Optimization on Riemannian Manifolds in Julia

Ronny Bergmann

Robotics and Automation group research seminar

Trondheim, November 16, 2023

NTNU
Norges teknisk-naturvitenskapelige universitet

Plan for today

- Motivation – Why Optimize on Manifolds?
- Software – Why Julia and How to get started?
- An Example – The Difference of Convex Algorithm

Motivation

The Rayleigh Quotient

When minimizing the Rayleigh quotient for a symmetric $A \in \mathbb{R}^{n\times n}$

$$\operatorname*{arg\,min}_{x\in\mathbb{R}^n\setminus\{0\}} \frac{x^\mathrm{T}Ax}{\|x\|^2}$$

- Any eigenvector x^* to the smallest EV λ is a minimizer
- no isolated minima and Newton's method diverges
- Constrain the problem to unit vectors $\|x\| = 1$!

classic constrained optimization (ALM, EPM,...)

Today Utilize the geometry of the sphere

- unconstrained optimization $\operatorname*{arg\,min}_{p\in\mathbb{S}^{n-1}} p^\mathrm{T}Ap$
- adapt unconstrained optimization to Riemannian manifolds.

The Generalized Rayleigh Quotient

More general. Find a basis for the space of eigenvectors to $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_k$:

$$\operatorname*{arg\,min}_{X \in \mathrm{St}(n,k)} \mathrm{tr}(X^\mathsf{T} A X), \qquad \mathrm{St}(n,k) := \left\{ X \in \mathbb{R}^{n\times k} \,\middle|\, X^\mathsf{T} X = I \right\},$$

a problem on the Stiefel manifold $\mathrm{St}(n,k)$

Invariant under rotations within a k-dim subspace.

Find the best subspace!

$$\operatorname*{arg\,min}_{\mathrm{span}(X) \in \mathrm{Gr}(n,k)} \mathrm{tr}(X^\mathsf{T} A X), \qquad \mathrm{Gr}(n,k) := \left\{ \mathrm{span}(X) \,\middle|\, X \in \mathrm{St}(n,k) \right\},$$

a problem on the Grassmann manifold $\mathrm{Gr}(n,k) = \mathrm{St}(n,k)/O(k)$.

Optimization on Riemannian Manifolds

We are looking for numerical algorithms to find

$$\operatorname*{arg\,min}_{p\in\mathcal{M}} f(p)$$

where

- $\mathcal{M}$ is a Riemannian manifold
- $f\colon \mathcal{M} \to \overline{\mathbb{R}}$ is a function

⚠ f might be nonsmooth and/or nonconvex

⚠ $\mathcal{M}$ might be high-dimensional

A Riemannian Manifold $\mathcal{M}$

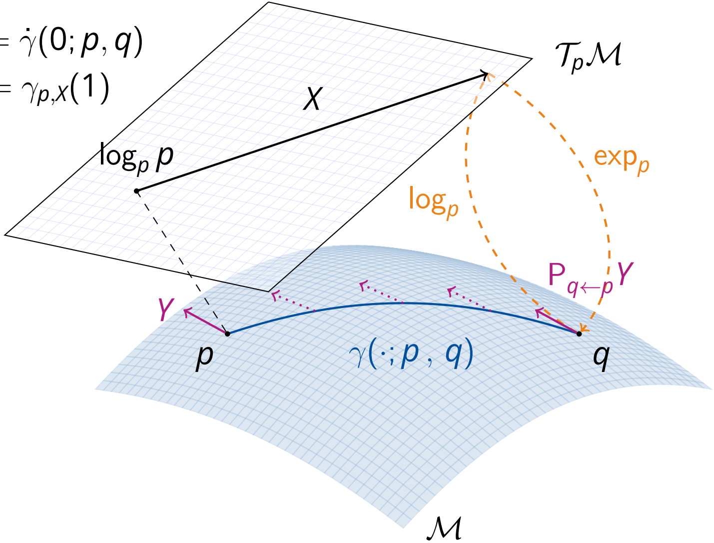
A d-dimensional Riemannian manifold can be informally defined as a set $\mathcal{M}$ covered with a "suitable" collection of charts, that identify subsets of $\mathcal{M}$ with open subsets of $\mathbb{R}^d$ and a continuously varying inner product on the tangent spaces.

[Absil, Mahony, and Sepulchre 2008]

A Riemannian Manifold $\mathcal{M}$

Notation.

- Logarithmic map $\log_p q = \dot\gamma(0; p, q)$
- Exponential map $\exp_p X = \gamma_{p,X}(1)$
- Geodesic $\gamma(\cdot; p\,,\, q)$
- Tangent space $\mathcal{T}_p\mathcal{M}$
- inner product $(\cdot\,,\,\cdot)_p$
- parallel transport $\mathcal{P}_{q\leftarrow p}X$



Software

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.

ManifoldsBase.jl – Motivation

Goal. Provide a generic interface to manifolds for

- defining own (new) manifolds
- implementing generic algorithms on an arbitrary manifold $\mathcal{M}$

A Manifold. a Riemannian manifold is a subtype of `AbstractManifold{𝔽}`

- $\mathbb{F} \in \{\mathbb{R}, \mathbb{C}, \mathbb{H}\}$: field the manifold is build on
- stores all "general" information, (mainly) the manifold dimension
- example (form `Manifolds.jl`): `M = Sphere(2)`

Points and Tangent vectors.

- by default not typed, often `<:AbstractArray`
- we provide `<:AbstractManifoldPoint` and `<:TVector` for more advanced ones

ManifoldsBase.jl – Functions

Goal. Efficient and reusable functions.

Functions. We provide functions like

- `exp(M, p, X)`, `log(M, p, q)`, `inner(M, p, X, Y)`, `parallel_transport(M, p, X, q)`
- defaults, for example `norm(M, p, X)` or `shortest_geodesic(M, p, q)`
- `retract(M, p, X, method)` to approx. $\exp_p X$, with different `method`s,
- similarly `inverse_retract(M, p, q, m)` and `vector_transport(M, p, X, q, m)`

Efficient. For all functions we design

- `exp!(M, q, p, X)` to work in-place of `q`
- `exp(M, p, X)` allocates and falls back to `exp!`

$\Rightarrow$ only one implementation, avoiding memory allocation where possible

ManifoldsBase.jl – Beyond functions

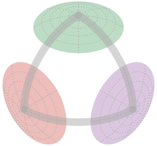
Decorators. A manifold can be decorated

- with an embedding, e. g. $\mathbb{S}^2 \subset \mathbb{R}^3$, to pass implementation (`inner(M, p, X, Y)`) to the embedding
- Specify more than one metric

Generic Manifolds. The interface provides generic (meta) manifolds like

- `TpM = TangentSpace(M,p)` $T_p\mathcal{M}$
- `M = ProductManifold(N1,N2)` for $\mathcal{M} = \mathcal{N}_1 \times \mathcal{N}_2$, short: `M = N1×N2`
- `M = PowerManifold(N,k)` for $\mathcal{M} = \mathcal{N}^k$, short: `M = N^k` or even `M = N^(k,l)`

Manifolds.jl



Goal. Provide a library of Riemannian manifolds, that is efficiently implemented and well-documented

[Axen, Baran, RB, and Rzecki 2023]

Euclidean. $\mathbb{F}^{d_1 \times d_2 \times d_3 \times \cdots}$, $\mathbb{F} \in \{\mathbb{R}, \mathbb{C}, \mathbb{H}\}$

Matrices.

- centered, symmetric, skew-symmetric
- symmetric positive definite
- (sym. pos. semidef.) fixed rank
- multinomial, multinom. sym.
- multilin. doubly stochastic
- unit norm, symmetric, symplectic

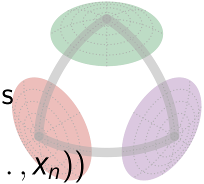
Groups. (incl. product & power groups)

- SO(n), SE(n), SU(n)
- (General, Special) Linear
- Heisenberg, circle, translation

Furthermore.

- circle, torus, (Array) sphere, oblique
- essential manifold, elliptope, flag
- (generalized, symplectic) Stiefel
- (generalized) Grassmann
- hyperbolic space & Lorentzian
- Kendall's (pre) shape space
- positive numbers
- probability simplex
- projective space
- rotations
- Tucker

Generic implementation of Bézier curves



Idea. Generalize de Casteljau's algorithm for $x_0, \ldots, x_n \in \mathbb{R}^n$ as

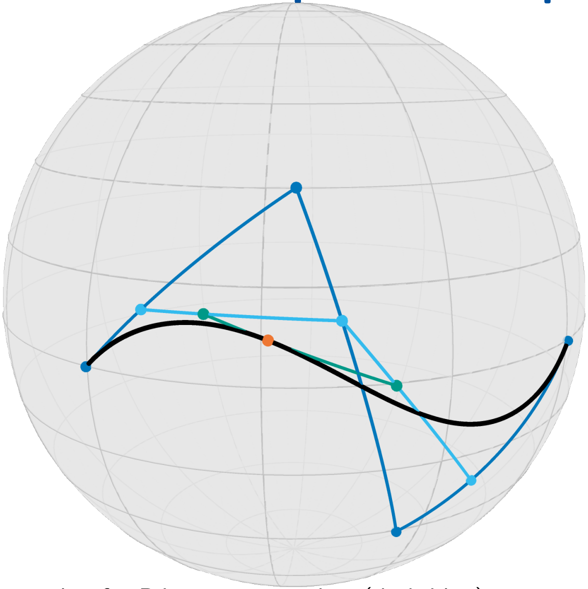
$$
\begin{aligned}
b_n(t; x_0, \ldots, x_n) &= b_1(t; b_{n-1}(t; x_0, \ldots, x_{n-1}), b_{n-1}(t; x_1, \ldots, x_n))\\
b_1(t; x_0, x_1) &= x_0 + t(x_1 - x_0)
\end{aligned}
$$

by replacing the straight line $b_1(\cdot; a, b)$ by shortest geodesics $\gamma(t; p, q)$

[Gousenbourger, Massart, and Absil 2018; RB and Gousenbourger 2018]
[Axen, Baran, RB, and Rzecki 2023]

```
function bezier(M::AbstractManifold, t, pts::NTuple)
    p = bezier(M, t, pts[1:(end - 1)])
    q = bezier(M, t, pts[2:end])
    return shortest_geodesic(M, p, q, t)
end
function bezier(M::AbstractManifold, t, pts::NTuple{2})
    return shortest_geodesic(M, pts[1], pts[2], t)
end
```

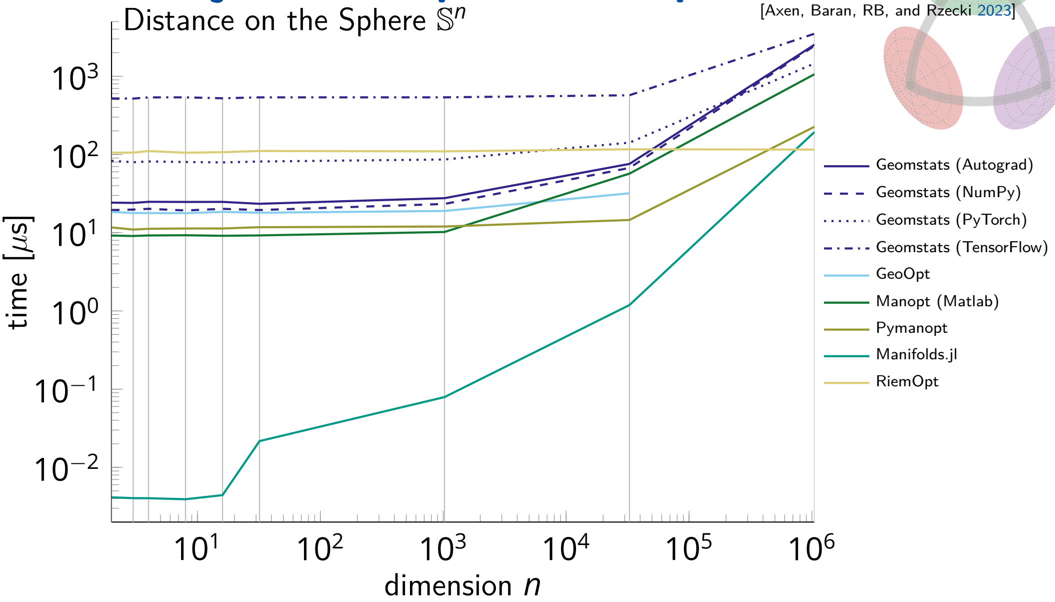
Bezier Curves – An example on the sphere



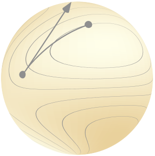
An example of a Bézier curve with 4 (dark blue) points on $\mathbb{S}^2$.

Manifolds.jl – A comparison in speed

[Axen, Baran, RB, and Rzecki 2023]



Manopt.jl



Goal. Provide optimization algorithms on Riemannian manifolds.

Features. Given a `Problem p` and a `SolverState s`,
implement `initialize_solver!(p, s)` and `step_solver!(p, s, i)`
$\Rightarrow$ an algorithm in the `Manopt.jl` interface

Highlevel interface like `gradient_descent(M, f, grad_f)`
on any manifold `M` from `Manifolds.jl`.

Provide debug output, recording, cache & counting capabilities,
as well as a library of step sizes and stopping criteria.

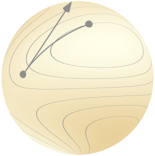
Manopt family.

manoptjl.org [RB 2022]

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

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

Manopt.jl



Algorithms.

Cost-based Nelder-Mead, Particle Swarm

Subgradient-based Subgradient Method

Gradient-based Gradient Descent, Conjugate Gradient, Stochastic,
Momentum, Nesterov, Averaged, …
Quasi-Newton: (L-)BFGS, DFP, Broyden, SR1,…

Hessian-based Trust Regions, Adaptive Regularized Cubics (ARC)

nonsmooth Chambolle-Pock, Douglas-Rachford, Cyclic Proximal Point

constrained Augmented Lagrangian, Exact Penalty, Frank-Wolfe

nonconvex Difference of Convex Algorithm, DCPPA

manoptjl.org

Calling a Manopt Solver: Gradient Descent

[Axen, Baran, RB, and Rzecki 2023]

Let's compute the Riemannian Center of Mass (mean) on the Sphere[1].

```julia
using Manopt, Manifolds, LinearAlgebra
M = Sphere(2)
N = 100

# generate random points on M
pts = [normalize(randn(3)) for _ in 1:N]

# define the loss function and its gradient
f(M,q) = sum(p -> distance(M, p, q)^2 / 2N, pts)
grad_f(M,q) = sum(p -> grad_distance(M, p, q) / N, pts)

# compute the mean
p_mean = gradient_descent(M, f, grad_f, pts[1])
```

[1] cf. https://manoptjl.org/stable/solvers/gradient_descent/

The Difference of Convex Algorithm

Difference of Convex

We aim to solve

$$\operatorname*{arg\,min}_{p \in \mathcal{M}} f(p)$$

where

- $\mathcal{M}$ is a Riemannian manifold
- $f\colon \mathcal{M} \to \mathbb{R}$ is a difference of convex function, i. e. of the form

$$f(p) = g(p) - h(p)$$

- $g, h\colon \mathcal{M} \to \overline{\mathbb{R}}$ are convex, lower semicontinuous, and proper

The Euclidean DCA

Idea 1. At x_k, approximate $h(x)$ by its affine minorization $h_k(x) := h(x^{(k)}) + \langle x - x^{(k)}, y^{(k)}\rangle$ for some $y^{(k)} \in \partial h(x^k)$.

$\Rightarrow$ minimize $g(x) - h_k(x) = g(x) + h(x^{(k)}) - \langle x - x^{(k)}, y^{(k)}\rangle$ instead.

Idea 2. Using duality theory finding a new $y^{(k)} \in \partial h(x^{(k)})$ is equivalent to

$$y^{(k)} \in \operatorname*{arg\,min}_{y\in\mathbb{R}^n}\Bigl\{h^*(y) - g^*(y^{(k-1)}) - \langle y - y^{(k-1)}, x^{(k)}\rangle\Bigr\}$$

Idea 3. Reformulate 2 using a proximal map $\Rightarrow$ DCPPA

On manifolds: [Almeida, Neto, Oliveira, and Souza 2020; Souza and Oliveira 2015]

In the Euclidean case, all three models are equivalent.

Derivation of the Riemannian DCA

We consider the linearization of h at some point $p^{(k)}$:
With $\xi \in \partial h(p^{(k)})$ we get

$$h_k(p) = h(p^{(k)}) + \langle \xi \,, \log_{p^{(k)}} p \rangle_{p^{(k)}}$$

Using musical isomorphisms we identify $X = \xi^\sharp \in T_p\mathcal{M}$,
where we call X a subgradient. Locally h_k minorizes h, i. e.

$$h_k(q) \leq h(q) \text{ locally around } p^{(k)}$$

$\Rightarrow$ Use $-h_k(p)$ as upper bound for $-h(p)$ in f.

Note. On $\mathbb{R}^n$ the function h_k is linear.
On a manifold h_k is not necessarily convex, even on a Hadamard manifold.

The Riemannian DC Algorithm

[RB, Ferreira, Santos, and Souza 2023]

Input: An initial point $p^0 \in \operatorname{dom}(g)$, g and $\partial_{\mathcal{M}} h$

1: Set $k = 0$.
2: **while** not converged **do**
3: Take $X^{(k)} \in \partial_{\mathcal{M}} h(p^{(k)})$
4: Compute the next iterate p^{k+1} as

$$p^{(k+1)} \in \operatorname*{arg\,min}_{p \in \mathcal{M}} g(p) - \left(X_k\,,\, \log_{p^{(k)}} p\right)_{p^{(k)}}. \tag{$*$}$$

5: Set $k \leftarrow k + 1$
6: **end while**

Note. In general the subproblem $(*)$ can not be solved in closed form. But an approximate solution yields a good candidate.

Convergence of the Riemannian DCA

[RB, Ferreira, Santos, and Souza 2023]

Let $\{p^{(k)}\}_{k\in\mathbb{N}}$ and $\{X^{(k)}\}_{k\in\mathbb{N}}$ be the iterates and subgradients of the RDCA.

Theorem.
If $\bar{p}$ is a cluster point of $\{p^{(k)}\}_{k\in\mathbb{N}}$, then $\bar{p} \in \operatorname{dom}(g)$ and there exists a cluster point $\bar{X}$ of $\{X^{(k)}\}_{k\in\mathbb{N}}$ s. t. $\bar{X} \in \partial g(\bar{p}) \cap \partial h(\bar{p})$.

$\Rightarrow$ Every cluster point of $\{p^{(k)}\}_{k\in\mathbb{N}}$, if any, is a critical point of f.

Proposition. Let g be σ-strongly (geodesically) convex. Then

$$f(p_{k+1}) \leq f(p^{(k)}) - \frac{\sigma}{2} d^2(p^{(k)}, p_{k+1}).$$

and $\displaystyle\sum_{k=0}^{\infty} d^2(p^{(k)}, p^{(k+1)}) < \infty$, so in particular $\displaystyle\lim_{k\to\infty} d(p^{(k)}, p^{(k+1)}) = 0$.

Implementation of the DCA

The algorithm is implemented and released in Julia using `Manopt.jl`[2]. It can be used with any manifold from `Manifolds.jl`

A solver call looks like

```
q = difference_of_convex_algorithm(M, f, g, ∂h, p0)
```

where one has to implement `f(M, p)`, `g(M, p)`, and `∂h(M, p)`.

- a sub problem is automatically generated
- an efficient version of its cost and gradient is provided
- you can specify the sub-solver to using `sub_state=` to also set up the specific parameters of your favourite algorithm

[2] see `https://manoptjl.org/stable/solvers/difference_of_convex/`

Summary.

- We considered Optimization on Riemannian Manifolds $\operatorname*{arg\,min}_{p \in \mathcal{M}} f(p)$.
- `ManifoldsBase.jl` is an Interface in Julia for Riemannian manifolds
- `Manifolds.jl` is a library of fast implementations of manifolds
- `Manopt.jl` provides optimization algorithms on manifolds
- We saw the Difference of Convex algorithm as an example

Further.

- `ManifoldDiff.jl` couples AD tools with differential geometry
- `ManoptExamples.jl` provides examples and their code
- `ManifoldDiffEq.jl` (first steps to) solving differential equations on manifolds

See juliamanifolds.github.io for further details on these.

Selected References

Axen, S. D., M. Baran, RB, and K. Rzecki (2023). "Manifolds.jl: An Extensible Julia Framework for Data Analysis on Manifolds". In: *ACM Transactions on Mathematical Software*. Accepted for pulication. DOI: 10.1145/3618296. 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.

RB, O. P. Ferreira, E. M. Santos, and J. C. d. O. Souza (2023). *The difference of convex algorithm on Hadamard manifolds*. arXiv: 2112.05250.

RB and P.-Y. Gousenbourger (2018). "A variational model for data fitting on manifolds by minimizing the acceleration of a Bézier curve". In: *Frontiers in Applied Mathematics and Statistics*. DOI: 10.3389/fams.2018.00059. arXiv: 1807.1009.

Boumal, N. (2023). *An introduction to optimization on smooth manifolds*. Cambridge University Press. URL: https://www.nicolasboumal.net/book.

Souza, J. C. d. O. and P. R. Oliveira (2015). "A proximal point algorithm for DC fuctions on Hadamard manifolds". In: *Journal of Global Optimization* 63.4, pp. 797–810. DOI: 10.1007/s10898-015-0282-7.

Interested in Numerical Differential Geometry? Join numdiffgeo.zulipchat.com!

ronnybergmann.net/talks/2023-Trondheim-Manopt.pdf