Rapid Stochastic Gradient Descent
Accelerating Machine Learning

Nicol N. Schraudolph

Statistical Machine Learning Program
www.nicta.com.au
Overview

1. Why Stochastic Gradient?
2. Stochastic Meta-Descent (SMD)
   - Derivation and Algorithm
   - Properties and Benchmark Results
   - Applications and Ongoing Work
3. Summary and Outlook
The flood of information caused by

- plentiful, affordable sensors (such as webcams)
- ever-increasing networking of these sensors

overwhelms our processing ability in, e.g.,

- **science** - pulsar survey at Arecibo: 1 TB/day
- **business** - Dell website: over 100 page requests/sec
- **security** - London: over 500’000 security cameras

We need intelligent, adaptive filters to cope!
A Challenge for ML

Coping with the info glut requires ML alg.s for

- large, complex, nonlinear models
  millions of degrees of freedom
- large volumes of low-quality data
  noisy, correlated, non-stationary, outliers
- efficient real-time, online adaptation
  no fixed training set, life-long learning

Current ML techniques have difficulty with this.
Online Learning Paradigm

classical optimization:

- iterative optimizer
- objective fn.
- training data set
- nested loops!

online learning:

- online optimizer
- training data stream
- (aka adaptive filtering, stochastic approximation, ...)

(aka adaptive filtering, stochastic approximation, ...)

Stochastic Approximation

Classical formulation of optimization problem:

\[ \theta^* = \arg\min_{\theta} : \quad E_x[J(\theta, x)] \approx \frac{1}{|X|} \sum_{x_i \in X} J(\theta, x_i) \]

- inefficient for large data sets \( X \)
- inappropriate for never-ending, potentially non-stationary data streams

\[ \Rightarrow \text{must resort to stochastic approximation:} \]

\[ \theta_{t+1} \approx \arg\min_{\theta} J(\theta_t, x_t) \quad (t = 0, 1, 2, \ldots) \]
The Key Problem

- **online, scalable**
- Optimization algorithms:
  - evolutionary algorithms
  - gradient descent
  - conjugate gradient
  - quasi-Newton
  - Kalman Filter
  - Levenberg Marquardt

---

**cost per iteration**

- $O(1)$
- $O(n)$
- $O(n^2)$
- $O(n^3)$

**convergence speed**
The Key Problem

Stochastic approximation breaks many optimizers:
- conjugate directions break down due to noise
- line minimizations (CG, quasi-Newton) inaccurate
- Newton, Levenberg-Marquardt, Kalman filter - too expensive for large-scale problems

This only leaves
- evolutionary alg.s - very inefficient (don’t use gradient)
- simple gradient descent - can be slow to converge
Gain Vector Adaptation

Given stochastic gradient \( g_t := \partial_{\theta} J(\theta_t, x_t) \), adapt \( \theta \) by gradient descent with gain vector \( \eta \):

\[
\theta_{t+1} = \theta_t - \eta_t \cdot g_t
\]

Key idea: simultaneously adapt \( \eta \) by exponentiated gradient:

\[
\ln \eta_t = \ln \eta_{t-1} - \mu \partial_{\ln \eta} J(\theta_t, x_t)
\]

\[
\eta_t = \eta_{t-1} \cdot \exp(-\mu \partial_{\theta} J(\theta_t, x_t) \cdot \partial_{\ln \eta} \theta_t)
\]

\[
\approx \eta_{t-1} \cdot \max\left(\frac{1}{2}, 1 - \mu g_t \cdot v_t\right)
\]
Conventionally, \( v_{t+1} := \partial_{\ln \eta_t} \theta_{t+1} = -\eta_t \cdot g_t \)

(recall that \( \theta_{t+1} = \theta_t - \eta_t \cdot g_t \))

giving \( \eta_t = \eta_{t-1} \cdot \max\left(\frac{1}{2}, 1 + \mu \eta_{t-1} \cdot g_{t-1} \cdot g_t\right) \)

\( \Rightarrow \) adaptation of \( \eta \) driven by autocorrelation of \( g \):
To capture long-term dependence of $\theta$ on $\eta$:

\[ v_{t+1} := \sum_{i=0}^{t} \lambda^i \frac{\partial \theta_{t+1}}{\partial \ln \eta_{t-i}} \]

define decay $0 \leq \lambda \leq 1$ (free parameter)
SMD’s $\mathbf{v}$-update

\[ \mathbf{v}_{t+1} = \lambda \mathbf{v}_t - \eta_t \cdot (\mathbf{g}_t + \lambda \mathbf{H}_t \mathbf{v}_t) \]

- we obtain a simple iterative update for $\mathbf{v}$
- correct smoothing over correlated input signals
- involves implicit Hessian-vector ($\mathbf{H} \mathbf{v}$) product
  - can be computed as efficiently as 2-3 gradient eval.s
  - can be done automatically via algorithmic differentiation
Fixpoint of $v$

The fixpoint of

$$v_{t+1} = \lambda v_t - \eta_t \cdot (g_t + \lambda H_t v_t)$$

is a Levenberg-Marquardt style gradient step:

$$v \rightarrow -[\lambda H + (1 - \lambda) \text{diag}(\eta)^{-1}]^{-1} g$$

- $v$ is too noisy to use directly; SMD achieves stability by means of the double integration $v \rightarrow \eta \rightarrow \theta$
- $v \cdot g$ is well-behaved (self-normalizing property)
- SMD uses Gauss-Newton approximation of $H$
Four Regions Benchmark

Compare simple stochastic gradient (SGD), conventional gain vector adaptation (ALAP), stochastic meta-descent (SMD), and a global extended Kalman filter (GEKF).
Benchmark: Convergence

The imagination driving Australia’s ICT future.
## Computational Cost

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>storage weight</th>
<th>flops update</th>
<th>CPU ms pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>SGD</td>
<td>1</td>
<td>6</td>
<td>0.5</td>
</tr>
<tr>
<td>SMD</td>
<td>3</td>
<td>18</td>
<td>1.0</td>
</tr>
<tr>
<td>ALAP</td>
<td>4</td>
<td>18</td>
<td>1.0</td>
</tr>
<tr>
<td>GEKF</td>
<td>&gt;90</td>
<td>&gt;1500</td>
<td>40</td>
</tr>
</tbody>
</table>
Benchmark: CPU Usage
Autocorrelated Data

- i.i.d. uniform
- Sobol
- Brownian

Statistical Machine Learning Program

www.nicta.com.au
Comparison to CG

Conjugate Gradient
- deterministic
- (1000 pts)
- overfits

SMD
- stochastic
- (5 pts/iteration)
- converges

Deterministic (1000 pts) vs. Stochastic (1000 pts/iteration) vs. Stochastic (5 pts/iteration)
Application: Turbulent Flow

(PhD thesis of M. Milano, Inst. of Computational Science, ETH Zürich)

original flow
(75,000 d.o.f.)

linear PCA
(160 p.c.)

neural network
(160 nonlinear p.c.)
Turbulent Flow Model

Very high-dimensional optimization problem:
- 15 neural networks, each about 180,000 parameters
- The generic model has over 20 million parameters!

Here SMD:
- Outperformed Matlab toolbox
- Was able to train generic model
Application: Hand Tracking

(PhD thesis of M. Bray, Computer Vision Lab, ETH Zürich)

- detailed hand model (10k vertices, 26 d.o.f.)
- randomly sample a few points on model surface
- project them to image
- compare with camera image at these points
- SMD uses resulting stochastic gradient to adjust model
Hand Tracking with SMD

state of the art:
Annealed Particle Filter
(114 sec/frame)

our algorithm:
Stochastic Meta-Descent
(3 sec/frame)
Hand Tracking: Results

Through stochastic sampling, SMD achieves:
- 40x speedup over state of the art (3 vs. 114 s/frame)
- better tracking: noise helps escape local minima
- robustness wrt. clutter, shadows, occlusions, ...

Ongoing work at NICTA:
- use multiple, ordinary (even cheap) video cameras
- simultaneous real-time tracking of hands, face & body
SMD for Online SVM

Online SVM aka NORMA (Kivinen, Smola, Williamson 2004):

- online kernel method
- stochastic gradient in expansion coefficients
- employs scalar gain $\eta$

Application of SMD:

- $v$ is function in RKHS
- $\langle g, v \rangle$ can be maintained incrementally in $O(n)$

NIPS’05 workshop (large-scale kernel machines)
More applications of SMD:
- policy gradient reinforcement learning \((\text{NIPS’05})\)
- generalized Hebbian algorithm for Kernel PCA
- parameter estimation in conditional random fields

Also working on:
- SMD convergence and stability analysis
- further refinement of the algorithm
- other ways to accelerate stochastic gradient
Summary:
- data-rich ML problems need stochastic approximation
- classical gradient methods are not up to the task
- SMD: excellent gain adaptation for stochastic gradient
  \((Hv \text{ product gives cheap second-order information})\)

Outlook:
- increasing demand for stochastic gradient methods
- SMD can greatly accelerate stochastic gradient