Wydział Elektroniki i Technik Informacyjnych
Politechnika Warszawska

Projektowanie układów sterowania (projekt grupowy)

Sprawozdanie z projektu i ćwiczenia laboratoryjnego nr 1, zadanie nr 1

Imię i Nazwisko, Imię i Nazwisko, Imię i Nazwisko

Warszawa, 2017

Spis treści

1. Wstęp

Sprawozdania przygotowywane w ramach projektów i ćwiczeń laboratoryjnych muszą być opracowane w systemie LaTeX. System składu dokumentów LaTeX jest całkowicie darmowy, ale umożliwia opracowanie bardzo dobrze złożonych dokumentów. Do przygotowania sprawozdania należy wykorzystać klasę `mwrep` z pakietu klas `mwcls` [4] oraz klasę `polski`. W przypadku dłuższych opracowań (książek, prac dyplomowych) należy wykorzystać klasę `mwbk`.

Jeżeli dostępne są rysunki w formacie `pdf`, najwygodniej do przetworzenia dokumentu użyć polecenia `pdflatex`, które bezpośrednio generuje dokument w formacie `pdf`. Polecenie `latex` wymaga rysunków w formacie `ps` lub `eps` i generuje dokument w formacie `dvi`, który następnie można przekształcić do formatu `eps` lub `pdf`. Nie używamy rysunków zapisanych w plikach bitmapowych (`bmp`, `jpg`, `png`). Jedynym wyjątkiem są zdjęcia.

Istnieje wiele podręczników do nauki zasad składania dokumentów w LaTeXu, np. doskonała praca zbiorowa [2] lub ew. podręcznik Wikibooks [5]. Do edycji dokumentów można wykorzystać np. program TeXnicCenter, dostępny pod adresem `http://www.texniccenter.org`. W przypadku problemów warto poszukać rozwiązania na forum `http://tex.stackexchange.com`.

W dalszej części dokumentu podano najważniejsze wymagania dotyczące wzorów matematycznych, tabeli i rysunków. Najszybszą metodą prowadzącą do otrzymania dokumentu jest modyfikacja niniejszego szablonu.

2. Wzory matematyczne

Stosujemy przecinek dziesiętny, a nie kropkę dziesiętną. Aby uniknąć dodatkowego odstępu, stosujemy pakiet `siunitx`, umożliwiający zapis `\num{1,2345}` lub `\num{1.2345}`, co prowadzi do 1,2345, a nie `$1,2345$`, co prowadzi do $1, 2345$. Stosujemy zapis $1{,}2345 \cdot 10^{10}$, a nie $1{,}2345 \times 10^{10}$. Powyższy zapis można stosować również w trybie matematycznym, np. `$\num{1.2345e10}$` skompiluje się do $1{,}2345 \cdot 10^{10}$.

2.1. Stałe i zmienne, indeksowanie

Skalarne stałe i zmienne zapisujemy w trybie matematycznym, np. x, y, z. Stosujemy indeksy dolne, np. x_i, górne, np. x^j, lub oba, np. x_i^j. Można również zastosować indeksy w nawiasach, np. $y(k)$. Jeżeli indeks zapisany jest czcionką pochyłą, spodziewamy się, że przyjmuje on wartość liczbową (liczby naturalne), np. x_i dla $i = 1, \ldots, 10$. Jeżeli natomiast zastosujemy oznaczenie x_{i}, to wówczas indeks i nie przyjmuje żadnej wartości, jest on integralną częścią zmiennej lub stałej. Dlatego oznaczając horyzont sterowania stosujemy symbol N_{u}, a nie N_u, co by sugerowało, że indeks u przyjmuje pewne wartości z zakresu liczb naturalnych. Analogicznie, stała czasowa całkowania oznaczana jest jako T_{i}, a nie jako T_i, stała czasowa różniczkowania to T_{d}, a nie T_d. Sygnał wartości zadanej oznaczamy przez y^{zad}, a nie przez y^{zad}.

Nie należy stosować czcionki pochyłej również do tekstów, które uzupełniają wyrażenia matematyczne, np. zamiast błędnej postaci

$$y(x) = \begin{cases} x^2 & gdy\ x \leqslant 0 \\ x^3 & gdy\ x > 0 \end{cases}$$

powinno być

$$y(x) = \begin{cases} x^2 & \text{gdy } x \leqslant 0 \\ x^3 & \text{gdy } x > 0 \end{cases}$$

Odstępy w trybie matematycznym wymuszamy za pomocą instrukcji `\,`, `\quad`, `\qquad` itd.

2.2. Wektory

Do oznaczenia wektorów najczęściej stosujemy symbole pogrubione, np. $\boldsymbol{x}$, $\triangle\boldsymbol{u}(k)$. Pamiętamy, że w matematyce wektory zawsze są pionowe. Wektory, których elementami są skalary, zapisujemy więc jako

$$\triangle\boldsymbol{u}(k) = [\triangle u(k|k) \ \ldots \ \triangle u(k + N_{\mathrm{u}} - 1|k)]^{\mathrm{T}} \tag{2.1}$$

lub w postaci

$$\triangle\boldsymbol{u}(k) = \begin{bmatrix} \triangle u(k|k) \\ \vdots \\ \triangle u(k + N_{\mathrm{u}} - 1|k) \end{bmatrix} \tag{2.2}$$

Jeżeli używamy wektorów, których elementami składowymi są inne wektory, najwygodniej zapisać je pionowo. Np. elementami wektora (2.2) są podwektory

$$\triangle u(k+p|k) = \begin{bmatrix} \triangle u_1(k+p|k) \\ \vdots \\ \triangle u_{n_\mathrm{u}}(k+p|k) \end{bmatrix} \tag{2.3}$$

gdzie $p = 1, \ldots, N_\mathrm{u}$. A więc każdy z wektorów (2.3) ma długość n_u, wektor (2.2) ma długość $n_\mathrm{u} N_\mathrm{u}$.

2.3. Macierze

Do oznaczenia macierzy najczęściej stosujemy symbole pogrubione, np. macierz dynamiczna w algorytmie DMC dla procesu o jednym wejściu i jednym wyjściu ma wymiar $N \times N_\mathrm{u}$ i strukturę

$$\boldsymbol{G} = \begin{bmatrix} s_1 & 0 & \ldots & 0 \\ s_2 & s_1 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ s_N & s_{N-1} & \ldots & s_{N-N_\mathrm{u}+1} \end{bmatrix} \tag{2.4}$$

W przypadku procesu o n_u wejściach i n_y wyjściach ma ona wymiar $N \times N_\mathrm{u}$ i postać

$$\boldsymbol{G} = \begin{bmatrix} \boldsymbol{S}_1 & \mathbf{0}_{n_\mathrm{y} \times n_\mathrm{u}} & \ldots & \mathbf{0}_{n_\mathrm{y} \times n_\mathrm{u}} \\ \boldsymbol{S}_2 & \boldsymbol{S}_1 & \ldots & \mathbf{0}_{n_\mathrm{y} \times n_\mathrm{u}} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{S}_N & \boldsymbol{S}_{N-1} & \ldots & \boldsymbol{S}_{N-N_\mathrm{u}+1} \end{bmatrix} \tag{2.5}$$

gdzie każda z macierzy składowych ma wymiar $n_\mathrm{y} \times n_\mathrm{u}$

$$\boldsymbol{S}_p = \begin{bmatrix} s_p^{1,1} & \ldots & s_p^{1,n_\mathrm{u}} \\ \vdots & \ddots & \vdots \\ s_p^{n_\mathrm{y},1} & \ldots & s_p^{n_\mathrm{y},n_\mathrm{u}} \end{bmatrix} \tag{2.6}$$

gdzie $p = 1, \ldots, N$. A więc macierz (2.5) ma wymiar $n_\mathrm{y} N \times n_\mathrm{u} N_\mathrm{u}$.

2.4. Większe wyrażenia matematyczne

W przypadku długich wzorów nie należy korzystać z otoczenia `equation`, ponieważ wzór taki zwykle nie mieści się na stronie o przyjętej szerokości, np.

$$y(k) = b_1 u(k-1) + b_2 u(k-2) + b_3 u(k-3) + b_4 u(k-4) + b_5 u(k-5) - a_1 y(k-1) - a_2 y(k-2) - a_3 y(k-3) - a_4 y(k-4) - a_5 y \tag{2.7}$$

Należy zastosować otoczenie `align`, co prowadzi do wzoru

$$\begin{aligned} y(k) = {} & b_1 u(k-1) + b_2 u(k-2) + b_3 u(k-3) + b_4 u(k-4) + b_5 u(k-5) \\ & - a_1 y(k-1) - a_2 y(k-2) - a_3 y(k-3) - a_4 y(k-4) - a_5 y(k-5) \end{aligned} \tag{2.8}$$

Nie stosujemy otoczenia `split` z powodu błędnego centrowania. Numer wzoru złożonego z wielu wierszy umieszczamy tylko w ostatnim wierszu.

3. Tabele

W praktyce bardzo często należy wyrównać liczby względem cyfr znaczących w poszczególnych kolumnach (czyli przecinek dziesiętny ma być we wszystkich wierszach tabeli umieszczony w tym samym miejscu w pionie). Do wyrównania liczb można wykorzystać pakiet `siunitx` (pakiety `rccol` oraz `dcolumn` mają mniejsze możliwości). Kod źródłowy służący do otrzymania tab. 3.1 jest następujący:

```
\begin{table}
 [b] \caption{Tabela z~wyrównaniem liczb do znaku przecinka dziesiętnego}
 \label{t_wyrownanie_do_znaku_przecinek1}
 \centering
 \sisetup{table-format = 2.4}
 \begin{small}
  \begin{tabular}{|l|S[table-format=2]|S|S|S|}
   \hline
    \multicolumn{1}{|c|}{Model\rule{0pt}{3.5mm}} & LP & $\mathrm{SSE_{ucz}}$
                & $\mathrm{SSE_{wer}}$ & $\mathrm{SSE_{test}}$ \\ \hline
    Liniowy \rule{0pt}{3.5mm} & 4 & 90.1815 & 70.7787 & \textemdash \\
    Neuronowy, $K=1$ & 7 & 10.1649 & 10.3895 & \textemdash \\
    Neuronowy, $K=2$ & 13 & 0.3282 & 0.3257  & \textemdash \\
    Neuronowy, $K=3$ & 19 & 0.2014 & 0.1827  & 0.1468 \\
    Neuronowy, $K=4$ & 25 & 0.1987 & 0.1906 & \textemdash \\
    Neuronowy, $K=5$ & 31 & 0.1364 & 0.1971 & \textemdash \\
    Neuronowy, $K=6$ & 37 & 0.1340 & 0.2044 & \textemdash \\
    \hline
  \end{tabular}
 \end{small}
\end{table}
```

Zwróćmy uwagę, że metoda ta działa również wówczas, gdy stosuje się notację wykładniczą, co demonstruje tab. 3.2. Polecenie `\multicolumn` wyśrodkowuje nagłówki tabeli.

Jeżeli standardowa szerokość kolumn jest za mała, należy w dowolnym wierszu wstawić z obu stron zawartości komórki polecenia `\hspace{odległość}`, które zapewniają odpowiednią szerokość. Modyfikację taką zastosowano w drugiej kolumnie tab. 3.2.

Jeżeli tabela jest szersza niż szerokość strony, należy zastosować otoczenie `sidewaystable` z pakietu `rotating`, co wykorzystano w tab. 3.3.

W zamieszczonych tabelkach wykorzystano polecenie `\rule` do wstawienia linii o zerowej szerokości do wierszy tabelek, które są zbyt wąskie.

Tab. 3.1. Tabela z wyrównaniem liczb do znaku przecinka dziesiętnego

Model	LP	$\mathrm{SSE_{ucz}}$	$\mathrm{SSE_{wer}}$	$\mathrm{SSE_{test}}$
Liniowy	4	90,1815	70,7787	—
Neuronowy, $K = 1$	7	10,1649	10,3895	—
Neuronowy, $K = 2$	13	0,3282	0,3257	—
Neuronowy, $K = 3$	19	0,2014	0,1827	0,1468
Neuronowy, $K = 4$	25	0,1987	0,1906	—
Neuronowy, $K = 5$	31	0,1364	0,1971	—
Neuronowy, $K = 6$	37	0,1340	0,2044	—

Tab. 3.2. Tabela z wyrównaniem liczb do znaku przecinka dziesiętnego i notacją wykładniczą

Model	LP	$\mathrm{SSE}_{\mathrm{ucz}}$	$\mathrm{SSE}_{\mathrm{wer}}$	$\mathrm{SSE}_{\mathrm{test}}$
Liniowy	4	$9{,}1815 \cdot 10^{1}$	$7{,}7787 \cdot 10^{1}$	—
Neuronowy, $K = 1$	7	$1{,}1649 \cdot 10^{1}$	$1{,}3895 \cdot 10^{1}$	—
Neuronowy, $K = 2$	13	$3{,}2821 \cdot 10^{-1}$	$3{,}2568 \cdot 10^{-1}$	—
Neuronowy, $K = 3$	19	$2{,}0137 \cdot 10^{-1}$	$1{,}8273 \cdot 10^{-1}$	$1{,}4682 \cdot 10^{-1}$
Neuronowy, $K = 4$	25	$1{,}9868 \cdot 10^{-1}$	$1{,}9063 \cdot 10^{-1}$	—
Neuronowy, $K = 5$	31	$1{,}3642 \cdot 10^{-1}$	$1{,}9712 \cdot 10^{-1}$	—
Neuronowy, $K = 6$	37	$1{,}3404 \cdot 10^{-1}$	$2{,}0440 \cdot 10^{-1}$	—

Tab. 3.3. Tabela obrócona o 90°

Algorytm	N	$N_\mathrm{u} = 1$	$N_\mathrm{u} = 2$	$N_\mathrm{u} = 3$	$N_\mathrm{u} = 4$	$N_\mathrm{u} = 5$	$N_\mathrm{u} = 10$	$N_\mathrm{u} = 15$	$N_\mathrm{u} = 20$	$N_\mathrm{u} = 30$
NPL	5	0,40	0,53	0,85	1,29	1,92	—	—	—	—
NO	5	2,61	5,04	8,00	12,65	18,37	—	—	—	—
NO_apr	5	2,47	4,32	7,98	15,25	26,53	—	—	—	—

4. Rysunki

Wszystkie elementy dokumentu opracowanego w systemie LaTeXpowinny wyglądać jednolicie. Do wykonywania rysunków korzystamy więc z mechanizmów oferowanych przez dodatkowe pakiety LaTeXa, nie dołączamy rysunków wykonanych jakościowo różnych, np. wykonanych w programie Word. Nie używamy rysunkach zapisanych w plikach bitmapowych, lecz w plikach wektorowych (`pdf`, ew. `ps` lub `eps`). Jedynym wyjątkiem są zdjęcia.

4.1. Schematy blokowe

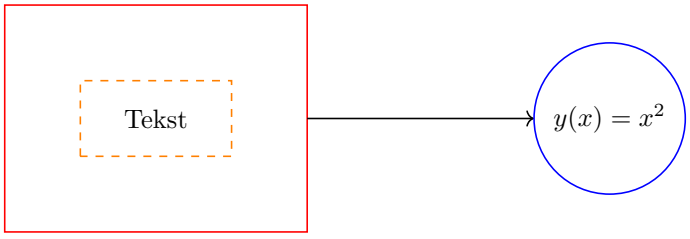
Do opracowania schematów blokowych najlepiej wykorzystać język opisu rysunków `TikZ/PGF` [3]. Przy wykonywaniu prostych rysunków po prostu opisujemy je za pomocą poleceń dodających kolejne elementy, tzn. prostokąty, okręgi, linie. Na przykład, ciąg poleceń:

```
\begin{figure}[b]
\centering
\begin{tikzpicture}
\draw [red, semithick] (0,0) rectangle (4,3);
\draw [orange, semithick,dashed] (1,1) rectangle (3,2);
\draw [->,semithick] (4,1.5) -- (7,1.5);
\draw [blue, semithick] (8,1.5) circle [radius=1];
\node at (2,1.5) {Tekst};
\node at (8,1.5) {$y(x)=x^2$};
\end{tikzpicture}
\end{figure}
```

pozwala narysować figury geometryczne przedstawione na rys. 4.1. Zwróćmy uwagę, że napis oraz wzór są złożone aktualnie wykorzystywaną czcionką, jej wielkość jest taka sama jak w całym dokumencie.

Przy większych rysunkach można wykorzystać programy umożliwiające ich przygotowanie przy wykorzystaniu środowiska graficznego, np. `TikzEdt` (`http://www.tikzedt.org/`), `TpX` (`http://tpx.sourceforge.net/`), `ktikz` (`https://www.linux-apps.com/p/1126914/`), `GraTeX` (`https://sourceforge.net/projects/gratex/`).

Można również wykorzystać starsze, klasyczne pakiety `picture`, `epic`, `eepic`. Również w tych przypadkach można „ręcznie” opisywać poszczególne elementy graficzne lub skorzystać ze środowiska graficznego, np. `LaTeXPiX` (`http://latexpix.software.informer.com/`), które znacznie przyspiesza pracę. Inne narzędzia, umożliwiające opracowanie rysunków wysokiej jakości, to `METAPOST` oraz `PSTricks`.



Rys. 4.1. Tekst Przykładowy rysunek wykorzystany w języku `TikZ/PGF`

4.2. Funkcje statyczne

Do wykonywania wykresów prezentujących wyniki symulacji i eksperymentów stosuje się pakiet PGFPLOTS [1]. Załóżmy, że w katalogu rysunki/dane_stat znajduje się plik dane_fx.txt zawierający w pierwszej kolumnie wartości argumentu x, natomiast w drugiej kolumnie wartości funkcji $f(x)$

```
 -10.0000 -808.7350
  -9.0000 -696.4791
  -8.0000 -539.7850
  -7.0000 -386.1268
  -6.0000 -291.5881
  -5.0000 -267.6436
  -4.0000 -268.9551
  -3.0000 -233.3995
  -2.0000 -137.5303
  -1.0000  -16.4788
        0   70.0000
   1.0000   94.1212
   2.0000   87.2697
   3.0000  112.8005
   4.0000  209.4449
   5.0000  357.3564
   6.0000  498.0119
   7.0000  589.6732
   8.0000  647.4150
   9.0000  730.9209
  10.0000  891.2650
```

oraz podobny plik dane_gx.txt, definiujący funkcję $g(x)$. Aby narysować te funkcje stosuje się polecenia:

```
\begin{figure}[t]
\centering
\begin{tikzpicture}
\begin{axis}[
width=0.5\textwidth,
xmin=-10,xmax=10,ymin=-1000,ymax=1000,
xlabel={$x$},
ylabel={$f(x), \ g(x)$},
xtick={-10,-5,0,5,10},
ytick={-1000,-500,0,500,1000},
legend pos=south east,
y tick label style={/pgf/number format/1000 sep=},
]
\addplot[red,semithick] file {rysunki/dane_stat/dane_fx.txt};
\addplot[blue,semithick,densely dashed]
file {rysunki/dane_stat/dane_gx.txt};
\legend{$f(x)$,$g(x)$}
\end{axis}
\end{tikzpicture}
\caption{Przykładowy rysunek funkcji $f(x)$ i~$g(x)$ wykonany
w~języku \texttt{PGFPLOTS}}
\label{r_pgfplots_funkcje}
\end{figure}
```

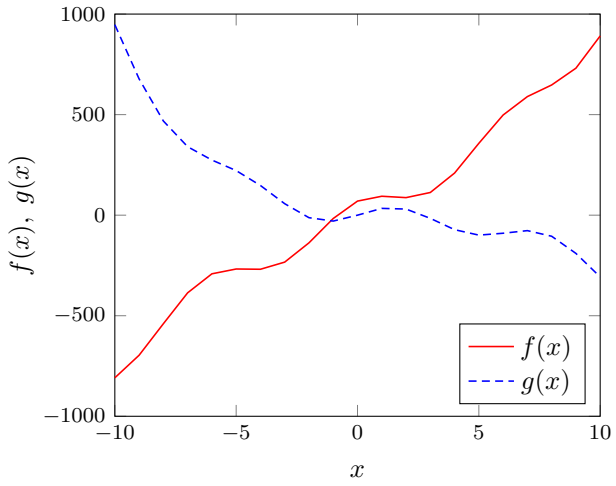
Otrzymany rezultat przedstawiono na rys. 4.2. Istnieje możliwość ustawienia wielkości czcionek liczb umieszczonych: na osiach (tick label style), w oznaczeniach osi (label style), w legendzie (legend style) oraz w tytule rysunku (title style). Przykładowa konfiguracja zmieniająca wielkość czcionek jest następująca:

```
\pgfplotsset{
```

```
tick label style={font=\tiny},
label style={font=\footnotesize},
legend style={font=\footnotesize},
title style={font=\footnotesize}
}
```

Opisany sposób implementacji rysunków jest poprawny, ale ma poważną wadę, ponieważ LaTeX potrzebuje dość dużo czasu na ich przetworzenie. Okazuje się to dużym mankamentem szczególnie wówczas, gdy w dokumencie znajduje się dużo skomplikowanych rysunków. Skutecznym rozwiązaniem jest przygotowanie rysunków i zapis ich do plików `pdf`, a następnie dołączenie ich do głównego dokumentu poleceniem `\includegraphics`. Plik `zapisz_pdf_funkcje.tex`, umożliwiający zapisanie rysunku do pliku `funkcje.pdf`, znajduje się w katalogu `rysunki/zapisz_pdf` i ma następującą postać:

```
\documentclass[a4paper,11pt]{article}
\usepackage{pgfplots}
\usetikzlibrary{pgfplots.groupplots}
\pgfplotsset{compat=1.11}
\usepgfplotslibrary{external}
\tikzexternalize

\textwidth 160mm \textheight 247mm

\pgfplotsset{width=\figurewidth,compat=1.11}
\pgfplotsset{
tick label style={font=\tiny},
label style={font=\footnotesize},
legend style={font=\footnotesize},
title style={font=\footnotesize}
}

\begin{document}

\tikzsetnextfilename{}

\begin{figure}[tb]
\tikzsetnextfilename{funkcje}
\begin{tikzpicture}
\begin{axis}[
width=0.5\textwidth,
```



Rys. 4.2. Przykładowy rysunek funkcji $f(x)$ i $g(x)$ wykonany w języku `PGFPLOTS` (rysunek jest kompilowany przy każdym przetworzenia pliku źródłowego

```
xmin=-10,xmax=10,ymin=-1000,ymax=1000,
xlabel={$x$},
ylabel={$f(x), \ g(x)$},
xtick={-10,-5,0,5,10},
ytick={-1000,-500,0,500,1000},
legend pos=south east,
y tick label style={/pgf/number format/1000 sep=},
]
\addplot[red,semithick] file {../dane_stat/dane_fx.txt};
\addplot[blue,semithick,densely dashed] file {../dane_stat/dane_gx.txt};
\legend{$f(x)$,$g(x)$}
\end{axis}
\end{tikzpicture}
\end{figure}

\end{document}
```

Polecenie

```
pdflatex -shell-escape zapisz_pdf_funkcje.tex
```

zapisuje plik `funkcje.pdf`. Umieszczając wiele definicji rysunków w jednym pliku źródłowym można generować wiele rysunków w postaci plików `pdf`. Rysunek dołącza się do dokumentu ciągiem instrukcji:

```
\begin{figure}[tb]
\centering
\includegraphics[scale=1]{pgfplots_pdf/funkcje}
\caption{Przykładowy rysunek funkcji $f(x)$ i~$g(x)$ wykonany
w~języku \texttt{PGFPLOTS}}
\label{r_pgfplots_funkcje}
\end{figure}
```

Otrzymany rezultat przedstawiono na rys. 4.3. Oczywiście, rysunki 4.2 i 4.3 są bardzo podobne, jedyną różnicą jest wielkość czcionek.

Nie należy skalować rysunku, gdyż zmieni to wielkość zastosowanej czcionki. Jeżeli zachodzi konieczność zmiany wielkości rysunku, należy zmodyfikować plik źródłowy generujący rysunek.

Jeżeli rysunek jest szerszy niż szerokość strony, należy zastosować otoczenie `sidewaysfigure` z pakietu `rotating`, które działa analogicznie jak otoczenie `sidewaystable`.

Przy dużych zbiorach danych LaTeX zgłasza błąd pamięci. Należy wówczas zastosować LuaLaTeX.

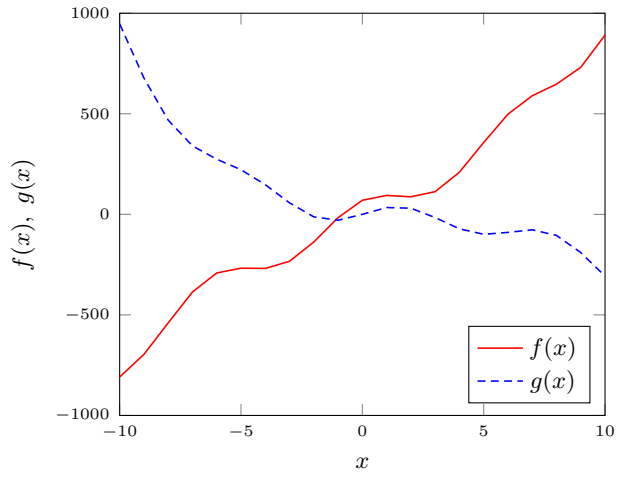
W bardzo podobny sposób przygotowuje się rysunki trójwymiarowe [1].

4.3. Wyniki symulacji i eksperymentów

Załóżmy, że w katalogu `rysunki/symulacje11` znajduje się plik `yzad.txt` zawierający w pierwszej kolumnie pomiary czasu t (w sekundach), natomiast w drugiej kolumnie próbki sygnału wartości zadanej y^{zad}. Wykonano symulacje algorytmu regulacji GPC przy pięciu różnych wartościach parametru λ: 0,1, 0,2, 0,5, 1 i 2. Przebiegi sygnału sterującego u zapisano w plikach `u_lambda_0_1.txt`, `u_lambda_0_2.txt`, `u_lambda_0_5.txt`, `u_lambda_1.txt` i `u_lambda_2.txt`, natomiast przebiegi sygnału wyjściowego procesu y zapisano w plikach `y_lambda_0_1.txt`, `y_lambda_0_2.txt`, `y_lambda_0_5.txt`, `y_lambda_1.txt` i `y_lambda_2.txt`. Przygotowano plik `zapisz_pdf_symulacje11.tex`, umożliwiający zapisanie rysunku do pliku `symulacje11.pdf`. Znajduje się on w katalogu `rysunki/zapisz_pdf` i ma następującą postać:

```
\documentclass[a4paper,11pt]{article}
\usepackage{pgfplots}
\usetikzlibrary{pgfplots.groupplots}
\pgfplotsset{compat=1.11}
```

```
\usepgfplotslibrary{external}
\tikzexternalize

\textwidth 160mm \textheight 247mm

\pgfplotsset{width=\figurewidth,compat=1.11}
\pgfplotsset{
tick label style={font=\tiny},
label style={font=\footnotesize},
legend style={font=\footnotesize},
title style={font=\footnotesize}
}

\newcommand{\szer}{16cm}
\newcommand{\wys}{5.6cm}
\newcommand{\odstepionowy}{1.2cm}

\definecolor{kolor4}{rgb}{0,0,0.1724}
\definecolor{kolor5}{rgb}{1.0000,0.1034,0.7241}
\definecolor{kolor6}{rgb}{1.0000,0.8276,0}

\begin{document}

\tikzsetnextfilename{}

\begin{figure}[tb]
\tikzsetnextfilename{symulacje11}
\begin{tikzpicture}
\begin{groupplot}[group style={group size=1 by 2,vertical sep=\odstepionowy},
width=\szer,height=\wys]
%%1
\nextgroupplot
[xmin=0,xmax=1,ymin=-4.5,ymax=8.5,
xtick={0,0.2,0.4,0.6,0.8,1},ytick={-4,-2,0,2,4,6,8},
xlabel=$t \ (\mathrm{s})$,ylabel=$u$,legend cell align=left,
legend pos=north east]
\addplot[const plot,color=blue,semithick]
file {../symulacje11/u_lambda_0_1.txt};
\addplot[const plot,color=red,semithick,densely dashed]
file {../symulacje11/u_lambda_0_2.txt};
\addplot[const plot,color=green,semithick,densely dashdotted]
```



Rys. 4.3. Przykładowy rysunek funkcji $f(x)$ i $g(x)$ wykonany w języku PGFPLOTS i zapisany w pliku `funkcje.pdf`

```
file {../symulacje11/u_lambda_0_5.txt};
\addplot[const plot,color=kolor4,semithick,densely dashdotdotted]
file {../symulacje11/u_lambda_1.txt};
\addplot[const plot,color=kolor5,semithick,densely dotted]
file {../symulacje11/u_lambda_2.txt};
\legend{$\lambda=0.1$,$\lambda=0.2$,$\lambda=0.5$,$\lambda=1$,$\lambda=2$}
%%2
\nextgroupplot
[xmin=0,xmax=1,ymin=-1.25,ymax=2.25,
xtick={0,0.2,0.4,0.6,0.8,1},ytick={-1,-0.5,0,0.5,1,1.5,2},
xlabel=$t \ (\mathrm{s})$,ylabel={$y^{\mathrm{zad}}, \ y$},
legend cell align=left,legend style={at={(axis cs:0.6,-0.3)},anchor=south west}]
\addplot[const plot,color=gray,thick]
file {../symulacje11/yzad.txt};
\addplot[color=blue,semithick]
file {../symulacje11/y_lambda_0_1.txt};
\addplot[const plot,color=red,semithick,densely dashed]
file {../symulacje11/y_lambda_0_2.txt};
\addplot[const plot,color=green,semithick,densely dashdotted]
file {../symulacje11/y_lambda_0_5.txt};
\addplot[const plot,color=kolor4,semithick,densely dashdotdotted]
file {../symulacje11/y_lambda_1.txt};
\addplot[const plot,color=kolor5,semithick,densely dotted]
file {../symulacje11/y_lambda_2.txt};
\legend{$y^{\mathrm{zad}}$,$\lambda=0.1$,$\lambda=0.2$,
$\lambda=0.5$,$\lambda=1$,$\lambda=2$}
\end{groupplot}
\end{tikzpicture}
\end{figure}

\end{document}
```

Polecenie

```
pdflatex -shell-escape zapisz_pdf_symulacje11.tex
```

zapisuje plik `symulacje11.pdf`. Rysunek dołącza się do dokumentu instrukcją `\includegraphics`. Efekt przedstawiono na rys. 4.4. Zwróćmy uwagę, że do narysowania wyników symulacji dla kolejnych wartości parametru λ zastosowano różne kolory oraz różne style linii (linia ciągła, linia przerywana, itd.). Umożliwia to łatwe rozróżnienie dokumentów na czarno-białym wydruku. Przy wydruku kolorowym oraz dokumentach elektronicznych można zrezygnować ze stosowania różnych stylów linii, do ich rozróżnienia wystarczające są kolory, pod warunkiem jednak, że kolejne krzywe nie są położone bardzo blisko siebie.

Pakiet `PGFPLOTS` można skonfigurować w taki sposób, aby na osiach stosowane były liczby z przecinkiem dziesiętnym w miejsce kropki dziesiętnej [1].

Czasami ze względu na ograniczoną objętość dokumentu należy zmniejszyć rysunki. Aby zmniejszyć objętość można zastosować ułożenie poziome dwóch rysunków, prezentujących wyniki symulacji procesu. Przygotowano plik `zapisz_pdf_symulacje11_wersja2.tex`, umożliwiający zapisanie rysunku do pliku `symulacje11_wersja2.pdf`. Znajduje się on w katalogu `rysunki/zapisz_pdf` i ma następującą postać:

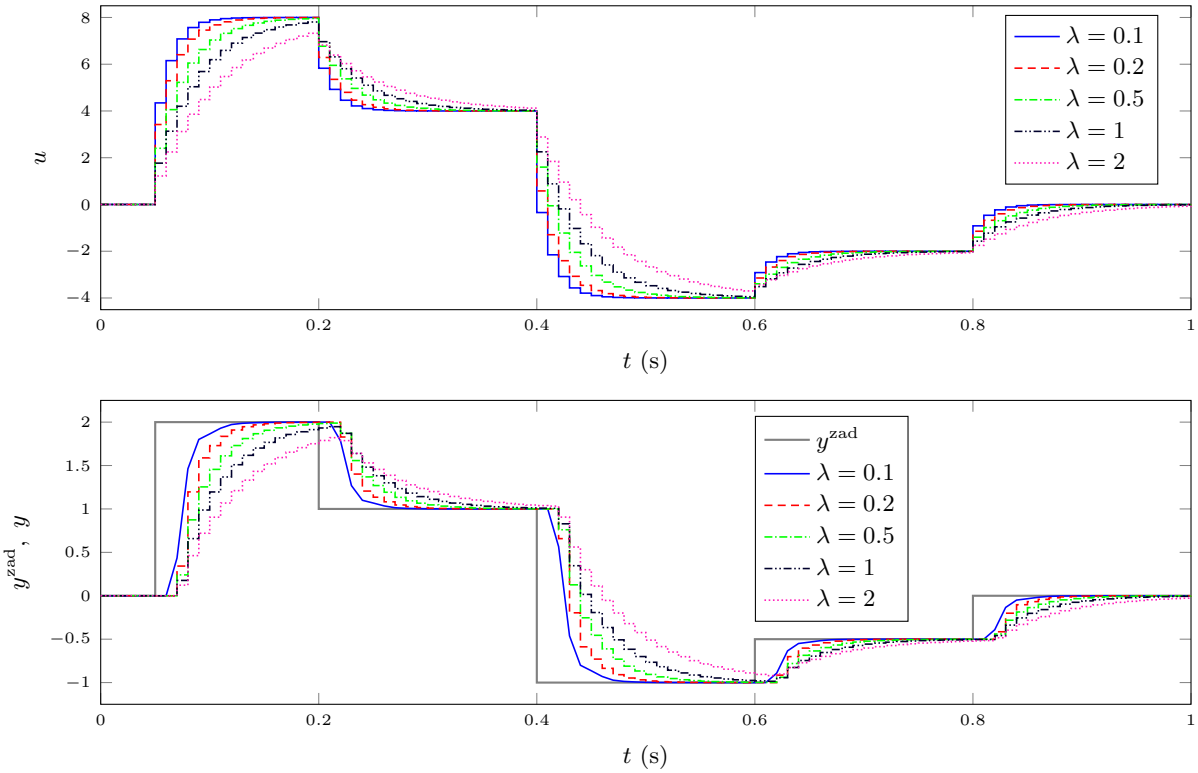
```
\documentclass[a4paper,11pt]{article}
\usepackage{pgfplots}
\usetikzlibrary{pgfplots.groupplots}
\pgfplotsset{compat=1.11}
\usepgfplotslibrary{external}
\tikzexternalize

\textwidth 160mm \textheight 247mm

\pgfplotsset{width=\figurewidth,compat=1.11}
```

```
\pgfplotsset{
tick label style={font=\tiny},
label style={font=\footnotesize},
legend style={font=\footnotesize},
title style={font=\footnotesize}
}

\newcommand{\szer}{8cm}
\newcommand{\wys}{5.6cm}
\newcommand{\odstepoziomy}{1.9cm}

\definecolor{kolor4}{rgb}{0,0,0.1724}
\definecolor{kolor5}{rgb}{1.0000,0.1034,0.7241}
\definecolor{kolor6}{rgb}{1.0000,0.8276,0}

\begin{document}

\tikzsetnextfilename{}

\begin{figure}[tb]
\tikzsetnextfilename{symulacje11_wersja2}
\begin{tikzpicture}
\begin{groupplot}[group style={group size=2 by 1,horizontal sep=\odstepoziomy},
width=\szer,height=\wys]
%%1
\nextgroupplot
[xmin=0,xmax=1,ymin=-4.5,ymax=8.5,
xtick={0,0.2,0.4,0.6,0.8,1},ytick={-4,-2,0,2,4,6,8},
xlabel=$t \ (\mathrm{s})$,ylabel=$u$,,legend style={at={(0,1.15)},anchor=west},
legend columns=6,legend style={/tikz/every even column/.append style=
{column sep=1.4cm}}]
\addplot[const plot,color=blue,semithick]
file {../symulacje11/u_lambda_0_1.txt};
\addplot[const plot,color=red,semithick,densely dashed]
file {../symulacje11/u_lambda_0_2.txt};
\addplot[const plot,color=green,semithick,densely dashdotted]
file {../symulacje11/u_lambda_0_5.txt};
\addplot[const plot,color=kolor4,semithick,densely dashdotdotted]
file {../symulacje11/u_lambda_1.txt};
\addplot[const plot,color=kolor5,semithick,densely dotted]
file {../symulacje11/u_lambda_2.txt};
\legend{$\lambda=0.1$,$\lambda=0.2$,$\lambda=0.5$,$\lambda=1$,$\lambda=2$}
%%2
\nextgroupplot
[xmin=0,xmax=1,ymin=-1.25,ymax=2.25,
xtick={0,0.2,0.4,0.6,0.8,1},ytick={-1,-0.5,0,0.5,1,1.5,2},
xlabel=$t \ (\mathrm{s})$,ylabel={$y^{\mathrm{zad}}, \ y$},
legend cell align=left]
\addplot[const plot,color=gray,thick] file {../symulacje11/yzad.txt};
\addplot[color=blue,semithick] file {../symulacje11/y_lambda_0_1.txt};
\addplot[const plot,color=red,semithick,densely dashed]
file {../symulacje11/y_lambda_0_2.txt};
\addplot[const plot,color=green,semithick,densely dashdotted]
file {../symulacje11/y_lambda_0_5.txt};
\addplot[const plot,color=kolor4,semithick,densely dashdotdotted]
file {../symulacje11/y_lambda_1.txt};
\addplot[const plot,color=kolor5,semithick,densely dotted]
file {../symulacje11/y_lambda_2.txt};
\legend{$y^{\mathrm{zad}}$}
\end{groupplot}
\end{tikzpicture}
\end{figure}
```

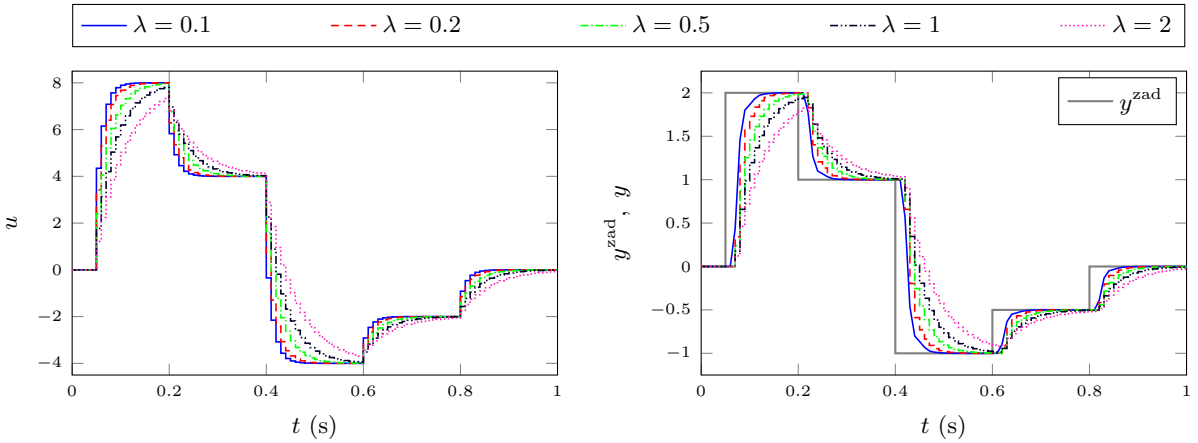
```
\end{document}
```

Rysunek dołącza się do dokumentu instrukcją `\includegraphics`. Efekt przedstawiono na rys. 4.5.

Załóżmy, że w katalogu `rysunki/symulacje22` znajdują się wyniki symulacji procesu o dwóch wejściach i dwóch wyjściach zapisane w plikach: `u1.txt`, `u2.txt`, `y1.txt`, `y2.txt`, `yzad1.txt`, `yzad2.txt`. W pierwszej kolumnie tych plików podano czas t (w sekundach), natomiast w drugiej kolumnie wartość odpowiedniej zmiennej. Plik `zapisz_pdf_symulacje22.tex`, umożliwiający zapisanie rysunku do pliku `symulacje22.pdf` znajduje się w katalogu `rysunki/zapisz_pdf` i ma następującą postać:



Rys. 4.4. Przykładowy rysunek wyników symulacji procesu jednowymiarowego wykonany w języku PGFPLOTS i zapisany w pliku `symulacje11.pdf`



Rys. 4.5. Przykładowy rysunek wyników symulacji procesu jednowymiarowego wykonany w języku PGFPLOTS i zapisany w pliku `symulacje11_wersja2.pdf`

```
\documentclass[a4paper,11pt]{article}
\usepackage{pgfplots}
\usetikzlibrary{pgfplots.groupplots}
\pgfplotsset{compat=1.11}
\usepgfplotslibrary{external}
\tikzexternalize

\textwidth 160mm \textheight 247mm

\pgfplotsset{width=\figurewidth,compat=1.11}
\pgfplotsset{
tick label style={font=\tiny},
label style={font=\footnotesize},
legend style={font=\footnotesize},
title style={font=\footnotesize}
}

\newcommand{\szer}{16cm}
\newcommand{\wys}{6cm}
\newcommand{\odstepionowy}{1.2cm}

\begin{document}

\tikzsetnextfilename{}

\begin{figure}[tb]
\tikzsetnextfilename{symulacje22}
\begin{tikzpicture}
\begin{groupplot}[group style={group size=1 by 4,vertical sep=\odstepionowy},
width=\szer,height=\wys]
%%1
\nextgroupplot
[xmin=0,xmax=1,ymin=-0.1,ymax=3,
xtick={0,0.2,0.4,0.6,0.8,1},ytick={0,1,2,3},
xlabel=$t \ (\mathrm{s})$,ylabel=$u_1$,legend cell align=left,
legend pos=north east]
\addplot[const plot,color=blue,semithick] file {../symulacje22/u1.txt};
%%2
\nextgroupplot
[xmin=0,xmax=1,ymin=-0.5,ymax=1.5,
xtick={0,0.2,0.4,0.6,0.8,1},ytick={-0.5,0,0.5,1,1.5},
xlabel=$t \ (\mathrm{s})$,ylabel=$u_2$,legend cell align=left,
legend pos=north east]
\addplot[const plot,color=blue,semithick] file {../symulacje22/u2.txt};
%%3
\nextgroupplot
[xmin=0,xmax=1,ymin=-0.25,ymax=1.5,
xtick={0,0.2,0.4,0.6,0.8,1},ytick={0,0.5,1,1.5},
xlabel=$t \ (\mathrm{s})$,ylabel={$y_1^{\mathrm{zad}}, \ y_1$},
legend cell align=left,legend pos=north east]
\addplot[const plot,color=gray,thick] file {../symulacje22/yzad1.txt};
\addplot[color=blue,semithick] file {../symulacje22/y1.txt};
\legend{$y_1^{\mathrm{zad}}$,$y_1$}
%%4
\nextgroupplot
[xmin=0,xmax=1,ymin=-0.25,ymax=1.5,
xtick={0,0.2,0.4,0.6,0.8,1},ytick={0,0.5,1,1.5},
xlabel=$t \ (\mathrm{s})$,ylabel={$y_2^{\mathrm{zad}}, \ y_2$},
legend cell align=left,legend pos=north east]
\addplot[const plot,color=gray,thick] file {../symulacje22/yzad2.txt};
\addplot[color=blue,semithick] file {../symulacje22/y2.txt};
\legend{$y_2^{\mathrm{zad}}$,$y_2$}
\end{groupplot}
```

```
\end{tikzpicture}
\end{figure}

\end{document}
```

Rysunek dołącza się do dokumentu instrukcją `\includegraphics`. Efekt przedstawiono na rys. 4.6.

Aby zmniejszyć objętość można nieco inaczej ułożyć 4 rysunki, prezentujące wyniki symulacji procesu dwuwymiarowego. Przygotowano plik `zapisz_pdf_symulacje2_wersja2.tex`, umożliwiający zapisanie rysunku do pliku `symulacje22_wersja2.pdf`. Znajduje się on w katalogu `rysunki/zapisz_pdf` i ma następującą postać:

```
\documentclass[a4paper,11pt]{article}
\usepackage{pgfplots}
\usetikzlibrary{pgfplots.groupplots}
\pgfplotsset{compat=1.11}
\usepgfplotslibrary{external}
\tikzexternalize

\textwidth 160mm \textheight 247mm

\pgfplotsset{width=\figurewidth,compat=1.11}
\pgfplotsset{
tick label style={font=\tiny},
label style={font=\footnotesize},
legend style={font=\footnotesize},
title style={font=\footnotesize}
}

\newcommand{\szer}{8cm}
\newcommand{\wys}{5.6cm}
\newcommand{\odstepionowy}{1.2cm}
\newcommand{\odstepoziomy}{1.9cm}

\begin{document}

\tikzsetnextfilename{}

\begin{figure}[tb]
\tikzsetnextfilename{symulacje22_wersja2}
\begin{tikzpicture}
\begin{groupplot}[group style={group size=2 by 2,horizontal sep=\odstepoziomy,
vertical sep=\odstepionowy},width=\szer,height=\wys]
%%1
\nextgroupplot
[xmin=0,xmax=1,ymin=-0.1,ymax=3,
xtick={0,0.2,0.4,0.6,0.8,1},ytick={0,1,2,3},
xlabel=$t \ (\mathrm{s})$,ylabel=$u_1$,legend cell align=left,
legend pos=north east]
\addplot[const plot,color=blue,semithick] file {../symulacje22/u1.txt};
%%2
\nextgroupplot
[xmin=0,xmax=1,ymin=-0.25,ymax=1.5,
xtick={0,0.2,0.4,0.6,0.8,1},ytick={0,0.5,1,1.5},
xlabel=$t \ (\mathrm{s})$,ylabel={$y_1^{\mathrm{zad}}, \ y_1$},
legend cell align=left,legend pos=south east]
\addplot[const plot,color=gray,thick] file {../symulacje22/yzad1.txt};
\addplot[color=blue,semithick] file {../symulacje22/y1.txt};
\legend{$y_1^{\mathrm{zad}}$,$y_1$}
%%3
\nextgroupplot
[xmin=0,xmax=1,ymin=-0.5,ymax=1.5,
xtick={0,0.2,0.4,0.6,0.8,1},ytick={-0.5,0,0.5,1,1.5},
```

```
xlabel=$t \ (\mathrm{s})$,ylabel=$u_2$,legend cell align=left,
legend pos=north east]
\addplot[const plot,color=blue,semithick] file {../symulacje22/u2.txt};
%%4
\nextgroupplot
[xmin=0,xmax=1,ymin=-0.25,ymax=1.5,
xtick={0,0.2,0.4,0.6,0.8,1},ytick={0,0.5,1,1.5},
xlabel=$t \ (\mathrm{s})$,ylabel={$y_2^{\mathrm{zad}}, \ y_2$},
legend cell align=left,legend pos=south east]
\addplot[const plot,color=gray,thick] file {../symulacje22/yzad2.txt};
\addplot[color=blue,semithick] file {../symulacje22/y2.txt};
\legend{$y_2^{\mathrm{zad}}$,$y_2$}
\end{groupplot}
\end{tikzpicture}
\end{figure}

\end{document}
```

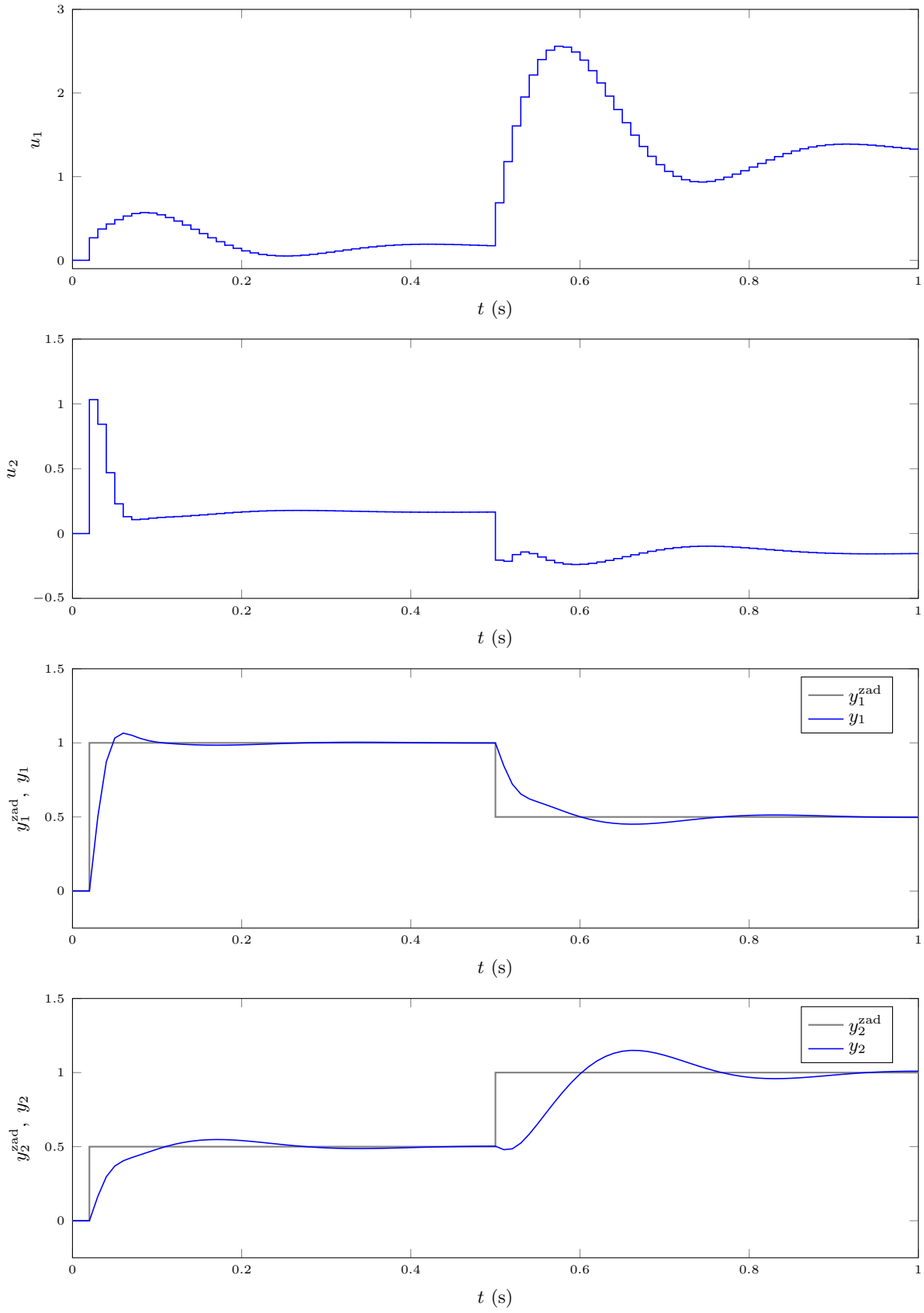
Rysunek dołącza się do dokumentu instrukcją `\includegraphics`. Efekt przedstawiono na rys. 4.7.

W katalogu `rysunki/symulacje43` znajdują się wyniki symulacji procesu o czterech wejściach i trzech wyjściach zapisane w plikach: `u1.txt`, `u2.txt`, `u3.txt`, `u4.txt`, `y1.txt`, `y2.txt`, `y3.txt`, `y1zad.txt`, `y2zad.txt`, `y3zad.txt`. W pierwszej kolumnie tych plików podano czas t (w sekundach), natomiast w drugiej kolumnie wartość odpowiedniej zmiennej. W katalogu `rysunki/zapisz_pdf` znajduje się plik `zapisz_pdf_symulacje43.tex`, który zapisuje dwa pliki zawierające wyniki symulacji: `symulacje43u.pdf` (sygnały sterujące) oraz `symulacje43y.pdf` (sygnały wartości zadanych oraz sygnały wyjść procesu). Oba rysunki dołącza się do dokumentu dwoma instrukcjami `\includegraphics`, między rysunkami wstawiono niewielki odstęp. Efekt przedstawiono na rys. 4.8.

4.4. Kolory

Przy umieszczaniu kilku wykresów na tym samym rysunku należy zastosować kolory różniące się od siebie w znacznym stopniu, nie można stosować kolorów podobnych, np. kilku odcieni tego samego koloru. Do generacji palety kolorów spełniającej takie wymagania można użyć funkcji `distinguishable_colors.m`, udostępnionej na stronie `https://www.mathworks.com/matlabcentral/`. Zestaw dwudziestu kolorów został przedstawiony na rys. 4.9. Ich definicja w palecie RGB jest następująca:

```
         0         0    1.0000
    1.0000         0         0
         0    1.0000         0
         0         0    0.1724
    1.0000    0.1034    0.7241
    1.0000    0.8276         0
         0    0.3448         0
    0.5172    0.5172    1.0000
    0.6207    0.3103    0.2759
         0    1.0000    0.7586
         0    0.5172    0.5862
         0         0    0.4828
    0.5862    0.8276    0.3103
    0.9655    0.6207    0.8621
    0.8276    0.0690    1.0000
    0.4828    0.1034    0.4138
    0.9655    0.0690    0.3793
    1.0000    0.7586    0.5172
    0.1379    0.1379    0.0345
    0.5517    0.6552    0.4828
```

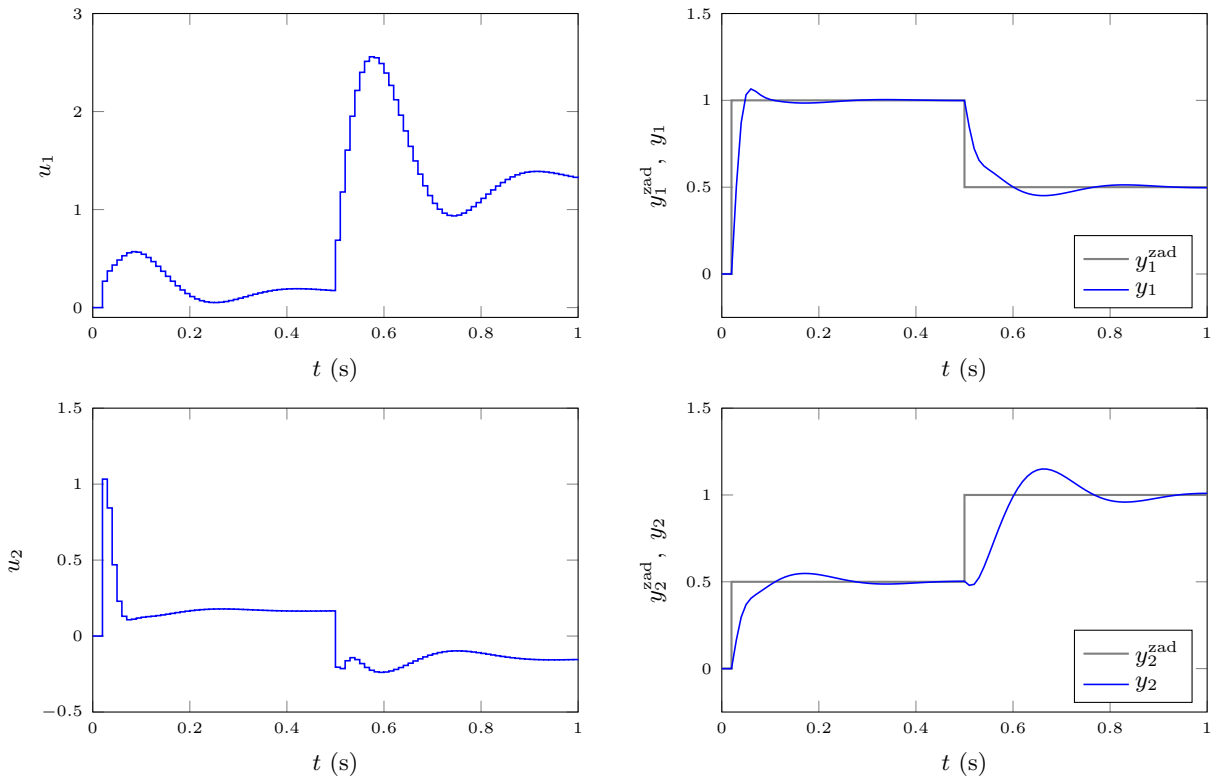


Rys. 4.6. Przykładowy rysunek wyników symulacji procesu dwuwymiarowego wykonany w języku PGFPLOTS i zapisany w pliku `symulacje22.pdf`

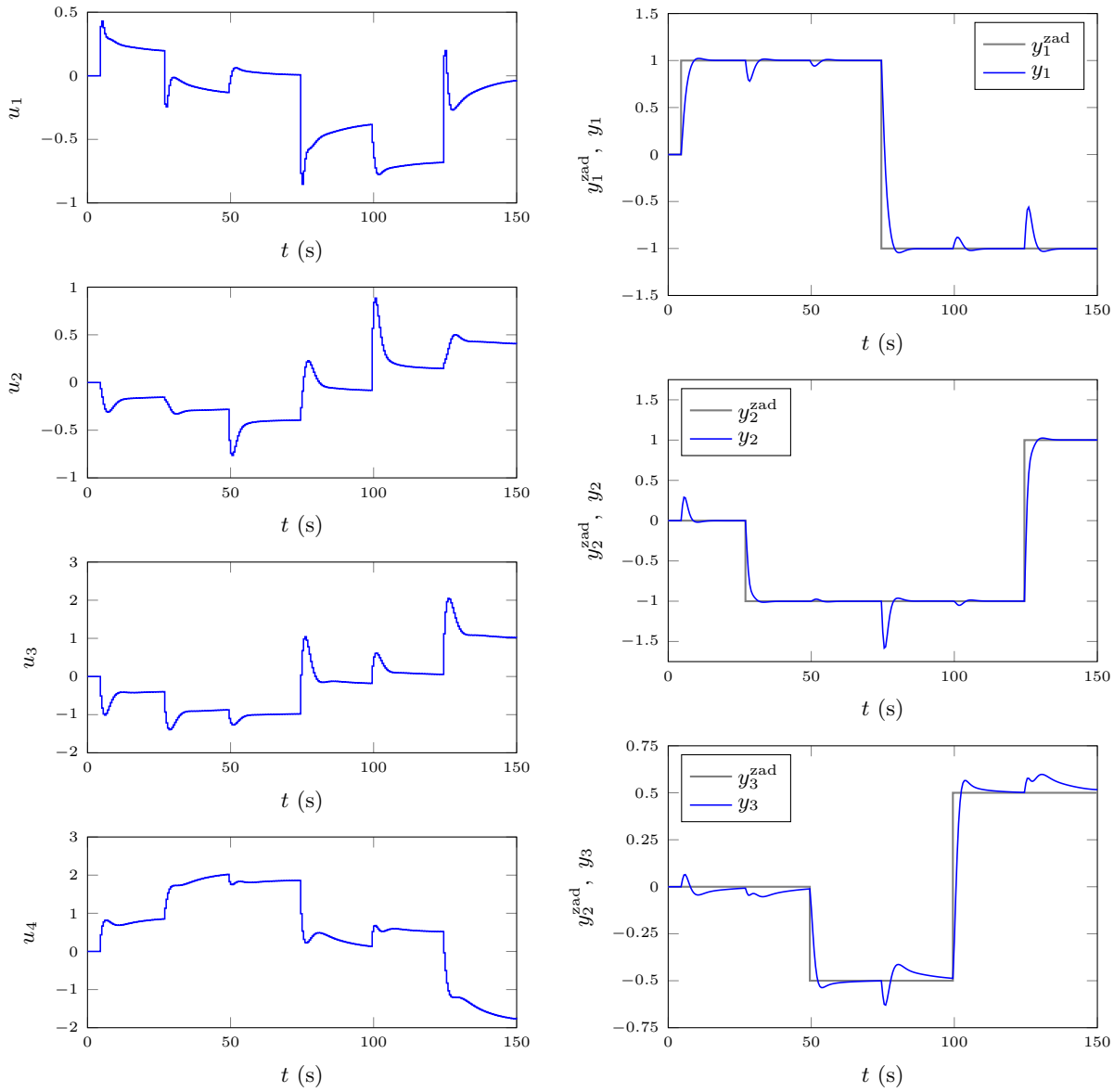
Oprócz kolorów standardowych oraz dodatkowych, które są zdefiniowane w pakiecie `xcolor`, własne kolory definiujemy poleceniem `\definecolor`. Na przykład, piąty kolor z zestawu definiujemy poleceniem `\definecolor{kolor5}{rgb}{1.0000,0.1034,0.7241}`, co umożliwia osiągnięcie następującego efektu.

4.5. Lokalizacja rysunków (i tabel)

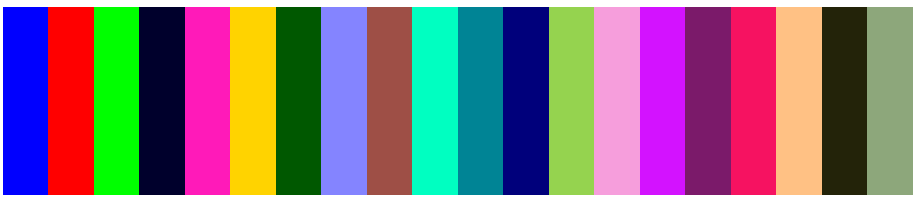
Wszystkie rysunki i tabele umieszczamy jako „pływające”, a więc w otoczeniu `figure` oraz `table` nie stosujemy opcji `[h]` i `[H]`. Aby uniknąć umieszczenia rysunków w tekście kolejnego rozdziału należy zastosować polecenie `\FloatBarrier` z pakietu `placeins`.



Rys. 4.7. Przykładowy rysunek wyników symulacji procesu dwuwymiarowego wykonany w języku PGFPLOTS i zapisany w pliku `symulacje22_wersja2.pdf`



Rys. 4.8. Przykładowy rysunek wyników symulacji procesu o czterech wejściach i trzech wyjściach wykonany w języku `PGFPLOTS` i zapisany w plikach `symulacje43u.pdf` oraz `symulacje43y.pdf`



Rys. 4.9. Paleta 20 kolorów

5. Listingi programów

Do zamieszczenia programów można zastosować otoczenie `verbatim`, ale znacznie większe możliwości oferuje pakiet `listings`. Przykładowy program w języku C ma postać:

```c
//Hello World in C
#include <stdio.h>
int main (void)
{
   puts ("Hello World!");
   return 0;
}
```

natomiast przykładowy program w języku `Matlab` jest następujący:

```
%Hello World in Matlab
clear all;

disp('Hello World!');
```

Bibliografia

[1] C. Feuersänger: Manual for package PGFPLOTS. ftp://ftp.gust.org.pl/TeX/graphics/pgf/contrib/pgfplots/doc/pgfplots.pdf, 2016.

[2] T. Oetiker, H. Partl, I. Hyna, E. Schlegl: Nie za krótkie wprowadzenie do systemu LaTeX 2_ε. ftp://ftp.gust.org.pl/pub/CTAN/info/lshort/polish/lshort2e.pdf, 2007.

[3] T. Tantau: The TikZ and PGF Packages Manual for version 3.0.1a. ftp://ftp.gust.org.pl/TeX/graphics/pgf/base/doc/pgfmanual.pdf, 2015.

[4] M. Woliński: Moje Własne CLaSy dokumentów dla LaTeXa 2e http://marcinwolinski.pl/mwcls.html, 2013.

[5] Wikibooks: LaTeX. https://en.wikibooks.org/wiki/LaTeX, 2017.