Methodische und Praktische Grundlagen der Informatik 4

Übungsblatt 6

mpgi4@cg.tu-berlin.de

WiSe 2013/2014

Teil 1: Ausgleichsrechnung

Sei $A \in \mathbb{R}^{m \times n}$ eine Matrix und $b \in \mathbb{R}^m$ ein Vektor. In der Vorlesung wurde gezeigt, dass ein $x \in \mathbb{R}^n$ genau dann die Länge des Fehlers $e = Ax - b$ minimiert, wenn der e orthogonal zum Spaltenraum $W = \{Az : z \in \mathbb{R}^n\}$ ist. Oder mittels Formeln ausgedrückt:

$$\langle Az, e \rangle = \langle Az, Ax - b \rangle = 0 \quad \text{für alle } z \in \mathbb{R}^n \tag{1}$$

Nutzen wir die Eigenschaft des Skalarproduktes, dass $\langle Ax, y \rangle = (Ax)^T y = x^T (A^T y) = \langle x, A^T y \rangle$ ist, so ergibt sich folgende äquivalente Darstellung:

$$\langle z, A^T(Ax - b) \rangle = 0 \quad \text{für alle } z \in \mathbb{R}^n \tag{2}$$

Stellen wir nun $z = z_1 e_1 + \ldots + z_n e_n = \sum_{i=1}^n z_i e_i$ als Linearkombination der Einheitsvektoren dar, so gilt aufgrund der Bilinearität des Skalarproduktes $\langle z, x \rangle = \langle \sum_{i=1}^n z_i e_i, x \rangle = \sum_{i=1}^n z_i \langle e_i, x \rangle$ und wir sehen, dass $\langle z, x \rangle$ genau dann für alle $z \in \mathbb{R}^n$ Null ist wenn $\langle e_i, x \rangle$ für alle Einheitsvektoren Null ist. Die Bedingung in Gleichung (2) ist somit äquivalent zu:

$$\langle e_i, A^T(Ax - b) \rangle = 0 \quad \text{für } i = 1, \ldots, n \tag{3}$$

Das bedeutet aber nichts anderes als, dass die Komponenten des Vektors $A^T(Ax - b)$ alle Null sein müssen. Dies ist wiederum gleichbedeutend damit, dass $A^T(Ax - b)$ gleich dem Nullvektor ist. Ein Vektor $x \in \mathbb{R}^n$ minimiert also genau dann $\|e\| = \|Ax - b\|$, wenn $A^T Ax - A^T b = 0$ ist. Um ein solches x zu finden, muss also das folgende Gleichungsssystem gelöst werden:

$$A^T Ax = A^T b \tag{4}$$

Teil 2: Ausgleichsrechnung und SVD

Sei $A \in \mathbb{R}^{m \times n}$ eine Matrix und $b \in \mathbb{R}^m$ ein Vektor. Wir wollen die Lösung des Ausgleichproblems $Ax - b$ unter Verwendung der SVD von A bestimmen.

Nehmen wir dazu zunächst an, dass Σ eine Diagonalmatrix mit absteigend sortierten Diagonalelementen $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n$ ist. Dann haben wir:

$$\Sigma x - b = \begin{pmatrix} \sigma_1 & & 0 \\ & \ddots & \\ 0 & & \sigma_n \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} - \begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix} = \begin{pmatrix} \sigma_1 x_1 - b_1 \\ \vdots \\ \sigma_n x_n - b_n \end{pmatrix} \tag{5}$$

Bezeichnet r den Rang von Σ, so ist $\sigma_i = 0$ für $i = r+1, \ldots, n$. Wir erhalten also:

$$\|\Sigma x - b\|^2 = \sum_{i=1}^n \left(\sigma_i x_i - b_i\right)^2 \tag{6a}$$

$$= \sum_{i=1}^r \left(\sigma_i x_i - b_i\right)^2 + \sum_{i=r+1}^n b_i^2 \tag{6b}$$

Der erste Term ist genau dann minimal, wenn $x_i = \frac{b_i}{\sigma_i}$ für $i = 1, \ldots, r$ ist. Der zweite Term in Gleichung (6b) ist konstant und spielt daher bei einer Minimierung keine Rolle. Die Komponenten x_i können also für $i = r+1, \ldots, n$ beliebig gewählt werden:

$$x = \left(\frac{b_1}{\sigma_1}, \ldots, \frac{b_r}{\sigma_r}, \xi_{r+1}, \ldots, \xi_n \right)^T \tag{7}$$

Mit Hilfe der Pseudoinversen

$$\Sigma^+ = \begin{pmatrix} \frac{1}{\sigma_1} & & & 0 \\ & \ddots & & \\ & & \frac{1}{\sigma_r} & \\ 0 & & & 0 \end{pmatrix} \tag{8}$$

von Σ lässt sich x auch schreiben als:

$$x = \Sigma^+ b + \sum_{i=r+1}^{n} \xi_i e_i \tag{9}$$

Sei nun A beliebig und $A = U\Sigma V^T$ eine SVD von A. Da U eine orthogonale Matrix ist, gilt $\|U^T z\| = \|z\|$ für alle $z \in \mathbb{R}^m$. Daraus folgt:

$$\|Ax - b\|^2 = \|U^T(Ax - b)\|^2 = \|U^T Ax - U^T b\|^2 \tag{10a}$$

$$= \|U^T U \Sigma V^T x - U^T b\|^2 = \|\Sigma V^T x - U^T b\|^2 \tag{10b}$$

Sei nun $y = V^T x$. Aufgrund der Orthogonalität von V gilt dann $\|y\| = \|V^T x\| = \|x\|$. Da V außerdem vollen Rang hat (d.h. für alle y gibt es ein x mit $V^T x = y$), können wir also, anstatt über alle x zu minimieren, auch über alle y minimieren:

$$\arg\min_{x \in \mathbb{R}^n} \|Ax - b\|^2 = \arg\min_{x \in \mathbb{R}^n} \|\Sigma V^T x - U^T b\|^2 \Leftrightarrow \arg\min_{y \in \mathbb{R}^n} \|\Sigma y - U^T b\|^2 \tag{11}$$

Bei letzterem Minimierungsproblem handelt es sich um ein Ausgleichsproblem in Diagonalgestalt, dessen Lösung sich somit aus Gleichung (9) ergibt:

$$x = Vy = V\left(\Sigma^+ U^T b + \sum_{i=r+1}^{n} \xi_i e_i \right) = V\Sigma^+ U^T b + \sum_{i=r+1}^{n} \xi_i v_i \tag{12}$$

Hierbei bezeichnet v_i die i-te Spalte von V.

Ist der Rang r kleiner der Anzahl der Spalten von A so erhalten wir einen $n - r$-dimensionalen Lösungsraum. Bei vielen technischen Anwendungen reicht es oft eine Lösung zu wählen, z.B. die Lösung mit minimaler Norm. Anhand von Gleichnung (12) sieht man, dass x genau dann minimal ist, wenn der Summenterm verschwindet. Da die Spalten von V linear unabhängig sind, ist dies gleichbedeutend mit $\xi_i = 0$ für $i = r+1, \ldots, n$. Die Lösung mit kleinster Norm x^+ für das Ausgleichsproblem $Ax - b$ ist somit eindeutig bestimmt durch:

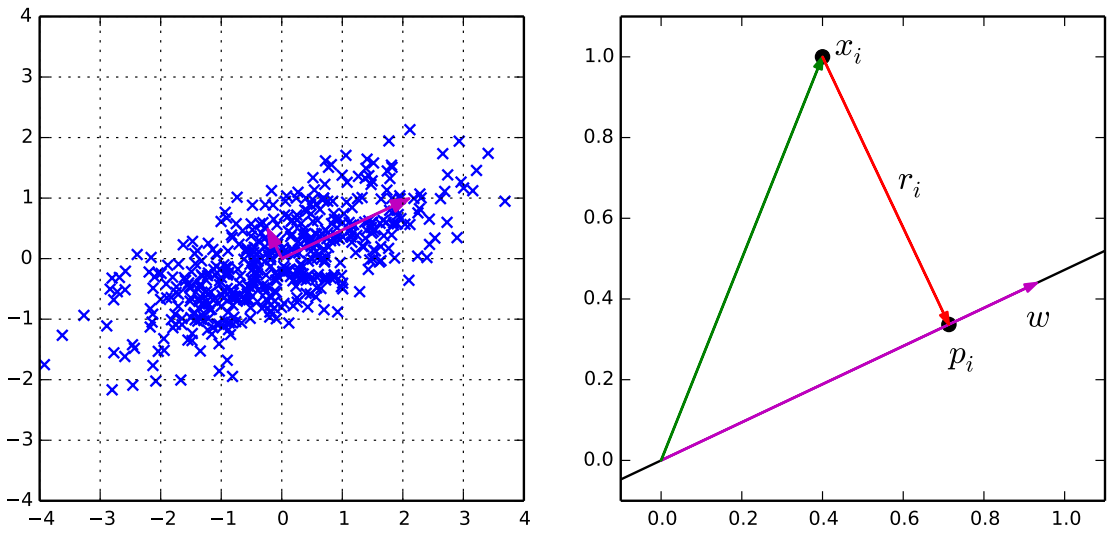
$$x = V\Sigma^+ U^T b \tag{13}$$

Teil 3: Matrixnorm und SVD

Sei $A \in \mathbb{R}^{m \times n}$ und es bezeichne σ_1 den größten Singulärwert von A. Dann gilt:

$$\|A\| := \max_{\|x\|=1} \|Ax\| = \sigma_1 \tag{14}$$

Für positive reelle Zahlen ist Quadrieren streng monoton. D.h. es gilt $a < b \Leftrightarrow a^2 < b^2$ für alle $a, b \geq 0$. Äquivalent zu Gleichung (14) kann also das Maximum von $\|Ax\|^2$ gesucht werden.



(a) (b)

Abbildung 1: (a) Eine Menge von zweidimensionalen Datenpunkten. (b) Projektion p_i des Datenpunktes x_i auf auf den Einheitsvektor w. Der Fehler der Projektion ist r_i.

Betrachten zunächst den Fall, dass A eine Diagonalmatrix mit absteigend sortierten Diagonalelementen $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n$ ist. Dann gilt

$$\Sigma x = \begin{pmatrix} \sigma_1 & & 0 \\ & \ddots & \\ 0 & & \sigma_n \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} \sigma_1 x_1 \\ \vdots \\ \sigma_n x_n \end{pmatrix} \tag{15}$$

und wir können abschätzen:

$$\|\Sigma x\|^2 = \sum_{i=1}^{n} (\sigma_i x_i)^2 \leq \sigma_1^2 \sum_{i=1}^{n} x_i^2 = \sigma_1^2 \|x\|^2 \tag{16}$$

Daraus erhalten wir $\max_{\|x\|=1} \|\Sigma x\|^2 \leq \sigma_1^2$. Da für den Vektor $x = (1, 0, \ldots, 0)^T$ Gleichheit angenommen wird, folgt:

$$\max_{\|x\|=1} \|\Sigma x\|^2 = \sigma_1^2 \tag{17}$$

Sei nun A beliebig und $A = U\Sigma V^T$ eine SVD von A. Dann gilt:

$$\|Ax\|^2 = \langle Ax, Ax \rangle = (Ax)^T Ax = x^T A^T A x \tag{18a}$$

$$= x^T \big(U\Sigma V^T\big)^T \big(U\Sigma V^T\big) x = x^T V \Sigma^T U^T U \Sigma V^T x \tag{18b}$$

$$= x^T V \Sigma^T \Sigma V^T x \tag{18c}$$

Sei nun $y = V^T x$. Da $\|y\| = \|V^T x\| = \|x\|$ und V vollen Rang hat, können wir anstatt über alle x zu maximieren auch über alle y maximieren:

$$\max_{\|x\|=1} \|Ax\|^2 = \max_{\|x\|=1} x^T V \Sigma^T \Sigma V^T x = \max_{\|y\|=1} y^T \Sigma^T \Sigma y = \max_{\|y\|=1} \|\Sigma y\|^2 = \sigma_1^2 \tag{19}$$

Teil 4: Hauptkomponentenanalyse für Punktwolken

Ziel der Hauptkomponentenanalyse (engl. principal component analysis (PCA)) ist es einen Datensatz, welcher in Form einer Menge von Punkten im $\mathbb{R}^n$ gegeben ist, in einem neuen Koordinatensystem darzustellen. Die erste Koordinatenachse wird dabei so gewählt, dass der Fehler, welcher bei der Projektion der Datenpunkte auf die Koordinatenachse entsteht, minimal unter allen Koordinatenachsen ist. Analog werden die anderen Koordinatenachsen gewählt. Die zweite Koordinatenachse wird zum Beispiel

so gewählt, dass der Fehler der Projektion auf den Unterraum, welcher durch die erste und zweite Koordinatenachse aufgespannt wird, minimal ist.

Wir wollen uns die Hauptkomponentenanalyse an einem einfachen Beispiel anschauen. Gegeben sei dazu eine Menge von zweidimensionalen Punkten $x_i \in \mathbb{R}^2$, $i = 1, \ldots, m$ (Abbildung 1(a)). Der Einfachheit halber wollen wir annehmen, dass der Schwerpunkt der Daten im Ursprung liegt. Gesucht ist nun eine neue Koordinatenachse, so dass beim projizieren der Datenpunkte ein minimaler Fehler entsteht. Eine Koordinatenachse lässt sich einfach mittels eines Einheitsvektor w (Abbildung 1) beschreiben. Die Projektion p_i des Datenpunktes x_i auf den durch w aufgespannten Unterraum (d.h. die durch w definierte Gerade) ist dann gegeben durch:

$$p_i = \langle x_i, w \rangle w \tag{20}$$

Der Fehler, welcher bei der Projektion entsteht, lässt sich leicht mittels des Satzes von Pythagoras bestimmen:

$$\|r_i\|^2 = \|x_i\|^2 - \|p_i\|^2 \tag{21}$$

Summieren wir nun $\|r_i\|^2$ für alle Datenpunkte auf, so erhalten wir ein Maß für den Fehler der Projektion der Datenpunkte auf die Koordinatenachse w. Indem wir über alle Koordinatenachsen minimieren, erhalten wir die Koordinatenachse mit minimalem Fehler:

$$\arg\min_{\|w\|=1} \sum_{i=1}^{m} \|r_i\|^2 = \arg\min_{\|w\|=1} \sum_{i=1}^{m} \left(\underbrace{\|x_i\|^2}_{=\text{const}} - \|p_i\|^2 \right) = \arg\max_{\|w\|=1} \sum_{i=1}^{m} \|p_i\|^2 \tag{22}$$

Dabei nutzten wir aus, dass $\|x_i\|^2$ bzgl. der Minimierung konstant ist und vernachlässigt werden kann. Außerdem nutzen wir aus, dass das Minimieren des Negativen eines nicht negativen Ausdruckes äquivalent zum Maximieren des Ausdruckes ist.

Wir wollen nun das Maximierungsproblem unter Verwendung der SVD lösen. Dazu müssen wir zunächst Gleichung (22) in Matrixnotation schreiben. Als erstes schreiben wir $\|p_i\|^2$ als Skalarprodukt

$$\|p_i\|^2 = \left\| \langle x_i, w \rangle w \right\|^2 = \langle x_i, w \rangle^2 \|w\|^2 = \langle x_i, w \rangle^2 , \tag{23}$$

wobei wir ausnutzen, dass w ein Einheitsvektor ist. Die Datenpunkte fassen wir in einer Matrix

$$X = \begin{pmatrix} x_1^T \\ \vdots \\ x_m^T \end{pmatrix} \tag{24}$$

zusammen und zwar so, dass jede Zeile genau einem Datenpunkt entspricht. Der Vektor Xw besteht dann aus den Skalarprodukten $x_i^T w = \|p_i\|$ und wir erhalten:

$$\sum_{i=1}^{n} \|p_i\|^2 = \|Xw\|^2 \tag{25}$$

Gleichung (22) können wir also auch schreiben als:

$$\arg\max_{\|w\|=1} \sum_{i=1}^{n} \|p_i\|^2 = \arg\max_{\|w\|=1} \|Xw\|^2 \tag{26}$$

Sei nun $X = U\Sigma V^T$ eine SVD von X. In Teil 3 hatten wir gesehen, dass $\max_{\|w\|=1} \|Xw\|^2 = \sigma_1^2$ ist. Das Maximum wird dabei von der ersten Spalte von V angenommen, denn es gilt:

$$\|Xv_1\|^2 = \|U\Sigma V^T v_1\|^2 = \|U\Sigma e_1\|^2 = \|\Sigma e_1\|^2 = \|\sigma_1 e_1\|^2 = \sigma_1^2 \tag{27}$$

Zusammengefasst erhalten wir somit:

$$\arg\max_{\|w\|=1} \|Xw\|^2 = v_1 \tag{28}$$

In folgendem Beispiel werden die Hauptkomponenten einer künstlich erzeugten Punktwolke berechnet. Die Erzeugung der Punktwolke erfolgt in zwei Schritten. Zunächst wird eine Matrix mit zufälligen (normalverteilten) Koordinaten erzeugt (`np.random.randn`). Durch Multiplikation der ersten und zweiten Spalte mit $\sqrt{2.0}$ und $\sqrt{0.25}$ wird dann erreicht, dass die erste und zweite Koordinate der Punkte unterschiedliche Varianzen haben (nämlich 2.0 und 0.25). Das bedeutet also, dass die Punktwolke stärker entlang der ersten Koordinatenachse ausgedehnt ist. Durch Rotation um $25°$ erhalten wir somit einen Datensatz welcher entlang der $25°$-Achse ausgedehnt ist.

Für die Bestimmung der Hauptkomponenten wird zunächst der Schwerpunkt der Punktwolke berechnet und dann abgezogen um den Schwerpunkt in den Koordinatenursprung zu verschieben. Nun wird eine SVD der Punktwolke bestimmt. Zu beachten ist hier, dass von `numpy.linalg.svd` nicht V sondern V^T zurückgegeben wird. Das bedeutet die i-te Spalte von V findet sich in `V[i]`. Die Singulärwerte stellen ein Maß für die Ausdehnung der Daten in Richtung der jeweiligen Hauptachse dar (genauer: $\sqrt{\sigma_i^2/(m-1)}$ ist die korrigierte Stichprobenvarianz). Zur besseren Visualisierung werden daher die Hauptkomponenten mit dem dreifachen der geschätzten Standardabweichungen multipliziert.

```
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
from math import sqrt, atan2

N = 500
std_x = sqrt(2.0)
std_y = sqrt(0.25)
theta = np.deg2rad(25.0)

# generate point cloud
A = np.random.randn(N, 2)
A[:,0] *= std_x
A[:,1] *= std_y
cn = np.cos(theta)
sn = np.sin(theta)
R =  np.array([[cn,-sn],[sn,cn]])
P = R.dot(A.T).T

# subtract mean to move center to coordinate origin
A -= A.mean(0)

# compute principal components
U,S,V = np.linalg.svd(P, full_matrices=0)
std1 = sqrt(S[0]**2 / (N-1))
std2 = sqrt(S[1]**2 / (N-1))
pc1 = V[0] * 3.0 * std1
pc2 = V[1] * 3.0 * std2

# show ellipse defined by principal components and estimated standard deviation
plt.gca().add_artist(matplotlib.patches.Ellipse(xy=(0,0), width=6*std1,
                                                height=6*std2,
                                                angle=np.rad2deg(atan2(pc1[1], pc1[0])),
                                                fill=False, color='y'))

# plot point cloud
plt.scatter(P[:,0], P[:,1], c='b', marker='x')

# plot principal components
plt.arrow( 0.0, 0.0, pc1[0], pc1[1], color='r', width=0.005, length_includes_head=True)
plt.arrow( 0.0, 0.0, pc2[0], pc2[1], color='g', width=0.005, length_includes_head=True)

plt.xlim(-4, 4)
plt.ylim(-4, 4)
plt.grid(True)
plt.gca().set_aspect('equal')
plt.show()
```

Im folgenden ist die Ausgabe eines Programmlaufes zu sehen. Die erste Hauptachse ist in rot und die zweite Hauptachse ist in grün eingezeichnet. Zur besseren Visualisierung ist außerdem die durch die skalierten Hauptkomponenten definierte Ellipse eingezeichnet.