Modelowanie Systemów dynamicznych – Adam Frydel

Automatyka i Robotyka II Rok, Sprawozdanie

„Podstawy automatyki”

Wstęp:

Po serii kilku zajęć podczas, których modelowaliśmy zachowanie parametrów obiektów w czasie za pomocą kodu w Matlabie przeszliśmy do Simulinka. Zajęcia polegały na przeniesieniu równań oraz opisania układu w sposób bloczkowy. Począwszy od prostych modeli skończyliśmy na modelu układu z rozdziału nr 4 konspektu 2. Modelowanie metodą bloczkową różni się znacznie od modelowania układu za pomocą macierzy i równań stanu. Jeżeli chodzi o proste układy to jest to znacznie szybsza i dogodniejsza metoda. Problem robi się gdy równań stanu jest więcej lub pojawia się trzecia pochodna np. zryw. Wówczas układ staje się nieczytelny, a szukanie w nim ewentualnych błędów kiedy symulacja przejdzie, ale zachowa się niepoprawnie nie należy do najłatwiejszych zadań. Metodę bloczkową jednak można bardzo szybko wprowadzić z równań stanu.

```
k = 6 ;
m = 14;
X0 = 0.1;
F = 1;
A = [0 1 ; -k/m 0];
B = [0 ; 1/m];
F4 = 1000;
M = 1000;
alfa = 500;
c = 400;
```

W miejscu powyżej zapisałem wszystkie zmienne potrzebne do wykonania zadań. Począwszy od tych które były potrzebne do rozwiązania trzeciego modułu jak i tych do wykonania konspektu z rozdziału 4 konspektu nr 2. Zmienne zdefiniowałem w skrypcie aby uniknąć ich "ucieczki" przy ponownym uruchomieniu Matlaba gdybym je po prostu wpisał do terminala.

Zgodnie z zaleceniami prowadzącego zamieszczam wykonane modele używając w tym celu ścieżki Format -> Screenshot -> Send Windows Metafile to Clipboard.

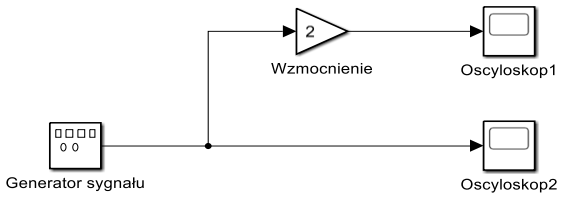
ROZDZIAŁ I

Celem tego ćwiczenia było zapoznanie się z pakietem SIMULINK, który jest rozszerzeniem programu MATLAB. Dzięki temu środowisku mogłem poznać proste symulacje i je przeanalizować. W trakcie ćwiczenia poznałem podstawowe bloki dostępne w SIMULINK, takie jak oscyloskopy do wykreślania przebiegów, generatory sygnałów oraz bloki wzmacniające, całkujące i przechowujące równania stanu. Zdobyta wiedza pozwoliła mi wykonać II rozdział oraz w trzecim przeanalizować model masy zawieszonej na sprężynie, korzystając z różnych jego reprezentacji w środowisku SIMULINK.

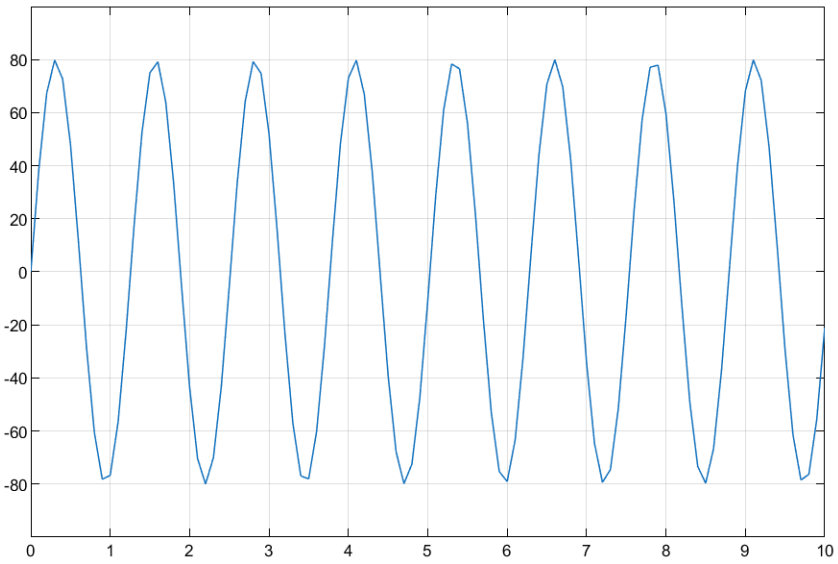
ROZDZIAŁ II

Model 1:

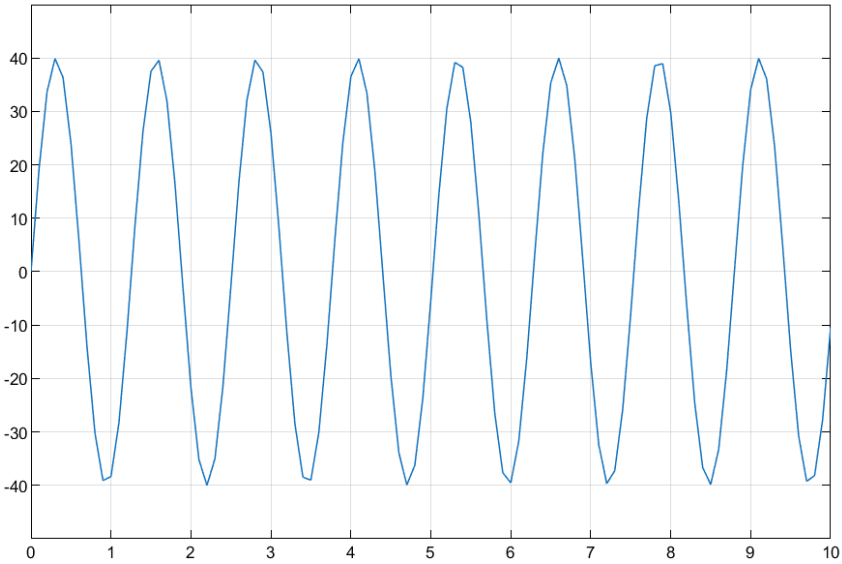
W tym zadaniu należało zaobserwować prostą odpowiedź sinusoidalną widzianą z perspektywy dwóch oscyloskopów. Do bloczka generującego sygnał podpiąłem oscyloskop oraz równolegle wzmocnienie. Do wzmocnienia podpiąłem drugi oscyloskop. Po wejściu w bloczek opisany przeze mnie jako Generator sygnału ustawiłem wartości amplitudy na 40 oraz częstotliwość na 5. Na wykresach widać że opcja ze wzmocnieniem cechuje się 2 razy większą amplitudą (w tym przypadku 80) niż wersja bez wzmocnienia.



Oscyloskop1 wersja z dwukrotnym wzmocnieniem

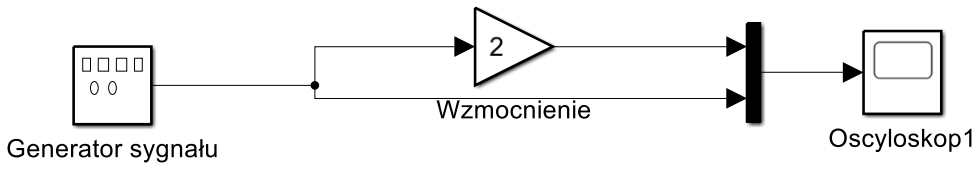


Oscyloskop2 – wersja bez wzmocnienia (zwykły przebieg sinusoidalny o amplitudzie 40)

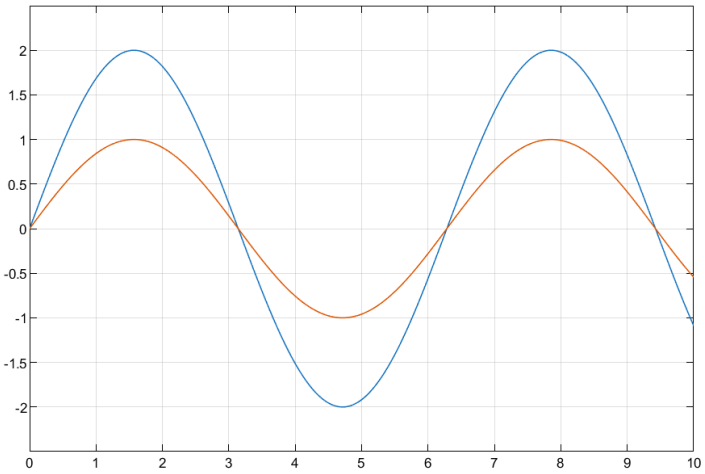


Model 2:

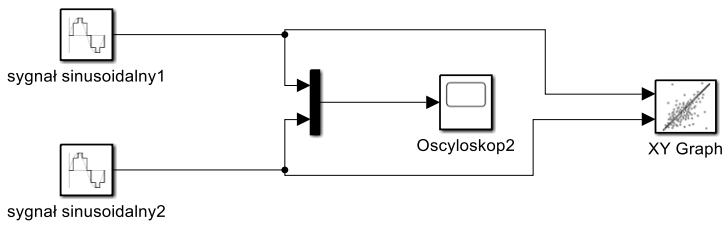
Zadanie było analogiczne do poprzedniego, jednak teraz zainstalowałem multiplekser, który pozwolił mi zobaczyć dwa wykresy na oscyloskopie. I tak w jednym oknie widać że sygnał wzmocniony, który jest oznaczony kolorem niebieskim ma dwa razy większą amplitudę niż sygnał oznaczony kolorem czerwonym. Jest to oczywiście spowodowane bloczkiem Gain oznaczonym u mnie jako Wzmocnienie gdzie przepuszczony sygnał posiada dwa razy większą amplitudę niż wpuszczony.



Niebieski wykres – sygnał wzmocniony, Czerwony wykres – sygnał oryginalny

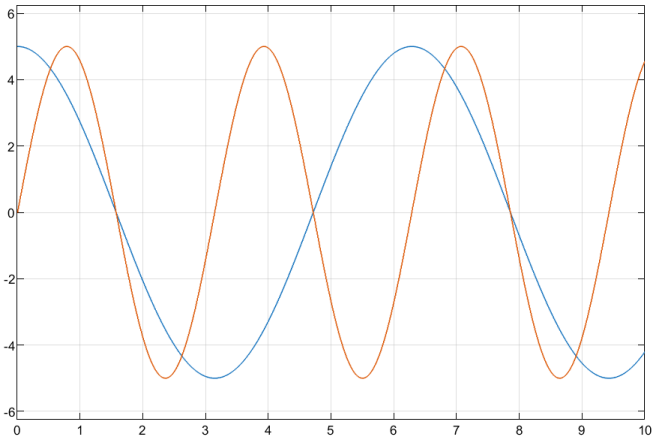


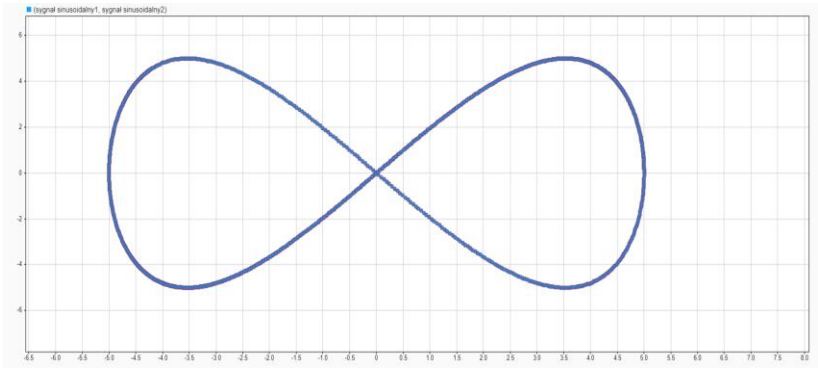
Model 3:



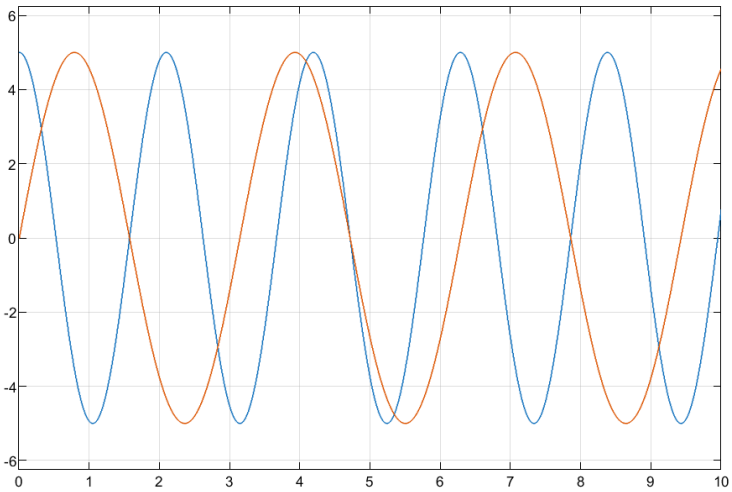
Powyższy model miał na celu zbudowanie układu do generowania krzywych Lissajous'a - są to krzywe parametryczne wykreślane przez punkt materialny wykonujący drgania harmoniczne w dwóch wzajemnie prostopadłych kierunkach – (źródło Wikipedia: https://pl.wikipedia.org/wiki/Krzywa_Lissajous). Chodziło o to aby tak zmienić parametry w bloczkach SinWave aby ukazały się poniższe kształty. Po wczytaniu się w dokumentację zamieszczoną na Wikipedii ustawiłem w pierwszej konfiguracji parametry a i b jako 1 oraz 2, a w drugiej jako 3 oraz 2. Parametry a oraz b ustawiałem w zakładce Frequency w ustawieniach generatora sygnału. Ważne było również zmniejszenie czasu próbkowania aby wykresy były ciągłe. Do bloczków generujących sygnał sinusoidalny dołączyłem multiplekser, którego wyjście połączyłem z oscyloskopem. Podpiąłem je również pod dwa wejścia bloczku rysującego wykresy.

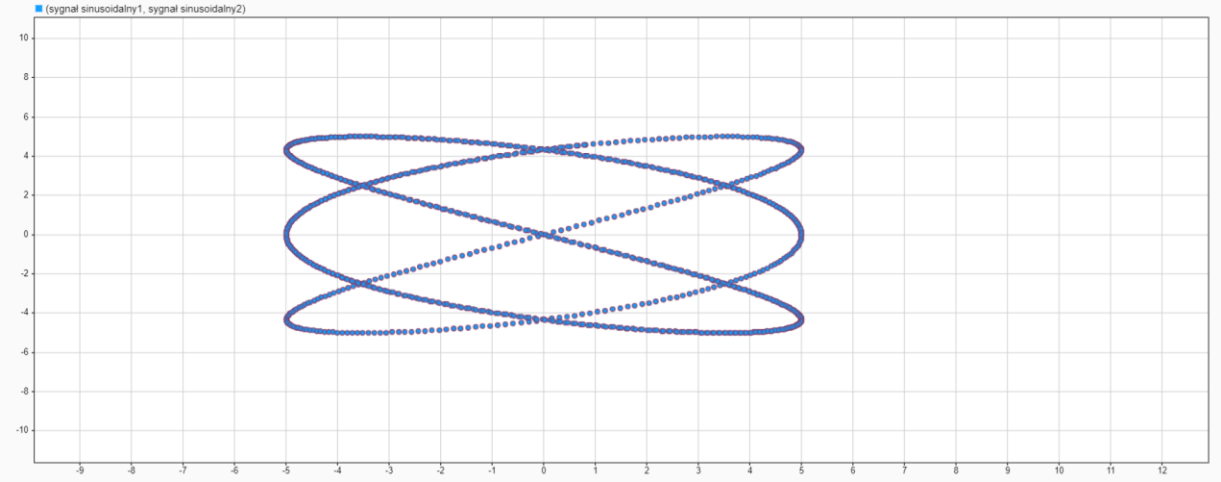
Dla parametrów a = 1 oraz b = 2





Dla parametrów a = 3 oraz b = 2

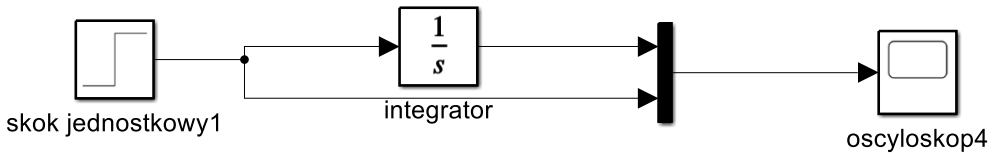




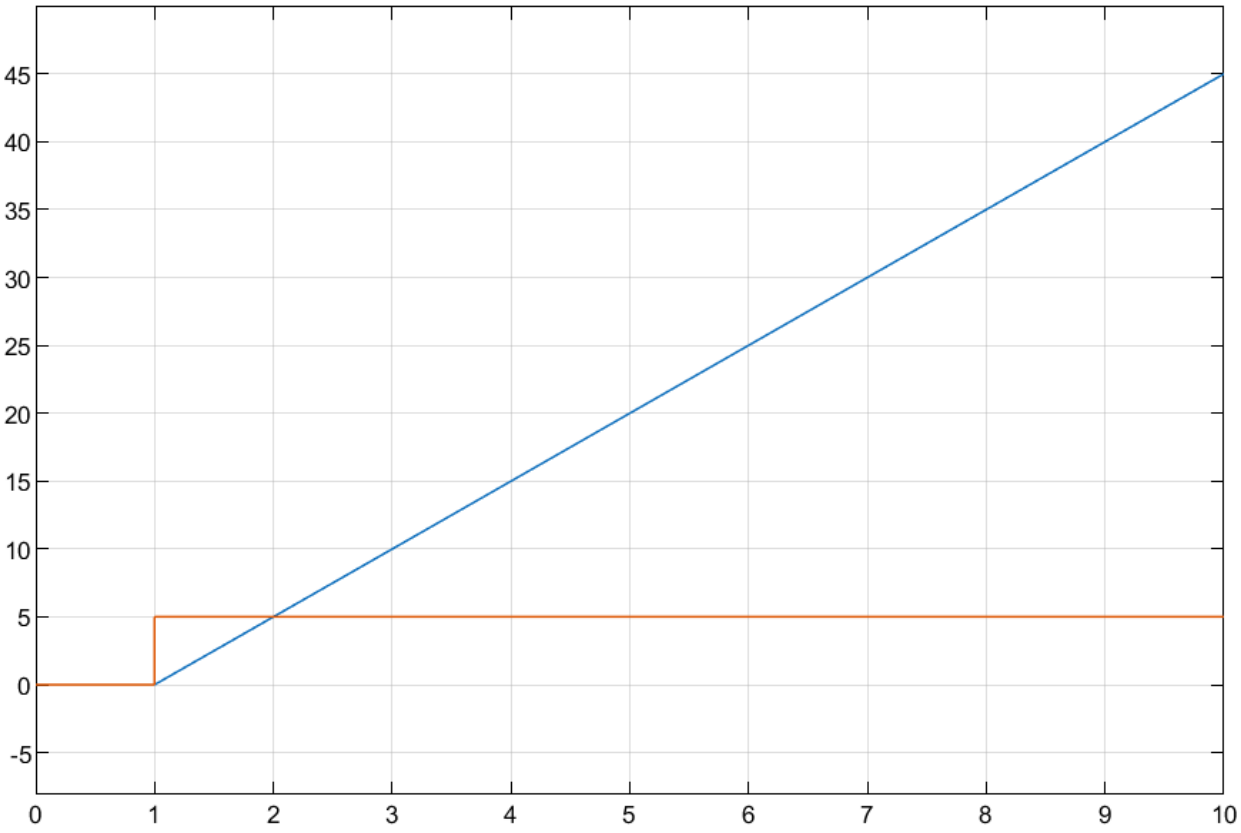
W warunkach stworzonych przez SIMULINKa byłem w stanie w sposób idealny odtworzyć zadane krzywe. Całość byłem w stanie otworzyć za pośrednictwem bloczka XY Graph, gdzie początkowo odtworzyłem pierwszą a potem drugą wersję powstałego wykresu przez nałożenie się funkcji o dwóch różnych częstotliwościach.

Model 4:

W tym module skorzystałem z dwóch nowych bloczków – step oraz integrator. Ten pierwszy pozwala mi ustawić czas oraz wielkość zadanego sygnału skokowego oraz sprawdzić odpowiedź układu na niego natomiast integrator pozwolił nadać wartość początkową badanego parametru np. położenia czy też obniżyć stopień pochodnej tegoż parametru. W bloczku Skok jednostkowy w modelu układu ustawiłem wartości Step time : 1 oraz Final value : 5.



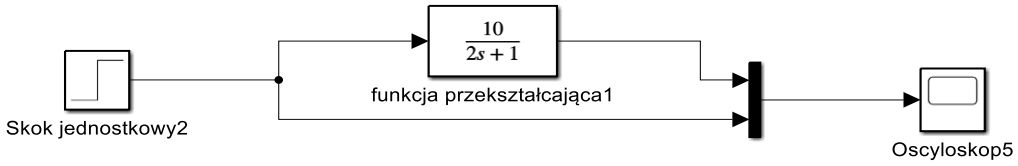
Układ składa się z bloczku Step, którego wyjście połączyłem równolegle z multiplekserem do jednego wejścia bez członu całkującego oraz do drugiego wejścia używając członu całkującego. Wyjście multipleksera jest obserwowane przez oscyloskop.



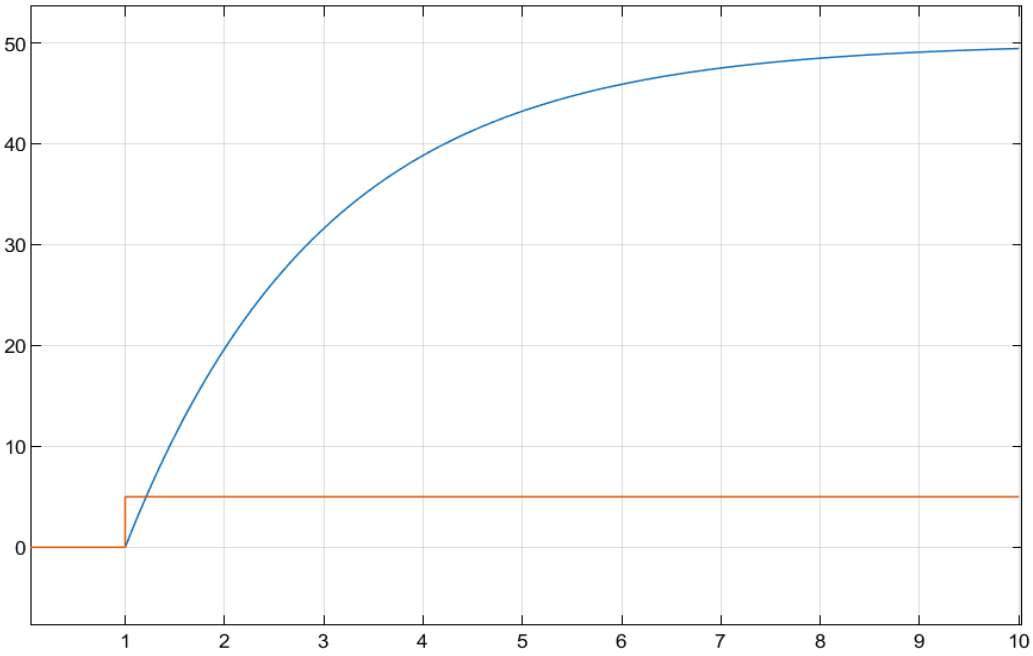
Układ prezentuje skok na kolor czerwony oraz w kolorze niebieskim odpowiedź skokową członu całkującego idealnego.

Model 5:

Różnica pomiędzy tym a poprzednim modelem jest taka, że teraz patrzymy jak zareaguje układ zadany transmitancją na skok jednostkowy.

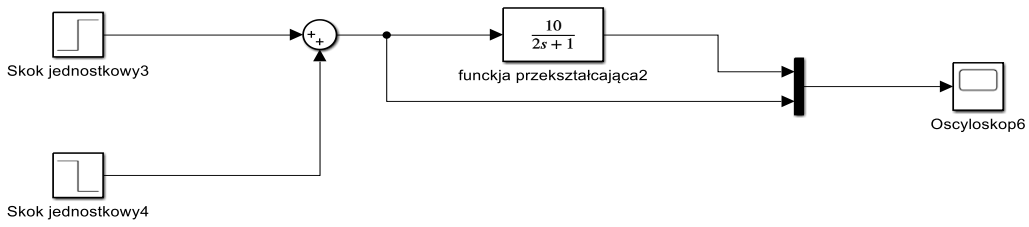


MATLAB pozwala na jej przechowywanie jako współczynniki licznika i mianownika lub obiekt typu transfer function, a Simulink umożliwia łatwą implementację takich modeli za pomocą bloku Transfer Fcn. Blok ten pozwala na bezpośrednie podanie współczynników transmitancji oraz sygnału sterującego, co daje możliwość wygodnego uzyskania odpowiedzi układu. Warto dodać, że Simulink ma dostęp do pamięci MATLABa, stąd



Na rysunku widzimy jak wpływa na zachowanie układu skok jednostkowy. W bloczku step ustawiłem wartości step time: 1 oraz final value :5.

Model 6:



Aby przeanalizować odpowiedź impulsową układu, należy zmodyfikować schemat. W Simulinku jest osobny bloczek do generowania sygnału impulsowego Discrete impulse natomiast zadanie

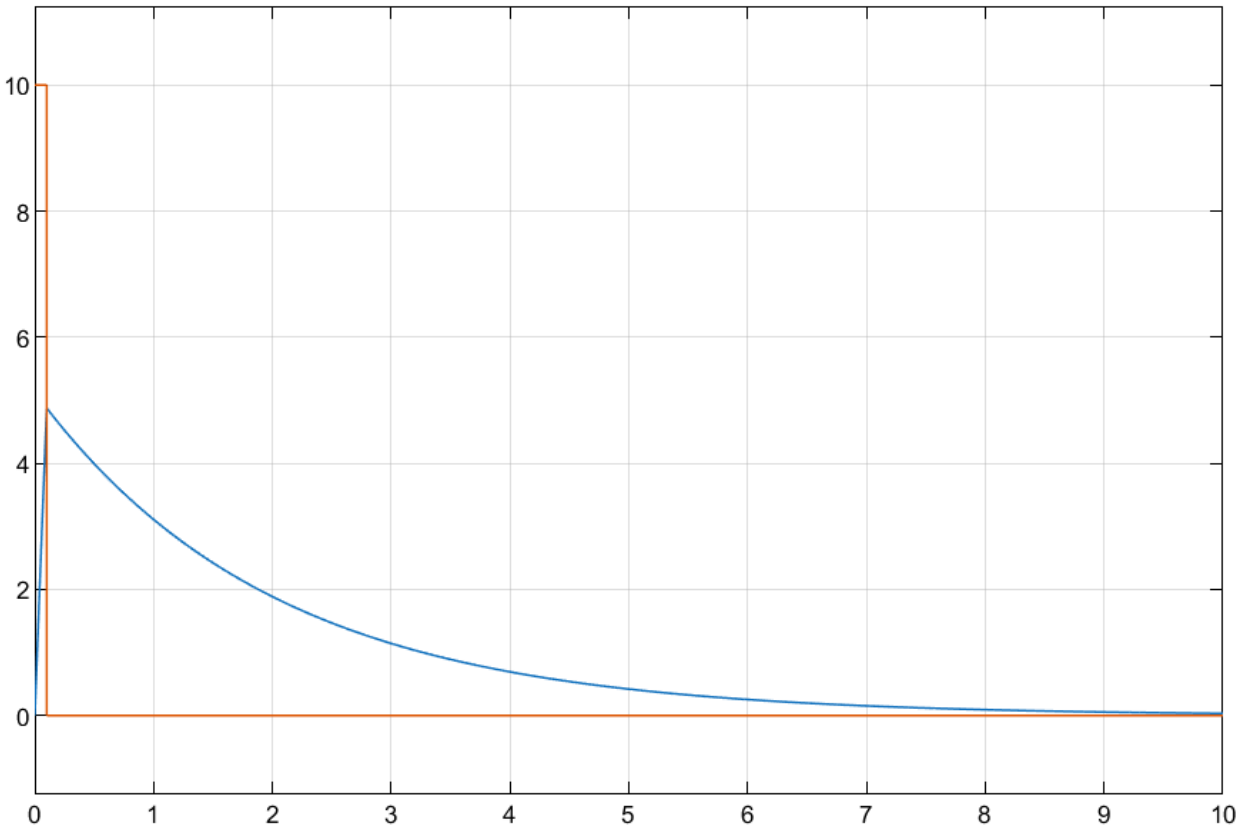
wymagało zapodania sygnałem step bardzo krótkiego skoku. Sygnał zapodany z dwóch bloczków step został zsumowany a następnie dołączony do multipleksera raz żeby na oscyloskopie pokazać impuls oraz na drugie wejście przechodząc przez transmitancję. Dzięki temu na wykresie widać przebiegi czasowe odpowiedz badanego układu.

Parametry bloków:

- Zbocze narastające: Step Time: 0, Initial Value: 0, Final Value: 10
- Zbocze opadające: Step Time: 0.1, Initial Value: 0, Final Value: -10

Dzięki zmniejszeniu kroku symulacji uzyskujemy dokładniejszy i bardziej płynny przebieg.

W taki sposób prezentuje się wynik powyższych działań:



Wykres posiada taki sam kształt w przypadku szukania odpowiedzi impulsowej w poprzednich konspektach.

ROZDZIAŁ III

W tym zadaniu za pomocą simulinka zamodelowałem bardziej złożony układ. Był to bloczek zawieszony na sprężynie o pewnej masie. Początkowo należało przyjąć wartości odpowiednich zmiennych :

- k = 6 N/m,

- m = 14 kg

- x0 = 0.1 m

- F = 1 N

Układ można opisać równaniem różniczkowym:

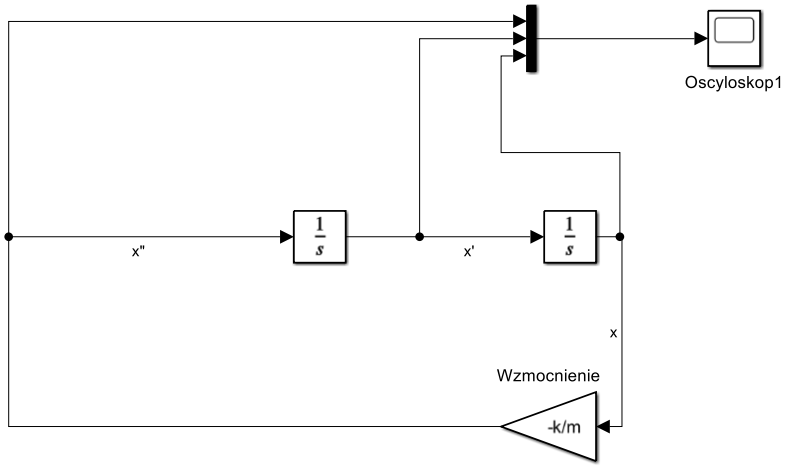
$mx\ddot{}+kx=F$

Obierając zmienne jako położenie oraz prędkość. Równanie stanu będzie postaci:

$[x1\dot{}\ x2\dot{}]=[01;\ k/m\ 0][x1\ x2]+[0\ 0]u$

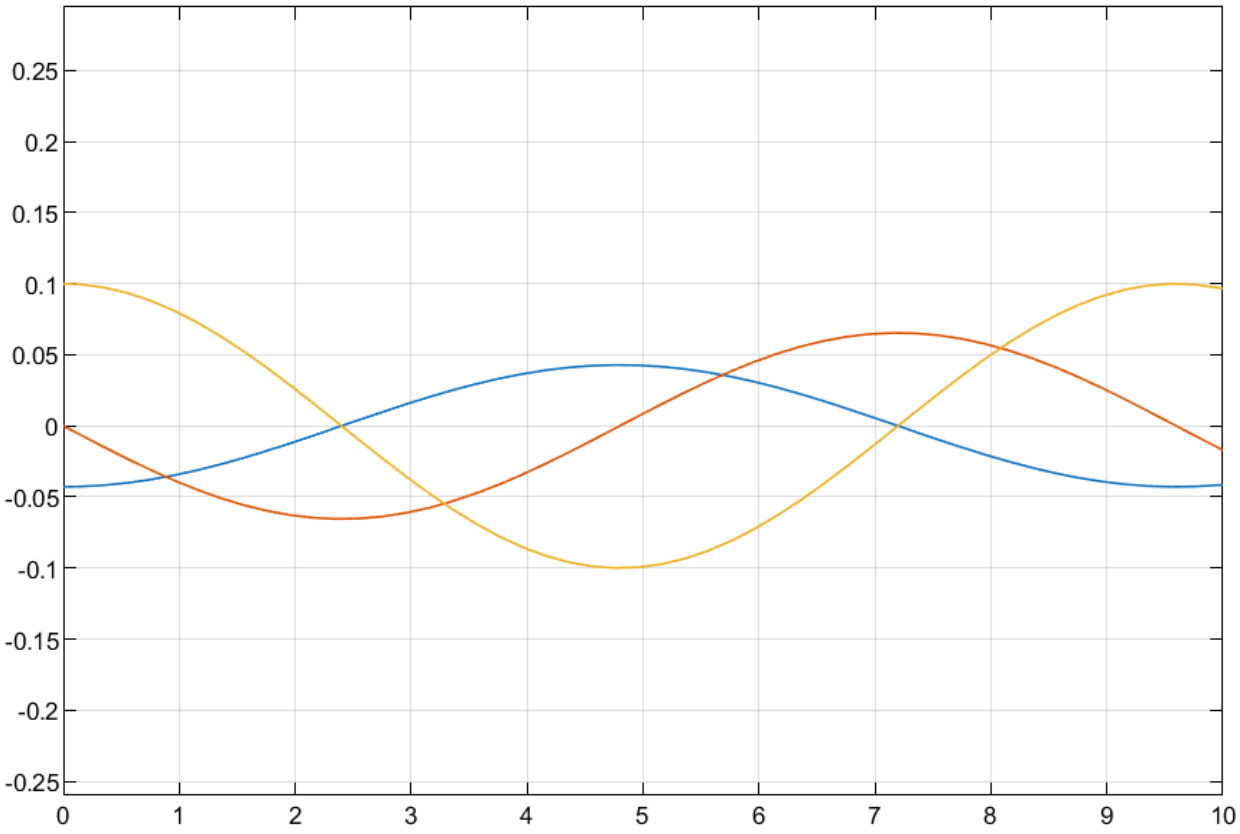
Sposób pierwszy:

a) Wersja swobodna:



Do zbudowania tego modelu użyłem wyłącznie znanych mi bloków, w tym dwóch elementów całkujących. Dzięki temu mogłem przejść od wartości docelowego przyspieszenia do wyznaczenia prędkości i przemieszczenia, które są kluczowe do dalszych obliczeń. Odpowiednie współczynniki w równaniu różniczkowym zostały zaimplementowane przez pomnożenie sygnałów z wykorzystaniem

wzmacniaczy. Zmienne zapisałem symbolicznie w skrypcie MATLAB, a następnie przypisałem im konkretne wartości liczbowe.

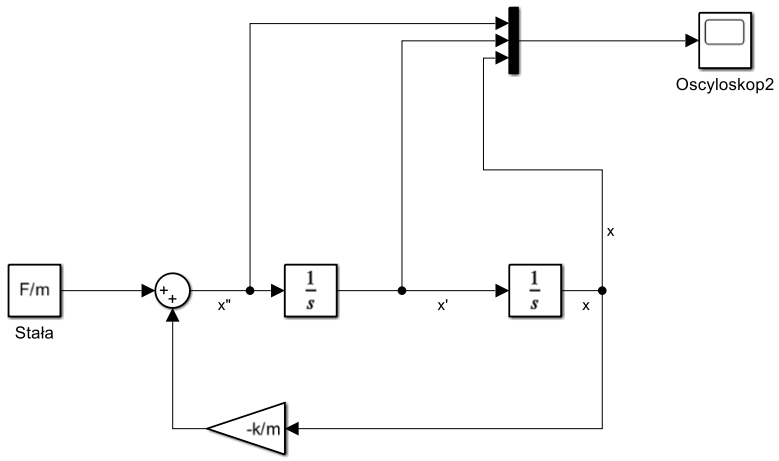


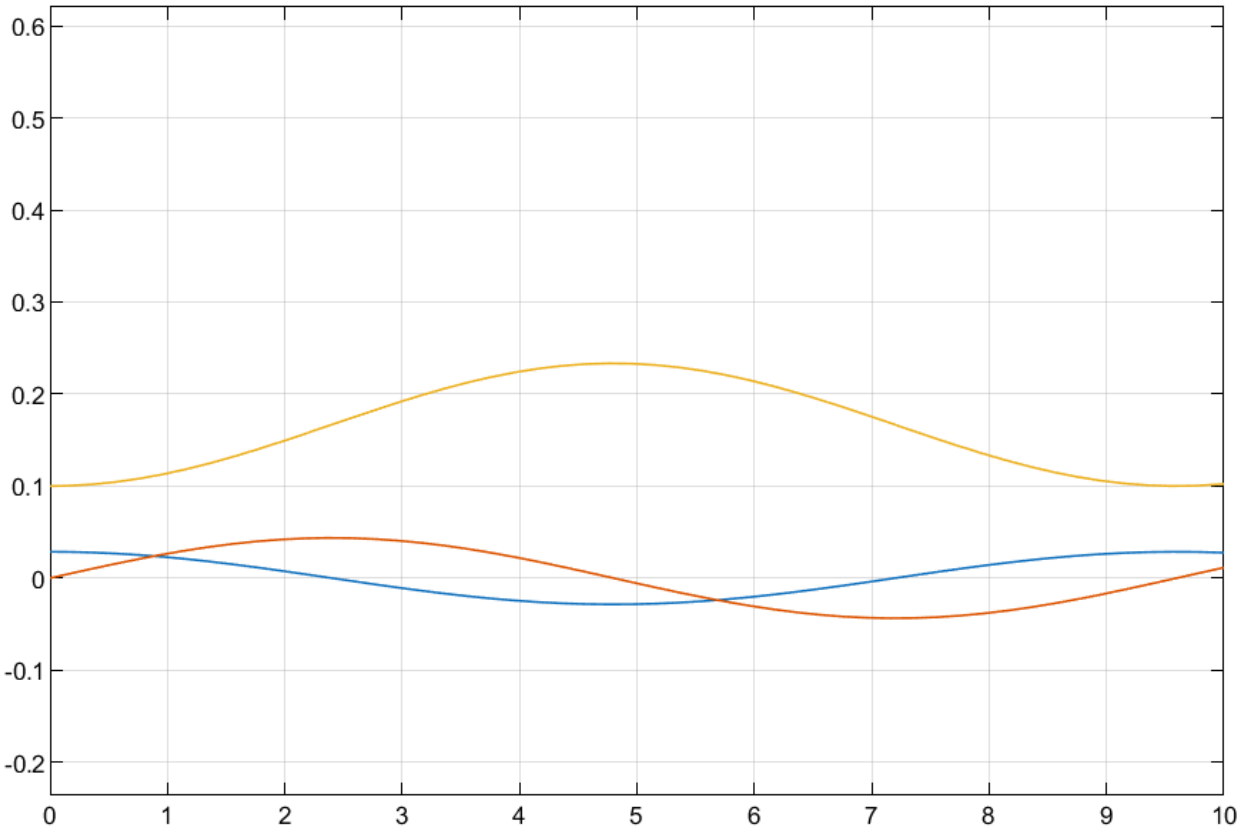
Żółte – położenie, niebieski – przyspieszenie, pomarańczowy – prędkość

Oczywiście aby układ zadziałał konieczne są warunki początkowe. Gdy domyślnie są one równe 0 to wykresy będą o wartości zerowej. Stąd musimy uwzględnić początkowe położenie masy względem sprężyny. Jak już wcześniej pisałem można to zrobić za pośrednictwem integratora gdzie odpowiednią wartość w naszym przypadku 0.1 wpisuję w pole initial condition.

b) Wersja z wymuszeniem:

W tym wypadku nie potrzebne jest już wymuszenie gdyż bloczek const pełni jego funkcję. Tam zapodaję wartość siły oraz masy a reszta jest już odbywana niejako w tle. W porównaniu do schematu z przykładu a) dodałem sumator który łączy nam sygnał pochodzący od stałej z tym od wzmocnienia.





Żółte – położenie, niebieski – przyspieszenie, pomarańczowy – prędkość

Sposób drugi:

W simulinku można również zaprojektować ten sam układ drogą macierzową. Wymaga ona użycia jedynie bloku macierzowego z wartością początkową. Sposób ten jest podobny do programowania w

matlabie jednak po wpisaniu macierzy proces odbywa się w tle. Skorzystałem z bloczka State-Space gdzie wpisałem następujące wartości dla macierzy:

A = [0 1 ; -k/m 0]

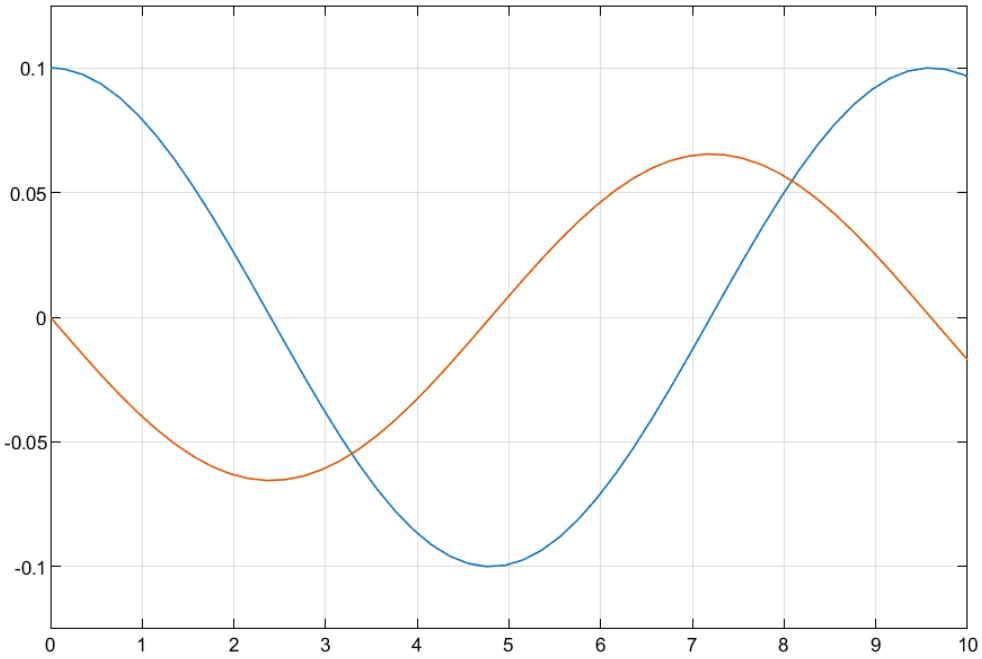
B = [0 0]

C = [1 0; 0 1]

D = [0 0]

Initial conditions(dla schematu pierwszego) = [0.1 0]

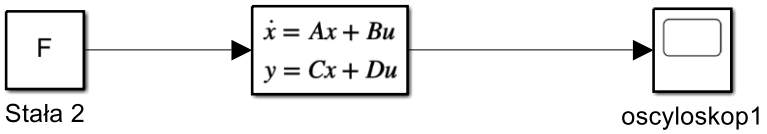
a) Wersja swobodna:

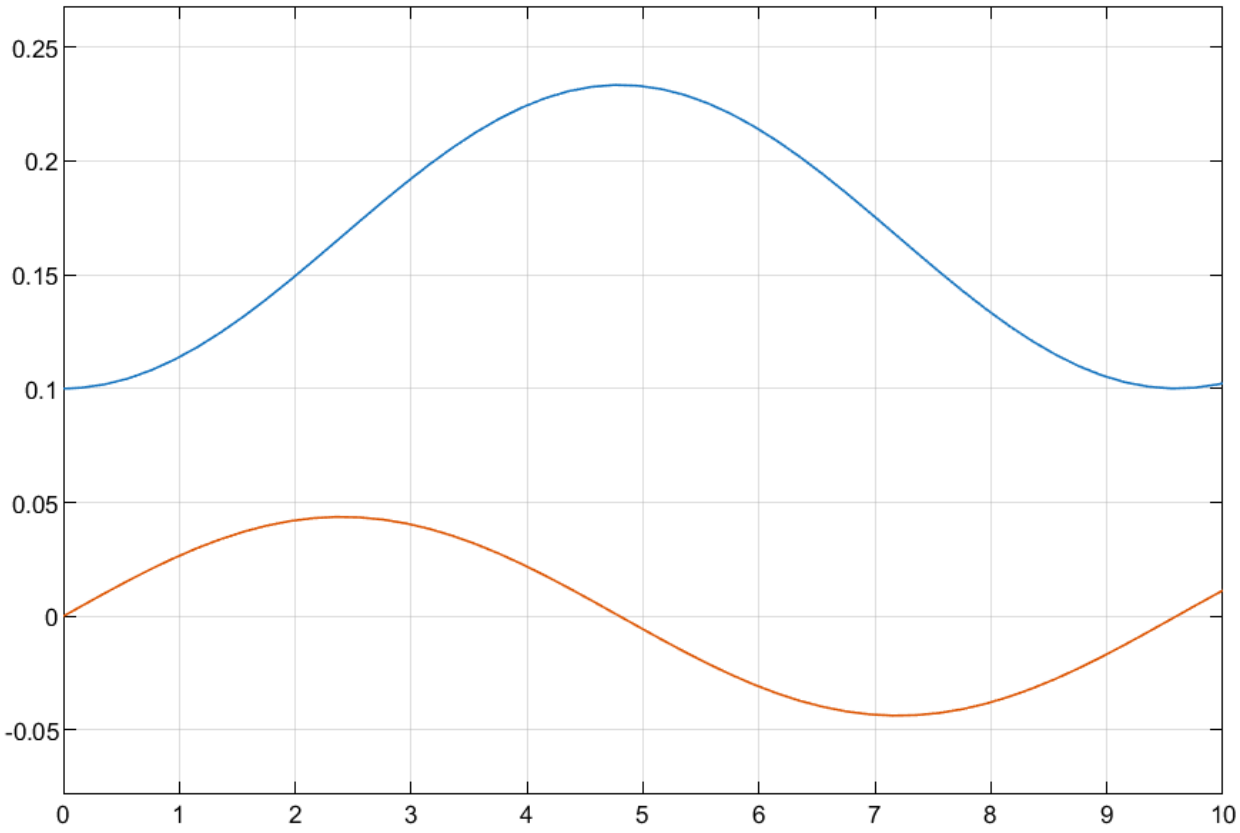


Pomarańczowy – prędkość, niebieski – przemieszczenie

b) Wersja z wymuszeniem:

W tej wersji dodatkowo dodałem początkowe wymuszenie w postaci stałej siły.





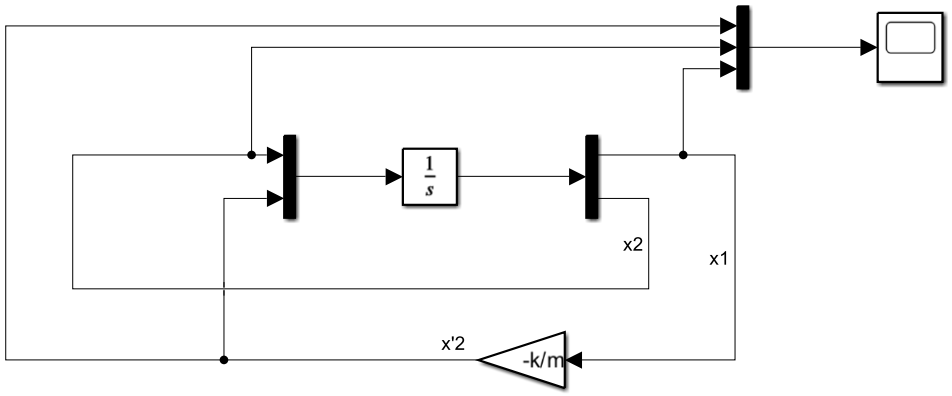
Pomarańczowy – prędkość, niebieski – przemieszczenie

Wynik jest taki sam jak w przypadku wersji pierwszej.

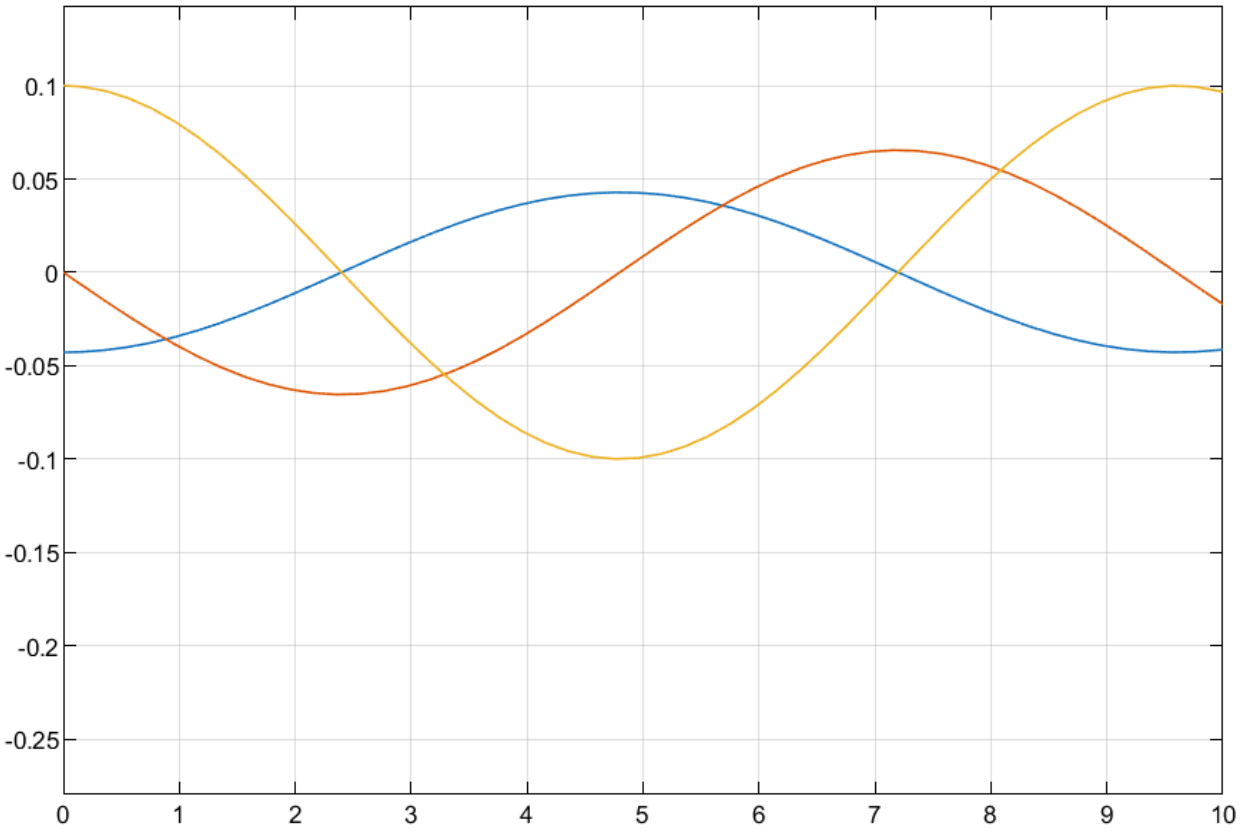
Sposób trzeci:

a) Wersja swobodna:

Do stworzenia tego modelu użyłem już wcześniej używanych bloków, w tym jednego integratora, które pozwolił mi przeliczyć przyspieszenie na prędkość i położenie, niezbędne do dalszych obliczeń. Współczynniki pojawiające się w równaniu różniczkowym uwzględniłem, wzmacniając odpowiednie sygnały za pomocą bloków o właściwej sile. Wszystkie zmienne zapisałem symbolicznie w skrypcie MATLAB, a potem nadałem im konkretne wartości liczbowe, zgodnie z wcześniejszymi założeniami.

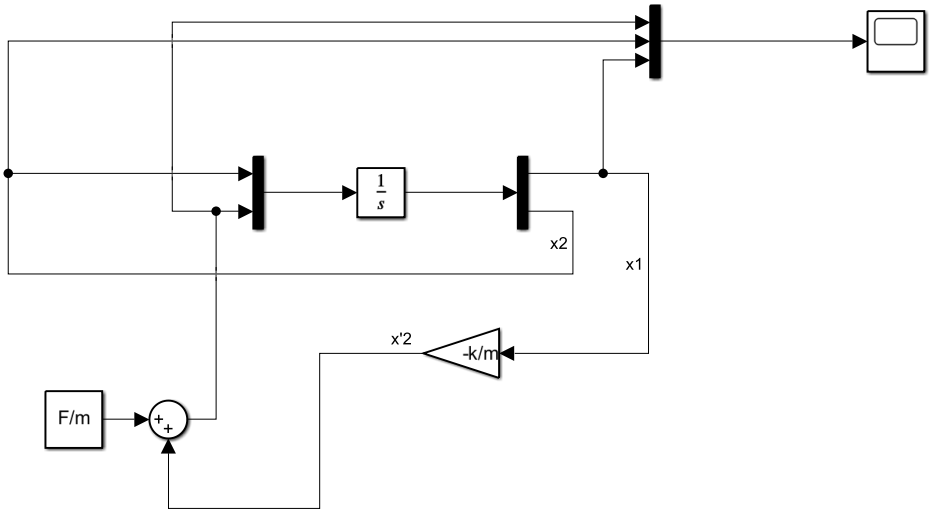


Oczywiście w tym przypadku należało odpowiednio określić wartości początkowe w integratorze.

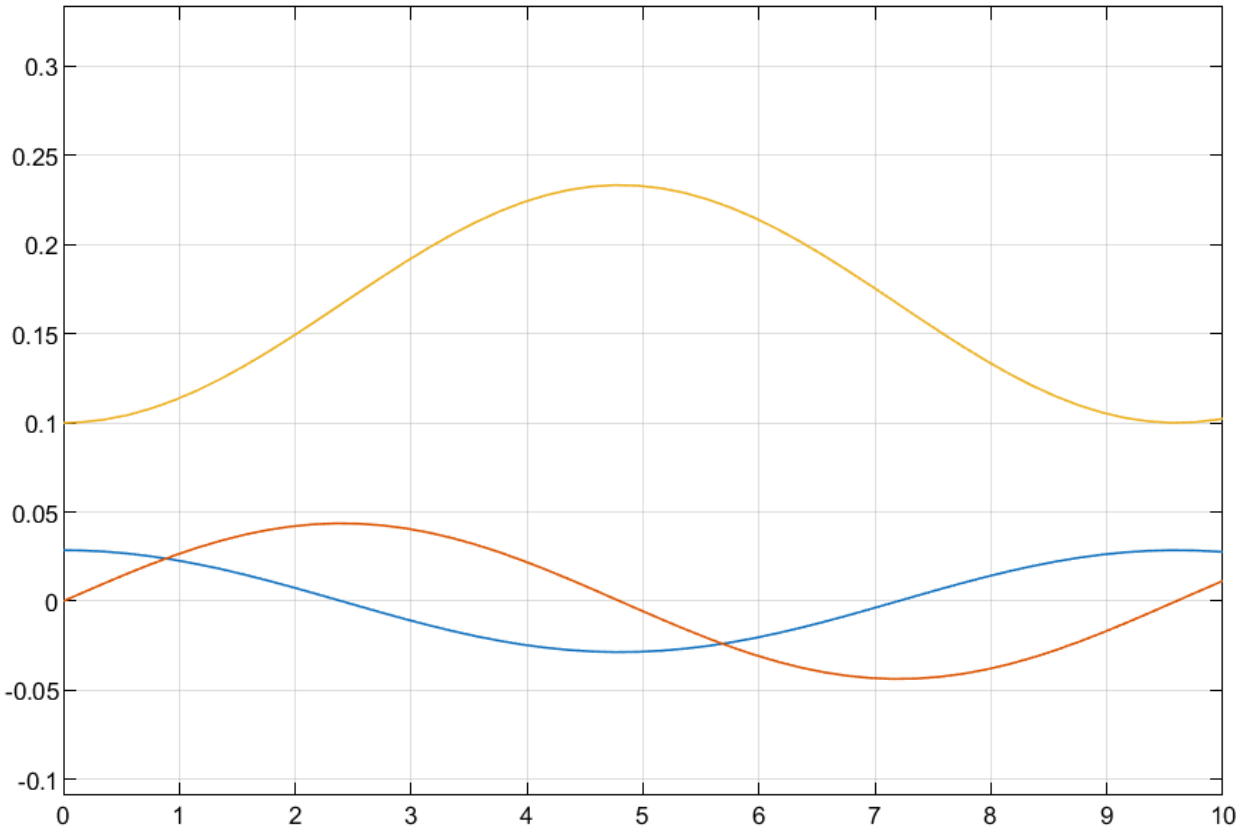


Żółte – położenie, niebieski – przyspieszenie, pomarańczowy – prędkość

b) Wersja z wymuszeniem:



Ten przykład różni się jak zwykle w stosunku do wcześniejszego zapodaniem wymuszenia w postaci bloczka constant. Połączyłem go z wyjściem gaina, na którym zamieściłem x'2 za pomocą sumatora.



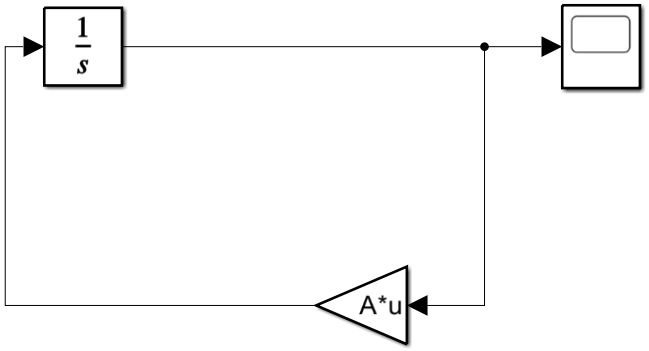
Żółte – położenie, niebieski – przyspieszenie, pomarańczowy – prędkość

Wyniki są oczywiście takie same jak w przypadku dwóch poprzednich metod.

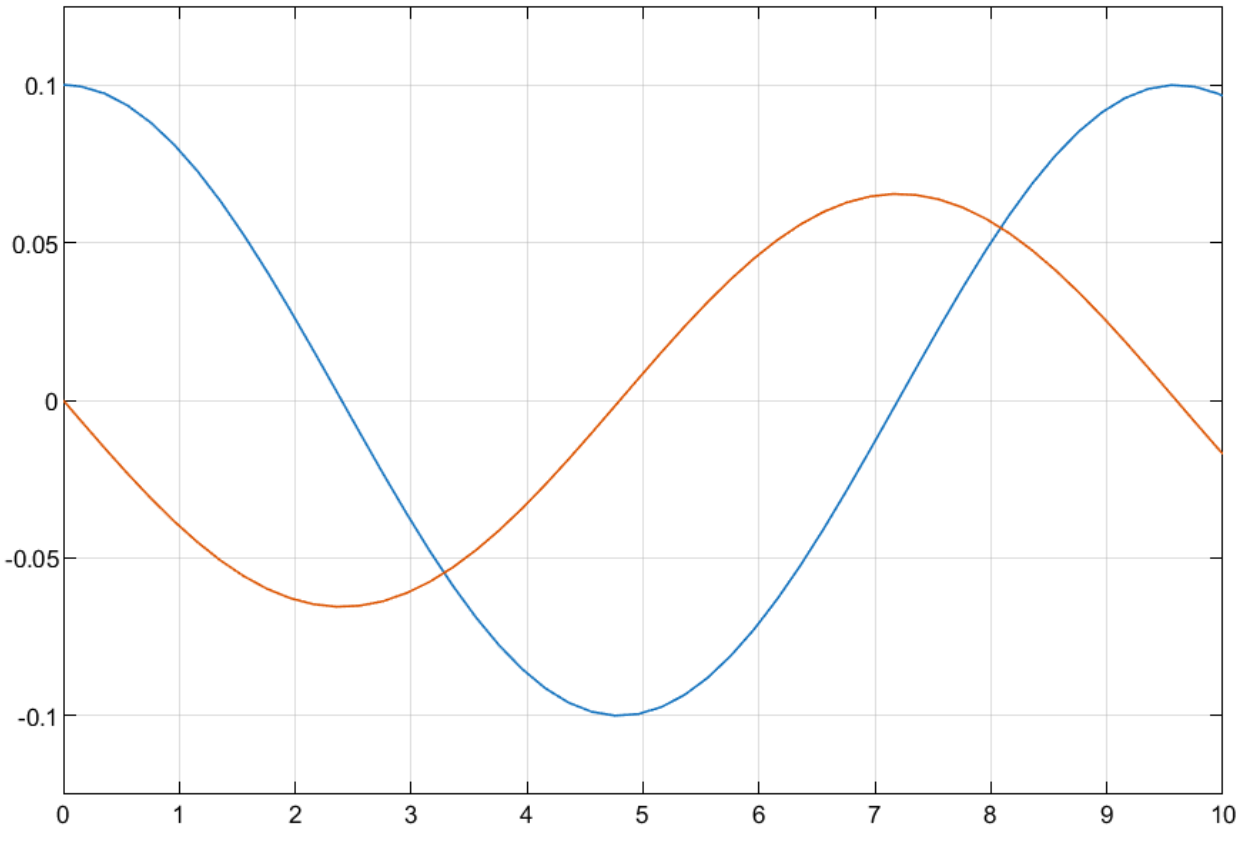
Sposób czwarty:

a) Wersja swobodna:

W tym zadaniu pokazuję, że układ można zapisać jeszcze prościej niż w przypadku dwóch poprzednich metod, a mianowicie za pomocą wzmocnienia macierzowego. W bloczku gain jesteśmy wtedy zmuszeni na zmianę mnożenia na mnożenie macierzowe. A jako Gain podajemu macierz A.

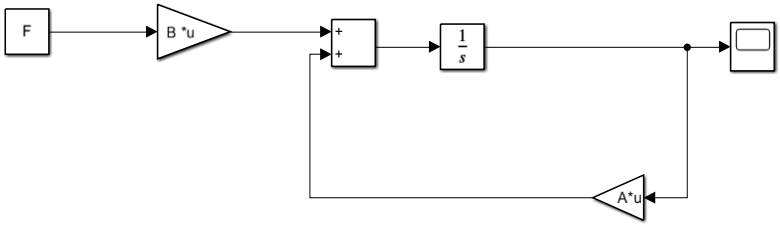


Po uruchomieniu symulacji uzyskamy taki wynik:

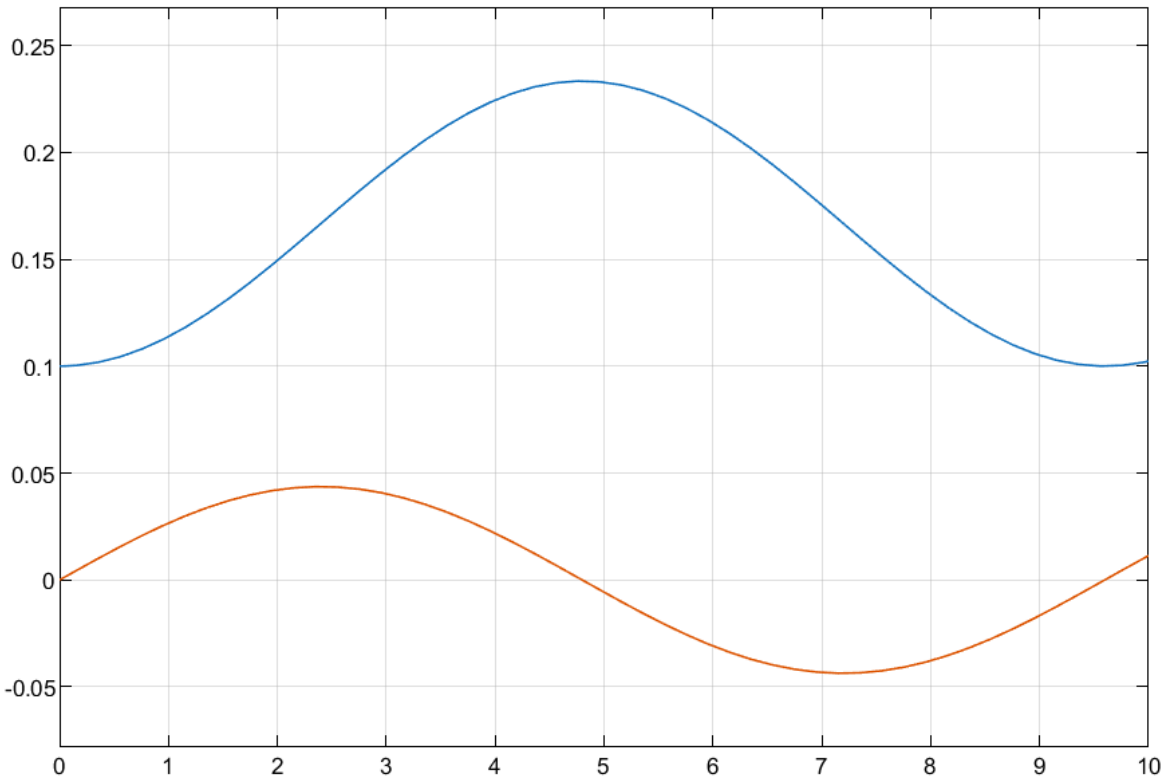


Niebieski – położenie, pomarańczowy – prędkość

b) Wersja z wymuszeniem:
Dodajemy wymuszenie poprzez zastosowanie sumatora



Tutaj również dla drugiego członu musimy wybrać wzmocnienie macierzowe:



Niebieski – położenie, pomarańczowy – prędkość

Podsumowanie III Rozdziału

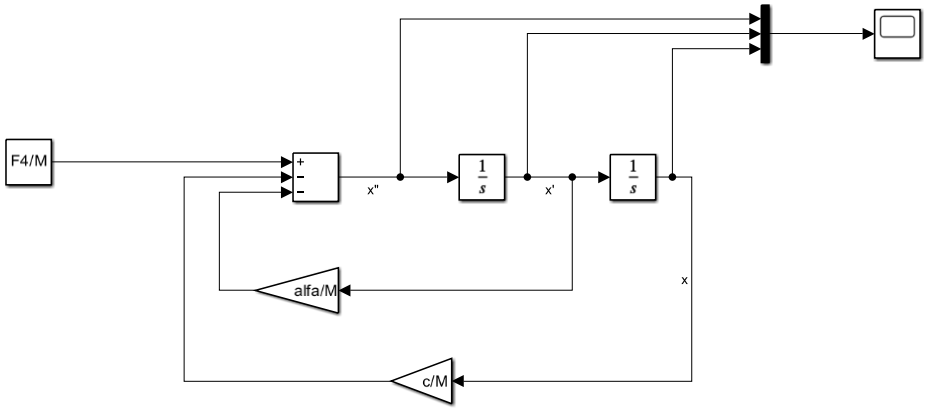
Wszystkie metody modelowania zaprezentowane w rozdziale 3 dają ten sam wynik dlatego można je stosować zamienie. Najbardziej czytelna dla rozpatrywanego układu i najprostsza w wykonaniu jest dla mnie metoda 1 jednak przy bardziej złożonych modelach przydatne mogą się okazać również kolejne sposoby. Również moją sympatię zaskarbił sposób nr 2. Jest on prosty w zrozumieniu i najbardziej matematyczny. Pomaga również we w miarę szybkiej lokalizacji błędów.

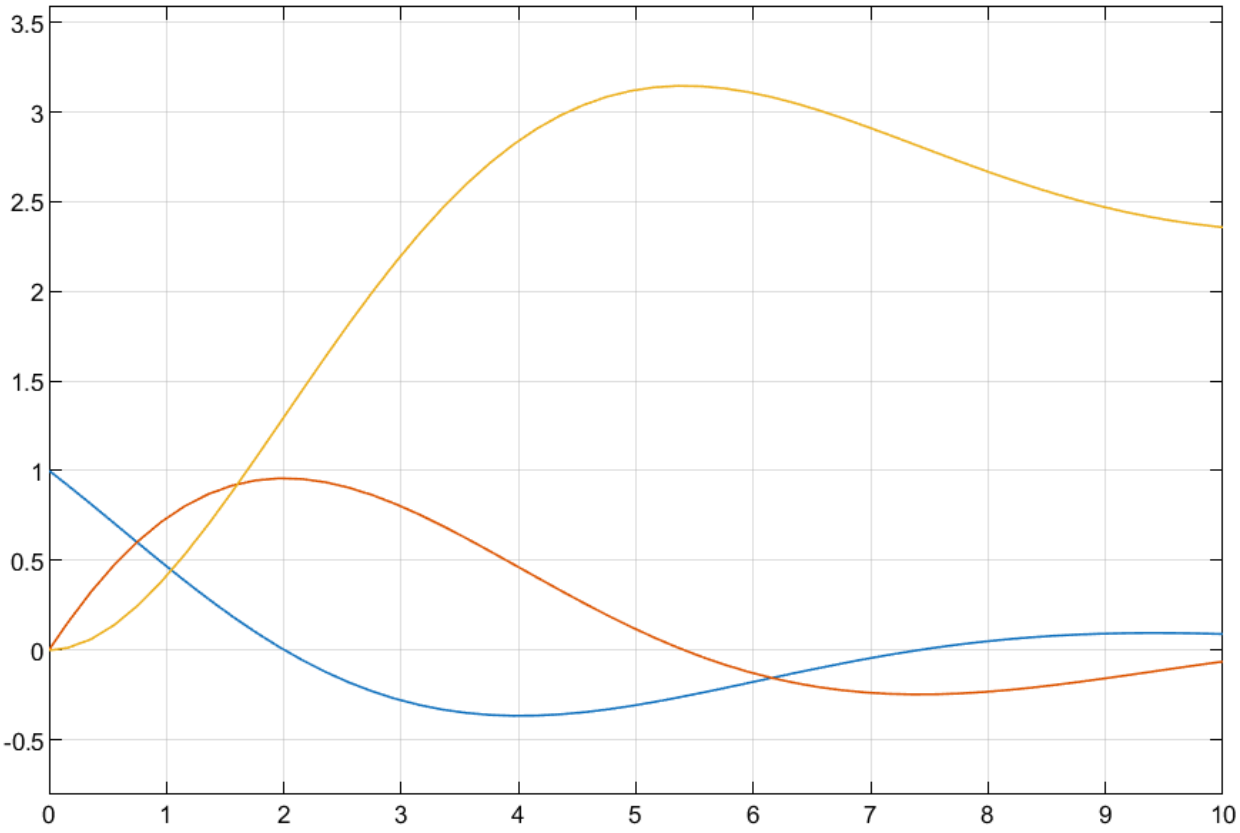
Model zawieszenia samochodu:

Na zakończenie przyszło mi zamodelować układ z 4 rozdziału konspektu 2. Początkowo układ doprowadziłem do postaci:

x” = F/M – alfa/M* x’ – (c/m) * x,

a następnie za pomocą integratorów całkowałem wyższe pochodne do niższych i całość połączyłem z bloczkiem const reprezentowanym przez F/M sumatorem (parametry + - -). Następnie każde z wartości x” , x’ oraz x podpiąłem do wejścia multipleksera. Jego wyjście podpiąłem do oscylatora dzięki czemu mogłem obserwować trzy przebiegi na jednym wykresie.





Żółte – położenie, niebieski – przyspieszenie, pomarańczowy – prędkość