Optimization Techniques for Learning and Data Analysis

Stephen Wright

University of Wisconsin-Madison

IPAM Summer School, July 2015

Outline

I. Background: Big Data and Optimization.

II. Sketch some canonical formulations of data analysis / machine learning problems as optimization problems.

III. Optimization toolbox: Optimization techniques to formulate and solve data analysis problems as optimization problems.

(Example of a *randomized asynchronous algorithm: Kaczmarz.*)

Big Data

Much excitement (hype?) around big data.

> *(Big data) opens the door to a new approach to understanding the world and making decisions. (NYT, 11 Feb 2013)*

Some application areas:

- Speech, language, text processing (speech recognition systems).
- Image and video processing (denoising / deblurring, medical imaging, computer vision).
- Biology and bioinformatics (identify risk factors for diseases).
- Feature identification in geographical and astronomical images.
- Online advertising.
- Social network analysis.

Definitions

Data Analysis: Extraction of knowledge from data.

Machine Learning: Learn from data to make predictions about other (similar) data.

Highly interdisciplinary areas, drawing on

- statistics,
- information theory,
- signal processing,
- computer science (artificial intelligence, databases, architecture, systems, parallel processing),
- optimization,
- application-specific expertise.

Optimization is also useful in turning the knowledge into decisions.

Regression and Classification

Given many items of data a_i and the outputs y_i associated with some items, can we learn a function ϕ that maps the data to its output: $y_i \approx \phi(a_i)$?

Why? ϕ can be applied to future data a, to predict output $\phi(a)$.

Formulate as an optimization problem by

- parametrizing the function ϕ;
- applying statistical principles that relate a_i to y_i, e.g. express the likelihood of outputs y_i, given inputs a_i and the parameters of ϕ — then maximize this likelihood.

Regression / classification problems (optimization!):

- Least squares regression;
- Robust regression (ℓ_1, Huber);
- Logistic regression;
- Support vector machines (SVM) (structured QP and LP).

Data Representation

Data in its raw form is often difficult to work with. Often need to transform it, to allow more effective and tractable learning / analysis.

Kernels: Implicitly apply a nonlinear transformation to data vectors a_i prior to classification, regression. Allows more powerful classification (e.g. nonlinear boundaries between classes).

Deep Learning: transform data by passing it though a neural network.

- The transformed data may be easier to classify.
- Optimization needed to find the best weights in the neural network.

Express data using a basis of fundamental objects called atoms, where "low dimensional structure" = "few atoms."

- The basis can be predefined, or built up during the computation.

Low-Dimensional Structure

Data items exist in a high-dimensional ambient space. The knowledge we seek often forms a low-dimensional structure in this space.

- Find a few base pairs in a genome that indicate risk of a disease.
- Find a particular function of the pixel intensities in an image of a digit, that makes it easy to discriminate among digits 0 through 9.
- Complex electromagnetic signals often contain just a few frequencies.
- A graph may contain just a few significant structures (e.g. cliques).

Two key issues in low-dimensional structure identification:

- **data representation** (see above).
- **tractable formulations / algorithms**.

Tractable Formulations and Efficient Algorithms

Finding low-dimensional structures is essentially intractable. But in many interesting cases, tractable formulations are possible. Example:

> *Given A and b, find $x \in R^n$ with few nonzeros that approximately minimizes $\|Ax - b\|_2^2$.*

Generally, need to look at all $\binom{n}{k}$ possibilities.

But compressed sensing has shown that for some matrices A, we can solve it as a convex optimization problem:

$$\min_x \frac{1}{2}\|Ax - b\|_2^2 + \tau\|x\|_1 \qquad \text{for some } \tau > 0.$$

The ℓ_1 norm is a regularization function that induces desired structure in x — in this case, sparsity in x.

Sparse optimization is the study of regularized formulations and algorithms.

Other Optimization Issues in Data Analysis

- Objective functions have a simple form. "Let the data define the model." (There's a view that less sophisticated models are needed when data is abundant.)
- Low-accuracy solutions are good enough! The objective is an approximation to some unknown underlying objective; precise minimization would be "overfitting" the data.
- Optimization formulations contain scalar parameters that balance data-fitting with desired structure. Need to tune these parameters.
- Data scientists love to know theoretical complexity of algorithms — convergence in terms of iteration count t and data dimension n.

II. Canonical Formulations

- Linear regression
- + variable selection (LASSO)
- Support vector machines
- Logistic regression
- Matrix completion
- Deep belief networks
- Image processing
- Data assimilation.

Linear Regression

Given a set of feature vectors $a_i \in \mathbb{R}^n$ and outcomes b_i, $i = 1, 2, \ldots, m$, find weights x that predict the outcome accurately: $a_i^T x \approx b_i$.

Least Squares: Under certain assumptions on measurement error / noise, can find a suitable x by solving a least squares problem

$$\min_x \frac{1}{2}\|Ax - b\|_2^2 = \frac{1}{2}\sum_{i=1}^m (a_i^T x - b_i)^2$$

where the rows of A are a_i^T, $i = 1, 2, \ldots, m$.

Robust Regression: Can replace the sum-of-squares with loss functions that are less sensitive to outliers. Objectives are still separable, one term per data element.

$$\ell_1: \quad \min_x \|Ax - b\|_1 = \sum_{i=1}^m |a_i^T x - b_i|,$$

$$\text{Huber:} \quad \min_x \sum_{i=1}^m h(a_i^T x - b_i), \quad (h \text{ is hybrid of } \|\cdot\|_2^2 \text{ and } \|\cdot\|_1).$$

Feature Selection and Compressed Sensing

Can modify least-squares for feature selection by adding a LASSO regularizer (Tibshirani, 1996):

$$\textbf{LASSO:} \quad \min_x \frac{1}{2}\|Ax - b\|_2^2 + \lambda\|x\|_1,$$

for some parameter $\lambda > 0$. This identifies an approximate minimizer of the least-squares loss with few nonzeros (sparse).

Nonconvex regularizers are sometimes used in place of $\|x\|_1$, for example, SCAD and MCP. These yield unbiased solutions — but you need to solve a nonconvex problem.

In compressed sensing, A has more columns than rows, has certain "restricted isometry" or "incoherence" properties, and is known to have a nearly-sparse optimal solution.

Support Vector Classification

Given data vectors $a_i \in \mathbb{R}^n$, for $i = 1, 2, \ldots, m$ and labels $y_i = \pm 1$ to indicate the class to which a_i belongs.

Seek z such that (usually) we have

$$a_i^T z \geq 1 \text{ when } y_i = +1 \text{ and } a_i^T z \leq -1 \text{ when } y_i = -1.$$

SVM with hinge loss to penalize misclassifications. Objective is separable:

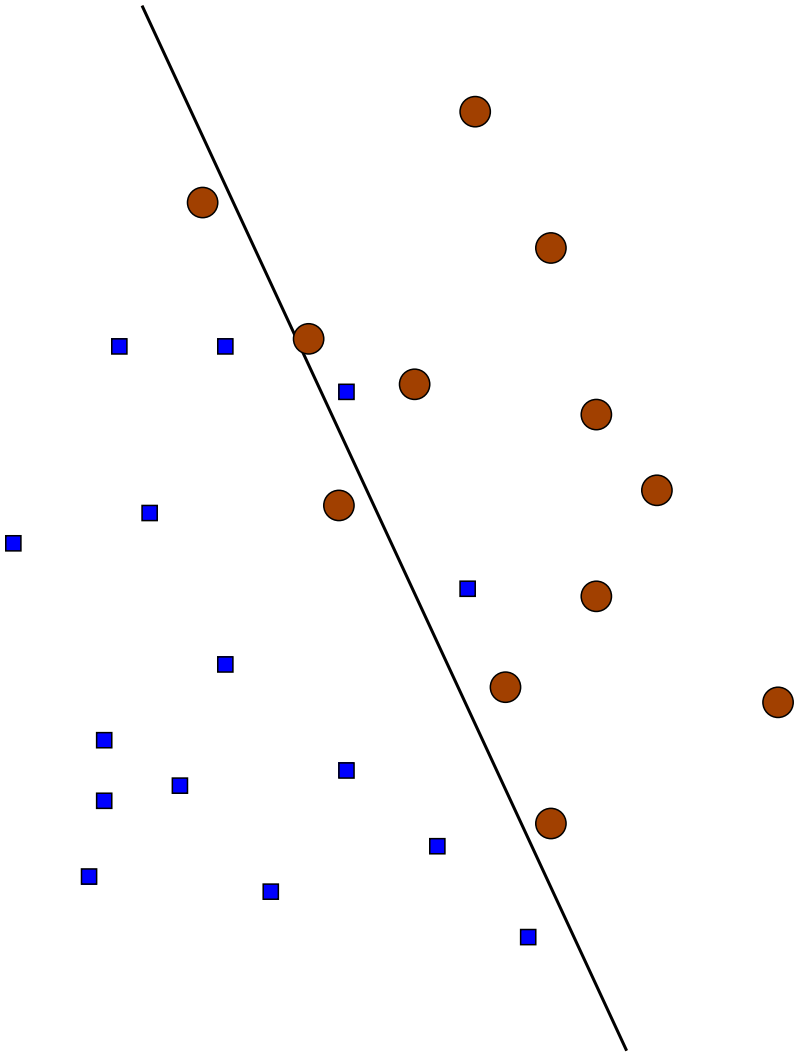
$$f(z) = C \sum_{i=1}^{m} \max(1 - y_i(z^T a_i), 0) + \frac{1}{2}\|z\|^2,$$

where $C > 0$ is a parameter. Define $K_{ij} = y_i y_j a_i^T a_j$ for dual:

$$\min_{\alpha} \frac{1}{2}\alpha^T K \alpha - \mathbf{1}^T \alpha \quad \text{subject to } 0 \leq \alpha \leq C\mathbf{1}.$$

Extends to nonlinear kernel: $K_{ij} := y_i y_j k(a_i, a_j)$ for kernel function $k(\cdot, \cdot)$. Lift then Classify. (Boser et al., 1992; Vapnik, 1999)

Linear SVM



(Regularized) Logistic Regression

Seek odds function parametrized by $z \in \mathbb{R}^n$:

$$p_+(a; z) := (1 + e^{z^T a})^{-1}, \quad p_-(a; z) := 1 - p_+(a; z),$$

choosing z so that $p_+(a_i; z) \approx 1$ when $y_i = +1$ and $p_-(a_i; z) \approx 1$ when $y_i = -1$. Maximize the negative log likelihood function $\mathcal{L}(z)$:

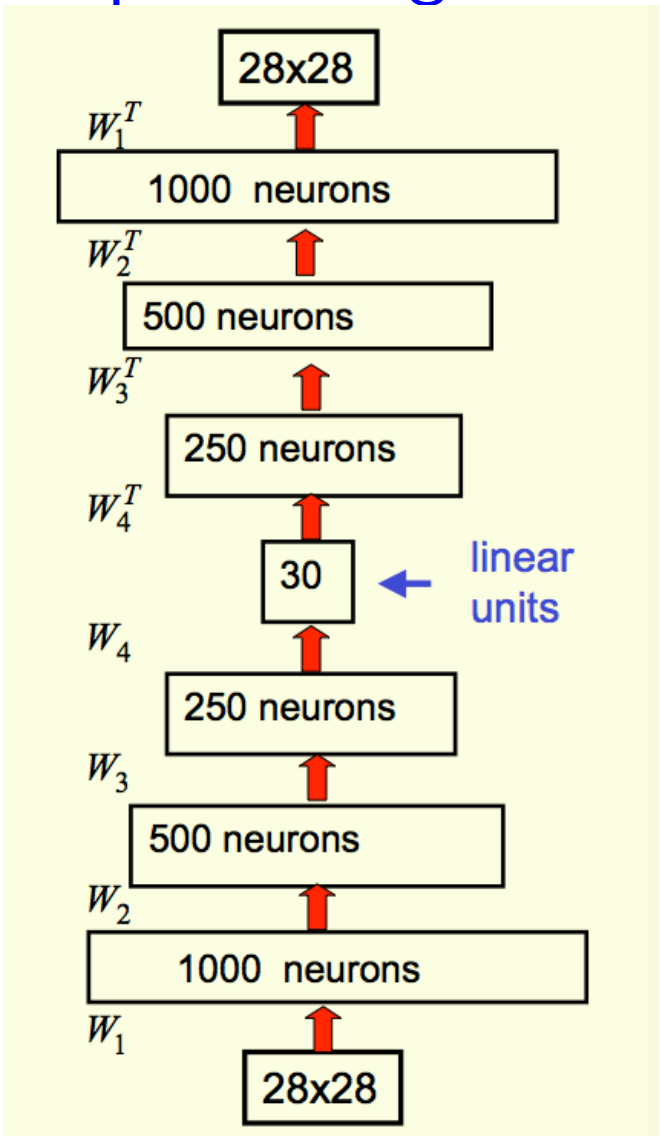
$$\mathcal{L}(z) = -\frac{1}{m} \left[\sum_{y_i=-1} \log p_-(a_i; z) + \sum_{y_i=1} \log p_+(a_i; z) \right]$$

Add regularizer $\lambda \|z\|_1$ to select features.

M **classes**: $y_{ij} = 1$ if data point i is in class j; $y_{ij} = 0$ otherwise. $z_{[j]}$ is the subvector of z for class j.

$$f(z) = -\frac{1}{N} \sum_{i=1}^{N} \left[\sum_{j=1}^{M} y_{ij} (z_{[j]}^T a_i) - \log \left(\sum_{j=1}^{M} \exp(z_{[j]}^T a_i) \right) \right].$$

Deep Learning

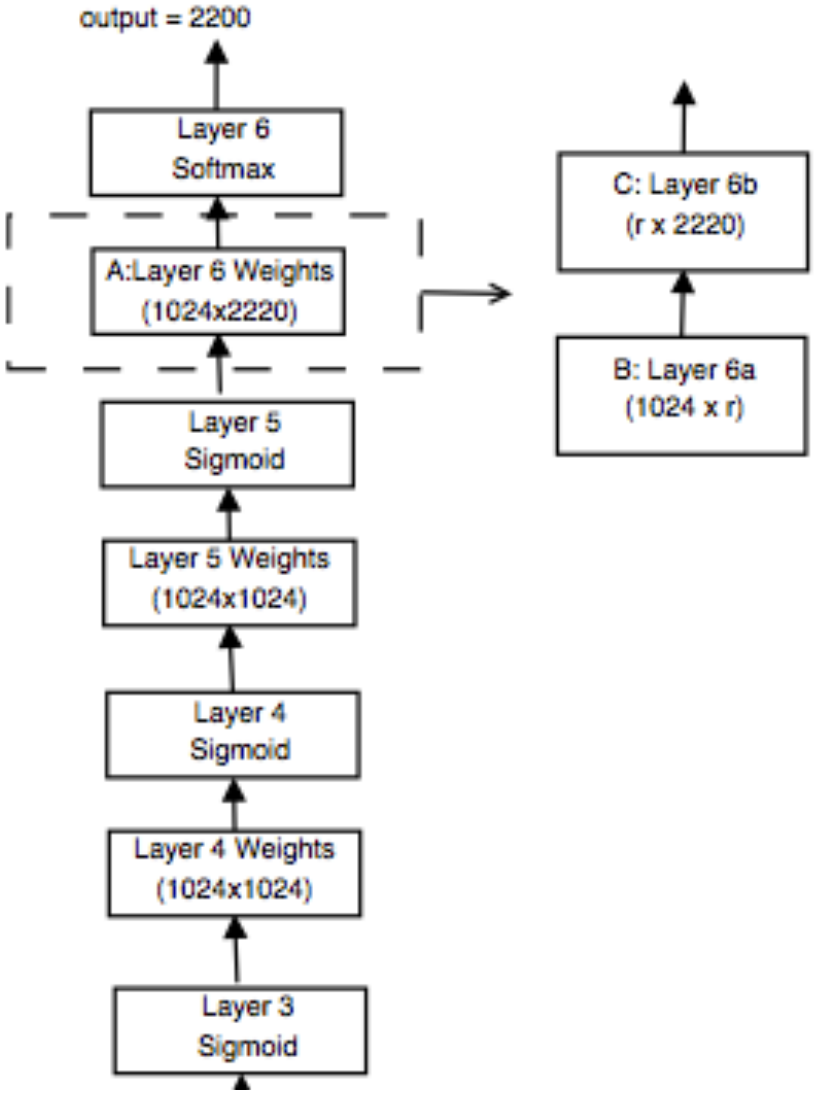


Deep Belief Nets / Neural Nets transform feature vectors prior to classification.

Example of a deep belief network for autoencoding (Hinton, 2007). Output (at top) depends on input (at bottom) of an image with 28×28 pixels. The unknowns are parameters of the matrices W_1, W_2, W_3, W_4.

Nonlinear, nonconvex.

Deep Learning in Speech Processing



Sainath et al. (2013) Break a stream of audio data into phonemes and aim to learn how to identify them from a labelled sample. May use context (phonemes before and after).

Every second layer has $\approx 10^3$ inputs and outputs; the parameter is the transformation matrix from input to output ($\approx 10^6$ parameters).

Deep Learning

Output of a neural network can form the input to a classifier (e.g. SVM, or something simpler, like a max of the output features).

Objectives in learning problems based on neural nets are

- separable: objective is composed of terms that each depend on one item of data (e.g. one utterance, one character, one image) and possibly its neighbors in space or time.
- nonlinear, nonconvex: each layer is simple (linear transformation, sigmoid, softmax), but their composition is not.
- possibly regularized with terms that impose structure. e.g. phoneme class depends on sounds that came before and after.

Matrix Completion

Seek a matrix $X \in \mathbb{R}^{m \times n}$ with some desired structure (e.g. low rank) that matches certain observations, possibly noisy.

$$\min_X \frac{1}{2}\|\mathcal{A}(X) - b\|_2^2 + \lambda\psi(X),$$

where $\mathcal{A}(X)$ is a linear mapping of the components of X (e.g. observations of certain elements of X).

Setting ψ as the nuclear norm (sum of singular values) promotes low rank (in the same way as $\|x\|_1$ tends to promote sparsity of a vector x).

Can impose other structures, e.g. X is the sum of sparse matrix and a low-rank matrix. (Element-wise 1-norm $\|X\|_1$ is useful for sparsity.)

Used in recommender systems, e.g. Netflix, Amazon.

(Recht et al., 2010)

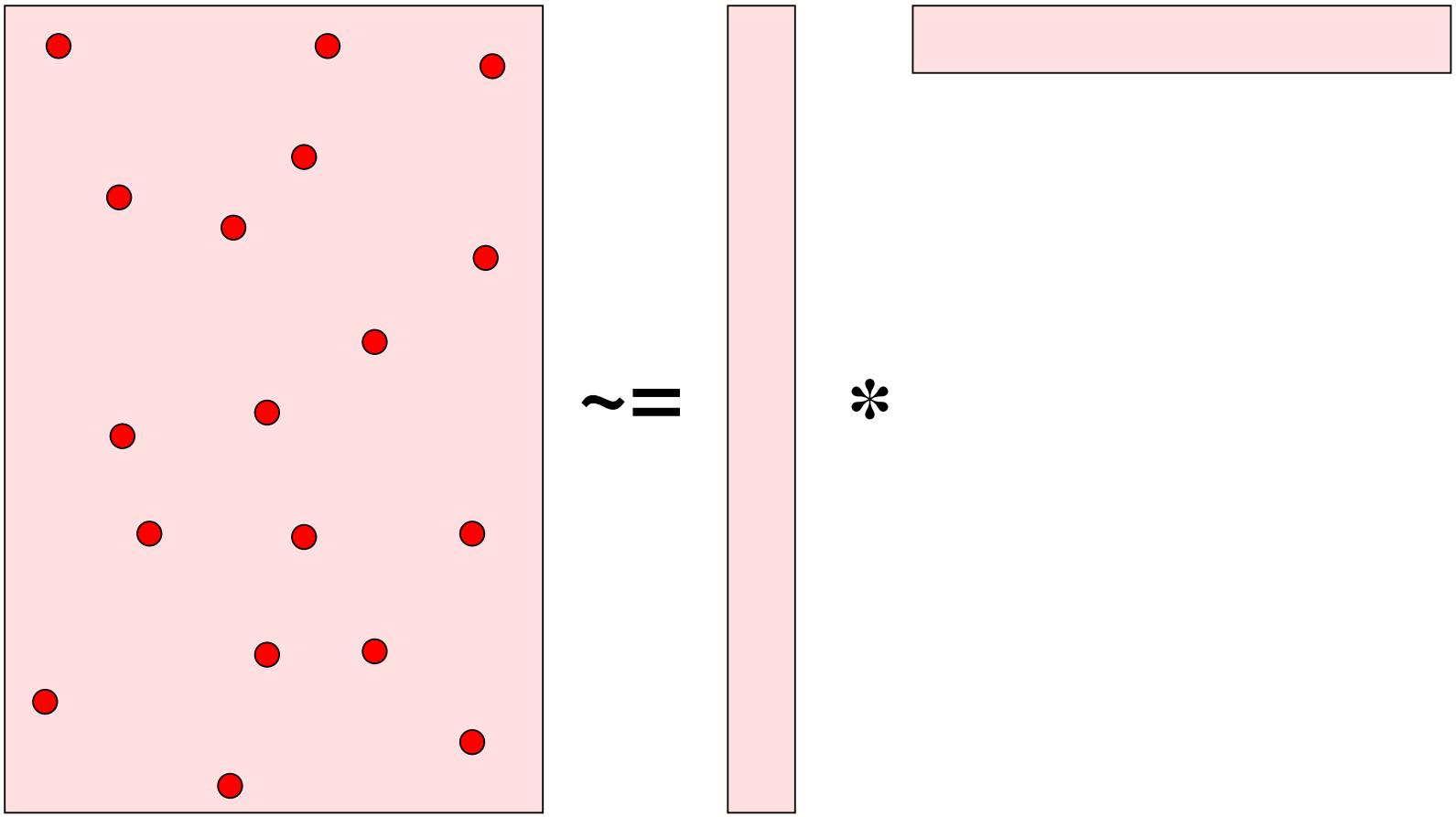
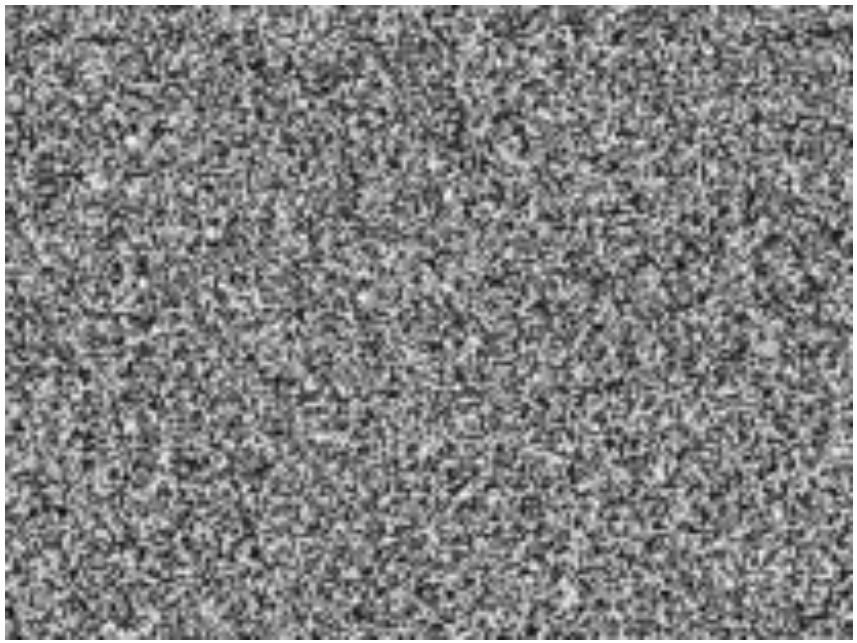


Image Processing

Natural images are not random! They tend to have large areas of near-constant intensity or color, separated by sharp edges.



Denoising / Deblurring: Given an image with noise or blur, seek a "nearby natural image."

Total Variation Regularization

Apply an ℓ_1 penalty to spatial gradients in the 2D image, defined by

$$u : \Omega \to \mathbb{R}, \qquad \Omega := [0,1] \times [0,1],$$

Given a noisy image $f : \Omega \to \mathbb{R}$, solve for u: (Rudin et al., 1992)

$$\min_u \int_\Omega (u(x) - f(x))^2 \, dx + \lambda \int_\Omega \|\nabla u(x)\|_2 \, dx.$$

Data Assimilation

There are thriving communities in computational science that study PDE-constrained optimization and in particular data assimilation. The latter is the basis of weather forecasting.

These are based on parametrized partial differential equation models, whose parameters are determined from

- data (huge, heterogeneous): observations of the system state at different points in space and time;
- statistical models of noise, in both the PDE model and observations;
- prior knowledge about the solution, such as a guess of the optimal value and an estimate of its reliability.

Needs models (meteorology and oceanography), statistics, optimization, scientific computing, physics, applied math,...

There is active research on better noise models (better covariances).

III. Optimization Formulations: Typical Properties

- Data, from which we want to extract key information, make inferences about future / missing data, or guide decisions.
- Parametrized model that captures the relationship between the data and the meaning we are trying to extract.
- Objective that measures the mismatch between current model / parameters and observed data; also deviation from prior knowledge or desired structure.

In some cases, the optimization formulation is well settled: See above.

In other areas, formulation is a matter of ongoing debate!

Optimization Toolbox

A selection of fundamental optimization techniques that feature strongly in the applications above.

Most have a long history, but the slew of interesting new applications and contexts has led to new twists and better understanding.

- Accelerated Gradient (and its cousins)
- Stochastic Gradient
- Coordinate Descent
- Asynchronous Parallel
- Shrinking techniques for regularized formulations
- Higher-order methods
- Augmented Lagrangians, Splitting, ADMM.

Describe each briefly, then show how they are deployed to solve the applications in Part I.

Gradient Methods: Steepest Descent

$\min f(x)$, with smooth convex f. First-order methods calculate $\nabla f(x_k)$ at each iteration, and do something with it.

Compare these methods on the smooth convex case:

$$\mu I \preceq \nabla^2 f(x) \preceq LI \text{ for all } x \qquad (0 \leq \mu \leq L). \quad \text{Conditioning: } \kappa = L/\mu.$$

Steepest Descent sets

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k), \qquad \text{for some } \alpha_k > 0.$$

When $\mu > 0$, set $\alpha_k \equiv 2/(\mu + L)$ to get linear convergence, at rate depending on conditioning κ:

$$f(x_k) - f(x^*) \leq \frac{L}{2}\left(1 - \frac{2}{\kappa + 1}\right)^{2k} \|x_0 - x^*\|^2.$$

Need $O(\kappa \log \epsilon)$ iterations to reduce the error by a factor ϵ.

We can't improve much on these rates by using more sophisticated choices of α_k — they're a fundamental limitation of searching along $-\nabla f(x_k)$.

Momentum!

First-order methods can be improved dramatically using momentum:

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k) + \beta_k (x_k - x_{k-1}).$$

Search direction is a combination of previous search direction $x_k - x_{k-1}$ and latest gradient $\nabla f(x_k)$. Methods in this class include: Heavy-Ball, Conjugate Gradient, Accelerated Gradient, Dual Averaging.

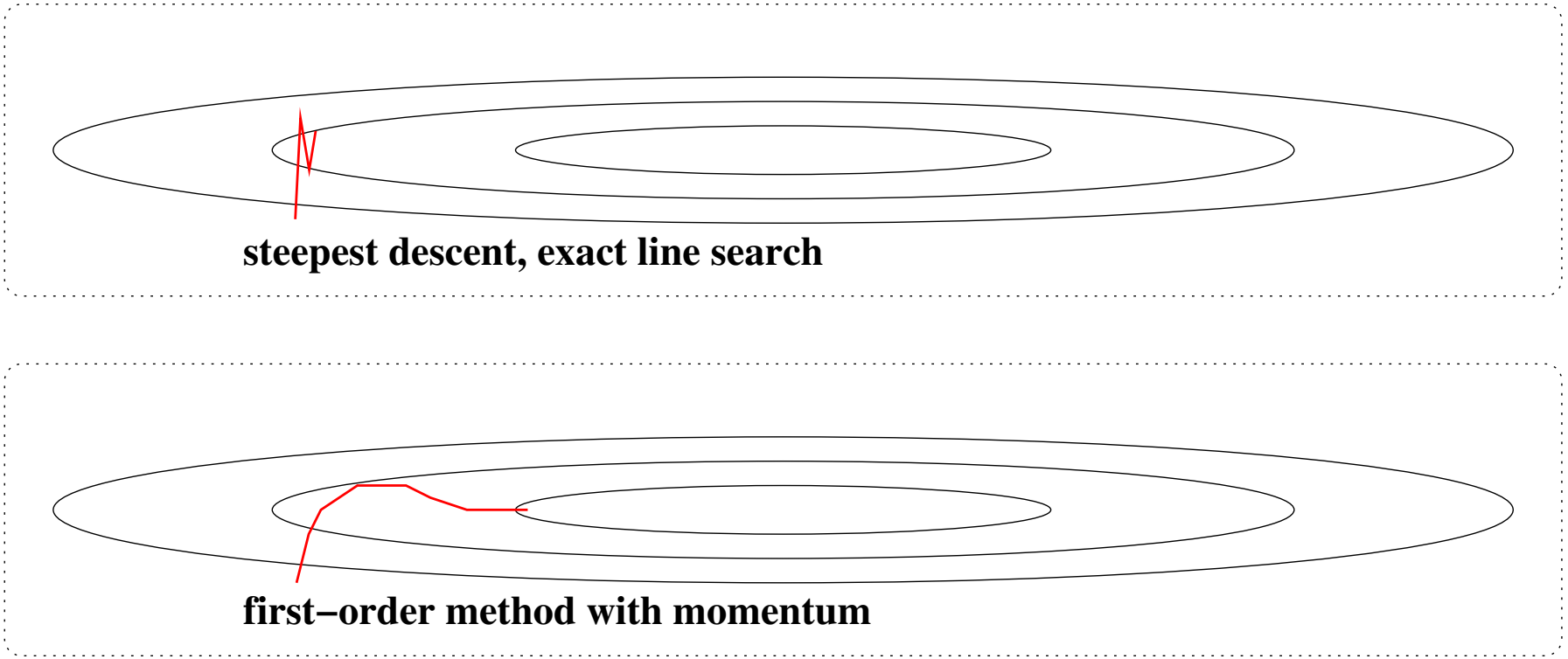
Heavy-ball sets

$$\alpha_k \equiv \frac{4}{L} \frac{1}{(1 + 1/\sqrt{\kappa})^2}, \quad \beta_k \equiv \left(1 - \frac{2}{\sqrt{\kappa} + 1}\right)^2.$$

to get a linear convergence rate with constant approximately $1 - 2/\sqrt{\kappa}$.

Thus requires about $O(\sqrt{\kappa} \log \epsilon)$ to achieve precision of ϵ, vs. about $O(\kappa \log \epsilon)$ for steepest descent. (Polyak, 1987)

First-Order Methods and Momentum



Accelerated Gradient Methods

Accelerate the rate to $1/k^2$ for weakly convex, while retaining the linear rate (based on $\sqrt{\kappa}$) for strongly convex case.

One of Nesterov's methods (Nesterov, 1983, 2004) is:

0: Choose x_0, $\alpha_0 \in (0,1)$; set $y_0 \leftarrow x_0$./

k: $x_{k+1} \leftarrow y_k - \frac{1}{L}\nabla f(y_k)$; (*short-step gradient*)
solve for $\alpha_{k+1} \in (0,1)$: $\alpha_{k+1}^2 = (1-\alpha_{k+1})\alpha_k^2 + \alpha_{k+1}/\kappa$;
set $\beta_k = \alpha_k(1-\alpha_k)/(\alpha_k^2 + \alpha_{k+1})$;
set $y_{k+1} \leftarrow x_{k+1} + \beta_k(x_{k+1} - x_k)$. (*update with momentum*)

- Separates "steepest descent" contribution from "momentum" contribution, producing two sequences $\{x_k\}$ and $\{y_k\}$.
- Still works for weakly convex ($\kappa = \infty$).
- FISTA (Beck and Teboulle, 2009) is similar.

Extends easily to problems with convex constraints, regularization, etc.

Stochastic Gradient (SG)

Still deal with (weakly or strongly) convex f. But change the rules:

- Allow f nonsmooth.
- Don't calculate function values $f(x)$.
- Can evaluate cheaply an unbiased estimate of a vector from the subgradient ∂f.

Consider the finite sum:

$$f(x) = \frac{1}{m} \sum_{i=1}^{m} f_i(x),$$

where each f_i is convex and m is huge. Often, each f_i is a loss function associated with ith data item (SVM, regression, ...), or a mini-batch.

Classical SG: Choose index $i_k \in \{1, 2, \ldots, m\}$ uniformly at random at iteration k, set

$$x_{k+1} = x_k - \alpha_k \nabla f_{i_k}(x_k),$$

for some steplength $\alpha_k > 0$. (Robbins and Monro, 1951)

Classical SG

Suppose f is strongly convex with modulus μ, there is a bound M on the size of the gradient estimates:

$$\frac{1}{m}\sum_{i=1}^{m}\|\nabla f_i(x)\|^2 \leq M^2$$

for all x of interest. Convergence obtained for the expected square error:

$$a_k := \frac{1}{2}E(\|x_k - x^*\|^2).$$

Elementary argument shows a recurrence:

$$a_{k+1} \leq (1 - 2\mu\alpha_k)a_k + \frac{1}{2}\alpha_k^2 M^2.$$

When we set $\alpha_k = 1/(k\mu)$, a neat inductive argument reveals a $1/k$ rate:

$$a_k \leq \frac{Q}{2k}, \qquad \text{for } Q := \max\left(\|x_1 - x^*\|^2, \frac{M^2}{\mu^2}\right).$$

Many variants: constant stepsize, primal averaging, dual averaging.

Coordinate Descent (CD)

Again consider unconstrained minimization for smooth $f : \mathbb{R}^n \to \mathbb{R}$:

$$\min_{x \in \mathbb{R}^n} f(x).$$

Iteration k of coordinate descent (CD) picks one index i_k and takes a step in the i_k component of x to decrease f, typically

$$x_{k+1} = x_k - \alpha_k [\nabla f(x_k)]_{i_k} e_{i_k},$$

where e_{i_k} is the unit vector with 1 in the i_k location and 0 elsewhere.

- Deterministic CD: choose i_k in some fixed order e.g. cyclic;
- Stochastic CD: choose i_k at random from $\{1, 2, \ldots, n\}$.

CD is a reasonable choice when it's cheap to evaluate individual elements of $\nabla f(x)$ (at $1/n$ of the cost of a full gradient, say).

Coordinate Descent: Extensions and Convergence

Block variants of CD choose a subset $I_k \subset \{1, 2, \ldots, n\}$ of components at iteration k, and take a step in that block of components.

Can also apply coordinate descent when there are bounds on components of x. Or, more generally, constraints that are separable with respect to the blocks in a block CD method.

Similar extensions to separable regularization functions (see below).

Convergence: Deterministic (Luo and Tseng, 1992; Tseng, 2001), linear rate (Beck and Tetruashvili, 2013). Stochastic, linear rate: (Nesterov, 2012).

Much recent work on parallel variants (synchronous and asynchronous).

Relating CD and SG

There is kind of duality relationship between CD and SG.

See this most evidently in the feasible linear system $Aw = b$, where A is $m \times n$ and possibly rank deficient: the Kaczmarz algorithm.

Write as a least-squares problem

$$\min_w \frac{1}{2m} \sum_{i=1}^{m} (A_i w - b_i)^2,$$

where A_i is the ith row of A (assume normalized: $\|A_i\|_2 = 1$ for all $i = 1, 2, \ldots, m$).

SG updates with $\alpha_k \equiv 1$ are

$$w^{k+1} = w^k - A_{i_k}^T (A_{i_k} w^k - b_{i_k}) \quad \text{Kaczmarz step!}$$

Project onto the hyperplane define by the i_k equation.

Suppose we seek a minimum-norm solution from the formulation

$$\textbf{P:} \quad \min_{w \in \mathbb{R}^n} \frac{1}{2}\|w\|^2 \quad \text{subject to } Aw = b,$$

for which the dual is

$$\textbf{D:} \quad \min_{x \in \mathbb{R}^n} \frac{1}{2}\|A^T x\|_2^2 - b^T x,$$

where primal and dual solutions are related by $w = A^T x$.

CD updates applied to the dual with $\alpha_k \equiv 1$ are

$$x^{k+1} = x^k - (A_{i_k} A^T x^k - b_{i_k}) e_{i_k}.$$

Multiplying by A^T and using the identity $w = A^T x$, we recover the Kaczmarz step:

$$w^{k+1} = w^k - A_{i_k}^T (A_{i_k} w^k - b_{i_k}).$$

References I

Beck, A. and Teboulle, M. (2009). A fast iterative shrinkage-threshold algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202.

Beck, A. and Tetruashvili, L. (2013). On the convergence of block coordinate descent type methods. *SIAM Journal on Optimization*, 23(4):2037–2060.

Boser, B. E., Guyon, I. M., and Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, pages 144–152.

Luo, Z. Q. and Tseng, P. (1992). On the convergence of the coordinate descent method for convex differentiable minimization. *Journal of Optimization Theory and Applications*, 72(1):7–35.

Nesterov, Y. (1983). A method for unconstrained convex problem with the rate of convergence $O(1/k^2)$. *Doklady AN SSSR*, 269:543–547.

Nesterov, Y. (2004). *Introductory Lectures on Convex Optimization: A Basic Course*. Springer Science and Business Media, New York.

Nesterov, Y. (2012). Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22:341–362.

Polyak, B. T. (1987). *Introduction to Optimization*. Optimization Software.

Recht, B., Fazel, M., and Parrilo, P. (2010). Guaranteed minimum-rank solutions to linear matrix equations via nuclear norm minimization. *SIAM Review*, 52(3):471–501.

References II

Robbins, H. and Monro, S. (1951). A stochastic approximation method. *Annals of Mathematical Statistics*, 22(3).

Rudin, L., Osher, S., and Fatemi, E. (1992). Nonlinear total variation based noise removal algorithms. *Physica D*, 60:259–268.

Sainath, T. N., Kingsbury, B., Soltau, H., and Ramabhadran, B. (2013). Optimization techniques to improve training speed of deep neural networks for large speech tasks. *IEEE Transactions on Audio, Speech, and Language Processing*. To appear.

Strohmer, T. and Vershynin, R. (2009). A randomized Kaczmarz algorithm with exponential convergence. *Journal of Fourier Analysis and Applications*, 15:262–278.

Tibshirani, R. (1996). Regression shrinkage and selection via the LASSO. *Journal of the Royal Statistical Society B*, 58:267–288.

Tseng, P. (2001). Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of Optimization Theory and Applications*, 109(3):475–494.

Vapnik, V. N. (1999). *The Nature of Statistical Learning Theory*. Statistics for Engineering and Information Science. Springer, second edition.