Wprowadzenie do komputerowo wspomaganego projektowania systemów sterowania – analiza liniowa

(wersja robocza 24.03.2017)

Załączniki A: Funkcje CACSD w pakiecie Matlab
Załączniki B: Przykładowe programy kursów

Tylko transmitancje i równania stanu:
- najpierw funkcje tf i ss pod Matlabem i analiza
- potem bloki tf i ss pod Simulinkiem.
Najpierw analiza – toolsy Linear Analysis. Na podstawowych członach (na koniec tylko proste obiekty)
- wspomnieć o możliwości przybliżenia opóźnienia i przesuwnika fazowego (bez szczegółów?)
- pokazać przetworzenie obiektu do podstawowych członów.

Regulacja na najprostszych obiektach - analiza
Projektowanie na obiekatch LTI

Połączenie szeregowe, równoległe i przeciwsobne – Oppelt/383
Zmienna procesowa (wielkość regulowana x), Zmienna sterująca (wielkość nastawiana y)
Wartość zadana (wielkość przewodnia w)
Transmitancja (przepustowość – Oppelt/74)
Charakterystyka układu zamkniętego (przewodnia, czyli od SP – Oppelt/306)
Charakterystyka zakłóceniowa (Oppelt/306)

Literatura:
Nyquist 1932 (p.Oppelt/257)

Wprowadzenie

Podręcznik stanowi pierwszą część serii praktycznego wprowadzenia do projektowania układów sterowania z wykorzystaniem pakietu Matlab. Przedstawiono **metody definiowania i analizowania** własności na przykładzie prostych, ogólnych transmitancji i równań stanu, tak by można je było łatwo wykonać i symulacyjnie, i analitycznie. Dogłębne zrozumienie tych prostych przykładów ułatwi interpretację wyników dla rzeczywistych obiektów. Teoretyczne opracowanie omawianych metod zawarte jest w podanej literaturze (w szczególności [1, 2]), natomiast niniejsze opracowanie kładzie nacisk na zastosowanie metod i interpretację wyników – **mało teorii, dużo praktyki**.

Założono, że Czytelnik ma podstawową znajomość pakietu Matlab w zakresie definicji i wykorzystania macierzy, rysowania wykresów, pisania skryptów oraz rysowania schematów w środowisku Simulink. Praktyczny opis tych zagadnień można znaleźć na przykład w skrypcie pt. „Scilab i Matlab - podstawowe zastosowania inżynierskie" [3] (skrypt dostępny w Dolnośląskiej Bibliotece Cyfrowej). Natomiast szczegółowy opis poszczególnych funkcji czy używanych narzędzi jest łatwo dostępny w podręcznej pomocy Matlaba. Skrypty i schematy zawarte w podręczniku prezentują jedynie przykładowe sposoby rozwiązania zadań, oparte na tradycyjnych funkcjach i narzędziach, które są dostępne nawet dla użytkowników starszych wersji Matlaba (np. R2010)[1]. Część funkcji/narzędzi wymaga zainstalowania dodatkowych przyborników Matlaba (Matlab Toolbox) – w szczególności przybornika Control Toolbox (patrz Załącznik??). Fragmenty skryptów oraz operacje wykonywane w Matlabie są prezentowane w szarych ramkach, z zachowaniem domyślnych czcionek i kolorów stosowanych przez edytor tekstowy Matlaba. Wiele operacji Matlaba jest dostępnych poprzez menu kontekstowe różnych elementów graficznych wywoływane przez prawy lub lewy przycisk myszy (nazywane w skrócie prawe lub lewe menu).

Oprócz umiejętności przeprowadzenia odpowiednich badań doświadczalnych (w tym symulacyjnych), ważną umiejętnością jest również opracowanie wyników badań. Sprawozdanie z przeprowadzonych badań jest formą dokumentacji i jako takie powinno zawierać:

a) informacje, które pozwolą powtórzyć badania – wartości parametrów, punkty pracy, wykorzystywany model (jeśli to transmitancja to jakie wejścia i wyjścia, jaka postać), strukturę i nastawy regulatora (sterownika)
b) rysunki, wykresy i tabele z wynikami – ponumerowane i podpisane, opracowane (czyli zazwyczaj przetworzone, pogrupowane, ...)
c) wnioski z przeprowadzonych badań zilustrowane odpowiednimi wykresami czy wskaźnikami – we wnioskach należy odwołać się do wykresów/tabel, które je ilustrują.

Numeracja przykładowych skryptów?

[1] Nowsze opcje funkcji, sygnalizowane w przykładach, zostały przygotowane i sprawdzone w wersji R2013. Zazwyczaj te nowe opcje są jedynie innym (prostszym) sposobem realizacji tej samej operacji.

Niniejsze opracowanie zawiera wiele odnośników do dwóch głównych źródeł w języku angielskim – dokumentacji Matlaba oraz podręczników [1, 2]. Poniżej przedstawiono zestawienie oznaczeń stosowanych w podręczniku i w głównych źródłach.

Podręcznik	Matlab	[1, 2] Åström
P – proces technologiczny (realny obiekt) ???		
G – obiekt, proces (fizyczny obiekt lub dokładny model)	G (żółty)	P – Plant
G_m – model obiektu/procesu, który jest częścią regulatora		
C – regulator, kontroler (zmienić R na C, ze względu na oznaczenia w Matlab???)	C (czerwony)	C – Controller Feedback Compensator
F – filtr, FF	F (zielony)	F – Filtr Feedforward Compensator
	H	
t – czas		
u – sterowanie (zmienna wejściowa obiektu, wykorzystana do sterowania)	u – control	u – control
x – zmienna stanu		
y – wyjście	y – output	y – output
y^* – wartość zadana	r – reference	y_{sp} – set point
z – zakłócenia (zmienna wejściowa obiektu, która wpływa na jego zachowanie)	d – distrubances	d – distrubances
G_o – transmitancja układu otwartego,		
G_z – transmitancja układu zamkniętego		

$z(t)$, $u(t)$, $x(t)$, $y(t)$, $y^*(t)$ – przebiegi czasowe sygnałów (zmiennych)
$z(s)$, $u(s)$, $x(s)$, $y(s)$, $y^*(s)$ – transformaty przebiegów czasowych

$\frac{d^k x(t)}{dt^k} \leftrightarrow x^{(k)}(t)$ - skrótowe oznaczenie operacji różniczkowania

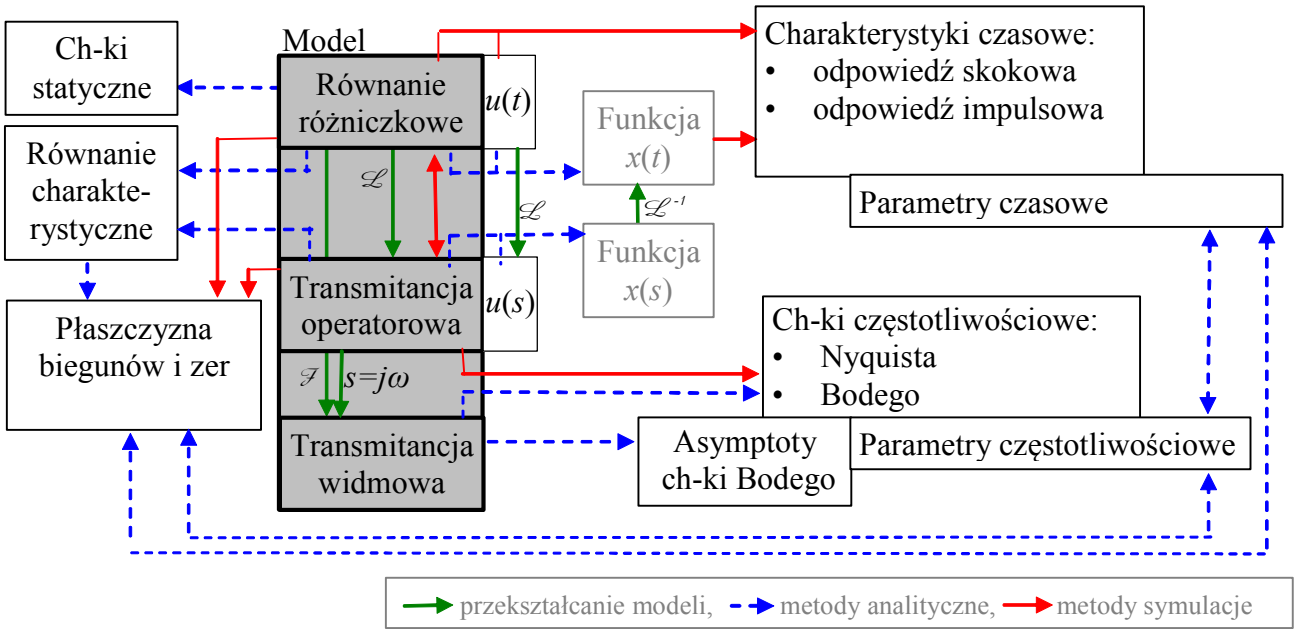
Funcje Matlab: `ss()` Courier New 10
Bloki Simulink: Integrator Arial 10 (zamiast Helvetica 10)

1. Podstawowe formy opisu dynamiki układów (wprowadzenie)

1.1. Układy liniowe

1.1.1 Modele i charakterystyki układów

Układem będziemy nazywać zarówno obiekt (układ fizyczny, proces technologiczny), jak i system sterowania (układ automatyki), który ma na celu doprowadzenie i utrzymywanie obiektu w określonym stanie, niezależnie od występujących zakłóceń. Podstawowe badania polegają na opisaniu reakcji układu na różne wymuszenia (funkcje wejściowe). Jest to zadanie wymagające stosowania opisu układu z uwzględnieniem własności dynamicznych. Opis dynamiki układu może mieć formę analityczną (równania różniczkowe, transmitancje) lub graficzną (charakterystyki). Szczególne własności układów liniowych (A.1) zapewniają jednoznaczne, ilościowe relacje pomiędzy różnymi opisami (rys. 1.1), które są wykorzystywane podczas analizy i projektowania obiektów i systemów sterowania.



Rys. 1.1. Formy modeli i metody analizy dynamiki układów liniowych

Analityczne formy opisu układów liniowych, czyli modele, to przede wszystkim **liniowe równanie różniczkowe**:

$$a_n x^{(n)}(t) + \ldots + a_1 \dot{x}(t) + a_0 x(t) = b_m u^m(t) + \ldots + b_1 \dot{u}(t) + b_0 u(t) \tag{1-1}$$

i **transmitancja operatorowa** $G(s)$:

$$\frac{x(s)}{u(s)} = \frac{b_m s^m + \ldots + b_1 s + b_0}{a_n s^n + \ldots + a_1 s + a_0} = G(s) \tag{1-2}$$

Równanie różniczkowe jest najbardziej pierwotną i ogólną formą opisu dynamiki, ponieważ jest konstruowane na podstawie elementarnych praw zachowania (np. energii, masy, ładunków) obowiązujących w każdej chwili czasu. Rozwiązanie równania różniczkowego oznacza wyznaczenie wzoru $x(t)$, opisującego reakcję układu za zadane wymuszenie $u(t)$ i przy założonych warunkach początkowych [dodatek?].

Natomiast transmitancja $G(s)$ wynika z równania różniczkowego, poddanego przekształceniu Laplace'a $\mathscr{L}$ (A.2). Jest to przekształcenie operatorowe, które w praktyce inżynierskiej polega na zastąpieniu operacji różniczkowania przez operację potęgowania, czyli zastosowanie następującej równoważności:

$$\frac{d^k}{dt^k} \leftrightarrow s^k \tag{1-3}$$

Jednocześnie funkcje zmienne w czasie ($u(t)$, $x(t)$) są zastępowane przez transformaty funkcji, czyli funkcje zmiennej s. Wprowadzenie transmitancji wymaga założenia o zerowych warunkach początkowych (w chwili t=0 wartości funkcji $u(t)$ i $x(t)$ oraz ich pochodnych są równe 0). Na podstawie transmitancji operatorowej $G(s)$ i transformaty $\mathscr{L}$ funkcji wejściowej $u(s)$ w prosty sposób można wyznaczyć transformatę funkcji na wyjściu układu:

$$x(s) = G(s)u(s) \tag{1-4}$$

Stosowana jest też **transmitancja widmowa** $G(j\omega)$, opisująca przenoszenie przez układ sygnałów sinusoidalnych o różnych częstotliwościach. Postać transmitancji $G(j\omega)$ wynika przekształcenia równania różniczkowego za pomocą przekształcenia Fouriera $\mathcal{F}$, jednak zazwyczaj wykorzystuje się zależność pomiędzy przekształceniem Laplace'a i Fouriera:

$$s \leftrightarrow j\omega \tag{1-5}$$

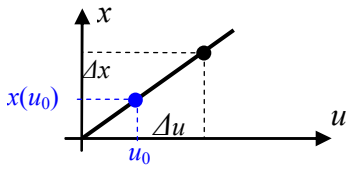
W przeciwieństwie do zmiennej s, zmienna ω ma interpretację fizyczną i oznacza pulsację sygnału sinusoidalnego podawanego na wejście układu, natomiast transmitancja $G(j\omega)$ opisuje sposób przekształcenia tego sygnału przez układ.

Na podstawie tych trzech form modeli liniowych można przeprowadzić analityczne badania własności układów, w szczególności wyznaczanie różnego typu charakterystyk:

1. **Charakterystyka statyczna** $x(u)$ powstaje na podstawie równania statycznego, które otrzymujemy po wyzerowaniu pochodnych w równaniu różniczkowym (1-1):

$$a_0 x(t) = b_0 u(t) \quad \leftrightarrow \quad a_0 x = b_0 u \tag{1-6}$$

Może też być wyznaczona doświadczalnie na podstawie pomiaru zmiennych u i x, w czasie gdy na obiekcie nie zachodzą żadne zmiany. Charakterystyka statyczna (równanie statyczne) pozwala obliczać **punkt równowagi** układu x_0 przy stałym wymuszeniu u_0:



Rys. 1.2. Charakterystyka statyczna i punkt równowagi

$$x_0 = \frac{b_0}{a_0} u_0 \tag{1-7}$$

oraz **wzmocnienie układu** przy stałym wymuszeniu:

$$k_{ukł} = \frac{\Delta x}{\Delta u} = \frac{b_0}{a_0} \tag{1-8}$$

Jeśli analizowany model ma postać transmitancji, to punkt równowagi można wyznaczyć na podstawie twierdzenia o wartości końcowej:

$$\lim_{t\to\infty} x(t) = \lim_{s\to 0} s \frac{b_m s^m + \ldots + b_1 s + b_0}{a_n s^n + \ldots + a_1 s + a_0} \cdot \frac{u_0}{s} = \frac{b_0}{a_0} u_0 \tag{1-9}$$

2. **Charakterystyki czasowe** przedstawiają rozwiązanie $x(t)$ równania różniczkowego przy zadanej funkcji wejściowej $u(t)$ i dla zadanych warunków początkowych. Najczęściej jest to reakcja na wymuszenie skokowe lub impulsowe, które jest podawane na układ, znajdujący się w stanie równowagi - **odpowiedź skokowa** to reakcja na skok jednostkowy $1(t)$, **odpowiedź impulsowa** to reakcja na impuls Diraca $\delta(t)$:

$$1(t) = \begin{cases} 0 \text{ dla } t < 0 \\ 1 \text{ dla } t \geq 0 \end{cases}; \qquad \delta(t) = \begin{cases} 0 \text{ dla } t \neq 0 \\ +\infty \text{ dla } t = 0 \end{cases} \text{ i } \int_{-\infty}^{\infty} \delta(t)dt = 1 \text{ (czyli „pole"=1).} \tag{1-10}$$

Rozwiązanie $x(t)$ liniowego równania różniczkowego można wyznaczyć analityczne [4] i na tej podstawie narysować charakterystykę czasową. Jednak znajomość wzoru $x(t)$ zazwyczaj nie jest ani konieczna, ani wygodna. Często też charakterystyka czasowa jest uzyskiwana w sposób doświadczalny, przez pomiary na obiekcie. W praktyce inżynierskiej najczęściej wystarczy jedynie znać i stosować parametry odpowiedzi skokowych (lub impulsowych), które określają charakterystyczne cechy reakcji, takie jak stabilność (czy układ osiąga stan równowagi), czas dochodzenia do stanu równowagi, itd. (p. 2.2.2),
3. **Charakterystyki częstotliwościowe** opisują przenoszenie przez układ sygnałów sinusoidalnych o różnych częstotliwościach. Charakterystyki częstotliwościowe można wyznaczyć na podstawie transmitancji $G(j\omega)$ lub doświadczalnie (p. 2.2.4).
4. **Płaszczyzna pierwiastków** jest płaszczyzną zespoloną, która służy przedstawienia **biegunów** transmitancji (pierwiastków mianownika) i **zer** transmitancji (pierwiastków licznika).

W programach symulacyjnych takich jak Matlab, dostępne są specjalistyczne funkcje do definiowania zarówno równań różniczkowych, jak i transmitancji, a także metody konwersji jednych w drugie. Wszystkie układy liniowe, stacjonarne (LTI, ang. Linear Time-Invariant System) zdefiniowane w Matlabie za pomocą takich funkcji są dostępne jako zmienne typu **object LTI**.

Niezależnie od formy definicji modelu jaka zostanie użyta do zdefiniowania modelu w Matlabie, można wygenerować każdy typ charakterystyki czasowej czy częstotliwościowej (program automatycznie wykonuje konieczne konwersje modelu).

1.1.2 Podstawowe obiekty (człony) dynamiki

W praktyce inżynierskiej wśród podstawowych form opisu dynamiki układów często stosuje się podstawowe obiekty (człony) dynamiki, czyli najprostsze przypadki transmitancji, o sformalizowanej postaci i parametrach.

Wprowadzenie formy i parametrów - charakterystyki

Człon inercyjny

Człon oscylacyjny

$$G(s) = \frac{\omega_n^2}{s^2 + 2\xi\,\omega_n\, s + \omega_n^2}, \tag{1-11}$$

Jeśli $\xi^2<1$, to człon ma parę biegunów zespolonych:

$$p_{1,2} = \alpha \pm j\omega = -\sigma \pm j\omega_r \tag{1-12}$$

gdzie: $\alpha = -\xi\omega_n, \sigma = \xi\omega_n,\ \omega_r = \omega_n\sqrt{1-\xi^2}$. Parametr α oznacza część rzeczywistą biegunów i jest stosowany przy opisywaniu stabilności (układ jest stabilny jeśli wszystkie bieguny układu mają ujemne części rzeczywiste). Parametr σ jest stosowany głównie w przypadku układów stabilnych do opisania odległości biegunów od osi Im (jest wówczas wygodniejszy niż ujemna α).

-

- układ unilateralny (typu kaskada niewspółdziałająca)
- układ nieunilateralny (typu kaskada współdziałająca)

Może wyróżnić pierwsze użycie terminów, które: 1) powinny być znane (te powyżej, dostępne w załączniku), 2) są zdefiniowane tu

1.1.3 Relacje pomiędzy charakterystykami

Cel sterowania - w szczególności stabilność i czas reakcji

W kolejnym rozdziale będzie analiza

1.2. Układy nieliniowe i zlinearyzowanie

Transmitancja w punkcie pracy – w dalszych opisach układ zawsze staruje od 0.

Przesunięcie układu współrzędnych

Skąd wziąć model obiektu?

Układanie równań bilansowych - różniczkowych i operatorowych – przykłady w rozdziałach

Konwersja formy

Najprostsze modele zastępcze – Küpfmüllera, Strejca, przybliżenie Pade

1.3. Opis układu regulacji

tu????

2. Analiza liniowa z wykorzystaniem funkcji Matlab-Control

2.1. Obiekty LTI - definicja i parametry

2.1.1 Definiowanie modelu SISO[1]

Najprostsze modele dynamiki, opisujące liniowe układy z jednym wejściem i jednym wyjściem mają zazwyczaj postać równania różniczkowego lub transmitancji. Modele tego typu opisano poniżej na przykładowych obiektach inercyjnych:

1) rzędu pierwszego

$$a_1\dot{x}(t) + a_0 x(t) = b_0 u(t)$$

$$G_1(s) = \frac{b_0}{a_1 s + a_0}$$

2) rzędu drugiego

$$a_2\ddot{x}(t) + a_1\dot{x}(t) + a_0 x(t) = b_1\dot{u}(t) + b_0 u(t) \qquad (2\text{-}1)$$

$$G_2(s) = \frac{b_1 s + b_0}{a_2 s^2 + a_1 s + a_0} \qquad (2\text{-}2)$$

Są to typowe transmitancje w postaci funkcji wymiernych, gdzie stopień wielomianu w liczniku jest mniejszy niż stopień wielomianu w mianowniku. Wyznaczając **bieguny** p i **zera** z układu zapisać transmitancje (2-2) w postaci iloczynowej:

$$G_1(s) = \frac{k_1}{s - p_1} \qquad G_2(s) = \frac{k_1(s - z_1)}{(s - p_1)(s - p_2)} \qquad (2\text{-}3)$$

Jeśli bieguny i zera mają wartości rzeczywiste dodatnie, to stosuje się zapis transmitancji (2-3) za pomocą iloczynu podstawowych obiektów (członów) dynamiki :

1) człon inercyjny 1 rzędu

$$G_1(s) = \frac{k}{T_1 s + 1}$$

2) człon inercyjny 2 rzędu i człon forsujący

$$G_2(s) = \frac{k}{(T_1 s + 1)(T_2 s + 1)} \cdot (T_{z1} s + 1) \qquad (2\text{-}4)$$

Pomiędzy parametrami poszczególnych form transmitancji (2-2)÷(2-4) zachodzą oczywiście jednoznaczne związki (B.1), co pozwala przetwarzać transmitancje na dowolną formę.

W automatyce często stosuje się występuje definiowanie modelu za pomocą stałych czasowych (T_1, T_2, T_{z1}) i wzmocnienia (k). Należy zwrócić uwagę, że w różnych formach transmitancji różne parametry są nazywane wzmocnieniem - k_1 (2-2), k (2-3). Parametr definiowany jako **wzmocnienie obiektu (układu)** $k_{ukł}$ nie zależy od formy modelu i oznacza wzmocnienie sygnału wejściowego przy stałym wymuszeniu (1-8).

Przykład: Obiekty inercyjne. W przykładach prezentowanych w tej części podręcznika, będą wykorzystywane dwa obiekty opisane za pomocą następujących parametrów:

- **ObiektG1**: układ ma wzmocnienie 2 i stałą czasową równą 2,
- **ObiektG2**: układ ma wzmocnienie 2, stałe czasowe równe 2 i 4, oraz jedno zero o wartości -2.

Transmitancje o powyższych własnościach można zdefiniować w dowolnej formie (2-2)÷(2-4) przeliczając odpowiednio zadane parametry (tab. 2-1).

Tab. 2-1. Parametry przykładowych układów ObiektG1 i ObiektG2 dla różnych form transmitancji

	ObiektG1	ObiektG2
Dane układu:	$k_{ukł}$=2, T_1 =2	$k_{ukł}$=2, T_1 =2, T_2 =4, z_1=-2
Odtworzenie transmitancji na podstawie danych:	$\lim_{s\to 0} \frac{k}{2s+1} = 2 \rightarrow k = 2$	$\lim_{s\to 0} \frac{a(s+2)}{(2s+1)(4s+1)} = 2 \rightarrow a = 1$
a) współczynniki do (2-2)	a_1=2, a_0=1, b_0=2	a_2=8, a_1=6, a_0=1, b_1=1, b_0=2
b) bieguny i zera do (2-3)	k_1=1, p_1= -0.5	k_1= 0.125, $\boldsymbol{z_{z1}}$**= -2**, p_1= -0.5, p_2= -0.25
c) człony dynamiki do (2-4)	$\boldsymbol{k}$**= 2,** $\boldsymbol{T_1}$**= 2**	$\boldsymbol{k}$**= 2,** T_{z1}= 0.5, $\boldsymbol{T_1}$**= 2,** $\boldsymbol{T_2}$**= 4**

Poniższe fragmenty skryptów przedstawiają alternatywne **warianty definicji transmitancji** za pomocą funkcji `tf()` i `zpk()`, wraz z przygotowaniem odpowiedniego zestawu parametrów na podstawie wartości k, T_1, T_2, T_{z1}:

	ObiektG1	ObiektG2
a) wg (2-2)	`k=2;  T1=2;` `ObiektG1a= tf(k, [T1, 1]);`	`k=2;  T1=2;  T2=4;  Tz1=0.5;` `tabL = [k*Tz1, k];`

[1] patrz skrypt_21_1tf.m

		`tabM = [T1*T2, T1+T2, 1];` `ObiektG2a= tf(tabL, tabM);`

b) wg (2-3)	`k=2;  T1=2;`
	`ObiektG1b= zpk([], [-1/T1], k/T1);`
		`k=2;  T1=2;  T2=4;  Tz1=0.5;`
		`tabZ = [-1/Tz1];`
		`tabP = [-1/T1, -1/T2];`
		`ObiektG2b= zpk(tabZ, tabP, k*Tz1/(T1*T2));`

c) wg (2-4)	`k=2;  T1=2;`
	`s=tf('s');`
	`ObiektG1c= k/(T1*s+1);`
		`k=2;  T1=2;  T2=4;  Tz1=0.5;`
		`s=tf('s');`
		`ObiektG2c= k*(Tz1*s+1)/((T1*s+1)*(T2*s+1));`

Różne formy transmitancji danego obiektu są równoważne, jednak sposób prezentowania ich w Matlabie zależy od zastosowanego wariantu definicji, na przykład:

>>ObiektG2a	>>ObiektG2b	>>ObiektG2c
Transfer function: s + 2 --- 8 s^2 + 6 s + 1	Zero/pole/gain: 0.125 (s+2) --- (s+0.5) (s+0.25)	Transfer function: s + 2 --- 8 s^2 + 6 s + 1

Parametry dowolnej transmitancji można odczytać (obliczyć) za pomocą następujących funkcji: **`dcgain()`** – wzmocnienie układu; **`pole()`** – bieguny układu; **`zero()`** – zera układu.

>> dcgain(ObiektG2a)	>> pole(ObiektG2a)	>> zero(ObiektG2a)	>> damp(ObiektG2a)		
ans =	ans =	ans =	Eigenvalue	Damping	Freq. (rad/s)
2	-0.5000	-2	-2.50e-001	1.00e+000	2.50e-001
	-0.2500		-5.00e-001	1.00e+000	5.00e-001

Można też zastosować funkcję **`damp()`**, która w przypadku układów inercyjnych zwraca wartości biegunów $p_{1,2}$ (Eigenvalue), tłumienie (Damping) zawsze równe 1 i pulsacje (Freq) interpretowane przez Matlab jako odwrotność stałych czasowych.

Przykład: Obiekt oscylacyjny G3. Wśród podstawowych obiektów dynamiki, szczególne znaczenie praktyczne ma człon oscylacyjny:

$$G(s) = \frac{k_1}{s^2 + 2\xi\,\omega_n\, s + \omega_n^2} = \frac{k_1}{(s - p_1)(s - p_2)},\ \omega_n > 0 \qquad (2\text{-}5)$$

$$p_{1,2} = -\xi\,\omega_n \pm \omega_n\sqrt{\xi^2 - 1}$$

Ponieważ typ biegunów $p_{1,2}$ zależy od wartości tłumienia ξ, to człon może reprezentować bardzo odmienne własności. W przykładach w tej części podręcznika będzie wykorzystywany obiekt stabilny z oscylacjami ($0<\xi<1$), opisany następujący sposób:

- **ObiektG3**: wzmocnienie jednostkowe, tłumienie 0.4, a pulsacja drgań własnych 0.1rad/sek (tzn. częstotliwość około 15.9 mHz).

Transmitancję o powyższych własnościach można zdefiniować na różne sposoby (2-2), (2-3), (2-5).

Tab. 2-2. Parametry przykładowego układu ObiektG3

	ObiektG3
Dane układu:	$k_{ukł}$=1, $\xi = 0.4$, $\omega_n = 0.1$
Odtworzenie transmitancji na podstawie danych:	$\xi = 0.4$, $\omega_n = 0.1$, $\lim_{s\to 0}\frac{k_1}{s^2 + 2\xi\,\omega_n\, s + \omega_n^2} = 1 \rightarrow k_1 = \omega_n^2$
a) współczynniki do (2-2)	a_2=1, a_1=0.08, a_0=0.01, b_0=0.01
b) bieguny i zera do (2-3)	k_1= 0.01, p_1= -0.04-j0.0917, p_2= -0.04+j0.0917
c) człon oscylacyjny do (2-5)	**k_1= 0.01, ξ=0.4, ω_n=0.1**

Poniżej przedstawiono różne warianty definicji transmitancji w skrypcie:

	ObiektG3
a) wg (2-2)	`ksi=0.4; w=0.1;` `ObiektG3a= tf(w^2, [1, 2*ksi*w, w^2]);`

b) wg (2-3)	`ksi=0.4; w=0.1;  delta=(2*ksi*w)^2 - 4*w*w;`
	`p1 = (-2*ksi*w+sqrt(delta))/2;`
	`p2 = (-2*ksi*w-sqrt(delta))/2;`
	`ObiektG3b= zpk([],[p1, p2], w^2);`

c) wg (2-5)	`ksi=0.4; w=0.1;`
	`s=tf('s');`
	`ObiektG3c= w*w/(s^2+2*ksi*w*s+w*w);`

Ze względu na zespolone wartości biegunów Matlab prezentuje wszystkie trzy warianty transmitancji w ten sam sposób, niezależnie od sposobu definicji obiektu:

```
>>ObiektG3a                 >>ObiektG3b                 >>ObiektG3c
  Transfer function:          Zero/pole/gain:             Transfer function:
        0.01                        0.01                        0.01
  ----------------------      ----------------------      ----------------------
  s^2 + 00.8 s + 0.01         s^2 + 00.8 s + 0.01         s^2 + 00.8 s + 0.01
```

Parametry członu oscylacyjnego można odczytać za pomocą funkcji `dcgain()`, `pole()` i `zero()`, ale przede wszystkim za pomocą funkcji `damp()`, która zwraca wartości biegunów $p_{1,2}$ (Eigenvalue), tłumienie ξ (Damping) i pulsację drgań własnych ω_n (Freq.)

```
>> dcgain(ObiektG3a)
ans =
     1

>> pole(ObiektG3a)
ans =
  -0.0400 + 0.0917i
  -0.0400 - 0.0917i

>> zero(ObiektG3a)
ans =

>> damp(ObiektG3a)
      Eigenvalue              Damping      Freq. (rad/s)
-4.00e-002 + 9.17e-002i   4.00e-001      1.00e-001
-4.00e-002 - 9.17e-002i   4.00e-001      1.00e-001
```

Transmitancje złożonych układów można oczywiście wyznaczyć analityczne i zdefiniować analogicznie jak opisane przypadki. Ale można też konstruować schematy blokowe za pomocą funkcji, realizujących elementarne połączenia szeregowe, równoległe, sprzężenie zwrotne (p. 2.3).

Przykład: Obiekty nietypowe. Typowe transmitancje mają postać funkcji wymiernych a stopień wielomianu w liczniku jest mniejszy niż stopień wielomianu w mianowniku. W tej definicji nie mieszczą się szczególne przypadki transmitancji, takie jak:

a) wzmocnienie	b) człon różniczkujący	c) opóźnienie	d) przesuwnik fazowy
$G_k(s) = k$	$G_d(s) = T_d s$	$G_{op}(s) = e^{-sT_o}$	

(2-6)

Na etapie definicji takich modeli w Matlabie nie ma problemu:

```
>>Gk = tf(2)                >> Gd = 2*s       %lub: Gd = tf([2, 0], [1])
  Transfer function:          Transfer function:
  2                           2 s

>> Gop = exp(-s*2)
     Transfer function:
     exp(-2*s) * (1)
```

Jeśli transmitancja ma więcej zer niż biegunów, to pojawiają się problemy podczas symulacji. Czy tu przesuwnik fazowy? W starszych wersjach Matlaba (lub w „klonach" Matlaba ....
Rozwiązania: człon różniczkujący rzeczywisty, aproksymacja Pade (badanie własności później – jako zadanie)

2.1.2 Definiowanie modelu MIMO[1]

Typową formą liniowych modeli o wielu wejściach i wyjściach (MIMO) są **równania stanu**. Modelem tego typu jest na przykład układ drugiego rzędu z dwoma sygnałami wejściowymi:

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix} \tag{2-7}$$

Równania stanu zawierają pełny opis dynamiki układu, ale często są uzupełniane równaniami wyjściowymi, które na podstawie zmiennych stanu $x(t)$ i sygnałów wejściowych $u(t)$ definiują dowolną ilość zmiennych wyjściowych $y(t)$. W Matlabie wielokrotnie zachodzi potrzeba zdefiniowania zmiennych wyjściowych, które są równe zmiennych stanu:

$$\begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix} \tag{2-8}$$

Ponieważ Matlab jest ukierunkowany na operacje na macierzach, stąd zapis parametrów funkcji często opiera się na ogólnej postaci równań stanu i równań wyjściowych:

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \\ \mathbf{y}(t) &= \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t) \end{aligned} \tag{2-9}$$

gdzie: **x** – wektor zmiennych stanu; **u** – wektor zmiennych wejściowych; **y** – wektor zmiennych wyjściowych, **A**, **B**, **C**, **D** – macierze o odpowiednich wymiarach.

Model MIMO można również zdefiniować **za pomocą transmitancji**. Równaniom stanu (2-7) odpowiadają wówczas cztery transmitancje:

$$x_1(s) = \frac{L_{11}(s)}{M(s)} u_1(s) + \frac{L_{12}(s)}{M(s)} u_2(s) \tag{2-10}$$

[1] patrz skrypt_21_1ss.m

$$x_2(s)=\frac{L_{21}(s)}{M(s)}u_1(s)+\frac{L_{22}(s)}{M(s)}u_2(s)$$

gdzie: $L_{11}(s)=b_{11}s+a_{12}b_{21}-a_{22}b_{11}$, $L_{12}(s)=b_{12}s+a_{12}b_{22}-a_{22}b_{12}$

$L_{21}(s)=b_{21}s+a_{21}b_{11}-a_{11}b_{21}$, $L_{22}(s)=b_{22}s+a_{21}b_{12}-a_{11}b_{22}$

$M(s)=(s-a_{11})(s-a_{22})-a_{12}a_{21}=s^2-(a_{11}+a_{22})s+(a_{11}a_{22}-a_{12}a_{21})$

Przykład: Obiekt M1 i M2. W przykładach prezentowanych w tej części podręcznika, będą wykorzystywane dwa obiekty MIMO opisane za pomocą następujących równań i transmitancji:

- ~~**ObiektM1**: przykładowy układ unilateralny (typu kaskada niewspółdziałająca)~~

$$\begin{cases}\dot{x}_1(t)=-x_1(t)+2u_1(t)\\ \dot{x}_2(t)=x_1(t)-2x_2(t)+u_2(t)\end{cases}\rightarrow\begin{array}{l}x_1(s)=\dfrac{2}{s+1}u_1(s)\\ x_2(s)=\dfrac{2}{(s+1)(s+2)}u_1(s)+\dfrac{1}{(s+2)}u_2(s)\end{array}$$

~~W równaniach stanu widoczna jest unilateralna struktura układu, która uzasadnia uproszczone postaci transmitancji układu.~~

- **ObiektM2**: przykładowy układ nieunilateralny (typu kaskada współdziałająca)

$$\begin{cases}\dot{x}_1(t)=-x_1(t)+x_2(t)+2u_1(t)\\ \dot{x}_2(t)=x_1(t)-2x_2(t)+u_2(t)\end{cases}\rightarrow\begin{array}{l}x_1(s)=\dfrac{2(s+2)}{(s+1)(s+2)-1}u_1(s)+\dfrac{1}{(s+1)(s+2)-1}u_2(s)\\ x_2(s)=\dfrac{2}{(s+1)(s+2)-1}u_1(s)+\dfrac{s+1}{(s+1)(s+2)-1}u_2(s)\end{array}$$

Tab. 2-3. Parametry przykładowych układów ObiektM1 i ObiektM2.

Parametry:	ObiektM1	ObiektM2
a) macierze równań stanu do (2-7)	$\mathbf{A}=\begin{bmatrix}-1 & 0\\ 1 & -2\end{bmatrix}$, $\mathbf{B}=\begin{bmatrix}2 & 0\\ 0 & 1\end{bmatrix}$	$\mathbf{A}=\begin{bmatrix}-1 & 1\\ 1 & -2\end{bmatrix}$, $\mathbf{B}=\begin{bmatrix}2 & 0\\ 0 & 1\end{bmatrix}$
b) współczynniki transmitancji do (2-10)	$L_{11}=2$, $M_{11}=s+1$, $L_{21}=2$, $L_{22}=1$, $M_{21}=s^2+3s+2$, $M_{21}=s+2$	$L_{11}=2s+4$, $L_{12}=1$, $L_{21}=2$, $L_{22}=s+1$, $M=s^2+3s+1$

Poniższy skrypt przedstawia **definicję równań stanu** za pomocą funkcji `ss()` i `tf()`.

	ObiektM1	ObiektM2

a)	`A=[-1 0;1 -2]; B=[2 0; 0 1];`
	`C=eye(2); D=zeros(2);`
	
	`ObiektM1a = ss(A,B,C,D);`
		`A=[-1 1;1 -2]; B=[2 0; 0 1];`
		`C=eye(2); D=zeros(2);`
		
		`ObiektM2a = ss(A,B,C,D);`

b)	`L11=[2];  L12=[0];`
	`M11=[1 1];  M12=[1];`
	
	`L21=[2];  L22=[1];`
	`M21=[1 3 2];  M22=[1 2];`
	
	`ObiektM1b = tf( {L11,L12;L21,L22},...`
	`{M11,M12;M21,M22})`
		`L11=[2 4];  L12=[1];`
		`L21=[2];  L22=[1 1];`
		
		`M=[1 3 1];`
		
		`ObiektM2b = tf( {L11,L12;L21,L22},...`
		`{M,M;M,M})`

Zdefiniowane obiekty są prezentowane w następujący sposób

```
>> ObiektM1a
a =
     x1  x2
  x1 -1   0
  x2  1  -2

b =
     u1  u2
  x1  2   0
  x2  0   1

c =
     x1  x2
```

```
>> ObiektM1b
Transfer function from input 1 to output...
      2
 #1: -----
    s + 1

        2
 #2: -------------
    s^2 + 3 s + 2

Transfer function from input 2 to output...
```

```
>> ObiektM2a
a =
     x1  x2
  x1 -1   1
  x2  1  -2

b =
     u1  u2
  x1  2   0
  x2  0   1

c =
     x1  x2
```

```
>> ObiektM2b
 Transfer function from input 1 to output...
      2 s + 4
 #1: -------------
    s^2 + 3 s + 1

        2
 #2: -------------
    s^2 + 3 s + 1

Transfer function from input 2 to output...
```

```
  y1  1  0          #1:  0                    y1  1  0              1
  y2  0  1                                    y2  0  1        #1:  ------------
                          1                                        s^2 + 3 s + 1
d =                 #2:  -----          d =
     u1  u2              s + 2                   u1  u2              s + 1
  y1  0  0                                    y1  0  0        #2:  ------------
  y2  0  0                                    y2  0  0             s^2 + 3 s + 1
```

Na bazie obiektów MIMO mogą być również definiowane złożone układy (p. 2.3).

Podczas definicji modelu za pomocą funkcji `ss()` można wprowadzić własne nazwy zmiennych wejściowych, zmiennych stanu i zmiennych wyjściowych:

```
ObiektM1a = ss(A1,B1,C1,D1,'InputName',['wej1';'wej2'], ...
           'StateName',['zx1';'zx2'],'OutputName',['wyj1';'wyj2'])
```

Uwaga - nazwy w poszczególnych grupach zmiennych muszą mieć taką samą długość (Matlab przechowuje je w macierzy).

Zdefiniowane nazwy są używane jako opisy wykresów, a także przy prezentacji obiektów:

```
a =                 b =                 c =                 d =
      zx1   zx2           wej1  wej2           zx1   zx2           wej1  wej2
 zx1   -1     0     zx1     2     0     wyj1    1     0     wyj1    0     0
 zx2    1    -2     zx2     0     1     wyj2    0     1     wyj2    0     0
```

W starszych wersjach Matlaba funkcja `tf()` nie umożliwia wprowadzania własnych nazw podczas definicji modelu, jednak podczas konwersji modelu z równań stanu (p.2.1.3) stosuje nazwy zdefiniowane dla równań stanu.

2.1.3 Konwersja i parametry modelu

Model zdefiniowany na podstawie dowolnej transmitancji czy równań stanu można przekonwertować na inne formy obiektów LTI i wyświetlić je, lub też odczytać wartości charakterystycznych parametrów.

Konwersja dowolnej transmitancji na postać zpk (zero/pole/gain – zero/biegun/wzmocnienie) pozwala odczytać wartości biegunów i zer modelu. Parametry te można też odczytać używając funkcji **`zpkdata()`** lub **`pole()`** i **`zero()`**.

```
>> zpk(ObiektG2a)
    0.125 (s+2)
  ---------------
  (s+0.5) (s+0.25)
```

```
>> [Z,P,K] = zpkdata(ObiektG2a,'v')
  Z =   -2
  P =  -0.5000
       -0.2500
  K =  0.1250
```

```
>>pole(ObiektG2a)
   -0.5000
   -0.2500
```

```
>> zero(ObiektG2a)
   -2
```

Uwaga: wzmocnienie, które jest parametrem modelu zpk nie oznacza wzmocnienia układu, które jest wyznaczane za pomocą funkcja **`dcgain()`** .

Postaci kanoniczne [Franklin/452=443, 454][Amborski/29-30, 32]

$$\frac{b(s)}{a(s)} = \frac{b_1 s^{n-1} + b_2 s^{n-2} + \ldots + b_n}{s^n + a_1 s^{n-1} + a_2 s^{n-2} + \ldots + a_n} = \frac{k_1}{s - p_1} + \frac{k_2}{s - p_2} + \ldots$$

– **control cannonical form** (CCF)

$$\dot{\mathbf{x}} = \mathbf{A_c}\mathbf{x} + \mathbf{B_c}u$$
$$y = \mathbf{C_c}\mathbf{x} + D_c u$$
$$\mathbf{A_c} = \begin{bmatrix} -a_1 & -a_2 & \cdots & \cdots & -a_n \\ 1 & 0 & \cdots & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ \vdots & & \ddots & 0 & \cdots \\ 0 & 0 & \cdots & 1 & 0 \end{bmatrix}, \mathbf{B_c} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$
$$\mathbf{C_c} = \begin{bmatrix} b_1 & b_2 & \cdots & \cdots & b_n \end{bmatrix}, \; D_c = 0$$

CCF – nazywana jest też upper companion form (bo współczynniki równania char. są w wierszu 1)

– **modal cannonical form** (MCF)

$$\dot{\mathbf{x}} = \mathbf{A_m}\mathbf{x} + \mathbf{B_m}u$$
$$y = \mathbf{C_m}\mathbf{x} + D_m u$$
$$\mathbf{A_m} = \begin{bmatrix} p_1 & 0 & \cdots \\ 0 & p_2 & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix}, \; \mathbf{B_m} = \begin{bmatrix} 1 \\ 1 \\ \vdots \end{bmatrix}$$
$$\mathbf{C_m} = \begin{bmatrix} k_1 & k_2 & \cdots \end{bmatrix}, \; D_m = 0$$

$\mathbf{A_m}$ – macierz modalna

Z kolei po konwersji dowolnej formy transmitancji na równania stanu, model jest prezentowany w postaci macierzy A, B, C, D:

```
>> ss(ObiektG2a)
a =
        x1     x2
  x1  -0.75   -0.5
  x2   0.25      0

b =
      u1
  x1   1
  x2   0

c =
        x1     x2
  y1  0.125      1

d =
      u1
  y1   0
```

```
>>
```

Dodać metodę konwersji tf na ss. Konwersja ss na tf

ObiektM1 (obiekt unilateralny)– transmitancje uproszczone

Założenie o skracalności Oblicza transmitancje y/u (a nie x/u)

2.2. Charakterystyki i własności

2.2.1 Formatowanie wykresów

Jednym z najprostszych sposobów badania własności dynamicznych obiektów jest analizowanie charakterystyk narysowanych za pomocą funkcji **`plot()`**. W skryptach, które zilustrują podstawowe metody analizy własności obiektów/układów wykorzystywane będą obiekty zdefiniowane wcześniej (p. 2.1). W prezentowanych skryptach pominięto niektóre operacje formatujące, które pozwalają w prosty sposób dopasować wygenerowane wykresy do wymogów dokumentacji. Operacje te wykorzystują obiektowy charakter graficznych okien Matlaba oraz związane z tym funkcje, takie jak:

- odczytywanie i zapisywanie własności obiektów: **`get()`**, **`set()`**,
- odczytywanie identyfikatorów (uchwytów) obiektów: **`gcf()`**, **`gca()`**[1]

Typowe operacje stosowane na wykresach wygenerowanych za pomocą funkcji **`plot()`** to:

F1) ustawienie białego koloru tła aktywnego okna graficznego

```
set(gcf(), 'Color',[1,1,1]);    %biały kolor tła
```

F2) zmiana wielkości aktywnego okna graficznego

```
pos=get(gcf(),'Position');      %odczytaj pozycję i wielkość
pos(3)=300;  pos(4)=180;        %pos(3)-szerokość, pos(4)-wysokość
set(gcf(), 'Position', pos);    %ustaw pozycję i wielkość
```

Dla zapewnienia czytelności wykresów (np. opisów) stosowane są również operacje na tekstach opisujących tytuły i osie wykresów, takie jak:

F3) ustawienie wielkości fontów i grubości linii dla siatki wykresu

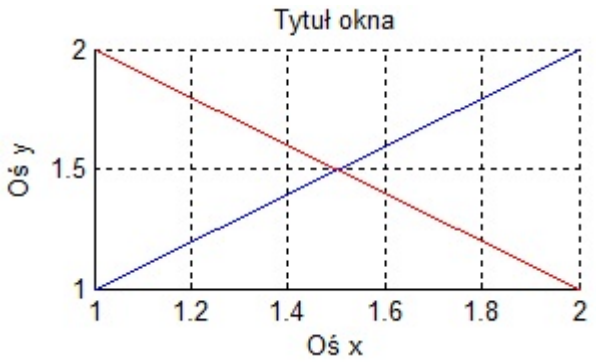
```
set(gca(), 'FontSize', 7)       %opis siatki (skali), domyślnie: 10
set(gca(), 'LineWidth', 1)      %grubość linii siatki, domyślnie: 0.5
```

F4) ustawienie wielkości, koloru i typu fontów w tytułach i opisach osi wykresu

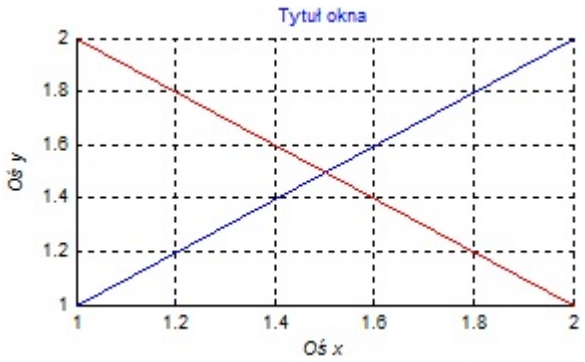
```
set(get(gca(),'Title'), 'FontSize', 7,'Color',[0 0 1])    % domyślnie: 10;[0 0 0]
set(get(gca(),'XLabel'),'FontAngle','italic','FontSize', 7)
set(get(gca(),'YLabel'),'FontAngle','italic','FontSize', 7)
```

Na rys. 2.1 i rys. 2.2 przedstawiono pośredni i końcowy wykres wygenerowany przez skrypt:

```
figure; hold on, grid on
plot([1,2],[1,2]); plot([2,1],[1,2],'r');
title('Tytuł okna');
ylabel('Oś y'); xlabel('Oś x');

%operacje F1÷2
set(gcf(), 'Color',[1,1,1]);    %biały kolor tła
pos=get(gcf(),'Position');      %odczyt pozycji i wielkości
pos(3)=320;  pos(4)=180;        %pos(3)-szerokość, pos(4)-wysokość
set(gcf(), 'Position', pos);    %ustaw pozycję i wielkość

%operacje F3÷4
set(gca(), 'FontSize', 7)       %opis siatki (skali), domyślnie: 10
set(gca(), 'LineWidth', 1)      %grubość linii siatki, domyślnie: 0.5
set(get(gca(),'Title'), 'FontSize', 7,'Color',[0 0 1])    % domyślnie: 10;[0 0 0]
set(get(gca(),'XLabel'),'FontAngle','italic','FontSize', 7)
set(get(gca(),'YLabel'),'FontAngle','italic','FontSize', 7)
```



Rys. 2.1. Wykres po operacjach F1÷2



Rys. 2.2. Wykres po operacjach F1÷4

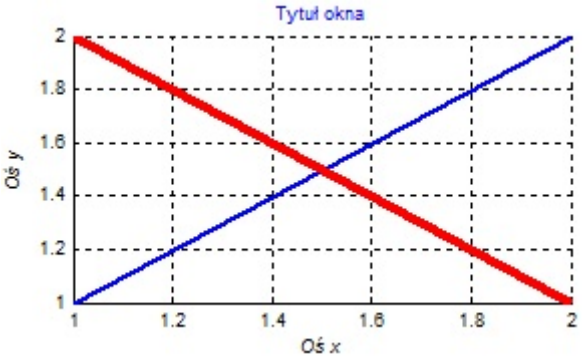
[1] Uwaga: Identyfikatory niektórych obiektów można również uzyskać i zapamiętać podczas generowania tych obiektów

Możliwe jest również formatowanie narysowanych linii poszczególnych wykresów, na przykład:

F5) zmiana grubości linii wykresów

```
tabline = get(gca(),'Children') %odczyt identyfikatorów linii
set(tabline(1),'LineWidth',3)   %ustaw grubość 1.linii
set(tabline(2),'LineWidth',2)   %ustaw grubość 1.linii
```

Po zastosowaniu operacji F5 wykres z rys. 2.2 uzyskuje końcową postać jak na rys. 2.3.

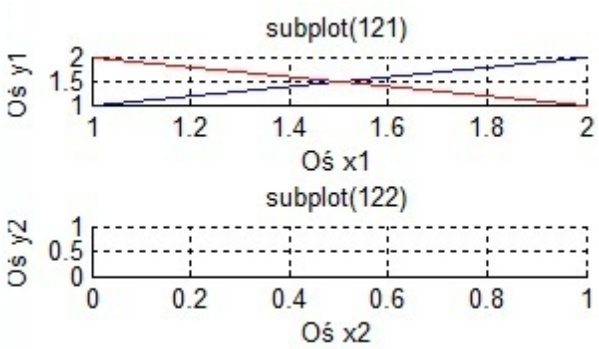


Rys. 2.3. Wykres po operacjach F1÷5

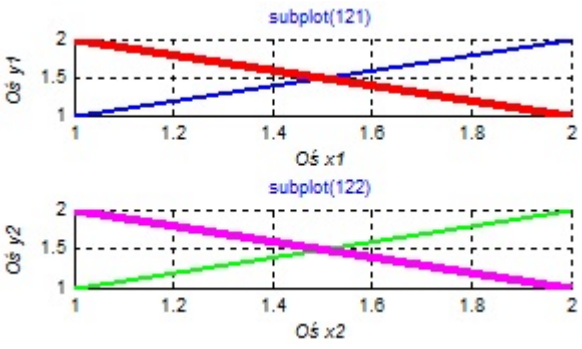
Analogiczne formatowanie można przeprowadzić kilku wykresach wygenerowanych za pomocą funkcji **`subplot()`**, w jednym oknie graficznym. Wymaga to odczytania identyfikatorów i powtórzenia operacji na kilku wykresach:

```
figure;
title('Tytuł okna');
subplot(211);hold on, grid on
plot([1,2],[1,2]);plot([2,1],[1,2],'r');
title('subplot(121)'); ylabel('Oś y1'); xlabel('Oś x1');

subplot(212);hold on, grid on
plot([1,2],[1,2],'g');plot([2,1],[1,2],'m');
title('subplot(122)'); ylabel('Oś y2'); xlabel('Oś x2');

%operacje formatujące F1-2
set(gcf(), 'Color',[1,1,1]);     %biały kolor tła
pos=get(gcf(),'Position');       %odczytaj pozycję i wielkość
pos(3)=320;  pos(4)=180;         %pos(3)-szerokość, pos(4)-wysokość
set(gcf(), 'Position', pos);     %ustaw pozycję i wielkość

tabaxis = get(gcf(),'Children');    %odczyt identyfikatorów wykresów (axes)
for i=1:size(tabaxis,1)
    %operacje formatujące F3-4
    set(tabaxis(i), 'FontSize', 7)        %opis siatki (skali), domyślnie: 10
    set(tabaxis(i), 'LineWidth', 1)       %grubość linii siatki, domyślnie: 0.5
    set(get(tabaxis(i),'Title'), 'FontSize', 7,'Color',[0 0 1])
    set(get(tabaxis(i),'XLabel'),'FontAngle','italic','FontSize', 7)
    set(get(tabaxis(i),'YLabel'),'FontAngle','italic','FontSize', 7)

    %operacje formatujące F5
    tabline = get(tabaxis(i),'Children') %odczyt identyfikatorów linii
    set(tabline(1),'LineWidth',3)        %ustaw grubość 1.linii
    set(tabline(2),'LineWidth',2)        %ustaw grubość 2.linii
end
```

Na rys. 2.4 i rys. 2.5 przedstawiono pośredni i końcowy efekt wykonania powyższego skryptu.



Rys. 2.4. Wykresy po operacjach F1÷2



Rys. 2.5. Wykresy po operacjach F1÷5

Skrypty przedstawione powyżej wykorzystują strukturę okna graficznego jaką generuje funkcja `plot()`, w której obiektami potomnymi okna graficznego są tylko obiekty typu osie (`axes`), a obiektami osi są tylko obiekty typu linie (`line`).

W badaniach własności dynamicznych często wykorzystywane są jeszcze inne narzędzia, np. `step()`, `impluse()`. Są to funkcje, które dostarczają interfejs graficzny, to znaczy, że nie tylko rysują określone typy wykresów ale umożliwiają dodatkowe funkcje, np. formatowanie wykresy, odczytywanie wartości. Po wykonaniu takich funkcji w strukturze okien i wykresów tworzone są również inne typy obiektów. Poniższy fragmentu skryptu realizuje bardziej **uniwersalną formę realizacji operacji formatowania** F1÷5:

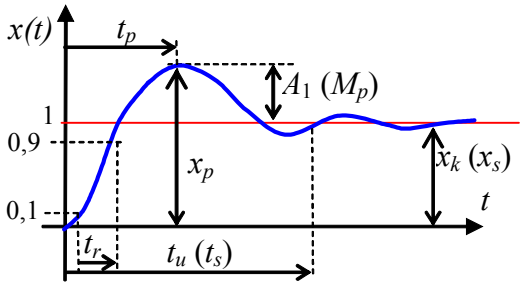
```
%operacje formatujące F1-2
set(gcf(), 'Color',[1,1,1]);     %biały kolor tła
pos=get(gcf(),'Position');       %odczytaj pozycję i wielkość
pos(3)=320;  pos(4)=180;         %pos(3)-szerokość, pos(4)-wysokość
set(gcf(), 'Position', pos);     %ustaw pozycję i wielkość

tabaxis = get(gcf(),'Children')     %odczyt identyfikatorów wykresów (axes)
for i=1:size(tabaxis,1)
    if strcmp (get(tabaxis(i), 'Type'), 'axes') %jeśli to wykres (osie)

        %operacje formatujące F3-4
        set(tabaxis(i), 'FontSize', 7)       %opis siatki (skali), domyślnie: 10
        set(tabaxis(i), 'LineWidth', 1)      %grubość linii siatki, domyślnie: 0.5
        set(get(tabaxis(i),'Title'), 'FontSize', 7,'Color',[0 0 1])
        set(get(tabaxis(i),'XLabel'),'FontAngle','italic','FontSize', 7)
        set(get(tabaxis(i),'YLabel'),'FontAngle','italic','FontSize', 7)

        %operacje formatujące F5
        tabline = get(tabaxis(i),'Children') %odczyt identyfikatorów linii
        for j=1:size(tabline,1)
            if strcmp (get(tabline(j), 'Type'), 'line') %jeśli to linia (line)
                set(tabline(j),'LineWidth',2)           %ustaw grubość linii
            end
        end
    end

end
```

Uwaga: Ze względu na funkcje realizowane przez interfejsy graficzne, operacja zmiany własności obiektu nie zawsze przynosi efekt (własności wyświetlanego obiektu mogą być blokowane ze względu na powiązane z innymi obiektami/zmiennymi interfejsu).

2.2.2 Charakterystyki czasowe i analiza w dziedzinie czasu (Time-domain analysis)

Definicje i parametry. Charakterystyki czasowe to reakcje układu na najprostsze wymuszenia, takie jak skokowa i impulsowa zmiana wartości wejściowej. Jest to bardzo intuicyjna forma prezentacji własności dynamicznych układu i opisania ich za pomocą parametrów (rys. 2.6):

- x_k – wartość końcowa, stan ustalony (x_s - steady state, final value)
- t_u – czas ustalania, regulacji (t_s - settling time)
- x_p – wartość szczytowa, maksymalna (peak amplitude)/
- A_1 – przeregulowanie (M_p – overshoot)
- t_p – czas pierwszego przeregulowania (peak time)
- t_r – czas narostu (rise time)



Rys. 2.6. Parametry odpowiedzi skokowej

Obliczenia analityczne. W przypadku podstawowych układów dynamiki powyższe parametry można dość łatwo obliczyć, ponieważ znane są rozwiązania analityczne (B.4). Badania przedstawimy na przykładzie ogólnych wzorów członu oscylacyjnego ObiektG3:

$$G_3(s) = \frac{\omega_n^2}{s^2 + 2\xi\,\omega_n\, s + \omega_n^2}, \tag{2-11}$$

który dla $|\xi| < 1$ ma dwa bieguny zespolone:

$$p_{1,2} = -\sigma \pm j\omega_r, \text{ gdzie: } \sigma = \xi\omega_n,\ \omega_r = \omega_n\sqrt{1-\xi^2}, \tag{2-12}$$

Odpowiedź skokową można przedstawić w postaci:

$$x(t) = 1 - e^{-\sigma t}\left(\cos\omega_r t + \frac{\sigma}{\omega_r}\sin\omega_r t\right) \tag{2-13}$$

Wartość końcowa x_k wynosi 1, co można stwierdzić na podstawie (2-11) lub (2-13).
Badanie przebiegu zmienności funkcji pozwala wyznaczyć czas wystąpienia maksimum reakcji:

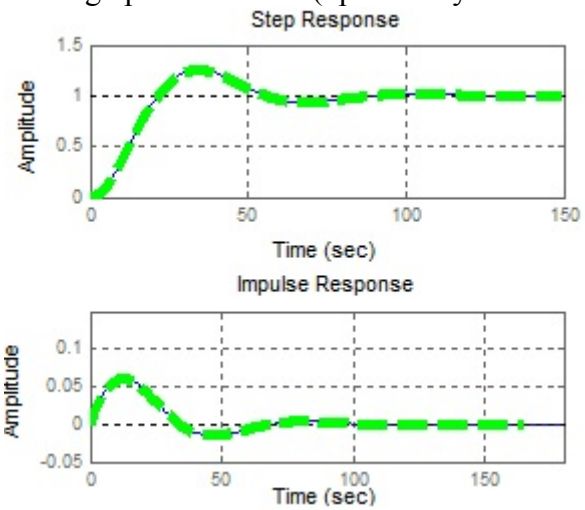
$$\dot{x}(t) = e^{-\sigma t}\left(\frac{\sigma^2}{\omega_r}\sin\omega_r t + \omega_r\sin\omega_r t\right) = 0 \tag{2-14}$$

$$\rightarrow \sin(\omega_r t) = 0 \quad \rightarrow \quad \omega_r t = \pi \quad \rightarrow \quad t_p = \frac{\pi}{\omega_r} = \frac{\pi}{\omega_n\sqrt{1-\xi^2}}$$

Co umożliwia obliczenie wartości tego maksimum i przeregulowanie A_1:

$$x_p = x(t_p) = 1 - e^{-\sigma\pi/\omega_r}\left(\cos\omega_r\frac{\pi}{\omega_r} + \frac{\sigma}{\omega_r}\sin\omega_r\frac{\pi}{\omega_r}\right) = 1 + e^{-\sigma\pi/\omega_r} = 1 + A_1 \tag{2-15}$$

Badania symulacyjne. Podstawowe funkcje do rysowania charakterystyk czasowych, to **`step()`** i **`impluse()`**. Pozwalają one narysować wykres lub zapamiętać dane (wektory wartości) do dalszego przetwarzania (np. do narysowania za pomocą `plot()`), co zilustrowano na rys. 2.7.



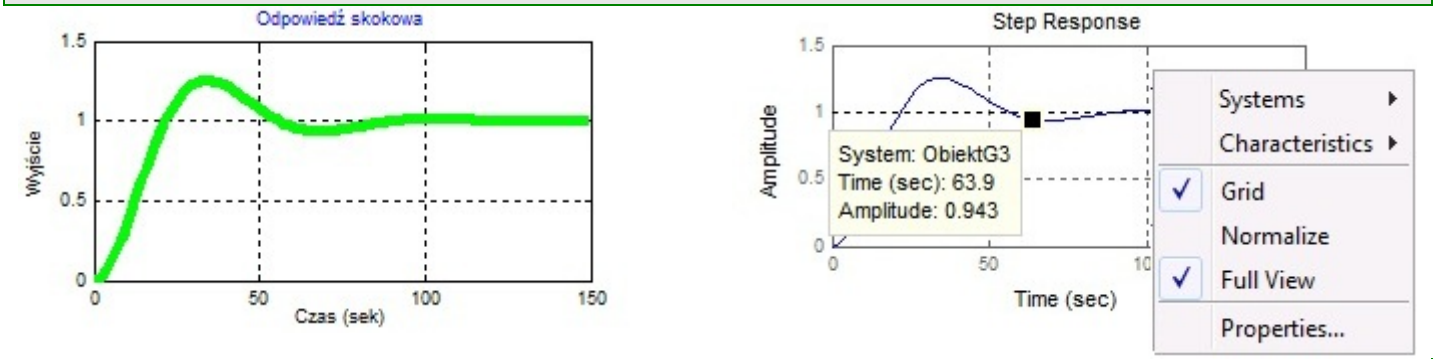
Rys. 2.7. Odpowiedzi czasowe układu (ObiektG2) narysowane za pomocą funkcji step (—) i plot (--)

```
%definicja ObiektG2 ....
figure; set(gcf(), 'Color',[1,1,1]);

%odpowiedź skokowa (na dwa sposoby)
subplot(211);  hold on;
step(ObiektG2);                 %niebieska linia

[y, t] = step(ObiektG2);
plot(t, y, 'g--');              %zielona --

%odpowiedź impulsowa (na dwa sposoby)
subplot(212); hold on;
impulse(ObiektG2);              %niebieska linia

[y, t] = impulse(ObiektG2);
plot(t, y, 'g--');              %zielona --

%operacje formatowania ...
```

Rysowanie wykresu za pomocą funkcji `plot()` wymaga samodzielnego wygenerowania tytułu i opisu osi, ale łatwo można formatować wykres poprzez zmianę własności obiektów graficznych za pomocą operacji w skrypcie (przykład poniżej). Funkcje `step()` i `impluse()` poza rysowaniem wykresu, automatycznie dodają standardowy tytuł i opis osi oraz dostarczają dodatkowych funkcjonalności poprzez menu kontekstowe, takich jak odczytywanie dowolnej wartości na wykresie (lewe menu), formatowanie wykresu (prawe menu / Properties) – **Błąd! Nie można odnaleźć źródła odwołania.**.

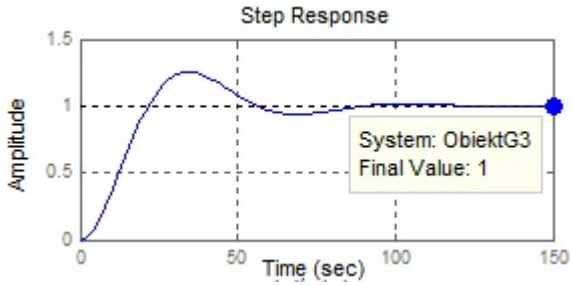
```
%definicja ObiektG3 ....
figure; set(gcf(), 'Color',[1,1,1]);

[y, t] = step(ObiektG3);
plot(t, y, 'g--'); grid on
title('Odpowiedź skokowa');
ylabel('Wyjście'); xlabel('Czas (sek)')
%operacje formatowania w skrypcie ...
```

```
%definicja ObiektG3 ....
figure; set(gcf(), 'Color',[1,1,1]);

step(ObiektG3);
%operacje formatowania poprzez menu
```

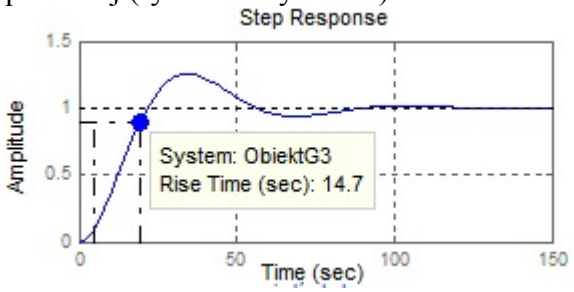


Rys. 2.8. Funkcjonalność i `plot()` i `step()` ObiektG3

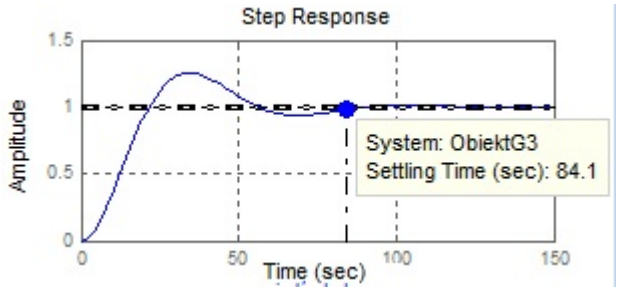
W menu kontekstowym (prawe menu / Characteristics) zaznaczenie punktów do odczytywania podstawowych parametrów odpowiedzi skokowej/impulsowej (rys. 2.9 ÷ rys. 2.12).



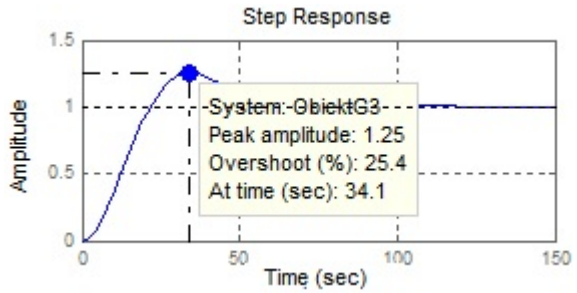
Rys. 2.9. Opcja Steady State - wartość końcowa x_k (x_s)



Rys. 2.10. Opcja Rise Time - czas narostu t_r



Rys. 2.11. Opcja Settling Time - czas ustalania t_u (t_s)



Rys. 2.12. Opcja Peak Response - x_p, $A_1(M_p)$, t_p

Wszystkie parametry odczytywane z wykresów można uzyskać również za pomocą funkcji **`stepinfo()`**, którą można uruchomić na kilka sposób

RiseTime: 13.0029

SettlingTime: 43.3765

SettlingMin: 0.9053

SettlingMax: 1.1158

Overshoot: 11.5818

Undershoot: 0

Peak: 1.1158

PeakTime: 27.0627

W tab. 2-4 przedstawiono porównanie wartości parametrów odpowiedzi skokowej układu ObiektG3 (ξ=0.4, ω_n=0.1) obliczonych na podstawie wzorów i odczytanych z wykresów.

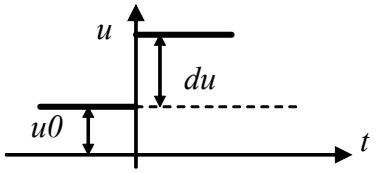
Tab. 2-4. Parametry układu ObiektG3 obliczone na podstawie (2-11)÷(2-15) i odczytane z (rys. 2.9 ÷ rys. 2.12)

Nazwa parametru	ze wzorów	Nazwa na wykresach	Wartość z wykresów
Wartość końcowa x_k	1	Final value (x_s)	1
Czas narostu t_r		Rise time (t_r)	14.7
Czas ustalania t_u		Settling time (t_s)	84.1
Wartość maksymalna x_p	1.2538	Peak amplitude (x_p)	1.25
Przeregulowanie A_1 w %	25.38	Overshoot % (M_p)	25.4
Czas przeregulowania t_p	34.2776	Peak time (t_p)	34.1

Wyznaczenie parametrów przebiegu czasowego wymaga wygenerowania tego przebiegu – funkcja `step()` nie wyznacza ich analitycznie, tylko na podstawie danych z symulacji (nie ma innych funkcji, które podawałyby parametry bez rysowania rozwiązania).

Funkcja `step()` generuje odpowiedź skokową badanego układu, czyli odpowiedź na skok jednostkowy pojawiający się w chwili zero. Aby uzyskać odpowiedź na dowolne wymuszenie skokowe (rys. 2.13) można **przeskalować i przesunąć odpowiedź skokową** do punktu równowagi, wykorzystując wektory wartości generowane przez funkcję `step()`:

```
u0=1; du=2;          %parametry wymuszenia skokowego
x0=3;                %punkt równowagi dla u0
[y, t] = step(ObiektG3);
plot(t, x0+y*du);
```



Rys. 2.13. Parametry wymuszenia skokowego

W nowszych wersjach Matlaba można zdefiniować parametry wymuszenia skokowego (wartość początkową, wartość skoku) wykorzystywane przez funkcję `step()` – funkcja `stepDataOptions()`

```
u0=1; du=2;          %parametry wymuszenia skokowego
stepDataOptions();   %odczytanie parametrów
opcje = stepDataOptions('InputOffset',u0);
opcje = stepDataOptions('StepAmplitude',du);
step(ObiektG3);
```

Badania w rozdziale 3 wyjaśnią wielkość skoku wpływa na parametry przebiegu czasowego.

Wykresy wygenerowane przez funkcję `step()` i `impluse()` można formatować tylko poprzez menu kontekstowe. Aby przetwarzać wykresy (np. formatować) za pomocą funkcji, wykorzystuje się rysowanie wykresów za pomocą funkcji `plot()` na podstawie danych (wektorów wartości) wygenerowanych przez `step()` i `impluse()`. Alternatywą dla tego rozwiązania jest zastosowanie funkcji **`stepplot()`** i **`impluseplot()`**, które rysują wykres i zwracają ich identyfikatory (uchwyty). Można wówczas formatować wykresy za pomocą funkcji `setoptions()` i `getoptions()`.

2.2.3 Położenie biegunów i zer

Definicje i parametry. Bieguny i zera układu (2.1.1) są najbardziej pierwotną formą opisywania własności dynamicznych, ponieważ odwołują się do parametrów rozwiązania równania różniczkowego. Kluczowe znaczenia dla własności układu mają bieguny, ponieważ każdy wprowadza do przebiegu przejściowego swój składnik:

- bieguny rzeczywiste p:

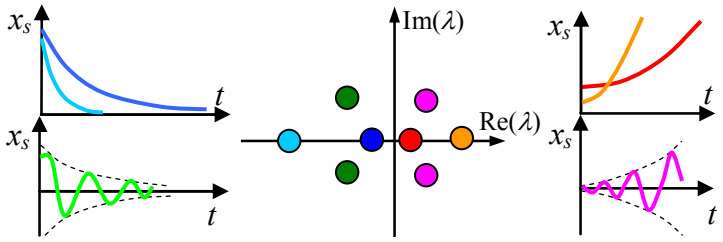
  Ae^{pt}

- bieguny zespolone $p_{1,2} = -\sigma \pm j\omega_r$

  $Ae^{-\sigma t}\sin(\omega_r t + \varphi_1)$

- bieguny wielokrotne, np. $p_1 = p_2$

  $(A + At)e^{p_1 t}$



Rys. 2.14. Położenie biegunów a składniki rozwiązania

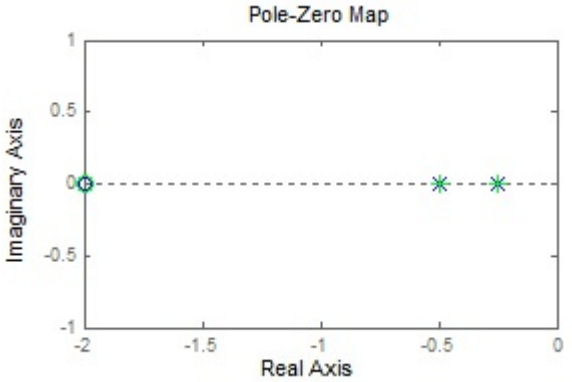
Odpowiedzi skokowe i impulsowe są sumą stałej składowej wymuszonej i składowej swobodnej, zawierającej składniki od wszystkich biegunów. Jednak analizę układu można ograniczyć do najbardziej znaczących biegunów, które decydują o:

- stabilności – układ jest stabilny gdy wszystkie bieguny maja ujemną część rzeczywistą,
- czasie zanikania – jeśli układ jest stabilny, to o czasie zanikania decyduje biegun najbliżej osi Im (wprowadza składnik, który najdłużej zanika),
- oscylacyjności – bieguny zespolone wprowadzają składnik oscylacyjny.

Zera układu mogą kompensować (osłabiać) wpływ biegunów.

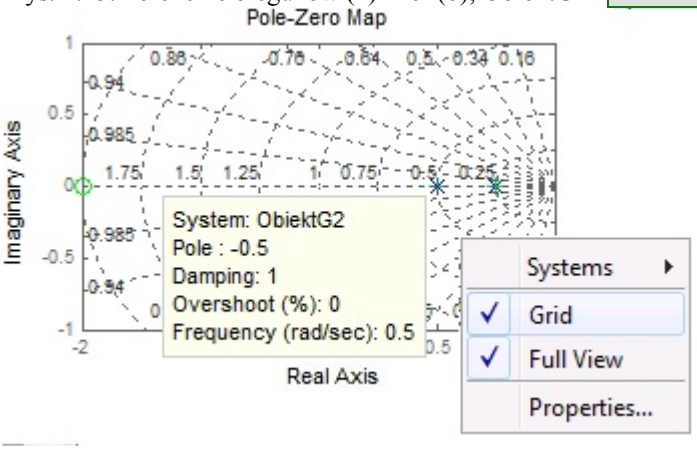
Obliczenia analityczne. Bieguny układu są pierwiastkami równania charakterystycznego, które można wyznaczyć na podstawie mianownika transmitancji lub macierzy stanu. Zera są pierwiastkami licznika transmitancji.

Badania symulacyjne. Podstawowa funkcja do graficznego przedstawienia biegunów i zer na płaszczyźnie zespolonej, to **`pzmap()`**. Za jej pomocą można narysować te parametry lub zapamiętać wektor z biegunami i wektor z zerami, do dalszego przetwarzania (np. do narysowania za pomocą `plot()`). Obie te możliwości zilustrowano na rys. 2.15, przedstawiającym mapę zer i biegunów modelu ObiektG2, który ma dwa bieguny -0,5 i -0,25 oraz jedno zero o wartości -2. Zgodnie z utrwaloną tradycją bieguny są oznaczane krzyżykami, a zera – kółkami.



Rys. 2.15. Położenie biegunów (x) i zer (o); ObiektG2

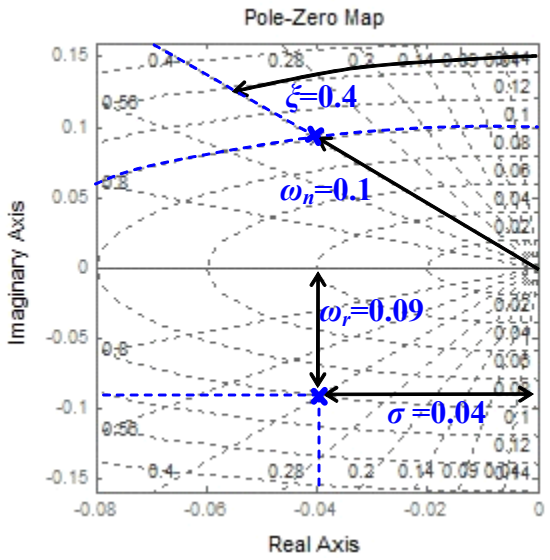
```
figure; set(gcf(), 'Color',[1,1,1]);

% bieguny 'x' i zera 'o'      (na dwa sposoby)
pzmap(ObiektG2);              %niebieskie
grid off

[P, Z] = pzmap(ObiektG2);
plot(real(P),imag(P), 'g+');  %zielone
plot(real(Z),imag(Z), 'go');  %zielone

%operacje formatowania ...
```



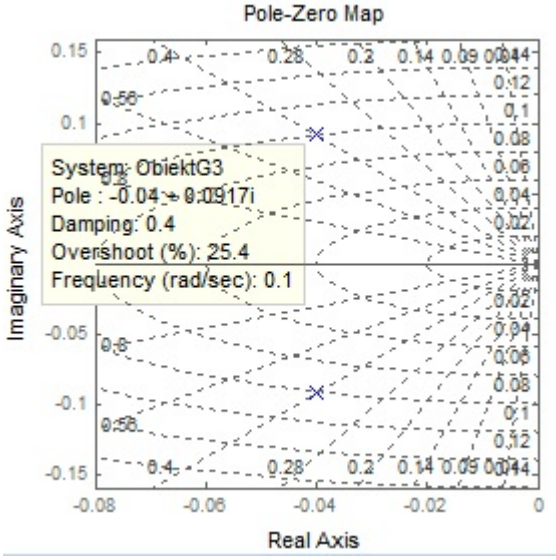
Rys. 2.16. Funkcjonalności `pzmap()`; ObiektG2

Funkcja `pzmap()` nie tylko rysuje bieguny i zera, ale dodaje standardowy tytuł i opis osi, a poprzez menu kontekstowe dostarcza takich funkcjonalności jak odczytywanie wartości biegunów i zer z wykresu (lewe menu) oraz formatowanie wykresu (prawe menu - Properties) - rys. 2.16. Możliwe jest też włączenie dodatkowej siatki, która ma zastosowanie głównie w przypadku układów o zespolonych biegunach.

Dodatkowa siatki sprawdza się szczególnie w przypadku układów oscylacyjnych - na rys. 2.17 przedstawiono interpretację dodatkowej siatki na przykładzie ObiektG3 o następujących parametrach: $p_{1,2}$= -0.04±j0.0917, $\xi = 0.4$, $\omega_n = 0.1$.

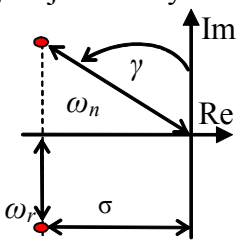


Rys. 2.17. Interpretacja dodatkowej siatki; ObiektG3



Rys. 2.18. Odczytywanie wartości z wykresu; ObiektG3

Interpretacja tej siatki wynika z parametrów członu oscylacyjnego (2-12) i relacji geometrycznych:



Rys. 2.19. Pierwiastki równania oscylacyjnego

$$\sqrt{\sigma^2+\omega_r{}^2}=\sqrt{(\xi\omega_n)^2+\left(\omega_n\sqrt{1-\xi^2}\right)^2}=\omega_n$$
$$\sin\gamma=\frac{\sigma}{\sqrt{\sigma^2+\omega_r^2}}=\frac{\xi\omega_n}{\sqrt{\xi^2\omega_n^2+\omega_n^2(1-\xi^2)}}=\xi \qquad (2\text{-}16)$$

To oznacza, że:
- siatka prostokątna służy do odczytywania wartości biegunów $-\sigma \pm j\omega_r$ (Pole),
- półproste wychodzące ze środka układu odpowiadają stałej wartości tłumienia ξ (Damping),
- półokręgi siatki odpowiadają stałym wartościom pulsacji ω_n (Frequency rad/sec).

Wartości parametrów, które można odczytać za pomocą tych siatek, można również uzyskać poprzez menu kontekstowe (lewe menu) związane z poszczególnymi biegunami (rys. 2.18).
Wszystkie parametry odczytywane z wykresów można uzyskać również za pomocą funkcji **`pole()`**, **`zero()`** i **`damp()`**. W Tab. 2-1 przedstawiono parametru układu ObiektG2 (p_1= -0.5, p_2= -0.25, z_{z1}= -2) uzyskane za pomocą tych funkcji i odczytane na wykresie (Rys. 2.16).

Tab. 2-5. Parametry układu ObiektG2 z funkcji i odczytane z menu kontekstowego (Rys. 2.16)

Parametr	Funkcja		Menu kontekstowe
p_1= -0.5 (T_1=2)	>> pole(ObiektG2) ans = -0.5000 -0.2500	>> damp(ObiektG2) Eigenvalue Damping Freq. (rad/s) -2.50e-001 1.00e+000 2.50e-001 -5.00e-001 1.00e+000 5.00e-001	System: ObiektG2 Pole: -0.5 Damping: 1 Overshoot (%): 0 Frequency (rad/sec): 0.5
p_2= -0.25 (T_2=4)			System: ObiektG2 Pole: -0.25 Damping: 1 Overshoot (%): 0 Frequency (rad/sec): 0.25
z_{z1}= -2 (T_{z1}=0.5)	>> zero(ObiektG2) ans = -2		System: ObiektG2 Pole: -2 Damping: 1 Overshoot (%): 0 Frequency (rad/sec): 2

W przypadku rzeczywistych biegunów i zer wartość tłumienia (Damping) zawsze jest podawana jako 1, a pulsacja (Frequency) jest obliczana jako odwrotność stałej czasowej.
W Tab. 2-1 przedstawiono parametru układu ObiektG3 (ξ = 0.4, ω_n = 0.1.) uzyskane za pomocą funkcji i odczytane na wykresie (Rys. 2.17)

Tab. 2-6. Parametry układu ObiektG3 z funkcji i odczytane z menu kontekstowego (Rys. 2.17)

Parametr	Funkcja		Menu kontekstowe
p_1=	>> pole(ObiektG3)	>> damp(ObiektG3)	System: ObiektG2

-0.04+j0.0917	ans = -0.040+0.09117i -0.040-0.09117i	Eigenvalue Damping Freq. (rad/s) -4.00e-002 + 9.17e-002i 4.00e-001 1.00e-001 -4.00e-002 - 9.17e-002i 4.00e-001 1.00e-001	Pole: -0.04+0.0917i Damping: 0.4 Overshoot (%): 25.4 Frequency (rad/sec): 0.1
p_1= -0.04-j0.0917			System: ObiektG2 Pole: -0.04-0.0917i Damping: 0.4 Overshoot (%): 25.4 Frequency (rad/sec): 0.1

2.2.4 Charakterystyki częstotliwościowe i analiza w dziedzinie częstotliwości (Frequency-domain analysis) [3/100,314]

Definicje i parametry. Charakterystyki częstotliwościowe opisują odpowiedź układu na wymuszenie sinusoidalne, czyli tak zwaną odpowiedź częstotliwościową (frequency response). W przypadku układów liniowych jeśli funkcją wymuszającą jest przebieg sinusoidalny, to na wyjściu w stanie ustalonym też będzie przebieg sinusoidalny o takiej samej częstotliwości, jedynie wzmocniony i przesunięty w fazie. To wzmocnienie i przesunięcie zależy od częstotliwości i jest opisywane za pomocą transmitancji widmowej, którą można przedstawić w różnych postaciach:

$$G(j\omega) = \frac{L(j\omega)}{M(j\omega)} = P(\omega) + jQ(\omega) = A(\omega)e^{j\varphi(\omega)} \tag{2-17}$$

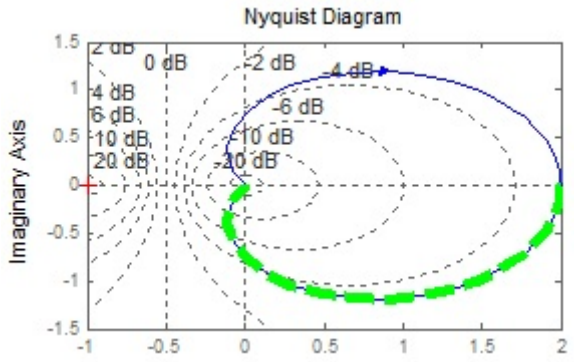
którym odpowiadają różne typy charakterystyk częstotliwościowych (tab. 2-7).

Tab. 2-7 Charakterystyki częstotliwościowe

Nazwa charakterystyki	Zależność	Funkcja Matlab
charakterystyka części rzeczywistej	$P(\omega) = \mathrm{Re}(G(j\omega))$	
charakterystyka części urojonej	$Q(\omega) = \mathrm{Im}(G(j\omega))$	
charakterystyka amplitudowo-fazowa (Nyquista)	$P(Q)$	`nyquist()`
charakterystyka amplitudowa	$A(\omega)$	
charakterystyka fazowa	$\varphi(\omega)$	
logarytmiczna charakterystyka modułu (Bodego)	$M(\omega) = 20\lg A(\omega)$	`bode()`,`nichols()`
logarytmiczna charakterystyka fazy	$\varphi(\omega)$	`bode()`
logarytmiczna charakterystyka amplitudowo-fazowa	$M(\varphi)$	

O rysowniu asymptot Bodego

Generowanie. Podstawowe funkcje do rysowania charakterystyk czasowych, to `nyquist()` i `bode()`. Pozwalają one narysować wykres lub zapamiętać dane (wektory wartości) do dalszego przetwarzania (np. do narysowania za pomocą `plot()`), co zilustrowano na rys. 2.20 i rys. 2.21, przedstawiającym wykresy dla zdefiniowanego wcześniej modelu ObiektG2.



Rys. 2.20. Ch.Nyquista układu; ObiektG2

```
figure; set(gcf(), 'Color',[1,1,1]); hold on

%ch.Nyquista (na dwa sposoby)
nyquist(ObiektG2);                %niebieska linia
[p, q, wn] = nyquist(ObiektG2);
p = squeeze(p); q = squeeze(q);
plot(p, q, 'g--');                %zielona linia
grid on

%operacje formatowania ...
```

Charakterystyka amplitudowo-fazowa jest wykreślana również dla ujemnych częstotliwości, co nie ma wprawdzie sensu fizycznego, ale pomaga przy badaniu stabilności, podobnie jak oznaczenie charakterystycznego punktu (-1, j0) (gdzie będzie o kr.Nyquista?). Charakterystyki dla dodatnich i ujemnych częstotliwości są symetryczne względem osi poziomej.

Rys. 2.21. Ch.Bodego układu; ObiektG2

```
figure; set(gcf(), 'Color',[1,1,1]); hold on

%ch.czestotliwosciowa (na dwa sposoby)

%operacje formatowania ...
```

Funkcja `nyquist()` zwraca ....

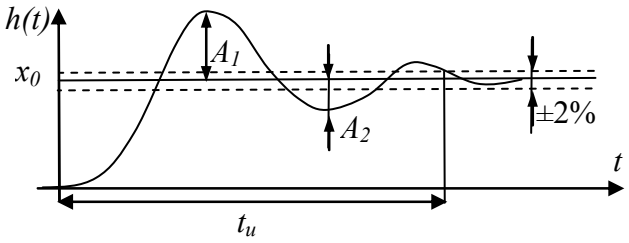
Oppelt/76: a) ch-ka dla ujemnych częstotliwości, b) ujemna ch-ka; c) odwrócona ch-ka?

Wykresy wygenerowane przez funkcję `nyquist()` i `bode()` można formatować tylko poprzez menu kontekstowe. Aby przetwarzać wykresy (np. formatować) za pomocą funkcji, wykorzystuje się rysowanie wykresów za pomocą funkcji `plot()` na podstawie danych (wektorów wartości)

wygenerowanych przez `nyquist()` i `bode()`. Alternatywą dla tego rozwiązania jest zastosowanie funkcji **`nyquistplot()`** i **`bodeeplot()`**, które rysują wykres i zwracają ich identyfikatory (uchwyty). Można wówczas formatować wykresy za pomocą funkcji `setoptions()` i `getoptions()`.

2.2.5 Parametry - zestawienie?

- stabilności
- czasie zanikania
- oscylacyjności

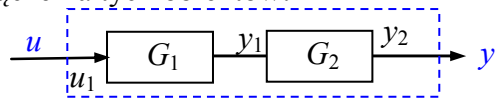


Rys. 2.22. Parametry odpowiedzi skokowej

2.3. Układy złożone – schematy blokowe

2.3.1 Szeregowe łączenie obiektów

Na podstawie transmitancji obiektów bardzo łatwo wyznacza się transmitancję szeregowego połączenia tych obiektów:



Rys. 2.23. Szeregowe połączenie transmitancji

$$\frac{y(s)}{u(s)} = G_{sz}(s) = G_1(s)G_2(s) \qquad (2\text{-}18)$$

W Matlabie można również wykonać mnożenie transmitancji lub zastosować funkcję `series()`:

```
G1= ...; G2= ...;
Gsz= G1* G2;
```

```
G1= ...; G2= ...;
Gsz= series(G1, G2);
```

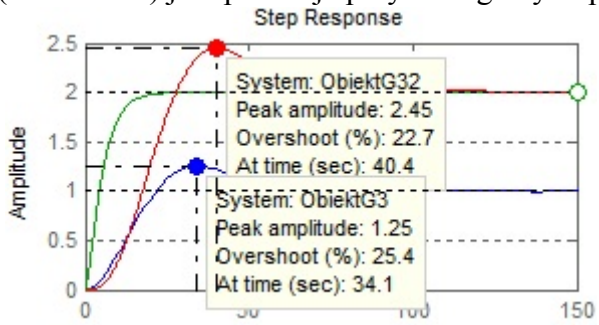
Przykład: Obiekt złożony G32. Iloczyn członu oscylacyjnego G3 i członu inercyjnego G2

```
>>ObiektG3
    Transfer function:
          0.01
    ------------------
    s^2 + 0.08 s + 0.01
```

```
>>ObiektG2
    Transfer function:
        s + 2
    ---------------
    8 s^2 + 6 s + 1
```

```
>>ObiektG32
    Transfer function:
                0.01 s + 0.02
    ---------------------------------------------
    8 s^4 + 6.64 s^3 + 1.56 s^2 + 0.14 s + 0.01
```

Dla prostych układów (ObiektG3, ObiektG2) można dość łatwo i dokładnie obliczyć wartości parametrów odpowiedzi czasowych (p.2.2.2). Możliwość odczytywania parametrów z wykresu bardzo ułatwia analizę złożonych układów. Dla przykładu na rys. 2.24 porównano odpowiedzi członu inercyjnego (ObiektG2) i członu oscylacyjnego (ObiektG3) z odpowiedzią złożonego układu (ObiektG32) jaki powstaje przy szeregowym połączeniu tych prostych członów.

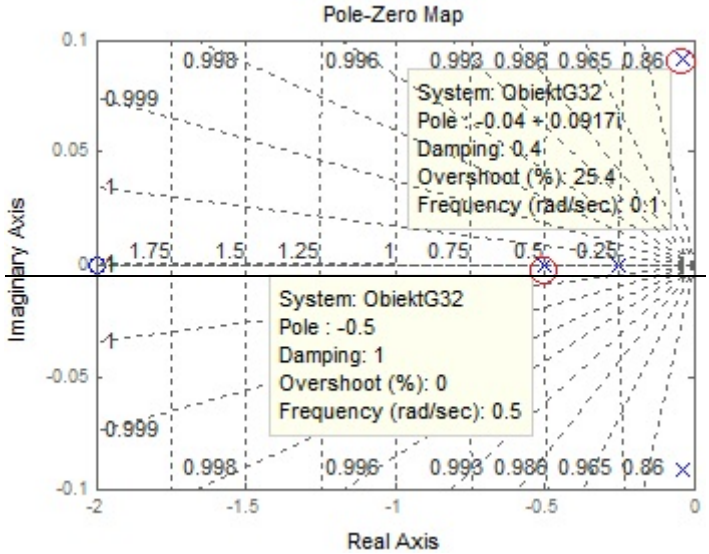


Rys. 2.24. Odpowiedź złożonego układu ObiektG32

```
figure; set(gcf(), 'Color',[1,1,1]); hold on;

step(ObiektG2,'g',ObiektG3,'b',ObiektG32,'r')

%operacje formatowania ...
```

Oscylacje w odpowiedzi układu złożonego wynikają z obecności członu oscylacyjnego, ale porównanie wartości w punkcie x_p pokazuje, że parametry tych oscylacji są różne (rozwiązanie w załączniku?).

Wygenerowanie mapy biegunów i zer złożonego układu, takiego jak ObiektG32, przedstawia wszystkie bieguny i zera układu oraz pozwala odczytać parametry odpowiadające poszczególnym biegunom (rys. 2.25).



Rys. 2.25. Bieguny złożonego układu ObiektG32

Są to te same wartości, które dla poszczególnych biegunów układu zwraca funkcja **`damp()`**:

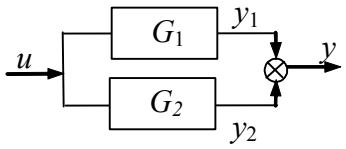
```
>> damp(ObiektG32)
          Eigenvalue              Damping       Freq. (rad/s)
```

```
-4.00e-002 + 9.17e-002i   4.00e-001   1.00e-001
-4.00e-002 - 9.17e-002i   4.00e-001   1.00e-001
-2.50e-001                1.00e+000   2.50e-001
-5.00e-001                1.00e+000   5.00e-001
```

Dodatkowo na wykresie podawana jest też wartość przeregulowania (Overshoot %) odpowiadająca danemu biegunowi, czyli wartość przeregulowania, jakie występuje w odpowiedzi podstawowego członu dynamiki z danym biegunem (porównaj parametry na rys. 2.25 i rys. 2.24). Mapa nie udostępnia parametrów odpowiedzi złożonego układu, które można było odczytać rys. 2.24.
Ch. częstotliwościowe ...

2.3.2 Równoległe łączenie obiektów

Wyznaczenie transmitancji równoległego połączenia obiektów na podstawie transmitancji jest również bardzo proste



$$\frac{y(s)}{u(s)} = G_{ro}(s) = G_1(s) + G_2(s) \qquad (2\text{-}19)$$

Rys. 2.26. Równoległe połączenie transmitancji

W Matlabie można wykonać dodawanie transmitancji lub zastosować funkcję `parallel()`:

```
G1= ...; G2= ...;
Gro= G1 + G2;
```

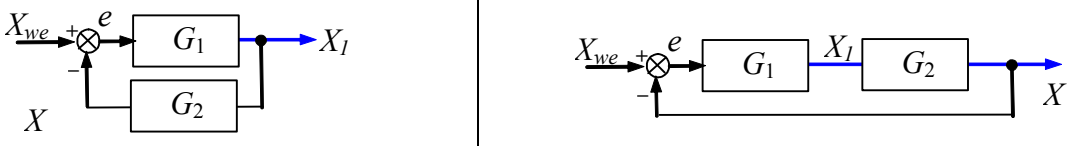
```
G1= ...; G2= ...;
Gro= parallel(G1,G2);
```

.....

W szczególności równoległe i – na przykładzie regulatora PID (różne struktury)
Definicje regulatora s=tf('s'); reg = Kp + KTi/s

2.3.3 Układ z ujemnym sprzężeniem zwrotnym (układ korekcji i układ regulacji)

Podstawową strukturą występującą w układach sterowania jest układ z ujemnym sprzężeniem zwrotnym (Rys. 2.27).



Rys. 2.27. Układy ujemnym sprzężeniem zwrotnym

Układ można opisać za pomocą trzech transmitancji:

$$X_1 = \frac{G_1}{1+G_1G_2}X_{we}, \quad X = \frac{G_1G_2}{1+G_1G_2}X_{we}, \quad e = \frac{1}{1+G_1G_2}X_{we}. \qquad (2\text{-}20)$$

Definicję układów ze sprzężeniem zwrotnym można wykonać zapisują odpowiednie wyrażenie matematyczne (2-20) lub przy użyciu funkcji `feedback()`:

```
G1 = ...; G2 = ...;
wyj_X1 = G1 / (1+G1*G2);
wyj_X  = G1*G2 / (1+G1*G2);
wyj_e  = 1 / (1+G1*G2);
```

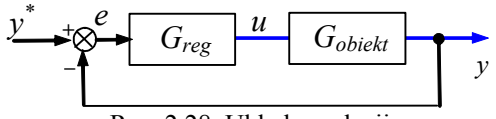
```
G1 = ...; G2 = ...;
wyj_X1 = feedback(G1, G2);
wyj_X  = G2 * feedback(G1, G2);
wyj_e =
```

Pokazać inne sposoby definicji (ręczne L i M) i konsekwencje (nieuproszczonee bieguny)???

Funkcja `feedback()` upraszcza nadmiarowe bieguny i zera

W zależności od interpretacji transmitancji G_1 i G_2 rozróżnia się:
- układ regulacji – gdy G_1 to urządzenie sterujące (regulator), a G_2 to obiekt,
- układ korekcji - gdy G_1 to obiekt, a G_2 to urządzenie sterujące (korektor).

W dalszej części podręcznika analizowany będzie przede wszystkim układ regulacji (Rys. 2.28).



Rys. 2.28. Układ regulacji

$$G_o = G_{reg}G_{obiekt} \qquad (2\text{-}21)$$

$$G_z = \frac{y}{y^*} = \frac{G_{reg}G_{obiekt}}{1+G_{reg}G_{obiekt}} \qquad (2\text{-}22)$$

Do badania i projektowania układów regulacji wykorzystywane są głównie transmitancje układu otwartego G_o (2-21) i układu zamkniętego G_z (2-22). Do definicji tych transmitancji stosowane będą następujące wyrażenia:

```
uklad_Gotw  = Greg * Gobiekt;
uklad_Gz = feedback(Greg * Gobiekt, 1);
```

Podstawowe badania układu otwartego i zamkniętego

`step(uklad_Gz)` – odpowiedź skokowa układu zamkniętego
`pole(uklad_Gz)` – bieguny układu zamkniętego
`rlocus(uklad_Go)` – linie pierwistkowe

Charakterystyki układu otwartego do badania stabilności układu zamkniętego:

`nyquist(uklad_Gotw)` – charakterystyka Nyquista układu otwartego
`bode(uklad_Gotw)` – charakterystyka Bodego układu otwartego

Kryterium Nuqusta (p.Oppelt/257-260, 272)
Jeśli układ otwarty jest stabilny i jeśli punkt charakterystyczny (-1,j0) leży w obszarze po lewej stronie obserwatora poruszającego się wzdłuż charakterystyki amplitudowo-fazowej w kierunku wzrastających częstotliwości, to układ ze sprzężeniem zwrotnym też jest stabilny.
Przy zawiłym przebiegu charakterystyki częstotliwościowej rozróżnienie obszaru na „lewo" i „prawo" jest łatwiejsze gdy zostanie wykreślona charakterystyka dla ujemnych częstotliwości, która jest symetryczna względem osi części rzeczywistej *P*. W ten sposób realizuje to funkcja `nyquist()`.
o funkcji nichols
Wspomnieć o liniach pierwiastakowych??? (Oppelt/308)

2.3.4 Łączenie układów MIMO

2.3.5 Układy liniowe z opóźnieniami

http://www.mathworks.com/help/control/ug/time-delays-in-linear-systems.html

2.4. Interfejs użytkownika do analizy obiektów - ltiview

2.4.1 Analizowanie i porównywanie modeli SISO

Załóżmy, że istnieją dwa obiekty LTI, zdefiniowane na podstawie transmitancji G_0 i G_1:zamienić na ObiektG1 i ObiektG2

$$G_0(s) = \frac{k}{(T_1 s + 1)(T_2 s + 1)}$$

$$G_1(s) = \frac{k(T_{z1} s + 1)}{(T_1 s + 1)(T_2 s + 1)}$$

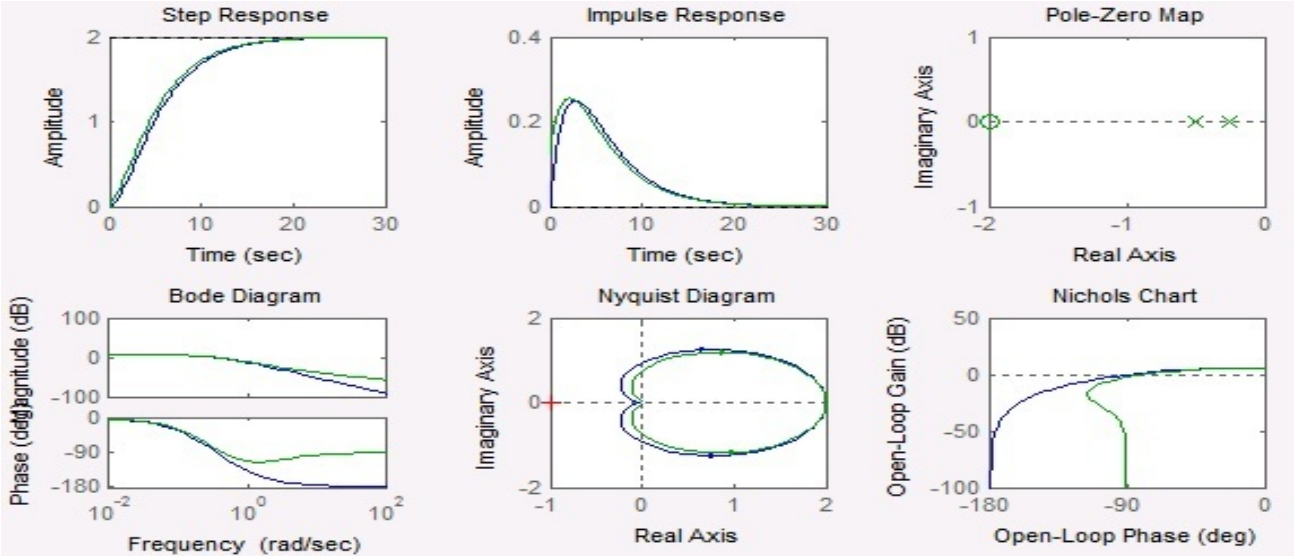
```
k=2;    T1=2;    T2=4;    Tz1=T1/4;
s=tf('s');
obiekt0= k/((T1*s+1)*(T2*s+1));
obiekt1= k*(Tz1*s+1)/((T1*s+1)*(T2*s+1));
```

Obiekty należy porównać na podstawie charakterystyk czasowych i częstotliwościowych. Poniżej przedstawiono dwa sposoby realizacji zadania.

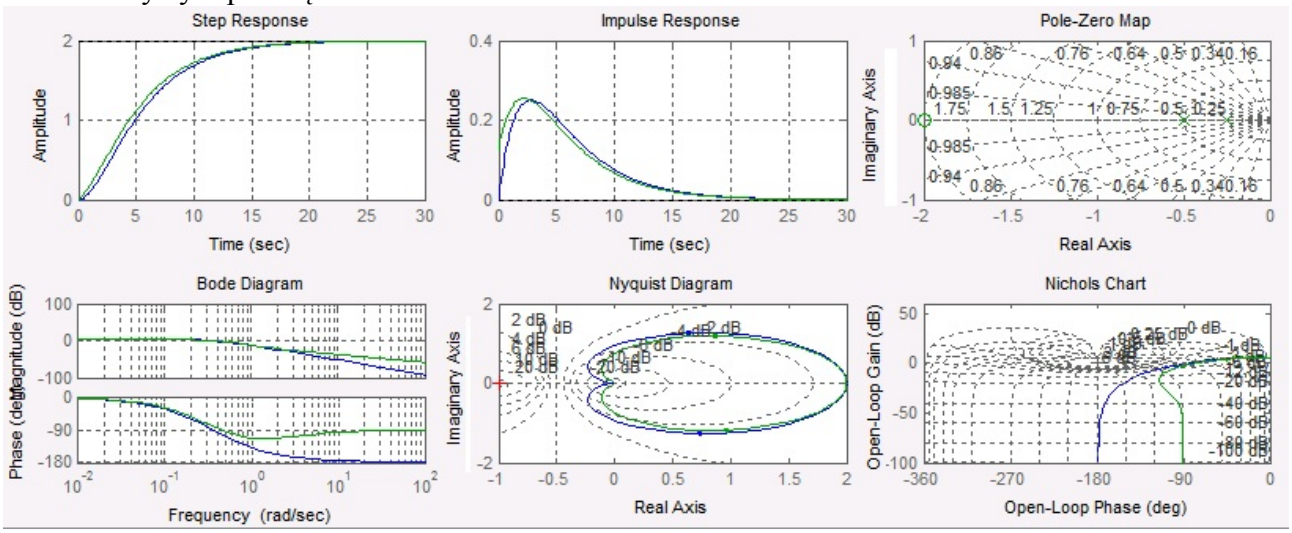
1. Interfejs użytkownika ltiview – pozwala wybierać charakterystyki a następnie automatycznie uruchamia symulacje i wykonuje charakterystyki (niebieskie - obiekt0, zielone – obiekt1):

```
ltiview({'step','impulse','pzmap', 'nyquist','bode','nichols'},obiekt0,obiekt1);
```

Użytkownik może konfigurować charakterystyki przy pomocy menu okna , np.
- charakterystyki bez dodatkowej siatki



- charakterystyki po włączeniu siatek:



2. Skrypt – wymaga skonfigurowania okna i wpisania wybranych charakterystyk, ale może pełnić rolę dokumentacji badań (pozwala zawsze powtórzyć/rozszerzyć badania)

```
subplot(321); step(obiekt0,obiekt1); grid on
subplot(322); impulse(obiekt0,obiekt1); grid on
subplot(323); pzmap(obiekt0,obiekt1); grid on
subplot(324); bode(obiekt0,obiekt1); grid on
subplot(325); nyquist(obiekt0,obiekt1); grid on
subplot(326); nichols(obiekt0,obiekt1); grid on
```

2.4.2 Analizowanie modelu MIMO

Warto nadać własne nazwy

```
obiektS= ss(A,B,C,D,'InputName',['we1';'we2'],'StateName',['wy1
';'wy2'],'OutputName',['wy1 ';'wy2']);
```

Jak przeskalować i przesunąć wyniki

Patrz Przykłady DoWspomagania

-

2.5. Analiza liniowa z wykorzystaniem interfejsu Matlab-Simulink

2.5.1 Definicje na schemacie

Na przykładzie modeli **ObiektG1**, **ObiektG2**, **ObiektM1**, **ObiektM2**

ObiektG1 i **ObiektG2** to transmitancje zdefiniowane w formie (2-2)÷(2-4) o parametrach zgodnie z (tab. 2-1).

1) rzędu pierwszego

$$a_1\dot{x}(t) + a_0 x(t) = b_0 u(t)$$

$$G_1(s) = \frac{b_0}{a_1 s + a_0}$$

$$G_1(s) = \frac{k_1}{s - p_1}$$

$$G_1(s) = \frac{k}{T_1 s + 1}$$

2) rzędu drugiego

$$a_2\ddot{x}(t) + a_1\dot{x}(t) + a_0 x(t) = b_1\dot{u}(t) + b_0 u(t) \quad (2\text{-}23)$$

$$G_2(s) = \frac{b_1 s + b_0}{a_2 s^2 + a_1 s + a_0} \quad (2\text{-}24)$$

$$G_2(s) = \frac{k_1(s - z_1)}{(s - p_1)(s - p_2)} \quad (2\text{-}25)$$

$$G_2(s) = \frac{k}{(T_1 s + 1)(T_2 s + 1)} \cdot (T_{z1} s + 1) \quad (2\text{-}26)$$

Definicje za pomocą bloku ObiektLTI

Definicje za pomocą bloków transmitancji
- na schemacie używane są zmienne a ich wartości są definiowane w skrypcie
- na schemacie wpisywane są wprost wartości parametrów
- uruchamianie symulacji i rysowanie wykresów w skrypcie (wsadowo) – sim, plot
- uruchamiane symulacji i wizualizacja wyników na schemacie –

Definicje za pomocą bloków równań stanu
- jw

Schematy złożone

2.5.2 Charakterystyki

Charakterystyki czasowe – scope, To Workspace, czas
Charakterystyki czasowe

2.5.3 Interfejs użytkownika do analizy obiektów - Linear Analysis

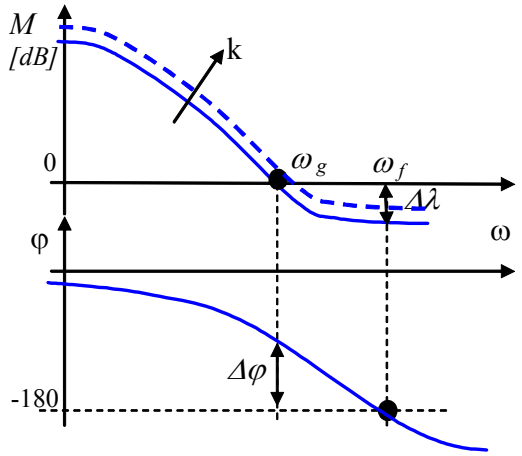
3. Badania podstawowych obiektów (członów) dynamiki...

3.1. Badania

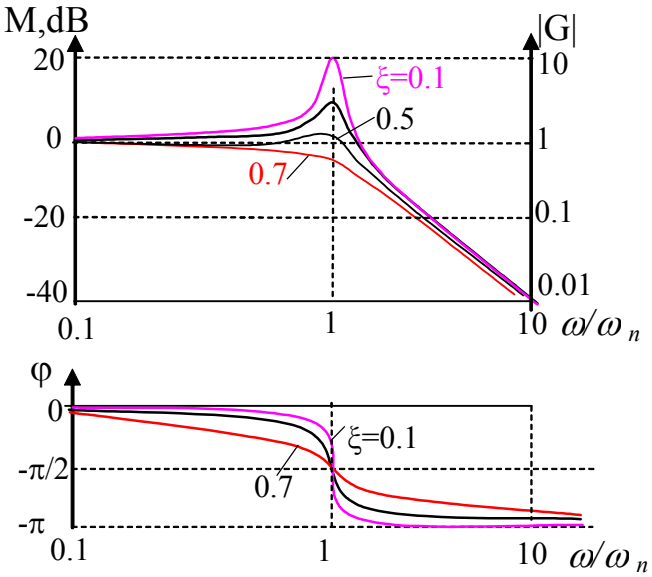
1) Człony inercyjne:
   a) człon inercyjny o różnych stałych czasowych – styczna
   b) to samo wzmocnienie, różne stałe czasowe – porównywalne i znacznie różne

Rząd układu np. m.Sterjca

Wpływ wzmocnienia na ch.Bodego



2) Człon oscylacyjny



- szczyt rezonansowy na ch.Bodego

Zera transmitancji (wpływ zer układu na własności dynamiki)

3) Układ z dodatnim i ujemnych zerem

Opóźnienie transportowe i aproksymacja Pade

4) Złożenie dwóch członów oscylacyjnych – to samo tłumienie i różne pulsacje
5) Układy z całkowaniem – inercja z całkowaniem
6) Wpływ różniczkowania
7)

Układy minimalno-fazowe – p.Oppelt/77 (przypis)

4. Badania prostych układów regulacji[1]

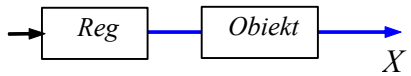
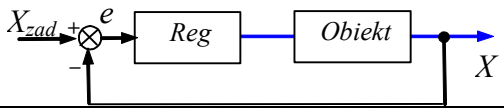
4.1. Opis badanych przypadków

4.1.1 Modele obiektów, regulatorów i badanych układów

Prezentowane układy regulacji (UR) składają się z prostego obiektu i regulatora P lub PI:

Obiekt G_{obiekt}	$G_{reg} = K_p$	$G_{reg} = K_p\left(1+\frac{1}{T_i s}\right) = K_p\left(1+\frac{K_i}{s}\right)$
oscylacyjny (ξ<1) $\frac{k}{s^2+2\xi\omega s+\omega^2}$	k=2; ξ=0.4; ω_n=2; (1b) K_p=2;	k=2; ξ=0.4; ω_n=2; (1a) K_p=2; T_i=2
inercyjny 2.rzędu (ξ>1) $\frac{k}{s^2+2\xi\omega s+\omega^2}$	k=2; ξ=1.4; ω_n=2; (2b) K_p=1;	k=2; ξ=1.4; ω_n=2; (2a) K_p=1; T_i=2
	k=2; ξ=1.4; ω_n=2; (2d) K_p=20;	k=2; ξ=1.4; ω_n=2 ; (2c) K_p=20;T_i=2
oscylacyjny + inercyjny (ξ<1) $\frac{k}{s^2+2\xi\omega s+\omega^2}\frac{1}{Ts+1}$,	k=2; ξ=0.4; ω_n=2; T=1 (4b) K_p=2;	k=2; ξ=0.4; ω_n=2; T=1; (4a) K_p=2; T_i=2
inercyjny 3.rzędu (ξ>1) $\frac{k}{s^2+2\xi\omega s+\omega^2}\frac{1}{Ts+1}$,	k=2; ξ=1.4; ω_n=2; T=1; (5b) K_p=1;	k=2; ξ=1.4; ω_n=2; T=1; (5a) K_p=1; T_i=2
	k=2; ξ=1.4; ω_n=2; T=1 (5d) K_p=20;	k=2; ξ=1.4; ω_n=2; T=1 (5c) K_p=20;T_i=2
$\frac{k}{Ts+1}e^{-sT_o}$	k=2; T=2; T_o=1; (6b)	

Własności układów będą porównywane analitycznie oraz na podstawie odpowiedzi skokowych, położenia biegunów i charakterystyk Bodego dla:

układu otwartego	układu zamkniętego
	
$G_o = G_{reg}G_{obiekt} = \frac{L_{reg}L_{obiekt}}{M_{reg}M_{obiekt}}$	$G_z = \frac{X}{X_{zad}} = \frac{G_{reg}G_{obiekt}}{1+G_{reg}G_{obiekt}} = \frac{L_{reg}L_{obiekt}}{M_{reg}M_{obiekt}+L_{reg}L_{obiekt}}$ $G_e = \frac{e}{X_{zad}} = \frac{1}{1+G_{reg}G_{obiekt}} = \frac{M_{reg}M_{obiekt}}{M_{reg}M_{obiekt}+L_{reg}L_{obiekt}}$

Analiza wzorów opisujących badane układy regulacji pozwoli wyprowadzić ogólne wnioski (własności). Zakłada się, że badane obiekty są stabilne, a wzmocnienia obiektu i regulatora mają wartości dodatnie (k>0, K_p>0). Analiza wykresów potwierdzi te własności dla konkretnych ...

[1] patrz skrypt testy_ObiektReg

Definicję każdego z obiektów i układów regulacji, odczyt podstawowych parametrów oraz wygenerowanie wykresów można przeprowadzić za pomocą prostego skryptu, wykorzystującego funkcje tf, feedback, dcgain, damp, step, pzmap, bode, rlocus, np.:

```
s=tf('s');
k=2;ksi=0.4; w=2, T=0;
Obiekt = k/(s*s+2*ksi*w*s+w*w);
Kp=2; Ti=2; Ki=1/Ti;
Reg = Kp*(1+Ki/s);
Go = Reg*Obiekt;
Gz = feedback(Go,1);

dcgain(Obiekt), damp(Obiekt)
dcgain(Go), damp(Go)
dcgain(Gz), damp(Gz)

figure, hold on, figure, step(Gz,'b-', Obiekt,'g--'); title('Gz (-), Obiekt (--)');
figure, hold on, rlocus(Go), plot(real(pole(Gz)),imag(pole(Gz)),'ms');
title('linie p. i bieguny Gz');
figure, hold on, pzmap(Go); title('Go');
figure, hold on, pzmap(Gz); title('Gz');
figure, hold on, bode(Go); title('Go');
figure, hold on, bode(Gz); title('Gz');
```

4.2. Obiekt 2.rzędu (oscylacyjny lub inercyjny) i regulator P

4.2.1 Analiza

$$G_{obiekt} = \frac{k}{s^2 + 2\xi\omega_n s + \omega_n^2},\quad G_{reg} = \frac{K_P}{1},$$

$$G_o = \frac{k \cdot K_p}{s^2 + 2\xi\omega_n s + \omega_n^2} \tag{4-1}$$

$$G_z = \frac{kK_p}{s^2 + 2\xi\omega_n s + \omega_n^2 + kK_p} = \frac{kK_p}{s^2 + 2\xi_R\omega_R s + \omega_R^2} \tag{4-2}$$

gdzie: $\omega_R = \sqrt{\omega_n^2 + kK_p}$, $\quad 2\xi_R\omega_R = 2\xi\omega_n \quad \rightarrow \quad \xi_R = \xi\dfrac{\omega_n}{\sqrt{\omega_n^2 + kK_p}}$.

	G_{obiekt}	G_o	G_z
$k_{ukł}$	$k_{ukł} = \dfrac{k}{\omega_n^2}$	$k_{ukł} = \dfrac{kK_p}{\omega_n^2}$	$k_{ukł} = \dfrac{kK_p}{\omega_n^2 + kK_p}$
Bieguny: $p_{1,2}$	$-\xi\omega_n \pm \omega_n\sqrt{\xi^2 - 1}$	$-\xi\omega_n \pm \omega_n\sqrt{\xi^2 - 1}$	$-\xi\omega_n \pm \xi\sqrt{-kK_p}$

Wnioski (ξ>0, k>0, K_p>0).:

- Układ zamknięty ma tyle samo biegunów i zer, co układ otwarty,
- Jeśli obiekt jest stabilny, to układ z regulatorem P też będzie stabilny (ξ>0 → ξ_R>0),
- Tłumienie układu regulacji jest zawsze mniejsze obiektu: $\xi_R < \xi$,
- Jeśli obiekt jest oscylacyjny (ξ<1), to układ z regulatorem P też będzie oscylacyjny ,
- Jeśli obiekt jest inercyjny, to układ z regulatorem P może być oscylacyjny (gdy duże kK_p),
- Pulsacja UR jest zawsze wyższa niż pulsacja obiektu: $\omega_R > \omega_n$,
- Im większe K_p, tym mniejsze tłumienie ξ_R i wyższa częstotliwość oscylacji ω_R,
- Wzmocnienie układu zamkniętego jest mniejsze wzmocnienie układu otwartego (gdy K_p>1)
- Jeśli obiekt jest oscylacyjny (ξ<1), to bieguny układu otwartego i zamkniętego mają taką samą część rzeczywistą
- Odpowiedź skokowa stabilnego UR osiąga stan ustalony:

$$\lim_{t\to\infty} x(t) = \lim_{s\to 0} sG_z(s)u(s) = \lim_{s\to 0} \frac{kK_p}{s^2 + 2\xi\omega_n s + \omega_n^2 + kK_p} = \frac{kK_p}{\omega_n^2 + kK_p} \tag{4-3}$$

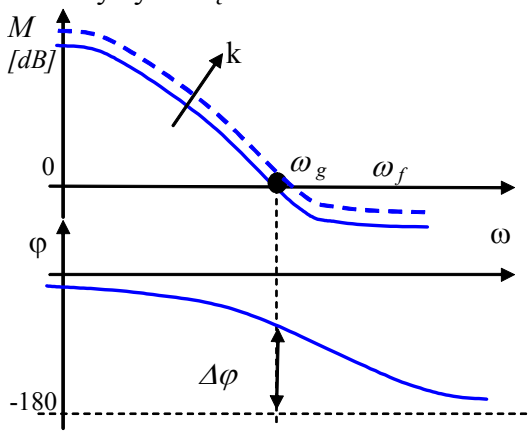
ale to nie znaczy, że jest uchyb regulacji jest zerowy, co można wyznaczyć na podstawie:

$$G_e = \frac{s^2 + 2\xi\omega_n s + \omega_n^2}{s^2 + 2\xi\omega_n s + \omega_n^2 + kK_p} \tag{4-4}$$

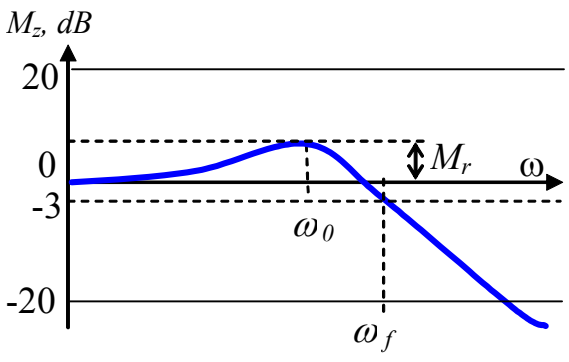
Jeśli regulator P to uchyb jest niezerowy:

$$\lim_{t\to\infty} e(t) = \lim_{s\to 0} sG_e(s)u(s) = \lim_{s\to 0} \frac{s^2 + 2\xi\omega_n s + \omega_n^2}{s^2 + 2\xi\omega_n s + \omega_n^2 + kK_p} = \frac{\omega_n^2}{\omega_n^2 + kK_p} \tag{4-5}$$

Charakterystyki częstotliwościowe – ilustracja do ....



Rys. 4.1. Wzmocnienie a stabilność



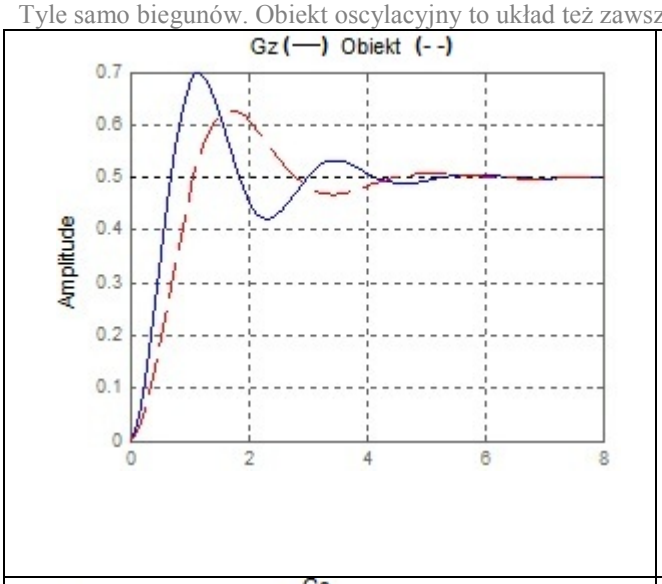
Rys. 4.2. Wzmocnienie a pasmo przenoszenia (szybkość)

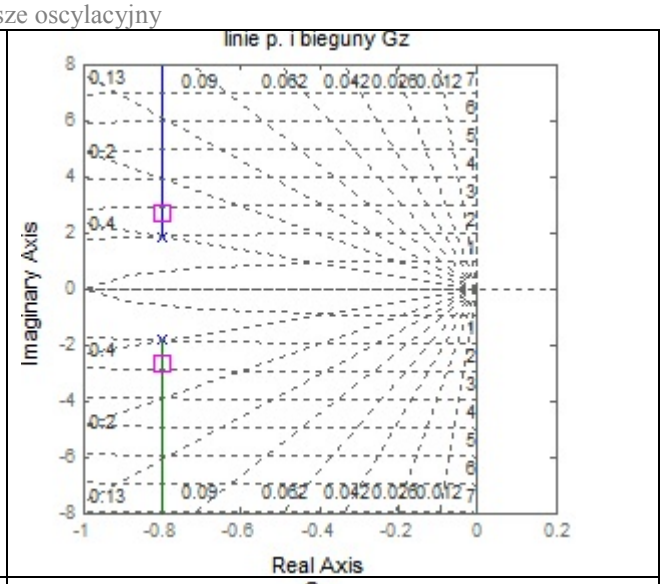
4.2.2 Człon oscylacyjny i regulator P

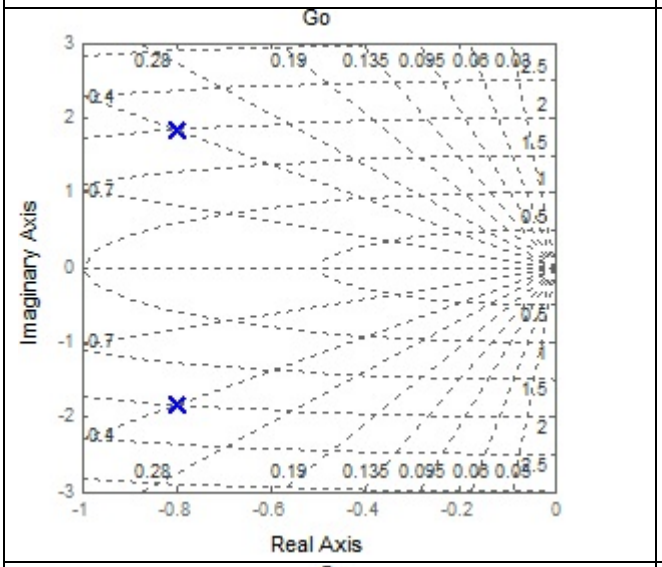
1a) k=2; **ksi=0.4**; w=2; Kp=2; (małe Kp);

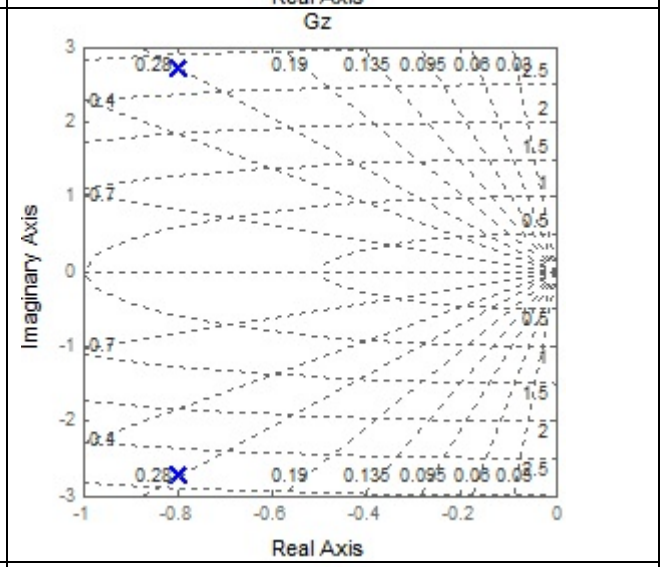
	dcgain	Eigenvalue	Damping	Freq. (rad/s)
Obiekt	0.5	-0.8+j1.83; -0.8-j1.83	0.4	2
Go	1	-0.8+j1.83; -0.8-j1.83	0.4	2
Gz	0.5	-0.8+j2.71; -0.8-j2.71	0.283	2.83

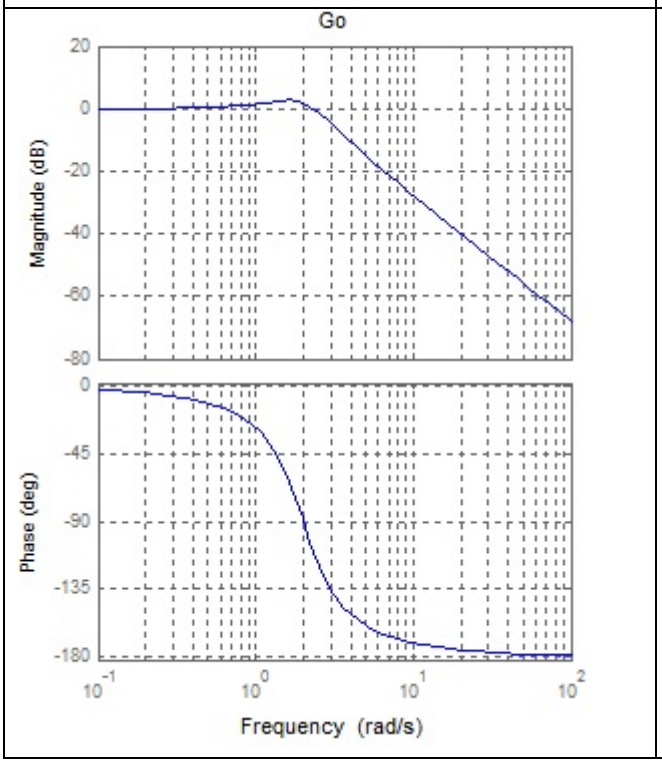
Tyle samo biegunów. Obiekt oscylacyjny to układ też zawsze oscylacyjny

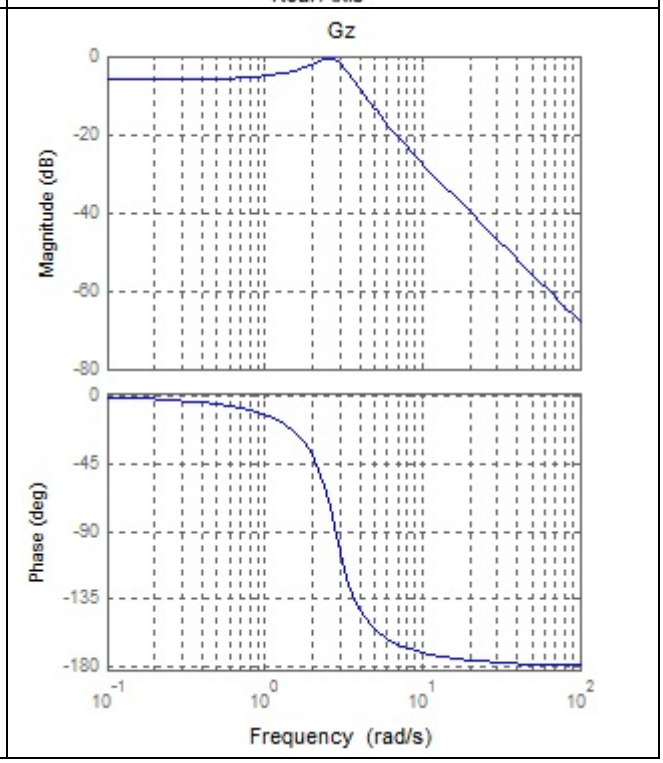










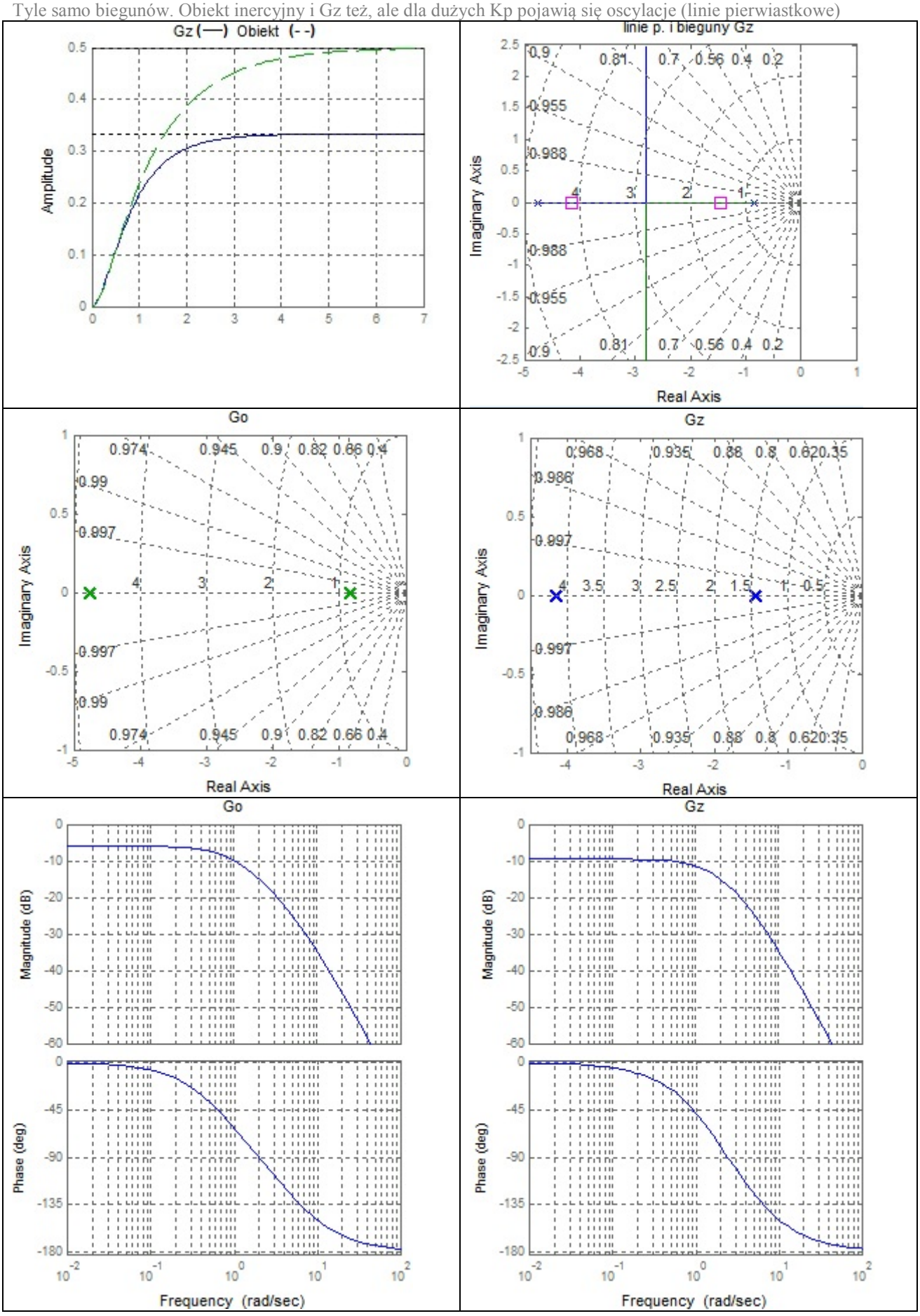


4.2.3 Człon inercyjny 2.rzędu i regulator P

2a) k=2; **ksi=1.4;** w=2; %Kp=1; (małe Kp)

	dcgain	Eigenvalue	Damping	Freq. (rad/s)
Obiekt	0.5	-0.84; -4.76	>1	0.84; 4.76
Go	0.5	-0.84; -4.76	>1	0.84; 4.76
Gz	0.333	-1.44; -4.16	>1	1.44; 4.16

Tyle samo biegunów. Obiekt inercyjny i Gz też, ale dla dużych Kp pojawią się oscylacje (linie pierwiastkowe)

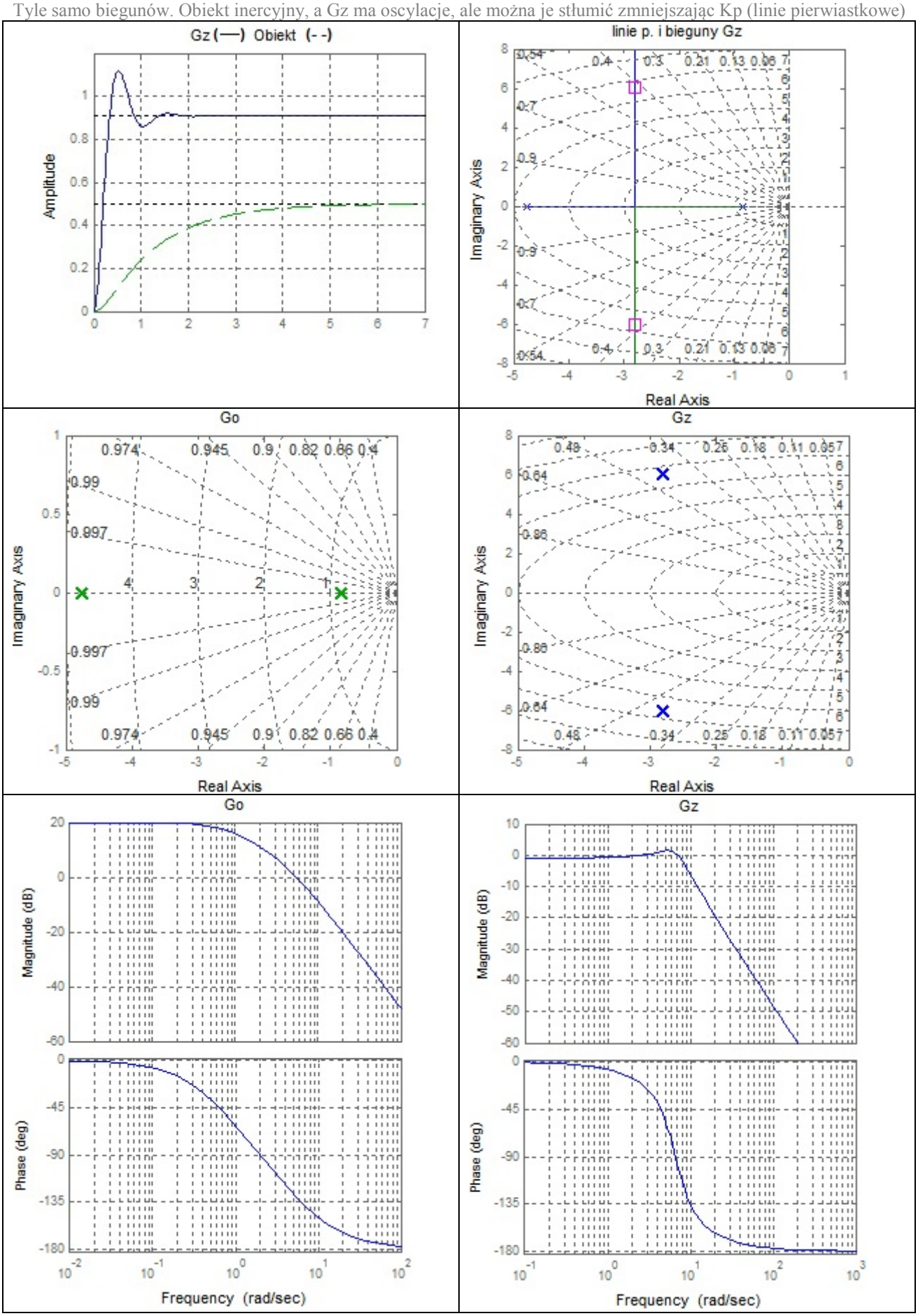


4.2.4 Człon inercyjny 2.rzędu i regulator P

```
2c) k=2;    ksi=1.4; w=2;       %Kp=20;  (duże Kp)
```

	dcgain	Eigenvalue	Damping	Freq. (rad/s)
Obiekt			>1	
Go			>1	
Gz			>1	

Tyle samo biegunów. Obiekt inercyjny, a Gz ma oscylacje, ale można je stłumić zmniejszając Kp (linie pierwiastkowe)



4.3. Obiekt 2.rzędu (oscylacyjny lub inercyjny) i regulator PI

4.3.1 Analiza

$$G_{obiekt} = \frac{k}{s^2 + 2\xi\omega s + \omega^2} \cdot \frac{1}{Ts+1},\ G_{reg} = K_P\left(1 + \frac{K_i}{s}\right) = K_P\frac{s+K_i}{s}$$

$$G_o = \frac{kK_p(s+K_i)}{\left(s^2 + 2\xi\omega_n s + \omega_n^2\right)s} \tag{4-6}$$

$$G_z = \frac{kK_p(s+K_i)}{\left(s^2 + 2\xi\omega_n s + \omega_n^2\right)s + kK_p(s+K_i)} = \frac{kK_p(s+K_i)}{s^3 + 2\xi\omega_n s^2 + \left(\omega_n^2 + kK_p\right)s + kK_pK_i} \tag{4-7}$$

Wnioski (ξ>0, k>0, K_p>0).:

- Rząd układu otwartego i zamkniętego jest o jeden wyższy niż rząd obiektu,
- Regulator PI wprowadza do transmitancji układu otwartego G_o zerowy biegun oraz jedno zero (o ujemnej wartości),
- Regulator PI wprowadza do transmitancji układu otwartego G_o jeden biegun i jedno zero,
  - Odpowiedź skokowa stabilnego UR osiąga stan ustalony:

$$\lim_{t\to\infty} x(t) = \lim_{s\to 0} sG_z(s)u(s) = \lim_{s\to 0} \frac{kK_p(s+K_i)}{s^3 + 2\xi\omega_n s^2 + \left(\omega_n^2 + kK_p\right)s + kK_pK_i} = \frac{kK_pK_i}{kK_pK_i} = 1 \tag{4-8}$$

a uchyb regulacji można wyznaczyć na podstawie:

$$G_e = \frac{\left(s^2 + 2\xi\omega_n s + \omega_n^2\right)s}{\left(s^2 + 2\xi\omega_n s + \omega_n^2\right)s + kK_p(s+K_i)} \tag{4-9}$$

Jeśli regulator PI to uchyb zerowy:

$$\lim_{t\to\infty} e(t) = \lim_{s\to 0} sG_e(s)u(s) = \lim_{s\to 0} \frac{\left(s^2 + 2\xi\omega_n s + \omega_n^2\right)s}{\left(s^2 + 2\xi\omega_n s + \omega_n^2\right)s + kK_p(s+K_i)} = 0 \tag{4-10}$$

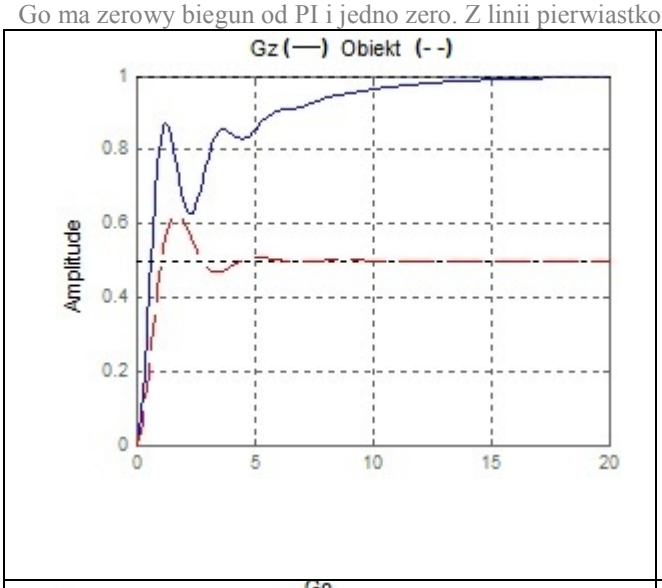
- Jeśli obiekt jest stabilny (ξ>0), to układ z regulatorem PI też będzie stabilny? Kryterium lub linie pierwiastkowe....

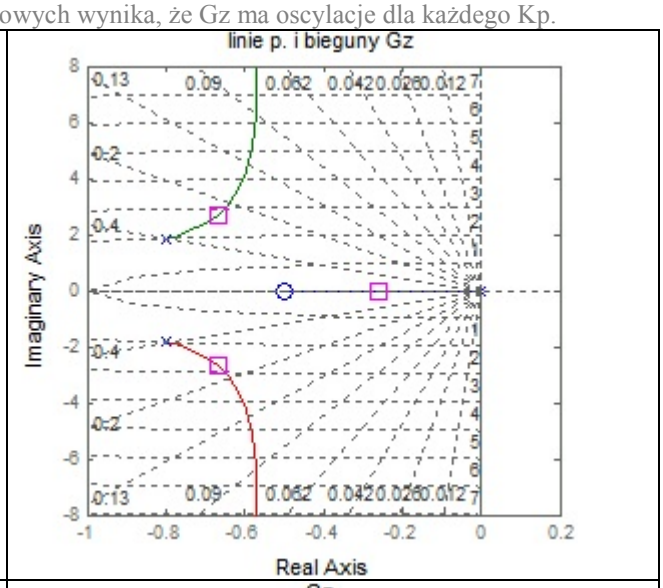
4.3.2 Człon oscylacyjny i regulator PI

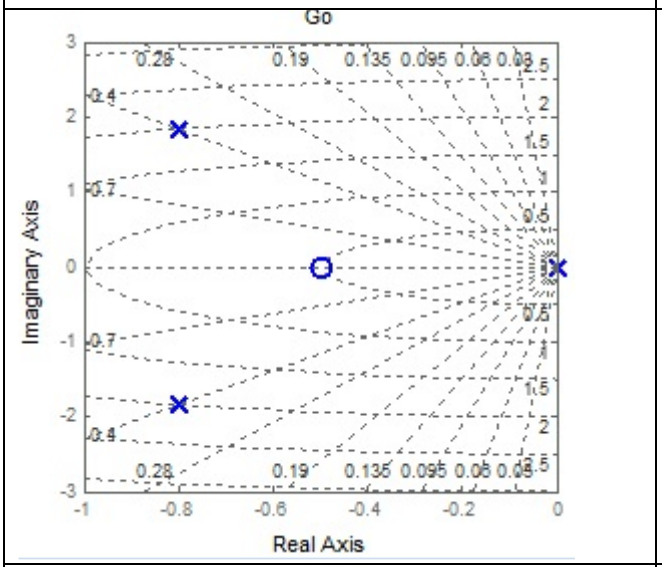
1b) k=2; **ksi=0.4**; w=2 %Kp=2; Ti=2; (małe Kp)

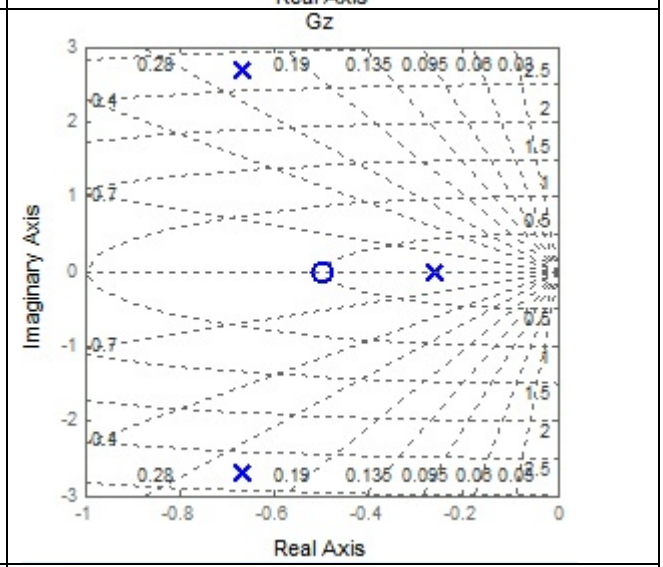
	dcgain	Eigenvalue	Damping	Freq. (rad/s)
Obiekt	0.5	-0.8 +j1.83; -0.8 -j1.83	0.4	2
Go	inf	-0.8 +j1.83; -0.8 -j1.83; 0	0.4	2
Gz	1	-0.669+j2.68; -0.669-j2.68; -0.261	0.242	2.77

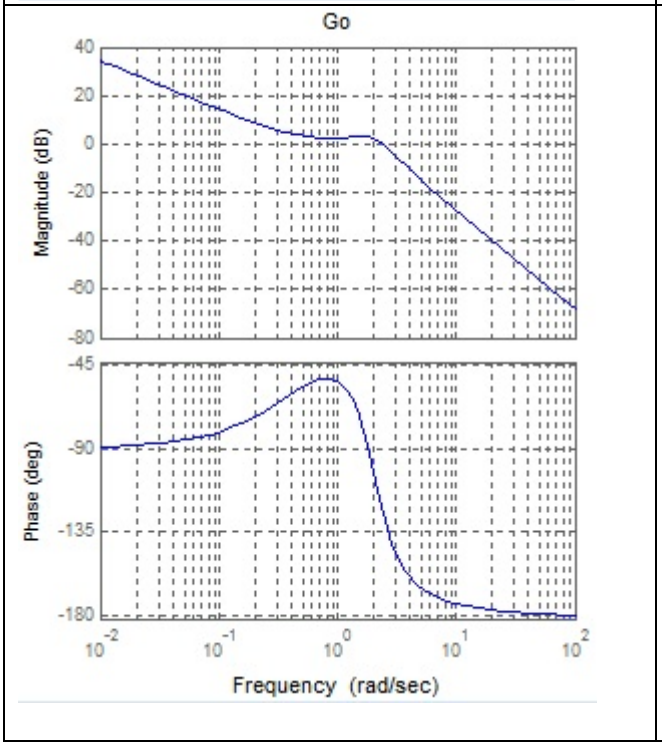
Go ma zerowy biegun od PI i jedno zero. Z linii pierwiastkowych wynika, że Gz ma oscylacje dla każdego Kp.

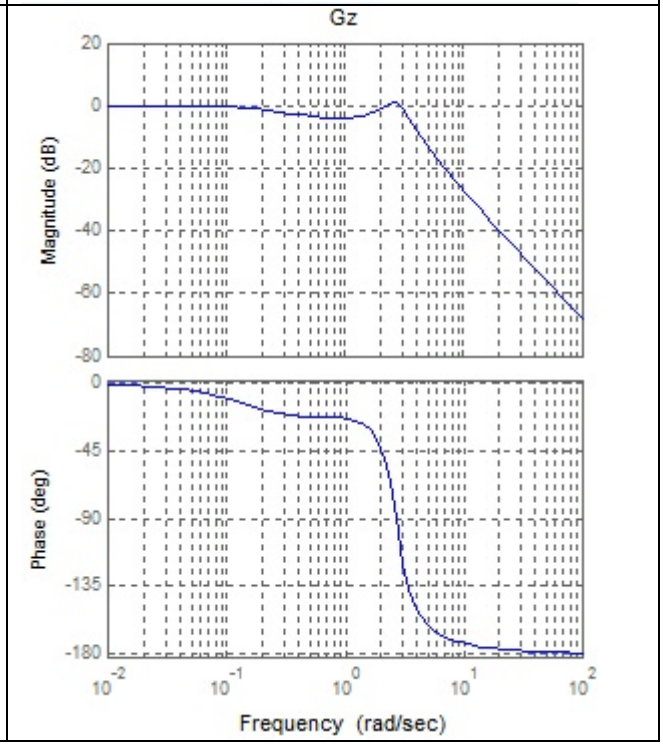










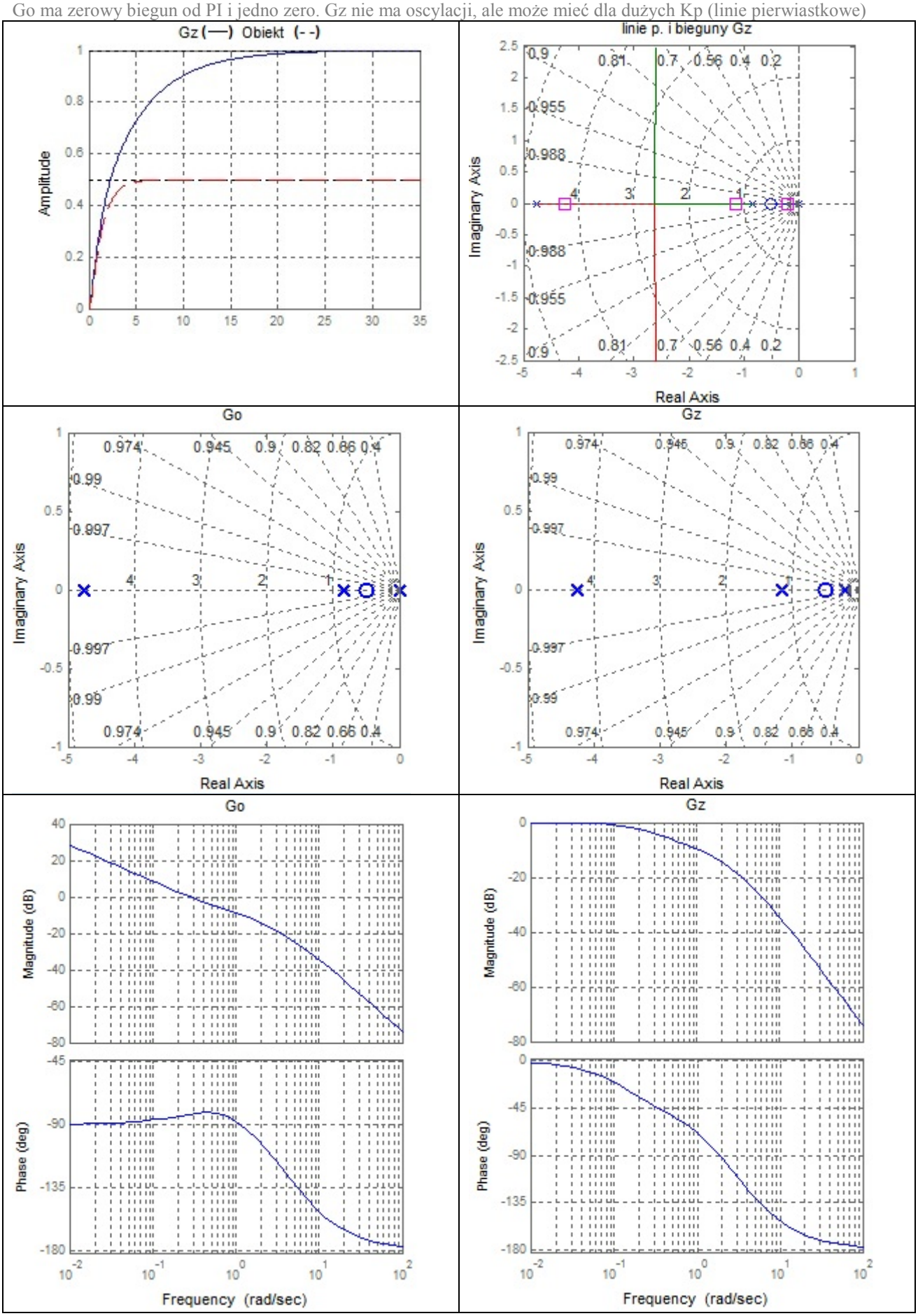


4.3.3 Człon inercyjny 2.rzędu i regulator PI

2b) k=2; **ksi=1.4;** w=2 %Kp=1; Ti=2 (małe Kp)

	dcgain	Eigenvalue	Damping	Freq. (rad/s)
Obiekt				
Go				
Gz				

Go ma zerowy biegun od PI i jedno zero. Gz nie ma oscylacji, ale może mieć dla dużych Kp (linie pierwiastkowe)

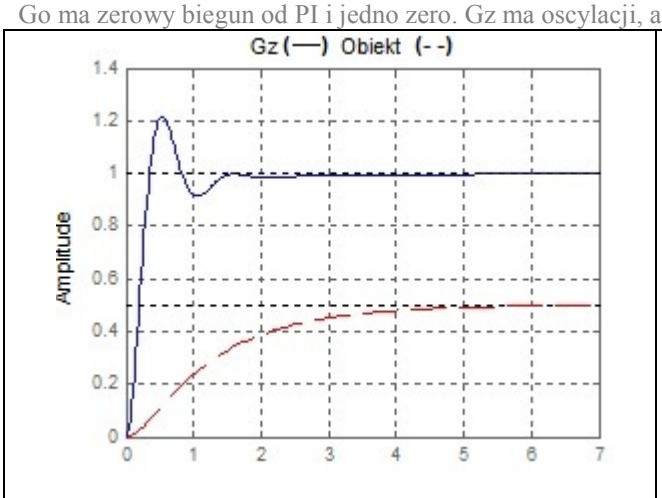


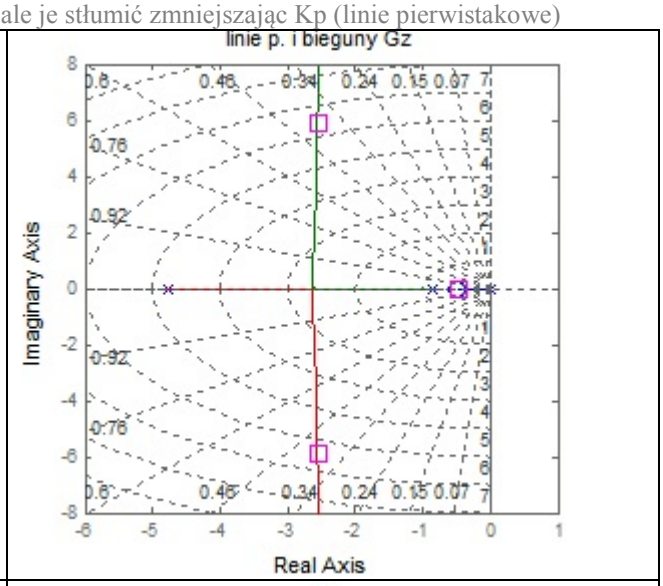
4.3.4 Człon inercyjny 2.rzędu i regulator PI

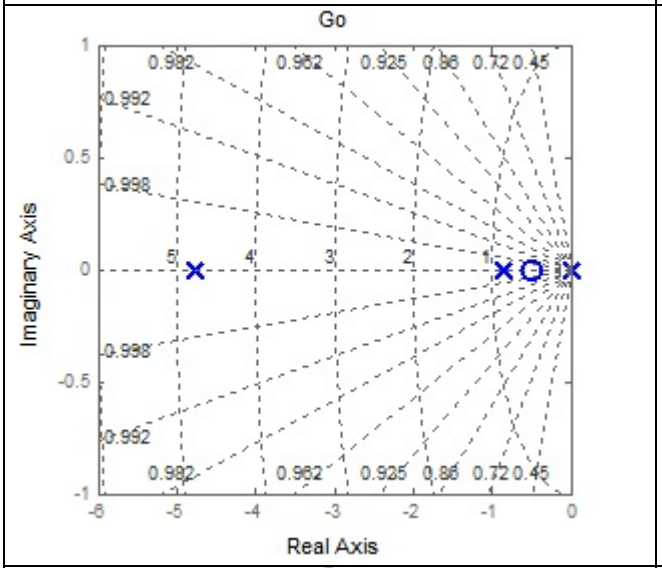
```
2d) k=2;    ksi=1.4; w=2   %Kp=20; Ti=2    (duże Kp)
```

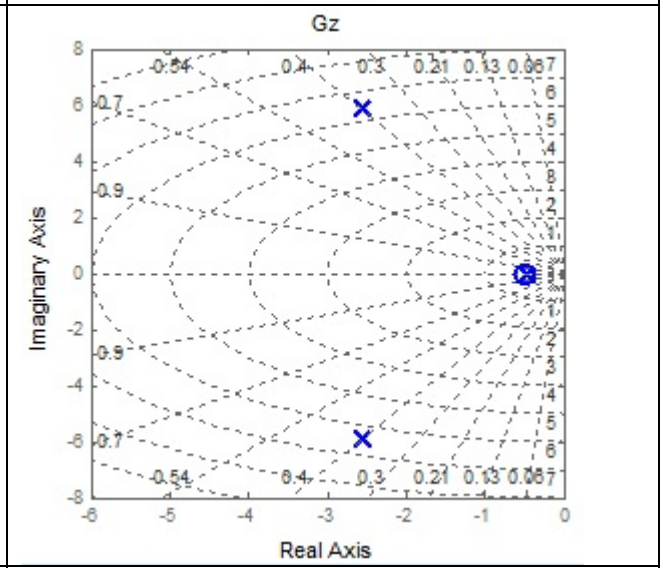
	dcgain	Eigenvalue	Damping	Freq. (rad/s)
Obiekt				
Go				
Gz				

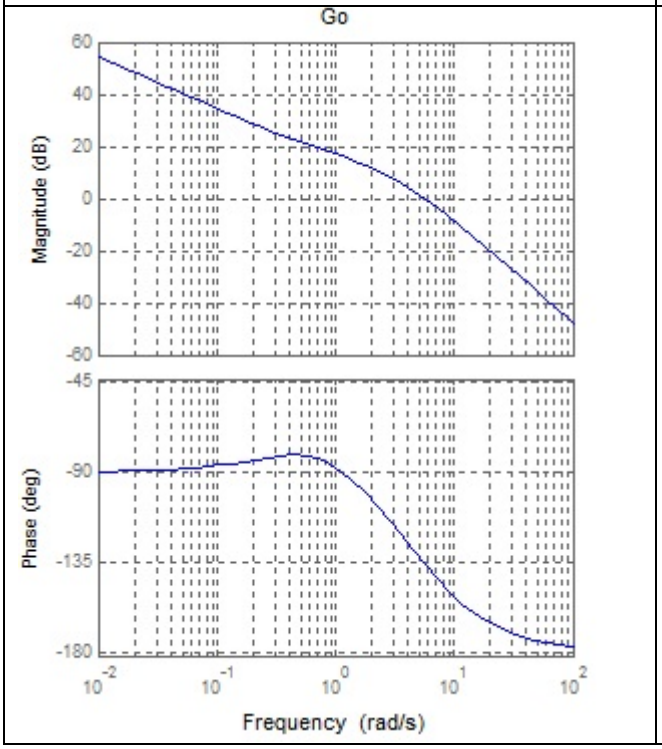
Go ma zerowy biegun od PI i jedno zero. Gz ma oscylacji, ale je stłumić zmniejszając Kp (linie pierwiastkowe)

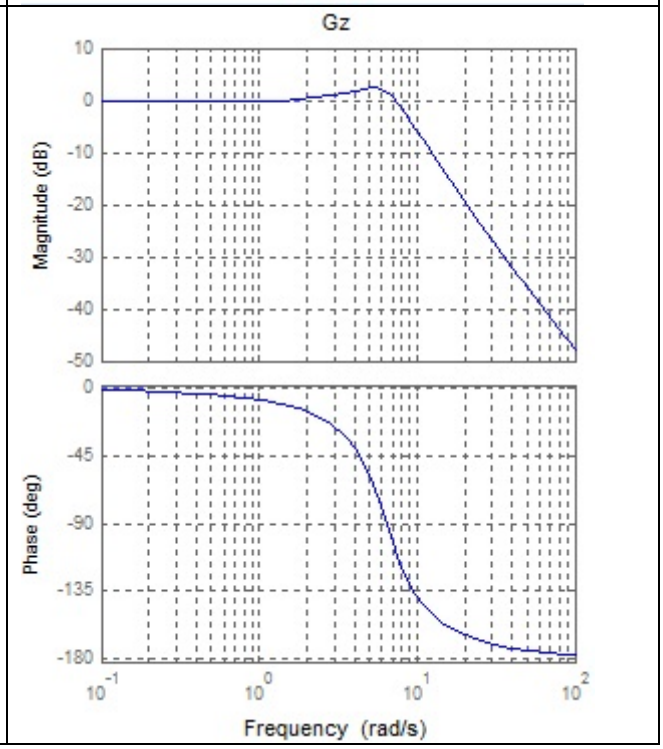












4.4. Obiekt 3.rzędu (oscylacyjny i inercyjny) i regulator P

4.4.1 Analiza

$$G_{obiekt} = \frac{k}{s^2 + 2\xi\omega s + \omega^2} \cdot \frac{1}{Ts+1},\ G_{reg} = \frac{K_P}{1}$$

$$G_o = \frac{kK_p}{\left(s^2 + 2\xi\omega_n s + \omega_n^2\right)(Ts+1)} \tag{4-11}$$

$$G_z = \frac{kK_p}{\left(s^2 + 2\xi\omega_n s + \omega_n^2\right)(Ts+1) + kK_p} = \frac{kK_p}{s^3 + \ldots.} \tag{4-12}$$

Wnioski (ξ>0, k>0, K_p>0).:

- Odpowiedź skokowa stabilnego UR osiąga stan ustalony:

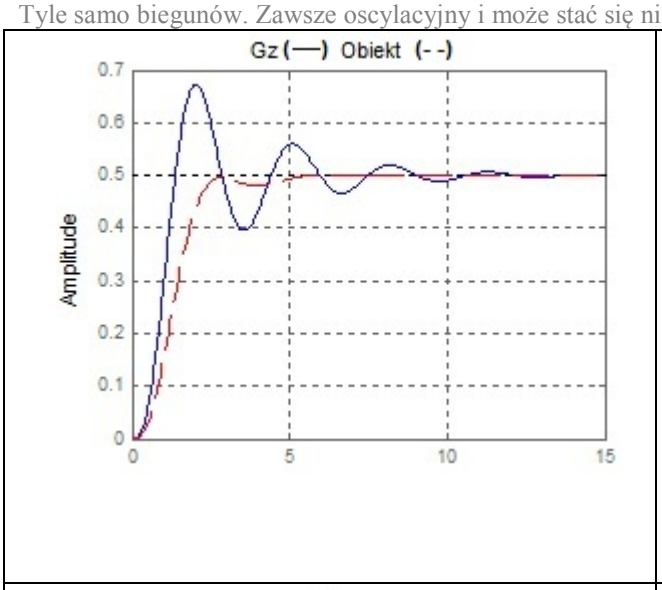
$$\lim_{t\to\infty} x(t) = \lim_{s\to 0} sG_z(s)u(s) = \lim_{s\to 0} \frac{kK_p}{\left(s^2 + 2\xi\omega_n s + \omega_n^2\right)(Ts+1) + kK_p} = \frac{kK_p}{\omega_n^2 + kK_p} \tag{4-13}$$

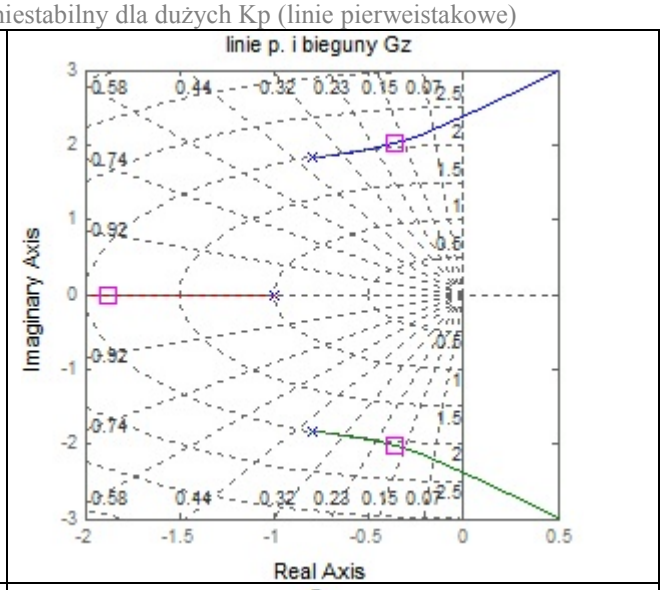
4.4.2 Człon oscylacyjny + inercyjny i regulator P

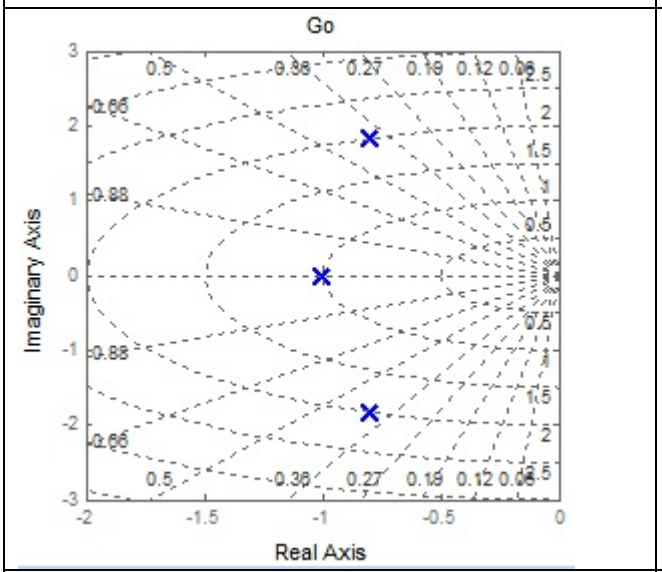
4a) k=2; **ksi=0.4**; w=2; T=1 Kp=2; (małe Kp);

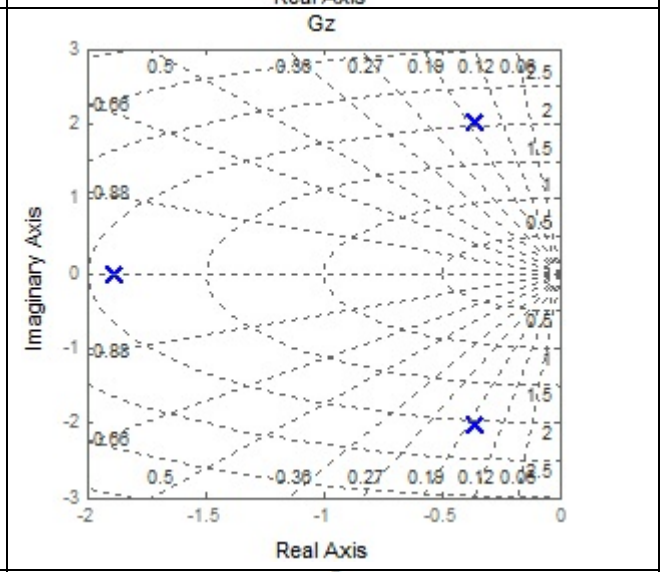
	dcgain	Eigenvalue	Damping	Freq. (rad/s)
Obiekt				
Go				
Gz				

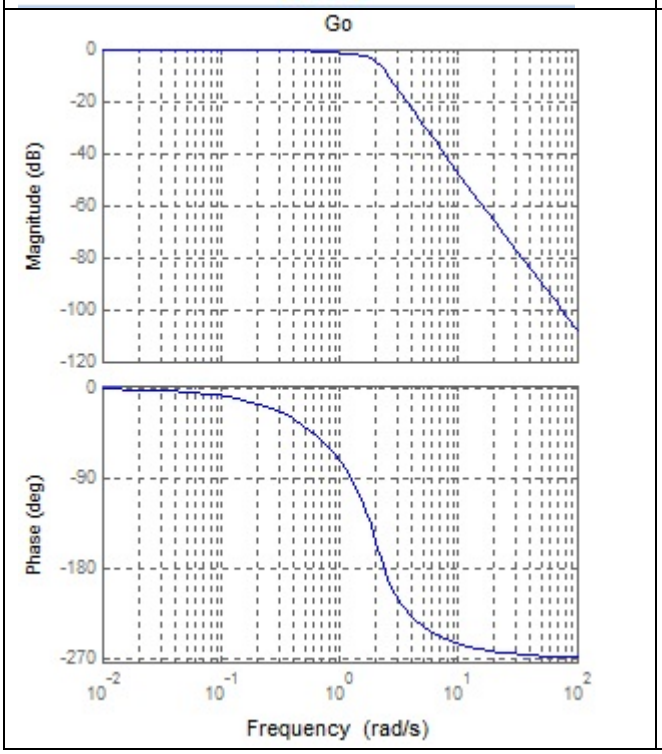
Tyle samo biegunów. Zawsze oscylacyjny i może stać się niestabilny dla dużych Kp (linie pierwiastkowe)

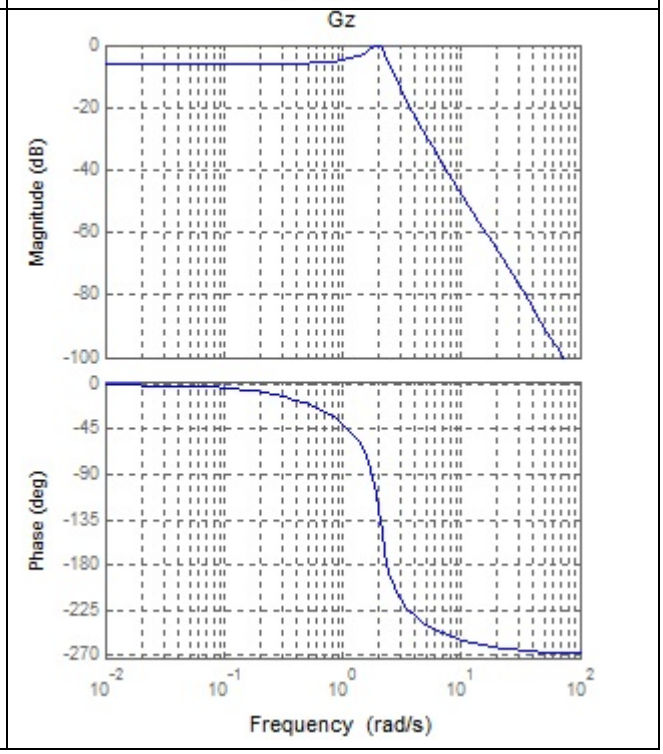










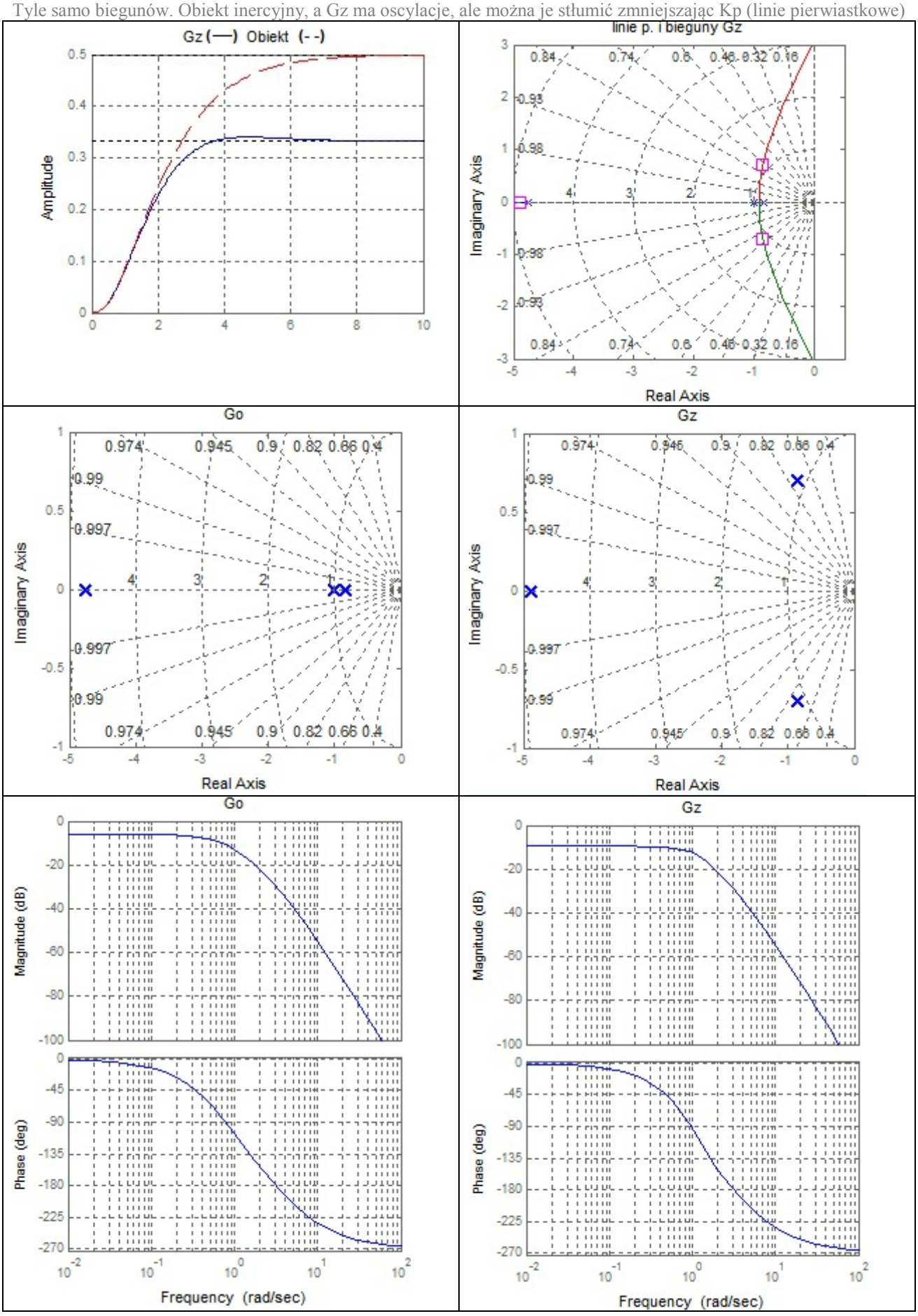


4.4.3 Człon inercyjny 3.rzędu i regulator P

```
5a) k=2;     ksi=1.4; w=2; T=1        %Kp=1; (duże Kp)
```

	dcgain	Eigenvalue	Damping	Freq. (rad/s)
Obiekt				
Go				
Gz				

Tyle samo biegunów. Obiekt inercyjny, a Gz ma oscylacje, ale można je stłumić zmniejszając Kp (linie pierwiastkowe)



4.5. Obiekt 3.rzędu (oscylacyjny i inercyjny) i regulator PI

4.5.1 Analiza

$$G_{obiekt} = \frac{k}{s^2 + 2\xi\omega s + \omega^2} \cdot \frac{1}{Ts+1},\ G_{reg} = K_P\left(1 + \frac{K_i}{s}\right) = K_P\frac{s+K_i}{s}$$

$$G_o = \frac{kK_p\left(s+K_i\right)}{\left(s^2 + 2\xi\omega_n s + \omega_n^2\right)\left(Ts+1\right)s} \tag{4-14}$$

$$G_z = \frac{kK_p\left(s+K_i\right)}{\left(s^2 + 2\xi\omega_n s + \omega_n^2\right)\left(Ts+1\right)s + kK_p\left(s+K_i\right)} = \frac{kK_p(s+K_i)}{s^3 + ...} \tag{4-15}$$

Wnioski (ξ>0, k>0, K_p>0).:

....

4.5.2 Człon oscylacyjny +inercyjny i regulator PI

```
4b) k=2;    ksi=0.4; w=2; T=1      %Kp=2; Ti=2;   (małe Kp)
```

	dcgain	Eigenvalue	Damping	Freq. (rad/s)
Obiekt				
Go				
Gz				

Go ma zerowy biegun od PI i jedno zero od PI. Gz ma oscylacje dla każdego Kp i może stać się niestabilny