Prof. Dr. M. Helbig
Mathematik – Übung

Hochschule Rosenheim
University of Applied Sciences

Anwendungen Eigenwerttheorie & Google PageRank

Fragen?

Anwendungen von linearer Algebra und Eigenwerten

- Mathematik
  - Geometrie (z.B. Drehung, Spiegelung, Streckung)
  - lineare Differentialgleichungen (Schwingungen in Physik: Federpendel, elektrischer Schwingkreis)
  - Graphentheorie: Vorlesung GDI
  - Stochastik/Statistik: Vorlesung Mathe 3
- Informatik
  - Bildbearbeitung (z.B. Drehung, Spiegelung, Streckung, Kompression)
  - Kryptographie (AES/DES)
  - ...
  - Google PageRank-Algorithmus: Jetzt!

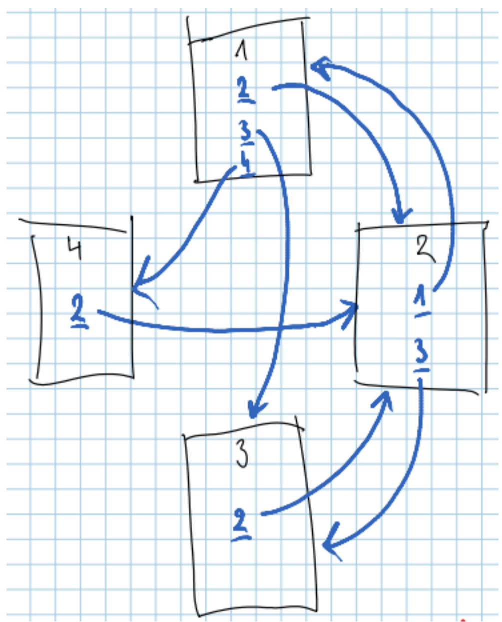
Google PageRank

Bei der Google-Suche wird eine Auflistung von Websites gegeben, die den Suchbegriff enthalten. In welcher Reihenfolge sollen die Ergebnisse angezeigt werden? → Rangliste von Webseiten (PageRank).

Frage: Welche Webseite ist "wichtig" und soll vorne angezeigt werden?

Grundlegende Idee: Seite, die am meisten verlinkt ist, sollte wichtig sein. Auch sollte eine Verlinkung von einer wichtigen Seite mehr Einfluss haben.

Beispiel mit vier Webseiten:



Mathematisches Modell: Link-Matrix $L = (L_{ij}) \in \mathbb{R}^{4\times 4}$ mit $L_{ij} :=$

ergibt

$$L =$$

Gebe einer Seite i ein Gewicht

$$x_i = \sum_{j=1}^{4} L_{ji} \overset{i=2}{=} \text{\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_}$$

1. Verbesserung. Jede Seite i soll insgesamt nur eine Stimme haben und diese gleichmäßig auf ihre Links verteilen:

$$x_i = \sum_{j=1}^{4} \frac{L_{ji}}{n_j} \overset{i=2}{=} \text{\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_}$$

mit $n_j :=$ Anzahl der Links auf Seite j. Ich bekomme somit eine neue gewichtete Link-Matrix

$$A = \left(\frac{L_{ji}}{n_j}\right) =$$

und bekomme ein Ranking mit Hilfe der Gewichte:

1. Seite ___ mit Gewicht $x =$ _____
2. Seite ___ mit Gewicht $x =$ _____
3. Seite ___ mit Gewicht $x =$ _____
4. Seite ___ mit Gewicht $x =$ _____

2. Verbesserung. Ein Link von einer Seite mit hohem Gewicht (also wichtige Seite) soll mehr Einfluss haben, als eine Seite mit niedrigem Gewicht:

$$x_i = \sum_{j=1}^{4} \frac{L_{ji}}{n_j} \cdot x_j$$

Problem. Die Maus beisst sich in den Schwanz: Zur Berechnung des Gewichts x_i brauche ich bereits alle Gewichte x_j.

Dies lässt sich aber in Matrixform schreiben:

also $x = Ax$, d.h. $Ax = 1 \cdot x$, d.h. ______________________________

Übung. Berechnen Sie x.

Problem. Hier ist $n = 4$, in Realität ist n sehr groß! Das Gauß-Verfahren zur Berechnung des Eigenvektors x braucht zu lange.

Lösung. Man führt eine Iteration von Vektoren

$$x(k+1) = A \cdot x(k)$$

mit einem Startwert $x(0)$ ein. Da A viele Nullen besitzt (eine Seite verlinkt nur auf wenige Seiten im Internet), kann man diese Matrixmultiplikationen vergleichsweise schnell berechnen.

Deutung als Zufallssurfer-Modell: $x(k)$ gibt an wie viele Surfer sich im k-ten Schritt auf jeder Seite befinden und in jedem Schritt clickt der Surfer zufällig einen Link. Nach vielen Iterationen stellt sich ein Gleichgewicht ein und die meisten Surfer finden sich dann auf einer Seite mit hohem Gewicht.

Weiteres Problem. Dies muss jedoch nicht immer konvergieren (man stelle sich vor, dass Surfer auf eine Seite gelangen, die keine Links enthält; im Laufe der Zeit würden hier viele Surfer gefangen, obwohl diese Seite evtl. gar kein so hohes Gewicht hat).

Lösung. Es wird noch ein Dämpfungsfaktor $\alpha \in (0,1)$ eingeführt (in der Praxis $\alpha = 0.85$):

$$x = (1-\alpha)\begin{pmatrix}1\\ \vdots\\ 1\end{pmatrix} + \alpha A x$$

D.h. jede Seite bekommt ein Grundgewicht von $1-\alpha$, den Rest des Gewichts wie bisher, aber mit Faktor α versehen.

Die exakte Lösung dieser Gleichung ist $x = (1-\alpha)(E_n - \alpha A)^{-1}\begin{pmatrix}1\\ \vdots\\ 1\end{pmatrix}$, die wieder zu aufwändig zu berechnen wäre. Deswegen macht man wieder eine Iteration von Vektoren und erhält den **PageRank**-Algorithmus:

$$x(k+1) = (1-\alpha)\begin{pmatrix}1\\ \vdots\\ 1\end{pmatrix} + \alpha A x(k)$$