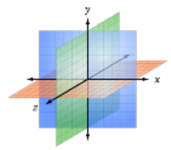
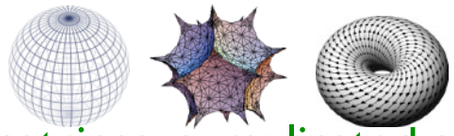
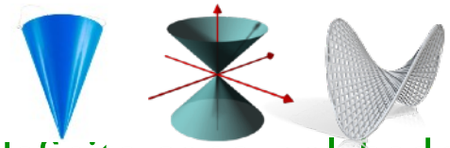
Optimization on Manifolds for ML

Reshad Hosseini
Department of Machine Intelligence and Robotics
University of Tehran

Invited Talk at Amirkabir AI Summer Symposium

Joint Work with Suvrit Sra

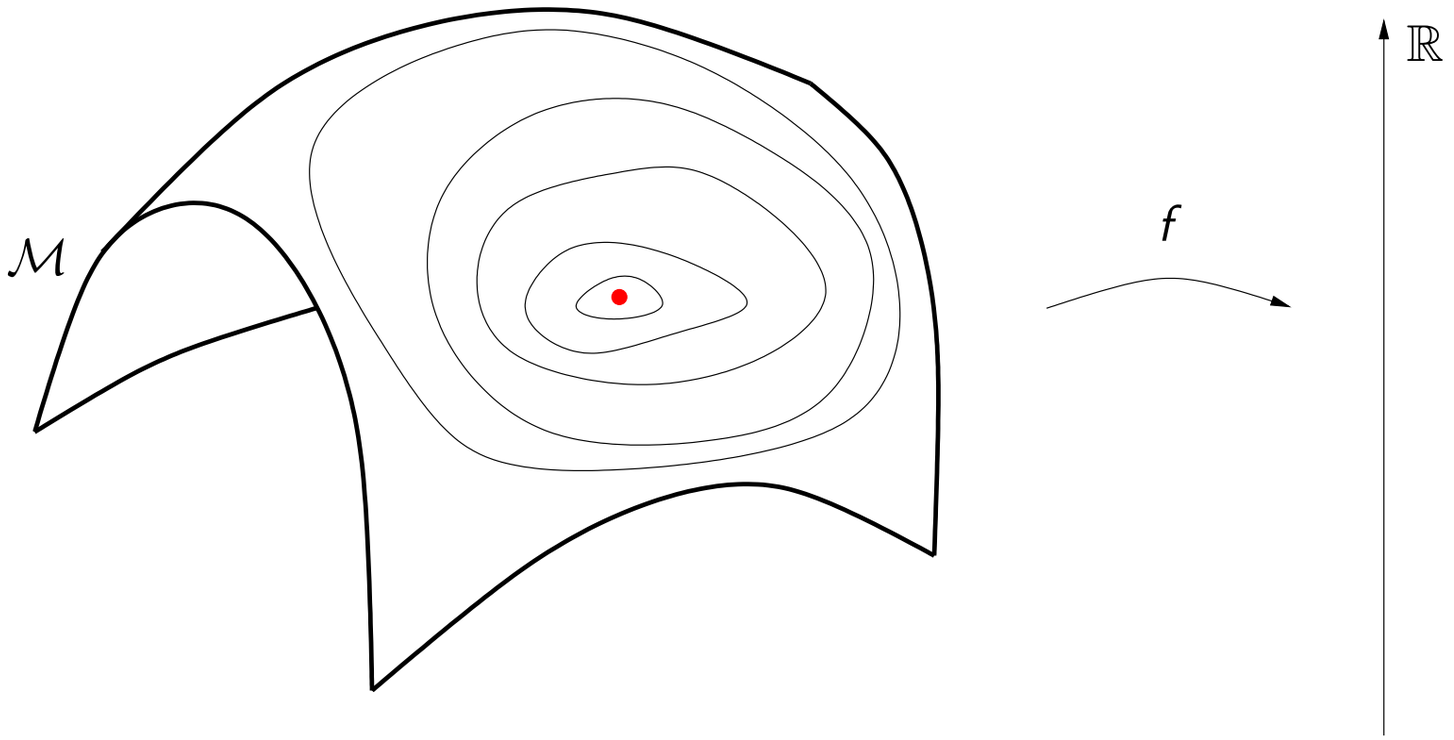
Optimization on manifolds for Machine Learning

- **Vector spaces** 
- **Manifolds** 
(hypersphere, orthogonal matrices, complicated surfaces)
- **Convex sets** 
(probability simplex, semidefinite cone, polyhedra)
- **Metric spaces** 
(tree space, Wasserstein spaces, CAT(0), space-of-spaces)

Machine Learning
Graphics
Robotics
Vision
BCI
NLP
Statistics

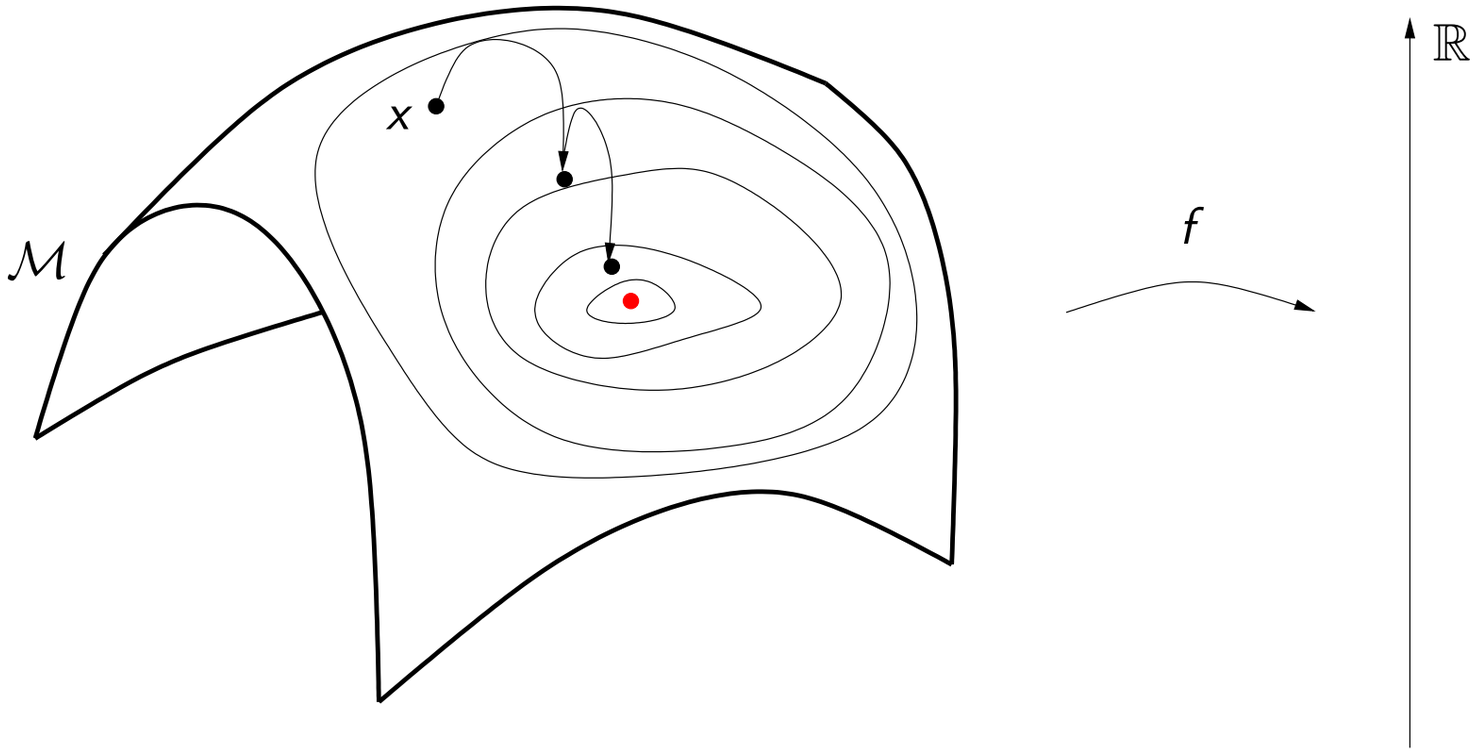
Optimization on Manifolds

Optimization on manifolds in one picture



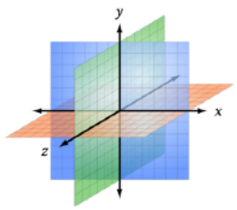
Slide Credit: Pierre-Antoine Absil

Optimization on manifolds in one picture



Slide Credit: Pierre-Antoine Absil

Example: Riemannian optimization



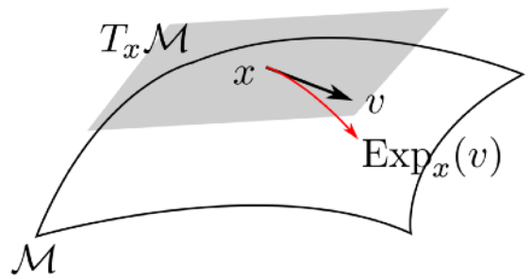
Vector space optimization

- Orthogonality constraint
- Fixed-rank constraint
- Positive semi-definite constraint
- … …

⇔

Riemannian optimization

- Stiefel manifold
- Grassmann manifold
- PSD manifold
- … …



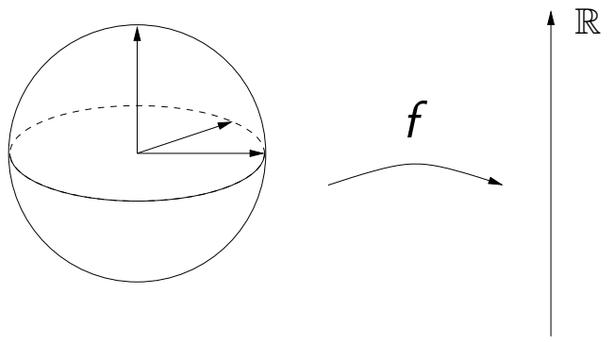
[Udriste, 1994; Absil et al., 2009]

Example: Riemannian optimization

Given $A = A^T \in \mathbb{R}^{n \times n}$ and $N = \mathrm{diag}(1, \ldots, p)$,

$$\min f(X) = \mathrm{trace}(X^T A X N)$$

subj. to $X \in \mathbb{R}^{n \times p} : X^T X = I$



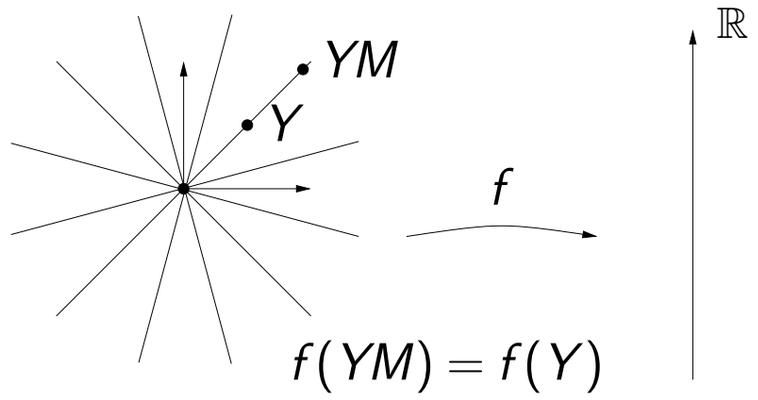
Domain: $\mathrm{St}(p, n)$
$= \{X \in \mathbb{R}^{n \times p} : X^T X = I\}$

Embedded submanifold

Given $A = A^T \in \mathbb{R}^{n \times n}$,

$$\min f(Y) = \mathrm{trace}\left((Y^T Y)^{-1}(Y^T A Y)\right)$$

subj. to $Y \in \mathbb{R}_*^{n \times p}$ (i.e., Y full rank)

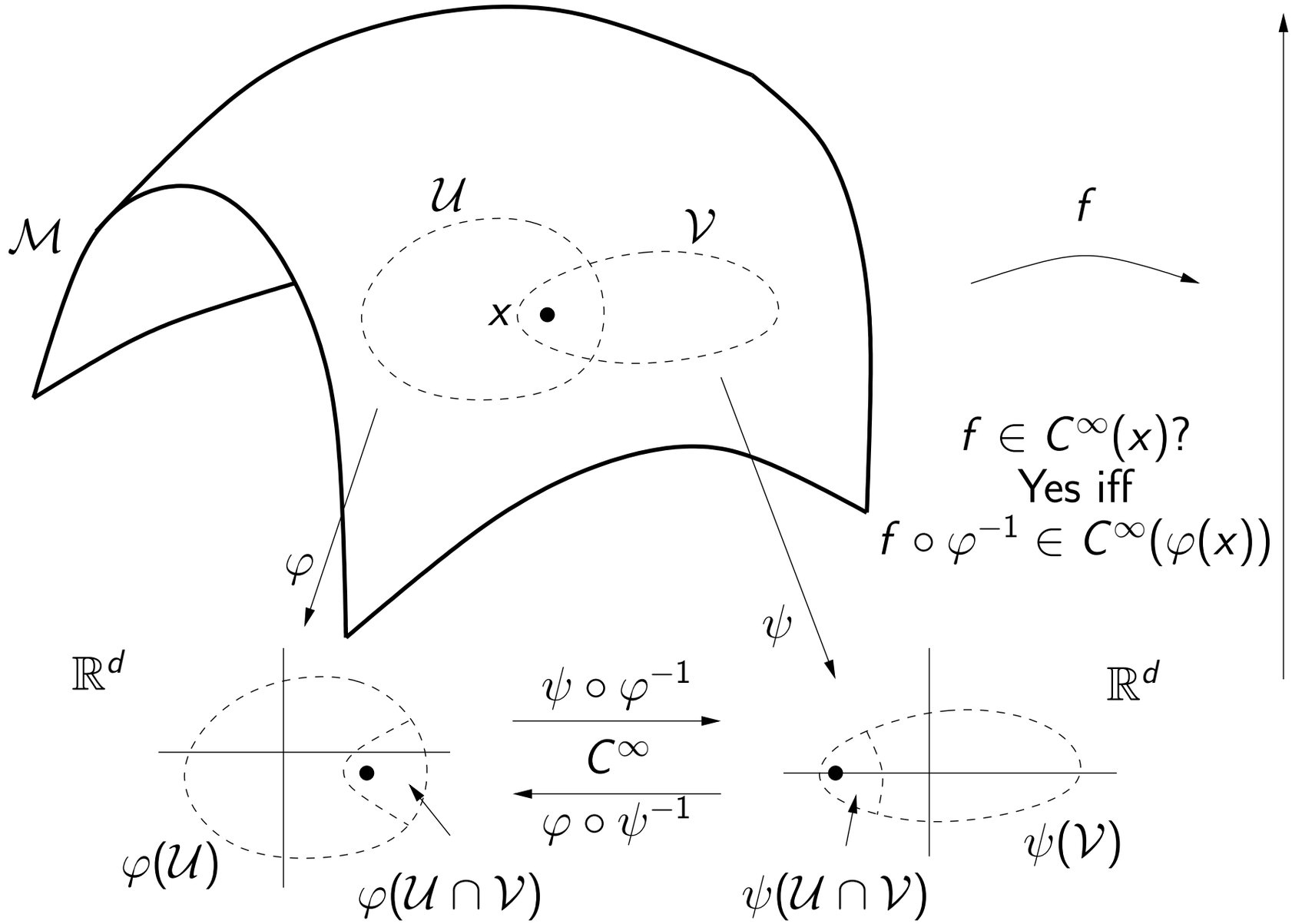


Domain: $\mathrm{Gr}(p, n)$
$= \left\{\{YM : M \in \mathbb{R}_*^{p \times p}\} : Y \in \mathrm{R}_*^{n \times p}\right\}$

Quotient manifold

Slide Credit: Pierre-Antoine Absil

Smooth manifolds and smooth functions



Slide Credit: Pierre-Antoine Absil

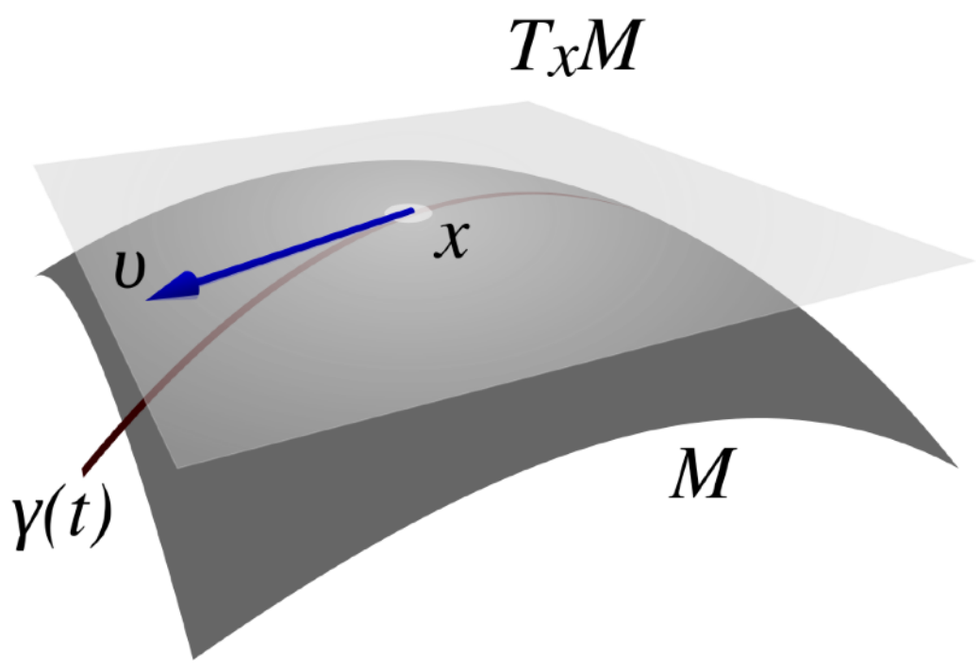
Algorithm on Riemannian manifolds

- **Steepest Descent –** Needs Retraction
- **Newton –** Needs Affine Connection and Retraction
- **Conjugate Gradient –** Needs Retraction and Vector Transport
- **BFGS –** Needs Retraction and Vector Transport
- **Trust Region –** Needs Retraction

Tangent space

- Doing Calculus needs **differential structure**
- This can be obtained by defining a tangent bundle,

  that is, the set of tangent spaces for all points



Riemannian metric

- Define an inner product called **Riemannian metric**

$$g_x(\xi, \eta) : T_xM \times T_xM \to \mathbb{R}$$

Riemannian metric

- Define an inner product called **Riemannian metric**

$$g_x(\xi, \eta) : T_x M \times T_x M \to \mathbb{R}$$

- This metric induces a norm

Riemannian metric

- Define an inner product called **Riemannian metric**

$$g_x(\xi, \eta) : T_x M \times T_x M \to \mathbb{R}$$

- This metric induces a norm

$$||\xi||_x = \sqrt{g_x(\xi, \xi)}$$

Riemannian metric

- Define an inner product called **Riemannian metric**

$$g_x(\xi, \eta) : T_xM \times T_xM \to \mathbb{R}$$

- This metric induces a norm

$$||\xi||_x = \sqrt{g_x(\xi, \xi)}$$

- This can be used to measure the **length** of a curve

Riemannian metric

- Define an inner product called **Riemannian metric**

$$g_x(\xi, \eta) : T_xM \times T_xM \to \mathbb{R}$$

- This metric induces a norm

$$||\xi||_x = \sqrt{g_x(\xi, \xi)}$$

- This can be used to measure the **length** of a curve

$$L(x(t)) = \int_a^b \sqrt{||\dot{x}(t)||}$$

Riemannian metric

- Define an inner product called **Riemannian metric**

$$g_x(\xi, \eta) : T_xM \times T_xM \to \mathbb{R}$$

- This metric induces a norm

$$||\xi||_x = \sqrt{g_x(\xi, \xi)}$$

- This can be used to measure the **length** of a curve

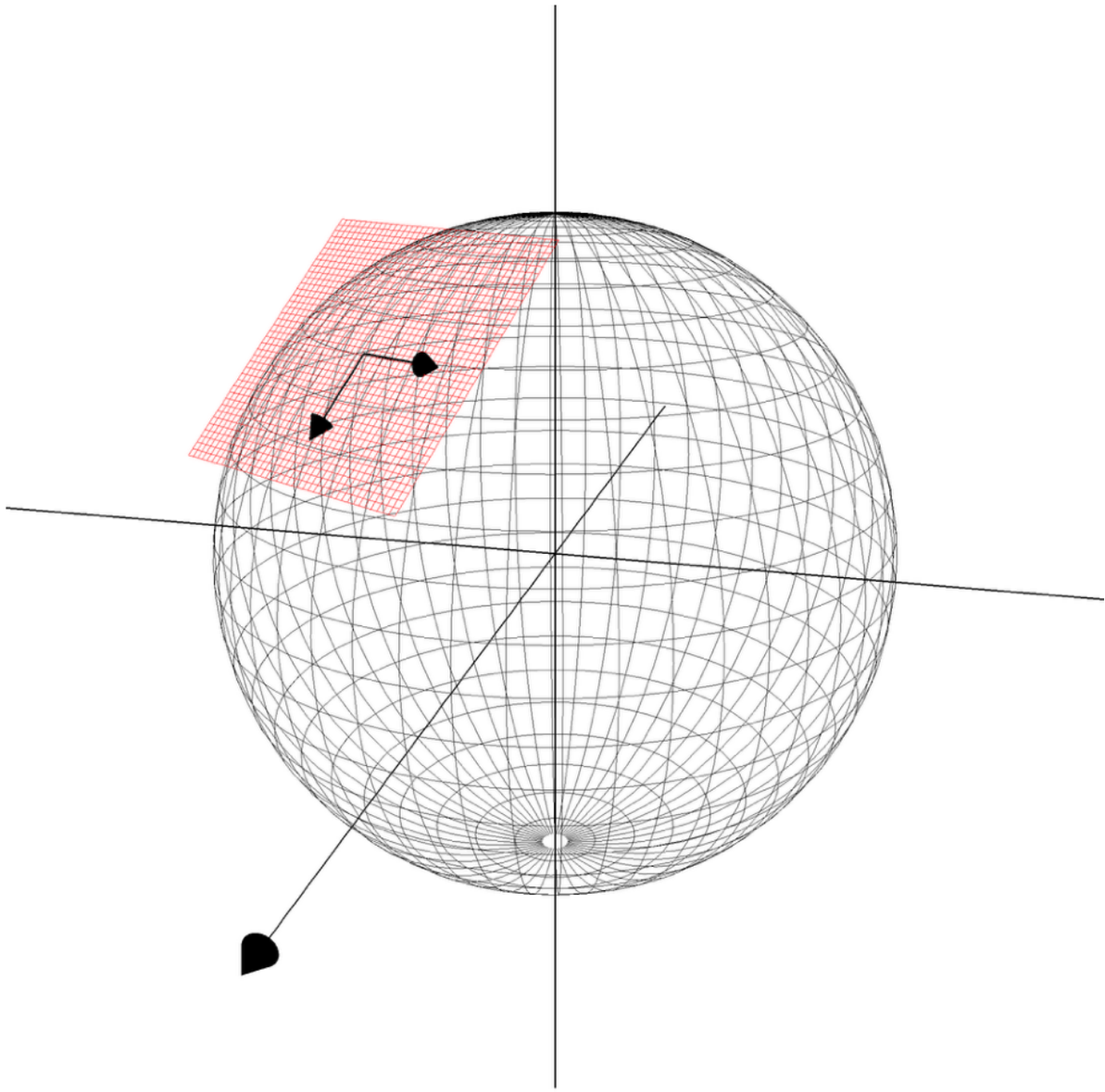
$$L(x(t)) = \int_a^b \sqrt{||\dot{x}(t)||}$$

- **Geodesics** are locally minimizing curves between two points

Example: n-sphere

- The tangent space is given by

$$T_x M = \{v \in \mathbb{R}^n | v^T x = 0\}$$

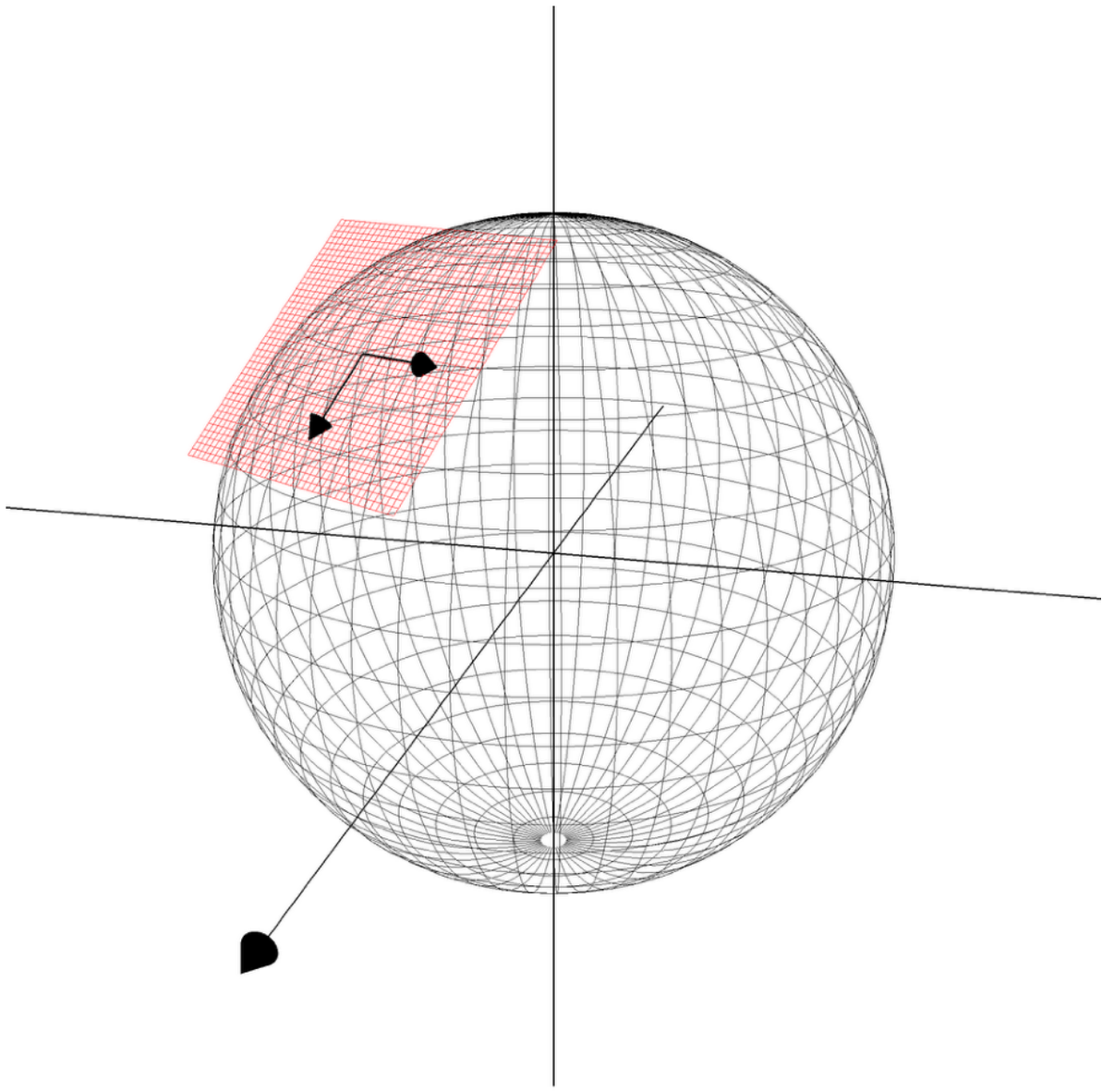


Example: n-sphere

- The tangent space is given by

$$T_x M = \{v \in \mathbb{R}^n | v^T x = 0\}$$

- The inner product is inherited by the embedding Euclidean space and is **standard inner product**



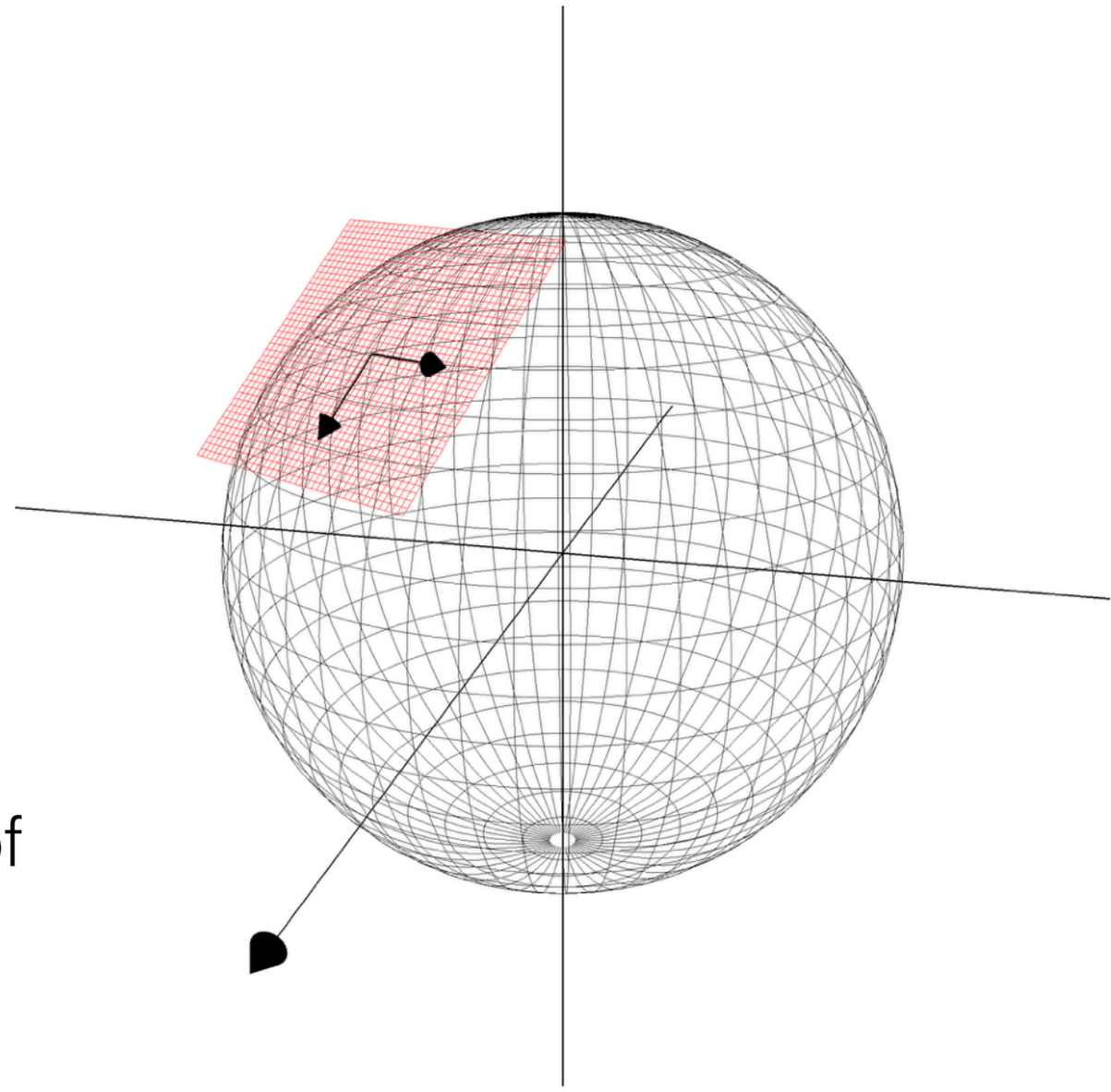
Example: n-sphere

- The tangent space is given by

$$T_x M = \{v \in \mathbb{R}^n | v^T x = 0\}$$

- The inner product is inherited by the embedding Euclidean space and is **standard inner product**

- Geodesics are the **great circles** of the sphere



Riemannian gradient

- **Riemannian gradient** is the direction of the steepest descent

Riemannian gradient

- **Riemannian gradient** is the direction of the steepest descent

$$g_x(\nabla f, \xi) = D_\xi f(x)$$

Riemannian gradient

- **Riemannian gradient** is the direction of the steepest descent

$$g_x(\nabla f, \xi) = D_\xi f(x)$$

- Where

Riemannian gradient

- **Riemannian gradient** is the direction of the steepest descent

$$g_x(\nabla f, \xi) = D_\xi f(x)$$

- Where

$$D_\xi f(x) = \frac{d}{dt} f(\delta(t)), \quad \dot{\delta}(t) = \xi$$

Riemannian gradient descent

- Consider Euclidean gradient descent

$$x_{t+1} = x_t - \lambda \nabla f(x_t)$$

Riemannian gradient descent

- Consider Euclidean gradient descent

$$x_{t+1} = x_t - \lambda \nabla f(x_t)$$

- For arbitrary manifold this does not make sense

Riemannian gradient descent

- Consider Euclidean gradient descent

$$x_{t+1} = x_t - \lambda \nabla f(x_t)$$

- For arbitrary manifold this does not make sense

- Moving along straight line is replaced by moving along geodesics that is denoted by **Exponential map**

Riemannian gradient descent

- Consider Euclidean gradient descent

$$x_{t+1} = x_t - \lambda \nabla f(x_t)$$

- For arbitrary manifold this does not make sense

- Moving along straight line is replaced by moving along geodesics that is denoted by **Exponential map**

$$x_{t+1} = \mathrm{Exp}_{x_t}(-\lambda \nabla f(x_t))$$

Riemannian gradient descent

- Consider Euclidean gradient descent

$$x_{t+1} = x_t - \lambda \nabla f(x_t)$$

- For arbitrary manifold this does not make sense

- Moving along straight line is replaced by moving along geodesics that is denoted by **Exponential map**

$$x_{t+1} = \mathrm{Exp}_{x_t}(-\lambda \nabla f(x_t))$$

- Exponential map is moving along geodesic γ(t)

Riemannian gradient descent

- Consider Euclidean gradient descent

$$x_{t+1} = x_t - \lambda \nabla f(x_t)$$

- For arbitrary manifold this does not make sense

- Moving along straight line is replaced by moving along geodesics that is denoted by **Exponential map**

$$x_{t+1} = \mathrm{Exp}_{x_t}(-\lambda \nabla f(x_t))$$

- Exponential map is moving along geodesic γ(t)

$$\gamma(0) = x_t, \quad \dot{\gamma}(0) = \nabla f(x_t)$$

Retraction map

- **Exponential maps** can be **hard to compute**. We can replace that with **retraction**

Retraction map

- **Exponential maps** can be **hard to compute**. We can replace that with **retraction**

- Retraction is a mapping from the tangent vector to a point in manifold.

Retraction map

- **Exponential maps** can be **hard to compute**. We can replace that with **retraction**

- Retraction is a mapping from the tangent vector to a point in manifold.

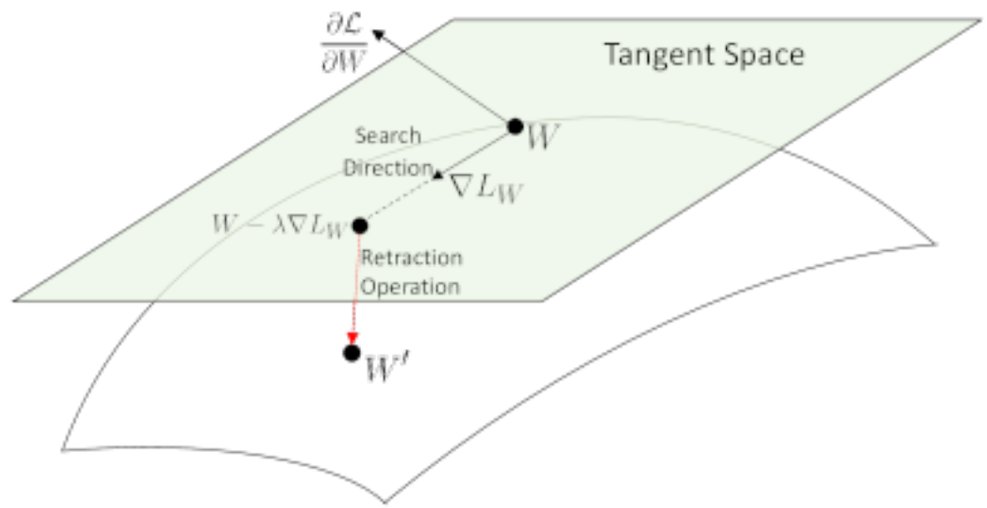
  - It is like moving along a curve with velocity equal to input tangent vector

Retraction map

- **Exponential maps** can be **hard to compute**. We can replace that with **retraction**

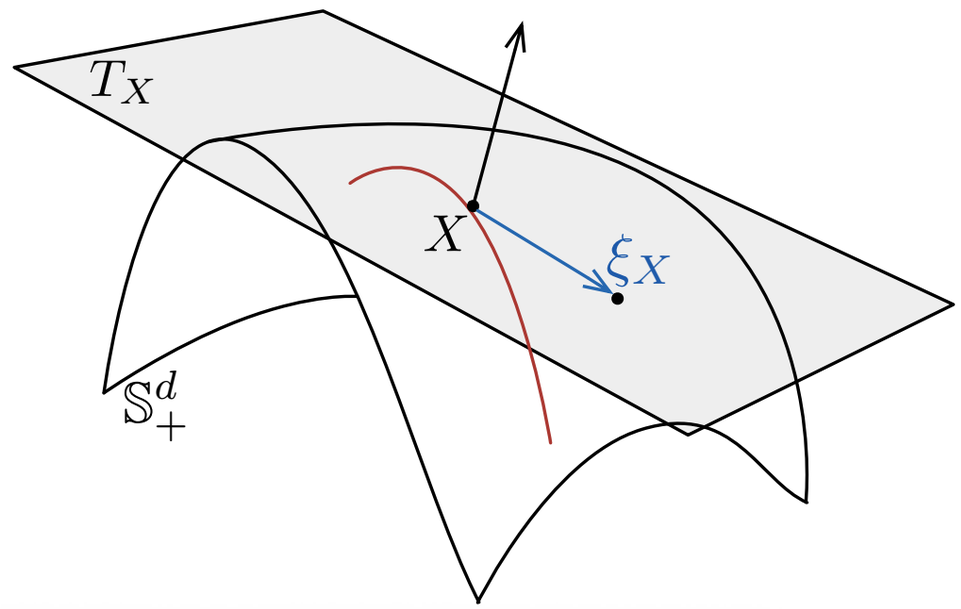
- Retraction is a mapping from the tangent vector to a point in manifold.

- It is like moving along a curve with velocity equal to input tangent vector



Riemannian L-BFGS

(own version)



Retraction

initial stepsize

Strong-Wolfe line-search

$$f(R_{X_k}(\alpha\xi_k)) \leq f(X_k) + c_1\alpha Df(X_k)\xi_k,$$
$$|Df(X_{k+1})\xi_{k+1}| \leq c_2 Df(X_k)\xi_k$$

$$\alpha^* = 2\frac{f(X_k) - f(X_{k-1})}{Df(X_{k-1})\xi_{k-1}}.$$

github.com/utvisionlab/mixest

Riemannian Newton Method

- For finding Newton direction solve

$$Hf(x)\xi = -\nabla f(x)$$

Riemannian Newton Method

- For finding Newton direction solve

$$Hf(x)\xi = -\nabla f(x)$$

- Where

Riemannian Newton Method

- For finding Newton direction solve

$$Hf(x)\xi = -\nabla f(x)$$

- Where

$$Hf(x)\xi = \partial_\xi \nabla f(x), \partial \text{ is affine connection}$$

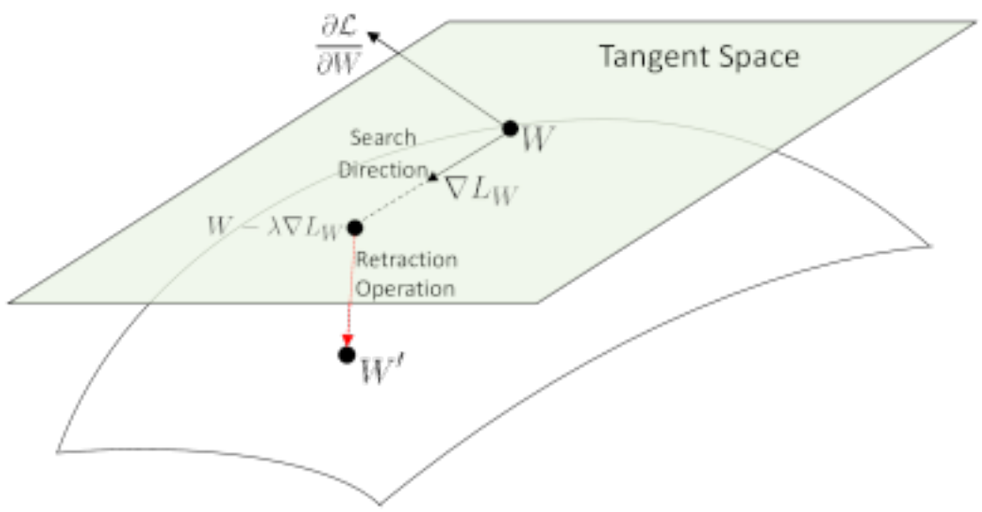
Riemannian Newton Method

- For finding Newton direction solve

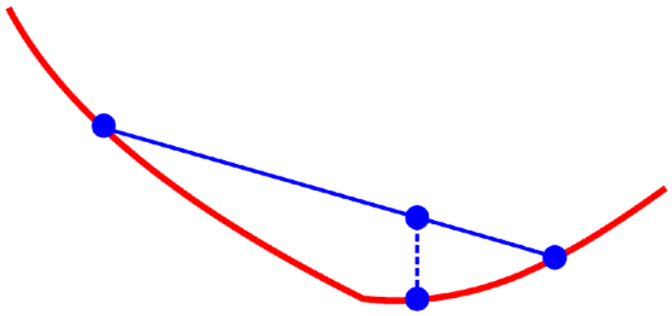
$$Hf(x)\xi = -\nabla f(x)$$

- Where

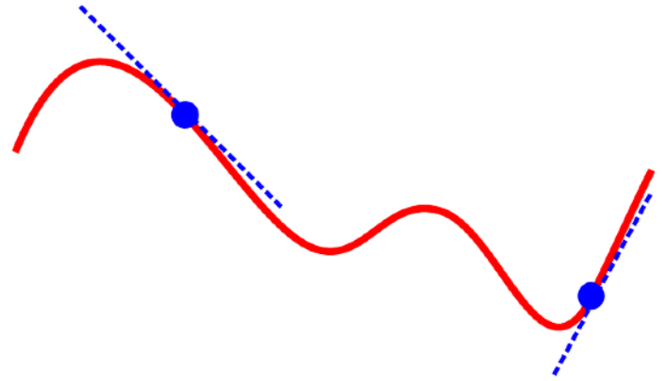
$$Hf(x)\xi = \partial_\xi \nabla f(x), \partial \text{ is affine connection}$$



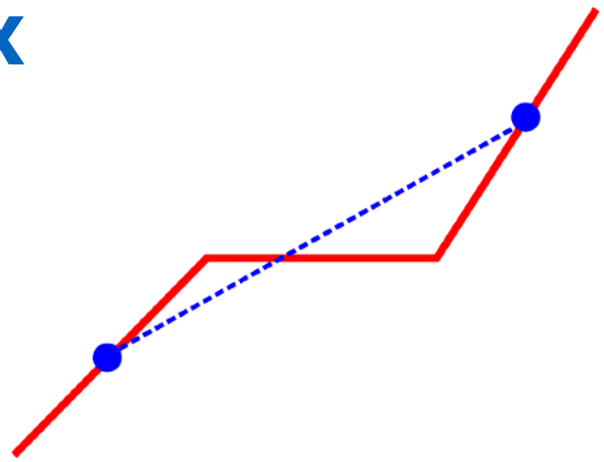
Classes of function in optimization



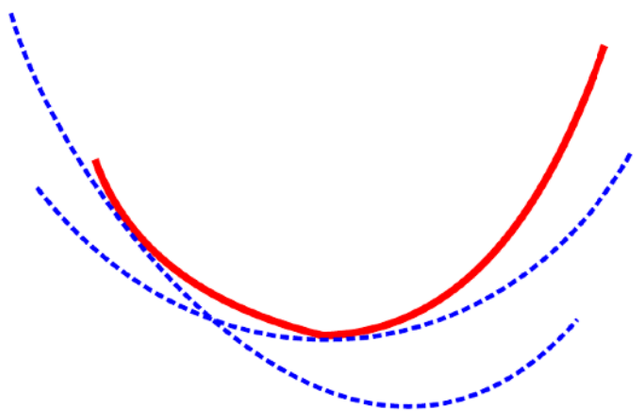
Convex



Smooth

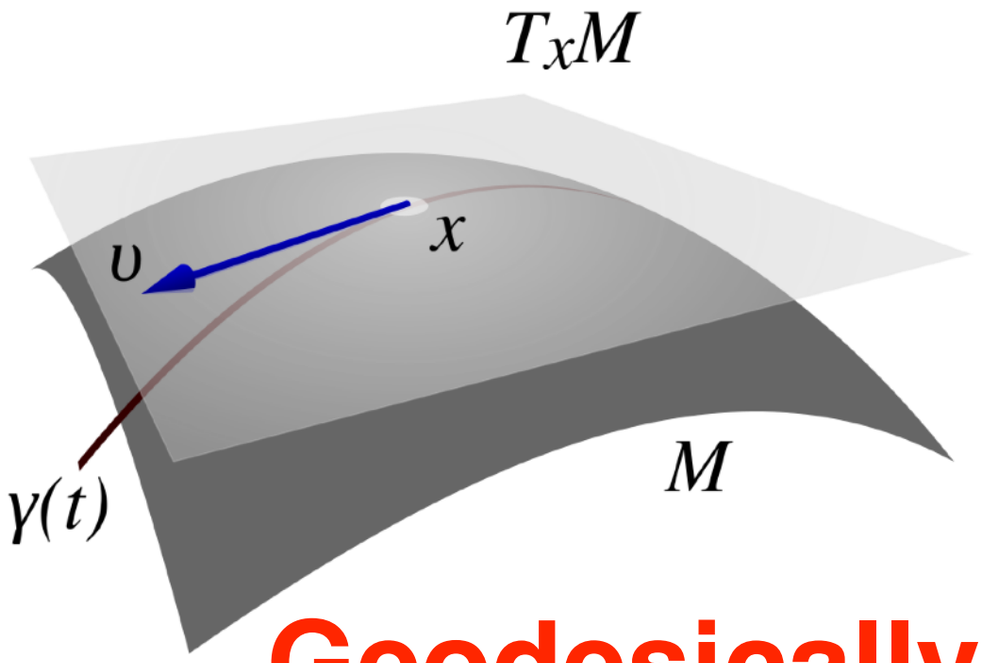


Lipschitz

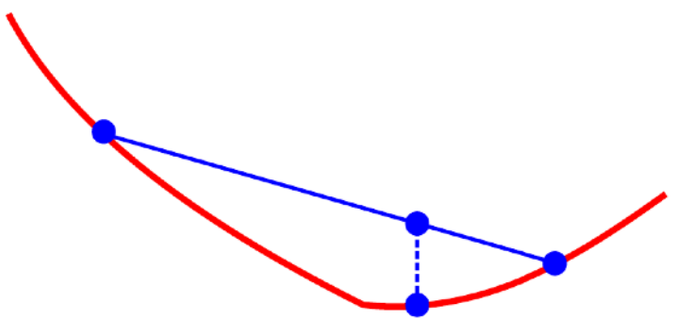


Strongly convex

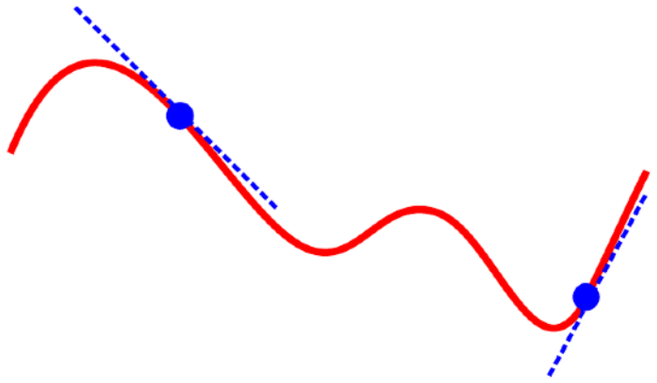
Classes of function in optimization



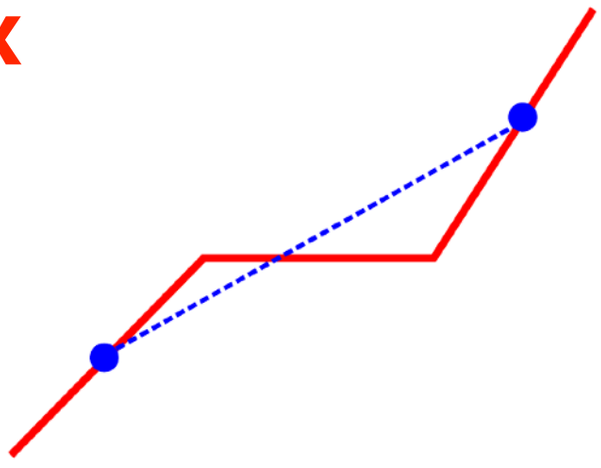
Geodesically



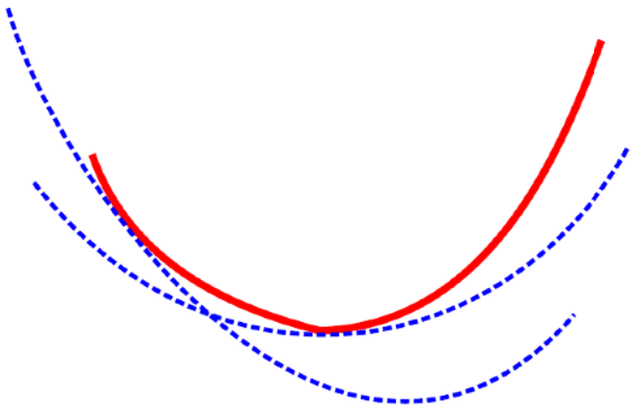
Convex



Smooth



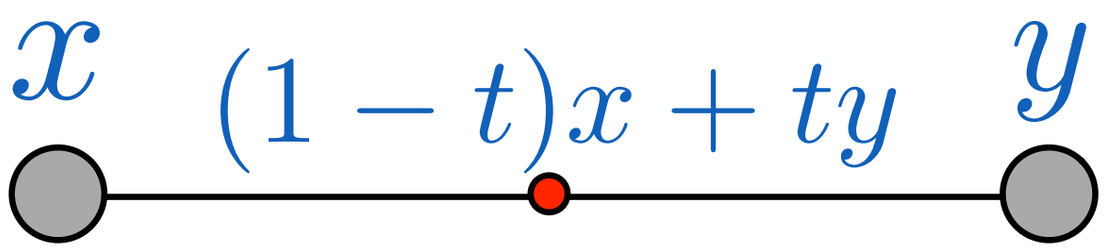
Lipschitz



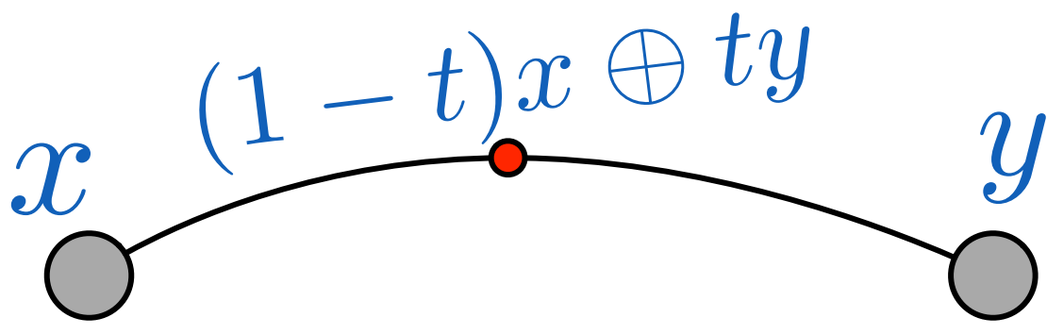
Strongly convex

What is geodesic convexity?

Convexity



Geodesic convexity



$$f((1-t)x \oplus ty) \leq (1-t)f(x) + tf(y)$$

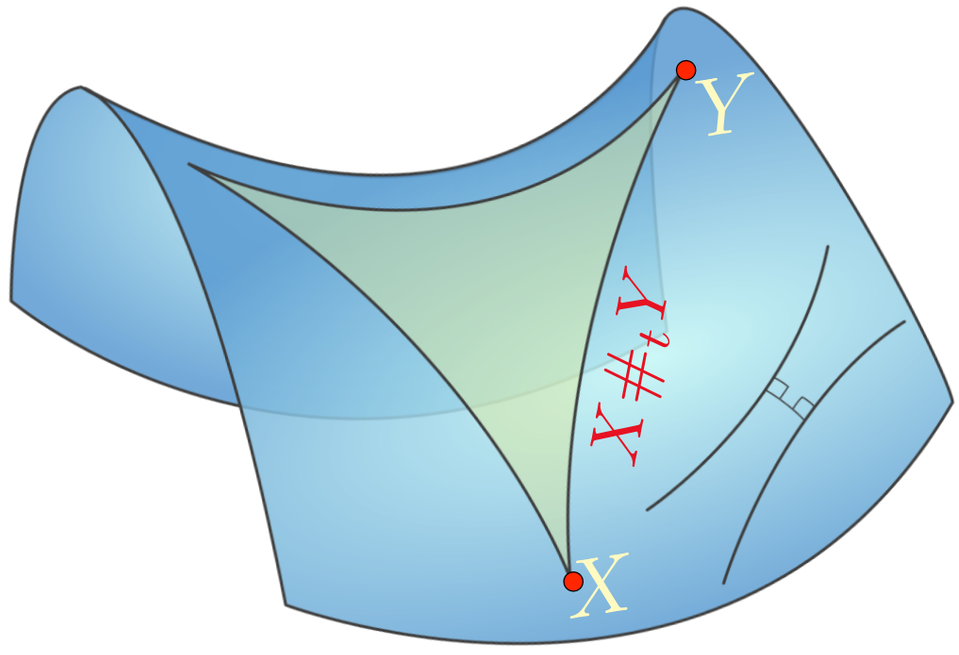
on a Riemannian manifold $f(y) \geq f(x) + \left\langle g_x, \mathrm{Exp}_x^{-1}(y) \right\rangle_x$

Metric spaces & curvature: [Menger; Alexandrov; Busemann; Bridson, Häflinger; Gromov; Perelman]

Example: Positive definite matrices

Geodesic

$$X\#_t Y := X^{\frac{1}{2}}(X^{-\frac{1}{2}}YX^{-\frac{1}{2}})^t X^{\frac{1}{2}}$$
$$= (1-t)X \oplus tY$$



Examples

$$f(X) = \begin{cases} \log\det(X), & \log\mathrm{tr}(X), \\ \mathrm{tr}(X^\alpha), & \|X^\alpha\|. \quad (\alpha > 0) \end{cases}$$

Exercise: verify for above examples

$$f(X\#_t Y) \leq (1-t)f(X) + tf(Y)$$

PSD manifold and g-convexity

Recognizing, constructing, and optimizing g-convex functions

[Sra, Hosseini (2013,2015)]

- *[Wiesel 2012]*
- *[Rápcsák 1984]*
- *[Udriste 1994]*

Corollaries

$$X \mapsto \log\det(B + \sum\nolimits_i A_i^* X A_i)$$

$$X \mapsto \log\mathrm{per}(B + \sum\nolimits_i A_i^* X A_i)$$

$$\delta_R^2(X,Y), \quad \delta_S^2(X,Y)$$

(jointly g-convex)

Many more theorems and corollaries

One-D version known as: **Geometric Programming**
www.stanford.edu/~boyd/papers/gp_tutorial.html
[Boyd, Kim, Vandenberghe, Hassibi (2007). 61pp.]

PSD manifold and g-convexity

Recognizing, constructing, and optimizing g-convex functions

[Sra, Hosseini (2013,2015)]

- *[Wiesel 2012]*
- *[Rápcsák 1984]*
- *[Udriste 1994]*

Corollaries

$$X \mapsto \log\det(B + \sum\nolimits_i A_i^* X A_i)$$

$$X \mapsto \log\mathrm{per}(B + \sum\nolimits_i A_i^* X A_i)$$

$$\delta_R^2(X,Y), \quad \delta_S^2(X,Y)$$

(jointly g-convex)

Many more theorems and corollaries

One-D version known as: **Geometric Programming**
www.stanford.edu/~boyd/papers/gp_tutorial.html
[Boyd, Kim, Vandenberghe, Hassibi (2007). 61pp.]

PSD manifold and g-convexity

Recognizing, constructing, and optimizing g-convex functions

[Sra, Hosseini (2013,2015)]

- *[Wiesel 2012]*
- *[Rápcsák 1984]*
- *[Udriste 1994]*

Corollaries

$$X \mapsto \log\det(B + \sum\nolimits_i A_i^* X A_i)$$

$$X \mapsto \log\mathrm{per}(B + \sum\nolimits_i A_i^* X A_i)$$

$$\delta_R^2(X,Y), \quad \delta_S^2(X,Y)$$

(jointly g-convex)

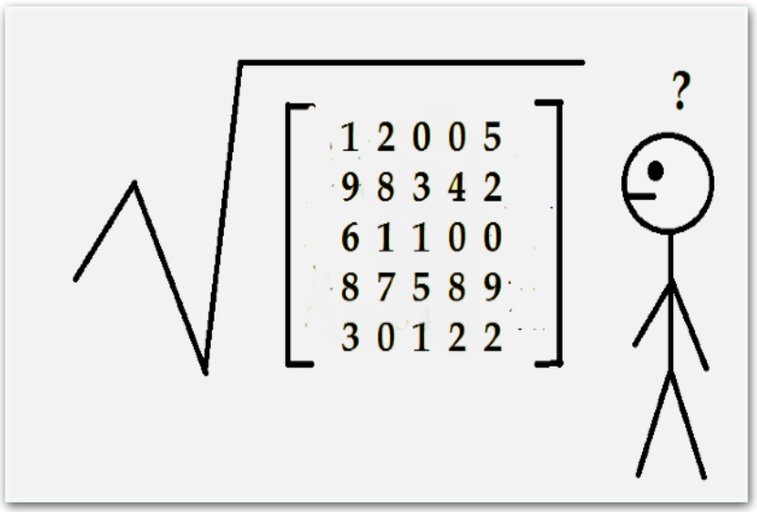
Many more theorems and corollaries

One-D version known as: **Geometric Programming**
www.stanford.edu/~boyd/papers/gp_tutorial.html
[Boyd, Kim, Vandenberghe, Hassibi (2007). 61pp.]

Examples

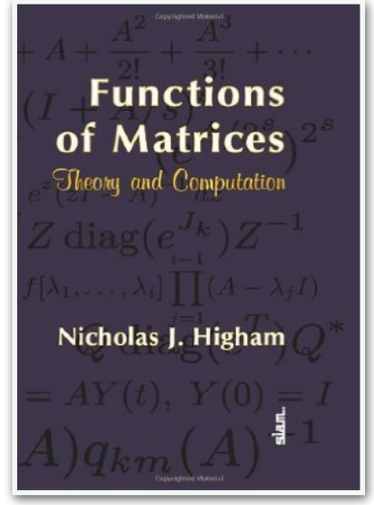
$$X \succ 0$$

Matrix square root



Broadly applicable

Key to ‘**logm**’



Matrix square root

Nonconvex optimization through the Euclidean lens

[Jain, Jin, Kakade, Netrapalli; Jul 2015]

$$\min_{X \in \mathbb{R}^{n \times n}} \|M - X^2\|_F^2$$

Matrix square root

Nonconvex optimization through the Euclidean lens

[Jain, Jin, Kakade, Netrapalli; Jul 2015]

$$\min_{X \in \mathbb{R}^{n \times n}} \|M - X^2\|_F^2$$

Gradient descent

$$X_{t+1} \leftarrow X_t - \eta(X_t^2 - M)X_t - \eta X_t(X_t^2 - M)$$

Simple algorithm; linear convergence; **nontrivial** analysis

Matrix square root

Geodesic

$$X \#_t Y := X^{\frac{1}{2}} (X^{-\frac{1}{2}} Y X^{-\frac{1}{2}})^t X^{\frac{1}{2}}$$

Midpoint

$$A^{\frac{1}{2}} = A \#_{\frac{1}{2}} I$$

Matrix square root

Nonconvex optimization through **non-Euclidean** lens

[Sra; Jul 2015]

$$\min_{X \succ 0} \quad \delta_S^2(X, A) + \delta_S^2(X, I)$$

$$\delta_S^2(X, Y) := \tfrac{1}{2} \log\det\left(\tfrac{X+Y}{2}\right) - \tfrac{1}{2}\log\det(XY)$$

(Curiously, this is the Jensen-Shannon divergence between multivariate Gaussians!)

Matrix square root

Nonconvex optimization through **non-Euclidean** lens

[Sra; Jul 2015]

$$\min_{X \succ 0} \quad \delta_S^2(X, A) + \delta_S^2(X, I)$$

Fixed-point iteration

$$X_{k+1} \leftarrow [(X_k + A)^{-1} + (X_k + I)^{-1}]^{-1}$$

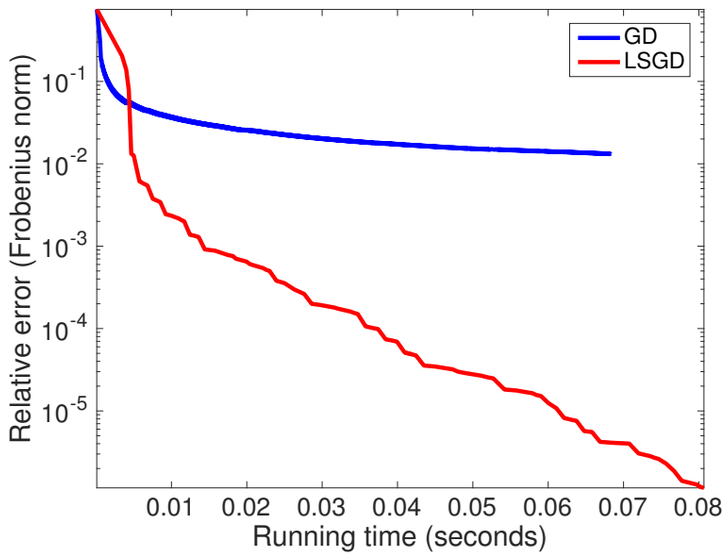
Simple method; linear convergence; 1/2 page analysis!

Global optimality thanks to geodesic convexity

$$\delta_S^2(X, Y) := \tfrac{1}{2} \log\det\left(\tfrac{X+Y}{2}\right) - \tfrac{1}{2} \log\det(XY)$$

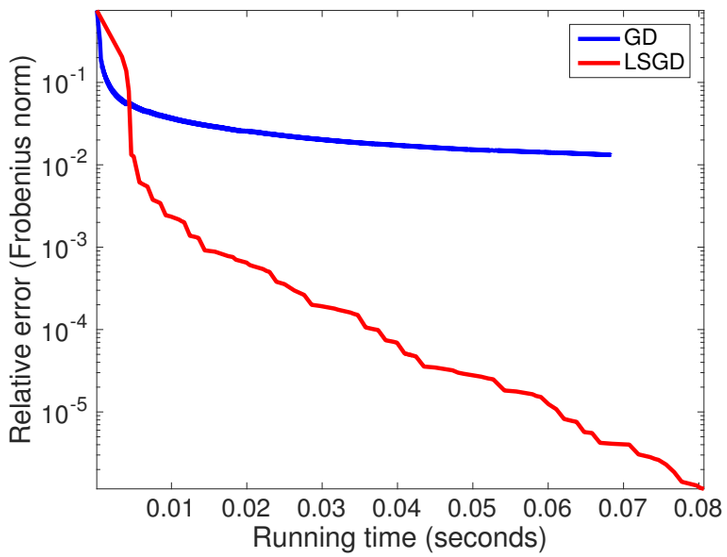
(Curiously, this is the Jensen-Shannon divergence between multivariate Gaussians!)

Matrix square root



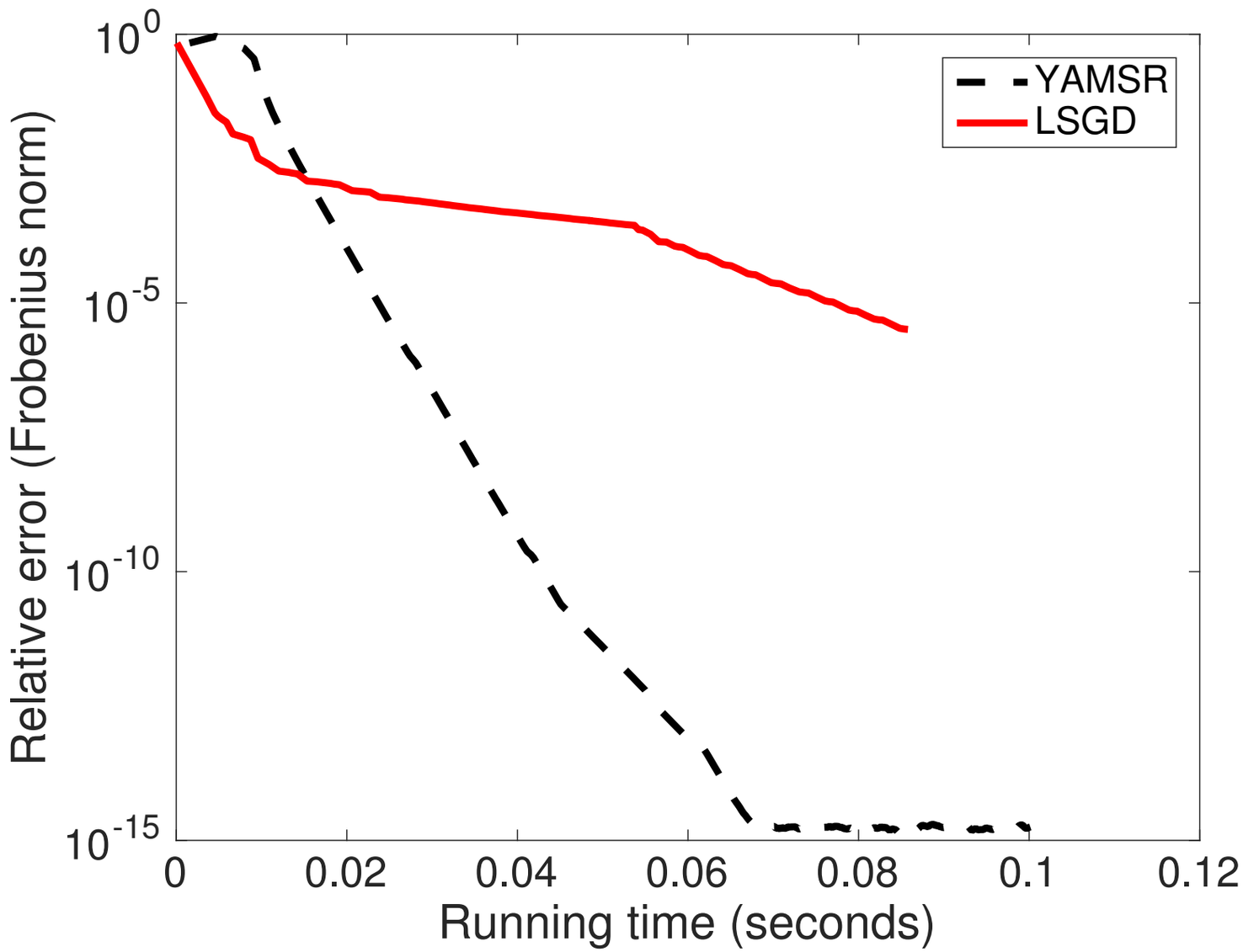
$$50 \times 50 \text{ matrix } I + \beta U U^T$$
$$\kappa \approx 64$$

Matrix square root



50×50 matrix $I + \beta U U^T$

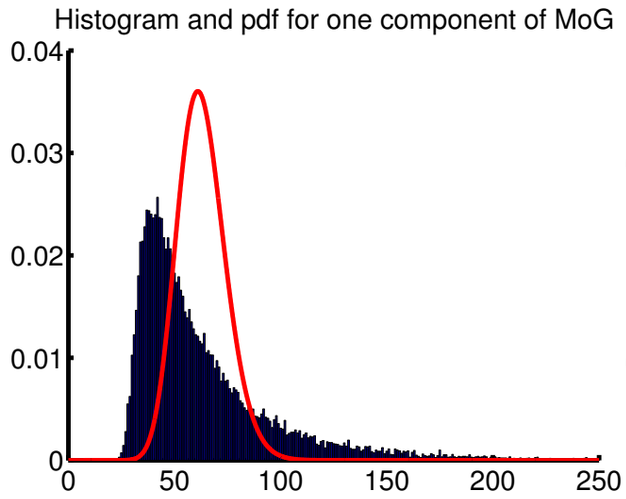
$\kappa \approx 64$

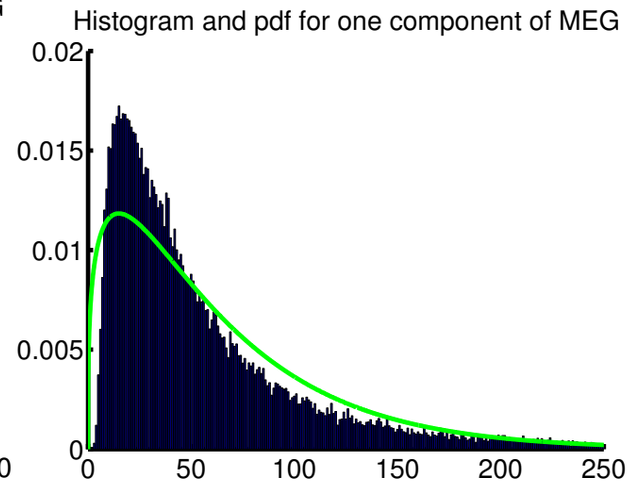


M-estimators: Elliptical Distributions

Natural Image Statistics

- Extract 200,000 training patches from 4167 images
- 10 sets of 100,000 test patches
- Log-transform intensities; add small amount of white noise

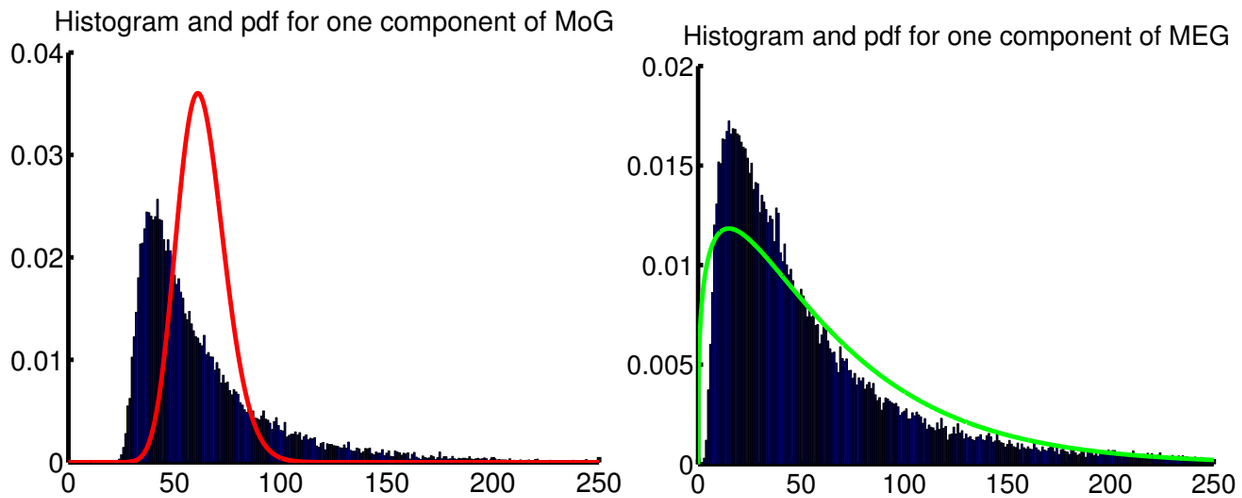


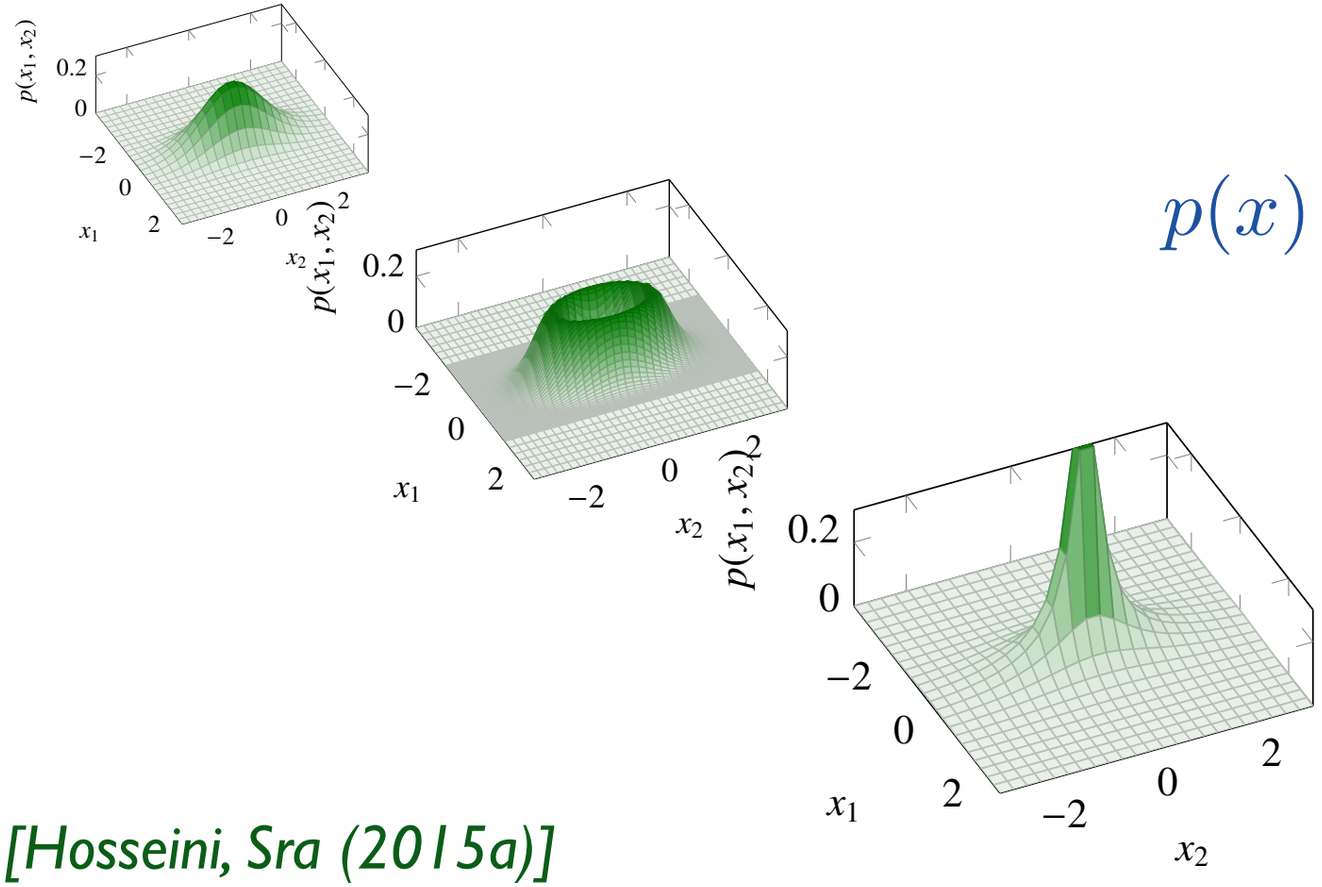


M-estimators: Elliptical Distributions

Natural Image Statistics

- Extract 200,000 training patches from 4167 images
- 10 sets of 100,000 test patches
- Log-transform intensities; add small amount of white noise





$$p(x) \propto \frac{(x^T \Sigma^{-1} x)^{a-\frac{d}{2}} e^{-\frac{a}{d} x^T \Sigma^{-1} x}}{\det(\Sigma)^{1/2}}$$

Elliptically Contoured Distributions (ECD)

[Hosseini, Sra (2015a)]

M-estimators for ECDs

Given observations $x_1, x_2, \ldots, x_n$ find m.l.e. by solving

$$\frac{n}{2}\log\det(\Sigma) - (a - \frac{d}{2})\sum_{i=1}^{n}\log(x_i^T\Sigma^{-1}x_i)$$
$$+ \frac{a}{d}\mathrm{trace}(\Sigma^{-1}\sum_i x_i x_i^T)$$

[Hosseini, Sra (2015)]

M-estimators for ECDs

Given observations $x_1, x_2, \ldots, x_n$ find m.l.e. by solving

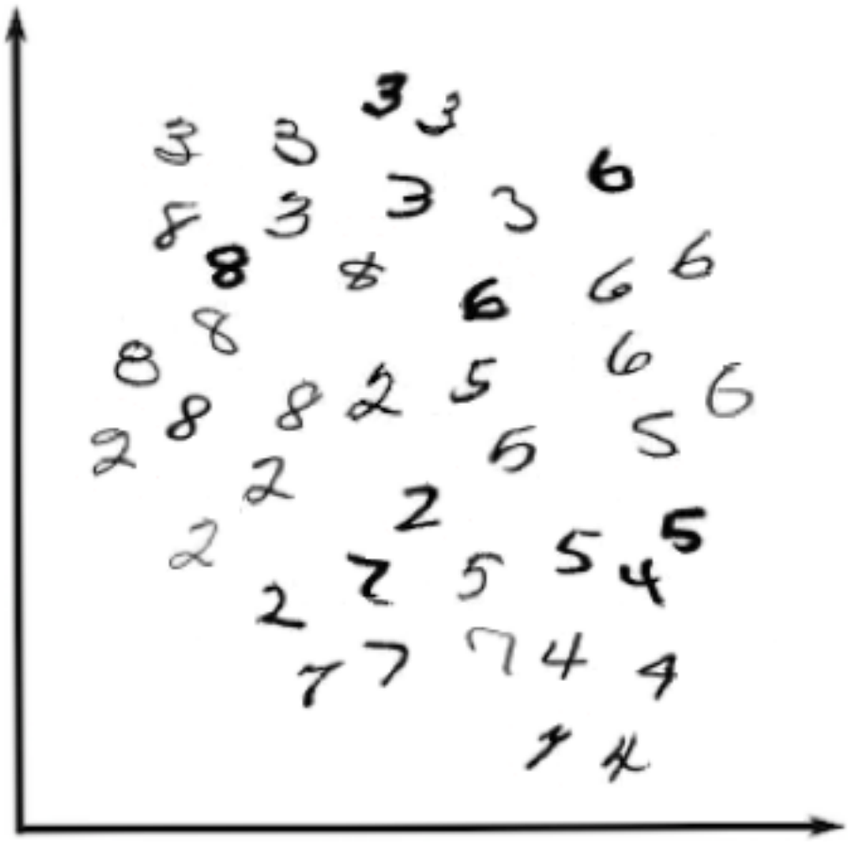
$$\frac{n}{2}\log\det(\Sigma) - (a - \tfrac{d}{2})\sum_{i=1}^{n}\log(x_i^T\Sigma^{-1}x_i) + \frac{a}{d}\mathrm{trace}(\Sigma^{-1}\sum_i x_i x_i^T)$$

convex or nonconvex, but
globally solvable due to g-convexity!

[Hosseini, Sra (2015)]

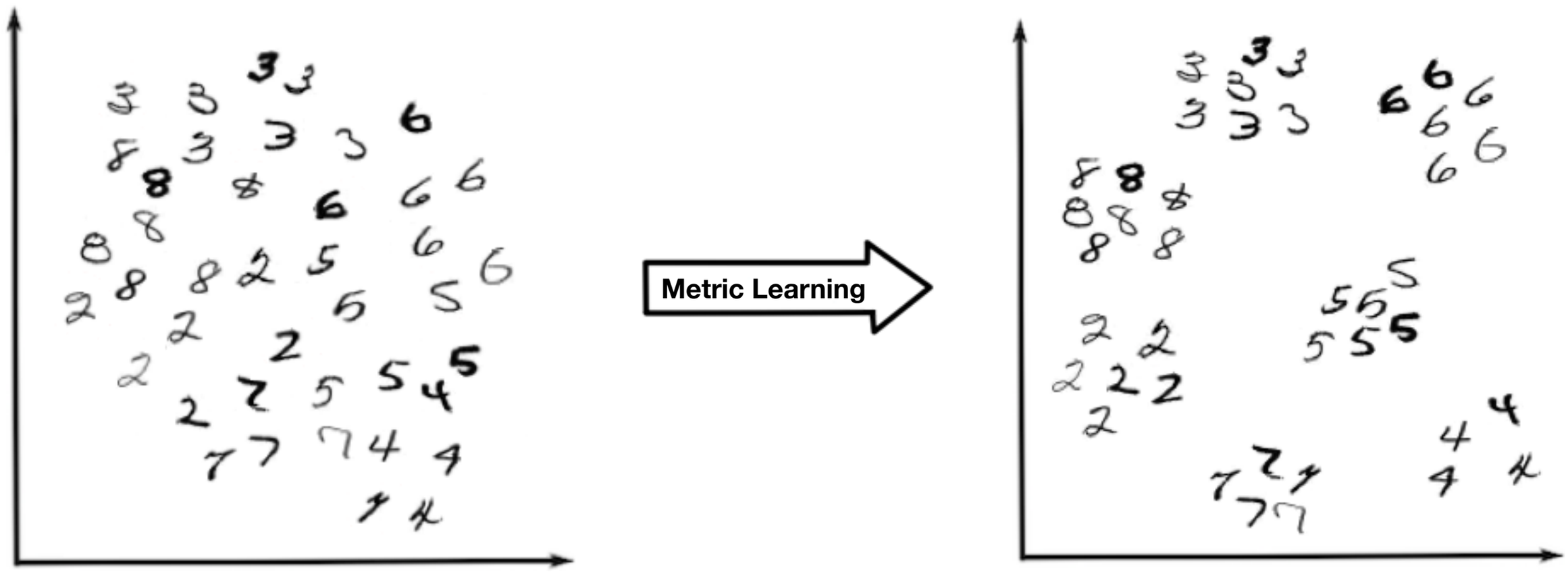
Metric learning

What does a metric learning method do?



Metric learning

What does a metric learning method do?



Euclidean metric learning

Pairwise constraints

$$\mathcal{S} := \{(\boldsymbol{x}_i, \boldsymbol{x}_j) \mid \boldsymbol{x}_i \text{ and } \boldsymbol{x}_j \text{ are in the same class}\}$$
$$\mathcal{D} := \{(\boldsymbol{x}_i, \boldsymbol{x}_j) \mid \boldsymbol{x}_i \text{ and } \boldsymbol{x}_j \text{ are in different classes}\}$$

Goal

given pairwise constraints learn Mahalanobis distance

$$d_{\boldsymbol{A}}(\boldsymbol{x}, \boldsymbol{y}) := (\boldsymbol{x} - \boldsymbol{y})^T \boldsymbol{A} (\boldsymbol{x} - \boldsymbol{y})$$

Positive definite matrix ***A***

Metric learning methods

MMC

[Xing, Jordan, Russell, Ng 2002]

LMNN

[Weinberger, Saul 2005]

ITML

[Davis, Kulis, Jain, Sra, Dhillon 2007]

Metric learning methods

MMC

[Xing, Jordan, Russell, Ng 2002]

$$\min_{\boldsymbol{A}\succeq 0} \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{S}} d_{\boldsymbol{A}}(\boldsymbol{x}_i,\boldsymbol{x}_j)$$
$$\text{such that} \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{D}} \sqrt{d_{\boldsymbol{A}}(\boldsymbol{x}_i,\boldsymbol{x}_j)} \geq 1$$

LMNN

[Weinberger, Saul 2005]

$$\min_{\boldsymbol{A}\succeq 0} \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{S}} \Big[(1-\mu)d_{\boldsymbol{A}}(\boldsymbol{x}_i,\boldsymbol{x}_j) + \mu\sum\nolimits_l (1-y_{il})\xi_{ijl}\Big]$$
$$d_{\boldsymbol{A}}(\boldsymbol{x}_i,\boldsymbol{x}_l) - d_{\boldsymbol{A}}(\boldsymbol{x}_i,\boldsymbol{x}_j) \geq 1-\xi_{ijl}$$
$$\xi_{ijl} \geq 0$$

ITML

[Davis, Kulis, Jain, Sra, Dhillon 2007]

$$\min_{\boldsymbol{A}\succeq 0} \quad D_{\mathrm{ld}}(\boldsymbol{A},\boldsymbol{A}_0)$$
$$\text{such that} \quad d_{\boldsymbol{A}}(\boldsymbol{x},\boldsymbol{y}) \leq u, \quad (\boldsymbol{x},\boldsymbol{y})\in\mathcal{S},$$
$$d_{\boldsymbol{A}}(\boldsymbol{x},\boldsymbol{y}) \geq l, \quad (\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}$$

$$D_{\mathrm{ld}}(\boldsymbol{A},\boldsymbol{A}_0) := \mathrm{tr}(\boldsymbol{A}\boldsymbol{A}_0^{-1}) - \log\det(\boldsymbol{A}\boldsymbol{A}_0^{-1}) - d$$

Metric learning methods

MMC

[Xing, Jordan, Russell, Ng 2002]

$$\min_{\boldsymbol{A}\succeq 0} \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{S}} d_{\boldsymbol{A}}(\boldsymbol{x}_i,\boldsymbol{x}_j)$$
$$\text{such that} \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{D}} \sqrt{d_{\boldsymbol{A}}(\boldsymbol{x}_i,\boldsymbol{x}_j)} \geq 1$$

LMNN

[Weinberger, Saul 2005]

$$\min_{\boldsymbol{A}\succeq 0} \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{S}} \Big[(1-\mu)d_{\boldsymbol{A}}(\boldsymbol{x}_i,\boldsymbol{x}_j) + \mu\sum\nolimits_l (1-y_{il})\xi_{ijl}\Big]$$
$$d_{\boldsymbol{A}}(\boldsymbol{x}_i,\boldsymbol{x}_l) - d_{\boldsymbol{A}}(\boldsymbol{x}_i,\boldsymbol{x}_j) \geq 1-\xi_{ijl}$$
$$\xi_{ijl} \geq 0$$

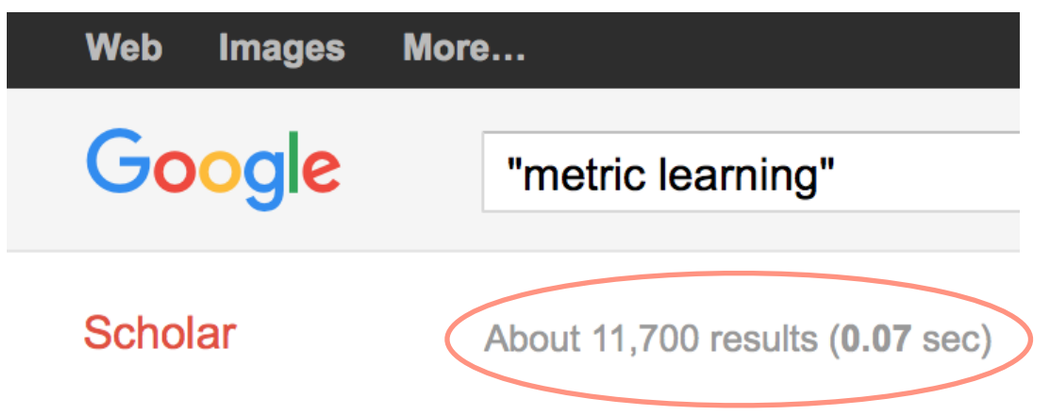
ITML

[Davis, Kulis, Jain, Sra, Dhillon 2007]

$$\min_{\boldsymbol{A}\succeq 0} \quad D_{\mathrm{ld}}(\boldsymbol{A},\boldsymbol{A}_0)$$
$$\text{such that} \quad d_{\boldsymbol{A}}(\boldsymbol{x},\boldsymbol{y}) \leq u, \quad (\boldsymbol{x},\boldsymbol{y})\in\mathcal{S},$$
$$d_{\boldsymbol{A}}(\boldsymbol{x},\boldsymbol{y}) \geq l, \quad (\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}$$

$$D_{\mathrm{ld}}(\boldsymbol{A},\boldsymbol{A}_0) := \mathrm{tr}(\boldsymbol{A}\boldsymbol{A}_0^{-1}) - \log\det(\boldsymbol{A}\boldsymbol{A}_0^{-1}) - d$$

tons of other methods!



Related work

- FlatGeo *[Meyer, Bonnabel, Sepulchre '11]*

$$\min_{\boldsymbol{A}\succeq 0} \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{S}} \max\big(\,0\,,\; l - d_{\boldsymbol{A}}(\boldsymbol{x}_i,\boldsymbol{x}_j)\,\big)^2 + \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{D}} \max\big(\,0\,,\; d_{\boldsymbol{A}}(\boldsymbol{x}_i,\boldsymbol{x}_j) - u\,\big)^2$$

- LMNN *[Weinberger, Saul '05]*; and 100s of other methods out there!

Related work

- FlatGeo *[Meyer, Bonnabel, Sepulchre '11]*

$$\min_{\boldsymbol{A}\succeq 0} \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{S}} \max\big(\,0\,,\; l - d_{\boldsymbol{A}}(\boldsymbol{x}_i,\boldsymbol{x}_j)\,\big)^2 + \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{D}} \max\big(\,0\,,\; d_{\boldsymbol{A}}(\boldsymbol{x}_i,\boldsymbol{x}_j) - u\,\big)^2$$

- LMNN *[Weinberger, Saul '05]*; and 100s of other methods out there!

- **Main concerns**

  They does not scale well to the large problems, with respect to:

  1. The number of constraints
  2. The dimensionality

A simple new way for metric learning

Euclidean idea

$$\min_{\boldsymbol{A}\succeq 0} \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{S}} d_{\boldsymbol{A}}(\boldsymbol{x}_i,\boldsymbol{x}_j) - \lambda \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{D}} d_{\boldsymbol{A}}(\boldsymbol{x}_i,\boldsymbol{x}_j)$$

A simple new way for metric learning

Euclidean idea

$$\min_{\boldsymbol{A}\succeq 0} \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{S}} d_{\boldsymbol{A}}(\boldsymbol{x}_i,\boldsymbol{x}_j) - \lambda \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{D}} d_{\boldsymbol{A}}(\boldsymbol{x}_i,\boldsymbol{x}_j)$$

A simple new way for metric learning

Euclidean idea

$$\min_{\boldsymbol{A}\succeq 0} \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{S}} d_{\boldsymbol{A}}(\boldsymbol{x}_i,\boldsymbol{x}_j) - \lambda \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{D}} d_{\boldsymbol{A}}(\boldsymbol{x}_i,\boldsymbol{x}_j)$$

New idea

$$\min_{\boldsymbol{A}\succeq 0} \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{S}} d_{\boldsymbol{A}}(\boldsymbol{x}_i,\boldsymbol{x}_j) + \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{D}} d_{\boldsymbol{A}^{-1}}(\boldsymbol{x}_i,\boldsymbol{x}_j)$$

A simple new way for metric learning

Euclidean idea

$$\min_{\boldsymbol{A}\succeq 0} \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{S}} d_{\boldsymbol{A}}(\boldsymbol{x}_i,\boldsymbol{x}_j) - \lambda \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{D}} d_{\boldsymbol{A}}(\boldsymbol{x}_i,\boldsymbol{x}_j)$$

New idea

$$\min_{\boldsymbol{A}\succeq 0} \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{S}} d_{\boldsymbol{A}}(\boldsymbol{x}_i,\boldsymbol{x}_j) + \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{D}} d_{\boldsymbol{A}^{-1}}(\boldsymbol{x}_i,\boldsymbol{x}_j)$$

Equivalently solve

A simple new way for metric learning

Euclidean idea

$$\min_{\boldsymbol{A}\succeq 0} \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{S}} d_{\boldsymbol{A}}(\boldsymbol{x}_i,\boldsymbol{x}_j) - \lambda \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{D}} d_{\boldsymbol{A}}(\boldsymbol{x}_i,\boldsymbol{x}_j)$$

New idea

$$\min_{\boldsymbol{A}\succeq 0} \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{S}} d_{\boldsymbol{A}}(\boldsymbol{x}_i,\boldsymbol{x}_j) + \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{D}} d_{\boldsymbol{A}^{-1}}(\boldsymbol{x}_i,\boldsymbol{x}_j)$$

Equivalently solve

$$\min_{\boldsymbol{A}\succ 0} \quad h(\boldsymbol{A}) := \operatorname{tr}(\boldsymbol{A}\boldsymbol{S}) + \operatorname{tr}(\boldsymbol{A}^{-1}\boldsymbol{D})$$

$$\boldsymbol{S} := \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{S}} (\boldsymbol{x}_i-\boldsymbol{x}_j)(\boldsymbol{x}_i-\boldsymbol{x}_j)^T,$$

$$\boldsymbol{D} := \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{D}} (\boldsymbol{x}_i-\boldsymbol{x}_j)(\boldsymbol{x}_i-\boldsymbol{x}_j)^T$$

[Habibzadeh, Hosseini, Sra, ICML 2016]

A simple new way for metric learning

Closed form solution!

$$\nabla h(\boldsymbol{A}) = 0 \quad \Leftrightarrow \quad \boldsymbol{S} - \boldsymbol{A}^{-1}\boldsymbol{D}\boldsymbol{A}^{-1} = 0$$

A simple new way for metric learning

$$X\#_t Y := X^{\frac{1}{2}}(X^{-\frac{1}{2}}YX^{-\frac{1}{2}})^t X^{\frac{1}{2}}$$

Closed form solution!

$$\nabla h(\boldsymbol{A}) = 0 \quad \Leftrightarrow \quad \boldsymbol{S} - \boldsymbol{A}^{-1}\boldsymbol{D}\boldsymbol{A}^{-1} = 0$$

$$\boldsymbol{A} = \boldsymbol{S}^{-1}\#_{\frac{1}{2}}\boldsymbol{D}$$

A simple new way for metric learning

$$X\#_t Y := X^{\frac{1}{2}}(X^{-\frac{1}{2}}YX^{-\frac{1}{2}})^t X^{\frac{1}{2}}$$

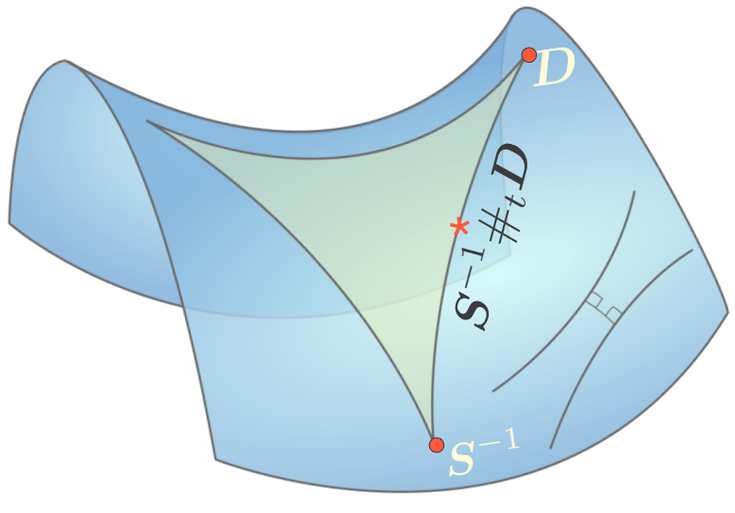
Closed form solution!

$$\nabla h(\boldsymbol{A}) = 0 \quad \Leftrightarrow \quad \boldsymbol{S} - \boldsymbol{A}^{-1}\boldsymbol{D}\boldsymbol{A}^{-1} = 0$$

$$\boldsymbol{A} = \boldsymbol{S}^{-1}\#_{\frac{1}{2}}\boldsymbol{D}$$

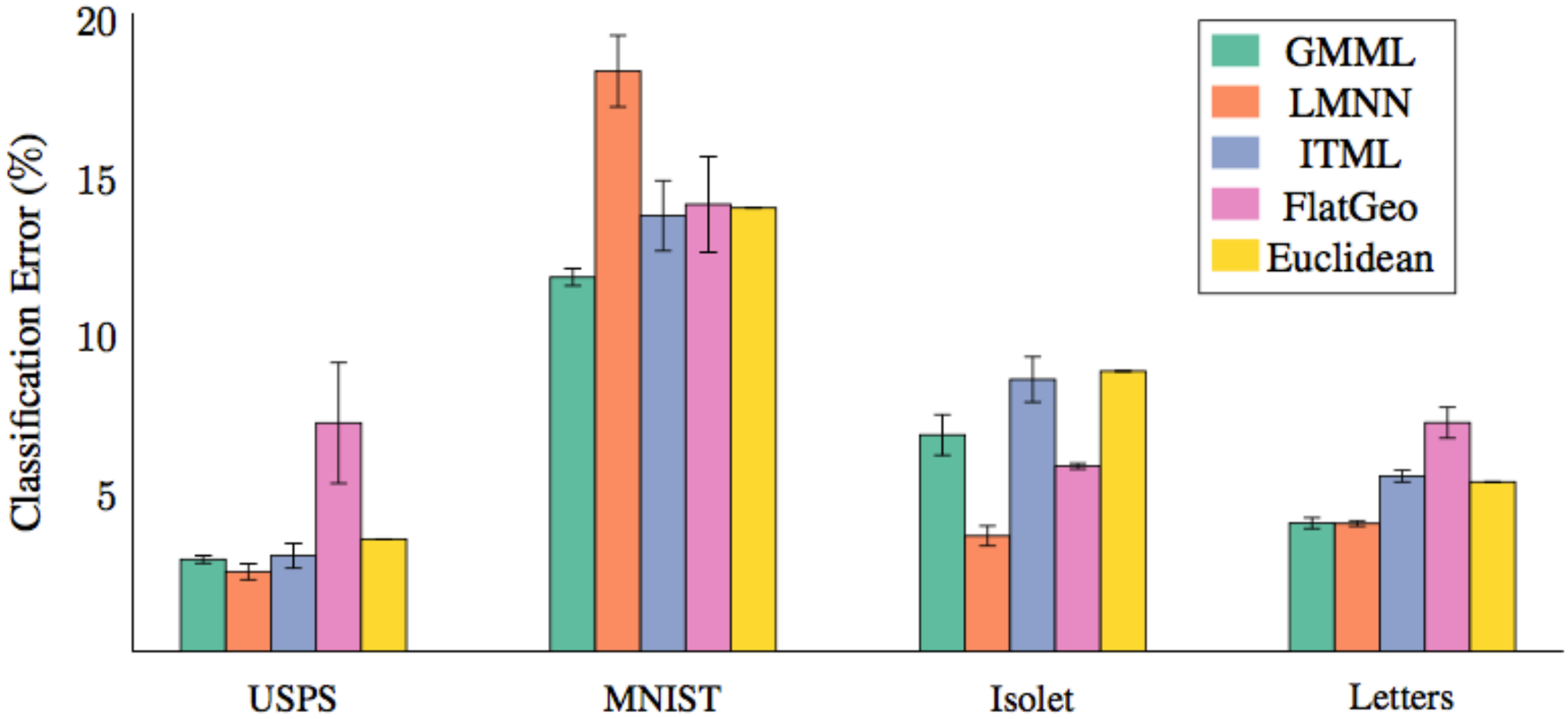
More generally

$$\min_{\boldsymbol{A}\succ 0} \quad (1-t)\delta_R^2(\boldsymbol{S}^{-1}, \boldsymbol{A}) + t\delta_R^2(\boldsymbol{D}, \boldsymbol{A})$$



$$\boldsymbol{S}^{-1}\#_t\boldsymbol{D}$$

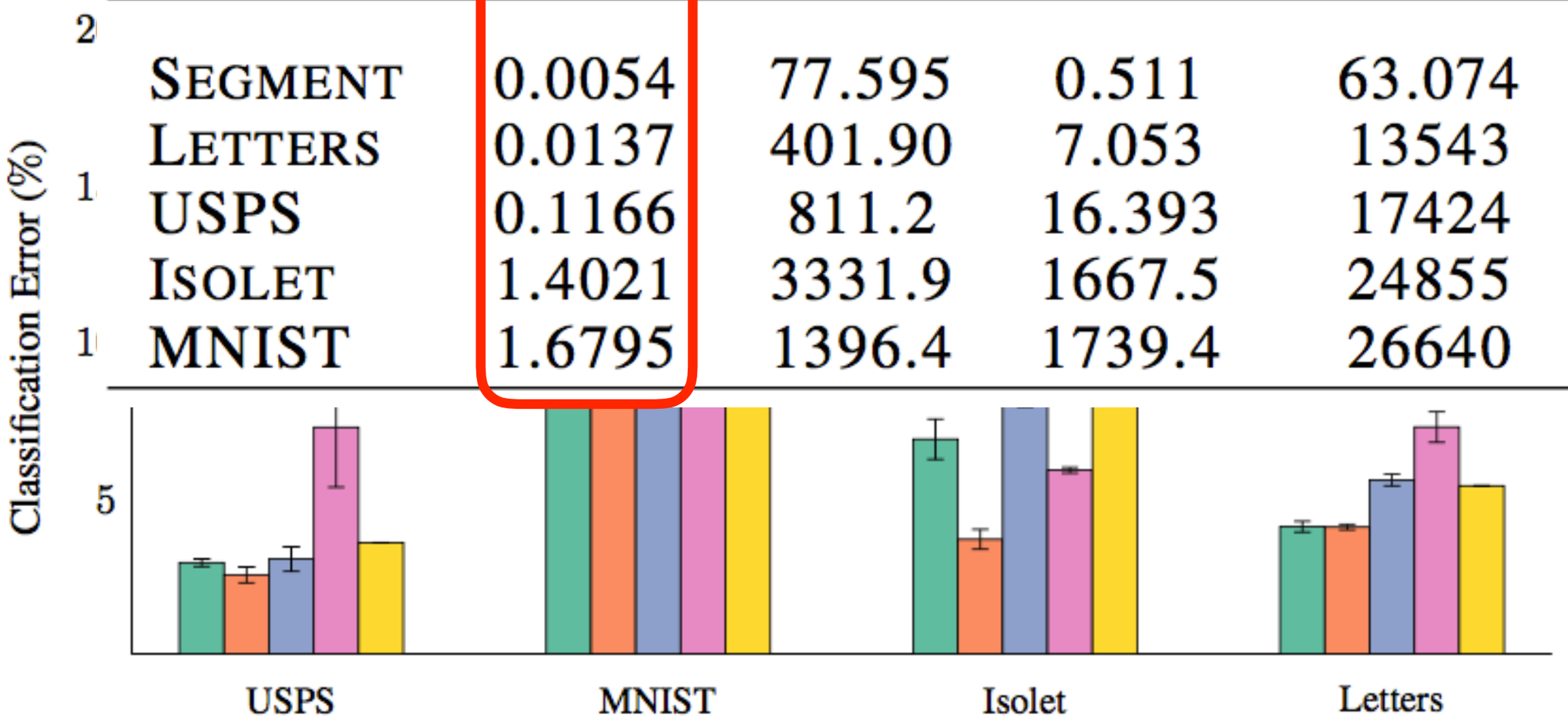
Experiments



[Habibzadeh, Hosseini, Sra ICML 2016]

Experiments

Data set	GMML	LMNN	ITML	FlatGeo
Segment	0.0054	77.595	0.511	63.074
Letters	0.0137	401.90	7.053	13543
USPS	0.1166	811.2	16.393	17424
Isolet	1.4021	3331.9	1667.5	24855
MNIST	1.6795	1396.4	1739.4	26640



[Habibzadeh, Hosseini, Sra ICML 2016]

Brascamp-Lieb Constant

Brascamp-Lieb Constant

$$\int_{\mathbb{R}^n} \prod_{i=1}^{m} f_i(B_i x)^{p_i}\, dx \leq D^{-1/2} \prod_{i=1}^{m} \left(\int_{\mathbb{R}^{n_i}} f_i(y)\, dy \right)^{p_i}$$

$$p_i > 0, f_i \geq 0 \qquad \sum_{i=1}^{m} p_i n_i = n$$

powerful inequality; includes Hölder, Loomis-Whitney, Young's, many others!

Brascamp-Lieb Constant

$$\int_{\mathbb{R}^n} \prod_{i=1}^{m} f_i(B_i x)^{p_i}\, dx \leq D^{-1/2} \prod_{i=1}^{m} \left(\int_{\mathbb{R}^{n_i}} f_i(y)\, dy \right)^{p_i}$$

$$D := \inf \left\{ \frac{\det\left(\sum_i p_i B_i^* X_i B_i\right)}{\prod_i (\det X_i)^{p_i}} \middle| X_i \succ 0, n_i \times n_i, \right\}$$

$$p_i > 0, f_i \geq 0 \qquad \sum_{i=1}^{m} p_i n_i = n$$

powerful inequality; includes Hölder, Loomis-Whitney, Young's, many others!

Brascamp-Lieb constant

$$\min_{X_1,\ldots,X_m \succ 0} \log\det\left(\sum\nolimits_i p_i B_i^* X_i B_i\right) - \sum\nolimits_i p_i \log\det X_i$$

- Arises in geometric complexity theory:
  [Garg, Gurvits, Oliveira, Wigderson; Jul 2016]

Brascamp-Lieb constant

$$\min_{X_1,\ldots,X_m \succ 0} \log\det\left(\sum\nolimits_i p_i B_i^* X_i B_i\right) - \sum\nolimits_i p_i \log\det X_i$$

- Arises in geometric complexity theory:
  [Garg, Gurvits, Oliveira, Wigderson; Jul 2016]

Exercise

Prove this is a g-convex opt problem

Brascamp-Lieb constant

$$\min_{X_1,\ldots,X_m \succ 0} \log\det\left(\sum\nolimits_i p_i B_i^* X_i B_i\right) - \sum\nolimits_i p_i \log\det X_i$$

- Arises in geometric complexity theory:
  [Garg, Gurvits, Oliveira, Wigderson; Jul 2016]

Exercise

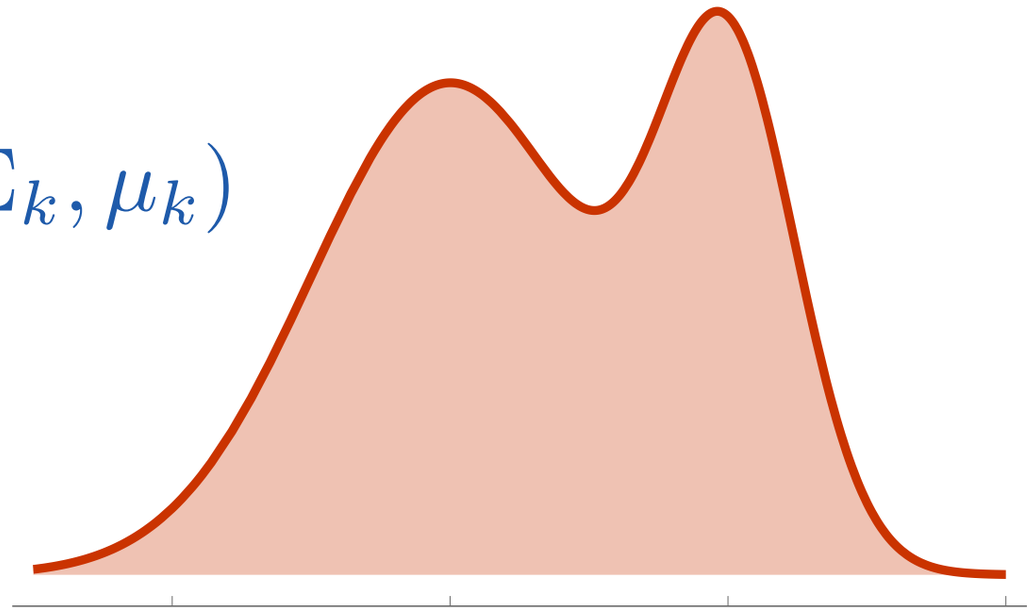
Prove this is a g-convex opt problem

- G-convexity yields transparent algorithms & complexity analysis for global optimum.

Gaussian mixture models

$$p_{\text{mix}}(x) := \sum_{k=1}^{K} \pi_k p_{\mathcal{N}}(x; \Sigma_k, \mu_k)$$

$$\max \prod_i p_{\text{mix}}(x_i)$$



Expectation maximization (EM): default choice

$$p_{\mathcal{N}}(x; \Sigma, \mu) \propto \frac{1}{\sqrt{\det(\Sigma)}} \exp\left(-\tfrac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right)$$

Gaussian mixture models

- **Nonconvex –** difficult, possibly several local optima
- **GMMs –** Recent surge of theoretical results
- **In Practice –** EM still default choice

Difficulty: Positive definiteness constraint on Σ_k

[Hosseini, Sra NIPS 2015]

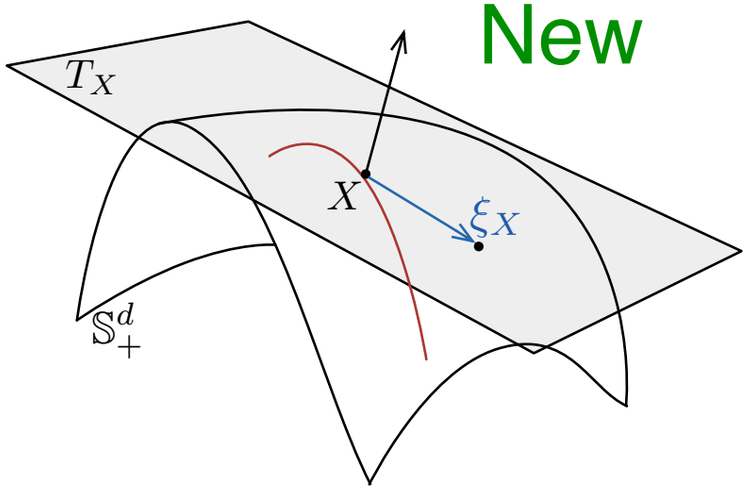
Gaussian mixture models

- **Nonconvex –** difficult, possibly several local optima
- **GMMs –** Recent surge of theoretical results
- **In Practice –** EM still default choice

Difficulty: Positive definiteness constraint on Σ_k

Geometric opt

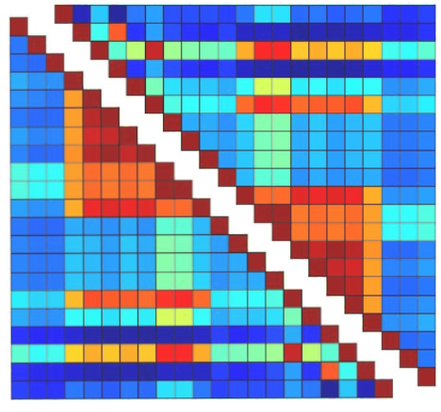
New



Unconstrained, Cholesky

Folklore

$$LL^T$$



[Hosseini, Sra NIPS 2015]

Failure of "obvious" LL^T

sep.	EM	CG-LL^T
0.2	52s // 12.7	614s // 12.7
1	160s // 13.4	435s // 13.5
5	72s // 12.8	426s // 12.8

$$\|\mu_i - \mu_j\| \geq \text{sep} \max_{ij}\{\text{tr}\Sigma_i, \text{tr}\Sigma_j\}$$

d=20
simulation

Failure of Riemannian optimization

K	EM	Riem-CG
2	17s // 29.28	947s // 29.28
5	202s // 32.07	5262s // 32.07
10	2159s // 33.05	17712s // 33.03

manopt.org

Riemannian opt. toolbox

d=35 images dataset

What’s wrong?

What's wrong?

log-likelihood for one component

$$-\frac{n}{2}\log\det\Sigma - \frac{1}{2}\sum_{i=1}^{n}(x_i-\mu)^T\Sigma^{-1}(x_i-\mu)$$

What's wrong?

log-likelihood for one component

$$-\frac{n}{2}\log\det\Sigma - \frac{1}{2}\sum_{i=1}^{n}(x_i-\mu)^T\Sigma^{-1}(x_i-\mu)$$

Euclidean convex problem
Not geodesically convex

What's wrong?

log-likelihood for one component

$$-\frac{n}{2}\log\det\Sigma - \frac{1}{2}\sum_{i=1}^{n}(x_i-\mu)^T\Sigma^{-1}(x_i-\mu)$$

Euclidean convex problem
Not geodesically convex

Reformulate as g-convex

$$y_i = \begin{bmatrix} x_i \\ 1 \end{bmatrix} \quad S = \begin{bmatrix} \Sigma + \mu\mu^T & \mu \\ \mu^T & 1 \end{bmatrix}$$

$$\max_{S\succ 0} \ \widehat{\mathcal{L}}(S) := \sum_{i=1}^{n} \log q_{\mathcal{N}}(y_i; S),$$

Thm. The modified log-likelihood is g-convex. Local max of modified mixture LL is local max of original.

Success of geometric optimization

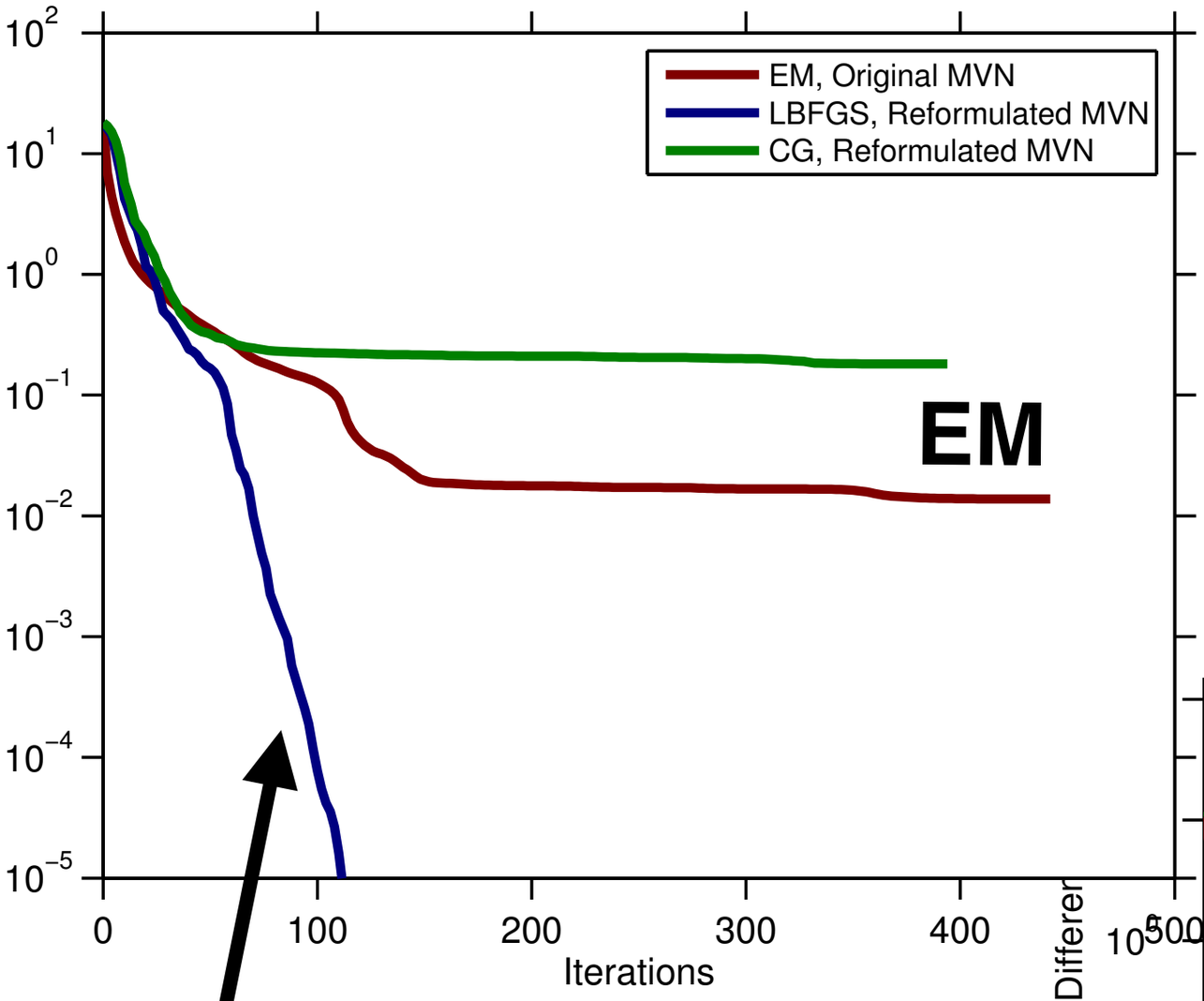
K	EM	Riem-CG	L-RBFGS
2	17s // 29.28	**18s** // 29.28	**14s** // 29.28
5	202s // 32.07	**140s** // 32.07	**117s** // 32.07
10	2159s // 33.05	**1048s** // 33.06	**658s** // 33.06

Riem-CG (manopt) savings:

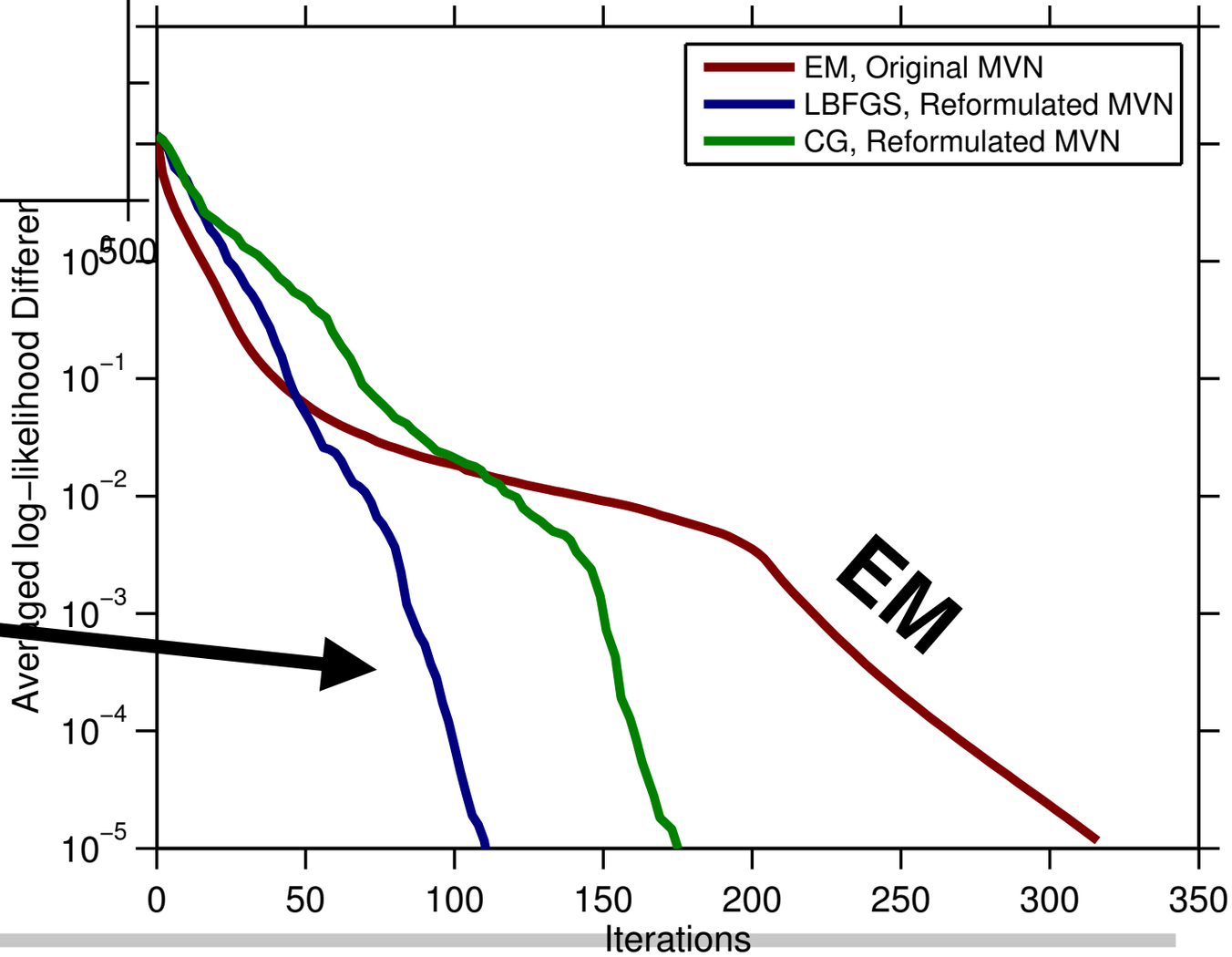
947→**18**; 5262 →**140**; 17712→**1048**

d=35 images dataset

github.com/utvisionlab/mixest



corel, K=7, d=57



year, K=9, d=90

Extensions

- ➡ Large-scale mixture models
  - Stochastic optimization
  - Parallel & distributed
- ➡ Easily use priors (Wishart and beyond)
- ➡ Allows additional constraints
- ➡ Non Gaussian mixtures

github.com/utvisionlab/mixest

suvrit.de/gopt.html

Extensions

➡ Large-scale mixture models

Stochastic optimization ✓

Parallel & distributed

➡ Easily use priors (Wishart and beyond)

➡ Allows additional constraints

➡ Non Gaussian mixtures

github.com/utvisionlab/mixest

suvrit.de/gopt.html

Extensions

➡ Large-scale mixture models

Stochastic optimization ✓

Parallel & distributed

➡ Easily use priors (Wishart and beyond) ✓

➡ Allows additional constraints

➡ Non Gaussian mixtures

github.com/utvisionlab/mixest

suvrit.de/gopt.html

Extensions

➡ Large-scale mixture models

Stochastic optimization ✓

Parallel & distributed

➡ Easily use priors (Wishart and beyond) ✓

➡ Allows additional constraints ✓

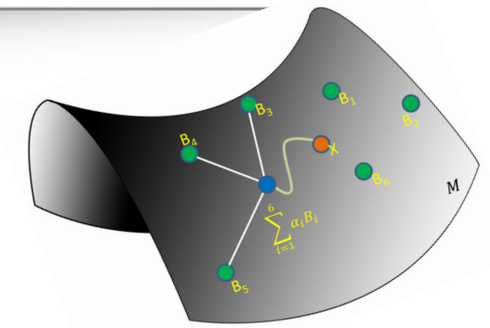
➡ Non Gaussian mixtures

github.com/utvisionlab/mixest

suvrit.de/gopt.html

Riemannian finite-sum problems

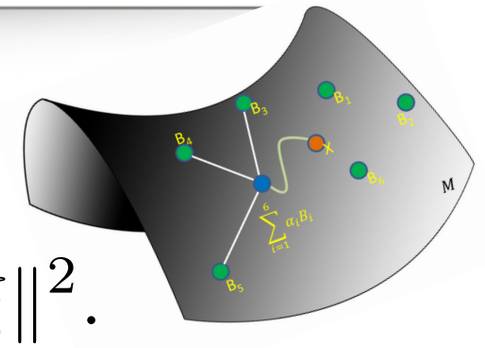
$$\min_{x \in \mathcal{M}} \quad f(x) := \frac{1}{n} \sum_{i=1}^{n} f_i(x)$$



- $\mathcal{M}$ is a Riemannian manifold
- g-convex and g-nonconvex ‘f’ allowed
- First global complexity results for stochastic methods on Riemannian manifolds
- Riemannian SVRG

[Zhang, Reddi, Sra, NIPS 2016]

Riemannian SGD with Arbitrary Retraction



(i) The function satisfies the Lipschitz growth bound

$$f(\mathrm{Ret}_x(\xi)) \leq f(x) + \langle \nabla f(x), \xi \rangle + \frac{L}{2}\|\xi\|^2.$$

(ii) The stochastic gradients in all iterations are unbiased, i.e.,

$$\mathbb{E}[\nabla f_{i_t}(x_t) - \nabla f(x_t)] = 0.$$

(iii) The stochastic gradients have bounded variance, so that

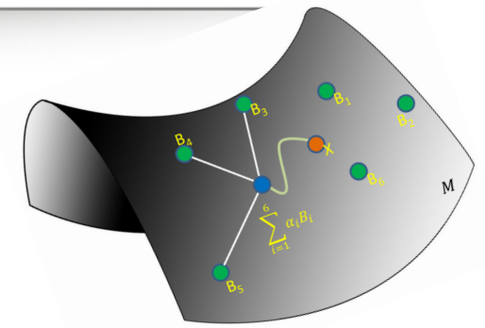
$$\mathbb{E}[\|\nabla f_{i_t}(x_t) - \nabla f(x_t)\|^2] \leq \sigma^2, \qquad 0 \leq \sigma < \infty.$$

- **We can proof the following convergence result**

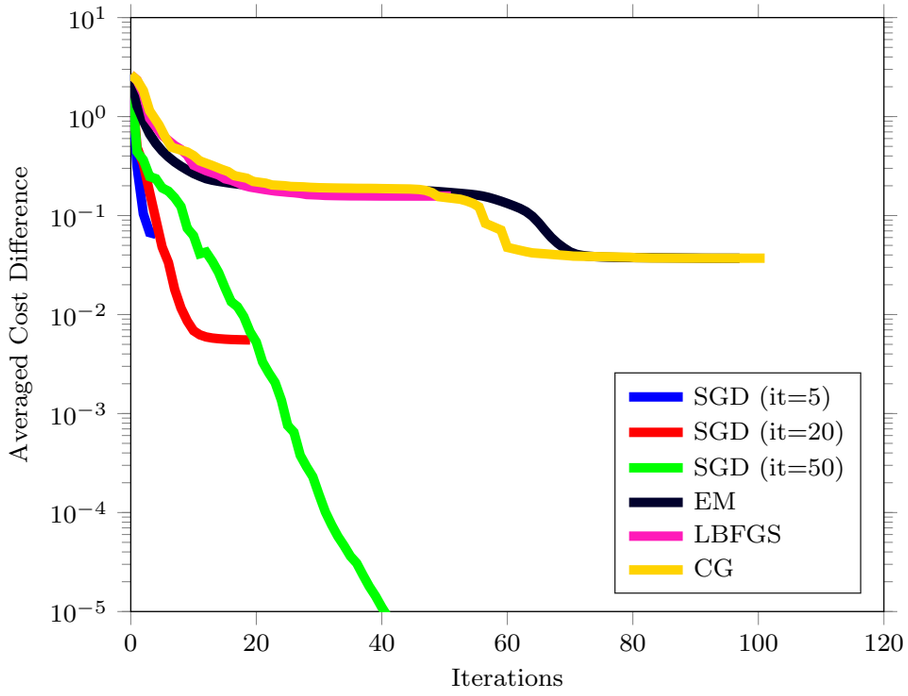
$$\mathbb{E}[\|\nabla f(x_a)\|^2] \leq \frac{2L\Delta_1}{T} + \left(c + c^{-1}\Delta_1\right)\frac{L\sigma}{\sqrt{T}} = \mathcal{O}\left(\frac{1}{T}\right) + \mathcal{O}\left(\frac{1}{\sqrt{T}}\right)$$

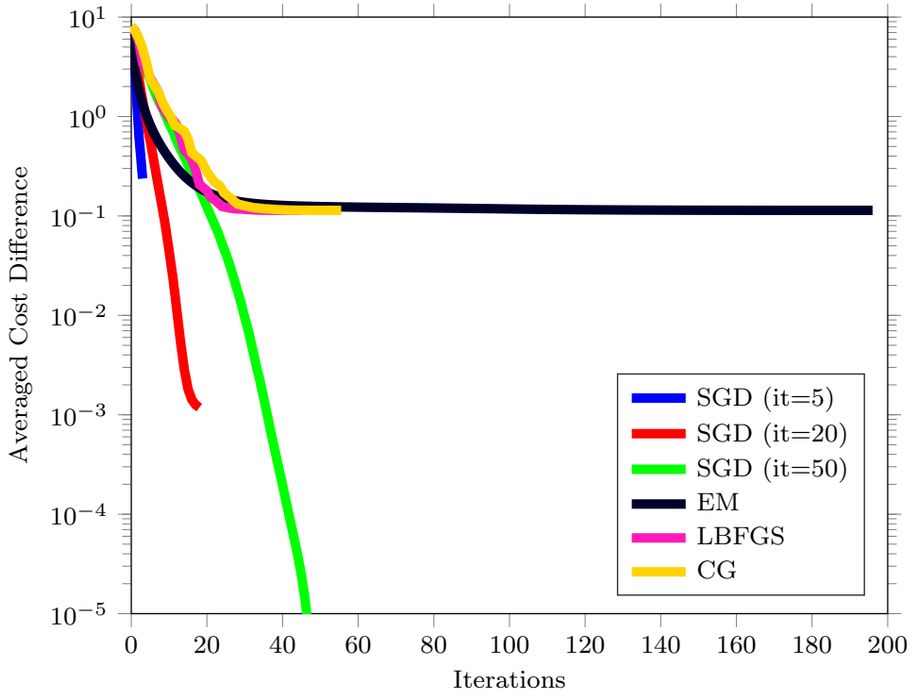
[Hosseini, Sra, MAPR 2019]

Riemannian SGD with Arbitrary Retraction



- The approach successfully applied to the case of GMMs
- To show that gradients are bounded, we proved a key theorem that iterations stay within a compact set





[Hosseini, Sra, MAPR]

In particular, we study the global complexity of first-order g-convex optimization

Global Complexity

Gradient Descent

Stochastic Gradient Descent

Coordinate Descent

Accelerated Gradient Descent

Fast Incremental Gradient

... ...

$$\mathbb{E}[f(x_a) - f(x^*)] \leq ?$$

Convex Optimization

G-Convex Optimization

Convergence rates depend on lower bounds on the sectional curvature

(Sub)gradient	convex	g-convex
Lipschitz	$O\left(\sqrt{\frac{1}{t}}\right)$	$O\left(\sqrt{\frac{\zeta_{\max}}{t}}\right)$
Strongly convex / smooth	$O\left(\frac{1}{t}\right)$	$O\left(\frac{\zeta_{\max}}{t}\right)$
Strongly convex & smooth	$O\left(\left(1-\frac{\mu}{L_g}\right)^t\right)$	$O\left(\left(1-\min\left\{\frac{1}{\zeta_{\max}}, \frac{\mu}{L_g}\right\}\right)^t\right)$
Stochastic (sub)gradient	... ...	

$$\zeta_{\max} \triangleq \frac{\sqrt{|\kappa_{\min}|}D}{\tanh\left(\sqrt{|\kappa_{\min}|}D\right)}$$

See paper for other interesting results *[Zhang, Sra, COLT 2016]*

Invitation for more!

- Other ML application
- Faster stochastic optimization
- Beyond Riemannian manifolds
- Faster gradient based algorithms
- Handling more complex constraints
- Lower bound theory
- ...

Invitation for more!

- Other ML application
- Faster stochastic optimization
- Beyond Riemannian manifolds
- Faster gradient based algorithms
- Handling more complex constraints
- Lower bound theory
- ...

Thanks!