Zusatztext zur Vorlesung "Optimierung"

Hier werden einige kleine Zutaten zu den vorhandenen Skripten aufgelistet, aber die Beweise werden i.a. nur skizziert und an der Tafel im Detail vorgeführt.

1 Anwendungen der linearen Optimierung

1.1 Minimaxaufgaben

1.1.1 Problemstellung

Gegeben sei ein überbestimmtes lineares Gleichumgssystem

$$By = z,\ B \in I\!R^{m\times k}.$$

Die Grundidee bei der Lösung solcher Probleme ist, stattdessen eine Fehlerminimierung zu versuchen. Das ist, nebenbei, ein Standardtrick bei allen Arten von "unlösbaren" Problemen. Man wähle also eine Norm $\|.\|$ auf $I\!R^m$ und minimiere

$$\min_{y\in I\!R^k} \|z - By\|.$$

Das Ergebnis hängt von der gewählten Norm ab. Im Falle $\|.\| = \|.\|_2$ bekommt man die klassische Ausgleichsrechnung (Methode der kleinsten Quadrate von Gauß). Sie führt (in der Theorie!) auf das Gaußsche Normalgleichungssystem $B^TBy = B^Tz$, das man aber aus Stabilitatsgründen besser gar nicht erst aufstellt. Stattdessen verwendet man geeignete Orthogonaltransformationen, aber dieses Thema gehört in die Numerikvorlesung und nicht hierher. Man kann das Ganze zwar auch als quadratische Optimierungsaufgabe sehen, aber das werden wir erst später tun.

Im Falle $\|.\| = \|.\|_\infty$ bekommt man ein **Minimaxproblem**

$$\min_{y\in I\!R^k} \max_{1\le i\le m} |z_i - \sum_{j=1}^{k} b_{ij}y_j| \tag{1}$$

und im Falle $\|.\| = \|.\|_1$ das L_1**-Problem**

$$\min_{y\in I\!R^k} \sum_{i=1}^{m} |z_i - \sum_{j=1}^{k} b_{ij}y_j|.$$

Das riecht nach nichtlinearer Optimierung, aber läßt sich als lineare Optimierung schreiben, denn es gibt ein paar

1.1.2 Standardtricks

Es seien $f, f_1, f_2, \ldots$ affin-lineare Ausdrücke.

Trick 1
Kommt irgendwo $|f|$ vor, so setzt man eine Gleichung $f = u - v$ mit neuen Variablen $u, v \geq 0$ an und ersetzt $|f|$ durch $u + v$.

Trick 2
Kommt irgendwo $\max(f_1, f_2, \ldots)$ vor, so führt man neue Ungleichungen $f_j \leq u$ mit einer neuen Variablen u ein.

Trick 3
Kommt irgendwo $\min(f_1, f_2, \ldots)$ vor, so führt man neue Ungleichungen $f_j \geq v$ mit einer neuen Variablen v ein.

Achtung: Die beiden letzten Tricks helfen nur, wenn man u klein und v gross halten kann (siehe Minimaxproblem). So etwas muß man in der Regel irgendwie in die Zielfunktion einbauen, wenn es nicht schon ohnehin drin ist.

1.1.3 Anwendung der Standardtricks auf Minimaxprobleme

Standardtrick Nummer 2 bei Minimaxproblemen ergibt eine Umformulierung als lineares Optimierungsproblem:

> Minimiere $\epsilon \geq 0$ unter den Nebenbedingungen
>
> $$-\epsilon \leq z_i - \sum_{j=1}^{k} b_{ij} y_j \leq \epsilon, \ 1 \leq i \leq m$$
>
> und den $k + 1$ Variablen $\epsilon, y_1, \ldots, y_k$,

denn dann hat man

$$\max_{1 \leq j \leq m} |z_i - \sum_{j=1}^{k} b_{ij} y_j| \leq \epsilon \rightarrow Min.$$

Das bedeutet bei vektorieller Ausformulierung gerade

$$-\epsilon \mathbf{1} \leq z - By \leq \epsilon \mathbf{1}$$

oder

$$\begin{array}{rcrcr} By & - & \epsilon \mathbf{1} & \leq & z \\ -By & - & \epsilon \mathbf{1} & \leq & -z \end{array}$$

und läßt sich als "Dual"problem

$$\begin{array}{rcl} \underbrace{\begin{pmatrix} B & -\mathbf{1} \\ -B & -\mathbf{1} \end{pmatrix}}_{=:A^T} \underbrace{\begin{pmatrix} y \\ \epsilon \end{pmatrix}}_{=:w} & \leq & \underbrace{\begin{pmatrix} z \\ -z \end{pmatrix}}_{=:p} \\ A^T w & \leq & p \\ b^T w := (\mathbf{0}_k^T, -1)^T w & = & -\epsilon \to Max! \end{array}$$

schreiben.

1.1.4 Dualisierung bei Minimaxproblemen

Das zugehörige Dualproblem zum Minimaxproblem ist also das Primalproblem

$$\begin{array}{rcl} Ax & = & b \\ x & \geq & 0 \\ p^T x & = & Min! \end{array}$$

zu obigem "Dual"problem, d.h.

$$\begin{array}{rcl} \underbrace{\begin{pmatrix} B^T & -B^T \\ -\mathbf{1}^T & -\mathbf{1}^T \end{pmatrix}}_{=A} \underbrace{\begin{pmatrix} u \\ v \end{pmatrix}}_{:=x} & = & \underbrace{\begin{pmatrix} \mathbf{0}_k \\ -1 \end{pmatrix}}_{=b} \\ u & \geq & 0 \\ v & \geq & 0 \\ p^T x = z^T(u-v) & = & Min! \end{array} \tag{2}$$

Es ist klar, daß das Ausgangs-Minimax-Problem (als Minimierungsproblem für ϵ) eine nach unten beschränkte Zielfunktion und eine nichtleere zulässige Menge hat. Deshalb ist es lösbar, ebenso das obige Dualproblem. Im Folgenden werden wir zwecks Ausschaltung gewisser seltener Sonderfälle annehmen, daß der Wert ϵ^* im Optimalpunkt positiv ist.

Die Komplementarität der Optimallösungen $\epsilon^*,\ y^*,\ u^*,\ v^*$ liefert die Gleichungen

$$\begin{array}{rcl} (x^*)^T(p - A^T w^*) & = & 0, \quad \text{d.h.} \\ u_j^*(z_j - (By^*)_j + \epsilon^*) & = & 0, \quad 1 \leq j \leq m \\ v_j^*(-z_j + (By^*)_j + \epsilon^*) & = & 0, \quad 1 \leq j \leq m. \end{array}$$

Ferner kann die zur Optimallösung $(x^*)^T = ((u^*)^T, (v^*)^T)$ gehörige Ecken-Indexmenge nicht mehr als $k+1$ Elemente enthalten, denn das ist die Zeilenzahl von A. Man kann dann die zwei Indexmengen $I_+ := I_{u^*}$ und $I_- := I_{v^*}$ mit zusammen nicht mehr als $k+1$ Elementen hernehmen und feststellen, daß

$$\begin{array}{rcll} (By^*)_j - z_j & = & +\epsilon^* & \text{für alle } j \in I_+ \\ (By^*)_j - z_j & = & -\epsilon^* & \text{für alle } j \in I_- \end{array} \tag{3}$$

gilt. Im Falle $\epsilon^* > 0$ sind die beiden Indexmengen disjunkt. Der Fehler "alterniert" also im Vorzeichen an den Komponenten mit Indizes $j \in I_+ \cup I_-$ und nimmt dort betragsmäßig seinen Extremwert ϵ^* an. In allen anderen Komponenten gilt wegen der Optimalität der Minimaxlösung noch

$$|(By^*)_j - z_j| \leq \epsilon^*,\ 1 \leq j \leq m.$$

Man spricht dann von einer "Alternante".

Satz 1 *Ein Minimaxproblem der Form* (1) *hat immer eine Lösung, die in einer gewissen Anzahl von Komponenten des m-dimensionalen Bildraums alterniert, d.h. betragsmässig den Optimalfehler ϵ^* annimmt. Im Falle $\epsilon^* > 0$ gibt es eine maximal $(k+1)$-elementige Teilmenge $I := I_+ \cup I_-$ von $\{1, \ldots, m\}$ mit* (3)*. Sie hat die Eigenschaft, daß das auf die Komponenten mit Indizes aus I eingeschränkte Minimaxproblem dieselbe Lösung hat, d.h. die übrigen Komponenten hätte man gar nicht betrachten müssen, wenn man sie vorab gekannt hätte.*

Wir müssen nur noch den Nachsatz beweisen. Das machen wir allgemeiner:

Satz 2 *Es sei ein lösbares Normalformproblem*

$$Ax = b,\ x \geq 0,\ p^T x = \min_x,\ A \in I\!R^{m \times n},\ x, p \in I\!R^n,\ b \in I\!R^m$$

mit Optimallösung x^* und zugehöriger Indexmange X^* gegeben. Dann löst x_{X^*} das Problem

$$A_{X^*} z = b,\ z \geq 0,\ p_{X^*}^T z = \min_z,\ A_{X^*} \in I\!R^{m \times |X^*|},\ z, p_{X^*} \in I\!R^{|X^*|},\ b \in I\!R^m$$

und läßt sich ohne alle Optimierung als Lösung des Gleichungssystems $A_{X^*} z = b$ ausrechnen. Die Optimallösung w^* des Dualproblems des Ausgangsproblems ist als Lösung des Systems $A_{X^*}^T w^* = p_{X^*}$ direkt ausrechenbar, und sie löst das zum obigen eingeschränkten Problem duale Problem.

Die Optimalität ist klar, weil x_{X^*} für das zweite Problem zulässig ist, die Zielfunktionswerte $p^T x^* = p_{X^*}^T x_{X^*}$ gleich sind, und das zweite Problem eine Einschränkung des ersten ist, d.h. keinen kleineren optimalen Zielfunktionswert haben kann. Das System $A_{X^*} z = b$ ist lösbar und hat maximalen Spaltenrang, also ist x_{X^*} dadurch eindeutig bestimmt. Die Berechenbarkeitsaussage über w^* gilt immer, und dieser Vektor ist zulässig und optimal für das Dualproblem des eingeschränkten Problems. □

Das eingeschränkte Problem des obigen Satzes ist nur formell ein Optimierungsproblem, denn es gilt $|X^*| \leq m$ und somit ist das primale Ergebnis nicht

verwunderlich. Die interessante Ausage ist die zum Dualproblem, weil sie besagt, dass man bei Vorab-Kenntnis der optimalen "aktiven" Restriktionen in $A^T w \leq p$ sich das Leben leicht machen könnte, indem man $A^T_{X^*} w^* = p_{X^*}$ löst.

Die Anwendung dieses Satzes auf Minimaxprobleme mit Alternante entnimmt die optimale Indexmenge aus dem Normalformproblem als Duales zum Minimaxproblem und wendet den obigen Satz an. Dabei ist eine Spaltenselektion von A eine Zeilenselektion von B, und das im Satz gemeinte Dualproblem ist genau ein Minimaxproblem mit Einschränkung der betrachteten Komponenten aus $\{1, \ldots, m\}$ auf die Komponenten mit Indizes aus der Alternante.

1.1.5 Programmbeispiel zu Minimaxproblemen

In MATLAB kann man Minimaxaufgaben einfach (und ineffizient) durch einen passenden Aufruf von `linprog` bewerkstelligen, obwohl ein duales Simplexverfahren sicher besser wäre:

```
function [x, fval]=myminimax(A,b)
[m n]=size(A);
B=[A -ones(m,1); -A -ones(m,1)];
p=[b; -b];
z=[zeros(n,1) ; 1];
options = optimset('LargeScale','off')
[y fval]=linprog(z,B,p,[],[],[],[],[],options);
x=y(1:n);
```

Das Kommando `options = optimset('LargeScale','off')` dient zur exakteren Ausrechnung der Ecke, denn das ansonsten verwendete Innere-Punkte-Verfahren liefert Ergebnisse, die manchmal ziemlich neben der Theorie liegen, weil sie keiner exakten Ecke entsprechen.

Ein passender Treiber ist

```
clear all;
t=-1:0.15:1;
% Punktesatz
f=t.^2-0.2*t.^3+0.02*(2*rand(size(t))-1);
```

```
% verrauschte Daten
ft=t.^2-0.2*t.^3; % Originaldaten
A=[ones(size(t))' t' t.^2' t.^3' t.^4']
% Approximationsmatrix, Gread <=4
[x fval]=myminimax(A,f')  % Minimaxrechnung
g=A*x % Ergebnis in Funktionswerten
xset=find(abs(f'-g)>fval-100*length(t)*eps)
% hole Extremalpunktindizes
plot(t,ft,t,f,'.',t,g,'+',t(xset),f(xset),'o')
% Plotten Funktion, Daten, Reproduktion
figure(2)
plot(t,ft'-g,t,f'-g,'.',t(xset),f(xset)'-g(xset),'o')
% Plotten Fehlerfunktion
```

und in der zugehörigen Plotausgabe sieht man die Alternationspunkte an den Stellen, wo die kleinen Punkte (verrauschte Daten, Komponenten von z) von den zugehörigen Kreisen (Komponenten von By^*) am weitesten, nämlich um ϵ^* entfernt liegen. Im Beispiel ist $k = 5$ und es gibt $k + 1 = 6$ Alternationspunkte.

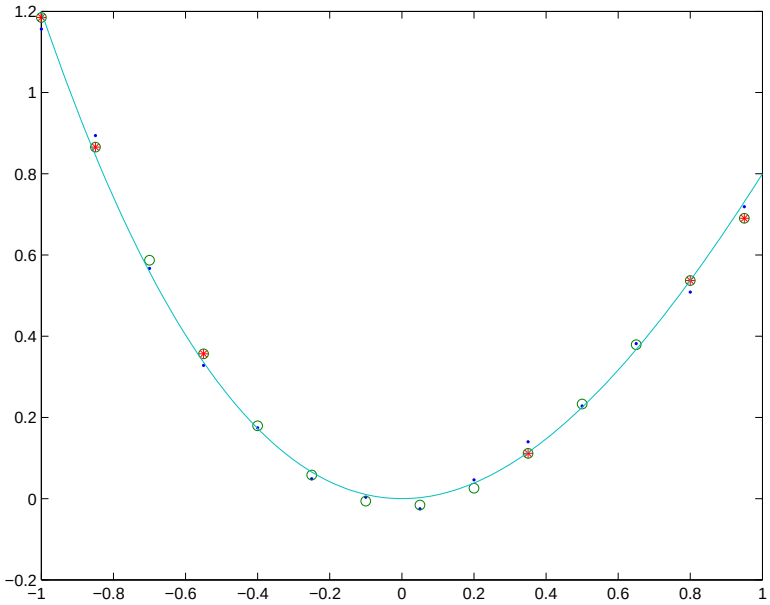

Abbildung 1: Ausgabe zum Minimaxproblem

1.1.6 Noch etwas zur Dualität

Das Optimierungsproblem (2) kann man mit $s := u - v,\ u, v \geq 0$ noch etwas umformulieren in

$$\begin{array}{rcl} B^T s & = & \mathbf{0}_k \\ \sum_j (u_j + v_j) = \mathbf{1}^T(u+v) = \|s\|_1 & = & 1 \\ z^T s & = & Min! \end{array} \tag{4}$$

was wieder einmal täuschend nichtlinear aussieht.

Die Zielfunktion des obigen Problems wird wegen unserer Annahme $\epsilon^* > 0$ sicher negativ, nämlich im Optimalfall gleich $-\epsilon^*$, so daß man auch $\|s\|_1 \leq 1$ zulassen kann, ohne die Lösungsmenge zu verändern. Ist nämlich $s^* \neq 0$ eine Lösung des erweiterten Problems mit $\|s\|_1 < 1$ und $z^T s^* < 0$, so erfüllt $s^*/\|s^*\|_1$ das auf $\|s\|_1 = 1$ eingeschränkte Problem mit kleinerem Zielfunktionswert, was nicht möglich ist.

Gleichung (4) zeigt also, dass die Dualitätstheorie des Minimaxproblems für $B \in I\!R^{m\times k},\ z \in I\!R^m$ die Ausage

$$\min_{y\in I\!R^k} \|By - z\|_\infty = \max_{s\in I\!R^m, B^T s=0, \|s\|_1\leq 1} |z^T s|.$$

liefert. Analog gilt aber auch

$$\min_{y\in I\!R^k} \|By - z\|_1 = \max_{u\in I\!R^m, B^T u=0, \|u\|_\infty\leq 1} |z^T u|$$

wobei die Normen $\|.\|_1$ und $\|.\|_\infty$ vertauscht sind. Der Beweis war als Übungsaufgabe gestellt und wird hier kurz skizziert. Das L_1-Problem ist mit unseren Standardtricks als

$$B(y^+ - y^-) - z = u^+ - u^-, \mathbf{1}^T(u^+ + u^-) = min!$$

zu schreiben, und es wird dualisiert zu

$$B^T u = 0, -\mathbf{1} \leq u \leq \mathbf{1}, z^T u = Max,$$

was zu beweisen war.

Eigenartigerweise transformiert das Dualisieren also die $\|.\|_1$-Norm in die $\|.\|_\infty$-Norm und umgekehrt. Das ist kein Zufall, sondert lehrt, dass **der Dualitätsbegriff der Optimierung zusammenfällt mit dem der normierten Vektorräume**. Um das zu erklären, nehmen wir einen normierten

Vektorraum V mit (primaler) Norm $\|.\|_V$ und bilden seinen (topologischen) Dualraum

$$V^* := \{\lambda \ : \ V \to I\!R \ : \ \text{linear und beschränkt}\}$$

wobei Beschränktheit eines Funktionals λ meint, daß eine Konstante c_λ existiert mit

$$|\lambda(v)| \leq c_\lambda \|v\|_V \text{ für alle } v \in V,$$

und diese Eigenschaft ist äquivalent zur Stetigkeit von λ als reellwertige Abbildung auf einem normierten Vektorraum. Dann kann man eine (duale) Norm auf dem topologischen Dualraum V^* definieren als

$$\|\lambda\|_{V^*} := \sup_{v \neq 0} \frac{\lambda(v)}{\|v\|_V} \leq c_\lambda.$$

Im Sonderfall $V = I\!R^n$ ist V^* nicht nur algebraisch isomorph zu V, sondern auch *topologisch*, d.h. es gibt einen *stetigen* Isomorphismus zwischen V und V^*. Deshalb ist auf $V = I\!R^n$ die zu einer Norm $\|.\|_V$ duale Norm definiert als

$$\|z\|_{V^*} := \sup_{v \neq 0} \frac{z^T v}{\|v\|_V},$$

wobei wir benutzt haben, wie Funktionale des $I\!R^n$ als Dualraum auf den $I\!R^n$ als "Primalraum" wirken. Als Konsequenz bekommt man

$$z^T v \leq \|z\|_{V^*} \|v\|_V \text{ für alle } z, v \in I\!R^n.$$

Bei dieser Dualitätsbeziehung in normierten Vektorräumen erweisen sich die Normen $\|.\|_p$ und $\|.\|_q$ als dual zueinander, sobald $\frac{1}{p} + \frac{1}{q} = 1$ gilt, und dabei kann man $1 \leq p, q \leq \infty$ zulassen. Die p-Norm $\|.\|_p$ für $1 \leq p < \infty$ wird dabei definiert über

$$\|x\|_p^p := \sum_{j=1}^{n} |x_j|^p \text{ für alle } x \in I\!R^n,$$

und der zugehörige Beweis verwendet die **Minkowskische Ungleichung**

$$z^T v \leq \|z\|_p \|v\|_q \text{ für alle } v \in I\!R^n, \ 1 \leq p \leq \infty, \ \frac{1}{p} + \frac{1}{q} = 1.$$

Die drei wichtigsten Fälle sind der "selbstduale" euklidische Fall $p = q = 2$ und die oben schon bemerkten Situationen $p = 1, \ q = \infty$ und umgekehrt.

1.2 Lernen mit Kernen

1.2.1 Problemstellung

Eine wichtige heutige Anwendung der Optimierung ist das "maschinelle Lernen". Das wurde in früheren Jahren bevorzugt mit neuronalen Netzen durchgeführt, aber es hat sich gezeigt, dass "kernbasierte" Lernverfahren leistungsfähiger sind, weil sie nicht an die biologische Modellbildung gebunden sind.

Gesucht ist ein System, das auf Reize x Reaktionen y produziert, also (mathematisch) eine Abbildung $f : X \to Y$ darstellt. Ein System, das Eingaben x in zwei Kategorien (gut $\Leftrightarrow$ schlecht, spam $\Leftrightarrow$ kein spam) klassifiziert, benutzt die Wertemenge $Y = \{-1, +1\}$. In anderen Fällen werden die Reaktionen $y \in Y$ reellwertig sein, etwa wenn Grundstückspreise aus diversen Informationen geschätzt werden sollen (Regression, $Y = IR$). Im allgemeinen trägt die Menge X der Reize oder Eingaben keine mathematische Struktur, denn sie kann z.B. auch aus Bildern oder Texten bestehen.

Neben anderen Formen des maschinellen Lernens ist das **supervidierte** Lernen (supervised learning) besonders wichtig. Es benutzt vorgegebene *Trainingsdaten*, die als Paare $(x_j, y_j) \in X \times Y,\ 1 \le j \le m$ vorliegen und von einem Supervisor, Trainer oder *master mind* als Soll-Reaktionen $y_j = f(x_j)$ anerkannt sind. Unter *Training* versteht man dann die Berechnung einer Abbildung g, die einigermaßen gut die Trainingsdaten reproduziert, d.h. es sollte gelten

$$y_j \approx g(x_j),\ 1 \le j \le m.$$

Nach dem Training wird dann die "gelernte" Abbildung g (es sollte besser "gelehrte" heißen) auf die reale Welt losgelassen und muß ihren Wert beweisen, indem sie zu ganz neuen Eingaben x eigene Ausgaben $g(x)$ macht. Deshalb verwendet man zusätzliche *Testdaten*, die man nach dem Lernen einsetzt, um die Qualität des Gelernten zu überprüfen. Gewisse Ähnlichkeiten mit dem mathematischen Übungbetrieb liegen auf der Hand: die Vorlesungen und die Übungsaufgaben sind die Trainingsdaten, und die abschließenden Klausuraufgaben machen einen Praxistest an bisher unbekannten Aufgaben.

1.2.2 *Feature Maps* und Kerne

Auf einer unstrukturierten Menge kann man keine brauchbare Mathematik treiben. Also muß eine Struktur her. Das geschieht dadurch, daß man zu jeder denkbaren Eingabe $x \in X$ eine möglichst lange Liste von quantifizierbaren Eigenschaften assoziiert. Man beschreibt also x durch einen *feature vector* $\phi(x)$, der möglichst viel Typisches über x aussagt.

Beispiel: Will Aschenputtel die guten von den schlechten Erbsen unterscheiden, so sollte sie vielleicht Farbe, Größe, Gewicht und Form der Erbsen in den *feature vector* aufnehmen.

Mathematisch wird das durch eine Abbildung (*feature map*)

$$\phi \ : \ X \to \mathcal{F}$$

mit Werten in einem *feature space* $\mathcal{F}$ beschrieben, und dieser Raum sollte ein Vektorraum über IR sein, der ein Skalarprodukt $\langle . \rangle$ trägt, damit man dort "euklidisch messen" kann.

Ab sofort wird dann fast nur noch mit den *feature vectors* $\phi(x) \in \mathcal{F}$ statt mit den Eingaben $x \in X$ gearbeitet. Das hat zur Folge, daß Eingaben x und y mit $\phi(x) = \phi(y)$ nicht mehr unterscheidbar werden, d.h. man arbeitet praktisch "modulo gleicher features". Deshalb sollte man sicher gehen, dass die *feature map* so reichhaltig ist, dass sie alle wichtigen Unterschiede zwischen möglichen Eingaben auch berücksichtigt.

Ein zugehöriger **Kern** ist dann

$$K \ : \ X \times X \to IR, \ \ K(x,y) := \langle \phi(x), \phi(y) \rangle \text{ für alle } x, y \in X.$$

Er erzeugt eine "schöne" mathematische Struktur auf X, z.B einen (schwachen) Abstandsbegriff

$$d^2(x,y) := \|\phi(x) - \phi(y)\|^2_{\mathcal{F}} := K(x,x) - 2K(x,y) + K(y,y) \text{ für alle } x, y \in X,$$

was man durch Ausmultiplizieren von

$$\|\phi(x) - \phi(y)\|^2_{\mathcal{F}} := \langle \phi(x) - \phi(y), \phi(x) - \phi(y) \rangle$$

sieht. Obendrein hat man jetzt auch plötzlich einen Vorrat von Funktionen auf der unstrukturierten Menge X, nämlich zu jedem $y \in X$ die Funktion

$$x \mapsto K(x,y) = \langle \phi(x), \phi(y) \rangle \text{ für alle } x \in X.$$

1.2.3 Lernen mit Kernen

Hat man Trainingsdaten $(x_j, y_j) \in X \times IR, \ 1 \leq j \leq m$, so liegt es nahe, einen Ansatz der Form

$$g(x) := \sum_{i=1}^{m} \alpha_i K(x, x_i) = \sum_{i=1}^{m} \alpha_i \langle \phi(x), \phi(x_i) \rangle, \ \alpha_i \in IR$$

zu machen und das "Lernen" von g als Berechnung geeigneter Koeffizienten $\alpha_1, \ldots, \alpha_m$ zu verstehen. Dieser Ansatz läßt sich sogar durch ein Optimierungsargument in unendlichdimensionalen Räumen begründen (siehe unten Satz 17), aber das kann hier noch nicht dargestellt werden. Im Idealfall würde man also das lineare $m \times m$ Gleichungssystem

$$y_j = g(x_j) = \sum_{i=1}^{m} \alpha_i K(x_j, x_i) = \sum_{i=1}^{m} \alpha_i \langle \phi(x_j), \phi(x_i) \rangle,\ 1 \leq j \leq m \tag{5}$$

ansetzen, dessen Koeffizientenmatrix mit den Einträgen

$$K(x_j, x_i) = \langle \phi(x_j), \phi(x_i) \rangle,\ 1 \leq i, j \leq m$$

als **Kernmatrix** bezeichnet wird. Diese ist immer symmetrisch und positiv semidefinit (weil sie eine **Gramsche** Matrix ist), aber sie kann riesig und singulär sein. Obendrein darf die Lösung nicht dramatisch von einzelnen der Trainingsdaten abhängen, wenn sie einigermaßen "stabile" Resultate produzieren soll. Denn sobald sich Zufall und Fehler in die Eingabedaten einschleichen, wäre der Ausgang vollkommen ungewiss. Deshalb verwendet man diverse, meist durch einen stochastischen Hintergrund motivierte Tricks, die eine exakte Lösung des Systems (5) gar nicht erst versuchen, sondern ein simpleres Modell einsetzen, das nicht alle Trainingsdaten exakt reproduziert und weniger "anfällig" ist. Man hat immer eine Abwägung zwischen Reproduktionsgenauigkeit der Trainingsdaten und Stabilitat des Modells zu treffen.

Wir behandeln hier als Einführung nur den simplen Spezialfall, daß wir weniger Ansatzfunktionen als Daten benutzen und dann ein Minimaxproblem aufstellen. Das bekommt die Form

$$\epsilon = Min!,\ -\epsilon \leq y_j - \sum_{i=1}^{k} \alpha_i K(x_j, y_i) \leq \epsilon,\ 1 \leq j \leq m \tag{6}$$

mit $k < m$ und gewissen $y_1 \ldots, y_k \in X$, die wir eventuell als Teilmenge der Trainingsdaten $x_1, \ldots, x_m$ wählen. Dieses Problem läßt sich mit den Methoden des vorigen Abschnitts behandeln, und wir bekommen im Allgemeinen gewisse Alternanten als Auswahl von maximal $k+1$ Punkten aus den Trainingspunkten $x_1, \ldots, x_k$. Raffiniertere Techniken folgen später.

1.2.4 Beispiel: Klassifikation als Minimaxaufgabe

Hier ist ein halbwegs kommentiertes Beispiel, in dem ein nichtsahnendes Programm lernen soll, Punkte innerhalb und außerhalb des Kreises

$$(x - 0.5)^2 + (y - 0.5)^2 = 0.1$$

sauber zu unterscheiden. Als Trainingsdaten werden 50 zufällige Punkte x_j aus $[0,1]^2$ genommen und die Werte y_j auf 1 für draußen liegende und auf -1 für innen liegende Punkte gesetzt. Die *feature map* wird so gebaut, daß ein Gitter aus Punkten $z_k \in [0,1]^2$ vorgegeben wird, und dann besteht $\phi(x)$ für festes $x \in I\!R^2$ aus dem Vektor aller $\|x - z_k\|_\infty$, wobei die z_k über das Gitter laufen. Die "features" von x sind also die Abstände zu den Gitterpunkten; sie haben nichts mit der zu lernenden Figur zu tun. Durch Verfeinerung des Gitters kann man das Auflösungsvermögen des Lernprogramms leicht steigern, egal was da zu lernen ist.

Die Wahl der Ansatzpunkte y_i aus dem obigen Text wird sehr grob so gemacht, dass je 5 Trainingsdaten drinnen und draußen ausgewählt werden. Weil die Trainingsdaten ohnehin zufällig sind, kann man die ersten 5 drinnen und die ersten 5 draussen nehmen. Der obere Plot zeigt die Testdaten (+ und o), den exakten Kreis (affin verzerrt, also als Ellipse) und die Ansatzpunkte (x). Man sieht, welche Testpunkte als Ansatzpunkte ausgesucht wurden.

Der Rest ist dann ziemlich klar: man setzt ein Minimaxproblem auf und löst es. Danach werden 250 zufällige Testdaten in $[0,1]^2$ generiert und getestet, ob sie das Programm richtig klassifiziert. Dazu wertet man g an jeder Teststelle aus, und deklariert einen Testpunkt als "drinnen", wenn g negativ ist, sonst als "draußen". Schließlich haben wir ja die Trainingswerte y_j auf 1 für draußen liegende und auf -1 für innen liegende Punkte gesetzt. Das Ergebnis zeigt dann der zweite Plot.

Der dritte zeigt die Alternationspunkte, d.h. diejenigen Trainingspunkte, an denen der Fehler extremal war. Man könnte mit diesen als Ansatzpunkten das Verfahren wiederholen, denn in der Regel gibt es genau einen Alternationspunkt mehr als Ansatzpunkte. Hier ist reichlich Platz zum Experimentieren. Noch etwas: Der Zufallsgenerator wurde nicht rückgesetzt, so daß alle neuen Rechnungen verschieden ausfallen. Es ist ziemlich einfach, andere Parameter durchzuspielen und das Programm andere Formen lernen zu lassen. Man wird immer sehen, dass die Klassifizierung von neuen Testdaten dort besonders schlecht ausfällt, wo keine oder nur wenige Trainingsdaten vorhanden sind. Im Beispiel sieht man, das das Programm den linken Rand nicht genau festlegen kann, weil ihm nicht "klar" ist, ob die Ellipse nicht "links" etwas kleiner ist. Im Prinzip benutzt das Programm eine kleinere Figur um die als "innen" vorgegebenen Trainingsdaten. Das kann man ihm nicht übelnehmen.

Fazit: *Was nicht geübt wird, kann auch nicht gelernt werden* (alte Grundregel des Mathematik- und Klavierstudiums).

```
clear all;
np=50; % Anzahl der Trainingsdaten
% hier die Trainingsdaten, zufällig in [0,1]
randx=rand(np,1);
randy=rand(np,1);
radsq=0.1; % Radius zum Quadrat
testval=(randx-0.5).^2+(randy-0.5).^2;
% denn wir wollen einen Kreis lernen
kreisx=0.5+sqrt(radsq)*cos(2*pi*[0:0.01:1]);
% exakter Kreis, feine Plotdaten
kreisy=0.5+sqrt(radsq)*sin(2*pi*[0:0.01:1]);
xset=find(testval<=radsq); % holt Indizes der inneren Punkte
val=ones(np,1);  % und wir setzen die Trainingswerte
val(xset,1)=-1; % drinnen -1, draussen +1
posset=find(val>0);  % zum Plotten splitten wir die Daten
negset=find(val<0);
% Wir müssen jetzt die feature vectors wählen
[X Y]=meshgrid(0:0.1:1);
% ein gleichmäßiges Gitter zwecks feature vectors
XX=X(:); % die x- Gitterwerte als Liste
YY=Y(:); % dito y
nd=length(XX); % das wird dann die Länge der feature vectors
fv=zeros(np,nd);     % Matrix der feature vectors aufbauen
for i=1:nd    % wir nehmen die Distanzwerte zum Gitter
    fv(:,i)=max(abs(randx(:,1)-XX(i)),abs(randy(:,1)-YY(i))); %
    %  das war die  Maximumsnorm - Distanz
end
% Jetzt wählen wir die Ansatzpunkte
nq=5; % halbe Anzahl der Ansatzdaten
% Wir nehmen je die ersten nq
% aus den inneren und äußeren Punkten
% Ziemlich wahllos, das geht besser.....
Xset=[posset(1:nq) negset(1:nq)]
% und das war schon unsere Selektion
subplot(3,1,1) % und plotten sie
% als ersten Plot in einer 3x1 Konfiguration
plot(randx(posset),randy(posset),'+',kreisx,kreisy)
hold on  % das friert die Skalierung ein
plot(randx(negset),randy(negset),'o')
plot(randx(Xset),randy(Xset),'x')
axis([0 1 0 1])
```

```
title('Trainings- und Ansatzdaten (+,o und x)')
% Das ergibt eine nichtquadratische Kernmatrix
Kmat=fv*fv(Xset,:)';
[x fval]=myminimax(Kmat,val); % und rein ins Minimaxproblem
% Ab hier wird getestet
neval=250; % Anzahl der Testpunkte
npx=rand(neval,1); % und zufällige Auswahl
npy=rand(neval,1);
fp=zeros(neval,nd);
% deren feature vectors ausrechnen, wie oben
for i=1:nd
    fp(:,i)=max(abs(npx(:,1)-XX(i)),abs(npy(:,1)-YY(i)));
end
zp=fp*fv(Xset,:)'*x;
% das ist der Vorhersagewert des gelernten Modells
% Zum Plotten brauchen wir die Entscheidungen, wer
% drin ist und wer draussen
posfset=find(zp>0);
negfset=find(zp<0);
subplot(3,1,2)
plot(npx(posfset),npy(posfset),'+',kreisx,kreisy)
hold on
plot(npx(negfset),npy(negfset),'o')
axis([0 1 0 1])
title('Testdaten')
% und jetzt plotten wir noch Alternationspunkte
resid=abs(Kmat*x-val);
yset=find(resid>fval-0.0001);
posyset=find(val(yset)>0);
negyset=find(val(yset)<0);
% und plotten sie hier
subplot(3,1,3)
plot(randx(yset(posyset)),randy(yset(posyset)),'+',kreisx,kreisy)
hold on
plot(randx(yset(negyset)),randy(yset(negyset)),'o')
axis([0 1 0 1])
title('Alternationspunkte')
```

1.2.5 Beispiel: Klassifikation als Trennungsaufgabe

.

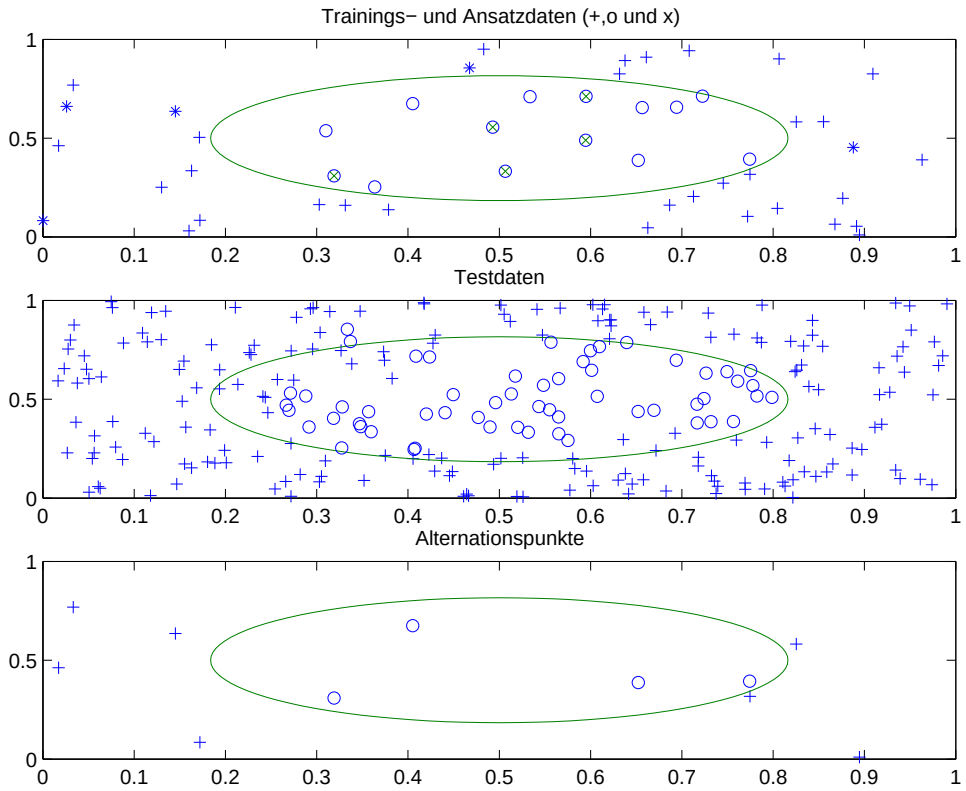

Abbildung 2: Ausgabe zum Lernproblem

(oder auch *Aschenputtel's support vector machine*).

Aschenputtel muß lernen, Erbsen in gute und schlechte zu klassifizieren. Sie erhebt jeweils m reellwertige Merkmale von ihren Erbsen, z.B. Durchmesser in mm, Gewicht in Gramm, etc. Sie hat von der bösen Stiefmutter einen Lernsatz mit n_+ guten und n_- schlechten Erbsen bekommen. Die Merkmale dieser Erbsen ergeben je eine $n_+ \times m$- und $n_- \times m$-Matrix, die Aschenputtel M^+ und M^- nennt. Allerdings sind n_+ und n_- viel größer als $m \geq 2$, so daß Aschenputtel, die sich im $I\!R^m$ gut auskennt, schnell sieht, daß die Zeilen von M^+ und von M^- als Vektoren des $I\!R^m$ durch eine Hyperebene im $I\!R^m$ trennbar sind. Es gibt also einen Vektor $x \in I\!R^m \setminus \{0\}$ und eine relle Zahl β, so daß

$$M^+x + \beta\mathbf{1} \geq 0, 0 \geq M^-x + \beta\mathbf{1}$$

gilt. Wer sich nicht so gut im $I\!R^m$ auskennt wie Aschenputtel, möge sich mal für ein paar "trennbare" Punkte des $I\!R^2$ klarmachen, wieso dies "Trennung" bedeutet.

Auf Grund dieser Trennbarkeit kommt Aschenputtel auf die gute Idee, zu jeder Erbse e den zugehörigen Merkmalsvektor $\phi(e) \in I\!R^m$ zu bilden, dann $f(e) := \phi(e)^T x + \beta$ auszurechnen, und Erbsen e mit $f(e) \geq 0$ als "gut" und solche mit $f(e) < 0$ als "schlecht" zu klassifizieren. Denn diese Regel würde auf allen Testerbsen richtige Ergebnisse bringen.

Sie merkt aber auch, dass es bei ihrem Testsatz unendlich viele solche trennende Hyperebenen gibt, und sie will eine optimale Hyperebene finden, die eine möglichst sichere Unterscheidung ermöglicht. Also "verbreitert" sie die Hyperebene $\{z \in I\!R^m \ : \ z^T x + \beta = 0\}$ auf einen "Streifen" $\{z \in I\!R^m \ : \ |z^T x + \beta| \leq \epsilon\}$ (der "Breite" $2\epsilon/\|x\|_2$, aber das ist hier nicht wichtig). Damit will sie einen möglichst breiten Streifen zwischen die Merkmalsvektoren der guten und schlechten Testerbsen legen. Sie will also ein maximales ϵ suchen, so daß

$$M^+ x + \beta \mathbf{1} \geq \epsilon \mathbf{1} > 0 \geq -\epsilon \mathbf{1} \geq M^- x + \beta \mathbf{1} \tag{7}$$

gilt. Weil man diese Ungleichungskette aber mit beliebig großen positiven Zahlen multiplizieren könnte, um ϵ hochzutreiben, muß Aschenputtel den Vektor x in Schach halten. Weil Aschenputtel (noch) nichts von quadratischer Optimierung weiss, fügt sie die Nebenbedingung $\|x\|_\infty \leq 1$ hinzu, von der sie weiss, dass sie sich "linearisieren" läßt. Jetzt hat sie ein wunderbares lineares Optimierungsproblem, und kann ihre Erbsen bis zum Beginn des Balls sehr zur Zufriedenheit der bösen Stiefmutter klassifizieren.

Als Übungsaufgabe wurde folgendes gestellt:

1. Wie sieht das komplette Optimierungsproblem von Aschenputtel aus, und was ist das Dualproblem?

2. Warum hätte Aschenputtel alle ihre Testerbsen bis auf höchstens $m+2$ wichtige wegwerfen können, ohne ein anderes Ergebnis zu bekommen?

3. Wodurch sind diese wichtigen "Stütz"erbsen bestimmt?

Man verwende dazu den Satz 2, der auch beim Beweis des Alternantensatzes wichtig war.

Hier ist eine Lösungsskizze. Das Problem ist

$$\begin{pmatrix} -M^+ & \mathbf{1} & -\mathbf{1} \\ M^- & \mathbf{1} & \mathbf{1} \\ I & 0 & 0 \\ -I & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} x \\ \epsilon \\ \beta \end{pmatrix} \leq \begin{pmatrix} 0 \\ 0 \\ \mathbf{1} \\ \mathbf{1} \\ = Max! \end{pmatrix}$$

und das Duale ist

$$\begin{pmatrix} -(M^+)^T & (M^-)^T & I & -I \\ \mathbf{1}^T & \mathbf{1}^T & 0^T & 0^T \\ -\mathbf{1}^T & \mathbf{1}^T & 0^T & 0^T \\ 0^T & 0^T & \mathbf{1}^T & \mathbf{1}^T \end{pmatrix} \begin{pmatrix} u \\ v \\ r \\ s \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ Min! \end{pmatrix}$$

Das Problem ist sicher lösbar, weil $\epsilon = 0$ wegen der vorausgesetzten Trennbarkeit erlaubt ist, aber beliebig große ϵ nicht mehr trennen würden. Die zulässige Menge ist also nicht leer, und die Zielfunktion ist nach oben beschränkt, also ist das Problem lösbar. Das Dualproblem ist ein Normalformproblem mit $m+2$ Zeilen, und deshalb haben Ecken maximal $m+2$ von Null verschiedene Komponenten. Wir haben also eine Indexmenge zu einer Optimallösung mit maximal $m+2$ Einträgen. Komplementarität liefert dann im Ausgangsproblem, dass die entsprechenden Zeilen des Ausgangsproblems exakt erfüllt sind, d.h. es gibt eine Anzahl von Indizes j und k mit $e_j^T M^+ x^* + \beta = \epsilon^*$ und $e_k^T M^+ x^* + \beta = -\epsilon^*$. Diese bestimmen die wichtigen "Testerbsen" nach dem Satz 2, und das löst Teile 2 und 3. Man nennt diese Vektoren "support vectors". Sie liegen auf dem "margin" des trennenden Streifens.

1.2.6 Aschenputtel's Programm und Ergebnis

```
clear all;
np=25 % Anzahl der guten Punkte
nn=25 % Anzahl der bösen Punkte
r=[0.2 0.5]; % Richtungsvektor der idealen Hyperebene
nor=[-0.5 0.2] % Normale dazu
bs=[0 0]; % Aufpunkt für Strahl auf Hyperebene
% wir gehen zufällig vor und berechnen Punkte
% entlang der Geraden und gleichzeitig links und rechts
for ip=1:np
    Mp(ip,:)=bs+rand(1,1)*r+0.2*rand(1,1)*nor;
    Mn(ip,:)=bs+rand(1,1)*r-0.2*rand(1,1)*nor;
end
% So, jetzt bauen wir das Aschenputtel-Problem auf
A=[-Mp ones(np,1) -ones(np,1);...
    Mn ones(nn,1) ones(nn,1);...
  eye(2) zeros(2,2); -eye(2) zeros(2,2)];
b=[zeros(np+nn,1) ;ones(4,1)];
p=zeros(4,1);
p(3,1)=-1;
% und lösen es
```

```
[x,fval]=linprog(p,A,b);
% Wir wollen die trennende Ebene malen
tt=-0:0.01:0.2; % das werden die x-Werte
% und es kommen die umgerechneten y-Werte
% dreier paralleler Geraden
y0=(       -x(4,1)-x(1,1)*tt)/x(2,1);
yp=( x(3,1)-x(4,1)-x(1,1)*tt)/x(2,1);
yn=(-x(3,1)-x(4,1)-x(1,1)*tt)/x(2,1);
% und die malen wir
plot(tt,y0,tt,yp,tt,yn)
hold on
% mit den gegebenen Daten
plot(Mp(:,1),Mp(:,2),'+',Mn(:,1),Mn(:,2),'o')
% Achtung, die Geometrie ist nicht euklidisch!
```

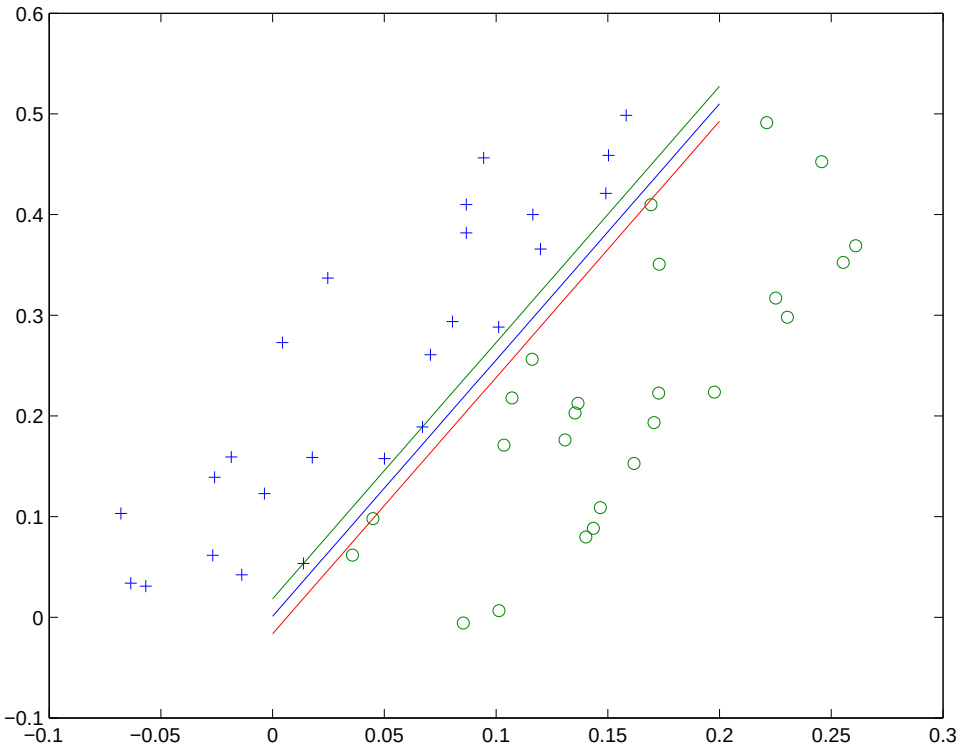

Abbildung 3: Ausgabe zum Aschenputtelproblem

Es sollten 4 Testerbsen ausreichen, um sauber zu klassifizieren, und das sind 4 Datenpunkte, die auf dem Rand des kritischen Streifens liegen.

Wie man sich von der Voraussetzung der Trennbarkeit befreit, wird später behandelt.

2 Konvexe Optimierung

2.1 Gâteaux-Differential

Es sei f eine konvexe Funktion auf einer nichtleeren konvexen ("zulässigen") Menge $\mathcal{M}$ in einem **nicht notwendig endlichdimensionalen** Vektorraum V über IR gegeben.

Lemma 3 *Es sei $x \in \mathcal{M}$ gegeben, und es sei $y \in V$ eine zulässige Richtung, d.h. $x + hy \in \mathcal{M}$ für $h \in [0, h_0]$ mit einem $h_0 > 0$. Dann sind die Differenzenquotienten*

$$\frac{f(x+hy) - f(x)}{h},\ h \in (0, h_0]$$

schwach monoton steigend als Funktion von h.
(Veranschaulichung durch Zeichnung!)

Beweisidee: man wählt $0 < s \leq t \leq h_0$ und schreibt $x + sy$ als Konvexkombination von x und $x + ty$. Darauf wendet man die Konvexitätsvoraussetzung von f an und rechnet die Behauptung herbei.

Lemma 4 *Es sei $x \in \mathcal{M}$ gegeben, und es seien $y,\ -y \in V$ zulässige Richtungen. Dann gilt*

$$\frac{f(x) - f(x - sy)}{s} \leq \frac{f(x + ty) - f(x)}{t}$$

und die linke Seite ist schwach monoton fallend als Funktion von s für kleine s.
(Veranschaulichung durch Zeichnung!)

Beweisidee: man schreibt x als Konvexkombination von $x - sy$ und $x + ty$. Darauf wendet man die Konvexitätsvoraussetzung von f an und rechnet die erste Behauptung herbei. Die zweite ergibt sich wie im Lemma 3.

Lemma 5 *Es sei $x \in \mathcal{M}$ gegeben, und es seien $y,\ -y \in V$ zulässige Richtungen. Dann ist f auf einer Umgebung von x auf der Strecke $[x - y, x + y]$ stetig.*

Beweisidee: Im vorigen Lemma müssen die Zähler der beiden Seiten gegen Null gehen, wenn s und t gegen Null gehen.

Lemma 6 *Es sei V endlichdimensional, und es sei $x \in \mathcal{M}$ ein innerer Punkt von $\mathcal{M}$, d.h. alle $y \in V$ sind zulässige Richtungen. Dann ist f in x stetig.*

Beweisidee: Man kann das vorige Lemma "gleichmäßig" für alle Richtungen anwenden, denn bei endlichdimensionalem V kann man die Richtungen auf die kompakte Einheitskugel einschränken.

Definition 7 *Es sei $x \in \mathcal{M}$ gegeben, und es seien $y,\ -y \in V$ zulässige Richtungen. Dann existieren (nach Lemma 4) die Limiten*

$$\lim_{s \searrow 0} \frac{f(x) - f(x - sy)}{s} =: -f'_+(x, -y) \leq f'_+(x, y) := \lim_{t \searrow 0} \frac{f(x + ty) - f(x)}{t}$$

und werden **Gâteaux-Richtungsableitungen** *im Punkt x in Richtung y und $-y$ genannt. Ist $f'_+(x, y)$ eine lineare Abbildung als Funktion von y, so spricht man vom* **Gâteaux-Differential**.

Allgemeiner:

Definition 8 *Es sei $x \in \mathcal{M}$ gegeben, und es sei $y \in V$ eine zulässige Richtung bezüglich $\mathcal{M}$ in x, aber er werde* **nicht** *vorausgesetzt, dass f oder $\mathcal{M}$ konvex seien. Wenn der Limes*

$$f'_+(x, y) := \lim_{t \searrow 0} \frac{f(x + ty) - f(x)}{t}$$

existiert, wird er **Gâteaux-Richtungsableitung** *im Punkt x in Richtung y genannt. Ist $f'_+(x, y)$ eine lineare Abbildung als Funktion von y, so spricht man vom* **Gâteaux-Differential**.

Lemma 9 *Die Gâteaux-Richtungsableitungen haben einige Eigenschaften:*

1. $f'_+(x, \alpha y) = \alpha f'_+(x, y)$, *für alle* $\alpha \geq 0$
2. *Ist f konvex, so ist $f'_+(x, y)$ konvex als Funktion von y auf dem Kegel der zulässigen Richtungen in x bezüglich $\mathcal{M}$. Deshalb kann man in beliebiger Weise Gâteaux-Richtungsableitungen von Gâteaux-Richtungsableitungen usw. bilden, sofern Konvexität vorliegt.*
3. *Ist f im klassischen oder Fréchet-Sinne in x differenzierbar mit der Ableitung $\nabla f(x)$, so gilt*

   $$(\nabla f(x))(y) = f'_+(x, y)$$

   *und ist als Funktion von y linear. Das erklärt den Begriff des Gâteaux-***Differentials**.

Hier kommt eine sehr einfache Verallgemeinerung dessen, was man von der Schule her kennt:

Satz 10 *Es sei f eine konvexe Funktion auf einer nichtleeren konvexen Menge $\mathcal{M}$ in einem nicht notwendig endlichdimensionalen Vektorraum V. Ferner sei $x \in \mathcal{M}$ ein zulässiger Punkt, in dem die Gâteaux-Richtungsableitungen in alle zulässigen Richtungen existieren. Dann gilt:*
x ist genau dann ein Minimum von f auf $\mathcal{M}$, wenn $f'_+(x, y) \geq 0$ für alle zulässigen Richtungen y in x gilt.

Beweisskizze: Für beide Richtungen wendet man Lemma 3 und die Definition der Gâteaux-Richtungsableitung an.

Ein Problem bei der Anwendung des obigen Satzes entsteht, weil x normalerweise "am Rand" von $\mathcal{M}$ liegt, und dann ist die Existenz von Gâteaux-Richtungsableitungen in alle zulässigen Richtungen nicht automatisch garantiert (Übungsaufgabe), sondern muß gesondert nachgewiesen werden. In vielen Fällen hilft aber eine allgemeine Differenzierbarkeit von f über diese Hürde hinweg.

Satz 11 *Es sei f eine* **nicht notwendig konvexe** *Funktion auf einer nichtleeren* **nicht notwendig konvexen** *Menge $\mathcal{M}$ in einem nicht notwendig endlichdimensionalen Vektorraum V. Ferner sei $x \in \mathcal{M}$ ein zulässiger Punkt, in dem die Gâteaux-Richtungsableitungen in alle zulässigen Richtungen existieren. Dann gilt: Ist x ein* **lokales** *Minimum von f auf $\mathcal{M}$, so folgt $f'_+(x, y) \geq 0$ für alle zulässigen Richtungen y in x.*

Beweisskizze: Das folgt aus der Definition der Gâteaux-Richtungsableitung.

Achtung:
Die Konvexität in Satz 10 liefert eine **notwendige und hinreichende** bedingung für ein **globales** Minimum, während Satz 11 zwar ohne Konvexität auskommt, aber dann nur eine **notwendige** Bedingung für ein **lokales** Minimum liefert.

Beide Sätze liefern **keine** Existenzaussage. Stattdessen liefern sie sogenannte **Variationsungleichungen** der Form

$$f'_+(x, y) \geq 0 \text{ für alle zulässigen Richtungen } y \text{ in } x$$

als notwendige und im konvexen Fall auch hinreichende Bedingungen für Optimallösungen. In vielen Fällen muß man damit zufrieden sein, insbesondere bei heiklen Optimierungsproblemen in unendlichdimensionalen Räumen.

Ist die Gâteaux-Ableitung $f'_+(x,y)$ in y linear und bilden die zulässigen Richtungen y einen linearen Raum V, so sind die obigen Variations**un**gleichungen äquivalent zu Variations**gleichungen**

$$f'_+(x,y) = 0 \text{ für alle zulässigen Richtungen } y \text{ in } x,$$

was sich in diversen Fällen sehr schön auswerten läßt, wie wir gleich sehen werden.

2.2 Lagrange-Multiplikatoren

In allen Texten über Optimierung treten gewisse "Lagrange-Multiplikatoren" mit gewissen Vorzeichenbedingungen auf. Sie ergeben sich formal immer über Funktionale, die gewisse konvexe Mengen "trennen", aber wir wollen sie hier durch etwas naheliegendere Argumente motivierend einführen.

Wir gehen der Einfachheit halber erst von einem konvexen Problem $f(x) = Min!$ auf dem IR^n mit differenzierbarer Zielfunktion f und $m < n$ affin-linearen Gleichungs-Nebenbedingungen $h(x) := Ax - b = 0$ aus. Satz 10 und die Bemerkung am Ende des vorigen Abschnitts besagen dann, daß die Variationsgleichung

$$f'_+(x,y) = (\nabla^T f(x))y = 0 \text{ für alle } y \text{ mit } Ay = 0$$

notwendig und hinreichend für eine Optimallösung x ist. Führen wir für einen Moment bei festem x die lineare Abbildung

$$B \ : \ IR^n \to IR, \ y \mapsto (\nabla^T f(x))y$$

ein, so haben wir die formale Situation

$$By = 0 \text{ für alle } y \text{ mit } Ay = 0 \tag{8}$$

für zwei lineare Abbildungen $A \ : \ U \to A(U) =: V, \ B \ : \ U \to W$ zwischen gewissen Vektorräumen $U, \ V, \ W$. So etwas tritt in der Mathematik sehr oft auf, wird aber in den Anfängervorlesungen nicht mit dem notwendigen Nachdruck behandelt.

Unter schwachen Zusatzvoraussetzungen neben (8) **faktorisiert** nämlich B über das **Bild** von A, d.h. es gibt eine lineare Abbildung $C \ : \ V = A(U) \to W$ mit

$$B = C \circ A.$$

Bevor wir die genauen Voraussetzungen für die Faktorisierung klären, stellen wir in unserem Fall fest, daß es dann einen Vektor $v \in I\!R^m$ geben muß, so daß

$$\nabla^T f(x) = v^T A$$

gilt, und das ist der einfachste Fall eines Vektors von "Lagrange-Multiplikatoren".

Im Falle endlichdimensionaler Vektorräume (d.h. also auch in unserem Fall) ist die Faktorisierung eine einfache Folgerung aus dem bekannten Isomorphiesatz

$$A(U) = V \simeq U/ker\ A,$$

denn man kann C auf $A(U) = V \simeq U/ker\ A$ durch

$$C(A(u)) := Bu$$

vertreterinvariant als lineare Abbildung definieren. Man kann sie auf jeden endlichdimensionalen Vektorraum T, der $V = A(U) \subseteq T$ als Untervektorraum hat, problemlos fortsetzen, so daß wir keine Rangvoraussetzung an unsere Matrix A brauchen und unsere reellwertige lineare Abbildung C als Funktional auf dem ganzen $I\!R^m$ wählen können. Im unendlichdimensionalen Fall muß man etwas aufpassen und Zusatzforderungen (Stetigkeit, und Fortsetzbarkeit mit dem Satz von Hahn-Banach) stellen, aber das wollen wir hier nicht vertiefen. Bestenfalls ist noch darauf hinzuweisen, daß (bei trivialem Beweis analog wie oben) der Faktorisierungssatz bei Verzicht auf Linearität auch in der folgenden abstrakten Form gilt:

Satz 12 *Sind $A : U \to V := A(U)$ und $B : U \to W$ Abbildungen mit der Eigenschaft*

$$B(x) = B(y) \text{ für alle } x, y \in U \text{ mit } A(x) = A(y),$$

so gibt es eine Abbildung $C : V \to W$ mit $B = C \circ A$.

Wir sollten aber noch den Fall von Ungleichungsnebenbedingungen der Form $g_j(x) \leq 0,\ 1 \leq j \leq \ell$ mit konvexen und differenzierbaren Funktionen g_j auf $I\!R^n$ ansehen, wobei wir aber der Einfachheit halber jetzt die affin-linearen Gleichungsnebenbedingungen weglassen. Wann ist ein Vektor $y \in I\!R^n$ eine zulässige Richtung? Es sollte

$$g_j(x + hy) \leq 0 \text{ für alle } j,\ 1 \leq j \leq \ell,\ h \in [0, h_0] \tag{9}$$

mit einem $h_0 > 0$ gelten. Für die j mit $g_j(x) < 0$ stellt das keine Bedingung an y, weil unter unseren Voraussetzungen die g_j stetig sind. Für die j mit $g_j(x) = 0$, die "aktiven" Restriktionen, muß dann

$$\lim_{h \searrow 0} \frac{1}{h} \left(g_j(x + hy) - g_j(x)\right) = g'_{j+}(x, y) = (\nabla g_j(x))y \leq 0$$

gefordert werden, aber das ist nur notwendig, nicht hinreichend für (9). Dieses Problem wird uns noch beschäftigen.

Wenn wir erst einmal nur mit den notwendigen Bedingungen für zulässige Richtungen y weitermachen, bekommen wir die notwendigen Variationsungleichungen $(\nabla f(x))y \geq 0$ für alle $y \in I\!R^n$ mit $(\nabla g_j(x))y \leq 0$ für alle $j,\ 1 \leq j \leq \ell$ mit $g_j(x) = 0$. Das kann man analog zu unserem obigen Vorgehen formalisieren zu einer Aussage der Form

$$By \geq 0 \text{ für alle } y \in I\!R^n \text{ mit } Gy \leq 0 \tag{10}$$

mit linearen Abbildungen

$$B\ :\ I\!R^n \to I\!R,\ G\ :\ I\!R^n \to I\!R^k,\ k \leq \ell.$$

Betrachtet man erst einmal den Teilraum $U = \ker G$ der y mit $Gy = 0$, so folgt aus der vorausgesetzten Linearität sofort

$$By = 0 \text{ für alle } y \in I\!R^n \text{ mit } Gy = 0$$

und es faktorisiert B über das Bild von G im Raum $I\!R^k$, wie wir oben schon gesehen haben. Es gibt also einen Vektor $u \in I\!R^k$ von "Lagrange-Multiplikatoren" mit

$$By = u^T Gy \text{ für alle } y \in I\!R^n.$$

Setzt man das in (10) ein, so folgt

$$By = u^T Gy \geq 0 \text{ für alle } y \in I\!R^n \text{ mit } \ Gy \leq 0.$$

Das ist sicher erfüllt, wenn wir zusätzlich $u \leq 0$ fordern, aber $u \leq 0$ ist nicht ohne weiteres als notwendige Bedingung an u zu erschließen. Obendrein kann man leider nicht erwarten, dass jedes u, das sich durch das Faktorisierungsargument ergibt, zwingend nichtpositive Komponenten hat.

Man kann aber durch nichttriviale Zusatzüberlegungen die Existenz eines nichtpositiven u mit der obigen Eigenschaft erschließen. Die obige Bedingung besagt nämlich, daß es keine zulässigen y gibt mit $-Gy \geq 0$ und $(u^T G)y < 0$. Das Farkas-Lemma (siehe Werner-Skript, S. 23, Lemma 1.6) liefert dann die Existenz eines $x \geq 0$ mit $-G^T x = G^T u$, und wir können unser u durch $-x \leq 0$ ersetzen.

Wir erweitern unser $u \leq 0$ noch durch Nullen auf die Komponenten j mit $g_j(x) < 0$ und erhalten die **Komplementaritätsbedingungen**

$$u_j g_j(x) = 0,\ 1 \leq j \leq \ell.$$

Wir könne das Ganze zu den notwendigen Optimalitätsbedingungen

$$\begin{aligned}(\nabla f)(x) + u^T(\nabla g)(x) + v^T(\nabla h)(x) &= 0\\ h(x) := Ax - b &= 0\\ g(x) &\leq 0\\ u &\geq 0\\ u_j g_j(x) &= 0\end{aligned}$$

zusammenfassen, wenn wir das Vorzeichen von u bei der Umsetzung auf die linke Seite berücksichtigen und (ohne Beweis) annehmen, daß sich Ungleichungsbedingungen und Gleichungsbedingungen additiv zusammenpacken lassen.

Bei diesem Zugang ist einigermaßen klar, wie die Lagrange-Multiplikatoren zustandekommen, und es verwundert nicht, daß man

$$L(x, u, v) := f(x) + u^T g(x) + v^T h(x)$$

die "Lagrange-Funktion" nennt.

2.3 Beispiele

2.3.1 Normen

Normen sind global definierte konvexe Funktionen, deshalb haben sie überall Gâteaux-Richtungsableitungen, die wieder konvexe Funktionen sind. Im Nullpunkt sind diese trivial:

Lemma 13 *Ist $\|.\|$ eine Norm auf einem Vektorraum V, so gilt (in naheliegender Notation)*

$$\|'_+(0, y) = \|y\| \text{ für alle } y \in V.$$

Außerhalb des Nullpunktes kann das nicht so simpel sein. Zuerst:

Lemma 14 *Ist $\|.\|$ eine "euklidische" Norm auf einem Vektorraum V, die aus einem Skalarprodukt $(.,.)$ durch $\|x\|^2 := (x, x)$ entsteht, so gilt (in naheliegender Notation)*

$$\|'_+(x, y) = \frac{(x, y)}{\|x\|} \text{ für alle } y \in V, x \in V \setminus \{0\}.$$

Das ist netterweise linear in y. Anders ist es bei

Lemma 15 *Es sei $\|.\| = \|.\|_\infty$ die Maximumsnorm auf $V = IR^n$. Dann gilt*

$$\|'_{\infty,+}(x, y) = \max_{i;:\, |x_i| = \|x\|_\infty} y_i \cdot \operatorname{sgn}(x_i) \text{ für alle } y \in IR^n, x \in IR^n \setminus \{0\}.$$

Machen wir das doch im Unendlichdimensionalen, etwa mit der Norm

$$\|x\|_\infty := \max_{a \le t \le b} |x(t)|$$

auf $V := C[a,b],\ a < b \in IR$. Erwartungsgemäß bekommt man

$$\|'_{\infty,+}(x,y) = \max_{t \in [a,b]\,:\,|x(t)|=\|x\|_\infty} y(t) \cdot \text{sgn}\,(x(t))$$

für alle $y \in C[a,b], x \in C[a,b] \setminus \{0\}$, was denn sonst? (Beweise als Tafeldemo oder Übung).

Wenn wir auf $V := C[a,b]$ die euklidische Norm $\|.\|_2$ über das Skalarprodukt

$$(x,y)_2 := \int_a^b x(t)y(t)dt \text{ für alle } x, y \in C[a,b]$$

definieren, können wir Lemma 14 direkt anwenden und bekommen

$$\|'_{2,+}(x,y) = \frac{\int_a^b x(t)y(t)dt}{\sqrt{\int_a^b x^2(t)dt}}$$

2.3.2 Variationsrechnung

Wir stellen uns das Problem, eine Kurve kürzester Bogenlänge im IR^2 zwischen den Punkten $(0,0)$ und $(1,1)$ zu finden. Eine Gummibandüberlegung zeigt, dass die Verbindungsgerade vermutlich die beste Lösung ist, mit der Bogenlänge $\sqrt{2}$. Allgemeinere und sehr viel interessantere Probleme dieser Art befassen sich mit "Geodätischen" auf Mannigfaltigkeiten. Beispielsweise weiss jeder Pilot und jeder Kapitän, dass die kürzesten Verbindungen auf der Kugel entlang Großkreisen verlaufen. Und Captain Kirk weiß seit Albert Einstein, dass sich Himmelskörper und Raumschiffe entlang von Geodätischen in der Raumzeit der allgemeinen Relativitätstheorie bewegen.

Zu minimieren ist in unserem simplen Fall

$$f(x) := \int_0^1 \sqrt{1 + x'^2(t)}dt$$

unter allen stetig differenzierbaren Funktionen x auf $[0,1]$ mit $x(0) = 0,\ x(1) = 1$. Wir haben also den unendlichdimensionalen Raum $V = C^1[0,1]$ und wollen Gâteaux-Richtungsableitungen von f in zulässige Richtungen y berechnen. Diese sind klar: sie sind die $y \in C^1[a,b]$ mit $y(0) = y(1) = 0$, bilden also einen Unterraum von $V = C^1[a,b]$ mit "Kodimension" 2, und sie hängen gar nicht vom "Aufpunkt" x ab.

Bevor wir uns zu Fuß auf den Weg machen, die Ableitungen über die Definition in diesem Spezialfall auszurechnen, sollten wir das Problem verallgemeinern und uns

$$f(x) := \int_a^b F(t, x(t), x'(t))dt$$

mit einer differenzierbaren Funktion $F = F(t, u, v)$ vornehmen. Es folgt

$$\begin{aligned} f(x+hy) &= \int_a^b F(t, x(t)+hy(t), x'(t)+hy'(t))dt \\ &= f(x) + \mathcal{O}(h^2) + \\ &\quad + \int_a^b \left(hy(t)\frac{\partial F}{\partial u}(t, x(t), x'(t)) + hy'(t)\frac{\partial F}{\partial v}(t, x(t), x'(t)) \right) dt \end{aligned}$$

durch Entwicklung, und man bekommt das Gâteaux-Differential

$$f'_+(x, y) = \int_a^b \left(y(t)\frac{\partial F}{\partial u}(t, x(t), x'(t)) + y'(t)\frac{\partial F}{\partial v}(t, x(t), x'(t)) \right) dt.$$

In einem lokalen Optimum x wird dann die Variationsungleichung

$$\int_a^b \left(y(t)\frac{\partial F}{\partial u}(t, x(t), x'(t)) + y'(t)\frac{\partial F}{\partial v}(t, x(t), x'(t)) \right) dt \geq 0$$

für alle zulässigen Richtungen y gelten. Wenn, wie in unserem Spezialfall, die Menge der zulässigen Richtungen der komplette lineare Unterraum der Funktionen y mit $y(a) = y(b) = 0$ ist, und wenn wir die Linearitat der Gâteaux-Ableitung ausnutzen, so wird aus der Variationsungleichung die Variations**gleichung**

$$\int_a^b \left(y(t)\frac{\partial F}{\partial u}(t, x(t), x'(t)) + y'(t)\frac{\partial F}{\partial v}(t, x(t), x'(t)) \right) dt = 0$$

für alle $y \in C^1[a, b]$ mit $y(a) = y(b) = 0$. Unter vorausgesetzter Differenzierbarkeit (die sich mit dem "Fundamentallemma der Variationsrechnung" aber auch erschließen läßt) kann man partiell integrieren und bekommt

$$\int_a^b y(t) \left(\frac{\partial F}{\partial u}(t, x(t), x'(t)) - \frac{d}{dt}\frac{\partial F}{\partial v}(t, x(t), x'(t)) \right) dt = 0$$

unter Ausnutzung der Randbedingungen $y(a) = y(b) = 0$. Ist der Klammerausdruck noch stetig, so kann die obige Gleichung nur dann für alle besagten y Null sein, wenn der Klammerausdruck selber Null ist, denn man kann winzige "Hütchenfunktionen" y dort ansetzen, wo der Klammerausdruck nicht Null ist und sein Vorzeichen nicht wechselt.

Es folgt dann die berühmte **Eulergleichung**

$$\frac{\partial F}{\partial u}(t, x(t), x'(t)) = \frac{d}{dt}\frac{\partial F}{\partial v}(t, x(t), x'(t)),\ F = F(t, u, v)$$

als notwendige Bedingung für ein lokales Optimum. Der Weg von einem Optimierungsproblem über eine Variations**un**gleichung zu einer Variationsgleichung und schließlich zu einer **Differentialgleichung für die Optimallösung** ist typisch für solche Aufgaben aus der klassischen **Variationsrechnung**. Die zulässigen Richtungen y werden von Physikern und Ingenieuren mit phantasievollen Namen wie "infinitesimale Verschiebungen" (in der Elastizitätstheorie und der Mechanik) belegt, sind aber nichts als zulässige Richtungen einer Optimierung. Die Eulergleichung ist eine Konsequenz von Satz 11 unter zusätzlichen Voraussetzungen.

In unserem Spezialfall haben wir $F(t, u, v) = \sqrt{1 + v^2}$ und bekommen die Eulergleichung

$$0 = \frac{d}{dt}\frac{x'(t)}{\sqrt{1 + x'^2(t)}}.$$

Also muß

$$\frac{x'(t)}{\sqrt{1 + x'^2(t)}}$$

und dann nach kurzer Rechnung auch x' konstant sein, und die Randbedingungen $x(0) = 0$ und $x(1) = 1$ lassen dann nur noch die Lösung $x(t) = t$ zu, die sich aus der notwendigen Bedingung für eine Lösung des Optimierungsproblems ergibt. Wir haben aber die Existenz einer Lösung und reichlich Differenzierbarkeit vorausgesetzt, so daß dieses Vorgehen nur zeigt, dass, wenn es eine hinreichend glatte Lösung gibt, diese notwendig die besagte Form hat.

2.3.3 Beispiel: Spline-Funktionen

Wir suchen eine mindestens zweimal stetig differenzierbare Funktion u auf $[a, b] \subset I\!R$, die das Integral

$$f(u) := \frac{1}{2}\int_a^b (u'')^2(t)dt$$

minimiert und dabei die Interpolations-Bedingungen

$$u(x_j) = y_j,\ 0 \leq j \leq n$$

mit vorgegebenen $x_j, y_j \in I\!R,\ 0 \leq j \leq n$ erfüllt, wobei die Stützstellen x_j eine Zerlegung

$$a = x_0 < x_1 < \ldots < x_{n-1} < x_n = b$$

des Intervalls $[a, b]$ bilden. Das ist eine konvexe Optimierungsaufgabe im unendlichdimensionalen Raum $C^2[a, b]$ mit affin-linearen Nebenbedingungen. Zulässige Richtungen sind Funktionen $v \in C^2[a, b]$ mit $v(x_j) = 0,\ 0 \leq j \leq n$ und bilden also einen linearen Unterraum V von $C^2[a, b]$. Die Gâteaux-Ableitung von f ergibt sich als

$$f'_+(u, v) = \int_a^b u''(t)v''(t)dt$$

nach einfacher Rechnung. Eine Funktion $u \in C^2[a, b]$ ist genau dann Optimallösung, wenn $f'_+(u, v) \geq 0$ für alle zulässigen Richtungen v gilt. Wegen Linearität in v ist das äquivalent zu $f'_+(u, v) = 0$ für alle zulässigen Richtungen v. Unter vorläufiger Annahme von reichlich Differenzierbarkeit in den Teilintervallen $(x_{j-1}, x_j),\ 1 \leq j \leq n$ kann man das auswerten:

$$\begin{aligned}
0 &= f'_+(u, v) \\
&= \int_a^b u''(t)v''(t)dt \\
&= \sum_{j=1}^{n} \int_{x_{j-1}}^{x_j} u''(t)v''(t)dt \\
&= \sum_{j=1}^{n} \left(- \int_{x_{j-1}}^{x_j} u'''(t)v'(t)dt + [u'' \cdot v']_{x_{j-1}}^{x_j} \right) \\
&= [u'' \cdot v']_a^b + \sum_{j=1}^{n} \left(\int_{x_{j-1}}^{x_j} u''''(t)v(t)dt + [u''' \cdot v]_{x_{j-1}}^{x_j} \right) \\
&= [u'' \cdot v']_a^b + \sum_{j=1}^{n} \int_{x_{j-1}}^{x_j} u''''(t)v(t)dt
\end{aligned}$$

Wie kann man das erfüllen durch eine geeignete Funktion $u \in C^2[a, b]$?

Wenn man u aus Stücken zusammenbaut, die auf jedem Teilintervall $(x_{j-1}, x_j),\ 1 \leq j \leq n$ ein Polynom dritten Grades sind, verschwinden alle lokalen Integrale, und wenn man auch noch $u''(a) = u''(b) = 0$ verlangt, ist die obige Gleichung erfüllt. Mit Argumenten, die nicht in eine Optimierungsvorlesung gehören, kann man zeigen, dass es immer genau eine Funktion $u \in C^2[a, b]$ gibt, die allen interpolationsbedingungen genügt, in jedem Teilintervall ein Polynom dritten Grades ist und in den Randintervallen affin-linear ist. Man konstruiert so eine Funktion durch Lösen eines nichtsingulären linearen Gleichungssystems mit einer tridiagonalen Koeffizientenmatrix. Funktionen dieser Art heißen *kubische Splines*, und sie sind in der Numerischen Mathematik sehr wichtig.

Es ist für solche Situationen typisch, dass man die konvexe Optimierungtheorie zunächst nur heuristisch anwendet, um die notwendigen Optimalitätsbedingungen auszuwerten, obwohl man keineswegs weiß, ob eine Lösung existiert. Wenn man dann auf ganz anderem Wege beweisen kann, daß die notwendigen Bedingungen erfüllbar sind, benutzt man, daß diese ja auch hinreichend sind, und ist fertig.

Es ist nach dem obigen Schema relativ einfach zu zeigen, daß ein stetiger stückweise affin-linearer Polygonzug u das Minimum von

$$f(u) := \frac{1}{2}\int_a^b (u')^2(t)dt$$

unter den Interpolations-Bedingungen

$$u(x_j) = y_j,\ 0 \leq j \leq n$$

realisiert (das ist die "connect-the-dots"-Interpolation). Man kann allerdings dabei nicht auf $C^1[a,b]$ arbeiten, aber findige Leser werden herausbekommen, wie man das Ganze sauber ausführen kann.

2.4 Quadratisch optimierende Lernalgorithmen

2.4.1 Optimale Modelle

Wir gehen zurück zur Lerntheorie aus Abschnitt 1.2 und benutzen die *feature map* $\phi : X \to \mathcal{F}$ und den *Kern* $K : X \times X \to IR$ mit

$$K(x,y) := \langle \phi(x), \phi(y) \rangle \text{ für alle } x, y \in X.$$

Wir definieren den Raum

$$\mathcal{K} := \text{span}\ \{K(\cdot, x) : x \in X\}$$

von Funktionen auf X, weil wir sonst nichts haben, was wir als Funktion auf X benutzen können. Auf diesem Raum gibt es eine Bilinearform, die durch Fortsetzung der Definition

$$(K(\cdot,x), K(\cdot,y))_{\mathcal{K}} := K(x,y) \text{ für alle } x, y \in X$$

auf beliebige Linearkombinationen ensteht:

$$\left(\sum_j \alpha_j K(\cdot, x_j), \sum_k \beta_k K(\cdot, y_k)\right)_{\mathcal{K}} = \sum_j \sum_k \alpha_j \beta_k K(x_j, y_k).$$

Sie ist positiv definit und damit ein Skalarprodukt, wenn der Kern die folgende Eigenschaft hat:

Definition 16 *Ein Kern $K : X \times X \to IR$ ist positiv definit, wenn für beliebige endliche Teilmengen $X_n := \{x_1, \ldots, x_n\}$ von X die $n \times n$ Matrix mit Einträgen $K(x_j, x_k),\ 1 \leq j, k \leq n$ positiv definit ist.*

Das ist gleichbedeutend damit, dass die Funktionen $K(\cdot, x_j)$ für verschiedene x_j immer linear unabhängig sind.

In einer weiter fortgeschrittenen Veranstaltung würde man jetzt zur Hilbertraum-Vervollständigung von $\mathcal{K}$ übergehen, aber das wollen wir hier unterlassen. Wir spezialisieren aber die obige Gleichung zu

$$\left(\sum_j \alpha_j K(\cdot, x_j), K(\cdot, y) \right)_{\mathcal{K}} = \sum_j \alpha_j K(x_j, y),$$

was dann für beliebige Funktionen $g \in \mathcal{K}$ zur *Reproduktionsgleichung*

$$(g, K(\cdot, y))_{\mathcal{K}} = g(y) \text{ für alle } y \in X,\ g \in \mathcal{K}$$

wird. Kerne mit so einer Eigenschaft nennt man *reproduzierend* für einen Raum $\mathcal{K}$ von Funktionen auf X.

Wir gehen wieder davon aus, daß wir m Trainingsdaten $(x_j, y_j) \in X \times IR$ mit $y_j \approx g(x_j)$ für eine zu "lernende" Funktion g haben. Wir werden jetzt unter allen Funktionen $g \in \mathcal{K}$, die eine exakte Reproduktion $y_j = g(x_j),\ 1 \leq j \leq m$ leisten, eine optimale heraussuchen, indem wir eine mit minimaler Norm $\|.\|_{\mathcal{K}}$ berechnen. Wir landen dabei punktgenau bei der damals "vom Himmel gefallenen" Gleichung (5)

Satz 17 *Es sei K ein positiv definiter Kern auf X. Dann hat das quadratische Minimierungsproblem*

$$\begin{aligned} \|g\|_{\mathcal{K}}^2 &= \min_{g \in \mathcal{K}} \\ g(x_j) &= y_j,\ 1 \leq j \leq m \end{aligned}$$

eine eindeutige Lösung der Form

$$g^*(x) := \sum_{j=1}^{m} \alpha_j K(x, x_j),\ x \in X,$$

die sich durch Lösen des Gleichungssystems

$$\sum_{j=1}^{m} \alpha_j K(x_k, x_j) = y_k,\ 1 \leq k \leq m$$

mit symmetrischer und positiv definiter Koeffizientenmatrix berechnen läßt.

Der **Beweis** ist auf verschiedene Weisen möglich. Da der Raum $\mathcal{K}$ nicht notwendig endlichdimensional ist, kann man nicht ohne weiteres die Existenz einer Lösung erschließen. Aber wir haben einen Kandidaten, und wir können Satz 10 anwenden. Die Funktion $f(g) := \|g\|_{\mathcal{K}}^2$ hat die Gâteaux-Ableitung $2(g^*, v)_{\mathcal{K}}$ in g^* in jede zulässige Richtung v, und diese Richtungen bestehen aus den $v \in \mathcal{K}$ mit $v(x_j) = 0,\ 1 \le j \le m$. Der Raum dieser Richtungen ist linear, und so wird aus der notwendigen und hinreichenden Variations**ungleichung** des Satzes 10 die Variations**gleichung**

$$(g^*, v)_{\mathcal{K}} \quad = \quad 0 \text{ für alle } v \in \mathcal{K},\ v(x_j) = 0,\ 1 \le j \le m.$$

Setzen wir unser spezielles g^* ein und verwenden die Reproduktiongleichung, so folgt

$$\left(\sum_{j=1}^{m} \alpha_j K(\cdot, x_j), v \right)_{\mathcal{K}} = \sum_{j=1}^{m} \alpha_j v(x_j) = 0,$$

d.h. g^* erfüllt die notwendige und hinreichende Optimalitätsbedingung. □

Wer die Form der Optimallösung g^* nicht "raten" mag, kann sie auch erschließen. Denn wenn g^* eine Funktion aus $\mathcal{K}$ ist, die der Variationsgleichung genügt, so kann man die *Datenabbildung* $T \ : \ \mathcal{K} \to I\!R^m$ mit $T(u) := (u(x_1), \ldots, u(x_m))$ definieren und benutzen, dass

$$(g^*, v)_{\mathcal{K}} = 0 \text{ für alle } v \text{ mit } T(v) = 0$$

gilt. Dann faktorisiert (siehe Abschnitt 2.2) unter schwachen, hier erfüllten Voraussetzungen das lineare Funktional $v \mapsto (g^*, v)_{\mathcal{K}}$ über das Bild von T, d.h. es gibt einen Vektor $\alpha \in I\!R^n$ mit

$$(g^*, v)_{\mathcal{K}} = \alpha^T T v = \sum_{j=1}^{m} \alpha_j v(x_j) = \left(\sum_{j=1}^{m} \alpha_j K(\cdot, x_j), v \right)_{\mathcal{K}} \text{ für alle } v \in \mathcal{K},$$

und weil dies eine Variationsgleichung für **alle** $v \in \mathcal{K}$ ist, muss g^* die behauptete Form haben.

2.4.2 Inexakte Reproduktion

Es macht wenig Sinn, beim obigen Vorgehen auf exakter Reproduktion aller Trainingsdaten zu bestehen, weil dann für jedes neue Trainingsdatum eine neue Rechnung nötig wäre und das Lernergebnis von **allen** Trainingsdaten sehr sensibel abhängig wäre. Das "Relaxieren" der Bedingungen $y_j = g(x_j)$

kann auf verschiedene Weise geschehen und mit der Zielfunktion $\|g\|_{\mathcal{K}}^2$ verbunden werden. Eine typische Variante ist, die linearen Nebenbedingungen

$$-\epsilon \leq y_k - \sum_{j=1}^{m} \alpha_j K(x_k, x_j), \leq \epsilon,\ 1 \leq k \leq m$$

zu fordern und dann die quadratische Zielfunktion

$$\frac{1}{2}\|g\|_{\mathcal{K}}^2 + C\epsilon = \frac{1}{2}\sum_{j,k=1}^{m} \alpha_j \alpha_k K(x_k, x_j) + C\epsilon \tag{11}$$

zu minimieren, wobei die positive Konstante C es gestattet, entweder auf gute Reproduktion der Einzeldaten oder auf gute "Generalisierung" des Modells zu setzen.

Diese quadratische Aufgabe mit affin-linearen Ungleichungsnebenbedingungen wollen wir etwas genauer analysieren. Die Variablen sind ϵ und $\alpha_1, \ldots, \alpha_m$, und die Lagrangefunktion bekommt die Form

$$L(\alpha, \epsilon, \lambda, \mu) = \frac{1}{2}\alpha^T Q\alpha + C\epsilon + \lambda^T(-\epsilon\mathbf{1} + Q\alpha - y) + \mu^T(-\epsilon\mathbf{1} - Q\alpha + y)$$

mit der "Kernmatrix" Q aus den $K(x_j, x_k)$. Die Lagrange-Multiplikatoren-Vektoren λ und μ sind nichtnegativ und aus dem $I\!R^m$ zu nehmen.

Wir gehen direkt auf die Idealsituation der primalen und dualen Lösbarkeit zu. Nach den bekannten Sätzen ist das Problem lösbar, weil es zulässig ist und die Zielfunktion nach unten beschränkt ist. Ferner ist durch die oben diskutierte exakte Rekonstruktionsfunktion g mit $\epsilon = 0$ auch die Slater-Bedingung erfüllt, so daß der verschärfte starke Dualitätssatz gilt. Also existieren optimale Lösungen $\alpha^*,\ \epsilon^* \geq 0,\ \lambda^* \geq 0,\ \mu^* \geq 0$ mit

$$\begin{aligned}(-\epsilon^*\mathbf{1} + Q\alpha^* - y)_j\lambda_j^* &= 0,\ 1 \leq j \leq m\\ (-\epsilon^*\mathbf{1} - Q\alpha^* + y)_j\mu_j^* &= 0,\ 1 \leq j \leq m\end{aligned}$$

d.h.

$$\begin{aligned}\text{aus } \lambda_j^* > 0 \text{ folgt } (Q\alpha^* - y)_j &= \epsilon^*\\ \text{aus } \mu_j^* > 0 \text{ folgt } (Q\alpha^* - y)_j &= -\epsilon^*\end{aligned}$$

und wir sind wieder bei unserer bekannten Alternationseigenschaft und bei den "support" Vektoren. Differenzieren wir die Lagrangefunktion im Optimalpunkt nach α, so folgt $Q\alpha^* = Q(\mu^* - \lambda^*)$, also $\alpha^* = \mu^* - \lambda^*$. Der optimale Koeffizientenvektor α^* hat also nur so viele von Null verschiedene Komponenten wie es "aktive Restriktionen" gibt, und das Vorzeichen der Komponenten ist durch das Vorzeichen des "Fehlers" bestimmt. Trainingsdaten, die

nicht zu aktiven Restriktionen im Optimalpunkt führen, kommen in der Optimallösung nicht vor und sind bei a-posteriori-Betrachtung irrelevant. Das ist der wichtigste Vorteil von Lernalgorithmen dieser Art.

Wir haben aber noch nach ϵ zu differenzieren. Im Falle $\epsilon^* > 0$ kann es keine Indizes j geben, für die λ_j^* und μ_j^* beide positiv sind. Deshalb folgt dann aus $\alpha^* = \mu^* - \lambda^*$ auch $|\alpha_j^*| = \mu_j^* + \lambda_j^*$. Wir bekommen damit

$$C = \mathbf{1}^T(\mu^* + \lambda^*) = \|\alpha^*\|_1.$$

als Ableitung der Lagrangefunktion nach ϵ, was zeigt, daß die Kontrolle von C auch die Kontrolle über die Größe der Koeffizienten im Optimalpunkt erlaubt.

Es ist lehrreich, das Dualproblem auszurechnen, aber das lassen wir als Übungsaufgabe.

Natürlich kann man das weiter vorn stehende Beispiel des "Lernens" eines Kreises oder einer anderen geometrischen Figur mit den Methoden dieses Abschnittes behandeln, indem man die damalige Zielfunktion ϵ durch (11) ersetzt und die in (6) auftretenden k Punkte y_i durch alle m Punkte x_j ersetzt. Die Selektion einer "aktiven" Teilmenge von "support vectors" geschieht nun automatisch durch die quadratische Optimierung mit linearen Nebenbedingungen. Es sind nur so viele Koeffizienten der Optimallösung von Null verschieden, wie es aktive Restriktionen gibt.

Das folgende MATLAB-Programm setzt diesen Ansatz um. Es ist allerdings nicht identisch mit dem früheren Programm, denn es kann beliebige sternförmige Figuren in $[-1, 1]^2$ lernen.

```
clear all;
np=75; % Anzahl der Trainingsdaten
[X Y]=meshgrid(-1:0.05:1); % ein Gitter zwecks feature vectors
XX=X(:);
YY=Y(:);
nd=length(XX) % das wird später die Länge der feature vectors
randx=2*rand(np,1)-1; % hier die Trainingsdaten
randy=2*rand(np,1)-1;
testval=randx.^2+randy.^2; % aktuelle Radienquadrate
[theta rho]=cart2pol(randx,randy);
sollrad=radi(theta);
xset=find(testval<=sollrad.^2);
```

```
[kreisx kreisy]=pol2cart(2*pi*[0:0.01:1],radi(2*pi*[0:0.01:1]));
val=ones(np,1);  % und wir setzen die Trainingswerte
val(xset,1)=-1;
posset=find(val>0);  % zum Plotten splitten wir die Daten
negset=find(val<0);
subplot(3,1,1)
plot(randx(posset),randy(posset),'+',kreisx,kreisy) % und plotten sie
axis([-1,1,-1,1])
hold on
plot(randx(negset),randy(negset),'o')
title('Trainingsdaten')
fv=zeros(np,nd);     % Matrix der feature vectors
for i=1:nd    % wir nehmen die Distanzwerte zum Gitter
    fv(:,i)=max(abs(randx(:,1)-XX(i)),abs(randy(:,1)-YY(i))); %
%   Maximumsnorm
    % fv(:,i)=sqrt((randx(:,1)-XX(i)).^2.+(randy(:,1)-YY(i)).^2);
    % oder 2-Norm
end
Kmat=fv*fv'; % das wird die Kernmatrix
c=1
[x fval]=mylearner(Kmat,val,c) % und rein ins Kernelproblem
neval=250; % Anzahl der Testpunkte
npx=2*rand(neval,1)-1;
npy=2*rand(neval,1)-1;
% neval=np;
% npx=randx;
% npy=randy;
fp=zeros(neval,nd); % deren feature vectors
for i=1:nd
    fp(:,i)=max(abs(npx(:,1)-XX(i)),abs(npy(:,1)-YY(i)));
end
zp=fp*fv'*x; % und deren Wert als Vorhersage
posset=find(zp>0); % zum Plotten brauchen wir die Entscheidungen...
negset=find(zp<0);
subplot(3,1,2)
plot(npx(posset),npy(posset),'+',kreisx,kreisy)
axis([-1,1,-1,1])
hold on
plot(npx(negset),npy(negset),'o')
hold on
title('Testdaten')
```

```
resid=abs(Kmat*x-val);
xset=find(resid>fval-0.0001);
posxset=find(val(xset)>0);
negxset=find(val(xset)<0);
subplot(3,1,3)
plot(randx(xset(posxset)),randy(xset(posxset)),'+',kreisx,kreisy) % und plotten
axis([-1,1,-1,1])
hold on
plot(randx(xset(negxset)),randy(xset(negxset)),'o')
title('Support-Vektoren')
```

Die zu lernende Figur wird spezifiziert durch eine Polarkoodinatenfunktion wie

```
function val=radi(winkel)
val=sqrt(0.3)*(1-0.5*cos(4.*winkel)).*ones(size(winkel));
```

Ferner wird auf eine Funktion der Form

```
[alpha  wert]=mylearner(Q,y,C)
```

zurückgegriffen, die als Übungsaufgabe gestellt wird (sie wird später hier eingebaut). Diese Funktion arbeitet genau so wie im obigen Text beschrieben. Sie erwartet eine $m \times m$ Kernmatrix `Q`, einen Datenvektor `y` mit m Komponenten und das Gewicht `C`. Dann gibt sie den optimalen Koeffizientenvektor `alpha` und den finalen Zielfunktionswert zurück.

Eine typische Ausgabe ist in Abbildung 4 zu sehen. Es ist erstaunlich, wie wenig support-Vektoren nötig sind.

2.4.3 Klassifikation durch Trennung

Wir wollen uns aber auch noch einmal um Aschenputtel kümmern. Inzwischen können wir quadratisch optimieren, und wir wollen uns von der Voraussetzung der Trennbarkeit der gegebenen Trainingsdaten befreien. Wir wollen wieder einen "trennenden Streifen" finden, dessen Breite wir maximieren wollen, aber wir wollen zulassen, dass die Daten gar nicht trennbar sind. Deshalb "bestrafen" wir nicht trennbare Trainingsdaten auf geeignete Weise, und zwar durch Aufnahme in die Zielfunktion. Weil der Rand des trennenden Streifens "aufgeweicht" wird, spricht man von "soft margin classifiers".

Die Bezeichnungen seien wie im Abschnitt 1.2.5. Statt der Restriktionen (7) verwenden wir

$$M^+x + \beta\mathbf{1} + y^+ \geq \epsilon\mathbf{1}, \quad -\epsilon\mathbf{1} + y^- \geq M^-x + \beta\mathbf{1}$$

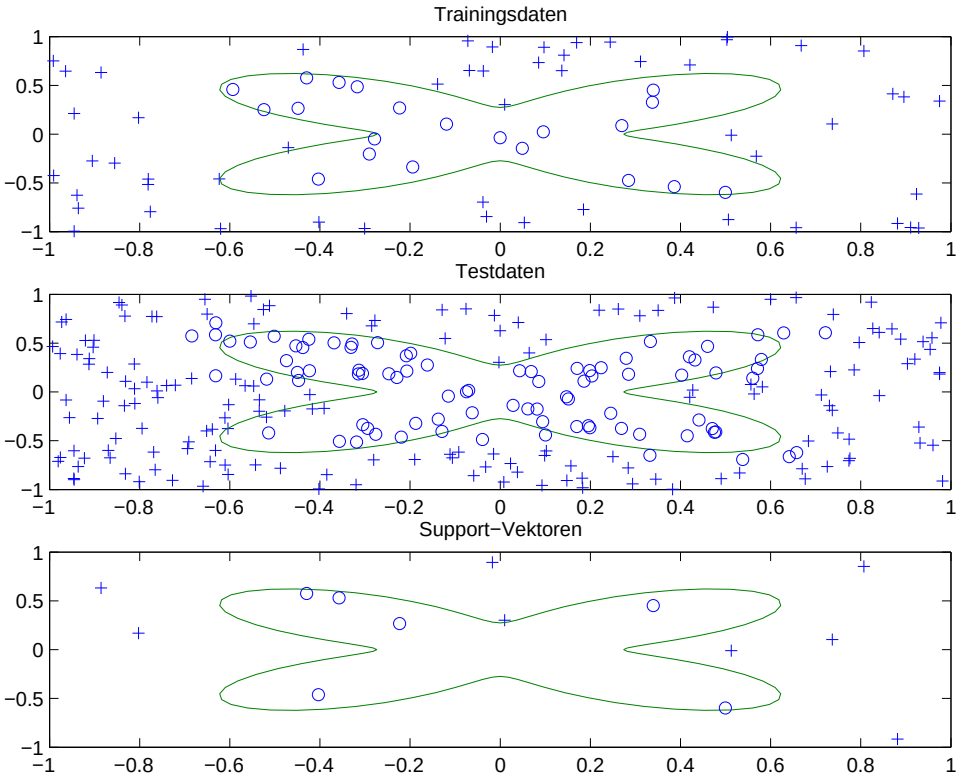

Abbildung 4: Figurenlernen mit Kernen

mit nichtnegativen Vektoren y^+, y^- von Schlupfvariablen, die das Nichterfülltsein der ursprünglichen Trennung "messen". Diese Vektoren müssen wir klein halten, und wir wollen gleichzeitig die (nunmehr euklidisch gemessene) Streifenbreite $2\epsilon/\|x\|_2$ maximieren. Dazu renormieren wir die obigen Ungleichungen auf $\epsilon = 1$ zu

$$M^+x + \beta\mathbf{1} + y^+ \geq \mathbf{1}, \quad -\mathbf{1} + y^- \geq M^-x + \beta\mathbf{1}$$

und minimieren $\|x\|_2^2$ stattdessen. Offen bleibt noch, wie wir große y^+, y^- bestrafen wollen. Das kann man durch eine gewichtete quadratische Zielfunktion

$$\frac{1}{2}\|x\|_2^2 + C(\|y^+\|_2^2 + \|y^-\|_2^2)$$

erreichen. Das folgende Programm ist eine Adaptation des früheren Aschenputtel-Programms:

```
clear all;
np=25 % Anzahl der guten Punkte
```

```
nn=25 % Anzahl der bösen Punkte
r=[0.2 0.5]; % Richtungsvektor der idealen Hyperebene
nor=[-0.5 0.2] % Normale dazu
bs=[0 0]; % Aufpunkt für Strahl auf Hyperebene
% wir gehen zufällig vor und berechnen Punkte
% entlang der Geraden und gleichzeitig links und rechts, mit
overlap=0.2
for ip=1:np
    Mp(ip,:)=bs+rand(1,1)*r+0.2*(rand(1,1)-overlap)*nor;
    Mn(ip,:)=bs+rand(1,1)*r-0.2*(rand(1,1)-overlap)*nor;
end
% plot(Mp(:,1),Mp(:,2),'+',Mn(:,1),Mn(:,2),'o')
% figure(2)
% So, jetzt bauen wir das Aschenputtel-Problem auf
% das wird das Gewicht
c=1.0e5
A=[-Mp  -ones(np,1) -eye(np) zeros(np,np);...
    Mn   ones(np,1) zeros(np,np) -eye(np)];
b=[-ones(np,1);-ones(np,1)];
p=zeros(2*np+3,1);
Q=c*eye(2*np+3);
Q(1:2,1:2)=eye(2);
Q(3,3)=0.0001;
lb=zeros(2*np+3,1);
ub=[];
lb(1:3,1)=-1.0e12;
[x fval]=quadprog(Q,p,A,b,[],[],lb,ub)
% Wir wollen die trennende Ebene malen
tt=-0:0.01:0.2; % das werden die x-Werte
% und es kommen die umgerechneten y-Werte
y0=(       -x(3,1)-x(1,1)*tt)/x(2,1);
% und die malen wir
plot(tt,y0)
hold on
% mit den gegebenen Daten
plot(Mp(:,1),Mp(:,2),'+',Mn(:,1),Mn(:,2),'o')
```

Eine typische Ausgabe ist in Abbildung 5 zu sehen. Man mache sich klar, dass unsere Programmierung des Aschenputtelproblems ziemlich unrealistisch ist, weil wir einen nur zweidimensionalen feature space benutzen. Die allgemeinere Technik mit Kernen, die durch vernünftige feature maps definiert sind,

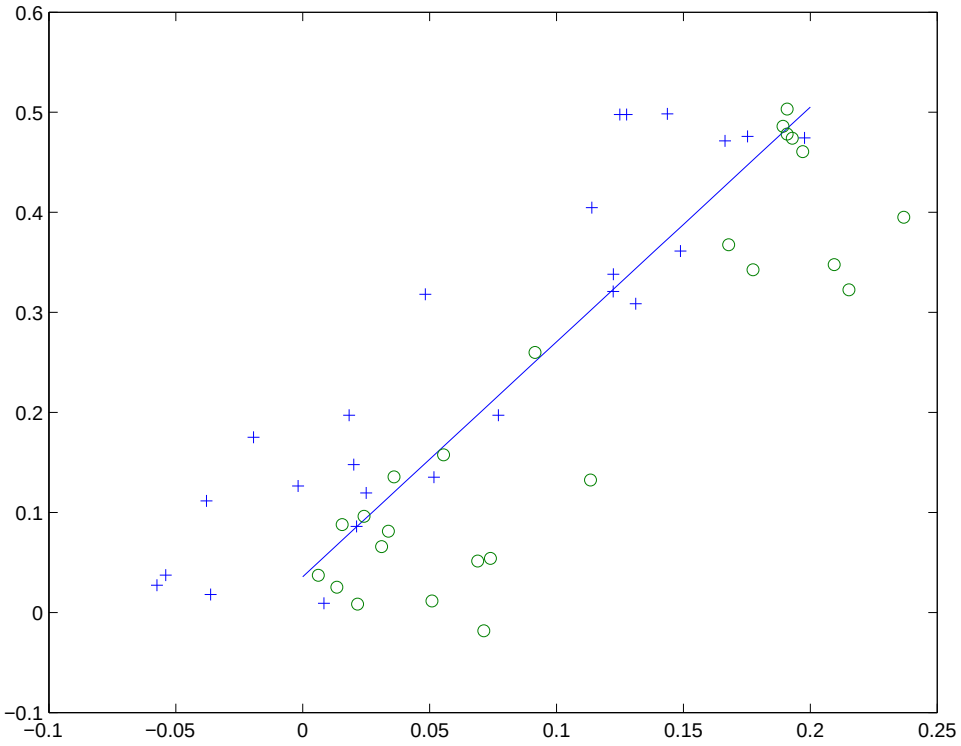

Abbildung 5: Aschenputtelproblem bei nicht trennbaren Daten

ist wesentlich leistungsfähiger.

3 Nichtlineare Optimierung

3.1 Rechentechnik

Wir fügen hier noch etwas an, was für die Rechenpraxis wichtig ist, aber in den Skripten nicht explizit steht. Wir gehen von einer nichtlinearen Optimierungsaufgabe

$$\begin{aligned} f(x) &= Min! \\ x &\in IR^n \\ g_i(x) &\leq 0,\ 1 \leq i \leq \ell \\ h_j(x) &= 0,\ 1 \leq j \leq m \end{aligned}$$

mit stetig differenzierbaren reellwertigen Funktionen f, g_i, h_j auf IR^n aus, und schließen den konvexen Fall ein, wobei wir aber auf die zusätzliche konvexe Menge C des Werner-Skripts verzichten.

Die Lagrange-Funktion ist

$$L(x,u,v) := f(x) + u^T g(x) + v^T h(x),\ x \in I\!R^n,\ u \in I\!R^\ell,\ v \in I\!R^m,$$

wenn man wie üblich die Funktionen $g_i,\ h_j$ zu Vektoren zusammenfaßt.

In der Praxis schert man sich wenig um die genauen Voraussetzungen, unter denen der Kuhn-Tucker-Satz gilt. Man wendet bei halbwegs komplizierten Problemen irgendwelche numerischen Standardverfahren an, die am Schluß der Vorlesung skizziert werden. Bei einfachen Problemen, bei denen man eine "Papier-und-Bleistift"-Lösung versuchen kann, setzt man die notwendigen Bedingungen 1. Ordnung als nichtlineares Gleichungssystem an. Das liefert

$$\begin{array}{rcll} \nabla f(x) + u^T \nabla g(x) + v^T \nabla h(x) & = & 0, \quad n & \text{Gleichungen} \\ h(x) & = & 0, \quad m & \text{Gleichungen} \\ u_i g_i(x) & = & 0, \quad \ell & \text{Gleichungen} \\ u_i & \geq & 0, \quad \ell & \text{Ungleichungen} \\ g_i(x) & \leq & 0, \quad \ell & \text{Ungleichungen} \end{array}$$

bei $n + \ell + m$ Unbekannten $x,\ u,\ v$. Mit etwas Glück kann man aus den ersten n Gleichungen x als Funktion von u und v ausrechnen. Das klappt z.B. immer dann, wenn ein quadratisches Optimierungsproblem mit positiv definiten quadratischen Formen vorliegt und die Menge C fehlt. Denn dann ist die Lösung von

$$\min_{x \in I\!R^n} L(x,u,v) = \min_{x \in I\!R^n} f(x) + u^T g(x) + v^T h(x)$$

bei festen u, v eine unrestringierte quadratische Optimierungsaufgabe mit positiv definiter quadratischer Form, die immer eine eindeutige Lösung $x(u,v)$ hat, die man durch Lösen von $\nabla f(x) + u^T \nabla g(x) + v^T \nabla h(x) = 0$ ausrechnen kann. Gleichzeitig liefert das im konvexen Fall die Zielfunktion des dualen Problems als $\Phi(u,v) = L(x(u,v),u,v)$. Wenn man $x(u,v)$ in das zweite System einsetzt, bekommt man $h(x(u,v)) = 0$ und kann mit etwas Glück nach v auflösen, z.B. dann, wenn h affin-linear ist und Vollrang hat (siehe Slater-Bedingung im konvexen Fall). Das liefert v als Funktion von u, und es bleiben die restlichen, leider nichtlinearen und mit Vorzeichenproblemen etwas überfrachteten Bedingungen an u und $g(x(u,v(u)))$, bei denen man nochmal reichlich Glück braucht, um durchzukommen. Natürlich wird man diese Bedingungen aufspalten in "aktive" der Form $g_j(x) = 0,\ u_j \geq 0$ und "inaktive" mit $g_j(x) < 0,\ u_j = 0$. Hat man k aktive und $\ell - k$ inaktive Bedingungen zu erwarten, so reduziert sich das System der ℓ Gleichungen $u_j g_j(x(u,v(u))) = 0,\ 1 \leq j \leq \ell$ auf k Gleichungen und k Unbekannte, aber es ist nicht immer einfach, die aktiven Restriktionen festzustellen.

Natürlich ist das obige Vorgehen im allgemeinen viel zu hemdsärmelig, um sicher zu funktionieren. Selbst wenn man vorzeichenkorrekte Lösungen des nichtlinearen Gleichungs/Ungleichungssystems finden kann, weiß man nicht, ob sie das ursprüngliche Problem lösen, weil man ja nur die notwendigen Bedingungen hineingesteckt hat. Und in allen Fällen mit vielen lokalen Minima wird das System notwendigerweise viele Lösungen haben, obwohl es aus $n + \ell + m$ Gleichungen (plus 2ℓ Ungleichungen) mit $n + \ell + m$ Unbekannten besteht. Beispielsweise berechnet es im allgemeinen nichtlinearen Fall ohne Ungleichungsnebenbedingungen natürlich auch die lokalen Maxima. Aber zumindestens weiß man, dass, wenn es ein Minimum gibt, dieses unter den Lösungen ist, und man kann bei Vorliegen von nur wenigen Kandidaten einfach die Zielfunktion auswerten, um das Minimum herauszupicken.

Man sollte so etwas auf jeden Fall einmal an Hand einer kleinen Übungsaufgabe durchgerechnet haben.