Politechnika Warszawska

WYDZIAŁ MECHATRONIKI

Instytut Metrologii i Inżynierii Biomedycznej

Praca dyplomowa magisterska

na kierunku Inżynieria Biomedyczna
w specjalności Aparatura Medyczna

Narzędzie do modelowania szeregów czasowych pochodzących od systemów o nieliniowej dynamice metodą NARMAX

numer pracy według wydziałowej ewidencji prac {liczba}

Aleksander Cudny
Numer albumu 242830

promotor
dr inż. Miłosz Jamroży

Warszawa, 2017

Karta pracy (osobna kartka)

Streszczenie

Celem niniejszej pracy jest implementacja oraz testowanie narzędzia do przeprowadzania modelowania sygnałów pochodzących od systemów o nieliniowej dynamice metodą NARMAX.

Model NARMAX (*ang. Non-linear Auto-Regressive Moving Average model with eXogenous input*), oznacza nieliniowy (N) model autoregresyjny (AR) ze średnią ruchomą (MA) oraz zewnętrznym wejściem (X). Oznacza to, że wartość modelowanego sygnału w danej chwili może zależeć od: wartości sygnału w przeszłości (człon AR), wartości szumu z przeszłości (człon MA), wartości innego sygnału zewnętrznego z możliwym opóźnieniem (człon X) oraz że wszystkie te zależności mogą być nieliniowe.

Poszukiwanie modelu tego typu zrealizowano za pomocą algorytmu FROLS (*ang. Forward Regression with Orthogonal Least Squares estimator*), czyli algorytmu regresji w przód z wykorzystaniem estymatora ortogonalnej sumy najmniejszych kwadratów. Algorytm opiera się na konstruowaniu modelu składającego się z ortogonalnych składowych, który w jak najlepszy sposób ma odwzorowywać modelowany sygnał. Miarą tego odwzorowania jest *ERR* (*ang. Error Reduction Ratio)*, czyli stopień obniżenia błędu pomiędzy tworzonym modelem ortogonalnym a modelowanym sygnałem. Modelowanie rozpoczyna tworzenie wektora regresorów, czyli potencjalnych składników modelu. Wektor zawiera opóźnione sygnały związane z członami AR i X, oraz wszystkie ich warianty wedle założonej nieliniowości. W niemniejszej pracy brano pod uwagę jedynie wielomiany, wobec tego w ramach wektora regresorów tworzone są wariacje z powtórzeniami opóźnionych sygnałów członów AR i X o maksymalnej ilości składników równej najwyższemu założonemu stopniu wielomianu. Następnie, z wektora regresorów kolejno wybierane są elementy, które cechują się najwyższym *ERR*. Wybrany element traktowany jest następnie jako kolejny człon zortogonalizowanego modelu. Elementy są wybierane aż do otrzymania satysfakcjonującego wyniku. Finalnie, otrzymany model ortogonalny jest transformowany do modelu ogólnego, który jest zwracany jako wynik modelowania. Składowymi otrzymanego modelu są wybrane regresory, wraz z ich wyznaczonymi parametrami. Do algorytmu wprowadzona została także modyfikacja oceny wartości *ERR*, która usprawnia przeprowadzanie modelowania.

W celu zrealizowania powyższej metody zaimplementowano oprogramowanie w języku *Python,* wraz z graficznym interfejsem użytkownika, które umożliwia krokową realizację algorytmu FROLS. Zaimplementowano także metody wyznaczania momentu końca modelowania, oceny poprawności modelu oraz jego walidacji. Parametry te oceniane są na bazie przebiegu sumy *ERR* modelu, autokorelacji różnic pomiędzy opracowanym modelem a sygnałem modelowanym oraz symulacji przyszłych wartości sygnału według wyznaczonego modelu. Program pozwala na wyznaczenie tych parametrów oraz wyświetlenie wykresów ich przebiegów.

W celu wykonania testów narzędzia zostało przeprowadzone modelowanie trzech sygnałów: liniowego stochastycznego, nieliniowego stochastycznego oraz funkcji logistycznej. Miarą poprawności przeprowadzonego modelowania było wyznaczenie poprawnych członów tworzących model oraz różnice pomiędzy obliczonymi a oryginalnymi parametrami. Zbadano wpływ długości sygnału oraz wariancji szumu obecnego w sygnale na wynik modelowania. Sprawdzono jak wykonana modyfikacja wypływa na wynik modelowania i otrzymywane parametry. Zbadano także jakie są różnice w czasie wykonywania pojedynczego kroku modelowania dla algorytmu podstawowego i algorytmu zmodyfikowanego.

Abstract

The objective of the thesis is to implement and test a tool dedicated to perform the NARMAX modelling of signals originated from systems with a non-linear dynamic.

NARMAX stands for *Non-linear (N) Auto-Regressive (AR) Moving Average (MA) model with an exogenous input (X)*. It means, that a current value of the signal could be calculated from the previous values of the signal (AR module), previous values of a noise that contributed to the signal (MA module), values of an exogenous signal (X module) and all of these influences could be described by a non-linear function.

Calculation of a NARMAX model was carried out with a FROLS algorithm, what means *Forward Regression algorithm with the Orthogonal Least Squares estimator*. Its core consists of an orthogonal model, which is composed during the process in order to provide the best representation of the modelled signal. Quality of the model is measured with the *ERR*, what stands for *Error Reduction Ratio*. It is a coefficient calculated using a variance of the modelled signal and the computed model outcome. It measures how well the signal is represented by the model and how the model elements contributed to this outcome. The modelling process starts with a composition of a regressors vector. A regressor is a potential signal that could be a part of the model. It is calculated from the lagged AR and X modules, and their variations within the assumed non-linearity. In the thesis, only the polynomial nonlinearity was considered. The regressors were computed as combinations with replacements of the maximal order equal to the highest assumed polynomial level. All regressors are stored in a vector and ranked by their *ERR* value. A modelling step requires choosing an element with the highest *ERR* and merging its orthogonalised version with the orthogonal model. Until the outcome is satisfactory, the selection process is repeated and further elements are joined with the model. A decision about the process termination is made based on the characteristic of the sum of the *ERR* values of the model components. Finally, the orthogonal model is transformed into the output model, which consists of the selected regressors with their parameters. Also, an algorithm modification was proposed. It concerns the modification of the regressor ranking, and its application improved the obtained results.

In order to perform this kind of modelling, a *Python* program with a graphical user interface was implemented. The program enables the user to execute the FROLS algorithm on the selected signal. Also, the program is able to help with the model evaluation and the model validation. The evaluation of is performed by the analysis of the autocorrelation of residuals between the modelled signal and the outcome of the computed model. The validation of the model consists of a comparison between the future signal values, and the future samples calculated from the procured model. The program also can plot the sum of the *ERR* values during the modelling process in order to determine the appropriate length of the model.

For the testing purpose, the modelling of three signals: a stochastic linear signal, a stochastic nonlinear signal and a logistic function, was carried out. The quality of the obtained models was assessed basing on the type of the selected regressors (were they correct?) and on the values of the procured parameters. Additionally, a research was conducted in order to determine how the signal length and the variance of the noise present in the modelled signal, influenced the modelling outcome. Also, an effectiveness of the modified FROLS algorithm was examined in the same research conditions. Finally, the impact on the computation time of the algorithm modification was measured.

Oświadczenie o samodzielności pracy (osobny pdf)

Oświadczenie

Oświadczam, że zachowując moje prawa autorskie udzielam Politechnice Warszawskiej nieograniczonej w czasie, nieodpłatnej licencji wyłącznej do korzystania z przedstawionej dokumentacji niniejszej pracy dyplomowej w zakresie jej publicznego udostępniania i rozpowszechniania w wersji drukowanej i elektronicznej.

Warszawa, dn. ……..

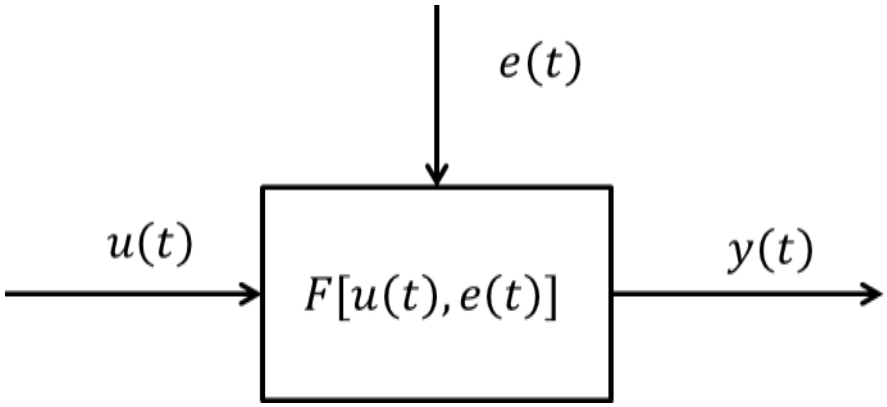
Aleksander Cudny

Spis treści

1. Wstęp teoretyczny

1.1 Modelowanie

Modelowanie to poszukiwanie matematycznego opisu danego systemu, który wiąże sygnały wejściowe z otrzymywanym sygnałem wyjściowym [1]. Opis ten może dotyczyć dziedziny czasu lub częstotliwości oraz reagować na szum, który dodawany jest do wyjściowego sygnału. Przykładowy model systemu tego typu przedstawiony został na rysunku 1.



Rysunek 1. Przykładowy model systemu, gdzie: $u(t)$ – sygnał wejściowy, $e(t)$ - szum, $y(t)$ – sygnał wyjściowy, $F[]$ – funkcja charakteryzująca zależność pomiędzy sygnałem wejściowym i wyjściowym.

Szukana zależność, w przypadku ogólnym, posiada charakter nieliniowy, a jedynie w szczególnych przypadkach istnieje możliwość jej linearyzacji przy przyjęciu odpowiednich założeń. Cechy systemu liniowego to:

- addytywność - odpowiedź systemu na wymuszenie sumą dwóch sygnałów jest sumą odpowiedzi na każde z poszczególnych wymuszeń,
- skalowanie – odpowiedź systemu na wymuszenie pomnożone przez stałą jest równe odpowiedzi systemu na to wymuszenie pomnożone przez tą samą stałą.

Można oba te wymagania zapisać za pomocą następującej równoważności:

$$F[a \cdot u(t) + b \cdot w(t)] = a \cdot F[u(t)] + b \cdot F[w(t)]$$

gdzie: u, w – dowolne funkcje wejściowe, a, b – stałe.

Dla systemów nieliniowych, powyższe zależności nie są spełnione, co utrudnia proces modelowania, ponieważ istnieje nieskończenie wiele funkcji nieliniowych. W przypadku sygnałów biomedycznych, badane systemy najczęściej pochodzą od systemów nieliniowych, co wymaga zastosowania metod modelowania nieliniowego. Przykładem takiej sytuacji może być zależność pomiędzy objętością wyrzutową serca i ciśnieniem tętniczym krwi, które są związane ze sobą trudną w identyfikacji zależnością nieliniową [2]. W niniejszej pracy skupiono się na modelowaniu sygnałów nieliniowych.

1.2 Przykłady znanych metod modelowania nieliniowego

1.2.1 Serie Volterry [3]

Jest to metoda charakteryzująca układ SISO (*ang. Single Input Single Output*), czyli system przyjmujący jeden sygnał wejściowy oraz zwracający jeden sygnał wyjściowy. Model ten zakłada, iż sygnał wyjściowy zależy od kombinacji przebiegu wejściowego oraz jego opóźnionej wersji:

$$\int_{-\infty}^{+\infty} \dots \int_{-\infty}^{+\infty} h_n(\tau_1, \dots, \tau_n) \prod_{i=1}^{n} u(t-\tau_i)\, d\tau_i, n \geq 1$$

gdzie: $h_n(\tau_1, \dots, \tau_n)$ – to funkcje parametryczne modelu, $u(t-\tau_i)$ – sygnał wejściowy opóźniony o stałą czasową τ_i.

Modelowanie tą metodą opiera się na doborze odpowiedniego rzędu modelu (n) oraz zestawu parametrów (h_n).

1.2.2 Szereg Wienera [4]

Model bazujący na szeregu Weinera jest rozszerzeniem modelu opartego o serie Volterry. Zakłada się, że funkcje parametryczne $h_n(\tau_1, \dots, \tau_n)$ są aproksymowane za pomocą skończonej liczby ortogonalnych względem siebie baz:

$$h_n(\tau_1, \dots, \tau_n) \approx \int_{-\infty}^{+\infty} \dots \int_{-\infty}^{+\infty} \alpha(i_1, \dots, i_n) \prod_{j=1}^{n} \psi_{i_j}(\tau_j)\, di_n, n \geq 1$$

gdzie: $\alpha(i_1, \dots, i_n)$ – współczynniki, $\psi_{i_j}(\tau_j)$ – bazy ortogonalne.

Bazy ortogonalne mogą być reprezentowane na przykład poprzez wielomiany Chebysheva lub Legendrea. Pomimo, iż najczęściej nie są one związane bezpośrednio z modelowanym sygnałem, odpowiedni dobór parametrów pozwala na wiernie odtworzenie badanego sygnału.

1.3 Metoda NARMAX

Metoda NARMAX jest rodzajem uogólnienia podejść zaprezentowanych w rozdziale 1.2. NARMAX [5], jest akronimem od angielskiego: *nonlinear auto-regressive moving average model with exogenous inputs*, co oznacza nieliniowy (N) model autoregresyjny (AR) ze średnią ruchomą (MA) oraz zewnętrznym wejściem (X). Oznacza to, iż wartość modelowanego sygnału w danej chwili może zależeć od:

- wartości sygnału w przeszłości (człon AR),
- wartości szumu z przeszłości (człon MA),
- wartości innego sygnału z możliwym opóźnieniem (człon X),

oraz wszystkie powyższe zależności mogą być nieliniowe. Dodatkowo, przyjmuje się, że modelowany sygnał jest dyskretny. W danej chwili czasu k, sygnał wyjściowy systemu może zostać zatem przedstawiony za pomocą następującego równania:

$$y(k) = F[y(k-1), y(k-2), \dots, y(k-n_y), u(k-d), u(k-d-1), \dots, u(k-d-n_u) e(k-1), e(k-2), \dots, e(k-n_e)] + e(k) \quad (1)$$

gdzie: F[] – nieznana funkcja (nieliniowa),

y(k - x) – wartości próbek sygnału w chwili czasowej x,

u(k - x) – wartości próbek sygnału sterującego w chwili czasowej x,

e(k - x) – wartości szumu w chwili czasowej x,

e(k) – nowa wartość szumu,

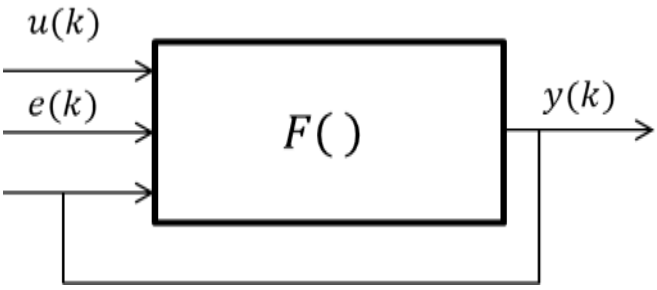
n_y – maksymalne opóźnienie członu AR,

n_e - maksymalne opóźnienie członu MA,

n_u - maksymalne opóźnienie członu X,

d, x∈{1, 2, …}.

Schemat tego typu układu został przedstawiony na rysunku 2.



Rysunek 2. Schemat modelu typu NARMAX.

Filozofia NARMAX opracowana pod kierownictwem S. A. Billingsa [6] opiera się na pięciu krokach, których celem jest znalezienie odpowiedzi na powiązane z nimi pytania:

1) Wykrycie struktury – „Jakie człony znajdują się w modelu?"
2) Estymacja parametrów – „Jakie są wartości współczynników modelu?"
3) Walidacja modelu – „Czy model jest poprawny oraz nieobciążony?"
4) Predykcja – „Jak wygląda modelowany sygnał w przyszłości?"
5) Analiza – „Jakie są własności dynamiczne systemu?"

Istnieje kilka metod modelowania według filozofii NARMAX np. FROLS, MFROLS, każda z nich opiera się na powyższych zasadach, pomimo różnic w metodologii. Naczelnym celem tego zespołu algorytmów jest znalezienie najprostszego modelu poprawnie opisującego strukturę danego zjawiska, w celu opisania głównych zasad rządzących systemem. Metoda tego typu, powinna więc rozważać wiele potencjalnych elementów modelu, z których wybrane zostanie ten, który jest najbardziej adekwatny. W sytuacji, gdy analizowany będzie sygnał pochodzący z systemu liniowego, modelowanie zgodnie z filozofią NARMAX powinno zwrócić jako wynik model liniowy [6].

Problemem w takiej sytuacji jest zaproponowanie możliwych typów nieliniowości, które zostaną wzięte pod uwagę w trakcie modelowania. Ponieważ równanie (1) nie nakłada ograniczeń na funkcję F[], liczba typów nieliniowości jest nieskończona. Na przykład:

- wielomian dowolnego stopnia,
- pierwiastek dowolnego stopnia,

- logarytm,
- sieć neuronowa,
- superpozycja powyższych metod,
- każda inna funkcja nieliniowa.

Z tego powodu, modelowanie nieliniowe zawsze niesie ze sobą możliwość nieświadomego popełniania błędu grubego. Ponieważ, jeśli badany system posiada nieliniowość, która nie została założona, wynik modelowania nie będzie mógł być poprawny. W metodach typu NARMAX, jedynym ograniczeniem są założenia przyjęte przez modelującego. Zastosowanie tej filozofii zapewnia ocenę wszystkich założonych *a priori* nieliniowości jako potencjalnych elementów modelu [6].

1.4 OLS

W celu zaimplementowania metod typu NARMAX, wykorzystano estymator OLS (*ang. Orthogonal Least Squares*), czyli estymator ortogonalnej sumy najmniejszych kwadratów [5]. Główną ideą estymatora OLS jest wprowadzenie pomocniczego modelu, którego elementy są ortogonalne względem sygnału, który jest modelowany, podobnie jak w Szeregu Weinera. Wtedy, kolejne człony modelu ortogonalnego mogą być kolejno wyznaczane, a następnie na ich podstawie mogą zostać obliczone parametry modelu szukanego. Iteracyjne powtarzanie kroków tej metody pozwala nie tylko na znalezienie nieobciążonych estymatorów modelu, ale także pokazania, jaki wkład w końcowy wynik modelowania miał każdy z nich.

Rozważania należy rozpocząć od założenia ogólnego modelu:

$$y(k) = \sum_{i=1}^{M} \theta_i p_i(k) + e(k) \tag{2}$$

gdzie: $y(k)$ – modelowana odpowiedź systemu dla $k = 1,2,\dots,N$; θ_i – parametry modelu ($\theta_i \in R$) znajdujące się przy regresorach (p_i) tworzących model; $p_i(k)$ – regresor, czyli składowy wektor tworzący model w chwili k; $e(k)$ – zewnętrzny szum lub błąd w chwili k; $i = 1,2,\dots,M$. Regresory p_i są określane jako kombinacja opóźnionych wartości sygnału lub opóźnionych sygnałów zewnętrznych, na przykład: $y(k-1), y(k-2), \dots, u(k-1), u(k-2), \dots$ i tak dalej. W przypadku ogólnym, funkcja opóźnionych członów modelu może przyjmować dowolną postać nieliniową. Dla modelu zakłada się także, że każdy regresor p_i jest niezależny od parametrów modelu θ_i, wobec czego [6]:

$$\frac{\partial p_i}{\partial \theta_j} = 0, \text{ dla } i = 1,2,\dots,M, \text{ oraz } j = 1,2,\dots,M \tag{3}$$

Celem estymatora jest przekształcenie modelu określonego w równaniu (2) do modelu pomocniczego, którego elementy są ortogonalne względem siebie. Model tego typu posiada postać:

$$y(k) = \sum_{i=1}^{M} g_i q_i(k) + e(k) \tag{4}$$

gdzie: g_i - parametry modelu ortogonalnego; $q_i(k)(\ i = 1,2, \dots, M)$ – ortogonalne składowe modelu określającego sygnał o N próbkach. Warunek Ortogonalności przedstawiona jest wtedy jako:

$$\sum_{k=1}^{N} q_i(k)q_j(k) = \begin{cases} d_i = \sum_{k=1}^{N} q_i^2(k) \neq 0, & i = j \\ 0, & i \neq j \end{cases} \tag{5}$$

Procedura ortogonalizacyjna dla modelu z równania (1) może zostać podsumowana jako:

$$\begin{cases} q_1(k) = p_1(k) \\ q_2(k) = p_2(k) - a_{1,2}q_1(k) \\ q_3(k) = p_3(k) - a_{1,3}q_1(k) - a_{2,3}q_2(k) \\ \vdots \qquad\qquad \vdots \qquad\qquad\qquad \vdots \\ q_m(k) = p_m(k) - \sum_{r=1}^{m-1} a_{r,m}q_r(k)\,, m = 2,3, \dots, M \end{cases} \tag{6}$$

gdzie parametr $a_{r,m}$ wiążący składowe modelu wyjściowego z równania (2) z modelem zortogonalizowanym ma postać:

$$a_{r,m} = \frac{\sum_{k=1}^{N} p_m(k)q_r(k)}{\sum_{k=1}^{N} {q_r}^2(k)}, \quad 1 \leq r \leq m-1 \tag{7}$$

na tej podstawie:

$$g_i = \frac{\sum_{k=1}^{N} y(k)q_i(k)}{\sum_{k=1}^{N} {q_i}^2(k)}, \quad i = 1,2, \dots, M \tag{8}$$

wobec tego:

$$\begin{cases} \theta_M = g_M \\ \theta_{M-1} = g_{M-1} - a_{M-1,M}\theta_M \\ \theta_{M-2} = g_{M-2} - a_{M-2,M-1}\theta_{M-1} - a_{M-2,M}\theta_M \\ \vdots \qquad\qquad \vdots \qquad\qquad\qquad \vdots \\ \theta_m = g_m - \sum_{j=m+1}^{M} a_{m,j}\theta_j\,, m = M-1, M-2, \dots, 1 \end{cases} \tag{9}$$

Pozwala to na podstawie założonych wcześniej regresorów opracować model złożony z ortogonalnych składowych, oraz z modelu tego typu odtworzyć model podstawowy.

1.5 ERR

Problemem w realizacji estymacji za pomocą OLS jest kryterium wyboru kolejnych regresorów do tworzonego modelu ortogonalnego. Pamiętając, że w modelu obecny jest także szum $\boldsymbol{e}$, energia (tudzież wariancja) systemu może zostać przedstawiona jako [7]:

$$\frac{1}{N}\boldsymbol{y}^T\boldsymbol{y} = \frac{1}{N}\sum_{i=1}^{M} g_i^2\boldsymbol{q}_i^T\boldsymbol{q}_i + \frac{1}{N}\boldsymbol{e}^T\boldsymbol{e} \tag{10}$$

gdzie: $\boldsymbol{y}$ – wektor modelowanego sygnału; g_i – współczynniki stojące przy elementach modelu zortogonalizowanego; $\boldsymbol{w}_i$ – wektor próbek elementy modelu zortogonalizowanego; $\boldsymbol{e}$ – wektor próbek szumu; N – liczba próbek sygnału.

W wyniku procedur ortogonalizacyjnych takich jak metoda Garm'a-Schidt'a czy metoda Householder [7], zostało określone, że:

$$1 = \frac{\sum_{i=1}^{M} g_i^2 \boldsymbol{q}_i^T \boldsymbol{q}_i}{\boldsymbol{y}^T \boldsymbol{y}} + \frac{\boldsymbol{e}^T \boldsymbol{e}}{\boldsymbol{y}^T \boldsymbol{y}} = \sum_{i=1}^{M} ERR_i + ESR \tag{10}$$

gdzie: ERR_i – stopień redukcji błędu (*ang. Error Reduction Ratio*) danego elementu modelu; ESR – stosunek szumu do sygnału (*ang. Error to Signal Ratio*).

W takim wypadku, *ERR* może być stosowane jako kryterium oceny poprawności wybranego regresora jako elementu modelu, jak i poprawności całego modelu. Suma kolejnych wartości *ERR* może oznaczać koniec modelowania gdy $\sum_{i=1}^{M} ERR_i \to 1$. Porównanie poszczególnych wartości *ERR* dla różnych elementów modelu pokazuje, który z nich najbardziej wpłynął na sygnał wyznaczonego modelu.

Przykładem obrazującym potencjał estymatora OLS oraz wskaźnika *ERR* jest sygnał opisany przez S. A. Billingsa [6]. Założono model który jest kombinacją liniową trzech sygnałów oraz zewnętrznego szumu. W celu utrudnienia procesu modelowania, przyjęto, że szukany model zależny jest od czterech sygnałów, gdzie dodatkowy czwarty sygnał został wyznaczony jako kombinacja liniowa dwóch sygnałów składowych tego modelu oraz innego szumu. Pożądaną cechą danego estymatora jest wskazanie poprawnych składowe modelu oraz odrzucenie sygnału który nie ma na niego wpływu. Określony liniowy model ma postać:

$$y(k) = x_1(k) + x_2(k) + x_3(k) + e(k) \tag{11}$$

gdzie: x – pewne sygnały; $e(k)$ - szum.

Wprowadzono dodatkowy sygnał x_4, który jest liniową kombinacją dwóch innych sygnałów tworzących model oraz szumu $\eta(k)$:

$$x_4(k) = 0{,}25x_1(k) + 0{,}75x_2(k) + \eta(k) \tag{12}$$

Wobec tego, szukany model jest postaci:

$$y(k) = \theta_1 x_1(k) + \theta_2 x_2(k) + \theta_3 x_3(k) + \theta_4 x_4(k) + e(k) \tag{13}$$

Wartości sygnałów zebrano w tabeli 1. Rozpoczęto od wyznaczenia parametrów modelu metodą regresji liniowej najmniejszych kwadratów. W wyniku zastosowania tej metody liniowej, otrzymano wektor parametrów postaci: $\boldsymbol{\theta} = [0{,}8569; 0{,}5363; 0{,}9873; 0{,}5977]$. Nie jest to poprawny zestaw parametrów dla modelu, a otrzymane różnice są znaczne. Dodatkowo, nierozwiązany został problem zmiennej x_4, która jedynie pozornie ma wpływ na sygnał.

Tabela 1. Wartości sygnałów dla przykładu z rozdziału 1.5.

Numer próbki (k)	x_1	x_2	x_3	x_4	y
1	9	-5	5	-1,53	9,08
2	1	-1	8	-0,39	7,87
3	2	-5	6	-3,26	3,01
4	8	-2	0	0,36	5,89
5	0	0	9	0,13	9,05

Następnie, zastosowano estymator OLS kolejno dla modelu dwu-, trzy- oraz czteroparametrowego, wyniki przedstawiono w tabeli 2. Analizując wartość *ESR*, stwierdzono że

model dwuparametrowy nie jest modelem pełnym, gdyż $ESR = 5{,}36\%$. Model trzyparametrowy, nie dość, że cechuje się niską wartością *ESR* równą 0,0086%, dodatkowo posiada wartości θ bardzo zbliżone do parametrów modelu początkowego (oscylują one wokół jedności). Natomiast model czteroparametrowy różni się wartością *ESR* niewiele od i tak niskiego ESR dla modelu poprzedniego. Można także wskazać, że wartość *ERR* dla parametru x_4 jest o cztery rzędy wielkości niższa niż *ERR* dla pozostałych zmiennych. Wskazuje to na jej znikomy wkład w ostateczną wartość sygnału oraz to że model trzyparametrowy jest modelem wystarczającym.

Tabela 2. Wyniki modelowania metodą NARMAX dla przykładu z rozdziału 1.5.

Typ modelu	Wybrane zmienne	θ	ERR	$ESR = 1 - \sum ERR$
Dwuparametrowy	x_3	0,8194	0,7737	0,0536
	x_1	0,6013	0,1727	
Trzyparametrowy	x_3	0,9967	0,7737	0,000086
	x_1	1,0004	0,1727	
	x_2	0,9917	0,0535	
Czteroparametrowy	x_3	0,9873	0,7737	0,000081
	x_1	0,8569	0,1727	
	x_2	0,5363	0,0535	
	x_4	0,5977	0,000005	

1.6 FROLS

Wykorzystując estymator OLS oraz współczynnik *ERR*, opisano algorytm modelowania nieliniowego FROLS (*ang. Forward Regression with Orthogonal Least Squares*) [5], czyli algorytm regresji w przód wykorzystujący ortogonalną sumę najmniejszych kwadratów, czyli estymator opisany w rozdziale 1.4. Założono, iż wybór kolejnych regresorów dla estymatora OLS powinien być uwarunkowany najwyższą wartością *ERR* dla danego regresora. Pozwala to na wybór takiego członu modelu, który w najwyższym stopniu zmniejsza błąd modelowania oraz na zaprzestanie modelowania, gdy *ESR* będzie posiadało zadowalającą wartość. W kolejnych podrozdziałach zostaną przedstawione kroki w modelowaniu według algorytmu FROLS. Przebieg modelowania jest zgodny z filozofią NARMAX przedstawioną w rozdziale 1.3.

1.6.1 Krok 1. – Zbieranie danych

W celu wykonania modelowania wymagana jest akwizycja przebiegów sygnału badanego $y(k)$ oraz zewnętrznych sygnałów wejściowych $x(k)$, które mają wpływ na badany system. Sygnał powinien być zbierany jako dyskretny lub do takiej formy zostać przekształcony. Dodatkowo, rejestrowany przebieg powinien posiadać jak najmniej szumów zewnętrznych oraz nie być poddawany filtracji, która może zaburzyć proces identyfikacji. Według autora metody, jeśli szum obecny w sygnale jest szumem białym o zerowej średniej, to nie wpływa on na wartość *ERR* [6] oraz proces identyfikacji.

1.6.2 Krok 2. – Określenie ram modelowania

Ponieważ liczba możliwych nieliniowości jest nieskończona, wymagane jest określenie *a priori* obszaru w jakim dokonywane są poszukiwania. Wymagania te można sformułować jako następujące pytania:

- Jakie jest maksymalne opóźnienie członu AR (n_y)?
- Jakie jest maksymalne opóźnienie sygnałów zewnętrznych, czyli (n_u)?
- Jakie nieliniowości są przewidywane oraz jaki jest ich maksymalny stopień (l)?

Po podaniu odpowiedzi na te pytania, znane są możliwe człony modelu: $y(k-1)$, $y(k-2)$, …, $y(k-n_y)$, $u_i(k-1)$,$u_i(k-2)$, …, $u_i(k-n_{x_i})$, gdzie: y – badany sygnał; u_i – jeden z zewnętrznych sygnałów wejściowych; n_y – określone maksymalne opóźnienie członu AR; n_{x_i} – maksymalne opóźnienie jednego z zewnętrznych sygnałów wejściowych. Znana jest także nieliniowość, jaka wiąże te człony, co pozwala na realizację następnego kroku. W przypadku gdy nieliniowością jest wielomian, maksymalny stopień nieliniowości określa maksymalną potęgę występującą w tym wielomianie.

1.6.3 Krok 3. – Wyznaczenie wektora regresorów

Znając ramy modelowania określone w poprzednim kroku, należy wyznaczyć wektor zawierający wszystkie możliwe regresory dla danego sygnału. Na przykład, szukany model posiada wyjście y oraz jedno wejście u, określono że $n_y = n_x = 2$, oraz że nieliniowość to wielomian, który jest maksymalnie drugiego stopnia, czyli $l = 2$. Wtedy, wektor regresorów będzie składał się z kombinacji sygnałów: $y(k-1), y(k-2), u(k-1), u(k-2)$, zgodnie z maksymalnym stopniem wielomianu. Uwzględniając dodatkowo składową stałą oznaczoną jako $const.$, wektor regresorów oznaczony jako D, będzie określony jako:

$$\begin{gathered} D = \{const., y(k-1), y(k-2), u(k-1), u(k-2)\} \cup \\ \{y^2(k-1), y(k-1)y(k-2), y(k-1)u(k-1), y(k-1)u(k-2)\} \cup \\ \{y^2(k-2), y(k-2)u(k-1), y(k-2)u(k-2)\} \cup \\ \{u^2(k-1), u(k-1)u(k-2)\} \cup \\ \{u^2(k-2)\} \end{gathered} \tag{14}$$

Jako inny przykład może posłuż sytuacja, gdy model posiada jedno wejście y oraz jedno wyjście u, $n_y = n_x = 1$, oraz maksymalny stopień wielomianu $l = 3$. Wtedy, wektor D ma postać:

$$\begin{gathered} D = \{const., y(k-1), u(k-1)\} \cup \\ \{y^2(k-1), y(k-1)u(k-1), u^2(k-1)\} \cup \\ \{y^3(k-1), y^2(k-1)u(k-1), u^2(k-1)y(k-1), u^3(k-1)\} \end{gathered} \tag{15}$$

Skomponowanie wektora tego typu stwarza więc problem obliczeniowy, który wraz ze zwiększaniem ram modelowania rośnie wykładniczo. Liczba regresorów jest określona zależnością:

$$M = \binom{n+l}{l} = \frac{(n+l)(n+l-1)\cdots(n+1)}{l!}$$

gdzie: $n = n_y + n_{x_1} + \cdots + n_{x_i}$, l – maksymalny stopień wielomianu branego pod uwagę. Przykładowo, zwiększenie ram do $l = 3$ oraz $n = 5 + 5 = 10$ oznacza wzrost potencjalnej liczby regresorów do $M = 286$. Stanowi to wyzwanie dla mocy obliczeniowej komputera, na którym modelowanie jest wykonywane oraz potencjalnie zwiększa znacząco czas obliczeń dla bardziej złożonych modeli.

1.6.4 Krok 4. – Wybór pierwszego elementu

Gdy wektor regresorów został określony, znana jest baza możliwych członów tworzonego modelu. Wybór pierwszego z nich wymaga wyznaczenia wartości ERR dla każdego elementu wektora $D = \{\boldsymbol{p}_1, \boldsymbol{p}_2, \dots, \boldsymbol{p}_m\}$, dla $m = 1,2, \dots, M$, gdzie M to liczba potencjalnych regresorów. Określając $\sigma = \boldsymbol{y}^T\boldsymbol{y}$, szukany regresor o indeksie l_1, jako:

$$g_m = \frac{\boldsymbol{y}^T\boldsymbol{p}_m}{\boldsymbol{p}_m^T\boldsymbol{p}_m}$$

$$ERR[m] = g_m^2(\boldsymbol{p}_m^T\boldsymbol{p}_m)/\sigma$$

$$l_1 = \arg\max_{1 \le m \le M}\{ERR[m]\}$$

Znaleziony indeks l_1 odpowiadający regresorowi o najwyższym ERR, jest wtedy przyjmowany jako pierwszy element modelu ortogonalnego. Jego parametr g oraz odpowiadająca mu wartość ERR są zapisywane. Można to zapisać jako:

$$\boldsymbol{q}_1 = \boldsymbol{p}_{l_1}\ ;\ g_1 = g_{l_1}\ ;\ \ \boldsymbol{err}[1] = ERR[l_1]\ \ ;\ \ D_1 = D - \boldsymbol{p}_{l_1}\ ($$

gdzie: $\boldsymbol{err}$ – wektor w którym przechowywane są wartości ERR dla kolejno wybranych regresorów.

Po zapisaniu znalezionego regresora oraz związanych z nim wartości, poszukiwanie rozpoczynane jest od nowa w celu znalezienia kolejnego członu modelu ortogonalnego.

1.6.5 Krok 5. – Wybór następnych elementów modelu

Następne kroki wykonywane są analogicznie do kroku pierwszego, z tą różnicą, że model ortogonalny posiada już pewną liczbę członów, zależną od liczby wykonanych wcześniej kroków. Obliczanie wartości ERR dla potencjalnych nowych członów modelu trzeba wykonać na nowo, w odniesieniu do aktualnej postaci modelu. Szukany jest wobec tego indeks regresora (oznaczony jako l_s) o najwyższym ERR dla aktualnej postaci modelu, gdzie s jest bieżącym numerem kroku. Wektor regresorów jest oznaczony jako D_{s-1} i nie zawiera członów wybranych w krokach poprzednich. Wtedy, dla $m = 1,2, \dots, M - (s-1)$ oraz $m \neq l_1 \neq l_2 \neq \cdots \neq l_{s-1}$:

$$\boldsymbol{q}_m = \boldsymbol{p}_m - \sum_{r=1}^{s-1}\frac{\boldsymbol{p}_m^T\boldsymbol{q}_r}{\boldsymbol{q}_r^T\boldsymbol{q}_r}\boldsymbol{q}_r,\quad \boldsymbol{p}_m \in D_{s-1}$$

$$g_m = \frac{\boldsymbol{y}^T\boldsymbol{q}_m}{\boldsymbol{q}_m^T\boldsymbol{q}_m}$$

$$ERR[m] = g_m^2(\boldsymbol{q}_m^T\boldsymbol{q}_m)/\sigma$$

$$l_s = \arg\max_{1 \le m \le M-(s+1)}\{ERR[m]\}$$

oraz po znalezieniu l_s które odpowiada regresorowi o najwyższym *ERR*, model ortogonalny jest rozszerzany o kolejny człon $\boldsymbol{q}_s$ wraz z odpowiadającym mu parametrem g_s. Wartość *ERR* dla wybranego elementu jest zapisywana, a z wektora *D* usuwany jest wybrany regresor. Dodatkowo, wyznaczane są wartości parametrów $a_{r,s}$, które wiążą model wyjściowy ze wzoru (2) z modelem ortogonalnym ze wzoru (4). Zapisać te działania można jako:

$$\boldsymbol{q}_s = \boldsymbol{q}_{l_s}\ ;\ \ g_s = g_{l_s}\ ;\ \ \boldsymbol{err}[s] = ERR[l_s]\ \ ;\ \ D_s = D_{s-1} - \boldsymbol{p}_{l_s}\ ;\ \ a_{r,s} = \frac{\boldsymbol{q}_r^T \boldsymbol{p}_{l_s}}{\boldsymbol{q}_r^T \boldsymbol{q}_r}, r = 1,2,..,s-1$$

Krok ten jest powtarzany aż wartość *ESR* będzie zadowalająca:

$$ESR = 1 - \sum_{s=1}^{M_0} \boldsymbol{err}[s]$$

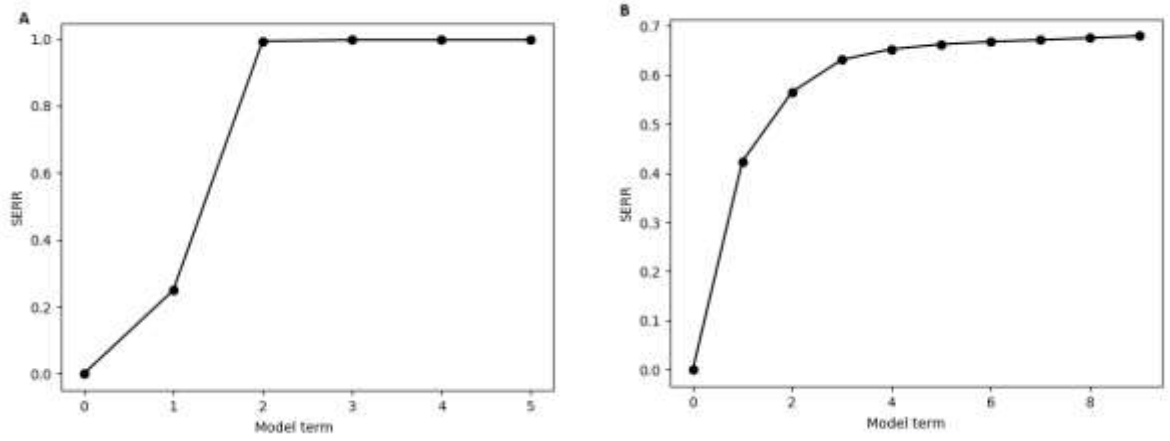
gdzie: M_0 jest liczbą wybranych regresorów. Konieczne wprowadzenie jest pewnej granicy wartości *ESR* która pozwala zadecydować o końcu modelowania.

1.7 Analiza procesu modelowania

Istotną decyzją podczas wykonywania modelowania metodą typu NARMAX jest moment zaprzestania dodawania kolejnych regresorów do modelu. Właściwy moment na jej podjęcie nie posiada ścisłej definicji, a raczej jest określony przez zbiór czynników, których superpozycja jest przesłanką do jej podjęcia. W przypadku metody NARMAX, jednym z podstawowych parametrów jest suma *ERR* dla modelu, nazywana *SERR*.

$$SERR = \sum ERR$$

Dwoma głównymi parametrami, na podstawie których oceniany jest aktualny model, są wartość *SERR* oraz jej przebieg podczas modelowania. Zakłada się, że dobrze dopasowany model posiada *SERR* wyższe niż 0,8 [6]. Jednakże, dużo ważniejszym sygnałem do zaprzestania modelowania jest przebieg SERR, taki jak na rysunku 3.



Rysunek 3. Przykładowe przebiegi SERR. A – sytuacja oczywista, B – sytuacja nieoczywista.

Sygnałem do zaprzestania dodawania kolejnych regresorów jest spłaszczenie charakterystyki *SERR*. Moment rozpoczęcia wygładzenia się wykresu powinien zostać uznany za sygnał, iż aktualny model jest wystarczający. W sytuacji pierwszej (A) rysunku 3. widoczne jest gwałtowne spłaszczenie charakterystyki *SERR* po wyborze elementów 3. i 4. Wskazuje to, że

poprawny model powinien posiadać jedynie człony 1. i 2. W przypadku wykresu drugiego (B) na rysunku 3., punkt ten jest trudniejszy do określenia i w związku z tym poprawny model może być zarówno cztero- jak i pięcioelementowy. Wtedy wymagane jest sprawdzenie poprawności potencjalnych modeli i wybrania tego, który najpomyślniej przejdzie walidację. W sytuacji, gdy *SERR* jest niskie, ale ustabilizowane, należy posiłkować się metodami walidacji modelu oraz zastanowić się, czy inny czynnik nie wzięty pod uwagę nie ma wpływu na sygnał.

1.8 Wyznaczenie ostatecznego modelu

Gdy wybrano M_0 regresorów, konieczne jest wyznaczenie parametrów które znajdują się przy nich. Zakładając, że ze względu na niską wartość ESR, model ogólny oraz zortogonalizowany są sobie równe:

$$y(k) = \sum_{i=1}^{M} \theta_i p_i(k) + e(k) = \sum_{i=1}^{M} g_i q_i(k) + e(k)$$

W powyższym równaniu, jedynymi niewiadomymi są parametry θ_i, ponieważ regresory $p_i(k)$ zostały wybrane w trakcie modelowania, a elementy modelu ortogonalnego $q_i(k)$ wraz z ich parametrami g_i były wyznaczane na bieżąco podczas tego procesu. Wobec tego, konieczne jest wykonanie konwersji z modelu ortogonalnego na model wyjściowy poprzez wykonanie procesu odwrotnego do wykonywanej uprzednio ortogonalizacji. Znając wartości składników $a_{r,s}$ dla $1 \leq r < s \leq M_0$, można określić macierz:

$$A = \begin{pmatrix} 1 & a_{12} & a_{13} & \cdots & a_{1M_0} \\ 0 & 1 & a_{21} & \cdots & a_{2M_0} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & a_{M_0-1,M_0} \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix}$$

Która reprezentuje działanie opisane w równaniu (9), które można zawrzeć w równaniu:

$$A\boldsymbol{\theta} = \boldsymbol{g}$$

gdzie: $\boldsymbol{g} = [g_1, g_2, \dots, g_g]$, $\boldsymbol{\theta} = [\theta_1, \theta_2, \dots, \theta_{M_0}]$. Rozwiązanie tego równania pozwala na otrzymanie parametrów modelu, wobec czego wyznaczenie pełnego modelu wyjściowego określonego w równaniu (2).

1.9 Metody walidacji modelu

Po wyznaczeniu zadowalającego modelu, wymagana jest weryfikacja, czy określony został poprawny model. W tym celu, poza ESR, stosowane są dwa mechanizmy:

- symulacja,
- autokorelacja różnic modelu i sygnału.

Symulacja wymaga podzielenia danych na dwie części, z których do jednej z nich dopasowany jest model, natomiast druga służy jako wzór poprawnego przebiegu sygnału. Po wykonaniu modelowania, wyznaczane są przyszłe wartości próbek sygnału generowanego przez model. Wygenerowany sygnał porównywany jest następnie z zachowaną częścią sygnału. Zgodność obu przebiegów pozwala na ocenę poprawności modelu, tak jak na rysunku 4.

Autokorelacja dwóch sygnałów, na przykład *x* oraz *y*, została określona wzorem [6]:

$$\phi_{xy}(\tau) = \frac{\sum_{k=1}^{N-\tau}[x(k) - \bar{x}][y(k+\tau) - \bar{y}]}{\sqrt{\sum_{k=1}^{N}[x(k) - \bar{x}]^2}\sqrt{\sum_{k=1}^{N}[y(k) - \bar{y}]^2}} \tag{10}$$

Autokorelacja różnic pozwala określić, czy w różnicach pomiędzy modelem a badanym sygnałem znajdują się istotne podobieństwa. Przyjęto [6], że z 95% dokładnością, korelacja wzajemna dwóch sygnałów nie posiada wartości istotnej statystycznie jeśli nie przekracza $\pm\frac{1{,}96}{\sqrt{N}}$, gdzie *N* to liczba próbek korelowanych sygnałów. Ograniczenie to nie dotyczy oczywiście opóźnienia równego zero, dla którego autokorelacja będzie posiadać najwyższą wartość. Przykładowe wykresy autokorelacji różnic dla poprawnego i błędnego modelu zostały przedstawione na rysunku 4.

Podane metody zostały zobrazowane przykładami błędnego oraz poprawnego modelowania. Wykresy z rysunku 4. zostały wykonane dla funkcji logistycznej postaci:

$$y = ry(k-1) - ry^2(k-1)$$

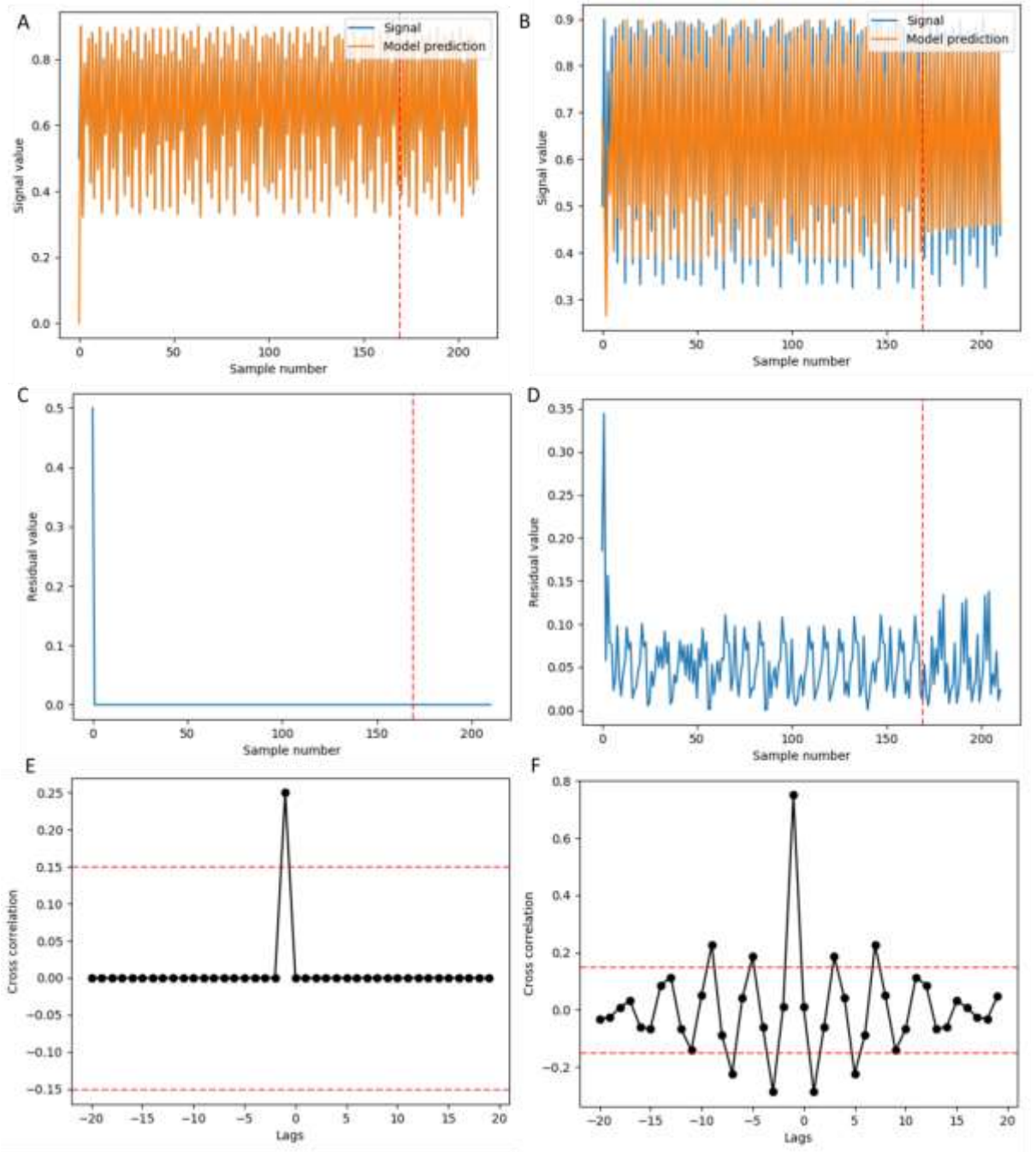
gdzie: r – stała. W wyniku modelowania otrzymano model poprawny:

$$y = 3{,}6012y(k-1) - 3{,}6012y^2(k-1)\ ;\quad ESR = 0{,}0032$$

oraz model błędny postaci:

$$y = 0{,}6855 + 0{,}3199y(k-4) - 0{,}5182y^2(k-1)\ ;\ ESR = 0{,}0095$$

W przypadku obu powyższych modeli, wartość ESR może zostać uznana za dostatecznie niską – błąd został zredukowany więcej niż 99%. Jednak wyniki symulacji, oraz autokorelacji różnic pokazują znaczne rozbieżności w modelach i jednoznacznie potwierdzają poprawność modelu pierwszego.



Rysunek 4. Wykresy weryfikacji modelowania dla poprawnego i błędnego modelu funkcji logistycznej. Przedstawiono wykres symulacji (A, B) kolejno modelu poprawnego oraz błędnego. Na wykresach środkowych (C, D) przedstawiono różnice modelowania kolejno modelu poprawnego oraz błędnego. Na dolnych wykresach (E, F) przedstawiono wykres autokorelacji różnic kolejno z wykresów C oraz D. Na wykresach A, B, C i D pionową linią przerywaną zaznaczono rozpoczęcie danych otrzymanych w wyniku symulacji. Na wykresach E i F liniami poziomymi zaznaczono zakres wartości $\pm\frac{1,96}{\sqrt{N}}$, gdzie *N* to długość danych z wykresów A, B, C i D, które wyznaczają obszar, wewnątrz którego wartości autokorelacji nie są istotne statystycznie z prawdopodobieństwem 95%.

1.10 Modyfikacja algorytmu FROLS

W trakcie implementacji algorytmu FROLS autor zauważył, iż istnieją sytuacje, kiedy kryterium najwyższego ERR nie prowadzi do wskazania poprawnego modelu. W przypadku, gdy system złożony jest z członów równoważnych, istnieje możliwość, że jeden z potencjalnych regresorów który nie jest składową modelu posiada najwyższą wartość ERR. Prowadzi to do wyboru błędnego członu w jednym z pierwszych kroków i błędny element jest dodawany do tworzonego modelu ortogonalnego. Powoduje to niepowodzenie całego procesu modelowania, ponieważ każdy element wybrany po błędnym wskazaniu z założenia będzie niepoprawny. Przykładem takiej sytuacji jest modelowanie przebiegu funkcji logistycznej danej wzorem:

$$y(k) = ry(k-1) - ry^2(k-1)$$

gdzie: r – stała, której pewne wartości warunkują generowanie przebiegów chaotycznych. System ten posiada dwa równoważne komponenty. Wyznaczając wartości ERR dla różnych regresorów autor otrzymał następujące wyniki w pierwszym kroku:

$$p = y(k-1) \Leftrightarrow ERR = 0{,}4299$$
$$p = y^2(k-1) \Leftrightarrow ERR = 0{,}5676$$
$$p = y(k-4) \Leftrightarrow ERR = 0{,}88$$

gdzie: p – potencjalny regresor.

W takim wypadku, wybrany zostanie regresor błędny, pomimo iż poprawnie została wyznaczona wartość ERR. Wybór niepoprawnego regresora prowadzi do uznania go jako kolejny człon modelu ortogonalnego i tym samym uniemożliwia otrzymanie poprawnego wyniku modelowania.

Autorzy metody radzą sobie z tego typu przypadkiem rozpatrując kilka przebiegów pochodzących z danego systemu i poszukując modelu który potrafi poprawnie odwzorować je wszystkie [6]. Natomiast w przypadku sygnałów biomedycznych nie zawsze jest to możliwe, a zwykle jest sytuacją niewskazaną. Wymaga to bowiem przeprowadzenia dwóch lub więcej badań, lub jednego dłuższego, co wydłuża czas pobytu pacjenta w placówce. Zmniejsza to więc drastycznie liczbę potencjalnych pacjentów którzy mogą zostać dzięki temu zdiagnozowani. Korzystne zatem jest skrócenie czasu badania do niezbędnego minimum.

Zaproponowano autorską modyfikację algorytmu FROLS do postaci dwukrokowej, czerpiącej z idei sortowania bąbelkowego. Każdy analizowany regresor traktowany jest jako potencjalna baza do wyboru następnego regresora. Wybór nie zostaje podjęty na podstawie wartości ERR, jaka jest przypisana do danego regresora, ale na podstawie sumy ERR danego regresora oraz tego, który zostałby wybrany jako następny, gdyby aktualny regresor uznać za najlepszy. Metoda ta potencjalnie zwiększa czas obliczeń, pozwala jednak na usprawnienie procesu modelowania, co zostanie przedstawione w następnych rozdziałach.

Zakładając, że aktualny krok modelowania wynosi s, szukany jest indeks l_s, a wektorem regresorów jest D_{s-1}, czyli wektor bez członów wybranych w krokach poprzednich. Wtedy, dla $m = 1,2,\dots,M-(s-1)$ oraz $m \neq l_1 \neq l_2 \neq \cdots \neq l_{s-1}$, wprowadzono nowy iterator $j = 1,2,\dots,M-s$, gdzie $j \neq l_1 \neq l_2 \neq \cdots \neq l_{s-1} \neq m$. Wtedy:

$$\boldsymbol{q}_m = \boldsymbol{p}_m - \sum_{r=1}^{s-1} \frac{\boldsymbol{p}_m^T \boldsymbol{q}_r}{\boldsymbol{q}_r^T \boldsymbol{q}_r} \boldsymbol{q}_r, \quad \boldsymbol{p}_m \in D_{s-1}$$

$$g_m = \frac{\boldsymbol{y}^T \boldsymbol{q}_m}{\boldsymbol{q}_m^T \boldsymbol{q}_m}$$

$$ERR[m] = g_m^2 (\boldsymbol{q}_m^T \boldsymbol{q}_m)/\sigma$$

następnie:

$$\boldsymbol{q}_j = \boldsymbol{p}_j - \sum_{r=1}^{s-1} \frac{\boldsymbol{p}_j^T \boldsymbol{q}_r}{\boldsymbol{q}_r^T \boldsymbol{q}_r} \boldsymbol{q}_r - \frac{\boldsymbol{p}_j^T \boldsymbol{q}_m}{\boldsymbol{q}_m^T \boldsymbol{q}_m} \boldsymbol{q}_m, \quad \boldsymbol{p}_m \in D_{s-1} - \boldsymbol{p}_m$$

$$g_j = \frac{\boldsymbol{y}^T \boldsymbol{q}_j}{\boldsymbol{q}_j^T \boldsymbol{q}_j}$$

$$ERR[j] = g_j^2 \left(\boldsymbol{q}_j^T \boldsymbol{q}_j\right)/\sigma$$

wybór następuje wtedy, gdy:

$$l_s = \arg\left(\max_{\substack{1 \le m \le M-(s+1) \\ 1 \le j \le M-s}} \{ERR[m] + ERR[j]\} \right)$$

Następnie krok jest finalizowany tak jak w rozdziale 1.6.5., a wartości związane z wybranym regresorem są zapisane w niezmieniony sposób.

2. Cel i zakres pracy

Celem niniejszej pracy jest implementacja oraz test oprogramowania umożliwiającego modelowanie szeregów czasowych, pochodzących od systemów o nieliniowej dynamice, metodą NARMAX [6] za pomocą algorytmu FROLS (*Forward Regression with Orthogonal Least Squares*) wykorzystując estymator OLS (*Orthogonal Least Squares*) oraz wskaźnik ERR (*Error Reduction Ratio*).

W pracy przedstawiono założenia teoretyczne modelowania metodą NARMAX, opis wykonanego oprogramowania, wyniki modelowania przykładowych sygnałów oraz wpływ ich parametrów na wyniki modelowania. Dodatkowo, w pracy zaproponowano modyfikację algorytmu, która może potencjalnie usprawnić zastosowany algorytm.

Jako przykłady, modelowane będą przebiegi pochodzące od:

- liniowego systemu stochastycznego,
- nieliniowego systemu stochastycznego,
- funkcji logistycznej.

Zbadany zostanie wpływ:

- szumu białego o różnej wartości amplitudy,
- długości rekordu danych,
- wyboru metody podstawowej lub zmodyfikowanej,

na wynik modelowania.

Opracowana metoda mogłaby w przyszłości posłużyć do modelowania sygnału pochodzącego na przykład ze stetoskopu elektronicznego lub innego sygnału biomedycznego. Badania tego typu miałyby na celu określenie, czy istnieje ogólny model danego typu sygnału charakteryzujący osoby zdrowe. Model taki mógłby stanowić podstawę do wypracowania metody diagnostycznej, opartej o zgodność zarejestrowanego przebiegu z modelem określającym sygnał pochodzący od osoby zdrowej. Jest to jedynie perspektywa rozwoju, która zostanie nakreślona na końcu niniejszej pracy.

3. Metodologia

3.1 Wymagane funkcjonalności oraz wybór technologii

Do realizacji założeń zawartych w rozdziale 2., wymagana jest implementacja oprogramowania, które realizuje następujące funkcjonalności:

- Wczytanie przebiegu modelowanego sygnału oraz wejściowych sygnałów zewnętrznych do pamięci programu.
- Określenie ram modelowania: maksymalnego opóźnienia sygnałów, maksymalnego stopnia szukanego wielomianu, określenia, jaka część danych zostanie zachowana do przeprowadzenia symulacji oraz wyboru pomiędzy metodą FROLS a jej zmodyfikowaną wersją.
- Rozpoczęcie oraz możliwość krokowego wykonywania algorytmu FROLS wraz z ciągłym monitorowaniem parametrów aktualnego modelu: wybranych członów, ich parametrów, wartości *ERR* dla każdego z członów oraz wartości sumy *ERR*, czyli wartości $1 - ESR$.
- Sporządzenie wykresów: różnic pomiędzy modelem a sygnałem, autokorelacji tych różnic, symulacji przyszłych wartości sygnału modelu razem z zachowaną uprzednio częścią sygnału oraz różnic pomiędzy symulacją modelu a badanym sygnałem.
- Zapis aktualnego modelu do pliku tekstowego.

Do realizacji wyżej wymienionych celów wykorzystano język *Python* 3.5 [8] wraz z rozszerzeniami:

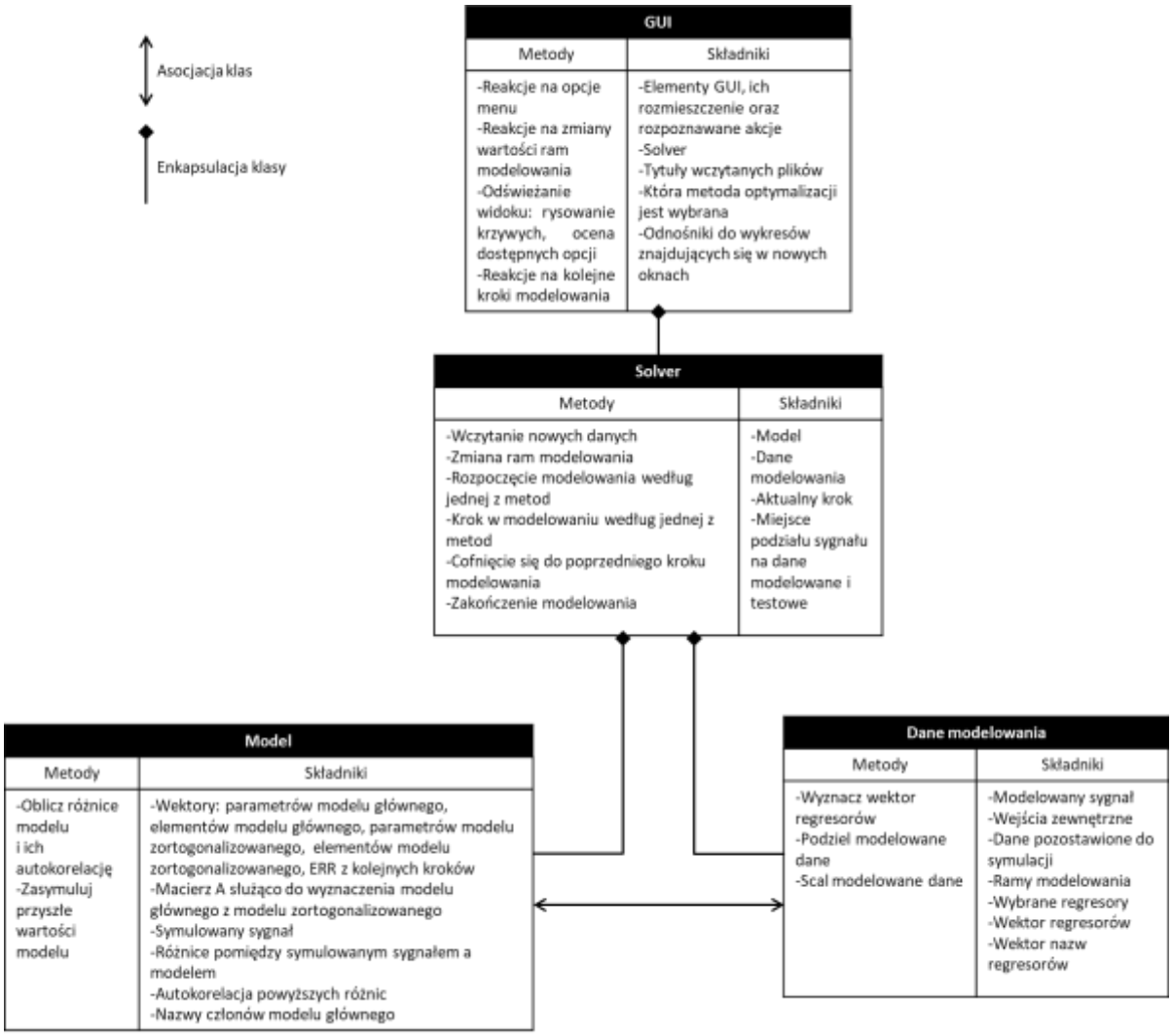
- *numpy* - zestaw funkcji do obliczeń numerycznych,
- *scipy* - zestaw wielu narzędzi naukowych, tutaj używane do wyznaczenia autokorelacji,
- *matplotliub* – biblioteka do sporządzania wykresów,
- *PyQt5* – biblioteka do tworzenia interfejsu graficznego.

Kod źródłowy został opracowany w środowisku *PyCharm 2017* [9], w którym także wygenerowane zostały przebiegi badanych funkcji. Zaimplementowane oprogramowanie jest dedykowane systemowi *Windows 10* pracującemu na procesorze o architekturze 64-bit.

3.2 Architektura

Zaproponowana została architektura składająca się z czterech głównych klas, której schemat został przedstawiony na rysunku 5. Tworzone oprogramowanie zostało podzielone na dwie odrębne części: interfejs graficzny (*GUI*) oraz rdzeń obliczeniowy (*Solver*). Pozwala to na uniezależnienie od siebie segmentu komunikacji z użytkownikiem oraz segmentu wykonywania obliczeń. Rdzeń programu posiada dzięki temu niezależność oraz możliwość niezależnego testowania. Niezależność tego typu wymaga, aby *Solver* był częścią składową *GUI*, użytkowaną według sygnałów pobranych od użytkownika. *Solver* posiada możliwość edycji modelowanych sygnałów oraz modyfikacji ram modelowania. Pozwala także na rozpoczęcie, krokowe wykonywanie i finalizację modelowania metodą NARMAX według algorytmu FROLS lub zmodyfikowanego algorytmu FROLS. Sam rdzeń zawiera w sobie dwie dodatkowe klasy: *Model* oraz *Dane modelowania*. W klasie *Model* znajdują się parametry tworzonego modelu jak i wyniki modelowania, o których mowa w rozdziale 1.6. Posiada ona także metody pozwalające na

wyznaczenie różnic pomiędzy sygnałem i modelem oraz na obliczenie autokorelacji tych. Klasa *Dane modelowania* zawiera natomiast elementy, które są wymagane do przeprowadzenia modelowania, jednak nie są bezpośrednimi składnikami modelu. Są to: modelowany sygnał, zewnętrzne sygnały wejściowe, ramy modelowania, wektor regresorów, wybrane regresory oraz związane z regresorami ich nazwy. Metody tej klasy pozwalają na wyznaczenie wektora regresorów na podstawie sygnałów wejściowych, podziału sygnałów wejściowych na dane modelowane oraz dane do symulacji, a także na scalenie uprzednio podzielonych danych.



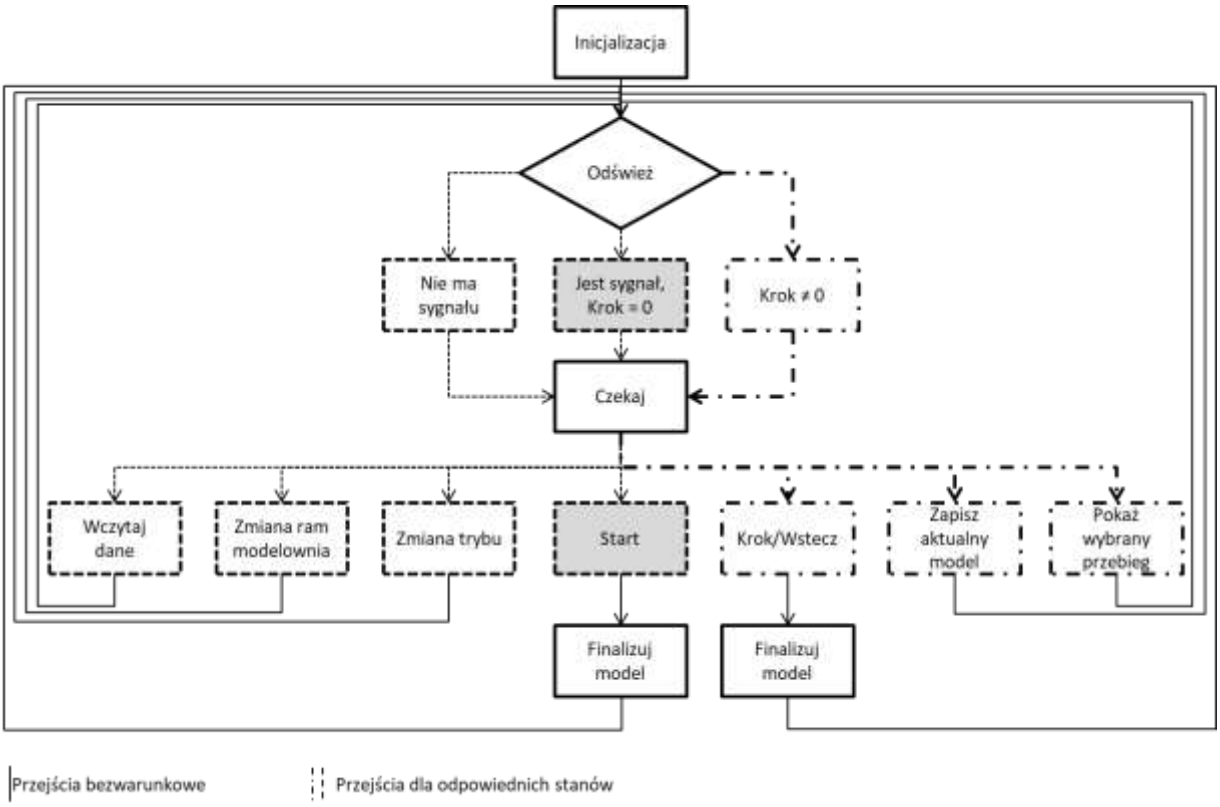
Rysunek 5. Architektura (diagram klas) projektowanego systemu.

3.3 Główna pętla działania programu

Główna pętla programu, przedstawiona na rysunku 6, skupia się na wykonywaniu kroków modelowania w odpowiedniej sekwencji, jednocześnie wypisując na ekran aktualne wartości parametrów modelowania. Wspomnianym wypisywaniem zajmuje się funkcja *Odśwież*, która zostanie opisana w późniejszych rozdziałach dotyczących interfejsu graficznego. Każda interakcja użytkownika z programem kończy się wywołaniem tej funkcji, w celu pokazania aktualnego stanu programu poprzez *GUI*. Najważniejszym skutkiem jej zaimplementowania jest podział działania programu na trzy główne stany:

- Stan początkowy – w pamięci nie ma sygnału, modelowanie nie zostało rozpoczęte.
- Stan gotowości – program posiada wczytany sygnał, jednak modelowanie nie zostało rozpoczęte.
- Stan modelowania – rozpoczęto modelowanie.

W każdym ze stanów użytkownik ma możliwość wykonania różnych akcji, które są adekwatne do aktualnego momentu procesu modelowania. W stanie początkowym, użytkownik posiada możliwość zmiany parametrów modelowania oraz wczytania modelowanego sygnału. Wartości te zapisywane są w elemencie klasy *Dane modelowania*, który odpowiada za przechowywanie tego typu wartości. Gdy modelowany sygnał zostanie wczytany, program przechodzi do stanu gotowości, co daje użytkownikowi możliwość rozpoczęcia modelowania. Wtedy, użytkownik dalej ma możliwość wykonywania komend ze stanu początkowego. Gdy modelowanie zostanie rozpoczęte, program przechodzi do stanu modelowania i poprzednie funkcjonalności zostają zablokowane dla użytkownika. W zamian, może on kontynuować modelowanie, zapisać aktualny model, lub wykreślić żądany przebieg: różnic pomiędzy modelem a sygnałem modelowanym, autokorelacji tych różnic, porównania symulacji modelu z pozostawionymi danymi testowymi oraz różnic tego porównania. Każdy przebieg zostanie przedstawiony w nowym oknie wraz z odpowiednią etykietą oraz oznaczeniami. Użytkownik ma także możliwość cofnięcia się do poprzedniego kroku, co pozwala na przywrócenie poprzednich stanów programu i edycji określonych wcześniej parametrów.



Rysunek 6. Główna pętla działania programu (diagram stanów).

Każdy krok modelowania: pierwszy, następny lub cofnięcie się, wywołuje funkcję finalizującą model. Zostaje ona wywołana w celu przekształcenia modelu ortogonalnego w model właściwy, wyznaczenia jego parametrów, różnic pomiędzy modelem a modelowanym

sygnałem, autokorelacji tych różnic oraz symulacji przyszłych próbek modelu. Obliczenia te zapewniają możliwość nadzorowania działania programu i jego krokowego wykonywania.

3.4 System zapisu regresorów

Ważnym elementem programu jest system zapisu regresorów oraz kontrola ich pozycji. Algorytm FROLS wymaga analizy wszystkich dostępnych regresorów, które są kombinacjami opóźnionych sygnałów mających wpływ na sygnał zwracany przez model. Kombinacje te ulegają zmianie przy każdym sygnale czy innych ramach modelowania. Regresory są jedynie wektorami próbek, które są nierozróżnialne bez znajomości ich wcześniejszego rozmieszczenia. Wobec czego, w celu rozróżnienia oraz lokalizacji regresorów, równolegle z wektorem zawierającym próbki sygnałów konstruowany jest wektor nazw, który obrazuje to, jakie regresory znajdują się w pamięci programu. Operacje na tych wektorach są zawsze identyczne oraz wykonywane są jednocześnie, w celu śledzenia informacji o tym, jakie wektory znajdują się w pamięci programu. Wobec tego, kiedy w niniejszej pracy opisywane jest dodanie lub usunięcie elementu z wektora regresorów, konieczne jest też wykonanie równolegle takiej samej akcji na wektorze nazw regresorów. Przykładowa relacja pomiędzy tymi wektorami została zaprezentowana w tabeli 3, używając przykładowych sygnałów postaci:

$$y = \{1, 2, 3, 4, 5, 6, 7, 8\};\ x_1 = \{1, 2, 3, 4, 5, 6, 7, 8\}$$

Nazwy regresorów zawierają kombinację sygnałów na podstawie których zostały wyznaczone, na przykład, jeśli regresor p jest postaci: $p(k) = y(k-1) \cdot x_1(k-3)$, to zostanie on oznaczony jako: „y-1, x1-3”. Oznaczenie tego typu pozwala zidentyfikować odpowiedni regresor oraz odtworzyć formułę, według której został obliczany, co zostanie później użyte w symulacji modelowanego sygnału.

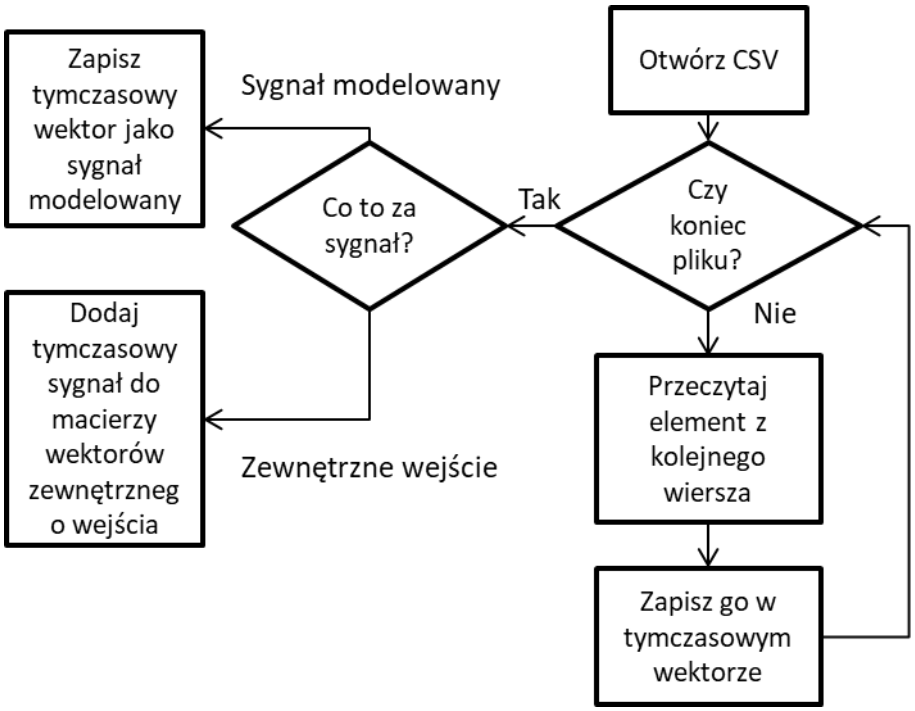
Tabela 3. Przykładowy układ struktury wektora regresorów oraz wektora nazw regresorów, na podstawie sygnałów ().

Wektor regresorów	Wektor nazw	Regresor
⋮	⋮	⋮
$\{0,1,2,3,4,5,6,7\}$	„y-1”	$y(k-1)$
$\{0,0,1,2,3,4,5,6\}$	„y-2”	$y(k-2)$
$\{1,2,3,4,5,6,7,8\}$	„x1-0”	$x_1(k-0)$
$\{0,0,1,4,9,16,25,36\}$	„y-2, y-2”	$y^2(k-2)$
$\{0,2,6,12,20,30,42,56\}$	„y-1, x1-0”	$y(k-1)\,x_1(k-1)$
⋮	⋮	⋮

3.5 Poszczególne procedury

3.5.1 *Solver.*Wczytanie danych

Wczytywanie danych, przedstawione schematycznie na rysunku 7., wygląda tak samo dla sygnału modelowanego jak i zewnętrznych sygnałów wejściowych. Wymagany jest plik o rozszerzeniu .csv, który w każdej nowej linijce ma zapisaną kolejną wartość danego wektora. Program wczytuje plik tego typu do pamięci, a następnie każdą odczytaną wartość zapisuje w wektorze tymczasowym. Po zakończeniu odczytu, w zależności do tego jaki sygnał był wczytywany, tymczasowy wektor zapisywany jest jako modelowany sygnał albo dodawany jest do macierzy wektorów zewnętrznych wejść.



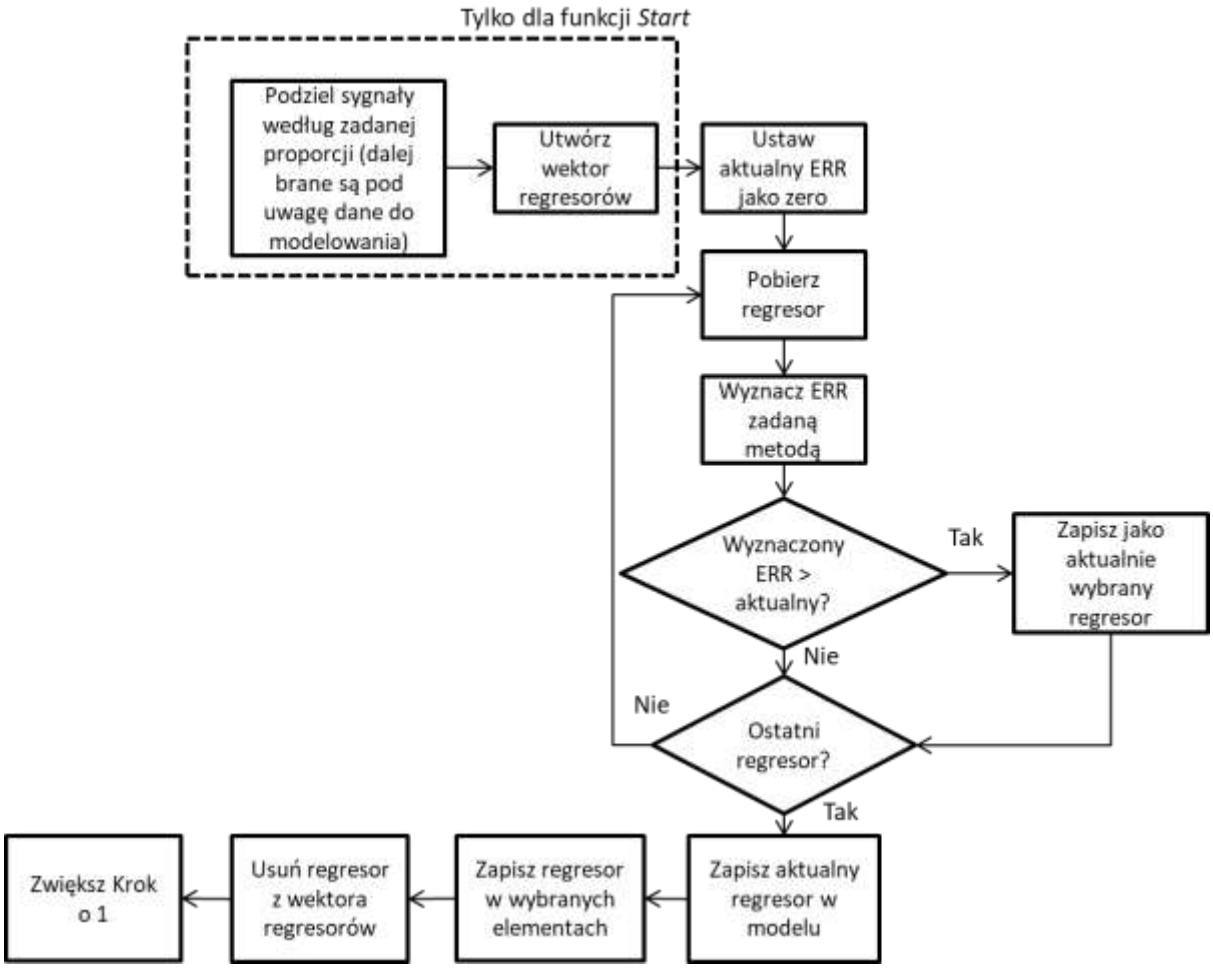
Rysunek 7. Algorytm odczytywania danych.

3.5.2 *Solver.*Start oraz *Solver.*Krok

Rozpoczęcie modelowania przebiega tak samo dla obu wariantów algorytmu FROLS, z wyjątkiem zmian w wyznaczaniu wartości *ERR*, które zostały opisane w rozdziale 1.10. Algorytm, przedstawiony na rysunku 8., rozpoczyna się podziałem sygnałów na dane testowe oraz dane pozostawione do porównania z przyszłą symulacją. Następnie, na podstawie sygnałów znajdujących się w pamięci programu, tworzony jest wektor regresorów, wraz z którym równolegle tworzony jest wektor nazw regresorów. Aktualna najwyższa wartość *ERR* zostaje przyjęta jako zero i rozpoczynana jest praca pętli. Wewnątrz niej, program pobiera regresor na podstawie maksymalnego *ERR.* Współczynnik obliczany jest według formuły z rozdziału 1.101.6.4 lub 1.10, w zależności od wybranej wcześniej metody. Następnie otrzymana wartość porównywana jest z aktualnie zapamiętaną. Jeśli wartość wyznaczona jest wyższa niż zapamiętana, aktualny regresor zostaje przyjęty jako najlepszy i zostaje zapisany w pamięci podręcznej. Po przeanalizowaniu wszystkich dostępnych regresorów, ten, który jest

aktualnie przechowywany w pamięci, uznany zostaje jako najlepszy i zostaje włączony do modelu. Regresor ten zostaje następnie usunięty z wektora regresorów, a także zapisany w wektorze regresorów wybranych, w razie ewentualnego cofnięcia kroku przez użytkownika. Dodatkowo, na koniec licznik kroków zostaje zwiększony o 1.

W wypadku funkcji *Krok*, która jest wykonywana dopiero, jeśli uprzednio wywołana została funkcja *Start*, pomijane są dwa pierwsze kroki, ponieważ dane zostały już podzielone, a wektor regresorów został już skomponowany. Poza tą różnicą, wybór następnego regresora przebiega tak samo.



Rysunek 8. Algorytm wykonywania kroku modelowania – wyboru następnego regresora.

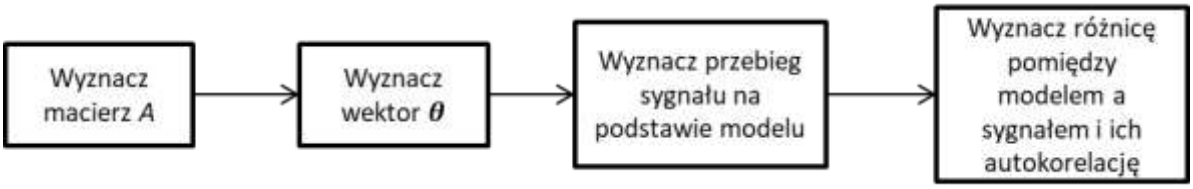
3.5.3 *Solver.*Finalizuj model

Finalizacja modelu odbywa się po każdorazowym wyborze nowego regresora będącego komponentem modelu. Schemat algorytmu tej metody przedstawiono na rysunku 9. Każdorazowe przeprowadzanie tych obliczeń po zakończeniu kroku nie obciąża znacząco pamięci programu. Umożliwia późniejszą analizę poprawności aktualnie skomponowanego modelu oraz ocenę, czy wymagany jest wybór następnych regresorów. Procedura rozpoczyna się od wyznaczenia macierzy A według formuły z rozdziału 1.8. Macierz ta zostaje następnie użyta do rozwiązania równania $A\boldsymbol{\theta} = \boldsymbol{g}$, w celu wyznaczenia wektora parametrów modelu $\boldsymbol{\theta}$. Równanie zostaje rozwiązane za pomocą funkcji *linalg.solve()* z biblioteki *numpy*, która

wykorzystuje odwracanie macierzy A. Mając parametry modelu, wyznaczany jest jego przebieg według formuły:

$$y_{mod}(k) = \sum_{i=1}^{M} \theta_i p_i(k)$$

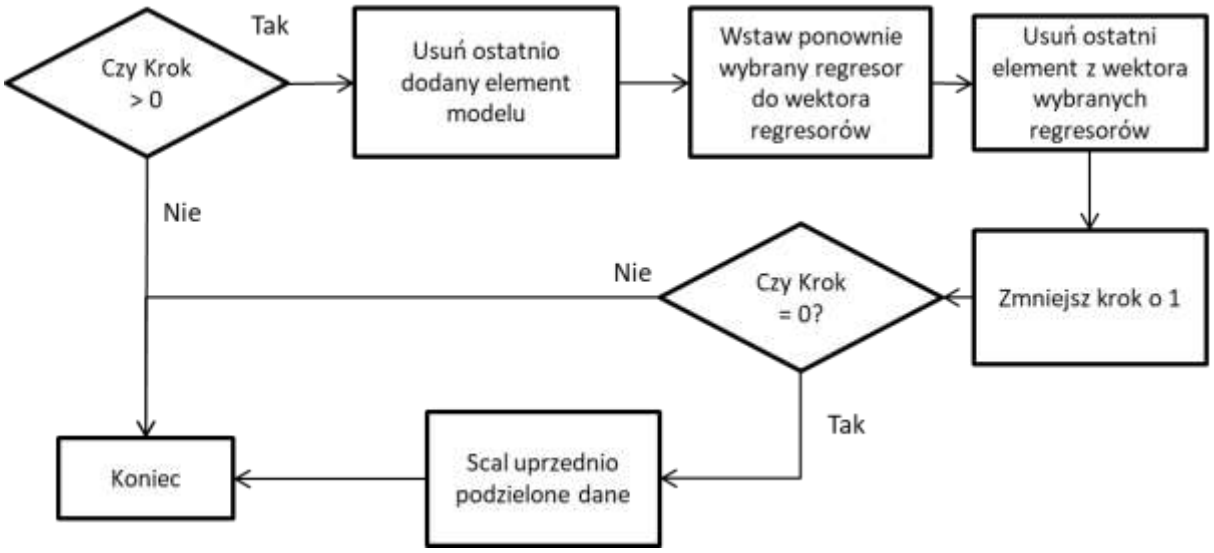
Szum jest w tym wypadku nieznany i pomijany, ponieważ nie wpływa on bezpośrednio na charakter modelu. Finalnie, wyznaczane są różnice pomiędzy sygnałem a przebiegiem wygenerowanym przez model oraz autokorelacja tych różnic.



Rysunek 9. Algorytm finalizacji modelu.

3.5.4 *Solver.*Wstecz

Funkcja odpowiadająca za powrót do poprzedniego kroku modelowania, której algorytm znajduje się na rysunku 10. wywoływana jest jedynie wtedy, gdy aktualny krok jest większy od zera. Oznacza to, że istnieje stan, do którego można powrócić. Procedura usuwa ostatnio dodany człon do modelu z wektorów $\boldsymbol{\theta}, \boldsymbol{g}, \boldsymbol{p}, err$; obliczana jest na nowo macierz A oraz usuwany jest ostatni element z wektora wybranych regresorów. Finalnie, wartość aktualnego kroku jest zmniejszana o jeden. Jeśli po tej zmianie krok jest równy zero, dane testowe oraz dane do modelowania są scalane, gdyż może zajść potrzeba ich modyfikacji .



Rysunek 10. Algorytm powrotu do poprzedniego etapu modelowania.

3.5.5 *Dane modelowania.*Wyznacz wektor regresorów

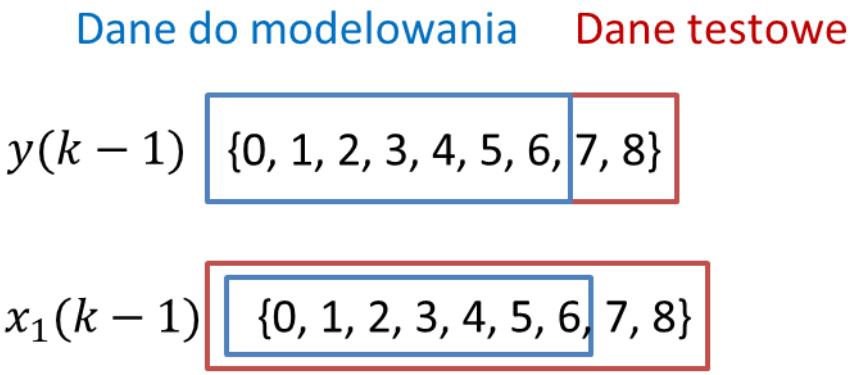
Wektor regresorów wyznaczany jest dwuetapowo, równolegle dla wektora nazw oraz wektora regresorów. Pierwszy etap składa się z generacji bazy regresorów na podstawie znanych sygnałów oraz ich opóźnień. Wtedy generowane są sygnały pierwszego stopnia, na przykład: $y(k-1), y(k-2), x_i(k-0)$ i tym podobne. Następnie, za pomocą funkcji *combinations_with_replacement* generowana jest wariacja z powtórzeniami dla kolejnych poziomów wielomianu. W pierwszej iteracji powstają więc zmienne rzędu drugiego, w drugiej rzędu trzeciego, aż do maksymalnego rzędu określonego przez użytkownika. Po wygenerowaniu wariacji, wektory sygnałów są mnożone przez siebie i zapisywane w wyjściowym wektorze. Wektor nazw regresorów nie wymaga mnożenia, zachowuje więc on formę elementów wymienionych po przecinku, tak jak w tabeli 3. Finalnie, do wektora dodawany jest także sygnał skomponowany z samych próbek o wartości 1, nazwany „const.". Oznacza on składową stałą, której wartość regulowana jest poprzez parametr który zostanie do niej przypisany.

3.5.6 *Dane modelowania.*Podziel/scal dane

Podział danych znajdujących się w pamięci programu wykonywany jest w miejscu określonym przez użytkownika poprzez odpowiedni ułamek. Indeks, który zostanie uznany jako indeks graniczny obliczany jest za pomocą formuły:

$$Indeks \approx podany\ ułamek \cdot długość\ danych$$

gdzie zaokrąglenie wykonywanie jest do liczby naturalnej. Następnie wszystkie sygnały: modelowany oraz wejścia zewnętrzne dzielone są na dane służące do modelowania oraz na dane testowe. Schematycznie, podział ten został przedstawiony na rysunku 11, dla użytych już wcześniej przykładowych sygnałów z równania ().



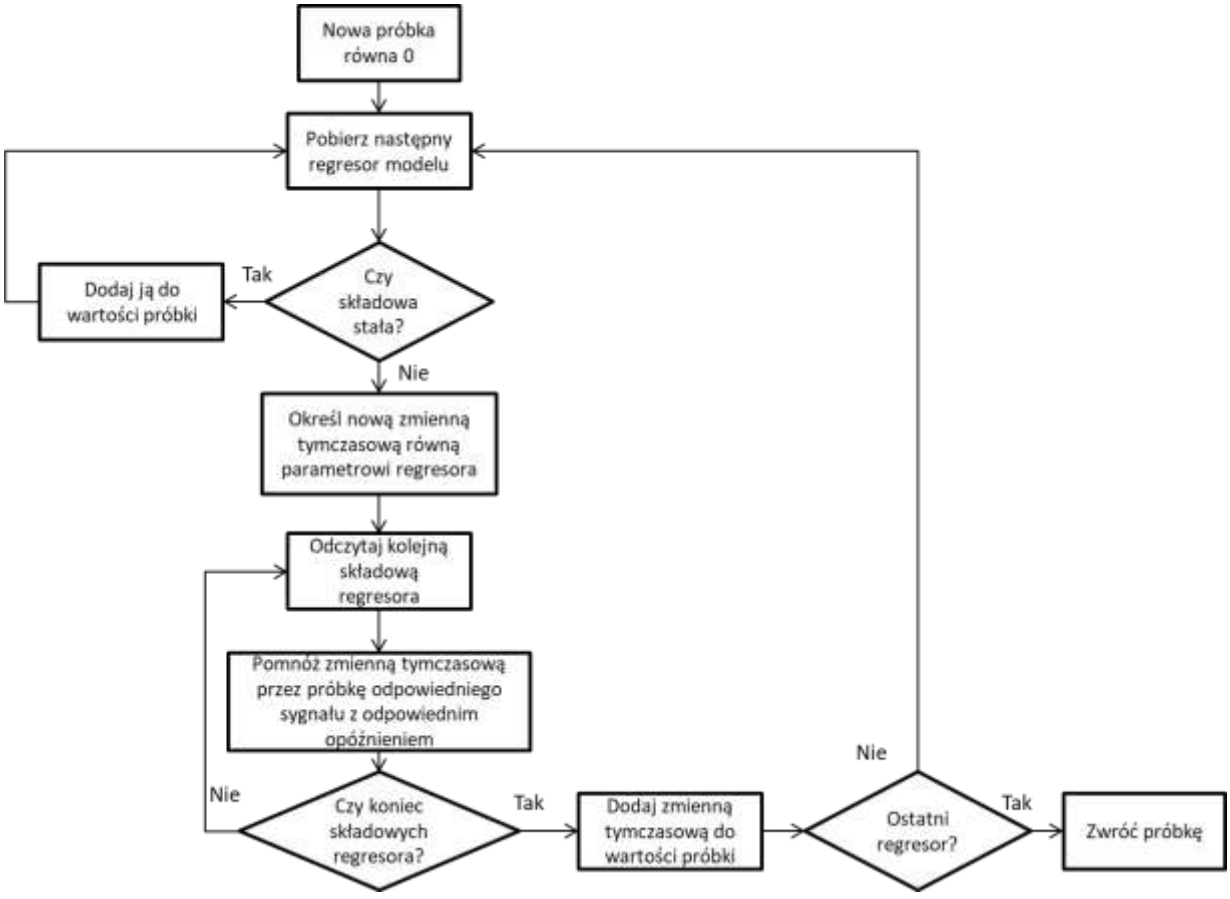
Rysunek 11. Schemat podziału sygnałów na dane służące do modelowania oraz dane testowe.

Dane do modelowania są początkowymi częściami wektorów do wyznaczonego indeksu wektorów sygnałów od pierwszego elementu do elementu o wyznaczonym indeksie. Sytuacja ta jest taka sama dla sygnału modelowanego oraz dla sygnałów wejść zewnętrznych. Różnice dotyczą danych, które zostają pozostawione jako testowe. W przypadku modelowanego sygnału dane testowe są pozostałą częścią pierwotnego wektora, ponieważ służą jedynie do porównania z wynikiem symulacji. Dane wejściowe wykorzystywane są przy przeprowadzaniu symulacji, wobec czego wymagane jest posiadanie w pamięci ich pełnego przebiegu. Z tego powodu, jako dane testowe zapisywany jest pełny przebieg sygnałów wejść zewnętrznych.

3.5.7 *Model.*Symulacja

Symulacja danych wykonywana jest próbka po próbce, inaczej niż w wypadku wyznaczania wartości sygnału według aktualnego modelu. Tak jak zostało to opisane w rozdziale 3.5.3, obliczenie wartości sygnału według danego modelu wymaga wykonania znanych operacji na wektorach. W celu symulacji przyszłych wartości sygnału, wymagany jest odczyt nazw regresorów z pamięci modelu. Następnie, znając wymagane opóźnienia, iteracyjnie wyznaczana jest wartość kolejnej próbki. Schemat tego algorytmu został przedstawiony na rysunku 12.

Wyznaczanie wartości kolejnej próbki, rozpoczyna się od zainicjowania jej wartości jako zero. Następnie, kolejno pobierane są nazwy regresorów z aktualnego modelu, gdzie wiadomo, że każdy regresor może składać się z kilku członów na przykład: $y(k-1) \cdot y(k-2)$. Na początku sprawdzane jest, czy regresor nie jest składową stałą, która zawsze występuje samotnie. Wtedy, parametr składowej stałej dodawany jest do wartości próbki i pobierany jest następny regresor. W przeciwnym wypadku, inicjowana jest nowa zmienna tymczasowa, o wartości równej parametrowi danego regresora. Następnie, w zagnieżdżonej pętli tymczasowa zmienna mnożona jest przez odpowiednie próbki sygnałów odczytane według członów regresora. W wyniku działania tej pętli, zmienna tymczasowa będzie posiadała wartość iloczynu odpowiednich próbek składających się na regresor oraz parametru znajdującego się przy nim.



Rysunek 12. Algorytm wyznaczania kolejnej próbki w sygnale symulowanym.

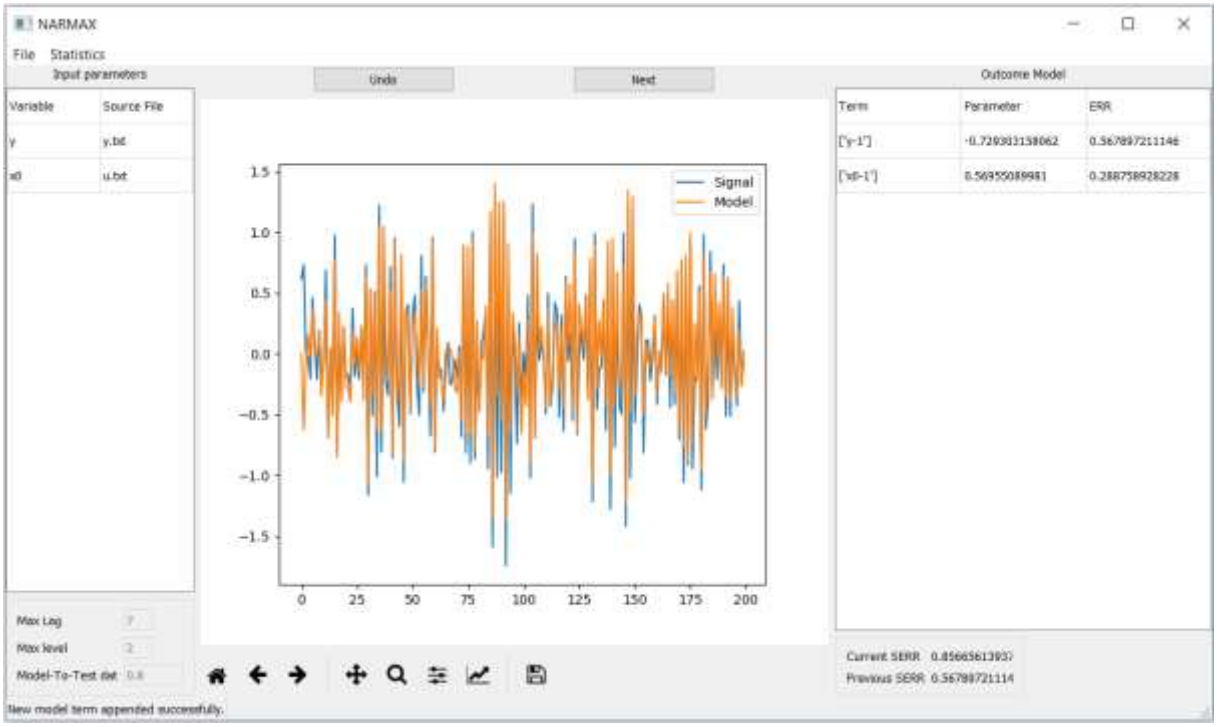
Po wyznaczeniu wartości kontrybucji danego regresora do wartości próbki w danej chwili, jest ona dodawana do wartości próbki, która jest aktualnie obliczana. Procedura ta jest powtarzana dla każdego regresora, w wyniku czego wyznaczona zostaje wartość próbki w danej chwili. Algorytm generuje tyle próbek, ile posiadają dane pozostawione do testów. Istnieje wtedy możliwość ich porównania oraz określenia różnic pomiędzy symulacją a danymi testowymi.

3.6 Interfejs graficzny

Interfejs graficzny (*GUI*), opracowany został za pomocą biblioteki *PyQt5* [10], przedstawiony został na rysunku 13. Można wyróżnić w nim trzy główne bloki:

- lewy (parametrów modelowania),
- środkowy (sterowania modelowaniem),
- prawy (wyników modelowania).

Lewy blok służy do modyfikacji maksymalnego opóźnienia, maksymalnego poziomu nieliniowości i miejsca podziału danych na część używaną do modelowania oraz dane testowe. Blok środkowy służy do wykreślenia przebiegu sygnału modelowanego oraz tego, który jest wyjściem aktualnego modelu. Dodatkowo znajdują się w nim przyciski pozwalające na wykonywanie kolejnych kroków modelowania a także manipulacji wykresem. W bloku prawym przedstawione są szczegóły aktualnego modelu: wybrane regresory, ich parametry oraz *ERR*. Dodatkowo podana jest *SERR* dla modelu oraz to jaka była wartość tej sumy w poprzedni kroku. *GUI* posiada również menu, pasek stanu komunikujący użytkownikowi operacje, które są wykonywane przez program. W menu użytkownik może wybrać takie akcje jak: wczytanie sygnału, zapis aktualnego modelu do pliku .txt, czy wyświetlenie wyników testowania aktualnego modelu.



Rysunek 13. Interfejs graficzny programu.

3.6.1 Funkcja *Odśwież*

GUI pokazywane jest za pomocą funkcji *Odśwież*, która jest wywoływana po każdej akcji wykonanej przez program, tak jak na rysunku 6. Określa ona trzy główne sytuacje wymagające dostosowania interfejsu graficznego:

1. Brak wczytanego sygnału modelowanego.
2. Sygnał modelowany został wczytany, ale aktualny krok jest równy zero.
3. Krok jest różny od zera.

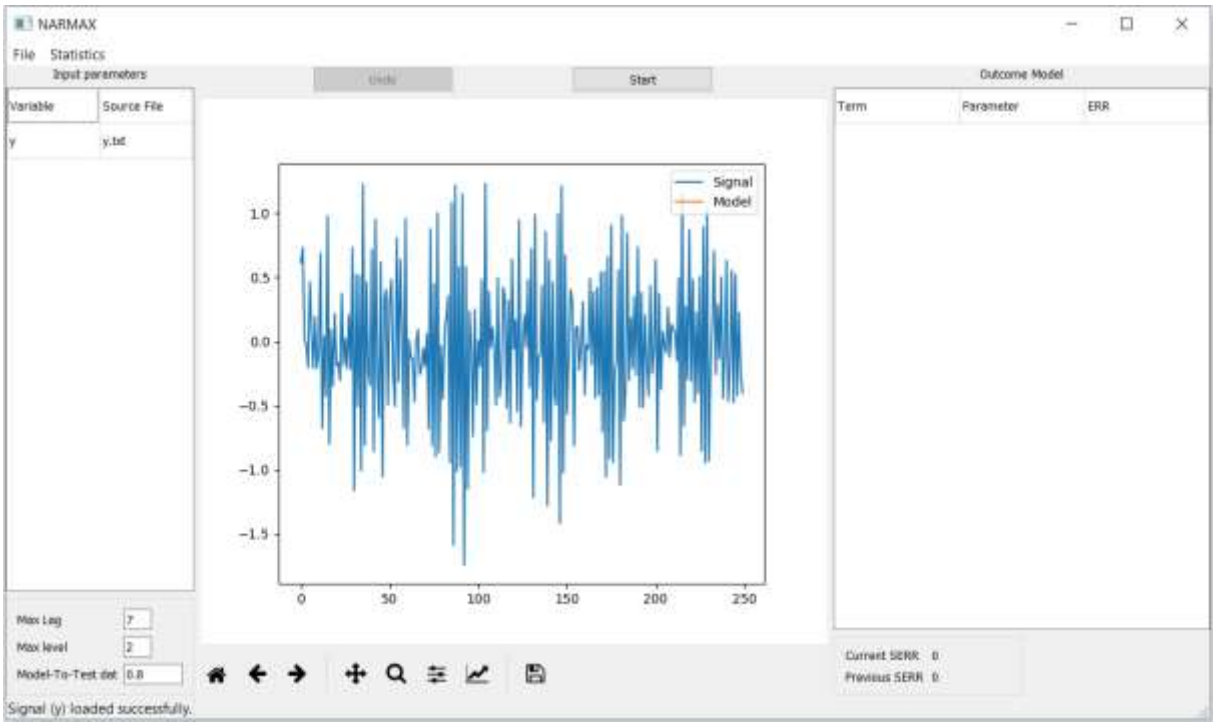
W sytuacji początkowej istnieje możliwość doboru parametrów modelowania w lewym dolnym rogu, jednak przyciski: „Start” oraz „Wróć” są nieaktywne. W menu dostępne są jedynie polecenia:

- „Wybór trybu” - oznacza wybór wariantu algorytmu FROLS.
- „Wczytaj sygnał”.

Po wczytaniu sygnału krok jest dalej równy zero, więc poprzednio możliwe akcje są dalej dostępne, ale dodatkowo możliwy jest wybór akcji:

- „Wczytaj wejście zewnętrzne”
- „Start”

Wczytanie wejść zewnętrznych zostaje umożliwione dopiero po wczytaniu sygnału modelowanego, w celu sprawdzenia długości wczytywanych danych. Najważniejszy jest sygnał modelowany. Jeśli wczytywany sygnał wejść zewnętrznych jest zbyt długi, zostaje on skrócony tak, aby jego długość odpowiadała modelowanemu sygnałowi. Gdy jest on za krótki, nie zostaje on zapisany, a użytkownik otrzymuje stosowny komunikat. Interfejs po wczytaniu sygnału przedstawiony został na rysunku 14.



Rysunek 14. Wygląd *GUI* po wczytaniu sygnału.

Po naciśnięciu przycisku „Start”, wykonywany jest pierwszy krok. Wobec czego wartość zmiennej *Krok* jest różna od zera i wyświetlany jest trzeci wariant *GUI*. Możliwe dla niego, akcje:

- „Wybór trybu”,
- „Wczytaj sygnał”,
- „Wczytaj wejście zewnętrzne”,
- modyfikacje parametrów modelowania.

Są nieaktywne, gdyż zmiana parametrów modelowania w jego trakcie jest niewskazana i prowadzi do powstawania błędów. Możliwe stają się za to akcje:

- „Zapisz” – zapisuje parametry modelu do pliku .txt,
- „Wykreśl autokorelację różnic modelu”,
- „Wykreśl przebieg różnic modelu”,
- „Wykreśl przebieg symulacji aktualnego modelu”,
- „Wykreśl przebieg różnic pomiędzy symulacją a danymi testowymi”,
- „Wykreśl przebieg SERR”,
- przycisk „Wstecz” staje się aktywny,
- przycisk „Start” zmienia tekst na „Następny krok”.

Dodatkowo, niezależnie od aktualnego stanu, program wyświetla elementy aktualnego modelu w prawym panelu. Jeśli ich nie ma, pola pozostają puste, jednak nie wymaga to wprowadzenia zależności od stanu programu. To samo dotyczy wykreślenia przebiegu sygnału oraz wyjścia aktualnego modelu. Próba ich wykreślenia jest zawsze podejmowana, lecz jeśli zmienne zawierające ich wartości są puste, żaden przebieg nie jest rysowany.

4. Wyniki

Za pomocą programu wykonane zostało modelowanie dla trzech sygnałów:

- Liniowego sygnału stochastycznego: $y(k) = y(k-1) - y(k-2) - x(k-1) + e(k)$, gdzie $x(k)$ pochodzi z rozkładu równomiernego z zakresu $(-1,1)$, a $e(k)$ – szum pochodzący z rozkładu Gaussa o zerowej średniej.
- Nieliniowego sygnału stochastycznego: $y(k) = -0{,}6y(k-1) - 0{,}15y^2(k-2) + 0{,}5x(k-1) - 0{,}25x(k-2) + e(k)$, gdzie $x(k)$ pochodzi z rozkładu równomiernego z zakresu $(-1,1)$, a $e(k)$ – szum pochodzący z rozkładu Gaussa o zerowej średniej.
- Funkcji logistycznej: $y(k) = 3{,}5y(k-1) - 3{,}5y^2(k-1) + e(k)$, gdzie $e(k)$ – szum pochodzący z rozkładu Gaussa o zerowej średniej.

Każdy sygnał modelowano za pomocą algorytmu FROLS, opisanego w rozdziale 1.6, oraz za pomocą zmodyfikowanego algorytmu FROLS, opisanego w rozdziale 1.10.

Dla każdego z sygnałów zbadano wpływ wybranej metody, obecności szumu oraz długości sygnału na wynik modelowania. Wygenerowano sygnały dla szumu $e(k)$ o wariancji równej:

- 0,05
- 0,1
- 0,2
- 0,4
- 0,5

Wyjątkiem był przypadek funkcji logistycznej, która przyjmuje wartości z zakresu [0,1], co powoduje że szum wpływa na ten sygnał w znacznie większym stopniu. Z tego powodu, dla funkcji logistycznej wygenerowano przebiegi, do których dodany szum posiadał wariancję dwukrotnie mniejszą.

Według autorów metody [6], powinna ona działać poprawnie dla sygnału o długości 300-500 próbek. Wobec czego, zdecydowano się na zbadanie sygnałów o następujących długościach (podanych w liczbie próbek):

- 200
- 300
- 500
- 700
- 1000

Dla wszystkich 75 sygnałów przeprowadzono modelowanie z przyjętymi parametrami:

- Maksymalne opóźnienie – 7
- Maksymalny stopień nieliniowości – 3
- Punkt podziału danych na dane testowe oraz dane modelowania – 0,8

W każdym wypadku modelowanie prowadzono do momentu otrzymania $SERR > 0{,}95$ lub gdy przebieg SERR ulegał spłaszczeniu. Jeśli wybrana liczba regresorów do wyznaczenia modelu przekroczyła dziesięć, modelowanie uznawano za zakończone niepowodzeniem, bez względu na wartość *SERR*. Modelowanie uznawano także za poprawne, jeśli analiza autokorelacji różnic modelu oraz symulacji dały wynik pozytywny.

Wyniki modelowań zestawiono w tabelach 4., 5., 6., 7. oraz 8. Każda tabela dotyczy modelowania jednego sygnału za pomocą danego algorytmu. W każdej tabeli wpisano otrzymane końcowe wartości *SERR* oraz średnie procentowe odchylenia wyznaczonych parametrów od ich wartości rzeczywistych (δ). Kolorem zielonym zaznaczono sytuacje, gdy opracowany model był poprawny, a kolorem czerwonym błędne wyniki modelowania. Kolorem żółtym oznaczono sytuacje niejednoznaczne, których wynik nie daje się zakwalifikować do żadnej ze wcześniejszych grup. Dodatkowo, kolejnymi literami oznaczono grupy przypadków, które będą następnie omawiane.

Tabela 4. Wyniki modelowania sygnału liniowego niezmodyfikowanym algorytmem FROLS.

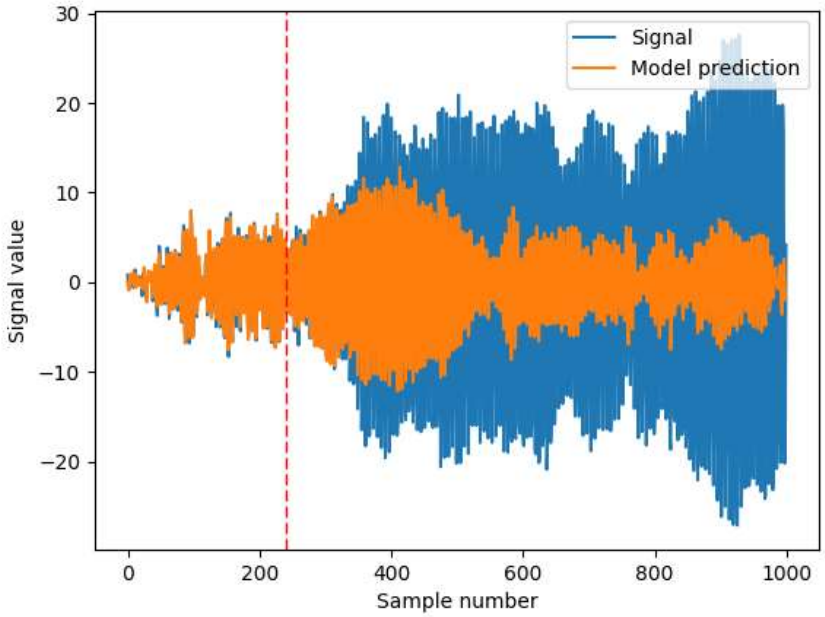
	Wariancja dodanego szumu				
Długość sygnału	0,05	0,1	0,2	0,4	0,5
200	SERR=0,9945	SERR =0,9909	SERR =0,9844	SERR =0,9581	SERR =0,9537
300	SERR =0,9975	SERR =0,9930	SERR =0,9570	SERR =0,9574	SERR =0,9975
500	SERR =0,9790	SERR =0,9701	SERR =0,9827	SERR =0,9837	SERR =0,9689
700	SERR =0,9898	ERR=0,9850	ERR=0,9898	ERR=0,9937	ERR=0,9838
1000	SERR =0,9945	SERR =0,9838	SERR =0,9886	SERR =0,9953	SERR =0,9880

Podczas modelowania sygnału liniowego niezmodyfikowaną metodą FROLS (tabela 4.), program w żadnym z wypadków nie znalazł odpowiedniego modelu. W każdym z nich, pierwszym wybieranym regresorem był $(y-3)$, którego *ERR* wynosiło ok. 0,8. Wobec czego każdy następny model, pomimo nasycenia *SERR* oraz jego wysokiej wartości, był modelem błędnym i nie był w stanie zwrócić poprawnej symulacji danych wyjściowych. Sytuacja ta była motywacją do modyfikacji algorytmu.

Tabela 5. Wyniki modelowania sygnału liniowego zmodyfikowanym algorytmem FROLS.

	Wariancja dodanego szumu				
Długość sygnału	0,05	0,1	0,2	0,4	0,5
200	SERR=0,9946 *A*	SERR=0,9953 *B* δ=1,02%	SERR=0,9947 *B* δ=0,91%	SERR=0,9784 *C* δ=3,27%	SERR =0,9699 *C* δ=3,19%
300	SERR=0,9699 *B* δ=3,19%	SERR=0,9964 *B* δ=1,13%	SERR=0,9968 *B* δ=3,52%	SERR=0,9732 *C* δ=1,58%	SERR=0,9765 *C* δ=2,80%
500	SERR=0,9998 *B* δ=0,19%	SERR=0,9993 *B* δ=0,55%	SERR=0,9988 *D* δ=0,91%	SERR=0,9972 *D* δ=1,23%	SERR=0,9933 *D* δ=1,40%
700	SERR=0,9999 *B* δ=0,09%	SERR=0,9997 *D* δ=0,56%	SERR=0,9993 *D* δ=0,60%	SERR=0,9989 *D* δ=1,48%	SERR=0,9966 *D* δ=1,60%
1000	SERR=0,9999 *D* δ=0,01%	SERR=0,9997 *D* δ=0,47%	SER =0,9993 *D* δ=0,75%	SERR=0,9992 δ=1,43%	SERR=0,9975 *D* δ=1,32%

Gdy modelowano sygnał liniowy zmodyfikowaną metodą FROLS, jedynie w jednym przypadku (A) program błędnie wybrał regresor $(y-3)$ jako pierwszy. W przypadkach poprawnego modelowania (B) wszystkie metody weryfikacji potwierdziły poprawność modeli oraz otrzymane parametry zgadzają się co do wartości z założonym sygnałem. Dla wyników oznaczonych literą C, otrzymany model posiadał poprawne parametry, jednak symulacje modelu były niepoprawne. Przykład takiego przebiegu przedstawiono na rysunku 15. Wobec tego, pomimo iż model posiada poprawne parametry, nie można uznać go za w pełni poprawny. Dla przebiegów oznaczonych literą D, przebieg $SERR$ wskazałby na wybranie jedynie modelu dwuelementowego, z pominięciem członu $x(k-1)$. Doprowadziłoby to do wyboru modelu błędnego. Jednak gdyby rozszerzono model o kolejny człon, otrzymany byłby poprawny. Z tego powodu, nie można uznać tego modelowania za w pełni błędne, gdyż istnieje szansa że, znaleziony model byłby poprawny, jednak nie jest to pewne.

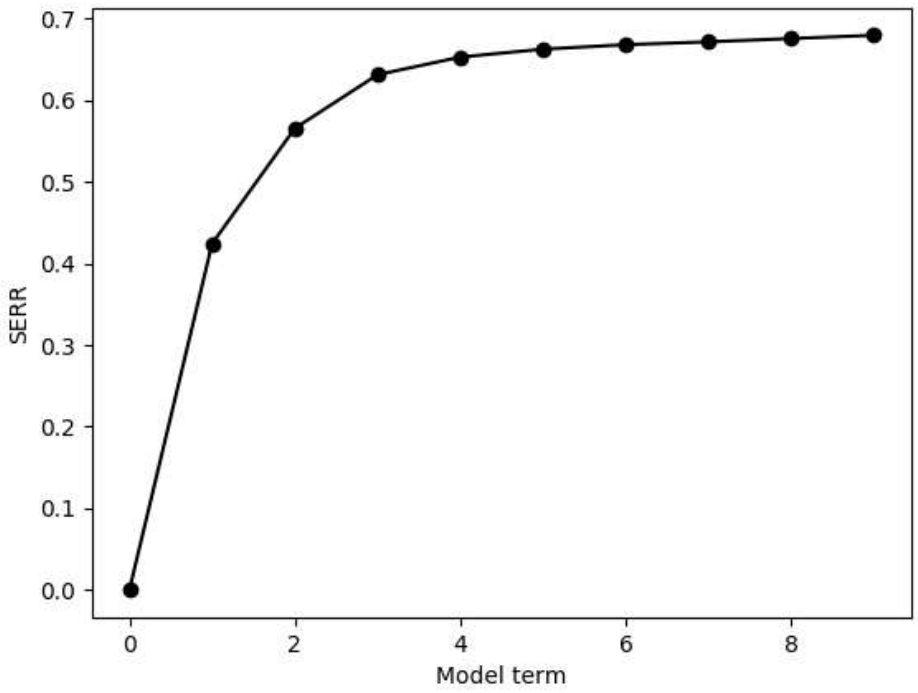


Rysunek 15. Błędna predykcja wyniku modelowania sygnału liniowego za pomocą zmodyfikowanego algorytmu FROLS.

Tabela 6. Wyniki modelowania sygnału nieliniowego algorytmem FROLS (wyniki były takie same dla obu wariantów algorytmu).

	Wariancja dodanego szumu				
Długość sygnału	0,05	0,1	0,2	0,4	0,5
200	SERR=0,9649 *A* δ=1,51%	SERR=0,9454 *A* δ=6,56%	SERR=0,8789 *C* δ=11,37%	SERR=0,7142 *D*	SERR=0,8027 *D*
300	SERR=0,9709 *A* δ=1,84%	SERR=0,9458 *A* δ=3,94%	SERR=0,8713 *B* δ=0,84%	SERR=0,7078 *D*	SERR=0,6292 *E* δ=10,46%
500	SERR=0,9784 *A* δ=1,03%	SERR=0,9532 *A* δ=4,50%	SERR=0,8720 *B* δ=4,02%	SERR=0,6892 *E* δ=12,49%	SERR=0,6526 *E* δ=10,59%
700	SERR=0,9808 *A* δ=0,82%	SERR=0,9554 *A* δ=2,67%	SERR=0,8720 *B* δ=5,35%	SERR=0,6683 *E* δ=6,12%	SERR=0,6117 *E* δ=10,01%
1000	SERR=0,9841 *A* δ=0,54%	SERR=0,9576 *A* δ=2,02%	SERR=0,8734 *B* δ=1,45%	SERR=0,6848 *E* δ=2,76%	SERR=0,6129 *E* δ=8,31%

Wyniki modelowania sygnału nieliniowego są takie same dla obu badanych metod, gdyż we wszystkich przypadkach wybrane regresory były identyczne oraz posiadały te same wartości *ERR*. W większości przypadków modelowanie dało wynik pozytywny. Dla sygnałów oznaczonych literą A, otrzymano poprawny wynik, potwierdzony metodami weryfikacji. Dla sygnałów oznaczonych literą B, wybrany model był poprawny, poprawne były także symulacje wykonywane za pomocą tego modelu, jednak końcowa wartość *SERR* była niska, co mogłoby budzić wątpliwości. Ostatecznie, otrzymano właściwy model, który poprawnie przeszedł weryfikację. W przypadku C, pojawia się niejasność związana z gładkim nasyceniem krzywej *SERR*, co nie wskazuje na konkretną liczbę wymaganych regresorów. Jednak analiza wszystkich modeli wokół przegięcia charakterystyki mogłaby doprowadzić do otrzymania modelu poprawnego. Dla przypadków oznaczonych literą D, modelowanie nie przyniosło zakładanego skutku, gdyż z wykresu *SERR* nie można było wskazać poprawnej długości modelu. Dla przypadków oznaczonych literą E, sytuacja była podobna, jednak występowało niewielkie przegięcie charakterystyki SERR, które mogłoby być przesłanką do miejsca poszukiwania poprawnego modelu i daje szanse jego otrzymania. Przykład przebiegu tego typu przedstawiony został na rysunku 16.

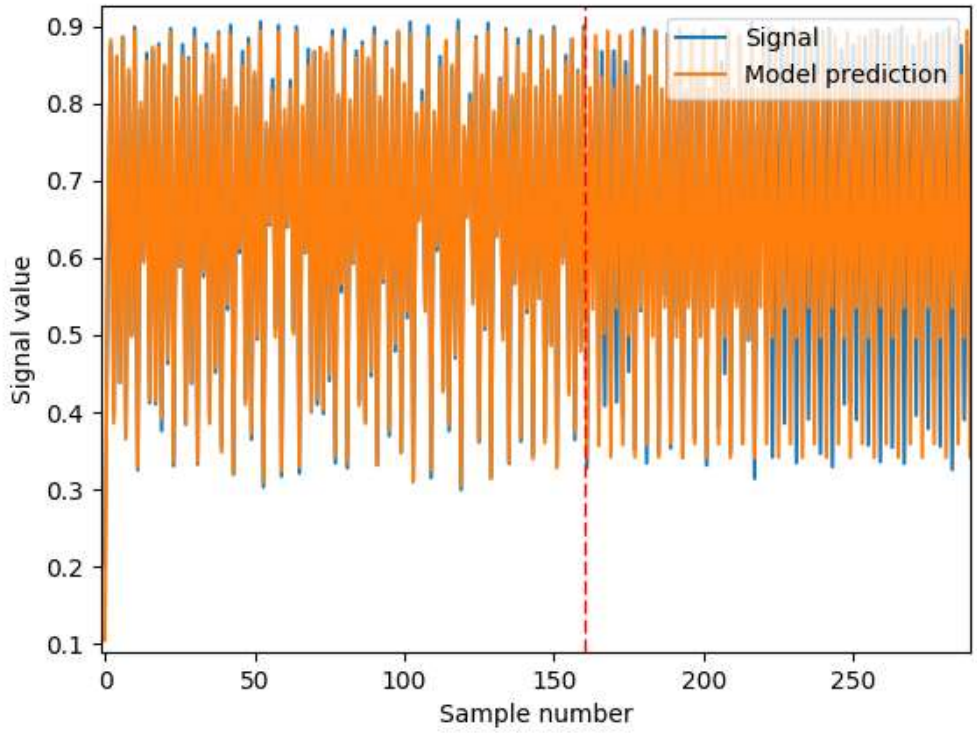


Rysunek 16. Przykładowy przebieg SERR dla sygnału nieliniowego z szumem o wariancji 0,5.

Tabela 7. Wyniki modelowania sygnału funkcji logistyczne za pomocą niezmodyfikowanego algorytmu FROLS.

	Wariancja dodanego szumu			
Długość sygnału	0,005	0,01	0,02	0,04
200	SERR =0,9997	SERR =0,9995	SERR =0,9988	SERR =0,9957
300	SERR =0,9998	SERR =0,9996	SERR =0,9989	SERR =0,9963
500	SERR =0,9998	SERR =0,9996	SERR =0,9990	SERR =0,9967
700	SERR =0,9998	SERR =0,9997	SERR =0,9991	SERR =0,9966
1000	SERR =0,9999	SERR =0,9997	SERR =0,9991	SERR =0,9965

W przypadku wyników modelowania funkcji logistycznej za pomocą niezmodyfikowanej metody FROLS, żadne modelowanie nie zwróciło poprawnego wyniku. Program wybierał regresor $(y-4)$ jako pierwszy, co prowadziło do wskazania błędnego modelu, pomimo wysokich wartości *SERR*. Spowodowane to było jego wysoką wartością ERR, która zwykle wynosiła ≈0,8, czyli znacznie więcej niż pozostałe. Następnie wybierana była składowa stała, a po niej poprawne elementy modelu. Po ich wyborze wykres SERR wskazywał na zaprzestanie modelowania, a parametr stojący przy $(y-4)$ był z reguły niewielki (≈0,005) w porównaniu do parametrów znajdujących się przy rzeczywistych regresorach tworzących model (pomiędzy 3 a 3,5). Pomimo to wybierany model był błędny i jego weryfikacja nie przebiegala pomyślnie. Przykład symulacji otrzymanego wyniku przedstawiono na rysunku 17.

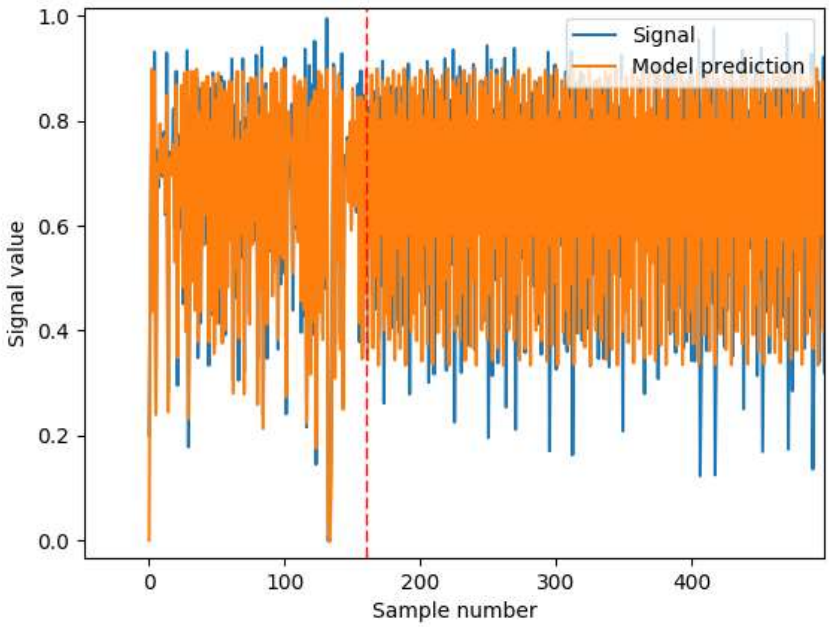


Rysunek 17. Wynik symulacji modelu opracowanego na podstawie sygnału funkcji logistycznej za pomocą niezmodyfikowanej metody FROLS.

Tabela 8. Wyniki modelowania sygnału funkcji logistycznej za pomocą zmodyfikowanego algorytmu FROLS.

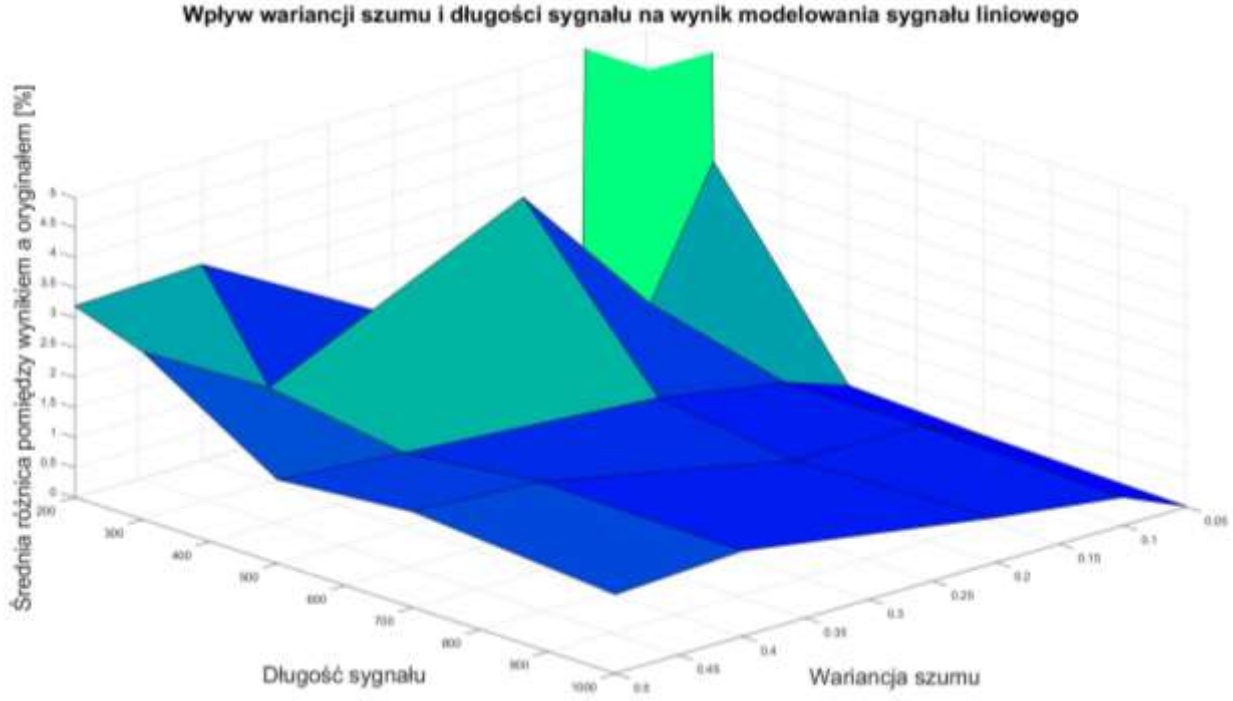
	Wariancja dodanego szumu				
Długość sygnału	0,025	0,05	0,1	0,2	0,25
200	SERR =0,9981 *A* δ=0,50%	SERR =0,9730 *A* δ=2,70%	SERR =0,9738 *A* δ=0,50%	SERR=0,8962 *B* δ=2,54%	SERR =0,8856 *B* δ=8,54%
300	SERR =0,9984 *A* δ=0,18%	SERR =0,9740 *A* δ=2,11%	SERR =0,9760 *A* δ=0,18%	SERR=0,9076 *B* δ=3,81%	SERR =0,8906 *B* δ=4,88%
500	SERR =0,9985 *A* δ=0,58%	SERR =0,9766 *A* δ=0,93%	SERR =0,9776 *A* δ=0,54%	SERR=0,9174 *B* δ=3,85%	SERR =0,8977 *B* δ=6,45%
700	SERR =0,9985 *A* δ=0,07%	SERR =0,9771 δ=0,63%	SERR =0,9789 *A* δ=0,71%	SERR=0,9176 *B* δ=8,11%	SERR =0,8894 *B* δ=6,13%
1000	SERR =0,9986 *A* δ=0,19%	SERR =0,9768 *A* δ=0,70%	SERR =0,9790 *A* δ=1,74%	SERR=0,9151 *B* δ=4,89%	SERR=0,8837 *B* δ=6,83%

W przypadku modelowania sygnałów pochodzących od funkcji logistycznej zmodyfikowanym algorytmem FROLS, w przypadkach oznaczonych literą A, metoda zwracała poprawny model, który poprawnie przechodził weryfikację. Dla przypadków oznaczonych literą B, otrzymywany model był poprawny, jednak symulacje znacznie odbiegały od przebiegów rzeczywistych, przykład przedstawiono na rysunku 18. Wobec tego, pomimo poprawności otrzymanego modelu, nie można uznać że zostanie za poprawny uznany w przypadku nieznajomości oryginalnego modelu.

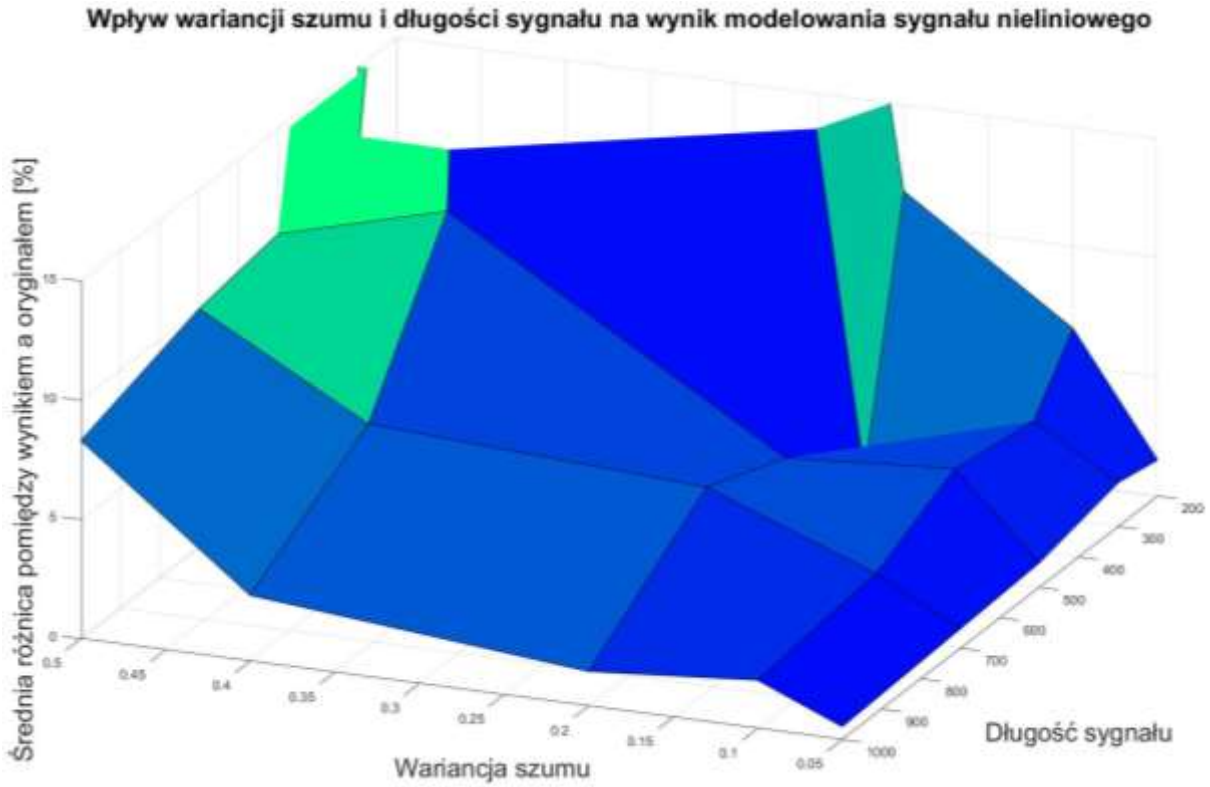


Rysunek 18. Wynik symulacji modelu opracowanego na podstawie sygnału funkcji logistycznej za pomocą zmodyfikowanej metody FROLS dla szumu o wariancji 0,04.

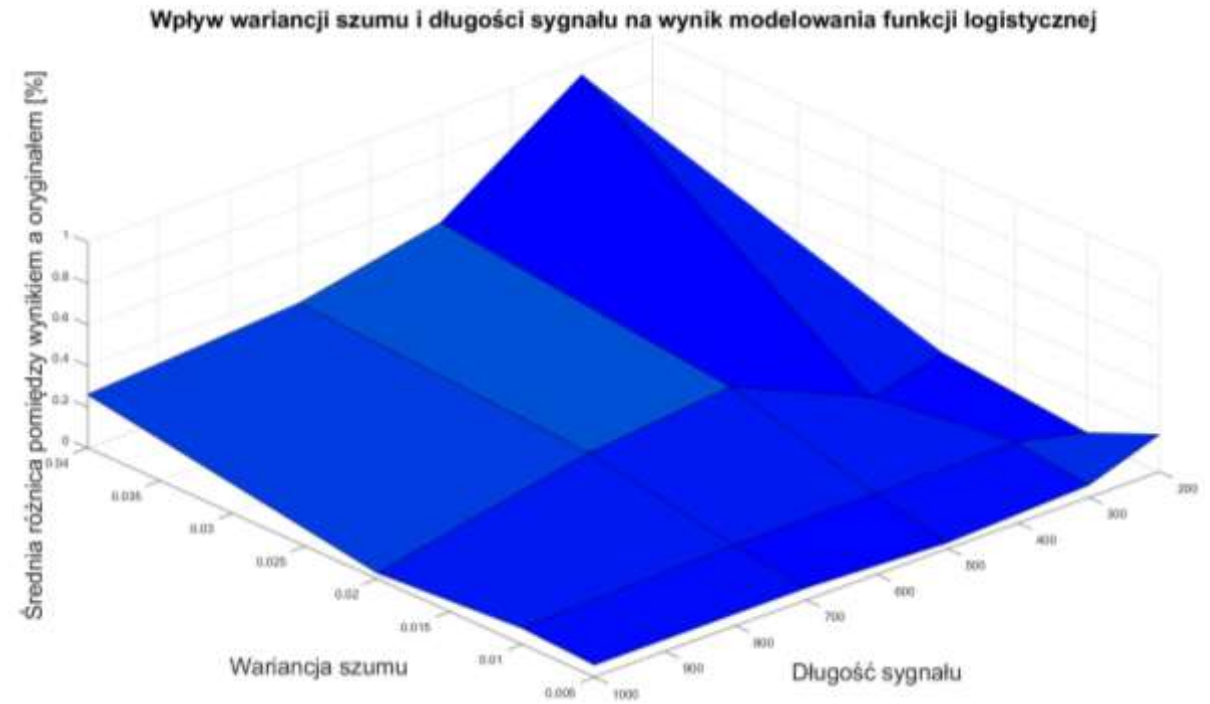
Wartości różnic pomiędzy wyznaczonym modelem a oryginalnym przedstawiono na rysunkach 19., 20. i 21. w celu zobrazowania wpływu wariancji szumu oraz długości sygnału na wynik modelowania. W przypadkach gdy model wyznaczony został błędnie, przyjęto bardzo wysoką wartość różnicy, znacznie poza skalą. Dla wyników błędnych, przyjęto bardzo wysoką wartość błędu, dążącą do nieskończoności.



Rysunek 19. Wyniki modelowania z tabeli 5 przedstawione w formie wykresu trójwymiarowego.



Rysunek 20. Wyniki modelowania z tabeli 6 przedstawione w formie wykresu trójwymiarowego.



Rysunek 21. Wyniki modelowania z tabeli 8 przedstawione w formie wykresu trójwymiarowego.

4.1 Analiza czasu poszukiwania

W celu ewaluacji szybkości wykonywania użytych algorytmów, zmierzono średni czas poszukiwania kolejnego regresora dla trzech wybranych długości sygnału. Według warunków podanych na początku rozdziału oraz formuły z rozdziału 1.6.3, liczba potencjalnych regresorów będzie równa:

$$M = \binom{7+7+3}{3} = 680$$

Dla zmodyfikowanego algorytmu FROLS, przeszukiwanie odbywa się dwukrotnie, wobec czego dla niego:

$$M' = M^2 = 462400$$

Każdy regresor wymaga przeanalizowania, więc spodziewano się, że czas wykonywania algorytmu zmodyfikowanego będzie zdecydowanie dłuższy. Wyniki przedstawiono w tabeli 9. Przedstawione wyniki są rezultatem uśrednienia dwudziestu pomiarów. Dane zbierano w trakcie modelowania sygnału nieliniowego. Modelowanie wykonywano za pomocą komputera wyposażonego w procesor Intel® Core™ i7-6700K CPU @ 4.00GHz.

Tabela 9. Wyniki pomiarów czasu wykonywania algorytmu FROLS oraz jego zmodyfikowanej wersji dla maksymalnego opóźnienia równego 7 oraz maksymalnego stopnia nieliniowości równego 3. Sygnał posiada jedno wejście.

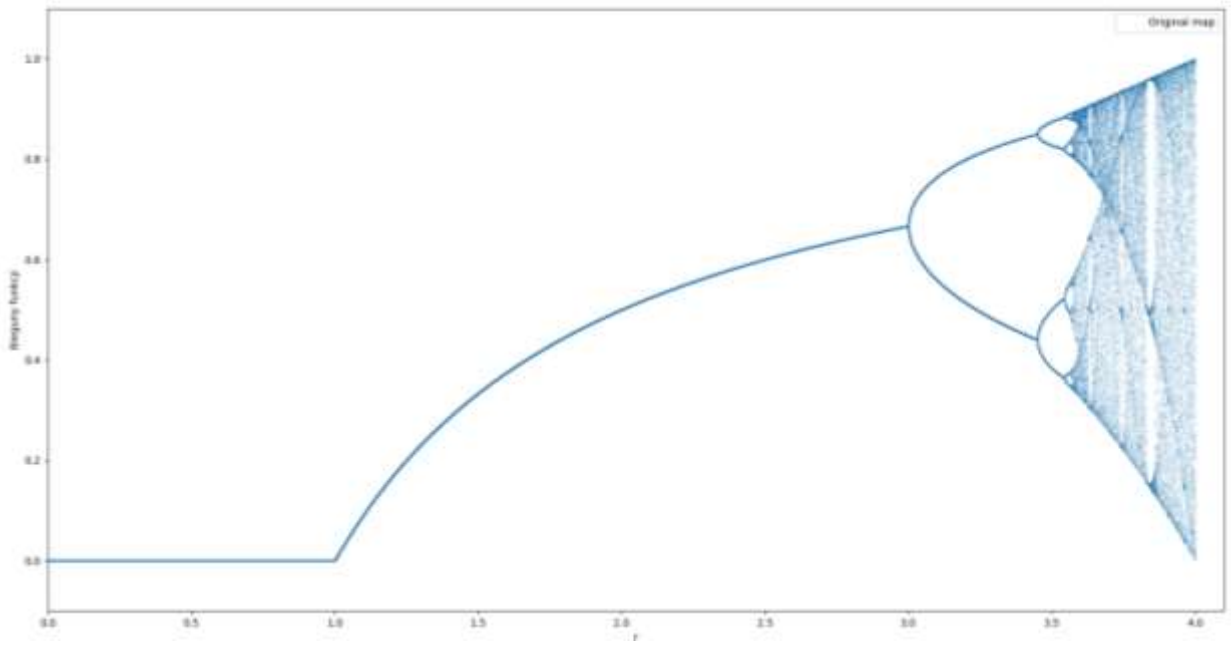
Długość badanego sygnału	Średni czas wykonywania algorytmu FROLS [s]	Średni czas wykonywania zmodyfikowanego algorytmu FROLS [s]
1000	$0{,}015 \pm 0{,}005$	$85{,}012 \pm 3{,}475$
500	$0{,}013 \pm 0{,}004$	$47{,}027 \pm 2{,}930$
200	$0{,}012 \pm 0{,}004$	$24{,}306 \pm 2{,}955$

4.2 Rekonstrukcja funkcji logistycznej

Funkcja logistyczna jest przykładem układu, który w zależności od wartości parametru r znajdującego się w jego formule może wykazywać zachowania chaotyczne. Wzór określający funkcję logistyczną ma postać:

$$y(k) = ry(k-1) - ry^2(k-1)$$

W zależności od wartości parametru r, zmienia się liczba asymptot do których dążą wartości funkcji. Zbiorczo zachowanie tego typu można przedstawić za pomocą wykresu bifurkacyjnego, przedstawionego na rysunku 22. Na osi pionowej przedstawione są wartości asymptot do których dąży sygnał, natomiast na osi poziomej przedstawione są wartości współczynnika r. Dla $r \in (1,3)$, wartości funkcji dążą do jednej wartości. Następnie dla rosnącej wartości współczynnika wartości funkcji oscylują pomiędzy dwoma, następnie czterema biegunami, aby dalej osiągnąć stan chaosu deterministycznego. Oznacza to, że funkcja dąży do bardzo dużej liczby biegunów, co sprawia wrażenie losowości, jest jednak procesem deterministycznym.



Rysunek 22. Wykres bifurkacyjny funkcji logistycznej.

W wyniku modelowania funkcji logistycznej, otrzymywany model najczęściej posiada formułę:

$$y(k) = r_1 y(k-1) - r_2 y^2(k-1)$$

gdzie: parametry $r_1 \neq r_2$. Różnice w wartościach parametrów są wynikiem szumu obecnego w sygnale, błędów wynikających z architektury systemu komputerowego oraz algorytmu modelowania. Istnieje możliwość odtworzenia wykresu bifurkacyjnego na podstawie wyznaczonego modelu pomimo nierówności w jego parametrach. Wzór z równania () można zapisać jako:

$$y(k) = \beta[\gamma - y(k-1)]y(k-1)$$

gdzie: $\beta \sim r \sim r_1 \sim r_2, \gamma \sim 1$. Wtedy, wykreślenie mapy logistycznej można przeprowadzić wobec zmiennej β, natomiast stały współczynnik γ bliski jedności oznacza stopień zgodności z prawdziwą formułą funkcji logistycznej.

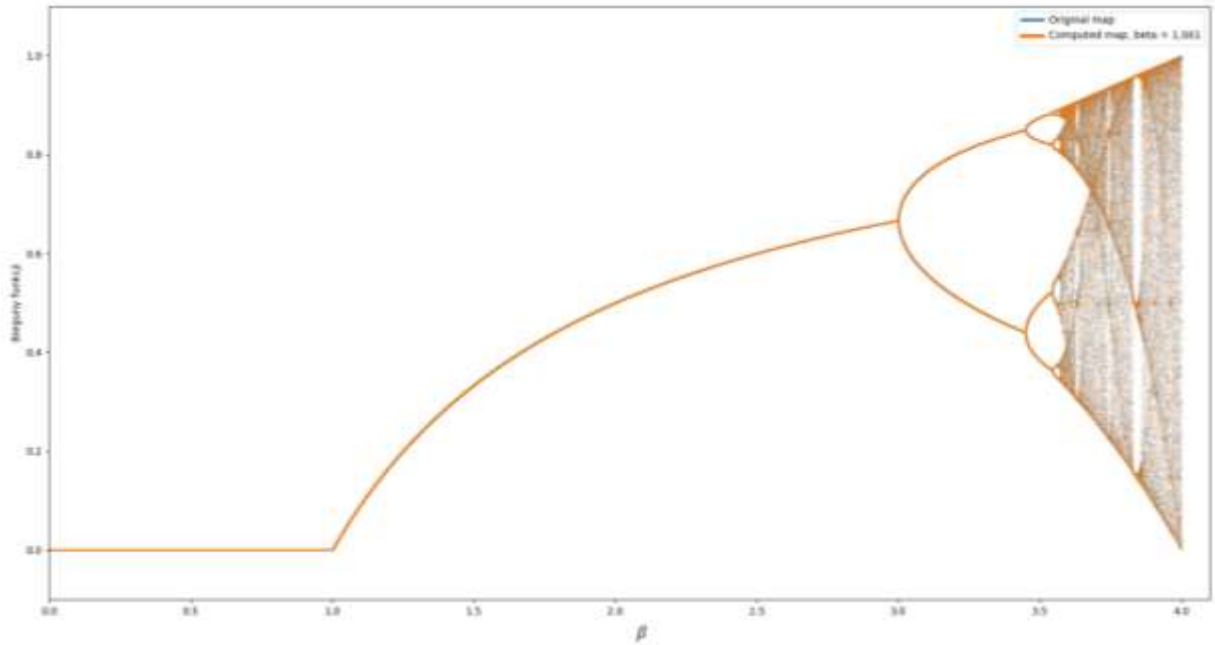
Za pomocą powyższej metody wykreślono dwa wykresy bifurkacyjne dla wybranych modeli z tabeli 8:

- Najlepiej dopasowany model poprawny – wynik oznaczony na zielono o najniższym δ=0,07% (700 próbek, wariancja szumu równa 0,025), wynik na rysunku 23,

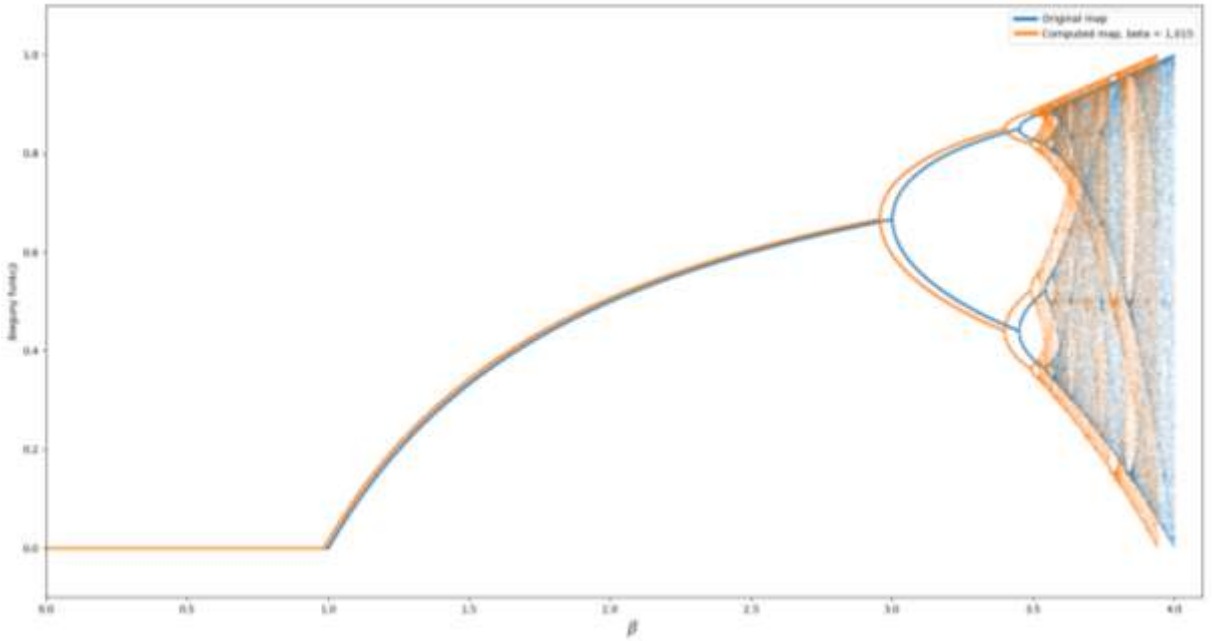
$$y_1(k) = 3{,}4987(k-1) - 3{,}4967y^2(k-1) = 3{,}4967[1{,}001 - y(k-1)]y(k-1)$$
$$= \beta_1[1{,}001 - y(k-1)]y(k-1)$$

- Najgorzej dopasowany model poprawny – wynik oznaczony na zielono o najwyższym δ=2,70% (200 próbek, wariancja szumu równa 0,05), wynik na rysunku 24,

$$y_2(k) = 3{,}4306y(k-1) - 3{,}3802y^2(k-1) = 3{,}3802[1{,}015 - y(k-1)]y(k-1)$$
$$= \beta_2[1{,}015 - y(k-1)]y(k-1)$$



Rysunek 23. Wykres bifurkacyjny dla funkcji logistycznej wraz z wykresem otrzymanym na podstawie modelu o δ=0,07% (700 próbek, wariancja szumu równa 0,025), γ=1,001.



Rysunek 24. Wykres bifurkacyjny dla funkcji logistycznej wraz z wykresem otrzymanym na podstawie modelu o δ=2,70% (200 próbek, wariancja szumu równa 0,05), γ=1,015.

5. Analiza wyników

Na podstawie rysunków 19., 20. i 21. oraz tabel 5., 6. i 8., można ocenić wpływ długości modelowanego sygnału, wariancji dodanego szumu oraz stopnia skomplikowania modelu na wynik modelowania.

Badane sygnały pochodziły od modeli dwu-, trzy- oraz czteroczłonowych. Wraz ze wzrostem skomplikowania systemu wzrastała liczba błędnych przypadków modelowania. Z drugiej strony, liczba sytuacji niepewnych była wyższa dla modelowania systemu trzyelementowego niż systemu czteroelementowego. Niewątpliwie, ma to wpływ na wyniki modelowania. W celu wyciągnięcia konkretnych wniosków na temat wpływu tego warunku na wynik modelowania, wymagane jest przeprowadzenie dalszych badań w tym temacie.

W przypadku wszystkich otrzymanych wyników, wzrost wariancja dodanego szumu powodowała większą rozbieżność pomiędzy otrzymanym wynikiem a rzeczywistymi parametrami modelu. Jest to zrozumiałe, gdyż szum o wyższej wariancji wprowadza większe zaburzenia do sygnału. Dodatkowo, wszystkie badane sygnały posiadają człon AR, co sprawia, że szum ma wpływ nie tylko na wartość sygnału w danej chwili, ale także na przyszłe wartości sygnału. Dla wszystkich typów modeli można wykreślić granicę wariancji szumu, dla której większość modelowań przebiega poprawnie. W przypadku sygnałów liniowego oraz nieliniowego, szum o wariancji wyższej lub równej 0,4 warunkował pełne lub częściowe niepowodzenie modelowania. Dla funkcji logistycznej wartość ta wynosi 0,04. W przypadku modelowań poprawnych, szum o wyższej wariancji wpływał na wzrost różnic pomiędzy otrzymanym wynikiem a rzeczywistym modelem. Jednak dla poprawnych wyników różnica ta zawierała się w przedziale (0,01; 5,35)%, zwykle będąc niższą niż 1%. W przypadku sytuacji niepewnych, różnice te zawierały się w przedziale (0,01; 12,49)%, zwykle osiągając wartości niższe niż 3%.

Wpływ długości badanego sygnału na wynik modelowania był odwrotny niż wpływ wariancji dodanego szumu. Wzrost długości sygnału skutkował mniejszymi rozbieżnościami pomiędzy otrzymanym wynikiem a modelem rzeczywistym. W przypadku sygnałów: liniowego oraz nieliniowego, można wskazać minimalną długość sygnału, która skutkuje znacznie lepszym wynikiem modelowania. Jest to zgodne z tezą przedstawioną przez autorów metody [6], mówiącą o minimalnych 300-500 próbkach sygnału w celu wykonania poprawnego modelowania. Dla sygnałów o długości 500 próbek lub dłuższych, jeśli stosowano metodę zmodyfikowaną, nie otrzymano błędnego wyniku.

Można także zaobserwować, że negatywny wpływ wariancji szumu na wynik modelowania jest większy niż pozytywny wpływ zwiększania długości sygnału. Pokazuje to tabela 5., gdzie widoczna jest granica pomiędzy wynikami poprawnymi oraz niepewnymi. Wzrost wariancji szumu wprowadzał znaczne zaburzenia do sygnału, których skompensowanie przez wydłużenie sygnału modelowanego nie było wystarczające, by przeprowadzić poprawne modelowanie. W sytuacji, gdyby te wpływy były równoważne, granica ta byłaby zarazem przekątną tabeli. W tym przypadku, granica jest przesunięta w stronę niższych wartości wariancji szumu, co oznacza jego wyższy wpływ na wynik modelowania.

Analizując otrzymane wykresy bifurkacyjne można zauważyć, że związane są one z jakością otrzymanego wyniku. Dla modeli niewiele różniących się od oryginału, wyznaczony wykres bifurkacyjny praktycznie nakłada się na wykres analityczny. W przypadku modelu poprawnego który najbardziej odbiegał od modelowanej funkcji, wykres bifurkacyjny zachował swój

przebieg, został jednak nieznacznie przesunięty w stronę niższych wartości parametru β. Największe różnice można zaobserwować dla $\beta > 3$, gdzie przesunięcie wykresu wynosi ok 0,1, co stanowi ok. 3,33% wartości parametru. Dla wartości niższych, krzywe się pokrywają i dla przegięcia w punkcie $\beta = 1$ przesunięcie jest nieznaczne.

Ważnym parametrem jest także czas wykonywania jednego kroku modelowania przez program w zależności od wybranej wersji algorytmu, zawarty w tabeli 9. Czas wykonywania kroku w przypadku algorytmu zmodyfikowanego jest równy kilkunastu milisekundom niezależnie od długości sygnału. Dla algorytmu zmodyfikowanego, czas wykonywania jest dłuższy nawet 5667 razy w ramach jednego kroku. Wobec tego, dopasowanie pięciu elementów do modelu w przypadku metody zmodyfikowanej zajmie od $(121{,}530 \pm 14{,}775)s$ do $(425{,}060 \pm 17{,}375)s$, w zależności od długości danych.

6. Wnioski

Wyniki przedstawione w rozdziale 4, pozwalają stwierdzić, że modyfikacja metody w znacznym stopniu usprawniła jej działanie. W przypadku modelowania sygnału liniowego oraz funkcji logistycznej, wykorzystanie niezmodyfikowanego algorytmu FROLS nie prowadziło do otrzymania poprawnego modelu. Mogło to wynikać z faktu, że model składał się z członów o równoważnych parametrach, zatem kontrybucja każdego z nich do sygnału wyjściowego była podobna. Skutkowało to zbliżonymi wartościami *ERR*, które dopiero w sumie okazywały się posiadać znaczącą wartość. Każde z nich pojedynczo posiadało niższy ERR niż inne regresory, pomimo iż nie były one składnikami modelu. Wybór innego regresora jako pierwszego prowadził wobec tego do wyznaczenia błędnego modelu ortogonalnego, który następnie prowadził do porażki całego procesu modelowania. Niezmodyfikowany algorytm FROLS w sytuacji sprzyjającej, to znaczy gdy parametry modelu znacznie różnią się od siebie, daje identyczne wyniki jak jego zmodyfikowana wersja. Wobec tego, zastosowanie zmodyfikowanego algorytmu jest bardziej uniwersalne.

Ponadto, długość badanego sygnału oraz wariancja dodanego szumu mają istotny wpływ na wynik modelowania. W celu uzyskania najlepszych wyników, konieczne jest zadbanie o wysoką jakość zbierania próbek badanego sygnału w celu wyeliminowania negatywnego wpływu szumu na otrzymany wynik. Należy jednocześnie pamiętać, że filtracja sygnału po jego rejestracji jest niewskazana, ponieważ zaburza ona poszukiwany model [6]. Nakłada to wysokie wymagania co do jakości systemu zbierającego sygnał, który ma zostać poddany modelowaniu. Wskazane jest także zbieranie jak najdłuższych przebiegów, co pozwoli skompensować negatywny wpływ szumu na badany sygnał. Dodatkowo, na podstawie otrzymanych wyników zaleca się modelowanie sygnałów o długości co najmniej 500 próbek.

Jakość otrzymanego modelu może zostać także oceniona na podstawie zgodności otrzymanego wykresu bifurkacyjnego modelowanej funkcji logistycznej z wykresem analitycznym. Ocena ta pozwala na wprowadzenie dodatkowego kryterium klasyfikacji modeli poprawnych, jednak dotyczy jedynie funkcji logistycznej. Dodatkowo, wyznaczenie wykresu bifurkacyjnego zgodnego z wykresem analitycznym świadczy pozytywnie o zmodyfikowanym algorytmie FROLS oraz pokazuje jej potencjalne możliwości w analizie sygnałów pochodzących od nieznanych systemów.

Należy wziąć także pod uwagę, że długi sygnał będzie zwiększał czas modelowania, co pokazały wyniki badań. Modyfikacja wprowadzona do algorytmu wydłuża czas wykonywania jednego kroku o cztery rzędy wielkości. Jeśli czas modelowania odgrywa rolę w danym przypadku, wskazane jest wykorzystanie metody niezmodyfikowanej, pamiętając o jej ograniczeniach. Możliwym scenariuszem wydaje się początkowe poszukiwanie modelu za pomocą metody zmodyfikowanej, aby następnie śledzić jego zmiany za pomocą metody niezmodyfikowanej, która jest szybsza. Otrzymany czas nie stanowi jednak znacznej przeszkody w modelowaniu i pozwala w czasie kilkudziesięciu minut doprowadzić do otrzymania wyniku.

7. Dyskusja

Opracowany program spełnia przyjęte założenia i realizuje metodę, która w większości przypadkach poprawnie wyznaczyła parametry modelowanego sygnału. Jednakże, istnieje wiele innych algorytmów zgodnych z filozofią NARMAX, które mógłby lepiej spełnić to zadanie. Przykładem jest metoda MFROLS [6], która dopasowuje model do wielu sygnałów pochodzących z tego samego systemu. Dodatkowo, istnieją algorytmy identyfikacji źródeł szumu oraz ich wpływu na sygnał [6], które także mogą zostać wykorzystane w celu przeprowadzenia pełniejszego modelowania. Interesujące byłoby też zbadanie wpływu stopnia skomplikowania modelu na wynik modelowania oraz przeprowadzenia modelowania dla sygnałów rzeczywistych. Co więcej, pozytywnie na jakość wyników modelowania wpłynęłyby dodatkowe metody walidacji modelu oraz określenia końca modelowania.

Wykonane oprogramowanie mogłoby zostać zrealizowanie za pomocą struktury wielowątkowej, co skróciłoby czas obliczeń. Inną opcją jest implementacja metody na procesorze graficznym za pomocą środowiska *CUDA*, co potencjalnie mogłoby nawet zniwelować różnice pomiędzy dwoma wariantami algorytmu.

Odnotowania wymaga także fakt, iż w przypadku sygnału liniowego, modelowanie metodą nieliniową doprowadziło do otrzymania modelu liniowego. Jest to zgodne z filozofią NARMAX oraz zapewnia swojego rodzaju bufor bezpieczeństwa. Gdy modelowany jest sygnał pochodzący z systemu o nieznanej naturze, istnieje prawdopodobieństwo, że szukany model jest liniowy. Zastosowanie metody NARMAX, pomimo iż jest ona dedykowana poszukiwaniu modeli nieliniowych, jest w stanie także pełnić rolę narzędzia do modelowania liniowego w metodach ARMAX.

Wobec przedstawionych wyników wykorzystanie metody NARMAX do analizy sygnałów biomedycznych wydaje się posiadać pewne zalety. Korzystając ze zmodyfikowanego algorytmu FROLS, istnieje możliwość opracowania modelu sygnału pochodzącego od osoby zdrowej. Model taki, mógłby służyć jako potencjalny klasyfikator stanu zdrowia. Klasyfikacja mogłaby opierać się o porównanie modelu otrzymanego na podstawie zarejestrowanego sygnału biomedycznego z ogólnym modelem określającym przebieg pochodzący od osoby zdrowej. Jednak działanie tego typu powinno zostać poparte większą ilością badań w celu potwierdzenia niniejszej hipotezy.

Wykonana autorska modyfikacja została sprawdzona dla systemów znanych oraz posiadających maksymalnie cztery człony oraz niewielkie opóźnienie. Efektywność metody oraz czas obliczeń, należy przetestować dla modeli bardziej skomplikowanych. Możliwe, że wymagane będzie wprowadzenie dalszych modyfikacji w celu utrzymania aktualnej efektywności algorytmu. Jednakże jeśli istnieje przesłanka, że badany system jest nieliniowy w zakresie zbadanym w niniejszej pracy, opisana metoda może okazać się skutecznym rozwiązaniem prowadzącym do otrzymania poprawnego wyniku. Szersze jej zastosowanie wymaga jednak większej ilości badań w bardziej różnorodnych warunkach.

8. Bibliografia

[1] G. Mzyk, PARAMETRYCZNA IDENTYFIKACJA SYSTEMÓW O ZŁOŻONEJ STRUKTURZE (rozprawa doktorska)., Wrocław: Politechnika Wrocławska, 2002.

[2] M. A. Aboamer, A. T. Azar, K. Wahba i A. S. A. Mohamed, „Linear model-based estimation of blood pressure and cardiac output for Normal and Paranoid cases.," *Neural Computing and Applications,* tom 25, nr 6, pp. 1223 -1240, 2014.

[3] S. A. Billings i B. Zhang, „Volterra series truncation and kernel estimation of nonlinear systems in the frequency domain," *Mechanical Systems and Signal Processing,* tom 84 (2017), pp. 39-57, 2016.

[4] S. d. Silva, „Non-linear model updating of a three-dimensional portal frame based on Wiener series.," *International Journal of Non-Linear Mechanics,* tom 46 (2011), pp. 312-320, 2010.

[5] S. A. Billings, M. J. Korenberg i S. Chen, „Identification of non-linear output affine sustems using an orthogonal least-squares algorithm.," *International Journal of Systems Science,* tom 19(8), pp. 1559-1568, 1988.

[6] S. A. Billings, Nonlinear System Identification: NARMAX Methods in the Time, Frequency, and Spatio-Temporal Domains., Sheffield: John Wiley & Sons, 2013.

[7] S. Chen, S. A. Billings i W. Luo, „Orthogonal least squares methods and their application to non-linear system identification.," *Internal Journal of Control,* tom 50(5), pp. 1873-1896, 1989.

[8] „Dokumentacja Środowiska Python 3.6," 06 2017. [Online]. Available: https://docs.python.org/3/.

[9] „Strona producenta oprogramowania PyCharm - firma JetBrains," 06 2017. [Online]. Available: https://www.jetbrains.com/pycharm/. [Data uzyskania dostępu: 06 2017].

[10] „Dokumentacja biblioteki PyQt5," 06 2017. [Online]. Available: https://pypi.python.org/pypi/PyQt5/5.8.2.

9. Wykaz symboli i skrótów

$y(k)$ – wartość modelowanego sygnału w chwili czasu k
$x(k), u(k)$ – wartość sygnału wejść zewnętrznych w chwili czasu k
$e(k)$ – wartość szumu w chwili czasu k
$F[]$ – nieznana funkcja
$SISO$ – *ang. Single Input Single Output*, system posiadający jedno wejście oraz jedno wyjście
$h_n(\tau_1, \dots, \tau_n)$ – parametr w seriach Volterry
$a_n(i_1, \dots, i_n)$ - współczynnik w szeregu Weinera
$\psi_{i,j}$ – bazy ortogonalne szeregu Weinera
AR – człon autoregresyjny, *ang. Auto-Regressive*, oznaczający wpływ poprzednich wartości sygnału na jego wartość w danej chwili k
MA – człon średniej ruchomej, *ang. Moving Average,* oznaczający wpływ poprzednich wartości szumu na wartość sygnału w danej chwili k
X – człon zewnętrznych sygnałów wejściowych, *ang. eXogenous input*, oznaczający wpływ zewnętrznych sygnałów na wartość sygnału wyjściowego w danej chwili k
$ARMAX$ – model liniowy zawierający człon autoregresyjny, człon średniej ruchomej oraz wejście zewnętrzne
$NARMAX$ – model nieliniowy zawierający człon autoregresyjny, człon średniej ruchomej oraz wejście zewnętrzne
OLS – *ang. Orthogonal Least Squares*, estymator ortogonalnej sumy najmniejszych kwadratów
$FROLS$ – *ang. Forward Regression with Orthogonal Least Squares estimator,* algorytm regresji w przód z wykorzystaniem estymatora ortogonalnej sumy najmniejszych kwadratów
$p(k)$ – wartość danego regresora w chwili czasu k
$w(k)$ – wartość podstawowego sygnału używanego do konstruowania regresorów, w chwili k
$q(k)$ – wartość członu modelu ortogonalnego w chwili czasu k
g – wartość parametru członu modelu ortogonalnego
θ– wartość parametru członu modelu szukanego
$\boldsymbol{p}$ – wektor próbek danego regresora
$\boldsymbol{w}$ – wektor próbek podstawowego sygnału używanego do konstruowania regresorów
$\boldsymbol{q}$ – wektor próbek członu modelu ortogonalnego
$\boldsymbol{y}$ – wektor próbek modelowanego sygnału
$\boldsymbol{x}, \boldsymbol{u}$– wektor próbek sygnału wejść zewnętrznych
$\boldsymbol{e}$ – wektor próbek szumu
M – liczba potencjalnych regresorów
D – wektor regresorów
σ – energia (wariancja) sygnału
σ – energia (wariancja) sygnału
σ – energia (wariancja) sygnału
ERR – *ang. Error Reduction Ratio*, stopień redukcji błędu, współczynnik oznaczający jak duży wpływ na redukcję błędu miał dany regresor
$\boldsymbol{err}$ – wektor zawierający wartości ERR dla wybranych regresorów
ESR – *ang. Error to Signal Ratio*, stosunek błędów do sygnał
$SERR$ –suma ERR dla danego modelu
l – indeks wybranego regresora w danym kroku
a – parametr ortogonalizacyjny
A – macierz parametrów ortogonalizacyjnych
r – stała funkcji logistycznej
s – aktualny numer kroku w algorytmie FROLS
k – aktualna chwila czasu dyskretnego
n, i, j – liczniki
$\boldsymbol{c}^T$ – transpozycja wektora c
$max(\boldsymbol{c})$ – maksymalna wartość wektora c

10. Spis rysunków

11. Spis tabel

Załącznik A – Kod źródłowy programu