NTNU

Norwegian University of Science and Technology

An Introduction to Optimization on Riemannian Manifolds

Ronny Bergmann

NTNU, Trondheim, Norway.

Constrained Optimization Methods for Models in Data Fusion,
Simula Oslo, Oslo, Norway.

March 27, 2023

Motivation: Constraint vs Unconstraint Optimization

We want to consider a special case of constrained optimisation

$$\min_{x \in S} f(x) \operatorname*{arg\,min}_{x \in S} f(x),$$

instead of minimal value $f(x^*)$ often minimizer x^* of interest
classical: Constrained Optimization. Describe $S \subset \mathbb{R}^n$ with constraints

$$S = \{x \,|\, g(x) \leq 0 \text{ and } h(x) = 0\}, \qquad g\colon \mathbb{R}^n \to \mathbb{R}^{m_1}, h\colon \mathbb{R}^n \to \mathbb{R}^{m_2}$$

- special algorithms necessary (ALM, EPM)
- g, h might have complicated gradients or be high-dimensional.

today. If $S = \mathcal{M}$ is "nice", i. e. a Riemannian manifold $\mathcal{M}$:

- different notion of e. g. gradient and "means to move around"
- we obtain unconstrained problems on $\mathcal{M}$

$\Rightarrow$ use gradient descent, CG, quasi Newton, trust region,… on $\mathcal{M}$!

Overview

1. A few examples
2. (embedded) Manifolds & Tangent spaces
3. Retractions (moving around on a manifold)
4. First order methods (differentials and gradients)
5. Algorithms & Software

Literature.

- Riemannian Manifolds. do Carmo 1992; Lee 2018
- Optimization on Manifolds. Absil, Mahony, and Sepulchre 2008; Boumal 2023

Example 1: Rayleigh Quotient

Let $A \in \mathbb{R}^{n\times n}$, $A = A^\mathrm{T}$, with eigenvalues $\lambda_1 \leq \cdots \leq \lambda_n$ be given.
We can find an eigenvector v_1 by

$$\operatorname*{arg\,min}_{\substack{x\in\mathbb{R}^n\\ x\neq 0}} f(x), \qquad f(x) = \frac{\langle x, Ax\rangle}{\langle x, x\rangle}.$$

- Since $Av_1 = \lambda v_1 \Rightarrow f(v_1) = \lambda_1$
- ⚠ any scaled αv_1, $\alpha \neq 0$ is also a minimizer!
- $\Rightarrow$ Newton iteration might even diverge.

Solution. We rephrase the problem to

$$\operatorname*{arg\,min}_{x\in\mathbb{S}^n} \frac{\langle x, Ax\rangle}{\langle x, x\rangle}\langle x, Ax\rangle, \qquad \mathbb{S}^{n-1} = \left\{x \in \mathbb{R}^n \,\middle|\, \|x\| = 1\right\}$$

An optimisation problem on the $(n-1)$-sphere in $\mathbb{R}^n$.

Example 2: multiple Eigenvectors & The Stiefel manifold

Goal. Find a orthonormal basis $X \in \mathbb{R}^{n\times p}$ for the space spanned by $v_1, \ldots, v_p$ corresponding $\lambda_1 \leq \cdots \leq \lambda_p$.

Then we use columns of X an ONB $\Leftrightarrow X^T X = I_p$, the unit matrix $I_p \in \mathbb{R}^{p\times p}$.

We collect all such matrices representing ONBs for any p-dimensional subspace in

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

called the Stiefel manifold.

Our optimization problem to find a best basis reads

$$\operatorname*{arg\,min}_{X\in\mathrm{St}(n,p)} f(X), \qquad f(X) = \mathrm{tr}(X^\mathrm{T} AX)$$

and at a minimizer X^* we have the minimal value $f(X^*) = \sum_{i=1}^{p} \lambda_i$.

Example 3: Subspaces & The Grassmann manifold

Observation. The order of basis vectors in the last example is irrelevant.
Even more. $f(X) = f(Y)$ for $X, Y \in \mathrm{St}(n,p)$ whenever $\mathrm{span}(X) = \mathrm{span}(Y)$

Interpretation. Rotating the basis of the subspace to a new basis of the subspace still yields the same value
$\Rightarrow$ Let's built equivalence classes

$$[X] := \left\{ Y \in \mathrm{St}(n,p) \,\middle|\, \mathrm{span}(X) = \mathrm{span}(Y) \right\}$$

New Goal. Find the subspace

$$\operatorname*{arg\,min}_{[X] \in \mathrm{Gr}(n,p)} g([X]), \qquad g([X]) = \mathrm{tr}(X^{\mathrm{T}} A X)$$

where

$$\mathrm{Gr}(n,p) := \left\{ [X] \,\middle|\, X \in \mathrm{St}(n,p) \right\},$$

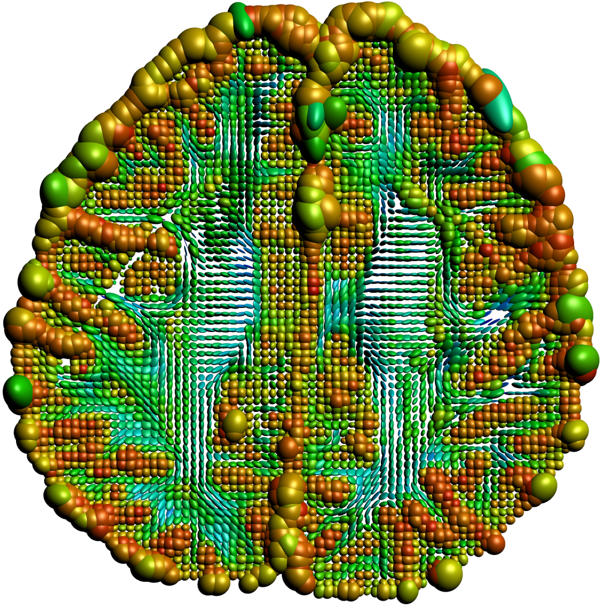
is the Grassmann manifold, i. e. the space of all p-dimensional subspaces of $\mathbb{R}^n$.

Example 4: DT-MRI & Image Denoising

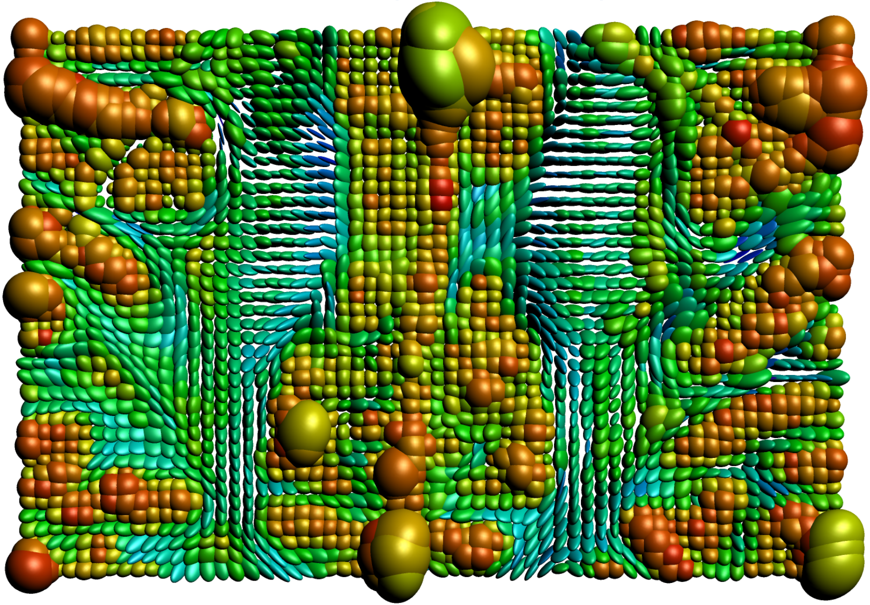
$\mathcal{M} = (\mathcal{P}_3)^{n \times m}$

An image of
"diffusion tensor pixel"
➜ denoising.

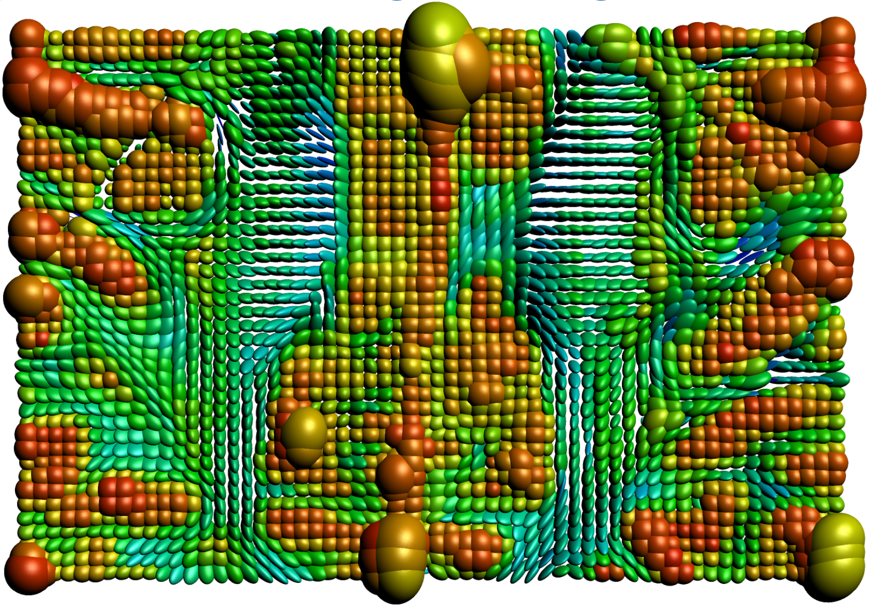
e. g. using ℓ_2–TV



Example 4: DT-MRI & Image Denoising



Example 4: DT-MRI & Image Denoising



Intuition to Tangent space & (sub)manifolds

Intuitive Definition. A (smooth, Riemannian) manifold $\mathcal{M}$ is a set that "locally looks like" $\mathbb{R}^d$
$\Rightarrow$ Collecting all derivatives $c'(0)$ of curves $c\colon I \to \mathcal{M}$ through $c(0) = p$
We obtain a "space of directions"

Example. For the sphere $\mathbb{S}^{n-1} \subset \mathbb{R}^n$ at $c(0) = p$ fulfils $p^\mathrm{T} p = \lVert p \rVert^2 = 1$.
Hence

$$c'(t) \in T_p\mathbb{S}^{n-1} := \{X \in \mathbb{R}^n \,|\, X^\mathrm{T} p + p^\mathrm{T} X = 0\} = \{X \in \mathbb{R}^n \,|\, X^\mathrm{T} p = 0\}$$

is a $(n-1)$-dimensional vector space called the tangent space $T_p\mathbb{S}^{n-1}$ at p.

In general. In order to have a manifold, this "looks like" $\mathbb{R}^d$ always has to be the same dimension d.

Embedded submanifold & Tangent spaces

Definition (Boumal 2023, Def. 3.10,)

Let $\mathcal{E}$ be a linear space of dimension n. A nonempty subset $\mathcal{M}$ of $\mathcal{E}$ is a (smooth) embedded submanifold of $\mathcal{E}$ of dimension d if either

1. $d = n$ and $\mathcal{M}$ is open in $\mathcal{E}$
2. $d = n - k$ for some $k \geq 1$ and for each $p \in \mathcal{M}$ there exists a neighbourhood $\mathcal{U} \subset \mathcal{E}$ and $h\colon \mathcal{U} \to R^k$ such that
   - 2.1 If $y \in \mathcal{U}$ then $h(y) = 0 \Leftrightarrow y \in \mathcal{M}$
   - 2.2 rank $Dh(p) = k$

Tangent space. The rank condition ensures that $\ker Dh(p)$ is d-dimensional. This also forms a vector space, the tangent soace $T_p\mathcal{M}$ at p.

- inherits an inner product by restriction from $\mathcal{E}$, denoted by $\langle \cdot, \cdot \rangle_p$
- the disjoint union of all $\mathcal{T}_p\mathcal{M}$ is the tangent bundle $T\mathcal{M}$ with elements (p, X).

For $\mathbb{S}^{n-1}$ even more: we one global $h(p) = \|p\|^2 - 1$ with $\ker Dh(p) = \{X \in \mathbb{R}^n \,|\, \langle X, p \rangle = 0\}$.

Smooth functions and their Differential

Smooth functions. A function $f\colon \mathcal{M} \to \mathbb{R}$ is called smooth if it is given (locally) as the restriction of a function $\bar{f}\colon \to\mathbb{R}$, i.e. $f = \bar{f}\big|_{\mathcal{M}}$

The (Euclidean) Differential. Classically (for the embedded $\bar{f}$) we have

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

but for f we have the problem that $x + tv$ is not necessarily on $\mathcal{M}$!

Idea. use as "directions" in the directional derivative
the curves $c\colon I \to \mathcal{M}$ with $c\colon I \to \mathcal{M}, c(0) = p, c'(0) = X$ and define

The Differential of $f\colon \mathcal{M} \to \mathbb{R}$.

$$Df(p)[X] \coloneqq \frac{\mathrm{d}}{\mathrm{d}t} f(c(t)),$$

Fortunately. Both are equivalent, i. e. restricting the Euclidean differential to $T_p\mathcal{M}$ yields the Riemannian one: $Df(p) = D\bar{f}(p)\big|_{T_p\mathcal{M}}$.

Retractions: Moving around on a Manifold.

Iterative Algorithms usually are at some point x, find a (descent) direction v and a step size s and obtain $x^{(k+1)} = x^{(k)} + sv$

How to move on $\mathcal{M}$ given some $p^{(k)}$ and a $X \in T_p\mathcal{M}$?

Definition (Boumal 2023, Def. 3.47)

A retraction on a manifold $\mathcal{M}$ is a smooth map

$$R\colon T\mathcal{M} \to \mathcal{M}, \qquad (p, X) \mapsto R_p(X) \in \mathcal{M}$$

such that each curve $c(t) = R_p(tX)$ satisfies $c(0) = p$, $c'(0) = X$.

Example 1. on $\mathcal{M} = \mathbb{S}^{n-1}$ one can use $R_p(X) = \frac{p+X}{\lVert p+X \rVert}$

Example 2. on $\mathcal{M} = \mathbb{S}^{n-1}$ one can use $R_p(X) = \cos\bigl(\lVert X \rVert_p\bigr)p + \sin\bigl(\lVert X \rVert_p\bigr)\frac{X}{\lVert X \rVert_p}$
$\Rightarrow$ we trace great circles (shortest paths)
This retraction has a special name: the exponential map $\exp_p X$.

The Riemannian Gradient

Definition (Boumal 2023, Def. 3.58)

Let $f\colon \mathcal{M} \to \mathbb{R}$ be smooth on a Riemannian manifold $\mathcal{M}$
The Riemannian gradient of f is the vector field $\operatorname{grad} f$ on $\mathcal{M}$ uniquely defined by the following identities:

$$\text{For all } (p, X) \in T\mathcal{M} \text{ it holds } Df(p)[X] = \langle X, \operatorname{grad} f(p) \rangle_p,$$

where Df denotes the differential.

- $\operatorname{grad} f(p) \in T_p\mathcal{M}$ is the (tangent) direction of steepest ascent
- for the embedded Riemannian submanifolds: $\operatorname{grad} f = \operatorname{proj}_{T_p\mathcal{M}}(\operatorname{grad} \bar{f}(p))$
- (like in $\mathbb{R}^n$:) p is a critical point of $f \Leftrightarrow \operatorname{grad} f(p) = 0 \in T_p\mathcal{M}$.

Gradient Descent

Euclidean Gradient Descent. $x^{(k+1)} = x^{(k)} - \alpha_k \operatorname{grad} f(x^{(k)})$ for some $\alpha_k > 0$.

Riemannian Gradient Descent.
Use the Riemannian gradient and replace "$-$".

Input: $p^{(0)} \in \mathcal{M}$

1: $k \leftarrow 0$
2: **while** not converged **do**
3: Pick a step size $\alpha_k > 0$
4: $p^{(k+1)} = R_{p^{(k)}}\bigl(-\alpha_k s^{(k)}\bigr), \quad s^{(k)} = \operatorname{grad} f(p^{(k)})$
5: $k \leftarrow k+1$
6: **end while**

Output: $p^{(N)}$

Stepsize. For example: Armijo line-search along $\varphi(t) = f\bigl(R_{p^{(k)}}(-ts^{(k)})\bigr)$

Stopping Criteria & the Distance on a manifold

Variant 1. The inner product $\langle\cdot,\cdot\rangle_p$ induces a norm $\|\cdot\|_p$ on any $T_p\mathcal{M}$.

$\Rightarrow$ Given a tolerance $\varepsilon_1 > 0$ stop when $\|\operatorname{grad} f(p^{(k)})\|_{p^{(k)}} < \varepsilon_1$

Variant 2. There is a measure of length for curves $c\colon I \to \mathcal{M}$ induced by $\langle\cdot,\cdot\rangle_p$. Introduce a distance $d_{\mathcal{M}}(p,q)$ as the length of the shortest curve connecting both (a shortest geodesic).

$\Rightarrow$ Given a tolerance $\varepsilon_2 > 0$ stop when $d_{\mathcal{M}}(p^{(k-1)}, p^{(k)}) < \varepsilon_2$

Variant 3. ...as a fallback of course after a maximal number N of iterations.

“Comparing” points and vectors

For Quasi Newton one classically (Euclidean) needs for the secant equation

- $s^{(k)} = x^{(k+1)} - x^{(k)}$
- $y^{(k)} = \operatorname{grad} f(x^{(k+1)}) - \operatorname{grad} f(x^{(k)})$

Problem 1. We do not have a difference of points.
$\Rightarrow$ Interpret $d = z - x$ is the direction pointing from x to z.

We are looking for X such that $R_p(X) = q$ or the inverse retraction $R_p^{-1}(q)$!
For the special case of $R_p = \exp_p$ the inverse is called logarithmic map $\log_p \cdot$.
Obs! the logarithmic map is often not globally defined.

Problem 2. For two gradients $\operatorname{grad} f(p) \in T_p\mathcal{M}$ and $\operatorname{grad} f(q) \in T_q\mathcal{M}$ the difference is not defined, because they live in different spaces

We need a function $T_{q\leftarrow p}$ to “transport” tangent vectors.

Vector Transport

Definition (Absil, Mahony, and Sepulchre 2008, Def. 8.1.1)

Let $\mathcal{M}$ be a manifold, and $p \in \mathcal{M}$ and $X \in T_p\mathcal{M}$.

Then a vector transport $T_{p,X} : T_p\mathcal{M} \to T_q\mathcal{M}$ is a smooth mapping associated to a retraction with $R_p(X) = q$ such that

1. $\mathcal{T}_{p,X} Y \in T_q\mathcal{M}$
2. $\mathcal{T}_{p,0_p} Y = Y$ for all $Y \in T_p\mathcal{M}$,
3. $\mathcal{T}_{p,X}(\alpha Y + \beta Z) = \alpha T_{p,X} Y + \beta T_{p,X} Z$ for all $\alpha, \beta \in \mathbb{R}$, $Y, Z \in T_p\mathcal{M}$

hold.

Alternative Notation. $T_{q\leftarrow p}$ as long as X such that $q = R_p(X)$ is uniquely defined.

Special case. There exists a vector transport that preserves norms $\lVert Y \rVert_p = \lVert T_{p,X} Y \rVert_q$ and angles $\langle Y, Z \rangle_p = \langle T_{p,X} Y, T_{p,X} Z \rangle_q$.
This vector transport is called parallel transport $P_{p,X}$ or $P_{q\leftarrow p}$.

Quasi Newton – Idea

For the Hessian of f we can also start intuitively: How does the gradient $\operatorname{grad} f$ change?

Given a point $p \in \mathcal{M}$ and a direction $X \in T_p\mathcal{M}$ we introduce again a curve $c(t) = R_p(tX)$ to define[1]

$$\operatorname{Hess} f(p)[X] := \lim_{t \to 0} \frac{T_{p \leftarrow c(t)} \operatorname{grad} f(c(t)) - \operatorname{grad} f(p)}{t}$$

Newton equation. We can find a descent direction $X \in T_{p^{(k)}}\mathcal{M}$ by solving

$$\operatorname{Hess} f(p^{(k)})[X] = -\operatorname{grad} f(p^{(k)})$$

Goal. Approximate $\operatorname{Hess} f(p^{(k)}) \approx \mathcal{H}_k \colon T_p\mathcal{M} \to T_p\mathcal{M}$.

[1] formerly done using a connection ∇ which "describes how the metric changes" and then define $\operatorname{Hess} f(p)[X] = \nabla_X \operatorname{grad} f(p)$.

The Riemannian Secant equation

We want to choose $\mathcal{H}_{k+1}$ such that it fulfils the secant equation

$$\mathcal{H}_{k+1}[s^{(k)}] = y^{(k)} \quad \text{or equivalently} \quad \mathcal{B}_{k+1}[y^{(k)}] = s^{(k)}$$

where

- $s^{(k)} = T_{p^{(k+1)} \leftarrow p^{(k)}} R^{-1}_{p^{(k)}} p^{(k+1)}$
- $y^{(k)} = \operatorname{grad} f(p^{(k+1)}) - T_{p^{(k+1)} \leftarrow p^{(k)}} \operatorname{grad} f(p^{(k+1)})$.

Updates similar to the Euclidean case for both $\mathcal{H}_{k+1}$ or $\mathcal{B}_{k+1}$

- BFGS [Huang, Absil, and Gallivan 2018]
- DFP
- Broyden [Huang, Gallivan, and Absil 2015]
- limited memory BFGS
- Symmetric Rank 1 (SR1)

Implementing Manifolds & Optimisation – in Julia.

Goals.

- abstract definition of manifolds and properties thereon
  e. g. different metrics, retractions, embeddings

$\Rightarrow$ implement abstract algorithms for generic manifolds

- easy to implement own manifolds & easy to use
- well-documented and well-tested
- fast.

Why Julia?

- high-level language, properly typed
- multiple dispatch (cf. `f(x)`, `f(x::Number)`, `f(x::Int)`)
- just-in-time compilation, solves two-language problem
- I like the language – and the community.

Implementing a Riemannian Manifold

`ManifoldsBase.jl` uses a `AbstractManifold{𝔽}` with type parameter $\mathbb{F} \in \{\mathbb{R}, \mathbb{C}, \mathbb{H}\}$ to provide an interface for implementing functions like

- `inner(M, p, X, Y)` for the Riemannian metric $\langle X, Y \rangle_p$
- `exp(M, p, X)` and `log(M, p, q)`,
- more general: `retract(M, p, X, m)`, where `m` is a retraction method
- similarly: `parallel_transport(M, p, X, q)` and `vector_transport_to(M, p, X, q, m)`

for your manifold `M` a subtype of the `AbstractManifold{𝔽}`.

☺ mutating version `exp!(M, q, p, X)` works in place in `q`

➔ basis for generic algorithms working on any `Manifold` and generic functions like `norm(M,p,X)`, `geodesic(M, p, X)` and `shortest_geodesic(M, p, q)`

🔗 juliamanifolds.github.io/ManifoldsBase.jl/

Manifolds.jl – A library of manifolds in Julia

`Manifolds.jl` is build upon `ManifoldsBase.jl` interface. [Axen, Baran, RB, and Rzecki 2021]

Features.

- different metrics
- Lie groups
- Build manifolds using
  - Product manifold $\mathcal{M}_1 \times \mathcal{M}_2$
  - Power manifold $\mathcal{M}^{n \times m}$
  - Tangent bundle
- Quotient manifolds
- Embedded manifolds
- perform statistics
- well-documented, including formulae and references
- well-tested, >98 % code cov.

Manifolds. For example

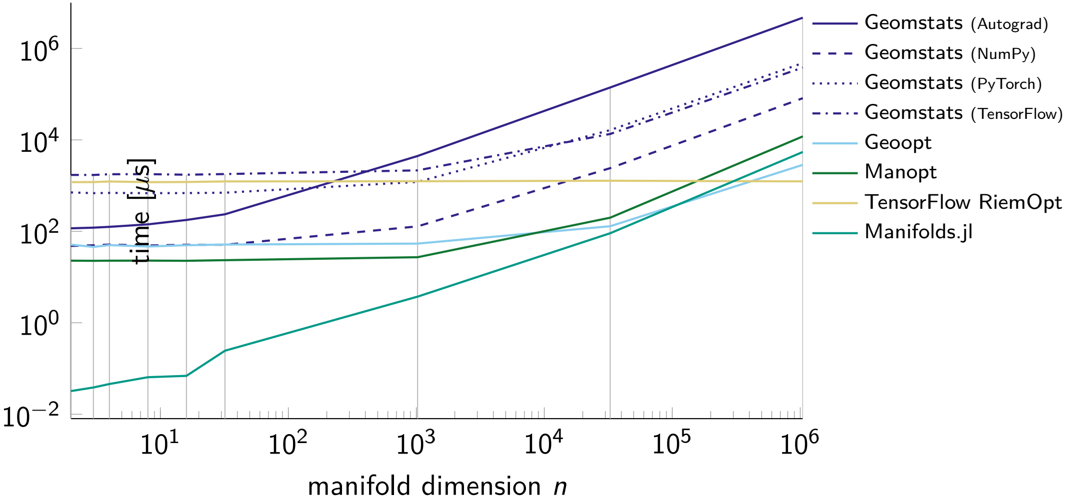
- (unit) Sphere, Circle & Torus
- Fixed Rank Matrices
- (Generalized) Stiefel & Grassmann
- Hyperbolic space
- Rotations, $\mathrm{O}(n)$, $\mathrm{SO}(n)$, $\mathrm{SU}(n)$
- several further Lie groups
- Symmetric positive definite matrices
- Symplectic & Symplectic Stiefel
- Kendall's shape space
- …

juliamanifolds.github.io/Manifolds.jl/

JuliaCon 2020 youtu.be/md-FnDGCh9M

Benchmark of $\log_p$ Hyperbolic space $\mathbb{H}^n$

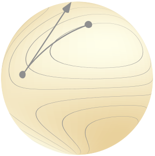
For $n = 2, 3, 2^2, 2^3, 2^4, 2^5, 2^{10}, 2^{15}, 2^{20}$ we compare



➔ For $n > 2^{16}$: PyTorch & TensorFlow based packages faster.

…we could maybe try using `LazyArrays.jl` in Julia.

Manopt.jl – A framework



Goal. Provide optimisation algorithms on Riemannian manifolds, using `ManifoldsBase.jl` to work on any manifold from `Manifolds.jl`.

Generic Framework.

- `AbstractManifoldProblem p` contains static information: $\mathcal{M}$, f, $\operatorname{grad} f$,...
- `AbstractManoptSolverState s` specifies a solver, stores its parameters and values

For your own solver, implement

- `initialize_solver!(p, s)`
- `step_solver!(p, s, i)`

To run an algorithm: `solve!(p, s)`

High level interfaces. E.g.

`gradient_descent(M, f, grad_f, p0)`

setup problem & state, run the algorithm.

Easy access to

- debug, record & status
- step size algorithms
- (modular) stopping criteria.

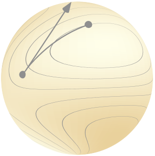
Manopt Family.

manoptjl.org [RB 2022]

manopt.org [Boumal, Mishra, Absil, and Sepulchre 2014]

pymanopt.org [Townsend, Koep, and Weichwald 2016]

Manopt.jl – Algorithms



Derivative-free

- Nelder-Mead
- Particle Swarm

First order

- Gradient descent:
  Alternating, Conjugate Gradient,
  Momentum, Nesterov, Stochastic
- Subgradient Method
- Quasi Newton
  L-BFGS, BFGS, DFP, Broyden, SR1, …
- Levenberg Marquardt

Proximal map based

- Cyclic Proximal Point Algorithm
- Douglas-Rachford

Primal-Dual

- Chambolle-Pock
- Primal-Dual Semismooth Newton

Second order

- Trust Regions with TCG sub-solver

Constrained

- Augmented Lagrangian Method
- Exact Penalty Method
- Frank-Wolfe Method

manoptjl.org

JuliaCon 2022 youtu.be/thbekfsyhCE

Code Example: Rayleigh Quotient

For the Rayleigh quotient

$$f(p) = p^{\mathrm{T}} A p$$

on the sphere $\mathcal{M} = \mathbb{S}^{n-1}$ we can easily state the Riemannian gradient can also be stated direction (or using the projection)

$$\operatorname{grad} f(p) = 2(I_n - pp^{\mathrm{T}})Ap$$

Let's take a look at the numerics

Outlook: Constrained Optimisation on Manifolds

One can consider problems like

[Liu and Boumal 2019; RB and Herzog 2019]

$$
\begin{aligned}
&\operatorname*{arg\,min}_{p\in\mathcal{M}} f(p)\\
&\text{subject to } g_i(p) \leq 0, \quad i = 1, \ldots, m\\
&\qquad\qquad\quad h_j(p) = 0, \quad j = 1, \ldots, p
\end{aligned}
$$

where $g_i, h_j \colon \mathcal{M} \to \mathbb{R}$ describe constraints to p.

$\Rightarrow$ Classical algorithms (ALM, EPM) adapted

falconlightbulb We can choose our own trade-off between geometry and constraint.

Software packages – An Overview

We[2] founded the `JuliaManifolds`, GitHub Community for manifold related packages in Julia

Currently our main packages are (ordered by age)

Manopt.jl Optimisation on Riemannian manifolds, based on `ManifoldsBase.jl` [RB 2022]

Manifolds.jl A library of Riemannian manifolds and Lie groups [Axen, Baran, RB, and Rzecki 2021]

ManifoldsBase.jl A lightweight interface to implement and work on manifolds

ManifoldDiff.jl (automatic) differentiation on Riemannian manifolds and a function library of differentials, gradients,...

ManifoldDiffEq.jl differential equations on Riemannian manifolds

ManoptExamples.jl A collection of examples and benchmarks for `Manopt.jl`

[2] Seth Axen, U Tübingen; Mateusz Baran, AGH Krakow; RB, NTNU

Summary

- constrained optimization turns into unconstraint optimization on a manifold $\mathcal{M}$
- many algorithms can (and have been) generalized to manifolds
- Implementations exist in several languages
- We considered manifolds and algorithms in Julia

Outlook

- manifolds can be defined more general, without an embedding
- numerically we embed somewhere to represent points as arrays
- Riemannian Hessians
- Euclidean AD tools can be used (with some post-processing) to compute Riemannian gradients and Hessians

Selected References

- Absil, P.-A., R. Mahony, and R. Sepulchre (2008). *Optimization Algorithms on Matrix Manifolds*. Princeton University Press. DOI: 10.1515/9781400830244.
- RB and R. Herzog (2019). “Intrinsic formulation of KKT conditions and constraint qualifications on smooth manifolds”. In: *SIAM Journal on Optimization* 29.4, pp. 2423–2444. DOI: 10.1137/18M1181602. arXiv: 1804.06214.
- Boumal, N. (2023). *An introduction to optimization on smooth manifolds*. Cambridge University Press. URL: https://www.nicolasboumal.net/book.
- do Carmo, M. P. (1992). *Riemannian Geometry*. Mathematics: Theory & Applications. Boston, MA: Birkhäuser Boston, Inc.
- Huang, W., P.-A. Absil, and K. A. Gallivan (2018). “A Riemannian BFGS method without differentiated retraction for nonconvex optimization problems”. In: *SIAM Journal on Optimization* 28.1, pp. 470–495. DOI: 10.1137/17M1127582.
- Lee, J. M. (2018). *Introduction to Riemannian Manifolds*. Springer International Publishing. DOI: 10.1007/978-3-319-91755-9.
- Liu, C. and N. Boumal (Mar. 2019). “Simple algorithms for optimization on Riemannian manifolds with constraints”. In: *Applied Mathematics & Optimization*. DOI: 10.1007/s00245-019-09564-3.

NTNU

Selected Software References

Axen, S. D., M. Baran, RB, and K. Rzecki (2021). *Manifolds.jl: An Extensible Julia Framework for Data Analysis on Manifolds*. arXiv: 2106.08777.

RB (2022). "Manopt.jl: Optimization on Manifolds in Julia". In: *Journal of Open Source Software* 7.70, p. 3866. DOI: 10.21105/joss.03866.

Boumal, N., B. Mishra, P.-A. Absil, and R. Sepulchre (2014). "Manopt, a Matlab toolbox for optimization on manifolds". In: *The Journal of Machine Learning Research* 15, pp. 1455–1459. URL: https://www.jmlr.org/papers/v15/boumal14a.html.

Miolane, N. et al. (2020). "Geomstats: A Python Package for Riemannian Geometry in Machine Learning". In: *Journal of Machine Learning Research* 21.223, pp. 1–9. URL: http://jmlr.org/papers/v21/19-027.html.

Townsend, J., N. Koep, and S. Weichwald (2016). "Pymanopt: A Python Toolbox for Optimization on Manifolds using Automatic Differentiation". In: *Journal of Machine Learning Research* 17.137, pp. 1–5. URL: http://jmlr.org/papers/v17/16-177.html.

ronnybergmann.net/talks/2023-Oslo-Intro-Manopt.pdf