To rescale or to project?
Solving quadratic programing problems with Lagrange multipliers methods

Igor Griva

A workshop on Optimization and Operator Theory
Haifa 2017

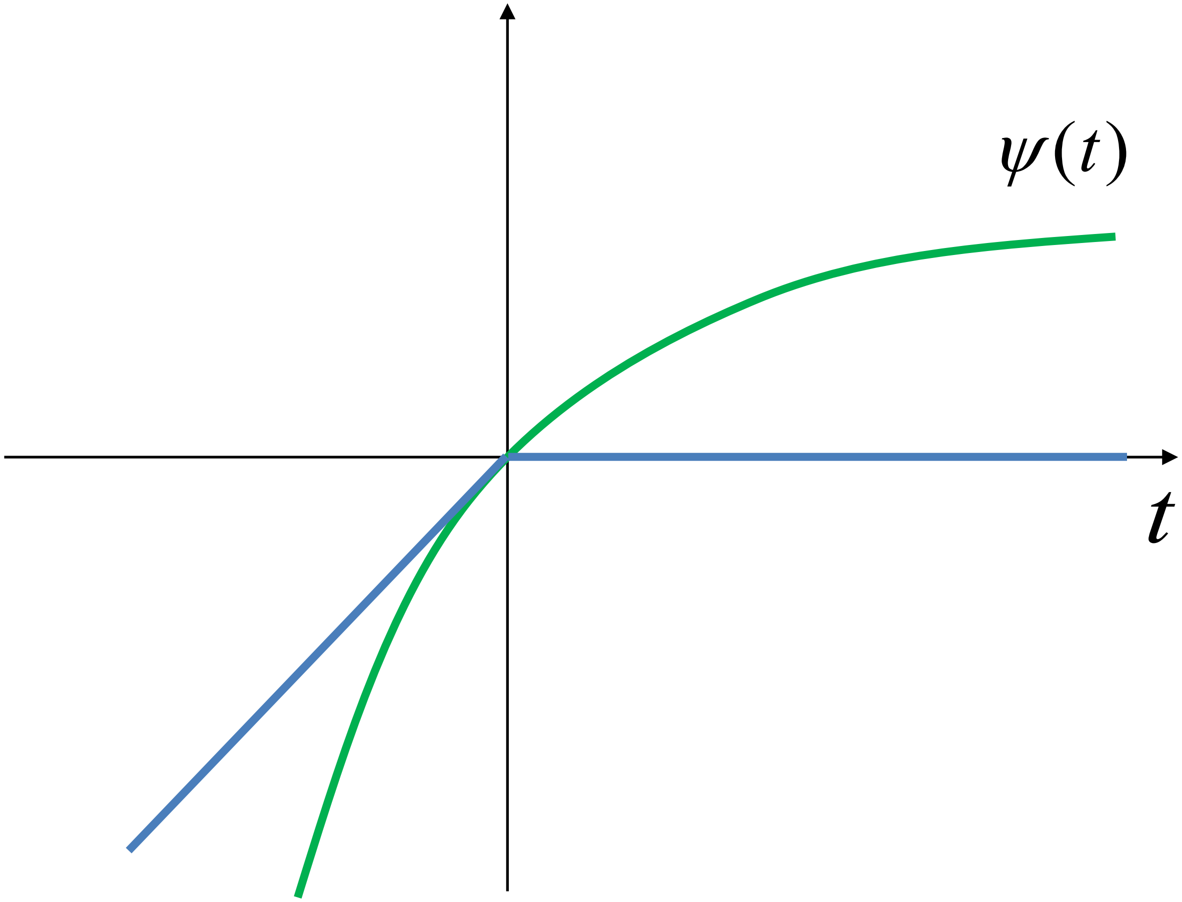
Projection can be computationally inexpensive

$$B = \{x \in \mathbb{R}^n : l_i \leq x_i \leq u_i,\ i = 1, \ldots n\}$$

1. Loop over all $i = 1, \ldots, n$.
2. If $x_i < l_i$ then Set $x_i = l_i$
3. If $x_i > u_i$ then Set $x_i = u_i$
4. Return x.

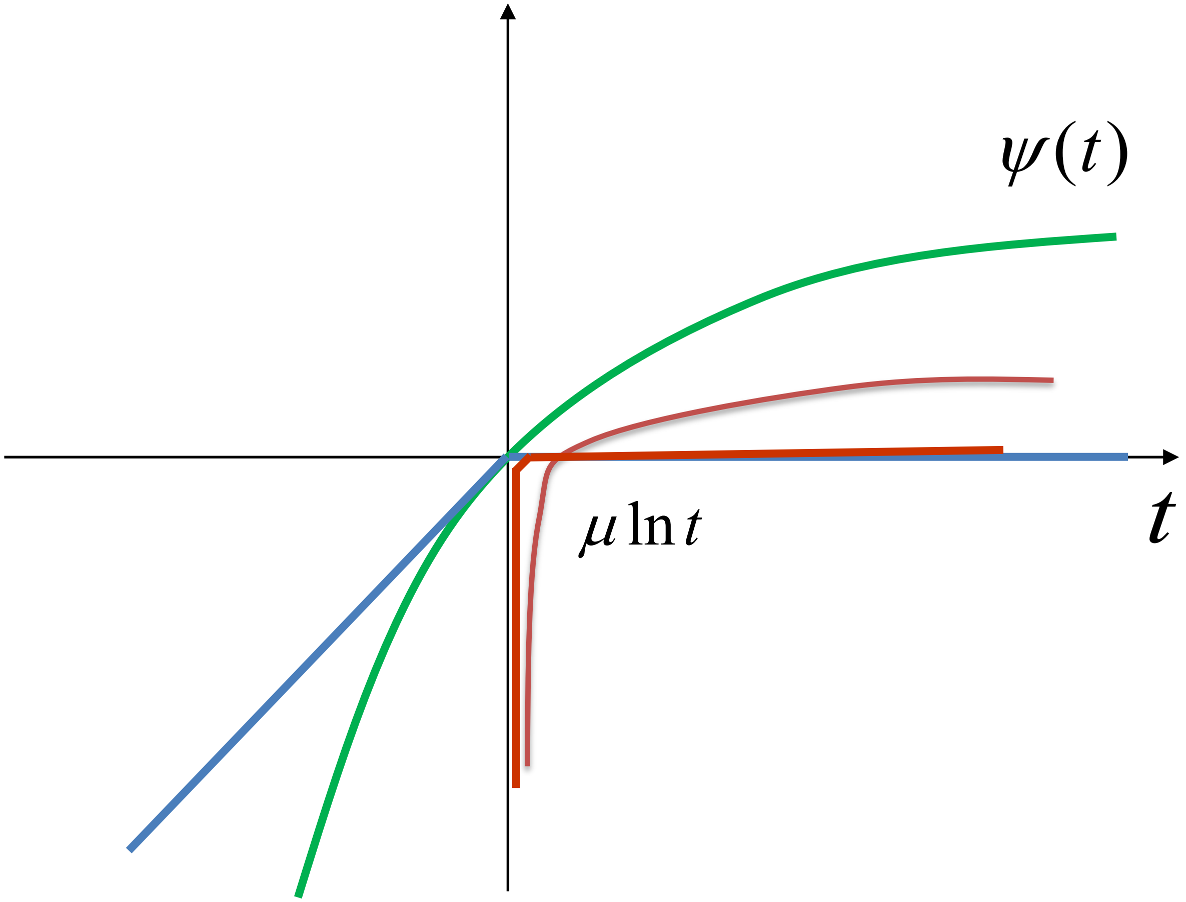
Figure 2: Operator P_B : Projection of $x \in \mathbb{R}^m$ onto the set B

Projection vs Rescaling = nonsmooth vs smooth transformations



To project or to rescale?

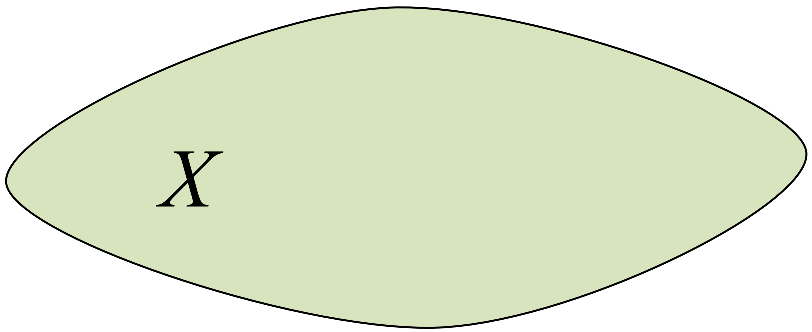
Projection vs Rescaling = nonsmooth vs smooth transformations



To project or to rescale?

Constrained convex quadratic optimization problem with linear constraints

$$f : \Re^n \to \Re^1$$



$$\min f(x), \;\; x \in X = I \cap E$$

Inequality constraints

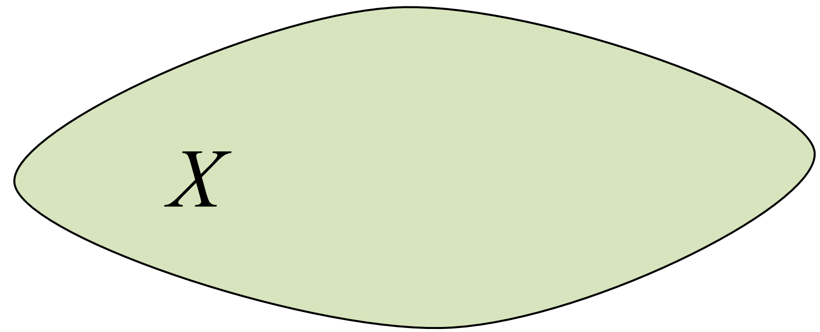
$$I = \{x \in \Re^n : c_i(x) \geq 0, \; i = 1,\ldots,m\}$$

Equality constraints

$$E = \{x \in \Re^n : g_i(x) = 0, \; i = 1,\ldots,p\}$$

Constrained convex quadratic optimization problem with linear constraints

$$f : \Re^n \rightarrow \Re^1$$



$$\min f(x), \;\; x \in X = I \cap E$$

Inequality constraints

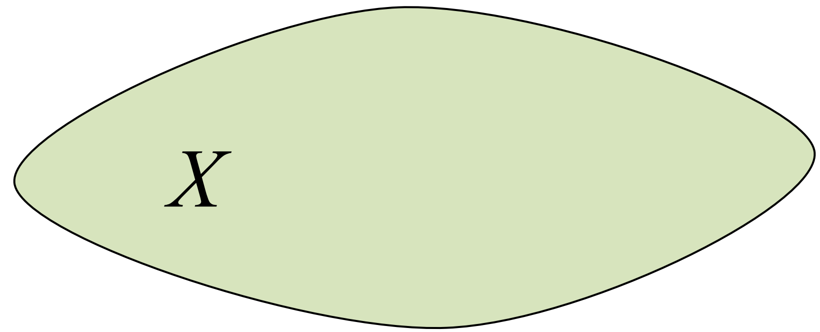
$$I = \{ x \in \Re^n : c_i(x) \geq 0, \; i = 1, \ldots, m \}$$

Equality constraints

$$E = \{ x \in \Re^n : g_i(x) = 0, \; i = 1, \ldots, p \}$$

Constrained convex quadratic optimization problem with linear constraints

$$f : \Re^n \rightarrow \Re^1$$



$$\min f(x), \;\; x \in X = I \cap E$$

Inequality constraints

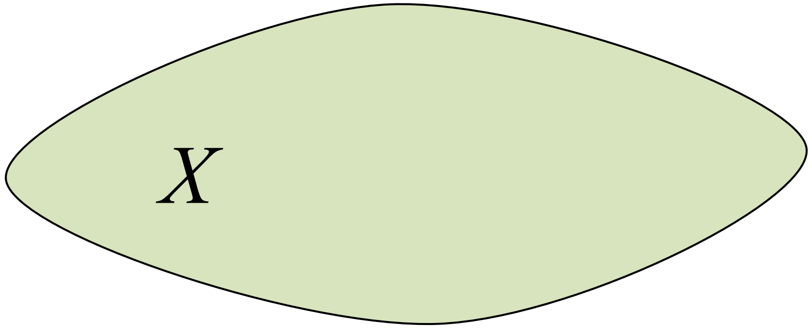
$$I = \{x \in \Re^n, s \in \Re^m : c_i(x) - s_i = 0, s_i \geq 0, i = 1, \ldots, m\}$$

Equality constraints

$$E = \left\{x \in \Re^n : g_i(x) = 0, \; i = 1, \ldots, p\right\}$$

Constrained convex quadratic optimization problem with linear constraints

$$f : \Re^n \to \Re^1$$



$$\min f(x), \;\; x \in X = I \cap E$$

Inequality constraints

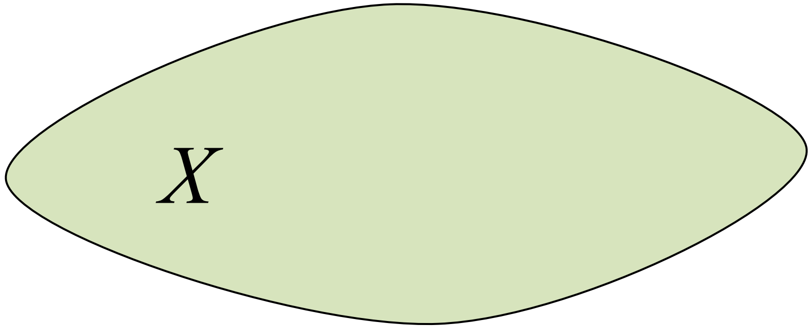
$$I = \{s \in \Re^m : s_i \geq 0, i = 1, \dots, m\}$$

Equality constraints

$$E = \{x \in \Re^n, s \in \Re^m : g_i(x, s) = 0, i = 1, \dots, \bar{p}\}$$

Constrained convex quadratic optimization problem with linear constraints

$$f : \Re^n \to \Re^1$$



$$\min f(x), \;\; x \in X = I \cap E$$

Inequality constraints

$$B = \{x \in \Re^n : l_i \le x_i \le u_i, i = 1, \dots, n\}$$

Equality constraints

$$E = \{x \in \Re^n : g_i(x) = 0, i = 1, \dots, p\}$$

Augmented Lagrangian Method (Hestenes, Powell, 1969)

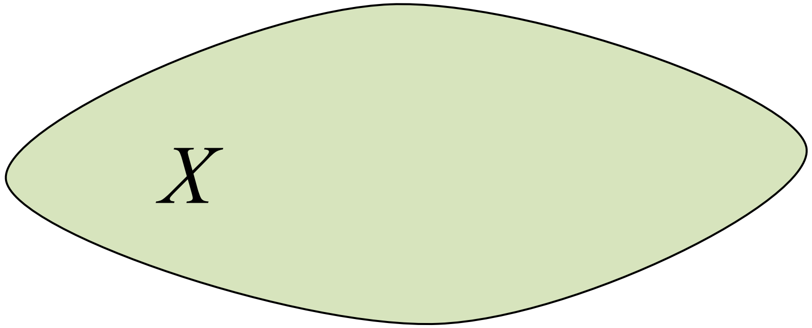
$$\mathcal{L}_k(x,\lambda) = f(x) - \lambda^T g(x) + \frac{k}{2} g(x)^T g(x)$$

$$\hat{x} \approx \hat{x}(\lambda) = \underset{x \in B}{\operatorname{argmin}}\ \mathcal{L}_k(x,\lambda)$$

$$\hat{\lambda} = \lambda - k g(\hat{x})$$

Constrained convex quadratic optimization problem with linear constraints

$$f : \Re^n \to \Re^1$$



$$\min f(x), \;\; x \in X = I \cap E$$

Inequality constraints

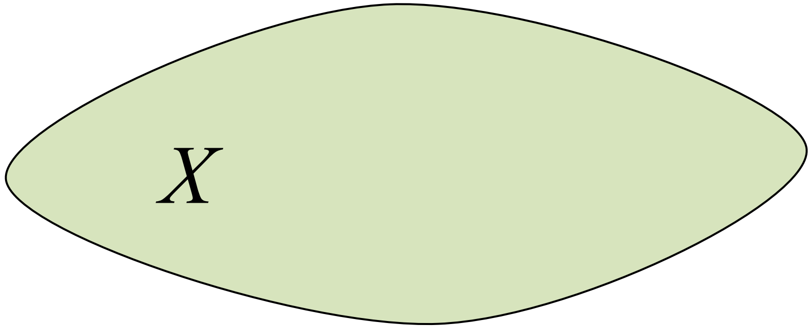
$$I = \{ x \in \Re^n : c_i(x) \geq 0, \; i = 1,\ldots,m \}$$

Equality constraints

$$E = \{ x \in \Re^n : g_i(x) = 0, \; i = 1,\ldots,p \}$$

Constrained convex quadratic optimization problem with linear constraints

$$f : \Re^n \rightarrow \Re^1$$



$$\min f(x), \;\; x \in X = I \cap E$$

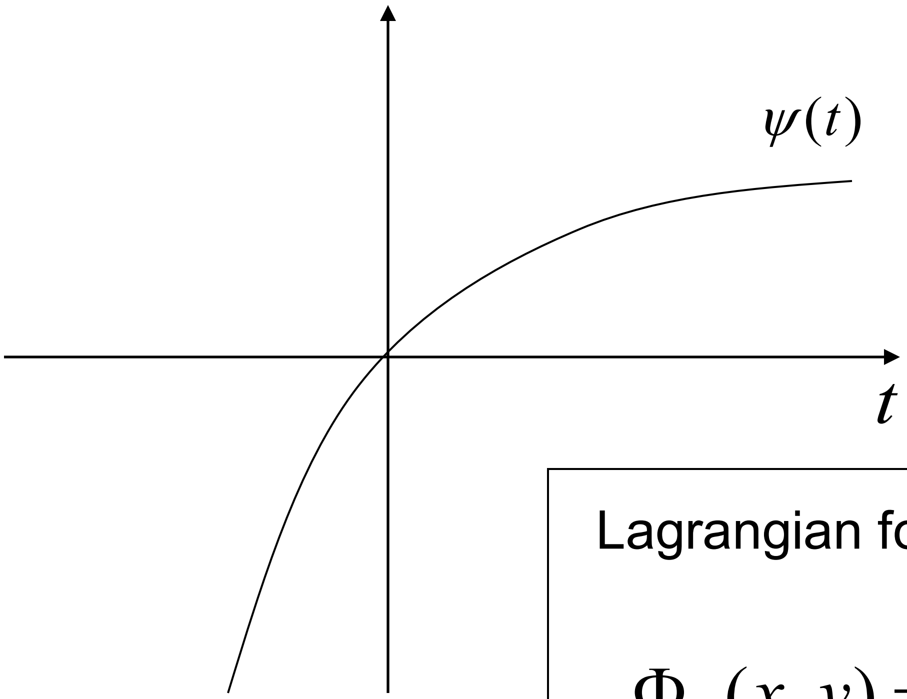
Inequality constraints

$$B = \{x \in \Re^n : l_i \leq x_i \leq u_i, i = 1, \ldots, n\}$$

Equality constraints

$$E = \{x \in \Re^n : g_i(x) = 0, i = 1, \ldots, p\}$$

Nonlinear Rescaling of constraints (R. Polyak)



Equivalent problem

$$\min f(x)$$

$$\text{s.t.} \quad k^{-1}\psi\left(kc_i(x)\right) \geq 0$$

k is the scaling parameter

Lagrangian for the equivalent problem

$$\Phi_k(x, y) = f(x) - k^{-1}\sum_{i=1}^{m} y_i \psi\left(kc_i(x)\right)$$

$\psi_1(t) = \log(1+t)$ - Modified Barrier, Polyak

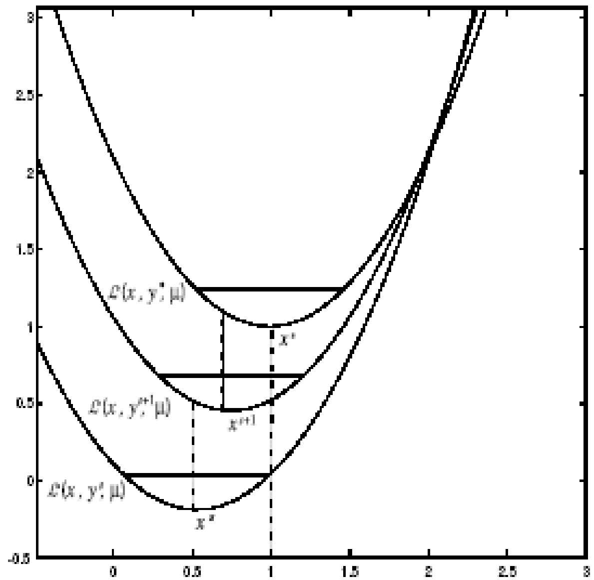
$\psi_2(t) = 1 - e^{-t}$ - Exponential, Kort and Bertsekas

Nonlinear rescaling method

$$\hat{x} = \arg\min_{x \in \Re^n} \Phi_k(x, y)$$

$$\hat{y} = \Psi'\big(kc(\hat{x})\big) Ye$$

k is fixed!!!



nonlinear rescaling

$$\Psi' = \mathrm{diag}\big(\psi'(kc_i(\hat{x}))\big)$$

$$Y = \mathrm{diag}(y_i)$$

Nonlinear Rescaling – augmented Lagrangian Method

$$\Phi_k(x,y,z) = f(x) - k^{-1}\sum_{i=1}^{m} y_i \psi\big(kc_i(x)\big) - z^T g(x) + \frac{k}{2}\|g(x)\|_2^2$$

$$\hat{x} = \arg\min_{x \in \Re^n} \Phi_k(x,y,z)$$

$$\hat{y} = \Psi'\big(kc(\hat{x})\big)Ye$$

$$\hat{z} = z - kg(\hat{x})$$

Proximal-point Nonlinear Rescaling – augmented Lagrangian Method

$$\Phi_{k,a}(x,y,z) = f(x) - k^{-1}\sum_{i=1}^{m} y_i \psi\left(kc_i(x)\right) - z^T g(x) + \frac{k}{2}\|g(x)\|_2^2$$

$$+ \frac{1}{2k}(x-a)^T(x-a)$$

$$\hat{x} = \arg\min_{x \in \Re^n} \Phi_{k,a}(x,y,z)$$

$$\hat{y} = \Psi'\left(kc(\hat{x})\right)Ye$$

$$\hat{z} = z - kg(\hat{x})$$

Can we say when we should project or rescale?

Maximal monotone operator

Definition.

Let H be a real Hilbert space with inner product $(\cdot,\cdot)$.
A multifunction $T: H \to H$ is a monotone operator if

$(z - z', w - w') \geq 0$ **whenever** $w \in T(z),\ w' \in T(z')$

The operator is maximal monotone if, in addition, the graph

$$G(T) = \{(z, w) \in H \times H \mid w \in T(z)\}$$

is not properly contained in the graph of any other monotone operator $T' : H \to H$.

A fundamental problem is to determine z such that $0 \in T(z)$.

The *proximal point algorithm* generates for any starting point z_0 a sequence $\{z_p\}$ in H by the approximate rule

$$z_{p+1} \approx P(z_p), \text{ where } P = (I + kT)^{-1}$$

Augmented Lagrangian Method

$$\mathcal{L}_k(x, \lambda) = f(x) - \lambda^T g(x) + \frac{k}{2} g(x)^T g(x)$$

$$\hat{x} \approx \hat{x}(\lambda) = \operatorname*{argmin}_{x \in B} \mathcal{L}_k(x, \lambda)$$

$$\hat{\lambda} = \lambda - k g(\hat{x})$$

Adding a Proximal term

$$\mathcal{L}_k(x, a, \lambda) = f(x) - \lambda^T g(x) + \frac{k}{2} g(x)^T g(x) + \frac{1}{2k}(x - a)^T (x - a)$$

Augmented Lagrangian method is a proximal-point algorithm

$$I_B(x) = \begin{cases} 0, & \text{if} \quad x \in B, \\ +\infty, & \text{if} \quad x \notin B. \end{cases}$$

$$\hat{f}(x) = f(x) + I_B(x) = \begin{cases} f(x), & \text{if} \quad x \in B, \\ +\infty, & \text{if} \quad x \notin B. \end{cases}$$

$$\hat{L}(x,\lambda) = \hat{f}(x) - \lambda^T g(x) = L(x,\lambda) + I_B(x)$$

$$\partial_x \hat{L}(x,\lambda) = \partial_x \hat{L}(x,\lambda)_1 \times \partial_x \hat{L}(x,\lambda)_2 \times \cdots \times \partial_x \hat{L}(x,\lambda)_n$$

$$\partial_x \hat{L}(x,\lambda)_i = \begin{cases} \nabla_x L(x,\lambda)_i, & \text{if} \quad l_i < x_i < u_i \\ (-\infty, \nabla_x L(x,\lambda)_i], & \text{if} \quad x_i \le l_i, \\ [\nabla_x L(x,\lambda)_i, +\infty), & \text{if} \quad x_i \ge u_i. \end{cases}$$

$$z = (x,\lambda)$$

$$T(z) = (\partial_x \hat{L}(x,\lambda), -\partial_\lambda \hat{L}(x,\lambda)) = (\partial_x \hat{L}(x,\lambda), g(x))$$

Augmented Lagrangian Method

1. Set $x \in B$, $\lambda = 0$, $rec = accur(x, x, \lambda)$.
   Select $k > 0$, $\epsilon > 0$, $0 < \theta < 1$, $\delta \geq 1$.
2. Find $\hat{x} \approx \underset{v \in B}{\operatorname{argmin}} \mathcal{L}_k(v, x, \lambda)$ with FPGM such that $\mu(\hat{x}, x, \lambda) \leq \epsilon / k$
3. Set $rec := accur(\hat{x}, x, \lambda)$
4. Find $\hat{\lambda} = \lambda - k g(\hat{x})$.
5. Set $x := \hat{x}$, $\lambda := \hat{\lambda}$, $\epsilon := \theta\epsilon$, $k := \delta k$.
6. If $rec > RequiredAccuracy$ then Goto 2.
7. Stop.

FIGURE 1. Boxed Augmented Lagrangian FPG Method

$$\mu_i(x, a, \lambda) = \begin{cases} |(\nabla_x \mathcal{L}(x, a, \lambda))_i|, & \text{if } l_i < x_i < u_i, \\ \max\{0, -(\nabla_x \mathcal{L}(x, a, \lambda))_i\}, & \text{if } x_i = l_i, \\ \max\{0, (\nabla_x \mathcal{L}(x, a, \lambda))_i\}, & \text{if } x_i = u_i, \end{cases}$$

$$\mu(x, a, \lambda) = \max_{1 \leq i \leq m} \mu_i(x, a, \lambda)$$

$$accur(x, a, \lambda) =: \max\{\mu(x, x, \lambda), \|g(x)\|\}$$

$$accur(x, a, \lambda) = 0 \Leftrightarrow \boxed{x = x^*, \lambda = \lambda^*, a = x^*}$$

Augmented Lagrangian method is a proximal-point algorithm

Lemma 5.1. *The augmented Lagrangian method is equivalent to the following proximal point method*

(5.5) $$\text{Find } z_{p+1} : \text{dist}(0, S_p(z_{p+1})) \le \frac{\epsilon_p}{k}$$

where $z_p = (x_p, \lambda_p)$, $\sum \epsilon_p < \infty$,

$$S_p(z) = T(z) + k^{-1}(z - z_p),$$
$$T(z) = \partial_z \hat{L}(z) = (\partial_x \hat{L}(x, \lambda), -\partial_\lambda \hat{L}(x, \lambda)),$$

$z^* = (x^*, \lambda^*)$ is the solution $\Longleftrightarrow$ $0 \in \partial_x \hat{L}(x^*, \lambda^*), \quad g(x^*) = 0.$

$\Longleftrightarrow$ $0 \in T(z^*)$

Fast projected gradient method (Beck-Teboulle, Nesterov, Polyak)

1. Input (x, λ), $v := x$.
2. Set $\bar{v} = v$, $t = 1$. Select $L > 0$.
3. Set $\hat{v} = P_B(v - \frac{1}{L}\nabla_v \mathcal{L}_k(v, x, \lambda))$
4. Set $\bar{t} = 0.5(1 + \sqrt{1 + 4t^2})$
5. Set $v = \hat{v} + (\hat{v} - \bar{v})(t - 1)/\bar{t}$
6. Set $\bar{v} = \hat{v}$, $t = \bar{t}$
7. If $\mu(\hat{v}, x, \lambda) > RequiredAccuracy$, Goto 3.
8. Output $\hat{v}$.

FIGURE 3. Fast Projected Gradient Method

Lemma 5.2. *For the sequence $\{x_s\}$ generated by the FPGM in Figure 3 for a convex quadratic problem the following bound takes place*

$$\mathcal{L}_k(x_s, x, \lambda) - \mathcal{L}_k(x_{\min}(x, \lambda), x, \lambda) \leq \frac{2L\|x_0 - x_{\min}(x, \lambda)\|^2}{(s+1)^2} \tag{5.6}$$

where $L > 0$ is the Lipschitz constant mentioned earlier in the text and

$$x_{\min}(x, \lambda) = \operatorname*{argmin}_{v \in B} \mathcal{L}_k(v, x, \lambda).$$ (Follows from Polyak, 2015)

Convergence

Theorem 1. The sequence $\{(x_p, \lambda_p)\}$ generated by the AL-FPGM in Figure 1 has a unique cluster primal-dual pair (x^*, λ^*) that satisfies the first order optimality conditions

$$\langle \nabla_x L(x^*, \lambda^*), x - x^* \rangle \geq 0 \quad \forall x \in B$$

and

$$g(x^*) = 0.$$

i.e. $\{(x_p, \lambda_p)\}$ converges to the optimal solution of problem (1) in the weak sense.

The proof is based on showing that the method satisfies conditions of Theorem 1 in Rockafellar (1976) for a general proximal-point method
(details can be found in Pure and Applied Functional Analysis, 3(3), 417-428, 2018)

Proximal-point nonlinear rescaling (PPNR) method

$$\mathcal{L} : \mathbb{R}^n \times \mathbb{R}^m_{++} \times \mathbb{R}^m_{++} \to \mathbb{R} : \mathcal{L}(x, y, \boldsymbol{k}) = f(x) - \sum_{i=1}^{m} y_i k_i^{-1} \psi(k_i c_i(x))$$

$x^0 \in \mathbb{R}^n$ and $\mathbb{R}^m_{++}$ are initial primal and dual approximations, $\{k_s > 0\}$ is the nondecreasing sequence.

The PPNR method generates three sequences $\{\boldsymbol{k}^s\} \subset \mathbb{R}^m_{++}$, $\{x^s\} \subset \mathbb{R}^n$, $\{y^s\} \subset \mathbb{R}^m_{++}$, by formulas

$$\boldsymbol{k}^s = (k_i^s = k_s (y_i^s)^{-1}, \quad i = 1, \ldots, m), \tag{5}$$

$$x^{s+1} = \arg\min\{\mathcal{L}(x, y^s, \boldsymbol{k}^s) + \frac{1}{2k_s}\|x - x^s\|^2 \mid x \in \mathbb{R}^n\} \tag{6}$$

$$y^{s+1} = (y_i^{s+1} = y_i^s \psi'(k_i^s c_i(x^{s+1})), \quad i = 1, \ldots, m.) \tag{7}$$

Proximal-point nonlinear rescaling (PPNR) method

Lemma 4.1. *One step of the PPNR method (5)–(7) is equivalent to finding a saddle point* (x^{s+1}, y^{s+1}) *of the following function*

$$M(x, y, x^s, y^s) = L(x, y) + \frac{1}{2k_s}\|x - x^s\|^2 - \frac{1}{2k_s}\|y - y^s\|^2_{R^{-1}_{s+1}}$$

where $\|y\|^2_{R^{-1}_{s+1}} = y^T R^{-1}_{s+1} y$, *and* R^{-1}_{s+1} *is a diagonal matrix with positive entries,*

$$(x^{s+1}, y^{s+1}) : \max_{y \in \mathbb{R}^m_+} \min_{x \in \mathbb{R}^n} M(x, y, x^s, y^s) = \min_{x \in \mathbb{R}^n} \max_{y \in \mathbb{R}^m_+} M(x, y, x^s, y^s)$$

where $R_{s+1} = \mathrm{diag}\,(r_i^{s+1})_{i=1}^m$, $r_i^{s+1} = -\left[\psi''(\theta_i^s k_i^s c_i(x^{s+1}))\right] > 0$ and $0 < \theta_i^s < 1$.

Rescaling of the primal constraint functions leads to rescaling of dual variables!

Rescaled in the dual space proximal-point method

$$T_R(z) = \begin{pmatrix} \nabla_x L(x,y) \\ -R\nabla_y L(x,y) \end{pmatrix}$$ is a monotone operator

$$P_R(z) = (I + kT_R)^{-1}$$

$z^0 = (x^0, y^0)$, $x^0 \in \mathbb{R}^n$, $y^0 \in \mathbb{R}^m_{++}$ $\quad z^{s+1} = P_{R_{s+1}}(z^s)$, $s = 0, 1, \ldots$

Proximal-point nonlinear rescaling (PPNR) method G., Polyak, 2011)

Let $\{\varepsilon_s > 0\}$ be a sequence such that $\sum_{s=0}^{\infty} \varepsilon_s < \infty$. Then the modified PPNR method generates the following three sequences $\{k^s\} \subset \mathbb{R}^m_{++}$, $\{x^s\} \subset \mathbb{R}^n$, $\{y^s\} \subset \mathbb{R}^m_{++}$,

$$k^s = (k_i^s = k_s (y_i^s)^{-1}, \quad i = 1, \ldots, m), \tag{8}$$

$$x^{s+1} : \|\nabla_x \mathcal{L}(x^{s+1}, y^s, k^s) + \frac{1}{k_s}(x^{s+1} - x^s)\| \le \frac{\varepsilon_s}{k_s} \tag{9}$$

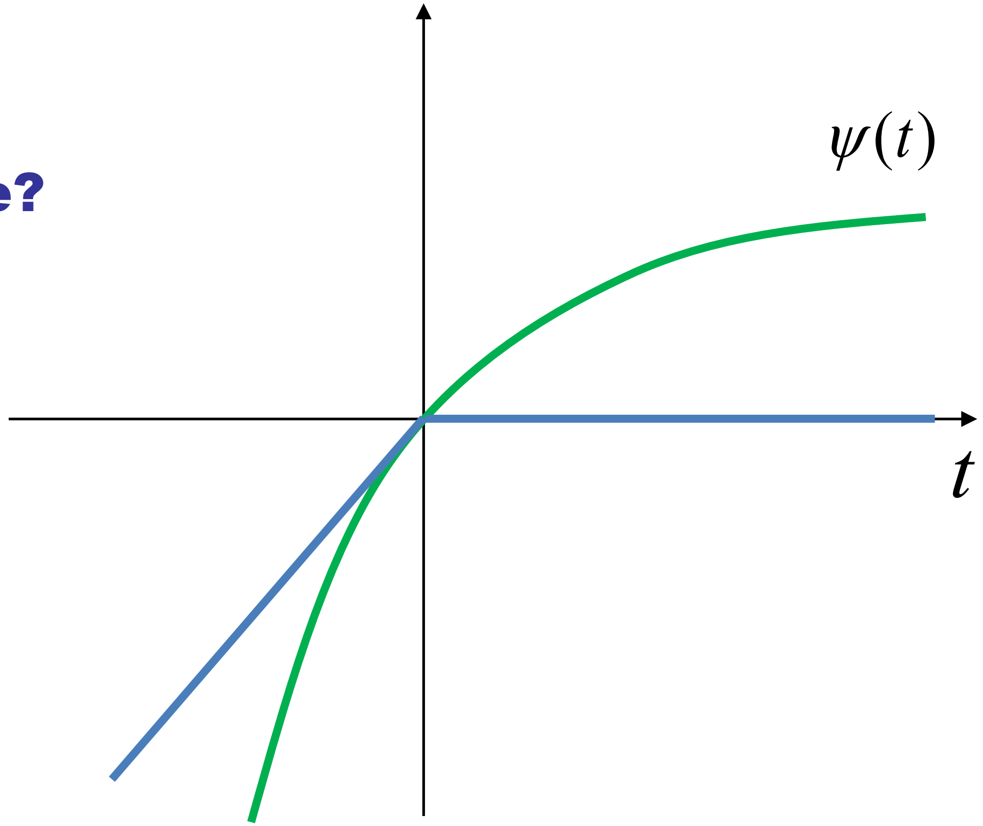
$$y^{s+1} = (y_i^{s+1} = y_i^s \psi'(k_i^s c_i(x^{s+1})), \quad i = 1, \ldots, m.) \tag{10}$$

Theorem 4.10. *If assumptions* **A** *and* **B** *are satisfied and* $\{k_s > 0\}$ *is a nondecreasing bounded sequence, then any limit point* $\bar{z} = (\bar{x}, \bar{y})$ *of the sequence* $\{z^s = (x^s, y^s)\}$ *generated by the modified PPNR method (8)–(10) is the primal-dual solution, i.e.* $(\bar{x}, \bar{y}) \in X^* \times Y^*$.

Published in Numerical Algebra, Control & Optimization, 1(2), 283-299, 2011

Projection vs Rescaling = nonsmooth vs smooth transformations

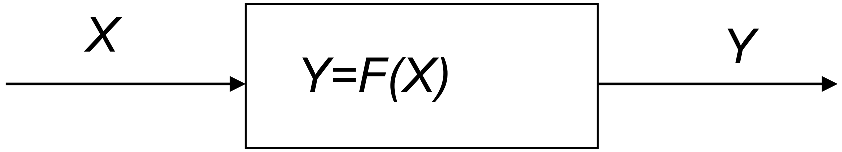
To project or to rescale?



For smaller smooth problems we can use Newton's method, while for larger nonsmooth problems, we can use fast projected gradient methods.

But where is the switching point from smaller to larger problems?

Numerical experiments: Support Vector Machines (V. Vapnik)



$$Box = \{\alpha \in \mathbb{R}^m : 0 \le \alpha_i \le C,\ i = 1, \dots m\}$$

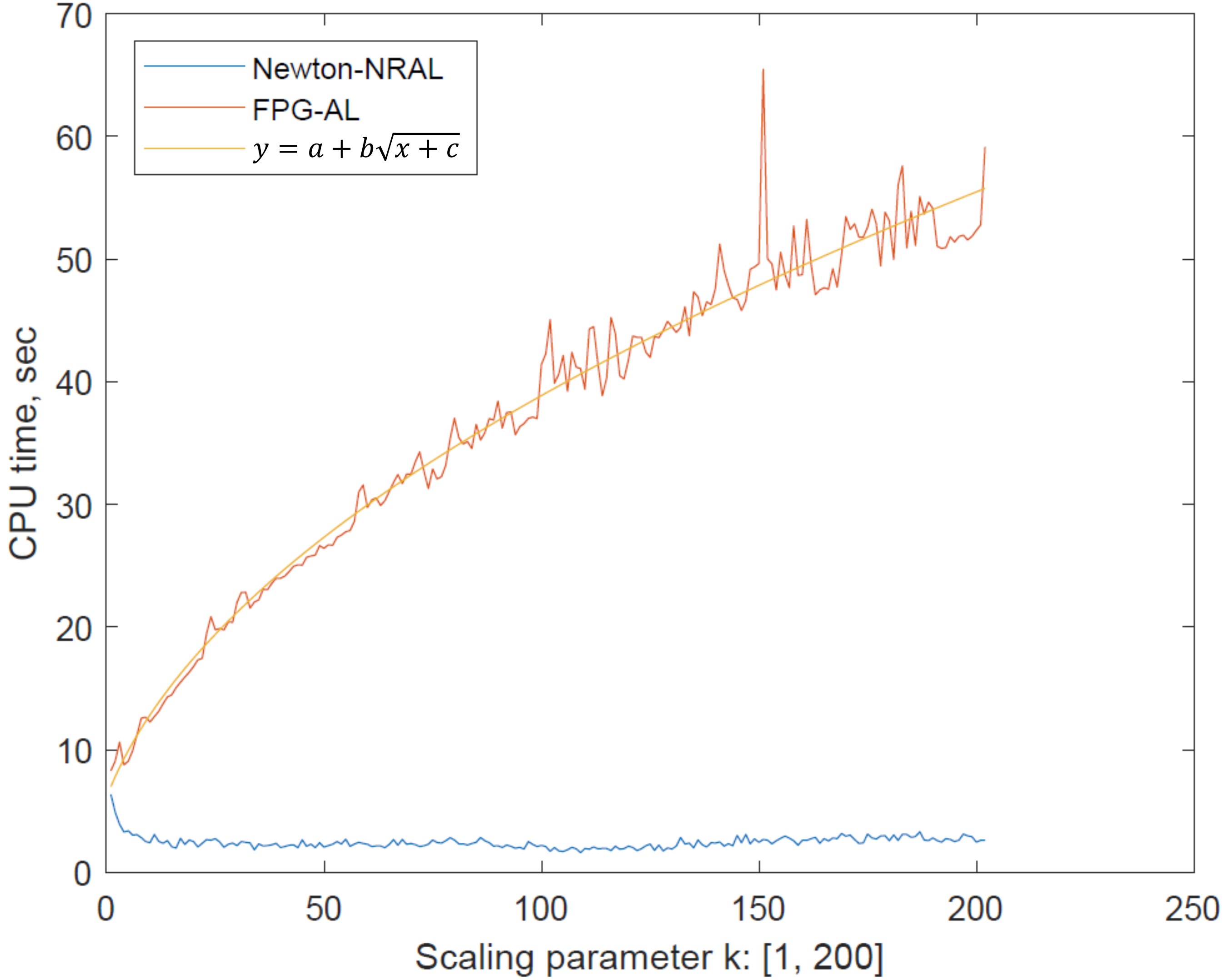
$$\underset{\alpha \in Box}{\text{minimize}}\ f(\alpha) = \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} y_i y_j \alpha_i \alpha_j K(x_i, x_j) - \sum_{i=1}^{m} \alpha_i$$

$$\text{subject to } g(\alpha) = \sum^{m} y_i \alpha_i = 0.$$

$$Q_{ij} = y_i y_j K(x_i, x_j)$$

$$K(x_i, x_j) = \exp\left(-\gamma \|x_i - x_j\|_2^2\right)$$

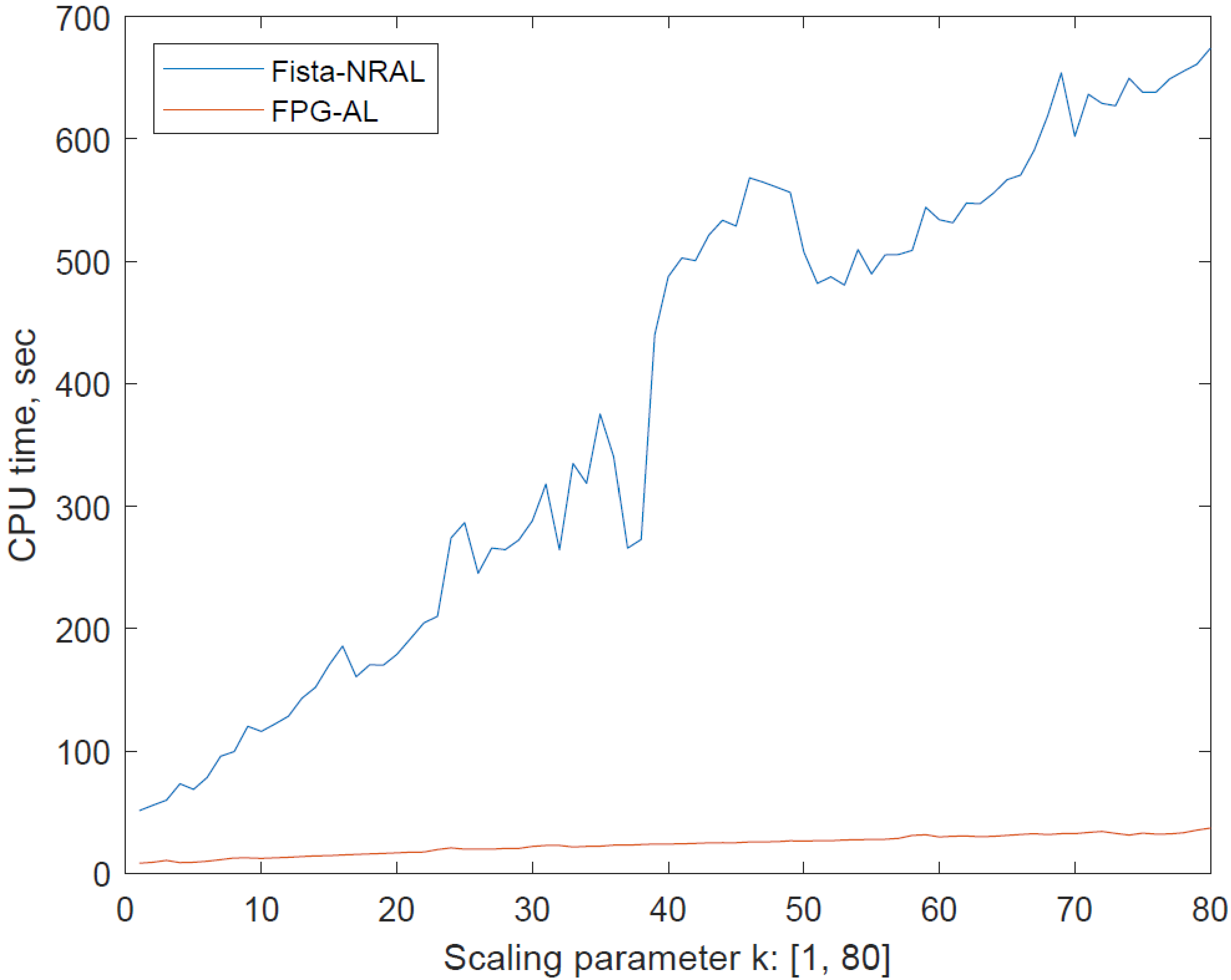
Increasing the scaling parameter k, n=1000 variables



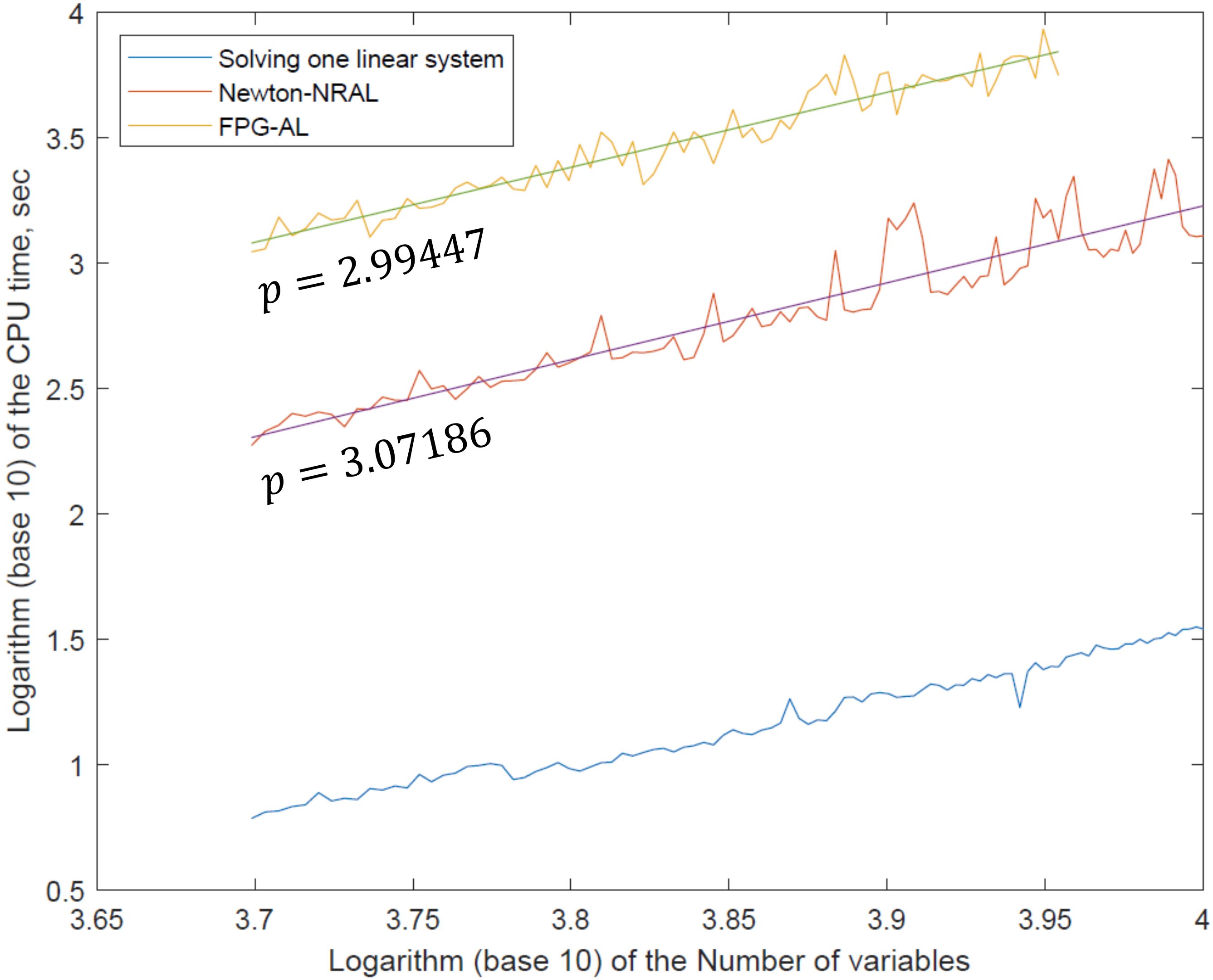
$$\nabla_{xx}\mathcal{L}_k(\cdot) = Q + ky^T y$$

$$\mathcal{L}_k(x_s, x, \lambda) - \mathcal{L}_k(x_{\min}(x,\lambda), x, \lambda) \leq \frac{2L\|x_0 - x_{\min}(x,\lambda)\|^2}{(s+1)^2}$$

Increasing the scaling parameter k, n=1000 variables



Increasing the number of variables: 5000,5050,...,10000



$$\mathcal{L}_k(x_s, x, \lambda) - \mathcal{L}_k(x_{\min}(x,\lambda), x, \lambda) \le \frac{2L\|x_0 - x_{\min}(x,\lambda)\|^2}{(s+1)^2}$$

Conclusions

- **Lagrange multipliers methods based on both nonlinear rescaling and projection lead to proximal-point methods that can be analyzed with the theory of maximal monotone operators**
- **Projection to the feasible set leads to a nonsmooth treatment of the optimization problem with rescaling not required**
- **Nonlinear rescaling method rescales the distance in the dual space for the implied proximal-point method, and, in turn, rescales the dual component of the image of the maximal monotone operator, but leads to a smooth treatment of the optimization problem**
- **While rescaling allows using Newton's method the projection, projection makes Newton's method useless**
- **If the first-order methods to be used, then projection could be a better choice than rescaling**
- **Numerical experiments suggest that the size of the problem may need to be very large for the quadratic programming problems, so Newton based nonlinear rescaling methods become less efficient than projection based first order methods.**
- **The investigation to be continued…**

Thank you!

Questions?