Jedná se o rozpracovaný návod k programu wxmaxima pro naprosté začátečníky. Návod lze libovolně kopírovat a používat ke komerčním i osobním účelům. Momentálně chybí mnoho důležitých kapitol které budou postupně doplňovány, pokud bude mít kdokoliv chuť, může samostatně vytvořit nějakou kapitolu a tu poslat na mou adresu, rád ji přidám. Jedinou odměnou vám může být zvěčnění vašeho jména u dané kapiloly. Maximě zdar a praktickému využití zvlášť, Michal Sivčák(www.kmp.tul.cz)

NEWS:

- 09.04.2019 - doplněna lineární algebra .
- 13.03.2017 - doplněn příklad z dynamiky bodu - numerické řešení dif.rovnic.
- 19.08.2015 - aktualizace na novější verze.
- 02.12.2010 - doplněn zápis konstant Pi, e, i, phi. - doplněn popis parametrů pro vykreslení grafů.
- 26.11.2010 - vydána první nekompletní příručka.

# Úvod do programu MAXIMA

## 1 Základní operátory a funkce

Na začátek každého programu doporučuji napsat příkaz "kill(all)$" který vymaže z paměti předchozí výsledky. Pro zpracování (výpočet) "vstupního pole"slouží kombinace kláves "shift + enter" a pro celý sešit "shift + ctrl + R"."Enter" slouží pouze pro odsazení odstavce. Vlastnosti "shift + enter" a "enter" lze prohodit v nastavení programu. Maxima ignoruje vícenásobné mezery a odsazení odstavců. Program se píše do "vstupního pole" (automaticky uvozeno znakem %i, jeho zobrazení lze zrušit v nastavení programu). Komentáře se zapisují do "textových polí" (textová pole mají různé úrovně s jejichž pomocí lze efektivně formátovat dokument). Zpracována jsou pouze "vstupní pole" a to postupně zhora dolů ("shift + ctrl +R"). Vstup je ukončen středníkem ";" pokud nás zajímá výsledek, nebo dolarem "$" pokud výsledek nechceme zobrazit (např. je velmi dlouhý nebo nezajímavý). Jednotlivé vstupy lze vyhodnocovat opakovaně ("shift + Enter").

**(% i1)** kill(all)$

Přiřazení se v Maximě provádí pomocí dvojtečky, pro a=5 a b=4 bude vypadat zápis následovně:

**(% i2)** a:5; b:4;

(a) 5

(b) 4

Součet, rozdíl, součin a podíl

**(% i3)** a+b;

() 9

**(% i4)** a-b;

() 1

**(% i5)** a*b;

() 20

2

(% i6) a/b;

() $\frac{5}{4}$

Chceme li výstup zobrazit ve formátu desetiného čísla, použijeme příkaz"float".

(% i7) float(a/b);

() 1.25

Mocnina a druhá odmocnina.

(% i8) a^2;

() 25

(% i9) sqrt(a);

() $\sqrt{5}$

Zápis goniometrických funkcí a funkcí k nim inverzních. Maxima, tak jako většina matematických programů, používá výhradně radiány!!

(% i13) sin(a);cos(b);tan(a);cot(b);

() $\sin(5)$

() $\cos(4)$

() $\tan(5)$

() $\cot(4)$

(% i17) asin(1);acos(1);atan(1);acot(1);

() $\frac{\pi}{2}$

() 0

() $\frac{\pi}{4}$

() $\frac{\pi}{4}$

Použití základních matematických konstant - Ludolfovo číslo "Pi", základ přirozeného logaritmu "e", imaginární jednotka "i", zlatý řez "phi". Výše uvedené znaky ("%pi", "%e", "%i" a "%phi") nepoužívejte pro označení jiných hodnot.

**(% i21)** %pi;%e; %i; %phi;

() $\pi$

() $e$

() $i$

() $\phi$

**(% i25)** float(%pi);float(%e); float(%i); float(%phi);

() 3.141592653589793

() 2.718281828459045

() $i$

() 1.618033988749895

(% i26) log(%e);

() 1

Jak je zřejmé, funkce "log" je přirozený logaritmus.

(% i27) %i^2;

() $-1$

(% i28) float(1/%phi-%phi);

() $-1.0$

## 2 Přiřazení výrazu, rovnice a definice funkce.

Pro další práci je nutné pochopit rozdíl mezi přiřazením, funkcí a rovnicí. Přiřazení je pojmenování výrazu pomocí nového názvu. Funkce je program nebo procedura jejíž výstup je závislý na vstupních hodnotách.

(% i29) kill(all)$

(% i2) a:5; b:10;

(a) 5

(b) 10

(% i3) prirazeni:a+b;

(prirazeni) 15

Syntaxe pro zápis (definici) funkce je následující: "název funkce(vstupní proměnná 1, vstupní proměnná 2, ...):=funkční závislost;" Funkce se tedy definuje pomocí znaku ":=". Hodnotu funkce dostaneme pokud napíšeme "název funkce(vstup 1, vstup 2, ...);", kde vstupní proměnná je hodnota (nebo obor hodnot, např. čas) ve které nás velikost funkce zajímá.

(% i4) soucet(a,b):=a+b;

() $\mathrm{soucet}\,(a,b) := a + b$

(% i5) soucet(a,b);

() 15

Pokud chceme zrušit přiřazení (zapomenout) a=5 a b=4, použijeme příkaz "kill(a,b)"

(% i6) kill(a,b);

() *done*

(% i7) soucet(a,b);

() $b + a$

Rovnici lze definovat jako výraz, kde levá strana (lhs) je od pravé strany (rhs) oddělena znakem "=" a zároveň platí, že levá strana se rovná pravé. Pokud je levá (pravá) strana rovna nule, lze nulovou staranu včetně znaku "=" vynechat. Definujme soustavu rovnic "a+b=10" a "a+2b-13=0".

(% i9) rovnice_1:a+b=10; rovnice_2:a+2*b-13;

(rovnice_ 1) $b + a = 10$

(rovnice_ 2) $2b + a - 13$

(% i10) lhs(rovnice_1);

() $b + a$

(% i11) rhs(rovnice_1);

() 10

Pro vyřešení této soustavy dvou lineárních rovnic o dvou neznámých, použijeme příkaz "linsolve ([rovnice_1, ..., rovnice_n], [neznámá_1, ...,neznámá_n])". Počet rovnic musí být samozřejmě stejný jako počet neznámých. Nejprve je nutné nastavit proměnnou "globalsolve" z hodnoty "false" na "true" (Maxima potom automaticky přiřadí hledaným neznámým vypočítané hodnoty, pokud existuje více možných řešení Maxima výsledek nepřiřadí a správnou hodnotu je třeba přiřadit ručně). Bohužel velmi často i přes správně nastavené "globalsolve" Maxima jednoznačný výsledek nepřiřadí - poznáme to tak, že ve výsledku je místo znaku pro přiřazení":" znak pro rovnost "=". Tato chyba se projevuje

náhodně (obvykle u příkazu pro počítání s nelineárními rovnicemi "solve") a nepodařilo se mi ji nijak opravit. I v tomto případě je tedy nutné opět výsledky přiřadit ručně. V našem případě tedy:

**(% i13)** globalsolve: true; linsolve([rovnice_1,rovnice_2],[a,b]);

(globalsolve) true

() $[a : 7, b : 3]$

**(% i14)** a;

() 7

**(% i15)** rovnice_3:a*x^2+b*x-a;

(rovnice_ 3) $7x^2 + 3x - 7$

**(% i16)** reseni_kvadr_fce:solve(rovnice_3,x);

(reseni_ kvadr_ fce) $[x = -\frac{\sqrt{205}+3}{14}, x = \frac{\sqrt{205}-3}{14}]$

Je zřejmé, že kvadratická rovnice "rovnice_3" má dvě řešení. Tato řešení jsme pojmenovali jako "reseni_kvadr_fce". "reseni_kvadr_fce" je pole o dvou prvcích, kde jednotlivé prvky jsou rovnice. První řešení je na pravé straně rovnice v prvním prvku a druhé řešení na pravé straně rovnice v druhém prvku. Výběr pravé (levé) strany provedeme pomocí příkazu "rhs" ("lhs").

**(% i18)** x_1:rhs(reseni_kvadr_fce[1]); x_2:rhs(reseni_kvadr_fce[2]);

(x_ 1) $-\frac{\sqrt{205}+3}{14}$

(x_ 2) $\frac{\sqrt{205}-3}{14}$

**(% i19)** x_1;

() $-\frac{\sqrt{205}+3}{14}$

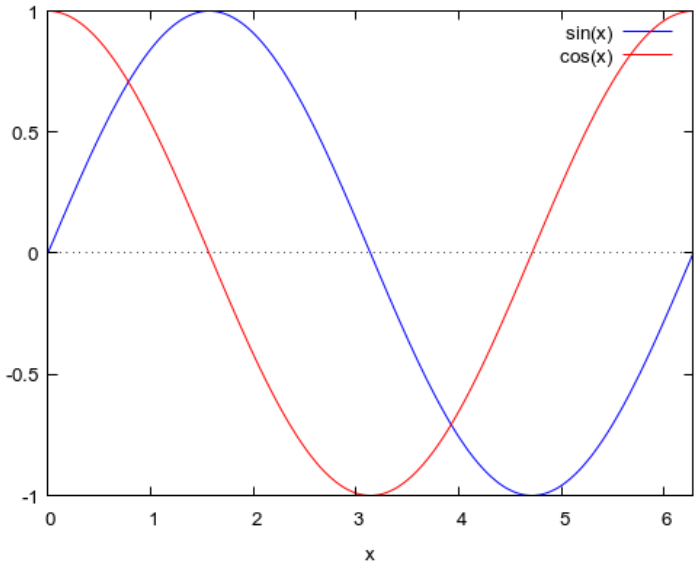
# 3 Kreslení grafů

(% i20) kill(all)$

Pro vykreslení 2D nebo 3D grafů slouží příkaz "plot2d" resp. "plot3d". Zpracováním těchto příkazů otevře Maxima nové okno ve kterém vykreslí požadovaný graf. Tyto grafy jsou interaktivní a lze s nimi dále pracovat. Chceme-li, aby byl graf součástí dokumentu, použijeme příkaz "wxplot2d" resp."wxplot3d". Maxima na základě těchto příkazů vytvoří v dočasné složce obrázek a ten vloží do dokumentu pod zadaný příkaz. Základní zápis příkazu je následující: "wxplot2d([funkce_1,funkce_2,...],[parametr,odkud,kam])". Příkaz pro vykreslení funkce y1 = sin(x) a y2 = cos (x) pro x = 0..2*Pi bude vypadat:

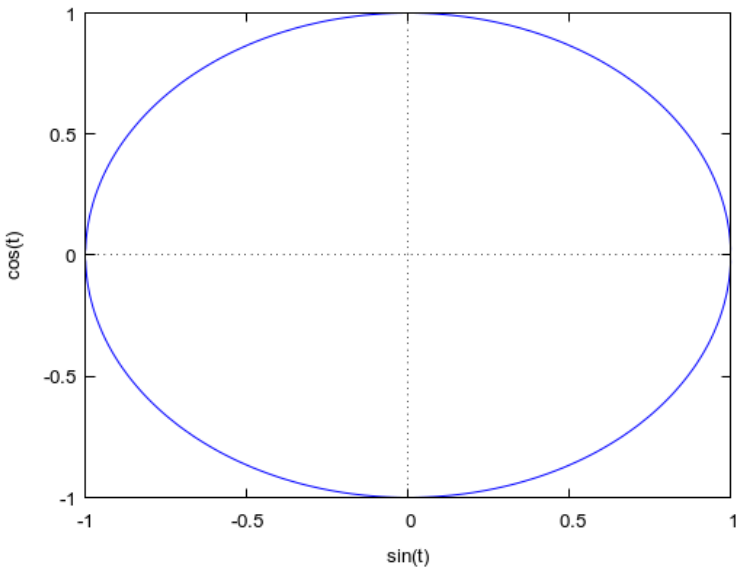
(% i1) wxplot2d([sin(x),cos(x)],[x,0,2*%pi])$

()




Pro vykreslení parametricky zadané křivky x = sin(t) a y = cos(t) s parametrem t = -Pi..Pi (hodí se pro kreslení trajektorií a fázorových diagramů):

```
(% i2)      wxplot2d([parametric, sin(t), cos(t), [t, -%pi, %pi]])$
()
```




Grafy lze pomocí mnoha parametrů doplnit o mnoho užitečných vlastností. Základní si ukážeme na zobrazení naměřených bodů "xy" a funkci "y=2*Pi*sqrt(x/980)" pro x = 0..50.

První řádek definuje sadu hodnot které jsme získali např. z experimentálního měření (pozn. "0.6" lze zapsat s vynecháním nuly ".6").

Dále následuje "wxplot2d" pro zobrazení grafů

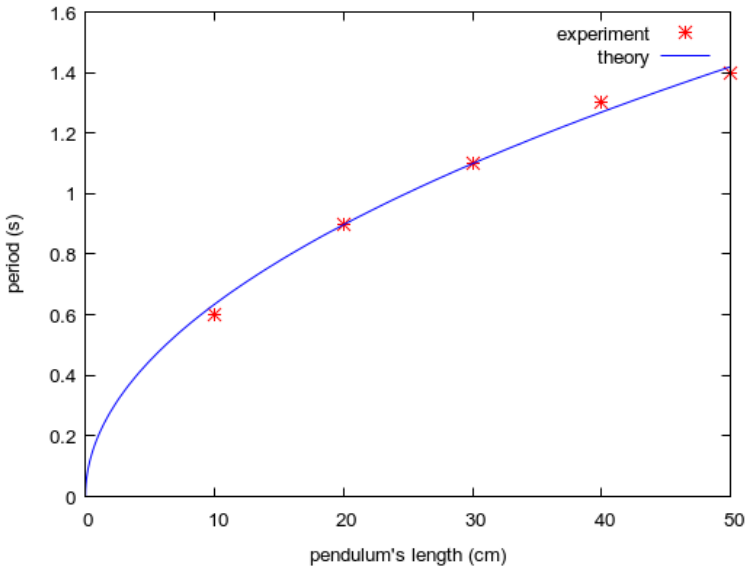
- "[discrete,xy]" znamená vykresli sadu nespojitých (diskrétních) hodnot ze seznamu "xy" - první funkce
- "[2*Pi*sqrt(x/980)]" definování funkce "y=2*Pi*sqrt(x/980)" - druhá funkce
- "[x,0,50]" hodnoty v ose x jsou od 0 do 50
- "[style, points, lines]" definuje styl zobrazených grafů, "[style, nastavení první funkce, nastavení druhé funkce,...]"
  - points - zobraz jako body (lze nastavit tvar)
  - lines - zobraz jako čáry (lze nastavit tloušťku)
  - linespoints - čáry + body
  - dot - tečky
- "[color, red, blue]" definuje barvu zobrazených grafů - "[color, barva první funkce, barva druhé funkce, ...]"

Lze použít následující barvy: 1: blue, 2: red, 3: magenta, 4:orange, 5:brown, 6: lime and 7: aqua

- "[point_type, asterisk]" pokud použijete pro vykreslení grafu styl "points" lze nastavit tvar bodu : bullet, circle, plus, times, asterisk, box, square, triangle, delta, wedge, nabla, diamond, lozenge
- "[legend, "experiment", "theory"]" - legenda ke grafům "[legend, "název první funkce", "název druhé funkce", ...]" !názvy musí býtv úvozovkách! Parametr "[legend, false]" skryje zobrazení legendy.
- "[xlabel, "pendulum's length (cm)"]" "[ylabel, "period (s)"]" - popisky os. !Opět musí být celý název osy v úvozovkách! Více se lze dočíst v nápovědě pod heslem "plot_options".

```
(% i4)  xy: [[10, .6], [20, .9], [30, 1.1], [40, 1.3], [50, 1.4]]$
        wxplot2d( [ [discrete, xy], 2*%pi*sqrt(x/980) ], [x,0,50],
        [style, points, lines], [color, red, blue], [point_type, asterisk],
        [legend, "experiment", "theory"],
        [xlabel, "pendulum's length (cm)"], [ylabel, "period (s)"])$
```

()




# 4 Derivace a integrály

```
(% i5)  kill(all);
```

() *done*

Pro derivování výrazu používáme funkci "diff", která se zapisuje pomocí následující syntaxe: "diff(derivovaná funkce nebo výraz, proměnná_1, proměnná 2,...)" nebo pro násobné derivace "diff(derivovaná funkce nebo výraz, proměnná, kolikátá derivace)". Napíšeme-li před jakoukoliv funkci jednoduchou úvozovku (v našem případě 'diff) potom se zobrazí tato funkce symbolicky, ale program funkci nespustí. Pro spuštění této funkce (např. až v konečném výsledku) slouží parametr "nouns". Ukažme si použití funkce "diff" na jednoduchém příkladu z kinematiky. Dráha je zadána jako "x(t)=A*cos(omega*t)", určete rychlost a zrychlení.

**(% i1)** x(t):=A*cos(%omega*t);

() $$\mathrm{x}(t) := A\cos(\omega t)$$

**(% i2)** v:diff(x(t),t);

(v) $$-\omega A\sin(\omega t)$$

**(% i3)** a:diff(v,t);

(a) $$-\omega^2 A\cos(\omega t)$$

Zrychlení lze vypočítat také jako druhou derivaci dráhy

**(% i4)** a:diff(x(t),t,2);

(a) $$-\omega^2 A\cos(\omega t)$$

Ukázka využití jednoduchého uvozovaní příkazu (derivace jsou symbolické, provedou se po použití parametru "nouns"

**(% i5)** v:'diff(x(t),t);

(v) $$\frac{d}{dt}\left(A\cos(\omega t)\right)$$

**(% i7)** a:'diff(v,t); a,nouns;

(a) $$\frac{d^2}{dt^2}\left(A\cos(\omega t)\right)$$

() $$-\omega^2 A\cos(\omega t)$$

(% i8) kill(all);

() *done*

ukázka derivace kvadratické funkce

(% i1) rov:a*x(t)^2+b*x(t)+c;

(rov) $$a\,\mathrm{x}(t)^2 + b\,\mathrm{x}(t) + c$$

(% i2) drov:diff(rov,t);

(drov) $$2a\,\mathrm{x}(t)\left(\frac{d}{dt}\,\mathrm{x}(t)\right) + b\left(\frac{d}{dt}\,\mathrm{x}(t)\right)$$

(% i3) ddrov:diff(rov,x(t));

(ddrov) $$2a\,\mathrm{x}(t) + b$$

Ukažme si neurčitou integraci předchozího výrazu

(% i4) integrate(ddrov, x(t));

() $$a\,\mathrm{x}(t)^2 + b\,\mathrm{x}(t)$$

Je zřejmé, že maxima zapomíná připočítat integrační konstantu!!

Pro praktické použití je nutné řešit úlohy typu f'(x)=g(t) - obvykle v kinematice nebo f"(x)=g(t) - v dynamice Pro volný pád lze sestavit pohybovou rovnici

(% i5) rovnice:diff(v(t),t)=-g;

(rovnice) $$\frac{d}{dt}\,\mathrm{v}(t) = -g$$

rovnici lze ručně (a to nechceme)!! separovat a řešit, nebo lze použít nástroj pro řešení ODR (obyčejných diferenciálních rovnic, anglicky ODE)

(% i6) reseni:ode2(rovnice,v(t),t);

(reseni) $$\mathrm{v}(t) = \%c - gt$$

Jak je vidět, maximě chybí počáteční podmínky a úlohu tedy řeší pomocí obecné integrační konstanty. Počáteční podmínky předepíšeme příkazem "ic1", předpokládejme v(t=0)=v0

(% i7) ic1(reseni,v(t)=v0,t=0);

() $$\mathrm{v}(t) = v0 - gt$$

## 4.1 Příklad na řešení nelineárních diferenciálních rovnic Lagrangeovy rovnice druhého druhu aplikované na kyvadlo

**(% i1)** kill(all);globalsolve:true;

() *done*

(globalsolve) true

Definujme složky polohového vektoru bodu na kyvadle v závislosti na polohovém úhlu phi(t) - zobecněná souřadnice

**(% i3)** x(phi):= R*sin(phi(t)); y(phi):=-R*cos(phi(t));

() $$\mathrm{x}(phi) := R \sin\left(\mathrm{phi}(t)\right)$$

() $$\mathrm{y}(phi) := (-R) \cos\left(\mathrm{phi}(t)\right)$$

Složky rychlostí jsou časové derivace souřadnic x a y

**(% i5)** vx:diff(x(phi),t); vy:diff(y(phi),t);

(vx) $$R \cos\left(\mathrm{phi}(t)\right) \left(\frac{d}{dt}\,\mathrm{phi}(t)\right)$$

(vy) $$R \sin\left(\mathrm{phi}(t)\right) \left(\frac{d}{dt}\,\mathrm{phi}(t)\right)$$

Kinetická energie kyvadla je "1/2 * m * vˆ2", kde "vˆ2 = vxˆ2 +vyˆ2"

**(% i6)** K:trigsimp(1/2*m*(vxˆ2+vyˆ2));

(K) $$\frac{R^2 m \left(\frac{d}{dt}\,\mathrm{phi}(t)\right)^2}{2}$$

Derivace kinetické energie podle časové derivace phi - zobecněné rychlosti (všimněte si apostrofu v derivaci, derivujeme podle derivace jakožto symbolu, nikoliv operace!!)

(% i7) dK:diff(K,'diff(phi(t),t));

(dK) $$R^2 m \left(\frac{d}{dt}\,\mathrm{phi}(t)\right)$$

Časová derivace derivace kinetické energie podle zobecněné rychlosti

(% i8) dtK:diff(dK,t);

(dtK) $$R^2 m \left(\frac{d^2}{dt^2}\,\mathrm{phi}(t)\right)$$

Potenciální energie

(% i9) U:m*g*y(phi);

(U) $$-Rgm\cos\left(\mathrm{phi}(t)\right)$$

Derivace potenciální energie podle zobecněné souřadnice (phi)

(% i10) dU:diff(U,phi(t));

(dU) $$Rgm\sin\left(\mathrm{phi}(t)\right)$$

Pohybová rovnice je součet dtK a dU

(% i11) rov:dtK+dU;

(rov) $$R^2 m \left(\frac{d^2}{dt^2}\,\mathrm{phi}(t)\right) + Rgm\sin\left(\mathrm{phi}(t)\right)$$

Jedná se o nelineární diferenciální rovnici druhého řádu, kterou budeme muset řešit numericky metodou Runge-Kutta. Jelikož se jedná o numerickou metodu, musíme si zadat číselné konstanty.

(% i14) R:0.5;g:9.81;m:0.2;

(R) 0.5

(g) 9.81

(m) 0.2

(% i15) rov:ev(rov);

$$\text{(rov)} \qquad 0.05\left(\frac{d^2}{dt^2}\,\text{phi}(t)\right) + 0.9810000000000001\sin\left(\text{phi}(t)\right)$$

RK metody jsou určené pro řešení soustav diferenciálních rovnic prvního řádu, proto musíme naši rovnici druhého řádu převést pomocí jednoduché substituce na soustavu dvou rovnic prvního řádu. Zavedeme označení pro časovou derivaci phi "diff(phi(t),t) = omega" - to je zároveň první z našich dif. rovnic prvního řádu (rov1). Tuto substituci dosadíme do rovnice "rov".

(% i16) rov1:omega='diff(phi(t),t);

$$\text{(rov1)} \qquad omega = \frac{d}{dt}\,\text{phi}(t)$$

Dostáváme druhou diferenciální rovnici prvního řádu (rov2).

(% i17) rov2:subst(['diff(phi(t),t,2)='diff(omega,t),phi(t)=phi],rov);

$$\text{(rov2)} \qquad 0.9810000000000001\sin\left(phi\right) + 0.05\left(\frac{d}{dt}omega\right)$$

RK metoda v Maximě je explicitní metoda. Je tedy nutné rovnice rov1 a rov2 napsat ve tvaru "diff(r(t),t) = f(r,t)", kde r jevektor zobecněných souřadnic a rychlostí. f(r,t) je potom vstup do RK metody.

Z rovnice rov1 vyjádři derivaci zobecněné souřadnice 'diff(phi(t),t) -> z tohoto vyjádření vem první člen [1] -> pravou stranu tohoto členu pojmenuj dphi.

(% i18) dphi: rhs( solve( rov1,'diff(phi(t),t) )[1] );

$$\text{(dphi)} \qquad omega$$

Z rovnice rov2 vyjádři derivaci zobecněné rychlosti 'diff(omega,t) -> z tohoto vyjádření vem první člen [1] -> pravou stranu tohoto členu pojmenuj domega.

(% i19) domega: rhs( solve( rov2,'diff(omega,t) )[1] );

rat: replaced 0.05 by 1/20 = 0.05rat: replaced 0.9810000000000001 by 981/1000 = 0.981

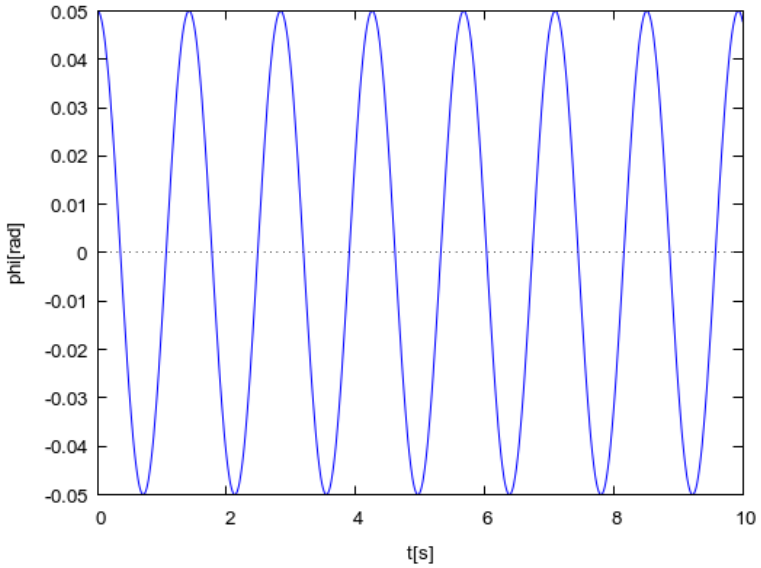
$$\text{(domega)} \qquad -\frac{981\sin\left(phi\right)}{50}$$

Vlastní řešení RK metodou spočívá v zadání vektoru pravých stran f(r,t) - [dphi,domega], definice zobecněných souřadnic a rychlostí - [phi,omega], definice

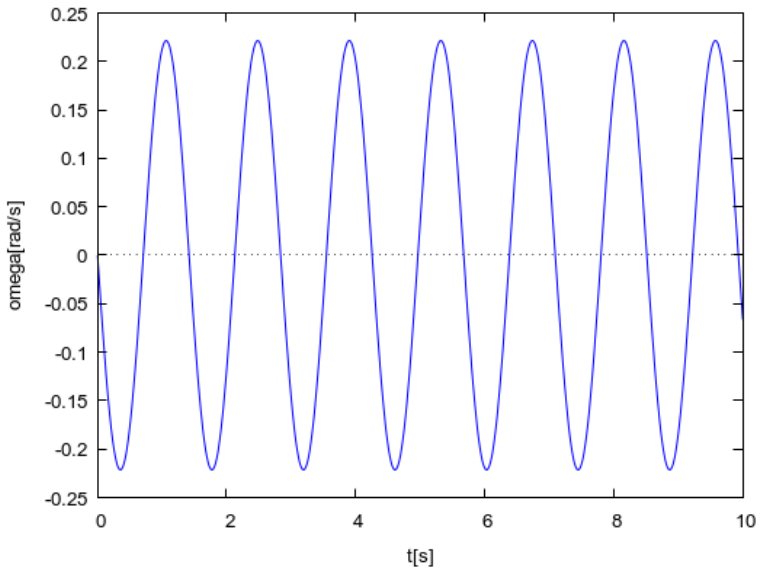
počátečních podmínek phi(0) a omega(0) - [0.1,0], zadání rozsahu nezávisle promněnné t [název promněnné, počáteční stav, koncový stav, krok]. Výsledkem je pole hodnot ve formátu [t,phi,omega] pro t = počáteční stav ... koncový čas se zadaným krokem.

**(% i20)** sol:rk([dphi,domega],[phi,omega],[0.05,0],[t,0,10,0.01])$
Pro vykreslení lze použít např. následující konstrukci, kde p[1] je čas, p[2] je výchylka, p[3] je rychlost.

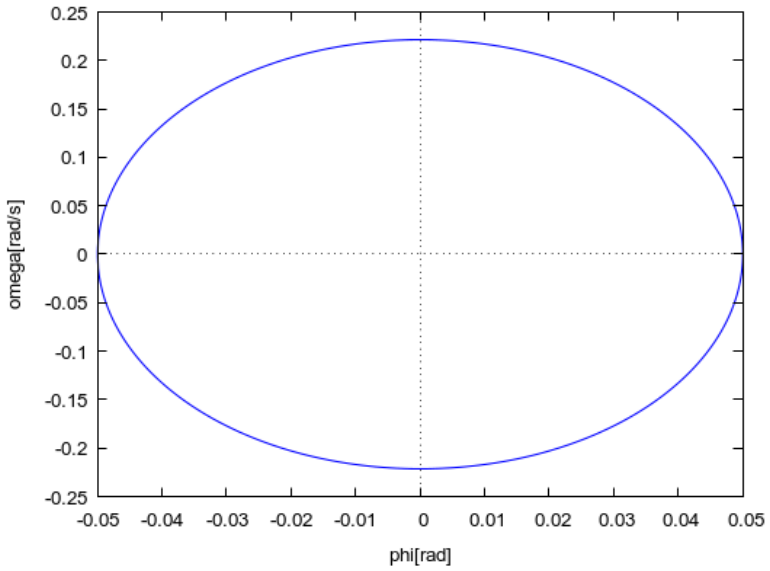
**(% i21)** wxplot2d ([discrete,makelist([p[1],p[2]],p,sol)],[xlabel,"t[s]"],[ylabel,"phi[rad]"])$
()



(% i22) wxplot2d ([discrete,makelist([p[1],p[3]],p,sol)],[xlabel,"t[s]"],[ylabel,"omega[rad/s]"])$
()




(% i23) wxplot2d ([discrete,makelist([p[2],p[3]],p,sol)],[xlabel,"phi[rad]"],[ylabel,"omega[rad/s]"])$
()




Pro rychlou představu o vlastnostech dif. rovnice lze použít příkaz plotdf([vektorpravých stran],[vektor zobecněných souřadnic a rychlostí],[trajectory_at,phi(0),omega(0)],[tstep,

krok metody],[rozsah osy x],[rozsah osy y], [direction,forward], [nsteps,počet krokůmetody],[versus_t,1])$

Další počáteční podmínky lze zadat také kliknutím do fázorové roviny(phi,omega), zobrazí se fázorový portrét pohybu a v sousedním grafu křivky phi(t) a omega(t). Pro plné využití doporučuji pečlivě prostudovat manuálovéstránky příkazu.

```
(% i24)  plotdf([dphi,domega], [phi,omega], [trajectory_at,0.1,0],[tstep,0.01], [phi,-
         %pi,%pi], [omega,-10,10], [direction,forward], [nsteps,500], [versus_t,1])$
```

# 5 Lineární algebra

Pro počítání s vektory a maticemi používá Maxima knihovny eigen a vect. Načteme je příkazem load("název knihovny"). Při používání maticového počtu je potřeba dávat velký pozor na to, s jakým objektem (typem) požadovanou operaci provádím. Matice 1x3 není totéž jako vektor! Další úskalí spočívá v používání operátorů. Většina operátorů (+, -, *, /, ^, log, sin) je aplikována na jednotlivé členy matice, nikoliv na matici (u součtu a rozdílu to nevadí). Maticový součin a skalární součin vektorů je obyčejná tečka (.), vektorový součin je vlnka (tilda, ∼).

```
(% i2)  kill(all);load ("eigen"); load("vect");
```

() *done*

()

/usr/share/maxima/5.41.0/share/matrix/eigen.mac

()

/usr/share/maxima/5.41.0/share/vector/vect.mac

## 5.1 Matice

Obecnou matici lze vytvořit pomocí příkazu genmatrix(název členu, počet řádků, počet sloupců) nebo příkazem matrix ([první řádek], [druhý ...])

```
(% i3)  A:genmatrix(A,3,3);
```

(A)
$$\begin{pmatrix} A_{1,1} & A_{1,2} & A_{1,3} \\ A_{2,1} & A_{2,2} & A_{2,3} \\ A_{3,1} & A_{3,2} & A_{3,3} \end{pmatrix}$$

(% i4) B:genmatrix(B,3,3);

(B)
$$\begin{pmatrix} B_{1,1} & B_{1,2} & B_{1,3} \\ B_{2,1} & B_{2,2} & B_{2,3} \\ B_{3,1} & B_{3,2} & B_{3,3} \end{pmatrix}$$

Matice lze v Maximě sčítat a odčítat, ale pozor lze i násobit a dělit matici maticí- to je v rozporu s operacemi v klasické algebře!! Pokud použijete mezi maticemi operátor "krát" (klasická hvězdička na numerické klávesnici, v novějších verzích Maximy reprezentováno tečkou) nebo "děleno" (lomítko),provede se požadovaná operace po jednotlivých členech.

(% i5) rozdíl:A-B;

(rozdíl)
$$\begin{pmatrix} A_{1,1}-B_{1,1} & A_{1,2}-B_{1,2} & A_{1,3}-B_{1,3} \\ A_{2,1}-B_{2,1} & A_{2,2}-B_{2,2} & A_{2,3}-B_{2,3} \\ A_{3,1}-B_{3,1} & A_{3,2}-B_{3,2} & A_{3,3}-B_{3,3} \end{pmatrix}$$

(% i6) soucin_clenu:A*B;

(soucin_ clenu)
$$\begin{pmatrix} A_{1,1}\,B_{1,1} & A_{1,2}\,B_{1,2} & A_{1,3}\,B_{1,3} \\ A_{2,1}\,B_{2,1} & A_{2,2}\,B_{2,2} & A_{2,3}\,B_{2,3} \\ A_{3,1}\,B_{3,1} & A_{3,2}\,B_{3,2} & A_{3,3}\,B_{3,3} \end{pmatrix}$$

(% i7) podil_clenu:A/B;

(podil_ clenu)
$$\begin{pmatrix} \frac{A_{1,1}}{B_{1,1}} & \frac{A_{1,2}}{B_{1,2}} & \frac{A_{1,3}}{B_{1,3}} \\ \frac{A_{2,1}}{B_{2,1}} & \frac{A_{2,2}}{B_{2,2}} & \frac{A_{2,3}}{B_{2,3}} \\ \frac{A_{3,1}}{B_{3,1}} & \frac{A_{3,2}}{B_{3,2}} & \frac{A_{3,3}}{B_{3,3}} \end{pmatrix}$$

(% i8) log(A)/log(B);

()
$$\begin{pmatrix} \frac{\log(A_{1,1})}{\log(B_{1,1})} & \frac{\log(A_{1,2})}{\log(B_{1,2})} & \frac{\log(A_{1,3})}{\log(B_{1,3})} \\ \frac{\log(A_{2,1})}{\log(B_{2,1})} & \frac{\log(A_{2,2})}{\log(B_{2,2})} & \frac{\log(A_{2,3})}{\log(B_{2,3})} \\ \frac{\log(A_{3,1})}{\log(B_{3,1})} & \frac{\log(A_{3,2})}{\log(B_{3,2})} & \frac{\log(A_{3,3})}{\log(B_{3,3})} \end{pmatrix}$$

Pozor, rovněž $A^2$ není rovno A.A

(% i9) mocnina_clenu:A^2;

(mocnina_ clenu)
$$\begin{pmatrix} A_{1,1}^2 & A_{1,2}^2 & A_{1,3}^2 \\ A_{2,1}^2 & A_{2,2}^2 & A_{2,3}^2 \\ A_{3,1}^2 & A_{3,2}^2 & A_{3,3}^2 \end{pmatrix}$$

**(% i10)** mocnina_na_cleny:2^A;

(mocnina_ na_ cleny)
$$\begin{pmatrix} 2^{A_{1,1}} & 2^{A_{1,2}} & 2^{A_{1,3}} \\ 2^{A_{2,1}} & 2^{A_{2,2}} & 2^{A_{2,3}} \\ 2^{A_{3,1}} & 2^{A_{3,2}} & 2^{A_{3,3}} \end{pmatrix}$$

Klasický maticový součin se prování pomocí obyčejné tečky.

**(% i11)** maticovy_soucin:A.B;
(maticovy_ soucin)
$$\begin{pmatrix} A_{1,3}\,B_{3,1}+A_{1,2}\,B_{2,1}+A_{1,1}\,B_{1,1} & A_{1,3}\,B_{3,2}+A_{1,2}\,B_{2,2}+A_{1,1}\,B_{1,2} & A_{1,3}\,B_{3,3}+A_{1,2}\,B_{2,3}+A_{1,1}\,B_{1,3} \\ A_{2,3}\,B_{3,1}+B_{2,1}\,A_{2,2}+B_{1,1}\,A_{2,1} & A_{2,3}\,B_{3,2}+A_{2,2}\,B_{2,2}+B_{1,2}\,A_{2,1} & A_{2,3}\,B_{3,3}+A_{2,2}\,B_{2,3}+B_{1,3}\,A_{2,1} \\ B_{3,1}\,A_{3,3}+B_{2,1}\,A_{3,2}+B_{1,1}\,A_{3,1} & B_{3,2}\,A_{3,3}+B_{2,2}\,A_{3,2}+B_{1,2}\,A_{3,1} & A_{3,3}\,B_{3,3}+B_{2,3}\,A_{3,2}+B_{1,3}\,A_{3,1} \end{pmatrix}$$

Transponovaná matice

**(% i12)** transpose(A);

()
$$\begin{pmatrix} A_{1,1} & A_{2,1} & A_{3,1} \\ A_{1,2} & A_{2,2} & A_{3,2} \\ A_{1,3} & A_{2,3} & A_{3,3} \end{pmatrix}$$

Inverze matice.

**(% i13)** Inverzni:matrix([1,2],[2,1]) ^^ -1;

(Inverzni)
$$\begin{pmatrix} -\frac{1}{3} & \frac{2}{3} \\ \frac{2}{3} & -\frac{1}{3} \end{pmatrix}$$

**(% i14)** Inverzni.matrix([1,2],[2,1]) ;

()
$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

Porovnejme inverzní matici s výsledky získanými za pomoci jiných operátorů.

**(% i15)** matrix([1,2],[2,1]) ^ -1;

()
$$\begin{pmatrix} 1 & \frac{1}{2} \\ \frac{1}{2} & 1 \end{pmatrix}$$

**(% i16)** 1/matrix([1,2],[2,1]) ;

()
$$\begin{pmatrix} 1 & \frac{1}{2} \\ \frac{1}{2} & 1 \end{pmatrix}$$

## 5.2 Vektory

Vektory se zadávají pomocí hranatých závorek - potom se chovají jako tzv. list(seznam) ne jako jako matice 1xn, všimněte si jiného fontu u matice u.

**(% i18)** v:[v[1],v[2],v[3]]; u:matrix([u[1],u[2],u[3]]);

(v) $$[v_1, v_2, v_3]$$

(u) $$\begin{pmatrix} u_1 & u_2 & u_3 \end{pmatrix}$$

Definujme vektor "u" znovu a lépe

**(% i19)** kill(u);

() *done*

**(% i20)** u:[u[1],u[2],u[3]];

(u) $$[u_1, u_2, u_3]$$

Skalární součin se provádí pomocí obyčejné tečky

**(% i21)** u.v;

() $$u_3\, v_3 + u_2\, v_2 + u_1\, v_1$$

**(% i22)** u.u;

() $$u_3^2 + u_2^2 + u_1^2$$

Pozor, jako u matic tak i u vektorů $u^2$ není u.u

**(% i23)** u^2;

() $$[u_1^2, u_2^2, u_3^2]$$

Pro vektorový součin používáme vlnovku $\sim$, nicméně se daná operace provede až na vyžádání příkazem express(%);

**(% i24)** w:u~v;

(w) $$[u_1, u_2, u_3] \sim [v_1, v_2, v_3]$$

(% i25) w:express(w);

(w) $$[u_2\,v_3 - v_2\,u_3, v_1\,u_3 - u_1\,v_3, u_1\,v_2 - v_1\,u_2]$$

Z obecného vektoru lze vytvořit vektor jednotkový (unitární)

(% i26) uvect(u);

() $$[\frac{u_1}{\sqrt{u_3^2+u_2^2+u_1^2}}, \frac{u_2}{\sqrt{u_3^2+u_2^2+u_1^2}}, \frac{u_3}{\sqrt{u_3^2+u_2^2+u_1^2}}]$$

Pozor, příkaz pro traspozici vektoru převede vektor na matici 3x1, musíme pro indexování používat dva indexy.

(% i27) vt:transpose(v);

(vt) $$\begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix}$$

(% i28) vt[3,1];

() $$v_3$$

Stejně se chová i příkaz pro sloupcový vektor

(% i29) a:covect([1,2,3]);

(a) $$\begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$$

(% i30) a[2,1];

() 2

V Maximě není prakticky nutné používat sloupcové vektory!

## 5.3 Smíšené operace

Součin matice a řádkového vektoru je shodný se součinem matice a sloupcového vektoru.

(% i31) A.v;

() $$\begin{pmatrix} A_{1,3}\,v_3 + A_{1,2}\,v_2 + v_1\,A_{1,1} \\ A_{2,3}\,v_3 + v_2\,A_{2,2} + v_1\,A_{2,1} \\ v_3\,A_{3,3} + v_2\,A_{3,2} + v_1\,A_{3,1} \end{pmatrix}$$

(% i32) A.transpose(v);

$$() \qquad \begin{pmatrix} A_{1,3}\,v_3 + A_{1,2}\,v_2 + v_1\,A_{1,1} \\ A_{2,3}\,v_3 + v_2\,A_{2,2} + v_1\,A_{2,1} \\ v_3\,A_{3,3} + v_2\,A_{3,2} + v_1\,A_{3,1} \end{pmatrix}$$

(% i33) A.transpose(v)-A.v;

$$() \qquad \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

Matice může být i výstupem funkce, např. lze vytvořit funkci pro generování transformační matice

(% i34) T(x):=matrix([cos(x),sin(x)],[-sin(x),cos(x)]);

$$() \qquad \mathrm{T}(x) := \begin{pmatrix} \cos(x) & \sin(x) \\ -\sin(x) & \cos(x) \end{pmatrix}$$

(% i35) T(phi);

$$() \qquad \begin{pmatrix} \cos(phi) & \sin(phi) \\ -\sin(phi) & \cos(phi) \end{pmatrix}$$

Transformační matici lze derivovat podle času

(% i36) diff(T(phi(t)),t);

$$() \qquad \begin{pmatrix} -\sin(\mathrm{phi}(t))\left(\frac{d}{dt}\,\mathrm{phi}(t)\right) & \cos(\mathrm{phi}(t))\left(\frac{d}{dt}\,\mathrm{phi}(t)\right) \\ -\cos(\mathrm{phi}(t))\left(\frac{d}{dt}\,\mathrm{phi}(t)\right) & -\sin(\mathrm{phi}(t))\left(\frac{d}{dt}\,\mathrm{phi}(t)\right) \end{pmatrix}$$

nebo podle argumentu

(% i37) diff(T(phi(t)),phi(t));

$$() \qquad \begin{pmatrix} -\sin(\mathrm{phi}(t)) & \cos(\mathrm{phi}(t)) \\ -\cos(\mathrm{phi}(t)) & -\sin(\mathrm{phi}(t)) \end{pmatrix}$$

Ukažme, že inverze a transpozice transformační matice jsou shodné.

(% i38) trigsimp(T(phi)^^-1-transpose(T(phi)));

$$() \qquad \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$$