Recent Developments on (Practical) Optimization Methods for Convex and Nonconvex Optimization

GEORGIA TECH

NOVEMBER 10, 2022

Yinyu Ye
Stanford University and CUHKSZ (Sabbatical Leave)
(Currently Visiting IEDA HKUST)

Today's Talk

- New developments of ADMM-based interior point (ABIP) Method
- Optimal Diagonal Preconditioner and HDSDP
- A Dimension Reduced Trust-Region Method
- A Homogeneous Second-Order Descent Method

ABIP(Lin, Ma, Zhang and Y, 2021)

- An ADMM (Glowinski and Marroco 75, He et al. 12, Monteiro and Svaiter 13) based interior point method solver for LP problems

$$
(P) \quad \begin{array}{ll} \min & \mathbf{c}^\top \mathbf{x} \\ \text{s.t.} & A\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq 0 \end{array} \qquad\qquad (D) \quad \begin{array}{ll} \max & \mathbf{b}^\top \mathbf{y} \\ \text{s.t.} & A^\top \mathbf{y} + \mathbf{s} = \mathbf{c} \\ & \mathbf{s} \geq 0 \end{array}
$$

- Consider homogeneous and self-dual (HSD) LP here!

$$
\begin{array}{ll} \min & \beta(n+1)\theta + \mathbf{1}(\mathbf{r} = 0) + \mathbf{1}(\xi = -n-1) \\ \text{s.t.} & Q\mathbf{u} = \mathbf{v}, \\ & \mathbf{y} \text{ free}, \mathbf{x} \geq 0, \tau \geq 0, \theta \text{ free}, \mathbf{s} \geq 0, \kappa \geq 0 \end{array}
$$

where

$$
Q = \begin{bmatrix} 0 & A & -\mathbf{b} & \bar{\mathbf{b}} \\ -A^\top & 0 & \mathbf{c} & -\bar{\mathbf{c}} \\ \mathbf{b}^\top & -\mathbf{c}^\top & 0 & \bar{\mathbf{z}} \\ -\bar{\mathbf{b}}^\top & \bar{\mathbf{c}}^\top & -\bar{\mathbf{z}} & 0 \end{bmatrix}, \quad \mathbf{u} = \begin{bmatrix} \mathbf{y} \\ \mathbf{x} \\ \tau \\ \theta \end{bmatrix}, \quad \mathbf{v} = \begin{bmatrix} \mathbf{r} \\ \mathbf{s} \\ \kappa \\ \xi \end{bmatrix}, \quad \bar{\mathbf{b}} = \mathbf{b} - A\mathbf{e}, \quad \bar{\mathbf{c}} = \mathbf{c} - \mathbf{e}, \quad \bar{\mathbf{z}} = \mathbf{c}^\top \mathbf{e} + 1
$$

ABIP – Subproblem

- Introduce log-barrier function for HSD LP

$$\begin{aligned} \min \quad & B(\mathbf{u}, \mathbf{v}, \mu) \\ \text{s.t.} \quad & Q\mathbf{u} = \mathbf{v} \end{aligned}$$

where $B(\mathbf{u}, \mathbf{v}, \mu)$ barrier function

- Traditional IPM, one uses Newton's method to solve the KKT system of the above problem, the cost is too expensive when problem is large!

- Now we apply ADMM to solve it inexactly

$$\begin{aligned} \min \quad & \mathbf{1}(Q\tilde{\mathbf{u}} = \tilde{\mathbf{v}}) + B\big(\mathbf{u}, \mathbf{v}, \mu^k\big) \\ \text{s.t.} \quad & (\tilde{\mathbf{u}}, \tilde{\mathbf{v}}) = (\mathbf{u}, \mathbf{v}) \end{aligned}$$

The augmented Lagrangian function: only need to factorize a matrix once or find good diagonal preconditioners once

$$\mathcal{L}_\beta\big(\tilde{\mathbf{u}}, \tilde{\mathbf{v}}, \mathbf{u}, \mathbf{v}, \mu^k, \mathbf{p}, \mathbf{q}\big) := \mathbf{1}(Q\tilde{\mathbf{u}} = \tilde{\mathbf{v}}) + B\big(\mathbf{u}, \mathbf{v}, \mu^k\big) - \langle \beta(\mathbf{p}, \mathbf{q}), (\tilde{\mathbf{u}}, \tilde{\mathbf{v}}) - (\mathbf{u}, \mathbf{v}) \rangle + \frac{\beta}{2} \|(\tilde{\mathbf{u}}, \tilde{\mathbf{v}}) - (\mathbf{u}, \mathbf{v})\|^2$$

ADMM Based Interior-Point (ABIP)+ Method (Deng et al. 2022)

- Different strategies/parameters may be significantly different among problems being solved
- An integration strategy based on decision tree is integrated into ABIP

$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & Ax = b \\ & x \geq 0 \end{aligned}$$

Dimension

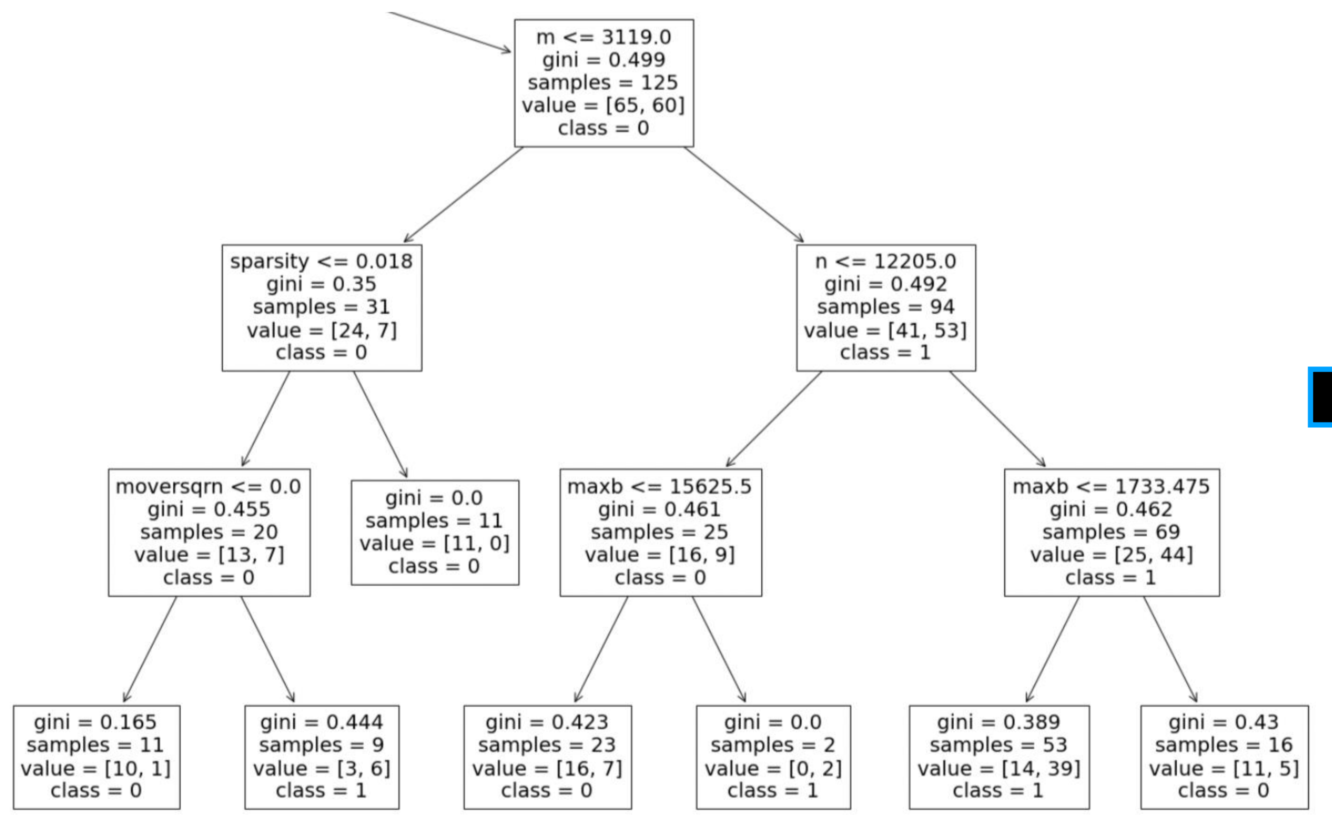
Sparsity

Constraint

Coefficient

Null Objective

…



Strategy 1 (restart)

Strategy 2 (scaling)

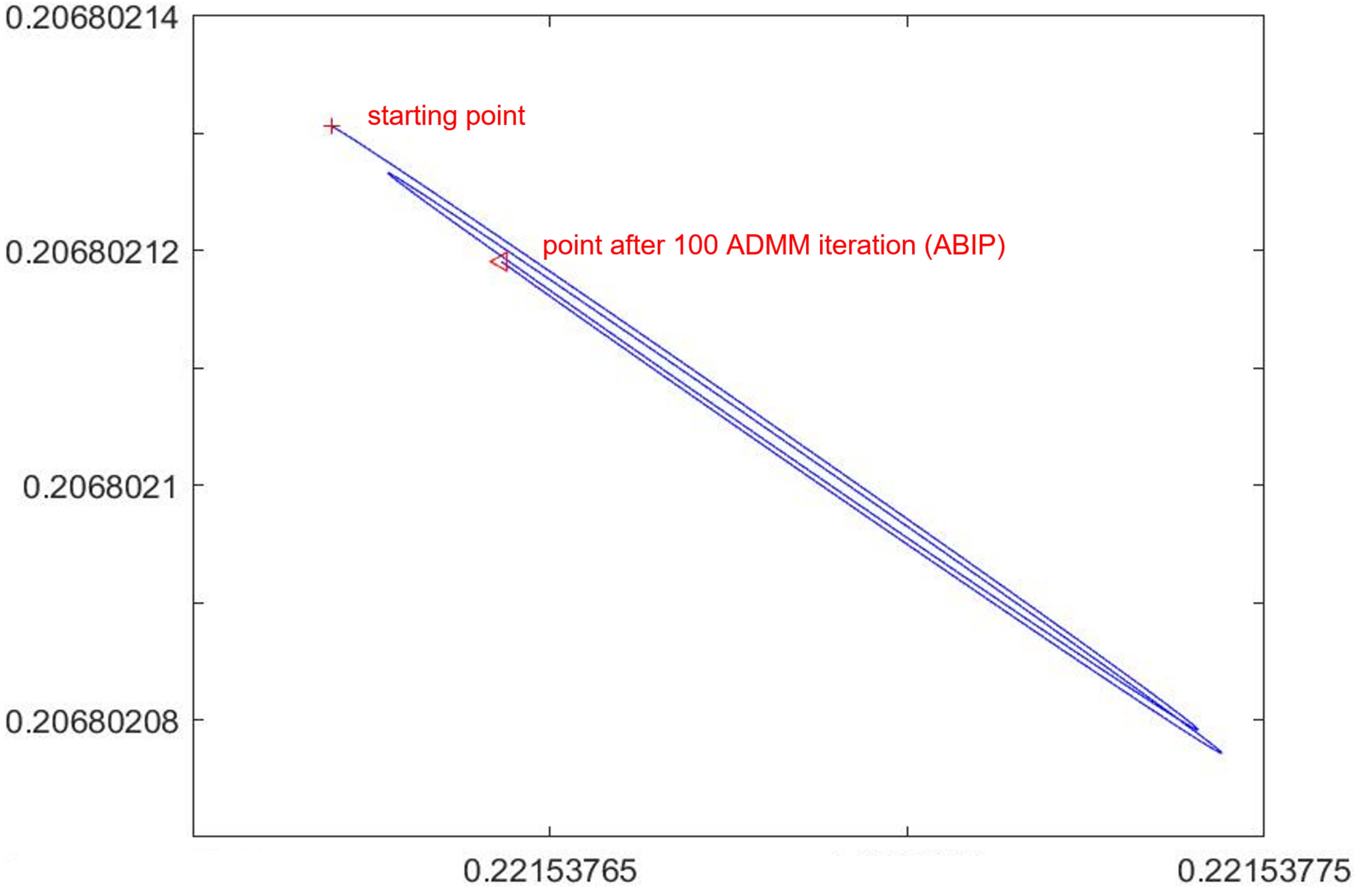
…

Strategy *k (μ reduction)*

- **A simple feature-to-strategy mapping is derived from a machine learning model**
- **For generalization limit the number of strategies (2 or 3 types)**

ABIP – Restart Strategy I

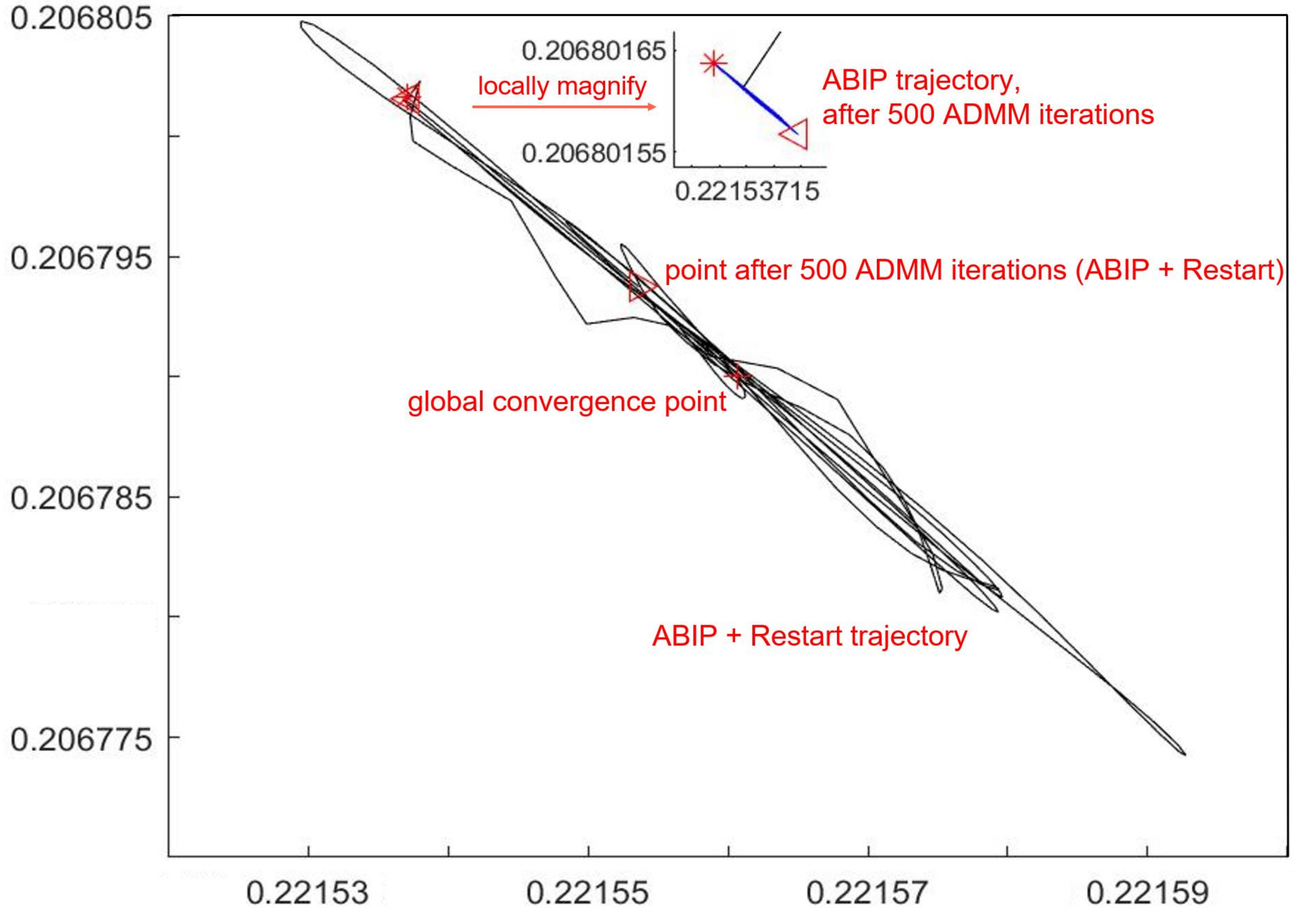
- ABIP tends to induce a spiral trajectory



Instance SC50B (only plot the first two dimension,)

ABIP – Restart Strategy II

- After restart, ABIP moves more aggressively and converges faster (reduce almost 70% ADMM iterations) !



Instance SC50B (only plot the first two dimension, after restart)

ABIP – Netlib

- Selected 105 Netlib instances
- $\epsilon = 10^{-6}$, use the direct method, 10^6 max ADMM iterations

Method	# Solved	# IPM	# ADMM	Avg.Time (s)
ABIP	65	74	265418	87.07
+ restart	68	74	88257	23.63
+ rescale	84	72	77925	20.44
+ hybrid μ (=ABIP+)	**86**	**22**	**73738**	**14.97**

- Hybrid μ : If $\mu > \epsilon$ use the aggressive strategy, otherwise use another strategy
- ABIP+ decreases both # IPM iterations and # ADMM iterations significantly

ABIP – MIP2017

- 240 MIP2017 instances
- $\epsilon = 10^{-4}$, presolved by PaPILO, use the direct method, 10^6 max ADMM iterations

Method	# Solved	SGM
COPT	**240**	**1**
PDLP(Julia)	202	17.4
ABIP	192	34.8
ABIP3+ Integration	**212**	**16.7**

- PDLP (Lu et al. 2021) is a practical first-order method (i.e., the primal-dual hybrid gradient (PDHG) method) for linear programming, and it enhences PDHG by a few implementation tricks.
- SGM stands for Shifted Geometric Mean, a standard measurement of solvers' performance

ABIP – PageRank

- 117 instances, generated from sparse matrix datasets: DIMACS10, Gleich, Newman and SNAP. Second order methods in commercial solver fail in most of these instances.

- $\epsilon = 10^{-4}$, use the indirect method, 5000 max ADMM iterations.

Method	# Solved	SGM
PDLP(Julia)	**122**	**1**
ABIP3+	119	1.31

- Examples:

Instance	# nodes	PDLP (Julia)	ABIP3+
amazon0601	403394	117.54	**71.15**
coAuthorsDBLP	299067	51.66	**24.70**
web-BerkStan	685230	447.68	**139.75**
web-Google	916428	293.01	**148.18**

ABIP – PageRank

- Generated by Google code

- When # nodes equals to # edges, the generated instance is a staircase matrix. For example,

-1.0000	0.1980	0	0	0	0	0	0	0	0
0.9900	-1.0000	0.4950	0.9900	0.4950	0.4950	0	0	0	0
0	0.1980	-1.0000	0	0	0	0.4950	0	0	0
0	0.1980	0	-1.0000	0	0	0	0	0	0
0	0.1980	0	0	-1.0000	0	0	0.9900	0	0
0	0.1980	0	0	0	-1.0000	0	0	0.9900	0
0	0	0.4950	0	0	0	-1.0000	0	0	0.9900
0	0	0	0	0.4950	0	0	-1.0000	0	0
0	0	0	0	0	0.4950	0	0	-1.0000	0
0	0	0	0	0	0	0.4950	0	0	-1.0000

Staircase matrix instance (# nodes = 10)

- In this case, ABIP+ is significantly faster than PDLP!

# nodes	PDLP (Julia)	ABIP+
10^4	8.60	**0.93**
10^5	135.67	**10.36**
10^6	2248.40	**60.32**

ABIP – Extension to Conice Linear Program

ABIP iteration remains valid for general conic linear program

$$\begin{aligned} \min\ & c^T x \\ \text{s.t. } & Ax = b \\ & x \in \mathcal{K} \end{aligned}$$

- ABIP-subproblem requires to solve a proximal mapping $x^+ = \operatorname{argmin} \lambda F(x) + \frac{1}{2}\|x - c\|^2$ with respect to the log-barrier functions $F(x)$ in $B(u, v, \mu^k)$

Positive orthant

- $F(x) = -\log(x)$
- $x = \operatorname{argmin} \lambda F(x) + \frac{1}{2}|x - c|^2$
  $= \frac{c+\sqrt{c^2+4\lambda}}{2}$

Second-order cone

- $F(\boldsymbol{x}) = -\log(t^2 - \|x\|^2)\,, \boldsymbol{x} = (t; x)$
- Can be solved by finding the root of quadratic functions

Positive semidefinite cone

- $F(\boldsymbol{x}) = -\log(\det x)$
- Equivalent to solve $-\lambda x^{-1} - c + x = 0$
- Can be solved by eigen decomposition

- The total IPM and ADMM iteration complexities of ABIP for conic linear program are respectively:

$$T_{IPM} = O\left(\log\left(\frac{1}{\varepsilon}\right)\right), \quad T_{ADMM} = O\left(\frac{1}{\varepsilon}\log\left(\frac{1}{\varepsilon}\right)\right)$$

ABIP – Numerical results for large sparse SDPs (Joachim Dahl et al . 2022

- Large sparse SDP problems from Mittelmann's library
- Relative tolerance $\epsilon = 10^{-6}$ used for stopping criteria

Name	cone dim	# constraints	# iterations	CPU time
theta12	600	17979	151	7s
theta102	500	37467	139	6s
theta123	600	90020	125	7s
hamming_8_3_4	256	16384	103	1s
hamming_9_5_6	512	53761	150	8s
fap09	174	30276	191	79s

(Performance on an AMD Ryzen 9 5900X Linux computer)

Summary

ABIP is

- a general purpose LP solver
- using ADMM to solve the subproblem
- developed with heuristics and intuitions from various strategies
- equipped with several new computational tricks
- Smart dual updates?

Today's Talk

- New developments of ADMM-based interior point (ABIP) Method
- Optimal Diagonal Preconditioner and HDSDP
- A Dimension Reduced Trust-Region Method
- A Homogeneous Second-Order Descent Method

Interior point method for SDPs

SDP is solvable in polynomial time using the interior point methods

- **Take Newton step towards the perturbed KKT system**

$$\begin{aligned} \mathcal{A}X &= b \\ \mathcal{A}^*y + S &= C \\ XS &= 0 \end{aligned} \qquad \begin{aligned} \mathcal{A}X &= b \\ \mathcal{A}^*y + S &= C \\ XS &= \mu I \end{aligned} \qquad \begin{aligned} \mathcal{A}\Delta X &= -R_P \\ \mathcal{A}^*\Delta y + \Delta S &= -R_D \\ H_P(X\Delta S + \Delta X S) &= -R_\mu \end{aligned}$$

- **Efficient numerical solvers have been developed**

  COPT, Mosek, SDPT3, SDPA, DSDP…

- **Most IPM solvers adopt primal-dual path-following IPMs except DSDP**

  DSDP (Dual-scaling SDP) implements a dual potential reduction method

Homogeneous dual-scaling algorithm

From arbitrary starting dual solution $(y, S \succ 0, \tau > 0)$ with dual residual *R*

$$
\begin{aligned}
\mathcal{A}X - b\tau &= 0 \\
-\mathcal{A}^*y + C\tau - S &= 0 \\
b^\top y - \langle C, X\rangle - \kappa &= 0
\end{aligned}
$$

$$\color{red}{X = \mu S^{-1} \qquad \kappa = \mu\tau^{-1}}$$

$$
\begin{aligned}
\mathcal{A}(X + \Delta X) - b(\tau + \Delta\tau) &= 0 \\
-\mathcal{A}^*(y + \Delta y) + C(\tau + \Delta\tau) - (S + \Delta S) &= 0 \\
\mu S^{-1}\Delta S S^{-1} + \Delta X &= \mu S^{-1} - X \\
\mu\tau^{-2}\Delta\tau + \Delta\kappa &= \mu\tau^{-1} - \kappa
\end{aligned}
$$

$$
\begin{pmatrix} \mu M & -b - \mu\mathcal{A}S^{-1}CS^{-1} \\ -b + \mu\mathcal{A}S^{-1}CS^{-1} & -\mu(\langle C, S^{-1}CS^{-1}\rangle + \tau^{-2}) \end{pmatrix}\begin{pmatrix} \Delta y \\ \Delta\tau \end{pmatrix} = \begin{pmatrix} b\tau \\ b^\top y - \mu\tau^{-1} \end{pmatrix} - \mu\begin{pmatrix} \mathcal{A}S^{-1} \\ \langle C, S^{-1}\rangle \end{pmatrix} + \mu\begin{pmatrix} \mathcal{A}S^{-1}RS^{-1} \\ \langle C, S^{-1}RS^{-1}\rangle \end{pmatrix}
$$

- **Primal iterations can still be fully eliminated**
- $S = -\mathcal{A}^*y + C\tau - R$ **inherits sparsity pattern of data**

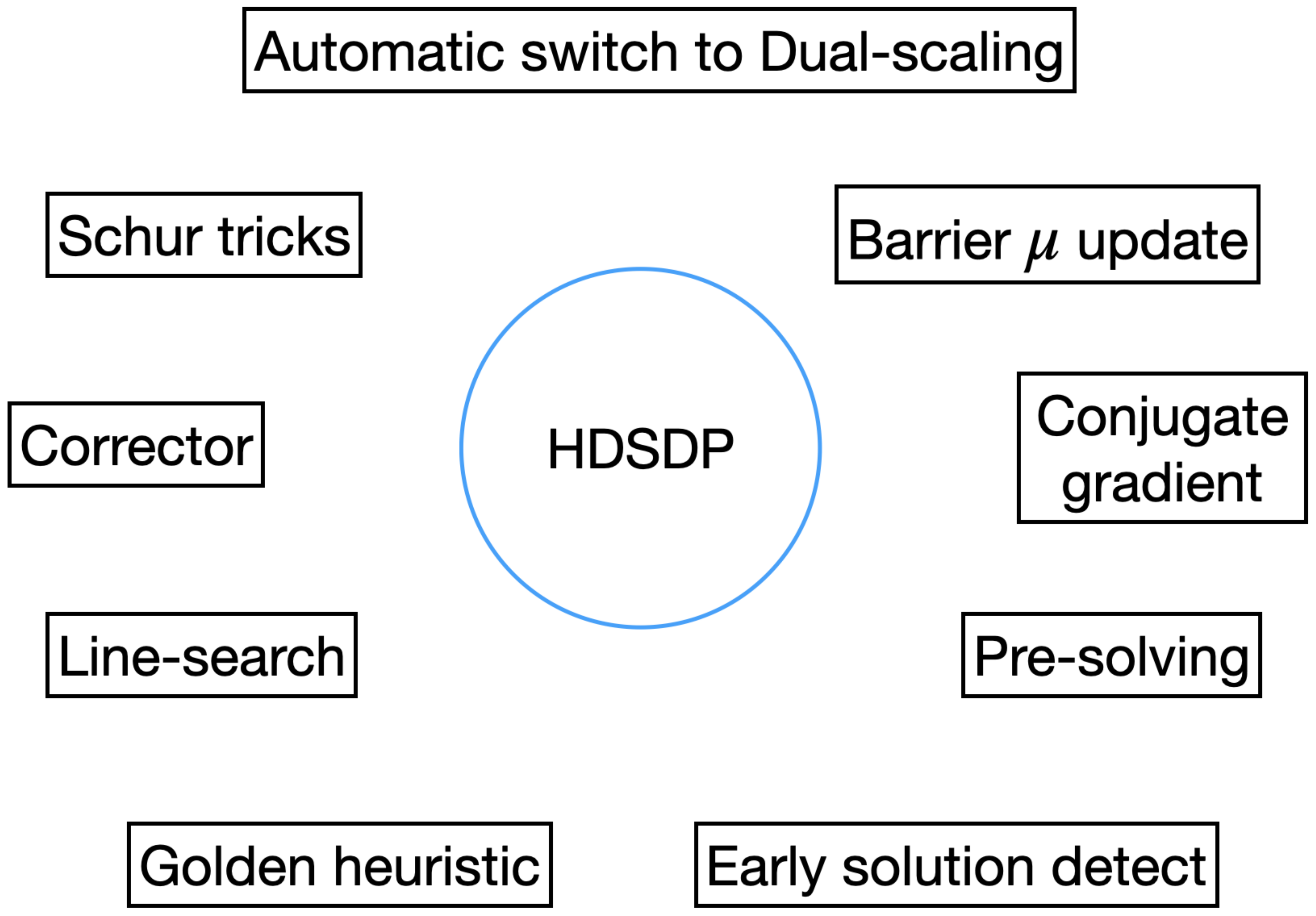
  Less memory and since X is generally dense
- **Infeasibility or an early feasible solution can be detected via the embedding**

New strategies are tailored for the method

Computational aspects for HDSDP Solver

To enhance performance, HDSDP (written in ANSI C) is equipped with

- **Pre-solving that detects special structure and dependency**
- **Line-searches over barrier to balance optimality & centrality**
- **Heuristics to update the barrier parameter μ**
- **Corrector strategy to reuse the Schur matrix**
- **A complete dual-scaling algorithm from DSDP5.8**
- **More delicate strategies for the Schur system**



Computational results

- **HDSDP is tuned and tested for many benchmark datasets**
- **Good performance on problems with both low-rank structure and sparsity**
- **Solve around 70/75 Mittelmann's benchmark problems**
- **Solve 90/92 SDPLIB problems**

Instance	DSDP5.8	HDSDP	Mosek v9	SDPT3	COPT v5
G40_mb	18	7	174	25	18
G48_mb	36	8	191	49	35
G48mc	11	2	71	24	18
G55mc	200	179	679	191	301
G59mc	347	246	646	256	442
G60_mb	700	213	7979	592	714
G60mc	712	212	8005	590	713

Instance	DSDP5.8	HDSDP	Mosek v9	SDPT3	COPT v5
checker1.5	87	41	72	71	81
foot	28	14	533	32	234
hand	4	2	76	8	40
ice_2.0	833	369	4584	484	1044
p_auss2	832	419	5948	640	721
r1_2000	17	8	333	20	187
torusg3-15	101	22	219	61	84

Selected Mittelmann's benchmark problems where HDSDP is fastest (all the constraints are rank-one)

(Results run on an intel i11700K machine)

Optimal Diagonal Pre-Conditioner [QGHYZ 20]

Given matrix $M = X^\top X \succ 0$, iterative method (e.g., CG) is often applied to solve

$$Mx = b$$

- Convergence of iterative methods depends on the condition number $\kappa(M)$
- Good performance needs pre-conditioning and we solve $P^{-1/2} M P^{-1/2} x' = b$

  A good pre-conditioner reduces $\kappa(P^{-1/2} M P^{-1/2})$
- Diagonal $P = D$ is called diagonal pre-conditioner

More generally, we wish to find D (or E) such that $\kappa(D \cdot X \cdot E)$ is minimized ?

Is it possible to find optimal D^* and E^* ?

SDP works!

Application: Optimal Diagonal Pre-Conditioner

$$\min_{D \text{ diagonal}, D \succeq 0} \kappa(DMD) \quad \Rightarrow \quad \begin{array}{rl} \min\limits_{D,\kappa} & \kappa \\ \text{subject to} & I \preceq DMD \preceq \kappa I \end{array} \quad \Rightarrow \quad \begin{array}{rl} \min\limits_{D,\kappa} & \kappa \\ \text{subject to} & D \preceq M \\ & \kappa D \succeq M \end{array}$$

$$\min_{D \text{ diagonal}, D \succeq 0} \kappa(X^T D X) \quad \Rightarrow \quad \begin{array}{rl} \min\limits_{\kappa, D \succeq 0} & \kappa \\ \text{subject to} & \kappa X^T D X \succeq I \\ & I \succeq X^T D X \end{array} \quad \Rightarrow \quad \begin{array}{rl} \max\limits_{\tau, D \succeq 0} & \tau \\ \text{subject to} & X^T D X \succeq \tau \\ & I \succeq X^T D X \end{array}$$

- Finding the optimal diagonal pre-conditioner is an SDP
- Two SDP blocks and sparse coefficient matrices
- Trivial dual interior-feasible solution
- An ideal formulation for dual SDP methods $D = \sum d_i e_i e_i^T$

What about two-sided ?

Two-Sided Pre-Conditioner

$$\min_{D_1 \succeq 0, D_2 \succeq 0} \kappa(D_1 X D_2)$$

- Common in practice and popular heuristics exist

  e.g. Ruiz-scaling, matrix equilibration & balancing

- Not directly solvable using SDP
- Can be solved by *iteratively* fixing $D_1 (D_2)$ and optimizing the other side

  Solving a sequence of SDPs

- Answer a question: how far can diagonal pre-conditioners go

Computational Results: Solving for the Optimal Pre-Conditioner

$$\begin{aligned} \min_{D,\kappa} \quad & \kappa \\ \text{subject to} \quad & D \preceq M \\ & \kappa D \succeq M \end{aligned}$$

$$\begin{aligned} \max_{\delta,d} \quad & \delta \\ \text{subject to} \quad & D - M \preceq 0 \\ & \delta M - D \preceq 0 \end{aligned}$$

SDP from optimal drag pre-conditioning problem

- Perfectly in the dual form
- Trivial dual feasible interior point solution
- 1 is an upper-bound for the optimal objective value

HDSDP

- A dual SDP algorithm (successor of DSDP5.8 by Benson)
- Support initial dual solution
- Customization for the diagonal pre-conditioner

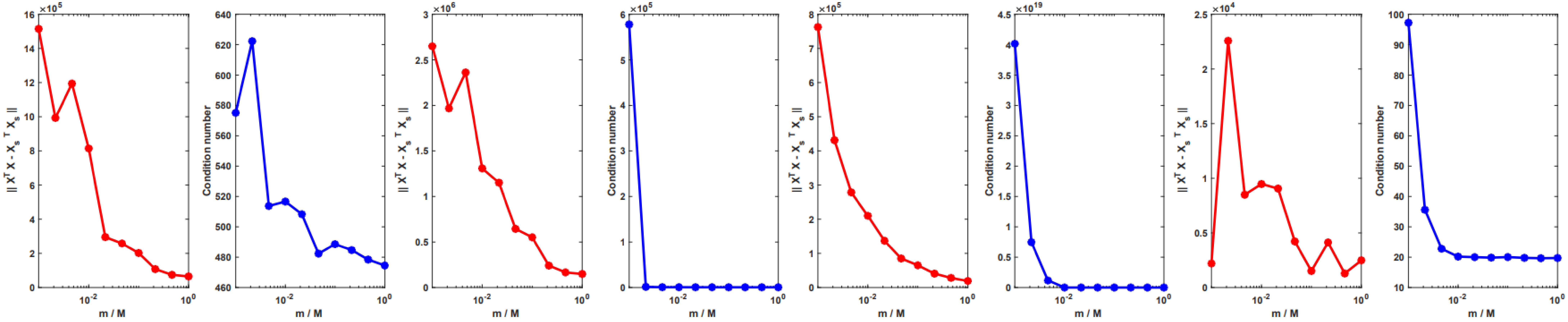
n	Sparsity	HDSDP (start from $(-10^6, 0)$)	COPT	Mosek	SDPT3
500	0.05	7.1	6.8	9.1	18.0
1000	0.09	44.5	53.9	54.2	327.0
2000	0.002	34.3	307.1	374.7	572.3
5000	0.0002	64.3	>1200	>1200	>1200

Computational results: Randomized preconditioner

- Many matrices result from statistical datasets
- $M = X^T X$ estimates the covariance matrix
- It suffices to use a few samples to approximate

How few?

As few as $\boldsymbol{O}(\mathbf{log}(\mathbf{sample}))$!



Experiment over regression datasets shows that

- It generally takes 1% to 5% of the samples to approximate well
- Scales well with dimension and saves much time for matrix-matrix multiplication

Computational Results: Optimal Diagonal Pre-Conditioner

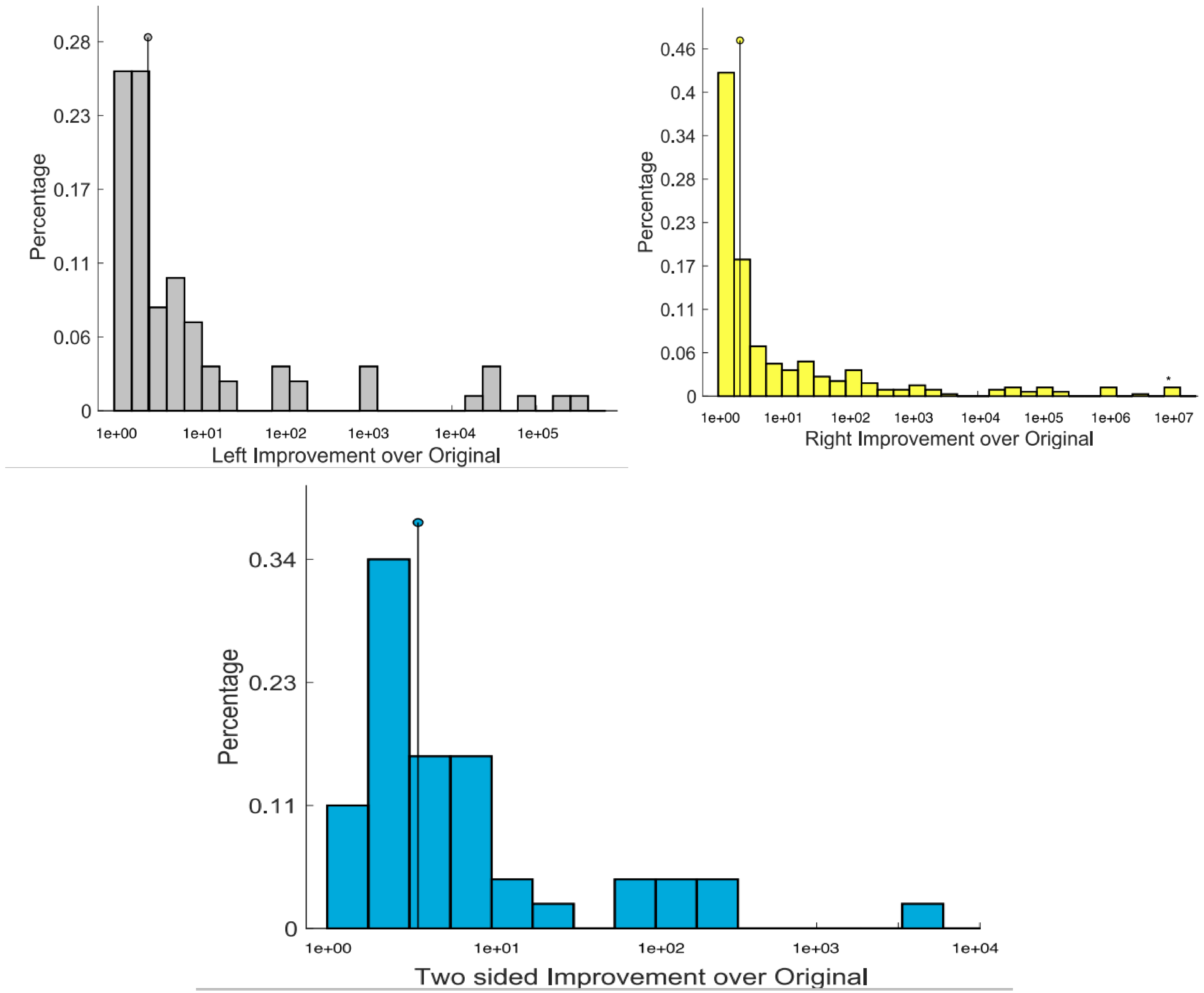
- Test over 491 Suite Sparse Matrices of fewer than 1000 columns

Reduction	Number
≥80%	121
≥50%	190
≥20%	261

Average reduction	49.7%
Better than diagonal	36.0%
Average time	1.29

- LIBSVM datasets

Mat	Size	Cbef	Caft	Reduce
YearPredictionMSD	90	5233000.00	470.20	0.999910
YearPredictionMSD.t	90	5521000.00	359900.00	0.934816
abalone_scale.txt	8	2419.00	2038.00	0.157291
bodyfat_scale.txt	14	1281.00	669.10	0.477475
cadata.txt	8	8982000.00	7632.00	0.999150
cpusmall_scale.txt	12	20000.00	6325.00	0.683813
eunite2001.t	16	52450000.00	8530.00	0.999837
eunite2001.txt	16	67300000.00	3591.00	0.999947
housing_scale.txt	13	153.90	83.22	0.459371
mg_scale.txt	6	10.67	10.03	0.059988
mpg_scale.txt	7	142.50	107.20	0.247842
pyrim_scale.txt	27	49100000.00	3307.00	0.999933
space_ga_scale.txt	6	1061.00	729.60	0.312041
triazines_scale.txt	60	24580000.00	15460000.00	0.371034



Distribution of condition number reduction (Factor of improvement)

Summary

HDSDP is

- a general purpose SDP solver
- using dual-scaling and simplified HSD
- developed with heuristics and intuitions from DSDP
- equipped with several new computational tricks
- more iterative methods for solving subproblems?

Today’s Talk

- **New developments of ADMM-based interior point (ABIP) Method**
- **Optimal Diagonal Preconditioner and HDSDP**
- **A Dimension Reduced Trust-Region Method**
- **A Homogeneous Second-Order Descent Method**

Early Complexity Analyses for Nonconvex Optimization

$$\min f(x), x \in X \ in \ \mathbb{R}^n,$$

- where f is nonconvex and twice-differentiable,

$$g_k = \nabla f(x_k), H_k = \nabla^2 f(x_k)$$

- Goal: find x_k such that:

$\| \nabla f(x_k) \| \leq \epsilon$ (primary, first-order condition)

$\lambda_{min}(H_k) \geq -\sqrt{\epsilon}$ (in active subspace, secondary, second-order condition)

- For the ball-constrained nonconvex QP: $\min \ c^T x + 0.5xTQx \ s.t. \| x \|_2 \leq 1$

  O($\log\log(\epsilon^{-1})$); see Y (1989,93), Vavasis&Zippel (1990)

- For nonconvex QP with polyhedral constraints: O(ϵ^{-1}); see Y (1998), Vavasis (2001)

Standard methods for general nonconvex optimization I

First-order Method (FOM): Gradient-Type Methods

- Assume f has L-Lipschitz cont. gradient
- Global convergence by, e.g., linear-search (LS)
- No guarantee for the second-order condition
- Worst-case complexity, $O(\epsilon^{-2})$; see the textbook by Nesterov (2004)

Each iteration requires $O(n^2)$ operations

Standard methods for general nonconvex optimization II

Second-order Method (SOM): Hessian-Type Methods

- Assume f has M-Lipschitz cont. Hessian
- Global convergence by, e.g., linear-search (LS), Trust-region (TR), or Cubic Regularization
- Convergence to second-order points
- No better than $O(\epsilon^{-2})$, for traditional methods (steepest descent and Newton); according to Cartis et al. (2010) .

Each iteration requires $O(n^3)$ operations

Analyses of SOM for general nonconvex optimization since 2000

Variants of SOM

- Trust-region with the fixed-radius strategy, $O(\epsilon^{-3/2})$, see the lecture notes by Y since 2005
- Cubic regularization, $O(\epsilon^{-3/2})$, see Nesterov and Polyak (2006), Cartis, Gould, and Toint (2011)
- A new trust-region framework, $O(\epsilon^{-3/2})$, Curtis, Robinson, and Samadi (2017)

With “slight” modification, complexity of SOM reduces from $O(\epsilon^{-2})$ to $O(\epsilon^{-3/2})$

Motivation from multi-directional FOM

- Two-directional FOM, with d_k being the momentum direction $(x_k - x_{k-1})$

$$x_{k+1} = x_k - \alpha_k^1 \nabla f(x_k) + \alpha_k^2 d_k \ = x_k + d_{k+1}$$

  where step-sizes are constructed; including CG, PT, AGD, Polyak, ADAM and many others.

- In SOM, a method typically minimizes a full dimensional quadratic Taylor expansion to obtain direction vector d_{k+1}. For example, one TR step solves for d_{k+1} from

$$\min_d \quad (g_k)^T d + 0.5 d T H_k d \quad s.t. ||\mathrm{d}||_2 \leq \Delta_k$$

  where Δ_k is the trust-region radius.

- DRSOM: Dimension Reduced Second-Order Method

  Motivation: using few directions in SOM

DRSOM I

- The DRSOM in general uses m-independent directions

$$d(\alpha) := D_k \alpha \,, D_k \in R^{nm},\ \alpha \in R^m$$

- Plug the expression into the full-dimension TR quadratic minimization problem, we minimize a m-dimension trust-region subproblem to decide "m stepsizes":

$$\min\ m_k^{\alpha}(\alpha) := (c_k)^T \alpha + \frac{1}{2} \alpha^T Q_k \alpha$$

$$||\alpha||_{G_k} \le \Delta_k$$

$$G_k = D_k^T D_k, Q_k = D_k^T H_k D_k, c_k = (g_k)^T D_k$$

How to choose D_k? How great would *m* be? Rank of H_k?
(Randomized) rank reduction of a symmetric matrix to log(n) (So et al. 08)?

DRSOM II

- In following, as an example, DRSOM adopts two FOM directions

$$d = -\alpha^1 \nabla f(x_k) + \alpha^2 d_k := d(\alpha)$$

where $g_k = \nabla f(x_k), H_k = \nabla^2 f\left(x^k\right), d_k = x_k - x_{k-1}$

- Then we minimize a 2-D trust-region problem to decide “two step-sizes”:

$$\min\ m_k^{\alpha}(\alpha) \coloneqq f(x_k) + (c_k)^T \alpha + \frac{1}{2}\alpha^T Q_k \alpha$$

$$||\alpha||_{G_k} \leq \Delta_k$$

$$G_k = \begin{bmatrix} g_k^T g_k & -g_k^T d_k \\ -g_k^T d_k & d_k^T d_k \end{bmatrix}, Q_k = \begin{bmatrix} g_k^T H_k g_k & -g_k^T H_k d_k \\ -g_k^T H_k d_k & d_k^T H_k d_k \end{bmatrix}, c_k = \begin{bmatrix} -||g_k||^2 \\ g_k^T d_k \end{bmatrix}$$

DRSOM III

DRSOM can be seen as:

- "Adaptive" **Accelerated Gradient Method** (Polyak's momentum 60)
- A second-order method minimizing quadratic model in the reduced 2-D

  $$m_k(d) = f(x_k) + \nabla f(x_k)^T d + \frac{1}{2} d^T \nabla^2 f(x_k) d, d \in \mathrm{span}\{-g_k, d_k\}$$

  compare to, e.g., Dogleg method, 2-D Newton **Trust-Region Method**

  $d \in \mathrm{span}\{g_k, [H(x_k)]^{-1} g_k\}$ (e.g., Powell 70)
- A conjugate direction method for convex optimization exploring the **Krylov Subspace** (e.g., Yuan&Stoer 95)
- For convex quadratic programming with no radius limit, terminates in n steps

Computing Hessian-Vector Product in DRSOM is the Key

In the DRSOM with two directions:

$$Q_k = \begin{bmatrix} g_k^T H_k g_k & -g_k^T H_k d_k \\ -g_k^T H_k d_k & d_k^T H_k d_k \end{bmatrix}, c_k = \begin{bmatrix} -||g_k||^2 \\ g_k^T d_k \end{bmatrix}$$

How to cheaply obtain Q? Compute $H_k g_k, H_k d_k$ first.

- Finite difference:

$$H_k \cdot v \approx \frac{1}{\epsilon}[g(x_k + \epsilon \cdot v) - g_k],$$

- Analytic approach to fit modern automatic differentiation,

$$H_k g_k = \nabla(\frac{1}{2} g_k^T g_k), H_k d_k = \nabla(d_k^T g_k),$$

- or use Hessian if readily available !

DRSOM: key assumptions and theoretical results (Zhang at al. SHUFE)

Assumption. (a) f has Lipschitz continuous Hessian. (b) DRSOM iterates with a fixed-radius strategy: $\Delta_k = \epsilon/\beta$) c) **If the Lagrangian multiplier $\boldsymbol{\lambda_k < \sqrt{\epsilon}}$, assume $\boldsymbol{\| (H_k - \widetilde{H}_k) d_{k+1} \| \leq C \| d_{k+1} \|^2}$ (Cartis et al.),** where $\widetilde{H}_k$ is the projected Hessian in the subspace (commonly adopted for approximate Hessian)

Theorem 1. If we apply DRSOM to QP, then the algorithms terminates in at most n steps to find a first-order stationary point

Theorem 2. (Global convergence rate) For *f* with second-order Lipschitz condition, DRSOM terminates in $O(\epsilon^{-3/2})$ iterations. Furthermore, the iterate x_k satisfies the first-order condition, and the Hessian is positive semi-definite in the subspace spanned by the gradient and momentum.

Theorem 3. (Local convergence rate) If the iterate x_k converges to a strict local optimum x^* such that $H(x^*) \succ 0$, and if **Assumption (c)** is satisfied as soon as $\lambda_k \leq C_\lambda \| d_{k+1} \|$, then DRSOM has a local superlinear (quadratic) speed of convergence, namely: $\| x_{k+1} - x^* \| = O(\| x_k - x^* \|^2)$

Sensor Network Location (SNL)

- Consider Sensor Network Location (SNL)

$$N_x = \{(i,j) : \|x_i - x_j\| = d_{ij} \le r_d\}, N_a = \{(i,k) : \|x_i - a_k\| = d_{ik} \le r_d\}$$

where r_d is a fixed parameter known as the radio range. The SNL problem considers the following QCQP feasibility problem,

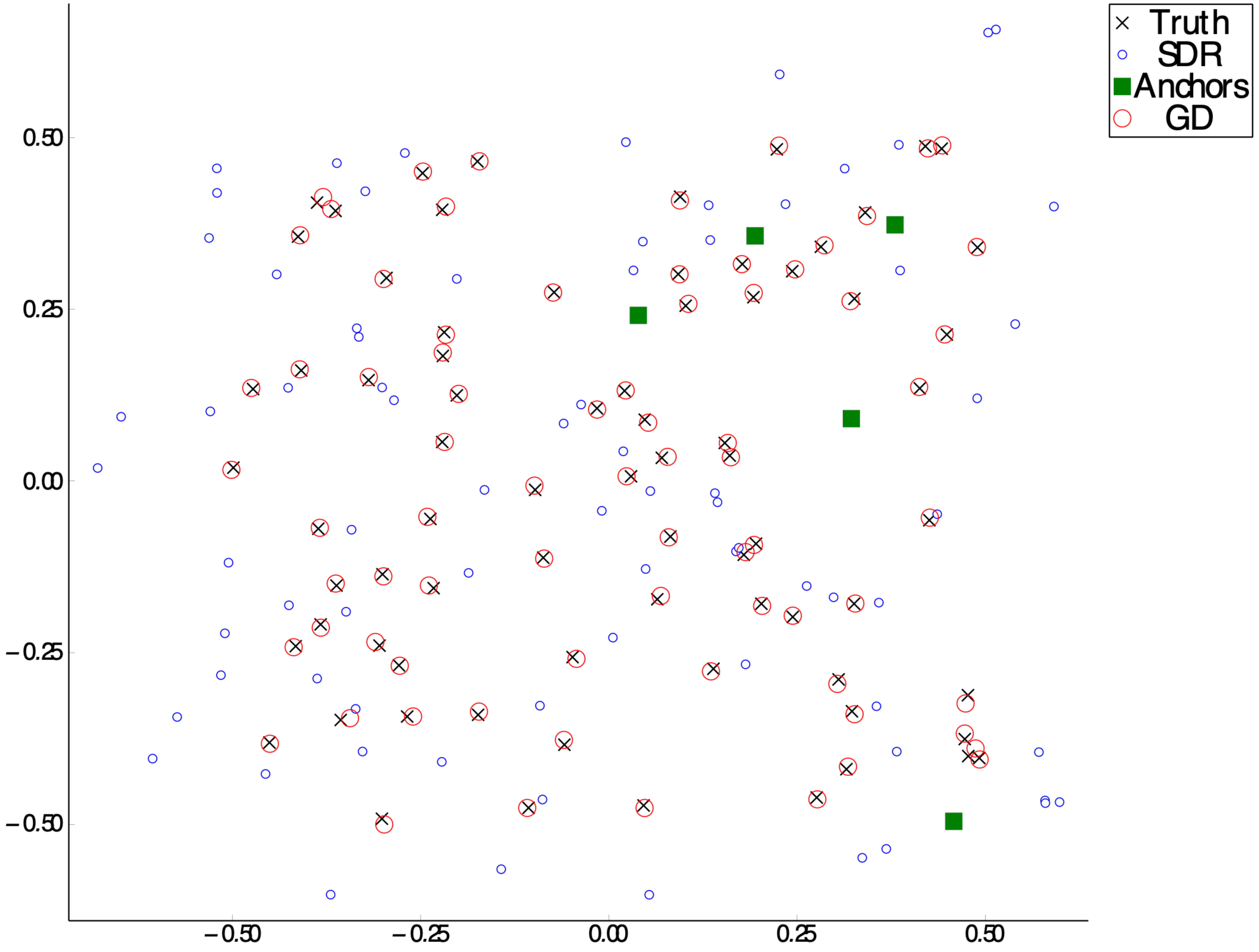
$$\|x_i - x_j\|^2 = d_{ij}^2, \forall (i,j) \in N_x$$
$$\|x_i - a_k\|^2 = \bar{d}_{ik}^2, \forall (i,k) \in N_a$$

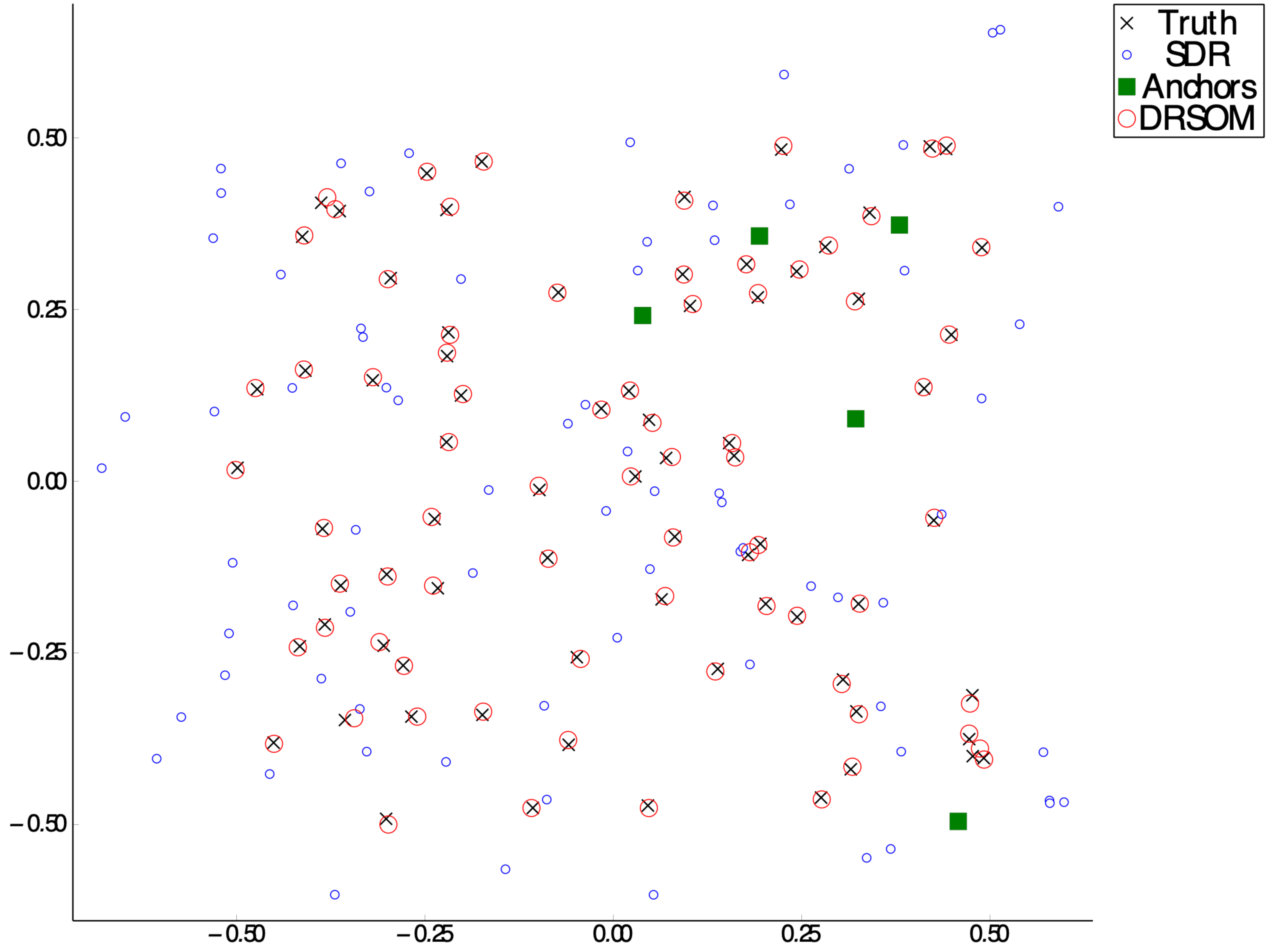
- We can solve SNL by the nonconvex nonlinear least square (NLS) problem

$$\min_X \sum_{(i<j,j)\in N_x} (\|x_i - x_j\|^2 - d_{ij}^2)^2 + \sum_{(k,j)\in N_a} (\|a_k - x_j\|^2 - \bar{d}_{kj}^2)^2.$$

Sensor Network Location (SNL)

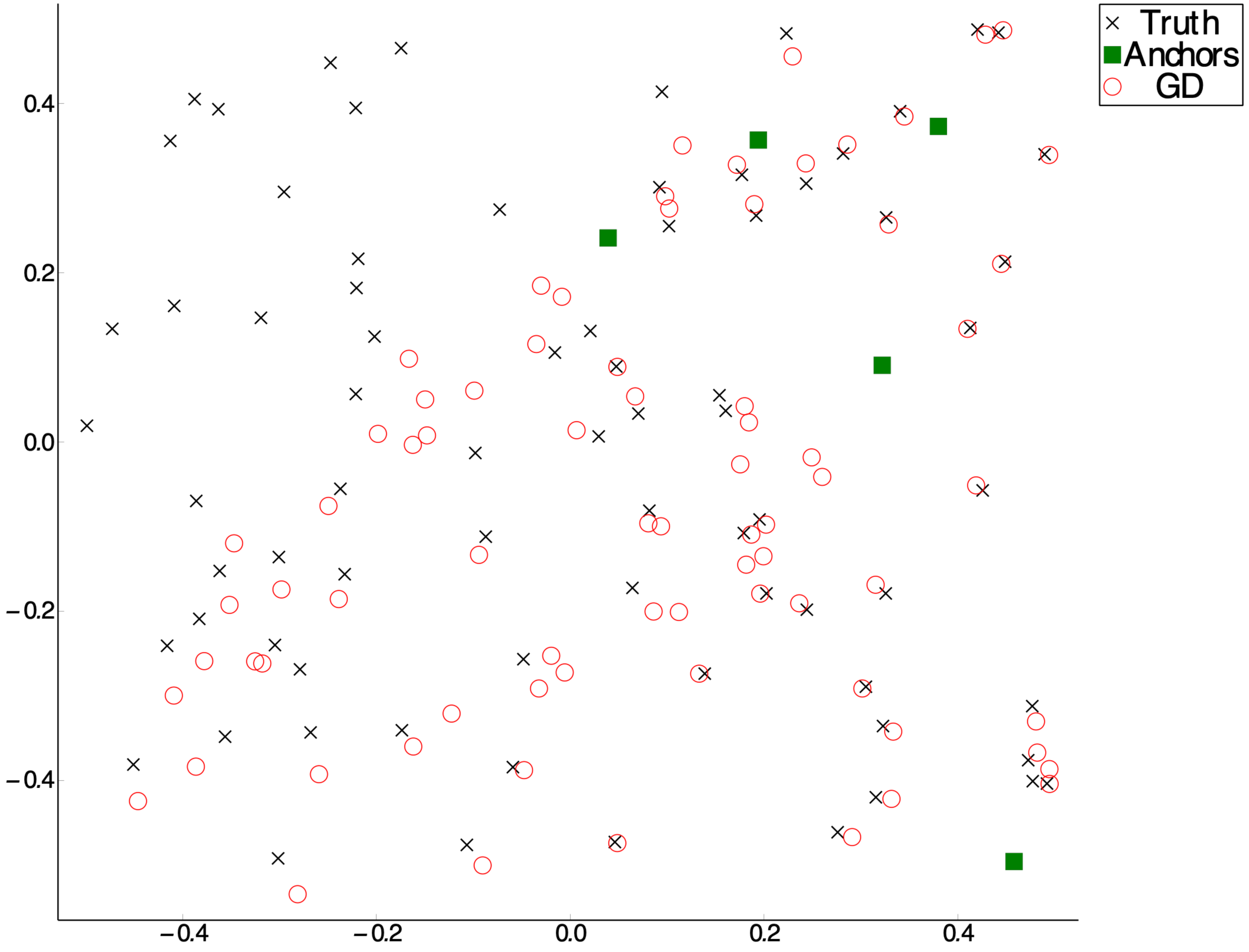
- Graphical results using SDP relaxation to initialize the NLS
- n = 80, m = 5 (anchors), radio range = 0.5, degree = 25, noise factor = 0.05
- Both Gradient Descent and DRSOM can find good solutions !

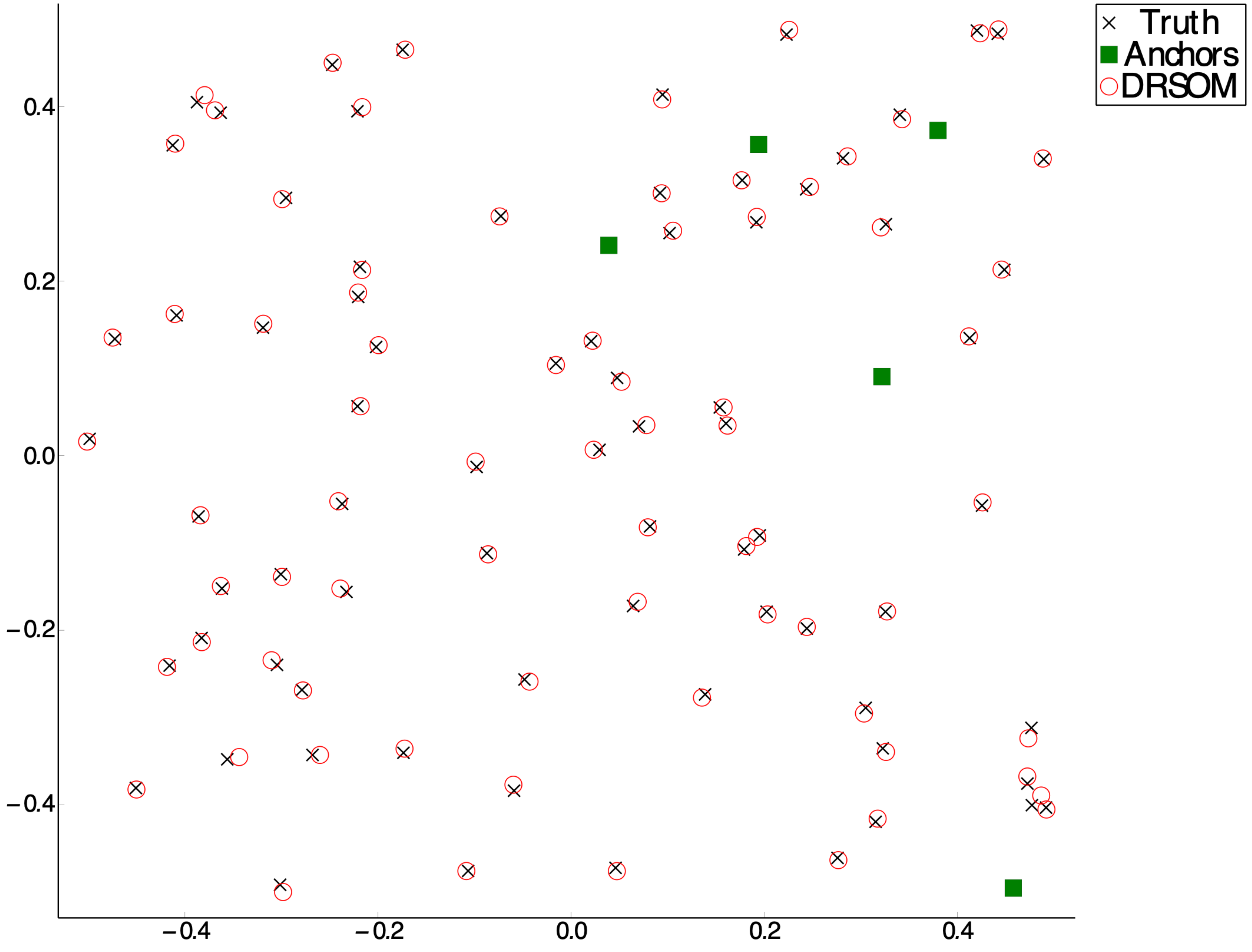




Sensor Network Location (SNL)

- Graphical results without SDP relaxation
- DRSOM can still converge to optimal solutions

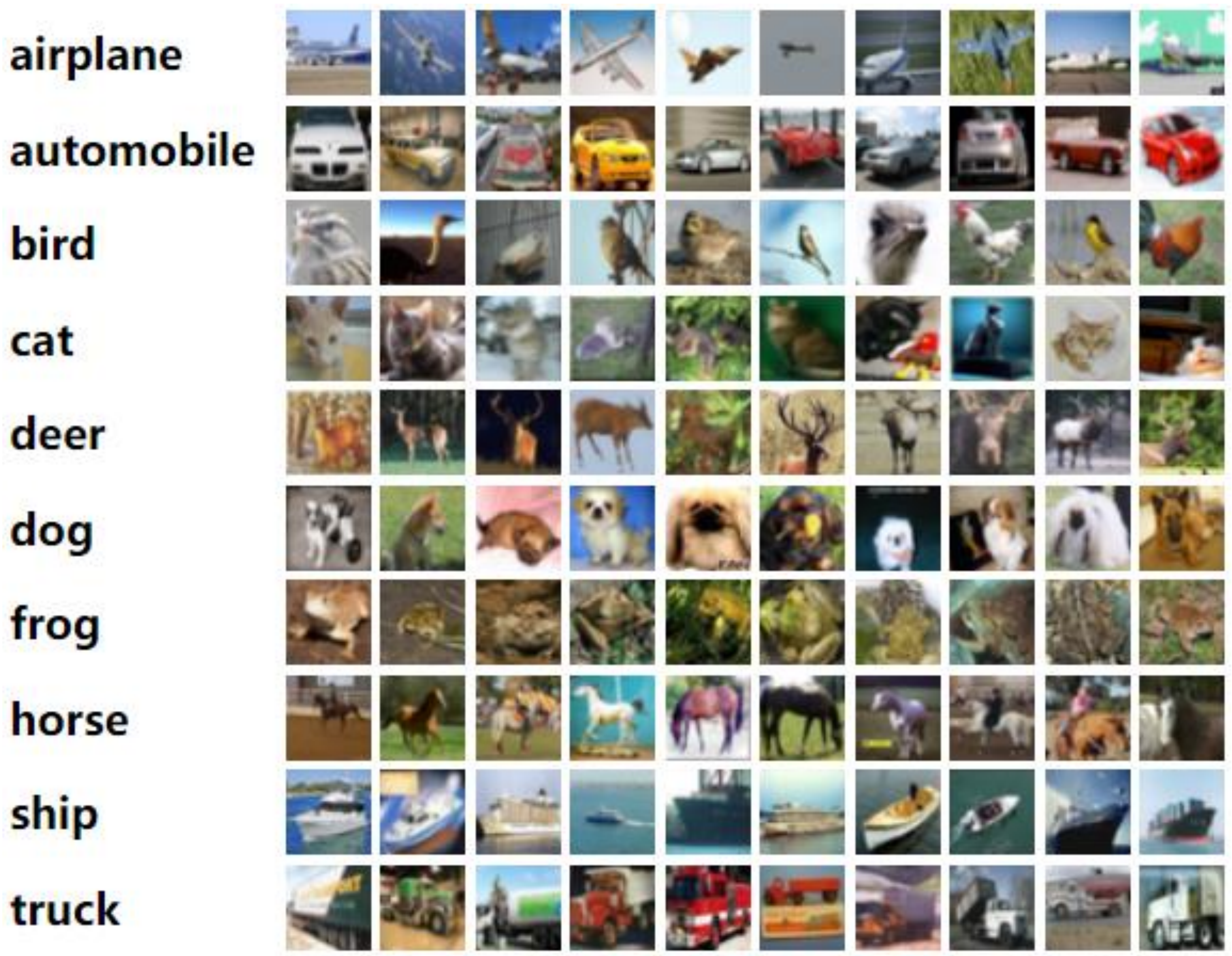




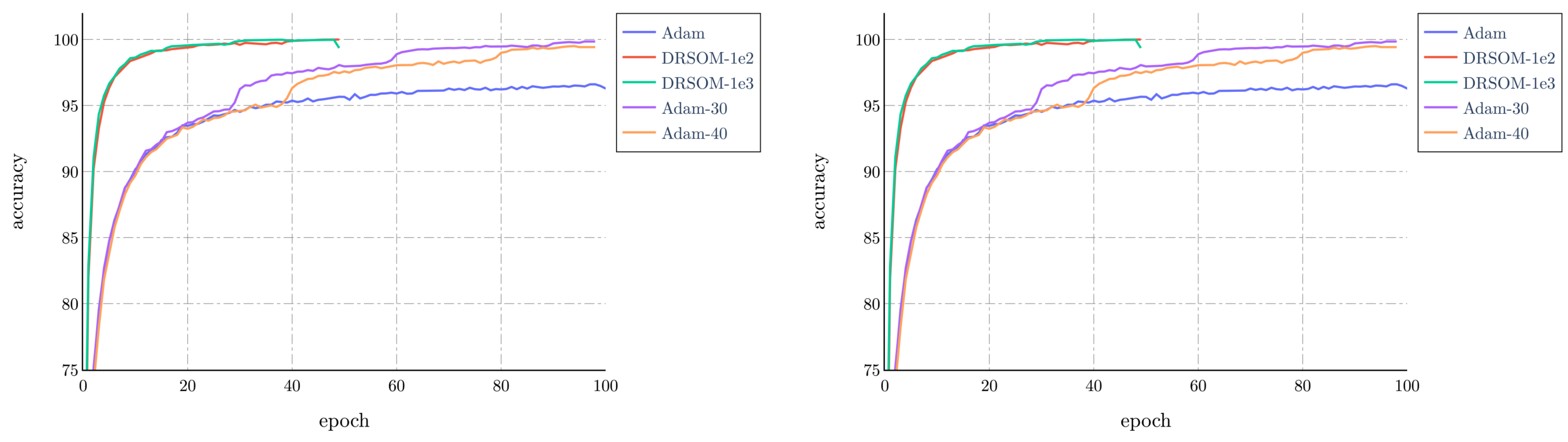
Neural Networks and Deep Learning

To use DRSOM in machine learning problems

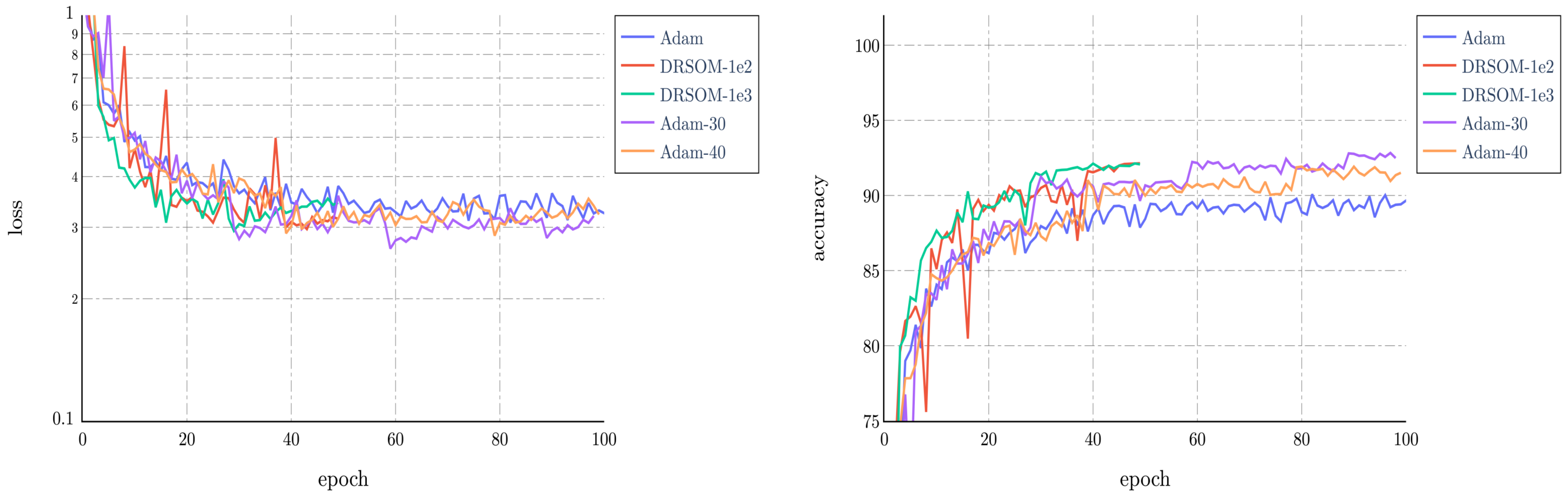
- We apply the mini-batch strategy to a vanilla DRSOM
- Use Automatic Differentiation to compute gradients
- Train ResNet18 Model with CIFAR 10
- Set Adam with initial learning rate 1e-3



Neural Networks and Deep Learning



Training results for ResNet18 with DRSOM and Adam



Test results for ResNet18 with DRSOM and Adam

Pros

- DRSOM has rapid convergence (30 epochs)
- DRSOM needs little tuning

Cons

- DRSOM may overfit the models
- Needs 4~5x time than Adam to run same number of epoch

Good potential to be a standard optimizer for deep learning!

DRSOM for TRPO I (Xue et al. SHUFE)

- **TRPO** attempts to optimize a surrogate function (based on the current iterate) of the objective function while keep a KL divergence constraint

$$\begin{aligned} \max_\theta \quad & L_{\theta_k}(\theta) \\ \text{s.t.} \quad & \mathrm{KL}\left(\mathrm{Pr}_\mu^{\pi_{\theta_k}} \,\|\, \mathrm{Pr}_\mu^{\pi_\theta}\right) \le \delta \end{aligned}$$

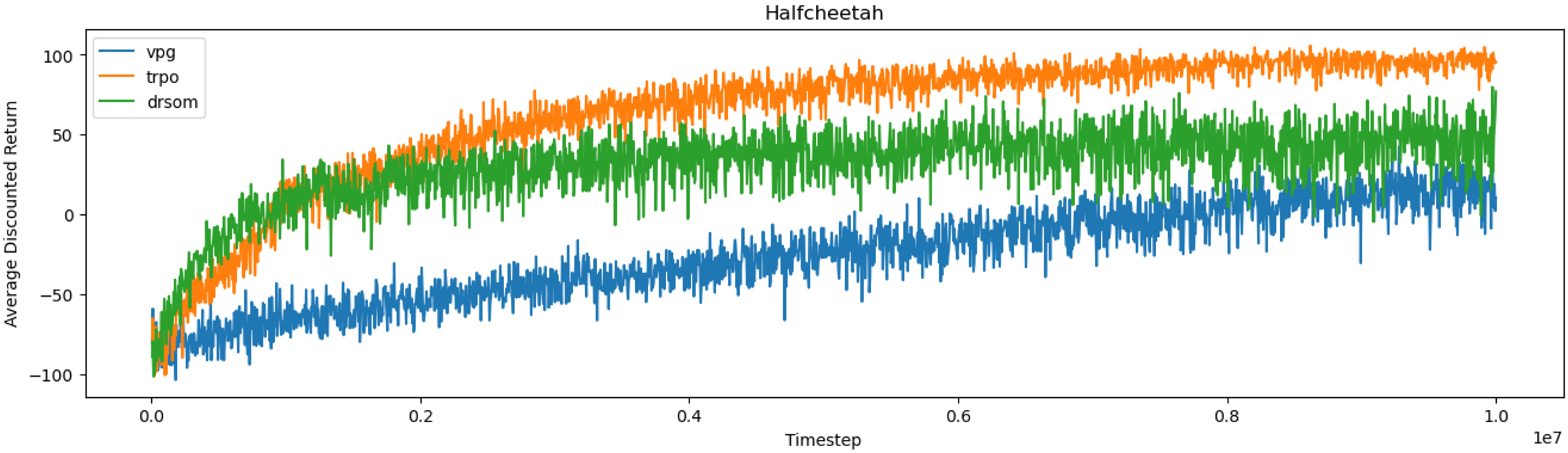
- In practice, it linearizes the surrogate function, quadratizes the KL constraint, and obtain

$$\begin{aligned} \max_\theta \quad & g_k^T(\theta - \theta_k) \\ \text{s.t.} \quad & \frac{1}{2}(\theta - \theta_k)^T F_k (\theta - \theta_k) \le \delta \end{aligned}$$

where F_k is the Hessian of the KL divergence.

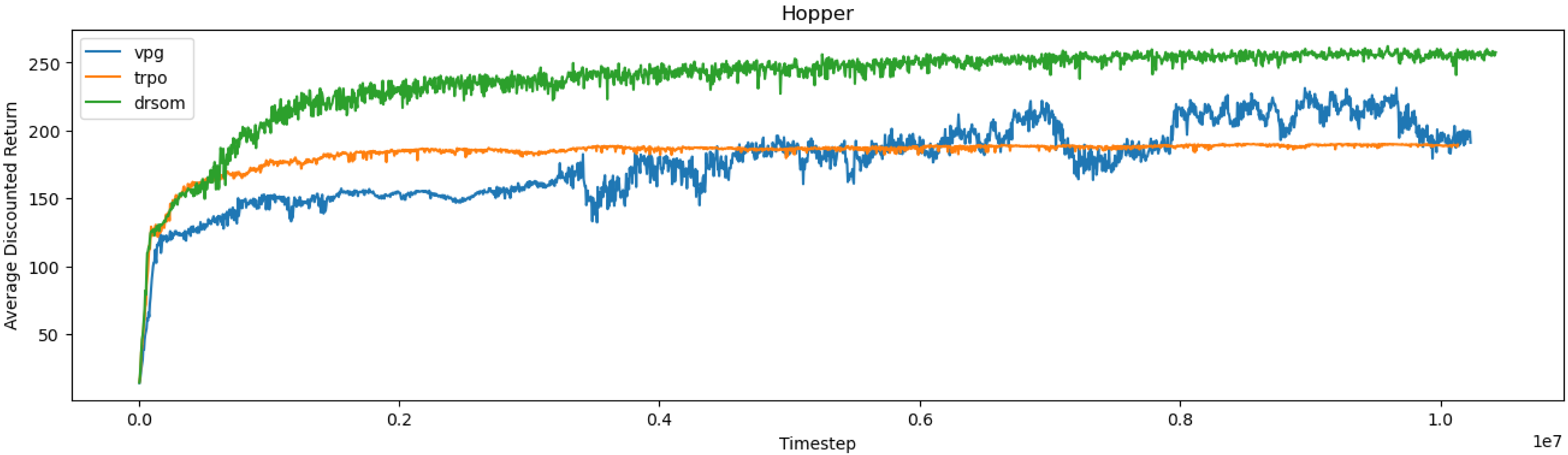
DRSOM/TRPO Preliminary Results I

- Although we only maintain the linear approximation of the surrogate function, surprisingly the algorithm works well in some RL environments



DRSOM/TRPO Preliminary Results II

- Sometimes even better than TRPO !



DRSOM for LP Potential Reduction (Gao et al. SHUFE)

We consider a simplex-constrained QP model

$$\begin{array}{ll} \min\limits_{x} & \frac{1}{2}\|Ax\|^2 \quad =: f(x) \\ \text{subject to} & e^\top x = 1 \\ & x \geq 0 \end{array}$$

$$\begin{array}{rcl} Ax - b\tau & = & 0 \\ -A^\top y - s + c\tau & = & 0 \\ b^\top y - c^\top x - \kappa & = & 0 \\ e_n^\top x + e_n^\top s + \kappa + \tau & = & 1 \end{array}$$

The self-dual embedding builds a bridge

We wish to solve a standard LP (and its dual)

$$\begin{array}{ll} \min\limits_{x} & c^\top x \\ \text{subject to} & Ax = b \\ & x \geq 0 \end{array}$$

$$\begin{array}{ll} \max\limits_{y,s} & b^\top y \\ \text{subject to} & A^\top y + s = c \\ & s \geq 0 \end{array}$$

- The homogeneous QP seems so restrictive!

Then we define the (nonconvex) potential function and apply DRSOM to it

- How to solve much more general LPs?

$$\phi(x) := \rho \log(f(x)) - \sum_{i=1}^{n} \log x_i$$

$$\nabla \phi(x) = \frac{\rho \nabla f(x)}{f(x)} - X^{-1} e$$

$$\nabla^2 \phi(x) = -\frac{\rho \nabla f(x) \nabla f(x)^\top}{f(x)^2} + \rho \frac{A^\top A}{f(x)} + X^{-2}$$

Combined with scaled gradient(Hessian) projection, the method solves LPs

DR-Potential Reduction: Preliminary Results

One feature of the DR-Potential reduction is the use of negative curvature of

$$\nabla^2 \phi(x) = -\frac{\rho \nabla f(x) \nabla f(x)^\top}{f(x)^2} + \rho \frac{A^\top A}{f(x)} + X^{-2}$$

- Computable using Lanczos iteration
- Getting LPs to high accuracy $10^{-6} \sim 10^{-8}$ if negative curvature is efficiently computed

Problem	PInfeas	DInfeas.	Compl.	Problem	PInfeas	DInfeas.	Compl.
ADLITTLE	1.347e-10	2.308e-10	2.960e-09	KB2	5.455e-11	6.417e-10	7.562e-11
AFIRO	7.641e-11	7.375e-11	3.130e-10	LOTFI	2.164e-09	4.155e-09	8.663e-08
AGG2	3.374e-08	4.859e-08	6.286e-07	MODSZK1	1.527e-06	5.415e-05	2.597e-04
AGG3	2.248e-05	1.151e-06	1.518e-05	RECIPELP	5.868e-08	6.300e-08	1.285e-07
BANDM	2.444e-09	4.886e-09	3.769e-08	SC105	7.315e-11	5.970e-11	2.435e-10
BEACONFD	5.765e-12	9.853e-12	1.022e-10	SC205	6.392e-11	5.710e-11	2.650e-10
BLEND	2.018e-10	3.729e-10	1.179e-09	SC50A	1.078e-05	6.098e-06	4.279e-05
BOEING2	1.144e-07	1.110e-08	2.307e-07	SC50B	4.647e-11	3.269e-11	1.747e-10
BORE3D	2.389e-08	5.013e-08	1.165e-07	SCAGR25	1.048e-07	5.298e-08	1.289e-06
BRANDY	2.702e-05	7.818e-06	1.849e-05	SCAGR7	1.087e-07	1.173e-08	2.601e-07
CAPRI	7.575e-05	4.488e-05	4.880e-05	SCFXM1	4.323e-06	5.244e-06	8.681e-06
E226	2.656e-06	4.742e-06	2.512e-05	SCORPION	1.674e-09	1.892e-09	1.737e-08
FINNIS	8.577e-07	8.367e-07	1.001e-05	SCTAP1	5.567e-07	8.430e-07	5.081e-06
FORPLAN	5.874e-07	2.084e-07	4.979e-06	SEBA	2.919e-11	5.729e-11	1.448e-10
GFRD-PNC	4.558e-05	1.052e-05	4.363e-05	SHARE1B	3.367e-07	1.339e-06	3.578e-06
GROW7	1.276e-04	4.906e-06	1.024e-04	SHARE2B	2.142e-04	2.014e-05	6.146e-05
ISRAEL	1.422e-06	1.336e-06	1.404e-05	STAIR	5.549e-04	8.566e-06	2.861e-05
STANDATA	5.645e-08	2.735e-07	5.130e-06	STANDGUB	2.934e-08	1.467e-07	2.753e-06
STOCFOR1	6.633e-09	9.701e-09	4.811e-08	VTP-BASE	1.349e-10	5.098e-11	2.342e-10

- Now solving small and medium Netlib instances in 10 seconds

  within 1000 iterations
- In MATLAB and getting transferred into C for acceleration

DRSOM for Riemannian Optimization (Tang et al. NUS)

$$\min_{x\in\mathcal{M}} \quad f(x) \qquad \text{(ROP)}$$

- $\mathcal{M}$ is a Riemannian manifold embeded in Euclidean space $\mathbb{R}^n$.
- $f : \mathbb{R}^n \to \mathbb{R}$ is a second-order continuously differentiable function that is lower bounded in $\mathcal{M}$.

R-DRSOM: Choose an initial point $x_0 \in \mathcal{M}$, set $k = 0$, $p_{-1} = 0$;
for $k = 0, 1, \ldots, T$ **do**
Step 1. Compute $g_k = \mathrm{grad} f(x_k)$, $d_k = \mathrm{T}_{x_k \leftarrow x_{k-1}}(p_{k-1})$, $H_k g_k = \mathrm{Hess} f(x_k)[g_k]$ and $H_k d_k = \mathrm{Hess} f(x_k)[d_k]$;
Step 2. Compute the vector $c_k = \begin{bmatrix} -\langle g_k, g_k\rangle_{x_k} \\ \langle g_k, d_k\rangle_{x_k} \end{bmatrix}$ and the following matrices

$$Q_k = \begin{bmatrix} \langle g_k, H_k g_k\rangle_{x_k} & \langle -d_k, H_k g_k\rangle_k \\ \langle -d_k, H_k g_k\rangle_{x_k} & \langle d_k, H_k d_k\rangle_{x_k} \end{bmatrix}, \ , \ G_k := \begin{bmatrix} \langle g_k, g_k\rangle_{x_k} & -\langle d_k, g_k\rangle_{x_k} \\ -\langle d_k, g_k\rangle_{x_k} & \langle d_k, d_k\rangle_{x_k} \end{bmatrix}.$$

Step 3. Solve the following 2 by 2 trust region subproblem with radius $\triangle_k > 0$

$$\alpha_k := \arg \min_{\|\alpha_k\|_{G_k} \le \triangle_k} f(x_k) + c_k^\top \alpha + \frac{1}{2}\alpha^\top Q_k \alpha;$$

Step 4. $x_{k+1} := \mathcal{R}_{x_k}\left(x_k - \alpha_k^1 g_k + \alpha_k^2 d_k\right)$;
end
Return x_k.

Max-CUT SDP

$$\textbf{Max-Cut:}\ \min\left\{-\langle L, X\rangle:\ \operatorname{diag}(X) = e,\ X \in \mathbb{S}^n_+\right\}. \tag{1}$$

$$\min\left\{-\left\langle L, RR^\top\right\rangle:\ \operatorname{diag}(RR^\top) = e,\ R \in \mathbb{R}^{n\times r}\right\}. \tag{2}$$

g67	Fval	-30977.7	-30977.7	-30977.7	-30977.7	-30977.7
n=10000	Residue	1.3e-10	2.4e-10	9.7e-10	2.6e-10	8.3e-09
m=20000	Time [s]	131.0	1371.4	177.8	1114.4	356.9
g70	Fval	-39446.1	-39446.1	-39446.1	-39446.1	-39446.1
n=10000	Residue	2.2e-10	3.7e-12	1.6e-09	2.3e-10	3.4e-09
m=9999	Time [s]	36.2	288.4	63.5	250.8	100.7
g72	Fval	-31234.2	-31234.2	-31234.2	-31234.2	-31234.2
n=10000	Residue	8.2e-11	1.8e-12	5.8e-10	2.0e-10	1.1e-08
m=20000	Time [s]	110.4	881.2	191.9	907.5	359.2
g77	Fval	-44182.7	-44182.7	-44182.7	-44182.7	-44182.7
n=14000	Residue	7.8e-11	1.4e-10	7.1e-10	1.2e-10	1.0e-08
m=28000	Time [s]	268.3	1576.9	450.4	2402.6	603.8
g81	Fval	-62624.8	-62624.8	-62624.8	-62624.8	-62624.8
n=20000	Residue	4.6e-11	1.3e-10	1.4e-09	7.9e-11	2.0e-08
m=40000	Time [s]	650.1	4283.9	1219.0	6087.4	1062.1

1D-Kohn-Sham Equation

$$\min\left\{\frac{1}{2}\mathrm{tr}(R^\top LR)+\frac{\alpha}{4}\mathrm{diag}(RR^\top)^\top L^{-1}\mathrm{diag}(RR^\top):\ R^\top R=I_p,\ R\in\mathbb{R}^{n\times r}\right\}, \qquad (3)$$

where L is a tri-diagonal matrix with 2 on its diagonal and -1 on its subdiagonal and $\alpha>0$ is a parameter. We terminate algorithms when $\|\mathrm{grad} f(R)\|<10^{-4}$.

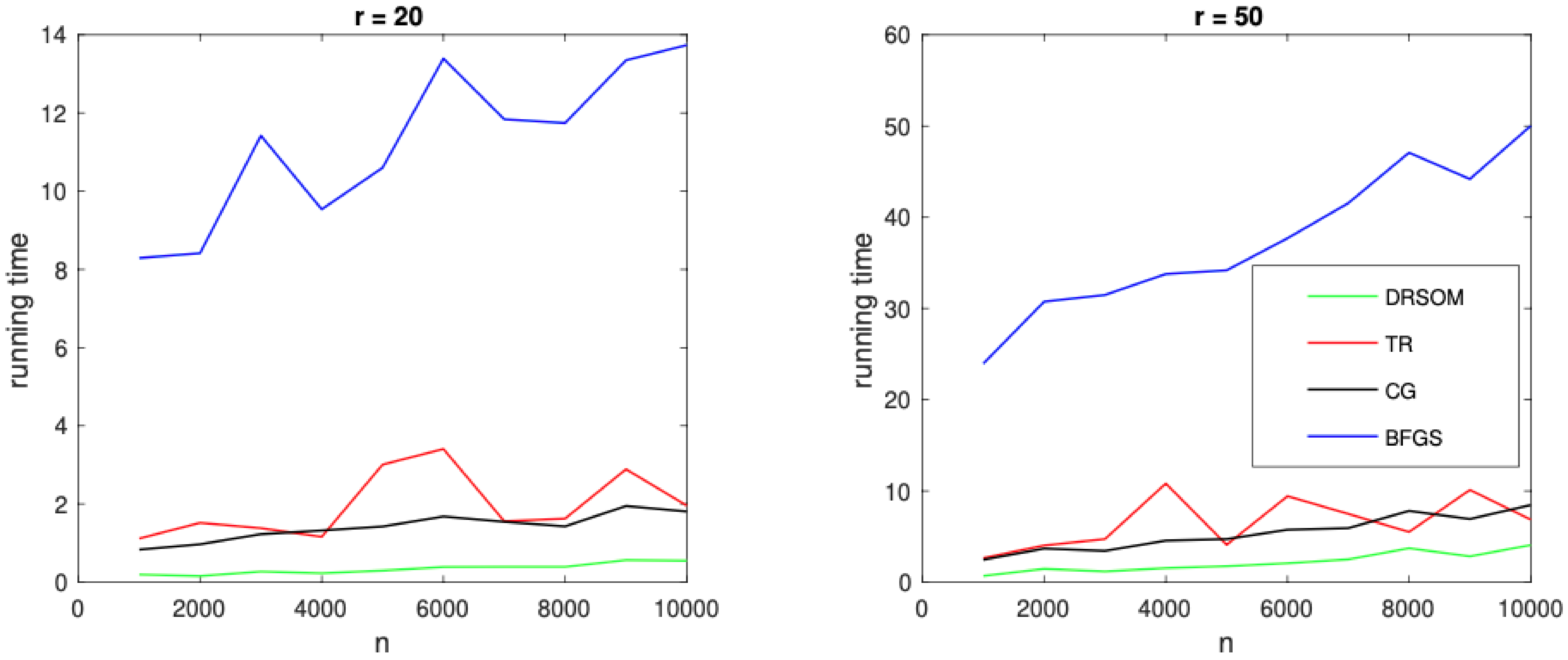


Figure 1: Results for Discretized 1D Kohn-Sham Equation. $\alpha=1$.

Today’s Talk

- **New developments of ADMM-based interior point (ABIP) Method**
- **Optimal Diagonal Preconditioner and HDSDP**
- **A Dimension Reduced Trust-Region Method**
- **A Homogeneous Second-Order Descent Method**

A Descent Direction Using the Homogenized Quadratic Model I

- **Big Question: How to drop Assumption (c) in DRSOM analyses?**

Recall the classical trust-region method minimizes the quadratic model

$$\min_{d \in \mathbb{R}^n} m_k(d) := g_k^T d + \frac{1}{2} d^T H_k d$$

$$\text{s.t.} \|d\| \le \Delta_k .$$

- **-g_k is the first-order steepest descent direction but ignores Hessian; the direction of H_k-negative curvature v meets Assumption (c) and also enables $O(\epsilon^{1.5})$ decrease if**

$$R(H_k, v) = v^T H_k v / \|v\|^2 < -\sqrt{\epsilon},$$

but such direction does not exist if it becomes nearly convex…

- **Could we construct a direction integrating both?**

Answer: Use the homogenized quadratic model!

A Descent Direction Using the Homogenized Quadratic Model II

- **Using the homogenization trick by lifting with extra scalar t:**

$$\psi_k\left(\xi_0, t; \delta\right) := \frac{1}{2}\begin{bmatrix}\xi_0 \\ t\end{bmatrix}^T \begin{bmatrix}H_k & g_k \\ g_k^T & -\delta\end{bmatrix}\begin{bmatrix}\xi_0 \\ t\end{bmatrix} = \frac{t^2}{2}\begin{bmatrix}\xi_0/t \\ 1\end{bmatrix}^T \begin{bmatrix}H_k & g_k \\ g_k^T & -\delta\end{bmatrix}\begin{bmatrix}\xi_0/t \\ 1\end{bmatrix}$$

- **The homogeneous model is equivalent to m_k up to scaling:**

$$\psi_k(\xi_0, t; \delta) = t^2 \cdot (m_k(\xi_0/t) - \delta)$$

- **Find a good direction $\xi = \xi_0/t$ (if *t = 0* then set *t=1*) by the leftmost eigenvector:**

$$\min_{\|[\xi_0;t]\|\leq 1} \psi_k\left(\xi_0, t; \delta\right)$$

- **Accessible at the cost of $O\left(\epsilon^{-1/4}\right)$ via the randomized Lanczos method.**

This is the Classical Homogenization Trick in QCQP via SDP

- **For inhomogeneous QP (and QCQP):**

$$\begin{aligned} \min\ & x^T Q_0 x - 2b_0^T x \\ \text{s.t.}\ & x^T Q_i x - 2b_i^T x + c_i \le 0, \quad i = 1,\dots,m \end{aligned} \quad \Rightarrow \quad \begin{aligned} \min\ & x^T Q_0 x - 2b_0^T xt \\ \text{s.t.}\ & x^T Q_i x - 2b_i^T xt + c_i t^2 \le 0, \quad i = 1,\dots,m \\ & t^2 = 1 \end{aligned}$$

- **Used with SDP relaxation:**

$$\begin{aligned} \min\ & M_0 \bullet X \\ \text{s.t.}\ & M_i \bullet X \le 0, \quad i = 1,\dots,m \\ & X_{00} = 1, X \succeq 0 \end{aligned} \quad \Leftarrow \quad M_i = \begin{bmatrix} c_i & b_i^T \\ b_i & Q_i \end{bmatrix}, X = \begin{bmatrix} 1 & x^T \\ x^T & X_0 \end{bmatrix}$$

- **Homogenized QCQP and SDP relaxation enables strong performance and theoretical analysis, and it guarantees a rank-one solution if *m=1*.**

*** Rojas and Sorensen 2001**

The Descent Direction Using the Homogenized Quadratic Model

- **Define the following parametrized (δ) homogenized quadratic model at x_k:**

$$\psi_k(\xi_0, t; \delta) := \frac{1}{2}\begin{bmatrix}\xi_0 \\ t\end{bmatrix}^T \begin{bmatrix}H_k & g_k \\ g_k^T & -\delta\end{bmatrix}\begin{bmatrix}\xi_0 \\ t\end{bmatrix} = \frac{t^2}{2}\begin{bmatrix}\xi_0/t \\ 1\end{bmatrix}^T \begin{bmatrix}H_k & g_k \\ g_k^T & -\delta\end{bmatrix}\begin{bmatrix}\xi_0/t \\ 1\end{bmatrix}$$

  - **The “un-homogenized vector” $\xi = \xi_0/t$ can be found by the leftmost eigenvalue computation and scaling (if *t = 0* then set *t=1*) ;**
  - **Lemma 1 (strict negative curvature) : if $g_k \neq 0, H_k \neq 0,$ let λ_1 be the leftmost eigenvalue of $\begin{bmatrix}H_k & g_k \\ {g_k}^T & -\delta\end{bmatrix}$, then $\lambda_1 \leq -\delta$.**
  - **The motivates us to use ξ as a second-order descent direction resulting a <u>single-looped (easy-to-implement)</u> method**

Theoretical Guarantees of HSODM

- **Consider use the second-order homogenized direction, and the length of each step $\|\eta\xi\|$ is fixed: $\|\eta\xi\| \leq \Delta_k = \frac{2\sqrt{\epsilon}}{M}$ where $f(x)$ has L-Lipschitz gradient and M-Lipschitz Hessian.**

- **Theorem 1 (Global convergence rate) : if $f(x)$ satisfies the Lipchitz Assumption and $\delta = \sqrt{\varepsilon}$, the iterate moves along homogeneous vector ξ: $x_{k+1} = x_k + \eta_k\xi$, then, if we choose $\eta_k = \Delta_k/\|\xi\|$, and terminate at $\|\xi\| < \Delta_k$, then algorithm has $O(\epsilon^{-3/2})$ iteration complexity. Furthermore, x_{k+1} satisfies approximate first-order and second-order conditions.**

Global Convergence Rate: Outline of Analysis

- **A concise analysis using fixed radius Δ**

Let $x_{k+1} = x_k + \eta\xi$, $R(H_k, \xi) = \xi^T H_k \xi / \|\xi\|^2, \xi = \xi_0 / t$

- **(sufficient decrease in large step) If** $\|\xi\| \geq \Delta$, **we choose** $\eta = \Delta / \|\xi\|$
  - $f(x_{k+1}) - f(x_k) \leq -\frac{\delta\Delta^2}{2} + \frac{M}{6}\Delta^3$, **regardless of** $t = 0$ **or not**
  - δ **must be some** greater than $O(\sqrt{\epsilon})$ **to have** $O\left(\epsilon^{\frac{3}{2}}\right)$ decrease
- **(small step means convergence) Otherwise** $\|\xi\| < \Delta$, **then we choose step-size** $\eta = 1$ **and**
  - $\|g_{k+1}\| \leq 4(L + \delta)^2 \Delta^3 + \frac{M}{2}\Delta^2 + (2L\delta + 2\delta^2)\,\Delta$
  - δ **must be some** less than $O(\sqrt{\epsilon})$ **and converge**

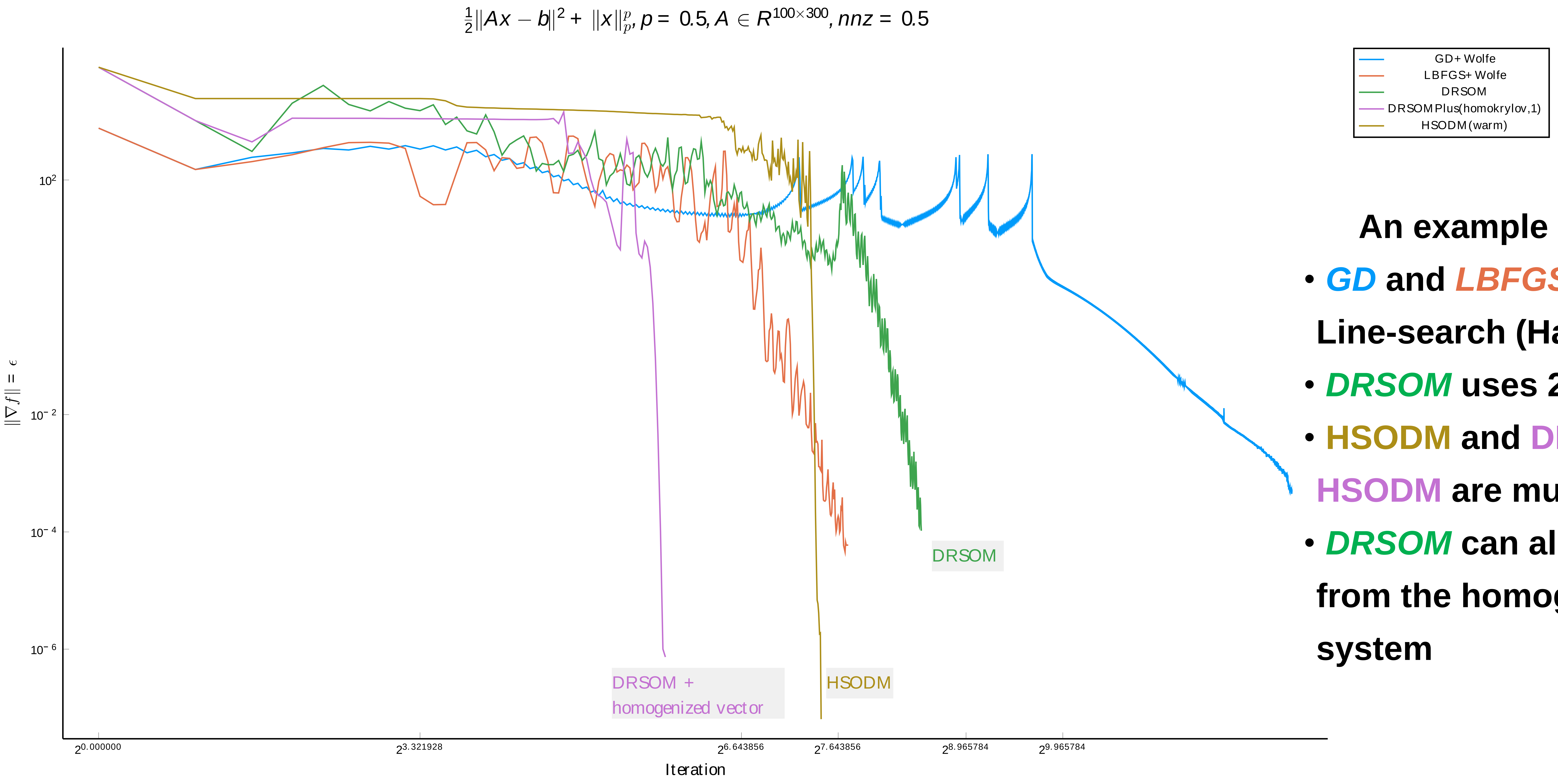
*** The eigenvector does not change, and we do not have to solve ξ again.**

Theoretical Guarantees of HSODM (cont.)

- **Theorem 2 (Local convergence rate): If the iterate x_k of HSODM converges to a strict local optimum x^* such that $H(x^*) \succ 0$,and then $\eta_k = 1$ if k is sufficiently large. If we do not terminate HSODM and set $\delta = 0$, then HSODM has a local superlinear (quadratic) speed of convergence, namely: $\| x_{k+1} - x^* \| = O(\| x_k - x^* \|^2)$**

- **The local convergence property of HSODM is very similar to classical trust-region method when the iterate becomes unconstrained Newton steps**

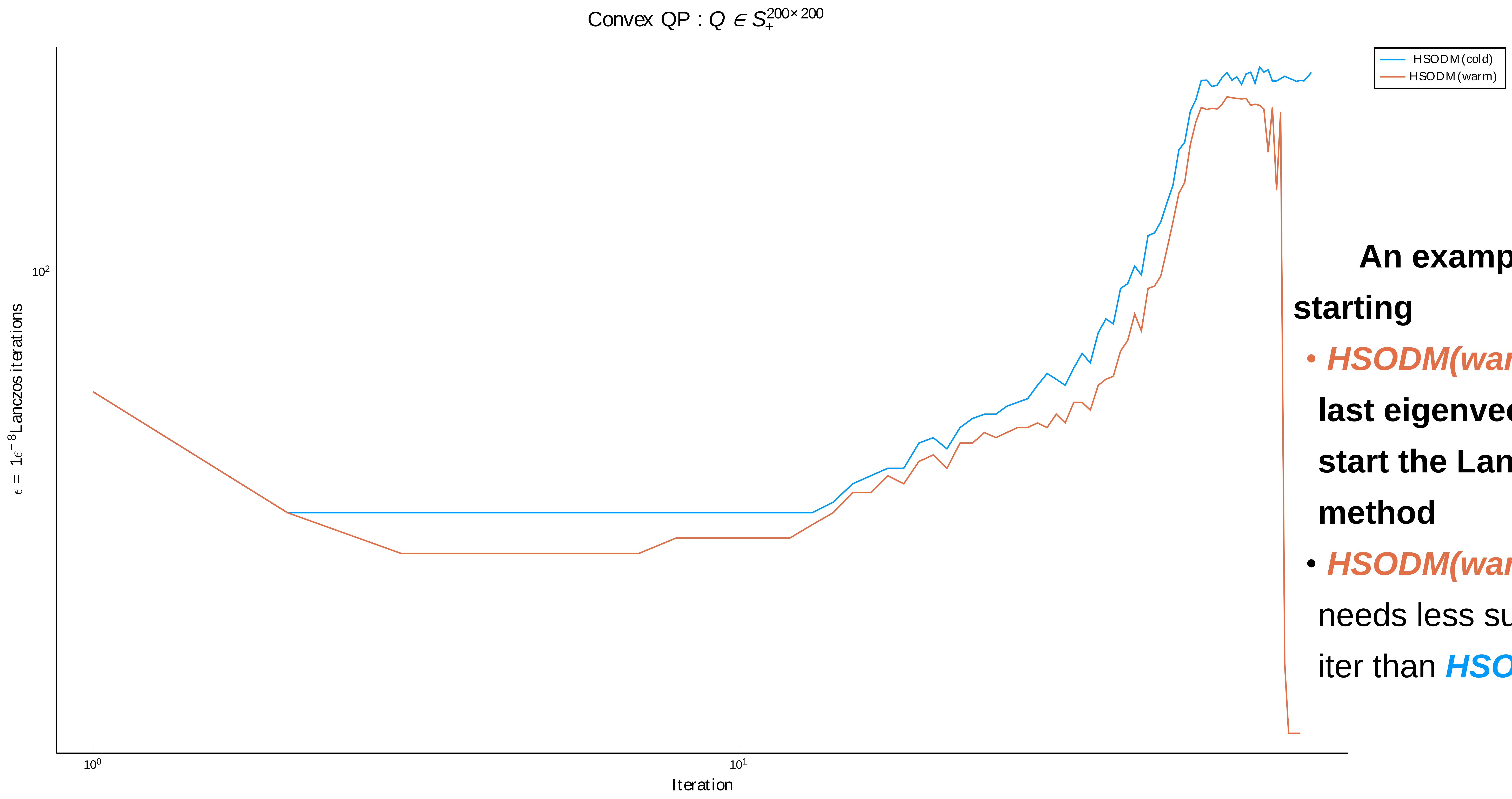
Preliminary results: HSODM and DRSOM + HSODM



An example of L2-Lp

- ***GD*** **and** ***LBFGS*** **both use a Line-search (Hager-Zhang)**
- ***DRSOM*** **uses 2-D subspace**
- **HSODM and DRSOM + HSODM are much better!**
- ***DRSOM*** **can also benefit from the homogenized system**

The Effect of Warm-Starting the Eigenvector



An example of warm starting

- ***HSODM(warm)*** **uses the last eigenvector to warm start the Lanczos method**
- ***HSODM(warm)*** always needs less subproblem iter than ***HSODM(cold)***

Ongoing Research and Future Directions on DRSOM

- Are there other alternatives to remove Assumption c) in DRSOM analyses?
- Low-rank approximation of the homogenized matrix $\begin{bmatrix} H_k & g_k \\ g_k{}^T & \mathbf{0} \end{bmatrix}$(+μ•I, that is, adding sufficiently large scalar μ so that it is positive definite if necessary) to make the leftmost eigenvector computing easier (Randomized rank reduction of a symmetric matrix to log(n), So et al. 08) and "Hot-Start" eigenvector computing by Power Methods (linear convergence of Liu et al. 2017)?
- Indefinite and Randomized Hessian rank-one updating via BFGS/SR1
- Dimension Reduced Non-Smooth/Semi-Smooth Newton

Takeaway: Second-Order Information matters and better to integrate FOM and SOM!

- THANK YOU