Tadeusz BIAŁOŃ

Politechnika Śląska, Katedra Elektrotechniki i Informatyki
ORCID: 0000-0001-5056-0514

doi:10.15199/48.2022.11.57

Optymalizacja wzmocnień obserwatora Luenbergera silnika indukcyjnego metodą roju cząstek

Streszczenie. *W artykule opisano eksperymenty z doborem wzmocnień obserwatora zmiennych stanu silnika indukcyjnego przy wykorzystaniu metody optymalizacyjnej opartej na roju cząstek (PSO). W badaniach skupiono się na porównaniu różnych wersji algorytmu PSO, poddając optymalizacji funkcję celu o zawsze takich samych parametrach. Przeanalizowano trzy różne metody uczenia, GB (Global Best), LB (Local Best) oraz FIPS (Fully Informed Particle Swarm). Dwie ostatnie metody działają w oparciu o zadaną topologię roju, do wyboru spośród kraty pierścieniowej, kraty Von Neumanna oraz FDR (Fitness Distance Ratio). Przeanalizowano zagadnienia zbieżności i stabilności algorytmu, zależne od parametrów takich jak współczynnik uczenia.*

Abstract*. The paper describes experiments with the gain selection of an induction motor state observer, using particle swarm optimization (PSO) method. The research focused on comparing different versions of the PSO algorithm, optimizing the fitness function elaborated during preceding research. Three different learning methods were analyzed, GB (Global Best), LB (Local Best) and FIPS (Fully Informed Particle Swarm). The last two methods operate on the basis of a given swarm topology, to be selected from a ring lattice, Von Neumann lattice and FDR (Fitness Distance Ratio). The problems of convergence and stability of the algorithm, depending on parameters such as a cognition factor, were analyzed. The results for 1000 runs of PSO with 20 different sets of parameters were presented and compared. (**Particle swarm optimization of an inductionmotor Luenberger observer**).*

Słowa kluczowe: Optymalizacja rojem cząstek, obserwator Luenbergera, silnik indukcyjny.
Keywords: Particle swarm optimization, Luenberger observer, induction motor.

Wstęp

W napędach elektrycznych z silnikami indukcyjnymi opartymi na dynamicznych metodach sterowania, takich jak sterowanie polowo zorientowane (FOC) [1,2,3,4], konieczne jest odtwarzanie zmiennych stanu silnika, takich jak składowe strumieni magnetycznych sprzężonych z uzwojeniami stojana i wirnika. W tym celu można wykorzystać obserwator Luenbergera [1,5,6,7]. Dodatkowo można odtwarzać prędkość kątową wirnika silnika, na podstawie zmiennych stanu odtworzonych w obserwatorze [5,7]. Umożliwia to uzyskanie bezczujnikowego układu sterowania, w którym pomiar prędkości kątowej nie jest wymagany [1,8,9]. Inną możliwością jest wykorzystanie odtwarzanej prędkości kątowej jednocześnie z jej wartością zmierzoną, tworząc tzw. napędy bezpieczne (odporne na awarie), które przełączają się w tryb bezczujnikowy w przypadku awarii czujnika prędkości kątowej [10], lub stosując strategie diagnostyczne on-line służące do wykrywania uszkodzeń, oparte na porównaniu zmierzonej prędkości kątowej z odtwarzaną w obserwatorze [3,7].

We wszystkich wyżej wymienionych przypadkach jakość rekonstrukcji zmiennych stanu ma decydujący wpływ na działanie układu sterowania. Obserwatorem Luenbergera najczęściej stosowanym do rekonstrukcji zmiennych stanu silnika indukcyjnego jest najprostszy, proporcjonalny [5,11]. Lepsze rezultaty można jednak uzyskać stosując bardziej zawansowane struktury, takie jak obserwator PI o zredukowanym rzędzie członu całkującego (PIr) [12], którego pierwsze zastosowanie w układzie sterowania silnika indukcyjnego opisano w pracy [13]. Zaproponowany w pracy [13] obserwator, tutaj wykorzystany w wersji PIrR (R od ang. rotor, sygnał korekcyjny członu całkującego obserwatora bezpośrednio oddziałuje na odtwarzane zmienne stanu obwodu uzwojenia wirnika), ma wzmocnienia dobierane optymalizacyjnie [14]. Do tej pory używano w tym celu autorskiej metody opartej na algorytmie genetycznym [14], lecz ze względu na złożoność problemu, wynik optymalizacji nie zawsze był zadowalający. W związku z tym, zamiast algorytmu genetycznego do optymalizacji wzmocnień zastosowano bardziej zaawansowaną metodę – algorytm roju cząstek PSO (ang. Particle Swarm Optimization), a eksperymenty z jego zastosowaniem opisano w tym artykule.

Model matematyczny obserwatora PIrR

Przedmiotem rozważań jest silnik indukcyjny, opisany macierzowym, różniczkowym równaniem stanu oraz macierzowym algebraicznym równaniem wyjścia [5,15,16]:

$$\dot{\mathbf{z}} = \mathbf{A}(\omega)\cdot\mathbf{z} + \mathbf{B}\cdot\mathbf{u}, \quad \mathbf{y} = \mathbf{C}\cdot\mathbf{z}, \tag{1}$$

gdzie wektory stanu $\mathbf{z}$, wymuszeń $\mathbf{u}$ oraz odpowiedzi $\mathbf{y}$ mają postaci:

$$\begin{aligned} \mathbf{z} &= \begin{bmatrix}\psi_{s\alpha} & \psi_{s\beta} & \psi_{r\alpha} & \psi_{r\beta}\end{bmatrix}^{\mathrm{T}}, \\ \mathbf{u} &= \begin{bmatrix}u_{s\alpha} & u_{s\beta}\end{bmatrix}^{\mathrm{T}}, \quad \mathbf{y} = \begin{bmatrix}i_{s\alpha} & i_{s\beta}\end{bmatrix}^{\mathrm{T}}, \end{aligned} \tag{2}$$

Wektor stanu $\mathbf{z}$ zawiera strumienie magnetyczne ψ sprzężone z uzwojeniami stojana (indeks s) i wirnika (r) silnika w prostokątnym układzie współrzędnych α–β. Wektory $\mathbf{u}$ i $\mathbf{y}$ zawierają napięcia u oraz prądy i uzwojenia stojana. Macierze $\mathbf{A}$, $\mathbf{B}$ i $\mathbf{C}$ występujące w równaniach (1) mają odpowiednie wymiary oraz postaci podane w [13,14]. Wartości elementów macierzy są stałe w czasie (za wyjątkiem macierzy $\mathbf{A}$, której elementy są parametrycznie zależne od prędkości kątowej wirnika silnika ω) i zależne od parametrów schematu zastępczego silnika.

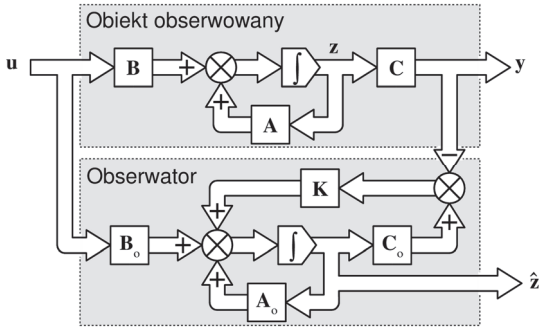
Do odtwarzania zmiennych stanu systemu (1) zastosowano obserwator PIrR [13,14] opisany równaniem w formie sprowadzonej do postaci obserwatora proporcjonalnego:

$$\dot{\mathbf{z}}_{\mathrm{o}} = \mathbf{A}_{\mathrm{o}}(\omega)\cdot\mathbf{z}_{\mathrm{o}} + \mathbf{B}_{\mathrm{o}}\cdot\mathbf{u} + \mathbf{K}(\omega)\cdot\left(\mathbf{C}_{\mathrm{o}}\cdot\mathbf{z}_{\mathrm{o}} - \mathbf{y}\right), \tag{3}$$

gdzie:

$$\begin{aligned} \mathbf{z}_{\mathrm{o}} &= \begin{bmatrix}\hat{\mathbf{z}} \\ \mathbf{h}\end{bmatrix}, \quad \mathbf{A}_{\mathrm{o}}(\omega) = \begin{bmatrix}\mathbf{A}(\omega) & \begin{bmatrix}\mathbf{0}_{2\times 2} \\ \mathbf{1}_2\end{bmatrix} \\ \mathbf{0}_{2\times 4} & \tau^{-1}\mathbf{1}_2\end{bmatrix}, \\ \mathbf{B}_{\mathrm{o}} &= \begin{bmatrix}\mathbf{B} \\ \mathbf{0}_{2\times 2}\end{bmatrix}, \quad \mathbf{C}_{\mathrm{o}} = \begin{bmatrix}\mathbf{C} & \mathbf{0}_{2\times 2}\end{bmatrix} \end{aligned} \tag{4}$$

Wektor zmiennych stanu obserwatora $\mathbf{z}_o$ zawiera wektor odtwarzanych zmiennych stanu systemu (1) $\hat{\mathbf{z}}$ oraz wektor zmiennych stanu członu całkującego obserwatora $\mathbf{h}$. Przez $\mathbf{1}$ i $\mathbf{0}$ oznaczono odpowiednio macierz jednostkową i zerową o podanych wymiarach, τ oznacza stałą czasową zmodyfikowanego integratora członu całkującego [13]. $\mathbf{K}$ to macierz wzmocnień obserwatora o postaci:



Rys.1. Schemat blokowy obserwatora Luenbergera PIrR sprowadzonego do postaci obserwatora proporcjonalnego

$$\mathbf{K}(\omega)=\begin{bmatrix} a\cdot\mathbf{1}_2 + c\cdot\omega\cdot\mathbf{J} \\ \mathbf{0}_{2\times 2} \\ b\cdot\mathbf{1}_2 + d\cdot\omega\cdot\mathbf{J} \end{bmatrix}, \quad \mathbf{J}=\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}. \tag{5}$$

Współczynniki a, b, c i d to wzmocnienia obserwatora podlegające optymalizacji. Zastosowana struktura macierzy wzmocnień ma taką samą symetrię jak macierze modelu matematycznego silnika (1), co jest korzystne z punktu widzenia właściwości użytkowych obserwatora [13]. Schemat blokowy układu obiekt obserwowany – obserwator przedstawiono na rysunku 1. Dodatkowo, obserwator (3) współpracuje z mechanizmem odtwarzania prędkości kątowej ω, opisanym w [5].

Funkcja celu

Zastosowany algorytm optymalizacji poszukuje minimum funkcji celu F, zawierającej składniki opisujące kolejne kryteria doboru obserwatora, takie jak stabilność, stałe czasowe czy odporność na zakłócenia. Pełną postać funkcji celu opisano w pracy [14]. Funkcja celu F zawiera składniki zależne wprost od współczynników a, b, c i d, jak i zależne pośrednio, obliczane na podstawie wyznaczanych numerycznie wartości własnych λ macierzy $\mathbf{A}_o$. Jest to funkcja kary, czyli im gorzej spełnione są kryteria doboru, tym większa jej wartość. Dzięki temu funkcja ma wartości nieujemne, jest ograniczona od dołu, ma więc minimum.

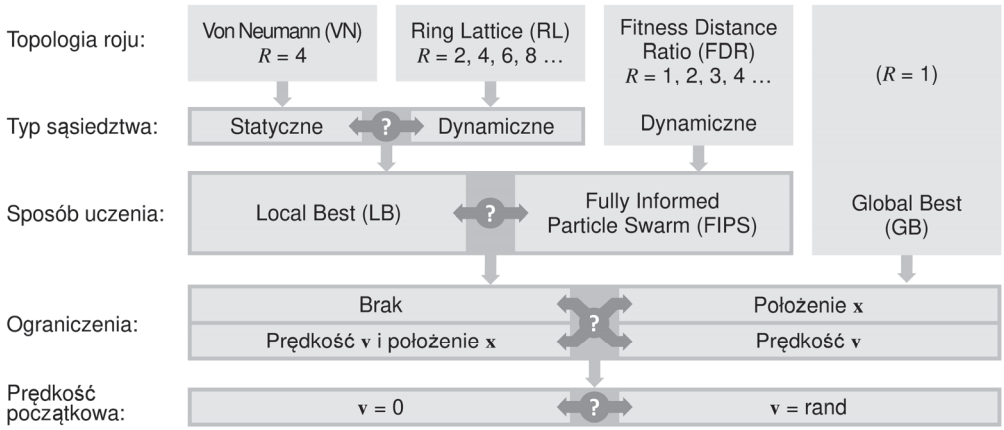
Algorytm optymalizacji rojem cząstek

W przestrzeni poszukiwań znajduje się rój N cząstek. Każda cząstka jest opisana dwoma wektorami, wektorem położenia $\mathbf{x}$ oraz wektorem prędkości $\mathbf{v}$. W rozpatrywanym przypadku przestrzeń poszukiwań ma M = 4 wymiary, odpowiadające optymalizowanym współczynnikom a, b, c i d. W każdym i-tym kroku algorytmu położenie każdej cząstki $\mathbf{x}$ jest wyznaczane na podstawie położenia w kroku poprzednim oraz wektora prędkości $\mathbf{v}$, którego wartość odpowiada przesunięciu cząstki:

$$\mathbf{x} \leftarrow \mathbf{x} + \mathbf{v}\,. \tag{6}$$

Prędkość cząstki $\mathbf{v}$ jest wyznaczana w procesie uczenia. Na proces uczenia mają wpływ aktualne położenie cząstki i odpowiadająca mu wartość funkcji celu $F(\mathbf{x})$, prędkość cząstki w poprzednim kroku algorytmu, ale również zapamiętane najlepsze z poprzednich położeń, jak i wartości funkcji celu odpowiadające położeniom innych cząstek w roju. Cząstki więc aktywnie poszukują minimum funkcji celu, współpracując ze sobą i zapamiętując poprzednio uzyskane dobre wyniki. Sposób współpracy między cząstkami zależy od topologii roju, czyli sieci ich wzajemnych relacji. Dynamikę procesu uczenia można kontrolować zmieniając wartość współczynnika uczenia. Dodatkowo, zarówno na położenie cząstek jaki i ich prędkość można nakładać ograniczenia.

Na potrzeby badań napisano własny program algorytmu PSO, implementujący wiele z najczęściej opisywanych w literaturze rozwiązań. Schemat blokowy opcji konfiguracji wykorzystanego algorytmu przedstawiono na rysunku 2. Celem przeprowadzonych badań było znalezienie konfiguracji najbardziej odpowiedniej dla rozpatrywanego problemu doboru wzmocnień obserwatora.



Rys.2. Opcje konfiguracji algorytmu PSO

Sposób uczenia

Z punktu widzenia właściwości dynamicznych algorytmu najważniejszy jest sposób w jaki w każdej iteracji obliczana jest występująca w równaniu (6) prędkość cząstki. Jest to tzw. sposób uczenia, opisujący jak dana cząstka modyfikuje swoje położenie w oparciu o informację o wartościach funkcji celu innych cząstek oraz o zapamiętanym najlepszym z własnych poprzednich położeń. Wyróżniane są tutaj trzy główne metody [17], GB (ang. Global Best,

również gbest), LB (ang. Local Best, również lbest) oraz FIPS (ang. Fully Informed Particle Swarm).

Metoda GB, historycznie pierwsza i jednocześnie najprostsza [17,18], jest opisana równaniem:

$$\mathbf{v} \leftarrow \chi \cdot \left[\mathbf{v} + \frac{\varphi}{2} \cdot \mathbf{rnd} \cdot \left(\mathbf{x}_{\mathrm{pb}} - \mathbf{x} \right) + \frac{\varphi}{2} \cdot \mathbf{rnd} \cdot \left(\mathbf{x}_{\mathrm{gb}} - \mathbf{x} \right) \right]. \tag{7}$$

Na przebieg procesu uczenia opisanego wzorem (7) mają wpływ dwa współczynniki, współczynnik uczenia (ang. cognition factor) φ oraz współczynnik ograniczający χ (ang. constriction coefficient). Współczynnik ograniczający jest równocześnie współczynnikiem bezwładności (ang. inertia weight) [19,20]. Wynika to z faktu, że nowa wartość prędkości $\mathbf{v}$ (po lewej stronie równania) jest zależna od iloczynu $\chi \cdot \mathbf{v}$ (po prawej stronie równania). Z kolei współczynnik uczenia określa jak duży wpływ na prędkość danej cząstki mają inne cząstki w roju o lepszych wartościach funkcji celu oraz zapamiętane poprzednie najlepsze położenie tej cząstki. Od wzajemnych relacji pomiędzy współczynnikiem uczenia a współczynnikiem ograniczającym zależą między innymi stabilność algorytmu oraz liczba iteracji potrzebna do uzyskania zbieżności. Wartości współczynników φ i χ można zadawać niezależnie od siebie. Jest to jednak trudne, gdyż przy nieodpowiedniej kombinacji ich wartości algorytm może utracić stabilność [20]. W opisanym tutaj algorytmie zastosowano więc metodę zaproponowaną w [19,17], polegającą na wzajemnym powiązaniu obydwu współczynników zgodnie z zależnością:

$$\chi = \frac{2}{\varphi - 2 + \sqrt{\varphi^2 - 4\varphi}}. \tag{8}$$

We wzorze (7) **rnd** oznacza macierz diagonalną rzędu M = 4 o wartościach losowych, dodatkowo zależnych od porównania wartości funkcji celu:

$$\mathbf{rnd} = \begin{cases} \mathrm{diag}\left(\rho_1, \rho_2 \ldots \rho_M\right) & \text{gdy} \quad F\left(\mathbf{x}_{\mathrm{ref}}\right) > F\left(\mathbf{x}\right) \\ \mathbf{0}_{M \times M} & \text{gdy} \quad F\left(\mathbf{x}_{\mathrm{ref}}\right) \le F\left(\mathbf{x}\right) \end{cases} \tag{9}$$

Gdy wartość funkcji celu odpowiadająca położeniu $\mathbf{x}$ rozpatrywanej cząstki jest lepsza (czyli w rozważanym przypadku mniejsza) od wartości funkcji celu odpowiadającej położeniu odniesienia $\mathbf{x}_{\mathrm{ref}}$, wtedy **rnd** jest macierzą diagonalną rzędu M o wartościach losowych ρ z przedziału od 0 do 1. Każda wartość ρ jest losowana osobno. Gdy wartość funkcji celu dla położenia odniesienia jest gorsza, wtedy **rnd** jest macierzą zerową. Oznacza to, że składniki równia (7) zawierające macierz **rnd** mają wpływ na obliczaną prędkość cząstki tylko gdy wartość funkcji celu odpowiadająca położeniu $\mathbf{x}$ jest gorsza od najlepszej zapamiętanej, lub odpowiednio, gorsza od wartości funkcji celu dla najlepszej cząstki w roju. W wzorze (7), odniesieniem $\mathbf{x}_{\mathrm{ref}}$ są kolejno $\mathbf{x}_{\mathrm{pb}}$, czyli zapamiętane, najlepsze ze wszystkich dotychczasowych położeń cząstki oraz $\mathbf{x}_{\mathrm{gb}}$, czyli położenie cząstki o najlepszej wartości funkcji celu w całym roju.

Badania wykazały [17], że algorytm oparty na metodzie uczenia GB ma tendencję do przedwczesnej zbieżności i przez to stosunkowo niską sprawność odnajdowania ekstremum globalnego funkcji celu. Lepsze właściwości pod tym względem przejawia metoda LB, opisana równaniem:

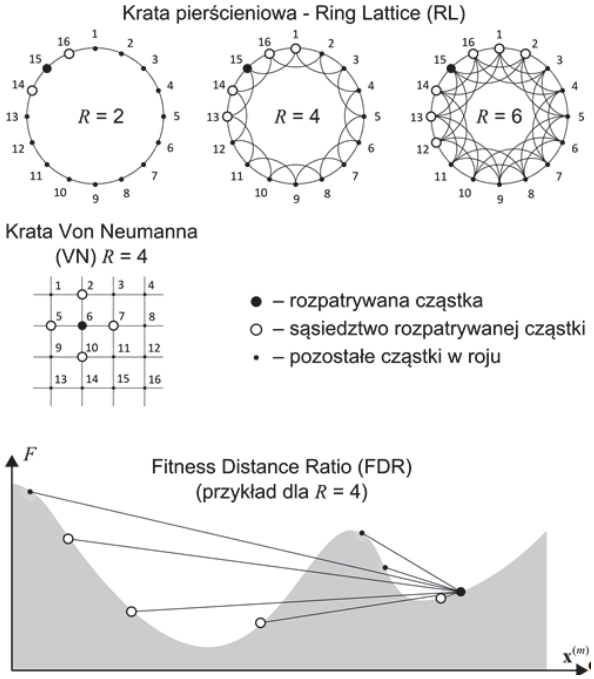
$$\mathbf{v} \leftarrow \chi \cdot \left[\mathbf{v} + \frac{\varphi}{2} \cdot \mathbf{rnd} \cdot \left(\mathbf{x}_{\mathrm{pb}} - \mathbf{x} \right) + \frac{\varphi}{2} \cdot \mathbf{rnd} \cdot \left(\mathbf{x}_{\mathrm{lb}} - \mathbf{x} \right) \right]. \tag{10}$$

Jedyną różnicą w stosunku do równania (7) jest tutaj odniesienie zastosowane w drugim składniku losowym, $\mathbf{x}_{\mathrm{lb}}$ zamiast $\mathbf{x}_{\mathrm{gb}}$. Jest to położenie cząstki o najlepszej wartości funkcji celu, ale nie w całym roju, tylko w jego określonym fragmencie, zwanym sąsiedztwem S rozważanej cząstki o położeniu $\mathbf{x}$. To w jaki sposób jest wybierane sąsiedztwo, oraz jaki jest jego rząd R, czyli liczba cząstek do niego należących, jest określone przez topologię roju.

Kolejnym rozwinięciem metody uczenia, jest metoda FIPS (ang. Fully Informed Particle Swarm) opisana wzorem [17]:

$$\mathbf{x} \leftarrow \chi \cdot \left[\mathbf{v} + \frac{\varphi}{R+1} \cdot \mathbf{rnd} \cdot \left(\mathbf{x}_{\mathrm{pb}} - \mathbf{x} \right) + \frac{\varphi}{R+1} \cdot \mathbf{rnd} \cdot \sum_{r \in S} \left(\mathbf{x}_r - \mathbf{x} \right) \right]. \tag{11}$$

Tak jak w przypadku LB, proces uczenia jest oparty nie na całym roju lecz na sąsiedztwie S danej cząstki. Różnica polega na tym, że w przypadku FIPS w procesie uczenia jest wykorzystywane wszystkie R cząstek w sąsiedztwie, a nie tylko jedna, o najlepszej wartości funkcji celu.



Rys.3. Wybrane topologie roju algorytmu PSO

Topologia roju

Topologia roju określa wzajemne powiązania pomiędzy cząstkami, czyli sposób w jaki wybierane jest sąsiedztwo. W historycznie pierwszych wersjach algorytmów PSO o przynależności do sąsiedztwa decydowała odległość pomiędzy cząstkami w przestrzeni poszukiwań. Podejście takie okazało się jednak mało efektywne [17] i jednocześnie kosztowne pod względem nakładu obliczeń, dlatego obecnie nie jest stosowane. Obecnie wyróżniamy dwa podstawowe typy sąsiedztw, statyczne i dynamiczne.

Sąsiedztwo statyczne polega na tym, że dla danej cząstki raz, przed uruchomieniem algorytmu, ustalamy zbiór S zawierający R cząstek, i w trakcie działania algorytmu zbiór ten nie ulega zmianie. W rozpatrywanym algorytmie zastosowano sąsiedztwa statyczne o topologii kraty pierścieniowej różnego rzędu oraz dwuwymiarową topologię Von Neumanna (rys. 3) [20,21]. Należy zauważyć, że o zaliczeniu do sąsiedztwa decydują tylko i wyłącznie arbitralnie nadawane numery cząstek. W związku z tym,

sąsiadami mogą być cząstki położone na przeciwległych krańcach przestrzeni poszukiwań, podczas gdy dwie inne cząstki położone blisko siebie sąsiadami nie będą. Sąsiedztwu statycznemu można nadać cechy dynamiczne, co zadaną liczbę iteracji algorytmu Δi losując cząstkom nowe numery, tym samym zmieniając ich szyk w roju i zmieniając sąsiedztwo. W rozpatrywanym algorytmie zaimplementowano takie rozwiązanie.

W odróżnieniu do statycznego, sąsiedztwo dynamiczne jest na nowo ustalane w każdej iteracji algorytmu. W związku z tym, algorytmy oparte na tego typu sąsiedztwie wymagają większego nakładu obliczeń. Obecnie najbardziej rozpowszechnionym rodzajem sąsiedztwa dynamicznego jest FDR (ang. Fitness Distance Ratio) [22,20]. Wybierając sąsiedztwo danej cząstki o położeniu $\mathbf{x}$, należy najpierw dla wszystkich pozostałych cząstek w roju (o położeniach $\mathbf{x}_n$) wyznaczyć wartość współczynnika w, osobno dla każdego m-tego wymiaru przestrzeni poszukiwań:

$$w_n^{(m)} = \frac{F\left(\mathbf{x}^{(m)}\right) - F\left(\mathbf{x}_n^{(m)}\right)}{\left|\mathbf{x}^{(m)} - \mathbf{x}_n^{(m)}\right|}, \tag{12}$$

gdzie $\mathbf{x}^{(m)}$ oznacza m-ty element wektora $\mathbf{x}$, a n to numer cząstki w roju, różny od numeru cząstki dla której ustalane jest sąsiedztwo. Następnie, do sąsiedztwa należy zaliczyć R cząstek o największych wartościach współczynnika w. Sąsiedztwo ustala się dla każdego wymiaru przestrzeni poszukiwań z osobna, więc jeżeli wymiar sąsiedztwa R wynosi na przykład 4, a optymalizowane są 3 zmienne, to każda cząstka ma 12 sąsiadów. W związku z tym, jeżeli stosowana jest metoda uczenia FIPS, występującą we wzorze (11) wartość R należy dodatkowo pomnożyć przez liczbę wymiarów przestrzeni poszukiwań.

Warunki początkowe i ograniczenia

Dla każdej z optymalizowanych zmiennych a, b, c i d ustalany jest indywidualny zakres wartości w ramach którego poszukiwane jest rozwiązanie. Następnie zakres ten jest normalizowany do przedziału od 0 do 1, więc przestrzeń poszukiwań ma postać czterowymiarowego hipersześcianu o boku długości 1. Początkowy rój cząstek, dla iteracji algorytmu i = 0, jest losowo generowany wewnątrz tego sześcianu. W trakcie kolejnych iteracji algorytmu, gdy położenia cząstek są aktualizowane zgodnie z formułą (6), może się zdarzyć, że cząstki opuszczą założoną przestrzeń poszukiwań, czyli że jeden lub więcej elementów wektora $\mathbf{x}$ osiągnie wartość mniejszą od 0 lub większą od 1. W rozpatrywanym algorytmie można tego uniknąć włączając sztywne ograniczenie położenia w przestrzeni poszukiwań. Podobnie można ograniczyć maksymalną wartość prędkości, zadając maksymalną wartość modułu wektora prędkości $\mathbf{v}$. W przypadku roju początkowego (dla i = 0) można wybrać, czy początkowa prędkość ma być zerowa ($\mathbf{v} = \mathbf{0}_{4\times1}$) czy losowa.

Zbieżność i stabilność

Każda cząstka ma położenie i prędkość, które są od siebie zależne. W związku z tym każda cząstka jest systemem dynamicznym. Systemem dynamicznym jest więc również rój złożony z wzajemnie powiązanych cząstek. Pojawia się więc tutaj problem stabilności [20, 17]. O właściwościach dynamicznych cząstki, i pośrednio roju, decyduje wartość współczynnika uczenia φ i współczynnika ograniczającego χ. W literaturze przedmiotu [17] stwierdza się, że dla współczynnika ograniczającego χ powiązanego ze współczynnikiem uczenia φ wzorem (8), rój traci stabilność dla φ mniejszego niż około 4, przy czym dokładna wartość graniczna stabilności zleży od przyjętej metody uczenia, topologii roju oraz przyjętych ograniczeń. Gdy rój jest niestabilny, wtedy wraz z kolejnymi iteracjami algorytmu jego cząstki oddalają się od siebie w tempie wykładniczym, a ich prędkości rosną w podobny sposób. Uzyskanie zbieżności algorytmu nie jest więc możliwe. Gdy algorytm uzyskuje zbieżność, wtedy wraz z kolejnymi iteracjami, prędkości cząstek oraz odległości pomiędzy nimi asymptotycznie maleją do zera. W tym artykule umownie przyjęto, że algorytm uzyskuje zbieżność w pierwszej iteracji i, w której zarówno średnia prędkość cząstek v_{av} jak i średnia odległość pomiędzy cząstkami w roju d_{av} spadają do wartości mniejszych niż 0,02.

Eksperymenty z doborem wzmocnień obserwatora

Celem przeprowadzonych badań było znalezienie konfiguracji algorytmu PSO najlepszej z punktu widzenia rozpatrywanego problemu doboru wzmocnień obserwatora zmiennych stanu silnika indukcyjnego. Przede wszystkim, algorytm powinien zapewniać wysoką sprawność znajdowania rozwiązania o możliwie najmniejszej wartości funkcji celu F, jednocześnie oferując wysoką powtarzalność – kolejne uruchomienia algorytmu powinny zwracać jak najbardziej zbliżone wyniki. Drugorzędnym parametrem jest tutaj czas obliczeń, na który składają się czas przetwarzania jednej iteracji oraz liczba iteracji algorytmu koniecznych do uzyskania zbieżności.

Badania przeprowadzono w następujący sposób. Dla wybranej konfiguracji uruchamiano algorytm 50 razy. Zebrane wyniki z 50 uruchomień stanowią jeden test. Takich testów dla różnych konfiguracji przeprowadzono wiele, tutaj zaprezentowano 20 wybranych, które podsumowano w tabeli 1. Dodatkowo, szczegółowe wyniki 4 wybranych testów przedstawiono na rysunkach od 4 do 7.

Na rysunku 4 przedstawiono wyniki testu 1 (tab. 1). Wykresy po lewej stronie obrazują przebieg procesu optymalizacji, wraz z kolejnymi iteracjami i. Dane przedstawiono w postaci słupków ilustrujących rozkład wartości dla 50 uruchomień algorytmu PSO. Strzałką zaznaczono umowny moment osiągnięcia zbieżności. W teście 1 zbieżność została osiągnięta stosunkowo szybko, już w 25 iteracji. Wynika to z zastosowania topologii FDR razem z metodą uczenia FIPS. Są to dwie najbardziej zaawansowane z przebadanych metod, wymagające jednak największego nakładu obliczeń. Widać to po wartości uzyskanego względnego czasu obliczeń (tabela 1), wynoszącego tutaj ponad 30, co oznacza, że czas obliczeń był trzydziestokrotnie dłuższy niż dla najszybszego z przeprowadzonych testów (test 6). Po prawej stronie rysunku 4 przedstawiono zależności części rzeczywistych i urojonych wartości własnych obserwatora w funkcji prędkości kątowej silnika, obliczone na podstawie uzyskanych w wyniku optymalizacji wzmocnień a, b, c i d. Na wykresach przestawiono nałożone na siebie wartości uzyskane dla wszystkich 50 uruchomień algorytmu. Jest to dobra miara powtarzalności algorytmu, gdyż wartości własne obserwatora są bardzo wrażliwe na zmiany wzmocnień. Jeżeli więc krzywe dla poszczególnych doborów się pokrywają, to znaczy, że w wyniku optymalizacji uzyskiwano bardzo zbliżone wyniki.

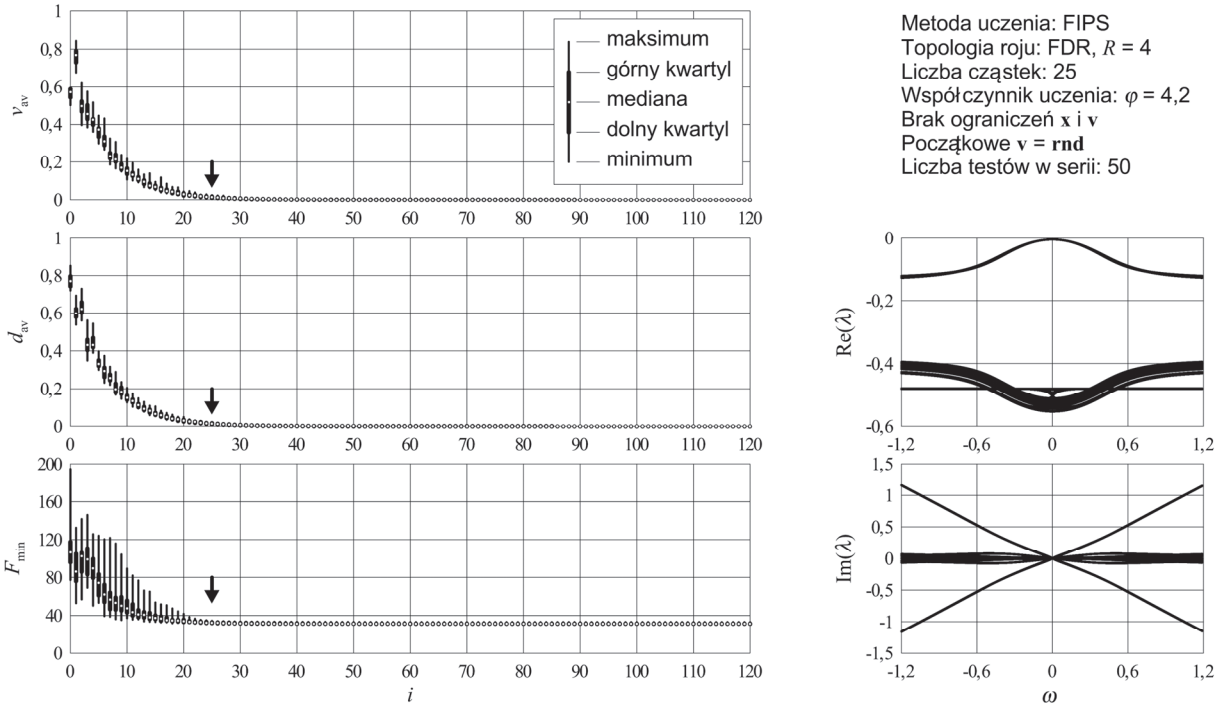
Powtarzalność algorytmu można również ocenić na podstawie podanych w tabeli 1 wartości minimalnej i maksymalnej funkcji celu. W przypadku testu 1 przedział ten jest stosunkowo szeroki, od 31,18 do 33,14. Dużo lepszy wynik uzyskano w przypadku testu 17, od 31,19 do 31,23. Również porównanie wykresów wartości własnych dla testu 17 (rys. 7) z wykresami dla testu 1 (rys. 4), uwidacznia lepszą powtarzalność algorytmu w teście 17. W tym przypadku korzystniejsze okazało się późniejsze osiągniecie zbieżności (62 iteracja), osiągnięte poprzez

zadanie niższej wartości współczynnika uczenia φ (4,1 w stosunku do 4,2). Należy jednak zaznaczyć, że wartość współczynnika uczenia na poziomie 4,1 nie gwarantuje stabilności algorytmu w każdych warunkach. O ile w teście 17 (rys. 7) algorytm był stabilny, to w teście 11 (rys. 5), po zmianie metody uczenia z FIPS na LB oraz topologii z VN na RL, czyli metody mniej zaawansowane, algorytm utracił stabilność. Na rysunku 5 widoczne jest narastanie odległości między cząstkami oraz wartości ich prędkości wraz z kolejnymi iteracjami, co oznacza, że cząstki roju zamiast dążyć w stronę minimum funkcji celu, rozbiegają się po przestrzeni poszukiwań w całkowicie losowy sposób. W teście 12 (rys. 6) wykorzystano tą samą topologię, metodę uczenia oraz wartość współczynnika uczenia co w teście 11, lecz wprowadzono ograniczenia. W odróżnieniu do testu 11 cząstki nie mogą teraz opuścić założonej przestrzeni poszukiwać (wartości elementów wektora **x** są ograniczone do przedziału od 0 do 1), a moduł wektora prędkości **v** nie może przekroczyć wartości 0,5.

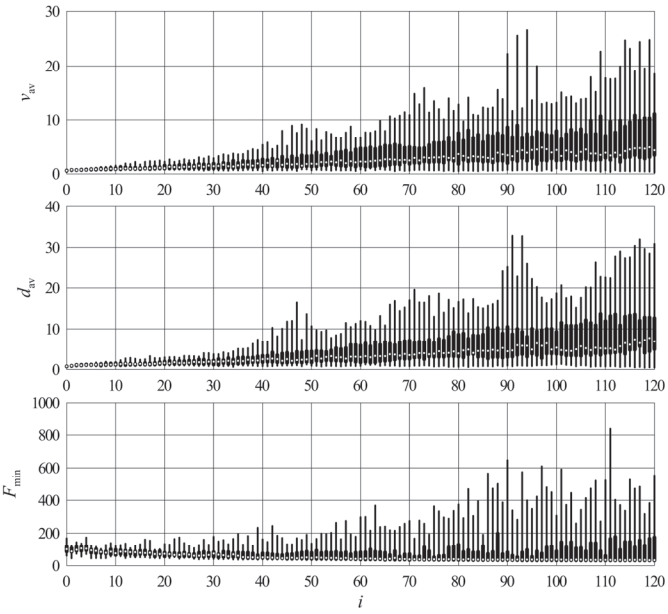
Wprowadzenie ograniczeń ustabilizowało proces optymalizacji. Na rysunku 6 widoczne są malejące wartości zarówno prędkości jak i odległości pomiędzy cząstkami, algorytm dąży więc do osiągnięcia zbieżności. Zbieżność nie została jednak osiągnięta w ciągu wykonanych 120 iteracji.

Nr	Metoda uczenia	Topologia	R	N	φ	Limit	$\mathbf{v}_{i=0}$	Wartość F najlepszej cząstki w roju dla iteracji i = 120			Zbieżność	Czas iteracji t / $t_{\min}$
								$\min(F_{\min})$	$\mathrm{M}(F_{\min})$	$\max(F_{\min})$		
1	FIPS	FDR	4	25	4,2	-	**rnd**	31,1808	31,2177	33,1389	i = 25	30,7
2	LB	FDR	4	25	4,2	-	**rnd**	31,1833	31,2068	31,4620	i = 50	24,7
3	GB	-	-	16	4,2	-	**rnd**	31,1824	31,2118	31,4725	i = 47	1,6
4	GB	-	-	25	4,2	-	**rnd**	31,1815	31,1948	31,2658	i = 44	2,8
5	GB	-	-	36	4,2	-	**rnd**	31,1818	31,1854	31,2339	i = 44	3,7
6	GB	-	-	9	4,2	-	**rnd**	31,1828	31,3031	46,6434	i = 51	1
7	FIPS	RL	4	36	4,2	-	**rnd**	31,1811	31,1908	31,3188	i = 49	6,6
8	FIPS	RL	4	36	4,5	-	**rnd**	31,1839	31,8440	34,0821	i = 31	6,3
9	LB	RL	4	25	4,2	-	**rnd**	31,1831	31,2298	31,3524	i = 79	4,1
10	LB	RL	4	36	4	-	**rnd**	4,98e+3	2,18e+5	5,90e+6	Brak	5,2
11	LB	RL	4	36	4,1	-	**rnd**	31,3985	33,6991	553,406	Brak	5,3
12	LB	RL	4	36	4,1	**v** i **x**	**rnd**	31,3873	31,8802	33,6809	i > 120	5,4
13	LB	RL	4	36	4,15	-	**rnd**	31,2087	31,3701	31,7950	i > 120	5,2
14	LB	RL Δi = 10	4	36	4,15	-	**rnd**	31,2578	31,3674	33,6104	i > 120	5,2
15	LB	RL	4	36	4,2	-	**rnd**	31,1852	31,2118	31,3046	i = 80	5,3
16	LB	RL	8	36	4,2	-	**rnd**	31,1829	31,1976	31,3812	i = 63	6,8
17	FIPS	VN	4	36	4,1	-	**0**	31,1851	31,1933	31,2334	i = 62	5,9
18	FIPS	VN	4	36	4,1	-	**rnd**	31,1858	31,2139	31,2633	i = 67	5,9
19	FIPS	VN	4	36	4,15	-	**rnd**	31,1810	31,1858	31,2512	i = 46	6,0
20	FIPS	VN	4	36	4,2	-	**rnd**	31,1808	31,2045	31,6111	i = 36	6,3

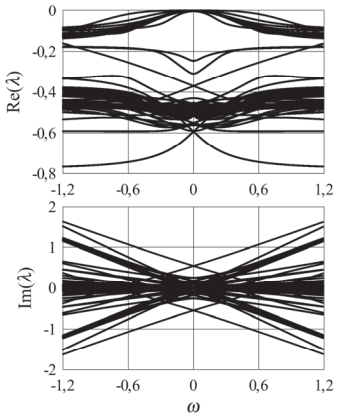
Tabela 1. Podsumowanie wyników 20 testów algorytmu PSO, każdy test to 50 uruchomień algorytmu



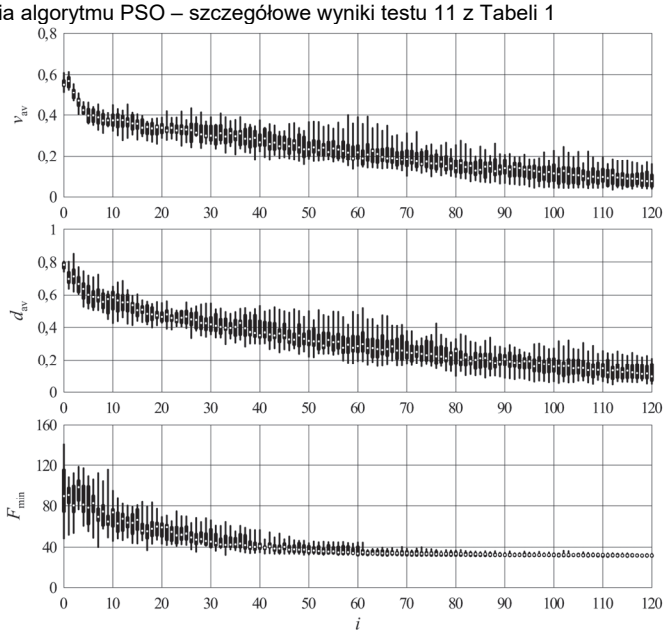
Rys.4. Badania algorytmu PSO – szczegółowe wyniki testu 1 z Tabeli 1



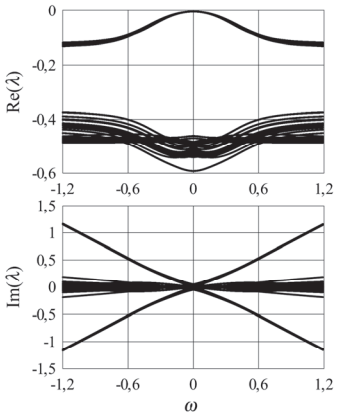
Metoda uczenia: LB
Topologia roju: RL, R = 4
Liczba cząstek: 36
Współczynnik uczenia: φ = 4,1
Brak ograniczeń **x** i **v**
Początkowe **v** = **rnd**
Liczba testów w serii: 50



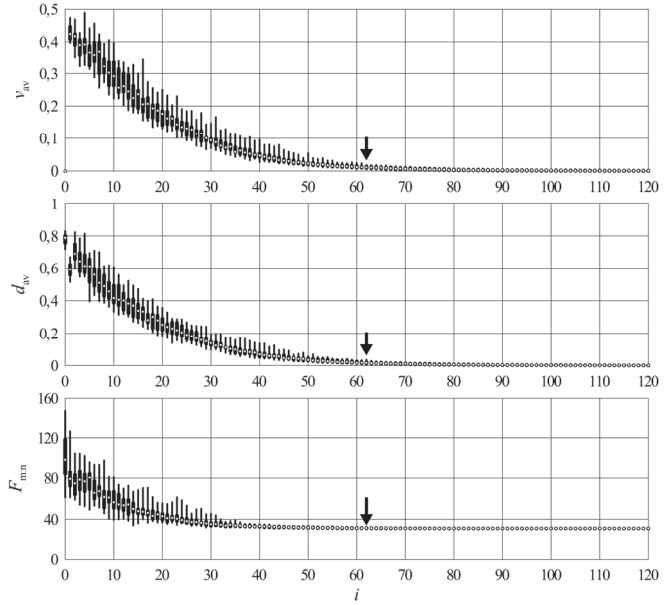
Rys.5. Badania algorytmu PSO – szczegółowe wyniki testu 11 z Tabeli 1



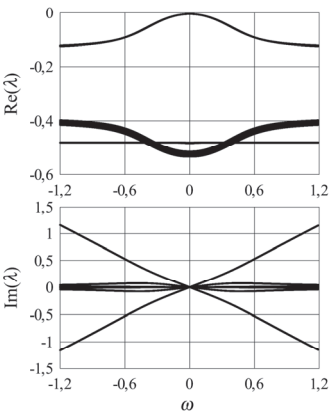
Metoda uczenia: LB
Topologia roju: RL, R = 4
Liczba cząstek: 36
Współczynnik uczenia: φ = 4,1
Ograniczenia 0 < **x** < 1 i **v** < 0,5
Początkowe **v** = **rnd**
Liczba tesów w serii: 50



Rys.6. Badania algorytmu PSO – szczegółowe wyniki testu 12 z Tabeli 1



Metoda uczenia: FIPS
Topologia roju: VN, R = 4
Liczba cząstek: 36
Współczynnik uczenia: φ = 4,1
Brak ograniczeń **x** i **v**
Początkowe **v** = 0
Liczba testów w serii: 50



Rys.7. Badania algorytmu PSO – szczegółowe wyniki testu 17 z Tabeli 1

Na podstawie wyników pozostałych testów, zamieszczonych w tabeli 1, można stwierdzić co następuje. Wartość współczynnika uczenia większa niż 4,2 gwarantuje stabilność algorytmu i uzyskanie zbieżności w mniej niż 120 iteracjach. Dla niektórych nastawie (np. testy 1, 7 i 20) algorytm może dla tej wartości uzyskać zbieżność zbyt szybko (nawet w 25 iteracjach), co źle wpływa na jego powtarzalność. Dlatego dla bardziej zaawansowanych metod, w szczególności dla metody uczenia FIPS, korzystniejsze jest stosowanie niższej wartości współczynnika uczenia, na poziomie 4,15 lub nawet 4,1.

Zastosowanie topologii FDR znacząco wydłuża czas obliczeń w stosunku do topologii RL i VN. Zastosowanie dynamicznych topologii VN i RL (poprzez reorganizację roju co zadaną liczbę iteracji Δi) nie przyniosło widocznej poprawy uzyskiwanych wyników (testy 13 i 14). Zwiększanie rzędu topologii RL poprawia jakość uzyskanych wyników. W przypadku testów 15 i 16, zwiększenie rzędu topologii z 4 do 8 przyspieszyło moment uzyskania zbieżności przy jednoczesnej nieznacznej poprawie uzyskiwanych wartości minimalnych funkcji celu. Zwiększanie liczy cząstek w roju (testy od 3 do 6) nieznacznie przyspiesza moment uzyskania zbieżności, jednocześnie poprawiając powtarzalność uzyskiwanych wyników, powoduje jednak proporcjonalny wzrost czasu obliczeń.

Podsumowanie i wnioski

Przeprowadzone badania umożliwiły znalezienie optymalnych parametrów algorytmu PSO dla rozpatrywanego problemu doboru wzmocnień obserwatora. Za najlepsze uznano nastawy dla testu 17 z tabeli 1. Wykazano, że poprawnie sparametryzowany algorytm PSO zawsze osiąga zbieżność, jednocześnie zapewniając wysoką powtarzalność uzyskiwanych wyników. Uzyskane w wyniku optymalizacji obserwatory PIrR poddano badaniom laboratoryjnym, których wyniki opisano w pracy [13]. Przeprowadzone badania laboratoryjne udowodniły poprawną pracę uzyskanych obserwatorów.

Autor: *dr inż. Tadeusz Białoń, Politechnika Śląska, Wydział Elektryczny, Katedra Elektrotechniki i Informatyki, ul. Akademicka 10, 44-100 Gliwice, E-mail: tadeusz.bialon@polsl.pl*

LITERATURA

[1] Laatra Y., Lotfi H., Abdelhani B., Speed Sensorless Vector Control of Induction Machine with Luenberger observer and Kalman Filter, *Proceedings of International Conference on Control, Decision and Information Technologies (CoDIT'17)*, (2017), 5–7

[2] Wang F., Zhang Z., Mei X., Rodríguez J., Kennel R., Advanced Control Strategies of Induction Machine: Field Oriented Control, Direct Torque Control and Model Predictive Control, *Energies*, 11 (2018), No. 1:120, 1-13

[3] Kuchar M., Palacky P., Simonik P., Strossa J., Self-Tuning Observer for Sensor Fault-Tolerant Control of Induction Motor Drive, *Energies*, 14 (2021), No. 9:2564, 1-16

[4] Toumi D., Segueir Boucherit M., Tadjine M., Observer-based fault diagnosis and field oriented fault tolerant control of induction motor with stator inter-turn fault, *Arch. Electrical Eng.*, 61 (2012), No. 2, 165-188

[5] Kubota H., Matsuse K., Nakano T., DSP-based speed adaptive flux observer of induction motor, *IEEE Trans. Ind. Appl.*, 29 (1993), No. 2, 344-348.

[6] Białoń T., Pasko M., Niestrój R., Developing Induction Motor State Observers with Increased Robustness, *Energies*, 13 (2020), No. 20:5487, 1-24

[7] Azzoug V., Menacer A., Pusca R., Romary R., Ameid T., Ammar A., Fault Tolerant Control for Speed Sensor Failure in Induc-tion Motor Drive based on Direct Torque Control and Adaptive Stator Flux Observer, *Proceedings of International Conference on Applied and Theoretical Electricity (ICATE)*, (2018), 1-6

[8] Zaky M.S., Khater M., Yasin H., Shokralla, S.S., Review of different speed estimation schemes for sensorless induction motor drives, *JEE*, 8 (2017), 102-140

[9] Kadrine A., Tir Z., Malik O.P., Hamida M.A., Reatti A., Houari A., Adaptive non-linear high gain observer based sensor-less speed estimation of an induction motor, *J. Franklin Inst.*, 357 (2020), 8995-9024

[10] Najafabadi T. A., Salmasi F. R., Jabehdar-Maralani P., Detection and Isolation of Speed-, DC-Link Voltage-, and Current-Sensor Faults Based on an Adaptive Observer in Induction-Motor Drives, *IEEE Trans. Ind. El.*, 58 (2011), No. 5, 1662-1672

[11] Pimkumwong N., Wang M.S., Full-order observer for direct torque control of induction motor based on constant V/F control technique, *ISA Trans*., 73 (2018), 189-200

[12] Hussein A.A., Salih S.S., Ghasm Y.G., Implementation of Proportional-Integral-Observer Techniques for Load Frequency Control of Power System, *Proceedings of International Conference on Ambient Systems, Networks and Technologies, ANT-2017 and the 7th International Conference on Sustainable Energy Information Technology, SEIT 2017*, (2017), 754-762

[13] Białoń T., Niestrój R., Michalak J., Pasko M., Induction Motor PI Observer with Reduced-Order Integrating Unit, *Energies*, 14 (2021), No. 16:4906, 1-12

[14] Białoń T., Pasko M., Niestrój R., Developing Induction Motor State Observers with Increased Robustness, *Energies*, 13 (2020), No. 20:5487, 1-24

[15] Krzemiński Z., Lewicki A., Morawiec M., Speed observer based on extended model of induction machine, *Proceedings of IEEE International Symposium on Industrial Electronics*, (2010) 3017-3112

[16] Białoń T., Lewicki A., Pasko M., Niestrój R., Non-proportional full-order Luenberger observers of induction motors, *Arch. Electrical Eng*, 67 (2018), 925-937

[17] Poli R.,·Kennedy J.,·Blackwell T., Particle swarm optimization: An overview, *Swarm Intell*, 1 (2007), 33-57

[18] Li W., Fan Y., Jiang Q., Xu Q., Velocity-Driven Particle Swarm Optimization, *Proceedings of the International Conference on Computing and Pattern Recognition (ICCPR '19). Association for Computing Machinery*, (2019), 9-16

[19] Clerc M., Kennedy J., The particle swarm – explosion, stability, and convergence in a multidimensional complex space. *IEEE Transaction on Evolutionary Computation*, 6 (2002), No. 1, 58-73

[20] Cleghorn C.W., Engelbrecht A.P., Fitness-distance-ratio particle swarm optimization: stability analysis, *Proceedings of the Genetic and Evolutionary Computation Conference*, (2017), 12-18

[21] Fernandes C.M., Rosa A.C., Fachada N., Laredo J.L.J., Merelo J. J., Particle swarm and population structure, *Proceedings of the Genetic and Evolutionary Computation Conference Companion. Association for Computing Machinery*, (2018), 85-86

[22] Peram T., Veeramachaneni K., Mohan C.K., Fitness-distance-ratio based particle swarm optimization, *Proceedings of the 2003 IEEE Swarm Intelligence Symposium*, (2003), 174-181