Chapter 1
Examples of Ill-Posed Problems

Ill-Posed Problems in Image and Signal Processing
WS 2014/2015
What are ill-posed problems?

**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*. 
Differentiation

Data from: *Microsoft Research GeoLife GPS Trajectories*

<table>
<thead>
<tr>
<th>Time</th>
<th>'12:44:12'</th>
<th>'12:44:13'</th>
<th>'12:44:15'</th>
</tr>
</thead>
<tbody>
<tr>
<td>Latitude</td>
<td>39.974408918</td>
<td>39.974397078</td>
<td>39.973982524</td>
</tr>
<tr>
<td>Longitude</td>
<td>116.30352210</td>
<td>116.30352693</td>
<td>116.30362184</td>
</tr>
</tbody>
</table>

How fast did this person go?
Examples of Ill-Posed Problems

Michael Moeller

Differentiation

New world record? Top speed of 161.78 km/h?
Differentiation is ill-posed

Computation:

The solution does not depend continuously on the data.

**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!
Why is that interesting in practice?

In practice, measurements are **NEVER** exact!

![Graph showing measurements when dropping a ball](image1)

![Graph showing acceleration during free fall](image2)
What can we do?

We need more information or additional assumptions!

**Goal:** Continuous dependence of $\| \partial_x f^\delta - \partial_x f \|_X$ on $\| f^\delta - f \|_Y$. 
What can we do?

We need more information or additional assumptions!

**Goal:** Continuous dependence of \( \| \partial_x f^\delta - \partial_x f \|_X \) on \( \| f^\delta - f \|_Y \).

**Option 1:** Bound the noise in a stronger norm, e.g.

\[
\| n^\delta \|_Y^2 = \int_0^1 |n^\delta(x)|^2 \, dx + \int_0^1 |\partial_x n^\delta(x)|^2 \, dx,
\]

(a norm in the *Sobolev space* \( H^1([0, 1]) \)).
What can we do?

We need more information or additional assumptions!

Goal: Continuous dependence of $\| \partial_x f^\delta - \partial_x f \|_X$ on $\| f^\delta - f \|_Y$.

Option 1: Bound the noise in a stronger norm, e.g.

$$\| n^\delta \|_Y^2 = \int_0^1 |n^\delta(x)|^2 \, dx + \int_0^1 |\partial_x n^\delta(x)|^2 \, dx,$$

(a norm in the Sobolev space $H^1([0, 1])$).

→ Unrealistic in practice!
→ In our example this would assume the prior knowledge that the frequency $k$ of the noise is bounded!
What can we do?

We need more information or additional assumptions!

**Goal:** Continuous dependence of $\| \partial_x f^\delta - \partial_x f \|_X$ on $\| f^\delta - f \|_Y$.
What can we do?

We need more information or additional assumptions!

**Goal:** Continuous dependence of $\|\partial_x f^\delta - \partial_x f\|_X$ on $\|f^\delta - f\|_Y$.

**Option 2:** Assume additional regularity of the estimated solution $f_\alpha$ and use regularization. Solve

$$-\alpha \partial_{xx} f_\alpha(x) + f_\alpha(x) = f^\delta(x) \quad \text{(RD)}$$

for $f_\alpha$. 
What can we do?

We need more information or additional assumptions!

**Goal:** Continuous dependence of \( \| \partial_x f^\delta - \partial_x f \|_X \) on \( \| f^\delta - f \|_Y \).

Option 2: Assume additional regularity of the estimated solution \( f_\alpha \) and use *regularization*. Solve

\[
-\alpha \partial_{xx} f_\alpha(x) + f_\alpha(x) = f^\delta(x) \quad \text{(RD)}
\]

for \( f_\alpha \).

→ Allows us to bound the error in the derivatives.
→ **Computation on the board.**
Error estimation for regularized derivatives

For $f_\alpha$ determined by

$$-\alpha \partial_{xx} f_\alpha(x) + f_\alpha(x) = f^\delta(x)$$

(RD)

and twice continuously differentiable $f$, we can choose $\alpha$ such that

$$\| \partial_x f_\alpha - \partial_x f \|_2 \leq \sqrt{C} \sqrt{\delta},$$

although

$$\| f^\delta - f \|_2 \leq \delta.$$

Observation

Even when regularization is used, the order of the reconstruction error is worse than the order of the error in the data.
Error estimation for regularized derivatives

For \( f_\alpha \) determined by

\[-\alpha \partial_{xx} f_\alpha(x) + f_\alpha(x) = f_\delta(x),\quad \text{(RD)}\]

let us make a (seemingly small) change and only assume that \( f \) is one time continuously differentiable.

Computation on the board shows:
Error estimation for regularized derivatives

For \( f_\alpha \) determined by

\[-\alpha \partial_{xx} f_\alpha (x) + f_\alpha (x) = f_\delta (x), \quad (RD)\]

let us make a (seemingly small) change and only assume that \( f \) is one time continuously differentiable.

**Computation on the board shows:**

**Observation**

Without additional smoothness assumptions on the exact solution, the convergence of the regularized solutions is arbitrarily slow!
Looking ahead...

What did we do by computing $-\alpha \partial_{xx} f_\alpha(x) + f_\alpha(x) = f^\delta(x)$?
Looking ahead...

What did we do by computing $-\alpha \partial_{xx} f_\alpha(x) + f_\alpha(x) = f^\delta(x)$?

In this lecture we will learn that $f_\alpha$ as computed above, solves

$$f_\alpha = \arg \min_u \| u - f^\delta \|^2_2 + \alpha \| \partial_x u \|^2_2.$$
Looking ahead...

What did we do by computing \(-\alpha \partial_{xx} f_\alpha(x) + f_\alpha(x) = f_\delta(x)\)?

In this lecture we will learn that \(f_\alpha\) as computed above, solves

\[
    f_\alpha = \arg \min_u \|u - f_\delta\|^2_2 + \alpha \|\partial_x u\|^2_2.
\]

→ Tikhonov regularization!
Discrete differentiation by finite differences

In practice: Evaluations $f_i$ of the function at grid points $x_i$ with $x_{i+1} - x_i = h$. Typically, one computes

\[
\partial_x f(x_i) \approx f_i - f_{i-1} \quad \text{left sided differences}
\]

\[
\partial_x f(x_i) \approx f_{i+1} - f_i \quad \text{right sided differences}
\]

\[
\partial_x f(x_i) \approx \frac{f_{i+1} - f_{i-1}}{2h} \quad \text{central differences}
\]

We will see in the exercises:

- The finite difference approximation of the derivative gets better as $h$ decreases.
- The error due to taking finite differences of noisy data increases as $h$ decreases.

→ The step size has to be chosen carefully!
Discrete differentiation by finite differences

In practice: Evaluations $f_i$ of the function at grid points $x_i$ with $x_{i+1} - x_i = h$. Typically, one computes

\[
\partial_x f(x_i) \approx \frac{f_i - f_{i-1}}{h} \quad \text{left sided differences}
\]
\[
\partial_x f(x_i) \approx \frac{f_{i+1} - f_i}{h} \quad \text{right sided differences}
\]
\[
\partial_x f(x_i) \approx \frac{f_{i+1} - f_{i-1}}{2h} \quad \text{central differences}
\]
Discrete differentiation by finite differences

In practice: Evaluations $f_i$ of the function at grid points $x_i$ with $x_{i+1} - x_i = h$. Typically, one computes

\[
\partial_x f(x_i) \approx \frac{f_i - f_{i-1}}{h} \quad \text{left sided differences}
\]
\[
\partial_x f(x_i) \approx \frac{f_{i+1} - f_i}{h} \quad \text{right sided differences}
\]
\[
\partial_x f(x_i) \approx \frac{f_{i+1} - f_{i-1}}{2h} \quad \text{central differences}
\]

We will see in the exercises:

- The finite difference approximation of the derivative gets better as $h$ decreases.
- The error due to taking finite differences of noisy data increases as $h$ decreases.

→ The step size has to be chosen carefully!
Summary: Differentiation

What we have learned about the differentiation of a function:

1. Without regularization an arbitrarily small error in the data can lead to an arbitrarily large error in the derivative!

2. Even with regularization the order with which the error in the derivative decays is worse than the order of the error in the data!

3. We need additional smoothness assumption on the true function $f$ to even derive an estimate on the error!
Heat Equation

The heat equation with zero Dirichlet boundary conditions is given by the following partial differential equation (PDE):

\[ \partial_t u(x, t) = \partial_{xx} u(x, t) \quad \text{for } x \in ]0, \pi[, \quad t \in \mathbb{R}^+ \]

\[ u(0, t) = 0 \quad \forall t \in \mathbb{R}^+ \]

\[ u(\pi, t) = 0 \quad \forall t \in \mathbb{R}^+ \]

\[ u(x, 0) = f(x) \quad \text{for } x \in ]0, \pi[ \]

Naive idea:

\[ \frac{u_{i,j+1} - u_{i,j}}{\Delta t} \approx \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{\Delta x} \]

Does it work both ways - forward and backward?
The 1d heat equation

First homework:
\[ f(y) = \sin(my), \quad \forall \delta, T, \exists m : \|u(\cdot, T) - 0\|_2 \leq \delta, \text{ but } \|f\|_{\infty} = 1. \]

Conclusion from homework

The backward heat equation is ill-posed!

A computation on the board shows:
The 1d heat equation

First homework:
\[ f(y) = \sin(my), \quad \forall \delta, T, \exists m : \|u(\cdot, T) - 0\|_2 \leq \delta, \text{ but } \|f\|_\infty = 1. \]

Conclusion from homework
The backward heat equation is ill-posed!

A computation on the board shows:

Solution of the heat equation
The solution of the 1d heat equation with zero Dirichlet boundary conditions is given by

\[ u(x, t) = \int_0^\pi k(x, y, t)f(y) \, dy, \]

\[ k(x, y, t) = \frac{2}{\pi} \sum_{n=1}^{\infty} e^{-n^2 t} \sin(nx) \sin(ny). \]
Deblurring

Examples of Ill-Posed Problems
Michael Moeller

Deblurring

Original image
Deblurring

Blurry image $f = k \ast u$
Deblurring

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

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

Reconstruction by $\mathcal{F}^{-1}(\mathcal{F}(f)/\mathcal{F}(k))$
## Some things about image processing

### What is an image?

**Continuous representation:** Function $u : \Omega \rightarrow \mathbb{R}$ (grayscale) or $u : \Omega \rightarrow \mathbb{R}^3$ (color), where $\Omega \subset \mathbb{R}^2$ (typically open and bounded).

**Discrete representation:** Matrix $u \in \mathbb{R}^{n \times m}$ (grayscale) or three matrices $u_1, u_2, u_3 \in \mathbb{R}^{n \times m}$ (color). The discrete points/entries of the matrix are called *pixels*. 
Blurring

Continuous model for a blurred image:

\[ f(x, y) = \int_{\mathbb{R}^2} k(s - x, t - y)u(s, t) \, ds \, dt \]

with a convolution kernel \( k \), e.g.

\[ k(s, t) = \frac{1}{2\pi \sigma^2} \exp \left( -\frac{s^2 + t^2}{2\sigma^2} \right) \]
Deblurring

The Fourier Theorem states that

\[ f = k \ast u \Rightarrow \mathcal{F}(f) = \mathcal{F}(k)\mathcal{F}(u). \]

### Riemann-Lebesgue Lemma

Let \( k : \mathbb{R}^2 \to \mathbb{R} \) be absolutely integrable, i.e.

\[
\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} |k(x, y)| \, dx \, dy < \infty.
\]

Then \( |\mathcal{F}(k)(\mu, \nu)| \to 0 \) for \( \| (\mu, \nu) \| \to \infty \).
Deblurring

The Fourier Theorem states that

\[ f = k \ast u \implies \mathcal{F}(f) = \mathcal{F}(k)\mathcal{F}(u). \]

Riemann-Lebesgue Lemma

Let \( k : \mathbb{R}^2 \to \mathbb{R} \) be absolutely integrable, i.e.

\[
\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} |k(x, y)| \, dx \, dy < \infty.
\]

Then \( |\mathcal{F}(k)(\mu, \nu)| \to 0 \) for \( \|(\mu, \nu)\| \to \infty \).

The reconstruction of

\[ u = \mathcal{F}^{-1} \left( \frac{\mathcal{F}(f)}{\mathcal{F}(k)} \right) \]

becomes unstable!
Deblurring - discretizations

How can we discretize a blur?

**Image:** $u \in \mathbb{R}^{n \times m}$.

**Blur kernel:** $k \in \mathbb{R}^{r \times r}$, typically $r << \min\{n, m\}$. Assume zero values outside.
Deblurring - discretizations

How can we discretize a blur?

**Image:** $u \in \mathbb{R}^{n \times m}$.

**Blur kernel:** $k \in \mathbb{R}^{r \times r}$, typically $r << \min\{n, m\}$. Assume zero values outside. For example

$$k = \begin{pmatrix}
0.0030 & 0.0133 & 0.0219 & 0.0133 & 0.0030 \\
0.0133 & 0.0596 & 0.0983 & 0.0596 & 0.0133 \\
0.0219 & 0.0983 & 0.1621 & 0.0983 & 0.0219 \\
0.0133 & 0.0596 & 0.0983 & 0.0596 & 0.0133 \\
0.0030 & 0.0133 & 0.0219 & 0.0133 & 0.0030
\end{pmatrix}$$

Let $r$ be odd.

$$f_{i,j} = \sum_{h=1}^{r} \sum_{l=1}^{r} k_{h,r} u_{i+h-\frac{r+1}{2}, j+l-\frac{r+1}{2}}$$
Deblurring - discretizations

How can we discretize a Gaussian blur?

**Image:** $u \in \mathbb{R}^{n \times m}$.

**Gaussian blur kernel:** Separable!
Deblurring - discretizations

How can we discretize a Gaussian blur?

**Image:** $u \in \mathbb{R}^{n \times m}$.

**Gaussian blur kernel:** Separable!

$$f(x, y) = \frac{1}{2\pi\sigma^2} \int_{\Omega} \exp \left( -\frac{(s - x)^2 + (t - y)^2}{2\sigma^2} \right) u(s, t) \, ds \, dt$$

$$= \frac{1}{\sqrt{2\pi\sigma}} \int exp \left( -\frac{(t - y)^2}{2\sigma^2} \right) v(x, t) \, dt$$

with

$$v(x, t) = \frac{1}{\sqrt{2\pi\sigma}} \int exp \left( -\frac{(s - x)^2}{2\sigma^2} \right) u(s, t) \, ds$$

We only have to do two 1d convolutions!
Deblurring - discretizations

Gaussian blur kernel: Separable!

\[ k = \begin{pmatrix} 0.1615 & 0.2180 & 0.2409 & 0.2180 & 0.1615 \end{pmatrix} \]

\[ f_{i,j} = \sum_{h=1}^{r} k_h \sum_{l=1}^{r} k_l \ u_{i+h-\frac{r+1}{2},j+l-\frac{r+1}{2}} \]

Or in the pure 1d case

\[ c_j = \sum_{h=1}^{r} k_h b_{i+h-\frac{r+1}{2}} \]

can be written as

\[ \vec{c} = \begin{pmatrix} \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & k & 0 & \cdots & 0 \\ 0 & 0 & k & \cdots & 0 \\ 0 & \cdots & 0 & k & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{pmatrix} \begin{pmatrix} \vec{b} \end{pmatrix} \]

\[ A_1 \]
Deblurring - discretizations

\[ \tilde{c} = \begin{pmatrix} \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & k & 0 & \ldots & 0 \\ 0 & 0 & k & \ldots & 0 \\ 0 & \ldots & 0 & k & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{pmatrix} \begin{pmatrix} b_0 \\ b_1 \\ \vdots \end{pmatrix} \]

What happens at the boundary? What are \( b_0, b_{-1}, \ldots \)?
Deblurring - discretizations

\[ \vec{c} = \begin{pmatrix} \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & k & 0 & \cdots & 0 \\ 0 & 0 & k & \cdots & 0 \\ 0 & \cdots & 0 & k & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{pmatrix} \begin{pmatrix} b \\ A_1 \end{pmatrix} \] 

What happens at the boundary? What are \( b_0, b_{-1}, \ldots \)?

Most common assumption for image blurring:
\( b_h = b_1 \) for \( h \leq 1 \), \( b_h = b_n \) for \( h \geq n \).
Deblurring - discretizations

\[ \vec{c} = \begin{pmatrix} \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & k & 0 & \cdots & 0 \\ 0 & 0 & k & \cdots & 0 \\ 0 & \cdots & 0 & k & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{pmatrix} \begin{pmatrix} \vec{b} \\ A_1 \end{pmatrix} \]

What happens at the boundary? What are \( b_0, b_{-1}, \ldots \)?

Most common assumption for image blurring:
\( b_h = b_1 \) for \( h \leq 1 \), \( b_h = b_n \) for \( h \geq n \).

First rows of the matrix \( A_1 \):

\[
\begin{pmatrix}
    k_1 + k_2 + k_3 & k_4 & k_5 & \cdots & 0 \\
    k_1 + k_2 & k_3 & k_4 & \cdots & 0 \\
    k_1 & k_2 & k_3 & \cdots & 0 \\
    0 & k & 0 & \cdots & 0 \\
    \vdots & \vdots & \vdots & \vdots & \vdots
\end{pmatrix}
\]
Deblurring - discretizations

- We know how to write a 1d blur as \( c = A_1 b \). For image \( u \):
  \[
  v = A_1 u \in \mathbb{R}^{n \times m}.
  \]

- Now we need to blur \( v \) in \( x \)-direction. Generate matrix \( A_2 \in \mathbb{R}^{m \times m} \) with the kernel \( k \) appearing in the columns, and the boundaries treated similar to the \( y \)-direction case.

- Compute
  \[
  f = A_1 u A_2.
  \]
Deblurring - discretizations

- We know how to write a 1d blur as $c = A_1 b$. For image $u$:
  \[ \nu = A_1 u \in \mathbb{R}^{n \times m}. \]

- Now we need to blur $\nu$ in $x$-direction. Generate matrix $A_2 \in \mathbb{R}^{m \times m}$ with the kernel $k$ appearing in the columns, and the boundaries treated similar to the $y$-direction case.

- Compute
  \[ f = A_1 u A_2. \]

- Kronecker product:
  \[
  A \otimes B = \begin{pmatrix}
  A_{1,1}B & A_{1,2}B & \ldots & A_{1,m}B \\
  A_{2,1}B & A_{2,2}B & \ldots & A_{2,m}B \\
  \vdots & \vdots & \ddots & \vdots \\
  A_{m,1}B & A_{m,2}B & \ldots & A_{m,m}B
  \end{pmatrix} \in \mathbb{R}^{nm \times nm}
  \]
Deblurring - discretizations

- We know how to write a 1d blur as \( c = A_1 b \). For image \( u \):

\[
v = A_1 u \in \mathbb{R}^{n \times m}.
\]

- Now we need to blur \( v \) in \( x \)-direction. Generate matrix \( A_2 \in \mathbb{R}^{m \times m} \) with the kernel \( k \) appearing in the columns, and the boundaries treated similar to the \( y \)-direction case.

- Compute

\[
f = A_1 u A_2.
\]

- **Kronecker product:**

\[
A \otimes B = \begin{pmatrix}
A_{1,1}B & A_{1,2}B & \cdots & A_{1,m}B \\
A_{2,1}B & A_{2,2}B & \cdots & A_{2,m}B \\
\vdots & \vdots & \ddots & \vdots \\
A_{m,1}B & A_{m,2}B & \cdots & A_{m,m}B
\end{pmatrix} \in \mathbb{R}^{nm \times nm}
\]

**Vectorization**

\[
f = A_1 u A_2 \iff \text{vec}(f) = (A_2^T \otimes A_1)\text{vec}(u).
\]
Deblurring as a linear equation

- We have discretized $f = k \ast u$ as

$$\vec{f} = A\vec{u}.$$

- Matlab: $A$ is invertible!
- Unique solution with $A^{-1} \rightarrow$ Problem not ill-posed?!?
Deblurring as a linear equation

- We have discretized $f = k \ast u$ as
  \[
  \vec{f} = A\vec{u}.
  \]
- Matlab: $A$ is invertible!
- Unique solution with $A^{-1} \rightarrow$ Problem not ill-posed?!?
- Give it a try! Use backslash.
Deblurring as a linear equation

- We have discretized $f = k * u$ as $\vec{f} = A\vec{u}$.
- Matlab: $A$ is invertible!
- Unique solution with $A^{-1}$ → Problem not ill-posed?!?
- Give it a try! Use backslash.

How is this possible?
Deblurring as a linear equation

- We have discretized $f = k \ast u$ as
  $$\vec{f} = A\tilde{u}.$$  

- Matlab: $A$ is invertible!

- Unique solution with $A^{-1}$ → Problem not ill-posed?!?

- Give it a try! Use backslash.

How is this possible?
Examples of Ill-Posed Problems
Michael Moeller

Implementation

- When writing a convolution as a matrix vector multiplication, always use sparse matrices!

- A full double matrix \((A_2^T \otimes A_1)\) for a \(256 \times 256\) image is over 34GB!

- See “help spdiags“ in Matlab.

- See “help kron” in Matlab.

- See “help reshape” in Matlab.