A Riemannian optimization approach for computing low-rank solutions of Lyapunov equations

Bart Vandereycken

Seminar for Applied Mathematics, ETH Zürich

15th Leslie Fox Prize Meeting in Numerical Analysis
June 27, 2011

The low-rank picture

The low-rank picture

Given a matrix X, compute its low-rank approximation.

- Why low rank?
  - Like sparsity, low rank is a popular parsimonious structure.
  - Unlike sparsity, it can have global support.

- Why not SVD?
  - Assume that X is available.
  - Does not work if (some of) the entries of X are unavailable.

- Typical application: matrix equations and matrix completion.

Define the set

$$\mathcal{S}_+(k,n) = \{X : X \in \mathbb{R}^{n\times n},\ X = X^T,\ X \succeq 0,\ \text{rank}(X) = k\}.$$

Our low-rank solver in [V./Vandewalle '10] is based on

$$\min_X f(X) \quad \text{subject to } X \in \mathcal{S}_+(k,n). \tag{1}$$

Key points:

- Exploit smoothness of $\mathcal{S}_+(k,n)$ as a Riemannian manifold.
- Solve (1) using Riemannian optimization.
- Make the algorithm efficient: precondition the Hessian.

The Lyapunov equation

- Matrix equations: Lyapunov, Sylvester, Riccati

  Abound in $\mathcal{H}_2$-control, model reduction by balanced truncation, stability analysis [Moore '81], [Antoulas '05], [Benner/Mehrmann/Sorensen '05], [Meerbergen/Spence '10], ...

- The generalized symmetric Lyapunov equation:

$$AXM^T + MXA^T = C$$

  Given $A, M, C \in \mathbb{R}^{n\times n}$, solve for unknown matrix $X \in \mathbb{R}^{n\times n}$

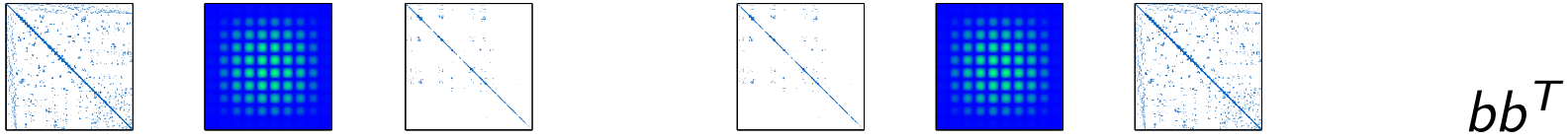
  - Assume symmetry: $A = A^T, M = M^T, C = C^T \Rightarrow X = X^T$
  - Assume coercivity: $A, M \succ 0, C \succeq 0 \Rightarrow X \succeq 0$

- Matrix X is not directly available without first solving the Lyapunov equation.

Matrix equations applied to large-scale problems, e.g. PDEs

- FEM discretized system A and mass M matrix
- rhs (load) matrix $C = bb^T$, $b \in \mathbb{R}^{n \times k_C}$.

$$A \quad X \quad M^T \quad + \quad M \quad X \quad A^T \quad = \quad C$$



$$bb^T$$

Main problem

sparse A, M is $O(n)$ $\quad \leftrightarrow \quad$ solving dense X is $O(n^2)$ memory and $O(n^3)$ flops [Bartels/Stewart '72]

$\rightsquigarrow$ so $n \gg 1000$ is problematic

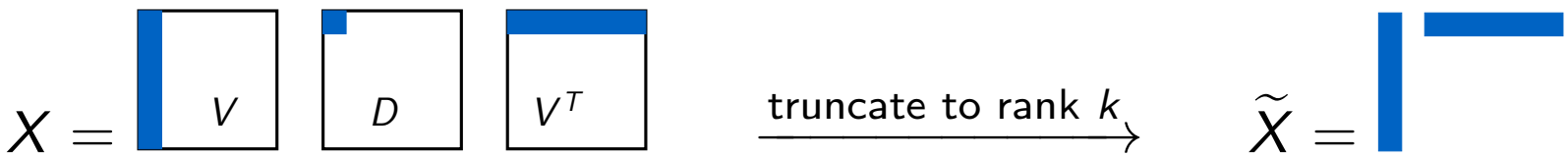
Low-rank approximation

Under reasonable conditions, we have the low-rank phenomenon: the singular values of X decay exponentially fast.

$\rightsquigarrow$ Decay depends on rank rhs C, spectrum of $A - \lambda M$, see [Penzl '00], [Antoulas/Sorenson/Zhou '02], [Grasedyck '04], ...

This means that X has low numerical rank k for a precision ϵ.

$\rightsquigarrow$ A has conditioning $\kappa(A)$: $k = O(\log(1/\epsilon)\log(\kappa(A))$



Main problem

Compute the "best" rank-k approximation $\widetilde{X}$ efficiently in $O(nk^c)$.

Existing methods

$n < O(10^4)$	Schur form [Bartels/Stewart '72] Hammarling, Jonsson, Kågström, Sorensen, Zhou, Quintana-Ortí, van de Geijn, Granat, Kressner, ...
store A^{-1}	Sign function iteration [Roberts '71] Beavers, Denman, Byers, Benner, Quintana-Ortí, Grasedyck, Bauer, ...
apply $(A - \sigma I)^{-1}$	ADI [Wachspress '88] Penzl, Li, White, Gugercin, Simoncini, Hodel, Saak, ...
apply A	Krylov subspace [Saad '90] Hu, Reichel, Jaimoukha, Kasenally, Hochbruck, Starke, Simoncini, Kressner, ...
levels A_i	Multilevel methods [Rosen-Wang '95] Penzl, Grasedyck, Hackbusch, V., Vandewalle, ...

... many other solvers and hybrid combinations

Krylov methods

Krylov subspace methods for $AX + XA^T = bb^T$

- construct a Krylov basis V_k

$$V_k = \text{span}\{A^i b\} \quad \text{with } i = 0 \dots k \text{ or } i = -k \dots k$$

- Galerkin condition: solve small Lyapunov equation for x_k

$$(V_k^T A V_k) x_k + x_k (V_k^T A V_k)^T = E_k$$

- Approximation is $X_k = V_k x_k V_k^T$ with x_k such that the *energy norm* is minimized for the basis $V_k \otimes V_k$

Drawbacks (and similar for most other methods):

- Compute low-rank solutions as a (deliberate) side-effect
- Factors V_k are not very good: only x_k optimized
- Slow convergence $\rightsquigarrow$ high-rank factors $\rightsquigarrow$ needs truncation

Proposed solution: improve factors by optimizing V_k directly.

Optimization on the manifold of low-rank matrices

The method we proposed in [V./Vandewalle '10] will

- minimize the energy norm,
- over the manifold of positive semidefinite (PSD) matrices of fixed rank k.

$$\begin{aligned} \min \quad & f : \mathcal{S}_+(k,n) \to \mathbb{R},\ X \mapsto \operatorname{tr}(XAXM) - \operatorname{tr}(XC), \\ \text{s.t.} \quad & \mathcal{S}_+(k,n) = \{X : X \in \mathbb{R}^{n\times n},\ X \succ 0,\ \operatorname{rank}(X) = k\}. \end{aligned}$$

Scalability constraint for each step

- all operations,
- all data structures

must be $O(nk^c)$, c small.

The objective function

$$f : \mathcal{S}_+(k, n) \to \mathbb{R}, \; X \mapsto \mathrm{tr}(XAXM) - \mathrm{tr}(XC),$$

reflects a weighted norm of the error.

Proof:

- The vec$(\cdot)$ operator gives the isomorphism $\mathbb{R}^{n^2} \simeq \mathbb{R}^{n \times n}$ as

$$\mathrm{tr}(X^T Y) = \mathrm{vec}(X)^T \mathrm{vec}(Y).$$

- $AXM + MXA = C$ is a linear system of size n^2:

$$\mathcal{L}\,\mathrm{vec}(X) = \mathrm{vec}(C) \quad \text{with } \mathcal{L} = A \otimes M + M \otimes A.$$

The objective function

- Take $\mathcal{L}$-norm of the error $E = X - X_*$:

$$\begin{aligned} \| \operatorname{vec}(E) \|_{\mathcal{L}}^2 &= \operatorname{vec}(E)^T \mathcal{L} \operatorname{vec}(E) \\ &= \operatorname{vec}(E)^T (A \otimes M + M \otimes A) \operatorname{vec}(E) \\ &= 2 \operatorname{tr}(EMEA). \end{aligned}$$

- Work out the error E:

$$\begin{aligned} \| \operatorname{vec}(E) \|_{\mathcal{L}}^2 &= 2 \operatorname{tr}[(X - X_*)M(X - X_*)A] \\ &= 2 \operatorname{tr}(XMXA) - 2 \operatorname{tr}(XC) + 2 \operatorname{tr}(X_* M X_* A) \\ &= 2f(X) + 2 \operatorname{tr}(X_* M X_* A). \end{aligned}$$

Minimizing $f(X)$ $\iff$ minimizing $\| \operatorname{vec}(E(X)) \|_{\mathcal{L}}$

Does $\| \operatorname{vec}(E) \|_{\mathcal{L}}$ make sense? If $A, M \succ 0$, then $\mathcal{L} \succ 0$.

How do we optimize over

$$\mathcal{S}_+(k,n) = \{X : X \in \mathbb{R}^{n\times n},\ X \succ 0,\ \text{rank}(X) = k\}?$$

Main obstacle: $\mathcal{S}_+(k,n)$ is not a vector space since

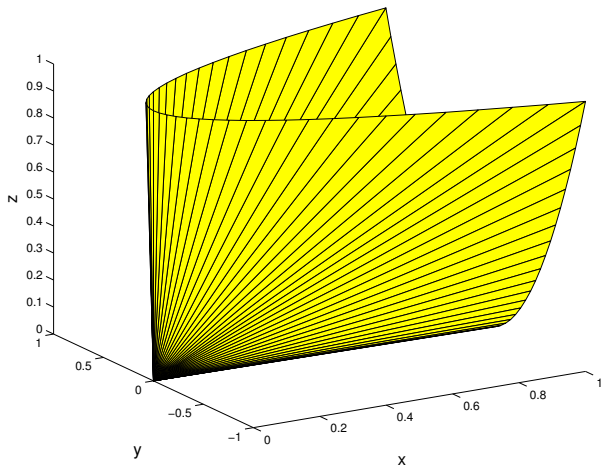
$$\exists X, Y \in \mathcal{S}_+(k,n) \ \Rightarrow\ X + Y \notin \mathcal{S}_+(k,n).$$

In general, rank constraints are very difficult. Existing approaches

- Factoring $X = YY^T$ (non-local optimizers YQ)
- SDP relaxation (drop rank constraint)

are not suitable.

Problem: How to optimize on the *curved* space $\mathcal{S}_+(k,n)$?



$\mathcal{S}_+(1,2)$

$\mathcal{S}_+(k,n)$ is a C^∞ smooth manifold.

Manifold property is well known in [algebraic geometry] and [Helmke/Moore '94].

Riemannian algorithms

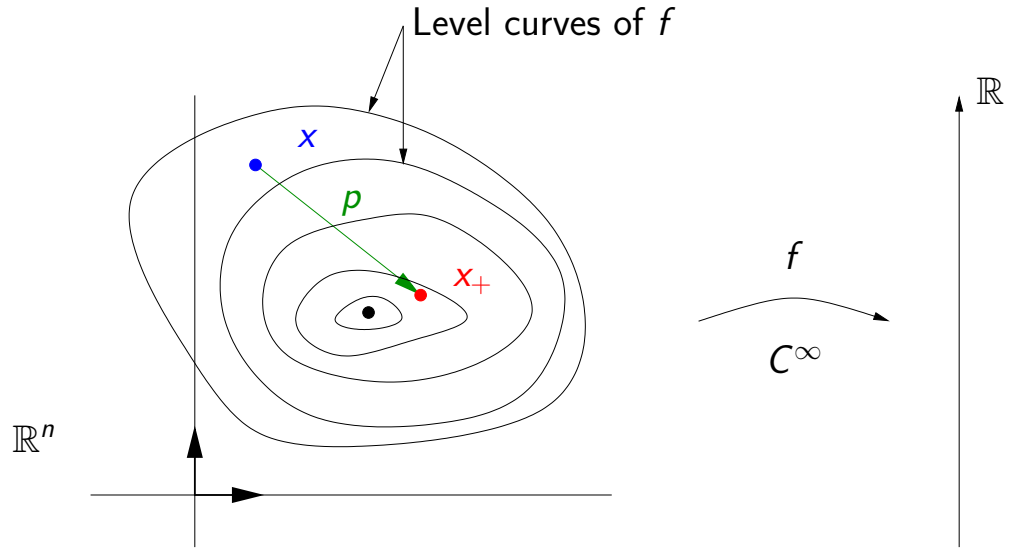
- The idea of exploiting the geometry of manifolds turns up in several areas: geometric integration, Lie group methods, ...

- Riemannian optimization: several "classic" algorithms for unconstrained optimization have been adapted to smooth manifolds
  - Steepest descent, conjugate gradients (CG), Newton
  - See: [Luenberger '72], [Gabay '82], [Shub '86], [Smith '93], [Udrişte '94], [Helmke/Moore '94], [Mahony '94], [Owren/Welfert '96], [Edelman/Arias/Smith '98], ..., [Adler *et.al.* '02], [Absil/Mahony/Sepulchre '08], ...

- Relies on a few basic principles from differential geometry.

- We need new derivations for the geometry of $\mathcal{S}_+(k, n)$.

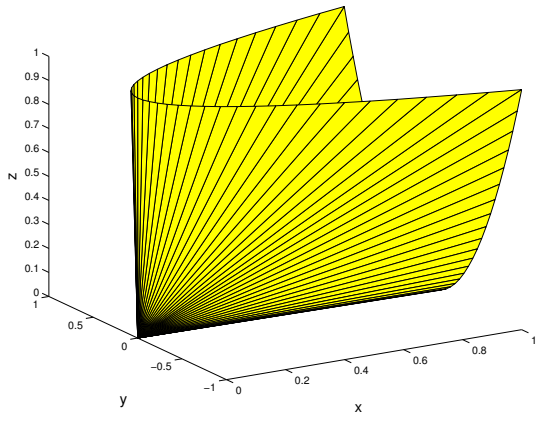
Classic unconstrained optimization: find $\min f$ on $\mathbb{R}^n$

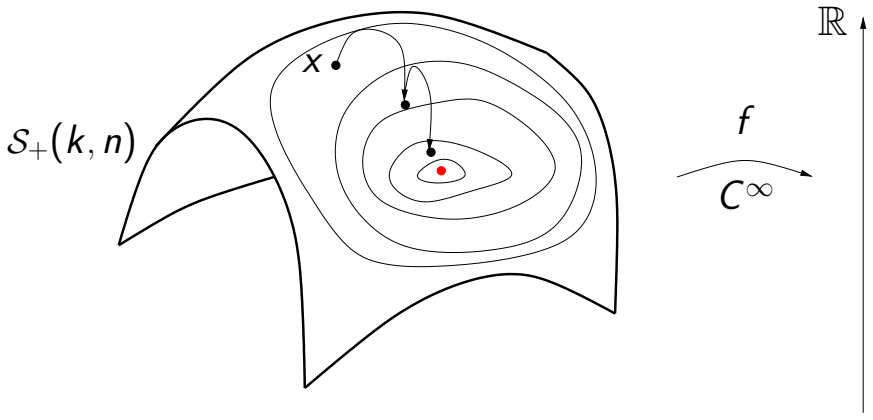


At the current iterate x

1. Determine a step p
   e.g. steepest descent, conjugate gradient, newton direction
2. Compute a better point $x_+ = x + p$
   robust with line-search or trust region
3. Loop: $x \leftarrow x_+$

Optimization on manifolds





- What are the steps p ?

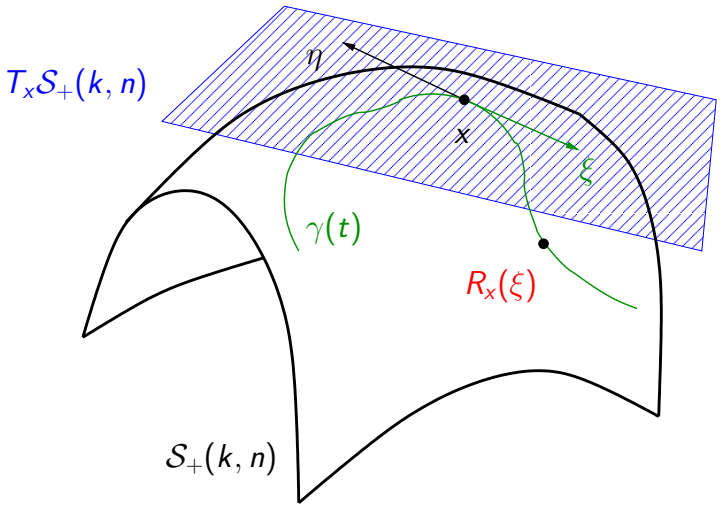
  steepest descent: $\operatorname{grad} f(x)$

  Newton direction: second-order model with $\operatorname{Hess} f(x)$

- How to get $x_+ = x + p$?

  every iterate $x, y \in \mathcal{S}_+(k,n)$ but $x + y \notin \mathcal{S}_+(k,n)$

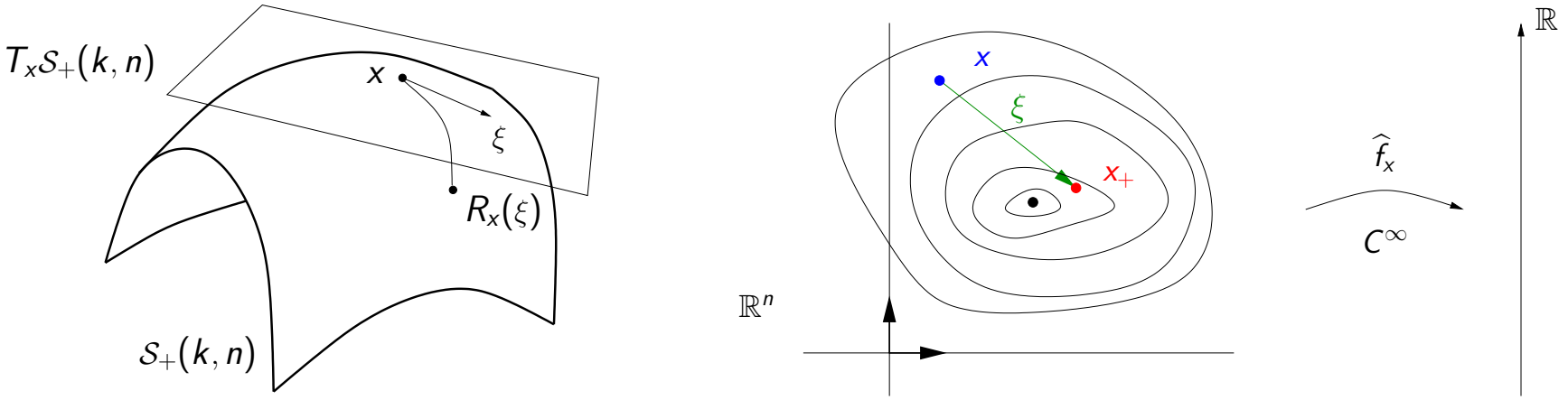
Property: $\mathcal{S}_+(k,n)$ is locally Euclidean $=$ *tangent space*



Properties of the tangent space $T_x\mathcal{S}_+(k,n)$:

- Contains tangent vectors
  $\dot{\gamma}(0) = \xi$ with a curve $\gamma(t)$ on $\mathcal{S}_+(k,n)$
- Linear space: $\xi + \eta \in T_x\mathcal{S}_+(k,n)$ for all $\xi, \eta \in T_x\mathcal{S}_+(k,n)$
- We can go back to the manifold by retracting, e.g., projecting
  retraction R_x is a smooth map $T_x\mathcal{S}_+(k,n) \to \mathcal{S}_+(k,n)$

Result: *Standard unconstrained* optimization



At the current iterate x in the tangent space $T_x\mathcal{S}_+(k,n)$:

1. Determine a step $\xi \in T_x\mathcal{S}_+(k,n)$
   steps are based on the Riemannian gradient and Hessian
2. Compute a better point $x_+ = R_x(\xi)$
3. Loop: $x \leftarrow x_+$

Optimize $\widehat{f}_x$, the pullback of f through $T_x\mathcal{S}_+(k,n)$:

$$\widehat{f}_x : T_x\mathcal{S}_+(k,n) \to \mathbb{R},\ \xi \mapsto f \circ R_x.$$

The embedded geometry of $\mathcal{S}_+(k, n)$

$\mathcal{S}_+(k,n)$ as an embedded submanifold

Elements of $\mathcal{S}_+(k,n)$: $X = VDV^T$ as EVD,

$$X = \begin{bmatrix} V & V_\perp \end{bmatrix} \begin{bmatrix} D & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} V^T \\ V_\perp^T \end{bmatrix}, \quad D \in \mathbb{R}^{k\times k} \text{ diagonal.}$$

Tangent space at $X \in \mathcal{S}_+(k,n)$,

$$\begin{aligned} T_X\mathcal{S}_+(k,n) &= \begin{bmatrix} V & V_\perp \end{bmatrix} \begin{bmatrix} S & C^T \\ C & 0 \end{bmatrix} \begin{bmatrix} V^T \\ V_\perp^T \end{bmatrix}, \; S = S^T \in \mathbb{R}^{k\times k}, C \in \mathbb{R}^{n-k\times k} \\ &= VSV^T + \underbrace{(V_\perp C)}_{Z_U \in \mathbb{R}^{n\times k}} V^T + V \underbrace{(V_\perp C)}_{Z_V \in \mathbb{R}^{n\times k}}{}^T. \end{aligned}$$

The Euclidean **metric** $\langle \cdot, \cdot \rangle$ restricted to $T_X\mathcal{S}_+(k,n)$

$$\langle \xi, \eta \rangle_X := \operatorname{tr}(\xi^T \eta).$$

Retraction: orthogonal projection onto a non-convex set

$$R_x : T_x\mathcal{S}_+(k,n) \to \mathcal{S}_+(k,n)$$
$$: \xi \mapsto P_{\mathcal{S}_+(k,n)}(X+\xi) = \arg\min\{\|X+\xi-Z\|_F : Z \in \mathcal{S}_+(k,n)\}.$$

In [V./Vandewalle '10] we showed:

- Locally, well-defined and C^∞.
- Since $\operatorname{rank}(X+\xi) = 2k$, can be computed in $O(nk^2)$.
- Second-order approximation of the geodesic with expansion

$$R_X(\xi) = X + \xi + P_X^p(\xi)\, X^\dagger\, P_X^p + O(\|\xi\|^3)$$

Useful for deriving the Riemannian Hessian ...

The Newton **step** ξ is the minimizer of the second-order model

$$m_k(\xi) = f(X_k) + \langle \text{grad } f(X_k), \xi \rangle_{X_k} + \frac{1}{2} \langle \text{Hess } f(X_k)[\xi], \xi \rangle_{X_k}$$

- The Euclidean metric $\langle \cdot, \cdot \rangle$ on each $T_X \mathcal{S}_+(k, n)$:

$$\langle \xi, \eta \rangle_X := \text{tr}(\xi^T \eta).$$

- The Riemannian gradient of f is the vector grad f such that

$$\frac{\text{grad } f(X)}{\| \text{grad } f(X) \|} = \arg \max_{\xi \in T_X \mathcal{S}_+(k,n),\, \|\xi\|=1} Df(X)[\xi].$$

- Gradient is the direction of steepest ascent w.r.t. $\langle \cdot, \cdot \rangle_X$.

The Newton **step** ξ is the minimizer of the second-order model

$$m_k(\xi) = f(X_k) + \langle \operatorname{grad} f(X_k), \xi \rangle_{X_k} + \frac{1}{2} \langle \operatorname{Hess} f(X_k)[\xi], \xi \rangle_{X_k}$$

- The Riemannian Hessian of f is the unique linear and symmetric mapping $\operatorname{Hess} f$

  $$\operatorname{Hess} f \ : \ T_X \mathcal{S}_+(k,n) \to T_X \mathcal{S}_+(k,n),$$

  such that

  $$\langle \operatorname{Hess} f(X)[\xi], \xi \rangle_X = \left. \frac{\mathrm{d}^2}{\mathrm{d}t^2} \right|_{t=0} f(R_X(t\,\xi)).$$

- Valid because R_X is a second-order appr. to geodesic!

Applied to $f(X) = \operatorname{tr}(XAXM) - \operatorname{tr}(XC)$, we obtained analytical expressions for

- the gradient of $f(X)$

$$\operatorname{grad} f(X) = P_T(R), \quad R := AXM + MXA - C,$$

with $P_T(Z) := P_V Z P_V + P_V^\perp Z P_V + P_V Z P_V^\perp$ the orthogonal projection onto $T_X\mathcal{S}_+(k,n)$.

- the Hessian as matrix vector product

$$\operatorname{Hess} f(X)[\xi] = P_T(A\xi M + M\xi A) + P_T^p(RP_T^p(\xi)X^\dagger + X^\dagger P_T^p(\xi)R)$$

with $P_T^p(Z) := P_V^\perp Z P_V + P_V Z P_V^\perp$

$\rightsquigarrow$ second-order model can be evaluated in $O(nk^2)$ flops.

Trust-Region Newton on the manifold

RLyap: final algorithm to solve for low-rank approximation of X.

- Choose[1] a rank k, minimize

$$\min_{X \in \mathcal{S}_+(k,n)} \operatorname{tr}(XAXM) - \operatorname{tr}(XC).$$

by the Riemannian Trust-Region (RTR) method of [Absil/Baker/Gallivan '07].

- Key step: solve Newton system

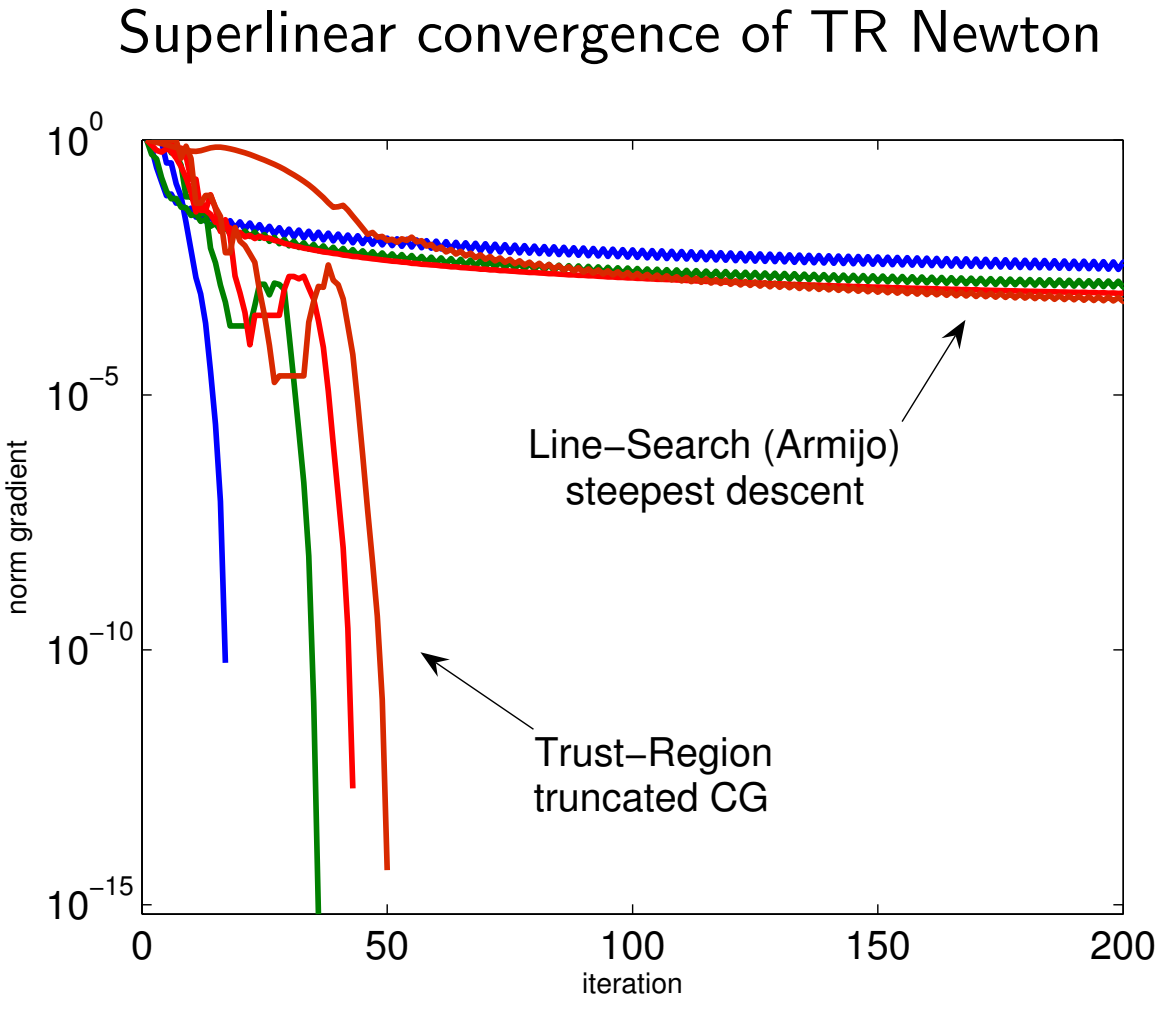
$$m_k : T_{X_k}\mathcal{S}_+(k,n) \to \mathbb{R},$$
$$\xi \mapsto f(X_k) + \langle \operatorname{grad} f(X_k), \xi \rangle + \frac{1}{2}\langle \operatorname{Hess} f(X_k)[\xi], \xi \rangle$$

with truncated PCG to obtain step

$$\eta_k = \arg\min m_k(\xi) \quad \text{s.t. } \|\xi\| \leq \Delta_k.$$

[1] perform an outer loop to get the minimum rank for a desired residual

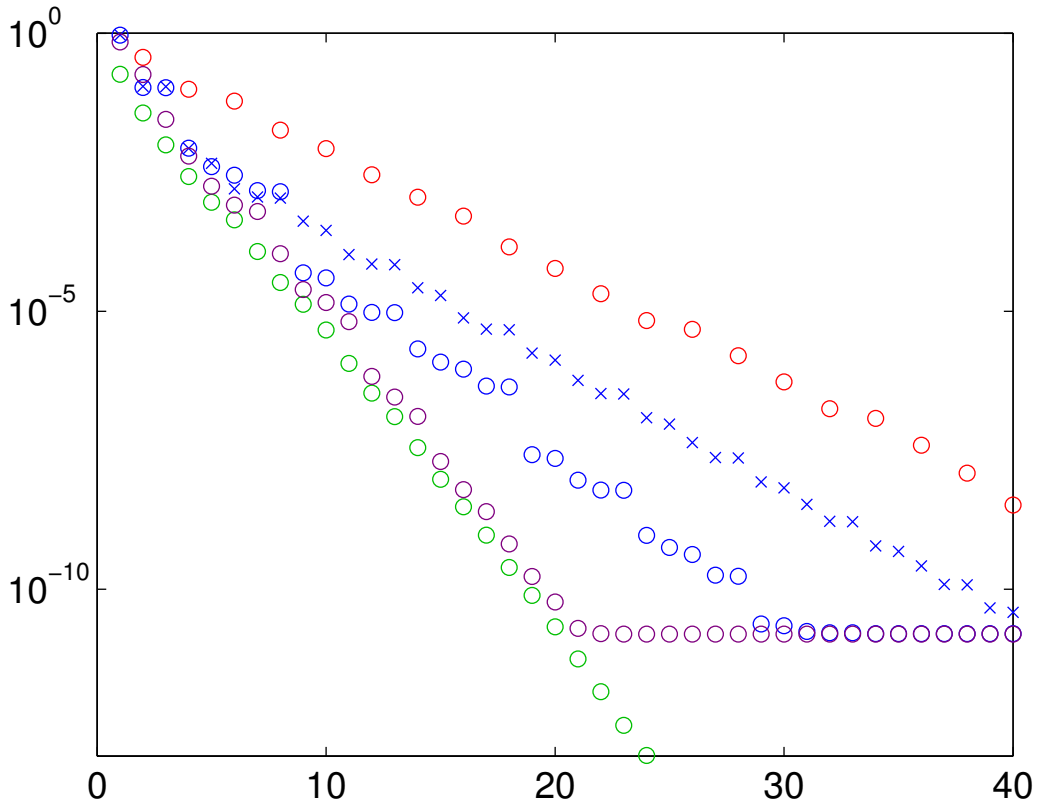
Experimental results for RTR



Test problem: 1D Laplace with $n = 1000$, ranks $k = 5, 10, 15, 20$.

Experimental results for RTR

Relative error of low-rank approximations for different ranks



Comparing truncated SVD (o), min f_E (o),
CFADI[2](5,20,10) (×), CFADI[1](10,50,25) (o), and KPIK[2] (o).
Test problem: RAIL benchmark [Benner/Saak '04] with $n = 1357$.

[2][Penzl '99],[Li/White '04] [2][Simoncini '07]

Preconditioning

Preconditioning

RTR uses (truncated) CG to solve the Newton system.

- Too many iterations for PDEs.
  Example RTR for 2D Laplace, $k = 15$, tol. gradient $= 10^{-10}$:

n	150^2	200^2	250^2	300^2	350^2	400^2	450^2	500^2
n_{outer}	46	44	49	44	43	44	56	48
$\sum n_{\text{inner}}$	1913	2173	2984	3158	4076	4185	5375	5622
$\max n_{\text{inner}}$	414	529	624	731	757	858	1004	1080

- Can we precondition CG?

The Riemannian Hessian is a modified Euclidean Hessian:

$$\mathcal{H}_X = P_X \underbrace{(A \otimes M + M \otimes A)}_{\mathcal{L}} P_X + P_X^p (X^\dagger \otimes R + R \otimes X^\dagger) P_X^p.$$

- Neglect curvature $\rightsquigarrow$ precondition with $P_X \mathcal{L} P_X$.
- $P_X \mathcal{L} P_X$ is the (first-order) Gauss–Newton model of $f(X)$ on $\mathcal{S}_+(k, n)$, cfr. [Adler/Dedieu/Margulies/Martens/Shub '02].

Preconditiong with Gauss–Newton

Does it reduce the number of iterations?

- Observed to be mesh-independent.
- Same 2D Laplace example:

prec.	n	150^2	200^2	250^2	300^2	350^2	400^2	450^2	500^2
none	n_{outer}	46	44	49	44	43	44	56	48
	$\sum n_{\text{inner}}$	1913	2173	2984	3158	4076	4185	5375	5622
	max n_{inner}	414	529	624	731	757	858	1004	1080
$P_x \mathcal{L} P_x$	n_{outer}	39	40	42	46	47	48	47	49
	$\sum n_{\text{inner}}$	83	83	91	94	96	101	88	93
	max n_{inner}	14	13	15	13	13	13	12	10

Is it faster?

- Can be solved analytically [V./Vandewalle '10] for $M = I$.
- Assuming $(A + \lambda I)^{-1}$ is $O(n)$, total cost is $O(nk^2) \rightsquigarrow$ AMG.
- Gauss–Newton *as solver* is not efficient (not small residual).

Numerical results for RLyap

Performance of the Riemannian optimization approach is comparable with the state-of-the-art, yet more general.

RLyap compared with CFADI [Penzl '99],[Li/White '04] and KPIK [Simoncini '07] for 2D Laplace, rank one rhs.

		PCG with AMG		
		RLyap	ADI	Krylov
$n = 500^2$	time (s.)	40	70	24
	rank X	12	19	36
$n = 1000^2$	time (s.)	175	310	118
	rank X	12	18	38
$n = 1500^2$	time (s.)	443	811	354
	rank X	12	19	44

Tol. on rel. residual $= 10^{-6}$; $(A + \lambda I)^{-1}$ solved by PCG+AMG.

Numerical results for RLyap

When the r.h.s. C is not of low rank, RLyap can be more efficient.

- Laplace; full matrix C, rank k approximated C_k.

	solver rhs	RLyap C	CF-ADI C_{15}	RLyap C_{15}	CF-ADI C_{30}	RLyap C_{30}
$n = 40\,000$	time (s.)	70.3	(38.7)	48.9	111.2	61.6
$\tau =$ 1e−6	rank X	23	35	25	49	27
	residual	9.87e−7	2.67e−6	9.30e−7	8.61e−7	9.86e−7
$n = 80\,000$	time (s.)	169.7	(103.1)	116.8	(232.1)	128.4
$\tau =$ 1e−6	rank X	25	35	25	50	25
	residual	9.89e−7	2.68e−6	9.81e−7	2.98e−6	9.90e−7
$n = 160\,000$	time (s.)	176.8	139.5	104.7	300.9	125.5
$\tau =$ 5e−5	rank X	12	33	12	48	12
	residual	1.44e−5	3.57e−5	3.35e−5	3.47e−5	1.44e−5

- RLyap can use a matrix-free C, other methods can not.

Thank you for your attention

Accuracy of models with different Hessians

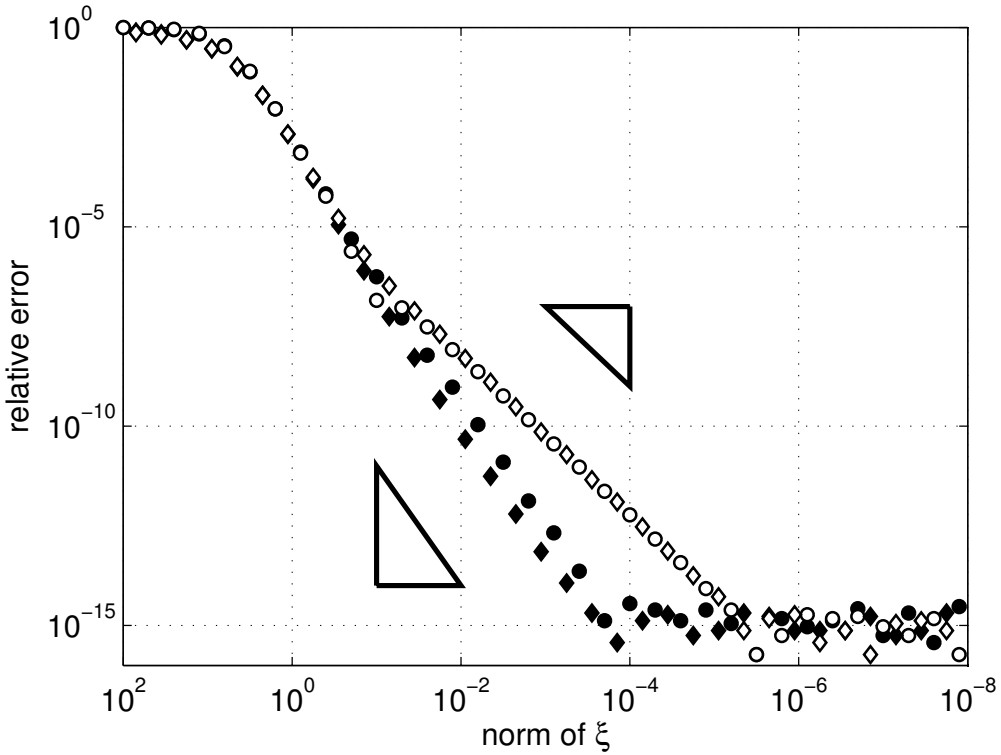


Figure: Relative error of the linear and quadratic models. The triangles indicate the second and third order convergence of the error.

Optimization problem is non-convex $\rightsquigarrow$ robustify Newton by TR!