MATLAB®
pro začínající uživatele

Karel Zaplatílek

OBSAH

Předmluva

Vážený čtenáři, tato kniha je učebnicí základů práce v programovém prostředí MATLAB®&Simulink®. Tento celosvětově rozšířený software byl od svého vzniku v roce 1985 určen především technikům, inženýrům a vědcům, vyžadujícím výkonné výpočetní prostředí pro vývoj a testování algoritmů bez nutnosti programovat v jazycích jako byly C a Fortran.

V MATLAB lze řadu věcí realizovat více než jedním postupem; můžete pracovat ponejvíce myší, ručně psát a spouštět příkazy (zdrojové kódy) v klasickém editoru, používat nově tzv. *Live Editor*, kombinující zdrojové kódy s texty, rovnicemi, obrázky, odkazy na internet a možností publikování v několika formátech, např. *PDF*. Někdo rád programuje ručně, jiný dává přednost více interaktivnímu přístupu. Pracovat lze také blokovou formou v nadstavbě s názvem Simulink.

MATLAB je dnes rozsáhlým prostředím, které si uživatel může nadále rozšiřovat zakoupením velkého množství doplňkových knihoven. Moje praxe učitele na technické fakultě univerzity a mnoho zkušeností s výukou MATLAB v akademickém i komerčním prostředí však ukazuje, že základy jsou v principu neměnné.

Cílem této knihy je pomoci vám vytvořit dobré návyky při práci v MATLAB a tím získat pevné základy, které již budete moci sami rozvíjet. Mnohokrát jsem se přesvědčil, že bez pevných základů je práce v takto výkonném a rozsáhlém prostředí neproduktivní a člověka příliš nebaví. Jakmile se přesvědčíte, že začít pracovat a tvořit v MATLAB je snadné a že vás program *poslouchá*, začnete si svou tvorbu užívat.

Přeji si, aby se kniha stala praktickou příručkou práce s MATLAB pro každý den, abyste při ní cítili radost z tvořivé práce a objevovali své vlastní nové postupy a možnosti.

Knihu jsem napsal po mnoha letech intenzivní a naplňující práce v MATLAB&Simulink, obohacen o cenné pedagogické zkušenosti při výuce studentů mnoha úrovní studia a při práci na vědeckých a odborných projektech. Všem, kteří k tomu přispěli, upřímně děkuji.

Autor

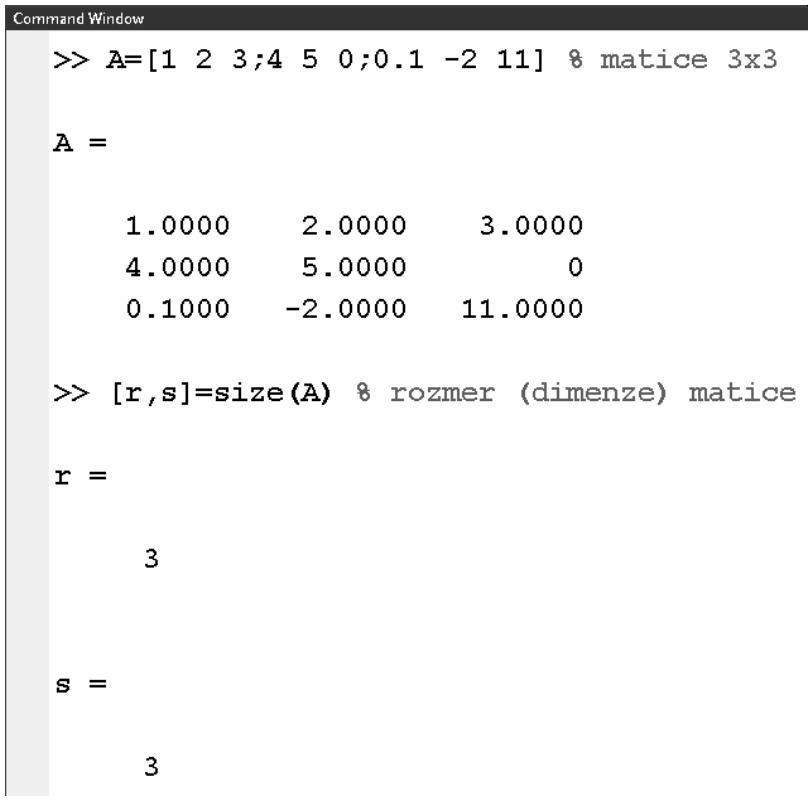
5 Práce s maticemi

Opět pracujeme *v příkazové řádce* stylem *příkaz + Enter*. Potvrzený příkaz je znovu dostupný buďto kurzorovými šipkami na klávesnici (nahoru, dolů) anebo v okně *Command History*.

Matice je podobná vektoru, má však oba rozměry větší než jedna. Vektor je tak zvláštním případem matice. Také práce s maticemi je analogická vektorům. Na případné rozdíly poukážeme níže.

5.1 Ruční tvorba matic

Je analogická tvorbě vektorů. Prvky v řádku oddělujeme mezerami nebo čárkami, řádky od sebe středníkem. Kromě ruční tvorby vlastní matice můžeme v MATAB tvořit matice *zvláštní*, mající své vlastní příkazy. Mnohé použité příkazy jsou již podrobně vysvětlené u vektorů.



Obr.5.1. Ruční tvorba malé matice s určením jejího rozměru.

Tip: *whos A*

```
Command Window
>> B=magic(3) % stejne soucty v radcich a sloupcich

B =

     8     1     6
     3     5     7
     4     9     2

>> C=6+2*randn(3,4) % pseudonahodna matice

C =

    3.3846   13.1568   12.0698    7.4295
    5.1328   11.5389    7.4508    5.5901
    6.6852    3.3002    5.8739    5.7517
```

Obr.5.2. Ruční tvorba zvláštních matic.

```
Command Window
>> D=zeros(3,5) % matice samych nul

D =

     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0

>> D(:,:)=5 % naplneni matice cislem 5

D =

     5     5     5     5     5
     5     5     5     5     5
     5     5     5     5     5
```

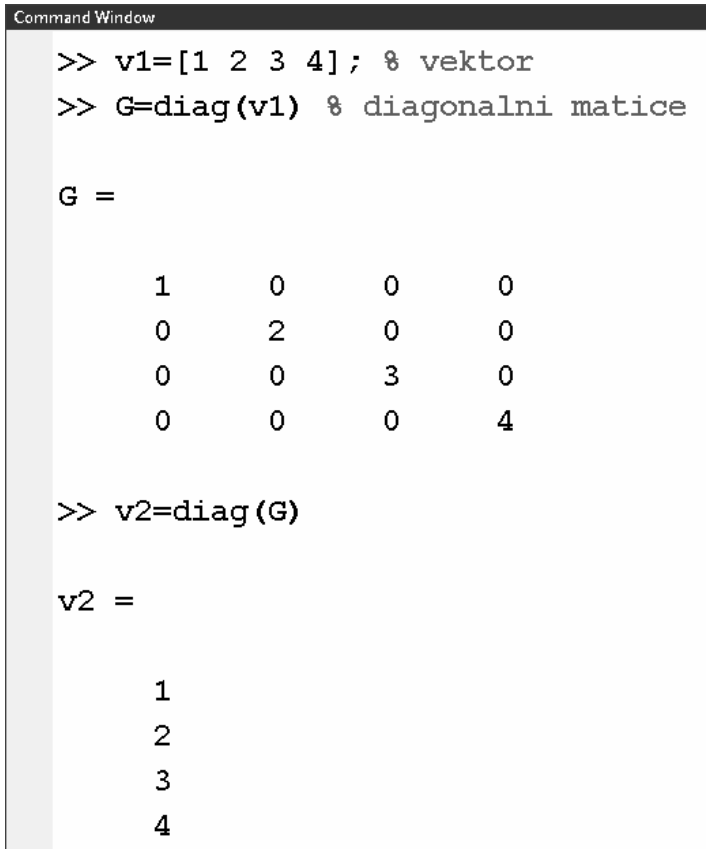
Obr.5.3. Ruční tvorba zvláštních matic.

```
Command Window
>> E=ones(2,3) % matice samych 1

E =

     1     1     1
     1     1     1

>> F=eye(3) % jednotkova matice

F =

     1     0     0
     0     1     0
     0     0     1
```

Obr.5.4. Ruční tvorba zvláštních matic.

```
Command Window
>> H=zeros(3,3,2) % vicerozmerna matice 3x3x2

H(:,:,1) =

     0     0     0
     0     0     0
     0     0     0


H(:,:,2) =

     0     0     0
     0     0     0
     0     0     0
```

Obr.5.5. Ruční tvorba vícerozměrné matice.



Obr.5.6. Ruční tvorba zvláštních matic.

9 Základy grafiky 2-D

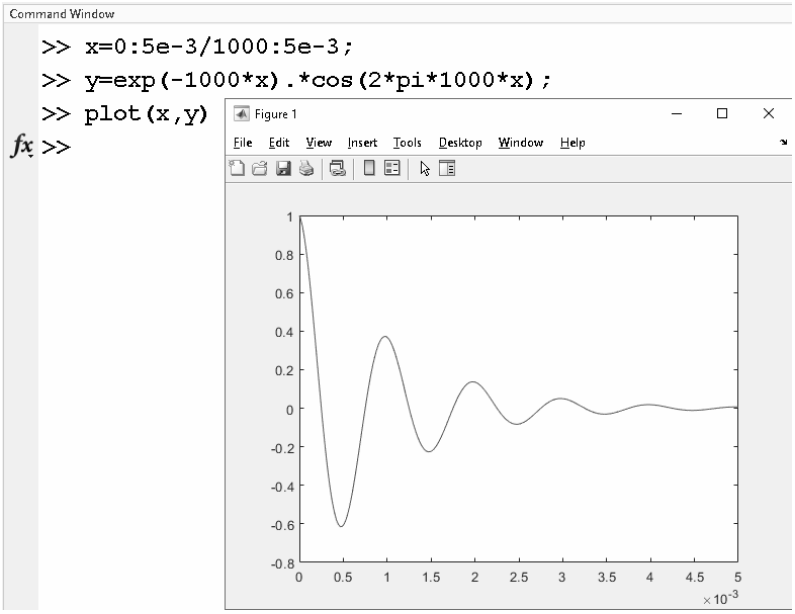
Grafika 2-D a 3-D je v MATLAB opravdu povedená a výkonná. Tvořit základní grafy je snadné; můžete používat příkazy anebo více klikat myší na proměnné v okně *Workspace*.

Ke každému příkazu pro kreslení existuje řada dalších parametrů, kterými můžete výsledný graf ovlivnit. V této knize se seznámíme s několika základními přístupy pro kreslení, ale důraz je kladen na pochopení principu práce s *grafickým subsystémem* v MATLAB. Jakmile pochopíte princip, můžete již více tvořit samostatně s použitím *helpu*.

9.1 Tvorba a editace grafu 2-D, příkaz *plot*

Grafy 2-D jsou grafy v rovině a mohou být čárové (*line*) anebo plošné (*surface*). Grafy 2-D mají dvě osy, jednu vodorovnou (nezávisle proměnnou), druhou svislou (závisle proměnnou). Budete-li kreslit jen skalár (číslo), bude grafem bod v rovině. Abychom získali křivky, musejí být na obou osách *vektory stejné délky*. Počet prvků (vzorků) vektoru bude hrát roli v tom, zda výsledná křivka bude hladká nebo kostrbatá.

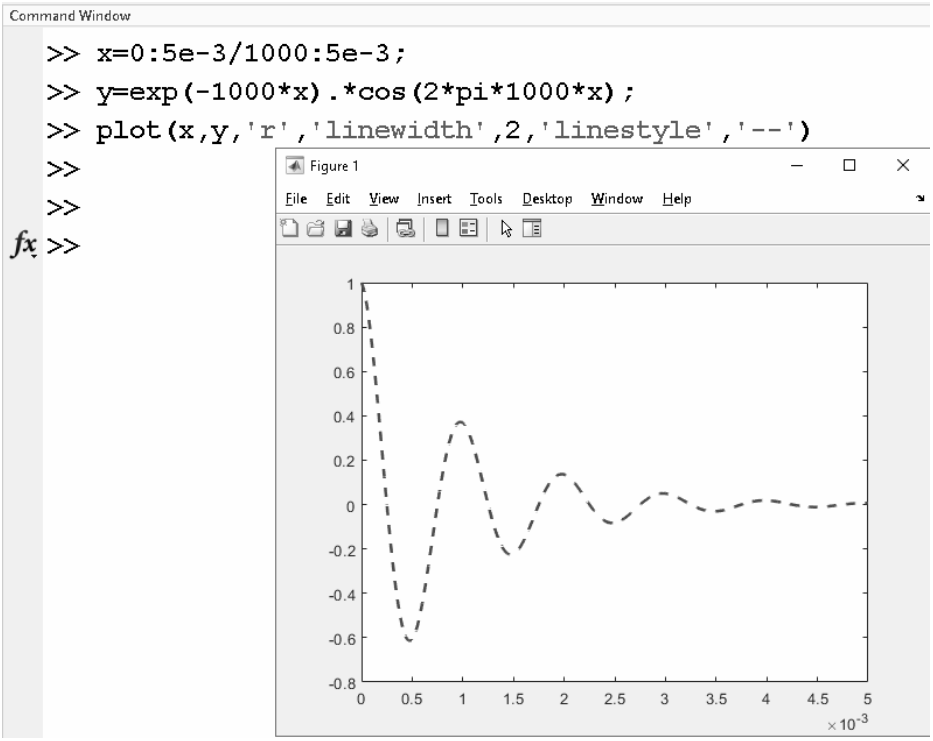
Kromě křivek v rovině však mohou být 2-D grafy také sloupcové, koláčové, vektorové, schodovité, polární, aj. Kromě dvou vektorů lze také kreslit celé matice. Začneme v příkazové řádce.



Obr.9.1. Základní 2-D čárový graf příkazem „plot“.

Na *obr.9.1* jsme vytvořili dva vektory o délce 1001 vzorků (1000 mezer) a vykreslili je příkazem *plot*. Graf je samozřejmě barevný, systém zvolí implicitní modrou barvu a automaticky nastaví rozsahy obou os.

Příkaz plot můžeme obohatit (původní okno s obrázkem zavřete):



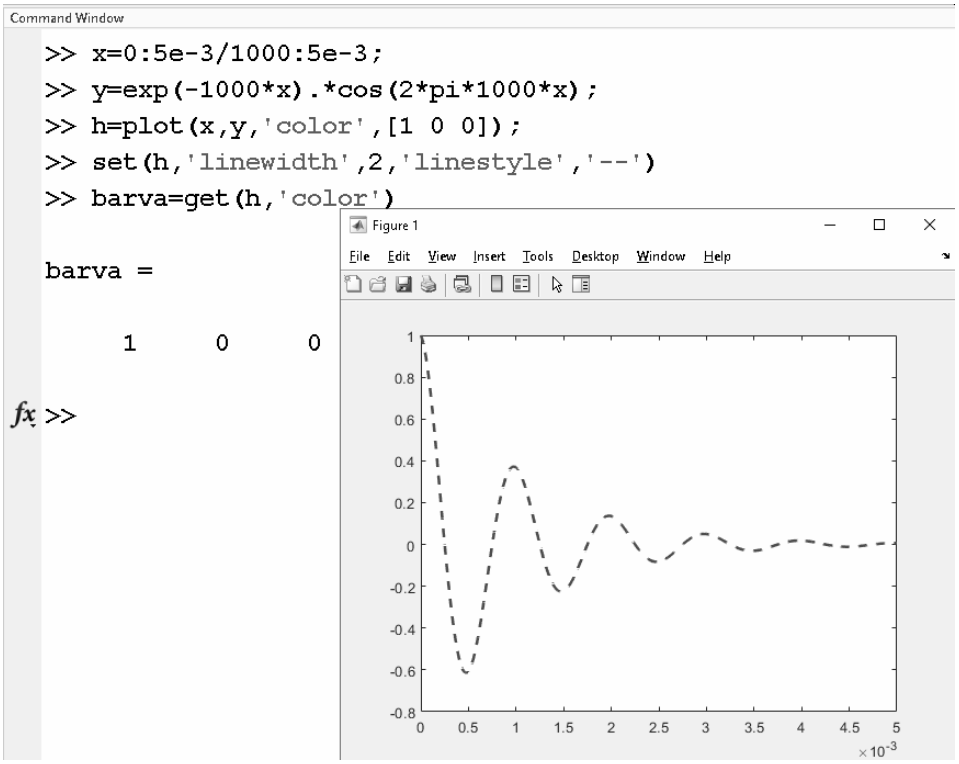
Obr.9.2. Další parametry příkazu „plot“.

Na *obr.9.2* je nastavena barva na červenou (*red*), tloušťka čáry na *2* a styl na čárkovaný. Tento způsob je plně funkční, já vám ale nabídnu ještě výkonnější způsob práce s grafikou. Okno s grafem zavřete.

Na *obr.9.3* opět kreslíme z příkazové řádky, ale před příkazem *plot* je uvedena proměnná *h* (nebo jiná). Tuto proměnnou nazýváme *handle*. Pomocí této proměnné pak můžeme využívat příkaz *set* k nastavení parametrů grafu a *get* k dotazům na parametry již nastavené.

Tato trojice *handle, set, get* tvoří klíč k ovládání prakticky veškeré grafiky v MATLAB. Všimněte si, že barva je nastavena pomocí parametru *color* a RGB čísel, každé od 0 do 1. Proměnná *barva* obsahuje nastavenou barvu jako vektor tří prvků RGB.

Za příkaz plot dávejte vždy *středník*, aby se obsah *handle* zbytečně nevypisoval.



Obr.9.3. Použití tzv. „handle“ a příkazy „set“ a „get“.

Vyzkoušejte si např. tyto příkazy (graf nezavírejte):

set(h,'visible','off ')
set(h,'visible','on')
set(h,'linestyle','-')
m=get(h,'marker')

Do grafu můžete dále dokreslit mřížku, nastavit jinak rozsahy obou os (zoom), popsat obě osy a doplnit titulek grafu, viz *obr.9.4*. V příkazu *axis* pro nastavení rozsahu obou os (vždy *min*, *max*) je uvedeno *-Inf* u minima svislé osy, což znamená *„nechám to programe na tobě, neřeším to“*. Pokud bych svislou osu nechtěl řešit vůbec, nastavil bych zde:

axis([x(1) x(300) -Inf Inf]).

Je zřejmé, že počet příkazů již začíná narůstat; je tedy čas začít je zapisovat do m-souboru ve formě *scriptu*, viz kapitola 8.1 a dále.

10 Základy grafiky 3-D

Grafy 3-D jsou dvojího druhu a to čárové (anglicky *line*) a plošné (*surface*), podobně jako grafy 2-D, viz úvod kapitoly 9.1. Některé 2-D příkazy pro kreslení mají také své 3-D ekvivalenty, např. *plot - plot3*, *bar - bar3*, *stem* - stem3, jiné jsou originální, zejména plošné 3-D grafy.

Mnoho postupů a technik jako je práce s *handly*, použití příkazů *set* a *get*, práce s objekty *axes* a *figure*, příkaz *subplot*, aj. jsou shodné; nebudou zde tedy opakovány. Zaměříme se jen na podstatné základní techniky. Zde více než jinde doporučuji pracovat s velkým *helpem* a jeho příklady z grafiky, viz výše kapitola 9.14.

10.1 Čárový graf 3-D, příkazy *plot3* a *stem3*

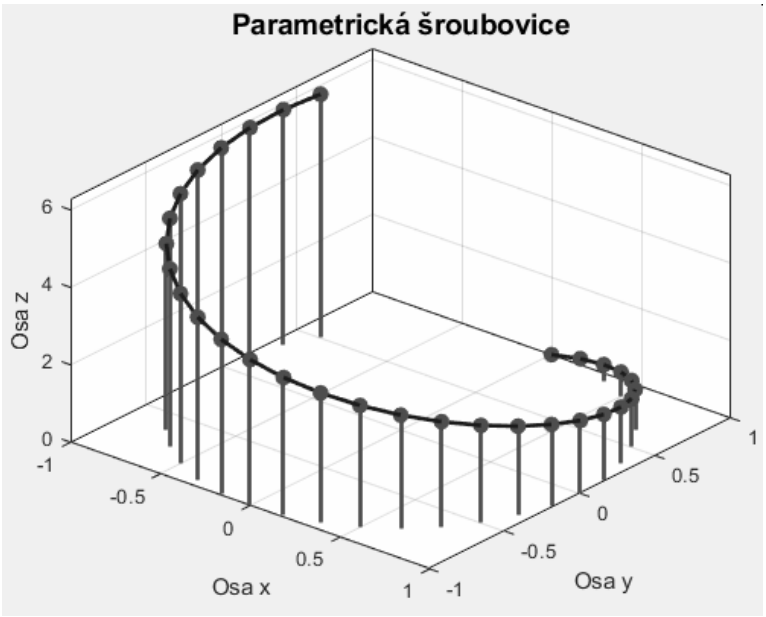
Čárové 3-D grafy jsou velmi podobné svým 2-D variantám. Mají pochopitelně tři osy, dvě vodorovné (nezávisle proměnné) a jednu svislou (závisle proměnnou). Na všech třech osách musejí být *vektory shodné délky* (kreslíme křivky ve 3-D) anebo skaláry (bod ve 3-D).

Čárový (*line*) 3-D graf je křivka ve 3-D (ne plocha). Na *obr.10.1* je základní použití příkazu *plot3*. Je třeba naplnit vektory ve všech třech osách. Na ose *z* je zde čas, což je v technických disciplínách časté. Z hlediska programování je to však podružné, potřebujeme zkrátka tři stejně dlouhé vektory. Pauza je zde opět pro lepší vizuální efekt.

Novým je zde příkaz *view*, který vrací (anebo nastavuje) *azimut* a *elevaci*, tedy úhel natočení ve vodorovném a ve vertikálním směru. Kromě příkazu *plot3* je zde použit také příkaz *stem3* pro zvýraznění vzorkového charakteru křivky.

Graf_12.m

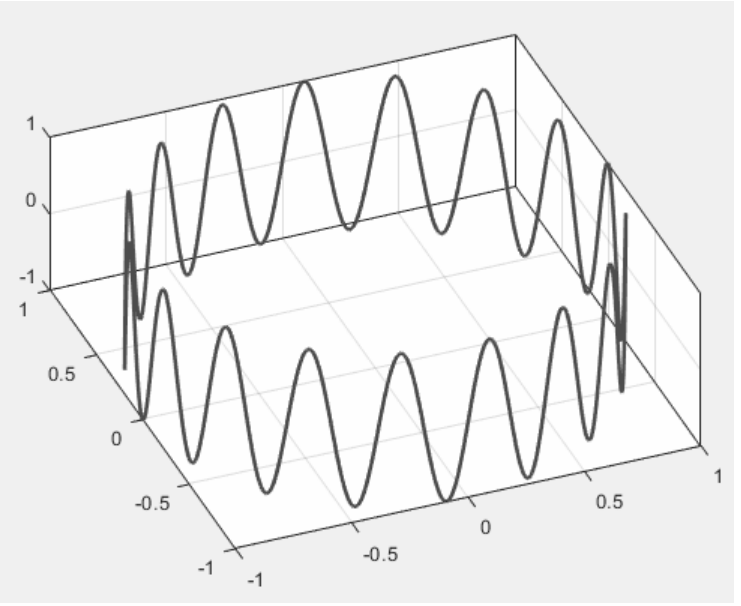
```
% 3-D grafy
% 18.7.2020
%----------
close all
t=0:2*pi/30:2*pi;
x=sin(t/1.2); y=cos(t/1.2); z=t; % param. sroubovice
h1=plot3(x,y,z,'linewidth',2,'color','b');
box on
grid on
xlabel('Osa x')
ylabel('Osa y')
zlabel('Osa z')
title('Parametrická šroubovice','fontsize',14)
[az,ele]=view % vychozi natoceni grafu
view(40,39) % vlastni natoceni grafu
hold on
pause(1.5)
h2=stem3(x,y,z,'fill'); % vzorkovy charakter
pause(1)
set(h2,'linewidth',2,'color','r','marker','o')
%---
```



Obr.10.1. Parametrická šroubovice jako čárový 3-D graf.

Graf_13.m

```
% 3-D grafy
% 18.7.2020
%----------
close all
n=17; % prohnuti 3-D cary
t=0:2*pi/1000:2*pi;
N=length(t);
h3=plot3(sin(t),cos(t),sin(n*t),'linewidth',2);
set(h3,'color',[1 0 0])
box on
grid on
view(-21.8,61.2)
hold on
%---
```

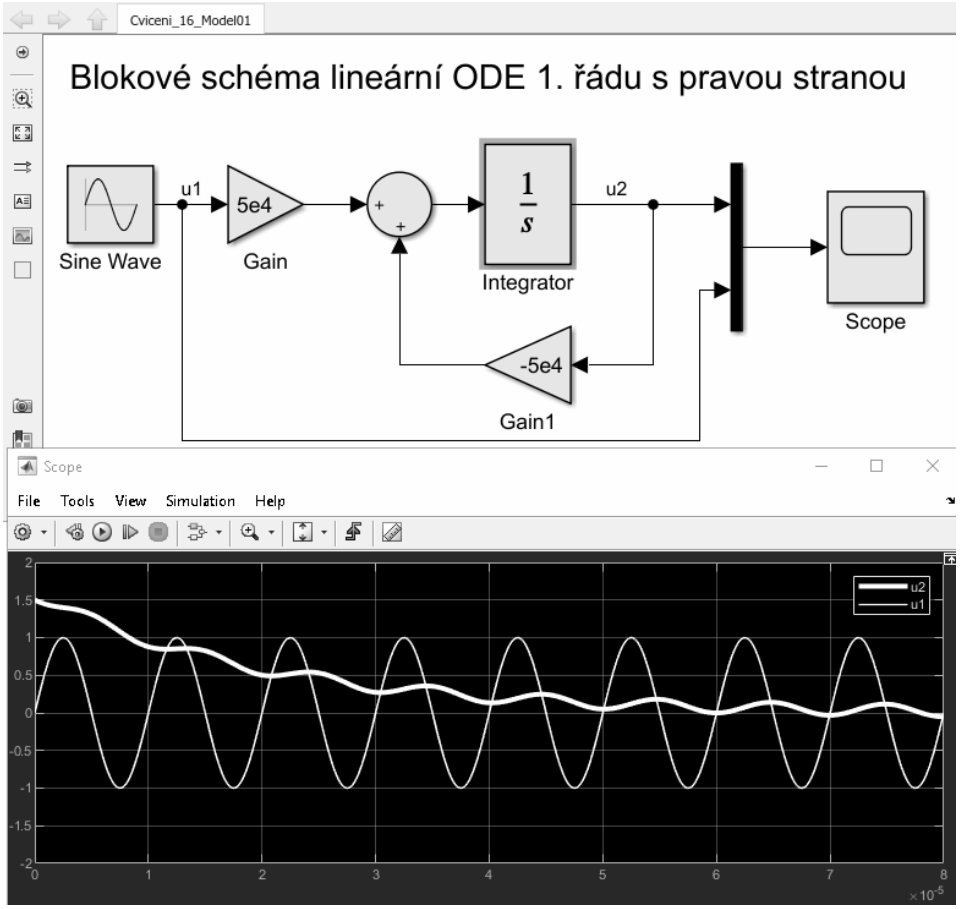


Obr.10.2. Další příklad 3-D čárového grafu.

V helpu na heslo *plot3* naleznete další a velmi povedené příklady 3-D čárových grafů s mnoha možnostmi editace zejména barev.

15 Úvod do prostředí Simulink

Simulink je placenou nadstavbou systému MATLAB, kde se pracuje s bloky, viz příklad na *obr.14.1* (řešení lineární ODE 1. řádu z *příkladu 1* kapitoly 12.2).



Obr.15.1. Příklad blokového modelu v prostředí Simulink.

Simulink spustíte z příkazové řádky příkazem *simulink* anebo tlačítkem uprostřed horní lišty v záložce *Home*. Máte-li již uložený nějaký model, stačí na něj poklepat myší. V novějších verzích mají modely příponu **.slx* (neplést s příponou **.mlx* u souborů z *Live Editoru*).

Tato kniha není primárně určena pro práci v Simulink, proto se zde zaměřím na úplné základy, abyste získali určitý námět na vaši další práci.

15.1 Řešení lineární ODE 2. řádu

V rámci kapitoly 12.2 je tato rovnice řešena klasickým textovým způsobem s využitím vestavěné numerické metody *ode45*, viz výše *obr.12.5* a *obr.12.6* Zde si ukážeme transformaci této ODE do blokového modelu, potřebná nastavení a spuštění simulace.

Připomeňme si, že hlavním úkolem je namodelovat (popsat) a vyřešit lineární ODE 2. řádu s konstantními koeficienty:

$$\frac{d^2y}{dt^2} + K_1\frac{dy}{dt} + K_2 y = 0,$$

kde $K_1 = 150$, $K_2 = 6{,}25.10^5$, počáteční podmínky $A_1 = 0{,}02$ a $A_2 = -0{,}02$, čas řešení $t \in \langle 0, 50 \rangle ms$. Rovnice má na pravé straně nulu, jde tedy o tzv. *homogenní* ODE (fyzikálně bez budicí veličiny). Hledaným řešením je opět časový průběh funkce $y = f(t)$.

Vezmeme nyní výše uvedenou ODE, osamostatníme vlevo nejvyšší (druhou) derivaci a rovnici dvakrát (2. derivace) zintegrujeme:

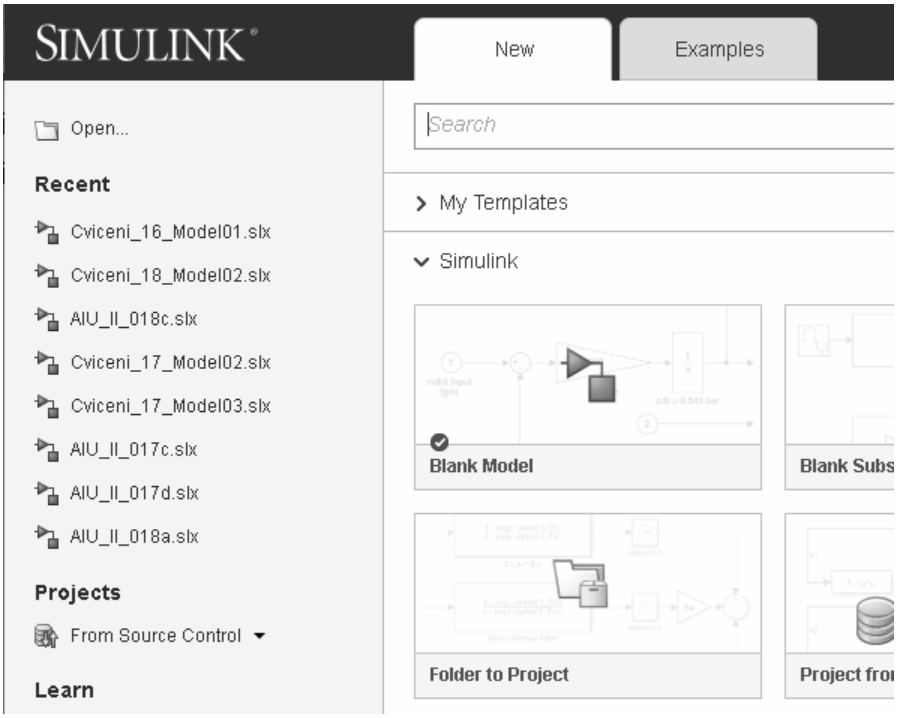
$$\frac{d^2y}{dt^2} = -K_1\frac{dy}{dt} - K_2 y \quad \rightarrow \quad y(t) = \iint_t \left\{ -K_1\frac{dy}{dt} - K_2 y \right\} dt \qquad (1)$$

Poslední rovnice vpravo nám dává návod na tvorbu modelu v Simulink. Jaké bloky budeme potřebovat?

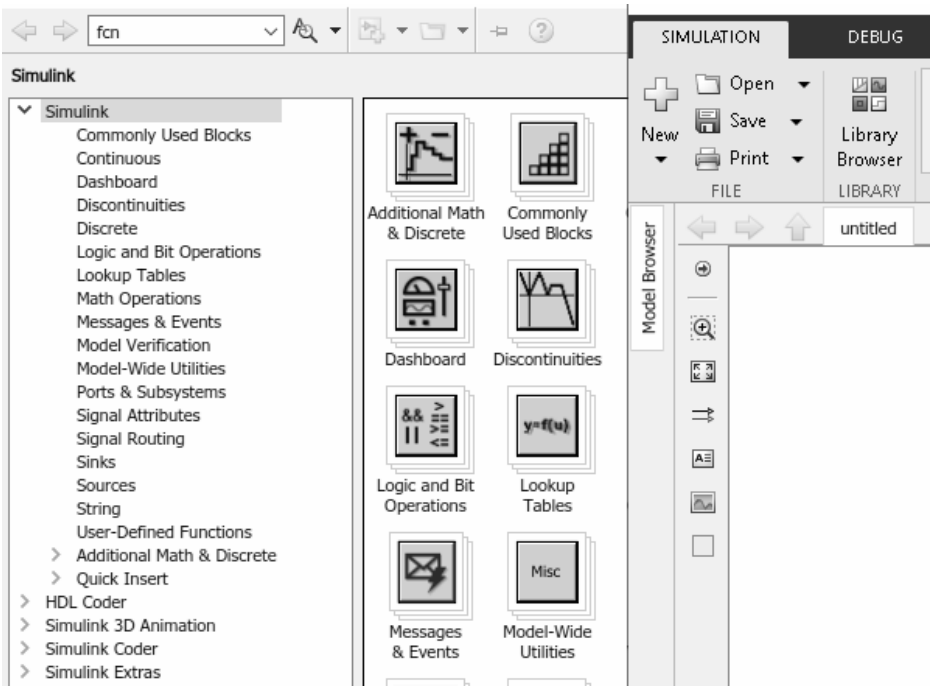
- dva bloky integrátoru (2x blok s názvem *Integrator*),
- dva bloky násobení konstantami K_1 a K_2 (2x blok *Gain*),
- jeden blok pro sčítání nebo odčítání (blok *Sum*).

Spustíme Simulink. Objeví se úvodní okno pode *obr.15.2*. Protože ještě nemáme náš modle vytvořený, zvolíme myší volbu *Blank Model* (zatím prázdné okno s modelem). Touto volbou se otevře nové zatím prázdné okno našeho budoucího modelu. Model uložte (*Save*/*Save As ...*) pod jménem *Sim_01.slx* (příponu *slx* doplní systém). Uložený model uvidíte ve své složce v okně *Current Folder*. Nyní myší klikněte v novém okně modelu nahoře na tlačítko *Library Browser* pro otevření knihovny bloků. Prostředí nyní vypadá podle *obr.15.3*. Vlevo máme okno s rozsáhlou knihovnou bloků a vpravo náš zatím prázdný model.

Máme tedy spuštěný MATLAB, knihovnu Simulink a nový model *Sim_01.slx*.

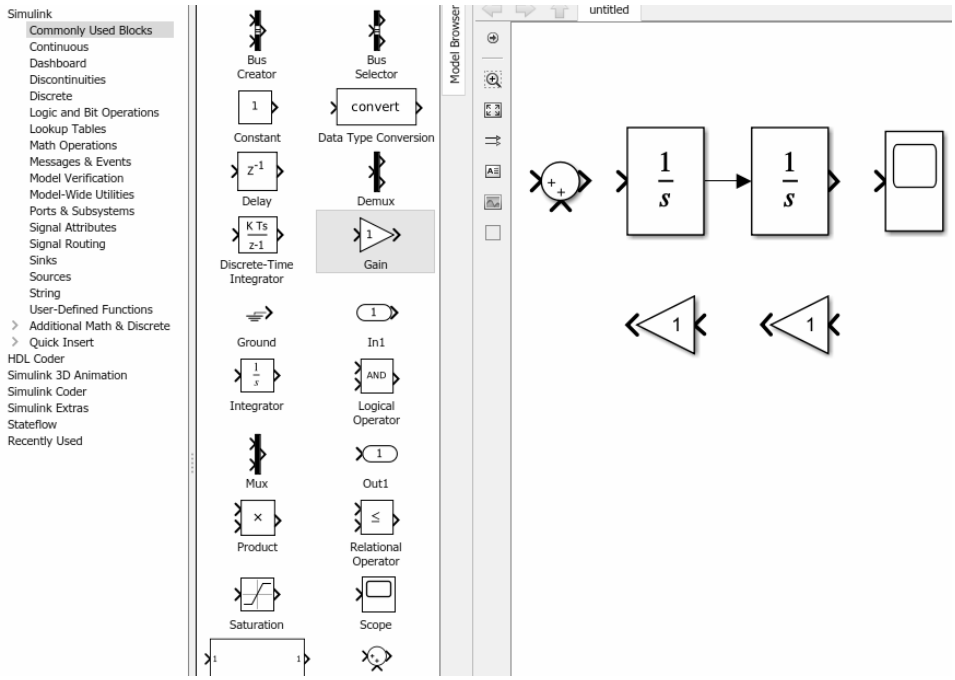


Obr.15.2. První spuštění prostředí Simulink.



Obr.15.3. Nové zatím prázdné okno modelu (vpravo) a knihovna bloků.

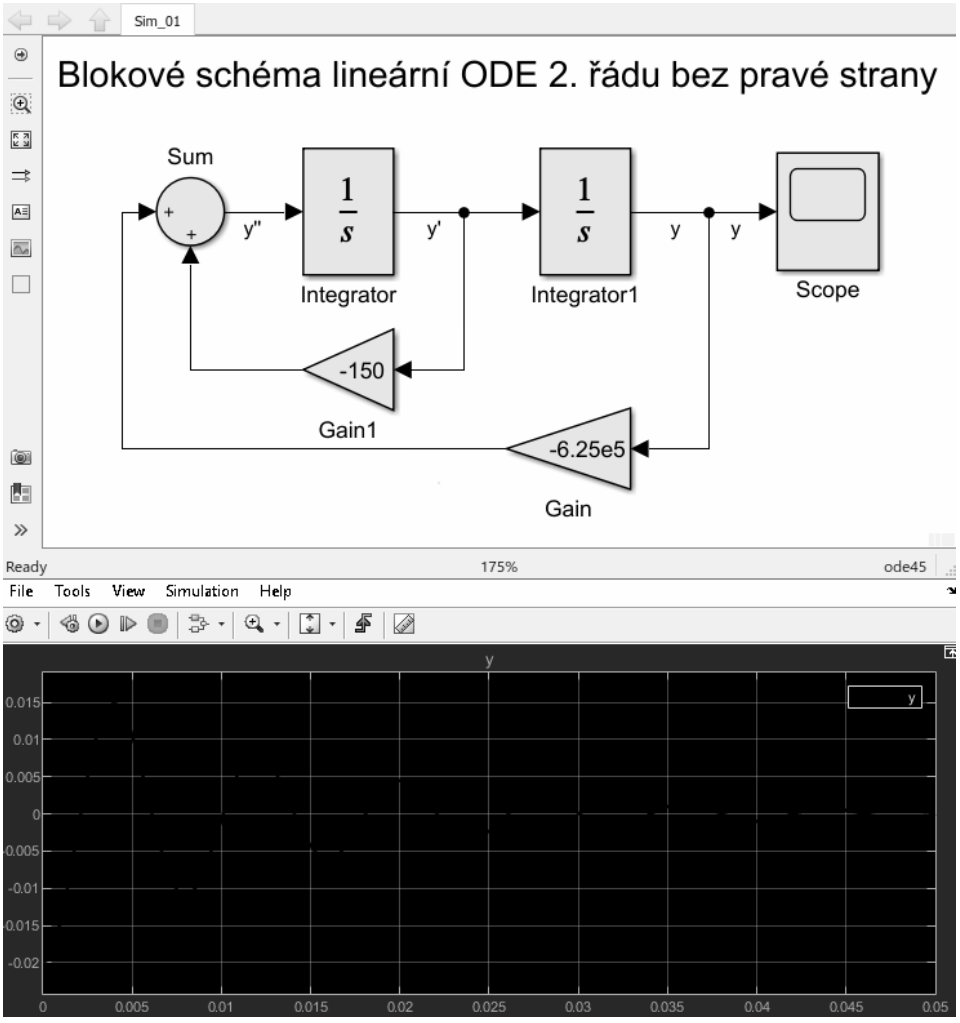
Naším úkolem bude nyní najít výše popsané bloky v knihovně, myší je přetáhnout do okna modelu, poklepáním na bloky zapsat jejich případné parametry, nastavit další parametry simulace a simulaci spustit.



Obr.15.4. Bloky přetažené myší z knihovny „Commonly Used Blocks“.

Na *obr.15.4* je rozbalená knihovna *Commnonly Used Blocks* a přetažené potřebné bloky do modelu. Myší jsou pro ilustraci propojené dva integrátory. Poklepáním na bloky *Gain* a *Gain1*, viz *obr.15.5*, lze nastavit konstanty podle zadání; model uložíte např. *Ctrl-s*. Barvu pozadí bloků můžete změnit v menu *Format*; zde také můžete zobrazit (skrýt) jména bloků, bloky rotovat (také *Ctrl-r*), měnit fonty textů, aj. Myší můžete změnit rozměry bloků a pomocí *Ctrl+* a *Ctrl-* (nebo tlačítkem pro *zoom* v levém svislém proužku okna modelu) měnit velikost modelu.

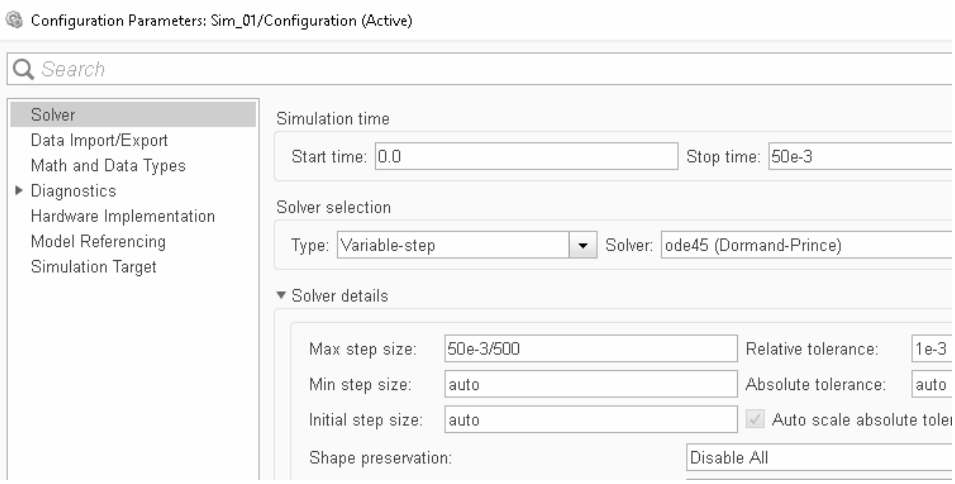
Myší propojíme zbylé bloky podle rovnice (1). Poklepáním na *dráty* je můžeme pojmenovat. Na výstupu je hledané řešení *y*, na vstupech integrátorů jsou jeho derivace. Počáteční podmínky nastavíme poklepáním na bloky integrátorů (hodnoty +0.02 a -0.02 v položkách *Initial Condition*). Blok *Scope* slouží pro sledování časového průběhu výstupu. Poklepáním myší na něj se zobrazí černé okno, určené pro časové průběhy, viz *obr.15.5*.



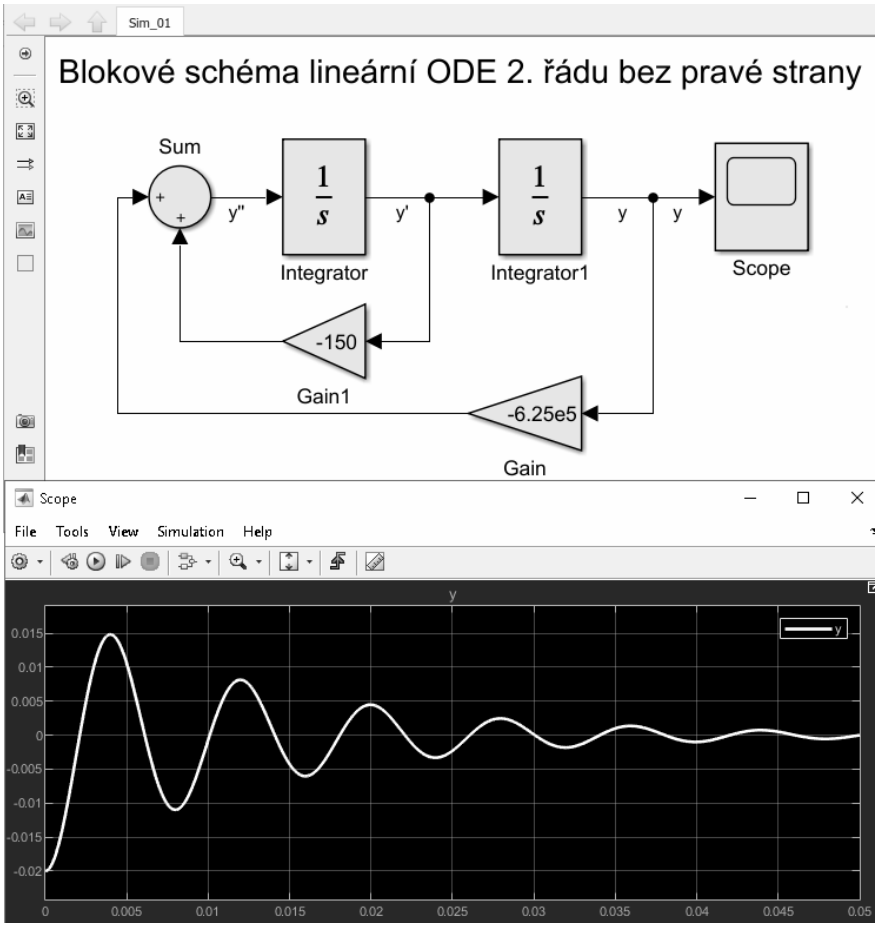
Obr.15.5. Model „Sim_01.slx" po propojení všech bloků a okno bloku „Scope".

Model podle *obr.15.5* je již připravený na nastavení hlavních parametrů simulace a její spuštění. Je třeba nastavit tři věci a to čas simulace, maximální krok při simulaci a vybrat numerickou metodou pro řešení.

Otevřeme nyní okno pro nastavení parametrů simulace. Učiníme tak kliknutím dovnitř modelu a volbou *Ctrl-e* nebo nahoře v záložce *Modeling* výběrem tlačítka *Model Settings*. Otevře se důležité okno podle *obr.15.6*. Vyplníme čas simulace na 50e-3 (50 ms) a maximální krok na 500 úseků na časové ose (501 vzorků). Jako typ numerické metody (*Solver*) vybereme *ode45* anebo *auto*. Simulaci spustíme tlačítkem *Run* nahoře v záložce *SIMULATION* anebo *Ctrl-t*, viz *obr.15.7*.



Obr.15.6. Nastavení parametrů simulace.



Obr.15.7. Simulace modelu Sim_01.slx.

Webové stránky v ČR a SR

https://www.humusoft.cz/ - stránky prodejce systému MATLAB
https://www.humusoft.cz/events/www-seminars/ - online semináře
https://www.humusoft.cz/matlab/products/ - aplikační knihovny
https://www.humusoft.cz/matlab/licensing/ - licence MATLAB
http://www.matlab.sk - fórum na téma MATLAB

Webové stránky tvůrce systému MATLAB

https://www.mathworks.com/ - stránky tvůrce MATLAB v USA
https://www.mathworks.com/matlabcentral/ - zkušenosti, diskuse
https://www.mathworks.com/company/aboutus/policies_statements/trademarks.html - přehled ochranných známek tvůrce MATLAB

MATLAB®
Pro začínající uživatele

prof. Ing. Karel Zaplatílek, Ph.D.