OSQP

An Operator Splitting Solver for Quadratic Programs

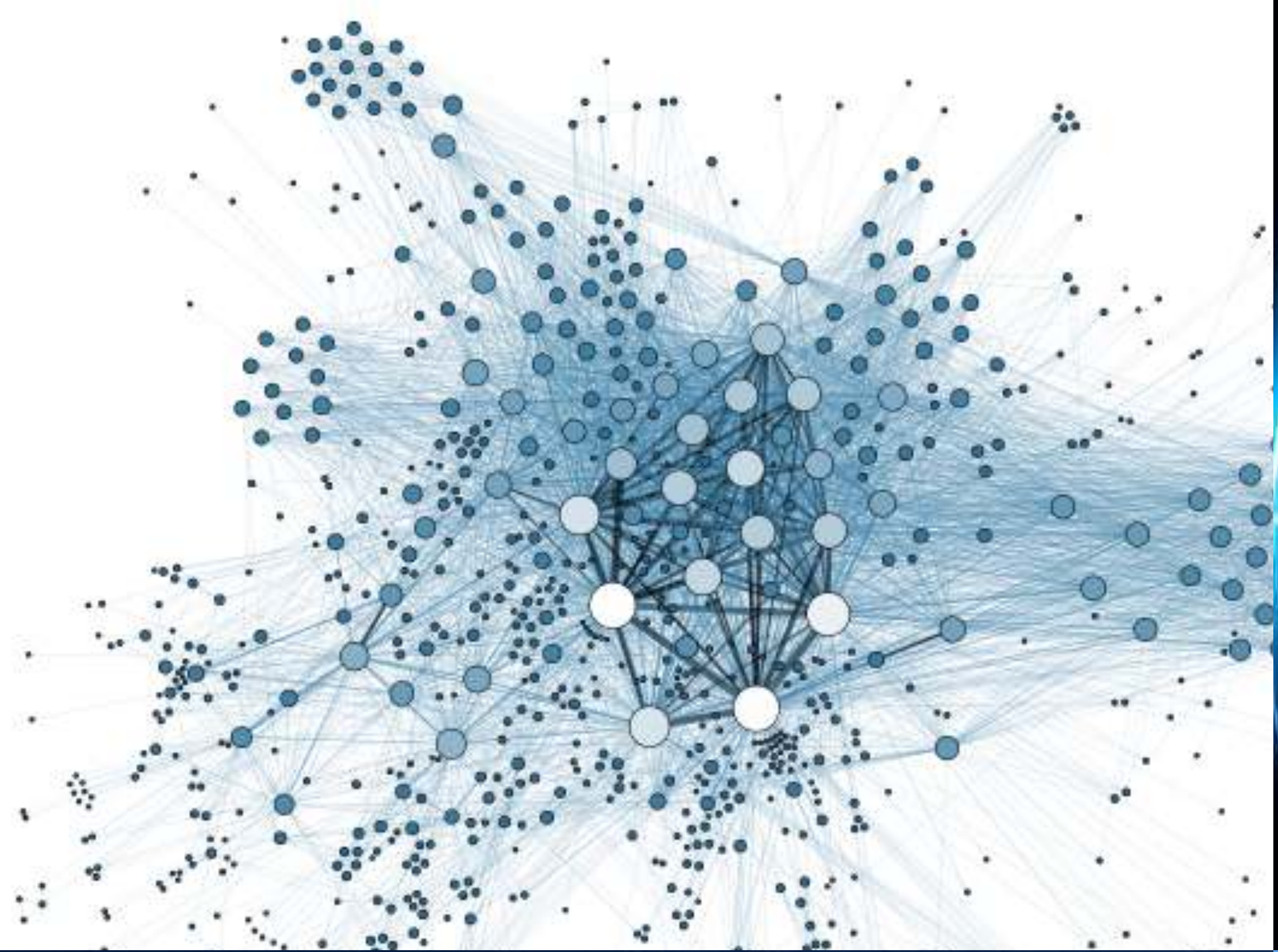
Bartolomeo Stellato

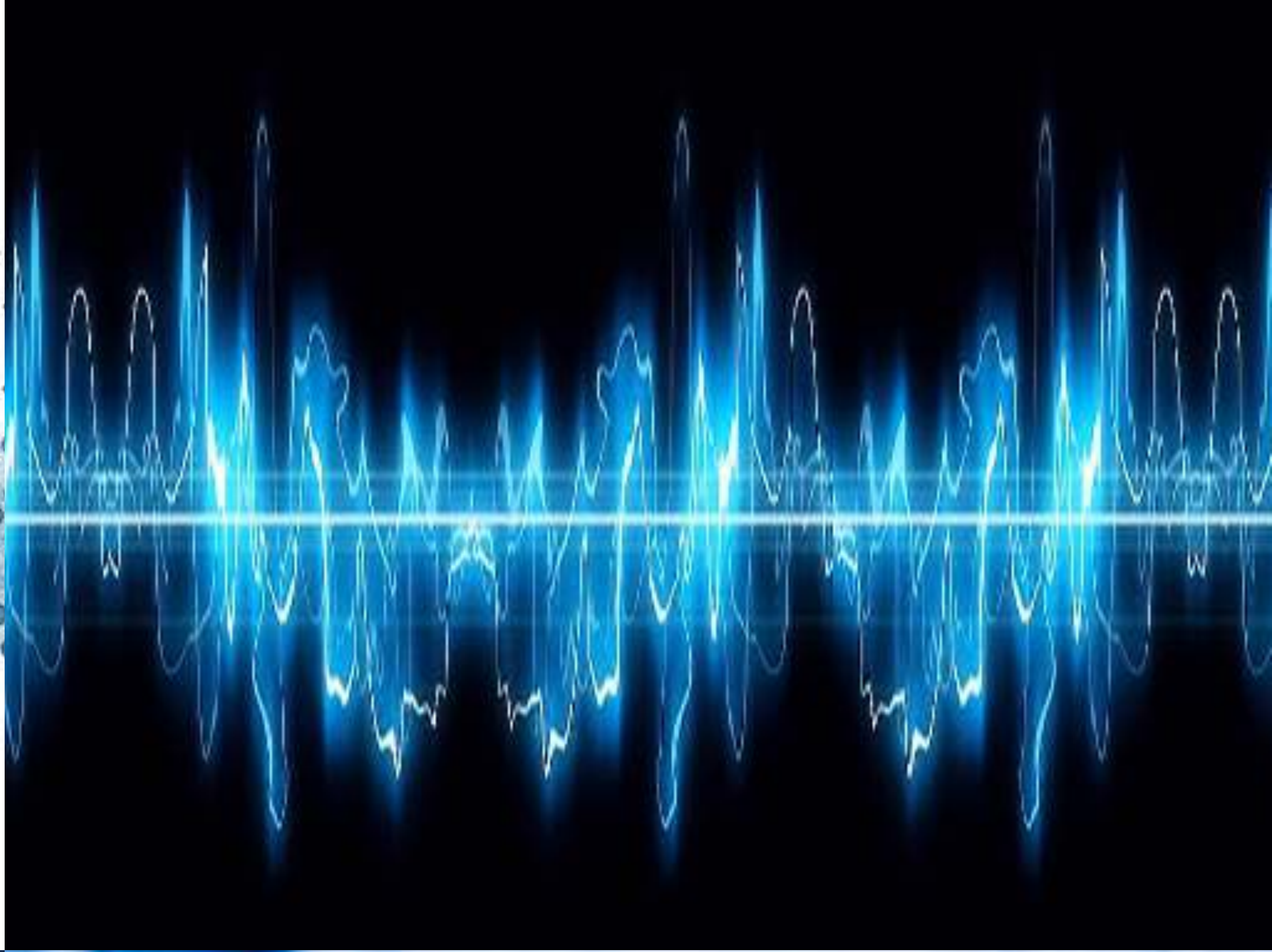
joint work with Goran Banjac,

Nicholas Moehle, Paul Goulart, Alberto Bemporad, Stephen Boyd

Numerical Analysis Group Internal Seminar, University of Oxford, Nov 7 2017

Why quadratic programming?





June
123.74
145
140
135
130
125
Tues.
142.78
Chg

First-order methods

Pros	Cons
Warm starting	Low accuracy solutions
Handle large-scale problems	Can’t detect infeasibility
Embeddable	Problem data dependent

General Purpose Solver

Based on first-order methods

Robust

Accurate

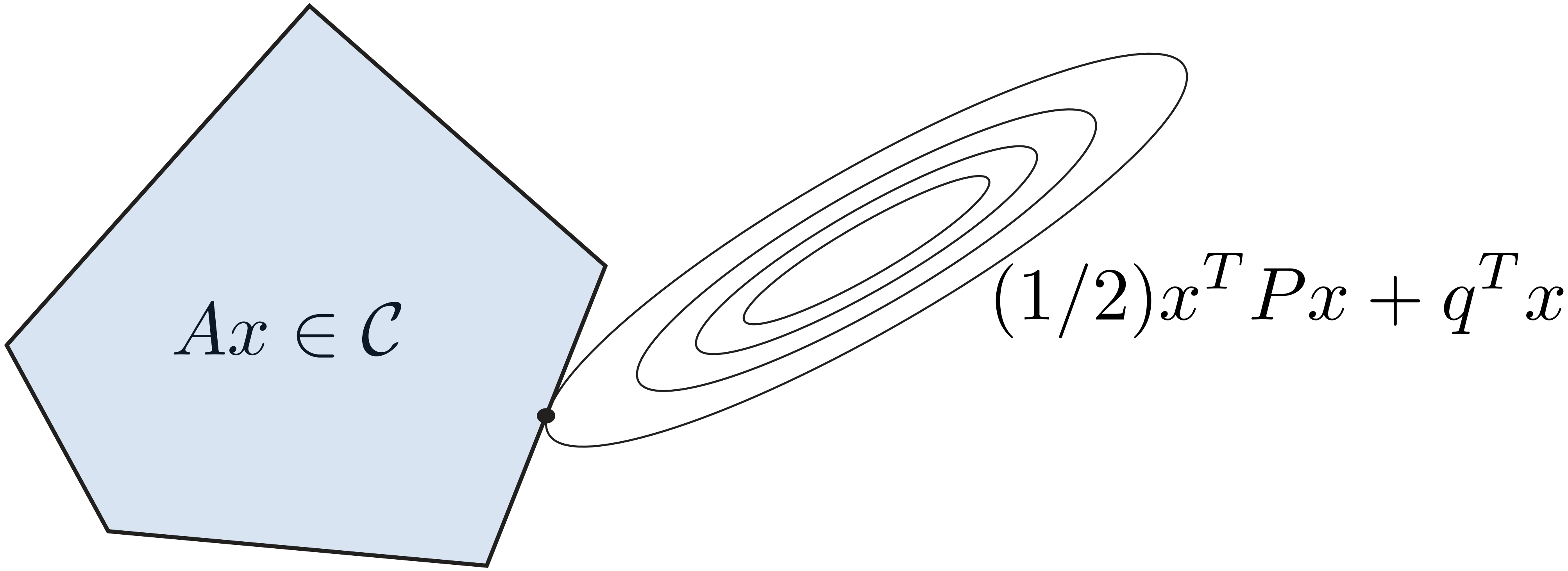
Detects Infeasibility

The OSQP Solver

The problem

$$\begin{array}{ll} \text{minimize} & (1/2)x^TPx + q^Tx \\ \text{subject to} & Ax \in \mathcal{C} \end{array}$$

Quadratic Program $\quad \mathcal{C} = [l, u]$



ADMM

$$\text{minimize} \quad f(x) + g(x) \quad \longrightarrow \quad \begin{array}{ll} \text{minimize} & f(\tilde{x}) + g(x) \\ \text{subject to} & \tilde{x} = x \end{array}$$

ADMM

minimize $\quad f(x)+g(x) \quad \longrightarrow \quad$ minimize $\quad f(\tilde{x})+g(x)$
subject to $\quad \tilde{x}=x$

1 $\quad \tilde{x}^{k+1} \leftarrow \underset{\tilde{x}}{\operatorname{argmin}} \left(f(\tilde{x}) + (\rho/2) \left\| \tilde{x} - x^k + \rho^{-1} y^k \right\|^2 \right)$

2 $\quad x^{k+1} \leftarrow \underset{x}{\operatorname{argmin}} \left(g(x) + (\rho/2) \left\| x - \tilde{x}^{k+1} - \rho^{-1} y^k \right\|^2 \right)$

3 $\quad y^{k+1} \leftarrow y^k + \rho \left(\tilde{x}^{k+1} - x^{k+1} \right)$

How to split the QP?

$$\begin{array}{ll} \text{minimize} & (1/2)x^T P x + q^T x \\ \text{subject to} & Ax = z \\ & z \in \mathcal{C} \end{array}$$

$$\begin{array}{ll} \text{minimize} & (1/2)\tilde{x}^T P \tilde{x} + q^T \tilde{x} + \mathcal{I}_{Ax=z}(\tilde{x}, \tilde{z}) + \mathcal{I}_{\mathcal{C}}(z) \\ \text{subject to} & (\tilde{x}, \tilde{z}) = (x, z) \end{array}$$

How to split the QP?

$$\begin{array}{ll} \begin{array}{l}\text{minimize} \\ \text{subject to}\end{array} & \left.\begin{array}{l}(1/2)x^TPx + q^Tx \\ Ax = z\end{array}\right\} f \\ & z \in \mathcal{C} \end{array}$$

$$\begin{array}{ll} \text{minimize} & \overbrace{(1/2)\tilde{x}^TP\tilde{x} + q^T\tilde{x} + \mathcal{I}_{Ax=z}(\tilde{x}, \tilde{z})}^{f} + \mathcal{I}_{\mathcal{C}}(z) \\ \text{subject to} & (\tilde{x}, \tilde{z}) = (x, z) \end{array}$$

How to split the QP?

$$\begin{array}{ll} \text{minimize} & \left.\begin{array}{l}(1/2)x^TPx + q^Tx \\ Ax = z\end{array}\right\} f \\ \text{subject to} & \\ & \left.z \in \mathcal{C}\right\} g \end{array}$$

$$\begin{array}{ll} \text{minimize} & \overbrace{(1/2)\tilde{x}^TP\tilde{x} + q^T\tilde{x} + \mathcal{I}_{Ax=z}(\tilde{x},\tilde{z})}^{f} + \overbrace{\mathcal{I}_{\mathcal{C}}(z)}^{g} \\ \text{subject to} & (\tilde{x},\tilde{z}) = (x,z) \end{array}$$

ADMM iterations

1 $$(x^{k+1}, \tilde{z}^{k+1}) \leftarrow \underset{(x,z):Ax=z}{\text{argmin}} \; (1/2)x^T P x + q^T x + (\sigma/2)\left\|x - x^k\right\|^2 + (\rho/2)\left\|z - z^k + \rho^{-1} y^k\right\|^2$$

2 $$z^{k+1} \leftarrow \Pi\left(\tilde{z}^{k+1} + \rho^{-1} y^k\right)$$

3 $$y^{k+1} \leftarrow y^k + \rho\left(\tilde{z}^{k+1} - z^{k+1}\right)$$

ADMM iterations

Inner QP

1 $$(x^{k+1}, \tilde{z}^{k+1}) \leftarrow \underset{(x,z):Ax=z}{\text{argmin}} \; (1/2)x^T P x + q^T x + (\sigma/2)\left\|x - x^k\right\|^2 + (\rho/2)\left\|z - z^k + \rho^{-1}y^k\right\|^2$$

2 $$z^{k+1} \leftarrow \Pi\left(\tilde{z}^{k+1} + \rho^{-1}y^k\right)$$

3 $$y^{k+1} \leftarrow y^k + \rho\left(\tilde{z}^{k+1} - z^{k+1}\right)$$

ADMM iterations

Inner QP

1 $(x^{k+1}, \tilde{z}^{k+1}) \leftarrow \underset{(x,z):Ax=z}{\text{argmin}} \; (1/2)x^T P x + q^T x + (\sigma/2) \left\| x - x^k \right\|^2 + (\rho/2) \left\| z - z^k + \rho^{-1} y^k \right\|^2$

2 $z^{k+1} \leftarrow \Pi \left(\tilde{z}^{k+1} + \rho^{-1} y^k \right)$

Projection onto $\mathcal{C}$

3 $y^{k+1} \leftarrow y^k + \rho \left(\tilde{z}^{k+1} - z^{k+1} \right)$

Solving the inner QP

$$\begin{array}{ll}\text{minimize} & (1/2)x^T P x + q^T x + (\sigma/2)\left\|x - x^k\right\|^2 + (\rho/2)\left\|z - z^k + \rho^{-1}y^k\right\|^2 \\ \text{subject to} & Ax = z\end{array}$$

Reduced KKT system

$$\begin{bmatrix} P + \sigma I & A^T \\ A & -\rho^{-1} I \end{bmatrix} \begin{bmatrix} x \\ \nu \end{bmatrix} = \begin{bmatrix} \sigma x^k - q \\ z^k - \rho^{-1} y^k \end{bmatrix}$$

$$\tilde{z}^{k+1} = z^k + \rho^{-1}(\nu - y^k)$$

Solving the inner QP

minimize $\quad (1/2)x^TPx + q^Tx + (\sigma/2)\left\|x - x^k\right\|^2 + (\rho/2)\left\|z - z^k + \rho^{-1}y^k\right\|^2$

subject to $\quad Ax = z$

Reduced KKT system

$$\begin{bmatrix} P + \sigma I & A^T \\ A & -\rho^{-1}I \end{bmatrix} \begin{bmatrix} x \\ \nu \end{bmatrix} = \begin{bmatrix} \sigma x^k - q \\ z^k - \rho^{-1}y^k \end{bmatrix}$$

Always solvable!

$$\tilde{z}^{k+1} = z^k + \rho^{-1}(\nu - y^k)$$

Solving the linear system

Direct Method

$$\begin{bmatrix} P + \sigma I & A^T \\ A & -\rho^{-1} I \end{bmatrix} \begin{bmatrix} x \\ \nu \end{bmatrix} = \begin{bmatrix} \sigma x^k - q \\ z^k - \rho^{-1} y^k \end{bmatrix}$$

Solving the linear system

Direct Method

Quasi-definite matrix

$$\begin{bmatrix} P + \sigma I & A^T \\ A & -\rho^{-1} I \end{bmatrix} \begin{bmatrix} x \\ \nu \end{bmatrix} = \begin{bmatrix} \sigma x^k - q \\ z^k - \rho^{-1} y^k \end{bmatrix}$$

Well defined LDL^T factorization

Solving the linear system

Direct Method

Quasi-definite matrix

$$\begin{bmatrix} P + \sigma I & A^T \\ A & -\rho^{-1} I \end{bmatrix} \begin{bmatrix} x \\ \nu \end{bmatrix} = \begin{bmatrix} \sigma x^k - q \\ z^k - \rho^{-1} y^k \end{bmatrix}$$

Well defined LDL^T factorization

Factorization caching

Solving the linear system

Indirect Method

$$\left(P + \sigma I + \rho A^T A\right) x = \sigma x^k - q + A^T(\rho z^k - y^k)$$

Solving the linear system

Indirect Method

Positive definite matrix

$$\underbrace{\left(P + \sigma I + \rho A^T A\right)} x = \sigma x^k - q + A^T(\rho z^k - y^k)$$

Conjugate gradient

Solving the linear system

Indirect Method

Positive definite matrix

$$\left(P + \sigma I + \rho A^T A\right) x = \sigma x^k - q + A^T(\rho z^k - y^k)$$

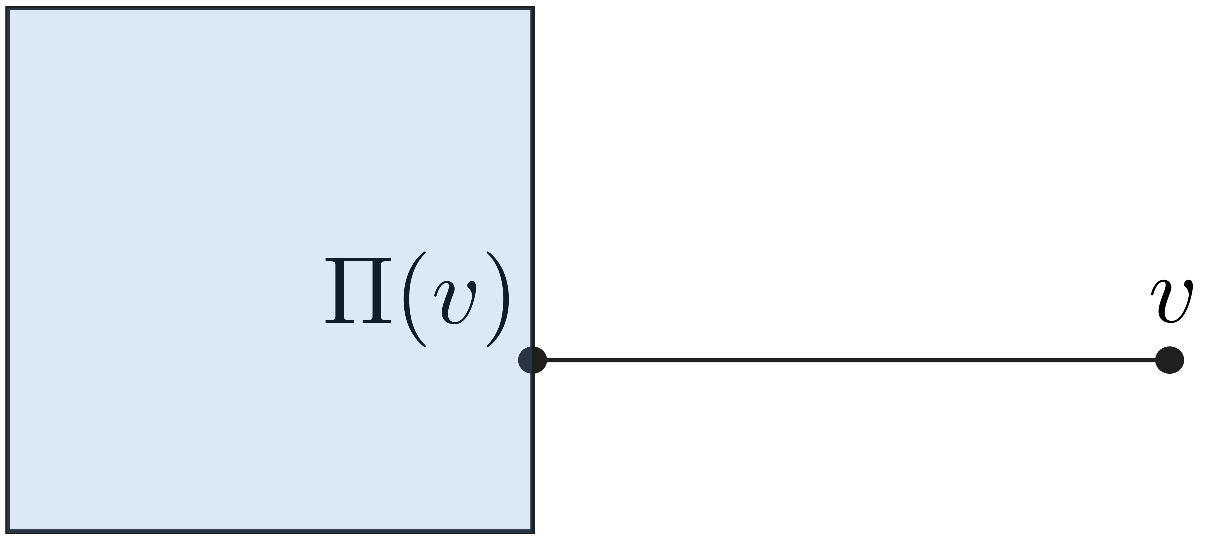
Conjugate gradient

Solve very large systems

Computing the projection

Box projection

$$\Pi(v) = \max(\min(v, u), l)$$



Final algorithm

Problem

$$\begin{array}{ll} \text{minimize} & (1/2)x^T P x + q^T x \\ \text{subject to} & l \leq Ax \leq u \end{array}$$

Algorithm

$$\boxed{1}\left\{\begin{array}{l} (x^{k+1}, \nu^{k+1}) \leftarrow \text{solve} \begin{bmatrix} P + \sigma I & A^T \\ A & -\rho^{-1} I \end{bmatrix} \begin{bmatrix} x^{k+1} \\ \nu^{k+1} \end{bmatrix} = \begin{bmatrix} \sigma x^k - q \\ z^k - \rho^{-1} y^k \end{bmatrix} \\ \tilde{z}^{k+1} \leftarrow z^k + \rho^{-1}(\nu^{k+1} - y^k) \end{array}\right.$$

$$\boxed{2}\left\{ z^{k+1} \leftarrow \Pi\left(\tilde{z}^{k+1} + \rho^{-1} y^k\right)\right.$$

$$\boxed{3}\left\{ y^{k+1} \leftarrow y^k + \rho\left(\tilde{z}^{k+1} - z^{k+1}\right)\right.$$

Final algorithm

Problem

$$\begin{array}{ll} \text{minimize} & (1/2)x^T P x + q^T x \\ \text{subject to} & l \leq Ax \leq u \end{array}$$

Algorithm

Linear system solve

$$(x^{k+1}, \nu^{k+1}) \leftarrow \text{solve} \begin{bmatrix} P + \sigma I & A^T \\ A & -\rho^{-1} I \end{bmatrix} \begin{bmatrix} x^{k+1} \\ \nu^{k+1} \end{bmatrix} = \begin{bmatrix} \sigma x^k - q \\ z^k - \rho^{-1} y^k \end{bmatrix}$$

$$\tilde{z}^{k+1} \leftarrow z^k + \rho^{-1}(\nu^{k+1} - y^k)$$

$$z^{k+1} \leftarrow \Pi\left(\tilde{z}^{k+1} + \rho^{-1} y^k\right)$$

$$y^{k+1} \leftarrow y^k + \rho\left(\tilde{z}^{k+1} - z^{k+1}\right)$$

Final algorithm

Problem

$$\begin{array}{ll} \text{minimize} & (1/2)x^T P x + q^T x \\ \text{subject to} & l \leq Ax \leq u \end{array}$$

Algorithm

Linear system solve

$$(x^{k+1}, \nu^{k+1}) \leftarrow \text{solve} \begin{bmatrix} P + \sigma I & A^T \\ A & -\rho^{-1} I \end{bmatrix} \begin{bmatrix} x^{k+1} \\ \nu^{k+1} \end{bmatrix} = \begin{bmatrix} \sigma x^k - q \\ z^k - \rho^{-1} y^k \end{bmatrix}$$

Easy operations

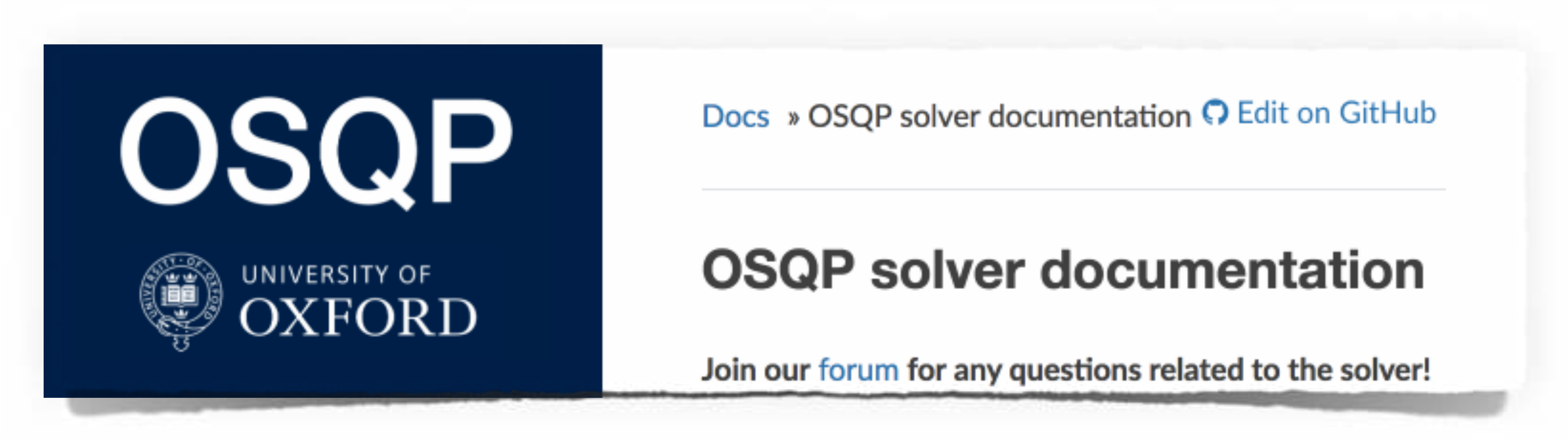
$$\tilde{z}^{k+1} \leftarrow z^k + \rho^{-1}(\nu^{k+1} - y^k)$$

$$z^{k+1} \leftarrow \Pi\left(\tilde{z}^{k+1} + \rho^{-1} y^k\right)$$

$$y^{k+1} \leftarrow y^k + \rho\left(\tilde{z}^{k+1} - z^{k+1}\right)$$

OSQP

`osqp.readthedocs.io`



Docs » OSQP solver documentation Edit on GitHub

OSQP solver documentation

Join our forum for any questions related to the solver!

Library free

Multiple interfaces

Embeddable

Interfaces

Languages

julia

python™

MATLAB®

Parsers

JuMP

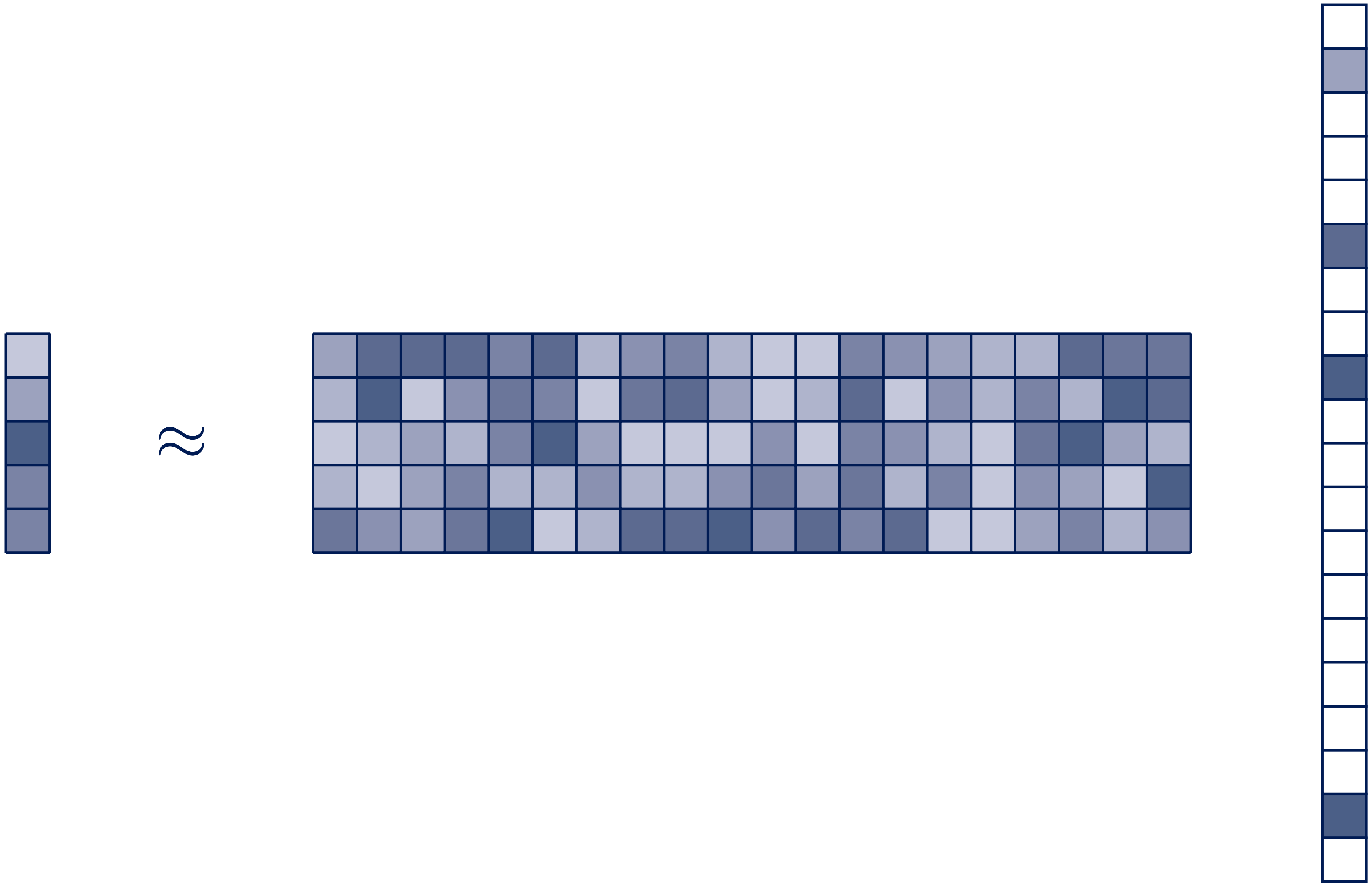
CVXPY

YALMIP

Numerical Examples

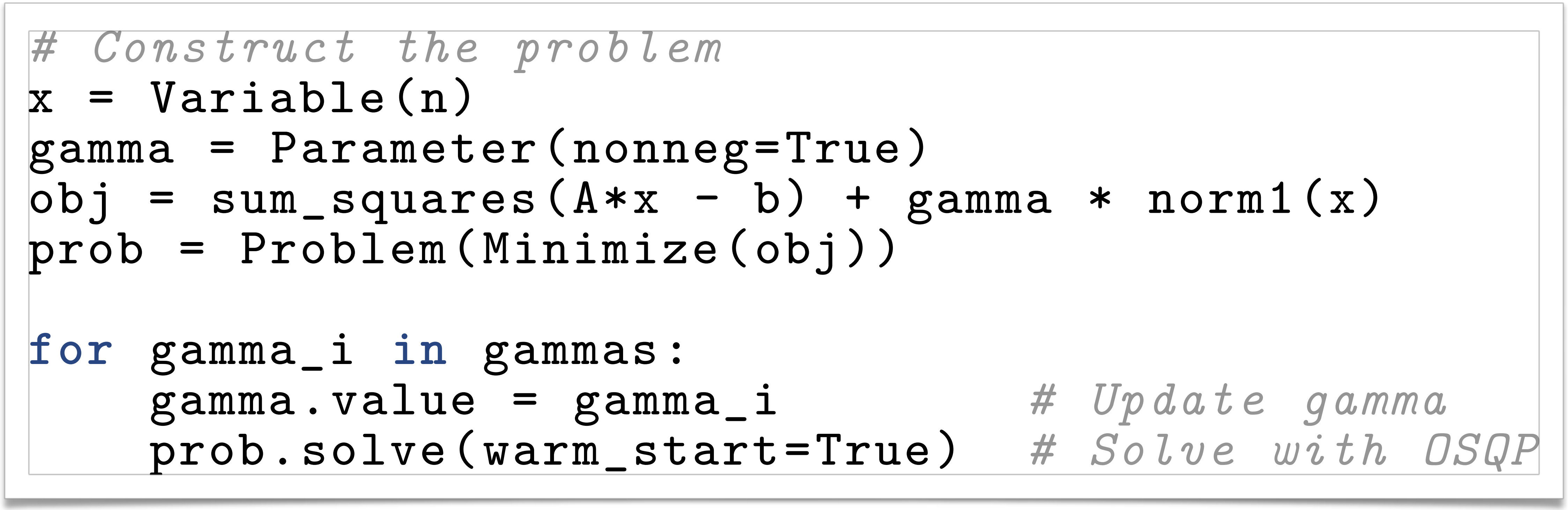
Lasso

minimize $\quad \|Ax - b\|_2^2 + \gamma \|x\|_1$



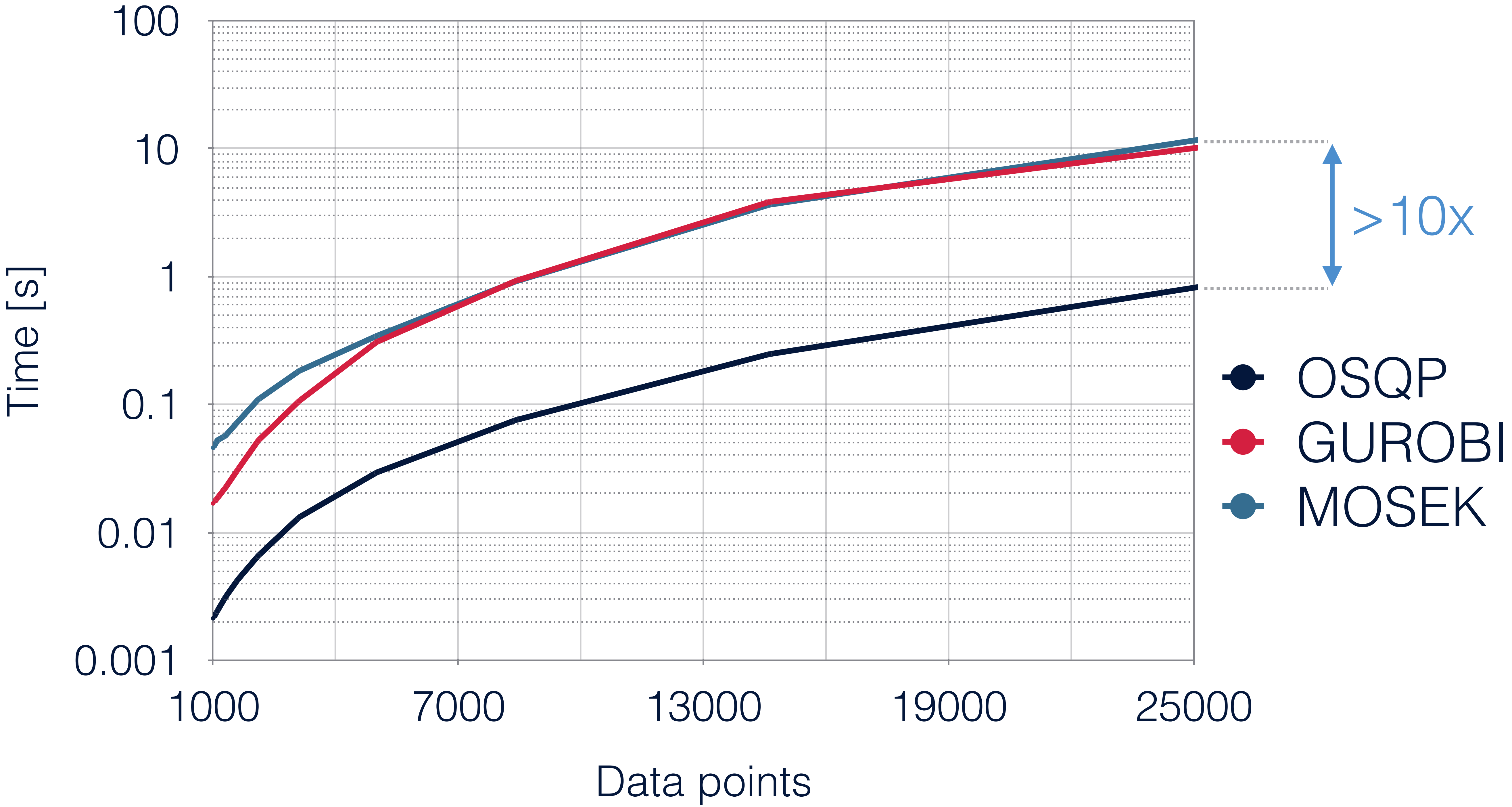
Lasso

$$\text{minimize} \quad \|Ax - b\|_2^2 + \gamma \|x\|_1$$



```python
# Construct the problem
x = Variable(n)
gamma = Parameter(nonneg=True)
obj = sum_squares(A*x - b) + gamma * norm1(x)
prob = Problem(Minimize(obj))

for gamma_i in gammas:
    gamma.value = gamma_i        # Update gamma
    prob.solve(warm_start=True)  # Solve with OSQP
```

Lasso timings



Benchmark Problems

Random

Equality Constrained

Portfolio

Lasso

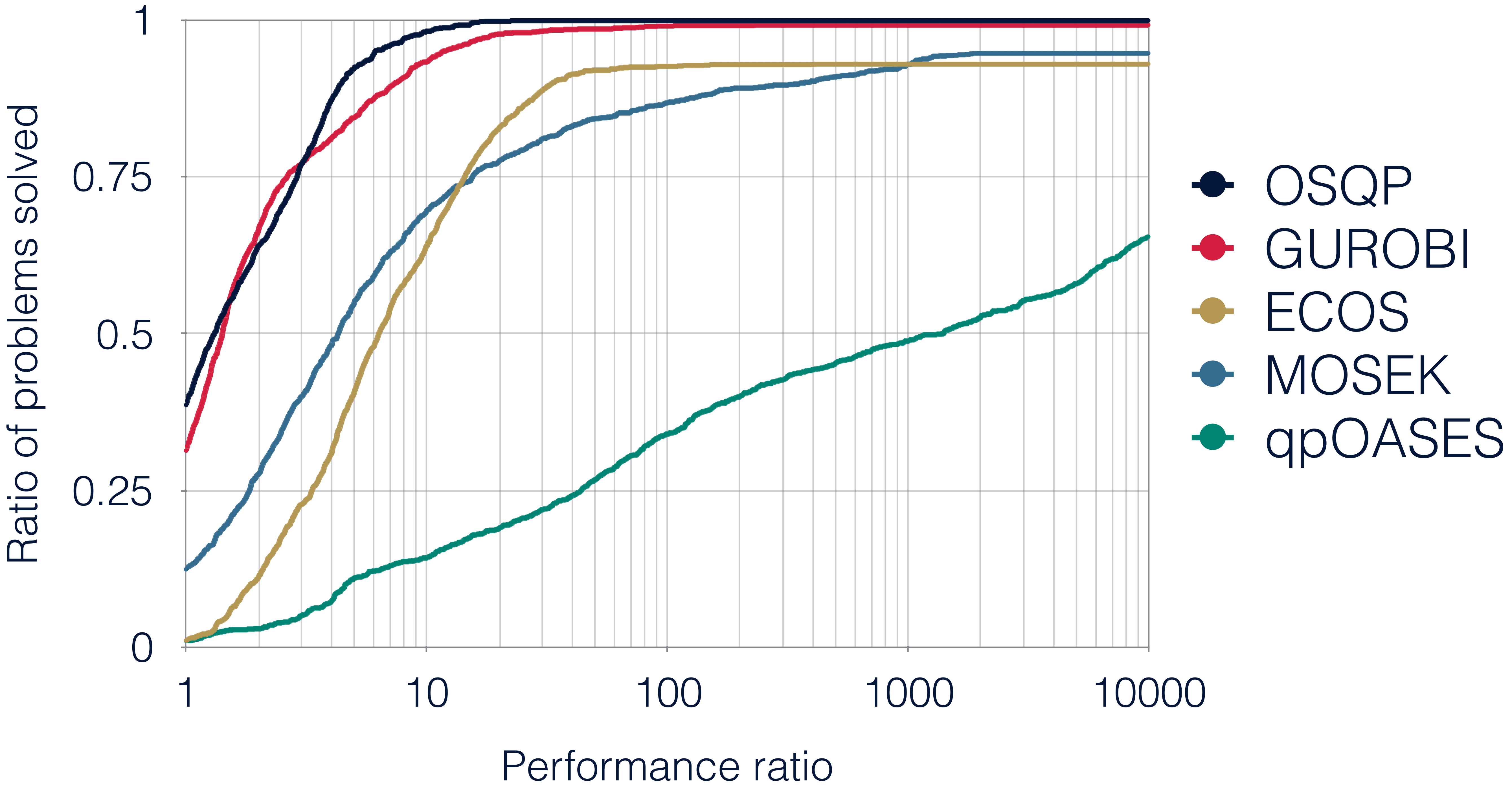
Huber Fitting

SVM

Control

1400 Problems

Performance Profiles



Infeasibility detection

What happens if the problem is infeasible?

$$Ax \notin \mathcal{C}$$

$$y^{k+1} \leftarrow y^k + \rho \left(\tilde{z}^{k+1} - z^{k+1} \right)$$

What happens if the problem is infeasible?

$$\boxed{Ax \notin \mathcal{C}}$$

$$y^{k+1} \leftarrow y^k + \rho \underbrace{\left(\overset{\overset{Ax}{\rotatebox{90}{$\in$}}}{\tilde{z}^{k+1}} - \overset{\overset{\mathcal{C}}{\rotatebox{90}{$\in$}}}{z^{k+1}}\right)}_{\neq 0}$$

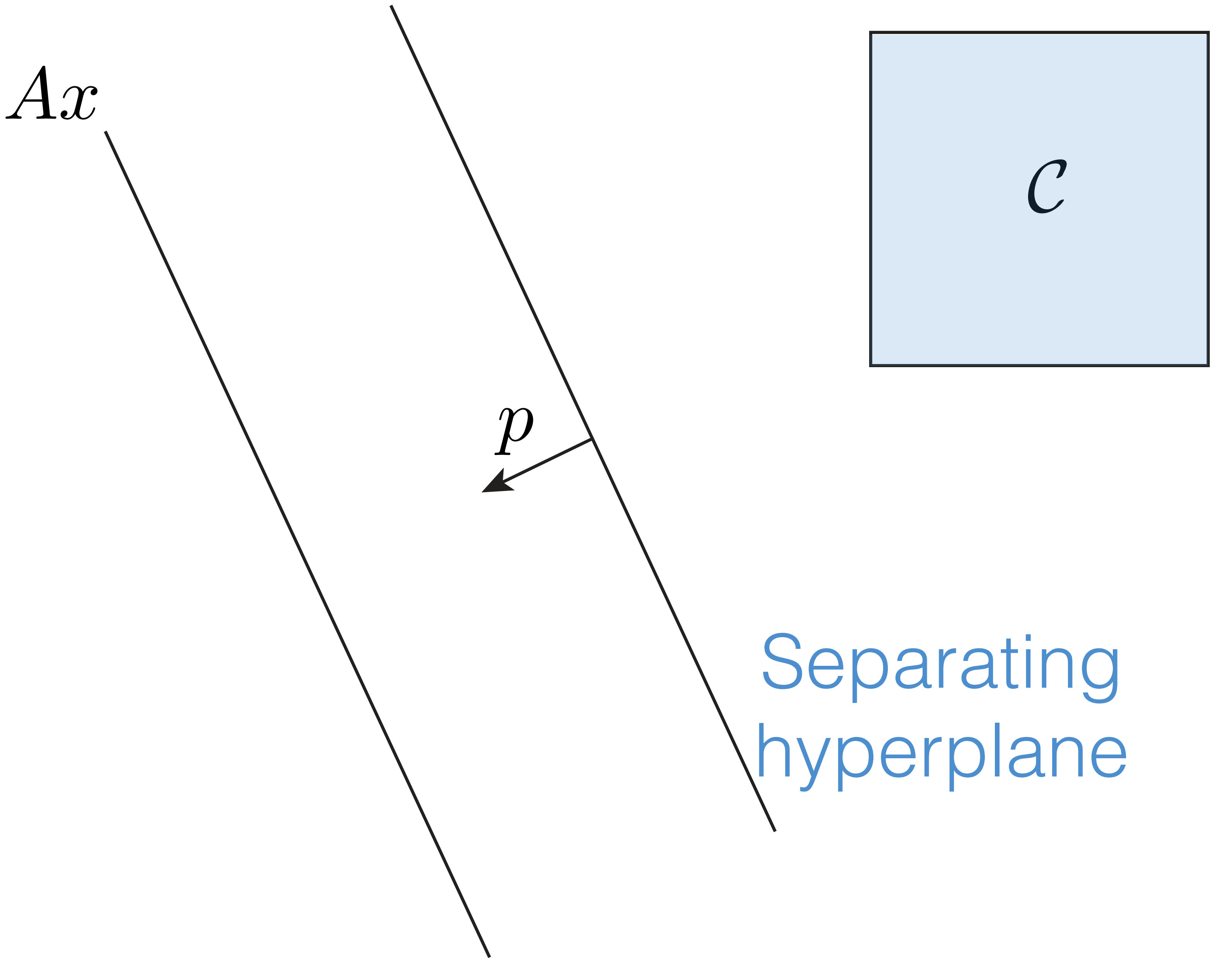
↓

y^k does not converge!

Farkas' Lemma

Primal infeasibility

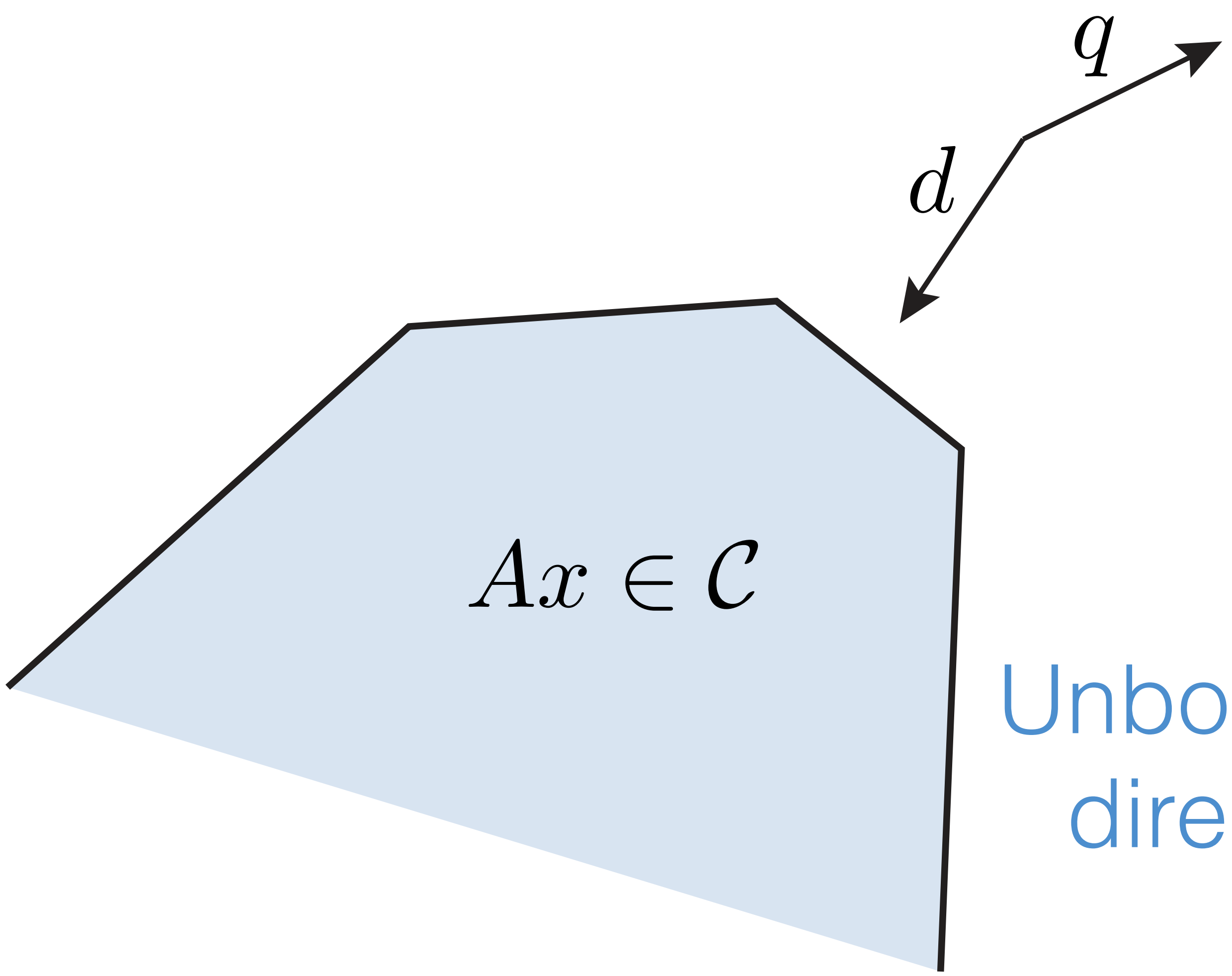
$$A^T p = 0 \qquad u^T p_+ + l^T p_- < 0$$



Farkas' Lemma

Dual infeasibility

$$Pd = 0 \qquad q^T d < 0 \qquad (Ad)_i \begin{cases} = 0 & l_i, u_i \neq \infty \\ \geq 0 & u_i = +\infty \\ \leq 0 & l_i = -\infty \end{cases}$$



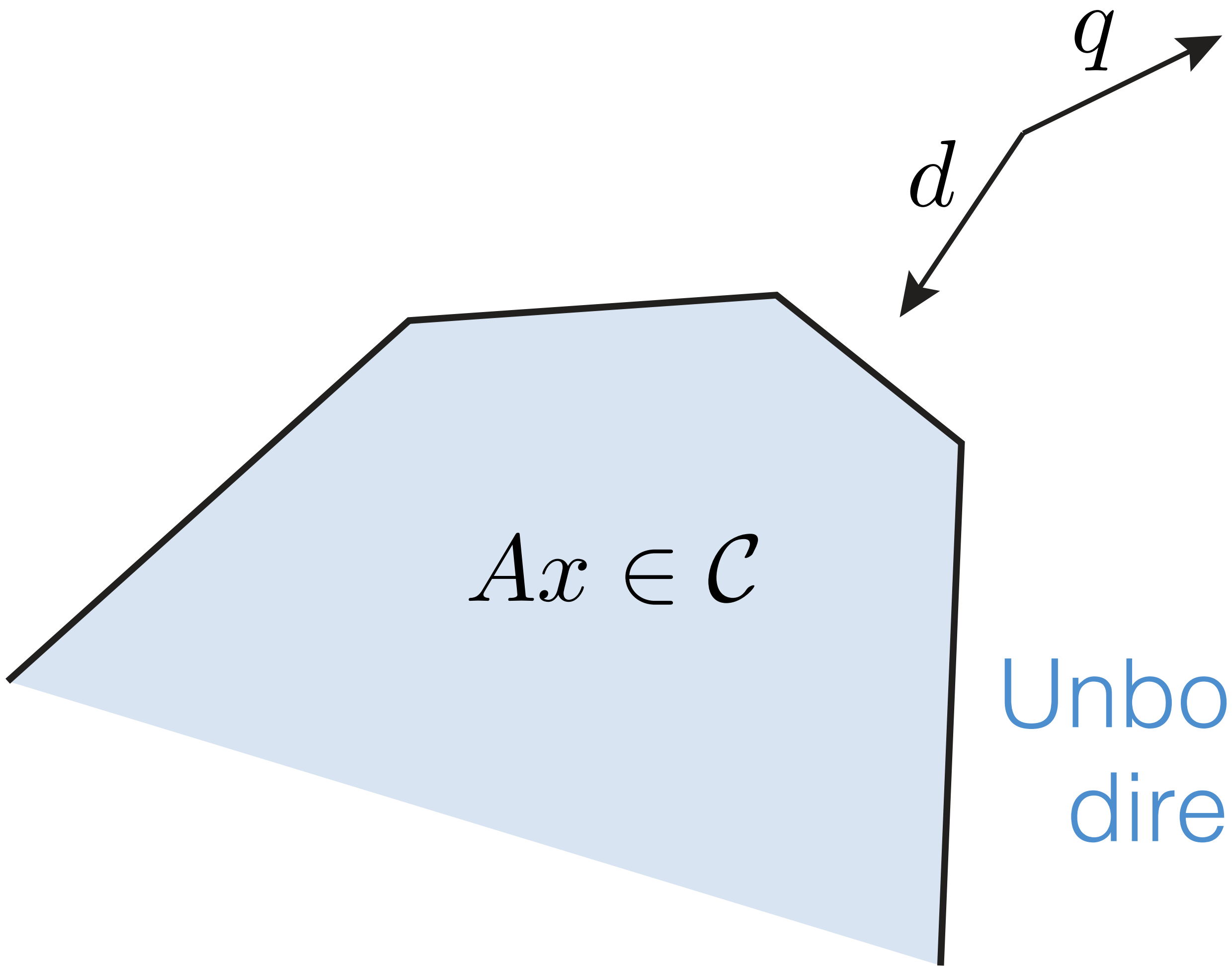
Unbounded direction

Farkas' Lemma

Dual infeasibility

$$Pd = 0 \qquad q^T d < 0 \qquad (Ad)_i \begin{cases} = 0 & l_i, u_i \neq \infty \\ \geq 0 & u_i = +\infty \\ \leq 0 & l_i = -\infty \end{cases}$$

$Ad \in \mathcal{C}^\infty$ Recession cone



Unbounded direction

Infeasibility detection

Primal infeasibility: $\delta y^k = y^k - y^{k-1} \neq 0$

$$A^T \delta y^k \approx 0 \qquad u^T \delta y^k_+ + l^T \delta y^k_- < 0$$

Dual infeasibility: $\delta x^k = x^k - x^{k-1} \neq 0$

$$P \delta x^k \approx 0 \qquad q^T \delta x^k < 0 \qquad (A \delta x^k)_i \begin{cases} \approx 0 & l_i, u_i \neq \infty \\ \geq 0 & u_i = +\infty \\ \leq 0 & l_i = -\infty \end{cases}$$

Conclusions

Acknowledgements

Goran
Banjac

Oxford

Nicholas
Moehle

Stanford

Paul
Goulart

Oxford

Alberto
Bemporad

IMT Lucca

Stephen
Boyd

Stanford

Final remarks

OSQP

Robust

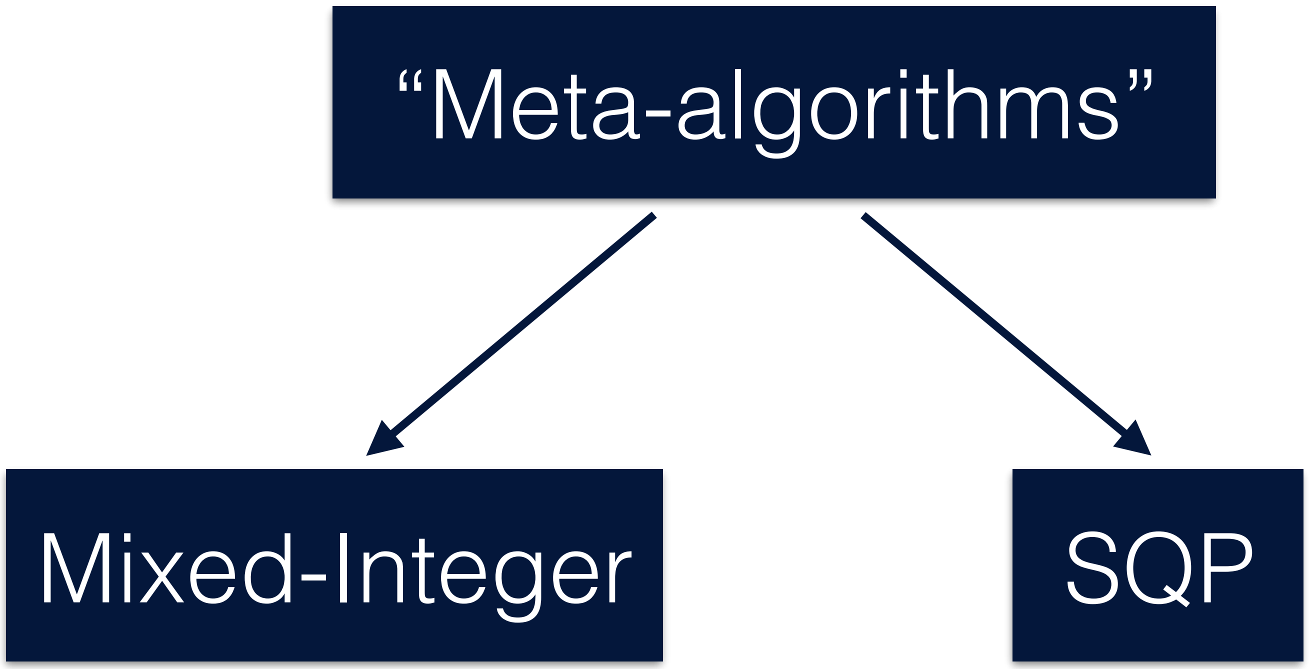
Embeddable

Warm-starting

Detects infeasibility

Future work

Semidefinite programs


“Meta-algorithms”

Mixed-Integer

SQP

References

B. Stellato, G. Banjac, P. Goulart, A. Bemporad and S. Boyd. ***OSQP: An Operator Splitting Solver for Quadratic Programs.*** *(Coming soon!)*

G. Banjac, P. Goulart, B. Stellato, and S. Boyd. ***Infeasibility detection in the alternating direction method of multipliers for convex optimization.*** optimization-online.org, 2017

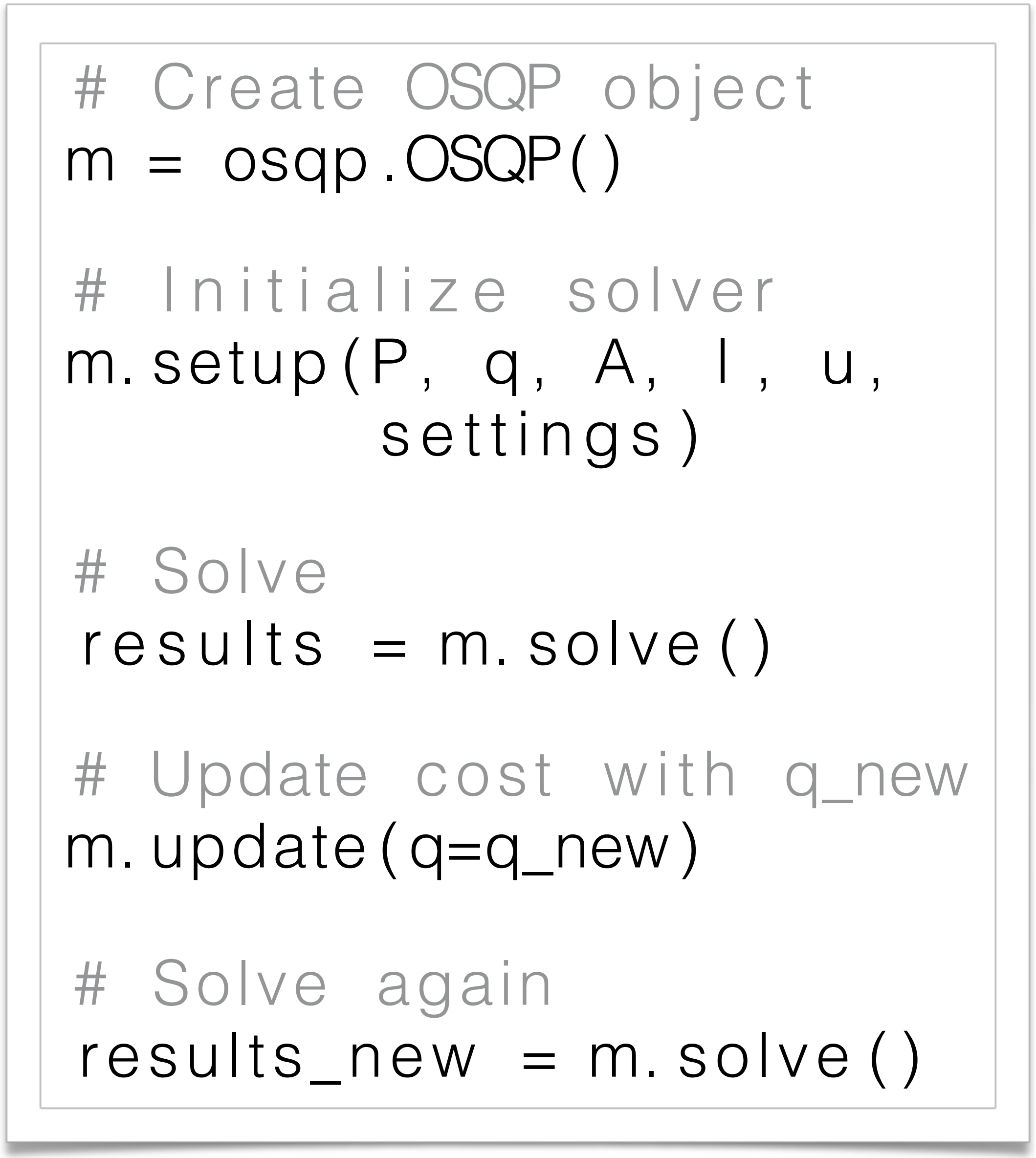
G. Banjac, B.Stellato, N. Moehle, P. Goulart, A. Bemporad and S. Boyd. ***Embedded code generation using the OSQP solver.*** *IEEE Conference on Decision and Control (CDC) (submitted), 2017*

B. Stellato, V. Naik, A. Bemporad, P. Goulart, and S. Boyd. ***Embedded mixed-integer quadratic optimization using the OSQP solver.*** *European Control Conference* (submitted), 2018

Extra Slides

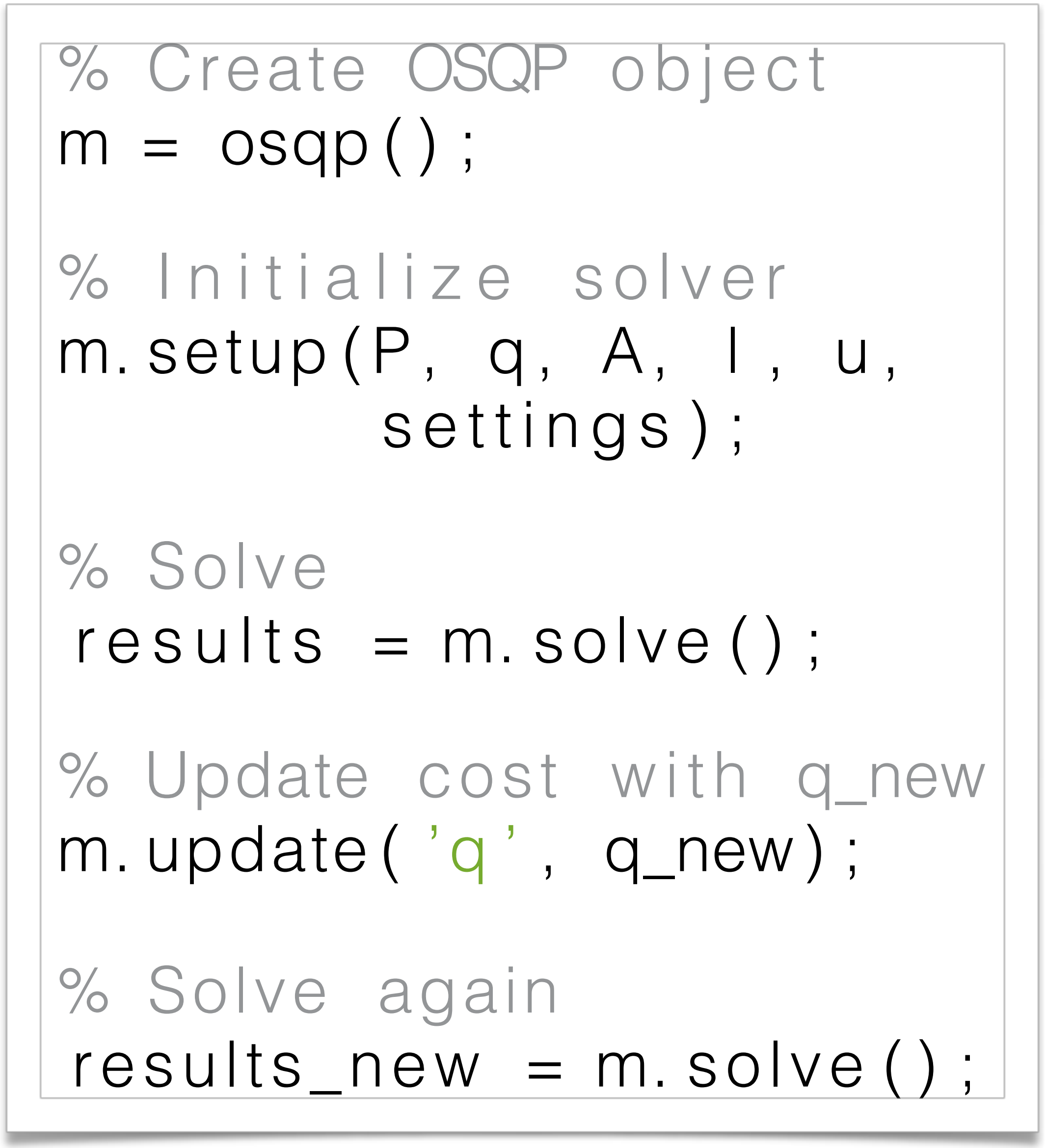
OSQP interface

python



```python
# Create OSQP object
m = osqp.OSQP()

# Initialize solver
m.setup(P, q, A, l, u,
            settings)

# Solve
 results = m.solve()

# Update cost with q_new
m.update(q=q_new)

# Solve again
 results_new = m.solve()
```

MATLAB

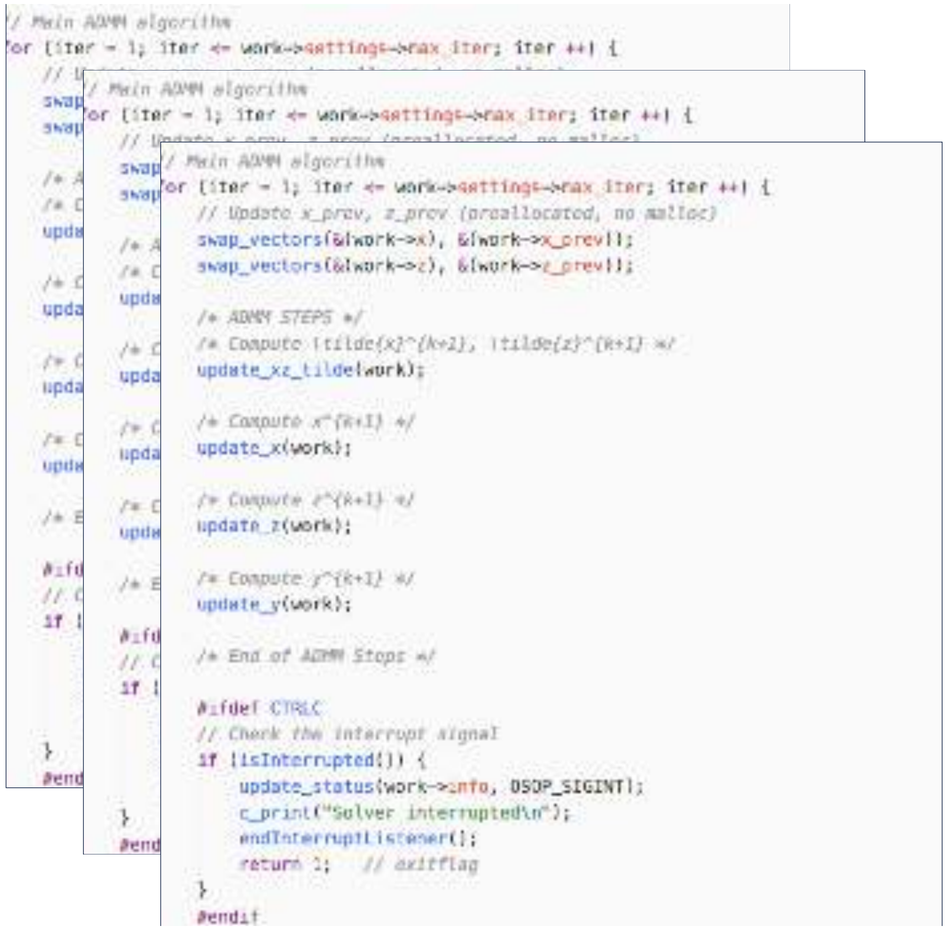


```matlab
% Create OSQP object
m = osqp();

% Initialize solver
m.setup(P, q, A, l, u,
            settings);

% Solve
 results = m.solve();

% Update cost with q_new
m.update('q', q_new);

% Solve again
 results_new = m.solve();
```

Code generation

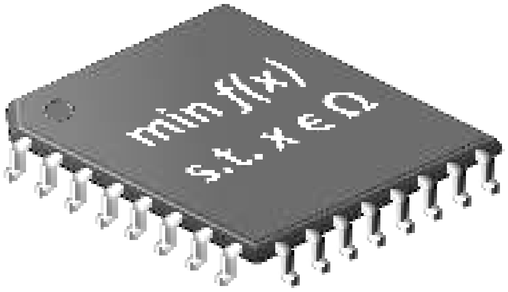
```python
# Create OSQP object
m = osqp.OSQP()

# Initialize solver
m.setup(P, q, A, l, u,
        settings)

# Generate C code
m.codegen('folder_name')
```

Optimized
C code

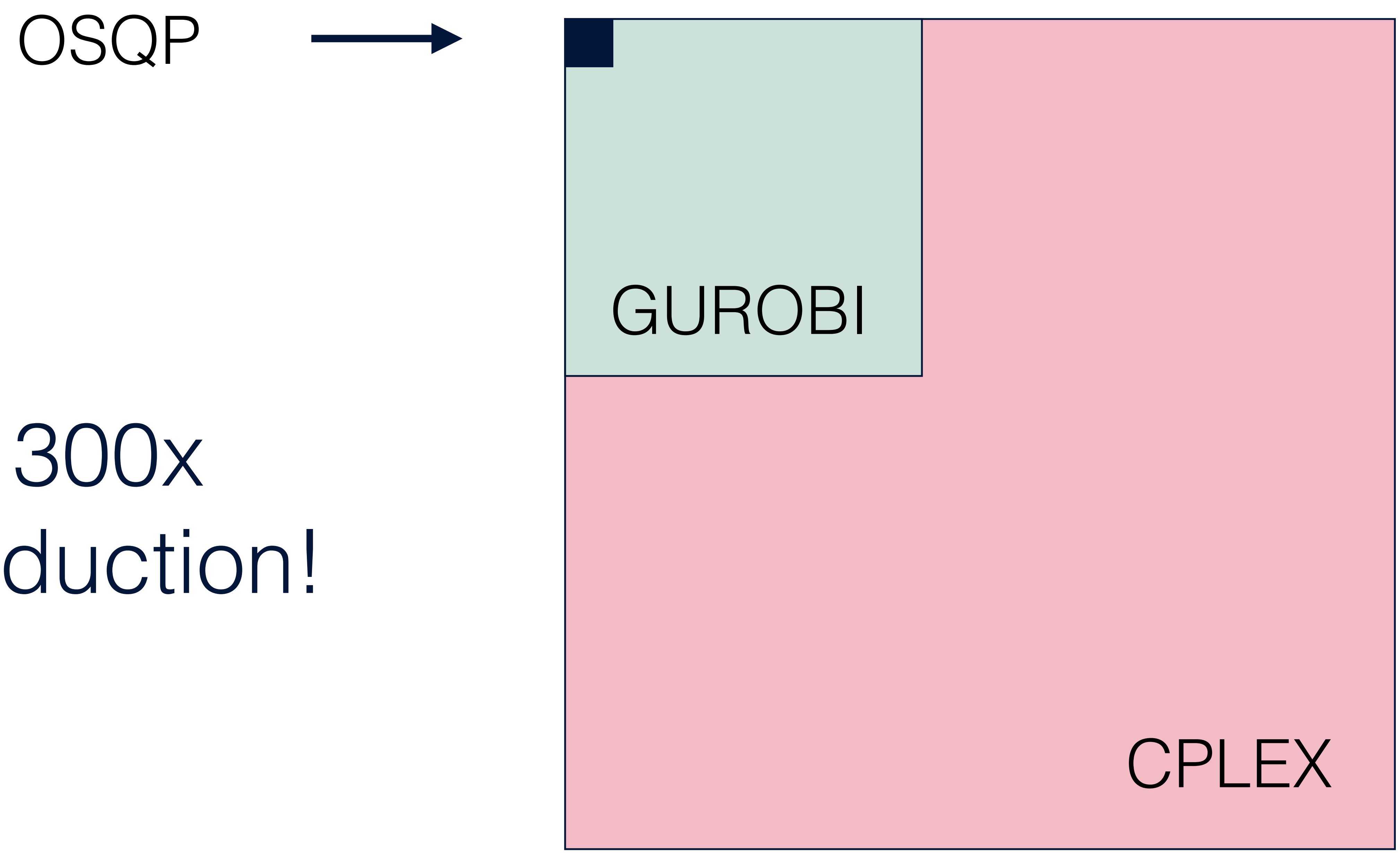


```c
/ Main ADMM algorithm
or (iter = 1; iter <= work->settings->max_iter; iter ++) {
    // Update x_prev, z_prev (preallocated, no malloc)
    swap_vectors(&(work->x), &(work->x_prev));
    swap_vectors(&(work->z), &(work->z_prev));

    /* ADMM STEPS */
    /* Compute \tilde{x}^{k+1}, \tilde{z}^{k+1} */
    update_xz_tilde(work);

    /* Compute x^{k+1} */
    update_x(work);

    /* Compute z^{k+1} */
    update_z(work);

    /* Compute y^{k+1} */
    update_y(work);

    /* End of ADMM Steps */

    #ifdef CTRLC
    // Check the interrupt signal
    if (isInterrupted()) {
        update_status(work->info, OSQP_SIGINT);
        c_print("Solver interrupted\n");
        endInterruptListener();
        return 1;   // exitflag
    }
    #endif
```

Embedded
hardware



Compiled code size ~80kb



300x
Reduction!

Infeasibility

OSQP Algorithm

Averaged non-expansive operator

$$(x^{k+1}, v^{k+1}) = T(x^k, v^k)$$

Original variables

$$z^k = \Pi(v^k) \qquad y^k = \rho(I - \Pi)(v^k)$$

Asymptotic behavior

Averaged non-expansive operator

$$(x^{k+1}, v^{k+1}) = T(x^k, v^k)$$

Differences

$$\delta x^k = x^k - x^{k-1} \qquad \delta v^k = v^k - v^{k-1}$$

Convergence to smallest vector

$$\lim_{k\to\infty} (\delta x^k, \delta v^k) = (\delta x, \delta v) \in \operatorname*{argmin}_{\overline{\mathrm{ran}}(T-I)} \|(\delta x, \delta v)\|$$

[A. Pazy, 1971]

Auxiliary results

Difference of dual iterates: δy

$$A^T \delta y = 0 \qquad u^T \delta y_+ + l^T \delta y_- = -\frac{1}{\rho}\|\delta y\|^2$$

Difference of primal iterates: δx

$$P\delta x = 0 \qquad q^T \delta x = -\sigma\|\delta x\|^2 - \rho\|A\delta x\|^2 \qquad A\delta x \in C^\infty$$

Example

Simple QP

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}x^T \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} x \\ \text{subject to} & 1 \leq \begin{bmatrix} 1 & 0 \end{bmatrix} \leq 2 \end{array}$$

Optimal solution

$$x = (1, 0)$$

Example

Primal infeasible

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}x^T \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} x \\ \text{subject to} & \begin{bmatrix} 1 \\ 1 \end{bmatrix} \leq \begin{bmatrix} 1 & 0 \\ -1 & 0 \end{bmatrix} \leq \begin{bmatrix} 2 \\ 2 \end{bmatrix} \end{array}$$

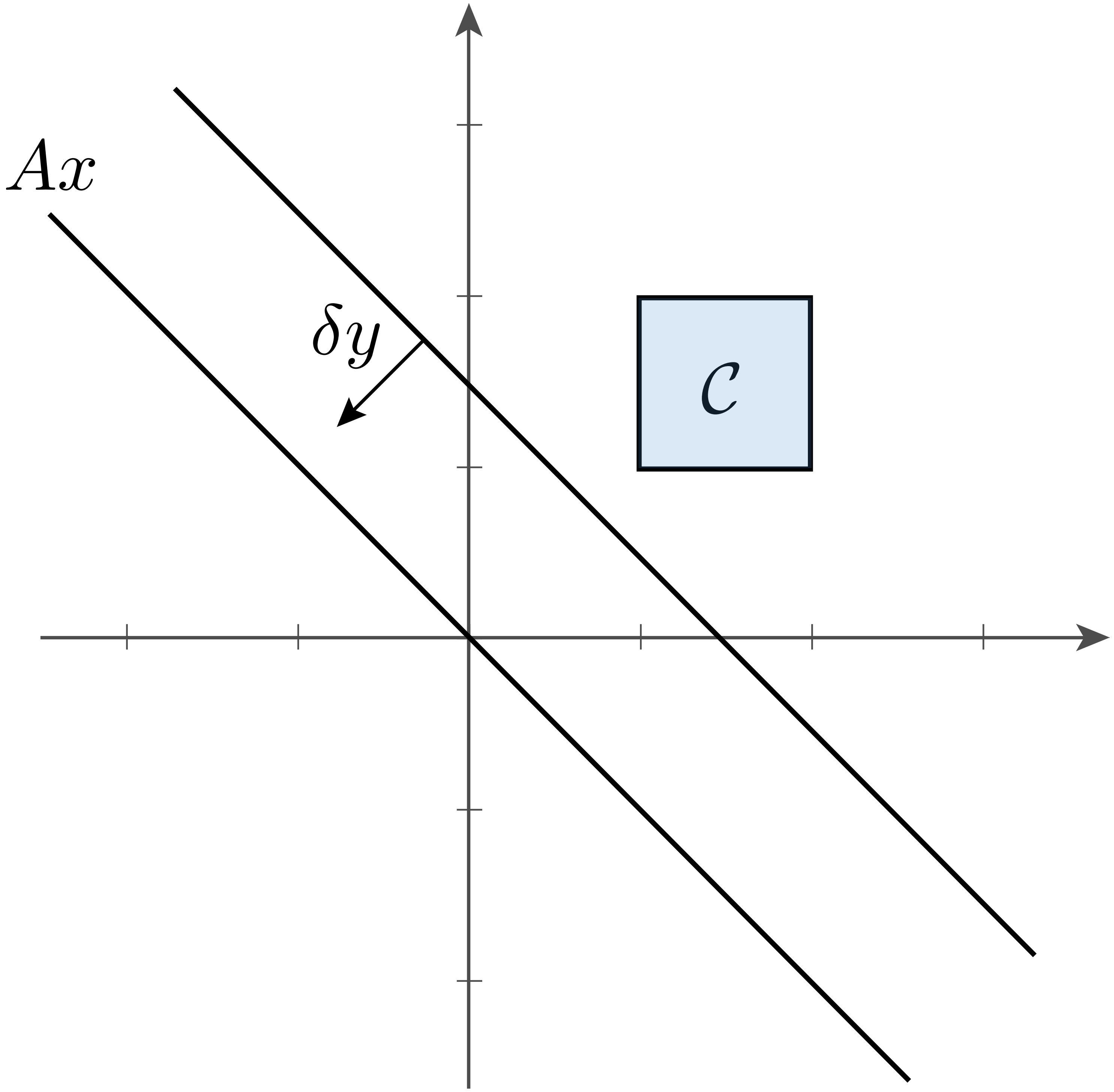
Conflicting constraint

Certificate

$$\delta y = (-1, -1)$$

Example

Primal infeasible



Certificate

$$\delta y = (-1, -1)$$

Example

Dual infeasible

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}x^T \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} x + \boxed{\begin{bmatrix} 0 & 1 \end{bmatrix} x} \\ \text{subject to} & 1 \leq \begin{bmatrix} 1 & 0 \end{bmatrix} \leq 2 \end{array}$$

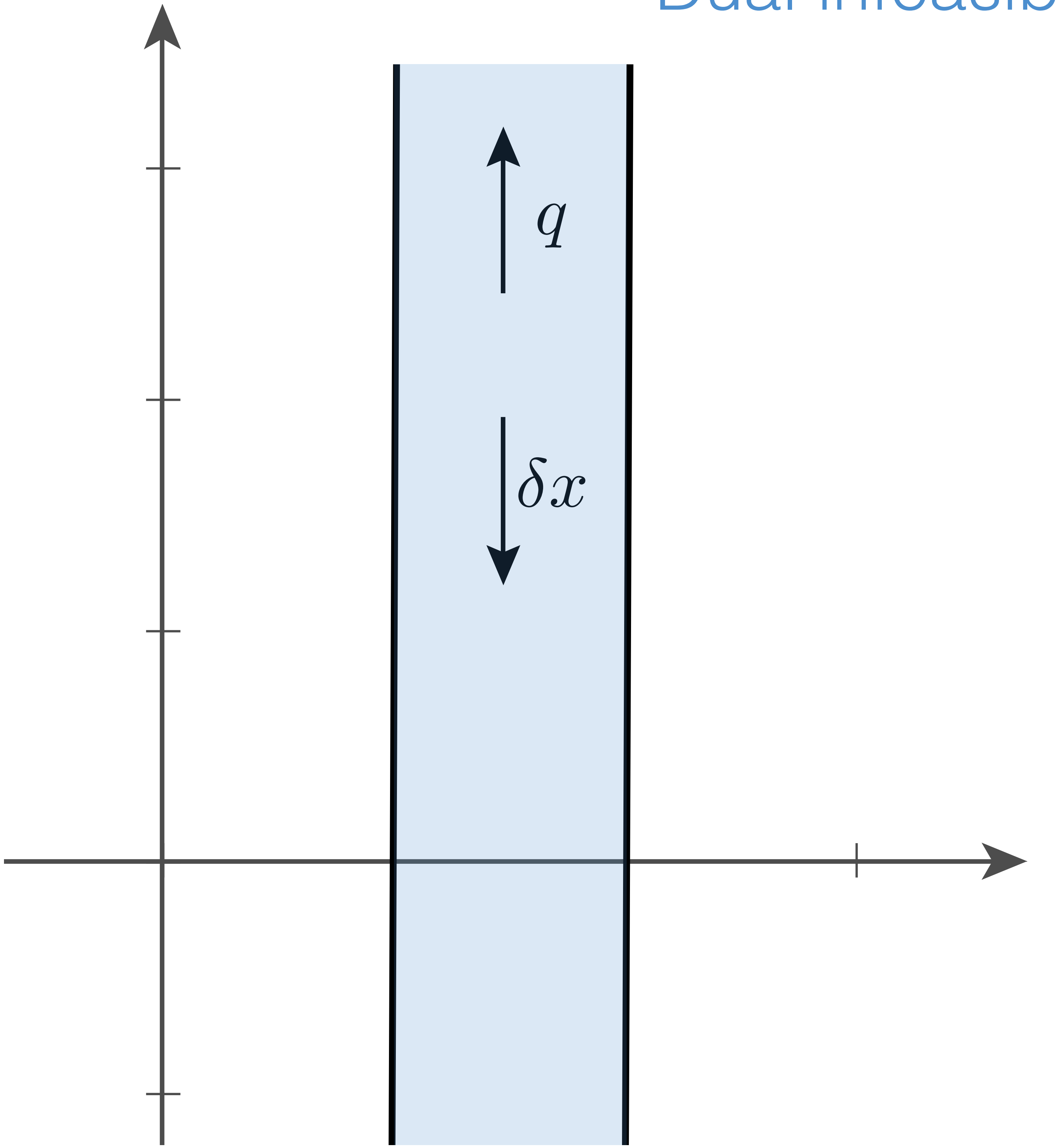
Unbounded direction

Certificate

$$\delta x = (0, -1)$$

Example

Dual infeasible



Certificate

$$\delta x = (0, -1)$$

Example

Primal and dual infeasible

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}x^T \begin{bmatrix}1 & 0\\0 & 0\end{bmatrix} x + \begin{bmatrix}0 & 1\end{bmatrix} x \\ \text{subject to} & \begin{bmatrix}1\\1\end{bmatrix} \leq \begin{bmatrix}1 & 0\\-1 & 0\end{bmatrix} \leq \begin{bmatrix}2\\2\end{bmatrix} \end{array}$$

Unbounded direction

Conflicting constraint

Certificates

$$\delta x = (0, -1)$$
$$\delta y = (-1, -1)$$

Solution Polishing

Obtaining the active set

Lower active

$$\mathcal{L} = \{i \mid (Ax)_i = l_i \wedge y_i < 0\}$$

Upper active

$$\mathcal{U} = \{i \mid (Ax)_i = u_i \wedge y_i > 0\}$$

Solving single linear system

$$\begin{bmatrix} P & A_{\mathcal{L}}^T & A_{\mathcal{U}}^T \\ A_{\mathcal{L}} & & \\ A_{\mathcal{U}} & & \end{bmatrix} \begin{bmatrix} x \\ y_{\mathcal{L}} \\ y_{\mathcal{U}} \end{bmatrix} = \begin{bmatrix} -q \\ l_{\mathcal{L}} \\ u_{\mathcal{U}} \end{bmatrix}$$

Inactive constraints

$$y_i = 0 \quad i \notin (\mathcal{L} \cup \mathcal{U})$$

Solving single linear system

$$\underbrace{\begin{bmatrix} P & A_{\mathcal{L}}^T & A_{\mathcal{U}}^T \\ A_{\mathcal{L}} & & \\ A_{\mathcal{U}} & & \end{bmatrix}}_{M} \begin{bmatrix} x \\ y_{\mathcal{L}} \\ y_{\mathcal{U}} \end{bmatrix} = \underbrace{\begin{bmatrix} -q \\ l_{\mathcal{L}} \\ u_{\mathcal{U}} \end{bmatrix}}_{g}$$

Inactive constraints

$$y_i = 0 \quad i \notin (\mathcal{L} \cup \mathcal{U})$$

Iterative refinement

Perturbed system

$$\begin{bmatrix} P+\delta I & A_{\mathcal{L}}^T & A_{\mathcal{U}}^T \\ A_{\mathcal{L}} & -\delta I & \\ A_{\mathcal{U}} & & -\delta I \end{bmatrix} \begin{bmatrix} \hat{x} \\ \hat{y}_{\mathcal{L}} \\ \hat{y}_{\mathcal{U}} \end{bmatrix} = \begin{bmatrix} -q \\ l_{\mathcal{L}} \\ u_{\mathcal{U}} \end{bmatrix}$$

Iterative refinement

Perturbed system

Quasi-definite →

$$\begin{bmatrix} P+\delta I & A_{\mathcal{L}}^T & A_{\mathcal{U}}^T \\ A_{\mathcal{L}} & -\delta I & \\ A_{\mathcal{U}} & & -\delta I \end{bmatrix} \begin{bmatrix} \hat{x} \\ \hat{y}_{\mathcal{L}} \\ \hat{y}_{\mathcal{U}} \end{bmatrix} = \begin{bmatrix} -q \\ l_{\mathcal{L}} \\ u_{\mathcal{U}} \end{bmatrix}$$

Iterative refinement

Perturbed system

Quasi-definite →

$$\underbrace{\begin{bmatrix} P+\delta I & A_{\mathcal{L}}^T & A_{\mathcal{U}}^T \\ A_{\mathcal{L}} & -\delta I & \\ A_{\mathcal{U}} & & -\delta I \end{bmatrix}}_{M+\Delta} \begin{bmatrix} \hat{x} \\ \hat{y}_{\mathcal{L}} \\ \hat{y}_{\mathcal{U}} \end{bmatrix} = \underbrace{\begin{bmatrix} -q \\ l_{\mathcal{L}} \\ u_{\mathcal{U}} \end{bmatrix}}_{g}$$

Iterative refinement

Perturbed system

Quasi-definite →

$$\underbrace{\begin{bmatrix} P+\delta I & A_{\mathcal{L}}^T & A_{\mathcal{U}}^T \\ A_{\mathcal{L}} & -\delta I & \\ A_{\mathcal{U}} & & -\delta I \end{bmatrix}}_{M+\Delta} \begin{bmatrix} \hat{x} \\ \hat{y}_{\mathcal{L}} \\ \hat{y}_{\mathcal{U}} \end{bmatrix} = \underbrace{\begin{bmatrix} -q \\ l_{\mathcal{L}} \\ u_{\mathcal{U}} \end{bmatrix}}_{g}$$

Iterations

$$\delta t^k \leftarrow \text{solve } (M+\Delta)\delta t^k = g - Kt^k$$

$$t^{k+1} \leftarrow t^k + \delta t^k$$