Leveraging augmented-Lagrangian techniques for differentiating over infeasible quadratic programs in machine learning

Antoine Bambade[1,2]

[1]Inria and ENS Paris : Willow and Sierra teams
[2]École des Ponts Paris Tech

Joint work with Fabian Schramm, Adrien Taylor, Justin Carpentier

Quadratic programming layer pipeline

More recent literature considers differentiable optimization problems as layers.

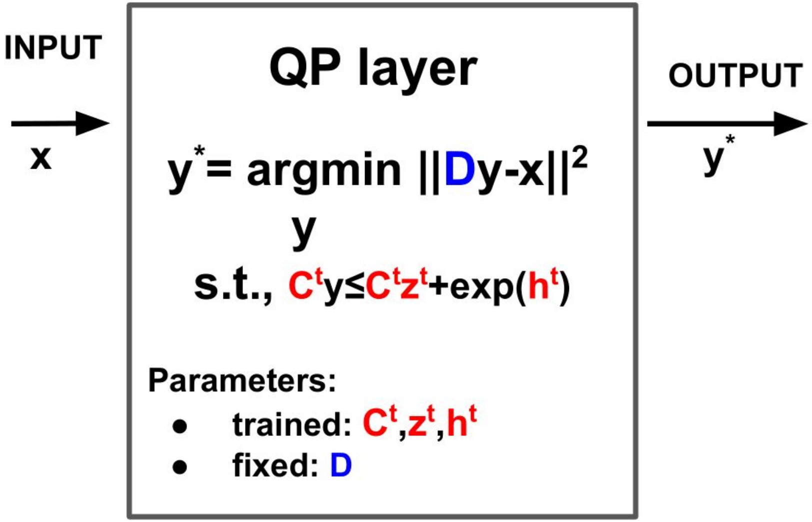


Figure: Example of a Quadratic Programming Layer (with D nonsingular)

QP layers in machine learning

Convex QP layers performs better than a ConvNet for solving Sudokus.

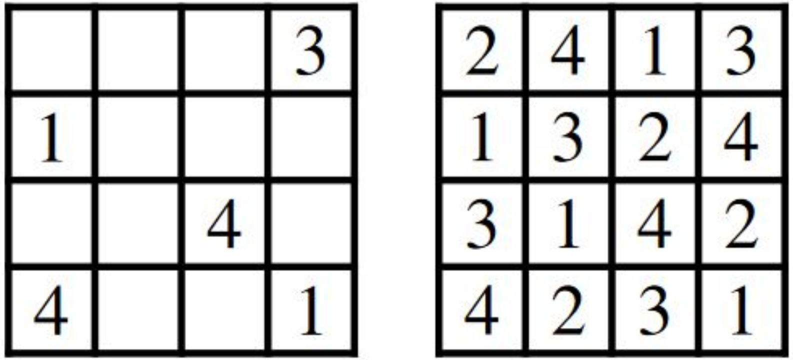


Figure: Example of Sudoku.

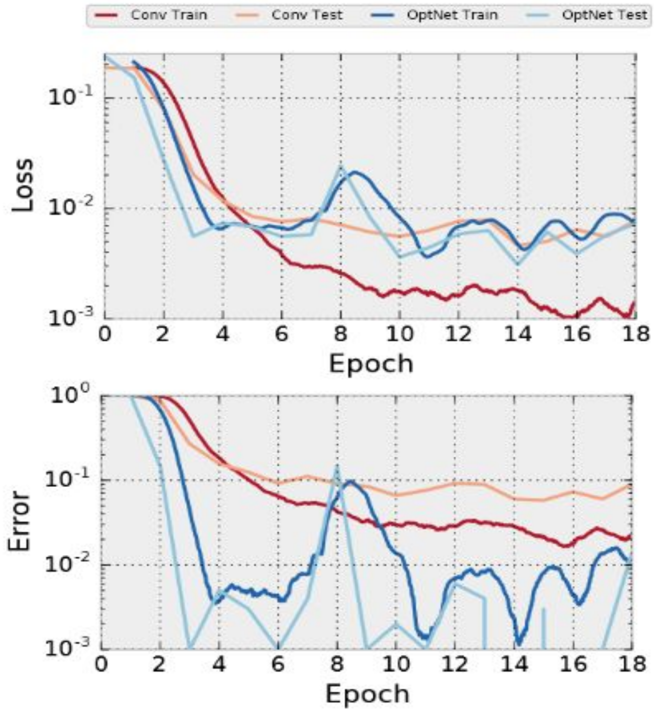


Figure: Training and test plots[1].

[1]B. Amos, Z. Kolter (2021)

QP layers cons: limited trainable architecture

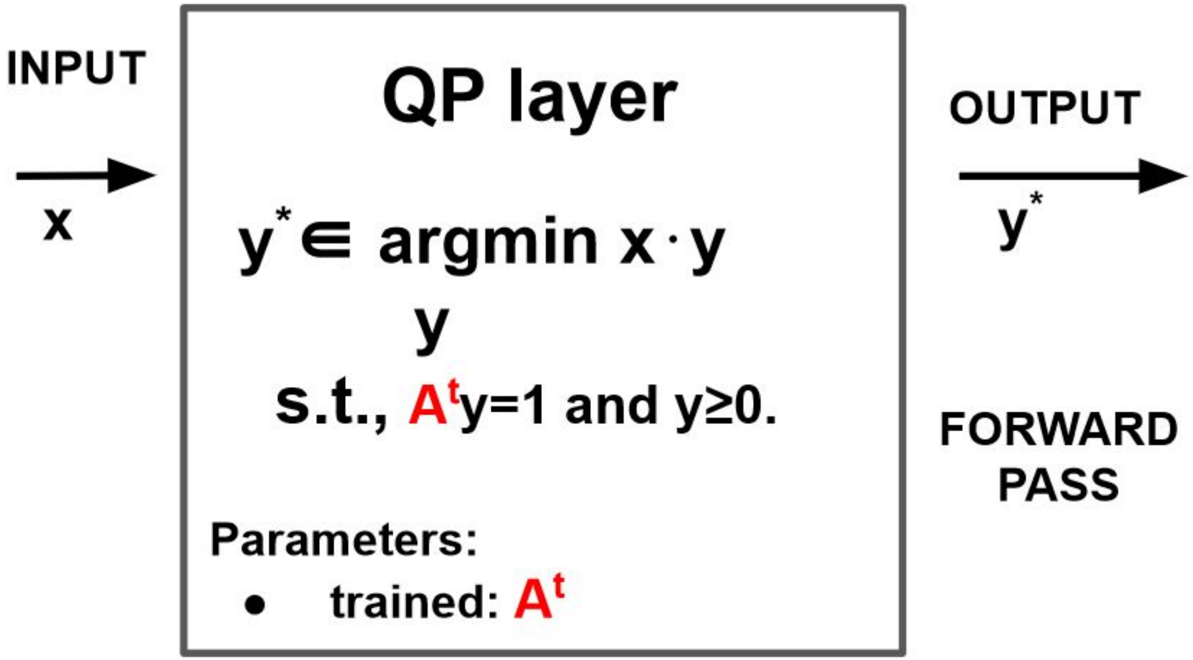
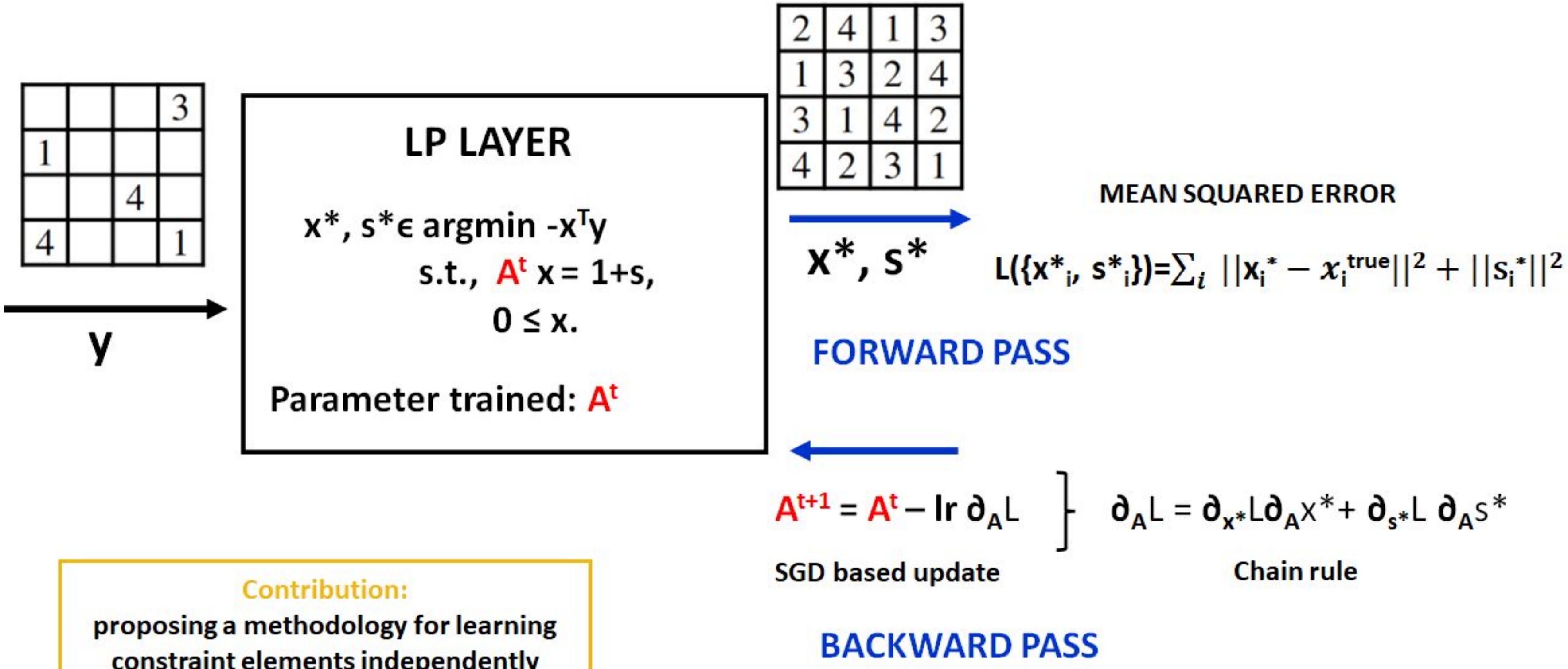


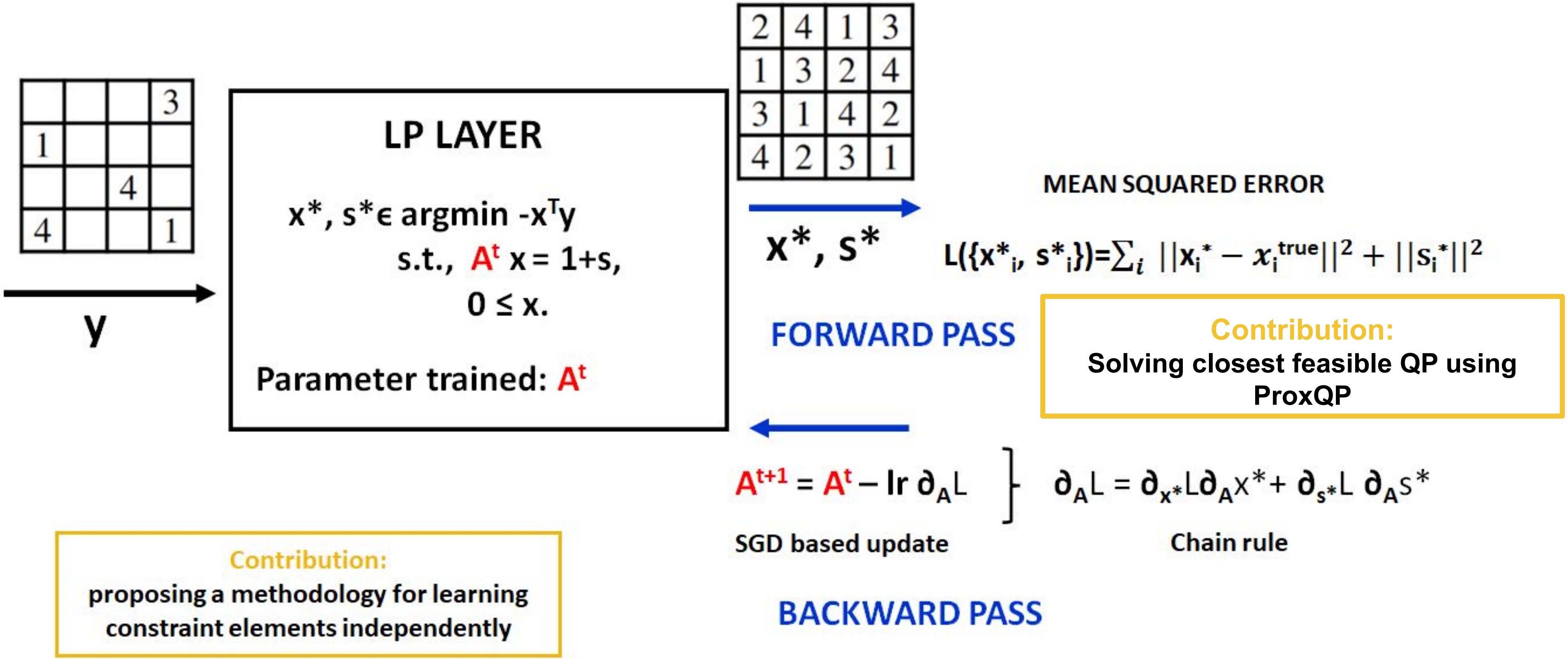
Figure: a LP layer. Nothing guarantees during training that the vector of 1 lies in the range space of A^t.

Solution outline: ideal pipeline

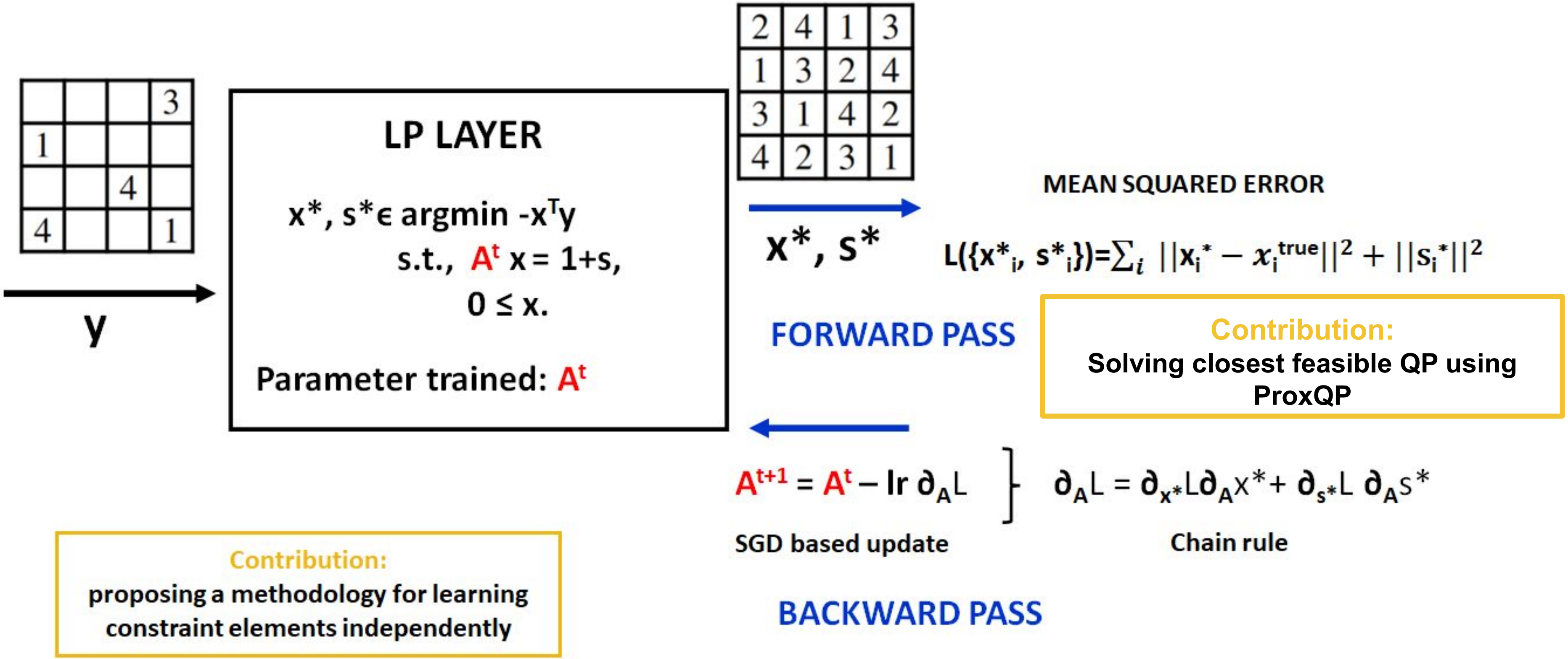


Contribution:
proposing a methodology for learning constraint elements independently

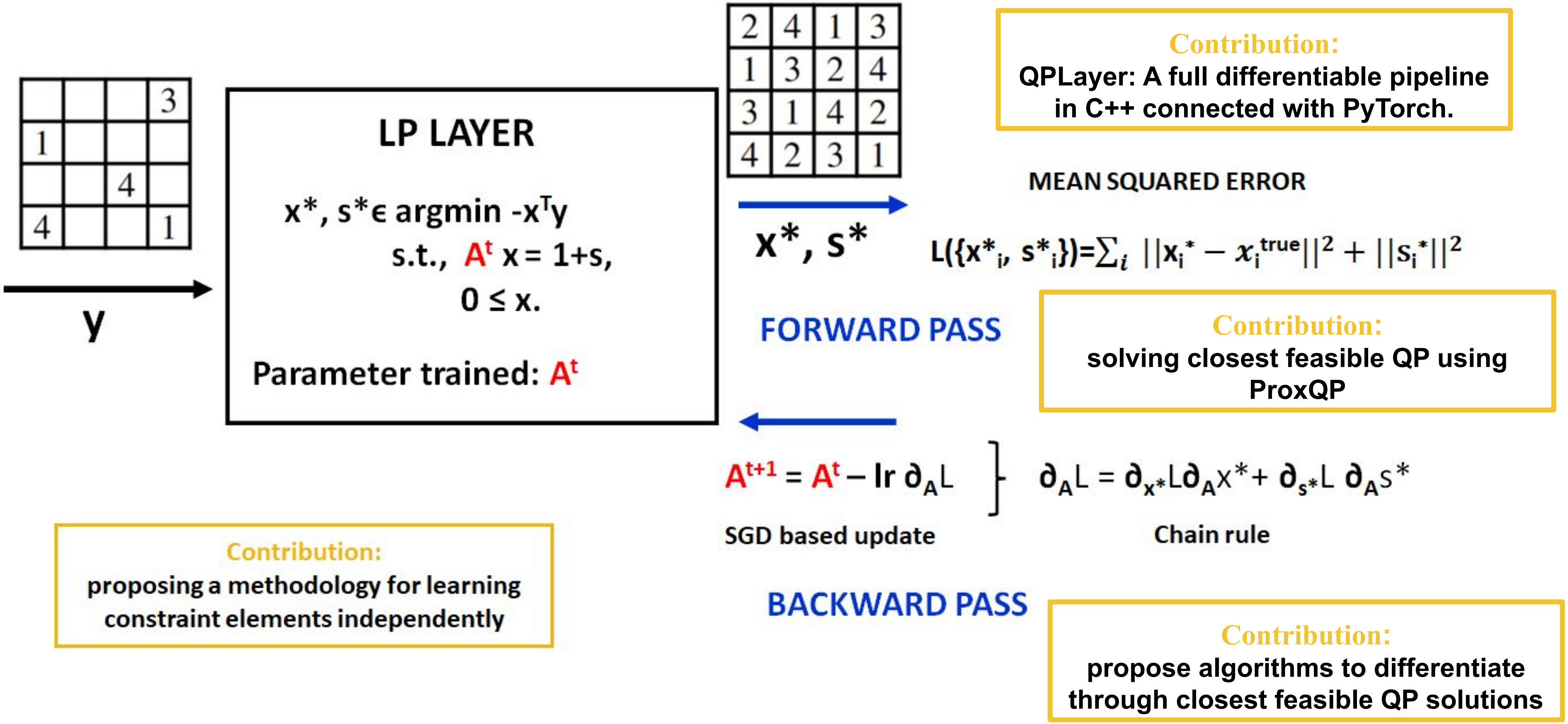
Solution outline: ideal pipeline



A. Chiche, J-C. Gilbert (2016)



A. Chiche, J-C. Gilbert (2016)



3
1
4
4
1
y
LP LAYER
x*, s*ϵ argmin -x^T y
s.t., A^t x = 1+s,
0 ≤ x.
Parameter trained: A^t
2 4 1 3
1 3 2 4
3 1 4 2
4 2 3 1
x*, s*
FORWARD PASS
Contribution:
QPLayer: A full differentiable pipeline in C++ connected with PyTorch.
MEAN SQUARED ERROR
L({x*_i, s*_i})=∑_i ||x_i* − x_i^true||^2 + ||s_i*||^2
Contribution:
solving closest feasible QP using ProxQP
A^{t+1} = A^t – lr ∂_A L
SGD based update
∂_A L = ∂_{x*} L ∂_A x* + ∂_{s*} L ∂_A s*
Chain rule
BACKWARD PASS
Contribution:
proposing a methodology for learning constraint elements independently
Contribution:
propose algorithms to differentiate through closest feasible QP solutions

Software contribution

ProxSuite

THE ADVANCED PROXIMAL OPTIMIZATION TOOLBOX

License BSD 2-Clause | docs online | CI - Linux/OSX/Windows - Cond passing | pypi package 0.6.1 | Anaconda.org 0.6.1

- ✓ **fast**: C++ implementation, with homemade linear Cholesky solver
- ✓ **scalable:** various backends for dense, sparse and matrix-free optimization
- ✓ **easy-to-use:** API closed to OSQP, Python and Julia bindings
- ✓ **open-source:** BSD-license, easily installable

Conda | Files | Labels | Badges

- License: BSD-2-Clause
- Home: https://github.com/simple-robotics/proxsuite
- </> Development: https://github.com/simple-robotics/proxsuite
- 160860 total downloads
- Last upload: 1 month and 25 days ago

Watch 14 | Fork 40 | Starred 303

Summary

PyPI link

https://pypi.org/project/proxsuite

Total downloads

180,196

Total downloads - 30 days

11,791

Total downloads - 7 days

3,257

The backward pass: differentiating closest QP solutions

The backward pass: differentiating closest QP solutions

A classical technique: **the Implicit Function Theorem.**

The backward pass: differentiating closest QP solutions

$$s^\star(\theta) = \arg \min_{s \in \mathbb{R}^{n_i}} \frac{1}{2}\|s\|_2^2$$

$$\text{s.t. } x^\star(\theta), z^\star(\theta) \in \arg \min_{x \in \mathbb{R}^n} \max_{z \in \mathbb{R}_+^{n_i}} L(x, z, s;\ \theta),$$

$$\text{with } L(x, z, s;\ \theta) \triangleq f(x;\ \theta) + z^\top (C(\theta)x - u(\theta) - s).$$

A classical technique: **the Implicit Function Theorem.**

The backward pass: differentiating closest QP solutions

$$s^\star(\theta) = \arg \min_{s \in \mathbb{R}^{n_i}} \tfrac{1}{2}\|s\|_2^2$$
$$\text{s.t. } x^\star(\theta), z^\star(\theta) \in \arg \min_{x \in \mathbb{R}^n} \max_{z \in \mathbb{R}^{n_i}_+} L(x, z, s;\ \theta),$$

$$\text{with } L(x, z, s;\ \theta) \triangleq f(x;\ \theta) + z^\top (C(\theta)x - u(\theta) - s).$$

A classical technique: **the Implicit Function Theorem.**

$$G(x, z, t;\ \theta) \triangleq \begin{bmatrix} \nabla_x f(x;\ \theta) + C(\theta)^\top z \\ C(\theta)x - u(\theta) - t \\ [[t]_- + z]_+ - z \\ C(\theta)^\top [t]_+ \end{bmatrix}$$

The backward pass: differentiating closest QP solutions

$$
\begin{aligned}
s^\star(\theta) =& \arg\min_{s\in\mathbb{R}^{n_i}} \ \tfrac{1}{2}\|s\|_2^2 \\
& \text{s.t. } x^\star(\theta), z^\star(\theta) \in \arg\min_{x\in\mathbb{R}^n} \max_{z\in\mathbb{R}_+^{n_i}} L(x,z,s;\ \theta),
\end{aligned}
$$

$$
\text{with } L(x,z,s;\ \theta) \triangleq f(x;\ \theta) + z^\top(C(\theta)x - u(\theta) - s).
$$

A classical technique: **the Implicit Function Theorem.**

$$
G(x,z,t;\ \theta) \triangleq \begin{bmatrix} \nabla_x f(x;\ \theta) + C(\theta)^\top z \\ C(\theta)x - u(\theta) - t \\ [[t]_- + z]_+ - z \\ C(\theta)^\top [t]_+ \end{bmatrix}
$$

The map is **path-differentiable**.

E. Pauwels et al. (2019)

The backward pass: differentiating closest QP solutions

$$s^\star(\theta) = \arg \min_{s \in \mathbb{R}^{n_i}} \tfrac{1}{2}\|s\|_2^2$$
$$\text{s.t. } x^\star(\theta), z^\star(\theta) \in \arg \min_{x \in \mathbb{R}^n} \max_{z \in \mathbb{R}_+^{n_i}} L(x, z, s;\ \theta),$$

with $L(x, z, s;\ \theta) \triangleq f(x;\ \theta) + z^\top (C(\theta)x - u(\theta) - s)$.

A classical technique: **the Implicit Function Theorem.**

$$G(x, z, t;\ \theta) \triangleq \begin{bmatrix} \nabla_x f(x;\ \theta) + C(\theta)^\top z \\ C(\theta)x - u(\theta) - t \\ [[t]_- + z]_+ - z \\ C(\theta)^\top [t]_+ \end{bmatrix}$$

The map is **path-differentiable**.

$$\left(\frac{\partial x^\star}{\partial \theta}, \frac{\partial z^\star}{\partial \theta}\right) \in \arg\min_w \left\| \frac{\partial G(x^\star, z^\star;\ \theta)}{\partial v^\star} w + \frac{\partial G(x^\star, z^\star;\ \theta)}{\partial \theta} \right\|_2^2,$$

$$\Pi \frac{\partial t^\star}{\partial \theta} \in \frac{\partial s^\star}{\partial \theta}, \text{ with } \Pi \in \partial([.]_+)(t^\star).$$

The backward pass: differentiating closest QP solutions

$$s^\star(\theta) = \arg\min_{s\in\mathbb{R}^{n_i}} \tfrac{1}{2}\|s\|_2^2$$
$$\text{s.t. } x^\star(\theta), z^\star(\theta) \in \arg\min_{x\in\mathbb{R}^n}\max_{z\in\mathbb{R}^{n_i}_+} L(x,z,s;\ \theta),$$

with $L(x,z,s;\ \theta) \triangleq f(x;\ \theta) + z^\top(C(\theta)x - u(\theta) - s)$.

A classical technique: **the Implicit Function Theorem.**

Contribution:
Extend the technique for the closest feasible QP solutions.

$$G(x,z,t;\ \theta) \triangleq \begin{bmatrix} \nabla_x f(x;\ \theta) + C(\theta)^\top z \\ C(\theta)x - u(\theta) - t \\ [[t]_- + z]_+ - z \\ C(\theta)^\top [t]_+ \end{bmatrix}$$

The map is **path-differentiable**.

$$\left(\frac{\partial x^\star}{\partial\theta}, \frac{\partial z^\star}{\partial\theta}\right) \in \arg\min_w \left\| \frac{\partial G(x^\star, z^\star;\ \theta)}{\partial v^\star} w + \frac{\partial G(x^\star, z^\star;\ \theta)}{\partial\theta} \right\|_2^2,$$

$$\Pi\frac{\partial t^\star}{\partial\theta} \in \frac{\partial s^\star}{\partial\theta}, \text{ with } \Pi \in \partial([.]_+)(t^\star).$$

The backward pass: differentiating closest QP solutions

$$s^\star(\theta) = \arg\min_{s\in\mathbb{R}^{n_i}} \tfrac{1}{2}\|s\|_2^2$$
$$\text{s.t. } x^\star(\theta), z^\star(\theta) \in \arg\min_{x\in\mathbb{R}^n} \max_{z\in\mathbb{R}_+^{n_i}} L(x,z,s;\ \theta),$$

with $L(x,z,s;\ \theta) \triangleq f(x;\ \theta) + z^\top(C(\theta)x - u(\theta) - s)$.

A classical technique: **the Implicit Function Theorem.**

Contribution:
Extend the technique for the closest feasible QP solutions.

$$G(x,z,t;\ \theta) \triangleq \begin{bmatrix} \nabla_x f(x;\ \theta) + C(\theta)^\top z \\ C(\theta)x - u(\theta) - t \\ [[t]_- + z]_+ - z \\ C(\theta)^\top [t]_+ \end{bmatrix}$$

The map is **path-differentiable**.

$$\left(\frac{\partial x^\star}{\partial \theta}, \frac{\partial z^\star}{\partial \theta}\right) \in \arg\min_w \left\| \frac{\partial G(x^\star, z^\star;\ \theta)}{\partial v^\star} w + \frac{\partial G(x^\star, z^\star;\ \theta)}{\partial \theta} \right\|_2^2,$$

$$\Pi \frac{\partial t^\star}{\partial \theta} \in \frac{\partial s^\star}{\partial \theta}, \text{ with } \Pi \in \partial([.]_+)(t^\star).$$

Contribution:
Efficient algorithms to solve these problems.

The backward pass: differentiating closest QP solutions

$$s^\star(\theta) = \arg\min_{s\in\mathbb{R}^{n_i}} \tfrac{1}{2}\|s\|_2^2$$
$$\text{s.t. } x^\star(\theta), z^\star(\theta) \in \arg\min_{x\in\mathbb{R}^n} \max_{z\in\mathbb{R}_+^{n_i}} L(x, z, s;\ \theta),$$

$$\text{with } L(x, z, s;\ \theta) \triangleq f(x;\ \theta) + z^\top (C(\theta)x - u(\theta) - s).$$

Contribution:
QPLayer: A full differentiable pipeline in C++ connected with PyTorch.

A classical technique: **the Implicit Function Theorem.**

$$G(x, z, t;\ \theta) \triangleq \begin{bmatrix} \nabla_x f(x;\ \theta) + C(\theta)^\top z \\ C(\theta)x - u(\theta) - t \\ [[t]_- + z]_+ - z \\ C(\theta)^\top [t]_+ \end{bmatrix}$$

Contribution:
Extend the technique for the closest feasible QP solutions.

The map is **path-differentiable**.

$$\left(\frac{\partial x^\star}{\partial \theta}, \frac{\partial z^\star}{\partial \theta}\right) \in \arg\min_w \left\| \frac{\partial G(x^\star, z^\star;\ \theta)}{\partial v^\star} w + \frac{\partial G(x^\star, z^\star;\ \theta)}{\partial \theta} \right\|_2^2,$$

$$\Pi \frac{\partial t^\star}{\partial \theta} \in \frac{\partial s^\star}{\partial \theta}, \text{ with } \Pi \in \partial([.]_+)(t^\star).$$

Contribution:
Efficient algorithms to solve these problems.

$$s^\star(\theta) = \arg\min_{s \in \mathbb{R}^{n_i}} \tfrac{1}{2}\|s\|_2^2$$
$$\text{s.t. } x^\star(\theta), z^\star(\theta) \in \arg\min_{x\in\mathbb{R}^n} \max_{z \in \mathbb{R}^{n_i}_+} L(x, z, s;\ \theta),$$
$$\text{with } L(x, z, s;\ \theta) \triangleq f(x;\ \theta) + z^\top (C(\theta)x - u(\theta) - s).$$

Contribution:
QPLayer: A full differentiable pipeline in C++ connected with PyTorch.

A classical technique: **the Implicit Function Theorem.**

$$G(x, z, t;\ \theta) \triangleq \begin{bmatrix} \nabla_x f(x;\ \theta) + C(\theta)^\top z \\ C(\theta)x - u(\theta) - t \\ [[t]_- + z]_+ - z \\ C(\theta)^\top [t]_+ \end{bmatrix}$$

Contribution:
Extend the technique for the closest feasible QP solutions.

The map is **path-differentiable**.

$$\left(\frac{\partial x^\star}{\partial \theta}, \frac{\partial z^\star}{\partial \theta}\right) \in \arg\min_w \left\| \frac{\partial G(x^\star, z^\star;\ \theta)}{\partial v^\star} w + \frac{\partial G(x^\star, z^\star;\ \theta)}{\partial \theta} \right\|_2^2,$$
$$\Pi \frac{\partial t^\star}{\partial \theta} \in \frac{\partial s^\star}{\partial \theta}, \text{ with } \Pi \in \partial([.]_+)(t^\star).$$

Contribution:
Efficient algorithms to solve these problems.

Numerical benchmark: back to the Sodoku problem.

Convex QP layers performs better than a ConvNet for solving Sudokus.

			3
1			
		4	
4			1

2	4	1	3
1	3	2	4
3	1	4	2
4	2	3	1

Figure: Example of Sudoku.

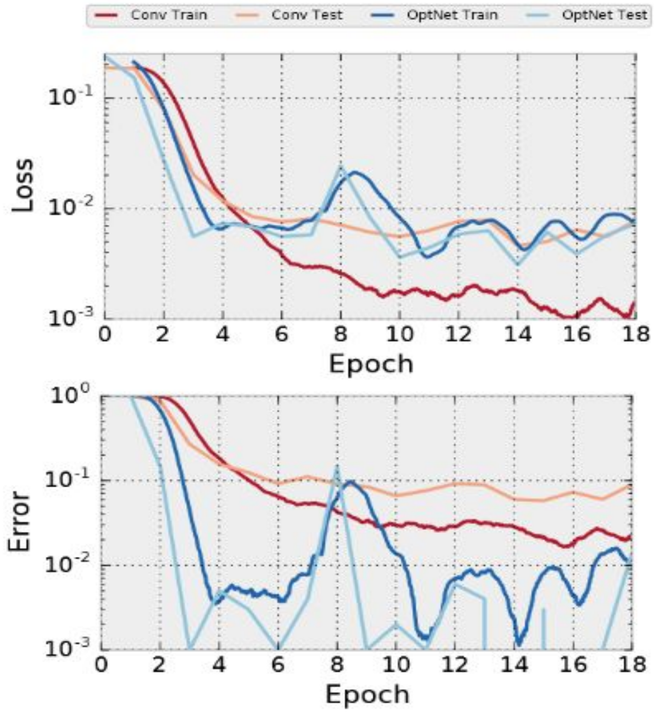
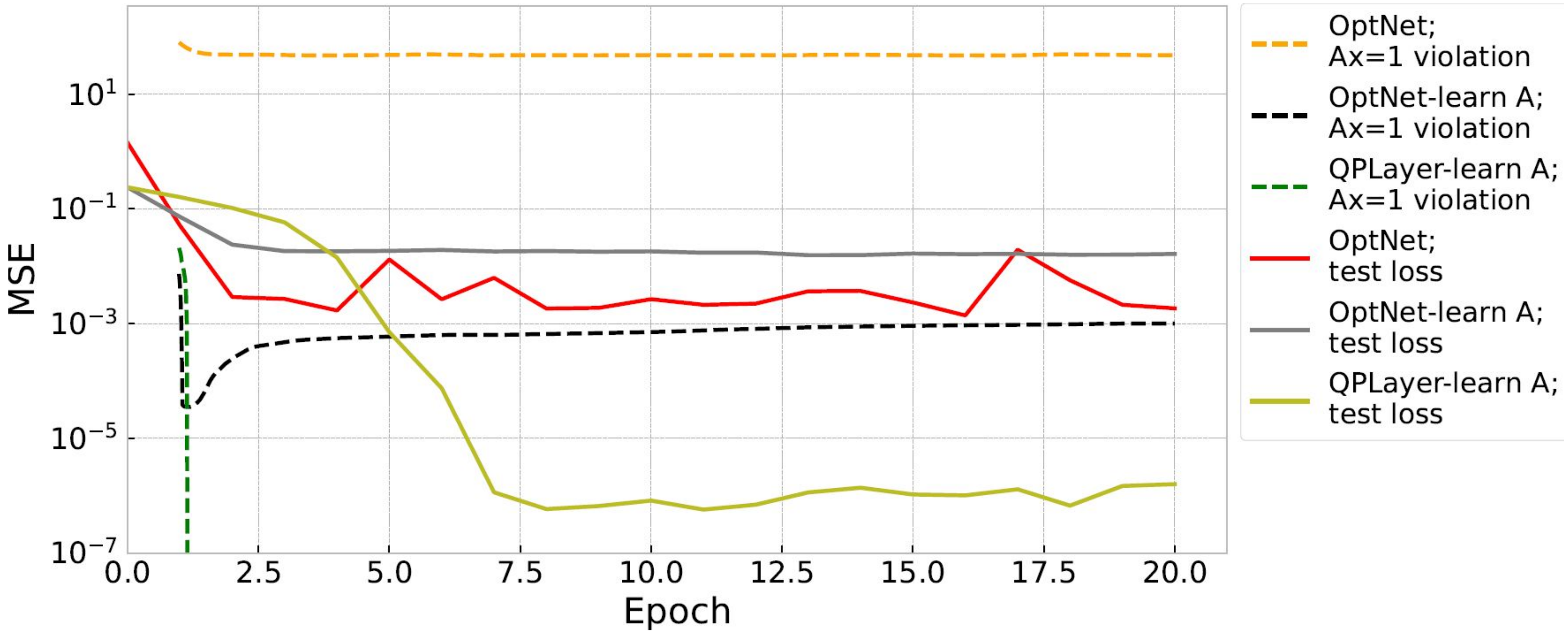


Figure: Training and test plots[1].

[1] B. Amos, Z. Kolter (2021)

Loss comparison



- **Methodology for learning new QP layers**
  - IFT for closest feasible QPs
  - Extended conservative Jacobians
- **QPlayer: open-source differentiable pipeline**
  - Use Augmented-Lagrangian techniques
  - Connected with PyTorch

Antoine Bambade[1,2]

[1]*Inria and ENS Paris : Willow and Sierra teams*

[2]*École des Ponts Paris Tech*

https://github.com/Simple-Robotics/proxsuite

ProxSuite

THE ADVANCED PROXIMAL OPTIMIZATION TOOLBOX

License BSD 2-Clause | docs online | CI - Linux/OSX/Windows - Cond passing | pypi package 0.6.1 | Anaconda.org 0.6.1

- ✓ **fast**: C++ implementation, with homemade linear Cholesky solver
- ✓ **scalable:** various backends for dense, sparse and matrix-free optimization
- ✓ **easy-to-use:** API closed to OSQP, Python and Julia bindings
- ✓ **open-source:** BSD-license, easily installable

Conda | Files | Labels | Badges

License: BSD-2-Clause

Home: https://github.com/simple-robotics/proxsuite

</> Development: https://github.com/simple-robotics/proxsuite

160860 total downloads

Last upload: 1 month and 25 days ago

Watch 14 | Fork 40 | Starred 303

Summary

PyPI link

https://pypi.org/project/proxsuite

Total downloads

180,196

Total downloads - 30 days

11,791

Total downloads - 7 days

3,257

Software contribution

Exact Method of Multipliers launched from $\mathbf{z^0}$ with $\sum_k 1/\mu^k$ finite

$$z^{k+1} = z^k - 1/\mu^k s^k, \quad s^k \in \partial\delta(z^{k+1})$$