

NUMERISCHE MATHEMATIK I | NAHTLOSES KLONEN

JONAS BRESCH

1. DISKRETE ABLEITUNGEN

Sei $g : [0, N-1] \times [0, M-1] \rightarrow [0, L-1]$ ein kontinuierliches Grauwertbild. In diesem Abschnitt nehmen wir an, dass g differenzierbar ist. Dann enthält g eine Kante oder Textur, wenn sich der Wert von g schnell ändert. Mit anderen Worten sind Informationen über Kanten oder Texturen im Gradienten von g enthalten. Deshalb möchten wir eine Entsprechung des Gradienten für diskrete Grauwertbilder finden.

Vorwärtsdifferenzen. Sei $v : [0, N-1] \rightarrow \mathbb{R}$ eine differenzierbare Funktion und $\mathbf{u} = (u_i)_{i=0}^{N-1}$ gegeben durch $u_i = v(i)$ das diskrete Gegenstück dazu. Dann gilt für $i = 0, \dots, N-2$, dass

$$v'(i) = \lim_{h \rightarrow 0} \frac{v(i+h) - v(i)}{h}.$$

Unter der Annahme, dass N groß ist und sich die Ableitung v' nicht stark verändert, approximieren wir den Grenzwert $h \rightarrow 0$ durch $h = 1$. Dann erhalten wir

$$v'(i) \approx v(i+1) - v(i) = u_{i+1} - u_i.$$

Das motiviert die folgende Definition.

Definition 1.1. Sei $\mathbf{u} = (u_i)_{i=0}^{N-1} \in \mathbb{R}^N$ ein Vektor, wobei wir formal $u_i = 0$ für $i \in \mathbb{Z} \setminus \{0, \dots, N-1\}$ schreiben.

- (i) Wir definieren den Vektor der *Vorwärtsdifferenzen* oder die *diskretisierte Ableitung* von \mathbf{u} als

$$\mathbf{u}' = (u'_i)_{i=0}^{N-1}, \quad u'_i = u_{i+1} - u_i.$$

- (ii) Analog definieren wir den Vektor der *Rückwärtsdifferenzen* von \mathbf{u} als

$$\mathbf{u}^{\leftarrow} = (u_{\leftarrow, i})_{i=0}^{N-1}, \quad u_i^{\leftarrow} = u_i - u_{i-1}.$$

- (iii) Wir nennen

$$\mathbf{u}'' = (u''_i)_{i=0}^{N-1}, \quad u''_i = (\mathbf{u}')'_{i-1} = u_{i+1} - 2u_i + u_{i-1}$$

die *diskretisierte zweite Ableitung* von u .

- (iv) Sei $\mathbf{f} = (f_{i,j})_{i=0, j=0}^{N-1, M-1} \in \mathbb{R}^{N \times M}$ eine Matrix. Dann nennen wir $\nabla \mathbf{f}$ gegeben durch

$$\nabla \mathbf{f}_{i,j} = \begin{pmatrix} f_{i+1,j} - f_{i,j} \\ f_{i,j+1} - f_{i,j} \end{pmatrix}$$

den *diskreten Gradienten* von f . Wir notieren $([\nabla \mathbf{f}]_0)_{i,j} = f_{i+1,j} - f_{i,j}$ und $([\nabla \mathbf{f}]_1)_{i,j} = f_{i,j+1} - f_{i,j}$.

Die Funktion, die einen Vektor auf seine diskrete Ableitungen abbildet, ist linear und kann dementsprechend als Matrixmultiplikation geschrieben werden. Genauer gilt

$$\mathbf{u}' = D_N^v \mathbf{u}, \quad \mathbf{u}^{\leftarrow} = D_N^r \mathbf{u}, \quad \mathbf{u}'' = D_N^{(2)} \mathbf{u}$$

mit

$$D_N^v = \begin{pmatrix} -1 & 1 & & & \\ & -1 & 1 & & \\ & & \ddots & \ddots & \\ & & & -1 & 1 \\ & & & & -1 \end{pmatrix}, \quad D_N^r = \begin{pmatrix} 1 & & & & \\ -1 & 1 & & & \\ & \ddots & \ddots & & \\ & & -1 & 1 & \\ & & & -1 & 1 \end{pmatrix},$$

und

$$D_N^{(2)} = \begin{pmatrix} -2 & 1 & & & \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ & & & 1 & -2 \end{pmatrix}.$$

Dementsprechend ist für $\mathbf{f} = (f_{i,j})_{i=0,j=0}^{N-1,M-1} \in \mathbb{R}^{N \times M}$, dass

$$[\nabla \mathbf{f}]_0 = D_N^v \mathbf{f}, \quad [\nabla \mathbf{f}]_1 = \mathbf{f} (D_M^v)^T.$$

Der Laplace-Operator und vektorisierte Bilder. Für eine offene Menge $\Omega \subset \mathbb{R}^2$ und eine zweimal differenzierbare Funktion $g : \Omega \rightarrow \mathbb{R}$ ist der Laplace-Operator definiert als

$$\Delta g = \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2}.$$

Indem wir die beiden auftretenden zweiten Ableitungen durch die Matrix-Vektor-Multiplikation diskretisieren, definieren wir für eine Matrix $\mathbf{f} = (f_{i,j}) \in \mathbb{R}^{N \times M}$ den *diskreten Laplace-Operator* als

$$\Delta \mathbf{f} = D_N^{(2)} \mathbf{f} + \mathbf{f} D_M^{(2)} \in \mathbb{R}^{N \times M},$$

d. h.

$$\Delta \mathbf{f}_{i,j} = -4f_{i,j} + f_{i+1,j} + f_{i-1,j} + f_{i,j+1} + f_{i,j-1}.$$

Wir möchten den Operator Δ ebenfalls als Matrix-Vektor-Multiplikation ausdrücken. Dafür benötigen wir folgende Definition.

Definition 1.2. Für zwei Matrizen $A \in \mathbb{R}^{N \times N}$ und $B \in \mathbb{R}^{M \times M}$ mit Einträgen

$$A = (a_{p,q})_{p,q=0}^{N-1}, \quad B = (b_{r,s})_{r,s=0}^{M-1},$$

definieren wir das Kronecker-Produkt als Blockmatrix

$$A \otimes B := \begin{pmatrix} a_{0,0}B & \cdots & a_{0,N-1}B \\ \vdots & \ddots & \vdots \\ a_{N-1,0}B & \cdots & a_{N-1,N-1}B \end{pmatrix} \in \mathbb{R}^{NM \times NM}.$$

Des Weiteren definieren wir die Vektorisierung einer Matrix $A = (a_{i,j})_{i=0,j=0}^{N-1,M-1} \in \mathbb{R}^{N \times M}$ als den Vektor $\text{vec}(A) = (\text{vec}(A)_\ell)_{\ell=0}^{MN-1} \in \mathbb{R}^{MN}$ als die Aneinanderreihung der Spalten von A , d. h. $\text{vec}(A)_{i+Nj} = a_{i,j}$.

Mit diesen Definitionen erhalten wir das folgende Lemma.

Lemma 1.3. (i) Seien $A \in \mathbb{R}^{N \times N}$, $\mathbf{f} \in \mathbb{R}^{N \times M}$ und $B \in \mathbb{R}^{M \times M}$. Dann gilt

$$\text{vec}(A\mathbf{f}B) = (B^T \otimes A) \text{vec}(\mathbf{f}).$$

(ii) Für $\mathbf{f} \in \mathbb{R}^{N \times M}$ gilt

$$\text{vec}(\Delta \mathbf{f}) = (I_M \otimes D_N^{(2)} + D_M^{(2)} \otimes I_N) \text{vec}(\mathbf{f}).$$



ABBILDUNG 1. Original (links) und kantenloses Einfügen (rechts).

Beweis. (i) Sei b_i die i -te Spalte von B und \mathbf{f}_i die i -te Spalte von \mathbf{f} . Dann

$$\begin{aligned} \text{vec}(A\mathbf{f}B) &= \begin{pmatrix} A\mathbf{f}b_0 \\ \vdots \\ A\mathbf{f}b_{M-1} \end{pmatrix} = \begin{pmatrix} A(b_{0,0}\mathbf{f}_0 + \cdots + b_{M-1,0}\mathbf{f}_{M-1}) \\ \vdots \\ A(b_{0,M-1}\mathbf{f}_0 + \cdots + b_{M-1,M-1}\mathbf{f}_{M-1}) \end{pmatrix} \\ &= \begin{pmatrix} Ab_{0,0} & \cdots & Ab_{M-1,0} \\ \vdots & & \vdots \\ Ab_{0,M-1} & \cdots & Ab_{M-1,M-1} \end{pmatrix} \begin{pmatrix} \mathbf{f}_0 \\ \vdots \\ \mathbf{f}_{M-1} \end{pmatrix} = (B^T \otimes A) \text{vec}(\mathbf{f}). \end{aligned}$$

(ii) Folgt direkt aus (i) zusammen mit der Definition $\text{vec}(\Delta\mathbf{f}) = \text{vec}(D_N^{(2)}\mathbf{f}) + \text{vec}(\mathbf{f}D_M^{(2)})$, Denn es gilt

$$\text{vec}(\Delta\mathbf{f}) = \text{vec}(D_N^{(2)}\mathbf{f}) + \text{vec}(\mathbf{f}D_M^{(2)}) = \text{vec}(D_N^{(2)}\mathbf{f}I_M) + \text{vec}(I_N\mathbf{f}D_M^{(2)}) = (I_M \otimes D_N^{(2)} + D_M^{(2)} \otimes I_N) \text{vec}(\mathbf{f}).$$

□

Bemerkung 1.4 (Umsetzung in Python). Der vektorisierte Laplace-Operator ist sehr dünn besetzt. Das heißt, fast alle Einträge von $(I_M \otimes D_N^{(2)} + D_M^{(2)} \otimes I_N)$ sind 0. Solche Matrizen können in Python als *sparse* Matrizen dargestellt werden. Um solche sparsen Matrizen zu erzeugen, können in Python die Funktionen

- `scipy.sparse.eye` (sparse Diagonalmatrix)
- `scipy.sparse.kron` (Kronecker-Produkt für sparse Matrizen)

verwendet werden.

2. DIE POISSON-GLEICHUNG FÜR SEAMLESS CLONING

Problemstellung. Im Folgenden möchten wir Ausschnitte aus einem Bild in ein anderes einfügen, ohne unnatürliche Kanten zu erzeugen. Dieser Prozess wird „seamless cloning“, also „nahtloses Klonen“ genannt. Figure 1 gibt ein Beispiel dazu. Dafür betrachten wir zunächst Grauwertbilder und verwenden folgende Notation. Sei $f^* : S \rightarrow \mathbb{R}$ das Hintergrundbild mit Definitionsbereich $S \subset \mathbb{R}^2$. Außerdem sei $\Omega \subset S$ eine abgeschlossene Teilmenge von S und $g : \Omega \rightarrow \mathbb{R}$ der einzufügende Bildausschnitt. Dabei schreiben wir $\partial\Omega$ für den Rand und $\overset{\circ}{\Omega}$ für das Innere von Ω . Unser Ziel ist es nun g nahtlos in f^* einzufügen.

Wir möchten ein Bild $h^* : S \rightarrow \mathbb{R}$ berechnen, sodass

- der Hintergrund erhalten bleibt, das heißt

$$h^*|_{S \setminus \Omega} = f^*,$$

- der ersetzte Bildausschnitt $h^*|_{\Omega}$ auf dem Rand von Ω mit f^* übereinstimmt, das heißt

$$h^*|_{\partial\Omega} = f^*|_{\partial\Omega},$$

- die Gradienten von h^* und g auf Ω möglichst ähnlich sind, d. h.

$$\frac{1}{2} \int_{\Omega} \|\nabla h^* - \nabla g\|^2 d(x, y)$$

ist möglichst klein.

Da h^* und f^* außerhalb von Ω gleich sein sollen, genügt es also folgendes Optimierungsproblem zu lösen:

$$(2.1) \quad h^*|_{\Omega} \in \operatorname{argmin}_{h: \Omega \rightarrow \mathbb{R}} \frac{1}{2} \int_{\Omega} \|\nabla h - \nabla g\|^2 \quad \text{s.t.} \quad h|_{\partial\Omega} = f^*|_{\partial\Omega}.$$

Diskretisierung. Um dieses Optimierungsproblem numerisch zu lösen, müssen wir es diskretisieren und insbesondere von kontinuierlichen zu diskreten Bildern übergehen. Der Einfachheit halber nehmen wir an, dass Ω rechteckig ist und diskretisieren es gleichmäßig in beide Richtungen. Dann erhalten wir das rechteckige Gitter mit Randpunkten $\partial\Omega$ und inneren Punkten $\mathring{\Omega}$.

$$\Omega = \begin{pmatrix} \times & \times & \times & \cdots & \times & \times & \times \\ \times & \bullet & \bullet & \cdots & \bullet & \bullet & \times \\ \times & \bullet & \bullet & \cdots & \bullet & \bullet & \times \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots \\ \times & \bullet & \bullet & \cdots & \bullet & \bullet & \times \\ \times & \bullet & \bullet & \cdots & \bullet & \bullet & \times \\ \times & \times & \times & \cdots & \times & \times & \times \end{pmatrix},$$

Hier stellen \times die Elemente in $\partial\Omega$ markiert und \bullet die Elemente in $\mathring{\Omega}$. Dementsprechend können wir g und h als $N \times M$ -Matrizen $\mathbf{g} = (g_{i,j})$ und $\mathbf{h} = (h_{i,j})$ darstellen. Wir erhalten also die Aufteilung

$$\Omega = \{(i, j) : i \in \{0, \dots, N-1\}, j \in \{0, \dots, M-1\}\},$$

$$\mathring{\Omega} = \{(i, j) : i \in \{1, \dots, N-2\}, j \in \{1, \dots, M-2\}\}$$

$$\partial\Omega = \{(i, j) : i \in \{0, N-1\}, j \in \{0, \dots, M-1\}\} \cup \{(i, j) : i \in \{0, \dots, N-1\}, j \in \{0, M-1\}\}.$$

Indem wir die Ableitungen in (2.1) diskretisieren, erhalten wir das Optimierungsproblem

$$(2.2) \quad h^*|_{\Omega} \in \operatorname{argmin}_{h \in \mathbb{R}^{N \times M}} F(h) := \frac{1}{2} \sum_{i=0}^{N-1} \sum_{j=0}^{M-1} \|\nabla h_{i,j} - \nabla g_{i,j}\|^2 \quad \text{s.t.} \quad h|_{\partial\Omega} = f^*|_{\partial\Omega}.$$

Um dieses Problem zu lösen, suchen wir Nullstellen der Ableitung von F bezüglich $h|_{\mathring{\Omega}}$. Dabei erhält man

$$\frac{\partial F}{\partial h_{i,j}} = 4h_{i,j} - h_{i+1,j} - h_{i-1,j} - h_{i,j+1} - h_{i,j-1} - (4g_{i,j} - g_{i+1,j} - g_{i-1,j} - g_{i,j+1} - g_{i,j-1}) = -\Delta \mathbf{h}_{i,j} + \Delta \mathbf{g}_{i,j}.$$

Wenn wir die Ableitung auf 0 setzen, erhalten wir das Gleichungssystem

$$\Delta \mathbf{h} = \Delta \mathbf{g} \quad \text{auf } \mathring{\Omega} \quad \text{mit} \quad \mathbf{h}|_{\partial\Omega} = \mathbf{f}^*|_{\partial\Omega}.$$

Dies ist eine diskrete Form der Poisson-Gleichung. Durch Lemma 3 (ii) können wir sie wie folgt vektorisieren:

$$(I_M \otimes D_N^{(2)} + D_M^{(2)} \otimes I_N) \operatorname{vec}(\mathbf{h}) = (I_M \otimes D_N^{(2)} + D_M^{(2)} \otimes I_N) \operatorname{vec}(\mathbf{g}) \quad \text{auf } \mathring{\Omega}$$



ABBILDUNG 2. Ausgangsbilder (links), keine Struktur von \mathbf{f}^* auf Ω (mitte), und Struktur aus Hintergrundbild \mathbf{f}^* und \mathbf{g} auf Ω übertragen (rechts).

mit Randbedingungen $\mathbf{h}|_{\partial\Omega} = \mathbf{f}^*|_{\partial\Omega}$. Dieses Gleichungssystem kann beispielsweise mit dem CG-Verfahren (Conjugate Gradient) für sparse Matrizen gelöst werden.

Bemerkung 2.1. Für das CG-Verfahren für sparse Matrizen kann in Python die Funktion `scipy.sparse.linalg.cg` verwendet werden. Achtet darauf, dass das Abbruchkriterium so gewählt ist, dass genügend Iterationen durchlaufen werden.

Allgemeinere Vektorfelder in der Poisson-Gleichung. Bisher werden mit der oben beschriebenen Methode alle Strukturen des Hintergrundbildes \mathbf{f}^* auf Ω gelöscht. Figure 2 gibt ein Beispiel dazu. In manchen Fällen kann das zu unerwünschten Ergebnissen führen. Dafür ersetzen wir den Gradienten von g in (2.1) durch ein allgemeineres Vektorfeld \mathbf{v} :

$$h^*|_{\Omega} \in \operatorname{argmin}_{h:\Omega \rightarrow \mathbb{R}} \frac{1}{2} \int_{\Omega} \|\nabla h - \mathbf{v}\|^2 \quad \text{s.t.} \quad h|_{\partial\Omega} = f^*|_{\partial\Omega}.$$

Erneutes Diskretisieren liefert mit $\mathbf{v} = (v_{i,j,k})$ analog zu (2.2) das Problem

$$h^*|_{\Omega} \in \operatorname{argmin}_{\mathbf{h} \in \mathbb{R}^{N \times M}} F_{\mathbf{v}}(\mathbf{h}) := \frac{1}{2} \sum_{i=0}^{N-1} \sum_{j=0}^{M-1} \|\nabla \mathbf{h}_{i,j} - \mathbf{v}_{i,j}\|^2 \quad \text{s.t.} \quad \mathbf{h}|_{\partial\Omega} = \mathbf{f}^*|_{\partial\Omega}.$$

Die Zielfunktion $F_{\mathbf{v}}$ hat hierbei die Ableitung

$$\frac{\partial F_{\mathbf{v}}}{\partial h_{i,j}} = 4h_{i,j} - h_{i+1,j} - h_{i-1,j} - h_{i,j+1} - h_{i,j-1} + v_{i,j,0} - v_{i-1,j,0} + v_{i,j,1} - v_{i,j-1,1} = -\Delta \mathbf{h}_{i,j} + \operatorname{div} \mathbf{v}_{i,j},$$

wobei $\operatorname{div} \mathbf{v} = (\operatorname{div} v_{i,j})$ die *diskretisierte Divergenz* von \mathbf{v} ist. Damit erhalten wir die allgemeinere Form der diskreten Poisson-Gleichung:

$$\Delta \mathbf{h} = \operatorname{div} \mathbf{v} \quad \text{auf} \quad \overset{\circ}{\Omega} \quad \text{mit} \quad \mathbf{h}|_{\partial\Omega} = \mathbf{f}^*|_{\partial\Omega}.$$

Um diese Gleichung wieder in Matrix-Vektor-Form zu überführen, bemerken wir, dass

$$\operatorname{div} \mathbf{v} = D_N^r[\mathbf{v}]_0 + [\mathbf{v}]_1(D_M^r)^T, \quad [\mathbf{v}]_k = (v_{i,j,k})_{i,j},$$

wobei D_N^r die Matrix der Rückwärtsdifferenzen ist.

Abschließend müssen wir noch das Vektorfeld \mathbf{v} passend wählen, um Objekte aus beiden Bildern zu überlagern. Dafür übernehmen wir an jeder Stelle (i,j) immer den Gradienten mit größerer Norm,

das heißt wir wählen

$$\mathbf{v}_{i,j} = \begin{cases} \nabla \mathbf{f}_{i,j}^*, & \text{falls } \|\nabla \mathbf{f}_{i,j}^*\| > \|\nabla \mathbf{g}_{i,j}\|, \\ \nabla \mathbf{g}_{i,j}, & \text{sonst.} \end{cases}$$

Bemerkung 2.2 (Farbbilder). Für Seamless Cloning für Farbbilder löst man das Problem für jeden Farbkanal separat.