MASARYKOVA UNIVERZITA

PŘÍRODOVĚDECKÁ FAKULTA

MOCNINNÉ ŘADY S PROGRAMEM MAPLE

RIGORÓZNÍ PRÁCE

duben 2006

Pavel Kříž

Prohlašuji, že jsem rigorózní práci vypracoval samostatně s použitím uvedené literatury.

V Brně 10. dubna 2006

Děkuji prof. RNDr. Zuzaně Došlé, DSc. za cenné rady a připomínky k tématu rigorózní práce. Děkuji také doc. RNDr. Stanislavu Bartoňovi, CSc. za rady týkající se problematiky programu Maple.

Obsah

Úvod

Obrovský rozmach informačních technologií, které v dnešní době zasahují do mnoha odvětví vědy, je jedním z hlavních podnětů pro propojení matematiky s možnostmi výpočetní techniky.

Cílem rigorózní práce je vypracovat učební text se zaměřením na využití programu Maple a informačních technologií při analýze mocninných řad.

Práce je rozdělena do čtyř kapitol. Začátky prvních dvou kapitol jsou spíše teoretické a jsou zde zavedeny potřebné pojmy a vlastnosti, na které navazují řešené příklady. Prezentujeme metodu řešení „krok za krokem“ s využitím početních a vizualizačních možností, které Maple při řešení poskytuje. Vysvětlujeme také výhody automatizace a zavádíme tři nové procedury, které doplňující současné možnosti programu Maple. Uvedeme vhodné možnosti interaktivního publikování matematiky na internetu a představíme novou internetovou aplikaci zabývající se mocninnými řadami.

V úvodní části první kapitoly jsou zavedeny pojmy mocninná řada, součet a poloměr konvergence. V další části ukazujeme výpočet poloměru a oboru konvergence mocninných řad metodou „krok za krokem“ a automatizovanými procedurami `ConvergenceRadius` a `ConvergenceTestOfPowerSeries`, které byly v práci nově vytvořeny. Srovnáme tyto procedury s jinými dostupnými procedurami pro mocninné řady.

Druhá kapitola je zaměřena na rozvoj funkcí do mocninných řad, výpočet a vizualizaci Taylorových a Maclaurinových polynomů. Na výpočtu Taylorových polynomů srovnáme užití standardní procedury `taylor` a nové funkce `TaylorPolynomial`. Předvedeme základní možnosti grafické vizualizace, tvorbu grafů a animací Taylorových polynomů nejprve metodou „krok za krokem“ a poté novou procedurou `TaylorPolynomialAnimation`.

Internetové prezentaci, jakožto rozšíření využitelosti práce i pro uživatele, kteří nemají Maple k dispozici, je věnována třetí kapitola. Uvedeme novou interaktivní aplikaci nazvanou `Mocninné řady s Maple`, která je dostupná na internetové adrese `http://www.math.muni.cz/~kriz/pseries`. Ukážeme rozsah jednotlivých částí aplikace, předvedeme uživatelské prostředí a užití aplikace na příkladech.

Veškeré procedury, které byly v rigorózní práci nově vytvořeny jsou implementovány do nové knihovny `PowerSeries`. Závěrečná kapitola obsahuje technické informace o této knihovně a její instalaci.

Součástí práce je i digitální příloha ve formě internetové prezentace. Kromě zmíněné knihovny `PowerSeries` obsahuje i zdrojové kódy nových procedur, popis a předvedení jejich využití na řešených příkladech. Prezentace je dostupná jak na internetové adrese `http://www.math.muni.cz/~kriz/pseries/cd` tak i na přiloženém CD-ROMu. V tištěné verzi zdrojové kódy neuvádíme, aby nebyla narušena přehlednost práce.

Veškerá teorie je zpracována podle [3]. Tvrzení jsou uváděna bez důkazů. Používaná symbolika je shodná se symbolikou užívanou v [3]; zejména symbol $\mathbb{R}^*$ označuje rozšířenou množinu reálných čísel, tj. $\mathbb{R}^* = \mathbb{R} \cup \{-\infty, \infty\}$.

Kapitola 1

Mocninné řady

Úvodní část první kapitoly je zaměřena na zavedení základních vlastností mocninných řad. Seznámíme se s pojmy poloměr a obor konvergence mocninné řady. Závěr sekce 1.1 je věnován součtu mocninné řady. Ukazujeme užití procedury `sum`, kterou lze použít k součtu konečných i nekonečných řad.

Zbylé sekce kapitoly jsou formou řešených příkladů zaměřeny na výpočet poloměru a oboru konvergence mocninné řady. Uvádíme různé metody řešení. Metoda „krok za krokem“ ukazuje klasický postup řešení, kterým bychom příklad řešili přímo v Maple.

Vysvětlujeme základní principy, přednosti či nevýhody automatizace. Maple jako výpočetní systém je velmi rozsáhlý a zasahuje do mnoha matematických oborů, avšak proceduru pro určování oboru konvergence mocninných řad neobsahuje. Postup řešení příkladů metodou „krok za krokem“ je proto zobecněný a automatizovaný do nových procedur. Nové procedury nazýváme `ConvergenceRadius` (viz sekce 1.4) a `ConvergenceTestOfPowerSeries` (viz sekce 1.6).

1.1 Základy teorie

Definice 1.1. Buď $\{a_n\}_{n=0}^{\infty}$ posloupnost reálných čísel, x_0 libovolné reálné číslo. *Mocninnou řadou* se středem v bodě x_0 a koeficienty a_n rozumíme řadu funkcí tvaru

$$a_0 + a_1(x - x_0) + a_2(x - x_0)^2 + \cdots + a_n(x - x_0)^n + \cdots = \sum_{n=0}^{\infty} a_n(x - x_0)^n.$$

Poznámka 1.1. Bez újmy na obecnosti lze předpokládat, že středem mocninné řady je číslo $x_0 = 0$. Jinak pomocí substituce $x - x_0 = y$ lze převést řadu o středu v bodě x_0 na mocninnou řadu o středu v počátku.

Věta 1.1. *Nechť* $\sum a_n x^n$ *je mocninná řada a nechť*

$$a = \limsup \sqrt[n]{|a_n|}\ .$$

Je-li $a = 0$, pak řada absolutně konverguje pro všechna $x \in \mathbb{R}$ — říkáme, že řada vždy konverguje.

Je-li $a = \infty$, pak řada diverguje pro všechna $x \neq 0$ — říkáme, že řada vždy diverguje.

Je-li $0 < a < \infty$, pak řada absolutně konverguje pro $|x| < \frac{1}{a}$ a diverguje pro $|x| > \frac{1}{a}$.

Je-li $0 < a < \infty$, pak se číslo $r = \frac{1}{a}$ nazývá *poloměr konvergence* a interval $(-r, r)$ se nazývá *konvergenční interval.* Chování řady v krajních bodech konvergenčního intervalu je třeba vyšetřit zvlášť, protože závisí na tvaru mocninné řady. Oborem konvergence mocninné řady, která vždy nediverguje, je proto konvergenční interval s případnými jeho krajními body, pokud v nich řada konverguje.

Jestliže řada $\sum a_n x^n$ vždy konverguje, tj. $a = 0$, definujeme její poloměr konvergence jako $r = \infty$ a její konvergenční interval jako $(-\infty, \infty)$.

Jestliže řada $\sum a_n x^n$ vždy diverguje, tj. $a = \infty$, definujeme její poloměr konvergence jako $r = 0$.

Poznámka 1.2. Existuje-li $\lim \sqrt[n]{|a_n|} = a$, pak má mocninná řada $\sum a_n x^n$ poloměr konvergence

$$r = \frac{1}{\lim \sqrt[n]{|a_n|}}$$

(přitom klademe $r = \infty$, je-li $a = 0$, a $r = 0$, je-li $a = \infty$).

Dále platí, že existuje-li $\lim |\frac{a_{n+1}}{a_n}|$, pak existuje také $\lim \sqrt[n]{|a_n|}$ a obě jsou si rovny. Proto pokud existuje tato limita, lze poloměr konvergence určit jako

$$r = \lim \left| \frac{a_n}{a_{n+1}} \right| .$$

Mocninná řada je speciálním případem řady funkcí, u níž hraje klíčovou roli stejnoměrná konvergence. Následující věta říká, na jakém intervalu je mocninná řada stejnoměrně konvergentní.

Věta 1.2. *Nechť $r > 0$ je poloměr konvergence mocninné řady $\sum a_n x^n$. Pak tato řada stejnoměrně konverguje na každém uzavřeném podintervalu $[-\rho, \rho]$ intervalu $(-r, r)$.*

Tato věta má následující tři důsledky o součtu, integraci a derivaci mocninných řad.

Důsledek 1.1. *Nechť mocninná řada $\sum a_n x^n$ má poloměr konvergence $r > 0$. Pak součet této řady je spojitá funkce na intervalu $(-r, r)$.*

Důsledek 1.2. *Nechť mocninná řada $\sum a_n x^n$ má poloměr konvergence $r > 0$. Pak pro všechna $x \in (-r, r)$ platí*

$$\int_0^x \left(\sum_{n=0}^{\infty} a_n t^n \right) \mathrm{d}t = \sum_{n=0}^{\infty} \int_0^x a_n t^n \, \mathrm{d}t = \sum_{n=0}^{\infty} a_n \frac{x^{n+1}}{n+1} \,, \tag{1.1}$$

přičemž mocninná řada na pravé straně má stejný poloměr konvergence r.

Důsledek 1.3. *Nechť mocninná řada* $\sum a_n x^n$ *má poloměr konvergence* $r > 0$. *Pak pro všechna* $x \in (-r, r)$ *platí*

$$\left(\sum_{n=1}^{\infty} a_n x^n \right)' = \sum_{n=1}^{\infty} (a_n x^n)' = \sum_{n=1}^{\infty} n a_n x^{n-1}, \tag{1.2}$$

přičemž mocninná řada na pravé straně má opět poloměr konvergence r.

Příklad 1.1. Určete součet následujících řad:

$$\text{a)} \quad \sum_{n=1}^{\infty} x^n \qquad \text{b)} \quad \sum_{n=1}^{\infty} \frac{x^n}{n} \qquad \text{c)} \quad \sum_{n=1}^{\infty} \frac{x^n}{n(n+1)}\,.$$

Řešení. a) Danou mocninnou řadu lze sečíst jako geometrickou řadu s kvocientem x, kde $|x| < 1$. Dostaneme

$$\sum_{n=0}^{\infty} x^n = 1 + x + x^2 + \cdots + x^n + \cdots = \frac{1}{1-x}\,, \qquad |x| < 1.$$

Další možností je použití procedury `sum`, která slouží k výpočtu součtu řady. U nekonečných řad se Maple snaží najít součet řady. V případě, že se mu součet nepodaří spočítat, vrátí pouze její předpis.

Pro přehlednější formu zápisu lze využít proceduru `Sum`, která má stejné použití, avšak nepočítá součet řady a slouží pouze pro vypsání jejího předpisu.

Použití zmíněných procedur je snadné a bude dobře patrné při řešení tohoto příkladu.

```
> Sum(x^n, n=1..infinity)=sum(x^n, n=1..infinity);
```

$$\sum_{n=1}^{\infty} x^n = -\frac{1}{x-1}$$

Až na zápis se oba součty rovnají.

b) Využijeme důsledek 1.2 věty 1.2. Pro $|x| < 1$ platí

$$\sum_{n=1}^{\infty} \frac{x^n}{n} = \sum_{n=1}^{\infty} \int_0^x t^{n-1}\,\mathrm{d}t = \int_0^x \sum_{n=1}^{\infty} t^{n-1}\,\mathrm{d}t = \int_0^x \frac{1}{1-t} = -\ln(1-x)\,.$$

Nyní provedeme opět kontrolní součet řady s využitím Maple. V tomto případě však zvolíme další možnost použití procedury `Sum` spolu s procedurou `value`, která dokáže předpisy řad (limit, integrálů, diferenciálních rovnic, ...) vyhodnotit. Ke zvýšení efektivnosti zápisu použijeme systémovou proměnnou označovanou symbolem „`%`“, která odkazuje na poslední vyhodnocený výsledek.

```
> Sum(x^n/n, n=1..infinity):%=value(%);
```

$$\sum_{n=1}^{\infty} \frac{x^n}{n} = -\ln(1-x)$$

Oba výsledky se shodují.

c) Postup řešení je obdobný. Z využitím rovnosti $\int\limits_0^x \frac{t^n}{n}\,\mathrm{d}t = \frac{x^{n+1}}{n(n+1)}$ dále platí

$$\sum_{n=1}^{\infty} \frac{x^n}{n(n+1)} = \sum_{n=1}^{\infty} \frac{1}{x} \int\limits_0^x \frac{t^n}{n}\,\mathrm{d}t = \frac{1}{x} \sum_{n=1}^{\infty} \int\limits_0^x \frac{t^n}{n} = \frac{1}{x} \int\limits_0^x \sum_{n=1}^{\infty} \frac{t^n}{n} = -\frac{1}{x} \int\limits_0^x \ln(1-t)\,\mathrm{d}t\,.$$

Pro výpočet integrálu nabízí Maple standardní proceduru `int`. Této proceduře se zadávají dva parametry. Prvním je integrovaný výraz a druhým neznámá, vzhledem ke které se bude výraz integrovat. Pro výpočet určitého integrálu navíc specifikujeme interval.

Pro pouhý výpis integrálu slouží procedura `Int`. Podobně jako `Sum`, lze i `Int` uvést v kombinaci s procedurou `value`.

```
> Int(ln(1-t), t=0..x):%=value(%);
```

$$\int\limits_0^x \ln(1-t)\,\mathrm{d}t = -\ln(1-x) + \ln(1-x)x - x$$

Tudíž platí

$$\sum_{n=1}^{\infty} \frac{x^n}{n(n+1)} = \frac{\ln(1-x) - \ln(1-x)x + x}{x} = 1 - \frac{(x-1)\ln(1-x)}{x}\,.$$

Součet zadané řady vypočteme v Maple i přímo.

```
> Sum(x^n/(n*(n+1)), n=1..infinity):%=value(%);
```

$$\sum_{n=1}^{\infty} \frac{x^n}{n(n+1)} = 1 - \frac{(x-1)\ln(1-x)}{x}$$

Oba výsledky se opět shodují.

1.2 Poloměr konvergence metodou „krok za krokem“

Při metodě „krok za krokem“ budeme postupovat klasickou cestou hledání řešení, používanou při ručním provádění výpočtu. Maple budeme využívat jako výpočetní nástroj k získávání mezivýsledků.

Pro výpočet limity funkce slouží v Maple procedura `limit`. V základní podobě má dva parametry. První parametr udává funkci, pro kterou se limita počítá. Druhý parametr definuje hodnotu nezávisle proměnné, pro niž se limita počítá. Při výpočtu nevlastních limit se užívá konstanta `infinity` reprezentující nekonečno.

Pro zefektivnění výstupů můžeme opět kombinovat proceduru `Limit`, která danou limitu vypíše v tradiční matematické formě, s procedurou `value`.

Příklad 1.2. Určete poloměr a interval konvergence následujících mocninných řad:

$$\text{a)}\quad \sum_{n=1}^{\infty} \frac{n\,x^n}{n!} \qquad \text{b)}\quad \sum_{n=1}^{\infty} 3^n n\,x^n\,.$$

Řešení. a) Nadefinujeme n-tý člen zadané řady jako funkci proměnné n.

```
> a:=n->n/n!;
```

$$a := n \to \frac{n}{n!}$$

Podle poznámky 1.2 pro poloměr konvergence platí $r = \lim\limits_{n\to\infty} \left|\frac{a_n}{a_{n+1}}\right|$.

```
> Limit(abs(a(n)/a(n+1)), n=infinity):%=value(%);
```

$$\lim \left|\frac{n(n+1)!}{n!(n+1)}\right| = \infty$$

Poloměr konvergence je roven ∞, tudíž řada konverguje absolutně.

b) Postupujeme analogicky. Nadefinujeme člen a_n.

```
> a[n]:=3^n*n;
```

$$a_n := 3^n n$$

Podle poznámky 1.2 pro poloměr konvergence také platí $r = \frac{1}{\lim\limits_{n\to\infty} \sqrt[n]{|a_n|}}$.

```
> 1/Limit(abs(a[n])^(1/n), n=infinity):%=value(%);
```

$$\frac{1}{\lim\limits_{n\to\infty} \sqrt[n]{|3^n n|}} = \frac{1}{3}$$

Poloměr konvergence je roven $\frac{1}{3}$, konvergenční interval je proto $(-\frac{1}{3}, \frac{1}{3})$.

1.3 Automatizace

Prvním cílem automatizace je zjednodušit a zobecnit klasický postup řešení metody „krok za krokem“ a implementovat jej do nové procedury. Druhým cílem je schopnost řešit nejenom jeden zadaný příklad, nýbrž celou skupinu příkladů, které spolu souvisejí, na základě nějaké společné vlastnosti. Používání hotové procedury vyžaduje jen minimální znalosti programu Maple, proto se hodí i pro začínajícího uživatele. Další velkou předností procedury je rychlost, kterou v případě, že potřebujeme jen výsledek výpočtu, ocení i zkušení uživatelé. Jako příklad efektivního užití procedury uveďme rychlou kontrolu výsledků „ručního“ počítání.

Realizace nutně závisí v nalezení algoritmu, který bude řešit daný typ úloh. Jistě si dokážeme uvědomit, že ne každý matematický postup se dá algoritmizovat, tudíž ani automatizovat.

Obvyklým postupem při automatizování rozsáhlejších problémů je rozložení na části, jejichž řešení bývá snadnější. Zpětnou kompozicí již dostaneme řešení celého problému.

1.4 Procedura `ConvergenceRadius`

Metoda „krok za krokem“ s asistencí Maple představuje při výpočtu poloměru konvergence převážně ulehčení práce při řešení limit. Výpočet poloměru konvergence budeme potřebovat i ve zbylém textu kapitoly při vyšetřování oboru konvergence mocninných řad. Proto je výpočet vhodné automatizovat procedurou, jejíž užití je snadnější a efektivnější.

Po zavedení nové procedury se jistě nabízí otázka ohledně konkurenceschopnosti. Samotný Maple proceduru řešící tento problém neobsahuje. To však neznamená, že žádné nejsou k dispozici.

V následujícím textu si proto zavedeme novou proceduru `ConvergenceRadius` a na vybraných příkladech si prezentujeme její funkčnost a porovnáme ji s procedurou `Polomer` uvedenou v [4] a procedurou `PSconv` obsaženou v matematické knihovně `math` [10].

`ConvergenceRadius(PSeries)`

Povinný parametr:

`PSeries` – mocninná řada, jejíž zápis předpokládáme ve tvaru $\sum\limits_{n=1}^{\infty} a_n x^n$, případně ve tvaru $\sum\limits_{n=1}^{\infty} a_n \left(x - x_0\right)^n$. Dovolený je i zápis ve tvaru $a_n x^n$, případně $a_n \left(x - x_0\right)^n$.

Volitelný parametr:

`1` = {`quo, root`} – jedná se o první poziční parametr, jehož hodnota se uvádí v hlavičce procedury až za povinným parametrem `PSeries`. Parametr určuje jaké kritérium má být při výpočtu použito. `Quo` – podílové kritérium. `Root` – odmocninové kritérium. Pokud není parametr uveden, pracuje procedura v režimu, který automaticky zvolí vhodnější kritérium.

Příklad 1.3. Užitím procedury `ConvergenceRadius` určete poloměr konvergence následujících řad:

$$\text{a)}\quad \sum_{n=1}^{\infty} \frac{(n!)^2}{(2n)!} x^n \qquad \text{b)}\quad \sum_{n=1}^{\infty} 2^n x^{2n} \qquad \text{c)}\quad \sum_{n=1}^{\infty} \frac{(-1)^n (x+2)^n}{n+\sqrt{n}}\,.$$

Výsledek porovnejte s procedurami `Polomer` a `PSconv`.

Řešení. a) Nadefinujeme mocninnou řadu.

```
> rada:=Sum((n!)^2/(2*n)!*x^n, n=1..infinity);
```

$$rada := \sum_{n=1}^{\infty} \frac{(n!)^2}{(2n)!} x^n$$

Řešíme s využitím uvedené procedury `ConvergenceRadius`.

```
> ConvergenceRadius(rada);
```

$$4$$

Porovnání výsledku nejprve provedeme procedurou `Polomer`.

```
> Polomer(op(1, rada));
```

$$4$$

A nyní pomocí procedury `PSconv`. Předpokládáme, že knihovna `math` je řádně nainstalována. Před použitím procedury `PSconv` musíme zavést do paměti její definici pomocí příkazu `with`.

```
> with(math):
> PSconv(rada);
```

$$4$$

Všechny tři výpočty jsou si rovny, poloměr konvergence je 4.

b) Při řešení budeme postupovat obdobně jako u varianty a). Nadefinujeme mocninnou řadu a řešíme procedurou `ConvergenceRadius`.

```
> rada:=Sum(2^n*x^(2*n), n=1..infinity);
```

$$rada := \sum_{n=1}^{\infty} 2^n x^{2n}$$

```
> ConvergenceRadius(rada);
```

$$\frac{\sqrt{2}}{2}$$

Získaný výsledek opět porovnáme. Nejprve s procedurou `Polomer`.

```
> Polomer(op(1, rada));
```

$$\frac{\sqrt{2}}{2}$$

Použijeme-li ovšem při řešení proceduru `PSconv`, dostáváme chybný výsledek.

```
> PSconv(rada);
```

$$\frac{1}{2}$$

c) Postupujeme analogicky.

```
> rada:=Sum((-1)^n*(x+2)^n/(n+sqrt(n)), n=1..infinity);
```

$$rada := \sum_{n=1}^{\infty} \frac{(-1)^n (x+2)^n}{n+\sqrt{n}}$$

```
> ConvergenceRadius(rada);
```

$$1$$

Při řešení procedurou `Polomer` dostáváme nečekaný výsledek ve formě zápisu limity.

```
> Polomer(op(1, rada));
```

$$\lim_{n\to\infty} \left| \frac{(-1)^n \left(n+1+\sqrt{n+1}\right)}{(n+\sqrt{n})\,(-1)^{n+1}} \right|$$

Z výstupu je patrné, že procedura při výpočtu použila podílového kritéria. Při řešení však pravděpodobně pracuje s tak neupraveným výrazem s jehož výpočtem si Maple nedokáže poradit. Je tedy nutná asistence, buď během výpočtu, čímž bychom byli nuceni zasahovat do zdrojového kódu, nebo se pokusit výstup vhodně upravit. Ke zjednodušování výrazů slouží standardní procedura `simplify`. Použitím procedury `simplify` na výstup procedury `Polomer` již dostáváme správný výsledek.

```
> simplify(%);
```

$$1$$

Naopak s touto řadou si dokáže poradit procedura `PSconv`, která vrací korektní výsledek.

```
> PSconv(rada);
```

$$1$$

Z poznatků získaných nejen při řešení tohoto příkladu, ale zejména během širšího testování, se ukázalo, že procedura `ConvergenceRadius` vykazuje dobré početní výsledky a v porovnání s konkurenčními procedurami i nejširší možnost využití. Tudíž ji budeme používat při řešení poloměru konvergence i ve zbytku této kapitoly.

1.5 Obor konvergence metodou „krok za krokem“

Při vyšetřování oboru konvergence budeme postupovat metodou „krok za krokem“, jako bychom výpočet prováděli ručně. Maple budeme opět využívat jako výpočetní nástroj k řešení mezivýsledků. Při výpočtu využijeme proceduru `ConvergenceRadius`, ukážeme si určování konvergence číselných řad jak procedurou `sum` tak procedurou `csum`, která je součástí knihovny `Maple Advisor Database` [7].

Příklad 1.4. Určete poloměr a obor konvergence následujících mocninných řad:

$$\text{a)}\quad \sum_{n=1}^{\infty} \frac{x^n}{n(n+1)} \qquad \text{b)}\quad \sum_{n=1}^{\infty} \frac{(-1)^{n-1}\, x^{2n}}{\sqrt{n}\,(2n-1)}\,.$$

Řešení. a) Nadefinujeme mocninnou řadu.

```
> rada:=Sum(x^n/(n*(n+1)), n=1..infinity);
```

$$rada := \sum_{n=1}^{\infty} \frac{x^n}{n(n+1)}$$

Procedurou `ConvergenceRadius` určíme poloměr konvergence.

```
> ConvergenceRadius(rada);
```

$$1$$

Konvergenční interval je proto $(-1, 1)$.

Nyní vyšetříme krajní body tohoto intervalu. Dosazením těchto bodů do dané mocninné řady dostaneme číselné řady. S využitím procedury `sum` se pokusíme najít jejich součty a tím rozhodnout o jejich konvergenci či divergenci.

Dosazením bodu $x = -1$ dostaneme řadu $\sum\limits_{n=1}^{\infty} \frac{(-1)^n}{n(n+1)}$. Vypočteme její součet.

```
> Sum((-1)^n/(n*(n+1)), n=1..infinity):%=value(%);
```

$$\sum_{n=1}^{\infty} \frac{(-1)^n}{n(n+1)} = 1 - \ln 2$$

Obdobně pro bod $x = 1$ dostaneme řadu $\sum\limits_{n=1}^{\infty} \frac{1}{n(n+1)}$. Spočteme součet této řady.

```
> Sum(1/(n*(n+1)), n=1..infinity):%=value(%);
```

$$\sum_{n=1}^{\infty} \frac{1}{n(n+1)} = 1$$

Z teorie popsané v [3] vyplývá, že obě číselné řady jsou konvergentní. Oborem konvergence je tedy interval $[-1, 1]$.

b) Postupujeme analogicky. Vyjádříme mocninnou řadu v potřebném tvaru.

```
> rada:=Sum((-1)^(n-1)*x^(2*n)/sqrt(n)/(2*n-1), n=1..infinity);
```

$$rada := \sum_{n=1}^{\infty} \frac{(-1)^{n-1}\, x^{2n}}{\sqrt{n}\,(2n-1)}$$

Vypočteme poloměr konvergence.

```
> ConvergenceRadius(rada);
```

$$1$$

Středem této řady je $x_0 = 0$, tudíž řada konverguje na intervalu $(-1, 1)$.

Dosazením obou krajních bodů konvergenčního intervalu dostaneme číselnou řadu $\sum_{n=1}^{\infty} \frac{(-1)^{n-1}}{\sqrt{n}(2\,n-1)}$. Pokusíme se spočítat její součet pomocí procedury `sum`.

```
> soucet[[-1,1]]:=value(subs(x=1, rada));
```

$$soucet_{[-1,1]} := \sum_{n=1}^{\infty} \frac{(-1)^{n-1}}{\sqrt{n}\,(2\,n-1)}$$

Procedura `sum` vrátila pouze předpis řady, tudíž Maple nedokáže součet spočítat. Pokusíme se proto o konvergenci či divergenci této řady rozhodnout pomocí procedury `csum`[1]. Procedura `csum` je součástí rozsáhlé knihovny `Maple Advisor Database`, nicméně její instalace není nutná. Autor nabízí na svých stránkách zdrojový kód k dispozici volně ke stažení[2].

Výstupní hodnota může nabývat hodnoty *true* – řada konverguje, *false* – řada diverguje a hodnoty *FAIL* – kdy nedokáže rozhodnout.

Předpokládáme, že procedura je dostupná v textovém souboru `csum5.txt`. Nyní již stačí její definici načíst do paměti pomocí standardní procedury `read`.

```
> read "csum5.txt";
> csum(op(1, subs(x=1, rada)), n);
```

$$true$$

V obou krajních bodech intervalu řada konverguje, tudíž obor konvergence je $[-1, 1]$.

[1]Kompletní nápověda obsahující popis a vzorové příklady je dostupná na adrese `http://www.math.ubc.ca/~israel/advisor/advisor6/h35r1.htm`.

[2]Definice procedury je dostupná na adrese `http://www.math.ubc.ca/~israel/advisor/advisor6/csum5.txt`.

1.6 Procedura `ConvergenceTestOfPowerSeries`

V předchozím odstavci jsme prezentovali výpočet oboru konvergence pomocí metody „krok za krokem“. Z příkladu 1.4 je patrné, že Maple při výpočtu nevyžaduje asistenci. Je tedy výhodné celý postup řešení automatizovat.

Ve zbylém textu kapitoly si uvedeme a na vybraných příkladech představíme novou proceduru `ConvergenceTestOfPowerSeries`, jejíž základní výpočetní jádro tvoří již dříve uvedené procedury `ConvergenceRadius` a `csum`.

Procedura slouží nejenom k výpočtu oboru konvergence, ale dokáže i graficky znázornit chování částečných součtů mocninné řady na konvergenčním intervalu. Jednotlivé grafy částečných součtů jsou od sebe barevně odlišeny přechodem od modré po červenou barvu. Konvergenční interval je v grafu pro přehlednost znázorněn barevným pruhem.

`ConvergenceTestOfPowerSeries(PSeries)`

Povinný parametr:

`PSeries` – mocninná řada, jejíž zápis předpokládáme ve tvaru $\sum\limits_{n=1}^{k} a_n x^n$, případně ve tvaru $\sum\limits_{n=1}^{k} a_n (x - x_0)^n$. Dovolený je i zápis ve tvaru $a_n x^n$, případně $a_n (x - x_0)^n$. V tomto případě je však automaticky doplněn na prvně uvedený tvar s implicitně nastavenou hodnotou $k = 10$.

Volitelné parametry:

`output` = {`convergence, plot, animation`}[3] – specifikuje výstupní formu procedury. Volba `convergence` vrací obor konvergence a jako jediná představuje textový výstup. Zbylé hodnoty `plot`, resp. `animation` slouží ke grafickému znázornění částečných součtů řady a to buď společně, do jednoho obrázku, nebo v animaci. Implicitní nastavení je `output=convergence`.

`view` = [$\mathtt{x_{min}..x_{max}}$, $\mathtt{y_{min}..y_{max}}$] – dvojice celočíselných intervalů určuje rozsah zobrazovaných hodnot v grafu na osách x a y. Ve většině zkoumaných případů nás zajímá chování částečných součtů v okolí konvergenčního intervalu, tudíž implicitně je zvoleno nastavení `view=[`$\mathtt{x_0}$`−2r..`$\mathtt{x_0}$`+2r,−5..5]`, kde `r` je poloměr konvergence a $\mathtt{x_0}$ je střed mocninné řady.

`step` – kladná celočíselná hodnota, určuje rozdíl v indexech dvou po sobě následujících částečných součtů řady. Implicitně je roven jedné. Nastavení počátečního a koncového indexu není nutné, jejich hodnoty vycházejí z rozsahu, který je definován parametrem n v mocninné řade.

`1` = {`PSsum`} – první poziční parametr rozšiřuje grafický výstup o zobrazení grafu součtu zadané řady. Graf součtu je definován pouze na konvergenčním intervalu, v obrázku je znázorněn tučnou žlutou čarou.

[3]Parametr může nabývat pouze jediné uvedené hodnoty.

Příklad 1.5. Určete obor konvergence mocninné řady

$$\sum_{n=1}^{\infty} \frac{x^n}{n}.$$

Řešte pomocí procedury `ConvergenceTestOfPowerSeries`. Výsledek doplňte obrázkem obsahujícím grafy částečných součtů s_n této řady pro $n = 5, \ldots, 30$ s indexovým krokem 5.

Řešení. Nadefinujeme řadu, kde omezíme rozsah indexové proměnné n intervalem $5, \ldots, 30$. Pro výpočet oboru konvergence je tento údaj nepodstatný. Nicméně doplňující část úkolu hovoří o zobrazení částečných součtů řady, tudíž je nezbytné index n omezit konečným intervalem.

```
> rada:=Sum(x^n/n, n=5..30);
```

$$rada := \sum_{n=5}^{30} \frac{x^n}{n}$$

Nejprve určíme obor konvergence.

```
> ConvergenceTestOfPowerSeries(rada);
```

Obor konvergence je interval:

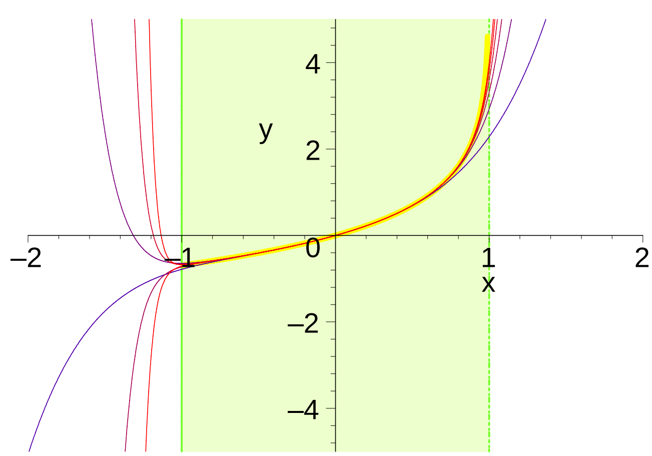
$$[-1, 1)$$

Pro kreslení grafů částečných součtů použijeme předchozí proceduru s volbu `output=plot`. Volbou `step=5` omezíme interval pouze na vybrané částečné součty $s_5, s_{10}, s_{15}, s_{20}, s_{25}, s_{30}$. Doplňující parametr `PSsum` vykreslí na pozadí také graf součtu mocninné řady.

```
> ConvergenceTestOfPowerSeries(rada, output=plot, step=5, PSsum);
```

Obor konvergence je interval:

$$[-1, 1)$$



Obr. 1.1: Částečné součty $s_5, s_{10}, s_{15}, s_{20}, s_{25}, s_{30}$ řady $\sum \frac{x^n}{n}$

Součet zadané řady zde uvádět nebudeme, je k dispozici na straně 6. Připomeňme jen, že tento součet je definován na konvergenčním intervalu $[-1, 1)$, tudíž graf součtu je jeho pásem omezen. Krajní čáry pásu ukazují, dle svého zobrazení, zda řada v krajních bodech intervalu konverguje (tučná čára), resp. diverguje (čerchovaná čára).

Všimněme si chování částečných součtů převážně v okolí krajních bodů intervalu. Na obrázku je patrné, že v levém okolí bodu -1 součty oscilují a rovnají se ∞, resp. $-\infty$. Naopak v bodě 1 lze dobře pozorovat divergenci řady, kdy grafy součtů utíkají do ∞.

Příklad 1.6. Užitím procedury `ConvergenceTestOfPowerSeries` určete obor konvergence mocninné řady

$$\sum_{n=1}^{\infty}(n+1)3^{n-1}x^{n-1}\,.$$

Řešení doplňte animací grafů částečných součtů s_n této řady pro $n = 3, \ldots, 8$.

Řešení. Nadefinujeme mocninnou řadu. Příkaz ukončíme dvojtečkou. Příkaz se provede standardně, ale jeho výstup bude potlačen.

```
> rada:=Sum((n+1)*3^(n-1)*x^(n-1), n=3..8):
```

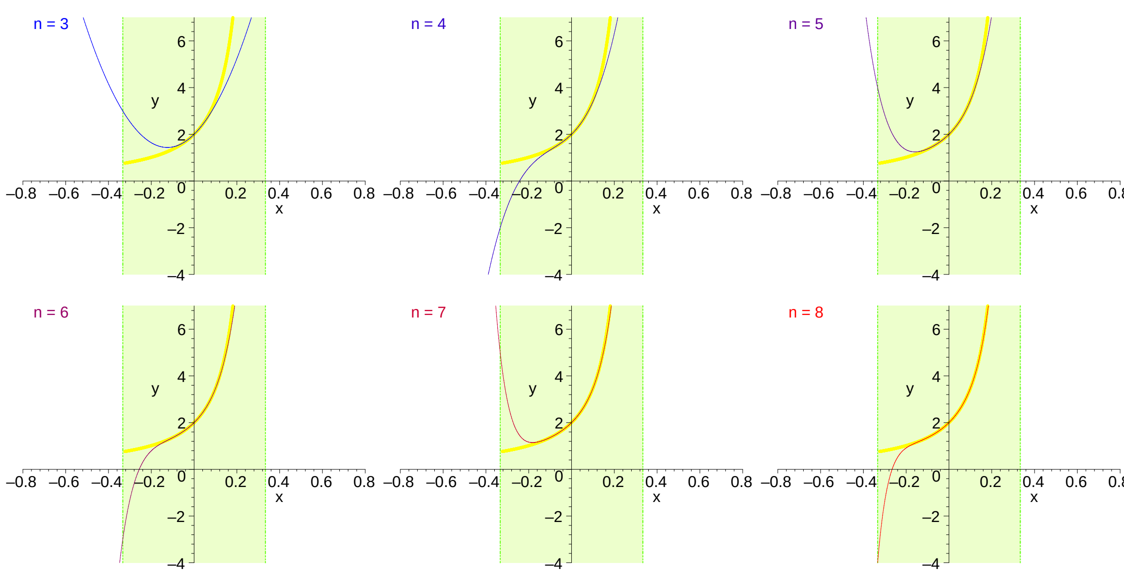
Animaci částečných součtů s_n vytvoříme pomocí volby `output=animation`. Pro porovnání se pokusíme vykreslit i součet zadané řady volbou `PSsum` a rozsah zobrazovaných hodnot upravíme proměnnou `view`.

Animaci lze zobrazit pouze na počítači, tudíž celou animaci rozložíme do šesti snímků (viz obrázek 1.2).

```
> ConvergenceTestOfPowerSeries(rada, output=animation, PSsum,
  view=[-0.8...0.8,-4..7]);
```

Obor konvergence je interval:

$$\left(-\frac{1}{3}, \frac{1}{3}\right)$$



Obr. 1.2: Náhled z animace částečných součtů s_n řady $\sum(n+1)3^{n-1}x^{n-1}$ pro $n = 3, \ldots, 8$

Příklad 1.7. Určete obor konvergence následujících řad.

a) $\displaystyle\sum_{n=1}^{\infty} \frac{x^n}{n10^{\,n-1}}$ b) $\displaystyle\sum_{n=1}^{\infty} \left(1+\frac{1}{n}\right)^{n^2} x^n$ c) $\displaystyle\sum_{n=1}^{\infty} \frac{(n!)^2 x^n}{(2n)!}$

d) $\displaystyle\sum_{n=1}^{\infty} \frac{3^n+(-2)^n}{n}(x+1)^n$ e) $\displaystyle\sum_{n=1}^{\infty} \frac{3^{-\sqrt{n}}x^n}{\sqrt{n^2+1}}$ f) $\displaystyle\sum_{n=1}^{\infty} \frac{(-1)^n(x+2)^n}{n+\sqrt{n}}$.

U variant a) a f) řešení vhodně modifikujte a zobrazte i grafy částečných součtů s_n. Pro a) $n = 1,\dots,10$. Pro f) $n = 5,\dots,20$ s krokovým indexem 3. Při řešení využijte proceduru `ConvergenceTestOfPowerSeries`.

Řešení. a) Nadefinujeme mocninnou řadu.

```
> rada:=Sum(x^n/(n*10^(n-1)), n=1..10);
```

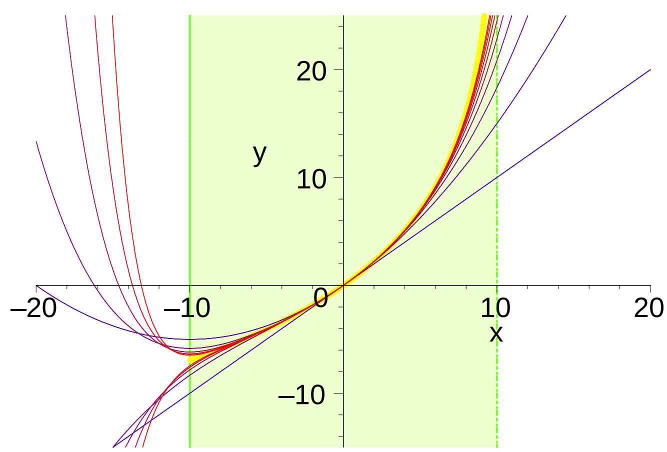
$$rada := \sum_{n=1}^{10} \frac{x^n}{n10^{\,n-1}}$$

Řešení realizujeme v jednom kroku. Vypočteme obor konvergence řady a vykreslíme grafy částečných součtů s_n pro $n = 1,\dots,10$. Volbou `PSsum` se pokusíme spočítat součet řady a zobrazit jeho graf na pozadí. Rozsah zobrazovaných hodnot pro osy x a y nadefinujeme proměnnou `view`.

```
> ConvergenceTestOfPowerSeries(rada, output=plot, PSsum,
  view=[-20..20, -15..25]);
```

Obor konvergence je interval:

$$[-10, 10)$$



Obr. 1.3: Částečné součty s_n řady $\sum \frac{x^n}{n10^{\,n-1}}$ pro $n = 1,\dots,10$

b) Nadefinujeme mocninnou řadu a určíme obor konvergence.

```
> rada:=Sum((1+1/n)^(n^2)*x^n, n=1..infinity);
```

$$rada := \sum_{n=1}^{\infty} \left(1 + \frac{1}{n}\right)^{n^2} x^n$$

```
> ConvergenceTestOfPowerSeries(rada);
```

Obor konvergence je interval:

$$\left(-\frac{1}{e}, \frac{1}{e}\right)$$

c) Postupujeme analogicky.

```
> rada:=Sum(n!^2*x^n/(2*n)!, n=1..infinity);
```

$$\sum_{n=1}^{\infty} \frac{(n!)^2 x^n}{(2n)!}$$

```
> ConvergenceTestOfPowerSeries(rada);
```

Obor konvergence je interval:

$$(-4, 4)$$

d) Nyní si ukážeme druhou formu stručnějšího zápisu mocninné řady. Řadu nadefinujeme pouze ve tvaru $a_n x^n$. Nutno připomenout, že tento zápis není přesným určením řady, tudíž se jeho použití doporučuje pouze pro textový výstup při výpočtu oboru konvergence.

```
> rada:=(3^n+(-2)^n)/n*(x+1)^n;
```

$$rada := \frac{3^n + (-2)^n}{n}(x+1)^n$$

```
> ConvergenceTestOfPowerSeries(rada);
```

Obor konvergence je interval:

$$\left(-\frac{4}{3}, -\frac{2}{3}\right)$$

e) Postupujeme analogicky.

```
> rada:=3^(-sqrt(n))*x^n/sqrt(n^2+1);
```

$$rada := \frac{3^{-\sqrt{n}} x^n}{\sqrt{n^2+1}}$$

```
> ConvergenceTestOfPowerSeries(rada);
```

Obor konvergence je interval:

$$[-1, 1]$$

f) Určíme obor konvergence, zobrazíme grafy částečných součtů a pokusíme se na pozadí vykreslit i součet řady.

```
> rada:=Sum((-1)^n*(x+2)^n/(n+sqrt(n)), n=5..20);
```

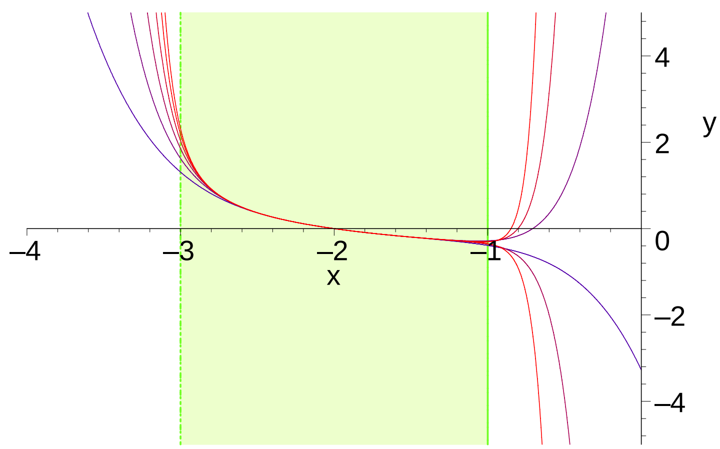
$$rada := \sum_{n=5}^{20} \frac{(-1)^n (x+2)^n}{n+\sqrt{n}}$$

```
> ConvergenceTestOfPowerSeries(rada, step=3, output=plot, PSsum);
```

Obor konvergence je interval:

$$(-3, 1]$$

Upozornění: řadu Sum((-1)^n*(x+2)^n/(n+n^(1/2)), n = 1..infinity) nelze sečíst.



Obr. 1.4: Částečné součty $s_5, s_8, s_{11}, s_{14}, s_{17}, s_{20}$ řady $\sum \frac{(-1)^n(x+2)^n}{n+\sqrt{n}}$

Všimněme si upozornění umístěného před obrázkem, kdy nás procedura informuje o tom, že řadu nelze sečíst. Nejedná se o upozornění na chybu vzniklou v průběhu výpočtu či na jiné selhání, ale pouze o doprovodnou informaci o stavu výpočtu. V tomto případě se nepodařilo dostupnými metodami programu Maple součet dané řady vyřešit, tudíž její graf v obrázku chybí.

Kapitola 2

Rozvoj funkcí do mocninných řad

V této kapitole budeme vyšetřovat rozvoj funkcí do mocninných řad. Zavedení pojmů Taylorova řada, Maclaurinova řada a Taylorův zbytek je věnována první část.

Metodou „krok za krokem“ ukážeme výpočet Taylorových polynomů. Poté celý postup zobecníme novou funkcí `TaylorPolynomial` a srovnáme s procedurou `taylor`, která je součástí Maple.

Zbývající sekce plynně navazují na část předcházející a využívají další možnost, kterou nám Maple nabízí. Jedná se o vizualizaci. Uvedeme význam standardních procedur `plot` a `display`, které slouží pro práci s grafikou v Maple.

Na řešených příkladech vysvětlíme metodou „krok za krokem“ základní dovednosti nutné pro zobrazování grafů Taylorových, resp. Maclaurinových polynomů a tvorbu animací. Jedná se o jakousi předmluvu, na kterou plynně navazuje uvedení nové automatizované procedury `TaylorPolynomialAnimation`.

2.1 Taylorova a Maclaurinova řada

Připomeňme Taylorovu větu z diferenciálního počtu, kdy je funkce vyjádřena ve tvaru polynomu a zbytku.

Věta 2.1 (Taylorova věta). *Nechť f je funkce, pro kterou existují derivace až do řádu $n+1$ v uzavřeném intervalu I, jehož krajní body jsou čísla x a x_0. Pak platí*

$$f(x) = f(x_0) + \frac{f'(x_0)}{1!}(x-x_0) + \cdots + \frac{f^{(n)}(x_0)}{n!}(x-x_0)^n + R_n(x),$$

kde $R_n(x)$ je Taylorův zbytek, pro který platí

$$R_n(x) = \frac{(x-x_0)^{n+1}}{(n+1)!} f^{(n+1)}(\vartheta), \quad \text{kde } \vartheta \in I, \vartheta \neq x, x_0. \tag{2.1}$$

Přirozeně se nabízí zavést následující definici:

Definice 2.1. Nechť funkce f má v bodě x_0 derivace všech řádů. Mocninnou řadu

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

nazýváme *Taylorovou řadou* funkce f v bodě x_0.

Je-li $x_0 = 0$, mluvíme o Maclaurinově řadě, která je tedy tvaru $\sum_{n=0}^{\infty} \frac{f^{(n)}(0)}{n!}x^n$.

Obecně nemusí platit, že součet Taylorovy řady funkce f je roven této funkci. Následující věta udává podmínku, kdy tato rovnost platí.

Věta 2.2. *Nechť funkce f má v nějakém bodě x_0 derivace všech řádů. Pak platí*

$$f(x) = \sum_{n=0}^{\infty} \frac{f^{(n)}(x_0)}{n!}(x-x_0)^n \tag{2.2}$$

na intervalu I obsahujícím bod x_0 právě tehdy, když pro posloupnost $\{R_n(x)\}$ Taylorových zbytků platí $\lim\limits_{n\to\infty} R_n(x) = 0$ pro všechna $x \in I$.

2.2 Taylorův polynom metodou „krok za krokem“

Budeme postupovat „krok za krokem“, jako bychom výpočet prováděli ručně. Maple budeme opět využívat jako výpočetní nástroj k řešení mezivýsledků.

Příklad 2.1. Určete Taylorův polynom 3. stupně funkce

$$\ln(1+x)$$

se středem v bodě 0.

Řešení. Nadefinujeme funkci.

```
> f:=x->ln(1+x);
```

$$f := x \to \ln(1+x)$$

Spočteme potřebné derivace.

```
> derivace[1]:=(D)(f);
```

$$derivace_1 := x \to \frac{1}{1+x}$$

```
> derivace[2]:=(D@@2)(f);
```

$$derivace_2 := x \to -\frac{1}{(1+x)^2}$$

```
> derivace[3]:=(D@@3)(f);
```

$$derivace_3 := x \to 2\frac{1}{(1+x)^3}$$

Podle věty 2.2 platí:

```
> TP[3]:=f(0)+derivace[1](0)*x+derivace[2](0)*x^2/2+derivace[3](0)*x^3/6;
```

$$TP_3 := x - \frac{1}{2}x^2 + \frac{1}{3}x^3$$

Taylorův polynom 3. stupně se středem v bodě 0 funkce $\ln(1+x)$ je tvaru $x - \frac{1}{2}x^2 + \frac{1}{3}x^3$.

2.3 Funkce `TaylorPolynomial`

Předchozí postup lze jednoduše zobecnit pro libovolnou funkci splňující předpoklady definice.

```
> TaylorPolynomial:=(f, x0, n)->sum((D@@i)(f)(x0)/i!*(x-x0)^i, i=0..n);
```

$$TaylorPolynomial := (f, x0, n) \to \sum_{i=0}^{n} \frac{(D^{(i)}(f)(x0)(x-x0)^i}{i!}$$

Pomocí této funkce vyřešíme předcházející příklad 2.1.

```
> TP[3]:=TaylorPolynomial(f, 0, 3);
```

$$TP_3 := x - \frac{1}{2}x^2 + \frac{1}{3}x^3$$

Ke kontrole výpočtu můžeme použít předdefinovanou proceduru `taylor`, kterou voláme příkazem `taylor(f, eqn, n)`, kde `eqn` je rovnice tvaru $x = c$, c je střed Taylorova polynomu. Pro výpočet Taylorova polynomu se středem v bodě 0 lze zápis $x = 0$ zkrátit pouhým x. Pro takto zadané n platí, že je-li $T(x)$ Taylorův polynom stupně $n-1$ a $R(x) = |f(x) - T(x)|$, pak $\lim_{x\to 0} \frac{R(x)}{x^n} < \infty$. Výsledkem je datová struktura typu `series`. Převod na datový typ `polynom` provedeme příkazem `convert`.

Řešit příklad 2.1 použitím příkazu `taylor` lze takto:

```
> TP[3]:=convert(taylor(f(x), x=0, 4), polynom);
```

$$TP_3 := x - \frac{1}{2}x^2 + \frac{1}{3}x^3$$

Porovnáním s řešením příkladu 2.1 je vidět, že výsledky se shodují.

2.4 Animace metodou „krok za krokem“

Následující příklady prezentují rozsáhlé možnosti Maple pro zobrazování grafů funkcí. Toho využijeme pro další zpracování Taylorových polynomů. Jedná se o grafický výstup buď ve formě klasických grafů nebo animací.

Procedury `plot` a `display`

V tomto odstavci uvedeme základní možnosti grafické vizualizace v Maple. Základním příkazem pro vytvoření dvourozměrných grafů je příkaz `plot`.

V jeho nejjednodušší variantě mu stačí zadat funkci nebo i výraz, jejichž graf má vytvořit, a interval nezávisle proměnné, kterým je omezen horizontální rozsah grafu.

Oba následující příkazy vykreslí graf funkce $f(x) = \cos x, x \in [-\pi, \pi]$.

```
> plot(cos(x), x=-Pi..Pi);
> f:=x->cos(x): plot(f(x), x=-Pi..Pi);
```

Vertikální rozsah grafu je automaticky zvolen tak, aby se všechny hodnoty funkce na daném intervalu podařilo zobrazit. I ten však můžeme nastavit třetím volitelným parametrem.

```
> plot(cos(x), x=-Pi..Pi, -2..2);
```

Jestliže již umíme vytvářet grafy funkcí, je často vhodné umístit více grafů do jednoho obrázku. Příkazu `plot` lze tudíž zadat seznam nebo množinu funkcí či výrazů, jejichž grafy se vykreslí do jednoho obrázku.

```
> plot([sin(x),cos(x)], x=-Pi..Pi, -2..2);
```

Výslednou podobu grafu lze ovlivnit velkým množstvím voleb. Uvedeme si zde pouze některé. Jejich celkový přehled nalezneme v nápovědě příkazem `?plot[options]`.

Pokud má obrázek obsahovat více grafů, je vhodné grafy jednotlivých funkcí od sebe odlišit. Ideální je volba `color`, která určuje jakou barvou se má graf vykreslit. Barvu lze zadat buďto z nabídky základních barev slovním popisem (blue, red, green apod.) nebo její RGB hodnotou.

Volbou `scaling` lze nastavit poměr os. Pokud chceme zachovat stejná měřítka na osách, použijeme volbu `scaling=constrained`.

Tloušťku kreslené čáry ovlivňuje volba `thickness`, jejíž hodnota může nabývat hodnot 0 až 15.

Širší možnosti, zejména pro animace, poskytuje procedura `display` z knihovny `plots`. Syntaxe je obdobná jako u `plot` s tím rozdílem, že jí předávaný parametr, buď ve formě seznamu nebo množiny, obsahuje struktury reprezentující graf.

Volbou `insequence=true` lze proceduru `display` modifikovat pro vytváření animací. Jednotlivé grafové struktury ze zadaného seznamu se pak zobrazují samostatně jako jednotlivé snímky animace.

Kombinací uvedených procedur `plot` a `display` lze vytvořit velmi rozsáhlé a složité animace.

Použití příkazu `plot` si ukážeme na následujícím příkladu.

Příklad 2.2. Pro funkci

$$\sin 2x$$

určete Maclaurinovy polynomy stupně $3, 5, 9$ a jejich grafy zobrazte do jediného obrázku.

Řešení. K výpočtu Maclaurinových polynomů použijeme funkci `TaylorPolynomial`.

```
> MP[3]:=TaylorPolynomial(sin(2*x), 0, 3);
```

$$MP_3 := 2x - \frac{4}{3}x^3$$

```
> MP[5]:=TaylorPolynomial(sin(2*x), 0, 5);
```

$$MP_5 := 2x - \frac{4}{3}x^3 + \frac{4}{15}x^5$$

```
> MP[9]:=TaylorPolynomial(sin(2*x), 0, 9);
```

$$MP_9 := 2x - \frac{4}{3}x^3 + \frac{4}{15}x^5 - \frac{8}{315}x^7 + \frac{4}{2835}x^9$$

V následujícím kroku si do proměnných *plotMP3*, *plotMP5* a *plotMP9* uložíme struktury definující grafy spočtených Maclaurinových polynomů. Pro každý polynom si zvolíme jinou barvu.

Grafové struktury jsou nepřehledné a pro nás nezajímavé, proto příkazy ukončíme dvojtečkou.

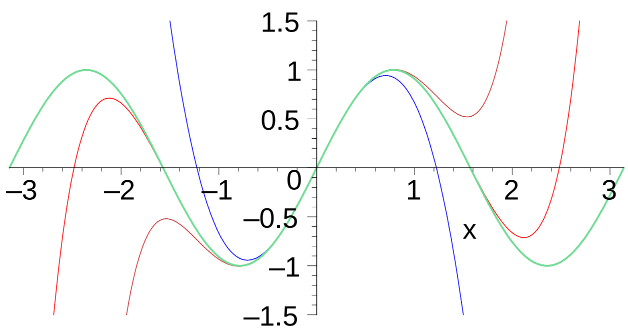
```
> plotMP3:=plot(MP[3], x=-Pi..Pi, -1.5..1.5, color=blue):
> plotMP5:=plot(MP[5], x=-Pi..Pi, -1.5..1.5, color=orange):
> plotMP9:=plot(MP[9], x=-Pi..Pi, -1.5..1.5, color=red):
```

Strukturu grafu funkce $\sin 2x$ si uložíme do proměnné *plotSin*.

```
> plotSin:=plot(sin(2*x), x=-Pi..Pi, -1.5..1.5, color=aquamarine):
```

Závěrem zobrazíme vše do jednoho grafu pomocí příkazu `display`.

```
> with(plots):
> display([plotSin, plotMP3, plotMP5, plotMP9], scaling=constrained);
```



Obr. 2.1: Funkce $\sin 2x$ a její Maclaurinovy polynomy stupně n pro $n = 3, 5, 9$

Cílem příkladu 2.2 bylo vykreslit pouze statický obrázek Maclaurinových polynomů. My však postoupíme dále a na následujícím příkladu si ukážeme, jak vytvořit animaci Taylorových či Maclaurinových polynomů.

Příklad 2.3. Vypočtěte Maclaurinovy polynomy stupně n=1, . . . ,10 funkce

$$\cos^2 x\,.$$

Z těchto polynomů následně vytvořte animaci.

Řešení. Maclaurinovy polynomy spočteme funkcí `TaylorPolynomial`. Praktické použití této funkce ilustrovalo řešení předchozího příkladu 2.2. Jednotlivé příkazy jsou však velmi podobné a jejich opisování by bylo zdlouhavé, proto využijeme cyklu `for`.

```
> for n from 1 to 10 do
>     MP[n]:=TaylorPolynomial(cos(x)^2, 0, n);
> od;
```

$$\begin{gathered}
MP_1 := 1\\
MP_2 := 1 - x^2\\
MP_3 := 1 - x^2\\
MP_4 := 1 - x^2 + \frac{1}{3}x^4\\
MP_5 := 1 - x^2 + \frac{1}{3}x^4\\
MP_6 := 1 - x^2 + \frac{1}{3}x^4 - \frac{2}{45}x^6\\
MP_7 := 1 - x^2 + \frac{1}{3}x^4 - \frac{2}{45}x^6\\
MP_8 := 1 - x^2 + \frac{1}{3}x^4 - \frac{2}{45}x^6 + \frac{1}{315}x^8\\
MP_9 := 1 - x^2 + \frac{1}{3}x^4 - \frac{2}{45}x^6 + \frac{1}{315}x^8\\
MP_{10} = 1 - x^2 + \frac{1}{3}x^4 - \frac{2}{45}x^6 + \frac{1}{315}x^8 - \frac{2}{14175}x^{10}
\end{gathered} \tag{2.3}$$

Nyní z grafů těchto polynomů vytvoříme animaci. Ukážeme si dvě varianty.

a) První varianta není dokonalá co se týče kvality výsledné animace, ale je přehledná. Tudíž se na ní dá dobře prezentovat použitý postup při programovaní animací. Základní princip opět spočívá v sestavení posloupnosti grafů, kterou zobrazíme procedurou `display`.

Grafy uvedených polynomů vygenerujeme v sekvenci procedurou `seq`. Sestavíme tak posloupnost o 10-ti členech, kde každý představuje graf Maclaurinova polynomu.

Rozsah zobrazení omezíme intervalem $[-\pi, \pi]$ pro osu x a intervalem $[-\frac{1}{2}, \frac{3}{2}]$ pro osu y. Posloupnost reprezentující animaci uložíme do proměnné *plotsMP*, kde každý člen představuje jeden snímek animace.

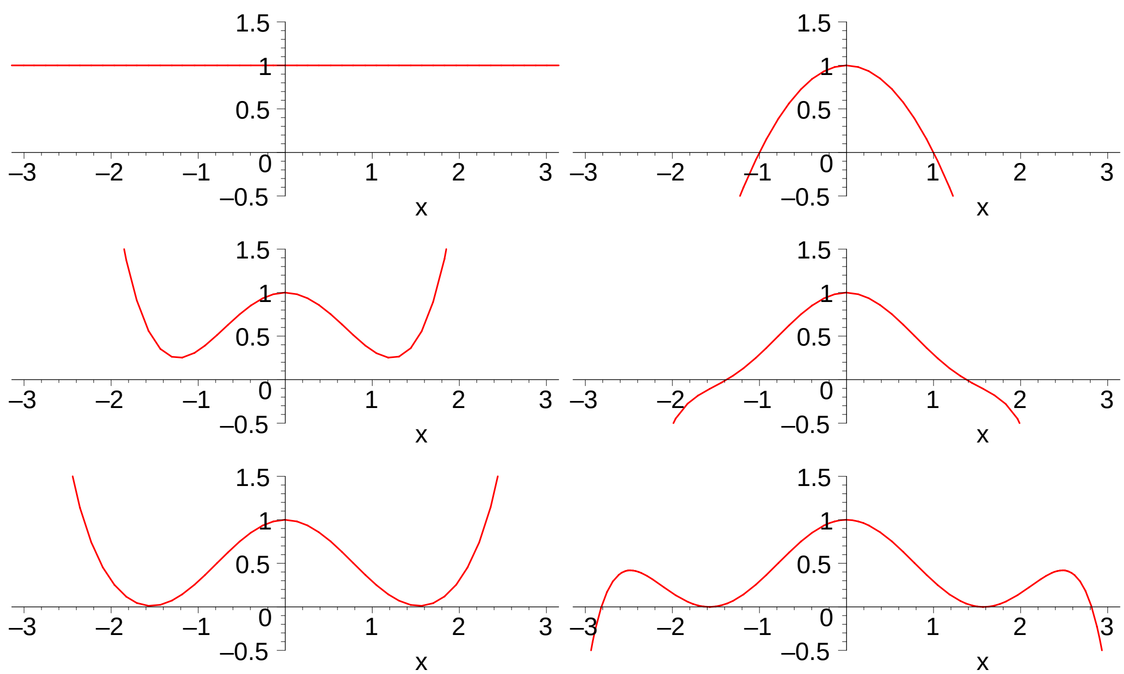
Obsah proměnné *plotsMP* je velmi dlouhý a pro nás nikterak zajímavý, proto příkaz ukončíme dvojtečkou.

```
> plotsMP:=seq(plot(MP[i], x=-Pi..Pi, -0.5..1.5), i=1..10):
```

Příkazem `display` s volbou `insequence=true` animaci zobrazíme, avšak její průběh jsme schopni pozorovat pouze na počítači. Proto jsme nuceni celou animaci rozdělit na jednotlivé snímky. Grafy Maclaurinových polynomů (2.3) tvoří 10 snímků animace (viz náhledy na obrázku 2.2).

Rozvoj funkce $\cos^2 x$ do mocninné řady má pouze sudé členy, liché členy jsou rovny 0. Liché snímky animace se opakují a tudíž je v náhledu uvádět nebudeme.

```
> display(plotsMP, scaling=constrained, insequence=true);
```



Obr. 2.2: Náhledy animace Maclaurinových polynomů funkce $\cos^2 x$ pro $n = 1, 2, 4, 6, 8, 10$

b) Na rozdíl od předešlého řešení bude tato varianta obsahovat tři nové prvky. Každý graf (snímek) animace obarvíme a zobrazíme jemu odpovídající stupeň Maclaurinova polynomu. Mimo to vykreslíme na pozadí každého snímku i graf aproximované funkce $\cos^2 x$.

Po implementační stránce bude sestavování posloupnosti podobné jako v předchozí variantě.

```
> plotsMP:=seq(display([
            plot([MP[i], cos(x)^2], x=-Pi..Pi, -0.5..1.5,
              color=[COLOR(RGB, 0+1/9*(i-1), 0, 1-1/9*(i-1)), aquamarine]),
            textplot([-Pi, 1.5, n = i],
              color=COLOR(RGB, 0+1/9*(i-1), 0, 1-1/9*(i-1)))
           ]), i=1..10):
```

Příkaz působí na první pohled dost nečitelně, proto ho dostatečně okomentujeme. První parametr procedury `plot` z varianty a) nyní nahradíme seznamem dvou položek, kde první je Maclaurinův polynom M_i a druhá aproximovaná funkce $\cos^2 x$. Jejich grafy budou tak vykresleny ve stejném snímku.

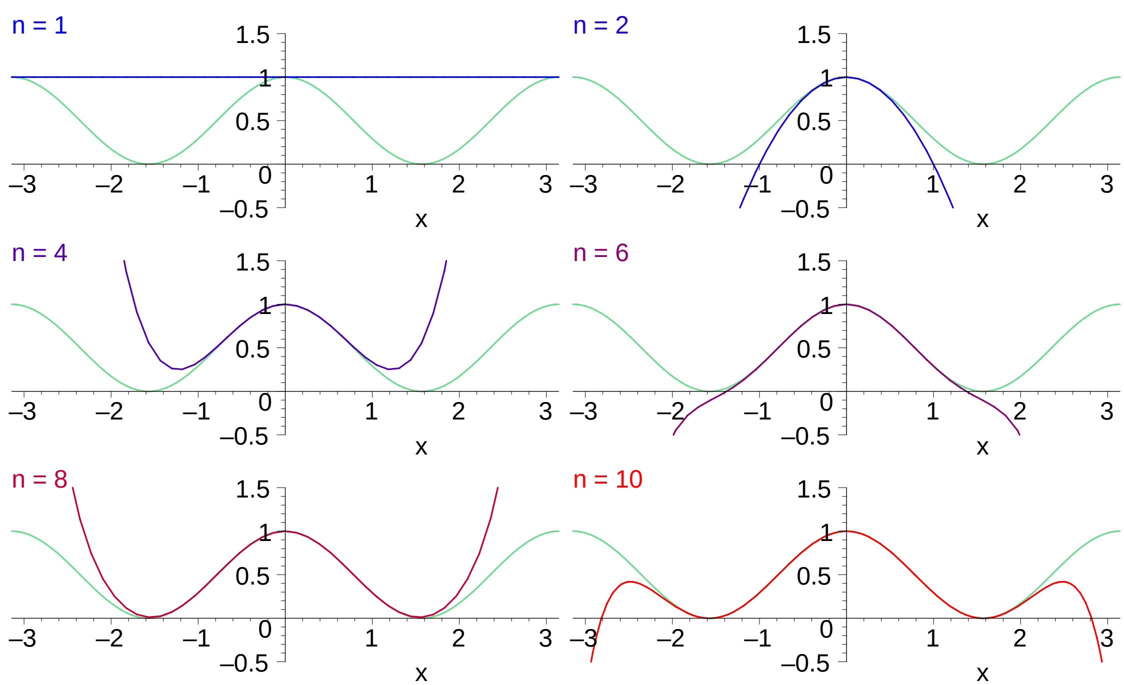
Volbou `color` nastavíme barvu čáry grafu. Po krátkém zamyšlení není jistě těžké přijít na to, jak procedura `COLOR` barvu v závislosti na indexu i generuje.

Popis grafu realizujeme procedurou `textplot`. Povinný parametr je tříprvkový seznam, resp. množina, kde první dvě číselné hodnoty určují souřadnici místa kam se má třetí textová hodnota vypsat.

Procedurou `display` pak zobrazíme tento text, jemu odpovídající graf polynomu M_i a graf funkce $\cos^2 x$ do stejného obrázku. Pomocí procedury `seq` již vytvoříme posloupnost snímku animace.

Nezbývá už nic jiného než celou animaci zobrazit. Počítačový výstup opět nahradíme náhledem na jednotlivé snímky animace.

```
> display(plotsMP, scaling=constrained, insequence=true);
```



Obr. 2.3: Náhledy animace Maclaurinových polynomů funkce $\cos^2 x$ pro $n = 1,\ 2,\ 4,\ 6,\ 8,\ 10$

2.5 Procedura `TaylorPolynomialAnimation`

Procedura `TaylorPolynomialAnimation` ukazuje široké možnosti, které matematice výpočetní technika poskytuje.

Kromě výpočtu Taylorova polynomu n-tého stupně ze zadané funkce a jeho grafického znázornění umožňuje vytvářet z těchto polynomů animace. Právě na nich se názorně předvede aproximace funkce mocninnou řadou. Bez dlouhého přemýšlení a počítání dokážeme z grafů a animací vyčíst přesnost a chyby aproximace.

Grafické výstupy procedury obsahují mnohdy několik grafů v jednom obrázku, tudíž grafy Taylorových polynomů jsou od sebe barevně odlišeny přechodem od modré po červenou barvu. Modrá barva odpovídá polynomu s nejmenším stupněm a červená naopak polynomu s největším stupněm. Každý grafický výstup obsahuje i graf rozvíjené funkce, ten je vždy znázorněn zelenou barvou.

`TaylorPolynomialAnimation(function, center, n)`

Povinné parametry:

`function` – výraz reprezentující funkci proměnné x. Omezíme se pouze na funkce, které splňují předpoklady definice 2.1.

`center` – proměnná udávající střed Taylorova polynomu. Může nabývat hodnot dvou typů. Buď je ve tvaru rovnice `x=c`, kde `c` představuje střed nebo obsahuje celočíselnou hodnotu, kterou zapisujeme pouhým `c`. Obě varianty mají stejný sémantický význam.

`n` – kladná celočíselná hodnota nebo kladný celočíselný interval jejichž hodnota určuje stupeň Taylorova polynomu.

Volitelné parametry:

`output` = {`polynomial, plot, animation, animations`}[1] – slouží jako volba výstupu. Volba `polynomial` vrací Taylorovy polynomy, jejichž stupně jsou určeny parametrem `n` a jako jediná představuje textový výstup. Další možnosti již zahrnují mnohé varianty od zobrazení (viz `plot`) až po animace. Volby `animation` a `animations` se liší způsobem zobrazování animace. `Animation` zobrazuje spolu s původní funkcí vždy pouze jeden z Taylorových polynomů a `animations` Taylorovy polynomy do animace postupně přidává. Implicitní nastavení je `output=polynomial`.

`view` = [$\mathtt{x_{min}..x_{max}}$, $\mathtt{y_{min}..y_{max}}$] – dvojice celočíselných intervalů určuje rozsah zobrazovaných hodnot v grafu na osách x a y. Ve většině zkoumaných případů nás zajímá chování v okolí středu Taylorova polynomu, tudíž implicitně je zvoleno nastavení `view=[center−2..center+2,−2..2]`.

[1] Parametr může nabývat pouze jediné uvedené hodnoty.

Nyní si na řešených příkladech předvedeme jednotlivé výstupní formy této procedury.

Příklad 2.4. Vypočtěte Maclaurinovy polynomy stupně n pro $n = 3, 4, 5, 6, 7$ funkcí

$$\text{a)} \quad (1+x)\,\mathrm{e}^{-x+1} \qquad \text{b)} \quad -\ln\cos x\,.$$

Užijte proceduru `TaylorPolynomialAnimation`.

Řešení. a) Procedura má textový výstup implicitně nastaven, proto nemusíme volbu `output=polynomial` uvádět.

```
> TaylorPolynomialAnimation((1+x)*exp(-x+1), x=0, 3..7);
```

Taylorovy polynomy stupně n *se středem v bodě* 0 *funkce* $(1+x)\,\mathrm{e}^{-x+1}$*:*

$$n=3 \quad : \quad \mathrm{e} - \frac{1}{2}\mathrm{e}\,x^2 + \frac{1}{3}\mathrm{e}\,x^3$$

$$n=4 \quad : \quad \mathrm{e} - \frac{1}{2}\mathrm{e}\,x^2 + \frac{1}{3}\mathrm{e}\,x^3 - \frac{1}{8}\mathrm{e}\,x^4$$

$$n=5 \quad : \quad \mathrm{e} - \frac{1}{2}\mathrm{e}\,x^2 + \frac{1}{3}\mathrm{e}\,x^3 - \frac{1}{8}\mathrm{e}\,x^4 + \frac{1}{30}\mathrm{e}\,x^5$$

$$n=6 \quad : \quad \mathrm{e} - \frac{1}{2}\mathrm{e}\,x^2 + \frac{1}{3}\mathrm{e}\,x^3 - \frac{1}{8}\mathrm{e}\,x^4 + \frac{1}{30}\mathrm{e}\,x^5 - \frac{1}{144}\mathrm{e}\,x^6$$

$$n=7 \quad : \quad \mathrm{e} - \frac{1}{2}\mathrm{e}\,x^2 + \frac{1}{3}\mathrm{e}\,x^3 - \frac{1}{8}\mathrm{e}\,x^4 + \frac{1}{30}\mathrm{e}\,x^5 - \frac{1}{144}\mathrm{e}\,x^6 + \frac{1}{840}\mathrm{e}\,x^7$$

b) Postupujeme analogicky.

```
> TaylorPolynomialAnimation(-ln(cos(x)), x=0, 3..7);
```

Taylorovy polynomy stupně n *se středem v bodě* 0 *funkce* $-\ln\cos x$*:*

$$n=3 \quad : \quad \frac{1}{2}x^2$$

$$n=4 \quad : \quad \frac{1}{2}x^2 + \frac{1}{12}x^3$$

$$n=5 \quad : \quad \frac{1}{2}x^2 + \frac{1}{12}x^3$$

$$n=6 \quad : \quad \frac{1}{2}x^2 + \frac{1}{12}x^3 + \frac{1}{45}x^6$$

$$n=7 \quad : \quad \frac{1}{2}x^2 + \frac{1}{12}x^3 + \frac{1}{45}x^6$$

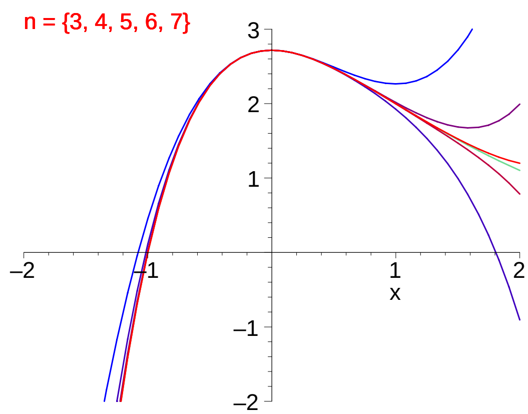
V následujícím příkladě si představíme grafický výstup `plot`.

Příklad 2.5. Procedurou `TaylorPolynomialAnimation` nakreslete grafy Maclaurinových polynomů stupně n pro $n = 3, 4, 5, 6, 7$ funkcí

$$\text{a)} \quad (1+x)\,\mathrm{e}^{-x+1} \qquad \text{b)} \quad -\ln\cos x\,.$$

Řešení. a) Volbou `output=plot` je textový výstup nahrazen obrázkem, který obsahuje grafy vybraných Maclaurinových polynomů.

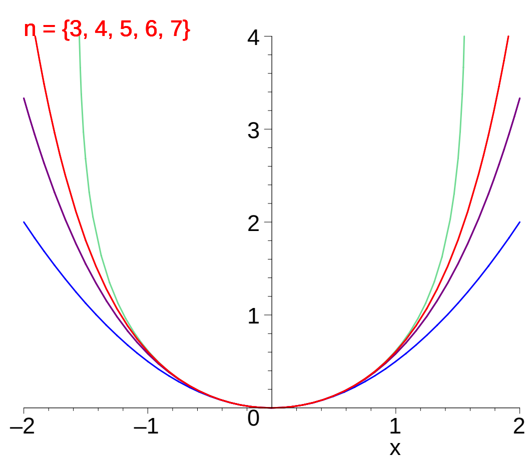
```
> TaylorPolynomialAnimation((1+x)*exp(-x+1), x=0, 3..7, output=plot,
  view=[-2..2,-2..3]);
```



Obr. 2.4: Funkce $(1+x)\,\mathrm{e}^{-x+1}$ a její Maclaurinovy polynomy stupně n pro $n = 3, 4, 5, 6, 7$

b) Postupujeme analogicky. Rozsah zobrazovaných hodnot upravíme volbou `view`.

```
> TaylorPolynomialAnimation(-ln(cos(x)), x=0, 3..7, output=plot,
  view=[-2..2,0..4]);
```



Obr. 2.5: Funkce $-\ln\cos x$ a její Maclaurinovy polynomy stupně n pro $n = 3, 4, 5, 6, 7$

Všimněme si, že grafy Maclaurinových polynomů jsou na obrázcích 2.4 a 2.5 barevně odlišeny přechodem od modré po červenou barvu, kde modrá barva odpovídá polynomu stupně $n = 3$ a červená naopak polynomu stupně $n = 7$. Zelenou barvou je vykreslen graf původní funkce.

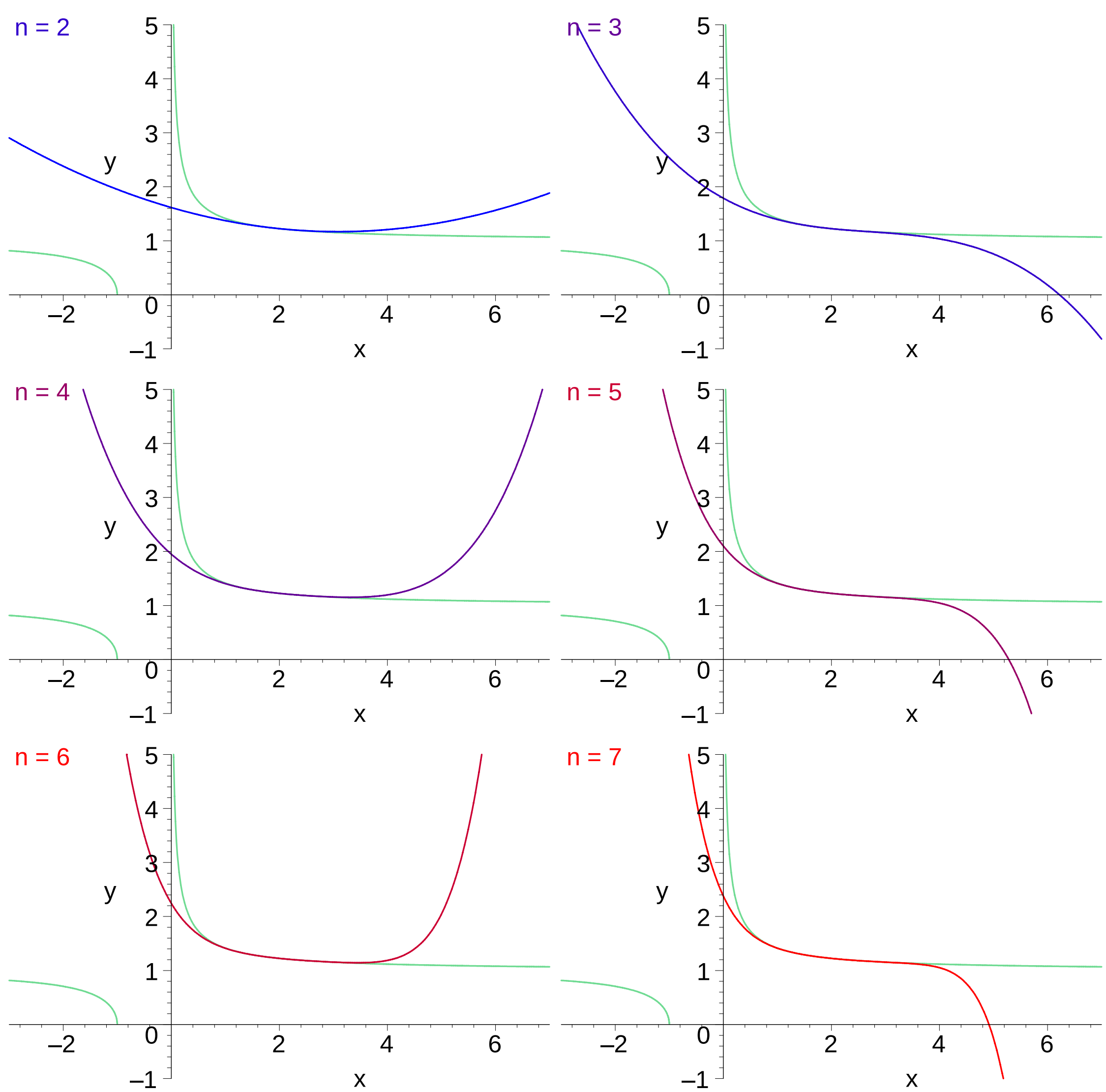
Zbývající grafické výstupy procedury `TaylorPolynomialAnimation` pro vytváření animací ilustrují následující dva příklady.

Příklad 2.6. Vytvořte animaci grafů Taylorových polynomů stupně n pro $n = 2, \ldots, 7$ se středem v bodě 2 funkce

$$\sqrt{1+\frac{1}{x}}\,.$$

Řešení. Animaci vytvoříme procedurou `TaylorPolynomialAnimation`. K vykreslení animace je nutné změnit výstup procedury volbou `output=animation`. Animaci lze zobrazit pouze na počítači, tudíž celou animaci rozložíme do šesti snímků (viz obrázek 2.6).

```
> TaylorPolynomialAnimation(sqrt(1+1/x), x=2, 2..7, output=animation,
  view=[-3..7,-1..5]);
```



Obr. 2.6: Náhled animace Taylorových polynomů stupně n pro $n = 2, \ldots, 7$ se středem v bodě 2 funkce $\sqrt{1+\frac{1}{x}}$

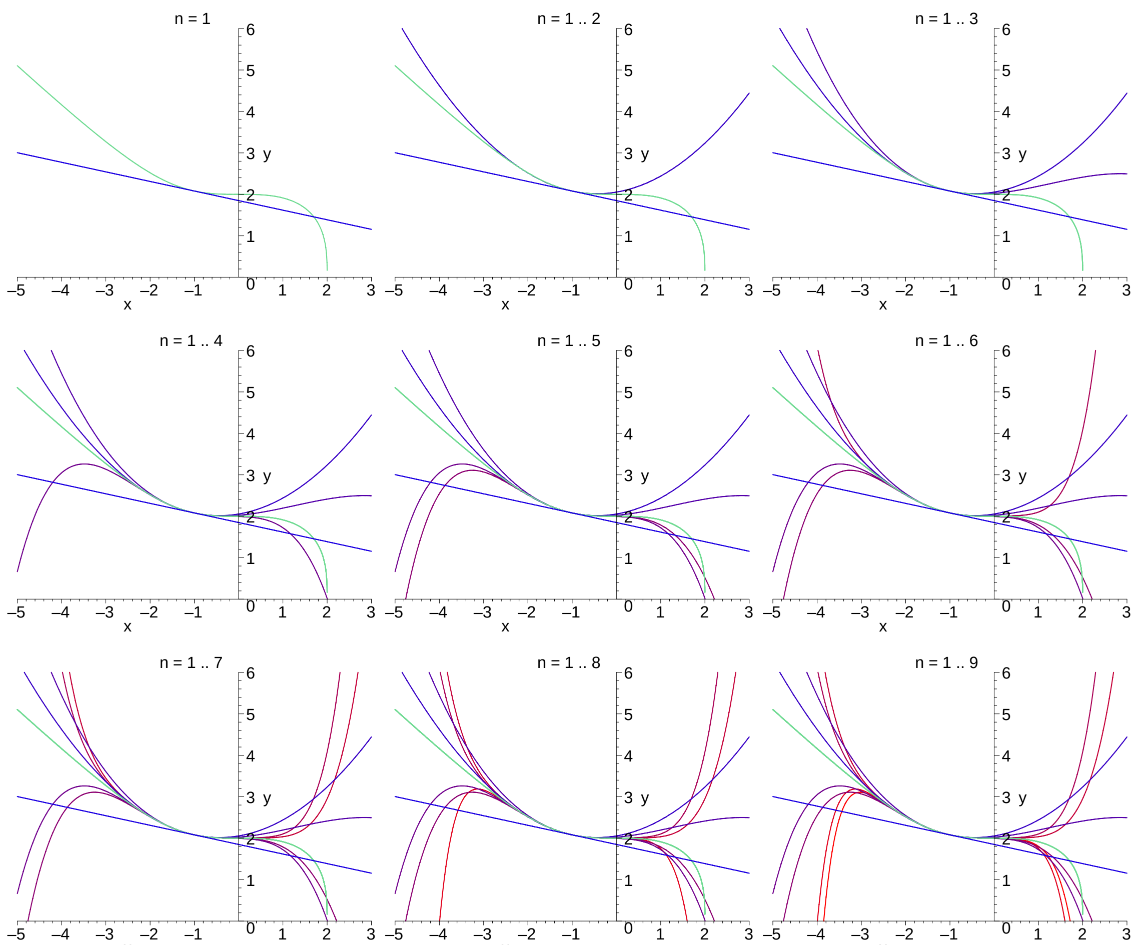
Příklad 2.7. Procedurou `TaylorPolynomialAnimation` vytvořte doplňující animaci Taylorových polynomů stupně n pro $n = 1, \ldots, 9$ se středem v bodě -1 funkce

$$\sqrt[3]{8 - x^3}\,.$$

Doplňující animací rozumíme, že každý snímek animace obsahuje snímek předchozí a navíc je doplněn o nový graf. Grafy Taylorových polynomů se do obrázku animace postupně přidávají.

Řešení. Pro vytvoření doplňující animace zvolíme nastavení `output=animations`. Celou animaci rozložíme do devíti snímků (viz obrázek 2.7).

```
> TaylorPolynomialAnimation((8-x^3)^(1/3), x=-1, 1..9, output=animations,
  view=[-5..3,0..6]);
```



Obr. 2.7: Náhled animace Taylorových polynomů stupně n pro $n = 1, \ldots, 9$ se středem v bodě -1 funkce $\sqrt[3]{8 - x^3}$

Kapitola 3

Internetové rozšíření a prezentace

V předchozích dvou kapitolách jsme ukázali základní vlastnosti mocninných řad, využití programu Maple jako výpočetního systému při řešení problémů týkajících se uvedené problematiky a prezentovali jsme nové procedury pro mocninné řady.

Na otázku, jak představit nově vytvořené doplňující Maple procedury veřejnosti, či jakkoliv ji efektivně publikovat, se ihned nabízí odpověď ve formě vhodné prezentace na internetu. Ovšem požadujeme-li dostatečnou kvalitu, je obecně publikace matematického textu na internetu v současné době obtížná. Pokud se jedná o interaktivní výpočty závislé na uživatelském ovládání je situace ještě komplikovanější. Jazyky používané v současnosti pro budování webových stránek a aplikací neobsahují vybavení jaké bychom pro matematiku potřebovali. Mnohé programovací jazyky obsahují sice matematické knihovny, nicméně jejich užití je dostatečné pouze pro základní matematiku.

V následujícím textu si ukážeme jednu z možností, jak interaktivně prezentovat práci v Maple i širší veřejnosti. Řešení problému spočívá ve vytvoření uživatelského prostředí, které je řešeno formou internetové stránky a v návrhu komunikace mezi uživatelem a serverem, na kterém běží Maple. Díky velké rozšířenosti internetu je tato aplikace dobře uživatelsky dostupná a je nezávislá na instalaci programu Maple na počítači uživatele.

Vlastní návrh a implementace aplikace je mírně komplikovanější. Svým technickým zaměřením ani nezapadá do tématického směru práce, tudíž zde nebude podrobně vysvětlen. Následující text nelze proto brát jako návod pro zbudování takovéto aplikace. Ukážeme naopak možnosti, které nám tento způsob prezentace nabízí. Zmíníme výhody spočívající v nízké náročnosti na technické a programové vybavení a v neposlední řadě prezentujeme její funkčnost a využití.

Závěr bude věnován stručnému nastínění použitých technologií a získaných zkušeností při implementaci a porovnání s jinými dostupnými prostředky pro prezentaci matematiky na internetu.

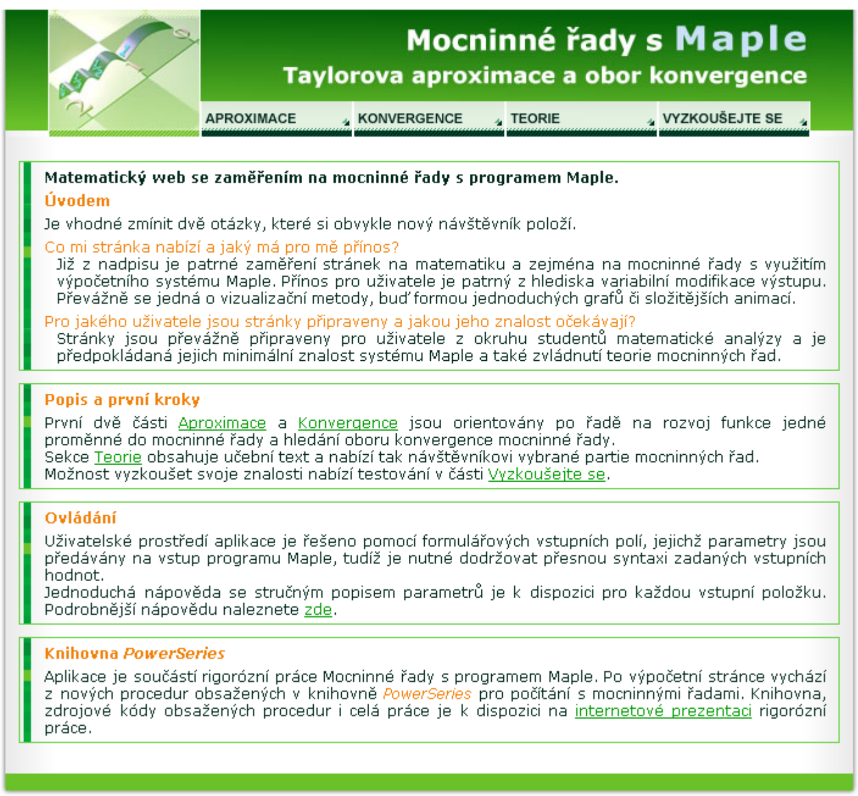
3.1 Internetová aplikace Mocninné řady s Maple

V následujících několika odstavcích si předvedeme rozsah a zaměření internetové aplikace, dostupné na stránce `http://www.math.muni.cz/~kriz/pseries`. Na první pohled působí sice jako běžná internetová stránka, ale díky svým výpočetním a vizualizačním schopnostem nabízí pomoc a didaktickou podporu nejen pro studenty matematické analýzy.

3.1.1 Hlavní stránka

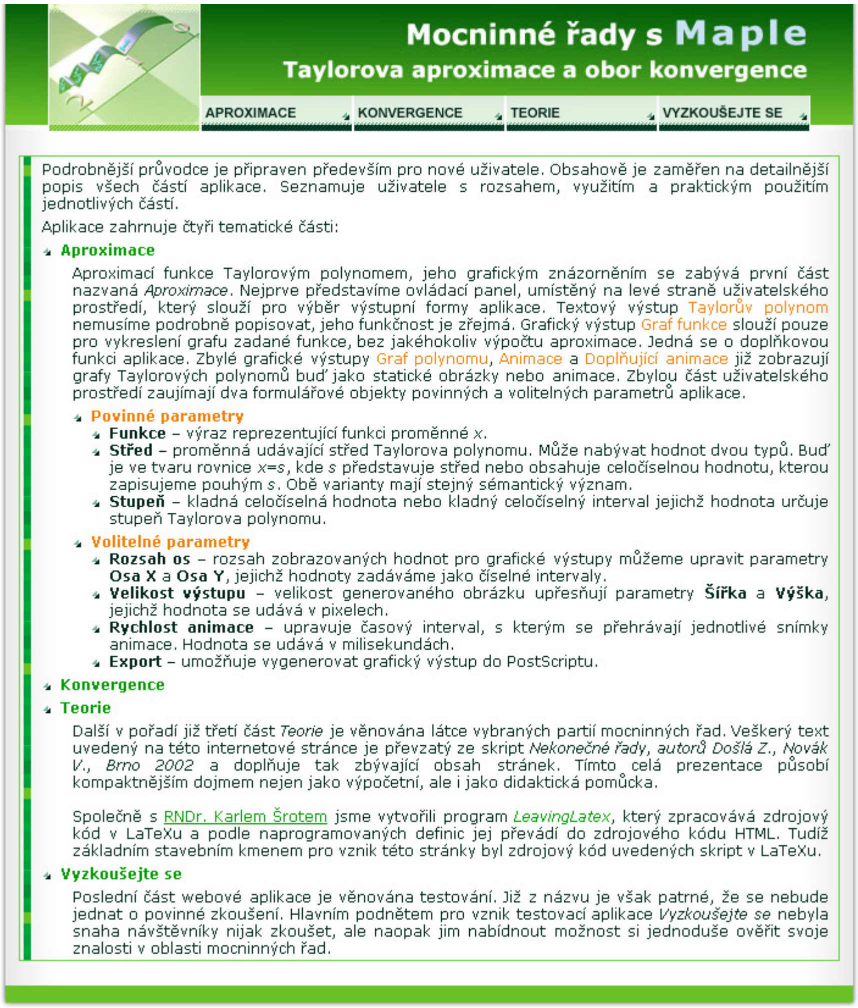
Nyní se seznámíme s obsahem, budeme se věnovat popisu stránek a přiblížíme si jejich funkce na některých příkladech.

Již z nadpisu je patrné zaměření na mocninné řady s využitím programu Maple. Hlavní stránka (náhled na obrázku 3.1) uvádí návštěvníka do probírané problematiky a je vstupní branou pro nové uživatele. Úvodní slova seznamují uživatele s tím, co mu stránka může nabídnout a stručně popisují jednotlivé části aplikace a její ovládání.



Obr. 3.1: Úvodní stránka internetové aplikace

Celá aplikace je strukturována tematicky do čtyř částí, *Aproximace*, *Konvergence*, *Teorie* a *Vyzkoušejte se*. Každé části bude věnován samostatný odstavec této sekce. Pro uživatele, kteří nejsou s programem Maple dostatečně seznámení je připraven *Podrobnější průvodce* (náhled na obrázku 3.2). Obsahově je zaměřen na detailnější popis všech částí aplikace. Seznamuje uživatele s rozsahem, využitím a praktickým použitím jednotlivých částí. Nechybí ani stručná nápověda pro každý vstupní parametr, ovládaná přes funkci *Popis parametrů* v ovládacím panelu. Tato funkce je orientována na uživatele, kteří již mají zkušenosti se syntaxí Maple. Upřesňuje význam parametrů a popisuje formát vstupních hodnot.



Obr. 3.2: Podrobnější průvodce aplikace

3.1.2 Aproximace

Rozvojem funkce do mocninné řady se zabývá první část nazvaná *Aproximace*. Aplikace po výpočetní stránce vychází z automatizované procedury `TaylorPolynomialAnimation` uvedené v druhé kapitole.

Uživatelské rozhraní se skládá s ovládacího panelu umístěného v levé části okna a dvou formulářových objektů obsahujících vstupní pole povinných a nepovinných parametrů.

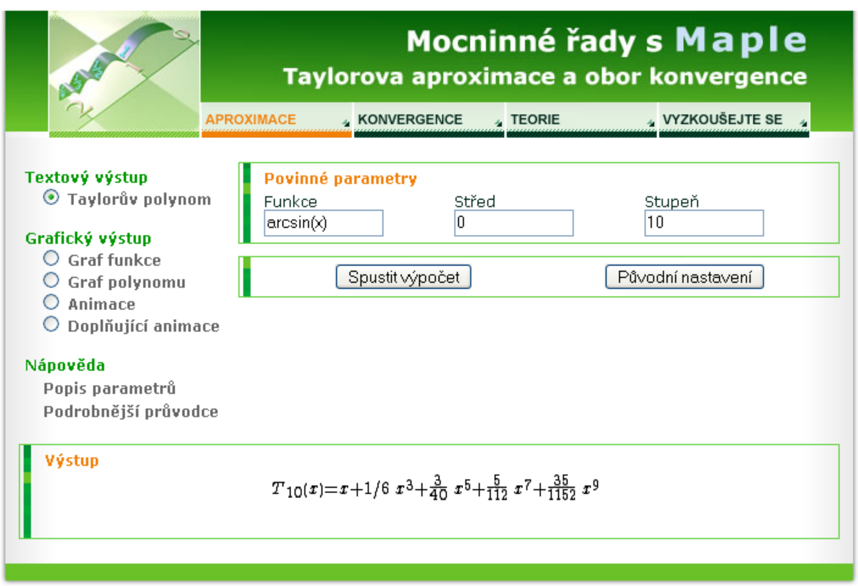
Nejprve si představíme ovládací panel, který slouží pro výběr výstupní formy aplikace. Textový výstup *Taylorův polynom* nemusíme podrobně popisovat, jeho funkčnost je zřejmá. Grafický výstup *Graf funkce* slouží pouze pro vykreslení grafu zadané funkce, bez jakéhokoliv výpočtu aproximace. Jedná se o doplňkovou funkci aplikace. Zbylé grafické výstupy *Graf polynomu*, *Animace* a *Doplňující animace* již zobrazují grafy Taylorových polynomů buď jako statické obrázky nebo animace. Z pěti uvedených výstupů, které aplikace nabízí, lze v tištěné formě prezentovat pouze první tři. Jedná se tedy o textový výstup *Taylorův polynom* a první dva grafické výstupy *Graf funkce* a *Graf polynomu*. Z tohoto důvodu budou na vybraných příkladech prezentovány pouze uvedené výstupní formy.

Povinné či některé volitelné parametry mají totožný význam jako parametry popsané na straně 28 a jsou jejich českými ekvivalenty. Nechybí zde ani parametry pro nastavení šířky a výšky grafického výstupu, změnu rychlosti animace mezi jednotlivými snímky nebo možnost exportovat obrázek do PostScriptu.

Jak již bylo uvedeno dříve, je nutné dodržovat přesný formát vstupních hodnot parametrů i jejich zápis v syntaxi Maple. Z tohoto důvodu je každý formulářový prvek doplněný stručnou nápovědou.

Příklad 3.1. Vypočtěte prvních deset členů Maclaurinovy řady funkce $\arcsin x$.

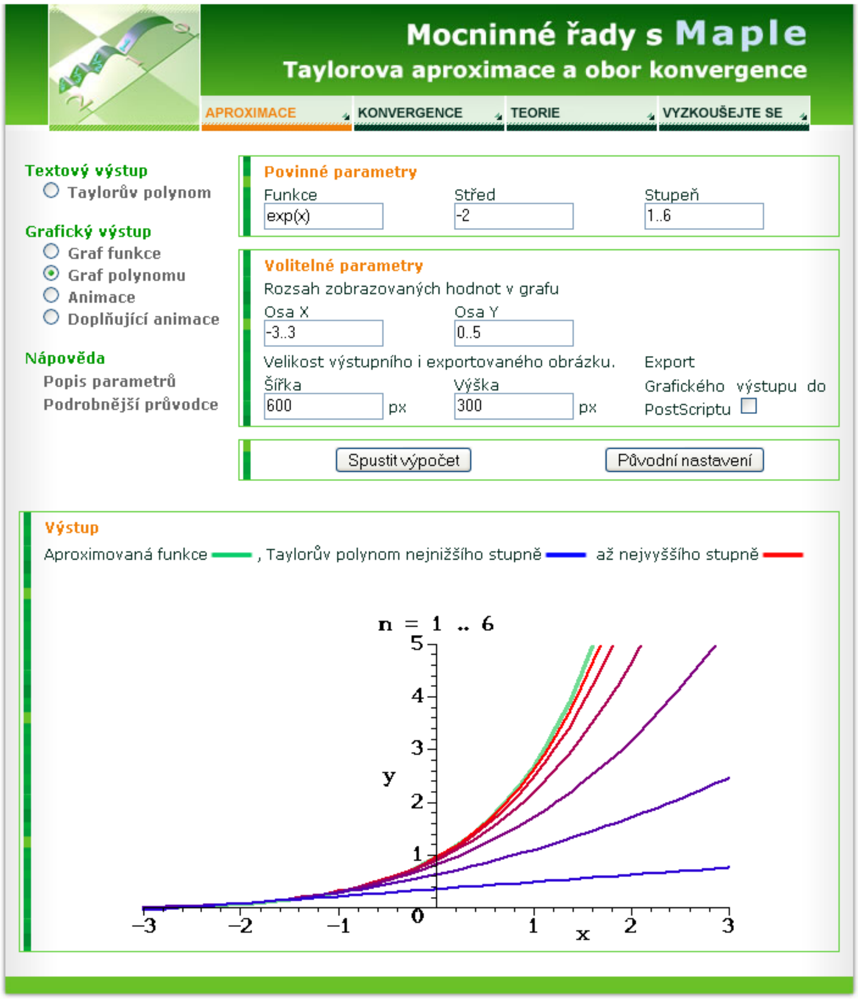
Řešení. Řešení zde budeme uvádět jako náhled aplikace ve formě obrázku, který jsme získali z webového prohlížeče.



Obr. 3.3: Maclaurinův polynom desátého stupně funkce $\arcsin x$

Příklad 3.2. Nakreslete grafy Taylorových polynomů stupně n pro $n = 1, \ldots, 6$ se středem v bodě -2 funkce e^x.

Řešení. Volbu výstupu nastavíme na grafický výstup *Graf polynomu*.



Obr. 3.4: Grafy Taylorových polynomů stupně n pro $n = 1, \ldots, 6$ funkce e^x

3.1.3 Konvergence

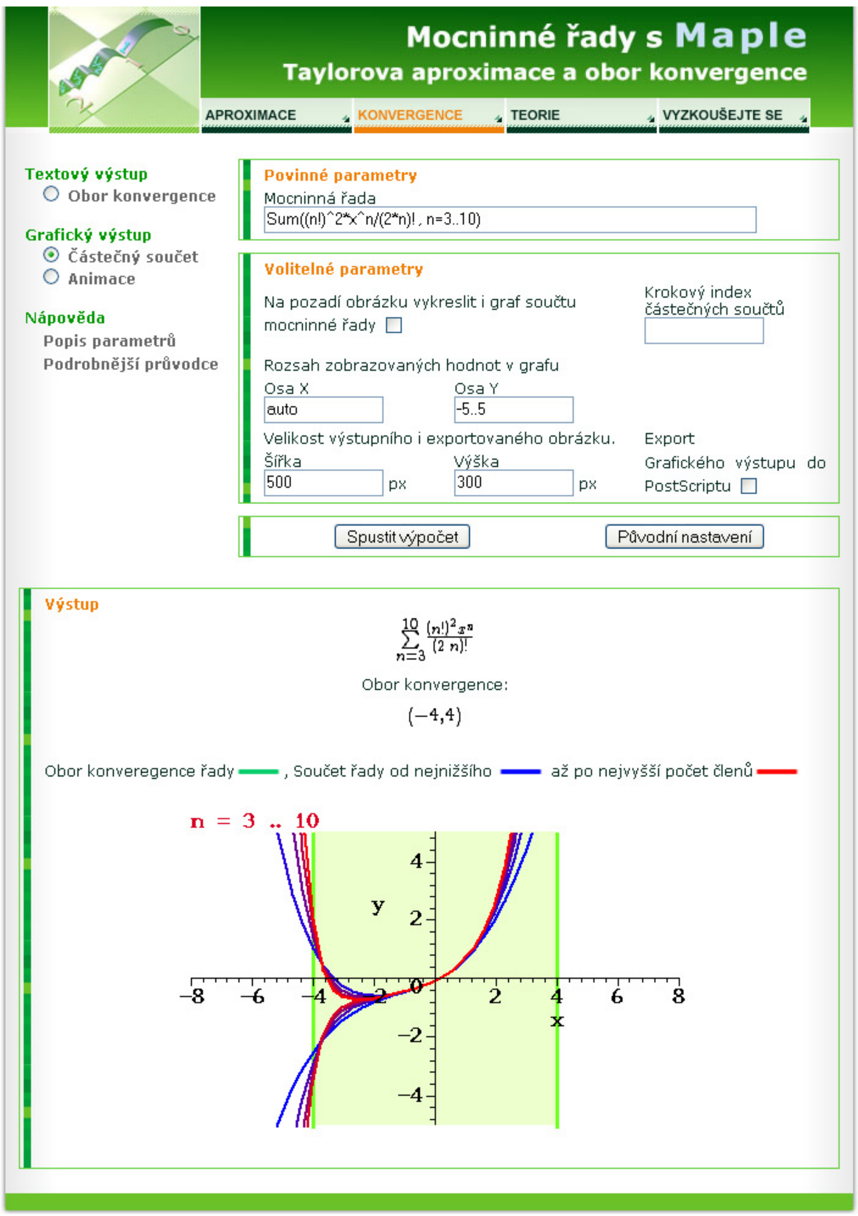
Druhá část nazvaná *Konvergence* se věnuje oboru konvergence a grafickému znázornění částečných součtu mocninných řad. Struktura, vzhled i ovládání uživatelského prostředí jsou velmi podobné s předchozí částí *Aproximace*. Po výpočetní stránce vychází aplikace z automatizované procedury `ConvergenceTestOfPowerSeries` uvedené v první kapitole.

Výstupní volba aplikace se standardně nastavuje v ovládacím panelu v levé části uživatelského rozhraní a nabízí jeden textový *Obor konvergence* a dva grafické režimy *Částečný součet* pro zobrazení grafů částečných součtů a *Animace* pro vytvoření animace částečných součtů mocninné řady.

Volba *Popis parametrů* opět ovládá stručnou nápovědu pro vstupní pole parametrů popisující formát a tvar vstupních hodnot. Popis parametrů je podrobněji popsán na straně 14.

Příklad 3.3. Určete obor konvergence mocninné řady $\sum_{n=1}^{\infty} \frac{(n!)^2 x^n}{(2n)!}$ a znázorněte její částečné součty s_n pro $n = 3, \ldots, 10$.

Řešení. Řešení zde budeme uvádět jako náhled aplikace ve formě obrázku, který jsme získali z webového prohlížeče.

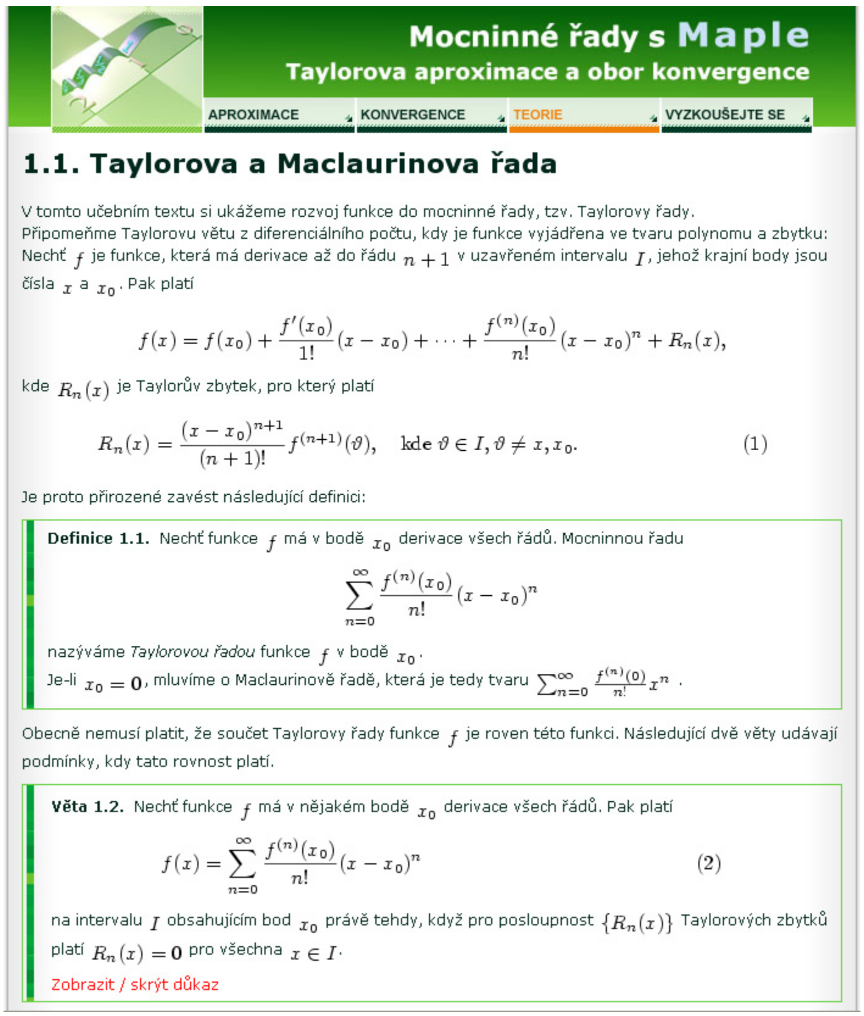


Obr. 3.5: Obor konvergence a částečné součty s_n řady $\sum \frac{(n!)^2 x^n}{(2n)!}$ pro $n = 3, \ldots, 10$

3.1.4 Teorie

Další v pořadí již třetí část *Teorie* je věnována látce vybraných partií mocninných řad. Veškerý text uvedený na této internetové stránce je převzatý z [3] a doplňuje tak zbývající obsah stránek. Tímto celá prezentace působí kompaktnějším dojmem nejen jako výpočetní, ale i jako didaktická pomůcka.

Společně s RNDr. Karlem Šrotem jsme vytvořili program *LeavingLatex*,[1] který zpracovává zdrojový kód v LaTeXu a podle naprogramovaných definic jej převádí do zdrojového kódu HTML. Tudíž základním stavebním kmenem pro vznik této stránky byl zdrojový kód skript [3] v LaTeXu.[2]



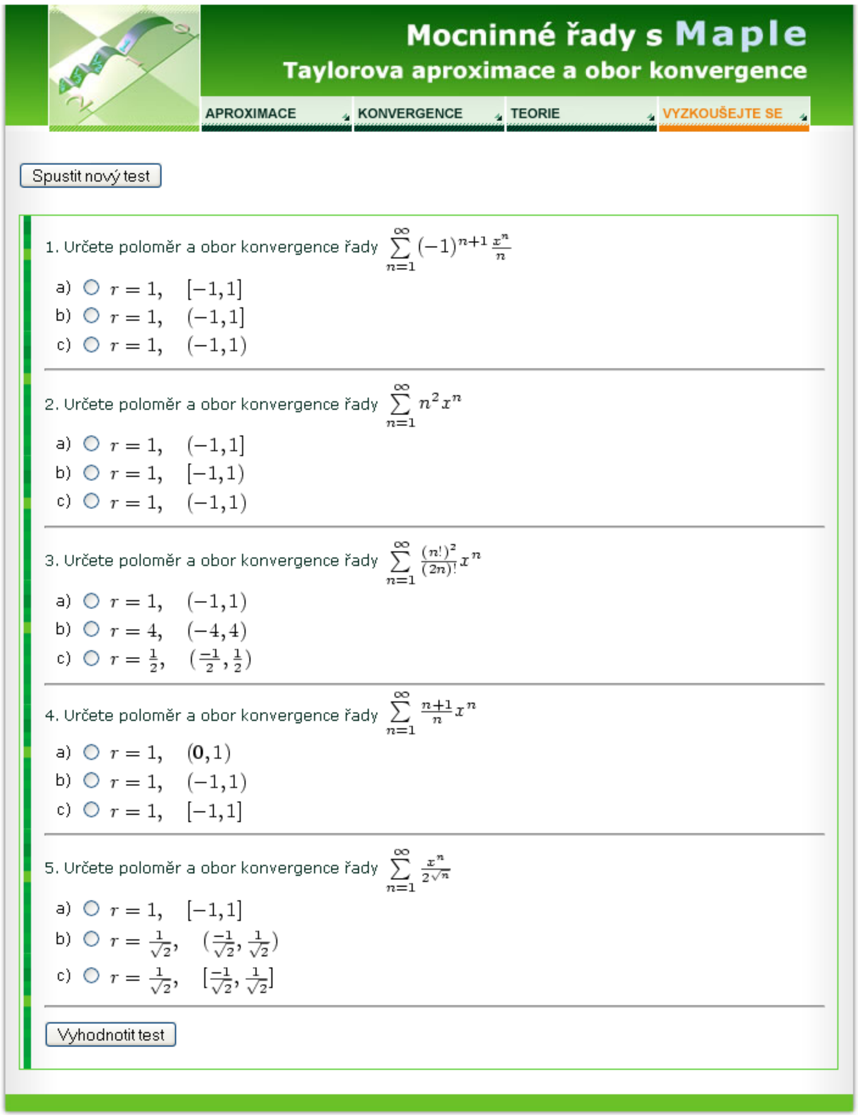
Obr. 3.6: Teorie – učební text

[1]Program prochází stále ještě vývojem a testováním, tudíž není volně dostupný. V případě zájmu kontaktujte autory. Více na stránách `http://www.math.muni.cz/~kriz`, resp. `http://www.math.muni.cz/~xsrot`.

[2]Se svolením a velkou podporou autorů bylo možné tento učební text vystavit v elektronické podobě na internetu. Tímto jim chci velmi poděkovat.

3.1.5 Vyzkoušejte se

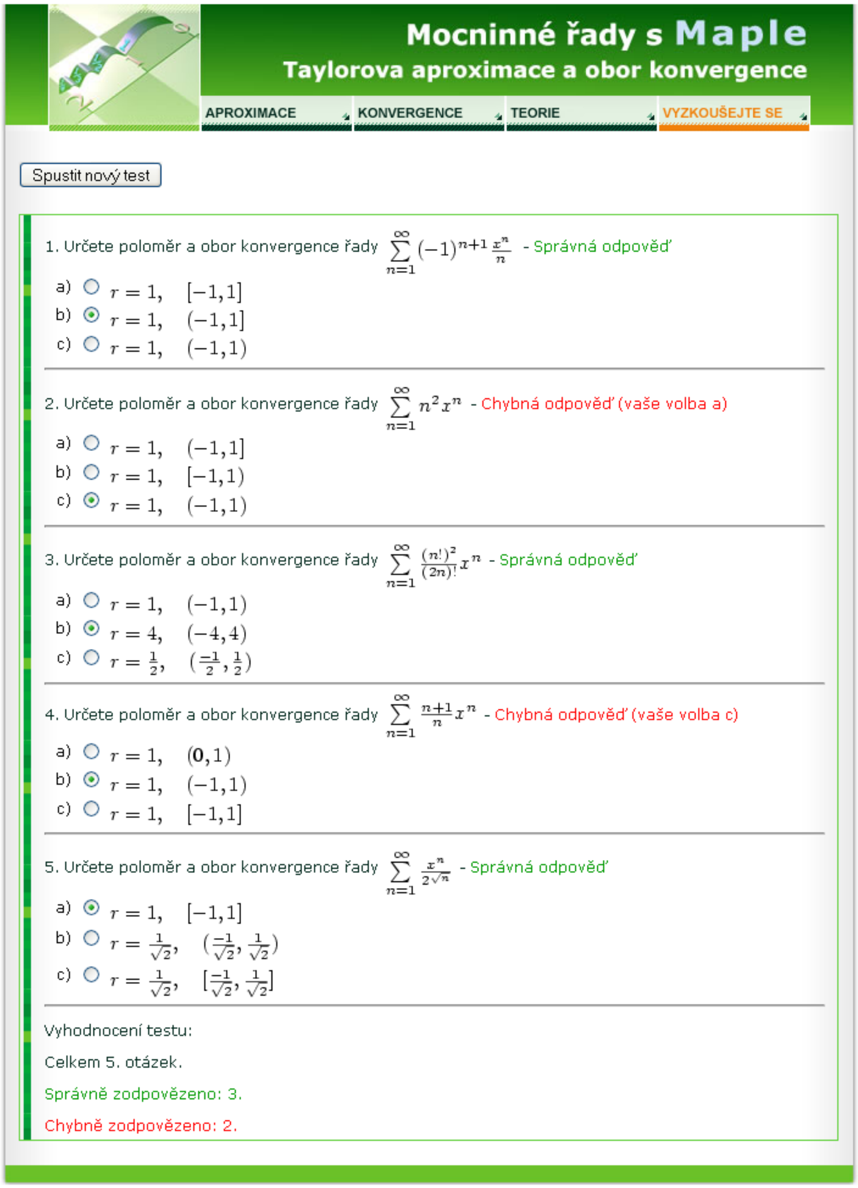
Poslední část webové aplikace je věnována testování. Již z názvu je však patrné, že se nebude jednat o povinné zkoušení. Hlavním podnětem pro vznik testovací aplikace *Vyzkoušejte se* nebyla snaha návštěvníky nijak zkoušet, ale naopak jim nabídnout možnost si jednoduše ověřit svoje znalosti v oblasti mocninných řad.



Obr. 3.7: Vygenerovaný test

Test samotný se skládá z pěti otázek, které jsou náhodně vybírány z připravené testové databáze. Každá otázka nabízí tři možné odpovědi, z nichž právě jedna je správná. Po zodpovězení všech otázek je zpřístupněna volba pro vyhodnocení testu. Automatická oprava odkryje správné a upozorní na případné chybné odpovědi.

Následující obrázek 3.8 ukazuje náhled na vyhodnocení předešlého testu.



Obr. 3.8: Opravený a vyhodnocený test

Testy jsou veřejně dostupné pro všechny uživatele. Počet přístupů k testům není nijak omezen, tudíž si návštěvníci mohou testy skládat opakovaně.

3.2 Technický popis aplikace a implementace

V několika málo odstavcích si přiblížíme technický popis, tvorbu a použité technologie využité při implementaci aplikace.

Vizuální forma stránek je realizována jazykem DHTML s podporou kaskádových stylů a jazyka JavaScript. Převážná část je budována pomocí CGI skriptů jazyka Perl, které zajišťují celkovou vnitřní obsluhu a vytvářejí komunikaci mezi uživatelským prostředím a serverem. Při návrhu rozhraní byl kladen důraz nejen na obsah, ale i na estetiku, přehlednost a srozumitelnost. Snažili jsme se nabídnout uživatelům a návštěvníkům stránek zajímavé prostředí, co by je upoutalo natolik, aby se na stránky vraceli i v budoucnu.

Uživatelské prostředí *Aproximace*, *Konvergence* a *Vyzkoušejte se* je obsluhováno skriptem, který převezme vstupní data zadané uživatelem. Během vstupní analýzy se tyto data otestují proti případnému zneužití a útoku na server. Je-li splněna prvotní kontrola, skript vygeneruje vstupní textový zápisník pro program Maple. Procedury `ConvergenceRadius`, `ConvergenceTestOfPowerSeries` a `TaylorPolynomialAnimation` jsou vhodně upraveny a jejich definice jsou uloženy do textových souborů, které se spolu se vstupním souborem předají textové verzi programu Maple. Průběh výpočtu spolu s výstupem je zaznamenáván do souborů, které jsou následně zpracovány.

Kvalita exportovaných grafických animací není dostatečná, velká rychlost znemožňuje jejich sledování. Z těchto důvodů je rychlost animací upravována programem `convert`, který je součástí programu `ImageMagic`.[3]

Pro zlepšení čitelnosti, méně přehledného výstupu v textové formě, jsou veškeré textové výstupy převáděny do zdrojového kódu v LaTeXu, z něhož jsou generovány obrázky,[4] které jsou do výsledné stránky následně vloženy.

3.3 Publikace s využitím technologie `MapleNet`

Jiná možnost prezentace spočívá ve využití komerčních produktů. Jedním z nich je systém MapleNet[5], který společně se systémem Maple T.A. tvoří výukový LMS (Learning Management System) společnosti Maplesoft, jakožto produkt vytvořený na podporu e-learningu matematiky. Oba tyto systémy vycházejí z programu Maple, přičemž systém MapleNet je navázán na již nainstalovaný Maple, zatímco systém Maple T.A. je kompletní systém obsahující výpočetní nástroj Maple jako svoji součást. Zatímco MapleNet pracuje na podobném principu, který je popsán v úvodu kapitoly, tedy využívá možností programu Maple prostřednictvím webového prohlížeče, Maple T.A. tvoří v podstatě informační systém založený na procvičování, zkoušení a testování studentů.

Publikování práce v Maple na serveru MapleNet se standardně provádí ve formě Mapletů a je závislé ze strany uživatele na prohlížečích podporujících Javu. V porovnání

---

[3] Oficiální stránky programu `http://www.imagemagick.org/`.

[4] Obrázky jsou generovány s využitím skriptu `mimetex.cgi` který je dostupný na stránkách Mgr. Michala Bulanta, Ph.D. `http://cgi.math.muni.cz/~bulik/`.

[5] Oficiální stránky produktu `http://www.maplesoft.com/products/maplenet/`.

s uvedenou webovou aplikací *Mocninné řady s programem Maple* nabízí obě varianty své pro i proti. Po programátorské a technické stránce je publikování na MapleNetu jednoduší než implementace vlastní webové aplikace. Nenabízí však takové možnosti jako je navržení vlastního uživatelského prostředí či integrace výpočtů do internetové stránky. Nevýhodou je ovšem nedostačující kvalita výstupů a variabilita pro využití při jiných výpočtech.

V rámci rozvoje e-learningu a distančního vzdělávání byl tento rok zahájen provoz MapleNetu na Masarykově univerzitě. Po úspěšném zavedení nyní probíhá počáteční testování a doladění komunikace mezi Informačním systémem MU.

Na podobném principu rozvoje e-learningu matematiky pracuje i systém WebMathematika,[6] který ale není na Masarykově univerzitě dostupný.

[6] Oficiální stránky produktu `http://www.wolfram.com/products/webmathematica/`.

Kapitola 4

Knihovna `PowerSeries`

Knihovna obsahuje procedury `ConvergenceRadius`, `ConvergenceTestOfPowerSeries` a `TaylorPolynomialAnimation`. Byla implementována pro Maple 9.5. V nižších verzích nebyla testována, tudíž nemůžeme zaručit její správnou funkčnost.

Instalace knihovny

Následující postup instalace se týká Maplu 9.5, lze jej však realizovat i pro verze 7, 8 a 9.

1. Knihovna se skládá ze dvou souborů `maple.lib` a `maple.ind`. Nejdříve je třeba vytvořit adresář a tyto soubory do tohoto adresáře zkopírovat. Předpokládejme, že knihovna se nachází v adresáři `c:/maple/lib/powerseries`.[1]

2. Jména adresářů, ve kterých se moduly hledají, jsou uloženy v proměnné `libname`. Pro použití modulu tedy stačí zadat následující příkaz.

   ```
   > libname:=libname, "c:/maple/lib/powerseries":
   ```

3. Při opakovaném spuštění programu Maple se proměnné `libname` nastaví implicitní hodnota. Abychom nemuseli uvedený příkaz zadávat při každém spuštění, vytvoříme tzv. inicializační soubor, do kterého tento příkaz napíšeme. Na platformě Windows se tento soubor jmenuje `maple.ini`[2] a Maple jej hledá v adresářích `"Maple 9.5/Lib"`,[3] pracovním adresáři, `"Maple 9.5/Users"` či v adresáři s uživatelovým profilem. V Unixových systémech se jmenuje `.mapleinit` a musíme jej umístit do svého domovského adresáře.

4. Po spuštění programu Maple již lze knihovnu načíst příkazem `with(PowerSeries);`.

[1] V Maple lze místo zpětného lomítka použít v cestě lomítko běžné. Zpětné lomítko je speciální znak a bylo by tedy nutné jej psát zdvojeně, navíc se tak snižují rozdíly mezi platformami Unix popř. Linux a Windows.

[2] Pozor, nezaměňovat se souborem `maple9.5.ini` či se souborem podobným v jiných verzích.

[3] `Maple 9.5` zde označuje adresář, ve kterém je Maple nainstalovaný.

Pokud všechny pokusy o instalaci selžou, je nutné před použitím procedur z knihovny PowerSeries načíst jejich definice ze zdrojových kódů. Pro tento případ jsou uloženy v souboru PowerSeries_sources.mws.

Literatura

[1] Berman G. N., *Sbornik zadač po kursu matematičeskogo analyza*, Nauka, Moskva 1971

[2] Děmidovič B. P., *Sbornik zadač i upražněnij po matematičeskomu analyzu*, Nauka, Moskva 1964

[3] Došlá Z., Novák V., *Nekonečné řady*, Masarykova univerzita, Brno 2002

[4] Došlá Z., Plch R., Sojka P. *Nekonečné řady s programem Maple CD-ROM*, Masarykova univerzita, Brno 2002

[5] Hřebíček J., Pešl J., Ráček J., *Úvod do Maple 7*, Masarykova univerzita, Brno 2002

[6] Hřebíček J., `www.fi.muni.cz/~hrebicek/maple`, *Úvod do Maple 9.5*, HTML prezentace, 2004

[7] Israel R., Maple Advisor Database, `http://www.math.ubc.ca/~israel/advisor/`, 1998

[8] Kubínová M., Novotná J., *Posloupnosti a řady*, Karolinum, Praha 1997

[9] Veselý J. *Matemetická analýza pro učitele*, Matfyzpress, Praha 1997

[10] Walz A. F., The math package, `http://sunsite.informatik.rwth-aachen.de/maple/frame06.htm`, 2006