

Přehled matematického software pro výuku

Petr Jurša

Bakalářská práce
2025



Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav počítačových a komunikačních systémů

Akademický rok: 2024/2025

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení: Petr Jurša
Osobní číslo: A22109
Studijní program: B0688A140008 Informační technologie v administrativě
Forma studia: Prezenční
Téma práce: Přehled matematického software pro výuku
Téma práce anglicky: An Overview of Mathematical Software for Teaching

Zásady pro vypracování

- Vytvořte literární rešerši z oblasti matematického software, který lze použít při výuce.
- Sestavte výukové prezentace popisující matematický software.
- Vypracujte sadu příkladů demonstrujících základy práce v tomto software.
- Vytvořte český manuál k vybranému software.
- Prakticky ověřte vámi vytvořené podklady pomocí dotazníku.

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam doporučené literatury:

1. SCHMIDT HANSEN, Jesper. GNU Octave: beginner's guide : become a proficient Octave user by learning this high-level scientific numerical tool from the ground up. Learn by doing: less theory, more results. Birmingham: Packt, 2011. ISBN 978-1-849513-32-6.
2. SCILAB ENTERPRISES a GOMEZ, Christine, 2013. Scilab for very beginners. [online]. Dostupné také z: https://www.scilab.org/sites/default/files/Scilab_beginners.pdf
3. GEOGEBRA, 2011. GeoGebra Manual: The official manual of GeoGebra. [online]. Dostupné z: https://research.shu.ac.uk/geogebra/GIS_Guides/Official%20GeoGebra%20Manual.pdf. [cit. 2024-10-23]
4. SYMPY DEVELOPMENT TEAM, 2024. SymPy Documentation. [online]. Release 1.13.3. Dostupné z: <https://github.com/sympy/sympy/releases/download/1.13.3/sympy-docs-pdf-1.13.3.pdf>. [cit. 2024-10-23]
5. KELECSÉNYI, Klára; OSZTÉNYINÉ, Krauczi Éva a DIÁNA, Zita, 2021. Photomath : A blessing or a Curse. No 1. Gradus. ISSN 2064-8014.

Vedoucí bakalářské práce: **Ing. Karel Perůtka, Ph.D.**
Ústav řízení procesů

Datum zadání bakalářské práce: **4. prosince 2024**
Termín odevzdání bakalářské práce: **9. června 2025**

doc. Ing. Jiří Vojtěšek, Ph.D. v.r.
děkan



Ing. Miroslav Matýšek, Ph.D. v.r.
ředitel ústavu

Ve Zlíně dne 4. prosince 2024

PROHLÁŠENÍ AUTORA BAKALÁŘSKÉ PRÁCE

Beru na vědomí, že

- odevzdáním bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb., v platném znění bez ohledu na výsledek obhajoby;
- bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému a bude dostupná k nahlédnutí;
- jedno vyhotovení bakalářské práce v listinné podobě bude ponecháno Univerzitě Tomáše Bati ve Zlíně k uložení;
- na moji bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- podle § 60 odst. 1 autorského zákona má Univerzita Tomáše Bati ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- podle § 60 odst. 2 a 3 mohu užít své dílo – bakalářskou práci – nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše);
- pokud bylo k vypracování bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tj. k nekomerčnímu využití), nelze výsledky bakalářské práce využít ke komerčním účelům;
- pokud je výstupem bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá; neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji, že

- jsem na bakalářské práci pracoval(a) samostatně a použitou literaturu jsem řádně citoval(a); v případě publikace výsledků budu uveden(a) jako spoluautor;
- odevzdaná verze bakalářské práce a verze elektronická nahraná do IS/STAG jsou obsahově totožné.

Ve Zlíně, dne

.....

podpis autora

ABSTRAKT

Tato bakalářská práce analyzuje matematický software vhodný pro vzdělávací účely a vytváří výukové materiály pro programovací jazyk Python a jeho matematické knihovny. Teoretická část systematicky představuje osm vybraných softwarových nástrojů rozdělených do kategorií vědeckých výpočetních systémů, dynamických geometrických systémů, systémů pro symbolickou matematiku a mobilních aplikací. Praktická část se zaměřuje na tvorbu komplexní sady výukových materiálů v českém jazyce zahrnující čtyři výukové prezentace, třicet demonstračních příkladů a český manuál o rozsahu sedmdesáti sedmi stran. Vytvořené materiály byly ověřeny dotazníkovým šetřením.

Klíčová slova: matematický software, Python, NumPy, SciPy, Matplotlib, výukové materiály, vzdělávací technologie

ABSTRACT

This bachelor thesis analyses mathematical software suitable for educational purposes and creates tutorials for the Python programming language and its mathematical libraries. The theoretical part systematically presents eight selected software tools categorized into scientific computing systems, dynamic geometric systems, symbolic mathematics systems and mobile applications. The practical part focuses on the creation of a comprehensive set of tutorial materials in Czech language, including four tutorial presentations, thirty demonstration examples and a Czech manual of seventy-seven pages. The developed materials were validated by a questionnaire survey.

Keywords: mathematical software, Python, NumPy, SciPy, Matplotlib, teaching materials, educational technology

Rád bych vyjádřil upřímné poděkování svému vedoucímu práce Ing. Karlu Perůtkovi, Ph.D., za odborné vedení, cenné rady a metodické pokyny, které významně přispěly k realizaci této bakalářské práce.

Děkuji také všem respondentům dotazníkového šetření za poskytnutí zpětné vazby k vytvořeným výukovým materiálům.

V neposlední řadě patří poděkování mé rodině za podporu a pochopení během celého procesu psaní této práce.

OBSAH

ÚVOD.....	13
I TEORETICKÁ ČÁST	15
1 CHARAKTERISTIKA VYBRANÝCH MATEMATICKÝCH SOFTWARE	16
2 VĚDECKÉ VÝPOČETNÍ SYSTÉMY	18
2.1 GNU OCTAVE	18
2.1.1 Historie a vývoj	18
2.1.2 Základní charakteristiky a funkce	18
2.1.3 Uživatelské rozhraní.....	19
2.1.4 Možnosti využití ve výuce	19
2.2 SCILAB	20
2.2.1 Historie a vývoj	20
2.2.2 Základní charakteristiky a funkce	20
2.2.3 Uživatelské rozhraní.....	20
2.2.4 Možnosti využití ve výuce	21
2.3 POROVNÁNÍ S KOMERČNÍMI ALTERNATIVAMI (MATLAB)	21
3 DYNAMICKÉ GEOMETRICKÉ SYSTÉMY	23
3.1 GEOGEBRA	23
3.1.1 Historie a vývoj	23
3.1.2 Základní charakteristiky a funkce	23
3.1.3 Uživatelské rozhraní.....	24
3.1.4 Možnosti využití ve výuce	24
4 SYSTÉMY PRO SYMBOLICKOU MATEMATIKU	26
4.1 SYMPY	26
4.1.1 Historie a vývoj	26
4.1.2 Základní charakteristiky a funkce	26
4.1.3 Integrace s Pythonem a dalšími nástroji.....	27
4.1.4 Uživatelské rozhraní.....	27
4.1.5 Možnosti využití ve výuce	27
4.2 KOMERČNÍ SYSTÉMY PRO SYMBOLICKOU MATEMATIKU	28
4.3 MAPLE	28
4.3.1 Historie a vývoj	28
4.3.2 Základní charakteristiky a funkce	28
4.3.3 Uživatelské rozhraní.....	29
4.3.4 Možnosti využití ve výuce	29
4.4 WOLFRAM MATHEMATICA.....	29
4.4.1 Historie a vývoj	29
4.4.2 Základní charakteristiky a funkce	30
4.4.3 Uživatelské rozhraní.....	30
4.4.4 Možnosti využití ve výuce	30

5	MOBILNÍ APLIKACE A ONLINE NÁSTROJE PRO VÝUKU.....	32
5.1	PHOTOMATH.....	32
5.1.1	Historie a vývoj	32
5.1.2	Základní charakteristiky a funkce	32
5.1.3	Oblasti matematiky pokryté aplikací.....	32
5.1.4	Vzdělávací aspekty a kontroverze	33
5.2	MICROSOFT MATH SOLVER.....	34
5.2.1	Historie a vývoj	34
5.2.2	Základní charakteristiky a funkce	34
5.2.3	Uživatelské rozhraní.....	34
5.2.4	Oblasti matematiky pokryté aplikací.....	35
6	PYTHON A JEHO MATEMATICKÉ KNIHOVNY	36
6.1	HISTORIE A VÝVOJ PYTHONU	36
6.2	NUMPY.....	36
6.2.1	Základní charakteristiky a funkce	36
6.2.2	Uživatelské rozhraní.....	37
6.2.3	Možnosti využití ve výuce	37
6.3	SCIPY	38
6.3.1	Základní charakteristiky a funkce	38
6.3.2	Uživatelské rozhraní.....	38
6.3.3	Možnosti využití ve výuce	38
6.4	MATPLOTLIB	39
6.4.1	Základní charakteristiky a funkce	39
6.4.2	Uživatelské rozhraní.....	39
6.4.3	Možnosti využití ve výuce	40
6.5	INTEGRACE S DALŠÍMI NÁSTROJI A KNIHOVNAMI.....	40
6.6	POROVNÁNÍ S DALŠÍMI NÁSTROJI PRO MATEMATICKÉ VÝPOČTY	41
II	PRAKTICKÁ ČÁST	42
7	VÝUKOVÉ MATERIÁLY PRO PYTHON A JEHO MATEMATICKÉ KNIHOVNY.....	43
7.1	METODOLOGIE A CÍLE VÝUKOVÝCH MATERIÁLŮ	44
7.1.1	Cílové skupiny a metodologické přístupy	44
7.1.2	Cíle a struktura výukových materiálů	44
7.2	VÝUKOVÉ PREZENTACE	44
7.2.1	Úvod do Pythonu.....	45
7.2.2	Základy práce s NumPy	49
7.2.3	Vědecké výpočty s SciPy	51
7.2.4	Vizualizace dat s Matplotlib.....	53
7.3	DEMONSTRAČNÍ PŘÍKLADY.....	55
7.3.1	Příklady pro základní operace s NumPy	55
7.3.2	Příklady pro vědecké výpočty s SciPy	57

7.3.3	Příklady pro vizualizaci s Matplotlib	58
7.4	ČESKÝ MANUÁL PRO PYTHON A JEHO MATEMATICKÉ KNIHOVNY.....	60
7.4.1	Struktura a obsah manuálu	60
7.4.2	Specifika českého názvosloví a terminologie	62
7.5	PRAKTICKÉ OVĚŘENÍ MATERIÁLŮ.....	63
7.5.1	Metodologie hodnocení.....	63
7.5.2	Analýza zpětné vazby	64
7.5.3	Výsledky dotazníkového šetření	65
7.6	NÁVRHY NA DALŠÍ ROZVOJ VÝUKOVÝCH MATERIÁLŮ	66
ZÁVĚR		68
SEZNAM POUŽITÉ LITERATURY.....		70
SEZNAM OBRÁZKŮ		74
SEZNAM TABULEK.....		75
SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK		76
SEZNAM PŘÍLOH.....		77

ÚVOD

Moderní vzdělávání v oblasti matematiky a příbuzných technických disciplín čelí výzvě integrace pokročilých technologických nástrojů do tradičních výukových metod. Matematický software představuje klíčový prostředek pro zpřístupnění abstraktních konceptů prostřednictvím interaktivních vizualizací, praktických experimentů a aplikací teoretických poznatků na reálné problémy. Zatímco v oblasti numerických výpočtů a symbolické matematiky existuje široká škála dostupných řešení, volba vhodného nástroje pro konkrétní vzdělávací účely vyžaduje systematické porovnání jejich možností a limitací.

Současný stav matematického softwaru zahrnuje pestrou škálu produktů od specializovaných vědeckých výpočetních systémů přes dynamické geometrické nástroje až po moderní mobilní aplikace využívající technologie rozpoznávání obrazu. Tato diverzita současně představuje příležitost i výzvu pro pedagogy, kteří musí navigovat mezi open-source a komerčními řešeními s ohledem na technické možnosti, licenční podmínky a pedagogické cíle svých kurzů. Zvláště významnou roli v tomto prostředí hraje programovací jazyk Python, který díky své jednoduchosti a rozsáhlému ekosystému matematických knihoven získává rostoucí popularitu ve vědecké komunitě i vzdělávacích institucích.

Problematiku implementace matematického softwaru do výuky komplikuje nedostatek kvalitních výukových materiálů v českém jazyce, což představuje významnou bariéru pro širší adopci těchto nástrojů v českém vzdělávacím prostředí. Zatímco v angličtině existuje rozsáhlá dokumentace a výukové zdroje, česká vzdělávací komunita často čelí nutnosti samostatné adaptace zahraničních materiálů nebo spoléhání na neúplné překladové zdroje.

Tato bakalářská práce si klade za cíl poskytnout komplexní přehled současného stavu matematického softwaru vhodného pro vzdělávací účely a současně přispět k řešení identifikovaných nedostatků vytvořením ucelené sady výukových materiálů v českém jazyce. Teoretická část systematicky analyzuje osm reprezentativních softwarových nástrojů rozdělených do logických kategorií podle jejich zaměření a funkcionality. Praktická část se zaměřuje na tvorbu výukových materiálů pro Python a jeho klíčové matematické knihovny NumPy, SciPy a Matplotlib, které představují moderní a dostupný nástroj pro výuku matematiky s využitím programování.

Hlavní přínos práce spočívá v poskytnutí objektivního srovnání dostupných možností matematického softwaru pro pedagogické využití a ve vytvoření prakticky orientovaných výukových zdrojů, které mohou přispět ke zkvalitnění výuky matematiky na středních a

vysokých školách. Výsledky práce mají potenciál podporovat rozvoj digitálních kompetencí studentů a usnadnit integraci moderních výpočetních nástrojů do matematického vzdělávání v českém prostředí.

I TEORETICKÁ ČÁST

1 CHARAKTERISTIKA VYBRANÝCH MATEMATICKÝCH SOFTWARE

V této kapitole je představeno osm reprezentativních matematických softwarových nástrojů, které jsou vhodné pro implementaci do výuky matematiky na různých úrovních vzdělávacího systému. Výběr nástrojů byl proveden s cílem poskytnout komplexní přehled dostupných možností od volně šiřitelných až po komerční produkty, které pokrývají široké spektrum matematických disciplín a pedagogických přístupů.

Analyzované nástroje jsou rozčleněny do logických skupin podle jejich zaměření a funkcionality. Vědecké výpočetní systémy zahrnují GNU Octave a Scilab – open-source alternativy ke komerčnímu MATLABu, které se zaměřují na numerické výpočty a maticové operace. V oblasti dynamických geometrických systémů je představena aplikace GeoGebra, která propojuje geometrické konstrukce s algebraickými výrazy. Mezi systémy pro symbolickou matematiku jsou zařazeny open-source knihovna SymPy pro Python a komerční systémy Maple a Wolfram Mathematica, které umožňují manipulaci s matematickými výrazy v symbolické formě. Dále jsou analyzovány moderní mobilní aplikace a online nástroje, konkrétně Photomath a Microsoft Math Solver, které využívají technologie rozpoznávání obrazu pro řešení matematických úloh. Samostatná pozornost je věnována programovacímu jazyku Python a jeho matematickým knihovnám NumPy, SciPy a Matplotlib, které tvoří základ pro praktickou část této práce.

Tyto nástroje byly vybrány na základě několika kritérií: dostupnosti pro vzdělávací instituce i studenty, pedagogického potenciálu při vysvětlování abstraktních matematických konceptů, míry pokrytí různých oblastí matematiky od základní aritmetiky až po pokročilý kalkulus a schopnosti integrovat se do současných výukových metod. Zvláštní pozornost je věnována porovnání možností a limitací open-source a komerčních řešení, jejich uživatelské přívětivosti pro různé věkové skupiny studentů a schopnosti podporovat aktivní formy učení prostřednictvím interaktivních vizualizací a experimentování s matematickými koncepty.

Každý z analyzovaných nástrojů je představen z hlediska historie a vývoje, základních charakteristik a funkcí, uživatelského rozhraní a možností využití ve výuce. Takto strukturovaný přístup umožňuje porovnat jednotlivé nástroje podle standardizovaných kritérií a poskytuje pedagogům komplexní základ pro výběr vhodného softwaru podle specifických vzdělávacích cílů, technických podmínek a potřeb studentů.

Analýza těchto nástrojů tvoří teoretickou základnu pro praktickou část práce, která se zaměřuje na tvorbu výukových materiálů pro Python a jeho matematické knihovny. Tyto knihovny byly vybrány pro praktickou část vzhledem k jejich rostoucí popularitě ve vědecké komunitě, otevřenosti, dostupnosti a širokým možnostem využití od základních matematických operací až po pokročilé vědecké výpočty a vizualizace.

2 VĚDECKÉ VÝPOČETNÍ SYSTÉMY

Vědecké výpočetní systémy představují kategorii matematického softwaru, který je primárně zaměřen na numerické výpočty, analýzu dat a vědecké modelování. Tyto systémy poskytují vysokou výpočetní výkonnost a široké spektrum matematických funkcí. V této části se zaměříme na dva open-source zástupce této kategorie: GNU Octave a Scilab.

2.1 GNU Octave

GNU Octave je výkonný vysokoúrovňový programovací jazyk primárně určený pro numerické výpočty. Jedná se o open-source software, který je dostupný na většině operačních systémů a je vyvíjen jako volná alternativa ke komerčnímu softwaru MATLAB. [1]

2.1.1 Historie a vývoj

Projekt GNU Octave byl zahájen v roce 1988 Johnem W. Eadem jako součást kurzu chemického inženýrství na University of Texas. Původním cílem bylo vytvořit nástroj pro návrh chemických reaktorů. První veřejná verze GNU Octave byla vydána v roce 1993 a od té doby prošel software značným vývojem. V roce 2011 byla vydána verze 3.4.0, která přinesla významné vylepšení uživatelského rozhraní a grafických možností. Aktuální verze softwaru nabízí téměř kompletní kompatibilitu s MATLABem a rozsáhlé možnosti rozšíření prostřednictvím balíčků.

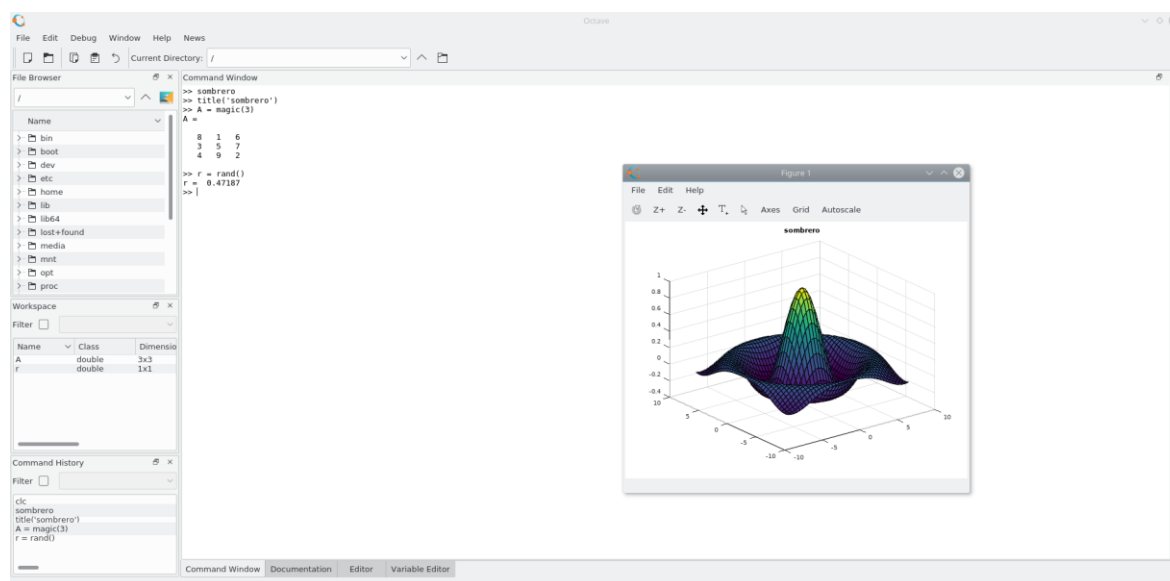
2.1.2 Základní charakteristiky a funkce

GNU Octave nabízí řadu funkcí a vlastností, které z něj činí vhodný nástroj pro vědecké výpočty a výuku matematiky. Software používá maticově orientovaný jazyk podobný MATLABu a poskytuje širokou knihovnu matematických funkcí pro lineární algebru, statistiku, numerickou integraci a řešení diferenciálních rovnic. [23]

Jako interpretovaný jazyk umožňuje Octave interaktivní práci a rychlé prototypování algoritmů bez nutnosti kompilace. Software nabízí pokročilé grafické možnosti pro vizualizaci dat včetně 2D a 3D grafů a podporuje rozšiřitelnost prostřednictvím vlastních funkcí a balíčků. Významnou výhodou je vysoká kompatibilita s MATLABem a multiplatformní dostupnost pro Windows, macOS, Linux a další operační systémy. [1][6]

2.1.3 Uživatelské rozhraní

GNU Octave nabízí jak příkazovou řádku, tak grafické uživatelské rozhraní (GUI), které bylo představeno ve verzi 3.8.0. GUI poskytuje editor kódu, správce proměnných, prohlížeč souborů a další nástroje, které usnadňují práci zejména začátečníkům (jak je ukázáno na obrázku 1). Příkazová řádka nabízí výhody pro pokročilé uživatele, včetně možnosti skriptování a automatizace. [6][23]



Obrázek 1 Uživatelské prostředí programu GNU Octave

(zdroj: vlastní)

2.1.4 Možnosti využití ve výuce

GNU Octave je vhodný nástroj pro výuku matematiky, hlavně na vysokých školách, protože je zdarma, výkonný a podobný MATLABu. Lze ho používat v mnoha oblastech: v lineární algebře pro řešení rovnic, práci s maticemi a výpočty vlastních čísel. V numerických metodách pro integraci, řešení diferenciálních rovnic a aproximace. Ve statistice pro různé výpočty, regresní analýzu a zpracování signálů. Pro modelování a simulace fyzikálních, chemických, biologických i ekonomických systémů. A pro vizualizaci dat - tvorbu grafů, které pomáhají pochopit matematické vztahy.

Největší výhodou GNU Octave pro školy je, že je zdarma. Studenti si ho mohou nainstalovat doma i ve škole bez placení licencí.

2.2 Scilab

Scilab je další významný open-source software pro numerické výpočty, který poskytuje výkonné matematické funkce a grafické nástroje. Podobně jako GNU Octave představuje alternativu ke komerčnímu MATLABu, ale s určitými specifickými vlastnostmi a zaměřením. [26][27]

2.2.1 Historie a vývoj

Scilab byl vyvinut v roce 1990 vědci z INRIA (Institut National de Recherche en Informatique et en Automatique) a ENPC (École Nationale des Ponts et Chaussées) ve Francii. V roce 2003 byl vytvořen Scilab Consortium pro podporu a další vývoj softwaru. Od roku 2008 je Scilab vyvíjen Scilab Enterprises a je distribuován pod open-source licencí CeCILL, která je kompatibilní s GNU GPL.

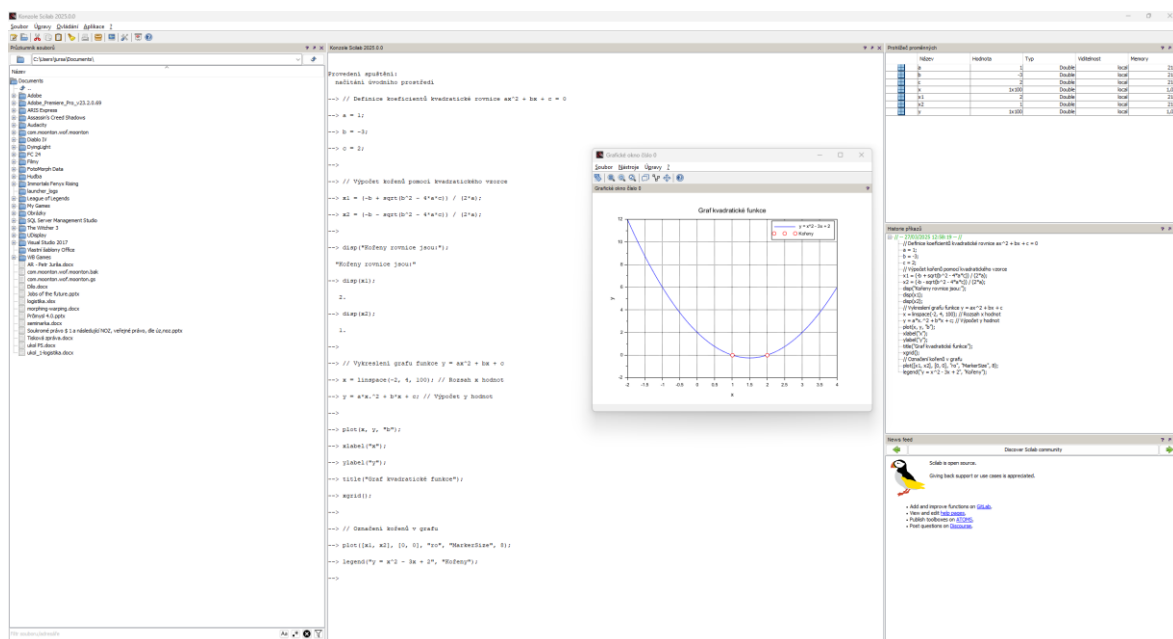
2.2.2 Základní charakteristiky a funkce

Scilab nabízí komplexní sadu nástrojů a funkcí pro vědecké a inženýrské výpočty. Software využívá vysokoúrovňový programovací jazyk podobný Octave a MATLABu, avšak s určitými syntaktickými rozdíly. Obsahuje rozsáhlou knihovnu funkcí pro lineární algebru, optimalizaci, statistiku a zpracování signálů. Významnou součástí je Xcos - integrované grafické prostředí pro modelování a simulaci dynamických systémů, které je podobné Simulinku v MATLABu. Pro správu rozšíření slouží systém ATOMS umožňující instalaci balíčků.[24][28]

Software nabízí pokročilé grafické nástroje pro vizualizaci dat včetně 2D a 3D grafů a specializované funkce pro analýzu signálů a obrazu. Scilab je multiplatformní s dostupností pro Windows, macOS a Linux. [2][26]

2.2.3 Uživatelské rozhraní

Program nabízí moderní grafické uživatelské rozhraní, které zahrnuje editor kódu, správce proměnných, prohlížeč nápovědy a další nástroje. GUI je intuitivní a uživatelsky přívětivé, což usnadňuje učení a používání softwaru (jak je vidět na obrázku 2). Scilab také nabízí příkazový řádek pro interaktivní práci a skriptování. Významnou součástí je Xcos, grafické prostředí pro modelování a simulaci dynamických systémů. Xcos umožňuje vytvářet bloková schémata a simulovat chování komplexních systémů bez nutnosti psát kód, což je zvláště užitečné pro výuku inženýrských disciplín. [2][28]



Obrázek 2 Uživatelské prostředí programu Scilab

(zdroj: vlastní)

2.2.4 Možnosti využití ve výuce

Scilab nabízí široké možnosti využití ve výukovém procesu, zejména v oblastech numerické matematiky (řešení rovnic, integrace, diferenciální rovnice), lineární algebry (maticové operace, řešení soustav lineárních rovnic), modelování a simulace s využitím prostředí Xcos, zpracování signálů (analýza a zpracování signálů, filtrace), optimalizace (řešení optimalizačních problémů) a statistiky (statistické výpočty a vizualizace). Scilab je obzvláště vhodný pro výuku díky své dostupnosti, uživatelské přívětivosti a komplexní dokumentaci v různých jazycích. Pro francouzsky mluvící země může být Scilab preferovanou volbou vzhledem k jeho francouzskému původu a dostupné dokumentaci ve francouzštině. Významnou výhodou pro vzdělávací instituce je také oficiální publikace "Scilab for very beginners" (Scilab pro úplné začátečníky), která poskytuje jednoduchý úvod do práce s tímto softwarem. [2][27]

2.3 Porovnání s komerčními alternativami (MATLAB)

MATLAB (Matrix Laboratory) je komerční software vyvinutý společností MathWorks, který představuje průmyslový standard pro vědecké výpočty v akademickém prostředí i průmyslu. V porovnání s open-source alternativami GNU Octave a Scilab nabízí MATLAB několik výhod i nevýhod. Co se týče kompatibility, GNU Octave je navržen tak, aby byl maximálně kompatibilní s MATLABem, přičemž většina kódu funguje bez úprav, zatímco

Scilab má odlišnou syntaxi a vyžaduje konverzi kódu. MATLAB poskytuje širší škálu specializovaných toolboxů pro různé aplikační oblasti než jeho alternativy, které sice nabízejí podobné funkce prostřednictvím balíčků, ale s méně komplexním pokrytím. V oblasti grafických nástrojů a GUI aplikací je MATLAB pokročilejší, a pro simulaci dynamických systémů nabízí Simulink, zatímco Scilab poskytuje podobnou funkčnost přes Xcos a GNU Octave nemá přímý ekvivalent. [7][29]

Z hlediska výkonu je MATLAB obecně považován za nejspolehlivější, zejména při práci s velkými datovými soubory, ačkoliv rozdíly se v novějších verzích open-source alternativ postupně zmenšují. Pro vzdělávací instituce je rozhodujícím faktorem cena - MATLAB vyžaduje nákladné licence (i přes zvýhodněné akademické verze), zatímco GNU Octave a Scilab jsou zdarma, což umožňuje neomezené používání ve školách i doma. Pro základní výuku numerických metod a lineární algebry jsou open-source alternativy plně dostačující, zatímco pro pokročilé specializované aplikace může být MATLAB vhodnější. Volba mezi GNU Octave a Scilab závisí na konkrétních požadavcích - Octave je lepší volbou pro přípravu studentů na práci s MATLABem díky své kompatibilitě, zatímco Scilab může být vhodnější pro výuku zahrnující modelování dynamických systémů díky prostředí Xcos. [7][24]

3 DYNAMICKÉ GEOMETRICKÉ SYSTÉMY

Dynamické geometrické systémy představují kategorii matematického softwaru, který umožňuje interaktivní práci s geometrickými objekty. Tyto systémy jsou zvláště cenné pro výuku geometrie, protože umožňují studentům experimentovat s geometrickými konstrukcemi a vizualizovat geometrické vztahy. V této části se zaměříme na GeoGebra, jeden z nejpopulárnějších dynamických geometrických systémů, který kombinuje geometrii, algebru, statistiku a další oblasti matematiky.

3.1 GeoGebra

GeoGebra je dynamický matematický software, který propojuje geometrii, algebru, tabulkové procesory, grafy, statistiku a infinitezimální počet. Tento software je vyvíjen mezinárodním týmem pod vedením Markuse Hohenwartera a je dostupný jako open-source.

3.1.1 Historie a vývoj

GeoGebra vznikla v roce 2001 jako součást diplomové práce Markuse Hohenwartera na univerzitě v Salzburgu. Původním cílem bylo vytvořit software, který by kombinoval dynamickou geometrii a algebraické vyjádření, což byl v té době inovativní přístup. První veřejná verze byla vydána v roce 2002. Od té doby prošla významným vývojem. V roce 2006 byl spuštěn GeoGebra Institute, který podporuje vývoj softwaru a poskytuje vzdělávací zdroje. V průběhu let byly do integrovány další matematické oblasti, včetně 3D geometrie, pravděpodobnosti, statistiky a kalkulu. V současnosti je dostupná nejen jako desktopová aplikace, ale také jako webová aplikace a mobilní aplikace pro různé platformy. [30]

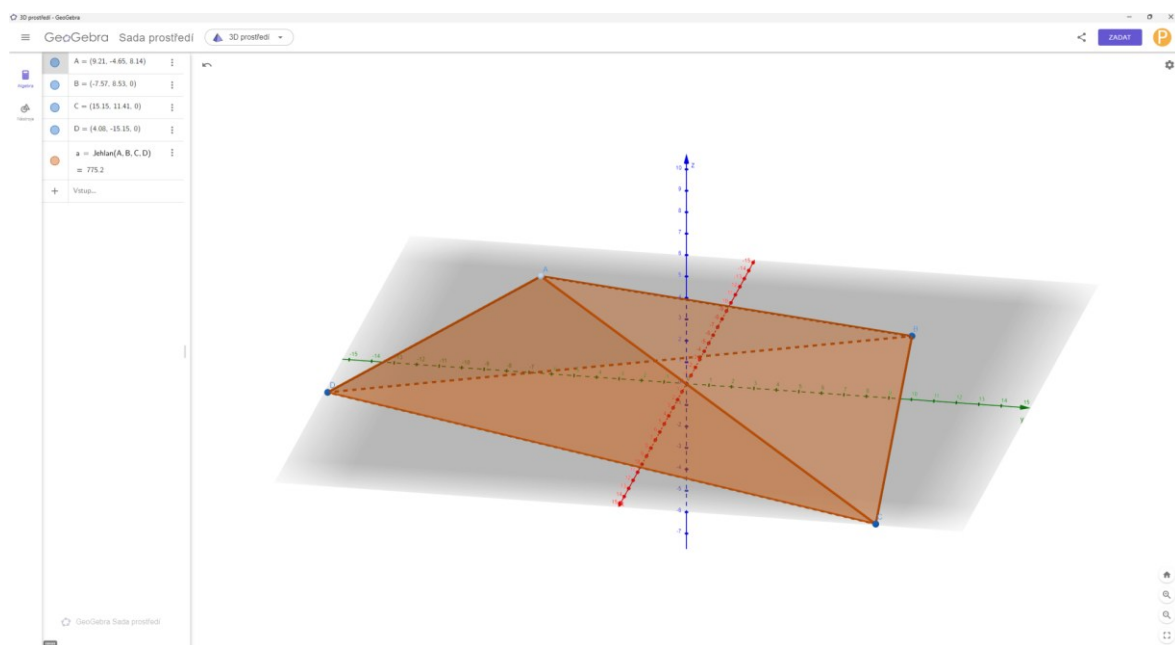
3.1.2 Základní charakteristiky a funkce

GeoGebra nabízí širokou škálu funkcí, které z ní činí univerzální nástroj pro výuku matematiky: dynamickou geometrii umožňující vytvářet geometrické konstrukce s body, vektory, úsečkami, přímkami, kuželosečkami a funkcemi, které lze následně dynamicky měnit; algebraický vstup, kdy všechny objekty lze zadávat a modifikovat přímým zadáním algebraických výrazů; integrovaný tabulkový procesor umožňující práci s daty a jejich vizualizaci; CAS (Computer Algebra System) - systém pro symbolické výpočty umožňující práci s proměnnými, polynomy, derivacemi, integrály a dalšími matematickými výrazy; 3D geometrii, kdy od verze 5.0 nabízí GeoGebra možnosti pro práci s 3D objekty a konstrukcemi; nástroje pro práci s pravděpodobnostními distribucemi, statistickými testy a

vizualizací dat; možnost vykreslovat a analyzovat funkce jedné i více proměnných; a rozsáhlou knihovnu předpřipravených interaktivních materiálů a aplikací. [30][31]

3.1.3 Uživatelské rozhraní

GeoGebra nabízí intuitivní a přizpůsobitelné uživatelské rozhraní, které lze konfigurovat podle potřeb uživatele a konkrétní matematické oblasti (ukázka uživatelského prostředí je na obrázku 3). Základními komponentami rozhraní jsou: náčrtna pro grafické znázornění geometrických objektů; algebraické okno zobrazující algebraické vyjádření objektů vytvořených v náčrtně; vstupní řádek pro přímé zadávání příkazů a algebraických výrazů; tabulkové okno pro práci s daty ve formátu tabulkového procesoru; CAS okno pro symbolické výpočty; a 3D náčrtna pro práci s trojrozměrnými objekty. Rozhraní je navrženo tak, aby bylo přístupné i začátečníkům, zároveň však poskytuje pokročilé možnosti pro zkušené uživatele. Významnou vlastností je možnost skrýt různé komponenty rozhraní podle potřeby, což umožňuje přizpůsobit prostředí různým věkovým skupinám a úrovním znalostí. [3][30]



Obrázek 3 Uživatelské prostředí programu GeoGebra

(zdroj: vlastní)

3.1.4 Možnosti využití ve výuce

GeoGebra je velmi užitečný nástroj pro výuku matematiky na všech stupních škol - od základních až po vysoké. Lze ji použít v mnoha oblastech: v základní geometrii pro kreslení

geometrických útvarů, zkoumání jejich vlastností a dokazování vět. V analytické geometrii pomáhá s rovnicemi přímek a kružnic a s transformacemi. V algebře vizualizuje funkce a pomáhá řešit rovnice. Ve vyšší matematice ukazuje derivace, integrály a limity. Ve statistice simuluje náhodné jevy a vytváří různé grafy. Umí také trojrozměrnou geometrii. Největší výhodou GeoGebry je, že dokáže ukázat, jak spolu souvisí různé způsoby vyjádření matematiky - geometrický obrázek, algebraický vzorec a číselné hodnoty. Když změníte jednu věc, ostatní se automaticky přizpůsobí. To pomáhá studentům lépe pochopit matematické pojmy a jejich vzájemné vztahy. [10][31]

GeoGebra podporuje sdílení výukových materiálů a spolupráci mezi učiteli i studenty prostřednictvím GeoGebra Materials Platform - online platformy s tisíci volně dostupnými výukovými materiály vytvořenými učiteli z celého světa, možnosti exportu konstrukcí do různých formátů pro sdílení nebo začlenění do jiných výukových materiálů a spolupráce studentů na společných projektech s možností online sdílení konstrukcí. [3][10]

4 SYSTÉMY PRO SYMBOLICKOU MATEMATIKU

Systémy pro symbolickou matematiku (CAS - Computer Algebra Systems) představují kategorii matematického softwaru, který umožňuje manipulaci s matematickými výrazy v symbolické formě. Na rozdíl od numerických systémů, které pracují s přibližnými číselnými hodnotami, mohou CAS provádět algebraické operace, derivace, integrály a další symbolické manipulace v přesném tvaru. V této části se zaměříme na SymPy, open-source knihovnu pro symbolickou matematiku implementovanou v Pythonu.

4.1 SymPy

SymPy je open-source knihovna pro symbolickou matematiku napsaná v programovacím jazyce Python. Poskytuje komplexní sadu nástrojů pro řešení problémů v různých oblastech matematiky, od základní algebry až po pokročilý kalkulus a diferenciální rovnice.

4.1.1 Historie a vývoj

SymPy byl založen v roce 2006 Ondřejem Čertíkem jako hobby projekt. Od té doby se rozvinul v komplexní knihovnu s aktivní komunitou vývojářů a uživatelů. Projekt je vyvíjen jako open-source pod licencí BSD, což umožňuje jeho svobodné používání, úpravy a distribuci. Důležitým aspektem vývoje SymPy je jeho integrace se systémem Pythonu pro vědecké výpočty, včetně knihoven jako NumPy, SciPy, matplotlib a Jupyter. Tato integrace umožňuje kombinovat symbolické výpočty s numerickými metodami, analýzou dat a vizualizací v jednotném prostředí. [33]

4.1.2 Základní charakteristiky a funkce

SymPy umí mnoho věcí v symbolické matematice. Zvládá základní algebraické operace - zjednodušuje výrazy, dosazuje hodnoty, rozkládá na součin a pracuje s polynomy. Řeší rovnice, soustavy rovnic a nerovnice symbolicky. Z vyšší matematiky umí derivace, integrály, limity a Taylorovy řady. Pracuje s maticemi a vektory, počítá determinanty a vlastní čísla. Řeší diferenciální rovnice. Z diskrétní matematiky zvládá kombinatoriku, teorii grafů a posloupnosti. Umí geometrii - práci s body, přímkami a kruhy. Má funkce pro fyziku včetně jednotek, vektorů a mechaniky. Obsahuje logiku s Booleovou algebrou. Nakonec má speciální matematické funkce jako gamma funkci nebo Besselovy funkce. [4][32]

4.1.3 Integrace s Pythonem a dalšími nástroji

Hlavní výhodou SymPy je, že funguje s Pythonem, který je stále oblíbenější pro vědecké výpočty a práci s daty. Toto propojení má několik výhod: v jednom prostředí lze kombinovat symbolické výpočty s číselnou matematikou, analýzou dat a strojovým učením. Jupyter Notebook umožňuje vytvářet dokumenty, které spojují kód, výsledky, texty a obrázky. Snadno se propojuje s knihovnami pro tvorbu grafů, jako je matplotlib, takže lze matematické výsledky hned zobrazit. Navíc si uživatelé mohou vytvořit vlastní funkce a rozšíření podle svých potřeb. [34][33]

4.1.4 Uživatelské rozhraní

SymPy je primárně používán jako knihovna v Pythonu, což znamená, že nemá samostatné grafické uživatelské rozhraní. Uživatelé pracují se SymPy prostřednictvím Pythonu, buď v interaktivním režimu, ve skriptech nebo v prostředí Jupyter Notebook. Toto může představovat určitou bariéru pro začátečníky neznalé programování, ale zároveň poskytuje velkou flexibilitu a možnosti integrace s dalšími nástroji. Pro uživatele, kteří preferují grafické rozhraní, existují alternativy jako SymPy Gamma - webová aplikace inspirovaná WolframAlpha, která umožňuje jednoduché symbolické výpočty bez nutnosti programování; SymPy Live Shell - interaktivní webové rozhraní pro práci s SymPy; nebo Jupyter Notebook s nainstalovaným SymPy poskytující interaktivní prostředí s možností vizualizace výsledků. [4][32]

4.1.5 Možnosti využití ve výuce

SymPy nabízí řadu možností pro využití ve výuce matematiky, zejména na vysokoškolské úrovni, v oblastech algebry a rovnic (zjednodušování výrazů, řešení rovnic, práce s polynomy), kalkulu (vizualizace a symbolické výpočty derivací, integrálů, limit a řad), lineární algebry (maticové operace, řešení soustav lineárních rovnic, vlastní čísla a vektory), diferenciálních rovnic (analytické řešení diferenciálních rovnic, vizualizace řešení) a diskrétní matematiky (kombinatorické výpočty, operace s množinami). SymPy může být obzvláště užitečný v kurzech, které kombinují matematiku s programováním nebo datovou vědou. Studenti se tak učí nejen matematické koncepty, ale zároveň získávají dovednosti v programování a využívání moderních nástrojů pro vědecké výpočty.

4.2 Komerční systémy pro symbolickou matematiku

Vedle open-source řešení existují na trhu také vysoce výkonné komerční systémy pro symbolickou matematiku. Tyto systémy nabízejí komplexní řešení pro vědecké výpočty, matematické modelování a vizualizaci dat. V této části se zaměříme na dva významné představitele této kategorie: Maple a Wolfram Mathematica.

4.3 Maple

Maple je komerční matematický software vyvinutý společností Maplesoft, který poskytuje prostředí pro symbolické a numerické výpočty s pokročilými možnostmi vizualizace a dokumentace.

4.3.1 Historie a vývoj

Maple byl původně vyvinut na University of Waterloo v Kanadě v roce 1980. První komerční verze byla vydána v roce 1984. Od té doby prošel software značným vývojem. V roce 2003 byla vydána verze Maple 9, která přinesla významné vylepšení uživatelského rozhraní a grafických možností. Současné verze nabízejí komplexní řešení pro symbolické výpočty, vědecké modelování a technické výpočty. Software je pravidelně aktualizován, přičemž každá nová verze přináší rozšíření funkcionality a optimalizaci výkonu.

4.3.2 Základní charakteristiky a funkce

Maple nabízí širokou škálu funkcí pro matematické výpočty a modelování. Software poskytuje výkonné nástroje pro symbolické výpočty, které umožňují manipulaci s algebraickými výrazy, řešení rovnic, derivace, integrály a další operace v symbolické formě. Pro numericky náročné úlohy obsahuje pokročilé numerické metody zahrnující výpočty s vysokou přesností, řešení soustav rovnic a optimalizační algoritmy. Z hlediska vizualizace dat nabízí Maple možnosti tvorby pokročilých 2D a 3D grafů, animací a interaktivních vizualizací. [13]

Software disponuje vlastním programovacím jazykem, který umožňuje tvorbu algoritmů a rozšíření funkcionality podle specifických potřeb uživatelů. Součástí Maple je také MapleSim, integrované prostředí pro modelování a simulaci fyzikálních systémů, které nachází uplatnění zejména v inženýrských aplikacích. Pro specializované oblasti výpočtů obsahuje Maple rozsáhlou knihovnu funkcí pokrývajících různé oblasti matematiky, fyziky a inženýrství. [13][35]

4.3.3 Uživatelské rozhraní

Maple nabízí několik režimů práce přizpůsobených různým potřebám uživatelů. Dokument režim představuje prostředí podobné textovému editoru, které umožňuje kombinaci textu, rovnic a grafů do ucelených dokumentů. Pro pokročilejší práci je k dispozici Worksheet režim, který poskytuje tradiční přístup s příkazovou řádkou. Pro interaktivní práci a tvorbu dynamických výpočtů nabízí Maple interaktivní komponenty jako tlačítka, posuvníky a další prvky. [36]

Toto flexibilní rozhraní umožňuje vytvářet komplexní dokumenty kombinující matematické výpočty, vysvětlující texty a vizualizace, což je zvláště užitečné pro výukové a výzkumné účely. [13][35]

4.3.4 Možnosti využití ve výuce

V oblasti vzdělávání Maple poskytuje několik specifických nástrojů. Pro zjednodušení výuky byla vyvinuta verze Maple Learn, která je určena primárně pro vzdělávací účely a nabízí intuitivnější rozhraní. Pro testování matematických znalostí studentů slouží systém Maple T.A. Software je také doplněn rozsáhlou knihovnou výukových materiálů obsahující příklady a cvičení z různých oblastí matematiky.

Pro vzdělávací instituce jsou dostupné zvýhodněné akademické licence, ačkoli náklady mohou být stále překážkou pro některé školy ve srovnání s open-source alternativami. Přes vyšší pořizovací cenu nabízí Maple komplexní řešení, které může významně obohatit výuku matematiky a příbuzných disciplín.

4.4 Wolfram Mathematica

Wolfram Mathematica je komplexní výpočetní software vyvinutý společností Wolfram Research. Představuje jeden z nejpokročilejších systémů pro technické výpočty, který integruje symbolické a numerické výpočty, vizualizaci a programování v jednotném prostředí.

4.4.1 Historie a vývoj

Mathematica byla poprvé vydána v roce 1988 Stephenem Wolframem, matematikem a fyzikem, který předtím pracoval na vývoji systému SMP (Symbolic Manipulation Program). Od té doby se software významně rozvinul a stal se jedním z nejpokročilejších nástrojů pro symbolické výpočty a vědecké modelování. Významným milníkem byl vznik služby

Wolfram Alpha v roce 2009, která zpřístupnila výpočetní schopnosti Mathematicy širší veřejnosti prostřednictvím přirozeného jazyka. Vývojový tým pravidelně přidává nové funkce a vylepšení, přičemž v současných verzích klade důraz na integraci metod umělé inteligence a strojového učení.

4.4.2 Základní charakteristiky a funkce

Mathematica nabízí mimořádně široké spektrum funkcí pro matematické výpočty a modelování. Jádrem systému je Wolfram Language, vlastní vysokoúrovňový programovací jazyk optimalizovaný pro symbolické výpočty a práci s daty. Software poskytuje pokročilé nástroje pro symbolické výpočty umožňující manipulaci s matematickými výrazy a formulami v přesném tvaru. Pro zpracování rozsáhlých výpočtů jsou k dispozici numerické metody s vysokou přesností a efektivitou. [14]

V oblasti datové analýzy nabízí Mathematica nástroje pro statistiku, zpracování dat a strojové učení. Vizualizační schopnosti zahrnují pokročilé možnosti pro 2D a 3D vizualizaci matematických objektů a dat. Významnou výhodou je integrace s Wolfram Alpha, která poskytuje přímý přístup k rozsáhlé znalostní databázi. Pro tvorbu interaktivních modelů a aplikací jsou k dispozici různé dynamické prvky, které umožňují vytvářet interaktivní vizualizace a simulace. [15][25]

4.4.3 Uživatelské rozhraní

Mathematica používá koncept tzv. "notebook" rozhraní, které integruje různé typy obsahu do jednoho dokumentu. Toto rozhraní zahrnuje buňky s kódem pro zadávání příkazů v jazyce Wolfram Language, textové buňky pro komentáře a vysvětlení, grafické buňky pro zobrazení grafů a vizualizací a dynamické prvky pro interaktivní modely a demonstrace. [14][37]

Tento přístup umožňuje vytvářet dokumenty, které kombinují výpočty, text a vizualizace v jednotném prostředí, což je užitečné pro vzdělávací i výzkumné účely. Uživatelské rozhraní také zahrnuje různé palety nástrojů, které usnadňují přístup k často používaným funkcím a symbolům bez nutnosti pamatovat si specifickou syntaxi. [15][25]

4.4.4 Možnosti využití ve výuce

Mathematica nabízí několik vzdělávacích nástrojů a zdrojů, které podporují její využití ve výuce matematiky a příbuzných předmětů. Wolfram Demonstrations Project představuje

rozsáhlou knihovnu interaktivních demonstrací matematických konceptů vytvořených komunitou uživatelů. Pro vzdělávací účely je vhodný formát Computational Essay, který kombinuje vysvětlující texty s výpočty a vizualizacemi. Učitelé mohou využívat zdroje z Wolfram Education Portal, který obsahuje plány hodin a vzdělávací aktivity.

I přes vyšší cenu komerčních licencí nabízí Wolfram Research zvýhodněné akademické licence a studentské verze, které zvyšují dostupnost pro vzdělávací účely. Pro mnohé vzdělávací instituce však mohou být licenční náklady stále významnou překážkou pro plošné nasazení. [37]

5 MOBILNÍ APLIKACE A ONLINE NÁSTROJE PRO VÝUKU

S rozvojem mobilních technologií a rozšířením chytrých telefonů a tabletů se otevřely nové možnosti pro výuku matematiky. Mobilní aplikace a online nástroje nabízejí flexibilní a dostupnou alternativu k tradičním softwarovým řešením. V této části se zaměříme na dvě populární aplikace: Photomath a Microsoft Math Solver.

5.1 Photomath

Photomath je mobilní aplikace, která umožňuje řešit matematické problémy pomocí fotoaparátu chytrého telefonu nebo tabletu. Aplikace dokáže rozpoznat matematické výrazy a rovnice z fotografie a poskytnout jejich řešení spolu s podrobným postupem.

5.1.1 Historie a vývoj

Photomath byl vyvinut chorvatskou společností Photomath, Inc. a poprvé uveden na trh v roce 2014. Aplikace rychle získala popularitu a v současnosti patří mezi nejstahovanější vzdělávací aplikace na světě s více než 250 miliony stažení. Aplikace je dostupná pro zařízení s operačními systémy iOS a Android. Od svého uvedení prošel Photomath významným vývojem. Původní verze byla zaměřena především na základní aritmetické operace a jednoduché rovnice, zatímco novější verze podporují širší spektrum matematických témat včetně lineární algebry, kalkulu a statistiky. Zlepšila se také přesnost rozpoznávání textu a kvalita poskytovaných vysvětlení.

5.1.2 Základní charakteristiky a funkce

Photomath má několik užitečných funkcí pro studenty. Umí rozpoznat matematické příklady z fotky nebo z ručně napsaného textu. Ukáže podrobný postup řešení s vysvětlením u většiny úloh. Má grafickou kalkulačku, která nakreslí graf funkce a pomůže pochopit její vlastnosti. S grafy lze manipulovat, což pomáhá lépe pochopit matematické vztahy. U některých příkladů ukáže více způsobů řešení, takže studenti poznají různé postupy. Obsahuje také vědeckou kalkulačku pro základní i složité výpočty. Výhodou je, že základní funkce fungují i bez internetu. [5]

5.1.3 Oblasti matematiky pokryté aplikací

Photomath pokrývá široké spektrum matematických témat, včetně aritmetiky (základní operace, zlomky, desetinná čísla, procenta), algebry (rovnice, nerovnice, soustavy rovnic, polynomy, funkce), trigonometrie (trigonometrické funkce, identity, rovnice), kalkulu

(limity, derivace, integrály, řady), lineární algebry (matice, determinanty, vektorové prostory) a statistiky (základní statistické výpočty, pravděpodobnost). Uživatelské prostředí aplikace je zobrazeno na obrázku 4. [5][11]



Obrázek 4 Prostředí aplikace Photomath

(zdroj: vlastní)

5.1.4 Vzdělávací aspekty a kontroverze

Photomath a podobné aplikace vyvolávají v pedagogické komunitě diskuse o jejich roli ve vzdělávacím procesu. Na jedné straně mohou tyto nástroje pomoci studentům lépe pochopit matematické koncepty a poskytnout okamžitou zpětnou vazbu. Na druhé straně existují obavy, že by studenti mohli aplikaci používat k pouhému opisování výsledků bez skutečného porozumění. Podle studie Kelecsényi, Osztényiné a Diána (2021) je Photomath často využíván studenty jako nástroj pro kontrolu svých výsledků nebo jako pomůcka při samostudiu. Studie však také upozorňuje na riziko vytvoření závislosti na aplikaci možné snížení schopnosti studentů řešit problémy samostatně. [5]

5.2 Microsoft Math Solver

Microsoft Math Solver je webová a mobilní aplikace vyvinutá společností Microsoft, která poskytuje řešení různých matematických problémů. Podobně jako Photomath nabízí možnost skenování příkladů pomocí fotoaparátu, ale také umožňuje zadávání příkladů pomocí klávesnice nebo kreslení rukou.

5.2.1 Historie a vývoj

Microsoft Math Solver byl představen v roce 2019 jako nástupce dřívější aplikace Microsoft Mathematics, která byla dostupná pro Windows. Na rozdíl od svého předchůdce je Math Solver navržen jako multiplatformní nástroj dostupný prostřednictvím webového prohlížeče a mobilních aplikací pro iOS a Android. Aplikace vznikla jako součást vzdělávacích iniciativ Microsoftu a využívá pokročilé technologie umělé inteligence a počítačového vidění. Od svého uvedení byla aplikace průběžně vylepšována, zejména v oblasti rozpoznávání ručně psaných matematických výrazů a poskytování podrobnějších vysvětlení řešení. [11]

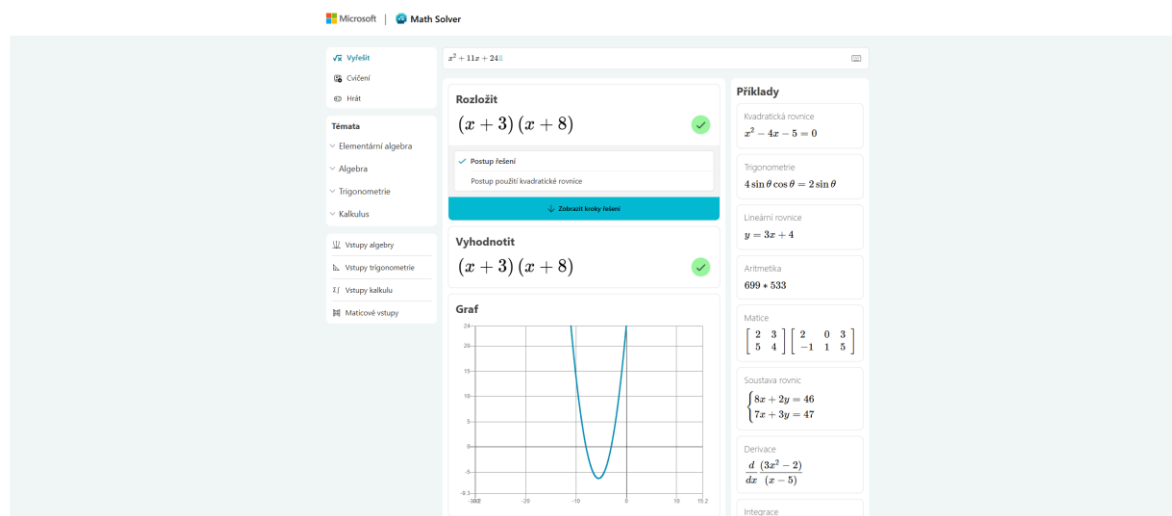
5.2.2 Základní charakteristiky a funkce

Microsoft Math Solver funguje podobně jako jiné matematické aplikace, ale má několik zvláštních funkcí. Příklady lze zadat třemi způsoby - vyfotit, nakreslit prstem na obrazovku nebo napsat na klávesnici. Aplikace ukáže podrobný postup řešení s vysvětlením, aby studenti pochopili, jak se úloha řeší. Umí také nakreslit graf funkce, se kterým lze pracovat. Navrhne podobné příklady pro procvičení a ukáže výuková videa od známých vzdělávacích společností jako Khan Academy. Zajímavé je, že často ukáže více způsobů, jak stejný příklad vyřešit, takže studenti poznají různé postupy. [11][12]

5.2.3 Uživatelské rozhraní

Math Solver nabízí čisté a intuitivní uživatelské rozhraní, které je navrženo s důrazem na snadné použití pro studenty (jak je znázorněno na obrázku 5). Webová i mobilní verze sdílí podobný design, což usnadňuje přechod mezi různými platformami. Hlavní obrazovka nabízí tři možnosti zadání příkladu - fotoaparát, kreslení nebo psaní. Po zadání příkladu aplikace zobrazí výsledek a nabídne podrobné řešení s vysvětleními. Uživatelské rozhraní je rozděleno do logických sekcí, včetně výsledku, krokového řešení, grafu (pokud je relevantní) a souvisejících výukových materiálů. Design je přizpůsoben pro použití ve vzdělávacím prostředí, s jasnými popisky a přehlednými vysvětleními. Aplikace také nabízí

tmavý režim, který může být příjemnější pro dlouhodobé používání, zejména v podmínkách s nižším osvětlením. [12]



Obrázek 5 Uživatelské prostředí aplikace Microsoft Math Solver

(zdroj: vlastní)

5.2.4 Oblasti matematiky pokryté aplikací

Microsoft Math Solver umí řešit širokou škálu matematických úloh pro studenty od základní školy až po vysokou školu. Zvládá základní počítání se zlomky, desetinnými čísly a procenty. Dále řeší jednoduché i složitější rovnice, soustavy rovnic, nerovnice a práci s polynomy. Z pokročilejších témat umí kvadratické funkce, logaritmy a exponenciální rovnice. Pomůže také s trigonometrií - trigonometrickými funkcemi a rovnicemi. Pro vysokoškoláky zvládá limity, derivace, integrály a diferenciální rovnice. Nakonec umí i základní statistiku a pravděpodobnost. [12]

6 PYTHON A JEHO MATEMATICKÉ KNIHOVNY

Python představuje jeden z nejpopulárnějších programovacích jazyků současnosti, který nachází široké uplatnění v oblasti vědeckých výpočtů, analýzy dat a strojového učení. Díky své jednoduchosti, čitelnosti a bohatému systému knihoven se Python stal preferovanou volbou pro výuku programování i pro profesionální aplikace v oblasti matematiky, přírodních věd a inženýrství. V této části se zaměříme na Python a jeho klíčové matematické knihovny – NumPy, SciPy a Matplotlib.

6.1 Historie a vývoj Pythonu

Python byl vytvořen Guido van Rossumem a první verze byla vydána v roce 1991. Původním záměrem bylo vytvořit jazyk, který by kombinoval čitelnost a jednoduchost s výkonností a univerzálností. Od té doby prošel Python významným vývojem, přičemž v současnosti se používá Python 3.13, který přinesl mnoho vylepšení zejména v oblasti práce s textem, internacionalizace a celkové konzistence jazyka.

Klíčovým milníkem pro uplatnění Pythonu ve vědeckých výpočtech bylo vytvoření knihovny NumPy v roce 2006, která výrazně rozšířila možnosti jazyka pro práci s numerickými daty. Následovaly další specializované knihovny jako SciPy, Matplotlib, Pandas a mnohé další, které společně vytvořily robustní systém pro vědecké výpočty a datovou analýzu. [9][16]

6.2 NumPy

NumPy (Numerical Python) představuje základní knihovnu pro vědecké výpočty v Pythonu. Tato knihovna poskytuje podporu pro práci s vícerozměrnými poli a maticemi, spolu s rozsáhlou kolekcí vysokoúrovňových matematických funkcí pro operace s těmito datovými strukturami. [17]

6.2.1 Základní charakteristiky a funkce

Jádrem NumPy je objekt ndarray (n-dimensional array), který poskytuje efektivní datovou strukturu pro manipulaci s velkými poli dat a umožňuje provádět operace nad celými bloky dat bez nutnosti explicitních cyklů. Knihovna obsahuje rozsáhlou sadu matematických funkcí pro široké spektrum operací, včetně trigonometrických, statistických a algebraických funkcí.[8]

V oblasti lineární algebry poskytuje NumPy funkce pro operace s maticemi, řešení soustav lineárních rovnic, výpočet vlastních čísel a vektorů. Pro statistické aplikace a simulace nabízí knihovna generátory náhodných čísel s různými pravděpodobnostními distribucemi. NumPy obsahuje také implementaci rychlé Fourierovy transformace a pokročilé indexování pro sofistikované metody výběru a manipulace s částmi datových struktur. Důležitým aspektem NumPy je jeho optimalizace pro výkon. Kritické operace jsou implementovány v jazyce C a Fortran, což umožňuje efektivní práci s velkými objemy dat. Díky tomu je výrazně rychlejší než ekvivalentní kód napsaný v čistém Pythonu. [16][20]

6.2.2 Uživatelské rozhraní

NumPy nemá samostatné grafické uživatelské rozhraní, ale je typicky používán v rámci různých interaktivních prostředí a vývojových nástrojů. Nejčastěji se NumPy používá v rámci Jupyter notebooků, což jsou interaktivní dokumenty umožňující kombinovat kód, výstup, vizualizace a vysvětlující text. Jupyter notebook poskytuje ideální prostředí pro výuku a experimentování s NumPy, protože umožňuje okamžitou vizualizaci výsledků a interaktivní práci s kódem. [17][22]

6.2.3 Možnosti využití ve výuce

NumPy nabízí široké možnosti využití ve výuce matematiky a příbuzných disciplín. V oblasti lineární algebry umožňuje práci s vektory a maticemi, což podporuje názornou demonstraci konceptů jako maticové operace, determinanty, vlastní čísla a vektory nebo řešení soustav lineárních rovnic.[22]

Pro výuku numerické matematiky poskytuje NumPy nástroje pro implementaci různých numerických metod, jako je numerická integrace, řešení diferenciálních rovnic nebo interpolace. V oblasti statistiky umožňuje provádět statistické výpočty a analýzy dat, včetně výpočtu statistických charakteristik, generování náhodných vzorků a testování hypotéz. NumPy je také vhodný pro tvorbu matematických a fyzikálních simulací, které mohou pomoci studentům lépe pochopit abstraktní koncepty. Významnou výhodou NumPy pro vzdělávací účely je jeho dostupnost jako open-source software, což umožňuje studentům používat stejné nástroje doma i ve škole bez licenčních omezení. [20]

6.3 SciPy

SciPy (Scientific Python) rozšiřuje funkcionalitu NumPy o další specializované algoritmy a nástroje pro vědecké výpočty. SciPy je organizována do submodulů zaměřených na specifické oblasti vědeckých výpočtů. [18]

6.3.1 Základní charakteristiky a funkce

SciPy poskytuje širokou škálu funkcí pro vědecké výpočty a modelování, které jsou organizovány do tematických podbalíčků. Modul `scipy.special` obsahuje implementaci mnoha speciálních matematických funkcí, zatímco `scipy.integrate` nabízí nástroje pro numerickou integraci a řešení diferenciálních rovnic. V oblasti optimalizace poskytuje SciPy modul `scipy.optimize`, který zahrnuje algoritmy pro minimalizaci (nebo maximalizaci) funkcí. Pro interpolaci dat slouží modul `scipy.interpolate`, zatímco zpracování signálů a systémů je pokryto modulem `scipy.signal`. [16]

Modul `scipy.linalg` rozšiřuje možnosti NumPy v oblasti lineární algebry, zatímco `scipy.stats` obsahuje rozsáhlou sadu statistických funkcí a rozdělení. Pro práci s prostorovými daty poskytuje SciPy modul `scipy.spatial` a pro zpracování obrazových dat modul `scipy.ndimage`. SciPy je postavena na NumPy a rozšiřuje její funkcionalitu o algoritmy vyšší úrovně. Obě knihovny společně poskytují široké zázemí pro téměř jakoukoliv oblast vědeckých výpočtů. [20][22]

6.3.2 Uživatelské rozhraní

Podobně jako NumPy, i SciPy je primárně používána jako knihovna v rámci Pythonu a nemá vlastní samostatné grafické uživatelské rozhraní. Uživatelé pracují s přes Jupyter notebooky, vývojová prostředí nebo příkazovou řádku Python. Integrace s Jupyter notebooky je zvláště vhodná pro výukové účely, protože umožňuje kombinovat kód, vysvětlující text, matematické vzorce a vizualizace v jednom dokumentu. SciPy je navržena tak, aby poskytovala konzistentní a intuitivní API (Application Programming Interface), které usnadňuje její používání i pro začátečníky. Rozhraní SciPy je strukturováno do logických modulů podle oblastí aplikace, což usnadňuje orientaci a nalezení potřebných funkcí. [18]

6.3.3 Možnosti využití ve výuce

SciPy nabízí široké možnosti využití ve výuce matematiky a příbuzných disciplín, zejména na vysokoškolské úrovni. V oblasti numerických metod umožňuje implementaci a

experimentování s různými algoritmy, jako je numerická integrace, řešení diferenciálních rovnic, optimalizace nebo interpolace. Pro výuku aplikované matematiky poskytuje nástroje pro řešení reálných problémů z oblasti inženýrství, fyziky, ekonomie nebo biologie. V oblasti statistiky a analýzy dat nabízí nástroje pro statistickou analýzu, testování hypotéz a modelování, což je užitečné pro výuku statistiky a datové vědy.

SciPy je také vhodná pro výuku zpracování signálů a obrazů prostřednictvím příslušných modulů. V oblasti optimalizace umožňuje řešení různých optimalizačních úloh, od jednoduchých lineárních programů až po složité nelineární problémy. Zvláště je cenná pro propojení teoretických konceptů s praktickými aplikacemi. Studenti mohou implementovat a testovat matematické algoritmy na reálných datech a problémech, což podporuje hlubší porozumění a schopnost aplikovat matematické znalosti v praxi.

6.4 Matplotlib

Matplotlib je knihovna pro tvorbu vizualizací a grafů v Pythonu, která umožňuje vytvářet statické, interaktivní a animované vizualizace s publikační kvalitou v různých formátech. [19]

6.4.1 Základní charakteristiky a funkce

Matplotlib podporuje tvorbu různých typů grafů, včetně čárových, sloupcových, koláčových, bodových, histogramů a mnoha dalších. Knihovna nabízí flexibilní systém os a souřadnic, který umožňuje vytvářet jednoduché i složité grafické kompozice s mnoha osami. Pro profesionální využití nabízí Matplotlib pokročilé možnosti formátování všech aspektů grafů. Kromě dvourozměrných grafů podporuje Matplotlib také tvorbu trojrozměrných vizualizací a animací. Pro sdílení a publikování výsledků nabízí podporu pro ukládání grafů v různých formátech, včetně PNG, PDF, SVG a EPS. [19][21]

Matplotlib je navržena tak, aby byla kompatibilní s NumPy a SciPy, což umožňuje snadnou vizualizaci výsledků vědeckých výpočtů. Jako základní vizualizační nástroj v Pythonu slouží Matplotlib také jako základ pro další specializované vizualizační knihovny. [20][22]

6.4.2 Uživatelské rozhraní

Matplotlib nabízí dva hlavní přístupy k tvorbě vizualizací: procedurální ("pyplot") a objektově orientovaný. Procedurální přístup je inspirován MATLABem a poskytuje

jednoduché rozhraní pro rychlou tvorbu grafů. Objektově orientovaný přístup poskytuje větší kontrolu nad všemi aspekty vizualizace a je vhodnější pro složitější grafické kompozice.

Podobně jako NumPy a SciPy, i Matplotlib je často používán v rámci Jupyter notebooků, což umožňuje interaktivní tvorbu a úpravu vizualizací. V interaktivních prostředích nabízí Matplotlib možnosti jako posuvníky, tlačítka a další interaktivní prvky. [16][21]

6.4.3 Možnosti využití ve výuce

Matplotlib představuje cenný nástroj pro výuku matematiky a příbuzných disciplín, protože umožňuje studentům vizualizovat abstraktní koncepty. Vizualizace funkcí je jednou z klíčových aplikací, kde studenti mohou vykreslovat grafy funkcí, zkoumat jejich vlastnosti a vizualizovat koncepty jako limity, derivace nebo integrály.[21]

V oblasti statistiky umožňuje Matplotlib tvorbu různých statistických grafů, jako jsou histogramy, krabicové grafy nebo bodové grafy. Tyto vizualizace pomáhají studentům pochopit rozložení dat, identifikovat trendy a vztahy mezi proměnnými. Matplotlib lze také využít pro ilustraci matematických konceptů prostřednictvím ilustrací a animací. Pro studentské projekty a výzkum poskytuje nástroje pro prezentaci výsledků. Pomocí interaktivních prvků mohou studenti experimentovat s parametry funkcí nebo modelů a sledovat, jak se mění výsledky. Flexibilita a možnost přizpůsobení vizualizací činí Matplotlib vhodným nástrojem pro výuku na různých úrovních vzdělávání, od základních kurzů až po pokročilý výzkum.

6.5 Integrace s dalšími nástroji a knihovnami

Jednou z významných předností Pythonu a jeho matematických knihoven je jejich schopnost integrace s dalšími nástroji a knihovnami. Jupyter Notebook představuje interaktivní prostředí, které umožňuje kombinovat živý kód, rovnice, vizualizace a vysvětlující text v jednom dokumentu, což je mimořádně užitečné pro výuku a výzkum. JupyterLab rozšiřuje tuto funkcionalitu o další nástroje pro vědecké výpočty a analýzu dat.

Python a jeho základní matematické knihovny lze snadno integrovat s dalšími nástroji pro vědecké výpočty. Pandas je knihovna pro manipulaci a analýzu strukturovaných dat, SymPy poskytuje možnosti pro symbolickou matematiku, scikit-learn nabízí nástroje pro strojové učení a statsmodels slouží pro statistické modelování. Pro hluboké učení a

neuronové sítě jsou k dispozici knihovny TensorFlow a PyTorch. Tato integrace umožňuje vytvářet komplexní řešení, která kombinují různé přístupy a techniky, od symbolických výpočtů přes numerické metody až po strojové učení. [20][16]

6.6 Porovnání s dalšími nástroji pro matematické výpočty

Python a jeho matematické knihovny nabízejí řadu výhod ve srovnání s dalšími nástroji pro matematické výpočty, ale mají také určité limitace. K hlavním výhodám patří otevřenost a dostupnost (Python a jeho knihovny jsou open-source a zdarma), jednoduchost a čitelnost syntaxe, univerzálnost coby obecného programovacího jazyka, rozsáhlý systém knihoven a velká, aktivní komunita uživatelů a vývojářů. [16]

Limitace ve srovnání s komerčními nástroji zahrnují omezenější možnosti v oblasti symbolických výpočtů oproti systémům jako Maple nebo Mathematica, absenci jednotného integrovaného prostředí, méně uživatelsky přívětivou dokumentaci a menší počet předpřipravených specializovaných funkcí.

V porovnání s jinými open-source alternativami jako GNU Octave nebo Scilab nabízí Python větší flexibilitu a širší spektrum aplikací, zatímco tyto specializované nástroje mohou být přímočařejší pro specifické matematické úlohy, zejména pro uživatele zvyklé na syntax MATLABu. Z hlediska výkonu jsou NumPy a SciPy optimalizovány pro efektivitu a rychlost, což umožňuje jejich použití i pro náročné výpočty.

II PRAKTICKÁ ČÁST

7 VÝUKOVÉ MATERIÁLY PRO PYTHON A JEHO MATEMATICKÉ KNIHOVNY

V této kapitole je představena praktická část bakalářské práce zaměřená na tvorbu výukových materiálů pro Python a jeho matematické knihovny NumPy, SciPy a Matplotlib. Tyto materiály jsou navrženy tak, aby poskytly komplexní podporu pro výuku matematiky s využitím Pythonu na středních a vysokých školách. Cílem praktické části je vytvořit ucelený soubor výukových nástrojů, které budou reagovat na aktuální potřeby pedagogů a studentů v oblasti matematického vzdělávání s využitím moderních programovacích prostředků.

Na základě analýzy existujících výukových materiálů a zkušeností z pedagogické praxe byly identifikovány klíčové nedostatky v dostupnosti kvalitních českých výukových zdrojů pro Python v kontextu matematického vzdělávání. Zatímco v angličtině existuje rozsáhlá nabídka materiálů, v českém prostředí je jejich počet omezený, což představuje značnou bariéru zejména pro začínající uživatele a studenty středních škol. Tato práce si klade za cíl tuto mezeru částečně zaplnit vytvořením ucelené sady výukových materiálů v českém jazyce. Praktická část zahrnuje čtyři hlavní komponenty: sérii výukových prezentací pokrývajících základy Pythonu a jeho matematických knihoven, sadu demonstračních příkladů odstupňovaných podle obtížnosti, český manuál poskytující přehled funkcí a možností jednotlivých knihoven a dotazníkové šetření pro ověření vytvořených materiálů v praxi. Všechny komponenty jsou navrženy tak, aby společně tvořily komplexní výukový rámec, který může být flexibilně implementován do různých typů kurzů a přizpůsoben různým úrovním znalostí studentů.

Při tvorbě materiálů byl kladen důraz na několik klíčových principů. Prvním z nich je postupné budování znalostí, kdy materiály začínají od základních konceptů a postupně přecházejí ke složitějším tématům. Druhým principem je praktická orientace, kdy teoretické koncepty jsou vždy doplněny konkrétními příklady a aplikacemi. Třetím principem je interaktivita, která je podporována využitím Jupyter notebooků umožňujících studentům experimentovat s kódem a okamžitě vidět výsledky. Čtvrtým principem je vizualizace, kdy složitější matematické koncepty jsou doprovázeny grafickými reprezentacemi pro lepší pochopení.

Výsledky praktické části této práce mají potenciál přispět ke zkvalitnění výuky matematiky s využitím programování a podpořit rozvoj digitálních kompetencí studentů,

které jsou nejen stále důležitější na trhu práce, ale i klíčové pro rozvoj analytického a algoritmického myšlení.

7.1 Metodologie a cíle výukových materiálů

Vytvoření efektivních výukových materiálů pro oblast matematického programování vyžaduje systematický přístup, který reflektuje jak pedagogické principy, tak specifika technologického obsahu. V této podkapitole jsou popsány metodologické přístupy použité při tvorbě výukových materiálů a definovány hlavní cíle, které tyto materiály sledují.

7.1.1 Cílové skupiny a metodologické přístupy

Výukové materiály byly navrženy s ohledem na dvě hlavní cílové skupiny: studenty a pedagogy. Primární cílovou skupinou jsou studenti středních a vysokých škol, kteří se setkávají s matematickým softwarem v rámci výuky matematiky, informatiky nebo příbuzných technických předmětů. Tato skupina byla rozdělena podle úrovně znalostí na začátečníky, mírně pokročilé a pokročilé uživatele. Sekundární cílovou skupinou jsou pedagogové, kteří tyto materiály budou využívat ve své výuce. Pro tuto skupinu byly materiály doplněny metodickými poznámkami.

Při tvorbě výukových materiálů bylo aplikováno několik klíčových metodologických přístupů. Konstruktivistický přístup podporuje aktivní konstrukci znalostí na základě vlastní zkušenosti s kódem. Materiály využívají kontextualizaci matematických konceptů do reálných problémů a multimodální přístup kombinující různé formy reprezentace pro podporu různých učebních stylů.

7.1.2 Cíle a struktura výukových materiálů

Výukové materiály byly vytvořeny s cílem rozvinout programovací dovednosti v Pythonu, představit matematické knihovny NumPy, SciPy a Matplotlib a propojit matematickou teorii s praktickou implementací. Dalšími cíli bylo rozvíjet algoritmické myšlení, budovat kompetence v oblasti datové analýzy a vizualizace, podporovat samostatné učení a poskytnout českým pedagogům komplexní podklady pro výuku.

7.2 Výukové prezentace

V rámci praktické části bakalářské práce byly vytvořeny čtyři výukové prezentace, které pokrývají postupný přechod od základů programovacího jazyka Python až po pokročilé aplikace jeho matematických knihoven. Prezentace jsou koncipovány jako ucelený

vzdělávací systém zahrnující "Úvod do programovacího jazyka Python", "Základy práce s NumPy", "Vědecké výpočty s SciPy" a "Vizualizace dat s Matplotlib". Tato posloupnost respektuje přirozený tok učení a zajišťuje systematické osvojení nezbytných programovacích základů před přechodem k specializovaným matematickým nástrojům.

Všechny prezentace byly vytvořeny s důrazem na praktickou orientaci, kdy každý teoretický koncept je doprovázen konkrétními příklady kódu demonstrujícími jeho použití v reálných matematických situacích. Struktura každé prezentace respektuje pedagogické principy efektivního vzdělávání s motivačním úvodem objasňujícím praktický význam probírané tematiky, systematickým představením teoretických základů s důrazem na pochopení principů a praktickými částmi obsahujícími řešené příklady z různých oblastí matematiky. Významnou pozornost byla věnována propojení jednotlivých prezentací, kdy NumPy je představen jako základ pro práci s numerickými daty, na kterém staví pokročilejší funkce SciPy, zatímco Matplotlib slouží pro vizualizaci výsledků získaných pomocí obou předchozích knihoven.

Prezentace byly navrženy s ohledem na flexibilitu použití v různých vzdělávacích kontextech včetně tradičního frontálního vyučování, seminárních forem výuky nebo samostudia, přičemž modulární struktura umožňuje pedagogům přizpůsobit konkrétní části podle specifických potřeb jejich kurzu. Každá prezentace je doplněna metodickými poznámkami obsahujícími návrhy na rozšíření nebo zjednodušení obsahu podle úrovně studentů a tipy pro řešení častých problémů při výuce. Technická realizace prezentací zohledňuje běžné vybavení vzdělávacích institucí s kompatibilitou se standardními projektory a interaktivními tabulemi.

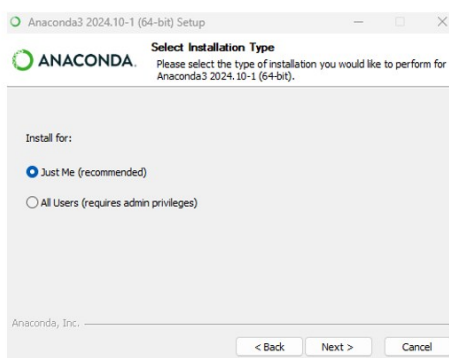
7.2.1 Úvod do Pythonu

První prezentace představuje fundamentální základy programovacího jazyka Python a slouží jako vstupní bod pro studenty bez předchozích zkušeností s programováním. Prezentace obsahuje 12 klíčových kapitol, které systematicky pokrývají nezbytné koncepty od elementárních základů až po komplexní aplikace. Úvodní část objasňuje význam Pythonu v současném technologickém prostředí a představuje jeho klíčové výhody včetně jednoduché syntaxe, interpretované povahy jazyka a obrovské komunity vývojářů.

Navazující kapitoly plynule přecházejí od teoretických základů k praktické přípravě vývojového prostředí. Prezentace představuje jak standardní instalaci oficiální distribuce Python, tak komplexnější řešení prostřednictvím Anaconda distribuce, která je zvláště vhodná pro matematické aplikace díky předinstalovaným knihovnám. Současně s instalačním procesem jsou představena různá vývojová prostředí včetně základního IDLE editoru, moderního Jupyter Notebook a profesionálního Visual Studio Code.

Návod instalace - Anacondy

- Spustíte stažený soubor
- Postupujte podle průvodce:
 - Typ instalace: Just Me (doporučeno)
 - Umístění: ponechte výchozí
- Dokončete instalaci a spustíte:
 - Anaconda Navigator (grafické rozhraní)
 - Anaconda Prompt (příkazová řádka s conda)



Obrázek 6 Návod na instalace Anacondy

(zdroj: vlastní)

Po dokončení přípravy prostředí prezentace systematicky zavádí základy syntaxe zahrnující datové typy, proměnné a operátory pro matematické výpočty, přičemž každý koncept je ihned doplněn praktickými příklady. Tyto základy jsou následně rozšířeny o řídicí struktury zahrnující podmínky a cykly, které jsou demonstrovány prostřednictvím

praktických cvičení jako hra "Hádej číslo" a generování tabulky násobků, jež účinně spojují teoretické poznatky s algoritmickým myšlením.

Cvičení – hra „Hádej číslo“

Zadání:

- Program náhodně vygeneruje číslo od 1 do 10
- Uživatel hádá, dokud se netrefí

```
import random
tajne = random.randint(1, 10)
hadani = 0

while hadani != tajne:
    hadani = int(input("Hádej číslo: "))
    if hadani < tajne:
        print("Příliš malé!")
    elif hadani > tajne:
        print("Příliš velké!")
print("Správně!")
```

```
Hádej číslo: 4
Příliš velké!
Hádej číslo: 2
Příliš malé!
Hádej číslo: 3
Správně!
```

Obrázek 7 Ukázka cvičení v prezentaci

(zdroj: vlastní)

Střední část prezentace systematicky navazuje na již osvojené základy a rozšiřuje znalosti studentů o práci s textem a datovými strukturami. Kapitola věnovaná manipulaci s řetězci zavádí techniky indexování, řezání a moderního formátování pomocí f-strings, které studenti okamžitě využívají v následujících částech při práci s datovými strukturami. Seznamy, n-tice, množiny a slovníky jsou představeny jako postupně složitější nástroje pro organizaci dat, přičemž každá struktura staví na konceptech představených v předchozích kapitolách.

Seznamy – užitečné metody

Další metody listu:

```
cisla = [3, 1, 4, 2]
cisla.sort()
cisla.reverse()
```

Metoda	Popis
.append(x)	přidá na konec
.insert(i,x)	vloží na pozici i
.remove(x)	odstraní první výskyt hodnoty
.pop()	odstraní poslední prvek
.sort()	seřadí vzestupně
.reverse()	obráťí pořadí

Obrázek 8 Užitečné příkazy pro práci se seznamy

(zdroj: vlastní)

Zvláštní pozornost věnovaná slovníkům pro ukládání strukturovaných dat vytváří základ pro následující kapitolu o funkcích a modularitě. Tato oblast zavádí koncepty strukturovaného programování včetně definice funkcí a předávání parametrů, které studenti ihned aplikují při reorganizaci kódu z předchozích cvičení do modulárních celků.

Co je funkce a proč ji používat?

Definice funkce pomocí **def**

- Funkce = pojmenovaný blok kódu, který lze volat vícekrát

- Základní struktura:

```
def pozdrav():
    print("Ahoj světe!")
```

- Volání funkce:

```
pozdrav()
```

Ahoj světe!

Pomáhá modularizovat program, zjednodušuje ladění

Obrázek 9 Použití funkce

(zdroj: vlastní)

Závěrečné kapitoly integrují všechny předchozí poznatky do komplexních dovedností nezbytných pro tvorbu robustních programů. Práce se soubory navazuje na znalosti datových struktur a umožňuje studentům ukládat a načítat data vytvořená v předchozích cvičeních. Bezpečné techniky pomocí kontextových manažerů logicky vedou k následující kapitole o ošetřování chyb prostřednictvím konstrukcí try-except, které studenti uplatňují při práci s externími soubory. Programátorské myšlení zahrnující algoritmizaci a dodržování konvencí podle PEP8 propojuje všechny předchozí poznatky do ucelené metodologie psaní kvalitního kódu.

Celá prezentace končí mini projektem evidence známek, který představuje syntézu všech probíraných konceptů od základních datových typů přes datové struktury a funkce až po práci se soubory a ošetřování chyb. Tento komplexní projekt současně demonstruje praktické využití Pythonu pro řešení reálných úloh a připravuje studenty na přechod k specializovaným matematickým knihovnám, kde budou využívat zde získané programovací základy pro pokročilejší aplikace.

Zadání úlohy – Evidence známek

Zadání:

Vytvořte jednoduchý konzolový program pro správu známek studentů:

- **Uživatel může:**
- Přidat studenta a jeho známky
- Zobrazit průměr známek studenta
- Uložit data do souboru
- Načíst data ze souboru

Základní požadavky:

- Použití slovníků (**dict**)
- Funkce (**def**)
- Zápis a čtení ze souboru (**open()**)
- Ošetření chyb (**try-except**)

Obrázek 10 Zadání závěrečného projektu

(zdroj: vlastní)

7.2.2 Základy práce s NumPy

Druhá prezentace se zaměřuje na knihovnu NumPy jako základ pro vědecké výpočty v Pythonu a navazuje na programovací dovednosti získané v úvodní prezentaci. Obsah systematicky pokrývá koncepty od základního pochopení až po praktické aplikace v oblasti

lineární algebry a generování náhodných čísel. Úvodní část objasňuje význam NumPy ve vědeckém systému Pythonu a představuje klíčové výhody knihovny včetně efektivní práce s vícerozměrnými poli, vysoké výpočetní rychlosti díky implementaci v jazyce C a rozsáhlé sady matematických funkcí. Presentace zdůrazňuje postavení jako základní knihovny, na které stavějí další specializované nástroje jako SciPy, Pandas a scikit-learn.

Instalace a import knihovny plynule navazují na znalosti z předchozí prezentace a rozšiřují je o specifické techniky práce s vědeckými knihovnami. Studenti se učí instalovat NumPy pomocí správců balíčků pip a conda, přičemž je zdůrazněna výhoda Anaconda distribuce pro matematické aplikace. Konvence importu knihovny jako np je představena jako standardní postup usnadňující psaní kódu. Presentace systematicky zavádí koncept ndarray jako základní datové struktury, která představuje n-rozměrné pole s homogenními datovými typy. Tvorba polí je demonstrována prostřednictvím různých metod od přímého vytváření pomocí np.array až po specializované funkce jako np.zeros, np.ones, np.eye pro jednotkovou matici a np.linspace pro generování číselných řad. Tyto znalosti jsou ihned aplikovány v praktických příkladech ukazujících rozdíly mezi seznamy Pythonu a NumPy poli z hlediska efektivity a funkcionality.

Pokročilé techniky práce s poli rozšiřují možnosti manipulace dat o sofistikované metody indexování a selekce. Základní indexování navazuje na znalosti z práce se seznamy v Pythonu, ale rozšiřuje je o práci s vícerozměrnými strukturami využívající zápis řádek a sloupec. Dalé zahrnují slicing pro výběr výseků dat, filtrování pomocí logických podmínek vytvářejících booleovské masky a fancy indexing umožňující výběr nesouvislých prvků podle seznamu indexů. Tyto techniky jsou propojeny s operacemi s poli, které představují vektorizované výpočty jako klíčovou výhodu oproti standardnímu Pythonu. Studenti se učí využívat aritmetické operace aplikované na celá pole bez nutnosti explicitních cyklů, univerzální funkce jako np.sin, np.exp, np.sqrt pro matematické výpočty a agregační funkce včetně np.sum, np.mean, np.std pro statistické analýzy.

Shrnutí pole – součty, průměry, rozptyl

Vypočet základní statistiky.

Funkce	Popis	Příklad	Výstup
np.sum(x)	Součet všech hodnot	np.sum([1, 2, 3])	6
np.mean(x)	Aritmetický průměr	np.mean([2, 4])	3.0
np.std(x)	Směrodatná odchylka	np.std([1, 2, 3])	~0.816
np.min(x)	Nejmenší hodnota	np.min([3, 1])	1
np.max(x)	Největší hodnota	np.max([3, 1])	3

• Příklad:

```
A = np.array([[1, 2],
               [3, 4]])
np.sum(A) # Výstup: 10
np.sum(A, axis=0) # Výstup: [4, 6] – součet sloupců
np.mean(A, axis=1) # Výstup: [1.5, 3.5] – průměr řádků
```

Obrázek 11 Funkce pro statistické analýzy

(zdroj: vlastní)

Koncept broadcasting je představen jako automatické rozšiřování polí různých tvarů pro kompatibilní operace. Lineární algebra a práce s náhodnými čísly integrují všechny poznatky do komplexních oblastí vědeckých výpočtů. Pokročilé vlastnosti polí zahrnují změnu tvaru pomocí reshape, zploštění prostřednictvím ravel a flatten a bezpečnou práci s kopiemi dat pomocí copy metody pro zabránění nechtěným změnám originálních dat. Lineární algebra představuje maticové operace včetně součinu matic pomocí np.dot a operátoru @, výpočtu inverzní matice a determinantu prostřednictvím np.linalg modulu a řešení soustav lineárních rovnic pomocí efektivní solve funkce. Práce s náhodnými čísly rozšiřuje možnosti NumPy o generování pseudonáhodných dat z různých distribucí, řízení náhodnosti pomocí seed funkcí pro opakovatelné experimenty a praktické simulace jevů pro modelování reálných situací. Celá prezentace je završena praktickými ukázkami zahrnujícími normalizaci dat a simulaci návštěvnosti obchodu, které demonstrují syntézu všech probíraných konceptů a připravují studenty na pokročilejší vědecké výpočty s knihovnou SciPy.

7.2.3 Vědecké výpočty s SciPy

Třetí prezentace představuje knihovnu jako pokročilou nadstavbu nad NumPy určenou pro specializované vědecké výpočty a matematické aplikace. Obsah systematicky pokrývá široké spektrum oblastí od numerické integrace přes optimalizaci až po statistické analýzy a

zpracování signálů. Úvodní část objasňuje postavení této knihovny v Python systému jako open-source nástroje rozšiřujícího možnosti NumPy o pokročilé algoritmy a funkce pro vědecké modelování. Prezentace zdůrazňuje modulární strukturu, kde každý submodule se zaměřuje na specifickou oblast vědeckých výpočtů, což umožňuje efektivní využití pouze potřebných funkcí. Současně je představeno využití v různých praktických oblastech včetně fyziky, inženýrství, biologie a ekonomie pro modelování komplexních systémů a analýzu dat.

Instalace a přehled modulů navazují na znalosti z předchozích prezentací a rozšiřují je o práci s specializovanými vědeckými nástroji. Studenti se seznámí s instalačním procesem prostřednictvím pip nebo conda správce balíčků a konvencemi importu jednotlivých modulů podle potřeby konkrétní aplikace. Prezentace systematicky představuje klíčové moduly včetně `scipy.integrate` pro numerickou integraci a řešení diferenciálních rovnic, `scipy.optimize` pro optimalizační úlohy a hledání extrémů funkcí, `scipy.stats` pro statistické výpočty a práci s pravděpodobnostními distribucemi a `scipy.interpolate` pro aproximaci dat. Praktické aplikace jsou demonstrovány prostřednictvím výpočtu plochy pod křivkou pomocí funkce `quad`, řešení diferenciálních rovnic s využitím `solve_ivp` pro modelování dynamických systémů a optimalizace funkcí pomocí `minimize` pro nalezení nejlepších řešení v dané oblasti.

Použití `minimize()` – jednoduchý příklad

Hledání minima funkce

- Definujeme cílovou funkci
- Určíme počáteční odhad (`x0`)
- Funkce najde minimum

```
from scipy.optimize import minimize

f = lambda x: (x - 3)**2 + 1
res = minimize(f, x0=0)
```

- Výstup: `res.x` (hodnota `x`, kde je minimum), `res.fun` (minimální hodnota funkce)

Tip: Pro maximum stačí minimalizovat $-f(x)$

Obrázek 12 Nalezení minima funkce

(zdroj: vlastní)

Pokročilé aplikace integrují různé moduly pro řešení komplexních vědeckých problémů vyžadujících kombinaci více technik. Interpolace dat pomocí `interp1d` umožňuje odhadovat chybějící hodnoty a vyhlazovat naměřená data, což je propojeno s pokročilými technikami pro zpracování 2D dat a povrchových modelů. Fourierova transformace prostřednictvím `scipy.fft` poskytuje nástroje pro analýzu periodických jevů a frekvenčního spektra signálů, což nachází uplatnění v oblasti zpracování zvuku, vibrací a obrazových dat. Statistické analýzy využívají rozsáhlou sadu funkcí pro práci s různými pravděpodobnostními distribucemi, odhad parametrů z naměřených dat pomocí funkcí pro přizpůsobení rozdělení a testování statistických hypotéz prostřednictvím specializovaných testů.

Lineární algebra prostřednictvím `scipy.linalg` rozšiřuje možnosti NumPy o pokročilé algoritmy pro rozklady matic, řešení soustav rovnic s vyšší stabilitou a výpočty vlastních čísel s využitím optimalizovaných implementací. Celá prezentace je završena praktickými příklady demonstrujícími syntézu různých modulů pro řešení reálných vědeckých problémů, které připravují studenty na komplexní aplikace matematického modelování a zároveň vytváří základ pro pochopení vizualizačních technik představených v následující prezentaci o Matplotlib.

7.2.4 Vizualizace dat s Matplotlib

Čtvrtá prezentace se zaměřuje na knihovnu Matplotlib jako základní nástroj pro tvorbu vizualizací v Pythonu a tvoří závěrečný prvek komplexního vzdělávacího systému. Obsah systematicky pokrývá široké spektrum vizualizačních technik od základních typů grafů až po pokročilé možnosti stylování a exportu. Úvodní část objasňuje klíčový význam této knihovny jako nejpoužívanějšího nástroje pro vizualizaci dat v Python systému a představuje její schopnost vytvářet publikační kvalitní grafy v různých formátech. Prezentace zdůrazňuje kompatibilitu s NumPy a dalšími vědeckými knihovnami, což umožňuje snadnou vizualizaci výsledků numerických výpočtů a analýz provedených v předchozích částech kurzu. Současně je představena flexibilita knihovny umožňující tvorbu jak jednoduchých grafů pro rychlou analýzu, tak složitých kompozic vhodných pro vědecké publikace.

Základní typy grafů a práce s osami navazují na znalosti získané v předchozích prezentacích a rozšiřují je o vizualizační možnosti pro prezentaci dat. Studenti se seznámí s instalací prostřednictvím `pip` nebo `conda` správce balíčků a standardní konvencí importu jako `plt`. Prezentace systematicky představuje základní typy grafů včetně spojnicových grafů pomocí `plot()` pro zobrazení funkcí a trendů, sloupcových grafů prostřednictvím `bar()` a

barh() pro porovnání kategorií, bodových grafů využívajících scatter() pro analýzu korelací a histogramů pomocí hist() pro zobrazení distribuce dat. Každý typ grafu je doplněn praktickými příklady demonstrujícími jeho nejvhodnější využití a kombinací s daty vytvořenými pomocí NumPy funkcí. Práce s osami zahrnuje nastavení rozsahu pomocí xlim() a ylim(), využití logaritmického měřítka prostřednictvím semilogx() a loglog() funkcí a přizpůsobení popisků os včetně natočení textu pro lepší čitelnost.

Pokročilé techniky stylování a export grafů integrují všechny poznatky do komplexních vizualizačních řešení vhodných pro profesionální použití. Stylování grafů zahrnuje přizpůsobení barev, stylů čar a značek pomocí parametrů color, linestyle a marker pro vytváření přehledných a esteticky působivých vizualizací. Nastavení fontů a velikostí textu umožňuje optimalizaci čitelnosti pro různé výstupní média včetně tiskových materiálů a prezentací. Využití přednastavených témat prostřednictvím style.use() poskytuje rychlé způsoby dosažení konzistentního vzhledu podle různých stylových konvencí. Subploty a rozvržení grafů pomocí subplot() a subplots() funkcí umožňují vytváření komplexních kompozic kombinujících více vizualizací v jednom obrázku.

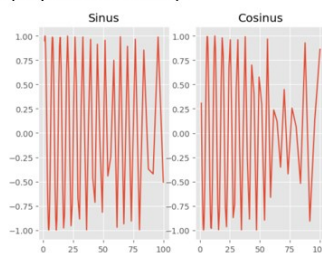
Více grafů v jednom obrázku – subplot()

Subplot() umožňuje zobrazit více grafů vedle sebe (např. 2x2 mřížku):

```
plt.subplot(1, 2, 1)
plt.plot(x, np.sin(x))
plt.title('Sinus')

plt.subplot(1, 2, 2)
plt.plot(x, np.cos(x))
plt.title('Cosinus')

plt.show()
```



Parametry subplot(**nrows**, **ncols**, **index**):

- **nrows**, **ncols** – počet řádků a sloupců
- **index** – pozice grafu (číslováno zleva)

Dobré pro srovnání více funkcí nebo dat

Obrázek 13 Zobrazení více grafů

(zdroj: vlastní)

Export grafů prostřednictvím savefig() funkcí s nastavením kvality, rozlišení a průhlednosti poskytuje možnosti pro využití ve vědeckých publikacích a prezentacích. Celá prezentace je završena praktickými příklady zahrnujícími porovnání datových sad a

vizualizaci reálných dat, které demonstrují syntézu všech probíraných konceptů a poskytují studentům kompletní sadu nástrojů pro efektivní komunikaci matematických a vědeckých výsledků prostřednictvím kvalitních vizualizací.

7.3 Demonstrační příklady

V rámci praktické části bakalářské práce byla vytvořena sada 30 demonstračních příkladů rozdělených do tří kategorií podle zaměření na konkrétní knihovny. NumPy příklady pokrývají operace s vektory a maticemi, statistické výpočty a řešení soustav lineárních rovnic. SciPy příklady demonstruje pokročilé vědecké výpočty včetně numerické integrace, optimalizace a interpolace. Matplotlib příklady jsou věnovány vizualizačním technikám od základních typů grafů až po pokročilé stylování a export.

Každý příklad je strukturován podle jednotného schématu zahrnujícího zadání s vysvětlením praktického významu, podrobný postup řešení rozdělený do logických kroků a kompletní kód implementace s komentáři. Tato struktura podporuje pochopení nejen technických aspektů implementace, ale také matematického nebo vizualizačního pozadí řešené úlohy pro flexibilní použití ve frontální výuce i samostatné práci studentů.

Výběr příkladů systematicky pokrývá základní funkcionality každé knihovny a jejich typické aplikace v matematicém vzdělávání. Metodické zpracování zahrnuje klasifikaci podle obtížnosti a poznámky pro pedagogy s doporučeními pro prezentaci a identifikaci častých chyb, které studenti mohou při řešení potkat.

7.3.1 Příklady pro základní operace s NumPy

Sada příkladů pro knihovnu NumPy obsahuje deset systematicky vybraných úloh pokrývajících základní i pokročilé operace s vícerozměrnými poli a maticemi. Příklady začínají elementárními operacemi jako sčítání vektorů a postupně přechází k složitějším aplikacím zahrnujícím řešení soustav lineárních rovnic a výpočty vlastních čísel. Každá úloha demonstruje specifickou funkcionalitu knihovny a poskytuje praktický kontext pro její využití v matematických výpočtech.

Fundamentální operace s vektory a maticemi jsou představeny prostřednictvím tří základních příkladů. Sčítání dvou stejně dlouhých vektorů demonstruje elementární aritmetické operace s poli a výhody vektorizovaných výpočtů oproti standardním Python seznamům. Výpočet skalárního součinu dvou vektorů využívá funkci pro maticový součin a

ilustruje význam této operace v kontextu geometrických a fyzikálních aplikací. Transpozice matice představuje základní operaci lineární algebry, která přehodí řádky a sloupce matice.

Příklad 3 – Transpozice matice

Zadání:

Máme matici, kde každý řádek představuje jednoho studenta a každý sloupec jeho výsledek v jednom ze tří testů. Cílem je zaměnit řádky a sloupce tak, aby každý nový řádek obsahoval výsledky jednoho testu napříč všemi studenty.

V tomto příkladu využijeme funkci `np.array()` pro vytvoření matice a atribut `.T` pro její transpozici.

Postup řešení:

1. Nejprve načteme knihovnu NumPy, protože budeme pracovat s dvourozměrným polem – tedy maticí.
2. Vytvoříme původní matici pomocí `np.array()`, kde každý řádek obsahuje tři čísla. Například první řádek `[1, 2, 3]` může být student A, druhý řádek `[4, 5, 6]` student B.
3. Zamyslíme se nad tím, co chceme změnit – naším cílem je zaměnit řádky za sloupce.
4. Použijeme atribut `.T`, který v NumPy provede transpozici matice – první sloupec se stane prvním řádkem, druhý sloupec druhým atd.
5. Výsledná matice bude mít tři řádky a dva sloupce. Například: `[1, 4], [2, 5], [3, 6]`.
6. Novou podobu matice uložíme a vypíšeme pomocí funkce `print()`.

Obrázek 14 Zadání příkladu transpozice matice

(zdroj: vlastní)

Analýza dat a statistické operace jsou zastoupeny příklady zaměřenými na výpočet základních charakteristik datových souborů. Výpočet statistických vlastností vektoru kombinuje funkce pro aritmetický průměr, medián a směrodatnou odchylku pro získání komplexního přehledu o rozložení dat. Normalizace vektoru demonstruje postup pro převod vektoru na jednotkovou délku využívající funkci pro výpočet normy vektoru, což je častá operace v oblasti strojového učení a zpracování signálů.

Specializované techniky zahrnují automatické přizpůsobování tvarů polí, manipulaci s tvarem polí a řešení lineárních systémů. Broadcasting je demonstrován prostřednictvím sčítání matice a vektoru, kde automaticky rozšiřuje menší pole pro kompatibilní operace. Implementace řešení soustavy lineárních rovnic využívá funkci pro lineární systémy namísto výpočtu inverzní matice, což představuje numericky stabilnější přístup. Závěrečné příklady pokrývají výpočet inverzní matice a vlastních čísel s vlastními vektory, které dokončují pokrytí základních operací lineární algebry. Komplexní aplikace je reprezentována generováním a řešením náhodné soustavy pěti rovnic o pěti neznámých s využitím generátorů náhodných čísel a lineárních řešičů.

7.3.2 Příklady pro vědecké výpočty s SciPy

Kolekce deseti příkladů pro knihovnu SciPy pokrývá široké spektrum pokročilých vědeckých výpočtů a specializovaných algoritmů podle tematických oblastí odpovídajících hlavním modulům knihovny. Příklady demonstrují aplikace numerických metod v různých vědeckých disciplínách.

Speciální matematické funkce jsou reprezentovány výpočtem Besselovy funkce prvního druhu prostřednictvím specializované funkce pro speciální matematické funkce, což demonstruje práci s funkcemi využívanými při řešení diferenciálních rovnic v inženýrských aplikacích. Kombinatorické výpočty ilustruje výpočet faktoriálu celého čísla pomocí optimalizované funkce pro statistické a pravděpodobnostní výpočty.

Příklad 2 – Výpočet faktoriálu celého čísla

Zadání:

Máme celé nezáporné číslo (například počet prvků v množině) a naším úkolem je vypočítat jeho faktoriál, tedy součin všech celých čísel od 1 do n . Tato operace se často používá při výpočtu kombinací, permutací nebo ve statistice.

V tomto příkladu použijeme funkci **factorial()** z modulu **scipy.special**.

Postup řešení:

1. Na začátku programu importujeme funkci **factorial** z modulu **scipy.special**. Tato funkce umí efektivně spočítat faktoriál zadaného čísla.
2. Zvolíme celé číslo, pro které budeme faktoriál počítat. Například $n = 5$.
3. Funkci **factorial()** zavoláme s tímto číslem jako vstupním argumentem a výsledek uložíme do proměnné.
4. Pokud si přejeme výsledek zobrazit jako celé číslo, můžeme jej převést pomocí **int()**.
5. Pomocí **print()** výsledek zobrazíme – pro $n = 5$ očekáváme výstup 120.

Obrázek 15 Zadání pro výpočet faktoriálu

(zdroj: vlastní)

Aproximační metody jsou zastoupeny příklady derivování a integrace funkcí. Numerické derivování využívá funkci pro aproximaci derivací tam, kde analytické řešení není praktické. Integrační techniky demonstruje výpočet určitého integrálu reálné funkce prostřednictvím adaptivních kvadrurních metod, což představuje základní nástroj pro výpočet ploch, objemů a dalších fyzikálních veličin.

Řešení rovnic a optimalizace zahrnují algoritmy pro hledání kořenů a extrémů funkcí. Nelineární rovnice jsou řešeny pomocí funkce pro hledání kořenů, což ukazuje numerické metody pro nulové body složitých rovnic bez analytického řešení. Datová interpolace

využívá lineární interpolaci prostřednictvím funkcí pro vytváření interpolačních objektů pro odhad hodnot mezi známými datovými body při zpracování experimentálních dat.

Modelování a simulace zahrnují regresní analýzu, diferenciální rovnice a signálové zpracování. Regresní techniky demonstruje aproximace dat metodou nejmenších čtverců pomocí funkcí pro přizpůsobení křivek pro nalezení optimálních parametrů matematických modelů na základě experimentálních měření. Dynamické systémy modeluje numerické řešení obyčejné diferenciální rovnice využívající funkce pro řešení počátečních úloh. Optimalizační úlohy reprezentuje hledání minima funkce jedné proměnné prostřednictvím funkcí pro skalární optimalizaci. Frekvenční analýza je ilustrována výpočtem diskrétní Fourierovy transformace pomocí funkcí pro rychlou Fourierovu transformaci při analýze periodických signálů a frekvenčního spektra.

7.3.3 Příklady pro vizualizaci s Matplotlib

Sada deseti příkladů pro knihovnu Matplotlib pokrývá vizualizační techniky od elementárních grafů po pokročilé možnosti stylování a exportu. Příklady umožňují studentům postupně osvojovat dovednosti potřebné pro tvorbu publikační kvality vizualizací vhodných pro vědecké prezentace a technickou dokumentaci.

Elementární vizualizace zahrnují nejčastěji používané grafické formáty. Spojnicové grafy jsou vykresleny prostřednictvím funkcí pro kreslení a zobrazení grafů, což demonstruje fundamentální strukturu příkazů pro vizualizaci a zavádí koncepty práce s osami a datovými body.

Příklad 1 – Vykreslení jednoduchého spojnitého grafu

Zadání:

Máme dvě sady údajů: první obsahuje čas v sekundách a druhá teplotu v daných časech. Cílem je vytvořit jednoduchý spojnité graf, který ukáže, jak se teplota v průběhu času měnila.

Použijeme funkce **plot()** a **show()** z modulu **matplotlib.pyplot**.

Postup řešení:

1. Nejdříve importujeme knihovnu Matplotlib, konkrétně její část pyplot, která se nejčastěji používá pro tvorbu grafů. Uděláme to příkazem: `import matplotlib.pyplot as plt`
2. Připravíme si dvě datová pole:
 - První pole bude obsahovat hodnoty osy x – v našem případě čas: `cas = [0, 1, 2, 3, 4, 5]`
 - Druhé pole bude obsahovat hodnoty teploty ve stejných časech:
`teplota = [22, 23, 23.5, 24, 24.5, 25]`
3. Použijeme funkci **plot(cas, teplota)**, která spojí datové body čarou. Tím vznikne základní spojnité graf.
4. Aby se graf opravdu zobrazil, přidáme příkaz **plt.show()**. Tento příkaz otevře nové okno s vykresleným grafem.
5. Výsledkem je jednoduchý graf, kde osa x zobrazuje čas a osa y hodnoty teploty. Ještě neobsahuje popisky ani nadpis – těm se budeme věnovat v dalším příkladu.

Obrázek 16 Zadání příkladu pro Matplotlib

(zdroj: vlastní)

Popisování grafů rozšiřuje základní vizualizaci o nezbytné metadata pomocí funkcí pro popisky os a názvy grafů. Kombinované vizualizace ilustruje vykreslení více křivek do jednoho grafu s využitím legendy pro porovnávání různých datových řad a rozlišování jednotlivých křivek prostřednictvím barev a popisků.

Kategorické a distribuční vizualizace rozšiřují spektrum možností pro různé typy dat a analýz. Sloupcové diagramy demonstruje vytvoření sloupcového grafu pro vizualizaci kategorických dat a srovnávání hodnot mezi různými skupinami. Frekvenční analýza využívá histogramy pro četnostní rozdělení a interpretaci tvarů distribucí. Korelační vizualizace umožňuje vytvoření bodového grafu prostřednictvím rozptylových diagramů pro zobrazení vztahů mezi dvěma proměnnými a identifikaci korelací nebo odlehklých hodnot.

Kompoziční techniky a estetické úpravy dokončují sadu vizualizačních dovedností. Vícenásobné grafy demonstruje zobrazení více grafů vedle sebe pomocí funkcí pro vytváření subgrafů pro porovnání různých aspektů dat nebo datových souborů v jedné vizualizaci. Pomocné prvky zahrnují přidání mřížky a poznámky do grafu prostřednictvím funkcí pro pomocné čáry a textové anotace, což zvyšuje čitelnost a umožňuje zvýrazňování důležitých bodů ve vizualizaci. Export a archivace představuje uložení grafu do souboru pomocí funkcí pro export grafů s různými možnostmi formátování jako nezbytnou dovednost pro použití

vizualizací v dokumentech a prezentacích. Stylování a přizpůsobení využívá parametry pro barvy, styly čar, tloušťku čar a značky bodů pro vytváření grafů odpovídajících konkrétním požadavkům nebo standardům organizace.

7.4 Český manuál pro Python a jeho matematické knihovny

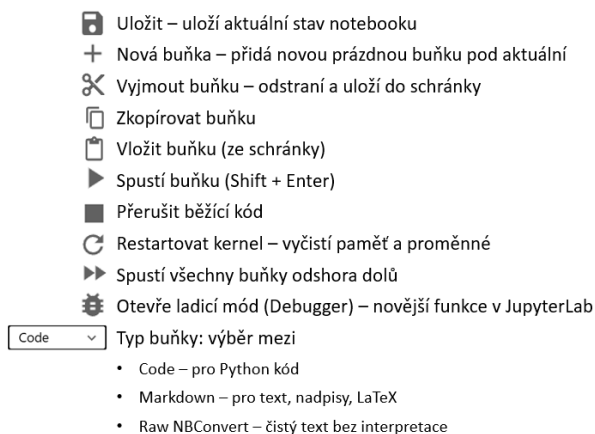
Tvorba českého manuálu představovala klíčovou komponentu praktické části této práce. Manuál vznikl jako reakce na nedostatek komplexních českých zdrojů pokrývajících Python a jeho matematické knihovny, což představuje bariéru zejména pro studenty středních škol a začínající uživatele na vysokých školách. Výsledný dokument o rozsahu 77 stran poskytuje ucelený přehled od základů programovacího jazyka Python až po pokročilé aplikace knihoven NumPy, SciPy a Matplotlib.

7.4.1 Struktura a obsah manuálu

Architektura manuálu vychází z pedagogických principů postupného budování znalostí a modulární struktury umožňující flexibilní využití v různých vzdělávacích kontextech. Dokument je rozdělen do sedmi tematických celků, přičemž každý celek tvoří samostatnou funkční jednotku s jasnými návaznostmi na předchozí a následující kapitoly.

Úvodní část věnovaná základům programovacího jazyka Python zaujímá zásadní pozici v celkové struktuře manuálu. Tato kapitola pokrývá instalační procesy pro různé operační systémy, představuje vývojová prostředí včetně Jupyter Notebook a systematicky zavádí základní programovací koncepty.

Jupyter Notebook



Obrázek 17 Seznámení s panelem nástrojů

(zdroj: vlastní)

Obsah postupuje od elementárních datových typů přes řídicí struktury až k pokročilým tématům jako objektivě orientované programování a zpracování výjimek.

Následující kapitoly věnované matematickým knihovnám zachovávají konzistentní organizační schéma. Každá kapitola začíná kontextualizací knihovny v rámci vědeckého systému Python, pokračuje instalačními pokyny a základními koncepty, následuje systematický přehled klíčových funkcí a končí praktickými aplikacemi. NumPy kapitola se zaměřuje na vícerozměrná pole jako fundamentální datovou strukturu, indexování, operace s poli a základy lineární algebry. SciPy kapitola rozšiřuje spektrum o specializované vědecké výpočty včetně integrace, diferenciálních rovnic, optimalizace a statistických funkcí. Matplotlib kapitola poskytuje komplexní průvodce vizualizačními technikami od základních typů grafů až po pokročilé možnosti formátování a exportu.

Struktura jednotlivých kapitol respektuje různé způsoby využití manuálu. Čtenáři mohli postupovat sekvenčně od začátku pro komplexní osvojení problematiky, nebo využívat manuál jako referenční příručku pro rychlé vyhledání konkrétních informací. Každá kapitola obsahuje přehledné sekce s jasnými nadpisy, což usnadňuje orientaci a rychlé nalezení potřebných informací.

7.4.2 Specifika českého názvosloví a terminologie

Tvorba české terminologie představovala zásadní výzvu při zachování přesnosti technických pojmů a současném zajištění srozumitelnosti pro českou cílovou skupinu. Metodologický přístup vycházel z analýzy existující české informatické terminologie s důrazem na konzistenci napříč celým dokumentem a návaznost na etablované konvence v českém vzdělávacím prostředí.

Základní programovací koncepty využívají etablovanou českou informatickou terminologii. Termíny jako proměnná, funkce, cyklus, podmínka nebo seznam mají v češtině přijaté ekvivalenty, které byly důsledně používány. Složitější terminologické otázky vznikaly u specifických konceptů knihoven, kde často neexistují přímé české ekvivalenty. V těchto případech byla aplikována strategie kombinující popisný český název s uvedením původního anglického termínu pro zachování možnosti orientace v mezinárodní literatuře. Python datové struktury byly překládány s respektováním jejich matematického významu. Termín list je konsistentně používán jako seznam, tuple jako n-tice, dictionary jako slovník a set jako množina.

PRÁCE S DATOVÝMI STRUKTURAMI

Při psaní větších programů nebo při zpracování většího množství dat nám nestačí jednoduché proměnné jako $x = 5$. Potřebujeme **uchovat více hodnot naráz**, umět k nim přistupovat, měnit je nebo je strukturovat. Právě k tomu slouží tzv. **datové struktury**.

Seznamy (list)

Seznam (list) si můžeme představit jako řadu krabiček vedle sebe, z nichž každá obsahuje nějakou hodnotu. Seznam má:

- pořadí (první, druhý, třetí prvek...),
- délku (kolik prvků obsahuje),
- index (číslo, kterým prvek označujeme, začíná od 0!).

Jak vytvořit seznam?

```
[44]: ovoce = ["jablko", "banán", "hruška"]
```

Seznam může obsahovat cokoli: čísla, texty, další seznamy – nebo i smíšeně.

Obrázek 18 Ukázka překladu na českou terminologii

(zdroj: vlastní)

NumPy pole ndarray je označováno jako n-rozměrné pole s vysvětlením jeho charakteristik jako homogenní datové struktury s pevnou velikostí. Koncept broadcasting byl přeložen jako automatické přizpůsobování tvarů polí s detailním vysvětlením mechanismu této funkcionality.

Matematická terminologie respektuje českou matematickou tradici. Termíny jako matice, vektor, determinant, vlastní čísla a vlastní vektory zachovávají své standardní české názvy. Specializované funkce knihoven jsou popisovány českými termíny s uvedením původních názvů funkcí, například funkce pro numerickou integraci, optimalizaci nebo statistické výpočty.

7.5 Praktické ověření materiálů

Praktické ověření vytvořených výukových materiálů představuje klíčovou komponentu validačního procesu, jejímž cílem bylo získat systematickou zpětnou vazbu o kvalitě a použitelnosti prezentací, manuálu a demonstračních příkladů. Proces evaluace byl navržen s důrazem na objektivní hodnocení pedagogické efektivity materiálů a jejich vhodnosti pro různé typy uživatelů v oblasti matematického programování.

7.5.1 Metodologie hodnocení

Metodický přístup k hodnocení vycházel z kombinace číselného hodnocení a slovních komentářů přizpůsobených specifickým vlastnostem vzdělávacích materiálů pro technické obory. Hodnotící proces byl rozdělen do tří hlavních etap zahrnujících přípravu hodnotících nástrojů, shromáždění dat od respondentů a rozbor získané zpětné vazby.

Přípravná etapa zahrnovala vytvoření strukturovaného dotazníku kombinujícího uzavřené otázky s desetibodovou stupnicí pro číselné vyjádření postojů respondentů a otevřené otázky pro získání podrobných slovních komentářů. Dotazník byl navržen tak, aby systematicky pokryl všechny části výukových materiálů a umožnil identifikaci silných stránek i oblastí vyžadujících zlepšení. Zvláštní pozornost byla věnována hodnocení vhodnosti materiálů pro různé úrovně předchozích znalostí a zkušeností s programováním. Fáze shromažďování dat využívala online distribuci dotazníku prostřednictvím internetové platformy, která zajišťovala anonymitu respondentů a efektivní sběr informací. Respondenti měli k dispozici kompletní sadu výukových materiálů a byli instruováni k jejich prohlédnutí před vyplněním hodnotícího formuláře. Tento postup zajistil, že všechna hodnocení vycházela z přímé zkušenosti s materiály.

Fáze rozboru kombinovala statistické zpracování číselných dat s tematickou analýzou slovních odpovědí. Číselná data byla analyzována pomocí základních statistických metod pro identifikaci celkových trendů v hodnocení, zatímco slovní odpovědi byly systematicky roztříděny podle tematických oblastí pro identifikace klíčových vzorců ve zpětné vazbě.

7.5.2 Analýza zpětné vazby

Analýza získané zpětné vazby byla provedena kombinací kvantitativních a kvalitativních metod s cílem získat komplexní pohled na kvalitu a použitelnost vytvořených výukových materiálů. Kvantitativní část zahrnovala statistické zpracování číselných hodnocení podle desetibodové stupnice, zatímco kvalitativní část byla zaměřena na tematickou analýzu slovních komentářů respondentů.

Struktura respondentů ukázala, že 83,3 % účastníků dotazníkového šetření mělo předchozí zkušenosti s programovacím jazykem Python. Rozdělení těchto zkušeností podle způsobu získání je uvedeno v tabulce 1.

Tabulka 1 Rozdělení zkušeností respondentů s Pythonem

Respondenti se zkušenostmi s Pythonem	Podíl respondentů
Studium na VŠ	41,7 %
Práce / praxe	16,7 %
Samostudium	16,7 %
Jiné	24,9 %

(zdroj: vlastní)

Míra zapojení respondentů do evaluačního procesu byla hodnocena na základě využití jednotlivých částí výukových materiálů (tabulka 2). Respondenti měli možnost vybrat více částí současně, což vysvětluje rozdělení preferencí mezi jednotlivými materiály.

Tabulka 2 Míra prohlédnutí jednotlivých částí

Komponenta	Podíl respondentů, kteří si komponentu prohlédli
Výukové prezentace	32,0 %
Manuál	39,0 %
Sady příkladů	29,0 %

(zdroj: vlastní)

Rozložení podle ročníků studia bylo rovnoměrné s mírnou převahou studentů druhého a třetího ročníku. Tematická analýza slovních komentářů identifikovala několik klíčových oblastí pozornosti. Respondenti nejčastěji zmiňovali potřebu zkrácení prezentačních snímků pro lepší přehlednost a soustředění pozornosti. Dalším významným tématem byla absence testovacích prvků, přičemž více respondentů navrhlo doplnění krátkých testů nebo kontrolních otázek po jednotlivých kapitolách. Pozitivní komentáře zdůrazňovaly kvalitu českého překladu a terminologie, systematickou strukturu materiálů a dobrou návaznost mezi teoretickými částmi a praktickými příklady.

7.5.3 Výsledky dotazníkového šetření

Celkové hodnocení výukových materiálů dosáhlo průměrné hodnoty 7,58 bodů z maximálních 10 bodů, což indikuje vysokou úroveň spokojenosti respondentů s kvalitou vytvořených materiálů. Toto hodnocení lze interpretovat jako potvrzení úspěšného naplnění stanovených cílů praktické části bakalářské práce. Nejvyšší hodnocení získala srozumitelnost a jazyková úroveň textu v manuálu s průměrnou hodnotou 8,55 bodů. Tento výsledek potvrzuje úspěšnost aplikovaného přístupu k tvorbě české terminologie a srozumitelné prezentaci technických konceptů pro českou cílovou skupinu. Vysoké hodnocení jazykové kvality je zvláště významné vzhledem k identifikovanému nedostatku kvalitních českých výukových materiálů v této oblasti.

Návaznost příkladů na výukové materiály byla hodnocena průměrnou známkou 8,78 bodů, což představuje nejlepší hodnocení ze všech sledovaných parametrů. Tento výsledek potvrzuje úspěšnost koncepce modulárního přístupu, kdy prezentace, manuál a příklady tvoří vzájemně provázaný systém podporující postupné budování znalostí a dovedností studentů. Užitečnost manuálu při práci s příklady dosáhla průměrného hodnocení 8,27 bodů, což dokládá efektivitu manuálu jako referenčního zdroje podporujícího praktické aplikace. Srozumitelnost zadání příkladů byla hodnocena průměrně 8,33 body, což potvrzuje jasnost a přehlednost formulací praktických úloh.

Grafická úroveň a vizualizace v prezentacích získaly nejnižší průměrné hodnocení 7,44 bodů, ačkoli i tato hodnota představuje nadprůměrnou úroveň spokojenosti. Tento výsledek identifikuje oblast s potenciálem pro budoucí zlepšení, zejména v kontextu požadavků respondentů na kratší a přehlednější snímky. Hodnocení obtížnosti příkladů ukázalo převažující vnímání přiměřené náročnosti, kdy 44,4 % respondentů označilo příklady jako snadné, 33,3 % jako přiměřené a pouze 11,1 % jako těžké. Žádný respondent neoznačil

příklady jako velmi snadné nebo velmi těžké, což potvrzuje úspěšné nastavení gradace obtížnosti odpovídající cílové skupině.

Významným indikátorem praktické hodnoty materiálů je skutečnost, že 83,3 % respondentů by materiály doporučilo jako součást výuky, zatímco žádný respondent nevyjádřil negativní postoj. Zbývajících 16,7 % respondentů uvedlo nerozhodnost, což může odrážet potřebu delšího období pro komplexní vyhodnocení pedagogické efektivity materiálů. Respondenti byli dotázáni na to, kterou část výukových materiálů považují za nejpřínosnější. Výsledky tohoto hodnocení jsou uvedeny v tabulce 3.

Tabulka 3 Hodnocení nejpřínosnější části výukových materiálů

Část výukových materiálů	Podíl respondentů
Kombinace všech částí	50,0 %
Výukové prezentace	16,7 %
Manuál	16,7 %
Sady příkladů	16,7 %

(zdroj: vlastní)

Jak vyplývá z výsledků, polovina respondentů (50,0 %) považuje kombinaci všech částí za nejpřínosnější, což potvrzuje správnost koncepce integrovaného přístupu k tvorbě výukových materiálů. Zbýající respondenti se rovnoměrně rozdělili mezi jednotlivé části, přičemž každá získala shodně 16,7 % preferencí. Toto vyrovnané rozdělení naznačuje, že všechny části materiálů mají svou hodnotu a odpovídají různým učebním stylům a preferencím studentů.

7.6 Návrhy na další rozvoj výukových materiálů

Na základě analýzy zpětné vazby z dotazníkového šetření a identifikovaných oblastí pro zlepšení byly formulovány návrhy na další rozvoj výukových materiálů. Tyto návrhy jsou strukturovány podle časového horizontu implementace a priorit pro zvýšení pedagogické efektivity materiálů. Krátkodobá vylepšení zahrnují úpravy, které lze implementovat bez zásadních strukturálních změn stávajících materiálů. Nejvyšší prioritu má rozdělení rozsáhlejších prezentačních snímků na menší logické celky pro zvýšení přehlednosti a udržení pozornosti studentů během výuky. Respondenti opakovaně zmiňovali potřebu

kratších snímků, což indikuje překročení optimální míry informací na jednotlivých snímcích. Doplnění krátkých testovacích otázek po každé kapitole představuje další významné vylepšení podporující aktivní učení a umožňující studentům ověřit úroveň porozumění probíranému tématu. Tyto testy by měly být koncipovány jako formativní hodnocení s okamžitou zpětnou vazbou, nikoliv jako nástroje pro klasifikaci studentů.

Střednědobé rozvojové aktivity se zaměřují na rozšíření funkcionality a dostupnosti materiálů. Vytvoření online platformy s interaktivními příklady by umožnilo studentům pracovat s kódem přímo v prohlížeči bez nutnosti instalace softwaru, což eliminuje technické bariéry zejména pro začínající uživatele. Tato platforma by mohla integrovat funkcionalitu pro automatické hodnocení řešení a poskytování personalizované zpětné vazby. Tvorba doplňkových video tutoriálů představuje možnost oslovení studentů s různými učebními styly a preferencemi. Video materiály by měly doplňovat, nikoliv nahrazovat stávající prezentace, a zaměřit se na demonstraci praktických postupů a řešení nejčastějších problémů při implementaci kódu.

Dlouhodobé rozvojové směry zahrnují systematické rozšíření tematického pokrytí o další významné knihovny Python systému. Integrace pandas pro analýzu dat by umožnila studentům pracovat s reálnými datovými soubory a provádět pokročilé datové transformace. Zahrnutí základů scikit-learn by poskytlo vstup do oblasti strojového učení, což odpovídá rostoucí poptávce po těchto dovednostech na trhu práce. Vývoj pokročilých modulů pro specifické aplikační oblasti jako zpracování obrazu pomocí OpenCV, analýza časových řad nebo webové aplikace s využitím frameworků jako Flask by umožnil diferenciaci výuky podle zájmů a specializace studentů. Implementace systému pro sběr průběžné zpětné vazby od pedagogů využívajících materiály v praxi by poskytla cenné informace o dlouhodobé efektivitě materiálů a identifikovala oblasti vyžadující aktualizaci v souladu s vývojem technologií a pedagogických přístupů.

ZÁVĚR

Tato bakalářská práce si kladla za cíl poskytnout komplexní přehled matematického softwaru vhodného pro vzdělávací účely a současně přispět k řešení identifikovaných nedostatků v oblasti českých výukových materiálů pro matematické programování. Stanovené cíle byly naplněny prostřednictvím systematické analýzy osmi reprezentativních softwarových nástrojů a vytvoření ucelené sady výukových materiálů pro Python a jeho matematické knihovny NumPy, SciPy a Matplotlib.

Teoretická část práce poskytla strukturovaný přehled současného stavu matematického softwaru napříč čtyřmi hlavními kategoriemi. Analýza vědeckých výpočetních systémů GNU Octave a Scilab ukázala jejich potenciál jako dostupných alternativ ke komerčnímu MATLABu pro výuku numerických metod a lineární algebry. Charakteristika dynamického geometrického systému GeoGebra potvrdila jeho význam pro vizualizaci matematických konceptů a podporu interaktivního učení. Hodnocení systémů pro symbolickou matematiku odhalilo rozdíly mezi open-source řešením SymPy a komerčními produkty Maple a Wolfram Mathematica z hlediska funkcionality a dostupnosti pro vzdělávací instituce. Analýza mobilních aplikací Photomath a Microsoft Math Solver identifikovala jejich potenciál i rizika v kontextu moderního matematického vzdělávání.

Praktická část práce realizovala komplexní sadu výukových materiálů v českém jazyce, která zahrnuje čtyři výukové prezentace systematicky pokrývající postupný přechod od základů programovacího jazyka Python až po pokročilé aplikace jeho matematických knihoven. Vytvořených třicet demonstračních příkladů poskytuje praktické aplikace teoretických konceptů v různých oblastech matematiky. Český manuál o rozsahu sedmdesáti sedmi stran představuje unikátní příspěvek k české odborné literatuře v oblasti matematického programování, který reflektuje specifika českého názvosloví a terminologie.

Dotazníkové šetření provedené mezi cílovou skupinou studentů potvrdilo kvalitu vytvořených materiálů s celkovým průměrným hodnocením 7,58 bodů z maximálních deseti bodů. Zvláště pozitivně byla hodnocena srozumitelnost a jazyková úroveň manuálu s průměrnou známkou 8,55 bodů a návaznost příkladů na výukové materiály s hodnocením 8,78 bodů. Významným ukazatelem praktické hodnoty materiálů je skutečnost, že 83,3 % respondentů by je doporučilo jako součást výuky.

Získaná zpětná vazba současně identifikovala oblasti pro další zlepšení, zejména potřebu zkrácení prezentačních snímků a doplnění testovacích prvků pro ověření

porozumění studentů. Tyto poznatky byly systematicky zpracovány do návrhů na další rozvoj materiálů strukturovaných podle časového horizontu implementace od krátkodobých úprav až po dlouhodobé rozšíření tematického pokrytí. Hlavní přínos práce spočívá ve vytvoření prakticky orientovaných výukových zdrojů, které mohou přispět ke zkvalitnění výuky matematiky na středních a vysokých školách prostřednictvím integrace moderních programovacích nástrojů. Vytvořené materiály podporují rozvoj digitálních kompetencí studentů, algoritmického myšlení a schopnosti aplikovat matematické znalosti v praktických situacích. Systematický přehled matematického softwaru poskytuje pedagogům objektivní základ pro výběr vhodných nástrojů podle specifických vzdělávacích cílů a technických podmínek. Vytvořené materiály mají potenciál podporovat rozvoj matematického vzdělávání v českém prostředí a přispět k přípravě studentů na požadavky současného technologicky orientovaného trhu práce.

SEZNAM POUŽITÉ LITERATURY

- [1] SCHMIDT HANSEN, Jesper. GNU Octave: beginner's guide: become a proficient Octave user by learning this high-level scientific numerical tool from the ground up. Learn by doing: less theory, more results. Birmingham: Packt, 2011. ISBN 978-1-849513-32-6.
- [2] SCILAB ENTERPRISES a GOMEZ, Christine. Scilab for very beginners. Online. 2013. Dostupné z: https://www.scilab.org/sites/default/files/Scilab_beginners.pdf. [cit. 2024-11-11].
- [3] GEOGEBRA. GeoGebra Manual: The official manual of GeoGebra. Online. 2011. Dostupné z: https://research.shu.ac.uk/geogebra/GIS_Guides/Official%20GeoGebra%20Manual.pdf. [cit. 2024-11-11].
- [4] SYMPY DEVELOPMENT TEAM. SymPy Documentation. Release 1.13.3. Online. 2024. Dostupné z: <https://github.com/sympy/sympy/releases/download/1.13.3/sympy-docs-pdf-1.13.3.pdf>. [cit. 2024-11-11].
- [5] KELECSÉNYI, Klára, OSZTÉNYINÉ, Krauczi Éva a DIÁNA, Zita. Photomath: A blessing or a Curse. Gradus. Online. 2021, no. 1. ISSN 2064-8014. [cit. 2024-11-11].
- [6] EATON, John W., BATEMAN, David, HAUBERG, Søren a WEHBRING, Rik. GNU Octave: A high-level interactive language for numerical computations. Online. 2023. Dostupné z: <https://octave.org/octave.pdf>. [cit. 2024-11-11].
- [7] DANIŠ, Stanislav. Základy programování v prostředí Octave a Matlab. Vydání druhé, upravené a rozšířené. Odborná edice Matfyzpress. Praha: Matfyzpress, nakladatelství Matematicko-fyzikální fakulty Univerzity Karlovy, 2022. ISBN 978-80-7378-474-4.
- [8] SAHA, Amit. Doing math with Python: use programming to explore algebra, statistics, calculus, and more!. San Francisco: No Starch Press, 2015. ISBN 978-1-59327-640-9.
- [9] LINGE, Svein a Hans Petter LANGTANGEN. Programming for Computations – Python: A Gentle Introduction to Numerical Simulations with Python. Texts in

- Computational Science and Engineering, sv. 15. Cham: Springer International Publishing, 2016. ISBN 978-3-319-32427-2.
- [10] GERGELITSOVÁ, Šárka. Počítač ve výuce nejen geometrie: průvodce Geogebrou. Praha: Generation Europe, 2011. ISBN 978-80-904974-3-6.
- [11] MAHNELOVÁ, Hana. Užití aplikací Microsoft Math Solver nebo Photomath při řešení slovních úloh. Metodický portál: Články. Online. 2023. Dostupné z: <https://clanky.rvp.cz/clanek/23635/UZITI-APLIKACI-MICROSOFT-MATH-SOLVER-NEBO-PHOTOMATH-PRI-RESENI-SLOVNICH-ULOHO.html>. ISSN 1802-4785. [cit. 2025-03-20].
- [12] MICROSOFT CORPORATION. Microsoft Math Solver - Math Problem Solver & Calculator. Online. 2023. Dostupné z: <https://math.microsoft.com/>. [cit. 2025-03-19].
- [13] LIENGME, Bernard V. Maple: A Primer. San Rafael: Morgan & Claypool, 2019. ISBN 978-1-64327-485-0.
- [14] WOLFRAM RESEARCH. Wolfram Mathematica: Modern Technical Computing. Online. ©2024. Dostupné z: <https://www.wolfram.com/mathematica/>. [cit. 2025-03-21].
- [15] BOUŠKA, Martin. Příručka programu Mathematica. Online. Praha, 2012. Dostupné z: <https://www.pslib.cz/jaromir.oscadal/PRG%20-V1/Mathematica%20-%20p%C5%99%C3%ADru%C4%8Dka3.pdf>. [cit. 2025-03-21].
- [16] JOHANSSON, Robert. Numerical Python: a practical techniques approach for industry. Expert's voice in Python. New York: Apress, 2015. ISBN 978-1-4842-0554-9.
- [17] NUMPY DOCUMENTATION TEAM. NumPy User Guide and Reference. Online. 2023. Dostupné z: <https://numpy.org/doc/stable/>. [cit. 2025-05-02].
- [18] SCIPY COMMUNITY. SciPy Reference Guide. Online. 2023. Dostupné z: <https://docs.scipy.org/doc/scipy/reference/>. [cit. 2025-05-02].
- [19] MATPLOTLIB DEVELOPMENT TEAM. Matplotlib User Guide. Online. 2023. Dostupné z: <https://matplotlib.org/stable/users/index.html>. [cit. 2025-05-02].

- [20] FÜHRER, Claus; SOLEM, Jan Erik a VERDIER, Olivier. Scientific computing with Python: high-performance scientific computing with NumPy, SciPy, and pandas. Second edition. Birmingham: Packt, 2021. ISBN 978-1-83882-232-3.
- [21] PAJANKAR, Ashwin. Hands-on Matplotlib - Learn Plotting and Visualizations with Python 3. Berkeley: Apress, 2022. ISBN 978-1-4842-7409-5.
- [22] HUNT, John. Advanced Guide to Python 3 Programming [online]. 1st ed. Undergraduate Topics in Computer Science. Cham: Springer, 2019 [cit. 2025-05-28]. ISBN 978-3-030-25943-3.
- [23] GNU OCTAVE COMMUNITY. GNU Octave: Scientific Programming Language. Online. ©2024. Dostupné z: <https://www.octave.org/>. [cit. 2025-05-28].
- [24] MATHWORKS. MATLAB - The Language of Technical Computing. Online. Dostupné z: <https://uk.mathworks.com/products/matlab/getting-started.html>. [cit. 2025-05-28].
- [25] VILLALOBOS ALVA, Jalil. Beginning Mathematica and Wolfram for Data Science - Applications in Data Analysis, Machine Learning, and Neural Networks. Berkeley, CA: Apress, an imprint of Springer Nature, 2021. Dostupné z: <https://doi.org/10.1007/978-1-4842-6594-9>.
- [26] PECHERKOVÁ, Pavla. Úvod do SCILABu. Online. 2017. Dostupné z: https://personal.fd.cvut.cz/nemcopav/materialy/Scilab_komplet_201404.pdf. [cit. 2025-05-28].
- [27] X-ENGINEER.ORG. Introduction to Scilab. Online. 2024. Dostupné z: <https://x-engineer.org/introduction-scilab/>. [cit. 2025-05-28].
- [28] SCILAB ENTERPRISES. Scilab - Open Source Software for Numerical Computation. Online. ©2024. Dostupné z: <https://www.scilab.org/>. [cit. 2025-05-28].
- [29] ATTAWAY, Stormy. *MATLAB: a practical introduction to programming and problem solving*. Sixth edition. Oxford: Butterworth-Heinemann, 2023. ISBN 978-0-323-91750-6.
- [30] GEOGEBRA INTERNATIONAL. *GeoGebra*. Online. 2024. Dostupné z: <https://www.geogebra.org/>. [cit. 2025-05-28].

- [31] LISÝ, Štěpán. *Využití programu GeoGebra při řešení úloh o pohybu*. Online. Plzeň, 2024. Bakalářská práce. Západočeská univerzita v Plzni, Fakulta pedagogická. Dostupné z: <https://theses.cz/id/dgnzz7/>. [cit. 2025-05-28].
- [32] DATASCIENTEST. *SymPy: Everything You Need to Know About the Python Symbolic Computation Library*. Online. 2024. Dostupné z: <https://datascientest.com/en/sympy-everything-you-need-to-know-about-the-python-symbolic-computation-library>. [cit. 2025-05-28].
- [33] MEURER, A. a kol. SymPy: symbolic computing in Python. *PeerJ Computer Science*. Online. 2017, 3, e103. DOI <https://doi.org/10.7717/peerj-cs.103>. Dostupné z: <https://peerj.com/articles/cs-103/>. [cit. 2025-05-28].
- [34] SYMPY DEVELOPMENT TEAM. *SymPy*. Online. 2024. Dostupné z: <https://www.sympy.org/en/index.html>. [cit. 2025-05-28].
- [35] MAPLESOFT. *Maple*. Online. 2024. Dostupné z: <https://www.maplesoft.com/products/maple/>. [cit. 2025-05-28].
- [36] LIENGME, Bernard V. *Maple: A primer*. San Rafael, CA: Morgan & Claypool Publishers, 2019. ISBN 978-1-64327-488-1. DOI: [10.1088/2053-2571/ab0bb3](https://doi.org/10.1088/2053-2571/ab0bb3).
- [37] WOLFRAM RESEARCH. *Wolfram Mathematica: Modern Technical Computing*. Online. 2024. Dostupné z: <https://www.wolfram.com/mathematica/?source=nav>. [cit. 2025-05-28].

SEZNAM OBRÁZKŮ

Obrázek 1 Uživatelské prostředí programu GNU Octave	19
Obrázek 2 Uživatelské prostředí programu Scilab	21
Obrázek 3 Uživatelské prostředí programu GeoGebra	24
Obrázek 4 Prostedí aplikace Photomath	33
Obrázek 5 Uživatelské prostředí aplikace Microsoft Math Solver	35
Obrázek 6 Návod na instalace Anacondy	46
Obrázek 7 Ukázka cvičení v prezentaci	47
Obrázek 8 Užitečné příkazy pro práci se seznamy	48
Obrázek 9 Použití funkce	48
Obrázek 10 Zadaní závěrečného projektu	49
Obrázek 11 Funkce pro statistické analýzy	51
Obrázek 12 Nalezení minima funkce	52
Obrázek 13 Zobrazení více grafů	54
Obrázek 14 Zadaní příkladu transpozice matice	56
Obrázek 15 Zadaní pro výpočet faktoriálu	57
Obrázek 16 Zadaní příkladu pro Matplotlib	59
Obrázek 17 Seznámení s panelem nástrojů	61
Obrázek 18 Ukázka překladu na českou terminologii	62

SEZNAM TABULEK

Tabulka 1 Rozdělení zkušeností respondentů s Pythonem.....	64
Tabulka 2 Míra prohlédnutí jednotlivých částí.....	64
Tabulka 3 Hodnocení nejpřínosnější části výukových materiálů	66

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

API	Application Programming Interface (rozhraní pro programování aplikací)
CAS	Computer Algebra System (systém pro symbolickou matematiku)
DPI	Dots Per Inch (počet bodů na palec)
FFT	Fast Fourier Transform (rychlá Fourierova transformace)
GPL	General Public License (všeobecná veřejná licence)
GUI	Graphical User Interface (grafické uživatelské rozhraní)
IDE	Integrated Development Environment (integrované vývojové prostředí)
ODE	Ordinary Differential Equation (obyčejná diferenciální rovnice)
PEP8	Python Enhancement Proposal 8 (doporučení, jak psát čitelný kód)
2D	Two-Dimensional (dvourozměrný)
3D	Three-Dimensional (trojrozměrný)
BSD	Berkeley Software Distribution (typ open-source licence)

SEZNAM PŘÍLOH

Příloha P I: ZIP soubor s výukovými podklady a manuálem v IS STAG

PŘÍLOHA P I: PRILOHY.ZIP

Přílohou bakalářské práce je ZIP soubor nahraný do systému STAG. Archiv obsahuje výukové podklady, které jsou rozděleny podle typu.

Výukové podklady dle složek:

Složka „Manual“ obsahuje tento soubor:

Manual_Python.pdf

Složka „Python“ obsahuje tento soubor:

01_Python.pptx

Složka „NumPy“ obsahuje tyto soubory:

01_NumPy.pptx

01_Priklady_NumPy.docx

01_Reseni_NumPy.docx

Složka „SciPy“ obsahuje tyto soubory:

01_SciPy.pptx

01_Priklady_SciPy.docx

01_Reseni_SciPy.docx

Složka „Matplotlib“ obsahuje tyto soubory:

01_Matplotlib.pptx

01_Priklady_Matplotlib.docx

01_Reseni_Matplotlib.docx