MASARYKOVA UNIVERZITA
PŘÍRODOVĚDECKÁ FAKULTA
ÚSTAV MATEMATIKY A STATISTIKY

Diplomová práce

BRNO 2018 **MIKULÁŠ MÚDRY**

MASARYKOVA UNIVERZITA
PŘÍRODOVĚDECKÁ FAKULTA
ÚSTAV MATEMATIKY A STATISTIKY

Diferenciální rovnice s programem Maxima

Diplomová práce

Mikuláš Múdry

Vedoucí práce: RNDr. Roman Plch, Ph.D. **Brno 2018**

Bibliografický záznam

Autor:	Bc. Mikuláš Múdry Přírodovědecká fakulta, Masarykova univerzita Ústav matematiky a statistiky
Název práce:	Diferenciální rovnice s programem Maxima
Studijní program:	Matematika
Studijní obor:	Matematika s informatikou
Vedoucí práce:	RNDr. Roman Plch, Ph.D.
Akademický rok:	2017/2018
Počet stran:	xiv + 104
Klíčová slova:	Maxima; wxMaxima; systém počítačové algebry; diferenciální; rovnice; ODR; prvního řádu; druhého řádu; derivace; analytické; numerické; nerozřešené

Bibliographic Entry

Author:	Bc. Mikuláš Múdry Faculty of Science, Masaryk University Department of Mathematics and Statistics
Title of Thesis:	Differential equations with the Maxima program
Degree Programme:	Mathematics
Field of Study:	Mathematics with Informatics
Supervisor:	RNDr. Roman Plch, Ph.D.
Academic Year:	2017/2018
Number of Pages:	xiv + 104
Keywords:	Maxima; wxMaxima; computer algebra system; CAS; differential; equation; equations; ODE; first order; second order; derivative; exact; numerical; unsolvable

Abstrakt

Tato diplomová práce popisuje možnosti systému počítačové algebry Maxima v oblasti řešení a vizualizace diferenciálních rovnic prvního a druhého řádu. Použity jsou jak příkazy k získání přesného, tak numericky spočteného přibližného řešení. Zvláště je kladen důraz na vhodné příklady ilustrující konkrétní příkazy a jejich volitelné parametry. Přílohu práce tvoří balíček příkazů k řešení diferenciálních rovnic prvního řádu nerozřešených vzhledem k derivaci.

Abstract

This Master's thesis describes abilities of computer algebra system Maxima in field of solving and visualising first and second order differential equations. Commands for obtaining both exact and numerically computed approximate solutions are used. We put emphasis on appropriate examples that illustrate commands and their options. Package with commands for solving first order differential equations unsolvable with respect to the derivative is attached.

MASARYKOVA UNIVERZITA
Přírodovědecká fakulta

ZADÁNÍ DIPLOMOVÉ PRÁCE

Akademický rok: 2014/2015

Ústav: Ústav matematiky a statistiky

Student: Bc. Mikuláš Múdry

Program: Matematika

Obor: Matematika s informatikou

Ředitel *Ústavu matematiky a statistiky* PřF MU Vám ve smyslu Studijního a zkušebního řádu MU určuje diplomovou práci s tématem:

Téma práce: Diferenciální rovnice s programem Maxima

Téma práce anglicky: Differential equations with the Maxima program

Oficiální zadání:
V první části práce popište obecné možnosti CAS systému Maxima pro rešení a visualizace v oblasti diferenciálních rovnic. V druhé části práce se zaměřte na rovnice prvního řádu nerozřešené vzhledem k derivaci a vytvořte pomocí programu Maxima podpůrné materiály pro řešení tohoto typu rovnic.

Literatura:
KALAS, Josef a Miloš RÁB. *Obyčejné diferenciální rovnice*. 2. vyd. Brno: Masarykova univerzita, 2001. 207 s. ISBN 80-210-2589-1.

PLCH, Roman. *Příklady z matematické analýzy, Diferenciální rovnice*. 1. vydání. Brno: Masarykova univerzita, 2002. 31 s. ISBN 80-210-2806-8.

Jazyk závěrečné práce:

Vedoucí práce: RNDr. Roman Plch, Ph.D.

Datum zadání práce: 19. 9. 2014

V Brně dne: 23. 10. 2014

Souhlasím se zadáním (podpis, datum):

Múdry 4.11.2014
Bc. Mikuláš Múdry
student

RNDr. Roman Plch, Ph.D.
vedoucí práce

v.z. Paul Rosický
prof. RNDr. Jiří Rosický, DrSc.
ředitel Ústavu matematiky a statistiky

Poděkování

Na tomto místě bych chtěl poděkovat vedoucímu své práce RNDr. Romanu Plchovi, Ph.D. za trpělivost, čas strávený při konzultacích a poskytnutí vhodných materiálů týkajících se tématu této práce. Velký dík za trpělivost a pochopení patří i mé přítelkyni Vlastě.

Prohlášení

Prohlašuji, že jsem svou diplomovou práci vypracoval samostatně s využitím informačních zdrojů, které jsou v práci citovány.

Brno 4. ledna 2018 .....

Mikuláš Múdry

Obsah

Úvod

Řešení diferenciálních rovnic patří mezi základní partie matematiky a matematické analýzy. Jejich zkoumání může značně zjednodušit použití vhodných systémů počítačové algebry, ať už k ověření správnosti vlastních výpočtů, či ke grafickému znázornění průběhu konkrétních funkcí, partikulárních i obecných řešení.

V této práci popisujeme možnosti volně dostupného systému počítačové algebry Maxima v oblasti řešení a vizualizace diferenciálních rovnic prvního a druhého řádu. Pomocí příkazů, které nacházejí analytické (přesné) řešení, i příkazů počítajících řešení numerickými metodami řešíme jak základní známé typy rovnic, tak rovnice obecných tvarů. Zvláštní důraz je kladen na to, abychom se kromě použití nabízených příkazů Maximy pokusili řešení nalézt i sami ručně pomocí elementárních úprav, které u každého typu rovnice popisujeme. Před každým novým příkazem je uvedena stručná teorie, která definuje základní pojmy a nastiňuje způsob výpočtu pomocí daného příkazu.

Práce předpokládá základní znalost systému Maxima a práce v něm, jako je vyhodnocování a ukládání výrazů, jejich úprava nebo aplikace substitucí, neboť jednotlivé nejběžnější příkazy už nejsou zvlášť popisovány. Díky detailním komentářům postupů ale s porozuměním příkazům nebude mít problém ani čtenář, který nemá zkušenosti přímo s Maximou, ale alespoň s jiným systémem počítačové algebry.

Podobně práce předpokládá elementární znalost diferenciálního počtu funkcí jedné a více proměnných, například pojmů derivování a integrace.

Během celé práce pracujeme v grafické nástavbě wxMaxima, ačkoli o ní vždy mluvíme zkráceně jako o Maximě.

Kapitola 1

Obyčejné diferenciální rovnice prvního řádu

Diferenciální rovnicí prvního řádu rozumíme rovnici

$$y' = f(x,y), \quad y' = \frac{\mathrm{d}y}{\mathrm{d}x},$$

kde funkce f je reálná funkce definovaná na nějaké podmnožině G euklidovského prostoru $\mathbb{R}^2$. Jejím řešením je funkce $y(x)$, která je diferencovatelná na nějakém intervalu J a splňuje

$$[x, y(x)] \in G \quad \text{a} \quad y'(x) = f(x, y(x))$$

pro všechna $x \in J$.

Nechť $[x_0, y_0]$ je bod v G. Tzv. počáteční úloha je úloha určit řešení rovnice $y' = f(x,y)$ vyhovující (počáteční) podmínce $y(x_0) = y_0$. Toto řešení pak nazýváme partikulární. Obecným řešením rovnice $y' = f(x,y)$ chápeme funkci závislou na parametru C takovou, že konkrétní volbou C můžeme získat řešení každé počáteční úlohy $y(x_0) = y_0$, tedy každé partikulární řešení.

V této kapitole ukážeme příkazy Maximy, které vrací analytické i numerické řešení obecných i speciálních typů diferenciálních rovnic prvního řádu. Na konci kapitoly pak jednu z geometrických aplikací v podobě izogonální trajektorie.

1.1 Analytické řešení

1.1.1 Příkaz `ode2`

Základní příkaz pro řešení obyčejných diferenciálních rovnic prvního i druhého řádu v Maximě představuje příkaz `ode2`. Jeho syntaxe je

`ode2(difRce, y, x)`,

kde `difRce` je analyzovaná diferenciální rovnice buď tvaru $\texttt{difRce}_{\texttt{L}} = \texttt{difRce}_{\texttt{P}}$ nebo pouze výraz bez znaku rovnosti (přičemž `ode2` pak automaticky předpokládá, že tento výraz pokládáme roven nule), `y` je závislá a `x` nezávislá proměnná. Nejprve příkaz testuje, zda je daná rovnice prvního nebo druhého řádu. U rovnic prvního řádu pak nalezne řešení

tak, že zkouší danou diferenciální rovnici postupně řešit metodami (v tomto pořadí): lineární, se separovanými proměnnými, exaktní (včetně případného integračního faktoru), homogenní, Bernoulliovou a zobecněnou homogenní metodou. Příkaz `ode2` vrací řešení pro závislou proměnnou buď v explicitní nebo implicitní formě. Po vyřešení Maxima uloží do proměnné `method` řetězec označující, která z metod byla pomocí `ode2` úspěšně použita. K reprezentaci integrační konstanty v řešení je použita sekvence znaků `%c`, případně `%cN`, kde $N \in \mathbb{N}$, pokud je během výpočtů potřeba použít integračních konstant více a je potřeba je odlišit. V případě neúspěšného výpočtu vrací příkaz `ode2` výsledek `false`.

Pro zápis derivace y', kde $y = y(x)$, pomocí příkazu
`diff(y, x)`
máme dvě možnosti, jak docílit toho, aby Maxima nevyhodnotila výraz jako 0, což většinou nechceme. Buď předem definujeme, že y je funkce proměnné x pomocí
`depends(y, x)`
nebo můžeme vyhodnocení příkazu `diff` potlačit symbolem `'` zapsaným před příkaz, tedy `'diff(y, x)`.
Ve většině případů bude vhodnější použít první způsob pomocí `depends`.

Rovnice se separovanými proměnnými

Diferenciální rovnice se separovanými proměnnými jsou rovnice tvaru

$$y' = \frac{f(x)}{g(y)}.$$

Řešení najdeme tak, že proměnné x a y převedeme na opačné strany rovnice a tyto strany zvlášť integrujeme. Tedy z tvaru rovnice $g(y)\mathrm{d}y = f(x)\mathrm{d}x$ dostaneme tvar $G(y) = F(x) + c$, kde $G(y) = \int g(y)\,\mathrm{d}y$, $F(x) = \int f(x)\,\mathrm{d}x$ a $c \in \mathbb{R}$.

Příklad 1.1.1. Vyřešíme rovnici se separovanými proměnnými $y - y^2 + xy' = 0$.

Ruční postup by byl následující:

```
(%i2) depends(y, x)$
      DR: y - y^2 + x*diff(y,x) = 0;
(DR)
```

$$x\left(\frac{d}{dx}y\right) - y^2 + y = 0$$

```
(%i3) DR2: solve(DR, diff(y,x))[1];
(DR2)
```

$$\frac{d}{dx}y = \frac{y^2 - y}{x}$$

```
(%i6) f: 1/denom(rhs(DR2));
      g: 1/num(rhs(DR2));
      integrate(f, x) = integrate(g, y) + %c;
(f)
```

$$\frac{1}{x}$$

```
(g)
```

$$\frac{1}{y^2-y}$$

```
(%o6)
```

$$\log(x) = -\log(y) + \log(y-1) + \%c$$

Při použití příkazu `ode2`, kterému jako argument dáme původní neupravenou rovnici, dostaneme stejný výsledek. Ten můžeme dále upravit, abychom vyjádřili y.

```
(%i7) ode2(DR, y, x);
(%o7)
```

$$\log(y-1) - \log(y) = \log(x) + \%c$$

```
(%i8) exp(lhs(%)) = exp(rhs(%));
(%o8)
```

$$\frac{y-1}{y} = \%e^{\%c}x$$

```
(%i9) OR: solve(%, y)[1];
(OR)
```

$$y = -\frac{1}{\%e^{\%c}x - 1}$$

```
(%i10) method;
(%o10)
```

separable

Tím jsme získali obecné řešení, které příkaz `ode2` spočítal metodou separovaných proměnných. K nalezení partikulárního řešení zadaného počáteční podmínkou nabízí Maxima příkaz
`ic1(OR, x=x0, y=y0)`,
kde `OR` je obecné řešení diferenciální rovnice a `[x0,y0]` zadaný bod, kterým má partikulární řešení procházet.

Najdeme partikulární řešení dané diferenciální rovnice pro počáteční podmínku $y(1) = 2$.

```
(%i11) PR: ic1(OR, x=1, y=2);
(PR)
```

$$y = -\frac{2}{x-2}$$

Nyní ještě provedeme zkoušku, zda je toto partikulární řešení opravdu řešením původní rovnice. Jednak ověříme, zda řešení vyhovuje parametrům příkazu `ic1`, a jednak, že řešení dosazené zpět do původního výrazu `DR` nám dá požadovanou hodnotu 0.

```
(%i12) PR, x=1;
(%o12)
```

$$y = 2$$

```
(%i13) DR, PR;
(%o13)
```

$$\left(\frac{d}{dx}\left(-\frac{2}{x-2}\right)\right)x - \frac{2}{x-2} - \frac{4}{(x-2)^2} = 0$$

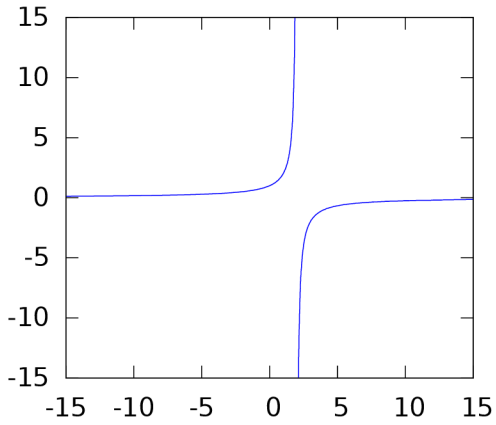
```
(%i14) %, diff, ratsimp;
(%o14)
```

$$0 = 0$$

Obě zkoušky tedy prošly. Nakonec si zobrazíme graf partikulárního řešení rovnice. Jednou z možností je použít příkaz `draw2d` z balíčku `draw`, který musíme nejprve načíst. V příkazu `draw2d` nastavíme vhodné rozmezí os x a y volbami `xrange=[xMin,xMax]` a `yrange=[yMin,yMax]` a předáme k vykreslení objekt typu `explicit(f(x), x, xMin, xMax)`, vykreslující explicitně zadanou funkci $f(x)$, kterou získáme jako pravou stranu rovnice partikulárního řešení.

```
(%i16) load(draw)$
       draw2d(xrange=[-15,15], yrange=[-15,15],
         explicit(rhs(PR), x, -15, 15))$
```

(viz obrázek 1.1)



Obrázek 1.1: $y = -\frac{2}{x-2}$ (řešení rovnice $y - y^2 + xy' = 0$ pro podmínku $y(1) = 2$)

U diferenciálních rovnic tvaru

$$y' = f(ax + by + c)$$

použijeme nejprve substituci $u = ax+by+c$. Z ní dostáváme, že $y = \frac{u-ax-c}{b}$ a tedy $y' = \frac{u'-a}{b}$. Dosazením do původního tvaru rovnice máme $u' = bf(u)+a$, tedy $\frac{\mathrm{d}u}{bf(u)+a} = \mathrm{d}x$, čímž získáváme rovnici se separovanými proměnnými.

Příklad 1.1.2. Vyřešíme diferenciální rovnici $y' = (4x+y)^2 + 4(4x+y)$.

Nejprve opět pomocí ručních úprav:

```
(%i2) depends(y, x)$
      DR: diff(y,x) = (4*x+y)^2 + 4*(4*x+y);
(DR)
```

$$\frac{d}{dx}y = (4x+y)^2 + 4\,(4x+y)$$

```
(%i4) depends(u, x)$
      SUB: u = 4*x+y;
(SUB)
```

$$u = y+4x$$

```
(%i6) solve(SUB, y)[1]$
      diff(%, x);
(%o6)
```

$$\frac{d}{dx}y = \frac{d}{dx}u - 4$$

```
(%i8) subst(rhs(%), diff(y,x), DR)$
      subst(lhs(SUB), rhs(SUB), %);
(%o8)
```

$$\frac{d}{dx}u - 4 = u^2 + 4u$$

```
(%i9) DR_SEP: solve(%, diff(u,x))[1];
(DR_SEP)
```

$$\frac{d}{dx}u = u^2 + 4u + 4$$

Úpravami jsme zadanou rovnici převedli na rovnici se separovanými proměnnými, na kterou použijeme příkaz `ode2`.

```
(%i10) ode2(DR_SEP, u, x);
(%o10)
```

$$-\frac{1}{u+2} = x + \%c$$

```
(%i13) subst(rhs(SUB), lhs(SUB), %)$
       solve(%, y)[1]$
       OR: partfrac(%, x);
(OR)
```

$$y = -\frac{1}{x+\%c} - 4x - 2$$

```
(%i14) method;
(%o14)
```

separable

Tím jsme získali obecné řešení. Použitím příkazu `ode2` na rovnici ze zadání přímo, bez předchozích úprav, příkaz výsledek v tomto případě nenajde a vrátí `false`.

```
(%i16) ode2(DR, y, x);
       method;
(%o15)
```

false

```
(%o16)
```

none

Homogenní rovnice

Homogenní diferenciální rovnice je tvaru

$$y' = f(\frac{y}{x})$$

nebo obecně

$$y' = f\left(\frac{ax+by+c}{Ax+By+C}\right).$$

V prvním případě aplikací vhodné substituce $u = \frac{y}{x}$, tj. $y' = u + u'x$, rovnici převedeme na rovnici se separovanými proměnnými $u + u'x = f(u)$ neboli $u' = \frac{f(u)-u}{x}$. V druhém případě záleží na tom, zda má soustava $\{ax+by+c=0, Ax+By+C=0\}$ řešení nebo ne.

- Má-li řešení $x = x_0, y = y_0$, provedeme substituce $u = x - x_0, v = y - y_0$, díky kterým původní obecný tvar homogenní rovnice převedeme na jednodušší tvar $\frac{\mathrm{d}v}{\mathrm{d}u} = F\left(\frac{v}{u}\right)$, který vyřešíme výše uvedeným způsobem.
- Pokud soustava řešení nemá, znamená to, že existuje $k \in \mathbb{R}$ takové, že $ax + by = k(Ax + By)$ a $c \neq kC$. Pak substitucí $u = ax + by$ získáme rovnici se separovanými proměnnými.

Příklad 1.1.3. Vyřešíme homogenní diferenciální rovnici $xy' - y = xe^{\frac{y}{x}}$.

Nejprve postupem popsaným výše, to je převedením pomocí substituce na rovnici se separovanými proměnnými.

```
(%i2) depends(y, x)$
      DR: x*diff(y,x) - y = x*exp(y/x);
(DR)
```

$$x\left(\frac{d}{dx}y\right)-y=x\,\%e^{\frac{y}{x}}$$

```
(%i3) DR2: expand(solve(DR, diff(y,x))[1]);
(DR2)
```

$$\frac{d}{dx}y=\%e^{\frac{y}{x}}+\frac{y}{x}$$

```
(%i5) depends(u, x)$
      SUB: u = y/x;
(SUB)
```

$$u=\frac{y}{x}$$

```
(%i7) solve(SUB, y)[1]$
      SUB2: diff(%, x);
(SUB2)
```

$$\frac{d}{dx}y=\left(\frac{d}{dx}u\right)x+u$$

```
(%i10) subst(rhs(SUB2), lhs(SUB2), DR2)$
       subst(lhs(SUB), rhs(SUB), %)$
       DR_SEP: solve(%, diff(u,x))[1];
(DR_SEP)
```

$$\frac{d}{dx}u=\frac{\%e^{u}}{x}$$

Nyní už máme rovnici se separovanými proměnnými, kterou můžeme vyřešit pomocí příkazu `ode2`.

```
(%i12) ode2(DR_SEP, u, x)$
       subst(rhs(SUB), lhs(SUB), %);
(%o12)
```

$$-\%e^{-\frac{y}{x}}=\log(x)+\%c$$

```
(%i13) solve(%, y)[1];
(%o13)
```

$$y=x\log\left(-\frac{1}{\log(x)+\%c}\right)$$

```
(%i14) method;
(%o14)
```

separable

Při použití příkazu `ode2` na rovnici přímo, bez dalších ručních úprav, dostaneme po vyjádření y stejný výsledek (liší se jen opticky, protože $\ln(cx)=\ln(c)+\ln(x)=\ln(x)+\bar{c}$).

```
(%i15) ode2(DR, y, x);
(%o15)
```

$$\%cx = \%e^{-\%e^{-\frac{y}{x}}}$$

```
(%i17) log(%)$
       solve(%, y)[1];
(%o17)
```

$$y = x \log\left(-\frac{1}{\log(\%cx)}\right)$$

```
(%i18) method;
(%o18)
```

homogeneous

Příklad 1.1.4. Vyřešíme i homogenní diferenciální rovnici v obecnějším tvaru $y' = \left(\frac{y+1}{y-x}\right)^2$.

Při ručním výše popsaném postupu bychom mohli postupovat následně:

```
(%i2) depends(y, x)$
      DR: diff(y,x) = ((y + 1)/(y - x))^2;
(DR)
```

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

```
(%i3) [x0,y0]: linsolve([y+1,y-x], [x,y]);
(%o3)
```

$$[x = -1, y = -1]$$

```
(%i7) depends(v, u)$
      SUB1: u = x - rhs(x0);
      SUB2: v = y - rhs(y0);
      SUB3: diff(lhs(SUB2), u) = diff(rhs(SUB2), x);
(SUB1)
```

$$u = x + 1$$

```
(SUB2)
```

$$v = y + 1$$

```
(SUB3)
```

$$\frac{d}{du}v = \frac{d}{dx}y$$

```
(%i10) subst(lhs(SUB3), rhs(SUB3), DR)$
       ratsubst(lhs(SUB1), rhs(SUB1), %)$
       DR2: ratsubst(lhs(SUB2), rhs(SUB2), %);
(DR2)
```

$$\frac{d}{du}v = \frac{v^2}{v^2 - 2u\,v + u^2}$$

V tuto chvíli máme původní diferenciální rovnici převedenou do jednodušší formy $\frac{\mathrm{d}v}{\mathrm{d}u} = F\left(\frac{v}{u}\right)$ (což lze vidět při rozšíření zlomku na pravé straně výrazem $\frac{1}{u^2}$) a použijeme na ni příkaz `ode2`.

```
(%i13) ode2(DR2, v, u)$
       ratsubst(rhs(SUB2), lhs(SUB2), %)$
       ratsubst(rhs(SUB1), lhs(SUB1), %);
(%o13)
```

$$\%c\,(x+1) = \%e^{-\frac{\sqrt{5}\log\left(\frac{1+y}{x+1}\right)+\log\left(\frac{-1+\sqrt{5}(-x-1)-3x+2y}{2y+\sqrt{5}(1+x)-3x-1}\right)}{\sqrt{5}}}$$

```
(%i15) radcan(%)$
       solve(%, y)[1];
(%o15)
```

$$y = \frac{\left(-1+\sqrt{5}+\left(\sqrt{5}-3\right)x+2y\right)^{\frac{1}{\sqrt{5}}} - \%c\left(-1-\sqrt{5}+\left(-\sqrt{5}-3\right)x+2y\right)^{\frac{1}{\sqrt{5}}}}{\%c\left(-1-\sqrt{5}+\left(-\sqrt{5}-3\right)x+2y\right)^{\frac{1}{\sqrt{5}}}}$$

```
(%i16) method;
(%o16)
```

homogeneous

Použití `ode2` na původní neupravenou rovnici selže a dostaneme chybovou hlášku.

```
(%i18) ode2(DR, y, x);
       method;
(%o17)
```

false

```
(%o18)
```

none

Lineární rovnice 1. řádu

Rovnice lineární 1. řádu mají tvar

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

Při jejich řešení postupujeme tak, že nejprve vyřešíme jejich zkrácenou (homogenní) rovnici $y' = a(x)y$. Jedná se o rovnici se separovanými proměnnými, takže jde opět o postup popsaný výše. Obecné řešení homogenní rovnice je $y = ce^{\int a(x)\,dx}$, kde $c \in \mathbb{R}$ je integrační konstanta. Původní, nehomogenní, rovnici pak vyřešíme metodou „variace konstanty“: vycházíme z předpokladu, že obecné řešení homogenní rovnice bude řešením i rovnice nehomogenní, pokud integrační konstantu nahradíme určitou funkcí $c(x)$, tedy

$$y_1 = c(x)e^{\int a(x)\,dx}.$$

Řešení y_1 dosadíme do nehomogenní rovnice místo y, podobně i jeho derivaci

$$y_1' = c'(x)e^{\int a(x)\,dx} + c(x)e^{\int a(x)\,dx}a(x).$$

Tím získáme rovnici

$$c'(x)e^{\int a(x)\,dx} + c(x)e^{\int a(x)\,dx}a(x) = a(x)c(x)e^{\int a(x)\,dx} + b(x),$$

která se po zkrácení členů s $c(x)$ na obou stranách zjednoduší na

$$c'(x)e^{\int a(x)\,dx} = b(x).$$

Odtud integrací spočteme hledanou funkci $c(x)$, dosadíme do y_1 a máme tak celkové řešení.

Příklad 1.1.5. Spočteme lineární rovnici 1. řádu $2xy' + x^2 - 6y = 0$.

Při ručním počítání metodou variace konstanty bychom postupovali následně:

```
(%i2) depends(y, x)$
      DR: 2*x*diff(y,x) + x^2 - 6*y = 0;
(DR)
```

$$2x\left(\frac{d}{dx}y\right) - 6y + x^2 = 0$$

```
(%i3) DR2: expand(solve(DR, diff(y,x))[1]);
(DR2)
```

$$\frac{d}{dx}y = \frac{3y}{x} - \frac{x}{2}$$

```
(%i4) HOM: lhs(DR2) = part(rhs(DR2), 1);
(HOM)
```

$$\frac{d}{dx}y = \frac{3y}{x}$$

```
(%i5) HOM_OR: ode2(HOM, y, x);
(HOM_OR)
```

$$y = \%c x^3$$

```
(%i7) depends(C, x)$
      Y1: subst(C, %c, HOM_OR);
(Y1)
```

$$y = x^3 C$$

```
(%i10) DR2, diff(Y1, x), Y1$
       solve(%, diff(C, x))[1]$
       C_RES: C = rhs(integrate(%, x));
(C_RES)
```

$$C = \frac{1}{2x} + \%c1$$

```
(%i11) Y1, C_RES;
(%o11)
```

$$y = \left(\frac{1}{2x} + \%c1\right) x^3$$

Použitím příkazu `ode2` na původní neupravenou rovnici dostaneme tentýž výsledek.

```
(%i13) ode2(DR, y, x);
       method;
(%o12)
```

$$y = \left(\frac{1}{2x} + \%c\right) x^3$$

```
(%o13)
```

linear

Bernoulliova rovnice

Při řešení Bernoulliovy diferenciální rovnice řádu r, která má tvar

$$y' + a(x)y = b(x)y^r,$$

kde $r \in \mathbb{R} \wedge r \neq 1 \wedge r \neq 0$ (pro $r = 0 \vee r = 1$ dostáváme lineární rovnici), nejdříve celou rovnici vydělíme y^r, čímž dostaneme $\frac{y'}{y^r} + a(x)y^{1-r} = b(x)$. Pomocí substituce mocniny y u $a(x)$, tedy $z = y^{1-r}$, a její derivace $z' = (1-r)\frac{y'}{y^r}$ rovnici převedeme na lineární rovnici $\frac{z'}{1-r} + a(x)z = b(x)$, neboli

$$z' + c(x)z = d(x).$$

Po vyřešení Bernoulliovy rovnice je do proměnné `odeindex` uložen řád rovnice.

Příklad 1.1.6. Vyřešíme Bernoulliovu rovnici $xy' + y = y^2 \ln x$.

Nejdříve postupem pomocí úprav a substituce popsaných výše:

```
(%i2) depends(y, x)$
      DR: x*diff(y,x) + y = y^2*log(x);
(DR)
```

$$x\left(\frac{d}{dx}y\right) + y = \log(x)\,y^2$$

```
(%i5) depends(z, x)$
      SUB: z = y^(1-2);
      SUB2: diff(SUB, x);
(SUB)
```

$$z = \frac{1}{y}$$

```
(SUB2)
```

$$\frac{d}{dx}z = -\frac{\frac{d}{dx}y}{y^2}$$

```
(%i8) expand(solve(DR, diff(y,x))[1]/y^2)$
      subst(lhs(SUB), rhs(SUB), %)$
      DR2: ratsubst(lhs(SUB2), rhs(SUB2), %);
(DR2)
```

$$-\frac{d}{dx}z = -\frac{z - \log(x)}{x}$$

```
(%i11) ode2(DR2, z, x), SUB$
       OR: solve(%, y)[1];
       method;
(OR)
```

$$y = \frac{1}{\log(x) + \%cx + 1}$$

```
(%o11)
```

linear

Při použití příkazu `ode2` na diferenciální rovnici bez předchozích úprav dostaneme stejný výsledek.

```
(%i14) expand(ode2(DR, y, x));
       method;
       odeindex;
(%o12)
```

$$y = \frac{1}{\log(x) + \%cx + 1}$$

```
(%o13)
```

bernoulli

```
(%o14)
```

2

Exaktní rovnice

Diferenciální rovnice tvaru

$$M(x,y)\mathrm{d}x + N(x,y)\mathrm{d}y = 0$$

je exaktní tehdy, pokud je její levá strana rovna totálnímu diferenciálu určité kmenové funkce $F(x,y)$, tj. můžeme ji napsat jako

$$\mathrm{d}F(x,y) = \frac{\partial F}{\partial x}\mathrm{d}x + \frac{\partial F}{\partial y}\mathrm{d}y = 0$$

neboli $M(x,y) = \frac{\partial F}{\partial x}, N(x,y) = \frac{\partial F}{\partial y}$. Za předpokladu spojitosti parciálních derivací $\frac{\partial M}{\partial y}$ a $\frac{\partial N}{\partial x}$ ze Schwarzovy věty o rovnosti smíšených derivací dostáváme $\frac{\partial M}{\partial y} = \frac{\partial N}{\partial x}$ a celkem tedy platí, že rovnice je exaktní právě tehdy, když platí tato rovnost. Abychom vyřešili exaktní diferenciální rovnici, chceme nalézt zmíněnou kmenovou funkci $F(x,y)$ a obecné řešení pak můžeme psát implicitně jako $F(x,y) = c$, kde $c \in \mathbb{R}$.

Postup k nalezení kmenové funkce může být například následující. Víme, že má platit vztah

$$M(x,y) = \frac{\partial F}{\partial x} \implies F = \int M(x,y)\partial x + G(y).$$

Dále má platit $N(x,y) = \frac{\partial F}{\partial y}$, kam dosadíme F z předchozího a máme

$$N(x,y) = \frac{\partial}{\partial y}\left(\int M(x,y)\partial x + G(y)\right) = \frac{\partial}{\partial y}\int M(x,y)\partial x + \frac{\mathrm{d}}{\mathrm{d}y}G(y)$$

a tedy po převedení známých funkcí M a N na jednu stranu získáváme rovnici

$$G'(y) = N(x,y) - \frac{\partial}{\partial y}\int M(x,y)\partial x,$$

odkud po integrování máme

$$G(y) = \int N(x,y)\mathrm{d}y - \int\left(\frac{\partial}{\partial y}\int M(x,y)\partial x\right)\mathrm{d}y.$$

Po dosazení právě spočítané $G(y)$ nalezneme hledanou kmenovou funkci $F(x,y)$.

Příklad 1.1.7. Spočteme exaktní rovnici tvaru $(y\cos x - x\sin y)\mathrm{d}x + (\sin x - \frac{x^2}{2}\cos y + + \operatorname{tg} y)\mathrm{d}y = 0$.

Ručním počítáním bychom postupovali následně:

```
(%i1) DR: (y*cos(x) - x*sin(y))
      + (sin(x) - x^2/2*cos(y) + tan(y))*'diff(y,x) = 0;
(DR)
```

$$\left(\sin(x) - \frac{x^2\cos(y)}{2} + \tan(y)\right)\left(\frac{d}{dx}y\right) - x\sin(y) + \cos(x)y = 0$$

```
(%i3) M: -num(rhs(solve(DR, 'diff(y,x))[1]));
      N: denom(rhs(solve(DR, 'diff(y,x))[1]));
(M)
```

$$2\cos(x)y - 2x\sin(y)$$

```
(N)
```

$$2\tan(y) - x^2\cos(y) + 2\sin(x)$$

```
(%i4) is(equal(diff(M,y), diff(N,x)));
(%o4)
```

$$\text{true}$$

```
(%i6) depends(G, y)$
      F: integrate(M, x) + G;
(F)
```

$$G - x^2\sin(y) + 2\sin(x)y$$

```
(%i9) N = diff(F,y)$
      solve(%, diff(G,y))[1]$
      integrate(rhs(%), y);
(%o9)
```

$$2\log(\sec(y))$$

```
(%i10) OR: subst(%, G, F) = %c;
(OR)
```

$$2\log(\sec(y)) - x^2\sin(y) + 2\sin(x)y = \%c$$

Použitím příkazu `ode2` přímo na zadanou rovnici získáme stejný výsledek (lišící se pouze vizuálně – po vynásobení dvojkou bychom dostali úplně totožný výsledek).

```
(%i12) ode2(DR, y, x);
       method;
(%o11)
```

$$\frac{2\sin(x)y - x^2\sin(y) + 2\log(\sec(y))}{2} = \%c$$

```
(%o12)
```

exact

V případě, že má diferenciální rovnice tvar $M(x,y)\mathrm{d}x + N(x,y)\mathrm{d}y = 0$, ale není exaktní, tedy $\frac{\partial M}{\partial y} \neq \frac{\partial N}{\partial x}$, a současně funkce $M(x,y)$ a $N(x,y)$ mají spojité parciální derivace a nejsou rovny nule, existuje funkce $R(x,y)$ taková, že rovnice $R(x,y)M(x,y)\mathrm{d}x + R(x,y)N(x,y)\mathrm{d}y = 0$ už exaktní je, neboli platí $\frac{\partial}{\partial y}(RM) = \frac{\partial}{\partial x}(RN)$. Této funkci $R(x,y)$ pak říkáme integrační faktor. Maxima ukládá použitý integrační faktor do proměnné `intfactor` (pokud zatím žádná exaktní diferenciální rovnice řešena nebyla, je nastavena implicitní hodnota `false`, pokud zadaná rovnice byla exaktní a násobení faktorem nebylo potřeba, má hodnotu 1).

Příklad 1.1.8. Vyřešíme diferenciální rovnici $(2xy^2 - y)\mathrm{d}x + (y^2 + x)\mathrm{d}y = 0$.

Ručně bychom při hledání integračního faktoru postupovali například takto:

```
(%i1) DR: 2*x*y^2-y + (y^2+x)*'diff(y,x) = 0;
(DR)
```

$$(x + y^2)\left(\frac{d}{dx}y\right) + 2xy^2 - y = 0$$

```
(%i3) M: -num(rhs(solve(DR, 'diff(y,x))[1]));
      N: denom(rhs(solve(DR, 'diff(y,x))[1]));
(M)
```

$$2xy^2 - y$$

```
(N)
```

$$y^2 + x$$

```
(%i4) is(equal(diff(M,y), diff(N,x)));
(%o4)
```

unknown

Pokud by byla rovnice exaktní, parciální derivace výše by musely být stejné. Funkce `is` ale nevrátila `true`. O exaktní rovnici se tedy nejedná, a tak se pokusíme nalézt integrační faktor. Předpokládejme například, že se jedná o funkci proměnné y, tj. $R = R(y)$.

```
(%i8) depends(R, y)$
      RMy: diff(R*M, y)$
      RNx: diff(R*N, x)$
      ode2(RMy = RNx, R, y);
(%o8)
```

$$R = \frac{\%c}{y^2}$$

Z podmínek, které musí pro exaktní rovnici platit, jsme získali integrační faktor. Za konstantu dosadíme například 1, ověříme, že rovnice je po vynásobení spočteným faktorem exaktní, a dopočteme kmenovou funkci $F(x,y)$.

```
(%i9) IF: rhs(subst(%c = 1, %));
(IF)
```

$$\frac{1}{y^2}$$

```
(%i10) is(equal(diff(IF*M,y), diff(IF*N,x)));
(%o10)
```

true

```
(%i13) ode2(DR*IF, y, x);
       method;
       intfactor;
(%o11)
```

$$\frac{-x+x^2y+y^2}{y} = \%c$$

```
(%o12)
```

exact

```
(%o13)
```

1

Pokud použijeme příkaz `ode2` na neupravenou rovnici ze zadání, vrátí nám stejný výsledek.

```
(%i16) ode2(DR, y, x);
       method;
       intfactor;
(%o14)
```

$$\frac{-x+x^2y+y^2}{y} = \%c$$

```
(%o15)
```

exact

```
(%o16)
```

$$\frac{1}{y^2}$$

Zobecněná homogenní rovnice

Zobecněnou homogenní diferenciální rovnici řádu n ve tvaru

$$y' = \frac{y}{x} G(yx^n),$$

kde $n \in \mathbb{R} \wedge n \neq 0 \wedge n \neq -1$ (v případě $n = 0$ by se jednalo o rovnici se separovanými proměnnými, v případě $n = -1$ o homogenní rovnici, kterou jsme řešili výše), řešíme pomocí substituce $u = yx^n$. Odtud $y' = \frac{u'}{x^n} - n\frac{u}{x^{n+1}} = \frac{u}{x^{n+1}} G(u)$, převedením proměnných u a x na opačné strany dostaneme

$$u' = \frac{u\,(n + G(u))}{x},$$

tedy rovnici se separovanými proměnnými. Po vyřešení zobecněné homogenní rovnice je do proměnné `odeindex` uložen řád rovnice.

Příklad 1.1.9. Vyřešíme diferenciální rovnici $xy' = y\ln(yx^2) - 2y$.

Pomocí ručně provedených substitucí popsaných výše by byl postup následující:

```
(%i2) depends(y, x)$
      DR: x*diff(y,x) = y*log(y*x^2)-2*y;
(DR)
```

$$x\left(\frac{d}{dx}y\right) = y\log\left(x^2 y\right) - 2y$$

```
(%i3) DR2: solve(DR, diff(y,x))[1];
(DR2)
```

$$\frac{d}{dx}y = \frac{y\log\left(x^2 y\right) - 2y}{x}$$

```
(%i7) depends(u, x)$
      SUB: u = y*x^2;
      SUB2: SUB/x^3;
      SUB3: diff(solve(SUB, y)[1], x);
(SUB)
```

$$u = x^2 y$$

```
(SUB2)
```

$$\frac{u}{x^3} = \frac{y}{x}$$

```
(SUB3)
```

$$\frac{d}{dx}y = \frac{\frac{d}{dx}u}{x^2} - \frac{2u}{x^3}$$

```
(%i10) ratsubst(lhs(SUB2), rhs(SUB2), DR2)$
       subst(rhs(SUB3), lhs(SUB3), %)$
       DR_SEP: solve(%, diff(u,x))[1];
(DR_SEP)
```

$$\frac{d}{dx}u = \frac{u\log(u)}{x}$$

```
(%i12) OR: ode2(DR_SEP, u, x);
       method;
(OR)
```

$$\log(\log(u)) = \log(x) + \%c$$

```
(%o12)
```

separable

```
(%i14) subst(rhs(SUB), lhs(SUB), OR)$
       solve(%, y)[1];
(%o14)
```

$$y = \frac{\%e^{\%e^{\%c}x}}{x^2}$$

Použitím `ode2` na neupravenou rovnici ze zadání dostaneme po vyjádření y stejný výsledek (liší se pouze vizuálně, protože $\mathrm{e}^{\mathrm{e}^c x} = \mathrm{e}^{\frac{1}{\mathrm{e}^c}x} = \mathrm{e}^{\frac{x}{\mathrm{e}^c}}$).

```
(%i18) ode2(DR, y, x)$
       solve(%, y)[1];
       method;
       odeindex;
(%o16)
```

$$y = \frac{\%e^{\frac{x}{\%c}}}{x^2}$$

```
(%o17)
```

genhom

```
(%o18)
```

Příklad rovnic, které `ode2` nevyřeší

Kromě konkrétních tvarů diferenciálních rovnic, které jsme viděli výše a které `ode2` nedokáže vyřešit, příkaz nenajde řešení diferenciálních rovnic nerozřešených vzhledem k derivaci. Protože u nich není možné vyjádřit y', nemůže aplikovat žádný z postupů, kterými řeší známé typy diferenciálních rovnic prvního řádu (viz úvod k příkazu `ode2` v kapitole 1.1.1). Například Lagrangeovu rovnici, která má tvar $y = f(y')x + g(y')$ – samozřejmě pokud nejsou funkce f a g takového tvaru, že by Lagrangeova rovnice byla zároveň některým ze základních typů diferenciálních rovnic prvního řádu, které jsme řešili dříve. Konkrétně můžeme ukázat její speciální případ, Clairautovu rovnici tvaru $y = y'x + g(y')$, tedy případ, kdy $f(y') \equiv y'$.

Příklad 1.1.10. Vyřešíme diferenciální rovnici $y = xy' + y' + y'^2$.

Použitím `ode2` zjistíme, že řešení tohoto typu rovnic příkaz nenajde:

```
(%i2) depends(y, x)$
      DR: y = x*diff(y,x) + diff(y,x) + diff(y,x)^2;
(DR)
```

$$y = \left(\frac{d}{dx}y\right)^2 + x\left(\frac{d}{dx}y\right) + \frac{d}{dx}y$$

```
(%i4) ode2(DR, y, x);
      method;
(%t3)
```

$$y = \left(\frac{d}{dx}y\right)^2 + x\left(\frac{d}{dx}y\right) + \frac{d}{dx}y$$

first order equation not linear in y'

```
(%o3)
```

false

```
(%o4)
```

none

Nejen Lagrangeově a Clairautově rovnici ale obecně rovnicím prvního řádu nerozřešeným vzhledem k derivaci se věnuje kapitola 3. Popisuje také podpůrný balíček funkcí, který vznikl jako příloha této práce a který zmíněný typ rovnic, na rozdíl od příkazu `ode2`, dokáže řešit.

1.1.2 Příkaz `desolve`

Jiným příkazem, který můžeme použít k řešení konkrétně lineárních diferenciálních rovnic, je příkaz `desolve`. Hledá řešení pomocí Laplaceovy transformace (viz níže) a má tvar
`desolve(difRce, y(x))`,
kde `difRce` je zkoumaná diferenciální rovnice a `y(x)` jí příslušící závislá proměnná. Na rozdíl od příkazu `ode2` musí být závislost na nezávislé proměnné přímo naznačena jak u závislé proměnné, tak i u její derivace, například `y(x)` a `diff(y(x),x)`. Dalším rozdílem je podmínka, aby všechny `difRce` obsahovaly i znak rovnosti $=$, na rozdíl od `ode2`, který při absenci rovnítka pokládal výraz roven nule. Během hledání řešení pomocí `desolve` Maxima zavádí generované proměnné tvaru `gABCDE` – do nich transformuje zadanou rovnici a na ně nakonec aplikuje inverzní transformaci k získání celkového řešení. Pokud nedojde k žádnému problému při transformacích, neměly by tyto symboly ve výstupu `desolve` nijak figurovat (viz rozdílné výstupy v příkladech 1.1.11 a 1.1.12). V případě, že `desolve` nedokáže řešení najít, vrací `false`.

Laplaceova transformace

Jedním z postupů, jak řešit diferenciální rovnice, je postup využívající Laplaceovu transformaci. Ta je pro funkci $f(x)$, definovanou pro všechna $x \geq 0$ a splňující $|f(x)| \leq Me^{-kx}$ pro nějaké konstanty M a k, dána vzorcem

$$\mathscr{L}[f(x)](t) = \int_0^\infty f(x)e^{-tx}dx = F(t).$$

Z tohoto vzorce plyne vlastnost transformace, kdy se derivace převádí na násobení nezávislou proměnnou. Platí totiž, že $\mathscr{L}[f'(x)] = \int_0^\infty f'(x)e^{-tx}dx$, a odtud dále metodou per partes získáme $[f(x)e^{-tx}]_0^\infty + t\int_0^\infty f(x)e^{-tx}dx = (0 - f(0)) + t\mathscr{L}[f(x)] = t\mathscr{L}[f(x)] - f(0)$. Díky této vlastnosti je možné původní diferenciální rovnici převést na obyčejnou lineární rovnici, tu vyřešit a pomocí inverzní Laplaceovy transformace získat řešení v původní nezávislé proměnné. Nevýhodou je, že naopak násobení nezávislou proměnnou se převádí na derivaci, konkrétně $\mathscr{L}[xf(x)] = -(\mathscr{L}[f(x)])'$. V některých případech tedy transformace rovnici nezjednoduší, ale naopak převede na diferenciální rovnici vyššího řádu.

Inverzní transformace je definována jako

$$\mathscr{L}^{-1}[F(t)](x) = \frac{1}{2\pi i}\int_{\gamma-i\infty}^{\gamma+i\infty} F(t)e^{tx}dt = f(x).$$

Maxima nabízí k aplikaci Laplaceovy transformace funkci
`laplace(f(x), x, t)`,
kde $f(x)$ je funkce, kterou chceme transformovat, x její nezávislá proměnná a t nezávislá proměnná obrazu $f(x)$.

Inverzní Laplaceovu transformaci vrací příkaz `ilt` s podobnou syntaxí jako `laplace`:
`ilt(f(t), t, x)`,
kde $f(t)$ je transformovaná funkce a x proměnná jejího obrazu.

Lineární rovnice 1. řádu

Kromě obecného řešení dokáže `desolve` najít i řešení partikulární, a to například v kombinaci s příkazem
`atvalue(y(x), x=x0, y0)`,
čímž v Maximě nastavíme, že pro funkci $y(x)$ platí $y(x_0) = y_0$. Příkaz `atvalue` musí být zadán před použitím `desolve`. Pokud počáteční podmínku nezadáme, `desolve` vrátí výsledek, který bude obsahovat konstanty `y(0)`, což plyne z definice Laplaceovy transformace výše. Konkrétně před voláním `desolve` použijeme
`atvalue(y(x), x=0, y0)`,
protože `desolve` při zadané hodnotě $y(x_1)$, kde $x_1 \neq 0$, neodvozuje, jaká je hodnota $y(0)$.

Příklad 1.1.11. Najdeme řešení jednoduché lineární rovnice $y' + y = x$.

```
(%i3) atvalue(y(x), x=0, 1)$
      DR: diff(y(x),x) + y(x) = x;
      PR: desolve(DR, y(x));
(DR)
```

$$\frac{d}{dx}y(x) + y(x) = x$$

```
(PR)
```

$$y(x) = 2\%e^{-x} + x - 1$$

Správnost řešení můžeme ověřit jednak dosazením 0 za x a jednak dosazením celého partikulárního řešení do zadání. Následně můžeme ověřit, že kombinace příkazů `ode2` a `ic1` by vrátila stejné řešení. K vyhodnocení derivací použijeme příkaz `ev` s argumentem `nouns` – díky němu se vyhodnotí nevyhodnocené funkce jako například `diff` nebo `integrate`.

```
(%i6) PR, x=0;
      DR, PR$
      ev(%, nouns);
(%o4)
```

$$1 = 1$$

```
(%o6)
```

$$x = x$$

```
(%i8) ode2(DR, y(x), x)$
      expand(ic1(%, x=0, y(x)=1));
(%o8)
```

$$y(x) = 2\%e^{-x} + x - 1$$

Obě zkoušky tedy prošly a `ode2` nachází totožné řešení.

U většiny složitějších rovnic nám ale bohužel `desolve` řešení nenajde. Vidět to bude na následujícím příkladu, kdy vyřešíme stejnou lineární rovnici jako v příkladu 1.1.5:

Příklad 1.1.12. Vyřešíme lineární rovnici $2xy' + x^2 - 6y = 0$.

```
(%i2) DR: 2*x*diff(y(x),x) + x^2 - 6*y(x) = 0;
      desolve(DR, y(x));
(DR)
```

$$2x\left(\frac{d}{dx}y(x)\right) - 6y(x) + x^2 = 0$$

```
(%o2)
```

$$y(x) = \text{ilt}\left(-\frac{g19036^4\left(\frac{d}{dg19036}\text{laplace}\left(y(x),x,g19036\right)\right) - 1}{4g19036^3}, g19036, x\right)$$

Použitím `desolve` výsledek nezískáme. Vidíme, že se Maximě nepodařilo spočítat inverzní Laplaceovu transformaci, pravděpodobně proto, že její argument obsahoval derivaci Laplaceovy transformace funkce $y(x)$ a celá rovnost ve výstupu `(%o2)` tak tvoří novou lineární diferenciální rovnici. Rovnici ze zadání ale můžeme vyřešit ručně podobným postupem, jaký používá `desolve` s tím, že až na „vloženou“ lineární rovnici narazíme, vyřešíme ji pomocí `ode2` (pro větší přehlednost zavedeme substituci).

```
(%i4) laplace(DR, x, t);
      DR2: solve(%, diff(laplace(y(x),x,t),t))[1];
(%o3)
```

$$2\left(-\text{laplace}\left(y(x),x,t\right) - t\left(\frac{d}{dt}\text{laplace}\left(y(x),x,t\right)\right)\right) - 6\text{laplace}\left(y(x),x,t\right) + \frac{2}{t^3} = 0$$

```
(DR2)
```

$$\frac{d}{dt}\text{laplace}\left(y(x),x,t\right) = -\frac{4t^3\,\text{laplace}\left(y(x),x,t\right) - 1}{t^4}$$

```
(%i8) SUB: u = laplace(y(x),x,t)$
      subst(lhs(SUB), rhs(SUB), DR2);
      DR3: ode2(%, u, t);
      method;
(%o6)
```

$$\frac{d}{dt}u = -\frac{4t^3u - 1}{t^4}$$

```
(DR3)
```

$$u = \frac{\%c + t}{t^4}$$

```
(%o8)
```

linear

```
(%i10) subst(rhs(SUB), lhs(SUB), DR3);
       ilt(%,t,x);
(%o9)
```

$$\text{laplace}\,(y(x),x,t) = \frac{\%c+t}{t^4}$$

```
(%o10)
```

$$y(x) = \frac{\%c\,x^3}{6} + \frac{x^2}{2}$$

Dosazením tohoto řešení do zadání a následným zjednodušením výrazu ověříme, že se skutečně jedná o řešení zadané rovnice.

```
(%i12) DR, %$
       ratsimp(ev(%, nouns));
(%o12)
```

$$0 = 0$$

1.2 Numerické řešení

1.2.1 Příkaz `plotdf`

Numerické řešení obyčejné diferenciální rovnice prvního řádu zároveň s jeho znázorněním je možné získat použitím příkazu `plotdf`. Příkaz pomocí Adamsovy-Moultonovy metody (viz níže) numericky integruje rovnici zadanou ve formě $y' = f(x,y)$ a vykreslí směrové pole vektorů znázorněné implicitně modrými šipkami, které pokrývá všechna partikulární řešení. Délka jednotlivých šipek reflektuje velikost změny y v okolí daného bodu mřížky. Příkaz `plotdf` tedy neumí zobrazovat řešení rovnic nerozřešených vzhledem k derivaci y'. Úhel vektorů α je v jednotlivých bodech mřížky dán vztahem $\text{tg}\,\alpha = f(x,y)$.

K použití `plotdf` je nutné mít nainstalován program Xmaxima, který je k samotnému vykreslení použit. Vykreslení probíhá vždy do nového okna a není možné graf zobrazit přímo v zápisníku Maximy přidáním předpony „wx“. Syntaxe příkazu je
`plotdf(f(x,y))`,
případně
`plotdf(f(t,u), [t,u])`,
pokud funkce f nezávisí na proměnných x a y (seznam proměnných `[t,u]` pak musí být uveden hned jako druhý parametr `plotdf`). Jako další v pořadí mohou být uvedeny volitelné parametry, které postupně použijeme níže u jednotlivých typů rovnic.

V případě, že Xmaxima nedokáže pro zadanou rovnici směrové pole spočítat, vypíše chybovou hlášku, například `Input file has syntax errors` spolu s podrobnějším popisem, kde na chybu narazila. Typicky se může jednat o chybu `domain error`, kdy zadaná rovnice není definována na celém intervalu, ve kterém se má numericky integrovat, a je tedy potřeba upravit rozmezí jedné nebo obou proměnných, kterých se omezený definiční obor týká.

Adamsovy-Moultonovy metody

Uvažme diferenciální rovnici ve tvaru $y'(x) = f(x, y(x))$ s počáteční podmínkou $y(x_0) = y_0$. Numerickou metodou chceme hledat neznámou funkci $y(x)$ a to tak, že budeme počítat přibližnou hodnotu v bodech (tzv. uzlech) x_i, kde $i = 0, \ldots, N$. Jednotlivé uzly jsou ekvidistantní se vzdáleností (krokem) h, tj. $x_{i+1} = x_i + h$ pro $i = 0, 1, \ldots, N-1$. V uzlu x_0 známe přesnou hodnotu $y(x_0) = y_0$, v dalších uzlech pak hodnoty $y_i \approx y(x_i)$ představují přibližné řešení, které získáme numericky Adamsovou-Moultonovou metodou.

Adamsovy-Moultonovy metody stojí na následujících myšlenkách. Integrací diferenciální rovnice $y'(x) = f(x, y(x))$, kterou řešíme, od uzlu x_i do x_{i+1} dostaneme

$$y(x_{i+1}) - y(x_i) = \int_{x_i}^{x_{i+1}} f(x, y(x)) \mathrm{d}x.$$

Pokud pak funkci f aproximujeme vhodným interpolačním polynomem $P(x)$ stupně $k-1$, který splňuje $P(x_{i+1-j}) = f(x_{i+1-j}, y(x_{i+1-j}))$ pro $j = 0, 1, \ldots, k-1$, získáme přibližnou hodnotu

$$y(x_{i+1}) \approx y(x_i) + \int_{x_i}^{x_{i+1}} P(x) \mathrm{d}x$$

a následným nahrazením přibližné rovnosti přesnou a přesných hodnot přibližnými dostaneme vzorec

$$y_{i+1} = y_i + \int_{x_i}^{x_{i+1}} P(x) \mathrm{d}x.$$

Rovnice se separovanými proměnnými

Jedním z volitelných parametrů, které můžeme použít, je `[trajectory_at, x0, y0]`, díky kterému `plotdf` současně zobrazí červenou křivkou partikulární řešení dané diferenciální rovnice vyhovující počáteční podmínce $y(x_0) = y_0$. Dále můžeme pojmenovat osy volbami `[xaxislabel, "text"]` a `[yaxislabel, "text"]`. Implicitně jsou pojmenovány podle nezávislé a závislé proměnné.

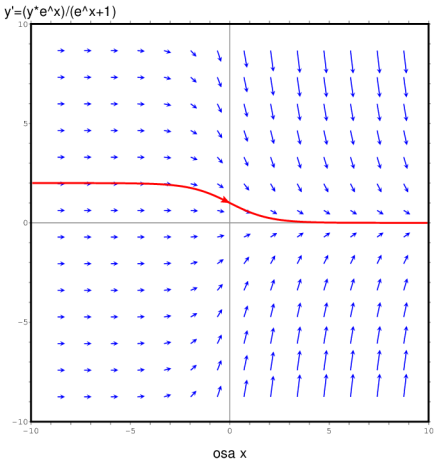
Příklad 1.2.1. Zobrazíme řešení rovnice se separovanými proměnnými $(1+\mathrm{e}^x)\frac{y'}{y} + \mathrm{e}^x = 0$ a zvýrazníme partikulární řešení pro $y(0) = 1$.

```
(%i3) depends(y, x)$
      DR: (1+exp(x))*diff(y,x)/y + exp(x) = 0$
      DR2: solve(DR, diff(y,x))[1];
(DR2)
```

$$\frac{d}{dx}y = -\frac{\%e^x y}{\%e^x + 1}$$

```
(%i4) plotdf(rhs(DR2), [trajectory_at,0,1],
        [xaxislabel, "osa x"], [yaxislabel, "y'=(y*e^x)/(e^x+1)"])$
```

(viz obrázek 1.2)



Obrázek 1.2: Obecné řešení rovnice $(1+\mathrm{e}^x)\frac{y'}{y}+\mathrm{e}^x=0$ se zvýrazněním řešení pro počáteční podmínku $y(0)=1$.

Homogenní rovnice

Výsledný graf příkazu `plotdf` se zobrazí v novém okně spolu s grafickým rozhraním, ve kterém můžeme měnit vzhled a parametry zobrazovaného grafu. K dispozici máme základní nástroje jako zavřít okno, editace parametrů zobrazovaného grafu, opětovné vykreslení, export do PostScriptu, oddálení, přiblížení, vykreslení grafu parametricky zadaných křivek $x(t)$, $y(t)$ a nápověda. Do samotného grafu je možné kliknutím přidat další křivky partikulárních řešení procházející bodem, ve kterém jsme kurzorem klikli. Pohybem myši se současně se stisklým pravým tlačítkem grafem posunujeme.

Mezi další parametry patří zadání rozsahů proměnných ve tvaru `[x, xMin, xMax]` a `[y, yMin, yMax]`, oba s implicitními hodnotami od -10 do 10. Parametrem `[xfun, "funkce"]` do grafu vykreslíme další funkci nebo více funkcí proměnné x. Za `funkce` dosadíme řetězec funkcí oddělených středníkem.

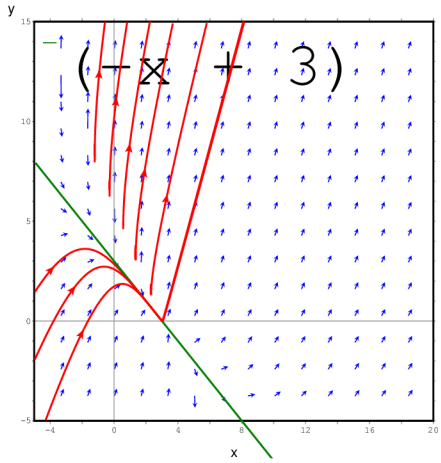
Příklad 1.2.2. Zobrazíme řešení homogenní rovnice $y'=\frac{3x+4y-9}{2x+y-6}$, myší zobrazíme několik partikulárních řešení a necháme vykreslit i funkci $y=-x+3$.

```
(%i2) depends(y, x)$
      DR: diff(y,x) = (3*x+4*y-9)/(2*x+y-6);
(DR)
```

$$\frac{d}{dx}y=\frac{-9+3x+4y}{y+2x-6}$$

```
(%i3) plotdf(rhs(DR), [x,-5,20], [y,-5,15], [xfun,"-x+3"])$
```

(viz obrázek 1.3)



Obrázek 1.3: Obecné řešení rovnice $y' = \frac{3x+4y-9}{2x+y-6}$ s několika partikulárními řešeními. Vidíme, že legenda k funkcím vloženým parametrem `xfun` se do PostScriptu neexportuje optimálně.

Lineární rovnice 1. řádu

Příklad 1.2.3. Necháme zobrazit řešení lineární rovnice $(\mathrm{e}^y - x)y' = y$, procházející bodem $[\mathrm{e}, 1]$.

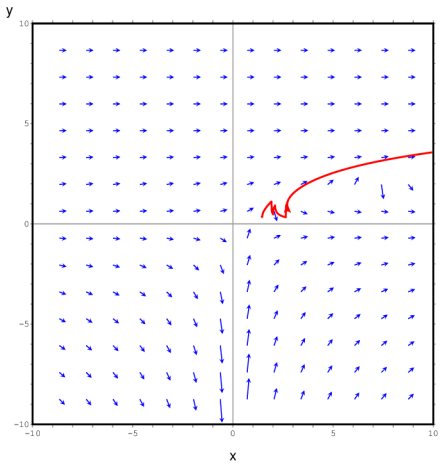
Pro Eulerovo číslo použijeme přibližnou hodnotu 2,718, protože `plotdf` neakceptuje konstanty jako například `%e`.

```
(%i3) depends(y, x)$
      DR: (exp(y)-x)*diff(y,x) = y$
      DR2: solve(DR, diff(y,x))[1];
(DR2)
```

$$\frac{d}{dx}y = \frac{y}{\%e^y - x}$$

```
(%i4) plotdf(rhs(DR2), [trajectory_at, 2.718, 1])$
```

(viz obrázek 1.4)

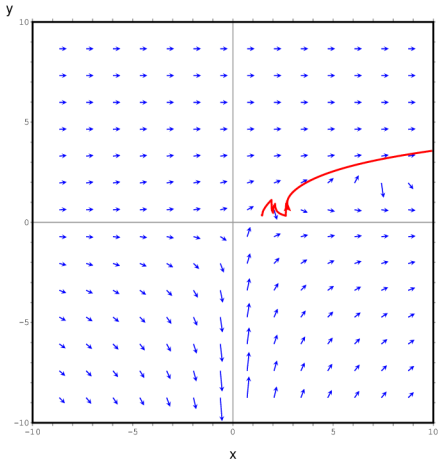


Obrázek 1.4: Řešení $(e^y - x)y' = y$ procházející bodem $[e, 1]$.

Graf zobrazující řešení v tomto bodě nevypadá správně. Upravíme krok, o jaký se zvětšuje proměnná x při integraci. K tomu slouží parametr `[tstep, krok]`, kde za `krok` dosadíme požadované reálné číslo. Změníme implicitní hodnotu 0,1 na 0,001.

```
(%i5) plotdf(rhs(DR2), [trajectory_at, 2.718, 1], [tstep, 0.001])$
```

(viz obrázek 1.5)



Obrázek 1.5: Řešení $(e^y - x)y' = y$ procházející bodem $[e, 1]$. Změněn parametr `tstep`.

Bohužel nám v tomto případě změna hodnoty `tstep` nepomohla a řešení vypadá totožně.

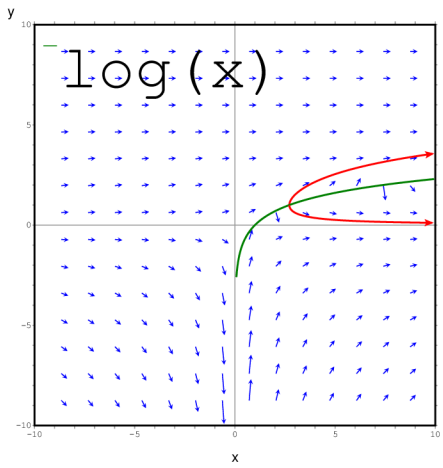
Ze zadání je vidět, že aby se jednalo o diferenciální rovnici musí platit $e^y \neq x$, tedy $y \neq \ln(x)$. Zobrazíme i tuto funkci a nastavíme, aby křivka partikulárního řešení pokračovala od zadaného bodu pouze jedním směrem a ne oběma, jak je nastaveno implicitně. Pomocí parametru `[direction, forward]` zadáme, aby se numerická integrace počítala pouze ve

směru rostoucího x (další možné hodnoty jsou `backward` a implicitní `both`). V opačném směru se totiž `plotdf` přiblíží rovnici $y = \ln(x)$ a pokračuje ve směru křivky jiného partikulárního řešení, což se následně stane ještě v několika bodech a výsledná křivka vypadá „zubatě“.

Nakonec ještě přidáme další křivku, vycházející z bodu těsně pod výchozím bodem první křivky. Abychom bod, ze kterého bude vycházet, zvolili přesně, a ne jen přibližně jako při kliknutí myší do grafu, použijeme následující postup: V okně s grafem, které se po zavolání `plotdf` otevře, zvolíme nástroj editace parametrů, přepíšeme hodnotu parametru `Trajectory at` na hodnotu `2.718 0.99` a potvrdíme Enterem. Tak se přidá další křivka partikulárního řešení a předchozí křivky zůstanou také vykresleny. Naopak nástrojem opětovného vykreslení grafu se zobrazí vždy jen poslední zadaná křivka.

```
(%i6) plotdf(rhs(DR2), [trajectory_at, 2.718, 1], [xfun, "log(x)"],
        [direction, forward])$
```

(viz obrázek 1.6)



Obrázek 1.6: Řešení $(e^y - x)y' = y$ procházející bodem $[e, 1]$. Integrace pouze ve směru zvětšujícího se x, vykreslena i funkce $y = \ln(x)$ a ručně přidána druhá část křivky.

Bernoulliova rovnice

Vhodným volitelným parametrem ke zkoumání vlivu určité konstanty v rovnici na výsledné řešení je `[sliders, hodnoty]`, kde za řetězec `hodnoty` obalený uvozovkami dosadíme jednotlivé konstanty, spolu s jejich rozsahem, oddělené čárkami. Syntaxe pro jednu konstantu je `k=kMin:kMax`. Abychom k mohli použít v rovnici, kterou předáváme `plotdf` k vykreslení, zadefinujeme ji spolu s její výchozí hodnotu k_0 pomocí `[parameters, "k=k0"]` (v případě více konstant bychom je opět oddělili čárkami). Díky použitému parametru `sliders` se pak pod vykresleným grafem zobrazí posuvník, kterým budeme moct interaktivně měnit hodnotu k. Použití `parameters` je vhodné i bez `sliders`, pokud máme určitou konstantu ve více částech rovnice, chceme její hodnotu jednoduše změnit a znovu pomocí `plotdf` vykreslit.

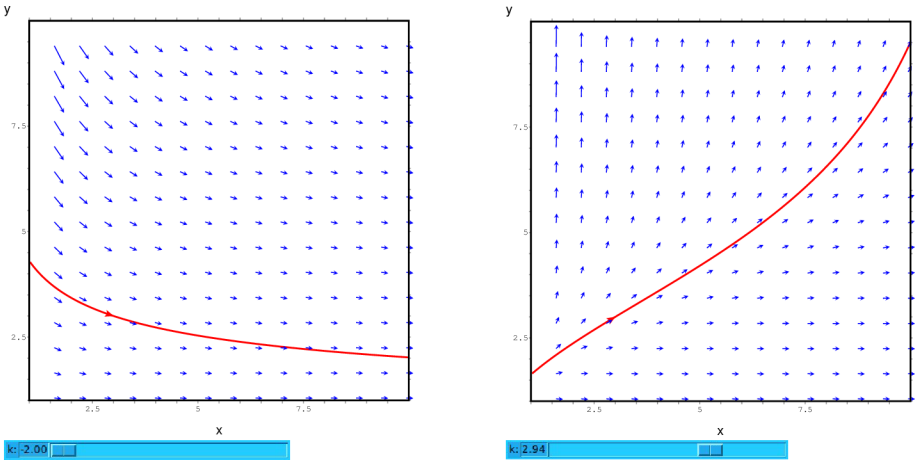
Příklad 1.2.4. Zobrazíme řešení Bernoulliovy rovnice $3x^2y' + xy = y^k$, kde stupeň k bude možné měnit v intervalu $[-2,5]$, procházející bodem $[3,3]$.

```
(%i3) depends(y, x)$
      DR: 3*x^2*diff(y,x) + x*y = y^k$
      DR2: solve(DR, diff(y,x))[1];
(DR2)
```

$$\frac{d}{dx}y = \frac{y^k - xy}{3x^2}$$

```
(%i4) plotdf(rhs(DR2), [x, 1, 10], [y, 1, 10], [trajectory_at, 3, 3],
        [parameters, "k=-2"], [sliders, "k=-2:5"])$
```

(viz obrázek 1.7)



Obrázek 1.7: Řešení $3x^2y' + xy = y^k$ procházející bodem $[3,3]$ pro $k = -2$ a $k = 2{,}94$.

Exaktní rovnice

V nástroji editace parametrů máme možnost ovlivnit kromě jiného i nastavení barev. Barvu křivek partikulárních řešení je možné nastavit parametrem `fieldlines`, barvu šipek směrového pole pomocí `vectors`. Křivkám partikulárních řešení můžeme také v `linewidth` změnit tloušťku z implicitní hodnoty 2.

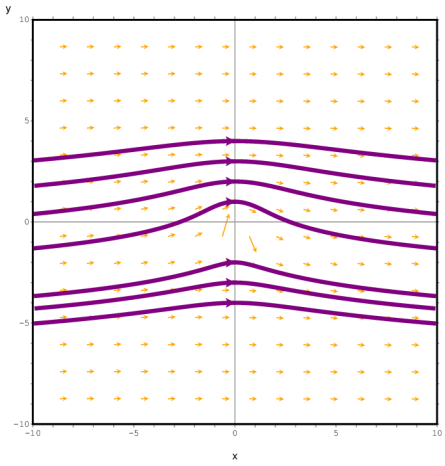
Příklad 1.2.5. Zobrazíme řešení exaktní rovnice $y' = \frac{-x}{x^2+y^2+y}$.

```
(%i2) depends(y, x)$
      DR: diff(y,x) = -x/(x^2 + y^2 + y);
(DR)
```

$$\frac{d}{dx}y = -\frac{x}{y^2 + y + x^2}$$

```
(%i3) plotdf(rhs(DR))$
```

(viz obrázek 1.8)



Obrázek 1.8: Řešení $y' = \frac{-x}{x^2+y^2+y}$ se změnou barev a tučnějšími křivkami partikulárních řešení.

Rovnice mimo základní typy

Příkaz `plotdf` můžeme použít i na rovnice, které nepatří mezi žádné ze základních typů řešených výše. Pokud v Maximě nenajdeme přesné řešení například pomocí `ode2`, ale rovnice je rozřešitelná vzhledem k derivaci, její řešení si můžeme alespoň graficky znázornit pomocí `plotdf`, například jako rovnici v následujícím příkladu.

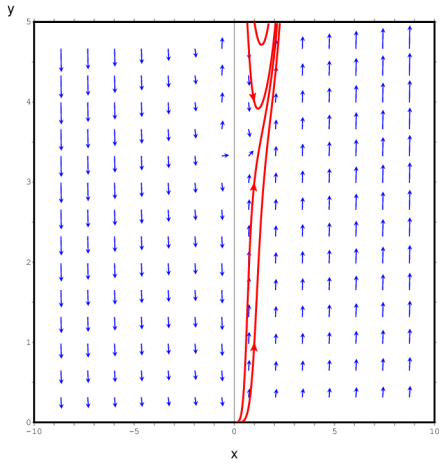
Příklad 1.2.6. Vykreslíme řešení obecné diferenciální rovnice prvního řádu $y' = \frac{4}{x}\sin(y) + x\sqrt{y}$. Protože obsahuje odmocninu $\sqrt{y}$, bude potřeba změnit rozsah osy pro y.

```
(%i2) depends(y, x)$
      DR: diff(y,x) = 4/x*sin(y) + x*sqrt(y);
(DR)
```

$$\frac{d}{dx}y = \frac{4\sin(y)}{x} + x\sqrt{y}$$

```
(%i3) plotdf(rhs(DR), [y, 0, 5])$
```

(viz obrázek 1.9)



Obrázek 1.9: Řešení obecné diferenciální rovnice rovnice $y' = \frac{4}{x}\sin(y) + x\sqrt{y}$ s několika partikulárními řešeními.

1.2.2 Příkaz `drawdf`

Nové možnosti oproti `plotdf` přináší příkaz `drawdf`, respektive stejnojmenný balíček, který před použitím musíme načíst přes `load(drawdf)`. Vychází z balíčku `draw` a rovněž používá k vykreslení program `gnuplot`. Kromě vykreslení směrového pole vektorů znázorňujícího obecné řešení diferenciální rovnice, umožňuje přidat do výsledného grafu i další objekty z `draw` a jeho příkazu `draw2d`, a tak jej doplnit o širokou škálu grafických objektů a upřesňujících voleb. Podobně `drawdf` podporuje i některé parametry z `plotdf`.

Stejné jsou u `drawdf`, v porovnání s `plotdf`, i požadavky na tvar rovnice, který příkaz dokáže vykreslit, a následná syntaxe. Opět můžeme nechat vykreslit obecné řešení obyčejnou diferenciální rovnici prvního řádu tvaru $y' = f(x,y)$ příkazy
`drawdf(f(x,y))`
nebo
`drawdf(f(t,u), [t,u])`,
pokud funkce f nezávisí na proměnných x a y – v tomto případě je možné spojit definici těchto proměnných se zadáním jejich rozsahu následně
`drawdf(f(t,u), [t,tMin,tMax], [u,uMin,uMax])`.

Implicitně příkaz `drawdf` vykresluje grafiku pomocí programu `gnuplot` do nového okna. Jednou z dalších možností je vykreslit graf přímo do pracovního okna Maximy přidáním předpony „wx“, tedy `wxdrawdf`.

Rovnice se separovanými proměnnými

Jedním z grafických objektů, který `drawdf` přináší navíc oproti `draw` je `soln_at(x0,y0)`, přidávající do grafu partikulární řešení pro počáteční úlohu $y(x_0) = y_0$, respektive jeho varianta pro současné zobrazení více partikulárních řešení `solns_at([x1,y1], [x2,y2], ..., [xN,yN])`.

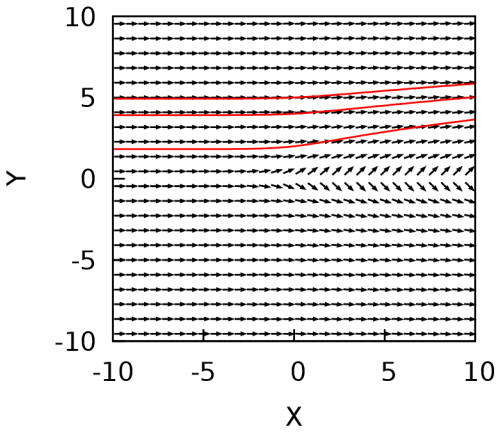
Příklad 1.2.7. Vykreslíme řešení rovnice se separovanými proměnnými $2(1+\mathrm{e}^x)yy' = \mathrm{e}^x$ a několik partikulárních řešení.

```
(%i4) load(drawdf)$
      depends(y, x)$
      DR: 2*(1+exp(x))*y*diff(y,x)=exp(x)$
      DR2: solve(DR, diff(y,x))[1];
(DR2)
```

$$\frac{d}{dx}y = \frac{\%e^x}{(2\%e^x+2)\,y}$$

```
(%i5) drawdf(rhs(DR2), solns_at([0,2],[0,4],[0,5]))$
```

(viz obrázek 1.10)



Obrázek 1.10: Řešení rovnice $2(1+\mathrm{e}^x)yy' = \mathrm{e}^x$.

Okno programu `gnuplot` ve 2D nabízí několik interaktivních prvků:

- Stiskem `Ctrl` a otáčením kolečkem myši nahoru/dolů graf přiblížíme či oddálíme do/od pozice kurzoru. Stejného efektu docílíme pomocí kláves `+` a `-`. Alternativně kliknutím pravým tlačítkem, výběrem obdélníkové oblasti a potvrzení druhým kliknutím libovolným tlačítkem dojde k přiblížení vybrané oblasti. Klávesou `u` (z anglického unzoom) zrušíme veškeré přiblížení/oddálení.

- Posouvání grafu pak docílíme otáčením kolečka myši – implicitně dojde k vertikálnímu posouvání, se stisknutou klávesou `Shift` k horizontálnímu. Případně veškeré posouvání můžeme realizovat klávesami `Nahoru`, `Dolů`, `Doleva` a `Doprava`.
- Pomocí klávesy `7` přepínáme poměry stran grafu. Buď jsou značky na osách v poměru $1:1$, nebo je graf vykreslen do čtverce, nebo je graf roztažen do celého okna `gnuplot`.
- Klávesa `l` přepíná zobrazení hodnot na ose y do logaritmického (logarithmic) měřítka a zpět.
- Tečkovanou mřížku (grid) můžeme nechat zobrazit klávesou `g`.
- Klávesa `r` zafixuje v místě kurzoru pravoúhlé pravítko (ruler). Ve spodní liště okna `gnuplot` pak vidíme kromě aktuálních souřadnic kurzoru $[k_x, k_y]$ i souřadnice středu pravítka $[s_x, s_y]$ a x-ovou $(k_x - s_x)$ i y-ovou $(k_y - s_y)$ vzdálenost kurzoru od něj. Stiskem klávesy `5` se pak aktuální pozice kurzoru spojí úsečkou se středem pravítka a ve spodní liště se zobrazí i Euklidovská vzdálenost kurzoru od středu pravítka daná $\sqrt{(k_x-s_x)^2+(k_y-s_y)^2}$ a sklon jejich spojnice – buď jako úhel α ve stupních v rozsahu $[-180, 180)$ nebo jako tangens tohoto úhlu $\operatorname{tg}\alpha = \frac{k_y-s_y}{k_x-s_x} \in (-\infty, \infty)$.
- Kliknutím prostředním tlačítkem myši v místě kurzoru zaneseme nový bod se svými souřadnicemi.
- Klávesou `e` provedeme opětovné vykreslení celého grafu.
- Klávesa `q` zavírá (quit) okno `gnuplot` s grafem.

Homogenní rovnice

Dalším novým grafickým objektem, který v `drawdf` můžeme použít, je `point_at(x0,y0)`. Zanáší do grafu bod o souřadnicích $[x_0, y_0]$ a existuje i jeho varianta pro více bodů `points_at([x1,y1], [x2,y2], ..., [xN,yN])`. Volba `field_color=barva` nastavuje barvu šipek směrového pole.

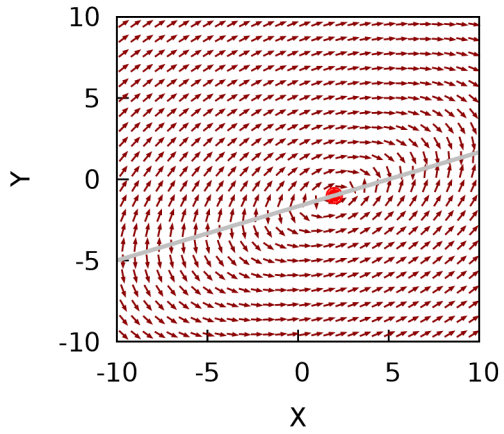
Příklad 1.2.8. Zobrazíme řešení homogenní rovnice $y' = \frac{2x-y-5}{x-3y-5}$.

```
(%i3) load(drawdf)$
      depends(y, x)$
      DR: diff(y,x) = (2*x-y-5)/(x-3*y-5);
(DR)
```

$$\frac{d}{dx}y = \frac{-y+2x-5}{-3y+x-5}$$

```
(%i4) drawdf(rhs(DR), field_color=dark-red,
        point_size=2, point_at(2,-1),
        color=grey, line_width=2.5, explicit((x-5)/3,x,-10,10))$
```

(viz obrázek 1.11)



Obrázek 1.11: Obecné řešení rovnice $y' = \frac{2x-y-5}{x-3y-5}$, asymptotická přímka $y = \frac{x-5}{3}$ a singulární bod $[2,-1]$.

Při vykreslení jsme z `draw` využili tyto objekty a volby: `point_size=velikost`, `color=barva`, `line_width=tloušťka` a `explicit(f(x), x, xMin, xMax)`. Jak je vidět, vyhodnocení vstupu probíhá sekvenčně, takže například změna barvy se aplikuje až na přímku, nikoliv na bod. Podobně se samotné vykreslování objektů realizuje v pořadí, v jakém byly zadány, takže přímka je vykreslena přes bod.

Lineární rovnice 1. řádu

Mezi další nové parametry v `drawdf` patří `field_degree=hodnota`, upřesňující vzhled směrového pole. Jeho hodnota může být buď implicitní `1`, kdy je směrové pole tvořeno vektory ve tvaru úseček, nebo `2` (pole kvadratických splajnů), nebo `'solns`, kdy je každý prvek pole malou křivkou partikulárního řešení spočtenou pomocí Rungovy-Kuttovy metody 4. řádu (viz kapitolu 1.2.3).

Pomocí `field_grid=[sloupce,řádky]` nastavíme počet sloupců a řádků mřížky směrového pole, tj. počet prvků pole v horizontálním a vertikálním směru. Implicitní nastavení je `field_grid=[30,22]`.

Volbou `soln_arrows=true` můžeme povolit zobrazení šipek na křivkách partikulárních řešení. Implicitně se pak prohodí barvy, tedy směrové pole bude červené a křivky partikulárních řešení černé. Podobně volbou `field_arrows=false` můžeme zakázat zobrazení šipek u vektorů v poli.

Jak je vidět v příkladu níže, `drawdf` na rozdíl od `plotdf` akceptuje i známé konstanty jako například `%pi`.

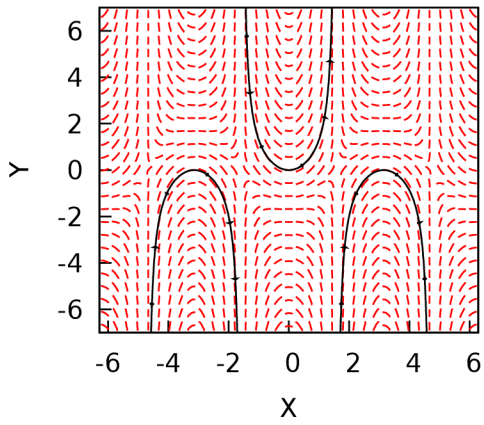
Příklad 1.2.9. Vykreslíme obecné řešení lineární rovnice $y' \cos x = (y + 2\cos x)\sin x$.

```
(%i4) load(drawdf)$
      depends(y, x)$
      DR: diff(y,x)*cos(x) = (y+2*cos(x))*sin(x)$
      DR2: solve(DR, diff(y,x))[1];
(DR2)
```

$$\frac{d}{dx}y = \frac{\sin(x)y + 2\cos(x)\sin(x)}{\cos(x)}$$

```
(%i5) drawdf(rhs(DR2), [x,-2*%pi,2*%pi], [y,-7,7],
        field_degree='solns, field_grid=[35,25],
        soln_arrows=true, solns_at([-%pi,0],[0,0],[%pi,0]))$
```

(viz obrázek 1.12)



Obrázek 1.12: Řešení lineární rovnice $y' \cos x = (y + 2\cos x)\sin x$. Obecné řešení tvořeno malými křivkami partikulárních řešení.

Bernoulliova rovnice

Dalšími parametry ovlivňujícími křivky partikulárních řešení, které `drawdf` nabízí, jsou `tstep=krok` a `nsteps=n`, kde `krok` definuje maximální délku kroku, která se použije při numerické integraci (implicitně 0,1), a `n` počet těchto kroků. Podobně jako u `plotdf` můžeme také zadat, zda se má křivka vykreslit ve směru rostoucí či klesající nezávislé proměnné. Toto nastavuje volba `direction=směr`, kde za `směr` dosadíme `left`, `right`, nebo implicitní `both`. Všechny tyto volby jsou opět vyhodnocovány sekvenčně, takže aby jejich použití mělo efekt, musí předcházet objektům `soln_at`, `solns_at` či `trajectory_at`.

V příkladu níže nastavíme menší maximální krok integrace. Toto nastavení umožňuje přibližovat graf do větší úrovně detailu při stále hladkém vzhledu křivek (díky tomuto nastavení bylo možné relativně přesně určit body, ve kterých je $y' = 0$, viz obrázek 1.13).

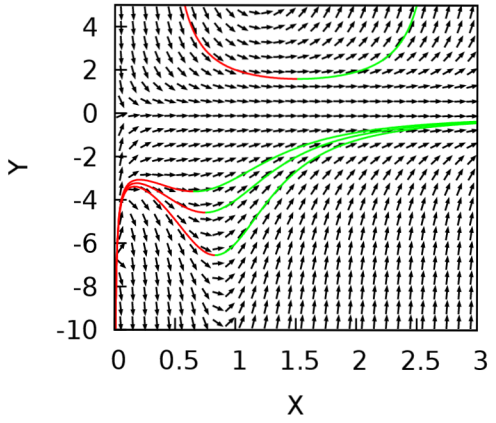
Příklad 1.2.10. Zobrazíme řešení Bernoulliovy rovnice $xy' + y = y^2 x \ln x$.

```
(%i4) load(drawdf)$
      depends(y, x)$
      DR: x*diff(y,x) + y = y^2*x*log(x)$
      DR2: solve(DR, diff(y,x))[1];
(DR2)
```

$$\frac{d}{dx}y = \frac{x\log(x)y^2 - y}{x}$$

```
(%i5) drawdf(rhs(DR2), [x,0,3], [y,-10,5], tstep=0.01,
        direction=left,
        solns_at([0.83,-6.55],[0.75,-4.58],[0.65,-3.61], [1.51,1.6]),
        color=green, direction=right,
        solns_at([0.83,-6.55],[0.75,-4.58],[0.65,-3.61],[1.51,1.6]))$
```

(viz obrázek 1.13)



Obrázek 1.13: Obecné řešení Bernoulliovy rovnice $xy' + y = y^2 x \ln x$.

Exaktní rovnice

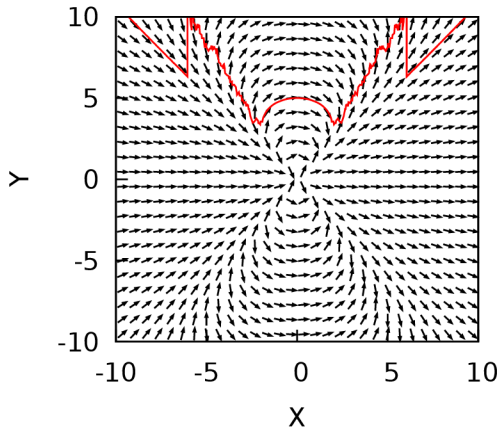
Příklad 1.2.11. Vykreslíme obecné řešení exaktní rovnice $y' = -\frac{2xy}{y^2-3x^2}$ a partikulární řešení vyhovující $y(0) = 5$.

```
(%i3) load(drawdf)$
      depends(y, x)$
      DR: diff(y,x) = -(2*x*y)/(y^2-3*x^2);
(DR)
```

$$\frac{d}{dx}y = -\frac{2xy}{y^2 - 3x^2}$$

```
(%i4) drawdf(rhs(DR), solns_at([0,5]))$
```

(viz obrázek 1.14)



Obrázek 1.14: Řešení exaktní rovnice $y' = -\frac{2xy}{y^2-3x^2}$ pro počáteční podmínku $y(0) = 5$.

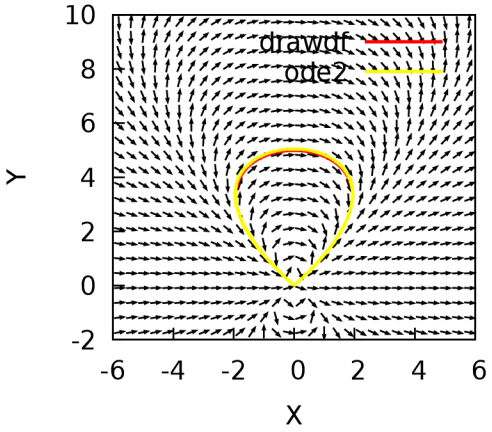
Je vidět, že toto řešení pravděpodobně nebude správné. Křivka v bodech, ve kterých by se měla stáčet směrem dolů, vždy přejde do křivky jiného partikulárního řešení a ve výsledku je „zubatá“. Parametry `tstep` a `nsteps` můžeme nastavit, aby byl krok numerické integrace menší a aby těchto kroků bylo jen tolik, aby křivka nezačala přecházet do křivek jiných řešení. Do výsledného grafu přidáme graf přesného řešení, jak jej nalezne například příkaz `ode2` v kombinaci s `ic1` (řešení nepatrně posuneme, aby nepřekrývalo křivku spočtenou `drawdf`).

```
(%i6) ode2(DR, y, x)$
      PR: ic1(%, x=0, y=5);
(PR)
```

$$-\frac{y^2-x^2}{y^3} = -\frac{1}{5}$$

```
(%i7) drawdf(rhs(DR), [x,-6,6], [y,-2,10], line_width=2,
        nsteps=380, tstep=0.005, key="drawdf", soln_at(0,5),
        color=yellow, ip_grid=[70,70], key="ode2",
        implicit(rhs(PR)*99/100=lhs(PR),x,-2.5,2.5,y,0,5.2))$
```

(viz obrázek 1.15)



Obrázek 1.15: Řešení exaktní rovnice $y' = -\frac{2xy}{y^2-3x^2}$ pro počáteční podmínku $y(0) = 5$. Červeně část řešení pomocí `drawdf`, žlutě nepatrně posunuté přesné řešení.

K vykreslení řešení v implicitní formě jsme použili objekt z `draw` `implicit(rovnice, x, xMin, xMax, y, yMin, yMax)` v kombinaci s parametrem `ip_grid=[početX,početY]`, který nastaví počet referenčních bodů pro jeho vykreslení na obou osách. Obě křivky jsou pojmenovány v legendě volbou `key="popis"`.

Rovnice mimo základní typy

Příkaz `drawdf` můžeme podobně jako `plotdf` vhodně použít i na grafické znázornění numericky spočteného řešení diferenciální rovnice mimo základní známé typy. V příkladu níže toto řešení doplníme následujícími volbami z balíčku `draw`: `title="název"`, `xlabel="text"` a `ylabel="text"` zadávajícími nadpis grafu a popis os.

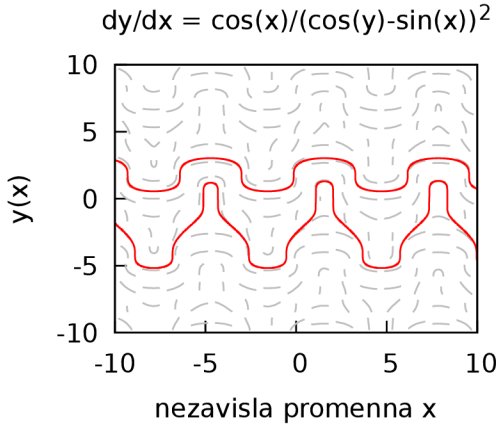
Příklad 1.2.12. Zobrazíme obecné řešení diferenciální rovnice $y' = \frac{\cos x}{(\cos y - \sin x)^2}$.

```
(%i3) load(drawdf)$
      depends(y, x)$
      DR: diff(y,x) = cos(x)/(cos(y)-sin(x))^2;
(DR)
```

$$\frac{d}{dx}y = \frac{\cos(x)}{(\cos(y) - \sin(x))^2}$$

```
(%i4) drawdf(rhs(DR), field_grid=[15,15], field_arrows=false,
        field_color=gray, field_degree='solns,
        tstep=0.005, nsteps=7000, solns_at([2,3],[2,1]),
        title="dy/dx = cos(x)/(cos(y)-sin(x))^2",
        xlabel="nezavisla promenna x", ylabel="y(x)")$
```

(viz obrázek 1.16)



Obrázek 1.16: Obecné řešení rovnice $y' = \frac{\cos x}{(\cos y - \sin x)^2}$ a dvě partikulární řešení.

1.2.3 Příkaz `rk`

V případě, že nám nestačí vykreslit numericky získané řešení diferenciální rovnice, ale chceme získat i body, kterými prochází, můžeme použít příkaz `rk`, který vrátí seznam souřadnic bodů, spočtených Rungovou-Kuttovou metodou 4. řádu (viz níže). Syntaxe pro jednu diferenciální rovnici prvního řádu $y' = f(x,y)$ je
`rk(f(x,y), y, y(x0), [x, x0, xN, h])`,
kde `y` je závislá proměnná, `y(x0)` její počáteční hodnota, `x` nezávislá proměnná, `x0` a `xN` dolní a hodní hranice intervalu a `h` krok mezi jednotlivými uzly, ve kterých se bude numericky počítat řešení.

Příkaz `rk` vrací seznam, jehož každý prvek obsahuje 2 souřadnice: uzel a spočtenou hodnotu závislé proměnné v tomto uzlu. Délka seznamu je rovna nejvýše $N+1$ (tedy počtu uzlů), kde $N = \frac{x_N - x_0}{h}$. Pokud v průběhu výpočtů závislá proměnná dosáhne příliš velké absolutní hodnoty, výpočet v tomto uzlu skončí a `rk` vrátí seznam o N či méně prvcích.

Rungovy-Kuttovy metody

Podobně jako při řešení Adamsovou-Moultonovou metodou (viz kapitolu 1.2.1) předpokládejme, že řešíme diferenciální rovnici ve tvaru $y'(x) = f(x, y(x))$ pro počáteční podmínku $y(x_0) = y_0$. Řešení budeme hledat na diskrétní síti uzlů x_i, kde $i = 0, \ldots, N$ s krokem h, tj. $x_{i+1} = x_i + h, i = 0, 1, \ldots, N-1$. Známe přesnou hodnotu $y(x_0) = y_0$, v dalších uzlech pak spočteme přibližné hodnoty $y_i \approx y(x_i)$, které získáme numericky Rungovou-Kuttovou metodou.

Obecný tvar s-stupňové Rungovy-Kuttovy metody je

$$y_{i+1} = y_i + h \sum_{j=1}^{s} b_j k_j,$$

kde koeficienty k_j jsou dány vztahem

$$k_j = f(x_i + hc_j, y_i + h\sum_{l=1}^{s} a_{jl}k_l), \quad j = 1,2,\ldots s\ .$$

Reálné konstanty b_j, c_j a a_{jl} pak přesně definují konkrétní Rungovu-Kuttovu metodu. Jedná se tedy obecně o implicitní metody, kdy ke spočtení koeficientu k_j v každém uzlu x_i je potřeba vyřešit soustavu s rovnic o s neznámých $k_1, k_2, \ldots, k_s$.

Zjednodušení nastává u explicitních metod, kdy platí, že $a_{jl} = 0$ pro $l \geq j$, a tak v každém uzlu není potřeba řešit soustavu rovnic, ale pouze přímo počítat koeficienty $k_1, k_2, \ldots, k_s$ v tomto pořadí – tj. každý další koeficient k_i na základě předchozích $k_{i-1}, k_{i-2}, \ldots, k_1$. Nejznámější metodou je explicitní Rungova-Kuttova metoda 4. řádu a 4. stupně. Pro její koeficienty k_j tedy platí

$$k_j = f(x_i + hc_j, y_i + h\sum_{l=1}^{j-1} a_{jl}k_l), \quad j = 1,2,3,4.$$

Tzv. klasická Rungova-Kuttova metoda 4. řádu má například následující koeficienty zapsané Butcherovou tabulkou (levý sloupec představuje odshora dolů c_1 až c_4, dolní řádek zleva doprava b_1 až b_4 a vnitřek tabulky konstanty a_{jl}):

$$\begin{array}{c|cccc} 0 & & & & \\ \frac{1}{2} & \frac{1}{2} & & & \\ \frac{1}{2} & 0 & \frac{1}{2} & & \\ 1 & 0 & 0 & 1 & \\ \hline & \frac{1}{6} & \frac{1}{3} & \frac{1}{3} & \frac{1}{6} \end{array}$$

Neboli $y_{i+1} = y_i + \frac{1}{6}h(k_1 + 2k_2 + 2k_3 + k_4)$,
$k_1 = f(x_i, y_i)$,
$k_2 = f(x_i + \frac{1}{2}h, y_i + \frac{1}{2}hk_1)$,
$k_3 = f(x_i + \frac{1}{2}h, y_i + \frac{1}{2}hk_2)$,
$k_4 = f(x_i + h, y_i + hk_3)$.

Rovnice mimo základní typy

V příkladu níže použijeme příkaz `rk` na diferenciální rovnici prvního řádu, která nebude patřit mezi žádný ze základních typů. Seznam bodů, které `rk` vrátí můžeme rychle zkontrolovat pomocí příkazů `length(seznam)`, `first(seznam)` a `last(seznam)`, které vypíší počet prvků seznamu a jeho první a poslední prvek.

Příklad 1.2.13. Necháme Maximu spočítat souřadnice bodů funkce y, zadané rovnicí $y' = \mathrm{e}^y - x^2y^2$ a počáteční podmínkou $y(1) = 0$.

```
(%i2) depends(y, x)$
      DR: diff(y,x) = exp(y)-x^2*y^2;
(DR)
```

$$\frac{d}{dx}y = \%e^y - x^2\,y^2$$

```
(%i6) body: rk(rhs(DR), y, 0, [x, 1, 5, 0.05])$
      length(body);
      first(body);
      last(body);
(%o4)
```

$$81$$

```
(%o5)
```

$$[1.0, 0.0]$$

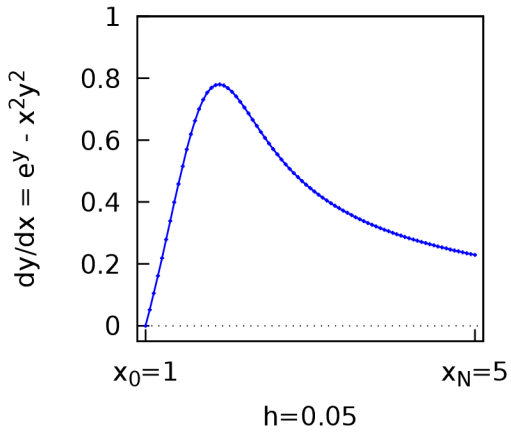
```
(%o6)
```

$$[5.0, 0.2290342806947805]$$

Získaný seznam souřadnic `body` vykreslíme pomocí příkazu `draw2d` a objektu `points([[x1,y1], [x2,y2], ..., [xN,yN]])`. Díky `points_joined=true` nastavíme, aby byly jednotlivé body spojeny úsečkou a `point_size=velikost` nastaví velikost jednotlivých značek. Pomocí
`xtics={["text1",x1],["text1",x2], ..., ["textN",xN]}`
upravíme zobrazované značky na ose x a samotnou osu x do grafu zaneseme volbou `xaxis=true`.

```
(%i8) load(draw)$
      draw2d(points_joined=true, point_size=0.5, points(body),
        xrange=[0.9,5.1], yrange=[-0.05,1],
        xtics={["x_0=1",1],["x_N=5",5]}, xaxis=true, xlabel="h=0.05",
        ylabel="dy/dx = e^y - x^2y^2")$
```

(viz obrázek 1.17)



Obrázek 1.17: Řešení $y' = \mathrm{e}^y - x^2y^2$ pro podmínku $y(1) = 0$.

1.3 Izogonální trajektorie

Závěrem kapitoly popisující možnosti Maximy při řešení diferenciálních rovnic prvního řádu ukážeme jednu z geometrických aplikací těchto rovnic: pro zadaný systém křivek můžeme uvažovat tzv. izogonální trajektorii, tj. nový systém křivek, kde každá z nich protíná všechny zadané křivky pod stejným úhlem.

Pro spočtení izogonální trajektorie protínající zadaný systém křivek

$$F(x,y,c) = 0$$

pod úhlem α začneme s určením diferenciální rovnice tohoto systému. Z věty o derivaci implicitní funkce máme rovnici

$$\frac{\partial F}{\partial x} + \frac{\partial F}{\partial y} y' = 0,$$

do které dosadíme za parametr c vyjádřený z první rovnice. Získáme diferenciální rovnici tvaru

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

- Pro $\alpha \neq 90°$
  udává izogonální trajektorii diferenciální rovnice

$$\operatorname{tg}\alpha = \frac{y' - f(x,y)}{1 + y' f(x,y)}.$$

- Pro $\alpha = 90°$
  udává izogonální trajektorii (v tomto případě nazývanou „ortogonální“) diferenciální rovnice

$$-\frac{1}{y'} = f(x,y),$$

  neboli přesně případ, kdy není zlomek v předchozí obecnější rovnici definován a platí $1 + y' f(x,y) = 0$.

Nakonec získanou diferenciální rovnici trajektorie vyřešíme.

Pro spočtení izogonální trajektorie byla v rámci přílohy této práce napsána procedura, která výše popsanou diferenciální rovnici trajektorie hledá a řeší. Balíček funkcí `ode_mm` (ve kterém je i tato procedura uložena a jehož procedury se zabývají diferenciálními rovnicemi nerozřešenými vzhledem k derivaci) je blíže popsán v kapitole 3.

Pomocí příkazu

```
isogonal(rce, y, x, c, uhel)
```

spočteme izogonální trajektorii daného systému křivek zadaného rovnicí `rce`, kde křivky trajektorie budou protínat původní křivky pod zvoleným úhlem (`uhel`, zadáván v radiánech, ideálně tedy pomocí zlomků konstanty `%pi`). Proměnné y a x jsou závislá a nezávislá proměnná zadaného systému a c je konstanta odlišující jeho jednotlivé křivky.

Procedura se kromě určení diferenciální rovnice, popisující izogonální trajektorii, pokusí tuto rovnici i vyřešit. V případě, že ji nevyřeší, vrátí `false`. Vždy ale uloží do proměnné `traj_ode` zmíněnou diferenciální rovnici, se kterou tak můžeme dále pracovat.

Příklad 1.3.1. Najdeme a vykreslíme ortogonální trajektorii k systému křivek zadanému rovnicí $x^2+y^2=cx$.

Ruční postup by mohl být následující.

```
(%i2) RCE: x^2 + y^2 = c*x$
      F: lhs(RCE) - rhs(RCE);
(F)
```

$$y^2+x^2-cx$$

```
(%i5) Fx: diff(F, x)$
      Fy: diff(F, y)$
      DR: Fx + Fy*'diff(y,x) = 0;
(DR)
```

$$2y\left(\frac{d}{dx}y\right)+2x-c=0$$

Nyní už po dosazení c z rovnice v zadání získáme diferenciální rovnici zadaného systému křivek.

```
(%i7) solve(RCE, c)[1]$
      subst(rhs(%), lhs(%), DR);
(%o7)
```

$$2y\left(\frac{d}{dx}y\right)-\frac{y^2+x^2}{x}+2x=0$$

Substitucí $-\frac{1}{y'}$ za y' dostaneme diferenciální rovnici hledané ortogonální trajektorie, kterou vyřešíme pomocí `ode2`.

```
(%i10) subst(-1/'diff(y,x), 'diff(y,x), %)$
       solve(%, 'diff(y,x))[1];
       ode2(%, y, x);
(%o9)
```

$$\frac{d}{dx}y=-\frac{2xy}{y^2-x^2}$$

```
(%o10)
```

$$-\frac{y}{y^2+x^2}=\%c$$

Příkaz z balíčku `ode_mm` vrátí totožný výsledek.

```
(%i13) load("/home/mik/ode_mm.mac")$
       TRAJ: isogonal(RCE, y, x, c, %pi/2);
       traj_ode;
(TRAJ)
```

$$-\frac{y}{y^2+x^2} = \%c$$

(%o13)

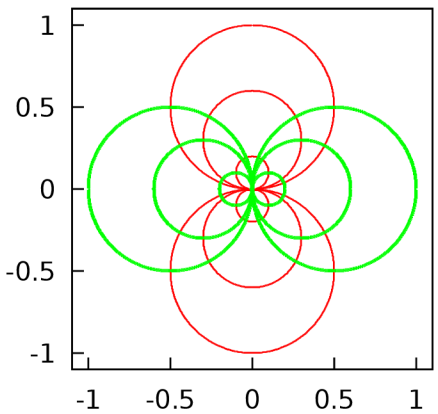
$$\frac{d}{dx}y = -\frac{2xy}{y^2-x^2}$$

Zadané křivky i jejich spočtenou trajektorii si můžeme zobrazit pomocí příkazu `draw2d`. Nejdříve bude vhodné si předchystat konkrétní křivky k vykreslení (níže se jedná o seznamy `RCE_PART` a `TRAJ_PART`). Do každého seznamu uložíme například 6 křivek a za konstanty dosadíme stejné hodnoty (kvůli odlišnému tvaru rovnic zadaného systému a spočtené trajektorie se bude jednat o inverzní zlomky). Tyto seznamy ještě můžeme předchystat jako seznamy objektů typu `implicit`, které `draw2d` vykresluje. Nakonec `draw2d` použijeme, a to se zvýšeným počtem referenčních bodů mřížky (parametr `ip_grid`) a se zachováním správného poměru délek os (`user_preamble="set size ratio 1"`).

```
(%i18) load(draw)$
       RCE_PART: makelist(subst((2*i-7)/5,c,RCE),i,1,6)$
       RCE_GR2D: makelist((implicit(RCE_PART[i],x,-1,1,y,-1,1)),
          i,1,length(RCE_PART))$
       TRAJ_PART: makelist(subst(5/(2*i-7),%c,TRAJ),i,1,6)$
       TRAJ_GR2D: makelist((implicit(TRAJ_PART[i],x,-1,1,y,-1,1)),
          i,1,length(TRAJ_PART))$

(%i19) draw2d(ip_grid=[100,100],
       color=red, TRAJ_GR2D,
       line_width=2, color=green, RCE_GR2D,
       xrange=[-1.1,1.1], yrange=[-1.1,1.1],
       user_preamble="set size ratio 1")$
```

(viz obrázek 1.18)



Obrázek 1.18: Systém křivek $x^2+y^2 = cx$ (zeleně) a jeho ortogonální trajektorie $x^2+y^2 = cy$ (červeně).

1.3.1 Zdrojový kód

Níže je uveden kód pomocné funkce, která hledá izogonální trajektorii zadaného systému křivek:

```
/*
Spocte izogonalni trajektorii dane soustavy krivek. Do pomocne
promenne ulozi diferencialni rovnici, ktera trajektorii urcuje.
- eqn je dana rovnice soustavy krivek
- y je zavisla promenna
- x je nezavisla promenna
- c je konstanta
- ang je uhel v radianech, pod kterym maji trajektorie protinat
  krivky
*/
isogonal(eqn,y,x,c,ang) := block(
     [r,s,sub,res:[]],

     r: lhs(eqn)-rhs(eqn),
     /*nahrazeni novymi promennymi, ve kterych se bude pocitat*/
     r: newVars(r,y,x,c),

     s: diff(r,xxx) + diff(r,yyy)*'diff(yyy,xxx) = 0,
     /*eliminace konstanty v s*/
     sub: solve(r,ppp)[1],
     s: ratsubst(rhs(sub),lhs(sub),s),
     s: solve(s,'diff(yyy,xxx))[1],

     if (is(equal(ang,%pi/2))) then (
          /*DR pokud uhel neni 90 st.*/
          s: ratsubst(-1/'diff(yyy,xxx),'diff(yyy,xxx),s)
     ) else (
          /*DR pokud je uhel 90 st.*/
          s: tan(ang)=('diff(yyy,xxx)-rhs(s))
               /(1+'diff(yyy,xxx)*rhs(s))
     ),

     /*vraceni zpet do puvodnich promennych a zruseni zavislosti*/
     s: solve(s,'diff(yyy,xxx))[1],
     s: oldVars(s,y,x,c),

     /*ulozeni DR do zvlastni promenne*/
     traj_ode: s,

     /*reseni DR trajektorie*/
     res: ode2(s,y,x),
```

```
     if (is(equal(res,false))) then
          res: ode_mm(s,y,x),

     return(res)
)$
```

1.4 Tabulky příkazů

`ode2`:

`ode2(difRce, y, x)`	analytické řešení diferenciální rovnice
`ic1(obecReseni, x=x0, y=y0)`	partikulární řešení dané $y(x_0) = y_0$

`desolve`:

`desolve(difRce, y(x))`	analytické řešení diferenciální rovnice
`atvalue(y(x), x=x0, y0)`	definice počáteční podmínky před použitím `desolve`

`plotdf`:

`plotdf(f(x,y))`	směrové pole vektorů dif. rovnice
`plotdf(f(t,u), [t,u])`	totéž pro jiné proměnné než x a y

`drawdf`:

`drawdf(f(x,y))`	směrové pole vektorů dif. rovnice
`drawdf(f(t,u), [t,u])`	totéž pro jiné proměnné než x a y
`drawdf(f(t,u), [t,tMin,tMax], [u,uMin,uMax])`	totéž pro jiné proměnné než x a y, spojena definice proměnných s jejich rozsahem

`rk`:

`rk(f(x,y), y, y(x0), [x, x0, xN, h])`	body numerického řešení dif. rovnice

Tabulka 1.1: Příkazy pro analytické i numerické řešení diferenciálních rovnic prvního řádu.

`plotdf`:

`[trajectory_at, x0, y0]`	partikulární řešení dané $y(x_0) = y_0$
`[x, xMin, xMax]`	rozmezí vykreslovaného intervalu na ose x
`[y, yMin, yMax]`	rozmezí vykreslovaného intervalu na ose y
`[xaxislabel, "text"]`	popis horizontální osy
`[yaxislabel, "text"]`	popis vertikální osy
`[xfun, "f1(x);...;fN(x)"]`	doplnění dalších funkcí do grafu
`[parameters, "k1=a1,...,kN=aN"]`	definice parametrů funkce a jejich hodnot
`[sliders, "k1=k1Min:k1Max,..., kN=kNMin:kNMax"]`	posuvníky pro změny hodnot parametrů

`drawdf`:

`soln_at(x0,y0)`	partikulární řešení dané $y(x_0) = y_0$
`solns_at([x1,y1], ..., [xN,yN])`	totéž pro více partikulárních řešení
`point_at(x0,y0)`	zobrazení bodu $[x_0, y_0]$
`points_at([x1,y1], ..., [xN,yN])`	totéž pro více bodů
`field_grid=[sloupce,řádky]`	počet referenčních bodů směrového pole
`field_degree=1/2/'solns`	tvar prvků směrového pole
`field_arrows=true/false`	zobrazení šipek prvků v poli
`field_color=barva`	barva prvků v poli
`tstep=h`	maximální krok numerického řešení
`nsteps=n`	počet kroků numerického řešení
`direction=left/right/both`	směr vykreslení křivky partikulár. řešení
`soln_arrows=true/false`	zobrazení šipek na křivce partikulár. řešení

Tabulka 1.2: Grafické objekty a volby příkazů `plotdf` a `drawdf`.

Kapitola 2

Obyčejné diferenciální rovnice druhého řádu

V této kapitole popíšeme příkazy Maximy, které umožňují najít analytické i numerické řešení diferenciálních rovnic druhého řádu. Konkrétně řešení lineární diferenciální rovnice druhého řádu, speciálních typů nelineárních rovnic druhého řádu a numerické řešení rovnic druhého řádu v obecném tvaru

$$F(x,y,y',y'') = 0.$$

2.1 Analytické řešení

2.1.1 Příkaz `ode2`

Analytické řešení obyčejných diferenciálních rovnic druhého řádu umožňuje, podobně jako u rovnic prvního řádu, příkaz `ode2`. Syntaxe zůstává stejná

```
ode2(difRce, y, x)
```

s tím, že řešená diferenciální rovnice `difRce` obsahuje i druhou derivaci závislé proměnné y, zapsanou například jako `diff(y,x,2)`. Opět platí, že pokud `difRce` neobsahuje znak rovnosti, Maxima tento výraz pokládá roven nule.

Poté, co `ode2` určil, že rovnice je druhého řádu, rozhoduje, zda se jedná o lineární rovnici, tedy zda je možné ji vyjádřit ve tvaru $y'' + p(x)y' + q(x)y = r(x)$. Pokud ano, pokusí se vyřešit jí příslušnou homogenní (zkrácenou) rovnici $y'' + p(x)y' + q(x)y = 0$, a to metodami podle jejího typu v tomto pořadí: rovnice s konstantními koeficienty, exaktní rovnice, Eulerova rovnice a Besselova rovnice. Pokud je původní rovnice rovna své homogenní rovnici, tedy $r(x) = 0$, vrátí `ode2` řešení spočtené některou z těchto metod ve tvaru $y = k_1 y_1 + k_2 y_2$, kde $k_1, k_2 \in \mathbb{R}$. Pokud naopak $r(x) \neq 0$, hledá `ode2` metodou variace konstant partikulární řešení, které k výsledku přičte, a vrátí tak řešení tvaru $y = k_1 y_1 + k_2 y_2 + y_p$.

V případě, že zadaná rovnice není lineární, příkaz ověří, zda se jedná o jeden z následujících typů, které dokáže řešit: rovnice, ve které chybí závislá proměnná, nebo rovnice, ve které chybí nezávislá proměnná.

Integrační konstanty jsou v řešení zastoupeny sekvencemi znaků `%k1` a `%k2`. Do proměnné `method` je opět po úspěšném vyřešení rovnice uložen řetězec, který označuje použitou metodu. Novou proměnnou u rovnic druhého řádu je pak proměnná `yp`, do které je

uloženo partikulární řešení nehomogenní rovnice při použití metody variace konstant.

Níže ukážeme řešení jednotlivých typů lineárních homogenních rovnic pomocí `ode2`.

Rovnice s konstantními koeficienty

Řešení lineární homogenní rovnice s konstantními koeficienty

$$y'' + a\,y' + by = 0, \quad a,b \in \mathbb{R},$$

má tvar

$$y = k_1 \mathrm{e}^{\lambda_1 x} + k_2 \mathrm{e}^{\lambda_2 x},$$

kde λ_1 a λ_2 jsou různá reálná řešení charakteristické rovnice $\lambda^2 + a\lambda + b = 0$ a k_1 a k_2 integrační konstanty. Pokud má charakteristická rovnice dvojnásobný reálný kořen λ_1, řešení je tvaru $y = k_1 \mathrm{e}^{\lambda_1 x} + k_2 x \mathrm{e}^{\lambda_1 x}$. Pokud jsou kořeny nereálné komplexně sdružené $\alpha \pm i\beta$, řešení má tvar $y = \mathrm{e}^{\alpha x}(k_1 \cos \beta x + k_2 \sin \beta x)$.

Příklad 2.1.1. Vyřešíme rovnici s konstantními koeficienty $y'' + y' - 2y = 0$.

Ručně bychom mohli postupovat například následně:

```
(%i2) depends(y, x)$
      DR: diff(y,x,2) + diff(y,x) - 2*y = 0;
(DR)
```

$$\frac{d^2}{dx^2}y + \frac{d}{dx}y - 2y = 0$$

```
(%i8) SUB: p^2 = diff(y,x,2)$
      SUB2: p = diff(y,x)$
      SUB3: 1 = y$
      subst(lhs(SUB), rhs(SUB), DR)$
      subst(lhs(SUB2), rhs(SUB2), %)$
      CHAR: subst(lhs(SUB3), rhs(SUB3), %);
(CHAR)
```

$$p^2 + p - 2 = 0$$

```
(%i11) CHAR_RES1: solve(CHAR, p)[1]$
       CHAR_RES2: solve(CHAR, p)[2]$
       OR: y = %k1*exp(rhs(CHAR_RES1)*x) + %k2*exp(rhs(CHAR_RES2)*x);
(OR)
```

$$y = \%k1\,\%e^{x} + \%k2\,\%e^{-2x}$$

Použitím `ode2` na zadanou rovnici přímo dostaneme stejný výsledek.

```
(%i13) ode2(DR, y, x);
       method;
(%o12)
```

$$y = \%k1\,\%e^{x} + \%k2\,\%e^{-2x}$$

```
(%o13)
```

constcoeff

Jednou z možností, jak najít partikulární řešení diferenciální rovnice druhého řádu splňující počáteční podmínku $y(x_0) = y_0$, $y'(x_0) = a$, je použití příkazu `ic2(OR, x=x0, y=y0, 'diff(y,x)=a)`, kde `OR` je obecné řešení. Najdeme partikulární řešení pro podmínku $y(0) = 1$, $y'(0) = -2$. Vyhodnocení příkazu `diff` není v našem případě potřeba potlačit apostrofem, neboť jsme předem definovali závislost $y(x)$ pomocí `depends`.

```
(%i14) PR: ic2(OR, x=0, y=1, diff(y,x)=-2);
(PR)
```

$$y = \%e^{-2x}$$

Ověříme tři podmínky: zda je partikulární řešení řešením zadané rovnice, zda v něm po dosazení $x = 0$ dostaneme $y = 1$ a zda pro něj platí $y'(0) = -2$.

```
(%i19) DR, PR$
       ev(%, nouns);
       PR, x=0;
       rhs(diff(PR,x))$
       %, x=0;
(%o16)
```

$$0 = 0$$

```
(%o17)
```

$$y = 1$$

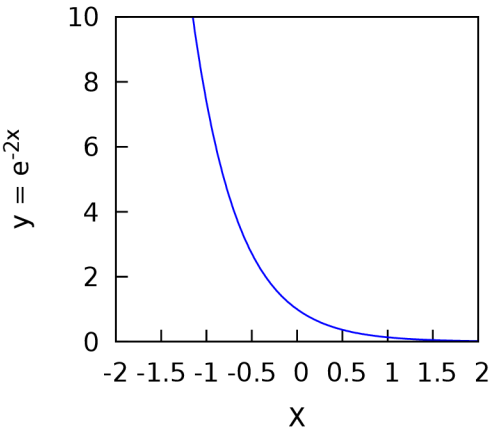
```
(%o19)
```

$$-2$$

Všechny zkoušky prošly a partikulární řešení si můžeme na závěr vykreslit příkazem `draw2d`.

```
(%i21) load(draw)$
       draw2d(explicit(rhs(PR), x, -2, 2), yrange=[0,10],
         ylabel="y = e^{-2x}")$
```

(viz obrázek 2.1)



Obrázek 2.1: $y = \mathrm{e}^{-2x}$ (řešení $y'' + y' - 2y = 0$ pro podmínku $y(0) = 1, y'(0) = -2$)

Exaktní rovnice

Pokud řešíme diferenciální rovnici druhého řádu

$$f(x)y'' + (f'(x) + g(x))y' + g'(x)y = 0,$$

neboli takovou, kterou můžeme upravit do tvaru $(f(x)y')' + (g(x)y)' = 0$, mluvíme o exaktní rovnici. Její integrací na $f(x)y' + g(x)y = k_1$, $k_1 \in \mathbb{R}$ dostaneme lineární rovnici prvního řádu $y' = -\frac{g(x)}{f(x)}y + \frac{k_1}{f(x)}$, kterou jsme řešili v kapitole 1.1.1.

Příklad 2.1.2. Najdeme řešení exaktní rovnice $x^3y'' + 4x^2y' + 2xy = 0$.

Ruční postup by byl následující:

```
(%i2) depends(y, x)$
      DR: x^3*diff(y,x,2) + 4*x^2*diff(y,x) + 2*x*y = 0;
(DR)
```

$$x^3\left(\frac{d^2}{dx^2}y\right) + 4x^2\left(\frac{d}{dx}y\right) + 2xy = 0$$

Ověříme, zda se jedná o exaktní rovnici. Funkci $f(x)$ získáme jako koeficient u y'', funkci $g'(x)$ jako koeficient u y a ověříme, že koeficient u y' je roven $f(x)' + g(x)$. Následně získáme podle vzorce výše lineární rovnici prvního řádu a vyřešíme ji pomocí `ode2`.

```
(%i6) f: coeff(lhs(DR), diff(y,x,2))$
      dgdx: coeff(lhs(DR), y)$
      g: integrate(dgdx,x)$
      is(diff(f,x) + g = coeff(lhs(DR), diff(y,x)));
(%o6)
```

true

```
(%i7) DR1_LIN: diff(y,x) = -g/f*y + %k1/f;
(DR1_LIN)
```

$$\frac{d}{dx}y = \frac{\%k1}{x^3} - \frac{y}{x}$$

```
(%i9) expand(ode2(DR1_LIN, y, x));
      method;
(%o8)
```

$$y = \frac{\%c}{x} - \frac{\%k1}{x^2}$$

```
(%o9)
```

linear

Příkaz `ode2` použitý na zadanou rovnici bez úprav najde stejné řešení.

```
(%i11) OR: ode2(DR, y, x);
       method;
(OR)
```

$$y = \frac{\%k1}{x} + \frac{\%k2}{x^2}$$

```
(%o11)
```

exact

Kromě `ic2` máme ještě druhou možnost, jak spočítat partikulární řešení rovnic druhého řádu. Příkaz `bc2` najde partikulární řešení pro okrajovou podmínku $y(x_1) = y_1$, $y(x_2) = y_2$ a má následující syntaxi: `bc2(OR, x=x1, y=y1, x=x2, y=y2)`. Najdeme partikulární řešení pro podmínku $y(1) = 0$, $y(-1) = 4$, ověříme, že se jedná skutečně o řešení zadané rovnice a že je okrajová podmínka splněna.

```
(%i12) PR: bc2(OR, x=1, y=0, x=-1, y=4);
(PR)
```

$$y = \frac{2}{x^2} - \frac{2}{x}$$

```
(%i16) DR, PR$
       expand(ev(%, nouns));
       PR, x=1;
       PR, x=-1;
(%o14)
```

$$0 = 0$$

```
(%o15)
```

$$y = 0$$

```
(%o16)
```

$$y = 4$$

Eulerova rovnice

Rovnice tvaru $x^2y'' + axy' + by = 0$, $a,b \in \mathbb{R}$ se nazývá Eulerova diferenciální rovnice. Pomocí transformace nezávislé proměnné $t = \ln x$ ji dokážeme převést na rovnici s konstantními koeficienty, neboť

$$y' = \frac{\mathrm{d}y}{\mathrm{d}x} = \frac{\mathrm{d}y}{\mathrm{d}t}\frac{\mathrm{d}t}{\mathrm{d}x} = \frac{\mathrm{d}y}{\mathrm{d}t}\frac{1}{x},$$

$$y'' = \left(\frac{\mathrm{d}y}{\mathrm{d}t}\frac{1}{x}\right)' = \frac{\mathrm{d}^2y}{\mathrm{d}t^2}\frac{1}{x}\frac{1}{x} - \frac{\mathrm{d}y}{\mathrm{d}t}\frac{1}{x^2} = \frac{1}{x^2}\left(\frac{\mathrm{d}^2y}{\mathrm{d}t^2} - \frac{\mathrm{d}y}{\mathrm{d}t}\right)$$

a dosazením do původní rovnice dostaneme

$$\frac{\mathrm{d}^2y}{\mathrm{d}t^2} - \frac{\mathrm{d}y}{\mathrm{d}t} + a\,\frac{\mathrm{d}y}{\mathrm{d}t} + by = \frac{\mathrm{d}^2y}{\mathrm{d}t^2} + (a\ -1)\frac{\mathrm{d}y}{\mathrm{d}t} + by = 0.$$

Charakteristická rovnice je tedy $\lambda^2 + (a-1)\lambda + b = 0$ a v závislosti na jejích kořenech získáme řešení původní rovnice: v případě dvou různých reálných kořenů λ_1, λ_2 se jedná o $y = k_1\mathrm{e}^{\lambda_1 t} + k_2\mathrm{e}^{\lambda_2 t} = k_1x^{\lambda_1} + k_2x^{\lambda_2}$, v případě jednoho dvojnásobného kořenu λ_1 je to $y = k_1\mathrm{e}^{\lambda_1 t} + k_2t\mathrm{e}^{\lambda_1 t} = k_1x^{\lambda_1} + k_2\ln(x)x^{\lambda_1}$ a v případě nereálných komplexně sdružených kořenů $\alpha \pm i\beta$ pak $y = \mathrm{e}^{\alpha t}(k_1\cos\beta t + k_2\sin\beta t) = x^{\alpha}(k_1\cos(\beta\ln x) + k_2\sin(\beta\ln x))$.

Příklad 2.1.3. Vyřešíme Eulerovu rovnici $x^2y'' + 3xy' + 10y = 0$.

Ručně bychom mohli postupovat takto:

```
(%i2) depends(y, x)$
      DR: x^2*diff(y,x,2) + 3*x*diff(y,x) + 10*y = 0;
(DR)
```

$$x^2\left(\frac{d^2}{dx^2}y\right) + 3x\left(\frac{d}{dx}y\right) + 10y = 0$$

V dalším kroku si kromě uložení substitucí uložíme znovu i rovnici ze zadání – nastavili jsme závislosti $y(t), t(x)$, takže první a druhá derivace y podle x budou mít tvary, na které se nám následně budou dobře aplikovat předchystané substituce.

```
(%i8) depends(y, t)$
      depends(t, x)$
      SUB: t = log(x)$
      SUB2: diff(SUB,x)$
      SUB3: diff(SUB2,x)$
      DR2: expand(x^2*diff(y,x,2) + 3*x*diff(y,x) + 10*y = 0);
(DR2)
```

$$\left(\frac{d}{dx}t\right)^2 x^2\left(\frac{d^2}{dt^2}y\right) + \left(\frac{d^2}{dx^2}t\right)x^2\left(\frac{d}{dt}y\right) + 3\left(\frac{d}{dx}t\right)x\left(\frac{d}{dt}y\right) + 10y = 0$$

```
(%i10) subst(rhs(SUB3), lhs(SUB3), DR2)$
       DR_KONST: subst(rhs(SUB2), lhs(SUB2), %);
(DR_KONST)
```

$$\frac{d^2}{dt^2}y+2\left(\frac{d}{dt}y\right)+10y=0$$

Tím jsme získali rovnici s konstantními koeficienty, kterou jsme již řešili výše. Příkazem `ode2` ji vyřešíme a provedeme zpět substituci do proměnné x.

```
(%i12) OR_t: ode2(DR_KONST, y, t);
       method;
(OR_t)
```

$$y=\%e^{-t}\left(\%k1\sin\left(3t\right)+\%k2\cos\left(3t\right)\right)$$

```
(%o12)
```

$$constcoeff$$

```
(%i13) subst(rhs(SUB), lhs(SUB), OR_t);
(%o13)
```

$$y=\frac{\%k1\sin\left(3\log\left(x\right)\right)+\%k2\cos\left(3\log\left(x\right)\right)}{x}$$

Použitím `ode2` na původní rovnici dostaneme stejný výsledek.

```
(%i15) ode2(DR, y, x);
       method;
(%o14)
```

$$y=\frac{\%k1\sin\left(3\log\left(x\right)\right)+\%k2\cos\left(3\log\left(x\right)\right)}{x}$$

```
(%o15)
```

$$euler$$

Besselova rovnice

Besselovou rovnicí řádu ν nazýváme diferenciální rovnici

$$x^2y''+xy'+(x^2-\nu^2)y=0.$$

Její řešení má tvar $y=k_1Y_\nu(x)+k_2J_\nu(x)$, kde

$$J_\nu(x)=\frac{1}{\pi}\int_0^\pi\cos(x\sin t-\nu t)\mathrm{d}t,$$

$$Y_\nu(x)=\frac{J_\nu(x)\cos\nu\pi-J_{-\nu}(x)}{\sin\nu\pi}$$

jsou Besselovy funkce prvního a druhého řádu. V této práci se jimi nebudeme více zabývat.

Příklad 2.1.4. Najdeme řešení Besselovy rovnice $x^2y'' + xy' + (x^2 - 9)y = 0$.

```
(%i2) depends(y, x)$
      DR: x^2*diff(y,x,2) + x*diff(y,x) + (x^2 - 9)*y = 0;
(DR)
```

$$x^2\left(\frac{d^2}{dx^2}y\right) + x\left(\frac{d}{dx}y\right) + \left(x^2 - 9\right)y = 0$$

```
(%i4) ode2(DR, y, x);
      method;
(%o3)
```

$$y = \text{bessel\_y}\,(3,x)\,\%k2 + \text{bessel\_j}\,(3,x)\,\%k1$$

```
(%o4)
```

bessel

Nehomogenní lineární rovnice

Poté, co příkaz `ode2` pro danou lineární rovnici

$$y'' + p(x)y' + q(x)y = r(x)$$

spočítal řešení homogenní rovnice $y'' + p(x)y' + q(x)y = 0$ ve tvaru $y = k_1y_1 + k_2y_2$, hledá metodou variace konstant partikulární řešení nehomogenní rovnice (za předpokladu, že v zadané rovnici $r(x) \neq 0$).

Metoda variace konstant pro rovnice druhého řádu spočívá v nahrazení integračních konstant k_1, k_2 v řešení homogenní rovnice neznámými funkcemi $k_1(x), k_2(x)$ a v řešení soustavy rovnic

$$k_1'(x)y_1 + k_2'(x)y_2 = 0,$$
$$k_1'(x)y_1' + k_2'(x)y_2' = r(x).$$

Z ní získáme partikulární řešení $y_p = k_1(x)y_1 + k_2(x)y_2$, po jehož přičtení k řešení homogenní rovnice dostaneme celkové řešení $y = k_1y_1 + k_2y_2 + y_p$.

Příklad 2.1.5. Spočteme řešení nehomogenní rovnice $y'' + y' - 2y = 8x^2e^{2x}$ (rovnice z příkladu 2.1.1 s nenulovou pravou stranou).

Ruční aplikace metody variace konstant by byla následující:

```
(%i2) depends(y, x)$
      DR: diff(y,x,2) + diff(y,x) - 2*y = 8*x^2*exp(2*x);
(DR)
```

$$\frac{d^2}{dx^2}y + \frac{d}{dx}y - 2y = 8x^2\,\%e^{2x}$$

Řešení homogenní rovnice s konstantními koeficienty, kterou jsme již řešili výše, necháme najít příkazem `ode2`.

```
(%i4) DR_HOM: lhs(DR) = 0$
      OR_HOM: ode2(DR_HOM, y, x);
(OR_HOM)
```

$$y = \%k1\,\%e^{x} + \%k2\,\%e^{-2x}$$

Uložíme si řešení y_1, y_2, zadefinujeme požadovanou soustavu rovnic a vyřešíme ji.

```
(%i10) Y1: coeff(rhs(OR_HOM), %k1)$
       Y2: coeff(rhs(OR_HOM), %k2)$
       depends(K1, x)$
       depends(K2, x)$
       PODM1: diff(K1,x)*Y1 + diff(K2,x)*Y2 = 0;
       PODM2: diff(K1,x)*diff(Y1,x) + diff(K2,x)*diff(Y2,x)
         = rhs(DR);
(PODM1)
```

$$\left(\frac{d}{dx}K1\right)\%e^{x} + \left(\frac{d}{dx}K2\right)\%e^{-2x} = 0$$

```
(PODM2)
```

$$\left(\frac{d}{dx}K1\right)\%e^{x} - 2\left(\frac{d}{dx}K2\right)\%e^{-2x} = 8x^2\,\%e^{2x}$$

```
(%i11) RES: solve([PODM1,PODM2], [diff(K1,x),diff(K2,x)]);
(RES)
```

$$[[\frac{d}{dx}K1 = \frac{8x^2\,\%e^{x}}{3}, \frac{d}{dx}K2 = -\frac{8x^2\,\%e^{4x}}{3}]]$$

```
(%i14) K1: integrate(rhs(RES[1][1]),x)$
       K2: integrate(rhs(RES[1][2]),x)$
       YP: ratsimp(K1*Y1 + K2*Y2);
(YP)
```

$$\frac{\left(8x^2 - 20x + 21\right)\%e^{2x}}{4}$$

Odtud tedy máme spočtené partikulární řešení nehomogenní rovnice, které přičteme k řešení homogenní rovnice.

```
(%i15) OR: lhs(OR_HOM) = rhs(OR_HOM) + YP;
(OR)
```

$$y = \frac{\left(8x^2 - 20x + 21\right)\%e^{2x}}{4} + \%k1\,\%e^{x} + \%k2\,\%e^{-2x}$$

Příkaz `ode2` najde metodou variace konstant totožné řešení nehomogenní rovnice. Rovněž partikulární řešení nehomogenní rovnice je stejné.

```
(%i18) ode2(DR, y, x);
       method;
       yp;
(%o16)
```

$$y = \frac{\left(8x^2 - 20x + 21\right)\ \%e^{2x}}{4} + \%k1\,\%e^{x} + \%k2\,\%e^{-2x}$$

```
(%o17)
```

$$variationofparameters$$

```
(%o18)
```

$$\frac{\left(8x^2 - 20x + 21\right)\ \%e^{2x}}{4}$$

Nelineární rovnice s chybějící závislou proměnnou

Jedním z typů nelineárních rovnic druhého řádu, které `ode2` dokáže řešit, jsou rovnice, ve kterých chybí nediferencovaná závislá proměnná y, tedy rovnice $F(x,y',y'')=0$. Substitucemi $y'=u$, $y''=u'$ můžeme rovnici převést na rovnici prvního řádu $F(x,u,u')=0$ a po jejím vyřešení získat integrací řešení původní rovnice.

Příklad 2.1.6. Nalezneme obecné řešení rovnice druhého řádu $x^2y''+(y')^2=0$.

Manuálně bychom mohli postupovat následně:

```
(%i2) depends(y, x)$
      DR: x^2*diff(y,x,2) + (diff(y,x))^2 = 0;
(DR)
```

$$x^2\left(\frac{d^2}{dx^2}y\right)+\left(\frac{d}{dx}y\right)^2=0$$

```
(%i7) depends(u, x)$
      SUB: diff(y,x) = u$
      SUB2: diff(y,x,2) = diff(u,x)$
      subst(rhs(SUB), lhs(SUB), DR)$
      DR_1R: subst(rhs(SUB2), lhs(SUB2), %);
(DR_1R)
```

$$\left(\frac{d}{dx}u\right)x^2+u^2=0$$

K vyřešení rovnice prvního řádu použijeme `ode2` a dále pomocí integrace získáme celkové řešení.

```
(%i10) ode2(DR_1R, u, x)$
       OR_1R: solve(%, u)[1];
       method;
(OR_1R)
```

$$u = \frac{x}{\%cx - 1}$$

```
(%o10)
```

separable

```
(%i11) OR: y = rhs(integrate(OR_1R, x));
(OR)
```

$$y = \frac{\log(\%cx - 1)}{\%c^2} + \frac{x}{\%c} + \%c1$$

Příkaz `ode2` použitý na zadanou rovnici druhého řádu vrátí stejný výsledek.

```
(%i13) ode2(DR, y, x);
       method;
(%o12)
```

$$y = \frac{\log(\%k1x - 1)}{\%k1^2} + \frac{x}{\%k1} + \%k2$$

```
(%o13)
```

freeofy

Nelineární rovnice s chybějící nezávislou proměnnou

Posledním typem rovnic druhého řádu, které dokáže `ode2` řešit, jsou rovnice, ve kterých explicitně chybí nezávislá proměnná, tedy $F(y, y', y'') = 0$. Tyto rovnice je možné řešit substitucemi $y' = u$, $y'' = u'$, přičemž pokud budeme dočasně považovat y za nezávislou proměnnou, dostáváme rovnici prvního řádu $F(y, u, u') = 0$. V této rovnici se ale nachází tři proměnné, neboť derivujeme u vzhledem k x. Protože ale platí

$$u' = \frac{\mathrm{d}u}{\mathrm{d}x} = \frac{\mathrm{d}u}{\mathrm{d}y}\frac{\mathrm{d}y}{\mathrm{d}x} = \frac{\mathrm{d}u}{\mathrm{d}y}u \,,$$

řešíme po aplikaci tohoto odvození rovnici prvního řádu $F(y, u, \frac{\mathrm{d}u}{\mathrm{d}y}u) = 0$ se závislou proměnnou $u(y)$ a nezávislou y. Po jejím vyřešení už zbývá jen vyřešit $y' = u(y)$, čímž získáme řešení původní rovnice.

Příklad 2.1.7. Vyřešíme rovnici druhého řádu $yy'' - 2(y')^3 = 0$.

Ručně bychom postupovali například následně:

```
(%i2) depends(y, x)$
      DR: y*diff(y,x,2) -2*(diff(y,x))^3 = 0;
(DR)
```

$$y\left(\frac{d^2}{dx^2}y\right) - 2\left(\frac{d}{dx}y\right)^3 = 0$$

Uložíme si a aplikujeme substituce popsané výše. Dostaneme rovnici prvního řádu, kterou vyřešíme pomocí `ode2`.

```
(%i8) depends(u, y)$
      SUB: u = diff(y,x)$
      diff(SUB,x)$
      SUB2: subst(lhs(SUB), rhs(SUB), %)$
      subst(lhs(SUB), rhs(SUB), DR)$
      DR_1R: subst(lhs(SUB2), rhs(SUB2), %);
(DR_1R)
```

$$u\left(\frac{d}{dy}u\right)y-2u^3=0$$

```
(%i11) ode2(DR_1R, u, y)$
       OR_1R: solve(%, u)[1];
       method;
(OR_1R)
```

$$u=-\frac{1}{2\log(y)+2\%c}$$

```
(%o11)
```

separable

Znovu použijeme ode2, abychom získali řešení původní rovnice ze zadání. Pro přehlednost nahradíme původní integrační konstantu, aby bylo jasné, která byla přidána prvním a která druhým voláním ode2.

```
(%i15) subst(rhs(SUB), lhs(SUB), OR_1R)$
       subst(%k1, %c, %)$
       OR: ode2(%, y, x);
       method;
(OR)
```

$$(2-2\%k1)\,y-2y\log(y)=x+\%c$$

```
(%o15)
```

separable

Použitím ode2 přímo na rovnici ze zadání dostaneme totožný výsledek.

```
(%i17) ode2(DR, y, x);
       method;
(%o16)
```

$$(2-2\%k1)\,y-2y\log(y)=x+\%k2$$

```
(%o17)
```

freeofx

2.1.2 Příkaz `desolve`

Příkaz `desolve`, který hledá řešení pomocí Laplaceovy transformace a který jsme v kapitole 1.1.2 použili k řešení lineárních diferenciálních rovnic 1. řádu, je možné použít i na lineární rovnice vyšších řádů. Buď opět jeho variantu
`desolve(difRce, y(x))`,
kde `difRce` je obecně diferenciální rovnice n. řádu, tj. obsahuje `diff(y(x),x,n)`, nebo variantu
`desolve([difRce1, ... , difRcen], [y1(x), ... , yn(x)])`,
která řeší soustavu n lineárních rovnic se závislými proměnnými $y_1(x), y_2(x), \ldots, y_n(x)$. Opět platí, že závislosti proměnných je nutné explicitně uvádět, jak u závislých proměnných, tak u jejich případných derivací, a že každá rovnice musí obsahovat symbol =.

Počáteční podmínku zadáme před samotným zavoláním `desolve` opět pomocí příkazu `atvalue`. U rovnic druhého řádu zadáme nejen $y(0) = y_0$, ale i $y'(0) = a$.

Příklad 2.1.8. Najdeme řešení nehomogenní lineární diferenciální rovnice druhého řádu $y'' + 2y' + y = \mathrm{e}^{-x}$ pro počáteční podmínku $y(0) = 1$, $y'(0) = -1$.

K výsledku dojdeme, pokud rovnici zadáme jako jednu rovnici druhého řádu nebo jako dvě rovnice prvního řádu.

```
(%i1) DR: diff(y(x),x,2) + 2*diff(y(x),x) + y(x) = exp(-x);
(DR)
```

$$\frac{d^2}{dx^2}y(x) + 2\left(\frac{d}{dx}y(x)\right) + y(x) = \%e^{-x}$$

```
(%i4) atvalue(y(x), x=0, 1)$
      atvalue(diff(y(x),x), x=0, -1)$
      desolve(DR, y(x));
(%o4)
```

$$y(x) = \frac{x^2\,\%e^{-x}}{2} + \%e^{-x}$$

Rovnici ze zadání můžeme rozdělit na dvě rovnice. Pokud například položíme $y' = u$, pak jako druhou rovnici dostaneme $u' + 2u + y = \mathrm{e}^{-x}$. Příkaz `desolve` vrátí stejný výsledek jako výše. Pomocí `atvalue` zadáme už jen hodnotu $u(0) = y'(0) = -1$, protože dříve zadané hodnoty si Maxima stále pamatuje.

```
(%i8) f: diff(y(x),x) = u(x)$
      g: diff(u(x),x) + 2*u(x) + y(x) = exp(-x)$
      atvalue(u(x), x=0, -1)$
      desolve([f,g], [y(x),u(x)]);
(%o8)
```

$$[y(x) = \frac{x^2\,\%e^{-x}}{2} + \%e^{-x}, u(x) = -\frac{x^2\,\%e^{-x}}{2} + x\,\%e^{-x} - \%e^{-x}]$$

2.2 Numerické řešení

2.2.1 Příkaz `plotdf`

Příkaz `plotdf` umožňuje vykreslit řešení dvou autonomních diferenciálních rovnic prvního řádu $x' = g(x,y)$ a $y' = f(x,y)$, kde $x = x(t)$ a $y = y(t)$. První představuje změnu proměnné vynesené na horizontální a druhá proměnné vynesené na vertikální ose v závislosti na parametru t (tj. například na čase). V tomto případě uvedeme seznam s funkcemi v pořadí g, f jako první parametr příkazu a syntaxe je tedy
`plotdf([g(x,y), f(x,y)])`,
případně
`plotdf([g(u,v), f(u,v)], [u,v])`,
pokud funkce g a f nezávisí na proměnných x a y (seznam s proměnnými musí být uveden opět jako druhý parametr).

Předpokládejme, že chceme vykreslit řešení diferenciální rovnice druhého řádu rozřešitelné vzhledem ke druhé derivaci tvaru $y'' = f(x,y,y')$. Pak tuto rovnici můžeme rozdělit na dvě autonomní diferenciální rovnice: položíme-li $y = y(t)$, $x = x(t) = y'(t)$, dostáváme $x'(t) = y''(t) = f(x(t),y(t))$. Výsledné dvě rovnice, které můžeme nechat `plotdf` vykreslit tedy budou

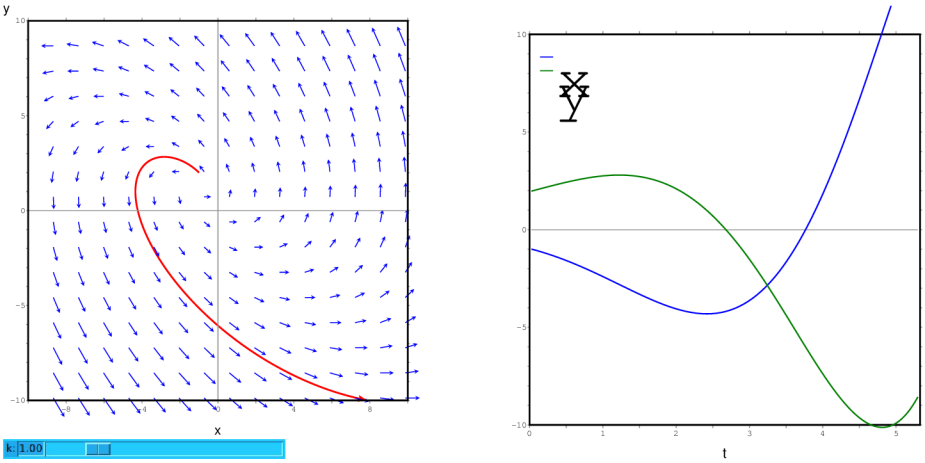
$$x'(t) = f(x(t),y(t)),$$
$$y'(t) = x(t).$$

U takto zadaných rovnic můžeme zapnout volbu `[versus_t, n]`, kde `n` je libovolné přirozené číslo, díky které se nám otevře i druhé okno zobrazující zvlášť křivky $x(t)$ a $y(t)$. Implicitní hodnota 0 nastavuje, aby se toto druhé okno neotevíralo. Při současném použití voleb `versus_t` a `sliders` se bude při změnách libovolného parametru na posuvníku tato změna projevovat současně i v druhém okně s parametrickými křivkami. Naopak až při dodatečném zobrazení druhého okna, což můžeme udělat kliknutím myší na ikonu příslušného nástroje, se zobrazí parametrické křivky fixně jen pro aktuální hodnotu parametrů na posuvnících a následné změny se už nebudou projevovat. Pro popis volby `sliders` a dalších, které jsou použity v příkladu níže, viz kapitolu 1.2.1 popisující možnosti `plotdf` na rovnicích prvního řádu.

Příklad 2.2.1. Zobrazíme graf funkce $y'' = y' - y + k$, $k \in \mathbb{R}$ tedy (po transformaci) rovnic $x'(t) = k - y(t)$, $y'(t) = x(t) + y(t)$ i zvlášť graf křivek $x(t)$, $y(t)$.

```
(%i3) dxdt: k-y$
      dydt: x+y$
      plotdf([dxdt, dydt], [versus_t, 1], [parameters, "k=1"],
        [sliders, "k=0.1:5"], [direction, forward],
        [trajectory_at, -1, 2])$
```

(viz obrázek 2.2)



Obrázek 2.2: Vlevo řešení $x'(t) = k - y$, $y'(t) = x + y$ pro $k = 1$. Vpravo zvlášť křivky $x(t)$, $y(t)$ (export legendy do PostScriptu opět není ideální).

2.2.2 Příkaz `drawdf`

Podobně máme možnost zobrazit řešení diferenciální rovnice druhého řádu i pomocí varianty příkazu `drawdf`, kdy vykresluje řešení dvou autonomních rovnic $x' = g(x,y)$, $y' = f(x,y)$, kde $x = x(t)$ a $y = y(t)$. Výrazy $g(x,y)$ a $f(x,y)$ musíme v tomto pořadí a ve formě seznamu uvést, stejně jako v `plotdf`, na prvním místě:
`drawdf([g(x,y), f(x,y)])`,
případně
`drawdf([g(u,v), f(u,v)], [u,v])`
nebo
`drawdf([g(u,v), f(u,v)], [u,uMin,uMax], [v,vMin,vMax])`.

Příkaz `drawdf` nabízí možnost, jak vykreslit partikulární řešení procházející sedlovým bodem. Při použití objektů
`saddle_at(x0,y0)`
nebo
`saddles_at([x1,y1], [x2,y2], ..., [xN,yN])`
se pokusí v daných bodech linearizovat rovnici a pokud se jedná o sedlový bod, vykreslit partikulární řešení. Konkrétně se tedy pokusí získat tvar

$$x'(t) = ax(t) + by(t)$$

$$y'(t) = cx(t) + dy(t),$$

přičemž o sedlový bod se jedná, pokud má matice $A = \left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right)$ jedno kladné a jedno záporné reálné vlastní číslo. Následně `drawdf` vykreslí obě partikulární řešení příslušné vlastním vektorům, které udávají jejich směr.

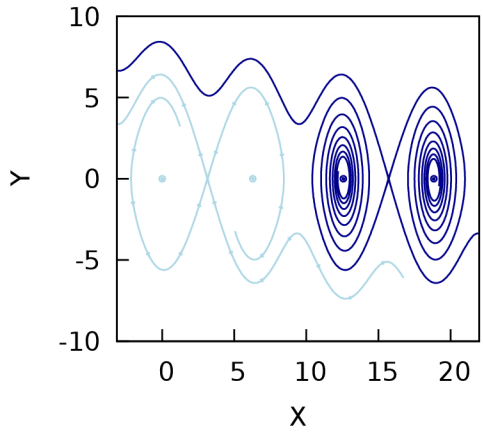
Díky volbě `show_field=false` máme možnost skrýt implicitně vykreslované směrové pole. V příkladu níže použijeme i volby `drawdf` popsané v kapitole 1.2.2, kde s příkazem řešíme rovnice prvního řádu.

Příklad 2.2.2. Vykreslíme dvě dvojice partikulárních řešení rovnice $y'' = -9y\cos x - \frac{y'}{5}$ (transformované na $x' = y$, $y' = -9\sin x - \frac{y}{5}$) v sedlových bodech $[\pi,0]$ a $[5\pi,0]$.

```
(%i3) load(drawdf)$
      dxdt: y$
      dydt: -9*sin(x)-y/5$

(%i4) drawdf([dxdt, dydt], [x,-%pi,7*%pi],
        show_field=false, tstep=0.01,
        nsteps=2000, color=dark-blue, saddle_at(5*%pi,0),
        points_at([4*%pi,0],[6*%pi,0]),
        nsteps=500, color=light-blue,
        soln_arrows=true, saddle_at(%pi,0),
        points_at([0,0],[2*%pi,0]))$
```

(viz obrázek 2.3)



Obrázek 2.3: Dva sedlové body rovnice $x' = y$, $y' = -9\sin x - \frac{y}{5}$ a jim příslušná čtyři partikulární řešení.

2.2.3 Příkaz `rk`

Příkaz `rk` je možné použít nejen na jednu diferenciální rovnici prvního řádu jako v kapitole 1.2.3, ale obecně na soustavu n rovnic

$$\begin{aligned} y_1' &= f_1(x, y_1, y_2, \ldots, y_n), \\ y_2' &= f_2(x, y_1, y_2, \ldots, y_n), \\ &\vdots \\ y_n' &= f_n(x, y_1, y_2, \ldots, y_n), \end{aligned}$$

ve kterých se vyskytují závislé proměnné $y_1, y_2, \ldots, y_n$ spolu s nezávislou proměnnou x (která je nutně všem rovnicím společná). Syntaxe příkazu je pak
`rk([f1, f2, ..., fn], [y1, y2, ..., yn], [y1(x0), y2(x0),`
`..., yn(x0)], [x, x0, xN, h]),`
kde význam všech parametrů je obdobný jako v případě jedné diferenciální rovnice – s tím rozdílem, že první tři argumenty příkazu netvoří vždy jeden výraz nebo hodnota ale seznam o n prvcích. V této variantě příkazu je potřeba dodržet pořadí prvků v prvních třech seznamech tak, aby spolu vždy korespondovaly funkce f_i definující derivaci závislé proměnné y_i, označení této proměnné a její počáteční hodnota $y_i(x_0)$.

Příkaz `rk` pak vrací seznam, jehož každý prvek je vektorem o $n+1$ prvcích: první složka představuje uzel a následují spočtené hodnoty závislých proměnných $y_1, y_2, \ldots, y_n$ v tomto uzlu. Stejně jako v případě jedné rovnice platí, že délka seznamu je nejvýše $N+1$, kde $N = \frac{x_N - x_0}{h}$. Pokud v průběhu výpočtů některá ze závislých proměnných dosáhne příliš velké absolutní hodnoty, výpočet v tomto uzlu skončí a `rk` vrátí seznam o N či méně prvcích.

Stejně jako u příkazů `plotdf` a `drawdf` pomocí transformace diferenciální rovnice druhého řádu na dvě autonomní rovnice využijeme příkaz `rk` pro řešení soustavy 2 rovnic $x' = g(x,y)$, $y' = f(x,y)$, kde $x = x(t)$ a $y = y(t)$:
`rk([g(x,y), f(x,y)], [x,y], [x(t0), y(t0)], [t, t0, tN, h])`

Příklad 2.2.3. Vykreslíme řešení obecné rovnice druhého řádu $y'' - xy' = \frac{x}{y^2}$ pro počáteční podmínku $y(-3) = 2$, $y'(-3) = 4$.

```
(%i2) depends(y, x)$
      DR: diff(y,x,2) - x*diff(y,x) = x/y^2;
(DR)
```

$$\frac{d^2}{dx^2}y - x\left(\frac{d}{dx}y\right) = \frac{x}{y^2}$$

Zadanou rovnici převedeme na dvě autonomní rovnice rozřešené vzhledem k derivaci. Ty pak jako argumenty předáme příkazu `rk`.

```
(%i7) depends(u, x)$
      SUB: u = diff(y,x);
      SUB2: diff(SUB,x)$
      subst(lhs(SUB), rhs(SUB), DR)$
      dudx: solve(subst(lhs(SUB2), rhs(SUB2), %), diff(u,x))[1];
(SUB)
```

$$u = \frac{d}{dx}y$$

```
(dudx)
```

$$\frac{d}{dx}u = \frac{uxy^2 + x}{y^2}$$

```
(%i11) body: rk([lhs(SUB), rhs(dudx)], [y, u], [2, 4],
         [x, -3, 3, 0.05])$
       length(body);
       first(body);
       last(body);
(%o9)
```

$$121$$

```
(%o10)
```

$$[-3.0, 2.0, 4.0]$$

```
(%o11)
```

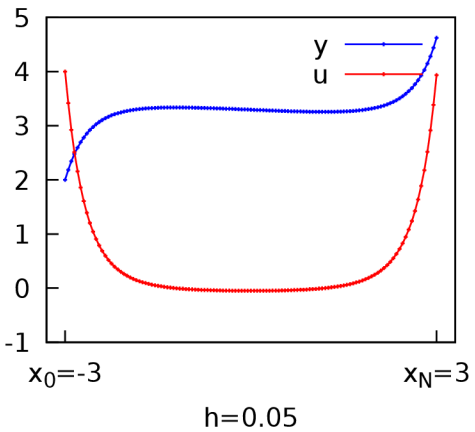
$$[3.0, 4.621339652840539, 3.933388307130357]$$

Pomocí `makelist` seznam `body` rozdělíme na dva seznamy bodů pro funkce $y(x)$ a $u = y'(x)$. Tyto body pak pomocí `draw2d` vykreslíme.

```
(%i13) yBody: makelist([body[i][1],body[i][2]],i,1,length(body))$
       uBody: makelist([body[i][1],body[i][3]],i,1,length(body))$

(%i15) load(draw)$
       draw2d(points_joined=true, point_size=0.5,
         xrange=[-3.3,3.3], yrange=[-1,5],
         key="y", points(yBody),
         color=red, key="u", points(uBody),
         xtics={["x_0=-3",-3],["x_N=3",3]}, xlabel="h=0.05",
         ylabel="")$
```

(viz obrázek 2.4)



Obrázek 2.4: Řešení rovnice $y'' - xy' = \frac{x}{y^2}$ pro počáteční podmínku $y(-3) = 2, y'(-3) = 4$. Vykresleny křivky $y(x)$ a $u(x) = y'(x)$.

2.3 Tabulky příkazů

`ode2`:

`ode2(difRce, y, x)`	analytické řešení diferenciální rovnice, `difRce` obsahuje `'diff(y,x,2)`

`desolve`:

`desolve([difRce1, ..., difRcen], [y1(x), ..., yn(x)])`	analytické řešení soustavy n diferenciálních rovnic

`plotdf`:

`plotdf([g(x,y), f(x,y)])`	směrové pole vektorů dif. rovnice zadané $x' = g(x,y), y' = f(x,y)$
`plotdf([g(u,v), f(u,v)], [u,v])`	totéž pro jiné proměnné než x a y

`drawdf`:

`drawdf([g(x,y), f(x,y)])`	směrové pole vektorů dif. rovnice zadané $x' = g(x,y), y' = f(x,y)$
`drawdf([g(u,v), f(u,v)], [u,v])`	totéž pro jiné proměnné než x a y
`drawdf([g(u,v), f(u,v)], [u,uMin,uMax], [v,vMin,vMax])`	totéž pro jiné proměnné než x a y, spojena definice proměnných s jejich rozsahem

`rk`:

`rk([f1, ..., fn], [y1, ..., yn], [y1(x0), ..., yn(x0)], [x, x0, xN, h])`	numerické řešení soustavy n rovnic, vrací seznam souřadnic bodů (uzel + hodnoty závislých prom. v tomto uzlu)

Tabulka 2.1: Příkazy pro analytické i numerické řešení diferenciálních rovnic druhého řádu.

`plotdf`:

`[versus_t, 1]`	zobrazení grafu křivek $x(t)$ a $y(t)$

`drawdf`:

`saddle_at(x0,y0)`	partikulární řešení v sedlovém bodě $[x_0, y_0]$
`saddles_at([x1,y1], ..., [xN,yN])`	totéž pro více sedlových bodů
`show_field=true/false`	zobrazení směrového pole

Tabulka 2.2: Grafické objekty a volby příkazů `plotdf` a `drawdf`.

Kapitola 3

Řešení diferenciálních rovnic prvního řádu nerozřešených vzhledem k derivaci

Druhou část této práce tvoří balíček funkcí `ode_mm`, které řeší diferenciální rovnice prvního řádu nerozřešené vzhledem k derivaci. V této kapitole postupně, podobně jako v předcházejících, projdeme jednotlivé typy rovnic a kromě ručních úprav a hledání řešení na zadané rovnice použijeme i příkazy z tohoto balíčku.

Balíček ve formátu `.mac` je potřeba načíst pomocí
`load("cesta/ode_mm.mac")`,
tedy například
`load("/home/user/ode_mm.mac")`.

3.1 Příkazy balíčku `ode_mm`

Balíček nabízí pět základních příkazů s podobnou syntaxí a několik pomocných funkcí pro větší přehlednost kódu. Za zmínku stojí dvojice funkcí, kde jedna z nich převede vstup (například rovnici, výraz nebo jejich seznam) do nových proměnných a druhá jej převede zpět do původních proměnných ze zadání. Důvod těchto převodů je, že původní proměnné mohou mít nastaveny určité závislosti, které by mohly zapříčinit nechtěné výsledky v následných derivacích. Proto v balíčku postupujeme bezpečnějším způsobem, kdy si potřebné závislosti zadáme v nových proměnných a na konci úprav tyto závislosti opět zrušíme.

3.1.1 Obecné řešení

Prvním příkazem balíčku je stejnojmenný
`ode_mm(difRce, y, x)`,
kde `difRce` je zkoumaná diferenciální rovnice prvního řádu, y její závislá a x nezávislá proměnná. Příkaz, který je napsán k řešení diferenciálních rovnic prvního řádu nerozřešených vzhledem k derivaci, postupně zkouší, zda je rovnice jednoho z typů popsaných níže, a volí příslušný postup řešení. Řešení je vraceno v parametrickém tvaru $x = x(p,c), y = y(p,c)$, kde $p = y', c \in \mathbb{R}$. Příkaz nehledá případná singulární řešení. Zadaná rovnice musí být diferenciální rovnicí prvního řádu, v opačném případě příkaz vrací `false`. Do proměnné `method` je pak v případě úspěšného výpočtu nastaven řetězec identifikující, o jaký typ

rovnice se jednalo. Pokud rovnice není žádným z těchto typů, příkaz vrátí řešení, které nachází `ode2` metodami popsanými v kapitole 1.1.1, případně příkaz `contrib_ode` ze stejnojmenného balíčku.

Balíček `contrib_ode` představuje funkce, které jsou ve fázi vývoje a nejsou zatím zařazeny mezi standardní balíčky Maximy. I tento balíček je nutné před použitím načíst pomocí `load(contrib_ode)`. Příkaz zkouší zadanou rovnici postupně řešit těmito postupy: faktorizace, Clairautova rovnice, Lagrangeova rovnice, Riccatiho rovnice, Abelova rovnice a Lieova metoda symetrií.

Pokud se nepodaří nalézt řešení pomocí `ode_mm`, `ode2` ani `contrib_ode`, vrátí příkaz `false`.

Základní kostra procedury `ode_mm` je uvedena níže. Kódy jednotlivých pomocných funkcí ukážeme v následujících podkapitolách.

```
/*
Hleda obecne reseni diferencialni rovnice prvniho radu nerozresene
vzhledem k derivaci. Pripadna singularni reseni nehleda.
- eqn je zkoumana diferencialni rovnice
- y je zavisla promenna
- x je nezavisla promenna
*/
ode_mm(eqn,y,x) := block(
     [res,deg],

     deg: derivdegree(eqn,y,x),
     /*Pokud se nejedna o diferencialni rovnici prvniho radu, vrati
     false.*/
     if (is(deg#1)) then
          return(false),

     /*Zkusi resit jako rovnici typu y = f(y') .*/
     res: yEqFp(eqn,y,x),
     if (is(res#false)) then
          return(res),

     /*Zkusi resit jako rovnici typu x = f(y') .*/
     res: xEqFp(eqn,y,x),
     if (is(res#false)) then
          return(res),

     /*Zkusi resit jako rovnici typu y = f(x,y') .*/
     res: yEqFxp(eqn,y,x),
     if (is(res#false)) then
          return(res),

     /*Zkusi resit jako rovnici typu x = f(y,y') .*/
     res: xEqFyp(eqn,y,x),
```

```
     if (is(res#false)) then
          return(res),

     /*Pokud se nepodari vyresit jako rovnice typu vyse,
     zkusi vyresit pomoci balicku contrib_ode (ten bude v budoucnu
     smazan a presunut do ode2, proto je testovana jeho pritomnost
     nize pomoci file_search). contrib_ode zkusi resit pomoci ode2
     (postupne jako linearni, se separovanymi promennymi, exaktni
     [vc. pripadneho integracniho faktoru], homogenni, Bernoulliovu a
     zobecnenou homogenni). Pokud stale nenajde reseni, tak pripadne
     zkusi dalsi metody: faktorizace, Clairautova rce,
     Lagrangeova rce, Riccatiho rce (pokud je jen castecne vyresena,
     zkusi metodu Lieovych symetrii) a Abelova rovnice.*/
     if (file_search(contrib_ode)#false) then (
          load(contrib_ode),
          res: contrib_ode(eqn,y,x)
     ) else res: ode2(eqn,y,x),

     return(res)
)$
```

3.1.2 Diskriminantní křivky

Druhým příkazem je
`discurve(difRce, y, x)`,
který nalezne diskriminantní křivky zadané diferenciální rovnice prvního řádu tvaru $F(x,y,p) = 0$, $p = y'$. Pokud `difRce` nepředstavuje diferenciální rovnici prvního řádu, vrací `false`. Jinak vrací řešení ve formě seznamu dvojic $[p = p(x), y = y(x)]$. Nastavení, kdy je kromě řešení $y = y(x)$ vracen i příšlušný tvar parametru $p = p(x)$, je vhodné z toho důvodu, že příkaz `discurve` je volán i příkazem `singular`, kde bychom jinak museli tvar parametru znovu hledat.

Níže je kód procedury:

```
/*
Hleda diskriminantni krivky diferencialni rovnice prvniho radu,
tj. resi soustavu rovnic F(x,y,p)=0 a F_p(x,y,p)=0. Vraci dvojice
[p, y].
- eqn je zkoumana diferencialni rovnice
- y je zavisla promenna
- x je nezavisla promenna
*/
discurve(eqn,y,x) := block(
     [r:lhs(eqn)-rhs(eqn)=0,s,ans:[]],

     /*Pokud se nejedna o diferencialni rovnici prvniho radu, vrati
     false.*/
```

```
    if (is(derivdegree(eqn,y,x)#1)) then
         return(false),

    /*nahrazeni novymi promennymi, ve kterych se bude pocitat*/
    r: newVars(r,y,x,p),

    depends(ppp, yyy),
    r: ratsubst(ppp,'diff(yyy,xxx),r),

    /*musi se jednat o diferencialni rovnici, tj. musi obsahovat
    nejakou derivaci*/
    if (is(freeof(ppp,r))) then
         return(false),

    s: diff(r,ppp),

    /*najit dvojice reseni [p = ..., y = ...]*/
    ans: solve([r,s],[ppp,yyy]),

    /*vraceni zpet do puvodnich promennych a zruseni zavislosti*/
    ans: oldVars(ans,y,x,p),

    return(factor(radcan(ans)))
)$
```

3.1.3 Singulární řešení

Příkaz
`singular(difRce, y, x)`
vrací seznam singulárních řešení tvaru $y = y(x)$ zadané diferenciální rovnice prvního řádu $F(x,y,p) = 0$, $p = y'$, tj. řešení, jejichž každým bodem prochází ještě nějaké další řešení této rovnice, a tak porušují podmínku jednoznačnosti. Pokud žádné singulární řešení nenalezne, vrací prázdný seznam.

Postup, který aplikují i příkazy `discurve` a `singular` je následující: vyřešením soustavy rovnic

$$F(x,y,p) = 0$$

$$F_p(x,y,p) = 0$$

získáme tzv. diskriminantní křivky. Tato soustava představuje nutné, ale ne postačující podmínky na singulární řešení, neboť každé singulární řešení je zároveň diskriminantní křivkou. Opačné tvrzení ale neplatí, a tak máme pro singulární řešení další dvě podmínky

$$F_x(x,y,p) + pF_y(x,y,p) = 0,$$

$$F_y(x,y,p) \neq 0.$$

Zde je kód procedury:

```
/*
Hleda singularni reseni diferencialni rovnice prvniho radu,
tj. diskriminantni krivky, ktere splnuji F_x + p*F_y = 0 a F_y != 0.
- eqn je zkoumana diferencialni rovnice
- y je zavisla promenna
- x je nezavisla promenna
*/
singular(eqn,y,x) := block(
     [F:lhs(eqn)-rhs(eqn),F_x,F_y,expr1,expr2,ans:[]],

     /*ziskat diskriminantni krivky*/
     ans: discurve(eqn,y,x),

     /*nahrazeni novymi promennymi, ve kterych se bude pocitat*/
     ans: newVars(ans,y,x,p),

     /*Pokud se nejednalo o diferencialni rovnici prvniho radu
     a discurve vratila false, vrati rovnez false.*/
     if (is(equal(ans,false))) then
          return(false),

     /*nahrazeni novymi promennymi, ve kterych se bude pocitat*/
     F: newVars(F,y,x,p),

     F: ratsubst(ppp,'diff(yyy,xxx),F),

     /*predpocitat parcialni derivace*/
     F_x: diff(F,xxx),
     F_y: diff(F,yyy),

     /*ponechat pouze diskriminantni krivky, ktere splnuji dodatecne
     dve podminky pro singularni reseni*/
     for t in ans do (
          expr1: F_x + ppp*F_y,
          expr1: ratsubst(rhs(t[1]),ppp,expr1),
          expr1: ratsubst(rhs(t[2]),yyy,expr1),
          expr2: F_y,
          expr2: ratsubst(rhs(t[1]),ppp,expr2),
          expr2: ratsubst(rhs(t[2]),yyy,expr2),

          if (not(is(equal(expr1,0)) and not(is(equal(expr2,0)))))
               then (
                    ans: delete(t,ans)
               )
```

```
     ),

     /*ponechat pouze reseni y = y(x), tj. jen druhy sloupec matice*/
     ans: makelist(ans[i][2],i,1,length(ans)),

     /*vraceni zpet do puvodnich promennych a zruseni zavislosti*/
     ans: oldVars(ans,y,x,p),

     return(ans)
)$
```

3.1.4 Řešení v grafu

Čtvrtým příkazem je
`draw_ode_mm(difRce, y, x)`,
kde `difRce` je opět zkoumaná diferenciální rovnice prvního řádu, y její závislá a x nezávislá proměnná. Příkaz najde obecná i singulární řešení zadané rovnice a vykreslí je do jednoho grafu pomocí příkazu `draw2d`. Obecná a singulární řešení barevně odliší a označí je v legendě grafu.

Kód procedury je vypsán níže:

```
/*
Vykresli do grafu obecne i singularni reseni diferencialni rovnice
prvniho radu.
- eqn je zkoumana diferencialni rovnice
- y je zavisla promenna
- x je nezavisla promenna
*/
draw_ode_mm(eqn,y,x) := block(
     [sol:[],sing_solns:[],solns,i,t,solns_gr2d,solns_gr2d_1,
     solns_gr2d_2,sing_solns_gr2d,sing_solns_gr2d_1,
     sing_solns_gr2d_2],

     /*spocteni obecneho a singularniho reseni*/
     sol: ode_mm(eqn,y,x),
     sing_solns: singular(eqn,y,x),

     /*dosazeni 3 hodnot za integracni konstantu obecneho reseni*/
     solns: [],
     i: 1,
     errormsg: false,
     while (length(solns)<3) do (
          t: errcatch(subst(i,%c,sol)),
          /*dosadit pouze, pokud nebude vysledkem chyba*/
          if (length(t)>0) then (
               solns: cons(t[1],solns)
```

```
     ),
     i: i+1
),
errormsg: true,

/*obecne reseni prevest na objekty parametric k vykresleni
v draw2d*/
solns_gr2d: makelist((parametric(rhs(solns[i][1]),
     rhs(solns[i][2]),p,-10,10)),i,1,length(solns)),

/*rovnice typu y=f(y') muze mit jeste jedno obecne reseni
- pridat jej, pokud existuje - procedura jej vraci v explicitnim
tvaru*/
if (is(equal(length(sol),3))) then
     solns_gr2d: cons(explicit(rhs(sol[3]),x,-10,10),
          solns_gr2d),

/*rozdeleni seznamu na dva (jinak by se opakovaly polozky
legendy vysledneho grafu)*/
solns_gr2d_1: solns_gr2d[1],
solns_gr2d_2: delete(solns_gr2d[1],solns_gr2d),

/*singularni reseni prevest na objekty explicit k vykresleni
v draw2d*/
sing_solns_gr2d: makelist((explicit(rhs(sing_solns[i]),
     x,-10,10)),i,1,length(sing_solns)),

/*opet rozdeleni seznamu*/
sing_solns_gr2d_1: [],
sing_solns_gr2d_2: [],
/*pouze pro neprazdny puvodni seznam*/
if (is(length(sing_solns_gr2d) > 0)) then (
     sing_solns_gr2d_1: sing_solns_gr2d[1],
     sing_solns_gr2d_2: delete(sing_solns_gr2d[1],
          sing_solns_gr2d)
),

/*vykresleni grafu - polozky legendy se neopakuji*/
load(draw),
draw2d(
     color=green,
     key="Obecne reseni",
     solns_gr2d_1,
     key="",
     solns_gr2d_2,
```

```
            color=red,
            key="Singularni reseni",
            sing_solns_gr2d_1,
            key="",
            sing_solns_gr2d_2
        )
)$
```

3.1.5 Izogonální trajektorie

Posledním příkazem je příkaz na spočtení izogonální trajektorie daného systému křivek. Tematicky patří více k diferenciálním rovnicím prvního řádu, a tak jsme jej popsali v kapitole 1.3.

Zmíněná dvojice procedur k převedení do nových proměnných a zpět do původních má následující kód:

```
/*
Pomocna funkce k prevedeni do novych promennych, aby bylo jiste, ze
na promennych, se kterymi pocitame, nejsou zadne zavislosti.
- eqn je zkoumana diferencialni rovnice (nebo vyraz)
- y je zavisla promenna, bude nahrazena za yyy
- x je nezavisla promenna, bude nahrazena za xxx
- p je zavedeny parametr, bude nahrazen za ppp
*/
newVars(eqn,y,x,p) := block(
     [ans],

     /*nahrazeni novymi promennymi*/
     ans: ratsubst(xxx,x,eqn),
     ans: ratsubst(yyy,y,ans),
     ans: ratsubst(ppp,p,ans),

     return(ans)
)$


/*
Pomocna funkce k prevedeni zpet do puvodnich promennych a zruseni
zavislosti mezi pomocnymi promennymi.
- eqn je zkoumana diferencialni rovnice (nebo vyraz)
- y je puvodni zavisla promenna, pomocna byla yyy
- x je puvodni nezavisla promenna, pomocna byla xxx
- p je puvodni zavedeny parametr, pomocny byl ppp
*/
oldVars(eqn,y,x,p) := block(
     [ans],

     /*nahrazeni puvodnimi promennymi*/
     ans: ratsubst(x,xxx,eqn),
     ans: ratsubst(y,yyy,ans),
     ans: ratsubst(p,ppp,ans),
     /*zruseni zavislosti*/
     remove(xxx,dependency),
     remove(yyy,dependency),
     remove(ppp,dependency),

     return(ans)
)$
```

3.2 Rovnice typu $y = f(y')$

Rovnici tvaru $y = f(y')$ lze řešit následujícím postupem. Substitucí $p = y'$, uvažováním parametru jakožto $p = p(x), p \neq 0$, a derivováním rovnice podle x dostáváme

$$p = f'(p)\frac{\mathrm{d}p}{\mathrm{d}x}$$

a odtud

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

Jedná se o rovnici se separovanými proměnnými, jejíž obecným řešením je

$$x = \int \frac{f'(p)}{p}\mathrm{d}p + c, \quad c \in \mathbb{R}.$$

Celkovým řešením je tedy

$$x = \int \frac{f'(p)}{p}\mathrm{d}p + c$$
$$y = f(p).$$

Pokud je funkce $f(p)$ definována pro $p = 0$, pak je přímka $y = f(0)$ rovněž řešením. Může, ale nemusí, se jednat o singulární řešení, neboť z podmínek pro singulární řešení máme:

$$F(x,y,p) = y - f(p) = 0,$$
$$F_p(x,y,p) = -f'(p) = 0,$$
$$F_x(x,y,p) + pF_y(x,y,p) = 0 + p \cdot 1 = 0,$$
$$F_y(x,y,p) = 1 \neq 0.$$

První podmínka říká, že singulární řešení je také řešením dané rovnice. Čtvrtá je u tohoto typu rovnic splněna vždy. Ze třetí podmínky plyne, že jediným kandidátem na singulární řešení tohoto typu rovnic je parametr $p = 0$, tedy přímka $y = f(0)$. O singulární řešení se bude jednat tehdy, pokud bude platit druhá podmínka.

Příklad 3.2.1. Najdeme řešení rovnice $y = y' + (y')^2$.

Ruční postup by byl následný:

```
(%i2) depends(y, x)$
      DR: y = diff(y,x) + (diff(y,x))^2;
(DR)
```

$$y = \left(\frac{d}{dx}y\right)^2 + \frac{d}{dx}y$$

```
(%i5) depends(p, x)$
      SUB: p = diff(y,x)$
      DR2: subst(lhs(SUB), rhs(SUB), DR);
(DR2)
```

$$y = p^2 + p$$

```
(%i8) diff(%, x)$
      subst(lhs(SUB), rhs(SUB), %)$
      solve(%, diff(p,x))[1];
(%o8)
```

$$\frac{d}{dx}p = \frac{p}{2p+1}$$

Získanou rovnici se separovanými proměnnými vyřešíme pomocí příkazu `ode2` a vypíšeme celkové řešení v parametrickém tvaru pro proměnné x a y.

```
(%i10) OR: ode2(%, p, x);
       method;
(OR)
```

$$\log(p) + 2p = x + \%c$$

```
(%o10)
```

separable

```
(%i12) solve(OR, x)[1];
       DR2;
(%o11)
```

$$x = \log(p) + 2p - \%c$$

```
(%o12)
```

$$y = p^2 + p$$

Nakonec ověříme, že $f(p)$ je definována pro $p = 0$, takže i tato přímka bude řešením zadané rovnice. Protože $f_p(0) \neq 0$, nejedná se o singulární řešení.

```
(%i15) subst(0, p, DR2);
       diff(rhs(DR2), p)$
       subst(0, p, %);
(%o13)
```

$$y = 0$$

```
(%o15)
```

1

Při použití příkazu `ode_mm` dostaneme stejná řešení. Stejně tak pomocí dvojice příkazů `discurve` a `singular` zjistíme, že tato rovnice nemá singulární řešení.

```
(%i18) load("/home/mik/ode_mm.mac")$
       ode_mm(DR, y, x);
       method;
(%o17)
```

$$[x = \log(p) + 2p + \%c, y = p\,(p+1), y = 0]$$

```
(%o18)
```

$$yEqualsFunctionOfP$$

```
(%i20) discurve(DR, y, x);
       singular(DR, y, x);
(%o19)
```

$$[[p = -\frac{1}{2}, y = -\frac{1}{2^2}]]$$

```
(%o20)
```

$$[]$$

3.2.1 Zdrojový kód

Níže je uveden kód pomocné funkce, která řeší tento typ rovnic:

```
/*
Hleda reseni diferencialni rovnice prvniho radu nerozresene vzhledem
k derivaci typu y = f(y').
- eqn je zkoumana diferencialni rovnice
- y je zavisla promenna
- x je nezavisla promenna
*/
yEqFp(eqn,y,x) := block(
     [r,s,t,f_p,ans:[]],

     /*prevedeni y na levou stranu rovnice - na prave strane
     nesmi byt zadne x ani y, coz se jeste testuje nize*/
     r: solve(eqn,y),
     if (length(r)#1) then
          return(false),
     r: r[1],

     /*prevod do tvaru y = f(p), kde p = y'*/
     r: ratsubst(p,'diff(y,x),r),

     /*druha cast testu, zda je typu y = f(p), kde p = y'*/
     if (not(freeof(x,y,rhs(r)))) then
          return(false),
```

```
     /*nahrazeni novymi promennymi, ve kterych se bude pocitat*/
     r: newVars(r,y,x,p),

     /*jedno z reseni je p = 0, tj. y = f(0), kde p = y', ale jen
     v pripade, ze f(0) je definovana*/
     errormsg: false,
     t: errcatch(ratsubst(0,ppp,r)),
     errormsg: true,

     if (length(t)>0) then
          /*take musi platit, ze f_p(0) != 0 (to by se jednalo o
          singularni reseni)*/
          f_p: diff(rhs(r),ppp),
          f_p: ratsubst(0,ppp,f_p),
          if (not(is(equal(f_p,0)))) then
               ans: cons(t[1],ans),

     ans: cons(factor(radcan(r)),ans),

     depends(ppp,xxx),
     depends(yyy,xxx),

     s: diff(r,xxx),
     /*prevod do tvaru p = f'(p)*dp/dx*/
     s: ratsubst(ppp,diff(yyy,xxx),s),

     /*reseni rovnice se separovanymi promennymi
     dx/dp = f'(p)/p*/
     s: solve(s,diff(ppp,xxx))[1],
     s: xxx=integrate(1/(rhs(s)),ppp)+%c,
     ans: cons(factor(radcan(s)),ans),

     /*vraceni zpet do puvodnich promennych a zruseni zavislosti*/
     ans: oldVars(ans,y,x,p),

     method: 'yEqualsFunctionOfP,
     return(ans)
)$
```

3.3 Rovnice typu $x = f(y')$

Pro rovnice tvaru $x = f(y')$ je postup podobný jako u předchozího typu. Zavedením substituce $p = y'$, uvažováním parametru jakožto $p = p(y)$ a derivováním podle y dostaneme

$$\frac{\mathrm{d}x}{\mathrm{d}y} = \frac{1}{p} = f'(p)\frac{\mathrm{d}p}{\mathrm{d}y},$$

to je úpravou

$$\mathrm{d}y = pf'(p)\mathrm{d}p.$$

Opět se jedná o rovnici se separovanými proměnnými, zde s řešením

$$y = \int pf'(p)\mathrm{d}p + c, \quad c \in \mathbb{R}.$$

Celkové řešení je pak

$$x = f(p)$$
$$y = \int pf'(p)\mathrm{d}p + c.$$

Co se týče singulárních řešení tohoto typu rovnic, jedna z podmínek požaduje

$$F_y(x,y,p) \neq 0,$$

což v tomto případě ale není nikdy splněno, naopak parciální derivace podle y je vždy nulová, a tak zde žádná singulární řešení nemáme.

Příklad 3.3.1. Najdeme řešení rovnice $x = y' + \frac{1}{(y')^2} - 2x$.

Nejprve ručním řešením krok za krokem:

```
(%i2) depends(y, x)$
      DR: x = diff(y,x) + 1/(diff(y,x))^2 - 2*x;
(DR)
```

$$x = \frac{d}{dx}y + \frac{1}{\left(\frac{d}{dx}y\right)^2} - 2x$$

```
(%i6) depends(p, y)$
      SUB: p = diff(y,x)$
      subst(lhs(SUB), rhs(SUB), DR)$
      DR2: solve(%, x)[1];
(DR2)
```

$$x = \frac{p^3+1}{3p^2}$$

```
(%i8) depends(x, y)$
      DR3: diff(DR2,y);
(DR3)
```

$$\frac{d}{dy}x = \frac{d}{dy}p - \frac{2\left(p^3+1\right)\left(\frac{d}{dy}p\right)}{3p^3}$$

```
(%i11) SUB2: 1/p = diff(x,y)$
       subst(lhs(SUB2), rhs(SUB2), DR3)$
       DR4: solve(%, diff(p,y))[1];
(DR4)
```

$$\frac{d}{dy}p = \frac{3p^2}{p^3-2}$$

Tím jsme získali požadovanou rovnici se separovanými proměnnými. Příkazem `ode2` najdeme řešení pro y a vypíšeme i dříve vyjádřenou rovnost pro proměnnou x.

```
(%i15) DR2;
       ode2(DR4, p, y)$
       solve(%, y)[1];
       method;
(%o12)
```

$$x = \frac{p^3+1}{3p^2}$$

```
(%o14)
```

$$y = \frac{p^3 - 6\%cp + 4}{6p}$$

```
(%o15)
```

separable

Příkaz `ode_mm` aplikovaný přímo na rovnici ze zadání najde totéž řešení. Ověříme také, že žádná singulární řešení tato rovnice nemá.

```
(%i18) load("/home/mik/ode_mm.mac")$
       ode_mm(DR, y, x);
       method;
(%o17)
```

$$[x = \frac{(p+1)\left(p^2-p+1\right)}{3p^2}, y = \frac{p^3+6\%cp+4}{6p}]$$

```
(%o18)
```

xEqualsFunctionOfP

```
(%i20) discurve(DR, y, x);
       singular(DR, y, x);
(%o19)
```

$$[]$$

```
(%o20)
```

$$[]$$

3.3.1 Zdrojový kód

Níže je uveden kód pomocné funkce, která řeší tento typ rovnic:

```
/*
Hleda reseni diferencialni rovnice prvniho radu nerozresene vzhledem
k derivaci typu x = f(y').
- eqn je zkoumana diferencialni rovnice
- y je zavisla promenna
- x je nezavisla promenna
*/
xEqFp(eqn,y,x) := block(
     [r,s,ans:[]],

     /*prevedeni x na levou stranu rovnice - na prave strane
     nesmi byt zadne x ani y, coz se jeste testuje nize*/
     r: solve(eqn,x),
     if (length(r)#1) then
          return(false),
     r: r[1],

     /*prevod do tvaru x = f(p), kde p = y'*/
     r: ratsubst(p,'diff(y,x),r),

     /*druha cast testu, zda je typu x = f(p), kde p = y'*/
     if (not(freeof(x,y,rhs(r)))) then
          return(false),

     /*nahrazeni novymi promennymi, ve kterych se bude pocitat*/
     r: newVars(r,y,x,p),

     ans: cons(factor(radcan(r)),ans),

     depends(ppp,yyy),
     depends(xxx,yyy),
     s: diff(r,yyy),
     /*prevod do tvaru 1/p = f'(p)*dp/dy*/
     s: ratsubst(1/ppp,diff(xxx,yyy),s),

     /*reseni rovnice se separovanymi promennymi
     dy/dp = p*f'(p)*/
     s: solve(s,diff(ppp,yyy))[1],
     s: yyy=integrate(1/(rhs(s)),ppp)+%c,
     ans: endcons(factor(radcan(s)),ans),

     /*vraceni zpet do puvodnich promennych a zruseni zavislosti*/
```

```
     ans: oldVars(ans,y,x,p),

     method: 'xEqualsFunctionOfP,
     return(ans)
)$
```

3.4 Rovnice typu $y = f(x,y')$

Pokud řešíme rovnici typu $y = f(x,y')$, zavádíme parametr $p = y'$, který uvažujeme jako $p = p(x)$. Následně derivací podle x získáváme

$$p = f_x(x,p) + f_p(x,p)\frac{\mathrm{d}p}{\mathrm{d}x}.$$

Odtud dostáváme

$$\frac{\mathrm{d}x}{\mathrm{d}p} = \frac{f_p(x,p)}{p - f_x(x,p)},$$

což je diferenciální rovnice prvního řádu. Její řešení má tvar

$$x = \phi(p,c), \quad c \in \mathbb{R},$$

a celkové řešení máme tedy ve tvaru

$$x = \phi(p,c),$$

$$y = f(\phi(p,c),p).$$

Nalézt řešení $x = \phi(p,c)$ ovšem nemusí být obecně možné, což ilustruje následující příklad.

Příklad 3.4.1. Najdeme řešení rovnice $y = y'x - \frac{x^2}{2} + \frac{y'^2}{2}$.

Ruční postup by mohl být například následující:

```
(%i2) depends(y, x)$
      DR: y = diff(y,x)*x - x^2/2 + diff(y,x)^2/2;
(DR)
```

$$y = \frac{\left(\frac{d}{dx}y\right)^2}{2} + x\left(\frac{d}{dx}y\right) - \frac{x^2}{2}$$

```
(%i5) depends(p, x)$
      SUB: p = diff(y,x)$
      DR2: subst(lhs(SUB), rhs(SUB), DR);
(DR2)
```

$$y = -\frac{x^2}{2} + px + \frac{p^2}{2}$$

```
(%i7) diff(%, x)$
      subst(lhs(SUB), rhs(SUB), %);
(%o7)
```

$$p = \left(\frac{d}{dx}p\right)x - x + p\left(\frac{d}{dx}p\right) + p$$

```
(%i9) solve(%, diff(p,x))[1];
      DR3: 'diff(x,p) = 1/rhs(%);
(%o8)
```

$$\frac{d}{dx}p = \frac{x}{x+p}$$

```
(DR3)
```

$$\frac{d}{dp}x = \frac{x+p}{x}$$

Tím jsme získali rovnici v požadovaném tvaru podle postupu popsaného výše. Jedná se o homogenní diferenciální rovnici, kterou můžeme vyřešit pomocí příkazu `ode2`.

```
(%i11) RES: ode2(DR3, x, p);
       method;
(RES)
```

$$\%cp = \%e^{-\frac{\log\left(\frac{2x+\left(-\sqrt{5}-1\right)p}{2x+\left(\sqrt{5}-1\right)p}\right)+\sqrt{5}\log\left(\frac{x^2-px-p^2}{p^2}\right)}{2\sqrt{5}}}$$

```
(%o11)
```

homogeneous

Ani následující úpravy nám ale nepomohou ve vyjádření x.

```
(%i13) radcan(RES);
       solve(%, x);
(%o12)
```

$$\%cp = \frac{p\left(2x+\left(\sqrt{5}-1\right)p\right)^{\frac{1}{2\sqrt{5}}}}{\left(2x+\left(-\sqrt{5}-1\right)p\right)^{\frac{1}{2\sqrt{5}}}\sqrt{x^2-px-p^2}}$$

```
(%o13)
```

$$[\left(2x+\left(\sqrt{5}-1\right)p\right)^{\frac{1}{2\sqrt{5}}} = \%c\left(2x+\left(-\sqrt{5}-1\right)p\right)^{\frac{1}{2\sqrt{5}}}\sqrt{x^2-px-p^2}]$$

3.4.1 Lagrangeova rovnice

Řešení se nám naopak vždy podaří najít při řešení rovnic tvaru $y = g(y')x + h(y')$, totiž při řešení tzv. Lagrangeovy rovnice, již jsme zmínili v kapitole 1.1.1, jakožto příklad rovnice, kterou příkaz `ode2` obecně nedokáže vyřešit (kromě případů, kdy se díky tvaru funkcí g a h jedná zároveň o některou ze základních typů diferenciálních rovnic prvního řádu).

Při aplikaci obecného postupu pro rovnice $y = f(x,y')$, popsaného výše, totiž dostáváme

$$\frac{\mathrm{d}x}{\mathrm{d}p} = \frac{f_p(x,p)}{p - f_x(x,p)}.$$

Po zavedení parametru p do Lagrangeovy rovnice

$$y = f(x,p) = g(p)x + h(p)$$

jsme dále schopni dopočítat

$$f_x(x,p) = g(p),$$
$$f_p(x,p) = g'(p)x + h'(p)$$

a po dosazení a za předpokladu $p \neq g(p)$ získáváme lineární diferenciální rovnici, kterou dokážeme vyřešit:

$$\frac{\mathrm{d}x}{\mathrm{d}p} = \frac{g'(p)x + h'(p)}{p - g(p)} = \frac{g'(p)}{p - g(p)}x + \frac{h'(p)}{p - g(p)}.$$

Nakonec stejně jako v obecném případě získáme řešení ve tvaru

$$x = \phi(p,c), \quad c \in \mathbb{R},$$

$$y = f(\phi(p,c),p) = g(p)\phi(p,c) + h(p).$$

Případ $p = g(p)$, za předpokladu, že v některém bodě platí $c = g(c)$, vede na singulární řešení

$$y = g(c)x + h(c).$$

Ze čtyř podmínek pro singulární řešení totiž máme:

$$F(x,y,p) = y - g(p)x - h(p) = 0,$$

$$F_p(x,y,p) = -g'(p)x - h'(p) = 0,$$
$$F_x(x,y,p) + pF_y(x,y,p) = -g(p) + p \cdot 1 = 0,$$
$$F_y(x,y,p) = 1 \neq 0,$$

přičemž fakt, že $p = g(p)$ vede na singulární řešení plyne ze třetí podmínky.

Příklad 3.4.2. Vyřešíme Lagrangeovu rovnici $y = y'^2 - 2y' + 1 + x$.

Nejdříve ruční aplikací postupu:

```
(%i2) depends(y, x)$
      DR: y = diff(y,x)^2 - 2*diff(y,x) + 1 + x;
(DR)
```

$$y = \left(\frac{d}{dx}y\right)^2 - 2\left(\frac{d}{dx}y\right) + x + 1$$

```
(%i5) depends(p, x)$
      SUB: p = diff(y,x)$
      DR2: subst(lhs(SUB), rhs(SUB), DR);
(DR2)
```

$$y = x + p^2 - 2p + 1$$

```
(%i8) diff(%, x)$
      subst(lhs(SUB), rhs(SUB), %)$
      solve(%, diff(p,x))[1];
(%o8)
```

$$\frac{d}{dx}p = \frac{1}{2}$$

```
(%i9) DR3: 'diff(x,p) = 1/rhs(%);
(DR3)
```

$$\frac{d}{dp}x = 2$$

Nyní máme lineární rovnici prvního řádu (v tomto příkladu dokonce rovnici se separovanými proměnnými) v závislé proměnné x, kterou můžeme vyřešit příkazem `ode2`. Spočtenou hodnotu x závislou na parametru p a integrační konstantě dosadíme do rovnice `DR2` a vypíšeme řešení v parametrickém tvaru pro obě proměnné x i y.

```
(%i12) RES_X: expand(ode2(DR3, x, p));
       RES_Y: expand(subst(rhs(%), lhs(%), DR2));
       method;
(RES_X)
```

$$x = 2p + \%c$$

```
(RES_Y)
```

$$y = p^2 + \%c + 1$$

```
(%o12)
```

linear

V tomto příkladu se nám podaří vyjádřit řešení i v explicitním tvaru.

```
(%i14) solve(RES_X, p)[1]$
       subst(rhs(%), lhs(%), RES_Y);
(%o14)
```

$$y = \frac{(x - \%c)^2}{4} + \%c + 1$$

Pokusíme se nalézt i diskriminantní křivky a singulární řešení. Aby bylo možné spočítat potřebné parciální derivace, zrušíme veškeré dříve nastavené závislosti mezi proměnnými příkazem `remove`. Následně si parciální derivace předpočítáme a nalezneme diskriminantní křivky.

```
(%i20) remove(y,dependency)$
       remove(p,dependency)$
       F: lhs(DR2) - rhs(DR2);
       Fx: diff(F, x);
       Fy: diff(F, y);
       Fp: diff(F, p);
(F)
```

$$y - x - p^2 + 2p - 1$$

```
(Fx)
```

$$-1$$

```
(Fy)
```

$$1$$

```
(Fp)
```

$$2 - 2p$$

```
(%i21) DK: solve([F=0,Fp=0], [p,y]);
(DK)
```

$$[[p = 1, y = x]]$$

Při ověření dalších dvou podmínek na singulární řešení, které jsou kladeny oproti diskriminantním křivkám, zjistíme, že jediný kandidát z diskriminantních křivek $y = x$ je i singulárním řešením. $F_y \neq 0$, což jsme viděli výše, a zbývá ověřit poslední podmínku $F_x + pF_y = 0$, která bude také platit.

```
(%i22) Fx + p*Fy;
(%o22)
```

$$p - 1$$

Při použití příkazu `ode_mm` dostaneme totožný výsledek.

```
(%i25) load("/home/mik/ode_mm.mac")$
       ode_mm(DR, y, x);
       method;
(%o24)
```

$$[x = 2p + \%c, y = p^2 + \%c + 1]$$

```
(%o25)
```

$$yEqualsFunctionOfXandP$$

```
(%i27) discurve(DR, y, x);
       singular(DR, y, x);
(%o26)
```

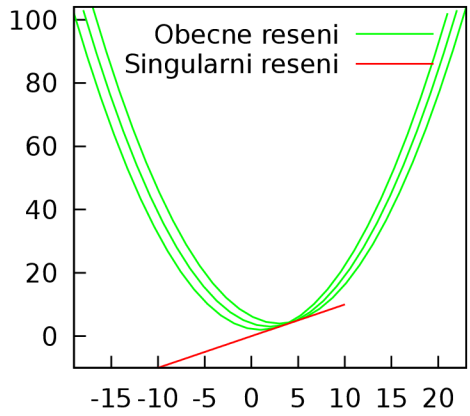
$$[[p = 1, y = x]]$$

```
(%o27)
```

$$[y = x]$$

```
(%i28) draw_ode_mm(DR, y, x)$
```

(viz obrázek 3.1)



Obrázek 3.1: Obecné a singulární řešení Lagrangeovy rovnice $y = y'^2 - 2y' + 1 + x$.

3.4.2 Clairautova rovnice

Speciálním případem Lagrangeovy rovnice $y = g(y')x + h(y')$ je Clairautova rovnice tvaru $y = y'x + h(y')$, tedy případ, kdy $g(y') \equiv y'$.

Ta se opět podobně řeší zavedením substituce $p = y'$, to je převedením zadané rovnice na

$$y = px + g(p)$$

a dále zderivováním podle x na

$$p = p'x + p + g'(p)p',$$

čímž se zbavíme y. Po odečtení p z obou stran máme $0 = p'(x + g'(p))$ a řešením je

$$p' = 0 \vee x + g'(p) = 0.$$

V prvním případě platí, že $p = c, c \in \mathbb{R}$, a tedy dosazením získáme řešení

$$y = cx + g(c).$$

Ve druhém dostáváme singulární řešení zadané parametricky

$$x = -g'(p),$$

$$y = p(-g'(p)) + g(p).$$

Z podmínek na singulární řešení totiž dostáváme:

$$F(x,y,p) = y - px - g(p) = 0,$$

$$F_p(x,y,p) = -x - g'(p) = 0,$$

$$F_x(x,y,p) + pF_y(x,y,p) = -p + p \cdot 1 = 0,$$

$$F_y(x,y,p) = 1 \neq 0.$$

Třetí a čtvrtá podmínka jsou splněny vždy, a tak zbývají pouze podmínky pro diskriminantní křivky. Pro Clairautovu rovnici tedy platí, že každá diskriminantní křivka je zároveň singulárním řešením.

Příklad 3.4.3. Vyřešíme diferenciální rovnici $y = xy' + y' + y'^2$, kterou jsme bez úspěchu zkusili v kapitole 1.1.1 vyřešit příkazem `ode2`.

Ručně bychom postupovali následně:

```
(%i2) depends(y, x)$
      DR: y = x*diff(y,x) + diff(y,x) + diff(y,x)^2;
(DR)
```

$$y = \left(\frac{d}{dx}y\right)^2 + x\left(\frac{d}{dx}y\right) + \frac{d}{dx}y$$

```
(%i5) depends(p, x)$
      SUB: p = diff(y,x)$
      DR2: subst(lhs(SUB), rhs(SUB), DR);
(DR2)
```

$$y = px + p^2 + p$$

```
(%i8) diff(%, x)$
      subst(lhs(SUB), rhs(SUB), %)$
      DR3: factor(% - p);
(DR3)
```

$$0 = \left(\frac{d}{dx}p\right)(x + 2p + 1)$$

```
(%i10) RES1_X: solve(DR3, x)[1];
       RES1_Y: expand(subst(RES1_X, DR2));
(RES1_X)
```

$$x = -2p - 1$$

```
(RES1_Y)
```

$$y = -p^2$$

Jako první řešení rovnice `DR3` nám příkaz `solve` do seznamu uložil řešení pro druhou závorku rovnu nule, tedy singulární řešení $x = -g'(p)$. V tomto případě můžeme spočítat i řešení v explicitním tvaru.

```
(%i12) solve(RES1_X, p)[1]$
       RES1: subst(%, RES1_Y);
(RES1)
```

$$y = -\frac{(x+1)^2}{4}$$

Jako druhé řešení rovnice `DR3` dostaneme obecné řešení $p = c$.

```
(%i14) solve(DR3, x)[2];
       RES2: subst(p=%c, DR2);
(%o13)
```

$$\frac{d}{dx}p = 0$$

```
(RES2)
```

$$y = \%c\,x + \%c^2 + \%c$$

Příkaz `ode_mm` aplikovaný na neupravenou rovnici ze zadání nalezne totéž obecné i singulární řešení.

```
(%i17) load("/home/mik/ode_mm.mac")$
       ode_mm(DR, y, x);
       method;
(%o16)
```

$$[x = p, y = \%c\ (p + \%c + 1)]$$

```
(%o17)
```

Clairaut

```
(%i19) discurve(DR, y, x);
       singular(DR, y, x);
(%o18)
```

$$[[p = -\frac{x+1}{2}, y = -\frac{(x+1)^2}{4}]]$$

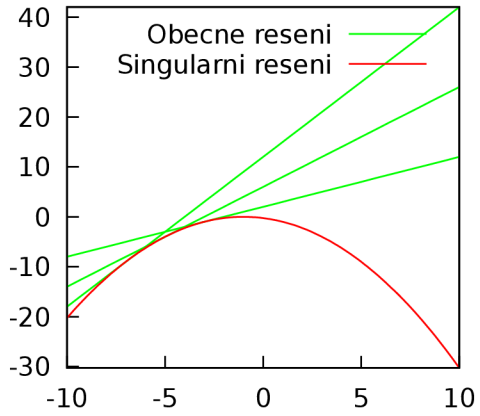
```
(%o19)
```

$$[y = -\frac{(x+1)^2}{4}]$$

Nakonec si můžeme řešení zobrazit do jednoho grafu. Uvidíme v něm dobře i vlastnost řešení Clairautovy rovnice, totiž že obecné řešení, procházející libovolně zvoleným bodem singulárního řešení, představuje vždy tečnu singulárního řešení v tomto bodě.

```
(%i20) draw_ode_mm(DR, y, x)$
```

(viz obrázek 3.2)



Obrázek 3.2: Obecné a singulární řešení Clairautovy rovnice $y = xy' + y' + y'^2$.

3.4.3 Zdrojový kód

Níže jsou uvedeny kódy dvou pomocných funkcí, které řeší tento typ rovnic:

```
/*
Hleda reseni diferencialni rovnice prvniho radu nerozresene vzhledem
k derivaci typu y = f(x,y').
- eqn je zkoumana diferencialni rovnice
- y je zavisla promenna
- x je nezavisla promenna
*/
yEqFxp(eqn,y,x) := block(
     [r,s,ans:[]],

     /*prevedeni y na levou stranu rovnice - na prave strane
     nesmi zadne y byt, coz se jeste testuje nize*/
     r: solve(eqn,y),
     if (length(r)#1) then
          return(false),
     r: r[1],

     /*prevod do tvaru y = f(x,p), kde p = y'*/
     r: ratsubst(p,'diff(y,x),r),

     /*druha cast testu, zda je typu y = f(x,p), kde p = y'*/
     if (not(freeof(y,rhs(r)))) then
          return(false),

     if (freeof(x,rhs(r)-p*x)) then (
          return(Clairaut(r))
     ),

     /*nahrazeni novymi promennymi, ve kterych se bude pocitat*/
     r: newVars(r,y,x,p),

     depends(ppp,xxx),
     depends(yyy,xxx),
     s: diff(r,xxx),
     /*prevod do tvaru p = f_x(x,p) + f_p(x,p)*dp/dx*/
     s: ratsubst(ppp,diff(yyy,xxx),s),

     /*reseni rovnice dx/dp = (f_p(x,p)) / (p - f_x(x,p))*/
     s: solve(s,diff(ppp,xxx))[1],
     s: 'diff(xxx,ppp) = rhs(s^(-1)),
     s: ode2(s,xxx,ppp),
     /*v pripade, ze ode2 nenajde reseni predchozi rovnice, cela
```

```
     tato procedura selhava*/
     if (is(equal(s,false))) then
          return(false),

     r: ratsubst(rhs(s),xxx,r),
     ans: cons(factor(radcan(r)),ans),
     ans: cons(factor(radcan(s)),ans),

     /*vraceni zpet do puvodnich promennych a zruseni zavislosti*/
     ans: oldVars(ans,y,x,p),

     method: 'yEqualsFunctionOfXandP,
     return(ans)
)$


/*
Hleda obecne reseni Clairautovy diferencialni rovnice. Vraci jej jako
ostatni funkce v parametrickem tvaru.
- eqn je zkoumana Clairautova diferencialni rovnice tvaru
  y = px + h(p)
*/
Clairaut(eqn) := block(
     [s,ans:[]],

     s: ratsubst(%c,p,eqn),
     s: ratsubst(p,x,s),
     ans: cons(factor(radcan(s)),ans),
     ans: cons(x=p,ans),

     method: 'Clairaut,
     return(ans)
)$
```

3.5 Rovnice typu $x = f(y, y')$

U rovnic typu $x = f(y,y')$ je postup opět analogický. Zavedením parametru $p = y'$, který uvažujeme jako $p = p(y)$, a derivováním podle y dostáváme z původní rovnice

$$\frac{1}{p} = f_y(y,p) + f_p(y,p)\frac{\mathrm{d}p}{\mathrm{d}y}.$$

Odtud můžeme vyjádřit

$$\frac{\mathrm{d}y}{\mathrm{d}p} = \frac{pf_p(y,p)}{1 - pf_y(y,p)},$$

což je diferenciální rovnice 1. řádu, mající řešení

$$y = \phi(p,c), \quad c \in \mathbb{R}.$$

Celkové řešení je pak tvaru

$$x = f(\phi(p,c),p)$$
$$y = \phi(p,c).$$

Když dosadíme do podmínek pro singulární řešení, máme:

$$F(x,y,p) = x - f(y,p) = 0,$$

$$F_p(x,y,p) = -f_p(y,p) = 0,$$
$$F_x(x,y,p) + pF_y(x,y,p) = 1 - pf_y(y,p) = 0,$$
$$F_y(x,y,p) = -f_y(y,p) \neq 0.$$

Ze třetí rovnice vidíme, že podmínka $1 - pf_y(y,p) = 0$, kterou musíme při hledání obecného řešení popsaného výše vyloučit, představuje právě kandidáta na singulární řešení.

Analogicky jako u obecného tvaru rovnice $y = f(x,y')$ nelze ani zde tvrdit, že nalezení funkce ϕ v rovnosti $y = \phi(p,c)$ bude vždy možné. V následujícím příkladu budeme ale řešit rovnici, ve které tuto funkci najdeme.

Příklad 3.5.1. Vyřešíme rovnici $y'^3 - 4xyy' + 8y^2 = 0$.

Nejdříve ručním postupem krok za krokem.

```
(%i2) depends(y, x)$
      DR: diff(y,x)^3 - 4*x*y*diff(y,x) + 8*y^2 = 0;
(DR)
```

$$\left(\frac{d}{dx}y\right)^3 - 4xy\left(\frac{d}{dx}y\right) + 8y^2 = 0$$

```
(%i6) depends(p, y)$
      SUB: p = diff(y,x)$
      solve(DR, x)[1]$
      DR2: subst(lhs(SUB), rhs(SUB), %);
(DR2)
```

$$x = \frac{8y^2 + p^3}{4py}$$

```
(%i11) depends(x, y)$
       SUB2: 1/p = diff(x,y)$
       diff(DR2, y)$
       subst(lhs(SUB2), rhs(SUB2), %)$
       DR3: solve(%, diff(p,y))[1];
(DR3)
```

$$\frac{d}{dy}p = \frac{p}{2y}$$

Získali jsme požadovanou diferenciální rovnici prvního řádu, kterou vyřešíme pomocí `ode2`. Výsledek musíme ještě upravit ručně, abychom získali vyjádření y. Na otázku Maximy, zda je cp kladné, záporné nebo rovno nule, odpovíme, že kladné, aby dokázala vyjádřit y explicitně.

```
(%i14) radcan(ode2(DR3, p, y));
       RES_Y: solve(%, y)[1];
       method;
(%o12)
```

$$p = \%c\sqrt{y}$$

```
"Is %c p positive, negative or zero?"pos;
(RES_Y)
```

$$y = \frac{p^2}{\%c^2}$$

```
(%o14)
```

$$linear$$

```
(%i15) factor(subst(RES_Y, DR2));
(%o15)
```

$$x = \frac{8p + \%c^4}{4\%c^2}$$

Dosazením do `DR2` jsme získali vyjádření i pro x. Dále se pokusíme nalézt diskriminantní křivky a singulární řešení. Aby bylo možné spočítat parciální derivace, zrušíme všechny dříve nastavené závislosti mezi proměnnými.

```
(%i22) remove(x,dependency)$
       remove(y,dependency)$
       remove(p,dependency)$
       F: subst(lhs(SUB),rhs(SUB),lhs(DR));
       Fx: factor(diff(F, x));
       Fy: factor(diff(F, y));
       Fp: factor(diff(F, p));
(F)
```

$$8y^2 - 4pxy + p^3$$

```
(Fx)
```

$$-4py$$

```
(Fy)
```

$$4\left(4y - px\right)$$

```
(Fp)
```

$$-\left(4xy - 3p^2\right)$$

```
(%i23) DK: solve([F=0,Fp=0], [p,y]);
(DK)
```

$$[[p = \frac{4x^2}{9}, y = \frac{4x^3}{27}], [p = 0, y = 0]]$$

Dosazením do dvou podmínek, které jsou navíc kladeny na singulární řešení oproti diskriminantním křivkám, zjistíme, že jedna křivka je a druhá není singulárním řešením zadané rovnice (neboť porušila $F_y \neq 0$).

```
(%i25) subst(DK[1], Fx+p*Fy);
       subst(DK[1], Fy);
(%o24)
```

$$0$$

```
(%o25)
```

$$\frac{16x^3}{27}$$

```
(%i27) subst(DK[2], Fx+p*Fy);
       subst(DK[2], Fy);
(%o26)
```

$$0$$

```
(%o27)
```

$$0$$

Jediným singulárním řešením je tedy $y = \frac{4x^3}{27}$. Použitím příkazů z balíčku `ode_mm` dostaneme totožné výsledky. Pouze s vizuálním rozdílem, kdy mezi integračními konstantami výsledku ručního postupu (R) a výsledku příkazu `ode_mm` (P) platí vztah $\frac{1}{c_R^2} = c_P$.

```
(%i30) load("/home/mik/ode_mm.mac")$
       ode_mm(DR, y, x);
       method;
(%o29)
```

$$[x = \frac{8\%c^2 p + 1}{4\%c}, y = \%c\, p^2]$$

```
(%o30)
```

xEqualsFunctionOfYandP

```
(%i32) discurve(DR, y, x);
       singular(DR, y, x);
(%o31)
```

$$[[p = \frac{4x^2}{9}, y = \frac{4x^3}{27}], [p = 0, y = 0]]$$

```
(%o32)
```

$$[y = \frac{4x^3}{27}]$$

3.5.1 Zdrojový kód

Níže je uveden kód pomocné funkce, která řeší tento typ rovnic:

```
/*
Hleda reseni diferencialni rovnice prvniho radu nerozresene vzhledem
k derivaci typu x = f(x,y').
- eqn je zkoumana diferencialni rovnice
- y je zavisla promenna
- x je nezavisla promenna
*/
xEqFyp(eqn,y,x) := block(
     [r,s,ans:[]],

     /*prevedeni x na levou stranu rovnice - na prave strane
     nesmi zadne x byt, coz se jeste testuje nize*/
     r: solve(eqn,x),
     if (length(r)#1) then
          return(false),
     r: r[1],

     /*prevod do tvaru x = f(y,p), kde p = y'*/
     r: ratsubst(p,'diff(y,x),r),

     /*druha cast testu, zda je typu x = f(y,p), kde p = y'*/
     if (not(freeof(x,rhs(r)))) then
          return(false),

     /*nahrazeni novymi promennymi, ve kterych se bude pocitat*/
     r: newVars(r,y,x,p),
```

```
    depends(ppp,yyy),
    depends(xxx,yyy),
    s: diff(r,yyy),
    /*prevod do tvaru 1/p - f_y(y,p) = f_p(y,p)*dp/dy*/
    s: ratsubst(1/ppp,diff(xxx,yyy),s),

    /*reseni rovnice dy/dp = (p*f_p(y,p)) / (1 - p*f_y(y,p))*/
    s: solve(s,diff(ppp,yyy))[1],
    s: 'diff(yyy,ppp) = rhs(s^(-1)),
    s: ode2(s,yyy,ppp),
    /*v pripade, ze ode2 nenajde reseni predchozi rovnice, cela
    tato procedura selhava*/
    if (is(equal(s,false))) then
        return(false),

    r: ratsubst(rhs(s),yyy,r),
    ans: cons(factor(radcan(s)),ans),
    ans: cons(factor(radcan(r)),ans),

    /*vraceni zpet do puvodnich promennych a zruseni zavislosti*/
    ans: oldVars(ans,y,x,p),

    method: 'xEqualsFunctionOfYandP,
    return(ans)
)$
```

3.6 Tabulka příkazů

`ode_mm(difRce, y, x)`	obecné řešení dané DR
`discurve(difRce, y, x)`	diskriminantní křivky
`singular(difRce, y, x)`	singulární řešení
`draw_ode_mm(difRce, y, x)`	graf s obecným a singulárním řešením
`isogonal(rce, y, x, c, uhel)`	izogonální trajektorie daného systému křivek, `c` je konstanta

pro všechny příkazy platí:

`difRce`	DR 1. řádu nerozřešená vzhledem k derivaci
`y`	závislá proměnná
`x`	nezávislá proměnná

pomocné proměnné nastavované procedurami

`method`	metoda použitá v hlavním příkazu (`ode_mm`)
`traj_ode`	DR trajektorie systému (příkaz `isogonal`)

Tabulka 3.1: Základní příkazy balíčku `ode_mm`, jejich parametry a pomocné proměnné.

Závěr

Systém počítačové algebry Maxima v současné době nabízí široké možnosti použití v oblasti diferenciálních rovnic. Díky tomu, že poskytuje hned několik příkazů k jejich řešení (ať už analytickému či numerickému) i následné vizualizaci, si může uživatel zvolit, který příkaz a které volitelné parametry budou pro řešení daného problému nejvhodnější. V budoucnu můžeme od vývojářů navíc čekat další vylepšování příkazů a procedur, a tak se pole řešených úloh ještě zvětší.

Přínos této práce spočívá zaprvé v popisu všech aktuálně dostupných příkazů Maximy týkajících se diferenciálních rovnic a jejich možností v této oblasti. Na vhodně zvolených příkladech bylo řešení rovnic ukázáno vždy jak s využitím těchto příkazů, tak i řešením krok za krokem pomocí elementárních obecnějších příkazů. Kromě typů rovnic, které příkazy dokáží vyřešit, byly uvedeny i příklady takových, které příkazy buď nedokáží řešit vůbec, nebo které je potřeba nejprve převést do jiného tvaru. Druhým přínosem je balíček funkcí, napsaný v rámci této práce, sloužící k řešení diferenciálních rovnic nerozřešených vzhledem k derivaci. Nabízí procedury na hledání řešení čtyř základních typů těchto rovnic, jejich diskriminantních křivek, singulárních řešení a k vykreslení přehledného grafu obsahujícího obecná a singulární řešení.

Všechny zápisníky wxMaximy (ve formátu `.wxm`, očíslované podle podkapitol, ve kterých v nich řešené příklady vyskytly), balíček `ode_mm` (`.mac`), stejně jako tento samotný text práce v elektronické podobě (`.pdf`) jsou k nalezení na přiloženém CD a dostupné na stránce https://is.muni.cz/www/394225/DP/index.html.

Při psaní diplomové práce jsem použil Maximu ve verzi 5.40.0 (grafické uživatelské rozhraní ve verzi 16.12.2) v operačním systému Ubuntu 17.10.

Výstupy wxMaximy jsou neupravované, sázené tak, jak je exportuje do LaTeXu, až na výjimky, kdy byly nezbytné drobné úpravy, aby bylo možné TeXový soubor přeložit.

Seznam použité literatury

[1] KALAS, Josef a Miloš RÁB. *Obyčejné diferenciální rovnice*. Vyd. 3. Brno: Masarykova univerzita, 2012. ISBN 978-80-210-5815-6.

[2] RÁB, Miloš. *Metody řešení obyčejných diferenciálních rovnic*. 3. vyd. Brno: Masarykova univerzita, 2004. ISBN 80-210-3416-5.

[3] PLCH, Roman. *Příklady z matematické analýzy: diferenciální rovnice*. Brno: Masarykova univerzita, 2002. ISBN 80-210-2806-8.

[4] ŠREMR, Jiří. *Geometrické aplikace diferenciálních rovnic*. Brno, 1999. Diplomová práce. Masarykova univerzita. Vedoucí práce Jaromír Vosmanský.

[5] *Maxima 5.39.0 Manual* [online]. 2016 [cit. 2017-05-12]. Dostupné z: http://maxima.sourceforge.net/docs/manual/maxima.html.

[6] WOOLLETT, Edwin L. *Maxima by Example Ch. 3, Ordinary Differential Equation Tools* [online]. Long Beach, Kalifornie, 2010 [cit. 2017-04-16]. Dostupné z: https://web.csulb.edu/ woollett/mbe3ode1.pdf.

[7] DE SOUZA, Paulo Ney, Richard J. FATEMAN, Joel MOSES a Cliff YAPP. *The Maxima Book* [online]. 2004 [cit. 2017-05-10]. Dostupné z: http://maxima.sourceforge.net/docs/maximabook/maximabook-19-Sept-2004.pdf.

[8] LIN, Cheng-Ren. *MAXIMA & ODE2* [online]. Kaohsiung, Taiwan, 2004 [cit. 2017-02-16]. Dostupné z: http://web.idv.nkmu.edu.tw/ crlin/Maxima-ode.pdf.

[9] LIN, Cheng-Ren. *MAXIMA & Laplace Transform* [online]. Kaohsiung, Taiwan, 2004 [cit. 2017-03-06]. Dostupné z: http://web.idv.nkmu.edu.tw/ crlin/Maxima-lpt.pdf.

[10] *Maxima Tutorial: Differential Equations - Symbolic Solutions* [online]. 2005 [cit. 2017-03-05]. Dostupné z: http://maxima.sourceforge.net/docs/tutorial/en/gaertner-tutorial-revision/Pages/ODE0002.htm.

[11] ČERMÁK, Libor. *Numerické metody II*. Brno: Akademické nakladatelství CERM, 2004. ISBN 80-214-2722-1.

[12] *Differential Equations - Phase Plane*. Paul's Online Math Notes [online]. Beaumont, Texas [cit. 2017-04-25]. Dostupné z: http://tutorial.math.lamar.edu/Classes/DE/PhasePlane.aspx.

[13] *Bessel functions*. Encyclopedia of Mathematics [online] 2016 [cit. 2017-05-01]. Dostupné z: http://www.encyclopediaofmath.org/index.php?title=Bessel_functions.

[14] *Neumann function*. Encyclopedia of Mathematics [online] 2014 [cit. 2017-05-01]. Dostupné z: http://www.encyclopediaofmath.org/index.php?title=Neumann_function.