Optimization with Superquantile Constraints: A Fast Computational Approach

Ying Cui
Department of Industrial Engineering and Operations Research
University of California, Berkeley

Joint work with Jake Roth (University of Minnesota)

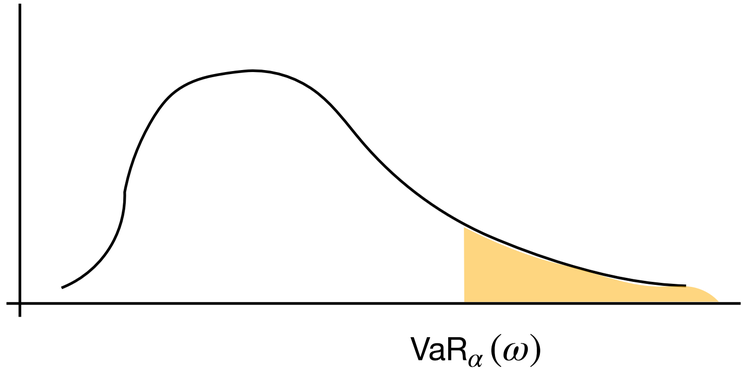
Superquantile

- Superquantile / conditional value-at-risk (CVaR) / average value-at-risk / tail value-at-risk / expected shortfall [Ben-Tal and Teboulle, Rockafellar and Uryasev, Rockafellar and Royset…]

$$\text{CVaR}_\alpha(\omega) = \text{Average of the worst } (1-\alpha)100\% \text{ outcomes of } \omega$$

$$= \min_\eta \ \eta + \frac{1}{1-\alpha}\mathbb{E}[\max(\omega - \eta, 0)]$$

- Many equivalent formulations
- ``top-k-sum" in machine learning



Superquantile Optimization

$$\begin{aligned} \min_{x \in X} \quad & \theta(x) + \mathsf{CVaR}_{\alpha_0}[\, f_0(x, \omega)\,] \\ \text{s.t.} \quad & \mathsf{CVaR}_{\alpha_i}[\, f_i(x, \omega)\,] \leq r_i, \quad i = 1, \cdots, L \end{aligned}$$

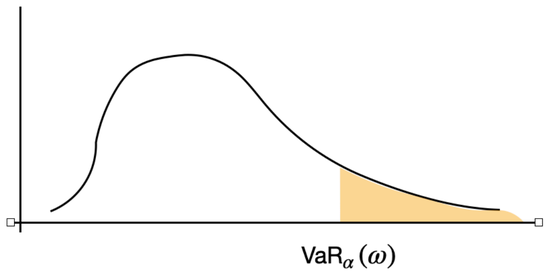
- The problem is convex if θ, $\{f_i(\,\bullet\,, \omega)\}_{i=0}^{L}$, X are convex
- Financial decisions, operational plans, military strategies, engineering designs, machine learning, statistical models… [survey paper by Royset (2023)]

Expectation vs Superquantile

$$\mathbb{E}[\, f(x,\omega)\,] \approx \frac{1}{S}\sum_{s=1}^{S} f(x,\omega^s)$$

$$\mathsf{CVaR}_\alpha\,[\, f(x,\omega)\,] \approx \frac{1}{(1-\alpha)S}\sum_{s=1}^{(1-\alpha)S} f(x,\omega^{[s]})$$

where $f(x,\omega^{[1]}) \geq f(x,\omega^{[2]}) \geq \cdots \geq f(x,\omega^{[S]})$



Expectation vs Superquantile

$$\mathbb{E}[\, f(x, \omega)\,] \approx \frac{1}{S} \sum_{s=1}^{S} f(x, \omega^s)$$

- **Separable**: samples are equally important

$$\text{CVaR}_\alpha\,[\, f(x, \omega)\,] \approx \frac{1}{(1-\alpha)S} \sum_{s=1}^{(1-\alpha)S} f(x, \omega^{[s]})$$

where $f(x, \omega^{[1]}) \geq f(x, \omega^{[2]}) \geq \cdots \geq f(x, \omega^{[S]})$

- **Non-separable**: samples are not equally important —> only care about **tail** expectation

Expectation vs Superquantile

$$\mathbb{E}[\, f(x, \omega) \,] \approx \frac{1}{S} \sum_{s=1}^{S} f(x, \omega^s)$$

- Separable: samples are equally important
- Can take an arbitrary batch of samples to unbiasedly estimate the function value and the (sub)gradient

$$\text{CVaR}_\alpha [\, f(x, \omega) \,] \approx \frac{1}{(1-\alpha)S} \sum_{s=1}^{(1-\alpha)S} f(x, \omega^{[s]})$$

where $f(x, \omega^{[1]}) \geq f(x, \omega^{[2]}) \geq \cdots \geq f(x, \omega^{[S]})$

- **Non-separable**: samples are not equally important —> only care about **tail** expectation
- $f(x, \omega^s)$ has to belong to the tail to generate a non-trivial (sub)gradient

Expectation vs Superquantile

$$\mathbb{E}[\, f(x, \omega)\,] \approx \frac{1}{S} \sum_{s=1}^{S} f(x, \omega^s)$$

- Separable: samples are equally important
- Can take an arbitrary batch of samples to unbiasedly estimate the function value and the (sub)gradient

$$\text{CVaR}_\alpha\,[\, f(x, \omega)\,] \approx \frac{1}{(1-\alpha)S} \sum_{s=1}^{(1-\alpha)S} f(x, \omega^{[s]})$$

$$\text{where } f(x, \omega^{[1]}) \geq f(x, \omega^{[2]}) \geq \cdots \geq f(x, \omega^{[S]})$$

- **Non-separable**: samples are not equally important —> only care about **tail** expectation
- $f(x, \omega^s)$ has to belong to the tail to generate a non-trivial (sub)gradient
- Function evaluations can be expensive, e.g., recourse functions, neural networks; in fact, even if $f(\,\bullet\,, \omega)$ is affine when the number of scenarios is large.

Superquantile Optimization

reduce the number of evaluations for function values and (sub)gradients

across all data

a second-order method?

Superquantile Optimization

reduce the number of evaluations for function values and (sub)gradients

across all data

a second-order method?

expensive to formulate ``Hessian" matrices + solve linear equations?

Superquantile Optimization

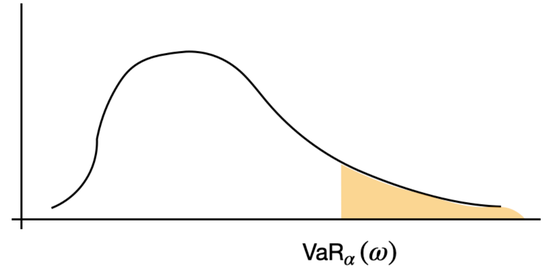
reduce the number of evaluations for function values and (sub)gradients

second order method?

cheap

~~expensive~~ to formulate “Hessian” matrices + solve linear equations!

Blessing of the tail risk: “Hessian” has structured sparsity



only a small proportion of scenarios matters

A Simplified Problem

Simplification: linear objective, one CVaR constraint, no side constraints

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & c^\top x \\ \text{subject to} \quad & CVaR_\alpha \left[A_\omega x + b_\omega \right] \leq r \end{aligned}$$

A Simplified Problem

Simplification: linear objective, one CVaR constraint, no side constraints

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & c^\top x \\ \text{subject to} \quad & CVaR_\alpha \left[A_\omega x + b_\omega \right] \leq r \end{aligned}$$

Collect empirical data $\{A_j, b_j\}_{j=1}^S$, set $k = (1-\alpha)S$ (assume to be an integer)

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & c^\top x \\ \text{subject to} \quad & \text{Top-k-sum} \left\{ A_j x + b_j \right\}_{j=1}^S \leq rk \end{aligned}$$

Lift and Split

$$\underset{x}{\text{minimize}} \quad c^\top x$$

$$\text{subject to} \quad \text{Top-k-sum} \left\{ A_j x + b_j \right\}_{j=1}^{S} \leq rk$$

$$\underset{x,\ y \in \mathbb{R}^S}{\text{minimize}} \quad c^\top x$$

$$\text{subject to} \quad y_j = A_j x + b_j, \forall j; \qquad \text{Top-k-sum} \left\{ y_j \right\}_{j=1}^{S} \leq rk$$

Augmented Lagrangian Method

$$\underset{x,\ y\in\mathbb{R}^S}{\text{minimize}} \quad c^\top x$$

$$\text{subject to} \quad y_j = A_j x + b_j,\ \forall j; \qquad \text{Top-k-sum}\left\{ y_j \right\}_{j=1}^{S} \leq rk$$

(Partial) augmented Lagrangian function:

$$\min_{x,\ y} \quad c^\top x + \lambda^\top (y - Ax - b) + \frac{\sigma}{2}\|y - Ax - b\|^2, \quad \text{where Top-k-sum}\left\{ y_j \right\}_{j=1}^{S} \leq rk$$

Augmented Lagrangian Method

$$\begin{aligned} &\underset{x,\ y\in\mathbb{R}^S}{\text{minimize}} && c^\top x \\ &\text{subject to} && y_j = A_j x + b_j,\ \forall j; \qquad \text{Top-k-sum}\left\{ y_j \right\}_{j=1}^{S} \le rk \end{aligned}$$

(Partial) augmented Lagrangian function:

$$\min_{x,\ y} \quad c^\top x + \lambda^\top (y - Ax - b) + \frac{\sigma}{2}\|y - Ax - b\|^2, \quad \text{where Top-k-sum}\left\{ y_j \right\}_{j=1}^{S} \le rk$$

eliminate y

$$\min_{x} \quad c^\top x + \frac{\sigma}{2} \left\| \Pi_{\text{Top-k-sum}(\bullet)\le kr}(Ax + b - \tilde{\lambda}/\sigma) - (Ax + b - \tilde{\lambda}/\sigma) \right\|^2$$

projection onto the tail sublevel set

Augmented Lagrangian Method

$$\min_{x} \; c^{\top}x + \frac{\sigma}{2} \left\| \Pi_{\text{Top-k-sum}(\bullet)\leq kr}(Ax + b - \tilde{\lambda}/\sigma) - (Ax + b - \tilde{\lambda}/\sigma) \right\|^2$$

Squared distance function, continuously differentiable

- Optimality condition:

$$c + \sigma A^{\top}\left[Ax + b - \tilde{\lambda}/\sigma - \Pi_{\text{Top-k-sum}(\bullet)\leq kr}\left(Ax + b - \tilde{\lambda}/\sigma\right) \right] = 0$$

Augmented Lagrangian Method

$$\min_{x} \; c^\top x + \frac{\sigma}{2} \left\| \Pi_{\text{Top-k-sum}(\bullet)\le kr}(Ax + b - \tilde{\lambda}/\sigma) - (Ax + b - \tilde{\lambda}/\sigma) \right\|^2$$

Squared distance function, continuously differentiable

- Optimality condition:

$$c + \sigma A^\top \left[Ax + b - \tilde{\lambda}/\sigma - \Pi_{\text{Top-k-sum}(\bullet)\le kr}(Ax + b - \tilde{\lambda}/\sigma) \right] = 0$$

piecewise affine

Augmented Lagrangian Method

$$\min_{x} \; c^\top x + \frac{\sigma}{2} \left\| \Pi_{\text{Top-k-sum}(\bullet)\le kr}(Ax + b - \widetilde{\lambda}/\sigma) - (Ax + b - \widetilde{\lambda}/\sigma) \right\|^2$$

continuously differentiable

- Optimality condition:

$$c + \sigma A^\top \left[Ax + b - \widetilde{\lambda}/\sigma - \Pi_{\text{Top-k-sum}(\bullet)\le kr}(Ax + b - \widetilde{\lambda}/\sigma) \right] = 0$$

piecewise affine

- Solve the equation via the semismooth (piecewise smooth) Newton method

Projection onto the tail sublevel set

$$c + \sigma A^\top \left[Ax + b - \widetilde{\lambda}/\sigma - \Pi_{\text{Top-k-sum}(\bullet)\le kr}(Ax + b - \widetilde{\lambda}/\sigma) \right] = 0$$

- Given $y^0 \in \mathbb{R}^S$, compute the projection $\Pi_{\text{Top-k-sum}(\bullet)\le kr}(y^0)$:

$$\begin{array}{ll} \underset{y \in \mathbb{R}^S}{\text{minimize}} & \frac{1}{2}\|y - y^0\|^2 \\ \text{subject to} & y_{[1]} + y_{[2]} + \cdots + y_{[k]} \le kr \end{array}$$

where $y_{[1]} \ge y_{[2]} \ge \cdots \ge y_{[S]}$

S: # total scenarios

Projection onto the tail sublevel set

$$c + \sigma A^\top \left[Ax + b - \widetilde{\lambda}/\sigma - \Pi_{\text{Top-k-sum}(\bullet)\leq kr}(Ax + b - \widetilde{\lambda}/\sigma) \right] = 0$$

- Given $y^0 \in \mathbb{R}^S$, compute the projection $\Pi_{\text{Top-k-sum}(\bullet)\leq kr}(y^0)$:

$$\begin{aligned} \underset{y\in\mathbb{R}^S}{\text{minimize}} \quad & \frac{1}{2}\|y - y^0\|^2 \\ \text{subject to} \quad & y_{[1]} + y_{[2]} + \cdots + y_{[k]} \leq kr \end{aligned}$$

where $y_{[1]} \geq y_{[2]} \geq \cdots \geq y_{[S]}$

S: # total scenarios

sort $y^0 \to y^{0\downarrow}$

$$\begin{aligned} \underset{y\in\mathbb{R}^S}{\text{minimize}} \quad & \frac{1}{2}\|y - y^{0\downarrow}\|^2 \\ \text{subject to} \quad & y_1 \geq y_2 \geq \cdots \geq y_S \\ & y_1 + y_2 + \cdots + y_k \leq kr \end{aligned}$$

Projection onto the tail sublevel set

$$c + \sigma A^\top \left[Ax + b - \widetilde{\lambda}/\sigma - \Pi_{\text{Top-k-sum}(\bullet)\le kr}(Ax + b - \widetilde{\lambda}/\sigma) \right] = 0$$

- Given $y^0 \in \mathbb{R}^S$, compute the projection $\Pi_{\text{Top-k-sum}(\bullet)\le kr}(y^0)$:

$$\begin{aligned} \underset{y\in\mathbb{R}^S}{\text{minimize}} \quad & \frac{1}{2}\|y - y^0\|^2 \\ \text{subject to} \quad & y_{[1]} + y_{[2]} + \cdots + y_{[k]} \le kr \end{aligned}$$

where $y_{[1]} \ge y_{[2]} \ge \cdots \ge y_{[S]}$

S: # total scenarios

sort $y^0 \to y^{0\downarrow}$

$$\begin{aligned} \underset{y\in\mathbb{R}^S}{\text{minimize}} \quad & \frac{1}{2}\|y - y^{0\downarrow}\|^2 \\ \text{subject to} \quad & y_1 \ge y_2 \ge \cdots \ge y_S \\ & y_1 + y_2 + \cdots + y_k \le kr \end{aligned}$$

(Strongly) Convex quadratic programs
S Variables, (S+1) constraints

Need a scalable and highly accurate solver

Projection onto the tail sublevel set

$$c + \sigma A^\top \left[Ax + b - \widetilde{\lambda}/\sigma - \Pi_{\text{Top-k-sum}(\bullet)\le kr}(Ax + b - \widetilde{\lambda}/\sigma) \right] = 0$$

- Given $y^0 \in \mathbb{R}^S$, compute the projection $\Pi_{\text{Top-k-sum}(\bullet)\le kr}(y^0)$:

$$\begin{aligned} \underset{y\in\mathbb{R}^S}{\text{minimize}} \quad & \frac{1}{2}\|y - y^0\|^2 \\ \text{subject to} \quad & y_{[1]} + y_{[2]} + \cdots + y_{[k]} \le kr \end{aligned}$$

where $y_{[1]} \ge y_{[2]} \ge \cdots \ge y_{[S]}$

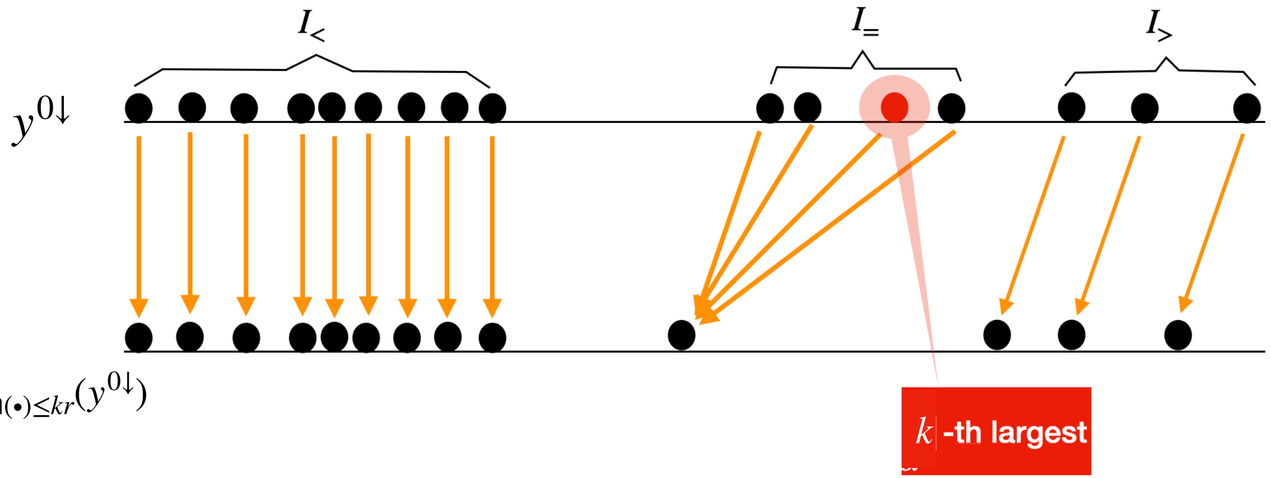
sort $y^0 \to y^{0\downarrow}$

$$\begin{aligned} \underset{y\in\mathbb{R}^S}{\text{minimize}} \quad & \frac{1}{2}\|y - y^{0\downarrow}\|^2 \\ \text{subject to} \quad & y_1 \ge y_2 \ge \cdots \ge y_S \\ & y_1 + y_2 + \cdots + y_k \le kr \end{aligned}$$

Isotonic constraints
+ one budget constraint

Projection onto the tail sublevel set

$$
\begin{aligned}
\underset{y \in \mathbb{R}^S}{\text{minimize}} \quad & \frac{1}{2}\|y - y^{0\downarrow}\|^2 \\
\text{subject to} \quad & y_1 \geq y_2 \geq \cdots \geq y_S \\
& y_1 + y_2 + \cdots + y_k \leq kr
\end{aligned}
$$



$$I_> := \{j : \bar{y}_j > \bar{y}_k\}$$

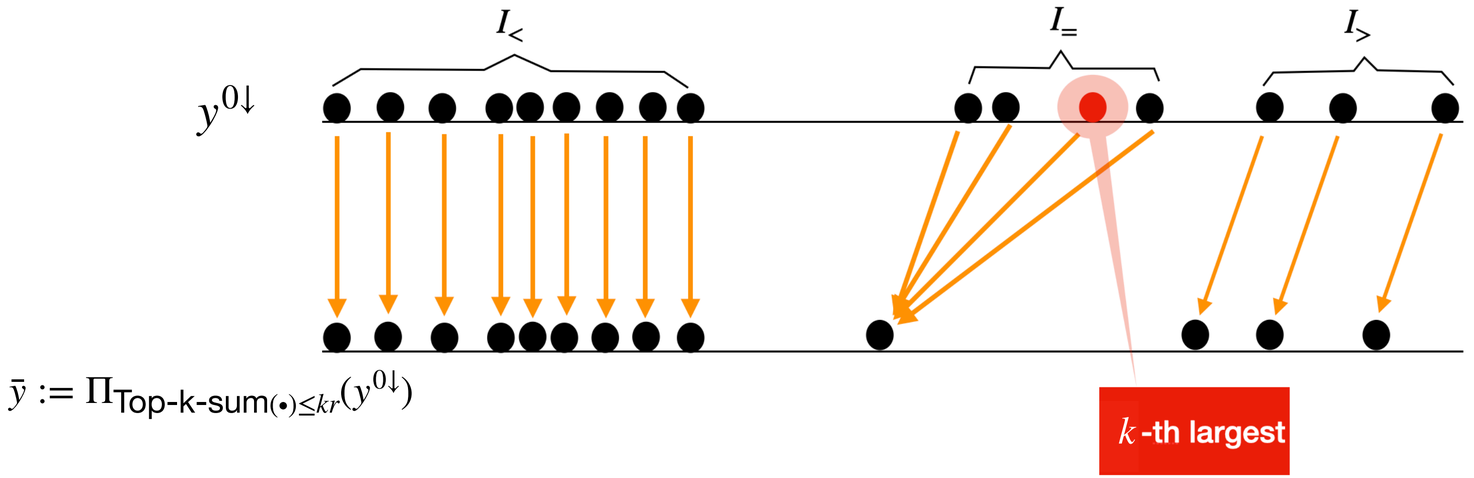
$$I_= := \{j : \bar{y}_j = \bar{y}_k\}$$

$$I_< := \{j : \bar{y}_j < \bar{y}_k\}$$

$$\bar{y} := \Pi_{\text{Top-k-sum}(\bullet) \leq kr}(y^{0\downarrow})$$

Projection onto the tail sublevel set

$$
\begin{aligned}
\underset{y \in \mathbb{R}^S}{\text{minimize}} \quad & \frac{1}{2}\|y - y^{0\downarrow}\|^2 \\
\text{subject to} \quad & y_1 \geq y_2 \geq \cdots \geq y_S \\
& y_1 + y_2 + \cdots + y_k \leq kr
\end{aligned}
$$



$$I_> := \{j : \bar{y}_j > \bar{y}_k\}$$

$$I_= := \{j : \bar{y}_j = \bar{y}_k\}$$

$$I_< := \{j : \bar{y}_j < \bar{y}_k\}$$

- Equivalent to a parametric linear complementarity problem with a **tri-diagonal Z-matrix** (off-diagonal entries are non-positive)
- Can be solved exactly with **O(S)** operations (after y^0 is sorted to $y^{0\downarrow}$)

Numerical results for the projection

	$m = 2^9$ ($\sim 10^3$)	$m = 2^{13}$ ($\sim 10^4$)	$m = 2^{17}$ ($\sim 10^5$)	$m = 2^{21}$ ($\sim 10^6$)	$m = 2^{25}$ ($\sim 10^7$)
our algorithm	**3.5e-6** (1e-6)	**7.2e-5** (2e-5)	**1.1e-3** (3e-4)	**1.2e-2** (5e-4)	**1.9e-1** (7e-3)
Gurobi	3.1e-3 (2e-4)	2.1e-1 (2e-2)	1.4e+1 (2e0)	2.9e3 (6.5e1)	>1e4 (–)
Sorting time	1.7e-5	3.8e-4	6.8e-3	1.4e-1	2.5e+0

- sorting dominates the computational time

Solve the Augmented Lagrangian Subproblem

- Each augmented Lagrangian subproblem needs to solve a piecewise affine equation

$$F(x) = c + \sigma A^\top \left[Ax + b - \widetilde{\lambda}/\sigma - \Pi_{\text{Top-k-sum}(\bullet)\le kr}\left(Ax + b - \widetilde{\lambda}/\sigma\right) \right] = 0$$

- Semismooth Newton Method (can be globalized with line-search):

$$x^{k+1} = x^k + d^k, \quad \text{where } J_F(x^k)d^k = -F(x^k)$$

- $J_F(x)$: generalized Jacobian at x

Solve the Augmented Lagrangian Subproblem

- Each subproblem needs to solve a piecewise affine equation

$$F(x) = c + \sigma A^\top \left[Ax + b - \widetilde{\lambda}/\sigma - \Pi_{\text{Top-k-sum}(\bullet)\le kr}\,(Ax + b - \widetilde{\lambda}/\sigma) \right] = 0$$

- Semismooth Newton Method:

$$x^{k+1} = x^k + d^k, \quad \text{where } J_F(x^k)d^k = -F(x^k)$$

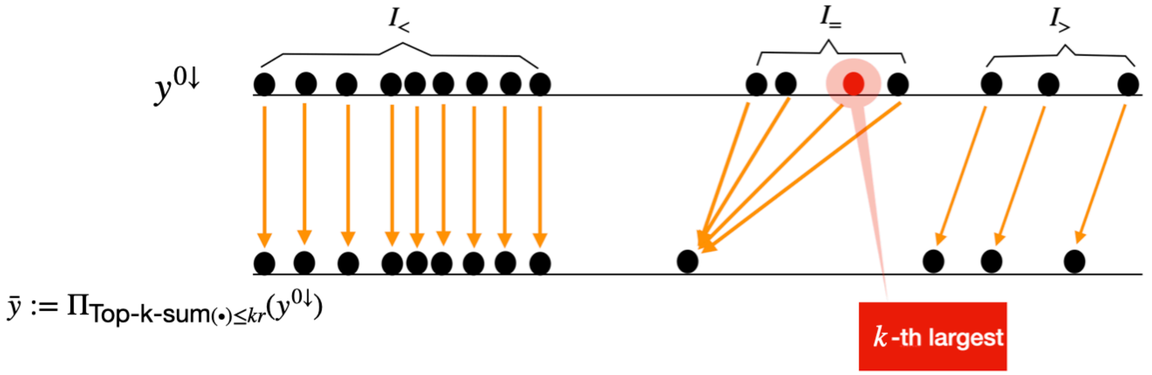
- $J_F(x)$: generalized Jacobian at x: $A^\top(I - J)A$

Solve the Augmented Lagrangian Subproblem

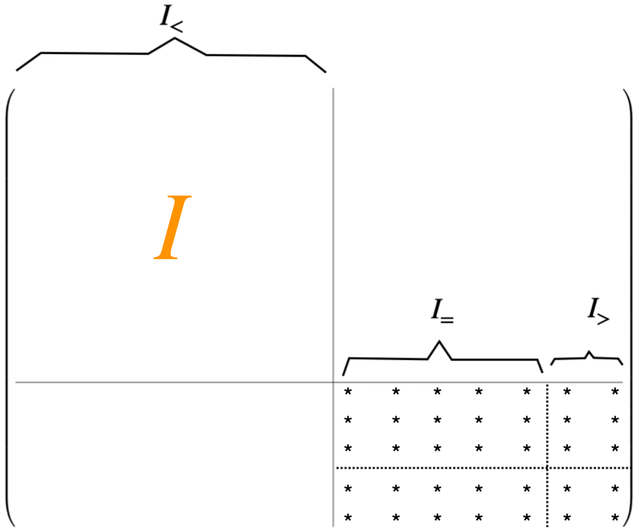
- Each subproblem needs to solve a piecewise affine equation

$$F(x) = c + \sigma A^\top \left[Ax + b - \tilde{\lambda}/\sigma - \Pi_{\text{Top-k-sum}(\bullet)\le kr}(Ax + b - \tilde{\lambda}/\sigma) \right] = 0$$

- $J_F(x)$: generalized Jacobian at x: $A^\top(I - J)A$



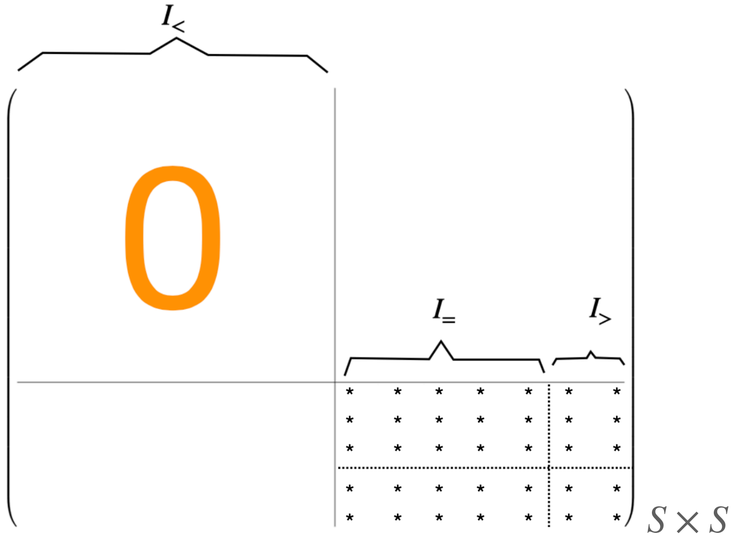
generalized Jacobian



Solve the Augmented Lagrangian Subproblem

$$F(x) = c + \sigma A^\top \left[Ax + b - \widetilde{\lambda}/\sigma - \Pi_{\text{Top-k-sum}(\bullet) \le kr} (Ax + b - \widetilde{\lambda}/\sigma) \right] = 0$$

- $J_F(x)$: generalized Jacobian at x: $A^\top (I - J) A$



$$I_> := \{j : \bar{y}_j > \bar{y}_k\}$$

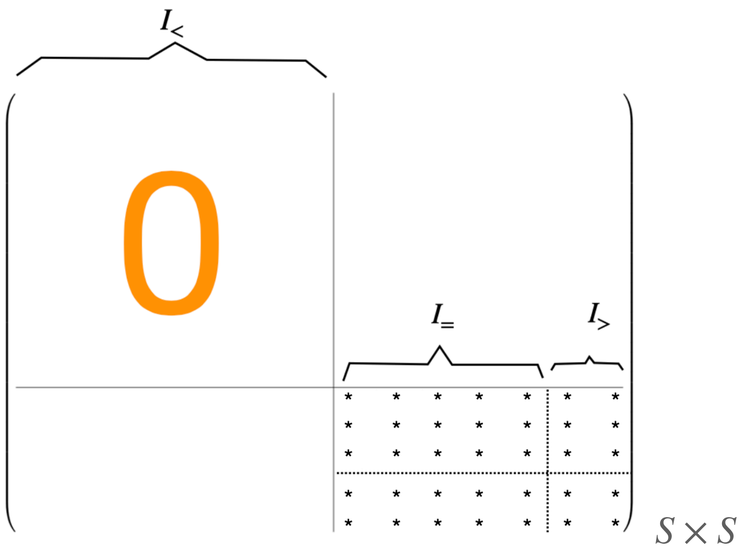
$$I_= := \{j : \bar{y}_j = \bar{y}_k\}$$

$$I_< := \{j : \bar{y}_j < \bar{y}_k\}$$

Generalized Jacobian

$$F(x) = c + \sigma A^\top \left[Ax + b - \tilde{\lambda}/\sigma - \Pi_{\text{Top-k-sum}(\bullet)\le kr}(Ax + b - \tilde{\lambda}/\sigma) \right] = 0$$

- Generalized Jacobian of F at x : $A^\top (I - J)_{S\times S}\ A_{S\times n}$



$$= \tilde{A}^\top \tilde{A}_{(|I_=|+1)\times n}$$

usually $\ll S$!!

Extracting rows from $I_>$ and $I_=$, and compute the weighted sum

$$I_> := \{j : \bar{y}_j > \bar{y}_k\}$$

$$I_= := \{j : \bar{y}_j = \bar{y}_k\}$$

$$I_< := \{j : \bar{y}_j < \bar{y}_k\}$$

Generalized Jacobian

$$F(x) = c + \sigma A^\top \left[Ax + b - \tilde{\lambda}/\sigma - \Pi_{\text{Top-k-sum}(\bullet)\le kr}\left(Ax + b - \tilde{\lambda}/\sigma\right) \right] = 0$$

- Generalized Jacobian of F at x : $A^\top (I - J)_{S\times S}\ A_{S\times n}$

$$= \tilde{A}^\top \tilde{A}_{(|I_=|+1)\times n}$$

usually $\ll S$!!

Gradient evaluation	**O(S* log(S) + S*n)**
Newton equation	**O(n (| I= | +1)^2 + (| I= | +1)^3)**

Computational cost

n: dim of x
S: # total scenarios

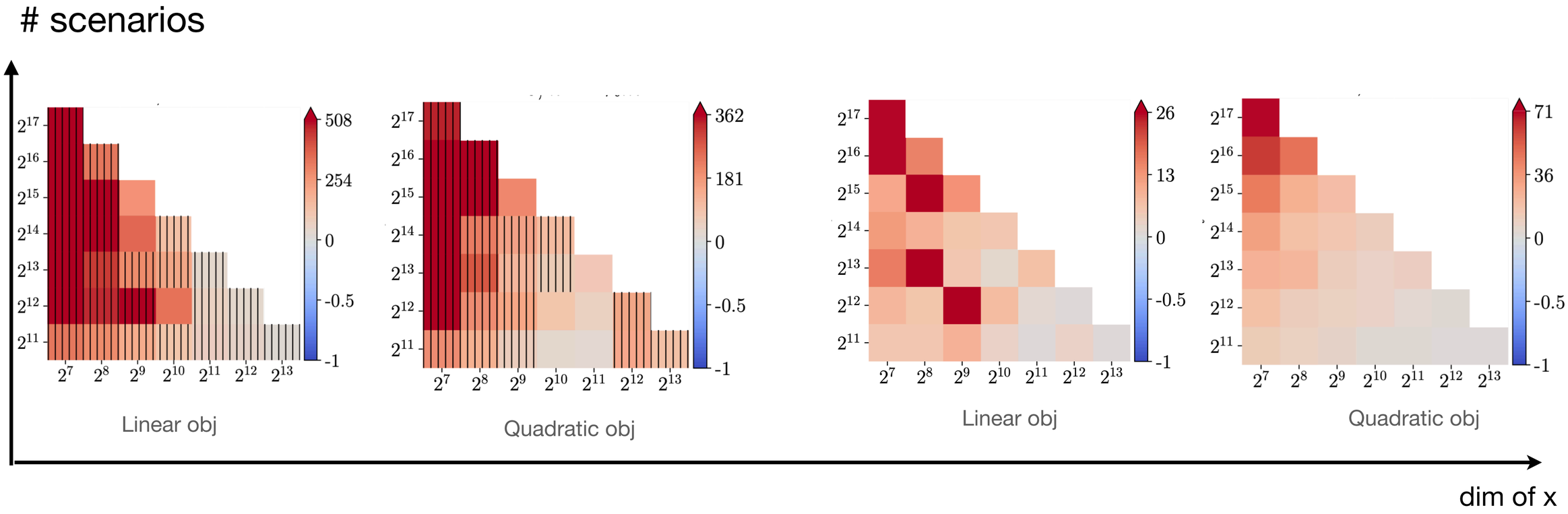
Summary

- **Outer loop:** augmented Lagrangian method
    - **Superlinear** convergence under proper error-bound assumptions

- **Inner loop:** semismooth Newton method
    - **Superlinear** convergence when the problem is non-degenerate

Gradient evaluation	**O(S* log(S) + S*n)**
Newton equation	**O(n (| I= | +1)^2 + (| I= | +1)^3)**

Comparison with OSQP & Gurobi



Compare with OSQP (a first-order ADMM)
for low-accurate solutions (1e-3)

Compare with Gurobi
for high-accurate solutions (1e-6)

Numerical Comparison with Gurobi and Portfolio Safeguard

- Compare with interior point based method (gurobi) and portfolio safeguard for solutions with high-accuracy (1e-6)

	iterations		p-infeas			d-infeas		rel-gap		walltime [sec]		
$(\mathtt{S}, n)$	ALM	GRB	ALM	GRB	PSG	ALM	GRB	ALM	GRB	ALM	GRB	PSG
$(10^5, 10^3)$	23	31	2.7e-7	2.3e-14	1.4	5.3e-12	5.6e-14	-2.6e-9	1.0e-9	27.7	2.6e2	20.9
$(10^4, 10^4)$	19	17	2.0e-7	2.6e-13	0.1	8.4e-12	3.1e-15	-1.3e-10	6.1e-8	20.8	1.6e3	1.3e3
$(10^3, 10^5)$	8	16	7.6e-7	1.2e-9	3.0e2	8.8e-12	2.0e-15	-3.9e-10	9.6e-9	6.9	2.1e2	3.8e3

ALM: our augmented Lagrangian solver | GRB: Gurobi | PSG: portfolio safeguard

Numerical Comparison with Gurobi and Portfolio Safeguard

- Compare with interior point based method gurobi and portfolio safeguard for solutions with high-accuracy (1e-6)

	iterations		p-infeas			d-infeas		rel-gap		walltime [sec]		
(S, n)	ALM	GRB	ALM	GRB	PSG	ALM	GRB	ALM	GRB	ALM	GRB	PSG
$(10^5, 10^3)$	23	31	2.7e-7	2.3e-14	1.4	5.3e-12	5.6e-14	-2.6e-9	1.0e-9	27.7	2.6e2	20.9
$(10^4, 10^4)$	19	17	2.0e-7	2.6e-13	0.1	8.4e-12	3.1e-15	-1.3e-10	6.1e-8	20.8	1.6e3	1.3e3
$(10^3, 10^5)$	8	16	7.6e-7	1.2e-9	3.0e2	8.8e-12	2.0e-15	-3.9e-10	9.6e-9	6.9	2.1e2	3.8e3

time breakdown

Gradient evaluation	Newton equation
7.3	4.2
4.2	9.4
1.4	2.9

When S is large and n is small, time spent on gradient evaluations can be longer than solving the Newton equations

Gradient evaluation	**O(S* log(S) + S*n)**
Newton equation	**O(n (| I= | +1)^2 + (| I= | +1)^3)**

A solver for superquantile constraints

Objective: convex and smooth, whose gradient is semismooth

$$\min_{x \in X} \quad \theta(x)$$

Side constraints X: convex with known projection (and its generalized Jacobian)

$$\text{s.t.} \quad \text{CVaR}_{\alpha_i}[\, f_i(x, \omega)\,] \leq r_i\,, \quad i = 1, \cdots, L\,.$$

convex and smooth in x

Our Solver

- Julia package, available at https://github.com/jacob-roth/superquantile-opt
- Currently, support the form

$$\begin{aligned} \min_{x \in X} \quad & \theta(x) \\ \text{s.t.} \quad & \text{CVaR}_{\alpha_i}[\, f_i(x, \omega)\,] \le r_i\,, \quad i = 1, \cdots, L\,. \end{aligned}$$

- In the near future:
  - More general risk functions (VaR, second-order CVaR, general optimized certainty equivalence…)
  - Allow nonconvex functions (in particular, neural networks) in objective/constraints
  - Two-/multi-stage
  - PDE constraints

Thank you!

Jake Roth and Ying Cui. On O(n) algorithms for projection onto the top-k-sum sublevel set.

Mathematical Programming Computation (2024)

Jake Roth and Ying Cui. Optimization with superquantile constraints: a fast computational approach.

Journal of Machine Learning Research (2025)