Stat Comput (2015) 25:781–795
DOI 10.1007/s11222-015-9560-y

Scalable estimation strategies based on stochastic approximations: classical results and new insights

Panos Toulis[1] · **Edoardo M. Airoldi**[1]

Accepted: 8 March 2015 / Published online: 11 June 2015

Abstract Estimation with large amounts of data can be facilitated by stochastic gradient methods, in which model parameters are updated sequentially using small batches of data at each step. Here, we review early work and modern results that illustrate the statistical properties of these methods, including convergence rates, stability, and asymptotic bias and variance. We then overview modern applications where these methods are useful, ranging from an online version of the EM algorithm to deep learning. In light of these results, we argue that stochastic gradient methods are poised to become benchmark principled estimation procedures for large datasets, especially those in the family of stable proximal methods, such as implicit stochastic gradient descent.

Keywords Maximum likelihood · Recursive estimation · Implicit stochastic gradient descent methods · Optimal learning rate · Asymptotic analysis · Big data

1 Introduction

Parameter estimation by optimization of an objective function, such as maximum likelihood and maximum a-posteriori, is a fundamental idea in statistics and machine learning (Fisher 1922; Lehmann and Casella 2003; Hastie et al. 2011). However, widely used optimization-based estimation algorithms, such as Fisher scoring, the EM algorithm, and iteratively reweighted least squares (Fisher 1925a; Dempster et al. 1977; Green 1984), are not scalable to modern datasets with hundreds of millions of data points and hundreds of thousands of covariates (National Research Council 2013).

To illustrate, let us consider the problem of estimating the true vector of parameters $\boldsymbol{\theta}_\star \in \mathbb{R}^p$ from an i.i.d. sample $\boldsymbol{Y} = \{\boldsymbol{y}_n\}$, for $n = 1, 2, \ldots, N$, where a data point $\boldsymbol{y}_n \in \mathbb{R}^d$ is distributed according to a density $f(\boldsymbol{y}_n; \boldsymbol{\theta}_\star)$ with log-likelihood function $\ell(\boldsymbol{\theta}; \boldsymbol{Y}) = \sum_{n=1}^N \log f(\boldsymbol{y}_n; \boldsymbol{\theta})$. Traditional estimation methods are typically *iterative* and have a running-time complexity that ranges between $\mathcal{O}(Np^3)$ and $\mathcal{O}(Np)$, in worst cases and best cases, respectively. Newton–Raphson methods, for instance, update an estimate $\boldsymbol{\theta}_{n-1}^{\text{nr}}$ of the parameters through the recursion

$$\boldsymbol{\theta}_n^{\text{nr}} = \boldsymbol{\theta}_{n-1}^{\text{nr}} - \boldsymbol{H}_{n-1}^{-1} \nabla \ell(\boldsymbol{\theta}_{n-1}^{\text{nr}}; \boldsymbol{Y}), \tag{1}$$

where $\boldsymbol{H}_n = \nabla\nabla\ell(\boldsymbol{\theta}_n^{\text{nr}}; \boldsymbol{Y})$ is the $p \times p$ Hessian matrix of the log-likelihood. The matrix inversion and the likelihood computation yield an algorithm with roughly $\mathcal{O}(Np^{2+\epsilon})$ complexity which makes it unsuitable for large datasets. Fisher scoring replaces the Hessian matrix with its expected value i.e., it uses the Fisher information matrix $\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}) = -\mathbb{E}\left(\nabla\nabla\ell(\boldsymbol{\theta}; \boldsymbol{y}_n)\right)$, where the expectation is over the random sample $\boldsymbol{y}_n$. The advantage of this method is that a steady increase in the likelihood is possible, as in the EM algorithm, since $\boldsymbol{\mathcal{I}}(\boldsymbol{\theta})$ is positive-definite, and thus the difference

$$\ell(\boldsymbol{\theta} + \epsilon\Delta\boldsymbol{\theta}; \boldsymbol{Y}) - \ell(\boldsymbol{\theta}; \boldsymbol{Y}) \approx \epsilon\ell(\boldsymbol{\theta}; \boldsymbol{Y})^\intercal \boldsymbol{\mathcal{I}}(\boldsymbol{\theta})^{-1} \ell(\boldsymbol{\theta}; \boldsymbol{Y}) + \mathcal{O}(\epsilon^2)$$

can be made positive for an appropriately small value $\epsilon > 0$. However, Fisher scoring performs very similarly to Newton–Raphson in practice, and the two algorithms are actually identical in the exponential family (Lange 2010). Furthermore, Fisher scoring is computationally comparable

✉ Edoardo M. Airoldi
airoldi@fas.harvard.edu

Panos Toulis
ptoulis@fas.harvard.edu

[1] Department of Statistics, Harvard University, Cambridge, MA 02138, USA

to Newton–Raphson and thus unsuited for problems with large datasets.

Quasi-Newton (QN) methods are a powerful alternative and are widely used in practice. In QN methods, the Hessian is approximated by a low-rank matrix that is updated at each iteration as new values of the gradient become available, thus yielding algorithms with complexity $\mathcal{O}(Np^2)$ or $\mathcal{O}(Np)$ in certain favorable cases (Hennig and Kiefel 2013). Other general estimation algorithms such as EM or iteratively reweighted least squares (Green 1984) involve computations (e.g., inversions or maximizations between iterations) that are significantly more expensive than QN methods.

However, estimation with massive datasets requires a running-time complexity that is roughly $\mathcal{O}(Np^{1-\epsilon})$ i.e., that is linear in N but sublinear in the parameter dimension p. The first requirement on N seems hard to overcome since an iteration over all data points needs to be performed, at least when data are i.i.d.; thus, sublinearity in p is crucial (Bousquet and Bottou 2008). Such computational requirements have recently sparked interest in algorithms that utilize only *first-order* information i.e., methods that utilize only gradient computations.[1] Such performance is achieved by the *stochastic gradient descent* (SGD) algorithm, which was initially proposed by Sakrison (1965) as a *recursive estimation method*, albeit not in first-order form. A typical first-order SGD is defined by the iteration

$$\boldsymbol{\theta}_n^{\text{sgd}} = \boldsymbol{\theta}_{n-1}^{\text{sgd}} + a_n \nabla \ell(\boldsymbol{\theta}_{n-1}^{\text{sgd}}; \boldsymbol{y}_n). \tag{2}$$

We will refer to Eq. (2) as SGD with *explicit updates*, or *explicit SGD* for short, because the next iterate $\boldsymbol{\theta}_n^{\text{sgd}}$ can be computed immediately after the new data point $\boldsymbol{y}_n$ is observed.[2] The sequence $a_n > 0$ is a carefully chosen *learning rate* sequence which is typically defined such that $na_n \to \alpha > 0$ as $n \to \infty$. The parameter $\alpha > 0$ is the *learning rate parameter*, and it is crucial for the convergence and stability of explicit SGD.

From a computational perspective, the SGD procedure (2) is appealing because the expensive inversion of $p \times p$ matrices, as in Newton–Raphson, is replaced by a single sequence $a_n > 0$. Furthermore, the log-likelihood is evaluated at a single observation $\boldsymbol{y}_n$, and not on the entire dataset $\boldsymbol{Y}$. Necessarily this incurs information loss which is important to quantify. From a theoretical perspective the explicit SGD updates are justified because, under typical regularity conditions, $\mathbb{E}\left(\nabla \ell(\boldsymbol{\theta}_\star; \boldsymbol{y}_n)\right) = 0$ and thus $\boldsymbol{\theta}_n \to \boldsymbol{\theta}_\star$ by the properties of the Robbins–Monro procedure (Robbins and Monro 1951). However, the explicit SGD procedure requires careful tuning of the learning rate parameter; small values of α will make the iteration (2) very slow to converge, whereas for large values of α explicit SGD will either have a large asymptotic variance, or even diverge numerically. As a recursive estimation method, explicit SGD was first proposed by Sakrison (1965) and has attracted attention in the machine learning community as a fast prediction method for large-scale problems (Le et al. 2004; Zhang 2004).

In order to stabilize explicit SGD without sacrificing computational efficiency, Toulis et al. (2014) defined the *implicit SGD* procedure through the iteration

$$\boldsymbol{\theta}_n^{\text{im}} = \boldsymbol{\theta}_{n-1}^{\text{im}} + a_n \nabla \ell(\boldsymbol{\theta}_n^{\text{im}}; \boldsymbol{y}_n). \tag{3}$$

Note that Eq. (3) is *implicit* because the next iterate $\boldsymbol{\theta}_n^{\text{im}}$ appears in both sides of the equation.[3] This simple tweak of the explicit SGD procedure has quite remarkable statistical properties. In particular, assuming a common starting point $\boldsymbol{\theta}_{n-1}^{\text{sgd}} = \boldsymbol{\theta}_{n-1}^{\text{im}} \triangleq \boldsymbol{\theta}$, one can show through a simple Taylor approximation of (3) around $\boldsymbol{\theta}$, that the implicit update satisfies

$$\Delta\boldsymbol{\theta}_n^{\text{im}} = (\boldsymbol{I} + a_n \hat{\boldsymbol{\mathcal{I}}}(\boldsymbol{\theta}; \boldsymbol{y}_n))^{-1} \Delta\boldsymbol{\theta}_n^{\text{sgd}} + \mathcal{O}(a_n^2), \tag{4}$$

where $\Delta\boldsymbol{\theta}_n = \boldsymbol{\theta}_n - \boldsymbol{\theta}_{n-1}$ for both methods, and $\hat{\boldsymbol{\mathcal{I}}}(\boldsymbol{\theta}; \boldsymbol{y}_n) = -\nabla\nabla\ell(\boldsymbol{\theta}; \boldsymbol{y}_n)$ is the *observed* Fisher information matrix. Thus, the implicit SGD procedure calculates updates that are a *shrinked* version of the explicit ones. In contrast to explicit SGD, implicit SGD is significantly more stable in small-samples, and it is also robust to misspecifications of the learning rate parameter α. Furthermore, implicit SGD computes iterates that belong in the support of the parameter space, whereas explicit SGD would normally require an additional projection step. Arguably, the normalized least mean squares (NLMS) filter (Nagumo and Noda 1967) was the first statistical model that used an implicit update as in Eq. (3) and was shown to be consistent and robust to input noise (Slock 1993). Theoretical justification for implicit SGD comes either from implicit variations of the Robbins–Monro procedure (Toulis et al. 2014), or through *proximal methods* in optimization (Parikh and Boyd 2013), such as mirror-descent (Nemirovski 1983; Beck and Teboulle 2003). Assuming differentiability of the log-likelihood, the implicit

[1] Second-order methods typically use the Hessian matrix of second-order derivatives of the log-likelihood and are discussed in detail in Sect. 3.

[2] Procedure (2) is actually an ascent algorithm because it aims to maximize the log-likelihood, and thus a more appropriate name would be stochastic gradient ascent. However, we will use the term "descent" in order to keep in line with the relevant optimization literature, which traditionally considers minimization problems through descent algorithms.

[3] The solution of the fixed-point equation (3) requires additional computations per iterations. However, Toulis et al. (2014) derive a computationally efficient implicit algorithm in the context of generalized linear models. Furthermore, approximate solutions of implicit updates are possible for any statistical model (see Eq. (4)).

SGD update (3) can be expressed as a proximal method through the solution of

$$\boldsymbol{\theta}_n^{\text{im}} = \arg\max_{\boldsymbol{\theta}} \left\{ -\frac{1}{2}||\boldsymbol{\theta} - \boldsymbol{\theta}_{n-1}^{\text{im}}||^2 + a_n \ell(\boldsymbol{\theta}; \boldsymbol{y}_n) \right\}, \tag{5}$$

where the right-hand side is the *proximal operator*. The update in Eq. (5) is the stochastic version of the deterministic proximal point algorithm by Rockafellar (1976), and has been analyzed recently, in various forms, for convergence and stability (Ryu and Boyd 2014; Rosasco et al. 2014). Recent work has established the consistency of certain implicit methods similar to (3) (Kivinen and Warmuth 1995; Kivinen et al. 2006; Kulis and Bartlett 2010) and their robustness has been useful in a range of modern machine learning problems (Nemirovski et al. 2009; Kulis and Bartlett 2010; Schuurmans and Caelli 2007).

The structure of this chapter is as follows. In Sect. 2 we give an overview of the Robbins–Monro procedure and Sakrison's recursive estimation method, which form the theoretical basis of SGD methods; we further provide a quick overview of early results on the statistical efficiency of the aforementioned methods. In Sect. 3, we formally introduce explicit and implicit SGD, and treat those procedures as *statistical estimation methods* that provide an estimator $\boldsymbol{\theta}_n$ of the model parameters $\boldsymbol{\theta}_\star$ after n iterations. In Sect. 3.1 we give results on the frequentist statistical properties of SGD estimators i.e., their asymptotic bias and asymptotic variance across multiple realizations of the dataset $\boldsymbol{Y}$. We then leverage those results to study optimal learning rate sequences a_n (Sect. 3.4), the loss of statistical efficiency in SGD and ways to fix it through reparameterization (Sect. 3.3). We briefly discuss stability in Sect. 3.2. In Sect. 3.5, we present significant extensions to first-order SGD, namely averaged SGD, variants of second-order SGD, and Monte-Carlo SGD. Finally, in Sect. 4, we review significant applications of SGD in various areas of statistics and machine learning, namely in online EM, MCMC posterior sampling, reinforcement learning, and deep learning.

2 Stochastic approximations

2.1 Robbins and Monro's procedure

Consider the one-dimensional setting where one data point is denoted by $y_n \in \mathbb{R}$ and it is controlled by a parameter θ with regression function $M(\theta) = \mathbb{E}(y|\,\theta)$ that is nondecreasing, and whose analytic form might be unknown. Robbins and Monro (1951) considered the problem of finding the unique point $\theta_\star$ for which $M(\theta_\star) = 0$. They devised a procedure, known as the *Robbins–Monro procedure*, in which an estimate θ_{n-1} of $\theta_\star$ is utilized to sample one new data point y_n such that $\mathbb{E}(y_n|\,\theta_{n-1}) = M(\theta_{n-1})$; the estimate is then updated according to the following simple rule:

$$\theta_n = \theta_{n-1} - a_n y_n. \tag{6}$$

The scalar $a_n > 0$ is the learning rate and should decay to zero, but not too fast in order to guarantee convergence. Robbins and Monro (1951) proved that $\mathbb{E}\left((\theta_n - \theta_\star)^2\right) \to 0$ when

(a) $(x - \theta_\star)M(x) > 0$ for x in a neighborhood of $\theta_\star$,
(b) $\mathbb{E}\left(y_n^2 \middle|\, \theta\right) < \infty$ for any θ, and
(c) $\sum_{i=1}^{\infty} a_i = \infty$ and $\sum_{i=1}^{\infty} a_i^2 < \infty$.

The original proof is technical but the main idea is straightforward. Let $b_n \triangleq \mathbb{E}\left((\theta_n - \theta_\star)^2\right)$ denote the squared error, then through iteration (6) one can obtain

$$b_n = b_{n-1} - 2a_n \mathbb{E}\left((\theta_{n-1} - \theta_\star)M(\theta_{n-1})\right) + a_n^2 \mathbb{E}\left(y_n^2\right). \tag{7}$$

In the neighborhood of $\theta_\star$ we have $M(\theta_{n-1}) \approx M'(\theta_\star)(\theta_{n-1} - \theta_\star)$, and thus

$$b_n = (1 - 2a_n M'(\theta_\star))b_{n-1} + a_n^2 \mathbb{E}\left(y_n^2\right). \tag{8}$$

For a learning rate sequence of the form $a_n = \alpha/n$, typical proof techniques in stochastic approximation (Chung 1954) can establish that $b_n \to 0$. Furthermore, it holds $nb_n \to \alpha^2\sigma^2(2\alpha M'(\theta_\star) - 1)^{-1}$ where $\sigma^2 \triangleq \mathbb{E}\left(y_n^2 \middle|\, \theta_\star\right)$ when this limit exists; this result was not given in the original paper by Robbins and Monro (1951) but it was soon derived by several other authors (Chung 1954; Sacks 1958; Fabian 1968). Thus, the learning parameter α is critical for the performance of the Robbins–Monro procedure. Its optimal value is $\alpha_\star = 1/M'(\theta_\star)$, which requires knowledge of the slope of $M(\cdot)$ at the true parameter values. In the multidimensional case, the efficiency of stochastic approximations—including stochastic gradient descent—depends on the Jacobian of the mean-value function of the statistic used in the iterations (see Sect. 3.1). This early result spawned an important line of research on *adaptive* stochastic approximation methods, such as the Venter process (Venter 1967), in which quantities that are important for the convergence of the stochastic process (e.g., the quantity $M'(\theta_\star)$) are also being estimated along the way.

2.2 Sakrison's recursive estimation method

Although initially applied in sequential experiment design, the Robbins–Monro procedure was soon adapted for estimation. Sakrison (1965) was interested in estimating the

parameters $\boldsymbol{\theta}_\star$ of a model that generated i.i.d. observations $\boldsymbol{y}_n$ in a way that is computationally and statistically efficient, similar to our setup in the introduction. He recognized that the statistical identity $\mathbb{E}\left(\nabla\ell(\boldsymbol{\theta}_\star; \boldsymbol{y}_n)\right) = 0$, where the expectation is over the observed data $\boldsymbol{y}_n$, provides the theoretical basis for a general estimation method using the Robbins–Monro procedure. Sakrison's *recursive estimation method* was essentially one of the first *explicit* SGD method proposed in the literature:

$$\boldsymbol{\theta}_n^{\text{sak}} \approx \boldsymbol{\theta}_{n-1}^{\text{sak}} + (1/n)\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_{n-1}^{\text{sak}})^{-1}\nabla\ell(\boldsymbol{\theta}_{n-1}^{\text{sak}}; \boldsymbol{y}_n), \tag{9}$$

The SGD procedure (9) is second-order since it is using a matrix to condition the gradient of the log-likelihood. Under typical regularity conditions $\boldsymbol{\theta}_n^{\text{sak}} \to \boldsymbol{\theta}_\star$, and thus $\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_n^{\text{sak}}) \to \boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star)$. Sakrison (1965) also proved that $n\mathbb{E}\left(||\boldsymbol{\theta}_n^{\text{sak}} - \boldsymbol{\theta}_\star||^2\right) \to \text{trace}(\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star)^{-1})$, and so the estimation of $\boldsymbol{\theta}_\star$ is asymptotically efficient under this norm objective. It is interesting to note that updates of the form (9) appeared very early in the statistical literature. For example, Fisher (1925b) suggested that an inefficient estimator $\boldsymbol{\theta}_N$ using N data points can be made asymptotically efficient by considering a new estimator $\boldsymbol{\theta}_N^+ = \boldsymbol{\theta}_N + (1/N)\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star)^{-1}\sum_{i=1}^N \nabla\ell(\boldsymbol{\theta}_N; \boldsymbol{y}_i)$. The surprising result in Sakrison's work was that asymptotically optimal estimation is also possible by using only gradients of the log-likelihood on single data points $\boldsymbol{y}_i$ in the iterated algorithm (9).

3 Estimation with stochastic gradient methods

For the rest of this chapter we will consider a simple generalization of explicit and implicit SGD that is similar to Sakrison's method as follows:

$$\boldsymbol{\theta}_n^{\text{sgd}} = \boldsymbol{\theta}_{n-1}^{\text{sgd}} + \boldsymbol{C}_n\nabla\ell(\boldsymbol{\theta}_{n-1}^{\text{sgd}}; \boldsymbol{y}_n), \tag{10}$$

$$\boldsymbol{\theta}_n^{\text{im}} = \boldsymbol{\theta}_{n-1}^{\text{im}} + \boldsymbol{C}_n\nabla\ell(\boldsymbol{\theta}_n^{\text{im}}; \boldsymbol{y}_n). \tag{11}$$

In general all $\boldsymbol{C}_n$ are symmetric and positive-definite matrices, and serve to stabilize and optimize stochastic iterations as in (10) and (11). In the limit $n\boldsymbol{C}_n \to \boldsymbol{C}$ where $\boldsymbol{C}$ is a symmetric and positive-definite matrix. If $\boldsymbol{C}_n$ is not trivial (e.g., scaled identity), we will refer to (10) and (11) as *second-order explicit SGD* and *second-order implicit SGD*, respectively. When $\boldsymbol{C}_n = a_n\boldsymbol{I}$ i.e., it is the scaled identity matrix for some sequence $a_n > 0$ satisfying the Robbins–Monro conditions, we will refer to (10) and (11) as *first-order explicit SGD* and *first-order implicit SGD*, respectively; in this case, definitions (10) and (11) are identical to definitions (2) and (3) in the introduction. In some cases, we will consider models in the exponential family under the *natural parameterization* with density

$$f(\boldsymbol{y}_n; \boldsymbol{\theta}_\star) = \exp\{\boldsymbol{\theta}_\star{}^\intercal\boldsymbol{s}(\boldsymbol{y}_n) - A(\boldsymbol{\theta}_\star) + B(\boldsymbol{y}_n)\}, \tag{12}$$

where $\boldsymbol{s}(\boldsymbol{y}_n)$ is the vector of p sufficient statistics, and $A(\cdot)$, $B(\cdot)$ are appropriate real-valued functions. The SGD procedures simplify to

$$\boldsymbol{\theta}_n^{\text{sgd}} = \boldsymbol{\theta}_{n-1}^{\text{sgd}} + \boldsymbol{C}_n(\boldsymbol{s}(\boldsymbol{y}_n) - \nabla A(\boldsymbol{\theta}_{n-1}^{\text{sgd}})), \tag{13}$$

$$\boldsymbol{\theta}_n^{\text{im}} = \boldsymbol{\theta}_{n-1}^{\text{im}} + \boldsymbol{C}_n(\boldsymbol{s}(\boldsymbol{y}_n) - \nabla A(\boldsymbol{\theta}_n^{\text{im}})). \tag{14}$$

In what follows, we will consider a *frequentist* evaluation of SGD as a statistical estimation method i.e., we will consider $\boldsymbol{\theta}_n^{\text{sgd}}$ (or $\boldsymbol{\theta}_n^{\text{im}}$) to be an *estimator* of $\boldsymbol{\theta}_\star$, and we will focus on its bias and variance across multiple realizations of the dataset $\boldsymbol{Y} = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_n\}$, under the same model and parameter $\boldsymbol{\theta}_\star$.[4]

3.1 Asymptotic bias and variance

Typically, online procedures such as SGD have two phases, namely the *exploration phase* (or search phase) and the *convergence phase* (Amari 1998; Benveniste et al. 2012). In the exploration phase the iterates rapidly approach $\boldsymbol{\theta}_\star$, whereas in the convergence phase they jitter around $\boldsymbol{\theta}_\star$ within a ball of slowly decreasing radius. We will overview a typical analysis of SGD in the final convergence phase in which we assume that a Taylor approximation in the neighborhood of $\boldsymbol{\theta}_\star$ is accurate (Murata 1998; Toulis et al. 2014). In particular let $\boldsymbol{\mu}(\boldsymbol{\theta}) = \mathbb{E}\left(\nabla\ell(\boldsymbol{\theta}; \boldsymbol{y}_n)\right)$, and assume that

$$\boldsymbol{\mu}(\boldsymbol{\theta}_n) = \boldsymbol{\mu}(\boldsymbol{\theta}_\star) + \boldsymbol{J}_\mu(\boldsymbol{\theta}_\star)(\boldsymbol{\theta}_n - \boldsymbol{\theta}_\star) + o(a_n), \tag{15}$$

where $\boldsymbol{J}_\mu$ is the Jacobian of the function $\boldsymbol{\mu}(\cdot)$, and $o(a_n)$ denotes a vector sequence $\boldsymbol{v}_n$ for which $||\boldsymbol{v}_n||/a_n \to 0$. Under typical regularity conditions $\boldsymbol{\mu}(\boldsymbol{\theta}_\star) = \boldsymbol{0}$ and $\boldsymbol{J}_\mu(\boldsymbol{\theta}_\star) = -\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star)$. Thus, if we denote the biases of the two SGD methods as $\mathbb{E}(\boldsymbol{\theta}_n^{\text{sgd}} - \boldsymbol{\theta}_\star) \triangleq \boldsymbol{b}_n^{\text{sgd}}$ and $\mathbb{E}(\boldsymbol{\theta}_n^{\text{im}} - \boldsymbol{\theta}_\star) \triangleq \boldsymbol{b}_n^{\text{im}}$, by taking expectations in Eqs. (10) and (11) we obtain

$$\boldsymbol{b}_n^{\text{sgd}} = (\boldsymbol{I} - \boldsymbol{C}_n\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star))\ \ \boldsymbol{b}_{n-1}^{\text{sgd}} + o(a_n), \tag{16}$$

$$\boldsymbol{b}_n^{\text{im}} = (\boldsymbol{I} + \boldsymbol{C}_n\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star))^{-1}\ \ \boldsymbol{b}_{n-1}^{\text{im}} + o(a_n). \tag{17}$$

We observe that the convergence rate at which the two methods become unbiased in the limit differs in two significant ways. First, the explicit SGD method converges

[4] This is an important distinction because, traditionally, the focus in optimization has been to obtain fast convergence to some point $\widehat{\boldsymbol{\theta}}$ that minimizes the empirical loss, e.g., the maximum-likelihood estimator. From a statistical viewpoint, under variability of the data, there is a trade-off between convergence to an estimator and its asymptotic variance (Le et al. 2004).

faster than the implicit one because $||(\boldsymbol{I} - \boldsymbol{C}_n\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star))|| < ||(\boldsymbol{I}+\boldsymbol{C}_n\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star))^{-1}||$, for sufficiently large n; the rates become equal in the limit as $a_n \to 0$. However, the implicit method compensates by being more stable in the specification of the condition matrices $\boldsymbol{C}_n$. For example, the explicit SGD requires that the sequence $\boldsymbol{I} - \boldsymbol{C}_n\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star)$ is comprised of matrices with eigenvalues less than one, in order to guarantee stability; this is a significant source of trouble when applying explicit SGD in practice. In contrast, for *any* specification of positive-definite $\boldsymbol{C}_n$, the eigenvalues of $(\boldsymbol{I} + \boldsymbol{C}_n\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star))^{-1}$ are less than one, and thus implicit SGD is *unconditionally stable*; we will discuss more about stability in Sect. 3.4.

In regard to statistical efficiency, Taylor approximation can also be used to establish recursive equations for the asymptotic variance of $\boldsymbol{\theta}_n^{\text{sgd}}$ and $\boldsymbol{\theta}_n^{\text{im}}$. For example, Toulis et al. (2014) show that if $\boldsymbol{C}$ is a symmetric matrix that commutes with $\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star)$ such that $(2\boldsymbol{C}\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star)-\boldsymbol{I})$ is positive-definite and $n\boldsymbol{C}_n \to \boldsymbol{C}$, it holds

$$
\begin{aligned}
n\text{Var}(\boldsymbol{\theta}_n^{\text{sgd}}) &\to (2\boldsymbol{C}\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star) - \boldsymbol{I})^{-1}\boldsymbol{C}\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star)\boldsymbol{C}^\intercal,\\
n\text{Var}(\boldsymbol{\theta}_n^{\text{im}}) &\to (2\boldsymbol{C}\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star) - \boldsymbol{I})^{-1}\boldsymbol{C}\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star)\boldsymbol{C}^\intercal;
\end{aligned}
\tag{18}
$$

i.e., both SGD methods have the same asymptotic variance. Thus, for first-order SGD procedures where $\boldsymbol{C}_n = a_n\boldsymbol{I}$ with $na_n \to \alpha > 0$ we obtain

$$
\begin{aligned}
n\text{Var}(\boldsymbol{\theta}_n^{\text{sgd}}) &\to \alpha^2(2\alpha\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star) - \boldsymbol{I})^{-1}\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star),\\
n\text{Var}(\boldsymbol{\theta}_n^{\text{im}}) &\to \alpha^2(2\alpha\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star) - \boldsymbol{I})^{-1}\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star).
\end{aligned}
\tag{19}
$$

The matrix term $(2\boldsymbol{C}\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star)-\boldsymbol{I})^{-1}$ represents the information that is lost by SGD, and it needs to be identity for optimal statistical efficiency (see Sect. 3.4). In fact, in more generality, this term is equal to $(2\boldsymbol{C}\boldsymbol{J}_\mu(\boldsymbol{\theta}_\star) - \boldsymbol{I})^{-1}$ where $\boldsymbol{\mu}(\boldsymbol{\theta})$ is mean-value function of the statistic used in SGD (see also Equation (15)), and $\boldsymbol{J}_\mu(\boldsymbol{\theta}_\star)$ is its Jacobian at the true parameter values. Therefore, the asymptotic efficiency of SGD methods depends crucially on the Jacobian of the mean-value function of the statistic used in the SGD iterations.

Asymptotic variance results similar to (18) were first studied in the stochastic approximation literature by Chung (1954), Sacks (1958), and followed by Fabian (1968) and several other authors (see also Ljung et al. 1992, Parts I, II), but not in a closed-form (18), as most analyses were not done under the context of recursive statistical estimation. Furthermore, Sakrison's asymptotic efficiency result (Sakrison 1965) can be recovered by setting $\boldsymbol{C}_n = (1/n)\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_{n-1})^{-1}$; in this case the asymptotic variance for both estimators is $(1/n)\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star)^{-1}$ i.e., it is the optimal asymptotic efficiency of the maximum-likelihood estimator.

3.2 Stability issues

Stability has been a well-known issue for explicit SGD. The main problem in practice is that the learning rate sequence needs to agree with the eigenvalues of the Fisher information matrix. To see this, let us simplify (16) and (17) by dropping the remainder terms $o(a_n)$. Then we obtain

$$
\boldsymbol{b}_n^{\text{sgd}} = (\boldsymbol{I} - \boldsymbol{C}_n\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star))\boldsymbol{b}_{n-1}^{\text{sgd}} = \boldsymbol{P}_1^n\boldsymbol{b}_0, \tag{20}
$$

$$
\boldsymbol{b}_n^{\text{im}} = (\boldsymbol{I} + \boldsymbol{C}_n\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star))^{-1}\boldsymbol{b}_{n-1}^{\text{im}} = \boldsymbol{Q}_1^n\boldsymbol{b}_0, \tag{21}
$$

where $\boldsymbol{P}_1^n = \prod_{i=1}^n(\boldsymbol{I} - \boldsymbol{C}_i\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star))$, $\boldsymbol{Q}_1^n = \prod_{i=1}^n(\boldsymbol{I} + \boldsymbol{C}_i\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star))^{-1}$, and $\boldsymbol{b}_0$ denotes the initial bias of the two procedures from some common starting point $\boldsymbol{\theta}_0$. Thus, the matrices $\boldsymbol{P}_1^n$ and $\boldsymbol{Q}_1^n$ describe how fast the initial bias decays for the explicit and implicit SGD, respectively. Assuming convergence, $\boldsymbol{P}_1^n \to \boldsymbol{0}$ and $\boldsymbol{Q}_1^n \to \boldsymbol{0}$, and thus we say that both methods are *asymptotically stable*. However, they have significant differences in small-to-moderate samples. For simplicity, let us compare the two SGD procedures in their first-order formulation where $\boldsymbol{C}_n = a_n\boldsymbol{I}$ and $a_n = \alpha/n$ for some $\alpha > 0$.

In explicit SGD, the eigenvalues of $\boldsymbol{P}_1^n$ can be calculated as $\lambda_i' = \prod_j(1 - \alpha\lambda_i/j) = \mathcal{O}(n^{-\alpha\lambda_i})$, for $0 < \alpha\lambda_i < 1$, where λ_i are the eigenvalues of the Fisher information matrix $\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star)$. Thus, the magnitude of $\boldsymbol{P}_1^n$ will be dominated by $\lambda_{\max}$, the maximum eigenvalue of $\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star)$, and the rate of convergence to zero will be dominated by $\lambda_{\min}$, the minimum eigenvalue of $\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star)$. The condition $\alpha\lambda_{\max} \leq 1 \Rightarrow \alpha \leq 1/\lambda_{\max}$ is required for stability, but for fast convergence we require $\alpha\lambda_{\min} \approx 1$. In high-dimensional settings, this could be the source of serious problems because $\lambda_{\max}$ could be at the order of p i.e., the number of model parameters. Thus, in explicit SGD the requirements for stability and speed of convergence are in conflict. A conservative learning rate sequence can guarantee stability but this comes at a price in convergence which will be at the order of $\mathcal{O}(n^{-\alpha\lambda_{\min}})$, and vice versa. In stark contrast, the implicit procedure is *unconditionally stable*. The eigenvalues of $\boldsymbol{Q}_1^n$ are $\lambda_i' = \prod_{j=1}^n 1/(1 + \alpha\lambda_i/j) = \mathcal{O}(n^{-\alpha\lambda_i})$, and thus are guaranteed to be less than one for any choice of the learning rate parameter α. The critical difference with explicit SGD is that it is no longer required to have a small α for stability because the eigenvalues of $\boldsymbol{Q}_1^n$ will always be less than one.

Based on this analysis the magnitude of $\boldsymbol{P}_1^n$ can become arbitrarily large, and thus explicit SGD is likely to numerically diverge. In contrast, $\boldsymbol{Q}_1^n$ is guaranteed to be bounded, and so under any misspecification of the learning rate parameter the implicit SGD procedure is guaranteed to remain stable. The instability of explicit SGD is well known, and requires careful work to be avoided in practice. For example, a typical learning rate for explicit SGD is of the form

$a_n = \alpha(\alpha\beta + n)^{-1}$, where β is chosen so that the explicit updates will not diverge; a reasonable choice is to set $\beta = \text{trace}(\mathcal{I}(\boldsymbol{\theta}_\star))$ and α to be set close to $1/\lambda_{min}$. Such *explicit* normalization of the learning rates is not necessary in implicit SGD because, as shown in Equation (4), the implicit update performs such normalization indirectly.

Finally, an important line of work in the stability of stochastic approximations has been inspired by Huber's work in robust statistics (Huber et al. 1964; Huber 2011). In our notation, robust stochastic approximation considers iterations of the following form

$$\boldsymbol{\theta}_n = \boldsymbol{\theta}_{n-1} + \boldsymbol{C}_n \boldsymbol{\psi}(\boldsymbol{s}(\boldsymbol{y}_n) - \boldsymbol{h}(\boldsymbol{\theta}_{n-1})), \tag{22}$$

where an appropriate function $\boldsymbol{\psi}$ is sought for robust estimation; in this problem we assume $\mathbb{E}\left(\boldsymbol{s}(\boldsymbol{y}_n)\right) = h(\boldsymbol{\theta}_\star)$ but the distribution of $\boldsymbol{s}(\boldsymbol{y}_n) - \boldsymbol{h}(\boldsymbol{\theta}_\star)$ – denoted by $f(\cdot)$ – is unknown. In a typical setup, $f(\cdot)$ is considered to belong to a family of distributions $\mathcal{P}$, and $\boldsymbol{\psi}$ is selected as

$$\boldsymbol{\psi}_\star = \arg_\psi \min \max_{f\in\mathcal{P}} \lim_{n\to\infty} n\text{Var}(\boldsymbol{\theta}_n)$$

i.e., such that the maximum possible variance over the family $\mathcal{P}$ is minimized. Several important results have been achieved by Martin and Masreliez (1975) and Polyak and Tsypkin (1979). For example, in linear models where $\boldsymbol{\mu}(\cdot)$ is linear in $\boldsymbol{\theta}$ and $\boldsymbol{s}(\boldsymbol{y}_n)$ is one-dimensional, consider the general family $\mathcal{P} = \{f : f(0) \geq \epsilon\}$ as the set of all symmetric densities that are positive at 0. Then the optimal choice is $\boldsymbol{\psi}_\star = \text{sign}(\cdot)$ i.e., the sign function, because it can be shown that the Laplace distribution is the member density of $\mathcal{P}$ that gives the least information about the parameters $\boldsymbol{\theta}_\star$.

3.3 Choice of parameterization and efficiency

First-order SGD methods are attractive for their computational performance, but the variance result (19) shows that they may suffer a significant loss in statistical efficiency. However, a reparameterization of the problem could yield a first-order SGD method that is optimal. The method can be described as follows. First, assume the exponential family (12) such that $\nabla\ell(\boldsymbol{\theta}; \boldsymbol{y}_n) = \boldsymbol{s}(\boldsymbol{y}_n) - \boldsymbol{h}(\boldsymbol{\theta})$, where $h(\boldsymbol{\theta}) = \nabla A(\boldsymbol{\theta}) = \mathbb{E}\left(\boldsymbol{s}(\boldsymbol{y}_n)\middle|\, \boldsymbol{\theta}_\star = \boldsymbol{\theta}\right)$, and consider the reparameterization

$$\boldsymbol{\omega} \triangleq \boldsymbol{h}(\boldsymbol{\theta}), \tag{23}$$

which we assume it exists, it is 1-1 and easy to compute; these are critical assumptions that are hard, but not impossible to hold in practice. We will refer to (23) as the *mean-value parameterization* and $\boldsymbol{\omega}$ as the mean-value parameters. Starting with an estimate $\boldsymbol{\omega}_0$ of $\boldsymbol{\omega}_\star = \boldsymbol{h}(\boldsymbol{\theta}_\star)$, we can define the SGD procedures on this new parameter space as

$$\boldsymbol{\omega}_n^{\text{sgd}} = \boldsymbol{\omega}_{n-1}^{\text{sgd}} + (1/n)(\boldsymbol{s}(\boldsymbol{y}_n) - \boldsymbol{\omega}_{n-1}^{\text{sgd}}), \tag{24}$$

$$\boldsymbol{\omega}_n^{\text{im}} = \boldsymbol{\omega}_{n-1}^{\text{im}} + (1/n)(\boldsymbol{s}(\boldsymbol{y}_n) - \boldsymbol{\omega}_n^{\text{im}}), \tag{25}$$

where we also set $\boldsymbol{C}_n = (1/n)\boldsymbol{I}$ so that $\boldsymbol{C} = \boldsymbol{I}$. In this case, the explicit SGD simply calculates the running average of the complete sufficient statistic i.e., $\boldsymbol{\omega}_n^{\text{sgd}} = n^{-1}\sum_{i=1}^n \boldsymbol{s}(\boldsymbol{y}_i)$, and thus it is identical to the MLE estimator; similarly the implicit SGD satisfies $\boldsymbol{\omega}_n^{\text{im}} = (n+1)^{-1}\sum_{i=1}^n \boldsymbol{s}(\boldsymbol{y}_i)$ i.e., it is a slightly biased version of the MLE. It is thus straightforward to show (see for example Toulis and Airoldi 2014) that the mean-value parameterization is optimal i.e.,

$$\text{Var}\left(\boldsymbol{h}^{-1}(\boldsymbol{\omega}_n^{\text{sgd}})\right) \to (1/n)\mathcal{I}(\boldsymbol{\theta}_\star)^{-1}, \text{Var}\left(\boldsymbol{h}^{-1}(\boldsymbol{\omega}_n^{\text{im}})\right) \to (1/n)\mathcal{I}(\boldsymbol{\theta}_\star)^{-1}. \tag{26}$$

Intuitively, the mean-value parameterization transforms all parameters into location parameters. The Jacobian of the regression function of the statistic is $\boldsymbol{J}_\mu(\boldsymbol{\omega}_\star) = \nabla_\omega \mathbb{E}\left(\boldsymbol{s}(\boldsymbol{y}_n)\middle|\, \boldsymbol{\omega} = \boldsymbol{\omega}_\star\right) = \boldsymbol{I}$, and thus the information loss described in Equation (18) is avoided since $(2\boldsymbol{C}\boldsymbol{J}_\mu(\boldsymbol{\omega}_\star) - \boldsymbol{I})^{-1} = \boldsymbol{I}$. Transforming back to the original parameter space incurs no information loss as well, and so estimation of $\boldsymbol{\theta}_\star$ is efficient. This method is illustrated in the following example.

Example. Consider the problem of estimating (μ, σ^2) from normal observations $y_n \sim \mathcal{N}(\mu, \sigma^2)$, and let $\boldsymbol{\theta}_\star = (\mu, \sigma^2)$ which is not the natural parameterization. Consider sufficient statistics $\boldsymbol{s}(\boldsymbol{y}_n) = (y_n, y_n^2)$ such that $\mathbb{E}\left(\boldsymbol{s}(\boldsymbol{y}_n)\right) = (\mu, \mu^2 + \sigma^2) \triangleq (\omega_1, \omega_2)$. The parameter $\boldsymbol{\omega} = (\omega_1, \omega_2)$ corresponds to the mean-value parameterization. The inverse transformation is $\mu = \omega_1$ and $\sigma^2 = \omega_2 - \omega_1^2$, and thus its Jacobian is

$$\boldsymbol{J}_h^{-1} = \begin{pmatrix} 1 & 0 \\ -2\omega_1 & 1 \end{pmatrix}.$$

The variance of $\boldsymbol{s}(\boldsymbol{y}_n)$ is given by

$$\boldsymbol{V}(\boldsymbol{\theta}_\star) = \begin{pmatrix} Q & 2\omega_1 Q \\ 2\omega_1 Q & 4\omega_1^2 Q + 2Q^2 \end{pmatrix},$$

where $Q = \omega_2 - \omega_1^2 = \sigma^2$. Thus the variance of $(\widehat{\omega_1}, \widehat{\omega_2})$ is $(1/n)\boldsymbol{V}(\boldsymbol{\theta}_\star)$ and the variance of $(\widehat{\mu}, \widehat{\sigma^2})$ is given by

$$\text{Var}\left((\widehat{\mu}, \widehat{\sigma^2})\right) = (1/n)\boldsymbol{J}_h^{-1}\boldsymbol{V}\boldsymbol{J}_h^{-1T} = (1/n)\begin{pmatrix} Q & 0 \\ 0 & 2Q^2 \end{pmatrix} = (1/n)\begin{pmatrix} \sigma^2 & 0 \\ 0 & 2\sigma^4 \end{pmatrix},$$

which is exactly the asymptotic variance of the MLE estimate. In practice, however, the mean-value transformation is rarely possible. Still, the intuition of transforming the model parameters into location parameters can be very useful in

many situations, even when such transformation is approximate.

3.4 Choice of learning rate sequence

An interesting observation on the asymptotic variance results (18) is that for any choice of the symmetric positive-definite matrix $\boldsymbol{C}$,

$$(2\boldsymbol{C}\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star) - \boldsymbol{I})^{-1}\boldsymbol{C}\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star)\boldsymbol{C}^\intercal \geq \boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star)^{-1}, \tag{27}$$

where $\boldsymbol{A} \geq \boldsymbol{B}$ for two matrices $\boldsymbol{A}, \boldsymbol{B}$ indicates that $\boldsymbol{A} - \boldsymbol{B}$ is nonnegative-definite. Even in second-order form, both methods incur an efficiency loss when compared to the maximum-likelihood estimator, which can be quantified exactly through (18). Thus, there are two ways to achieve asymptotic efficiency. First, one can design the condition matrix such that $n\boldsymbol{C}_n \to \boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star)^{-1} \triangleq \boldsymbol{C}_\star$.[5] However, this requires knowledge of the Fisher information matrix on the true parameters $\boldsymbol{\theta}_\star$, which is usually unknown. The Venter process (Venter 1967) was the first method to follow an adaptive approach to estimate this matrix, and was later analyzed and extended by several other authors (Fabian 1973; Lai and Robbins 1979; Amari et al. 2000; Bottou and Le Cun 2005). Adaptive methods that perform an approximation of the matrix $\boldsymbol{C}_\star$ (e.g., through a Quasi-Newton scheme) have recently been applied with considerable success (Schraudolph et al. 2007; Bordes et al. 2009); see Sect. 3.5.2 for more details.

In contrast, an efficiency loss is generally unavoidable in first-order SGD i.e., when $\boldsymbol{C}_n = a_n\boldsymbol{I}$ with $na_n \to \alpha$. Asymptotic efficiency can occur only when $\lambda_i = 1/\alpha$ i.e., when all eigenvalues λ_i of the Fisher information matrix $\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star)$ are identical. When λ_i's are distinct the eigenvalues of the asymptotic variance matrix $n\mathrm{Var}(\boldsymbol{\theta}_n^{\mathrm{sgd}})$ (or $n\mathrm{Var}(\boldsymbol{\theta}_n^{\mathrm{im}})$) are $\alpha^2\lambda_i/(2\alpha\lambda_i - 1)$ which is at least $1/\lambda_i$ for any α. In this case, one reasonable way to set the parameter α would be to minimize the trace of the asymptotic variance matrix i.e., solve

$$\hat{\alpha} = \arg\min_{\alpha} \sum_i \alpha^2\lambda_i/(2\alpha\lambda_i - 1), \tag{28}$$

under the constraint that $\alpha > 1/(2\lambda_{min})$, thus making an undesirable but necessary comprise for convergence in all parameter components. However, the eigenvalues $\{\lambda_i\}$ are unknown in practice and need to be estimated from the data. This problem has received significant attention recently and several methods exist (see Karoui 2008, and references within). A powerful alternative is to *reparametrize* the problem, apply SGD on the new parameter space, and then perform the inverse transformation, as in Sect. 3.3.

3.4.1 Practical considerations

There is voluminous amount of research literature on learning rate sequences for stochastic approximation and SGD. However, we decided to discuss this issue at the end of this section because the choice of the learning rate sequence conflates multiple design goals that are usually conflicting in practice, e.g., convergence (or bias), asymptotic variance, stability and so on.

In general, the theory presented so far indicates that the learning rate for first-order explicit SGD should be of the form $a_n = \alpha(\alpha\beta + n)^{-1}$. Note that $\lim_{n\to\infty} na_n = \alpha$, so α is indeed the learning rate parameter introduced in Sect. 1. Parameter α will control the asymptotic variance and a reasonable choice would be the solution of (28), which requires estimates of the eigenvalues of the Fisher information matrix $\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star)$. An easier method is to simply use $\alpha = 1/\lambda_{min}$, where λ_{min} is the minimum eigenvalue of $\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star)$; the value $1/\lambda_{min}$ is an approximate solution for (28), and also has good empirical performance (Xu 2011; Toulis et al. 2014). Parameter β can be used to stabilize explicit SGD. In particular, one would want to control the variance of the stochastic gradient $\mathrm{Var}\left(\nabla\ell(\boldsymbol{\theta}_n; \boldsymbol{y}_n)\right) = \boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star) + \mathcal{O}(a_n)$, for points near $\boldsymbol{\theta}_\star$; see also the stability analysis in Sect. 3.2. One reasonable value would thus be $\beta = \mathrm{trace}(\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star))$, which can be estimated easily by summing norms of the score function, i.e., $\hat{\beta} = \sum_{i=1}^n ||\nabla\ell(\boldsymbol{\theta}_i; \boldsymbol{y}_i)||^2$, similar to (Amari et al. 2000; Duchi et al. 2011)—see also Sect. (3.5.2).

For implicit SGD, the situation is a bit easier because a learning rate sequence $a_n = \alpha(\alpha + n)^{-1}$ works well in practice (Toulis et al. 2014). As before, α controls the efficiency of the method and so we can set $\alpha = 1/\lambda_{min}$ as in explicit SGD. The additional stability term (β) in explicit SGD is unnecessary because the implicit method performs such normalization (shrinkage) indirectly—see Eq. (4).

However, tuning the learning rate sequence eventually depends on problem-specific considerations, and there is a considerable variety of sequences that have been employed in practice (George and Powell 2006). Principled design of learning rates in SGD remains an important research topic (Schaul et al. 2012).

3.5 Some interesting extensions

3.5.1 Averaged stochastic gradient descent

Estimation with SGD can be optimized for statistical efficiency only with knowledge of the underlying model. For example, the optimal learning rate parameter α in first-order

[5] Similarly, a sequence of matrices $\boldsymbol{C}_n$ can be designed such that $\boldsymbol{C}_n \to \boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_\star)^{-1}$ (Sakrison 1965).

SGD requires knowledge of the eigenvalues of the Fisher information matrix $\mathcal{I}(\boldsymbol{\theta}_\star)$. In second-order SGD, optimality is achieved when one uses a sequence of matrices $\boldsymbol{C}_n$ such that $n\boldsymbol{C}_n \to \mathcal{I}(\boldsymbol{\theta}_\star)^{-1}$. Methods that approximate $\mathcal{I}(\boldsymbol{\theta}_\star)$ make up a significant class of methods in stochastic approximation. Another important class of stochastic approximation methods relies on *averaging* of the iterates. The corresponding SGD procedure is usually referred to as *averaged SGD*, or *ASGD* for short.[6]

Averaging of iterates in the Robbins–Monro procedures was studied independently by Ruppert (1988) and Bather (1989), and both proposed similar averaging schemes. If we use the notation of Sect. 2 (see also iteration (6)), Ruppert (1988) considered the following stochastic approximation procedure

$$\begin{aligned} \theta_n &= \theta_{n-1} - a_n y_n, \\ \bar{\theta}_n &= \frac{1}{n}\sum_{i=1}^{n} \theta_i, \end{aligned} \tag{29}$$

where $a_n = \alpha n^{-c}$ for $1/2 < c < 1$ and $\bar{\theta}_n$ are the estimates of the zero of the regression function $M(\theta)$. Under certain conditions, Ruppert (1988) showed that $n\mathrm{Var}(\bar{\theta}_n) \to \sigma^2/M'(\theta_\star)^2$, where $\sigma^2 = \mathrm{Var}\,(y_n|\,\theta_\star)$. Recall, that the typical Robbins–Monro procedure gives estimates with asymptotic variance $\alpha^2\sigma^2/(2\alpha M'(\theta_\star) - 1)$, which is at least equal to the variance of the averaged iterate. Ruppert (1988) provides a nice statistical intuition on why averaging gives such efficiency with larger learning rates. First, write $y_n = M(\theta_n) - \varepsilon_n$, where ε_n are zero-mean independent random variables with finite variance. The typical analysis in stochastic approximation starts by solving the recursion (6) to get an expression like the following

$$\theta_n - \theta_\star = \sum_{i=1}^{n} c(i,n) a_i \varepsilon_i + o(1), \tag{30}$$

where $c(i,n) = \exp\{-A(n) + A(i)\}$, $A(m) = K\sum_{j=1}^{m} a_j$ is the function of partial sums, and K is some constant. Ruppert (1988) shows that Eq. (30) can be rewritten as

$$\theta_n - \theta_\star = a_n \sum_{i=b(n)}^{n} c(i,n)\varepsilon_i + o(1), \tag{31}$$

where $b(n) = \lfloor n - Rn^c \log n \rfloor$ with R a positive constant, and $\lfloor\cdot\rfloor$ the positive integer floor function. Ruppert (1988) argues that when $a_n = \alpha/n$ then $b(n) = \mathcal{O}(1)$, and $\theta_n - \theta_\star$ is the weighted average of all noise variables ε_n. When $a_n = \alpha n^{-c}$ for $1/2 < c < 1$, then $\theta_n - \theta_\star$ is a weighted average of only $\mathcal{O}(n^c \log n)$ noise variables. Thus, in the former case there is significant autocorrelation in the series θ_n. In the latter case, for $0 < p_1 < p_2 < 1$ the variables $\theta_{\lfloor p_1 n\rfloor}$ and $\theta_{\lfloor p_2 n\rfloor}$ are asymptotically uncorrelated, and thus averaging improves the estimation efficiency.

Polyak and Juditsky (1992) derive further significant results for averaged SGD, showing in particular that ASGD can be asymptotically efficient as second-order SGD under certain mild assumptions. In fact, due to the authors' prior work in averaged stochastic approximation, ASGD is usually referred to as *Polyak-Ruppert* averaging scheme. Adoption of averaging schemes for statistical learning has been slow but steady over the years (Zhang 2004; Nemirovski et al. 2009; Bottou 2010; Cappé 2011). One practical reason is that averaging only helps when the underlying stochastic process is slow to converge, which is hard to know in practice; in fact, averaging can have an adverse effect when the underlying SGD process is converging well. Furthermore, the selection of the learning rate sequence is also important in ASGD, and a bad sequence can cause the algorithm to converge very slowly (Xu 2011), or even diverge. Research on ASGD is still ongoing as several directions, such as the combination of stable methods with averaging schemes, remain unexplored (e.g., stochastic proximal methods, implicit SGD). Furthermore, in a similar line of work, several methods have been developed that use averaging in order to reduce the variance of stochastic gradients (Johnson and Zhang 2013; Wang et al. 2013).

3.5.2 Second-order stochastic gradient descent

Sakrison's recursive estimation method (9) is the archetype of second-order SGD, but it requires an expensive matrix inversion at every iteration. Several methods have been developed that approximate such a matrix across iterations in stochastic approximation, and are generally termed *adaptive*. Early adaptive methods in stochastic approximation were given by Nevelson and Khasminskiĭ (1973) and Wei (1987); translated into a SGD procedure, such methods would recursively estimate $\mathcal{I}(\boldsymbol{\theta}_\star)$ by computing finite-differences $\boldsymbol{y}^j_{n,+} - \boldsymbol{y}^j_{n,-}$ sampled at $\boldsymbol{\theta}_n + c_n\boldsymbol{e}_j$ and $\boldsymbol{\theta}_n - c_n\boldsymbol{e}_j$, respectively, where $\boldsymbol{e}_j$ is the jth unit basis vector and c_n is an appropriate sequence of positive numbers. While such methods are very useful in sequential experiment design where one has control over the data generation process, they are impractical for modern online learning problems.

A simple and effective approach was proposed by Amari et al. (2000). The idea is to keep an estimate $\hat{\mathcal{I}}_n$ of $\mathcal{I}(\boldsymbol{\theta}_\star)$ and use an explicit SGD scheme as follows:

[6] The acronym ASGD is also used in machine learning to denote *asynchronous* SGD i.e., a variant of SGD that can be parallelized on multiple machines. We will not consider this variant here.

$$\hat{\mathcal{I}}_n = (1 - c_n)\hat{\mathcal{I}}_{n-1} + c_n \nabla \ell(\boldsymbol{\theta}_{n-1}; \boldsymbol{y}_n) \nabla \ell(\boldsymbol{\theta}_{n-1}; \boldsymbol{y}_n)^\intercal,$$
$$\boldsymbol{\theta}_n = \boldsymbol{\theta}_{n-1} + \hat{\mathcal{I}}_n^{-1} \nabla \ell(\boldsymbol{\theta}_{n-1}; \boldsymbol{y}_n). \quad (32)$$

Inversion of the estimate $\hat{\mathcal{I}}_n$ is (relatively) cheap by using the Sherman-Morrison formula. This scheme, however, introduces the additional problem of determining the sequence c_n in (32). In their work, Amari et al. (2000) advocated for a small constant $c_n = c > 0$ that can be determined through computer simulations.

Another notable approach based on Quasi-Newton methods (see Sect. 1) was developed by Bordes et al. (2009). Their method, termed *SGD-QN*, approximates the Fisher information matrix through a secant condition as in the original BFGS algorithm (Broyden 1965). The secant condition in SGD-QN is

$$\boldsymbol{\theta}_n - \boldsymbol{\theta}_{n-1} \approx \hat{\mathcal{I}}_{n-1}^{-1} \left[\nabla \ell(\boldsymbol{\theta}_n; \boldsymbol{y}_n) - \nabla \ell(\boldsymbol{\theta}_{n-1}; \boldsymbol{y}_n) \right] \triangleq \hat{\mathcal{I}}_{n-1}^{-1} \boldsymbol{\delta}_n, \quad (33)$$

where $\hat{\mathcal{I}}_n$ is kept diagonal. If we let $\boldsymbol{D}_n$ denote the diagonal matrix with ith diagonal element $d_{ii} = (\theta_{n,i} - \theta_{n-1,i})/\delta_{n,i}$, then the update of the approximation matrix in SGD-QN is given by

$$\hat{\mathcal{I}}_n \leftarrow \hat{\mathcal{I}}_{n-1} + \frac{2}{r}(\boldsymbol{D}_n - \hat{\mathcal{I}}_{n-1}), \quad (34)$$

and the update of $\boldsymbol{\theta}_n$ is similar to (32). The parameter r is controlled internally in the algorithm, and counts the number of times the update (34) has been performed.

A notable second-order method is also *AdaGrad* (Duchi et al. 2011), which adapts multiple learning rates using gradient information. In one popular variant of the method, AdaGrad keeps a diagonal $(p \times p)$ matrix $\boldsymbol{A}_n$ of learning rates that are updated at every iteration. Upon observing data $\boldsymbol{y}_n$, AdaGrad updates $\boldsymbol{A}_n$ as follows:

$$\boldsymbol{A}_n = \boldsymbol{A}_{n-1} + \operatorname{diag}(\nabla \ell(\boldsymbol{\theta}_{n-1}; \boldsymbol{y}_n) \nabla \ell(\boldsymbol{\theta}_{n-1}; \boldsymbol{y}_n)^\intercal), \quad (35)$$

where $\operatorname{diag}(\boldsymbol{A})$ is the diagonal matrix with the same diagonal as its matrix argument $\boldsymbol{A}$. Learning in AdaGrad proceeds through the iteration

$$\boldsymbol{\theta}_n = \boldsymbol{\theta}_{n-1} + \alpha \boldsymbol{A}_n^{-1/2} \circ \nabla \ell(\boldsymbol{\theta}_{n-1}; \boldsymbol{y}_n), \quad (36)$$

where $\alpha > 0$ is a learning rate parameter that is shared among all parameter components, and the symbol $\circ$ denotes elementwise multiplication. The original motivation for AdaGrad stems from proximal methods in optimization, but there is a statistical intuition why the update (36) is reasonable. In general, from an information perspective, a learning rate sequence a_n discounts an observation $\boldsymbol{y}_n$ according to the reciprocal of the *statistical information* that has been gathered so far for the parameter of interest $\boldsymbol{\theta}_\star$. The intuition behind a rate of the form $a_n = \alpha/n$ is that the information after n iterations is proportional to n, under the i.i.d. data assumption. In many dimensions where some parameter component affects outcomes less frequently than others, AdaGrad replaces the term n with an *estimate* of the information that has *actually* been received for that component. A (biased) estimate of this information is provided by the elements of $\boldsymbol{A}_n$ in (36), and is justified since $\mathbb{E}\left(\nabla \ell(\boldsymbol{\theta}; \boldsymbol{y}_n) \nabla \ell(\boldsymbol{\theta}; \boldsymbol{y}_n)^\intercal\right) = \mathcal{I}(\boldsymbol{\theta})$. Interestingly, implicit SGD and AdaGrad share the common property of shrinking explicit SGD estimates according to the Fisher information matrix. Second-order implicit SGD methods are yet to be explored, but further connections are possible.

3.5.3 Monte-Carlo stochastic gradient descent

A key requirement for the application of SGD procedures is that the likelihood is easy to evaluate. However, in many situations that are important in practice, this is not possible, for example when the likelihood is only known up to a normalizing constant. In such cases, definitions (10) and (11) cannot be applied directly since $\nabla \ell(\boldsymbol{\theta}; \boldsymbol{y}_n)$ cannot be computed. However, if unbiased samples of the log-likelihood gradients are available, then explicit SGD can be readily applied. This is possible if sampling from the model is relatively easy.

In particular, assume an exponential family model (12) that is easy to sample from, e.g., through Metropolis-Hastings. A variant of explicit SGD, termed *Monte-Carlo* SGD (Toulis and Airoldi 2014), can be constructed as follows. Starting from some estimate $\boldsymbol{\theta}_0^{\mathrm{mc}}$, iterate the following steps for each nth data point $\boldsymbol{y}_n$, where $n = 1, 2, \ldots, N$:

1. Get m samples from the model $\widetilde{\boldsymbol{y}}_i \sim f(\cdot; \boldsymbol{\theta}_{n-1}^{\mathrm{mc}})$, $i = 1, 2, \ldots, m$.
2. Compute average sufficient statistic $\widetilde{\boldsymbol{s}}_n = (1/m) \sum_{i=1}^m \boldsymbol{s}(\widetilde{\boldsymbol{y}}_i)$.
3. Update the estimate through

$$\boldsymbol{\theta}_n^{\mathrm{mc}} = \boldsymbol{\theta}_{n-1}^{\mathrm{mc}} + \boldsymbol{C}_n(\boldsymbol{s}(\boldsymbol{y}_n) - \widetilde{\boldsymbol{s}}_n). \quad (37)$$

The main idea of a Monte-Carlo SGD algorithm (37) is to use the current parameter estimate in order to *impute* the expected value of the sufficient statistic that would otherwise be available if the likelihood was easy to evaluate. Furthermore, assuming $n\boldsymbol{C}_n \to \boldsymbol{C}$, the asymptotic variance of the estimate satisfies

$$n\mathrm{Var}\left(\boldsymbol{\theta}_n^{\mathrm{mc}}\right) \to (1 + 1/m) \cdot (2\boldsymbol{C}\mathcal{I}(\boldsymbol{\theta}_\star) - \boldsymbol{I})^{-1} \boldsymbol{C}\mathcal{I}(\boldsymbol{\theta}_\star)\boldsymbol{C}^\intercal, \quad (38)$$

which exceeds the variance of the typical explicit SGD estimator by a factor of $(1 + 1/m)$. However, in its current form the Monte-Carlo SGD (37) is only explicit; an implicit version would require to sample data from the next iterate, which is technically challenging but an interesting open problem. Still, an approximate implicit implementation of Monte-Carlo SGD is possible using the intuition in Eq. (4). For example, one could simply run an explicit update as in (37), but then shrink according to $(\boldsymbol{I} + a_n \boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_n^{\mathrm{mc}}))^{-1}$, or more efficiently using a one-dimensional shrinkage factor $(1 + a_n \mathrm{trace}(\boldsymbol{\mathcal{I}}(\boldsymbol{\theta}_n^{\mathrm{mc}})))^{-1}$, for some decreasing sequence $a_n > 0$.

Theoretically Monte-Carlo SGD is based on *sampling-controlled* stochastic approximation methods in which the usual regression function of the Robbins–Monro procedure (6) is only accessible through sampling (Dupuis and Simha 1991), e.g., through MCMC. Convergence in such settings is subtle because it also depends on the ergodicity of the underlying Markov chain (Younes 1999). In practice, approximate variants of the aforementioned Monte-Carlo SGD procedure have been applied with considerable success to fit large models of neural networks, notably through the contrastive divergence algorithm, as we briefly discuss in Sect. 4.4.

4 Selected applications

SGD has found several important applications over the years. In this section we will review some of them, giving a preference to breadth over depth.

4.1 Online EM algorithm

The Expectation–Maximization algorithm (Dempster et al. 1977) is a numerically stable procedure to compute the maximum-likelihood estimator in latent variable models. Extending our notation, let $\boldsymbol{x}_n$ denote a latent variable at observed-data point $\boldsymbol{y}_n$, and let $f_{\mathrm{com}}(\boldsymbol{x}_n, \boldsymbol{y}_n; \boldsymbol{\theta})$ and $f_{\mathrm{obs}}(\boldsymbol{y}_n; \boldsymbol{\theta})$ denote the complete-data and observed-data density, respectively; similarly, ℓ_{com} and ℓ_{obs} will denote the respective log-likelihoods. For simplicity, we will assume that f_{com} is an exponential family model in the natural parameterization, as in (12), such that

$$f_{\mathrm{com}}(\boldsymbol{x}_n, \boldsymbol{y}_n; \boldsymbol{\theta}) = \exp\left\{\boldsymbol{s}(\boldsymbol{x}_n, \boldsymbol{y}_n)^{\intercal}\boldsymbol{\theta} - A(\boldsymbol{\theta}) + B(\boldsymbol{x}_n, \boldsymbol{y}_n)\right\}. \tag{39}$$

We will denote the corresponding Fisher information matrices as $\mathcal{I}_{\mathrm{com}}(\boldsymbol{\theta}) = -\mathbb{E}\left(\nabla\nabla\ell_{\mathrm{com}}(\boldsymbol{x}_n, \boldsymbol{y}_n; \boldsymbol{\theta})\right)$ and $\mathcal{I}_{\mathrm{obs}}(\boldsymbol{\theta}) = \mathbb{E}\left(\nabla\nabla\ell_{\mathrm{obs}}(\boldsymbol{y}_n; \boldsymbol{\theta})\right)$, where the expectations are considered with model parameters fixed at $\boldsymbol{\theta}$. Furthermore, let $\boldsymbol{Y} = (\boldsymbol{y}_1, \ldots, \boldsymbol{y}_N)$ denote the entire observed dataset as in Sect. 1, and $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N)$ be the corresponding latent variables. The traditional EM algorithm proceeds by iterating the following steps.

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}_n; \boldsymbol{Y}) = \mathbb{E}\left(\ell_{\mathrm{com}}(\boldsymbol{X}, \boldsymbol{Y}; \boldsymbol{\theta}) \middle| \boldsymbol{\theta}_n, \boldsymbol{Y}\right), \qquad \mathbf{E-step} \tag{40}$$

$$\boldsymbol{\theta}_{n+1} = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \boldsymbol{\theta}_n; \boldsymbol{Y}). \qquad \mathbf{M-step} \tag{41}$$

Dempster et al. (1977) showed that the EM algorithm converges to the maximum-likelihood estimator $\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \ell_{\mathrm{obs}}(\boldsymbol{Y}; \boldsymbol{\theta})$; furthermore, they showed that EM is an ascent algorithm i.e., the likelihood is strictly increasing at each iteration, and thus EM has a desirable numerical stability. However, the EM algorithm is impractical for the analysis of large datasets because it involves expensive operations, both in the expectation and maximization steps that need to be performed on the entire dataset. Therefore, online schemes are necessary for analysis of large models with latent variables.

Titterington (1984) considered a procedure defined through the iteration

$$\boldsymbol{\theta}_n = \boldsymbol{\theta}_{n-1} + a_n \mathcal{I}_{\mathrm{com}}(\boldsymbol{\theta}_{n-1})^{-1} \nabla \ell_{\mathrm{obs}}(\boldsymbol{y}_n; \boldsymbol{\theta}_{n-1}). \tag{42}$$

This procedure differs only marginally from Sakrison's recursive estimation method (see Sect. 2.2) by using the complete-data information matrix. In the univariate case where the true model parameter is $\theta_\star$, Titterington (1984) applied Fabian's theorem (Fabian 1968) to show that the estimate in (42) satisfies $\sqrt{n}(\theta_n - \theta_\star) \sim \mathcal{N}(0, \mathcal{I}_{\mathrm{com}}(\theta_\star)^{-2} \mathcal{I}_{\mathrm{obs}}(\theta_\star)/(2\mathcal{I}_{\mathrm{obs}}(\theta_\star)\mathcal{I}_{\mathrm{com}}(\theta_\star)^{-1} - 1)$. Thus, as in the traditional full-data EM algorithm, the efficiency of the online method (42) depends on the amount of missing information. Notably, Lange (1995) considered Newton–Raphson iterations for the M-step of the EM algorithm, and derived an online procedure that is similar to (42).

However, procedure (42) is essentially an explicit stochastic gradient method, and thus it may have serious stability and convergence problems, contrary to the desirable numerical stability of EM. In the exponential family model (39), Nowlan (1991) considered one of the first "true" online EM algorithms as follows:

$$\begin{aligned} \boldsymbol{s}_{n+1} &= \boldsymbol{s}_n + \alpha \mathbb{E}\left(\boldsymbol{s}(\boldsymbol{x}_n, \boldsymbol{y}_n; \boldsymbol{\theta}_n) \middle| \boldsymbol{\theta}_n, \boldsymbol{y}_n\right), && \mathbf{E-step} \\ \boldsymbol{\theta}_{n+1} &= \arg\max_{\boldsymbol{\theta}} \ell_{\mathrm{com}}(\boldsymbol{s}_{n+1}; \boldsymbol{\theta}), && \mathbf{M-step} \end{aligned} \tag{43}$$

where $\alpha \in (0, 1)$. In words, the algorithm starts from some initial sufficient statistic $\boldsymbol{s}_0$ and then updates it through a stochastic approximation scheme with a constant-step size α. The maximization step is identical to that of traditional EM. Online EM with decreasing step sizes was later developed by Sato and Ishii (2000) as follows:

$$\begin{aligned} \boldsymbol{s}_{n+1} &= \boldsymbol{s}_n + a_n \left[\mathbb{E}\left(\boldsymbol{s}(\boldsymbol{x}_n, \boldsymbol{y}_n; \boldsymbol{\theta}_n)\middle|\boldsymbol{\theta}_n, \boldsymbol{y}_n\right) - \boldsymbol{s}_n\right], \quad \mathbf{E-step} \\ \boldsymbol{\theta}_{n+1} &= \arg\max_{\boldsymbol{\theta}} \ell_{\text{com}}(\boldsymbol{s}_{n+1}; \boldsymbol{\theta}). \quad \mathbf{M-step} \end{aligned} \tag{44}$$

By the theory of stochastic approximation, procedure (44) will converge to the observed-data maximum-likelihood estimate $\hat{\boldsymbol{\theta}}$. In contrast, procedure (43) will not converge with a constant α, but it will reach a point in the vicinity of $\hat{\boldsymbol{\theta}}$ more rapidly than (44). Further extensions of the aforementioned online EM algorithms have been developed by several authors (Neal and Hinton 1998; Cappé and Moulines 2009). Examples of a growing body of applications of such methods can be found in (Neal and Hinton 1998; Sato and Ishii 2000; Liu et al. 2006; Cappé 2011).

4.2 MCMC sampling

Let $\boldsymbol{\theta}$ be the model parameters of observations $\boldsymbol{Y} = (\boldsymbol{y}_1, \cdots \boldsymbol{y}_N)$, with an assumed prior distribution denoted by $\pi(\boldsymbol{\theta})$. A common task in Bayesian statistics it to sample from the posterior distribution $f(\boldsymbol{\theta}|\boldsymbol{Y}) \propto \pi(\boldsymbol{\theta}) f(\boldsymbol{Y}|\boldsymbol{\theta})$. The Hamiltonian Monte-Carlo (HMC) (Neal 2011) is a method in which auxiliary variables $\boldsymbol{p}$ are introduced to the original variables $\boldsymbol{\theta}$ to improve sampling from $f(\boldsymbol{\theta}|\boldsymbol{Y})$. In the augmented parameter space, we consider a function $H(\boldsymbol{\theta}, \boldsymbol{p}) = U(\boldsymbol{\theta}) + K(\boldsymbol{p}) \in \mathbb{R}^+$, where $U(\boldsymbol{\theta}) = -\log f(\boldsymbol{\theta}|\boldsymbol{Y})$ and $K(\boldsymbol{p}) = (1/2)\boldsymbol{p}^\intercal \boldsymbol{M} \boldsymbol{p}$ with a symmetric positive-definite matrix $\boldsymbol{M}$. Next, we consider the density

$$\begin{aligned} h(\boldsymbol{\theta}, \boldsymbol{p}|\boldsymbol{Y}) &= \exp\{-H(\boldsymbol{\theta}, \boldsymbol{p})\} = \exp\{-U(\boldsymbol{\theta}) - K(\boldsymbol{p})\} \\ &= f(\boldsymbol{\theta}|\boldsymbol{Y}) \times \mathcal{N}(\boldsymbol{p}, \boldsymbol{M}^{-1}). \end{aligned}$$

In this parameterization, the variables $\boldsymbol{p}$ are independent of $\boldsymbol{\theta}$. Assuming some initial state $(\boldsymbol{\theta}_0, \boldsymbol{p}_0)$, HMC sampling proceeds in iterations indexed by $n = 1, \cdots$, as follows:

1. Sample $\boldsymbol{p}^* \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{M}^{-1})$.
2. Using *Hamiltonian dynamics*, compute $(\boldsymbol{\theta}_n, \boldsymbol{p}_n) = \text{ODE}(\boldsymbol{\theta}_{n-1}, p^*)$.
3. Perform a typical Metropolis-Hastings step for the proposed transition $(\boldsymbol{\theta}_{n-1}, \boldsymbol{p}^*) \to (\boldsymbol{\theta}_n, \boldsymbol{p}_n)$ with acceptance probability that is equal to $\min[1, \exp(-H(\boldsymbol{\theta}_n, \boldsymbol{p}_n) + H(\boldsymbol{\theta}_{n-1}, \boldsymbol{p}^*)]$.

Step 2. is the key idea in HMC. The variables $(\boldsymbol{\theta}, \boldsymbol{p})$ can be mapped to a physical system where $\boldsymbol{\theta}$ is the *position* of the system, and $\boldsymbol{p}$ is the *momentum*. The Hamiltonian dynamics refer to a set of ordinary differential equations (ODE) that govern the movement of the system, and thus calculate the future values of $(\boldsymbol{\theta}, \boldsymbol{p})$ given a pair of current values. Being a closed physical system, the *Hamiltonian* of the system is constant. Thus, in Step 3. of HMC it holds $-H(\boldsymbol{\theta}_n, \boldsymbol{p}_n) + H(\boldsymbol{\theta}_{n-1}, \boldsymbol{p}^*) = 0$, and thus the acceptance probability is one.

A special case of HMC, called *Langevin dynamics*, defines the sampling iterations as follows (Girolami and Calderhead 2011):

$$\begin{aligned} &\boldsymbol{\eta}_n \sim \mathcal{N}(\boldsymbol{0}, \epsilon \boldsymbol{I}), \\ &\boldsymbol{\theta}_n = \boldsymbol{\theta}_{n-1} + \frac{\epsilon}{2}(\nabla \log \pi(\boldsymbol{\theta}_{n-1}) + \log f(\boldsymbol{\theta}_{n-1}; \boldsymbol{Y})) + \boldsymbol{\eta}_n. \end{aligned} \tag{45}$$

The sampling procedure (45) follows from HMC by a numerical solution of the ODE method in Step 2. of the algorithm using the *leapfrog method*. Parameter $\epsilon > 0$ determines the size of the leapfrog in the numerical solution of Hamiltonian differential equations.

Welling and Teh (2011) studied a simple modification of Langevin dynamics (45) using a stochastic gradient as follows:

$$\begin{aligned} &\boldsymbol{\eta}_n \sim \mathcal{N}(0, \epsilon_n), \\ &\boldsymbol{\theta}_n = \boldsymbol{\theta}_{n-1} + \frac{\epsilon_n}{2}\Bigg(\nabla \log \pi(\boldsymbol{\theta}_{n-1}) + (N/b) \\ &\qquad \times \sum_{i \in \text{batch}} \nabla \log f(\boldsymbol{y}_i|\boldsymbol{\theta}_{n-1})\Bigg) + \boldsymbol{\eta}_n. \end{aligned} \tag{46}$$

The step sizes ϵ_n satisfy the typical requirements in stochastic approximation i.e., $\sum \epsilon_i = \infty$ and $\sum \epsilon_i^2 < \infty$. Procedure (46) is using stochastic gradients averaged over a *mini-batch* of b samples that are usually employed in SGD to reduce noise in the stochastic gradients. Notably, Sato and Nakagawa (2014) proved that procedure (46) converges to the true posterior $f(\boldsymbol{\theta}|\boldsymbol{Y})$ with an elegant use of stochastic calculus. Sampling through stochastic gradient Langevin dynamics has since generated a lot of significant work in MCMC sampling for very large datasets, and it is still a rapidly expanding research area with contributions from various disciplines (Hoffman et al. 2013; Pillai and Smith 2014; Korattikara et al. 2014).

4.3 Reinforcement learning

Reinforcement learning is the multidisciplinary study of how autonomous agents perceive, learn, and interact with their environment (Bertsekas and Tsitsiklis 1995). Typically, it is assumed that time t proceeds in discrete steps and at every step an *agent* is at state $\boldsymbol{x}_t \in \mathcal{X}$, where $\mathcal{X}$ is some state space. Upon entering a state $\boldsymbol{x}_t$ two things happen. First, an agent receives a probabilistic *reward* $R(\boldsymbol{x}_t) \in \mathbb{R}$, and then takes an *action* $a \in \mathcal{A}$, where $\mathcal{A}$ denotes the action-space. This action is determined by the agent's *policy*, which is a function $\pi : \mathcal{X} \to \mathcal{A}$, thus mapping a state to an action.

Nature then decides a *transition* to state $\boldsymbol{x}_{t+1}$ through a density $p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t)$ that is unknown to the agent.

One important task in reinforcement learning is to estimate the *value function* $V^{\pi}(\boldsymbol{x})$ which quantifies the expected value of a specific state $\boldsymbol{x} \in \mathcal{X}$ for an agent. This is defined as

$$V^{\pi}(\boldsymbol{x}) = \mathbb{E}\left(R(\boldsymbol{x})\right) + \gamma \mathbb{E}\left(R(\boldsymbol{x}_1)\right) + \gamma^2 \mathbb{E}\left(R(\boldsymbol{x}_2)\right) + \cdots, \tag{47}$$

where $\boldsymbol{x}_t$ denotes the state that will be reached starting at $\boldsymbol{x}$ after t transitions, and $\gamma \in (0, 1)$ is a parameter that discounts future rewards. Note that the variation of $R(\boldsymbol{x}_t)$ includes the uncertainty of the state $\boldsymbol{x}_t$ because of the stochasticity in transitions, and the uncertainty from the reward distribution. Thus, $V^{\pi}(\boldsymbol{x})$ admits a recursive definition as follows:

$$V^{\pi}(\boldsymbol{x}) = \mathbb{E}\left(R(\boldsymbol{x})\right) + \gamma \mathbb{E}\left(V^{\pi}(\boldsymbol{x}_1)\right). \tag{48}$$

When the state is a high-dimensional vector, one popular approach is to use the *linear value approximation* $V(\boldsymbol{x}) = \boldsymbol{\theta}_{\star}{}^{\mathsf{T}}\phi(\boldsymbol{x})$, where $\phi(\boldsymbol{x})$ maps a state to *features* in a space with fewer dimensions, and $\boldsymbol{\theta}_{\star}$ is a vector of fixed parameters. If an agent is at state $\boldsymbol{x}_t$, then the recursive equation (48) can be rewritten as

$$\mathbb{E}\left(R(\boldsymbol{x}_t) - (\boldsymbol{\theta}_{\star}{}^{\mathsf{T}}\boldsymbol{\phi}_t - \gamma \boldsymbol{\theta}_{\star}{}^{\mathsf{T}}\boldsymbol{\phi}_{t+1}) \big| \boldsymbol{\phi}_t\right) = 0, \tag{49}$$

where we set $\boldsymbol{\phi}_t = \phi(\boldsymbol{x}_t)$ for notational convenience. Similar to SGD, this suggests a stochastic approximation method to estimate $\boldsymbol{\theta}_{\star}$ through the following iteration:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + a_t \left[R(\boldsymbol{x}_t) - \left(\boldsymbol{\theta}_t^{\mathsf{T}}\boldsymbol{\phi}_t - \gamma \boldsymbol{\theta}_t^{\mathsf{T}}\boldsymbol{\phi}_{t+1}\right)\right]\boldsymbol{\phi}_t, \tag{50}$$

where a_t is a learning rate sequence that satisfies the Robbins–Monro conditions (see Sect. 2.1). Equation (50) is known as the *temporal differences* (TD) learning algorithm (Sutton 1988). Implicit versions of this algorithm have recently emerged in order to solve some of the known stability issues of the classical TD algorithm (Schapire and Warmuth 1996; Li 2008; Wang and Bertsekas 2013; Tamar et al. 2014). For example, Tamar et al. (2014) consider computing the term $\boldsymbol{\theta}_t^{\mathsf{T}}\boldsymbol{\phi}_t$ at the future iterate, and thus the resulting *implicit* TD algorithm is

$$\boldsymbol{\theta}_{t+1} = (\boldsymbol{I} + a_t \boldsymbol{\phi}_t \boldsymbol{\phi}_t^{\mathsf{T}})^{-1}\left[\boldsymbol{\theta}_t + a_t (R(\boldsymbol{x}_t) + \gamma \boldsymbol{\theta}_t^{\mathsf{T}}\boldsymbol{\phi}_{t+1})\boldsymbol{\phi}_t\right]. \tag{51}$$

Similar to implicit SGD, iteration (51) stabilizes the TD iterations. With the advent of online multiagent markets, methods and applications in reinforcement learning have been receiving a renewed stream of research effort (Gosavi 2009).

4.4 Deep learning

Deep learning is the task of estimating parameters of statistical models that can be represented by multiple layers of nonlinear operations, such as neural networks (Bengio 2009). Such models, also referred to as *deep architectures*, consist of *units* that can perform a basic prediction task, and are grouped in layers such that the output of one layer forms the input of another layer that sits directly on top. Furthermore, in most situations the models are augmented with *latent units* that are defined to represent structured quantities of interest, such as edges or shapes in an image.

One basic building block of deep architectures is the Restricted Boltzmann Machine (RBM). The complete-data density for an observation $(\boldsymbol{x}, \boldsymbol{y})$ of the states of hidden and observed input units, respectively, is given by

$$P(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta}) = \frac{\exp\{-\boldsymbol{b}'\boldsymbol{y} - \boldsymbol{c}'\boldsymbol{x} - \boldsymbol{x}'\boldsymbol{W}\boldsymbol{y}\}}{Z(\boldsymbol{\theta})}, \tag{52}$$

where $\boldsymbol{\theta} = (\boldsymbol{b}, \boldsymbol{c}, \boldsymbol{W})$ are the model parameters, and the normalizing constant is $Z(\boldsymbol{\theta}) = \sum_{\boldsymbol{x},\boldsymbol{y}} \exp\{-\boldsymbol{b}'\boldsymbol{y} - \boldsymbol{c}'\boldsymbol{x} - \boldsymbol{x}'\boldsymbol{W}\boldsymbol{y}\}$ (also known as the partition function). Furthermore, the sample spaces for $\boldsymbol{x}$ and $\boldsymbol{y}$ are discrete (e.g., binary) and finite. The observed-data density is thus $P(\boldsymbol{y}; \boldsymbol{\theta}) = \sum_{\boldsymbol{x}} P(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta})$. Let $H(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta}) = \boldsymbol{b}'\boldsymbol{y} + \boldsymbol{c}'\boldsymbol{x} + \boldsymbol{x}'\boldsymbol{W}\boldsymbol{y}$, such that $P(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta}) = \frac{e^{-H(\boldsymbol{x},\boldsymbol{y};\boldsymbol{\theta})}}{Z(\boldsymbol{\theta})}$. Through simple algebra one can obtain the log-likelihood of an observed sample $\boldsymbol{y}^{\text{obs}}$ in the following convenient form:

$$\nabla \ell(\boldsymbol{\theta}; \boldsymbol{y}^{\text{obs}}) = -\left[\mathbb{E}\left(\nabla H(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta})\right) - \mathbb{E}\left(\nabla H(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta}) | \boldsymbol{y}^{\text{obs}}\right)\right]. \tag{53}$$

In practical situations, the data $\boldsymbol{x}, \boldsymbol{y}$ are binary. Therefore, the conditional distribution of the missing data $\boldsymbol{x}|\boldsymbol{y}$ is readily available through the usual logistic regression GLM model, and thus the second term of (53) is easy to sample from. Similarly, $\boldsymbol{y}|\boldsymbol{x}$ is easy to sample from. However, the first term in (53) requires sampling from the joint distribution of the complete-data $(\boldsymbol{x}, \boldsymbol{y})$, which conceptually is easy to sample from using the aforementioned conditionals and a Gibbs sampling scheme (Geman and Geman 1984). However, the state space for both $\boldsymbol{x}$ and $\boldsymbol{y}$ is usually very large, e.g., comprised of thousands or millions of units, and thus a full Gibbs on the joint distribution is usually impossible.

The method of *contrastive divergence* (Hinton 2002; Carreira-Perpinan and Hinton 2005) has been applied for training such models with considerable success. The algorithm proceeds as follows for steps $i = 1, 2, \ldots$:

1. Sample one state $\boldsymbol{y}^{(i)}$ from the empirical distribution of observed states.
2. Sample $\boldsymbol{x}^{(i)}|\boldsymbol{y}^{(i)}$ i.e., the hidden state.

3. Sample $\boldsymbol{y}^{(i,\text{new})}|\boldsymbol{x}^{(i)}$.
4. Sample $\boldsymbol{x}^{(i,\text{new})}|\boldsymbol{y}^{(i,\text{new})}$.
5. Evaluate the gradient (53) using $(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})$ for the second term, and the sample $(\boldsymbol{x}^{(i,\text{new})}, \boldsymbol{y}^{(i,\text{new})})$ for the first term.
6. Update the parameters in $\boldsymbol{\theta}$ using constant-step size SGD and the estimated gradient from Step 4.

In other words, contrastive divergence estimates both terms of (53). This estimation is biased because $(\boldsymbol{x}^{(i,\text{new})}, \boldsymbol{y}^{(i,\text{new})})$ is assumed to be from the full joint distribution of $(\boldsymbol{x}, \boldsymbol{y})$. In fact, contrastive divergence might operate in k steps in which the Steps 3–4 are repeated k times, in an effort to approximate the joint distribution better by letting the chain run longer. Although in theory larger k should approximate the full joint better, it has been observed that $k = 1$ is enough for good performance in many learning tasks (Hinton 2002; Taylor et al. 2006; Salakhutdinov et al. 2007; Bengio 2009; Bengio and Delalleau 2009).

Acknowledgments The authors wish to thank Leon Bottou, Bob Carpenter, David Dunson, Andrew Gelman, Brian Kulis, Xiao-Li Meng, Natesh Pillai, Neil Shephard, Daniel Sussman and Alexander Volfovsky for useful comments and discussion. This research was sponsored, in part, by NSF CAREER award IIS-1149662, ARO MURI award W911NF-11-1-0036, and ONR YIP award N00014-14-1-0485. PT is a Google Fellow in Statistics. EMA is an Alfred P. Sloan Research Fellow.

References

Amari, S.-I.: Natural gradient works efficiently in learning. Neural Comput. **10**(2), 251–276 (1998)

Amari, S.-I., Park, H., Kenji, F.: Adaptive method of realizing natural gradient learning for multilayer perceptrons. Neural Comput. **12**(6), 1399–1409 (2000)

Bather, J.A.: Stochastic Approximation: A Generalisation of the Robbins–Monro Procedure, vol. 89. Cornell University, Mathematical Sciences Institute, New York (1989)

Beck, A., Teboulle, M.: Mirror descent and nonlinear projected subgradient methods for convex optimization. Oper. Res. Lett. **31**(3), 167–175 (2003)

Bengio, Y.: Learning deep architectures for ai. Foundations and trends ®. Mach. Learn. **2**, 1–127 (2009)

Bengio, Y., Delalleau, O.: Justifying and generalizing contrastive divergence. Neural Comput. **21**(6), 1601–1621 (2009)

Benveniste, A., Métivier, M., Priouret, P.: Adaptive Algorithms and Stochastic Approximations. Springer Publishing Company, Incorporated, New York (2012)

Bertsekas, D.P., Tsitsiklis, J.N.: Neuro-dynamic programming: an overview. In: Proceedings of the 34th IEEE Conference on Decision and Control, vol. 1, pp. 560–564 (1995)

Bordes, A., Bottou, L., Gallinari, P.: Sgd-qn: careful quasi-Newton stochastic gradient descent. J. Mach. Learn. Res. **10**, 1737–1754 (2009)

Bottou, L.: Large-scale machine learning with stochastic gradient descent. In: Proceedings of COMPSTAT'2010, pp. 177–186. Springer, New York (2010)

Bottou, L., Le Cun, Y.: On-line learning for very large data sets. Appl. Stoch. Models Bus. Ind. **21**(2), 137–151 (2005)

Bousquet, O., Bottou, L.: The tradeoffs of large scale learning. Adv. Neural Inf. Process. Syst. **20**, 161–168 (2008)

Broyden, C.G.: A class of methods for solving nonlinear simultaneous equations. Math. Comput. **19**, 577–593 (1965)

Cappé, O.: Online em algorithm for hidden Markov models. J. Comput. Graph. Stat. **20**(3), 728–749 (2011)

Cappé, O., Moulines, M.: On-line expectation-maximization algorithm for latent data models. J. R. Stat. Soc. **71**(3), 593–613 (2009)

Carreira-Perpinan, M.A., Hinton, G.E.: On contrastive divergence learning. In: Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics, pp. 33–40. Citeseer (2005)

Cheng, L., Vishwanathan, S.V.N., Schuurmans, D., Wang, S., Caelli, T.: Implicit online learning with kernels. In: Proceedings of the 2006 Conference Advances in Neural Information Processing Systems 19, vol. 19, p. 249. MIT Press, Cambridge, 2007

Chung, K.L.: On a stochastic approximation method. Ann. Math. Stat. **25**, 463–483 (1954)

Dempster, A., Laird, N., Rubin, D.: Maximum likelihood from incomplete data via the EM algorithm. J. R. Stat. Soc. Ser. B **39**, 1–38 (1977)

Duchi, J., Hazan, E., Singer, Y.: Adaptive subgradient methods for online learning and stochastic optimization. J. Mach. Learn. Res. **999999**, 2121–2159 (2011)

Dupuis, P., Simha, R.: On sampling controlled stochastic approximation. IEEE Trans. Autom. Control **36**(8), 915–924 (1991)

El Karoui, N.: Spectrum estimation for large dimensional covariance matrices using random matrix theory. Ann. Stat. **36**, 2757–2790 (2008)

Fabian, V.: On asymptotic normality in stochastic approximation. Ann. Math. Stat. **39**, 1327–1332 (1968)

Fabian, V.: Asymptotically efficient stochastic approximation; the RM case. Ann. Stat. **1**, 486–495 (1973)

Fisher, R.A.: On the mathematical foundations of theoretical statistics. Philos. Trans. R. Soc. Lond. Ser. A **222**, 309–368 (1922)

Fisher, R.A.: Statistical Methods for Research Workers. Oliver and Boyd, Edinburgh (1925a)

Fisher, R.A.: Theory of statistical estimation. In: Mathematical Proceedings of the Cambridge Philosophical Society, vol. 22, pp. 700–725. Cambridge University Press, Cambridge (1925b)

Geman, S., Geman, D.: Stochastic relaxation, gibbs distributions, and the Bayesian restoration of images. IEEE Trans. Pattern Anal. Mach. Intell. **6**, 721–741 (1984)

George, A.P., Powell, W.B.: Adaptive stepsizes for recursive estimation with applications in approximate dynamic programming. Machine Learn. **65**(1), 167–198 (2006)

Girolami, M.: Riemann manifold Langevin and Hamiltonian Monte Carlo methods. J. R. Stat. Soc. Ser. B **73**(2), 123–214 (2011)

Gosavi, A.: Reinforcement learning: a tutorial survey and recent advances. INFORMS J. Comput. **21**(2), 178–192 (2009)

Green, P.J.: Iteratively reweighted least squares for maximum likelihood estimation, and some robust and resistant alternatives. J. R. Stat. Soc. Ser. B **46**, 149–192 (1984)

Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2nd edn. Springer, New York (2011)

Hennig, P., Kiefel, M.: Quasi-Newton methods: a new direction. J. Mach. Learn. Res. **14**(1), 843–865 (2013)

Hinton, G.E.: Training products of experts by minimizing contrastive divergence. Neural Comput. **14**(8), 1771–1800 (2002)

Hoffman, M.D., Blei, D.M., Wang, C., Paisley, J.: Stochastic variational inference. J. Mach. Learn. Res. **14**(1), 1303–1347 (2013)

Huber, P.J., et al.: Robust estimation of a location parameter. Ann. Math. Stat. **35**(1), 73–101 (1964)

Huber, P.J.: Robust Statistics. Springer, New York (2011)
Johnson, R., Zhang, T.: Accelerating stochastic gradient descent using predictive variance reduction. Adv. Neural Inf. Process. Syst. **26**, 315–323 (2013)
Kivinen, J., Warmuth, M.K.: Additive versus exponentiated gradient updates for linear prediction. In: Proceedings of the Twenty-Seventh Annual ACM Symposium on Theory of Computing, pp. 209–218
Kivinen, J., Warmuth, M.K., Hassibi, B.: The p-norm generalization of the lms algorithm for adaptive filtering. IEEE Trans. Signal Process. **54**(5), 1782–1793 (2006)
Korattikara, A., Chen, Y., Welling, M.: Austerity in mcmc land: cutting the metropolis-hastings budget. In: Proceedings of the 31st International Conference on Machine Learning, pp. 181–189 (2014)
Kulis, B., Bartlett, P.L.: Implicit online learning. In: Proceedings of the 27th International Conference on Machine Learning (ICML-10), pp. 575–582 (2010)
Lai, T.L., Robbins, H.: Adaptive design and stochastic approximation. Ann. Stat. **7**, 1196–1221 (1979)
Lange, K.: A gradient algorithm locally equivalent to the EM algorithm. J. R. Stat. Soc. Ser. B **57**, 425–437 (1995)
Lange, K.: Numerical Analysis for Statisticians. Springer, New York (2010)
Le, C., Bottou Yann, L., Bottou, L.: Large scale online learning. Adv. Neural Inf. Process. Syst. **16**, 217 (2004)
Lehmann, E.H., Casella, G.: Theory of Point Estimation, 2nd edn. Springer, New York (2003)
Li, L.: A worst-case comparison between temporal difference and residual gradient with linear function approximation. In: Proceedings of the 25th International Conference on Machine Learning, ACM, pp. 560–567
Liu, Z., Almhana, J., Choulakian, V., McGorman, R.: Online em algorithm for mixture with application to internet traffic modeling. Comput. Stat. Data Anal. **50**(4), 1052–1071 (2006)
Ljung, L., Pflug, G., Walk, H.: Stochastic Approximation and Optimization of Random Systems, vol. 17. Springer, New York (1992)
Martin, R.D., Masreliez, C.: Robust estimation via stochastic approximation. IEEE Trans. Inf. Theory **21**(3), 263–271 (1975)
Murata, N.: A Statistical Study of On-line Learning. Online Learning and Neural Networks. Cambridge University Press, Cambridge (1998)
Nagumo, J.-I., Noda, A.: A learning method for system identification. IEEE Trans. Autom. Control **12**(3), 282–287 (1967)
National Research Council: Frontiers in Massive Data Analysis. The National Academies Press, Washington, DC (2013)
Neal, R.M., Hinton, G.E.: A view of the em algorithm that justifies incremental, sparse, and other variants. In: Learning in Graphical Models, pp. 355–368. Springer, New York (1998)
Neal, R.: Mcmc Using Hamiltonian Dynamics. Handbook of Markov Chain Monte Carlo 2 (2011)
Nemirovski, A.S., Yudin, D.B.: Problem Complexity and Method Efficiency in Optimization. Wiley, Chichester (1983)
Nemirovski, A., Juditsky, A., Lan, G., Shapiro, A.: Robust stochastic approximation approach to stochastic programming. SIAM J. Optim. **19**(4), 1574–1609 (2009)
Nevelson, M.B., Khasminskiĭ, R.Z.: Stochastic Approximation and Recursive Estimation, vol. 47. American Mathematical Society, Providence (1973)
Nowlan, S.J.: Soft Competitive Adaptation: Neural Network Learning Algorithms Based on Fitting Statistical Mixtures. Carnegie Mellon University, Pittsburgh (1991)
Parikh, N., Boyd, S.: Proximal algorithms. Found. Trends Optim. **1**(3), 123–231 (2013)
Pillai, N.S., Smith, A.: Ergodicity of approximate mcmc chains with applications to large data sets. arXiv preprint http://arxiv.org/abs/1405.0182 (2014)
Polyak, B.T., Tsypkin, Y.Z.: Adaptive algorithms of estimation (convergence, optimality, stability). Autom. Remote Control **3**, 74–84 (1979)
Polyak, B.T., Juditsky, A.B.: Acceleration of stochastic approximation by averaging. SIAM J. Control Optim. **30**(4), 838–855 (1992)
Robbins, H., Monro, S.: A stochastic approximation method. Ann. Math. Stat. **22**, 400–407 (1951)
Rockafellar, R.T.: Monotone operators and the proximal point algorithm. SIAM J. Control Optim. **14**(5), 877–898 (1976)
Rosasco, L., Villa, S., Công Vũ, B.: Convergence of stochastic proximal gradient algorithm. arXiv preprint http://arxiv.org/abs/1403.5074, 2014
Ruppert, D.: Efficient estimations from a slowly convergent robbins-monro process. Technical report, Cornell University Operations Research and Industrial Engineering (1988)
Ryu, E.K., Boyd, S.: Stochastic proximal iteration: a non-asymptotic improvement upon stochastic gradient descent. Working paper. http://web.stanford.edu/~eryu/papers/spi.pdf (2014)
Sacks, J.: Asymptotic distribution of stochastic approximation procedures. Ann. Math. Stat. **29**(2), 373–405 (1958)
Sakrison, D.J.: Efficient recursive estimation; application to estimating the parameters of a covariance function. Int. J. Eng. Sci. **3**(4), 461–483 (1965)
Salakhutdinov, R., Mnih, A., Hinton, G.: Restricted boltzmann machines for collaborative filtering. In: Proceedings of the 24th International Conference on Machine Learning, ACM, pp. 791–798 (2007)
Sato, M.-A., Ishii, S.: On-line em algorithm for the normalized Gaussian network. Neural Comput. **12**(2), 407–432 (2000)
Sato, I., Nakagawa, H.: Approximation analysis of stochastic gradient langevin dynamics by using Fokker-Planck equation and ito process. JMLR W&CP **32**(1), 982–990 (2014)
Schapire, R.E., Warmuth, M.K.: On the worst-case analysis of temporal-difference learning algorithms. Mach. Learn. **22**(1–3), 95–121 (1996)
Schaul, T., Zhang, S., LeCun, Y.: No more pesky learning rates. arXiv preprint. http://arxiv.org/abs/1206.1106, 2012
Schraudolph, N.N., Yu, J., Günter, S.: A stochastic quasi-Newton method for online convex optimization. In: Meila M., Shen X. (eds.) Proceedings of the 11th International Conference on Artificial Intelligence and Statistics (AISTATS), vol. 2, pp. 436–443. San Juan, Puerto Rico (2007)
Slock, D.T.M.: On the convergence behavior of the LMS and the normalized LMS algorithms. IEEE Trans. Signal Process. **41**(9), 2811–2825 (1993)
Sutton, R.S.: Learning to predict by the methods of temporal differences. Mach. Learn. **3**(1), 9–44 (1988)
Tamar, A., Toulis, P., Mannor, S., Airoldi, E.: Implicit temporal differences. In: Neural Information Processing Systems, Workshop on Large-Scale Reinforcement Learning (2014)
Taylor, G.W., Hinton, G.E., Roweis, S.T.: Modeling human motion using binary latent variables. Adv. Neural Inf. Process. Syst. **19**, 1345–1352 (2006)
Titterington, M.D.: Recursive parameter estimation using incomplete data. J. R. Stat. Soc. Ser. B **46**, 257–267 (1984)
Toulis, P., Airoldi, E.M.: Implicit stochastic gradient descent for principled estimation with large datasets. arXiv preprint http://arxiv.org/abs/1408.2923, 2014
Toulis, P., Airoldi, E., Rennie, J.: Statistical analysis of stochastic gradient methods for generalized linear models. JMLR W&CP **32**(1), 667–675 (2014)

Venter, J.H.: An extension of the robbins-monro procedur. Ann. Math. Stat. **38**, 181–190 (1967)

Wang, C., Chen, X., Smola, A., Xing, E.: Variance reduction for stochastic gradient optimization. Adv. Neural Inf. Process. Syst. **26**, 181–189 (2013)

Wang, M., Bertsekas, D.P.: Stabilization of stochastic iterative methods for singular and nearly singular linear systems. Math. Oper. Res. **39**(1), 1–30 (2013)

Wei, C.Z.: Multivariate adaptive stochastic approximation. Ann. Stat. **3**, 1115–1130 (1987)

Welling, M., Teh, Y.W.: Bayesian learning via stochastic gradient langevin dynamics. In: Proceedings of the 28th International Conference on Machine Learning (ICML-11), pp. 681–688 (2011)

Xu, W.: Towards optimal one pass large scale learning with averaged stochastic gradient descent. arXiv preprint http://arxiv.org/abs/1107.2490, 2011

Younes, L.: On the convergence of markovian stochastic algorithms with rapidly decreasing ergodicity rates. Stochastics **65**(3–4), 177–228 (1999)

Zhang, T.: Solving large scale linear prediction problems using stochastic gradient descent algorithms. In: Proceedings of the Twenty-First International Conference on Machine Learning, ACM, p. 116 (2004)