Projective Splitting Methods for Decomposing Convex Optimization Problems

Jonathan Eckstein
Rutgers University, New Jersey, USA

Various portions of this talk describe joint work with
Patrick Combettes — NC State University, USA
Patrick Johnstone — Rutgers University, USA
Benar F. Svaiter — IMPA, Brazil
Also: Jean-Paul Watson — Sandia National Labs, USA
David L. Woodruff — UC Davis, USA

RUTGERS

RUTGERS
Rutgers Business School
Newark and New Brunswick

Funded in part by NSF grants CCF-1115638, CCF-1617617, and AFOSR grant FA9550-15-1-0251

Introductory Remarks

- I did some of the earlier work on an optimization algorithm called the ADMM (the Alternating Direction Method of Multipliers)
  - But not the earliest work

Introductory Remarks

- I did some of the earlier work on an optimization algorithm called the ADMM (the Alternating Direction Method of Multipliers)
  - o But not the earliest work

- I know that the ADMM has been used in image processing because about 15 years ago I started being asked to referee a deluge of papers with this picture:

Introductory Remarks

- I did some of the earlier work on an optimization algorithm called the ADMM (the Alternating Direction Method of Multipliers)
  - o But not the earliest work

- I know that the ADMM has been used in image processing because about 15 years ago I started being asked to referee a deluge of papers with this picture:

Introductory Remarks

- I did some of the earlier work on an optimization algorithm called the ADMM (the Alternating Direction Method of Multipliers)
  - o But not the earliest work

- I know that the ADMM has been used in image processing because about 15 years ago I started being asked to referee a deluge of papers with this picture:

- Today I want to talk about an algorithm that uses similar building blocks to the ADMM but is much more flexible

More General Problem Setting

The algorithms in this talk can work for monotone inclusion problems of the form

$$0 \in \sum_{i=1}^{n} G_i^* T_i(G_i x)$$

where

- $\mathcal{H}_0, \ldots, \mathcal{H}_n$ are real Hilbert spaces
- $T_i : \mathcal{H}_i \rightrightarrows \mathcal{H}_i$ are (generally set-valued) maximal monotone operators, $i = 1, \ldots, n$
- $G_i : \mathcal{H}_0 \rightrightarrows \mathcal{H}_i$ are bounded linear maps, $i = 1, \ldots, n$

However, for this talk we will restrict ourselves to…

A General Convex Optimization Problem

$$\min_x \left\{ \sum_{i=1}^n f_i(G_i x) \right\}$$

- For $i = 1, \ldots, n$, $f_i : \mathbb{R}^{p_i} \to \mathbb{R} \cup \{+\infty\}$ is closed proper convex
- For $i = 1, \ldots, n$, G_i is a $p_i \times m$ real matrix
- Assume you have a class of such problems that is not suitable for standard LP/NLP solvers because either
  - o The problems are very large
  - o They is fairly large but also dense

Subgradient Maps of Convex Functions, Monotonicity

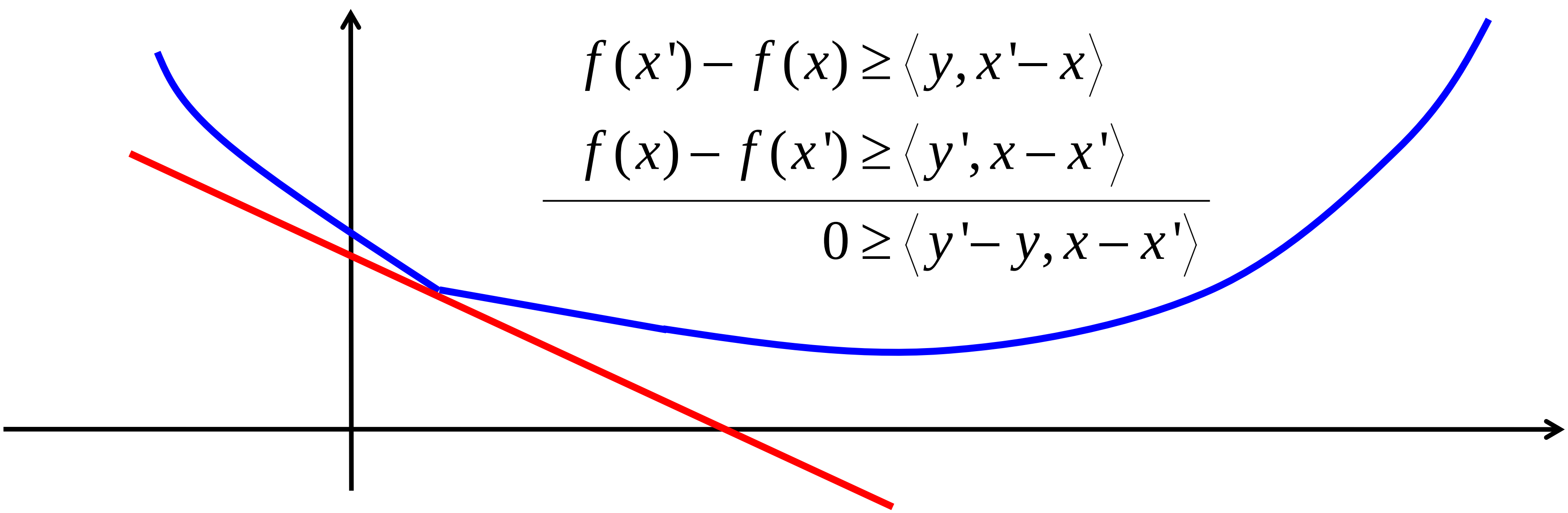
The *subgradient map* ∂f of a convex function $f : \mathbb{R}^p \to \mathbb{R} \cup \{+\infty\}$ is given by

$$\partial f(x) = \left\{ y \,\middle|\, f(x') \geq f(x) + \langle y, x' - x \rangle \ \forall x' \in \mathbb{R}^p \right\}.$$

This has the property that

$$y \in \partial f(x), y' \in \partial f(x') \quad \Rightarrow \quad \langle x - x', y - y' \rangle \geq 0$$

Proof:



$$\begin{array}{r} f(x') - f(x) \geq \langle y, x' - x \rangle \\ f(x) - f(x') \geq \langle y', x - x' \rangle \\ \hline 0 \geq \langle y' - y, x - x' \rangle \end{array}$$

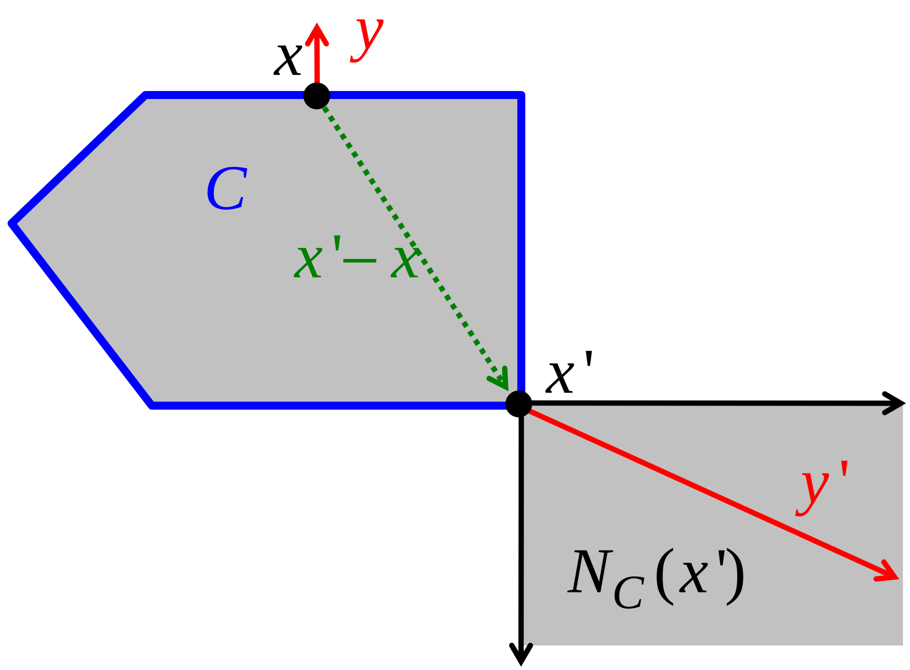
Normal Cone Maps

The *indicator function* of a nonempty closed convex set C is

$$\delta_C(x) = \begin{cases} 0, & x \in C \\ +\infty, & x \notin C \end{cases}$$

Its subgradient map is the *normal cone* map N_C of C:

$$\partial\delta_C(x) = N_C(x) = \begin{cases} \{ y \mid \langle y, x' - x \rangle \le 0 \ \forall x' \in C \}, & x \in C \\ \emptyset & x \notin C \end{cases}$$



$$\begin{array}{rr} & \langle y, x' - x \rangle \le 0 \\ + & \langle y', x - x' \rangle \le 0 \\ \hline & \langle y' - y, x - x' \rangle \le 0 \end{array}$$

A Subgradient Chain Rule

- Suppose $f : \mathbb{R}^p \to \mathbb{R} \cup \{+\infty\}$ is closed proper convex
- Suppose G is a $p \times m$ real matrix

Then for any x,

$$\partial(f \circ G)(x) \supseteq G^\top \partial f\left(Gx\right) = \left\{G^\top y \,\middle|\, y \in \partial f(Gx)\right\}$$

and "usually"

$$\partial(f \circ G)(x) = G^\top \partial f\left(Gx\right)$$

An Optimality Condition

Let's go back to

$$\min_x \left\{ \sum_{i=1}^n f_i(G_i x) \right\}$$

Suppose we have $z \in \mathbb{R}^m, w_1 \in \mathbb{R}^{p_1}, \ldots, w_n \in \mathbb{R}^{p_n}$ such that

$$\boxed{\begin{array}{ll} w_i \in \partial f_i(G_i z) & i = 1, \ldots, n \\ \displaystyle\sum_{i=1}^n G_i^\top w_i = 0 & \end{array}}$$

The chain rule then implies that $0 \in \partial \left[\sum_{i=1}^n f_i \circ G_i \right](z)$, so...

z is a solution to our problem

- This is always a sufficient optimality condition
- It's "usually" necessary as well
- The w_i are the Lagrange multipliers / dual variables

The Primal-Dual Solution Set (Kuhn-Tucker Set)

$$\mathcal{S} = \left\{ (z, w_1, \ldots, w_n) \,\middle|\, (\forall i = 1, \ldots n)\; w_i \in \partial f_i(G_i z),\; \sum_{i=1}^{n} G_i^{\mathsf{T}} w_i = 0 \right\}$$

Or, if we assume that $p_n = m, G_n = \mathrm{Id}_{\mathbb{R}^m}$,

$$\mathcal{S} = \left\{ (z, w_1, \ldots, w_{n-1}) \,\middle|\, (\forall i = 1, \ldots n-1)\; w_i \in \partial f_i(G_i z),\; -\sum_{i=1}^{n-1} G_i^{\mathsf{T}} w_i \in \partial f_n(z) \right\}$$

- This is the set of points satisfying the optimality conditions
- Standing assumption: $\mathcal{S}$ is nonempty
- Essentially in E & Svaiter 2009:

$\mathcal{S}$ is a closed convex set

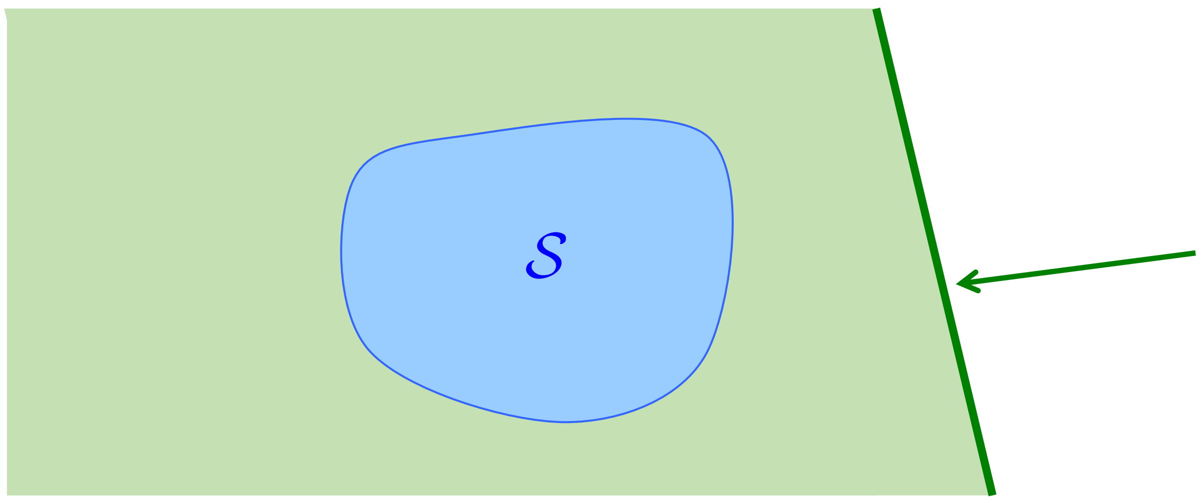
- In the $p_n = m, G_n = \mathrm{Id}_{\mathbb{R}^m}$ case, streamline notation:

For $\boldsymbol{w} \in \mathcal{H}_1 \times \cdots \times \mathcal{H}_{n-1}$, let $w_n \triangleq -\sum_{i=1}^{n-1} G_i^* w_i$

Valid Inequalities for $\mathcal{S}$

- Take some $x_i, y_i \in \mathbb{R}^{p_i}$ such that $y_i \in \partial f_i(x_i)$ for $i = 1, \ldots, n$
- If $(z, \boldsymbol{w}) \in \mathcal{S}$, then $w_i \in \partial f_i(G_i z)$ for $i = 1, \ldots, n$
- So, $\left\langle x_i - G_i z, y_i - w_i \right\rangle \geq 0$ for $i = 1, \ldots, n$
- Negate and add up:

$$\varphi(z, \boldsymbol{w}) = \sum_{i=1}^{n} \left\langle G_i z - x_i , y_i - w_i \right\rangle \leq 0 \qquad \forall (z, \boldsymbol{w}) \in \mathcal{S}$$



$H = \left\{ p \,\middle|\, \varphi(p) = 0 \right\}$

$\varphi(p) \leq 0 \quad \forall p \in \mathcal{S}$

Confirming that φ is Affine

The quadratic terms in $\varphi(z, \boldsymbol{w})$ take the form

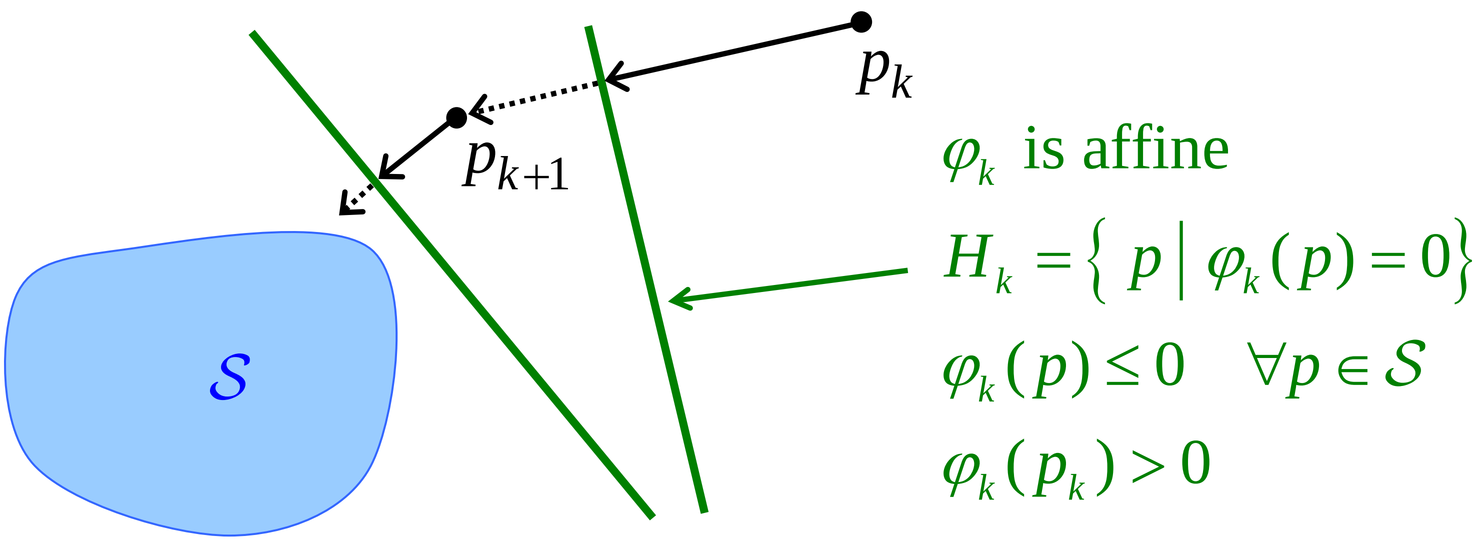
$$\sum_{i=1}^{n} \langle G_i z, -w_i \rangle = \sum_{i=1}^{n} \langle z, -G_i^{\mathsf{T}} w_i \rangle = \left\langle z, -\sum_{i=1}^{n} G_i^{\mathsf{T}} w_i \right\rangle = \langle z, -0 \rangle = 0$$

- Also true in the $p_n = m, G_n = \mathrm{Id}_{\mathbb{R}^m}$ case where we drop the n^{th} index
  - Slightly different proof, same basic idea

Generic Projection Method for a Closed Convex Set $\mathcal{S}$ in a Hilbert Space $\mathcal{H}$

Apply the following general template:

- Given $p^k \in \mathcal{H}$, choose some affine function φ_k with $\varphi_k(p) \le 0 \ \forall p \in \mathcal{S}$
- Project p^k onto $H_k = \left\{ p \,\middle|\, \varphi_k(p) = 0 \right\}$, possibly with an overrelaxation factor $\lambda_k \in [\varepsilon, 2-\varepsilon]$, giving p_{k+1}, and repeat…



In our case: $\mathcal{H} = \mathbb{R}^m \times \mathbb{R}^{p_1} \times \cdots \times \mathbb{R}^{p_n}$ and we find φ_k by picking some $x_i^k, y_i^k \in \mathbb{R}^{p_i} : y_i^k \in \partial f_i(x_i^k), i = 1, \ldots, n$ and using the construction above

General Properties of Projection Algorithms

Proposition. In such algorithms, assuming that $\mathcal{S} \neq \emptyset$,

- $\left\{\left\| p^k - p^* \right\|\right\}$ is nonincreasing for all $p^* \in \mathcal{S}$
- $\{p^k\}$ is bounded
- $p^{k+1} - p^k \to 0$
- If $\{\nabla \varphi_k\}$ is bounded, then $\limsup_{k\to\infty} \left\{\varphi_k(p^k)\right\} \leq 0$
- If all limit points of $\{p^k\}$ are in $\mathcal{S}$, then $\{p^k\}$ converges to a point in $\mathcal{S}$

The first three properties hold no matter how badly we choose φ_k

The idea is to pick φ_k so that the stipulations of the last two properties hold – then we have a convergent algorithm

If we pick φ_k badly, we may "stall"

Selecting the Right φ_k

- Selecting φ_k involves picking some $x_i^k, y_i^k \in \mathbb{R}^{p_i} : y_i^k \in \partial f_i(x_i^k)$, $i = 1, \ldots, n$

- It turns out there are many ways to pick x_i^k, y_i^k so that the last two properties of the proposition are satisfied

- One fundamental thing we would like is

$$\varphi_k(z^k, \boldsymbol{w}^k) \triangleq \sum_{i=1}^{n} \left\langle G_i z^k - x_i^k, y_i^k - w_i^k \right\rangle \geq 0$$

with strict inequality if $(z^k, \boldsymbol{w}^k) \notin \mathcal{S}$

- The oldest suggestion is "prox" (E & Svaiter 2008 & 2009)

The Prox Operation

- Suppose we have a convex function $f : \mathbb{R}^p \to \mathbb{R} \cup \{+\infty\}$
- Take any vector $r \in \mathbb{R}^p$ and scalar $c > 0$ and solve

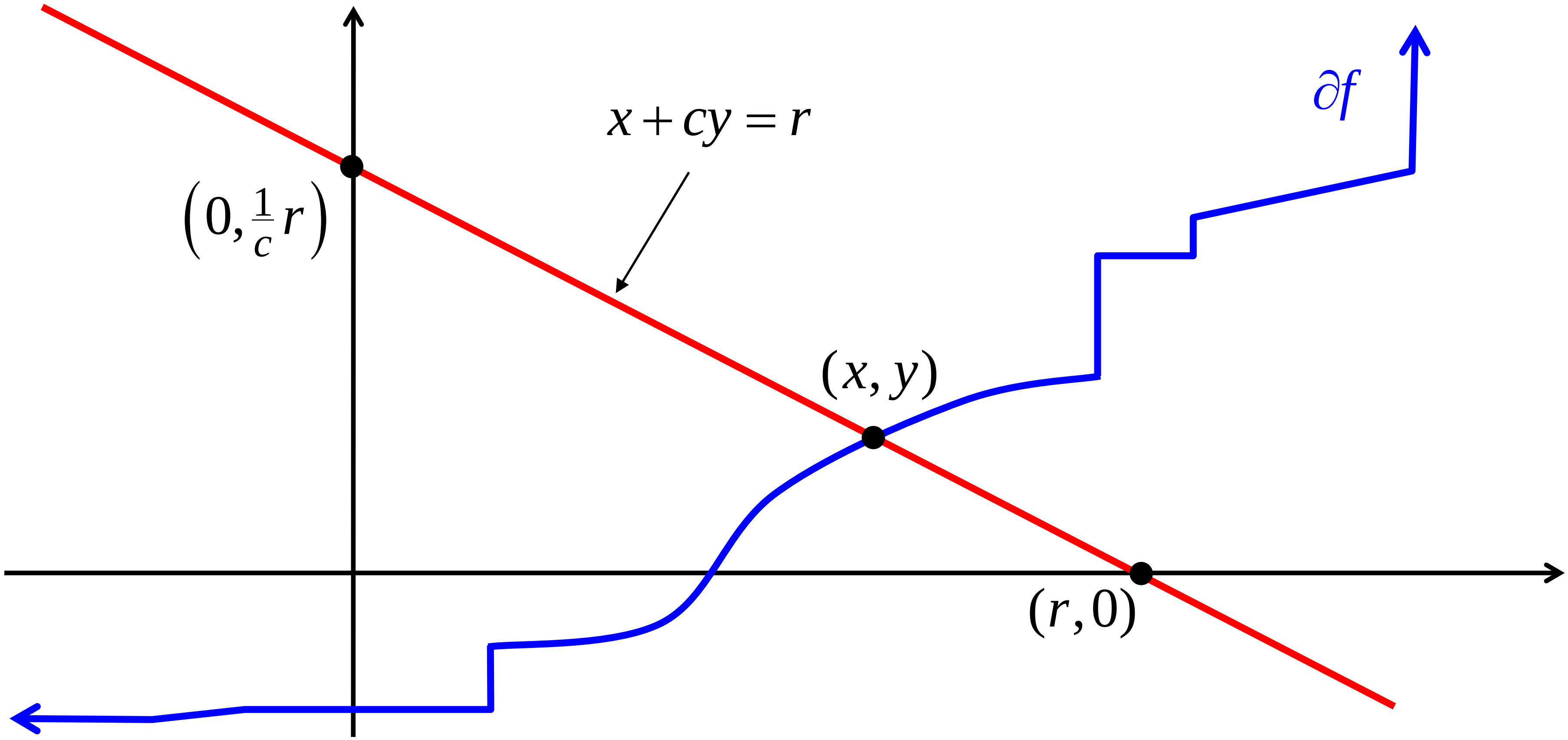
$$x = \arg\min_{x' \in \mathbb{R}^p} \left\{ f(x') + \frac{1}{2c} \left\| x' - r \right\|^2 \right\}$$

- Optimality condition for this minimization is

$$0 \in \partial f(x) + \frac{1}{c}(x - r)$$

- So we have $y \triangleq \frac{1}{c}(r - x) \in \partial f(x)$
- And $x + cy = x + c \cdot \frac{1}{c}(r - x) = r$
- So, we just found $x, y \in \mathbb{R}^p$ such that $y \in \partial f(x)$ and $x + cy = r$
- Call this $\mathrm{Prox}_{\partial f}^{c}(r)$

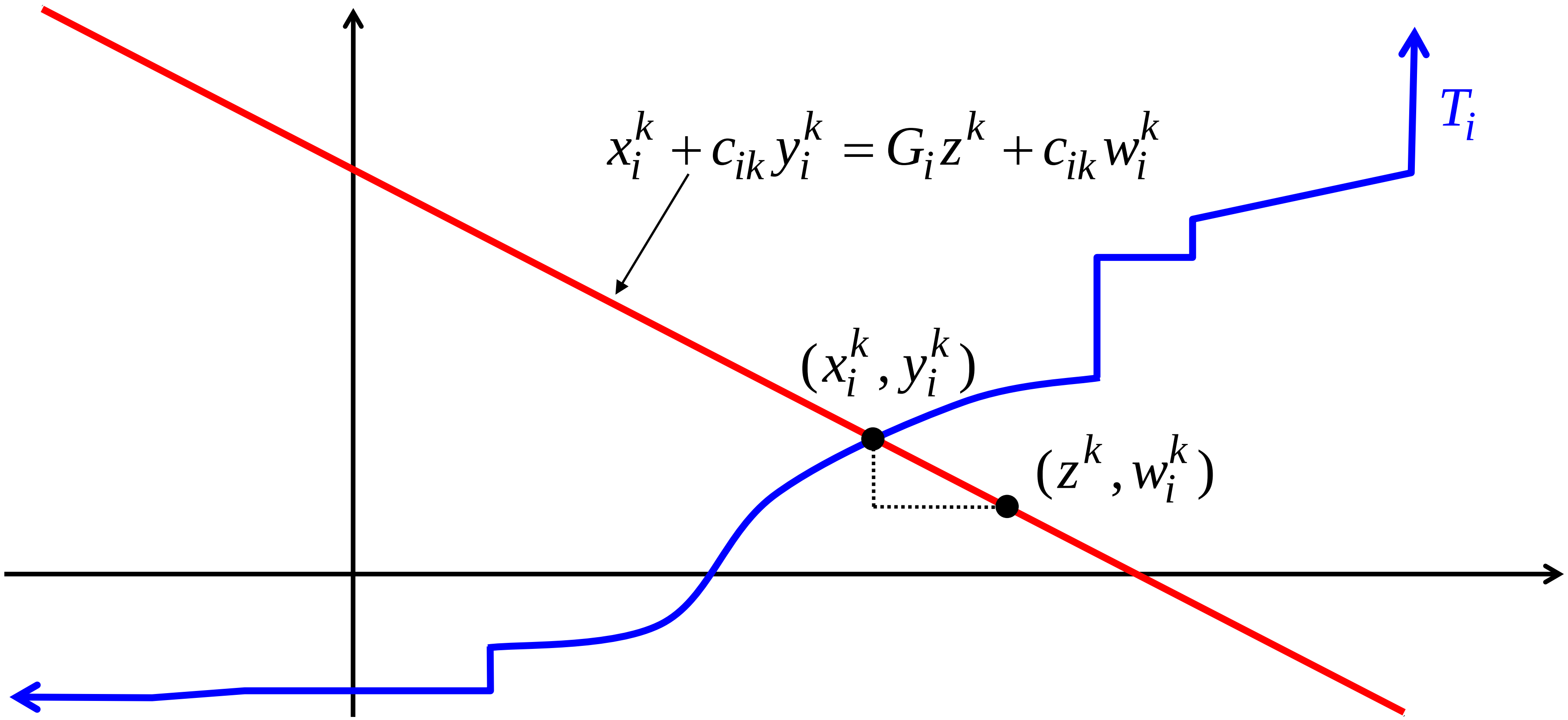
Picture



- The choice of $x, y \in \mathbb{R}^p$ such that $y \in \partial f(x)$ and $x + cy = r$ must be unique; otherwise ∂f would not be monotone
- If f is closed and proper, then this solution must exist
- Any vector $r \in \mathbb{R}^p$ can then be written in a unique way as $x + cy = r$, where $y \in \partial f(x)$
    - o Generalizes projection to a subspace and its complement

Prox Does the Job!

- We have an iterate $p^k = (z^k, \mathbf{w}^k) = (z^k, w_1^k, \ldots, w_n^k)$
- Take any $c_{1k}, \ldots, c_{nk} > 0$ and consider $(x_i^k, y_i^k) = \mathrm{Prox}_{\partial f_i}^{c_{ik}}(G_i z^k + c_{ik} w_i^k)$



- Then $x_i^k + c_{ik} y_i^k = G_i z^k + c_{ik} w_i^k \quad \Leftrightarrow \quad c_{ik}(y_i^k - w_i^k) = G_i z^k - x_i^k$
- Implying $\left\langle G_i z^k - x_i^k, y_i^k - w_i^k \right\rangle = c_{ik} \left\| G_i z^k - x_i^k \right\|^2 = c_{ik}^{-1} \left\| y_i^k - w_i^k \right\|^2 \geq 0$

Prox Finishes the Job

From

$$\left\langle G_i z^k - x_i^k, y_i^k - w_i^k \right\rangle = c_{ik} \left\| G_i z^k - x_i^k \right\|^2 = c_{ik}^{-1} \left\| y_i^k - w_i^k \right\|^2 \geq 0$$

we have that

$$\sum_{i=1}^{n} \left\langle G_i z^k - x_i^k, y_i^k - w_i^k \right\rangle \geq 0$$

and this inequality is strict unless $G_i z^k = x_i^k$ and $y_i^k = w_i^k$ for all i, which means that $(z^k, \boldsymbol{w}^k) \in \mathcal{S}$

The entire convergence proof follows from this same relationship.

A First Algorithm

- These conditions allow one to prove that the cuts are "deep enough" and we obtain convergence

Starting with an arbitrary $(z^0, w_1^0, \ldots, w_n^0)$:

For $k = 0, 1, 2, \ldots$

1. For $i = 1, \ldots, n$, compute $(x_i^k, y_i^k) = \mathrm{Prox}_{T_i}^{c_{i,k}}(G_i z^k + c_i w_i^k)$
   (Process operators: Decomposition Step)

2. Define $\varphi_k(z, w_1, \ldots, w_n) = \sum_{i=1}^{n} \left\langle G_i z - x_i^k, y_i^k - w_i \right\rangle$

3. Compute $(z^{k+1}, w_1^{k+1}, \ldots, w_n^{k+1})$ by projecting $(z^{k+1}, w_1^k, \ldots, w_n^k)$ onto the halfspace $\varphi_k(z, w_1, \ldots, w_n) \le 0$
   (possibly with some overrelaxation) (Coordination Step)

- This simple algorithm combines aspects of E & Svaiter 2009 and Alotaibi et al. 2014

Including the Details (Version 1: general case)

- Choose any $0 < \lambda_{\min} \le \lambda_{\max} < 2$
- For $k = 1, 2, \ldots$

Process operators to find $x_i^k, y_i^k \in \mathbb{R}^{p_i} : \; y_i^k \in \partial f_i(x_i^k), i = 1, \ldots, n$

$(u_1^k, \ldots, u_n^k) = \operatorname{proj}_{\mathcal{G}}(x_1^k, \ldots, x_n^k)$, where $\mathcal{G} = \left\{ (w_1, \ldots, w_n) \,\middle|\, \sum_{i=1}^n G_i^\top w_i = 0 \right\}$

$$v^k = \sum_{i=1}^n G_i^\top y_i^k$$

$$\theta_k = \frac{\max\left\{ \sum_{i=1}^n \left\langle G_i z - x_i^k, y_i^k - w_i \right\rangle, 0 \right\}}{\left\| v^k \right\|^2 + \sum_{i=1}^n \left\| u_i^k \right\|^2}$$

Pick any $\lambda \in [\lambda_{\min}, \lambda_{\max}]$

$$z^{k+1} = z^k - \lambda_k \theta_k v^k$$

$$w_i^{k+1} = w_i^k - \lambda_k \theta_k u_i^k, \quad i = 1, \ldots, n$$

Including the Details (Version 2: $p_n = m, G_n = \mathrm{Id}_{\mathbb{R}^m}$)

- Choose any $0 < \lambda_{\min} \le \lambda_{\max} < 2$
- For $k = 1, 2, \ldots$

Process operators to find $x_i^k, y_i^k \in \mathbb{R}^{p_i}: \; y_i^k \in \partial f_i(x_i^k), i = 1, \ldots, n$

$$u_i^k = x_i^k - G_i x_n^k, \quad i = 1, \ldots, n-1$$

$$v^k = \sum_{i=1}^{n-1} G_i^\top y_i^k + y_n^k$$

$$\theta_k = \frac{\max\left\{\sum_{i=1}^{n} \left\langle G_i z - x_i^k, y_i^k - w_i \right\rangle, 0\right\}}{\left\|v^k\right\|^2 + \sum_{i=1}^{n} \left\|u_i^k\right\|^2}$$

Pick any $\lambda \in [\lambda_{\min}, \lambda_{\max}]$

$$z^{k+1} = z^k - \lambda_k \theta_k v^k$$

$$w_i^{k+1} = w_i^k - \lambda_k \theta_k u_i^k \quad i = 1, \ldots, n-1$$

Many Variations Possible in "Process Operators"

1. Inexact processing: the prox operations may be performed approximately using a relative error criterion
   - E & Svaiter 2009

2. Block iterations: you do not have to process every operator at every iteration; you may process some subset and let $(x_i^k, y_i^k) = (x_i^{k-1}, y_i^{k-1})$ for the rest, so long as you process each operator at least once every M iterations
   - Combettes & E 2018, E 2017

3. Asynchrony: you may process operators using (boundedly) old information $(z^{d(i,k)}, \boldsymbol{w}^{d(i,k)})$, where $k \geq d(i,k) \geq k - K$
   - Combettes & E 2018, E 2017

4. Non-prox steps: For Lipschitz continuous gradients, procedures using one or two gradient steps may be substituted for the prox operations
   - Johnstone and E 2018, 2019
     also see Tranh-Dinh and Vũ 2015

+ "mix and match"

Another Variation: Primal-Dual Scaling

- Method performs projections in primal-dual space
- Consider scaling the problem: $f_i \rightarrow \alpha f_i, \;\; \alpha > 0$
- If α is large, dual convergence will be emphasized over primal
- If α is small, primal convergence will be emphasized over dual

- To compensate, use the inner product on $\mathcal{H}^{n+1}$ given by

$$\left\langle (z, w_1, \ldots, w_n), (z', w_1', \ldots, w_n') \right\rangle_\gamma = \gamma \left\langle z, z' \right\rangle + \sum_{i=1}^{n} \left\langle w_i, w_i' \right\rangle$$

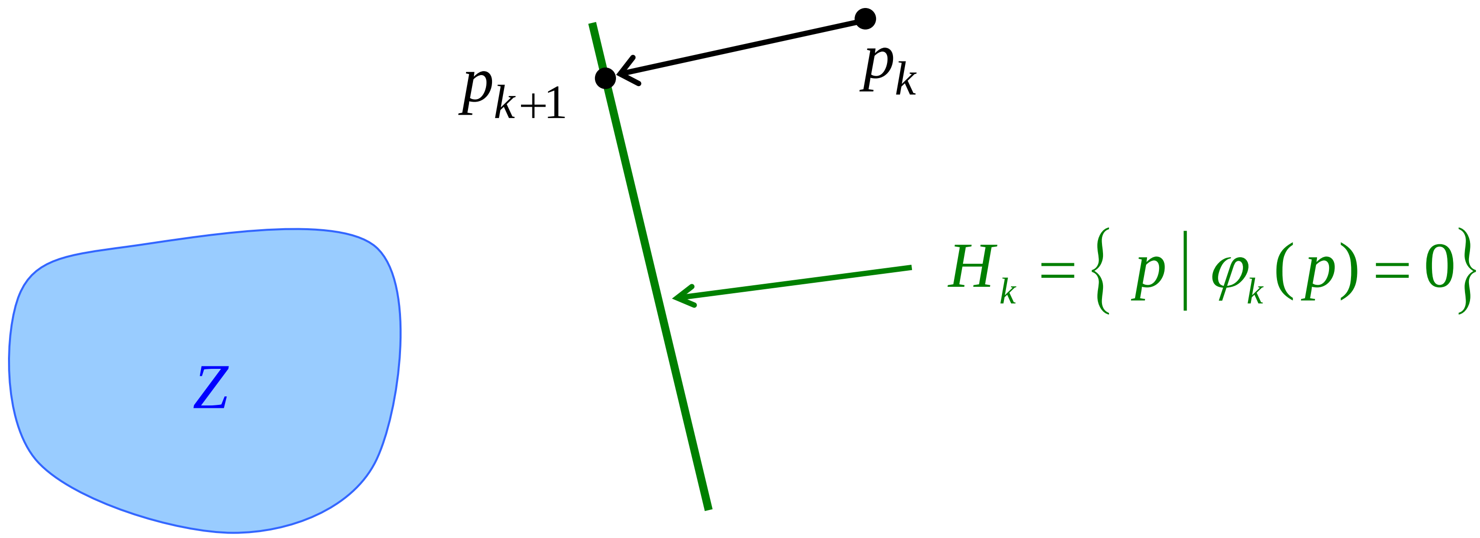
  and corresponding norm, for any scalar $\gamma > 0$

- In the ADMM and related methods the penalty parameter can compensate for problems scaling, but projective splitting is different

An Implementation Idea: Greedy Block Selection

- Our separating hyperplane is

$$\varphi_k(z, w_1, \ldots, w_{n-1}) = \sum_{i=1}^{n} \left\langle G_i z - x_i^k, y_i^k - w_i \right\rangle = 0$$



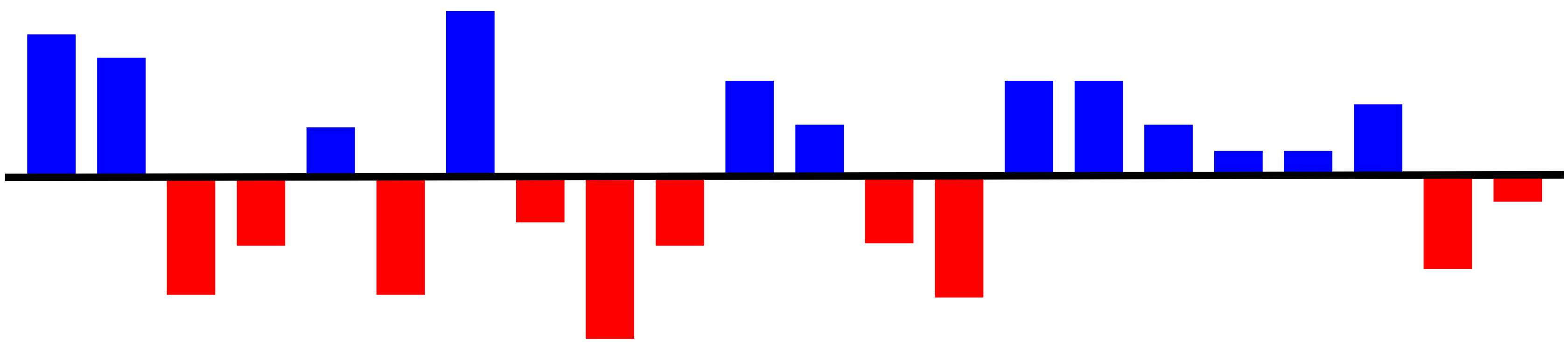
- If we project without any overrelaxation, we will have

$$\varphi_k(z^{k+1}, w_1^{k+1}, \ldots, w_{n-1}^{k+1}) = \sum_{i=1}^{n} \left\langle G_i z^{k+1} - x_i^k, y_i^k - w_i^{k+1} \right\rangle = 0$$

Greedy Block Selection (2a)

$$\sum_{i=1}^{n} \left\langle G_i z^{k+1} - x_i^k, y_i^k - w_i^{k+1} \right\rangle = 0$$

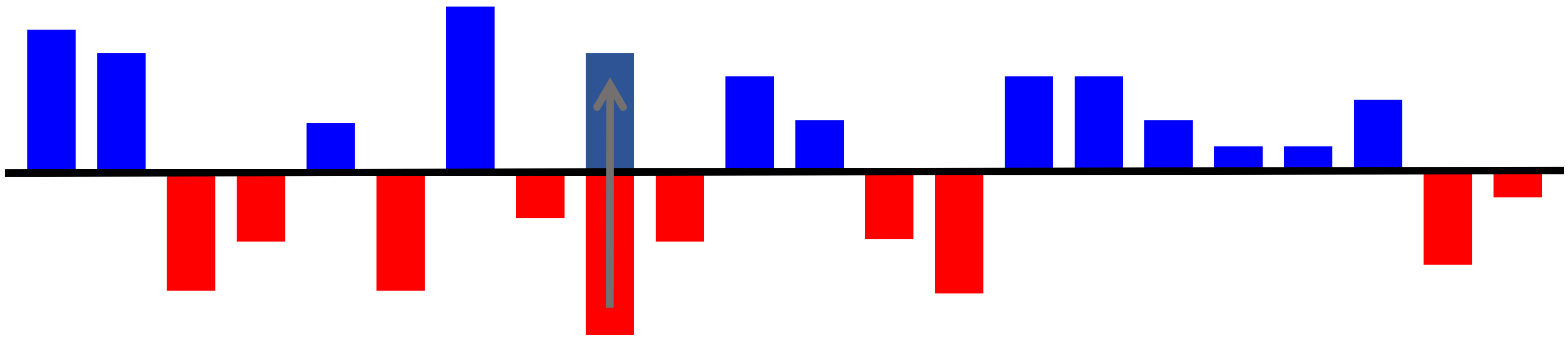
- If all the $\varphi_{ik} = \left\langle G_i z^{k+1} - x_i^k, y_i^k - w_i^{k+1} \right\rangle$ are zero, we are in $\mathcal{S}$
- Otherwise, some are positive and some are negative



Greedy Block Selection (2b)

$$\sum_{i=1}^{n} \left\langle G_i z^{k+1} - x_i^k , y_i^k - w_i^{k+1} \right\rangle = 0$$

- If all the $\varphi_{ik} = \left\langle G_i z^{k+1} - x_i^k , y_i^k - w_i^{k+1} \right\rangle$ are zero, we are in $\mathcal{S}$
- Otherwise, some are positive and some are negative

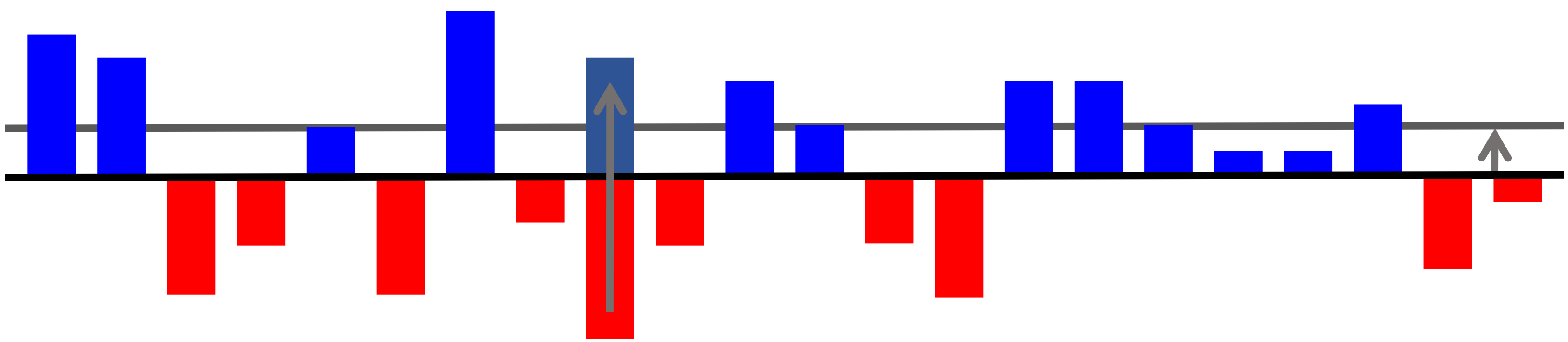


- Pick a block with $\varphi_{ik} < 0$
- Processing block i results in $\varphi_{ik} \geq 0$

Greedy Block Selection (2c)

$$\sum_{i=1}^{n} \left\langle G_i z^{k+1} - x_i^k, y_i^k - w_i^{k+1} \right\rangle = 0$$

- If all the $\varphi_{ik} = \left\langle G_i z^{k+1} - x_i^k, y_i^k - w_i^{k+1} \right\rangle$ are zero, we are in $\mathcal{S}$
- Otherwise, some are positive and some are negative



- Pick a block with $\varphi_{ik} < 0$
- Processing block i results in $\varphi_{ik} \geq 0$
- Will make the entire sum positive again
- $\Rightarrow$ Can cut off the current point by processing just one block

Greedy Block Selection (3)

- A simple "greedy" heuristic: prioritize the block i with the most negative φ_{ik}

This ignores several things:

- How large will φ_{ik} become after we process the block?
- The projection formula onto the hyperplane is

$$p_{k+1} = p_k - \left(\frac{\varphi_k(p_k)}{\left\|\nabla \varphi_k\right\|^2} \right) \nabla \varphi_k$$

So, the length of the step is

$$\frac{\varphi_k(p_k)}{\left\|\nabla \varphi_k\right\|}$$

The heuristic makes some attempt to obtain a large numerator, but ignores the denominator

Computational Experiments: LASSO

LASSO problems:

$$\min_{x\in\mathbb{R}^d}\left\{\tfrac{1}{2}\left\|Qx-b\right\|^2+\lambda\left\|x\right\|_1\right\}$$

Partition Q into r blocks of rows, set $n=r+1$

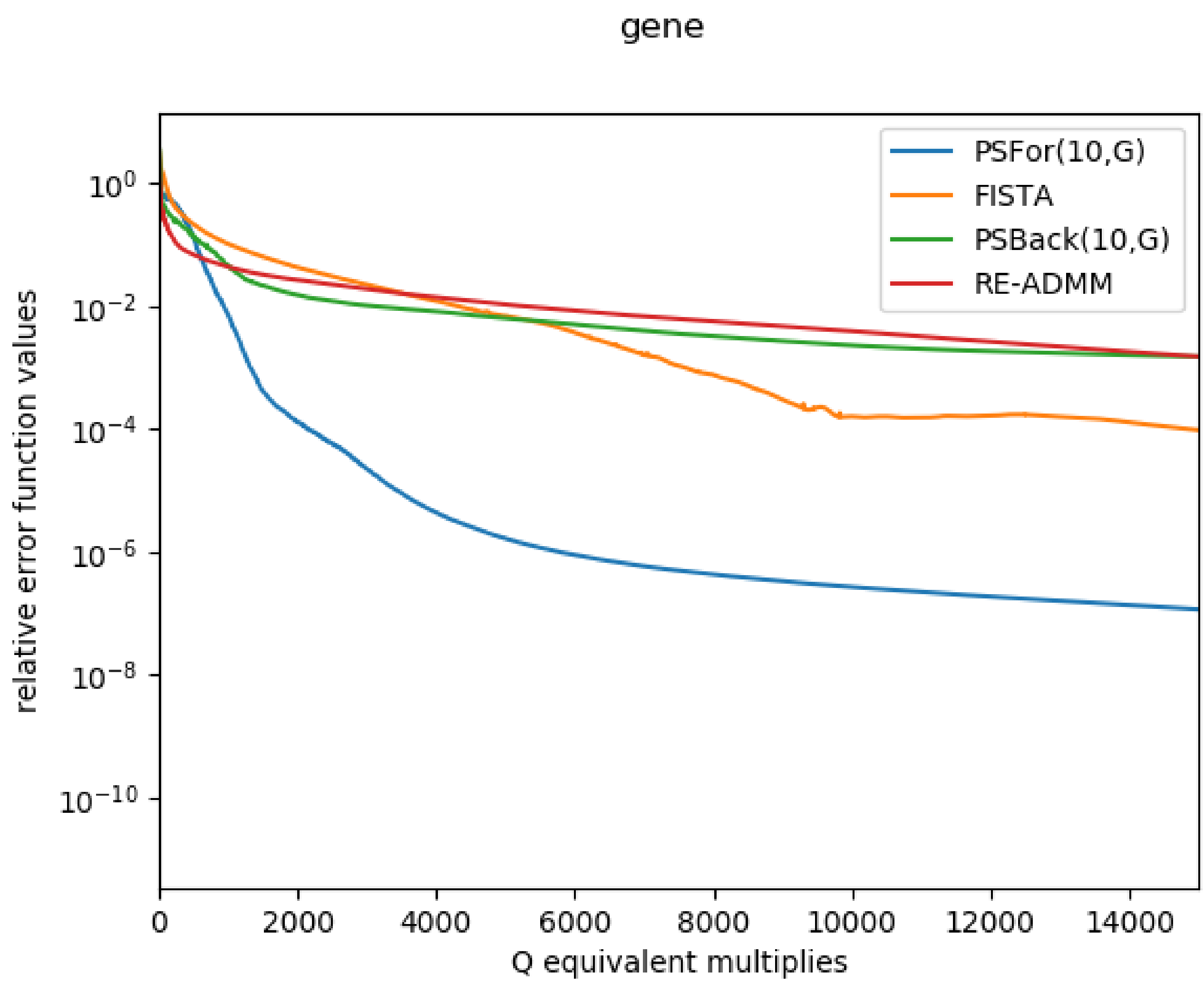
$$\min_{x\in\mathbb{R}^d}\left\{\sum_{i=1}^{r}\tfrac{1}{2}\left\|Q_i x-b_i\right\|^2+\lambda\left\|x\right\|_1\right\}$$

So we can set

$$T_i(x)=Q_i^{\mathsf{T}}(Q_i x-b_i),\ \forall i\in 1..n-1 \qquad T_n=\lambda\partial\left\|\cdot\right\|_1$$

- At each iteration, process blocks $\{i,n\}$, where $i\in 1..n-1$ is selected randomly or greedily
- Measure the number of "Q-equivalent" matrix multiplies

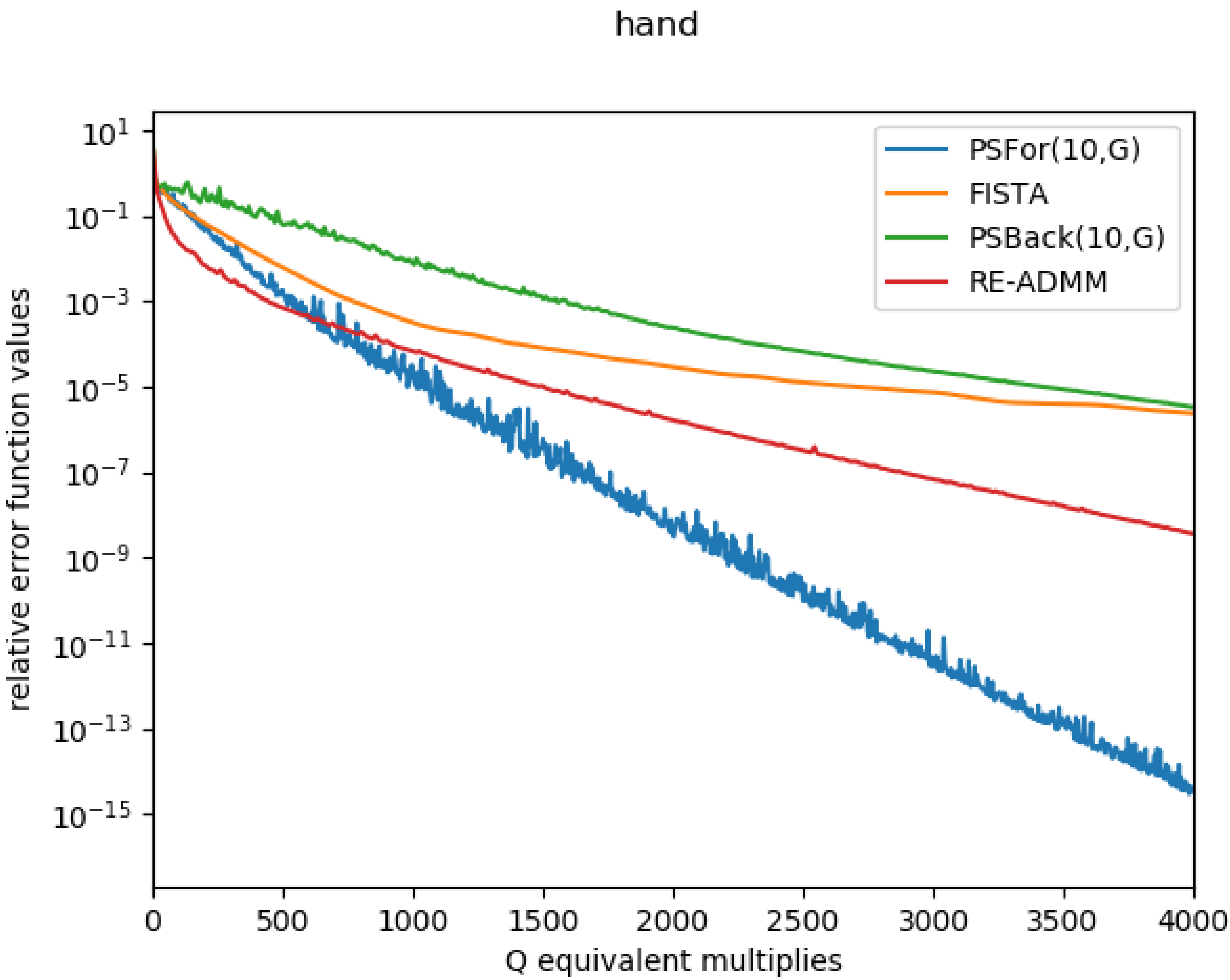
Augmented Cancer RNA Data: Dense, 3,204 × 20,531



526MB
of data

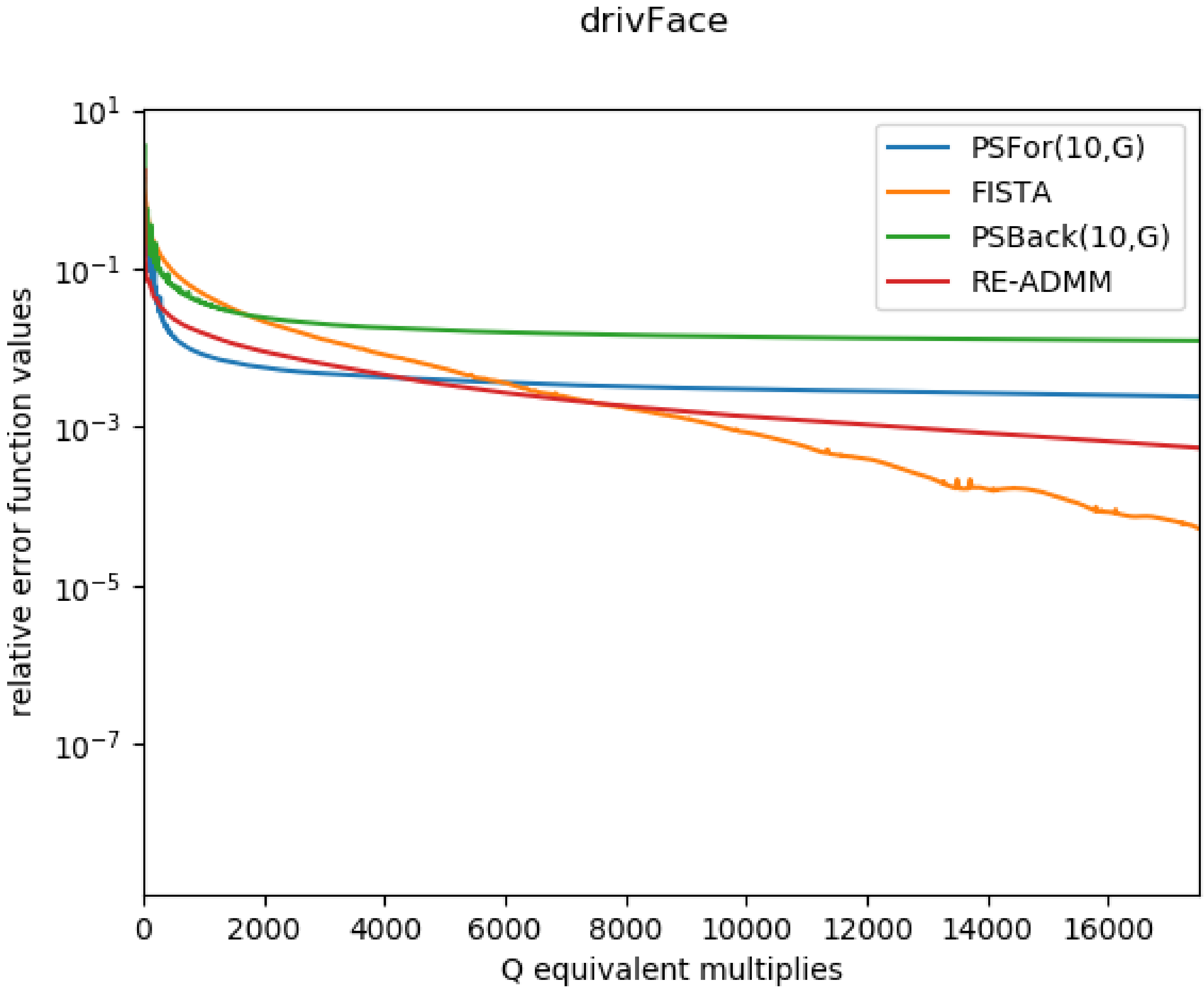
"PSFor" : forward steps for $i = 1, \ldots, r$
"PSBack" : proximal steps
"(10,G)" : $r = 10$, greedy selection

Hand Gesture Data: Dense, 1,500 × 3,000



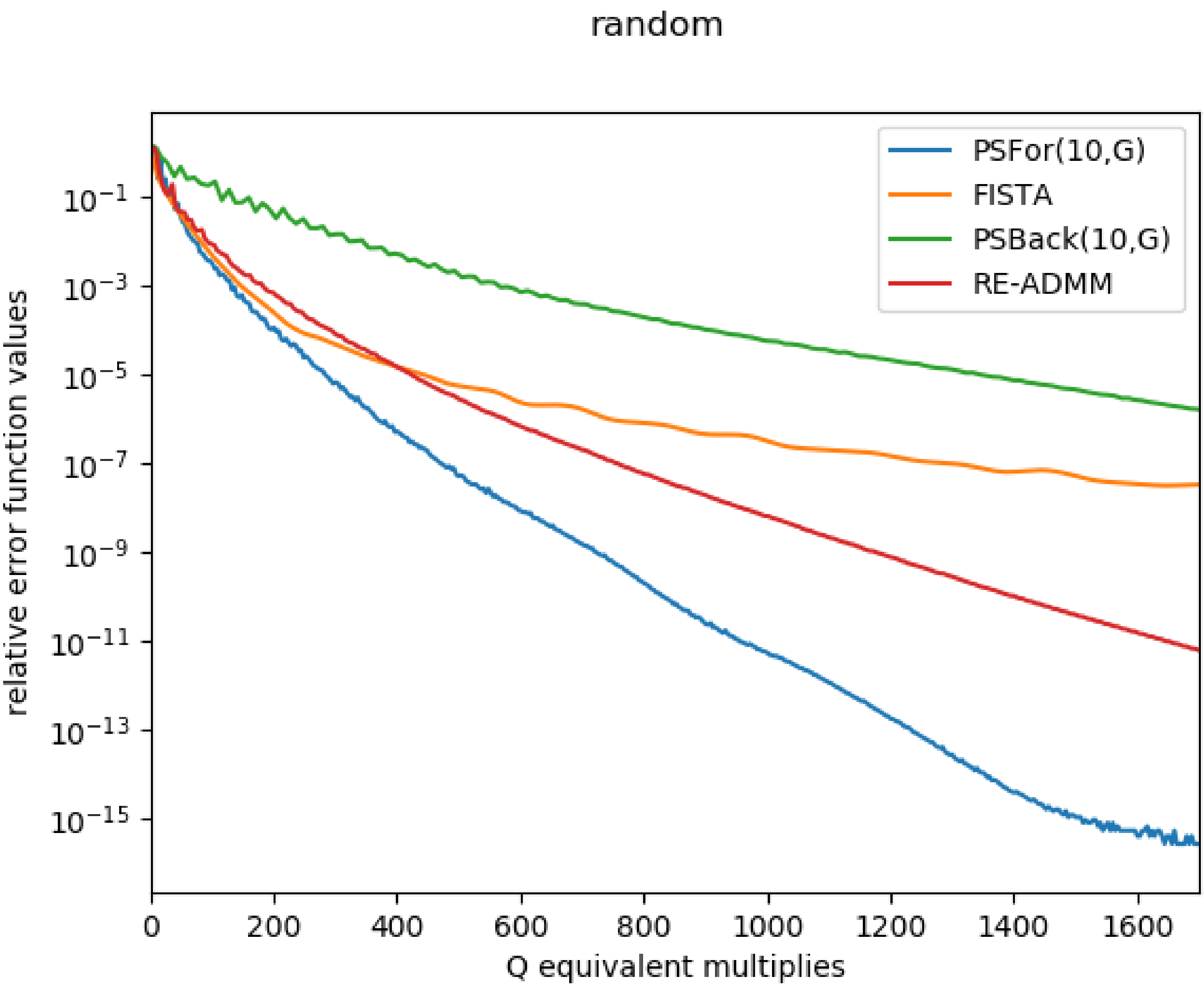
36MB of data

drivFace Data: Dense, 606 × 6,400



31MB
of data

Randomly Generated Data: Dense, 1,000 × 100,000



800MB
of data

A (not Very Realistic) Portfolio Selection Application

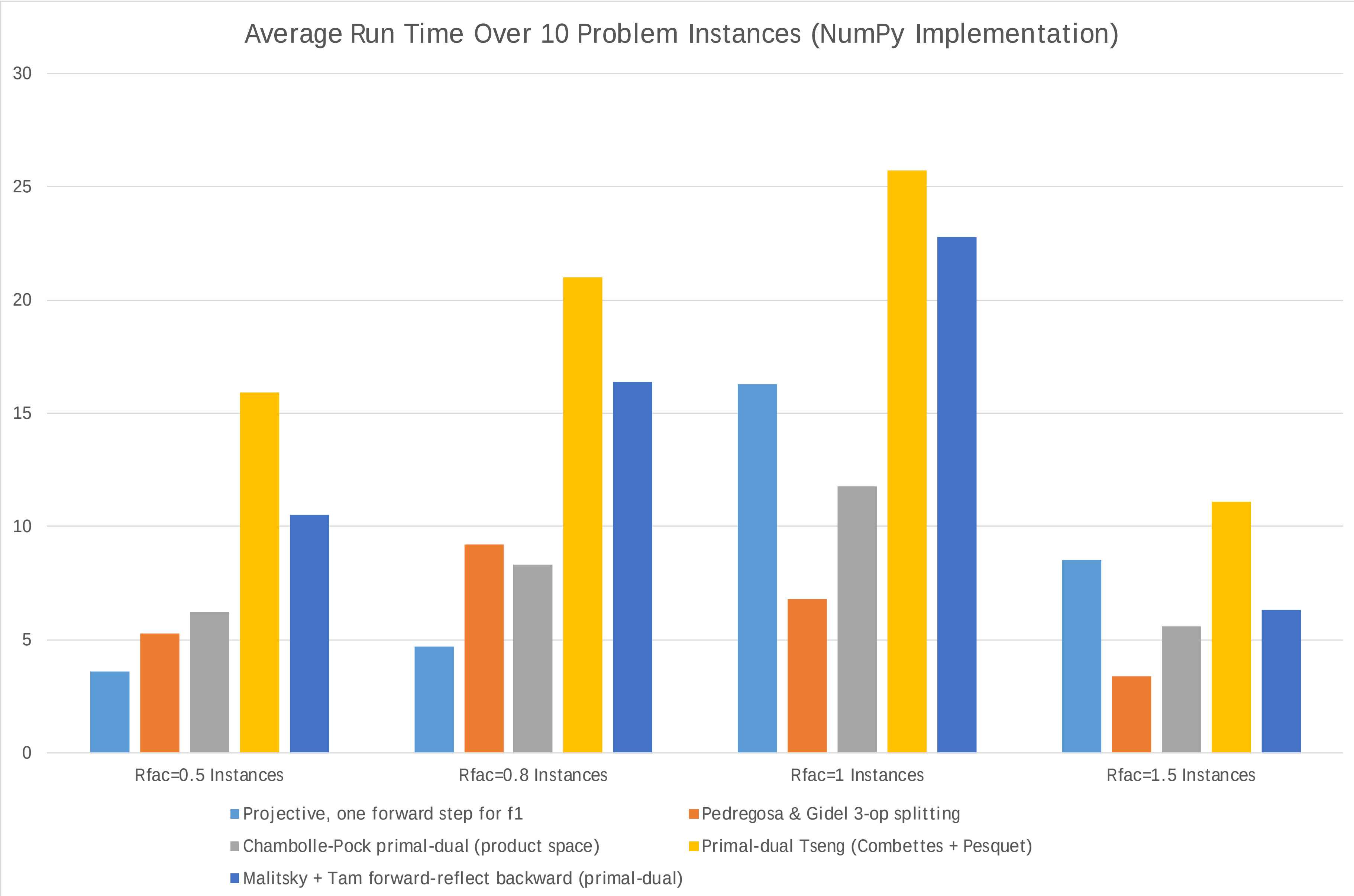
$$\begin{array}{ll} \min & \frac{1}{2} x^{\mathsf{T}} Q x \\ \mathrm{ST} & r^{\mathsf{T}} x \geq R \\ & \sum_{i=1}^{m} x_i = 1, \quad x \geq 0 \end{array}$$

- Q is a 10,000 × 10,000 dense positive semidefinite matrix
- Model as minimizing the sum of three functions $f_1 + f_2 + f_3$

$$f_1(x) = \frac{1}{2} x^{\mathsf{T}} Q x \quad f_2(x) = \begin{cases} 0, & r^{\mathsf{T}} x \geq R \\ +\infty, & r^{\mathsf{T}} x < R \end{cases} \quad f_2(x) = \begin{cases} 0, & \sum_{i=1}^{m} x_i = 1, \quad x \geq 0 \\ +\infty, & \text{otherwise} \end{cases}$$

- f_1 has a Lipschitz/cocoercive gradient
- f_2, f_3 have simple, linear-time prox operators
- The size and density of Q makes this problem hard for standard QP solvers

Run Time Results (Mixed)

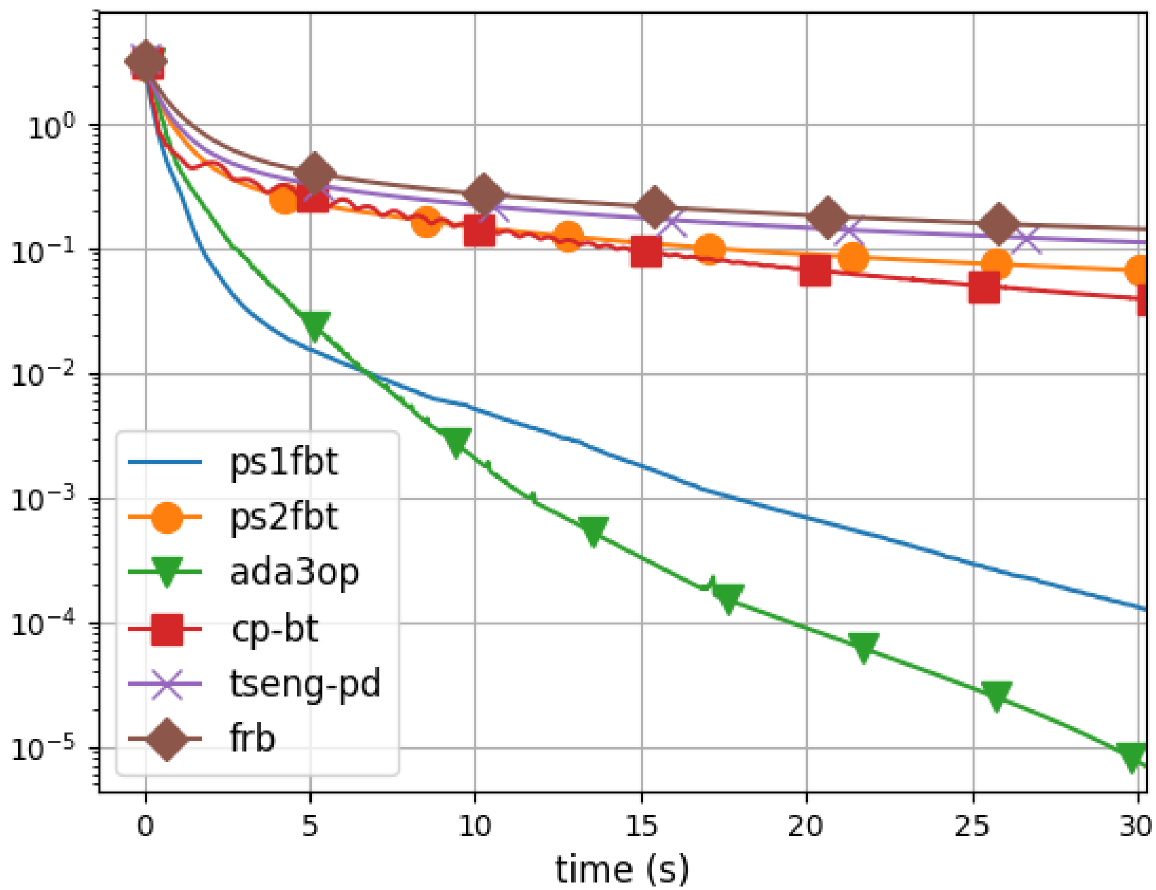


- R = (Rfac) × (average value of r_i)

Sparse Group-Regularized Logistic Regression, $\lambda_1 = \lambda_2 = 0.05$

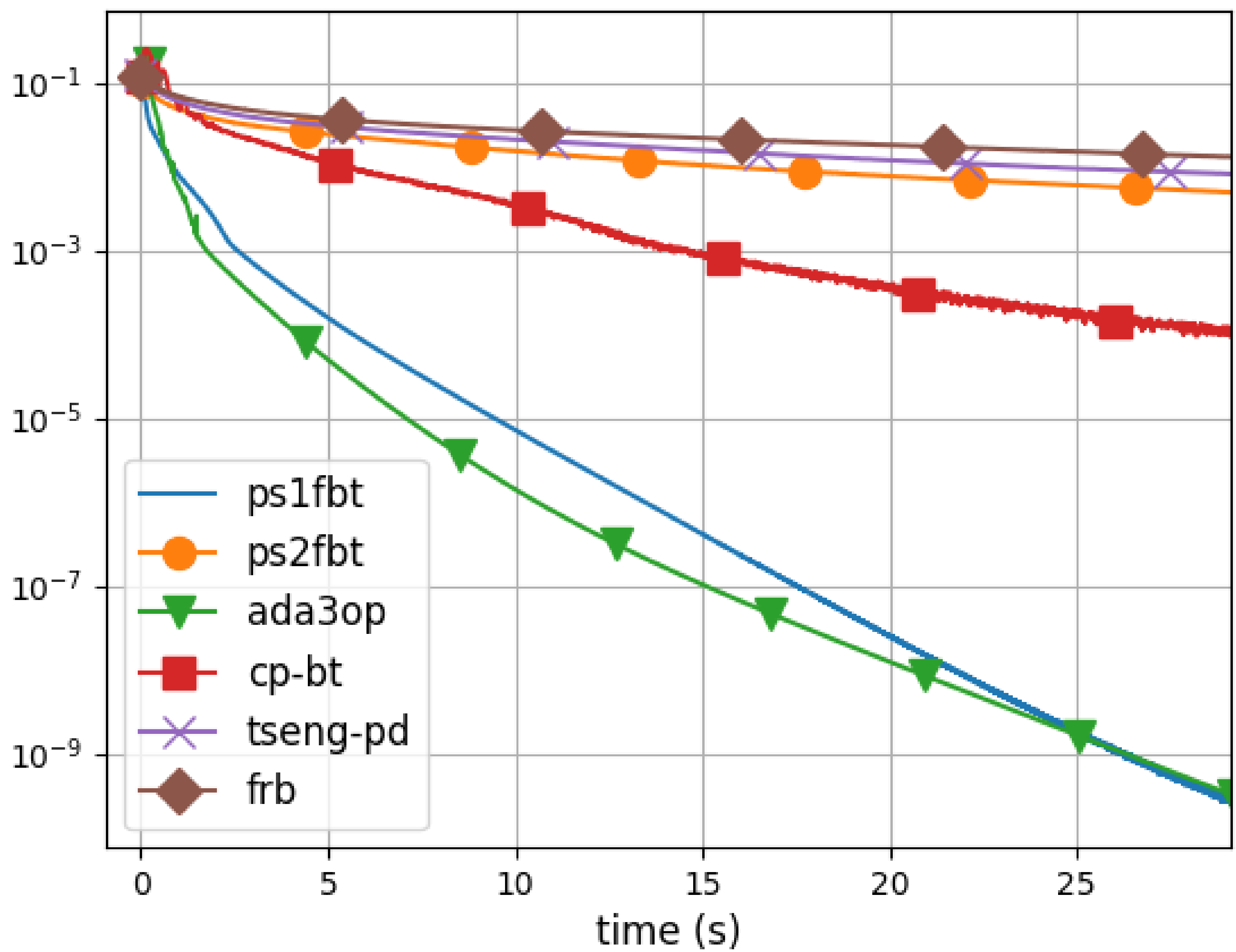
$$\min_{x_0 \in \mathbb{R}, x \in \mathbb{R}^d} \left\{ \sum_{i=1}^{n} \log\left(1 + \exp\left(-y_i (x_0 + a_i^\top x)\right)\right) + \lambda_1 \|x\|_1 + \lambda_2 \sum_{G \in \mathcal{G}} \|x_G\|_2 \right\}$$

where $\mathcal{G}$ is a disjoint collection of subsets of $\{1, \ldots, d\}$

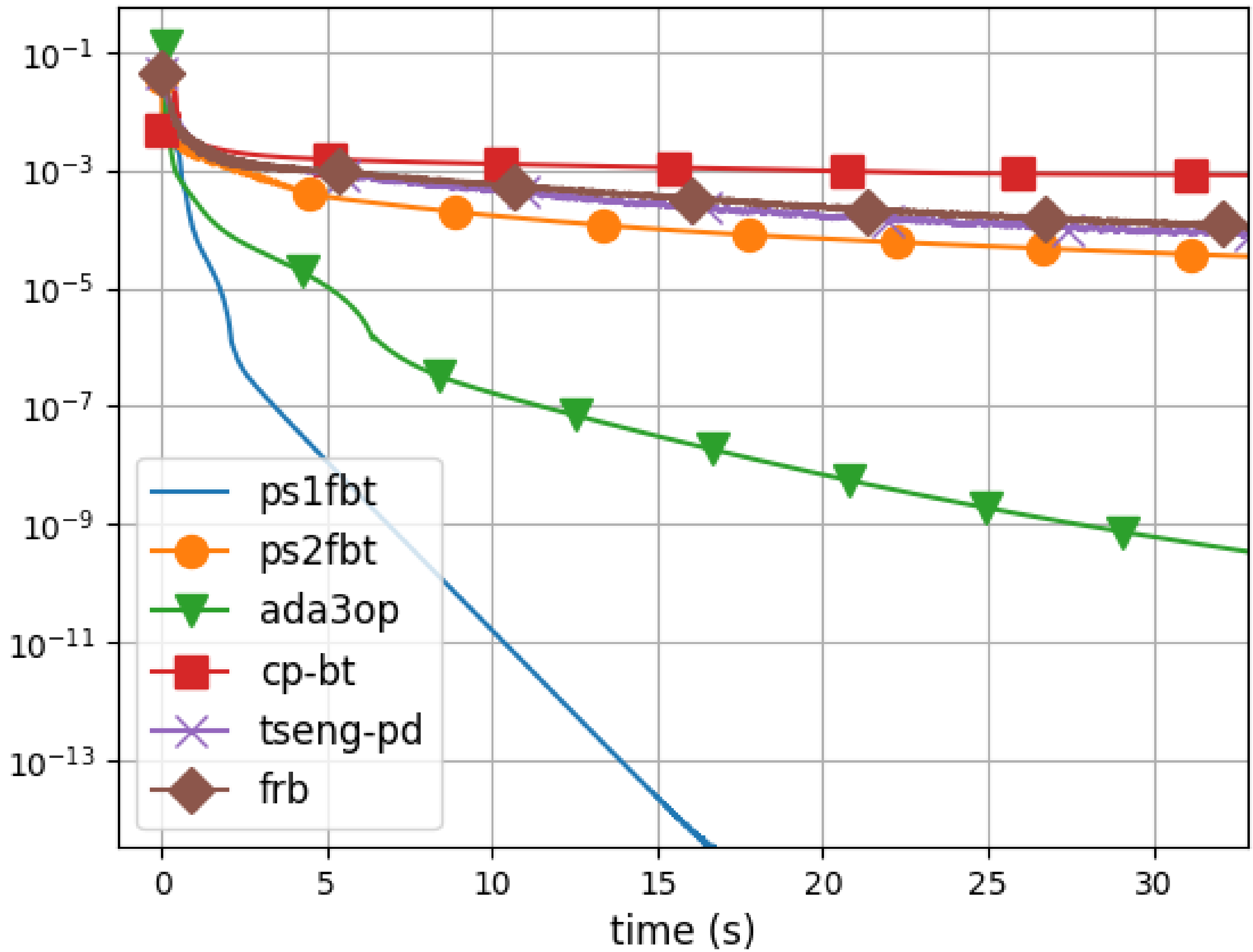


Breast cancer gene expression dataset (7705 genes × 60 patients)

Sparse Group-Regularized Logistic Regression, $\lambda_1 = \lambda_2 = 0.5$



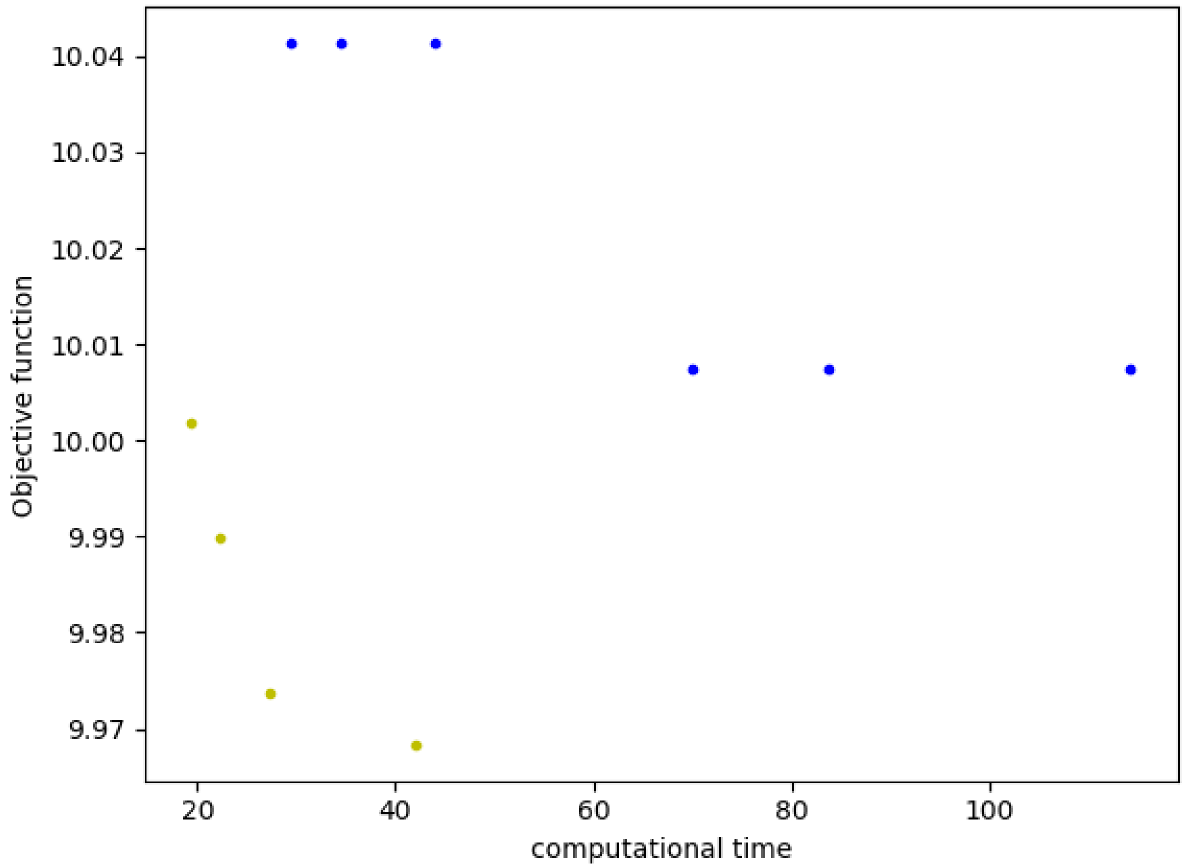
Sparse Group-Regularized Logistic Regression, $\lambda_1 = \lambda_2 = 0.85$



Another Application: Stochastic Programming

- Multi-stage linear programming problem over an unfolding tree of scenarios
- Application of projective splitting in a working paper by E, Watson and Woodruff
- None of the G_i are the identity
- Subproblems are quadratic programming problems for a single (multi-stage) scenario
- Results in a method resembling Rockafellar and Wets' progressive hedging (PH) method (blocks = scenarios)
- PH synchronous and processes every scenario at every iteration
- Our method is asynchronous and can process as few as one scenario per iteration
- Implemented within the Python-based PySP modeling/solution environment (Watson, Woodruff & Hart 2012)

Preliminary Results on a 32-Core Workstation (Woodruff)



$N = 10{,}000$ scenarios in $n = 20$ bundles, times in seconds
Blue points are PH on the same scenarios (and bundles)

- CPLEX cannot solve the extensive form of this problem in 3 days with 96 cores and 1TB RAM

Something to Keep in Mind

The projection operations, *e.g.*

$$\theta_k = \frac{\max\left\{\sum_{i=1}^{n}\left\langle G_i z - x_i^k, y_i^k - w_i\right\rangle, 0\right\}}{\left\|v^k\right\|^2 + \sum_{i=1}^{n}\left\|u_i^k\right\|^2}$$

$$z^{k+1} = z^k - \lambda_k \theta_k v^k$$

$$w_i^{k+1} = w_i^k - \lambda_k \theta_k u_i^k \quad i = 1,\ldots,n-1$$

- Require linear time (less in a parallel implementation)
- But do touch every primal and dual variable

- If processing an operator requires only a simple linear-time operation, one might as well do it every iteration
- Higher-complexity operations (matrix multiplication, quadratic programming) are different

Conclusions

- Projective splitting is a powerful framework for decomposing convex optimization problems
- Numerous variations are possible
- Does not care how many operators there are
- Accomplished "full splitting" when linear coupling matrices G_i are present
- Has applications in
  - Data analysis / statistics
  - Multistage stochastic programming
  - Vision and imaging ???????????????????????