Úvod do vyšší matematiky pro nematematiky
Stručně a srozumitelně

Michal Šitina

Ústav patologické fyziologie
Lékařská fakulta
Masarykova univerzita Brno

Obsah

Úvod

Předkládaný učební text slouží jako skriptum k předmětu „Úvod do vyšší matematiky pro nematematiky - Stručně a srozumitelně“. Budu rád, když text pomůže komukoli, kdo chce poměrně rychle porozumět základům vyšší matematiky. Pojem „vyšší matematika“ je vágní, většinou se jím myslí matematika přesahující středoškolské znalosti. Skriptum tyto znalosti předpokládá, přesto jsou některé důležité jednotlivosti stručně zopakovány.

Matematika je dnes široce užívána v mnoha vědních disciplínách, ať už v podobně analýzy dat či v podobě matematického modelování. Mimo klasických oblastí jako jsou fyzika nebo chemie je dnes matematika rozsáhle užívána i v biologii a medicíně, ale i ve společenských vědách jako např. v ekonomii či sociologii. A jistě i v mnoha dalších. Sám se zabývám matematickým modelováním v medicíně. Řada příkladů je proto volena právě z oblasti biologie a medicíny.

Základní moto textu je „stručně a srozumitelně“. Učebnice vyšší matematiky bývají psány precizním jazykem pro matematicky vzdělané čtenáře, poněkud rozvlekle, doplněny řadou složitých a málo srozumitelných důkazů. To řadu zájemců o vyšší matematiku po několika stranách zbytečně odradí. Vyšší matematiku lze přitom vcelku do hloubky, přinejmenším pro potřeby aplikovaného matematického modelování, chápat intuitivně, i bez exaktních formulací a formálních důkazů. V našem kurzu budeme postupovat touto méně precizní, intuitivní cestou, přitom ale vždy se snahou o skutečné porozumění. Objasňovat, ilustrovat a procvičovat budeme na mnoha příkladech.

Nejprve popíšeme některé základní pojmy matematiky jako jsou množina či zobrazení. Poté se budeme věnovat vybraným oblastem matematické analýzy, zejména problematice funkcí, limit, derivací, integrálů a diferenciálních rovnic. Dále se budeme zabývat vybranými tématy numerické matematiky a na závěr úvodem do lineární algebry. Zejména v kapitolách o numerické matematice zmíníme i implementaci numerických algoritmů v jazyce Python. V dodatku proto budou potřebné konstrukty tohoto jazyka stručně nastíněny.

Ve žlutých polích budeme definovat nové pojmy.

V modrých polích budeme řešit příklady.

Velice děkuji kolegovi Martinu Dvouletému, studentu Lékařské fakulty Masarykovy univerzity v Brně a VUT v Brně, za podrobné pročtení textu a cenné kritické připomínky.

Michal Šitina
Brno, 2023

1. Stavební kameny matematiky

V této úvodní kapitole velmi stručně popíšeme některé základní matematické pojmy a představy, které se vyskytují ve všech speciálních oblastech matematiky. Těmto pojmům je nutné dobře rozumět.

Učebnice vyšší matematiky bývají psány precizní formou „Definice – Věta – Důkaz“. My budeme postupovat méně precizně, tyto základní pojmy si však přesto krátce objasníme. **Definice** je přiřazení názvu jisté přesněji vymezené entitě. Definice sama o sobě neobsahuje žádnou novou informaci, žádný poznatek. Definice bývají často psány nepřesně formou implikací, tedy „Pokud platí to a ono, označujeme cosi jako ABC“. Vždy je ve skutečnosti rozuměna ekvivalence, tedy „Právě když platí to a ono, označujeme cosi jako ABC“.

Reálná funkce jedné reálné proměnné je každé zobrazení z množiny $\mathbb{R}$ do množiny $\mathbb{R}$.

V tomto případě jsme entitě „zobrazení z množiny $\mathbb{R}$ do množiny $\mathbb{R}$“ přiřadili název „reálná funkce jedné reálné proměnné“.

Věta je tvrzení o vlastnostech či vztazích definovaných pojmů. Za každou větou následuje **Důkaz**, který „krok za krokem“ převádí dříve definované pojmy a již dokázané věty na novou, právě dokazovanou větu. Musíme však „někde začít“. Proto některé věty akceptujeme a priori bez důkazu jako pravdivé. Označujeme je jako **axiomy**, nebo jako **implicitní definice** či jako **odvozovací pravidla**. Pojďme si vše demonstrovat na příkladu Pythagorovy věty.

Pythagorova věta

> *Obsah čtverce sestrojeného nad přeponou libovolného pravoúhlého trojúhelníku je roven součtu obsahů čtverců nad oběma jeho odvěsnami.*

Takto je Pythagorova věta sice formulována „prostým“ jazykem, ale není možná na první pohled zřejmé, že jde o implikaci. Větu můžeme přesněji reformulovat například takto:

> *Pokud je trojúhelník s odvěsnami délek a a b a přeponou délky c pravoúhlý, pak platí $a^2 + b^2 = c^2$.*

I kdybychom nyní platnost této věty dokázali, stále nám zbývá možnost, že existují trojúhelníky, pro které sice platí $a^2 + b^2 = c^2$, ale nejsou pravoúhlé. To však odporuje našim „znalostem a zkušenostem“ s pravoúhlými trojúhelníky.
V „plné verzi“ je totiž Pythagorova věta skutečně formulována jako ekvivalence.

> *Právě když je trojúhelník s odvěsnami délek a a b a přeponou délky c pravoúhlý, platí $a^2 + b^2 = c^2$.*

Obsahuje tak v sobě vlastně dvě věty:

1. *Pokud je trojúhelník s odvěsnami délek a a b a přeponou délky c pravoúhlý, pak platí $a^2 + b^2 = c^2$.*
2. *Pokud pro trojúhelník s odvěsnami délek a a b a přeponou délky c platí $a^2 + b^2 = c^2$, pak je daný trojúhelník pravoúhlý.*

Abychom dokázali Pythagorovu větu, musíme dokázat obě věty 1 a 2. Důkazů první části věty „zleva doprava“ je např. na Wikipedii uvedena řada. Ukažme si proto důkaz druhé části „zprava doleva“.

Mějme tedy nějaký trojúhelník T s odvěsnami délek a a b a přeponou délky c, pro nějž platí $a^2 + b^2 = c^2$. Zkonstruujme nyní pomocný pravoúhlý trojúhelník, který má odvěsny délek a a b. Pak je podle 1. části Pythagorovy věty (kterou již berme jako dokázanou) délka jeho přepony $c = \sqrt{a^2 + b^2}$. Oba trojúhelníky, T i pomocný pravoúhlý, tedy mají stejně dlouhé všechny tři strany a jsou tedy shodné. Mají proto shodné i všechny úhly. Proto je dokazovaný trojúhelník T též pravoúhlý.

Důkaz je zjevný, přesto však „někde začal“, něco předpokládal. Předem jsme definovali, co znamená pravoúhlý trojúhelník. Použili jsme první část Pythagorovy věty, která musela být předem dokázána. Dále jsme použili tvrzení, že dva trojúhelníky se všemi stejnými stranami jsou shodné. To lze vnímat i jako definici shodnosti. Pak bychom ovšem potřebovali větu, že dva shodné trojúhelníky mají stejné všechny tři úhly. Konečně jsme předpokládali, že délka strany nemůže být záporná, což lze brát jako axiom.

1.1. Množina

Množinou rozumíme souhrn jakýchkoli (myšlených či reálných) vzájemně odlišitelných objektů seskupených do jednoho celku. Jednotlivé objekty označujeme jako **prvky množiny**.

Představu množin prvně zformuloval německý matematik Georg Cantor[1]. Založil oblast matematiky dnes označovanou jako Teorie množin. V originále definuje Cantor množinu takto:

> *Unter einer Menge verstehen wir jede Zusammenfassung M von bestimmten wohlunterschiedenen Objekten m unserer Anschauung oder unseres Denkens (welche die Elemente von M genannt werden) zu einem Ganzen.*

Původní Cantorovu verzi označujeme jako Naivní teorii množin. Její nedostatky byly odhaleny počátkem 20. století a byla zprecizována v tzv. Axiomatické teorii množin, spojené se jmény Zermelo a Fraenkel. Teorie množin je hlubokým základem matematiky. Tvrdí se, že vše v matematice lze převést na operace s množinami.

1.2. Relace, zobrazení, funkce

Často nás zajímá vztah dvou množin A a B, tedy souvislost mezi jednotlivými prvky. Taková souvislost se velmi obecně označuje jako **relace**. Vztah množin popíšeme tak, že každému prvku

[1] Georg Cantor, 1845-1918, německý matematik, založil Teorii množin

jedné množiny přiřadíme některé prvky druhé množiny. Případně nemusí být některému prvku přiřazen žádný prvek.

Velmi speciální, ale nejdůležitější relace, kdy **každému** prvku množiny A přiřadíme **právě jeden** prvek množiny B, se označuje jako **zobrazení** nebo jako **funkce**. Funkce a zobrazení jsou obvykle vnímány jako synonyma. Pokud prvky množiny A označíme x a množiny B y, pak zápis

$$f : A \to B; x \to y = f(x)$$

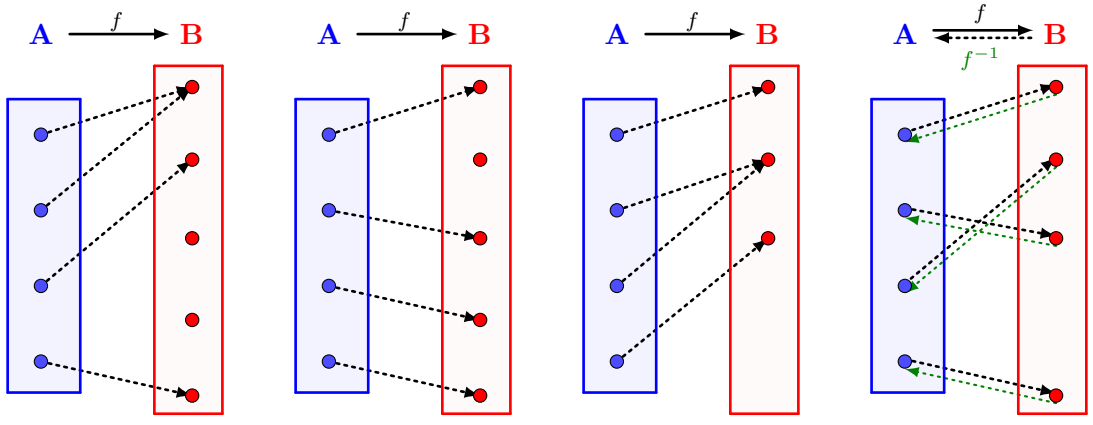
znamená zobrazení f z množiny A do množiny B, které každému $x \in A$ přiřazuje právě jedno $y \in B$.

Množinou A mohou být např. všichni občané České republiky, množinou B reálná čísla ($B = \mathbb{R}$). Zobrazení „měření výšky“ zapíšeme jako

$$\text{měření výšky} : \text{občané ČR} \to \mathbb{R}; \text{občan} \to \text{výška}.$$

Jak záhy uvidíme, důležité jsou vlastnosti zobrazení (obr. 1.1). Prosté neboli injektivní zobrazení přiřazuje každému $x \in A$ odlišné $y \in B$. „Zobrazení na“ neboli surjektivní zobrazení vyčerpává všechny $y \in B$, tedy pro každé $y \in B$ existuje prvek $x \in A$, který je na toto y zobrazován. Vzájemně jednoznačné neboli bijektivní zobrazení je současně injektivní i surjektivní. Je tedy každému $x \in A$ přiřazeno odlišné $y \in B$, přičemž žádné $y \in B$ „nezbyde volné“. Množiny A tak můžeme B spárovat.

Pokud je zobrazení $f : A \to B; x \to y = f(x)$ bijektivní, pak samozřejmě vždy víme, kterému $y \in B$ odpovídá které $x \in A$, a můžeme definovat **inverzní zobrazení** $f^{-1} : B \to A; y \to x = f^{-1}(y)$, pro něž platí $f^{-1}(f(x)) = x$.



(a) obecné (b) injektivní (c) surjektivní (d) bijektivní

Obrázek 1.1.: Typy zobrazení z množiny A do množiny B.

V centrálním dogmatu molekulární biologie DNA → RNA → proteiny by matematik viděl zobrazení.
Definujme množinu E = {A, T, C, G} jako 4 báze DNA a množinu F = {A, U, C, G} jako

4 báze RNA. Zobrazení $f : \mathrm{E} \to \mathrm{F}$ definované předpisem

$$f : \begin{cases} \mathrm{A} \to \mathrm{U} \\ \mathrm{T} \to \mathrm{A} \\ \mathrm{C} \to \mathrm{G} \\ \mathrm{G} \to \mathrm{C} \end{cases}$$

označují biologové jako transkripci. Zobrazení je prosté, a dokonce vzájemně jednoznačné. Proto je možné i inverzní zobrazení, známé jako reverzní transkripce.
Translace je zobrazení z množiny všech 64 uspořádaných trojic bází A,U,C,G do množiny 20 aminokyselin. Vlastní přiřazení aminokyseliny trojici určuje genetický kód.

$$\text{translace : množina všech uspořádaných trojic A, U, C, G} \to \text{20 aminokyselin}$$

Zobrazení není prosté, protože zobrazuje z větší množiny do menší množiny. Biologové tuto vlastnost označují jako degeneraci genetického kódu. Protože zobrazení není prosté, není ani vzájemně jednoznačné a neexistuje reverzní translace.

1.3. Komplexní čísla

Známe několik nekonečných množin čísel, např. přirozená čísla $\mathbb{N}$, celá čísla $\mathbb{Z}$, racionální čísla $\mathbb{Q}$ nebo reálná čísla $\mathbb{R}$. Nad těmito množinami jsou definované jisté operace, t.j. co můžeme s čísly dělat. Přirozená čísla můžeme sčítat a násobit, přičemž výsledek je opět přirozené číslo, ale nemůžeme např. odečíst větší číslo od menšího. To můžeme provádět s celými čísly, ta ale např. nemůžeme libovolně dělit. K tomu už potřebujeme ještě větší množinu čísel, totiž racionální čísla, která jsou vyjádřitelná zlomkem. Racionální čísla jsou první nekonečnou množinou čísel, s nimiž můžeme provádět všechny běžné operace s čísly a výsledek je vždy racionální číslo. Jak už ale zjistili v antickém Řecku, některá čísla, např. $\sqrt{2}$, mezi racionální nepatří. Pokud racionální čísla obohatíme o tato tzv. iracionální čísla, dostaneme reálná čísla. Na okraj poznamenejme, že iracionálních čísel je mnohem více než čísel racionálních.

Kromě reálných čísel ale existuje ještě jedna (a už žádná další!) nekonečná množina poněkud zvláštních čísel, označovaných jako **komplexní čísla**, s nimiž lze počítat jako s „normálními“ čísly. Neexistuje žádné reálné číslo, které umocněno na druhou dá -1.

Definujme proto nějaké nové číslo i, které není reálné, říkejme mu **imaginární**, pro nějž bude platit $i^2 = -1$. Vezměme dvě reálná čísla $a, b \in \mathbb{R}$ a definujme nové číslo z jako

$$z = a + bi.$$

Množinu všech takových čísel označme jako komplexní čísla $\mathbb{C}$.

Pokud budeme s komplexními čísly „normálně“ počítat, zjistíme, že vše bez problémů funguje. Dvě komplexní čísla vynásobíme např. takto

$$(3 + 2i) \,.\, (4 - 3i) = 12 - 9i + 8i - 6i^2 = 12 - i - 6.(-1) = 18 - i.$$

Komplexní číslo obsahuje dvě „proměnné“ a a b, dá se proto zobrazit jako bod v plošném grafu,

v tzv. Gaussově[2] rovině (obr. 1.2), kdy reálnou část a vynášíme na osu x a imaginární část b na osu y. Velikost neboli absolutní hodnota komplexního čísla $\|z\|$ je vzdálenost bodu od počátku, platí

$$\|z\| = \sqrt{a^2 + b^2}.$$

Spojnice bodu a počátku svírá s osou x úhel φ, označovaný jako fáze komplexního čísla. Platí proto $a = \|z\|.\cos\varphi$ a $b = \|z\|.\sin\varphi$. Komplexní číslo tedy můžeme vyjádřit pomocí goniometrických funkcí jako

$$z = a + bi = \|z\|\cos\varphi + i\|z\|\sin\varphi = \|z\|\left(\cos\varphi + i\sin\varphi\right).$$

Absolutní hodnota $\|z\|$ bývá též označována jako amplituda komplexního čísla A.

Leonhard Euler[3] zjistil mimořádnou souvislost mezi komplexní exponenciální funkcí, kdy exponentem je imaginární číslo, a goniometrickým vyjádřením komplexního čísla. Eulerův vztah, někdy popisovaný jako nejkrásnější vztah matematiky, zní

$$\cos\varphi + i\sin\varphi = \mathrm{e}^{i\varphi}.$$

Poněvadž je, jak uvidíme později, derivování exponenciálních funkcí velmi snadné, usnadní zavedení Eulerova vyjádření některé složitější výpočty, například při řešení diferenciálních rovnic.

Odvoďme pomocí Eulerova vyjádření známý vztah $\cos(a+b) = \cos a\cos b - \sin a\sin b$.

Všimneme si, že $\cos(a+b)$ je vlastně reálná část komplexního čísla $\cos(a+b) + i\sin(a+b)$. Reálnou část označme Re. Pak už je vše snadné.

$$\begin{aligned}
\cos(a+b) &= Re\left(\cos(a+b) + i\sin(a+b)\right) = Re\left(e^{i(a+b)}\right) \\
&= Re\left(e^{ia}.e^{ib}\right) = Re\left[(\cos a + i\sin a).(\cos b + i\sin b)\right] \\
&= Re\left[\cos a\cos b + i\cos a\sin b + i\sin a\cos b + i^2\sin a\sin b\right] \\
&= \cos a\cos b - \sin a\sin b
\end{aligned}$$

1.4. Umění zanedbat nepodstatné

Umění rozpoznat a zanedbat nepodstatné je cenná dovednost v životě i v matematice. V matematice vhodné zanedbání vede ke zjednodušení výrazu bez podstatného nárůstu chyby výsledku. Zanedbat můžeme v součtu, je-li jeden ze sčítanců podstatně menší. V součinu pochopitelně zanedbávat nelze. Pokud je tedy $d \ll x$, platí

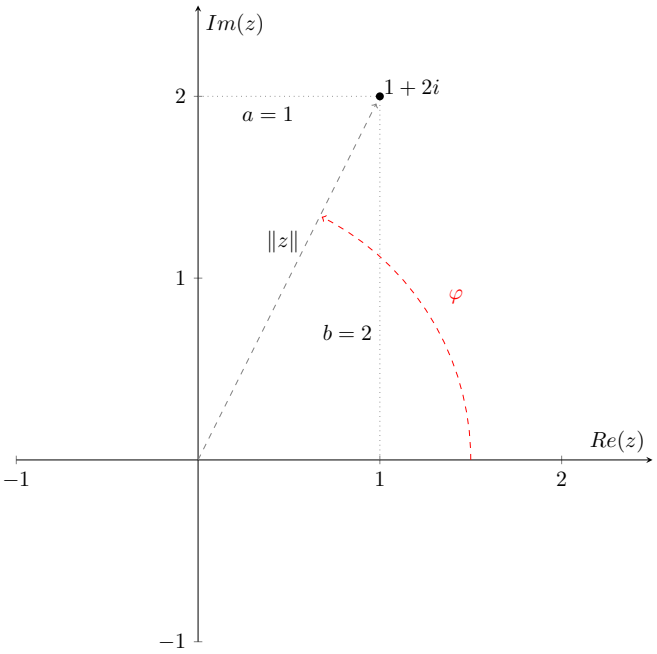
$$x + d \approx x.$$

Zanedbat můžeme zejména mocniny malých hodnot. Pokud je totiž $d \approx 0$, pak

$$d \gg d^2 \gg d^3 \gg \dots.$$

Ilustrujme si zanedbávání na následujícím příkladu.

[2] Carl Friedrich Gauss, 1777-1855, německý matematik a fyzik
[3] Leonhard Euler, 1707-1783, švýcarský matematik a fyzik



Obrázek 1.2.: Komplexní číslo v Gaussově rovině

Chceme určit hodnotu výrazu

$$y = \frac{x}{(x+d)^2}.$$

Nechť je $d \ll x$, např. $x = 1$ a $d = 0,01$. Pak $y = \frac{1}{(1+0,01)^2} = 0,980296$. Pokud bychom zanedbali přímo d, bude výsledek $y = 1$, vznikne tedy chyba asi 2 %. Úpravou vztahu a zanedbáním druhé mocniny d^2 dostaneme

$$y = \frac{x}{(x+d)^2} = \frac{x}{x^2+2dx+d^2} \approx \frac{x}{x^2+2dx} = \frac{1}{x+2d}.$$

Po dosazení zjistíme, že $y \approx \frac{1}{1+0,02} = 0,980392$. Chyba je zhruba 0,1 ‰, ale výraz je podstatně jednodušší. Pokud nás snad ještě obtěžuje součet ve jmenovateli, pak výraz rozšíříme a znovu zanedbáme druhé mocniny.

$$y \approx \frac{1}{x+2d} = \frac{1}{x+2d}.\frac{x-2d}{x-2d} = \frac{x-2d}{x^2-4d^2} \approx \frac{x-2d}{x^2}$$

Po dosazení máme nyní $y \approx \frac{1-0,02}{1} = 0,98$. Už nepotřebujeme ani kalkulačku, a přesto je chyba stále pod 1 ‰.

2. Úvod do matematické analýzy

2.1. Funkce

Jak už jsme uvedli, jsou pojmy zobrazení a funkce většinou chápány jako synonyma.

V užším smyslu rozumíme pod **reálnou funkcí jedné reálné proměnné** zobrazení $f : \mathbb{R} \to \mathbb{R} : x \to y = f(x)$.

Některé funkce jsou definované na podmnožině $\mathbb{R}$, např. logaritmická funkce na kladných reálných číslech $\mathbb{R}^+$. x a y se označují jako argument a hodnota (či obraz) funkce. Argument pochází z definičního oboru funkce $\mathcal{D}$, $x \in \mathcal{D}$, obraz z oboru hodnot $\mathcal{H}$, $y \in \mathcal{H}$. Pro uvedenou funkci f tedy platí $\mathcal{D}(f) = \mathcal{H}f() = \mathbb{R}$.

2.1.1. Inverzní funkce

Funkce f přiřazuje jistému x hodnotu y, $y = f(x)$. Mohla by nás však zajímat i opačná otázka, totiž kterému x byla přiřazena hodnota y (obr. 2.1).

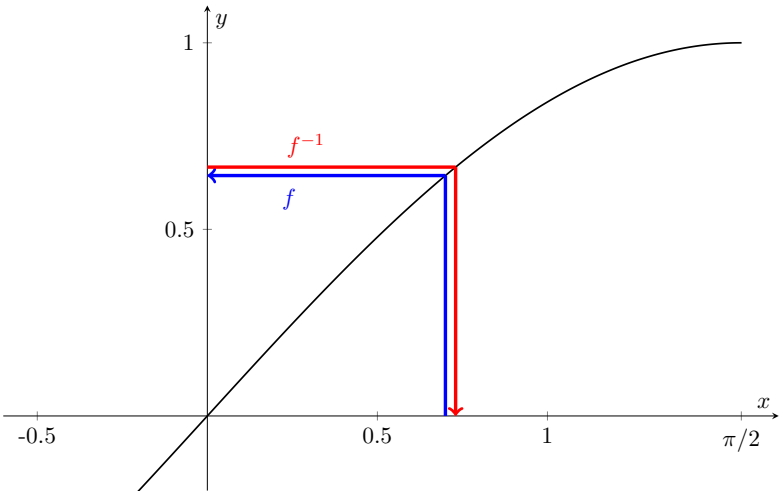
Hledáme tedy vlastně funkci, která dělá opak než funkce f. Označujeme ji jako **inverzní** a značíme f^{-1}. Platí

$$y = f(x) \Leftrightarrow x = f^{-1}(y).$$

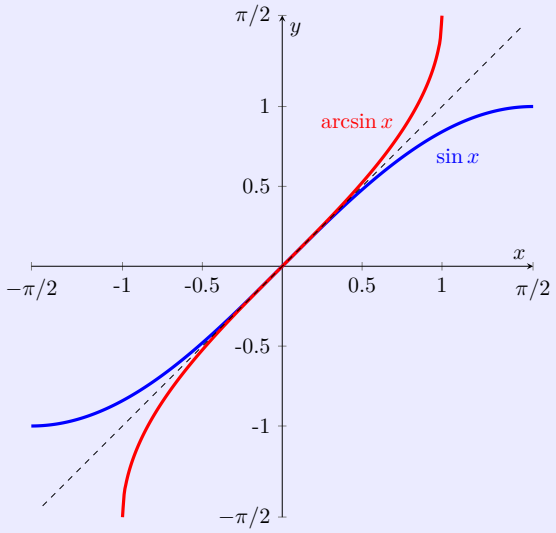
Ne vždy lze inverzní funkci vytvořit. Pokud existuje více hodnot x, jimž funkce f přiřazuje stejné y, pak nemůžeme určit, které x by měla inverzní funkce tomuto y přiřadit. Jak už jsme uvedli v kapitole 1.2, pro invertabilitu musí být funkce f prostá a musí pokrývat celý obor hodnot, x a y tedy musí být vzájemně jednoznačné. Např. funkce $y = x^2$ není prostá v celém definičním oboru $\mathcal{D} = \mathbb{R}$. Pokud ji však omezíme na $\mathbb{R}^+$, prostá již je a jí odpovídající inverzní funkce je $f^{-1}(x) = \sqrt{y}$.

Poznamenejme, že je lhostejné, jakými písmeny označíme argument a obraz, určující pro funkci $\sqrt{y}$ je $\sqrt{}$. Protože je zvykem používat x pro argument a y pro obraz, píšeme obvykle $y = f^{-1}(x) = \sqrt{x}$, a nikoli $x = f^{-1}(y) = \sqrt{y}$. Pokud takto zakreslíme f i f^{-1} do společného grafu, jsou oba grafy symetrické podle osy prvního a třetího kvadrantu ($y = x$), poněvadž jsme tak zaměnili x a y.

Nakresleme do společného grafu funkci $\sin x$ v intervalu $[-\pi/2, \pi/2]$ a funkci k ní inverzní, arcussinus ($\arcsin x$).



Obrázek 2.1.: Idea inverzní funkce



Vidíme, že obě funkce jsou skutečně symetrické podle osy prvního a třetího kvadrantu.

2.1.2. Transformace funkcí

Drobnou úpravou funkce můžeme dosáhnout úpravy jejího „tvaru“, např. posunout na osách x a y nebo roztáhnout či zúžit. Mějme funkci $y = f(x)$. Novou, transformovanou funkci označme jako $g(x)$.

1. **Posunutí na ose y** o konstantu c nahoru dosáhneme přičtením c, $g(x) = f(x) + c$
2. **Posunutí na ose x** o konstantu c doleva dosáhneme přičtením c v argumentu funkce, $g(x) = f(x + c)$
3. **Zvětšení na ose y** k-krát dosáhneme vynásobením funkce konstantou k, $g(x) = kf(x)$
4. **Zkrácení na ose x** k-krát dosáhneme vynásobením argumentu funkce konstantou k, $g(x) = f(kx)$

Kombinací všech úprav získáme funkci

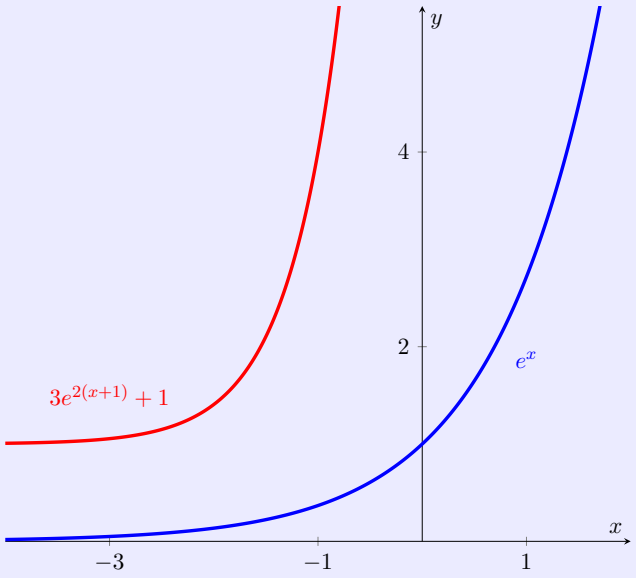
$$g(x) = af(bx + c) + d,$$

která je oproti původní funkci f posunuta o d nahoru, a-krát roztažena na ose y, b-krát užší na ose x a posunuta na ose x o c/b doleva.

Z funkce $f(x) = e^x$ vytvořme funkci $g(x)$, která je oproti $f(x)$ posunuta o 1 nahoru, o 1 doleva, dvakrát užší na ose x a třikrát „roztažena“ na ose y.

Řešením je funkce

$$g(x) = 3e^{2(x+1)} + 1 = 3e^{2x+2} + 1.$$



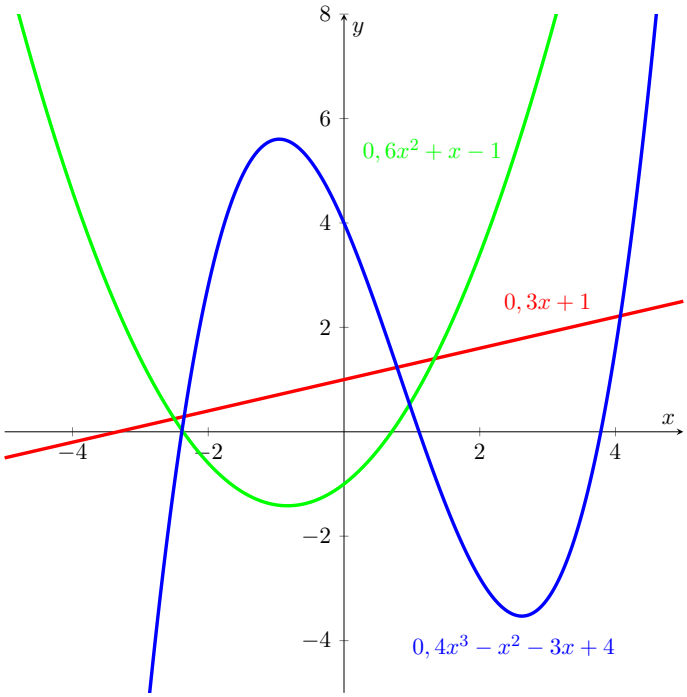
2.2. Přehled základních funkcí

Známe řadu tzv. elementární funkcí, z nichž jsou „sestaveny“ ostatní funkce. Funkce $x^2 + \sin x$ je například „sestavena“ z funkcí x^2 a $\sin x$. Existují však i funkce, které nelze takovou kombinací vyjádřit. Např. primitivní funkce (neurčitý integrál) ke Gaussově „zvonové“ funkci, kterou známe ze statistiky, není vyjádřitelná pomocí elementárních funkcí, ačkoli existuje. Jiné funkce, např. exponenciální nebo goniometrické, mohou být formálně definovány pomocí nekonečné řady mocninných funkcí, jak zmíníme později. Dále je uveden krátký přehled vybraných elementárních funkcí, s nimiž se běžně setkáváme.

2.2.1. Polynomické funkce

Příklady polynomických funkcí 1.-3. řádu ukazuje obr. 2.2. Nejjednodušší polynomickou funkcí je **konstantní funkce** $y = c$. Další v řadě je **lineární funkce** $y = ax + b$. Protíná osu y v hodnotě b, kdy platí $f(0) = a.0 + b = b$, a osu x v bodě $-b/a$, kdy platí $0 = ax + b$. a označujeme jako směrnici přímky, platí $a = \tan\varphi$, značí rychlost růstu přímky. Lineární funkce je prvního řádu, t.j. nejvyšší exponent u x je 1, a funkce protíná osu x v právě 1 bodě.

Kvadratická funkce $y = ax^2 + bx + c$ je druhého řádu a protíná osu x nejvýše ve 2 bodech,



Obrázek 2.2.: Polynomické funkce 1.-3. řádu

tzv. kořenech kvadratické rovnice

$$ax^2 + bx + c = 0.$$

Jsou jimi

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}.$$

V oboru komplexních čísel má kvadratická rovnice vždy 2 řešení. Křivka kvadratické funkce se označuje jako parabola.

1. Určeme průsečíky funkce $y = -2x^2 + 2x + 24$ s osou x.

   Pro průsečíky musí platit

   $$y = -2x^2 + 2x + 24 = -2\left(x^2 - x - 12\right) = -2\,(x - 4)\,(x + 3) = 0.$$

   Funkce proto protíná osu x v bodech $x_1 = 4$ a $x_2 = -3$

2. V kterých bodech se protínají funkce $f(x) = x^2 - x - 2$ a $g(x) = 3x + 5$?

   Musí platit

   $$x^2 - x - 2 = 3x + 5 \rightarrow x^2 - 4x - 7 = 0.$$

   Funkce se proto protínají ve 2 bodech

   $$x_{1,2} = \frac{4 \pm \sqrt{(-4)^2 - 4.(-7)}}{2} = 2 \pm \sqrt{11}.$$

3. Řesme rovnici $f(x) = x^2 + 2x + 2 = 0$ v obou komplexních čísel.

Řešením jsou 2 komplexně sdružené kořeny

$$x_{1,2} = \frac{-2 \pm \sqrt{4-8}}{2} = \frac{-2 \pm \sqrt{-4}}{2} = \frac{-2 \pm \sqrt{4i^2}}{2} = \frac{-2 \pm 2i}{2} = -1 \pm i.$$

Kubická funkce $y = ax^3 + bx^2 + cx + d$ je třetího řádu a protíná osu x nejvýše ve 3 bodech, kořenech kubické rovnice

$$ax^3 + bx^2 + cx + d = 0.$$

S rostoucím řádem funkce je „stále obtížnější vymýšlet názvy koeficientů“ a je snazší používat indexy. Můžeme tedy napsat

$$y = a_3x^3 + a_2x^2 + a_1x + a_0.$$

Polynom n-tého stupně je

$$y = a_nx^n + a_{n-1}x^{n-1} + \ldots + a_2x^2 + a_1x + a_0.$$

Pokud použijeme symbol součtu $\sum$, můžeme stejný polynom elegantně zapsat jako

$$y = \sum_{i=0}^{n} a_ix^i.$$

Indexy u symbolu $\sum$ znamenají, že za i postupně dosazujeme hodnoty od 0 do n. Jednotlivé dvojice a_ix^i pak sečteme.

Polynom n-tého stupně má n kořenů v oboru komplexních čísel. Pokud máme v rovině n+1 bodů, lze najít polynom n-tého stupně, který bude všemi body přesně procházet.

Poznamenejme, že pro součin má obdobný význam symbol $\prod$. Např.

$$\prod_{i=1}^{n} i = 1.2.3.....(n-1).n = n!$$

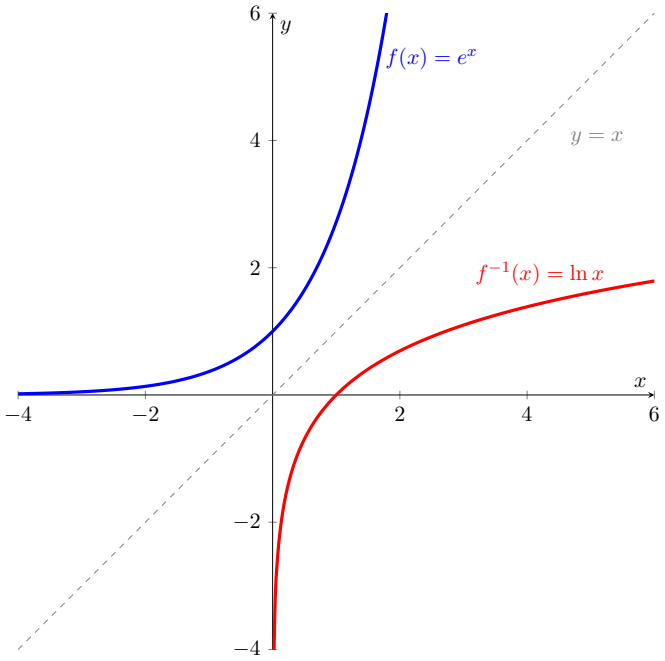
2.2.2. Exponenciální a logaritmické funkce

Řadu biologicky významných procesů, jako např. eliminaci léku ledvinami či rozpad radiofarmaka, lze popsat **exponenciální funkcí** $y = a^x$. Jako a se často používá Eulerovo číslo[1] $e \approx 2,718$, tedy $y = \mathrm{e}^x$. Exponenciální funkce, viz obr. 2.3, roste extrémně rychle. Inverzní k exponenciální funkci je **logaritmická funkce** $y = \log_a x$, viz obr. 2.3. a se označuje jako základ logaritmu. Platí

$$y = a^x \Leftrightarrow x = \log_a y.$$

Pokud $a = e$, označujeme logaritmus jako přirozený a značíme ln, tedy $y = \ln x$. Dekadický logaritmus má základ $a = 10$ a značí se log. Biologům je logaritmus dobře známý z definice pH; $\mathrm{pH} = -\log[\mathrm{H}^+]$.

[1] Leonhard Euler, 1707-1783, švýcarský matematik a fyzik



Obrázek 2.3.: Exponenciální a logaritmická funkce

Též dobře známé jsou vztahy pro logaritmus součinu a podílu

$$\ln xy = \ln x + \ln y$$
$$\ln \frac{x}{y} = \ln x - \ln y$$

Pro zábavu odvodíme vztah pro součet. Protože $\mathrm{e}^{\ln x} = x$, platí

$$\mathrm{e}^{\ln x + \ln y} = \mathrm{e}^{\ln x}.\mathrm{e}^{\ln y} = xy = \mathrm{e}^{\ln xy}.$$

Protože je e^x prostá funkce, plyne z rovnosti exponentů dokazovaný vztah.

V následujícím příkladu si ukážeme jednak práci s logaritmy, jednak důležitost vhodného zanedbání.

Jaké je pH roztoku kyseliny octové CH_3COOH o koncentraci $c = 0.0001$ mol/l a disociační konstantě $K_a = 1,75.10^{-5}$?

V roztoku kysliny octové souběžně probíhá částečná disociace kyseliny a částečná disociace vody podle následujících rovnic:

$$CH_3COOH \overset{K_A}{\rightleftharpoons} CH_3COO^- + H^+$$
$$H_2O \overset{K_w}{\rightleftharpoons} H^+ + OH^-$$

Pojďme dále pro stručnost zápisu označovat CH_3COO^- jako A^- a CH_3COOH jako HA. Rovnice tedy zní $HA \overset{K_A}{\rightleftharpoons} A^- + H^+$.
Veškeré H+ vznikají první nebo druhou reakcí. Z první reakce ve stejném množství vzniká

A^-, z druhé OH^-. Musí proto platit

$$[H^+] = [A^-] + [OH^-].$$

Tato podmínka je ekvivalentní požadavku elektroneutrality. Ze zákona zachování dále plyne podmínka $c = [HA] + [A^-]$.
Disociační konstanta, jak známo, je rovnovážná konstanta, která charakterizuje disociaci kyseliny takto:

$$K_a = \frac{[H^+].[A^-]}{[HA]}.$$

Pokud použijeme pravidla pro logaritmování a vynásobíme celou rovnici -1, dostaneme

$$-\log K_a = -\log [H^+] - \log \frac{[A^-]}{[HA]}.$$

Podle definice je přitom $pH = -\log [H^+]$ a $pK_a = -\log K_a$. Získali jsme tedy rovnici

$$pH = pK_a + \log \frac{[A^-]}{[HA]},$$

což je v biologii a medicíně dobře známá Henderson-Hasselbalchova rovnice. Ta nám ovšem pro naši úlohu bohužel není nic platná.
Obdobně iontový součin vody $K_w = 10^{-14}$ charakterizuje disociaci vody rovnicí $K_w = [H^+].[OH^-]$.
Dostáváme tak soustavu 4 rovnic, které všechny musí platit zároveň:

$$\begin{aligned} [H^+] &= [A^-] + [OH^-] \\ c &= [HA] + [A^-] \\ K_a &= \frac{[H^+].[A^-]}{[HA]} \\ K_w &= [H^+].[OH^-] \end{aligned}$$

Pokud dosadíme za proměnné z jedné rovnice do druhé, dostaneme pro $[H^+]$ obtížně řešitelnou kubickou rovnici. Pomůže nám ale vhodné zanedbání. Uvědomme si, že voda disociuje mnohem méně než kyselina octová, protože $K_w \ll K_a$. Proto je také $[OH^-] \ll [A^-]$ a téměř platí $[H^+] = [A^-]$. Tím se nám rovnice zjednoduší na

$$\begin{aligned} [H^+] &= [A^-] \\ K_a &= \frac{[H^+].[A^-]}{c - [A^-]} \end{aligned}$$

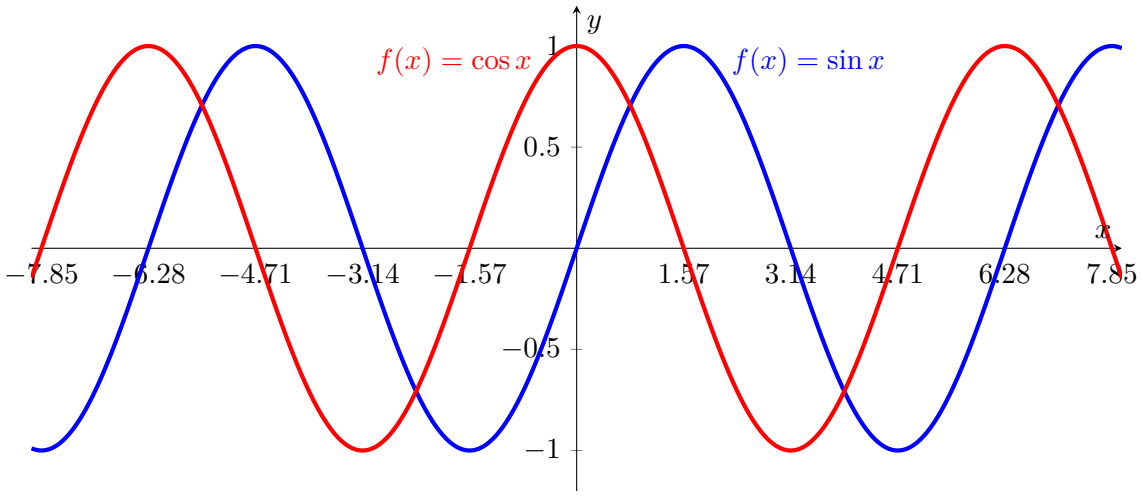
Pak už máme

$$K_a = \frac{[H^+]^2}{c - [H^+]},$$

odkud pro $[H^+]$ platí

$$[H^+] = \frac{1}{2}\left(-K_a + \sqrt{K_a^2 + 4K_a c}\right).$$

Hodnota $[H^+]$ tedy vychází $3,4.10^{-5}$ mol/l a $pH = 4{,}77$.
Kdybychom šli se zanedbáním ještě dále a mysleli si, že disociuje pouze velmi malá část kyseliny, mohli bychom psát $c - [A^-] \approx c$. Pak je rovnice vůbec jednoduchá: $[H^+]^2 = K_a c$ a $[H^+] = \sqrt{K_a c}$. pH pak vychází 4,38. To už je chyba poměrně značná. Předpoklad o



Obrázek 2.4.: Goniometrické funkce

velmi nízké disociaci totiž nebyl opodstatněný. Ale jako tzv. „kvalifikovaný odhad“ to není špatné.

2.2.3. Goniometrické funkce

Pro popis periodických dějů se používají goniometrické funkce sinus a cosinus (obr. 2.4), které se liší pouze posunutím o $\pi/2$ na ose x. Obě funkce jsou periodické s periodou 2π. Připomeňme si krásný již zmíněný Eulerův vztah mezi komplexními čísly, goniometrickými funkcemi a exponenciální funkcí

$$\cos\varphi + i\sin\varphi = \mathrm{e}^{i\varphi}.$$

2.3. Limita funkce

Limita funkce je první téma tradičně řazené do vyšší matematiky.

Intuitivně zformulováno představuje **limita funkce v bodě** x_0 hodnotu, jakou **by** funkce v bodě x_0 nabyla, **kdyby** se v tomto bodě chovala stejně, jako se chová v jeho blízkém okolí. Limitu funkce f „pro x jdoucí k x_0“ značíme

$$\lim_{x\to x_0} f(x).$$

Limita tedy nemluví o skutečném chování funkce v bodě, nýbrž o hypotetickém chování, které plyne z chování funkce v blízkém okolí.

Z grafů na obr. 2.5 je zřejmých několik možností „vztahu“ limity v bodě k průběhu funkce:

(a) Funkce je v bodě x_0 definovaná a je spojitá. Zprava i zleva se funkce blíží k hodnotě $f(x_0)$. Limita funkce v tomto bodě je tedy stejná jako hodnota funkce sama, $\lim_{x\to x_0} f(x) = f(x_0)$.

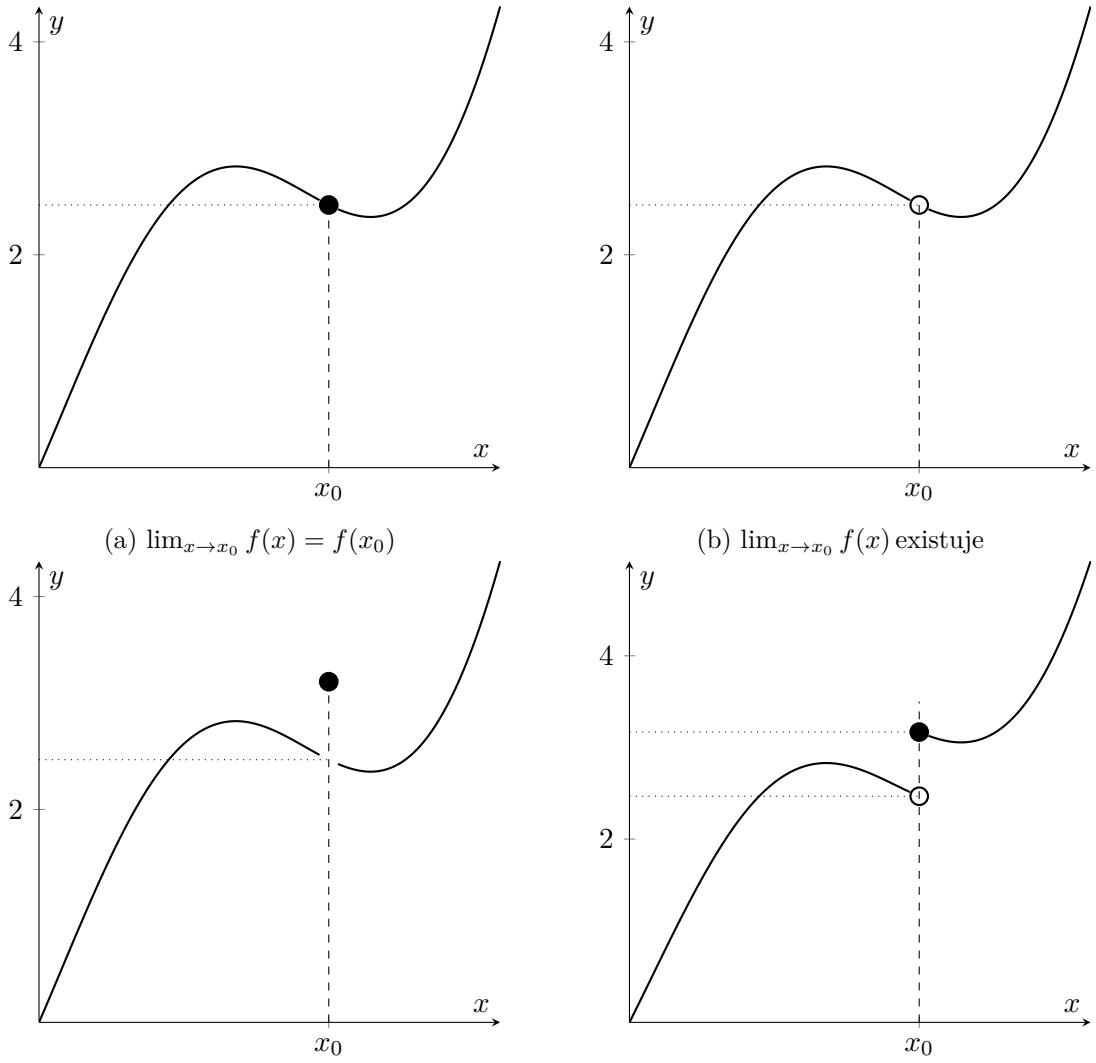
(b) Funkce není v bodě x_0 definovaná, ale míří zleva i zprava ke stejné hodnotě. Pak má v bodě limitu, ačkoli zde není definovaná. To pro limitu není problém, poněvadž odvozuje hodnotu v bodě na základě chování v jeho okolí.

Příkladem je funkce $\frac{\sin x}{x}$ v bodě nula. Tam funkce pochopitelně není definovaná, ale přesto zde má limitu. Jak později uvidíme,

$$\lim_{x \to 0} \frac{\sin x}{x} = 1$$

(c) Funkce je v bodě x_0 definovaná, má ale jinou hodnotu, než předpovídá její chování v okolí. Proto má limita jinou hodnotu, než je hodnota funkce, $\lim_{x \to x_0} f(x) \neq f(x_0)$

(d) Funkce je v bodě x_0 definovaná, ale chování funkce zprava a zleva se liší (liší se limita zprava a limita zleva). Proto nelze limitu definovat.



(a) $\lim_{x \to x_0} f(x) = f(x_0)$

(b) $\lim_{x \to x_0} f(x)$ existuje

(c) $\lim_{x \to x_0} f(x) \neq f(x_0)$

(d) $\lim_{x \to x_0} f(x)$ není definovaná

Obrázek 2.5.: Různé případy limity funkce v bodě x_0

Ve většině případů je určení limity snadné, stačí pouze dosadit x_0 za x, např.

$$\lim_{x \to 0} x^3 + 2x^2 + 4 = 0^3 + 2.0^2 + 4 = 4.$$

Jindy však prosté dosazení vede k neurčitému v $\frac{0}{0}$ nebo $\frac{\infty}{\infty}$. Např.

$$\lim_{x\to 1} \frac{x^2-2x+1}{x^2-1} = \frac{0}{0} = ?$$

Pak je třeba výraz nejprve vhodně upravit.

Určeme následující limitu.

$$\lim_{x\to 1} \frac{x^2-2x+1}{x^2-1} = \lim_{x\to 1} \frac{(x-1)^2}{(x+1)(x-1)} = \lim_{x\to 1} \frac{x-1}{x+1} = \frac{0}{2} = 0$$

2.4. Derivace funkce

Limita funkce by byla akademickou záležitostí, kdyby nebylo jedné zásadní aplikace, totiž derivace funkce. Derivace funkce je speciálním případem limity. Určuje rychlost růstu funkce. Nejprve si ukážeme, co derivace je, poté jak ji spočítáme, a nakonec k čemu je užitečná.

2.4.1. Intuitivní představa derivace

Chceme posoudit, jak rychle narůstá funkce $f(x)$ v bodě x_0, viz obr. 2.6. Řešení je v principu zřejmé: přiložíme k funkci v bodě $[x_0, f(x_0)]$ tečnu (červená přímka na obrázku) a určíme úhel α, který svírá s osou x, případně její směrnici $\tan\alpha$. Jak ale tuto tečnu a její směrnici určit? Můžeme najít přibližné řešení. Z obrázku je vidět, že směrnice $\tan\alpha$ modré úsečky, spojující body $[x_0, f(x_0)]$ a $[x_0+\Delta x, f(x_0+\Delta x)]$, je jen o trochu vyšší než sklon tečny. Platí tedy

$$\tan\alpha \approx \tan\alpha' = \frac{\Delta y}{\Delta x} = \frac{f(x_0+\Delta x) - f(x_0)}{\Delta x}.$$

Dále je jasné, že se modrá a červená přímka budou postupně přibližovat, jak klesá Δx, až v limitním přechodu pro $\lim_{x\to x_0}$ splynou.

$$\tan\alpha = \lim_{\Delta x\to 0} \frac{\Delta y}{\Delta x} = \lim_{\Delta x\to 0} \frac{f(x_0+\Delta x) - f(x_0)}{\Delta x}$$

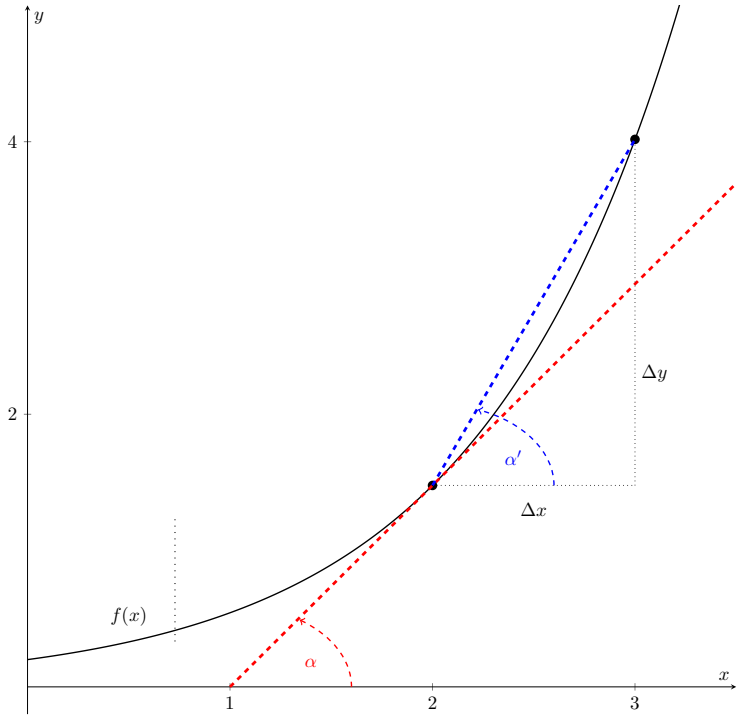
2.4.2. Exaktní definice derivace

V definici derivace se často místo symbolu Δx používá h.

Derivace funkce v bodě x se značí $f'(x)$ nebo $\frac{\mathrm{d}f(x)}{\mathrm{d}x}$. Derivace funkce f v bodě x je definovaná jako

$$f'(x) \equiv \frac{\mathrm{d}f(x)}{\mathrm{d}x} := \lim_{h\to 0} \frac{f(x+h) - f(x)}{h}$$

Symboly $\equiv$ i $:=$ znamenají definiční rovnost, tedy vlastně pojmenování něčeho něčím. Nejde o rovnici, kterou bychom měli řešit.



Obrázek 2.6.: Derivace funkce $f(x)$ v bodě $x_0 = 2$

Derivace je právě tím případem limity, kdy prosté dosazení vede k neurčitému výrazu typu $\frac{0}{0}$. Je proto nejprve potřeba použít nějakou úpravu či „trik“. Pro derivaci funkce x^2 nap59klad vychází

$$\frac{\mathrm{d}x^2}{\mathrm{d}x} = \lim_{h\to 0} \frac{(x+h)^2 - x^2}{h} = \frac{(x+0)^2 - x^2}{h} = \frac{0}{0} = ?$$

Po předchozí úpravě však dostaneme

$$\frac{\mathrm{d}x^2}{\mathrm{d}x} = \lim_{h\to 0} \frac{(x+h)^2 - x^2}{h} = \lim_{h\to 0} \frac{x^2 + 2hx + h^2 - x^2}{h} = \lim_{h\to 0} \frac{2hx + h^2}{h} = \lim_{h\to 0} 2x + h = 2x$$

Symbol **d**, který se používá ve značení derivace $\frac{\mathrm{d}f(x)}{\mathrm{d}x}$, vznikl jako vyjádření velmi malého Δ, $\mathrm{d}x = \lim_{\Delta x\to 0} \Delta x$. Derivace je tak podíl dvou velmi malých rozdílů. d se často označuje jako diferenciál, ačkoli tento pojem má v matematice přesnější význam. Někdy lze s těmito diferenciály „normálně počítat“, jak uvidíme u derivace inverzní funkce.

Všimněme si, že když funkci zderivujeme v každém bodě, dostaneme v každém bodě nějakou hodnotu, čímž je vlastně definována nová funkce. Derivace tedy přiřazuje jedné funkci jinou funkci, jde o zobrazení. Zatímco je funkce zobrazení z množiny čísel do množiny čísel, je derivace zobrazení z množiny funkcí do množiny funkcí. To pro nás však není úplná novinka, vždyť např. násobení funkce konstantou nebo umocnění funkce na druhou je též zobrazení z množiny funkcí do množiny funkcí.

2.4.3. Derivace elementárních funkcí

Lze odvodit následující vztahy pro derivace elementárních funkcí. k je libovolná konstanta.

$$\begin{aligned} k' &= 0 \\ [k.f(x)]' &= k.f'(x) \\ (x^n)' &= n.x^{n-1} \\ (\sin x)' &= \cos x \\ (\cos x)' &= -\sin x \\ (e^x)' &= e^x \end{aligned}$$

Zde vidíme unikátnost Eulerova čísla e jako základu exponenciální funkce. Derivace exponenciální funkce se základem e má totiž v každém bodě stejnou hodnotu jako funkce sama.

Určeme derivaci funkce $5x^3$.

$$\left(5x^3\right)' = 5.\left(x^3\right)' = 5.(3x^2) = 15x^2$$

2.4.4. Derivace součtu, rozdílu, součinu a podílu

Pro derivaci součtu, rozdílu, součinu a podílu platí

$$\begin{aligned} [f(x)+g(x)]' &= f'(x)+g'(x) \\ [f(x)-g(x)]' &= f'(x)-g'(x) \\ [f(x).g(x)]' &= f'(x).g(x)+f(x).g'(x) \\ \left[\frac{f(x)}{g(x)}\right]' &= \frac{f'(x).g(x)-f(x).g'(x)}{g^2(x)} \end{aligned}$$

Derivace součtu

$$\left(\sin^2 x+\cos^2 x\right)' = 2\sin x.\cos x + 2\cos x.(-\sin x) = 0$$

Výsledek nás samozřejmě nepřekvapil. Však víme, že derivace konstanty je nula a že

$$\sin^2 x+\cos^2 x = 1.$$

Derivace součinu
Podobně si ověříme pravidlo pro derivaci mocninné funkce, neboť

$$\left(x^5.x^4\right)' = 5x^4.x^4 + x^5.4x^3 = 5x^8+4x^8 = 9x^8 = \left(x^9\right)'.$$

Derivace podílu

$$\left(\frac{e^x}{\sin x}\right)' = \frac{e^x\sin x - e^x\cos x}{\sin^2 x}$$

Vidíme, že derivace součtu (či rozdílu) je součet derivací a derivace násobení konstantou je násobení derivace konstantou. Kombinace těchto dvou vlastností se v matematice vyskytuje často, je výsadní vlastností a je označovaná jako **linearita**. Derivace je tedy **lineární zobrazení**. Linearita je příjemná vlastnost, protože zjednodušuje výpočty. Naproti tomu např. „umocnění funkce na druhou“ není lineární zobrazení, protože $(x+y)^2 \neq x^2 + y^2$.

2.4.5. Derivace složených funkcí

Složenou funkcí $f(g(x))$ se rozumí kombinace dvou (nebo více) funkcí $f(x)$ a $g(x)$, kdy jedna z nich je „uvnitř“ druhé. Výsledek vnitřní funkce je argumentem vnější funkce. Ve výrazu $f(g(x))$ je f vnější a g vnitřní funkce.

Např. funkce $\sin x^2$ je složená, vnitřní funkce je x^2, vnější funkce je sin. Argumentem vnější funkce je celý výraz x^2.

Složenou funkci zderivujeme takto:

$$(f(g(x)))' \equiv \frac{\mathrm{d}f(g(x))}{\mathrm{d}x} = f'(g(x)).g'(x)$$

Slovy formulováno: „Derivace složené funkce je derivace vnější funkce krát derivace vnitřní funkce“. Vnější funkci přitom derivujeme podle vnitřní funkce a vnitřní funkci podle proměnné. Pravidlo je nutno uplatnit opakovaně, pokud je vnořeno více úrovní funkcí.

Zní to velmi komplikovaně. Pokud se však na derivaci díváme jako na podíl dvou diferenciálů, s nimiž, jak jsme již zmínili, můžeme „normálně počítat“, pak komplikovanost rychle zmizí. Zkusme vynásobit derivaci „speciální“ jedničkou, $1 = \frac{\mathrm{d}y}{\mathrm{d}y}$, a diferenciály ve jmenovatelích přeskupit. Výsledek vnitřní funkce označme jako y, tedy $y = g(x)$.

$$\frac{\mathrm{d}f}{\mathrm{d}x} = \frac{\mathrm{d}f}{\mathrm{d}x}.1 = \frac{\mathrm{d}f}{\mathrm{d}x}.\frac{\mathrm{d}y}{\mathrm{d}y} = \frac{\mathrm{d}f}{\mathrm{d}y}.\frac{\mathrm{d}y}{\mathrm{d}x}$$

To už je jasné. $\frac{\mathrm{d}f}{\mathrm{d}y}$ je derivace vnější funkce podle vnitřní funkce, $\frac{\mathrm{d}y}{\mathrm{d}x}$ je derivace vnitřní funkce podle proměnné.

Derivace složené funkce
Zderivujme funkci e^{x^2-2x}. Vnější funkce je „e na něco“, její vnitřní funkcí je $x^2 - 2x$.

$$\left(e^{x^2-2x}\right)' = e^{x^2-2x}.(2x-2)$$

Nyní budeme derivovat funkci $\sin^3 x^2$. Tato funkce je složena ze 3 funkcí. Vnější funkcí je „něco na třetí“, její vnitřní funkcí je sin, který má dále svou vlastní vnitřní funkci x^2. Opakovaně aplikuje pravidlo „derivace vnější funkce krát derivace vnitřní funkce“.

$$\left(\sin^3 x^2\right)' = 3\sin^2 x^2.\left(\sin x^2\right)' = 3\sin^2 x^2.\cos x^2\left(x^2\right)' = 3\sin^2 x^2.\cos x^2.2x$$

2.4.6. Derivace inverzních funkcí

Derivace funkce $f^{-1}(x)$ inverzní k funkci $f(x)$ je rovna převrácené hodnotě derivace funkce $f(x)$ v bodě $f^{-1}(x)$.

$$\left(f^{-1}(x)\right)' = \frac{1}{f'\left(f^{-1}(x)\right)}$$

To zní opět velmi komplikovaně. Trik s podílem dvou diferenciálů vše opět ozřejmí. Pokud $y = f^{-1}(x)$, pak $x = f(y)$ a platí

$$\frac{\mathrm{d}y}{\mathrm{d}x} = \frac{1}{\frac{\mathrm{d}x}{\mathrm{d}y}}.$$

$\frac{\mathrm{d}y}{\mathrm{d}x}$ je derivace funkce f^{-1}, kterou zde považujeme za „inverzní“, a $\frac{\mathrm{d}x}{\mathrm{d}y}$ je derivace funkce f. Pouze je třeba dosadit odpovídající hodnoty argumentů obou funkcí, jak objasní následující příklad.

Derivace logaritmu
Princip si nejprve ukážeme na derivaci logaritmu $\ln x$, tedy inverzní funkci k e^x, kterou už zderivovat umíme.

$$(\ln x)' = \frac{1}{(e^x)'\,[\ln x]} = \frac{1}{e^x[\ln x]} = \frac{1}{e^{\ln x}} = \frac{1}{x}$$

Výraz v hranatých závorkách znamená „v bodě“, např. $[\ln x]$ znamená v bodě $[\ln x]$.
Derivace odmocniny Podobně můžeme zderivovat druhou odmocninu, která je pro $x \geq 0$ inverzní funkcí k x^2.

$$\left(\sqrt{x}\right)' = \frac{1}{(x^2)'\,[\sqrt{x}]} = \frac{1}{2x[\sqrt{x}]} = \frac{1}{2\sqrt{x}}$$

Odmocninu můžeme derivovat i jako mocninou funkci podle vztahu $(x^n)' = nx^{n-1}$, čímž si ověříme správnost předchozího výsledku.

$$\left(\sqrt{x}\right)' = \left(x^{\frac{1}{2}}\right)' = \frac{1}{2}\left(x^{-\frac{1}{2}}\right) = \frac{1}{2\sqrt{x}}$$

2.4.7. Derivace vyššího řádu

Dosud jsme popsali derivaci prvního řádu, tzv. první derivace, která popisuje rychlost růstu funkce. Derivace sama je ovšem též funkcí a je proto (většinou) možné ji znovu derivovat. Získáme tak druhé, třetí a další derivace, které se značí $f''(x)$, $f'''(x)$ nebo $\frac{\mathrm{d^2f(x)}}{\mathrm{d}x^2}$, $\frac{\mathrm{d^3f(x)}}{\mathrm{d}x^3}$. N-tá derivace je $f^{(n)}x$ nebo $\frac{\mathrm{d^nf(x)}}{\mathrm{d}x^n}$.

Třetí derivace

$$\frac{\mathrm{d}^3 xe^x}{\mathrm{d}x^3} \equiv [xe^x]''' = [e^x + xe^x]'' = [2e^x + xe^x]' = 3e^x + xe^x = e^x\,(x+3)$$

2.4.8. Geometrický a fyzikální význam derivace

Geometrický význam derivace první derivace jsme již naznačili. Jde o směrnice tečny ke grafu funkce, určuje tedy strmost nárůstu funkce v daném bodě.

Pokud funkce popisuje závislost nějaké proměnné na čase, určuje derivace okamžitou rychlost změny této proměnné v daný okamžik. Pokud je proměnnou například dráha $s(t)$, kterou cyklista do doby t ujel, pak okamžitá rychlost cyklisty v čase t je

$$v = \frac{\mathrm{d}s(t)}{\mathrm{d}t}.$$

Pokud $c(t)$ je koncentrace léčiva v plazmě v čase t, pak

$$-\frac{\mathrm{d}c(t)}{\mathrm{d}t}.$$

je okamžitá rychlost poklesu plazmatické koncentrace léčiva.

Fyzikálním významem druhé derivace je zrychlení. Nechť je $s(t)$ opět dráha, kterou cyklista ujel do doby t. Jeho okamžitá rychlost je

$$v = \frac{\mathrm{d}s(t)}{\mathrm{d}t}.$$

Zrychlení a je nárůst okamžité rychlosti v čase, proto

$$a = \frac{\mathrm{d}v(t)}{\mathrm{d}t} = \frac{\mathrm{d}^2 s(t)}{\mathrm{d}t^2}.$$

Druhá derivace je pro fyziku velmi důležitá, protože jeden ze základních zákonů fyziky, druhý Newtonův zákon, popisuje vztah síly a zrychlení, tedy síly a druhé derivace polohy.

$$F = m.a = m.\frac{\mathrm{d}^2 s(t)}{\mathrm{d}t^2}$$

Druhá derivace též popisuje oscilace a vlnění, jak uvidíme později.

Geometrickým významem druhé derivace je míra „zakřivení“ funkce, tzv. konvexity (nebo konkavity) funkce. Popisuje, jak rychle se mění chování funkce. Například funkce $y = ax+b$ popisuje přímku, parametr a udává její strmost. Chování přímky se ale nemění, roste pořád stejně. Tomu odpovídá skutečnost, že druhá derivace je nulová a nezávisí na strmosti přímky. Pokud se strmost funkce, která je sama první derivací, naopak rychle mění, musí mít derivace strmosti, tedy vlastně druhá derivace funkce, vysokou hodnotu.

2.4.9. Vyšetřování průběhu funkce

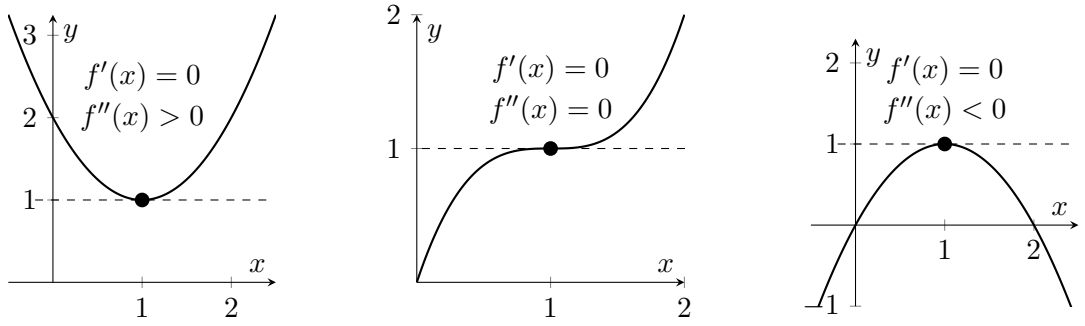
Věnujme se nyní možnostem vyšetření průběhu funkce pomocí první a druhé derivace. Nulová první derivace v bodě x_0 značí, že v tomto bodě nastává jedna ze 3 situací - lokální minimum,

lokální maximum nebo tzv. inflexní bod (obr. 2.7). Situace rozlišíme pomocí druhé derivace.

$$\frac{\mathrm{d}f(x_0)}{\mathrm{d}x} = 0 \text{ a zároveň } \begin{cases} \dfrac{\mathrm{d}^2 f(x_0)}{\mathrm{d}x^2} > 0 \ldots \text{lokální minimum} \\ \dfrac{\mathrm{d}^2 f(x_0)}{\mathrm{d}x^2} = 0 \ldots \text{inflexní bod} \\ \dfrac{\mathrm{d}^2 f(x_0)}{\mathrm{d}x^2} < 0 \ldots \text{lokální maximum} \end{cases}$$

První derivace funkce je v (lokálním) maximu i minimu nulová. V lokálním minimu se tvar funkce, „údolí“, označuje jako konvexní, strmost roste a druhá derivace je kladná. Vysoká kladná druhá derivace znamená ostré minimum, nízká druhá derivace pak ploché minimum. V lokálním maximu se funkce označuje jako konkávní, druhá derivace je záporná. Inflexní bod leží „právě mezi“ maximem a minimem, první i druhá derivace jsou zde proto nulové.

Maximum a minimum se souhrnně označují jako extrémy. Pro úplnost poznamenejme, že rozlišujeme globální a lokální extrémy. Globální extrém je nejvyšší nebo nejnižší bod křivky. Lokální extrémy jsou všechny body, které splňují výše uvedené charakteristiky.



(a) Lokální minimum (b) Inflexní bod (c) Lokální maximum

Obrázek 2.7.: Vyšetření průběhu funkce

Určeme lokální minima a lokální maxima následujících funkcí:

1. $f_1(x) = x^2.\cos x$ pro $x \in [-2, 2]$

   Pro extrémy a inflexní body musí platit $f_1'(x) = 2x\cos x - x^2 \sin x = 0$. Zjevným řešením je $x_1 = 0$. Po vydělením x dostaneme $2\cos x = x \sin x$, což je nelineární rovnice, 2 řešení jsou přibližná, $x_2 \approx 1,08$ a $x_3 = -x_2 \approx -1,08$. Pro zjištění typu extrému potřebujeme hodnotu 2. derivace v bodech x_1, x_2 a x_3.

   $$f_1''(x) = 2\cos x - 2x \sin x - 2x \sin x - x^2 \cos x$$

   Pro $x_1 = 0$ platí $f_1''(0) = 2\cos 0 = 2 > 0$, x_1 je tedy lokální minimum. Upozorňuji, že jde o lokální minimum, nikoli o globální minimum. Pro body x_2 x_3 dosadíme podmínku extrému $2\cos x = x \sin x$ a získáme

   $$f_1''(x) = x \sin x - 2x \sin x - 2x \sin x - x^2 \cos x = -3x \sin x - x^2 \cos x.$$

   Protože x a $\sin x$ mají v intervalu $(-\pi, \pi)$ stejné znaménko, je pro x_2 i x_3 $x \sin x > 0$, stejně tak i $x^2 \cos x > 0$. Proto $f_1''(x_2) < 0$ i $f_1''(x_3) < 0$, oba body jsou tedy lokálními

maximy.

2. $f_2(x) = e^{-(x-4)^2}$

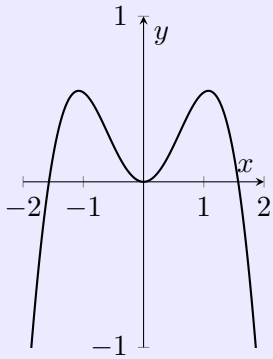
   Jediným řešením podmínky $f_2'(x) = -2(x-4).e^{-(x-4)^2} = 0$ je $x = 4$.

   $$f_2''(x) = -2.e^{-(x-4)^2} + 4(x-4)^2.e^{-(x-4)^2}$$

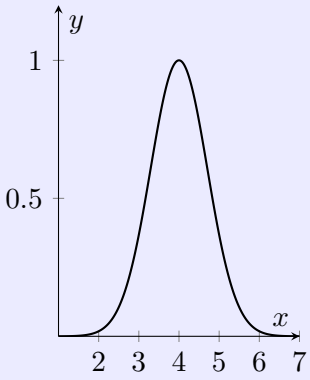
   Pro $x = 4$ je $f_2''(4) = -2.e^0 = -2 < 0$, jde tedy o maximum.

3. $f_3(x) = x^3 + 3x^2 + 1$

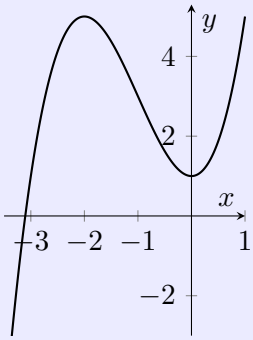
   $f_3'(x) = 3x^2 + 6x = 0$. Řešením jsou $x_1 = 0$ a $x_2 = -2$. Platí $f_3''(x) = 6x + 6$, tedy $f_3''(0) = 6 > 0$ a $f_3''(-2) = -6 < 0$. x_1 je lokální minimum, x_2 lokální maximum.



$f_1(x) = x^2.\cos x$



$f_2(x) = e^{-(x-4)^2}$



$f_3(x) = x^3 + 3x^2 + 1$

2.4.10. Taylorovy řady

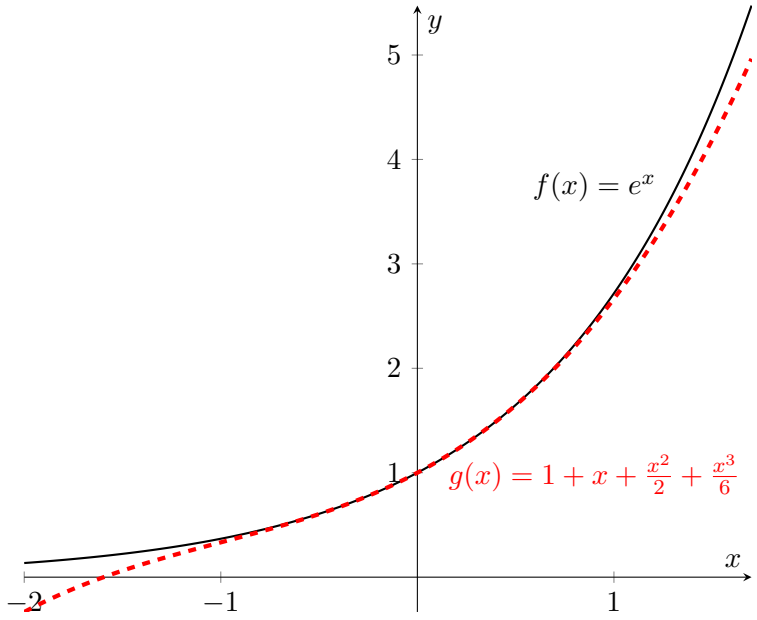
Mějme za cíl přibližně nahradit nějakou složitou funkci $f(x)$ v bodě x_0 a jeho blízkém okolí jinou, jednodušší funkcí $g(x)$. Například funkci $f(x) = e^x$ můžeme v okolí bodu $x_0 = 0$ aproximovat funkcí $g(x) = 1 + x + \frac{x^2}{2} + \frac{x^3}{6}$, jak ukazuje obr. 2.8. Vidíme, že v blízkém okolí bodu $x_0 = 0$ je aproximace téměř přesná.

Taylorovy řady či **Taylorův rozvoj** používají jakožto aproximující funkci $g(x)$ polynom. Pokusme se tedy nyní aproximovat obecnou funkci $f(x)$ v okolí bodu x_0 polynomem řádu n „centrovaným“ do bodu x_0, tedy funkcí

$$g(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)^2 + a_3(x - x_0)^3 \ldots + a_(x - x_0)^n = \sum_{i=0}^{n} a_i(x - x_0)^i.$$

Chceme, aby se v bodě x_0 shodovaly hodnoty obou funkcí i všech jejich (prvních) n derivací. Požadujeme proto platnost následujících rovností:

$$\begin{aligned} f(x_0) &= g(x_0) \\ f'(x_0) &= g'(x_0) \\ f''(x_0) &= g''(x_0) \\ &\vdots \\ f^{(n)}(x_0) &= g^{(n)}(x_0) \end{aligned}$$



Obrázek 2.8.: Aproximace funkce e^x polynomem

Spočítejme všechny potřebné derivace funkce g:

$$\begin{aligned}
g'(x) &= a_1 + 2.a_2(x-x_0) + 3.a_3(x-x_0)^2 \ldots + n.a_n(x-x_0)^{n-1} \\
g''(x) &= 2.a_2 + 3.2.a_3(x-x_0) \ldots + n.(n-1).a_n(x-x_0)^{n-2} \\
g'''(x) &= 3!.a_3 + 4.3.2.a_4(x-x_0) \ldots + n.(n-1).(n-2).a_n(x-x_0)^{n-3} \\
&\vdots \\
g^{(k)}(x) &= k!.a_k + (k+1)!.a_{k+1}(x-x_0) \ldots + n(n-k+1).a_n(x-x_0)^{n-k} \\
&\vdots \\
g^{(n)}(x) &= n!.a_n
\end{aligned}$$

Dosadíme nyní derivace v bodě x_0 do požadovaných rovností. Hrůza opadne, jakmile si uvědomíme, že pro $x = x_0$ jsou všechny členy obsahující $x - x_0$ nulové, čímž zmizí. Dostaneme tedy jednoduché vztahy

$$\begin{aligned}
f(x_0) &= a_0 \\
f'(x_0) &= a_1 \\
f''(x_0) &= 2a_2 \\
f'''(x_0) &= 3!a_3 \\
&\vdots \\
f^{(n)}(x_0) &= n!.a_n
\end{aligned}$$

Tudíž $a_0 = f(x_0)$, $a_1 = f'(x_0)$, $a_2 = \frac{f''(x_0)}{2}$, …, $a_n = \frac{f^{(n)}(x_0)}{n!}$. Dospěli jsme tak k požadovanému

polynomu

$$g(x) = \sum_{i=0}^{n} \frac{f^{(i)}(x_0)}{i!}(x - x_0)^i$$
$$= f(x_0) + f'(x_0)(x - x_0) + \frac{f''(x_0)}{2}(x - x_0)^2 + \frac{f'''(x_0)}{6}(x - x_0)^3 \ldots + \frac{f^{(n)}}{n!}(x_0)(x - x_0)^n$$

Co kdybychom se neomezili na polynom řádu n a shodu v n derivacích, ale použili bychom hned polynom řádu ∞, tedy nekonečnou mocninnou řadu? Opět budeme požadovat shodu v bodě x_0 i ve všech (nekonečně mnoha) derivacích v bodě x_0. Pak dostaneme polynom

$$g(x) = \sum_{i=0}^{\infty} \frac{f^{(i)}(x_0)}{i!}(x - x_0)^i$$

Lze ukázat, že za určitých podmínek pro některé funkce $f(x)$ polynom $g(x)$ konverguje k funkci $f(x)$ ve všech bodech x, tedy že lze funkci $f(x)$ zcela a všech bodech nahradit tímto polynomem $g(x)$. Platí pak

$$f(x) = \sum_{i=0}^{\infty} \frac{f^{(i)}(x_0)}{i!}(x - x_0)^i$$

Funkci $f(x)$ jsme tak rozvinuli v **Taylorovu řadu**.

Ilustrujme si aproximaci polynomem na příkladech.

Aproximace funkce $f(x) = e^x$ v okolí bodu 0.

Spočítejme jednotlivé derivace. Pro funkci $f(x) = e^x$ platí, že $f(x) = e^x$, $f'(x) = e^x$, $f''(x) = e^x$, $\ldots$, $f^{(n)}(x) = e^x$. Pro $x_0 = 0$ jsou tedy všechny derivace rovny 1. Po dosazení do Taylorova polynomu dostaneme

$$e^x \approx 1 + x + \frac{x^2}{2} + \frac{x^3}{6} \ldots + \frac{x^n}{n!} = \frac{x^0}{0!} + \frac{x^1}{1!} + \frac{x^2}{2!} + \frac{x^3}{6} \ldots + \frac{x^n}{n!} = \sum_{i=0}^{n} \frac{x^i}{i!}$$

To se dobře pamatuje. Shodu pro polynom řádu 3 vidíme na obr. 2.8. Funkce e^x je právě jednou z funkcí, které lze zcela nahradit polynomem. Platí

$$e^x = \sum_{i=0}^{\infty} \frac{x^i}{i!}.$$

Speciálně pro $x = 1$ dostaneme vyjádření Eulerova čísla $e = 2,7182818285\ldots$ pomocí nekonečné řady. Můžeme tak toto iracionální číslo vyjádřit s libovolnou přesností.

$$e = \frac{1}{0!} + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} \ldots$$

Prvních 5 členů např. dává součet $1 + 1 + \frac{1}{2} + \frac{1}{6} + \frac{1}{24} = 2,708$, již docela blízký hodnotě e.

Aproximace funkcí $\sin x$ a $\cos x$ v okolí bodu 0.

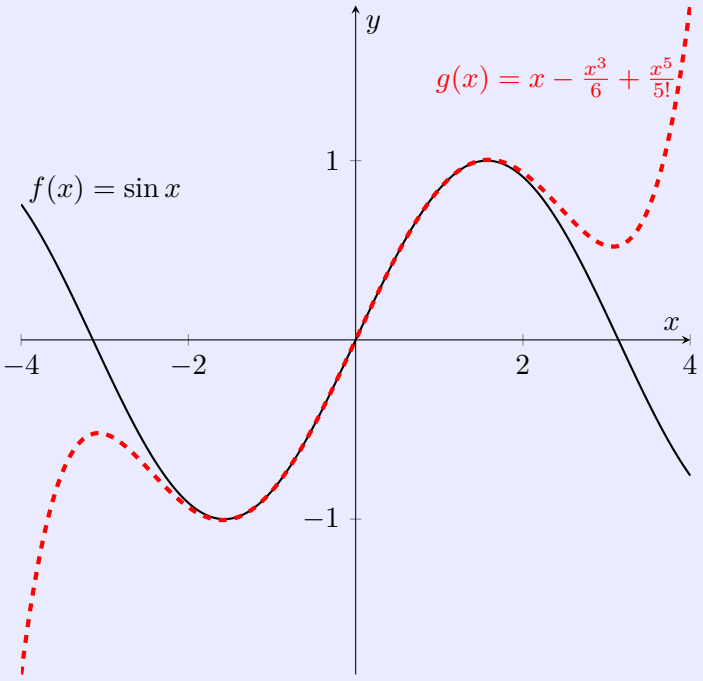
Jednotlivé derivace pro $f(x) = \sin x$ jsou $f'(x) = \cos x$, $f''(x) = -\sin x$, $f^{(3)}(x) = -\cos x$, $f^{(4)}(x) = \sin x$, a dále stále periodicky. Pro $x_0 = 0$ jsou tedy derivace (včetně 0. derivace) postupně rovny 0, 1, 0, -1, 0, 1, 0, -1 .... Po dosazení dostaneme

$$\sin x \approx x - \frac{x^3}{6} + \frac{x^5}{5!} + \dots$$

Shodu pro polynom řádu 5 vidíme na obr. **??**. Vidíme typickou vlastnost polynomu - polynom osciluje a na okraji intervalu rychle uniká k $\pm\infty$. I funkci $\sin x$ lze zcela nahradit polynomem. Platí

$$\sin^x = \sum_{i=0}^{\infty} \sin\left[(i \bmod 4)\frac{\pi}{2}\right]\frac{x^i}{i!}.$$

Pro obecný zápis jsme použili operaci modulo, t.j. zbytek po dělení. $i \bmod 4$ je zbytek po dělení čtyřmi. Výraz $\sin(i \bmod 4)$ tak elegantně poskytuje alternující znamení $0, 1, 0, -1, 0, 1, 0, -1 \dots$.



Pokud nyní zderivujeme Taylorův rozvoj pro $\sin x$, musíme získat Taylorův rozvoj pro $\cos x$, jak si sami můžete ověřit.

$$\cos x = (\sin x)' = \left(x - \frac{x^3}{6} + \frac{x^5}{5!} + \dots\right)' = 1 - \frac{x^2}{2} + \frac{x^4}{4!} + \dots = \sum_{i=0}^{\infty} \cos\left[(i \bmod 4)\frac{\pi}{2}\right]\frac{x^i}{i!}$$

Vidíme, že $\sin x$ obsahuje liché mocniny x a $\cos x$ sudé mocniny.

V posledním příkladu této sekce se pokusíme určit limitu funkce důležité v teorii vlnění a difrakce.

$$\lim_{x\to 0} \frac{\sin x}{x} = ??$$

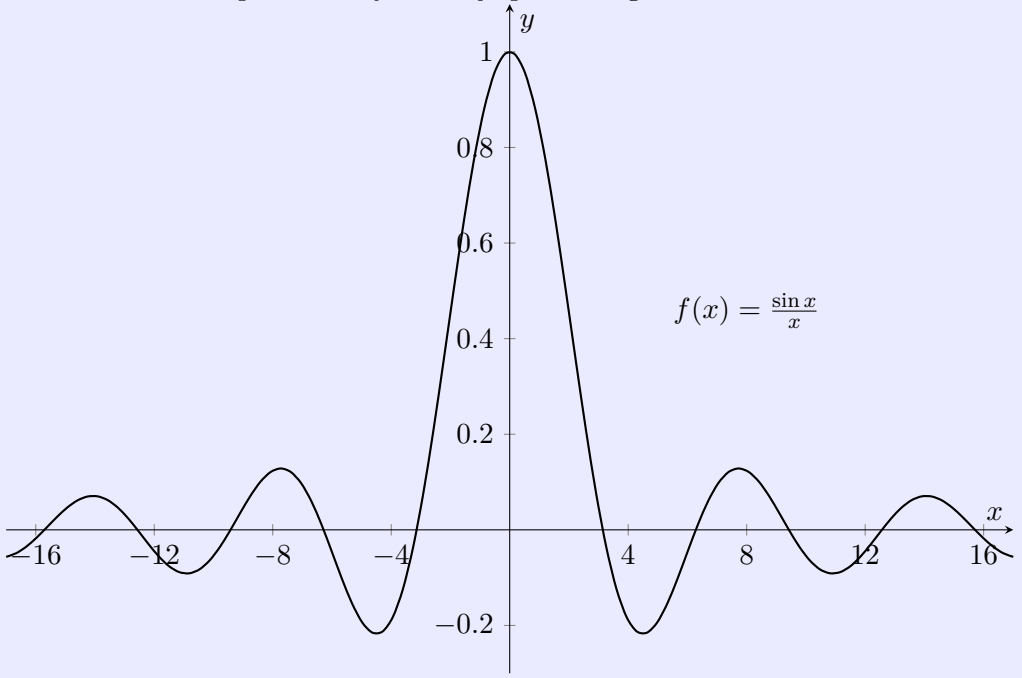
Limitu nemůžeme spočítat přímo dosazením, protože tak získáme neurčitý výraz 0/0.

$$\lim_{x\to 0}\frac{\sin x}{x}=\frac{\sin 0}{0}=\frac{0}{0}=??$$

Protože nás však zajímá chování v okolí bodu 0, může nám pomoci Taylorův rozvoj funkce $\sin x$ v okolí 0.

$$\lim_{x\to 0}\frac{\sin x}{x}=\lim_{x\to 0}\frac{x-\frac{x^3}{6}+\frac{x^5}{5!}+\dots}{x}=\lim_{x\to 0}\left(1-\frac{x^2}{6}+\frac{x^4}{5!}+\dots\right)=1$$

Správnost výsledku je patrna z grafu funkce.



V našem výkladu jsme postupovali tak, jako bychom funkce e^x, $\sin x$ nebo $\cos x$ „už znali“, a hledali jsme jejich Taylorův rozvoj. Přitom jsme si nevšimli, že tyto funkce tak docela „neznáme“. Přinejmenším neumíme bez kalkulačky spočítat jejich hodnotu. Ve formální, „čisté“ matematice se tyto funkce naopak mohou pomocí nekonečných řad definovat. Zkrátka řekneme, že jistý nekonečný polynom nazveme „sinus“ a dále pak zkoumáme vlastnosti této „nové“ funkce. Definujeme tedy

$$\sin x:=\sum_{i=0}^{\infty}\sin\left[(i \bmod 4)\frac{\pi}{2}\right]\frac{x^i}{i!}.$$

Podobně bychom se mohli ptát, který nekonečný polynom se po derivaci nezmění. Zjistili bychom, že to právě ten, který označujeme jako e^x.

Slíbili jsme, že se zdržíme matematických důkazů. V úchvatných případech však musíme učinit výjimku. Zkusme sečíst Taylorovy rozvoje funkcí $\cos x$ a $\sin x$, ale $\sin x$ vynásobme imaginární jednotkou i.

$$\cos x+i\sin x=1-\frac{x^2}{2}+\frac{x^4}{4!}+\dots+ix-i\frac{x^3}{6}+i\frac{x^5}{5!}+\dots=1+ix-\frac{x^2}{2}-i\frac{x^3}{6}+\frac{x^4}{4!}+i\frac{x^5}{5!}+\dots$$

Pokud si uvědomíme, že $i^0 = 1$, $i^1 = i$, $i^2 = -1$, $i^3 = -i$ ..., můžeme výraz přepsat na

$$\frac{i^0x^0}{0!} + \frac{i^1x^1}{1!} + \frac{i^2x^2}{2!} + \frac{i^3x^3}{3!} + \frac{i^4x^4}{4!} + \ldots = \frac{(ix)^0}{0!} + \frac{(ix)^1}{1!} + \frac{(ix)^2}{2!} + \frac{(ix)^3}{3!} + \frac{(ix)^4}{4!} \ldots .$$

Samozřejmě už vidíme, že výraz odpovídá Taylorově rozvoji funkce e^{ix}.

$$e^{ix} = \frac{(ix)^0}{0!} + \frac{(ix)^1}{1!} + \frac{(ix)^2}{2!} + \frac{(ix)^3}{3!} + \frac{(ix)^4}{4!} \ldots$$

Dokázali jsme tak dříve uvedený Eulerův vztah $e^{ix} = \cos x + i \sin x$.

2.5. Funkce více proměnných

Nyní se přesuneme k problematice funkcí více proměnných. Řada představ je pouhým rozšířením z funkcí jedné proměnné, ale některé problémy jsou nové.

Reálná funkce n reálných proměnných přiřazuje n-tici reálných čísel reálné číslo, je to tedy zobrazení

$$f : \mathbb{R}^n \to \mathbb{R} : (x_1, x_2, \ldots, x_n) \to f(x_1, x_2, \ldots, x_n).$$

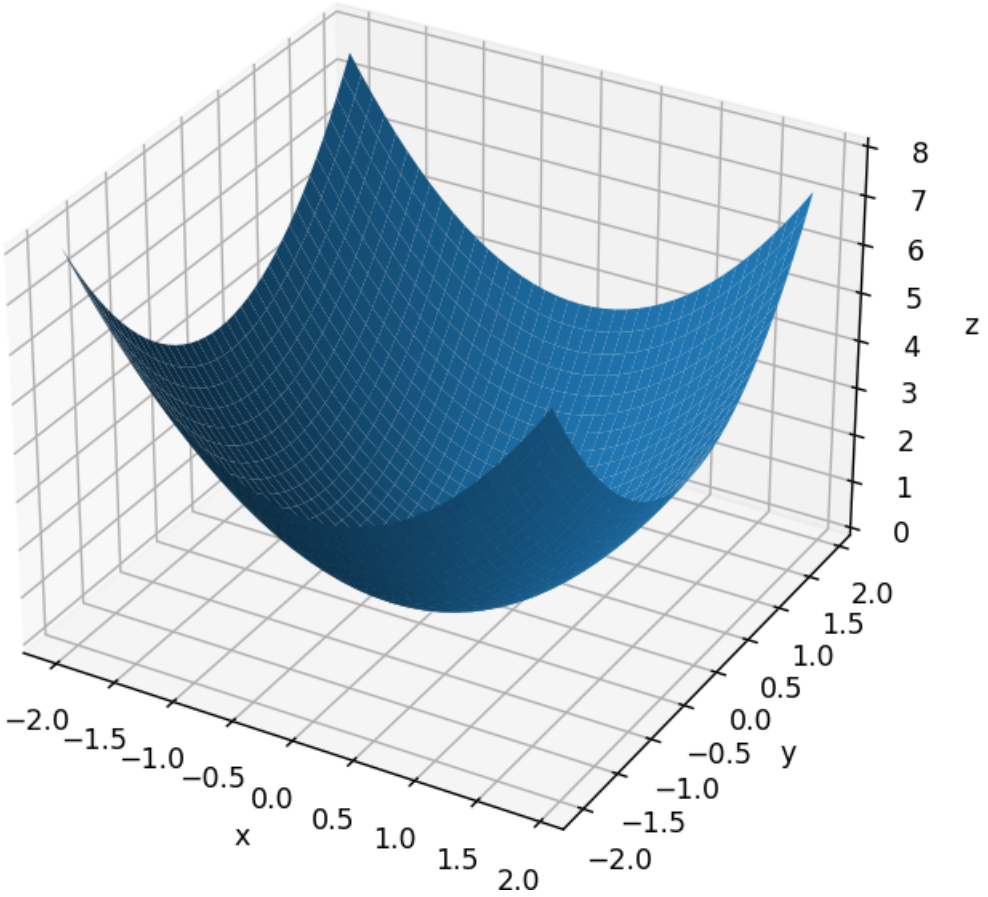
Funkce dvou proměnných má jasnou vizuální představu, např. kopce a údolí nad krajinou, viz obr. 2.9. Dva rozměry, x a y, „leží“ v rovině podstavy, funkční hodnotou je např. nadmořská výška. Funkce 3 proměnných je ještě představitelná, např. rozložení teploty v prostoru místnosti, kde je každému bodu přiřazena určitá teplota. Funkce více než 3 proměnných již nejsou vizuálně představitelné, ale matematicky s nimi lze nadále dobře pracovat.

Napišme program v Pythonu pro vykreslení grafu funkce dvou proměnných

$$f(x, y) = \frac{\sin\left(\sqrt{x^2 + y^2}\right)}{\sqrt{x^2 + y^2}}.$$

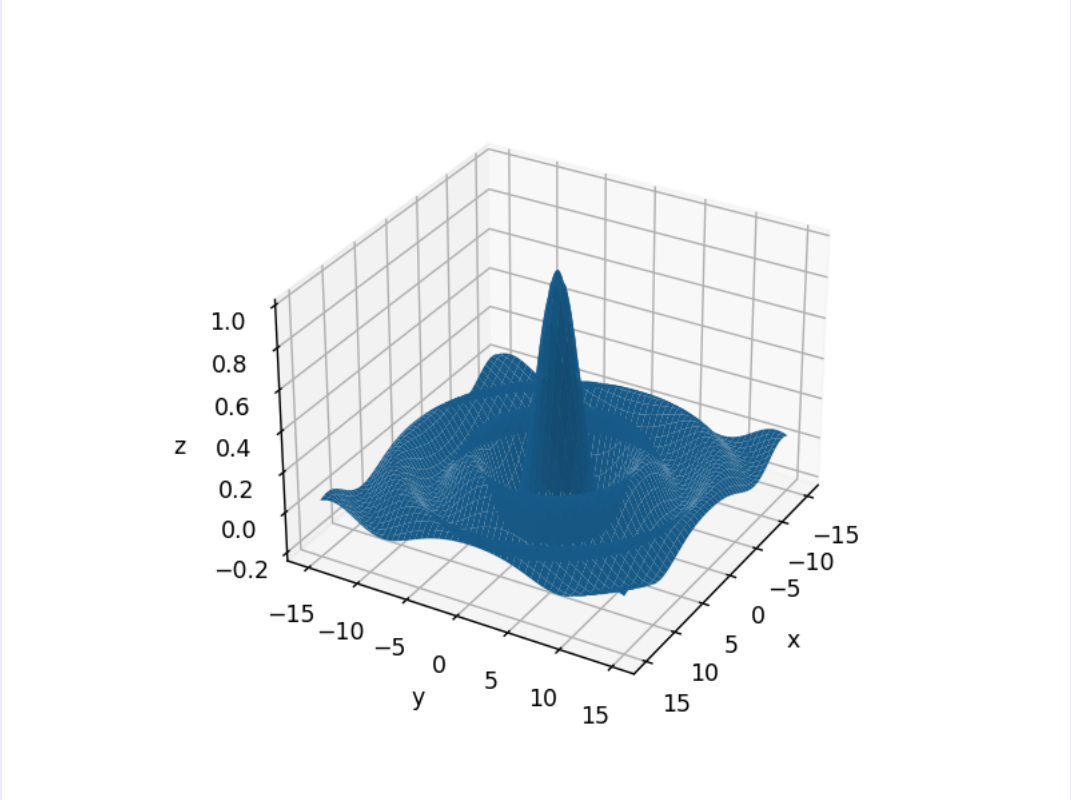
Návod lze najít v Appendixu B.2.

```
>>> import matplotlib.pyplot as plt
>>> from mpl_toolkits.mplot3d import Axes3D
>>> import numpy as np
>>> # import ipympl

>>> % matplotlib widget

>>> x = np.arange(-15, 15, 0.1) # definice proměnné x
>>> y= np.arange(-15, 15, 0.1)  # definice proměnné y

>>> def fun(x, y):
>>>     return np.sin(np.sqrt(x**2+y**2))/(np.sqrt(x**2+y**2))

>>> X, Y = np.meshgrid(x, y)
>>> Z = fun(X,Y)
```



Obrázek 2.9.: Rotační paraboloid: $z = x^2 + z^2$

```
>>> fig = plt.figure()
>>> ax = fig.add_subplot(111, projection='3d')

>>> ax.plot_surface(X, Y, Z) # vykreslení prostorového grafu

>>> ax.set_xlabel('osa x')
>>> ax.set_ylabel('osa y')
>>> ax.set_zlabel('osa z')

>>> ax.view_init(-30, 30)
>>> plt.show()
```



Vidíme, že funkce vykazuje radiální symetrii, je tedy v jakémkoli směru od středu stejná. To nás nepřekvapuje. Pokud má totiž je v nějakých bodech konstantní hodnotu $x^2 + y^2 = konst$, pak tyto body leží na kružnici a funkce f má na celé této kružnici stejnou hodnotu

$$f(x, y) = \frac{\sin\left(\sqrt{konst}\right)}{\sqrt{konst}}.$$

Je to dvourozměrná cirkulární varianta nám už známé funkce $\frac{\sin x}{x}$.

2.5.1. Parciální derivace

Parciální derivace je obdobou jednoduché derivace pro funkce více proměnných. Parciální derivaci funkce f podle x značíme $\frac{\partial f}{\partial x}$. Jak jsme popsali výše, pokud zjišťujeme derivaci funkce jedné proměnné v bodě x_0, sestrojíme nejprve tečnu k funkci v tomto bodě a pak určíme její směrnici. Významem derivace byla rychlost nárůstu funkce. Podobnou úlohu má i parciální derivace.

Představme si pro jednoduchost funkci dvou proměnných, tedy např. kopce v krajině. Problémem je, že u funkce dvou proměnných neexistuje jen jedna tečna v každém bodě, nýbrž celá tečná rovina. Kterákoli přímka v ní ležící a procházející bodem dotyku roviny s funkcí je tečnou k funkci. Kterou z tečen máme zvolit, abychom posoudili rychlost nárůst funkce? Vyberme si dvě speciální tečny. První tečna bude mířit ve směru osy x, její projekce do podstavy (t.j. roviny x-y) bude rovnoběžná s osou x. Druhá tečna bude naopak mířit ve směru osy y, její projekce je rovnoběžná s osou y. Směrnice těchto dvou tečen označujeme jako parciální derivaci podle x a podle y. Výhodou těchto speciálních tečen je skutečnost, že se při pohybu ve směru tečny mění jen jedna proměnná (x nebo y), ale druhá zůstává konstantní.

Parciální derivace funkce $f(x, y)$ v bodě (x_0, y_0) podle x (resp. y) tedy analogicky s obyčejnou derivací definujeme jako

$$\frac{\partial f}{\partial x}(x_0, y_0) := \lim_{h \to 0} \frac{f(x_0 + h, y_0) - f(x_0, y_0)}{h}$$
$$\frac{\partial f}{\partial y}(x_0, y_0) := \lim_{h \to 0} \frac{f(x_0, y_0 + h) - f(x_0, y_0)}{h}.$$

Udává, jak rychle se mění funkce $f(x, y)$, pokud se jedna z proměnných mění a všechny ostatní zůstávají neměnné.

Protože se mění pouze jedna proměnná a ostatní zůstávají konstantní, můžeme pro derivování použít nám již známá pravidla, pouze proměnnou, podle níž právě nederivujeme, považujeme za konstantu. Postup si ukážeme na příkladech.

1. Derivujme funkci $f(x, y) = x^3 + y^3$ prodle x a podle y.
   Pokud derivujeme podle x, „cokoli s y“ považujeme za konstantu, jako by tedy funkce zněla $f(x, y) = x^3 + c$. Při derivování podle y naopak funkce „vypadá“ jako $f(x, y) = c + y^3$.
   $$\frac{\partial f}{\partial x} = 3x^2$$
   $$\frac{\partial f}{\partial y} = 3y^2$$
2. Derivujme funkci $f(x, y) = \sin x . \cos y$.
   $$\frac{\partial f}{\partial x} = \cos x . \cos y$$
   $$\frac{\partial f}{\partial y} = -\sin x . \sin y$$
3. Derivujme funkci $f(x, y) = x^2 y . \sin x$.
   Platí známé pravidlo pro derivování součinu. Přitom v součinu vystupuje pouze funkce x, jako by funkce zněla $c.x^2 \sin x$.
   $$\frac{\partial f}{\partial x} = 2xy . \sin x + x^2 y . \cos x$$
   $$\frac{\partial f}{\partial y} = x^2 . \sin x$$
4. Derivujme funkci $f(x, y) = e^{x^2+y^2}$.
   Platí pravidlo pro derivování složené funkce, „vnější funkce krát vnitřní funkce“.
   $$\frac{\partial f}{\partial x} = e^{x^2+y^2} . 2x$$
   $$\frac{\partial f}{\partial y} = e^{x^2+y^2} . 2y$$
5. Derivujme funkci $f(x, y, z) = x^5 e^{xyz}$.

Stejný postup platí pro funkci 3 a více proměnných.

$$\frac{\partial f}{\partial x} = 5x^4 e^{xyz} + x^5 e^{xyz}.yz$$
$$\frac{\partial f}{\partial y} = x^5 e^{xyz}.xz$$
$$\frac{\partial f}{\partial z} = x^5 e^{xyz}.xy$$

Geometrickým významem parciální derivace funkce dvou proměnných je strmost funkce ve směru příslušné osy. Fyzikálním významem parciální derivace je rychlost nárůstu funkce se změnou jedné proměnné, přičemž ostatní proměnné zůstávají konstantní. Tento fyzikální význam je zachován i pro funkce 3 a více proměnných, ačkoli vizuální představu „tečné roviny“ již nelze uplatnit.

Demonstrujme fyzikální význam parciální derivace na příkladu.
Stavová rovnice ideálního plynu popisuje závislost tlaku ideálního plynu p na jeho látkovém množství n, objemu V a teplotě T. Zní

$$p = \frac{nRT}{V}.$$

Parciální derivace

$$\frac{\partial p}{\partial n} = \frac{RT}{V}$$
$$\frac{\partial p}{\partial T} = \frac{nR}{V}$$
$$\frac{\partial p}{\partial V} = -\frac{nRT}{V^2}$$

udávají, o kolik se změní tlak plynu, pokud se jednotlivé proměnné zvýší o jednotku a ostatní proměnné zůstanou konstantní. U derivace podle objemu máme záporné znaménko. To odpovídá naší fyzikální představě. Pokud se zvětší objem plynu, ale jeho látkové množství i teplota zůstanou konstantní, musí klesnou tlak plynu.
V termodynamice bývá zvykem v parciální derivaci vyznačit, které proměnné zůstávají konstantní. Výše uvedené derivace by tak byly spíše zapsány jako $\left(\frac{\partial p}{\partial n}\right)_{V,T}$, $\left(\frac{\partial p}{\partial T}\right)_{n,V}$ a $\left(\frac{\partial p}{\partial V}\right)_{n,T}$.

2.5.2. Vyšší a smíšené parciální derivace

Stejně jako u funkcí jedné proměnné je i výsledkem parciální derivace funkce více proměnných opět funkce více proměnných. Tuto novou funkci můžeme přirozeně opět podrobit derivování. Nyní ovšem máme více možností. Druhá derivace může být podle stejné nebo jiné proměnné. Pokud znovu derivujeme podle stejné proměnné, dospějeme k druhým, třetím a dalším derivacím.

$$\frac{\partial^2 f}{\partial x^2}, \frac{\partial^2 f}{\partial y^2}, \frac{\partial^3 f}{\partial x^3} \dots$$

Pokud derivujeme podle jiné proměnné než při předchozí derivaci, získáme derivace smíšené.

$$\frac{\partial^2 f}{\partial x \partial y}, \frac{\partial^3 f}{\partial x^2 \partial y}, \frac{\partial^3 f}{\partial x \partial y \partial z} \dots$$

Spočtěme všechny první a druhé parciální derivace funkce $f(x, y) = e^{x^2+y^2}$.

$$\begin{aligned}
\frac{\partial f}{\partial x} &= e^{x^2+y^2}.2x \\
\frac{\partial f}{\partial y} &= e^{x^2+y^2}.2y \\
\frac{\partial^2 f}{\partial x^2} &= e^{x^2+y^2}.4x^2 + 2e^{x^2+y^2} \\
\frac{\partial^2 f}{\partial y^2} &= e^{x^2+y^2}.4y^2 + 2e^{x^2+y^2} \\
\frac{\partial^2 f}{\partial x \partial y} &= \frac{\partial e^{x^2+y^2}.2x}{\partial y} = e^{x^2+y^2}.4xy
\end{aligned}$$

V případě smíšených druhých derivací máme dvě možnosti: derivovat nejprve podle x a poté podle y, nebo opačně. Dostaneme tak $\frac{\partial^2}{\partial x \partial y}$ nebo $\frac{\partial^2}{\partial y \partial x}$. V matematické analýze se dokazuje Schwarzova věta, která tvrdí, že nezáleží na pořadí derivování ve smíšených parciálních derivacích, tedy že

$$\frac{\partial^2 f}{\partial x \partial y} = \frac{\partial^2 f}{\partial y \partial x}.$$

Ověřme Schwarzovu větu na funkci $f(x, y) = e^{x^2+y^2}.2x$.

$$\begin{aligned}
\frac{\partial^2 f}{\partial x \partial y} &= \frac{\partial \left(e^{x^2+y^2}.2x \right)}{\partial y} = e^{x^2+y^2}.4xy \\
\frac{\partial^2 f}{\partial y \partial x} &= \frac{\partial \left(e^{x^2+y^2}.2y \right)}{\partial x} = e^{x^2+y^2}.4xy
\end{aligned}$$

2.5.3. Vyšetřování průběhu funkce více proměnných

Průběh funkce více proměnných může být podstatně složitější než průběh funkce jedné proměnné. V určitém bodě se například může nacházet lokální maximum nebo lokální minimum. Nebo se v určitém směru může nacházet minimum a v kolmém směru maximum, pak se jedná o tzv. sedlový bod. Obecně ale může být situace ještě komplikovanější.

Podobně jako v případě funkcí jedné proměnné je nutnou podmínkou pro extrém funkce $f(x, y)$ v bodě (x_0, y_0) nulovost obou parciálních derivací, tedy

$$\frac{\partial f(x_0, y_0)}{\partial x} = 0; \ \frac{\partial f(x_0, y_0)}{\partial y} = 0.$$

Není to však podmínka postačující. Může se stát, že jsou obě parciální derivace nulové, ale ve

směru „mezi osami“ funkce roste a nejedná se tedy o lokální extrém. V reálných problémech však podmínka většinou „postačující“ je.

Najděme lokální extrémy funkce $f(x, y) = x^2 + y^2$ (obr. 2.9).

Pro extrém musí platit

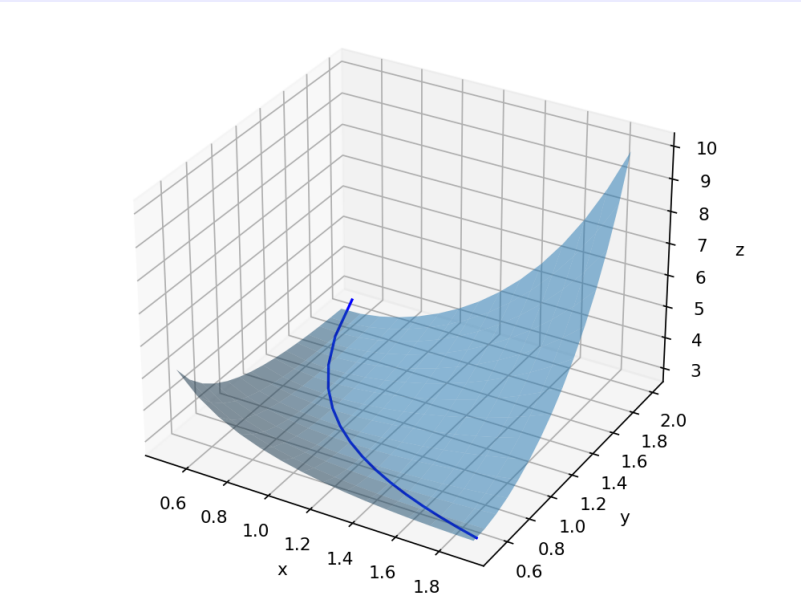
$$\frac{\partial(x^2 + y^2)}{\partial x} = 2x = 0$$
$$\frac{\partial(x^2 + y^2)}{\partial y} = 2y = 0$$

Jediným řešením je zjevně bod (0,0), kde funkce dosahuje minima.

Představme si čtvercovou krajinu, která je vymezena hodnotami x i y 0,5 a 2, tedy $x \in [0.5, 2]$ i $y \in [0.5, 2]$. Profil krajiny, tedy nadmořská výška h jednotlivých bodů, je dán funkcí

$$h = \frac{e^{xy}}{xy}.$$

V údolí krajiny, nejníže položenými body, teče řeka. Zjistěme, kudy řeka teče, t.j. určeme funkci $y = f(x)$, která body řeky popisuje. Určeme též, v jaké nadmořské výšce řeka teče.



Údolí, nejnižší body, musí splňovat podmínky $\frac{\partial h}{\partial x} = 0$ a $\frac{\partial h}{\partial x} = 0$. Odtud získáme 2 rovnice, určující podmínky pro vztah x a y.

$$\frac{\partial h}{\partial x} = \frac{\partial}{\partial x}\frac{e^{xy}}{xy} = \frac{ye^{xy}xy - e^{xy}y}{(xy)^2} = 0 \;\rightarrow xy = 1$$
$$\frac{\partial h}{\partial y} = \frac{\partial}{\partial y}\frac{e^{xy}}{xy} = \frac{xe^{xy}xy - e^{xy}x}{(xy)^2} = 0 \;\rightarrow xy = 1$$

Řeka je tedy popsána rovnicí $y = 1/x$. Odpovídající nadmořská výška je $h = e$. Poznamenejme pro úplnost, že obecně podmínky $\frac{\partial h}{\partial x} = 0$ a $\frac{\partial h}{\partial x} = 0$ pro řeku spíše neplatí, protože řeka teče z kopce dolů. Naše řeka byla speciální, vodorovná, jak už to v umělých příkladech bývá.

2.5.4. Aplikace funkcí více proměnných a parciálních derivací v biologii

Funkce více proměnných a s nimi i parciální derivace jsou v přírodních vědách všudypřítomné. Často se vyskytují v podobě rovnic, které se označují jako parciální diferenciální rovnice. Některé z nich mají centrální postavení ve fyzice, označují se dokonce jako rovnice matematické fyziky. Tzv. vlnová rovnice

$$\frac{1}{v^2}\frac{\partial^2 z(x,t)}{\partial t^2} = \frac{\partial^2 z(x,t)}{\partial x^2}$$

může například popsat šíření elektrického vzruchu po axonu. v je rychlost šíření vzruchu a z je membránový potenciál, který je funkcí polohy na axonu x a času t. My se těmto rovnicím nebudeme podrobněji věnovat. Ukážeme si však biologickou aplikaci funkcí více proměnných a parciálních derivací na případu teorie optimálního hematokritu, který je speciálním případem teorie evoluční optimality v biologii.

Při jistém zjednodušení evoluční teorie říká, že se organizmy během evoluce vyvinuly tak, že jsou v jistém smyslu nejlepší, optimální. V průběhu evoluce se tedy optimalizovala jistá vlastnost, kterou můžeme matematicky popsat nějakou veličinou S. S je ale funkcí řady jiných proměnných, x_1, x_2..., tedy $S = S(x_1, x_2, \ldots)$. Veličinou S může být např. dodávka kyslíku do tkání, která zřejmě závisí na arteriálním tlaku, množství hemoglobinu v krvi, funkci plic apod. Je přirozené očekávat, že tak důležitá veličina, jako je dodávka kyslíku, dosáhla během evoluce maxima ve vztahu k těmto proměnným. Hledáme tedy hodnotu nějaké proměnné x_1, např. koncentrace hemoglobinu v krvi, při níž S nabývá maxima. Pro tuto hodnotu $x_{1,max}$ musí platit

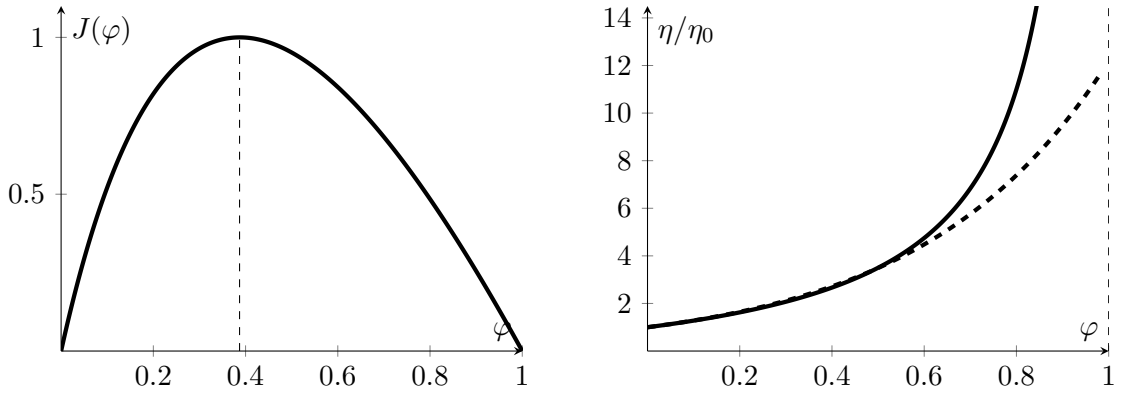
$$\frac{\partial S}{\partial x_1}(x_{1,max}) = 0.$$

Hematokrit je procento červených krvinek, erytrocytů, v krvi. U většiny savců i řady jiných zvířat se hematokrit pohybuje okolo 40 %. Vcelku nápadná shoda. Proč právě 40 %? Zkusme zjistit, zda 40 % náhodou není optimální hodnota maximalizující dodávku kyslíku. Hemoglobin váže kyslík, krev s vyšším hematokritem nese více kyslíku. Zároveň však vyšší hematokrit zvyšuje viskozitu krve, čímž zpomaluje její tok. Je-li hematokrit příliš nízký, krev teče rychle, ale nese málo kyslíku. Dodávka kyslíku do tkání je nízká. Je-li naopak příliš vysoký, krev sice nese hodně kyslíku, ale teče velmi pomalu. Optimální hodnota musí ležet někde uprostřed, jak je patrno z obr. 2.10 (a).

Viskozita obecně roste s hematokritem, viz obr. 2.10 (b). Pro výpočet optimálního hematokritu potřebujeme znát přesnou závislost viskozity krve na hematokritu. Byla odvozena řada teoretických vztahů. My použijeme jednoduché vztahy Arrhenia $\eta = \eta_0 e^{2,5\varphi}$ a Saita $\eta = \eta_0(1+2,5\frac{\varphi}{1-\varphi})$, kde η značí viskozitu krve, η_0 viskozitu krevní plazmy a φ hematokrit. $\eta(\varphi)$ zdůrazňuje, že je viskozita závislá na hematokritu.

Pokračujme dále. Pro průtok krve cévou Q platí známá Hagen-Poiseuillova rovnice

$$Q = \frac{\pi r^4 \Delta p}{8 l \eta(\varphi)}.$$



(a) Závislost dodávky kyslíku (b) Závislost viskozity podle Arrhenia a Saito

Obrázek 2.10.: Závislost dodávky kyslíku a viskozity krve na hematokritu

kde Δp je tlakový gradient mezi začátkem a koncem cévy, l a r jsou délka a poloměr cévy. Pokud budeme rozměr cévy považovat za konstantní a společně s ostatními konstantami je zahrneme do nové konstanty K, dostaneme tvar

$$Q = K\frac{\Delta p}{\eta(\varphi)}.$$

Množství kyslíku v litru krve C_{ox} je úměrné hematokritu φ, krev považujme za plně nasycenou kyslíkem. Platí tedy

$$C_{ox} = \kappa\varphi.$$

κ je konstanta úměrnosti. Pro dodávka kyslíku cévou do tkání J_{ox} pak samozřejmě platí

$$J_{ox} = C_{ox}Q.$$

Po dosazení obdržíme vztah

$$J_{ox} = K'\Delta p\frac{\varphi}{\eta(\varphi)},$$

kde jsme všechny, v tuto chvíli nepodstatné konstanty zahrnuli do nové konstanty $K = \kappa k$. Dodávka kyslíku je tedy funkcí dvou proměnných, tlakového gradientu a hematokritu.

Obr. 2.11 ukazuje závislost relativní dodávky kyslíku, t.j. procenta z maximální dodávky $J_{ox}/J_{ox,max}$, na hematokritu pro Arrheniovu a Saitovu závislost viskozity na hematokritu. Saitoův vztah poskytuje průběh funkce, jak jsme předpokládali. Tedy nulové hodnoty na okrajích a maximum uprostřed. Arrheniovův vztah je zjevně chybný pro vysoký hematokrit, protože počítá nenulovou dodávku kyslíku, ačkoli krev již téměř neteče (neboť už prakticky neobsahuje plazmu).

Hledejme nyní hematokrit, který za konstantního tlakového gradientu maximalizuje dodávku kyslíku. Zde vstupuje do hry parciální derivace. Musí tedy platit

$$0 = \frac{\partial J_{ox}}{\partial\varphi} = K'\Delta p\frac{\eta(\varphi) - \varphi\eta'(\varphi)}{\eta^2(\varphi)}.$$

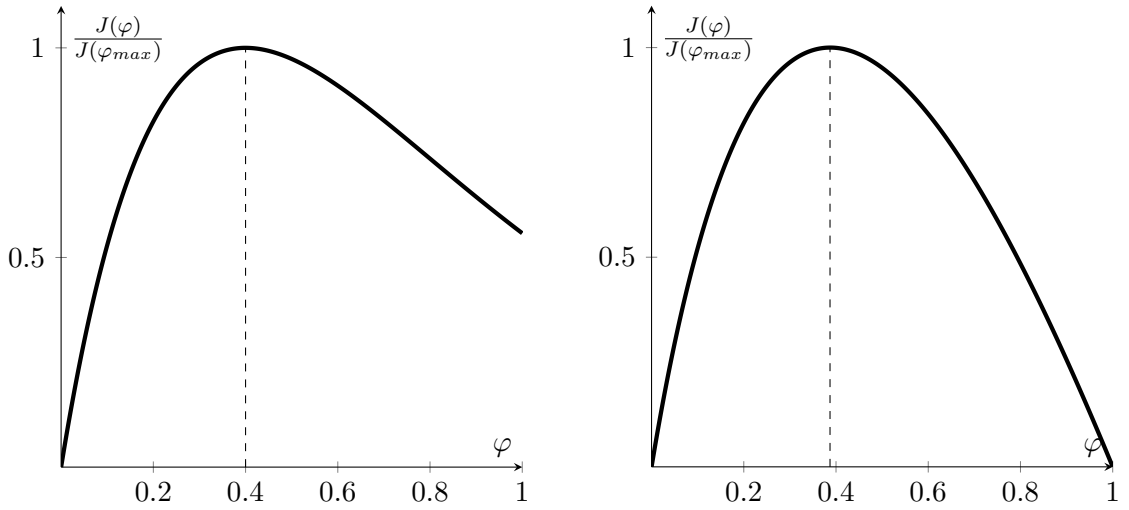
Odtud získáme rovnici

$$\eta(\varphi) - \varphi\eta'(\varphi) = 0,$$

z níž plyne podmínka optimálního hematokritu

$$\varphi_{max}^{\Delta p} = \frac{\eta(\varphi_{max}^{\Delta p})}{\eta'(\varphi_{max}^{\Delta p})}.$$

Horní index Δp značí, že jsme derivovali za konstantního tlaku, našli jsme tedy optimální hematokrit za situace, kdy by byl tlakový gradient Δp udržován konstantní. Abychom rovnici mohli dořešit, musíme znát konkrétní závislost viskozity na hematokritu. Určeme optimální hodnoty pro Saitoův a Arrheniův vztah:



(a) pro Arrheniův vztah $\eta = \eta_0.e^{2,5\varphi}$ (b) pro Saitoův vztah $\eta = \eta_0\left(1 + 2,5\varphi/(1-\varphi)\right)$

Obrázek 2.11.: Závislost dodávky kyslíku na hematokritu při konstantním perfúzním tlaku

1. S použitím Arrheniova vztahu:

$$\varphi_{max} = \frac{\eta_0.e^{2,5\varphi}}{\eta_0.2,5.e^{2,5\varphi}} = 0,4$$

2. S použitím Saitoova vztahu:

$$\varphi = \frac{\eta_0\left(1 + 2,5\frac{\varphi}{1-\varphi}\right)}{\eta_0\left(2,5\frac{1}{(1-\varphi)^2}\right)} = \frac{1 + 2,5\frac{\varphi}{1-\varphi}}{2,5\frac{1}{(1-\varphi)^2}} \rightarrow 2,5\varphi = (1-\varphi)^2 + 2,5\varphi(1-\varphi)$$

Odtud získáme rovnici

$$1.5\varphi^2 + 2\varphi - 1 = 0 \rightarrow \varphi_{max} = \frac{-2 + \sqrt{4+6}}{3} = 0,387.$$

Potvrdili jsme tak naši domněnku. Optimální hematokrit se skutečně pohybuje okolo 40 %, jak příroda již během evoluce dávno zjistila.

2.6. Integrální počet

Problematika derivací se obvykle označuje jako diferenciální počet, problematika integrálů jako integrální počet. Oba dohromady pak jako infinitezimální počet, protože oba mají co do činění

s nekonečně malými čísly. Derivace je podíl dvou nekonečně malých čísel. (Určitý) integrál je součet nekonečně mnoha nekonečně malých čísel.

2.6.1. Idea integrálního počtu

Pokud zderivujeme funkci x^2, získáme funkci $2x$. Položme si nyní opačnou úlohu. Kterou funkci jsme museli zderivovat, abychom získali funkci $2x$? Snadné. x^2. Ovšem též $x^2 + c$, kde c je libovolná konstanta. Integrování je zpětná cesta, určitý opak derivace. Integrál, který je opakem derivace, se přesněji označuje jako neurčitý integrál a značí se symbolem $\int$. Též se označuje jako primitivní funkce. Dále existuje integrál určitý, který „počítá“ plochu pod křivkou a úzce souvisí s neurčitým integrálem. Dále si přesněji definujeme oba integrály, popíšeme si jejich vlastnosti a způsoby výpočtu. Budeme se zabývat pouze funkcemi jedné proměnné.

2.6.2. Neurčitý integrál

Mějme funkci $F(x)$. Její derivaci označme $f(x)$, tedy $f(x) := F'(x)$. $F(x)$ označujeme jako **primitivní funkci** nebo synonymně jako **neurčitý integrál** k funkci $f(x)$ a značíme

$$F(x) = \int f(x)\,dx.$$

dx značí proměnnou, podle níž integrujeme, podobně jako dx značilo proměnnou, podle níž jsme derivovali. Neurčitý integrál též označujeme jako integrál Newton-Leibnitzův.

Pokud jakoukoli $F(x)$ funkci posuneme o konstantu, $F(x)+c$, její derivace se nezmění - $(F(x)+c)' = (F(x))' = f(x)$. Proto bychom pro integrál správněji měli psát

$$F(x) = \int f(x)\,dx + c,$$

neboť řešení integrace není jedna funkce, nýbrž celá nekonečná množina funkcí, které se navzájem liší o libovolnou, tzv. integrační konstantu.

2.6.3. Metody integrace

Integrování je podstatně složitější činnost než derivování. Pomocí pravidel pro derivování lze víceméně snadno zderivovat jakoukoli (běžnou) funkci. Integrovat jakoukoli funkci obecně nejen není snadné, ale ani možné. Např. integrál Gaussovy funkce

$$\int e^{x^2}\,dx$$

známé ze statistiky sice existuje, ale není vyjádřitelný v uzavřené formě pomocí elementárních funkcí jako e^x, $\ln x$ nebo $\sin x$. Ukážeme si dvě metody integrace - metodu substituční a metodu per partes, ale ani s nimi není řada funkcí snadno integrovatelná.

Přímá integrace

U základních elementárních funkcí, kde „původní derivaci přímo vidíme“, je integrace snadná.

$$\int 1\,dx = x + c$$
$$\int x\,dx = \frac{1}{2}x^2 + c$$
$$\int x^n\,dx = \frac{1}{n+1}x^{n+1} + c$$
$$\int \cos x\,dx = \sin x + c$$
$$\int \sin x\,dx = -\cos x + c$$
$$\int e^x\,dx = e^x + c$$
$$\int \frac{1}{x}\,dx = \ln x + c$$

Podobně jako derivace je i intergrál lineární zobrazení, platí tedy

$$\int f(x) \pm g(x)\,dx = \int f(x)\,dx \pm \int g(x)\,dx + c$$
$$\int k.f(x)\,dx = k.\int f(x)\,dx + c.$$

Vyřešme několik jednoduchých příkladů. Vynecháme integrační konstantu.

$$\int 2e^{3x}\,dx = 2\int e^{3x}\,dx = \frac{2}{3}e^{3x}$$
$$\int 2x^2 + 3x^3\,dx = 2\int x^2\,dx + 3\int x^3\,dx = \frac{2}{3}x^3 + \frac{3}{4}x^4$$
$$\int \sin 2x\,dx = -\frac{1}{2}\cos x$$
$$\int \frac{1}{x^2}\,dx = \int x^{-2}\,dx = -\frac{1}{x}$$

Cokoli dalšího už je složitější. Podobná pravidla jako pro derivaci součinu, podílu, inverzních funkcí apod. pro integrování neexistují. Někdy pomohou zmíněné metody substituční a per partes.

Substituční metoda

Pomocí substituční metody nahradíme, substituujeme, komplikovanější část integrované funkce jinou funkcí, čímž získáme jednodušší tvar, který už umíme přímo integrovat.

Jak už jsme říkali, s diferenciály d můžeme často operovat jako v jinými výrazy, násobit, dělit apod. Ukažme si metodu na příkladech.

Vyřešme integrál $\int x \sin x^2 \, dx$.
„Uhádnout“ správné řešení asi neumíme. Použijme však následující substituci: nahradíme funkci x^2 funkcí t, tedy vlastně definujeme novou funkci $t(x) := x^2$. Pro ni platí

$$\frac{dt}{dx} = 2x \to x \, dx = \frac{1}{2} \, dt.$$

„Svévolné manipulace“ s diferenciály vypadají možná podezřele, ale matematika tento postup pochopitelně exaktně dokázala. Nyní vše dosadíme do původního výrazu a upravíme. Dostaneme

$$\int x \sin x^2 \, dx = \int \frac{1}{2} \sin t \, dt = \frac{1}{2} \int \sin t \, dt = -\frac{1}{2} \cos t = -\frac{1}{2} \cos x^2.$$

Tímto „trikem“ jsme integrál převedli na známý tvar a přímo vyřešili. Derivací ověříme, že

$$\frac{d}{dx} \left(-\frac{1}{2} \cos x^2 \right) = \frac{1}{2} \frac{d \cos x^2}{dx} = x \sin x^2.$$

Zkusme nyní vyřešit integrál $\int \frac{\ln x}{x} \, dx$.

Substituci musíme volit chytře, aby se nám intergrál zjednodušil a převedl na tvar, který umíme řešit. Zde nám pomůže substituce $t := \ln x$, odkud $dt = \frac{dx}{x}$. Proto

$$\int \frac{\ln x}{x} \, dx = \int t \, dt = \frac{1}{2} t^2 = \frac{1}{2} \ln^2 x.$$

Derivací opět ověříme, že

$$\frac{d}{dx} \frac{1}{2} \ln^2 = \frac{1}{2} \frac{d \ln^2}{dx} = \frac{\ln x}{x}.$$

Metoda per partes

Druhou metodou, kterou si představíme, je metoda integrování „po částech“, per partes. Je aplikací pravidla pro derivování součinu

$$(f.g)' = f'.g + f.g' \to f'.g = (f.g)' - f.g'.$$

Nyní obě strany zintegrujeme a využijeme skutečnosti, že „integrál z derivace“ se rovná původní funkci.

$$\int f'(x).g(x) \, dx = \int (f(x).g(x))' \, dx - \int f(x).g'(x) \, dx$$

Tedy

$$\int f'(x).g(x) \, dx = f(x).g(x) - \int f(x).g'(x) \, dx.$$

Zdánlivě jsme mnoho nezískali. Funkce g se však po derivaci může výrazně zjednodušit. Ukažme si postup opět na příkladech.

Zintegrujme funkci $x \sin x$.

Lépe si funkci napišme jako $\sin x.x$. Představme si, že $f'(x) = \sin x$ a $g(x) = x$. Pak $f(x) = -\cos x$ a $g'(x) = 1$. Dosadíme do výše uvedeného vzorce.

$$\int \sin x.x\, dx = -x\cos x - \int -\cos x\, dx = -x\cos x + \sin x$$

Derivací ověříme, že

$$\frac{d}{dx}(-x\cos x + \sin x) = -\cos x + x\sin x + \cos x = x\sin x.$$

Vypočítejme nyní $\int \ln x\, dx$.

Zde máme pouze jednu funkci. Zkusme si ale integrál představit jako $\int 1.\ln x\, dx$ a definujme $f'(x) = 1$ a $g(x) = \ln x$. Pak $f(x) = x$ a $g'(x) = \frac{1}{x}$. Dosadíme do výše uvedeného vzorce.

$$\int \ln x\, dx = x\ln x - \int \frac{x}{x}\, dx = x\ln x - x = x(\ln x - 1)$$

Derivací opět ověříme, že

$$\frac{d}{dx}(x\ln x - x) = \ln x + \frac{x}{x} - 1 = \ln x.$$

Sami vidíte, že nevíme předem, kterou substituci nebo rozložení integrálu máme použít. Vždy je potřeba nový nápad. Proto je integrování obtížnější než derivování. V reálných problémech, jako je např. matematické modelování v biologii, ale určování primitivní funkce většinou nepotřebujeme, případně pomohou i některé online řešiče.

2.6.4. Určitý integrál

Určitý integrál, též označovaný jako Riemannův integrál, historicky vznikl nezávisle na integrálu neurčitém a nemá s ním na první pohled žádnou souvislost. Úkolem bylo určit velikost plochy pod křivkou. Princip spočíval v tom, že se celá plocha pod křivkou S ohraničená hodnotami $x = a$ a $x = b$ rozdělí na úzké obdélníky o šířce Δx a plochách S_i a sečte se plocha všech obdélníků (2.12), která přibližně odpovídá ploše pod křivkou.

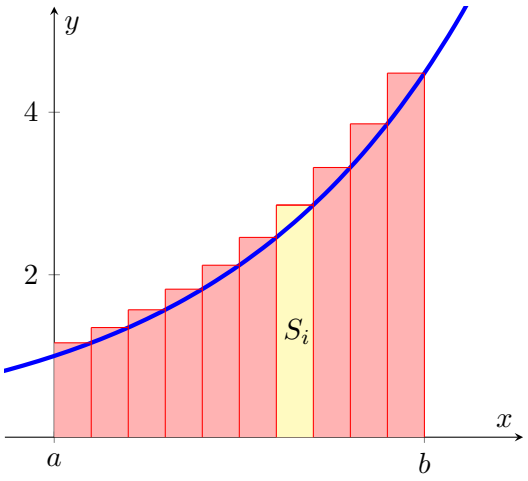
$$S \approx \sum_{i=1}^{n} S_i = \sum_{i=1}^{n} f(x_i)\Delta x$$

Je přitom jasné, že jak roste počet obdélníků a klesá jejich šířka, součet plochy obdélníků se stále více blíží ploše pod křivkou, až v limitě obě plochy splynou.

Určitý integrál $\int_a^b f(x)\, dx$ je vlastně „sčítání nekonečně velkého počtu nekonečně malých čísel“. Vyjadřuje plochu pod křivkou funkce $f(x)$ mezi hodnotami $x = a$ a $x = b$.

$$\int_a^b f(x)\, dx := \lim_{n\to\infty} \sum_{i=1}^{n} S_i = \lim_{n\to\infty} \sum_{i=1}^{n} f(x_i)\Delta x = S$$

Pro jasnost dodejme, že výsledkem určitého integrálu je číslo, zatímco výsledkem neurčitého integrálu je funkce. Výpočet plochy pod křivkou jako limity součtu je ovšem velmi obtížný. Elegantním řešením je výpočet určitého integrálu pomocí neurčitého integrálu.



Obrázek 2.12.: Idea určitého integrálu

Vztah neurčitého a určitého integrálu

Protože souvislost mezi určitým a neurčitým integrálem je zásadní, jsme nuceni ji odvodit, nikoli pouze prozradit. Chceme tedy určit plochu pod křivkou na obr. 2.13, přičemž plochu budeme měřit od nějakého konkrétního, ale libovolného bodu vlevo, např. od $x = -2$. V takovém případě je změřená plocha funkcí x. Např. v bodě x_0 je to $S = S(x_0)$. O malý úsek h dále je obsah roven $S(x_0 + h)$. Rozdíl mezi oběma obsahy je na obr. 2.13 vyznačen žlutě. Je zjevné, že je tento rozdíl téměř roven obsahu červeně vyšrafovaného obdélníku o rozměrech h a $f(x_0)$. Platí tedy

$$f(x_0).h \approx S(x_0 + h) - S(x_0).$$

Když se h zmenšuje, „žlutá a červeně vyšrafovaná plocha“ si jsou stále bližší, až v limitě pro $h \to 0$ splynou. Tedy

$$\lim_{h\to 0} f(x_0).h = \lim_{h\to 0} S(x_0 + h) - S(x_0).$$

Po úpravě dostaneme

$$f(x_0) = \lim_{h\to 0} \frac{S(x_0 + h) - S(x_0)}{h}.$$

Výraz na pravé straně rovnice je ale právě definice derivace S podle x. Tedy nutně

$$f(x) = \frac{dS(x)}{dx}.$$

Po integraci platí

$$S(x) = \int f(x)\, dx + .C$$

Konstanta C je určena bodem, kde jsme „začali měřit“ plochu. Pro plochu mezi body $x = a$ a $x = b$ platí

$$S(b) - S(a) = \left[\int f(x)dx\right]_{(b)} + C - \left[\int f(x)dx\right]_{(a)} - C =: \int_a^b f(x)dx.$$

$\left[\int f(x)dx\right]_{(b)}$ a $\left[\int f(x)dx\right]_{(a)}$ značí hodnotu neurčitého integrálu, primitivní funkce, v bodě $x = b$ a $x = a$. Vidíme, že hodnota konstanty C je nepodstatná, neb se odečte. Hodnoty $x = b$ a $x = a$ označujeme jako horní a dolní integrační mez.

Tím dostáváme způsob, jak vypočítat určitý integrál pomocí neurčitého. Stačí určit libovolnou primitivní funkci (neurčitý integrál), určit její hodnoty v horní i dolní integrační mezi a obě hodnoty odečíst.

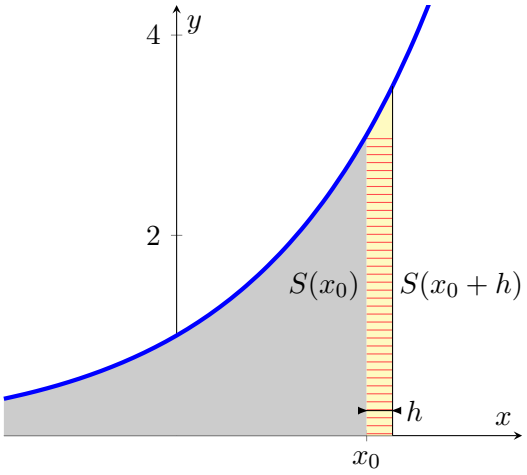
Jestliže primitivní funkci k $f(x)$ označíme $F(x)$, tedy

$$F(x) = \int f(x)\,dx.$$

pak pro určitý integrál platí

$$\int_a^b f(x)\,dx = [F(x)]_a^b = F(b) - F(a).$$

Bývá zvykem rozdíl $F(b) - F(a)$ označovat $[F(x)]_a^b$.



Obrázek 2.13.: Odvození vztahu určitého a neurčitého integrálu

Vypočtěme následující určité integrály:

1. $\int_0^1 e^3\,dx$

$$\int_0^1 e^x\,dx = [e^x]_0^1 = e^1 - e^0 = e - 1 \approx 1,72$$

2. $\int_0^{2\pi} \sin x\,dx$

$$\int_0^{2\pi} \sin x\,dx = [-\cos x]_0^{2\pi} = -\cos 2\pi - (-\cos 0) = -1 - (-1) = 0$$

3. $\int_1^2 x^3 + 4x^5\,dx$

$$\int_1^2 x^3 + 4x^5\,dx = \left[\frac{1}{4}x^4 + \frac{4}{6}x^6\right]_1^2 = \frac{16}{4} + \frac{128}{3} - \frac{1}{4} - \frac{2}{3} = \frac{15}{4} + \frac{126}{3} = 45,75$$

Aplikace určitého integrálu

Aplikací určitého integrálu je řada. Vtip je v tom, že nejprve použijeme Riemannovu definici integrálu $\sum f(x)\Delta x$, abychom určili, co vlastně chceme integrovat. Musíme tedy najít elementy $f(x)\Delta x$, které chceme sčítat. Poté nahradíme Δx pomocí $\mathrm{d}x$ a $\sum$ pomocí $\int$ a zintegrujeme, jak jsem výše popsali. Ostatně symbol $\int$ v minulosti vznikl právě úpravou písmene S, ze slov sum, Summe, součet apod.

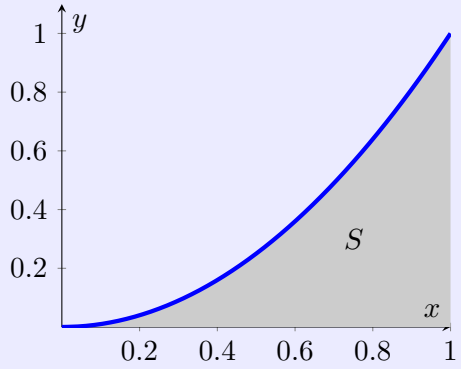
$$\sum f(x)\Delta x \to \int f(x)\,dx$$

Vše objasníme na příkladech.

Výpočet plochy pod grafem funkce

Chceme zjistit plochu pod grafem funkce $y = x^2$ pro $x \in [0, 1]$. Sčítaný element je $x^2.\Delta x$. Integrační proměnnou je x.

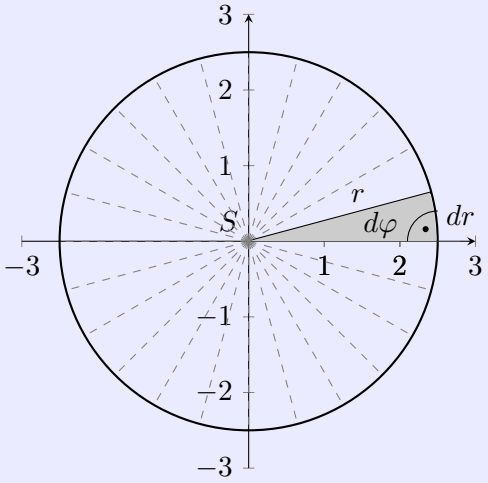
$$S = \lim_{n\to\infty} \sum_0^n x_i^2 \Delta x = \int_0^1 x^2 dx = \left[\frac{1}{3}x^3\right]_0^1 = \frac{1}{3}$$



Výpočet obsahu kruhu

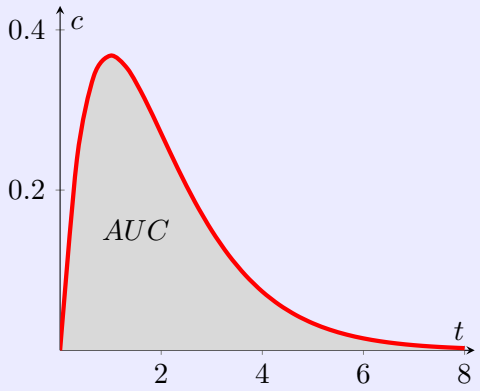
Chceme odvodit známý vztah pro výpočet obsahu kruhu $S = \pi r^2$. Postupujme podle obrázku. Máme kruh o poloměru r, který si celý rozdělíme na infinitezimálně úzké rovnoramenné trojúhelníky, z nichž každý svírá úhel $d\varphi$, má ramena délky r a základnu dr. Uvědomíme si, že úzký rovnoramenný trojúhelník je v podstatě pravoúhlý. Pro jeho obsah tedy platí $dS = r.dr/2$. Z definice úhlu víme, že $dr = r.d\varphi$. Sčítaný element bude $dS = r^2.d\varphi/2$. Pozor, integrační proměnnou nyní není r, nýbrž φ, r vystupuje jako konstanta, kterou vytkneme před integrál. Sčítat budeme všechny trojúhelníky přes celý kruh, tedy pro $\varphi \in [0, 2\pi]$.

$$S = \int_0^S dS = \int_0^{2\pi} \frac{1}{2}r^2 d\varphi = \frac{1}{2}r^2 \int_0^{2\pi} d\varphi = \frac{1}{2}r^2.2\pi = \pi r^2$$



Biologická dostupnost léku a AUC (area under curve)

Pokud nemocný spolkne lék, víme, kolik léku jsme podali (D). Nevíme však, jaká část léku se vstřebala do krevní plazmy a jaká se vyloučila stolicí. Označme množství léku, které se vstřebalo do krevní plazmy D_r. Můžeme tuto vstřebanou část, označovanou jako biologická dostupnost léku $F = D_r/D$, nějak zjistit? Je jasné, že veškeré množství léku, které se z plazmy později eliminovalo, musí být právě to množství, které se předtím vstřebalo. Eliminaci samu však bohužel též nemůžeme měřit. Můžeme však měřit koncentraci léku opakovaně v čase a eliminaci léku z této křivky dopočítat.



Potřebujeme k tomu znát některé farmakokinetické parametry léku. Jeden z nich, clearance, udává intenzitu odstraňování léku z těla. Clearance Cl je definovaná jako objem plazmy, který je od léku úplně očištěn za jednotku času, má tedy rozměr např. $ml.s^{-1}$. Za čas dt se tedy očistí $Cl.dt$ plazmy. Pokud v tu chvíli činila plazmatická koncentrace léku c, odstranilo se $c.Cl.dt$ léku. Pokud clearanci považujeme za konstantní, odstranilo se od okamžiku podání léku za dostatečně („nekonečně“) dlouhou dobu celkem

$$\int_0^\infty c.Cl.dt = Cl. \int_0^\infty cdt = D_r$$

léku. Integrál $\int_0^\infty cdt$ je přitom plocha pod koncentrační křivkou a označuje se jako area

under curve (AUC). Pro biologickou dostupnost tedy platí

$$F = \frac{D_r}{D} = \frac{Cl.AUC}{D}.$$

Co kdybychom neznali clearance Cl? Mohli bychom třeba stejné množství léku D aplikovat přímo do žíly, pak by platilo $D_r = D$ a $F = 1$. Znovu bychom měřili koncentrační křivku a zjistili $AUC_{nitrožilní}$. Pak se rovnice změní na

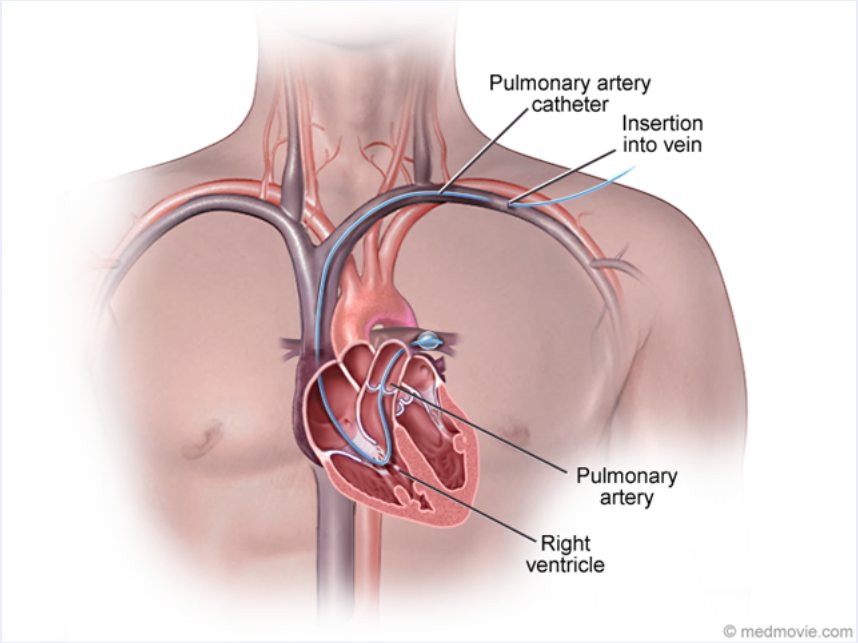
$$1 = \frac{Cl.AUC_{nitrožilní}}{D}.$$

Clearance zůstává v obou variantách podání léku stejná, protože popisuje eliminaci léku např. v játrech či ledvinách, která nezávisí na způsobu podání. Pokud vše dosadíme, můžeme zjistit F jako

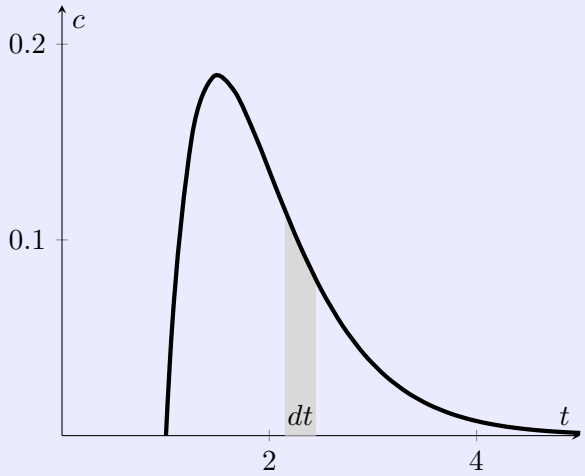
$$F = \frac{D_r}{D} = \frac{AUC_{perorální}}{AUC_{nitrožilní}}.$$

Měření srdečního výdeje pomocí Swan-Ganzova katetru

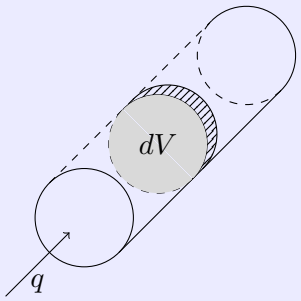
V roce 1970 vyvinuli Swan a Ganz speciální katetr, který umožňuje měřit srdeční výdej. Katetr se cestou centrální žíly (např. v. jugularis interna) zavede přes pravou srdeční síň a komoru do plicní tepny (a. pulmonalis). Tam měří jednak krevní tlak, jednak teplotu krve nebo koncentraci nějaké látky, tzv. indikátoru.



Princip techniky je následující: Chladná kapalina nebo nějaká měřitelná látka se rychle vstříkne do pravé síně. Kapalina ochladí kolem tekoucí krev nebo zředí vstřikovanou látku. Jakmile krev doputuje do a. pulmonalis, čidlo na konci katetru změří pokles teplotu nebo vzestup koncentrace indikátoru jako následující křivku.



Ze známého množství aplikovaného indikátoru a změřené křivky lze dopočítat srdeční výdej. Následující obrázek představuje plicní tepnu, v níž je křivka měřena.



Krev teče průtokem q, který v případě plicní tepny odpovídá minutovému srdečnímu výdeji. Za čas dt proteče kolmým průřezem plicní tepny objem krve $dV = q.dt$. Pokud je koncentrace indikátoru (nebo pokles teploty) v tomto elementárním objemu krve c, pak tento objem obsahuje $dN = c.dV = c.qdt$ indikátoru. Po celou dobu měření přitom musí postupně protéct veškeré množství podaného indikátoru N. Stačí tedy sečíst, zintegrovat, jednotlivá elementární množství látky v čase. Integrujeme opět od 0 do „nekonečna“.

$$N = \int_0^N dN = \int_0^\infty c.qdt = q \int_0^\infty cdt$$

Funkce $c(t)$ je změřená křivka. Integrál $\int_0^\infty cdt$ opět měří plochu pod křivkou. Pro srdeční výdej $CO = q$ (cardiac output) tedy dostáváme

$$CO = \frac{N}{\int_0^\infty cdt}.$$

Na první pohled kontraintuitivní je skutečnost, že čím nižší je srdeční výdej, tím větší je křivka i plocha pod ní.
Odvozenou rovnici je pro případ měření teploty krve namísto koncentrace indikátoru nutno lehce „obohatit“ o teplotu krve i podávané tekutiny a o tepelnou kapacitu krve. Rovnice se pak v literatuře označuje jako Stewart-Hamiltonova a zní

$$CO = \frac{V(T_b - T_i)k_1k_2}{\int_{t_1}^{t_2} \Delta T dt}.$$

kde V je objem podané tekutiny, T_b a T_i teploty krve a podané tekutiny, k_1 spojuje

hustotu a tepelnou kapacitu, k_2 je kalibrační konstanta a ΔT je aktuálně měřený rozdíl mezi teplotou protékající krve a teplotou krve před podáním roztoku.
Je zajímavé poznamenat, že se změřená křivka v průběhu putování krve cévami příliš nemění. Je téměř stejná ve větvích plicní tepny jako přímo v ní, ale podobná je i po průchodu plícemi v arteriální krvi. Toho lze využít k méně invazívnímu, ale jen lehce méně přesnému měření srdečního výdeje pomocí tzv. transpulmonální termodiluce (například systémem zvaným zvaným PiCCO). Chladný roztok se podává centrální katetrem do pravé síně a teplota se měří speciálním arteriálním katetrem až v arteria radialis nebo arteria femoralis. Odpadá tak invazivita a komplikace spojené s nutností zavedení katetru až do plicní tepny.

2.7. Obyčejné diferenciální rovnice

Nyní si v základech představíme další a zásadní oblast matematické analýzy, diferenciální rovnice. Jsou základním nástrojem matematického modelování. Tvrdí se, že diferenciální rovnice jsou nejpřirozenější formulací fyzikálních zákonů.

2.7.1. Idea diferenciální rovnice

Představme si jakoukoli algebraickou rovnici, např.

$$x^2 + 2x - 1 = 0.$$

Hledáme číslo x, které vyhovuje dané rovnici. Řešením algebraické rovnice je tedy číslo.

Řesme nyní jinou úlohu. Znáte nějakou funkci $f(x)$, jejíž derivace se rovná funkci samé? Samozřejmě, $f(x) = e^x$. Uvedená otázka je ale vlastně slovní popis rovnice

$$f'(x) = f(x).$$

Je to rovnice, která obsahuje derivaci nějaké neznámé funkce a jejímž řešením není číslo, nýbrž tato neznámá funkce. Takovou rovnici označujeme jako diferenciální. Řešením diferenciální rovnice je tedy funkce, nikoli číslo.

Je zřejmé, že této diferenciální rovnici vyhovuje kromě funkce $y(x) = e^x$ též funkce $y(x) = k.e^x$, kde k je libovolná konstanta. Řešení diferenciální rovnice tedy není jednoznačné, ale liší se v konstantě. Hodnotu konstanty pro konkrétní případ můžeme určit ze známých okolností popisovaného děje, např. ze známé hodnoty funkce v čase $t = 0$. Hovoříme o tzv. **počáteční nebo okrajové podmínce**.

Hledejme řešení rovnice $f'(x) = f(x)$, které zároveň splňuje požadavek, počáteční podmínku, aby hodnota funkce v bodě 0 byla 2, $f(0) = 2$.
Obecným řešení rovnice je $f(x) = k.e^x$. Aby platilo $2 = f(0) = k.e^0$, musí být $k = 2$. Hledaným řešení je tedy

$$f(x) = 2e^x.$$

Shrňme tedy, že **diferenciální rovnice** jsou rovnice, které obsahují derivaci nějaké neznámé funkce. Tato funkce je jejich řešením. Pokud je neznámá funkce funkcí jedné proměnné a rovnice tak obsahuje pouze obyčejné derivace, označuje se jako **obyčejná diferenciální rovnice**, často pod zkratkou ODE z anglického ordinary differential equations. Pokud je hledaná funkce funkcí více proměnných a rovnice tak obsahuje parciální derivace, označuje se jako **parciální diferenciální rovnice**, obvykle pod zkratkou PDE z anglického partial differential equations.

V této kapitole popíšeme analytické metody řešení ODE. **Analytickým řešením** se rozumí matematický výraz popisující hledanou funkci, např. $y = \cos x$. Poněvadž však většinu diferenciálních rovnic neumíme řešit analyticky, jsou pro praktické použití důležitější numerické metody řešení, které budou popsány v další kapitole. Výsledkem **numerického řešení** je graf nebo vybrané hodnoty hledané funkce, ale nikoli matematický výraz, který ji popisuje. V této kapitole se budeme věnovat analytickému řešení diferenciálních rovnic.

2.7.2. Základní pojmy

Vysvětlíme si několik základních pojmů z problematiky diferenciálních rovnic. Neznámou funkcí bude dále y, nezávisle proměnnou bude x, tedy $y(x)$. Pro derivaci budeme ekvivalentně používat zápis y' nebo $\frac{dy}{dx}$.

Řádem diferenciální rovnice se rozumí nejvyšší stupeň derivace přítomný v diferenciální rovnici. Je-li nejvyšší derivace první, pak jde o rovnici 1. řádu. Je-li nejvyšší derivace n-tá, pak je rovnice n-tého řád. Rovnice $y'x + y = 2x^2$ je tedy 1. řádu, rovnice $xy' + 2x = y'''$ je třetího řádu.

Lineární diferenciální rovnice je taková, kdy se neznámá funkce y nebo její derivace vyskytují pouze v součtu, nikoli v součinu. Rovnice $y' + 2y = x^2$ je tedy lineární, rovnice $yy' = x$ je nelineární. Poznamenejme, že x není neznámá, proto se součin x, resp. $f(x)$, a y v lineární rovnici vyskytovat může. Rovnice $xy' + 2y = x$ nebo $y \sin x = y'x^2$ jsou tedy též lineární.

Obecně můžeme každou **lineární diferenciální rovnici** 2. řádu (a obecně jakéhokoli řádu) zapsat jako

$$a(x)y'' + b(x)y' + c(x)y = d(x),$$

kde $a(x)$, $b(x)$, $c(x)$ i $d(x)$ jsou známé funkce x, ale nikoli y. Pokud jsou $a(x)$, $b(x)$ i $c(x)$ konstanty, hovoříme o **rovnici s konstantními koeficienty**. Na pravé straně rovnice se nevyskytuje neznámá y, pouze funkce x. Jestliže je pravá strana rovnice nulová, tedy $d(x) \equiv 0$, hovoříme o **homogenní** rovnici, jinak o rovnici **nehomogenní**.

Ve výše uvedeném příkladu jsme našli $y(x) = e^x$ jako jedno konkrétní řešení rovnice $y' = y$. Každé konkrétní řešení označuje jako **partikulární**. Existuje však, jak jsme již zmínili, nekonečně mnoho partikulárních řešení, která jsou všechna zapsatelná obecným tvarem $y(x) = k.e^x$, kde $k \in \mathbb{R}$. Takovému obecnému zápisu říkáme **obecné řešení**. Najít nějaké libovolné partikulární řešení, narozdíl od obecného řešení, může být snadné, lze jej např. uhádnout. Takovým zjevným partikulární řešení uvedené rovnice $y' = y$ je např. $y(x) = 0$.

2.7.3. Formulace diferenciální rovnice

Pro potřeby matematického modelování je důležité umět převést reálný problém do podoby diferenciální rovnice. Ukážeme si několik příkladů takového převodu. Opět využijeme nám již dobře známé manipulace s diferenciály.

Zformulujme diferenciální rovnice, které odpovídají následujícím popisům.

1. Molekuly látky spontánně degradují, přičemž pravděpodobnost degradace v následující sekundě je pro všechny molekuly stejná.

   Protože je pravděpodobnost degradace v následující sekundě pro všechny molekuly stejná, musí být počet molekul, které se v následujícím časovém úseku Δt rozpadnou ΔN, přímo úměrný jejich aktuálnímu počtu N. ΔN je kladné, pokud počet molekul roste. V našem případě počet molekul ubývá, platí proto $-\Delta N \approx k.N.\Delta t$. Pro velmi malé změny přechází Δ v diferenciál d. Hledaná diferenciální rovnice tedy zní

   $$\frac{dN}{dt} = -kN.$$

2. Bakterie se množí dělením jednotlivých buněk, zároveň pod vlivem antibiotik jednotlivé buňky umírají a zároveň růst bakterií omezuje konkurence o sdílené zdroje. Konkurence je úměrná skutečnosti, že se dvě bakterie setkají u stejného zdroje.

   Protože se bakterie množí dělením, je jejich přírůstek úměrný jejich počtu. Zároveň vlivem antibiotik jejich počet klesá, též úměrně počtu. Oba procesy se však liší rychlostní konstantou. Konkurence je vyjádřena pravděpodobností setkání u stejného zdroje, která je úměrná druhé mocnině počtu. Problém tak popisuje diferenciální rovnice

   $$\frac{dN}{dt} = k_1 N - k_2 N - k_3 N^2.$$

3. Pozitivní zpětná vazba je charakterizována skutečností, že rychlost nárůstu veličiny je (přímo) úměrná veličině samotné.

   Odpovídající diferenciální rovnice samozřejmě zní

   $$\frac{dx}{dt} = kx.$$

   Řešením je funkce

   $$x = x_0 e^{kx}.$$

   Taková veličina tedy exponenciálně rychle roste, což se vyústí v brzkou „katastrofu“, pokud jiný proces růst nezastaví.

2.7.4. Homogenní lineární diferenciální rovnice prvního řádu, metoda separace proměnných

Řešme nyní již zmíněnou rovnici $y' = y$ systematickým způsobem. Derivaci přepišme do tvaru podílu dvou diferenciálů, dy a dx,

$$\frac{dy}{dx} = y.$$

S diferenciály budeme opět zacházet jako s čísly, např. jimi násobit či dělit. Převedeme rovnici na tvar

$$\frac{dy}{y} = dx.$$

Tím jsme vše související s y přesunuli na levou stranu rovnice a vše související s x na pravou stranu. Postup označujeme jako **separaci proměnných**. Nyní před oba diferenciály napíšeme integrál a zintegrujeme, čímž vyřešíme rovnici.

$$\int \frac{dy}{y} = \int dx \;\rightarrow\; \ln y = x + C$$

Po úpravě dostaneme obecné řešení rovnice

$$y = e^{x+C} = e^C.e^x = ke^x,$$

kde $k := e^C$. Hodnotu konstanty získáme z počátečních podmínek. Požadujme například, aby $y(0) = y_0$. Pak je $k = y_0$ a odpovídající partikulární řešení zní

$$y = y_0 e^x.$$

Jen poznamenejme, že metoda má samozřejmě exaktní důkaz, popsaný formální postup sám o sobě důkazem není.

Vylučování antibiotika ledvinami

Antibiotikum vankomycin je z organizmu eliminováno převážně v nezměněné podobě ledvinami, přičemž se řídí tzv. kinetikou prvního řádu, kdy je rychlost eliminace látky přímo úměrná její plazmatické koncentraci c. Rychlostí eliminace látky se rozumí pokles její koncentrace v čase, tedy vlastně záporná derivace podle času, $-\frac{dc}{dt}$.
Eliminaci vankomycinu tak můžeme popsat diferenciální rovnicí

$$-\frac{dc}{dt} = kc.$$

kde k je konstanta úměrnosti, vyjadřující např. funkci ledvin. Rovnice je **separovatelná** do tvaru

$$\frac{dc}{c} = -kdt.$$

Po integraci dostaneme

$$\ln c = -kt + b \;\rightarrow\; c = B.e^{-kt},$$

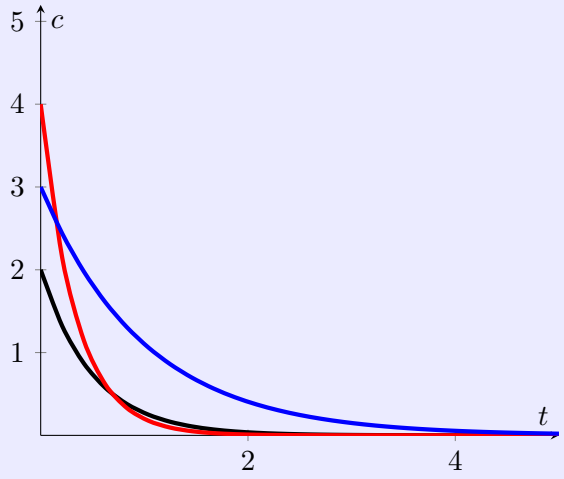
kde b, resp. $B = e^b$ je integrační konstanta. Tím jsme získali obecné řešení. Hodnotu konstanty B určíme z počáteční podmínky. Koncentraci vankomycinu v čase 0 označme c_0. Platí tedy

$$c_0 = B.e^{-k.0} = B.$$

Finálním partikulárním řešením proto je

$$c = c_0.e^{-kt}.$$

Příklady průběhů funkce pro různé počáteční koncentrace a hodnoty konstanty k ukazuje následující graf.



Poznamenejme ještě, že rovnice $\frac{dc}{dt} = -kc$ je lineární a homogenní, protože ji můžeme upravit na tvar s nulovou pravou stranou $c' + kc = 0$.

Viděli jsme, že kinetika prvního řádu vede na diferenciální rovnici, jejímž řešením je klesající exponenciála. K podobným diferenciálním rovnicím vede i řada dalších procesů známých z biologie a medicíny, jako jsou radioaktivní rozpad či monomolekulární přeměny molekul. Proto se klesající exponenciála vyskytuje v biologii tak často. Místo konstanty k se často používá poločas $t_{1/2}$. Za dobu jednoho poločasu klesne koncentrace látky na polovinu. Proto platí

$$c = \frac{c_0}{2} = c_0 e^{-kt_{1/2}} \rightarrow t_{1/2} = \frac{\ln 2}{k}.$$

Po 5 poločasech látka téměř zmizí, resp. klesne $2^5 = 32$-krát z původní koncentrace.

2.7.5. Nehomogenní lineární diferenciální rovnice prvního řádu, metoda variace konstant

Nyní si popíšeme způsob řešení nehomogenních lineárních diferenciálních rovnic prvního řádu, tzv. **metodu variace konstant**. Nejprve si postup ukážeme na konkrétním příkladu, poté obecně.

Nitrožilní podávání vankomycinu

Rozšíříme předchozí příklad s eliminací vankomycinu o situaci, kdy vankomycin současně kontinuálně podáváme infúzí rychlostí v_0. Zajímá nás, jak se vyvíjí plazmatická koncentrace v čase poté, co zapneme infuzi s vankomycinem. Problém nyní popisuje rovnice

$$\frac{dc}{dt} = v_0 - kc,$$

zahrnující jak eliminaci ledvinami, tak kontinuální podávání léku. Oproti předchozímu příkladu jde nyní o nehomogenní rovnici s nenulovou pravou stranou $c' + kc = v_0$, kde již nemůžeme separovat proměnné.
K řešení použijeme následující poněkud komplikovaný „trik“, označovaný jako **metoda variace konstant**. Nejprve si z nehomogenní rovnice vytvoříme rovnici homogenní tím,

že pravou stranu položíme rovnu 0, $c' + kc = 0$. Získáme tak odpovídající homogenní rovnici

$$\frac{dc}{c} = -kdt,$$

kterou jsme řešili již v předchozím příkladě a nalezli řešení

$$c = B.e^{-kt},$$

kde B byla konstanta. Variace konstant spočívá v tom, že konstantu B přestaneme považovat za konstantu, ale budeme ji dále považovat za funkci času, $B(t)$. Řešení původní nehomogenní rovnice tak budeme hledat ve tvaru

$$c = B(t).e^{-kt}.$$

Pro c' pak platí

$$\frac{dc}{dt} = B'(t).e^{-kt} - kB(t)e^{-kt}.$$

Po dosazení do nehomogenní rovnice dostaneme

$$B'(t).e^{-kt} - kB(t)e^{-kt} = v_0 - kB(t).e^{-kt}.$$

Po odečtení stejných členů na obou stranách získáme diferenciální rovnici pro $B(t)$

$$B'(t).e^{-kt} = v_0,$$

která je separovatelná do tvaru $dB = v_0 e^{kt} dt$. Proto

$$\int dB = \int v_0 e^{kt} dt \;\rightarrow\; B = \frac{v_0}{k} e^{kt} + D,$$

kde D je další integrační konstanta. Řešením původní nehomogenní rovnice tedy je

$$c = B(t)e^{-kt} = \left(\frac{v_0}{k} e^{kt} + D\right).e^{-kt} = \frac{v_0}{k} + De^{-kt}.$$

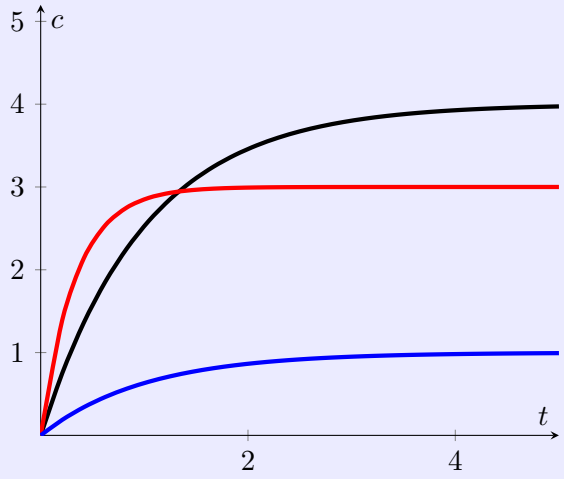
Na počátku požadujme nulovou koncentraci vankomycinu, tedy $c(0) = 0$, odkud plyne

$$0 = \frac{v_0}{k} + De^{-k.0} \;\rightarrow\; D = -\frac{v_0}{k}.$$

Konečně tak dostáváme definitivní partikulární řešení naší úlohy, totiž

$$c = \frac{v_0}{k}\left(1 - e^{-kt}\right).$$

Příklady řešení pro různé hodnoty parametrů, rychlosti infuze a funkce ledvin, ukazuje následující graf.



Koncentrace zpočátku rychle roste k setrvalé, stacionární hodnotě, pro níž (v nekonečném čase) platí

$$c_{stac} = \frac{v_0}{k}\left(1 - e^{-k\infty}\right) = \frac{v_0}{k}.$$

V reálném životě považujeme koncentraci za ustálenou po zhruba 5 poločasech.

Nyní zformulujeme postup pro obecný případ nehomogenní lineární diferenciální rovnici prvního řádu

$$a(x)y' + b(x)y = c(x).$$

V předchozím příkladu šlo o speciální případ s konstantními koeficienty $a = 1$, $b = k$ i $c = v_0$. Rovnice je nehomogenní, protože má nenulovou pravou stranu. Nejprve vyřešíme odpovídající homogenní rovnici

$$a(x)y' + b(x)y = 0.$$

Tu můžeme snadno separovat a dostaneme

$$\frac{dy}{y} = -\frac{b(x)}{a(x)}dx.$$

Po integraci pak máme

$$\ln y = \int -\frac{b(x)}{a(x)}dx + C \;\rightarrow\; y = ke^{\int -\frac{b(x)}{a(x)}dx},$$

kde k je konstanta. Nyní použijeme variaci konstanty k, kterou budeme dále považovat za funkci x, a budeme hledat řešení nehomogenní rovnice ve tvaru

$$y = k(x)e^{\int -\frac{b(x)}{a(x)}dx}.$$

Pro y' platí

$$y' = k'(x)e^{\int -\frac{b(x)}{a(x)}dx} - k(x)\frac{b(x)}{a(x)}e^{\int -\frac{b(x)}{a(x)}dx}.$$

Pro jasnost výkladu dodejme, že jsme zde derivovali pro nás možná neobvyklou složenou funkci $e^{\int -\frac{b(x)}{a(x)}dx}$, kde vnější funkce je $e^{něco}$ a vnitřní funkce je $\int -\frac{b(x)}{a(x)}dx$. Derivace vnitřní funkce je snadná, protože $\frac{d}{dx}\int f(x)dx = f(x)$. Aplikací pravidla pro derivaci složené funkce „vnější funkce

krát vnitřní funkce“ hned dostaneme

$$\frac{d}{dx}e^{\int -\frac{b(x)}{a(x)}dx} = e^{\int -\frac{b(x)}{a(x)}dx}\cdot\left(-\frac{b(x)}{a(x)}\right).$$

Dosadíme za y a y' do nehomogenní rovnice a dostaneme

$$a(x)k'(x)e^{\int -\frac{b(x)}{a(x)}dx} - a(x)k(x)\frac{b(x)}{a(x)}e^{\int -\frac{b(x)}{a(x)}dx} + b(x)k(x)e^{\int -\frac{b(x)}{a(x)}dx} = c(x).$$

Druhý a třetí člen se odečtou a získáme separovatelnou rovnici pro $k(x)$

$$\frac{dk(x)}{dx}a(x)e^{\int -\frac{b(x)}{a(x)}dx} = c(x).$$

Po separaci a integraci obdržíme

$$dk = \frac{c(x)}{a(x)}e^{\int \frac{b(x)}{a(x)}dx}dx \;\rightarrow\; k = \int \frac{c(x)}{a(x)}e^{\int \frac{b(x)}{a(x)}dx}dx + D.$$

Dosadíme do původní rovnice a získáme poněkud komplikovaný tvar obecného řešení.

$$y = e^{\int -\frac{b(x)}{a(x)}dx}\int \frac{c(x)}{a(x)}e^{\int \frac{b(x)}{a(x)}dx}dx + De^{\int -\frac{b(x)}{a(x)}dx}.$$

Počáteční podmínky pak vyberou potřebné konkrétní partikulární řešení.

2.7.6. Soustavy obyčejných diferenciálních rovnic prvního řádu

Soustavy obyčejných diferenciálních rovnic jsou mimořádně důležité, protože popisují řadu reálných procesů, kdy spolu různé proměnné interagují. Ilustrujme představu soustav diferenciálních rovnic dvěma příklady:

Chceme matematicky popsat vývoj koncentrace inzulínu a glukózy v čase. Obě látky se vzájemně ovlivňují. Inzulín snižuje koncentraci glukózy, zároveň je však tvorba inzulínu závislá na její koncentraci. Inzulín je současně odbouráván kinetikou prvního řádu, nezávisle na koncentraci glukózy. Glukózu budeme zároveň přivádět infúzí konstantní rychlostí v_0. Považujme nyní pro zjednodušení všechny vzájemné závislosti za lineární (s konstantami úměrnosti k_i), byť to neodpovídá fyziologické realitě. g bude značit koncentraci glukózy ($g := [\text{glukóza}]$) a I koncentraci inzulínu ($I := [\text{inzulín}]$). Změny koncentrace glukózy a inzulínu v čase můžeme popsat následující dvojicí diferenciálních rovnic

$$\begin{aligned}\frac{dg}{dt} &= v_0 - k_1 I\\ \frac{dI}{dt} &= k_2 g - k_3 I\end{aligned}$$

Hormon trijódthyronin (T3) tlumí či stimuluje tvorbu hormonu TSH, přičemž intenzita stimulace je popsána nějakou funkcí f koncentrace T3, $f([T3])$. Zároveň TSH stimuluje tvorbu T3, intenzita stimulace bude popsána funkcí $g([TSH])$. Jak TSH, tak T3 též spontánně degradují, s konstantami k_1 a k_2. Napišme odpovídající dvojici diferenciálních rovnic.

$$\frac{d[TSH]}{dt} = f([T3]) - k_1[TSH]$$
$$\frac{d[T3]}{dt} = g([TSH]) - k_2[T3]$$

V obou případech je dvojice diferenciálních rovnic „zvláštní“ tím, že ani jednu z rovnic nemůžeme řešit samostatně, neboť pro vyšetření každé z rovnic potřebujeme znát řešení druhé rovnice. Takovou dvojici rovnic označujeme jako **soustavu diferenciálních rovnic**.

Analogická situace nastává u soustavy algebraických rovnic, kde jsou jednotlivé rovnice též provázané. Soustava rovnic může být libovolně rozsáhlá. Řešit soustavu diferenciálních rovnic analyticky je možné jen v nejjednodušších případech, obvykle je nezbytné řešení numerické.

Soustavu algebraických rovnic můžeme řešit eliminací proměnných, kdy z jedné rovnice vyjádříme jednu z proměnných a dosadíme ji do ostatních rovnic. Zkusme použít podobný postup na soustavu diferenciálních rovnic popisujících glukózu a inzulín

$$\frac{dg}{dt} = v_0 - k_1 I$$
$$\frac{dI}{dt} = k_2 g - k_3 I.$$

Zkusíme eliminovat glukózu. Zderivujme druhou rovnici podle času a dosaďme poté za g' z první rovnice.

$$\frac{d^2 I}{dt^2} = k_2 \frac{dg}{dt} - k_3 \frac{dI}{dt} = k_2 v_0 - k_1 k_2 I - k_3 \frac{dI}{dt}$$

Skutečně jsme tak eliminovali glukózu a získali jsme nehomogenní lineární diferenciální rovnici druhého řádu s konstantními koeficienty

$$\frac{d^2 I}{dt^2} + k_3 \frac{dI}{dt} + k_1 k_2 I = k_2 v_0,$$

kterou budeme řešit v další kapitole. Jakmile pak nalezneme funkci $I(t)$, z první rovnice dopočteme $g(t)$. Později zjistíme, že oproti rovnicím prvního řádu určují hodnoty konstant k_i nejen konkrétní hodnoty nalezené funkce, ale i její kvalitativní charakteristiky, např. jestli řešení v čase konverguje, zda dochází k oscilacím apod.

Soustavu jsme tedy řešili tak, že jsme eliminovali proměnnou a ze soustavy dvou rovnic prvního řádu jsme získali jednu rovnici druhého řádu, kterou budeme dále analyticky řešit. Pro numerické řešení diferenciálních rovnic, které je z praktického pohledu důležitější, je ale postup obvykle opačný. Numericky řešit soustavu rovnic prvního řádu je totiž relativně snadné. Proto se nabízí, pokusit se rovnici vyššího řádu naopak převést na soustavu rovnic prvního řádu a tu pak numericky řešit. Postup je snadný. Jako příklad vezměme rovnici druhého řádu

$$y'' + by' + y = d$$

s neznámou funkcí $y(x)$. Definujme novou funkci $z(x) := y'$ jako derivaci funkce y. Pak platí

$z' = y''$ a rovnici můžeme přepsat do tvaru

$$z' + bz + y = d.$$

Můžeme sestavit novou soustavu dvou rovnic pro 2 neznámé funkce y a z, kterou dále numericky řešíme.

$$\frac{dz}{dx} = d - y - bz$$
$$\frac{dy}{dx} = z$$

Pro potřeby numerického řešení diferenciálních rovnic kteréhokoli řádu tedy stačí umět vyřešit soustavu rovnic prvního řádu.

2.7.7. Lineární diferenciální rovnice druhého řádu

V poslední části kapitol o diferenciálních rovnicích se budeme zabývat analytickým řešení nehomogenní rovnice druhého řádu s konstantními koeficienty. Takovou rovnicí je např. výše odvozená rovnice pro inzulín

$$\frac{d^2I}{dt^2} + k_3\frac{dI}{dt} + k_1k_2I = k_2v_0.$$

Pro jednoduchost zápisu přeznačme konstanty na b, c a d. Před nejvyšší derivací konstanta být nemusí, poněvadž je nutně nenulová a lze jí krátit.

$$\frac{d^2I}{dt^2} + b\frac{dI}{dt} + cI = d$$

Řešme opět nejprve homogenní rovnici

$$\frac{d^2I}{dt^2} + b\frac{dI}{dt} + cI = 0.$$

Vysvětlíme nejprve pojem **lineární kombinace**. Mějme libovolné funkce I_1 a I_2. Lineární kombinací funkcí I_1 a I_2 se rozumí jejich násobky a součty. Nechť p a q jsou libovolná reálná čísla. Pak každou funkci

$$I(x) := pI_1(x) + qI_2(x)$$

označíme jako lineární kombinaci funkcí I_1 a I_2.

Každá homogenní diferenciální rovnice má pozoruhodnou vlastnost. Jestliže nějaké funkce I_1 a I_2 jsou řešeními homogenní rovnice, pak i jejich jakákoli lineární kombinace je řešením. Důkaz je snadný. Nechť tedy I_1 a I_2 jsou řešeními homogenní rovnice. Pak nutně platí

$$\frac{d^2I_1}{dt^2} + b\frac{dI_1}{dt} + cI_1 = 0$$

a

$$\frac{d^2I_2}{dt^2} + b\frac{dI_2}{dt} + cI_2 = 0.$$

Dosaďme nyní $I(x) := pI_1(x) + qI_2(x)$ do homogenní rovnice, rovnici upravme a využijme, co

už víme o funkcích I_1 a I_2.

$$\frac{d^2I}{dt^2} + b\frac{dI}{dt} + cI = p\left(\frac{d^2I_1}{dt^2} + b\frac{dI_1}{dt} + cI_1\right) + q\left(\frac{d^2I_2}{dt^2} + b\frac{dI_2}{dt} + cI_2\right) = 0 + 0 = 0$$

Vidíme tedy, že i lineární kombinace $I(x)$ skutečně splňuje homogenní rovnici a je tedy též jejím řešením, což jsme chtěli dokázat.

Nehomogenní diferenciální rovnice tuto vlastnost bohužel nesplňuje. Má však jinou pozoruhodnou vlastnost. Obecné řešení nehomogenní rovnice I_{ON} lze získat jako součet kteréhokoli jednoho partikulárního řešení nehomogenní rovnice I_{PN} a obecného řešení homogenní rovnice I_{OH}.

$$I_{ON} = I_{PN} + I_{OH}$$

Najít nějaké partikulární řešení nehomogenní rovnice může být snadné, lze ho často „uhádnout“. Např. zjevným partikulární řešením nehomogenní rovnice

$$\frac{d^2I}{dt^2} + b\frac{dI}{dt} + cI = d$$

je konstantní řešení $I_{PN} = \frac{d}{c}$. Pak už stačí „pouze“ najít obecné řešení homogenní rovnice.

I zde je důkaz snadný, proto si jej ukážeme. Představme si, že už známe nějaké obecné řešení nehomogenní rovnice I_{ON} a známe rovněž jakékoli jedno řešení partikulárního řešení nehomogenní rovnice I_{PN}. Pak obě tato řešení splňují nehomogenní rovnici, platí tedy

$$\frac{d^2I_{ON}}{dt^2} + b\frac{dI_{ON}}{dt} + cI_{ON} = d$$

i

$$\frac{d^2I_{PN}}{dt^2} + b\frac{dI_{PN}}{dt} + I_{PN}I = d.$$

Pokud obě rovnice odečteme, zjistíme, že

$$\frac{d^2(I_{ON} - I_{PN})}{dt^2} + b\frac{d(I_{ON} - I_{PN})}{dt} + c(I_{ON} - I_{PN}) = 0.$$

$I_{ON} - I_{PN}$ je tedy nutně řešením homogenní rovnice $I_{OH} = I_{ON} - I_{PN}$. Proto je jasné, že platí zmíněná věta $I_{ON} = I_{PN} + I_{OH}$.

Pokračujme dále. Zkusme nyní najít řešení homogenní rovnice, a to ve tvaru $I = e^{kt}$. Po dosazení do rovnice a zkrácení e^{kt} dostaneme

$$k^2e^{kt} + bke^{kt} + ce^{kt} = 0 \;\rightarrow\; k^2 + bk + c = 0.$$

Aby tedy $I = e^{kt}$ bylo řešením, musí k splňovat rovnici, musí proto platit

$$k_{1,2} = \frac{-b \pm \sqrt{b^2 - 4c}}{2}.$$

Můžeme rozlišit 3 situace podle hodnoty diskriminantu $D = \sqrt{b^2 - 4c}$.

1. **D > 0**

   Dostáváme 2 reálné hodnoty $k_{1,2}$ a 2 jim odpovídající řešení $I_1 = e^{k_1t}$ a $I_2 = e^{k_2t}$. Řešení

jsou tzv. **lineárně nezávislá**, což znamená, že nelze vyjádřit jedno řešení jako násobek druhého řešení. V hlubší teorii diferenciálních rovnic se ukazuje, že rovnice n-tého řádu má nejvýše n lineárně nezávislých řešení. Protože řešíme rovnici druhého řádu, jsou řešení I_1 a I_2 veškerá lineárně nezávislá řešení. Jakékoli další řešení $I(x)$ homogenní rovnice lze vyjádřit v již známém tvaru $I(x) = pI_1(x) + qI_2(x)$, tedy

$$I_{OH} = pe^{k_1 t} + qe^{k_2 t}.$$

Obecné řešení nehomogenní rovnice je pak

$$I_{ON} = \frac{d}{c} + pe^{k_1 t} + qe^{k_2 t}.$$

Konstanty p a q se určí z počátečních podmínek, které v případě rovnice druhého řádu musí být dvě, např. počáteční koncentrace inzulínu a počáteční rychlost tvorby inzulínu (což vlastně nepřímo vyjadřuje počáteční koncentraci glukózy).

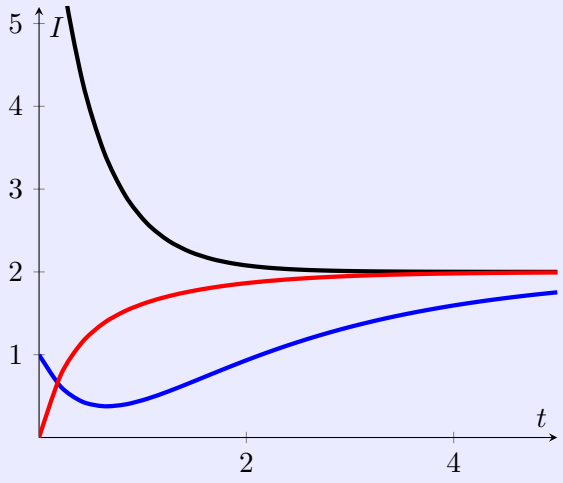
V předchozím příkladu s inzulínem byly b i c kladné, proto je v případě kladného diskriminantu určitě $0 < \sqrt{b^2 - 4c} < b$ a obě konstanty

$$k_{1,2} = \frac{-b \pm \sqrt{b^2 - 4c}}{2} < 0$$

jsou záporné. Obecné řešení I_{ON} proto můžeme napsat jako

$$I_{ON} = \frac{d}{c} + pe^{-|k_1|t} + qe^{-|k_2|t}.$$

S časem tedy exponenciální komponenty klesají k 0 a zůstává pouze stacionární hladina inzulínu $\frac{d}{c}$. Exponenciální komponenty se uplatní pouze zpočátku, než se ustaví stacionární hladina. Obrázek ukazuje možné průběhy koncentrací inzulínu, pokud je počáteční hladina nad nebo pod hladinou stacionární.



2. **D = 0**

Pro nulový diskriminant dostáváme jediné lineárně nezávislé řešení ve tvaru e^{kt}, totiž

$$I_1 = e^{\frac{-b}{2}t}.$$

Vzhledem k nulovému diskriminantu platí

$$b^2 - 4c = 0 \;\rightarrow\; c = \frac{b^2}{4},$$

čímž se diferenciální rovnice převádí na tvar

$$\frac{d^2I}{dt^2} + b\frac{dI}{dt} + \frac{b^2}{4}I = 0.$$

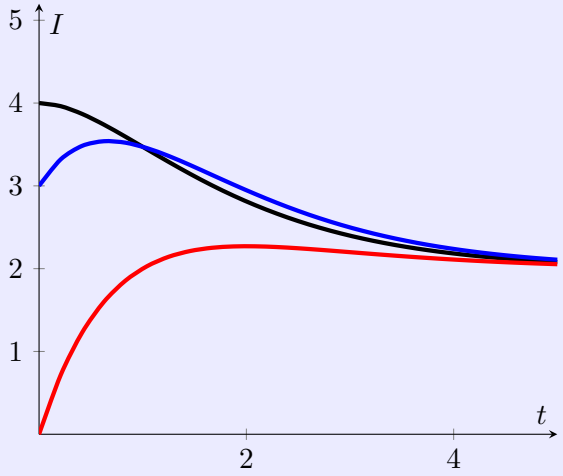
Prozradíme si její druhé lineárně nezávislé řešení. Je jím

$$I_2 = te^{\frac{-b}{2}t}.$$

Pro obecné řešení nehomogenní rovnice s nulovým diskriminantem tedy získáme

$$I_{ON} = \frac{d}{c} + pe^{\frac{-b}{2}t} + qte^{\frac{-b}{2}t}.$$

V příkladu s inzulínem bylo b kladné, takže řešení opět obsahuje klesající exponenciály a je kvalitativně podobné řešení s kladným diskriminantem, blíží se ke stacionární hladině. Několik možných průběhů funkcí ukazuje obrázek.



3. **D < 0**

Kvalitativně odlišné řešení obdržíme pro záporný diskriminant, kdy pro odmocninu neexistuje reálné řešení, ale existují 2 řešení v oboru komplexních čísel. Odmocninu ze záporného diskriminantu můžeme upravit do následujícího komplexního tvaru.

$$\sqrt{D} = \sqrt{-|D|} = \sqrt{-1}\sqrt{|D|} = i\sqrt{|D|}$$

Proto

$$k_{1,2} = \frac{-b \pm \sqrt{D}}{2} = \frac{-b \pm i\sqrt{|D|}}{2}.$$

Dostáváme tak 2 komplexně sdružená lineárně nezávislá řešení

$$I_1 = e^{\frac{-b+i\sqrt{|D|}}{2}} = e^{-\frac{b}{2}t}e^{i\frac{\sqrt{|D|}}{2}t}$$
$$I_2 = e^{\frac{-b-i\sqrt{|D|}}{2}} = e^{-\frac{b}{2}t}e^{-i\frac{\sqrt{|D|}}{2}t}.$$

To je na první pohled podivný výsledek, poněvadž v reálném fyzickém světě musí být řešení reálná. Komplexní řešení je pouze matematická konstrukce. Můžeme však využít výše popsanou vlastnost homogenní lineární rovnice, že součet řešení je též řešením. Zkusme obě komplexně sdružená řešení převést do goniometrického tvaru a sečíst.

$$\begin{aligned} I_1 + I_2 &= e^{-\frac{b}{2}t}.\left[e^{i\frac{\sqrt{|D|}}{2}t} + e^{-i\frac{\sqrt{|D|}}{2}t}\right] \\ &= e^{-\frac{b}{2}t}.\left[\cos\frac{\sqrt{|D|}}{2}t + i\sin\frac{\sqrt{|D|}}{2}t + \cos\frac{\sqrt{|D|}}{2}t - i\sin\frac{\sqrt{|D|}}{2}t\right] \\ &= 2e^{-\frac{b}{2}t}\cos\frac{\sqrt{|D|}}{2}t \end{aligned}$$

Eliminovali jsme tak komplexní členy a zbylo nám reálné řešení! Poznamenejme, že druhé lineárně nezávislé řešení bychom získali odečtením obou řešení. Řešení by bylo čistě imaginární a pro reálný svět irelevantní.

Pro obecné řešení nehomogenní rovnice se záporným diskriminantem tak získáme

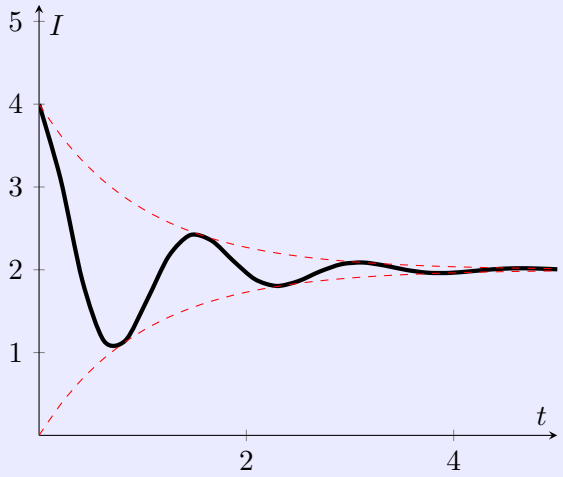
$$I_{ON} = \frac{d}{c} + pe^{-\frac{b}{2}t}\cos\frac{\sqrt{|D|}}{2}t.$$

Řešení je kvalitativně zcela odlišné od řešení předcházející, vyskytují se v něm totiž oscilace. Ve výrazu

$$e^{-\frac{b}{2}t}\cos\frac{\sqrt{|D|}}{2}t$$

můžeme rozlišit dvě části - oscilaci s konstantní amplitudou $\cos\frac{\sqrt{|D|}}{2}t$ a exponenciálně klesající amplitudu $e^{-\frac{b}{2}t}$. V čase tedy oscilující komponenta odezní a zbyde konstantní hladina inzulínu $I_0 = \frac{d}{c}$.

Obrázek ukazuje možný průběh koncentrace inzulínu. Červené čárkované linie vyznačují exponenciálně klesající amplitudu oscilací.



Všimněme si, že parametry původní rovnice, resp. soustavy rovnic, určovaly, zda dojde či nedojde k oscilacím. Oscilace v regulačních systémech je tedy možná, ale nikoli nutná.

3. Poznámky k numerické matematice

Řadu, či spíše většinu reálných úloh nelze řešit analyticky. Nepodaří se nám najít jednoduchý vztah, vzorec, do něhož můžeme dosadit a ihned zjistit výsledek. Téměř zcela to platí pro diferenciální rovnice. V kapitole 2.7 jsme řešili všechny úlohy analyticky, to byly ale ve skutečnosti pečlivě vybrané výjimky. Většinou je třeba úžít nějaký jiný postup, abychom získali alespoň číselný výsledek, když už nemáme analytické řešení. Rozvojem těchto postupů a a studiem jejich vlastností se zabývá numerická matematika. Formuluje též algoritmy k hledání řešení. Tím se pohybuje na hranici matematiky a informatiky. V téměř synonymním významu též hovoříme o disciplíně zvané výpočetní věda, computational science (nikoli computer science).

3.1. Numerické řešení algebraických rovnic

Analyticky umíme řešit jednoduché algebraické rovnice, jako $x^2 + 3x + 2 = 0$. Nelineární algebraické rovnice, např. $\cos x = x$ musíme řešit numericky. Zde si představíme 3 postupy numerického řešení algebraických rovnic - metodu půlení intervalů, metodu prosté iterace a metodu Newtonovu. Všechny tři metody mají **iterativní** charakter.

Iterace znamená, že se znovu a znovu opakuje stejný výpočet s odlišnými čísly, přičemž výstup i-tého kroku je vstupem pro krok $i + 1$. Na počátku je třeba zvolit nějaký odhad řešení, označme jej x_0. Tato hodnota je vstupem pro první iteraci. Výsledky je nový a snad i přesnější odhad řešení, x_1. Ten je vstupem 2. iterace, výstupem je ještě přesnější odhad x_2. Následují další iterace. Pokud se již řešení mezi jednotlivými iteracemi příliš nemění, změna je např. menší než předem zvolené **kritérium ukončení** ε, $|x_i - x_{i-1}| < \varepsilon$, program se ukončí a vypíše aktuální hodnotu x_i.
Různé postupy se liší konvergencí a efektivitou. **Konvergencí** se míní skutečnost, že se iterace stále více blíží správnému řešení. Konvergence může záviset na volbě počátečního x_0 - pro některá x_0 může iterace konvergovat, pro jiná nikoli. **Efektivitou** se rozumí počet provedených iterací, než program dosáhne kritéria ukončení.

Každou algebraickou rovnici lze upravit na tvar $f(x) = 0$. Hledejme tedy numerické řešení této rovnice. Nechť je řešením rovnice x_s, platí tedy $f(x_s) = 0$.

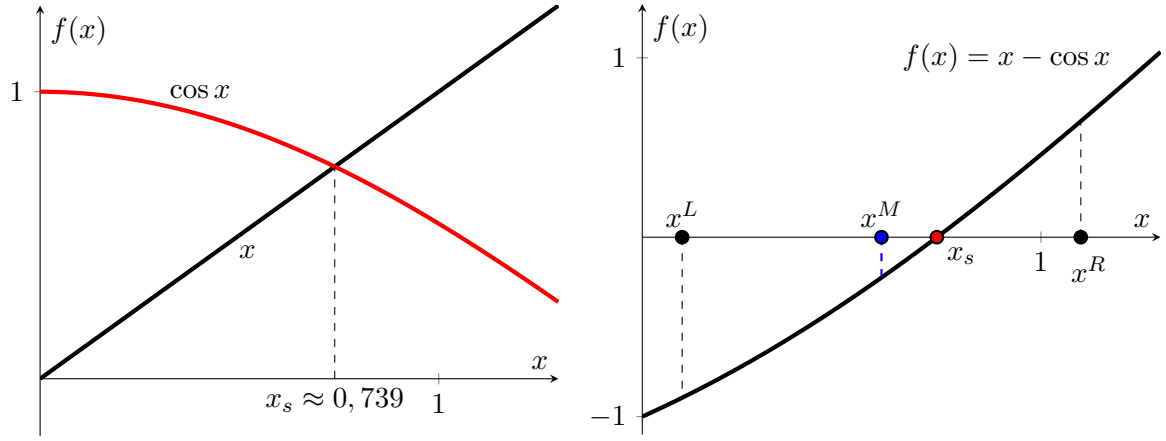
3.1.1. Metoda půlení intervalů

Metoda půlení intervalů je nejjednodušší. Obr. 3.1 ji demonstruje pro případ funkce $f(x) = x - \cos x$. Je třeba zvolit dvě počáteční hodnoty x_0^L a x_0^R, jednu vlevo a druhou vpravo od x_s. Tyto dvě hodnoty vymezují interval, v němž se nachází řešení. Budeme uvažujeme pouze situace, kdy funkce $f(x)$ protíná osu x (v bodě x_s) a v okolí bodu x_s je monotónní, jako na obr. 3.1. V takovém případě totiž musí mít $f(x_0^L)$ a $f(x_0^R)$ opačná znamení, protože v x_s je funkce nulová, $f(x_s) = 0$. Opačné znamení vyjádříme podmínkou $f(x_0^L).f(x_0^R) < 0$. Neuvažujeme tedy situaci, kdy funkce osu neprotíná, pouze se jí dotýká, jako v případě funkce x^2 v bodě $x = 0$.

V každé iteraci najdeme prostřední hodnotu x^M mezi oběma krajními hodnotami

$$x_i^M := \frac{x_i^R - x_i^L}{2}$$

a určíme její funkční hodnotu, $f(x_i^M)$. Uvažme, že se řešení x_s určitě musí nacházet mezi x_i^M a tou z krajních hodnot, která má opačné znaménko než $f(x_i^M)$. Pro další iteraci proto definujeme novou dvojici krajních hodnot x_{i+1}^L a x_{i+1}^R tak, že vždy jednou z nových krajních hodnot se stane x_i^M a druhá se nezmění. Přitom $x_{i+1}^L := x_i^M$ a $x_{i+1}^R := x_i^R$, pokud má pravá hodnota opačné znaménko ($f(x_i^M).f(x_i^R) < 0$), a $x_{i+1}^R := x_i^M$ a $x_{i+1}^L := x_i^L$, pokud má levá hodnota opačné znaménko. V každé iteraci se interval obsahují řešení x_s zkrátí na polovinu, odtud název metody. Iterace se provádějí tak dlouho, až je šířka intervalu menší než předem zvolené ε. Dokonce stačí polovina intervalu, protože x_i^M se od x_s liší nanejvýš o polovinu délky intervalu $[x_i^L, x_i^R]$. Výhodou metody je její jistá konvergence, pokud se nám podaří najít monotónní úsek funkce v okolí řešení.



(a) Rovnost funkcí. $x = \cos(x)$.

(b) Kořen rovnice. $f(x) = x - \cos(x) = 0$.

Obrázek 3.1.: Metoda půlení intervalů.

Napišme v Pythonu program, který implementuje metodu půlení intervalů a řeší rovnici $\cos(x) = x$.

```
import math

def g(x):
    return x-math.cos(x)

epsilon = 0.001 # požadovaná přesnost řešení
j=0         # sledování počtu iterací
xL=0.0      # levá krajní hodnota počátečního intervalu
xR=1.0      # pravá krajní hodnota počátečního intervalu

while abs(xR-xL)/2 > epsilon: # kritérium ukončení iterací
    xM = (xL+xR)/2     # definice středu intervalu
    if g(xM)*g(xR)<0: # volba nových krajních hodnot
        xL = xM
    else:
```

```
        xR = xM
    j+=1   # přibyla 1 iterace

print(xL) # levá krajní hodnota konečného intervalu
print(xR) # pravá krajní hodnota konečného intervalu
print(j)  # počet provedených iterací
.
```

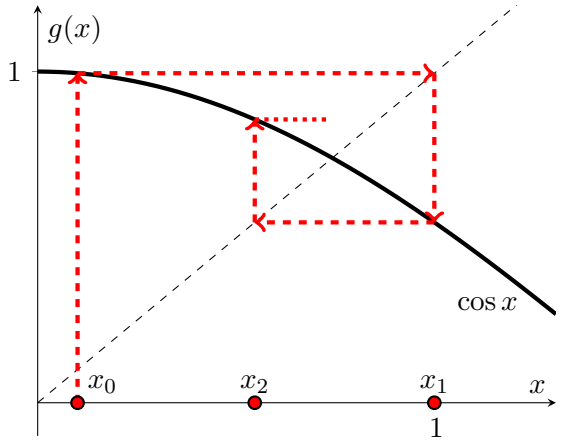
V daném příkladu získáme pro $\varepsilon = 0,001$ po 9 iteracích interval řešení $[0.73828125, 0.740234375]$. Pro představu přesnosti dodejme, že na 8 desetinných míst je $x_s = 0,73908513$.

3.1.2. Metoda prosté iterace

Další metodou, zajímavou, ale asi nejméně používanou, je metoda prosté iterace. Je nutné z řešené rovnice $f(x) = 0$ vyjádřit x „jako funkci x“, tedy rovnici přeformulovat do tvaru $x = g(x)$. Například rovnici $x - \cos x = 0$, kde $f(x) = x - \cos x$, přeformulujeme do tvaru $x = \cos x$, kde $g(x) = \cos x$. Opět se zvolí počáteční odhad řešení x_0. Každý krok iterace je charakterizován jednoduchým předpisem

$$x_{i+1} := g(x_i).$$

Princip algoritmu objasňuje obr. 3.2. Algoritmus „postupuje“ ve směru červených šipek. Problematické jsou podmínky konvergence. Aby algoritmus konvergoval, je nutné, aby pro všechny hodnoty mezi x_0 a řešením x_s platilo, že $|g'(x)| < 1$. Funkce tedy nesmí v žádném bodě tohoto intervalu růst nebo klesat příliš rychle. To často neplatí. Též rychlost konvergence závisí na $|g'(x)|$ a může být nízká.



Obrázek 3.2.: Metoda prosté iterace. $g(x) = \cos(x)$.

Implementujme v Pythonu algoritmus řešící rovnici $\cos(x) = x$ pomocí prosté iterace.

```
import math

def g(x):
    return math.cos(x)
```

```
j=0              # sledování počtu iterací
epsilon = 0.001  # požadovaná přesnost řešení
x1=0.9           # počáteční hodnota
x2=g(x1)

while abs(x1-x2) > epsilon: # kritérium ukončení iterací
    x1=x2        # iterační předpis
    x2=g(x1)
    j+=1         # přibyla 1 iterace

print(x1)        # předposlední hodnota iterace
print(x2)        # poslední hodnota iterace
print(j)         # počet provedených iterací
```

Pro stejnou přesnost $\varepsilon = 0,001$ jako v předchozím příkladě získáme z poslední iterace dvojici hodnot $0,7386421766270547$ a $0,7393834415834122$, avšak až po 15 iteracích, oproti předchozím 9 iteracím.

3.1.3. Newtonova metoda

Populární je Newtonova metoda, označovaná též Newton-Raphsonova metoda nebo metoda tečen. Konverguje velmi rychle, ale nikoli vždy. Řešíme opět rovnici $f(x) = 0$. Současně však musíme znát derivaci funkce f, f'. Opět je nutné poskytnou počáteční odhad řešení x_0. Princip metody ilustruje obr. 3.3. V bodě x_0 zjistíme derivaci $f'(x_0)$, čímž známe směrnici tečny v tomto bodě. Posuneme se do bodu x_1, kde tečna protíná osu x. Tím vznikne bod x_1. Zde opět zjistíme hodnotu derivace a „zkonstruujeme“ tečnu k funkci v tomto bodě. Obecně x_{i+1} je bod, kde tečna v bodě x_i protíná osu x. Z obr. 3.3 je zřejmé, že platí

$$f'(x_0) = \frac{f(x_0)}{x_0 - x_1} \rightarrow x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}.$$

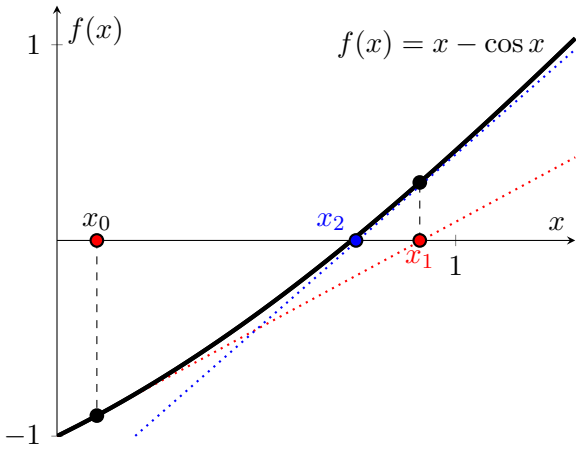
Pro iterační předpis proto obecně platí

$$x_{i+1} := x_i - \frac{f(x_i)}{f'(x_i)}.$$

Poznamenejme pro úplnost, že pokud derivaci neznáme a nahradíme ji odhadem derivace jako podílu 2 rozdílů ze dvou u sebe blízko ležících bodů, přechází metoda tečen v tzv. metodu sečen.

Implementujme v Pythonu algoritmus řešící opět stejnou rovnici $\cos(x) = x$, tentokrát Newtonovou metodou.

```
import math

def f(x):                    # řešená rovnice
    return x-math.cos(x)

def fdev(x):                 # derivace funkce f
```



Obrázek 3.3.: Newtonova metoda. $f(x) = x - \cos(x)$.

```
    return 1+math.sin(x)

j=0                     # sledování počtu iterací
epsilon = 0.001         # požadovaná přesnost řešení
x0=1                    # počáteční hodnota
x1=x0-f(x0)/fdev(x0)    # "nultá" iterace

while abs(x0-x1) > epsilon: # kritérium ukončení iterací
    x0=x1
    x1=x0-f(x0)/fdev(x0)    # iterační předpis
    j+=1                    # přibyla 1 iterace

print(x0)       # předposlední hodnota iterace
print(x1)       # poslední hodnota iterace
print(j)        # počet provedených iterací
```

Pro stejnou přesnost $\varepsilon = 0,001$ jako v předchozím příkladě získáme z poslední iterace dvojici hodnot $0,7391128909113617$ a $0,739085133385284$. Stačily nám k tomu pouhé 2 iterace!! Počet iterací přitom s rostoucí přesností roste velmi pomalu. Například pro řešení s přesností 11 desetinných míst ($\varepsilon = 0,00000000001$) stačí pouhé 4 iterace ($x_s = 0.7390851332151607$).

3.2. Numerické řešení diferenciálních rovnic

Numerické řešení diferenciálních rovnic je stěžejní téma pro potřeby matematického modelování. Představíme si dva nejznámější iterativní postupy, jednodušší Eulerovu metodu a složitější, ale přesnější metodu podle Runga a Kutty.

Diferenciální rovnici musíme nejprve převést do explicitního tvaru, kdy na levé straně rovnice se vyskytuje pouze derivace a na pravé straně všechny členy bez derivace, tedy do podoby

$$\frac{dy}{dx} = f(y, x).$$

Hledanou neznámou bude funkce $y(x)$. Pokud není derivace vyjádřena explicitně, což často ani není možné, hovoříme o tzv. implicitním tvaru diferenciální rovnice, který obecně můžeme zapsat jako $f(y', y, x) = 0$. Pak je do každého kroku numerického řešení nutno vložit řešení algebraické rovnice pomocí některé z metod uvedených v předchozí kapitole.

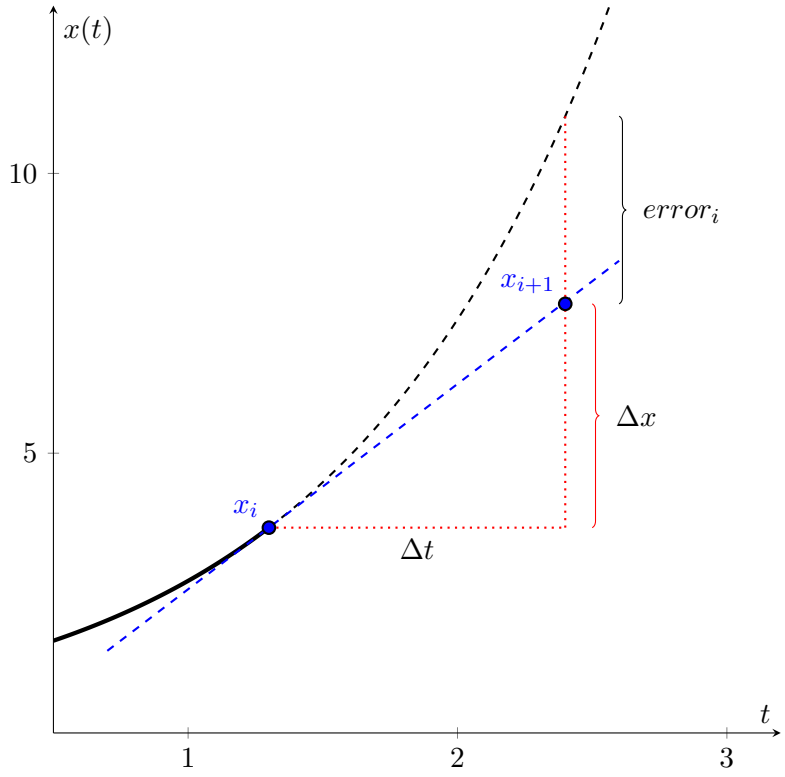
Vyjádřeme diferenciální rovnice v explicitním tvaru.

1. $xy' + 4y = x^2\sqrt{y}$. Neznámou funkcí je y, proměnnou je x.
$$y' = \frac{x^2\sqrt{y} - 4y}{x}$$
2. $\sqrt{x'}t + xt^2 = 0$. Neznámou funkcí je x, proměnnou je t.
$$x' = \left(\frac{-xt^2}{t}\right)^2 = x^2t^2$$
3. $\sin(z't^2) + (z')^2 = zt$. Neznámou funkcí je z, proměnnou je t.
V tomto případě nelze vyjádřit z' v explicitním tvaru. Rovnici musíme řešit v implicitním tvaru.

3.2.1. Eulerova metoda

Eulerova metoda využívá skutečnosti, že derivace je podíl dvou (nekonečně) malých rozdílů, přibližně tedy platí
$$\frac{dx}{dt} \approx \frac{\Delta x}{\Delta t}.$$
Výraz pro derivaci $x' = f(x, t)$ tak můžeme přepsat do přibližného tvaru
$$\frac{\Delta x}{\Delta t} \approx f(x, t) \;\rightarrow\; \Delta x \approx f(x, t).\Delta t.$$
Princip metody ilustruje obr. 3.4. Pokud známe derivaci v bodě, pak přibližně víme, jakou hodnotu bude funkce mít o Δt později. Musí být známa počáteční hodnota $x(0) = x_0$ v čase $t = 0$ resp. $t = t_0$. Musíme též zvolit časový krok Δt. V každém kroku metody se posuneme v čase o Δt dopředu a o Δx ve funkční hodnotě, čímž získáme novou dvojici hodnot x_{i+1} a t_{i+1} po $(i + 1)$-ním kroku z původních x_i a t_i po i-tém kroku. Iterativní předpis je tak definován výrazem
$$x_{i+1} := x_i + f(x, t).\Delta t$$
Je patrné, že se hodnota x_{i+1} liší od správné hodnoty $x(t+\Delta t)$. V i-tém kroku tak vzniká chyba $error_i = x(t + \Delta t) - x_{i+1}$. Chyba zjevně závisí na velikosti kroku Δt. Drobné chyby se však kumulují v každém kroku a výsledná celková chyba postupně roste. Celkovou chybu někdy nelze eliminovat ani zmenšením kroku Δt. Pak sice klesne chyba jednotlivého kroku, ale vzroste počet kroků a celková kumulativní chyba klesnout nemusí. Metoda podle Rungy a Kutty, popsaná v další podkapitole, tento problém řeší.

Implementujme v Pythonu Eulerovu metodu pro řešení diferenciální rovnice $x' = -x$. Počáteční podmínka zní $x(0) = 10$. Hledanou funkcí je x, proměnnou je t. Iterativní předpis bude $x_{i+1} := x_i - x_i.\Delta t$.

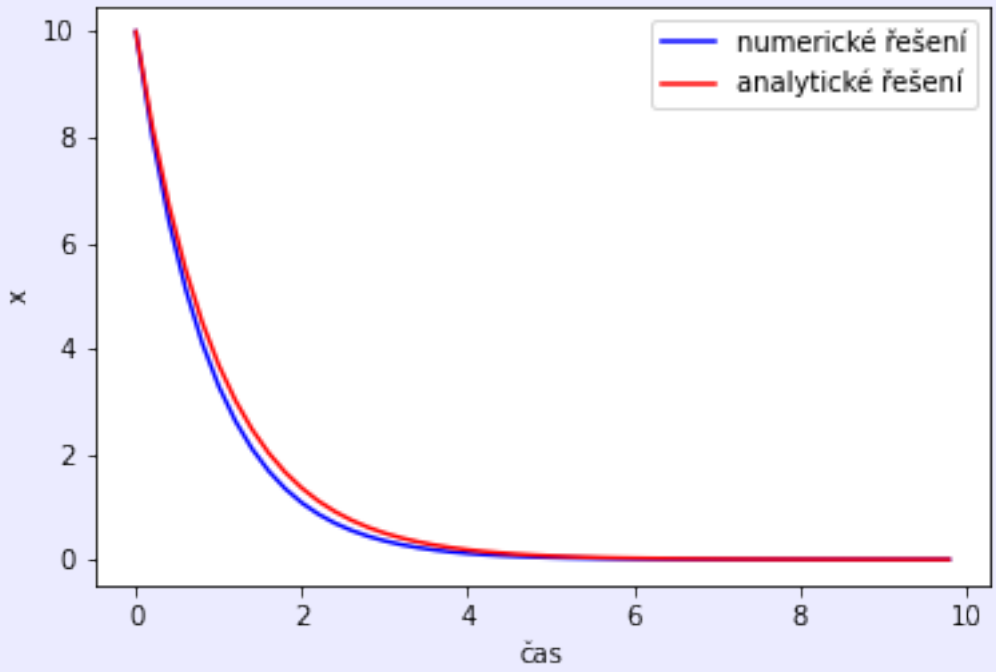


Obrázek 3.4.: Eulerova metoda

```
import numpy as np
import matplotlib.pyplot as plt

size = 50             # hodnotu funkce chceme najít v 50 bodech
t = np.linspace(0, 10, size)       # vektor časových bodů
x_num = np.linspace(0, 10, size)  # vektor numerických hodnot
x_anal = np.linspace(0, 10, size) # vektor analytických hodnot

def g(a):                          # derivace hledané funkce
    return -a

dt = 10/size   # časový krok (v tomto případě 10/50 = 0.2)

i=0
x[i] = 10  # počáteční hodnota funkce - x(t0)
t[i] = 0   # počáteční čas

while i < (size-1):     # iterace
    dx = dt * g(x_num[i])
    x_num[i+1] = x_num[i] + dx
    t[i+1] = t[i] + dt
    i += 1

x_anal = 10*np.exp(-t)  # analytické řešení
```

```
# graf řešení
fig, axes = plt.subplots()
axes.plot(t, x_num, 'b', label = 'numerické řešení')
axes.plot(t, x_anal, 'r', label = 'analytické řešení')
axes.set_xlabel('čas')
axes.set_ylabel('x')
axes.legend()
```

Při počáteční podmínce $x(0) = 10$ je analytickým řešením $x = 10e^{-t}$. Numerické i analytické řešení je zobrazeno na obrázku. Je patrna drobná odlišnost obou křivek, tedy chyba Eulerovy metody.



3.2.2. Metoda Rungeho a Kutty

Jak je jistě patrno, je Eulerova metoda velmi jednoduchá, ale nepříliš přesná. Výrazně přesnější metodu numerického řešení diferenciálních rovnic vypracovali počátkem 20. století němečtí matematici Carl Runge a Wilhelm Kutta. Metoda se snaží odstranit nepřesný odhad Δx po kroku Δt, tedy snížit chybu $error_i$. Hodnotu $x(t + \Delta t)$ neodhaduje jednou rovnicí, ale v několika navzájem propojených rovnicích, též označovaných jako „kroky“. Jde o tzv. vícekrokovou metodu. Představíme si nejpoužívanější variantu se 4 kroky, tzv. Runge-Kuttovu metodu 4. řádu (často označovanou RK4). Upozorňuji na terminologický guláš - v každém 1 kroku iterace, kterých může být třeba 5000, proběhnou 4 „kroky“ odhadu Δx. Řešíme opět rovnici

$$\frac{dx}{dt} = f(x, t).$$

Opět zvolíme časový krok Δt. Postupně spočítáme 4 odhady Δx, označené k_1 až k_4 a z jejich váženého průměru dopočítáme finální odhad Δx. K odhadu k_2 se použije znalost k_1, ke k_3 znalost

k_2 a ke k_4 znalost k_3. První krok, určení k_1, je identický s Eulerovou metodou. Existují i jiné drobné modifikace téhož postupu. Algoritmus je následující:

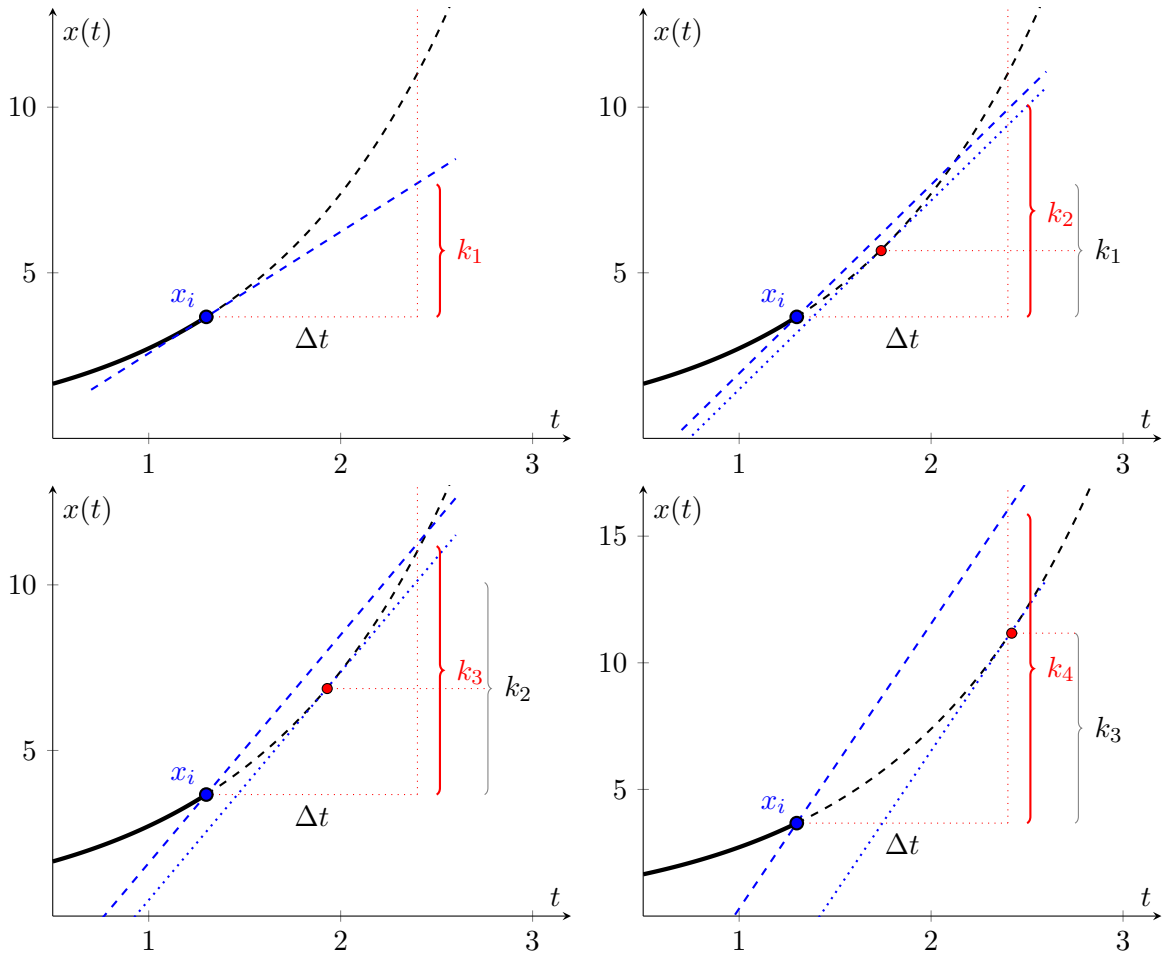
$$\begin{aligned} k_1 &= f\left(x_i, t_i\right) \Delta t \\ k_2 &= f\left(x_i + \frac{k_1}{2}, t_i + \frac{\Delta t}{2}\right) \Delta t \\ k_3 &= f\left(x_i + \frac{k_2}{2}, t_i + \frac{\Delta t}{2}\right) \Delta t \\ k_4 &= f\left(x_i + k_3, t_i + \Delta t\right) \Delta t. \end{aligned}$$

Princip algoritmu je vizualizován na obrázku 3.5. Odhad „průměrného“ Δx je

$$\Delta x := \frac{k_1 + 2k_2 + 2k_3 + k_4}{6}.$$

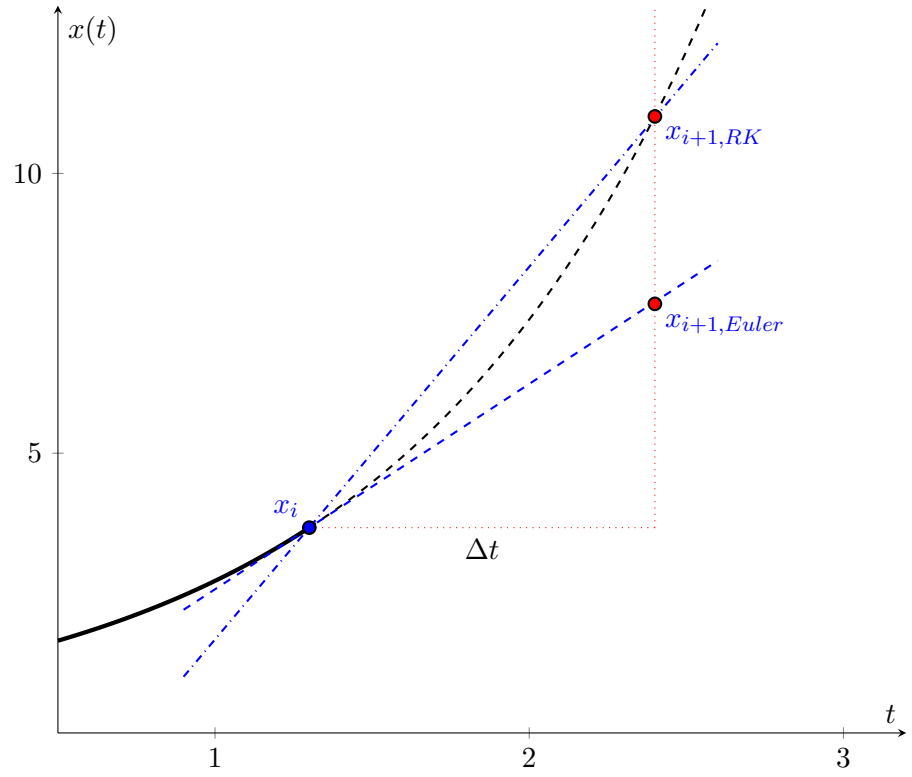
Odhady t_{i+1} a x_{i+1} definujeme jako

$$\begin{aligned} t_{i+1} &= t_i + \Delta t \\ x_{i+1} &= x_i + \Delta x. \end{aligned}$$



Obrázek 3.5.: Konstrukce k_1 až k_4 v Rungově-Kuttově metodě

Obr. 3.6 ukazuje výsledky obou metod po jednom kroku, počítáno z hodnot k_1 až k_4 z obr. 3.5.



Obrázek 3.6.: Porovnání přesnosti Eulerovy a Runge-Kuttovy metody

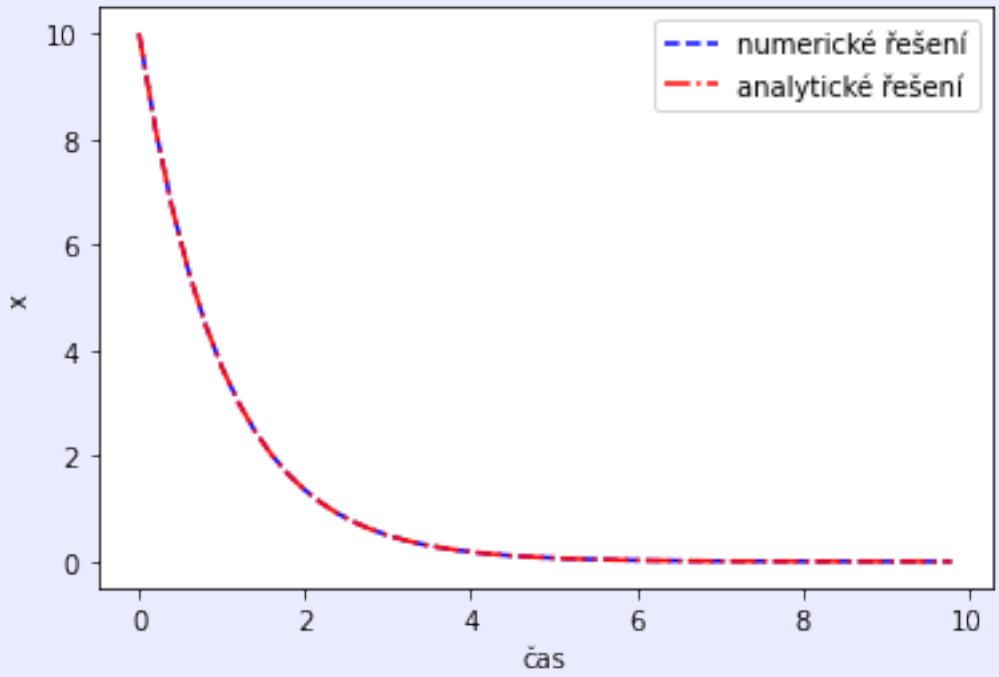
Je vidět, že RK4 metoda je výrazně přesnější než Eulerova metoda. Odhad hodnoty x_{i+1} RK4 metodou téměř přesně souhlasí s funkční hodnotou v bodě $t + \Delta t$, $x(t + \Delta)$.

Implementujme v Pythonu Runge-Kuttovu metody pro řešení stejné diferenciální rovnici $x' = -x$ jako v předchozí kapitole.

```
import numpy as np
import matplotlib.pyplot as plt

size = 50            # hodnotu funkce chceme najít v 50 bodech
t = np.linspace(0, 10, size)       # vektor časových bodů
x_num = np.linspace(0, 10, size)  # vektor numerických hodnot
x_anal = np.linspace(0, 10, size) # vektor analytických hodnot

def g(x,t=1):                      # derivace hledané funkce
    return -x

def RK(x,t=1):                     # 1 krok RK metody 4. řádu
    k1 = dt * g(x,t)
    k2 = dt * g(x+k1/2,t+dt/2)
    k3 = dt * g(x+k2/2,t+dt/2)
    k4 = dt * g(x+k3,t+dt)
    return x+(k1+2*k2+2*k3+k4)/6

dt = 10/size    # časový krok (v tomto případě 10/50 = 0.2)
```

```
i=0
x_num[i] = 10  # počáteční hodnota funkce – x_num(t0)
t[i] = 0       # počáteční čas

while i < (size-1):    # iterace
      x_num[i+1] = RK(x_num[i],t[i])
      t[i+1] = t[i] + dt
      i += 1

x_anal = 10*np.exp(-t) # analytické řešení

# graf řešení
fig, axes = plt.subplots()
axes.plot(t, x_num, 'b', label = 'numerické řešení')
axes.plot(t, x_anal, 'r', label = 'analytické řešení')
axes.set_xlabel('čas')
axes.set_ylabel('x')
axes.legend()
```

Numerické i analytické řešení je zobrazeno na následujícím obrázku. Obě křivky jsou prakticky totožné. Opět je patrno, že chyba metody RK4 je výrazně menší než metody Eulerovy, ačkoli časové kroky Δt byly zvoleny stejné.



3.3. Numerické řešení soustav diferenciálních rovnic

Poněvadž je analytické řešení soustav diferenciálních rovnic možné jen vzácně, jsou numerické metody řešení zásadní. Obě výše popsané metody se dají rozšířit pro případ soustavy diferenciálních rovnic. Uvažujme dvě funkce času, $x(t)$ a $y(t)$, které charakterizuje soustava diferenciálních

rovnic

$$\frac{dx}{dt} = f_1(x, y, t)$$
$$\frac{dy}{dt} = f_2(x, y, t)$$

s počátečními podmínkami $x(0) = x_0$, $y(0) = y_0$ a $t(0) = t_0$.

3.3.1. Eulerova metoda

Úprava Eulerovy metody pro případ soustavy rovnic je snadná. Rovnice analogicky upravíme na tvar

$$\frac{\Delta x}{\Delta t} \approx f_1(x, y, t)$$
$$\frac{\Delta y}{\Delta t} \approx f_2(x, y, t).$$

Časový krok Δt je společný pro obě rovnice. Každá z nich pak udává, o kolik se pro daný časových krok změní hodnota x a y. Iterační předpis musí znít

$$x_{i+1} := x_i + f_1(x, y, t).\Delta t$$
$$y_{i+1} := y_i + f_2(x, y, t).\Delta t$$
$$t_{i+1} := t_i + \Delta t.$$

Implementujme v Pythonu řešení soustavy diferenciálních rovnic

$$\frac{dx}{dt} = -x + 2y$$
$$\frac{dy}{dt} = -x + y$$

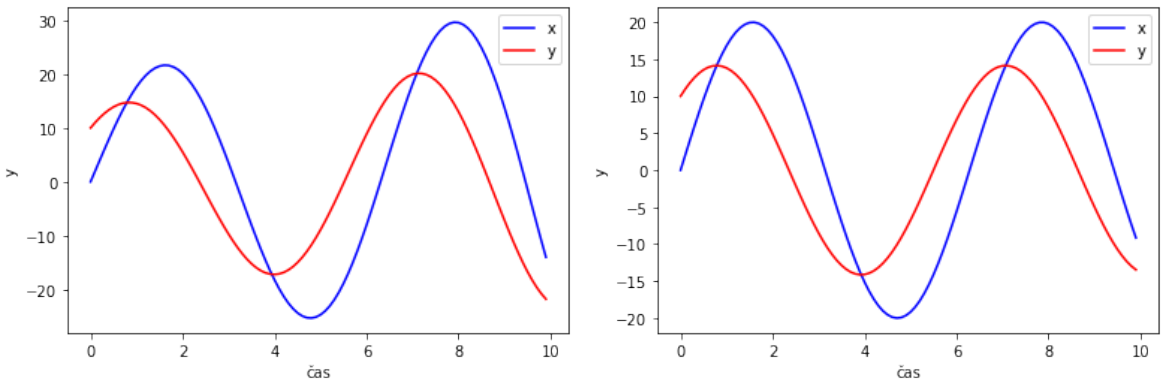
pomocí Eulerovy metody. Požadujeme $x(0) = 0$ a $y(0) = 10$. Doplňme, ač je to zřejmé, že ve vztahu k obecnému zadání platí $f_1(x, y, t) = -x + 2y$ a $f_2(x, y, t) = -x + y$.

```
import numpy as np
size = 100      # hodnotu obou funkcí chceme najít v 100 bodech
t = np.linspace(0, 50,size)  # vektor časových bodů
x = np.linspace(0, 50,size)  # vektor numerických hodnot x
y = np.linspace(0, 50,size)  # vektor numerických hodnot y

def f1(x,y,t):          # derivace hledané funkce x(t)
    return -x + 2*y

def f2(x,y,t):          # derivace hledané funkce y(t)
    return -x + y

dt = 0.1                # časový krok

i=0
x[i] = 0      # počáteční hodnota funkce x
```

```
y[i] = 10     # počáteční hodnota funkce y
t[i] = 0      # počáteční čas

while i < size-1:      # iterace
    dx = dt * f1(x[i],y[i],t[i])
    dy = dt * f2(x[i],y[i],t[i])
    x[i+1] = x[i] + dx
    y[i+1] = y[i] + dy
    t[i+1] = t[i] + dt
    i += 1

# graf řešení – vývoj v čase
axes.plot(t, x, 'b', label = 'x')
axes.plot(t, y, 'r', label = 'y')
axes.set_xlabel('čas')
axes.set_ylabel('y')
axes.set_title('vývoj v čase')
axes.legend()
```

Obr. 3.7 (a) ukazuje časový průběh řešení. Vidíme oscilace s rostoucí amplitudou.

Obr. 3.8 (a) ukazuje tzv. **fázový portrét**. Jde o vzájemné zobrazení vývoje funkcí x a y po eliminaci nezávisle proměnné, v tomto případě času. Kdybychom z výšky pozorovali objekt pohybující se po zemi, trajektorie jeho pohybu, kterou bychom viděli, je příkladem fázového portrétu. Samy polohy vůči osám, tedy $x(t)$ a $y(t)$, jsou funkcemi času. Pomocí fázového portrétu můžeme „vizualizovat chování“ soustav diferenciálních rovnice, např. hledat oscilace či stabilní a nestabilní body. Těmito (a dalšími) problémy se zabývá oblast matematické analýzy zvaná Teorie dynamických systémů.



(a) Eulerova metoda (b) Runge-Kuttova metoda

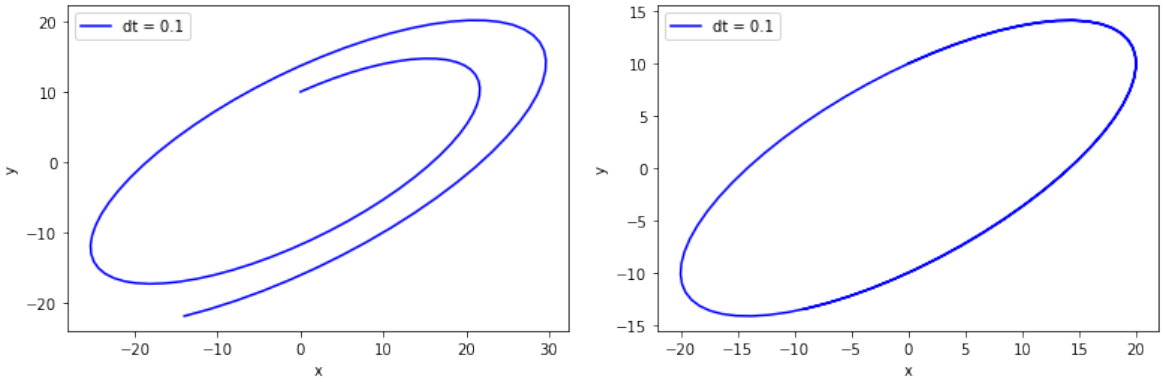
Obrázek 3.7.: Numerické řešení soustavy diferenciálních rovnic - vývoj v čase

Nakresleme fázový portrét řešení soustavy diferenciálních rovnic uvedené v předchozím příkladě.

```
#graf řešení – fázový portrét
fig, axes = plt.subplots()
```

```
axes.plot(x, y, 'b', label = 'dt = 0.1')
axes.set_xlabel('x')
axes.set_ylabel('y')
axes.set_title('fázový portrét')
axes.legend()
```

Výsledný portrét ukazuje obr. 3.8 (a). Okomentujeme jej později.



(a) Eulerova metoda

(b) Runge-Kuttova metoda

Obrázek 3.8.: Numerické řešení soustavy diferenciálních rovnic - fázový portrét

3.3.2. Runge-Kuttova metoda

Upravit RK4 metodu pro soustavy diferenciálních rovnic je složitější než v případě Eulerovy metody. V zásadě je třeba jednotlivá k_1 až k_4 počítat pro obě proměnné zároveň. k odpovídající proměnné x označme k_x, k_y budou odpovídat y. Pro výše uvedenou soustavu rovnic

$$\frac{dx}{dt} = f_1(x, y, t)$$
$$\frac{dy}{dt} = f_2(x, y, t)$$

s počátečními podmínkami $x(0) = x_0$, $y(0) = y_0$ a $t(0) = t_0$ zní algoritmus RK4 metody takto:

$$
\begin{aligned}
k_{1,x} &= f_1\left(x_i, y_i, t_i\right)\Delta t \\
k_{1,y} &= f_1\left(x_i, y_i, t_i\right)\Delta t \\
k_{2,x} &= f_1\left(x_i + \frac{k_{1,x}}{2}, y_i + \frac{k_{1,y}}{2}, t_i + \frac{\Delta t}{2}\right)\Delta t \\
k_{2,y} &= f_2\left(x_i + \frac{k_{1,x}}{2}, y_i + \frac{k_{1,y}}{2}, t_i + \frac{\Delta t}{2}\right)\Delta t \\
k_{3,x} &= f_1\left(x_i + \frac{k_{2,x}}{2}, y_i + \frac{k_{2,y}}{2}, t_i + \frac{\Delta t}{2}\right)\Delta t \\
k_{3,y} &= f_2\left(x_i + \frac{k_{2,x}}{2}, y_i + \frac{k_{2,y}}{2}, t_i + \frac{\Delta t}{2}\right)\Delta t \\
k_{4,x} &= f_1\left(x_i + k_{3,x}, y_i + k_{3,y}, t_i + \Delta t\right)\Delta t \\
k_{4,y} &= f_2\left(x_i + k_{3,x}, y_i + k_{3,y}, t_i + \Delta\right)\Delta t.
\end{aligned}
$$

První krok je opět identický s Eulerovou metodou. Odhady „průměrných“ Δx a Δy jsou

$$
\begin{aligned}
\Delta x &:= \frac{k_{1,x} + 2k_{2,x} + 2k_{3,x} + k_{4,x}}{6} \\
\Delta y &:= \frac{k_{1,y} + 2k_{2,y} + 2k_{3,y} + k_{4,y}}{6}.
\end{aligned}
$$

Iterační předpisy pro t_{i+1}, x_{i+1} a y_{i+1} v jednotlivých iteracích zní

$$
\begin{aligned}
t_{i+1} &= t_i + \Delta t \\
x_{i+1} &= x_i + \Delta x \\
y_{i+1} &= y_i + \Delta y.
\end{aligned}
$$

Implementujme v Pythonu řešení stejné soustavy diferenciálních rovnic

$$
\begin{aligned}
\frac{dx}{dt} &= -x + 2y \\
\frac{dy}{dt} &= -x + y
\end{aligned}
$$

pomocí Runge-Kuttovy metody RK4. Požadujme opět $x(0) = 0$ a $y(0) = 10$.

```
import numpy as np
size = 100      # hodnotu obou funkcí chceme najít v 100 bodech
t = np.linspace(0, 50,size)  # vektor časových bodů
x = np.linspace(0, 50,size)  # vektor numerických hodnot x
y = np.linspace(0, 50,size)  # vektor numerických hodnot x

def fx(x,y,t):       # derivace hledané funkce x(t)
    return -x + 2*y

def fy(x,y,t):       # derivace hledané funkce y(t)
```

```
    return -x + y

def RK(x,y,t=1):      # 4-kroková kalkulace k1 až k4
    k1x = dt * fx(x,y,t)
    k1y = dt * fy(x,y,t)
    k2x = dt * fx(x+k1x/2,y+k1y/2,t+dt/2)
    k2y = dt * fy(x+k1x/2,y+k1y/2,t+dt/2)
    k3x = dt * fx(x+k2x/2,y+k2y/2,t+dt/2)
    k3y = dt * fy(x+k2x/2,y+k2y/2,t+dt/2)
    k4x = dt * fx(x+k3x,y+k3y,t+dt)
    k4y = dt * fy(x+k3x,y+k3y,t+dt)
    return (x+(k1x+2*k2x+2*k3x+k4x)/6, y+(k1y+2*k2y+2*k3y+k4y)/6)

dt = 0.1      # časový krok

i=0
x[i] = 0              # počáteční hodnota funkce x
y[i] = 10             # počáteční hodnota funkce y
t[i] = 0              # počáteční čas

while i < size-1:     # iterace
    a,b = RK(x[i],y[i], t[i])
    x[i+1] = a
    y[i+1] = b
    t[i+1] = t[i] + dt
    i += 1
```

Obrázky 3.7 (b) a 3.8 (b) ukazují časový průběh obou funkcí a fázový portrét při řešení RK4 metodou. Fázový portrét se podstatně liší od portrétu téže soustavy rovnic řešené Eulerovou metodou. Jde o stacionární, stále stejné cyklus oproti rozbíhavému cyklu. Stejně tak vidíme konstantní amplitudy oscilací oproti rostoucí amplitudě Eulerovou metodou (obr. 3.7 (a) a 3.8 (a)).
Prozraďme, že analytické řešení soustavy s danými počátečními podmínkami je

$$x = 20 \sin t$$
$$y = 10\sqrt{2} \sin\left(t + \pi/2\right).$$

Vidíme, že x a y jsou fázově posunuty, mají však konstantní amplitudu a stejnou úhlovou frekvenci. Ve fázovém diagramu proto tvoří uzavřenou elipsu. Správné řešení tedy plyne z Runge-Kuttovy metody, což opět dokumentuje její přesnost oproti Eulerově metodě (při stejném časovém kroku Δt). Rozbíhání v Eulerově metodě je dáno postupnou kumulací chyb. Lze proto doporučit vždy upřednostnit RK metodu před Eulerovou metodou.

4. Poznámky k lineární algebře

Matematika bývá označována jako strukturální věda, čímž se míní, že „hledá“ strukturu či zákonitosti v jistých vztazích. Matematika však není přírodní věda, toto hledání má jinou povahu než hledání v přírodních vědách. Přírodní vědy hledají zákonitosti v něčem, co je, co existuje, v přírodním světě. Matematika často postupuje tak, že extrahuje zákonitosti z intuitivně známých skutečností, např. z počítání s přirozenými čísly, a zkoumá, jaké jsou obecné důsledky těchto abstraktních zákonitostí.

Lineární algebra je úchvatnou oblastí matematiky, často první výrazně abstrahující disciplínou, s níž se student matematiky setká. Buduje řadu základních představ a pojmů. Na její poznatky navazují další oblasti matematiky i fyziky, např. teorie diferenciálních rovnic či kvantová mechanika. Ilustrujme si zmíněný „abstrahující princip“ na příkladu.

Dítě se v první třídě učí sčítat. 2 hrušky + 3 hrušky = 5 hrušek, 2 auta + 3 auta = 5 aut, 2 domy + 3 domy = 5 domů. Postupně zjistí, že všechny tyto rovnosti mají něco společného, totiž, že $2 + 3 = 5$. Je to první stupeň abstrakce. Můžeme počítat s čísly, nepotřebujeme konkrétní předměty.
Mnohem později abstrakce pokračuje. $2.(3+4) = 2.3 + 2.4$, $4.(1+5) = 4.1 + 4.5$. Všechna čísla ve vztazích můžeme nahradit proměnnými a, b, c, zastupujícími libovolné číslo, ale početní zákonitost, zde označovaná jako distribuční zákon, tedy $a.(b + c) = a.b + a.c$, stále platí. Nepotřebujeme tedy přímo čísla, stačí nám proměnné, které se řídí stejnými pravidly. Mají v jistém smyslu „stejnou strukturu“. Lineární algebra pokračuje touto cestou dále. Dalším stupněm abstrakce může být pojem vektoru.

4.1. Vektory a vektorové prostory

Ukažme si, jak by cestou abstrakce matematika dospěla k pojmu **vektor**. Klasickou středoškolskou představou vektoru je „šipka mířící odněkud někam“. S taková „šipkou“ můžeme provádět některé „operace“, které mají charakteristické vlastnosti (viz obr. 4.1).

1. Násobení šipky číslem šipku prodlouží, zkrátí, nebo obrátí, ale šipka stále zůstává šipkou.

2. Pokud vynásobíme šipku číslem 1, nezmění se.

3. 2 šipky sečteme tak, že přiložíme druhou na konec první. Můžeme však též přiložit první na konec druhé. Výsledná šipka, resp. bod, kam šipky společně dosáhnou, je totožná.

4. „Nulovou“ šipku, tedy šipku délky O, můžeme přičíst k jakékoli šipce, aniž by se tím změnila.

Právě jsme jmenovali 2 operace, **sčítaní** šipek a **násobení** šipek reálným číslem (konstantou),

(a) Násobení vektoru konstantou (b) Sčítání vektorů

Obrázek 4.1.: Operace s vektory

a popsali některé jejich vlastnosti. Dalšími typickými vlastnostmi těchto operací jsou jejich vzájemné „interakce“, označované jako komutativita, distributivita a asociativita. Zapišme popsané vlastnosti formálně, matematicky. Pojďme nyní místo slova šipka používat slovo vektor, berme je zatím jako synonyma. Vektor označme $\vec{v}$, případně $\vec{v_1}$, $\vec{v_2}$ nebo $\vec{v_3}$. Reálné číslo, jímž násobíme, bude a.

Máme tedy 2 operace - násobení vektoru konstantou a sčítání vektorů. Obě operace jsou uzavřené, což znamená, že výsledkem je opět vektor, nikoli třeba reálné číslo.

$$\vec{v_2} = a.\vec{v_1} \qquad \text{násobení vektoru konstantou}$$
$$\vec{v} = \vec{v_1} + \vec{v_2} \qquad \text{sčítání vektorů}$$

Operace splňují následujících 8 vlastností, jak se můžeme na chování šipek sami snadno přesvědčit:

$$\vec{v} = \vec{0} + \vec{v} \qquad \text{existuje neutrální prvek sčítání}$$
$$\vec{v} + (-\vec{v}) = \vec{0} \qquad \text{existuje inverzní prvku ke každému prvku}$$
$$\vec{v_1} + \vec{v_2} = \vec{v_2} + \vec{v_1} \qquad \text{komutativita sčítání}$$
$$\vec{v_1} + (\vec{v_2} + \vec{v_3}) = (\vec{v_1} + (\vec{v_2}) + \vec{v_3} \qquad \text{asociativita sčítání}$$
$$\vec{v} = 1.\vec{v} \qquad \text{existuje neutrální prvek násobení}$$
$$a.(\vec{v_1} + \vec{v_2}) = (a.\vec{v_1}) + (a.\vec{v_2}) \qquad \text{distributivita}$$
$$(a+b).\vec{v} = a.\vec{v} + b.\vec{v} \qquad \text{distributivita}$$
$$(a.b).\vec{v} = a.(b.\vec{v}) \qquad \text{asociativita násobení}$$

Dosud jsme pouze „zkomplikovali“ banální představy o šipkách. Abstraktní krok je následující. Změňme od této chvíle význam slova vektor.

Vektor již nebude znamenat pouze šipku, nýbrž jakýkoli „objekt“, na němž lze definovat nějaké 2 operace, které označíme jako sčítání a násobení číslem, které mají všech 8 uvedených vlastností.

Budeme možná překvapeni, že takových objektů je řada. Krom šipek jsou jimi například n-tice čísel, polynomy kteréhokoli stupně, funkce, které jsou řešením homogenní diferenciální rovnice, matice konkrétního typu a další. Matematika se může dále zabývat pouze studiem 8 uvedených vlastností, aniž by ji zajímalo, který konkrétní objekt je „v pozadí“. Zjištěné výsledky pak budou automaticky platit pro všechny konkrétní případy vektorů. Šipky jsou tedy konkrétní, ale nikoli jediný příklad vektorů.

Zaveďme ještě důležitý pojem **vektorový prostor**, též označovaný jako lineární prostor. Poznamenejme jen, že slovo prostor je v matematice (přinejmenším zde) pouhé synonymum ke slovu množina, nemá jiný speciální význam.

Mějme nějakou množinu V, na jejíž prvcích nějak libovolně definujeme operaci násobení reálným číslem „.“ a operaci sčítání 2 prvků „+“, přičemž ale definované operace splňují 8 výše uvedených podmínek a jsou uzavřené. Trojici $(V, ., +)$ označíme jako **vektorový prostor**. Každý prvek vektorového prostoru nazveme **vektor**.

Množina všech šipek s dvěma výše definovanými operacemi tedy zjevně tvoří vektorový prostor.

Ukažme si v následujícím příkladu, že i polynomy 3. stupně můžeme považovat za vektory, prvky vektorového prostoru. Vezměme nějaké dva (obecné) polynomy 3. stupně

$$\begin{aligned} y_1 &= a_1x^3 + b_1x^2 + c_1x + d_1 \\ y_2 &= a_2x^3 + b_2x^2 + c_2x + d_2 \end{aligned}$$

Operaci násobení číslem a sčítání budeme chápat „klasicky“, jak jsme zvyklí. Násobme polynom y_1 konstantou $k \in \mathbb{R}$. Pak dostaneme

$$y = ky_1 = (ka_1)x^3 + (kb_1)x^2 + (kc_1)x + (kd_1),$$

což je zjevně opět polynom 3. stupně. Operace násobení reálným čísle je tedy uzavřená. Nyní sečteme polynomy y_1 a y_2.

$$y_1 + y_2 = (a_1 + a_2)x^3 + (b_1 + b_2)x^2 + (c_1 + c_2)x + (d_1 + d_2)$$

Výsledkem je opět polynom 3. stupně. Operace je též uzavřená. Velmi jednoduše se dá dále ukázat, že tyto dvě operace splňují všech 8 výše uvedených požadavků. Jako příklady ukažme komutativitu a existenci neutrálního prvku sčítání. Pro sčítání platí komutativita:

$$\begin{aligned} y_1 + y_2 &= (a_1 + a_2)x^3 + (b_1 + b_2)x^2 + (c_1 + c_2)x + (d_1 + d_2) \\ &= (a_2 + a_1)x^3 + (b_2 + b_1)x^2 + (c_2 + c_1)x + (d_2 + d_1) = y_2 + y_1. \end{aligned}$$

Neutrálním prvkem sčítání je $\vec{0} := 0.x^3 + 0.x^2 + 0.x + 0$. Pro součet s neutrálním prvkem platí

$$\begin{aligned} y + \vec{0} &= ax^3 + bx^2 + cx + d + 0x^3 + 0x^2 + 0x + 0 \\ &= (a + 0)x^3 + (b + 0)x^2 + (c + 0)x + (d + 0) = y. \end{aligned}$$

Polynom $\vec{0}$ zjevně splňuje požadovanou vlastnost. Neutrální prvek sčítání tedy existuje. Takto bychom mohli dále ukázat platnost všech 8 vlastností.
Polynomy 3. stupně tedy můžeme považovat za vektory. Množina všech polynomů 3. stupně společně s operacemi násobení reálným číslem a sčítání tvoří vektorový prostor.

V dalším textu zúžíme naši představu vektorů pouze na **n-tice čísel**. Rozlišujeme **řádkové vektory**

$$\vec{v} = \begin{pmatrix} 1, & 2, & 3 \end{pmatrix}$$

a **sloupcové vektory**

$$\vec{v} = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}.$$

Záměna řádků a sloupců se označuje jako **transpozice**, značí se T. Tedy

$$\begin{pmatrix} 1, & 2, & 3 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}^T.$$

Jednotlivá čísla uvnitř vektoru se označují **složky** nebo komponenty vektoru, označujeme je indexy, např. $v_2 = 2$. i-tá složka je v_i.

4.1.1. Operace s vektory

Sčítání vektorů a **násobení číslem** probíhá „po složkách“, angl. element-wise.

$$\begin{pmatrix} 5 \\ 7 \\ 9 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} + \begin{pmatrix} 4 \\ 5 \\ 6 \end{pmatrix}.$$

$$\begin{pmatrix} -3 \\ -6 \\ -9 \end{pmatrix} = -3. \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}.$$

Dále definujeme **skalární součin** 2 vektorů jako „pronásobení“ odpovídajících složek a součet výsledků

$$\begin{pmatrix} x_1, x_2, x_3 \end{pmatrix} . \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = x_1 y_1 + x_2 y_2 + x_3 y_3.$$

Elegantně můžeme skalární součin zapsat jako

$$\vec{u}.\vec{v} = \sum_{i=1}^{n} u_i v_i.$$

$$(1, 2, 3).(4, 5, 6)^T = 1.4 + 2.5 + 3.6 = 29$$

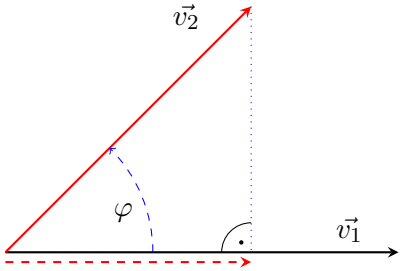
V kvantové mechanice se skalární součin vektorů $\vec{u}$ a $\vec{v}$ obvykle zapisuje pomocí tzv. Diracovy braketové notace jako $\langle \vec{u} | \vec{v} \rangle$.

Délka šipky $\vec{\text{š}}$, jejíž složky v osách by byly š_x, š_y a š_z, by dle Pythagorovy věty přirozeně byla

$$\text{š} = \left\| \vec{\text{š}} \right\| = \sqrt{\text{š}_x^2 + \text{š}_y^2 + \text{š}_z^2}.$$

Analogicky proto definujeme **velikost** neboli **normu** vektoru jako

$$v = \|\vec{v}\| = \sqrt{v_1^2 + v_2^2 + \ldots + v_n^2}.$$



Obrázek 4.2.: Skalární součin vektorů - geometrická interpretace

Pomocí skalárního součinu můžeme normu vektoru elegantně zapsat jako

$$v = \sqrt{\vec{v}.\vec{v}}.$$

Norma 6-rozměrného vektoru $\vec{v} = (1, 2, 3, 4, 5, 6)$ je

$$v = \sqrt{1^2 + 2^2 + 3^2 + 4^2 + 5^2 + 6^2} \approx 9,54.$$

Geometricky představuje skalární součin kolmou projekci jednoho vektoru do druhého (a následobenou druhým vektorem), viz obr 4.2. Pokud oba vektory svírají úhel φ, lze skalární součin spočítat jako

$$\vec{u}.\vec{v} = \|\vec{u}\| . \|\vec{v}\| . \cos\varphi.$$

Skalární součin dvou kolmých vektorů je 0. Například vektory $(0, 1)$ a $(1, 0)$ jsou zjevně kolmé, „leží v ose x a y“, a platí $0.1 + 1.0 = 0$. Podobně vektory $(3, 5)$ a $(5, -3)$ jsou kolmé a platí $3.5 + 5.(-3) = 0$.

Ve fyzice se skalární součin vyskytuje často. Je jím například definována práce jako skalární součin vektoru síly a vektoru dráhy - $W = \vec{F}.\vec{s}$. Pokud síla působí kolmo na dráhu, žádnou práci nekoná.

Jako rozšíření doplňme následující abstraktní poznámku. Pokud u vektorů, tedy prvků vektorového prostoru, umíme nějak definovat normu, pak kombinaci vektorového prostoru s normou označujeme jako **normovaný vektorový prostor**. A pokud u vektorů umíme definovat normu i skalární součin, pak takový vektorový prostor označujeme jako **Hilbertův prostor**. Hilbertův prostor je základní matematická struktura, v níž se „odehrává“ kvantová mechanika.

A ještě zmíníme jednu abstraktní drobnost závěrem. Pokud umíme definovat normu a skalární součin, pak můžeme výše uvedený vztah obrátit a naopak definovat abstraktní úhel mezi dvěma vektory jako

$$\cos\varphi := \frac{\vec{u}.\vec{v}}{\|\vec{u}\| . \|\vec{v}\|}.$$

Můžeme tak definovat úhel, který svírají dvě matice. Jde opět o velkou abstrakci, žádný reálný úhel matice samozřejmě nesvírají.

4.2. Matice

Maticí rozumíme seskupení několika čísel do obdélníkového nebo čtvercového tvaru. Matici obvykle uzavíráme do kulatých nebo hranatých závorek. Rozlišujeme řádky a sloupce matice. Např.

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

je matice nazvaná A s 3 řádky a 4 sloupci. Hovoříme o matici typu 3×4. Obecně má matice typu $m \times n$ m řádků a n sloupců. Matice typu $m \times m$ se označuje jako čtvercová.

Jednotlivá čísla označujeme jako prvky či elementy matice. Prvek v druhém řádku a třetím sloupci značíme a_{23}, pro uvedenou matici je $a_{23} = 8$. Obecně je prvek v i-tém řádku a j-tém sloupci a_{ij}. Prvek levý horní je a_{11}, pravý horní a_{1n}, levý dolní a_{m1} a prvek pravý dolní je a_{mn}.

$$\begin{pmatrix} a_{11} & \dots & a_{1n} \\ \vdots & a_{ij} & \vdots \\ a_{m1} & \dots & a_{mn} \end{pmatrix}$$

Jednolivé řádky matice můžeme považovat za řádkové vektory, jednotlivé sloupce za sloupcové vektory.

4.2.1. Typy matic

Kromě obecné matice výše uvedené, kdy je většina čísel v matici odlišných od 0 a není patrna žádná struktura uspořádání prvků, existují speciální typy matic.

Diagonální matice má prvky jen na hlavní diagonále.

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 7 \end{pmatrix}$$

Jednotková matice je diagonální matice se všemi prvky rovnými 1.

$$E_4 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Symetrická matice je symetrická vůči hlavní diagonále.

$$\begin{pmatrix} 1 & -1 & 7 & -3 \\ -1 & 3 & 5 & 4 \\ 7 & 5 & 2 & 0 \\ -3 & 4 & 0 & 7 \end{pmatrix}$$

Horní a **dolní trojúhelníková** matice mají pouze nuly pod nebo nad hlavní diagonálou.

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

Transponovaná matice má vyměněné řádky a sloupce. Transponovaná matice A^T k výše uvedené matici A je typu 4×3 a zní

$$A^T = \begin{pmatrix} 1 & 5 & 6 \\ 2 & 5 & 9 \\ 3 & 8 & 1 \\ 3 & 7 & 0 \end{pmatrix}.$$

Pro symetrickou matici zjevně platí $A^T = A$.

4.2.2. Operace s maticemi

Sčítání a odčítání matic **stejného typu** je prosté sčítání a odčítání po prvcích.

$$\begin{pmatrix} 1 & 3 & -5 \\ -2 & 1 & 4 \\ 3 & 4 & -3 \end{pmatrix} + \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} = \begin{pmatrix} 2 & 5 & -2 \\ 2 & 6 & 10 \\ 10 & 12 & 6 \end{pmatrix}.$$

Totéž platí pro **násobení matic číslem**.

$$3. \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} = \begin{pmatrix} 3 & 6 & 9 \\ 12 & 15 & 18 \\ 21 & 24 & 27 \end{pmatrix}.$$

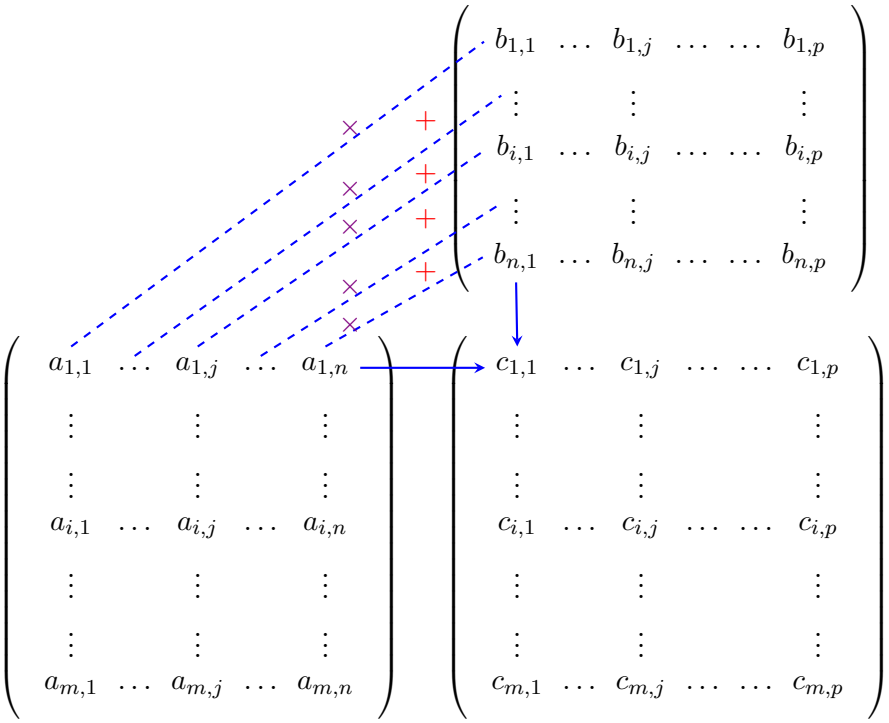
Operace splňují všech 8 výše popsaných vlastností. I množina všech matic stejného typu proto tvoří vektorový prostor. Chovají se totiž „jako šipky“.

Násobení vektoru maticí je složitější. Matici si představíme jako m řádkových vektorů napsaných pod sebou a provedeme skalární součin každého řádku s násobeným vektorem. Postupujeme systémem „řádek-krát-sloupec“.

$$\begin{pmatrix} 1 & 3 & -5 & 1 \\ -2 & 1 & 4 & 1 \\ 3 & 4 & -3 & 1 \end{pmatrix} . \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} = \begin{pmatrix} 1.1 + 3.2 + (-5).3 + 1.1 \\ (-2).1 + 1.2 + 4.3 + 1.2 \\ 3.1 + 4.2 + (-3).3 + 1.3 \end{pmatrix} = \begin{pmatrix} -7 \\ 14 \\ 5 \end{pmatrix}.$$

Výsledky vektor má tolik prvků, jako má matice řádků. Násobený vektor musí mít tolik prvků, jako matice sloupců. Pokud je matice A typu $m \times n$, v násobení $\vec{y} = A.\vec{x}$ má $\vec{x}$ n prvků a $\vec{y}$ m prvků.

Nejkomplikovanější je **násobení matice maticí**, $C = A.B$. První matici, A, typu $m \times n$ si



Obrázek 4.3.: Schéma násobení matic

představíme jako složenou z m řádkových vektorů o n komponentách, druhou matici B typu $n \times p$ jako složenou z p sloupcových vektorů délky n. Provedeme skalární součin každého řádkového vektoru matice A s každým sloupcovým vektorem matice B. Pro to je nutné, aby byl „vnitřní“ rozměr obou matic, v tomto případě n, stejný, tedy počet sloupců první matice musí být stejný jako počet řádků druhé matice. Výsledná matice C je typu $m \times p$. Obecně pro prvek s indexy i, j matice C platí

$$C_{ij} = \sum_{k=1}^{n} A_{ik}.B_{kj}.$$

Obr. 4.3 názorně ukazuje postup výpočtu.

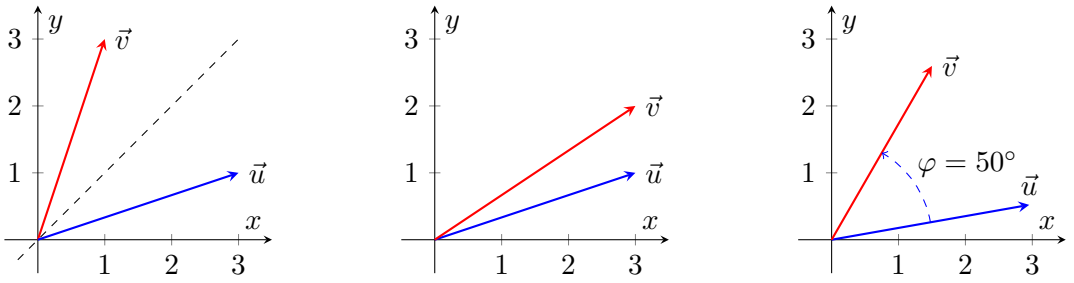
Násobení matic

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

$$B = \begin{pmatrix} 1 & 6 \\ -2 & 9 \\ 3 & -1 \end{pmatrix}$$

$$C = A.B = \begin{pmatrix} 1.1 + 5.(-2) + 6.3 & 1.6 + 5.9 + 6.(-1) \\ 2.1 + 5.(-2) + 9.3 & 2.6 + 5.9 + 9.(-1) \\ 3.1 + 8.(-2) + 1.3 & 3.6 + 8.9 + 1.(-1) \\ 3.1 + 7.(-2) + 0.3 & 3.6 + 7.9 + 0.(-1) \end{pmatrix} = \begin{pmatrix} 9 & 45 \\ 19 & 48 \\ -10 & 89 \\ -11 & 81 \end{pmatrix}.$$

Matice A je typu 4×3, matice B typu 3×2 a matice C typu 4×2.



(a) Zrcadlení podle osy $y = x$ (b) Vertikální roztažení (c) Rotace o 50°

Obrázek 4.4.: Geometrická interpretace násobení vektoru maticí

4.2.3. Geometrická interpretace násobení vektoru maticí

Mějme jako příklad dvojrozměrný vektor v rovině $\vec{u} = (u_1, u_2)$ a vynásobme jej čtvercovou maticí typu 2×2. Ukažme si několik příkladů, abychom „viděli“, co provede matice s vektorem (obr. 4.4).

Násobením maticí

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

dostaneme nový vektor $\vec{v} = A.u = (u_2, u_1)$. Došlo k výměně složek vektoru, což odpovídá zrcadlení vektoru podle osy $y = x$. Matice

$$A = \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix}$$

prodlouží „y-vou“ složku vektoru dvakrát a nezmění „x-vou složku“, provede tedy vertikální roztažení. Matice

$$A = \begin{pmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{pmatrix}$$

způsobí rotaci vektoru o úhel φ proti směru hodinových ručiček, ale nezmění velikost vektoru.

Různé matice tedy působí různé, ale pro danou matici charakteristické změny vektoru. Pokud vektor násobíme čtvercovou maticí, výsledný vektor má stejnou dimenzi. Je možné násobit vektor i maticí obdélníkovou. Pak klesá nebo roste dimenze vektoru a geometrická interpretace už není snadná. Mohlo by jít např. o projekci 3D-vektoru do nějaké roviny (2D-vektoru).

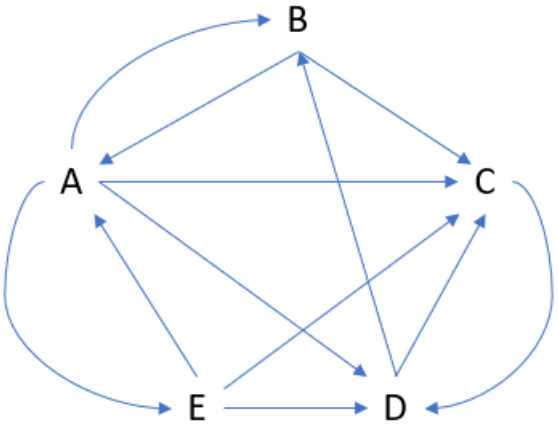
4.2.4. Příklady aplikací matic

Matice a lineární algebra obecně mají množství aplikací ve různých vědních oblastech. Ukážeme si čistě pro ilustraci dva příklady.

Reprezentace vztahů

Mějme 5 lidí, A až E. Schéma 4.5 (a) vyjadřuje vztahy mezi nimi. Šipka $X \to Y$ znamená „X má rád Y“. Některé vztahy jsou oboustranné, některé pouze jednostranné. Tyto vztahy můžeme vyjádřit pomocí matice I, někdy označované jako incidenční (4.5 (b)). Řádky znamenají „kdo má rád“, sloupce znamenají „koho má rád“. 1 a 0 značí „má rád“ a „nemá rád“. Například 1 v

2. řádku 1. sloupce znamená „B má rád A“, 0 ve 4. řádku 5. sloupce znamená „D nemá rád E“. Takto můžeme snadno zachytit vztahy libovolného množství osob a dále je například počítačově analyzovat.



(a) Vztahy mezi lidmi

$$I = \left(\begin{array}{c|ccccc} & A & B & C & D & E \\ \hline A & 0 & 1 & 1 & 1 & 1 \\ B & 1 & 0 & 1 & 0 & 0 \\ C & 0 & 0 & 0 & 1 & 0 \\ D & 0 & 1 & 1 & 0 & 0 \\ E & 1 & 0 & 1 & 1 & 0 \end{array}\right)$$

(b) Incidenční matice

Obrázek 4.5.: Vztahy mezi lidmi vyjádřené maticí

Co můžeme z matice zjistit? Matice je čtvercová, protože popisuje vztaky každého z každým. Nuly na hlavní diagonále znamenají, že nikdo nemá rád sám sebe. Ani jeden řádek a ani jeden sloupec neobsahují samé nuly. Není tedy nikdo, kdo by neměl nikoho rád, nebo někdo, koho by neměl nikdo rád.

Mohlo by nás zajímat, zda existují dvojice, které se mají rády navzájem. Pokud se symetricky oproti hlavní diagonále nacházejí jedničky, znamená to, že se tato dvoji lidí má navzájem ráda. Pokud matici I transponujeme a transponovanou matici I^T „položíme na původní matici“, pak tyto dvojice musí mít „na stejném místě jedničku“. U ostatních dvojic se čísla liší. Výjimkou jsou obě nuly, kdy se dvojice oboustranně nemají rády.

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

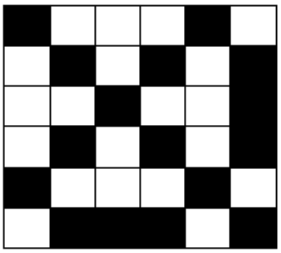
Pokud nyní obě matice odečteme, získáme novou matici Vzt, kde jsou na místech stejných vzájemných vztahů, ať pozitivních či negativních, nuly. Z matice hned vidíme (modré nuly), že dvojice se vzájemnými sympatiemi jsou (A,B), (A,D) a (C,D). Navržený postup má samozřejmě vadu na kráse v tom, že ukazuje 0 i při vzájemné antipatii (v matice je dvojice (B,E) vyznačena červeně).

$$Vzt = I - I^T = \begin{pmatrix} 0 & \mathbf{0} & 1 & 1 & \mathbf{0} \\ \mathbf{0} & 0 & 1 & -1 & \mathbf{0} \\ -1 & -1 & 0 & \mathbf{0} & -1 \\ -1 & 1 & \mathbf{0} & 0 & -1 \\ \mathbf{0} & \mathbf{0} & 1 & 1 & 0 \end{pmatrix}$$

Pro zajímavost dodejme, že matice rozdílů Vzt je antisymetrická, tedy až na znaméko symetrická vůči hlavní diagonále. Proto nám stačí „hledat nuly“ pouze nad hlavní diagonálou.

Analýza obrazu

Rastrový černobílý obraz můžeme vnímat jako síť černých a bílých čtverců. Pokud 0 bude bílá a 1 černá, můžeme obraz vyjádřit jako matici (obr. 4.6) a matematicky s ním dále pracovat. Pokud matici například vydělíme dvěma, obraz „zešedne“. Digitální analýza obrazu je rozsáhlé téma aplikované matematiky a informatiky, kde právě lineární algebra hraje významnou úlohu.



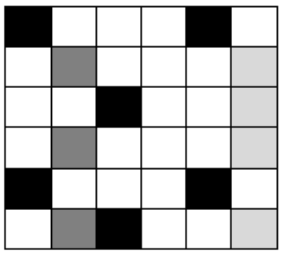
(a) Černobílý obraz

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

(b) Odpovídající matice

Obrázek 4.6.: Obraz jako matice

Ukažme si příklad matematické operace s maticí černobílého obrazu. Zkusme vynásobit matici původního čenobílého obrazu diagonální maticí s různými čísly od 0 do 1. Protože násobíme diagonální matici, bude vždy celý sloupec původní matice vynásoben stejným číslem. To znamená že „zešedne či zbělá“ celý sloupec stejně, jak můžemme vidět na obrázku 4.7. ObrKdyby nás třeba zajímala pouze část obrazu a zbytek ne, mohli bychom takto část obrazu „vymazat“.



(a) Šedý obraz

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.8 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.4 \end{pmatrix}$$

(b) Násobení diagonální maticí

Obrázek 4.7.: Změna obrazu jako násobení matic

4.3. Soustavy lineárních algebraických rovnic

Soustavou m lineárních algebraických rovnic o n neznámých se rozumí celek m lineárních vzájemně provázaných rovnic, v nichž se dohromady vyskytuje n neznámých. Soustavou 3 rovnic o 3 neznámých x_1, x_2, a x_3 je například

$$\begin{aligned} 2x_1 + x_2 - x_3 &= 1 \\ -x_1 - 3x_2 + 2x_3 &= 4 \\ 3x_1 - x_3 &= -2 \end{aligned}$$

Řešením této soustavy je trojice čísel x_1, x_2, a x_3, které můžeme považovat za složky vektoru $\vec{x} = (x_1, x_2, x_3)$. Na pravé straně soustavy, za „=“, stojí též 3 čísla, která můžeme považovat za složky nějakého vektoru, např. $\vec{b} = (1, 4, -2)$. Kdybychom z napsané soustavy „vypustili“

proměnné x_i, zbyla by nám struktura matice. Seskupme tedy koeficienty u proměnných do matice A.

$$A = \begin{pmatrix} 2 & 1 & -1 \\ -1 & -3 & 2 \\ 3 & 0 & -1 \end{pmatrix}$$

Pak ale můžeme celou soustavu rovnic ihned přepsat do maticové rovnice

$$A\vec{x} = \vec{b}.$$

Řešit soustavu rovnic tedy vlastně znamená hledat vektor $\vec{x}$, který vyhovuje dané maticové rovnici. Geometricky nahlíženo hledáme vektor $\vec{x}$, který matice A přetváří ve vektor $\vec{b}$. Tím jsme sice dospěli k novému „pohledu“ na řešení soustavy rovnic, ale zatím nám to jakkoli nepomohlo při vlastním řešení.

4.3.1. Gaussova eliminační metoda

Řešit soustavu rovnic lze například tak, že postupně vyjadřujeme jednu z proměnných a dosazujeme do ostatních rovnic. Postup je to jednoduchý, ale pracný a nesystematický. Představíme si systematickou metodu, kterou lze např. snadno implementovat v podobě počítačového programu, označovanou jako Gaussova eliminační metoda. Metodu si vysvětlíme na příkladu výše uvedené soustavy rovnic. Soustavu si přepíšeme do matice rozšířené o pravou stranu rovnic, kterou si pro přehlednost oddělíme svislou čarou, tedy

$$\left(\begin{array}{ccc|c} 2 & 1 & -1 & 1 \\ -1 & -3 & 2 & 4 \\ 3 & 0 & -1 & -2 \end{array}\right).$$

Principem metody je převedení matice na horní trojúhelníkový tvar s pouhými nulami pod hlavní diagonálou, tedy

$$\left(\begin{array}{ccc|c} \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \end{array}\right).$$

Z trojúhelníkové matice už snadno tzv. zpětným chodem zjistím řešení, jak hned uvidíme. Uvědomme si znovu, že každý řádek matice je pouze některá „přepsaná“ rovnice. Pokud bychom původní rovnici vynásobili nenulovým číslem, dostaneme rovnici novou, avšak se stejným řešením. Podobně můžeme např. ke 3. rovnici přičíst rovnici 1., nebo můžeme zaměnit pořadí rovnic. Řešení soustavy se též nezmění. Stejné operace můžeme proto provádět i s maticí - násobit řádek číslem, přičítat násobený řádek k jinému řádku a vyměňovat řádky. Pokud násobíme a přičítáme chytře, „vznikají“ na potřebných místech, pod hlavní diagonálou, nuly. Vyřešme tedy výše uvedenou soustavu. Nejprve vyměníme 1. a 2. řádek. Potom k 2. řádku přičteme 2-násobek řádku 1. a ke 3. řádku 3-násobek řádku 1. Nově vzniklé matice jsou stran řešení ekvivalentní s maticí původní, což vyznačíme symbolem „$\sim$“.

$$\left(\begin{array}{ccc|c} 2 & 1 & -1 & 1 \\ -1 & -3 & 2 & 4 \\ 3 & 0 & -1 & -2 \end{array}\right) \sim \left(\begin{array}{ccc|c} -1 & -3 & 2 & 4 \\ 2 & 1 & -1 & 1 \\ 3 & 0 & -1 & -2 \end{array}\right) \sim \left(\begin{array}{ccc|c} -1 & -3 & 2 & 4 \\ 0 & -5 & 3 & 9 \\ 0 & -9 & 5 & 10 \end{array}\right)$$

Nyní můžeme 2. řádek vynásobit $-\frac{9}{5}$ a přičíst k 2. Poté ještě pro přehlednost vynásobme 3. řádek -5-ti.

$$\left(\begin{array}{ccc|c} -1 & -3 & 2 & 4 \\ 0 & -5 & 3 & 9 \\ 0 & -9 & 5 & 10 \end{array}\right) \sim \left(\begin{array}{ccc|c} -1 & -3 & 2 & 4 \\ 0 & -5 & 3 & 9 \\ 0 & 0 & 5-\frac{27}{5} & 10-\frac{81}{5} \end{array}\right) \sim \left(\begin{array}{ccc|c} -1 & -3 & 2 & 4 \\ 0 & -5 & 3 & 9 \\ 0 & 0 & 2 & 31 \end{array}\right)$$

Dostali jsme tak ekvivalentní matici v dolním trojúhelníkovém tvaru. Přepišme ji pro čistě pro srozumitelnost výkladu znovu do tvaru 3 rovnic, ačkoli tento krok není nutný.

$$\begin{aligned} -x_1 - 3x_2 + 2x_3 &= 4 \\ -5x_2 + 3x_3 &= 9 \\ 2x_3 &= 31 \end{aligned}$$

Nyní už přímo získáme řešení „zezdola nahoru“. Z třetí rovnice plyne, že $x_3 = 15,5$. Dosazení x_3 do druhé rovnice získáme $x_2 = 7,5$. Z první rovnice pak $x_1 = 4,5$. Dosazením do původních rovnic se přesvědčíme o správnosti řešení. Gaussova eliminační metoda je však metoda výpočetně náročná, pro soustavu n rovnic o n neznámých vyžaduje řádově n^3 početních operací. Pro rozsáhlé speciální matice s velkým počtem 0, které vznikají např. při numerickém řešení parciálních diferenciálních rovnic, byly vytvořeny speciální účinnější postupy, např. podle Jacobiho nebo Gausse-Seidela.

4.3.2. Inverzní matice

Čísla 3 a $\frac{1}{3}$ můžeme označit jako vzájemně inverzní, protože

$$3.\frac{1}{3} = \frac{1}{3}.3 = 1$$

$\frac{1}{3}$ též zapisujeme jako 3^{-1}. Podobný význam má inverzní matice. Mějme čtvercovou matici A. Matici A^{-1} označíme jako inverzní, pokud platí

$$A.A^{-1} = A^{-1}.A = E$$

kde E je jednotková matice. Např. k matici

$$A = \begin{pmatrix} 2 & 0 \\ 0 & -1 \end{pmatrix}$$

je inverzní matice

$$A^{-1} = \begin{pmatrix} \frac{1}{2} & 0 \\ 0 & -1 \end{pmatrix}$$

protože platí

$$AA^{-1} = \begin{pmatrix} 2 & 0 \\ 0 & -1 \end{pmatrix} \cdot \begin{pmatrix} \frac{1}{2} & 0 \\ 0 & -1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = E_2$$

Pro úplnost se ještě přesvědčíme, že platí i $A^{-1}A = E_2$.

$$A^{-1}A = \begin{pmatrix} \frac{1}{2} & 0 \\ 0 & -1 \end{pmatrix} \cdot \begin{pmatrix} 2 & 0 \\ 0 & -1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = E_2$$

Znát inverzní matici je užitečné. Vraťme se k předchozímu příkladu řešení soustavy rovnic $A\vec{x} = \vec{b}$. Pokud bychom znali inverzní matici A^{-1}, pak stačí obě strany rovnice touto maticí vynásobit

a hned máme hledané řešení $\vec{x} = A^{-1}\vec{b}$, protože

$$A^{-1}.A\vec{x} = A^{-1}A.\vec{x} = E.\vec{x} = \vec{x} = A^{-1}\vec{b}$$

Museli bychom sice vynásobit matici A^{-1} vektorem $\vec{b}$, to je ale podstatně jednodušší než provést celou Gaussovu eliminaci. Obtížné je ovšem spočítat inverzní matici. Výpočet se provádí variantou Gaussovu eliminace, je tedy vposledku stejně náročný. Pokud ale potřebujeme opakovaně řešit stejnou soustavu rovnic s odlišnými pravými stranami $\vec{b}$, pak už se postup s inverzní maticí vyplatí. Z hlediska geometrické interpretace provádí inverzní matice s vektorem opačnou operaci. Jestliže např. matice otáčí vektor o 30° po směru hodinových ručiček, inverzní matice jej otáčí o 30° proti směru hodinových ručiček.

A. Základní programové konstrukty jazyka Python

Zde si stručně popíšeme základní konstrukty jazyka Python - datové typy, podmínky, cykly a funkce. Pro jednoduché programy nám budou stačit.

A.1. Obecné poznámky k Pythonu

Python je v současnosti jedním z nejpopulárnějších programovacích jazyků. Je široce použitelný v mnoha oblastech, jako jsou vědecké výpočty, analýza dat, machine learning, programování aplikací, práce s databázemi nebo webové aplikace. Jedná se o tzv. **objektově orientovaný** jazyk, ale lze v něm programovat i „jednoduše“ neobjektově, tzv. procedurálně.

Python je **interpretovaný** (synonymum skriptovací) jazyk. Programátor píše obyčejný textový soubor, většinou s příponou .py, obvykle označovaný jako **skript**. Tento soubor se předloží programu s názvem `python`. Program `python` je tzv. interpret, tedy program, který rozumí textu ve skriptu, čte jej řádek po řádku a převádí jednotlivé příkazy do strojových příkazů (instrukcí) pro procesor počítače, který je vykonává. Pythonský skript `soubor.py` bychom tedy spustili např. z příkazové řádky Windows pomocí příkazu `C:\python soubor.py`.

Obvykle však používáme nějaký program usnadňující psaní skriptu, obecně označovaný jako IDE (Integrated Development Environment). IDE pro Python jsou např. Jupyter Notebook, Spyder, PyCharm či IDLE. Obvykle jsou součástí nějaké distribuce, což je soubor několika programů usnadňujících programování - distribuce obsahuje vlastní interpret `python`, nějaké IDE a řadu doplňkových modulů. Jednotlivé distribuce se liší právě těmito doplňkovými produkty, pythonský interpret je ve všech distribucích téměř stejný. Nejznámější distribucí je Anaconda, případně její stručnější verze Miniconda. Většinou pak kliknout na ikonu „Run“ a program přímo spustit.

Jistá slova, tzv. **klíčová slova** (keywords), např. `print`, `for`, `in` nebo `range`, mají v Pythonu vlastní speciální význam a nelze je použít např. jako název proměnné. S těmito asi 35 klíčovými slovy lze naprogramovat v podstatě cokoli (A.1). Zvládnout programování v základním Pythonu tedy vlastně znamená naučit se syntaxi (jak přesně se příkazy zapisují, aby jim interpret rozuměl) a sémantiku (co přesně znamenají) těchto klíčových slov. Krom toho ale existuje obrovské množství balíků, též označovaných jako **moduly** nebo **packages**, které usnadňují a rozšiřují schopnosti Pythonu. Byly však naprogramovány v základním Pythonu.

Zápis programu v Pythonu je jednoduchý. Namísto středníků či závorek používaných v jiných programovacích jazycích se pro účely seskupování textu programu („co k sobě patří“) používá odsazení, tzv. indentace. Indentaci nelze vynechat či pozměnit, změnila by se logika programu. Např. v následujícím programu je první print() součástí for cyklu (vytiskne se při každém běhu cyklu), druhý print() již není součástí (vytiskne se pouze jednou).

Tabulka A.1.: 35 klíčových slov jazyka Python

False	continue	global	pass
None	def	if	raise
True	del	import	return
and	elif	in	try
as	else	is	while
assert	except	lambda	with
async	finaly	nonlocal	yield
await	for	not	class
break	from	or	

```
>>> for i in range(4): # komentáře následují za #
                       # slouží k lepší orientaci programátora
                       # a čtenáře, interpret je přeskočí
>>>     print('je součástí')   # patří pod for cyklus
>>> print('není součástí')     # nepatří pod for cyklus
je součástí
je součástí
je součástí
je součástí
není součástí
```

A.2. Datové typy

Každá proměnná v Pythonu je určitého datového typu, například číslo nebo text. Proměnným určitého typu odpovídají jisté operace. Lze například sčítat dvě čísla, ale nelze sčítat číslo s písmenem, jinak Python hlásí chybu. Pokud napíšeme `x=3`, Python sám pozná, že proměnná x je typu int (celé číslo), proměnnou x vytvoří a přiřadí ji hodnotu 3.

A.2.1. integer a float

Integer je datový typ celých čísel, float značí desetinná čísla.

```
>>> a = 7          # číslo typu int
>>> b = 1.2        # číslo typu float
>>> print(type(a)) # type zjistí datový typ
>>> print(type(b)) # print vytiskne hodnotu
<class 'int'>
<class 'float'>

>>> 3+4   # obě čísla jsou int, výsledek je automaticky int
7
```

```
>>> 3.0+4 # první číslo je int, druhé float
7.0       # výsledek je automaticky float

>>> 3**4  # 3 na 4.
81

>>> z=(a+b)*(a-b) # v proměnných už jsou uloženy hodnoty
>>> print(z)      # lze normálně počítat
47.559999999999995
```

A.2.2. string

String je datový typ textových řetězců. Vlastní typické textové funkce, jako délka řetězce, poslední písmeno apod.

```
>>> fraze = 'textový řetězec je typu string'
>>> print(fraze)
>>> print('Delka fraze: ', len(fraze))
           # len zjistí délku řetězce
>>> print('9 prvnich pismen:', fraze[0:9])
           # [2:8] vybere podřetězec od písmene 2 do 8
>>> print('Posledni slovo velkymi:', fraze.split()[-1].upper())
           # rozštěpí řetězec na slova a vrátí poslední slovo
textový řetězec je typu string
Delka fraze:  18
9 prvnich pismen: textový ř
Posledni slovo velkymi pismeny: STRING
```

A.2.3. boolean

Boolean je logický datový typ, nabývá pouze hodnot pravda a nepravda, **True** a **False**.

```
>>> a = False
>>> print(a)
>>> print(type(a))
False
<class 'bool'>
>>> 5>4      # Je 5 větší než 4? ano
True
>>> 5 == 7   # POZOR!! == rovnost, = přiřazení
False
>>> 5 != 8   # nerovná se
True
```

```
 >>> print((5>8) or (5>4)) # platí jedno nebo druhé
 >>> print((5>8) | (5>4))  # alternativa pro or
True
True
 >>> print((5>8) and (5>4)) # platí jedno i druhé
 >>> print((5>8) & (5>4))   # alternativa pro and
False
False
\end{priiiklad}
```

A.2.4. list

List, česky seznam, je datový typ kolekce. Jiné objekty jakéhokoli datového typu lze tak „seskupit“ pod jedno označení.

```
>>> muj_seznam = ['one','two','three',4,5, [1,2,'list2']]
                 # definice kolekce s názvem muj_seznam
>>> my_list[5] # 6. prvek kolekce s indexem 5
                 # POZOR! 1. prvek má index 0,Python čísluje od 0
[1, 2, 'list2']
>>> my_list[5][-1] # z 6. prvku kolekce vybere poslední prvek
'list2'
```

A.2.5. tuple

Tuple, česky n-tice, je datový typ kolekce podobně jako list. Po jejím vytvoření však už nelze měnit její obsah.

```
>>> t = (1,2,3)  # definice kolekce tuple
>>> t[0]         # vybere 1. prvek s indexem 0
1
>>> t.append(4) # přidá prvek ... což nelze
# Python hlásí chybu
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-28-ada7ed8a579e> in <module>
----> 1 t.append(4)
AttributeError: 'tuple' object has no attribute 'append'

>>> t[1] = 5 # změní hodnotu 2. prvku ... což opět nelze
# Python hlásí chybu
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-29-c53026d12066> in <module>
```

```
----> 1 t[1] = 5
TypeError: 'tuple' object does not support item assignment
```

A.2.6. dictionary

Jak název říká, dictionary je datový typ obdobný slovníku - je to kolekce dvojic klíč:hodnota. Zadáním klíče lze najít odpovídající hodnotu, podobně jako dům:house.

```
>>> my_dict = {'key1':123,'key2':[12,23],'key3':['it0','it1']}
>>> my_dict['key3'][0] # vypíše první hodnotu pro klíč key3
'it0'
>>> my_dict['key4'] = [4444] # do my_dict přidá pár key4:4444
>>> my_dict.keys()    # vypíše všechny klíče
dict_keys(['key1', 'key2', 'key3', 'key4'])
>>> my_dict.values()  # vypíše všechny hodnoty
dict_values([123, [12, ..... , [4444]])
>>> my_dict.items()   # vypíše celé dictionary
dict_items([('key1', 123), ....., ('key4', [4444])])
```

A.3. Importování knihoven

Knihovny, moduly, pakety, packages a balíky jsou synonyma. Jsou to pythonské skripty, které obsahují jistý program. Pokud se nám takový program v rámci našeho programu hodí, importujeme daný modul a program použijeme, aniž bychom ho celý museli psát od začátku. Příkladem je modul numpy, který obsahuje řadu funkcí pro numerické výpočty. Moduly lze stáhnout z internetu a uložit do adresáře vlastního počítače, kde Python moduly hledá. Během importování se tam Python podívá, a pokud odpovídající modul najde, nahraje jej do paměti.

```
>>> import numpy as np # importuje a nahraje celý modul numpy
                      # do paměti,dále jej ale bude používat
                      # pod zkratkou np
>>> print(np.sqrt(9))  # použije funkci sqrt z modulu numpy
>>> print(np.arange(5))  # použije funkci arange z modulu numpy
>>> print(np.random.rand(5)) # použije funkci random.rand
3.0
[0 1 2 3 4]
[0.89479446 0.74400804 0.76156047 0.60258417 0.74705343]

>>> from numpy.random import rand # importuje a nahraje do
          #  paměti pouze funkci random z modulu numpy,
          #  čímž šetří paměť
>>> rand(5) # použije f. rand, která generuje 5 náhodných čísel
array([0.34395972, 0.854625, 0.769615, 0.085601, 0.674828])
```

A.4. Podmínka

Nutným konstruktem každého programovacího jazyka je podmínka, která umožní odlišné chování programu za různých situací. Základní součást podmínky je logický výraz, který je vyhodnocen jako pravdivý (`True`) nebo nepravdivý (`False`). Podmínku utvoříme dvojicí příkazů `If` a `else`, případně ještě `elif`. Co k sobě patří, určuje indentace.

```
>>> x = 5
>>> if x>5:           # podmínka
>>>     print('ano') # provede, pokud podmínka platí
>>> else:
>>>     print('ne')  # provede, pokud podmínka neplatí
ne

>>> x = 5
>>> if x>5:
>>>     print('>5')
>>> elif x==5:        # pokud podmínka neplatí, táže se dále
>>>     print('=5')  # pokud není větší než 5, je přesně 5?
>>> else:
>>>     print('<5')
=5
```

A.5. Cykly

Dalším základním konstruktem každého programovacího jazyka jsou cykly, tedy úseky programu, které jsou opakovány tak dlouho, dokud platí nějaká podmínka. Obvykle jsou k dispozici 2 cykly - **while** a **for**.

A.5.1. Cyklus for

For cyklus prochází předem zadanou sekvenci, např. hodnoty 1 až 5. Lze jej nahradit cyklem while.

```
>>> for jmeno in ["Petr",'Pavel','Eva', 'Martina']:
>>>      print(jmeno) # za jmeno se postupně dosazují jména
Petr
Pavel
Eva
Martina

>>> for i in range(5): range(n) vytvoří posloupnost 0 až n-1
>>>      print(i*i)    # proměnná i v každém kroku nabývá další
                       # hodnoty posloupnosti
0
```

```
1
4
9
16
```

A.5.2. Cyklus while

While cyklus se provádí, dokud platí zadaná podmínka. Pokud se podmínka nikdy nesplní, může dojít i k „zacyklení“ programu, který pak nikdy neskončí.

```
>>> i=0
>>> while i<5: # Nejprve se vyhodnotí podmínka. Pokud platí,
               # proběhne 1 cyklus a znovu se hodnotí podmínka
>>>       print(i)
>>>       i+=1  # je ekvivalentní s i=i+1; zvýší i o 1
0
1
2
3
4
```

A.6. Funkce

Dalším typickým konstruktem programovacích jazyků jsou funkce. Je to úsek textu programu, jemuž je přiřazen vlastní název a může být v programu opakovaně volán. Text funkce by se dal opakovaně psát přímo do hlavního textu programu, ale s pomocí funkcí se program mnohem lépe strukturuje i rychleji probíhá. Chovají se podobně jako funkce v matematické - nějaké proměnné, označované jako parametry, do funkce vstupují, jiné proměnné jsou výsledkem funkce. Říkáme, že funkce vrací určité hodnoty (čísla, písmena, logickou hodnotu apod.). Funkce je v Python samostatným objektem, má tedy své jméno a adresu v paměti, tedy v tomto smyslu „existuje“.

Funkci obecně definujeme takto:

```
>>> def jmenoFunkce(parametry): # definice názvu a parametrů
>>>     prikazovy blok # odsazeni říká, co vše patří do funkce
>>>     return (...)   # říká, co má funkce vracet
```

Ukažme si 2 příklady funkcí:

```
>>> def pocetSamohlasek(slovo): # funkce určí počet samohlásek
>>>     pocet = 0
```

```
>>>     for pismeno in slovo:
>>>         if pismeno in aeiouAEIOU:
>>>             pocet = pocet + 1
>>>     return(pocet)

>>> type(pocetSamohlasek)
function        # funkce je v Python samostatným objektem

>>> text = input('zadejte slovo: ') # input=zadání z klávesnice
>>> pocetSsamohlasek(text) # volá se funkce se zadaným textem
zadejte slovo: pokus
2
```

```
>>> def fce_na_ukazku(i):  # definice funkce fce_na_ukazku
>>>    return str(i) +'.: nedelam nic' # funkce se provede
                                       # a toto vrátí
>>> for i in range(5):
>>>    print(fce_na_ukazku(i)) #5x se volá f. a tiskne výsledek
0.: nedelam nic
1.: nedelam nic
2.: nedelam nic
3.: nedelam nic
4.: nedelam nic
```

Některé funkce jsou vestavěné (built-in functions) přímo v základní konstrukci Pythonu. Například print(), len() nebo type(). Ty můžeme používat ihned, aniž bychom předtím cokoli definovali. Další funkce pak „vlastní“ objekty určitého datového typu. Například objekt typu dictionary automaticky poskytuje funkce keys() nebo values(). Ty můžeme volat pouze „na tomto objektu“, samostatně je Python nezná.

A.7. "Komplexní"příklad

Následující „komplexní“ příklad obsahuje téměř vše výše probrané. Je jistě názorný a srozumitelný i bez podrobnějšího komentáře.

```
>>> def zadani():
>>>     x = float(input('Zadejte první číslo: '))
>>>     y = float(input('Zadejte druhé číslo: '))
>>>     return(x,y)

>>> def soucet():
>>>     x,y=zadani()
>>>     return(x+y)
```

```
>>> def rozdil():
>>>     x,y=zadani()
>>>     return(x-y)

>>> i=5
>>> while i!=0:
>>>     i = int(input('1 = součet, 2 = rozdíl, 0 = konec: '))
>>>     if i==1:
>>>         print('Součet je: ', soucet())
>>>     elif i==2:
>>>         print('Rozdíl je: ', rozdil())
>>> print('Program ukončen')

1 = součet, 2 = rozdíl, 0 = konec: 1
Zadejte první číslo: 5
Zadejte druhé číslo: 6
Součet je:  11.0
1 = součet, 2 = rozdíl, 0 = konec: 2
Zadejte první číslo: 4
Zadejte druhé číslo: 6
Rozdíl je:  -2.0
1 = součet, 2 = rozdíl, 0 = konec: 0
Program ukončen
```

B. Grafy funkcí v Pythonu

Nutnou dovedností v matematickém modelování je vizualizace výsledků. Ukážeme si, jak v Pythonu kreslit grafy funkcí jedné a dvou proměnných. Budeme používat funkce knihovny matplotlib. Jiná pythonská knihovna pro vizualizaci dat je např. seaborn.

B.1. Funkce jedné proměnné

B.1.1. Definice zobrazované funkce

Matplotlib kreslí funkce tak, že pouze spojí zadané body. Nejprve musíme „definovat“ funkci, neboli zadat body, tedy dvojice (x,y), které se mají spojit. Definujeme vektor „x-ových“ souřadnic bodů a odpovídající vektor „y-vých“ souřadnic. Pro definici vektorů a matematické operace s nimi použijeme knihovnu `numpy`. Vše ozřejmí příklad.

```
>>> import numpy as np # importuje knihovnu numpy

>>> x = np.linspace(0, 5,100) # vektor 100 hodnot od 0 do 5
>>> y = x ** 2        # 1. funkce — 2. mocnina, aplikována
                      # na každý element vektoru x
>>> z = 10*x ** 0.5   # 2. funkce — 10 * odmocnina
```

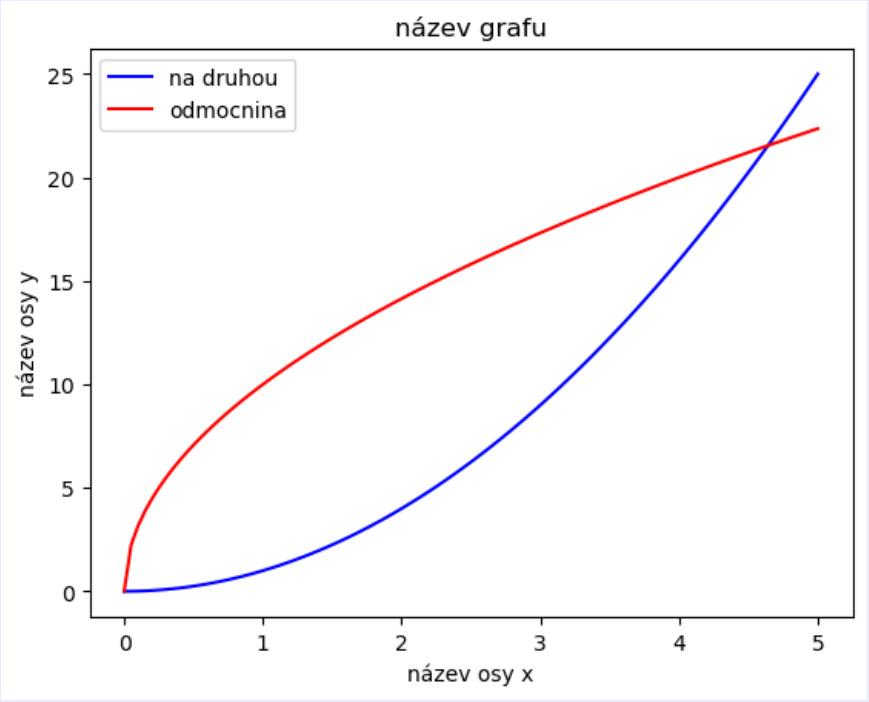
B.1.2. Vytvoření plátna a základního grafu

Kresba grafů má v Matplotlib „objektově orientovaný charakter“. Nejprve vezmeme „papír“, říkáme mu plátno nebo canvas, a v něm si vyhradíme zatím prázdný graf. Obojí provede funkce subplots(). Jejím výstupem jsou dva objekty - plátno a na něm graf. Nyní voláme různé funkce na objektu graf, které do něj dokreslují různé jednotlivosti, jako vlastní graf funkce, osy, názvy os apod.

```
>>> import matplotlib.pyplot as plt # importuje funkci pyplot
                                    # knihovny matplotlib
>>> fig, axes = plt.subplots()      # vytvoří "plátno" (canvas)
                                    # a prázdný graf

# nyní následuje kreslení do vytvořeného grafu
>>> axes.plot(x,y,'b',label='na druhou') # spojí dvojice x,y
>>> axes.plot(x,z,'r',label='odmocnina') # x—z označí odmocnina
>>> axes.set_xlabel('název osy x') # set_ v názvu metody
```

```
>>> axes.set_ylabel('název osy y') # nastavuje parametry grafu
>>> axes.set_title('název grafu')  # např. název grafu
>>> axes.legend()                  # do grafu se přidá legenda
>>> fig                            # vykreslí vytvořený graf
```

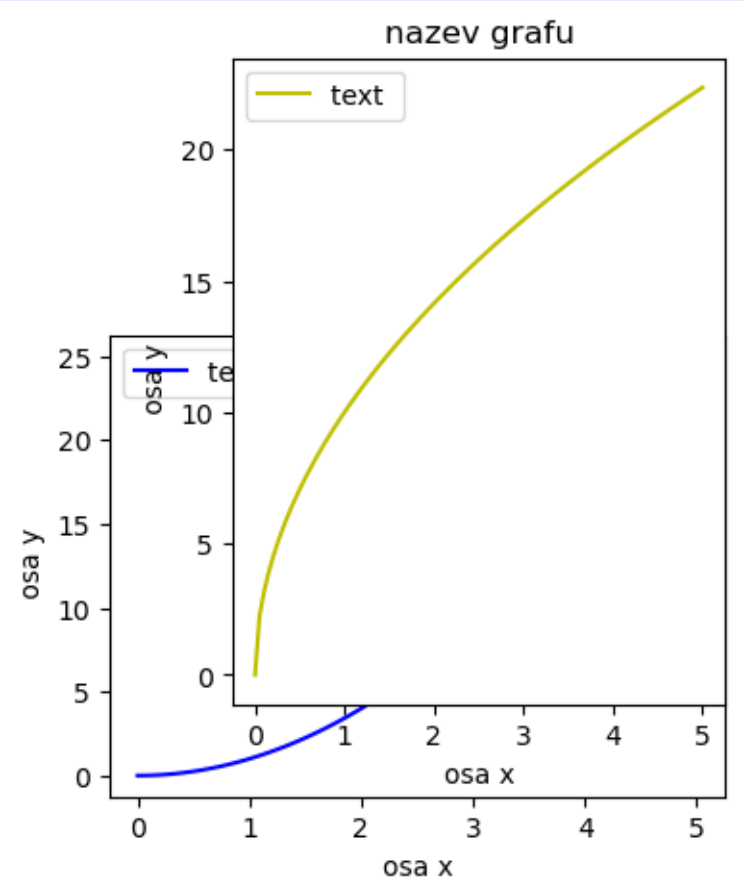
Uvedený program vykreslí následující graf s dvěma funkcemi.



Alternativou je vytvořit pouze plátno pomocí funkce `figure()` a na něm volat funkci `add_axis()`, která na plátně vytvoří prázdný graf. Umožní tak na 1 plátno nakreslit více grafů a nastavit, kde na plátně se grafy nacházejí.

```
>>> fig = plt.figure() # vytvoří pouze "plátno" (canvas)

# přidá axes (= graf) na plátno
>>> axes = fig.add_axes([0.1, 0.1, 0.5, 0.5])
                  # left, bottom, width, height (rozsah 0 to 1)
                  # určí, kde na plátně bude graf ležet
                  # ! relativní umístění
        axes2 = fig.add_axes([0.2, 0.2, 0.4, 0.7]) # druhý graf
                                                   # na stejném objektu fig

# kreslení do grafu 1
>>> axes.plot(x, y, 'b', label = 'text legendy')
>>> axes.set_xlabel('osa x')
>>> axes.set_ylabel(' osa y')
>>> axes.set_title('nazev grafu')
>>> axes.legend()

# kreslení do grafu 2
```

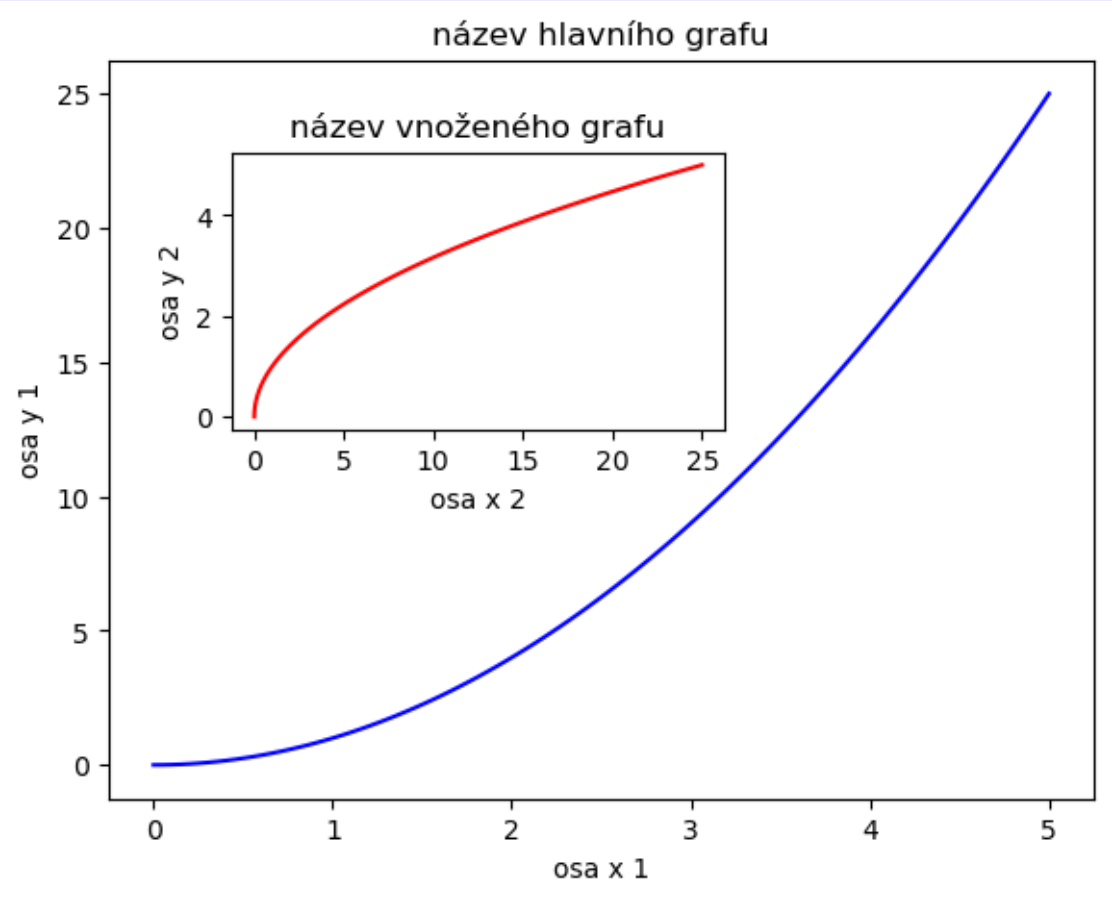
```
>>> axes2.plot(x, z, 'y', label = 'text ')
>>> axes2.set_xlabel('osa x')
>>> axes2.set_ylabel(' osa y')
>>> axes2.set_title('nazev grafu')
>>> axes2.legend()
```



Takto lze „fiktivně“ nakreslit i vnořený graf.

```
>>> fig = plt.figure()
>>> axes1 = fig.add_axes([0.1, 0.1, 0.8, 0.8]) # hlavní graf
>>> axes2 = fig.add_axes([0.2, 0.5, 0.4, 0.3]) # vnořený graf
                # vnořený graf je menší a má posunutou polohu
# hlavní graf
>>> axes1.plot(x, y, 'b')
>>> axes1.set_xlabel('osa x 1')
>>> axes1.set_ylabel('osa y 1')
>>> axes1.set_title('název hlavního grafu')

# vložený graf
>>> axes2.plot(y, x, 'r')
>>> axes2.set_xlabel('osa x 2')
>>> axes2.set_ylabel('osa y 2')
```

```
>>> axes2.set_title('název vnořeného grafu');
```



B.1.3. Uložení grafu

Vytvořený graf lze (v rámci programu) přímo uložit pomocí funkce `savefig()`.

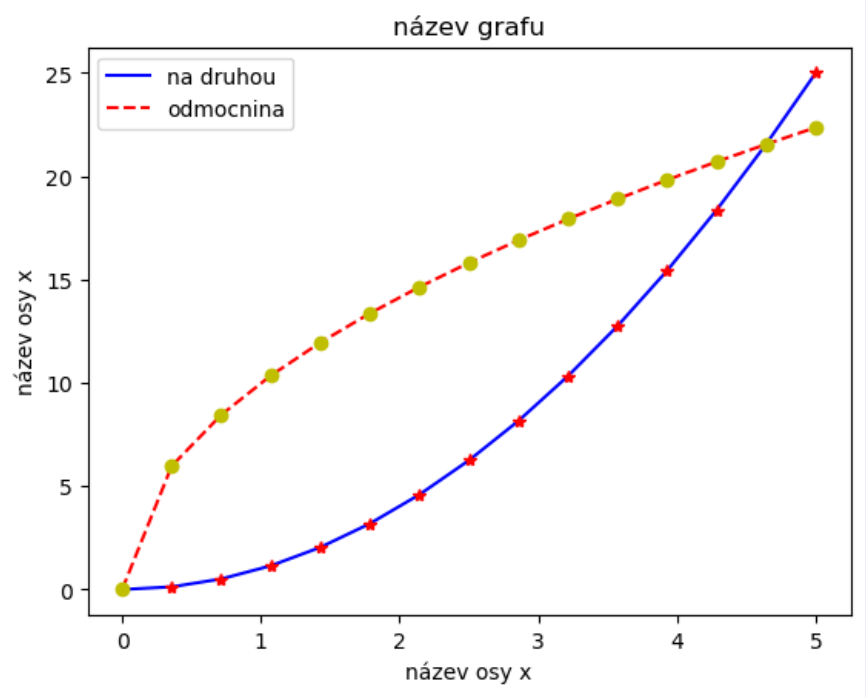
```
>>> plt.savefig('uloha.png', dpi = 300)
     # název souboru (možné uložit jako .PNG, .PDF, .SVG)
     # specifikace rozlišení
```

B.1.4. Specifikace podoby grafu funkce

Matplotlib nabízí řadu možností, jak upravit vzhled grafu podle potřeby. Například:

```
>>>  fig, axes = plt.subplots()

# kreslení do grafu
>>> axes.plot(x, y, 'b', label = 'na druhou')
>>> axes.plot(x, z, 'r', ls = '--', label = 'odmocnina')
```

```
>>> axes.plot(x, y,'*r') # červené *
>>> axes.plot(x, z,'oy') # žlutá o
>>> axes.set_xlabel('název osy x')
>>> axes.set_ylabel('název osy x')
>>> axes.set_title('název grafu')
>>> axes.legend()
fig
```

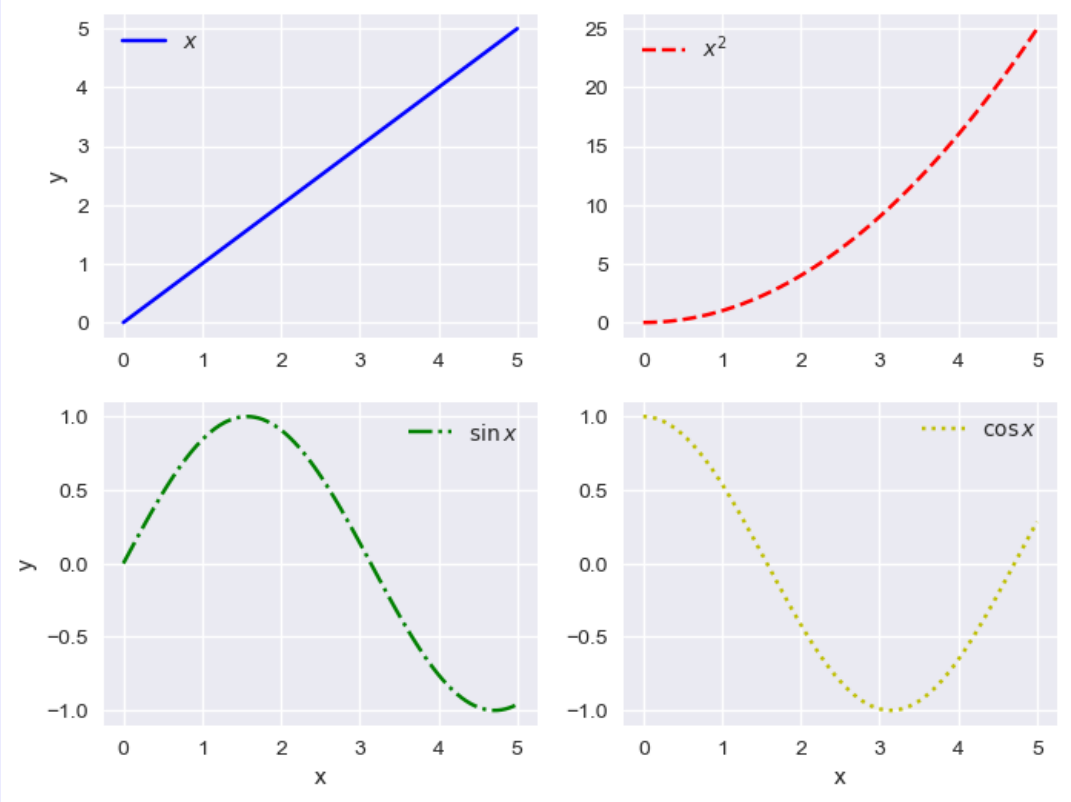


B.1.5. Více grafů na plátně

Více grafů na plátně lze nakreslit buď pomocí funkcí `figure()` a `add_axis()`, jak jsme již popsali, nebo přímo pomocí funkce `subplots()`, kde lze specifikovat „strukturu“ grafů (např. 1x3, 2x2) a jejich velikost. Jednotlivé grafy se pak volají dvojicí indexů [i,j].

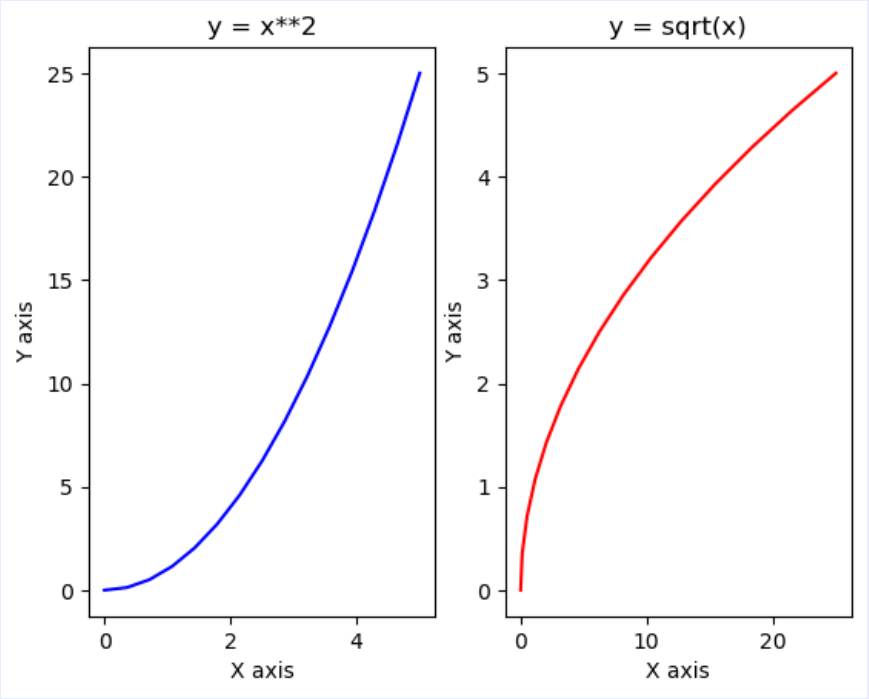
```
>>> import matplotlib.pyplot as plt
>>> import numpy as np

>>> fig, axes = plt.subplots(2,2, figsize = (8,6))
        # vytvoří na plátně 2x2 grafů, každý velikosti 8x6

>>> x = np.linspace(0, 5,100) # definice 4 funkcí
>>> y00 = x
>>> y01 = x*x
>>> y10 = np.sin(x)
>>> y11 = np.cos(x)

# kreslení do grafů
```

```
>>> axes[0,0].plot(x, y00, 'b', label='$x$') # levý horní graf
>>> axes[0,0].set_ylabel('y')
>>> axes[0,0].legend()

>>> axes[0,1].plot(x, y01, 'r',ls= '--', label = '$x^2$')
>>> axes[0,1].legend()

>>> axes[1,0].plot(x, y10, 'g', ls= '-.',label = '$\sin x$')
>>> axes[1,0].set_xlabel('x')
>>> axes[1,0].set_ylabel('y')
>>> axes[1,0].legend()

>>> axes[1,1].plot(x, y11, 'y', ls='dotted',label='$\cos x$')
>>> axes[1,1].set_xlabel('x')
>>> axes[1,1].legend()
```




```
>>> fig, axes = plt.subplots(nrows=1, ncols=2)

# levý graf
>>> axes[0].plot(x, y, 'b')
>>> axes[0].set_title('y = x**2')
>>> axes[0].set_xlabel('X axis')
>>> axes[0].set_ylabel('Y axis')

# pravý graf
```

```
>>> axes[1].plot(y, x, 'r')
>>> axes[1].set_title('y = sqrt(x)')
>>> axes[1].set_xlabel('X axis')
>>> axes[1].set_ylabel('Y axis')
```

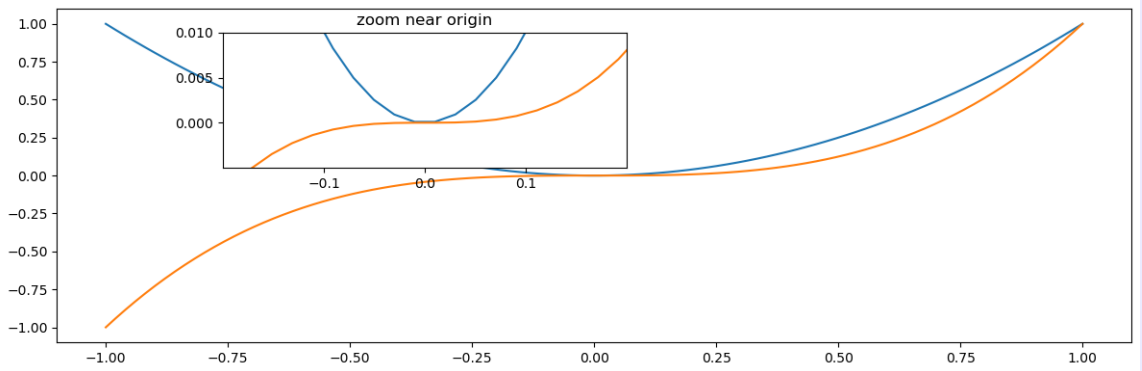


B.1.6. Nastavení os

Matplotlib umožňuje mnohá nastavení os. V příkladu je vidět vnořený graf, kde kreslíme stejné funkce, ale zaměřujeme se na bod dotyku.

```
>>> xx = np.linspace(-1,1,100)

>>> fig, ax = plt.subplots(figsize = (10,5)) # velikost plátna

>>> ax.plot(xx, xx**2, xx, xx**3)
>>> fig.tight_layout()   # minimalizuje překrývání grafů

# vnitřní graf
>>> vnitrni_ax = fig.add_axes([0.2, 0.55, 0.35, 0.35])
                         # poloha x,  poloha y, šířka, výška

>>> vnitrni_ax.plot(xx, xx**2, xx, xx**3)
>>> vnitrni_ax.set_title('zoom near origin')

# nastavení os vnitřního grafu
>>> vnitrni_ax.set_xlim(-.2, .2)     # rozsah osy x
>>> vnitrni_ax.set_ylim(-.005, .01)  # rozsah osy y
```

```
# nastavení ukazatelů os vnitřního grafu
>>> vnitrni_ax.set_yticks([0, 0.005, 0.01]) # ukazatele x
>>> vnitrni_ax.set_xticks([-0.1,0,.1]);     # ukazatele y
```



B.1.7. Formátování grafu

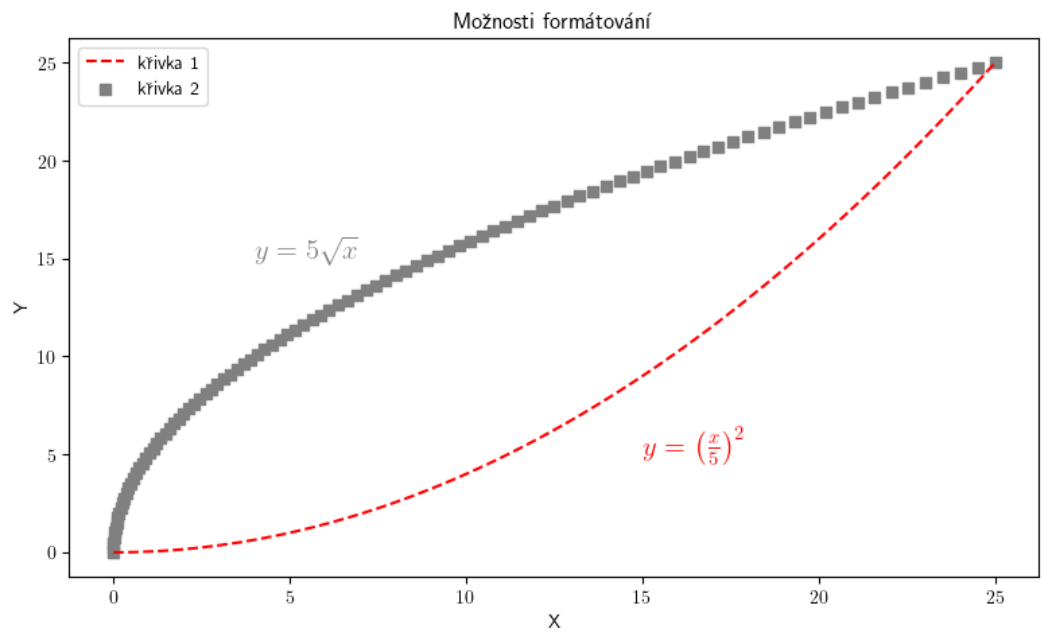
Matplotlib poskytuje mnoho možností formátování grafu. Buď můžeme nastavit jednotlivosti, jako jsou například typ, barva nebo šířka křivky, nebo si vybereme z mnoha předdefinovaných stylů. Detaily možných nastavení lze najít v dokumentaci pro Matplotlib:

Seznam typů markerů	https://matplotlib.org/3.1.1/api/markers_api.html
Seznam barev	https://matplotlib.org/3.1.0/gallery/color/named_colors.html
Úpravy textu v obrázcích	https://matplotlib.org/stable/users/explain/text/text_intro.html

V obrázcích lze použít i matematický font ve stylu LaTeXu. Některé možnosti ukazuje následujících příklad.

```
>>> fig = plt.figure(figsize = (9,5), dpi = 100)

>>> ax = fig.add_axes([0.1,0.1,0.8,0.8])
>>> ax.set_title('Možnosti formátování')

>>> ax.plot(x*5,y,
>>> color = 'red',      # barva
>>> ls = '--',          # styl křivky
>>> label = 'křivka 1' # název křivky
>>> )

>>> ax.scatter(y, x*5, # scatter - jako plot, ale nespojí body
>>> color = 'grey',
>>> marker = 's',       # styl markeru
>>> label = 'křivka 2'
>>> )
```

```
>>> ax.set_xlabel('X')  # název osy x
>>> ax.set_ylabel('Y')  # název osy x

>>> plt.legend()        # přidání legendy

# popisky = anotace
>>> plt.rcParams['text.usetex'] = True # zapne matematický styl

# matematický text stylu LaTeXu
>>> ax.text(15,5,r"$y=\left(\frac{x}{5}\right)^2$",
                                        fontsize=15, color="red")
>>> ax.text(4, 15, r"$y=5\sqrt{x}$", fontsize=15, color="grey")
```



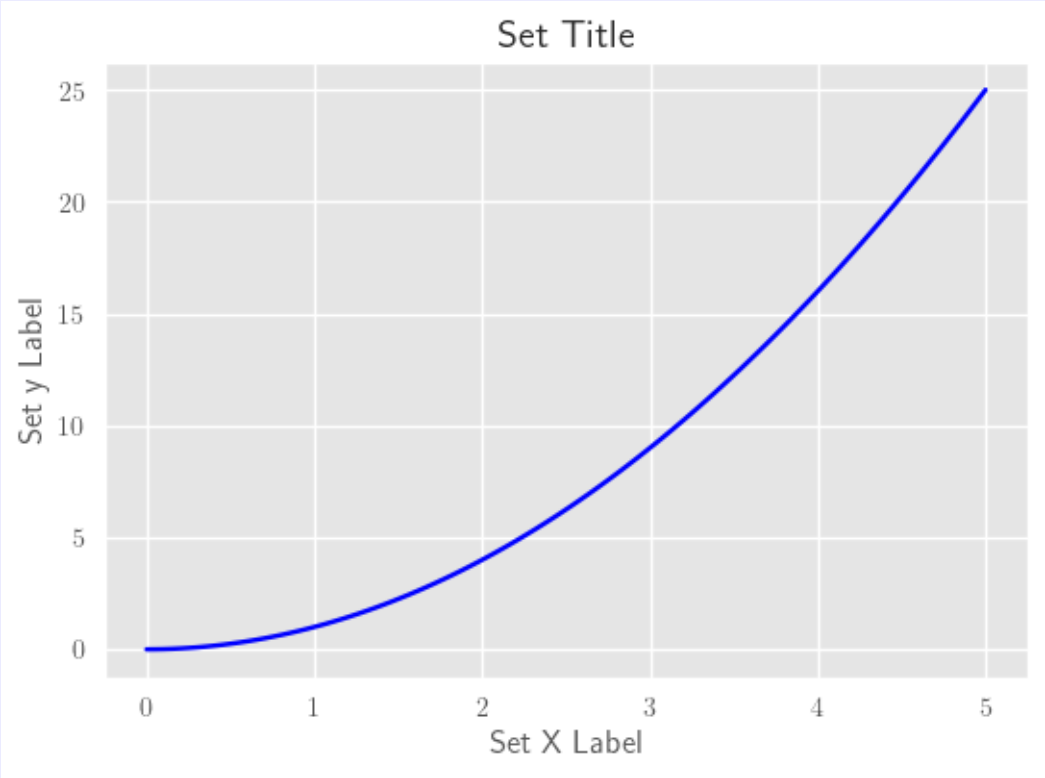
B.1.8. Styly

Stylem se rozumí komplexní předdefinovaná podoba grafu. Lze si vybrat z mnoha stylů, jak uvádí dokumentace pro Matplotlib - https://matplotlib.org/3.1.1/gallery/style_sheets/ style_sheets_reference.html.

V následujícím příkladu použijeme styl ggplot, který jinak používá stejnojmenná knihovna programovacího jazyka R.

```
# použití ggplot stylu
>>> matplotlib.style.use('ggplot') # volba stylu ggplot

>>> fig = plt.figure(figsize=(6,4))
>>> axes = fig.add_axes([0.1, 0.1, 0.8, 0.8])
>>> axes.plot(x, y, 'b')
>>> axes.set_xlabel('Set X Label')
```

```
>>> axes.set_ylabel('Set y Label')
>>> axes.set_title('Set Title')
```



Seznam stylů můžeme též najít takto:

```
# dostupné styly
>>> for i in matplotlib.style.available:
>>>     print(i)

Solarize_Light2
_classic_test_patch
bmh
...
```

B.2. Funkce dvou proměnných

Pro kresbu funkcí 2 proměnných opět použijeme knihovnu matplotlit, nyní funkci `plot_surface()`. Nejprve musíme definovat proměnné, obdobně jako v případě funkcí 1 proměnné. Dále musíme definovat vlastní funkci 2 proměnných pomocí příkazu `def`. Dále pomocí funkce `np.meshgrid` vytvoříme síť dvojic hodnot (x,y) v základně grafu, na níž definujeme proměnnou z. Tak vlastně definujeme skupinu bodů v prostoru, které funkce `ax.plot_surface(X, Y, Z)` pospojuje v prostorový graf.

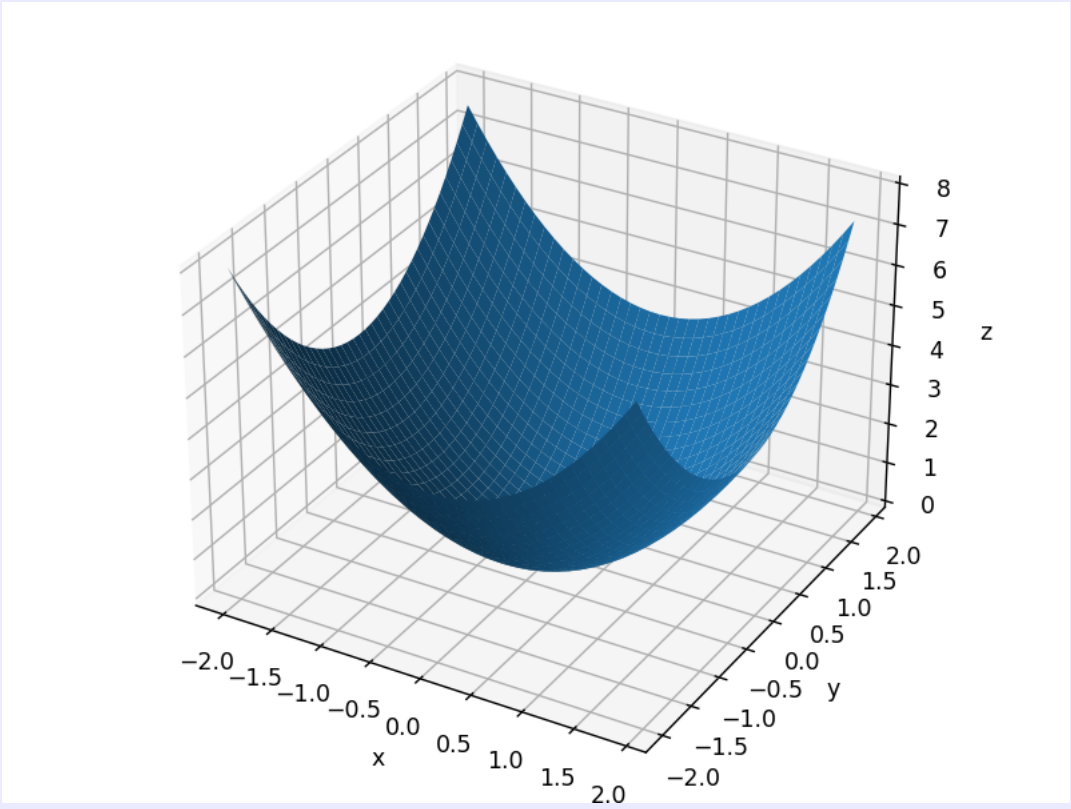
Pokud budeme chtít použít interaktivní mód, kdy lze myší otáčet grafem, musíme předem na-

instalovat a poté importovat knihovnu `ipympl`. [1]

Nakresleme graf gunkce $f(x, y) = x^2 + y$, tedy rotačního paraboloidu.

```
>>> import matplotlib.pyplot as plt
>>> from mpl_toolkits.mplot3d import Axes3D
>>> import numpy as np
>>> # import ipympl              # interaktivní mód

>>> #% matplotlib widget

>>> x = np.arange(-15, 15, 0.1) # definice proměnné x
>>> y = np.arange(-15, 15, 0.1) # definice proměnné y

>>> def fun(x, y):              # definice funkce
>>>     return x**2 + y**2

>>> X, Y = np.meshgrid(x, y)    # vytvoří X-Y síť
>>> Z = fun(X,Y)                # definice proměnné z

>>> fig = plt.figure()
>>> ax = fig.add_subplot(111, projection='3d')

>>> ax.plot_surface(X, Y, Z) # vykreslení prostorového grafu

>>> ax.set_xlabel('osa x')
>>> ax.set_ylabel('osa y')
>>> ax.set_zlabel('osa z')

>>> ax.view_init(-50, 50)    # iniciálního natočení grafu
>>> plt.show()               # zobrazení grafu
```

[1] Instalace knihovny `ipympl`: Otevřeme Anacondu - Anaconda Powershell Prompt a napišme **conda install ipympl**.


8
7
6
5
4
3
2
1
0
z
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
y
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
x

Použitá a doporučená literatura

1. Arens T, Hettlich F, Karpfinger C, Kockelkorn U, Lichtenegger K, Stachel H, eds. Mathematik. 2., korrigierter Nachdr. Spektrum, Akad. Verl; 2010.

2. Arfken G, Weber HJ, Harris FE. Mathematical Methods for Physicists: A Comprehensive Guide. 7th ed. Elsevier; 2013.

3. Bender EA. An Introduction to Mathematical Modeling. Dover Publications; 2000.

4. Chongchitnan S. Exploring University Mathematics with Python. Springer; 2023.

5. Jüngel A, Zachmann HG. Mathematik für Chemiker. 8. Auflage. Wiley-VCH; 2023.

6. Riley KF, Hobson MP, Bence SJ. Mathematical Methods for Physics and Engineering. 3rd ed. Cambridge University Press; 2006.

7. Mathematical Modeling. Springer Berlin Heidelberg; 2017.

8. McKinney W. Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython. Second edition. O'Reilly Media, Inc; 2018.

9. Pecinovský R. Python: kompletní příručka jazyka pro verzi 3.10. První vydání. Grada Publishing; 2021.

10. Jänich K. Linear Algebra. Springer; 1994.

11. Musilová J, Musilová P. Matematika I-III: pro porozumění i praxi : netradiční výklad tradičních témat vysokoškolské matematiky. 2., dopl. vyd. VUTIUM; 2009.

12. Kubíček M, Dubcová M, Janovská D. Numerické metody a algoritmy. Vyd. 2., opr. Vysoká škola chemicko-technologická; 2005.