Laboratorij i vještine - MATLAB

Simulink

Željko Ban
Mato Baotić
Jadranko Matuško

Fakultet elektrotehnike i računarstva

2014/2015

Simulink

- Simulink je grafički alat unutar MATLAB programskog sustava koji omogućuje modeliranje, simulaciju i analizu dinamičkih sustava
- **Dinamički sustav** – sustav čije stanje u nekom trenutku određeno ne samo pobudom već i stanjima u kojima je sustav bio u prethodnim trenucima.
- Podjela dinamičkih sustava:
  - Kontinuirani dinamički sustav
  - Diskretni dinamički sustav
  - Hibridni sustav – kombinacija dva prethodna tipa

Opis dinamičkih sustava

- Polazište kod modeliranja dinamičkih sustava u Simulinku je skup (općenito nelinearnih) diferencijalnih jednadžbi:

$$\dot{x}(t) = f(x(t), u(t)), \quad (1)$$

$$y(t) = g(x(t), u(t)), \quad (2)$$

- odnosno jednadžbi diferencija:

$$x(k+1) = f(x(k), u(k)), \quad (3)$$

$$yk) = g(x(k), u(k)), \quad (4)$$

Simulacija dinamičkih sustava u Simulinku

- Kako simulirati kontinuirani sustav na digitalnom računalu?
- Simulink koristi postupke numeričke integracije za numeričko rješavanje diferencijalne jednadžbe,
- Unutar Matlaba implementirano više metoda numeričke integracije (solvera).
- Solveri unutar Matlaba se mogu podijeliti u dvije skupine:
  - solveri s konstantnim korakom integracije, pri čemu je korak integracije konstantan tijekom cijele simulacije,
  - solveri s promjenjivim korakom integracije, pri čemu je korak integracije adaptivan i mijenja se tijekom simulacija na način da se zadrži predefinirana razina točnosti simulacije.

Osnovne akcije u Simulinku

- Pokretanje simulinka:
  - Naredbom `>>simulink` u komandnom prozoru
  - Klikom na ikonu simulinka u Matlabu
- Stvaranje novog modela:
  - File $\rightarrow$ New (iz simulinka)
- Otvaranje postojećeg simulink modela:
  - File $\rightarrow$ Open (iz simulinka)
  - Naredbom `>>ime_dat.mdl` (iz Matlab komandnog prozora)
- Spremanje simulink modela
  - File $\rightarrow$save (iz simulinka)

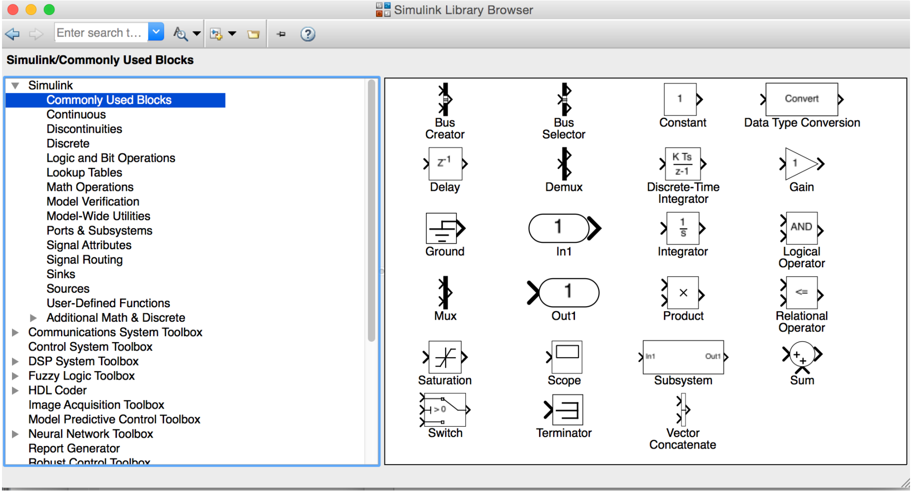
Izgradnja simulacijske sheme u Simulinku

Osnovni koraci

- Dodavanje blokova iz Simulink biblioteke blokova (Drag'n'Drop princip),
- Povezivanje blokova pomoću linija koje predstavljaju signale
- Podešavanje parametara simulacije kao što su npr. trajanje simulacije, tip numeričke integracije, točnost simulacije i sl.
- Pokretanje simulacije

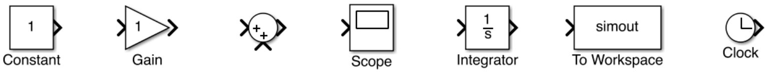
Izgradnja simulacijske sheme u Simulinku

Biblioteka blokova



Izgradnja simulacijske sheme u Simulinku

Osnovni blokovi



- `Constant` – definiranje konstantnog signala
- `Gain` – množenje signala s konstantom
- `Sum` – zbrajanje (oduzimanje) više signala
- `Scope` – grafički prikaz signala u vremenu (osciloskop)
- `Integrator` – integriranje ulaznog signala
- `To Workspace` –spremanje signala u varijablu radnog prostora
- `Clock` – signal vremena

Simulacija dinamičkih sustava u Simulinku

Primjer: Sustav drugog reda

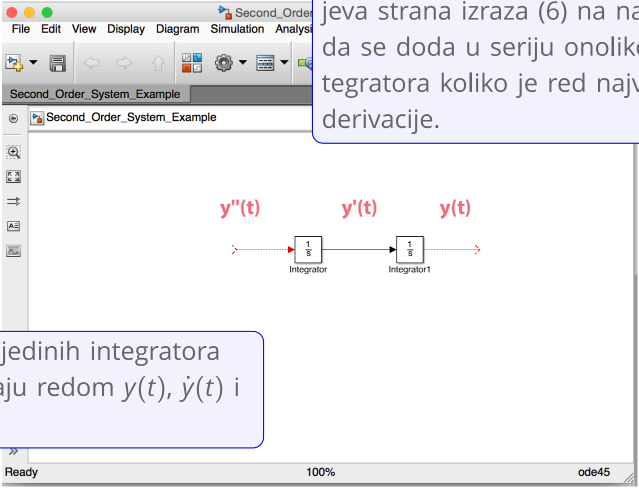
- Razmotrimo sustav drugog reda opisan sljedećom diferencijalnom jednadžbom:

$$\ddot{y}(t) + 5\dot{y}(t) + 6y(t) = u(t) \tag{5}$$

- Prvi je korak da se diferencijalna jednadžba zapiše na način da se na lijevoj strani jednakosti nalazi najveća derivacija varijable $y(t)$, tj. $\ddot{y}(t)$ u konkretnom primjeru, a lijevo od znaka jednakosti svi ostali članovi.

$$\ddot{y}(t) = u(t) - 5\dot{y}(t) - 6y(t) \tag{6}$$

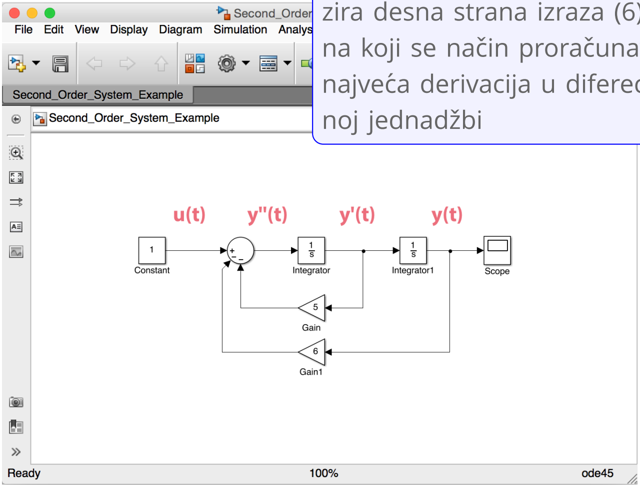
Izgradnja simulacijske sheme u Simulinku



U prvom se koraku realizira lijeva strana izraza (6) na način da se doda u seriju onoliko integratora koliko je red najveće derivacije.

Izlazi iz pojedinih integratora predstavljaju redom $y(t)$, $\dot{y}(t)$ i $\ddot{y}(t)$

Izgradnja simulacijske sheme u Simulinku

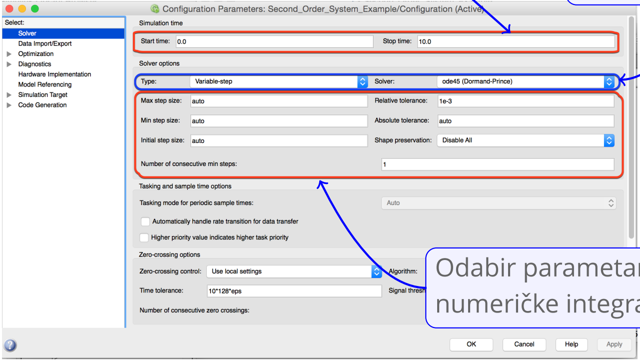


U narednom se koraku realizira desna strana izraza (6), tj. na koji se način proračunava najveća derivacija u diferecijalnoj jednadžbi

Podešavanje parametara simulacije u Simulinku

Podešavanje vremena simulacije

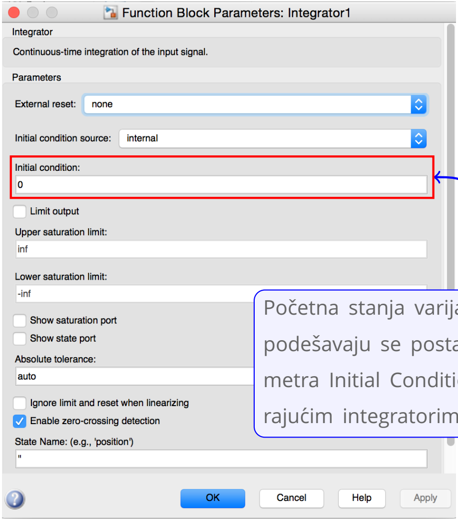
Odabir postupka numeričke integracije



Odabir parametara postupka numeričke integracije

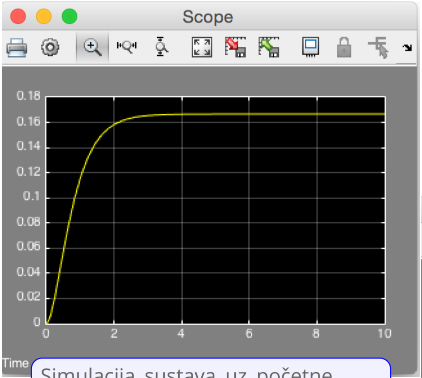
Podešavanje parametara simulacije u Simulinku

Postavljanje početnih uvjeta simulacije



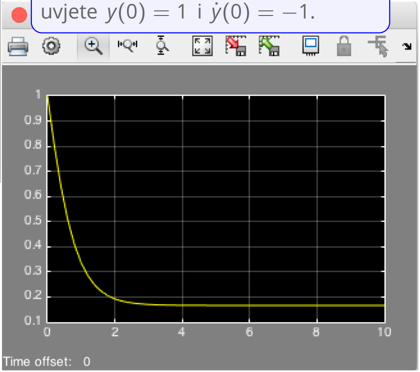
Početna stanja varijabli stanja podešavaju se postavljanjem parametra Initial Condition na odgovarajućim integratorima

Praćenje simulacije u Simulinku



Simulacija sustava uz početne uvjete $y(0) = 0$ i $\dot{y}(0) = 0$.

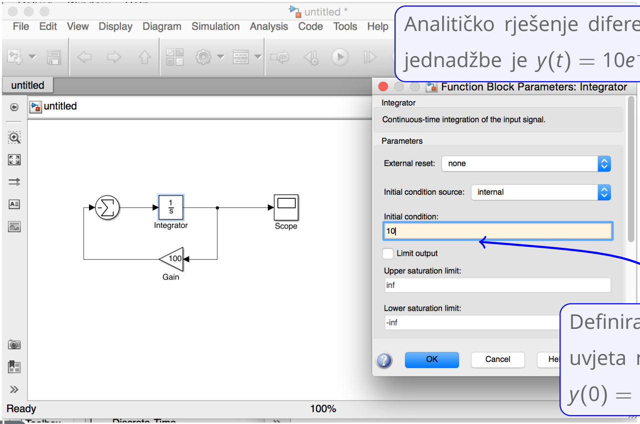
Simulacija sustava uz početne uvjete $y(0) = 1$ i $\dot{y}(0) = -1$.



Stabilnost postupka numeričke integracije

Primjer: Linearni sustav prvog reda

- Potrebno je simulirati homogeni sustav prvog reda $\dot{y}(t) + 100y(t) = 0$, $y(0) = 10$ uz korištenje Euleovog unazadnog postupka numeričke integracije.



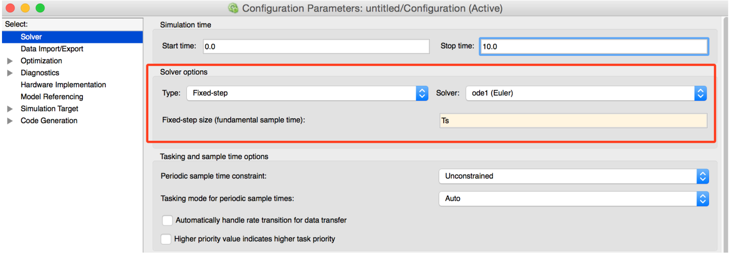
Analitičko rješenje diferencijalne jednadžbe je $y(t) = 10e^{-100t}$

Definiranje početnog uvjeta na integratoru $y(0) = 10$

Stabilnost postupka numeričke integracije

Primjer: Linearni sustav prvog reda

- Odabir postupka numeričke diskretizacije (s nepromjenjivim vremenom integracije, ode1(Euler)), i postavljanje koraka integracije.
- Eulerov unaprijedni postupak numeričke integracije za sustav $\frac{dy}{dt} = f(y, t)$ dan je relacijom $y(t_{k+1}) = y(t_k) + f(y(t_k), t_k)h$, gdje je h korak integracije.



Stabilnost postupka numeričke integracije

Primjer: Linearni sustav prvog reda

- Uzevši u obzir analitičko (stvarno) rješenje sustava:

$$y(t) = y(0)e^{-ct}, \tag{7}$$

i njegovim uvrštenjem u izraz za unaprijedni Eulerov postupak numeričke integracije $y(t_{k+1}) = y(t_k) + f(y(t_k), t_k) \cdot h$ dobiva se sljedeća jednadžba diferencija (uz pokratu $y(k) \triangleq y(t_k)$):

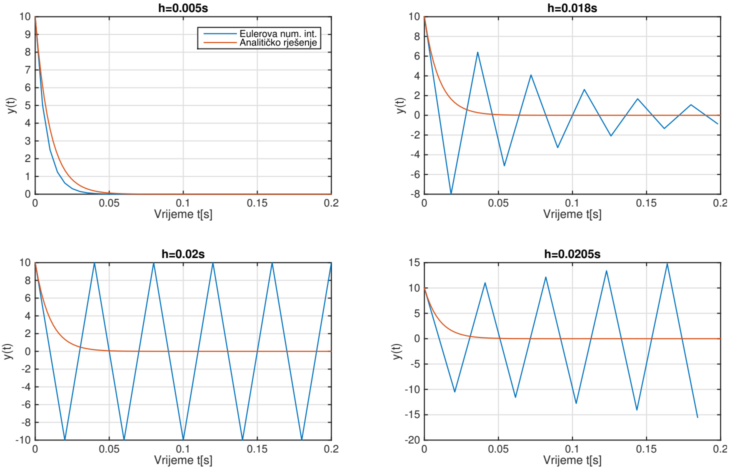
$$y(k+1) = y(k) - y(0)ce^{-ct}h = y(k) - hc \cdot y(k) = (1 - hc)y(k) \tag{8}$$

- Nužan uvjet za stabilnost postupka numeričke integracije:

$$\boxed{|1 - hc| \leq 1 \implies h \leq \frac{2}{c}} \tag{9}$$

Stabilnost postupka numeričke integracije

Primjer: Linearni sustav prvog reda



Stabilnost postupka numeričke integracije

Primjer: Linearni sustav prvog reda (nazadni Eulerov postupak)

- Ako sada stvarno analitičko (stvarno) rješenje sustava:

$$y(t) = y(0)e^{-ct}, \tag{10}$$

uvrstimo u izraz za unazadni Eulerov postupak numeričke integracije $y(t_{k+1}) = y(t_k) + f(y(t_{k+1}), t_{k+1}) \cdot h$ dobiva se sljedeća jednadžba diferencija (uz pokratu $y(k) \triangleq y(t_k)$):

$$y(k+1) = y(k) - y(0)ce^{-ct_{k+1}}h = y(k) - hc \cdot y(k+1) \tag{11}$$

tj.

$$y(k+1) = \frac{y(k)}{1+hc} \tag{12}$$

> Postupak numeričke integracije je stabilan uz sve iznose koraka integracije h (uz pretpostavku $c \geq 0$ tj. stabilan sustav).

Eksplicitni i implicitni postupci numeričke integracije

Eksplicitni postupak

$$y_{k+1} = y_k + h \cdot f(y_k, t_k)$$

$$\begin{aligned} y_1 &= y_0 + h \cdot f(y_0, t_0) \\ y_2 &= y_1 + h \cdot f(y_1, t_1) \\ &\vdots \end{aligned}$$

Stabilnost ovisi o iznosu koraka integracije

Implicitni postupak

$$y_{k+1} = y_k + h \cdot f(y_{k+1}, t_{k+1})$$

$$\begin{aligned} y_1 &= y_0 + h \cdot f(y_1, t_1) \\ y_2 &= y_1 + h \cdot f(y_2, t_2) \\ &\vdots \end{aligned}$$

Bezuvjetno stabilan postupak

Simulaciju krutih sustava

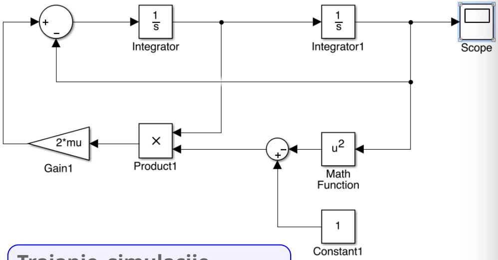
Primjer: unazadni i naprijedni Eulerov postupak

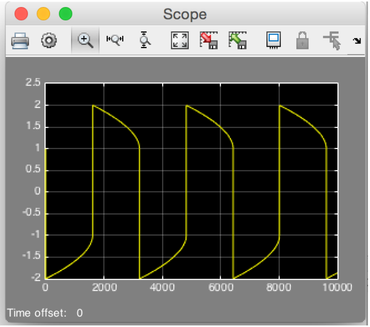
- Kruti sustavi su sustavi koji sadržavaju više, po dinamici, jako različitih dinamika.
- Ako se koristi eksplicitni postupak numeričke integracije (npr. predefinirani ODE45(Dormand-Prince)) tada najbrža dinamika uvodi ograničenja na maksimalni korak integracije s obzirom iz razloga stabilnosti samog postupka numeričke integracije.
- Najsporija dinamika određuje trajanje simulacije, te kao rezultat simulacija može biti dugotrajna.
- Rješenje je u korištenju implicitnih postupaka numeričke integracije (npr. ODE15s), budući da kod njih stabilnost ne ovisi o koraku integracije pa se po potrebi može povećati i time smanjiti trajanje simulacije.
- Nedostatak implicitnih postupaka je da se u svakom koraku mora razrješavati implicitna jednadžba (npr. Newton-Raphsonovim postupkom).

Simulaciju krutih sustava

Primjer: Van der Polov oscilator

Van der Polov oscilator je nelinearna diferencijalna jednadžba drugog reda $\ddot{y}(t) - 2\mu(1 - y^2(t))\dot{y}(t) + y(t) = 0$





Trajanje simulacije

ODE45: 62.046738 s

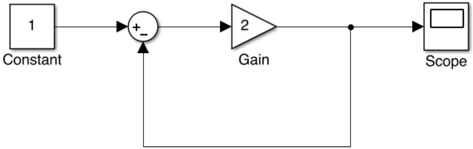
ODE15s: 0.113071 s

Simulacija obavljena uz $\mu = 1000$, $y(0) = 2$, $\dot{y}(0) = -1$

Algebarske petlje

- Pojavljuju se kod kod blokova koji imaju svojstvo direktnog proslijeđivanja ulaznog signala. Ukoliko se izlaz iz takvog bloka ponovo vraća kao ulaz u blok dolazi do pojave algebarske petlje.
- Trivijalni primjer algebarske petlje prikazan je na slici ispod.

Razvidno je da je rješenje u ovom slučaju trivijalno, tj. da se umjesto implicitno $y(k) = 2(u(k) - y(k))$ zapiše eksplicitnom tj. kao $y(k) = \frac{2}{3}u(k)$



Algebarske petlje

- U stvarnosti algebarske petlje se pojavljuju u mnogo složenijim (nelinearnim) oblicima uključujući i postojanje višestrukih rješenja.
- U takvim situacijama u pravilu se neće moći problem algebarske petlje riješiti transformacijom petlje u eksplicitni oblik.
- Ovisno o postavkama simulacije Simulink pojavu algebarskih petlji:
  - Ignorira
  - Upozorava porukom u komandnom prozoru
  - Zaustavlja simulaciju
- U prva dva slučaja poziva se u svakom koraku programska rutina za rješavanje alg. petlji koji usporava izvođenje simulacije (Newton-Raphsonov postupak).

Algebarske petlje

Rješavanje algebarskih petlji

Načelno se problemu rješavanja alberskih petlji može pristupiti na neki od sljedećih načina:

- Modifikacija simulacijske shema na osnovi rješenja alg. jednadžbe, što je u realnim situacija rijetko primjenjivo.
- Dodavanje memorijskog elementa `Memory` koji unosi kašnjenje u trajanju jednog integracijskog koraka. Metoda je primjenjiva samo kada je pojačanje petlje manje od 1.
- Na signalne linije može se dodati blok `IC` (početni uvjet) kako bi se osiguralo da solver krene iz "dobrih" početnih uvjeta (korisno u slučaju da algebarska petlja ima više od jednog rješenja).

Algebarske petlje

Primjer: Algebarska petlja s višestrukim rješenjima

- Razmotrimo algebarsku petlju opisanu sljedećoj jednadžbom:

$$y(k) = \frac{1}{5}\left(\frac{1}{4}u(k) - y^2(k)\right).$$

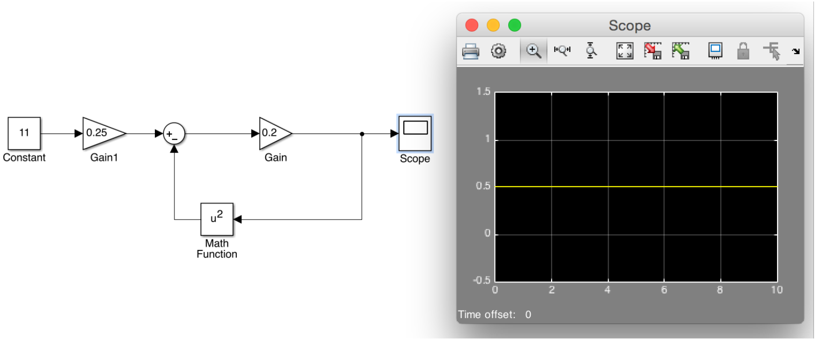
- Navedena algebarska petlja ima dva rješenja i to:

$$y_{1,2}(k) = \frac{-5 \pm \sqrt{25 + u(k)}}{2}$$

- Npr. za konstantan ulazni signal $u(k) = 11$ navedena rješenja iznose $y_1(k) = 0.5$ i $y_2(k) = -5.5$.

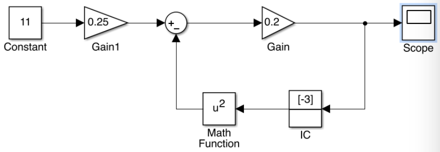
Algebarske petlje

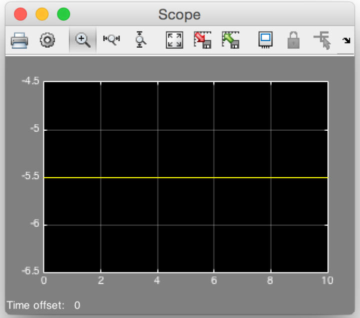
Primjer: Algebarska petlja s višestrukim rješenjima



Algebarske petlje

Primjer: Algebarska petlja s višestrukim rješenjima, dodan blok IC s početnim stanjem -3





Ovisno o iznosu početnog stanja u bloku IC rješenje petlje će konvergirati jednom od dva moguća rješenja (0.5 ili -5.5)

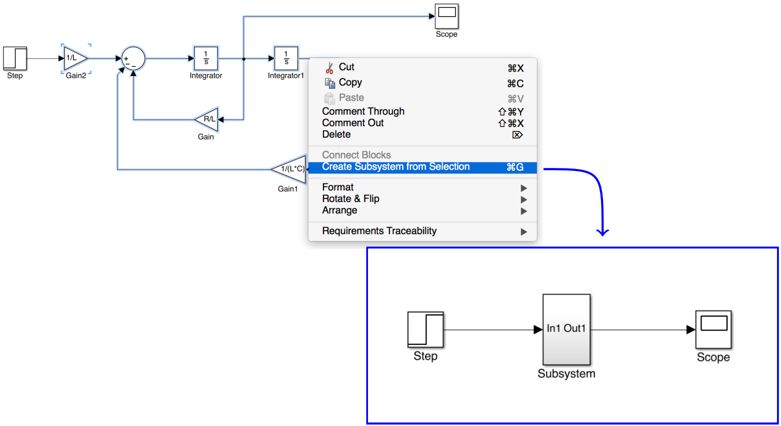
Podsustavi u Simulinku

- U pravilu je osnovna namjena podsustava u Simulinku je da se simulacijska shema učini čim preglednijom.
- Podsustavi se u Simulinku mogu kreirati na dva načina:
  - Dodavanjem bloka `Subsystem` te dodavanjem u njega potrebnih blokova skupa ulaznim i izlaznim portovima,
  - Iz već postojeće simulacijske sheme na način da se označi dio sheme od koje se želi stvoriti podsustav te odabirom opcije Edit-¿Create Subsystem.

Upravljanje simulacijom iz komandnog prozora

Postavljanje parametra bloka u simulacijskoj shemi

- `set_param('OBJ','PARAMETER','VALUE')`



Podsustavi u Simulinku

Podjela podsustava

Postoje dvije vrste podsustava u Simulniku i to:

- Virtualni podsustavi, pri čemu postojanje podsustava ni na koji ne utječe na način izvođenja simulacije. tj. na redosljed proračuna blokova. Predefinirana je postavka da je blok koji stvara virtualni.
- Nedjeljivi podsustavi, koje Simulink tretira kao novi blok i koji utječe na izvođenje simulacijske sheme na način kada jednom započne izvođenje operacija specificiranih u podsustavu tada se sve one moraju zavrđiti prije nego što se krene na izvođenje ostalih blokova.

Podsustavi u Simulinku

Podsustavi s uvjetnim izvršavanjem

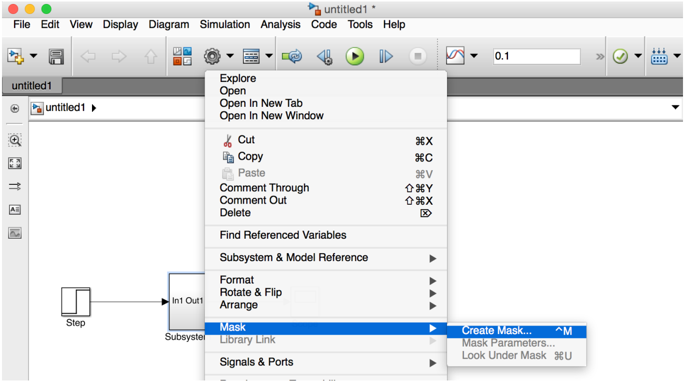
Osim već navedene namjene podsustava da se složena simulacijska shema grupira u manje logički povezane cjeline, oni se također mogu koristiti ako je da pojedini dijelovi simulacijske sheme izvršavaju uvjetno, te u tom pogledu razlikujemo:

- Podsustave čije je izvršavanje uvjetovanje razinom kontrolnog signala (engl. Enabled subsystem),
- Podsustave čije je izvršavanje uvjetovanje bridom kontrolnog signala (engl. Triggered subsystems),
- Podsustave čije je izvršavanje uvjetovanje bridom kontrolnog signala 1 i razinom kontrolnog signala 2.

Podsustavi u Simulinku

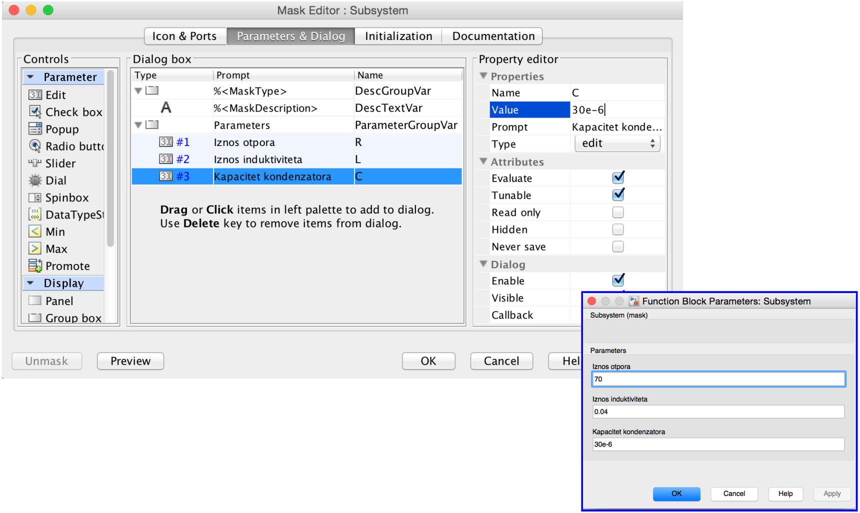
Maskiranje podustava

- Maskiranje podsustava se obavlja s ciljem njihovog jednostavnijeg korištenja od strane kranjneg korisnika, kolemu se preko maske omogućuje promjena samo određenih parametara podsustava.



Podsustavi u Simulinku

Maskiranje podustava



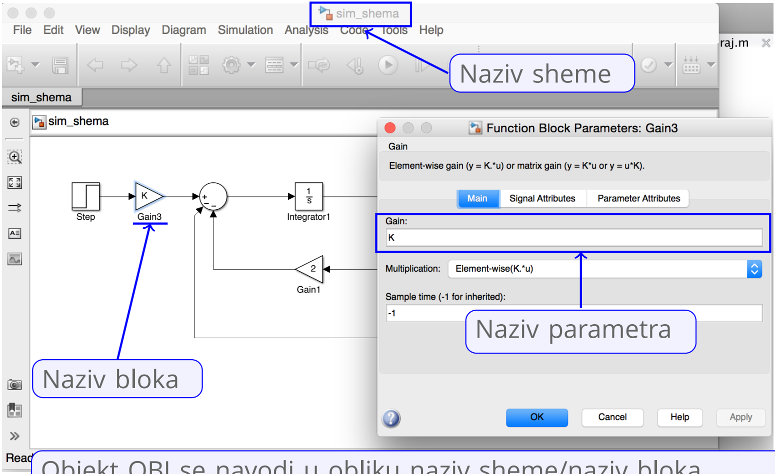
Upravljanje simulacijom iz komandnog prozora

- Ova funkcionalnost je bitna u slučajevima kada je simulaciju potrebno upravljati programski,
- Upravljanje simulacijom može se obavljati iz komandnog prozora kao i iz bilo koje m-funkcije.
- Osnovne funkcije za upravljanje simulacijom iz komandnog prozora su:
  - `sim('model','PARAMETER','VALUE',...)`
  - `set_param('OBJ','PARAMETER','VALUE')`
  - `get_param('OBJ','PARAMETER','VALUE')`
  - `add_block('SRC','DEST')`
  - `add_line('SYS','OPORT','IPORT')`

Upravljanje simulacijom iz komandnog prozora

Postavljanje parametra bloka u simulacijskoj shemi

- `set_param('OBJ','PARAMETER','VALUE')`



Objekt OBJ se navodi u obliku naziv_sheme/naziv_bloka

Upravljanje simulacijom iz komandnog prozora

Primjer

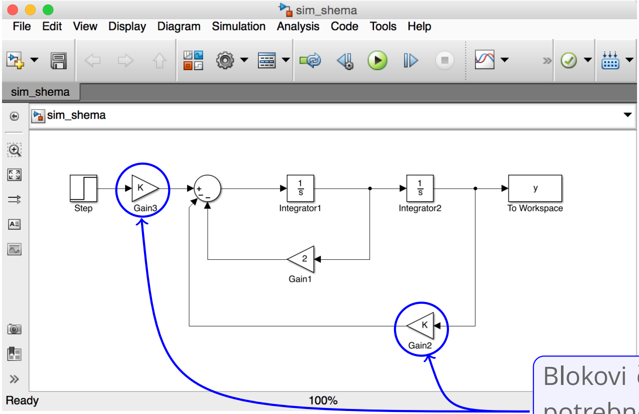
- Za sustav:

$$\ddot{y}(t) + 2\dot{y}(t) + Ky(t) = Ku(t), \tag{13}$$

  potrebno je odrediti krivulju ovisnosti nadvišenja u odzivu na jediničnu skokovitu promjenu ulaznog signala $u(t)$ o vrijednosti parametra K u intervalu [0,10].
- Što je potrebno napraviti:
  - Mijenjati pojačanje K od 1 do 10 s određenim korakom npr. 0.5,
  - Za svaku vrijednost pojačanja K potrebno je provesti simulaciju te iz odziva izračunati nadvišenje kao $\sigma = (y_{\mathrm{max}} - y_{ss})/y_{ss} * 100\%$

Upravljanje simulacijom iz komandnog prozora

Primjer



Blokovi čije je parametre potrebno mijenjati u svakom koraku

Upravljanje simulacijom iz komandnog prozora

Primjer

- Prijedlog rješenja

```
clear;
i=0;
for K=0:0.5:10
i=i+1;
    set_param('sim_shema/Gain2','Gain',mat2str(K));
    set_param('sim_shema/Gain3','Gain',mat2str(K));
    sim('sim_shema',10);
    sigma(i)=(max(y)-y(end))/y(end);
    k(i)=K;
end
plot(k,sigma);
```

Upravljanje simulacijom iz komandnog prozora

Primjer