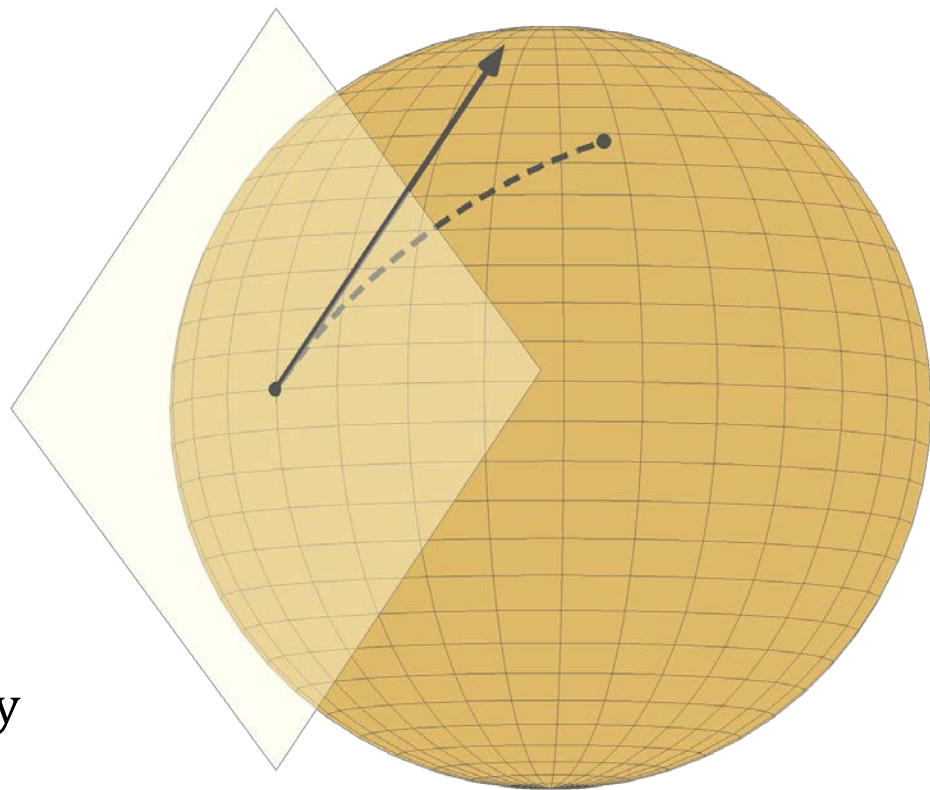


An introduction to optimization on manifolds

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



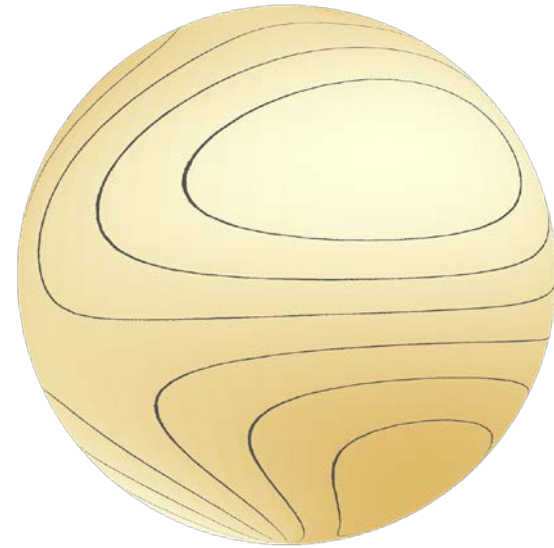
Nicolas Boumal, January 2020
Mathematics, Princeton University



Göttingen, Germany—1854

Optimization on smooth manifolds

$$\min_x f(x) \text{ subject to } x \in \mathcal{M}$$



Linear spaces

Unconstrained; linear equality constraints

Low rank (matrices, tensors)

Recommender systems, large-scale Lyapunov equations, ...

Orthonormality (Grassmann, Stiefel, rotations)

Dictionary learning, SfM, SLAM, PCA, ICA, SBM, Electr. Struct. Comp....

Positivity (positive definiteness, positive orthant)

Metric learning, Gaussian mixtures, diffusion tensor imaging, ...

Symmetry (quotient manifolds)

Invariance under group actions

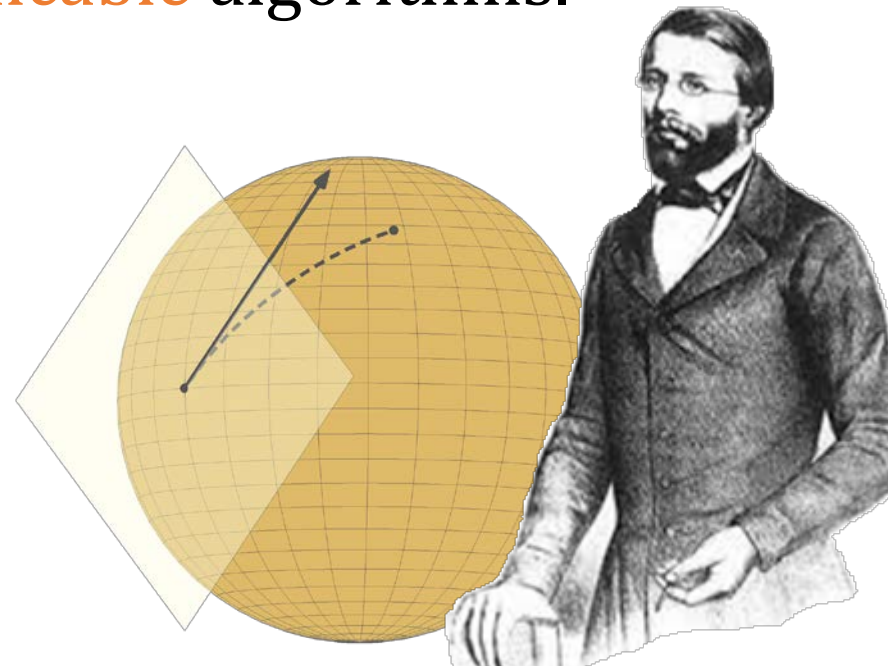
A Riemannian structure gives us gradients and Hessians

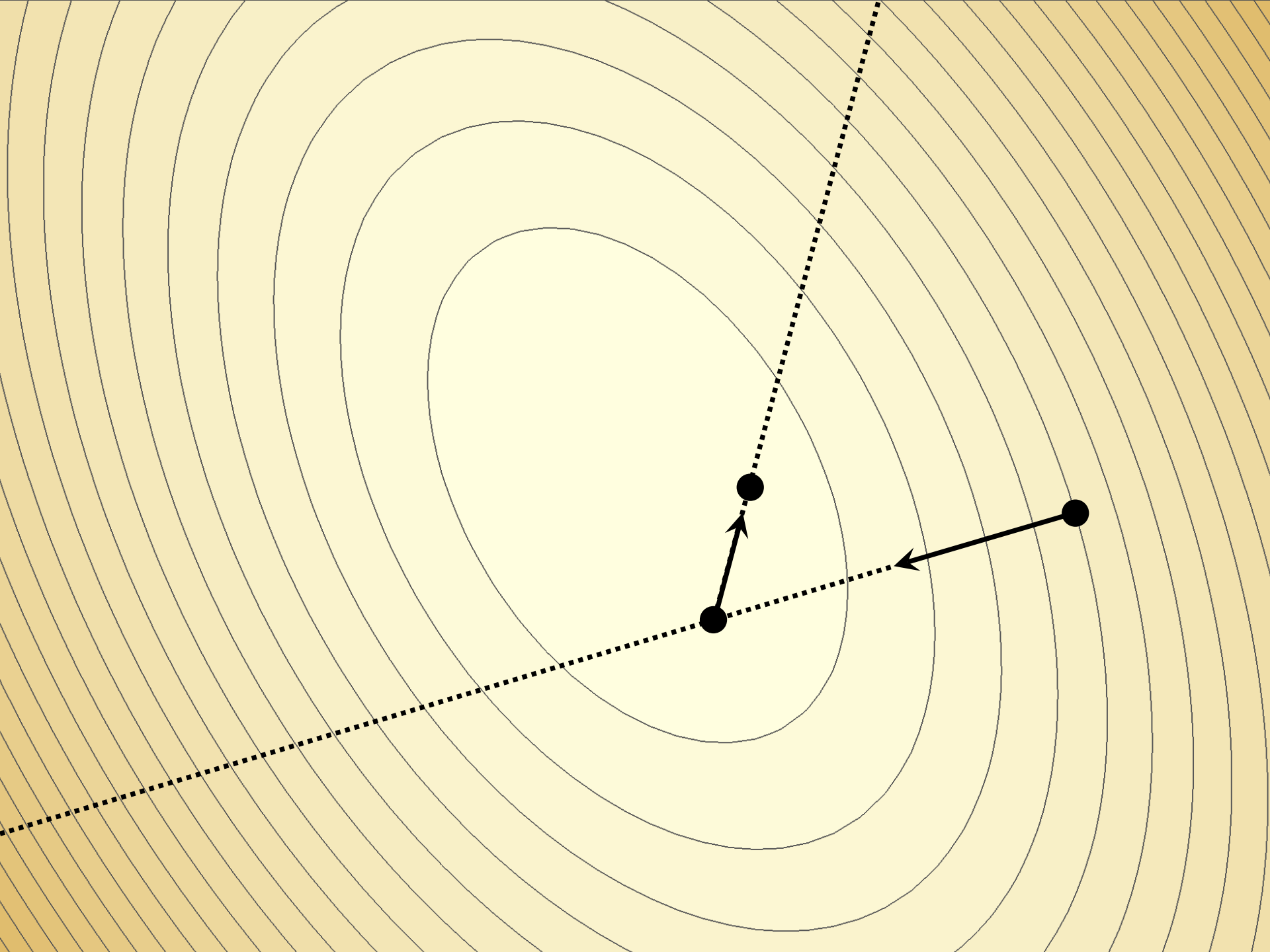
The essential tools of smooth optimization are defined generally on Riemannian manifolds.

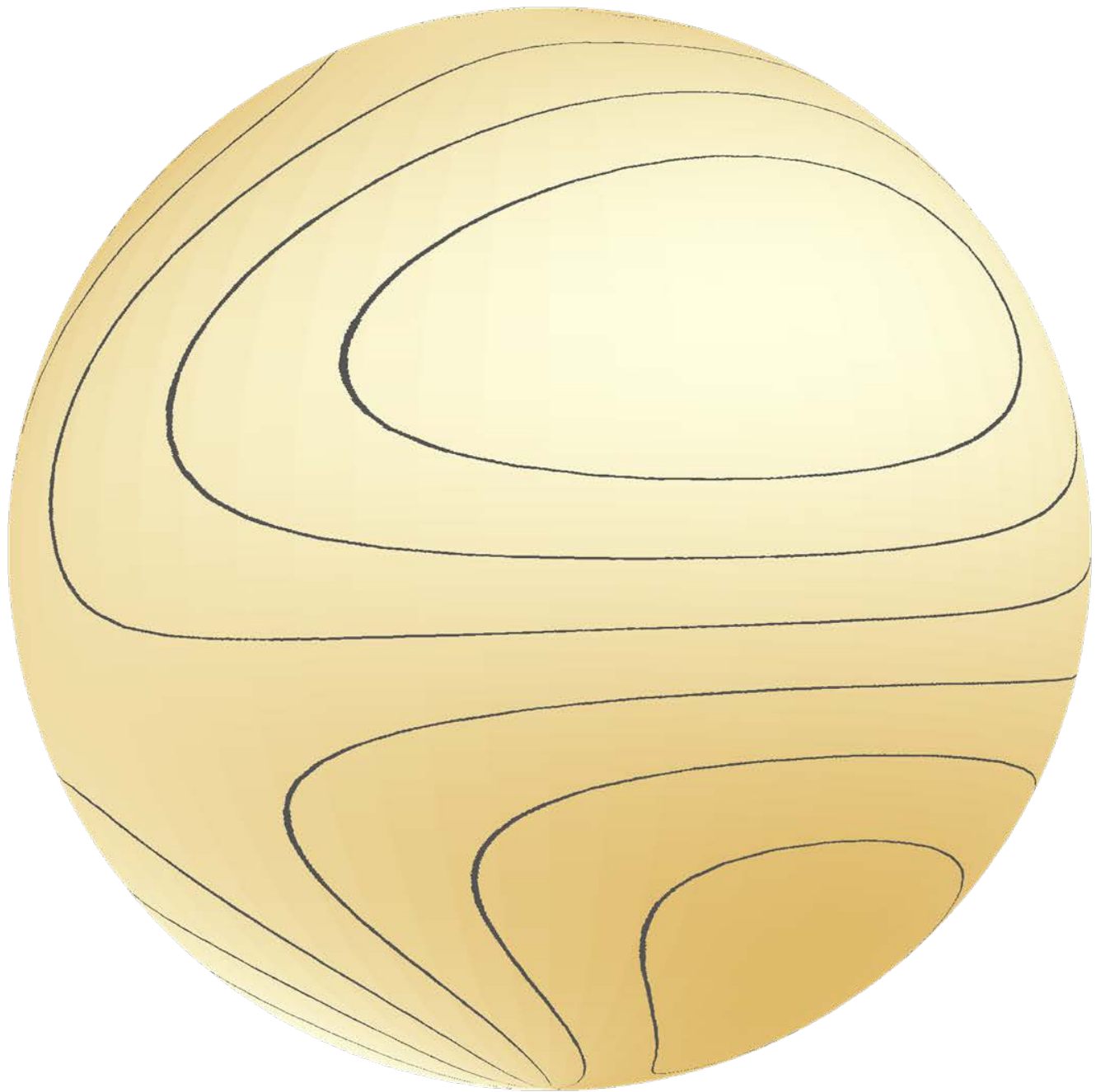
Unified theory, broadly applicable algorithms.

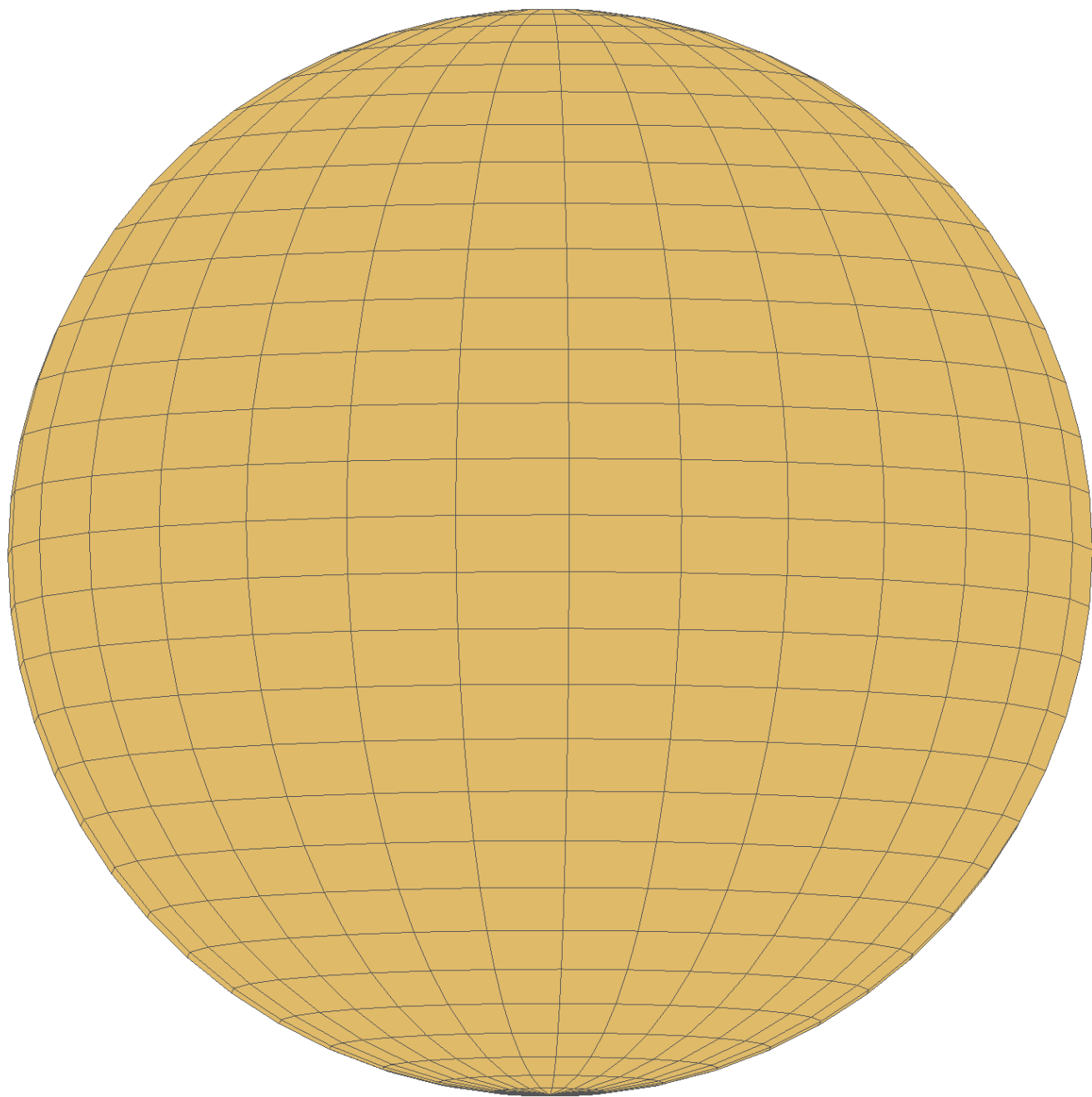
First ideas from the '70s.

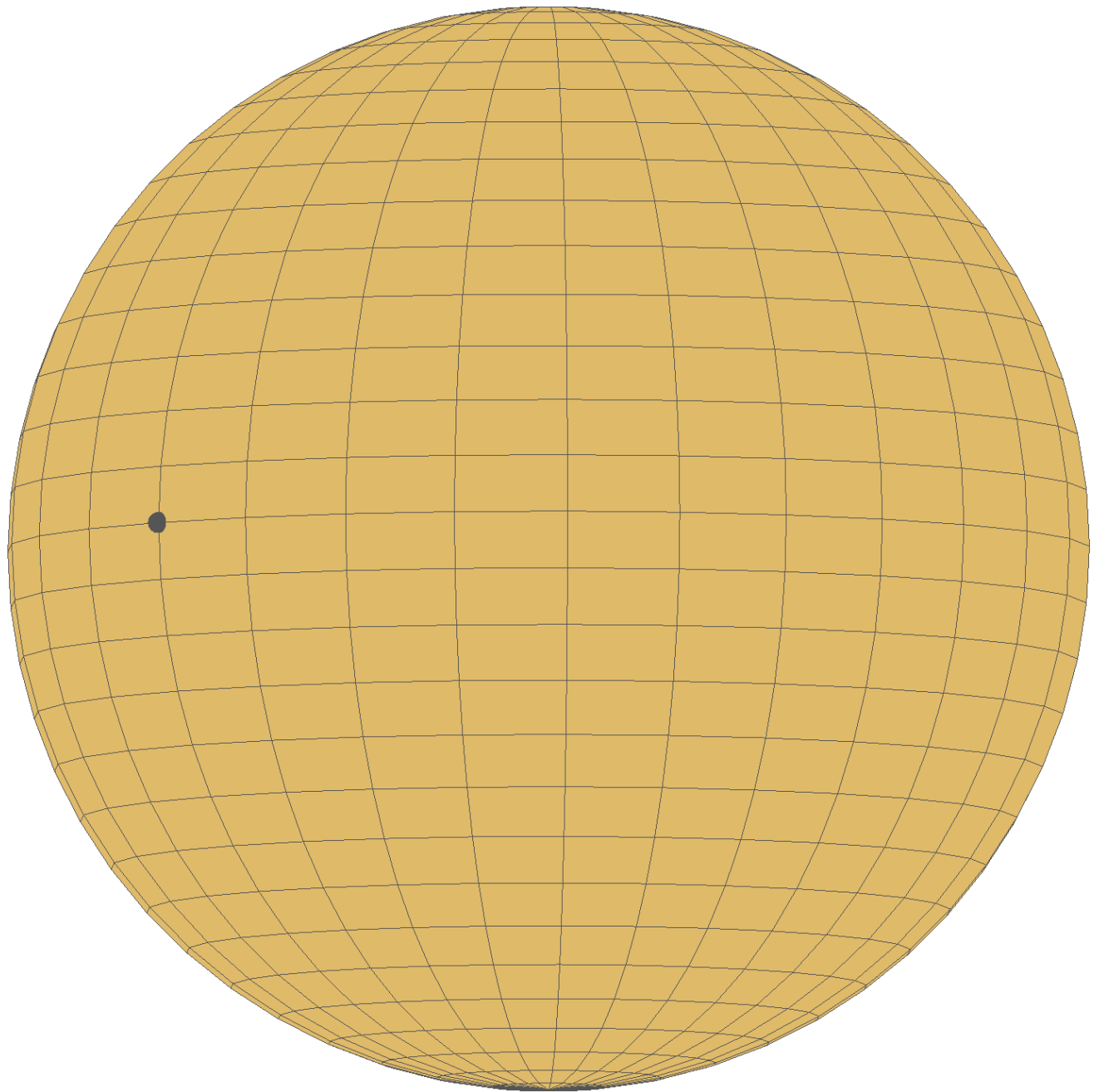
First practical in the '90s.

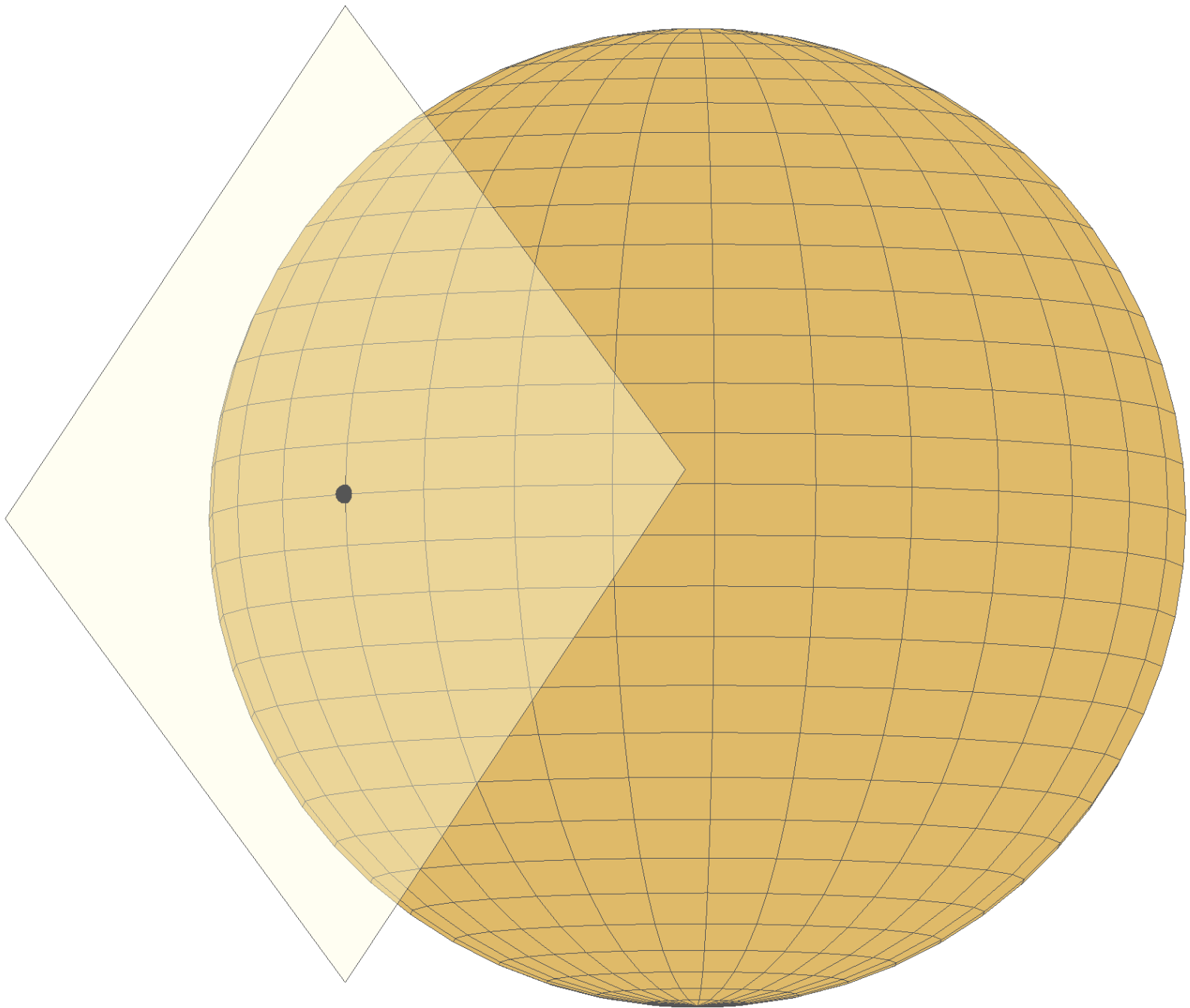


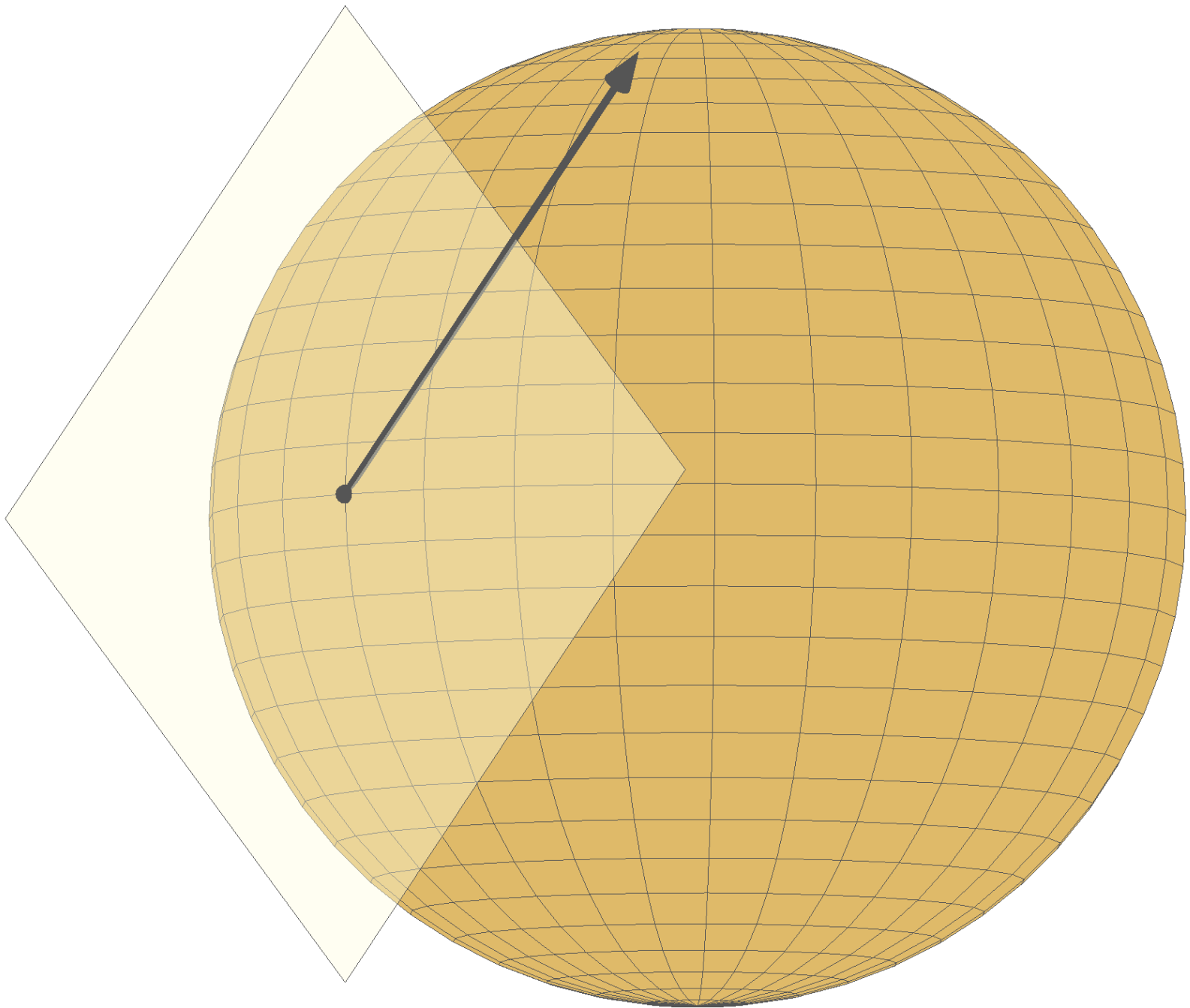


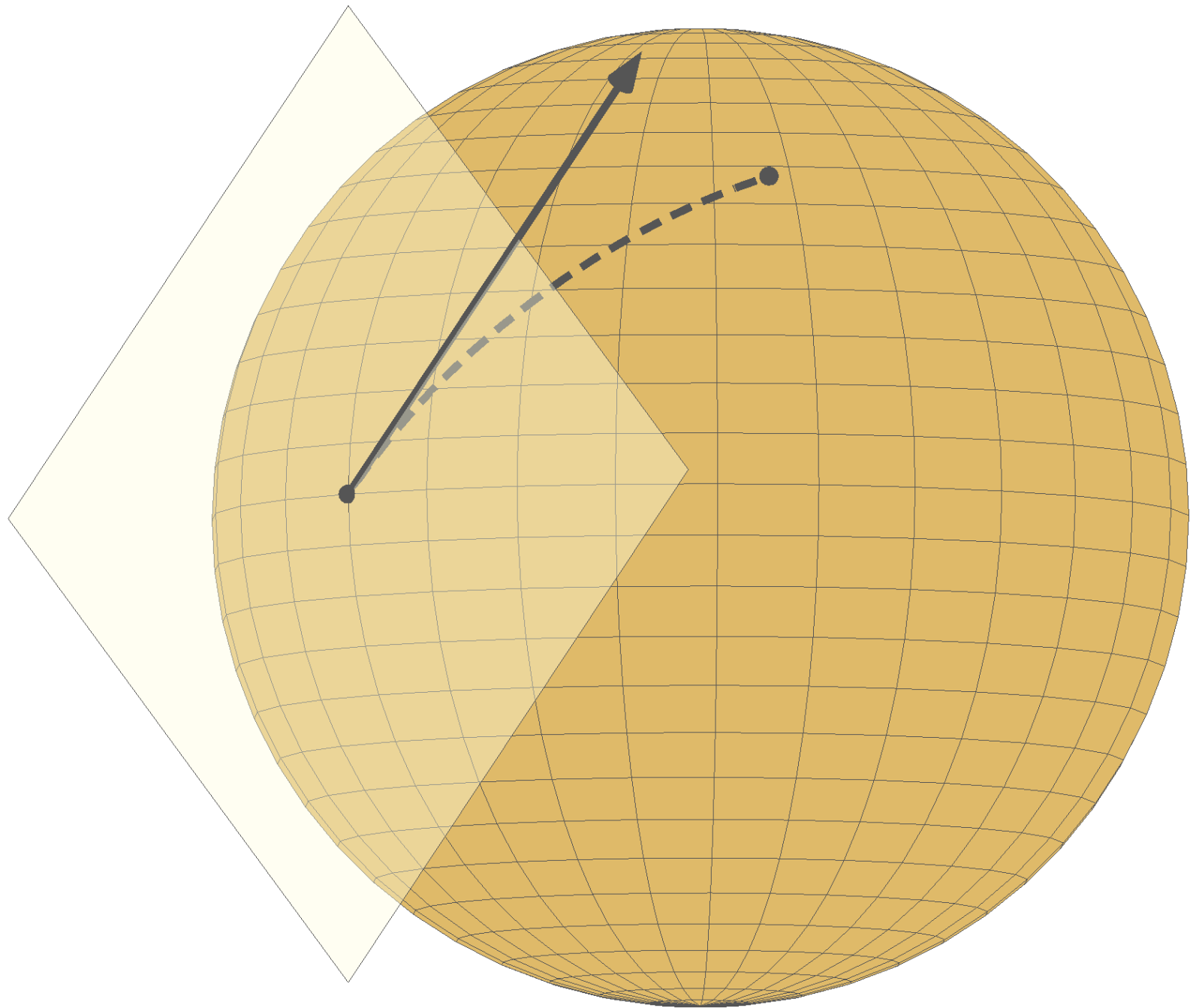














manopt.org

Manopt Home Tutorial Forum

Welcome to

A Matlab toolbox for

Optimization on manifolds is a powerful tool for solving various types of constraints that arise naturally in many applications.

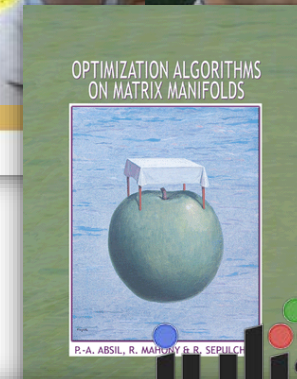
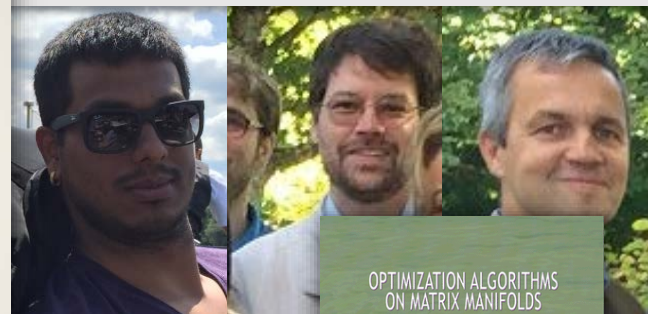
Download  Get started 

NICOLAS BOUMAL

AN INTRODUCTION TO OPTIMIZATION ON SMOOTH MANIFOLDS

DEPARTMENT OF MATHEMATICS, PRINCETON UNIVERSITY

With Mishra, Absil & Sepulchre



pymanopt.org

Pymanopt

Pymanopt is a Python toolbox for optimization on manifolds, that computes gradients and Hessians automatically. It builds upon the Matlab toolbox [Manopt](#) but is otherwise independent of it. Pymanopt aims to lower the barriers for users wishing to use state of the art techniques for optimization on manifolds, by relying on automatic differentiation for computing gradients and Hessians, saving users time and saving them from potential calculation and implementation errors.

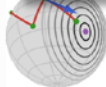
Pymanopt is modular and hence easy to use. All of the automatic differentiation is done behind the scenes so the amount of setup the user needs to do is minimal. Usually only the following steps are required:

1. Instantiate a manifold \mathcal{M} to optimise over
2. Define a cost function $f: \mathcal{M} \rightarrow \mathbb{R}$ to minimise

Lead by Townsend, Koep, Weichwald



manoptjl.org


Manopt.jl
v0.1.0

Search docs

Proximal Maps

Helpers

Data

» Home

Welcome to Manopt.jl

Manopt.Manopt — Module.

Manopt.jl — Optimization on Manifolds in Julia.

[source](#)

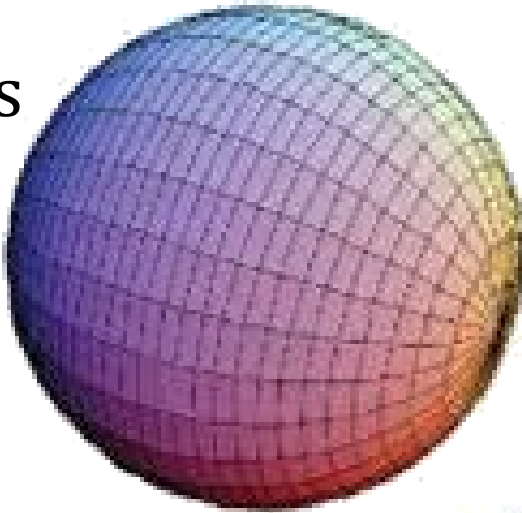
For a function $f: \mathcal{M} \rightarrow \mathbb{R}$ defined on a [Riemannian manifold](#) \mathcal{M} we aim to solve

Lead by Ronny Bergmann

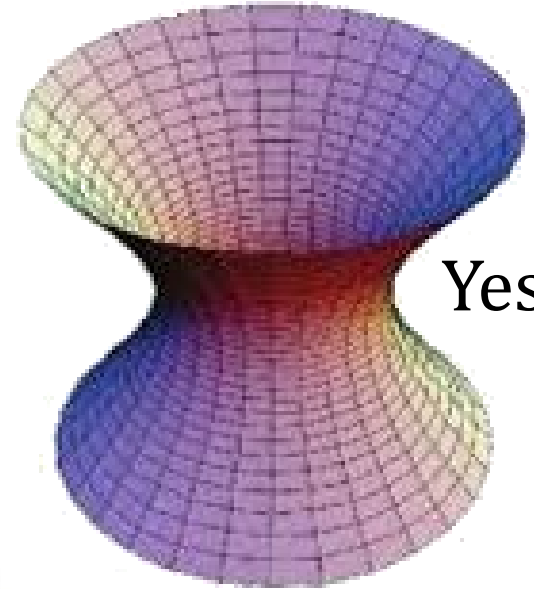
julia

What is a smooth manifold?

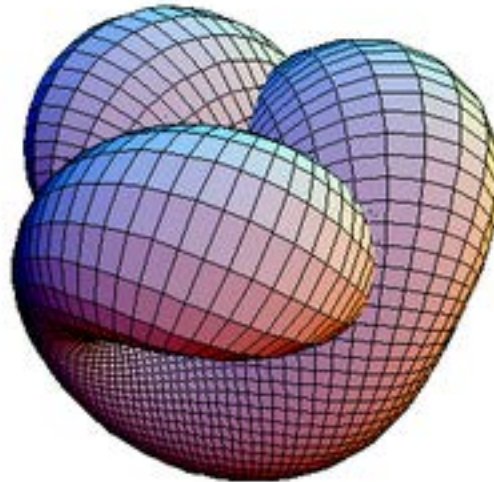
Yes



Yes



No



What is a smooth manifold? (2)

$\mathcal{M} \subseteq \mathcal{E}$ is an **embedded submanifold** of codimension k if:

For each $x \in \mathcal{M}$ there is a neighborhood U of x in \mathcal{E} and a smooth function $h: U \rightarrow \mathbf{R}^k$ such that:

- a) $Dh(x): \mathcal{E} \rightarrow \mathbf{R}^k$ has rank k , and
- b) $\mathcal{M} \cap U = h^{-1}(0) = \{x' \in U: h(x') = 0\}$

These properties allow us to **linearize** the set:

$$h(x + tv) = h(x) + tDh(x)[v] + O(t^2)$$

This is $O(t^2)$ iff $v \in \ker Dh(x)$.

→ **Tangent spaces:** $T_x \mathcal{M} = \ker Dh(x)$, subspace of \mathcal{E}

Bootstrapping our set of tools

1. Smooth maps
2. Differentials of smooth maps
3. Vector fields and tangent bundles
4. Retractions
5. Riemannian metrics
6. Riemannian gradients
7. Riemannian connections
8. Riemannian Hessians
9. Riemannian covariant derivatives along curves

Smooth maps between manifolds

With $\mathcal{M}, \mathcal{M}'$ embedded in $\mathcal{E}, \mathcal{E}'$,

Define: a map $F: \mathcal{M} \rightarrow \mathcal{M}'$ is smooth if:

There exists a neighborhood U of \mathcal{M} in \mathcal{E} and a smooth map $\bar{F}: U \rightarrow \mathcal{E}'$ such that $F = \bar{F}|_{\mathcal{M}}$.

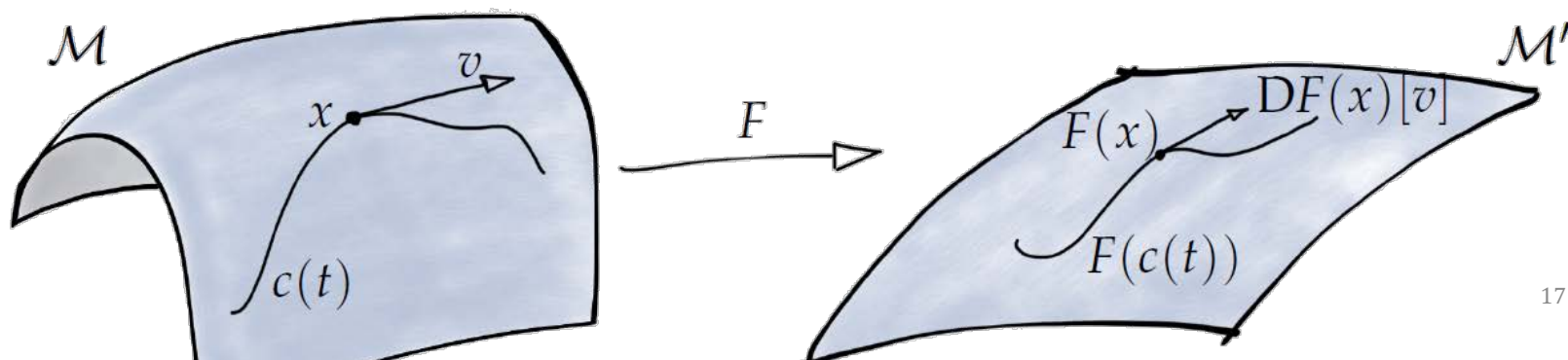
We call \bar{F} a smooth extension of F .

Differential of a smooth map

For $\bar{F}: \mathcal{E} \rightarrow \mathcal{E}'$, we have $D\bar{F}(x)[v] = \lim_{t \rightarrow 0} \frac{\bar{F}(x + tv) - \bar{F}(x)}{t}$.

Claim: for each $v \in T_x \mathcal{M}$, there exists a smooth curve $c: \mathbf{R} \rightarrow \mathcal{M}$ such that $c(0) = x$ and $c'(0) = v$.

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



Differential of a smooth map (2)

Claim: for each $v \in T_x \mathcal{M}$, there exists a smooth curve $c: \mathbf{R} \rightarrow \mathcal{M}$ such that $c(0) = x$ and $c'(0) = v$.

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

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

Claim: we retain linearity, product rule & chain rule.

Vector fields and the tangent bundle

A **vector field** V gives each x a vector $V(x) \in T_x\mathcal{M}$.

What does it mean for V to be smooth?

Define: the **tangent bundle** of \mathcal{M} is the disjoint union of all tangent spaces:

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

Claim: $T\mathcal{M}$ is a manifold (embedded in $\mathcal{E} \times \mathcal{E}$).

V is smooth if it is smooth as a map from \mathcal{M} to $T\mathcal{M}$.

Aside: new manifolds from old ones

Given a manifold \mathcal{M} , we can create a new manifold by considering its tangent bundle.

Here are other ways to recycle:

- **Products** of manifolds: $\mathcal{M} \times \mathcal{M}', \mathcal{M}^n$
- **Open subsets** of manifolds*
- (**Quotienting** some equivalence relations.)

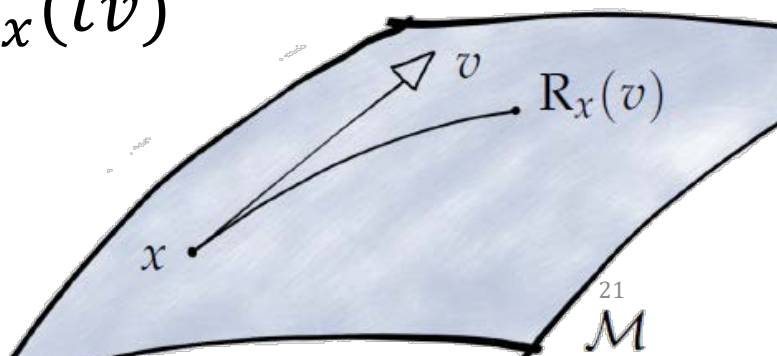
*For embedded submanifolds: subset topology.

Retractions: moving around

Given $(x, v) \in T\mathcal{M}$, we can move away from x along v using any $c: \mathbf{R} \rightarrow \mathcal{M}$ with $c(0) = x$ and $c'(0) = v$.

Retractions are a smooth choice of curves over $T\mathcal{M}$.

Define: A **retraction** is a smooth map $R: T\mathcal{M} \rightarrow \mathcal{M}$ such that if $c(t) = R(x, tv) = R_x(tv)$ then $c(0) = x$ and $c'(0) = v$.



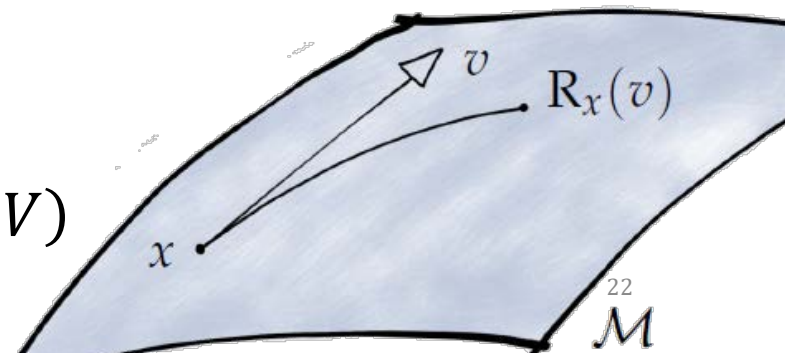
Retractions: moving around (2)

Define: a **retraction** is a smooth map $R: T\mathcal{M} \rightarrow \mathcal{M}$ such that if $c(t) = R(x, tv) = R_x(tv)$, then $c(0) = x$ and $c'(0) = v$.

Equivalently: $R_x(0) = x$ and $DR_x(0) = \text{Id}$.

Typical choice: $R_x(v) = \text{projection of } x + v \text{ to } \mathcal{M}$:

- $\mathcal{M} = \mathbf{R}^n$, $R_x(v) = x + v$
- $\mathcal{M} = S^{n-1}$, $R_x(v) = \frac{x+v}{\|x+v\|}$
- $\mathcal{M} = \mathbf{R}_r^{m \times n}$, $R_X(V) = \text{SVD}_r(X + V)$



Towards gradients: reminders from \mathbf{R}^n

The standard **inner product** on \mathbf{R}^n is $\langle u, v \rangle = u^\top v$.

The **gradient** of a smooth $\bar{f}: \mathbf{R}^n \rightarrow \mathbf{R}$ at x is defined by:

$$D\bar{f}(x)[v] = \langle \text{grad}\bar{f}(x), v \rangle \quad \text{for all } v \in \mathbf{R}^n$$

In particular, $\text{grad}\bar{f}(x)_i = \langle \text{grad}\bar{f}(x), e_i \rangle = \frac{\partial \bar{f}}{\partial x_i}(x)$.

Note: $\text{grad}\bar{f}$ is a smooth vector field on \mathbf{R}^n .

Riemannian metrics

$T_x\mathcal{M}$ is a linear space (subspace of \mathcal{E}).

Pick an **inner product** $\langle \cdot, \cdot \rangle_x$ for each $T_x\mathcal{M}$.

Define: $\langle \cdot, \cdot \rangle_x$ defines a **Riemannian metric** on \mathcal{M} if for any two smooth vector fields V, W the function $x \mapsto \langle V(x), W(x) \rangle_x$ is smooth.

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

Riemannian submanifolds

Let $\langle \cdot, \cdot \rangle$ be the inner product on \mathcal{E} .

Since $T_x \mathcal{M}$ is a linear subspace of \mathcal{E} , one choice is:

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

Claim: this defines a Riemannian metric on \mathcal{M} .

With this metric, \mathcal{M} is a Riemannian submanifold of \mathcal{E} .

Riemannian gradient

Let $f: \mathcal{M} \rightarrow \mathbf{R}$ be smooth on a Riemannian manifold.

Define: the **Riemannian gradient** of f at x is the unique tangent vector at x such that:

$$Df(x)[v] = \langle \operatorname{grad} f(x), v \rangle_x \quad \text{for all } v \in T_x \mathcal{M}$$

Claim: $\operatorname{grad} f$ is a smooth vector field.

Claim: if x is a local optimum of f , $\operatorname{grad} f(x) = 0$.

Gradients on Riemannian **sub**manifolds

Let \bar{f} be a smooth extension of f . For all $v \in T_x\mathcal{M}$:

$$\begin{aligned}\langle \operatorname{grad} f(x), v \rangle_x &= Df(x)[v] \\ &= D\bar{f}(x)[v] = \langle \operatorname{grad} \bar{f}(x), v \rangle\end{aligned}$$

Assume \mathcal{M} is a Riemannian submanifold of \mathcal{E} .
Since $\langle \cdot, \cdot \rangle = \langle \cdot, \cdot \rangle_x$, by uniqueness we conclude:

$$\operatorname{grad} f(x) = \operatorname{Proj}_x \left(\operatorname{grad} \bar{f}(x) \right)$$

Proj_x is the **orthogonal projector** from \mathcal{E} to $T_x\mathcal{M}$.

A first algorithm: gradient descent

For $\bar{f}: \mathbf{R}^n \rightarrow \mathbf{R}$, $x_{k+1} = x_k - \alpha_k \text{grad} \bar{f}(x_k)$

For $f: \mathcal{M} \rightarrow \mathbf{R}$, $x_{k+1} = R_{x_k}(-\alpha_k \text{grad} f(x_k))$

For the analysis, need to understand $f(x_{k+1})$:

The composition $f \circ R_x: T_x \mathcal{M} \rightarrow \mathbf{R}$ is on a linear space, hence we may Taylor expand it.

A first algorithm: gradient descent

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

The composition $f \circ R_x: T_x \mathcal{M} \rightarrow \mathbf{R}$ is on a linear space, hence we may **Taylor** expand it:

$$\begin{aligned} f(R_x(v)) &= f(R_x(0)) + \langle \operatorname{grad}(f \circ R_x)(0), v \rangle + O(\|v\|_x^2) \\ &= f(x) + \langle \operatorname{grad} f(x), v \rangle_x + O(\|v\|_x^2) \end{aligned}$$

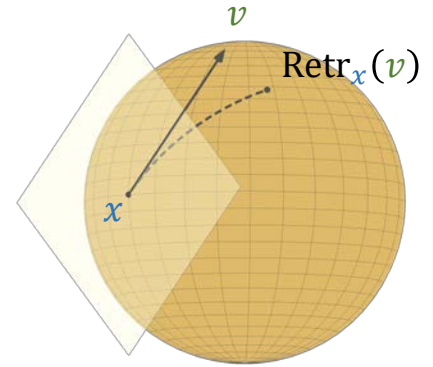
Indeed: $D(f \circ R_x)(0)[v] = Df(R_x(0))[DR_x(0)[v]] = Df(x)[v]$.

Gradient descent on \mathcal{M}

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

A2 $f(R_x(v)) \leq f(x) + \langle v, \text{grad}f(x) \rangle + \frac{L}{2} \|v\|^2$

Algorithm: $x_{k+1} = R_{x_k} \left(-\frac{1}{L} \text{grad}f(x_k) \right)$



Complexity: $\|\text{grad}f(x_k)\| \leq \varepsilon$ with some $k \leq 2L(f(x_0) - f_{\text{low}}) \frac{1}{\varepsilon^2}$

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

$$\Rightarrow f(x_k) - f(x_{k+1}) \geq \frac{1}{2L} \|\text{grad}f(x_k)\|^2$$

$$\mathbf{A1} \Rightarrow f(x_0) - f_{\text{low}} \geq \sum_{k=0}^{K-1} f(x_k) - f(x_{k+1}) > \frac{\varepsilon^2}{2L} K$$

(for contradiction)

Tips and tricks to get the gradient

Use **definition as starting point**: $Df(x)[v] = \dots$

Cheap gradient principle

Numerical check of gradient: Taylor $t \mapsto f(R_x(tv))$

Manopt: `checkgradient(problem)`

Automatic differentiation: Python, Julia

Not Matlab :/—this being said, for theory, often need to manipulate gradient “on paper” anyway.

On to second-order methods

Consider $\bar{f}: \mathbf{R}^n \rightarrow \mathbf{R}$ smooth. **Taylor** says:

$$\bar{f}(x + \boldsymbol{v}) \approx \bar{f}(x) + \langle \text{grad} \bar{f}(x), \boldsymbol{v} \rangle + \frac{1}{2} \langle \boldsymbol{v}, \text{Hess} \bar{f}(x)[\boldsymbol{v}] \rangle$$

If $\text{Hess} \bar{f}(x) \succ 0$, quadratic model minimized for \boldsymbol{v} s.t.:

$$\text{Hess} \bar{f}(x)[\boldsymbol{v}] = -\text{grad} \bar{f}(x)$$

From there, we can construct **Newton's method** etc.

Towards Hessians: reminders from \mathbf{R}^n

Consider $\bar{f}: \mathbf{R}^n \rightarrow \mathbf{R}$ smooth.

The **Hessian** of \bar{f} at x is a linear operator which tells us how the gradient vector field of \bar{f} varies:

$$\text{Hess}\bar{f}(x)[v] = \text{Dgrad}\bar{f}(x)[v]$$

With $\langle u, v \rangle = u^\top v$, yields: $\text{Hess}\bar{f}(x)_{ij} = \frac{\partial^2 \bar{f}}{\partial x_i \partial x_j}(x)$.

Notice that $\text{Hess}\bar{f}(x)[v]$ is a vector in \mathbf{R}^n .

A difficulty on manifolds

A smooth vector field V on \mathcal{M} is a smooth map:
we already have a notion of how to differentiate it.

Example: with $f(x) = \frac{1}{2}x^\top Ax$ on the sphere,

$$V(x) = \text{grad}f(x) = \text{Proj}_x(Ax) = Ax - (x^\top Ax)x$$

$$DV(x)[u] = \cdots = \text{Proj}_x(Au) - (x^\top Ax)u - (x^\top Au)x$$

Issue: $DV(x)[u]$ may not be a tangent vector at x !

Connections:

A tool to differentiate vector fields

Let $\mathcal{X}(\mathcal{M})$ be the set of smooth vector fields on \mathcal{M} .

Given $U \in \mathcal{X}(\mathcal{M})$ and smooth f , $(Uf)(x) = Df(x)[U(x)]$.

A map $\nabla: \mathcal{X}(\mathcal{M}) \times \mathcal{X}(\mathcal{M}) \rightarrow \mathcal{X}(\mathcal{M})$ is a **connection** if:

1. $\nabla_{fU+gW}(V) = f\nabla_U V + g\nabla_W V$
2. $\nabla_U(aV + bW) = a\nabla_U V + b\nabla_U W$
3. $\nabla_U(fV) = (Uf)V + f\nabla_U V$

Example: for $\mathcal{M} = \mathcal{E}$, $(\nabla_U V)(x) = DV(x)[U(x)]$

Example: for $\mathcal{M} \subseteq \mathcal{E}$, $(\nabla_U V)(x) = \text{Proj}_x(DV(x)[U(x)])$

Riemannian connections:

A unique and favorable choice

Let \mathcal{M} be a Riemannian manifold.

Claim: there **exists** a **unique** connection ∇ on \mathcal{M} s.t.:

$$4. \quad (\nabla_U V - \nabla_V U)f = U(Vf) - V(Uf)$$

$$5. \quad U\langle V, W \rangle = \langle \nabla_U V, W \rangle + \langle V, \nabla_U W \rangle$$

It is called the **Riemannian connection** (Levi-Civita).

Claim: if \mathcal{M} is a Riemannian submanifold of \mathcal{E} , then

$$(\nabla_U V)(x) = \text{Proj}_x(DV(x)[U(x)])$$

is the Riemannian connection on \mathcal{M} .

Riemannian Hessians

Claim: $(\nabla_U V)(x)$ depends on U only through $U(x)$.
This justifies the notation $\nabla_u V$; e.g.: $\nabla_u V = \text{Proj}_x(DV(x)[u])$

Define: the **Riemannian Hessian** of $f: \mathcal{M} \rightarrow \mathbf{R}$ at x is a linear operator from $T_x \mathcal{M}$ to $T_x \mathcal{M}$ defined by:

$$\text{Hess}f(x)[u] = \nabla_u \text{grad}f$$

where ∇ is the Riemannian connection.

Claim: $\text{Hess}f(x)$ is self-adjoint.

Claim: if x is a local minimum, then $\text{Hess}f(x) \succcurlyeq 0$.

Hessians on Riemannian submanifolds

$$\text{Hess}f(x)[u] = \nabla_u \text{grad}f(x)$$

On a Riemannian **sub**manifold of a linear space,

$$\nabla_u V = \text{Proj}_x(DV(x)[u])$$

Combining:

$$\text{Hess}f(x)[u] = \text{Proj}_x(D\text{grad}f(x)[u])$$

Hessians on Riemannian submanifolds (2)

$$\text{Hess}f(x)[u] = \text{Proj}_x(\text{Dgrad}f(x)[u])$$

Example: $f(x) = \frac{1}{2}x^\top Ax$ on the sphere in \mathbf{R}^n .

$$V(x) = \text{grad}f(x) = \text{Proj}_x(Ax) = Ax - (x^\top Ax)x$$

$$\text{DV}(x)[u] = \text{Proj}_x(Au) - (x^\top Ax)u - (x^\top Au)x$$

$$\text{Hess}f(x)[u] = \text{Proj}_x(Au) - (x^\top Ax)u$$

Remarkably, $\text{grad}f(x) = 0$ and $\text{Hess}f(x) \succcurlyeq 0$ iff x optimal.

Newton, Taylor and Riemann

Now that we have a Hessian, we might **guess**:

$$f(R_x(v)) \approx m_x(v) = f(x) + \langle \text{grad}f(x), v \rangle_x + \frac{1}{2} \langle v, \text{Hess}f(x)[v] \rangle_x$$

If $\text{Hess}f(x)$ is invertible, $m_x: T_x\mathcal{M} \rightarrow \mathbf{R}$ has one critical point, solution of this linear system:

$$\text{Hess}f(x)[v] = -\text{grad}f(x) \quad \text{for } v \in T_x\mathcal{M}$$

Newton's method on \mathcal{M} : $x_{\text{next}} = R_x(v)$.

Claim: if $\text{Hess}f(x^*) \succ 0$, quadratic local convergence.

We need one more tool...

The truncated expansion

$$f(R_x(v)) \approx f(x) + \langle \text{grad} f(x), v \rangle_x + \frac{1}{2} \langle v, \text{Hess} f(x)[v] \rangle_x$$

is **not quite correct** (well, not always...)

To see why, let's expand f along some smooth curve.

We need one more tool... (2)

Let's expand f along some smooth curve.

With $c: \mathbf{R} \rightarrow \mathcal{M}$ s.t. $c(0) = x$ and $c'(0) = v$, the composition $g = f \circ c$ maps $\mathbf{R} \rightarrow \mathbf{R}$, so Taylor holds:

$$g(t) \approx g(0) + tg'(0) + \frac{t^2}{2} g''(0)$$

$$g(0) = f(x)$$

$$g'(t) = Df(c(t))[c'(t)] = \langle \text{grad} f(c(t)), c'(t) \rangle_{c(t)}$$

$$g'(0) = \langle \text{grad} f(x), v \rangle_x$$

$$g''(0) = ???$$

Differentiating vector fields along curves

A map $Z: \mathbf{R} \rightarrow T\mathcal{M}$ is a **vector field along $c: \mathbf{R} \rightarrow \mathcal{M}$** if $Z(t)$ is a tangent vector at $c(t)$.

Let $\mathcal{X}(c)$ denote the set of smooth such fields.

Claim: there **exists** a **unique** operator $\frac{D}{dt}: \mathcal{X}(c) \rightarrow \mathcal{X}(c)$ s.t.

1. $\frac{D}{dt}(aY + bZ) = a \frac{D}{dt}Y + b \frac{D}{dt}Z$
2. $\frac{D}{dt}(gZ) = g'Z + g \frac{D}{dt}Z$
3. $\frac{D}{dt}(U(c(t))) = \nabla_{c'(t)}U$
4. $\frac{d}{dt}\langle Y(t), Z(t) \rangle_{c(t)} = \left\langle \frac{D}{dt}Y(t), Z \right\rangle_{c(t)} + \left\langle Y(t), \frac{D}{dt}Z(t) \right\rangle_{c(t)}$

where ∇ is the Riemannian connection.

Differentiating fields along curves (2)

Claim: there exists a unique operator $\frac{D}{dt}: \mathcal{X}(c) \rightarrow \mathcal{X}(c)$ s.t.

1. $\frac{D}{dt}(aY + bZ) = a \frac{D}{dt}Y + b \frac{D}{dt}Z$
2. $\frac{D}{dt}(gZ) = g'Z + g \frac{D}{dt}Z$
3. $\frac{D}{dt}(U(c(t))) = \nabla_{c'(t)}U$
4. $\frac{d}{dt}\langle Y(t), Z(t) \rangle_{c(t)} = \left\langle \frac{D}{dt}Y(t), Z \right\rangle_{c(t)} + \left\langle Y(t), \frac{D}{dt}Z(t) \right\rangle_{c(t)}$

where ∇ is the Riemannian connection.

Claim: if \mathcal{M} is a Riemannian submanifold of \mathcal{E} ,

$$\frac{D}{dt}Z(t) = \text{Proj}_{c(t)} \left(\frac{d}{dt}Z(t) \right)$$

1. $\frac{D}{dt}(aY + bZ) = a \frac{D}{dt}Y + b \frac{D}{dt}Z$
2. $\frac{D}{dt}(gZ) = g'Z + g \frac{D}{dt}Z$
3. $\frac{D}{dt}(U(c(t))) = \nabla_{c'(t)}U$
4. $\frac{d}{dt}\langle Y(t), Z(t) \rangle_{c(t)} = \left\langle \frac{D}{dt}Y(t), Z \right\rangle_{c(t)} + \left\langle Y(t), \frac{D}{dt}Z(t) \right\rangle_{c(t)}$

With $g(t) = f(c(t))$ and $g'(t) = \langle \text{grad}f(c(t)), c'(t) \rangle_{c(t)}$:

$$g''(t) = \left\langle \frac{D}{dt} \text{grad}f(c(t)), c'(t) \right\rangle_{c(t)} + \left\langle \text{grad}f(c(t)), \frac{D}{dt} c'(t) \right\rangle_{c(t)}$$

$$= \langle \nabla_{c'(t)} \text{grad}f, c'(t) \rangle_{c(t)} + \langle \text{grad}f(c(t)), c''(t) \rangle_{c(t)}$$

$$g''(0) = \langle \text{Hess}f(x)[v], v \rangle_x + \langle \text{grad}f(x), c''(0) \rangle_x$$

With $g(t) = f(c(t))$ and $g'(t) = \langle \text{grad} f(c(t)), c'(t) \rangle_{c(t)}$:

$$g''(t) = \left\langle \frac{D}{dt} \text{grad} f(c(t)), c'(t) \right\rangle_{c(t)} + \left\langle \text{grad} f(c(t)), \frac{D}{dt} c'(t) \right\rangle_{c(t)}$$

$$= \langle \nabla_{c'(t)} \text{grad} f, c'(t) \rangle_{c(t)} + \langle \text{grad} f(c(t)), c''(t) \rangle_{c(t)}$$

$$g''(0) = \langle \text{Hess} f(x)[v], v \rangle_x + \langle \text{grad} f(x), c''(0) \rangle_x$$

$$f(c(t)) = f(x) + t \cdot \langle \text{grad} f(x), v \rangle_x + \frac{t^2}{2} \cdot \langle \text{Hess} f(x)[v], v \rangle_x \\ + \frac{t^2}{2} \cdot \langle \text{grad} f(x), c''(0) \rangle_x + O(t^3)$$

The annoying term vanishes at critical points and for special curves (special retractions). Mostly fine for optimization.

Trust-region method: Newton's with a safeguard

With the same tools, we can design a Riemannian trust-region method: **RTR** (Absil, Baker & Gallivan '07).

Approximately **minimizes quadratic model** of $f \circ R_{x_k}$ **restricted to a ball** in $T_{x_k}\mathcal{M}$, with dynamic radius.

Complexity known. Excellent performance, also with approximate Hessian (e.g., finite differences of $\text{grad} f$).

In Manopt, call `trustregions(problem)`.

In the lecture notes:

- **Proofs** for all claims in these slides
- **References** to the (growing) literature
- Short descriptions of **applications**
- Fully worked out **manifolds**
E.g.: Stiefel, fixed-rank matrices, general $\{x \in \mathcal{E} : h(x) = 0\}$
- Details about **computation**, pitfalls and tricks
E.g.: how to compute gradients, checkgradient, checkhessian
- Theory for **general manifolds**
- Theory for **quotient manifolds**
- Discussion of more **advanced geometric tools**
E.g.: distance, exp, log, transports, Lipschitz, finite differences
- Basics of **geodesic convexity**