Zápisky z lineární algebry - Ultimátní průvodce

Jan Ráček

23. září 2025

Abstrakt

Tento dokument obsahuje přehledné a srozumitelné vysvětlení lineární algebry pro programátory. Všechny koncepty jsou vysvětleny pomocí přirovnání a vizualizací. Každá matematická operace je doplněna o její "programátorský význam".

Obsah

Přednáška 1 - Úvod do matic

1 Zobrazení (funkce)

1.1 Základní definice

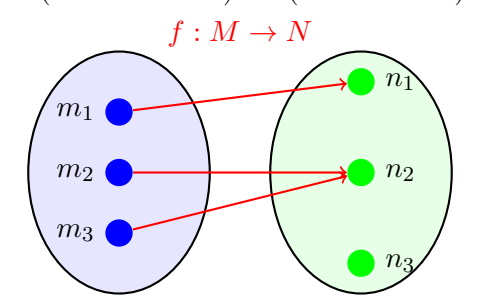
Definice (Zobrazení). Zobrazení *(neboli* funkce*) z množiny M do množiny N je předpis $f : M \to N$, který každému prvku $m \in M$ přiřadí právě jeden prvek $n \in N$. Zapisujeme $n = f(m)$.*

Poznámka (Programátorská analogie). *Zobrazení je jako funkce v programování:*

```
function f(m: M): N {
    return n; // právě jedna hodnota pro každý vstup
}
```

Definice (Vzor a obraz). *Pro zobrazení $f : M \to N$ a prvek $m \in M$, $n \in N$:*

- *Prvek m se nazývá **vzor** (angl. preimage)*
- *Prvek $n = f(m)$ se nazývá **obraz** (angl. image)*
- *Množina $f^{-1}(n) = \{x \in M \mid f(x) = n\}$ se nazývá **vzor prvku** n*



$f(m_1) = n_1$, $f(m_2) = n_2$, $f(m_3) = n_2$

1.2 Důležité pojmy

Definice (Definiční obor a obor hodnot).

- ***Definiční obor*** $D(f) \subseteq M$*: množina všech $m \in M$, pro které je $f(m)$ definováno*
- ***Obor hodnot*** $W(f) \subseteq N$*: množina všech $n \in N$, pro které existuje $m \in M$ s $f(m) = n$*

Definice (Prosté zobrazení (injektivní)). *Zobrazení $f : M \to N$ je **prosté**, pokud:*

$$m_1 \neq m_2 \implies f(m_1) \neq f(m_2)$$

Definice (Na zobrazení (surjektivní)). *Zobrazení $f : M \to N$ je **na**, pokud $W(f) = N$.*

Definice (Bijektivní zobrazení). *Zobrazení je **bijektivní**, pokud je zároveň prosté a na.*

Typ zobrazení	Matematická podmínka	Programátorská analogie
Prosté	Žádné dva vstupy nemají stejný výstup	Hash funkce bez kolizí
Na	Každý prvek v N má vzor	Funkce pokrývající celý cílový typ
Bijektivní	Obě podmínky zároveň	Perfect hash function

1.3 Inverzní zobrazení

Věta (Inverzní zobrazení). *Pro bijektivní zobrazení $f : M \to N$ existuje* ***inverzní zobrazení*** *$f^{-1} : N \to M$ takové, že:*

- $f^{-1}(f(m)) = m$ *pro všechna* $m \in M$
- $f(f^{-1}(n)) = n$ *pro všechna* $n \in N$

Příklad (Inverzní funkce).

- $f(x) = 2x$, $f^{-1}(x) = x/2$
- $f(x) = x^3$, $f^{-1}(x) = \sqrt[3]{x}$
- $f(x) = e^x$, $f^{-1}(x) = \ln x$

Poznámka (Programátorská analogie). *Inverzní zobrazení je jako dešifrovací funkce:*

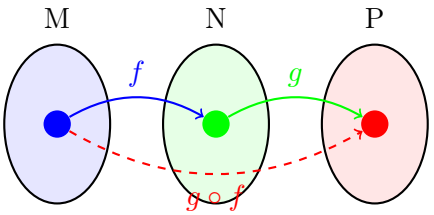
```
function encrypt(data: string): string { ... }
function decrypt(encrypted: string): string { ... }

decrypt(encrypt(data)) == data  // vždy platí
```

1.4 Složení zobrazení

Definice (Složené zobrazení). *Pro zobrazení $f : M \to N$ a $g : N \to P$ definujeme* ***složené zobrazení*** *$g \circ f : M \to P$ předpisem:*

$$(g \circ f)(m) = g(f(m))$$



$$(g \circ f)(m) = g(f(m))$$

Definice (Identita). ***Identita na množině*** *M je zobrazení $\mathrm{id}_M : M \to M$ definované předpisem:*

$$\mathrm{id}_M(m) = m \quad \textit{pro všechna } m \in M$$

Věta (Vlastnosti identity). *Pro libovolné zobrazení $f : M \to N$ platí:*

- $f \circ \mathrm{id}_M = f$
- $\mathrm{id}_N \circ f = f$

1.5 Příklady zobrazení na číselných množinách

Příklad (Aritmetické operace jako zobrazení).

- ***Sčítání:*** $f : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$, $f(x, y) = x + y$
- ***Násobení:*** $g : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$, $g(x, y) = x \cdot y$

Příklad (Řešení rovnic). *Pro pevné $b, c \in \mathbb{R}$:*

- *Rovnice $x + b = c$ má řešení $x = c - b$ (vždy existuje)*

- *Rovnice $x \cdot b = c$ má řešení $x = c/b$ (existuje právě když $b \neq 0$)*

Poznámka (Programátorská analogie)**.** *Řešení rovnic je jako hledání inverzní operace:*

```
// Sčítání a jeho inverze
function add(a: number, b: number): number { return a + b; }
function subtract(c: number, b: number): number { return c - b; }

// Násobení a jeho inverze (s podmínkou)
function multiply(a: number, b: number): number { return a * b; }
function divide(c: number, b: number): number | null {
    return b !== 0 ? c / b : null;
}
```

1.6 Zbytkové třídy modulo m

Definice (Zbytková třída)**.** *Pro pevné $m \in \mathbb{N}$ definujeme na $\mathbb{Z}$ relaci ekvivalence:*

$$a \equiv b \pmod{m} \iff m \mid (a - b)$$

***Zbytková třída** čísla a je množina:*

$$[a]_m = \{b \in \mathbb{Z} \mid b \equiv a \pmod{m}\}$$

Definice (Množina zbytků)**.**

$$\mathbb{Z}_m = \{0, 1, 2, \ldots, m-1\}$$

Každé celé číslo je kongruentní s právě jedním prvkem $\mathbb{Z}_m$.

Příklad (Aritmetika modulo 7)**.**

$$\begin{aligned} 3 + 5 = 8 &\equiv 1 \pmod{7} \\ 3 \cdot 5 = 15 &\equiv 1 \pmod{7} \\ 2^{-1} &= 4 \quad \textit{(protože } 2 \cdot 4 = 8 \equiv 1 \pmod{7}) \end{aligned}$$

Věta (Tělesa modulo prvočíslo)**.** *Je-li p prvočíslo, pak $\mathbb{Z}_p$ tvoří těleso. To znamená:*

- *Každá rovnice $ax \equiv b \pmod{p}$ má pro $a \not\equiv 0$ právě jedno řešení*
- *Každý nenulový prvek má inverzní prvek vzhledem k násobení*

Poznámka (Programátorská analogie)**.** *Aritmetika modulo je jako práce s cyklickými datovými typy:*

```
// Hodiny: aritmetika modulo 12
function addHours(current: number, hours: number): number {
    return (current + hours) % 12;
}

// Hash tabulky: aritmetika modulo velikosti tabulky
function hash(key: string, tableSize: number): number {
    return computeHash(key) % tableSize;
}
```

1.7 Shrnutí

Koncept	**Matematický zápis**	**Vlastnosti**
Zobrazení	$f : M \to N$	Přiřazuje každému $m \in M$ právě jedno $n \in N$
Prosté	$m_1 \neq m_2 \Rightarrow f(m_1) \neq f(m_2)$	Žádné kolize
Na	$W(f) = N$	Pokrývá celý cílový obor
Bijektivní	Prosté + na	Existuje inverze
Složení	$(g \circ f)(m) = g(f(m))$	Asociativní
Identita	$\mathrm{id}_M(m) = m$	Neutrální prvek pro skládání

Literatura

- Petr Olšák: *Úvod do algebry, zejména lineární*, ISBN 978-80-01-03775-1
- Koucký: *Kombinatorické metody I*

Přednáška 2 - Matematická indukce a kombinatorika

2 Matematická indukce a kombinatorika

2.1 Princip matematické indukce

Definice (Přirozená čísla)**.** *Množina přirozených čísel* $\mathbb{N}$ *obsahuje číslo 1 a pro každé* $n \in \mathbb{N}$ *také obsahuje* $n+1$*. Formálně:*

- $1 \in \mathbb{N}$
- *Pokud* $n \in \mathbb{N}$*, pak* $n+1 \in \mathbb{N}$

Poznámka. *Představte si to jako rekurzivní funkci v programování: máme základní případ (1) a rekurzivní krok (přidání 1).*

Věta (Princip matematické indukce)**.** *Pokud množina* $M \subseteq \mathbb{N}$ *splňuje:*

1. $1 \in M$ *(základní krok)*
2. *Pokud* $n \in M$*, pak* $n+1 \in M$ *(indukční krok)*

Pak $M = \mathbb{N}$*.*

Poznámka (Programátorská analogie)**.** *Matematická indukce je jako dokazování, že funkce pracuje správně pro všechny vstupy:*

```
// Základní případ: funguje pro n=1
if (n == 1) return true;

// Indukční krok: pokud funguje pro n, funguje i pro n+1
if (funguje(n)) then funguje(n+1) musí také platit
```

Příklad (Důkaz pomocí matematické indukce)**.** ***Tvrzení:*** *Každou matici lze převést na horní stupňovitý tvar pomocí Gaussových operací.*

Definice (Horní stupňovitý tvar)**.** *Matice je v horním stupňovitém tvaru pokud:*

- *Každý nenulový řádek má před prvním nenulovým prvkem více nul než předchozí řádek*
- *Nulové řádky jsou na konci*

Příklad horního stupňovitého tvaru:

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 0 & 5 & 6 & 7 \\ 0 & 0 & 8 & 9 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Důkaz matematickou indukcí. **Základní případ ($n = 1$):** Jednořádková matice je vždy v horním stupňovitém tvaru.

Indukční krok: Předpokládejme, že každá matice s n řádky se dá převést na horní stupňovitý tvar. Uvažujme matici s $n+1$ řádky:

1. Najdeme první sloupec s nenulovým prvkem (pivot)
2. Vyměníme řádek s pivotem s prvním řádkem
3. Vynulujeme sloupec pod pivotem pomocí řádkových operací

Transformace matice:

$$\begin{pmatrix} 0 & 0 & c \\ a & b & d \\ e & f & g \end{pmatrix} \xrightarrow{\text{vyměň řádky}} \begin{pmatrix} a & b & d \\ 0 & 0 & c \\ e & f & g \end{pmatrix} \xrightarrow{\text{vynuluj pod pivotem}} \begin{pmatrix} a & b & d \\ 0 & 0 & c \\ 0 & f' & g' \end{pmatrix}$$

Aplikujeme IP

Podmatice $n \times m$

Na zbylou $n \times m$ podmatici aplikujeme indukční předpoklad. □

2.2 Základní kombinatorické principy

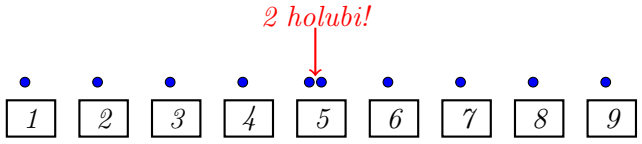
Definice (Ekvidominantní množiny)**.** *Množiny A a B jsou ekvivalentní pokud existuje bijekce (vzájemně jednoznačné zobrazení) $f : A \to B$.*

Poznámka. *Bijekce = každý prvek z A se zobrazí na jiný prvek v B a naopak. Jako hash funkce bez kolizí.*

Věta (Základní kombinatorické principy)**.**

1. ***Pravidlo součtu:*** *Pokud $A \cap B = \emptyset$, pak $|A \cup B| = |A| + |B|$*
2. ***Pravidlo součinu:*** $|A \times B| = |A| \cdot |B|$
3. ***Dirichletův princip:*** *Pokud n prvků rozdělíme do k množin, pak některá množina obsahuje alespoň $\lceil \frac{n}{k} \rceil$ prvků.*

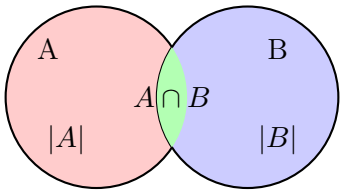
Příklad (Dirichletův princip - holubníkový princip)**.** *Máme 10 holubů a 9 holubníků. Některý holubník musí obsahovat alespoň 2 holuby.*



10 holubů, 9 holubníků ⇒ některý má ≥ 2 holuby

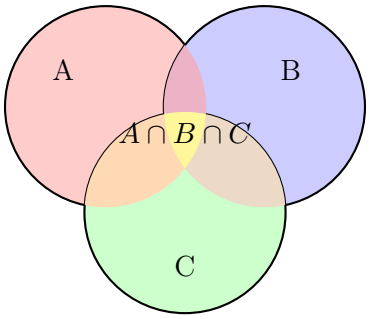
2.3 Princip inkluze a exkluze

Věta (Princip inkluze a exkluze pro 2 množiny)**.** $|A \cup B| = |A| + |B| - |A \cap B|$



$|A \cup B| = |A| + |B| - |A \cap B|$

Věta (Princip inkluze a exkluze pro 3 množiny)**.** $|A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C| + |A \cap B \cap C|$



$$|A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C| + |A \cap B \cap C|$$

2.4 Kombinatorické vzorce

Definice (Faktoriál). $n! = 1 \cdot 2 \cdot 3 \cdot \ldots \cdot n$ *s* $0! = 1$

Poznámka. *Faktoriál roste extrémně rychle - rychleji než exponenciála!*

n	*1*	*2*	*3*	*4*	*5*	*10*
$n!$	*1*	*2*	*6*	*24*	*120*	*3,628,800*
2^n	*2*	*4*	*8*	*16*	*32*	*1,024*

Definice (Permutace bez opakování). *Počet uspořádaných n-tic z n různých prvků:* $P(n) = n!$

Příklad. *Kolik způsobů lze seřadit 3 lidi?* $3! = 6$ *způsobů.*

Permutace	***Pořadí***
1	*A, B, C*
2	*A, C, B*
3	*B, A, C*
4	*B, C, A*
5	*C, A, B*
6	*C, B, A*

Definice (Permutace s opakováním). *Pokud máme* k_1 *prvků typu 1,* k_2 *prvků typu 2, ...,* k_m *prvků typu* m*, pak:*

$$P(k_1, k_2, \ldots, k_m) = \frac{(k_1 + k_2 + \ldots + k_m)!}{k_1! \cdot k_2! \cdot \ldots \cdot k_m!}$$

Příklad. *Kolik slov lze vytvořit z písmen "MAMA"? M=2, A=2, celkem 4 písmena:*

$$\frac{4!}{2! \cdot 2!} = \frac{24}{4} = 6$$

Definice (Variace).

- ***Bez opakování:*** $V(k, n) = \frac{n!}{(n-k)!}$ *(uspořádané k-tice z n prvků)*
- ***S opakováním:*** $\overline{V}(k, n) = n^k$

Poznámka (Programátorská analogie). *Variace s opakováním = počet k-ciferných čísel v číselné soustavě o základu n.*

Příklad (Počet podmnožin). *Počet všech podmnožin n-prvkové množiny je* 2^n*. Každou podmnožinu lze reprezentovat jako binární vektor délky n.*

Bit A	*Bit B*	*Bit C*	*Podmnožina*
0	*0*	*0*	$\emptyset$
0	*0*	*1*	$\{C\}$
0	*1*	*0*	$\{B\}$
0	*1*	*1*	$\{B, C\}$
1	*0*	*0*	$\{A\}$
1	*0*	*1*	$\{A, C\}$
1	*1*	*0*	$\{A, B\}$
1	*1*	*1*	$\{A, B, C\}$

2.5 Shrnující tabulka

Kombinatorický koncept	**Vzorec**	**Příklad**	**Programátorská analogie**
Permutace bez opakování	$n!$	Seřazení n prvků	`array.sort()`
Permutace s opakováním	$\frac{n!}{k_1!k_2!\ldots k_m!}$	Anagrams	`string.permutations()`
Variace bez opakování	$\frac{n!}{(n-k)!}$	Výběr výboru s funkcemi	`combinations with order`
Variace s opakováním	n^k	PIN kódy	`nested loops`
Kombinace	$\binom{n}{k} = \frac{n!}{k!(n-k)!}$	Loterie	`subset selection`