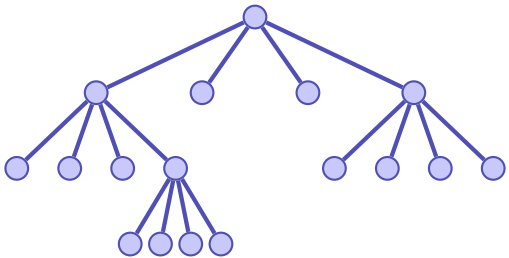
18. Quadtrees

Quadtrees, Bildsegmentierung, Funktionalminimierung, Reduktionsprinzip

Quadtree

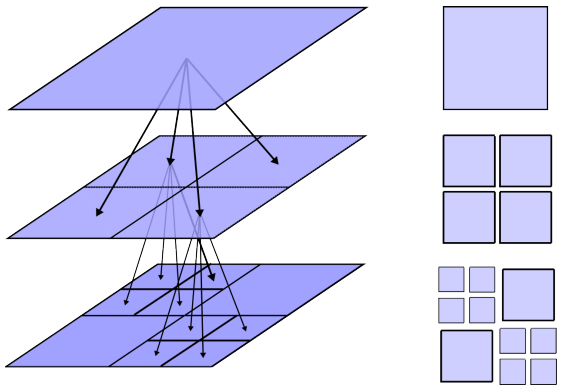
Ein Quadtree ist ein Baum der Ordnung 4.



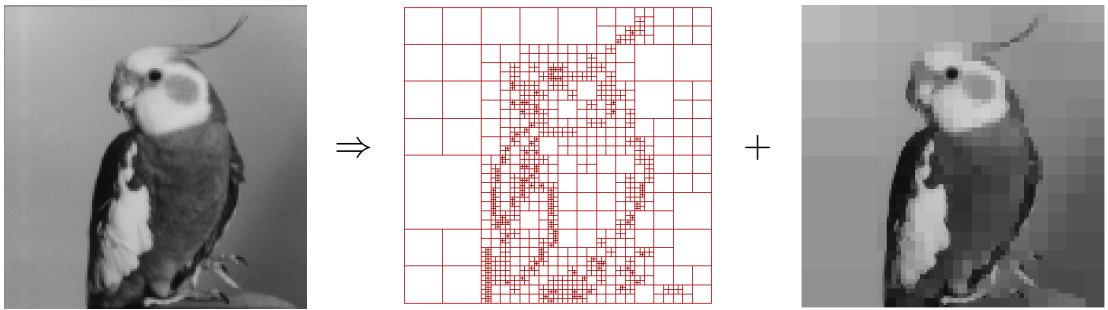
... und ist als solcher nicht besonders interessant, ausser man verwendet ihn zur...

Quadtree - Interpretation und Nutzen

Partitionierung eines zweidimensionalen Bereiches in 4 gleich grosse Teile.

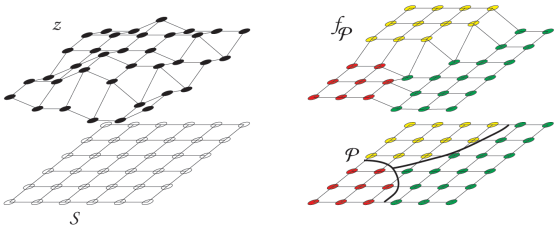


Bildsegmentierung



(Mögliche Anwendungen: Kompression, Entrauschen, Kantendetektion)

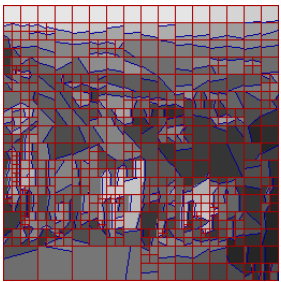
Etwas Notation



$S \subset \mathbb{Z}^2$	endliche rechteckige Indexmenge (‘Pixel’)
$z \in \mathbb{R}^S$	Bild
$\mathfrak{P}$	Familie von Partitionen $\mathcal{P} \subset 2^S$ von S
$\mathcal{F} = (\mathcal{F}_r)_{r \subset S}$	Familie von ‘Regressionsmodellen’ $\mathcal{F}_r \subset \mathbb{R}^r$
$f_{\mathcal{P}} \in \mathbb{R}^S$	‘Approximation’ mit $f_{\mathcal{P}}\|_r \in \mathcal{F}_r$, $r \in \mathcal{P}$
$\mathfrak{S}$	Familie von Segmentierungen $(\mathcal{P}, f_{\mathcal{P}})$

Anderes Beispiel

z



$(\mathcal{P}, f_{\mathcal{P}})$

$f_{\mathcal{P}}$

$\mathcal{P}$: Quadtrees mit zusätzlicher Unterteilung in Polygone (‘Wedges’),
$f_{\mathcal{P}}$: konstante Funktionen

Minimierungsproblem

$\mathcal{P}$ Partition $\quad$ $\gamma \geq 0$ Regularisierungsparameter
$f_{\mathcal{P}}$ Approximation $\quad$ z Bild = ‘Daten’

Ziel: Effiziente Minimierung des Funktionals

$$H_{\gamma,z} : \mathfrak{S} \to \mathbb{R}, \quad (\mathcal{P}, f_{\mathcal{P}}) \mapsto \gamma \cdot |\mathcal{P}| + \|z - f_{\mathcal{P}}\|_2^2.$$

Ergebnis $(\hat{\mathcal{P}}, \hat{f}_{\hat{\mathcal{P}}}) \in \operatorname{argmin}_{(\mathcal{P},f_{\mathcal{P}})} H_{\gamma,z}$ interpretierbar als *optimaler Kompromiss zwischen Regularität und Datentreue*.

Warum Quadtrees

$$H_{\gamma,z} : \mathfrak{S} \to \mathbb{R}, \quad (\mathcal{P}, f_{\mathcal{P}}) \mapsto \gamma \cdot |\mathcal{P}| + \|z - f_{\mathcal{P}}\|_2^2.$$

- Anzahl aller Partitionierungen extrem gross ($|\mathfrak{P}| > 2^{|S|}$)
- Einsatz von Markov-Chain-Monte-Carlo (MCMC) Methoden zur Minimierung des Funktionals H über alle Partitionierung möglich, aber zeitaufwändig und inexakt.
- $\Rightarrow$ Einschränkung des Suchraumes. Hierarchische Quadtree-Partitionierung besonders gut mit rekursivem Divide-And-Conquer Ansatz verträglich.[23]

[23] Wie Quicksort (nur 2d)!

Reduktionsprinzip

$$\min_{(\mathcal{P},f_{\mathcal{P}})\in\mathfrak{S}} \gamma|\mathcal{P}| + \|z - f_{\mathcal{P}}\|_2^2$$

$$= \min_{\mathcal{P}\in\mathfrak{P}} \left\{ \gamma|\mathcal{P}| + \sum_{r\in\mathcal{P}} \min_{f_r\in\mathcal{F}_r} \sum_{s\in r} \left(z(s) - f_r(s)\right)^2 \right\}$$

$\Rightarrow$ Separation von Partitionssuche und lokaler Projektion.

Algorithmus: Minimize(z,r,γ)

Input : Bilddaten $z \in \mathbb{R}^S$, Rechteck $r \subset S$, Regularisierung $\gamma > 0$
Output : $\min_{(\mathcal{P},f_\mathcal{P})\in\mathfrak{S}} \gamma|\mathcal{P}| + \|z - f_\mathcal{P}\|_2^2$

if $|r| = 0$ **then return** 0
$m \leftarrow \gamma + \min_{f_r \in \mathcal{F}_r} \sum_{s\in r} \left(z(s) - f_r(s)\right)^2$
if $|r| > 1$ **then**
    Split r into r_{ll},r_{lr},r_{ul},r_{ur}
    $m_1 \leftarrow \text{Minimize}(z, r_{ll})$
    $m_2 \leftarrow \text{Minimize}(z, r_{lr})$
    $m_3 \leftarrow \text{Minimize}(z, r_{ul})$
    $m_4 \leftarrow \text{Minimize}(z, r_{ur})$
    $m' \leftarrow m_1 + m_2 + m_3 + m_4$
else
    $m' \leftarrow \infty$
if $m' < m$ **then** $m \leftarrow m'$
return m

Konstante Funktionen

Minimiere

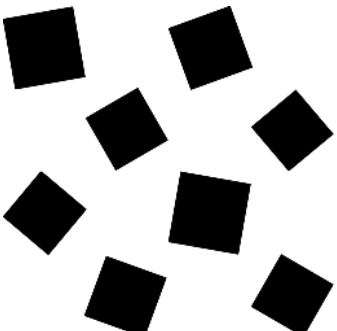
$$\min_{f_r \in \mathcal{F}_r} \sum_{s \in r} \left(z(s) - f_r(s)\right)^2$$

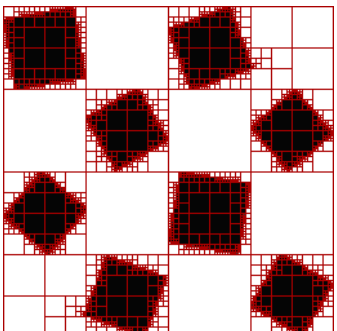
für die auf r konstanten Funktionen

Lösung: $\mu_r = \frac{1}{r} \sum_{s \in r} z(s)$

Schnelle Berechnung von μ_r auf Rechtecken möglich: Präfixsummen!

Multiskalenansatz!





Allgemeine Regression

Betrachte Familie von $n \in \mathbb{N}$ Funktionen $\varphi_i : S \to \mathbb{R}$, $1 \leq i \leq n$.

Ziel: Minimiere

$$\sum_{s \in r} (z_s - \sum_{i=1}^{n} a_i \varphi_i(s))^2$$

in $a \in \mathbb{R}^n$.

Normalengleichungen:

$$\sum_{s \in r} z_s \varphi_j(s) = \sum_{s \in r} \sum_{i=1}^{n} a_i \varphi_i(s) \varphi_j(s), 1 \leq j \leq n$$

$$\Leftrightarrow \sum_{s \in r} z_s \varphi_j(s) = \sum_{i=1}^{n} a_i \sum_{s \in r} \varphi_i(s) \varphi_j(s), 1 \leq j \leq n$$

Allgemeine Regression

Normalengleichungen in Matrixschreibweise:

$$Y = M \cdot a.$$

mit $a = (a_i)_{1 \le i \le n}$ und

$$Y := \left(\sum_{s \in r} z_s \varphi_j(s) \right)_{1 \le j \le n}, \quad M := \left(\sum_{s \in r} \varphi_i(s) \varphi_j(s) \right)_{1 \le i,j \le n}.$$

Allgemeine Regression

Sei $\hat{a}$ eine Lösung obigen Gleichungssystems. Berechnung des Approximationsfehlers:

$$\min_{f_r \in \mathcal{F}_r} \sum_{s \in r} (z_s - f_r(s))^2 = \sum_{s \in r} (z_s - \sum_{i=1}^{n} \hat{a}_i \varphi_i(s))^2$$

$$= \sum_{s \in r} z_s^2 - 2 \sum_{i=1}^{n} \hat{a}_i Y_i + \sum_{i=1}^{n} \hat{a}_i^2 M_{ii}.$$

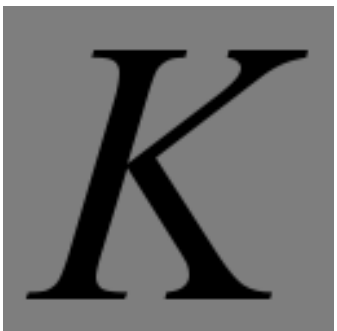
Beispiel: Affine Funktionen

$n = 3$

- $\varphi_0(s) = 1$,
- $\varphi_1(s) = s_1$ (x-Koordinate von s),
- $\varphi_2(2) = s_2$ (y-Koordinate von s)

Regression: Übung!

Affine Regression



Effiziente lokale Berechnung

Benötigt: schnelle Berechnung der $\frac{n(n+1)}{2} + n$ 'Momente'

$$\sum_s \varphi_i(s)\varphi_j(s) \text{ und } \sum_{s \in r} z_s \varphi_j(s), 1 \leq i, j \leq n,$$

und für den Approximationsfehler

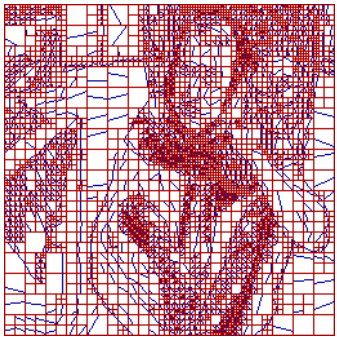
$$\sum_{s \in r} z_s^2.$$

Verwendung von Präfixsummen: Berechnung der lokalen Regression über Rechtecken in $\mathcal{O}(1)$.

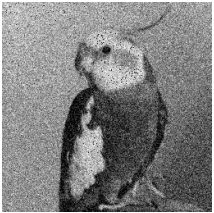
Analyse

Unter der Voraussetzung, dass die lokale Approximation in $\mathcal{O}(1)$ berechnet werden kann, benötigt der Minimierungsalgorithmus über dyadische Partitionen (Quadtree) $\mathcal{O}(|S| \log |S|)$ Schritte.

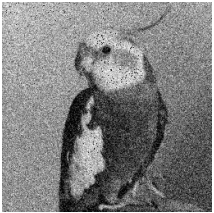
Affine Regression + Wedgelets



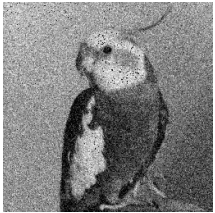
Entrauschen



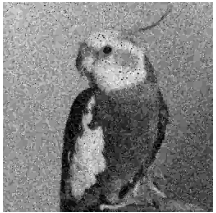
noised



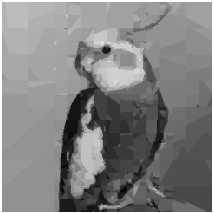
$\gamma = 0.003$



$\gamma = 0.01$

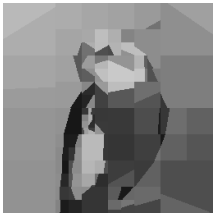


$\gamma = 0.03$



$\gamma = 0.1$

$\gamma = 0.3$



$\gamma = 1$

$\gamma = 3$

$\gamma = 10$

Andere Ideen

kein Quadtree: hierarchisch-eindimensionales Modell (benötigt Dynamic Programming)