Control Tools for Distributed Optimization

ADMM via operator theory and distributed optimization

Prof. Ivano Notarnicola

Dept. of Electrical, Electronic, Information Eng.
Università di Bologna
ivano.notarnicola@unibo.it

Prof. Ruggero Carli

Dept. of Information Engineering
Università di Padova
ruggero.carli@unipd.it

SIDRA Ph.D. Summer School
July, 10-12 2025 • Bertinoro, Italy

Friday Afternoon Trajectory

Distributed Optimization

$$\min_x \; f(x) = \sum_{i=1}^{N} f_i(x)$$

$$z(k+1) = (1-\alpha)z(k) \; + \; \alpha \, T_{PRS} \, z(k)$$

Operator Theory

$$\begin{array}{ll} \underset{x,y}{\text{minimize}} & f(x) + g(y) \\ \text{subject that} & Ax + By = c \end{array}$$

ADMM – Relaxed ADMM

ADMM

Consider the optimization problem

$$\begin{aligned} \min_{x,y} \quad & f(x) + g(y) \\ \text{s.t.} \quad & Ax + By = c \end{aligned}$$

where $x \in \mathbb{R}^n$, $y \in \mathbb{R}^m$, $A \in \mathbb{R}^{p\times n}$, $B \in \mathbb{R}^{p\times m}$, $c \in \mathbb{R}^p$.

ADMM

Consider the optimization problem

$$\begin{aligned} \min_{x,y} \quad & f(x) + g(y) \\ \text{s.t.} \quad & Ax + By = c \end{aligned}$$

where $x \in \mathbb{R}^n$, $y \in \mathbb{R}^m$, $A \in \mathbb{R}^{p \times n}$, $B \in \mathbb{R}^{p \times m}$, $c \in \mathbb{R}^p$.

Augmented Lagrangian (λ vector of Lagrange multipliers, $\rho > 0$)

$$\mathcal{L}_\rho(x, y, \lambda) = f(x) + g(y) + \lambda^\top (Ax + By - c) + \frac{\rho}{2}\|Ax + By - c\|^2$$

ADMM

Consider the optimization problem

$$\begin{aligned} \min_{x,y} \quad & f(x) + g(y) \\ \text{s.t.} \quad & Ax + By = c \end{aligned}$$

where $x \in \mathbb{R}^n$, $y \in \mathbb{R}^m$, $A \in \mathbb{R}^{p\times n}$, $B \in \mathbb{R}^{p\times m}$, $c \in \mathbb{R}^p$.

Augmented Lagrangian (λ vector of Lagrange multipliers, $\rho > 0$)

$$\mathcal{L}_\rho(x, y, \lambda) = f(x) + g(y) + \lambda^\top (Ax + By - c) + \frac{\rho}{2}\|Ax + By - c\|^2$$

ADMM - Alternating direction multipliers method

$$\begin{aligned} x_{k+1} &= \operatorname*{argmin}_x \mathcal{L}_\rho(x, y_k, \lambda_k) \\ y_{k+1} &= \operatorname*{argmin}_y \mathcal{L}_\rho(x_{k+1}, y, \lambda_k) \\ \lambda_{k+1} &= \lambda_k + \rho\,(Ax_{k+1} + By_{k+1} - c) \end{aligned}$$

ADMM - Relaxed ADMM

Consider the optimization problem

$$\min_{x,y} \quad f(x) + g(y)$$
$$\text{s.t.} \quad Ax + By = c$$

where $x \in \mathbb{R}^n$, $y \in \mathbb{R}^m$, $A \in \mathbb{R}^{p\times n}$, $B \in \mathbb{R}^{p\times m}$, $c \in \mathbb{R}^p$.

Augmented Lagrangian (λ vector of Lagrange multipliers, $\rho > 0$)

$$\mathcal{L}_\rho(x, y, \lambda) = f(x) + g(y) + \lambda^\top (Ax + By - c) + \frac{\rho}{2}\|Ax + By - c\|^2$$

Relaxed ADMM - classical ADMM for $\alpha = 1/2$)

$$y_{k+1} = \operatorname*{argmin}_y \left\{\mathcal{L}_\rho(x_k, y, \lambda_k) + \rho(2\alpha - 1)(By)^\top (Ax_k) + By_k - c\right\}$$
$$\lambda_{k+1} = \lambda_k + \rho\left(Ax_k + By_{k+1} - c\right) - \rho(2\alpha - 1)(Ax_k + By_k - c)$$
$$x_{k+1} = \operatorname*{argmin}_x \mathcal{L}_\rho(x, y_{k+1}, \lambda_{k+1})$$

ADMM - Relaxed ADMM

Consider the optimization problem

$$\begin{aligned} \min_{x,y} \quad & f(x) + g(y) \\ \text{s.t.} \quad & Ax + By = c \end{aligned}$$

where $x \in \mathbb{R}^n$, $y \in \mathbb{R}^m$, $A \in \mathbb{R}^{p\times n}$, $B \in \mathbb{R}^{p\times m}$, $c \in \mathbb{R}^p$.

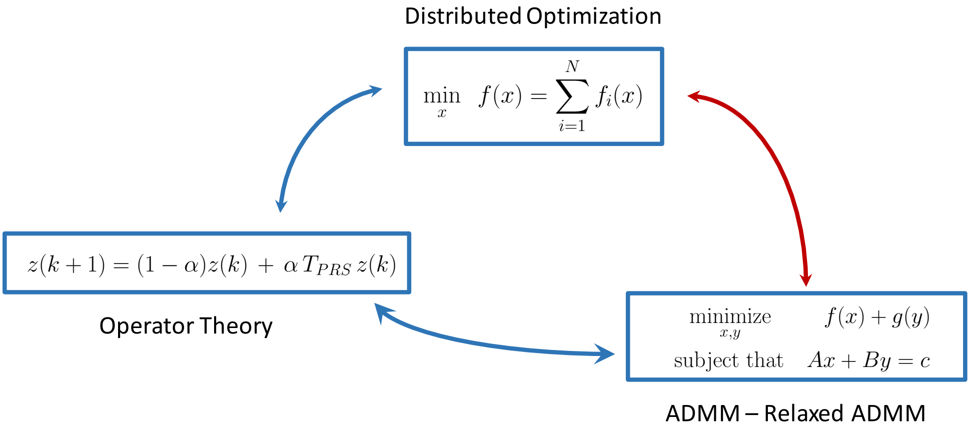
Relaxed ADMM - classical ADMM for $\alpha = 1/2$)

$$\begin{aligned} y_{k+1} &= \operatorname*{argmin}_y \left\{ \mathcal{L}_\rho(x_k, y, \lambda_k) + \rho(2\alpha - 1)(By)^\top (Ax_k) + By_k - c \right\} \\ \lambda_{k+1} &= \lambda_k + \rho\,(Ax_k + By_{k+1} - c) - \rho(2\alpha - 1)(Ax_k + By_k - c) \\ x_{k+1} &= \operatorname*{argmin}_x \mathcal{L}_\rho(x, y_{k+1}, \lambda_{k+1}) \end{aligned}$$

Convergence Under some mild assumptions on f and g, convergence to optimal solution is guaranteed if

$$0 < \alpha < 1 \qquad \text{and} \qquad \rho > 0$$

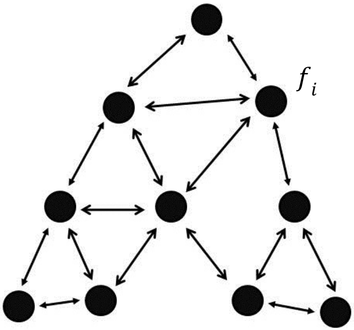
Friday Afternoon Trajectory



Optimization over networks

- $\mathcal{G} = (V, \mathcal{E})$ undirected
- $f_i : \mathbb{R} \to \mathbb{R}$, local function known only by node i

$$\min_x \sum_{i=1}^{N} f_i(x) \qquad (*)$$



Question : What is the relation between problem in $(*)$ and problem $(**)$?

$$\begin{aligned} \min_{x,y} \quad & f(x) + g(y) \qquad (**) \\ \text{s.t.} \quad & Ax + By = c \end{aligned}$$

From optimization over networks to consensus optimization

- $\mathcal{G} = (V, \mathcal{E})$ undirected
- $f_i : \mathbb{R} \to \mathbb{R}$, local function known only by node i

$$\min_{x} \sum_{i=1}^{N} f_i(x)$$

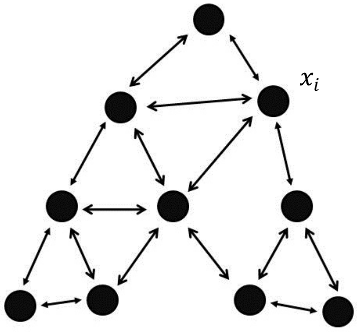
- x_i : *local copy* of x stored in memory by node i

$$\min_{x_1,\ldots,x_N} \sum_{i=1}^{N} f_i(x_i)$$

$$\text{s.t. } x_1 = \ldots = x_N$$

consensus constraint

- The two problems are equivalent



From optimization over networks to consensus optimization

- $\mathcal{G} = (V, \mathcal{E})$ undirected
- $f_i : \mathbb{R} \to \mathbb{R}$, local function known only by node i

$$\min_x \sum_{i=1}^{N} f_i(x)$$

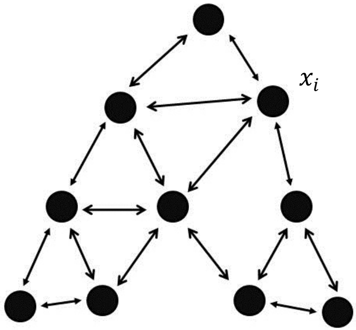
- x_i : *local copy* of x stored in memory by node i

$$\min_{x_1,\ldots,x_N} \sum_{i=1}^{N} f_i(x_i)$$

$$\text{s.t. } x_i = x_j \qquad \forall\,(i,j) \in \mathcal{E}$$

consensus constraint

- The two problems are equivalent if the graph $\mathcal{G}$ is *connected*



From optimization over networks to relaxed-ADMM

- x_i : *local copy* of x stored in memory by node i

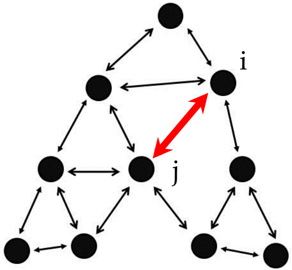
$$\min_{x_1,\dots,x_N} \sum_{i=1}^{N} f_i(x_i)$$

$$\text{s.t. } x_i = x_j \qquad \forall\ (i,j) \in \mathcal{E}$$

consensus constraint

- bridge variables : y_{ij}, y_{ji}

$$x_i = x_j \qquad \Leftrightarrow \qquad \begin{aligned} x_i &= y_{ij} \\ x_i &= y_{ji} \\ y_{ij} &= y_{ji} \end{aligned} \qquad \forall (i,j) \in \mathcal{E}$$

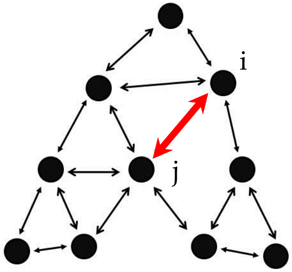


From optimization over networks to relaxed-ADMM

...we can consider the following problem...

$$\min_{x_1,\dots,x_N} \sum_{i=1}^{N} f_i(x_i)$$

$$\text{s.t.} \quad \begin{aligned} x_i &= y_{ij} \\ x_i &= y_{ji} \\ y_{ij} &= y_{ji} \end{aligned} \qquad \forall (i,j) \in \mathcal{E}$$



that can be rewritten as

$$\min_{x} \; f(x) = \sum_{i=1}^{N} f_i(x)$$

$$\text{s.t.} \quad \begin{aligned} & Ax + y = 0 \\ & y = Py \end{aligned}$$

$$x = [x_1, \dots, x_N]^\top \in \mathbb{R}^N \qquad y = [y_{ij}, y_{ji}] \in \mathbb{R}^{2|\mathcal{E}|}$$

Optimization over networks - Relaxed ADMM

Last step : From

$$\min_{x} \; f(x) = \sum_{i=1}^{N} f_i(x)$$

$$\text{s.t.} \quad \begin{array}{l} Ax + y = 0 \\ y = Py \end{array}$$

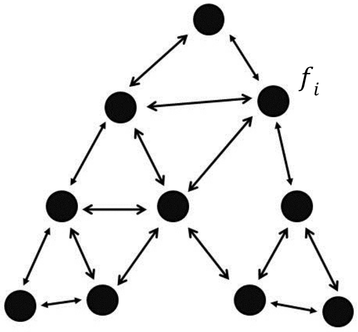
to

$$\min_{x,y} \; f(x) + \iota_{(I-P)}(y)$$

$$\text{s.t.} \quad Ax + y = 0$$

where

$$\iota_{I-P}(y) = \begin{cases} 0 & \text{if } \; y = Py \\ +\infty & \text{otherwise} \end{cases}$$



Optimization over networks - Relaxed ADMM

From

$$\min_x \sum_{i=1}^{N} f_i(x)$$

to

$$\min_{x,y} \; f(x) + \iota_{(I-P)}(y)$$
$$\text{s.t.} \; Ax + y = 0$$

Relaxed ADMM - classical ADMM for $\alpha = 1/2$)

$$y_{k+1} = \operatorname*{argmin}_y \left\{ \mathcal{L}_\rho(x_k, y, \lambda_k) + \rho(2\alpha - 1)(By)^\top (Ax_k) + By_k - c \right\}$$
$$\lambda_{k+1} = \lambda_k + \rho\,(Ax_k + By_{k+1} - c) - \rho(2\alpha - 1)(Ax_k + By_k - c)$$
$$x_{k+1} = \operatorname*{argmin}_x \mathcal{L}_\rho(x, y_{k+1}, \lambda_{k+1})$$

$\lambda \in \mathbb{R}^{2|\mathcal{E}|}$, $y \in \mathbb{R}^{2|\mathcal{E}|}$

Optimization over networks - Relaxed ADMM

Problem

$$\min_{x,y} \quad f(x) + \iota_{(I-P)}(y)$$
$$\text{s.t.} \quad Ax + y = 0$$

Relaxed ADMM - classical ADMM for $\alpha = 1/2$)

$$y_{k+1} = \operatorname*{argmin}_y \left\{ \mathcal{L}_\rho(x_k, y, \lambda_k) + \rho(2\alpha - 1)(By)^\top (Ax_k) + By_k - c \right\}$$
$$\lambda_{k+1} = \lambda_k + \rho\,(Ax_k + By_{k+1} - c) - \rho(2\alpha - 1)(Ax_k + By_k - c)$$
$$x_{k+1} = \operatorname*{argmin}_x \mathcal{L}_\rho(x, y_{k+1}, \lambda_{k+1})$$

Relaxed ADMM

$$y_{ij,k+1} = \frac{1}{2\rho}\left[(\lambda_{ij,k} + \lambda_{ji,k}) + 2\alpha\rho(x_{i,k} + x_{j,k}) - \rho(2\alpha - 1)(y_{ij,k} + y_{ji,k})\right]$$
$$\lambda_{ij,k+1} = \frac{1}{2}\left[(\lambda_{ij,k} - \lambda_{ji,k}) + 2\alpha\rho(x_{i,k} - x_{j,k}) - \rho(2\alpha - 1)(y_{ij,k} - y_{ji,k})\right]$$
$$x_{i,k+1} = \operatorname*{argmin}_{x_i} \left\{ f_i(x_i) + \frac{\rho}{2}|\mathcal{N}_i|\|x_i\|^2 + x_i{}^\top \left(\sum_{j \in \mathcal{N}_i} \rho(2\alpha - 1)y_{ji,k} - 2\alpha\rho x_{j,k} - \lambda_{ij,k} \right) \right\}$$

Distributed Relaxed-ADMM

Relaxed ADMM

$$y_{ij,k+1} = \frac{1}{2\rho}\left[(\lambda_{ij,k} + \lambda_{ji,k}) + 2\alpha\rho(x_{i,k} + x_{j,k}) - \rho(2\alpha - 1)(y_{ij,k} + y_{ji,k})\right]$$

$$\lambda_{ij,k+1} = \frac{1}{2}\left[(\lambda_{ij,k} - \lambda_{ji,k}) + 2\alpha\rho(x_{i,k} - x_{j,k}) - \rho(2\alpha - 1)(y_{ij,k} - y_{ji,k})\right]$$

$$x_{i,k+1} = \underset{x_i}{\operatorname{argmin}} \left\{ f_i(x_i) + \frac{\rho}{2}|\mathcal{N}_i| \|x_i\|^2 + {x_i}^\top \left(\sum_{j \in \mathcal{N}_i} \rho(2\alpha - 1) y_{ji,k} - 2\alpha\rho x_{j,k} - \lambda_{ij,k} \right) \right\}$$

Question: amenable of distributed implementation? Yes! Node i stores in memory $x_i, \{y_{ij}, \lambda_{ij}\}_{j \in \mathcal{N}_i}$

Relaxed-ADMM

$$y_{ij,k+1} = h_y\left(x_{i,k}, y_{ij,k}, \lambda_{ij,k}, \{x_{j,k}, y_{ji,k}, \lambda_{ji,k}\}_{j \in \mathcal{N}_i}\right)$$

$$\lambda_{ij,k+1} = h_\lambda\left(x_{i,k}, y_{ij,k}, \lambda_{ij,k}, \{x_{j,k}, y_{ji,k}, \lambda_{ji,k}\}_{j \in \mathcal{N}_i}\right)$$

$$x_{i,k+1} = h_x\left(x_{i,k}, y_{ij,k}, \lambda_{ij,k}, \{x_{j,k}, y_{ji,k}, \lambda_{ji,k}\}_{j \in \mathcal{N}_i}\right)$$

Distributed Relaxed-ADMM

Observation: it is possible to exploit the redundancy introduced in the constraints to simplify the algorithm?

$$x_i = x_j \qquad \Leftrightarrow \qquad \begin{aligned} x_i &= y_{ij} \\ x_i &= y_{ji} \\ y_{ij} &= y_{ji} \end{aligned} \qquad \forall (i,j) \in \mathcal{E}$$

By properly defining z_{ij} as function of $(y_{ij}, \lambda_{ij}, x_i)$ one can simplify the algorithm to the following two updates

$$x_{i,k+1} = \underset{x_i}{\operatorname{argmin}} \left\{ f_i(x_i) - \left(\sum_{j \in \mathcal{N}_i} z_{ij,k} \right)^\top x_i + \frac{\rho d_i}{2} \|x_i\|^2 \right\}$$

$$z_{ij,k+1} = (1-\alpha) z_{ij,k} - \alpha z_{ji,k} + 2\alpha\rho x_{j,k+1}$$

Distributed Relaxed-ADMM

Distributed algorithm?

$$x_{i,k+1} = \operatorname*{argmin}_{x_i} \left\{ f_i(x_i) - \left(\sum_{j \in \mathcal{N}_i} z_{ij,k} \right)^\top x_i + \frac{\rho d_i}{2} \|x_i\|^2 \right\}$$
$$z_{ij,k+1} = (1-\alpha) z_{ij,k} - \alpha z_{ji,k} + 2\alpha\rho x_{j,k+1}$$

- Node i keeps in memory x_i and $\{z_{ij}\}_{j \in \mathcal{N}_i}$
- Define $q_{j\to i} = -z_{ji,k} + 2\rho x_{j,k+1}$ (quantity sent from node j to node i)
- Then

$$z_{ij,k+1} = (1-\alpha) z_{ij,k} - \alpha z_{ji,k} + 2\alpha\rho x_{j,k+1}$$

can be rewritten as

$$z_{ij,k+1} = (1-\alpha) z_{ij,k} + \alpha q_{j\to i}$$

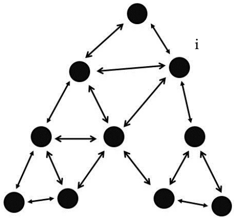
Distributed Relaxed-ADMM

Algorithm

Node i keeps in memory x_i and $\{z_{ij}\}_{j\in\mathcal{N}_i}$

1. Node i computes $x_{i,k+1}$ as

$$x_{i,k+1} = \underset{x_i}{\operatorname{argmin}} \left\{ f_i(x_i) - \left(\sum_{j\in\mathcal{N}_i} z_{ij,k} \right)^\top x_i + \frac{\rho d_i}{2} \|x_i\|^2 \right\}$$



Distributed Relaxed-ADMM

Algorithm

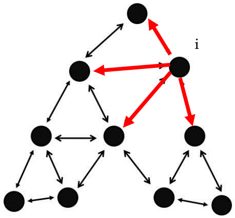
Node i keeps in memory x_i and $\{z_{ij}\}_{j\in\mathcal{N}_i}$

1. Node i computes $x_{i,k+1}$ as

$$x_{i,k+1} = \operatorname*{argmin}_{x_i} \left\{ f_i(x_i) - \left(\sum_{j\in\mathcal{N}_i} z_{ij,k} \right)^\top x_i + \frac{\rho d_i}{2} \|x_i\|^2 \right\}$$

2. Node i computes and transmits the temporary variable $q_{i\to j}$ for all $j \in \mathcal{N}_i$

$$q_{i\to j} = -z_{ij,k} + 2\rho x_{i,k+1}$$



Distributed Relaxed-ADMM

Algorithm

Node i keeps in memory x_i and $\{z_{ij}\}_{j \in \mathcal{N}_i}$

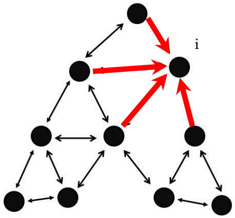
1. Node i computes $x_{i,k+1}$ as

$$x_{i,k+1} = \operatorname*{argmin}_{x_i} \left\{ f_i(x_i) - \left(\sum_{j \in \mathcal{N}_i} z_{ij,k} \right)^\top x_i + \frac{\rho d_i}{2} \|x_i\|^2 \right\}$$

2. Node i computes and transmits the temporary variable $q_{i \to j}$ for all $j \in \mathcal{N}_i$

$$q_{i \to j} = -z_{ij,k} + 2\rho x_{i,k+1}$$

3. Node i gathers $q_{j \to i}$ from all $j \in \mathcal{N}_i$;



Distributed Relaxed-ADMM

Algorithm

Node i keeps in memory x_i and $\{z_{ij}\}_{j\in\mathcal{N}_i}$

1. Node i computes $x_{i,k+1}$ as

$$x_{i,k+1} = \operatorname*{argmin}_{x_i} \left\{ f_i(x_i) - \left(\sum_{j\in\mathcal{N}_i} z_{ij,k} \right)^\top x_i + \frac{\rho d_i}{2} \|x_i\|^2 \right\}$$

2. Node i computes and transmits the temporary variable $q_{i\to j}$ for all $j \in \mathcal{N}_i$

$$q_{i\to j} = -z_{ij,k} + 2\rho x_{i,k+1}$$

3. Node i gathers $q_{j\to i}$ from all $j \in \mathcal{N}_i$;
4. Node i computes $z_{ij,k+1}$ as

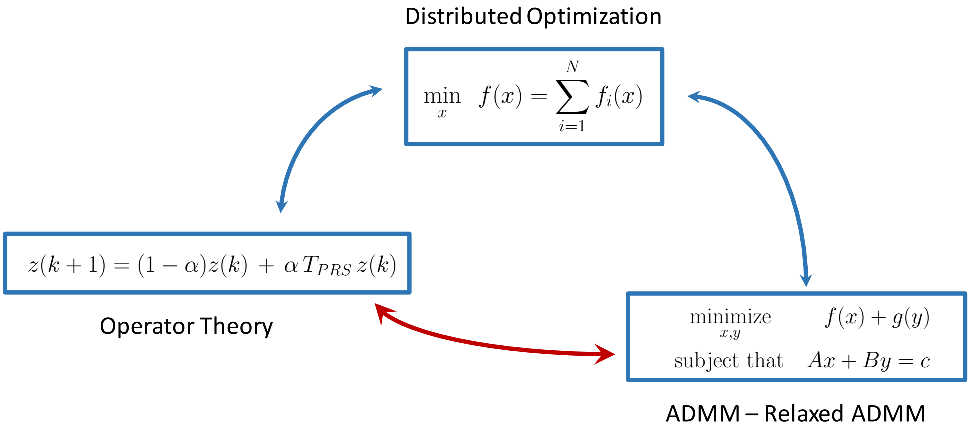
$$z_{ij,k+1} = (1-\alpha) z_{ij,k} + \alpha q_{j\to i}$$

Distributed Relaxed-ADMM

Proposition (*conditions on α and ρ for convergence*)
If $0 < \alpha < 1$ and $\rho > 0$ then Distributed Relaxed ADMM converges to the optimal solution.

Proposition (*conditions for exponential convergence*)
If f_i are strongly convex then convergence is exponentially fast.

Friday Afternoon Trajectory



ADMM

Definition (*Nonexpansive operator*)

An operator $T : \mathbb{R}^n \to \mathbb{R}^n$ is *nonexpansive* if for all x_A, x_B it holds

$$\|T(x_A) - T(x_B)\| \le \|x_A - x_B\|$$

Definition (*Contractive operator*)

An operator $T : \mathbb{R}^n \to \mathbb{R}^n$ is *contractive* if there exists $0 < \gamma < 1$ such that, for all x_A, x_B, it holds

$$\|T(x_A) - T(x_B)\| \le \gamma \|x_A - x_B\|$$

ADMM

Definition (*Fixed points*)

We say that $\bar{x}$ is a *fixed point* for $T : \mathbb{R}^n \to \mathbb{R}^n$ if

$$T(\bar{x}) = \bar{x}.$$

We denote by $\text{fix}(T)$ the set of fixed points of T, i.e.,

$$\text{fix}(T) = \{\bar{x} \,|\, T(\bar{x}) = \bar{x}\}$$

Finding fixed points

Proposition (*Banach-Picard iteration*) Let $T : \mathbb{R}^n \to \mathbb{R}^n$ be contractive. Then the iteration

$$x_{k+1} = T(x_k)$$

converges to the fixed point of T.

Proposition (*Krasnosel'skii-Mann iteration*) Let $T : \mathbb{R}^n \to \mathbb{R}^n$ be nonexpansive, with fix$(T) \neq \emptyset$. Take $0 < \alpha < 1$, then the iteration

$$x_{k+1} = (1 - \alpha)x_k + \alpha T(x_k)$$

converges to a fixed point of T.

Proximal operators

Definition (*Proximal operator*) Let $f : \mathbb{R}^n \to \mathbb{R}$ be closed, proper and convex and let $\rho > 0$ be a parameter. We define the *proximal operator*

$$\text{prox}_{\rho f}(y) = \underset{x \in \mathbb{R}^n}{\operatorname{argmin}} \left\{ f(x) + \frac{1}{2\rho} \|x - y\|^2 \right\}$$

converges to a fixed point of T.

Reflective operator : $\text{refl}_{\rho f} = 2\text{prox}_{\rho f} - I$

Fact. The proximal and reflective operator are nonexpansive.

Assumption. From now on we will assume all the functions to be closed, proper, convex and that all the optimization problems we consider have a unique minimizer.

Proximal operators

Consider

$$\min_{x} f(x)$$

If $x_\star$ is the minimizer of f then

$$\text{prox}_{\rho f}(x_\star) = \operatorname*{argmin}_{x} \left\{ f(x) + \frac{1}{2\rho}\|x - x_\star\|^2 \right\} = x_\star$$

Hence, $x_\star$ is a fixed point for the proximal operator

Proximal operators

Consider

$$\min_x f(x)$$

If $x_\star$ is the minimizer of f then

$$\mathsf{prox}_{\rho f}(x_\star) = \operatorname*{argmin}_x \left\{ f(x) + \frac{1}{2\rho}\|x - x_\star\|^2 \right\} = x_\star$$

Hence, $x_\star$ is a fixed point for the proximal operator

Krasnosel'skii-Mann iteration with $\mathsf{prox}_{\rho f}$. By applying

$$x_{k+1} = (1-\alpha)x(k) + \alpha\, \mathsf{prox}_{\rho f}(x_k)$$

we have that $x_k \to x_\star$.

Splitting operators

Consider the problem

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

To find minimizer x_* we could apply

$$x_{k+1} = (1 - \alpha) x_k + \alpha \, \mathsf{prox}_{\rho(f+g)}(x_k)$$

Splitting operators

Consider the problem

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

To find minimizer x_* we could apply

$$x_{k+1} = (1-\alpha)x_k + \alpha\, \mathrm{prox}_{\rho(f+g)}(x_k)$$

Observations

- Computing $\mathrm{prox}_{\rho(f+g)}$ could be difficult
- Computing $\mathrm{prox}_{\rho f}$ and $\mathrm{prox}_{\rho g}$ in a separate way could be easier

Splitting operators

Consider the problem

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

To find minimizer x_* we could apply

$$x_{k+1} = (1-\alpha)x_k + \alpha\,\mathsf{prox}_{\rho(f+g)}(x_k)$$

Observations

- Computing $\mathsf{prox}_{\rho(f+g)}$ could be difficult
- Computing $\mathsf{prox}_{\rho f}$ and $\mathsf{prox}_{\rho g}$ in a separate way could be easier

Possible alternatives proposed in the literature

- Proximal gradient mapping
- Peaceman Rachford splitting operator

Splitting operators : proximal gradient algorithm

Consider the problem

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

where

- f differentialble is L-smooth and μ strongly convex;
- g convex with non-expensive prox operator

Proximal Gradient Algorithm $x_{k+1} = \text{prox}_{\rho g}\left(x_k - \rho \nabla f(x_k)\right)$

Splitting operators : proximal gradient algorithm

Consider the problem

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

where

- f differentialble is L-smooth and μ strongly convex;
- g convex with non-expensive prox operator

Proximal Gradient Algorithm $x_{k+1} = \text{prox}_{\rho g}(x_k - \rho \nabla f(x_k))$

Convergence If $0 < \rho < \frac{1}{L}$, then

$$\|x_{k+1} - x_\star\|^2 \leq (1 - \mu\rho)\|x_k - x_\star\|^2$$

Splitting operators : Peaceman-Rachford

Consider the problem

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

Peaceman-Rachford Splitting (PRS) : $T_{PRS} = \mathsf{refl}_{\rho f} \circ \mathsf{refl}_{\rho g}$

Consider

$$z_{k+1} = (1-\alpha) z_k + \alpha \, T_{PRS}(z_k)$$

Then,

$$z_k \to z_* \qquad \Rightarrow \qquad x_* = \mathsf{prox}_{\rho g}(z_*)$$

Splitting operators : Peaceman-Rachford

Consider the problem

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

Peaceman-Rachford Splitting (PRS) : $T_{PRS} = \text{refl}_{\rho f} \circ \text{refl}_{\rho g}$

Consider

$$z_{k+1} = (1-\alpha) z_k + \alpha\, T_{PRS}(z_k)$$

Then,

$$z_k \to z_* \qquad \Rightarrow \qquad x_* = \text{prox}_{\rho g}(z_*)$$

Computationally: An efficient way to compute $z_{k+1} = (1-\alpha) z_k + \alpha\, T_{PRS}(z_k)$ is

$$\begin{aligned} x_k &= \text{prox}_{\rho g}(z_k) \\ \xi_k &= \text{prox}_{\rho f}(2x_k - z_k) \\ z_{k+1} &= z_k + 2\alpha(\xi_k - x_k) \end{aligned}$$

Duality

Consider

$$\begin{aligned} \min_x \quad & f_0(x) \\ \text{s.t.} \quad & f_i(x) \leq 0 \quad i = 1, \ldots, m \\ & h_i(x) = 0 \quad i = 1, \ldots, p \end{aligned}$$

where $x \in \mathbb{R}^n$.

Let

- $\mathcal{D} = \bigcap_{i=0}^{m} \text{dom} f_i \;\; \cap \;\; \bigcap_{i=1}^{p} \text{dom} h_i$
- p^* be the optimal solution

Duality

Consider

$$\begin{aligned} \min_x \quad & f_0(x) \\ \text{s.t.} \quad & f_i(x) \leq 0 \quad i = 1, \dots, m \\ & h_i(x) = 0 \quad i = 1, \dots, p \end{aligned}$$

Let us introduce the **Lagrangian function** $L : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}$

$$L(x, \lambda, \nu) = f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{i=1}^{\nu} \nu_i h_i(x)$$

Duality

Consider

$$\begin{aligned} \min_x \quad & f_0(x) \\ \text{s.t.} \quad & f_i(x) \leq 0 \quad i = 1, \dots, m \\ & h_i(x) = 0 \quad i = 1, \dots, p \end{aligned}$$

Let us introduce the **Lagrangian function** $L : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}$

$$L(x, \lambda, \nu) = f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{i=1}^{\nu} \nu_i h_i(x)$$

and, accordingly, let us derive the **dual function** $g : \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}$

$$g(\lambda, \nu) = \inf_{x \in \mathcal{D}} L(x, \lambda, \nu) = \inf_{x \in \mathcal{D}} \left(f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{i=1}^{\nu} \nu_i h_i(x) \right)$$

Duality

Consider

$$\begin{aligned} \min_x \quad & f_0(x) \\ \text{s.t.} \quad & f_i(x) \leq 0 \quad i = 1, \ldots, m \\ & h_i(x) = 0 \quad i = 1, \ldots, p \end{aligned}$$

Let us introduce the **Lagrangian function** $L : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}$

$$L(x, \lambda, \nu) = f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{i=1}^{\nu} \nu_i h_i(x)$$

and, accordingly, let us derive the **dual function** $g : \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}$

$$g(\lambda, \nu) = \inf_{x \in \mathcal{D}} L(x, \lambda, \nu) = \inf_{x \in \mathcal{D}} \left(f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{i=1}^{\nu} \nu_i h_i(x) \right)$$

Fact g is concave and $g(\lambda, \nu) \leq p_*$

Duality

Lagrange dual problem

$$\max_{\lambda,\nu} \quad g(\lambda,\nu)$$
$$\text{s.t.} \quad \lambda \geq 0$$

Let d_* be the optimal solution.

Weak duality : $d_* \leq p_*$

Strong duality : $d_* = p_*$ (*zero duality gap condition*)

Whay duality?

Lagrange dual problem

$$\begin{aligned} \max_{\lambda,\nu} \quad & g(\lambda,\nu) \\ \text{s.t.} \quad & \lambda \geq 0 \end{aligned}$$

- dual problem is unconstrained or has simple constraints;
- dual objective is differentiable or has a simple nondifferentiable term

Dual problem in our case

Question : What is the Lagrangian dual problem associated to

$$\begin{aligned} \min_{x,y} \quad & f(x) + g(y) \\ \text{s.t.} \quad & Ax + By = c \end{aligned}$$

where $x \in \mathbb{R}^n$, $y \in \mathbb{R}^m$, $A \in \mathbb{R}^{p\times n}$, $B \in \mathbb{R}^{p\times m}$, $c \in \mathbb{R}^p$?

Dual problem in our case

Question : What is the Lagrangian dual problem associated to

$$\min_{x,y} \quad f(x) + g(y)$$
$$\text{s.t.} \quad Ax + By = c$$

where $x \in \mathbb{R}^n$, $y \in \mathbb{R}^m$, $A \in \mathbb{R}^{p\times n}$, $B \in \mathbb{R}^{p\times m}$, $c \in \mathbb{R}^p$?

Lagrangian (λ vector of Lagrange multipliers, $\rho > 0$)

$$\mathcal{L}_\rho(x, y, \lambda) = f(x) + g(y) + \lambda^\top (Ax + By - c)$$

Dual problem in our case

Question : What is the Lagrangian dual problem associated to

$$\begin{aligned} \min_{x,y} \quad & f(x) + g(y) \\ \text{s.t.} \quad & Ax + By = c \end{aligned}$$

where $x \in \mathbb{R}^n$, $y \in \mathbb{R}^m$, $A \in \mathbb{R}^{p\times n}$, $B \in \mathbb{R}^{p\times m}$, $c \in \mathbb{R}^p$?

Lagrangian (λ vector of Lagrange multipliers, $\rho > 0$)

$$\mathcal{L}_\rho(x, y, \lambda) = f(x) + g(y) + \lambda^\top (Ax + By - c)$$

Dual Function :

$$\begin{aligned} d(\lambda) &= \min_{x,y} \mathcal{L}(x, y, \lambda) \\ &= \underbrace{\min_x \left(f(x) + \lambda^\top Ax \right)}_{d_f(\lambda)} + \underbrace{\min_y \left(g(y) + \lambda^\top (By - c) \right)}_{d_g(\lambda)} \end{aligned}$$

Dual problem in our case

Dual Problem :

$$\begin{aligned} \max_{\lambda} d(\lambda) &= \min_{\lambda} -d(\lambda) \\ &= \min_{\lambda} -d_f(\lambda) - d_g(\lambda) \qquad (*) \end{aligned}$$

Since the original problem

$$\begin{aligned} \min_{x,y} \quad & f(x) + g(y) \\ \text{s.t.} \quad & Ax + By = c \end{aligned}$$

is defined over convex functions and linear constraints **strong duality** holds.

We apply *iterative KM with PRS operator*

$$z_{k+1} = (1-\alpha) z_k + \alpha \, T_{PRS}(z_k)$$

to problem in $(*)$.

PRS operator to dual problem

$$\min_\lambda \; -d_f(\lambda) - d_g(\lambda) \qquad (*)$$

- $d_f(\lambda) = \min_x \left(f(x) + \lambda^\top Ax\right)$
- $d_g(\lambda) = \min_y \left(g(y) + \lambda^\top (By - c)\right)$

$$\min_x \quad f(x) + g(x)$$

Iteration of KM with PRS operator

$$z_{k+1} = (1-\alpha)z_k + \alpha\, T_{PRS}(z_k),$$

or equivalently

$$\begin{aligned} (i) &\quad x_k = \text{prox}_{\rho g}(z_k) \\ (ii) &\quad \xi_k = \text{prox}_{\rho f}(2x_k - z_k) \\ (iii) &\quad z_{k+1} = z_k + 2\alpha(\xi_k - x_k) \end{aligned}$$

In our case

$$\begin{aligned} (i) &\quad \lambda_k = \text{prox}_{-\rho d_g}(z_k) \\ (ii) &\quad \xi_k = \text{prox}_{-\rho d_f}(2\lambda_k - z_k) \\ (iii) &\quad z_{k+1} = z_k + 2\alpha(\xi_k - \lambda_k) \end{aligned}$$

PRS operator to dual problem

$\min_\lambda \; -d_f(\lambda) - d_g(\lambda) \qquad (*)$

- $d_f(\lambda) = \min_x \left(f(x) + \lambda^\top Ax\right)$
- $d_g(\lambda) = \min_y \left(g(y) + \lambda^\top (By - c)\right)$

Computationally

$$(i) \quad y_k = \operatorname*{argmin}_y \left\{g(y) - z_k{}^\top (By - c) + \frac{\rho}{2}\|By - c\|^2\right\}$$

$$\lambda_k = z_k - \rho(By_k - c)$$

$$\min_x \quad f(x) + g(x)$$

Iteration of KM with PRS operator

$$z_{k+1} = (1-\alpha) z_k + \alpha\, T_{PRS}(z_k),$$

or equivalently

$$\begin{aligned} (i) \quad & x_k = \text{prox}_{\rho g}(z_k) \\ (ii) \quad & \xi_k = \text{prox}_{\rho f}(2x_k - z_k) \\ (iii) \quad & z_{k+1} = z_k + 2\alpha(\xi_k - x_k) \end{aligned}$$

In our case

$$\begin{aligned} (i) \quad & \lambda_k = \text{prox}_{-\rho d_g}(z_k) \\ (ii) \quad & \xi_k = \text{prox}_{-\rho d_f}(2\lambda_k - z_k) \\ (iii) \quad & z_{k+1} = z_k + 2\alpha(\xi_k - \lambda_k) \end{aligned}$$

PRS operator to dual problem

$$\min_\lambda \; -d_f(\lambda) - d_g(\lambda) \qquad (*)$$

- $d_f(\lambda) = \min_x \left(f(x) + \lambda^\top Ax\right)$
- $d_g(\lambda) = \min_y \left(g(y) + \lambda^\top (By - c)\right)$

Computationally

$$(i) \quad y_k = \operatorname*{argmin}_y \left\{ g(y) - {z_k}^\top (By - c) + \frac{\rho}{2}\|By - c\|^2 \right\}$$

$$\lambda_k = z_k - \rho(By_k - c)$$

$$(ii) \quad x_k = \operatorname*{argmin}_x \left\{ f(x) - (2\lambda_k - z_k)^\top Ax + \frac{\rho}{2}\|Ax\|^2 \right\}$$

$$\xi_k = 2\lambda_k - z_k - \rho A x_k$$

$$\min_x \quad f(x) + g(x)$$

Iteration of KM with PRS operator

$$z_{k+1} = (1-\alpha) z_k + \alpha \, T_{PRS}(z_k),$$

or equivalently

$$\begin{aligned} (i) \quad & x_k = \operatorname{prox}_{\rho g}(z_k) \\ (ii) \quad & \xi_k = \operatorname{prox}_{\rho f}(2x_k - z_k) \\ (iii) \quad & z_{k+1} = z_k + 2\alpha(\xi_k - x_k) \end{aligned}$$

In our case

$$\begin{aligned} (i) \quad & \lambda_k = \operatorname{prox}_{-\rho d_g}(z_k) \\ (ii) \quad & \xi_k = \operatorname{prox}_{-\rho d_f}(2\lambda_k - z_k) \\ (iii) \quad & z_{k+1} = z_k + 2\alpha(\xi_k - \lambda_k) \end{aligned}$$

PRS operator to dual problem

$$\min_\lambda \; -d_f(\lambda) - d_g(\lambda) \qquad (*)$$

Computationally

$$(i) \quad y_k = \underset{y}{\operatorname{argmin}} \left\{ g(y) - {z_k}^\top (By - c) + \frac{\rho}{2} \|By - c\|^2 \right\}$$

$$\lambda_k = z_k - \rho(By_k - c)$$

$$(ii) \quad x_k = \underset{x}{\operatorname{argmin}} \left\{ f(x) - (2\lambda_k - z_k)^\top Ax + \frac{\rho}{2} \|Ax\|^2 \right\}$$

$$\xi_k = 2\lambda_k - z_k - \rho A x_k$$

$$(iii) \quad z_{k+1} = z_k + 2\alpha(\xi_k - \lambda_k)$$

A bit of redundancy, variable ξ and z are not needed

PRS operator to dual problem

$$\min_\lambda \; -d_f(\lambda) - d_g(\lambda) \qquad (*)$$

Computationally

$$(i) \quad y_k = \underset{y}{\operatorname{argmin}} \left\{ g(y) - {z_k}^\top (By - c) + \frac{\rho}{2}\|By - c\|^2 \right\}$$

$$\lambda_k = z_k - \rho(By_k - c)$$

$$(ii) \quad x_k = \underset{x}{\operatorname{argmin}} \left\{ f(x) - (2\lambda_k - z_k)^\top Ax + \frac{\rho}{2}\|Ax\|^2 \right\}$$

$$\xi_k = 2\lambda_k - z_k - \rho A x_k$$

$$(iii) \quad z_{k+1} = z_k + 2\alpha(\xi_k - \lambda_k)$$

A bit of redundancy, variable ξ and z ar

Relaxed ADMM - classical ADMM for $\alpha = 1/2$)

$$y_{k+1} = \underset{y}{\operatorname{argmin}} \left\{ \mathcal{L}_\rho(x_k, y, \lambda_k) + \rho(2\alpha - 1)(By)^\top (Ax_k) + By_k - c \right\}$$

$$\lambda_{k+1} = \lambda_k + \rho\,(Ax_k + By_{k+1} - c) - \rho(2\alpha - 1)(Ax_k + By_k - c)$$

$$x_{k+1} = \underset{x}{\operatorname{argmin}}\, \mathcal{L}_\rho(x, y_{k+1}, \lambda_{k+1})$$

PRS operator to dual problem

$$\min_\lambda \; -d_f(\lambda) - d_g(\lambda) \qquad (*)$$

Computationally

$$(i) \quad y_k = \underset{y}{\operatorname{argmin}} \left\{ g(y) - z_k{}^\top (By - c) + \frac{\rho}{2}\|By - c\|^2 \right\}$$

$$\lambda_k = z_k - \rho(By_k - c)$$

$$(ii) \quad x_k = \underset{x}{\operatorname{argmin}} \left\{ f(x) - (2\lambda_k - z_k)^\top Ax + \frac{\rho}{2}\|Ax\|^2 \right\}$$

$$\xi_k = 2\lambda_k - z_k - \rho A x_k$$

$$(iii) \quad z_{k+1} = z_k + 2\alpha(\xi_k - \lambda_k)$$

In our specific distributed optimization problem (A has a specific structure and B is the identity) variable y, λ and ξ_k are not needed

Distributed Relaxed-ADMM

Algorithm

Node i keeps in memory x_i and $\{z_{ij}\}_{j\in\mathcal{N}_i}$

1. Node i computes $x_{i,k+1}$ as

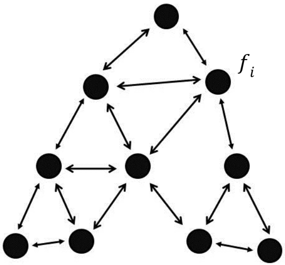
$$x_{i,k+1} = \underset{x_i}{\operatorname{argmin}} \left\{ f_i(x_i) - \left(\sum_{j\in\mathcal{N}_i} z_{ij,k} \right)^\top x_i + \frac{\rho d_i}{2} \|x_i\|^2 \right\}$$

2. Node i computes and transmits the temporary variable $q_{i\to j}$ for all $j \in \mathcal{N}_i$

$$q_{i\to j} = -z_{ij,k} + 2\rho x_{i,k+1}$$

3. Node i gathers $q_{j\to i}$ from all $j \in \mathcal{N}_i$;
4. Node i computes $z_{ij,k+1}$ as

$$z_{ij,k+1} = (1-\alpha) z_{ij,k} + \alpha q_{j\to i}$$



Friday Afternoon Trajectory

Distributed Optimization

$$\min_{x} \quad f(x) = \sum_{i=1}^{N} f_i(x)$$

$$z(k+1) = (1-\alpha)z(k) \; + \; \alpha \, T_{PRS} \, z(k)$$

Operator Theory

$$\begin{aligned} &\underset{x,y}{\text{minimize}} && f(x) + g(y) \\ &\text{subject that} && Ax + By = c \end{aligned}$$

ADMM – Relaxed ADMM

Distributed Relaxed-ADMM

Remark: So far reliable and synchronous communications.

Question: what about if nodes are not synchronized?

Question: what about if a packet is lost?

Asynchronous and robust relaxed-ADMM

Algorithm

Suppose node i is active ai iteration k.

1. Node i computes $x_{i,k+1}$ as

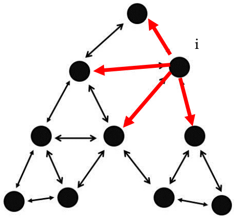
$$x_{i,k+1} = \underset{x_i}{\operatorname{argmin}} \left\{ f_i(x_i) - \left(\sum_{j \in \mathcal{N}_i} z_{ij,k} \right)^\top x_i + \frac{\rho d_i}{2} \|x_i\|^2 \right\}$$

2. Node i computes and transmits the temporary variable $q_{i \to j}$ for all $j \in \mathcal{N}_i$

$$q_{i \to j} = -z_{ij,k} + 2\rho x_{i,k+1}$$

3. For $j \in \mathcal{N}_i$, if node j receives $q_{i \to j}$, then it updates $z_{ji,k+1}$ as

$$z_{ji,k+1} = (1-\alpha) z_{ji,k} + \alpha q_{i \to j}$$



Convergence results - Asynchronous and robust relaxed-ADMM

Assumption(*Asynchronous update and transmission)*
At each iteration there is only one node performing the updating step and the transmissions (this node is randomly chosen)

Assumption(*Random packet losses)*
Each transmitted packet can be lost accordign to a certain probability.

Proposition (*conditions on α and ρ for convergence)*
If $0 < \alpha < 1$ and $\rho > 0$ then the asynchronous and robust distributed Relaxed ADMM converges almost surely to the optimal solution.

Proposition (*conditions for exponential) convergence*
If f_i are strongly convex then convergence is exponentially fast in mean-square sense.

Stochastic Krasnosel'skii-Mann iteration

Let T be a non-expansive operator

Krasnosel'skii-Mann iteration

$$z_{k+1} = (1-\alpha)z_k + \alpha T_{PRS}(z_k)$$

For $i = 1, \ldots, N$, and for $k \geq 0$, let $\beta_{i,k}$ be a binary random variable such that

$$\mathbb{P}[\beta_{i,k} = 1] = p_i \qquad (p_i \text{ is constant with the respect to index iteration } k)$$

Stochastic Krasnosel'skii-Mann iteration

Let T be a non-expansive operator

Krasnosel'skii-Mann iteration

$$z_{k+1} = (1-\alpha)z_k + \alpha\, T_{PRS}(z_k)$$

For $i = 1, \dots, N$, and for $k \geq 0$, let $\beta_{i,k}$ be a binary random variable such that

$$\mathbb{P}\left[\beta_{i,k} = 1\right] = p_i \qquad (p_i \text{ is constant with the respect to index iteration } k)$$

Stochastic Krasnosel'skii-Mann iteration

$$z_{i,k+1} = \begin{cases} (1-\alpha)z_{i,k} + \alpha\left[T(z_k)\right]_i & \text{if} \quad \beta_{i,k} = 1 \\ z_{i,k} & \text{if} \quad \beta_{i,k} = 0 \end{cases}$$

Stochastic Krasnosel'skii-Mann iteration

Let T be a non-expansive operator

Krasnosel'skii-Mann iteration

$$z_{k+1} = (1-\alpha) z_k + \alpha\, T_{PRS}(z_k)$$

For $i = 1, \ldots, N$, and for $k \geq 0$, let $\beta_{i,k}$ be a binary random variable such that

$$\mathbb{P}\left[\beta_{i,k} = 1\right] = p_i \qquad (p_i \text{ is constant with the respect to index iteration } k)$$

Stochastic Krasnosel'skii-Mann iteration

$$z_{i,k+1} = \begin{cases} (1-\alpha) z_{i,k} + \alpha \left[T(z_k)\right]_i & \text{if} \quad \beta_{i,k} = 1 \\ z_{i,k} & \text{if} \quad \beta_{i,k} = 0 \end{cases}$$

Proposition. The trajectory z_k, $k = 0, 1, 2, \ldots$, generated by the Stochastic Krasnosel'skii-Mann iteration converges almost surely to a fixed point of T.

Constrained-coupled optimization

Consider a constraint-coupled optimization problem

$$\begin{aligned} \min_{x_1,\ldots,x_N} \quad & \sum_{i=1}^{N} f_i(x_i) \\ \text{subj. to} \quad & \sum_{i=1}^{N} (H_i x_i - b_i) = 0 \\ & x_i \in \mathcal{X}_i, \qquad i = 1, \ldots, N \end{aligned}$$

ADMM-oriented reformulation of cc-opt

By manipulating the coupling constraint, the optimization problem can be reframed as

$$
\begin{aligned}
\min_{\substack{x_1,\dots,x_N \\ q_1,\dots,q_N}} \quad & \sum_{i=1}^{N} f_i(x_i) \\
\text{subj. to} \quad & H_i x_i = q_i, \qquad\qquad i = 1, \dots, N \\
& \sum_{i=1}^{N} (q_i - b_i) = 0 \\
& x_i \in \mathcal{X}_i, \qquad\qquad i = 1, \dots, N
\end{aligned}
$$

ADMM-oriented cc-opt in compact form

Let

- $\mathbf{1} \coloneqq (1, \dots, 1) \otimes I_p$
- $H_\mathrm{d} \coloneqq \mathrm{diag}(H_1, \dots, H_N)$
- collect $b \coloneqq (b_1, \dots, b_N)$ so that $\mathbf{1}^\top b = \sum_{i=1}^N b_i$

Then, we can write

$$\begin{aligned} \min_{x,q} \quad & f(x) \\ \text{subj. to} \quad & H_\mathrm{d} x = q \\ & \mathbf{1}^\top q = \mathbf{1}^\top b \\ & x \in \mathcal{X} \end{aligned}$$

The augmented Lagrangian is

$$\begin{aligned} L_c(x, q, \boldsymbol{\lambda}) &= f(x) + \boldsymbol{\lambda}^\top (H_\mathrm{d} x - q) + \tfrac{c}{2} \|H_\mathrm{d} x - q\|^2 \\ &= f(x) + \boldsymbol{\lambda}^\top (H_\mathrm{d} x - q) + \tfrac{1}{2c} \|c(H_\mathrm{d} x - q)\|^2 \\ &= f(x) + \tfrac{1}{2c} \|c(H_\mathrm{d} x - q) + \boldsymbol{\lambda}\|^2 - \tfrac{1}{2c} \|\boldsymbol{\lambda}\|^2 \end{aligned}$$

Useful result

Consider the constrained projection of $x_c \in \mathbb{R}^N$ onto $\{x \in \mathbb{R}^N \mid \mathbf{1}^\top x = \beta\}$, obtained as the solution of

$$\min_{x \in \mathbb{R}^N} \ \tfrac{1}{2}\|x - x_c\|^2$$
$$\text{subj. to } \mathbf{1}^\top x = \beta$$

The KKT conditions for this problem read

$$\begin{aligned}(x_\star - x_c) + \mathbf{1}\lambda_\star &= 0\\ \mathbf{1}^\top x_\star &= \beta\end{aligned}$$

From the first we obtain $x_\star = x_c - \mathbf{1}\lambda_\star$, which plugged in the second gives $\mathbf{1}^\top x_c - N\lambda_\star = \beta$. Hence

$$\lambda_\star = \tfrac{1}{N}(\mathbf{1}^\top x_c - \beta)$$

Therefore, the optimal solution can be expressed as

$$\begin{aligned}x_\star &= x_c - \tfrac{1}{N}\mathbf{1}(\mathbf{1}^\top x_c - \beta)\\ &= (I - J)x_c + \tfrac{1}{N}\mathbf{1}\beta\end{aligned}$$

with $J := \frac{1}{N}\mathbf{1}\mathbf{1}^\top$

ADMM for cc-opt

The ADMM reads

$$
\begin{aligned}
x_{k+1} &= \underset{x \in \mathcal{X}}{\operatorname{argmin}}\ f(x) + \tfrac{1}{2c}\left\| c\left(H_{\mathrm{d}} x - q_k\right) + \boldsymbol{\lambda}_k \right\|^2 \\
q_{k+1} &= \underset{q\,:\,\mathbf{1}^\top q = \mathbf{1}^\top b}{\operatorname{argmin}}\ \left\| q - \left(H_{\mathrm{d}} x_{k+1} + \tfrac{1}{c}\boldsymbol{\lambda}_k\right)\right\|^2 \\
\boldsymbol{\lambda}_{k+1} &= \boldsymbol{\lambda}_k + c\left(H_{\mathrm{d}} x_{k+1} - q_{k+1}\right)
\end{aligned}
$$

It holds

$$
\begin{aligned}
q_{k+1} &= H_{\mathrm{d}} x_{k+1} + \tfrac{1}{c}\boldsymbol{\lambda}_k - \tfrac{1}{N}\mathbf{1}(\mathbf{1}^\top (H_{\mathrm{d}} x_{k+1} + \tfrac{1}{c}\boldsymbol{\lambda}_k) - \mathbf{1}^\top b) \\
&= (I - J)(H_{\mathrm{d}} x_{k+1} + \tfrac{1}{c}\boldsymbol{\lambda}_k) + Jb
\end{aligned}
$$

Update simplifications

Substituting q_{k+1} in the update of $\boldsymbol{\lambda}_{k+1}$ gives

$$\begin{aligned}\boldsymbol{\lambda}_{k+1} &= \boldsymbol{\lambda}_k + c\left(H_{\mathrm{d}} x_{k+1} - (I - J)(H_{\mathrm{d}} x_{k+1} + \tfrac{1}{c}\boldsymbol{\lambda}_k) - Jb\right)\\ &= J\boldsymbol{\lambda}_k + c\,J(H_{\mathrm{d}} x_{k+1} - b)\end{aligned}$$

Remark. $\boldsymbol{\lambda}_k$ remains in the span of $\mathbf{1}$, namely $\boldsymbol{\lambda}_k := \mathbf{1}\lambda_k$ for all $k \in \mathbb{N}$

Hence, the dual update simplifies to a *lower-dimension update* given by

$$\lambda_{k+1} = \lambda_k + \tfrac{c}{N}\mathbf{1}^\top(H_{\mathrm{d}} x_{k+1} - b)$$

Putting back this fact in the expression of q_{k+1} results in

$$\begin{aligned}q_{k+1} &= (I - J)(H_{\mathrm{d}} x_{k+1} + \tfrac{1}{c}\mathbf{1}\lambda_k) + Jb\\ &= H_{\mathrm{d}} x_{k+1} - J(H_{\mathrm{d}} x_{k+1} - b)\end{aligned}$$

Update simplifications (continued)

Plugging the final expression for q_k in the optimization step yields

$$x_{k+1} = \operatorname*{argmin}_{x \in \mathcal{X}} \; f(x) + \tfrac{1}{2c} \| c\,(H_{\mathrm{d}} x - H_{\mathrm{d}} x_k) + \mathbf{1}\lambda_k + \underbrace{c\, J(H_{\mathrm{d}} x_k - b)}_{\mathbf{1}\sigma_k} \|^2$$

where we have defined

$$\sigma_k := \tfrac{c}{N} \mathbf{1}^\top (H_{\mathrm{d}} x_k - b)$$

Remark. The following identity holds $q_k = H_{\mathrm{d}} x_k - \mathbf{1}\sigma_k$

Exploiting the definition of σ_k, the (scalar) dual update reads

$$\lambda_{k+1} = \lambda_k + \sigma_{k+1}$$

ADMM for cc-opt is a parallel algorithm

Each agent $i = 1, \dots, N$ solves the local problem

$$x_{i,k+1} \in \operatorname*{argmin}_{x_i \in \mathcal{X}_i} \; f_i(x_i) + \tfrac{1}{2c}\left\| c\left(H_i x_i - H_i x_{i,k}\right) + \lambda_k + \sigma_k \right\|^2$$

Then, the master node updates the global variables

$$\sigma_{k+1} = \frac{c}{N}\left(\sum_{i=1}^{N}(H_i x_{i,k+1} - b_i)\right)$$

$$\lambda_{k+1} = \lambda_k + \sigma_{k+1}$$

Remark. The variable σ_k is the *average* of the local feasibility errors $c\,(H_i x_{i,k} - b_i)$

Tracking-ADMM

Idea. σ_k is the *average* of the local feasibility errors $c\,(H_i x_{i,k} - b_i)$. Hence, we can use the *dynamic average consensus* to obtain a distributed algorithm

Introduce a local copy of σ_k, denoted by $\sigma_{i,k}$, which is updated according to

$$\sigma_{i,k+1} = \sum_{j \in N_i} w_{ij}\sigma_{j,k} + c\,(H_i x_{i,k+1} - H_i x_{i,k})$$

where we canceled the common terms $-b_i$

Introduce a local copy of λ_k, denoted by $\lambda_{i,k}$, which is updated according to

$$\lambda_{i,k+1} = \sum_{j \in N_i} w_{ij}\lambda_{j,k} + \sigma_{i,k+1}$$

The local optimization is

$$x_{i,k+1} \in \underset{x_i \in \mathcal{X}_i}{\operatorname{argmin}} \; f_i(x_i) + \tfrac{1}{2c}\left\|c\,(H_i x_i - H_i x_{i,k}) + \sigma_{i,k} + \lambda_{i,k}\right\|^2$$