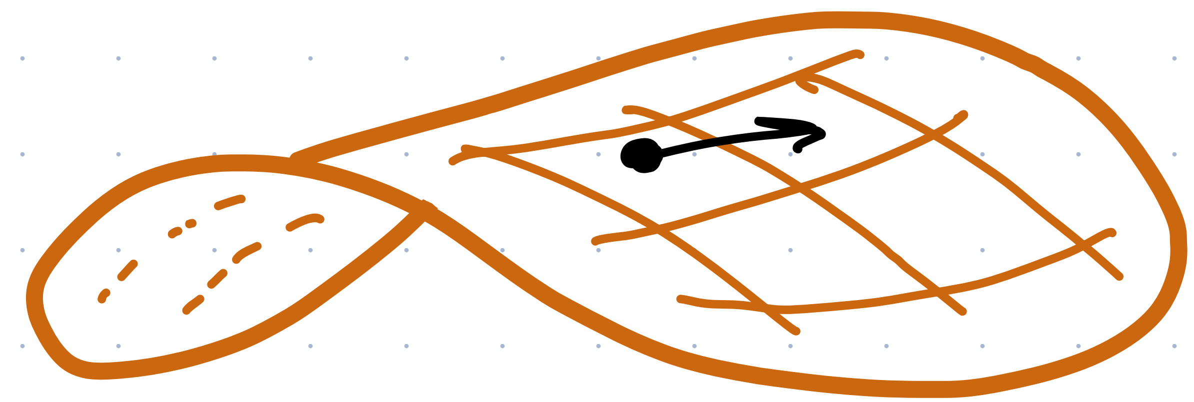
Nonsmooth Optimization on Riemannian Manifolds



HAJG JASA

SURE-AI Annual Conference, Oslo, 17-21 August 2026

Who cares anyways ?

Who cares anyways ?

You should !

Who cares anyways ?

You should !

if you work with

Who cares anyways ?

You should !

if you work with

- shape analysis

Who cares anyways ?

You should !

if you work with

- shape analysis
- signal/image processing

simula

Who cares anyways?

You should!

if you work with

- shape analysis
- signal/image processing
- (sparse) PCA, matrix completion

Who cares anyways ?

You should !

if you work with

- shape analysis
- signal/image processing
- (sparse) PCA, matrix completion
- tensor decompositon

simula

Who cares anyways?

You should!

if you work with

- shape analysis
- signal/image processing
- (sparse) PCA, matrix completion
- tensor decompositon
- neural networks

Who cares anyways ?

You should !

if you work with

- shape analysis
- signal / image processing
- (sparse) PCA , matrix completion
- tensor decompositon
- neural networks
- robotics and motion planning !

simula

3

Riemannian Optimization

→ algorithms to numerically solve

$$\text{minimize} \quad f(p) \, , \quad p \in \mathcal{M}$$

Riemannian Optimization

→ algorithms to numerically solve

$$\text{minimize } f(p), \quad p \in \mathcal{M}$$

with

- $\mathcal{M}$ a Riemannian manifold

Riemannian Optimization

→ algorithms to numerically solve

$$\text{minimize } f(p), \quad p \in \mathcal{M}$$

with

- $\mathcal{M}$ a Riemannian manifold
  - ↳ possibly high-dimensional

Riemannian Optimization

→ algorithms to numerically solve

minimize $f(p)$, $p \in \mathcal{M}$

with

- $\mathcal{M}$ a Riemannian manifold
  - ↳ possibly high-dimensional
  - ↳ no "+" or "−" operations

Riemannian Optimization

→ algorithms to numerically solve

minimize $f(p)$, $p \in \mathcal{M}$

with

- $\mathcal{M}$ a Riemannian manifold
  - ↳ possibly high-dimensional
  - ↳ no "+" or "−" operations
- $f: \mathcal{M} \to \overline{\mathbb{R}}$ a function

Riemannian Optimization

→ algorithms to numerically solve

$$\text{minimize } f(p)\,, \quad p \in \mathcal{M}$$

with

- $\mathcal{M}$ a Riemannian manifold
  - ↳ possibly high-dimensional
  - ↳ no "+" or "−" operations
- $f : \mathcal{M} \to \overline{\mathbb{R}}$ a function
  - ↳ possibly nonsmooth (gradient?)

Riemannian Optimization

→ algorithms to numerically solve

$$\text{minimize } f(p), \quad p \in \mathcal{M}$$

with

- $\mathcal{M}$ a Riemannian manifold
  - ↳ possibly high-dimensional
  - ↳ no "+" or "−" operations
- $f : \mathcal{M} \to \overline{\mathbb{R}}$ a function
  - ↳ possibly nonsmooth and nonconvex (local minima)

simula

So, how do we solve 'em ?

4

simula

So, how do we solve 'em ?

Classically, for a smooth, convex function

simula

So, how do we solve 'em ?

Classically, for a smooth, convex function

→ use gradient descent

$$x^{k+1} = x^k - \alpha^k \nabla f(x^k)$$

So, how do we solve 'em ?

Classically, for a smooth, convex function

→ use gradient descent

$$x^{k+1} = x^k - \alpha^k \nabla f(x^k)$$

- smoothness $\Rightarrow \exists \nabla f$

simula

So, how do we solve 'em ?

Classically, for a smooth, convex function

→ use gradient descent

$$x^{k+1} = x^k - \alpha^k \nabla f(x^k)$$

- smoothness $\Rightarrow \exists \nabla f$

- convexity $\Rightarrow$ optimality guarantees

So, how do we solve 'em ?

Classically, for a smooth, convex function

→ use gradient descent

$$x^{k+1} = x^k - \alpha^k \nabla f(x^k)$$

- ~~smoothness~~ $\Rightarrow \exists \nabla f$?

- convexity $\Rightarrow$ optimality guarantees

simula

So, how do we solve 'em ?

Classicaly, for a smooth, convex function

→ we subgradient "descent"

$$x^{k+1} = x^k - \alpha^k g(x^k), \quad g(x^k) \in \partial f(x^k)$$

- ~~smoothness~~ $\Rightarrow \exists \nabla f$?

  we can define a generalization : subdifferential ∂f

- convexity $\Rightarrow$ optimality guarantees

So, how do we solve 'em ?

Classically, for a smooth, convex function

→ use subgradient "descent"

$$x^{k+1} = x^k - \alpha^k g(x^k), \quad g(x^k) \in \partial f(x^k)$$

- ~~smoothness~~ $\Rightarrow \exists \nabla f$?

  we can define a generalization : subdifferential ∂f

- ~~convexity~~ $\Rightarrow$ optimality guarantees ?

simula

So, how do we solve 'em ?

Classicaly, for a smooth, convex function

→ we subgradient "descent"

$$x^{k+1} = x^k - \alpha^k g(x^k), \quad g(x^k) \in \partial f(x^k)$$

- ~~smoothness~~ ⇒ $\exists \nabla f$?

we can define a generalization : subdifferential ∂f

- ~~convexity~~ ⇒ optimality guarantees ?

ε-optimality : when $\|g(x^k)\| \leq \varepsilon$, for $g(x^k) \in \partial f(x^k)$

simula

simula

So, how do we solve 'em ?

On manifolds, for a smooth, convex function

→ we subgradient "descent"

$$p^{k+1} = p^k - \alpha^k g(p^k), \quad g(p^k) \in \partial f(p^k)$$

- ~~smoothness~~ $\Rightarrow \exists \nabla f$?

we can define a generalization : subdifferential ∂f

- ~~convexity~~ $\Rightarrow$ optimality guarantees ?

ε-optimality : when $\|g(p^k)\| \leq \varepsilon$, for $g(p^k) \in \partial f(p^k)$

So, how do we solve 'em ?

On manifolds, for a smooth, convex function

→ use subgradient "descent"

$$p^{k+1} = p^k - \alpha^k g(p^k), \quad g(p^k) \in \partial f(p^k)$$

In my Ph.D :

- bundle method : collect several subgradients and combine them

So, how do we solve 'em ?

On manifolds, for a smooth, convex function

→ use subgradient "descent"

$$p^{k+1} = p^k - \alpha^k g(p^k), \quad g(p^k) \in \partial f(p^k)$$

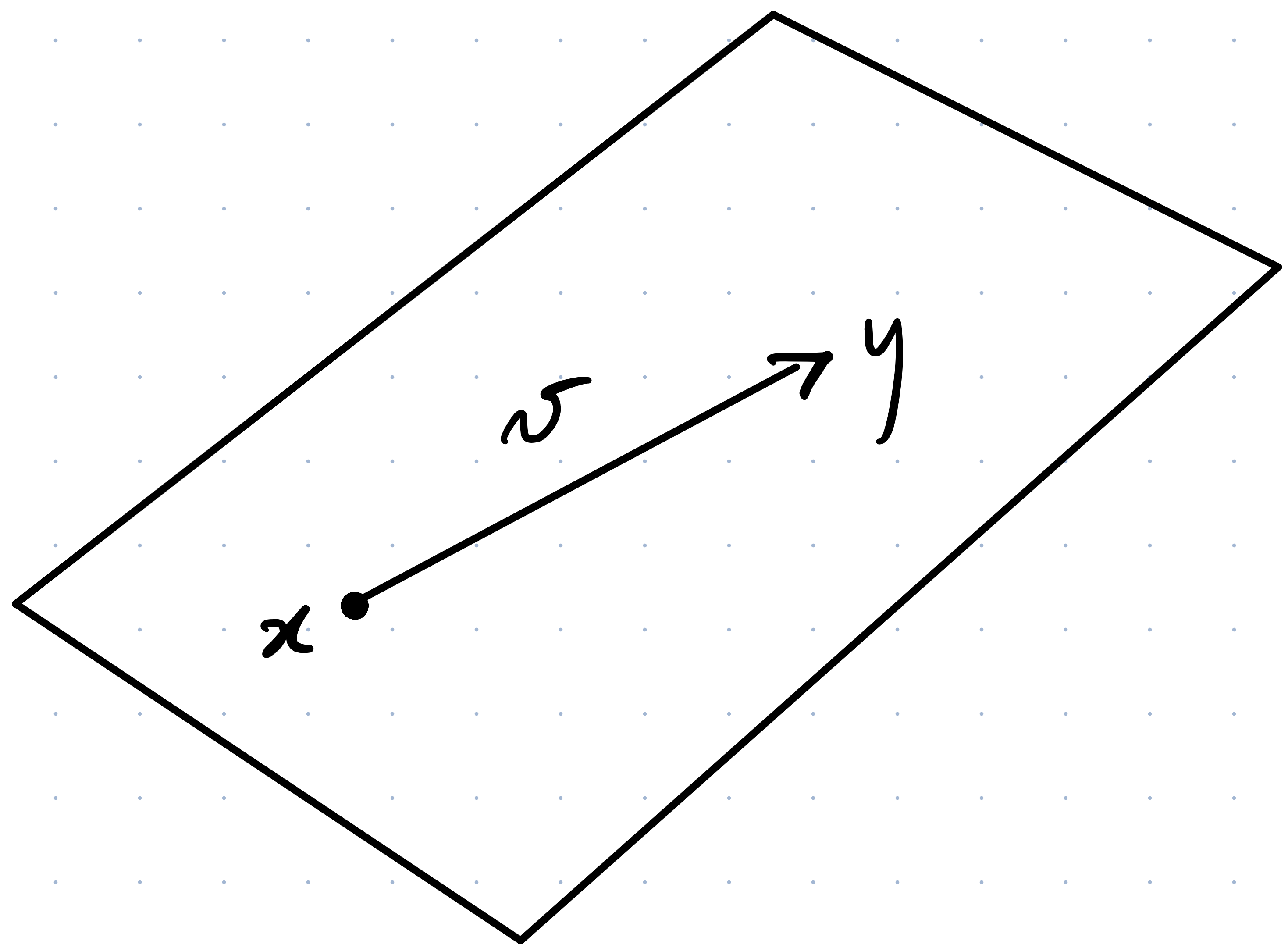
In my Ph.D :

- bundle method : collect several subgradients and combine them
- proximal gradient method : combine gradient and prox step

simula

Making sense of "±"

Making sense of "±"

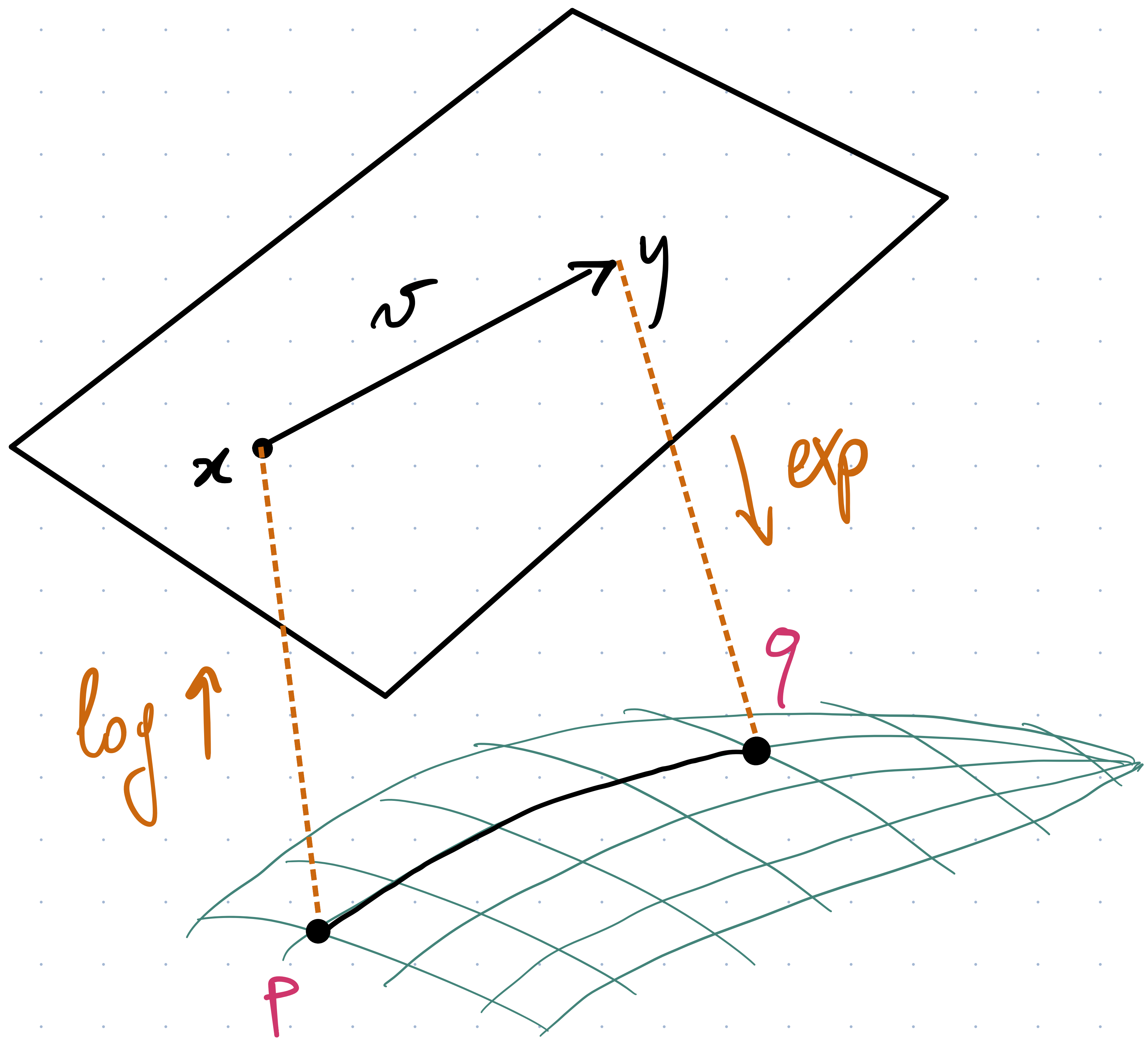


$v = y - x$

$y = x + v$

simula

Making sense of "±"



$$v = y - x$$

$$y = x + v$$

$$x = \log_p p$$

$$v = \log_p q$$

$$q = \exp_p v$$

Software

Optimization on manifolds : manopt family

JULIA : Manopt.jl

PYTHON : pymanopt

MATLAB : Manopt

Software

Optimization on manifolds : manopt family

JULIA : Manopt.jl
PYTHON : pymanopt
MATLAB : Manopt

in Julia, separate manifolds library in
Manifolds.jl

Summary

- motivation : real-world applications
- solutions : from Euclidean to Riemannian algorithms
- for the user : software

simula

Thank you
for your
attention !