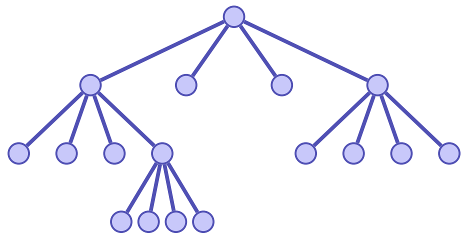
18. Quadtrees

Quadtrees, Image Segmentation, Functional Minimization, Reduction Principle

Quadtree

A quad tree is a tree of order 4.



... and as such it is not particularly interesting except when it is used for ...

Quadtree - Interpretation und Nutzen

Separation of a two-dimensional range into 4 equally seized parts.

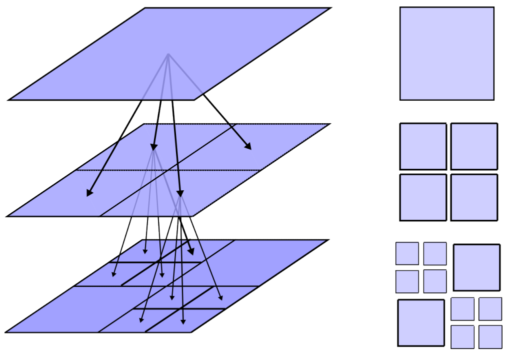
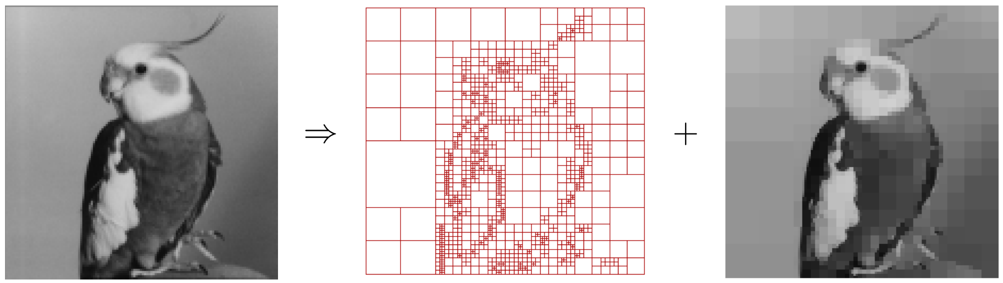
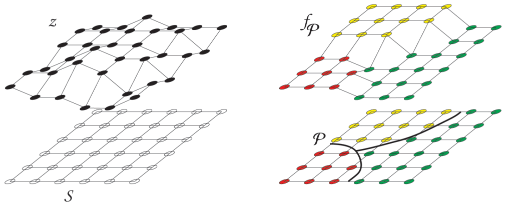


Image Segmentation



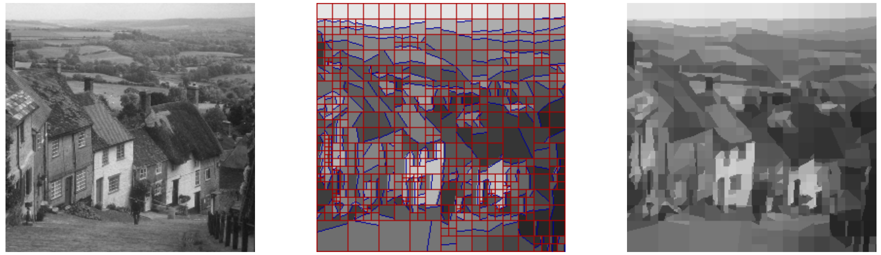
(Possible applications: compression, denoising, edge detection)

A little bit of Notation



$S \subset \mathbb{Z}^2$	finite rectangular index set ('Pixel')
$z \in \mathbb{R}^S$	image
$\mathfrak{P}$	family of Partitions $\mathcal{P} \subset 2^S$ von S
$\mathcal{F} = (\mathcal{F}_r)_{r \subset S}$	family of 'regression models' $\mathcal{F}_r \subset \mathbb{R}^r$
$f_{\mathcal{P}} \in \mathbb{R}^S$	'approximation' with $f_{\mathcal{P}}\|_r \in \mathcal{F}_r$, $r \in \mathcal{P}$
$\mathfrak{S}$	family of segmentations $(\mathcal{P}, f_{\mathcal{P}})$

Different Example



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

$\mathcal{P}$: quad-tree with additional partition into polygons ('wedges'),
$f_{\mathcal{P}}$: constant functions

Minimization Problem

$\mathcal{P}$ Partition $\quad$ $\gamma \geq 0$ regularization parameter
$f_{\mathcal{P}}$ approxmation $\quad$ z image = 'data'

Goal: Efficient mimization of the functional

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

Result $(\hat{\mathcal{P}}, \hat{f}_{\hat{\mathcal{P}}}) \in \operatorname{argmin}_{(\mathcal{P}, f_{\mathcal{P}})} H_{\gamma,z}$ can be interpreted as *optimal compromise between regularity and fidelity to data.*

Why Quadtrees?

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

- Number of all partitions extremely large ($|\mathfrak{P}| > 2^{|S|}$)
- Possible to approximately minimize H using Markov-Chain-Monte-Carlo (MCMC) Methods, very time- and compute-intensive.
- $\Rightarrow$ Restriction of the search space. Hierarchical partitioning using quadtrees particularly well suited for a divide-and-conquer approach.[23]

[23] Like quicksort (only 2d)!

Reduction Principle

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

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

⇒ Separation of searching for the best possible partition and the local projections.

Algorithmus: Minimize(z,r,γ)

Input : Image data $z \in \mathbb{R}^S$, rectangle $r \subset S$, regularization $\gamma > 0$
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 ← γ + min_{f_r ∈ F_r} Σ_{s∈r} (z(s) − f_r(s))^2
if |r| > 1 then
    Split r into r_ll, r_lr, r_ul, r_ur
    m_1 ← Minimize(z, r_ll)
    m_2 ← Minimize(z, r_lr)
    m_3 ← Minimize(z, r_ul)
    m_4 ← Minimize(z, r_ur)
    m' ← m_1 + m_2 + m_3 + m_4
else
    m' ← ∞
if m' < m then m ← m'
return m
```


Constant Functions

Minimize

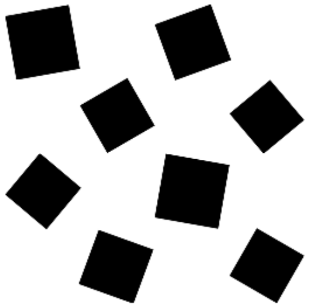
$$\min_{f_r\in\mathcal{F}_r} \sum_{s\in r} (z(s) - f_r(s))^2$$

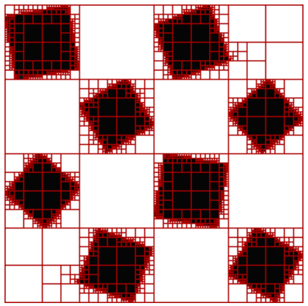
for all functions $\mathcal{F}_r = \mu_r$ being constant on r

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

Fast computation of μ_r is easily possible using prefix sums

Multiple Scales





General Regression

Consider a family of $n \in \mathbb{N}$ functions $\varphi_i : S \to \mathbb{R}$, $1 \le i \le n$.

Goal: minimize

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

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

Normal equations:

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

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

General Regression

Normal equations written in matrix form:

$$Y = M \cdot a.$$

with $a = (a_i)_{1 \le i \le n}$ and

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

General Regression

Let $\hat{a}$ be a solution of the system of equations above. Computation of the approximation error:

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

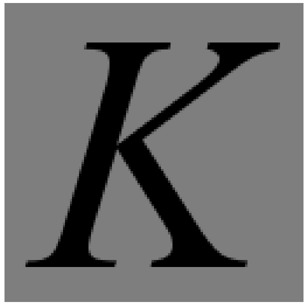
Example: Affine Functions

$n = 3$

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

Regression: exercise!

Affine Regression



Effiziente lokale Berechnung

Required: fast computation of the $\frac{n(n+1)}{2} + n$ ‘moments’

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

and for the computation of the approximation error

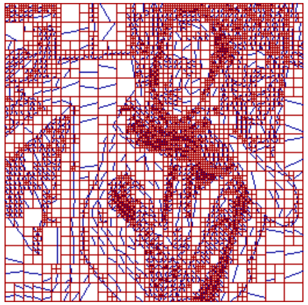
$$\sum_{s \in r} z_s^2.$$

Using prefix sums it is possible to compute the local regression over rectangles in $\mathcal{O}(1)$

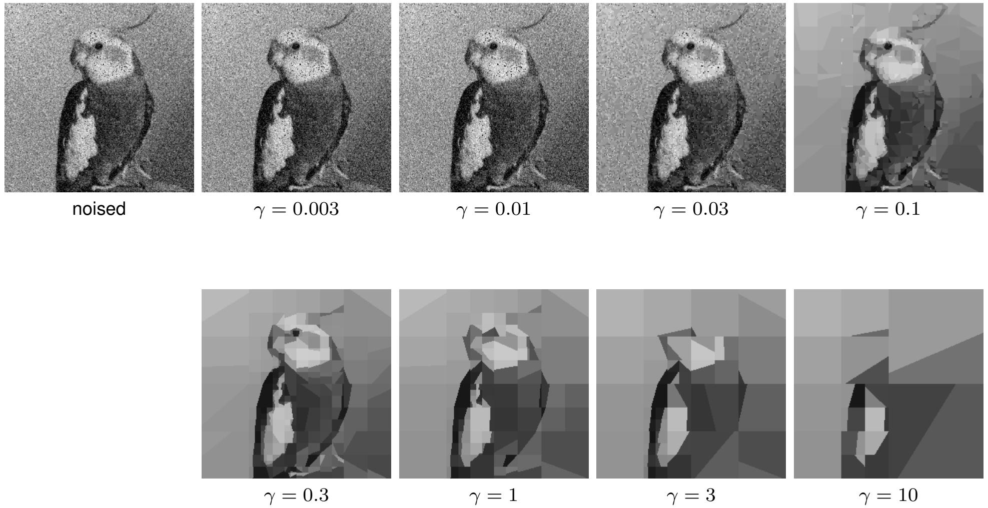
Analysis

Under the assumption that the local approximation can be computed in $\mathcal{O}(1)$ the minimization algorithm over dyadic partitions (quadtrees) takes $\mathcal{O}(|S| \log |S|)$ steps.

Affine Regression + Wedgelets



Denoising



Other ideas

no quadtree: hierarchical one-dimensional modell (requires dynamic programming)