XXII Seminarium

ZASTOSOWANIE KOMPUTERÓW W NAUCE I TECHNICE' 2012
Oddział Gdański PTETiS
Referat nr 27

BUDOWA MODELU PROGNOSTYCZNEGO FARMY WIATROWEJ W ŚRODOWISKU MATLAB

Tomasz RUBANOWICZ

Politechnika Gdańska, ul. G. Narutowicza 11/12, 80-952 Gdańsk
E-mail.: truban@ely.pg.gda.pl

Streszczenie: Liberalizacja rynku energii elektrycznej sprawiła, że branża elektroenergetyczna przechodzi obecnie dynamiczny rozwój różnych jej obszarów (aspektów). Jednym z aspektów jest prognozowanie mocy jednostek wytwórczych źródeł wiatrowych. W prognozowaniu wykorzystuje się różnego rodzaju narzędzia matematyczne. Autor niniejszej publikacji poświęcił szczególną uwagę sztucznym sieciom neuronowym. Za pomocą modeli neuronowych istnieje możliwość predykcji generacji mocy wytwórczej farm wiatrowych.
Budowa modelu prognostycznego wymaga umiejętności programistycznych. Powszechnym środowiskiem programistycznym, umożliwiającym budowę modeli, jest oprogramowanie naukowo-techniczne MATLAB. Wykorzystując wbudowane funkcje (gotowe moduły) istnieje możliwość zbudowania modelu prognostycznego farmy wiatrowej. W artykule przedstawiono sposób zamodelowania wybranej struktury neuronowej za pomocą modułu Neural Toolbox oraz przeprowadzono test nauczonej sieci.

Słowa kluczowe: elektrownie wiatrowe, prognozowanie mocy, sztuczne sieci neuronowe, MATLAB.

1. WSTĘP

Elektrownie wiatrowe, ze względu na zmienne warunki wiatrowe, uważane są za niespokojne źródła wytwórcze, które wymagają stałego planowania generacji mocy w dniu poprzedzającym w horyzoncie dobowo-godzinowym. Planowanie generacji mocy, ma wpływ na bezpieczeństwo pracy krajowego systemu elektroenergetycznego (KSE) i handel energią elektryczną na rynku. Ze względu na coraz większą moc znamionową elektrowni wiatrowych, przyłączoną do KSE, predykcja mocy odgrywa coraz bardziej istotną rolę.

W ostatnim czasie można zaobserwować dynamiczny rozwój branży elektroenergetycznej w różnych obszarach (aspektach). Zjawisko rozwoju zawdzięczyć można trwającemu procesowi liberalizacji rynku energii elektrycznej. Innowacyjne rozwiązania mają przyczynić się do poprawy pozycji podmiotów na konkurencyjnym rynku. Na rynku dostępnych jest wiele różnych narzędzi do predykcji mocy źródeł wiatrowych, lecz o niezadowalającym błędzie prognozy, o których mowa w pozycjach [1÷4]. W prognozowaniu coraz bardziej popularne stają się modele hybrydowe [5÷7]. Autor niniejszej publikacji poświęcił szczególną uwagę sztucznym sieciom neuronowym (SSN). Za pomocą modeli neuronowych istnieje możliwość predykcji generacji mocy wytwórczej dowolnej farmy wiatrowej.

Budowa modelu prognostycznego wymaga podstawowych umiejętności programistycznych. Wykorzystując wbudowane funkcje (gotowe moduły) istnieje możliwość zbudowania dowolnej struktury modelu prognostycznego farmy wiatrowej.

2. NORMALIZACJA DANYCH

Budowę modelu prognostycznego rozpoczęto od identyfikacji problemu badanego zjawiska, w celu ustalenia cech charakterystycznych obiektu (zależności liniowej lub nieliniowej). W analizie brano pod uwagę dwie główne zmienne mające największy wpływ na pracę farmy wiatrowej, tj. prędkość i kierunek wiatru (jako warunki zewnętrzne, oznaczone zmienną *P*) oraz generację mocy (jako warunki wewnętrzne, oznaczone zmienną *T*). Do analizy wykorzystano rzeczywiste dane meteorologiczne i produkcyjne farmy, zlokalizowanej w północnej części kraju. Badana próbka danych dotyczyła okresu zimowego, jednego tygodnia (licząca w sumie 18144 pomiarów 10-cio minutowych). Przy budowie sieci dane podzielono na dwie części: próbę uczącą (90%) i próbę testującą (10%). Po zakończeniu wstępnej analizy danych przygotowano dane do nauki SSN. Przygotowanie danych polegało m.in. na normalizacji danych wejściowych, sprowadzając je do wartości z przedziału <-1,1>. Do tego celu użyto w Matlabie następującego polecenia o treści, tj.: *premnmx(P,T).*

3. BUDOWA MODELU FARMY WIATROWEJ

Na potrzeby niniejszego artykułu wybrano jedną strukturę sieci - sieć rekurencyjną Elmana. Strukturę sieci Elmana utworzono za pomocą polecenia, tj.: *newelm(Lw,Ln,fw,fus,fuw,fb),* gdzie: *newelm* – tworzy sieć Elmana, *Lw* – macierz określa liczbę i zakres wartości wejść sieci, ,*Ln* – liczba neuronów w i-tej warstwie oraz liczba warstw sieci, *fw* - funkcja aktywacji neuronów w i-tej warstwie, *fus* – funkcja ucząca sieć, ,*fuw* – funkcja ucząca wagi, ,*fb* – funkcja błędu sieci.
SSN poddano procesowi nauki (danego problemu), który polega na osiągnięciu zbieżności przez estymowaną moc. W celu osiągnięcia zbieżności brano pod uwagę kilka parametrów, tj. liczbę neuronów w poszczególnych warstwach sieci, liczbę wykonanych iteracji (powtórzeń),

Recenzent: Prof. dr hab. inż. Zbigniew Lubośny –Wydział Elektrotechniki i Automatyki Politechnika Gdańska

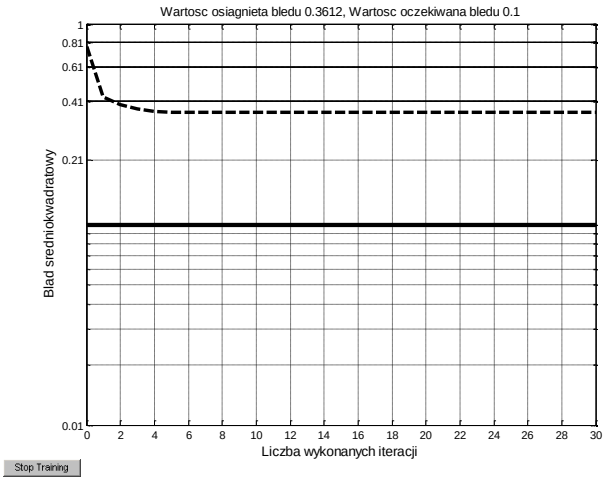
dopuszczalny błąd sieci, krok uczenia, funkcję aktywacji oraz czas jej uczenia [8].

W pierwszej kolejności założono, że sieć Elmana będzie składała się z następującej liczby neuronów i warstw *Ln*:

Ln1 = 5 (liczba neuronów w pierwszej warstwie),
Ln2 = 15 (liczba neuronów w drugiej warstwie),
Ln3 = 1 (liczba neuronów w trzeciej warstwie).

Następnie dobrano liczbę neuronów, metodą doświadczalną (obserwacyjną). Konsekwencją błędnie dobranej liczby neuronów było to, że sieć podczas nauki nie osiągnęła zbieżności w kierunku rzeczywistej mocy.

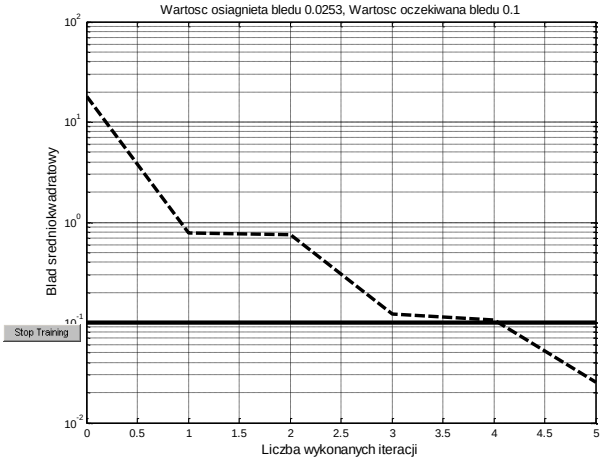
Następnie tą samą metodą ustalono liczbę iteracji uczenia sieci (max.30 iteracji). Posłużono się w tym celu poleceniem, tj. *net.trainParam.epochs*. Zauważono, że przy zbyt małej liczbie iteracji sieć nie nadąża z nauką danego problemu. Natomiast zbyt duża liczba iteracji doprowadziła do znaczącego wydłużenia czasu uczenia się sieci i zwiększyła ryzyko jej przeuczenia. Na rys.1 przedstawiono proces nauki sieci, na którym widać, że powyżej drugiej wykonanej iteracji, sieć nie potrafi się już lepiej nauczyć analizowanego przypadku, a dalsza nauka nie przynosi poprawy rezultatów. Po osiągnięciu maksymalnej liczby iteracji algorytm uczenia sieci został przerwany.



Rys.1 Proces nauki sieci bez rezultatu

Kolejnym ważnym parametrem było określenie dopuszczalnej wielkości błędu uczenia sieci, czyli jakości dopasowania sieci do danych empirycznych. Parametr dopuszczalnego błędu wyznaczono za pomocą polecenia tj. *net.trainParam.goal*. Do tego celu wybrano średni błąd kwadratowy (ang. Mean Squared Error, skrót MSE), najczęściej stosowaną miarę badania jakości sieci. Błąd wyliczono za pomocą obiektowej funkcji MSE. Na rys.1 widać, że sieć nie osiągnęła oczekiwanej wartości błędu (na poziomie 0.1).

Dalszy proces nauki sieci, przy jednoczesnym zwiększaniu liczby iteracji, nie umożliwił osiągnięcia zbieżności estymowanej mocy. W przypadku dobrze dobranych parametrów (omówionych powyżej, m.in. liczby neuronów, czy opisanej w dalszej części publikacji funkcji aktywacji sieci), szybko osiągnięto jej zbieżność, co przedstawiono graficznie na rys.2.



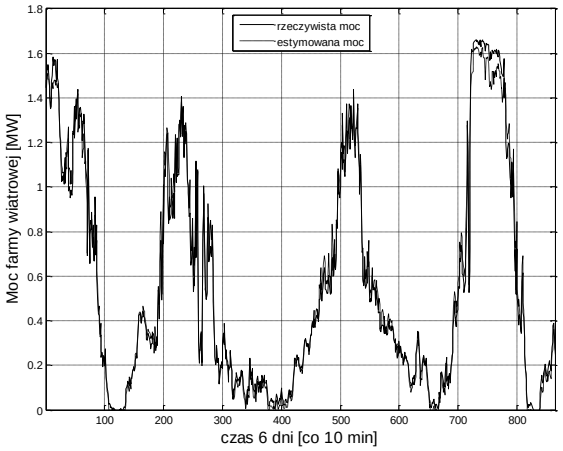
Rys.2 Dobrze dobrane parametry sieci pozwalają jej szybko osiągnąć oczekiwaną zbieżność

Kolejnym elementem w procesie budowy modelu było ustalenie kroku uczenia sieci. Krok uczenia wyznaczono za pomocą polecenia, tj.: *net.trainParam.lr*. Element ten odpowiada za wielkość zmian wag neuronów w każdym kolejnym kroku uczenia. Zbyt duży krok spowoduje problemy z uzyskaniem zbieżności ku rzeczywistej mocy. Zbyt mały krok spowoduje dłuższy czas nauki bez rezultatu.

Podczas badania obszaru zmienności analizowanego szeregu określono funkcję aktywacji. Sieć Elmana składa się z trzech warstw. Pierwsza warstwa dotyczy parametrów wejściowych, druga warstwa jest ukrytą, w której zachodzą wszelkie procesy uczenia, a trzecia zawiera oczekiwane rezultaty procesu uczenia. Analizując problem nieliniowy, w pierwszej warstwie wybrano funkcję aktywacji: *sigmoidalną* (logsig) ze względu na występujące wartości dodatnie. W warstwie ukrytej zastosowano funkcję tangens hiperboliczny, ze względu na wartości ujemne i dodatnie (wartości znormalizowane w przedziale wskazanym powyżej). Natomiast w ostatniej warstwie użyto funkcji liniowej. Wybór właściwych funkcji aktywacji umożliwił osiągnięcie w krótkim czasie przybliżonej zbieżności estymowanej mocy oraz lepszą zdolność generalizacji sieci, odporną na błędy pomiarowe.

Jednym z ostatnich ważnych parametrów sieci był czas jej uczenia. Wartości tego parametru zależą od pozostałych (już wymienionych) czynników, np. od liczby danych wejściowych i neuronów, (które powodują, że czas uczenia sieci rośnie wykładniczo), czy też liczby iteracji uczenia [7]. Czas uczenia sieci wyznaczono za pomocą polecenia, tj.: *net.trainParam.time*. Proces uczenia sieci uruchomiono za pomocą polecenia, tj.: *train(net,P,T)*.

Celem budowy modelu neuronowego było osiągnięcie jak najlepszej estymacji mocy na podstawie empirycznej wartości generacji mocy. Zbudowanie takiego modelu nie daje gwarancji najlepszej zdolności generalizacji. Na rys.3 przedstawiono jakość nauczonej sieci. Jak widać na tym rysunku sieć nauczyła się danego problemu, lecz nie w pełni idealnie, co może oznaczać, że zachowała zdolności generalizacyjne.



Rysunek 3Przykład dobrej estyma ty mocy nauczonej sieci

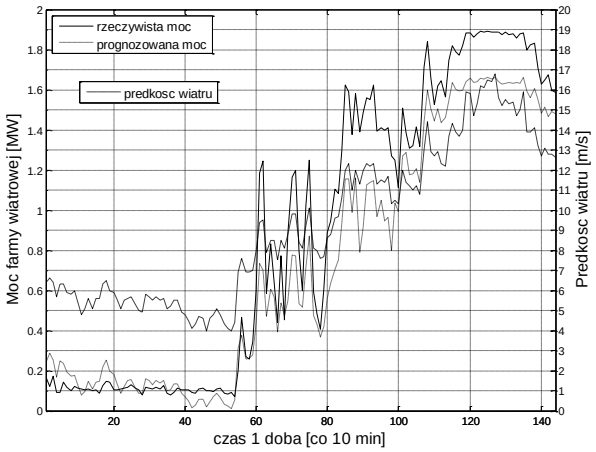
4. TESTOWANIE SIECI

Ostatnim etapem projektowania sieci jest jej testowanie (symulowanie). Symulowanie działania sieci można uruchomić za pomocą funkcji, tj.: *sim(net,P)*, gdzie: *sim* – funkcja symulacji pracy sieci, *net* – struktura SSN, *P* – wektor wejściowy do sieci. Symulacja pracy sieci pozwoliła ocenić możliwości wykorzystania SSN na potrzeby predykcji mocy wytwórczej farm wiatrowych.

Na rys.4 zamieszczono przykład jakości prognozy mocy w horyzoncie dwudziestoczterogodzinnym. Jak widać uzyskane wartości prognozowane mocy nie odbiegają zbyt znacząco od wartości rzeczywistych.

Dzięki zbudowanemu modelowi, za pomocą prostych poleceń, można rozpocząć kolejne etapy badań, m.in. w celu sprawdzenia czy sieć nie przeuczyła się.

W celu graficznej prezentacji wyników jakości nauczonej sieci użyto funkcji, tj.: *plot*. Dzięki której sporządzono rys.4.



Rysunek 4 Przykład prognozowanej mocy

5. PODSUMOWANIE

Autor przedstawił sposób zamodelowania wybranej struktury neuronowej za pomocą modułu Neural Toolbox oraz przeprowadził test nauczonej sieci. Analiza testów badania sieci wykazała, że SSN mogą być wykorzystywane w predykcji mocy wytwórczej farm wiatrowych. Niewielki błąd (0,0253) sieci uzyskany w procesie nauki sieci (rys.3) może świadczyć o dobrych jej właściwościach oraz dobrze dobranych parametrach sieci (o których była mowa wcześniej w pkt.3). Znajomość podstawowych poleceń MATLAB-a, umożliwia zbudowanie dowolnej struktury sieci i modelu. Dzięki programowaniu obiektowemu skraca się czas modelowania sieci przez projektanta. W zakresie grafiki MATLAB umożliwia rysowanie dwu i trójwymiarowych wykresów funkcji oraz wizualizację wyników obliczeń w postaci rysunków statycznych i animacji. Możliwe jest pobieranie danych pomiarowych z urządzenia zewnętrznego przez porty w celu ich obróbki [9].

Różnorodność projektów budowlanych farm wiatrowych, pod względem rozmieszczenia i lokalizacji w terenie pojedynczych siłowni wiatrowych, sprawia, że nie można zbudować uniwersalnego modelu prognostycznego dla każdej farmy. Zatem skutecznym i ekonomicznym rozwiązaniem jest zbudowanie kilku różnych modeli, a następnie metodą eksperymentalną, odpowiednie dobranie najlepszego dla wybranej farmy. Naukę programowania w MATLAB-ie warto rozpocząć od zapoznania się z wbudowaną biblioteką Pomocy (ang. help). Pomoc można uzyskać przez wpisanie plecenia *help* w okienku poleceń (ang. Command Window).

6. LITERATURA

1. Mabel M.C., Fernandez E., Analysis of wind power generation and prediction using ANN: A case study, Renewable Energy 33 (2008) 986–992, ScienceDirect, 2007.
2. Rubanowicz T., Metody predykcji produkcji mocy parku wiatrowego, Zeszyty Naukowe Wydziału Elektrotechniki i Automatyki Politechniki Gdańskiej nr 25, Gdańsk 2008.
3. Dobrzyński K., Przegląd systemów przeznaczonych do predykcji mocy wytwarzanej w farmach wiatrowych, Aktualne Problemy w Elektroenergetyce, Jurata, 2009.
4. Lingling Li, Minghui Wang, Fenfen Zhu, and Chengshan Wang, Wind Power Forecasting Based on Time Series and Neural Network, ISCSCT '09, ISBN 978-952-5726-07-7, Huangshan, P. R. China, 26-28,Dec. 2009, pp. 293-297
5. Catalão J.P.S., Pousinho H.M.I., Mendes V.M.F., Short-term wind power forecasting in Portugal by neural networks and wavelet transform, Renewable Energy 36 (2011) 1245e1251, ScienceDirect, 2010.
6. Blonbou R, Very short-term wind power forecasting with neural networks and adaptive Bayesian learning, Renewable Energy 36 (2011) 1118e1124, ScienceDirect, 2010.
7. An X., JiangD., Zhao M., Liu C., Short-term prediction of wind power using EMD and chaotic theory, Commun Nonlinear Sci Numer Simulat 17 (2012) 1036–1042, ScienceDirect, 2011.
8. Materiały edukacyjne Państwowej Wyższej Szkoły Zawodowej w Lesznie, Instytut Politechniczny o kierunku Elektrotechnika, http://elektrotechnika.ip.pwsz.edu.pl/
9. Encyklopedia Wikipedia, stan na dzień 15.10.2012 r., http://pl.wikipedia.org/wiki/MATLAB

CONSTRUCTION OF WIND FARMS FORECASTING MODEL IN MATLAB

Key-words: Wind power plants, prediction methods, artificial neural networks.

The liberalization of the electricity market has made electric power industry is undergoing rapid development its different areas (aspects). One aspect is the forecasting power generating units wind sources. The prediction uses various mathematical tools. The author of this publication has devoted special attention to artificial neural networks. Using neural models can predict power generation manufacturing of wind farms. Construction of a predictive model requires programming skills. A common programming environment that allows the construction of models, is the scientific and technical software MATLAB. Using the built-in (ready modules) it is possible to build a predictive model of the wind farm. This article shows you how to model a neural structure chosen by Neural Toolbox module and performed background and sensor network test.