Lecture 15:
Image Enhancement VI:
Discrete Variational Methods

Contents

Motivation

- The image enhancement methods we discussed so far were either point operations or local methods.
- Today we consider a *global* method, where the filtered image satisfies a global optimality criterion.
- Such approaches are called *variational methods* or *regularisation methods*.
- Variational methods allow a transparent mathematical modelling.
- They are of similar quality as diffusion filters, and they reveal similar advantages and shortcomings.
- It is even possible to prove that they approximate diffusion filters.

The One-Dimensional Case

Goals:

- smoothing image transformation that satisfies an optimality criterion
- clearly states all model assumptions without any hidden assumptions
- filtered image u should be
  - close to the original image f
  - as smooth as possible
- approach that can be modified such that one obtains edge-preserving smoothing
- automatically correct treatment of image boundaries

A Simple 1-D Approach

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

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

- smoothness weight $\alpha > 0$ is called *regularisation parameter*:
  larger values yield smoother (i.e. less fluctuating) solutions

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

The One-Dimensional Case (3)

Necessary Condition for a Minimum:

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

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

This is a linear system of equations with the unknowns u_1,...,u_N:

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

The One-Dimensional Case (4)

Properties of This Approach

- Image boundaries are automatically treated in a correct way.
- The smoothing effect can be tuned in a continuous way via α.
- The system matrix A is *strictly diagonally dominant*, i.e.

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

  Strictly diagonally dominant matrices are invertible (by Gerschgorin's theorem).
- Morover, for our specific system matrix A, it can be shown that the inverse matrix A^{-1} is a full matrix with positive entries only !
- Thus, $\mathbf{u}$ results from $\mathbf{f}$ 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, many of our convolutions were *FIR filters:* finite impulse response (convolution kernel has finite support)

Is This Solution Really a Global Minimum ?

Yes! If $E_f(\mathbf{u})$ is strictly convex in u, i.e.

$$E_f\left(\beta\mathbf{u}+(1-\beta)\mathbf{v}\right) \;<\; \beta E_f(\mathbf{u})+(1-\beta)E_f(\mathbf{v})$$

for all $0<\beta<1$ and for all $\mathbf{u},\mathbf{v}\in\mathbb{R}^N$ with $\mathbf{u}\neq\mathbf{v}$,
then $E_f(\mathbf{u})$ has a single extremum, and this extremum is a minimum.

Exercise:

Show that in our case, E_f is strictly convex.
Tip: Use the strict convexity of $g(s)=s^2$.
(Functions with positive second derivative are strictly convex.)

Thomas Algorithm for Tridiagonal Systems

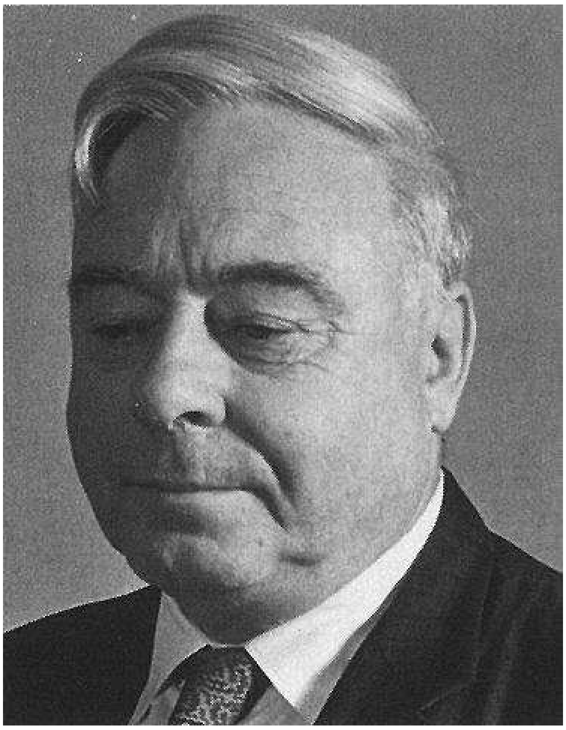
- The discrete 1-D variational method creates the linear system

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

- A matrix where all nonvanishing entries are located in the diagonal and a left and right off-diagonal is called *tridiagonal matrix*.
  Tridiagonal matrices are characteristic for 1-D problems.

- 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 L. H. Thomas in 1949.

Thomas Algorithm for Tridiagonal Systems (2)



Left: Carl–Friedrich Gauß (1777–1855) is regarded as one of the most famous mathematicians. Source: `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: `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

The Thomas Algorithm in Three Steps

We want to solve a linear system $B\mathbf{u} = \mathbf{d}$ with

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

This is done in three steps.

Thomas Algorithm for Tridiagonal Systems (4)

Step 1: LR Decomposition

We decompose B in a product of a lower bidiagonal matrix L and an upper bidiagonal matrix R:

$$B = LR = \begin{pmatrix} 1 & & & & \\ l_1 & 1 & & & \\ & \ddots & \ddots & & \\ & & l_{N-2} & 1 & \\ & & & l_{N-1} & 1 \end{pmatrix} \begin{pmatrix} m_1 & r_1 & & & \\ & m_2 & r_2 & & \\ & & \ddots & \ddots & \\ & & & m_{N-1} & r_{N-1} \\ & & & & m_N \end{pmatrix}$$

Computing this matrix product and comparing the coefficients with the ones for B shows that $r_i = \beta_i$ for all i.
Moreover, the coefficients m_i and l_i are given by

$$\begin{array}{l} m_1 := \alpha_1 \\ \texttt{for}\ \ i = 1, 2, ..., N-1: \\ \qquad l_i := \gamma_i / m_i \\ \qquad m_{i+1} := \alpha_{i+1} - l_i \beta_i \end{array}$$

Thomas Algorithm for Tridiagonal Systems (5)

Step 2: Forward Elimination

We solve $L\mathbf{y} = \mathbf{d}$ for $\mathbf{y}$. This gives

$$\begin{array}{l} y_1 := d_1 \\ \texttt{for}\ \ i = 2, 3, ..., N: \\ \qquad y_i := d_i - l_{i-1} y_{i-1} \end{array}$$

Step 3: Backward Substitution

We solve $R\mathbf{u} = \mathbf{y}$ for $\mathbf{u}$. This leads to

$$\begin{array}{l} u_N := y_N / m_N \\ \texttt{for}\ \ i = N-1, N-2, ..., 1: \\ \qquad u_i := (y_i - \beta_i u_{i+1}) / m_i \end{array}$$

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

Remarks

- The algorithm is 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

$$(N-1) + N + (N-1) \;=\; 3N-2$$

divisions,

$$(N-1) + (N-1) + (N-1) \;=\; 3N-3$$

multiplications, and

$$(N-1) + (N-1) + (N-1) \;=\; 3N-3$$

subtractions. Thus, its complexity is linear in N.

- The memory requirement is also linear in N.

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

The m-Dimensional Case

- Use a single index in order to count all pixels of an m-D image $\mathbf{f} = (f_k)$. Let $\mathcal{N}(k)$ be the set of neighbours of pixel k.
- Then the m-dimensional energy is given by

$$E_f(\mathbf{u}) \;:=\; \frac{1}{2} \sum_{k=1}^{N} \Big((u_k - f_k)^2 + \frac{\alpha}{2} \sum_{l \in \mathcal{N}(k)} (u_l - u_k)^2 \Big).$$

- Minimisation of $E_f(\mathbf{u})$ 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

Structure of the Linear System

- The system matrix is symmetric and sparse (dünn besetzt).
  Example:
  - 65536 unknowns for an image of size 256×256
  - yields a system matrix of size 65536×65536
  - A naive storing of all entries as floats (4 bytes) would take 17 GByte!
  - However, for a four-neighbourhood, the matrix has not more than 5 nonvanishing entries per row.
- 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

Jacobi Method (Gesamtschrittverfahren)

- simplest (and slowest) iterative method for solving a linear system $A\mathbf{x} = \mathbf{b}$.
- converges for any initialisation $\mathbf{x}^{(0)}$ if A is strictly diagonally dominant
- Let $A = D - N$ with a diagonal matrix D and a remainder N.
  Then the problem $D\mathbf{x} = N\mathbf{x} + \mathbf{b}$ is solved iteratively using

$$\mathbf{x}^{(k+1)} = D^{-1}(N\mathbf{x}^{(k)} + \mathbf{b})$$

- low computational effort per iteration if A is sparse:
  1 matrix–vector product, 1 vector addition, 1 vector scaling
- only small additional memory requirement: vector $\mathbf{x}^{(k)}$
- well-suited for parallel computing
- residue $\mathbf{r}^{(k)} := A\mathbf{x}^{(k)} - \mathbf{b}$ allows simple stopping criterion: stop if $\frac{|\mathbf{r}^{(k)}|}{|\mathbf{r}^{(0)}|} < \varepsilon$.

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

Application to Our Problem

- Move diagonal part to the left:

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

$|\mathcal{N}(i)|$ is the number of neighbours of pixel i.
(in 2-D: 4 for inner points, 3 for boundary points, 2 for corner points)

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

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

- As initialisation one takes e.g. $\mathbf{u}^{(0)} := \mathbf{f}$.

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

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

- From this convex combination it follows that

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

Thus, over- and undershoots cannot appear.

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

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

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

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)*.

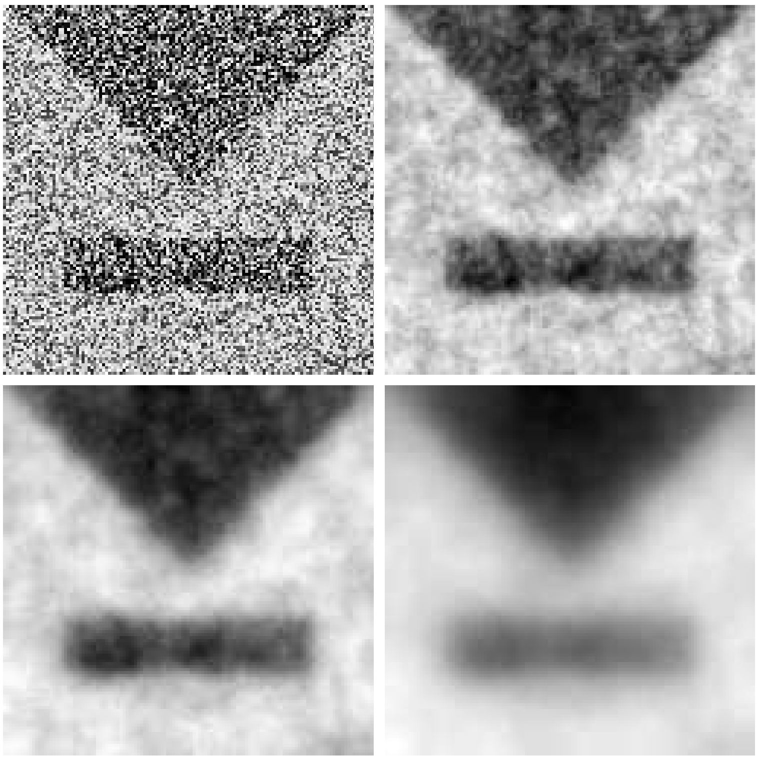
More Efficient Alternatives to the Jacobi Method

- 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)

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



(a) Top left: Test image, 128×128 pixels. **(b) Top right:** Variational method with $\alpha = 5$. **(c) Bottom left:** $\alpha = 20$. **(d) Bottom right:** $\alpha = 100$. For improving visibility, an affine greyscale transformation to $[0, 255]$ has been performed. Author: J. Weickert (2000).

The m-Dimensional Case (8)

(a) Left: Test image, 256×256 pixels. **(b) Middle:** With additive Gaussian noise, SNR $= 0$ dB. **(c) Right:** Variational method with $\alpha = 2.05$. This gives SNR $= 9.61$ dB. Authors: O. Scherzer, J. Weickert (1999).

Summary (1)

Summary

- Discrete variational methods minimise an energy with data and smoothness term.
- They create filter masks with maximal support (IIR filters).
- Image boundaries are automatically treated in a correct way.
- The 1-D case leads to a tridiagonal system with a diagonally dominant matrix.
- Such a system 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 should be solved iteratively (e.g. using the Jacobi method). Pyramid-like multigrid methods can lead to significant speed ups.

Literature

- H. R. Schwarz: *Numerische Mathematik*. Vierte Auflage, Teubner, Stuttgart, 1997.
  (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)
- 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)
- Y. Saad: *Iterative Methods for Sparse Linear Systems*, SIAM, Philadelphia, Second Edition, 2003.
  (very good book on iterative methods for solving linear systems of equations, in particular PCG methods)
- W. L. Briggs, V. E. Henson, S. F. McCormick: *A Multigrid Tutorial*. Second Edition, SIAM, Philadelphia, 2000.
  (well readable introduction to multigrid methods)