1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26

Lecture 17: Global Filters I: Discrete Variational Methods

Contents

1. Motivation
2. The One-Dimensional Case
3. Thomas Algorithm for Tridiagonal Systems
4. The m-Dimensional Case

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26

Motivation

- So far our image enhancement methods were more or less localised:
  - point operations
  - (linear or nonlinear) local methods analysing a patch around each pixel
  - nonlocal means where patches are compared within some neighbourhood
- Moreover, they were motivated heuristically and their optimality was unclear.
- Today we start considering our first *global* methods:
  The filtered image satisfies a global optimality criterion.
- Such approaches are called *variational methods* or *regularisation methods*.
- Both discrete and continuous models play a role.
- They allow a transparent mathematical modelling.
- Optimality is one key issue of modern image processing and computer vision.
- Optimality principles will also be useful for applications beyond denoising:
  deblurring, segmentation, motion analysis, stereo, shape-from-shading.

The One-Dimensional Case (1)

M I A
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26

The One-Dimensional Case

Goals

- find a smoothing process that satisfies an optimality criterion
- clearly state all model assumptions without any hidden assumptions
- filtered discrete image $\boldsymbol{u}$ should be
  - close to the original image $\boldsymbol{f}$
  - as smooth as possible
- approach that can be modified such that one obtains edge-preserving smoothing
- automatically correct treatment at boundaries of the image domain

The One-Dimensional Case (2)

M I A
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26

A Simple 1-D Approach

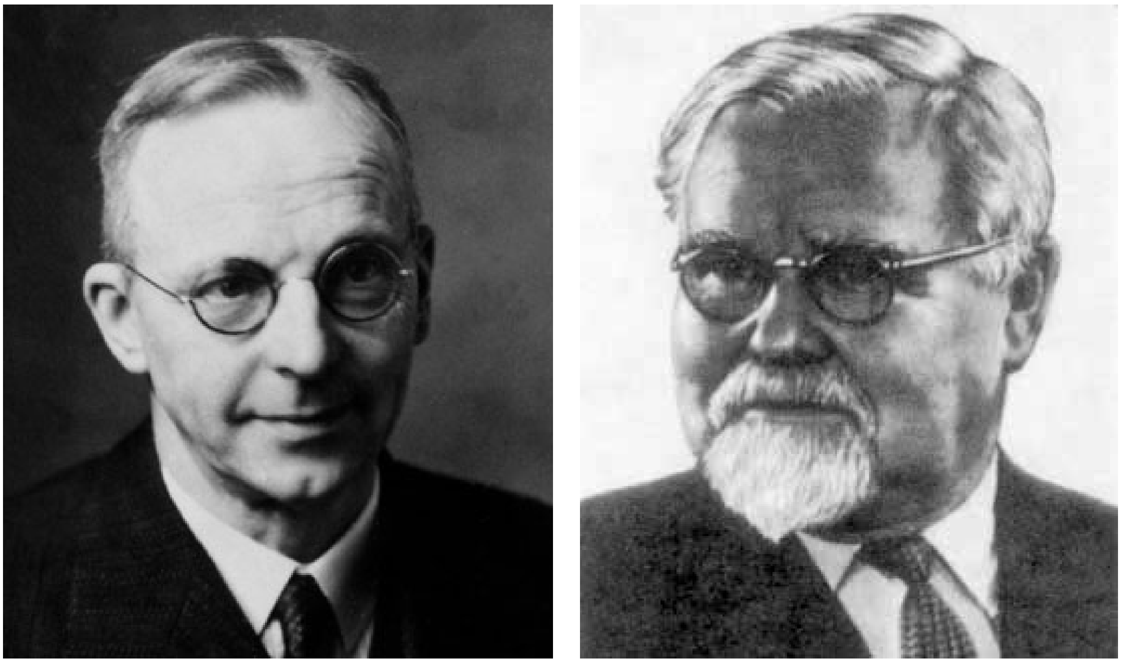
- goal: smooth a one-dimensional discrete signal $\boldsymbol{f} = (f_1, ..., f_N)^\top$
- filtered signal $\boldsymbol{u} = (u_1, ..., u_N)^\top$ as minimiser of the cost function *(energy)*

$$E(\boldsymbol{u}) \;:=\; \frac{1}{2}\underbrace{\sum_{k=1}^{N}(u_k - f_k)^2}_{\text{similarity}} + \frac{\alpha}{2}\underbrace{\sum_{k=1}^{N-1}(u_{k+1} - u_k)^2}_{\text{smoothness}}$$

- first term rewards similarity to $\boldsymbol{f}$: *data term*
- second term rewards smoothness of $\boldsymbol{u}$: *smoothness term*
- smoothness weight $\alpha > 0$ is called *regularisation parameter:*
  larger values yield smoother (i.e. less fluctuating) solutions
- Models with quadratic smoothness terms are often called Tikhonov regularisation.
  However, they have been studied earlier by Whittaker (1923).
  Thus, we call them *Whittaker–Tikhonov regularisation*.
- Quadratic optimisation models are convenient: Their derivatives give linear models.

The One-Dimensional Case (3)

M I A
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26



Left: Edmund Taylor Whittaker (1873–1956) proposed a discrete model for data regularisation already in 1923. He was also involved in the discovery of the sampling theorem (Lecture 5). Source: `http://www-history.mcs.st-andrews.ac.uk/PictDisplay/Whittaker.html`. **Right:** Andrei Nikolaevich Tikhonov (1906–1993) did extensive research on regularisation methods for solving ill-posed problems. Source: `http://turnbull.mcs.st-and.ac.uk/history/PictDisplay/Tikhonov.html`.

The One-Dimensional Case (4)

M I A
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26

Necessary Condition for a Minimum:

- first partial derivatives w.r.t. u_1,...,u_N must vanish:

$$\begin{aligned} 0 &= \frac{\partial E}{\partial u_1} = u_1 - f_1 + \alpha\,(u_1 - u_2), \\ 0 &= \frac{\partial E}{\partial u_i} = u_i - f_i + \alpha\,(-u_{i+1} + 2u_i - u_{i-1}) \qquad (i = 2, ..., N-1), \\ 0 &= \frac{\partial E}{\partial u_N} = u_N - f_N + \alpha\,(u_N - u_{N-1}). \end{aligned}$$

- Note that the boundaries are handled without additional assumptions.
- yields linear system of equations $\boldsymbol{B}\boldsymbol{u} = \boldsymbol{f}$ with unknown $\boldsymbol{u} = (u_1, ..., u_N)^\top$:

$$\begin{pmatrix} 1+\alpha & -\alpha & & & \\ -\alpha & 1+2\alpha & -\alpha & & \\ & \ddots & \ddots & \ddots & \\ & & -\alpha & 1+2\alpha & -\alpha \\ & & & -\alpha & 1+\alpha \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_{N-1} \\ u_N \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_{N-1} \\ f_N \end{pmatrix}.$$

The One-Dimensional Case (5)

M I A
1 2
3 4
5 6
7 8
9 10
11 12
13 14
15 16
17 18
19 20
21 22
23 24
25 26

Properties of This Approach

- Image domain boundaries are automatically taken into account.
- The system matrix $\boldsymbol{B}$ is *strictly diagonally dominant*, i.e.

$$|b_{i,i}| > \sum_{j,j\neq i} |b_{i,j}| \quad \text{for all } i.$$

- Strictly diagonally dominant matrices are invertible by *Gershgorin's theorem:*

  The eigenvalues of an $N \times N$ matrix $\boldsymbol{A} = (a_{i,j})$ lie in the union of all discs K_i with centre $a_{i,i}$ and radius $\sum\limits_{j,j\neq i} |a_{i,j}|$.

- Moreover, for our specific system matrix $\boldsymbol{B}$, the following can be shown:
  $\boldsymbol{B}^{-1}$ is a fully populated matrix with positive entries only !
- Thus, $\boldsymbol{u}$ results from $\boldsymbol{f}$ via averaging over a mask that does not vanish within the entire signal length.
- Such filters are called *IIR filters:* infinite impulse response.
- So far, most of our convolution filters were *FIR filters:*
  finite impulse response (convolution kernel has finite support).

The One-Dimensional Case (6)

M I A
1 2
3 4
5 6
7 8
9 10
11 12
13 14
15 16
17 18
19 20
21 22
23 24
25 26

Is This Solution Really a Global Minimum ?

- It is sufficient to show that $E(\boldsymbol{u})$ is *strictly convex* in $\boldsymbol{u}$, i.e.

$$E\big(\beta\boldsymbol{u} + (1-\beta)\boldsymbol{v}\big) \;<\; \beta\, E(\boldsymbol{u}) \,+\, (1-\beta)\, E(\boldsymbol{v})$$

  for all $0 < \beta < 1$ and for all $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^N$ with $\boldsymbol{u} \neq \boldsymbol{v}$.

- A strictly convex function $E(\boldsymbol{u})$ has a single extremum.
  Moreover, this extremum is a minimum.

Forthcoming Assignment:

- Show that in our case, $E(\boldsymbol{u})$ is strictly convex.
- Hint: Use the strict convexity of $g(s) = s^2$.
  (Functions with positive second derivative are strictly convex.)

Thomas Algorithm for Tridiagonal Systems (1)

M I A
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26

Thomas Algorithm for Tridiagonal Systems

- We have seen that the discrete 1-D variational method creates the linear system

$$\begin{pmatrix} 1+\alpha & -\alpha & & & \\ -\alpha & 1+2\alpha & -\alpha & & \\ & \ddots & \ddots & \ddots & \\ & & -\alpha & 1+2\alpha & -\alpha \\ & & & -\alpha & 1+\alpha \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_{N-1} \\ u_N \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_{N-1} \\ f_N \end{pmatrix}.$$

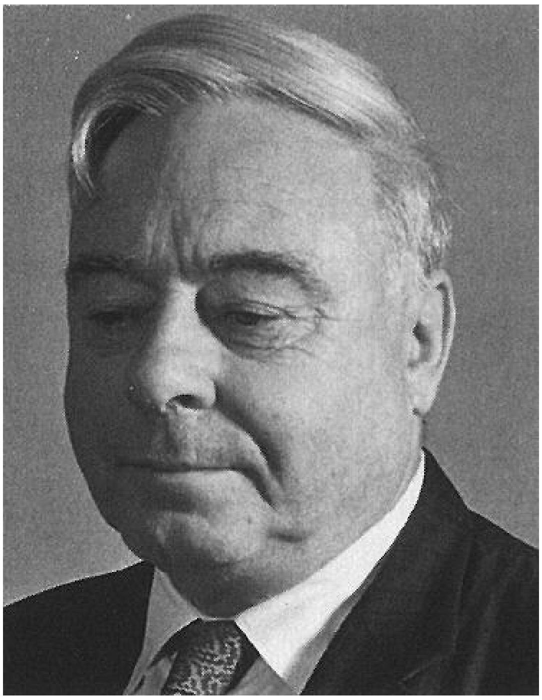
- A matrix where all nonvanishing entries are located in the diagonal and the neighbouring left and right off-diagonal is called *tridiagonal matrix*.

- Tridiagonal matrices appear frequently in 1-D problems (see also cubic B-spline interpolation in Lecture 9).

- For tridiagonal linear systems already a simple variant of the Gauß elimination algorithm is highly efficient. In the English literature this is often called *Thomas algorithm,* since it has been used by Llewellen Hilleth Thomas in 1949.

Thomas Algorithm for Tridiagonal Systems (2)

M I A
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26



Left: Carl–Friedrich Gauß (1777–1855) is regarded as one of the most famous mathematicians. Source: `http://www-gap.dcs.st-and.ac.uk/~history/PictDisplay/Gauss.html`. **Right:** Llewellen Hilleth Thomas (1903–1992) was a physicist, applied mathematician and computer scientist. He invented core memory in 1946. Source: `http://www.columbia.edu/acis/history/thomas.html`.

Thomas Algorithm for Tridiagonal Systems (3)

M I A
1 2
3 4
5 6
7 8
9 10
11 12
13 14
15 16
17 18
19 20
21 22
23 24
25 26

The Thomas Algorithm in Three Steps

We want to solve a linear system $\boldsymbol{Bu} = \boldsymbol{d}$ with

$$\boldsymbol{B} = \begin{pmatrix} \alpha_1 & \beta_1 & & & \\ \gamma_1 & \alpha_2 & \beta_2 & & \\ & \ddots & \ddots & \ddots & \\ & & \gamma_{N-2} & \alpha_{N-1} & \beta_{N-1} \\ & & & \gamma_{N-1} & \alpha_N \end{pmatrix}.$$

This is done in three steps:

1. **LR Decomposition:**
   $\boldsymbol{B} = \boldsymbol{LR}$ with a lower bidiagonal matrix $\boldsymbol{L}$ and an upper bidiagonal matrix $\boldsymbol{R}$.
2. **Forward Elimination:**
   Solve $\boldsymbol{Ly} = \boldsymbol{d}$ for $\boldsymbol{y}$.
3. **Backward Substitution:**
   Solve $\boldsymbol{Ru} = \boldsymbol{y}$ for $\boldsymbol{u}$.

Let us investigate these steps in more detail.

Thomas Algorithm for Tridiagonal Systems (4)

M I A
1 2
3 4
5 6
7 8
9 10
11 12
13 14
15 16
17 18
19 20
21 22
23 24
25 26

Step 1: LR Decomposition

We write $\boldsymbol{B}$ as a product of two simpler matrices:

- a lower bidiagonal matrix $\boldsymbol{L}$ with diagonal entries 1
- an upper bidiagonal matrix $\boldsymbol{R}$

$$\boldsymbol{B} \;=\; \boldsymbol{LR} \;=\; \begin{pmatrix} 1 & & & & \\ \ell_1 & 1 & & & \\ & \ddots & \ddots & & \\ & & \ell_{N-2} & 1 & \\ & & & \ell_{N-1} & 1 \end{pmatrix} \begin{pmatrix} m_1 & r_1 & & & \\ & m_2 & r_2 & & \\ & & \ddots & \ddots & \\ & & & m_{N-1} & r_{N-1} \\ & & & & m_N \end{pmatrix}$$

$$= \begin{pmatrix} m_1 & r_1 & & & \\ \ell_1 m_1 & \ell_1 r_1 + m_2 & r_2 & & \\ & \ddots & \ddots & \ddots & \\ & & \ell_{N-2}\, m_{N-2} & \ell_{N-2}\, r_{N-2} + m_{N-1} & r_{N-1} \\ & & & \ell_{N-1}\, m_{N-1} & \ell_{N-1}\, r_{N-1} + m_N \end{pmatrix}$$

Thomas Algorithm for Tridiagonal Systems (5)

M I A
1 2
3 4
5 6
7 8
9 10
11 12
13 14
15 16
17 18
19 20
21 22
23 24
25 26

Comparing this with the coefficients of $\boldsymbol{B}$ shows that

$$r_i = \beta_i \qquad (i = 1, ..., N-1).$$

Moreover, the coefficients m_i and ℓ_i can be computed via

```
$m_1 := \alpha_1$
for $i = 1, 2, ..., N-1$:
    $\ell_i := \gamma_i / m_i$
    $m_{i+1} := \alpha_{i+1} - \ell_i \beta_i$
```

Thomas Algorithm for Tridiagonal Systems (6)

M I A
1 2
3 4
5 6
7 8
9 10
11 12
13 14
15 16
17 18
19 20
21 22
23 24
25 26

Step 2: Forward Elimination

We solve $\boldsymbol{L}\boldsymbol{y} = \boldsymbol{d}$ for $\boldsymbol{y}$:

$$\begin{pmatrix} 1 & & & & \\ \ell_1 & 1 & & & \\ & \ddots & \ddots & & \\ & & \ell_{N-2} & 1 & \\ & & & \ell_{N-1} & 1 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_{N-1} \\ y_N \end{pmatrix} = \begin{pmatrix} d_1 \\ d_2 \\ \vdots \\ d_{N-1} \\ d_N \end{pmatrix}$$

Proceeding from the first to the last equation gives

```
$y_1 := d_1$
for $i = 2, 3, ..., N$:
    $y_i := d_i - \ell_{i-1}\, y_{i-1}$
```

Thomas Algorithm for Tridiagonal Systems (7)

M I A
1 2
3 4
5 6
7 8
9 10
11 12
13 14
15 16
17 18
19 20
21 22
23 24
25 26

Step 3: Backward Substitution

We solve $\boldsymbol{Ru} = \boldsymbol{y}$ for $\boldsymbol{u}$ (using $r_i = \beta_i$ for $i = 1,\ldots,N-1$):

$$\begin{pmatrix} m_1 & \beta_1 & & & \\ & m_2 & \beta_2 & & \\ & & \ddots & \ddots & \\ & & & m_{N-1} & \beta_{N-1} \\ & & & & m_N \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_{N-1} \\ u_N \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_{N-1} \\ y_N \end{pmatrix}$$

Proceeding from the last to the first equation gives

```
$u_N := y_N / m_N$
for $i = N-1, N-2, \ldots, 1$:
    $u_i := (y_i - \beta_i \, u_{i+1}) / m_i$
```

Thomas Algorithm for Tridiagonal Systems (8)

M I A
1 2
3 4
5 6
7 8
9 10
11 12
13 14
15 16
17 18
19 20
21 22
23 24
25 26

Remarks

- The Thomas algorithm can be shown to be stable for every strictly diagonally dominant system.
- It can be regarded as *recursive filtering*:
  - step 1: computation of the filter coefficients
  - step 2: *causal* filter $\longrightarrow \bullet \longrightarrow \bullet \longrightarrow \bullet \longrightarrow$
  - step 3: *anticausal (acausal)* filter $\longleftarrow \bullet \longleftarrow \bullet \longleftarrow \bullet \longleftarrow$
- The algorithm is highly efficient: It requires only
  - $(N-1) + N \;=\; 2N-1$ divisions,
  - $(N-1) + (N-1) + (N-1) \;=\; 3N-3$ multiplications,
  - $(N-1) + (N-1) + (N-1) \;=\; 3N-3$ subtractions.

  Thus, its complexity is linear in N.
- The memory requirement is also linear in N.

The m-Dimensional Case (1)

The m-Dimensional Case

- Use a single index in order to count all pixels of an m-D image $\boldsymbol{f} = (f_k)$.
  Let $\mathcal{N}(k)$ be the set of neighbours of pixel k
  (in 2-D: 4 for inner points, 3 for boundary points, 2 for corner points).

- Then the m-dimensional energy is given by

$$E(\boldsymbol{u}) \;:=\; \frac{1}{2}\sum_{k=1}^{N}\left((u_k - f_k)^2 + \frac{\alpha}{2}\sum_{j\in\mathcal{N}(k)}(u_j - u_k)^2\right).$$

- Minimisation of $E(\boldsymbol{u})$ yields the linear system

$$u_i + \alpha \sum_{j\in\mathcal{N}(i)} (u_i - u_j) \;=\; f_i \qquad (i = 1, ..., N).$$

- not (exactly) separable into simpler 1-D problems

M I
A
1 2
3 4
5 6
7 8
9 10
11 12
13 14
15 16
17 18
19 20
21 22
23 24
25 26

The m-Dimensional Case (2)

Structure of the Linear System

- The system matrix is large, symmetric and sparse (dünn besetzt).

- **Example:**
  - $256^2 = 65536$ unknowns for an image of size 256×256
  - yields a system matrix of size 65536×65536, i.e. with $4.3 \cdot 10^9$ entries
  - A naive storing of all entries as floats (4 bytes) would require 17 Gigabyte!
  - However, for a four-neighbourhood, the matrix has not more than 5 nonvanishing entries per row.
    Thus, at most $5 \cdot 65536 = 327680$ out of $4.3 \cdot 10^9$ entries are interesting.
    This is only 0.00763 %. The rest are zeroes.

- Direct methods such as a Gaussian algorithm would destroy many zeroes for dimensions $m > 1$, and would lead to a prohibitive computational burden.

- Iterative methods are reasonable alternatives:
  - Jacobi, Gauß-Seidel, SOR methods
  - preconditioned conjugate gradient (PCG) methods
  - multigrid methods

M I
A
1 2
3 4
5 6
7 8
9 10
11 12
13 14
15 16
17 18
19 20
21 22
23 24
25 26

The m-Dimensional Case (3)

M I A
1 2
3 4
5 6
7 8
9 10
11 12
13 14
15 16
17 18
19 20
21 22
23 24
25 26

Jacobi Method (Gesamtschrittverfahren)

- simplest (and slowest) iterative method for solving a linear system $\boldsymbol{Bx} = \boldsymbol{d}$
- Let $\boldsymbol{B} = \boldsymbol{D} - \boldsymbol{N}$ with an invertible diagonal matrix $\boldsymbol{D}$ and a remainder $\boldsymbol{N}$. Then $\boldsymbol{Dx} = \boldsymbol{Nx} + \boldsymbol{d}$ is solved iteratively with the fixed point scheme

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{D}^{-1}(\boldsymbol{Nx}^{(k)} + \boldsymbol{d}) \qquad (k = 0, 1, ...).$$

- low computational effort per iteration if $\boldsymbol{B}$ is sparse: 1 matrix–vector product, 1 vector addition, 1 vector scaling
- only small additional memory requirement: vector $\boldsymbol{x}^{(k)}$
- well-suited for parallel computing (multi-core processors, GPUs)
- converges for any initialisation $\boldsymbol{x}^{(0)}$ if $\boldsymbol{B}$ is strictly diagonally dominant
- residue $\boldsymbol{r}^{(k)} := \boldsymbol{Bx}^{(k)} - \boldsymbol{d}$ gives stopping criterion: stop if $\left|\boldsymbol{r}^{(k)}\right| \le \varepsilon \left|\boldsymbol{r}^{(0)}\right|$.

The m-Dimensional Case (4)

M I A
1 2
3 4
5 6
7 8
9 10
11 12
13 14
15 16
17 18
19 20
21 22
23 24
25 26

Application to Our Problem

- For our linear system of equations,

$$u_i + \alpha \sum_{j \in \mathcal{N}(i)} (u_i - u_j) \;=\; f_i \qquad (i = 1, ..., N),$$

  we move the non-diagonal parts to the right:

$$u_i + \alpha\, |\mathcal{N}(i)|\, u_i \;=\; f_i \;+\; \alpha \sum_{j \in \mathcal{N}(i)} u_j\,,$$

  where $|\mathcal{N}(i)|$ denotes the number of neighbours of pixel i.

- This gives the following iterative scheme for $i = 1,...,N$:

$$u_i^{(k+1)} \;:=\; \frac{f_i + \alpha \sum\limits_{j \in \mathcal{N}(i)} u_j^{(k)}}{1 + \alpha\, |\mathcal{N}(i)|}\;.$$

  This is what you implement. It's very simple.

- As initialisation it is natural to take $\boldsymbol{u}^{(0)} := \boldsymbol{f}$.

The m-Dimensional Case (5)

M I A
1 2
3 4
5 6
7 8
9 10
11 12
13 14
15 16
17 18
19 20
21 22
23 24
25 26

Stability Results

- This scheme performs local averaging of f_i and the neighbours $\{u_j^{(k)} \mid j \in \mathcal{N}(i)\}$.

- The weights $\frac{1}{1+\alpha\,|\mathcal{N}(i)|}$ and $\frac{\alpha}{1+\alpha\,|\mathcal{N}(i)|}$ (appears $|\mathcal{N}(i)|$ times) are nonnegative and sum up to 1.

- From this convex combination it follows that

$$\min_j f_j \;\le\; u_i^{(k)} \;\le\; \max_j f_j \qquad \forall\, i \in \{1, ..., N\}, \quad \forall\, k > 0.$$

  Thus, over- and undershoots cannot appear.

- global convergence, since the system matrix is strictly diagonally dominant

The m-Dimensional Case (6)

M I A
1 2
3 4
5 6
7 8
9 10
11 12
13 14
15 16
17 18
19 20
21 22
23 24
25 26

Can One Speed up the Jacobi Method ?

- better initialisations give faster convergence

- simplest two-grid method:
  - solve the system on a downsampled image
  - interpolate the solution to its original size
  - use it as initialisation for the fine scale *(nested iteration, cascadic multigrid)*

- Pyramid-like downsampling and interpolation gives a simple *multigrid method (Mehrgitterverfahren).*

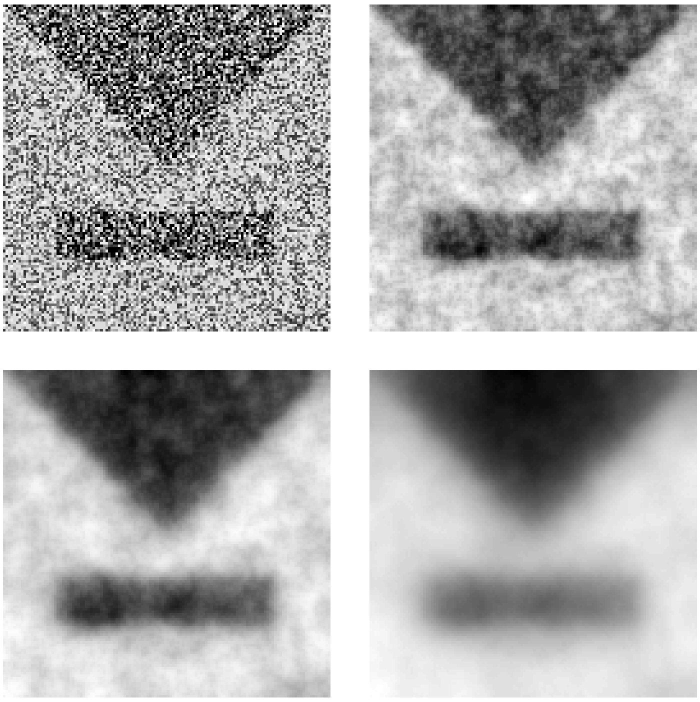
More Efficient Alternatives to the Jacobi Method

- Gauß–Seidel and SOR methods (Einzelschrittverfahren, Relaxationsverfahren)

- preconditioned conjugate gradient methods (PCG)

They are treated in many books on numerical linear algebra.

The m-Dimensional Case (7)



original, 128×128

$\alpha = 5$

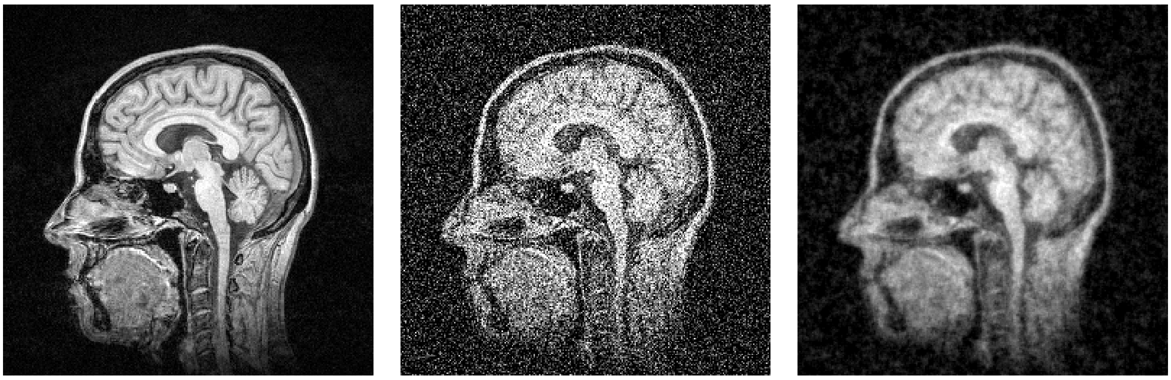
$\alpha = 20$

$\alpha = 100$

Influence of the the regularisation parameter α for variational denoising. For improving visibility, an affine greyscale transformation to $[0, 255]$ has been performed. Author: J. Weickert.

M I A
1 2
3 4
5 6
7 8
9 10
11 12
13 14
15 16
17 18
19 20
21 22
23 24
25 26

The m-Dimensional Case (8)



Left: Test image, 256×256 pixels. **Middle:** With additive Gaussian noise. **Right:** Variational method with $\alpha = 2.05$. Authors: O. Scherzer, J. Weickert.

M I A
1 2
3 4
5 6
7 8
9 10
11 12
13 14
15 16
17 18
19 20
21 22
23 24
25 26

Summary

M I A
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26

Summary

- Discrete variational methods minimise an energy with data and smoothness term.
- They create filter masks with maximal support (IIR filters).
- Image domain boundaries are taken into account automatically.
- The 1-D case leads to a tridiagonal system with a diagonally dominant matrix.
- It can be solved in a stable and efficient way with the Thomas algorithm.
- This algorithm consists of 3 steps:
  LR decomposition, forward elimination, backward substitution.
- It has linear complexity and can be regarded as a recursive filter.
- For dimensions ≥ 2, the system is solved iteratively, e.g. with the Jacobi method.
- Pyramid-like multigrid methods can lead to significant speed-ups.

References

M I A
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26

References

- G. Aubert, P. Kornprobst: *Mathematical Problems in Image Processing: Partial Differential Equations and the Calculus of Variations*. Second Edition, Springer, New York, 2006.
  (one of the best books on variational methods in image analysis)
- E. T. Whittaker: A new method of graduation. *Proceedings of the Edinburgh Mathematical Society*, Vol. 41, 63–75, 1923.
  (early publication on a discrete variational model for data regression)
- H. R. Schwarz, N. Köckler: *Numerische Mathematik*. Achte Auflage, Teubner, Stuttgart, 2011.
  English Edition:
  H. R. Schwarz, J. Waldvogel: *Numerical Analysis: A Comprehensive Introduction.* Wiley, 1989.
  (good description of the Thomas algorithm)
- L. H. Thomas: *Elliptic problems in linear difference equations over a network*. Technical Report, Watson Scientific Computing Laboratory, Columbia University, New York, NJ, 1949.
  (original report by Thomas)
- Y. Saad: *Iterative Methods for Sparse Linear Systems*. SIAM, Philadelphia, Second Edition, 2003.
  (iterative methods for solving linear systems of equations, in particular PCG methods)
- A. Meister: *Numerik linearer Gleichungssysteme*. Vieweg, Braunschweig, 5. Auflage, 2014.
  (excellent German book on numerical methods for linear systems of equations)
- W. L. Briggs, V. E. Henson, S. F. McCormick: *A Multigrid Tutorial*. Second Edition, SIAM, Philadelphia, 2000.
  (well readable introduction to multigrid methods)