3. 4 Die LU-Zerlegung

Idee: Faktorisierung von A in „einfache“ Matrizen.

3.4.1. Definition:

Sei $\mathbf{A_k}$ die Matrix, die im k-ten Schritt der Gauss-Elimination bearbeitet wird, also $\mathbf{A_1 = A}$ die Ausgangsmatrix und $\mathbf{A_n = U}$ die Endmatrix in oberer Dreiecksgestalt.

Die Gewichte $\mathbf{l(s,k)=l_{s,k}}$ aus obigem GE-Algorithmus schreiben wir in eine neue untere Dreiecksmatrix:

$$L := \begin{pmatrix} 1 & & & & & \\ l_{2,1} & \ddots & & & & \\ \vdots & \ddots & 1 & & & \\ \vdots & & l_{k+1,k} & 1 & & \\ \vdots & & \vdots & \ddots & \ddots & \\ l_{n,1} & \cdots & l_{n,k} & \cdots & l_{n,n-1} & 1 \end{pmatrix}$$

Für jede Spaltenelimination sammeln wir die Gewichte in

$$L_k := \begin{pmatrix} 0 & & & & \\ \vdots & \ddots & & & \\ \vdots & & 0 & & \\ \vdots & & l_{k+1,k} & 0 & \\ \vdots & & \vdots & \vdots & \ddots \\ 0 & & l_{n,k} & 0 & \cdots & 0 \end{pmatrix}$$

P sei die Permutationsmatrix zu den Pivotvertauschungen.

Beispiel: Vertauschung 1←→ 2 entspricht $P = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$

Im Folgenden betrachten wir zur Vereinfachung die Gauss-Elimination ohne Pivotsuche und ohne Permutation **P**.

Der k-te Schritt der GE kann als Matrixprodukt beschrieben werden in der Form

$$A_{k+1} = (I - L_k)A_k = A_k - L_k A_k$$ mit Einheitsmatrix I.

(Ziehe von den unteren Zeilen von A_k jeweils die entsprechend gewichtete k-te Zeile ab.)

$$L_k \cdot A_k = \begin{pmatrix} 0 & \cdots & 0 \\ \cdot & 0 & \cdot \\ \cdot & l_{k+1,k} & \cdot \\ \vdots & \vdots & \vdots \\ 0 & l_{n,k} & 0 \end{pmatrix} * \begin{pmatrix} * \\ A_{k,.} \\ * \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ l_{k+1,k} * A_{k,.} \\ \vdots \\ l_{n,k} * A_{k,.} \end{pmatrix}$$

Dabei bezeichnet $A_{k,.}$ die k-te Zeile von A_k .

Insgesamt:

$$U = A_n = (I - L_{n-1})A_{n-1} = \cdots = \overbrace{(I - L_{n-1})\ldots(I - L_1)}^{\tilde{L}}A_1 = \tilde{L}A$$

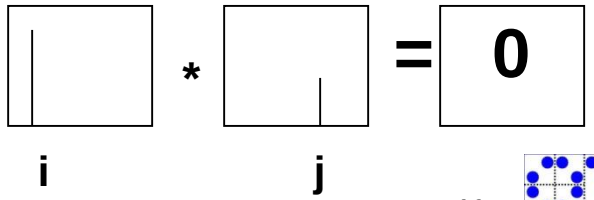
mit der Matrix

$$\tilde{L} := (I - L_{n-1})\ldots(I - L_2)(I - L_1)$$

Wie sieht die Inverse dieser Matrix aus? Es gilt

$$(I + L_j)\cdot(I - L_j) = I - L_j + L_j - L_j^2 = I$$

Dabei verwenden wir: $\mathbf{L_i L_j = 0}$ für **i<=j;**



Also:

$$\tilde{L}^{-1} = \left((I - L_{n-1}) \cdots (I - L_1)\right)^{-1} =$$

$$= (I - L_1)^{-1} \cdots (I - L_{n-1})^{-1} =$$

$$= (I + L_1) \ldots (I + L_{n-1})$$

und wegen

$$(I + L_1) \cdot (I + L_2) = I + L_1 + L_2 + L_1 L_2 = I + L_1 + L_2$$

dann auch

$$\tilde{L}^{-1} = (I + L_1) \ldots (I + L_{n-1}) = I + L_1 + L_2 + \ldots L_{n-1} = L.$$

Damit ergibt sich: $U = \tilde{L} A \quad \leftrightarrow \quad A = LU$

Allgemeiner mit Pivotsuche muss noch die Permutation **P** mitberücksichtigt werden, die durch das Auswählen der Pivotelemente hervorgerufen wird. Dann gilt:

$$PA = LU$$

Allgemeines Vorgehen:
Sammle aus dem Algorithmus 3.3.4 der Gauss-Elimination die Gewichte $l_{j,k}$ in Matrix L und bezeichne mit U die obere Dreiecksmatrix, die sich als Endresultat der GE ergibt.
Sammle die Pivotvertauschungen in Permutationsmatrix **P**.

In dem betrachteten Beispiel:

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 10 & -7 & 0 \\ -3 & 2 & 6 \\ 5 & -1 & 5 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0.5 & 1 & 0 \\ -0.3 & -0.04 & 1 \end{pmatrix} \cdot \begin{pmatrix} 10 & -7 & 0 \\ 0 & 2.5 & 5 \\ 0 & 0 & 6.2 \end{pmatrix}$$

Vorteil: Nach einmaliger Durchführung und Speicherung von L und U kann jedes weitere Gleichungssystem in A schnell gelöst werden:

$$Ax = P^T L(Ux) = b \quad \rightarrow \quad Ly = Pb \quad und \quad Ux = y$$

Bem.: Speichere ev. L im Array von A an Stelle der entstehenden Nullen.

3.5 Die Kondition eines linearen Gleichungssystems (einer Matrix)

Neu benötigt: Matrixnorm und ihre Eigenschaften.

|| . || Matrixnorm: ||A|| > 0 für A ≠ 0

||a*A|| = |a|*||A|| für a∈R

||A+B|| <= ||A|| + ||B||

Besonders wichtig sind Matrixnormen, die zu einer Vektornorm „passen“ :

3.5.1. Eine Matrixnorm ist mit einer Vektornorm verträglich, wenn $\|Ax\| \leq \|A\| \cdot \|x\|$

Vektor Matrix Vektornorm

3.5.2. Eine Matrixnorm heißt submultiplikativ, wenn stets gilt

$$\|A \cdot B\| \leq \|A\| \cdot \|B\|$$

3.5.3. Eine submultiplikative Matrixnorm, verträglich mit einer vorgegeben Vektornorm, kann leicht definiert werden durch

$$\|A\| = \sup_{x \neq 0} \frac{\|Ax\|}{\|x\|}$$

die sog. Grenzen-Norm oder lub-Norm (least upper bound):

$$\frac{\|Ay\|}{\|y\|} \leq \sup_x \frac{\|Ax\|}{\|x\|} = \|A\| \Rightarrow \|Ay\| \leq \|A\| \cdot \|y\| \qquad \forall y \neq 0$$

Für $Bx \neq 0$:

$$\|A \cdot B\| = \sup \frac{\|ABx\|}{\|x\|} = \sup_x \left(\frac{\|ABx\|}{\|Bx\|} \cdot \frac{\|Bx\|}{\|x\|} \right) \leq$$

$$\leq \sup_{y=Bx} \frac{\|Ay\|}{\|y\|} \cdot \sup_x \frac{\|Bx\|}{\|x\|} \leq \|A\| \cdot \|B\|$$

So erhält man zu den Vektornormen $\|x\|_1 = \sum_{i=1}^{n} |x_i|$

$$\|x\|_\infty = \max_{i=1,\ldots,n} |x_i|$$

dazugehörige Matrix-Grenzen-Normen $\|\cdot\|_1$ und $\|\cdot\|_\infty$.

3.5.4. Das innere Produkt $(x, y) = x^T y = \sum x_i \cdot y_i$

führt auf die

3.5.5. Euklid'sche Norm: $\|x\|_2^2 = (x, x) = x^T \cdot x = \sum x_i^2$

und die damit verträgliche **Matrixnorm**

3.5.6. (2-Norm): $\|A\|_2 = \sup_{x \neq 0} \frac{\|Ax\|_2}{\|x\|_2}$ als maximale Längenänderung

mit der Eigenschaft

$$\|Ax\|_2^2 = (Ax, Ax) = (Ax)^T (Ax) = x^T A^T Ax = x^T (A^T A) x$$

$$\|A\|_2^2 = \sup_{x \neq 0} \frac{\|Ax\|_2^2}{\|x\|_2^2} = \sup_{x \neq 0} \frac{x^T (A^T A) x}{x^T x} = \lambda_{\max} (A^T A)$$

3.5.7. Anmerkung:

Speziell wichtige Klasse von Matrizen mit Norm 1:

Q ist orthogonale Matrix ←→ $Q^{-1} = Q^T$

oder $Q\,Q^T = I = Q^T Q$,

$$\|Qx\|_2^2 = x^T Q^T Q x = x^T x = \|x\|_2^2;$$

$$\|Qx\|_2 = \|x\|_2 \quad \rightarrow \quad \|Q\|_2 = \sup_{x \neq 0} \frac{\|Qx\|_2}{\|x\|_2} = 1$$

Basistransformation auf “ideales” Koordinatensystem:

$$\|A\|_2 = \|U^T \Lambda U\|_2 = \|\Lambda\|_2 = \max_{i=1,\ldots,n} |\lambda_i|$$

Eigenwertzerlegung von $A=A^T$. U ist orthogonale Matrix

Sei A nun eine allgemeine Matrix → SVD
Dann kann man die Singuläre Werte Zerlegung $A=U^T\Sigma V$ benutzen

Basis zu A^TA und AA^T → zwei „ideale“ Koordinatensysteme

$A = U^T \Sigma V$ Wobei U die Eigenvektoren von AA^T sind,
V die Eigenvektoren von A^TA, und
Σ, die sog. Singulären Werte von A
sind die Wurzeln aus den Eigenwerten
von A^TA und auch von AA^T.

$$\|A\|_2 = \|U^T \Sigma V\|_2 = \|\Sigma\|_2 = \max_{i=1,\ldots,n} \sqrt{|\lambda_i (A^T A)|} = \sigma_{\max}$$

Also $\|A\|_2 = \sqrt{\lambda_{\max}(A^T A)} = \sigma_{\max}(A)$

wobei $\lambda_{\max}(A^T A)$ der größte Eigenwert von $A^T A$ ist, und $\sigma_{\max}(A)$ der größte Singulärwert von A.

Diese Matrixnorm ist i.A. nicht einfach berechenbar!

3.5.8. Def. der Frobeniusnorm: $\|A\|_F := \sqrt{\sum |a_{j,k}|^2}$

Fasse die Matrix A als Vektor auf und benutze für diesen langen Vektor die euklid'sche Vektornorm → $\|.\|_F$

Die Frobeniusnorm ist mit der euklid'scher Vektornorm verträglich und submultiplikativ.

3.6 Gestörte Eingabedaten

Für ein lineares Gleichungssystem **A x = b** mit Matrix **A,** Vektor der rechten Seite **b** und gesuchtem Lösungsvektor **x,** untersuchen wir wieder den Einfluss von Eingabefehlern bei sonst exakter Rechnung

→ Kondition der Matrix

Eingabedaten mit Fehler: $b \rightarrow b + \Delta b = \tilde{b}$

Damit ergibt sich an Stelle der exakten Lösung x die Näherung

$$x \rightarrow x + \Delta x = \tilde{x}$$

Mit **Ax = b** und $A\tilde{x} = A(x + \Delta x) = b + \Delta b = \tilde{b}$

gilt: $A\Delta x = \Delta b$ **oder** $\Delta x = A^{-1}\Delta b$

Die Matrix **A** wird hier als exakt angenommen!

Damit erhält man die Ungleichungen

$$\|\Delta x\|_2 = \|A^{-1}\Delta b\|_2 \le \|A^{-1}\|_2 \|\Delta b\|_2$$

$$\|b\|_2 = \|Ax\|_2 \le \|A\|_2 \|x\|_2$$

wegen der Verträglichkeit der euklid'schen Vektor- und Matrixnorm.

Also auch $\frac{1}{\|x\|_2} \le \frac{\|A\|_2}{\|b\|_2}$ und damit insgesamt:

3.6.1.

$$\frac{\|\Delta x\|_2}{\|x\|_2} \le \left(\|A^{-1}\|_2 \cdot \|A\|_2\right) \cdot \frac{\|\Delta b\|_2}{\|b\|_2}$$

rel.Fehler in x — **Kondition von A** — **rel. Fehler in b**

3.6.2 Definition: Die Kondition der Matrix A bzgl. der euklid'schen Norm ist gegeben durch

$$cond_2(A) = \left\| A^{-1} \right\|_2 \cdot \left\| A \right\|_2$$

(Genauso kann man Konditionszahlen bzgl. anderer verträglicher Normen definieren.)

Die Konditionszahl beschreibt also wieder, wie sich ein relativer Eingabefehler in Vektor b auf das Resultat, den Lösungsvektor x, auswirkt.

cond(A) groß → kleine Störungen in b bewirken große Fehler in x

Wieder: Numerisch stabil, wenn alle Zwischenmatrizen gut konditioniert sind!

$$cond_2(Q) = \| Q^{-1} \|_2 \| Q \|_2 = \| Q^T \|_2 \| Q \|_2 = 1$$

Außerdem gilt

$$cond_2(QA) = cond_2(A)$$

denn $$\|QA\|_2 = \sup_{x \neq 0} \frac{\|QAx\|_2}{\|x\|_2} = \sup_{x \neq 0} \frac{\|Ax\|_2}{\|x\|_2} = \|A\|_2$$

Orthogonale Matrizen sind selbst gut konditioniert und lassen bei Multiplikation mit einer Matrix die Kondition der Ausgangsmatrix unverändert!

Man beachte: Der relative Fehler des Gesamtvektors wird abgeschätzt, nicht der Fehler einzelner Komponenten!

Als Vektor betrachtet ist $\begin{pmatrix} 10^{-5} \\ 1 \end{pmatrix} \approx \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ 

Aber komponentenweise ist natürlich $10^{-5} \not\approx 0$

Beispiel: $A = \begin{pmatrix} 10^{-9} & 1 \\ 0 & 1 \end{pmatrix}$ *und* $A^{-1} = \begin{pmatrix} 10^{9} & -10^{9} \\ 0 & 1 \end{pmatrix}$

Normen: $\|A\|_2 \approx \sqrt{2}\,,\ \|A^{-1}\|_2 \approx \sqrt{2}\cdot 10^9\,,\ \ cond_2(A) \approx 2\cdot 10^9$

Wähle speziell: $b = \begin{pmatrix} 1 \\ 1 \end{pmatrix},\quad \Delta b = \begin{pmatrix} \Delta b_1 \\ 0 \end{pmatrix},\quad \Rightarrow\quad x = A^{-1}b = \begin{pmatrix} 0 \\ 1 \end{pmatrix},$

Dann folgt $|\Delta x_1| = 10^9 \cdot |\Delta b_1|, \Delta x_2 = 0 \quad \|x\| = 1, \quad \|b\| = \sqrt{2}$

Damit ergibt sich $\dfrac{\|\Delta x\|}{\|x\|} = \dfrac{10^9 \cdot |\Delta b_1|}{1} = \sqrt{2}\cdot 10^9 \cdot \dfrac{\|\Delta b\|}{\|b\|}$

Vgl.: $cond(A) = \|A\| \cdot \|A^{-1}\| = \sqrt{2}\cdot\sqrt{2}\cdot 10^9 = 2\cdot 10^9$

Ein kleiner Fehler in der ersten Komponente von b wirkt sich verstärkt um Faktor 10^9 in der ersten Komponente von x aus.

3.7 Kosten der Gauss-Elimination

Zunächst Dreieckssystem:

$$x_n = b_n / a_{nn};$$

$$für \quad i = n-1, n-2, \ldots, 1:$$

$$x_i = \frac{b_i - \sum_{j=i+1}^{n} a_{ij} x_j}{a_{ii}}$$

Programm:

```
FOR i = n,...,1 DO
    x(i) = b(i);
    FOR j=i+1,…,n DO
        x(i) = x(i) - a(i,j)x(j);
    ENDFOR
    x(i) = x(i)/a(i,i);
ENDFOR
```

In jedem Schritt i fallen eine Division und jeweils n-i Additionen und Multiplikationen an:

Also $$\sum_{i=1}^{n-1}(n-i)=\sum_{j=1}^{n-1} j=\frac{n(n-1)}{2}=\frac{n^2}{2}-\frac{n}{2}$$

Additionen und genauso viele Multiplikationen.

Dazu kommen n Divisionen.

3.7.2. Definition: Unter flop verstehen wir eine elementare ‚floating point operation' (Gleitpunktoperation + , - , * , /). Die Kosten eines Algorithmus werden üblicherweise in flop angegeben. Dazu gibt man in erster Näherung nur den Term höchster Ordnung an.

In unserem Fall: $$2\cdot\left(\frac{n^2}{2}-\frac{n}{2}\right)+n=n^2 \quad\rightarrow\quad n^2 \qquad flop$$

oder $$O(n^2) \qquad flop$$

3.7.3. Exkurs: Landau'sche Symbole:

1. Betrachte Funktion in n für n → ∞ :

$$f(n) = \mathrm{O}(g(n)), \quad \textit{falls} \quad \left|\frac{f(n)}{g(n)}\right| \leq M < \infty \quad \textit{für} \quad n \geq N$$

Beispiel: $n^2 + n = \mathrm{O}(n^2)$; denn n^2 ist der am stärksten wachsende Term → f(n) / g(n) = 1 + 1/n <= M

2. Betrachte Funktion in h für h → 0 :

$$f(h) = \mathrm{O}(g(h)), \quad \textit{falls} \quad \left|\frac{f(h)}{g(h)}\right| \leq c < \infty \quad \textit{für} \quad |h| \leq \delta$$

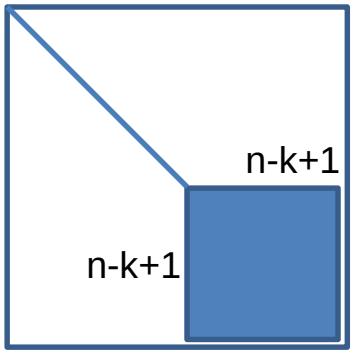
Beispiel: $h^2 + h = \mathrm{O}(h)$; denn h ist der am langsamsten schrumpfende Term → f(h) / g(h) = h + 1 <= c

O(..) bezeichnet also jeweils den dominanten Term!

3.7.4. Kosten der Gauss-Elimination:

Im k-ten Teilschritt arbeitet man in einer n-k+1 x n-k+1 Untermatrix A_k

In dieser Matrix wird für i=k+1,...,n neu berechnet :



$$a_{ij} = a_{ij} - l_{ik} a_{kj}$$

Das sind ca. $(n-k)^2$ Additionen und genauso viele Multiplikationen.

Insgesamt also

$$2\sum_{k=1}^{n-1}(n-k)^2 = 2\sum_{j=1}^{n-1} j^2 = \frac{2(n-1)(2n-1)n}{6} = \frac{2}{3}n^3 + \mathrm{O}(n^2) \qquad flop$$

Dazu kommen $O(n^2)$ flop für die Spaltenpivotsuche und $O(n^2)$ flop für das Auflösen des Dreiecksgleichungssystems.

Diese Kosten fallen aber praktisch nicht ins Gewicht gegenüber den obigen ***2/3*n^3*** .

3.7.5 Beispiel zur Verdeutlichung der Kondition

$$A = \begin{pmatrix} 1 & 0 & \cdots & 0 & 1 \\ -1 & 1 & \ddots & \vdots & 1 \\ -1 & -1 & \ddots & 0 & 1 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ -1 & -1 & \cdots & -1 & 1 \end{pmatrix} \begin{matrix} \downarrow \\ \\ \\ \\ \downarrow \end{matrix} \quad \begin{matrix} \\ \\ \\ \vdots \\ \\ \end{matrix} \quad \begin{matrix} + \\ \\ \\ \\ \downarrow \end{matrix}$$

Eliminiere die erste Spalte durch Addieren der ersten Zeile:

$$\begin{pmatrix} 1 & & & & 1 \\ 0 & 1 & & & 2 \\ 0 & -1 & 1 & & 2 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & -1 & \cdots & -1 & 2 \end{pmatrix} \begin{matrix} \downarrow \\ \\ \\ \\ \downarrow \end{matrix} \quad \begin{matrix} \\ \\ \\ \vdots \\ \\ \end{matrix} \quad \begin{matrix} + \\ \\ \\ \\ \downarrow \end{matrix}$$

Im nächsten Schritt

$$\begin{pmatrix} 1 & & & & 1 \\ 0 & 1 & & & 2 \\ 0 & 0 & 1 & & 4 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & -1 & 4 \end{pmatrix}$$

und schließlich

$$U = \begin{pmatrix} 1 & & & & 1 \\ & 1 & & & 2 \\ & & 1 & & 4 \\ & & & \ddots & \vdots \\ & & & 1 & 2^{n-2} \\ & & & & 2^{n-1} \end{pmatrix}$$

In jedem Schritt verdoppelt sich der größte Eintrag in der Matrix!

Kondition von A selbst ist **O(n),** Kondition von U ist **O(2^{n-1}) !**

Hilft Pivotsuche?

Im Verlauf der Gauss-Elimination *kann* die Kondition der Matrizen sehr stark anwachsen!

Aber: In der Praxis kommt das *so gut wie* nie vor!

Gauss-Elimination mit Pivotsuche *gilt als* numerisch stabil für gut konditionierte Systeme.

3.8 Methode der kleinsten Quadrate

(Least Squares, Normalgleichung)

Ausgangspunkt: Überbestimmtes System.

Mehr Gleichungen als Unbekannte

$$\boxed{A}\, x = b$$

Sei A eine m x n – Matrix mit m>n und maximal vollem Rang:

rang(A) = n, d.h. A bildet den $\mathbf{R^m}$ in den ganzen $\mathbf{R^n}$ ab.

Das System A x = b ist dann i.A. nicht lösbar!

Versuche, das Problem so gut wie möglich zu lösen!
Kompromiss!
Minimiere dazu die Abweichung A x - b in passender Norm!

Beispiele:

$$\begin{pmatrix}1\\1\end{pmatrix} x_1 = \begin{pmatrix}0\\2\end{pmatrix} \Rightarrow x_1 = 1$$

$$x_1 = 0$$
$$x_1 = 2$$

$$\begin{pmatrix}1 & 0\\1 & 0\end{pmatrix}\begin{pmatrix}x_1\\x_2\end{pmatrix} = \begin{pmatrix}0\\2\end{pmatrix} \Rightarrow x_1 = 1, x_2 \ \textit{beliebig}$$

$$x_1 + 0 * x_2 = 0$$
$$x_1 + 0 * x_2 = 2$$

$$\begin{pmatrix}1 & 1\\1 & 1\end{pmatrix}\begin{pmatrix}x_1\\x_2\end{pmatrix} = \begin{pmatrix}0\\2\end{pmatrix} \Rightarrow \begin{pmatrix}1\\1\end{pmatrix}(x_1 + x_2) = \begin{pmatrix}0\\2\end{pmatrix} \Rightarrow x_1 + x_2 = 1$$

$$x_1 + x_2 = 0$$
$$x_1 + x_2 = 2$$

$$\begin{pmatrix}0 & 0\\0 & 0\end{pmatrix}\begin{pmatrix}x_1\\x_2\end{pmatrix} = \begin{pmatrix}0\\2\end{pmatrix}$$

x_1 und x_2 beliebig

Minimiere die Abweichung $A\,x - b$ in passender Norm!

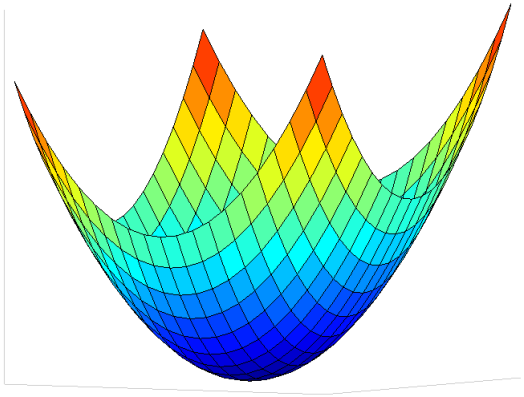
Am besten eignet sich dazu die euklid'sche Norm, da sie auf eine differenzierbare Funktion f führt:

3.8.1. : $\min_x \|Ax-b\|_2^2$

$$f(x_1,\ldots,x_n) := \|Ax-b\|_2^2 =$$

$$= (Ax-b)^T (Ax-b) = x^T A^T Ax - 2x^T A^T b + b^T b =$$

$$= \left\| \begin{pmatrix} (\sum_{j=1}^{n} a_{1,j} x_j) - b_1 \\ \vdots \\ (\sum_{j=1}^{n} a_{m,j} x_j) - b_m \end{pmatrix} \right\|_2 = \sum_{k=1}^{m} \left((\sum_{j=1}^{n} a_{k,j} x_j) - b_k \right)^2$$



Die Funktion f beschreibt einen Paraboloiden (n-dim. Parabel).

Das eindeutige Minimum dieser Funktion ist an der Stelle, an der die Ableitung gleich Null ist (waagrechte Tangente).

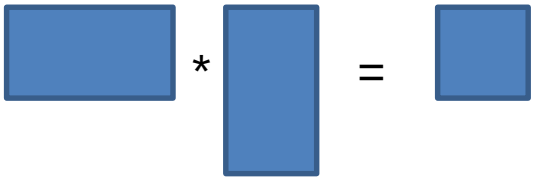
$$0 = \frac{df}{dx_i} = 2\sum_{k=1}^{m}\left(\sum_{j=1}^{n} a_{k,j}x_j - b_k\right)a_{k,i} \qquad \text{für i=1,...,n}$$

oder $$\sum_{k=1}^{m} a_{k,i}\sum_{j=1}^{n} a_{k,j}x_j = \sum_{k=1}^{m} a_{k,i}b_k \qquad \text{für i=1,...,n}$$

In Matrixschreibweise: $\left(A^T Ax\right)_i = \left(A^T b\right)_i$, i=1,...,n

3.8.2. Normalgleichung zu Ax=b: $A^T Ax = A^T b$

$$A^T \cdot \left(Ax = b\right) \Rightarrow A^T Ax = A^T b$$



Die Matrix $\mathbf{A^TA}$ ist eine n x n – Matrix von Rang n (wenn A Rang n hat) und beschreibt daher ein eindeutig lösbares, quadratisches lineares Gleichungssystem.

Allerdings ist die Kondition von A^TA oft sehr viel schlechter als die von A, denn:

$$cond_2(A^TA) = \left\|A^TA\right\|_2 \cdot \left\|(A^TA)^{-1}\right\|_2 =$$

$$= \sqrt{\lambda_{\max}(A^TAA^TA) * \lambda_{\max}((A^TA)^{-1}(A^TA)^{-1})} =$$

$$= \lambda_{\max}(A^TA) \cdot \lambda_{\max}(inv(A^TA)) =$$

$$= \lambda_{\max}(A^TA) / \lambda_{\min}(A^TA) = \sigma^2_{\max}(A) / \sigma^2_{\min}(A) =$$

$$= \left\|A\right\|_2^2 \cdot \left\|A^{-1}\right\|_2^2 = cond^2(A)$$

Im folgenden Abschnitt werden wir daher ein besseres Verfahren zur Lösung dieses Problems kennen lernen.

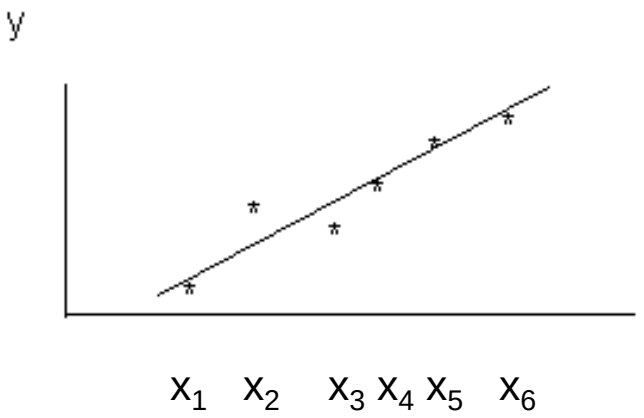
Dazu werden besser orthogonale Matrizen verwendet, um diese Konditionsverschlechterung zu vermeiden.

3.8.3. Lineares Ausgleichsproblem (Ausgleichsgerade)

Gegeben: Punktepaare in der Ebene , (x_i, y_i), i=1,...,n;

Gesucht: beste Gerade, die möglichst nahe an den Punkten liegt.

y = g(x) = ax + b .



Es soll also gelten:

$$\begin{pmatrix} a + bx_1 \\ \vdots \\ a + bx_n \end{pmatrix} \approx \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}$$

oder in Matrixschreibweise

$$A\begin{pmatrix} a \\ b \end{pmatrix} \approx y \quad \text{mit} \quad A = \begin{pmatrix} 1 & x_1 \\ \vdots & \vdots \\ 1 & x_n \end{pmatrix}, \quad \text{und} \quad y = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}.$$

Die Normalgleichung lautet also

$$A^T A\begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} n & \sum_{j=1}^{n} x_j \\ \sum_{j=1}^{n} x_j & \sum_{j=1}^{n} x_j^2 \end{pmatrix}\begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} \sum_{j=1}^{n} y_j \\ \sum_{j=1}^{n} x_j y_j \end{pmatrix}.$$

Die Lösung dieses 2 x 2 – Gleichungssystems liefert a und b, und damit die gesuchte Gerade y = ax + b .

MATLAB-Beispiel.

Allgemeiner:

Ansatzfunktionen $g_1(x),\ldots,g_m(x)$ z.B. cos(kx) und

Punkte $(x_1,y_1),\ldots,(x_n,y_n)$, n>m

Gesucht : $f(x)=\sum_{k=1}^{m} a_k g_k(x)$ mit $f(x_j)\approx y_j,\ j=1,\ldots,n$

Mit $G=\begin{pmatrix} g_1(x_1) & \cdots & g_m(x_1) \\ \vdots & & \vdots \\ g_1(x_n) & \cdots & g_m(x_n) \end{pmatrix}$ ist dann $G^T G\begin{pmatrix} a_1 \\ \vdots \\ a_m \end{pmatrix}=G^T\begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}$

zu lösen.

Ergebnis ist die ‚näheste' Funktion an den vorgegebenen Punkten, die aus den g_1,..., g_m linear zusammengesetzt ist.

$f(x)=a\cdot 1+b\cdot x^2+c\cdot\sin(3x)+d\cdot e^{-x}, optimale \quad a,b,c,d$

Beispiel: Schwingungsanteile

Gemessenes Audio-Signal $y = (y_1,\ldots,y_n)$ an Stellen $x = (x_1,\ldots,x_n)$

Gesucht: Frequenzen, Schwingungsanteile

Ansatzfunktionen: $g_k(x) = \cos(kx), k = 0,\ldots,m$

$$f(x) = a_0 + a_1\cos(x) + \ldots + a_m\cos(mx) \Rightarrow f(x_j) = y_j,\ j = 1,\ldots,n$$

$$\begin{pmatrix} 1 & \cos(x_1) & \cdots & \cos((m-1)x_1) & \cos(mx_1) \\ 1 & \cos(x_2) & \cdots & \cos((m-1)x_2) & \cos(mx_2) \\ \vdots & \vdots & & \vdots & \vdots \\ 1 & \cos(x_{n-1}) & \cdots & \cos((m-1)x_{n-1}) & \cos(mx_n) \\ 1 & \cos(x_n) & \cdots & \cos((m-1)x_{n-1}) & \cos(mx_n) \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{m-1} \\ a_m \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_{n-1} \\ y_n \end{pmatrix}$$

3.9. Die QR-Zerlegung einer Matrix

Schon vorher haben wir bemerkt:

- cond(U) in der Gauß-Elimination ev. groß, auch bei kleinem cond(A);
- falls A schlecht konditioniert: was ist der Rang von A? Welche Pivotelemente werten wir als 0?
- überbestimmte Systeme: cond($A^T A$) oft sehr groß.

Andererseits: cond(QA) = cond(A), falls Q orthogonal.

Also sind orthogonale Matrizen sehr gut für äquivalente Umformungen von A geeignet (vgl. LU-Zerlegung).

Außerdem gilt: $\mathbf{Q^{-1} = Q^T}$.

Also sind Gleichungssysteme in **Q** sehr leicht zu lösen.

Versuche daher, analog zur LU-Zerlegung A=LR ,
eine Zerlegung der Form

$$\mathbf{A = QR}$$

zu bestimmen mit

Q orthogonal

R obere Dreiecksmatrix

Vorteile:

- numerisch stabiler als LU
- „ähnliche“ Kosten, zumindest in O() bis auf Faktor
- Systeme in Q und R leicht zu lösen
- auch für überbestimmte Systeme.

3.9.2 Elementare orthogonale Matrizen

Orthogonale 2 x 2 – Matrix :

Frage: orthogonale 1 x 1 – Matrix?

$$G = \begin{pmatrix} \cos(\varphi) & \sin(\varphi) \\ \sin(\varphi) & -\cos(\varphi) \end{pmatrix}$$

heißt **Givensreflexion.** Denn

$$G^T G = GG = \begin{pmatrix} \cos(\varphi) & \sin(\varphi) \\ \sin(\varphi) & -\cos(\varphi) \end{pmatrix} \begin{pmatrix} \cos(\varphi) & \sin(\varphi) \\ \sin(\varphi) & -\cos(\varphi) \end{pmatrix} =$$

$$= \begin{pmatrix} \cos^2(\varphi) + \sin^2(\varphi) & \cos(\varphi)\sin(\varphi) - \sin(\varphi)\cos(\varphi) \\ \sin(\varphi)\cos(\varphi) - \cos(\varphi)\sin(\varphi) & \sin^2(\varphi) + \cos^2(\varphi) \end{pmatrix} =$$

$$= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix};$$

Wilkinson, Givens, Forsythe, Householder, Henrici, F.L.Bauer

Alternativ ***Givensrotation:***

$$\begin{pmatrix} \cos(\varphi) & \sin(\varphi) \\ -\sin(\varphi) & \cos(\varphi) \end{pmatrix}$$

G ist eindeutig bestimmt durch den ‘Winkel’ φ.
Bestimme nun φ so, dass

$$\tilde{A} = GA = \begin{pmatrix} \tilde{a}_{11} & \tilde{a}_{12} \\ \tilde{a}_{21} & \tilde{a}_{22} \end{pmatrix}$$

obere Dreiecksmatrix wird.

Dazu muss gelten:

$$\tilde{a}_{21} = \begin{pmatrix} \sin(\varphi) & -\cos(\varphi) \end{pmatrix} \begin{pmatrix} a_{11} \\ a_{21} \end{pmatrix} = \sin(\varphi) a_{11} - \cos(\varphi) a_{21} \overset{!}{=} 0$$

Lösung:

$$\cot(\varphi) = \frac{a_{11}}{a_{21}}; \quad \varphi = arccctg\left(\frac{a_{11}}{a_{21}}\right) \quad \text{oder} \quad \varphi = arctg\left(\frac{a_{21}}{a_{11}}\right)$$

Ist $a_{21} = 0$, so ist keine weitere Transformation nötig!

Numerisch stabilere Art der Berechnung :

(a_{21} oder a_{11} könnten fast 0 sein):

$$\rho = sign(a_{11})\sqrt{a_{11}^2 + a_{21}^2}; \quad \cos(\varphi) = \frac{a_{11}}{\rho}; \quad \sin(\varphi) = \frac{a_{21}}{\rho};$$

$$\sin(\varphi)a_{11} - \cos(\varphi)a_{21} = \frac{a_{21}}{\rho}a_{11} - \frac{a_{11}}{\rho}a_{21} = 0; \quad \left(\frac{a_{21}}{\rho}\right)^2 + \left(\frac{a_{11}}{\rho}\right)^2 = 1;$$

3.9.3. Givens-Reflexion für den allgemeinen n x n – Fall:

n-dimensionale Givens-Reflexion ist im Wesentlichen wie die Einheitsmatrix, bis auf den gerade zu betrachtenden 2 x 2 – Unterblock.

Dieser Block wird wieder - wie oben definiert - abhängig von φ bestimmt.

Man eliminiert wieder in der ersten Spalte $a_{2,1},\ldots,a_{m,1}$, und dann entsprechend in der zweiten Spalte die Unterdiagonalelemente, usw. wie bei Gauss.

Gestalt der allgemeinen Givens-Matrix