A Feasible Method for Optimization with Orthogonality Constraints

Zaiwen Wen[1] Wotao Yin[2]

[1]Shanghai Jiaotong University, Math Dept

[2]Rice University, CAAM Dept

May 2011

RICE SJTU

Orthogonality Constraints

Unknowns: q orthogonal matrices $X_{(1)}, \ldots, X_{(q)} \in \mathbb{R}^{n \times p}$

Optimization problem:

$$\min F(X), \quad \text{subject to} \quad X_{(i)}^\top X_{(i)} = I, \ i = 1, \ldots, q.$$

$X_{(i)}$ is constrained to the *Stiefel manifold*.

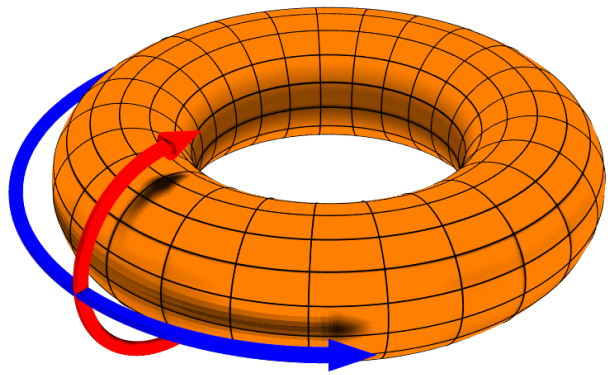
RICE SJTU

The Vector Case

If $p = 1$

- $X_{(i)}$ reduces to $x_{(i)} \in \mathbb{R}^n$
- $X_{(i)}^\top X_{(i)} = I$ becomes $\|x_{(i)}\|_2 = 1$ or $x_{(i)} \in S^{n-1}$.

Example: $n = 2$, $q = 2$, the feasible set is homeomorphic to a torus



By Dave Burke

Why are orthogonality/normalization constraints interesting?

- wide applications;
- non-convex in general, causing possibly local minima;
- expensive to stay feasible in the matrix case.

Applications arise with

1. Direction fields: $\|\vec{x}_{ij}\|_2 = 1$
2. Homogeneous objectives and normalization: $\|x\|_2 = 1$, $\|X\|_F = 1$
3. Eigenvalue problems: $X^\top X = I$
4. Certain SDPs (Burer et al): $0 \preceq Z = X^\top X$, $Z_{ii} = \|x_i\|_2 = 1$
5. Certain combinatorial problems:
   Square X is a permutation matrix $\Longleftrightarrow$ $X^\top X = I$ and $X \geq 0$
6. .....

The Method

Stay Feasible vs Infeasible

- Infeasible algorithms
  - Penalty, augmented Lagrangian methods
  - SDP relaxations / lifting
- Feasible algorithms
  - iterative projection
  - descent along geodesics (or retractions)

  Advantages:
  - Work when cost functions are meaningless outside of feas. set
  - In case of early termination, a feasible sol is returned
  - Fewer parameters (e.g., for penalty)

  Disadvantages: maintaining feasibility can be expensive

Basic Idea

Consider

$$\min F(X), \quad \text{subject to} \quad X^\top X = I.$$

At iteration k

$$\boxed{X_{k+1} \leftarrow \text{Orthogonalize} \leftarrow X_k + \tau \text{Proj}(-\nabla F_k)}$$

$$\Updownarrow$$

$$\boxed{X_{k+1} \leftarrow \text{Orthogonalize} \leftarrow X_k + \tau H_k X_k}$$

$$\Downarrow$$

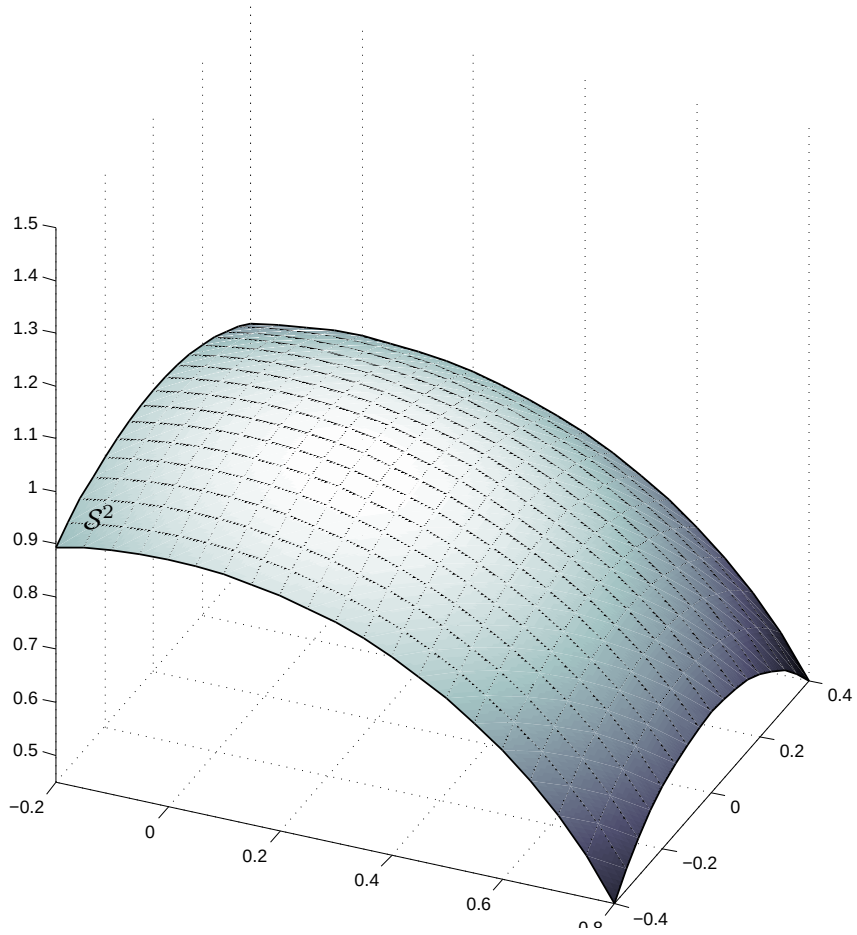
$$\boxed{X_{k+1} \leftarrow \text{solution } Y(\tau) \text{ of } Y = X_k + \tfrac{\tau}{2} H_k (X_k + Y)}$$

Need H_k such that

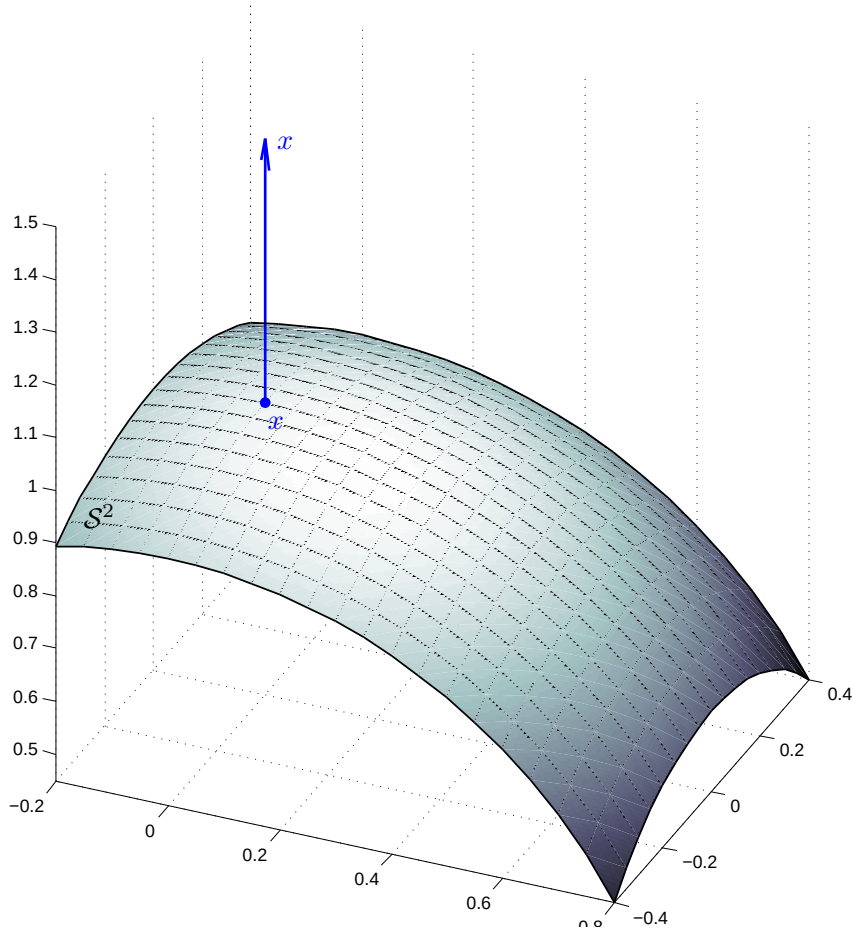
1. $Y(\tau)^\top Y(\tau) = X_k^\top X_k$;
2. $Y(\tau)$ is a *descent* curve;
3. solving for $Y(\tau)$ is fast.

RICE SJTU

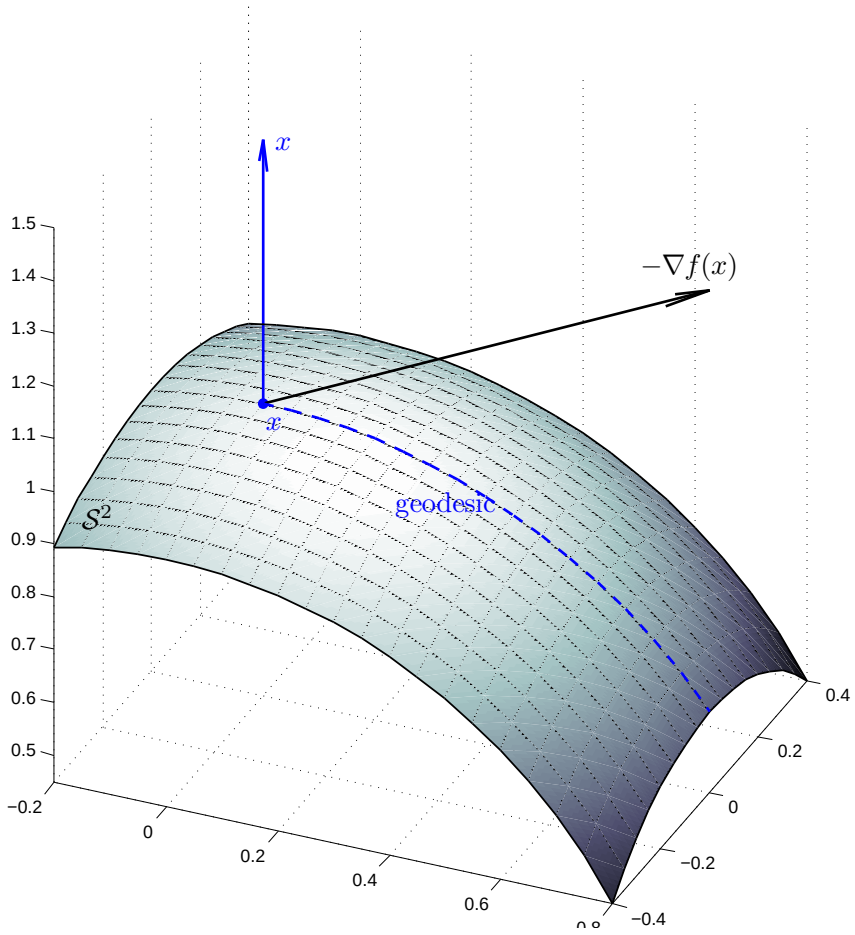
Example: $\mathcal{S}^2 = \{x \in \mathbb{R}^3 : \|x\|_2 = 1\}$



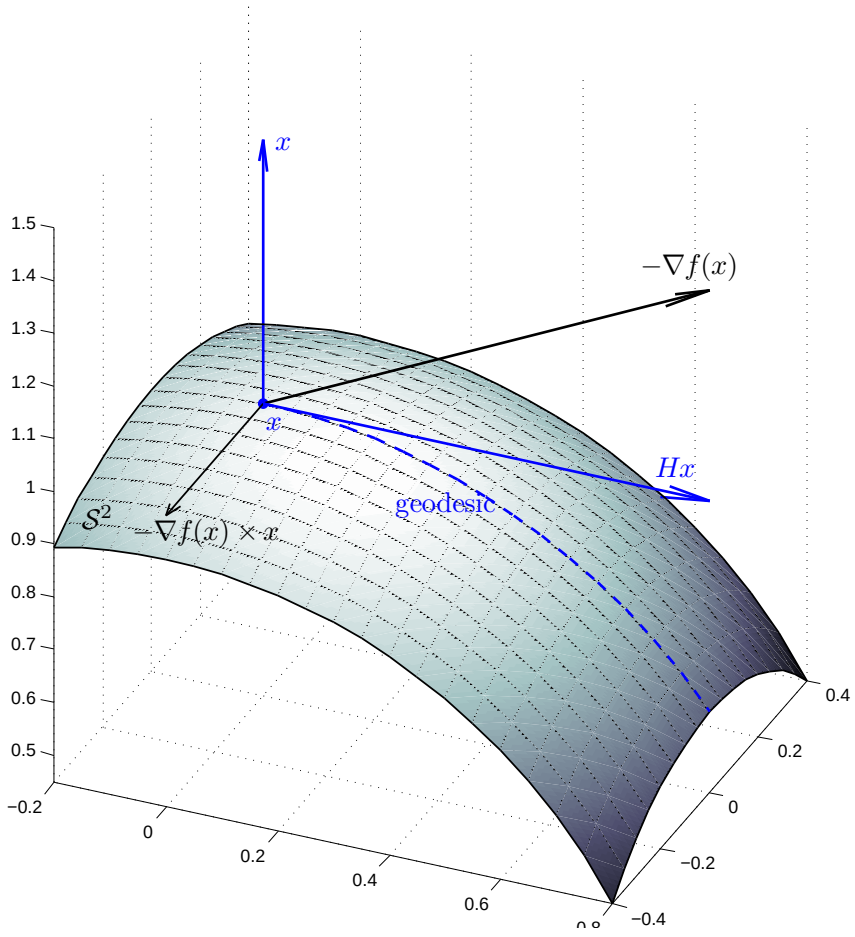
Example: $\mathcal{S}^2 = \{x \in \mathbb{R}^3 : \|x\|_2 = 1\}$



Example: $\mathcal{S}^2 = \{x \in \mathbb{R}^3 : \|x\|_2 = 1\}$

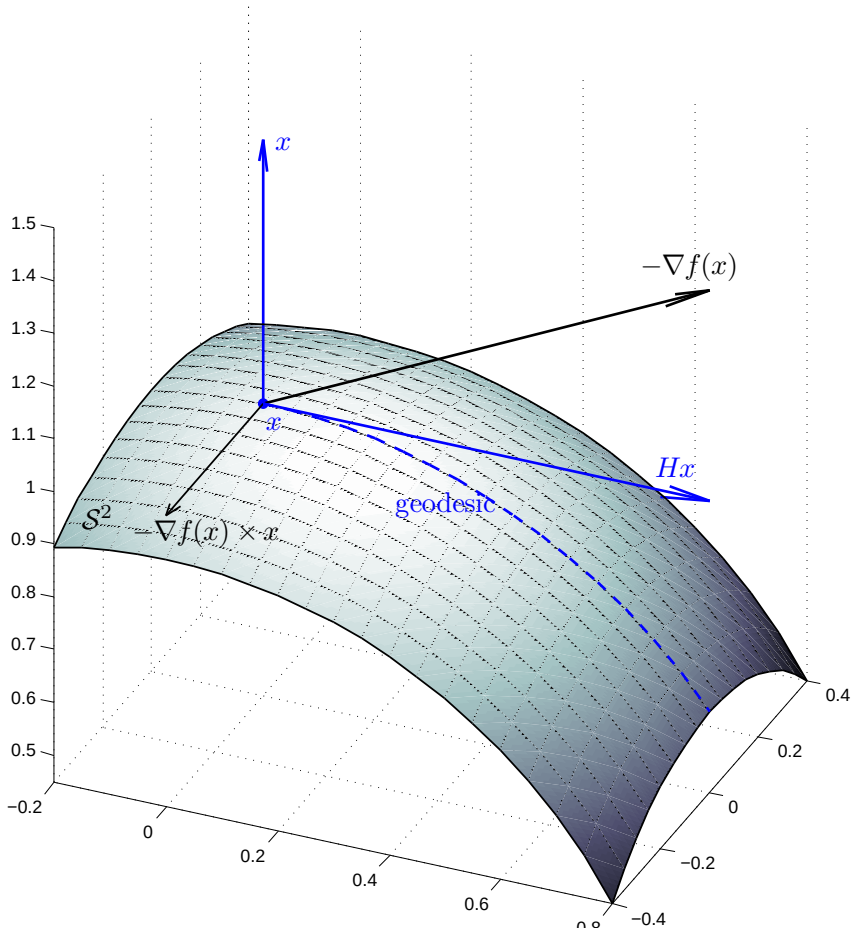


Example: $\mathcal{S}^2 = \{x \in \mathbb{R}^3 : \|x\|_2 = 1\}$



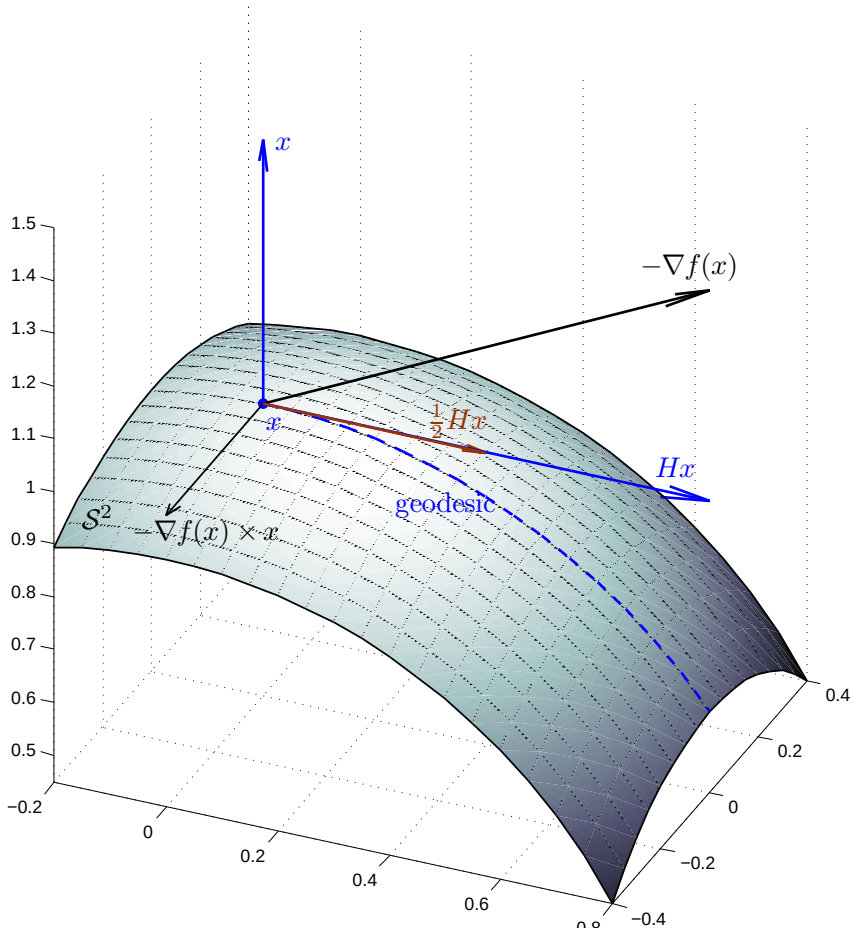
- $\text{Proj}(-\nabla f) = x \times (-\nabla f \times x)$

Example: $\mathcal{S}^2 = \{x \in \mathbb{R}^3 : \|x\|_2 = 1\}$



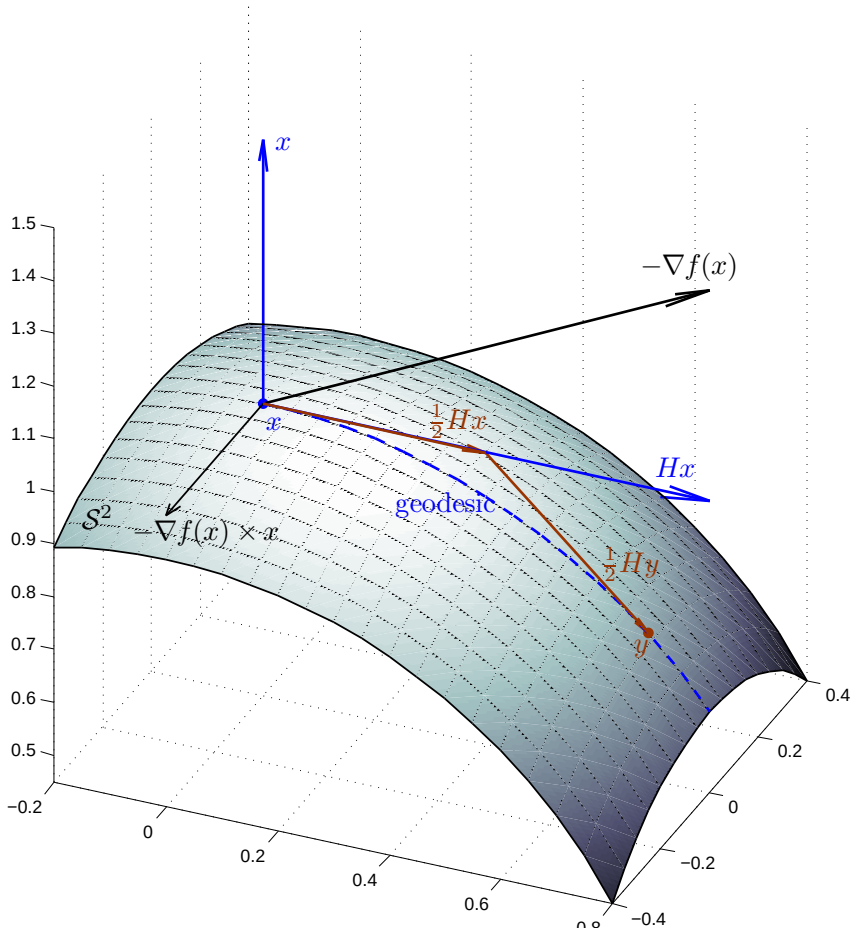
- $\mathrm{Proj}(-\nabla f) = x \times (-\nabla f \times x) = \left(x\nabla f^\top - \nabla f\, x^\top\right)x = Hx.$

Example: $\mathcal{S}^2 = \{x \in \mathbb{R}^3 : \|x\|_2 = 1\}$



- $\mathrm{Proj}(-\nabla f) = x \times (-\nabla f \times x) = \left(x\nabla f^\top - \nabla f\, x^\top\right)x = Hx.$

Example: $\mathcal{S}^2 = \{x \in \mathbb{R}^3 : \|x\|_2 = 1\}$

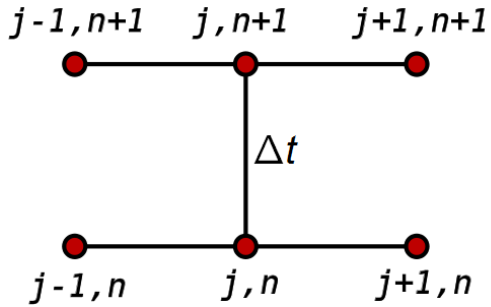


- $\mathrm{Proj}(-\nabla f) = (-\nabla f \times x) \times x = \left(x\nabla f^\top - \nabla f\, x^\top\right)x = Hx.$
- Solution y of $y = x + \frac{1}{2}H(x+y)$ is feasible.
- Solution $\{y(\tau)\}$ of $y = x + \frac{\tau}{2}H(x+y)$ is a geodesic.

RICE SJTU

Relation to Crank-Nicolson

Discretize 1D PDE $\frac{\partial u}{\partial t} = F(u, x, t, u'_x, u''_x)$:



$$\text{(forward Euler)} \quad \frac{u_j^{n+1} - u_j^n}{\Delta t} = F_j^n(\cdots)$$

$$\text{(backward Euler)} \quad \frac{u_j^{n+1} - u_j^n}{\Delta t} = F_j^{n+1}(\cdots)$$

$$\text{(Crank-Nicolson)} \quad \frac{u_j^{n+1} - u_j^n}{\Delta t} = \frac{1}{2}\left(F_j^{n+1}(\cdots) + F_j^n(\cdots)\right)$$

Crank-Nicolson is the average *forward Euler* and *backward Euler*:

RICE SJTU

The Matrix Update

Given X_k and $G_k = \nabla F(X_k)$, we let

$$H = X_k G_k^\top - G_k X_k^\top$$

and solve

$$Y = X + \frac{\tau}{2} H(X + Y)$$

for $Y(\tau)$. Update

$$X_{k+1} \leftarrow Y(\tau) = \left(I - \frac{\tau}{2} H\right)^{-1} \left(I + \frac{\tau}{2} H\right) X_k,$$

where τ is a step size.

Properties of $Y(\tau)$:

1. $Y(\tau)^\top Y(\tau) = X_k^\top X_k$;
2. $Y(\tau)$ is a descent curve;
3. $Y(\tau)$ has fast implementations.

Skew-Symmetric H Preserves $X^\top X$

Theorem (Cayley's Transform)

If $H^\top = -H$, then

1. $(I - H)$ *is nonsingular;*
2. $Y = (I - H)^{-1}(I + H)X$ *satisfies* $Y^\top Y = X^\top X$.

Proof.

1. $x^\top (I - H)x = x^\top x$ for any vector x.
2.

$$\begin{aligned} Y^\top Y &= X^\top (I + H)^\top (I - H)^{-T} (I - H)^{-1}(I + H)X \\ &= X^\top (I - H) \ (I + H)^{-1} (I - H)^{-1}(I + H)X \\ &= X^\top (I - H) \ (I - H)^{-1} (I + H)^{-1}(I + H)X \\ &= X^\top X. \end{aligned}$$

□

RICE SJTU

Generalization

For $M \succ 0$ and *complex-valued* constraints

$$X^\top M X = C.$$

The update

$$Y = (I - HM)^{-1}(I + HM)X$$

preserves

$$Y^\top M Y = X^\top M X.$$

provided that H is a skew-Hermitian matrix.

We now know that $X^\top X$ can be preserved by

$$X_{k+1} \leftarrow Y(\tau) = \left(I - \frac{\tau}{2}H\right)^{-1}\left(I + \frac{\tau}{2}H\right)X_k.$$

This is known as *Cayley's transform*, which has been used in matrix computations such as inverse eigenvalue problems [Friedland-Nocedal-Overton'87]. It is also called Crank-Nicolson schemes for heat PDEs, etc. It appears that it has not been systematically studied for minimizing a general differentiable $F(X)$ subject to $X^\top X = I$.

Properties of $Y(\tau)$:

1. $Y(\tau)^\top Y(\tau) = X_k^\top X_k$;
2. $Y(\tau)$ is a descent curve;
3. $Y(\tau)$ has fast implementations.

- $Y(\tau)$ is a descent curve since $Y'(0) =$ negative projected gradient.

$$Y'(0) = HX_k = \mathrm{Proj}^{\mathrm{c}}(-\nabla F_k; X_k).$$

$\mathrm{Proj}^{\mathrm{c}}$ is projection to the tangent of $\{X : X^\top X = I\}$ under the *canonical metric*:

$$\langle Y, Z\rangle_c = \mathrm{tr}(Y^\top (I - \frac{1}{2} XX^\top) Z), \quad Y, Z \in T_X.$$

- For projection under the *Euclidean metric*: $\langle Y, Z\rangle_e = \mathrm{tr}(Y^\top Z)$, use

$$H = XG_k^\top (I - \frac{1}{2} X_k X_k^\top) - (I - \frac{1}{2} X_k X_k^\top) G_k X^\top.$$

- Generally, given any tangent direction $D \in T_X$ (CG, Newton, etc.),

$$H = (I - \frac{1}{2} XX^\top) DX^\top - XD^\top (I - \frac{1}{2} XX^\top)$$

leads to $Y'(0) = D$.

RICE SJTU

Properties of $Y(\tau)$:

1. $Y(\tau)^\top Y(\tau) = X_k^\top X_k$;
2. $Y(\tau)$ is a descent curve;
3. $Y(\tau)$ has fast implementations.

Comparing with Descent along Geodesics

Moving along geodesics is a sensible choice (c.f. Smith'93, Mahony'94; Edelman-Arias-Smith'98) for optimization on manifold, but geodesics of $\{X : X^\top X = I\}$ are difficult to compute.

$Y(\tau) = \left(I - \frac{\tau}{2}H\right)^{-1}\left(I + \frac{\tau}{2}H\right)X$ is not a geodesic. It is a *retraction* ("approximate geodesic") of the Stiefel manifold; see book by Absil, Mahony, and Sepulchre.

Computing Cost

$$Y(\tau) = \left(I - \frac{\tau}{2}H\right)^{-1}\left(I + \frac{\tau}{2}H\right)X.$$

$p = 1$: $Y(\tau)$ is closed-form, a linear combination of X and ∇F,

small p: apply the SMW formula and solve a smaller $(2p \times 2p)$ system,

large p: solve an $n \times n$ system, OR perform low-rank approximates.

Cheaper than:

- geodesic steps: approximating matrix exponentials or solving 2nd-order PDEs;
- gradient projection: matrix orthogonalization, $n \times p$ SVD (e.g., Manton'02);
- gradient + orthogonalization: QR decomposition.

In short, we proposed a *curvilinear* descent path, which maintains feasible and easy to compute. The path $Y(\tau)$ is neither straight nor a geodesic, and τ is not linear in the length of $Y(\tau)$.

Convergence Properties

Theorem

Consider

$$\min F(X), \quad \textit{s.t.} \quad X^\top X = I.$$

X is a stationary point if and only if $X^\top X = I$ and $HX = 0$.
If $HX \neq 0$, then $Y(\tau)$ is a descent curve.

We also proved

- X_k globally converges to a stationary point,
- near a stationary point, τ is lower bounded,
- $HX = 0$ if and only if $H = 0$.
  As $X_k \to X_{station}$, $(I - \frac{\tau}{2}H) \to I$, so $(I - \frac{\tau}{2}H)$ is well-behaved.

Full Algorithm and Numerical Results

Algorithm:

1. Input: feasible X_0
2. $k \leftarrow 0$
3. While *stopping conditions not met* do
4. $\quad H \leftarrow X_k(\nabla F_k)^\top - (\nabla F_k)X_k^\top$
5. $\quad$ Compute an initial τ by Barzilai-Borwein
6. $\quad \tau \leftarrow$ non-monotonic "line" search
7. $\quad X_{k+1} \leftarrow Y(\tau) = \left(I - \frac{\tau}{2}H\right)^{-1}\left(I + \frac{\tau}{2}H\right)X_k$
8. $\quad k \leftarrow k + 1$

Code was written in MATLAB.

Three types of problems:

1. Guaranteed global minimum;
2. Numerical global minimum, but no proof yet; or just lucky?
3. Local (non-global) minima; we restart algorithm from random points, and pick the best solution.

Max-Cut SDP

SDP:

$$\max_X \ \mathrm{tr}(CX), \ \ \text{s.t.} \ \ X_{ii} = 1, \ i = 1, \cdots, n, \ X \succeq 0.$$

NLP: write $X = V^\top V$ where $V = [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n] \in \mathbb{R}^{p \times n}$

$$\max_{V \in \mathbb{R}^{p \times n}} \ \sum_{i,j} c_{ij} \boldsymbol{v}_i^\top \boldsymbol{v}_j, \ \ \text{s.t.} \ \|\boldsymbol{v}_i\| = 1, \ i = 1, \ldots, n.$$

If $p \geq \mathrm{rank}(X^*)$, NLP is equivalent to SDP.

Theorem (a slight extension of Burer-Monteiro'05)

There exists $\bar{p}(C) \leq n$ such that, if $p \geq \bar{p}(C)$, any local minimizer of NLP is globally optimal.

A simple upper bound:

$$\bar{p}(C) \leq n + 1 - \inf\{\mathrm{rank}(C + D) : D \text{ is diagonal}\}.$$

RICE SJTU

Max-Cut SDP

SDPLR for max-cut, C language				Our code				
Name	n	obj	CPU	p	obj	CPU	nfe	feasi
t..g3-15	3375	3.134523e+03	20.32	20	3.134567e+03	8.46	625	1.2e-14
t..pm3-15-50	3375	3.475072e+03	11.24	20	3.475131e+03	7.88	583	1.2e-14
G41	2000	2.865195e+03	25.86	20	2.865215e+03	2.50	378	9.3e-15
G42	2000	2.946239e+03	21.21	20	2.946251e+03	2.70	433	9.3e-15
G48	3000	5.999956e+03	7.38	20	6.000000e+03	2.14	251	1.2e-14
G50	3000	5.988114e+03	4.88	20	5.988172e+03	4.66	547	1.2e-14
G55	5000	1.103920e+04	14.72	20	1.103946e+04	7.75	407	1.5e-14
G57	5000	3.885369e+03	16.48	20	3.885403e+03	12.41	627	1.5e-14
G58	5000	2.013593e+04	90.80	20	2.013539e+04	13.63	620	1.5e-14
G60	7000	1.522191e+04	18.59	20	1.522224e+04	14.90	523	1.8e-14
G62	7000	5.430703e+03	23.19	20	5.430777e+03	16.96	623	1.7e-14
G63	7000	2.824334e+04	77.89	20	2.824284e+04	19.49	638	1.8e-14
G64	7000	1.046582e+04	128.52	20	1.046561e+04	20.19	644	1.7e-14
G65	8000	6.205298e+03	28.38	20	6.205384e+03	20.25	620	1.9e-14
G66	9000	7.076941e+03	33.58	20	7.077048e+03	22.02	624	2.0e-14
G67	10000	7.744093e+03	38.28	20	7.744265e+03	24.84	624	2.1e-14
G70	10000	9.861247e+03	52.12	20	9.861523e+03	24.78	626	2.1e-14
G72	10000	7.808215e+03	37.10	20	7.808381e+03	24.47	622	2.2e-14
G77	14000	1.104509e+04	51.25	20	1.104550e+04	33.10	636	2.5e-14
G81	20000	1.565514e+04	74.79	20	1.565574e+04	44.98	634	3.0e-14

SDRLP for max-cut by S. Burer and R. Monteiro

RICE SJTU

Low-Rank Nearest Correlation Matrix Estimation

Given a weight matrix W and a symmetric matrix C:

$$\min_{X \succeq 0} \ \frac{1}{2}\|W \odot (X - C)\|_F^2, \ X_{ii} = 1, \ i = 1, \ldots, n, \ \mathrm{rank}(X) \leq r.$$

Low-rank approximation: $X = V^\top V$, $V \in \mathbb{R}^{r \times n}$

$$\min_{V \in \mathbb{R}^{r \times n}} \ \frac{1}{2}\|W \odot (V^\top V - C)\|_F^2, \ \text{ s.t. } \|V_i\|_2 = 1, \ i = 1, \ldots, n.$$

Test examples:

Ex1: $n = 500$, $G_{ij} = 0.5 + 0.5e^{-0.05|i-j|}$, $W = I$

Ex3: $n = 500$, $G_{ij} = 0.5 + 0.5e^{-0.05|i-j|}$, random W

Ex5: $n = 943$, G is based on the Movielens data sets. W is given by T. Fushiki

RICE SJTU

Low-Rank nearest correlation problem

	Major			PenCorr			Our code		
r	obj	CPU	feasi	obj	CPU	feasi	obj	CPU	feasi
Ex1									
2	1.564e+02	8.34	1.2e-13	1.564e+02	53.20	4.3e-09	1.686e+02	0.25	3.5e-15
5	7.883e+01	4.54	1.4e-13	7.883e+01	10.34	3.1e-08	7.882e+01	1.12	3.7e-15
20	1.570e+01	12.64	5.6e-15	1.570e+01	5.31	7.8e-08	1.570e+01	1.26	5.1e-15
50	4.140e+00	142.30	7.1e-15	4.139e+00	2.29	5.2e-07	4.139e+00	5.71	5.9e-15
100	1.471e+00	928.58	9.5e-15	1.466e+00	2.56	2.4e-07	1.467e+00	19.46	8.0e-15
125	1.055e+00	1731.39	9.4e-15	1.048e+00	2.85	3.0e-08	1.049e+00	24.65	8.1e-15
Ex3									
2	9.106e+02	14.64	1.4e-13	9.109e+02	163.18	4.6e-07	9.778e+02	0.76	3.6e-15
5	4.536e+02	31.42	1.2e-13	4.537e+02	91.15	7.7e-07	4.535e+02	2.04	3.9e-15
20	8.812e+01	218.99	5.4e-15	8.812e+01	69.65	3.6e-07	8.851e+01	3.36	4.9e-15
50	2.203e+01	2022.31	7.1e-15	2.191e+01	94.46	8.6e-07	2.188e+01	27.36	6.0e-15
100	7.110e+00	9649.30	9.2e-15	6.389e+00	121.00	2.6e-07	6.457e+00	39.97	8.0e-15
125	5.030e+00	13801.88	1.0e-14	4.179e+00	151.09	9.6e-07	4.348e+00	48.05	8.8e-15
Ex5									
5	1.141e+04	407.47	2.6e-13	1.147e+04	2480.31	5.7e-07	1.140e+04	35.89	5.1e-15
10	7.586e+03	1299.88	6.1e-15	7.638e+03	2305.20	2.8e-07	7.586e+03	33.17	5.8e-15
20	5.200e+03	1733.57	7.3e-15	5.219e+03	2062.14	2.7e-07	5.194e+03	30.54	6.1e-15
50	3.712e+03	4154.29	9.0e-15	3.718e+03	1356.16	2.0e-09	3.711e+03	21.99	7.8e-15
100	3.503e+03	6426.36	1.2e-14	3.507e+03	906.02	3.3e-07	3.502e+03	22.05	9.7e-15
150	3.501e+03	8734.17	1.6e-14	3.505e+03	824.69	3.1e-07	3.500e+03	27.78	1.1e-14

Major: Pietersz-Groenen, PenCorr: Y.Gao-D.Sun

RICE SJTU

Linear eigenvalue problem

Let $\lambda_1 \geq \ldots \geq \lambda_n$ be the eigenvalues of A:

$$\sum_{i=1}^{p} \lambda_i := \max_{X \in \mathbb{R}^{n \times p}} \operatorname{tr}(X^\top A X) \text{ s.t. } X^\top X = I$$

Compared with MATLAB's *eigs* (which calls Fortran library ARPACK and incurs overhead) on:

- Dense random matrices
- UF Sparse Matrix Collection: 39 large sparse matrices with $n \geq 4000$

Dense matrix: sum of 6 largest eigenvalues

n	500	1000	2000	3000	4000	5000
eigs						
obj	1.153e+04	2.316e+04	4.717e+04	7.075e+04	9.469e+04	1.187e+05
feasi	5.972e-15	8.545e-15	4.295e-15	9.169e-15	6.618e-15	7.112e-15
nAx	132	202	248	354	290	335
cpu	0.113	0.500	2.132	6.040	9.008	16.052
Our code						
obj	1.153e+04	2.316e+04	4.717e+04	7.075e+04	9.469e+04	1.187e+05
feasi	6.468e-16	9.314e-16	8.921e-16	1.073e-15	1.419e-15	1.852e-15
nfe	58	43	74	59	67	84
cpu	0.127	0.410	2.030	3.408	7.094	13.727
err	1.255e-06	9.882e-07	4.649e-06	5.341e-06	4.936e-06	9.378e-06

Dense matrix: fix $n = 5000$

p	1	3	5	7	9	11	13
eigs							
obj	1.986e+04	5.940e+04	9.873e+04	1.379e+05	1.769e+05	2.157e+05	2.545e+05
nAx	200	350	350	372	439	515	468
feasi	1.776e-15	3.739e-15	6.531e-15	9.950e-15	8.200e-15	1.455e-14	1.337e-14
cpu	9.281	16.405	16.133	17.274	20.355	23.849	22.183
Our code							
obj	1.986e+04	5.940e+04	9.873e+04	1.379e+05	1.769e+05	2.157e+05	2.545e+05
feasi	4.441e-16	1.884e-15	1.587e-15	2.047e-15	1.740e-15	1.361e-15	2.384e-15
nfe	91	104	94	98	102	92	101
cpu	4.148	13.550	14.867	16.016	17.843	16.714	19.711
err	1.325e-06	2.102e-07	1.431e-07	1.438e-06	5.221e-06	4.307e-06	2.292e-07

Sparse matrix: sum of 2 largest eigenvalues

		eigs			Our code			
Name	n	obj	CPU	nAx	obj	err	CPU	nfe
crystm03	24696	1.957652e-12	1.24	195	1.957651e-12	8.7e-20	0.35	37
ct20stif	52329	1.773180e+12	0.71	38	1.773180e+12	1.8e-10	1.51	44
msc10848	10848	1.185087e+12	0.26	38	1.185087e+12	4.1e-09	0.27	36
msc23052	23052	1.409469e+09	0.72	91	1.409468e+09	1.1e-06	0.68	58
pwtk	217918	2.016957e+08	181.39	1920	2.016942e+08	7.2e-06	21.94	138
s3rmt3m1	5489	1.930698e+04	0.58	341	1.930698e+04	3.4e-07	0.24	101
bcsstk25	15439	2.120041e+15	0.17	55	2.120041e+15	1.5e-11	0.10	19
bcsstm25	15439	1.395419e+09	3.52	1776	1.395419e+09	3.7e-09	0.09	23
Kuu	7102	1.080637e+02	0.88	394	1.080637e+02	2.2e-07	0.27	86
Muu	7102	1.679791e-03	1.56	969	1.679771e-03	2.0e-08	0.50	194
finan512	74752	5.611798e+01	2.18	127	5.611798e+01	3.0e-09	1.65	43
nd3k	9000	2.541715e+02	2.85	214	2.541715e+02	2.2e-08	1.20	72
nasa4704	4704	4.132959e+08	0.04	38	4.132959e+08	2.6e-10	0.02	15
nasasrb	54870	5.296036e+09	5.28	272	5.296036e+09	9.0e-08	2.68	78
fv1	9604	0	24.07	17318	9.019638e+00		0.46	173
fv2	9801	0	26.56	17678	9.019546e+00		0.47	163
fv3	9801	0	26.14	17678	7.999393e+00		0.45	165
t2dal_e	4257	0	3.72	7688	4.144770e-05		0.04	41
aft01	8205	1.000000e+15	0.14	88	1.000000e+15	8.4e-13	0.03	11
cfd1	70656	1.358235e+01	4.77	215	1.358228e+01	5.3e-06	4.26	97

RICE SJTU

Quadratic Assignment Problem (QAP)

Minimizing a quadratic over permutation matrices:

$$\min_{X \in \mathbb{R}^{n \times n}} \operatorname{tr}(A^\top X B X^\top), \text{ s.t. } X^\top X = I, \ X \geq 0$$

Tested 134 problems in QAPLIB:

- rewrite the objective as $\operatorname{tr}(A^\top (X \odot X) B (X \odot X)^\top)$
- apply augmented Lagrangian to $X \geq 0$ with parameter μ
- run our code 40 times from random X_0 with $\mu = 0.1, 1, 10$ each
- pick the best out of 120 rounded solutions

QAPLIB: exact recovery

gap $= \frac{\text{best upper bound} - v}{\text{best upper bound}} \times 100\%$, CPU: average cpu

			Our code					
Name	n	obj	μ	min gap %	med gap %	max gap %	CPU	feasi
chr12b †	12	9742	1.0e+01	0.000	58.222	134.223	0.59	0.0e+00
esc128 †	128	64	1.0e+00	0.000	7.812	15.625	22.01	0.0e+00
esc16a †	16	68	1.0e+00	0.000	5.882	8.824	0.35	0.0e+00
esc16b †	16	292	1.0e+00	0.000	0.343	0.685	0.71	0.0e+00
esc16c †	16	160	1.0e+00	0.000	0.000	0.000	0.61	0.0e+00
esc16d †	16	16	1.0e+00	0.000	12.500	12.500	0.26	0.0e+00
esc16e †	16	28	1.0e+00	0.000	7.143	7.143	0.28	0.0e+00
esc16f †	16	0	1.0e+00	0.000	0.000	0.000	0.01	0.0e+00
esc16g †	16	26	1.0e+00	0.000	7.692	7.692	0.17	0.0e+00
esc16h †	16	996	1.0e+00	0.000	0.000	0.000	0.83	0.0e+00
esc16i †	16	14	1.0e+00	0.000	0.000	0.000	0.23	0.0e+00
esc16j †	16	8	1.0e+00	0.000	12.500	25.000	0.12	0.0e+00
esc32b †	32	168	1.0e+00	0.000	20.238	35.714	0.90	0.0e+00
esc32c †	32	642	1.0e+00	0.000	0.000	0.000	1.76	0.0e+00
esc32d †	32	200	1.0e+00	0.000	6.000	11.000	1.18	0.0e+00
esc32e †	32	2	1.0e+00	0.000	0.000	0.000	0.69	0.0e+00
esc32g †	32	6	1.0e+00	0.000	33.333	33.333	0.78	0.0e+00
esc64a †	64	116	1.0e+00	0.000	0.000	0.000	5.75	0.0e+00

RICE SJTU

QAPLIB: exact recovery

			Our code					
Name	n	obj	μ	min gap %	med gap %	max gap %	CPU	feasi
had12 †	12	1652	1.0e+00	0.000	0.726	2.179	0.70	0.0e+00
had14 †	14	2724	1.0e+00	0.000	0.404	1.762	0.81	0.0e+00
had16 †	16	3720	1.0e+00	0.000	0.215	0.591	0.93	0.0e+00
had18 †	18	5358	1.0e+00	0.000	0.336	0.859	1.19	0.0e+00
had20 †	20	6922	1.0e-01	0.000	0.520	2.427	1.22	0.0e+00
lipa20a †	20	3683	1.0e+00	0.000	2.675	3.204	1.18	0.0e+00
lipa20b †	20	27076	1.0e+00	0.000	14.614	15.951	0.79	0.0e+00
lipa30a †	30	13178	1.0e+00	0.000	1.829	2.254	2.34	0.0e+00
lipa30b †	30	151426	1.0e+00	0.000	16.295	16.640	1.38	0.0e+00
lipa40b †	40	476581	1.0e+00	0.000	9.576	18.493	2.26	0.0e+00
lipa50b †	50	1210244	1.0e+00	0.000	0.000	0.000	0.55	0.0e+00
lipa60b †	60	2520135	1.0e+00	0.000	19.244	19.889	7.63	0.0e+00
lipa70b †	70	4603200	1.0e+00	0.000	10.108	20.217	6.43	0.0e+00
lipa80b †	80	7763962	1.0e+00	0.000	21.205	21.550	15.08	0.0e+00
lipa90b †	90	12490441	1.0e+01	0.000	21.430	22.119	21.08	0.0e+00
nug12 †	12	578	1.0e+00	0.000	5.536	13.495	0.46	0.0e+00
nug14 †	14	1014	1.0e-01	0.000	5.128	10.454	0.53	0.0e+00
nug16a †	16	1610	1.0e+00	0.000	4.099	8.571	0.67	0.0e+00
nug16b †	16	1240	1.0e+00	0.000	1.129	2.258	0.60	0.0e+00
tai12a †	12	224416	1.0e+00	0.000	8.753	14.529	0.49	0.0e+00

RICE SJTU

QAPLIB: $n \geq 80$

			Our code					
Name	n	obj	μ	min gap%	med gap%	max gap%	CPU	feasi
esc128 †	128	64	1.0e+00	0.000	7.812	15.625	22.01	0.0e+00
lipa80a	80	253195	1.0e+00	0.682	0.813	1.048	17.69	0.0e+00
lipa80b †	80	7763962	1.0e+00	0.000	21.205	21.550	15.08	0.0e+00
lipa90a	90	360630	1.0e+00	0.627	0.732	0.933	23.82	0.0e+00
lipa90b †	90	12490441	1.0e+01	0.000	21.430	22.119	21.08	0.0e+00
sko100a	100	152002	1.0e+01	0.795	1.601	2.751	22.59	0.0e+00
sko100b	100	153890	1.0e-01	0.752	1.598	2.611	22.44	0.0e+00
sko100c	100	147862	1.0e-01	0.764	1.824	2.903	22.52	0.0e+00
sko100d	100	149576	1.0e+01	0.825	1.579	2.474	22.37	0.0e+00
sko100e	100	149150	1.0e-01	0.862	1.820	3.002	22.76	0.0e+00
sko100f	100	149036	1.0e+00	0.910	1.621	2.558	22.84	0.0e+00
sko81	81	90998	1.0e+01	1.123	1.875	2.750	14.47	0.0e+00
sko90	90	115534	1.0e+01	0.841	1.695	2.820	18.04	0.0e+00
tai100a	100	21052466	1.0e+00	2.773	3.342	4.068	26.46	0.0e+00
tai100b	100	1185996137	1.0e+00	1.741	5.642	9.388	51.31	0.0e+00
tai150b	150	498896643	1.0e+01	1.930	3.199	4.523	118.96	0.0e+00
tai256c	256	44759294	1.0e-01	0.842	1.352	2.401	270.49	0.0e+00
tai80a	80	13515450	1.0e+01	2.964	3.629	4.637	15.10	0.0e+00
tai80b	80	818415043	1.0e-01	0.761	4.859	8.515	29.33	0.0e+00
tho150	150	8133398	1.0e+00	1.194	2.040	2.795	84.77	0.0e+00
wil100	100	273038	1.0e+01	0.370	0.776	1.102	22.24	0.0e+00

† exact recovery

RICE SJTU

Polynomial Optimization

Consider minimizing the following polynomials subject to $\|x\|_2 = 1$:

1. $\sum_{1 \le i < j < k < l \le 50} (-i - j + k + l) x_i x_j x_k x_l$
2. $\sum_{1 \le i < j < k \le 49} x_i x_j x_k + x_i^2 x_j - x_i^2 x_k + x_j x_k^2$
3. $\sum_{1 \le i \le 20} x_i^6 + \sum_{1 \le i \le 19} x_i^3 x_{i+1}^3$
4. $\sum_{1 \le i < j < k \le 20} x_i^2 x_j^2 x_k^2 + x_i^3 x_j^2 x_k + x_i^2 x_j^3 x_k + x_i x_j^3 x_k^2$

SDP (Nie)			Our code			
F(x)	f_{sos}^{hmg}	cpu	rep	F(x) (min, mean, max)	CPU	feasi
1	-140.4051	3:30:00	10	(-140.4051,-140.4051,-140.4051)	2.45	8.0e-16
2	-124.9645	4:00:00	10	(-124.9645,-124.9645,-124.9645)	0.10	3.8e-16
3	3.446e-5	10:00:00	1000	(6e-5, 0.00143, 0.00391)	0.01	2.2e-15
4	-0.3827	11:00:00	1000	(-0.3827, -0.2725, -0.1059)	0.01	1.3e-15

Problems from J.Nie, *Regularization Methods for Sum of Squares Relaxations* .....

Nie's code solves an SDP relaxation, which has approximation guarantees.

RICE SJTU

Stability Number of Graph

Given a graph $G = (V, E)$, Motzkin and Straus showed

- $\alpha(G)^{-1} \leftarrow \min \sum_{i=1}^{n} x_i^4 + 2 \sum_{(i,j)\in E} x_i^2 x_j^2$, s.t. $\|x\|_2 = 1$
- Test random graphs G whose edges $e_{i,j}$ for $\{i, j\} \not\subset M$ are generated with probability $\frac{1}{2}$, where M is a subset chosen randomly from V with $|M| = n/2$.

SDP (Nie)		Our code			
n	CPU	$\alpha(G)$	rep	CPU (min, mean, max)	feasi
20	0:03:04	10	10	(0.004, 0.006, 0.007)	1.7e-15
30	0:39:58	15	10	(0.007, 0.009, 0.011)	7.8e-15
40	1:21:01	20	10	(0.005, 0.007, 0.010)	1.4e-15
50	26:27:12	25	10	(0.008, 0.010, 0.012)	5.5e-15
60	75:46:58	30	10	(0.009, 0.012, 0.015)	3.6e-15
80	—	40	10	(0.013, 0.015, 0.020)	2.0e-15
100	—	50	10	(0.018, 0.023, 0.036)	2.4e-15

RICE SJTU

1-Bit CS

Recover x from

$$\min \|x\|_1, \text{ s.t. } Ax \geq 0, \ \|x\|_2 = 1.$$

Our approach (joint J. Laska, Z. Wen, and R. Baraniuk)

1. **Outer**: form augmented Lagrangian subproblem

$$\min \mu\|x\|_1 + \frac{1}{2}\|Bx - b\|_2^2, \text{ s.t. } \|x\|_2 = 1,$$

   and update μ, B and b.

2. **Inner**: solve the above subproblem by iterating

$$x_{k+1} \leftarrow \min_{\|x\|_2=1} \mu\tau\|x\|_1 + \frac{1}{2}\|x - (x_k + \tau g_k)\|_2^2$$

Shrinkage on Sphere

Shrinkage (soft-thresholding):

$$\text{shrink}(y, \mu) \leftarrow \min_x \ \mu\|x\|_1 + \frac{1}{2}\|x - y\|_2^2.$$

Shrinkage on sphere: add constraint $\|x\|_2 = 1$

$$x_{\text{opt}} \leftarrow \begin{cases} \frac{\text{shrink}(y,\mu)}{\|\text{shrink}(y,\mu)\|_2}, & \text{shrink}(y, \mu) \neq 0, \\ \text{sign}(y_i)e_i, & \text{o.w.}, \end{cases}$$

where y_i is the largest (in magnitude) entry of y.

Matrix case:

$$\min_X \ \mu\|X\|_* + \frac{1}{2}\|X - Y\|_F^2, \ \text{s.t.} \ \|X\|_F = 1.$$

reduces to shrinkage on sphere over singular values.