First-order Riemannian optimization methods for the Karcher mean

Bruno **Iannazzo**, Università di **Perugia**, **Italy**
with Margherita **Porcelli**, Università di **Bologna**, **Italy**

Atlanta, October 26, 2015

The Karcher mean

The Karcher mean of positive definite matrices ($\mathcal{P}_n$)

$$\mathcal{K} : \mathcal{P}_n^m \to \mathcal{P}_n$$

$\mathcal{K}(A_1, \ldots, A_m)$ generalizes the **geometric mean** to matrices.

For positive scalars

$$\mathcal{K}(a_1, a_2, \ldots, a_m) = (a_1 a_2 \cdots a_m)^{1/m},$$

and verifies all other properties required by a mean.

Strong **averaging** and **interpolation** power $\to$ Applications: medical imaging, machine learning, statistic, signal processing…

The Karcher mean

For two matrices there is an explicit expression

$$\mathcal{K}(A, B) = A(A^{-1}B)^{1/2}.$$

For more than two matrices it is defined through a **Riemannian structure** on $\mathcal{P}_n$.

Geometry of $\mathcal{P}_n$

Open subset (**cone**) of the Euclidean space of Hermitian matrices $(\mathbb{H}_n) \implies$ differential structure on $\mathcal{P}_n$.

One coordinate chart is sufficient: $\varphi : \mathcal{P}_n \to \mathbb{H}_n$ such that $\varphi(A) = A$.

Tangent space: $T_X\mathcal{P}_n \cong \mathbb{H}_n$.

A **Riemannian structure** is given by :

$$\langle A, B\rangle_X = \mathrm{trace}(AX^{-1}BX^{-1}), \qquad A, B \in \mathbb{H}_n.$$

The Karcher mean

Given $A_1, \ldots, A_m$ in a Cartan-Hadamard manifold, the problem

$$\operatorname{argmin}_X \sum_{\ell=1}^{m} \delta^2(X, A_\ell)$$

has a **unique solution**, said to be barycenter of $A_1, \ldots, A_m$.

The Karcher mean of $A_1, \ldots, A_m$ is the **barycenter** in $\mathcal{P}_n$ (the Riemannian manifold of positive definite matrices).

Implicit definition, difficult to be handled. No explicit formula is known for $k > 2$.

Computing the Karcher mean

For computing the Karcher mean we have two equivalent approaches:

- Find the **minimum** over $\mathcal{P}_n$ of

$$\delta^2(X, A_1) + \delta^2(X, A_2) + \cdots + \delta^2(X, A_m).$$

- Find the **unique solution** in $\mathcal{P}_n$ of

$$X \log(X^{-1}A_1) + X \log(X^{-1}A_2) + \cdots + X \log(X^{-1}A_m) = 0.$$

There are other approaches, much less effective or accurate.

Computing the Karcher mean

Find the minimum over $\mathcal{P}_n$ of

$$\delta^2(X, A_1) + \delta^2(X, A_2) + \cdots + \delta^2(X, A_m).$$

Two optimization approaches:

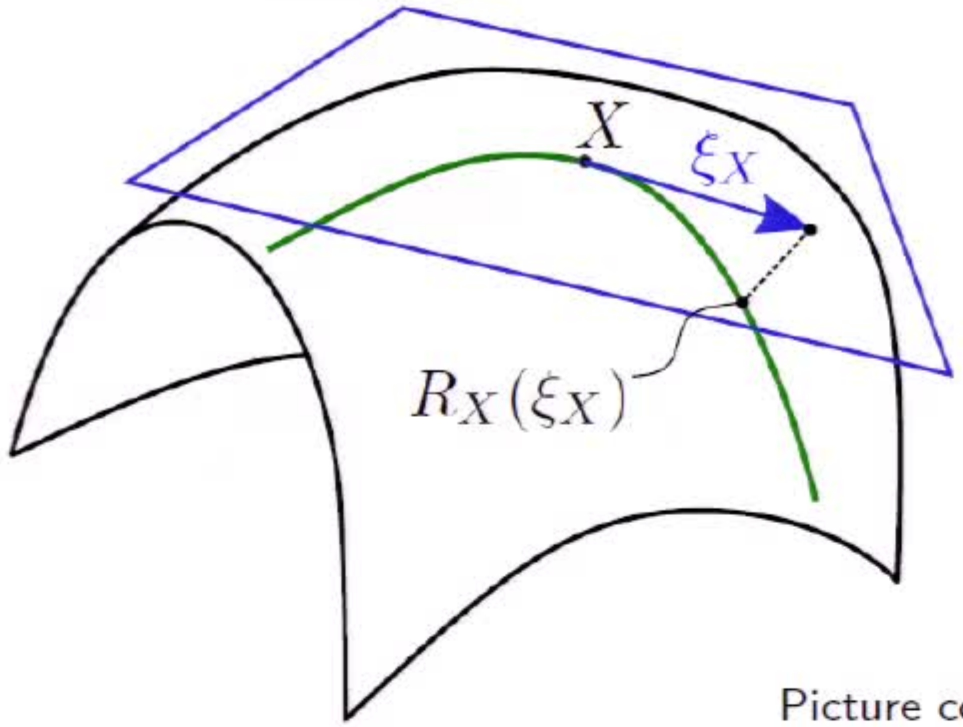
- Standard optimization;
- Riemannian optimization (**exploits the structure**).

The function is not convex, but it is geodesically convex.

Retraction-based optimization on manifolds

Retraction: approximation of the exponential map $R_x : T_x\mathcal{P}_n \to \mathcal{P}_n$.

- Projects the tangent space to the manifold;
- Approximates the exponential $DR_x(0) =$identity.

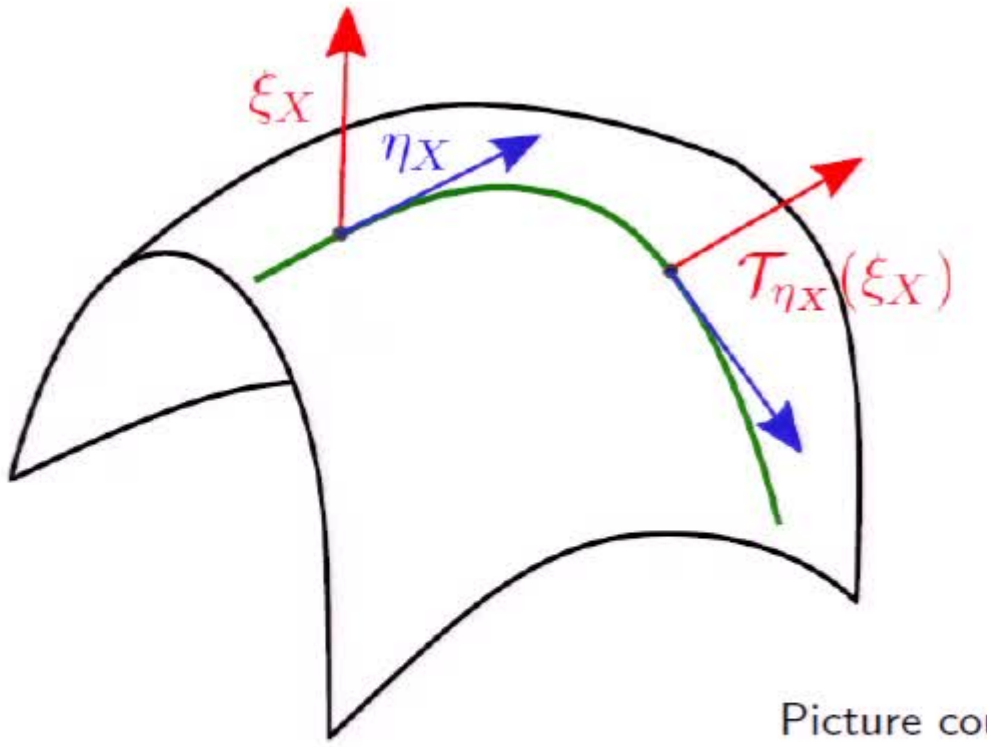


Picture courtesy of B. Jeuris

Retraction-based optimization on manifolds

Vector transport: approximation of the parallel trasport $\mathcal{T}_\eta(\xi) \in T_{R_x(\xi)}\mathcal{P}_n$ for $\eta, \xi \in T_x\mathcal{P}_n$.

- Moves vectors from a tangent space to another;
- Linear map.



Picture courtesy of B. Jeuris

Positive definite matrices and the Karcher mean

Metric: $\langle A, B\rangle_X = \text{trace}(X^{-1}AX^{-1}B)$.

Retraction: exponential map, follow geodesics

$$R_A(X) = A\exp(A^{-1}X).$$

Vector Transport: parallel transport, based on geodesics

$$\mathcal{T}_{A\to B}(X) = (A\#B)A^{-1}XA^{-1}(A\#B),$$

where $A\#B$ is the Karcher mean of A and B.

Euclidean vs. Riemannian gradient descent

First-order optimization: uses only gradients.

Euclidean:

$$X_{k+1} = F(X) := X_k - t_k g(X_k), \qquad g(X) = 2X^{-1}\sum_{j=1}^{m} \log(XA_j^{-1})$$

Riemannian:

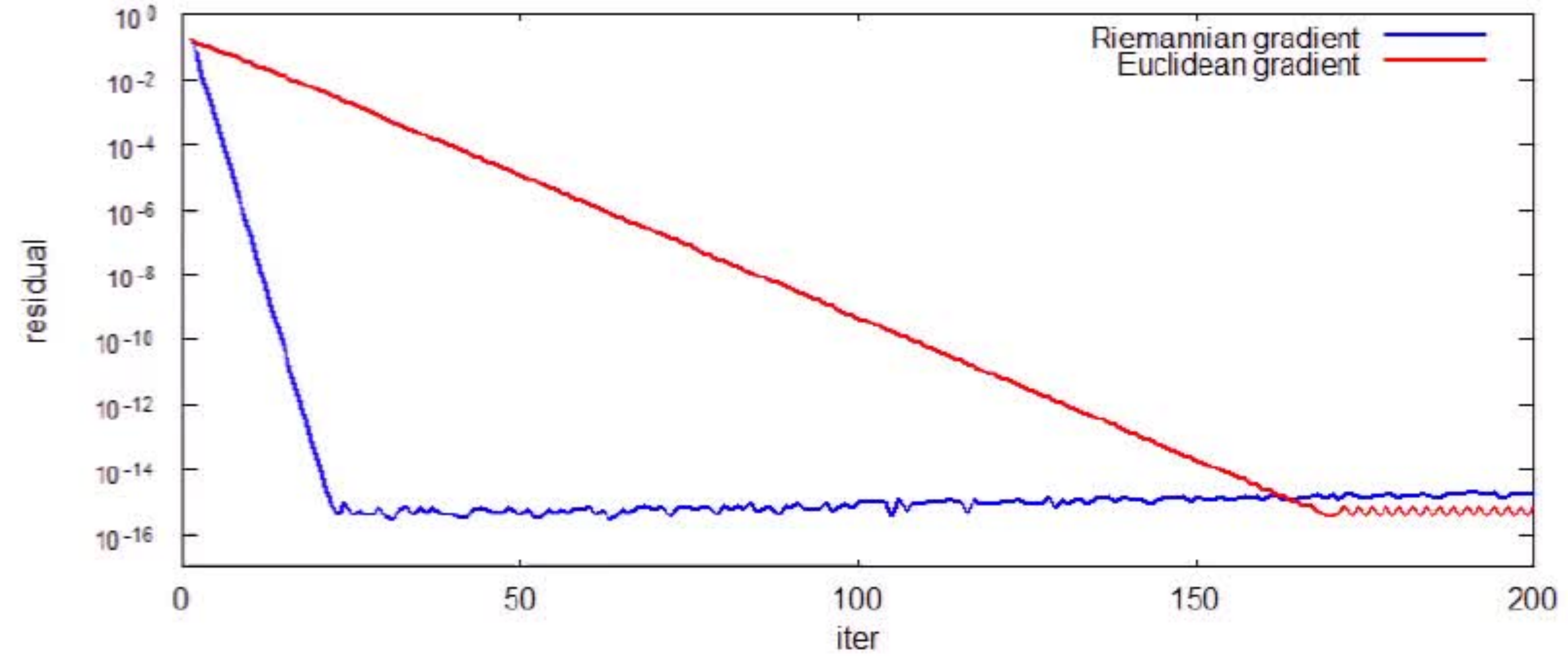
$$X_{k+1} = F(X_k) := X_k \exp(-t_k X_k^{-1} g(X_k)), \quad g(X) = 2X\sum_{i=1}^{m} \log(A_j^{-1}X)$$

Euclidean vs. Riemannian gradient descent

$$A = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}, \quad B = \begin{bmatrix} 3 & 1 \\ 1 & 1 \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 1 \\ 1 & 3 \end{bmatrix},$$

Starting with $X_0 = \frac{1}{3}(A + B + C)$ and $t_k = \theta_k = 1/5$ we try both gradient descent algs. and compute

$$\text{residual} = \| \log(A^{-1}X) + \log(B^{-1}X) + \log(C^{-1}X)\|_F / \|X\|_F$$



Choice of the step-size

Problem of first-order optimization: **choose the step-size**.

- Armijo's rule, reduce the step-size until some global convergence condition is fulfilled.
- Richardson-like iteration, chose the step-size optimal with respect to the derivative of $F(X)$ [Bini-I, 12].

Richardson-like iteration incorporates second-order information with no extra cost.

Second-order (Hessian based) optimization **not recommended in this problem** because too expensive.

The Barzilai-Borwein algorithm

Gradient-type algorithm

$$x_{k+1} = x_k - t_k g_k, \qquad g_k = \nabla f(x_k).$$

Quasi-Newton **secant equation**

$$A_{k+1} s_k = y_k, \qquad s_k := x_{k+1} - x_k, \qquad y_k := g_{k+1} - g_k.$$

Imposing $A_{k+1} = \alpha I$, we have

$$s_k \alpha = y_k$$

whose **least squares solution** gives (with $x_{k+1} \neq x_k$, $\alpha \neq 0$)

$$t_{k+1}^{BB} = \frac{1}{\alpha} = \frac{s_k^T s_k}{s_k^T y_k}.$$

The Riemannian Barzilai-Borwein algorithm

Gradient-type algorithm.

In Euclidean optimization

$$x_{k+1} = x_k - t_k g_k, \qquad g_k = \nabla f(x_k) \in \mathbb{R}^n$$

In Riemannian optimization

$$x_{k+1} = R_{x_k}(-t_k g_k), \qquad g_k = \nabla^{(\mathcal{R})} f(x_k) \in T\mathcal{M}_{x_k}$$

The Riemannian Barzilai-Borwein algorithm

Quasi-Newton secant equation.

In Euclidean optimization

$$A_{k+1}s_k = y_k, \qquad s_k := x_{k+1} - x_k, \qquad y_k := g_{k+1} - g_k.$$

In Riemannian optimization, approximation of the Hessian at $T\mathcal{M}_{x_{k+1}}$

$$A_{k+1}s_k = y_k, \quad s_k := \mathcal{T}_{x_k \to x_{k+1}}(-t_k g_k), \quad y_k := g_{k+1} - \mathcal{T}_{x_k \to x_{k+1}}(g_k).$$

The Riemannian Barzilai-Borwein algorithm

Least squares solution to $s_k \alpha = y_k$.

In Euclidean optimization

$$t_{k+1}^{BB} = \frac{1}{\alpha} = \frac{s_k^T s_k}{s_k^T y_k}.$$

In Riemannian optimization

$$t_{k+1}^{BB} = \frac{1}{\alpha} = \frac{\langle s_k, s_k \rangle_{x_{k+1}}}{\langle s_k, y_k \rangle_{x_{k+1}}}.$$

The Riemannian Barzilai-Borwein algorithm

Gradient-type algorithm

$$x_{k+1} = R_{x_k}(-t_k g_k), \qquad g_k = \nabla^{(\mathcal{R})} f(x_k) \in T\mathcal{M}_{x_k}$$

Quasi-Newton **secant equation**

$$A_{k+1} s_k = y_k, \quad s_k := \mathcal{T}_{x_k \to x_{k+1}}(-t_k g_k), \quad y_k := g_{k+1} - \mathcal{T}_{x_k \to x_{k+1}}(g_k).$$

Imposing $A_{k+1} = \alpha I$, we have

$$s_k \alpha = y_k$$

whose **least squares solution** gives (with $x_{k+1} \neq x_k$, $\alpha \neq 0$)

$$t_{k+1}^{BB} = \frac{1}{\alpha} = \frac{\langle s_k, s_k \rangle_{x_{k+1}}}{\langle s_k, y_k \rangle_{x_{k+1}}}.$$

Global convergence

Armijo's line search. Given t_k, $\sigma, \gamma \in (0,1)$ choose the smallest nonnegative h such that

$$f(R_{x_k}(\sigma^h t_k g_k)) \leq f(x_k) + \gamma \sigma^h t_k \langle g_k, g_k \rangle_{x_k}.$$

Armijo's line search reduces BB to steepest descent.

Nonmonotone line search. Require decrease not of f but of the maximum of f over the last $M > 1$ steps.

$$f(R_{x_k}(\sigma^h t_k g_k)) \leq f_{\max} + \gamma \sigma^h t_k \langle g_k, g_k \rangle_{x_k},$$

where $f_{\max} = \max_{1 \leq i \leq \min\{M, k+1\}} \{f(x_{k+1-i})\}$.

Global convergence theorem

Theorem

Under the assumptions

1. *f is bounded on $\{x \in \mathcal{M} : f(x) \leq f(x_0)\}$;*
2. *the domain of the retraction R is the whole tangent bundle.*

Let $\{x_k\}$ be generated by the BB algorithm with nomonotone line search. Every limit point of the sequence is stationary.

Computation: basic operations

1. **Function of simmetric matrices** (F)

$$Af(A^{-1}B),$$

with $A \in \mathcal{P}_n$ and $B \in \mathbb{H}_n$.

Using the Cholesky factorization $A = R^*R$, and using the spectral decomposition of $R^{-*}BR^{-1} = UDU^*$

$$Af(A^{-1}B) = R^*U\text{diag}(f(d_{11}), \dots, f(d_{nn}))U^*R$$

Costs $12.6n^3$ ops (reduced to $12.3n^3$ ops for $B \in \mathcal{P}_n$)

Computation: basic operations

2. **Riemannian distance** (D)

$$\| \log(A^{-1/2} B A^{-1/2}) \|_F$$

with $A, B \in \mathcal{P}_n$.

By a similar idea can be computed with $2.6n^3$ ops.

Computation

Gradient computation	mF	$12.3mn^3 ops$
Retraction	$1F$	$12.6n^3 ops$
Parallel transport	$1F$	$12.6n^3 ops$
Cost function evaluation	mD	$2.6mn^3$ ops

m = number of matrices;
n = size of matrices;
F = function of symmetric matrices;
D = distance.

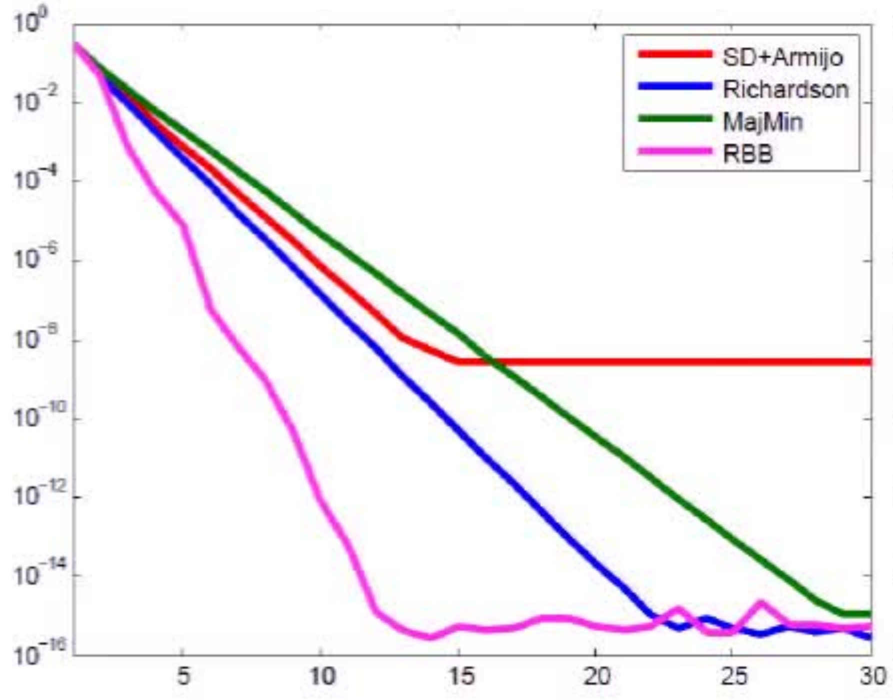
Computational costs

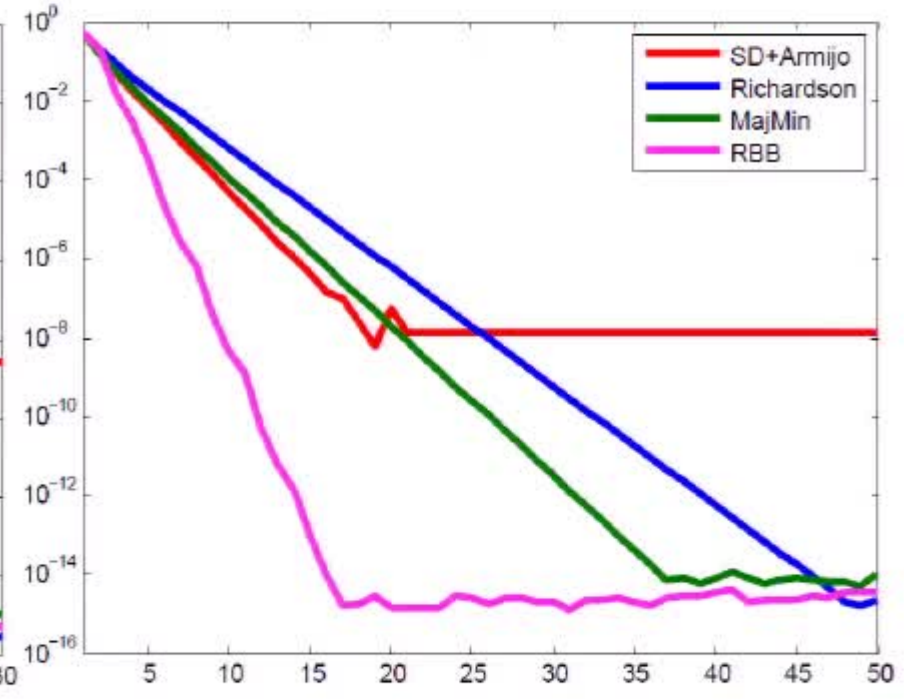
Algorithm	One step	Backtracking
SD + Armijo's rule [1,2]	$(m+1)F$	mD
Richardson-like [3]	$(m+1)F$	
MajMin [4]	$(m+2)F$	
BB nonmonotone	$(m+2)F$	mD
BB	$(m+2)F$	

References

[1] Geomean by Jeuris-Vandebril-Vandereycken
[2] ManOpt by Boumal-Mishra-Absil-Sepulchre
[3] MMToolbox by Bini-Iannazzo
[4] find_mean_MM by Zhang

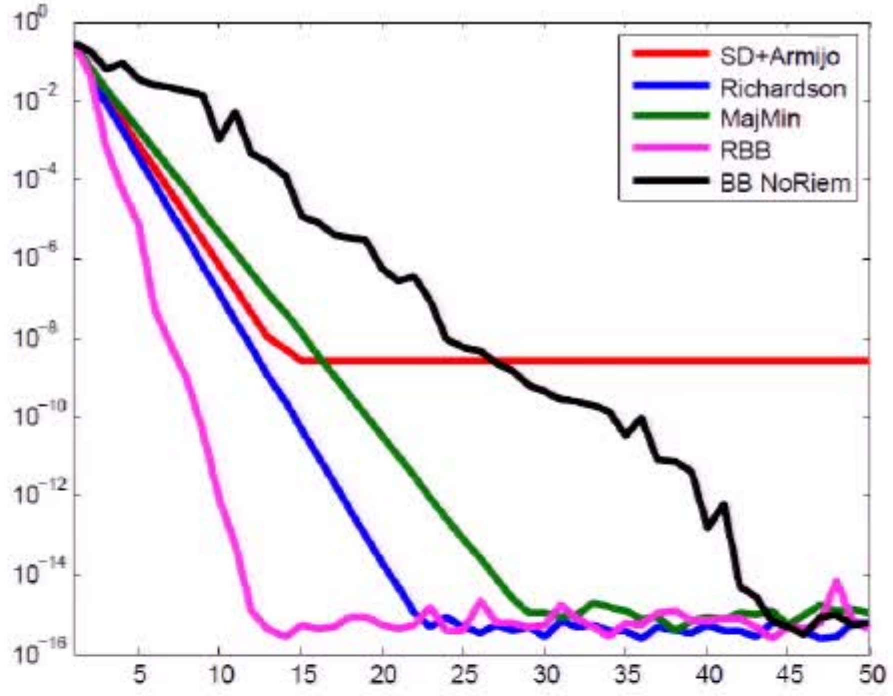
Experiments: random matrices





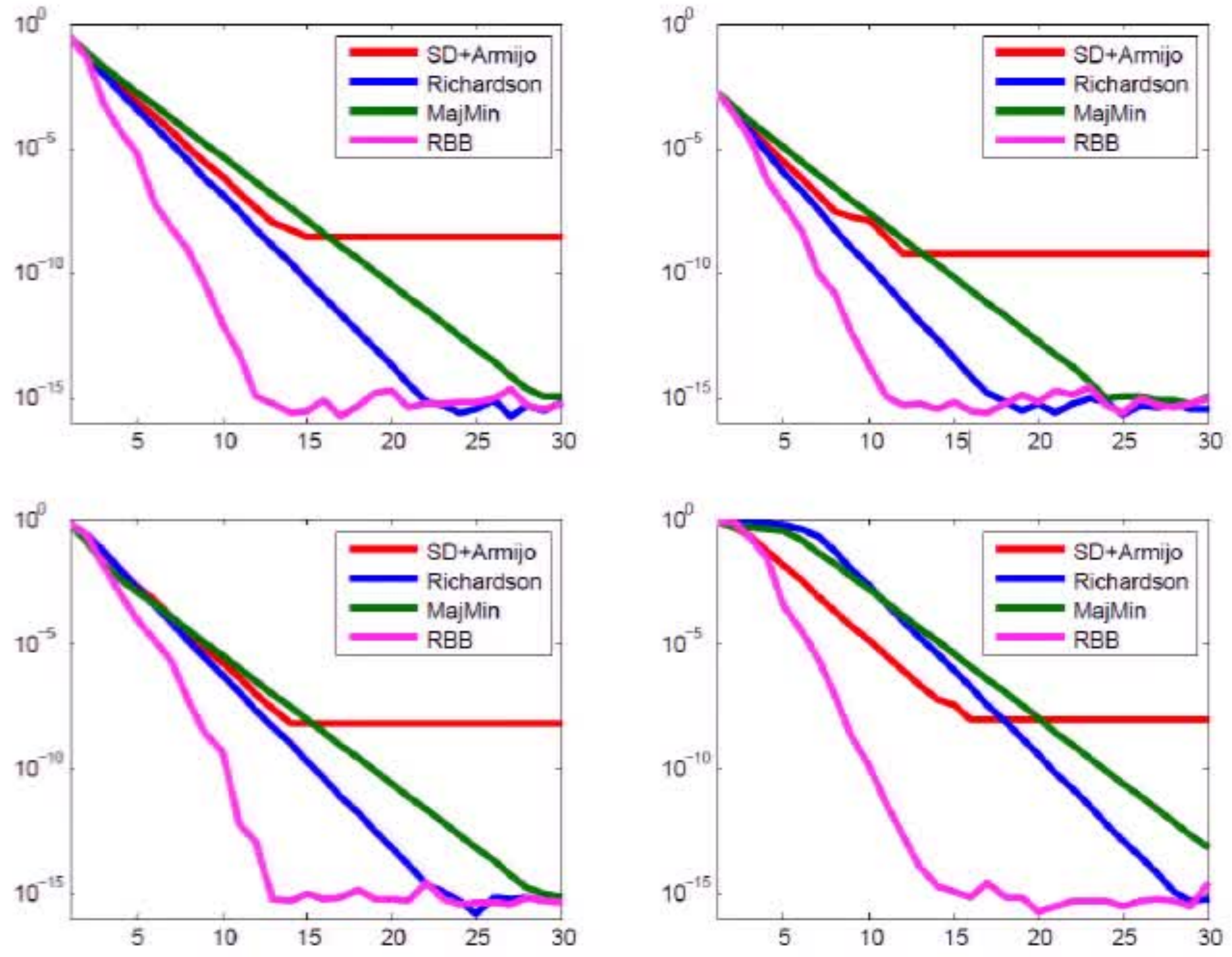
Left. $m = 3$, $n = 3$.
Right. $m = 10$, $n = 10$.
X_0 is the arithmetic mean.

Experiments: random matrices



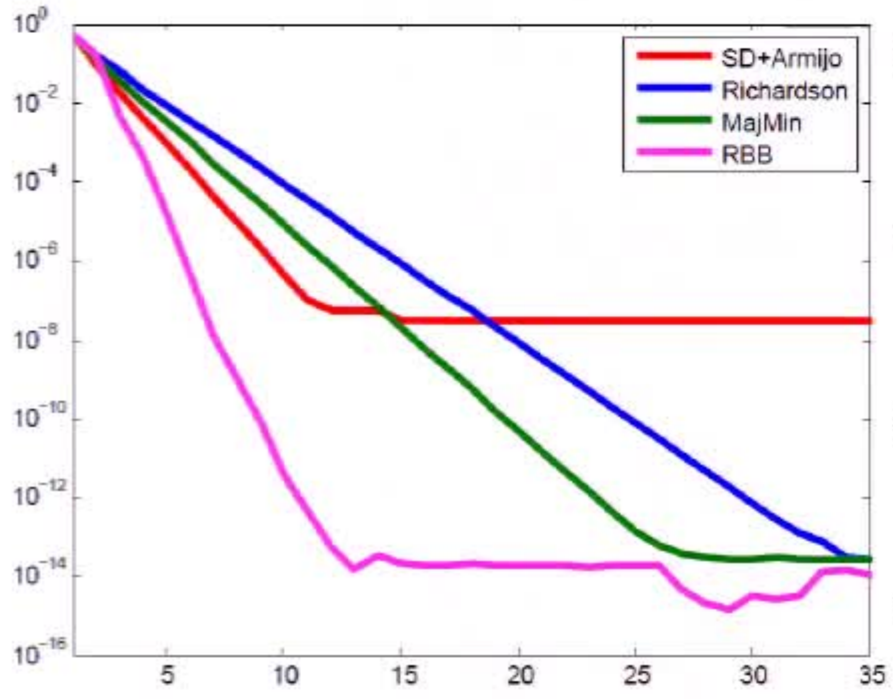
Comparison with standard (non-Riemannian) Barzilai-Borwein.

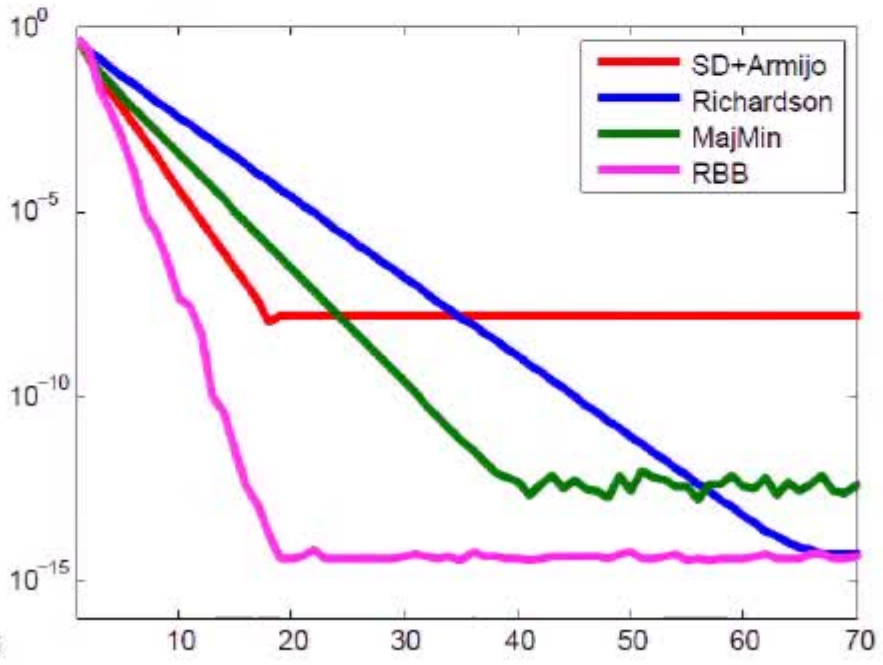
Experiments: random matrices



Inital value: arithmetic mean, cheap mean (near to the Karcher mean), random matrix, ill-conditioned random matrix.

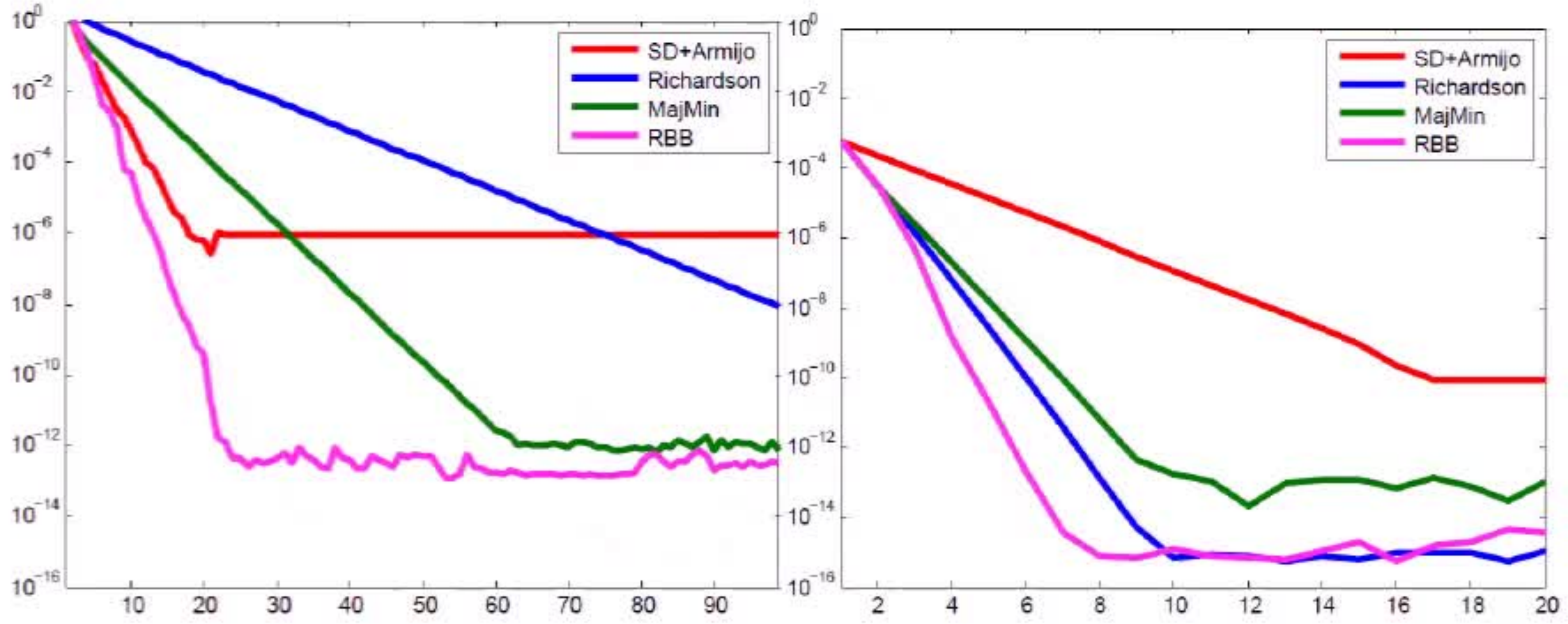
Experiments: random matrices





Left. $m = 100$, $n = 5$.
Right. $m = 5$, $n = 100$.

Experiments: ill-conditioned matrices



Left. Matrices with condition number 10^5, $m = 10$, $n = 10$.
Right. Matrices very near to each other, and far from identity, $m = 10$, $n = 10$.

Conclusions

We have adapted the Barzilai-Borwein method to retraction-based Riemannian optimization, with a globalization strategy.

- Requires **only** first-order information;
- it is **faster** than existing algorithms for the Karcher mean.

Open problem

- Explain why the proposed algorithm **does not require** globalization in practice.