

TFOCS: A General Framework for Solving Constrained Optimization

Stephen Becker

California Institute of Technology

BIRS, Jan 2011

Collaborators:

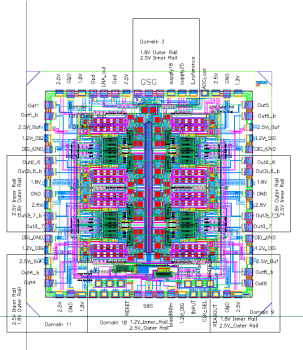
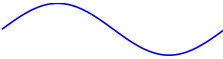
Michael Grant (Caltech, CVX Research)

Emmanuel Candès (Stanford)

Motivation: new analog-to-digital converter

Analog-to-*information* (A2I) converter

signal x
2.5 GHz



Sample at 400 MHz
Output:

$$b_k, \quad k \in \mathbb{Z}$$

Measurements are linear:

$$b = Ax$$

Convex optimization to recover x

Typical problems

$$b = Ax + z, \quad A \in \mathbb{R}^{m \times n}$$

x is sparse, $m \ll n$

Basis Pursuit BP

$$\min_x \|x\|_1 \quad \text{subject to} \quad Ax = b$$

or if x is “ W -sparse”: i.e. $\exists \alpha \in \mathbb{R}^d$ sparse, such that $W^T \alpha \simeq x$.

Basis Pursuit Denoising BP_ϵ , analysis

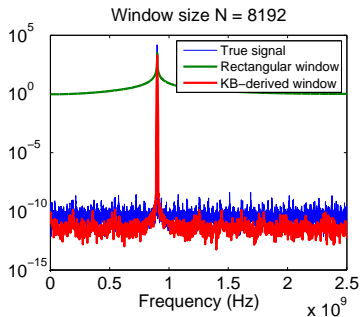
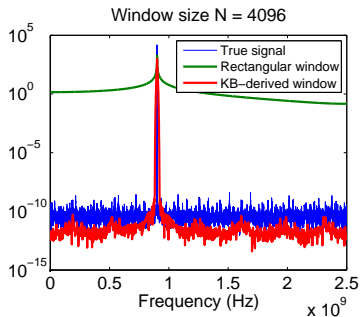
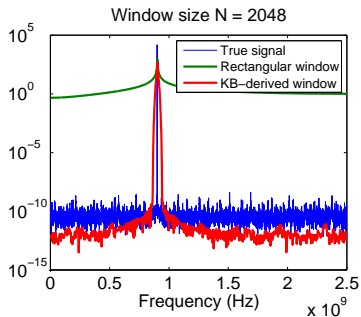
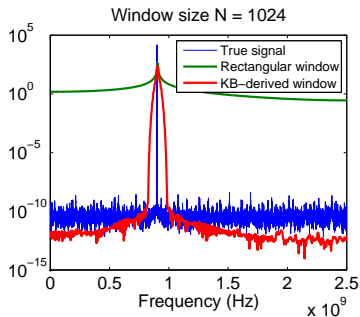
$$\min_x \|Wx\|_1 \quad \text{subject to} \quad \|Ax - b\|_2 \leq \epsilon$$

Alternatives:

Dantzig Selector

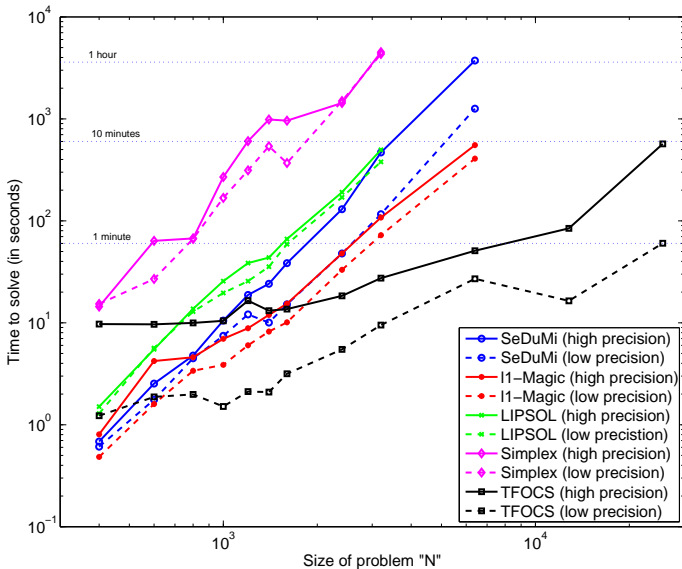
$$\min_x \|x\|_1 \quad \text{subject to} \quad \|A^T(Ax - b)\|_\infty \leq \delta$$

Why *largescale*?



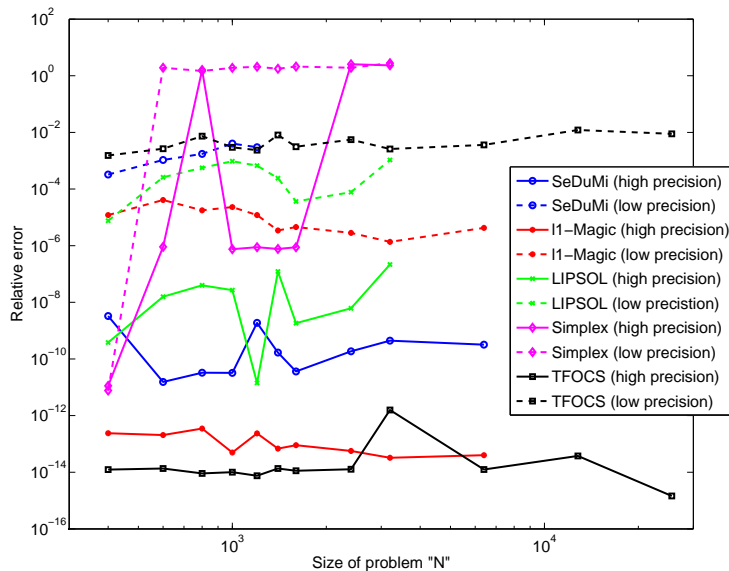
Interior Point methods

Experiments that ran on a cluster (2008) are now run on a laptop.



Interior Point methods (2)

But *accuracy* of first-order methods...? Not a problem.



First-order methods



DEFEAT

FOR EVERY WINNER, THERE ARE DOZENS OF LOSERS.
ODDS ARE YOU'RE ONE OF THEM.

www.despair.com

Conclusion: due to **size of problem**, first-order methods beat IPM for this application.

Difficult to exploit fast operators (FFT, ...)

Similar fact: homotopy-type methods don't do well either

Existing first-order solvers

$$\begin{aligned} \min_x \|Wx\|_1 \\ \text{subject to } \|Ax - b\|_2 \leq \varepsilon \end{aligned} \quad \min_x \|Wx\|_1 + \frac{\lambda}{2} \|Ax - b\|_2^2$$

- 1 Require $AA^T = I$ or solve inner subproblem
 - NESTA (B., Bobin, Candès)
 - C-SALSA (Afonso, Figueiredo, Bioucas-Dias)
 - Z. Lu (for the Dantzig)
 - (Lu, Pong, Zhang) ADM for Dantzig
- 2 Restrictions on W
 - SPGL1 (Friedlander, van den Berg)
- 3 Solve un-constrained version, or equality constraints only
 - FPC, FPC-AS, GPSR, SpaRSA, FISTA, Bregman, ...

Can we do better?

Challenges

- 1 How to project onto $\|Ax - b\|_2 \leq \varepsilon$?
- 2 Non-smooth, so *slower* convergence

TFOCS ideas: fundamentals

$$\min_x f(x) + \psi(\bar{\mathcal{A}}x + \bar{b})$$

- 1 Find conic formulation*
- 2 Add strongly convex term
 - $f_\mu(x) = f(x) + \frac{\mu}{2} \|x - x_0\|^2$
 - can now calculate dual
 - dual is smooth
- 3 Solve dual problem
 - composite approach
 - $g = g_{\text{smooth}} + h$
 - h nonsmooth but “nice”

Extends (e.g., atomic norms)

TFOCS ideas: fundamentals

$$\min_x f(x) + \psi(\bar{A}x + \bar{b})$$

- 1 Find conic formulation*
- 2 Add strongly convex term
 - $f_\mu(x) = f(x) + \frac{\mu}{2} \|x - x_0\|^2$
 - can now calculate dual
 - dual is smooth
- 3 Solve dual problem
 - composite approach
 - $g = g_{\text{smooth}} + h$
 - h nonsmooth but “nice”

Potential drawbacks:

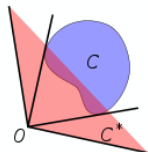
- Primal iterate is not feasible
 - $\|Ax - b\| \leq \varepsilon$, but ε is estimate!
- Effect of smoothing
 - use continuation
 - made rigorous in proximal point framework
 - accelerated continuation
 - sometimes no effect even for $\mu > 0$

Extends (e.g., atomic norms)

Benefits of duality 1: projection

Dual cone \mathcal{K}^*

$$\lambda \in \mathcal{K}^* \text{ if } \langle \lambda, x \rangle \geq 0 \quad \forall x \in \mathcal{K}$$



Lagrangian cross-terms: $-\langle \mathcal{A}(x), \lambda \rangle$ with $\mathcal{A}(x) \in \mathcal{K}$ and $\lambda \in \mathcal{K}^*$

Our constraint set, $C = \{x : \|Ax - b\|_2 \leq \varepsilon\}$ is a cone:

$$(Ax - b, \varepsilon) \in \mathcal{K}_2$$

Epigraph cone \mathcal{K}_p

$$\mathcal{K}_p \subset \mathbb{R}^{n+1}, \quad \mathcal{K}_p = \{(x, t) : \|x\|_p \leq t\}$$

ex. \mathcal{K}_1 and \mathcal{K}_∞ are dual, \mathcal{K}_2 is self-dual.

Benefits of duality 1: projection

Many projections known in closed form. Projecting onto \mathcal{K} vs onto \mathcal{K}^* makes little difference.

- 1 Project onto ℓ_2 ball: just re-scale.
 - Cost: $\mathcal{O}(N)$
 - Project onto \mathcal{K}_2 similar: $\mathcal{O}(N)$
- 2 Project onto ℓ_1 ball or \mathcal{K}_1 : sort, then soft-threshold.
 - Cost: $\mathcal{O}(N \log(N))$
- 3 Project onto ℓ_∞ ball: truncation

Key observation:

Projecting x onto $\{y : y \in \mathcal{K}\}$ is (usually) easy

Projecting x onto $\{y : Ay \in \mathcal{K}\}$ is (usually) hard

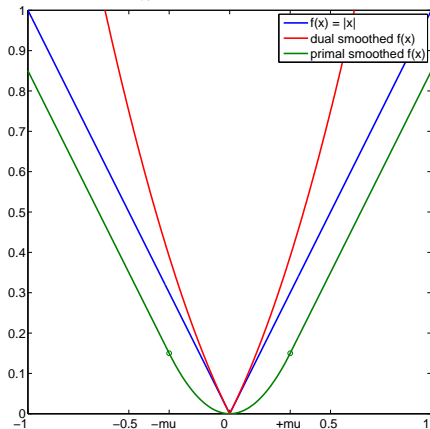
Benefits of duality 2: better smoothing

Dual smoothing:
primal has **kink**

$$f^*(\lambda) \equiv \sup_x \langle \lambda, x \rangle - f(x)$$

Smooth problems: *much*
faster convergence, i.e. $\mathcal{O}(\frac{1}{k^2})$

f strongly convex $\implies f^*$
differentiable and ∇f^*
Lipschitz



Example: Dantzig

$$\begin{array}{ll} \text{minimize} & \|x\|_1 \\ \text{subject to} & \|A^T(Ax - b)\|_\infty \leq \delta \end{array} \implies \begin{array}{ll} \text{minimize} & \|x\|_1 + \frac{\mu}{2}\|x - x_0\|^2 \\ \text{subject to} & (\bar{\mathcal{A}}(x) + \bar{b}, \delta) \in \mathcal{K}_\infty \end{array}$$
$$\bar{\mathcal{A}} \leftarrow A^T A, \quad \bar{b} \leftarrow -A^T b$$

Dual problem

$$\text{maximize}_\lambda \quad \underbrace{\inf_x \left\{ \|x\|_1 + \frac{\mu}{2}\|x - x_0\|^2 - \langle \lambda, \bar{\mathcal{A}}(x) + \bar{b} \rangle \right\}}_{-g_{\text{sm}}(\lambda)} - \underbrace{\delta \|\lambda\|_1}_{h(\lambda)}$$

- “Simple gradient” (x_λ unique minimizer above)

$$\nabla g_{\text{sm}}(z) = \bar{\mathcal{A}}(x_\lambda + \bar{b})$$

Example: Dantzig, version 2

$$\begin{array}{ll} \text{minimize} & \|x\|_1 \\ \text{subject to} & \|A^T(Ax - b)\|_\infty \leq \delta \end{array} \implies \begin{array}{ll} \text{minimize} & t + \frac{\mu}{2}\|x - x_0\|^2 \\ \text{subject to} & (\bar{\mathcal{A}}(x) + \bar{b}, \delta) \in \mathcal{K}_\infty \\ & (x, t) \in \mathcal{K}_1 \end{array}$$

$$\bar{\mathcal{A}} \leftarrow A^T A, \quad \bar{b} \leftarrow -A^T b$$

Dual problem

$$\begin{aligned} \max_{\lambda, (\nu, s) \in \mathcal{K}_\infty} & \underbrace{-\delta\|\lambda\|_1}_{h(\lambda)} + \dots \\ & \underbrace{\inf_{x, t} \left\{ t + \frac{\mu}{2}\|x - x_0\|^2 - \langle \lambda, \bar{\mathcal{A}}(x) + \bar{b} \rangle - \langle \nu, x \rangle - st \right\}}_{-g_{\text{sm}}(\lambda)} \end{aligned}$$

Similar algorithm, but now $x_{\lambda, \nu}$ is linear in λ and ν , so dual is constrained quadratic (and with $2 \times$ variables).

General form

Two viewpoints: **conic dual** or **Fenchel dual**

Fenchel duality view

$$\min f(x) + \sum_i \psi_i(A_i x + b_i)$$

where f, ψ_i^* are “prox-capable”, $\psi_i \rightarrow \bar{\mathbb{R}}$

- 1 Dantzig style 1 corresponds to:

$$f(x) = \|x\|_1, \quad \psi_1(A_i x + b_i) = \iota_C$$

- 2 Dantzig style 2 corresponds to:

$$f = 0, \quad \psi_1(x) = \|x\|_1, \quad \psi_2(A_i x + b_i) = \iota_C$$

where $C = \{x : \|A^T(Ax - b)\|_\infty \leq \delta\}$.

If $f = 0$, dual is always (constrained) quadratic.

Solving the dual

“Proximal gradient descent”, aka “forward-backward” algorithm.
Handles smooth + nonsmooth (Fukushima and Mine, 1981).

- Gradient projection step for minimizing smooth g :

$$\lambda_{k+1} \leftarrow \operatorname{argmin}_{\lambda \in \mathcal{K}^*} g(\lambda_k) + \langle \nabla g(\lambda_k), \lambda - \lambda_k \rangle + \frac{L}{2} \|\lambda - \lambda_k\|^2$$

- Generalized gradient projection for minimizing $g + h$ (h nonsmooth)

$$\lambda_{k+1} \leftarrow \operatorname{argmin}_{\lambda} g(\lambda_k) + \langle \nabla g(\lambda_k), \lambda - \lambda_k \rangle + \frac{L}{2} \|\lambda - \lambda_k\|^2 + h(\lambda)$$

- Solution is **proximity operator** of h . Often known.
 - Ex. $h = \chi_C$, then proximity operator is just projection onto C
 - Ex. $h = \|x\|_1$, then proximity operator is shrinkage
- Works with backtracking and Nesterov acceleration

Generic algorithm (Nesterov's style)

Auslander-Teboulle version, no backtracking

$$\min_x f(x) + \psi(\bar{A}x + \bar{b}), \quad h \stackrel{\text{def}}{=} \psi^*$$

Algorithm 1 Generic algorithm for the conic standard form

Require: $\lambda_0, x_0 \in \mathbb{R}^n$, $\mu > 0$, step sizes $\{t_k\}$

1: $\theta_0 \leftarrow 1, v_0 \leftarrow \lambda_0$

2: **for** $k = 0, 1, 2, \dots$ **do**

3: $\nu_k \leftarrow (1 - \theta_k)v_k + \theta_k \lambda_k$

4: $x_k \leftarrow \operatorname{argmin}_x f(x) + \mu/2 \|x - x_0\|^2 - \langle \bar{A}^T(\nu_k), x \rangle$

5: $\lambda_{k+1} \leftarrow \operatorname{argmin}_\lambda h(\lambda) + \frac{\theta_k}{2t_k} \|\lambda - \lambda_k\|^2 + \langle \bar{A}(x_k) + \bar{b}, \lambda \rangle$

6: $v_{k+1} \leftarrow (1 - \theta_k)v_k + \theta_k \lambda_{k+1}$

7: $\theta_{k+1} \leftarrow 2/(1 + (1 + 4/\theta_k^2)^{1/2})$

8: **end for**

x is primal

λ, ν, v are dual, θ is scalar

Algorithm for Dantzig selector

Algorithm 2 Algorithm excerpt for DS

4: $x_k \leftarrow \text{SoftThreshold}(x_0 - \mu^{-1}A^T A\nu_k, \mu^{-1})$.

5: $\lambda_{k+1} \leftarrow \text{SoftThreshold}(\lambda_k - \theta_k^{-1}t_k A^T(\nu - Ax_k), \theta_k^{-1}t_k\delta)$

$$\text{SoftThreshold}(x, \tau) = \text{sign}(x) \cdot \max\{|x| - \tau, 0\}$$

Not previously considered

TFOCS ideas: extras

Software is **modular**. Easy to add constraints, change solver...

Overview

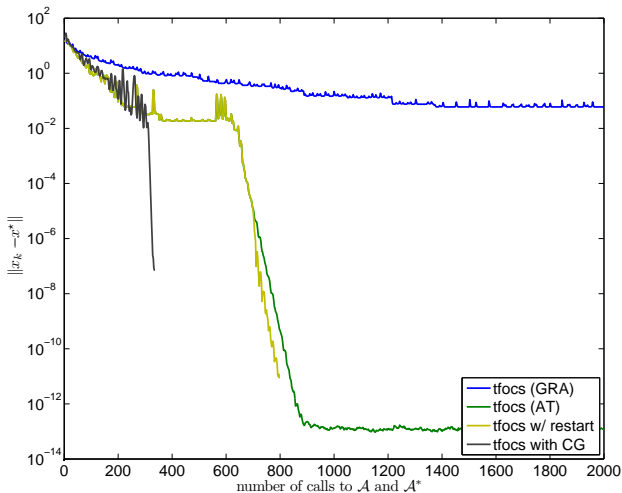
- 6 first-order methods (GRA + 5 accelerated methods)
- Efficient step size procedures (based on Tseng's convergence analysis): no Lipschitz constant needed
- Efficient use of linear operator structure: crucial when backtracking occurs

$$\text{minimize } g_{\text{sm}}(\mathcal{A}^T \lambda) + h(\lambda)$$

- Accelerated continuation
- Exact perturbation
- Restart strategies
- New convergence proofs

CG

Advantage of modularity: easy to try new solvers, line search.
Plans for non-linear CG, (L-)BFGS, SESOP, ...



Ex: Non-linear CG (Polak-Ribiere), noiseless basis pursuit, $N = 2048$.

Standard continuation

Want perturbation small

$$\begin{array}{ll} \text{minimize} & f(x) + \frac{1}{2}\mu\|x - x_0\|^2 \\ \text{subject to} & \mathcal{A}(x) + b \in \mathcal{K} \end{array}$$

Problem: $L \propto 1/\mu$

Algorithm 3 Standard continuation

Require: $Y_0, \mu_0 > 0, \beta < 1$

- 1: **for** $j = 0, 1, 2, \dots$ **do**
 - 2: $X_{j+1} \leftarrow \underset{\mathcal{A}(x)+b \in \mathcal{K}}{\operatorname{argmin}} f(x) + \frac{\mu_j}{2}\|x - Y_j\|_2^2$
 - 3: $Y_{j+1} \leftarrow X_{j+1}$ (or $Y_{j+1} \leftarrow Y_0$)
 - 4: $\mu_{j+1} \leftarrow \beta\mu_j$
 - 5: **end for**
-

Moreau-Yosida regularization

Moreau envelope $h(Y) = \min_{x \in C} f(x) + \frac{\mu}{2} \|x - Y\|_2^2$

Moreau proximity operator $X_Y = \operatorname{argmin}_{x \in C} f(x) + \frac{\mu}{2} \|x - Y\|_2^2$

Theorem

h is continuously differentiable with gradient

$$\nabla h(Y) = \mu(Y - X_Y)$$

The gradient is Lipschitz continuous with constant $L = \mu$

Minimizing h by gradient descent \rightarrow proximal point algorithm (PPA) (Rockafellar, 70s)

Accelerated continuation (Nesterov style)

Algorithm 4 Accelerated continuation

Require: $Y_0, \mu_0 > 0$

1: $X_0 \leftarrow Y_0$

2: **for** $j = 0, 1, 2, \dots$ **do**

3: $X_{j+1} \leftarrow \operatorname{argmin}_{\mathcal{A}(x)+b \in \mathcal{K}} f(x) + \frac{\mu_j}{2} \|x - Y_j\|_2^2$

4: $Y_{j+1} \leftarrow X_{j+1} + \frac{j}{j+3} (X_{j+1} - X_j)$

5: (optional) increase or decrease μ_j

6: **end for**

Keep $\mu_j \equiv \mu$ so subproblems quick to solve

Warm-start subproblems

For small μ , typically 5 iterations

Simple vs. accelerated continuation: LASSO example

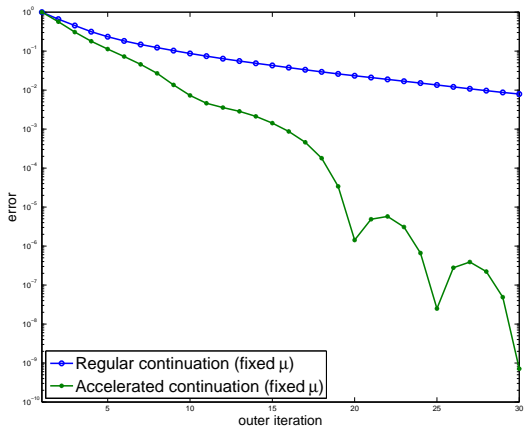


Figure: $\|x_k - x^*\| / \|x_0 - x^*\|$ vs. outer iteration count

Effect of perturbation

Nice surprise:

Linear programs (ex. Dantzig, Basis Pursuit) have exact penalty

Theorem (Exact penalty)

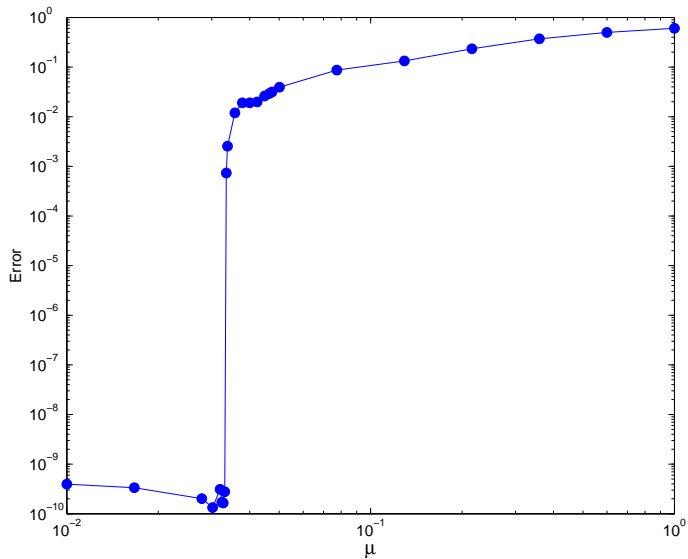
- *Arbitrary LP with objective $\langle c, x \rangle$ and with opt. solution*
- *Perturbed LP with objective $\langle c, x \rangle + \frac{1}{2}\mu\|x - x_0\|_Q^2$, $Q \succeq 0$*

There is $\mu_0 > 0$ s.t. for $0 < \mu \leq \mu_0$, any solution to perturbed problem is a solution to LP

- Special case (BP): Yin ('10)
- More general result: Friedlander and Tseng ('07)
- Combine with continuation \implies finite termination
Known since Bertsekas '75, Polyak and Tretjakov '74

Illustration

Exact penalty for Dantzig Selector



Parameters

Lipschitz Gradient

$$f(y) \leq f(x) + \langle y - x, \nabla f(x) \rangle + \frac{L}{2} \|x - y\|_2^2$$

Strong Convexity

$$f(y) \geq f(x) + \langle y - x, \nabla f(x) \rangle + \frac{m_f}{2} \|x - y\|_2^2$$

If $\nabla^2 f$ exists, equivalent to

$$m_f I \preceq \nabla^2 f \preceq L I$$

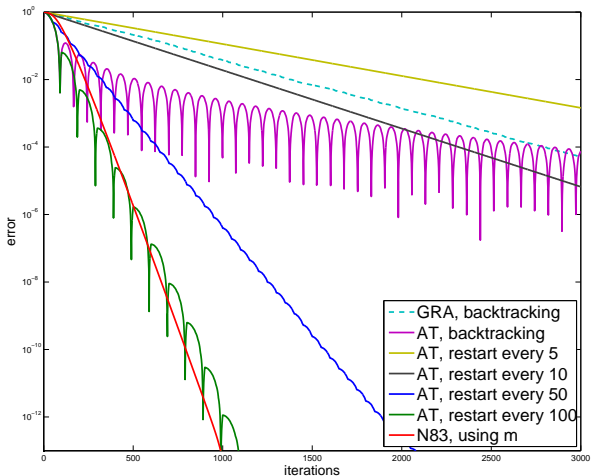
Goal: user needs no knowledge of m_f and L

- For L , trick: backtracking line search
- For m_f , trick: restart

Restart

Problem: accelerated schemes don't **automatically** take advantage of strong convexity.

i.e. m_f unknown \implies no linear convergence



Restart

Convergence of accelerated method:

$$f(x_k) - f^* \leq \frac{L}{k^2} \|x^* - x_0\|^2$$

If f is strongly convex with parameter m_f ,

$$\|x_k - x^*\|^2 \leq \frac{2L}{m_f} \frac{1}{k^2} \|x^* - x_0\|^2$$

With restart, x_0 is x_k of a previous sequence. Do this j times.

$$\|x_{jk} - x^*\| \leq \left(\sqrt{\frac{2L}{m_f} \frac{1}{k}} \right)^j \|x^* - \hat{x}_0\|$$

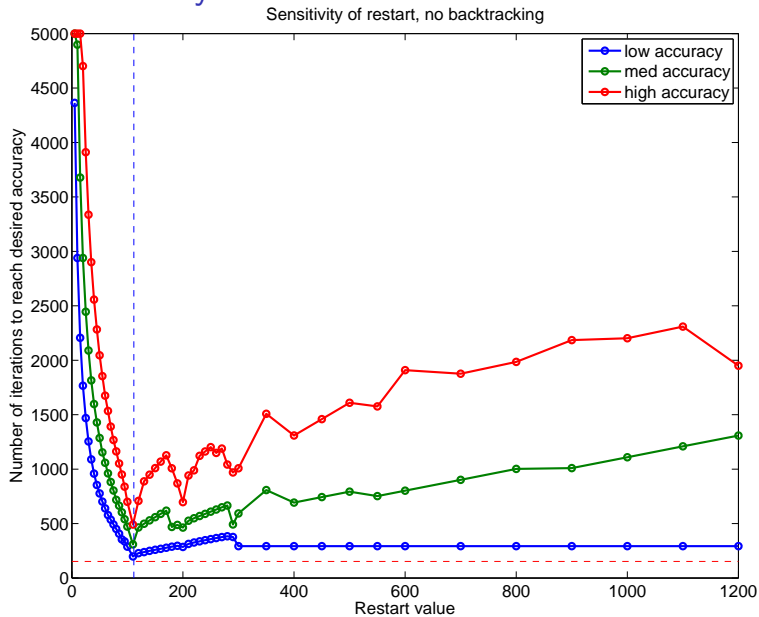
This is linear convergence with rate $\rho = \left(\sqrt{\frac{2L}{m_f} \frac{1}{k}} \right)^{1/k}$.

$$k_{\text{opt}} = e \sqrt{\frac{2L}{m_f}}$$

See PARNES paper (Gu, Lim, Wu 2009), Nesterov 2007.

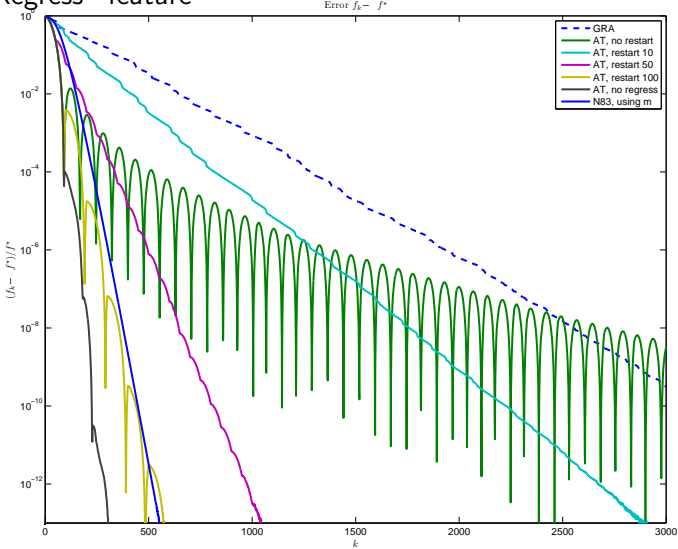
Goes back to Powell (1977) for non-linear CG.

Restart: sensitivity



Restart: improvements

“No Regress” feature



Basic convergence results

$$\min_Y \underbrace{\min_x f(x) + \frac{\mu}{2} \|x - Y\|^2}_{\phi(Y)}$$

Inner. Dual function $g_k \rightarrow g^*$ at rate $\mathcal{O}(\mu^{-1}/k^2)$

Outer. $\phi_J \rightarrow \phi^*$ at rate $\mathcal{O}(\mu/J^2)$ if inner iterate solved exactly.

Trade-off between μ large and μ small.

Goal: analyze allowing **inexact** solves.

O. Güler, 1992: Accelerated (and robust) proximal point.

Let $x_\mu^* = \operatorname{argmin} f(x) + \mu/2 \|x - Y_J\|^2$. If

$$\|x_k - x_\mu^*\|_2 \leq \frac{C}{J^{1.5}}$$

then outer iteration still converges in $\mathcal{O}(\mu/J^2)$.

How to get this?

Bounding the error

Let $x_\mu^\star = \operatorname{argmin} f(x) + \mu/2\|x - Y_J\|^2$. Want:

$$\|x_k - x_\mu^\star\|_2 \leq \frac{C}{J^{1.5}}$$

Note: primal is strongly convex, so bound f_k or x_k .
How to go from dual to primal? No duality gap.

Combettes, Dũng, Vũ 2010: to bound primal, either bound $\|\lambda_k - \lambda^\star\|$, or $\|\nabla g_{\text{sm}}(\lambda_k) - \nabla g_{\text{sm}}(\lambda^\star)\|$

Implications

- Easy results: convergence (and rate) if dual is unconstrained (e.g. BP) and solve via gradient descent. previously known: Cai, Osher, Shen 2008 for linear Bregman ($x_0 = 0$, gradient descent), no rate.
- Also known: convergence (no rate) if dual is solved via proximal gradient descent, since dual variable converges (Beck, Teboulle 2010).

Bounding the error

General case: unknown.

Issue is that gradient mapping converges, but ∇g_{sm} need not.

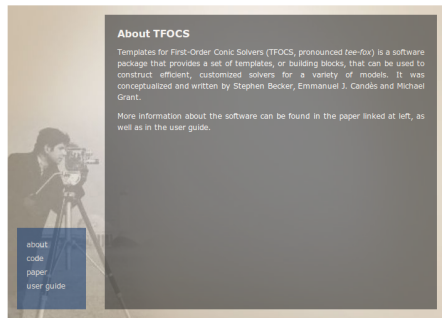
Ideas?

Other future work: given bounds, optimize choice of μ .

Software release

- Paper
- User guide
- Software (MATLAB)
 - solvers
 - many simple examples
 - a few real-world examples
 - continuation wrappers
 - compatible with SPOT
- Parameters: any $\mu > 0$

TFOCS Templates for First-Order Conic Solvers



The screenshot shows a webpage for TFOCS. On the left, there is a navigation menu with the following items: about, code, paper, and user guide. The main content area is titled 'About TFOCS' and contains the following text: 'Templates for First-Order Conic Solvers (TFOCS, pronounced fee-fox) is a software package that provides a set of templates, or building blocks, that can be used to construct efficient, customized solvers for a variety of models. It was conceptualized and written by Stephen Becker, Emmanuel J. Candès and Michael Grant. More information about the software can be found in the paper linked at left, as well as in the user guide.'

©2010, Caltech.

<http://tfocs.stanford.edu>

Example in TFOCS

Basis Pursuit Denoising BP_ϵ , analysis

$$\min_x \|Wx\|_1 \quad \text{subject to} \quad \|Ax - b\|_2 \leq \epsilon$$

```
prox   = { prox_l2( epsilon ), proj_linf };  
linear = { A, -b; W, 0 };  
x      = tfocs_SCD( [], linear, prox, mu, x0 );
```

Easy to add constraints, e.g. $x \geq 0$

```
prox   = { prox_l2( epsilon ), proj_linf, proj_Rplus };  
linear = { A, -b; W, 0; 1, 0 };
```

Of course, this is also builtin...

```
x = solver_sBPDN_W(A,W,b,epsilon,mu)
```

No Lipschitz constant or step size needed!