Convex Optimization with a Sequential Paradigm for Iterative Numerical Algorithms

Noah Singer

November 2017

Outline

Optimization

- **Optimization problem:** find minimum value of **objective** over **domain**
- All problems can be formulated as optimization problems
- In general very difficult (e.g. ILP or SAT)
- Specific classes of optimization problems can be easier

Convex functions

- "Curve up" everywhere
- All **local** optima are also **global** optima
- Easier to optimize using iterative **gradient descent** algorithm to follow contours downward
  1. Observe small local area
  2. Find direction of steepest descent (anti-gradient)
  3. Take small step in steepest descent direction
  4. Repeat

Convex function examples

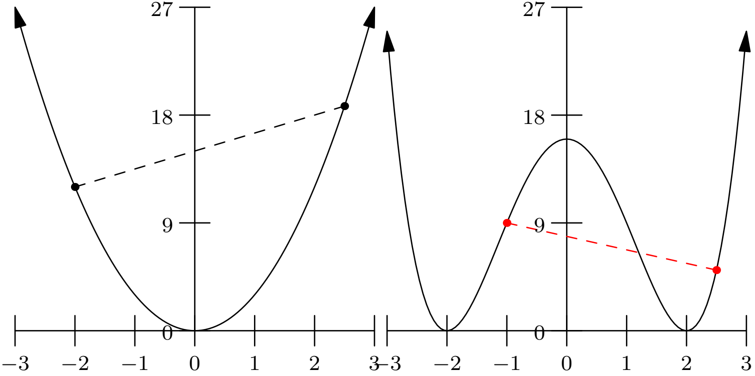


Figure: A convex and nonconvex function.

This research

- Developed a Python implementation of FASTA, Goldstein et al.'s algorithm for constrained convex optimization
- Based on gradient descent, with many of the latest techniques and improvements from the literature
- Implementation is high-quality usable Python code (as opposed to MATLAB)
- Visualizations to monitor progress of optimization algorithms
- Well-documented and organized
- Test on eleven example problems
- Develop FLOW to generalize structure of iterative numerical algorithms
- Numerical results on difficulty of problems

Gradient descent

- Follows "contours" (gradient) downward to local minimum
- Also global if objective is convex
- Basic update:

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \tau \nabla f(\mathbf{x}^{(k)})$$

  where τ is **stepsize**
- Convergence for τ bound by $2/L$ where L is Lipschitz constant of gradient
- Convergence guaranteed in $\mathcal{O}(1/k)$ where k is number of iterations

Backtracking

- L is generally difficult to know analytically or computationally
- Solution: approximate as

$$L_{est} = \frac{||\nabla f(\mathbf{x_1}) - \nabla f(\mathbf{x_2})||}{||\mathbf{x_1} - \mathbf{x_2}||}$$

  for $\mathbf{x_1}, \mathbf{x_2}$ random vectors
- Check **backtracking condition** and decrease τ by factors of β until satisfied at each iteration

Forward/backward splitting

- **Proximal operator** of g centered at $\mathbf{v}$ with stepsize τ defined as

$$\operatorname{prox}_g(\mathbf{v}, \tau) = \arg\min_{\mathbf{x}} \left(f(\mathbf{x}) + \frac{\tau}{2} ||\mathbf{x} - \mathbf{v}||^2 \right)$$

- Can be computed easily for certain classes of functions: Euclidean projection for characteristic functions, shrink operator for ℓ_1-norm, and many others
- When objective h is non-differentiable, split it into a sum of a differentiable function f and non-differentiable function g where proximal operator of g can be easily computed
- Alternate gradient (forward) and proximal (backward) steps

Accelerated descent

- Goal: converge in $\mathcal{O}(1/k^2)$ even for poorly-conditioned problems (theoretical bound)
- **Momentum**: take larger steps when gradients are correlated
- Acceleration is controlled by **acceleration parameter** α

Results of descent

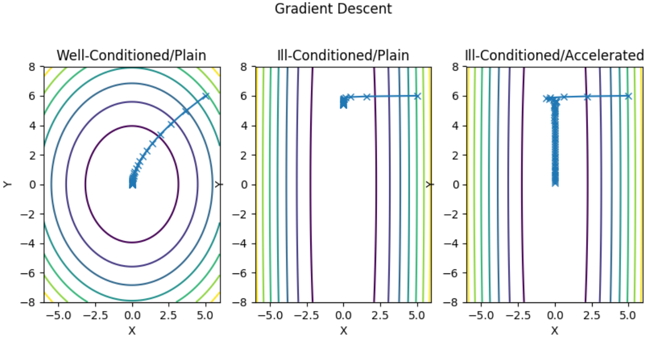


Figure: Contour plots, visualized with FASTA, of the convergence of various FASTA modes.

Adaptive stepsize selection

- Heuristic that almost always outperforms plain and accelerated modes
- Approximates objective as perfectly conditioned quadratic

$$q(\mathbf{x}) = \frac{1}{2}\mathbf{x}^\mathsf{T}\nabla^2 f(\mathbf{x})\mathbf{x} - \mathbf{b}$$

  and computes a stepsize τ by solving least squares problems on the approximation
- Fails when quadratic is poorly conditioned

Optimization problems

- Modified least squares problems

$$\arg\min_{\mathbf{x}} ||A\mathbf{x} - b||^2 + \gamma(x)$$

1. LASSO: restricts ℓ_1-norm of $\mathbf{x}$
2. Basis pursuit denoising: penalizes ℓ_1-norm of $\mathbf{x}$
3. Democratic representation: penalizes ℓ_∞-norm of $\mathbf{x}$
4. Non-negative least squares: restricts $\mathbf{x}$ to be non-negative
5. Sparse logistic least squares: penalizes ℓ_1-norm of $\mathbf{x}$ and uses logistic least-odds (logit) instead of ℓ_2-norm
6. Multiple measurement vector: on matrices, penalizes using a group sparsity prior and uses Frobenius norm instead of ℓ_2-norm

Optimization problems, continued

- Duality problems are converted to functions f^* of Lagrange multipliers $\boldsymbol{\lambda}$
  1. Total-variation denoising: minimize differences in intensities between adjacent pixels in an image to reduce noise
  2. Support vector machine: find the hyperplane which maximally separates two classes of data
- Non-convex problems
  1. Non-negative matrix factorization: factors a given matrix into the product of two non-negative matrices with certain constraints
  2. Max-norm optimization: used to compute max-cuts in graphs with important applications in clustering

Example: Total-variation denoising

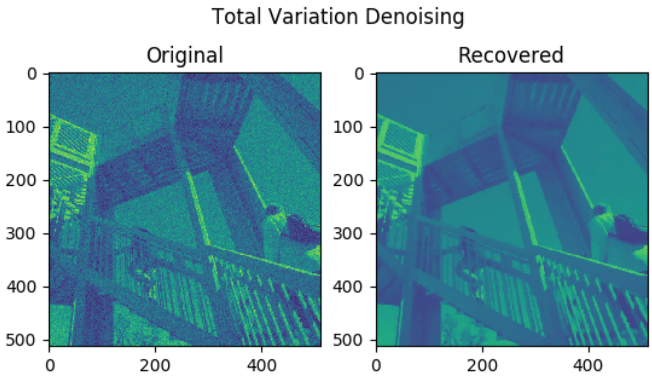


Figure: An image denoised using FASTA to optimize the total-variation denoising problem.

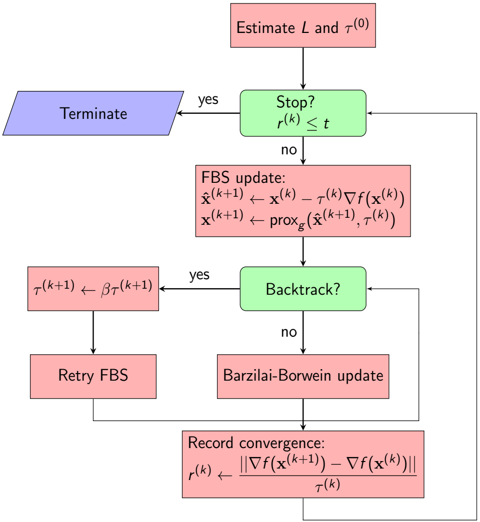
Issues with standard paradigms

- Variables must be manually tracked for the current and previous rounds, as well as possibly over all rounds
- No clear separation of configuration, structure, and mathematics
- Adjustment and testing of parameters and options cannot not easily be automated

FLOW definitions

- **Flow**: a discrete block of imperative code that is the fundamental unit of FLOW
- **State**: a collection of variables that's modified in place by a flow
- **Chain**: a linear combination of flows
- **Switch**: a flow that activates one of two flows depending on the value of the special *condition variable*
- **Loop**: a flow that repeatedly activates a *body flow* while a *condition flow* sets the condition variable
- **Tape**: another component of a state that tracks the value of a variable between iterations of a loop

FASTA in FLOW



Results

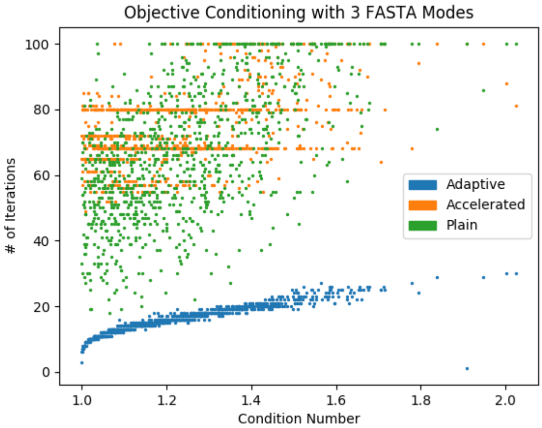


Figure: .

Summary

- Ubiquitous importance of optimization
- Implemented FASTA algorithm for convex optimization; easy-to-use and well-documented
- Visualizations and example problems
- Developed FLOW paradigm to facilitate development of FASTA and other iterative algorithms
- Numerical analysis of condition number

Applications

- Optimization problems appear in the field all branches of science, mathematics, and engineering:
  - Minimizing error in a statistical model
  - Maximizing the range of a rocket as a function of launch angle
  - Maximizing the efficiency of a resource allocation in industrial engineering
- Ease of use and analysis of convex optimization algorithms for other research
- Use in non-convex optimization problems (e.g. phase retrieval)

Future

- Further investigate behavior of modes of FASTA as a function of objective conditioning
- Implement other iterative algorithms in FLOW
- Develop FLOW to allow for automated analysis and testing
- Extend to non-convex phase retrieval problem

Acknowledgements

- Dr. Thomas Goldstein
- Rohan Chandra
- Val McCulloch, Ziyuan Zhong, Justin Hontz
- Dr. William Gasarch
- Ms. Bosse