# Probleme Bilevel-bundle Algorithmus

* mit airfoil\_self\_noise Testset, Startwert [C,eps] = [1,0]
* quadprog dauert sehr lange
* erreicht immer die maximalen 500 Iterationen
* BB-Algorithmus bricht bereits nach einem Schritt ab
* mit airfoil-set gekürzt auf 100 Datensätze und 3 folds (67 je Fold)
* quadprog löst in wenigen Sekunden
* erreicht nicht die 500 Iterationen

### Probleme:

**1 Problem:** wenn constraints C>0, eps>=0, dann sinnlose Ergebnisse (xi,d zu klein...)

Wenn diese Constraints nicht da, dann C negativ, dadurch Supproblem nicht konvex

--> dann hängt alles in Endlosschleife

auch wenn nicht bound constraints, sondern in inequality constraints drin (sogar schlimmer)

**LÖSUNG:** bound constraints waren falsh einprogrammiert, ist jetzt gelöst

**2 Problem:** Lower level problem braucht > 500 iterationen

**LÖSUNG:** "Constraints" zu eng gewählt -> ungenauere Lösung zulassen, dann wird auch gelöst

macht auch Sinn, da inexact Algorithm genau damit umgehen können sollte -> dafür ausgelegt

(Unterschied zu Bundle Subproblem, welches laut Theorie sehr exakt gelöst werden muss)

**3 Hauptproblem:**

- Stopping Conditions sehr unterschiedlich (Verhalten des Algos)

Unterprobleme:

1. Algorithmus stoppt entweder sehr schnell oder gar nicht

Lösungsansätze:

- Welche Stopping Conditions bringen gleiches Ergebnis?

1, 4, 5: erfolgreicher Stopp nach 11 Iterationen (8 null steps);

x\_hat = (42.2113;21.1455); auch hier eta recht groß

2, 3, 6: x\_hat = (42.5234;56.8496); k = 310; i\_null = 294;

dann c erstes Mal kleiner 0 weil b in einer Komponente falsch

- Idee: Stopping Conditions 1 vs. 2 zeigen: 1/t\*d ~= G -> klar, da Constraints im Subproblem

1/t\*d = G+v

-> Formulierungen der Stopping Conditions prüfen:

äquivalent (auch numerisch):

**LÖSUNG:** Stopping conditions waren nicht richtig implementiert G statt G+v (Unterschíed durch

constraints im Subproblem --> Reformulierung, Ergebnisse siehe Unterfrage 3)

1. C bzw. c (und delta) irgendwann < 0

Lösungsansätze:

* + prüfe, wann das auftritt: für air-datensatz (kurz) nach 310 Iterationen
  + warum? b stimmt nicht

! auffällig: t ist da bei t\_min angekommen -> bessere t-Veränderung???

! auffällig: in x enthält lauter gleich Werte -> prüfen, wie das sein kann

! auffällig: eta wird sehr groß; aber kein direkter Zusammenhang zwischen großem eta und negativem c feststellbar; eta springt auch viel zw. 2 und 1e~29

* alpha nicht "nah genung" bei 0 für rausnehmen aus bundle?

1. Test der neuen Stopping Conditions

* 3) stoppt genau nach 309 iterationen, 292 null steps; ZUFALL??? mit x\_hat = (42.5234;56.8496) (wie vorher), eta auch groß; anscheinend alles andere wie vorher
* 3a) alles genau wie oben
* 6), 6a) Stoppen jetzt wie 1)

**LÖSUNG:** einzig anderer Stopping test ist der von conv, inex 🡪 Unterschied verstehen?

* Vergleiche echtes Lernergebnisse die die jeweils gefundenen x\_hat = [C,eps] ergeben

**4 Hauptproblem:**

Quadprog löst Regression Problem seeeeeeeehr langsam (20 min) außer für Werte von eps = 56.8496 +-2; dann Lösung in <4sek

Lösungsansätze:

Unterprobleme:

1. Datensatz überhaupt nicht linear trennbar???

* Problem noch mal mit anderem Algorithmus lösen:
  + gelöst für erste 99 Datensätze des airfoil-sets, [C,eps] = [42.5234/3; 56.8496]  
    Ergebnis mit quadprog: W = [4.595762226113112e-05;-0.006082862314230;0.023484732332632;2.281886237215502;2.550177449925015e-04]  
    Ergebnis mit qpas: W = [4.595762226113345e-05;-0.006082862314230;0.023484732332632;2.281886237215502;2.550177449925003e-04]  
    Unterschied: 10^-15 🡪 eigentlich kein Unterschied
  + gelöst für erste 99 Datensätze des airfoil-sets, [C,eps] = [42.5234; 21.1455]  
    Ergebnis mit quadprog (unsicher): W = [0.001334505894522;2.200361584083511;1.279705679689002e+02;1.673168266205617;-2.931031012568054e+02]  
    Ergebnis mit qpas: Löst in Sekundenbruchteilen!  
    W = [-0.001586099220891;-1.817616651687124;2.098842793445674e+02;1.226857661128482;2.638118691075275]  
    Unterschied: 10^2, Unterschied W1\_qpas zu W2\_qpas ebenfalls 10^2

„Lösung 1“: Versuche erst mal mit Classification Datensätzen weiterzukommen

Wahrscheinlich keine lineare Regression möglich für airfoil Datensatz 🡪 siehe: <http://kernelsvm.tripod.com/>

„Lösung 2“: Schauen, ob schnell ein linearer (Regressions-)Datensatz gefunden werden kann

* Datensätze gefunden, die laut Quelle für lineare Regression geeignet
* Bei einem Test brach Algorithmus schon nach erstem Schritt ab – Warum???

**Klassifizierung anschauen**

* Datensätze aus Kunapuli benutzen (für Vergleichbarkeit)
* Schauen, ob Classification auch über least sqares Lossfunktion ausgerechnet werden kann 🡪 Vergleich mit Kunapuli
  + Inner Level Zielfunktion bei Kunapuli:  
    Unterschiede zu Moore:  
    λ als Regularisierer anstatt C 🡪 numerisch stabiler [Kunapuli, S.47]  
    ξ-Reformulierung hat zusätzlich Nebenbedingung ξ ≥ 0 🡪 bei Moore nicht, weil ξ in Zielfunktion quadratisch vorkommt?
  + Outer Level Zielfunktionen bei Kunapuli:  
    Durchschnitt der falsch klassifizierten Punkte: 🡪 diskontinuierliche Funktion; kann als Optimierungsproblem umgeschrieben werden 🡪 KKT-Bedingungen als Nebenbedingungen der Zielfunktion 🡪 ABLEITUNG???  
      
    Auch möglich: Hinge Loss
  + Logistic loss possible: 🡪 log{1+e^[-y(x’w-b)]}
  + Mean squared error possible? 🡪 [1-y(x’w-b)]^2 🡪 passt nicht gut zu anderen Loss Functions; eher nicht nehmen 🡪 laut Internet möglich, aber einige “Probleme”
* Löse für Pima-Set mit Moore inner Level, hinge-loss outer Level; vergleiche wie nah an Ergebnissen von Kunapuli

**Warum Cross Validation?**

* Methode gegen Overfitting 🡪 nicht nur „in-smaple-error“, sondern auch Generalisierungsfehler wird abgeschätzt
* Ist nicht „tragende Idee“ des Bilevel Problems aber in der Praxis und für eine sinnvolle Vergleichbarkeit der Fehler unerlässlich