j$ evaluated at $u^\sharp$:

$$\Gamma = \mathbb{E}\left(\nabla_u j(u^\sharp, \boldsymbol{W}) (\nabla_u j(u^\sharp, \boldsymbol{W}))^{\mathrm{T}} \right) .$$

Rate of Convergence (SA) (4)

- The result is valid only for unconstrained problems: $U^{\mathrm{ad}} = \mathbb{U}$.
- The result can be rephrased as

$$k^{\frac{\gamma}{2}} \left(\boldsymbol{U}^{(k)} - u^\sharp \right) \xrightarrow{\mathcal{D}} \mathcal{N}\left(0, \alpha \Sigma\right) ,$$

  so that β has in fact no influence on the convergence rate.
- The choice $\gamma = 1$ achieves the greatest convergence rate. We recover the rate $1/\sqrt{k}$ of a standard Monte Carlo estimator.
- If we refer back to the optimization problem $(\mathcal{P}_{\mathrm{ol}})$, that is, $h = -\nabla J$, we notice that H is the Hessian matrix of J at $u^\sharp$:

$$H = \nabla^2 J(u^\sharp) ,$$

  and that Γ is the covariance matrix of $\nabla_u j$ evaluated at $u^\sharp$:

$$\Gamma = \mathbb{E}\left(\nabla_u j(u^\sharp, \boldsymbol{W}) (\nabla_u j(u^\sharp, \boldsymbol{W}))^{\mathrm{T}} \right) .$$

Rate of Convergence (SA) (4)

- The result is valid only for unconstrained problems: $U^{\mathrm{ad}} = \mathbb{U}$.
- The result can be rephrased as

$$k^{\frac{\gamma}{2}} \left(\boldsymbol{U}^{(k)} - u^\sharp \right) \xrightarrow{\mathcal{D}} \mathcal{N}\left(0, \alpha \Sigma\right) ,$$

  so that β has in fact no influence on the convergence rate.
- The choice $\gamma = 1$ achieves the greatest convergence rate. We recover the rate $1/\sqrt{k}$ of a standard Monte Carlo estimator.
- If we refer back to the optimization problem $(\mathcal{P}_{\mathrm{ol}})$, that is, $h = -\nabla J$, we notice that H is the Hessian matrix of J at $u^\sharp$:

$$H = \nabla^2 J(u^\sharp) ,$$

  and that Γ is the covariance matrix of $\nabla_u j$ evaluated at $u^\sharp$:

$$\Gamma = \mathbb{E}\left(\nabla_u j(u^\sharp, \boldsymbol{W}) \left(\nabla_u j(u^\sharp, \boldsymbol{W})\right)^\top \right) .$$

1 Stochastic Gradient Algorithm

2 Connexion with Stochastic Approximation

3 Asymptotic Efficiency and Averaging

4 Practical Considerations

Stochastic Newton Algorithm (1)

Here, the step sizes $\epsilon^{(k)}$ are built using the optimal choice $\gamma = 1$. The *scalar* gain α is replaced by a matrix gain A, where A is a symmetric positive-definite matrix. The **SG** algorithm becomes

$$\boldsymbol{U}^{(k+1)} = \boldsymbol{U}^{(k)} - \frac{1}{k+\beta} A \nabla_u J(\boldsymbol{U}^{(k)}, \boldsymbol{W}^{(k+1)}) ,$$

which in the Stochastic Approximation setting writes

$$\boldsymbol{U}^{(k+1)} = \boldsymbol{U}^{(k)} + \frac{1}{k+\beta} \Big(A\, h(\boldsymbol{U}^{(k)}) + A\, \boldsymbol{\xi}^{(k+1)} \Big) .$$

The Central Limit Theorem is thus available, and we have

$$\sqrt{k} \Big(\boldsymbol{U}^{(k)} - u^\sharp \Big) \xrightarrow{\mathcal{D}} \mathcal{N}\big(0, \Sigma_A\big) ,$$

where Σ_A is the unique solution of

$$\Big(AH - \frac{I}{2} \Big) \Sigma_A + \Sigma_A \Big(HA - \frac{I}{2} \Big) = A \Gamma A .$$

Stochastic Newton Algorithm (1)

Here, the step sizes $\epsilon^{(k)}$ are built using the optimal choice $\gamma = 1$. The *scalar* gain α is replaced by a matrix gain A, where A is a symmetric positive-definite matrix. The **SG** algorithm becomes

$$\boldsymbol{U}^{(k+1)} = \boldsymbol{U}^{(k)} - \frac{1}{k+\beta} A \, \nabla_u j(\boldsymbol{U}^{(k)}, \boldsymbol{W}^{(k+1)}) \, ,$$

which in the Stochastic Approximation setting writes

$$\boldsymbol{U}^{(k+1)} = \boldsymbol{U}^{(k)} + \frac{1}{k+\beta} \Big(A \, h(\boldsymbol{U}^{(k)}) + A \, \boldsymbol{\xi}^{(k+1)} \Big) \, .$$

The Central Limit Theorem is thus available, and we have

$$\sqrt{k} \Big(\boldsymbol{U}^{(k)} - u^\sharp \Big) \xrightarrow{\mathcal{D}} \mathcal{N}\big(0, \Sigma_A\big) \, ,$$

where Σ_A is the unique solution of

$$\Big(AH - \frac{I}{2} \Big) \Sigma_A + \Sigma_A \Big(HA - \frac{I}{2} \Big) = A \Gamma A \, .$$

Stochastic Newton Algorithm (1)

Here, the step sizes $\epsilon^{(k)}$ are built using the optimal choice $\gamma = 1$. The *scalar* gain α is replaced by a matrix gain A, where A is a symmetric positive-definite matrix. The **SG** algorithm becomes

$$\boldsymbol{U}^{(k+1)} = \boldsymbol{U}^{(k)} - \frac{1}{k+\beta} A \, \nabla_u j(\boldsymbol{U}^{(k)}, \boldsymbol{W}^{(k+1)}) \, ,$$

which in the Stochastic Approximation setting writes

$$\boldsymbol{U}^{(k+1)} = \boldsymbol{U}^{(k)} + \frac{1}{k+\beta} \Big(A \, h(\boldsymbol{U}^{(k)}) + A \, \boldsymbol{\xi}^{(k+1)} \Big) \, .$$

The Central Limit Theorem is thus available, and we have

$$\sqrt{k} \Big(\boldsymbol{U}^{(k)} - u^\sharp \Big) \xrightarrow{\mathcal{D}} \mathcal{N}\big(0, \Sigma_A\big) \, ,$$

where Σ_A is the unique solution of

$$\Big(AH - \frac{I}{2} \Big) \Sigma_A + \Sigma_A \Big(HA - \frac{I}{2} \Big) = A \Gamma A \, .$$

Stochastic Newton Algorithm (2)

Let $\mathcal{C}_H$ be the set of symmetric positive-definite matrices A, such that $AH - I/2$ is a positive-definite matrix.

Theorem

The choice $A^\sharp = H^{-1}$ for the matrix A minimizes the asymptotic covariance matrix Σ_A over the set $\mathcal{C}_H$. The expression of the minimal asymptotic covariance matrix is

$$\Sigma_{A^\sharp} = H^{-1} \Gamma H^{-1} \, .$$

Sketch of proof. Rewrite the covariance matrix Σ_A as $\Delta_A + H^{-1} \Gamma H^{-1}$. Then the matrix Δ_A satisfies a Lyapunov equation, whose solution is thus semi-definite positive, hence the result. □

Stochastic Newton Algorithm (3)

Definition

A stochastic gradient algorithm is Newton-efficient if the sequence $\{\boldsymbol{U}^{(k)}\}_{k\in\mathbb{N}}$ it generates has the same asymptotic convergence rate as the optimal Newton algorithm, namely

$$\sqrt{k}\left(\boldsymbol{U}^{(k)} - u^\sharp\right) \xrightarrow{\mathcal{D}} \mathcal{N}\left(0, H^{-1}\Gamma H^{-1}\right) .$$

Note that the improvement is on the covariance matrix of the Gaussian distribution. The rate of convergence remains $1/\sqrt{k}$.

Question. How to obtain an implementable Newton-efficient stochastic gradient algorithm?

Stochastic Gradient Algorithm with Averaging (1)

[Polyak, 1992] proposed to implement a Newton-efficient algorithm by incorporating a new averaging stage in the standard algorithm.

Assuming that the admissible set U^{ad} is equal to the space $\mathbb{U}$, the standard stochastic gradient algorithm iteration

$$\boldsymbol{u}^{(k+1)} = \boldsymbol{u}^{(k)} - \epsilon^{(k)} \nabla_u j(\boldsymbol{u}^{(k)}, \boldsymbol{w}^{(k+1)}) ,$$

is replaced by

$$\boldsymbol{u}^{(k+1)} = \boldsymbol{u}^{(k)} - \epsilon^{(k)} \nabla_u j(\boldsymbol{u}^{(k)}, \boldsymbol{w}^{(k+1)}) ,$$

$$\boldsymbol{u}_{\mathrm{M}}^{(k+1)} = \frac{1}{k+1} \sum_{l=1}^{k+1} \boldsymbol{u}^{(l)} .$$

Note that an equivalent recursive form for the averaging stage is

$$\boldsymbol{u}_{\mathrm{M}}^{(k+1)} = \boldsymbol{u}_{\mathrm{M}}^{(k)} + \frac{1}{k+1} (\boldsymbol{u}^{(k+1)} - \boldsymbol{u}_{\mathrm{M}}^{(k)}) .$$

Stochastic Gradient Algorithm with Averaging (1)

[Polyak, 1992] proposed to implement a Newton-efficient algorithm by incorporating a new averaging stage in the standard algorithm. Assuming that the admissible set U^{ad} is equal to the space $\mathbb{U}$, the standard stochastic gradient algorithm iteration

$$\boldsymbol{U}^{(k+1)} = \boldsymbol{U}^{(k)} - \epsilon^{(k)} \nabla_u j(\boldsymbol{U}^{(k)}, \boldsymbol{W}^{(k+1)}) \,,$$

is replaced by

$$\boldsymbol{U}^{(k+1)} = \boldsymbol{U}^{(k)} - \epsilon^{(k)} \nabla_u j(\boldsymbol{U}^{(k)}, \boldsymbol{W}^{(k+1)}) \,,$$

$$\boldsymbol{U}_{\mathrm{M}}^{(k+1)} = \frac{1}{k+1} \sum_{l=1}^{k+1} \boldsymbol{U}^{(l)} \,.$$

Note that an equivalent recursive form for the averaging stage is

$$\boldsymbol{U}_{\mathrm{M}}^{(k+1)} = \boldsymbol{U}_{\mathrm{M}}^{(k)} + \frac{1}{k+1} (\boldsymbol{U}^{(k+1)} - \boldsymbol{U}_{\mathrm{M}}^{(k)}) \,.$$

Stochastic Gradient Algorithm with Averaging (1)

[Polyak, 1992] proposed to implement a Newton-efficient algorithm by incorporating a new averaging stage in the standard algorithm. Assuming that the admissible set U^{ad} is equal to the space $\mathbb{U}$, the standard stochastic gradient algorithm iteration

$$\boldsymbol{U}^{(k+1)} = \boldsymbol{U}^{(k)} - \epsilon^{(k)} \nabla_u j(\boldsymbol{U}^{(k)}, \boldsymbol{W}^{(k+1)}) \,,$$

is replaced by

$$\boldsymbol{U}^{(k+1)} = \boldsymbol{U}^{(k)} - \epsilon^{(k)} \nabla_u j(\boldsymbol{U}^{(k)}, \boldsymbol{W}^{(k+1)}) \,,$$

$$\boldsymbol{U}_{\mathrm{M}}^{(k+1)} = \frac{1}{k+1} \sum_{l=1}^{k+1} \boldsymbol{U}^{(l)} \,.$$

Note that an equivalent recursive form for the averaging stage is

$$\boldsymbol{U}_{\mathrm{M}}^{(k+1)} = \boldsymbol{U}_{\mathrm{M}}^{(k)} + \frac{1}{k+1} (\boldsymbol{U}^{(k+1)} - \boldsymbol{U}_{\mathrm{M}}^{(k)}) \,.$$

Stochastic Gradient Algorithm with Averaging (1)

[Polyak, 1992] proposed to implement a Newton-efficient algorithm by incorporating a new averaging stage in the standard algorithm. Assuming that the admissible set U^{ad} is equal to the space $\mathbb{U}$, the standard stochastic gradient algorithm iteration

$$\boldsymbol{U}^{(k+1)} = \boldsymbol{U}^{(k)} - \epsilon^{(k)} \nabla_u j(\boldsymbol{U}^{(k)}, \boldsymbol{W}^{(k+1)}) \,,$$

is replaced by

$$\boldsymbol{U}^{(k+1)} = \boldsymbol{U}^{(k)} - \epsilon^{(k)} \nabla_u j(\boldsymbol{U}^{(k)}, \boldsymbol{W}^{(k+1)}) \,,$$

$$\boldsymbol{U}_{\mathrm{M}}^{(k+1)} = \frac{1}{k+1} \sum_{l=1}^{k+1} \boldsymbol{U}^{(l)} \,.$$

Note that an equivalent recursive form for the averaging stage is

$$\boldsymbol{U}_{\mathrm{M}}^{(k+1)} = \boldsymbol{U}_{\mathrm{M}}^{(k)} + \frac{1}{k+1} \left(\boldsymbol{U}^{(k+1)} - \boldsymbol{U}_{\mathrm{M}}^{(k)} \right) .$$

Stochastic Gradient Algorithm with Averaging (2)

Theorem

Under the additional assumption that the $\sigma(\alpha, \beta, \gamma)$-sequence $\{\epsilon^{(k)}\}_{k\in\mathbb{N}}$ is such that $\gamma < 1$, the averaged stochastic gradient algorithm is Newton-efficient:

$$\sqrt{k}\left(\boldsymbol{U}_{\mathrm{M}}^{(k)} - u^{\sharp}\right) \xrightarrow{\mathcal{D}} \mathcal{N}\left(0, H^{-1}\Gamma H^{-1}\right) .$$

For a proof, see [Duflo, 1996, Chapter 4].

According to the standard theorem, the convergence rate achieved by the sequence $\{\boldsymbol{U}^{(k)}\}_{k\in\mathbb{N}}$ with $\gamma < 1$ is smaller than $1/\sqrt{k}$ and hence not optimal. The "nice" convergence properties are obtained regarding the averaged sequence $\{\boldsymbol{U}_{\mathrm{M}}^{(k)}\}_{k\in\mathbb{N}}$.

Stochastic Gradient Algorithm with Averaging (2)

Theorem

Under the additional assumption that the $\sigma(\alpha, \beta, \gamma)$-sequence $\{\epsilon^{(k)}\}_{k\in\mathbb{N}}$ is such that $\boldsymbol{\gamma < 1}$, the averaged stochastic gradient algorithm is Newton-efficient:

$$\sqrt{k}\left(\boldsymbol{U}_{\mathrm{M}}^{(k)} - u^{\sharp}\right) \xrightarrow{\mathcal{D}} \mathcal{N}\left(0, H^{-1}\Gamma H^{-1}\right) .$$

For a proof, see [Duflo, 1996, Chapter 4].

According to the standard theorem, the convergence rate achieved by the sequence $\{\boldsymbol{U}^{(k)}\}_{k\in\mathbb{N}}$ with $\gamma < 1$ is smaller than $1/\sqrt{k}$ and hence not optimal. The "nice" convergence properties are obtained regarding the averaged sequence $\{\boldsymbol{U}_{\mathrm{M}}^{(k)}\}_{k\in\mathbb{N}}$.

1 Stochastic Gradient Algorithm

2 Connexion with Stochastic Approximation

3 Asymptotic Efficiency and Averaging

4 Practical Considerations

A Toy Problem

Let us consider the following optimization problem:

$$\min_{u \in \mathbb{R}^{10}} \mathbb{E}\left(\frac{1}{2} u^\top B u + \boldsymbol{W}^\top u\right),$$

B being a symmetric positive definite matrix, and $\boldsymbol{W}$ being a $\mathbb{R}^{10}$-valued Gaussian random variable $\mathcal{N}(m, \Gamma)$.

The optimal solution of this problem is $u^\sharp = -B^{-1} m$.

It can be estimated either by Monte Carlo

$$\widehat{U}^{(k+1)} = -\frac{1}{k+1} \sum_{l=1}^{k+1} B^{-1} W^{(l)},$$

or by the standard stochastic gradient algorithm

$$U^{(k+1)} = U^{(k)} - c^{(k)} \left(B U^{(k)} + W^{(k+1)}\right).$$

A Toy Problem

Let us consider the following optimization problem:

$$\min_{u \in \mathbb{R}^{10}} \mathbb{E}\left(\frac{1}{2} u^\top B u + \boldsymbol{W}^\top u\right),$$

B being a symmetric positive definite matrix, and $\boldsymbol{W}$ being a $\mathbb{R}^{10}$-valued Gaussian random variable $\mathcal{N}(m, \Gamma)$.

The optimal solution of this problem is $u^\sharp = -B^{-1} m$.

It can be estimated either by Monte Carlo

$$\widehat{U}^{(k+1)} = -\frac{1}{k+1} \sum_{l=1}^{k+1} B^{-1} W^{(l)},$$

or by the standard stochastic gradient algorithm

$$U^{(k+1)} = U^{(k)} - c^{(k)} \left(B U^{(k)} + W^{(k+1)}\right).$$

A Toy Problem

Let us consider the following optimization problem:

$$\min_{u \in \mathbb{R}^{10}} \mathbb{E}\left(\frac{1}{2} u^\top B u + \boldsymbol{W}^\top u\right),$$

B being a symmetric positive definite matrix, and $\boldsymbol{W}$ being a $\mathbb{R}^{10}$-valued Gaussian random variable $\mathcal{N}(m, \Gamma)$.

The optimal solution of this problem is $u^\sharp = -B^{-1} m$.

It can be estimated either by Monte Carlo

$$\widehat{\boldsymbol{U}}^{(k+1)} = -\frac{1}{k+1} \sum_{l=1}^{k+1} B^{-1} \boldsymbol{W}^{(l)},$$

or by the standard stochastic gradient algorithm

$$\boldsymbol{U}^{(k+1)} = \boldsymbol{U}^{(k)} - \epsilon^{(k)} \left(B \boldsymbol{U}^{(k)} + \boldsymbol{W}^{(k+1)}\right).$$

Tuning the Standard Algorithm (1)

Let $\{\epsilon^{(k)}\}_{k\in\mathbb{N}}$ be a $\sigma(\alpha,\beta,\gamma)$-sequence, that is, $\epsilon^{(k)} = \dfrac{\alpha}{k^\gamma + \beta}$.

Tuning the Standard Algorithm (1)

Let $\{\epsilon^{(k)}\}_{k\in\mathbb{N}}$ be a $\sigma(\alpha,\beta,\gamma)$-sequence, that is, $\epsilon^{(k)} = \dfrac{\alpha}{k^\gamma + \beta}$.

- The best convergence rate is reached for $\gamma = 1$.
- The coefficient α influences the asymptotic behavior. The covariance matrix $\alpha\Sigma$ grows as α goes to $+\infty$,[6] but using too small values for α may generate very small gradient steps. The choice of α corresponds to a trade-off between stability and precision.
- Ultimately, the coefficient β makes it possible to regulate the transient behavior of the algorithm. During the first iterations, the coefficient $\epsilon^{(k)}$ is approximately equal to α/β. If α/β is too small, the transient phase may be slow. On the contrary, taking a too large ratio may lead to a numerical burst.

[6] remember that Σ depends on α. . .

Tuning the Standard Algorithm (1)

Let $\{\epsilon^{(k)}\}_{k\in\mathbb{N}}$ be a $\sigma(\alpha,\beta,\gamma)$-sequence, that is, $\epsilon^{(k)} = \dfrac{\alpha}{k^\gamma + \beta}$.

- The best convergence rate is reached for $\gamma = 1$.
- The coefficient α influences the asymptotic behavior. The covariance matrix $\alpha\Sigma$ grows as α goes to $+\infty$,[6] but using too small values for α may generate very small gradient steps. The choice of α corresponds to a trade-off between stability and precision.
- Ultimately, the coefficient β makes it possible to regulate the transient behavior of the algorithm. During the first iterations, the coefficient $\epsilon^{(k)}$ is approximately equal to α/β. If α/β is too small, the transient phase may be slow. On the contrary, taking a too large ratio may lead to a numerical burst.

[6] remember that Σ depends on α...

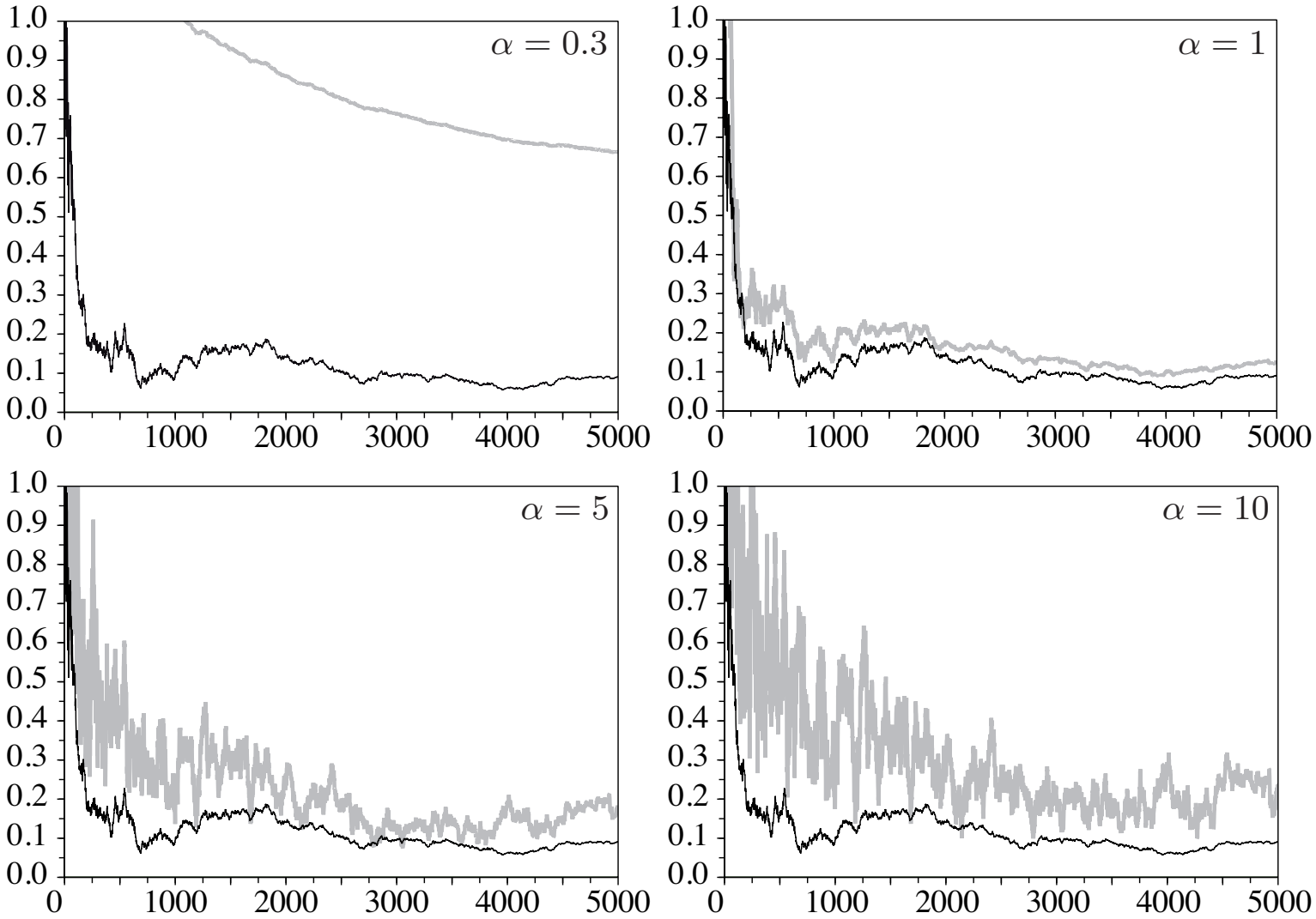
Tuning the Standard Algorithm (1)

Let $\{\epsilon^{(k)}\}_{k\in\mathbb{N}}$ be a $\sigma(\alpha,\beta,\gamma)$-sequence, that is, $\epsilon^{(k)} = \dfrac{\alpha}{k^\gamma+\beta}$.

- The best convergence rate is reached for $\gamma = 1$.
- The coefficient α influences the asymptotic behavior. The covariance matrix $\alpha\Sigma$ grows as α goes to $+\infty$,[6] but using too small values for α may generate very small gradient steps. The choice of α corresponds to a trade-off between stability and precision.
- Ultimately, the coefficient β makes it possible to regulate the transient behavior of the algorithm. During the first iterations, the coefficient $\epsilon^{(k)}$ is approximately equal to α/β. If α/β is too small, the transient phase may be slow. On the contrary, taking a too large ratio may lead to a numerical burst.

[6] remember that Σ depends on α...

Tuning the Standard Algorithm ($\alpha/\beta = 0.1$) (2)



Tuning the Averaged Algorithm (1)

Here, $\{\epsilon^{(k)}\}_{k\in\mathbb{N}}$ is a $\sigma(\alpha,\beta,\gamma)$-sequence, with $1/2 < \gamma < 1$.

The averaged stochastic gradient algorithm writes on our example

$$\boldsymbol{U}^{(k+1)} = \boldsymbol{U}^{(k)} - \frac{\alpha}{k^\gamma + \beta}\left(B\boldsymbol{U}^{(k)} + \boldsymbol{W}^{(k+1)}\right) \quad , \quad \boldsymbol{U}_{\mathrm{M}}^{(k+1)} = \frac{1}{k+1}\sum_{l=1}^{k+1}\boldsymbol{U}^{(l)} \, .$$

Tuning the Averaged Algorithm (1)

Here, $\{\epsilon^{(k)}\}_{k\in\mathbb{N}}$ is a $\sigma(\alpha,\beta,\gamma)$-sequence, with $1/2 < \gamma < 1$.

The averaged stochastic gradient algorithm writes on our example

$$\boldsymbol{U}^{(k+1)} = \boldsymbol{U}^{(k)} - \frac{\alpha}{k^\gamma + \beta}\left(B\boldsymbol{U}^{(k)} + \boldsymbol{W}^{(k+1)}\right) \quad , \quad \boldsymbol{U}_{\mathrm{M}}^{(k+1)} = \frac{1}{k+1}\sum_{l=1}^{k+1} \boldsymbol{U}^{(l)} \, .$$

- The value $\gamma = 2/3$ is considered as a good choice.
- The tuning of parameters α and β is much easier than for the standard algorithm. Indeed, the problem of "too small" step sizes arising from a bad choice of α is not so critical because the term $k^{-\gamma}$ goes down more slowly towards zero. Of course, the ratio α/β must be chosen in such a way that numerical bursts do not occur during the first iterations of the algorithm.

Tuning the Averaged Algorithm (1)

Here, $\{\epsilon^{(k)}\}_{k\in\mathbb{N}}$ is a $\sigma(\alpha,\beta,\gamma)$-sequence, with $1/2<\gamma<1$.

The averaged stochastic gradient algorithm writes on our example

$$\boldsymbol{U}^{(k+1)} = \boldsymbol{U}^{(k)} - \frac{\alpha}{k^\gamma+\beta}\left(B\boldsymbol{U}^{(k)} + \boldsymbol{W}^{(k+1)}\right) \quad , \quad \boldsymbol{U}_{\mathrm{M}}^{(k+1)} = \frac{1}{k+1}\sum_{l=1}^{k+1}\boldsymbol{U}^{(l)} \, .$$

- The value $\gamma = 2/3$ is considered as a good choice.
- The tuning of parameters α and β is much easier than for the standard algorithm. Indeed, the problem of "too small" step sizes arising from a bad choice of α is not so critical because the term $k^{-\gamma}$ goes down more slowly towards zero. Of course, the ratio α/β must be chosen in such a way that numerical bursts do not occur during the first iterations of the algorithm.

Tuning the Averaged Algorithm ($\alpha/\beta = 0.1$) (2)