dr inż. Maciej Ławryńczuk
Instytut Automatyki i Informatyki Stosowanej
Politechnika Warszawska

ANALITYCZNY NIELINIOWY ALGORYTM REGULACJI PREDYKCYJNEJ Z MODELAMI NEURONOWYMI

W pracy przedstawiono nieliniowy algorytm regulacji predykcyjnej wykorzystujący modele neuronowe typu perceptronowego MLP (ang. Multi Layer Perceptron). Model neuronowy jest linearyzowany w otoczeniu aktualnego punktu pracy. Aktualna wartość sygnału sterującego wyznaczana jest w sposób analityczny, bez potrzeby optymalizacji. Uzyskane rozwiązanie jest rzutowane na zbiór ograniczeń wartości i szybkości zmian sygnału sterującego. Algorytm jest efektywny obliczeniowo, wymaga jedynie cyklicznej dekompozycji macierzy i rozwiązania dwu równań liniowych. Algorytm charakteryzuje się dużą dokładnością regulacji, porównywalną z algorytmami wymagającymi bieżącej nieliniowej optymalizacji.

AN EXPLICIT NONLINEAR PREDICTIVE CONTROL ALGORITHM BASED ON NEURAL MODELS

This paper describes a nonlinear Model Predictive Control (MPC) algorithm based on MLP (Multi Layer Perceptron) neural models. The neural model is linearised on-line around the current operating point. The value of the manipulated variable is calculated explicitly without any optimisation. The obtained solution is projected onto the admissible set of constraints imposed on the magnitude and the increment of the manipulated variable. The algorithm is computationally efficient. It needs repeating on-line a matrix decomposition task and solving two linear equations. The algorithm gives good closed-loop control performance, comparable to that obtained in nonlinear MPC, which hinges on nonlinear optimisation.

1. WSTĘP

Regulacja predykcyjna (ang. Model Predictive Control, w skrócie MPC) jest jedyną zaawansowaną techniką regulacji, która nie tylko przyciągnęła uwagę teoretyków, ale również znalazła bardzo szerokie zastosowanie praktyczne [10, 17, 18, 19]. Algorytmy regulacji predykcyjnej są powszechnie stosowane do regulacji wielu procesów technologicznych, przede wszystkim w przemyśle chemicznym, petrochemicznym, papierniczym, metalurgicznym oraz przetwórczym. W porównaniu z innymi technikami regulacji, mają one kilka istotnych zalet. Po pierwsze, umożliwiają w naturalny sposób uwzględnienie ograniczeń sygnałów wejściowych i wyjściowych procesu, których spełnienie jest kluczowe z punktu widzenia jakości, efektywności ekonomicznej i bezpieczeństwa produkcji. Po drugie, algorytmy te można z powodzeniem zastosować do regulacji procesów wielowymiarowych, o wielu wejściach i wielu wyjściach. Co więcej, algorytmy regulacji predykcyjnej można wykorzystać do regulacji procesów o trudnej dynamice, np. procesów z opóźnieniem lub z odwrotną odpowiedzią skokową.

Cechą szczególną regulacji predykcyjnej jest wykorzystanie do prognozowania stanu procesu i obliczania (optymalizacji) wartości sygnałów sterujących dynamicznego modelu procesu. Ponieważ właściwości wielu procesów są nieliniowe, z coraz większym zainteresowanie spotykają się nieliniowe algorytmy regulacji predykcyjnej [4, 12, 17, 19]. Ogólna zasada regulacji predykcyjnej nie nakłada żadnych ograniczeń dotyczących rodzaju modelu.

W szczególności, w algorytmie regulacji predykcyjnej może zastosować modele fizykochemiczne. Rozwiązanie takie ma kila bardzo istotnych wad. Opracowanie modeli jest zwykle trudne, długotrwałe i kosztowne. Co więcej, modele fizykochemiczne są zazwyczaj złożone (nieliniowe układy równań różniczkowych i algebraicznych). Bezpośrednie zastosowanie takich modeli w algorytmach regulacji predykcyjnej nie jest zwykle możliwe, ponieważ w trakcie cyklicznego rozwiązywania równań nieliniowych nieuniknione są problemy numeryczne.

W algorytmach regulacji predykcyjnej stosuje się różne rodzaje modeli nieliniowych, np. modele wielomianowe, szeregi Volterry, modele rozmyte, sieci neuronowe. Autor niniejszej pracy prowadzi badania algorytmów wykorzystujących modele neuronowe [8, 9, 20]. W przeciwieństwie do modeli fizykochemicznych, mają one kilka zalet, a mianowicie:

a) Sieci neuronowe są uniwersalnymi aproksymatorami. Potrafią aproksymować dowolną funkcję nieliniową z bardzo dużą dokładnością [5], mogą one również dokładnie modelować zachowanie procesów technologicznych [6, 13].
b) Opracowano wiele algorytmów uczenia sieci i doboru ich architektury [3, 14].
c) Sieci neuronowe mają prostą, regularną strukturę oraz stosunkowo mało parametrów.
d) Sieci neuronowe mogą być stosunkowo łatwo wykorzystane w różnych odmianach algorytmów regulacji predykcyjnej [1, 6, 7, 8, 9, 13, 15, 16, 20, 21].
e) W algorytmach regulacji predykcyjnej z modelami neuronowymi nie występują problemy numeryczne typowe dla algorytmów z modelami fizykochemicznymi.

W pracy omówiono analityczny nieliniowy algorytm regulacji predykcyjnej wykorzystujący modele neuronowe. Aktualna wartość sygnału sterującego wyznaczana jest w sposób analityczny, bez potrzeby optymalizacji. Przedstawiony algorytm jest efektywny obliczeniowo, wymaga jedynie cyklicznej dekompozycji LU macierzy i rozwiązania dwóch prostych równań liniowych. Aby zapewnić spełnienie istniejących ograniczeń, obliczone rozwiązanie jest rzutowane na zbiór istniejących ograniczeń. Algorytm charakteryzuje się dużą dokładnością regulacji, porównywalną z algorytmami wymagającymi bieżącej nieliniowej optymalizacji.

2. REGULACJA PREDYKCYJNA

2.1. Zasada regulacji predykcyjnej

W algorytmie regulacji predykcyjnej, w każdej dyskretnej chwili k (iteracji algorytmu) wyznacza się wektor przyszłych przyrostów sygnału sterującego

$$\Delta \boldsymbol{u}(k) = \left[\Delta u(k \mid k) \quad \ldots \quad \Delta u(k + N_u - 1 \mid k) \right]^T \tag{1}$$

Zakłada się, że $\Delta u(k + p \mid k) = 0$ dla $p \geq N_u$, przy czym N_u jest horyzontem sterowania. Celem algorytmu jest minimalizacja różnic między trajektorią zadaną $y^{zad}(k + p \mid k)$ a prognozowanymi wartościami sygnału wyjściowego $\hat{y}(k + p \mid k)$ na całym horyzoncie predykcji, dla $p = 1, \ldots, N$, przy czym zwykle $N_u < N$. Minimalizowana funkcja kosztu ma postać

$$J(k) = \sum_{p=1}^{N} \mu_p (y^{zad}(k + p \mid k) - \hat{y}(k + p \mid k))^2 + \sum_{p=0}^{N_u - 1} \lambda_p (\Delta u(k + p \mid k))^2 \tag{2}$$

gdzie $\mu_p \geq 0$ i $\lambda_p > 0$ są współczynnikami wagowymi. Do sterowania wykorzystuje się jedynie pierwszy element wyznaczonej sekwencji (1), prawo regulacji ma postać

$$u(k) = \Delta u(k \mid k) + u(k-1) \tag{3}$$

W kolejnej dyskretnej chwili k+1 następuje aktualizacja pomiaru zmiennej wyjściowej, horyzont predykcji zostaje przesunięty o jeden krok do przodu i cała procedura jest powtórzona.

W regulacji predykcyjnej wykorzystuje się dynamiczny model procesu. Jest on stosowany do obliczania przewidywanych wartości zmiennej wyjściowej. Oznacza to, że jeżeli tylko dostępny model procesu jest dokładny, można zaprojektować bardzo dobry algorytm regulacji predykcyjnej, uwzględniający naturę procesu, jego właściwości statyczne i dynamiczne.

2.2. Optymalizacja polityki sterowania

Jak podkreślono we wstępie, możliwość uwzględnienia ograniczeń sygnałów procesowych jest jedną z zalet algorytmów regulacji predykcyjnej. W dalszej części pracy uwzględnia się ograniczenia minimalnej ($u_{\min}$) i maksymalnej wartości sygnału sterującego ($u_{\max}$) oraz ograniczenie maksymalnej szybkości zmian sygnału sterującego ($\Delta u_{\max}$). W każdej iteracji algorytmu regulacji minimalizowany jest wskaźnik jakości regulacji (2) przy uwzględnieniu ograniczeń. Odbywa się to w wyniku rozwiązania następującego zadania optymalizacji

$$\min_{\Delta \boldsymbol{u}(k)} \{J(k) = \sum_{p=1}^{N} \mu_p (y^{zad}(k+p \mid k) - \hat{y}(k+p \mid k))^2 + \sum_{p=0}^{N_u-1} \lambda_p (\Delta u(k+p \mid k))^2\}$$

przy ograniczeniach : (4)

$$u_{\min} \leq u(k+p \mid k) \leq u_{\max} \qquad p = 0, \ldots, N_u - 1$$
$$-\Delta u_{\max} \leq \Delta u(k+p \mid k) \leq \Delta u_{\max} \qquad p = 0, \ldots, N_u - 1$$

Definiując wektory o długości N

$$\begin{aligned} \boldsymbol{y}^{zad}(k) &= \left[y^{zad}(k+1 \mid k) \quad \ldots \quad y^{zad}(k+N \mid k) \right]^T \\ \hat{\boldsymbol{y}}(k) &= \left[\hat{y}(k+1 \mid k) \quad \ldots \quad \hat{y}(k+N \mid k) \right]^T \end{aligned} \tag{5}$$

oraz wektory o długości N_u

$$\begin{aligned} \boldsymbol{u}_{\min} &= \left[u_{\min} \quad \ldots \quad u_{\min} \right]^T, \quad \boldsymbol{u}^{k-1} = \left[u(k-1) \quad \ldots \quad u(k-1) \right]^T \\ \boldsymbol{u}_{\max} &= \left[u_{\max} \quad \ldots \quad u_{\max} \right]^T, \quad \Delta \boldsymbol{u}_{\max} = \left[\Delta u_{\max} \quad \ldots \quad \Delta u_{\max} \right]^T \end{aligned} \tag{6}$$

problem optymalizacji (4) można przedstawić jako

$$\min_{\Delta \boldsymbol{u}(k)} \{J(k) = \left\| \boldsymbol{y}^{zad}(k) - \hat{\boldsymbol{y}}(k) \right\|_{\boldsymbol{M}}^2 + \left\| \Delta \boldsymbol{u}(k) \right\|_{\boldsymbol{\Lambda}}^2\}$$

przy ograniczeniach : (7)

$$\boldsymbol{u}_{\min} \leq \boldsymbol{J} \Delta \boldsymbol{u}(k) + \boldsymbol{u}^{k-1} \leq \boldsymbol{u}_{\max}$$
$$-\Delta \boldsymbol{u}_{\max} \leq \Delta \boldsymbol{u}(k) \leq \Delta \boldsymbol{u}_{\max}$$

przy czym $\boldsymbol{J}$ jest trójkątną dolną macierzą o wymiarowości $N_u \times N_u$, natomiast $\boldsymbol{M}$ oraz $\boldsymbol{\Lambda}$ są diagonalnymi macierzami o wymiarowości, odpowiednio, $N \times N$ oraz $N_u \times N_u$ zawierającymi współczynniki wagowe μ_p, λ_p.

2.3. Klasyfikacja nieliniowych algorytmów regulacji predykcyjnej

W ogólności, można wyróżnić dwie klasy nieliniowych algorytmów regulacji predykcyjnej:

a) algorytmy z nieliniową optymalizacją

b) algorytmy suboptymalne z cykliczną linearyzacją nieliniowego modelu procesu.

W pierwszym przypadku do wyznaczenia aktualnej polityki sterowania (wektora przyszłych przyrostów sygnału sterującego) wykorzystuje się nieliniowy model procesu beż żadnych uproszczeń. Prognozowane wartości sygnału wyjściowego zależą wówczas w sposób nieliniowy od obliczanych przyszłych sygnałów sterujących. Oznacza to, że w każdej iteracji algorytmu w czasie rzeczywistym należy rozwiązać nieliniowe zadanie optymalizacji (7). Jest to nie tylko trudne, złożone obliczeniowe i czasochłonne. Przede wszystkim jednak, istnieje niebezpieczeństwo utknięcia procedury optymalizacji w płytkim minimum lokalnym.

Cechą szczególną algorytmów suboptymalnych jest cykliczna linearyzacja nieliniowego modelu procesu. Uzyskane przybliżenie liniowe jest następnie wykorzystywane w optymalizacji polityki sterowania. Dzięki linearyzacji prognozowane wartości sygnału wyjściowego zależą w sposób liniowy od obliczanych przyszłych sygnałów sterujących. W każdej iteracji algorytmu rozwiązuje się zadanie optymalizacji kwadratowej, co może być wykonane w skończonym czasie, możliwym do przewidzenia. Algorytmy suboptymalne zapewniają w praktyce dobrą jakość regulacji, niewiele gorszą od algorytmów z nieliniową optymalizacją [4, 19].

3. ANALITYCZNY NIELINIOWY ALGORYT REGULACJI PREDYKCYJNEJ

3.1. Struktura modelu neuronowego

Zakłada się, że model dynamiczny procesu nieliniowego o jednym wejściu i jednym wyjściu (ang. Single-Input Single-Output, w skrócie SISO) dany jest w postaci równania

$$y(k) = f(\boldsymbol{x}(k)) = f(u(k-\tau),\ldots,u(k-n_B),y(k-1),\ldots,y(k-n_A)) \tag{8}$$

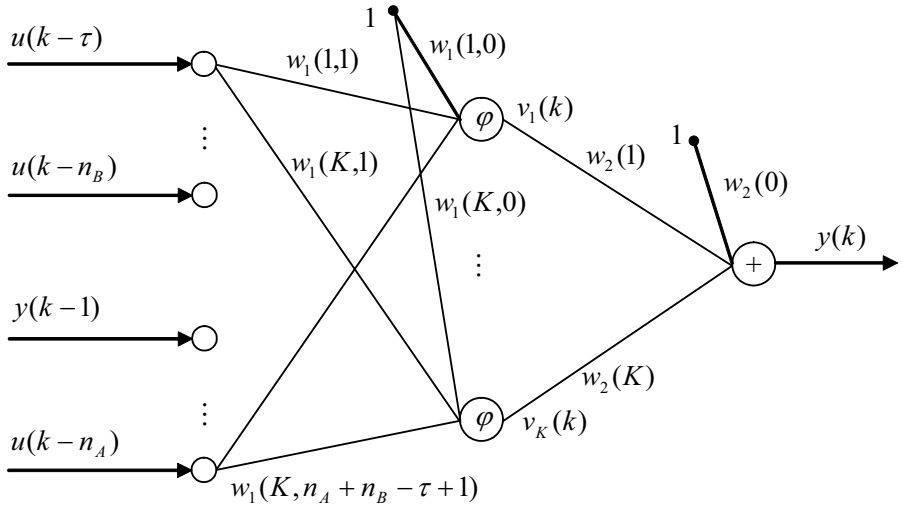
Aktualna wartość sygnału wyjściowego procesu jest nieliniową funkcją sygnału wejściowego i wyjściowego w poprzednich iteracjach algorytmu (chwilach próbkowania). Nieliniowa funkcja $f:\Re^{n_A+n_B-\tau+1}\to\Re$, gdzie $\tau \le n_B$, jest realizowana przez sieć neuronową typu perceptronowego (ang. Multilayer Perceptron, w skrócie MLP) z jedną warstwą ukrytą i liniowym węzłem wyjściowym [3, 14]. Wyjście sieci (modelu) obliczane jest ze wzoru

$$y(k) = w_0^2 + \sum_{i=1}^{K} w_i^2 \varphi(z_i(k)) \tag{9}$$

gdzie wielkości $z_i(k)$ jest sumą sygnałów wejściowych i-tego neuronu ukrytego, $\varphi:\Re\to\Re$ jest nieliniową funkcją aktywacji neuronów warstwy ukrytej (np. $\varphi=\tanh$), K jest ilością neuronów ukrytych. Na podstawie (8), dla sieci neuronowej otrzymuje się

$$z_i(k) = w_1(1,0) + \sum_{j=1}^{I_u} w_1(i,j)u(k-\tau+1-j) + \sum_{j=1}^{n_A} w_1(i,I_u+j)y(k-j) \tag{10}$$

Wagi pierwszej warstwy oznaczone są jako $w_1(i,j)$, przy czym $i=1,\ldots,K$, $j=0,\ldots,n_A+n_B-\tau+1$, natomiast wagi drugiej warstwy przez $w_2(i)$, gdzie $i=0,\ldots,K$. Struktura sieci neuronowej pokazana jest na rys. 1.



Rys. 1. Struktura modelu neuronowego

3.2. Optymalizacja polityki sterowania

Aby wyeliminować konieczność rozwiązywania w każdej iteracji algorytmu nieliniowego problemu optymalizacji (7), w opisywanym algorytmie zostanie wykorzystane podejście suboptymalne, bazujące na algorytmie z Nieliniową Predykcją i Linearyzacją (NPL) [8, 9, 19, 20]. W każdej iteracji k algorytmu NPL nieliniowy neuronowy model procesu jest wykorzystany dwa razy, a mianowicie do wyznaczenia lokalnego przybliżenia liniowego oraz do obliczenia nieliniowej trajektorii swobodnej. Zakłada się, że wektor predykcji sygnału wyjściowego $\hat{\boldsymbol{y}}(k)$ na horyzoncie predykcji można przedstawić jako sumę trajektorii wymuszonej, która zależy wyłącznie od przyszłości (czyli od przyszłych przyrostów sygnału sterującego $\Delta\boldsymbol{u}(k)$ na horyzoncie sterowania) oraz trajektorii swobodnej $\boldsymbol{y}^0(k)$, która zależy wyłącznie od przeszłości

$$\hat{\boldsymbol{y}}(k) = \boldsymbol{G}(k)\Delta\boldsymbol{u}(k) + \boldsymbol{y}^0(k) \tag{11}$$

Macierz dynamiczna o wymiarowości $N \times N_u$ zawiera współczynniki odpowiedzi skokowej aktualnie wyznaczonej liniowej aproksymacji modelu nieliniowego. Ma ona postać

$$\boldsymbol{G}(k) = \begin{bmatrix} s_1(k) & 0 & \dots & 0 \\ s_2(k) & s_1(k) & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ s_N(k) & s_{N-1}(k) & \dots & s_{N-N_u+1}(k) \end{bmatrix} \tag{12}$$

natomiast wektor trajektorii swobodnej ma długość N

$$\boldsymbol{y}^0(k) = \begin{bmatrix} y^0(k+1 \mid k) & \dots & y^0(k+N \mid k) \end{bmatrix}^T \tag{13}$$

Dzięki zastosowaniu suboptymalnej predykcji (11), nieliniowy problem optymalizacji rozwiązywany w każdej iteracji algorytmu regulacji predykcyjnej (7) jest w istocie następującym zadaniem programowania kwadratowego

$$\min_{\Delta \boldsymbol{u}(k)}\{J(k) = \left\| \boldsymbol{y}^{zad}(k) - \boldsymbol{G}(k)\Delta \boldsymbol{u}(k) - \boldsymbol{y}^{0}(k) \right\|_{\boldsymbol{M}}^{2} + \left\| \Delta \boldsymbol{u}(k) \right\|_{\boldsymbol{\Lambda}}^{2}\}$$

przy ograniczeniach : (14)

$$\boldsymbol{u}_{\min} \le \boldsymbol{J}\Delta \boldsymbol{u}(k) + \boldsymbol{u}^{k-1} \le \boldsymbol{u}_{\max}$$

$$-\Delta \boldsymbol{u}_{\max} \le \Delta \boldsymbol{u}(k) \le \Delta \boldsymbol{u}_{\max}$$

Klasyczny algorytm NPL uwzględnia ograniczenia w sposób ścisły, są one integralną częścią zadania optymalizacji (14). Algorytm taki jest nazywany algorytmem numerycznym. W będącym tematem pracy algorytmie analitycznym wektor zmiennych decyzyjnych $\Delta \boldsymbol{u}(k)$ oblicza się w sposób analityczny, unikając jakiejkolwiek optymalizacji. Jest to możliwe wówczas, gdy rozwiąże się zadanie optymalizacji funkcji kryterialnej bez jakichkolwiek ograniczeń, a następnie rzutuje uzyskane rozwiązanie na zbiór ograniczeń.

Zadanie optymalizacji analitycznego algorytmu NPL ma postać

$$\min_{\Delta \boldsymbol{u}(k)}\{J(k) = \left\| \boldsymbol{y}^{zad}(k) - \boldsymbol{G}(k)\Delta \boldsymbol{u}(k) - \boldsymbol{y}^{0}(k) \right\|_{\boldsymbol{M}}^{2} + \left\| \Delta \boldsymbol{u}(k) \right\|_{\boldsymbol{\Lambda}}^{2}\} \tag{15}$$

Ponieważ minimalizowana funkcja celu jest kwadratowa, aby uzyskać rozwiązanie wystarczy gradient tej funkcji

$$\frac{\partial J(k)}{\partial \Delta \boldsymbol{u}(k)} = -2\boldsymbol{G}^{T}(k)\boldsymbol{M}(\boldsymbol{y}^{zad}(k) - \boldsymbol{G}(k)\Delta \boldsymbol{u}(k) - \boldsymbol{y}^{0}(k)) + 2\boldsymbol{\Lambda}\Delta \boldsymbol{u}(k) \tag{16}$$

przyrównać do zera (wektorowo). Wektor optymalnych przyrostów sygnałów sterującego obliczany jest ze wzoru

$$\Delta \boldsymbol{u}(k) = \boldsymbol{K}(k)(\boldsymbol{y}^{zad}(k) - \boldsymbol{y}^{0}(k)) \tag{17}$$

gdzie

$$\boldsymbol{K}(k) = \left(\boldsymbol{G}^{T}(k)\boldsymbol{M}\boldsymbol{G}(k) + \boldsymbol{\Lambda}\right)^{-1}\boldsymbol{G}^{T}(k)\boldsymbol{M} \tag{18}$$

jest macierzą o wymiarowości $N_u \times N$. Jest ona zależna od macierzy dynamicznej $\boldsymbol{G}(k)$, obliczanej w każdej iteracji algorytmu NPL na podstawie linearyzacji modelu neuronowego. Oznacza to, że również macierz $\boldsymbol{K}(k)$ musi być obliczana w każdej iteracji algorytmu. Łatwo pokazać, że uzyskane rozwiązanie jest minimum globalnym zadania (15), ponieważ macierz drugich pochodnych (hesjan) funkcji kryterialnej $\frac{\partial^2 J(k)}{\partial \Delta \boldsymbol{u}^2(k)} = 2(\boldsymbol{G}^{T}(k)\boldsymbol{M}\boldsymbol{G}(k) + \boldsymbol{\Lambda})$ jest dodatnio określona dla współczynników wagowych $\mu_p \ge 0$ i $\lambda_p > 0$.

Do obliczania odwrotności macierzy $\boldsymbol{G}^{T}(k)\boldsymbol{M}\boldsymbol{G}(k) + \boldsymbol{\Lambda}$ stosuje się rozkład (dekompozycję) LU [2] tej macierzy, a następnie rozwiązuje się równania liniowe. Wyznacza się macierze $\boldsymbol{P}(k)$, $\boldsymbol{L}(k)$ oraz $\boldsymbol{U}(k)$, dla których

$$\boldsymbol{L}(k)\boldsymbol{U}(k) = \boldsymbol{P}(k)\left(\boldsymbol{G}^{T}(k)\boldsymbol{M}\boldsymbol{G}(k) + \boldsymbol{\Lambda}\right) \tag{19}$$

przy czym $\boldsymbol{L}(k)$ jest macierzą trójkątną dolną (z jedynkami na przekątnej), $\boldsymbol{U}(k)$ jest macierzą trójkątną górną

(20)

$$\boldsymbol{L}(k)=\begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ l_{2,1}(k) & 1 & 0 & \dots & 0 \\ l_{3,1}(k) & l_{3,2}(k) & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ l_{N_u,1}(k) & l_{N_u,2}(k) & l_{N_u,3}(k) & \dots & 1 \end{bmatrix},\quad \boldsymbol{U}(k)=\begin{bmatrix} u_{1,1}(k) & u_{1,2}(k) & u_{1,3}(k) & \dots & u_{1,N_u}(k) \\ 0 & u_{2,2}(k) & u_{2,3}(k) & \dots & u_{2,N_u}(k) \\ 0 & 0 & u_{3,3}(k) & \dots & u_{3,N_u}(k) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & u_{N_u,N_u}(k) \end{bmatrix}$$

Natomiast $\boldsymbol{P}(k)$ jest macierzą przestawień w eliminacji Gaussa, w każdym wierszu i kolumnie zawiera ona tylko jeden element równy 1, pozostałe elementy są zerowe.

Prawdziwa jest zależność

$$\boldsymbol{P}(k)\left(\boldsymbol{G}^T(k)\boldsymbol{M}\boldsymbol{G}(k)+\boldsymbol{\Lambda}\right)\left(\boldsymbol{G}^T(k)\boldsymbol{M}\boldsymbol{G}(k)+\boldsymbol{\Lambda}\right)^{-1}=\boldsymbol{P}(k)\boldsymbol{I} \tag{21}$$

Czyli, korzystając z (19), otrzymuje się

$$\boldsymbol{L}(k)\boldsymbol{U}(k)\left(\boldsymbol{G}^T(k)\boldsymbol{M}\boldsymbol{G}(k)+\boldsymbol{\Lambda}\right)^{-1}=\boldsymbol{P}(k)\boldsymbol{I} \tag{22}$$

Niech szukana macierz odwrotna ma postać

$$\boldsymbol{H}(k)=\left(\boldsymbol{G}^T(k)\boldsymbol{M}\boldsymbol{G}(k)+\boldsymbol{\Lambda}\right)^{-1}=\begin{bmatrix} h_{1,1}(k) & h_{1,2}(k) & \dots & h_{1,N_u}(k) \\ h_{2,1}(k) & h_{2,2}(k) & \dots & h_{2,N_u}(k) \\ \vdots & \vdots & \ddots & \vdots \\ h_{N_u,1}(k) & h_{N_u,1}(k) & \dots & h_{N_u,N_u}(k) \end{bmatrix} \tag{23}$$

Otrzymuje się zatem

$$\boldsymbol{L}(k)\boldsymbol{U}(k)\begin{bmatrix} h_{1,1}(k) & h_{1,2}(k) & \dots & h_{1,N_u}(k) \\ h_{2,1}(k) & h_{2,2}(k) & \dots & h_{2,N_u}(k) \\ \vdots & \vdots & \ddots & \vdots \\ h_{N_u,1}(k) & h_{N_u,1}(k) & \dots & h_{N_u,N_u}(k) \end{bmatrix}=\boldsymbol{P}(k)\begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{bmatrix} \tag{24}$$

Niech wektory $\tilde{h}_1(k),\dots,\tilde{h}_{N_u}(k)$ będą kolejnymi kolumnami macierzy odwrotnej, natomiast wektory $v_1(k),\dots,v_{N_u}(k)$ będą kolejnymi kolumnami iloczynu macierzy $\boldsymbol{P}(k)\boldsymbol{I}$. Powyższe równanie macierzowe odpowiada N_u układom równań liniowych

$$\boldsymbol{L}(k)\boldsymbol{U}(k)\tilde{h}_1(k)=v_1(k),\quad\cdots\quad\boldsymbol{L}(k)\boldsymbol{U}(k)\tilde{h}_{N_u}(k)=v_{N_u}(k) \tag{25}$$

Rozwiązanie powyższych równań odbywa się dwuetapowo. Dzięki zastosowaniu rozkładu LU najpierw należy rozwiązać układy równań $\boldsymbol{L}(k)q_p(k)=v_p(k)$, a następnie $\boldsymbol{U}(k)\tilde{h}_p(k)=q_p(k)$ dla p=1,…,N_u, przy czym $q_p(k)=\left[q_{p,1}(k)\quad\dots\quad q_{p,N_u}(k)\right]^T$ jest wektorem rozwiązań pierwszych równań podstawianym do drugich. Dzięki rozkładowi LU macierze $\boldsymbol{L}(k)$ oraz $\boldsymbol{U}(k)$ są trójkątne, rozwiązania obu układów równań są bardzo proste

$$\begin{aligned} q_{p,i}(k)&=v_{p,i}(k)-\sum_{j=1}^{i-1}l_{i,j}(k)q_{p,j}(k)\quad i=1,\dots,N_u \\ h_{p,i}(k)&=\frac{1}{u_{i,i}(k)}\left(q_{p,i}(k)-\sum_{j=i+1}^{N_u}u_{i,j}(k)h_{p,j}(k)\right)\quad i=N_u,\dots,1 \end{aligned} \tag{26}$$

Obliczone rozwiązanie może nie spełniać istniejących ograniczeń minimalnej ($u_{\min}$) lub maksymalnej wartości ($u_{\max}$) oraz maksymalnej szybkości zmian sygnału sterującego ($\Delta u_{\max}$). Dlatego też obliczony przyrost $\Delta u(k \mid k)$ zostaje rzutowany na zbiór ograniczeń. Innymi słowy, uzyskane rozwiązanie zostaje przycięte w taki sposób, aby spełniało istniejące ograniczenia. Procedura rzutowania ograniczeń jest następująca

$$\begin{aligned}
&\text{jeżeli} \quad \Delta u(k \mid k) < -\Delta u_{\max} \quad \text{przyjmij} \quad \Delta u(k \mid k) = -\Delta u_{\max} \\
&\text{jeżeli} \quad \Delta u(k \mid k) > \Delta u_{\max} \quad \text{przyjmij} \quad \Delta u(k \mid k) = \Delta u_{\max} \\
&u(k \mid k) = \Delta u(k \mid k) + u(k-1) \\
&\text{jeżeli} \quad u(k \mid k) < u_{\min} \quad \text{przyjmij} \quad u(k \mid k) = u_{\min} \\
&\text{jeżeli} \quad u(k \mid k) > u_{\max} \quad \text{przyjmij} \quad u(k \mid k) = u_{\max} \\
&u(k) = u(k \mid k)
\end{aligned} \tag{27}$$

Warto zauważyć, że w analitycznym algorytmie NPL nie ma potrzeby obliczania całego wektora przyszłych przyrostów sygnału sterującego $\Delta \boldsymbol{u}(k)$, a tylko pierwszy jego element $\Delta u(k \mid k)$. Wzory (17) i (18) pozostają w mocy, oblicza się tylko pierwszy wiersz macierzy $\boldsymbol{K}(k)$. Aby było to możliwe dekompozycji LU poddaje się macierz

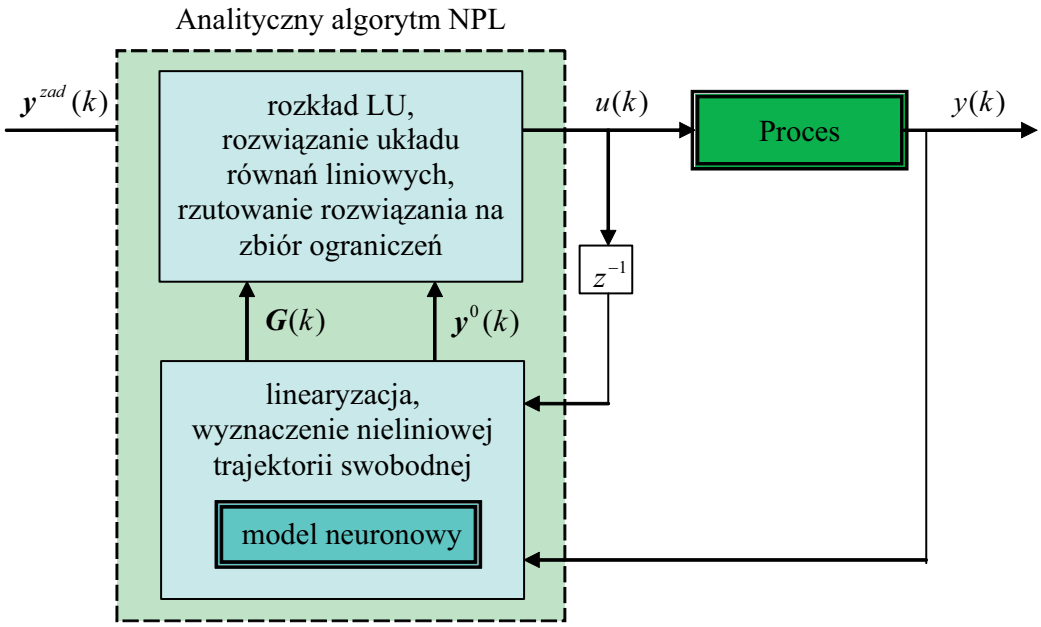
$$\boldsymbol{L}(k)\boldsymbol{U}(k) = \boldsymbol{P}(k)\left(\boldsymbol{G}^T(k)\boldsymbol{M}\boldsymbol{G}(k) + \boldsymbol{\Lambda}\right)^T \tag{28}$$

W rezultacie, oblicza się tylko pierwszą kolumnę macierzy odwrotnej $\boldsymbol{H}(k)$. Rozwiązuje się tylko pierwsze z równań (25). Równania (26) pozostają w mocy, ale tylko dla p=1.

Ogólna struktura algorytmu analitycznego z Nieliniową Predykcją i Linearyzacją (NPL) została przedstawiona na rys. 2. W każdej iteracji wykonane są następujące kroki:

1. Linearyzacja modelu neuronowego: wyznaczenie macierzy dynamicznej $\boldsymbol{G}(k)$.
2. Obliczenie nieliniowej trajektorii swobodnej $\boldsymbol{y}^0(k)$ na podstawie modelu neuronowego.
3. Wyznaczenie rozkładu LU macierzy $\left(\boldsymbol{G}^T(k)\boldsymbol{M}\boldsymbol{G}(k) + \boldsymbol{\Lambda}\right)^T$ – wzór (28).
4. Wyznaczenie pierwszej kolumny odwrotności macierzy $\left(\boldsymbol{G}^T(k)\boldsymbol{M}\boldsymbol{G}(k) + \boldsymbol{\Lambda}\right)^T$ przez rozwiązanie równania liniowego $\boldsymbol{L}(k)\boldsymbol{U}(k)\widetilde{h}_1(k) = v_1(k)$, pierwszego z układu (25).
5. Obliczenie pierwszego elementu $\Delta u(k \mid k)$ optymalnego wektora zmiennych decyzyjnych $\Delta \boldsymbol{u}(k)$ – wzór (17).
6. Rzutowanie wielkości $\Delta u(k \mid k)$ na zbiór ograniczeń – wzory (27).
7. Zastosowanie do sterowania obliczonego sterowania.
8. Podstawienie $k := k+1$, przejście do kroku 1.

Szczegóły dotyczące linearyzacji modelu neuronowego, obliczenia macierzy dynamicznej i wyznaczania trajektorii swobodnej podano w pracach [8, 19, 20].



Rys. 2. Struktura analitycznego algorytmu z Nieliniową Predykcją i Linearyzacją (NPL)

4. EKSPERYMENTY

4.1. Reaktor polimeryzacji

Rozważanym procesem jest reaktor polimeryzacji [11]. Jego wejściem (zmienną manipulowaną) jest natężenie dopływu inicjatora F_I [m^3/h], wyjściem (zmienną regulowaną) jest średnia masa cząsteczkowa *NAMW* (ang. Number Average Molecular Weight). Reaktor ten jest często stosowany w celu porównania nieliniowych algorytmów regulacji [8, 9, 19, 20].

4.2. Modelowanie procesu polimeryzacji

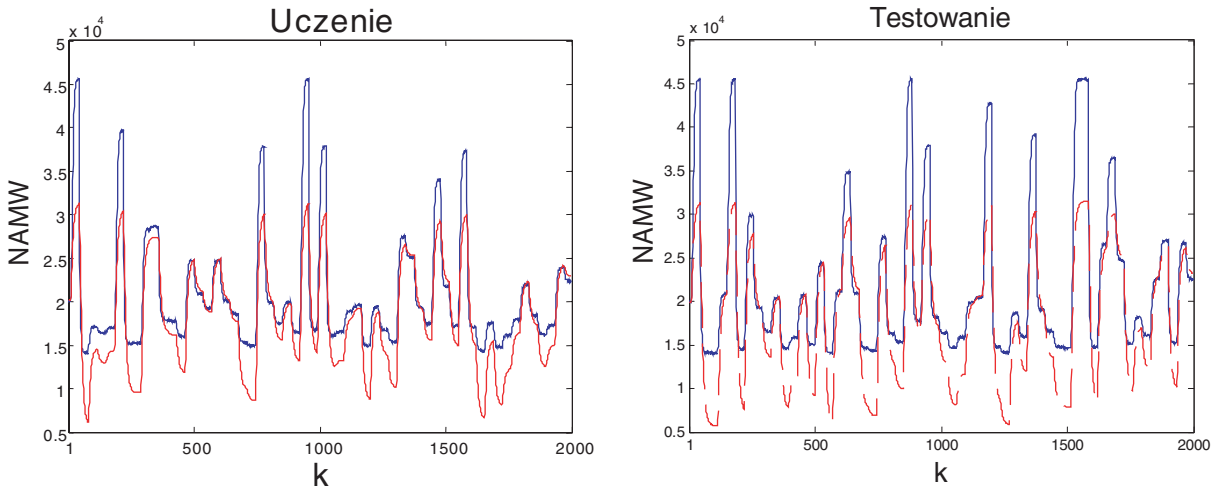
Model fizykochemiczny [11] jest traktowany podczas symulacji jako rzeczywisty proces. Zostały wygenerowane dwa zbiory danych liczące 2000 próbek, a mianowicie zbiór danych uczących oraz danych testowych. Pierwszy z nich służy do identyfikacji (uczenia) modeli, drugi – wyłącznie do oceny jakości otrzymanych modeli. Aby oddać warunki panujące w przemyśle do wyjścia procesu dodano niewielki szum. Przygotowano dwa modele, a mianowicie model liniowy

$$y(k) = b_2 u(k-2) - a_1 y(k-1) - a_2 y(k-2) \tag{29}$$

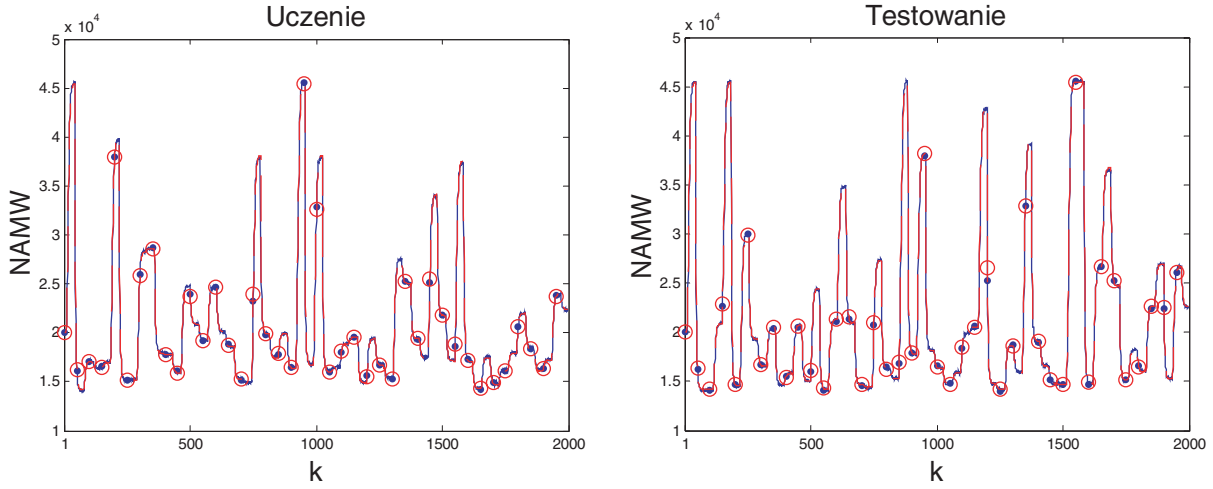
oraz model neuronowy

$$y(k) = f(u(k-2), y(k-1), y(k-2)) \tag{30}$$

Oba modele mają te same argumenty. Oczywiście, wykonano szereg eksperymentów mających na celu określenie rzędu dynamiki modeli, w pracy przedstawiono tylko modele będące efektem tych poszukiwań. Na rys. 3 pokazano wyjście modelu liniowego na tle obu zbiorów danych, natomiast na rys. 4 pokazano wyjście modelu neuronowego na tle danych. Niedokładność modelu liniowego jest bardzo duża, natomiast model neuronowy prawidłowo oddaje właściwości procesu.



Rys. 3. Dane oraz wyjście modelu liniowego dla zbioru uczącego i testowego



Rys. 4. Dane oraz wyjście modelu neuronowego dla zbioru uczącego i testowego

4.3. Regulacja predykcyjna procesu polimeryzacji

Porównano cztery rodzaje algorytmów regulacji predykcyjnej, a mianowicie:

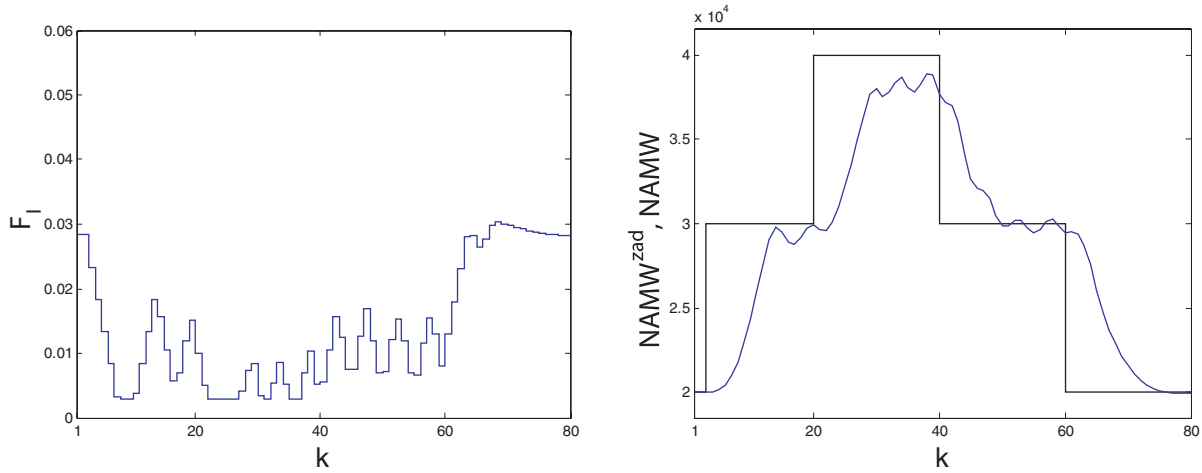
a) Algorytm liniowy z modelem liniowym (29).

b) Analityczny algorytm NPL (bez optymalizacji) z modelem neuronowym (30).

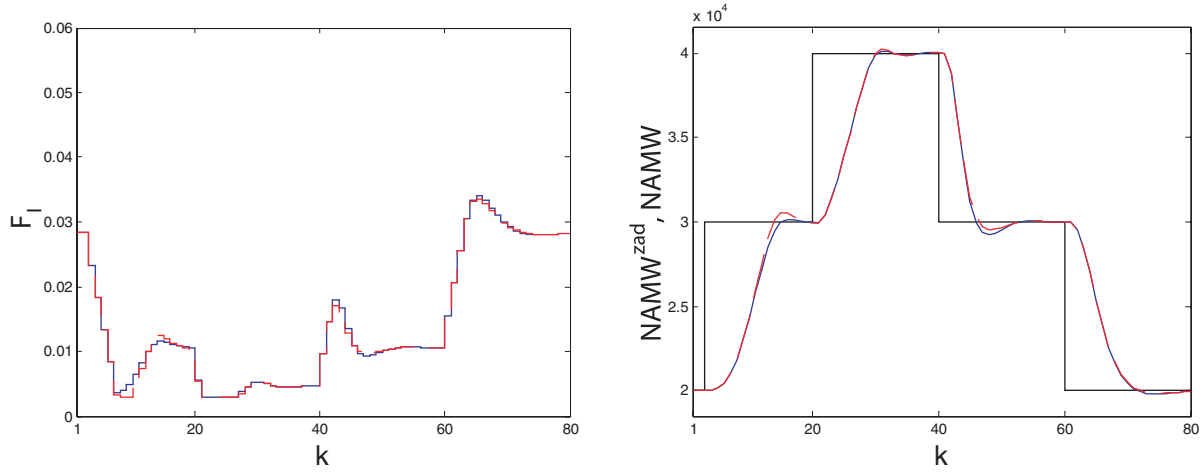
c) Numeryczny algorytm NPL (z optymalizacją kwadratową) z modelem neuronowym (30).

d) Numeryczny algorytm z nieliniową optymalizacją z modelem neuronowym (30).

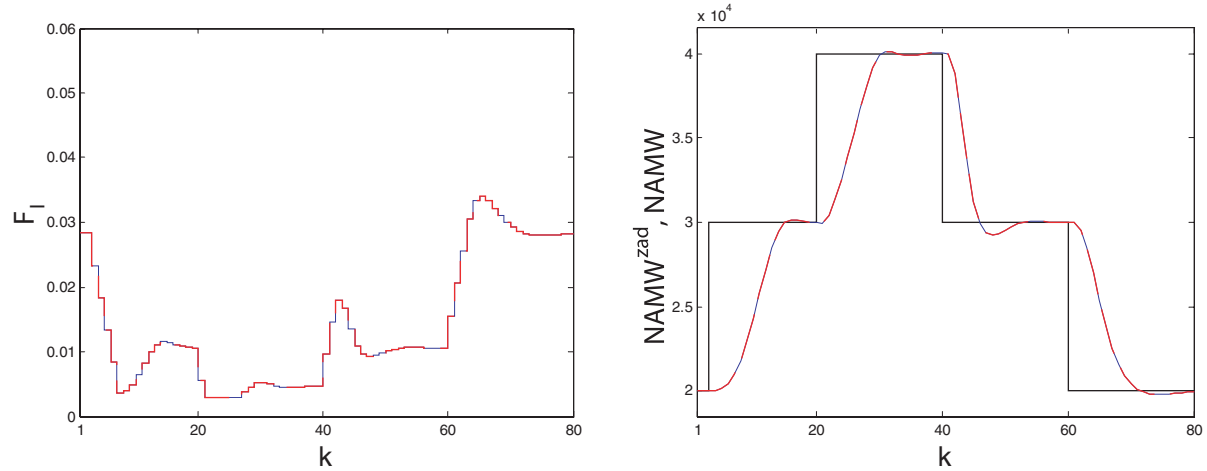
We wszystkich badanych algorytmach przyjęto te same parametry: N=10, N_u=3, μ_p=1, λ_p=0.2, $F_{I\min}$=0,003 m^3/h, $F_{I\max}$=0,06 m^3/h, $\Delta F_{I\max}$=0,005 m^3/h. Czas próbkowania wynosi 1.8 min.



Rys. 5. Wyniki symulacji algorytmu liniowego

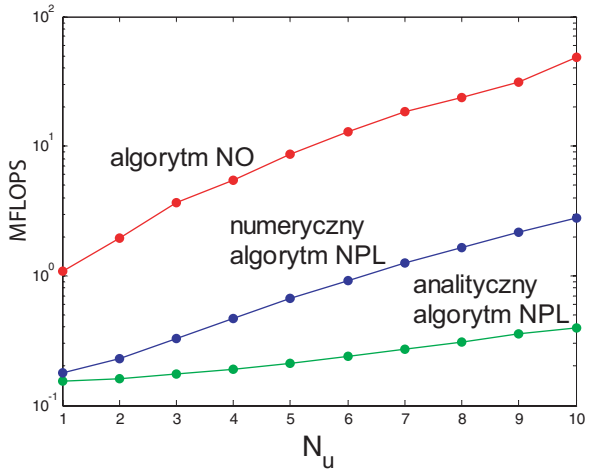


Rys. 6. Wyniki symulacji algorytmu
z nieliniową optymalizacją oraz NPL w wersji analitycznej



Rys. 7. Wyniki symulacji algorytmu
NPL w wersji numerycznej oraz NPL w wersji analitycznej

Ponieważ jakość modelu liniowego jest zła (rys. 3), algorytm wykorzystujący ten model pracuje źle, jest on niestabilny co pokazano na rys. 5. Wyniki symulacji algorytmu z nieliniową optymalizacją oraz NPL w wersji analitycznej przedstawiono na rys. 6. Oba porównywane algorytmy pracują bardzo podobnie. Algorytm suboptymalny z cykliczną linearyzacją, rozkładem LU i rzutowaniem ograniczeń pozwolił w badanym przypadku na osiągnięcie jakości regulacji praktycznie takiej samej jak w algorytmie z nieliniową optymalizacją powtarzaną w każdej iteracji. Rys. 7 przedstawia porównanie algorytmu NPL w wersji analitycznej i numerycznej z optymalizacją kwadratową powtarzaną w każdej iteracji. Różnice między algorytmem analitycznym i numerycznym są praktycznie niedostrzegalne.



Rys. 8. Złożoność obliczeniowa badanych algorytmów w zależności od horyzontu sterowania

Na rys. 8 porównano złożoność obliczeniową badanych nieliniowych algorytmów regulacji predykcyjnej w zależności od horyzontu sterowania, we wszystkich przypadkach horyzont predykcji jest stały (N=10). Można zaobserwować, że przedstawiony w pracy analityczny algorytm NPL jest nie tylko dużo bardziej efektywny obliczeniowo niż algorytm z nieliniową optymalizacją, ale również mniej złożony niż numeryczny algorytm NPL, w którym w każdej iteracji należy rozwiązać zadanie optymalizacji kwadratowej.

5. PODSUMOWANIE

Opisany w pracy analityczny algorytm regulacji predykcyjnej cechuje się dużą jakością regulacji. Jest ona porównywalna nie tylko z dokładnością algorytmu numerycznego z optymalizacją kwadratową, jest ona niewiele gorsza niż w algorytmie z nieliniową optymalizacją. Algorytm jest przy tym bardzo efektywny obliczeniowo, nie ma potrzeby optymalizacji. Algorytm wymaga rozkładu LU oraz rozwiązania dwóch prostych równań liniowych.

LITERATURA

[1] B. M. Åkesson, H. T. Toivonen (2006): A neural network model predictive controller. *Journal of Process Control*, tom 16, nr 3, str. 937-946.

[2] G. H. Golub, C. F. Van Loan (1989): *Matrix computations*. The Johns Hopkins University Press, Baltimore and London.

[3] S. Haykin (1999): *Neural networks – a comprehensive foundation*. Prentice Hall, Upper Saddle River.

[4] M. A. Henson (1998): Nonlinear model predictive control: current status and future directions. *Computers and Chemical Engineering*, tom 23, nr 2, str. 187-202.

[5] K. Hornik, M. Stinchcombe, H. White (1989): Multilayer feedforward networks are universal approximators. *Neural networks*, tom 2, nr 5, str. 359-366.

[6] M. A. Hussain (1999): Review of the applications of neural networks in chemical process control – simulation and online implementation. *Artificial Intelligence in Engineering*, tom 13, nr 1, str. 55-68.

[7] G. P. Liu, V. Kadirkamanathan, S. A. Billings (1998): Predictive control of nonlinear systems using neural networks. *International Journal of Control*, tom 71, nr 6, str. 1119-1132.

[8] M. Ławryńczuk (2007): A family of model predictive control algorithms with artificial neural networks. *International Journal of Applied Mathematics and Computer Science*, tom 17, nr 2, str. 217-232.

[9] M. Ławryńczuk, P. Tatjewski (2007): A computationally efficient nonlinear predictive control algorithm with RBF neural models and its application. *Lecture Notes in Artificial Intelligence, tom 4585, Springer: The International Conference Rough Sets and Emerging Intelligent Systems Paradigms, RSEISP 2007*, Warszawa, str. 603-612.

[10] J. M. Maciejowski (2002): *Predictive control with constraints*. Prentice Hall, Harlow.

[11] B. R. Maner, F. J. Doyle, B. A. Ogunnaike, R. K. Pearson (1996): Nonlinear model predictive control of a simulated multivariable polymerization reactor using second-order Volterra models. *Automatica*, tom 32, nr 9, str. 1285-1301.

[12] M. Morari, J. Lee (1999): Model predictive control: past, present and future. *Computers and Chemical Engineering*, tom 23, nr 4/5, str. 667-682.

[13] M. Nørgaard, O. Ravn, N. K. Poulsen, L. K. Hansen (2000): *Neural networks for modelling and control of dynamic systems*. Springer, London.

[14] S. Osowski (1996): *Sieci neuronowe w ujęciu algorytmicznym*. WNT, Warszawa.

[15] S. Piche, B. Sayyar-Rodsari, D. Johnson, M. Gerules (2000): Nonlinear model predictive control using neural networks. *IEEE Control Systems Magazine*, tom 20, nr 3, str. 56-62.

[16] M. Pottmann, D. E. Seborg (1997): A nonlinear predictive control strategy based on radial basis function networks. *Computers and Chemical Engineering*, tom 21, nr 9, str. 965-980.

[17] S. J. Qin, T. Badgwell (2003): A survey of industrial model predictive control technology. *Control Engineering Practice*, tom 11, nr 7, str. 733-764.

[18] J. A. Rossiter (2003): *Model-based predictive control*. CRC Press, Boca Raton.

[19] P. Tatjewski (2007): *Advanced control of industrial processes, structures and algorithms*. Springer, London.

[20] P. Tatjewski, M. Ławryńczuk (2006): Soft computing in model-based predictive control. *International Journal of Applied Mathematics and Computer Science*, tom 16, nr 1, str. 101-120.

[21] D. L. Yu, J. B. Gomm (2003): Implementation of neural network predictive control to a multivariable chemical reactor. *Control Engineering Practice*, tom 11, nr 11, str. 1315-1323.