This workshop brings together people from various backgrounds.

My postdoc Patrick Johnstone will cover some more recent contributions I’ve been involved in.

This talk is intended to give some historical perspective and give context for participants to understand aspects of the workshop in which they have less background.
Segment 0: Some History, From My Perspective

• When I was a doctoral student, my advisor Dimitri Bertsekas initially had me work on a “preflow-push” distributed algorithm for minimum-cost network flow

• The field was getting crowded, and nasty priority battles were flaring up

• Having implemented this particular algorithm, I had doubts about its utility
  o Making it efficient seemed to involve turning into the Hungarian algorithm, already well-known

• In early 1988, I told Dimitri I wanted to work on something different

• I gave him a write-up with some alternative ideas for min-cost flow, using quadratic cost perturbations

• I went on vacation for a week. When I came back...
“Your ideas are not very good”

- “I read your write-up. Your ideas are not very good.”
“Your ideas are not very good”

• “I read your write-up. Your ideas are not very good...”

• “… but they remind me of a topic I’ve been wondering about”

• “Read these papers by Rockafellar etc.”

• The topic:

What is the relationship between the standard augmented Lagrangian method, the proximal point algorithm, and the alternating direction method of multipliers (ADMM)?
The ADMM in 1988

- The augmented Lagrangian method (ALM) had been known since the 1970’s
- In 1976, Rockafellar showed that in the convex case it was an application of Martinet’s proximal method to the dual problem
- The ADMM was first described in 1975 in a paper (in French) by Glowinski and Marroco, as a kind of heuristic variant of the ALM
  - They were using a two-block alternating minimization algorithm to solve LM subproblems, and that found it worked best with just one pass over each block, followed by a multiplier adjustment
  - But there was no theory showing that this worked
  - It was still a little-known method in 1988, and continued to be so for 15-20 more years
Key Analyses Appearing in 1983

A 1983 edited volume by Fortin and Glowinski contained

- A convergence analysis of the ADMM by Fortin and Glowinski, using variational inequality methods
- An alternative analysis by Gabay, showing that it was a dual application of the Douglas-Rachford (DR) splitting method for set-valued monotone operators proposed by Lions and Mercier in 1979
  - The original DR method was for solving very particular kinds of linear equations
  - Lions-Mercier generalized it massively
  - Perhaps it should be called “LM”, not DR?
- These two analyses are the foundation of essentially all results about the ADMM on convex problems
- The two analyses have similarities but are not equivalent
Technical Segment 1 (Mostly for the PH People):
Subgradients of Convex Functions

• Consider a convex function \( f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\} \)

• \( f \) may not be smooth, but it has **subgradients**

• \( \partial f(x) \) denotes the set of subgradients of \( f \) at \( x \):

\[
\partial f(x) = \left\{ y \mid f(x') \geq f(x) + \langle y, x'-x \rangle \quad \forall x' \in \mathbb{R}^n \right\}
\]
Monotonicity

• Subgradient maps of convex functions are **monotone**

\[
y \in \partial f(x), \ y' \in \partial f(x') \implies \langle x - x', y - y' \rangle \geq 0
\]

\[
f(x') - f(x) \geq \langle y, x' - x \rangle
\]

\[
f(x) - f(x') \geq \langle y', x - x' \rangle
\]

\[
0 \geq \langle y' - y, x - x' \rangle
\]

• Any point-to-set map \( T : \mathbb{R}^n \to 2^{\mathbb{R}^n} \) with this property is called a **monotone operator** - not all such maps are subgradients
Why Is It Called Monotonicity?

\[ \text{gra } T = \{(x, y) \mid y \in T(x)\} \]

- Natural generalization to higher dimension of a function being monotone nondecreasing
For any scalar $c > 0$, each $t \in \mathbb{R}^n$ can be expressed in at most one way as $t = x + cy$, $y \in T(x)$

**Proof.** Rearrange $x + cy = x' + cy'$ into $x - x' = -c(y - y')$, from which monotonicity gives $0 \leq \langle x - x', y - y' \rangle = -c\|x - x'\|^2 \Rightarrow x = x'$
Minty’s Theorem

• We have just seen that any monotone operator $T$ gives a way of “decomposing” $\mathbb{R}^n$: any $t \in \mathbb{R}^n$ can be expressed at most one way as $t = x + cy$, $y \in T(x)$

• Minty’s theorem: if $T$ is maximal, that is, no points can be adjoined to the set $\text{gra } T = \{(x, y) \mid y \in T(x)\}$ without violating monotonicity, “at most one way” becomes “exactly one way” (looks simple, but annoyingly hard to prove)

• Also: when a convex function $f$ is lower semicontinuous (closed), the mapping $\partial f$ is maximal
  - Only very “weird” and “ugly” convex functions are not lower semicontinuous
Nonexpansive Maps from Monotone Operators

• Define $\text{refl}_{cT}$ to be the map $t \mapsto \text{refl}_{cT}(t)$ computed as follows:
  
  o Given $t \in \mathbb{R}^m$, express it uniquely as $t = x + cy$, with $y \in T(x)$
  
  o Return $x - cy$

  o Generalization of a reflection mapping

• This map is nonexpansive: $\|\text{refl}_{cT}(t) - \text{refl}_{cT}(t')\| \leq \|t - t'\| \quad \forall t, t' \in \mathbb{R}^n$
  
  o That is, Lipschitz continuous with modulus 1

• Defined everywhere if $T$ is maximal

• Also, the set of fixed point of $\text{refl}_{cT}$ is identical to the set of roots (or zeroes) of $T$, points $x : 0 \in T(x)$
  
  o In the case $T = \partial f$, these are the minimizers of $f$

• Proof follows...
Nonexpansive Maps from Monotone Operators — Proof

Proof. Take $t = x + cy$, $y \in T(x)$ and $t' = x' + cy'$, $y' \in T(x')$.

$$\|x + cy - (x' + cy')\|^2 = \|x - x'\|^2 + 2c \langle x - x', y - y' \rangle + c^2 \|y - y'\|^2$$

$$\|x - cy - (x' - cy')\|^2 = \|x - x'\|^2 - 2c \langle x - x', y - y' \rangle + c^2 \|y - y'\|^2$$

So we have

$$\|\text{refl}_{cT}(t) - \text{refl}_{cT}(t')\|^2 = \|x - cy - (x' - cy')\|^2$$

$$= \|x + cy - (x' + cy')\|^2 - 4c \langle x - x', y - y' \rangle$$

$$\leq \|x + cy - (x' + cy')\|^2 \quad \text{(monotonicity)}$$

$$= \|t - t'\|^2$$

Fixed point property:

$$t = x + cy = x - cy \iff y = 0, x = t \iff 0 \in T(t)$$
Connection to Algorithms

If we could evaluate \( \text{refl}_{cT} \), that might provide a way to solve the inclusion problem \( 0 \in T(x) \) (generalizes convex minimization)

- We have a nonexpansive map \( \text{refl}_{cT} \)
- Its fixed points are solutions to the dual problem
- Why not just iterate this map?
- But since the Lipschitz constant is 1, we could just “orbit”...
- So, “blend in” some of the identity map (Krasnosel'skii 1955):

\[
x^{k+1} = \left(1 - \frac{\rho_k}{2}\right)x^k + \left(\frac{\rho_k}{2}\right)\text{refl}_{cT}(x^k)
\]

Guaranteed to converge to a fixed point if \( 0 < \varepsilon \leq \rho_k \leq 2 - \varepsilon \ \forall k \)

- Example when \( \rho_k \equiv 1 \):

\[
x^{k+1} = \frac{1}{2}x^k + \frac{1}{2}\text{refl}_{cT}(x^k)
\]
The Proximal Point Algorithm (PPA)

• The map $\text{prox}_{cT} = \frac{1}{2} \text{Id} + \frac{1}{2} \text{refl}_{cT}$ works like this:
  
  o Given $t \in \mathbb{R}^m$, express it uniquely as $t = x + cy$, with $y \in T(x)$
  
  o Return just $x$

  o Generalization of a projection mapping

  o Also called the resolvent of $T$

• Rockafellar’s proximal point algorithm: iterate the map $\text{prox}_{cT}$, but you may vary $c$ with each iteration so long as it is bounded away from 0

  o Can be generalized to include inexact computation of resolvents and varying mixing factors (not just $\rho_k \equiv 1$)
Is that Implementable?

- Computing $\text{prox}_{cT}$ or equivalently $\text{refl}_{cT}$ involves computing the decomposition $t = x + cy, \ y \in T(x)$

- For $T = \partial f$ and a general function convex function $f$, this is as hard as minimizing $f$:

$$x = \arg \min_{x \in \mathbb{R}^m} \left\{ f(x) + \frac{1}{2c} \|x - t\|^2 \right\} \quad y = \frac{1}{c} (t - x)$$

This is the calculation in Martinet’s earlier algorithm

- Rockafellar 1976: when $f$ is obtained from the dual function of an optimization problem, evaluating $\text{prox}_{c\partial f}$ is not much harder than evaluating $f$

- And the ALM method is just the PPA applied to the subgradient mapping of the dual function
Operator Splitting

Simple example:

• Suppose that we want to solve $0 \in T_1(x) + T_2(x)$
  
  o That is, find $x, y: -y \in T_1(x), y \in T_2(x)$
  
  o Generalizes the problem $\min_x \{ f_1(x) + f_2(x) \}$

• Suppose also that $\text{prox}_{cT_1}, \text{prox}_{cT_2}$ (or equivalent reflections) are easy to compute, but $\text{prox}_{c(T_1+T_2)}$ is not

• Can we converge to a minimizer of the sum only through evaluations of the individual $\text{prox}_{cT_1}, \text{prox}_{cT_2}$, or equivalently $\text{refl}_{cT_1}, \text{refl}_{cT_2}$?
Operator Splitting - Douglas-Rachford

Yes, based on the following simple observations:

The composition of two nonexpansive mappings is nonexpansive

- Nonexpansive map \( \text{refl}_{cT_1} : \mathbb{R}^n \to \mathbb{R}^n \)
- Nonexpansive map \( \text{refl}_{cT_2} : \mathbb{R}^n \to \mathbb{R}^n \)
- Their composition \( \text{refl}_{cT_1} \circ \text{refl}_{cT_2} \) is nonexpansive

Furthermore, the fixed points of \( \text{refl}_{cT_1} \circ \text{refl}_{cT_2} \) are of the form

\[
\left\{ x + cy \mid y \in T_2(x), -y \in T_1(x) \right\}
\]

Sketch of proof. \( x + cy \mapsto x - cy \mapsto x + cy \). Easy to show this is the only possibility (two equations in two unknowns). \( \square \)
Operator Splitting - Douglas-Rachford II

- We have a nonexpansive map \( \text{refl}_{cT_1} \circ \text{refl}_{cT_2} \)

- From a fixed point \( t \) of this map, we can just write the unique decomposition \( t = x + cy \), where \( y \in T_2(x) \), and \( x \) solves our problem

- Much as we did before, find a fixed point by iterating the nonexpansive map “blended” with the identity: for \( \inf_k \rho_k > 0, \sup_k \rho_k < 2 \),

\[
S^{k+1} = \left(1 - \frac{\rho_k}{2}\right)S^k + \left(\frac{\rho_k}{2}\right)(\text{refl}_{cT_1} \circ \text{refl}_{cT_2})(S^k)
\]

\[
= \left(1 - \frac{\rho_k}{2}\right)S^k + \left(\frac{\rho_k}{2}\right)\text{refl}_{cT_1} \left(\text{refl}_{cT_2}(S^k)\right)
\]

- The case \( \rho_k \equiv 1 \) is Douglas-Rachford splitting

- The boundary case \( \rho_k \equiv 2 \) is Peaceman-Rachford splitting (converges under additional assumptions)
Other Forms of Two-Operator Splitting

- Forward-backward (generalizes gradient projection)
  \[ x^{k+1} = \text{prox}_{c_k T_2} \left( x^k - c_k T_1(x) \right) \]
  - Requires \( T_1 \) to be cocoercive (hence single-valued), along with restrictions on the stepsize \( c_k \)

- Tseng’s forward-backward-forward
  - Similar, but only requires \( T_1 \) to be Lipschitz (and thus single-valued)

- Projective splitting (also for more than 2 operators)
  - Started with E & Svaiter (2008), more from Patrick later
The ADMM - Problem Setting

• $f : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{+\infty\}$ is a convex function
• $g : \mathbb{R}^m \rightarrow \mathbb{R} \cup \{+\infty\}$ is a convex function
• $M$ is an $m \times n$ matrix

\[
\min f(x) + g(Mx)
\]

• Equivalent formulation:

\[
\begin{align*}
\min & \quad f(x) + g(z) \\
\text{ST} & \quad Mx = z
\end{align*}
\]

Standard augmented Lagrangian method for this formulation:

\[
\begin{align*}
(x^{k+1}, z^{k+1}) & \in \text{Arg min}_{x \in \mathbb{R}^n, z \in \mathbb{R}^m} \left\{ f(x) + g(z) + \langle \lambda^k, Mx - z \rangle + \frac{\rho}{2} \|Mx - z\|^2 \right\} \\
\lambda^{k+1} & = \lambda^k + \rho_k c(Mx^{k+1} - z^{k+1})
\end{align*}
\]
The Dual Problem

The dual function of the problem is:

\[ q(\lambda) = \min_{x \in \mathbb{R}^n, z \in \mathbb{R}^m} \{L(x, z, \lambda)\} \]

\[ = \min_{x \in \mathbb{R}^n, z \in \mathbb{R}^m} \{f(x) + g(z) + \langle \lambda, Mx - z \rangle\} \]

\[ = \min_{x \in \mathbb{R}^n} \left\{ f(x) + \langle \lambda, Mx \rangle \right\} + \min_{z \in \mathbb{R}^m} \left\{ g(z) - \langle \lambda, z \rangle \right\} \]

\[ = q_1(\lambda) + q_2(\lambda) \]

The dual problem is

\[ \max_{\lambda \in \mathbb{R}^m} \{q_1(\lambda) + q_2(\lambda)\} \]

...or equivalently...

\[ \min_{\lambda \in \mathbb{R}^m} \{d_1(\lambda) + d_2(\lambda)\}, \text{ where } d_1 = -q_1 \text{ and } d_2 = -q_2 \]

The functions \(d_1\) and \(d_2\) are convex and lower semicontinuous
The ADMM

Gabay’s insight (1983): applying DR splitting to this dual problem is equivalent to the algorithm

\[
    x^{k+1} \in \text{Arg min}_{x \in \mathbb{R}^n} \left\{ f(x) + \left\langle \lambda^k, Mx \right\rangle + \frac{\epsilon}{2} \| Mx - z^k \|^2 \right\}
\]

\[
    z^{k+1} \in \text{Arg min}_{z \in \mathbb{R}^m} \left\{ g(z) - \left\langle \lambda^k, z \right\rangle + \frac{\epsilon}{2} \| Mx^{k+1} - z \|^2 \right\}
\]

\[
    \lambda^{k+1} = \lambda^k + c(Mx^{k+1} - z^{k+1})
\]

This is the alternating direction method of multipliers (ADMM)

- Minimize augmented Lagrangian in \(x\) with \(z\) fixed at last value
- Fix this \(x\) and minimize over \(z\)
- Perform the usual multiplier update

...exactly the algorithm that Glowinski and Marrocco found heuristically in 1975
The Fortin-Glowinski Proof for the ADMM

- Fortin and Glowinski presented things slightly differently, but their proof can be viewed in the following manner:

- For the same problem, consider the Lagrangian function, which is convex in \((x, z)\), concave in \(\lambda\):

\[
L(x, z, \lambda) = f(x) + g(z) + \langle \lambda, Mx - z \rangle
= f(x) + \langle \lambda, Mx \rangle + g(z) - \langle \lambda, z \rangle
= L_1(x, \lambda) + L_2(z, \lambda)
\]

- Both \(L_1, L_2\), are convex in \((x, z)\), and concave in \(\lambda\)
- Although \(L_1\) ignores \(z\) and \(L_2\) ignores \(x\)
The FG Proof: Subgradients of Saddle Functions

For convex-concave function $H(x, z, \lambda)$, a subgradient at $(x, z, \lambda)$ is $(u, v, r)$ such that [written $(u, v, r) \in \partial H(x, z, \lambda)$]

$$H(x', z', \lambda) \geq H(x, z, \lambda) + \langle u, x' - x \rangle + \langle v, z' - z \rangle \quad \forall x', z'$$

$$H(x, z, \lambda') \leq H(x, z, \lambda) - \langle r, \lambda' - \lambda \rangle \quad \forall \lambda'$$

For a convex-concave function, subgradients are also monotone:

$$(u, v, r) \in \partial H(x, z, \lambda), (u', v', r') \in \partial H(x', z', \lambda')$$

$$\Rightarrow \quad \langle x - x', u - u' \rangle + \langle z - z', v - v' \rangle + \langle \lambda - \lambda', r - r' \rangle \geq 0$$

Key: Fejér monotonicity — use monotonicity of $\partial L_1, \partial L_2$ to show that whenever $(x^*, z^*)$ is an optimal solution and $\lambda^*$ is an optimal dual solution,

$$\left\| (\lambda^{k+1}, cz^{k+1}) - (\lambda^*, cz^*) \right\|^2 \leq \left\| (\lambda^k, cz^k) - (\lambda^*, cz^*) \right\|^2 - \cdots$$
Subtle Differences

• The Gabay/Lions-Mercier DR analysis for the ADMM is equivalent to Fejér monotonicity to the fixed points of $\text{refl}_{cT_1} \circ \text{refl}_{cT_2}$, equivalent to

$$\left\| (\lambda^{k+1} + cz^{k+1}) - (\lambda^* + cz^*) \right\|^2 \leq \left\| (\lambda^k + z^k) - (\lambda^* + cz^*) \right\|^2 - \cdots$$

• This is slightly different

• When you include overrelaxation ($\rho_k \neq 1$), you get two different classes of algorithms, with similar convergence properties

• The algorithms coincide exactly when $\rho_k \equiv 1$, but not otherwise
Technical Segment 2 (for non-PH People)
Progressive Hedging for ADMM and Operator Splitting People -
First, a Generic Problem Form

- Let \( n = pq \) and write \( x \in \mathbb{R}^n \) as \((x_1, \ldots, x_p)\), with \( x_1, \ldots, x_p \in \mathbb{R}^q\)
- \( f_i : \mathbb{R}^q \to \mathbb{R} \cup \{\infty\} \) is closed convex for \( i = 1, \ldots, p \)
- \( M_i \) is an \( q \times q' \) matrix for \( i = 1, \ldots, p \)
- Let \( m = pq' \) and \( V \) denote a linear subspace of \( \mathbb{R}^m = \mathbb{R}^{pq'} \)

\[
\begin{array}{c}
\min \sum_{i=1}^{p} f_i(x_i) \\
\text{ST} \quad (M_1 x_1, \ldots, M_p x_p) \in V
\end{array}
\]

- Note: can easily be made more general
Applying the ADMM

\[ f(x) = \sum_{i=1}^{p} f_i(x_i) \quad \text{and} \quad g(z) = \begin{cases} 0, & z \in V \\ +\infty, & z \notin V \end{cases} \]

Then the problem is just \( \min f(x) + g(Mx) \)

Applying the ADMM, we obtain (after some algebra)

\[
\begin{align*}
    x_i^{k+1} &= \arg \min_{x_i} \left\{ f_i(x_i) + \langle M_i x_i, \lambda_i^k \rangle + \frac{c}{2} \| M_i x_i - z_i^k \|^2 \right\} & i = 1, \ldots, p \\
    z^{k+1} &= \text{proj}_V \left( M_1 x_1^{k+1}, \ldots, M_p x_p^{k+1} \right) \\
    \lambda_i^{k+1} &= \lambda_i^k + c (M_i x_i^{k+1} - z_i^{k+1}) & i = 1, \ldots, p
\end{align*}
\]

• Note that we always have \( \lambda^k = (\lambda_1^k, \ldots, \lambda_p^k) \in V^\perp \)

• Applying DR to subspace indicator functions like \( g \) is equivalent to Spingarn’s method of partial inverses
Applying this Form to Stochastic Programming

- Consider a standard stochastic programming scenario tree:

\[ \pi_i \] is the probability of last-stage scenario \( i \)

- \( \pi_i \) is the probability of last-stage scenario \( i \)
Nonanticipativity Formulation

- Replicate decision variables: \( p \) copies at every stage
- \( x_{is} \) is the vector of decision variables for scenario \( i \) at stage \( s \),
  \[ x_i = (x_{i1}, \ldots, x_{iT}), \quad x = (x_1, \ldots, x_p) = ((x_{11}, \ldots, x_{1T}), \ldots, (x_{p1}, \ldots, x_{pT})) \]
- \( z = (z_1, \ldots, z_p) = ((z_{11}, \ldots, z_{1,S-1}), \ldots, (z_{p1}, \ldots, z_{p,S-1})) \) (no last stage)
- Let \( V \) denote the subspace meeting the nonanticipativity constraints that \( z_{is} = z_{js} \) whenever scenarios \( i \) and \( j \) are indistinguishable at stage \( s \)

\[ E \]

\[ F \]
Further, let

- \( f_i(x_i) = \pi_i h_i(x_i) \), where \( h_i(\cdot) \) is the cost function for scenario \( i \) and \( +\infty \) if any constraint within scenario \( i \) is violated

- \( M_i \) just drops the last-stage variables from scenario \( i \)

\( h_i \) encapsulates all the costs and constraints across all stages in the (hypothetical) situation that you know the final outcome will be scenario \( i \)

Then our stochastic program is exactly

\[
\min \sum_{i=1}^{p} f_i(x_i) \\
\text{ST} \quad (M_1 x_1, \ldots, M_p x_p) \in V
\]

Apply our ADMM-derived method this problem, and get...
Progressive Hedging (PH)

\[
x_{i}^{k+1} = \arg\min_{x_{i} \in X_{i}} \left\{ \pi_{i} h_{i}(x_{i}) + \frac{c}{2} \sum_{s=1}^{T-1} \left( \langle x_{is}, \lambda_{is}^{k} \rangle + \|x_{is} - z_{is}^{k}\|^{2} \right) \right\} \quad i = 1, \ldots, p
\]

\[
z_{is}^{k+1} = \frac{1}{|D(i,s)|} \sum_{j \in D(i,s)} x_{js}^{k+1} \quad i = 1, \ldots, p \quad s = 1, \ldots, S - 1
\]

\[
\lambda_{is}^{k+1} = \lambda_{is}^{k} + c(x_{is}^{k+1} - z_{is}^{k+1}) \quad i = 1, \ldots, p \quad s = 1, \ldots, S - 1
\]

- Here, \( D(i,s) \) denotes the set of scenarios indistinguishable from scenario \( i \) at stage \( s \)
- Will always have \( \sum_{j \in D(i,s)} \lambda_{j}^{k} = 0 \) for all \( i \) and \( s \) (from \( \lambda^{k} \in V^\perp \))
Progressive Hedging (Original Form)

- The above is not exactly the original form proposed by Rockafellar and Wets (1991)
- That form is obtained by reweighting the inner product by the scenario probabilities $\pi_i$

$$x_{i}^{k+1} = \left\{ h_i(x_i) + \frac{c}{2} \sum_{s=1}^{T-1} \left( \langle x_{is}^{k}, \lambda_{is}^{k} \rangle + \| x_{is} - z_{is}^{k}\|^2 \right) \right\} \quad i = 1, \ldots, p$$

$$z_{is}^{k+1} = \frac{1}{\left( \sum_{j \in D(i,s)} \pi_j \right)} \sum_{j \in D(i,s)} \pi_j x_{js}^{k+1} \quad i = 1, \ldots, p \quad s = 1, \ldots, S - 1$$

$$\lambda_{is}^{k+1} = \lambda_{is}^{k} + c(x_{is}^{k+1} - z_{is}^{k+1}) \quad i = 1, \ldots, p \quad s = 1, \ldots, S - 1$$

- Probabilities no longer appear in the subproblem objectives
- But averages are now weighted by probabilities
- Multipliers now obey $\sum_{j \in D(i,s)} \pi_j \lambda_j^k = 0$ for all $i$ and $s$
Segment 3: More History, Open Questions

- Almost everything I’ve discussed was known by 1990
- My own numerical experiments in the late 80’s didn’t show very encouraging results for classic OR optimization problems like min-cost network flow, general LP, etc.
- Progressive hedging did not “catch on” widely for a while

Hibernation for about 20 years, then

- The ADMM makes a comeback: it seems to work fairly well for machine-learning-related problems like LASSO, and becomes widely known
- The PySP system (Watson, Woodruff, Hart 2018) provides and accessible version of PH coupled with a flexible modeling environment
Discrete Variables

For stochastic programming with discrete variables (in power systems planning, for example) it is easy to add integrality constraints to the subproblems (Watson & Woodruff 2011), but that makes them nonconvex

- Usually more tractable than trying to solve the extensive form of the problem with a standard MIP solver (it’s just too big)
- But can this approach be more than a heuristic?
- Yes, in some situations one can get Lagrangian bounds after some suitable modifications - see Jim Luedtke’s talk this afternoon
- These ideas can apply in other splitting/ADMM contexts besides PH
Nonconvexity

• What about nonconvex problems?
  o For example, suppose the overall problem is nonconvex but the individual subproblems are respectively convex in $x$ and $z$, or are somehow solvable in closed form
  o Can the ADMM be more than a heuristic in such cases?

• Example problem: “dictionary learning” - given a data matrix $D$, solve

$$\min_{X,Z} \left\{ \frac{1}{2} \|D - XZ\|^2 + \nu_1 \|X\|_1 + \nu_2 \|Z\|_1 \right\},$$

Where $X, Z$ are smaller matrices of some appropriate dimension
In this workshop, we will start considering these questions...

...along with many other enhancements - both convex and nonconvex - to this family of methods and our understanding of them

Onward!