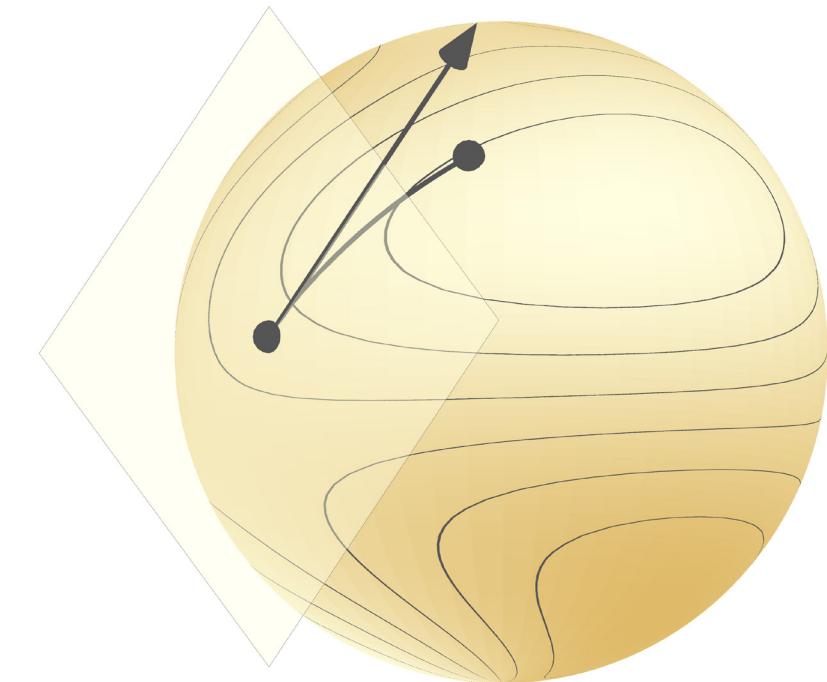


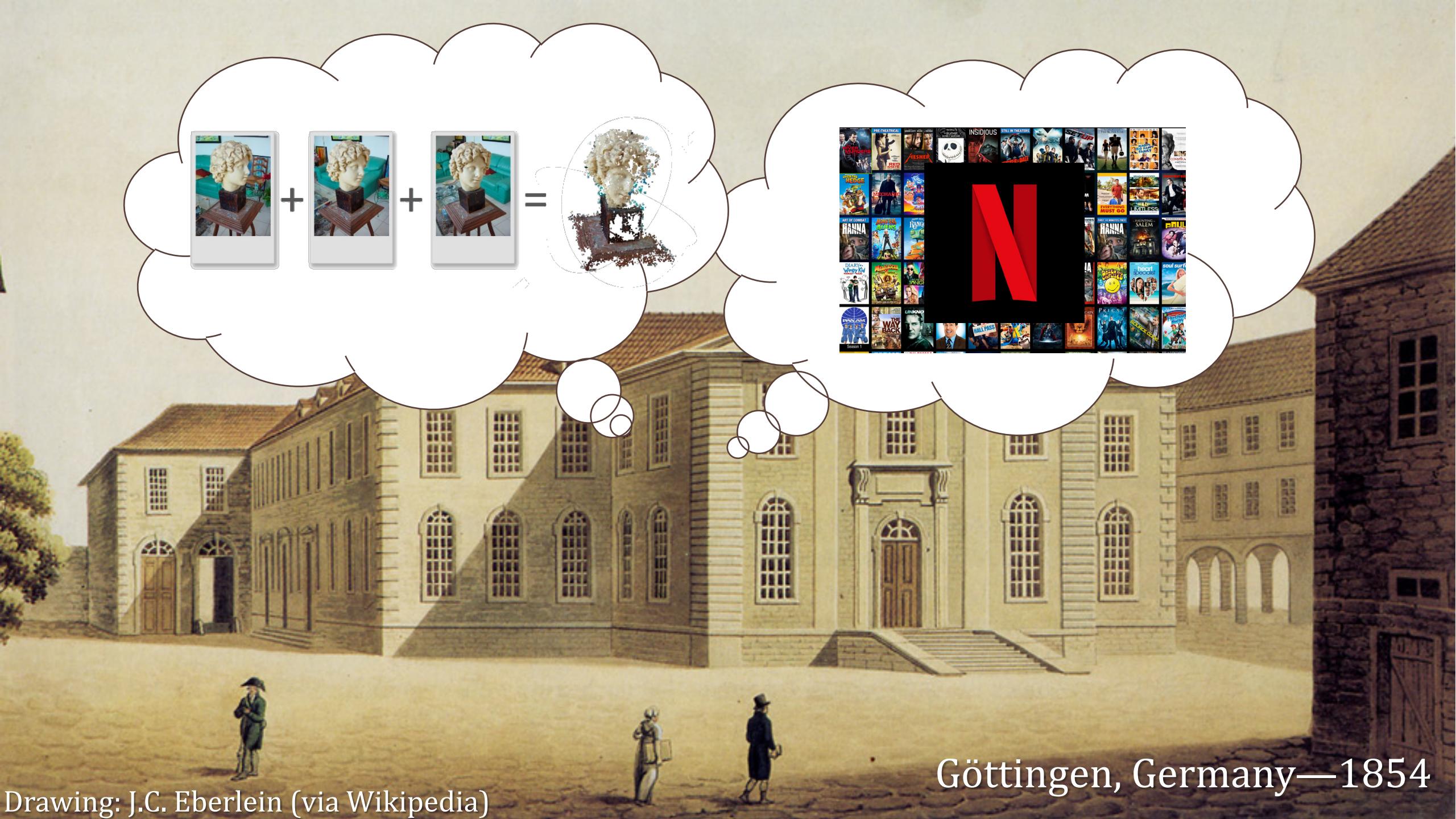
Slides and links: [nicolasboumal.net/SIAMOP2023](http://nicolasboumal.net/SIAMOP2023)

# A tutorial on Riemannian optimization

Context, geometry, algorithms, resources

SIAM Conference on Optimization, June 2023  
Nicolas Boumal – chair of continuous optimization  
Institute of Mathematics, EPFL





Göttingen, Germany—1854

Drawing: J.C. Eberlein (via Wikipedia)

# Step 0 in optimization

It starts with a **set**  $S$  and a **function**  $f: S \rightarrow \mathbf{R}$ . We want to compute:

$$\min_{x \in S} f(x)$$

These **bare objects** fully specify the problem.

Any additional **structure** on  $S$  and  $f$  may (and should) be exploited for **algorithmic purposes** but is not part of the problem.

# Classical unconstrained optimization

The search space *is* a **linear space**, e.g.,  $S = \mathbf{R}^n$ :

$$\min_{x \in \mathbf{R}^n} f(x)$$

We can *choose* to turn  $\mathbf{R}^n$  into a **Euclidean space**:  $\langle u, v \rangle = u^\top v$ .

If  $f$  is differentiable, we have a **gradient**  $\text{grad}f$  and **Hessian**  $\text{Hess}f$ .

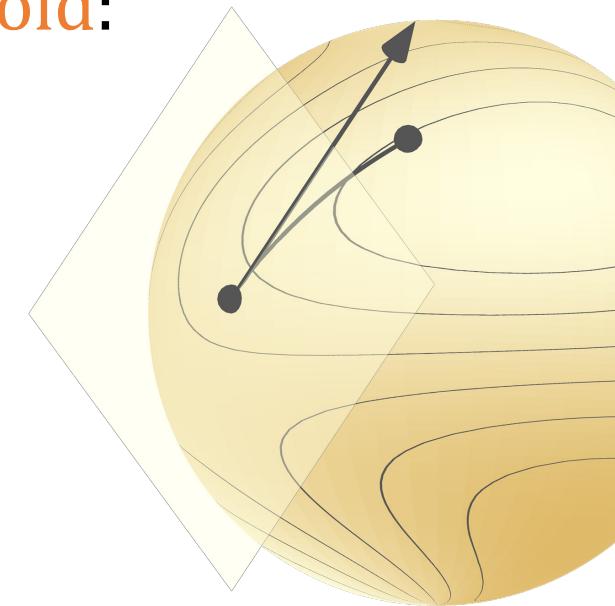
We can build **algorithms** with them: gradient descent, Newton's...

$$\langle \text{grad}f(x), v \rangle = Df(x)[v] = \lim_{t \rightarrow 0} \frac{f(x + tv) - f(x)}{t}$$
$$\text{Hess}f(x)[v] = D(\text{grad}f)(x)[v] = \lim_{t \rightarrow 0} \frac{\text{grad}f(x + tv) - \text{grad}f(x)}{t}$$

# This tutorial: optimization on manifolds

We target applications where  $S = \mathcal{M}$  is a **smooth manifold**:

$$\min_{x \in \mathcal{M}} f(x)$$



We can *choose* to turn  $\mathcal{M}$  into a **Riemannian manifold**.

If  $f$  is differentiable, we have a **Riemannian gradient** and **Hessian**.

We can build **algorithms** with them: gradient descent, Newton's...

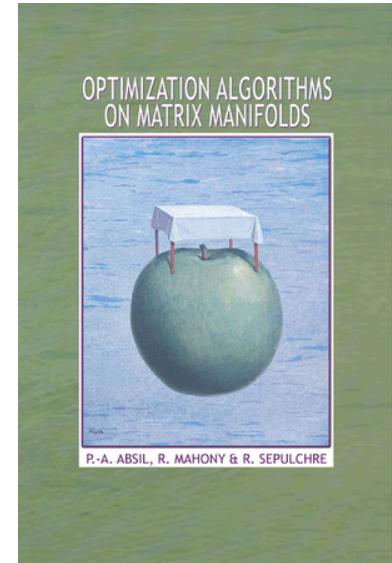
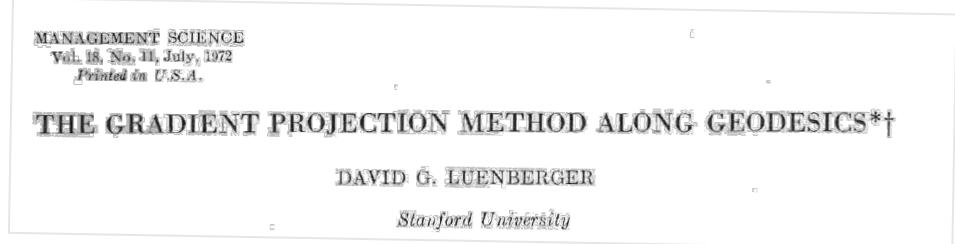
# Fifty years

Proposed by Luenberger in 1972.

Practical since the 1990s  
with numerical linear algebra.

Popularized in the 2010s  
by Absil, Mahony & Sepulchre's book.

Becoming mainstream now.



Communications and Control Engineering  
Series Editor: Alberto Isidori · Jan H. van Schuppen  
Eduardo D. Sontag · Manfred Thoma · Miroslav Krstic

Uwe Helmke · John B. Moore  
R. Brockett Editors

# Optimization and Dynamical Systems

1994

Springer Series in the Data Sciences

Nickolay Trendafilov  
Michele Gallo

# Multivariate Data Analysis on Matrix Manifolds

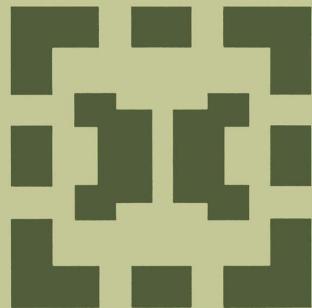
(with Manopt)

Springer  
2021

Mathematics and Its Applications

Constantin Udriște

Convex Functions and  
Optimization Methods on  
Riemannian Manifolds



Springer-Science+Business Media B.V.

1994

SPRINGER BRIEFS IN ELECTRICAL AND COMPUTER  
ENGINEERING · CONTROL, AUTOMATION AND ROBOTICS

Hiroyuki Sato

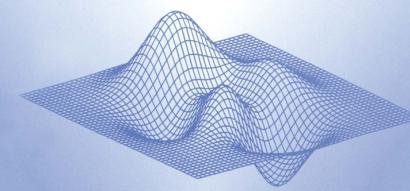
# Riemannian Optimization and Its Applications

Springer  
2021

NONCONVEX OPTIMIZATION AND ITS APPLICATIONS

Smooth Nonlinear  
Optimization in  $R^n$

Tamas Rapcsák



1997

Studies in Computational Intelligence 1046

Robert Simon Fong  
Peter Tino

# Population-Based Optimization on Riemannian Manifolds

Springer  
2022

OPTIMIZATION ALGORITHMS  
ON MATRIX MANIFOLDS

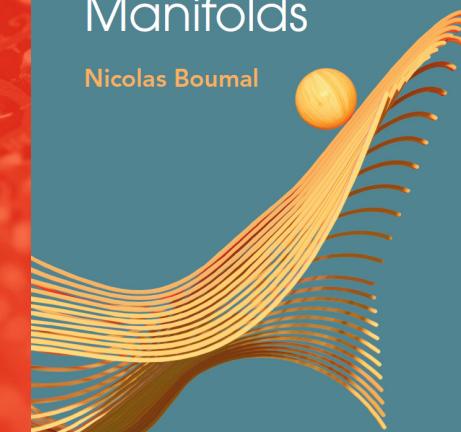


P.-A. ABSIL, R. MAHONY & R. SEPULCHRE

2008

AN INTRODUCTION TO  
Optimization  
on Smooth  
Manifolds

Nicolas Boumal



2023

# Software, book, lectures, slides

**Manopt** software packages

[manopt.org](http://manopt.org)

Matlab with Bamdev Mishra, P.-A. Absil, R. Sepulchre++

Julia by Ronny Bergmann++

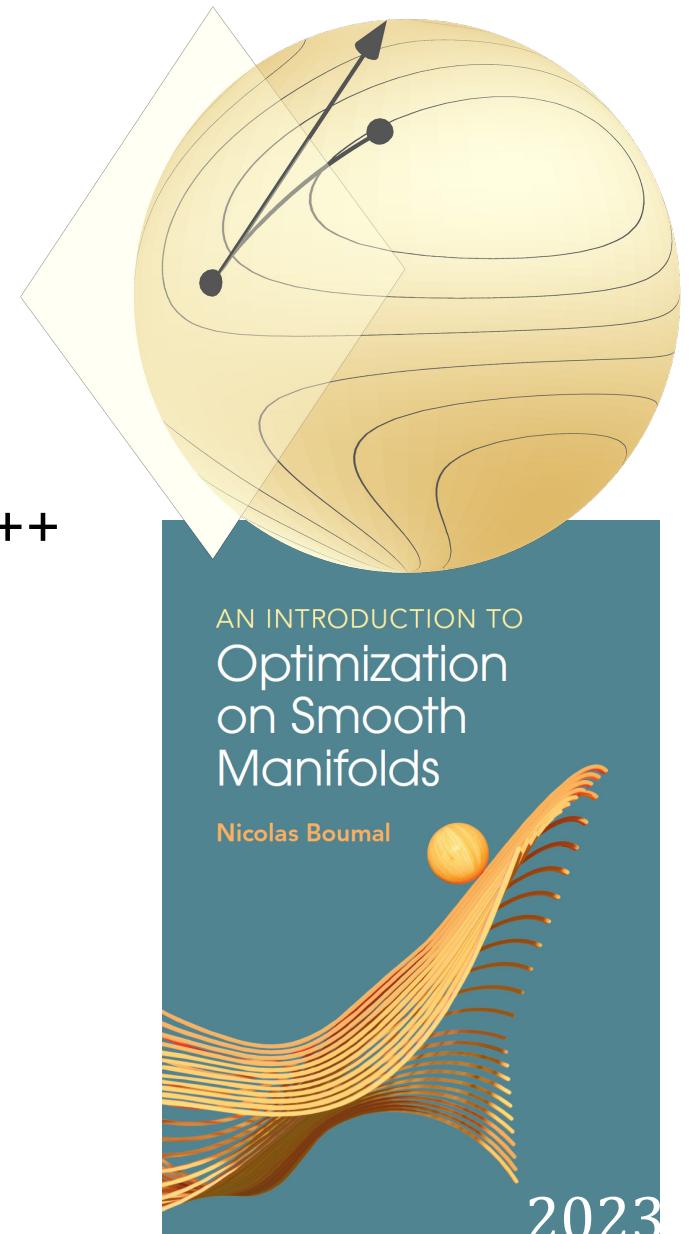
Python by James Townsend, Niklas Koep

and Sebastian Weichwald++

Book (pdf, lecture material, **videos**) and these **slides**

[nicolasboumal.net/book](http://nicolasboumal.net/book)

[nicolasboumal.net/SIAMOP23](http://nicolasboumal.net/SIAMOP23)



Many thanks to Cambridge University Press, who agreed for me to keep the preprint freely available online.

# How do manifolds arise in optimization?

Linear spaces

Symmetry

Orthonormality

Lifts/parameterizations

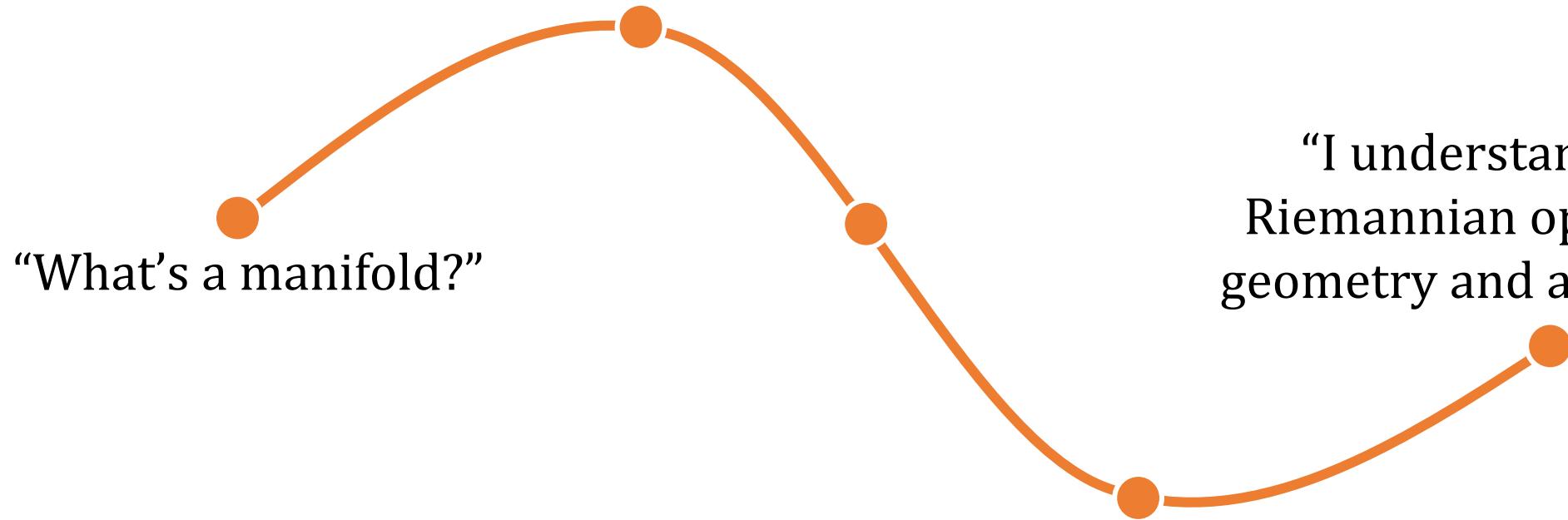
[arXiv:2207.03512](https://arxiv.org/abs/2207.03512), with Eitan Levin & Joe Kileel

Positivity

Rank

Products

# The goal for this tutorial



Main effort: building differential geometry in  $\sim 2$  hours.

Think of it as a technically precise bird's-eye view, focused on intuition.

# What do we need?

$$\min_x f(x)$$

Euclidean optimization

Riemannian optimization

Basic step:

$$x_{k+1} = x_k + \textcolor{brown}{s}_k$$

$$x_{k+1} = R_{x_k}(\textcolor{brown}{s}_k)$$

Gradient descent:

$$\textcolor{brown}{s}_k = -\alpha_k \text{grad}f(x_k)$$

same, with Riemannian gradient

Newton's method:

$$\text{Hess}f(x_k)[\textcolor{brown}{s}_k] = -\text{grad}f(x_k)$$

and Riemannian Hessian.

(Fancier algorithms involve more substantial differences, especially in analysis.)

$\text{Hess}f$

Today, we build the following tools, from the ground up.

Connections

$$\nabla, \frac{D}{dt}$$

$\text{grad}f$

Riemannian metric  $\langle u, v \rangle_x$

Vector fields

Retractions

$DF(x)[v]$

Tangent bundle  $T\mathcal{M}$

What is a smooth function?

What is a tangent vector?

What is a smooth set?

Focus on embedded submanifolds of linear spaces.

# What is a manifold? Take zero: words

Let  $\mathcal{E}$  be a linear space (say,  $\mathcal{E} = \mathbf{R}^d$ ).

A subset  $\mathcal{M}$  of that linear space is a smooth manifold if,

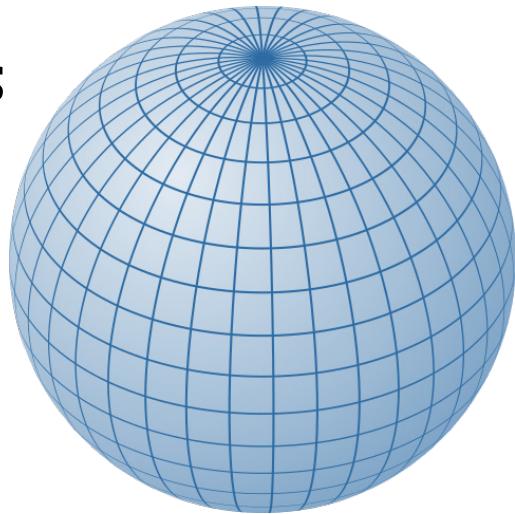
for each point  $x \in \mathcal{M}$ ,

if we zoom very close,

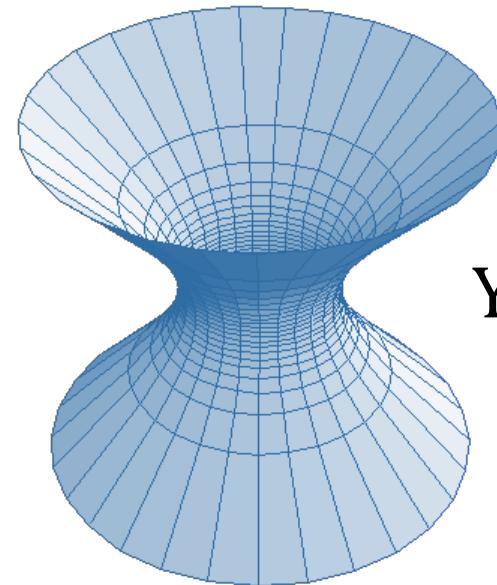
it's hard to tell whether  $\mathcal{M}$  is linear.

# What is a manifold? Take one: pictures

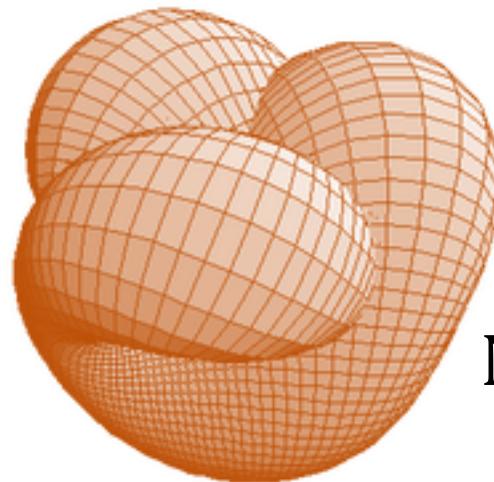
Yes



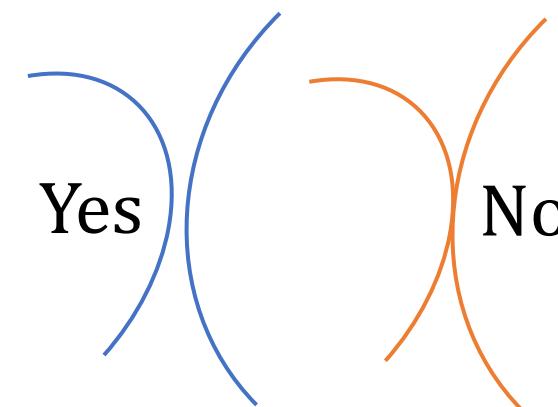
Yes



No



Yes



No

# What is a manifold? Take two: examples

Linear spaces:

$$\mathbf{R}^n, \mathbf{R}^{m \times n}, \dots$$

Stiefel manifold:

$$\{X \in \mathbf{R}^{n \times p}: X^\top X = I_p\}$$

Rotation group:

$$\{X \in \mathbf{R}^{n \times n}: X^\top X = I_n \text{ and } \det(X) = +1\}$$

Fixed-rank matrices:

$$\{X \in \mathbf{R}^{m \times n}: \text{rank}(X) = r\}$$

Grassmann manifold:

$$\{X \in \mathbf{R}^{n \times n}: X = X^\top, X^2 = X, \text{Tr}(X) = p\}$$

Positive definite cone:

$$\{X \in \mathbf{R}^{n \times n}: X = X^\top \text{ and } X > 0\}$$

Hyperbolic space:

$$\{x \in \mathbf{R}^{n+1}: x_0^2 = 1 + x_1^2 + \dots + x_n^2 \text{ and } x_0 > 0\}$$

...

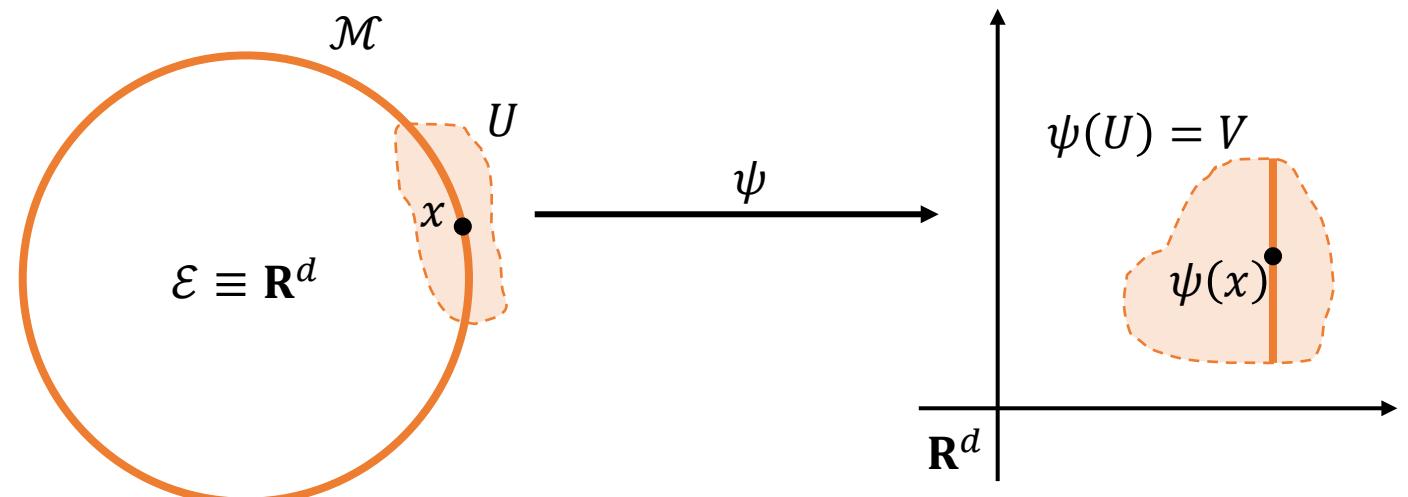
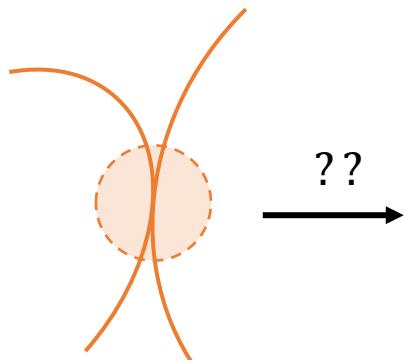
And products: if  $\mathcal{M}_1, \mathcal{M}_2$  are manifolds, then  $\mathcal{M}_1 \times \mathcal{M}_2$  is too.

# What is a manifold? Take three: math

A subset  $\mathcal{M}$  of a linear space  $\mathcal{E}$  of dimension  $\dim \mathcal{E} = d$  is a **smooth embedded submanifold** of dimension  $\dim \mathcal{M} = n$  if:

For all  $x \in \mathcal{M}$ , there exists a neighborhood  $U$  of  $x$  in  $\mathcal{E}$ , an open set  $V \subseteq \mathbf{R}^d$  and a **diffeomorphism**  $\psi: U \rightarrow V$  such that  $\psi(U \cap \mathcal{M}) = V \cap E$  where  $E$  is a linear subspace of dimension  $n$ .

We call  $\mathcal{E}$  the **embedding space**.



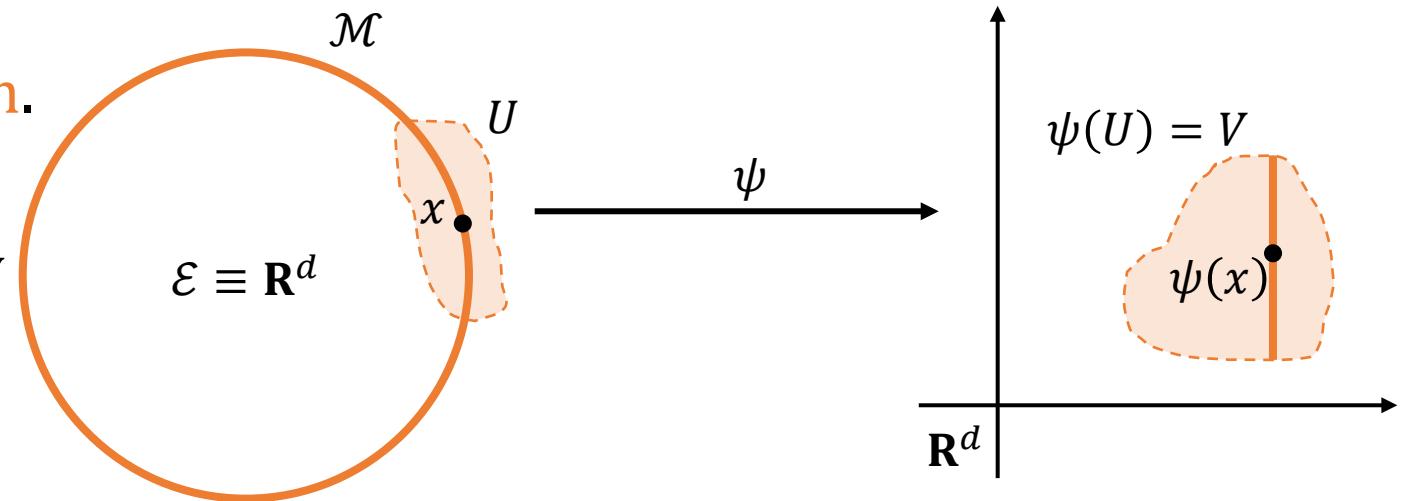
# What is a manifold? Take four: math (bis)

A subset  $\mathcal{M}$  of a linear space  $\mathcal{E}$  of dimension  $\dim \mathcal{E} = d$  is a **smooth embedded submanifold** of dimension  $\dim \mathcal{M} = n$  if:

For all  $x \in \mathcal{M}$ , there exists a neighborhood  $U$  of  $x$  in  $\mathcal{E}$  and a smooth function  $h: U \rightarrow \mathbf{R}^{d-n}$  such that  $\mathcal{M} \cap U = \{y \in U : h(y) = 0\}$  and  $Dh(x)$  has full rank.

We call  $h$  a **local defining function**.

In words:  $\mathcal{M}$  is locally defined by smooth, independent equations.



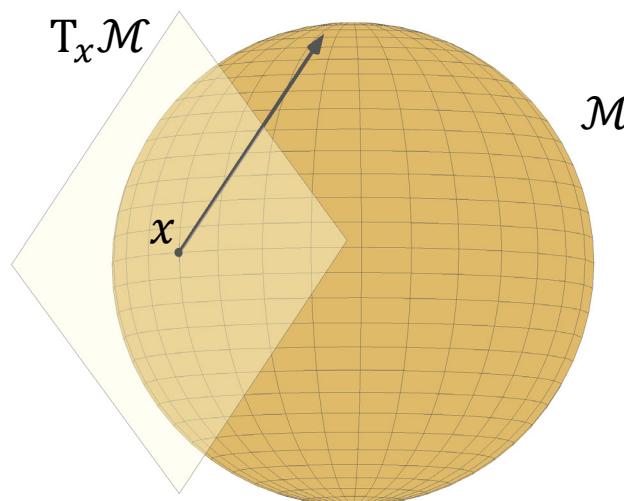
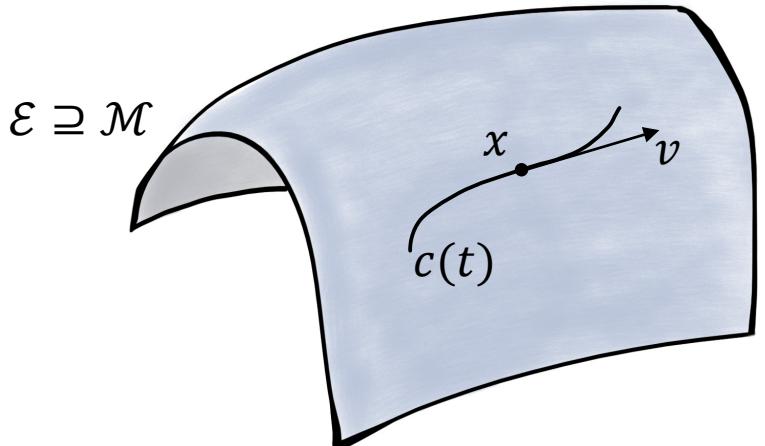
# Tangent vectors of $\mathcal{M}$ embedded in $\mathcal{E}$

A **tangent vector** at  $x$  is the velocity  $c'(0) = \lim_{t \rightarrow 0} \frac{c(t) - c(0)}{t}$  of a curve  $c: \mathbf{R} \rightarrow \mathcal{M}$  with  $c(0) = x$ .

The **tangent space**  $T_x \mathcal{M}$  is the set of all tangent vectors of  $\mathcal{M}$  at  $x$ .

It is a linear subspace of  $\mathcal{E}$  of the same dimension as  $\mathcal{M}$ .

If  $\mathcal{M} = \{x: h(x) = 0\}$  with  $h: \mathcal{E} \rightarrow \mathbf{R}^k$  smooth and  $\text{rank } Dh(x) = k$ , then  $T_x \mathcal{M} = \ker Dh(x)$ .



$$h(x) = x^T x - 1 = 0$$
$$\ker Dh(x) = \{v: x^T v = 0\}$$

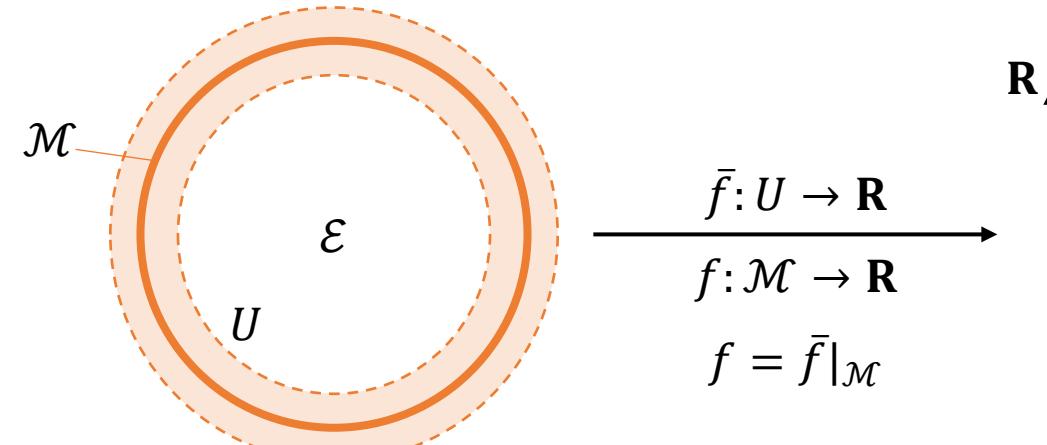
# Smooth maps on/to manifolds

Let  $\mathcal{M}, \mathcal{M}'$  be (smooth, embedded) submanifolds of linear spaces  $\mathcal{E}, \mathcal{E}'$ .

A map  $F: \mathcal{M} \rightarrow \mathcal{M}'$  is **smooth** if it has a **smooth extension**, i.e., if there exists a neighborhood  $U$  of  $\mathcal{M}$  in  $\mathcal{E}$  and a smooth map  $\bar{F}: U \rightarrow \mathcal{E}'$  such that  $F = \bar{F}|_{\mathcal{M}}$ .

Example: a **cost function**  $f: \mathcal{M} \rightarrow \mathbf{R}$  is smooth if it is the restriction of a smooth  $\bar{f}: U \rightarrow \mathbf{R}$ .

**Composition** preserves smoothness.



# Differential of a smooth map $F: \mathcal{M} \rightarrow \mathcal{M}'$

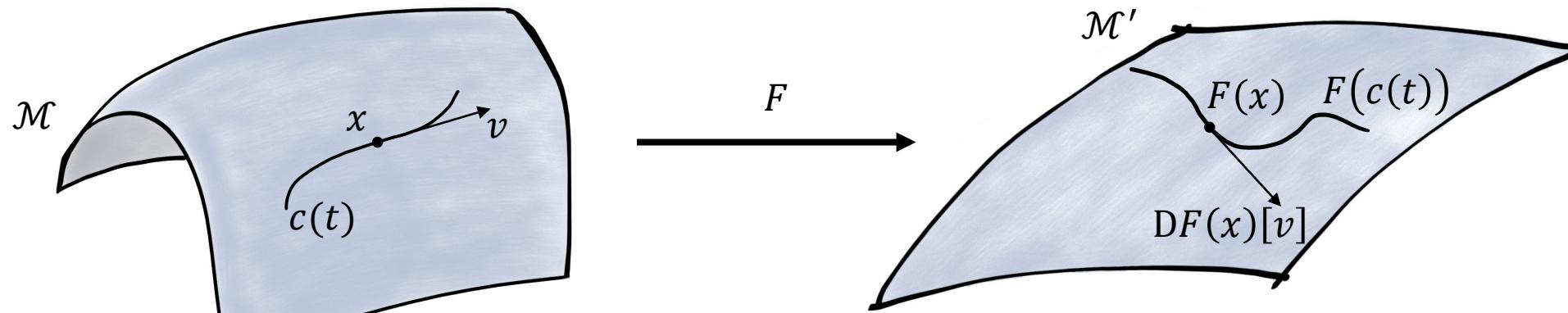
The **differential of  $F$  at  $x$**  is the map  $DF(x): T_x \mathcal{M} \rightarrow T_{F(x)} \mathcal{M}'$  defined by:

$$DF(x)[v] = (F \circ c)'(0) = \lim_{t \rightarrow 0} \frac{F(c(t)) - F(x)}{t}$$

where  $c: \mathbf{R} \rightarrow \mathcal{M}$  satisfies  $c(0) = x$  and  $c'(0) = v$ .

Claim:  $DF(x)$  is **well defined** and **linear**, and we have a **chain rule**.

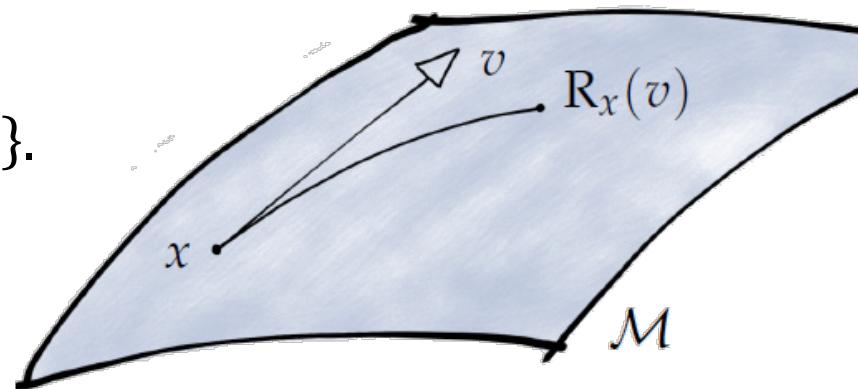
If  $\bar{F}$  is a smooth extension of  $F$ , then  $DF(x) = D\bar{F}(x)|_{T_x \mathcal{M}}$ .



# Retractions: moving around on $\mathcal{M}$

The **tangent bundle** is the set

$$T\mathcal{M} = \{(x, v) : x \in \mathcal{M} \text{ and } v \in T_x \mathcal{M}\}.$$



A **retraction** is a map  $R: T\mathcal{M} \rightarrow \mathcal{M}: (x, v) \mapsto R_x(v)$  such that each curve

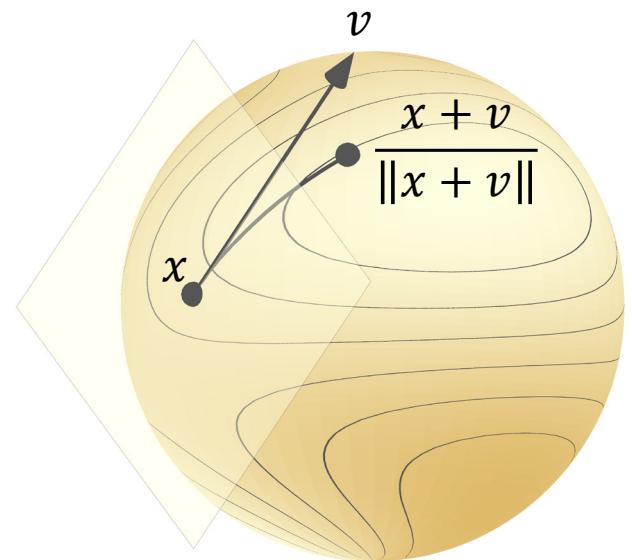
$$c(t) = R_x(tv)$$

satisfies  $c(0) = x$  and  $c'(0) = v$ .

E.g., **metric projection**:  $R_x(v)$  is the projection of  $x + v$  to  $\mathcal{M}$ .

$$\mathcal{M} = \mathbf{R}^n: R_x(v) = x + v; \quad \mathcal{M} = \{x: \|x\| = 1\}: R_x(v) = \frac{x+v}{\|x+v\|};$$

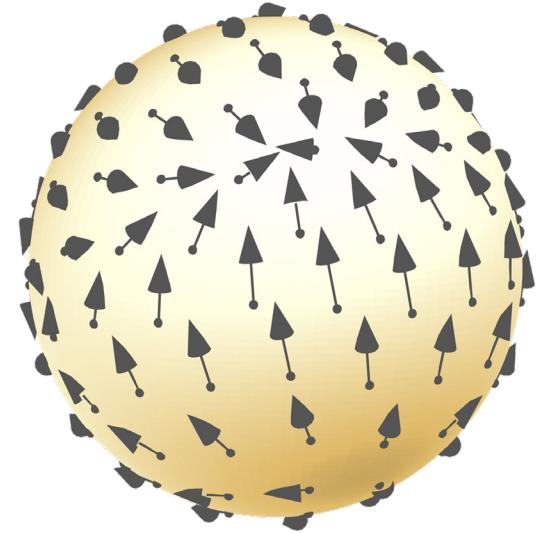
$$\mathcal{M} = \{X: \text{rank}(X) = r\}: R_X(V) = \text{SVD}_r(X + V).$$



# Riemannian manifolds

Each tangent space  $T_x \mathcal{M}$  is a linear space.

Endow each one with an inner product:  $\langle u, v \rangle_x$  for  $u, v \in T_x \mathcal{M}$ .



A **vector field** is a map  $V: \mathcal{M} \rightarrow T\mathcal{M}$  such that  $V(x)$  is tangent at  $x$  for all  $x$ .

We say **the inner products  $\langle \cdot, \cdot \rangle_x$  vary smoothly** with  $x$  if  $x \mapsto \langle U(x), V(x) \rangle_x$  is smooth for all smooth vector fields  $U, V$ .

If the inner products vary smoothly with  $x$ , they form a **Riemannian metric**.

A **Riemannian manifold** is a smooth manifold with a Riemannian metric.

# Riemannian structure and optimization

A **Riemannian manifold** is a smooth manifold with a smoothly varying choice of inner product on each tangent space.

A manifold can be endowed with **many** different Riemannian structures.

A problem  $\min_{x \in \mathcal{M}} f(x)$  is defined independently of any Riemannian structure.

We *choose* a metric for algorithmic purposes. Akin to **preconditioning**.

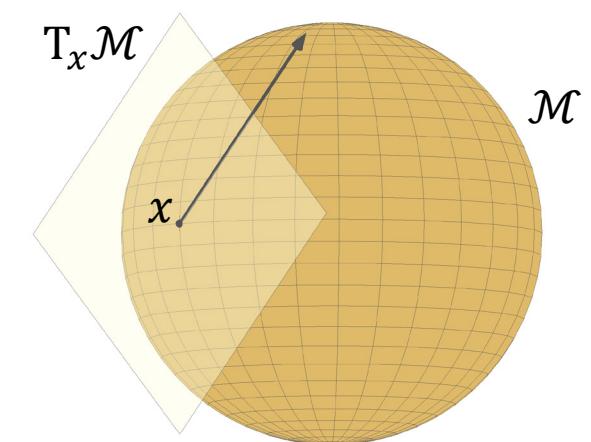
# Riemannian submanifolds

Let the **embedding space** of  $\mathcal{M}$  be a **Euclidean space**  $\mathcal{E}$  with metric  $\langle \cdot, \cdot \rangle$ .

For example:  $\mathcal{E} = \mathbf{R}^d$  and  $\langle u, v \rangle = u^\top v$  for all  $u, v \in \mathbf{R}^d$ .

A **convenient choice of Riemannian structure** for  $\mathcal{M}$  is to let:

$$\langle u, v \rangle_x = \langle u, v \rangle.$$



This is well defined because  $u, v \in T_x \mathcal{M}$  are, in particular, elements of  $\mathcal{E}$ .

This is a Riemannian metric. With it,  $\mathcal{M}$  is a **Riemannian submanifold** of  $\mathcal{E}$ .

$$\langle \text{grad}\bar{f}(x), v \rangle = D\bar{f}(x)[v] = \lim_{t \rightarrow 0} \frac{\bar{f}(x + tv) - \bar{f}(x)}{t}$$

# Riemannian gradients

(Reminders for  $\bar{f}: \mathbf{R}^d \rightarrow \mathbf{R}$ .)

The **Riemannian gradient** of a smooth  $f: \mathcal{M} \rightarrow \mathbf{R}$  is the vector field  $\text{grad}f$  defined by:

$$\forall (x, v) \in T\mathcal{M}, \quad \langle \text{grad}f(x), v \rangle_x = Df(x)[v].$$

Claim:  $\text{grad}f$  is a well-defined smooth vector field.

If  $\mathcal{M}$  is a Riemannian **submanifold** of a Euclidean space  $\mathcal{E}$ , then

$$\text{grad}f(x) = \text{Proj}_x(\text{grad}\bar{f}(x)),$$

where  $\text{Proj}_x$  is the orthogonal projector from  $\mathcal{E}$  to  $T_x\mathcal{M}$  and  $\bar{f}$  is a **smooth extension** of  $f$ .

# We're all set for gradient descent

$$x_{k+1} = R_{x_k}(-\alpha_k \text{grad}f(x_k))$$

How does  $f(x_{k+1})$  compare to  $f(x_k)$ ?

Consider a **Taylor expansion** of the **pullback**  $f \circ R_x: T_x \mathcal{M} \rightarrow \mathbf{R}$ :

$$f(R_x(\textcolor{blue}{s})) = f(x) + \langle \text{grad}f(x), \textcolor{blue}{s} \rangle_x + O(\|\textcolor{blue}{s}\|_x^2)$$

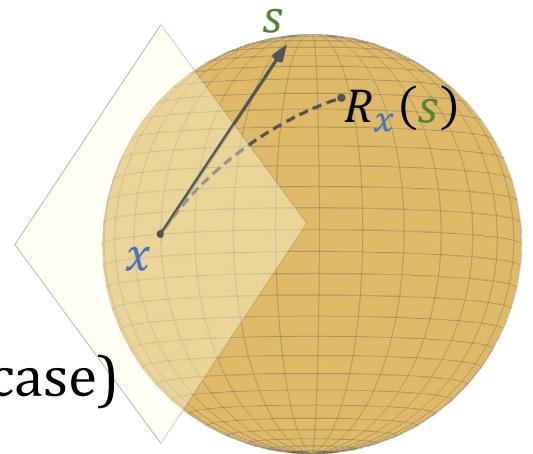
# Gradient descent on $\mathcal{M}$

**A1**  $f(\mathbf{x}) \geq f_{\text{low}}$  for all  $\mathbf{x} \in \mathcal{M}$

**A2**  $f(R_{\mathbf{x}}(\mathbf{s})) \leq f(\mathbf{x}) + \langle \mathbf{s}, \text{grad}f(\mathbf{x}) \rangle_{\mathbf{x}} + \frac{L}{2} \|\mathbf{s}\|_{\mathbf{x}}^2$

Algorithm:  $\mathbf{x}_{k+1} = R_{\mathbf{x}_k} \left( -\frac{1}{L} \text{grad}f(\mathbf{x}_k) \right)$

Complexity:  $\left[ \min_{k < K} \|\text{grad}f(\mathbf{x}_k)\|_{\mathbf{x}_k} \right] \leq \sqrt{\frac{2L(f(x_0) - f_{\text{low}})}{K}}$  (same as Euclidean case)



$$\mathbf{A2} \Rightarrow f(\mathbf{x}_{k+1}) \leq f(\mathbf{x}_k) - \frac{1}{L} \|\text{grad}f(\mathbf{x}_k)\|_{\mathbf{x}_k}^2 + \frac{1}{2L} \|\text{grad}f(\mathbf{x}_k)\|_{\mathbf{x}_k}^2$$

$$\Rightarrow f(\mathbf{x}_k) - f(\mathbf{x}_{k+1}) \geq \frac{1}{2L} \|\text{grad}f(\mathbf{x}_k)\|_{\mathbf{x}_k}^2$$

$$\mathbf{A1} \Rightarrow f(x_0) - f_{\text{low}} \geq f(x_0) - f(x_K) = \sum_{k=0}^{K-1} f(\mathbf{x}_k) - f(\mathbf{x}_{k+1}) \geq \frac{K}{2L} \min_{k < K} \|\text{grad}f(\mathbf{x}_k)\|_{\mathbf{x}_k}^2$$

# Riemannian Hessians

$$\langle \text{grad}\bar{f}(x), v \rangle = D\bar{f}(x)[v] = \lim_{t \rightarrow 0} \frac{\bar{f}(x + tv) - \bar{f}(x)}{t}$$

$$\text{Hess}\bar{f}(x)[v] = D(\text{grad}\bar{f})(x)[v] = \lim_{t \rightarrow 0} \frac{\text{grad}\bar{f}(x + tv) - \text{grad}\bar{f}(x)}{t}$$

(Reminders for  $\bar{f}: \mathbf{R}^d \rightarrow \mathbf{R}$ .)

The **Riemannian Hessian** of  $f$  at  $x$  should be a **symmetric linear map** describing gradient change:  $\text{Hess}f(x): T_x\mathcal{M} \rightarrow T_x\mathcal{M}$ .

Since  $\text{grad}f: \mathcal{M} \rightarrow T\mathcal{M}$  is a smooth map, a natural first attempt is:

$$\text{Hess}f(x)[v] \stackrel{?}{=} D\text{grad}f(x)[v].$$

However, the rhs is not always in  $T_x\mathcal{M}$ ... We need a new derivative for vector fields.

**Fundamental theorem of Riemannian geometry:**

There exists a unique way to differentiate vector fields that has “desirable properties”.

This **Riemannian connection**  $\nabla$  leads to the Riemannian Hessian

$$\text{Hess}f(x)[v] = \nabla_v \text{grad}f$$

being a symmetric map on  $T_x\mathcal{M}$ .

# Riemannian Hessians

$$\langle \text{grad}\bar{f}(x), v \rangle = D\bar{f}(x)[v] = \lim_{t \rightarrow 0} \frac{\bar{f}(x + tv) - \bar{f}(x)}{t}$$

$$\text{Hess}\bar{f}(x)[v] = D(\text{grad}\bar{f})(x)[v] = \lim_{t \rightarrow 0} \frac{\text{grad}\bar{f}(x + tv) - \text{grad}\bar{f}(x)}{t}$$

(Reminders for  $\bar{f}: \mathbf{R}^d \rightarrow \mathbf{R}$ .)

The **Riemannian Hessian** of  $f$  at  $x$  should be a **symmetric linear map** describing gradient change:  $\text{Hess}f(x): T_x\mathcal{M} \rightarrow T_x\mathcal{M}$ .

Since  $\text{grad}f: \mathcal{M} \rightarrow T\mathcal{M}$  is a smooth map, a natural first attempt is:

$$\text{Hess}f(x)[v] \stackrel{?}{=} D\text{grad}f(x)[v].$$

However, the rhs is not always in  $T_x\mathcal{M}$ ... We need a new derivative for vector fields.

If  $\mathcal{M}$  is a Riemannian **submanifold** of Euclidean space, then:

$$\begin{aligned} \text{Hess}f(x)[v] &= \text{Proj}_x(D\text{grad}f(x)[v]) \\ &= \text{Proj}_x(\text{Hess}\bar{f}(x)[v]) + W(v, \text{Proj}_x^\perp(\text{grad}\bar{f}(x))) \end{aligned}$$

where  $W$  is the Weingarten map of  $\mathcal{M}$ .

# Example: Rayleigh quotient optimization

Compute the smallest eigenvalue of a symmetric matrix  $A \in \mathbf{R}^{n \times n}$ :

$$\min_{x \in \mathcal{M}} \frac{1}{2} x^\top A x \quad \text{with} \quad \mathcal{M} = \{x \in \mathbf{R}^n : x^\top x = 1\}$$

The cost function  $f: \mathcal{M} \rightarrow \mathbf{R}$  is the restriction of the smooth function  $\bar{f}(x) = \frac{1}{2} x^\top A x$  from  $\mathbf{R}^n$  to  $\mathcal{M}$ .

Tangent spaces  $T_x \mathcal{M} = \{v \in \mathbf{R}^n : x^\top v = 0\}$ .

Make  $\mathcal{M}$  into a Riemannian submanifold of  $\mathbf{R}^n$  with  $\langle u, v \rangle = u^\top v$ .

Projection to  $T_x \mathcal{M}$ :  $\text{Proj}_x(z) = z - (x^\top z)x$ .

Gradient of  $\bar{f}$ :  $\text{grad}\bar{f}(x) = Ax$ .

Gradient of  $f$ :  $\text{grad}f(x) = \text{Proj}_x(\text{grad}\bar{f}(x)) = Ax - (x^\top Ax)x$ .

Differential of  $\text{grad}f$ :  $D\text{grad}f(x)[v] = Av - (v^\top Ax + x^\top Av)x - (x^\top Ax)v$ .

Hessian of  $f$ :  $\text{Hess}f(x)[v] = \text{Proj}_x(D\text{grad}f(x)[v]) = \text{Proj}_x(Av) - (x^\top Ax)v$ .

The following are equivalent for  $x \in \mathcal{M}$ :  $x$  is a global minimizer;  $x$  is a unit-norm eigenvector of  $A$  for the least eigenvalue;  $\text{grad}f(x) = 0$  and  $\text{Hess}f(x) \geq 0$ .

$\text{Hess}f$

Enough definitions.  
Now let's use this tower.

Connections

$$\nabla, \frac{D}{dt}$$

$\text{grad}f$

Riemannian  
metric  $\langle u, v \rangle_x$

Vector fields

Retractions

$DF(x)[v]$

Tangent  
bundle  $T\mathcal{M}$

What is  
a smooth function?

What is  
a tangent vector?

What is  
a smooth set?

Example:

Max-Cut with Manopt

# Full example: hands on with Manopt

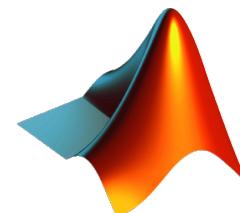
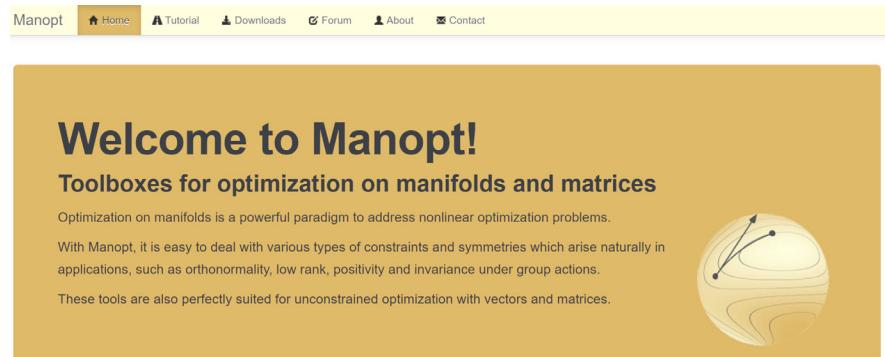
**Manopt** is a family of toolboxes for Riemannian optimization.

Go to [manopt.org](http://manopt.org) for code, a tutorial, a forum, and a list of other software.

Github: [github.com/NicolasBoumal/manopt](https://github.com/NicolasBoumal/manopt)

**Matlab** example for  $\min_{\|x\|=1} x^T A x$ :

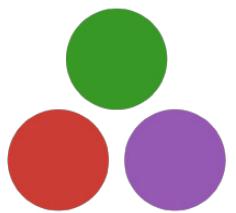
```
problem.M = spherefactory(n);  
problem.cost = @(x) x'*A*x;  
problem.egrad = @(x) 2*A*x;  
x = trustregions(problem);
```



With Bamdev Mishra,  
P.-A. Absil & R. Sepulchre



Lead by J. Townsend,  
N. Koep & S. Weichwald



Lead by Ronny Bergmann

# What's in a factory-produced manifold?

Example: stripped down and simplified `spherefactory`

```
function M = spherefactory(n)
    M.name = @(()) sprintf('Sphere S^%d', n-1);
    M.dim = @(()) n-1;
    M.inner = @(x, u, v) u'*v;
    M.norm = @(x, u) norm(u);
    M.dist = @(x, y) real(2*asin(.5*norm(x - y)));
    M.exp = @exponential;
    M.retr = @(x, u) (x+u)/norm(x+u);
    M.invretr = @inverse_retraction;
    M.log = @logarithm;
    M.hash = @(x) ['z' hashmd5(x)];
    M.rand = @(()) normalize(randn(n, 1));
)

function M = spherefactory(n)
    M.inner = @(x, u, v) u'*v;
    M.proj = @(x, u) u - x*(x'*u);
    M.egrad2rgrad = M.proj;
    M.ehess2rhess = @(x, egrad, ehess, u) ...
        M.proj(x, ehess - (x'*egrad)*u);
    M.retr = @(x, u) (x+u)/norm(x+u);
```

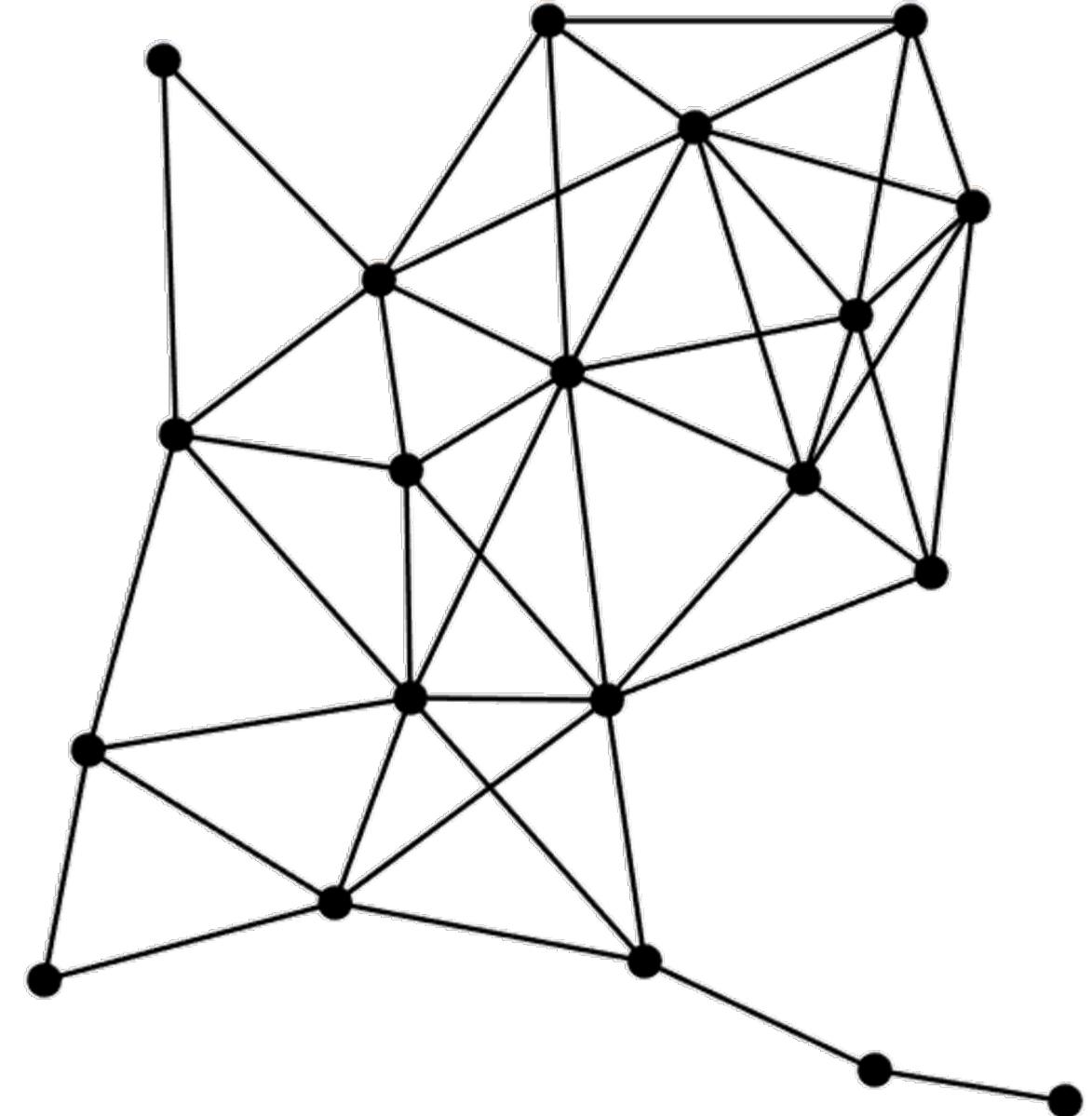
# Max-Cut

Input:

An undirected graph.

Output:

Vertex labels ( $+1$ ,  $-1$ )  
so that as many edges  
as possible connect  
different labels.



# Max-Cut

Input:

An undirected graph:  
adjacency matrix  $A$ .

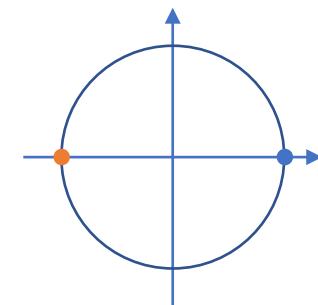
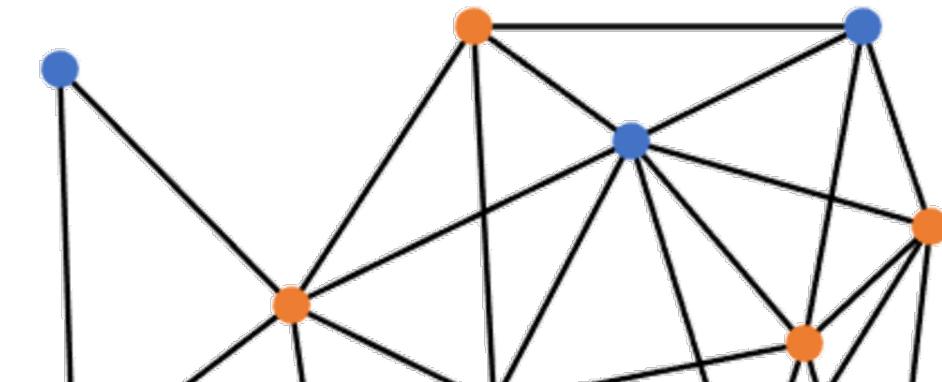
Output:

Vertex labels  $x_i \in \{+1, -1\}$   
so that as many edges  
as possible connect  
different labels.

$$\min_{x_1, \dots, x_n} \sum_{ij} a_{ij} x_i x_j \quad \text{s. t. } x_i \in \{\pm 1\}$$

Time-tested relaxation:

Let  $x_i$  be unit-norm in  $\mathbf{R}^p$ .



# Max-Cut via low-rank relaxation in Manopt

With adjacency matrix  $A \in \mathbf{R}^{n \times n}$ , want:

$$\min_{x_1, \dots, x_n \in \mathbf{R}^p} \sum_{ij} a_{ij} x_i^\top x_j \quad \text{s.t. } \|x_i\| = 1 \quad \forall i$$

The manifold is a product of  $n$  spheres:

$$\mathcal{M} = \{x \in \mathbf{R}^p : \|x\| = 1\}^n$$

$$\equiv \{X \in \mathbf{R}^{p \times n} : \|X_{:,i}\| = 1 \quad \forall i\}$$

Called the **oblique manifold**.

```
data = load('graph20.mat');
A = data.A; n = data.n;

p = 3;
problem.M = obliquefactory(p, n);
problem.cost = @(X) sum((X*A) .* X, 'all');
problem.egrad = @(X) 2*X*A;
problem.ehess = @(X, Xdot) 2*Xdot*A;

X = trustregions(problem);

s = sign(X'*randn(p, 1)); %rand round
```

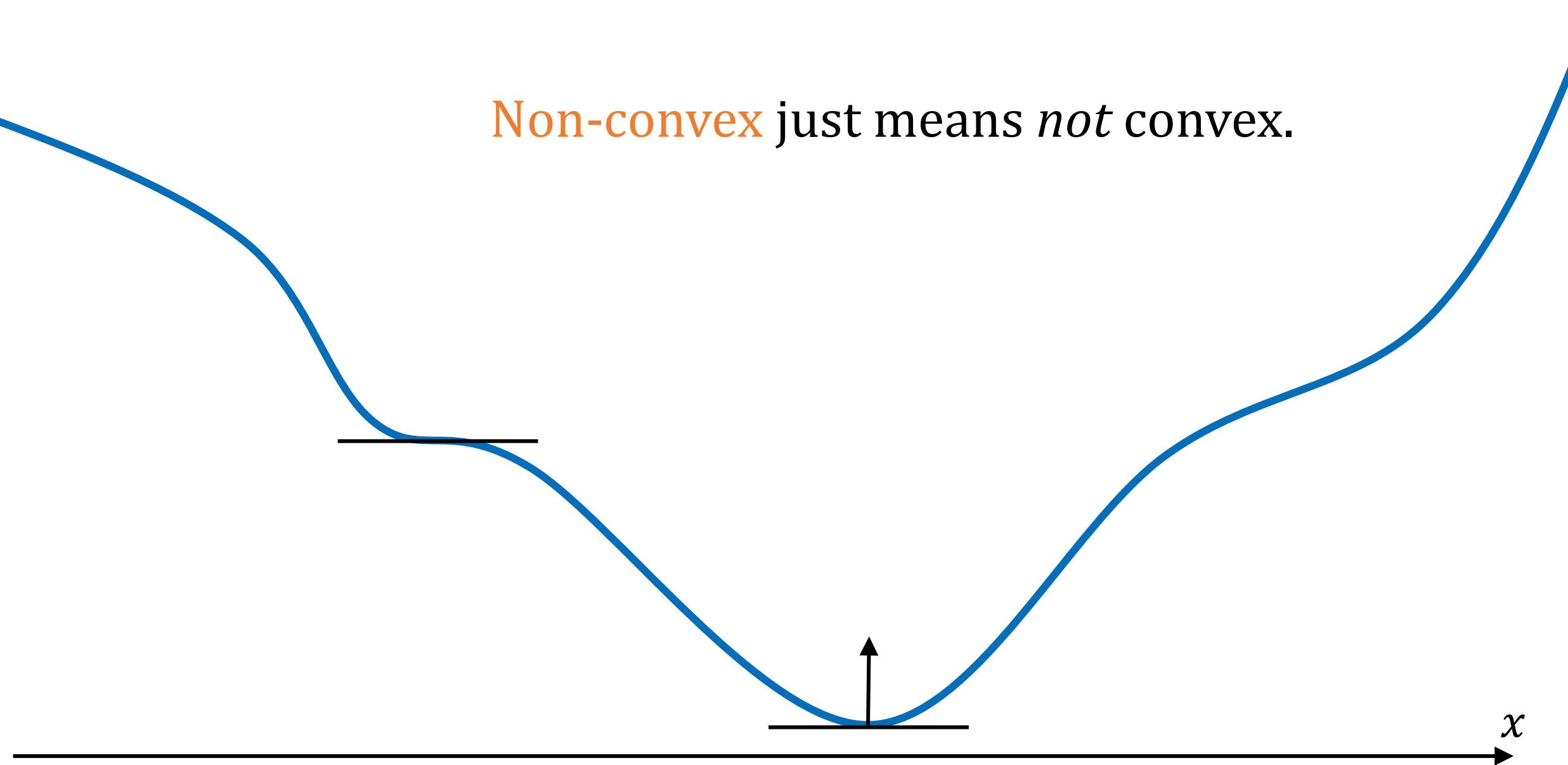
# Active research directions by many

- More algorithms: nonsmooth, stochastic, parallel, randomized, ...
- Constrained optimization on manifolds
- Applications, old and new
- Complexity (upper and lower bounds, acceleration)
- Role of curvature
- Geodesic convexity
- Solution tracking (homotopy, continuation), bilevel, min-max
- Infeasible methods (“off-the-manifold”, still using the structure)
- Broader generalizations: boundary, varieties, lift to smooth manifold, ...
- Benign **non-convexity**

*“... in fact, the great watershed in optimization isn't between linearity and nonlinearity, but **convexity** and **non-convexity**.”*

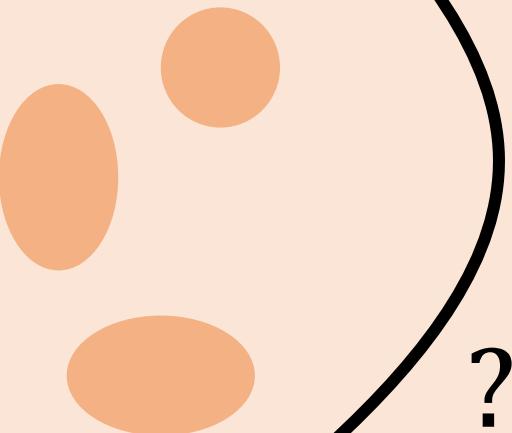
R. T. Rockafellar, in SIAM Review, 1993

Non-convex just means *not* convex.



*“... in fact, the great watershed in optimization isn't between linearity and nonlinearity, but **convexity** and **non-convexity**.”*

R. T. Rockafellar, in SIAM Review, 1993



# Pockets of benign non-convexity: Ju Sun's list

<https://sunju.org/research/nonconvex>, ~900 papers in March 2021; categories:

Matrix Completion/Sensing  
Tensor Recovery/Decomposition &  
Hidden Variable Models  
Phase Retrieval  
Dictionary Learning  
Deep Learning  
Sparse Vectors in Linear Subspaces  
Nonnegative/Sparse  
Principal Component Analysis  
Mixed Linear Regression  
Blind Deconvolution/Calibration  
Super Resolution

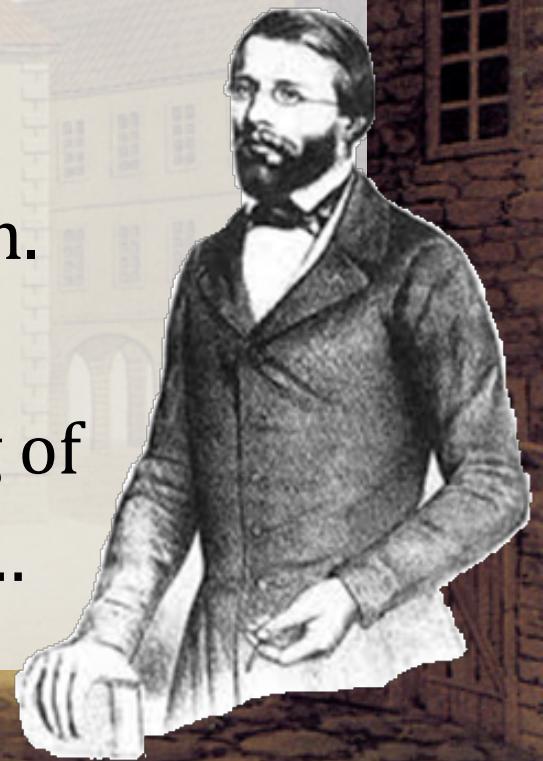
Synchronization Problems  
Community Detection  
Joint Alignment  
Numerical Linear Algebra  
Bayesian Inference  
Empirical Risk Minimization &  
Shallow Networks  
System Identification  
Burer-Monteiro Style Decomposition Algorithms  
Generic Structured Problems  
Nonconvex Feasibility Problems  
Separable Nonnegative Factorization (NMF)

# Back in Göttingen...

If Riemann didn't invent his geometry to pick Netflix movies, then why did he?

His motivation was to extend Gauss' work (his advisor), to understand **curvature** in spaces of arbitrary dimension.

Bit by bit, the community is building some understanding of the effect curvature has in optimization. To be continued...



# Software, book, lecture

**Manopt** software packages

[manopt.org](http://manopt.org)

Matlab with Bamdev Mishra, P.-A. Absil, R. Sepulchre++

Julia by Ronny Bergmann++

Python by James Townsend, Niklas Koep

and Sebastian Weichwald++

Book (pdf, lecture material, **videos**) and these **slides**

[nicolasboumal.net/book](http://nicolasboumal.net/book)

[nicolasboumal.net/SIAMOP23](http://nicolasboumal.net/SIAMOP23)

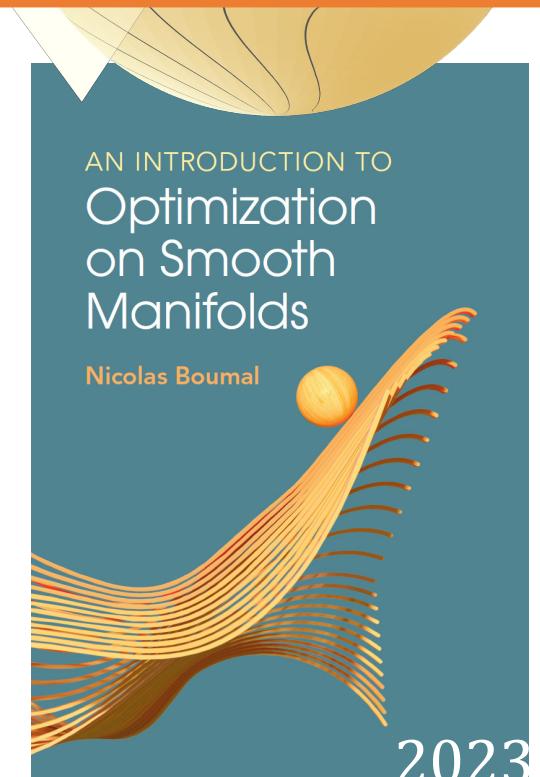
Saturday, June 3

**MS311**

**Riemannian Optimization - Part III of III**

3:15 PM - 4:45 PM

*Room: Redwood B, 2nd floor*



Many thanks to Cambridge University Press, who agreed for me to keep the preprint freely available online.

