SAND2008-5009P

Optimization on Manifolds: Problems and Solutions

Christopher G. Baker

Scalable Algorithms Department
Sandia National Laboratories

June 10, 2008
CSRI Summer Lecture Series

Acknowledgments

Thanks

- Denis Ridzal
- CSRI

Collaborators

- Pierre-Antoine Absil, Université catholique de Louvain
- Kyle Gallivan, Florida State University

Funding

- NSF Grants OCI0324944 and CCR9912415
- School of Computational Science, FSU
- Sandia National Laboratories

Outline

Motivating Problems
- Pose Estimation
- Face/Object Recognition
- Other Problems

Riemannian Optimization
- Euclidean versus Riemannian Optimization
- Components of Riemannian Manifolds
- Retraction-based Riemannian Optimization

Riemannian Optimization Methods
- Riemannian Newton Method
- Riemannian Trust-Region Method
- Riemannian Direct Search

Example #1: Pose Estimation Problem

Problem description

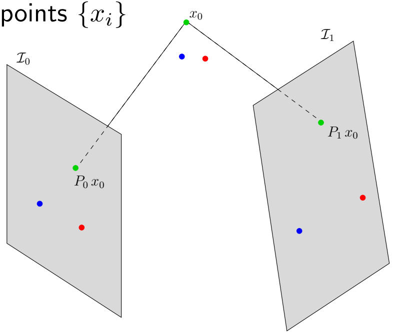
Given: a set of images $\{\mathcal{I}_j\}$ and identifications between feature points $\{x_i\}$ and their corresponding image points $\{P_j\{x_i\}\}$

Task: find the projections $\{P_j\}$ determining the pose of each camera

Bonus: find the 3-D location of the feature points $\{x_i\}$

Applications

- recover motion of camera
- recover structure of 3-D scene from 2-D images
- allow augmentation of scene with virtual objects



Problem Setting

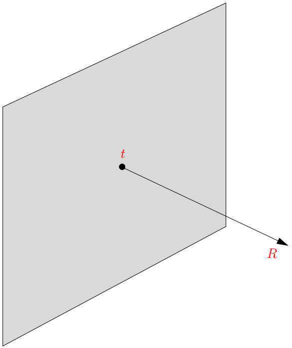
Camera Parameters

Determining the orientation of the camera amounts to finding

- the center of the camera in 3-D space
- the direction it is pointing

This amounts to finding

- a translation vector $t \in \mathbb{R}^3$
- a rotation matrix $R \in \mathrm{SO}(3)$
  - R is orthogonal
  - $\det(R) = +1$



Difficulties

It is not possible to find an analytic/exact solution to this problem:

- errors in the point correspondence algorithm
- problem matching discrete pixels against points in continuous space

Problem Setting

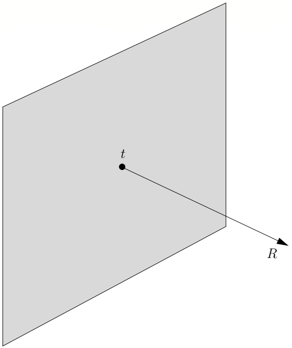
Camera Parameters

Determining the orientation of the camera amounts to finding

- the center of the camera in 3-D space
- the direction it is pointing

This amounts to finding

- a translation vector $t \in \mathbb{R}^3$
- a rotation matrix $R \in \mathrm{SO}(3)$
  - R is orthogonal
  - $\det(R) = +1$



Difficulties

It is not possible to find an analytic/exact solution to this problem:

- errors in the point correspondence algorithm
- problem matching discrete pixels against points in continuous space

Optimization Characterization

One approach to solving the problem is to apply an optimization algorithm:

$$\text{minimize } f(c_0, c_1, \ldots, c_{n-1})$$

where

- f is a measure of the error in the point correspondences
- $c_i \in \mathrm{SE}(3)$ are the coordinates for the i-th camera
- $\mathrm{SE}(3)$ is the special Euclidean group:

$$\mathrm{SE}(3) = \mathrm{SO}(3) \times \mathbb{R}^3$$

Riemannian Optimization Characterization

Our goal is the Riemannian optimization f:

$$f : \mathcal{M} \to \mathbb{R}$$

$$\mathcal{M} = \mathrm{SE}(3) \times \cdots \times \mathrm{SE}(3)$$

Optimization Characterization

One approach to solving the problem is to apply an optimization algorithm:

$$\text{minimize } f(c_0, c_1, \ldots, c_{n-1})$$

where

- f is a measure of the error in the point correspondences
- $c_i \in \mathrm{SE}(3)$ are the coordinates for the i-th camera
- $\mathrm{SE}(3)$ is the special Euclidean group:

$$\mathrm{SE}(3) = \mathrm{SO}(3) \times \mathbb{R}^3$$

Riemannian Optimization Characterization

Our goal is the Riemannian optimization f:

$$f : \mathcal{M} \to \mathbb{R}$$
$$\mathcal{M} = \mathrm{SE}(3) \times \cdots \times \mathrm{SE}(3)$$

Example #2: Face/Object Recognition

Problem description

Given an image I, identify the object/person in the image as a member of a set of known objects/people.

Difficulties

- Problem: images are often high-dimensional
- Solution: reduce the dimensionality of the images

Popular methods involve projecting the images onto a linear subspace:

Example #2: Face/Object Recognition

Problem description

Given an image I, identify the object/person in the image as a member of a set of known objects/people.

Difficulties

- Problem: images are often high-dimensional
- Solution: reduce the dimensionality of the images

Popular methods involve projecting the images onto a linear subspace:

Example #2: Face/Object Recognition

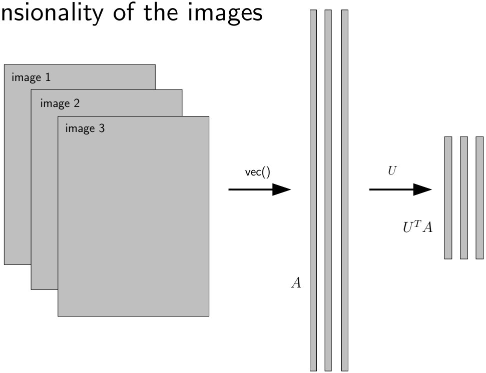
Problem description

Given an image I, identify the object/person in the image as a member of a set of known objects/people.

Difficulties

- Problem: images are often high-dimensional
- Solution: reduce the dimensionality of the images

Popular methods involve projecting the images onto a linear subspace:



PCA

- PCA chooses a basis U from the SVD of

$$A = \begin{bmatrix} \tilde{I}_0 & \tilde{I}_1 & \cdots & \tilde{I}_{n-1} \end{bmatrix}$$

- U is optimal in terms of minimizing the error

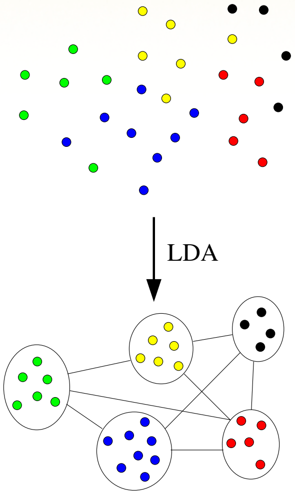
$$\|A - UU^T A\|_2$$

- Approach is motivated by the ability of U to capture the components of highest variance.
- U is computed via the SVD of A or the EVD of AA^T or $A^T A$.

Christopher DeCoro @ Princeton

LDA

- LDA chooses basis U as the vectors maximizing Fisher's linear discriminant.
- These vector maximize the distance between classes (e.g., people) while minimizing the distance inside classes.
- This basis is computed via a generalized eigenvalue or generalized SVD problem.



Optimal Basis Choice

What is Optimal?

- Both PCA and LDA choose bases that are optimal in some respect.
- However, neither is optimal with respect to recognition accuracy.
- Result: linear projection methods have a bad reputation.
- Before dismissing the entire class of methods, consider finding the optimal linear subspace with respect to recognition accuracy.

Riemannian Optimization Characterization

- Let $f(U)$ denote the recognition accuracy of the basis U.
- If f employs a nearest-neighbor classifier, then $f(U) = f(U \cdot M)$.
- Then f is a function over the Grassmann manifold:

$$\mathrm{Grass}(p, n) = \{\text{all } p\text{-dimensional subspaces of } \mathbb{R}^n\}$$

- Optimizing f over $\mathrm{Grass}(p, n)$ gives the optimal p-dimensional basis.

Optimal Basis Choice

What is Optimal?

- Both PCA and LDA choose bases that are optimal in some respect.
- However, neither is optimal with respect to recognition accuracy.
- Result: linear projection methods have a bad reputation.
- Before dismissing the entire class of methods, consider finding the optimal linear subspace with respect to recognition accuracy.

Riemannian Optimization Characterization

- Let $f(U)$ denote the recognition accuracy of the basis U.
- If f employs a nearest-neighbor classifier, then $f(U) = f(U \cdot M)$.
- Then f is a function over the Grassmann manifold:

$$\mathrm{Grass}(p, n) = \{\text{all } p\text{-dimensional subspaces of } \mathbb{R}^n\}$$

- Optimizing f over $\mathrm{Grass}(p, n)$ gives the optimal p-dimensional basis.

Significant Manifolds

Orthogonal Group

The manifold of orthogonal matrices:

$$\mathrm{O}(n) = \{U \in \mathbb{R}^{n \times n} : U^T U = U U^T = I\}$$

Compact Stiefel Manifold

The manifold of orthonormal bases:

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

Grassmann manifold

Manifold of linear subspaces:

$$\mathrm{Grass}(p, n) = \{p\text{-dimensional subspaces of } \mathbb{R}^n\}$$

Stiefel/Grassmann Applications

- dominant singular vectors of a matrix (Stiefel)

$$f(U,V) = \mathrm{trace}\left(U^T A V N\right)$$

- optimal-rank tensor factorization (Grassmann)

$$f(U,V,W) = \|A \bullet_1 U^T \bullet_2 V^T \bullet_3 W^T\|^2$$

- ICA, blind-source separation ("cocktail party problem") (Grassmann)
- eigenspaces of a generalized symmetric matrix pencil (Grassmann)

$$f(V) = \mathrm{trace}\left(\left(V^T B V\right)^{-1}\left(V^T A V\right)\right)$$

- computing H2-optimal reduced order models (Grassmann)

$$f(\hat{H}) = \|\hat{H}(s) - H(s)\|^2_{\mathcal{H}2}$$

Outline

Motivating Problems
- Pose Estimation
- Face/Object Recognition
- Other Problems

Riemannian Optimization
- Euclidean versus Riemannian Optimization
- Components of Riemannian Manifolds
- Retraction-based Riemannian Optimization

Riemannian Optimization Methods
- Riemannian Newton Method
- Riemannian Trust-Region Method
- Riemannian Direct Search

What is Riemannian Optimization?

Definition

Riemannian Optimization refers to the optimization of an objective function over a Riemannian manifold.

Objective

Given a Riemannian manifold $\mathcal{M}$ and a smooth function

$$f : \mathcal{M} \to \mathbb{R} \ ,$$

the goal is to find an extreme point:

$$\min_{x \in \mathcal{M}} f(x) \qquad \text{or} \qquad \max_{x \in \mathcal{M}} f(x)$$

Isn't this just constrained Euclidean optimization?

Euclidean vs. Riemannian

Euclidean	minimize $f : \mathbb{R}^n \to \mathbb{R}$
Constrained Euclidean	minimize $f : \mathcal{C} \subset \mathbb{R}^n \to \mathbb{R}$
Riemannian	minimize $f : \mathcal{M} \to \mathbb{R}$

Why bother with manifolds?

- You have no choice.
  - There may be no efficient embedding $\mathcal{M} \subset \mathbb{R}^n$.
- You don't like constrained optimization.
  - Riemannian optimization methods are feasible.
  - Riemannian optimization methods have "simpler" theory.

The difference

Riemannian optimization can be thought of as an unconstrained optimization in a constrained search space.

Isn't this just constrained Euclidean optimization?

Euclidean vs. Riemannian

Euclidean	minimize $f : \mathbb{R}^n \to \mathbb{R}$
Constrained Euclidean	minimize $f : \mathcal{C} \subset \mathbb{R}^n \to \mathbb{R}$
Riemannian	minimize $f : \mathcal{M} \to \mathbb{R}$

Why bother with manifolds?

- You have no choice.
  - There may be no efficient embedding $\mathcal{M} \subset \mathbb{R}^n$.
- You don't like constrained optimization.
  - Riemannian optimization methods are feasible.
  - Riemannian optimization methods have "simpler" theory.

The difference

Riemannian optimization can be thought of as an unconstrained optimization in a constrained search space.

Isn't this just constrained Euclidean optimization?

Euclidean vs. Riemannian

Euclidean	minimize $f : \mathbb{R}^n \to \mathbb{R}$
Constrained Euclidean	minimize $f : \mathcal{C} \subset \mathbb{R}^n \to \mathbb{R}$
Riemannian	minimize $f : \mathcal{M} \to \mathbb{R}$

Why bother with manifolds?

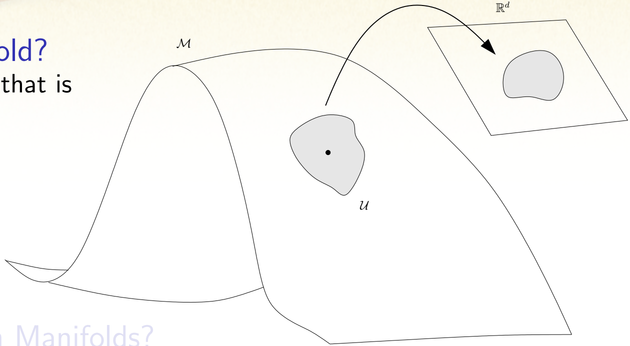
- You have no choice.
  - There may be no efficient embedding $\mathcal{M} \subset \mathbb{R}^n$.
- You don't like constrained optimization.
  - Riemannian optimization methods are feasible.
  - Riemannian optimization methods have "simpler" theory.

The difference

Riemannian optimization can be thought of as an unconstrained optimization in a constrained search space.

What is a Manifold?

A manifold is a set that is locally Euclidean.

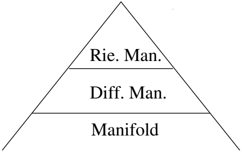


Why Riemannian Manifolds?

Riemannian manifold is a differentiable manifold with a Riemannian metric:

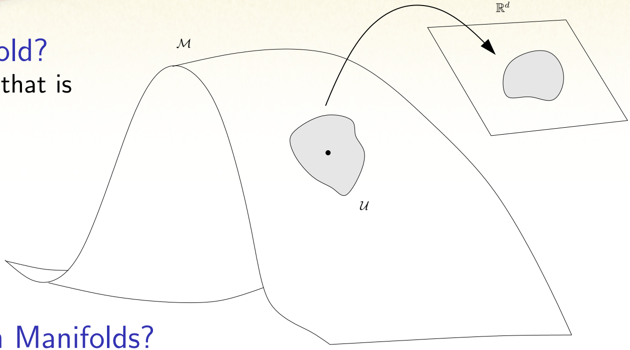
- The manifold gives us topology.
- Differentiability gives us calculus.
- The Riemannian metric gives us geometry.

Riemannian manifolds strike a balance between power and practicality.



What is a Manifold?

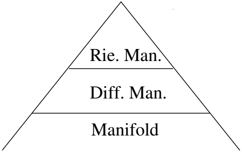
A manifold is a set that is locally Euclidean.



Why Riemannian Manifolds?

Riemannian manifold is a differentiable manifold with a Riemannian metric:

- The manifold gives us topology.
- Differentiability gives us calculus.
- The Riemannian metric gives us geometry.

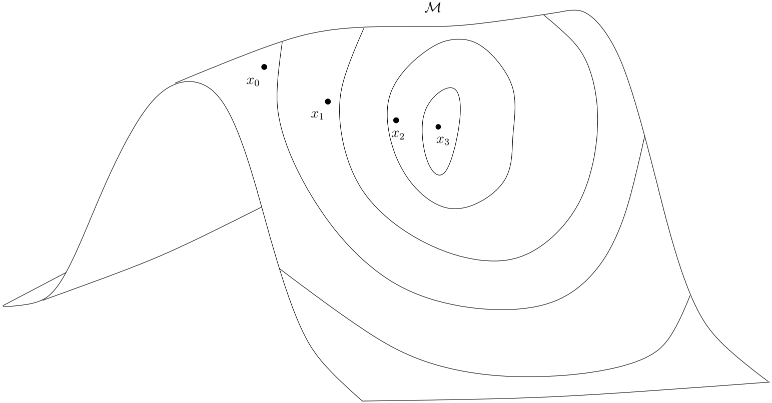


Riemannian manifolds strike a balance between power and practicality.

Iterative Methods

Goal

Given an objective function $f : \mathcal{M} \to \mathbb{R}$ and an initial iterate $x_0 \in \mathcal{M}$, construct a sequence $\{x_i\} \in \mathcal{M}$ which converges to a minimizer of f.



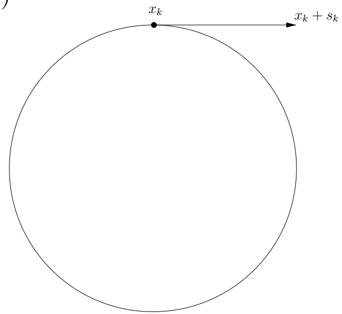
Iterations on the Manifold

Consider the following generic update for an iterative Euclidean optimization algorithm:

$$x_{k+1} = x_k + s_k \ .$$

This iteration is implemented in numerous ways, e.g.:

- Newton's method: $x_{k+1} = x_k - \alpha_k \left[\nabla^2 f(x_k)\right]^{-1} \nabla f(x_k)$
- Steepest descent: $x_{k+1} = x_k - \alpha_k \nabla f(x_k)$



We Need

- Riemannian concepts describing directions and movement on the manifold
- Riemannian analogues for gradient and Hessian

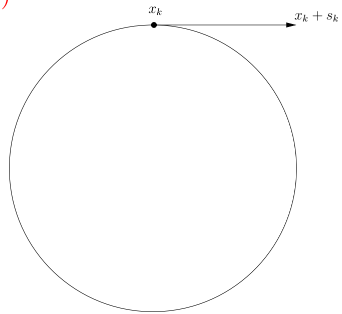
Iterations on the Manifold

Consider the following generic update for an iterative Euclidean optimization algorithm:

$$x_{k+1} = x_k + s_k \ .$$

This iteration is implemented in numerous ways, e.g.:

- Newton's method: $x_{k+1} = x_k - \alpha_k \left[\nabla^2 f(x_k)\right]^{-1} \nabla f(x_k)$
- Steepest descent: $x_{k+1} = x_k - \alpha_k \nabla f(x_k)$

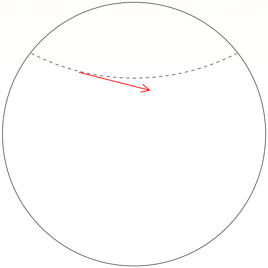


We Need

- Riemannian concepts describing directions and movement on the manifold
- Riemannian analogues for gradient and Hessian

Tangent Vectors

- The concept of direction is provided by tangent vectors.
- Intuitively, tangent vectors are tangent to curves on the manifold.
- Tangent vectors are an intrinsic property of a differentiable manifold.

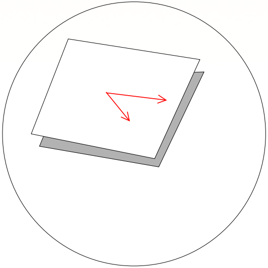


Definition

The tangent space $T_x\mathcal{M}$ is the vector space comprised of the tangent vectors at $x \in \mathcal{M}$. The Riemannian metric is an inner product on each tangent space.

Tangent Vectors

- The concept of direction is provided by tangent vectors.
- Intuitively, tangent vectors are tangent to curves on the manifold.
- Tangent vectors are an intrinsic property of a differentiable manifold.



Definition

The tangent space $T_x\mathcal{M}$ is the vector space comprised of the tangent vectors at $x \in \mathcal{M}$. The Riemannian metric is an inner product on each tangent space.

Riemannian gradient and Riemannian Hessian

Definition

The Riemannian gradient of f at x is the tangent vector in $T_x\mathcal{M}$ satisfying

$$\mathrm{D}\, f(x)[\eta] = \langle \operatorname{grad} f(x), \eta \rangle$$

Definition

The Riemannian Hessian of f at x is a symmetric linear operator from $T_x\mathcal{M}$ to $T_x\mathcal{M}$ defined as

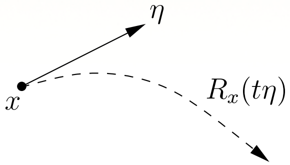
$$\operatorname{Hess} f(x)[\eta] = \mathrm{D} \operatorname{grad} f(x)[\eta]$$

Retractions

Definition

A retraction is a mapping R from $T\mathcal{M}$ to $\mathcal{M}$ satisfying the following:

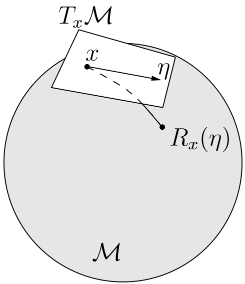
- R is continuously differentiable
- $R_x(0) = x$
- $\mathrm{D}\, R_x(0)[\eta] = \eta$



What is it good for?

- maps tangent vectors back to the manifold
- lifts objective function f from $\mathcal{M}$ to $T_x\mathcal{M}$, via the pullback

$$\hat{f}_x = f \circ R_x$$



Retraction-based Riemannian optimization

A novel optimization paradigm

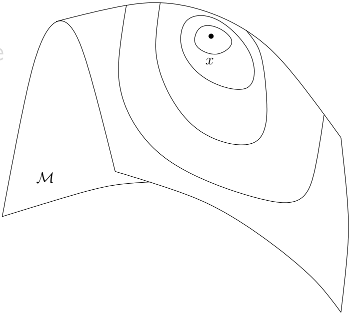
Q: How do we conduct optimization on a manifold?
A: We do it in the tangent spaces.

Benefits

- Can easily employ classical optimization techniques
- Less expensive than previous approaches
- Increased generality does not compromise the important theory

Sufficient Optimality Conditions

If $\text{grad}\,\hat{f}_x(0) = 0$ and $\text{Hess}\,\hat{f}_x(0) > 0$,
then $\text{grad}\,f(x) = 0$ and $\text{Hess}\,f(x) > 0$,
so that x is a local minimizer of f.



Retraction-based Riemannian optimization

A novel optimization paradigm

Q: How do we conduct optimization on a manifold?

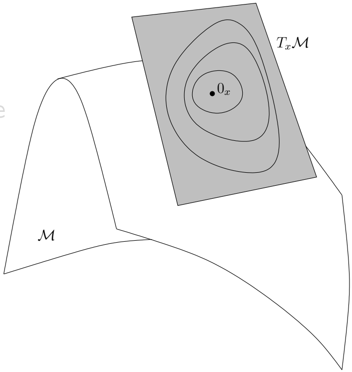
A: We do it in the tangent spaces.

Benefits

- Can easily employ classical optimization techniques
- Less expensive than previous approaches
- Increased generality does not compromise the important theory

Sufficient Optimality Conditions

If $\text{grad}\, \hat{f}_x(0) = 0$ and $\text{Hess}\, \hat{f}_x(0) > 0$,
then $\text{grad}\, f(x) = 0$ and $\text{Hess}\, f(x) > 0$,
so that x is a local minimizer of f.



Retraction-based Riemannian optimization

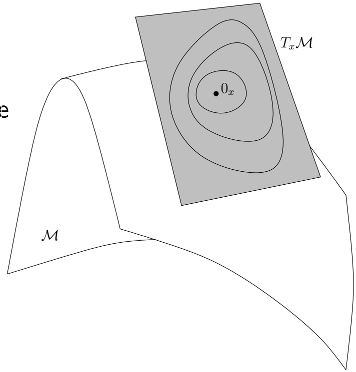
A novel optimization paradigm

Q: How do we conduct optimization on a manifold?

A: We do it in the tangent spaces.

Benefits

- Can easily employ classical optimization techniques
- Less expensive than previous approaches
- Increased generality does not compromise the important theory



Sufficient Optimality Conditions

If $\operatorname{grad} \hat{f}_x(0) = 0$ and $\operatorname{Hess} \hat{f}_x(0) > 0$,
then $\operatorname{grad} f(x) = 0$ and $\operatorname{Hess} f(x) > 0$,
so that x is a local minimizer of f.

Retraction-based Riemannian optimization

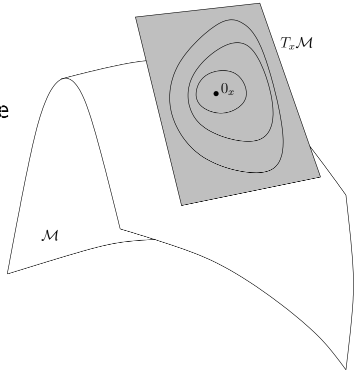
A novel optimization paradigm

Q: How do we conduct optimization on a manifold?

A: We do it in the tangent spaces.

Benefits

- Can easily employ classical optimization techniques
- Less expensive than previous approaches
- Increased generality does not compromise the important theory



Sufficient Optimality Conditions

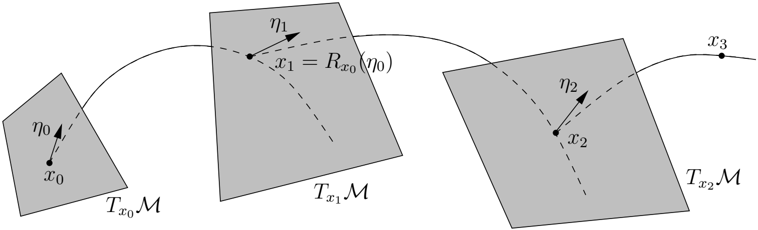
If $\mathrm{grad}\ \hat{f}_x(0) = 0$ and $\mathrm{Hess}\ \hat{f}_x(0) > 0$,
then $\mathrm{grad}\ f(x) = 0$ and $\mathrm{Hess}\ f(x) > 0$,
so that x is a local minimizer of f.

Generic Riemannian Optimization Algorithm

1. At iterate $x \in \mathcal{M}$, define $\hat{f}_x = f \circ R_x$.
2. Find minimizer η of $\hat{f}_x$.
3. Choose new iterate $x_+ = R_x(\eta)$.
4. Goto step 1.

A suitable setting

This paradigm is sufficient for describing numerous optimization methods.



Riemannian Newton Method

1a. At iterate x, define pullback $\hat{f}_x = f \circ R_x$

1. Find solution η of
$$\nabla^2 f(x)\,\eta = -\nabla f(x)$$
2. Choose step size α.
3. Compute new iterate:
$$x_+ = x + \alpha\eta$$

Convergence Properties

Retains convergence of Euclidean counterparts:

- Riemannian Newton: fast local convergence [Lue72, Gab82, Udr94, EAS98, MM02, ADM+02, DPM03, HT04]
- Riemannian Steepest Descent: robust global convergence [HM94,Udr94]

Riemannian Newton Method

1a. At iterate x, define pullback $\hat{f}_x = f \circ R_x$

1b. Find solution $\eta \in T_x\mathcal{M}$ of

$$\text{Hess } \hat{f}_x(0)\ \eta = -\text{grad } \hat{f}_x(0)$$

2. Choose step size α.
3. Compute new iterate:

$$x_+ = R_x(\alpha\eta)$$

Convergence Properties

Retains convergence of Euclidean counterparts:

- Riemannian Newton: fast local convergence [Lue72, Gab82, Udr94, EAS98, MM02, ADM+02, DPM03, HT04]
- Riemannian Steepest Descent: robust global convergence [HM94,Udr94]

Riemannian Newton Method

1a. At iterate x, define pullback $\hat{f}_x = f \circ R_x$

1b. Find solution $\eta \in T_x\mathcal{M}$ of

$$\text{Hess } \hat{f}_x(0)\ \eta = -\text{grad } \hat{f}_x(0)$$

2. Choose step size α.
3. Compute new iterate:

$$x_+ = R_x(\alpha\eta)$$

Convergence Properties

Retains convergence of Euclidean counterparts:

- Riemannian Newton: fast local convergence
  [Lue72, Gab82, Udr94, EAS98, MM02, ADM+02, DPM03, HT04]
- Riemannian Steepest Descent: robust global convergence [HM94,Udr94]

Riemannian Trust-Region Method

1a. At iterate x, define pullback $\hat{f}_x = f \circ R_x$

1. Construct quadratic model m_x of f around x
2. Find (approximate) solution to

$$\eta = \underset{\|\eta\| \leq \Delta}{\operatorname{argmin}}\ m_x(\eta)$$

3. Compute $\rho_x(\eta)$:

$$\rho_x(\eta) = \frac{f(x) - f(x+\eta)}{m_x(0) - m_x(\eta)}$$

4. Use $\rho_x(\eta)$ to adjust Δ and accept/reject new iterate:

$$x_+ = x + \eta$$

Convergence Properties

Retains convergence of Euclidean trust-region methods:

- robust global and fast local [ABG2007,BAG2008]

Riemannian Trust-Region Method

1a. At iterate x, define pullback $\hat{f}_x = f \circ R_x$

1b. Construct quadratic model m_x of $\hat{f}_x$

2. Find (approximate) solution to

$$\eta = \underset{\eta \in T_x \mathcal{M},\, \|\eta\| \leq \Delta}{\operatorname{argmin}} \quad m_x(\eta)$$

3. Compute $\rho_x(\eta)$:

$$\rho_x(\eta) = \frac{\hat{f}_x(0) - \hat{f}_x(\eta)}{m_x(0) - m_x(\eta)}$$

4. Use $\rho_x(\eta)$ to adjust Δ and accept/reject new iterate:

$$x_+ = R_x(\eta)$$

Convergence Properties

Retains convergence of Euclidean trust-region methods:

- robust global and fast local [ABG2007,BAG2008]

Riemannian Trust-Region Method

1a. At iterate x, define pullback $\hat{f}_x = f \circ R_x$

1b. Construct quadratic model m_x of $\hat{f}_x$

2. Find (approximate) solution to

$$\eta = \operatorname*{argmin}_{\eta \in T_x\mathcal{M},\, \|\eta\| \leq \Delta} m_x(\eta)$$

3. Compute $\rho_x(\eta)$:

$$\rho_x(\eta) = \frac{\hat{f}_x(0) - \hat{f}_x(\eta)}{m_x(0) - m_x(\eta)}$$

4. Use $\rho_x(\eta)$ to adjust Δ and accept/reject new iterate:

$$x_+ = R_x(\eta)$$

Convergence Properties

Retains convergence of Euclidean trust-region methods:

- robust global and fast local [ABG2007,BAG2008]

Riemannian Direct Search Methods

1. At iterate x, define pullback $\hat{f}_x = f \circ R_x$
2. Apply your favorite direct search technique to

$$\eta = \operatorname*{argmin}_{\eta \in T_x \mathcal{M}} \hat{f}_x(\eta)$$

3. Compute new iterate:

$$x_+ = R_x(\eta)$$

Useful for problems where we have no higher-order information about f:

- face recognition problems
- design optimization problems

See also:

- Dreisigmeyer (LANL)
- Liu, Srivastava, Gallivan (FSU)

In Summary...

Riemannian Optimization methods enjoy numerous benefits:

- The ability to tackle problems in natural setting
  - favors optimality over heuristic approaches
- The ability to handle constraints in an optimal way
  - coming from a recognition of the geometry of the problem
- Approaches for solving problems that aren't easily posed as constrained Euclidean problems
- Techniques from Euclidean optimization are easily moved to Riemannian setting, with convergence theory intact

Software Efforts

- Stiefel/Grassmann Optimization (SG_MIN) package
  http://www-math.mit.edu/∼lippert/sgmin.html
- Generic RTR (GenRTR) package
  http://www.scs.fsu.edu/∼cbaker/GenRTR

References

- Edelman, Arias, Smith: SIMAX '98
  "The Geometry of Algorithms with Orthogonality Constraints"
- Baker, Absil, Gallivan: IMAJNA '08
  "An Implicit Trust-Region Method on Riemannian Manifolds"
- Absil, Mahony, Sepulchre: Princeton, 2008
  "Optimization Algorithms on Matrix Manifolds"
  (the Magritte book)