Chapter 1
Basics and necessary tools

Variational Methods for Computer Vision
WS 17/18

Michael Moeller
Visual Scene Analysis
Department of Computer Science
University of Siegen

Repeating some math basics

Notation, norm, inner product

We will mostly work in the vector space $\mathbb{R}^n$ equipped with an inner product

$$\langle x, y \rangle = x^T y = \sum_{i=1}^{n} x_i y_i$$

for $x, y \in \mathbb{R}^n$.

The ℓ^2 norm is *induced* by this inner product, i.e.

$$\|x\|_2 = \sqrt{\langle x, x \rangle} = \sqrt{\sum_{i=1}^{n} x_i^2}.$$

There are other norms, e.g. the ℓ^1 or the ℓ^∞ norms

$$\|x\|_1 = \sum_{i=1}^{n} |x_i| \qquad , \qquad \|x\|_\infty = \max_i |x_i|$$

which we will use less frequently.

More norms

One can also define norms on function spaces! E.g. for a function $f : [0, 1] \to \mathbb{R}$ we define

$$\|f\|_2 = \sqrt{\int_0^1 (f(x))^2 \, dx}.$$

Even for functions, one has different options, e.g. using the L_1 instead of the L_2 norm:

$$\|f\|_1 = \int_0^1 |f(x)| \, dx.$$

As we will discuss in a couple of slides, one can also integrate in multiple variables and define norms on functions $f : U \subset \mathbb{R}^n \to \mathbb{R}$, e.g.

$$\|f\|_2 = \sqrt{\int_U (f(x))^2 \, dx}.$$

Continuity

We will need the concept of *continuous* functions. Do you remember when

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

is continuous?

Intuitive: You can draw f without lifting your pencil!

More mathematical: As $x \to x_0$ it holds that $f(x) \to f(x_0)$

The previous definition generalizes to vector valued functions!

f is continuous at x_0 if for all $\epsilon > 0$ there exists $\delta > 0$ such that for all x with $\|x - x_0\| \leq \delta$ it holds that $\|f(x) - f(x_0)\| \leq \epsilon$.

Mutlivariate derivatives

We will need to take derivatives of functions

$$f : \mathbb{R}^n \to \mathbb{R}^m$$

Do you remember/know how?

Definition: Jacobi matrix

For a function $f : U \subset \mathbb{R}^n \to \mathbb{R}^m$ with continuous partial derivatives we write $f \in C^1$ and call

$$Jf(x) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1}(x) & . & . & \frac{\partial f_1}{\partial x_n}(x) \\ . & . & . & . \\ . & . & . & . \\ \frac{\partial f_m}{\partial x_1}(x) & . & . & \frac{\partial f_m}{\partial x_n}(x) \end{pmatrix} \in \mathbb{R}^{m \times n}$$

the *Jacobi matrix* of f at $x \in U$. It is the first derivative of multivariate functions. Jf itself is a continuous function $Jf : U \to \mathbb{R}^{m \times n}$.

Example on the board $f : \mathbb{R}^n \to \mathbb{R}$, $f(x) = \frac{1}{2}\|x - y\|^2$.

Composite functions

Chain rule for multivariate functions

Let

$$f : \mathbb{R}^n \to \mathbb{R}^m \in C^1 \qquad \text{and} \qquad g : \mathbb{R}^m \to \mathbb{R}^k \in C^1.$$

Then the composite function $(g \circ f) : \mathbb{R}^n \to \mathbb{R}^k$ is continuously differentiable and its Jacobian $J(g \circ f)$ is given by

$$J(g \circ f)(x) = (Jg)(f(x)) \cdot Jf(x).$$

Example on the board $g : \mathbb{R}^m \to \mathbb{R}$, $g(x) = \frac{1}{2}\|x - y\|^2$, $f : \mathbb{R}^n \to \mathbb{R}^m$, $f(z) = Az$ for some matrix $A \in \mathbb{R}^{m \times n}$.

Multivariate integration

We will need to take integrate functions

$$f : U \subset \mathbb{R}^n \to \mathbb{R}$$

Do you remember/now how?

Integrate with respect to all variables sequentially!

Example:

$$f : \{(x, y) \mid 1 \leq x \leq 4, \ -2 \leq y \leq 1\} \to \mathbb{R}$$
$$f(x, y) = y^2 + 1 + \sin(x)y$$

Another example:

$$f : \{(x, y) \mid x^2 + y^2 \leq 1\} \to \mathbb{R}$$
$$f(x, y) = 1 + x\cos(y^6)$$

Eigendecompositions

Eigenvalues

Let $A \in \mathbb{R}^{n \times n}$ be a matrix. We say $\lambda \in \mathbb{R}$ is an *eigenvalue* of A if there exists a $v \neq 0$ such that

$$Av = \lambda v.$$

The corresponding v is called an *eigenvector*.

If there exist matrices $U \in \mathbb{R}^{n \times n}$ with $U^T U = UU^T = I$, and a diagonal matrix $D \in \mathbb{R}^{n \times n}$ such that

$$A = UDU^T$$

we call this an *eigendecomposition* of A. The diagonal elements of D are eigenvalues of A.

Singular value decomposition

Not every matrix $A \in \mathbb{R}^{n \times n}$ has a diagonal eigendecomposition over $\mathbb{R}$. It often is useful to use a *singular value decomposition*, which even works for matrices $A \in \mathbb{R}^{n \times m}$.

Singular value decomposition (SVD)

For any $A \in \mathbb{R}^{n \times m}$ there exist orthogonal matrices $U \in \mathbb{R}^{n \times n}$, $V \in \mathbb{R}^{m \times m}$, and a non-negative diagonal matrix $D \in \mathbb{R}^{n \times m}$ such that

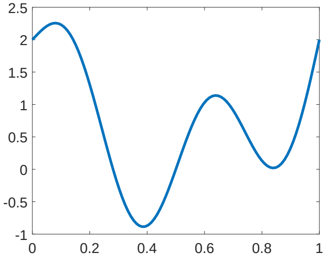
$$A = UDV^T$$

The diagonal entries of D are called singular values.

The number of nonzero singular values is the *rank* of A.

Signal Representation

How do we represent signals?



Continuous: Functions

$$f : [a, b] \to \mathbb{R}$$
$$x \mapsto f(x)$$

Discrete: Vectors $f \in \mathbb{R}^n$

One typically interprets/relates:

$$f_i = f(x_i), \qquad x_i = a + (i-1) \cdot \frac{b-a}{n-1}, \qquad \text{for } i \in \{1, \cdots, n\}.$$

How do we represent images?

Continuous: Functions

Grayscale

$$f : \Omega \subset \mathbb{R}^2 \to \mathbb{R}$$
$$x \mapsto f(x)$$

Color

$$f : \Omega \subset \mathbb{R}^2 \to \mathbb{R}^3$$
$$x \mapsto f(x) = (f_R(x), f_G(x), f_B(x))^T$$

Discrete: Matrices and Tensors

Grayscale

$f \in \mathbb{R}^{n \times m}$

Color

$f \in \mathbb{R}^{n \times m \times 3}$

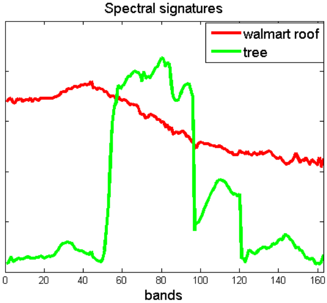
The points $x_{i,j}$ at which the continuous function f is sampled to obtain its discrete representation are called *pixels*.

Many more types of image data

$f : \Omega \subset \mathbb{R}^2 \to \mathbb{R}^n$ or $f : (\Omega \times \Gamma) \subset \mathbb{R}^3 \to \mathbb{R}$

E.g. hyperspectral images.

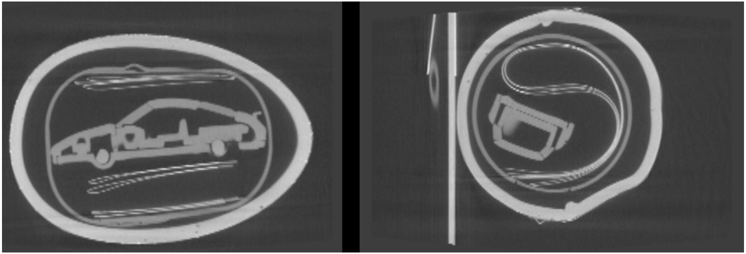
Hyperspectral cube with 163 bands



Many more types of image data

$f : \Omega \subset \mathbb{R}^3 \to \mathbb{R}$

E.g. medical imaging - three spatial dimension.



Many more types of image data

$f : (\Omega \times \Gamma) \subset \mathbb{R}^3 \to \mathbb{R}^3$

E.g. color videos.

▸ Coffee?

More types of discretization

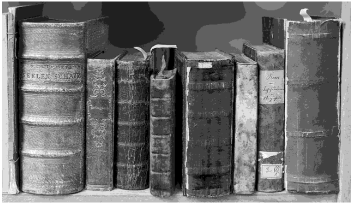
Besides the discretization of the *domain* Ω

$$f : \Omega \to \mathbb{R} \qquad \to \qquad f : \{x_{1,1}, \cdots, x_{n,m}\} \to \mathbb{R}$$

digital images may also have a discrete *range*, e.g.,

$$f : \{x_{1,1}, \cdots, x_{n,m}\} \to \{0, \cdots, 255\}$$

for an $8 - bit$ quantization.



Variational Methods

Variational methods

Define an energy E on continuous images, i.e.,

$$E : \mathcal{X} \to \mathbb{R} \cup \{\infty\} \tag{1}$$

from a suitable space $\mathcal{X}$ of images (typically a Banach space) to the extended real numbers, such that

- u with desirable properties $\to E(u)$ small,
- unrealistic/"bad" $u \to E(u)$ large.

If $\mathcal{X}$ is a function space (continuous formulation of images), then E is a function that maps functions to real numbers. We call E a *functional*.

For $\mathcal{X}$ being a function space, determining the solution of an imaging problem by determining

$$\hat{u} = \operatorname*{argmin}_u E(u),$$

is called a *variational method*.

Calculus of variation

Analyzing variational methods

$$\hat{u} = \operatorname*{argmin}_{u} E(u),$$

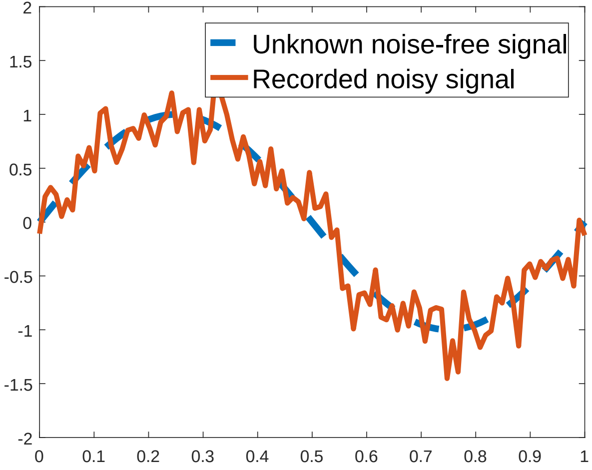
in terms of existence, uniqueness, optimality conditions and properties of the solution can be mathematically challenging and requires *functional analysis*.

We will

- Often formulate energies in a continuous setting.
- Not require prior knowledge in functional analysis.
- Occasionally do some analysis in infinite dimensions/function spaces.
- Often turn to a discrete point of view and use analysis instead of functional analysis.

A simple example

Let us consider a simple example:



How can we reduce the noise?

A simple example

The denoised signal should still look somewhat similar to the input data. But how should we measure similarity? Simple choice:

$$H_f(u) = \int_0^1 (u(x) - f(x))^2 \, dx =: \|u - f\|_2^2.$$

The denoised signal should be smoother, i.e., contain less oscillations. We need a regularization R that penalizes rapid changes of the signal! Simple choice:

$$R(u) = \int_0^1 \left(\partial_x u(x)\right)^2 \, dx = \|\partial_x u\|_2^2.$$

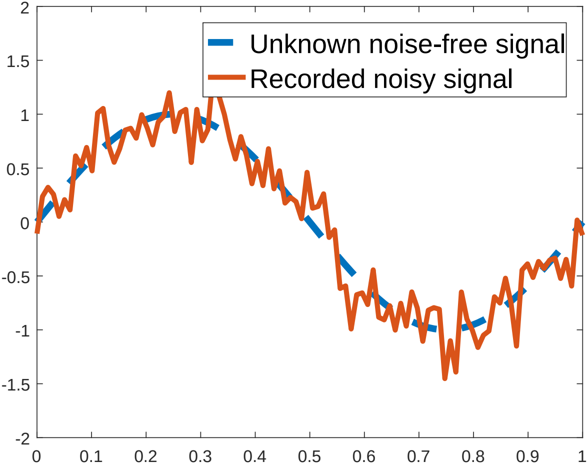
Overall variational method:

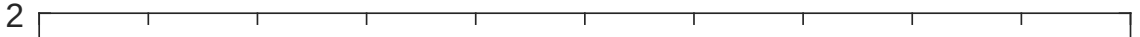
$$\hat{u} = \operatorname*{argmin}_u H_f(u) + \alpha R(u).$$

A simple example

Result of

$$\hat{u} = \underset{u}{\operatorname{argmin}} \|u - f\|_2^2 + 10 \cdot \|\partial_x u\|_2^2$$





A simple example

For the computation I, of course, discretized

$$\hat{u} = \underset{u}{\operatorname{argmin}} \|u - f\|_2^2 + 10 \cdot \|\partial_x u\|_2^2$$

and used

$$\mathbb{R}^n \ni \hat{u} = \underset{u \in \mathbb{R}^n}{\operatorname{argmin}} \sum_{i=1}^{n} (u_i - f_i)^2 + 10 \cdot \sum_{i=2}^{n} (u_i - u_{i-1})^2,$$
$$= \underset{u \in \mathbb{R}^n}{\operatorname{argmin}} \|u - f\|_2^2 + 10 \cdot \|Du\|_2^2,$$

with the discrete derivative matrix

$$\mathbb{R}^{n-1 \times n} \ni D = \begin{bmatrix} -1 & 1 & & & \\ & -1 & 1 & & \\ & & \ddots & \ddots & \\ & & & -1 & 1 \end{bmatrix}.$$

Why care about a continuous representation?

For our simple example, we had two formulations:

Continuous:

$$\hat{u} = \underset{u}{\operatorname{argmin}} \int_0^1 (u(x) - f(x))^2 \, dx + \alpha \int_0^1 \left(\partial_x u(x)\right)^2 \, dx$$

Discrete:

$$\hat{u} = \underset{u \in \mathbb{R}^n}{\operatorname{argmin}} \sum_{i=1}^{n} (u_i - f_i)^2 + \alpha \cdot \sum_{i=2}^{n} (u_i - u_{i-1})^2$$

Why should we care about a continuous formulation at all, if the computer can only compute discrete solutions anyways?

Reasons for variational methods (continuous formulation)

1. Beautifully concise formulation.

2. Independence of the discretization.

3. Some effects can only be explained in a continuous setting!

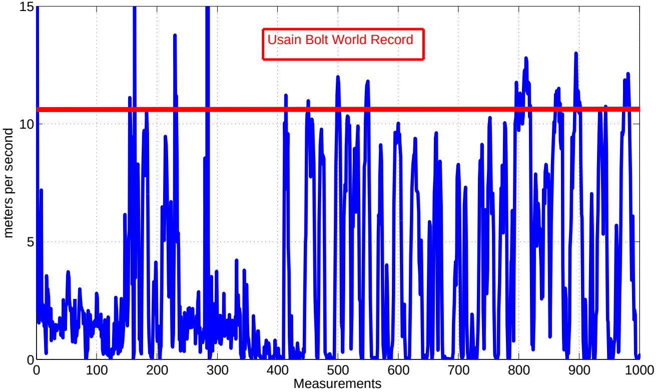
Differentiation

Data from: *Microsoft Research GeoLife GPS Trajectories*

Time	'12:44:12'	'12:44:13'	'12:44:15'
Latitude	39.974408918	39.974397078	39.973982524
Longitude	116.30352210	116.30352693	116.30362184

How fast did this person go?

Differentiation



New world record? Top speed of 161.78 km/h?

What went wrong?

Something makes the problem of differentiation nasty...

Definition (Well-posed problems (Hadamard))

A problem is *well-posed* if the following three properties hold.

1. **Existence**: For all suitable data, a solution exists.
2. **Uniqueness**: For all suitable data, the solution is unique.
3. **Stability**: The solution depends continuously on the data.

Definition (Ill-posed problems)

A problem that violates any of the three properties of well-posedness is called an *ill-posed problem*.

Stability?

What does stability really mean?

Continuous dependence on the data

Let f^δ be the measured data, and $I(f, \delta)$ the operation of recovering our desired solution (assuming existence and uniqueness).

We say that the solution depends continuously on the data if for any $f^\delta = f + n^\delta$ with $\|n^\delta\| \leq \delta$ it holds that $\|I(f, 0) - I(f^\delta, \delta)\| \to 0$ as $\delta \to 0$. In other words, I is continuous.

Differentiation is ill-posed

Computation on the board:

Ill-posedness of differentiation

For $f, f^\delta \in C^1([0,1])$, although the error in the data

$$\|f - f^\delta\| \leq \delta$$

is arbitrary small, the error between the derivatives

$$\|\partial_x f - \partial_x f^\delta\|$$

can be arbitrary large!

We understood the behavior in the continuous setting, i.e. independent of the discretization. What can we do?

$\rightarrow$ **Variational Methods!**

Variational methods can fight ill-posedness

The intuitive statement

Variational methods behave nicely

Variational methods allow to re-establish the continuous dependence on the data and therefore stabilize the problem!

Exemplary mathematical result

Proposition

Let $f \in C_0^2([0,1])$ be twice continuously differentiable. If we determine

$$u^\alpha = \underset{u}{\text{argmin}} \, \|u - f^\delta\|^2 + \alpha \cdot \|\partial_x u\|^2,$$

subject to $u(0) = u(1) = 0$, then there is a parameter choice rule $\alpha = \alpha(\delta)$ such that

$$\|\partial_x u^\alpha - \partial_x f\| \overset{\delta \to 0}{\to} 0.$$

Why not discrete?

Discrete (=finite dimensional) linear inverse problems never violate the criterion of "continuous dependence on the data"!

Let

$$D = \frac{1}{h}\begin{bmatrix} -1 & 1 & & & \\ & -1 & 1 & & \\ & & \ddots & \ddots & \\ & & & -1 & 1 \end{bmatrix}$$

be the finite difference matrix. Then

$$\lim_{\delta\to 0} Df^\delta = D\left(\lim_{\delta\to 0} f^\delta\right) = Df,$$

since matrices are continuous operators. This holds for any matrix D.

$\rightarrow$ **The discrete problem is not ill-posed!**

An imaging example for ill-posedness

Original image

An imaging example for ill-posedness

Blurry image $f = k * u$

An imaging example for ill-posedness

Reconstructed image $u = \mathcal{F}^{-1}(\mathcal{F}(f)/\mathcal{F}(k))$

An imaging example for ill-posedness

Blurry image $f = k * u$

An imaging example for ill-posedness

Blurry noisy image $f = k * u + n, \Rightarrow \mathcal{F}(f) \approx \mathcal{F}(k) \cdot \mathcal{F}(u)$

An imaging example for ill-posedness

Reconstruction by $\mathcal{F}^{-1}(\mathcal{F}(f)/\mathcal{F}(k))$

Summary

Variational methods

$$\hat{u} = \arg\min_u \underbrace{H_f(u)}_{\text{data term}} + \underbrace{\alpha}_{\text{regularization parameter}} \underbrace{R(u)}_{\text{regularization}}$$

Many practical problems do not depend on the data continuously, they are ill-posed!

Seen for the example of taking the derivative: Vartiational methods can stabilize such problems.

Besides a more concise formulation, the effect of ill-posedness could only be explained in function spaces.

$\rightarrow$ **Let us investigate variational methods in more detail! What are optimality conditions?**

Analysis I

Let us start with the simple (discrete) case:

$$\hat{u} = \underset{u \in \mathbb{R}^n}{\operatorname{argmin}} \sum_{i=1}^{n} (u_i - f_i)^2 + \alpha \cdot \sum_{i=2}^{n} (u_i - u_{i-1})^2$$

What is a necessary condition for optimality?

The gradient with respect to u is zero, i.e.,

$$\begin{aligned} 0 &= 2(u_i - f_i) + 2\alpha(u_i - u_{i-1}) + 2\alpha(u_i - u_{i+1}), \\ \Rightarrow &(1 + 2\alpha)u_i - \alpha u_{i-1} - \alpha u_{i+1} = f_i, \end{aligned}$$

for all $i \in \{1, \cdots, n\}$ with $u_0 = u_1$ and $u_{n+1} = u_n$.

Linear system with n equations and n unknowns.

Sufficient condition?

Sufficient condition for optimality

Definition: Convexity

We call $E : \mathbb{R}^n \to \mathbb{R}$ a convex function if for all $u, v \in C$ and all $\theta \in [0, 1]$ it holds that

$$E(\theta u + (1 - \theta)v) \leq \theta E(u) + (1 - \theta)E(v)$$

We call E strictly convex, if the inequality is strict for all $\theta \in]0, 1[$, and $v \neq u$.

Theorem

Let $E : \mathbb{R}^n \to \mathbb{R} \in C^1(\mathbb{R}^n)$ be a convex function. Then $\nabla E(\hat{u}) = 0$ implies that $\hat{u}$ is a global minimizer of E.

Proof: Exercise sheet 2.

Calculus of Variation

What about the continuous case?

Let us start with our simple denoising example

$$u^\alpha = \operatorname*{argmin}_{u \in H_0^1(\Omega)} E(u) \quad \text{with} \quad E(u) = \|u - f^\delta\|^2 + \alpha \cdot \|\partial_x u\|^2, \tag{2}$$

Board: Let us work with the idea that

$$E(u^\alpha) \leq E(u^\alpha + \epsilon h)$$

for arbitrary numbers $\epsilon \in \mathbb{R}$ and arbitrary functions $h \in H_0^1(\Omega)$.

Calculus of Variation

We use

Fundamental Lemma of Calculus of Variation

If a pair of continuous functions g, v on an interval $]a, b[$ meet

$$\int_a^b g(x)h(x) + v(x)\partial_x h(x)\; dx = 0$$

for all compactly supported smooth functions h on $]a, b[$, then v is differentiable, and $\partial_x v \equiv g$.

to show that the solution to (2) meets

$$0 = u^\alpha - f - \alpha \partial_{xx} u^\alpha!$$

Continuous case and Euler-Lagrange equations

Is there a systematic concept behind this?

Euler-Lagrange Equations

Let $\rho : \mathbb{R}^n \times \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}$ be a differentiable function with three arguments, $\rho(x, v, z)$, and consider the problem

$$\hat{u} \in \operatorname*{argmin}_u \int_\Omega \rho(x, u(x), \nabla u(x))\, dx.$$

Then $\hat{u}$ satisfies the **Euler-Lagrange Equations**

$$\left(\frac{d\rho}{dv} - \nabla_x \cdot \nabla_z \rho \right)(x, \hat{u}(x), \nabla \hat{u}(x)) = 0 \quad \forall x.$$

Further considerations

Sufficient conditions for global optimality?

- Depending on the boundary conditions of u, one can get an additional condition.
- To go from a critical point to a global minimum one again needs convexity.

Euler-Lagrange equations are typically too restrictive for variational problems in computer vision since they require ρ to be differentiable. A less restrictive analysis is based on **subgradients**.

We will not detail this continuous analysis too much. Funny things can happen in infininte dimensions which makes the analysis more complicated.

Simple Optimization

Discrete energy minimization only!

We will leave the continuous point-of-view for a while.

Consider

$$\min_{u \in \mathbb{R}^n} E(u)$$

for $E : \mathbb{R}^n \to \mathbb{R}$.

Strategy: Compute $\nabla E(u)$

- Can we solve $\nabla E(u) = 0$ for u directly?
- If not, apply gradient descent algorithm as presented in the following slides.

Example for solvable $\nabla E(u) = 0$

Remember our quadratic ℓ^2-denoising problem

$$\hat{u} = \underset{u}{\operatorname{argmin}} \|u - f\|^2 + \alpha \cdot \|Du\|^2,$$

for a discrete derivative matrix D.

We have shown / will show in the exercises that the optimality condition to such a problem is

$$0 = \hat{u} - f + \alpha D^T D\hat{u},$$
$$\Rightarrow \hat{u} = (I + \alpha D^T D)^{-1} f.$$

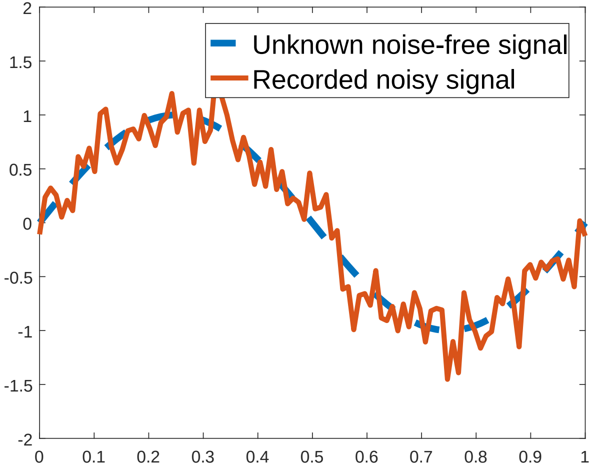
Numerical methods

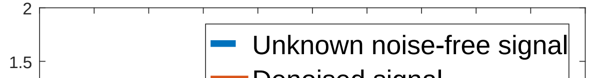
- Jacobi method
- Gauss-Seidel method
- Successive overrelaxation (SOR)
- Conjugate gradient (CG)

We won't detail the math. Use backslash or `pcg` in MATLAB. Make sure you declared $(I + \alpha D^T D)$ to be sparse!

Why do we need to go beyond quadratic ℓ^2-denoising?

We got good denoising results in 1d:





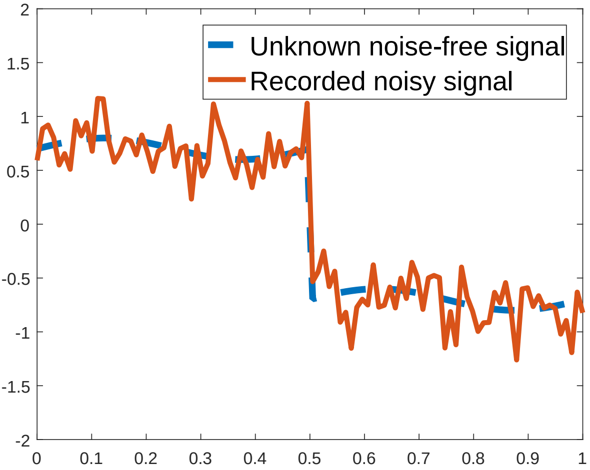
Why do we need to go beyond quadratic ℓ^2-denoising?

What happens in 2d, i.e. for images?

Why do we need to go beyond quadratic ℓ^2-denoising?

What went wrong?

Images are not continuous! Edges are extremely important for visual impression!



Why does quadratic ℓ^2-denoising oversmooth edges?

Going in several small steps is cheaper than one large step!

Assume we need to go from 0 to 1 in 10 steps. We penalize

$$E(u) = \sum_{i=1}^{10} |u_{i+1} - u_i|^2$$

and fix $u_1 = 0$, $u_{11} = 1$.

10 equal steps:

$$E(u) = \sum_{i=1}^{10} (1/10)^2 = 10 \cdot \frac{1}{100} = \frac{1}{10}.$$

1 big step

$$E(u) = \sum_{i=1, i\neq 5}^{10} 0^2 + 1 = 1.$$

It is 10 times more expensive to take one big step!!

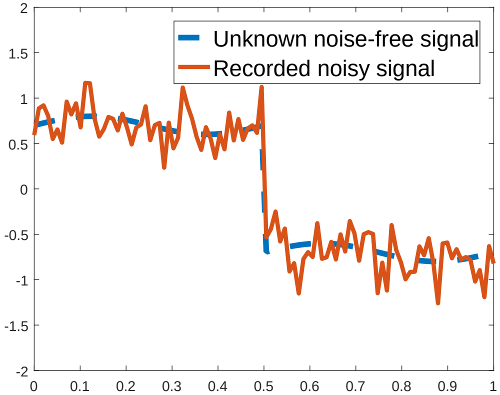
Any idea on how to fix it?

Use

$$E(u) = \sum_{i=1}^{10} |u_{i+1} - u_i|$$

instead!

The costs will be 1 for any monotonically increasing u!! The data term will decide if one needs a jump!



What about the optimization now?

Consider

$$\min_u \sum_{i=1}^{n}(u_i - f_i)^2 + \alpha \sum_{i=1}^{n-1} |u_{i+1} - u_i| \;=\; \min_u \|u - f\|^2 + \alpha \|Du\|_1.$$

What is the optimality condition now?

The ℓ^1 norm is not differentiable! What can we do?

While there are ways to handle the discontinuity, we will simply smooth the ℓ^1 norm by

$$S_\epsilon(d) = \sum_{i=1}^{m} \sqrt{\epsilon^2 + d_i^2},$$

and consider

$$\min_u \|u - f\|^2 + \alpha\, S_\epsilon(Du).$$

What about the optimization now?

What is the optimality condition for

$$\min_u \|u - f\|^2 + \alpha\ S_\epsilon(Du)?$$

Chain rule

Let $J : \mathbb{R}^m \to \mathbb{R}$ be differentiable and $A \in \mathbb{R}^{m\times n}$. Then

$$\nabla(J \circ A)(u) = A^T \nabla J(Au).$$

Therefore, we find the optimality condition

$$0 = 2(\hat{u} - f) + \alpha D^T \nabla S(D\hat{u})$$

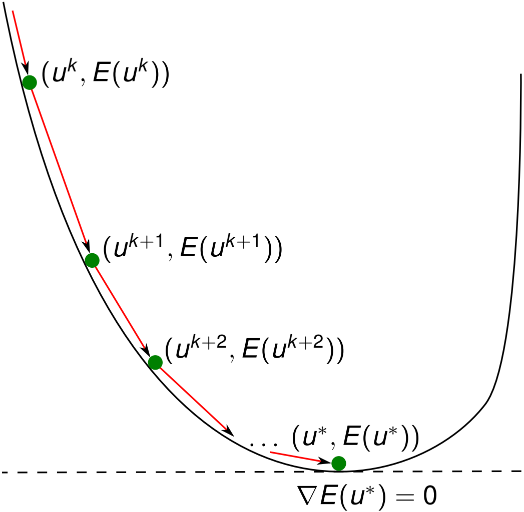
$$= 2(\hat{u} - f) + \alpha D^T \begin{pmatrix} \frac{(D\hat{u})_1}{\sqrt{\epsilon^2 + (D\hat{u})_1^2}} \\ \dots \\ \dots \\ \frac{(D\hat{u})_{n-1}}{\sqrt{\epsilon^2 + (D\hat{u})_{n-1}^2}} \end{pmatrix}$$

We will not be able to solve this in closed form!

Gradient descent algorithm

Idea: For minimizing the energy E, move into the direction of steepest descent

$$u^{k+1} = u^k - \tau^k \nabla E(u^k).$$



Gradient descent algorithm

Gradient descent with backtracking line seach

Pick $\alpha \in]0, 0.5[$ and $\beta \in]0, 1[$. Iterate:

- Given an estimate u^k, compute $E(u^k)$ and $\nabla E(u^k)$.
- Initialize $\tau_k = \tau^0$.
- Find a good τ_k by:

$$u^{test} = u^k - \tau_k \nabla E(u^k)$$

while $E\left(u^{test}\right) > E(u^k) - \alpha\tau_k \left\|\nabla E(u^k)\right\|^2$

$$\tau_k \leftarrow \beta\tau_k$$

$$u^{test} = u^k - \tau_k \nabla E(u^k)$$

end

- Once τ^k meets the criterion in the while-loop, update

$$u^{k+1} = u^{test}.$$

Gradient descent algorithm

Practical considerations:

- Guessing good values for α and β is often difficult and requires some problem-specific fine-tuning.
- Stopping criteria could be based on $\|u^k - u^{k+1}\| \leq \epsilon$ or $\|\nabla E(u^k)\| \leq \epsilon$.
- In practice one should definitely also define a maximum number of iterations.
- Allow the user to specify a starting point. Good guesses on the solution can improve the speed of convergence significantly!