A Brief Introduction to Optimization Algorithms on Matrix Manifolds

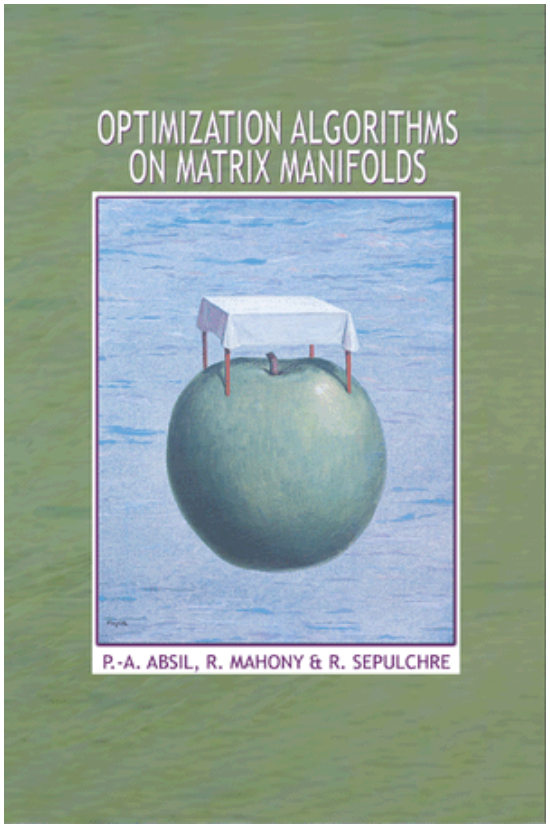
James Folberth

University of Colorado at Boulder

11 October, 2016

A Recent Textbook

Most of the material in this talk is from *Optimization Algorithms on Matrix Manifolds*, P.-A. Absil, R. Mahoney, R. Sepulchre, Princeton University Press, 2008.



1. Introduction
2. Motivation and Applications
3. Matrix Manifolds: First-Order Geometry
4. Line-Search Algorithms on Manifolds
5. Matrix Manifolds: Second-Order Geometry
6. Newton's Method
7. Trust-Region Methods
8. A Constellation of Superlinear Algorithms

Outline

1. A Motivating Example
2. Manifolds
3. Gradient Descent
4. Manopt
5. Some Recent Work

Extreme Eigenspace Computation

Consider a symmetric matrix $A \in \mathbb{R}^{n \times n}$ with eigenvalues $\lambda_1 \leq \cdots \leq \lambda_n$.

Proposition

Consider the generalized Rayleigh quotient

$$f : \mathbb{R}_*^{n \times p} \to \mathbb{R} : Y \mapsto f(Y) = \mathrm{tr}(Y^T A Y (Y^T Y)^{-1})$$

defined on $\mathbb{R}_*^{n \times p}$, the set of all real $n \times p$ full-rank matrices. TFAE:

1. *Y_* is a global minimizer of $f(Y)$ over $\mathbb{R}_*^{n \times p}$;*
2. *$\mathrm{span}(Y_*)$ is a leftmost invariant subspace of A;*
3. *$f(Y_*) = \sum_{i=1}^{p} \lambda_i$.*

Fact: $f(Y)$ depends only on $\mathrm{span}(Y)$.

Rayleigh Quotient

For simplicity, take $p = 1$ and assume λ_1 has multiplicity 1.

$$f : \mathbb{R}_*^n \to \mathbb{R} : y \mapsto f(y) = \frac{y^T A y}{y^T y}.$$

$\mathbb{R}_*^n$ is $\mathbb{R}^n$ with the origin removed[1].

[1] $\mathbb{R}_*^{n \times p}$ - full-rank, $n \times p$ matrices

Rayleigh Quotient

For simplicity, take $p = 1$ and assume λ_1 has multiplicity 1.

$$f : \mathbb{R}_*^n \to \mathbb{R} : y \mapsto f(y) = \frac{y^T A y}{y^T y}.$$

$\mathbb{R}_*^n$ is $\mathbb{R}^n$ with the origin removed[1].

The minimizers of $f(y)$ are *not* isolated: they appear as the continuum $v_1 \mathbb{R}_* = \{v_1 r : 0 \neq r \in \mathbb{R}\}$.

Consequently, important optimization algorithms may not apply.

For every non-critical y, Newton's method gives

$$y \mapsto 2y.$$

[1] $\mathbb{R}_*^{n \times p}$ - full-rank, $n \times p$ matrices

Rayleigh Quotient

A standard remedy is to restrict the Rayleigh quotient to the unit sphere

$$S^{n-1} := \left\{ y \in \mathbb{R}^n \,:\, y^T y = 1 \right\}.$$

Minimizers are now isolated, and we must work on the sphere.

Embedded submanifold

Treat all points on a two-sided ray $y\mathbb{R}_*$[1] as a single point (equivalence class).

$$\mathcal{M} = \{ y\mathbb{R}_* \,:\, y \in \mathbb{R}_*^n \},$$

the set of all 1-dimensional subspaces of $\mathbb{R}^n$.

Minimizers are isolated, but points are now equivalence classes.

Quotient manifold

[1] $\mathbb{R}_*^{n\times p}$ - full-rank, $n \times p$ matrices

Rayleigh Quotient

A standard remedy is to restrict the Rayleigh quotient to the unit sphere

$$S^{n-1} := \left\{ y \in \mathbb{R}^n \,:\, y^T y = 1 \right\}.$$

Minimizers are now isolated, and we must work on the sphere.

Embedded submanifold

Treat all points on a two-sided ray $y\mathbb{R}_*$[1] as a single point (equivalence class).

$$\mathcal{M} = \{ y\mathbb{R}_* \,:\, y \in \mathbb{R}^n_* \},$$

the set of all 1-dimensional subspaces of $\mathbb{R}^n$.

Minimizers are isolated, but points are now equivalence classes.

Quotient manifold

[1] $\mathbb{R}^{n\times p}_*$ - full-rank, $n \times p$ matrices

Some Benefits

For optimization-based eigenvalue algorithms:

- Solid framework for convergence analysis. Many algorithms exhibit almost global convergence.
- Speed of convergence is intrinsic to the algorithm. Gradient-based is linear; Newton-like are superlinear.
- Matrix-free methods, so A is used only as an operator $x \mapsto Ax$.

Outline

1 A Motivating Example

2 Manifolds

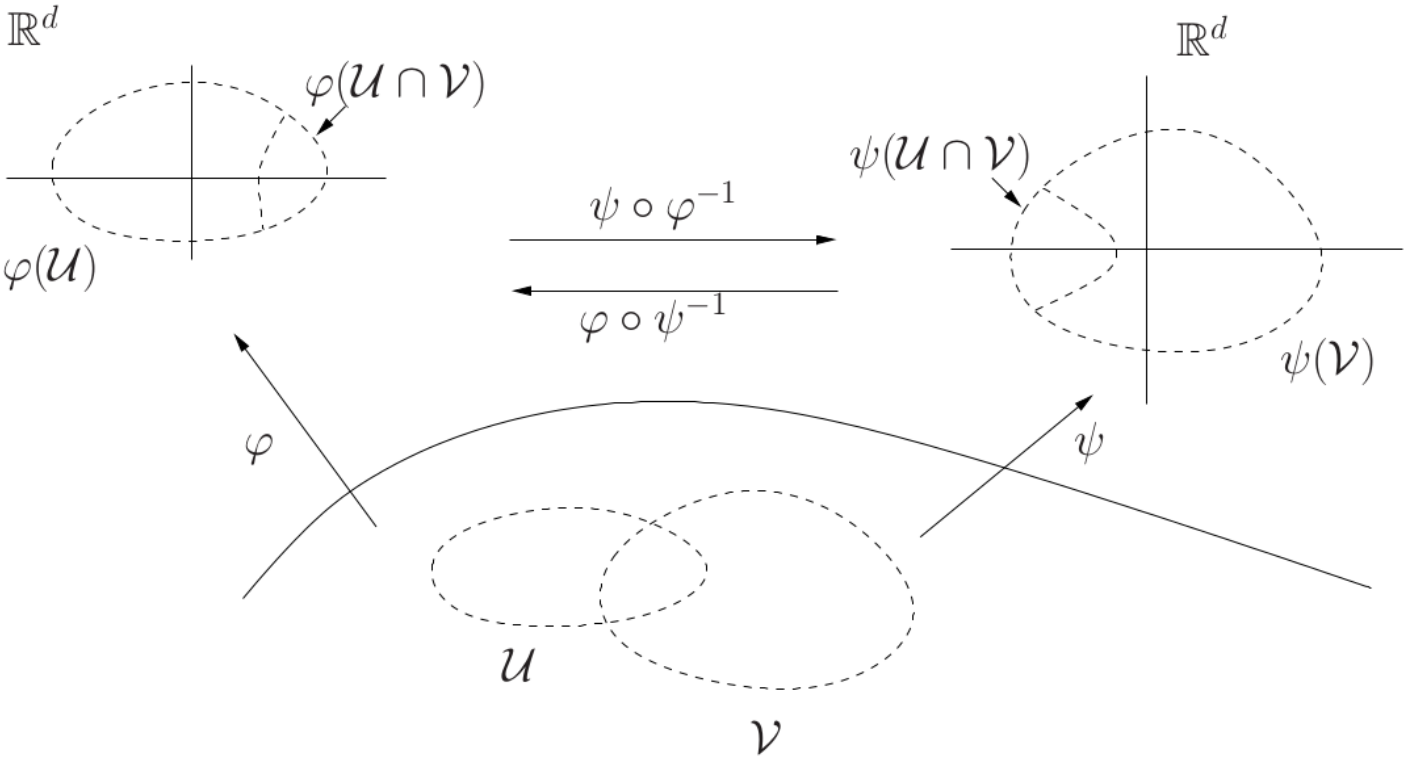
3 Gradient Descent

4 Manopt

5 Some Recent Work

What is a Smooth Manifold?

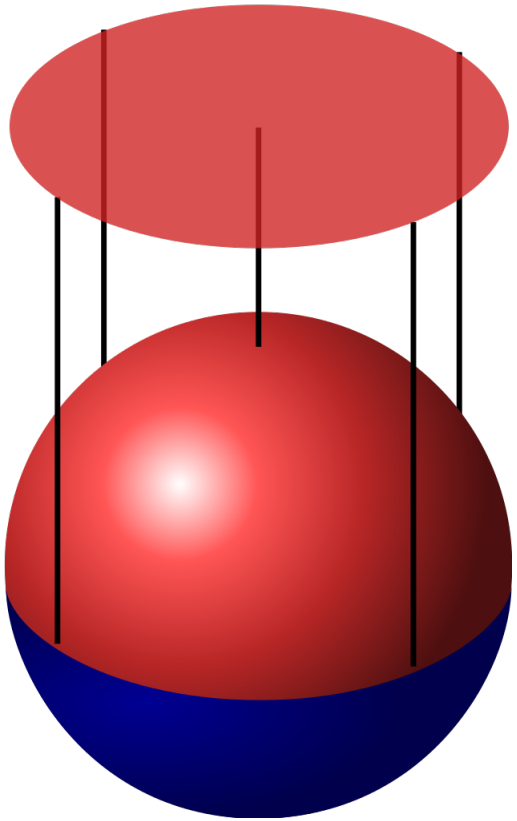
Very roughly, a (possibly) non-Euclidean space that locally resembles Euclidean space.



Some Common Manifolds

$\mathbb{R}^n, \mathbb{R}^{n\times p}$ Linear manifold viewed as vector space endowed with inner product (a Euclidean space).

Sphere $S^{n-1} := \left\{x \in \mathbb{R}^n : x^T x = 1\right\}$.



Some Common Manifolds

Stiefel Manifold $\mathrm{St}(p,n) := \{X \in \mathbb{R}^{n\times p} : X^TX = I_p\}$

X is tall-skinny, has orthonormal columns

$St(1,n) = S^{n-1}$

$St(n,n) = O_n$

Applications in computer vision, PCA, ICA.

Grassmann Manifold $\mathrm{Grass}(p,n) \simeq \mathbb{R}_*^{n\times p}/\mathrm{GL}_p$

The set of all p-dimensional subspaces in $\mathbb{R}^n$.

$\mathrm{Grass}(1,n) = \mathbb{RP}^{n-1}$

$\mathrm{Grass}(n,n) = \mathrm{GL}_n$

Applications in various dimension reduction problems.

There are many more.

$\mathbb{R}_*^{n\times p}$ - full-rank, $n \times p$ matrices

Some Common Manifolds

Stiefel Manifold $\mathrm{St}(p,n) := \{X \in \mathbb{R}^{n\times p} : X^TX = I_p\}$

X is tall-skinny, has orthonormal columns

$St(1,n) = S^{n-1}$

$St(n,n) = O_n$

Applications in computer vision, PCA, ICA.

Grassmann Manifold $\mathrm{Grass}(p,n) \simeq \mathbb{R}_*^{n\times p}/\mathrm{GL}_p$

The set of all p-dimensional subspaces in $\mathbb{R}^n$.

$\mathrm{Grass}(1,n) = \mathbb{RP}^{n-1}$

$\mathrm{Grass}(n,n) = \mathrm{GL}_n$

Applications in various dimension reduction problems.

There are many more.

$\mathbb{R}_*^{n\times p}$ - full-rank, $n \times p$ matrices

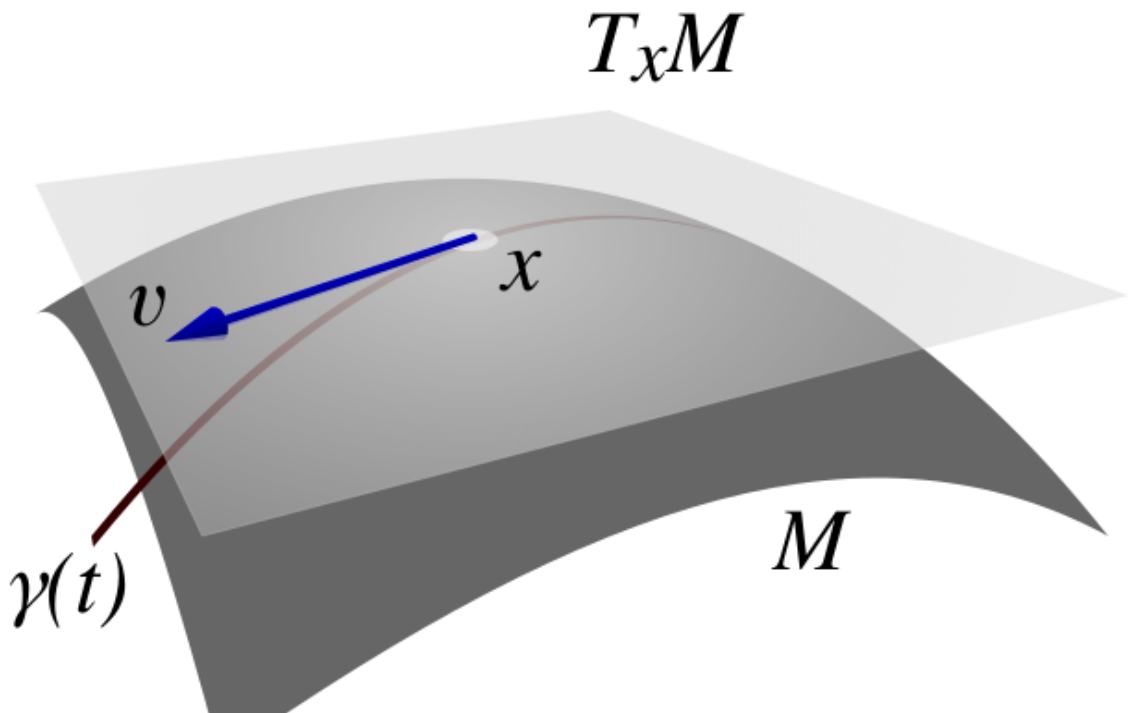
Tangent Vectors

Let $\gamma : \mathbb{R} \to \mathcal{M}$ be a curve on $\mathcal{M}$ s.t. $\gamma(0) = x$.

Can't always define tangent vectors as

$$\gamma'(0) = \lim_{\tau \to 0} \frac{\gamma(\tau) - \gamma(0)}{\tau},$$

as this requires a vector space structure on $\mathcal{M}$.

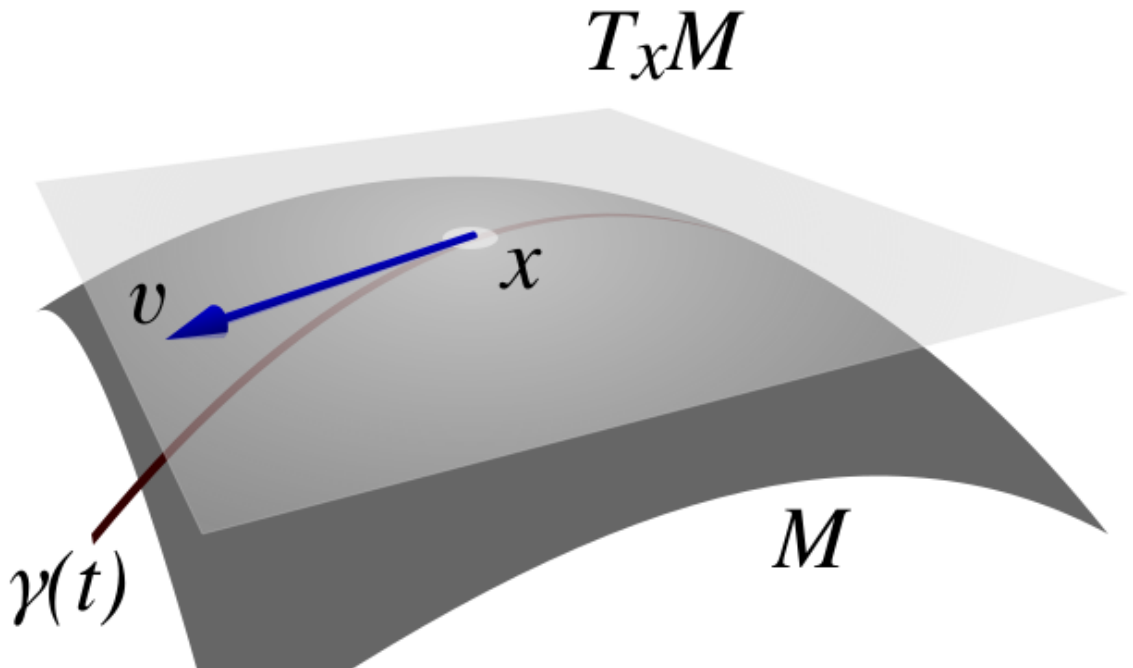


Tangent Vectors

A tangent vector ξ_x is a mapping that differentiates functions along curves:

$$\xi_x f = \dot{\gamma}(0) f := \left.\frac{\mathrm{d}(f(\gamma(t)))}{\mathrm{d}t}\right|_{t=0} = \lim_{\tau \to 0} \frac{f(\gamma(\tau)) - f(\gamma(0))}{\tau},$$

for all smooth $f : \mathcal{M} \to \mathbb{R}$ along the curve $\gamma(t)$.
Tangent space at x: $T_x\mathcal{M} = \{\xi_x\}$, is a *vector space*.



Tangent Vectors

Tangent vectors are used to generalize the directional derivative,

$$Df(x)[\xi_x] = \lim_{t \to 0} \frac{f(x + t\xi_x) - f(x)}{t},$$

where $x + t\xi_x$ only makes sense for vector spaces.

More generally, tangent vectors compute directional derivatives:

$$\begin{aligned} Df(x)[\xi_x] &= \xi_x f \\ &= \dot{\gamma}(0) f = \left. \frac{\mathrm{d}(f(\gamma(t)))}{\mathrm{d}t} \right|_{t=0}. \end{aligned}$$

Riemannian Metric

A Riemannian metric is a smoothly varying *inner product* on the tangent spaces of $\mathcal{M}$:

$$g(\xi_x, \zeta_x) = g_x(\xi_x, \zeta_x).$$

A Euclidean space $\mathcal{E}$ is a particular *Riemannian manifold*.

A Riemannian metric allows us to compute

- lengths

$$L(\gamma) = \int_a^b \sqrt{g(\dot{\gamma}(t), \dot{\gamma}(t))} \mathrm{d}t.$$

- distances (a metric space *metric*)

$$\mathrm{dist} : \mathcal{M} \times \mathcal{M} \to \mathbb{R} : \mathrm{dist}(x, y) = \inf_{\Gamma_{x \to y}} L(\gamma)$$

Riemannian Metric

A Riemannian metric is a smoothly varying *inner product* on the tangent spaces of $\mathcal{M}$:

$$g(\xi_x, \zeta_x) = g_x(\xi_x, \zeta_x).$$

A Euclidean space $\mathcal{E}$ is a particular *Riemannian manifold*.
A Riemannian metric allows us to compute

- lengths

$$L(\gamma) = \int_a^b \sqrt{g(\dot{\gamma}(t), \dot{\gamma}(t))} \mathrm{d}t.$$

- distances (a metric space *metric*)

$$\mathrm{dist} : \mathcal{M} \times \mathcal{M} \to \mathbb{R} : \mathrm{dist}(x, y) = \inf_{\Gamma_{x \to y}} L(\gamma)$$

Riemannian Metric

A Riemannian metric is a smoothly varying *inner product* on the tangent spaces of $\mathcal{M}$:

$$g(\xi_x, \zeta_x) = g_x(\xi_x, \zeta_x).$$

A Euclidean space $\mathcal{E}$ is a particular *Riemannian manifold*.
A Riemannian metric allows us to compute

- lengths

$$L(\gamma) = \int_a^b \sqrt{g(\dot{\gamma}(t), \dot{\gamma}(t))} \mathrm{d}t.$$

- distances (a metric space *metric*)

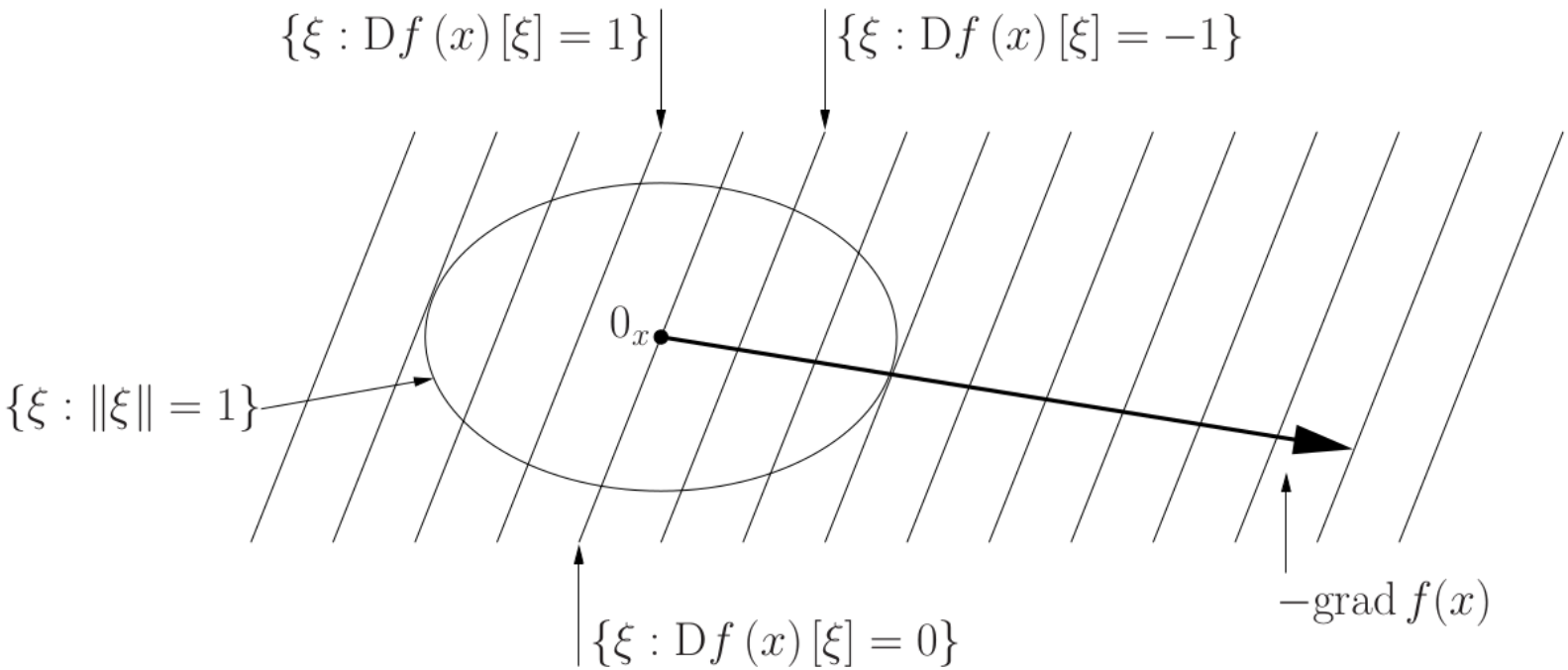
$$\mathrm{dist} : \mathcal{M} \times \mathcal{M} \to \mathbb{R} : \mathrm{dist}(x, y) = \inf_{\Gamma_{x \to y}} L(\gamma)$$

Riemannian Metric

- gradients: The unique element of $T_x\mathcal{M}$ that satisfies

$$g_x(\operatorname{grad} f(x), \xi) = Df(x)[\xi], \quad \forall \xi \in T_x\mathcal{M}.$$

Can compute the Euclidean gradient $\nabla \bar{f}(x)$ and convert to Riemannian gradient $\operatorname{grad} f(x)$.



Outline

1 A Motivating Example

2 Manifolds

3 Gradient Descent

4 Manopt

5 Some Recent Work

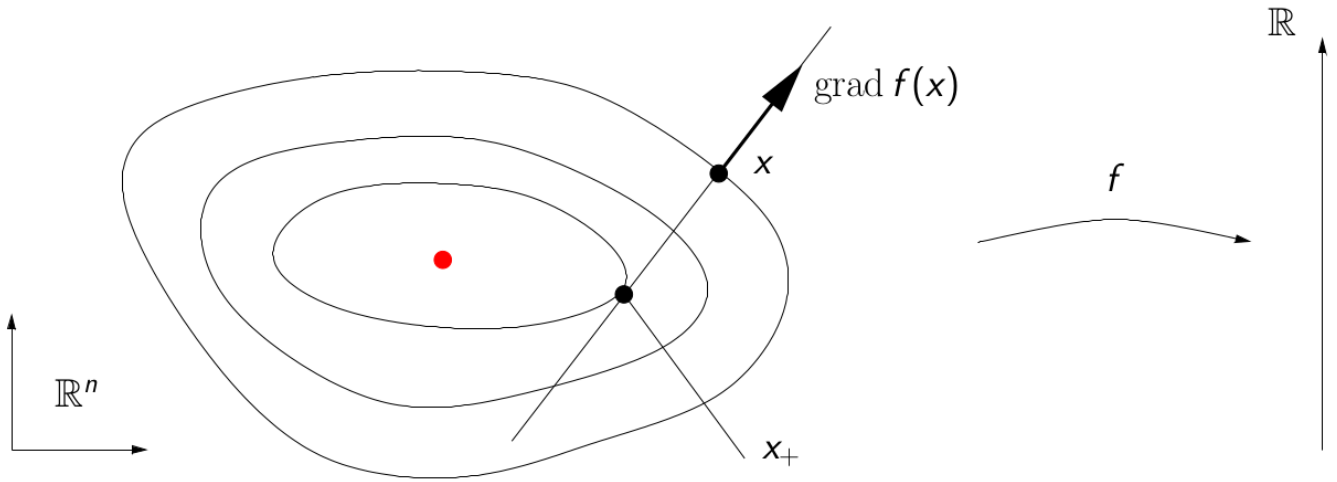
Gradient Descent on $\mathbb{R}^n$

Consider gradient descent on $\mathbb{R}^n$. Initialize $x \in \mathbb{R}^n$ and iterate

$$x_+ = x - t\nabla f(x).$$

The idea is simple: head in the locally most promising direction $(-\nabla f(x))$ for an appropriate distance.

Line-search over the curve $\gamma(t) = x - t\nabla f(x)$.

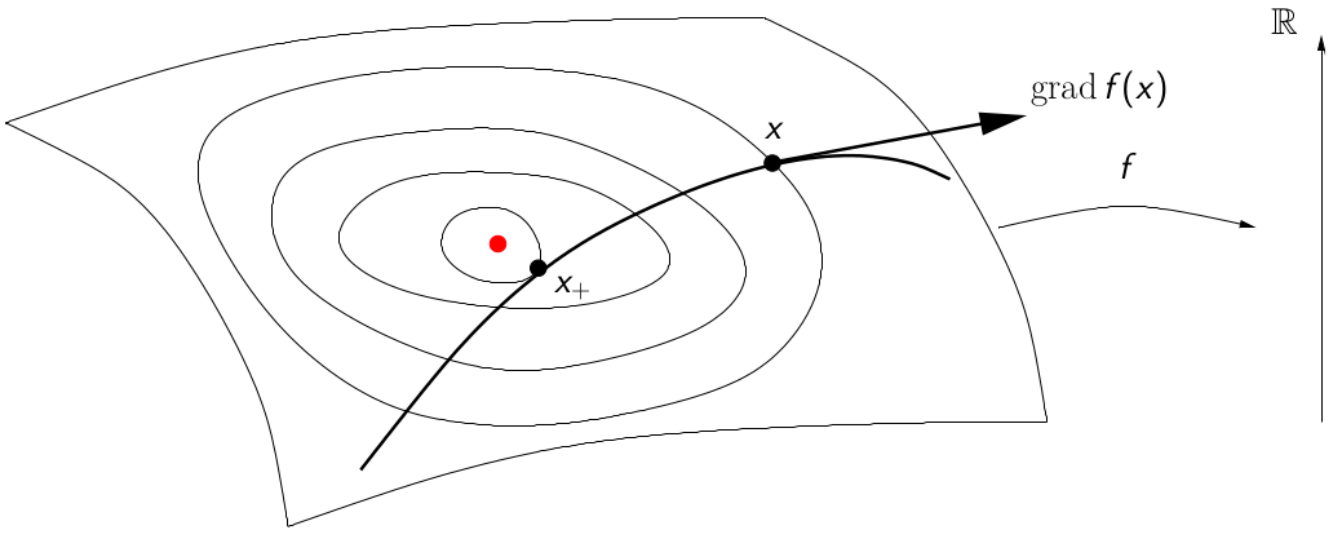


Gradient Descent on $\mathcal{M}$

Head in the direction of $-\operatorname{grad} f(x)$.

Line-search over a curve s.t. $\gamma(0) = x$ and $\dot{\gamma}(0) = -\operatorname{grad} f(x)$.

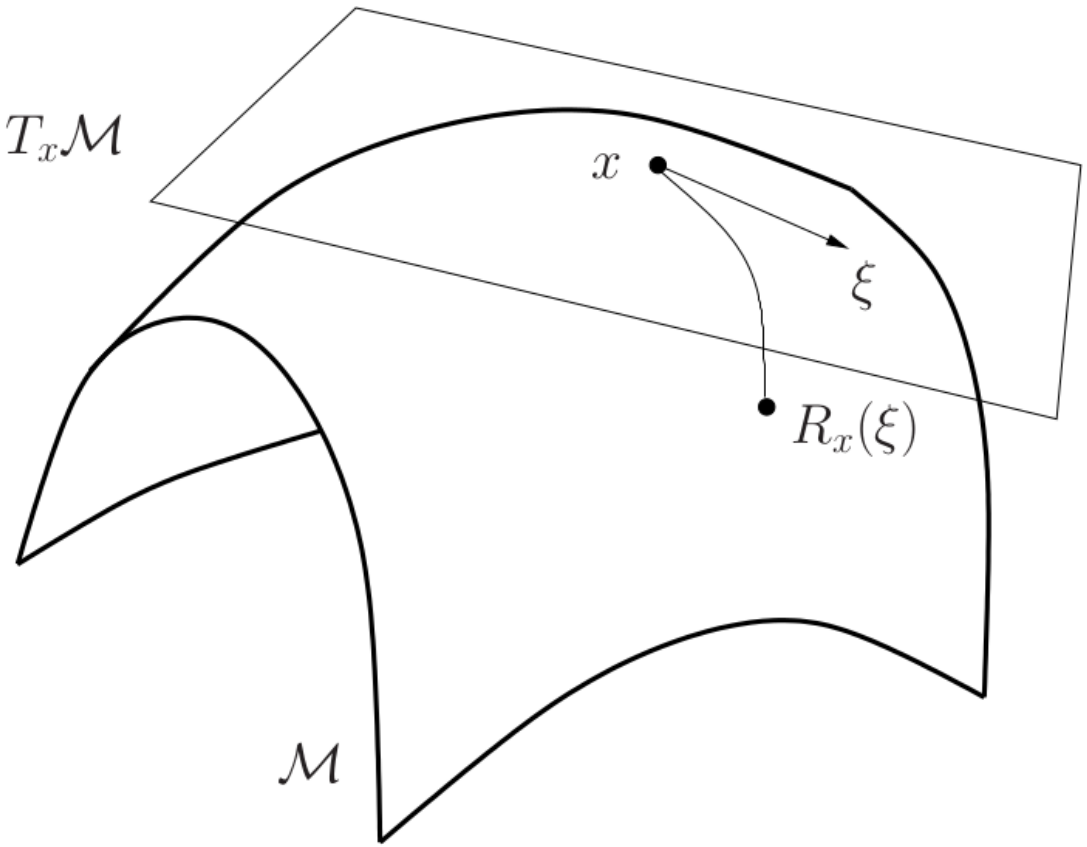
We need a method to cheaply find suitable curves (geodesics work, but are generally expensive to find).



Retractions

A *retraction* $R : T\mathcal{M} = \bigcup_x T_x\mathcal{M} \to \mathcal{M}$ satisfies

1. $R(0_x) = x$, where 0_x is the zero element of $T_x\mathcal{M}$
2. $\left.\frac{d}{dt} R(t\xi_x)\right|_{t=0} = \xi_x \quad \forall \xi_x \in T_x\mathcal{M}.$

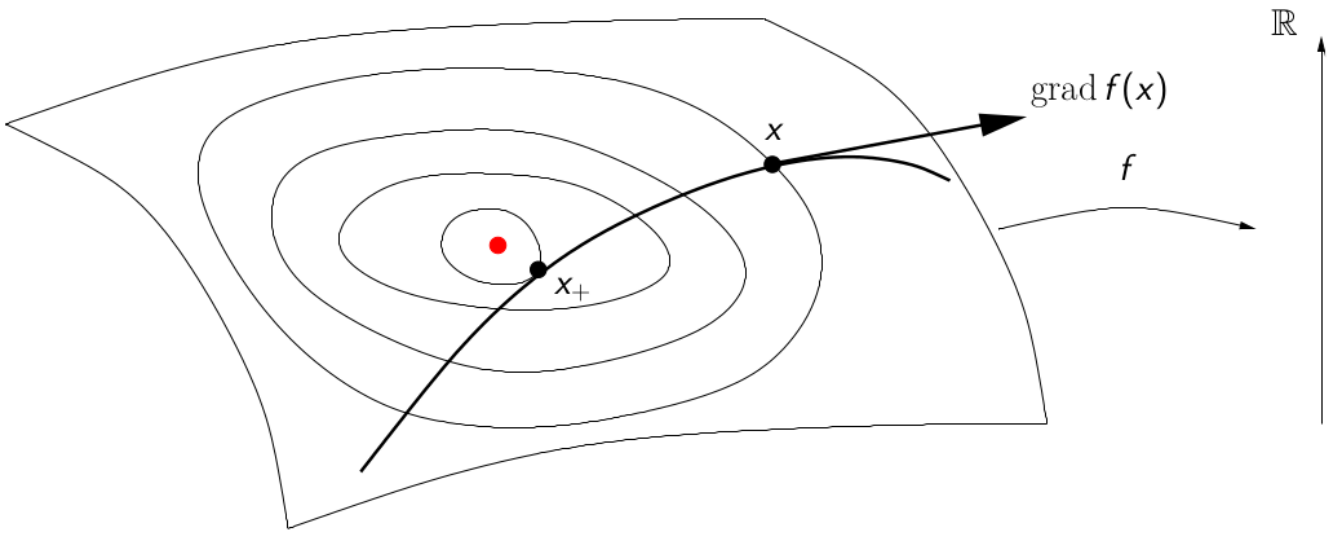


Retractions

Equipped with a retraction R, the curve $\gamma(t) = R(-t \operatorname{grad} f(x))$ satisfies

1. $\gamma(0) = x$;
2. $\dot{\gamma}(0) = -\operatorname{grad} f(x)$.

Along the curve, $f(\gamma(t))$ is a smooth function from $\mathbb{R}$ to $\mathbb{R}$. Can use standard line-search methods (e.g. Armijo backtracking).



Gradient Descent for the Rayleigh Quotient on S^{n-1}

Consider the sphere S^{n-1}. We can view $x \in S^{n-1}$ as an element of $\mathbb{R}^n$, and $\xi \in T_x S^{n-1}$ as an element of $T_x \mathbb{R}^n \simeq \mathbb{R}^n$.

A suitable retraction on S^{n-1} is the projection/renormalization:

$$R_x(\xi) = \frac{x + \xi}{\|x + \xi\|}.$$

Consider the Rayleigh quotient on the sphere:

$$f : S^{n-1} \to \mathbb{R} : x \mapsto f(x) = x^T A x.$$

It is easy to show

$$\operatorname{grad} f(x) = (I - xx^T)2Ax = 2(Ax - xx^T Ax).$$

Gradient Descent for the Rayleigh Quotient on S^{n-1}

Consider the sphere S^{n-1}. We can view $x \in S^{n-1}$ as an element of $\mathbb{R}^n$, and $\xi \in T_x S^{n-1}$ as an element of $T_x \mathbb{R}^n \simeq \mathbb{R}^n$.

A suitable retraction on S^{n-1} is the projection/renormalization:

$$R_x(\xi) = \frac{x + \xi}{\|x + \xi\|}.$$

Consider the Rayleigh quotient on the sphere:

$$f : S^{n-1} \to \mathbb{R} : x \mapsto f(x) = x^T A x.$$

It is easy to show

$$\operatorname{grad} f(x) = (I - xx^T)2Ax = 2(Ax - xx^T Ax).$$

Gradient Descent for the Rayleigh Quotient on S^{n-1}

Algorithm 1 Armijo line search for the Rayleigh quotient on S^{n-1}

Require: Symmetric matrix A, Armijo backtracking parameters $\overline{\alpha} > 0, \beta, \sigma \in (0,1)$.

Input: Initial iterate x_0 s.t. $\|x_0\| = 1$.

Output: Sequence of iterates $\{x_k\}$.

1: **for** k=0,1,2,... **do**

2: Gradient: compute $\eta_k = -2(Ax_k - x_k x_k^T Ax_k)$.

3: Backtrack: find the smallest integer $m \geq 0$ such that

$$f(R_{x_k}(\overline{\alpha}\beta^m \eta_k)) \leq f(x_k) - \sigma\overline{\alpha}\beta^m \eta_k^T \eta_k.$$

4: Step: $x_{k+1} = R_{x_k}(\overline{\alpha}\beta^m \eta_k)$.

5: **end for**

If we take $t_k = 1/(2x_k^T Ax_k)$, we recover the *power method*.

Higher-order Methods

"Optimization Algorithms on Matrix Manifolds" also discusses higher-order methods:

- Newton's method
- Trust-region methods
- quasi-Newton methods
- Conjugate gradients

Additional elements of differential geometry are required:

- Affine connections (e.g. Levi-Civita)
- Riemannian Hessian
- Vector transport

Outline

1 A Motivating Example

2 Manifolds

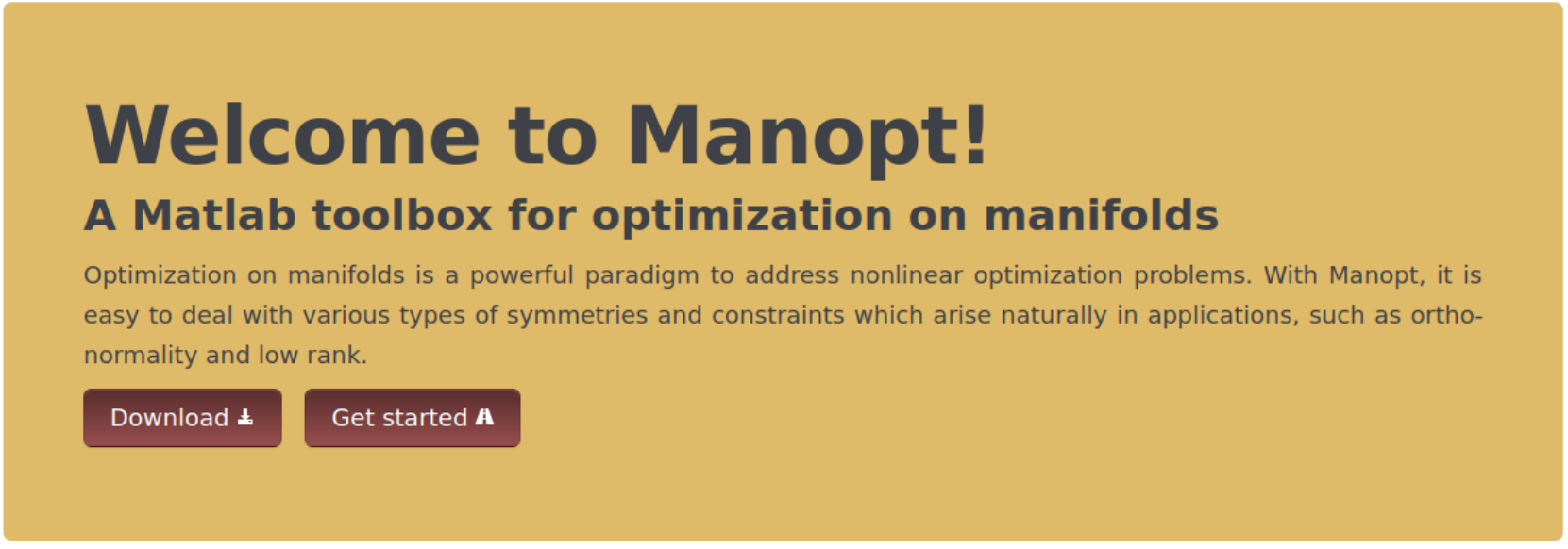
3 Gradient Descent

4 Manopt

5 Some Recent Work

Manopt

MATLAB toolbox developed by Nicolas Boumal and Bamdev Mishra, with many user contributions.



Manopt - Rayleigh Quotient on S^{n-1}

```
% Generate symmetric matrix
n = 1000;
A = randn(n); A = 0.5*(A+A.');

% Create problem structure
M = spherefactory(n);
problem.M = M;
```

Manopt - Rayleigh Quotient on S^{n-1}

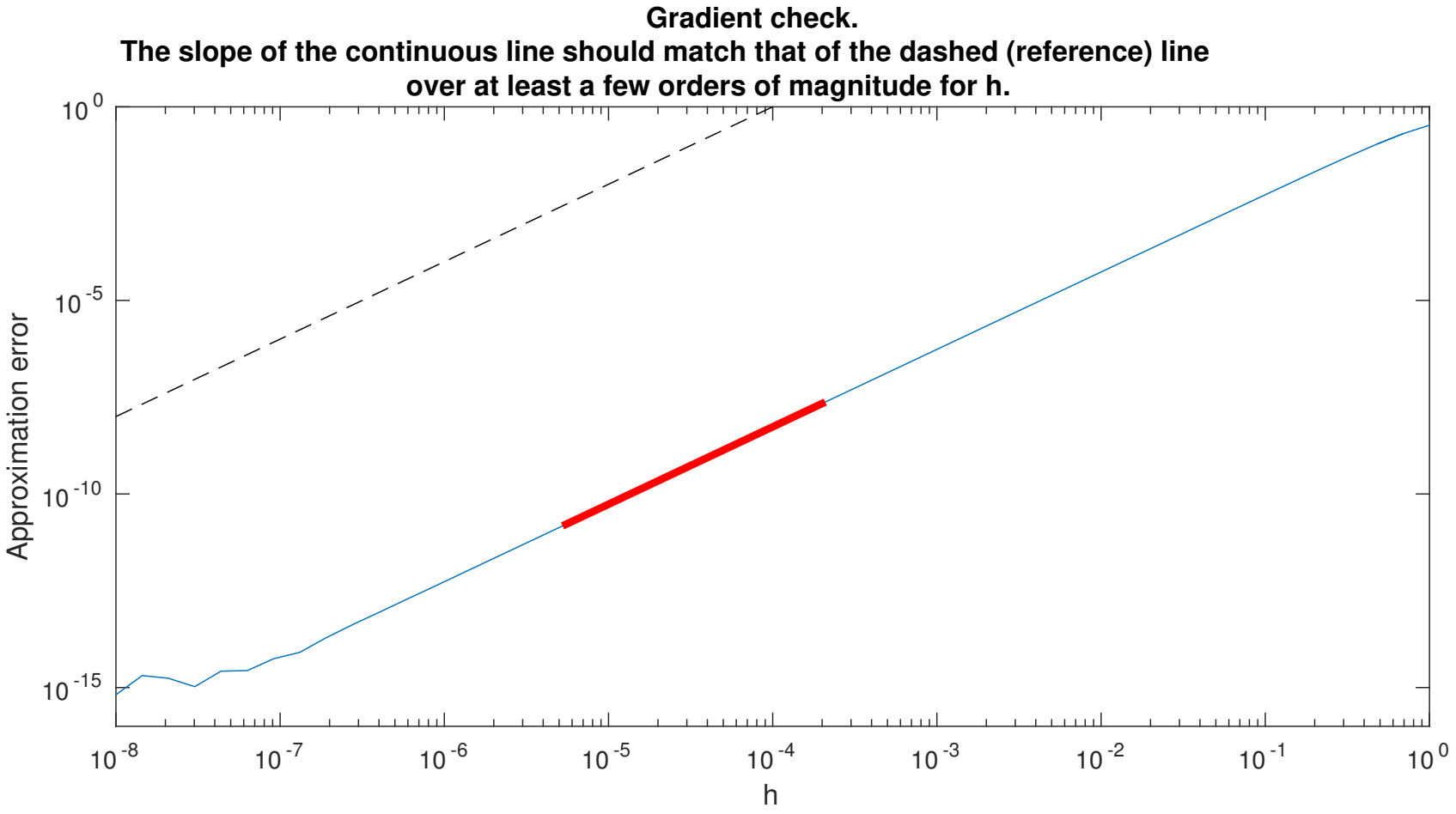
```
>> M

M =

                  name: @()sprintf('Sphere S^%d',n-1)
                   dim: @()n*m-1
                 inner: @(x,d1,d2)d1(:).'*d2(:)
                  norm: @(x,d)norm(d,'fro')
                  dist: @(x,y)real(acos(x(:).'*y(:)))
          typicaldist: @()pi
                  proj: @(x,d)d-x*(x(:).'*d(:))
              tangent: @(x,d)d-x*(x(:).'*d(:))
          egrad2rgrad: @(x,d)d-x*(x(:).'*d(:))
          ehess2rhess: @spherefactory/ehess2rhess
                  exp: @exponential
                 retr: @retraction
                  log: @spherefactory/logarithm
                 hash: @(x)['z',hashmd5(x(:))]
                 rand: @()random(n,m)
              randvec: @(x)randomvec(n,m,x)
              lincomb: @matrixlincomb
              zerovec: @(x)zeros(n,m)
               transp: @(x1,x2,d)M.proj(x2,d)
            pairmean: @spherefactory/pairmean
                  vec: @(x,u_mat)u_mat(:)
                  mat: @(x,u_vec)reshape(u_vec,[n,m])
    vecmatareisometries: @()true
```

Manopt - Rayleigh Quotient on S^{n-1}

```
% Generate symmetric matrix
n = 1000;
A = randn(n); A = 0.5*(A+A.');

% Create problem structure
M = spherefactory(n);
problem.M = M;

% Define cost function and Euclidean gradient
problem.cost = @(x) -x.'*(A*x);
problem.egrad = @(x) -2*A*x;

% Numerically check gradient consistency
checkgradient(problem);
```

Manopt - Rayleigh Quotient on S^{n-1}

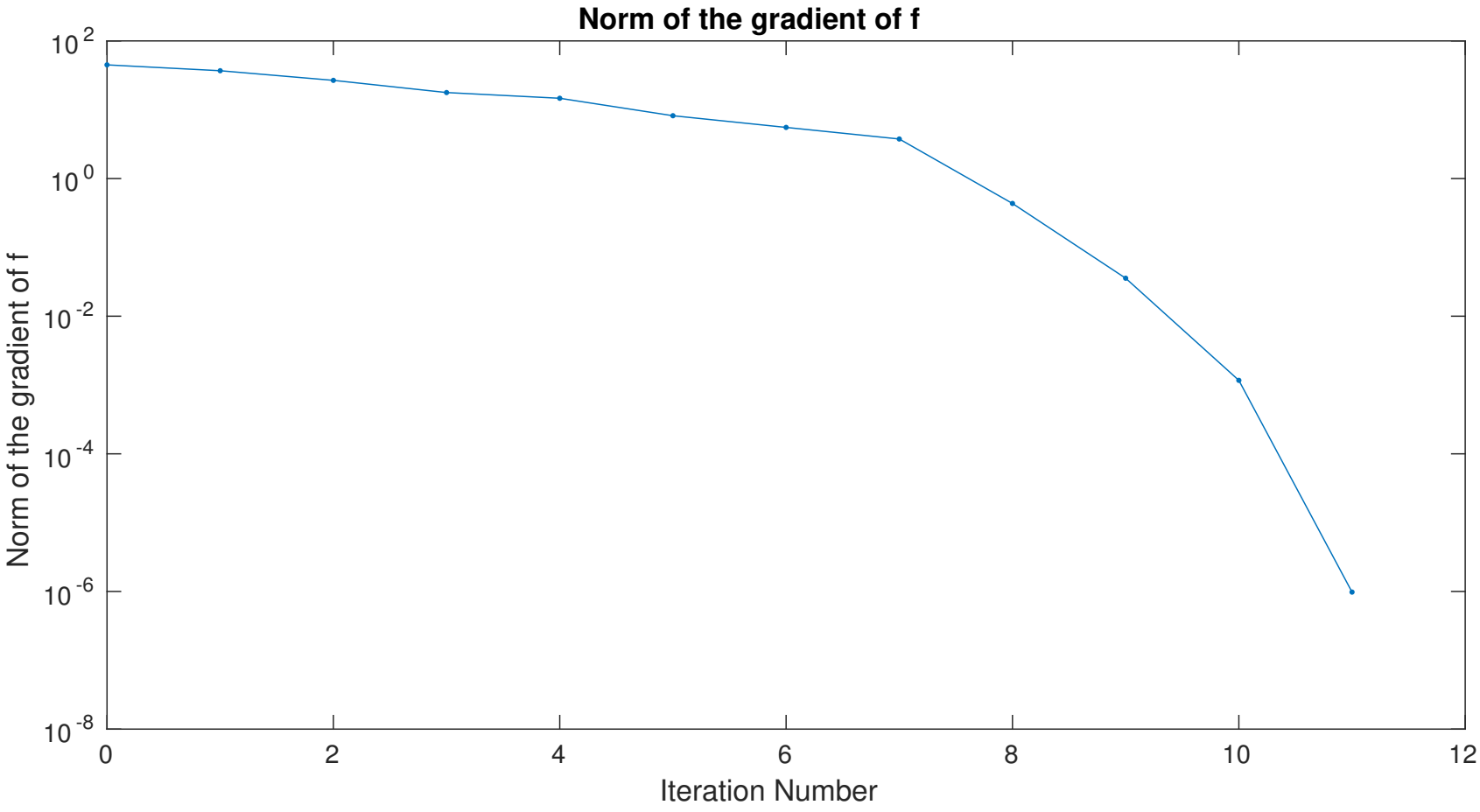
Manopt has tools to check gradients, Hessians, etc.



Manopt - Rayleigh Quotient on S^{n-1}

```
% Generate symmetric matrix
n = 1000;
A = randn(n); A = 0.5*(A+A.');

% Create problem structure
M = spherefactory(n);
problem.M = M;

% Define cost function and Euclidean gradient
problem.cost = @(x) -x.'*(A*x);
problem.egrad = @(x) -2*A*x;

% Numerically check gradient consistency
%checkgradient(problem);

% Solve
[x,xcost,info,opt] = trustregions(problem);
```

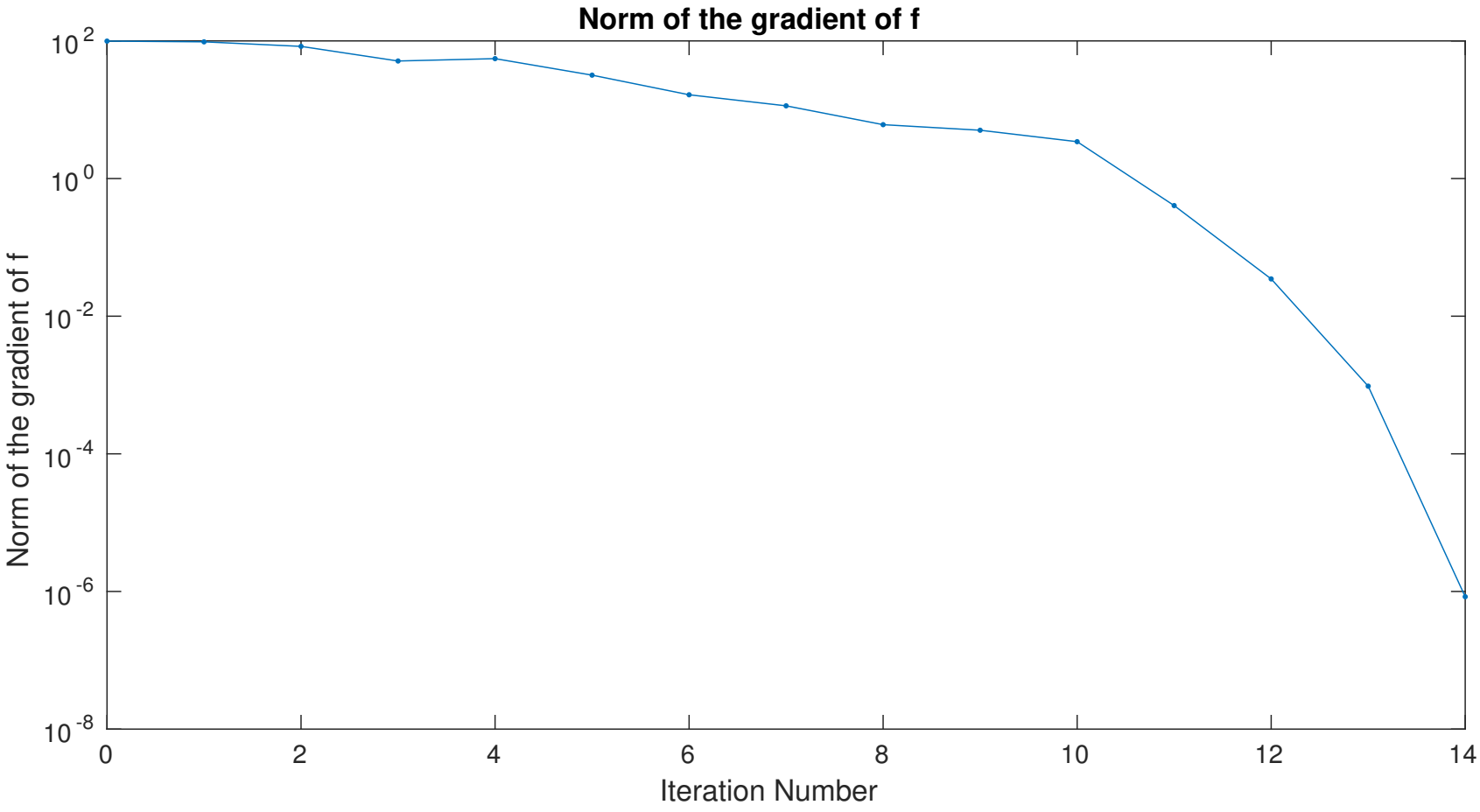
Manopt - Rayleigh Quotient on S^{n-1}



Manopt - Generalized Rayleigh Quotient on Grass(p, n)

```
% Generate symmetric matrix
n = 1000;
A = randn(n); A = 0.5*(A+A.');

% Create problem structure
p = 5;
M = grassmannfactory(n,p);
problem.M = M;

% Define cost function and Riemannian gradient
% In grassmannfactory, Y is an ON matrix
problem.cost = @(Y) -trace(Y'*A*Y);
problem.grad = @(Y) -2*(A*Y - Y*(Y'*A*Y));

% Numerically check gradient consistency
%checkgradient(problem);

% Solve
[Y,Ycost,info,opt] = trustregions(problem);
```

Manopt - Generalized Rayleigh Quotient on Grass(p, n)



Outline

1 A Motivating Example

2 Manifolds

3 Gradient Descent

4 Manopt

5 Some Recent Work

Geodesic Convexity for GMMs

Definition (g-convex sets)

A set $\mathcal{X} \subset \mathcal{M}$ is *geodesically convex* if any two points of $\mathcal{X}$ are joined by a geodesic in $\mathcal{X}$.

Definition (g-convex functions)

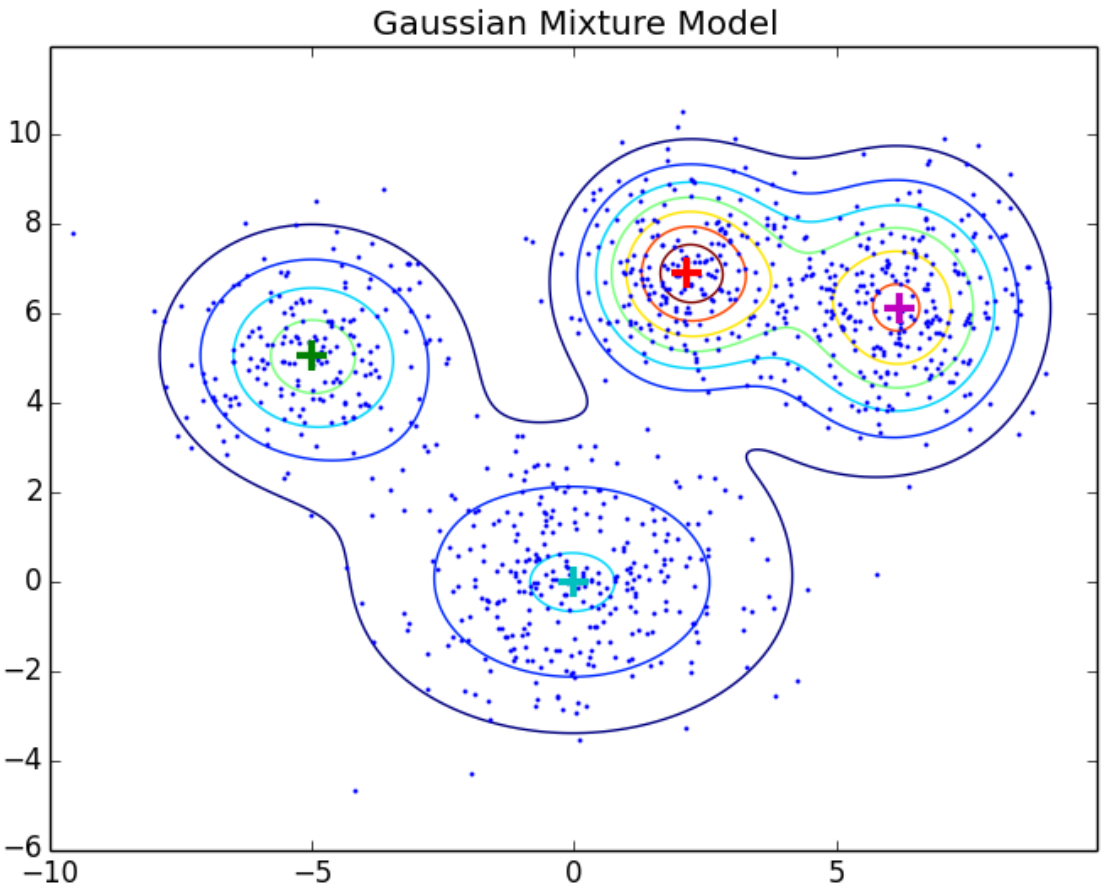
Let $\mathcal{X} \subset \mathcal{M}$ be g-convex. A function $f : \mathcal{X} \to \mathbb{R}$ is *geodesically convex* if $\forall x, y \in \mathcal{M}$,

$$f(\gamma(t)) \leq (1-t)f(x) + tf(y),$$

where $\gamma : [0,1] \to \mathcal{X}$ is a geodesic connecting x and y.

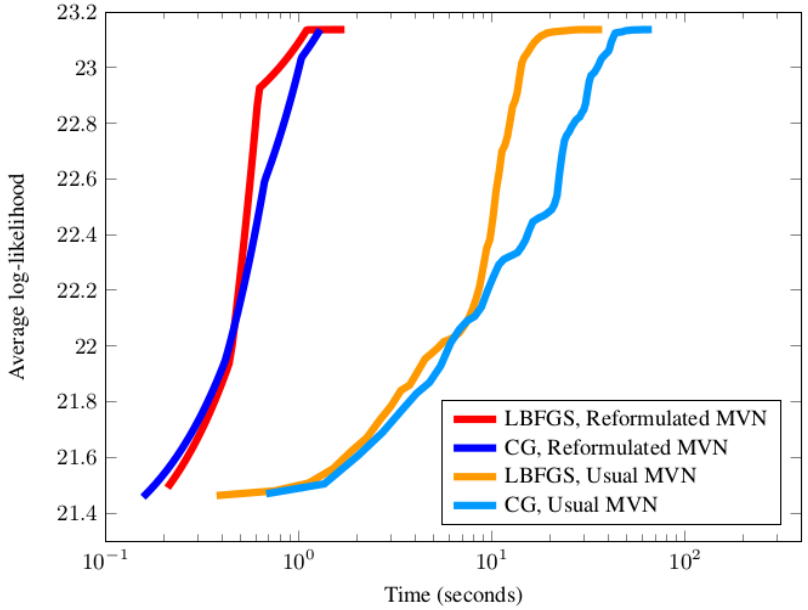
Geodesic Convexity for GMMs

Recent work by Suvrit Sra and Reshad Hosseini.
GMM loss is Euclidean non-convex and also not g-convex.

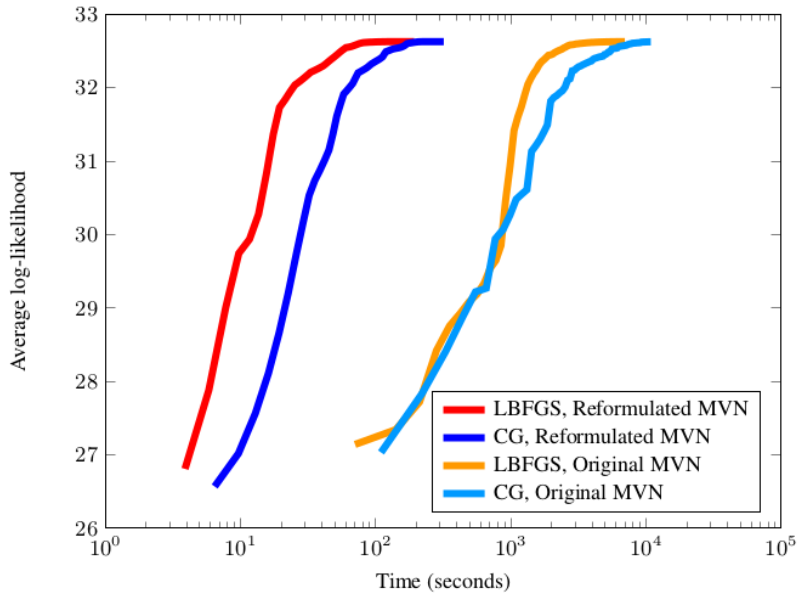


Geodesic Convexity for GMMs

Can reparameterize GMM loss over manifold of positive-definite matrices so fitting a *single* Gaussian *is* g-convex.
GMM loss is still not g-convex.
However, there *is* a substantial benefit (competitive with EM).

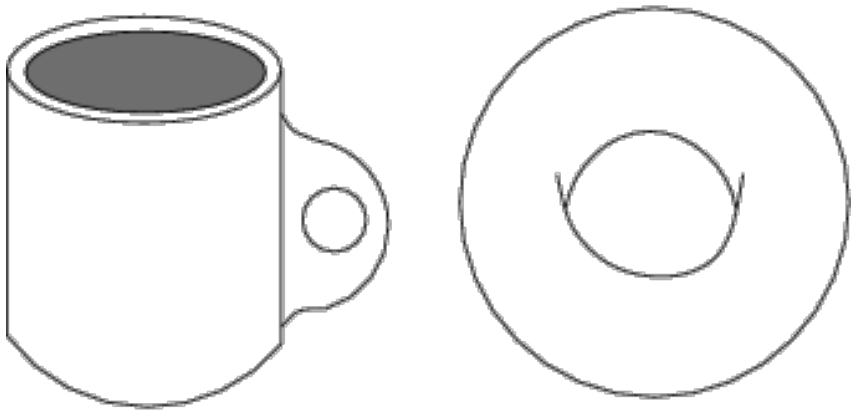


(a) Single Gaussian



(b) Mixtures of seven Gaussians

Thanks!



References:

- P.-A. Absil, R. Mahoney, R. Sepulchre, *Optimization Algorithms on Matrix Manifolds*, Princeton University Press (2008).
- N. Boumal, B. Mishra, P.-A. Absil, R. Sepulchre, *Manopt: a Matlab Toolbox for Optimization on Manifolds*, JMLR (2014).
- R. Hosseini, S. Sra, *Manifold Optimization for Gaussian Mixture Models*, Arxiv:1506.07677 (2015).

Pictures

I took various pictures from

- References above
- "Optimization on Manifolds: Methods and Applications", P.-A. Absil at British-French-German Conference on Optimization, 2009.
- Wikipedia
- http://yulearning.blogspot.com/2014/11/einsteins-most-famous-equation-is-emc2.html