Michał ŁASKAWSKI[1]

Politechnika Świętokrzyska, Zakład Urządzeń i Systemów Automatyki (1)

doi:10.15199/48.2017.01.07

Dobór nastaw regulatorów ciągłych z wykorzystaniem środowiska Scilab / Xcos

Streszczenie. *W pracy przedstawiono sposoby implementacji wybranych algorytmów doboru nastaw regulatorów ciągłych typu PI oraz PID przy pomocy środowiska SCILAB/XCOS. Użyto metod: Ziegler-Nicholsa, autorskiej oraz wykorzystującej wybrane metody optymalizacji. Przedstawiono i porównano wyniki symulacji.*

Abstract*. The paper deals with implementation methods of algorithms of tuning of continuous controllers using SCILAB/XCOS environment. Following procedures are shown: Ziegler-Nichols, authorial and the procedure using chosen optimization method. Simulation results are shown and compared. (**The tuning method of continuous controllers using SCILAB/XCOS environment**).*

Słowa kluczowe: regulator, PID, nastawy, optymalne.
Keywords: controller, PID, settings, optimal.

Wstęp

Środowisko SCILAB wyposażone jest w rozbudowany interfejs użytkownika i nowoczesny język programowania. Jego uzupełnienie stanowi program XCOS, umożliwiający graficzną syntezę schematów blokowych definiujących rozważany problem. Szeroka gama bibliotek, która dołączana jest do programu, zawiera algorytmy numeryczne z różnych dziedzin wiedzy. Umożliwia to zminimalizowanie czasu pracy, który potrzebny jest na implementację wybranego problemu, jak również ułatwia analizę danego problemu.

Celem pracy jest sprawdzenie możliwości zastosowania środowiska SCILAB/XCOS do symulacji pracy układów regulacji a w szczególności łatwej możliwości implementacji algorytmów doboru nastaw regulatorów ciągłych i szybkiej wizualizacji uzyskanych wyników.

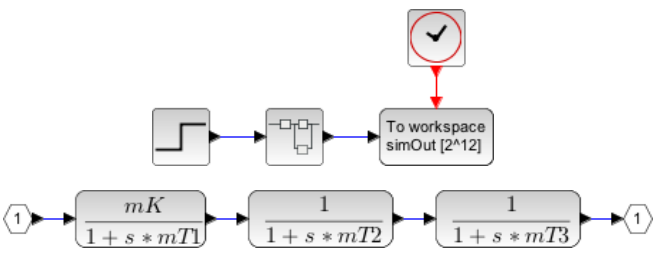
Dobór nastaw według metody Zieglera-Nicholsa

Jednymi z najczęściej wykorzystywanych metod doboru nastaw regulatorów ciągłych są metody Zieglera – Nicholsa (ZN) [1]. Jedną z nich, jest metoda próby skokowej. Polega ona na otwarciu pętli regulacji i podaniu wymuszenia skokowego na wejście obiektu regulacji. Na podstawie odpowiedzi obiektu na podane wymuszenie, identyfikuje się wartości współczynników modelu aproksymującego. Współczynniki te są parametrami stabelaryzowanych zależności, które określają nastawy regulatorów PI(D).

Jako obiekt regulacji przyjęto model w postaci członu inercyjnego trzeciego rzędu:

$$G_{(s)} = 1/(s+1)^3 \tag{1}$$

Na rysunku 1, przedstawiono otwarty układ sterowania, wykorzystany do identyfikacji parametrów modelu aproksymującego. Układ ten zrealizowano przy użyciu programu XCOS.



Rys. 1. Otwarty układ sterowania i podsystem definiujący obiekt regulacji

Na wejście podsystemu, podawany jest sygnał wymuszenia skokowego o zdefiniowanej amplitudzie. Odpowiedź modelu obiektu regulacji, przekazywana jest do bloku, którego zadaniem jest eksport danych do przestrzeni roboczej środowiska SCILAB. W celu wykonania symulacji, utworzono skrypt startowy, który na początku definiuje parametry bloków przedstawionych na rysunku 1.

```
simDt = 0.01;  // okres całkowania
sp = 1;        // wartość zadana
// parametry modelu
mK = 1; mT1 = 1; mT2 = 1; mT3 = 1;
```

Załadowanie schematu blokowego (rys. 1) programu XCOS do przestrzeni roboczej środowiska SCILAB, definicja czasu symulacji i symulacja w trybie wsadowym realizowana jest przy pomocy instrukcji:

```
// import schematu
importXcosDiagram('ukladOtwarty.zcos');
scs_m.props.tf = 25;          // czas symulacji
xcos_simulate(scs_m, 4);      // symulacja
```

Po zakończeniu symulacji do przestrzeni roboczej eksportowana jest zmienna `simOut` (rys. 1). Jest to struktura zawierająca dwa pola. Są nimi wektor czasu i macierz wartości przechowująca żądany sygnał wyjściowy.

```
t = simOut.time;      y = simOut.values;
```

Do identyfikacji parametrów (`tanL` i `tanB`) modelu aproksymującego, konieczne jest wyznaczenia stycznej do uzyskanej odpowiedzi w punkcie jej przegięcia. Wymaga to wyznaczenia pochodnej sygnału po czasie. W tym celu zdefiniowano funkcję:

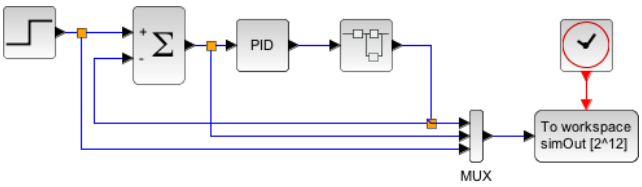
```
function [dx] = compDerivative(x,h)
   dx(1)=((-3/2)*x(1)+2*x(2)+(-1/2)*x(3))/h;
   for i=2:length(x)-1
        dx(i)=((-1/2)*x(i-1)+(1/2)*x(i+1))/h;
   end
   dx(i+1)=((3/2)*x($)-2*x($-1)+(1/2)*x($-2))/h;
endfunction
```

Styczną, parametry modelu aproksymującego i nastawy regulatorów wyznaczono przy pomocy instrukcji:

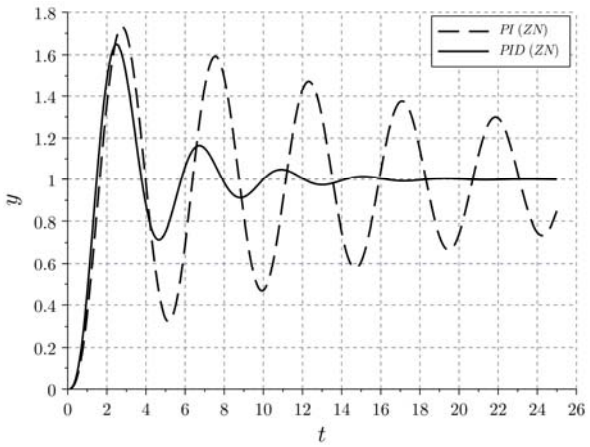
```
dy = compDerivative(y, simDt);
[tanA,id] = max(dy);
tanL = tanA .* t;
apxB = tanL(id) - y(id);
```

```
apxL = apxB / tanA;
KpPI = 0.9 / apxA; TiPI = 3*apxL;
KpPID = 1.2 / apxA; TiPID = 2*apxL;
TdPID = apxL/2;
```

Wykorzystując model przedstawiony na rysunku 2 oraz korzystając funkcji `plot2d()`, uzyskano przebiegi regulacji przedstawione na rysunku 3.



Rys. 2. Schemat programu XCOS zamkniętego układu regulacji z regulatorem PID i wybranym modelem inercyjnym.



Rys. 3. Odpowiedzi układów regulacji z parametrami regulatora określonymi metodą ZN.

Przedstawiona w pracy [1], częstotliwościowa metoda doboru nastaw regulatorów PI lub PID wymaga doprowadzenia zamkniętego układu regulacji do granicy stabilności. Może to być niebezpieczne dla układu sterowania. W przypadku gdy obiekt regulacji charakteryzuje się niskim rzędem, samo doprowadzenie układu regulacji do granicy stabilności może być niemożliwe. Wady te eliminuje procedura zaproponowana w pracy [2]. Procedura ta zakłada wykorzystanie regulatora dwustanowego do uzyskania oscylacji wielkości regulowanej w kontrolowanych warunkach. W pierwszym etapie, na podstawie znanej amplitudy regulatora dwustanowego d i zmierzonej amplitudy sygnału sterowanego a, aproksymowany jest współczynnik wzmocnienia krytycznego:

(2) $$k_{kr} = 4\,d / \pi\, a$$

Nastawy dla regulatora PID określono zależnościami [3]:

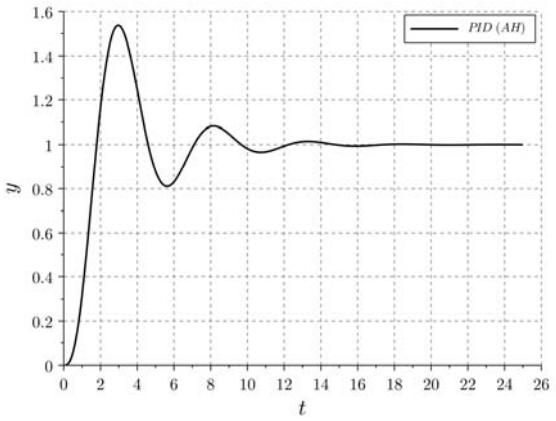
(3) $$K_p = k_{(kr)} r_b \cos(\varphi_b)$$
$$T_i = \frac{T_u}{\pi}\left(\frac{1+\sin(\varphi_b)}{\cos(\varphi_b)}\right) \qquad T_d = \frac{T_u}{4\pi}\left(\frac{1+\sin(\varphi_b)}{\cos(\varphi_b)}\right)$$

gdzie: r_b oraz φ_b określają żądane położenie wybranego punktu na charakterystyce Nyquista, T_u jest okresem oscylacji drgań wielkości regulowanej.

Wyznaczenie amplitudy drgań wielkości regulowanej *a* oraz okresu oscylacji T_u, dla układu z regulatorem dwustanowym, w języku SCILAB zrealizowano przy pomocy funkcji:

```
function [Tu, Aosc] = getTuAosc(t, e, y)
   de=compDerivative(e, t(2)-t(1));
   n=length(t);
   e1=e(1:n-1); e2=e(2:n); ee=e1.*e2;
   idx=find(ee<0); j=1;
   for i=1:length(idx)
        if de(idx(i)) > 0 then
               v(j)=t(idx(i)); j=j+1;
        end
   end
   Tu = mean(diff(v)); // okres
   e1=de(1:n-1); e2=de(2:n); ee=e1.*e2;
   idx=find(ee<0); j=1; m=1;
   for i=1:length(idx)
        if e(idx(i)) > 0 then
               minY(j) = y(idx(i)); j = j + 1;
        end
        if e(idx(i)) < 0 then
               maxY(m) = y(idx(i)); m = m + 1;
        end
   end
   Aosc = mean(maxY-minY);    // amplituda
endfunction
```

Wzmocnienie krytyczne oszacowano przy pomocy zależności (2). Wykorzystując przedstawioną powyżej procedurę oraz równania (3), określono nastawy regulatora PID. Na rysunku 4, przedstawiono przebieg przejściowy odpowiedzi układu regulacji z rysunku 2 dla wyznaczonych nastaw regulatora PID.



Rys. 4. Odpowiedzi układu regulacji z parametrami regulatora określonymi metodą AH.

Przedstawione wykresy (rys. 3 i 4), charakteryzują się dużymi przeregulowaniami i czasem ustalania. Dotyczy to zwłaszcza układu z regulatorem PI (rys 3).

Dobór nastaw z wykorzystaniem kryterium ITAE

W pracy [3], przedstawiono częstotliwościową metodę doboru nastaw regulatorów PID o nastawach optymalnych względem kryterium ITAE.

(4) $$ITAE = \int_0^{\infty} t\left|e_{(t)}\right| dt$$

Metoda ta polega na wyłączeniu członów: proporcjonalnego i różniczkującego regulatora PID a następnie podaniu sygnału sterującego tak powstałego regulatora całkującego na wejście obiektu regulacji. Taki układ obejmowany jest pętlą ujemnego sprzężenia zwrotnego i poprzez zmianę współczynnika wzmocnienia regulatora całkującego, doprowadzany jest na granicę stabilności. W tym stanie odczytywane są: wzmocnienie krytyczne regulatora całkującego $k_{i(kr)}$ oraz częstotliwość oscylacji ω_{osc}. Na podstawie tych parametrów

identyfikowana jest stała czasowa modelu aproksymującego obiekt regulacji T oraz współczynnik $\theta_\circ$.

$$(5)\quad T = \frac{\sqrt{(k_{i(kr)}k)^2 - \omega_{osc}^2}}{\omega_{osc}^2} \qquad \omega_{\circ osc} = \omega_{osc}T \qquad \theta_\circ = \frac{0{,}5\pi - arctg(\omega_{\circ osc})}{\omega_{\circ osc}}$$

Optymalne względem kryterium (4), nastawy regulatorów PI oraz PID, określają zależności:

$$(6)\quad \frac{K_{p(PI)}}{k} = 10^{\frac{0{,}49}{\sqrt{\theta_\circ}} - 0{,}67} \qquad T_{i(PI)} \cdot T = 0{,}0058\theta_\circ^2 + 0{,}31\theta_\circ + 0{,}91$$

$$\frac{K_{p(PID)}}{k} = 10^{-0{,}79 + \frac{0{,}81}{\sqrt[3]{\theta}}}$$

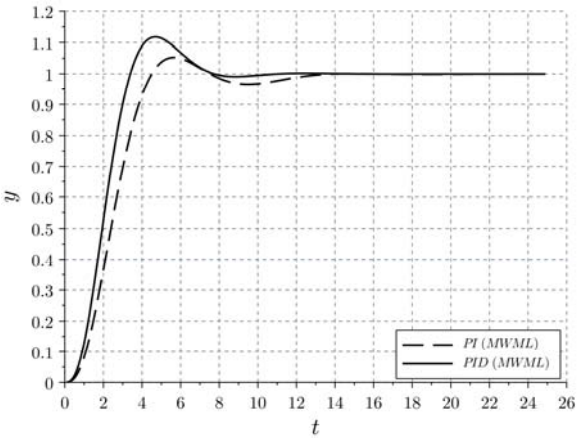
$$T_{i(PID)} \cdot T = 0{,}4\theta_\circ + 0{,}97 \qquad T_{d(PID)} \cdot T = 0{,}48\sqrt{\theta_\circ} - 0{,}16$$

gdzie: *k* jest wzmocnieniem statycznym modelu obiektu aproksymującego.

Implementacja procedury w języku SCILAB rozpoczyna się od wyznaczenia wzmocnienia krytycznego regulatora całkującego *ki(kr)*:

```
s = poly(0,"s");
// trans. ukl. otw. z Reg. I
Go = syslin('c', mK / (s*(s+1)^3));
am = 1/(10^(g_margin(Go)/20));
// wzmocnienie krytyczne
kkr = 1/am;
```

Następnie, korzystając z funkcji: `getTuAosc()`, wyznaczono amplitudę i okres oscylacji. Na podstawie zależności (5) i (6), określono nastawy regulatorów PI oraz PID. Przebiegi przejściowe sygnałów sterowanych dla wyznaczonych nastaw regulatorów przedstawiono na rysunku 5.



Rys. 5. Wyniki symulacji układu regulacji z parametrami regulatora określonymi według metody uwzględniającej kryterium ITAE.

Porównując uzyskane wyniki z wykresami na rysunkach: 3 i 4, widać że jakość regulacji uległa znaczącej poprawie.

Dobór nastaw z wykorzystaniem metod optymalizacji

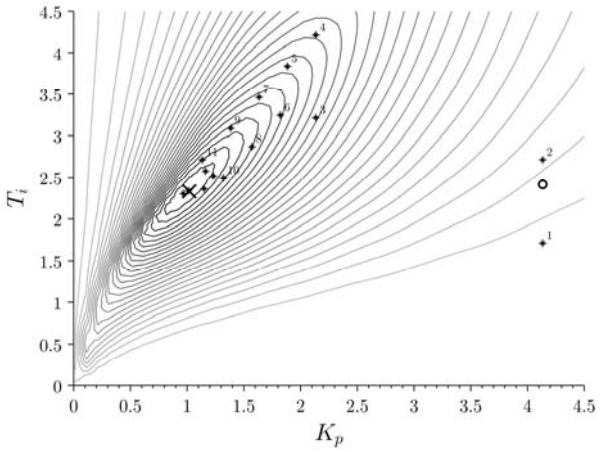
W przypadku gdy znany jest dokładny model obiektu regulacji, można wykorzystać metody optymalizacji do określenia optymalnych względem wybranego kryterium nastaw parametrów regulatorów PI oraz PID. Metody te wymagają zdefiniowana funkcji kosztu, która określa żądane kryterium optymalności. Poniżej przedstawiono implementację takiej funkcji w języku SCILAB dla regulatora PI. Funkcja ta jako kryterium optymalności definiuje całkowe kryterium ITAE (4). Do jej wykorzystania niezbędne jest załadowanie do przestrzeni roboczej środowiska SCILAB, schematu z rysunku 2 jak również zdefiniowanie zmiennych opisujących parametry modelu, wartość zadaną i okres całkowania.

```
function [f, index] = piCostFunction(x, index)
   cKp = x(1); cKi = cKp/x(2); cKd = 0;
   xcos_simulate(scs_m, 4);
   t = simOut.time; e = simOut.values(:,2);
   f = inttrap(t, t.*abs(e));
endfunction
```

Dla niewielkiej liczby zmiennych, jak w przypadku parametrów regulatora PI, do identyfikacji ich wartości można użyć metody Neldera-Meada. Jako punkt początkowy x0, wybrano nastawy uzyskane przy pomocy metody Zieglera – Nicholsa.

```
x0 = [KpPIzn, KpPIzn/TiPIzn];
nm = neldermead_new();
nm = neldermead_configure(nm,
       "-numberofvariables", 2);
nm = neldermead_configure(nm,"-function",
       piCostFunction);
nm = neldermead_configure(nm,"-x0", x0);
nm = neldermead_configure(nm,"-maxiter", 200);
nm = neldermead_configure(nm,"-maxfunevals",
       300);
nm = neldermead_configure(nm,"-tolfunrelative",
       1e-6);
nm = neldermead_configure(nm,"-tolxrelative",
       1e-6);
nm = neldermead_search(nm);
xopt = neldermead_get(nm,"-xopt");
nm = neldermead_destroy(nm);
```

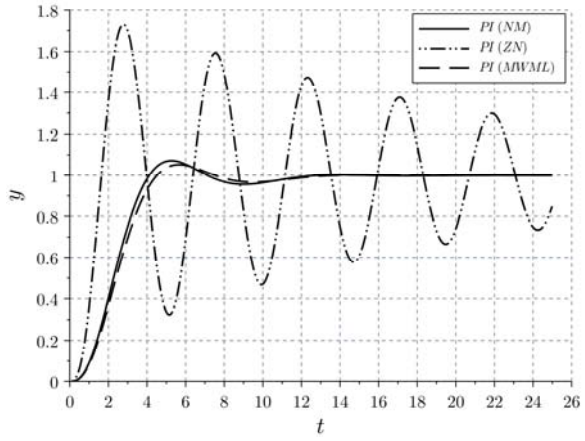
Przedstawiony zestaw instrukcji języka SCILAB tworzy obiekt o nazwie `nm`. Następnie określana jest: liczba zmiennych, funkcja kosztu, warunki początkowe, maksymalna ilość iteracji, maksymalna ilość wywołania funkcji kosztu, dopuszczalny błąd względy wartości funkcji kosztu i zmiany wartości jej parametrów. Rezultaty zwracane są do zmiennej `xopt`. Wyniki pracy metody przedstawiono na rysunku 6.



Rys. 6. Poszczególne etapy pracy algorytmu Neldera – Meada dla układu z regulatorem PI.

Uzyskane wyniki przedstawiono na tle rysunku konturowego określającego wartości wskaźnika *ITAE* w funkcji T_i oraz K_p. Okręgiem zaznaczono punkt startowy natomiast krzyżem wartość optymalną. Ze względu na czytelność ograniczono się do zaznaczenia kilkunastu pierwszych kroków metody (znak gwiazdy). Metoda znajduje, optymalne względem kryterium (4), nastawy regulatora PI w 55 krokach wykonując funkcję celu 110

razy. Na rysunku 7 przedstawiono przebiegi przejściowe uzyskane przy zastosowaniu procedury optymalizacji według metody Neldera-Meada.



Rys. 7. Procesy przejściowe odpowiedzi obiektu układu regulacji uzyskane dla nastaw według autorskiej procedury MWML, procedury ZN oraz optymalizacji NM dla regulatora PI.

Dla regulatora PI, wyniki uzyskane przy pomocy autorskiej metody, nieznacznie tylko różnią się od wyników dla nastaw optymalnych według wybranego kryterium, które otrzymano stosując metodę optymalizacji.

Do identyfikacji parametrów regulatora PID można ponownie użyć metody Neldera – Meada. Można również wykorzystać inne algorytmy dostarczane wraz ze środowiskiem SCILAB. Użycie algorytmu Broydena-Fletchera-Goldfarba-Shannona (BFGS), wymaga modyfikacji funkcji kosztu: `pidCostFunction()`. Nowa funkcja celu: `objectiveFunction()`, musi zwracać wartość funkcji kosztu i jej gradient.

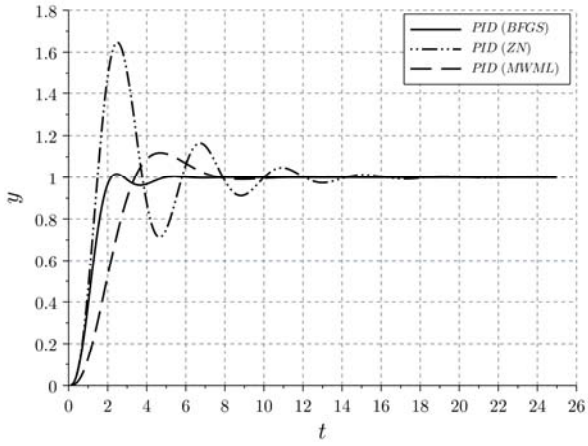
```
function [f] = pidCostFunction(x)
   cKp = x(1); cKi = cKp/x(2); cKd = cKp*x(3);
   xcos_simulate(scs_m, 4);
   t = simOut.time; e = simOut.values(:,2);
   f = inttrap(t,t.*abs(e));
endfunction

function [f, g, ind] = objectiveFunction(x, ind)
   f = pidCostFunction(x);
   g = numderivative(pidCostFunction, x, [], 4);
endfunction
```

Algorytm BFGS dostępny jest po wywołaniu funkcji `optim()`.

```
nap = 100;     // maks. Licz. wywołań f. celu
iter = 100;    // maks. ilość iteracji
epsg = 1e-6;   // dop. próg zmian gradientu
epsf = 1e-6;   // dop. próg zmian f. celu
// dop. wartości zmian parametrów f. c.
epsx = [1e-3 1e-3 1e-3];
[fopt, xopt] = optim(objectiveFunction, x0, "qn",
       "ar", nap, iter, epsg, epsf, epsx);
```

Przyjmując, że punkt początkowy określony jest poprzez nastawy uzyskane przy pomocy metody Zieglera – Nicholsa, algorytm BFGS ze zdefiniowanymi powyżej parametrami uzyskał rozwiązanie w 14 krokach wykonując funkcję celu 39 razy. Uzyskany przebieg regulacji przedstawiono na rysunku 8.



Rys. 8. Procesy przejściowe odpowiedzi obiektu układu regulacji uzyskane dla nastaw według autorskiej procedury MWML, procedury ZN oraz optymalizacji BFGS dla regulatora PID.

Z przedstawionych przebiegów regulacji, wynika że najlepszą jakość regulacji uzyskano dla regulatora PID o nastawach dobieranych przy pomocy metody BFGS.

Podsumowanie

Podobna do języka Matlab składnia języka SCILAB, umożliwia szybkie dostosowanie programów Matlaba do środowiska SCILAB.

Zaawansowane i rozbudowane środowisko graficzne, umożliwia szybką i wygodną prezentację wyników obliczeń.

Środowisko SCILAB wyposażone jest w dodatek XCOS, który wraz z rozbudowaną bazą bibliotek, umożliwia graficzną syntezę złożonych modeli matematycznych. Możliwości środowiska znacząco rozszerza jego możliwa integracja z takimi środowiskami jak Modelica.

Biblioteka CACSD, znacząco ułatwia symulację i analizę obiektów automatyki, a wydajna implementacja sprawdzonych algorytmów optymalizacji numerycznej ułatwia identyfikację parametrów układów sterowania.

Licencja programu, umożliwiająca jego nieodpłatne wykorzystanie oraz otwarte źródła kodu programu predysponują środowisko SCILAB/XCOS do zastosowań dydaktycznych.

***Autorzy**: dr inż. Michał S. Łaskawski, Politechnika Świętokrzyska, Katedra Elektrotechniki Przemysłowej i Automatyki, al. Tysiąclecia Państwa Polskiego 7, 25-413 Kielce, E-mail: michall@tu.kielce.pl;*

LITERATURA

[1] Zeigler J.G, Nichols N.B., Optimum setting for automatic controllers, Rochester N.Y. (1942)

[2] Astrom K.J., Hagglund T., Automatic Tunning of Simple regulators with specifications on chase and amplitude margins, Automatica 20(5) , s.645-651

[3] Wciślik M, Łaskawski M., Sposób doboru nastaw parametrów regulatorów ciągłych typu PI oraz typu PID. Patent numer: PL 222174 B1, 2015