PA081: Programování numerických výpočtů

5. Systémy lineárních rovnic a optimalizované implementace lineární algebry

Aleš Křenek

jaro 2019

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Motivace

- aparát vektorových a maticových operací
- součást metod řešení nelineárních rovnic, optimalizací atd.
- diferenciální rovnice
  - simulace dynamických systémů
  - metoda konečných prvků
- realistické osvětlení scény (radiosita)
- a mnoho dalších …

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Motivace

- aparát vektorových a maticových operací
- součást metod řešení nelineárních rovnic, optimalizací atd.
- diferenciální rovnice
  - simulace dynamických systémů
  - metoda konečných prvků
- realistické osvětlení scény (radiosita)
- a mnoho dalších ...
- relativně snadná implementace, podpora v HW

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Motivace

Simulace pohybu tělesa

- založena na Newtonově zákoně $F = ma$
- vede na systém diferenciálních rovnic (13 proměnných)

$$\frac{d\mathbf{y}(t)}{dt} = \begin{pmatrix} d\mathbf{x}/dt \\ d\mathbf{q}/dt \\ d\mathbf{P}/dt \\ d\mathbf{L}/dt \end{pmatrix} = \begin{pmatrix} \frac{1}{m}\mathbf{P} \\ \frac{1}{2}\omega_q \mathbf{q} \\ \mathbf{F} \\ \mathbf{T} \end{pmatrix}$$

- pro simulaci potřebujeme opakovaně řešit

$$\mathbf{y}_n = \mathbf{y}_{n-1} + \Delta t \left.\frac{d\mathbf{y}}{dt}\right|_{\mathbf{y}_n}$$

- triková aproximace – zanedbání vyšších členů Taylorova rozvoje $d\mathbf{y}/dt$ jako funkce $\mathbf{y}$
- konečný krok od $\mathbf{y}_{n-1}$ k $\mathbf{y}_n$ znamená řešení systému 13 lineárních rovnic

Motivace
Výkon současných CPU
Blokové algoritmy
BLAS
Asymptotická složitost
Systémy lineárních rovnic
Přehled metod
Gaussova eliminace
Rozklady matic
LU dekompozice
Choleského dekompozice
QR dekompozice
Dekompozice na singulární

Motivace

Modely měkkých tkání

- Interakce s měkkými tkáněmi
  - tréning chirurgů – vytvořit simulaci chování tkání
  - potřebujeme vytvořit matematický model tkáně
  - s touto tkání pak můžeme interagovat (např. hapticky)
  - je třeba simulovat chování tkáně s dostatečnou přesností
- použité matematické modely
  - deformovatelná tělesa
  - chování je popsáno parciálními diferenciálními rovnicemi, zpravidla neznáme analytické řešení
  - řešíme numericky pomocí diskretizace metodou konečných prvků (FEM)

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

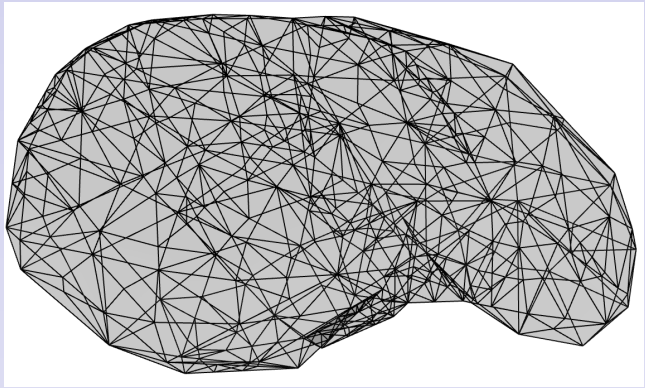
QR dekompozice

Dekompozice na singulární

Motivace

Metoda konečných prvků

- simulovaná tkáň je geometricky aproximována meshem



- deformace popisuje teorie elasticity
- geometricky lineární model – systém lineárních rovnic
- geometricky nelineární model – systém nelineárních rovnic
  - v každém kroku iterační metody systém lineárních rovnic
- systémy rovnic jsou řídké (ovlivňují se jen přímo spojené uzly)

Motivace
Výkon současných CPU
Blokové algoritmy
BLAS
Asymptotická složitost
Systémy lineárních rovnic
Přehled metod
Gaussova eliminace
Rozklady matic
LU dekompozice
Choleského dekompozice
QR dekompozice
Dekompozice na singulární

Motivace

Rekonstrukce signálu z ultrazvuku

- model vyšetření ultrazvukem
  - zdroj ultrazvuku, šíření a odraz vln v přístroji a tkáních
- pro výpočet šíření vln v objektu použijeme FEM
  - na vlnovou délku ultrazvuku potřebujeme několik lineárních elementů
  - velikost elementu je pak v desetinách mm
  - vyšetřovaný objekt má velikost v jednotkách až desítkách centimetrů
- systémy o jednotkách až desítkách miliónů rovnic
  - vysoké nároky na výpočetní kapacitu
  - vysoké paměťové nároky

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Výkon současných CPU

- taktování typicky 2–4 GHz
- 4–30 jader v socketu
- v jednom taktu na jednom jádru až 8 aritmetických operací ve `float`, tj. až 32 Gflop/s
- podmíněno dostatečným přísunem instrukcí a dat
- potřebný datový tok
  $4\,\text{GHz} \times 8\,\text{operací} \times 4\,\text{B} \times 20\,\text{jader} = 2,56\,\text{TB/s}$

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulárn

Vektorové instrukce

- historie – např. Cray v předsálí budovy D
- současné vektorové systémy (např. NEC SX)
  - řešení velmi specializovaných problémů
  - nízký výkon na skalárních operacích
- vektorová rozšíření v architektuře x86
  - MMX: pouze celá čísla, kolize s float registry
  - SSE: 8 (16) registrů po 128 bitech, postupně přidávané instrukce (rsqrt)
  - AVX*: 256–512 bitové registry, 3-operandové instrukce
- snažší využití plného výkonu procesoru

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Vektorové instrukce

- triviální příklad

```
        float a[4],b[4];
        int     i;
        for (i=0; i<4; i++) a[i] += b[i];
...
        movaps    32(%rsp), %xmm0
        addps     (%rsp), %xmm0
        movaps    %xmm0, 32(%rsp)
```

- ruční řešení (intrinsic funkce) je neportabilní, rychle zastarává
- vektorizaci kódu provede chytřejší kompilátor
  - musíme hlídat, abychom tomu nebránili
  - zarovnání dat, recyklace proměnných, vedlejší efekty in-line funkcí, ...
  - vyplatí se kontrola vygenerovaného assembleru

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Vektorové instrukce

- zarovnání dat

```
#define SIZ 200
#define DIM 3

float   x[SIZ][DIM],y[SIZ][DIM],z[SIZ][DIM];

...
        for (i=0; i<SIZ; i+=2) for (j=0; j<DIM; j++) {
                z[i][j] = x[i][j]*y[i][j];
                z[i+1][j] = x[i+1][j]-y[i+1][j];
        }
```

- nelze vektorizovat, vynucujeme skalární instrukce

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Vektorové instrukce

- zarovnání dat

```
#define SIZ 200
#define DIM 4

float   x[SIZ][DIM],y[SIZ][DIM],z[SIZ][DIM];

...
        for (i=0; i<SIZ; i+=2) for (j=0; j<DIM; j++) {
                z[i][j] = x[i][j]*y[i][j];
                z[i+1][j] = x[i+1][j]-y[i+1][j];
        }
```

- nelze vektorizovat, vynucujeme skalární instrukce
- přidání zbytečného výpočtu paradoxně celkově urychlí

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Paměť

- DDR4-3200, frekvence sběrnice 1600 MHz, 64 bitů, 2 přenosy v taktu … 25,6 GB/s
- 8 DIMMů na 4 kanálech CPU, teoreticky 204,8 GB/s
- nad možnosti paměťových řadičů

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Paměť

- DDR4-3200, frekvence sběrnice 1600 MHz, 64 bitů, 2 přenosy v taktu … 25,6 GB/s
- 8 DIMMů na 4 kanálech CPU, teoreticky 204,8 GB/s
- nad možnosti paměťových řadičů
- latence cca. 10 cyklů, tj. až 50 taktů CPU
  - dopředu musím vědět, co budu potřebovat

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Paměť

- DDR4-3200, frekvence sběrnice 1600 MHz, 64 bitů, 2 přenosy v taktu … 25,6 GB/s
- 8 DIMMů na 4 kanálech CPU, teoreticky 204,8 GB/s
- nad možnosti paměťových řadičů
- latence cca. 10 cyklů, tj. až 50 taktů CPU
  - dopředu musím vědět, co budu potřebovat
- paměť je podstatný zpomalující faktor
  - rychlou paměť v plné velikosti je technicky/finančně nemožné vyrobit
- hierarchie vyrovnávacích pamětí (cache)
- pro současné architektury Intel typicky
  - L1 – latence 4 cykly, 32 kB instrukce, 32 kB data/jádro
  - L2 – latence 12 cyklů, 256 kB/jádro
  - L3 – latence 28 cyklů, 8–40 MB/procesor (sdílená mezi jádry)

Vyrovnávací paměti

Organizace přístupu

- řádky (cache-line)
  - typicky 64 B (16× `float`, 8× `double`)
- přímo mapovaná cache
  - blok dat z hlavní paměti má dán jeden řádek v cache

    (adresa/velikost řádku) % počet řádků
  - snadno dojde ke kolizím, např. pro 32 kB

```
double pole[100][512][8];
for (i=0; i<100; i++) a += pole[i][0][0];
```

  - řešením je deklarace `pole[100][513][8]`

PA081: Programování numerických výpočtů
A. Křenek
Motivace
Výkon současných CPU
Blokové algoritmy
BLAS
Asymptotická složitost
Systémy lineárních rovnic
Přehled metod
Gaussova eliminace
Rozklady matic
LU dekompozice
Choleského dekompozice
QR dekompozice
Dekompozice na singulární

Vyrovnávací paměti

Organizace přístupu

- plně asociativní
  - blok dat z hlavní paměti může být umístěn v kterémkoli řádku
  - implementačně náročné, ve standardních CPU se nepoužívá
- n-cestná asociativní
  - kompromisní řešení
  - sady po n řádcích
  - blok z hlavní paměti může skončit v kterémkoli řádku sady
  - Intel Nehalem/Westmere/SandyBridge/...: $n = 8$

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Vyrovnávací paměti

Praktické zásady

- optimalizovat pro všechny úrovně
- využít celý řádek
  - např. může se vyplatit transponovat matici
- dovolit procesoru/kompilátoru současně přenášet data a počítat
  - dobře čitelný kód bez možných vedlejších efektů
- zabránit předčasným kolizím v jedné sadě řádků
  - diagnostické nástroje, včetně HW podpory
- měřit dosažený výkon (flop/s)
- atd. . . .

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Vyrovnávací paměti

Praktické zásady

- optimalizovat pro všechny úrovně
- využít celý řádek
  - např. může se vyplatit transponovat matici
- dovolit procesoru/kompilátoru současně přenášet data a počítat
  - dobře čitelný kód bez možných vedlejších efektů
- zabránit předčasným kolizím v jedné sadě řádků
  - diagnostické nástroje, včetně HW podpory
- měřit dosažený výkon (flop/s)
- atd. . . .
- aplikovat jen pro skutečně kritické sekce kódu
- používat speciální optimalizované knihovny, kde to jde

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Blokové algoritmy

- zjednodušený model jen s L1
- násobení vektoru maticí $\mathbf{y} := \mathbf{y} + \mathbf{A}\mathbf{x}$

načti $\mathbf{x}$ do cache
načti $\mathbf{y}$ do cache
`for` $i = 1, \ldots, n$
    načti řádek A_{i*} do cache
    `for` $j = 1, \ldots, n$
        $y_i = y_i + A_{ij}x_j$
zapiš $\mathbf{y}$ do hlavní paměti

- počet přístupů do pomalé paměti $m = 3n + n^2$
- počet aritmetických operací $f = 2n^2$
- efektivita $f/m \approx 2$
- algoritmus je limitovaný rychlostí paměti
  - pomůže recyklace řádků cache – vyplatí se ukládat vektory spojitě, ne jako sloupce matice (v C)
  - jinak se s tím nedá nic moc dělat

Motivace
Výkon současných CPU
Blokové algoritmy
BLAS
Asymptotická složitost
Systémy lineárních rovnic
Přehled metod
Gaussova eliminace
Rozklady matic
LU dekompozice
Choleského dekompozice
QR dekompozice
Dekompozice na singulární

Blokové algoritmy

Maticové násobení

- násobení matic $\mathbf{C} = \mathbf{C} + \mathbf{AB}$

```
for i = 1,...,n
    načti řádek A_{i*} do cache
    for j = 1,...,n
        načti C_{ij} do cache
        načti sloupec B_{*j} do cache
        for k = 1,...,n
            C_{ij} = C_{ij} + A_{ik}B_{kj}
        zapiš C_{ij} do hlavní paměti
```

- přístupy do paměti $m = n^2 + n^3 + 2n^2$
- aritmetické operace $f = 2n^3$
- efektivita opět ≈ 2

Blokové algoritmy

Maticové násobení

- matice rozdělíme na N^2 bloků velikosti $b \times b, b = n/N$

```
for i = 1,...,N
    for j = 1,...,N
        načti blok C_ij do cache
        for k = 1,...,N
            načti blok A_ik do cache
            načti blok B_kj do cache
            C_ij = C_ij + A_ik B_kj (maticové násobení)
        zapiš blok C_ij do cache
```

- přístupy do paměti
  - čtení bloků **A**: $N^3(n/N)^2 = N * n^2$
  - dtto **B**: $N * n^2$
  - čtení a zápis **C**: $2n^2$
- efektivita $f/m = \frac{2n^3}{(2N+2)n^2} \approx b$

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Blokové algoritmy

Maticové násobení

- velikost reálné L1 cache je 32 kB,
- do L1 se vejdou 3 matice velikosti 36×36 (v `double`)
- L2 cache je $10\times$ pomalejší
- procesor je dobře využit
  - i při využití vektorových instrukcí
  - proto je cache právě tak velká
- žádoucí aplikovat rekurzivně i pro L2 a L3
- implementováno v optimalizovaných knihovnách
- násobení matic je na současných procesorech jeden z nejefektivnějších algoritmů
  - redukce problému na mat. násobení při řádovém zachování počtu operací téměř vždy vede k výraznému zrychlení

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Knihovna BLAS

- *Basic Linear Algebra Subprograms*, `http://www.netlib.org/blas`
- základní operace, např.
  - `DOT`: $\mathbf{xy}$
  - `AXPY`: $\mathbf{y} + \mathbf{Ax}$
  - `NRM2`: $|\mathbf{x}|^2$
  - `TRMV`: $\mathbf{Ax}$
  - `TRSV`: $\mathbf{A}^{-1}\mathbf{x}$
  - `GEMM`: $\beta\mathbf{C} + \alpha\mathbf{AB}$
- verze pro typy `float`,`double`,`complex`
- specializované varianty pro symetrické, trojúhelníkové, a některé řídké matice
- de-facto standardizované rozhraní
- implementace vyladěné pro konkrétní typy CPU
- využívané v knihovnách vyšší úrovně (např. LU dekompozice)

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Asymptotická složitost

- časová složitost násobení matic je $O(n^3)$

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Asymptotická složitost

- časová složitost násobení matic je $O(n^3)$
- Strassen (1969): $O(n^{2.81})$
- rozdělení matic na 2×2 bloky, potom

$$\mathbf{M}_1 = (\mathbf{A}_{11} + \mathbf{A}_{22})(\mathbf{B}_{11} + \mathbf{B}_{22})$$
$$\mathbf{M}_2 = (\mathbf{A}_{21} + \mathbf{A}_{22})\mathbf{B}_{11}$$
$$\mathbf{M}_3 = \mathbf{A}_{11}(\mathbf{B}_{12} + \mathbf{B}_{22})$$
$$\mathbf{M}_4 = \mathbf{A}_{22}(\mathbf{B}_{21} - \mathbf{B}_{11})$$
$$\mathbf{M}_5 = (\mathbf{A}_{11} + \mathbf{A}_{12})\mathbf{B}_{22}$$
$$\mathbf{M}_6 = (\mathbf{A}_{21} - \mathbf{A}_{11})(\mathbf{B}_{11} + \mathbf{B}_{12})$$
$$\mathbf{M}_7 = (\mathbf{A}_{12} - \mathbf{A}_{22})(\mathbf{B}_{21} + \mathbf{B}_{22})$$

$$\mathbf{C}_{11} = \mathbf{M}_1 + \mathbf{M}_4 - \mathbf{M}_5 + \mathbf{M}_7$$
$$\mathbf{C}_{12} = \mathbf{M}_3 + \mathbf{M}_5$$
$$\mathbf{C}_{21} = \mathbf{M}_2 + \mathbf{M}_4$$
$$\mathbf{C}_{22} = \mathbf{M}_1 - \mathbf{M}_2 + \mathbf{M}_3 + \mathbf{M}_6$$

- rekurzivní aplikace, počet operací $T(n)$

$$T(n) = 7T(n/2) + 18(n/2)^2 = O(n^{\log_2 7}) = O(n^{2.81})$$

Motivace
Výkon současných CPU
Blokové algoritmy
BLAS
Asymptotická složitost
Systémy lineárních rovnic
Přehled metod
Gaussova eliminace
Rozklady matic
LU dekompozice
Choleského dekompozice
QR dekompozice
Dekompozice na singulární

Asymptotická složitost

- časová složitost násobení matic je $O(n^3)$
- Strassen (1969): $O(n^{2.81})$
- rozdělení matic na 2×2 bloky, potom

$$\mathbf{M}_1 = (\mathbf{A}_{11} + \mathbf{A}_{22})(\mathbf{B}_{11} + \mathbf{B}_{22})$$
$$\mathbf{M}_2 = (\mathbf{A}_{21} + \mathbf{A}_{22})\mathbf{B}_{11}$$
$$\mathbf{M}_3 = \mathbf{A}_{11}(\mathbf{B}_{12} + \mathbf{B}_{22})$$
$$\mathbf{M}_4 = \mathbf{A}_{22}(\mathbf{B}_{21} - \mathbf{B}_{11})$$
$$\mathbf{M}_5 = (\mathbf{A}_{11} + \mathbf{A}_{12})\mathbf{B}_{22}$$
$$\mathbf{M}_6 = (\mathbf{A}_{21} - \mathbf{A}_{11})(\mathbf{B}_{11} + \mathbf{B}_{12})$$
$$\mathbf{M}_7 = (\mathbf{A}_{12} - \mathbf{A}_{22})(\mathbf{B}_{21} + \mathbf{B}_{22})$$

$$\mathbf{C}_{11} = \mathbf{M}_1 + \mathbf{M}_4 - \mathbf{M}_5 + \mathbf{M}_7$$
$$\mathbf{C}_{12} = \mathbf{M}_3 + \mathbf{M}_5$$
$$\mathbf{C}_{21} = \mathbf{M}_2 + \mathbf{M}_4$$
$$\mathbf{C}_{22} = \mathbf{M}_1 - \mathbf{M}_2 + \mathbf{M}_3 + \mathbf{M}_6$$

- rekurzivní aplikace, počet operací $T(n)$

$$T(n) = 7T(n/2) + 18(n/2)^2 = O(n^{\log_2 7}) = O(n^{2.81})$$

- Coppersmith-Winograd (1987): $O(n^{2.376})$

Motivace
Výkon současných CPU
Blokové algoritmy
BLAS
Asymptotická složitost
Systémy lineárních rovnic
Přehled metod
Gaussova eliminace
Rozklady matic
LU dekompozice
Choleského dekompozice
QR dekompozice
Dekompozice na singulární

Systémy lineárních rovnic

- hledáme řešení systému

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1N}x_N &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2N}x_N &= b_2 \\ &\cdots \\ a_{M1}x_1 + a_{M2}x_2 + \cdots + a_{MN}x_N &= b_M \end{aligned}$$

- maticové vyjádření

$$\mathbf{Ax} = \mathbf{b}$$

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Klasifikace problémů

- $M = N$ – dobře podmíněné systémy
  - naděje na jednoznačné řešení (je-li $\mathbf{A}$ regulární)
- $M < N$ – nedostatečně podmíněné systémy
  - žádné řešení
  - nekonečně mnoho řešení ($N - M$-rozměrný prostor)
- $M > N$ – příliš podmíněné systémy
  - přesné řešení zpravidla neexistuje
  - hledáme „nejlepší kompromis“

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Klasifikace problémů

- $M = N$ – dobře podmíněné systémy
  - naděje na jednoznačné řešení (je-li $\mathbf{A}$ regulární)
- $M < N$ – nedostatečně podmíněné systémy
  - žádné řešení
  - nekonečně mnoho řešení ($N - M$-rozměrný prostor)
- $M > N$ – příliš podmíněné systémy
  - přesné řešení zpravidla neexistuje
  - hledáme „nejlepší kompromis“

- husté vs. řídké
  - $O(N^2)$ vs. $O(N)$ nenulových prvků

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Klasifikace problémů

- $M = N$ – dobře podmíněné systémy
  - naděje na jednoznačné řešení (je-li $\mathbf{A}$ regulární)
- $M < N$ – nedostatečně podmíněné systémy
  - žádné řešení
  - nekonečně mnoho řešení ($N - M$-rozměrný prostor)
- $M > N$ – příliš podmíněné systémy
  - přesné řešení zpravidla neexistuje
  - hledáme „nejlepší kompromis“

- husté vs. řídké
  - $O(N^2)$ vs. $O(N)$ nenulových prvků

- velikost – dopady kumulace chyby při naivním přístupu
  - $N \sim 10$ – zpravidla stačí `float`
  - $N \sim 50$ – `double`
  - větší – vyžadují chytřejší metody (pivoting)

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Přehled metod

- přímé
  - daný algoritmus, vždy stejný počet operací
  - nemusí fungovat dobře pro „skoro singulární“ nebo příliš velké systémy
  - preferované pro „normální“ problémy
  - Gaussova eliminace, různé dekompozice, …
- iterační
  - postupně vylepšované řešení
  - dosažení kritéria konvergence
  - různá náročnost pro různé vstupy
  - Jacobi, Gauss-Seidel, sdružené směry …

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Přehled metod

- specializované metody pro řídké matice
  - pro různé vzory řídkých matic
  - výrazně efektivnější, jediná možnost řešení velkých systémů
  - přímé i iterační
- metody pro singulární systémy
  - analyticky i numericky („skoro“) singulární
  - nalezení celého prostoru řešení
  - standardní metody zhavarují
  - dekompozice na singulární hodnoty
- řešení příliš podmíněných systémů
  - specializované metody nejmenších čtverců

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Dostupné knihovny

- zpravidla sáhneme k hotové knihovně
  - nejsme první, kdo tento problém řeší
  - implementace optimalizované pro různé případy
  - k volbě vhodné metody je třeba rámcově rozumět jejich principům
- Numerical Recipes in C
  - hlavní zdroj pro tuto přednášku
  - jednoduché přehledné implementace
- LAPACK `http://www.netlib.org/lapack/`
  - plnohodnotná volně dostupná knihovna
  - využívá nízkoúrovňové knihovny BLAS, zpravidla implementované strojově závisle
  - obecné a specializované funkce pro různé typické případy
- numpy.linalg a scipy.linarg `https://scipy.org/`
  - Python API, interně využívá zpravidla BLAS/LAPACK
- NAG `http://www.nag.co.uk/`
  - rozsáhlá komerční numerická knihovna
- a další ...

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Gaussova eliminace

Základní postup

- standardní „školní“ metoda
  - řešení systému se nezmění, nahradíme-li libovolný řádek **A** a odpovídající prvek **b** lineární kombinací tohoto řádku s libovolným dalším
- vynulujeme $a_{21}, \ldots, a_{M1}$ odečtením $\frac{a_{i1}}{a_{11}}$ násobku prvního řádku
- obdobně pokračujeme druhým sloupcem od a_{32}
- výsledkem je matice s vynulovanými prvky pod diagonálou
- řešení systému získáme zpětnou substitucí
  - poslední rovnice $a_{MM}x_M = b_M$ je triviální
  - do předposlední dosadíme získanou hodnotu x_M atd.

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Gaussova eliminace

Numerické problémy

- diagonální prvek a_{kk} je 0 nebo příliš malý
  - v k-té iteraci se jím dělí
  - dojde k přetečení
- při odečítání dochází k přílišné ztrátě platných cifer

$$\begin{bmatrix} \epsilon & 1 \\ 1 & 1 \end{bmatrix} \Rightarrow \begin{bmatrix} \epsilon & 1 \\ 0 & 1 - \frac{1}{\epsilon} \end{bmatrix}$$

  zaokrouhlení může vést až k $a_{22} = -\frac{1}{\epsilon}$, to odpovídá původní matici

$$\begin{bmatrix} \epsilon & 1 \\ 1 & 0 \end{bmatrix}$$

- metoda v této podobě je numericky nestabilní

Gaussova eliminace

Pivoting

- další tvrzení o systému
  - řešení systému se nezmění výměnou libovolné dvojice řádků
  - řešení systému se nezmění výměnou libovolné dvojice sloupců, zaměníme-li zároveň příslušné komponenty vektoru $\mathbf{x}$
- pivot je prvek, který po aplikaci těchto kroků použijeme k dělení při eliminaci k-tého sloupce
- částečný pivoting (parciální)
  - v iteraci k vybíráme z a_{jk} pro $j \geq k$, tj. jen z nulovaného sloupce
- plný pivoting
  - vybíráme z a_{ij} pro $i, j \geq k$, tj. z celé podmatice doprava a dolů
  - vyžaduje udržovat vznikající permutaci proměnných

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Gaussova eliminace

Pivoting

- za pivota volíme maximum z kandidátů
  - pro všechny faktory v eliminačních krocích platí $|\frac{a_{ik}}{a_{kk}}| \leq 1$
- parciální i plný pivoting jsou pak numericky stabilní
- přínos plného pivotingu je jen okrajový

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Gaussova eliminace

Pivoting

- za pivota volíme maximum z kandidátů
  - pro všechny faktory v eliminačních krocích platí $|\frac{a_{ik}}{a_{kk}}| \leq 1$
- parciální i plný pivoting jsou pak numericky stabilní
- přínos plného pivotingu je jen okrajový
- volba pivota závisí na řádu koeficientů původního systému
  - vynásobení jedné rovnice faktorem 10^{10} ...
  - za pivota lze volit koeficient, který by byl největší, kdyby byl celý původní systém normalizovaný, tj. největší koeficient všech rovnic roven 1
  - vyžaduje dodatečnou údržbu faktorů škálování, může přinést větší robustnost

Gaussova eliminace

Gauss-Jordanova eliminace

- Gaussovu metodu lze použít pro více pravých stran současně
- speciálně pro N bázových vektorů dostaneme výpočet matice $\mathbf{A}^{-1}$
- rozšíření – Gauss-Jordanova eliminace
  - v každé iteraci eliminujeme prvky nad i pod diagonálou $\mathbf{A}$
  - výsledkem je jednotková matice
  - řešení systému je přímo modifikovaný vektor $\mathbf{b}$
  - resp. dostáváme $\mathbf{A}^{-1}$
- srovnatelné se základní implementací
  - reálně provádíme tytéž operace

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Gaussova eliminace

Výhody a nevýhody

- jednoduchá, dobře pochopitelná a stabilní metoda
- je třeba znát pravé strany dopředu
  - jsou součástí celého výpočtu
- výpočet $\mathbf{A}^{-1}$ je zatížen poměrně velkou numerickou chybou
  - přímý výpočet řešení systému $\mathbf{Ax} = \mathbf{b}$ je přesnější než výpočet $\mathbf{A}^{-1}$ a následné $\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Rozklady matic

- danou matici $\mathbf{A}$ vyjádříme jako součin

$$\mathbf{A} = \mathbf{A}_1 \mathbf{A}_2 \ldots \mathbf{A}_n$$

- matice $\mathbf{A}_1, \mathbf{A}_2, \ldots \mathbf{A}_n$ mají nějaké speciální vlastnosti
- zřejmější vlastnosti problému popsaného původní $\mathbf{A}$
  - kritické body vektorového pole
  - statisticky významné vlastnosti nějakého jevu
  - podmíněnost systému lineárních rovnic
  - ...
- vlastnosti rozkladu lze prakticky využít
  - řešení systému lineárních rovnic
- existují implementace algoritmů se známými vlastnostmi
  - zpravidla numericky stabilní
  - optimalizované na konkrétní CPU, maximálně efektivní

Motivace
Výkon současných CPU
Blokové algoritmy
BLAS
Asymptotická složitost
Systémy lineárních rovnic
Přehled metod
Gaussova eliminace
Rozklady matic
LU dekompozice
Choleského dekompozice
QR dekompozice
Dekompozice na singulár

Rozklady matic

- $\mathbf{A} = \mathbf{LU}$, resp. $\mathbf{PA} = \mathbf{LU}$
  - $\mathbf{L}$ a $\mathbf{U}$ jsou dolní a horní trojúhelníková
  - $\mathbf{P}$ je permutace řádků
- Choleského: $\mathbf{A} = \mathbf{L}\mathbf{L}^T$
  - pro symetrickou pozitivně definitní $\mathbf{A}$
- $\mathbf{A} = \mathbf{QR}$
  - $\mathbf{Q}$ ortogonální, $\mathbf{R}$ horní trojúhelníková
  - symetrické varianty RQ, QL a LQ
- singulární hodnoty: $\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}$
  - $\mathbf{U}, \mathbf{V}$ ortogonální, Σ diagonální
- spektrální: $\mathbf{A} = \mathbf{V}\Lambda\mathbf{V}^{-1}$
  - Λ diagonální, vlastní hodnoty
  - varianta Jordanova: blokově diagonální Λ, násobné vlastní hodnoty
- Shurova: $\mathbf{A} = \mathbf{VSV}^T$
  - $\mathbf{V}$ ortogonální
  - $\mathbf{S}$ horní trojúhelníková s vlastními hodnotami $\mathbf{A}$ na diagonále

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

LU dekompozice

Princip

- předpokládejme, že dokážeme rozložit $\mathbf{A} = \mathbf{LU}$ tak, že
  - $\mathbf{L}$ je spodní trojúhelníková matice (prvky nad diagonálou jsou nulové)
  - $\mathbf{U}$ je horní trojúhelníková matice (prvky pod diagonálou jsou nulové)
- potom lze psát

$$\mathbf{Ax} = (\mathbf{LU})\mathbf{x} = \mathbf{L}(\mathbf{Ux}) = \mathbf{b}$$

- původní systém lze vyřešit postupným řešením

$$\mathbf{Ly} = \mathbf{b} \qquad \text{a} \qquad \mathbf{Ux} = \mathbf{y}$$

- to je triviální dopřednou a zpětnou substitucí

LU dekompozice

Výpočet rozkladu

- prvky rozkladu $\mathbf{A} = \mathbf{LU}$ lze, s ohledem na nuly psát

$$
\begin{aligned}
i < j &: a_{ij} = u_{i1}l_{1j} + u_{i2}l_{2j} + \cdots + u_{ii}l_{ij} \\
i = j &: a_{ij} = u_{i1}l_{1j} + u_{i2}l_{2j} + \cdots + u_{ii}l_{jj} \\
i > j &: a_{ij} = u_{i1}l_{1j} + u_{i2}l_{2j} + \cdots + u_{ij}l_{jj}
\end{aligned}
$$

- je to systém N^2 rovnic v $N^2 + N$ neznámých
  - diagonála je pokryta u i l
  - můžeme tedy volit $l_{ii} = 1$

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

LU dekompozice

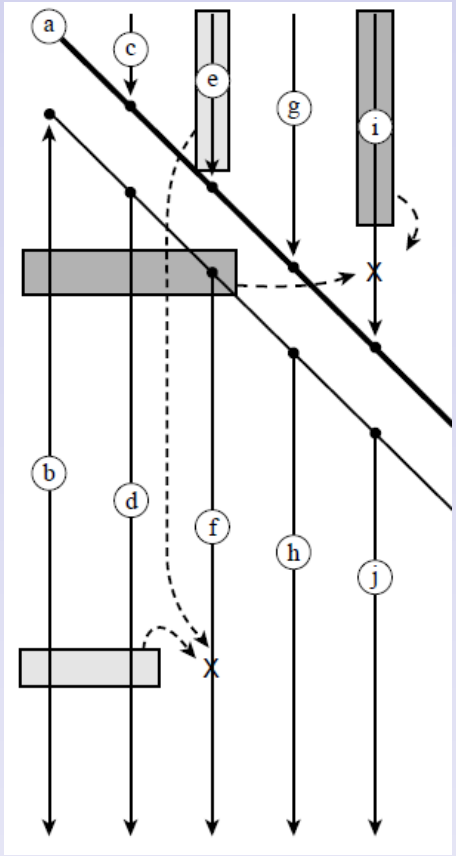
Croutův algoritmus

- postupně pro $j = 1, 2, \ldots, N$:
  - pro $i = 1, 2, \ldots j$ spočítáme na základě uvedených rovnic

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} l_{ik} u_{kj}$$

  - pro $i = j+1, j+2, \ldots, N$

$$l_{ij} = \frac{1}{u_{jj}} \left(a_{ij} - \sum_{k=1}^{j-1} l_{ik} u_{kj} \right)$$



- postup vždy využívá dříve spočtené prvky
- k numerické stabilitě je třeba navíc dodat pivoting l_{jj}

Motivace
Výkon současných CPU
Blokové algoritmy
BLAS
Asymptotická složitost
Systémy lineárních rovnic
Přehled metod
Gaussova eliminace
Rozklady matic
LU dekompozice
Choleského dekompozice
QR dekompozice
Dekompozice na singulární

LU dekompozice

Shrnutí a použití

- řešení lineárních rovnic pro libovolný počet **b**
  - Croutův algoritmus $O(N^3)$
  - dopředná a zpětná substituce $O(N^2)$
  - všechna **b** není třeba znát dopředu – hlavní přednost metody
- výpočet inverzní matice
  - řešení systému pro **b** = bázové vektory
  - potřebujeme-li počítat $\mathbf{A}^{-1}\mathbf{B}$, je lepší přímo pro sloupce **B** než explicitní vyjádření $\mathbf{A}^{-1}$

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

LU dekompozice

Shrnutí a použití

- tzv. driver routine v knihovně LAPACK
  - vlastní rozklad LU
  - řešení systému
  - iterační zpřesnění (viz dále)

```
int n = N, nrhs = NRHS, lda = LDA, ldb = LDB, info;
int ipiv[N];
float a[LDA*N] = { ... };
float b[LDB*NRHS] = { ... };

sgesv( &n, &nrhs, a, &lda, ipiv, b, &ldb, &info );
```

- kompilace `icc -mkl source.c`
- existují i volné verze

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Choleského dekompozice

- předpokládáme $\mathbf{A} = \mathbf{L}\mathbf{L}^T$
- po prvcích platí

$$l_{ii} = \sqrt{a_{ii} - \sum_{k=0}^{i-1} l_{ik}^2} \qquad l_{ji} = \frac{1}{l_{ii}} \left(a_{ij} - \sum_{k=0}^{i-1} l_{ik} l_{jk} \right)$$

- podobně jako při LU dekompozici vždy využíváme dříve spočtené prvky
- odmocninu lze vždy spočítat pro symetrickou pozitivně definitní $\mathbf{A}$
  - omezenější, ale stále významná třída problémů
- algoritmus je efektivnější a numericky stabilní
  - cca. $2\times$ méně operací než LU, menší paměťová náročnost
  - má smysl používat speciální funkce z knihoven

Motivace
Výkon současných CPU
Blokové algoritmy
BLAS
Asymptotická složitost
Systémy lineárních rovnic
Přehled metod
Gaussova eliminace
Rozklady matic
LU dekompozice
Choleského dekompozice
QR dekompozice
Dekompozice na singulární

QR dekompozice

- rozklad $\mathbf{A} = \mathbf{QR}$
  - $\mathbf{Q}$ ortogonální, $\mathbf{R}$ trojúhelníková
- systém $\mathbf{Ax} = \mathbf{b}$ lze psát

$$\mathbf{Rx} = \mathbf{Q}^T\mathbf{b}$$

- jednoduché řešení substitucí
- lepší numerické vlastnosti
- metody konstrukce – nulování prvků pod diagonálou
  - Gram-Schmidtův proces
  - Householderovy transformace
  - Givensovy rotace

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

QR dekompozice

Gram-Schmidtův proces

- Gram-Schmidtův ortogonalizační proces

$$\mathbf{u}_1 = \mathbf{a}_1 \qquad \mathbf{v}_1 = \frac{\mathbf{u}_1}{\|\mathbf{u}_1\|}$$

$$\mathbf{u}_2 = \mathbf{a}_2 - \mathrm{proj}_{\mathbf{v}_1}\mathbf{a}_2 \qquad \mathbf{v}_2 = \frac{\mathbf{u}_2}{\|\mathbf{u}_2\|}$$

$$\mathbf{u}_3 = \mathbf{a}_3 - \mathrm{proj}_{\mathbf{v}_1}\mathbf{a}_3 - \mathrm{proj}_{\mathbf{v}_2}\mathbf{a}_3 \qquad \mathbf{v}_3 = \frac{\mathbf{u}_3}{\|\mathbf{u}_3\|}$$

$$\vdots$$

- potom

$$\mathbf{Q} = (\mathbf{v}_1 \dots \mathbf{v}_n) \qquad \mathbf{R} = \begin{pmatrix} \mathbf{v}_1 \cdot \mathbf{a}_1 & \mathbf{v}_1 \cdot \mathbf{a}_2 & \dots \\ 0 & \mathbf{v}_2 \cdot \mathbf{a}_2 & \dots \\ \dots & & \end{pmatrix}$$

- numerický problém, jsou-li některé $\mathbf{a}_i, \mathbf{a}_j$ skoro kolmé

QR dekompozice

Householderova transformace

- zrcadlení daného vektoru podle nadroviny
  - zarovná s bázovým vektorem, zachová velikost
- pro libovolné $\mathbf{x}$:

$$\mathbf{u} = \mathbf{x} + \alpha \mathbf{e}_1 \qquad \mathbf{v} = \frac{\mathbf{u}}{\|\mathbf{u}\|} \qquad \mathbf{Q} = I - 2\mathbf{v}\mathbf{v}^T$$

- potom $\mathbf{Q}\mathbf{x} = (\alpha, 0, \ldots, 0)^T$

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulárn

QR dekompozice

Householderova transformace

- zrcadlení daného vektoru podle nadroviny
  - zarovná s bázovým vektorem, zachová velikost
- pro libovolné $\mathbf{x}$:

$$\mathbf{u} = \mathbf{x} + \alpha \mathbf{e}_1 \qquad \mathbf{v} = \frac{\mathbf{u}}{\|\mathbf{u}\|} \qquad \mathbf{Q} = I - 2\mathbf{v}\mathbf{v}^T$$

- potom $\mathbf{Q}\mathbf{x} = (\alpha, 0, \ldots, 0)^T$
- triangulace $\mathbf{A}$

$$\mathbf{R} = \mathbf{Q}_{n-1} \ldots \mathbf{Q}_1 \mathbf{A} \qquad \text{a tedy} \qquad \mathbf{Q} = \mathbf{Q}_1^T \mathbf{Q}_2^T \ldots \mathbf{Q}_{n-1}^T$$

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Základní dekompozice – shrnutí

- LU
  - ve variantě **PA** = **LU** existuje pro všechny čtvercové matice
  - odpovídá Gaussově eliminaci – trpí stejnými numerickými problémy
- Choleského
  - čtvercové, symetrické, pozitivně definitní matice
  - numericky stabilní (to je ale LU pro tyto matice také)
  - cca. $2\times$ méně operací
- QR
  - obecné $m \times n$ matice
  - existují numericky stabilní konstrukce (Householderova transformace)
  - výpočetně náročnější
- použití především k řešení systémů lineárních rovnic

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Dekompozice na singulární hodnoty

Základní tvrzení

- libovolnou reálnou (i komplexní) matici lze rozložit

$$\mathbf{A}_{M\times N} = \mathbf{U}_{M\times N} \cdot \Sigma_{N\times N} \cdot \mathbf{V}^T_{N\times N} \quad (= \mathbf{U}'_{M\times M} \cdot \Sigma'_{M\times N} \cdot \mathbf{V'}^T_{N\times N})$$

- $\mathbf{U}$ je sloupcově ortogonální
  - až na nulové sloupce v případě $M \leq N$
- Σ diagonální a $\mathbf{V}$ ortogonální
- rozklad je unikátní až na
  - současnou perumutaci sloupců všech tří matic
  - lineární kombinaci sloupců $\mathbf{U}, \mathbf{V}$ odpovídajících nulovým σ_i

Motivace
Výkon současných CPU
Blokové algoritmy
BLAS
Asymptotická složitost
Systémy lineárních rovnic
Přehled metod
Gaussova eliminace
Rozklady matic
LU dekompozice
Choleského dekompozice
QR dekompozice
Dekompozice na singulární

Dekompozice na singulární hodnoty

Geometrický význam

- $\mathbf{A}$ je složení transformací
  - rotace/zrcadlení $\mathbf{V}^{-1}$
  - zvětšení/zmenšení faktory σ_i ve směrech e_i, včetně degenerace ($\sigma_i = 0$)
  - rotace/zrcadlení a projekce do méně/více dimenzí $\mathbf{U}$

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Dekompozice na singulární hodnoty

Geometrický význam

- $\mathbf{A}$ je složení transformací
  - rotace/zrcadlení $\mathbf{V}^{-1}$
  - zvětšení/zmenšení faktory σ_i ve směrech e_i, včetně degenerace ($\sigma_i = 0$)
  - rotace/zrcadlení a projekce do méně/více dimenzí $\mathbf{U}$
- obor hodnot zobrazení $\mathbf{A}$
  - sloupce $\mathbf{U}$ odpovídající nenulovým σ_i jsou jeho generátory

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Dekompozice na singulární hodnoty

Geometrický význam

- $\mathbf{A}$ je složení transformací
  - rotace/zrcadlení $\mathbf{V}^{-1}$
  - zvětšení/zmenšení faktory σ_i ve směrech e_i, včetně degenerace ($\sigma_i = 0$)
  - rotace/zrcadlení a projekce do méně/více dimenzí $\mathbf{U}$
- obor hodnot zobrazení $\mathbf{A}$
  - sloupce $\mathbf{U}$ odpovídající nenulovým σ_i jsou jeho generátory
- nulový prostor $\mathcal{N}(\mathbf{A}) = \{\mathbf{x} \in \mathbb{R}^N : \mathbf{A}\mathbf{x} = 0\}$
  - řádky $\mathbf{V}^T$ odpovídající nulovým σ_i jsou jeho generátory

Motivace
Výkon současných CPU
Blokové algoritmy
BLAS
Asymptotická složitost
Systémy lineárních rovnic
Přehled metod
Gaussova eliminace
Rozklady matic
LU dekompozice
Choleského dekompozice
QR dekompozice
Dekompozice na singulární

Dekompozice na singulární hodnoty

Numerický význam

- „oddělení zrna od plev“
- sloupce **U** a **V** jsou kolmé a normované
- veškeré potenciální degenerace soustředěny do Σ
  - singularity **A** odpovídají nulovým σ_i
  - včetně numerických ($\sigma_i \approx 0$)
- numericky velmi stabilní algoritmus dekompozice
- lze použít na řešení systémů lineárních rovnic
  - $M < N$ a $M = N$ singulární: reprezentant řešení + generátor prostoru
  - $M > N$: nejbližší řešení

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Dekompozice na singulární hodnoty

Použití pro $M = N$

- řešení systému rovnic, resp. výpočet inverzní matice

$$\mathbf{A}^{-1} = \mathbf{V}[\mathrm{diag}(1/\sigma_i)]\mathbf{U}^T$$

- kdy to nejde
  - jedno nebo více σ_i je nulových – A byla singulární
  - $\sigma_{\min}/\sigma_{\max} < \epsilon$ (špatně podmíněná matice) – standardní metody řešení selhaly

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Dekompozice na singulární hodnoty

Použití pro $M = N$

- řešení systému rovnic, resp. výpočet inverzní matice

$$\mathbf{A}^{-1} = \mathbf{V}[\mathrm{diag}(1/\sigma_i)]\mathbf{U}^T$$

- kdy to nejde
  - jedno nebo více σ_i je nulových – A byla singulární
  - $\sigma_{\min}/\sigma_{\max} < \epsilon$ (špatně podmíněná matice) – standardní metody řešení selhaly
- označme

$$\Sigma' = \left[\mathrm{diag} \begin{cases} 1/\sigma_i & \sigma_i \neq 0 \\ 0 & \sigma_i = 0 \end{cases} \right]$$

- rovnice $\mathbf{Ax} = \mathbf{b}$ nemusí mít řešení, přesto zkusíme $\mathbf{x} = \mathbf{V}\Sigma'\mathbf{U}^T\mathbf{b}$

Dekompozice na singulární hodnoty

Použití pro $M = N$

- hledáme nejbližší řešení, tj. minimalizujeme $|\mathbf{Ax} - \mathbf{b}|$
- pro libovolné $\mathbf{x}'$ je $\mathbf{A}(\mathbf{x}+\mathbf{x}') - \mathbf{b} = \mathbf{Ax} - \mathbf{b} + \mathbf{b}'$, kde $\mathbf{b}' = \mathbf{Ax}'$

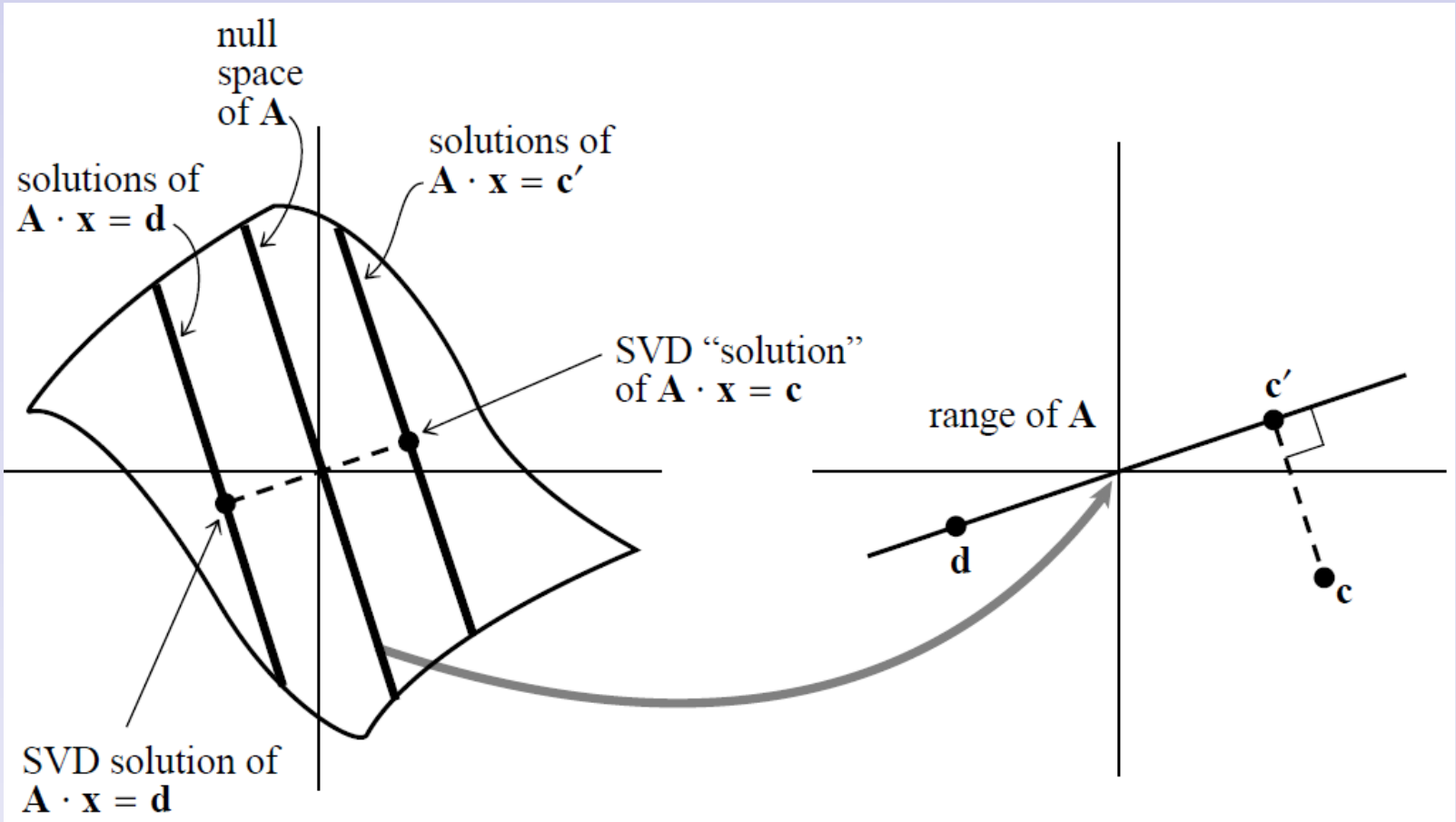
$$\begin{aligned}
|\mathbf{Ax} - \mathbf{b} + \mathbf{b}'| &= |(\mathbf{U}\Sigma\mathbf{V}^T)(\mathbf{V}\Sigma'\mathbf{U}^T\mathbf{b}) - \mathbf{b} + \mathbf{b}'| \\
&= |(\mathbf{U}\Sigma\Sigma'\mathbf{U}^T - I)\mathbf{b} + \mathbf{b}'| \\
&= |\mathbf{U}((\Sigma\Sigma' - I)\mathbf{U}^T\mathbf{b} + \mathbf{U}^T\mathbf{b}')| \\
&= |(\Sigma\Sigma' - I)\mathbf{U}^T\mathbf{b} + \mathbf{U}^T\mathbf{b}'|
\end{aligned}$$

- $\Sigma\Sigma' - I$ je diagonální s nenulovými prvky pro $\sigma_i = 0$
- $\mathbf{b}'$ je v oboru hodnot $\mathbf{A}$, tedy $\mathbf{U}^T\mathbf{b}'$ má nenulové prvky právě pro $\sigma_i \neq 0$
- minimum právě pro $\mathbf{b}' = 0$ a tedy i $\mathbf{x}' = 0$

Motivace
Výkon současných CPU
Blokové algoritmy
BLAS
Asymptotická složitost
Systémy lineárních rovnic
Přehled metod
Gaussova eliminace
Rozklady matic
LU dekompozice
Choleského dekompozice
QR dekompozice
Dekompozice na singulární

Dekompozice na singulární hodnoty

Použití pro $M = N$



Motivace
Výkon současných CPU
Blokové algoritmy
BLAS
Asymptotická složitost
Systémy lineárních rovnic
Přehled metod
Gaussova eliminace
Rozklady matic
LU dekompozice
Choleského dekompozice
QR dekompozice
Dekompozice na singulární

Dekompozice na singulární hodnoty

Použití pro $M = N$ – prakticky

- singularitu $\mathbf{A}$ detekujeme podle $\sigma_i = 0$
- vypočteme nejbližší řešení jako $\mathbf{x} = \mathbf{V}\Sigma'\mathbf{U}^T\mathbf{b}$
- dosazením ověříme, zda je to přesné řešení
  - když ne, víme, že přesné řešení neexistuje
  - máme nejbližší aproximaci

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Dekompozice na singulární hodnoty

Použití pro $M = N$ – prakticky

- singularitu $\mathbf{A}$ detekujeme podle $\sigma_i = 0$
- vypočteme nejbližší řešení jako $\mathbf{x} = \mathbf{V}\Sigma'\mathbf{U}^T\mathbf{b}$
- dosazením ověříme, zda je to přesné řešení
  - když ne, víme, že přesné řešení neexistuje
  - máme nejbližší aproximaci
- špatně podmíněná matice $|\sigma_{\max}| \gg |\sigma_{\min}|$
  - lépe v Σ' vynulovat i taková σ_i
  - paradoxní – zahazujeme část vstupní informace
  - v praxi dává lepší výsledky – právě tento vstup má tendenci škodit
  - stanovení prahu $\sigma_i \approx 0$ není triviální

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Dekompozice na singulární hodnoty

Použití pro $M \neq N$

- méně rovnic, $M < N$
- nekonečně mnoho řešení
- rozklad na singulární hodnoty – $N - M$ nulových σ_i
  - nemusí být přesně nulové (numerické nepřesnosti)
  - může jich být více díky dalším singularitám
- Σ' vypočítáme vynulováním problematických σ_i
- přímo vypočteme reprezentativní řešení $\mathbf{x}$
  - včetně ověření, zda je skutečně řešením
- sloupce $\mathbf{V}$ odpovídající nulovaným σ_i generují prostor dalších řešení

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Dekompozice na singulární hodnoty

Použití pro $M \neq N$

- více rovnic, $M > N$
- neexistuje přesné řešení, hledáme nejbližší aproximaci
- rozklad na singulární hodnoty
  - obecně nemusí dát žádná nulová σ_i
  - získáme nejbližší aproximaci řešení, viz naznačený důkaz

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Dekompozice na singulární hodnoty

Použití pro $M \neq N$

- více rovnic, $M > N$
- neexistuje přesné řešení, hledáme nejbližší aproximaci
- rozklad na singulární hodnoty
  - obecně nemusí dát žádná nulová σ_i
  - získáme nejbližší aproximaci řešení, viz naznačený důkaz
- (skoro) nulové singulární hodnoty
  - skrytá degenerace systému
  - může vést na jedno nebo i více přesných řešení

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Dekompozice na singulární hodnoty

Použití pro $M \neq N$

- více rovnic, $M > N$
- neexistuje přesné řešení, hledáme nejbližší aproximaci
- rozklad na singulární hodnoty
  - obecně nemusí dát žádná nulová σ_i
  - získáme nejbližší aproximaci řešení, viz naznačený důkaz
- (skoro) nulové singulární hodnoty
  - skrytá degenerace systému
  - může vést na jedno nebo i více přesných řešení
- velmi malé singulární hodnoty
  - ukazují na nízkou citlivost problému
  - právě ve směrech odpovídajících sloupců $\mathbf{V}$
  - zpravidla lépe vynulovat v Σ'

Dekompozice na singulární hodnoty

Aproximace matic

- původní matici lze vyjádřit

$$A_{ij} = \sum_k \sigma_k U_{ik} V_{jk}$$

- je-li většina σ_i skoro nulových
- má smysl ukládat jen několik sloupců **U** a **V**
  - stále dostáváme poměrně přesnou aproximaci **A**
- násobení **Ax** je výrazně efektivnější – $K(M + N)$ operací

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Dekompozice na singulární hodnoty

Algoritmus

- první fáze – redukce na bidiagonální formu
  - využívá Householderovy transformace
- druhá fáze – iterační, varianta výpočtu vlastních hodnot
- výjimečně stabilní
  - zaměření na vytažení problematických vlastností $\mathbf{A}$ do Σ
- použijeme existující implementaci :-)
  - už to za nás jednou někdo udělal
  - další vylepšující triky dodavatelů knihoven
  - nezbavuje to odpovědnosti za interpretaci výsledku
- původní algoritmus Golub a Reinsch, *Singular value decomposition and least squares solutions*, 1970

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

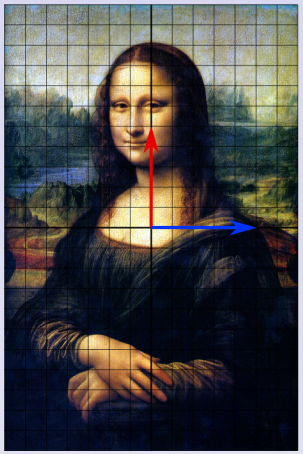
QR dekompozice

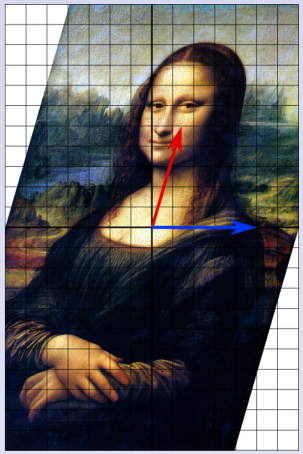
Dekompozice na singulární

Vlastní hodnoty a vektory

- vlastní hodnoty a vektory matice (transformace)

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{x}$$





- obecná reálná matice nemusí mít reálné vlastní hodnoty
- více viz kursy lineární algebry

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Vlastní hodnoty a vektory

Použití

- hledání kořenů polynomů
- výpočty vyšších mocnin matic

$$\mathbf{A}^n = \mathbf{V}\Lambda\mathbf{V}^{-1}\mathbf{V}\Lambda\mathbf{V}^{-1}\ldots\mathbf{V}\Lambda\mathbf{V}^{-1} = \mathbf{V}\Lambda^n\mathbf{V}^{-1}$$

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

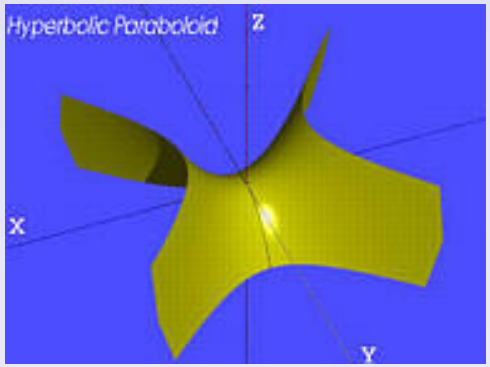
Vlastní hodnoty a vektory

Použití

- hledání kořenů polynomů
- výpočty vyšších mocnin matic

$$\mathbf{A}^n = \mathbf{V}\Lambda\mathbf{V}^{-1}\mathbf{V}\Lambda\mathbf{V}^{-1}\ldots\mathbf{V}\Lambda\mathbf{V}^{-1} = \mathbf{V}\Lambda^n\mathbf{V}^{-1}$$

- kritické body funkce více proměnných
  - první parciální derivace jsou nulové
  - Hessián (matice druhých parciálních derivací) určuje aproximaci kvadrikou v daném bodě
  - symetrická matice – reálné hodnoty, ortonormální vektory
  - extrémy – všechny λ kladné, sedla – některé záporné
  - absolutní hodnoty λ určují tvar, vektory orientaci

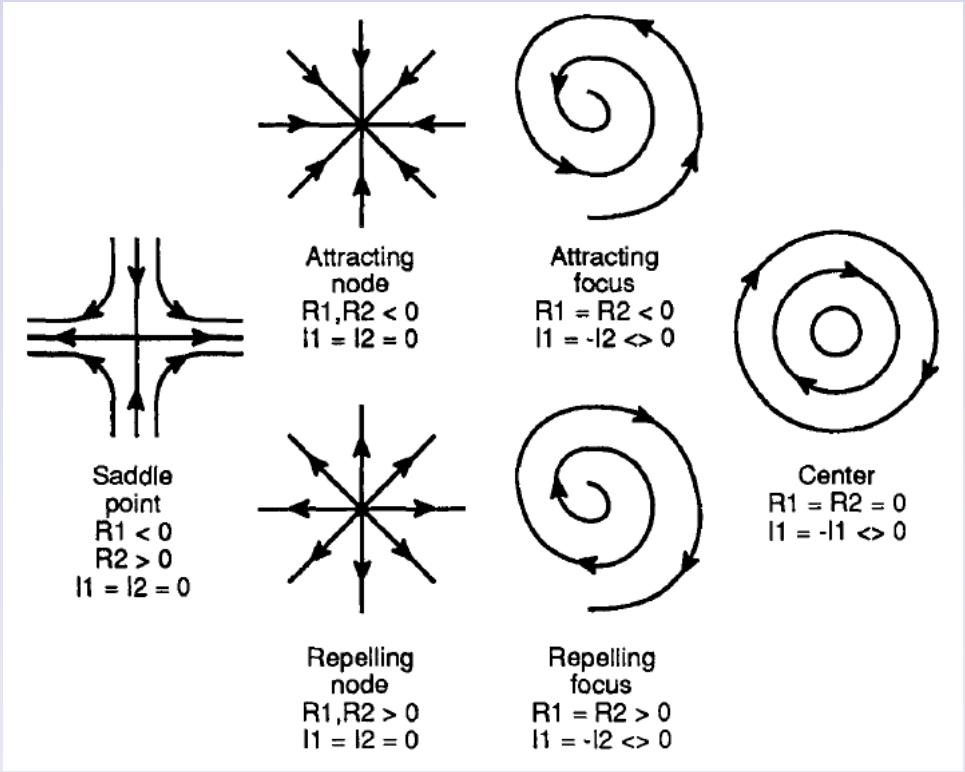


Motivace
Výkon současných CPU
Blokové algoritmy
BLAS
Asymptotická složitost
Systémy lineárních rovnic
Přehled metod
Gaussova eliminace
Rozklady matic
LU dekompozice
Choleského dekompozice
QR dekompozice
Dekompozice na singulární

Vlastní hodnoty a vektory

Použití

- kritické body vektorového pole
  - velikost vektoru je nulová
  - Jakobián (matice prvních parciálních derivací)

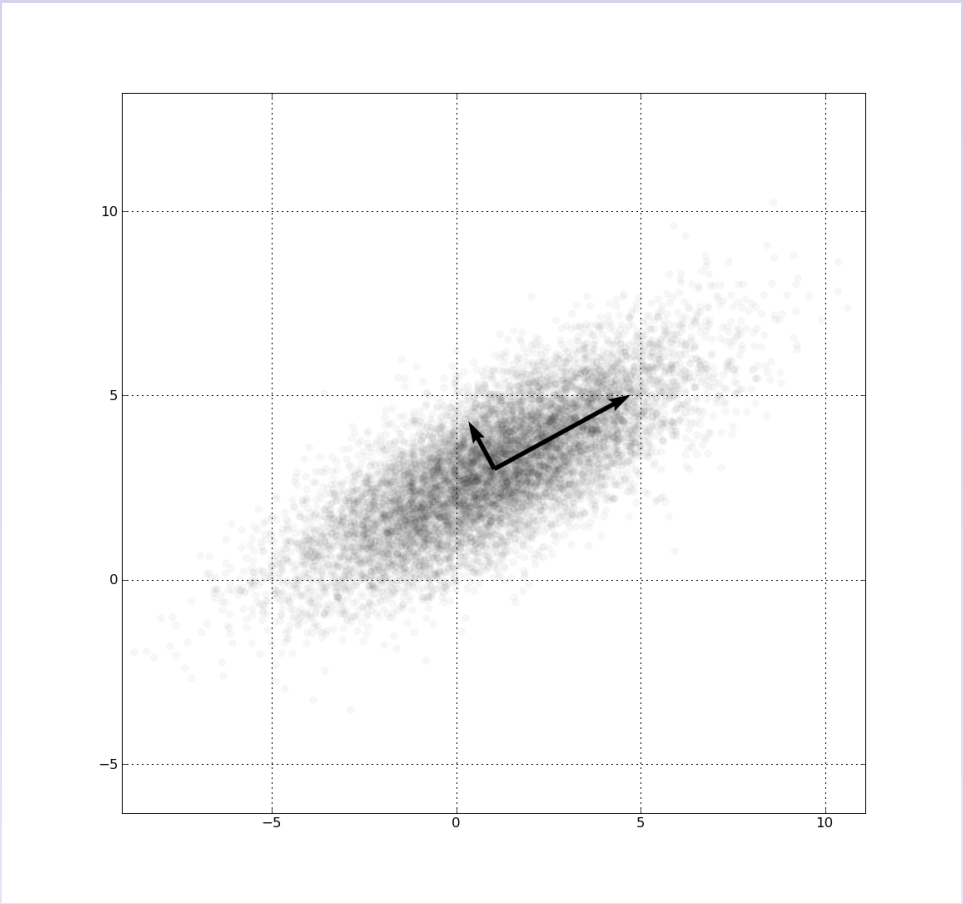


Motivace
Výkon současných CPU
Blokové algoritmy
BLAS
Asymptotická složitost
Systémy lineárních rovnic
Přehled metod
Gaussova eliminace
Rozklady matic
LU dekompozice
Choleského dekompozice
QR dekompozice
Dekompozice na singulární

Vlastní hodnoty a vektory

Použití

- analýza hlavních komponent



- téma příští přednášky

Motivace
Výkon současných CPU
Blokové algoritmy
BLAS
Asymptotická složitost
Systémy lineárních rovnic
Přehled metod
Gaussova eliminace
Rozklady matic
LU dekompozice
Choleského dekompozice
QR dekompozice
Dekompozice na singulární

Vlastní hodnoty a vektory

Algoritmy

- iterační algoritmus
  - $\mathbf{A}_0 := \mathbf{A}$
  - dekompozice $\mathbf{A}_k = \mathbf{Q}_k \mathbf{R}_k$
  - položíme $\mathbf{A}_{k+1} := \mathbf{R}_k \mathbf{Q}_k$
- platí $\mathbf{A}_{k+1} = \mathbf{R}_k \mathbf{Q}_k = \mathbf{Q}_k^T \mathbf{Q}_k \mathbf{R}_k \mathbf{Q}_k = \mathbf{Q}_k^{-1} \mathbf{A}_k \mathbf{Q}_k$
- tedy iterační krok zachovává vlastní hodnoty
- lze ukázat, že za jistých okolností konverguje k Shurově formě

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Vlastní hodnoty a vektory

Algoritmy

- numericky dost nepříjemný problém
- ještě jasnější případ, kdy sáhnout k hotovým řešením
- různé varianty pro různé případy
  - reálné a komplexní
  - vlastní hodnoty, vektory, obojí
  - různé speciální typy matic
- vztah k singulárním hodnotám
  - sloupce $\mathbf{U}$ v SVD jsou vlastní vektory $\mathbf{A}\mathbf{A}^T$
  - nenulové singulární hodnoty jsou odmocniny nenulových vlastních hodnot $\mathbf{A}\mathbf{A}^T$

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Shrnutí

- základní rozklady matic
  - LU, Cholského, QR
  - použití především k řešení systémů lineárních rovnic
  - existují i další varianty
- SVD
  - náročnější postup, větší stabilita
  - špatně podmíněné systémy
  - větší náhled do problému – další aplikace
- vlastní hodnoty
  - rozklad matice $\mathbf{A} = \mathbf{V}\Lambda\mathbf{V}^{-1}$
  - přímé využití pro charakteristiku řady problémů
  - (více v příští přednášce)
- existující implementace
  - téměř vždy je použijeme
  - je nutné vědět, co děláme a co můžeme od dané implementace čekat

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Iterační zpřesnění

- i přímé metody řešení lineárních rovnic ztrácí přesnost
  - v závislosti na velikosti systému a poměru koeficientů
  - kumulace chyb pocházejících ze sčítání/odčítání
  - v optimistickém případě 2–3 platná místa
  - u „skoro singulárních“ matic podstatně horší
- lze vylepšit iteračním procesem
- $\mathbf{x}$ je přesné řešení $\mathbf{Ax} = \mathbf{b}$, známe ale jen $\mathbf{x} + \delta\mathbf{x}$; položíme

$$\mathbf{A}(\mathbf{x} + \delta\mathbf{x}) = \mathbf{b} + \delta\mathbf{b}$$

odečtením dostaneme

$$\mathbf{A}\delta\mathbf{x} = \delta\mathbf{b} = \mathbf{A}(\mathbf{x} + \delta\mathbf{x}) - \mathbf{b}$$

pravou stranu umíme spočítat, řešíme nový systém rovnic pro $\delta\mathbf{x}$

  - pro výpočet pravé strany je nutná vyšší přesnost odčítání
  - s výhodou využijeme recyklaci LU dekompozice
- výsledným $\delta\mathbf{x}$ korigujeme původní $\mathbf{x}$

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

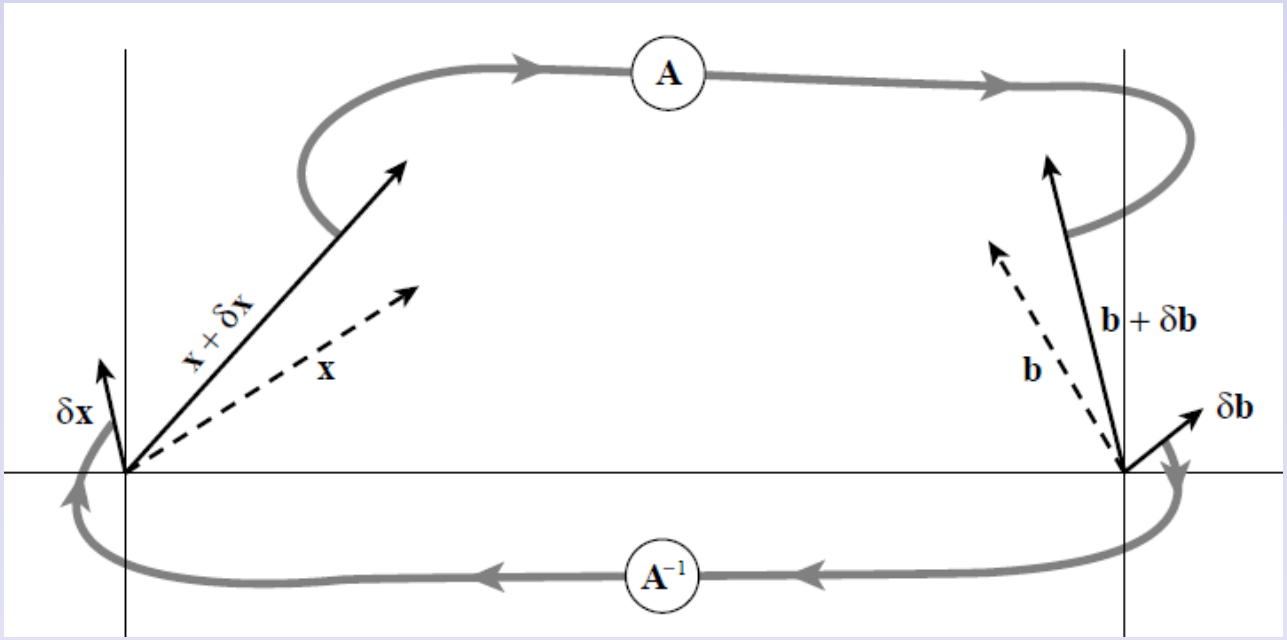
Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Iterační zpřesnění

- schematické znázornění



- postup lze opakovat, zpravidla stačí 1–2 krát

Motivace
Výkon současných CPU
Blokové algoritmy
BLAS
Asymptotická složitost
Systémy lineárních rovnic
Přehled metod
Gaussova eliminace
Rozklady matic
LU dekompozice
Choleského dekompozice
QR dekompozice
Dekompozice na singulární

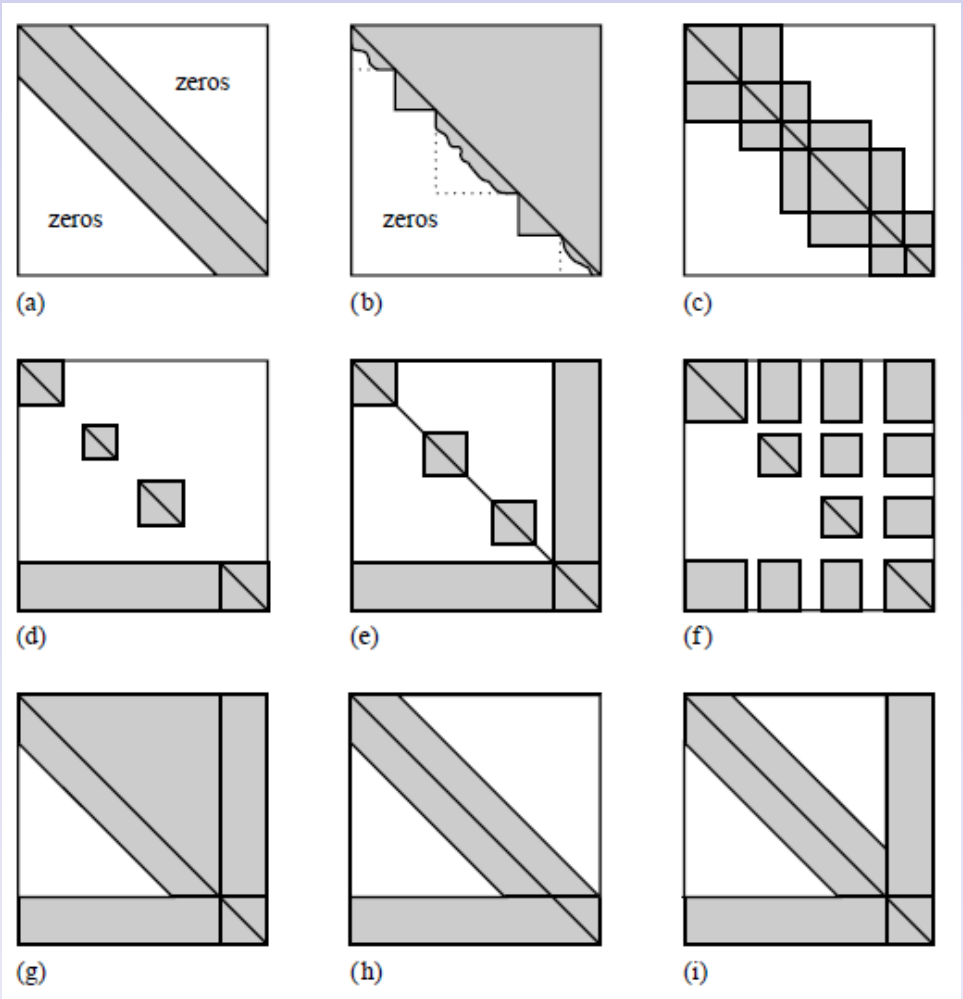
Řídké matice

- typicky rozsáhlé problémy (miliony rovnic)
  - ale počet nenulových koeficientů je malý, zpravidla $O(N)$
- v extrémním případě neřešitelné standardními metodami
  - příliš velká paměťová a časová náročnost
  - de-facto nepřekonatelné problémy s přesností výpočtu
- specializované metody
  - využívají nulových koeficientů
  - vyžadují jen $O(N)$ operací i paměti
  - závislé na konkrétním vzoru nenulových prvků
  - např. LU dekompozice tridiagonální matice – jen N prvkový vektor navíc a 2 cykly o $N-1$ iteracích
- v některých případech aproximační
  - nenulové prvky se během řešení „množí“ nad míru
  - je třeba některé zanedbávat

Motivace
Výkon současných CPU
Blokové algoritmy
BLAS
Asymptotická složitost
Systémy lineárních rovnic
Přehled metod
Gaussova eliminace
Rozklady matic
LU dekompozice
Choleského dekompozice
QR dekompozice
Dekompozice na singulární

Řídké matice

Některé speciální vzory



Řídké matice

Uložení v paměti

- problematická není jen výpočetní náročnost, ale zejména nároky na paměť
  - $N = 10^6$ by vyžadovalo ve floatech 4 TB
- existují různá schémata, např.
  - pole hodnot `float val[]` a indexů `int idx[]`
  - prvních N prvků `val[]` jsou všechny diagonální prvky, zpravidla bývají nenulové
  - prvních N prvků `idx[]` jsou indexy ve `val[]`, kde jsou uloženy první nenulové nediagonální prvky jednotlivých řádků matice
  - `idx[`$N+1$`]` ukazuje za poslední prvek `val[]`
  - další prvky `val[]` jsou nenulové nediagonální prvky matice uspořádané po řádcích a sloupcích
  - další prvky `idx[]` jsou indexy sloupců odpovídajících prvků `val[]`

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Řídké matice

Uložení v paměti

- původní matice

$$\begin{bmatrix} 3 & 0 & 1 & 0 & 0 \\ 0 & 4 & 0 & 0 & 0 \\ 0 & 7 & 5 & 9 & 0 \\ 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 6 & 5 \end{bmatrix}$$

- je reprezentována

	1	2	3	4	5	6	7	8	9	10	11
`idx[]`	7	8	8	10	11	12	3	2	4	5	4
`val[]`	3	4	5	0	5	n/a	1	7	9	2	6

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Řídké matice

Sherman-Morrisonův postup

- řídké matice, které se „trochu liší“ od známého vzoru
  - navíc řádek, sloupec apod.
  - obecně $\mathbf{A}' = \mathbf{A} + (\mathbf{u} \otimes \mathbf{v})$
- lze ukázat

$$(\mathbf{A}')^{-1} = (\mathbf{A} + (\mathbf{u} \otimes \mathbf{v}))^{-1} = \mathbf{A}^{-1} - \frac{(\mathbf{A}^{-1}\mathbf{u}) \otimes (\mathbf{v}\mathbf{A}^{-1})}{1 + \mathbf{v}\mathbf{A}^{-1}\mathbf{u}}$$

- některou z hotových metod vypočteme $\mathbf{A}^{-1}$
- aplikací vzorce získáme $(\mathbf{A}')^{-1}$
  - je-li $\mathbf{A}^{-1}$ řídká v řádu $O(N)$, náročnost celé metody je $O(N)$
- postup lze opakovat pro ruzná $\mathbf{u}, \mathbf{v}$
- existují další zobecnění (Woodbury, viz literatura)

Motivace
Výkon současných CPU
Blokové algoritmy
BLAS
Asymptotická složitost
Systémy lineárních rovnic
Přehled metod
Gaussova eliminace
Rozklady matic
LU dekompozice
Choleského dekompozice
QR dekompozice
Dekompozice na singulární

Iterační metody

- obecně méně přesné než přímé metody
- řešení řídkých systémů
  - neexistuje přímá metoda pro konkrétní vzor
  - výpočet iteračního kroju je zpravidla $O(N)$
  - nevyžaduje další paměť
- téměř singulární systémy
  - přímé metody ztrácí přesnost kumulací chyby

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Iterační metody

Prostá iterace

- rovnici $\mathbf{Ax} = \mathbf{b}$ převedeme na tvar $\mathbf{x} = \mathbf{A}'\mathbf{x} + \mathbf{b}'$
- opakovaně počítáme iterační krok
- kritérium konvergence
  - zobrazení $\mathbf{x} \mapsto \mathbf{A}'\mathbf{x} + \mathbf{b}'$ musí být kontrakce
  - stejné jako u nelineárních rovnic
- naplnění předpokladů věty o pevném bodě
  - $\lim_{k\to\infty} |(\mathbf{A}')^k| = 0$ pro nějakou maticovou normu
  - Frobeniova norma
    $$|\mathbf{A}| = \sqrt{\sum_{i,j} a_{ij}^2}$$
  - spektrální norma
    $$|\mathbf{A}| = \rho(\mathbf{A}^T\mathbf{A})$$
    kde $\rho()$ je maximální absolutní hotnota vlastních hodnot matice
  - maximální součet sloupce nebo řádku

Iterační metody

Prostá iterace – příklady konkrétních metod

- Jacobi: z rozkladu $\mathbf{A} = \mathbf{D} - \mathbf{L} - \mathbf{U}$ dostaneme $\mathbf{x} = \mathbf{D}^{-1}(\mathbf{L} + \mathbf{U})\mathbf{x} + \mathbf{D}^{-1}\mathbf{b}$, tj.

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j=1, j \neq i}^{n} a_{ij} x_j^{(k)} \right)$$

- Gauss-Seidel: $\mathbf{x} = (\mathbf{D} - \mathbf{L})^{-1}\mathbf{U}\mathbf{x} + (\mathbf{D} - \mathbf{L})^{-1}\mathbf{U}\mathbf{b}$

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

  lze recyklovat uložení $\mathbf{x}$
- konvergence není zaručena automaticky
  - pro konkrétní $\mathbf{A}$ některé metody konvergují, jiné ne
  - specifická kritéria viz literatura

Iterační metody

Metoda konjugovaných gradientů

- lze využít i metody řešení nelineárních rovnic
  - smysl to má jen ve velmi specifických případech
  - výjimkou je metoda konjugovaných gradientů
- minimalizujeme funkci

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}\mathbf{A}\mathbf{x} - \mathbf{b}\mathbf{x}$$

- v minimu je její gradient nulový, tj.

$$0 = \nabla f = \mathbf{A}\mathbf{x} - \mathbf{b}$$

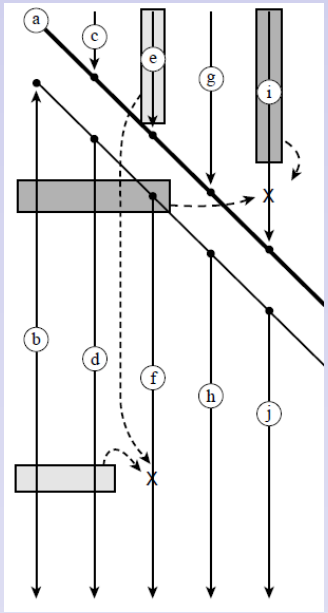
  tedy minimum f přesně odpovídá řešení $\mathbf{A}\mathbf{x} = \mathbf{b}$
- metoda tedy najde minimum po N iteracích
  - f je kvadratická forma, viz minulá přednáška
- takto jednoduše funguje jen pro pozitivně definitní $\mathbf{A}$, lze rozšířit
- při výpočtu je třeba jen násobit $\mathbf{A}\mathbf{x}$
  - to lze u řídkých matic velmi efektivně

Motivace
Výkon současných CPU
Blokové algoritmy
BLAS
Asymptotická složitost
Systémy lineárních rovnic
Přehled metod
Gaussova eliminace
Rozklady matic
LU dekompozice
Choleského dekompozice
QR dekompozice
Dekompozice na singulární

Bloková LU dekompozice

- rekurzivní formulace algoritmu

$$\left(\begin{array}{c|c|c} \mathbf{A}_{00} & \mathbf{a}_{01} & \mathbf{A}_{02} \\ \hline \mathbf{a}_{10}^T & \alpha_{11} & \mathbf{a}_{12}^T \\ \mathbf{A}_{20} & \mathbf{a}_{21} & \mathbf{A}_{22} \end{array}\right)$$



- skalární a bloková verze

$$\mathbf{a}_{21} = \mathbf{a}_{21} / \alpha_{11}$$
$$\mathbf{a}_{12}^T \quad \text{zůstává}$$
$$\mathbf{A}_{22} = \mathbf{A}_{22} - \mathbf{a}_{21}\mathbf{a}_{12}$$

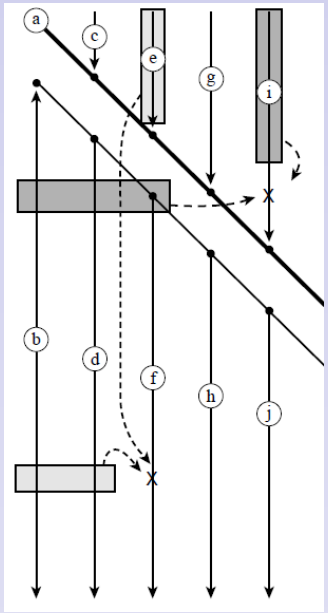
$$\left(\frac{\mathbf{A}_{11}}{\mathbf{A}_{21}}\right) = LU\left(\frac{\mathbf{A}_{11}}{\mathbf{A}_{21}}\right)$$
$$\mathbf{A}_{12} = \mathbf{L}_{11}^{-1}\mathbf{A}_{12}$$
$$\mathbf{A}_{22} = \mathbf{A}_{22} - \mathbf{L}_{21}\mathbf{A}_{12}$$

- podstatná část operací je násobení matic
- takto implementuje knihovna LAPACK

Bloková LU dekompozice

- rekurzivní formulace algoritmu

$$\left(\begin{array}{c|cc} \mathbf{A}_{00} & \mathbf{a}_{01} & \mathbf{A}_{02} \\ \hline \mathbf{a}_{10}^T & \alpha_{11} & \mathbf{a}_{12}^T \\ \mathbf{A}_{20} & \mathbf{a}_{21} & \mathbf{A}_{22} \end{array}\right)$$



- skalární a bloková verze

$$\mathbf{a}_{21} = \mathbf{a}_{21} / \alpha_{11}$$
$$\mathbf{a}_{12}^T \quad \text{zůstává}$$
$$\mathbf{A}_{22} = \mathbf{A}_{22} - \mathbf{a}_{21}\mathbf{a}_{12}$$

$$\left(\frac{\mathbf{A}_{11}}{\mathbf{A}_{21}}\right) = LU\left(\frac{\mathbf{A}_{11}}{\mathbf{A}_{21}}\right)$$
$$\mathbf{A}_{12} = \mathbf{L}_{11}^{-1}\mathbf{A}_{12}$$
$$\mathbf{A}_{22} = \mathbf{A}_{22} - \mathbf{L}_{21}\mathbf{A}_{12}$$

- podstatná část operací je násobení matic
- takto implementuje knihovna LAPACK
- problém algoritmu – „dlouhé“ matice $\mathbf{A}_{21}$ a $\mathbf{A}_{12}$
- Quintana et. al (2008): algoritmus s bloky $2p \times p$

Paralelní blokové algoritmy

- paralelismus na úrovni BLAS
  - existují optimalizované implementace
  - implicitní synchronizace na konci každého volání
  - nemusí být dosažitelný optimální výkon
- blokové operace jsou efektivní provedené vcelku
  - všechna data se dostanou naráz do cache
  - na úrovni L1/L2 se vyplatí na jednom jádru CPU
- vzájemně nezávislé blokové operace na více jádrech

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Paralelní blokové algoritmy

- Quintana et. al (2008) – prototypová implementace SuperMatrix
  - konkrétní algoritmus proběhne „abstraktně“
  - blokové operace s deklarovanými závislostmi se pouze zařadí do fronty
  - následně se vyhodnotí pořadí zpracování
  - operace se provedou potenciálně paralelně
- čitelná formulace algoritmu bez explicitního paralelismu
- efektivní provedení
  - matice $n > 5000$
  - 16 jader CPU
  - LU dekompozice na více než 50 % teoretického výkonu

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Využití GPU

- téměř vše, co platí o cache, platí také o GPU
  - rychlá paměť omezené velikosti
  - netriviální režie kopírování z/do hlavní paměti
- paralelní kód
  - daleko masivnější (desítky až stovky)
- viz PV197: GPU Programming

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární

Shrnutí

- řešení lineárních rovnic je součást řady problémů
- v operacích LA se tráví velká část výpočetního času
- má smysl hledat optimalizované a numericky stabilní algoritmy
- neexistuje univerzální řešení
- k dispozici řada optimalizovaných implementací

Motivace

Výkon současných CPU

Blokové algoritmy

BLAS

Asymptotická složitost

Systémy lineárních rovnic

Přehled metod

Gaussova eliminace

Rozklady matic

LU dekompozice

Choleského dekompozice

QR dekompozice

Dekompozice na singulární