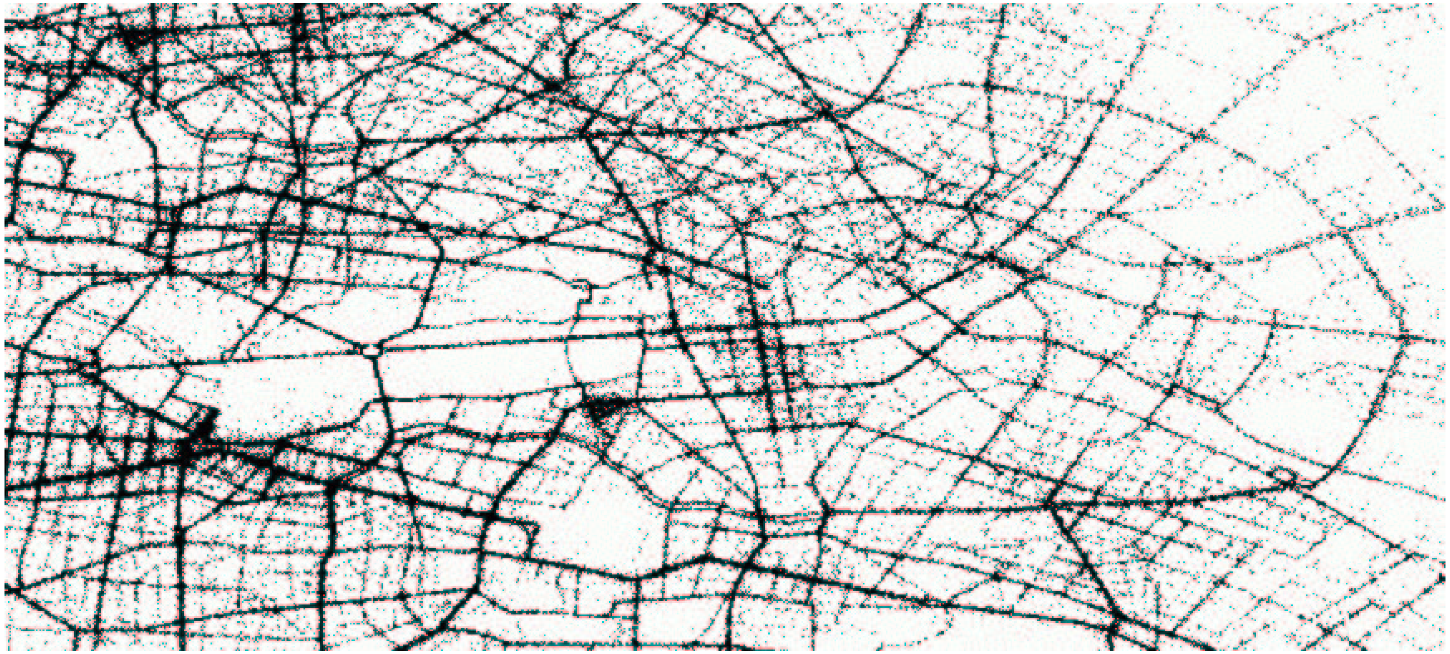
Effiziente Algorithmen für Ordinary Kriging

Universität Ulm
Abteilung Stochastik

H. Braxmeier
November 2002

1 Motivation

Lokalisationsdaten von Taxen aus Berlin Mitte



2 Bezeichnungen und Begriffe

Bezeichnungen

$Z(x)$	Zufallsfeld
$z(x)$	eine Realisierung von Z(x)
$x_1 \ldots x_n$	Meßstellen
D	Beobachtungsfenster
$g_1 \ldots g_{m^2}$	Gitterpunkte, an denen Z geschätzt werden soll

Komplexitätsbegriff

- Es wird versucht, algorithmische Probleme gemäß ihres Bedarfs an Berechnungssresourcen (Rechenzeit, Speicher) als Funktion einer Eingabelänge (in unsrem Fall z.B. n oder m) zu klassifizieren
- Landausche O-Notation: Komplextität wird in Abhängigkeit der Eingabelänge betrachtet, Konstanten fallen unter den Tisch
- Keine Unterscheidung zwischen Anzahl der Additionen und Multiplikationen - beide werden werden gleich bewertet
- Beispiel: Eingabelänge sei abhängig von n und m, der Algorithmus benötige $100 + n + \log n$ Additionen und m^2 Multiplikationen. Komplexität ist dann: $O(n + m^2)$

3 Ordinary Kriging

Schätzung des Wertes im Punkt x_0 mittels Ordinary Kriging

Voraussetzung: Intrinsische Hypothese

- $EZ(x) = m \quad \forall x \in D$, $\quad m$ unbekannt
- Variogrammfunktion:
  $\gamma(x_i - x_j) = \frac{1}{2}E(Z(x_i) - Z(x_j))^2$

Schätzer

$$\hat{Z}(x_0) = \sum_{i=1}^{n} w_i z(x_i)$$

Problem: Gewichte unbekannt

Gleichungssystem zur Berechnung der Gewichte

$$\sum_{j=1}^{n} w_j \gamma(x_i - x_j) + \mu = \gamma(x_i - x_0) \quad \forall i, \quad \sum_{j=1}^{n} w_j = 1$$

Gleichungsystem für Ordinary Kriging in Matrixform

$$\begin{pmatrix} 0 & \dots & \gamma(x_1 - x_n) & 1 \\ \vdots & \ddots & \vdots & \vdots \\ \gamma(x_n - x_1) & \dots & 0 & 1 \\ 1 & \dots & 1 & 0 \end{pmatrix} \begin{pmatrix} w_1 \\ \vdots \\ w_n \\ \mu \end{pmatrix} = \begin{pmatrix} \gamma(x_1 - x_0) \\ \vdots \\ \gamma(x_n - x_0) \\ 1 \end{pmatrix}$$

kurz: $A * w = b$

Komplexität für Ordinary Kriging mit Gaußverfahren

- Gleichungssystem $A * \omega = b$ mit Gaußverfahren: $O(n^3)$
- muß für jeden Gitterpunkt $g_i, i = 1...m^2$ gelöst werden!
- Gesamtkomplexität: $O(n^3 * m^2)$

Alternative: QR-Zerlegung

- Einmalige Berechnung der Matrix A^{-1}: $O(n^3)$
- Berechnung der Gewichte $\omega = A^{-1} * b$: $O(n^2)$
- Gesamtkomplexität: $O(n^2 * m^2 + n^3)$

In der Praxis

- QR-Zerlegung besser, falls m vergleichbar mit n
- Aber: QR-Zerlegung numerisch instabiler (empirischer Befund)
- Beide Verfahren (auf unseren Rechnern) max. für $n = 300$ und $m = 400$ anwendbar, da Matrizen sehr groß werden (Rechenzeit)

$\Rightarrow$ andere Verfahren?

Ordinary Kriging mit gleitender Umgebung

Schätzer

$$\hat{Z}(x_0) = \sum_{i:x_i - x_0 \in U} w_i z(x_i)$$

- Isotroper Fall: häufig radiale Umgebung $U = \{x : |x - x_0| \leq r\}$
- Anisotroper Fall: häufig Sektoren, Rechtecke usw.
- Beliebige Umgebung: Indikatorfunktion

$\Rightarrow$ Gauß-Verfahren hier besser, da A^{-1} bei QR-Zerlegung bei gleitender Umgebung für jeden zu schätzenden Punkt x_0 neu berechnet werden muß!

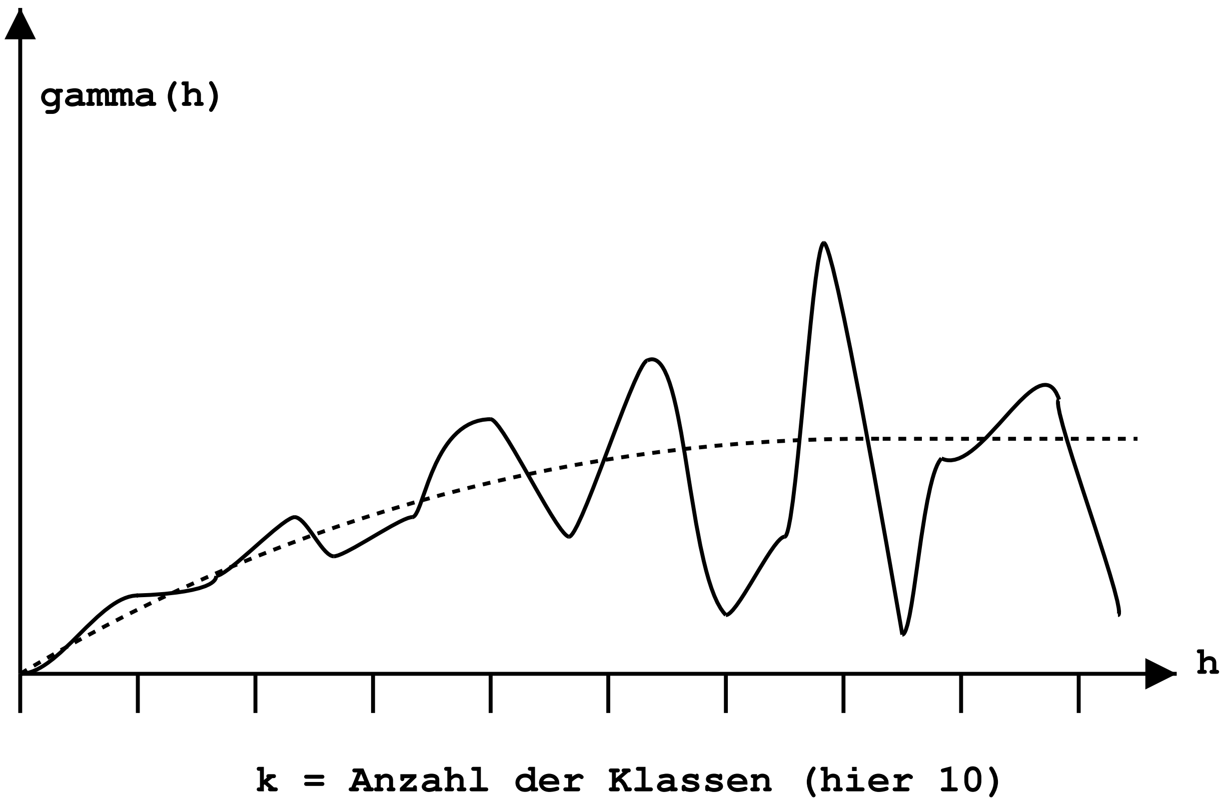
4 Experimentelle Variogramme

erwartungstreuer Schätzer für experimentelles Variogramm:

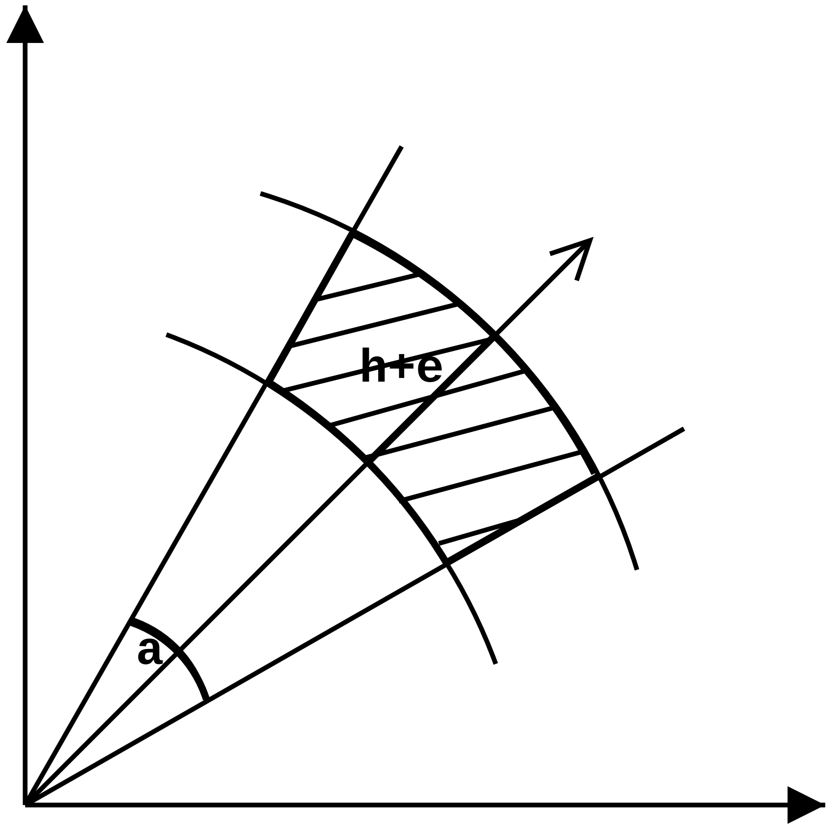
$$\hat{\gamma}(h) = \frac{1}{2N_h} \sum_{|x_i - x_j| = h} (z(x_i) - z(x_j))^2$$

in der Praxis oft Verwendung einer Vektorklasse H statt Abstand h

Beispiel eines experimentellen Variogramms



Beispiel einer Vektorklasse H



Berechnung eines experimentellen Variogramms

Zur Berechnung eines experimentellen Variogramms werden alle Punktepaare $p_i(i = 1...\frac{n*(n-1)}{2})$ aller Vektorklassen $H_j(j = 1...k)$ benötigt

Simpler Algorithmus

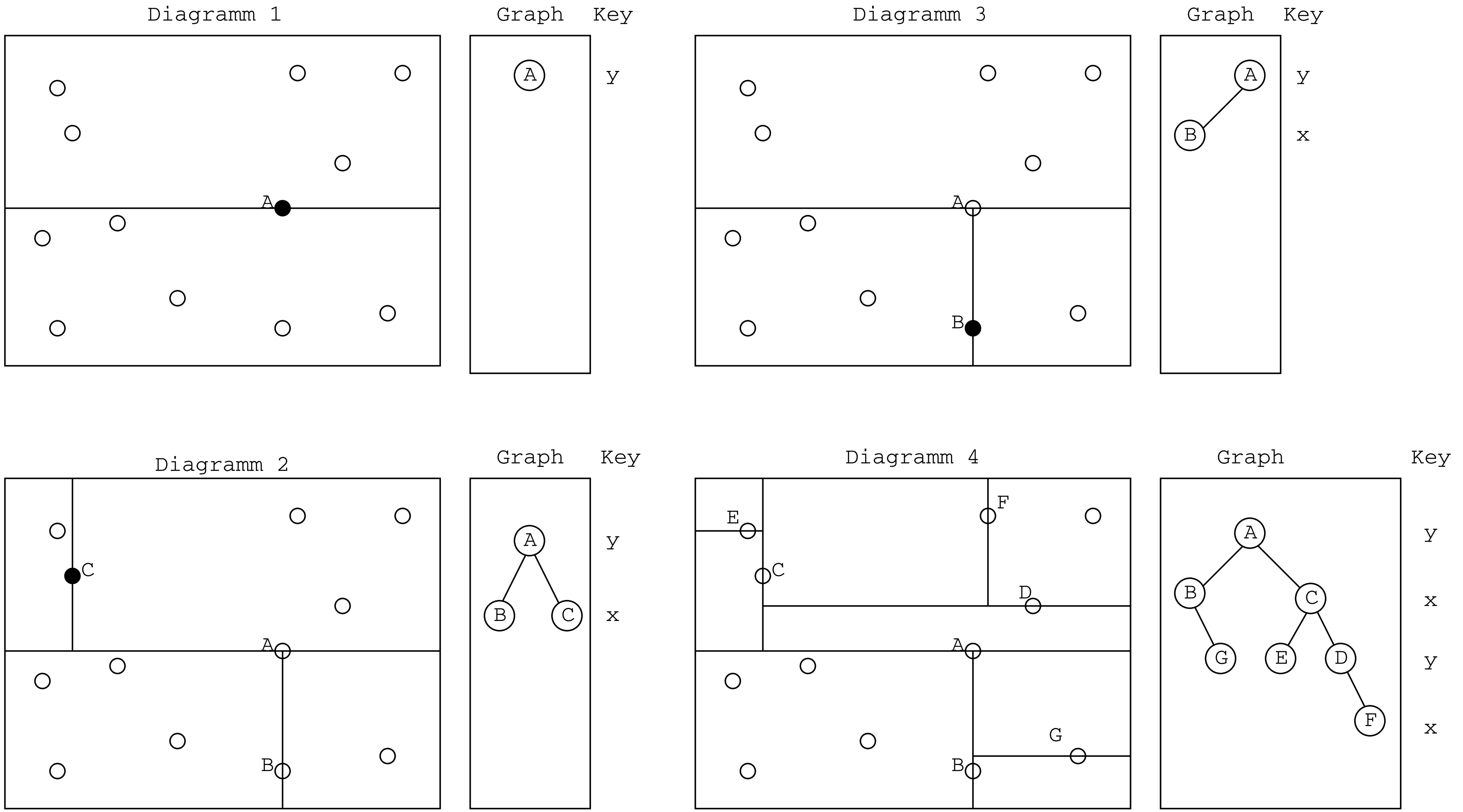
Simpler Algorithmus: Durchsuchen der kompletten Punktmenge nach allen Punktepaaren p_i, die zu einer Vektorklasse H_j gehören, Komplexität: $O(n^2)$, kein zusätzlicher Speicherverbrauch

Für alle Vektorklassen: $k * O(n^2)$

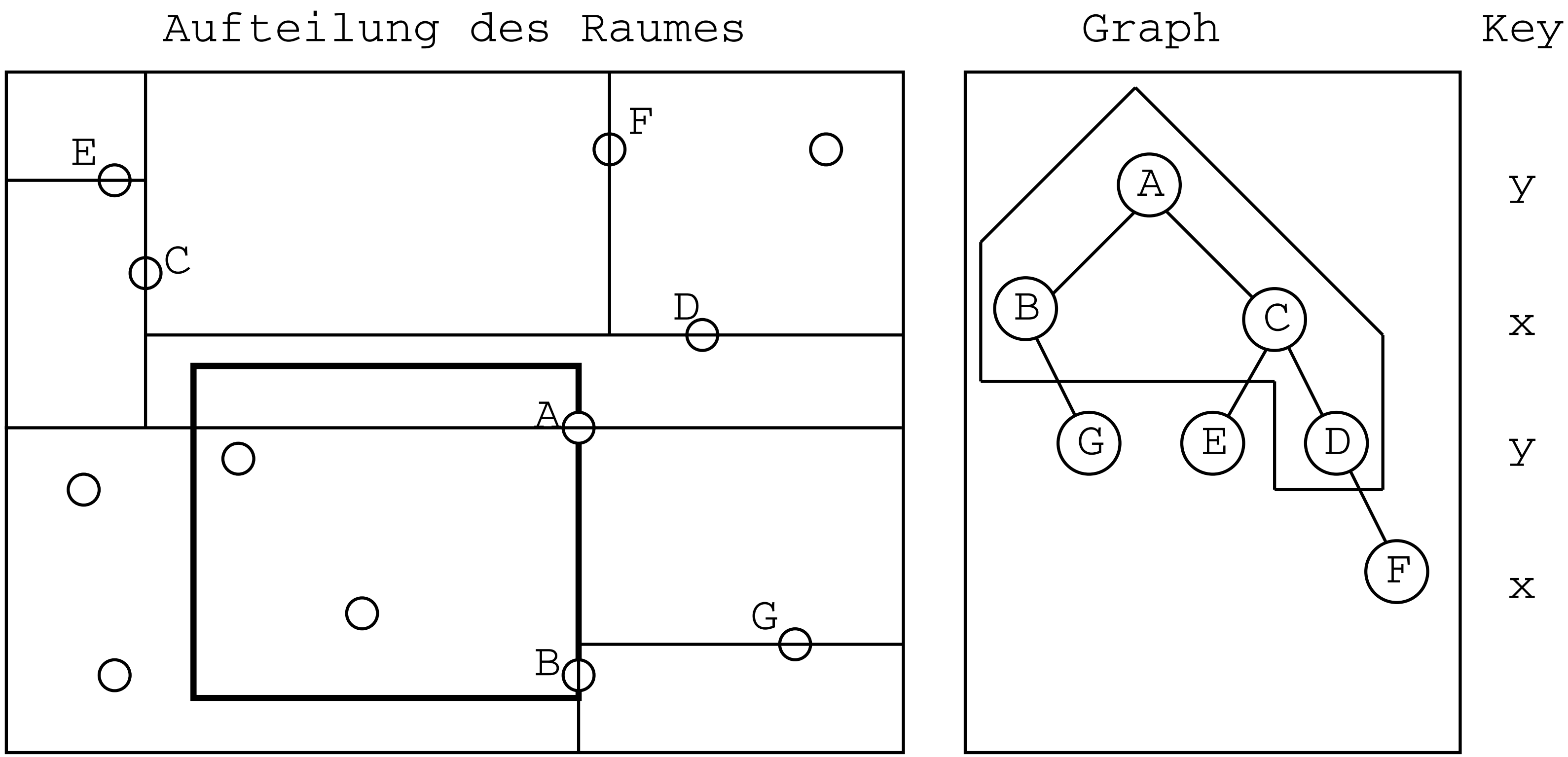
DTree Algorithmus

- Der DTree Algorithmus besteht aus 2 Teilen:
  - Aufbau eines Baumes (DTree)
  - Bereichssuche
- DTrees teilen einen geometrischen Raum in einer Weise auf, die für die Bereichssuche zweckmäßig ist
- Es werden Binärbäume erzeugt, wobei die Punkte eines geometrischen Raumes den Knoten des Baumes entsprechen
- Die Koordinaten der Punkte dienen in alternierender Folge als Schlüssel
- Die Punkte werden zufällig aus dem geometr. Raum ausgewählt

Beispiel mit 2 Koordinaten: Aufbau des Baumes



Beispiel mit 2 Koordinaten: Bereichssuche



DTree Komplexität

- Alle $\frac{n*(n-1)}{2}$ Punktepaare werden in einem DTree abgespeichert (zusätzlicher Speicherverbrauch)
- Komplexität zum Füllen des Baumes: $O(n^2)$
- Komplexität um aus n Punkten R Punkte in einem sinnvollen Gebiet (hier Rechteck) zu finden: $R + \log n$ (für den 2D-Fall empirisch ermittelt), hier: $R \leq n$, somit Komplexität: $O(n)$
- Für alle Vektorklassen: $O(n^2) + k * O(n)$
- Bemerkung: Für die Bereichssuche müssen mehr Knoten überprüft werden, als tatsächlich im ausgewählten Bereich liegen
- In der Praxis: auf unserem Rechner $n \geq 2000$ kritisch (Speicher)

Variogrammberechnung

- Simpler Algorithmus: $k * O(n^2)$
- DTree Algorithmus: $O(n^2) + k * O(n)$

Implementierung

- DTrees für beliebige Dimensionen implementiert

$\Rightarrow$ Verwendung beim Kriging im isotropen Fall:
Einmalige Berechnung eines experimentiellen Variogramms:
$O(n^2) + k * O(n)$

5 Anisotropie

- Es wird geprüft, ob die Daten irgendeine Richtungsabhängigkeit aufweisen
- D.h. für jede Richtung r muß ein Variogramm berechnet werden
- Wird eine geometrische Anisotropie festgestellt, so kann man die Koordinaten des isotropen Modells linear transformieren

6 Ausblick

- Untersuchung des anisotropen Falls
- Anpassung theoretischer Variogramme an die experimentellen Variogramme durch genetische Algorithmen
- ...und noch vieles mehr