

A Primal–Dual Penalty Method via Rounded Weighted- ℓ_1 Lagrangian Duality

Regina S. Burachik

STEM-University of South Australia

Variational Analysis Seminar
Mathematics of Computations and Optimization
(MoCaO – 23 September 2020)

Collaborators

This talk contains joint work with:

- C. Yalçın Kaya (University of South Australia)
- Christopher J. Price (University of Canterbury, New Zealand)

Collaborators

This talk contains joint work with:

- C. Yalçın Kaya (University of South Australia)
- Christopher J. Price (University of Canterbury, New Zealand)

Collaborators

This talk contains joint work with:

- C. Yalçın Kaya (University of South Australia)
- Christopher J. Price (University of Canterbury, New Zealand)

Outline

- 1 Duality Framework for non-convex optimization
- 2 Solving the sequence of dual problems
 - Search Direction
 - Stopping Criterion
- 3 Convergence Analysis
 - Convergence of dual values
 - Finite convergence
 - Primal convergence
- 4 Numerical Experiments
 - An equality-constrained problem
 - Kissing number problem
 - Markov–Dubins path

Outline

- 1 Duality Framework for non-convex optimization
- 2 Solving the sequence of dual problems
 - Search Direction
 - Stopping Criterion
- 3 Convergence Analysis
 - Convergence of dual values
 - Finite convergence
 - Primal convergence
- 4 Numerical Experiments
 - An equality-constrained problem
 - Kissing number problem
 - Markov–Dubins path

Outline

- 1 Duality Framework for non-convex optimization
- 2 Solving the sequence of dual problems
 - Search Direction
 - Stopping Criterion
- 3 Convergence Analysis
 - Convergence of dual values
 - Finite convergence
 - Primal convergence
- 4 Numerical Experiments
 - An equality-constrained problem
 - Kissing number problem
 - Markov–Dubins path

Outline

- 1 Duality Framework for non-convex optimization
- 2 Solving the sequence of dual problems
 - Search Direction
 - Stopping Criterion
- 3 Convergence Analysis
 - Convergence of dual values
 - Finite convergence
 - Primal convergence
- 4 Numerical Experiments
 - An equality-constrained problem
 - Kissing number problem
 - Markov–Dubins path

The Primal Problem

$$\text{minimize } f(x) \quad \text{s.t. } x \text{ in } X, \quad h(x) = 0, \quad g(x) \leq 0, \quad (P)$$


- $f: Y \rightarrow \mathbb{R}, h: Y \rightarrow \mathbb{R}^m, g: Y \rightarrow \mathbb{R}^r$. continuous
- Y a metric space, X compact subset of Y ,
- $\emptyset \subseteq X_0 := \{x \in X : h(x) = 0, g(x) \leq 0\} \subseteq X$,
- M_P optimal value, $\emptyset \neq S^*$ set of solutions.

 (P) not convex. Typical approach: use penalty methods and make (P) unconstrained!

The Primal Problem

$$\text{minimize } f(x) \quad \text{s.t. } x \text{ in } X, \quad h(x) = 0, \quad g(x) \leq 0, \quad (P)$$


- $f : Y \rightarrow \mathbb{R}, h : Y \rightarrow \mathbb{R}^m, g : Y \rightarrow \mathbb{R}^r$, continuous
- Y a metric space, X compact subset of Y ,
- $\emptyset \subsetneq X_0 := \{x \in X : h(x) = 0, g(x) \leq 0\} \subsetneq X$,
- M_P optimal value, $\emptyset \neq S^*$ set of solutions.

 (P) not convex. **Typical approach:** use penalty methods and make (P) unconstrained!

The Primal Problem

$$\text{minimize } f(x) \quad \text{s.t. } x \text{ in } X, \quad h(x) = 0, \quad g(x) \leq 0, \quad (P)$$


- $f : Y \rightarrow \mathbb{R}$, $h : Y \rightarrow \mathbb{R}^m$, $g : Y \rightarrow \mathbb{R}^r$, continuous
- Y a metric space, X compact subset of Y ,
- $\emptyset \subsetneq X_0 := \{x \in X : h(x) = 0, g(x) \leq 0\} \subsetneq X$,
- M_P optimal value, $\emptyset \neq S^*$ set of solutions.

 (P) not convex. Typical approach: use penalty methods and make (P) unconstrained!

The Primal Problem

$$\text{minimize } f(x) \quad \text{s.t. } x \text{ in } X, \quad h(x) = 0, \quad g(x) \leq 0, \quad (P)$$


- $f : Y \rightarrow \mathbb{R}$, $h : Y \rightarrow \mathbb{R}^m$, $g : Y \rightarrow \mathbb{R}^r$, continuous
- Y a metric space, X compact subset of Y ,
- $\emptyset \subsetneq X_0 := \{x \in X : h(x) = 0, g(x) \leq 0\} \subsetneq X$,
- M_P optimal value, $\emptyset \neq S^*$ set of solutions.

 (P) not convex. Typical approach: use penalty methods and make (P) unconstrained!

The Primal Problem

$$\text{minimize } f(x) \quad \text{s.t. } x \text{ in } X, \quad h(x) = 0, \quad g(x) \leq 0, \quad (P)$$


- $f : Y \rightarrow \mathbb{R}$, $h : Y \rightarrow \mathbb{R}^m$, $g : Y \rightarrow \mathbb{R}^r$, continuous
- Y a metric space, X compact subset of Y ,
- $\emptyset \subsetneq X_0 := \{x \in X : h(x) = 0, g(x) \leq 0\} \subsetneq X$,
- M_P optimal value, $\emptyset \neq S^*$ set of solutions.

 (P) not convex. Typical approach: use penalty methods and make (P) unconstrained!

The Primal Problem

$$\text{minimize } f(x) \quad \text{s.t. } x \text{ in } X, \quad h(x) = 0, \quad g(x) \leq 0, \quad (P)$$

- $f : Y \rightarrow \mathbb{R}$, $h : Y \rightarrow \mathbb{R}^m$, $g : Y \rightarrow \mathbb{R}^r$, continuous
- Y a metric space, X compact subset of Y ,
- $\emptyset \subsetneq X_0 := \{x \in X : h(x) = 0, g(x) \leq 0\} \subsetneq X$,
- M_P optimal value, $\emptyset \neq S^*$ set of solutions.

 (P) not convex. **Typical approach:** use penalty methods and make (P) unconstrained!

Our approach: ℓ_1 -penalty vs. smooth ℓ_1 -penalty

The (weighted) ℓ_1 -penalty function for (P) is

$$\varphi_{\ell_1}(x, u, v) := f(x) + \sum_{i=1}^m u_i |h_i(x)| + \sum_{j=1}^r v_j [g_j(x)]_+,$$

where $u \in \mathbb{R}_{++}^m$, $v \in \mathbb{R}_{++}^r$, $[\cdot]_+ := \max\{\cdot, 0\}$.

Our approach: ℓ_1 -penalty vs. smooth ℓ_1 -penalty

The (weighted) ℓ_1 -penalty function for (P) is

$$\varphi_{\ell_1}(x, u, v) := f(x) + \sum_{i=1}^m u_i |h_i(x)| + \sum_{j=1}^r v_j [g_j(x)]_+,$$

where $u \in \mathbb{R}_{++}^m$, $v \in \mathbb{R}_{++}^r$, $[\cdot]_+ := \max\{\cdot, 0\}$.

For (P) smooth, ℓ_1 -penalty function not attractive (destroys smoothness).

Our approach: ℓ_1 -penalty vs. smooth ℓ_1 -penalty

The (weighted) ℓ_1 -penalty function for (P) is

$$\varphi_{\ell_1}(x, u, v) := f(x) + \sum_{i=1}^m u_i |h_i(x)| + \sum_{j=1}^r v_j [g_j(x)]_+,$$

where $u \in \mathbb{R}_{++}^m$, $v \in \mathbb{R}_{++}^r$, $[\cdot]_+ := \max\{\cdot, 0\}$.

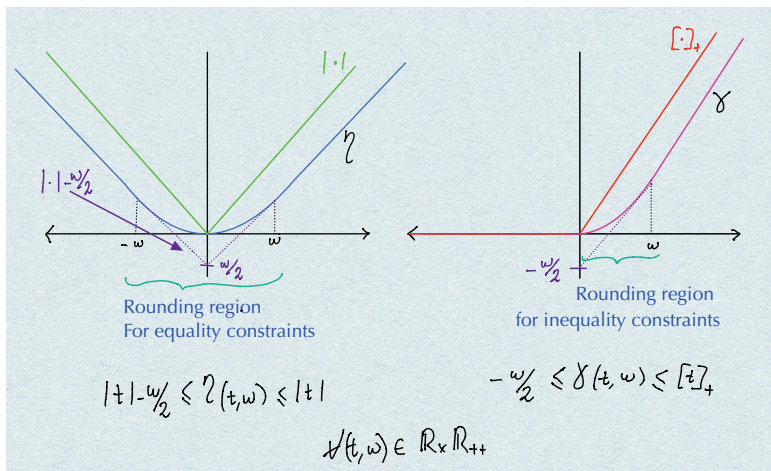
Our aim: devise a penalty approach that preserves the smoothness, and it behaves in a similar way as the ℓ_1 -penalty

Two auxiliary functions

Let $w \geq 0$. Define $\eta : \mathbb{R} \times \mathbb{R}_+ \rightarrow \mathbb{R}_+$, $\gamma : \mathbb{R} \times \mathbb{R}_+ \rightarrow \mathbb{R}_+$:

$$\eta(t, w) := \begin{cases} \frac{t^2}{2w} & \text{if } |t| < w, \\ |t| - \frac{w}{2} & \text{if } |t| \geq w. \end{cases} \quad \left| \quad \gamma(t, w) := \begin{cases} \frac{t^2}{2w}, & \text{if } 0 < t < w, \\ t - \frac{w}{2}, & \text{if } t \geq w, \\ 0, & \text{if } t \leq 0. \end{cases}$$

The functions η and γ



Properties of η and γ

- AKA *scaled Huber function*, η is quadratic over $(-w, w)$, a *rounding region* of ‘width’ w . Outside the rounding region, η behaves as an ℓ_1 -penalty term.
- η is a smooth minorant of $|\cdot|$. Hence, for small values of $w > 0$, it is a smooth approximation of $|\cdot|$.
- Similarly, γ is a smooth approximation of $[\cdot]_+ := \max\{\cdot, 0\}$ for small values of $w > 0$.
- Note $\eta = \min_{v \in \mathbb{R}} \{|v| + \frac{1}{2w}(x - v)^2\} =: |\cdot| \square \frac{1}{2w}(\cdot)^2$.

Properties of η and γ

- AKA *scaled Huber function*, η is quadratic over $(-w, w)$, a *rounding region* of ‘width’ w . Outside the rounding region, η behaves as an ℓ_1 -penalty term.
- η is a smooth minorant of $|\cdot|$. Hence, for small values of $w > 0$, it is a smooth approximation of $|\cdot|$.
- Similarly, γ is a smooth approximation of $[\cdot]_+ := \max\{\cdot, 0\}$ for small values of $w > 0$.
- Note $\eta = \min_{v \in \mathbb{R}} \{|v| + \frac{1}{2w}(x - v)^2\} =: |\cdot| \square \frac{1}{2w}(\cdot)^2$.

Properties of η and γ

- AKA *scaled Huber function*, η is quadratic over $(-w, w)$, a *rounding region* of ‘width’ w . Outside the rounding region, η behaves as an ℓ_1 -penalty term.
- η is a smooth minorant of $|\cdot|$. Hence, for small values of $w > 0$, it is a smooth approximation of $|\cdot|$.
- Similarly, γ is a smooth approximation of $[\cdot]_+ := \max\{\cdot, 0\}$ for small values of $w > 0$.
- Note $\eta = \min_{v \in \mathbb{R}} \{|v| + \frac{1}{2w}(x - v)^2\} =: |\cdot| \square \frac{1}{2w}(\cdot)^2$.

Properties of η and γ

- AKA *scaled Huber function*, η is quadratic over $(-w, w)$, a *rounding region* of ‘width’ w . Outside the rounding region, η behaves as an ℓ_1 -penalty term.
- η is a smooth minorant of $|\cdot|$. Hence, for small values of $w > 0$, it is a smooth approximation of $|\cdot|$.
- Similarly, γ is a smooth approximation of $[\cdot]_+ := \max\{\cdot, 0\}$ for small values of $w > 0$.
- Note $\eta = \min_{v \in \mathbb{R}} \{|v| + \frac{1}{2w}(x - v)^2\} =: |\cdot| \square \frac{1}{2w}(\cdot)^2$.

- We have seen these and many more examples of “smoothing functions” in Jein-Shan Chen’s VA seminar August 26!
- Check his slides for more on smoothing functions and techniques!
- We concentrate from now on in the construction of the Lagrangian by means of η and γ

- We have seen these and many more examples of “smoothing functions” in Jein-Shan Chen’s VA seminar August 26!
- Check his slides for more on smoothing functions and techniques!
- We concentrate from now on in the construction of the Lagrangian by means of η and γ

- We have seen these and many more examples of “smoothing functions” in Jein-Shan Chen’s VA seminar August 26!
- Check his slides for more on smoothing functions and techniques!
- We concentrate from now on in the construction of the Lagrangian by means of η and γ

The ℓ_1 -penalty function

Fix $w \geq 0$. For f, h, g as in (P) , recall φ_{ℓ_1} is

$$\varphi_{\ell_1}(x, u, v) := f(x) + \sum_{i=1}^m u_i |h_i(x)| + \sum_{j=1}^r v_j [g_j(x)]_+,$$

where $(u, v) > 0$,

The rounded ℓ_1 -Lagrangian

Fix $w \geq 0$. For f, h, g as in (P) , set L_w as

$$L_w(x, u, v) := f(x) + \sum_{i=1}^m u_i \eta(h_i(x), w) + \sum_{j=1}^r v_j \gamma(g_j(x), w),$$

where $(u, v) > 0$, from the definitions:

$$|L_w(x, u, v) - \varphi_{\ell_1}(x, u, v)| \leq \frac{w}{2} (\|u\|_1 + \|v\|_1),$$



We treat (u, v) as a dual variable, and w as a parameter!

The rounded ℓ_1 -Lagrangian

Fix $w \geq 0$. For f, h, g as in (P) , set L_w as

$$L_w(x, u, v) := f(x) + \sum_{i=1}^m u_i \eta(h_i(x), w) + \sum_{j=1}^r v_j \gamma(g_j(x), w),$$

where $(u, v) \geq 0$, from the definitions:

$$|L_w(x, u, v) - \varphi_{\ell_1}(x, u, v)| \leq \frac{w}{2}(\|u\|_1 + \|v\|_1),$$



We treat (u, v) as a dual variable, and w as a parameter!

The rounded ℓ_1 -Lagrangian

Fix $w \geq 0$. For f, h, g as in (P) , set L_w as

$$L_w(x, u, v) := f(x) + \sum_{i=1}^m u_i \eta(h_i(x), w) + \sum_{j=1}^r v_j \gamma(g_j(x), w),$$

where $(u, v) \geq 0$, from the definitions:

$$|L_w(x, u, v) - \varphi_{\ell_1}(x, u, v)| \leq \frac{w}{2}(\|u\|_1 + \|v\|_1),$$



We treat (u, v) as a dual variable, and w as a parameter!

The Dual Problem for fixed $w > 0$

Dual function q_w defined as

$$q_w(u, v) := \begin{cases} \min_{x \in X} L_w(x, u, v), & \text{if } (u, v) > 0, \\ -\infty & \text{c.c.}, \end{cases}$$

Dual problem (D_w) is given by

$$\sup_{(u, v) \in \mathbb{R}_{++}^m \times \mathbb{R}_{++}^r} q_w(u, v).$$

We denote its optimal value by M_{D_w} .



Generates a new dual problem for each $w > 0$!

The Dual Problem for fixed $w > 0$

Dual function q_w defined as

$$q_w(u, v) := \begin{cases} \min_{x \in X} L_w(x, u, v), & \text{if } (u, v) > 0, \\ -\infty & \text{c.c.}, \end{cases}$$

Dual problem (D_w) is given by

$$\sup_{(u, v) \in \mathbb{R}_{++}^m \times \mathbb{R}_{++}^r} q_w(u, v).$$

We denote its optimal value by M_{D_w} .



Generates a new dual problem for each $w > 0$!

The Dual Problem for fixed $w > 0$

Dual function q_w defined as

$$q_w(u, v) := \begin{cases} \min_{x \in X} L_w(x, u, v), & \text{if } (u, v) > 0, \\ -\infty & \text{c.c.}, \end{cases}$$

Dual problem (D_w) is given by

$$\sup_{(u, v) \in \mathbb{R}_{++}^m \times \mathbb{R}_{++}^r} q_w(u, v).$$

We denote its optimal value by M_{D_w} .



Generates a new dual problem for each $w > 0$!

The Dual Problem for fixed $w > 0$

Dual function q_w defined as

$$q_w(u, v) := \begin{cases} \min_{x \in X} L_w(x, u, v), & \text{if } (u, v) > 0, \\ -\infty & \text{c.c.}, \end{cases}$$

Dual problem (D_w) is given by

$$\sup_{(u, v) \in \mathbb{R}_{++}^m \times \mathbb{R}_{++}^r} q_w(u, v).$$

We denote its optimal value by M_{D_w} .



Generates a new dual problem for each $w > 0$!

The Dual Problem for fixed $w > 0$

Dual function q_w defined as

$$q_w(u, v) := \begin{cases} \min_{x \in X} L_w(x, u, v), & \text{if } (u, v) > 0, \\ -\infty & \text{c.c.}, \end{cases}$$

Dual problem (D_w) is given by

$$\sup_{(u, v) \in \mathbb{R}_{++}^m \times \mathbb{R}_{++}^r} q_w(u, v).$$

We denote its optimal value by M_{D_w} .



Generates a new dual problem for each $w > 0$!

Properties of (D_w) for $w > 0$ fixed

q_w is concave and continuous over its domain

- **Dual Function Values attained:** $\forall (u, v, w) > 0$, exists \hat{x} s.t.

$$q_w(u, v) = L_w(\hat{x}, u, v) \quad (\hat{x} \text{ depends on } (u, v, w))$$

- **Weak duality:** $\forall (u, v, w), q_w(u, v) \leq M_P.$
- **Therefore:** $M_{D_w} \leq M_P$ for all $w > 0$

Properties of (D_w) for $w > 0$ fixed

q_w is concave and continuous over its domain

- **Dual Function Values attained:** $\forall (u, v, w) > 0$, exists \hat{x} s.t.

$$q_w(u, v) = L_w(\hat{x}, u, v) \quad (\hat{x} \text{ depends on } (u, v, w))$$

- **Weak duality:** $\forall (u, v, w), q_w(u, v) \leq M_P$.
- **Therefore:** $M_{D_w} \leq M_P$ for all $w > 0$

Properties of (D_w) for $w > 0$ fixed

q_w is concave and continuous over its domain

- **Dual Function Values attained:** $\forall (u, v, w) > 0$, exists \hat{x} s.t.

$$q_w(u, v) = L_w(\hat{x}, u, v) \quad (\hat{x} \text{ depends on } (u, v, w))$$

- **Weak duality:** $\forall (u, v, w), q_w(u, v) \leq M_P$.

- **Therefore:** $M_{D_w} \leq M_P$ for all $w > 0$

Properties of (D_w) for $w > 0$ fixed

q_w is concave and continuous over its domain

- **Dual Function Values attained:** $\forall (u, v, w) > 0$, exists \hat{x} s.t.

$$q_w(u, v) = L_w(\hat{x}, u, v) \quad (\hat{x} \text{ depends on } (u, v, w))$$

- **Weak duality:** $\forall (u, v, w), q_w(u, v) \leq M_P$.
- **Therefore:** $M_{D_w} \leq M_P$ for all $w > 0$

Asymptotic Strong duality for $w_k \downarrow$:

$$L_k := L_{w_k}, D_k := D_{w_k}$$

- $\forall k, L_k \leq L_{k+1} \leq \varphi_{\ell_1}$, and $M_{D_k} \leq M_{D_{k+1}} \leq M_P$.

- If $w_k \downarrow 0$, then $\lim_{k \rightarrow \infty} L_k(\cdot) = \varphi_{\ell_1}(\cdot)$.

- If $w_k \downarrow w \geq 0$, then $\lim_{k \rightarrow \infty} M_{D_k} = \sup_{k \in \mathbb{N}} M_{D_k} = M_P$.

Asymptotic Strong duality for $w_k \downarrow$:

$$L_k := L_{w_k}, D_k := D_{w_k}$$

- $\forall k, L_k \leq L_{k+1} \leq \varphi_{\ell_1}$, and $M_{D_k} \leq M_{D_{k+1}} \leq M_P$.

- If $w_k \downarrow 0$, then

$$\lim_{k \rightarrow \infty} L_k(\cdot) = \varphi_{\ell_1}(\cdot).$$

- If $w_k \downarrow w \geq 0$, then

$$\lim_{k \rightarrow \infty} M_{D_k} = \sup_{k \in \mathbb{N}} M_{D_k} = M_P.$$

Asymptotic Strong duality for $w_k \downarrow$:

$$L_k := L_{w_k}, D_k := D_{w_k}$$

- $\forall k, L_k \leq L_{k+1} \leq \varphi_{\ell_1}$, and $M_{D_k} \leq M_{D_{k+1}} \leq M_P$.

- If $w_k \downarrow 0$, then

$$\lim_{k \rightarrow \infty} L_k(\cdot) = \varphi_{\ell_1}(\cdot).$$

- If $w_k \downarrow w \geq 0$, then

$$\lim_{k \rightarrow \infty} M_{D_k} = \sup_{k \in \mathbb{N}} M_{D_k} = M_P.$$

Asymptotic Strong duality for $w_k \downarrow$:

$$L_k := L_{w_k}, D_k := D_{w_k}$$

- $\forall k, L_k \leq L_{k+1} \leq \varphi_{\ell_1}$, and $M_{D_k} \leq M_{D_{k+1}} \leq M_P$.

- If $w_k \downarrow 0$, then

$$\lim_{k \rightarrow \infty} L_k(\cdot) = \varphi_{\ell_1}(\cdot).$$

- If $w_k \downarrow w \geq 0$, then

$$\lim_{k \rightarrow \infty} M_{D_k} = \sup_{k \in \mathbb{N}} M_{D_k} = M_P.$$



This justifies solving (D_k) !

Asymptotic Strong duality for $w_k \downarrow$:

$$L_k := L_{w_k}, D_k := D_{w_k}$$

- $\forall k, L_k \leq L_{k+1} \leq \varphi_{\ell_1}$, and $M_{D_k} \leq M_{D_{k+1}} \leq M_P$.

- If $w_k \downarrow 0$, then

$$\lim_{k \rightarrow \infty} L_k(\cdot) = \varphi_{\ell_1}(\cdot).$$

- If $w_k \downarrow w \geq 0$, then

$$\lim_{k \rightarrow \infty} M_{D_k} = \sup_{k \in \mathbb{N}} M_{D_k} = M_P.$$



This justifies solving (D_k) !

Measuring Infeasibility of x : define the vectors

$$p_\eta(x, w) := \begin{bmatrix} \eta(h_1(x), w) \\ \vdots \\ \eta(h_m(x), w) \end{bmatrix} \geq 0,$$

Measuring Infeasibility of x : define the vectors

$$p_\eta(x, w) := \begin{bmatrix} \eta(h_1(x), w) \\ \vdots \\ \eta(h_m(x), w) \end{bmatrix} \geq 0,$$

(x 's violation of $h(x) = 0$)

Measuring Infeasibility of x : define the vectors

$$p_\eta(x, w) := \begin{bmatrix} \eta(h_1(x), w) \\ \vdots \\ \eta(h_m(x), w) \end{bmatrix} \geq 0, \quad p_\gamma(x, w) := \begin{bmatrix} \gamma(g_1(x), w) \\ \vdots \\ \gamma(g_r(x), w) \end{bmatrix} \geq 0,$$

Measuring Infeasibility of x : define the vectors

$$p_\eta(x, w) := \begin{bmatrix} \eta(h_1(x), w) \\ \vdots \\ \eta(h_m(x), w) \end{bmatrix} \geq 0, \quad p_\gamma(x, w) := \begin{bmatrix} \gamma(g_1(x), w) \\ \vdots \\ \gamma(g_r(x), w) \end{bmatrix} \geq 0,$$

(x 's violation of $g(x) \leq 0$)

Measuring Infeasibility of x : define the vectors

$$p_\eta(x, w) := \begin{bmatrix} \eta(h_1(x), w) \\ \vdots \\ \eta(h_m(x), w) \end{bmatrix} \geq 0, \quad p_\gamma(x, w) := \begin{bmatrix} \gamma(g_1(x), w) \\ \vdots \\ \gamma(g_r(x), w) \end{bmatrix} \geq 0,$$

Set $p(x, w) := (p_\eta(x, w), p_\gamma(x, w))$. Fix $(u, v, w) > 0$, and

$$x(u, v, w) =: \hat{x} \in \operatorname{Argmin}_X L_w(x, u, v).$$

Measuring Infeasibility of x : define the vectors

$$p_\eta(x, w) := \begin{bmatrix} \eta(h_1(x), w) \\ \vdots \\ \eta(h_m(x), w) \end{bmatrix} \geq 0, \quad p_\gamma(x, w) := \begin{bmatrix} \gamma(g_1(x), w) \\ \vdots \\ \gamma(g_r(x), w) \end{bmatrix} \geq 0,$$

Set $p(x, w) := (p_\eta(x, w), p_\gamma(x, w))$. Fix $(u, v, w) > 0$, and

$x(u, v, w) =: \hat{x} \in \text{Argmin}_X L_w(x, u, v)$.



$p(\hat{x}, w) := (p_\eta(\hat{x}, w), p_\gamma(\hat{x}, w)) \in \partial q_w(u, v)$.

The search direction (II)



Moreover, denote $z := (u, v) \geq 0$ and $\pi_0 \geq 0$. Recall
 $\hat{x} \in \text{Argmin}_X L_w(x, u, v)$

$$p(\hat{x}, w) + \pi_0 \in \partial_\varepsilon q_w(z),$$

i.e., an ε -superdifferential of q_w , whenever $\varepsilon \geq \langle z, \pi_0 \rangle$



$p(\hat{x}, w) + \pi_0$ provides search direction for improving q_w !

The search direction (II)



Moreover, denote $z := (u, v) \geq 0$ and $\pi_0 \geq 0$. Recall $\hat{x} \in \text{Argmin}_X L_w(x, u, v)$


$$p(\hat{x}, w) + \pi_0 \in \partial_\varepsilon q_w(z),$$

i.e., an ε -superdifferential of q_w , whenever $\varepsilon \geq \langle z, \pi_0 \rangle$



$p(\hat{x}, w) + \pi_0$ provides search direction for improving q_w !

The search direction (II)

 Moreover, denote $z := (u, v) \geq 0$ and $\pi_0 \geq 0$. Recall

$$\hat{x} \in \operatorname{Argmin}_X L_w(x, u, v)$$

$$p(\hat{x}, w) + \pi_0 \in \partial_\varepsilon q_w(z),$$

i.e., an ε -superdifferential of q_w , whenever $\varepsilon \geq \langle z, \pi_0 \rangle$

 $p(\hat{x}, w) + \pi_0$ provides search direction for improving q_w !

The search direction (II)



Moreover, denote $z := (u, v) \geq 0$ and $\pi_0 \geq 0$. Recall $\hat{x} \in \text{Argmin}_X L_w(x, u, v)$

$$p(\hat{x}, w) + \pi_0 \in \partial_\varepsilon q_w(z),$$

i.e., an ε -superdifferential of q_w , whenever $\varepsilon \geq \langle z, \pi_0 \rangle$



$p(\hat{x}, w) + \pi_0$ provides search direction for improving q_w !

Feasibility implies Optimality! By definition of η, γ :

$$p(x, w) = 0 \iff x \in X_0, \forall w \geq 0.$$

Let $S(D) := \{(u, v, w) > 0 : q_w(u, v) = M_P\}$

$$\hat{x} \in \operatorname{Argmin}_X L_w(x, u, v) \cap X_0 \iff (\hat{x}, (u, v, w)) \in S^* \times S(D).$$

In this situation, $q_w(u, v) = L_w(\hat{x}, (u, v)) = f(\hat{x}) = M_P$ and



(u, v) are exact penalty parameters!

Feasibility implies Optimality! By definition of η, γ :

$$p(x, w) = 0 \iff x \in X_0, \forall w \geq 0.$$

Let $S(D) := \{(u, v, w) > 0 : q_w(u, v) = M_P\}$

$$\hat{x} \in \operatorname{Argmin}_X L_w(x, u, v) \cap X_0 \iff (\hat{x}, (u, v, w)) \in S^* \times S(D).$$

In this situation, $q_w(u, v) = L_w(\hat{x}, (u, v)) = f(\hat{x}) = M_P$ and



(u, v) are exact penalty parameters!

Feasibility implies Optimality! By definition of η, γ :

$$p(x, w) = 0 \iff x \in X_0, \forall w \geq 0.$$

Let $S(D) := \{(u, v, w) > 0 : q_w(u, v) = M_P\}$

$$\hat{x} \in \operatorname{Argmin}_X L_w(x, u, v) \cap X_0 \iff (\hat{x}, (u, v, w)) \in S^* \times S(D).$$

In this situation, $q_w(u, v) = L_w(\hat{x}, (u, v)) = f(\hat{x}) = M_P$ and



(u, v) are exact penalty parameters!

Feasibility implies Optimality! By definition of η, γ :

$$p(x, w) = 0 \iff x \in X_0, \forall w \geq 0.$$

Let $S(D) := \{(u, v, w) > 0 : q_w(u, v) = M_P\}$

$$\hat{x} \in \operatorname{Argmin}_X L_w(x, u, v) \cap X_0 \iff (\hat{x}, (u, v, w)) \in S^* \times S(D).$$

In this situation, $q_w(u, v) = L_w(\hat{x}, (u, v)) = f(\hat{x}) = M_P$ and



(u, v) are exact penalty parameters!

Feasibility implies Optimality! By definition of η, γ :

$$p(x, w) = 0 \iff x \in X_0, \forall w \geq 0.$$

Let $S(D) := \{(u, v, w) > 0 : q_w(u, v) = M_P\}$

$$\hat{x} \in \text{Argmin}_X L_w(x, u, v) \cap X_0 \iff (\hat{x}, (u, v, w)) \in S^* \times S(D).$$

In this situation, $q_w(u, v) = L_w(\hat{x}, (u, v)) = f(\hat{x}) = M_P$ and



(u, v) are exact penalty parameters!

Feasibility implies Optimality! By definition of η, γ :

$$p(x, w) = 0 \iff x \in X_0, \forall w \geq 0.$$

Let $S(D) := \{(u, v, w) > 0 : q_w(u, v) = M_P\}$

$$\hat{x} \in \operatorname{Argmin}_X L_w(x, u, v) \cap X_0 \iff (\hat{x}, (u, v, w)) \in S^* \times S(D).$$

In this situation, $q_w(u, v) = L_w(\hat{x}, (u, v)) = f(\hat{x}) = M_P$ and



(u, v) are exact penalty parameters!

The Algorithm (ε -subgradient method)

Choose $w_k \downarrow w \geq 0$. Recall $z^k = (u^k, v^k) > 0$

Step 1 Set $k = 1$. Select $(z^1, w_1) > 0$.

Step 2 Find $x^k \in \text{Argmin}_X L_k(\cdot, z^k)$, compute

$p^k := p(x^k, w_k)$. If $p^k = 0$, STOP.

Otherwise, go to **Step 3**.

Step 3 Set $z^{k+1} := z^k + s_k(p^k + d^k)$, with $s_k > 0$, $d^k \geq 0$.

Set $k := k + 1$ and go to **Step 2**.

The Algorithm (ε -subgradient method)

Choose $w_k \downarrow w \geq 0$. Recall $z^k = (u^k, v^k) > 0$

Step 1 Set $k = 1$. Select $(z^1, w_1) > 0$.

Step 2 Find $x^k \in \text{Argmin}_X L_k(\cdot, z^k)$, compute

$p^k := p(x^k, w_k)$. If $p^k = 0$, STOP.

Otherwise, go to **Step 3**.

Step 3 Set $z^{k+1} := z^k + s_k (p^k + d^k)$, with $s_k > 0$, $d^k \geq 0$.

Set $k := k + 1$ and go to **Step 2**.

The Algorithm (ε -subgradient method)

Choose $w_k \downarrow w \geq 0$. Recall $z^k = (u^k, v^k) > 0$

Step 1 Set $k = 1$. Select $(z^1, w_1) > 0$.

Step 2 Find $x^k \in \text{Argmin}_X L_k(\cdot, z^k)$, compute

$p^k := p(x^k, w_k)$. If $p^k = 0$, STOP.

Otherwise, go to **Step 3**.

Step 3 Set $z^{k+1} := z^k + s_k (p^k + d^k)$, with $s_k > 0$, $d^k \geq 0$.

Set $k := k + 1$ and go to **Step 2**.

The Algorithm (ε -subgradient method)

Choose $w_k \downarrow w \geq 0$. Recall $z^k = (u^k, v^k) > 0$

Step 1 Set $k = 1$. Select $(z^1, w_1) > 0$.

Step 2 Find $x^k \in \text{Argmin}_X L_k(\cdot, z^k)$, compute

$p^k := p(x^k, w_k)$. If $p^k = 0$, STOP.

Otherwise, go to **Step 3**.

Step 3 Set $z^{k+1} := z^k + s_k (p^k + d^k)$, with $s_k > 0$, $d^k \geq 0$.

Set $k := k + 1$ and go to **Step 2**.

Properties of the Algorithm

If Algorithm stops in Step 2: It does so at a **primal-dual** solution: $x^k \in S^*$ and $(z^k, w_k) \in S(D)$. This fact justifies the stopping criteria.

♥ What choices of s_k and d^k ensure convergence?



Classical subgradient direction **doesn't necessarily** improve objective value at each iteration \Rightarrow difficult to find good stepsize!

Properties of the Algorithm

If Algorithm stops in Step 2: It does so at a **primal-dual** solution: $x^k \in S^*$ and $(z^k, w_k) \in S(D)$. This fact justifies the stopping criteria.

♥ **What choices of s_k and d^k ensure convergence?**



Classical subgradient direction **doesn't necessarily** improve objective value at each iteration \Rightarrow difficult to find good stepsize!

Properties of the Algorithm

If Algorithm stops in Step 2: It does so at a **primal-dual** solution: $x^k \in S^*$ and $(z^k, w_k) \in S(D)$. This fact justifies the stopping criteria.

♥ What choices of s_k and d^k ensure convergence?



Classical subgradient direction **doesn't necessarily** improve objective value at each iteration \Rightarrow difficult to find good stepsize!

Convergence Results for every choice of $s_k > 0$

- $z^k \rightarrow z \iff \sum_{k=1}^{\infty} s_k(p^k + d^k) < \infty \iff (z^k) \text{ bounded.}$

- Monotone Convergence of dual values:

$$M_P \geq q_k := q_{w_k}(z^k) \uparrow \text{ (not necessarily strictly)}$$

Convergence Results for every choice of $s_k > 0$

- $z^k \rightarrow z \iff \sum_{k=1}^{\infty} s_k(p^k + d^k) < \infty \iff (z^k)$ bounded.

- Monotone Convergence of dual values:

$$M_P \geq q_k := q_{w_k}(z^k) \uparrow \text{ (not necessarily strictly)}$$

Primal or Dual Optimality means finite termination

$$p^k = 0 \iff x^k \in S^* \iff \text{algorithm stops.}$$

Primal or Dual Optimality means finite termination

$$p^k = 0 \iff x^k \in S^* \iff \text{algorithm stops.}$$

So primal optimality \iff finite termination

Primal or Dual Optimality means finite termination

$$p^k = 0 \iff x^k \in S^* \iff \text{algorithm stops.}$$

So **primal optimality** \iff finite termination

If $\exists \hat{d} > 0$ s.t. $d^k \geq \hat{d}$, $\forall k$, and $\exists k_0$ s.t. $(z^{k_0}, w_{k_0}) \in S(D)$ then
either $p^{k_0} = 0$ or $p^{k_0+1} = 0$.

Primal or Dual Optimality means finite termination

$$p^k = 0 \iff x^k \in S^* \iff \text{algorithm stops.}$$

So **primal optimality** \iff finite termination

If $\exists \hat{d} > 0$ s.t. $d^k \geq \hat{d}, \forall k$, and $\exists k_0$ s.t. $(z^{k_0}, w_{k_0}) \in S(D)$ then
either $p^{k_0} = 0$ or $p^{k_0+1} = 0$.

Hence, **dual optimality** \iff finite termination

Primal or Dual Optimality means finite termination

$$p^k = 0 \iff x^k \in S^* \iff \text{algorithm stops.}$$

So **primal optimality** \iff finite termination

If $\exists \hat{d} > 0$ s.t. $d^k \geq \hat{d}, \forall k$, and $\exists k_0$ s.t. $(z^{k_0}, w_{k_0}) \in S(D)$ then
either $p^{k_0} = 0$ or $p^{k_0+1} = 0$.

Hence, **dual optimality** \iff finite termination

So, if $k \rightarrow \infty \Rightarrow x^k \notin S^*, (z^k, w_k) \notin S(D), \forall k$.

Primal-Dual convergence results for two choices of s_k

(a) $s_k := \frac{1}{\|p^k\|_2}$, or (b) $s_k := \|p^k\|_2$, and $w_k \downarrow w \geq 0$. Then,

- Every accumulation point of (x^k) is in S^* and $q_k \uparrow M_P$.
- If (z^k) is bounded, then $z^k \rightarrow z$ with $(z, w) \in S(D)$,
- If $d^k \geq \hat{d} > 0$, then (q_k) is strictly increasing.

Primal-Dual convergence results for two choices of s_k

(a) $s_k := \frac{1}{\|p^k\|_2}$, or (b) $s_k := \|p^k\|_2$, and $w_k \downarrow w \geq 0$. Then,

- Every accumulation point of (x^k) is in S^* and $q_k \uparrow M_P$.
- If (z^k) is bounded, then $z^k \rightarrow z$ with $(z, w) \in S(D)$,
- If $d^k \geq \hat{d} > 0$, then (q_k) is strictly increasing.

Primal-Dual convergence results for two choices of s_k

(a) $s_k := \frac{1}{\|p^k\|_2}$, or (b) $s_k := \|p^k\|_2$, and $w_k \downarrow w \geq 0$. Then,

- Every accumulation point of (x^k) is in S^* and $q_k \uparrow M_P$.
- If (z^k) is bounded, then $z^k \rightarrow z$ with $(z, w) \in S(D)$,
- If $d^k \geq \hat{d} > 0$, then (q_k) is strictly increasing.

Primal-Dual convergence results for two choices of s_k

(a) $s_k := \frac{1}{\|p^k\|_2}$, or (b) $s_k := \|p^k\|_2$, and $w_k \downarrow w \geq 0$. Then,

- Every accumulation point of (x^k) is in S^* and $q_k \uparrow M_P$.
- If (z^k) is bounded, then $z^k \rightarrow z$ with $(z, w) \in S(D)$,
- If $d^k \geq \hat{d} > 0$, then (q_k) is strictly increasing.

The Problem

$$(Pc) \left\{ \begin{array}{ll} \min_{x \in X} & f(x) \\ \text{subject to} & h_i(x) = 0, \quad i = 1, \dots, m, \\ & g_j(x) \leq 0, \quad j = 1, \dots, r, \end{array} \right.$$

with $X \subset \mathbb{R}^n$, compact, and f, h, g are differentiable.

For $c \in \mathbb{R}^t$, denote $\|[c]_+\|_\infty := \max_j [c_j]_+$

The Primal-Dual-Penalty Method (PDPM)

Step 1 Set $z^0 = (u^0, v^0) > 0$, $w_0 = 1$, $\varepsilon > 0$, $k = 0$.

Step 2 (*Subproblem*) Given $z^k = (u^k, v^k)$, compute

$$x^k \in \underset{x \in X}{\operatorname{Argmin}} f(x) + \langle z^k, p(x, w_k) \rangle.$$

Step 3 (*stopping criterion*) If $\max\{\|h(x^k)\|_\infty, \|[g(x^k)]_+\|_\infty\} < \varepsilon$, stop.

Step 4 (*Update penalty parameters*)

$$\begin{aligned} u^{k+1} &:= u^k + s_k p_\eta(x^k, w_k), \\ v^{k+1} &:= v^k + s_k p_\gamma(x^k, w_k), \end{aligned}$$

with $s_k > 0$. Set $k = k + 1$, $w_{k+1} < w_k$, and go to **Step 2**.

For $c \in \mathbb{R}^t$, denote $\|[c]_+\|_\infty := \max_j [c_j]_+$

The Primal-Dual-Penalty Method (PDPM)

Step 1 Set $z^0 = (u^0, v^0) > 0$, $w_0 = 1$, $\varepsilon > 0$, $k = 0$.

Step 2 (*Subproblem*) Given $z^k = (u^k, v^k)$, compute

$$x^k \in \underset{x \in X}{\operatorname{Argmin}} f(x) + \langle z^k, p(x, w_k) \rangle.$$

Step 3 (*stopping criterion*) If $\max\{\|h(x^k)\|_\infty, \|[g(x^k)]_+\|_\infty\} < \varepsilon$, stop.

Step 4 (*Update penalty parameters*)

$$\begin{aligned} u^{k+1} &:= u^k + s_k p_\eta(x^k, w_k), \\ v^{k+1} &:= v^k + s_k p_\gamma(x^k, w_k), \end{aligned}$$

with $s_k > 0$. Set $k = k + 1$, $w_{k+1} < w_k$, and go to Step 2.

For $c \in \mathbb{R}^t$, denote $\|[c]_+\|_\infty := \max_j [c_j]_+$

The Primal-Dual-Penalty Method (PDPM)

Step 1 Set $z^0 = (u^0, v^0) > 0$, $w_0 = 1$, $\varepsilon > 0$, $k = 0$.

Step 2 (*Subproblem*) Given $z^k = (u^k, v^k)$, compute

$$x^k \in \underset{x \in X}{\operatorname{Argmin}} f(x) + \langle z^k, p(x, w_k) \rangle.$$

Step 3 (*stopping criterion*) If $\max\{\|h(x^k)\|_\infty, \|[g(x^k)]_+\|_\infty\} < \varepsilon$, stop.

Step 4 (*Update penalty parameters*)

$$u^{k+1} := u^k + s_k p_\eta(x^k, w_k),$$

$$v^{k+1} := v^k + s_k p_\gamma(x^k, w_k),$$

with $s_k > 0$. Set $k = k + 1$, $w_{k+1} < w_k$, and go to Step 2.

For $c \in \mathbb{R}^t$, denote $\|[c]_+\|_\infty := \max_j [c_j]_+$

The Primal-Dual-Penalty Method (PDPM)

Step 1 Set $z^0 = (u^0, v^0) > 0$, $w_0 = 1$, $\varepsilon > 0$, $k = 0$.

Step 2 (*Subproblem*) Given $z^k = (u^k, v^k)$, compute

$$x^k \in \underset{x \in X}{\operatorname{Argmin}} f(x) + \langle z^k, p(x, w_k) \rangle.$$

Step 3 (*stopping criterion*) If $\max\{\|h(x^k)\|_\infty, \|[g(x^k)]_+\|_\infty\} < \varepsilon$, stop.

Step 4 (*Update penalty parameters*)

$$\begin{aligned} u^{k+1} &:= u^k + s_k p_\eta(x^k, w_k), \\ v^{k+1} &:= v^k + s_k p_\gamma(x^k, w_k), \end{aligned}$$

with $s_k > 0$. Set $k = k + 1$, $w_{k+1} < w_k$, and go to Step 2.

Remarks on this implementation

- Used $d^k = 0, \forall k$ (pure subgradient method)
- Used only $s^k = \frac{1}{\|p(x^k, w_k)\|_2}$, easy to implement and small size in early iterations
- if $s^k = \|p(x^k, w_k)\|_2$, s^k may be too large initially and cause numerical instabilities
- **Subproblem** in **Step 2** solved with modelling language AMPL, paired with optimization software Knitro.
- Other solvers can be used, i.e., Algencan, IPOPT, SNOPT, etc.

Remarks on this implementation

- Used $d^k = 0, \forall k$ (pure subgradient method)
- Used only $s^k = \frac{1}{\|p(x^k, w_k)\|_2}$, easy to implement and small size in early iterations
- if $s^k = \frac{1}{\|p(x^k, w_k)\|_2}$, s^k may be too large initially and cause numerical instabilities
- **Subproblem** in **Step 2** solved with modelling language AMPL, paired with optimization software Knitro.
- Other solvers can be used, i.e., Algencon, IPOPT, SNOPT, etc.

Remarks on this implementation

- Used $d^k = 0, \forall k$ (pure subgradient method)
- Used only $s^k = \frac{1}{\|p(x^k, w_k)\|_2}$, easy to implement and small size in early iterations
- if $s^k = \|p(x^k, w_k)\|_2$, s^k may be too large initially and cause numerical instabilities
- **Subproblem** in **Step 2** solved with modelling language AMPL, paired with optimization software Knitro.
- Other solvers can be used, i.e., Algencon, IPOPT, SNOPT, etc.

Remarks on this implementation

- Used $d^k = 0, \forall k$ (pure subgradient method)
- Used only $s^k = \frac{1}{\|p(x^k, w_k)\|_2}$, easy to implement and small size in early iterations
- if $s^k = \|p(x^k, w_k)\|_2$, s^k may be too large initially and cause numerical instabilities
- **Subproblem** in **Step 2** solved with modelling language AMPL, paired with optimization software Knitro.
- Other solvers can be used, i.e., Algencon, IPOPT, SNOPT, etc.

Remarks on this implementation

- Used $d^k = 0, \forall k$ (pure subgradient method)
- Used only $s^k = \frac{1}{\|p(x^k, w_k)\|_2}$, easy to implement and small size in early iterations
- if $s^k = \|p(x^k, w_k)\|_2$, s^k may be too large initially and cause numerical instabilities
- **Subproblem** in **Step 2** solved with modelling language AMPL, paired with optimization software Knitro.
- Other solvers can be used, i.e., Algencon, IPOPT, SNOPT, etc.

The Numerical Experiments

Tested with non-convex differentiable test problems, so as to use the smoothing advantages.

- We compare with case when (P_c) is solved directly with AMPL+Knitro, in the conventional way, i.e., “Knitro on its own.”
- Overall: method is successful, and, in presence of multiple local minima, method can find deeper minima
- Global minimum can be obtained more often than the case when using a local optimization solver conventionally (on its own)

The Numerical Experiments

Tested with non-convex differentiable test problems, so as to use the smoothing advantages.

- We compare with case when (P_c) is solved directly with AMPL+Knitro, in the conventional way, i.e., “Knitro on its own.”
- Overall: method is successful, and, in presence of multiple local minima, method can find deeper minima
- Global minimum can be obtained more often than the case when using a local optimization solver conventionally (on its own)

The Numerical Experiments

Tested with non-convex differentiable test problems, so as to use the smoothing advantages.

- We compare with case when (P_c) is solved directly with AMPL+Knitro, in the conventional way, i.e., “Knitro on its own.”
- Overall: method is successful, and, in presence of multiple local minima, method can find deeper minima
- Global minimum can be obtained more often than the case when using a local optimization solver conventionally (on its own)

The Numerical Experiments

Tested with non-convex differentiable test problems, so as to use the smoothing advantages.

- We compare with case when (P_c) is solved directly with AMPL+Knitro, in the conventional way, i.e., “Knitro on its own.”
- Overall: method is successful, and, in presence of multiple local minima, method can find deeper minima
- Global minimum can be obtained more often than the case when using a local optimization solver conventionally (on its own)

Problem 79, W. Hock, and K. Schittkowski, 1981

Small-scale problem, $n = 5$ and $m = 3$, $r = 0$.

$$(P1) \begin{cases} \min (x_1 - 1)^2 + (x_1 - x_2)^2 + (x_2 - x_3)^2 + (x_3 - x_4)^4 + (x_4 - x_5)^4 \\ \text{s. t. } x_1 + x_2^2 + x_3^3 - 2 - 3\sqrt{2} = 0, \\ x_2 - x_3^2 + x_4 + 2 - 2\sqrt{2} = 0, x_1 x_5 - 2 = 0, \end{cases}$$

used $w_{k+1} = 1/(k+1)^6$.

We identified six isolated local minimizers, best one coincides with the one reported by Hock-Schittkowski.

Problem 79, W. Hock, and K. Schittkowski, 1981

Small-scale problem, $n = 5$ and $m = 3$, $r = 0$.

$$(P1) \begin{cases} \min (x_1 - 1)^2 + (x_1 - x_2)^2 + (x_2 - x_3)^2 + (x_3 - x_4)^4 + (x_4 - x_5)^4 \\ \text{s. t. } x_1 + x_2^2 + x_3^3 - 2 - 3\sqrt{2} = 0, \\ x_2 - x_3^2 + x_4 + 2 - 2\sqrt{2} = 0, x_1 x_5 - 2 = 0, \end{cases}$$

used $w_{k+1} = 1/(k+1)^6$.

We identified six isolated local minimizers, best one coincides with the one reported by Hock-Schittkowski.

Results for (P1)

	Solution 1	Solution 2	Solution 3	Solution 4	Solution 5	Solution 6
x^*	1.191127	2.717678	-0.766173	-1.246781	0.949471	-2.702207
	1.362603	2.033384	2.666726	2.422242	-2.266633	-2.989944
	1.472818	-0.847948	-0.468170	1.174983	0.537796	0.171917
	1.635017	-0.485941	-1.619116	-0.213229	3.384285	3.847927
	1.679081	0.735922	-2.610377	-1.604131	2.106436	-0.740136
$f(x^*)$	0.0787768	13.9668249	27.4520041	27.5219615	86.5275397	649.5048650
Knitro on its own	45%	12%	7%	11%	18%	7%
PDPM	100%	0%	0%	0%	0%	0%

Table: (P1) – Local optimal solutions and percentage of times “Knitro on its own” or PDPM finds a solution, after 30,000 runs of each approach with random initial guesses.

Observations for (P1), call solution 1 the “global” one.

- “Knitro-on its own” took ≈ 0.023 seconds (averaged over 30,000 runs), while PDPM took 0.12 seconds (five times longer).
- PDPM can be made faster by solving subproblem in [Step 2](#) with coarser feasibility and optimality tolerances, and refining these closer to convergence.
- If the aim is to find solution 1, Knitro on its own will be “expected” to be run more than twice, given the probability 0.45, for it to find solution 1.
- If we run PDPM only once, we get solution 1.

Observations for (P1), call solution 1 the “global” one.

- “Knitro-on its own” took ≈ 0.023 seconds (averaged over 30,000 runs), while PDPM took 0.12 seconds (five times longer).
- PDPM can be made faster by solving subproblem in [Step 2](#) with coarser feasibility and optimality tolerances, and refining these closer to convergence.
- If the aim is to find solution 1, Knitro on its own will be “expected” to be run more than twice, given the probability 0.45, for it to find solution 1.
- If we run PDPM only once, we get solution 1.

Observations for (P1), call solution 1 the “global” one.

- “Knitro-on its own” took ≈ 0.023 seconds (averaged over 30,000 runs), while PDPM took 0.12 seconds (five times longer).
- PDPM can be made faster by solving subproblem in [Step 2](#) with coarser feasibility and optimality tolerances, and refining these closer to convergence.
- If the aim is to find solution 1, Knitro on its own will be “expected” to be run more than twice, given the probability 0.45, for it to find solution 1.
- If we run PDPM only once, we get solution 1.

Observations for (P1), call solution 1 the “global” one.

- “Knitro-on its own” took ≈ 0.023 seconds (averaged over 30,000 runs), while PDPM took 0.12 seconds (five times longer).
- PDPM can be made faster by solving subproblem in [Step 2](#) with coarser feasibility and optimality tolerances, and refining these closer to convergence.
- If the aim is to find solution 1, Knitro on its own will be “expected” to be run more than twice, given the probability 0.45, for it to find solution 1.
- If we run PDPM only once, we get solution 1.

The *kissing number* κ_n

The *kissing number problem* seeks maximum number κ_n of non-overlapping spheres of radius r in \mathbb{R}^n that simultaneously touch (kiss) a central sphere of the same radius. Formulated as smooth problem:

$$(P2) \quad \begin{cases} \max & \alpha \\ \text{subject to} & \|y_k\|^2 = 1, \quad k = 1, \dots, p, \\ & \|y_i - y_j\|^2 \geq \alpha^2, \quad i, j = 1, \dots, p, \quad j > i, \end{cases}$$

where $\alpha \in \mathbb{R}$, $y_k \in \mathbb{R}^n$ (the coordinates of the k th sphere's centre), are the optimization variables. Here, the radii r of the spheres is $r = 1/2$.

We used $w_{k+1} = 1/(k+1)^4$.

The *kissing number* κ_n

The *kissing number problem* seeks maximum number κ_n of non-overlapping spheres of radius r in \mathbb{R}^n that simultaneously touch (kiss) a central sphere of the same radius. Formulated as smooth problem:

$$(P2) \quad \begin{cases} \max & \alpha \\ \text{subject to} & \|y_k\|^2 = 1, \quad k = 1, \dots, p, \\ & \|y_i - y_j\|^2 \geq \alpha^2, \quad i, j = 1, \dots, p, \quad j > i, \end{cases}$$

where $\alpha \in \mathbb{R}$, $y_k \in \mathbb{R}^n$ (the coordinates of the k th sphere's centre), are the optimization variables. Here, the radii r of the spheres is $r = 1/2$.

We used $w_{k+1} = 1/(k+1)^4$.

For $n = 3$, $\kappa_n = 12$

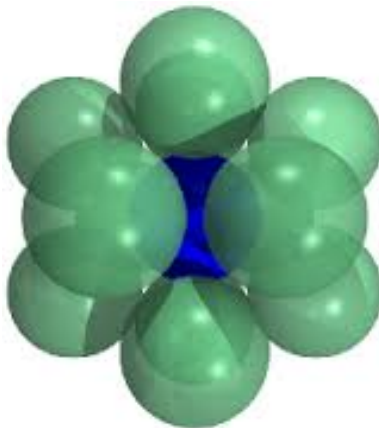


Figure from E.W. Weisstein, Kissing Number,
mathworld.wolfram.com/KissingNumber.html.

Background on (P2)

- The maximum p , for which $\alpha \geq 1$ is κ_n .
- Many stationary points, so κ_n difficult to compute.
- For $n = 1, 2, 3$ and 4 , $\kappa_n = 2, 6, 12$ and 24 , respectively.
- Only lower/upper bounds are known for $n = 5, 6$ and 7 , namely

$$40 \leq \kappa_5 \leq 44, \quad 72 \leq \kappa_6 \leq 78 \quad \text{and} \quad 126 \leq \kappa_7 \leq 134.$$

- Our aim is not to improve these bounds (very hard!), but test PDPM's performance, and compare it with Knitro on its own.

Background on (P2)

- The maximum p , for which $\alpha \geq 1$ is κ_n .
- Many stationary points, so κ_n difficult to compute.
- For $n = 1, 2, 3$ and 4 , $\kappa_n = 2, 6, 12$ and 24 , respectively.
- Only lower/upper bounds are known for $n = 5, 6$ and 7 , namely

$$40 \leq \kappa_5 \leq 44, \quad 72 \leq \kappa_6 \leq 78 \quad \text{and} \quad 126 \leq \kappa_7 \leq 134.$$

- Our aim is not to improve these bounds (very hard!), but test PDPM's performance, and compare it with Knitro on its own.

Background on (P2)

- The maximum p , for which $\alpha \geq 1$ is κ_n .
- Many stationary points, so κ_n difficult to compute.
- For $n = 1, 2, 3$ and 4 , $\kappa_n = 2, 6, 12$ and 24 , respectively.
- Only lower/upper bounds are known for $n = 5, 6$ and 7 , namely
$$40 \leq \kappa_5 \leq 44, \quad 72 \leq \kappa_6 \leq 78 \quad \text{and} \quad 126 \leq \kappa_7 \leq 134.$$
- Our aim is not to improve these bounds (very hard!), but test PDPM's performance, and compare it with Knitro on its own.

Background on (P2)

- The maximum p , for which $\alpha \geq 1$ is κ_n .
- Many stationary points, so κ_n difficult to compute.
- For $n = 1, 2, 3$ and 4 , $\kappa_n = 2, 6, 12$ and 24 , respectively.
- Only lower/upper bounds are known for $n = 5, 6$ and 7 , namely

$$40 \leq \kappa_5 \leq 44, \quad 72 \leq \kappa_6 \leq 78 \quad \text{and} \quad 126 \leq \kappa_7 \leq 134.$$

- Our aim is not to improve these bounds (very hard!), but test PDPM's performance, and compare it with Knitro on its own.

Background on (P2)

- The maximum p , for which $\alpha \geq 1$ is κ_n .
- Many stationary points, so κ_n difficult to compute.
- For $n = 1, 2, 3$ and 4 , $\kappa_n = 2, 6, 12$ and 24 , respectively.
- Only lower/upper bounds are known for $n = 5, 6$ and 7 , namely

$$40 \leq \kappa_5 \leq 44, \quad 72 \leq \kappa_6 \leq 78 \quad \text{and} \quad 126 \leq \kappa_7 \leq 134.$$

- Our aim is not to improve these bounds (very hard!), but test PDPM's performance, and compare it with Knitro on its own.

Results for (P2)

Problem size			Approach	Max of min dist. betw. two spheres			Percentage $\alpha^* > 1$	CPU time [sec]
$\begin{bmatrix} n \\ p \end{bmatrix}$	# of var.	# of constr.		α_{\min}^*	α_{ave}^*	α_{\max}^*		
$\begin{bmatrix} 5 \\ 38 \end{bmatrix}$	191	741	Knitro on its own	0.9807810	0.9927489	1.0019176	0.7%	0.89
			PDPM	0.9912749	0.9968095	1.0037513	16.7%	0.92
$\begin{bmatrix} 6 \\ 62 \end{bmatrix}$	373	1953	Knitro on its own	0.9804186	0.9890666	0.9961310	0.0%	8.8
			PDPM	0.9875817	0.9938880	1.0041174	1.1%	4.7
$\begin{bmatrix} 7 \\ 92 \end{bmatrix}$	645	4278	Knitro on its own	0.9886343	0.9953140	0.9991757	0.0%	65
			PDPM	0.9946906	0.9987890	1.0021734	13.2%	25

Table: (P2) – Numerical results with Knitro on its own and PDPM, after 1,000 runs. The locally optimal solutions (with values α^*) are found by each approach with random initial guesses.

Histogram of local optima: $n = 5, p = 38$

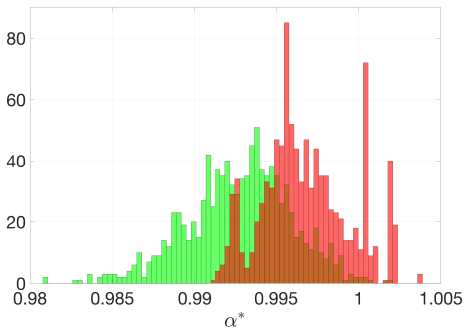


Figure: (P2) Knitro (green) vs. PDPM (red), overlaps (brown).

Histogram of local optima: $n = 6, p = 62$

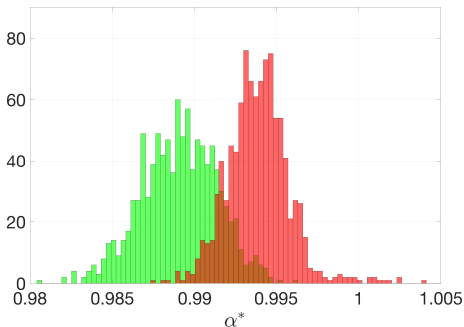


Figure: (P2) Knitro (green) vs. PDPM (red), overlaps (brown)

Histogram of local optima: $n = 7, p = 92$

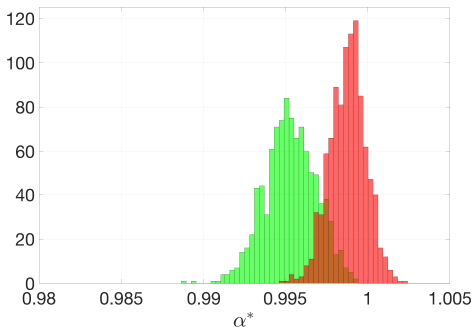


Figure: (P2) Knitro (green) vs. PDPM (red), overlaps (brown).

Observations for (P2)

- Distribution of maxima found by PDPM skewed to the right, indicating tendency to find higher maxima.
- Tendency in finding higher maxima accentuated for larger instances.
- For $n = 6, 7$, PDPM finds viable solution in ≈ 7.2 and 31.7 min, respectively, while Knitro is expected not to find any.

Observations for (P2)

- Distribution of maxima found by PDPM skewed to the right, indicating tendency to find higher maxima.
- Tendency in finding higher maxima accentuated for larger instances.
- For $n = 6, 7$, PDPM finds viable solution in ≈ 7.2 and 31.7 min, respectively, while Knitro is expected not to find any.

Observations for (P2)

- Distribution of maxima found by PDPM skewed to the right, indicating tendency to find higher maxima.
- Tendency in finding higher maxima accentuated for larger instances.
- For $n = 6, 7$, PDPM finds viable solution in ≈ 7.2 and 31.7 min, respectively, while Knitro is expected not to find any.

Posed, studied by Markov, 1889, solved by Dubins, 1957

Aim: Find the shortest planar curve of constrained curvature joining two points with prescribed tangents. [Kaya, 2017] re-formulates as optimal control problem, and by maximum principle reduces it to finite-dimensional, denoted below as (Ps).

$$\left\{ \begin{array}{l} \min \ell = \sum_{j=1}^5 \xi_j, \text{ s.t.} \\ x_0 - x_f + \frac{1}{a} (-\sin \theta_0 + 2 \sin \theta_1 - 2 \sin \theta_2 + 2 \sin \theta_4 - \sin \theta_f) + \xi_3 \cos \theta_2 = 0, \\ y_0 - y_f + \frac{1}{a} (\cos \theta_0 - 2 \cos \theta_1 + 2 \cos \theta_2 - 2 \cos \theta_4 + \cos \theta_f) + \xi_3 \sin \theta_2 = 0, \\ \sin \theta_f = \sin \theta_5, \quad \cos \theta_f = \cos \theta_5, \quad \xi_j \geq 0, \quad \text{for } j = 1, \dots, 5, \\ \text{where } \theta_1 = \theta_0 + a \xi_1, \quad \theta_2 = \theta_1 - a \xi_2, \quad \theta_4 = \theta_2 + a \xi_4, \text{ and} \\ \theta_5 = \theta_4 - a \xi_5. \end{array} \right.$$

Problem (Ps), run PDPM with $w_k = 1/(k + 1)^6$

- After simplification, (Ps) is expressed in terms of ξ and four equality constraints.
- We consider (Ps) with $((x_0, y_0), \theta_0) = ((0, 0), -\pi/3)$,
 $((x_f, y_f), \theta_f) = ((0.4, 0.4), -\pi/6)$, and $a = 3$
- [Kaya, 2017] reports 7 stationary solutions.

Problem (Ps), run PDPM with $w_k = 1/(k + 1)^6$

- After simplification, (Ps) is expressed in terms of ξ and four equality constraints.
- We consider (Ps) with $((x_0, y_0), \theta_0) = ((0, 0), -\pi/3)$,
 $((x_f, y_f), \theta_f) = ((0.4, 0.4), -\pi/6)$, and $a = 3$
- [Kaya, 2017] reports 7 stationary solutions.

Problem (Ps), run PDPM with $w_k = 1/(k + 1)^6$

- After simplification, (Ps) is expressed in terms of ξ and four equality constraints.
- We consider (Ps) with $((x_0, y_0), \theta_0) = ((0, 0), -\pi/3)$,
 $((x_f, y_f), \theta_f) = ((0.4, 0.4), -\pi/6)$, and $a = 3$
- [Kaya, 2017] reports 7 stationary solutions.

Results for (Ps)

	Soln 1	Soln 2	Soln 3	Soln 4	Soln 5	Soln 6	Soln 7
ξ^*	0.0000	0.0000	0.7096	0.8627	0.0000	1.6036	0.0000
	1.5822	1.7354	0.0000	0.3064	0.3818	1.7880	1.6781
	0.5914	0.0000	0.5914	0.0000	0.0000	0.0000	1.0093
	0.0000	0.3063	1.5594	1.7126	1.7880	0.3590	1.8526
	0.3376	0.4908	0.0000	0.0000	1.2317	0.0000	0.0000
ℓ^*	2.5113	2.5326	2.8603	2.8817	3.4015	3.7506	4.5401
Knitro on its own	8%	0%	3%	0%	40%	24%	25%
PDPM	36%	0%	62%	0%	2%	0%	0%

Table: (Ps) – Local optimal solutions, and percentage of the times Knitro vs. PDPM, to find solution of length ℓ^* , after 20,000 runs of each approach with random initial guesses.

Summary of results for (Ps)

- Knitro found 89% of the time solutions with $\ell^* \geq 3.4$, while PDPM found 98% of the time solutions with $\ell^* \leq 2.9$.
- We would expect to find good quality solution (Solutions 1 and 3) by running PDPM just once, in 0.3 sec. Knitro on its own is expected to be run about nine times (≈ 0.4 sec) to get the same quality solutions.
- As for Kissing Number Problem, PDPM is more time-efficient when compared with Knitro, for larger instances.

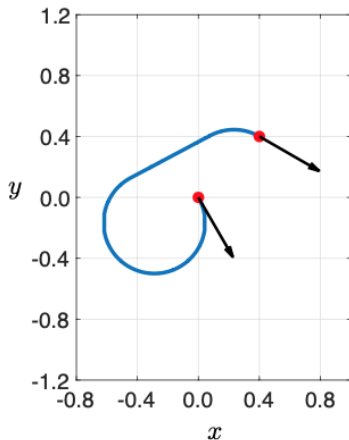
Summary of results for (Ps)

- Knitro found 89% of the time solutions with $\ell^* \geq 3.4$, while PDPM found 98% of the time solutions with $\ell^* \leq 2.9$.
- We would expect to find good quality solution (Solutions 1 and 3) by running PDPM just once, in 0.3 sec. Knitro on its own is expected to be run about nine times (≈ 0.4 sec) to get the same quality solutions.
- As for Kissing Number Problem, PDPM is more time-efficient when compared with Knitro, for larger instances.

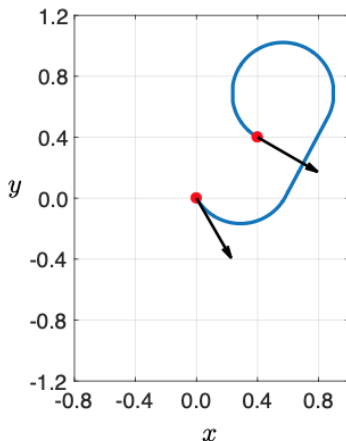
Summary of results for (Ps)

- Knitro found 89% of the time solutions with $\ell^* \geq 3.4$, while PDPM found 98% of the time solutions with $\ell^* \leq 2.9$.
- We would expect to find good quality solution (Solutions 1 and 3) by running PDPM just once, in 0.3 sec. Knitro on its own is expected to be run about nine times (≈ 0.4 sec) to get the same quality solutions.
- As for Kissing Number Problem, PDPM is more time-efficient when compared with Knitro, for larger instances.

Soln 1: $\ell^* = 2.5113$: 8% vs. 36% (*PDPM*)



Soln 3: $\ell^* = 2.8603$: 3% vs. 62% (PDPM)

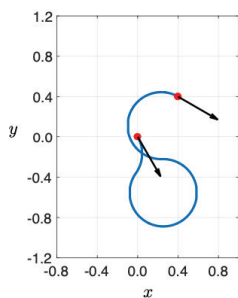


Sol 5: $l^* = 3.4015$: 40% vs. 2%(PDPM)

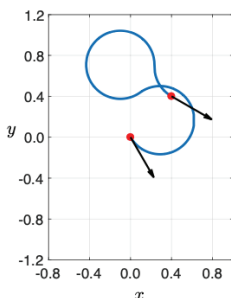
Sol 6: $l^* = 3.7506$: 24% vs. 0%(PDPM)

Sol 7: $l^* = 4.5401$: 25% vs. 0%(PDPM)

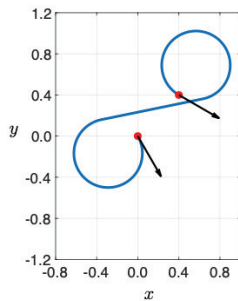
Total: 89% vs. 2%(PDPM)



Sol 5



Sol 6



Sol 7

Conclusion

- We analyzed a primal-dual scheme with a sequence of dual problems, established strong asymptotic duality.
- We used the theory to define a primal-dual ε -subgradient algorithm which reduces w_k in each iteration, forcing the Lagrangians to approach the weighted- ℓ_1 -penalty function.
- The primal sequence accumulates at a solution of (P). If the dual variables happen to be bounded, they converge to a dual solution (an exact penalty parameter)

Conclusion

- We analyzed a primal-dual scheme with a sequence of dual problems, established strong asymptotic duality.
- We used the theory to define a primal-dual ε -subgradient algorithm which reduces w_k in each iteration, forcing the Lagrangians to approach the weighted- ℓ_1 -penalty function.
- The primal sequence accumulates at a solution of (P). If the dual variables happen to be bounded, they converge to a dual solution (an exact penalty parameter)

Conclusion

- We analyzed a primal-dual scheme with a sequence of dual problems, established strong asymptotic duality.
- We used the theory to define a primal-dual ε -subgradient algorithm which reduces w_k in each iteration, forcing the Lagrangians to approach the weighted- ℓ_1 -penalty function.
- The primal sequence accumulates at a solution of (P). If the dual variables happen to be bounded, they converge to a dual solution (an exact penalty parameter)

Open questions

- Can we consider other types of Lagrangians that smoothly approximate a nonsmooth Lagrangian, and devise similar asymptotic schemes?
- What can be said if X is not compact?
- How will the numerical results be affected if we implement an ϵ -subgradient step?
- Extensions to semi-infinite programming?

Open questions

- Can we consider other types of Lagrangians that smoothly approximate a nonsmooth Lagrangian, and devise similar asymptotic schemes?
- What can be said if X is not compact?
- How will the numerical results be affected if we implement an ε -subgradient step?
- Extensions to semi-infinite programming?

Open questions

- Can we consider other types of Lagrangians that smoothly approximate a nonsmooth Lagrangian, and devise similar asymptotic schemes?
- What can be said if X is not compact?
- How will the numerical results be affected if we implement an ε -subgradient step?
- Extensions to semi-infinite programming?

Open questions

- Can we consider other types of Lagrangians that smoothly approximate a nonsmooth Lagrangian, and devise similar asymptotic schemes?
- What can be said if X is not compact?
- How will the numerical results be affected if we implement an ε -subgradient step?
- Extensions to semi-infinite programming?

Open questions

- Can we consider other types of Lagrangians that smoothly approximate a nonsmooth Lagrangian, and devise similar asymptotic schemes?
- What can be said if X is not compact?
- How will the numerical results be affected if we implement an ε -subgradient step?
- Extensions to semi-infinite programming?