Contents

Geometric optimization: convex and nonconvex In this talk, I will highlight some aspects of geometry and its role in optimization. In particular, I will talk about optimization problems whose parameters are constrained to lie on a manifold or in a specific metric space. These geometric constraints often make the problems numerically challenging, but they can also unravel properties that ensure tractable attainment of global optimality for certain otherwise non-convex problems.

We'll make our foray into geometric optimization via geodesic convexity, a concept that generalizes the usual notion of convexity to nonlinear metric spaces such as Riemannian manifolds. I will outline some of our results that contribute to g-convex analysis as well as to the theory of first-order g-convex optimization. I will mention several very interesting optimization problems where g-convexity proves remarkably useful. Time permitting, I will mention extensions to stochastic (non-convex) geometric optimization as well as some important open problems.

When we have geometric structure, we can leverage that structure to build math models and do optimization.

I'll share some examples.

What do I mean by geometry?

1. Vector spaces
2. Manifolds: parameters live on curved space like Riemannian manifold.
3. Convex sets: ex. convex cones. They enjoy vector space and other properties.
4. Metric spaces

I focus mostly on (Riemannian) manifolds. There is some nonlinear constraint, ex. orthogonality constraint, fixed-rank constraint, positive semi-definite constraint (you can view this as a nice manifold, not just a cone). These correspond to Stiefel, Grassmann, and PSD manifolds.

Classes of functions in optimization:

1. Convex

2. Lipschitz
3. Strongly convex
4. Smooth

We may have similar cost functions but instead of satisfying this on a vector space, they satisfy it on a manifold. We call this "geodesically convex", etc. This works in a larger class of metric spaces.

Parametrize the shortest path between 2 points.

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

On a Riemannian manifold

$$f(y) \geq f(x) + \left\langle g_x, \operatorname{Exp}_x^{-1}(y) \right\rangle_x.$$

This is a rich and beautiful concept.

Let's look at some examples.

On the manifold of PSD matrices, the geodesic is

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

(Noncommutative geometric matrix mean.) Examples of convex functions are $\ln\det(X)$, $\ln\operatorname{tr}(X)$, $\operatorname{tr}(X^\alpha)$, $\|X^\alpha\|$, $\alpha > 0$. The standard metric is the second derivative of $\ln\det(X)$.

Ex. determinant is multiplicative; we have equality.

Is this a local concept? For PSD, we have unique geodesics. Then geodesic is a local concept. Sometimes by nonconvex I mean I don't have global geodesic convexity.

Corollary:

$$X \mapsto \ln\det(B + \sum_i A_i^* X A_i) \tag{1}$$

$$X \mapsto \ln\operatorname{perm}(B + \sum_i A_i^* X A_i). \tag{2}$$

(Usually ln det is concave!) Geometric programming (see Stephen Boyd). We lose commutativity; not everything that holds in GP holds for us.

Examples.

1 Matrix square root

Matrix square root (of PSD matrices). This is a fundamental object. See Functions of Matrices, Nick Hayem.

I'll show both the Euclidean and non-Euclidean thinking.

Nonconvex optimization through Euclidean lens:

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

Gradient descent

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

This doesn't require computing the inverse. This converges to the square root, nontrivial analysis.

Geodesic:

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

Nonconvex optimization through non-Euclidean lens: Minimize sum of squared distances

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

where

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

JS divergence between multivariate Gaussian. This is convex on the manifold! Upshot: if I can find a stationary point, it is globally optimal.

This is open set. Create derivative, set to 0.

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

Fixed point iteration, linear convergence. Get global optimality thanks to geodesic convexity.

Show the fixed point is the midpoint on the geodesic. Fixed point iteration is much faster! Despite the inverses, flop count is smaller.

Other method requires careful step size selection. (Careful with underflow, etc.) There's no step selection here. (People in numerical linear algebra have even better algorithms. Matlab uses a small modification of a Newton method.)

2 Metric learning

Metric learning. Points that are similar should lie closer, if you measure distance according to the right distance function.

Given pairwise constraints S, D of pairs in the same and different classes, learn Mahalanobis distance

$$d_A(x, y) = (x - y)^T A(x - y).$$

Methods: MMC, LMNN, ITML. Cyclic projection, subgradient, etc.

Euclidean idea:

$$\min_{A \succeq 0} \sum_{(x_i, x_j) \in S} d_A(x_i, x_j) - \lambda \sum_{(x_i, x_j) \in D} d_A(x_i, x_j).$$

Observation: rather than do this,

$$\min_{A \succeq 0} \sum_{(x_i, x_j) \in S} d_A(x_i, x_j) + \sum_{(x_i, x_j) \in D} d_{A^{-1}}(x_i, x_j)$$

Inverse is distance reversing. Use A^{-1} for the dissimilar points. Equivalently solve: collect into scatter matrix $S = \sum_S (x_i - x_j)(x_i - x_j)^T$.

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

This is convex!

We have closed form solution (Riccati equation)

$$\nabla h(A) = 0 \iff S - A^{-1}DA^{-1} = 0$$

Solution

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

More generally, similar and dissimilar points are not equally valuable, $S^{-1}\#_t D$.

Most of the time it is competitively. Thanks to being closed form, you get huge speedups. This is purely geometric thinking.

I like to view this geometric way of doing things as "Supervised whitening transform."

3 Brascamp-Lieb inequalities

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

D is an inf of ratio of determinants, nonconvex. But it is geodesically convex! This arises in geometric complexity theory (noncommutative circuits)

$$\min_{X_1,\ldots,X_m \succ 0} \ln\det\left(\sum_i p_i B_i^* X_i B_i\right) - \sum_i p_i \ln\det X_i.$$

4 Gaussian mixture models

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

Find $\max \prod_i p_{mix}(x_i)$. EM is default choice.

Difficulty: positive definiteness constraint on Σ_k. PSD constraint makes use of general nonlinear method hard. It's hard to beat EM. Any blanket statement about which optimization method is best is always wrong.

Unconstrained reformulation, write as LL^T (folklore).

Geometric optimization. EM folks were right, trying to solve on manifold sucks.

`www.manopt.org`

Insight: one thing that makes EM work well is that the M-step is a convex optimization problem (with closed-form solution). Manifold optimization isn't using geometry correctly. Log-likelihood for 1 component is convex Euclideanly but not on manifold. Reformulate so convex on manifold. Do a natural change of parameters. 1-component problem becomes convex along manifold. Local min of one implies local min of other.

Now with that reformulation manifold methods outperform EM and have less variation in runtime.

Why? Convexity helps make things better conditioned? (I haven't defined conditioning on manifold.)

`github.com/utvisionlab/mixest`.

5 First-order algorithms

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

$x \leftarrow\!\shortmid \exp_x(-\eta\nabla f(x))$. (For practice approximate with detraction.)

There's a fairly complete theory on iteration complexity.

We want a similar theory for optimizing on manifolds. As curvature goes to 0 you should recover Euclidean method.

Core components.

1. Once you start analyzing $x_{t+1} = x_t - \eta_t g_t$, after making an update, how far is the next point from the optimal? The first thing to use is the law of cosines. Do stuff to that. This was our first stumbling point: we don't have access to the law of cosines!

   We had a corresponding inequality to bound $d^2(x_{t+1}, x^*)$ on manifolds.

   There is a hyperbolic cosine law for constant negative curvature.

   $$\cosh(-\kappa a) = \cosh(-\kappa b)\cosh(-\kappa c) + \sinh(-\kappa b)\sinh(-\kappa c)\cos(A).$$

2. Toponogov's theorem: Take triangles in curved space with variable but bounded curvature, compare what they look like to triangles in constant negative curvature.

3. Gr$\acute{}$onwall's theorem.

Reach an inequality which relates sides in nice way

$$a^2 \leq b^2 + \zeta(\kappa_{\min}, b)c^2 - 2bc\cos(A) \tag{3}$$

$$\zeta(\kappa_{\min}, b) := \frac{\sqrt{|\kappa_{\min}|}b}{\tanh(\sqrt{|\kappa_{\min}|}b)} \tag{4}$$

This can be generalized beyond manifolds.

When curvature is negative, because triangles get stretched you slow down in convergence.

6 Teasers

I close with a teaser. We've generalized to more fancy setups: ERM on manifold

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

If you're training a RNN, deep, they suffer from vanishing/exploding gradient. Use orthogonal matrices as weight vectors to prevent. They may use methods on optimization on manifolds. You can do variance reduction methods.

This helps in classical problems too! Stochastic eigenvector computation:

$$\min_{x^Tx} -x^T\left(\sum_{i=1}^{n} z_i z_i^T\right)x.$$

One can show this stoc hastic eigenvector problem satisfies beautiful geometric inequalities. You can analyze global optimality in half a page! (Zhang, Reddi, Sra, NIPS 2016).

Curved coordinate analogue of ordinary gradient descent. What is the curved coordinate analogue of mirror descent? One can view mirror descent as gradient descent on manifold using information geometry but that's a different thing.

It's trivial to do online parallel distributed versions. Scalability: as covariances are larger, linear algebra complexity is smaller for EM, but manifold still work better. The more Gaussians mix, the more EM suffers, intrinsic unidentifiability. This method is more robust, suffers less.