National Technical University of Athens
School of Naval Architecture and Marine Engineering
Department of Ship Design and Maritime Transport

Diploma Thesis:

Prediction of Resistance of MARAD Systematic Series' Hullforms using Artificial Neural Networks

Vasiliki Margari

Supervisor: George Zaraphonitis

Athens, 2017

Abstract

Maritime Administration (MarAd) in cooperation with Hydronautics Inc., motivated by the rising interest for full hull form merchant ships in the early 1970's and the lack of systematic data regarding the performance of these vessels, developed the so-called MARAD Systematic Series, comprised of 16 full hullforms, specifically designed for use as bulk carriers and tankers. The experimental resistance data are available in a series of diagrams illustrating the residual resistance coefficient as a function of the geometric characteristics of each hull for a range of Froude numbers. The present thesis investigates the potential of Artificial Neural Networks (ANNs) to estimate the residual resistance coefficient, and subsequently the resistance, of hullforms designed according to MARAD Systematic Series, given their main dimensions and block coefficient. The data used for the training of the networks were collected from five diagrams, which are provided by the Series, and illustrate the resistance coefficient for different combinations of their length to breadth and breadth to draft ratios and block coefficient.

To this end, three different types of ANNs have been considered; Multi-layer perceptron (MLP) networks, Radial Basis Function (RBF) networks, and Support Vector Machines (SVM). The performance of a network, regardless its type, is affected by its characteristics, among which its architecture and learning method are of particular importance. Several trials have been conducted for every type of network, changing systematically these characteristics. The developed networks have been thoroughly evaluated, assessing their potential to accurately predict the resistance of MARAD-type hullforms.

A total number of 616 networks were developed; 380 MLPs, 125 RBFs and 111 SVMs. Selected alternatives of these networks, i.e. 4 MLPs, 2 RBFs and 2 SVMs are presented and their potential for the prediction of MARAD hullforms' resistance is discussed. The deviations of the best-performing networks' predictions from the resistance data provided by MARAD were not higher than 1.6% for hullform characteristics inside the limits of the training dataset and 6.8% outside them.

ANNs are quick and effective in estimating the resistance of MARAD hullforms, eliminating the need for searching through the diagrams that provided their training data. The results indicate that their use could be successfully applied also in the case of other Systematic Series. In addition, it might be argued that ANNs could be successfully trained to estimate calm water resistance of selected hullform types, based on the results of systematic calculations using advanced CFD software tools.

Table of contents

Introduction

The prediction of a hull's resistance is a problem of great importance for the ship designer, closely related to the design and optimization of the hullform and propeller, the selection of the main engine, the ship's environmental impact and its fuel cost during the ship's entire life-cycle. Resistance predictions are traditionally based on tank testing, or nowadays, on software tools applying Computational Fluid Dynamics (CFD). However, in the preliminary design stage, resistance predictions are quite often carried out based on relevant data from systematic series, such as the Series 60 Methodical Series of Single-Screw Ships ([1]), the SSPA Cargo Liner Series ([2]), the BSRA Methodical Series ([3]), or the MARAD Systematic Series of full-form ship models ([4], [5]). The data from the systematic series are usually presented in graphical or tabular form, enabling the manual calculation of a ship's resistance following a relatively simple and straightforward procedure. Such a procedure might be perfectly suitable for the estimation of the resistance curve of one particular hullform, but when it comes to the systematic optimization of a ship design, manual calculation procedures are not efficient any more. In such cases programmable calculation procedures would be required, enabling the evaluation of a large number of alternative designs in minimal computation time. Regression techniques, utilizing polynomial interpolation of the data provided by the systematic series have been developed by several authors (see e.g. [6] to [11]). The present work investigates a different approach to this problem, i.e. the use of Artificial Neural Networks (ANNs) as a tool for the prediction of resistance of full hullforms, based on the resistance data provided by the MARAD Systematic Series.

The MARAD Systematic Series is comprised of 16 full hullforms, specifically designed for use as bulk carriers and tankers. The experimental resistance data are available in a series of diagrams. In these figures the residual resistance coefficient (C_R) is given for a range of Froude (Fn) numbers as a function of three geometric parameters; length to breadth ratio (L/B), breadth to draft ratio (B/T) and block coefficient (C_B). Data from these diagrams was collected in the form of points defined by five numbers (i.e. C_R, Fn, L/B, B/T, C_B), and has been used to train and evaluate a series of neural networks aiming to estimate the residual resistance coefficient of ships designed according to the MARAD Series.

Artificial neural networks have recently found applications across many scientific fields, providing an alternative to traditional methods of solving complicated non-linear problems, such as pattern recognition, classification or function approximation. They consist of processors (neurons), which communicate to each other with signals through weighted connections, mimicking the structure of a biological neural system. Trained ANNs are considered able to provide the desired output from a set of input parameters without the need for an exact function or model for the problem, even if the data are noisy. They also offer a number of advantages, including: sufficient accuracy of results, flexibility in implementing, availability of multiple training

algorithms, ability to implicitly detect complex non-linear relationships between independent variables. These features make ANNs rightly suited for application in a wide range of engineering problems. Examples from the use of ANNs for the prediction of resistance, propulsion, maneuvering and seakeeping characteristics of ship are presented in ([12]) to ([19]).

Among the numerous available types of neural networks, three were selected, to be tested in the present thesis. The first type of the examined neural networks was the multi-layer perceptron (MLP), which is the most commonly used. They comprise at least one hidden layer of neurons, between the input and the output layer, which enables them to solve non-linear problems. Additionally, the distribution of their neurons in more than one parallel hidden layers, usually leads to lower numbers of the total neurons used. The weights of the neural connections are defined applying a back-propagation algorithm that transfers the error from the output back to the input layer. Radial Basis Function (RBF) neural networks were also considered for this study. They usually consist of one hidden layer, and although they also execute non-linear transformations and linear weighted sums, they are trained using local transformations instead of the back-propagation algorithm, which is a computationally quicker process ([20]). Support Vector Machines (SVM), on the other hand, use a high principled learning method, that takes into account the structure of the network utilizing a different cost function compared to the other two ANNs and distinguishes between learning tasks. Different types of learning processes are used in order to solve regression or classification problems ([21]).

In order to determine which network fits better to the problem at hand, several trials have been conducted. Taking into account that a network's performance is affected by its characteristics, networks of the three aforementioned types were trained for a large number of combinations of these characteristics.

In total, 616 networks were developed; 380 MLPs, 125 RBFs and 111 SVMs. Selected alternatives of these networks, i.e. 4 MLPs, 2 RBFs and 2 SVMs are presented and their potential for the prediction of MARAD hullforms' resistance is discussed. Having been trained, they were used for estimating the resistance of MARAD-type hullforms and were evaluated accordingly. The best networks were, finally, selected and their estimations are presented in graphical and tabular form.

1. MARAD systematic series

1.1 Systematic series in ship design

Systematic series consist of hulls that share common geometric properties and have been developed in order to assist the naval architect in the design of an efficient hullform, particularly during the initial design stage. In addition, systematic series provide valuable data for the preliminary prediction of calm water resistance and, in some cases, maneuvering characteristics of ships, based on the systematic analysis of extensive tank testing. Systematic series are comprised by a number of hullforms, which are often developed from a parent hull, and usually in a dimensionless form. The hullform of the offsprings derives from the parent hull's, with the systematic variation of appropriate geometric characteristics, within a predefined range. A series of model hulls are constructed and tank tested in order for their resistance and maneuvering properties to be obtained. The collected data are analyzed and illustrated in graphical or tabular form as functions of selected main hull parameters ([22], [23]).

Systematic series' data are available for a large range of ship types, such as merchant ships, tugs, fishing vessels, semi-displacement or planning hulls. Systematic series for single-screw merchant ships are, for example, the Series 60 Methodical Series of Single-Screw Ships ([1]), the SSPA Cargo Liner Series ([2]), the BSRA Methodical Series ([3]), or the MARAD Systematic Series of full-form ship models ([4], [5]).

1.2 Development of MARAD systematic series

MARAD Systematic Series was developed by the U.S. Maritime Administration (MarAd) in cooperation with Hydronautics Inc., aiming to respond to the rising interest for full hull form merchant ships in the early 1970's, given the lack of systematic data regarding the performance of such vessels, that could be used in the design process. As a result, the systematic series is characterized by high block coefficient values (C_B). Low length to breadth ratios (L/B) and high breadth to draft ratios (B/T) were also adopted for economic and operational reasons respectively.

The initial task in the development of the MARAD Systematic Series was the selection of the appropriate parent hull form, from which the other hulls of the series derive. There were two options available regarding this task; either to choose an existing hull of known performance characteristics or to develop a new set of lines. The second one was eventually preferred and as a result research was conducted in order to define the appropriate geometric properties of the parent hull form. Consulting with experts in research facilities in UK, the Netherlands, West Germany, Denmark and Sweden and thorough review of the published literature was carried out. Ships in service, new designs and systematic series with geometric characteristics close to the desired range were taken into account. The considered data included information from the Series 60 Methodical Series of Single-Screw Ships ([1]), the SSPA Cargo Liner Series ([2]), the BSRA Methodical Series ([3]), the systematic series of tankers developed and tested by the Research institution of Japan ([24]) and

the results of the experiments regarding hulls with cylindrical bows that were conducted at the Netherlands Ship Model Basin ([25]).

The results of the analysis of the collected information were used for the development of the geometry of the parent hull and the selection of the range of geometric characteristics of the series. High values of C_B, combined with low values of L/B were observed with the increase in the size of tankers. Values up to 0.875 for the former and down to 4.5 for the latter parameter were expected to be within the range of interest for future ship designs. Restrictions in ship's laden draft would also lead to higher B/T values. As a result, the following range of variation of the series' parameters was selected: C_B between 0.8 and 0.875, L/B values from 4.5 to 6.5 and B/T values from 3 to 4.75. A cylindrical bow shape was chosen over a conventional one, as a means of achieving lower resistance values, according to experimental data regarding full-form hulls for a range of drafts. The ease of adaption of this configuration to different geometries required for the development of the systematic series discouraged the consideration of a bulbous bow. The stern's selection was made taking into account important hydrodynamic issues, such as flow separation and was based on the experience of Hydronautics Inc. regarding the afterbody design of ships with high B/T values. This approach resulted to an ending run with long, flat buttocks and conventional transom stern profile ([4], [5]).

Sets of lines were developed for the parent and the offspring hulls as a combination of three segments; the entrance, the parallel mid-body and two alternative ending runs, a short and a long one, that represent the two extreme values of L/B of the series (i.e. L/B_{long}=6.5 and L/B_{short}=4.5). Offsets for hulls with L/B values in the interim were obtained with interpolation methods. The lines of the three hull segments of the parent hull, also mentioned as Ship A, are presented in Fig. 1 to Fig. 3. Sixteen models were constructed and tank-tested in order to obtain their calm water resistance and maneuvering characteristics. Their longitudinal center of buoyancy (LCB) was placed 2.5% of length forward of amidships, as a result of preliminary resistance tests that were performed on three models. The sixteen model hulls comprising the MARAD series are characterized by their block coefficient (C_B), their breadth to draft ratio (B/T) and length to breadth ratio (L/B) as shown in Fig. 4.

1.3 Ship Resistance

A ship should be designed to travel efficiently through water with minimum resistance. The estimation of a ship's resistance is a task of great importance during the design process. The optimization of the hullform, the selection of the main engine, and the reduction of fuel consumption are problems closely related to the ship's resistance. Requirements for stability, good seakeeping and maneuvering properties also influence the hullform design. The estimation of a ship's resistance may be decomposed in the calculation of two main components, the pressure resistance and the frictional resistance. Each of these two components may be calculated separately, based on the hypothesis that the interactions between them are low.

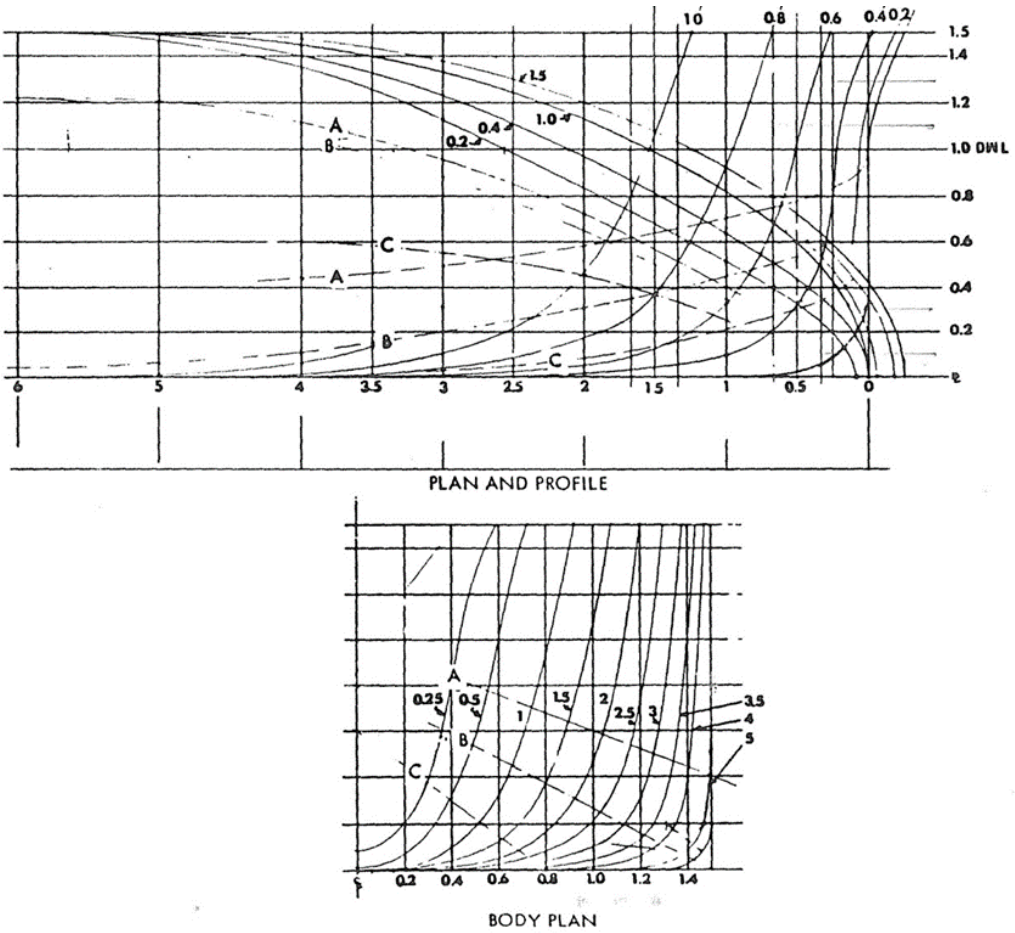
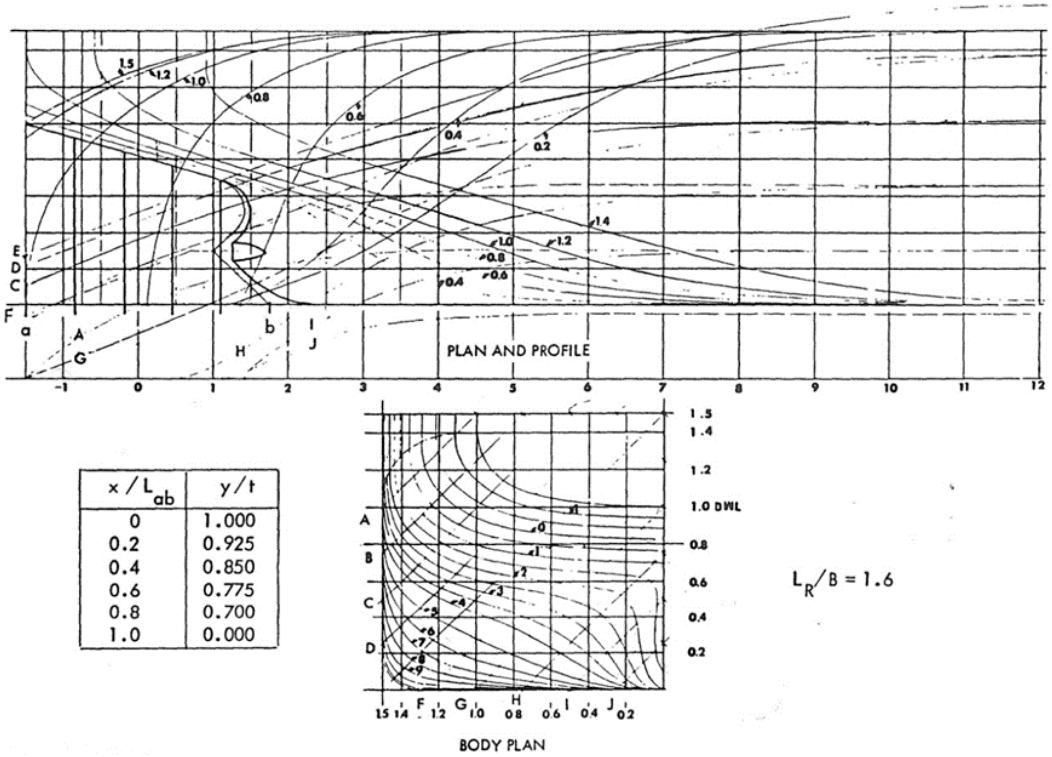
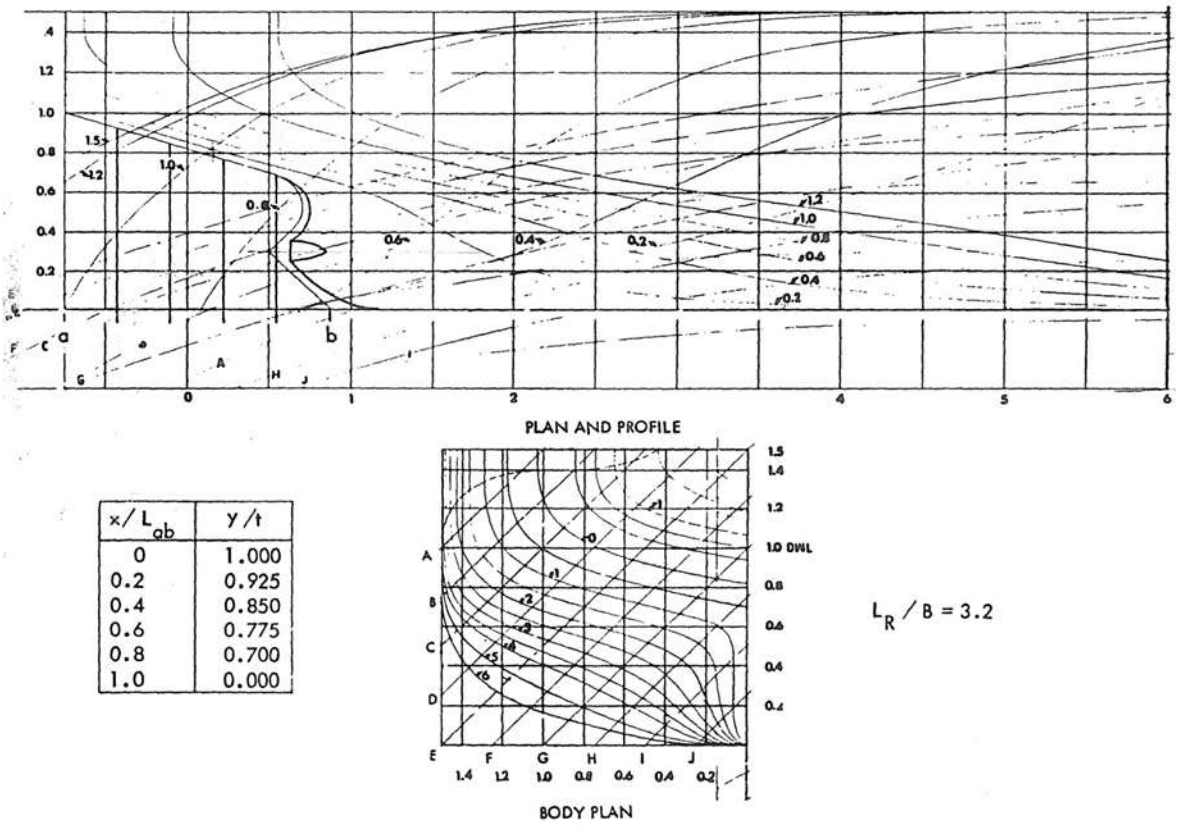


Fig. 1: Entrance lines of basic ship forms used to generate hull configuration for MARAD series [5]



x/L_{ab}	y/t
0	1.000
0.2	0.925
0.4	0.850
0.6	0.775
0.8	0.700
1.0	0.000

Fig. 2: Short run lines of basic ship forms used to generate hull configuration for MARAD series [5]



x/L_{ab}	y/t
0	1.000
0.2	0.925
0.4	0.850
0.6	0.775
0.8	0.700
1.0	0.000

Fig. 3: Long run lines of basic ship forms used to generate hull configuration for MARAD series [5]

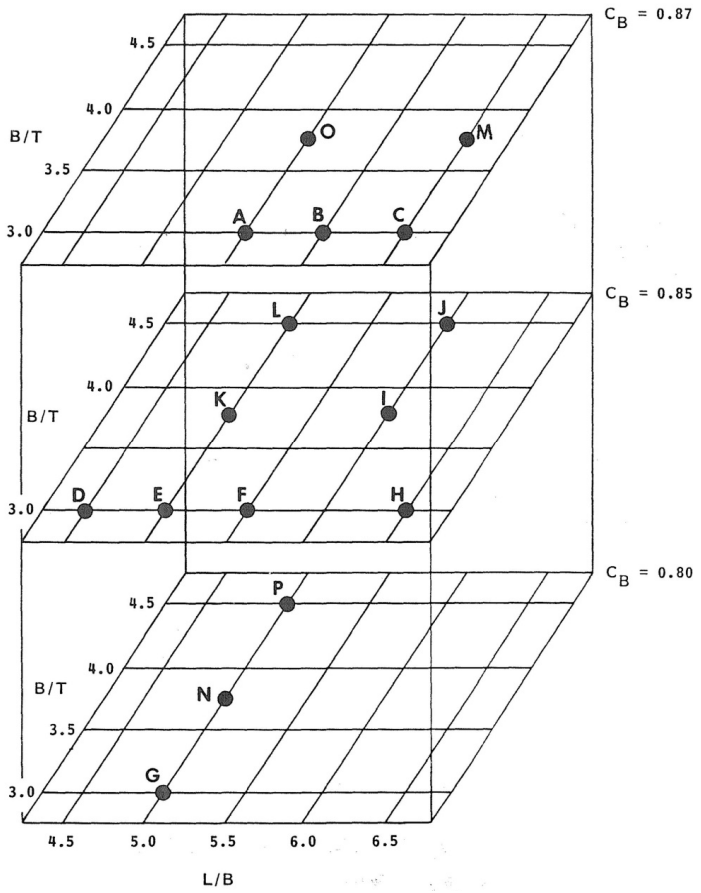


Fig. 4: Characteristics of the MARAD series hulls [5]

Tank tests are usually conducted and the resistance of the full scale ship is calculated from the resistance measurements of scaled models. The scaling of the tank test measurements for the calculation of the resistance of the full scale ship requires the understanding of the parts that comprise ship resistance and their behavior ([22], [23]).

- Frictional Resistance

This force component is present as a result of shear forces between the wetted surface and the elements of the fluid, due to its viscosity. The elements of the fluid close to the wetted surface have the velocity of the body. Subsequently, due to the created shear forces, layers of the fluid will follow the body's movement. As a result, the boundary layer is formed, with velocities ranging from that of the moving object to that of the flow field far away from the object. The tangential shear forces acting on each element of the hull can be summed over the wetted surface *S* of the object for the friction resistance to be calculated. It is a function of hull's wetted area, its velocity, its roughness and its geometry. It may be divided into two component forces; the flat plate frictional resistance and the form resistance, which is part of the residual resistance ([22], [23]).

- Pressure resistance

This part of the resistance arises from viscous effects and the hull's wave making, and based on that can be divided into viscous pressure resistance and wave resistance. Both forces are part of the residual resistance. The viscous pressure resistance is a function of the wetted area of the ship, its velocity and its hull's geometry. The creation of waves changes mainly the pressure distribution upon the hull and secondary the shear forces. As a result wave resistance is considered part of the pressure resistance and is affected by the hull's form and its velocity. The ship's pressure resistance may be calculated as the integral over the wetted surface of the pressure forces normal to the hull.

Except for the aforementioned components of ship resistance, there are also some secondary forces such as: wave breaking resistance, spray resistance, air resistance, resistance of appendages including propulsive equipment and added resistance due to turning ([22]).

Froude's Method of calculating ship resistance

According to this method the ship's total resistance is divided into a frictional component and a residual component. The frictional component is estimated as the resistance of a flat plate with length and surface equal to ship's length and wetted surface respectively, given by the following equation:

$$R_F = \frac{1}{2} C_F \rho S \upsilon^2 \qquad (1)$$

where ρ is the mass density of the liquid, S is the ship's wetted surface and C_F the frictional resistance coefficient which is nowadays usually expressed by the 1975 ITTC ([26]) formula as a function of Reynold's number (R_N):

$$C_F = \frac{0.075}{\left(\log_{10} R_N - 2\right)^2} \tag{2}$$

where

$$R_N = \frac{\upsilon L}{v} \tag{3}$$

and L is the ship's length, υ its speed and v the liquid's kinematic viscosity.

The difference between the total resistance and the frictional resistance, calculated according to eq. 1 is defined as the residual resistance which may be expressed in a form given in eq. (4):

$$R_R = \frac{1}{2} C_R \rho S \upsilon^2 \tag{4}$$

where C_R is the residual resistance coefficient given by:

$$C_R = C_T - C_F = \frac{R_T}{\frac{1}{2}\rho S \upsilon^2} - C_F \tag{5}$$

where C_T is the total resistance coefficient and R_T the total resistance of the ship.

Based on the above equations and the use of experimental data from a scale model's tests, the total resistance of a ship may be calculated. The model and the real ship share the same geometry and the same Froude number, defined as follows:

$$F_N = \frac{\upsilon}{\sqrt{gL}} \tag{6}$$

where g is the gravitational acceleration. Because the lengths of the ship and the model are different, the same applies to their velocities in order to have the same Froude number. If λ is the length ratio between the ship and the model ($\lambda = L_s / L_m$), then the model's speed should be equal to:

$$\upsilon_m = \frac{\upsilon_s}{\sqrt{\lambda}} \tag{7}$$

For a given speed range at full scale, tank tests are carried out at model speeds according to the above equation and the total resistance of the model is measured. Subsequently, the frictional resistance coefficient and the frictional resistance of the

model, based on eq. (2) and eq. (1) respectively, are calculated. The residual resistance of the model is calculated subtracting the frictional resistance from the total resistance and subsequently the residual resistance coefficient is calculated from eq. (5). For the same Froude number, the residual resistance coefficient at ship scale is assumed equal to that at model scale. Then, using eq. (2) the frictional resistance coefficient at ship scale is calculated and is subsequently summed with the residual resistance coefficient to derive the total resistance coefficient and ultimately the total resistance of the ship at the corresponding speed. The towing tanks usually introduce the so-called allowance coefficient, or correlation factor C_A. The total resistance coefficient of the ship in that case is given as follows:

$$C_{Ts} = C_{Rs} + C_{Fs} + C_A = C_{Rm} + C_{Fs} + C_A \tag{8}$$

where the index *m* corresponds to the model and *s* to the full scale ship.

1.4 MARAD diagrams of residual resistance coefficient

The experimental procedure described in the former chapter was followed by MarAd in order to obtain the residual resistance coefficient of each alternative hullform. Models of the 16 hulls were constructed with lengths ranging between 18 ft (5.49 m) to 27.48 ft (8.38 m). Bare hull resistance tests were performed for the 16 hulls at full load and ballast condition. The obtained results for the residual resistance coefficient (C_R) are presented in [5] in a series of diagrams, both at full load and ballast condition. The diagrams illustrate the C_R values versus length to breadth ratio (L/B) or breadth to draft ratio (B/T) for a series of Froude numbers (Fn), while block coefficient (C_B) and B/T or L/B were held constant respectively. In total, five diagrams for full load and five for ballast condition are presented in ([5]). However, the range of the diagrams does not coincide with the range of the hulls' parameters. In particular, the geometric properties of the ships denoted as M and O exceed the limits of the five diagrams. In addition, results for the C_R values of the 16 hulls at an assumed displacement of 350000t are presented in tabular form for a range of Froude numbers. The table corresponding to the full load condition copied from [5] is presented in the following (Table 1).

The data required for the networks' training in this study were obtained from the diagrams corresponding to full load condition. A total number of 1983 points were collected from these five graphs in the form of input/output pairs. The input is a four-dimensional vector consisting of the length to breadth ratio (L/B), the breadth to draft ratio (B/T), the block coefficient (C_B) and the Froude number (Fn), while the output is the residual resistance coefficient (C_R). The aforementioned diagrams are provided in Fig. 5 to

Fig. 9. Data from Table 1 were used for further testing of the networks. The total resistance of MARAD-type hullforms of an assumed displacement of 350000 t was calculated using C_R values from the diagrams or the table, the estimation of C_F based on eq. (2) and setting C_A equal to 0.00015 ([5]).

Table 1: Bare hull Residuary resistance coefficient versus Froude number for Δ=350000 t at full load condition (values divided by 10^3) [5]

	Designation															
Froude Number	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
0.100	0.960	0.650	0.560	1.200	0.780	0.580	0.520	0.470	0.430	0.340	0.530	0.580	0.410	0.490	0.850	0.470
0.110	0.960	0.650	0.560	1.200	0.780	0.580	0.520	0.470	0.430	0.340	0.530	0.580	0.410	0.490	0.850	0.470
0.120	0.960	0.650	0.560	1.200	0.780	0.580	0.520	0.470	0.430	0.340	0.530	0.580	0.410	0.490	0.850	0.470
0.130	0.960	0.650	0.560	1.200	0.780	0.580	0.520	0.470	0.430	0.340	0.530	0.580	0.410	0.490	0.850	0.470
0.140	1.010	0.790	0.620	1.290	0.900	0.610	0.540	0.480	0.450	0.360	0.580	0.640	0.450	0.500	0.980	0.480
0.145	1.080	0.920	0.690	1.360	0.980	0.660	0.550	0.520	0.480	0.380	0.620	0.660	0.500	0.500	1.030	0.490
0.150	1.210	1.060	0.810	1.500	1.040	0.690	0.560	0.560	0.500	0.420	0.680	0.680	0.590	0.510	1.070	0.510
0.155	1.340	1.200	0.960	1.550	1.080	0.750	0.590	0.560	0.540	0.490	0.740	0.700	0.730	0.530	1.120	0.540
0.160	1.480	1.280	1.030	1.560	1.120	0.760	0.620	0.560	0.630	0.530	0.800	0.740	0.880	0.560	1.190	0.590
0.165	1.580	1.380	1.120	1.580	1.160	0.800	0.660	0.570	0.680	0.560	0.850	0.790	1.020	0.600	1.300	0.660
0.170	1.700	1.540	1.290	1.650	1.200	0.860	0.690	0.580	0.740	0.640	0.900	0.850	1.170	0.670	1.450	0.740
0.175	1.830	1.720	1.500	1.720	1.300	0.950	0.710	0.690	0.820	0.750	0.980	0.950	1.330	0.720	1.630	0.790
0.180	2.000	1.910	1.730	1.860	1.450	1.080	0.730	0.800	0.940	0.890	1.110	1.050	1.530	0.750	1.830	0.840
0.185	2.220	2.100		2.020	1.600	1.210	0.750	0.950	1.090	1.070	1.260	1.190	1.770	0.770	2.040	0.880
0.190	2.520	2.360		2.220	1.760	1.360	0.780	1.110	1.260	1.260	1.420	1.350	2.050	0.800	2.270	0.920
0.195	2.870	2.560		2.410	1.940	1.520	0.830	1.370	1.440	1.490	1.490		2.350	0.880	2.520	1.000
0.200	3.240			2.630			0.900		1.740		1.630			1.000	2.770	
0.205				2.870												
0.210				3.120												

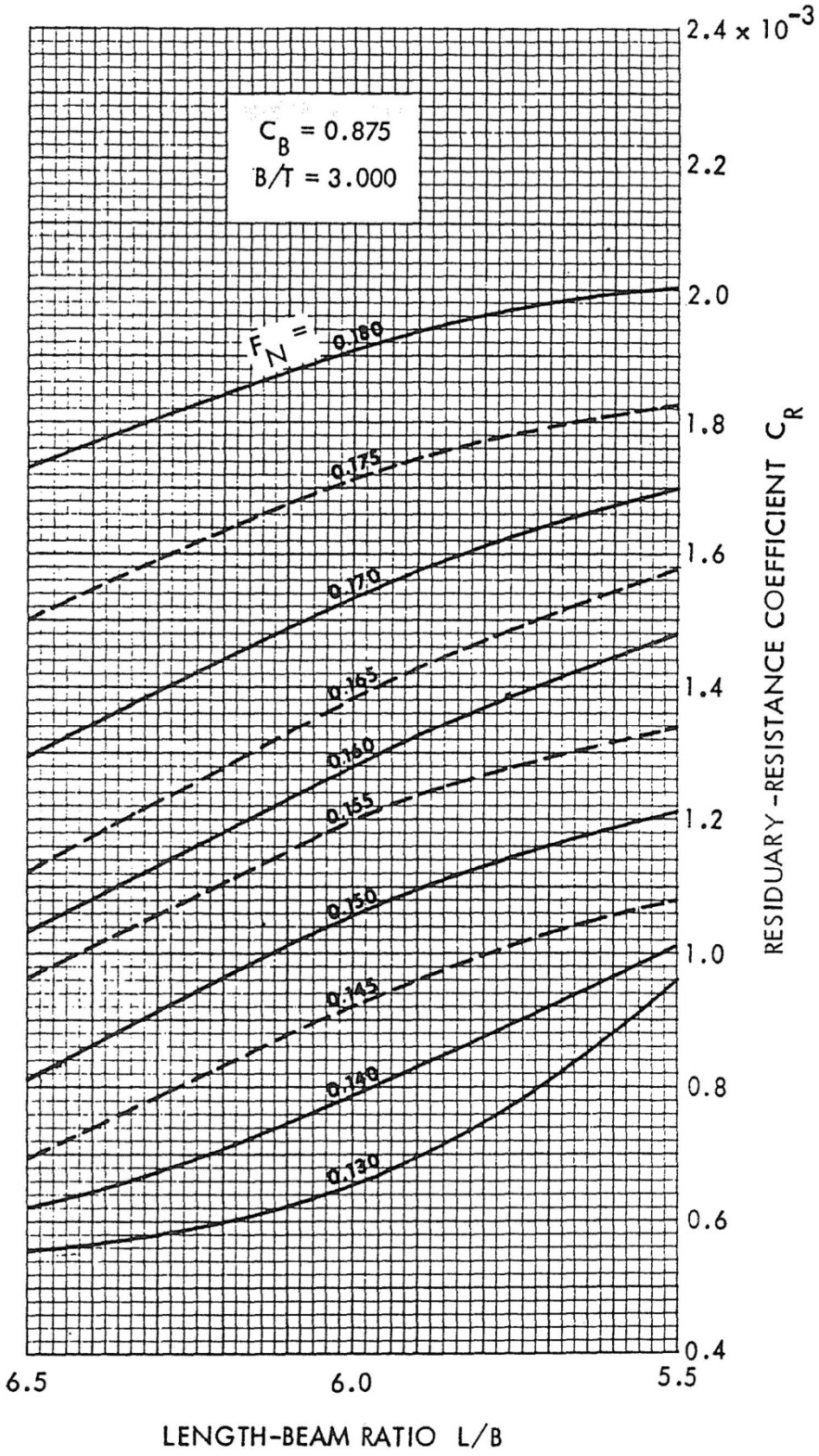


Fig. 5: C_R versus L/B for Fn values from 0.13 to 0.18

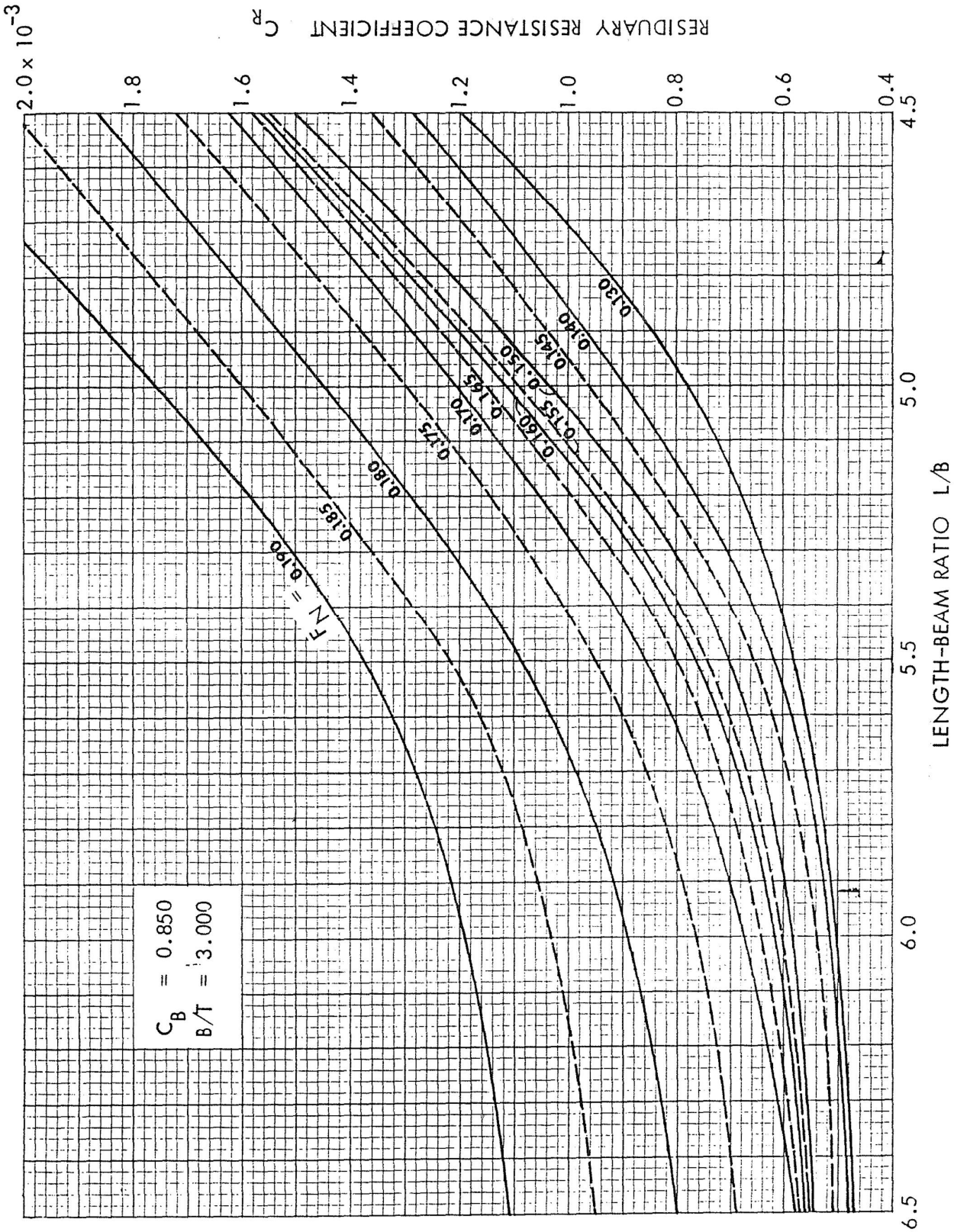


Fig. 6: CR versus L/B for Fn values from 0.13 to 0.19

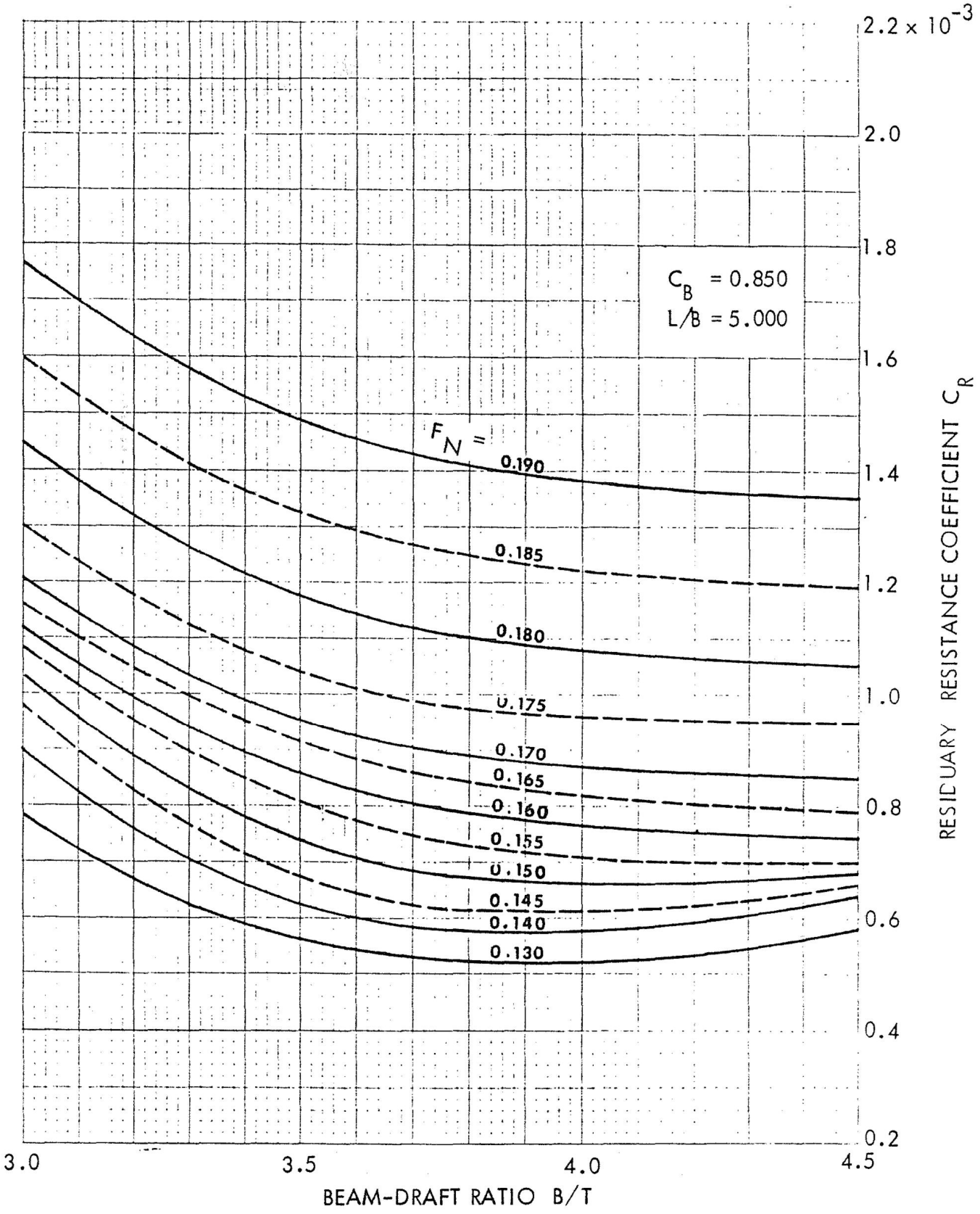


Fig. 7: C_R versus B/T for Fn values from 0.13 to 0.19

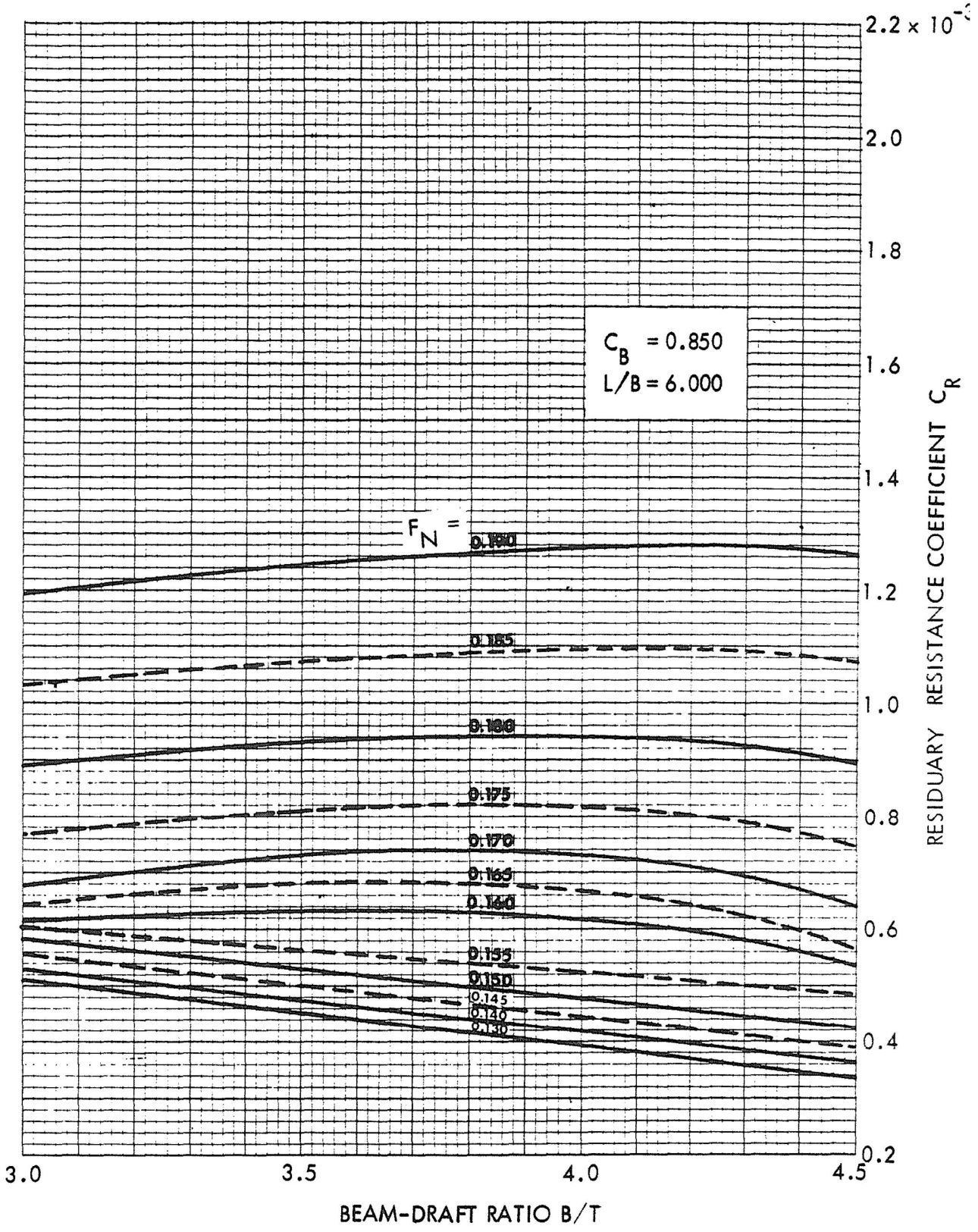


Fig. 8: C_R versus B/T for Fn values from 0.13 to 0.19

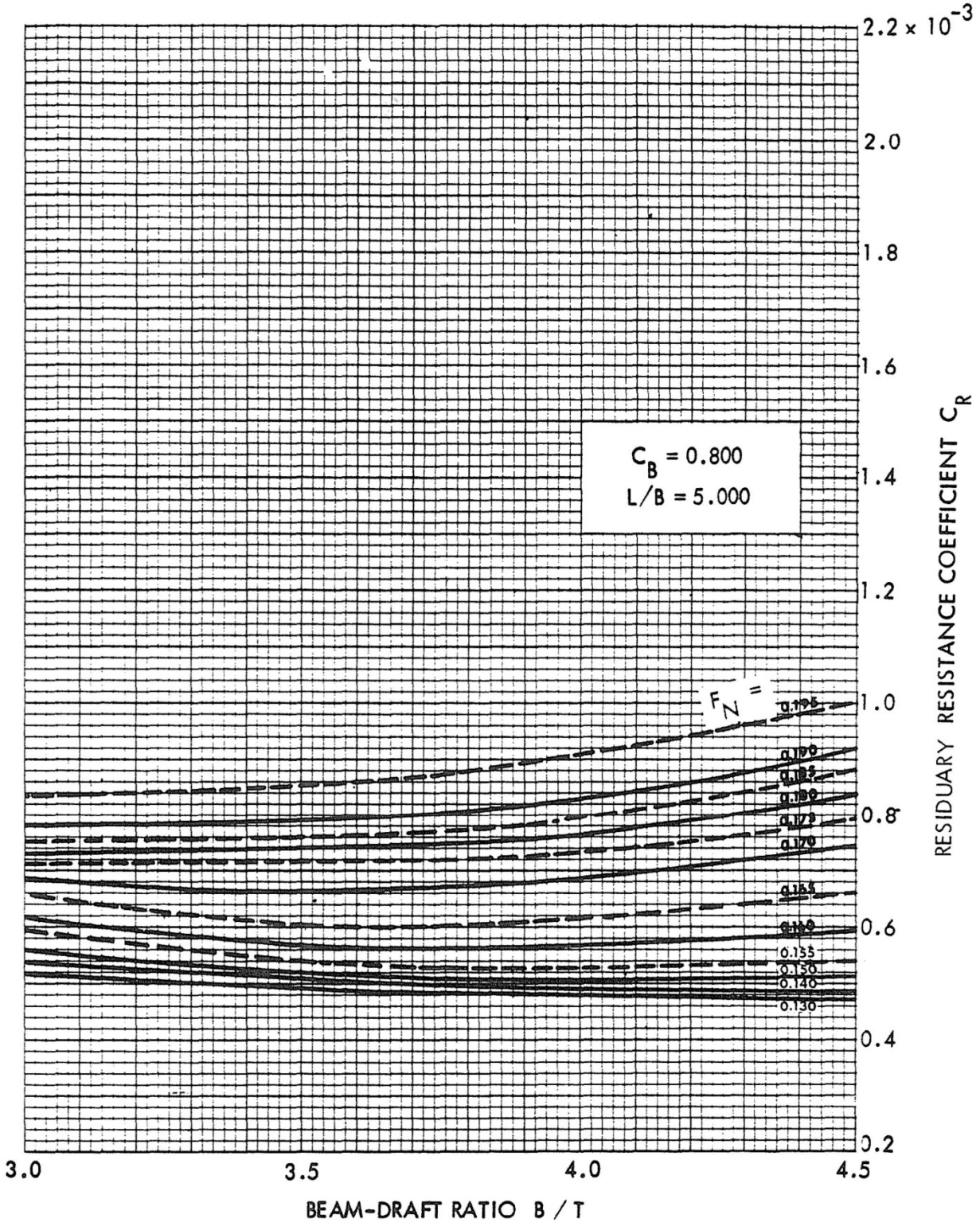


Fig. 9: C_R versus B/T for Fn values from 0.13 to 0.19

2. Artificial Neural Networks

2.1 Historical Information and Applications

Artificial Neural Networks (ANNs) are inspired by the human brain functionality and loosely model the way it processes sensory information, received from the environment. They consist of parallel-distributed, interconnected, non-linear processing units, the neurons. Information is acquired from their environment through a learning process and stored in the form of weights in the connections ([21]). A simplified model of the biological neuron was initially introduced in 1943 by McCulloch and Pitts ([27]), as a possible component of a computational system. This neuron had several inputs, both excitatory and inhibitory[1], and an output, which reflected the state of the neuron[2]. In order for the neuron to be activated, the sum of the excitatory signals had to exceed a threshold value T, while inhibitory signals were absent. In variations of this model, positive and negative stimulations were of the same importance, due to the use of adequate linear weights. Their work set the foundations of neural networks, forming a link between neurophysiology and mathematical logic ([21], [28]).

There were numerous milestones in the field of neural networks since they were originally introduced, regarding the architecture of the networks and their learning process. A well-known learning algorithm for ANNs is based on Hebb's theory, presented in 1949, regarding biological neural networks ([29]). According to this hypothesis, when the neurons connected by a synapse are stimulated repeatedly and simultaneously, this synapse becomes stronger, hence they are connected more effectively. In terms of ANNs, this is accomplished with the increase of synaptic weights ([21], [28]). One of the first ANNs to be developed was the perceptron, presented in 1957 by Rosenblatt ([30]). This simple neural network was a binary classifier that was based on the McCulloch–Pitts neuron, and had the ability of updating its weights through a numerical algorithm of supervised learning. This algorithm was based mathematically on the so-called perceptron convergence theorem. In 1960, the least mean-square (LMS) algorithm was introduced by Widrow and Hoff ([31]) and utilized for the development of the adaptive linear element (adaline), an early stage single-layer neural network, which also derived from the notes of McCulloch–Pitts. The multiple adaline (madaline) that was proposed by Widrow two years later ([32]), was one of the first multi-layered network architectures and also the first network to be utilized in a real life application, as a filter for elimination of echoes on phone lines, that is still in use.

Minsky and Papert analyzed several proposed networks and proved mathematically in 1969 that a single-layer network is not able to solve more complex problems, such as linear classification in more than two classes ([33]). The following decade was

[1] excitatory signals increase the probability of the neuron to be activated, whereas the inhibitory ones, in this model, prevent its activation
[2] the sum of the neuron's inputs

characterized by declined interest in neural networks, mainly because of the lack of the adequate technology, required for theory testing. However, in 1976 Willshaw and von der Malsburg ([34]) published a paper regarding the formation of neural connections on the brain using self-organization, that set the basis for later advancements.

In 1982 Hopfield ([35]) presented a complete mathematical analysis of recurrent neural networks with symmetric synaptic weights based on an Ising model, which is used in statistical physics. Although Hopfield drew information from earlier publications, this type of networks are usually referred to by his name ([21], [36]). In the same year, Kohonen ([37]) presented his work on self-organized maps using one and two dimensioned lattices, presenting differences from the model of Willshaw and von der Malsburg. Rumelhart, Hinton and Williams ([38]) reintroduced the back-propagation learning algorithm in 1986, proposing its use in machine learning and presenting its application for that cause. This algorithm adapts the synaptic weights in a way that minimizes the mean square error between the actual and desired output ([21], [36]). The publication of a book under the title Parallel Distributed Processing by Rumelhart and McClelland ([39]) contributed to the designation of the back-propagation algorithm as the most popular for the training of multi-layer perceptrons ([21]). In 1988, the use of radial basis functions to feed forward neural networks as an alternative to multi-layer perceptrons was given by Broomhead and Lowe ([40]), linking neural networks with the field of numerical analysis. In the early 1990's, the Support Vector Machines, a different type of supervised feed forward neural networks with robust computational properties and statistical learning background, were presented by Vapnik and coworkers ([41],[42]). Their work was on behalf of AT&Bell Laboratories, and as result these networks were oriented towards real life problems, such as character recognition ([21]).

Artificial neural networks evolved since the introduction of the McCulloch-Pitts model, with contributions from the disciplines of neurophysiology, mathematic logic, numerical analysis, psychology and physics. Nowadays, their application expands in many complex and practical problems related with several fields, such as economics, medicine and engineering. Autonomous drivers, weather or stock predictions, image recognition and optimal path selection could be solved with ANNs ([43]).

2.2 Artificial Neural Networks' characteristics

An Artificial Neural Network (ANN) consists of a distribution of interconnected neurons, arranged in layers, and is characterized by the number of its layers, the kind of its neural synapses and the learning paradigm it uses. According to the first characteristic they are divided into single-layer and multi-layer networks. The first type of networks contains only one layer of neurons as its output layer, while the later has 'hidden' layers between the input and the output layer ([21]).

With respect to the kind of their neural synapses, networks may be characterized as feed-forward and recurrent. The data in feed-forward networks flow in one direction,

from the input layer to the output, through the hidden layers, if existent. Recurrent networks contain at least one feedback interconnection, which links the output of a neuron with the input of another, placed in the same or previous layer ([21], [36]). A network can be also characterized as fully or partially connected if every node is connected with every unit of its forward layer or not respectively ([21]).

A fundamental feature of ANNs is their ability to learn. For this purpose, they adjust the weights of their connections and threshold values in order to present a satisfactory behavior according to a prescribed criterion. There are three different learning paradigms; supervised, unsupervised and reinforcement learning ([21]). In supervised learning, the network is provided with a series of input and output pairs in order to find a function that connects them. This procedure is evaluated by a cost function, usually the mean square error between actual and desired output, which needs to be minimized. In unsupervised learning, the network has to classify the input data without external information, finding a pattern ([21],[44]). In reinforcement learning, the network is in continuous communication with its environment, which rewards or not the input – output mapping that has been achieved, in order to minimize a performance indicator ([21]).

2.3 Feed-forward Multi-layer Perceptron

One of the most widely used neural networks, particularly for function approximation or pattern recognition problems, is the feed-forward, multi-layer perceptron (MLP, [45]). MLPs consist of an input layer, one or more hidden layers and an output layer (Fig. 10). The number of the input and output nodes is equal to the dimension of the input and output data respectively. The signal is transferred in a forward direction from the input to the output layer ([21]). Every node except for the input layers' is a neuron with a differentiable activation function that generates its output ([21]). Sigmoid transfer functions, such as logistic sigmoid or hyperbolic tangent sigmoid, are usually preferred as activation functions for MLPs, because consecutive layers of neurons with non-linear activation functions enable the network to identify non-linear relationships between input and output. The linear transfer function is often used for the neurons of the output layer ([44]). The logistic sigmoid function is given by eq. (9) while its graphical representation is presented in Fig. 11. The hyperbolic tangent sigmoid function is given by eq. (10) and its graphical representation in Fig. 12.

$$y_k = f(u_k) = \frac{1}{1+e^{-\lambda u_k}} \tag{9}$$

$$y_k = f(u_k) = \frac{1-e^{-\lambda u_k}}{1+e^{-\lambda u_k}} \tag{10}$$

In the above equations, y_k is the output of the k-neuron and u_k is the weighted sum of its input synapses.

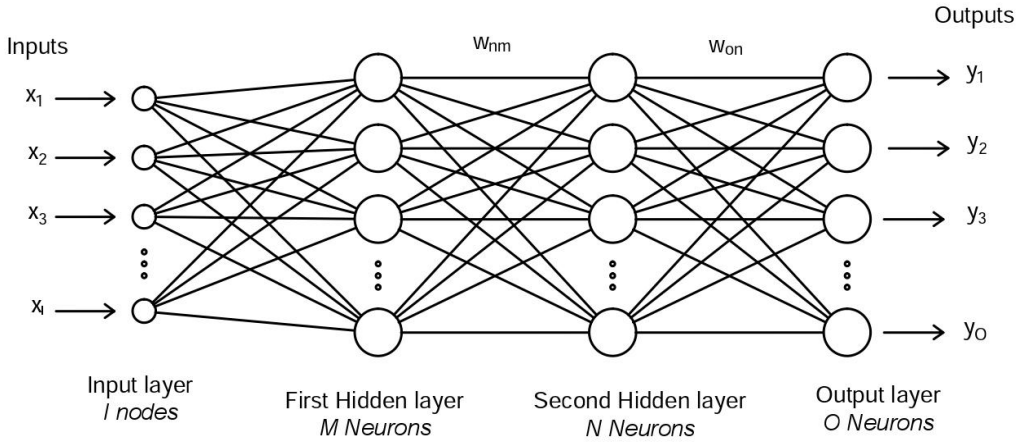


Fig. 10: Architecture of a multi-layer perceptron with two hidden layers

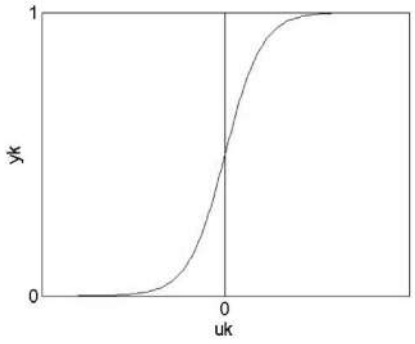


Fig. 11: Logistic sigmoid function

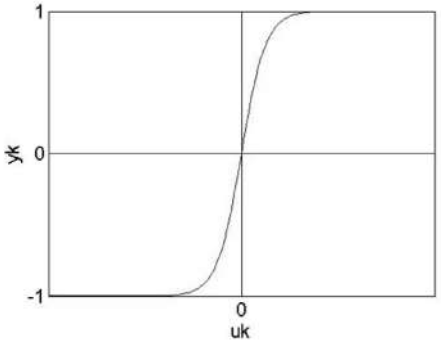


Fig. 12: Hyperbolic tangent sigmoid function

The training of these networks is accomplished with the use of the error back-propagation algorithm (BP), which is a supervised learning method ([39]). This algorithm is based on the error correction learning rule, and is executed in two stages: In the first stage, the input signal is transferred through the layers, in order for the output signal to be produced. During this phase, the synaptic weights remain constant. Then, the mean square error of the deviation between the actual response of the network and the desired output is calculated. In the second stage, these deviations are transferred backwards and the synaptic weights change accordingly, so that the average squared error is minimized ([21]). This cost function is given by eq.(11):

$$E_{av} = \frac{1}{T}\sum_{T} E(t) = \frac{1}{T}\sum_{K}\frac{1}{2}(d_k(t) - y_k(t))^2 \qquad t \in T \tag{11}$$

where k is the number of neurons in the output layer, T the number of input-output patterns in the training set, $d_k(t)$ the target output of the k neuron and $y_k(t)$ the actual output of this neuron given by the following equation:

$$y_k(t) = f(u_k(t)) \tag{12}$$

where u_k, the state of the neuron, is given by

$$u_k(t) = \sum_{i=0}^{n} w_{ki}(t) y_i(t) \tag{13}$$

where w_{ki} is the weight of the synapse between the i^{th} neuron of the previous layer and the k^{th} neuron of the output layer. A graphical representation of the structure of the k-output neuron is provided in Fig. 13. The gradient descent learning rule, known as delta rule is usually applied during the learning process for the adjustment of the weights. Other learning rules could be also used. The correction of the weights is given by eq. (14).

$$\Delta w_{ki}(t) = -\eta \frac{\partial E(T)}{\partial w_{ki}(t)} \tag{14}$$

where η is the learning rate parameter, which controls the rate of change of the weights and the bias of the network and affects the speed of convergence of the training ([21]).

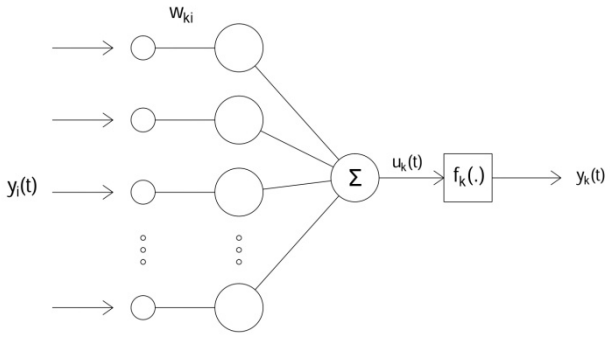


Fig. 13: Structure of k^{th} neuron output

2.3.1 Training algorithms

The training algorithm is the overall algorithm that is used to train the neural network to recognize a certain input and map it to a specified output. Back-propagation algorithms are used for the training of multi-layer perceptron networks and can be divided further into gradient descent, Newton-Gauss, conjugate gradient and quasi-Newton algorithms. In this study, three gradient descent algorithms and one quasi-Newton algorithm were utilized.

Gradient descent algorithms

They are first order optimization algorithms that employ the gradient vector of the cost function to decide the direction of the training, always pointing towards its negative value ([45]). The three gradient descent algorithms used in this study are:

- **Gradient descent back-propagation**

The weights and biases are updated in the direction of the negative gradient of the performance function. It is also called steepest descent.

- **Gradient descent with adaptive learning rate back-propagation**

This algorithm enables changes to the learning rate throughout the training, whereas in the steepest descent the learning rate is held constant. In this way, the learning rate adapts to the local complexity of the error surface. The algorithm aims to a high learning rate, as long as the errors decline. A higher learning rate is accepted, as far as the learning process stability is ensured; otherwise the learning rate is decreased as necessary ([46]).

- **Gradient descent with momentum and adaptive learning rate back-propagation**

It is similar to the gradient descent with adaptive learning with the addition of the momentum coefficient. The weights are adjusted according to gradient descent with momentum. This allows a network to respond not only to the local gradient, but also to recent trends in the error surface in order to ignore small features. As a result, the network would not stuck in a shallow local minimum ([46]).

Quasi-Newton algorithms

Newton's algorithm is a second order optimization algorithm that requires the calculation of a Hessian matrix, that contains second derivatives of the errors. Quasi-Newton Algorithms approximate the Hessian without the need of calculating second derivatives ([46]). The quasi-Newton algorithm used in this study is:

- **Levenberg-Marquardt algorithm**

The Levenberg-Marquardt ([47], [48]) training algorithm is mostly used for training small or medium sized networks ([49]). The update of the weights is defined as follows:

$$w_{m+1} = w_m - (\mathbf{J}^T\mathbf{J} + \mu\mathbf{I})^{-1}\mathbf{J}^T e \tag{15}$$

where **J** is the Jacobian matrix of the first derivatives of the network's errors with respect to its weights and biases, e the vector of network's errors, w_m the weights' vector after the m^{th} step and μ a positive parameter. The cost function of this method is the root mean square error:

$$E = \sqrt{\frac{\sum_{p=1}^{P}\sum_{m=1}^{M} e_{pm}^2}{PM}} \tag{16}$$

where P is the number of patterns and M the number of outputs. The μ parameter changes according to the error. If the error declines μ is divided by a constant factor, whereas if the error increases the last adjustment of weights is not valid and μ is multiplied by the constant factor.

2.3.2 Learning algorithms

The learning algorithm defines the way that weights and bias of the network will be updated during the training. Two different learning algorithms were utilized in this study: the gradient descent weight and bias and the gradient descent with momentum weight and bias.

2.4 Radial Basis Function Neural Networks

Radial Basis Function (RBF) neural networks belong to the category of supervised networks and are named after the activation functions they use. Their design does not resemble to a biological model, but originates from the field of numerical analysis. Radial basis functions were initially utilized in solving the real multivariate interpolation problem in 1987, as depicted in the paper of Powell ([50]), while their introduction in the design of neural networks was done by Broomhead and Lowe a year later ([40]). The learning process in RBFs could be considered as the problem of finding the best surface in a multi-dimensional space that provides best fit to the training data ([21]).

The standard three-layer architecture of RBF neural networks is shown in Fig. 14. The input layer consists of I source nodes, which is equal to the dimension of the input vector and connects the network to its environment. The hidden layer comprises K radial basis units (denoted as R_k) that apply non-linear transformation to the input vector. This layer is usually of high dimension, because its size is related to the network's capacity to approximate a smooth input-output mapping. The output layer is linear and gives the network's response to the input vector. The hidden and the output layers communicate through weighted connections ([21]). In some cases a bias term is added, as shown on the left diagram of Fig. 14 or be absent as in that on the right. The network with a single output can be used for function approximation problems, whereas that with multiple outputs usually for classification problems. In this study the first architecture is developed, and therefore analyzed herein.

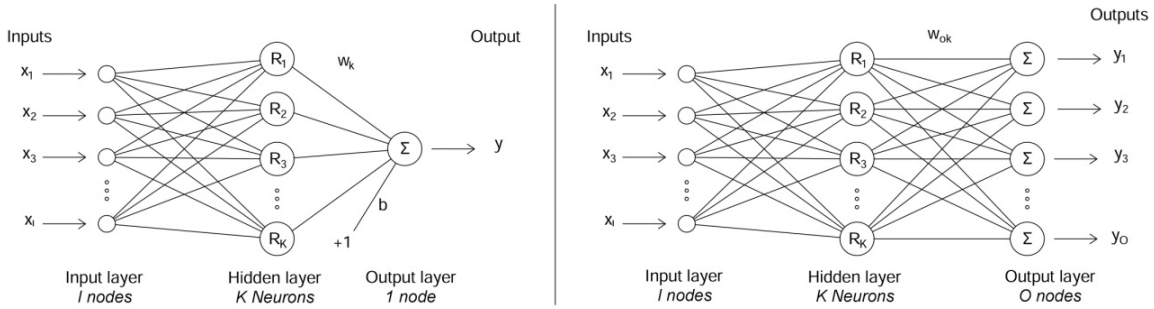


Fig. 14: Architecture of RBF neural networks with single output (left) and multiple outputs (right)

If $\mathbf{x_p}=[x_{p1}, x_{p2}, \ldots, x_{pI}]$ is the vector of the p^{th} pattern of the input data and R_k the radial basis function of the k^{th} neuron, w_k the weight of the connection between the k^{th} neuron of the hidden layer with the output, then the output of the network is given by eq. 17:

$$y_p = \sum_{k=1}^{K} w_k R_k(\mathbf{x_p}) \tag{17}$$

If a bias term b is existent, it is added to the right side of eq. (17) or preferably is considered as an additional weight w_0 with corresponding activation function $R_0(\mathbf{x_p})=1$ for every pattern p and is included to the sum. Commonly used types of radial basis functions include multiquadrics, inverse multiquadrics and polyharmonic splines, but the most preferred, and the one used in this study, is the Gaussian:

$$R_k(\mathbf{x_p}) = \phi(\|\mathbf{x_p} - \mathbf{c_k}\|) = \exp\left[\frac{-\|\mathbf{x_p} - \mathbf{c_k}\|^2}{2\sigma_k^2}\right] \tag{18}$$

where $\mathbf{c_k}$ is the I dimensional vector of the center of this radial basis function and σ_k^2 the standard deviation of the function of the k^{th} neuron. The selection of the center of this radial basis function is part of the learning process. As can be seen in eq. (18) the activations of the hidden units are determined by the distance between the input vector and a prototype vector (i.e. the center). In the following figure, a Gaussian-type one-dimensional RB function is presented for different values of σ_k^2.

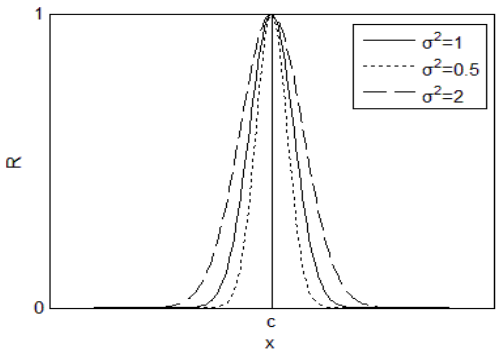


Fig. 15: Gaussian function for different standard deviation values

Radial basis functions can be normalized and used for developing normalized RBF neural networks. Let R_k' be the normalized radial basis function given by eq. (19).

$$R_k'(\mathbf{x_p}) = \frac{R_k(\mathbf{x_p})}{\sum_{k=0}^{K} R_k(\mathbf{x_p})} \tag{19}$$

The output of the network in that case is given by:

$$y_p = \frac{\sum_{k=0}^{K} w_k R_k(\mathbf{x_p})}{\sum_{k=0}^{K} R_k(\mathbf{x_p})} \tag{20}$$

In the above equations, a bias term is added, therefore the summation is starting from k=0. The normalized version of radial basis functions was selected in this study. The introduction of a regularization method in the design of the network is advised as a

means of improving its generalization ability, as seen in the work of Poggio and Girossi ([51]) and Broomhead and Lowe ([40]).

2.4.1 Learning strategies

The training of RBF neural networks is a two stage process. In the first stage, the parameters of the radial basis activation functions (i.e. the center vectors $\mathbf{c_k}$ and the standard deviations σ_k^2) are determined utilizing unsupervised learning methods (i.e. methods which use only the input data and not the target data). In the second stage, these parameters remain constant and the solution of a linear problem gives the weights of the connections between the hidden and the output layers. These sub-problems are easier to solve compared to the application of the back-propagation algorithm, and as result these networks are faster to train than multi-layer perceptrons ([20]).

For the selection of the radial basis functions' centers there are a few different approaches. The centers could be a randomly selected subset of the training data, but usually this method is used as a first step of an optimization process. Another option is to start with all data points (patterns) as centers of the activation functions of the neurons. These neurons are then removed one by one as long as the performance of the network remains intact. Alternative methods for the selection of the centers are e.g. the orthogonal least-squares and the Gaussian-mixture model. However, the most widely used training method for RBF neural networks, and the one adopted in this study, is the K-means clustering algorithm. The number of the neurons (K) is defined in advance and the value of standard deviation is also set and is kept constant for the activation functions of all neurons ([20]).

The algorithm aims to divide the training data $\{\mathbf{x_p}\}$ with p=1, 2, …, P into K disjoint subsets (clusters) S_k of N_k members each, in such a way that the following metric is minimized:

$$F = \sum_{k=1}^{K} \sum_{p \in S_k} \left\| \mathbf{x_p} - \mathbf{c_k} \right\|^2 \tag{21}$$

where $\mathbf{c_k}$ (center) is the mean of the data points included in the S_k subset and is given by

$$\mathbf{c_k} = \frac{1}{N_k} \sum_{p \in S_k} \mathbf{x_p} \tag{22}$$

In order to accomplish that the algorithm does the following: It begins by randomly assigning the data to the S_k subsets and computes the center $\mathbf{c_k}$ using eq. (22). Then, each pattern $\mathbf{x_p}$ is reassigned according to its distance from the clusters' centers, to the closest one. The new centers $\mathbf{c_k}$ of the clusters are recalculated. This process is repeated until no change in the partitioning of the data is observed. It is proven that through this process the value of F will not increase ([52]).

After the centers have been selected, the weights are given by the solution of a pseudo inverse problem. Let d_p be the desired output of the network for the p^{th} input pattern $\mathbf{x_p}$. In order to fit the network to the training data, the actual output of the network would be required to be equal to the desired output for every presented input pattern, which using eq. (17) leads to the following equation:

$$d_p = y_p = \sum_{k=1}^{K} w_k R_k(\mathbf{x_p}) \tag{23}$$

Let **G** be a matrix with $g_{ij}=R_i(\mathbf{x_j})$ for $i = 1, 2,\ldots, K$ and $j = , 2, \ldots, P$. If a bias value is used, the matrix has an extra column at the position $k+1$ with entries equal to one and the vector of weight has an extra entry, the bias, at the position $k+1$. The desired outputs and the weights could form a P-dimensioned and K-dimensioned column vector **d** and **w** respectively. Then eq. (23) could be rewritten as follows:

$$\mathbf{d} = \mathbf{G}\mathbf{w} \tag{24}$$

The matrix **G** does not have an inverse. In order to overcome this difficulty the pseudo-inverse matrix $\mathbf{G}^+$ is used; where:

$$\mathbf{G}^+ = (\mathbf{G}^T\mathbf{G})^{-1}\mathbf{G}^T \tag{25}$$

As a result the weights of the output synapses are calculated setting:

$$\mathbf{w} = \mathbf{G}^+\mathbf{d} \tag{26}$$

Additional information regarding this approach and its theoretical background could be found in ([40]) and ([51]).

2.5 Support Vector Machines

Support Vector Machines (SVMs) are supervised, feed forward neural networks with a single hidden layer that originated from the field of statistical learning and are named after the learning algorithm that is used for their training ([21]). The Support Vector (SV) algorithm derives from the generalization portrait algorithm, which was introduced by Vapnik and Lerner ([53]) in 1963 for the solution of a pattern recognition problem and further developed by Vapnik and Chervonenkis ([54]). The implementation of the SV algorithm by learning machines of a form close to that of currently used SVMs was presented in 1992 and subsequently evolved by Vapnik and several coworkers ([41], [42]) at AT & Bell Laboratories. Although, these machines were initially used for pattern classification problems, their application expanded successfully in other problems such as regression and time series ([55], [56]).

The SV algorithm differs in a few ways from the other neural network learning methods. The weights of an SVM network result from the solution of a Quadratic Programming (QP) problem with linear constraints, while that of other ANNs through solving a non-convex, unconstraint minimization problem ([57]), as it was described

in the case of MLPs and RBFs (see Chapters 2.3 and 2.4). Another difference, is that the cost function of an SVM takes into account the structure of the network.

Different types of learning processes are used in order to solve regression or classification problems ([21]) with SVMs. Considering that the problem investigated in this thesis is a regression one, the basic concepts regarding SVMs will be described through the corresponding learning process, which is usually referred to as Support Vector Regression (SVR). There are a few approaches regarding the solution of a regression problem with SVM. Among them, the most widely known are the ε-SVR and the v-SVR. The first one is used in this study, and analyzed herein.

2.5.1 Support Vector Regression

The SVM has a three-layered structure with one hidden layer, as already mentioned, and its output for the nonlinear regression problem is the following:

$$y_p = \sum_{m=0}^{M} w_m \phi_m(\mathbf{x_p}) = \mathbf{w}^T \varphi(\mathbf{x_p}) \tag{27}$$

where $\boldsymbol{x_p} = [x_{p1}, x_{p2}, \ldots, x_{pI}]$ is the p^{th} input pattern with $p = 1, 2, \ldots, P$, w_m is the weight of the connection between the m^{th} neuron of the hidden layer and the output layer and $\mathbf{w} = [w_0, w_1, \ldots, w_M]^{\text{T}}$, $\{\varphi_m(\mathbf{x})\}$ a set of non-linear basis functions with $m = 0, 1, \ldots, M$ and $\boldsymbol{\varphi}(\mathbf{x_p}) = [\varphi_0(\mathbf{x_p}), \varphi_1(\mathbf{x_p}), \ldots, \varphi_m(\mathbf{x_p})]$. The bias term is represented by the weight w_0 with corresponding activation function $\varphi_0(\mathbf{x_p}) = 1$ for every pattern p. The weights w and their number are the results of the learning process. The SVR learning algorithm defines the weights and the number M of the neurons, minimizing the following value, known as empirical risk:

$$R_{emp} = \frac{1}{P} \sum_{p=1}^{P} L_\varepsilon \left(d_p, y_p\right)^2 \tag{28}$$

subjected to

$$\|\mathbf{w}\|^2 \le c_o \tag{29}$$

where c_o is a constant, and

$$L_\varepsilon\left(d_p, y_p\right) = \begin{cases} \left|d_p - y_p\right| - \varepsilon & for \quad \left|d_p - y_p\right| \ge \varepsilon \\ 0 & otherwise \end{cases} \tag{30}$$

where d_p is the desired output for the p^{th} input pattern $\boldsymbol{x_p}$. This is the so-called ε-intensive loss function. It ignores errors less than ε, which is a user defined positive value, and also allows the model to approximate d_p with a specific accuracy ([58]). After appropriate calculations ([2142], [42]), this initial minimization problem is

reformed to the dual optimization problem of finding the values $\{\alpha_p\}$, $\{\alpha_p^*\}$ with $p = 1, 2, \ldots, P$ that maximize the objective function given in eq. (31):

$$Q(\alpha_p, \alpha_p^*) = \sum_{p=1}^{P} d_p(\alpha_p - \alpha_p^*) - \varepsilon \sum_{p=1}^{P}(\alpha_p + \alpha_p^*) - \frac{1}{2}\sum_{i=1}^{P}\sum_{j=1}^{P}(\alpha_i - \alpha_i^*)(\alpha_j + \alpha_j^*)K(\mathbf{x}_i, \mathbf{x}_j) \tag{31}$$

subjected to the following constraints:

$$\begin{aligned} &\sum_{p=1}^{P}(\alpha_p - \alpha_p^*) = 0 \\ &0 \le \alpha_p \le C \\ &0 \le \alpha_p^* \le C \end{aligned} \tag{32}$$

where C is a user defined parameter, also called box constraint, and K is the inner-product Kernel function:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \varphi^{\mathrm{T}}(\mathbf{x}_i)\varphi(\mathbf{x}_j) \tag{33}$$

The radial basis function φ maps the problem to the feature space, but the introduction of the Kernel function eliminates the need of computing the values $\varphi(\mathbf{x_p})$ for every pattern p for their inner product to be obtained. There are several types of inner-product Kernel functions appropriate for use in SVMs, such as the polynomial with equation:

$$K(\mathbf{x}_i, \mathbf{x}_j) = (\lambda \mathbf{x}_i \mathbf{x}_j + 1)^n \tag{34}$$

where its degree n and the kernel scale λ are user defined parameters, and the Gaussian function described by eq. (35), where only the kernel scale λ, which is equal to $1/2\sigma^2$, needs to be defined.

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{1}{2\sigma^2}\left\|\mathbf{x}_i - \mathbf{x}_j\right\|^2\right) \tag{35}$$

It is also proven that the vector of weights is given by the following equation:

$$\mathbf{w} = \sum_{p=1}^{P}(\alpha_p - \alpha_p^*)\varphi(\mathbf{x}) \tag{36}$$

The estimation y_p of the network for the pattern $\mathbf{x_p}$ given by eq. (27) is reformed taking into account eq. (36) and eq. (33) as follows:

$$y_p = \sum_{m=0}^{M}(\alpha_m - a_m^*)K(\mathbf{x_m}, \mathbf{x_p}) = \sum_{m=0}^{M} v_m K(\mathbf{x_m}, \mathbf{x_p}) \tag{37}$$

The input patterns $\{\mathbf{x_m}\}$ with corresponding multipliers $\alpha_m \neq \alpha_m^*$ are the support vectors of the SVM. Their number M is part of the optimization's problem solution. The ε and C are user defined parameters, along with the selection of the Kernel function K and its parameters ([21]).

The number of the occurring support vectors influences the complexity of the produced SVM. The ε value affects that number through the loss function that allows errors lower than ε. A higher ε value leads to fewer support vectors, and subsequently a more flat model (i.e. smaller w values), which is preferred. C value influences the weight values and the margin of acceptable errors above ε. Therefore, both parameters determine models complexity in different ways ([59]). These parameters should be defined simultaneously. The architecture of a trained SVM is depicted in Fig. 16.

It is worth mentioning, that the dual optimization problem is a QP problem that is computationally demanding to solve for large datasets. To this end, several algorithms were proposed. The chunking algorithm ([58]), the Osuna's algorithm ([57]) and the Sequential Minimal Optimization algorithm (SMO) ([60]). The last one is the most widely adopted.

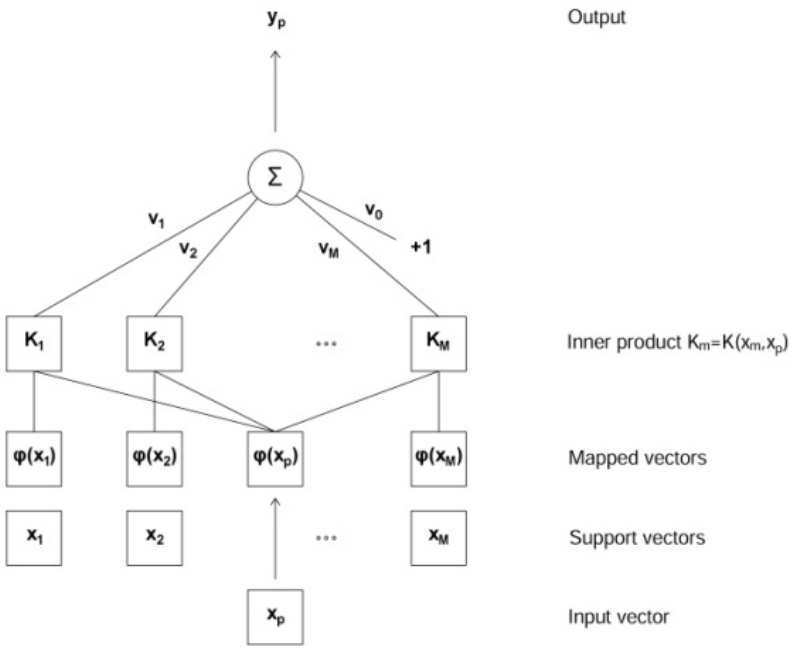


Fig. 16: Architecture of a trained SVM

2.6 Data sets

The available data for the network training are usually divided into three mutually exclusive sets.

a. Training Set: This set of data is used for the update of weights and biases of the network in order for the cost function to be minimized.

b. Validation Set: This set is used for the tuning of parameters other than weights and for comparison between different neural networks.

c. Test Set: This is the set of data used for evaluation of the predictive ability of the network.

The scheme of selecting a validation set as a percentage of the available for training data is also known as holdout validation. There is also another way of selecting a validation set, usually used for small datasets; the k-fold validation. This scheme divides the available data in k disjoint folds. Subsequently, trains the algorithm k times using each time a different fold as the validation set and the remaining as training set. After each passing the corresponding validation error is calculated. The validation error of the network is equal to the average of all calculated validation errors. The number of the folds k is user defined ([61]).

2.7 MATLAB Environment

The development of the ANNs for the approximation of the resistance of ships designed according to the MARAD series was carried out using the MATLAB ([62], [63]) software. MATLAB is a fourth-generation programming language that is widely used in applications of linear algebra, numerical analysis, implementation of control systems and image processing by both researchers and practitioners.

Except for a library of functions and a compiler MATLAB includes numerous Toolboxes specially designed for specific applications. In this study, two of these were used; the Neural Network Toolbox ([49],[64]) and the Statistics and Machine Learning Toolbox ([61], [65]). These added features provide functions and an app with a handy interface for designing, implementing, visualizing and evaluating neural networks and learning machines.

Neural Network Toolbox is suitable for developing MLPs, and through its function library allows the user to set the learning rate, the division ratio of the data and the distribution of the neurons in the layers. There is also a list of available options regarding the activation functions of the neurons, the training and learning algorithm of the network. Although the corresponding app is easy to handle, it does not enable changes to all of the aforementioned network configurations. As a result, the developed MLPs were trained using a script that was generated for this purpose, deploying the functions available by the function library of the Toolbox. The script was used to set the number and the distribution of the neurons to layers, the applied training and learning algorithm, the learning rate, the division ratio of the data to sets (training, validation, test) and a stopping criterion.

On the contrary, the available functions concerning RBFs were considerably fewer, and in any case not enough to allow for the use of Neural Network Toolbox. RBFs were trained with a user developed script that applied the learning strategy of K-means described in chapter 2.4.1 to a network with the Gaussian as activation function of its neurons. The generated script allows the user to set the number of

neurons of the hidden layer, the sigma parameter of the Gaussian, and also calculates the performance indices of the network, which are necessary for its evaluation. A modified algorithm, as a means of achieving better results, was also developed in the form of script.

The Statistics and Machine Learning Toolbox, and specifically the Regression Learner app, was used for the development of SVMs. This app offers a variety of regression learning algorithms and among them is the SVM algorithm, with six commonly used configurations for the kernel function. The app uses by default the SMO algorithm for the solution of the quadratic programming problem and the ε-intensive loss function. The user through the app's window defines the validation technique, the kernel function, the box constraint (C) and ε values. The app trains the network, calculates the performance indices and notes the most robust network among those produced by the user. Because of the Toolbox's flexibility it was used for the development of the SVMs.

All networks were developed on a personal portable computer equipped with a quad core processor (Intel® Core™ i7-2670QM at 2.2GHz) and a 4 GB of DDR3 RAM. The 2016b version of the MATLAB software was used for the development of MLPs and RBFs, and the 2017a for SVMs.

3. Development of Artificial Neural Networks

3.1 Multi-layer Perceptrons

3.1.1 Implementation

In order to determine the MLP that suits better to the problem at hand, several trials have been conducted with different configurations. Each configuration is determined by ([49]):

- the number of hidden layers used,
- the number and type of neurons that comprise each one of them,
- the training algorithm,
- learning algorithm and rate.

These trials have been carried out in two stages. In the first stage, 18 different network architectures have been tested. For each one of them, 16 different networks have been implemented, based on a combination of four back-propagation training algorithms and four different pairs of activation functions. As a result, 288 networks were constructed and tested. During this initial set of trials, the gradient descent weight and bias learning algorithm was used and the learning rate was set to 0.02. The available data was separated into mutually exclusive training (80%) and validation (20%) sets. The validation set comprises data unknown to the trained network and is usually used for the comparison and ranking of different network configurations. A graphical description of this stage of trials is given in Fig. 14. The four training algorithms used for this study are:

- the Levenberg-Marquardt algorithm,
- the gradient descent algorithm,
- the gradient descent with adaptive learning rate algorithm,
- the gradient descent with momentum and adaptive learning rate algorithm.

Three different activation functions have been used, i.e. the hyperbolic tangent sigmoid function (herein denoted as T), the pure linear function (denoted as P) and the logistic sigmoid function (denoted as L). These activation functions were tested in pairs as follows (Fig. 14):

- hyperbolic tangent sigmoid function for the neurons of the hidden layers and for those of the output layer,
- hyperbolic tangent sigmoid function for the neurons of the hidden layers and pure linear function for those of the output layer,
- logistic sigmoid function for the neurons of the hidden layers and for those of the output layer,
- logistic sigmoid function for the neurons of the hidden layers and pure linear function for those of the output layer.

For each one of these 288 networks, the corresponding mean squared error (MSE) and regression value (R) were calculated, based on the training set of data and on the validation set. The MSE is calculated as the average of the sum of squares of "errors". The error is the difference between the target output and the network output. The regression value is an indicator of the relationship between the actual and target outputs of the network and its absolute value ranges between one and zero, suggesting the existence of an exact linear relationship between outputs and targets or its absence respectively. The performance of these 288 networks was ranked according to their mean squared error (MSE) and regression value (R) based on the validation set.

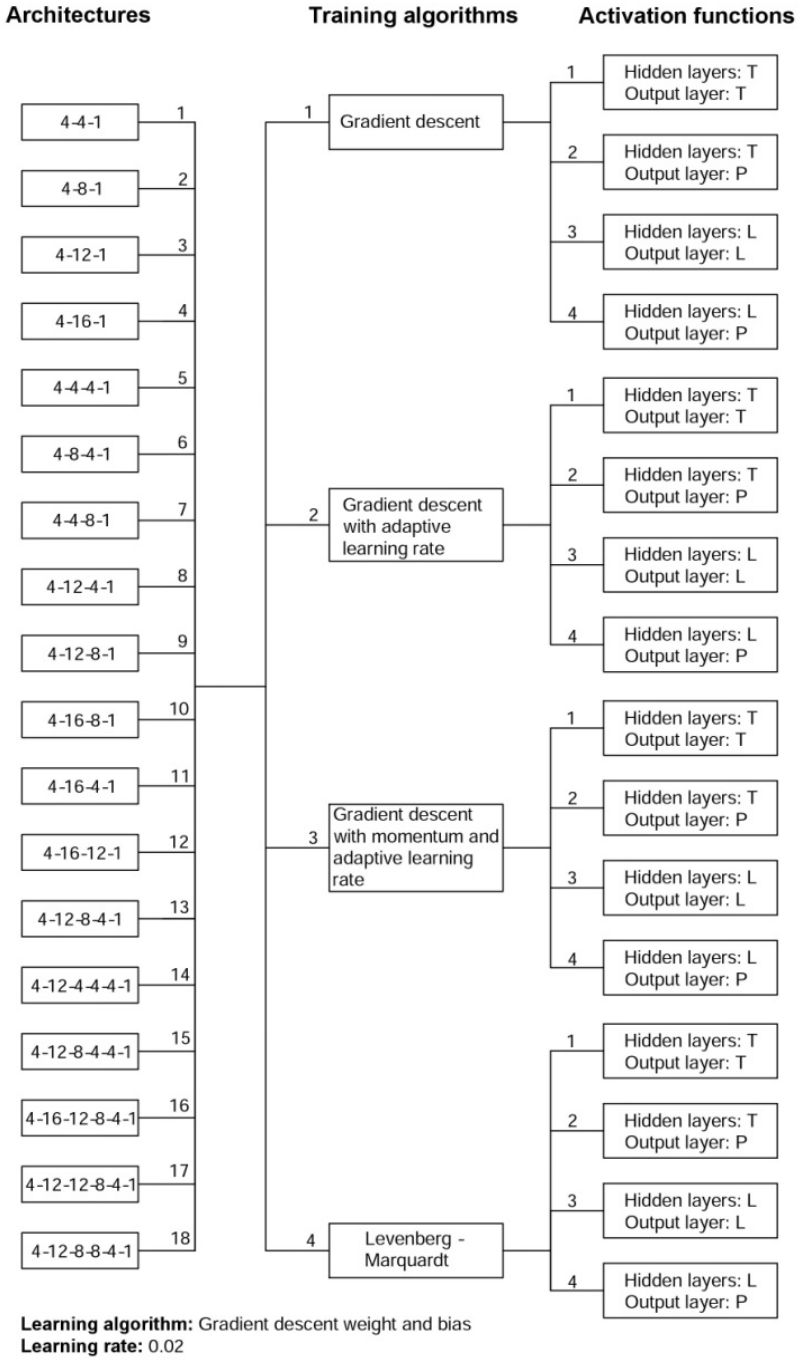


Fig. 17: Graphical description of the first stage of trials

Results from the evaluation of each one of the 18 architectures tested are presented in Table 2. The second column in the table shows the number of nodes per layer (input layer – hidden layer(s) – output layer). The number of nodes of the input and output layer is by default equal to 4 and 1 respectively, corresponding to the dimensions of the input and output vectors. The number of neurons and its distribution among the hidden layers varies. The minimum mean squared error values and the maximum regression values for each one of the 18 architectures obtained with the 16 combinations of training algorithms and activation functions, both for the training and validation sets of data, are presented in Table 2.

Table 2: Tested MLP architectures – first stage of trials

Serial number	Architecture	Training set		Validation set	
		min MSE	max R	min MSE	max R
1	4-4-1	0.0095	0.9625	0.0086	0.9819
2	4-8-1	0.0079	0.9695	0.0047	0.9694
3	4-12-1	0.0062	0.9750	0.0071	0.9738
4	4-16-1	0.0067	0.9749	0.0046	0.9808
5	4-4-4-1	0.0066	0.9699	0.0084	0.9670
6	4-8-4-1	0.0011	0.9733	0.0068	0.9744
7	4-4-8-1	0.0068	0.9736	0.0068	0.9776
8	4-12-4-1	0.0079	0.9778	0.0060	0.9814
9	4-12-8-1	0.0059	0.9776	0.0052	0.9836
10	4-16-8-1	0.0034	0.9864	0.0047	0.9791
11	4-16-4-1	0.0041	0.9834	0.0062	0.9800
12	4-16-12-1	0.0034	0.9866	0.0047	0.9798
13	4-12-8-4-1	0.0043	0.9832	0.0042	0.9845
14	4-12-4-4-4-1	0.0044	0.9831	0.0064	0.9762
15	4-12-8-4-4-1	0.0050	0.9808	0.0058	0.9822
16	4-16-12-8-4-1	0.0039	0.9845	0.0046	0.9796
17	4-12-12-8-4-1	0.0048	0.9819	0.0049	0.9796
18	4-12-8-8-4-1	0.0045	0.9817	0.0055	0.9598

Table 3: Properties of networks – second stage of trials

Code number	Architecture	Training algorithms	Activation function	
			Hidden layers	Output layer
2.4.2	4-8-1	LM	T	P
4.4.1	4-16-1	LM	T	T
10.4.2	4-16-8-1	LM	T	P
13.4.1	4-12-8-4-1	LM	T	T
16.4.2	4-16-12-8-4-1	LM	T	P
16.4.4	4-16-12-8-4-1	LM	L	P

The ten networks that presented the lower values of MSE and the ten with the higher values of R, based on the validation dataset, were selected for further testing. According to the above selection criteria, and due to the observed overlapping between them, six networks were eventually identified for further study. The architecture and the activation functions of these networks are presented in Fig. 18. The first digit of their code numbers corresponds to the serial number of their architecture, the second one to the training algorithm used and the third one to the combination of activation functions as described in Fig. 17.

The most important factor associated with improved MSE and R values was the training algorithm. The Levenberg-Marquardt training algorithm presented the best results, in comparison with the other three training algorithms that were tested. With respect to the pairs of activation functions, the networks that consisted only of logistic sigmoid neurons presented the worst performance. As a result, all the networks selected for the second stage of trials were using the Levenberg-Marquardt algorithm, while the networks with only logistic sigmoid neurons were excluded.

The objective of the second stage of trials was to determine the most appropriate learning algorithm and rate. To this end, tests were carried out using six different learning rates (0.01, 0.02, 0.05, 0.1, 0.15, 0.2) and two learning algorithms: the gradient descent weight and bias (used also during the first stage) and the gradient descent with momentum weight and bias. As a result, a total of 72 networks were tested during this second stage of trials (Fig. 18).

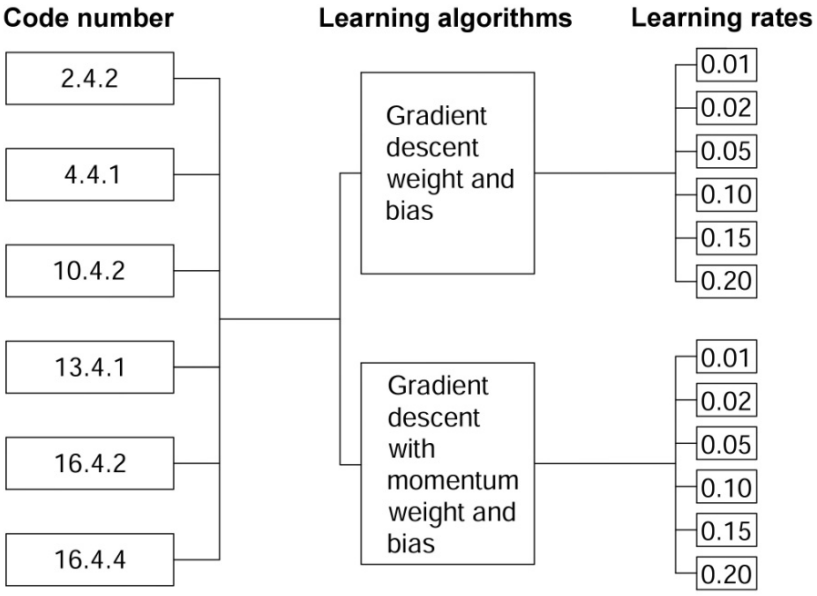


Fig. 18: Graphical description of the second stage of trials

For this stage, the available data was divided into training, validation and test dataset with ratios 80%, 12% and 8% respectively. For each one of the six networks listed in Table 4 the minimum MSE and maximum R values obtained by the 12 combinations of learning rates and learning algorithms, both for the training and validation sets of data, are presented in Table 4.

Table 4: Obtained MSE & R values – second stage of trials

a/a	Training set		Validation set	
	min MSE	max R	min MSE	max R
2.4.2	0.0031	0.9875	0.0006	0.9978
4.4.1	0.0028	0.9886	0.0006	0.9978
10.4.2	0.0005	0.9977	0.0002	0.9993
13.4.1	0.0016	0.9981	0.0003	0.9989
16.4.1	0.0003	0.9989	0.0003	0.9986
16.4.4	0.0002	0.9991	0.0001	0.9994

Additional trials were conducted for the effect of the division ratio of the dataset to the performance of a network to be examined. To this end, ten of the 72 network configurations developed in the second stage of trials were also trained with datasets divided with two other division settings; 75%, 15%, 10% and 85%, 10%,5% ratios for training, validation and test dataset respectively. The validation MSE values of networks 2.4.2/0.15/2 and 16.4.2/0.20/1, were lower when the training set accounted for the 75% of the available data. The last one presented higher validation R value for that division ratio too. Six of the networks trained with the selected for this study division ratio (i.e. 80% of the data are assigned to the training set) were the best or the second best compared to networks trained with other division ratios with respect to both MSE and R values of the validation set. The MSE and R values for all datasets for these 10 networks and the three division settings are presented in Fig. 19 and Fig. 20. Each network in these figures is characterized by three numbers divided by slashes; its code number, its learning rate and its learning algorithm, which is given by the code number 1 for the gradient descent weight and bias and 2 for the gradient descent with momentum weight and bias (see also Fig. 18).

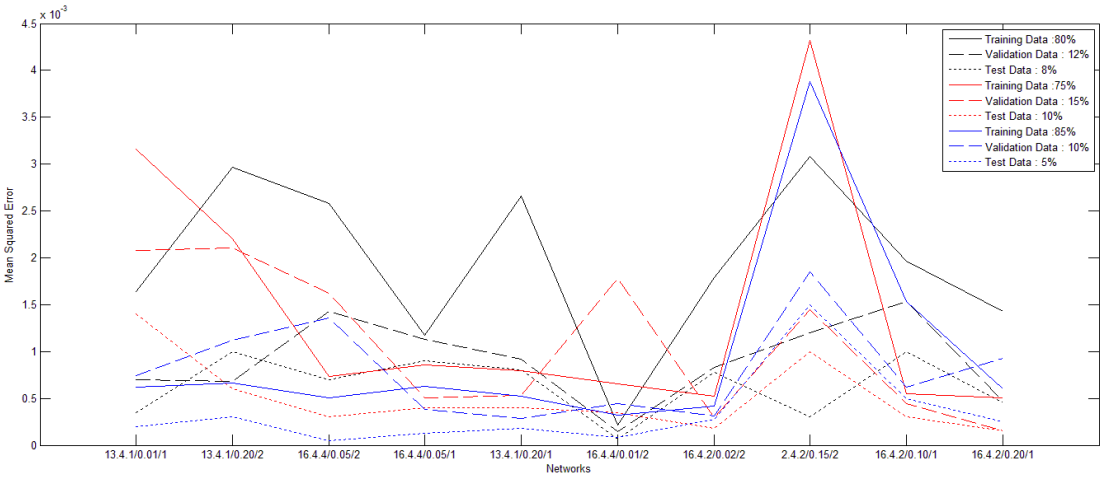


Fig. 19: MSE values of 10 networks for three different division settings

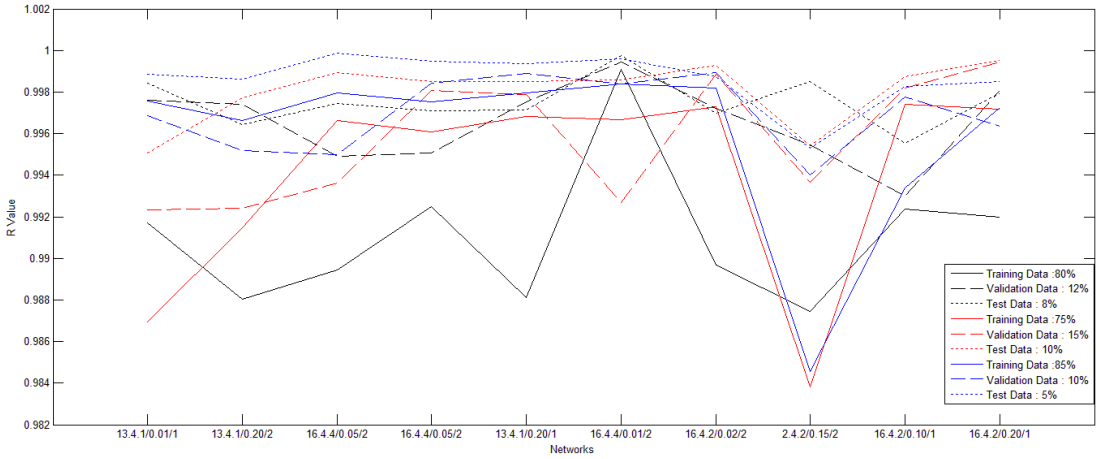


Fig. 20: R values of 10 networks for three different division settings

3.1.2 Discussion and Results

The trials, as already mentioned, were conducted in two stages. In the first stage, the most promising network configurations were determined (i.e. combinations of architecture, training algorithm and pairs of activation functions). In the second stage, the aforementioned combinations were tested in 12 pairs of learning algorithms and rates (Fig. 5), aiming to obtain improved results. The networks constructed in the second stage presented better validation MSE and R values (i.e. MSE and R values based on the validation set) compared to those in the first stage. For all 72 networks, the validation MSE value was lower by 52.9% to 96.9%, while the increase of the validation R value varied between 0.7% and 3.1%. Because R values were already very close to 1, the actual improvement may be better observed by looking at the 1-R values, which showed a reduction of 46.7% to 98.2% (Fig. 21).

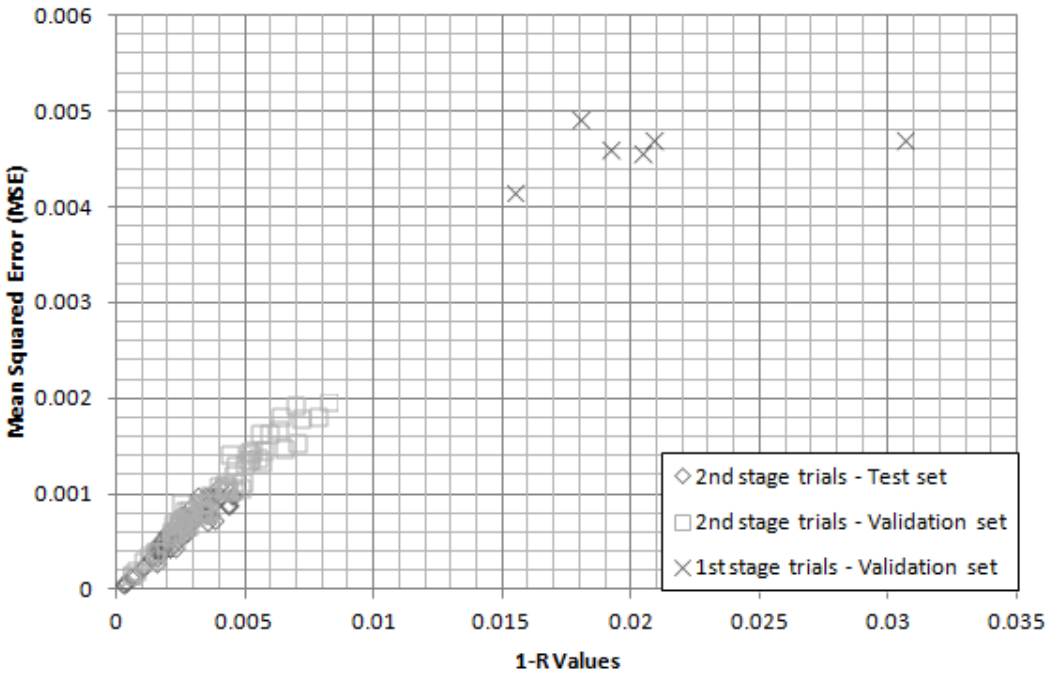


Fig. 21: MSE vs. 1-R values – second stage of trials

The networks were mainly evaluated using the validation and test MSE. These values should be lower than those obtained with the training data set and close to each other, with the test MSE value preferably lower than that of the validation. The MSE values for the 72 networks tested during the second stage of trials are illustrated in Fig. 22 where each network is defined by three characteristics; its code number, its learning algorithm, and its learning rate (see also Fig. 5).A network usually does not perform equally well with respect to all data sets. For example, for the network presenting the higher training MSE value, low validation and test MSE values were observed. This network is marked in Fig. 22 with an open circle and has code number 13.4.1, gradient descent weight and bias as learning algorithm and 0.05 as learning rate. However, the MSE values were not the only criterion taken into account during the networks' evaluation.

A major issue in ANNs is overfitting. A network may present low error during training, while failing to generalize to unknown inputs. The generalization ability of the second stage networks was evaluated using data from five MARAD hullforms listed in Table 5. It is worth noting that data from these hulls was not included in the data sets used for the training and evaluation of the networks. More importantly, for two of them (ships M and O) their geometric characteristics (i.e. their combination of C_B and B/T values) exceeded the limits of the training data set. As expected, the networks presenting the lower MSE and higher R values for the validation and test datasets, where those also providing better predictions for hulls B, E and H, i.e. those that were within the limits of the training dataset. However, a different behavior was observed for the remaining hulls (M and O): a few networks with relatively poor performance indices (i.e. MSE and R values according to the validation and test datasets) provided better predictions for these two hulls.

Table 5: Geometric properties of five MARAD hullforms

Ship	B	E	H	M	O
L/B	6.0	5.0	6.5	6.5	5.0
B/T	3.00	3.00	3.00	3.75	3.75
C_B	0.875	0.850	0.850	0.875	0.875
L [m]	349.9	333.4	372.6	397.5	355.6
B [m]	58.3	60.6	57.3	61.2	64.7
T [m]	19.4	20.2	19.1	16.3	17.2
WSA [m^2]	31111	29356	31877	33984	32252

Four networks were finally selected for the prediction of the residual resistance coefficient, and subsequently the resistance of hullforms based on the MARAD Systematic Series. The first two networks (denoted as MLP1 and MLP2) showed the lowest validation and test MSE values. Their generalization ability to ships exceeding the training set's limits was poor, but they were selected because of their superior fitting performance inside the training set's range. The other two (denoted as MLP3

and MLP4) were selected because of their generalization ability beyond the limits of the training data, indicated in the case of the two hulls tested (ships M and O), while performing quite well inside it.

The architecture of the first neural network (MLP1) is illustrated in Fig. 23. It comprises one hidden layer with 16 hyperbolic tangent sigmoid neurons and a linear output. The Levenberg-Marquardt training algorithm was used, along with the gradient descent with weight and bias learning algorithm and 0.20 learning rate. The progression of the MSE value during training is presented in Fig. 24. The training stopped when a minimum value of MSE for the validation dataset was reached. This network was selected for its low validation MSE, reached in 105th epoch of the training process. The MSEs based on the training and validation sets are relatively higher than that based on the test set. After the training process was completed, scatter diagrams comparing the actual outputs of the network with the desired ones for all three datasets (i.e. training, validation, test) were generated (see Fig. 25). Larger deviations between the actual and the desired output values may be observed in the scatter diagram corresponding to the training set. This is an expected result, since the network uses this set for updating its parameters, while validation and test sets are used for the evaluation of its predictive ability. Such deviations are missing from the scatter diagrams corresponding to the validation and test data sets, which present an almost linear relationship between actual and target outputs with R values over 0.99 for both of them.

The architecture of the second network (MLP2) is composed of four hidden layers of 16, 12, 8 and 4 hyperbolic tangent neurons each and one linear output layer (Fig. 26). The same training algorithm as for MLP1 was used, but the gradient descent with momentum bias and weight learning algorithm and 0.15 learning rate were applied. The training stopped in the 43rd epoch. The lower MSE value was obtained by the test set (Fig. 27). The R values for this network were also very close to 1, with the one corresponding to the test set being the higher (Fig. 28).

The third of the selected networks (MLP3) consists of hyperbolic tangent neurons arranged in three hidden layers, containing 12, 8 and 4 neurons each and one output layer (Fig. 29). It was trained using the same training and learning algorithms as MLP2, but the learning rate was set equal to 0.20. The training stopped in the 26th epoch after reaching a validation MSE higher than that of the two first networks. The performance indices (i.e. MSE and R value) of the test dataset were once again the best compared to those of the other datasets (see Fig. 30 and Fig. 31).

The fourth network (MLP4) comprises one hidden layer with 8 hyperbolic sigmoid neurons and a linear output (Fig. 32). The same training and learning algorithms as those for MLP1 were used. The value of the learning rate was set to 0.05. The training stopped in the 26th epoch. The lower MSE value was obtained by the test set in this case, as well (Fig. 33). The R values for this network were also very close to 1, with that of the test set being the higher (Fig. 34).

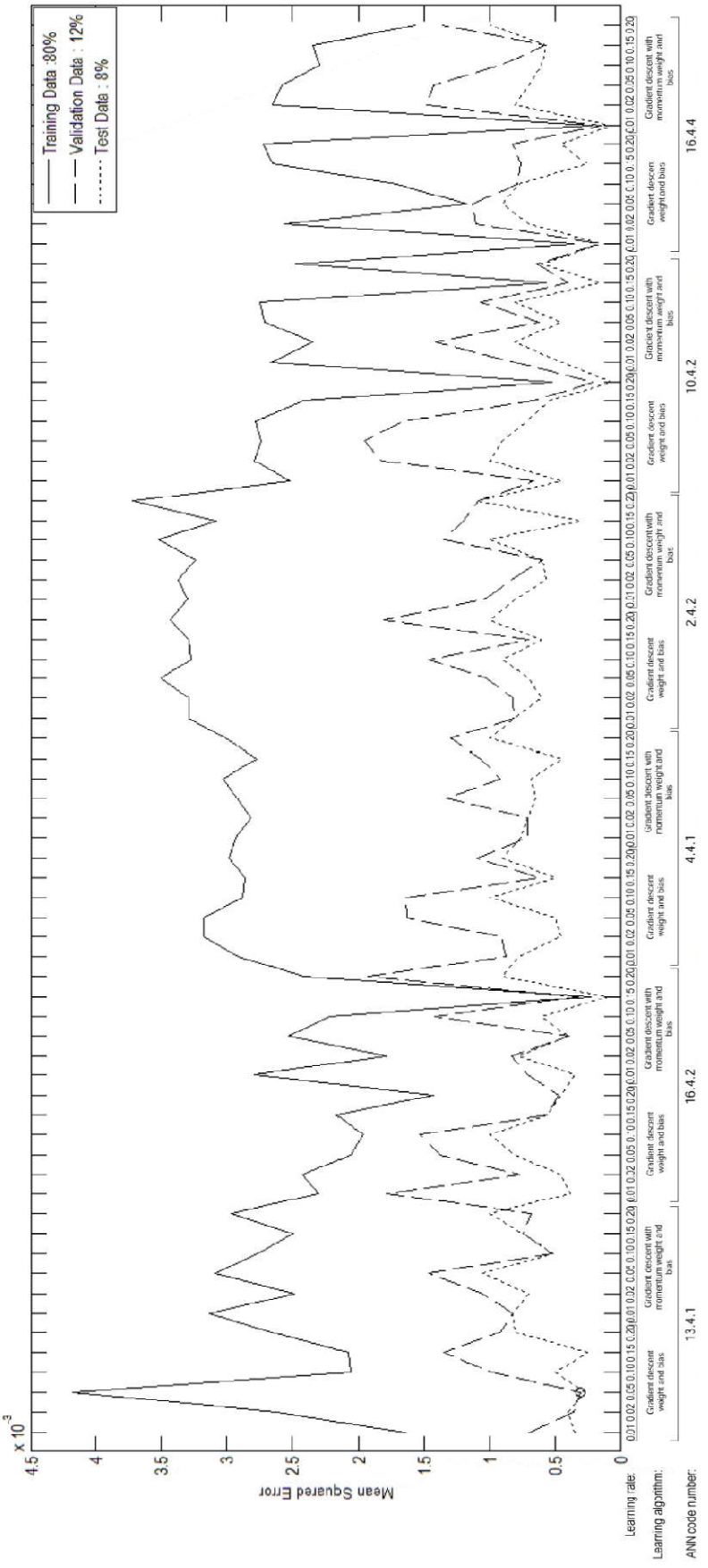


Fig. 22: MSE of ANNs – second stage of trials

MLP1

Training algorithm: Levenberg-Marquardt, Learning algorithm: Gradient descent weight and bias, Learning rate: 0.20

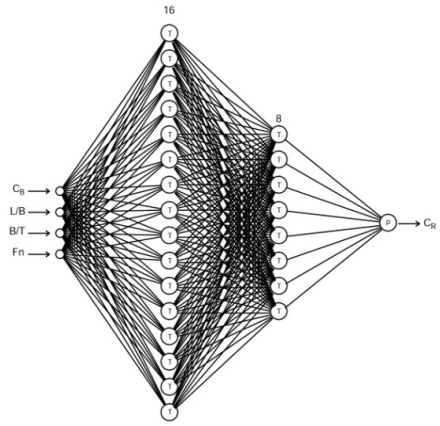


Fig. 23: Architecture of MLP1

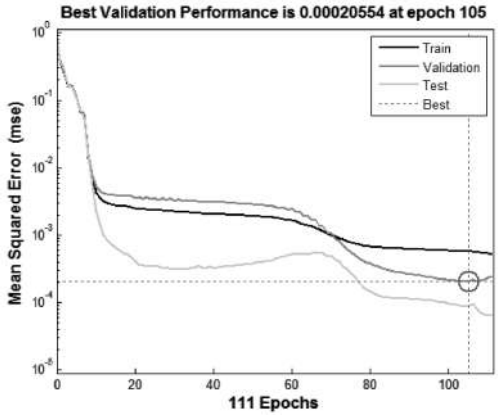


Fig. 24: MSE plot for MLP1 during training

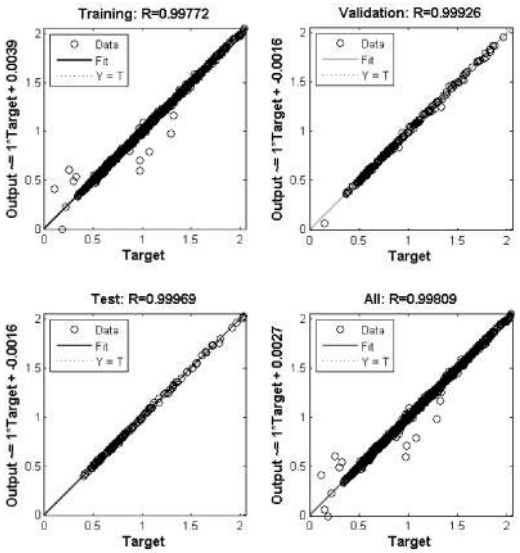


Fig. 25: Actual output vs. desired for MLP1 after training

MLP2

Training algorithm: Levenberg-Marquardt, Learning algorithm: Gradient descent with momentum weight and bias, Learning rate: 0.15

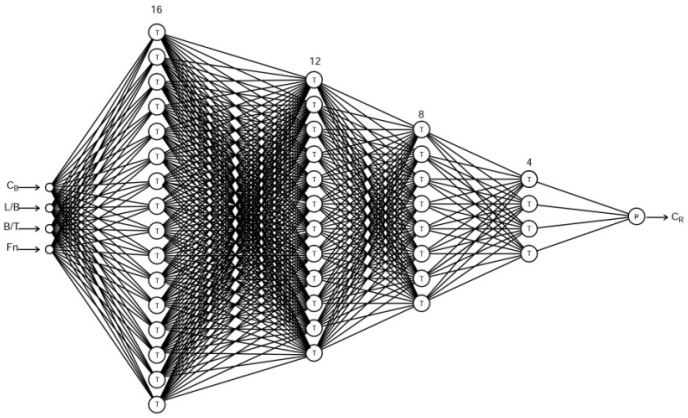


Fig. 26: Architecture of MLP2

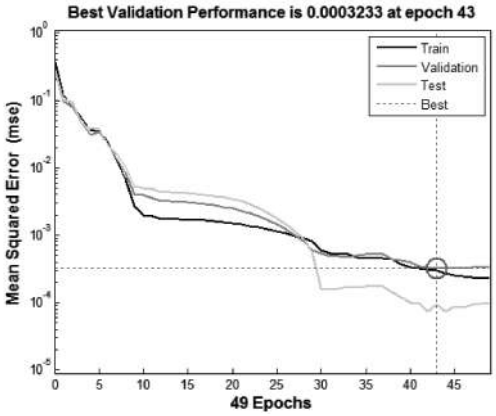


Fig. 27: MSE plot for MLP2 during training

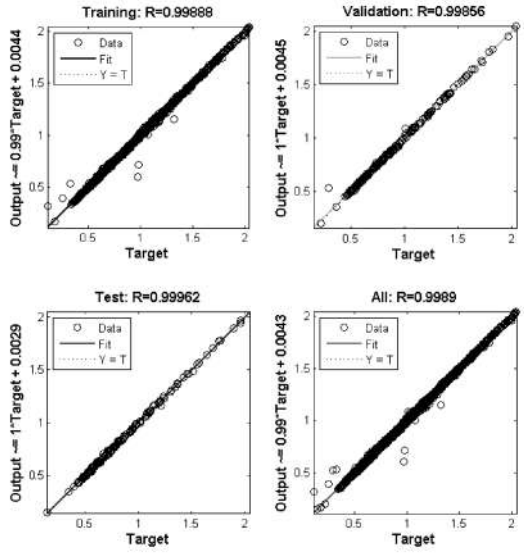


Fig. 28: Actual output vs. desired for MLP2 after training

MLP3

Training algorithm: Levenberg-Marquardt, Learning algorithm: Gradient descent with momentum weight and bias, Learning rate: 0.20

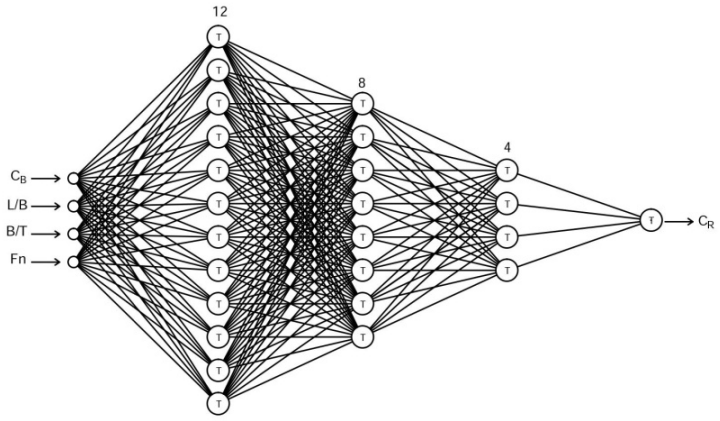


Fig. 29: Architecture of MLP3

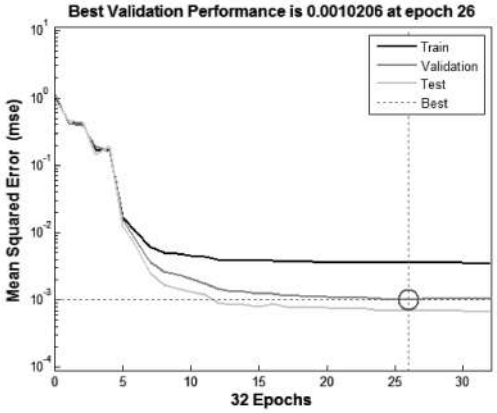


Fig. 30: MSE plot for MLP3 during training

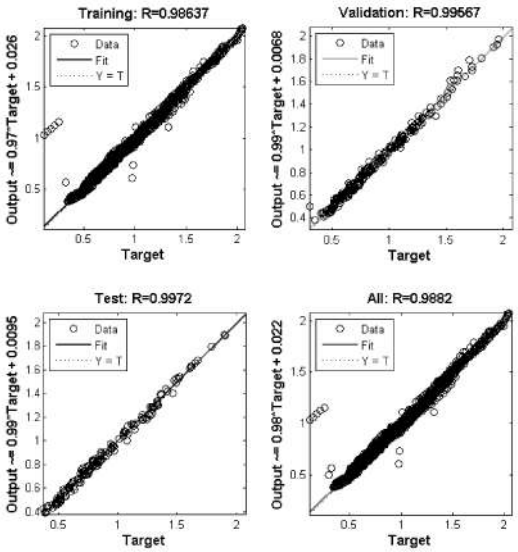


Fig. 31: Actual output vs. desired for MLP3 after training

MLP4

Training algorithm: Levenberg-Marquardt, Learning algorithm: Gradient descent weight and bias, Learning rate: 0.05

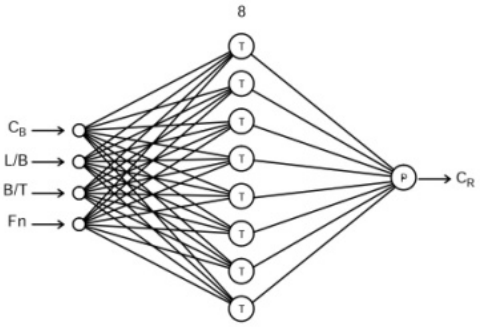


Fig. 32: Architecture of MLP4

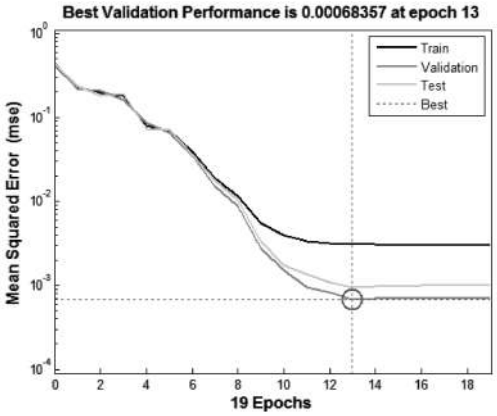


Fig. 33: MSE plot for MLP4 during training

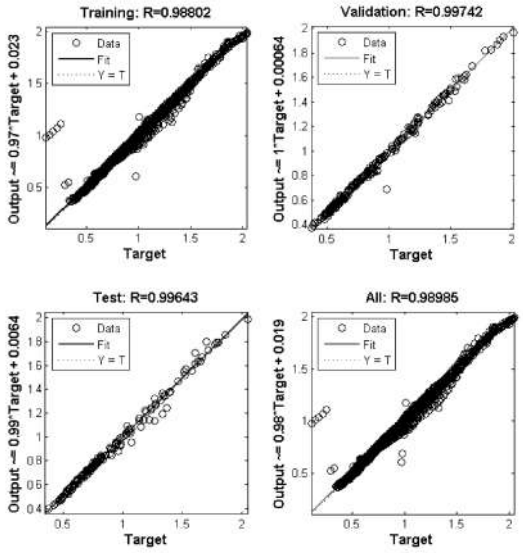


Fig. 34: Actual output vs. desired for MLP4 after training

The MSE values for MLP3 and MLP4 were higher than those for MLP1 and MLP2 for all three data sets. Nevertheless, MLP3 and MLP4 were selected because of their fairly good performance outside the data set, indicating possibly improved generalization ability. The comparison of the residual resistance coefficient (C_R) predicted by these four neural networks with the actual values according to the MARAD diagrams for the selected five ships is illustrated in Fig. 35 to Fig. 39 .

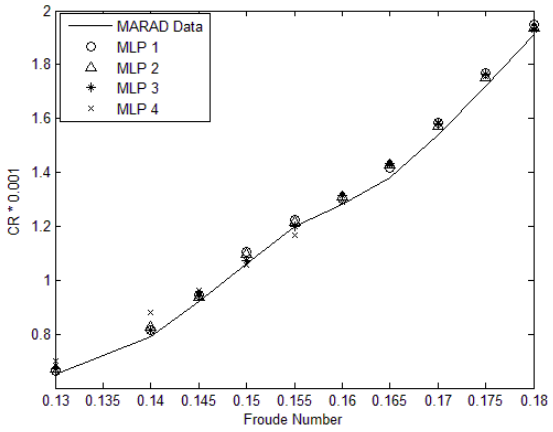


Fig. 35: Residual resistance coefficient for ship B

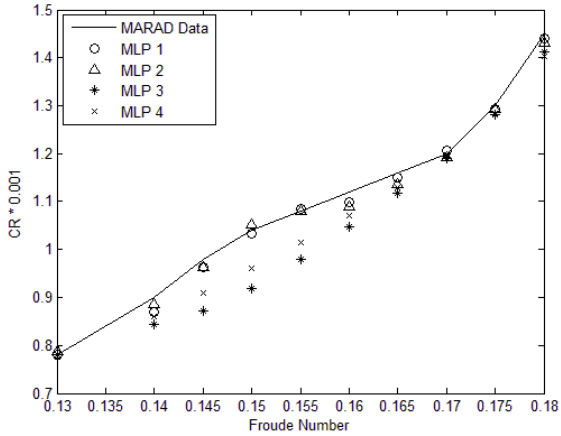


Fig. 36: Residual resistance coefficient for ship E

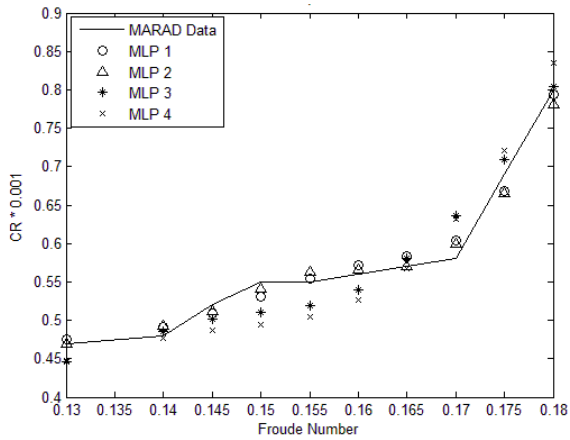


Fig. 37: Residual resistance coefficient for ship H

The predictions by the four networks, for the first three ships, i.e. the ones within the limits of the training set, are quite close to the real values. In particular, for ship B, the results derived by both networks present very small deviations from the target values (see Fig. 35). For the other two ships, i.e. E and H, the predictions by MLP1 and MLP2 are relatively better than those by the other two (see Fig. 36 and Fig. 37). The performance of MLP1 and MLP2 was expected to be superior for these three ships, because of its lower MSE and higher R values. As already mentioned, the geometric properties of ships M and O exceeded the range of the training data set. For these ships, the predictions by MLP3 and MLP4 were better than those by the first two (Fig. 38, Fig. 39). Although it's MSE and R values were not among the best, an improved generalization ability beyond the limits of the training data set is indicated by the obtained results.

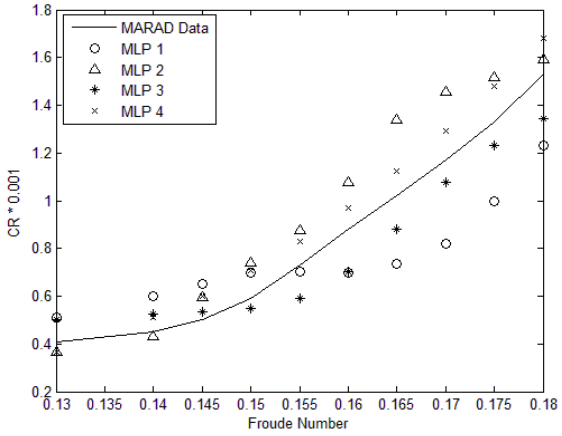


Fig. 38: Residual resistance coefficient for ship M

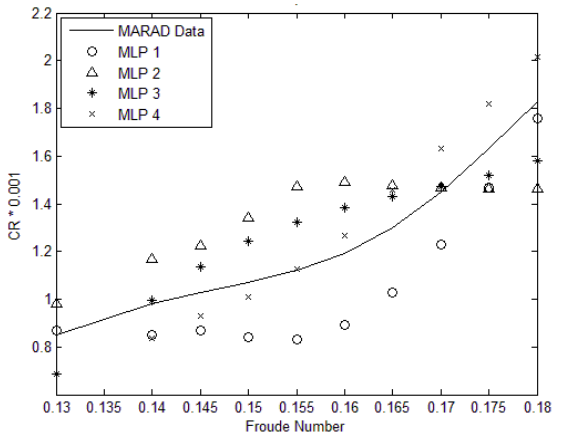


Fig. 39: Residual resistance coefficient for ship O

The estimated C_R values were used for the evaluation of total resistance of the five ships based on procedure described in Chapter 1.3. The results from the calculation of the total resistance based on the experimental C_R values and those estimated by the neural networks for the five hullforms are compared in Table 6 to Table 10 as well as in Fig. 40 to Fig. 44. The predictions by MLP1 and MLP2 for the first three ships with characteristics within the limits of the training dataset (ships B, E, H) present

very low deviations from the experimental values, in the order of 0.1% to 1.5% for both networks. The corresponding deviations of the predictions by MLP3 and MLP4 for the same ships ranged from 0.1% to 4.7% and to 3.8% respectively. For the other two ships exceeding the limits of the dataset, the deviations of the predictions obtained with MLP1 and MLP2 were quite larger. These of MLP1 ranged between 1.2% to 13.3% for ship M and 0.8% to 11.1% for ship O, while of MLP2 were 0.9% to 12.9% for ship M and 0.6% to 13.5% for ship O. However, the predictions obtained with MLP3 and MLP4 were considerably better, with practically acceptable deviations. The predictions obtained with MLP3 had deviations from the experimental values in the order of 1.6% to 7.6% and 0.6% to 7.8% for ship M and ship O respectively. These of MLP4 ranged between 2.3% to 5.7% for ship M and 0.3% to 6.8% for ship O.

Table 6: Total resistance prediction of ship B

		MARAD	MLP1		MLP2		MLP3		MLP4	
Fn	v [kn]	R_T [kN]	R_T [kN]	diff. [%]	R_T [kN]	diff. [%]	R_T [kN]	diff. [%]	R_T [kN]	diff. [%]
0.130	14.8	2025.4	2038.2	-0.6	2046.7	-1.1	2044.6	-0.9	2072.5	-2.3
0.140	15.9	2486.3	2512.0	-1.0	2526.5	-1.6	2511.9	-1.0	2582.0	-3.8
0.145	16.5	2810.3	2841.6	-1.1	2831.6	-0.8	2833.7	-0.8	2859.2	-1.7
0.150	17.1	3173.3	3230.0	-1.8	3220.5	-1.5	3191.1	-0.6	3170.0	0.1
0.155	17.6	3565.7	3600.2	-1.0	3589.5	-0.7	3563.8	0.1	3520.8	1.3
0.160	18.2	3904.5	3937.1	-0.8	3946.6	-1.1	3950.9	-1.2	3917.5	-0.3
0.165	18.8	4294.3	4346.8	-1.2	4367.2	-1.7	4371.8	-1.8	4366.5	-1.7
0.170	19.4	4804.3	4874.2	-1.5	4854.5	-1.0	4867.0	-1.3	4873.6	-1.4
0.175	19.9	5385.4	5466.6	-1.5	5440.6	-1.0	5462.4	-1.4	5444.0	-1.1
0.180	20.5	6027.0	6093.7	-1.1	6073.2	-0.8	6057.2	-0.5	6081.5	-0.9

Table 7: Total resistance prediction of ship E

		MARAD	MLP1		MLP2		MLP3		MLP4	
Fn	v [kn]	R_T [kN]	R_T [kN]	diff. [%]	R_T [kN]	diff. [%]	R_T [kN]	diff. [%]	R_T [kN]	diff. [%]
0.130	14.0	1832.1	1832.2	0.0	1838.2	-0.3	1832.6	0.0	1828.6	0.2
0.140	15.1	2222.2	2195.5	1.2	2208.9	0.6	2170.8	2.3	2186.8	1.6
0.145	15.6	2455.9	2439.4	0.7	2438.5	0.7	2351.9	4.2	2387.4	2.8
0.150	16.1	2684.7	2677.6	0.3	2697.1	-0.5	2557.7	4.7	2602.8	3.0
0.155	16.7	2905.1	2910.7	-0.2	2904.7	0.0	2793.1	3.9	2833.2	2.5
0.160	17.2	3136.7	3110.6	0.8	3101.9	1.1	3051.1	2.7	3078.8	1.8
0.165	17.8	3379.9	3367.9	0.4	3349.2	0.9	3325.2	1.6	3342.3	1.1
0.170	18.3	3634.8	3642.6	-0.2	3625.4	0.3	3621.3	0.4	3632.2	0.1
0.175	18.8	3986.7	3976.9	0.2	3975.8	0.3	3961.8	0.6	3963.5	0.6
0.180	19.4	4435.6	4420.5	0.3	4407.7	0.6	4378.1	1.3	4366.6	1.6
0.185	19.9	4915.8	4898.9	0.3	4899.7	0.3	4879.6	0.7	4881.2	0.7
0.190	20.5	5444.9	5435.8	0.2	5419.7	0.5	5421.2	0.4	5424.5	0.4

Table 8: Total resistance prediction of ship H

		MARAD	MLP1		MLP2		MLP3		MLP4	
Fn	v [kn]	R_T [kN]	R_T [kN]	diff. [%]	R_T [kN]	diff. [%]	R_T [kN]	diff. [%]	R_T [kN]	diff. [%]
0.130	15.3	2012.7	2018.6	-0.3	2012.5	0.0	1988.4	1.2	1990.3	1.1
0.140	16.5	2332.1	2345.6	-0.6	2347.1	-0.6	2337.7	-0.2	2328.7	0.1
0.145	17.0	2544.9	2531.2	0.5	2535.5	0.4	2521.8	0.9	2503.4	1.6
0.150	17.6	2756.6	2731.0	0.9	2745.2	0.4	2702.6	2.0	2682.0	2.7
0.155	18.2	2936.0	2941.6	-0.2	2955.4	-0.7	2890.8	1.5	2871.0	2.2
0.160	18.8	3136.2	3155.0	-0.6	3145.4	-0.3	3105.5	1.0	3084.4	1.7
0.165	19.4	3343.8	3366.1	-0.7	3344.9	0.0	3359.9	-0.5	3338.9	0.1
0.170	20.0	3558.8	3599.3	-1.1	3591.5	-0.9	3656.5	-2.7	3648.4	-2.5
0.175	20.6	3964.5	3925.2	1.0	3920.2	1.1	3998.4	-0.9	4022.0	-1.5
0.180	21.2	4398.9	4388.7	0.2	4360.8	0.9	4405.9	-0.2	4465.7	-1.5
0.185	21.7	4945.0	4941.4	0.1	4871.8	1.5	4914.5	0.6	4983.9	-0.8
0.190	22.3	5552.4	5516.4	0.6	5467.3	1.5	5542.9	0.2	5580.4	-0.5

Table 9: Total resistance prediction of ship M

		MARAD	MLP1		MLP2		MLP3		MLP4	
Fn	v [kn]	R_T [kN]	R_T [kN]	diff. [%]	R_T [kN]	diff. [%]	R_T [kN]	diff. [%]	R_T [kN]	diff. [%]
0.130	15.8	2202.5	2316.6	-5.2	2149.9	2.4	2309.7	-4.9	2151.2	2.3
0.140	17.0	2592.1	2789.3	-7.6	2568.8	0.9	2694.7	-4.0	2672.5	-3.1
0.145	17.6	2844.2	3062.0	-7.7	2981.1	-4.8	2890.6	-1.6	2989.6	-5.1
0.150	18.2	3173.3	3336.1	-5.1	3405.8	-7.3	3107.7	2.1	3353.3	-5.7
0.155	18.8	3608.8	3567.0	1.2	3849.5	-6.7	3383.8	6.2	3771.1	-4.5
0.160	19.4	4097.9	3780.3	7.8	4443.8	-8.4	3787.4	7.6	4250.6	-3.7
0.165	20.0	4608.4	4080.7	11.5	5201.5	-12.9	4353.1	5.5	4798.9	-4.1
0.170	20.6	5177.8	4487.3	13.3	5743.6	-10.9	4992.4	3.6	5422.1	-4.7
0.175	21.2	5810.8	5115.8	12.0	6196.4	-6.6	5602.0	3.6	6124.3	-5.4
0.180	21.8	6578.7	5918.7	10.0	6716.6	-2.1	6171.3	6.2	6907.0	-5.0

Table 10: Total resistance prediction of ship O

		MARAD	MLP1		MLP2		MLP3		MLP4	
Fn	v [kn]	R_T [kN]	R_T [kN]	diff. [%]	R_T [kN]	diff. [%]	R_T [kN]	diff. [%]	R_T [kN]	diff. [%]
0.130	14.9	2325.3	2344.6	-0.8	2451.8	-5.4	2165.6	6.9	2166.4	6.8
0.140	16.1	2830.3	2684.6	5.1	3043.7	-7.5	2846.5	-0.6	2670.2	5.7
0.145	16.6	3089.9	2892.4	6.4	3327.1	-7.7	3215.8	-4.1	2968.7	3.9
0.150	17.2	3351.6	3051.4	9.0	3702.4	-10.5	3576.9	-6.7	3275.4	2.3
0.155	17.8	3640.8	3238.0	11.1	4130.6	-13.5	3924.6	-7.8	3651.2	-0.3
0.160	18.4	3975.5	3537.3	11.0	4418.9	-11.2	4261.9	-7.2	4089.5	-2.9
0.165	18.9	4393.1	3968.8	9.7	4673.9	-6.4	4597.1	-4.6	4642.3	-5.7
0.170	19.5	4905.8	4535.7	7.5	4937.5	-0.6	4943.0	-0.8	5211.0	-6.2
0.175	20.1	5508.8	5219.8	5.2	5215.7	5.3	5314.8	3.5	5840.4	-6.0
0.180	20.7	6194.0	6059.7	2.2	5510.4	11.0	5728.8	7.5	6540.0	-5.6

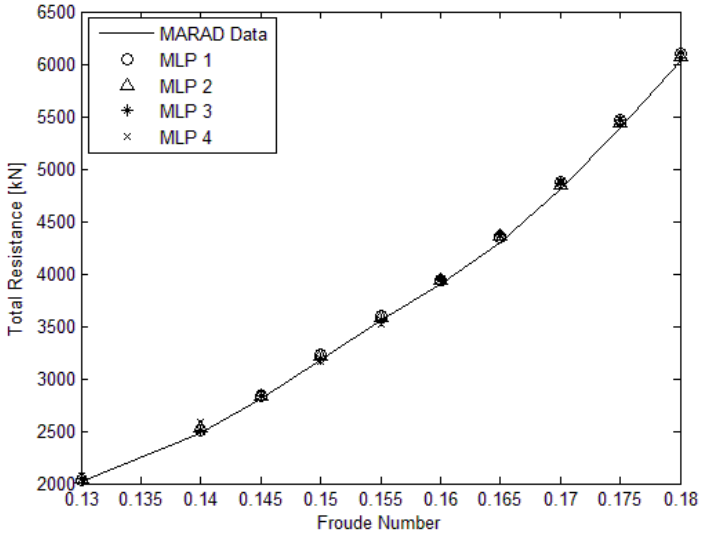


Fig. 40: Total resistance for ship B

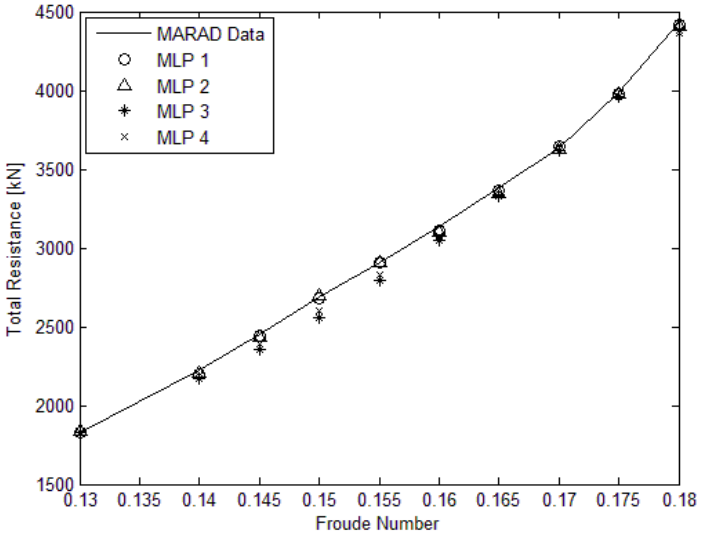


Fig. 41: Total resistance for ship E

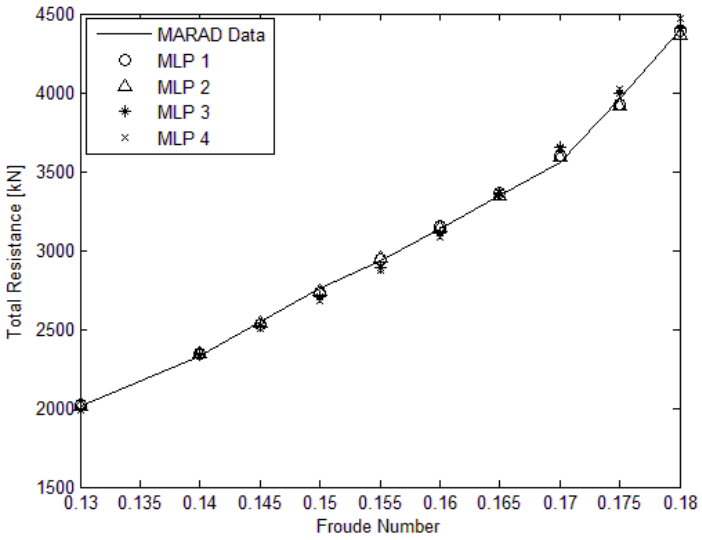


Fig. 42: Total resistance for ship H

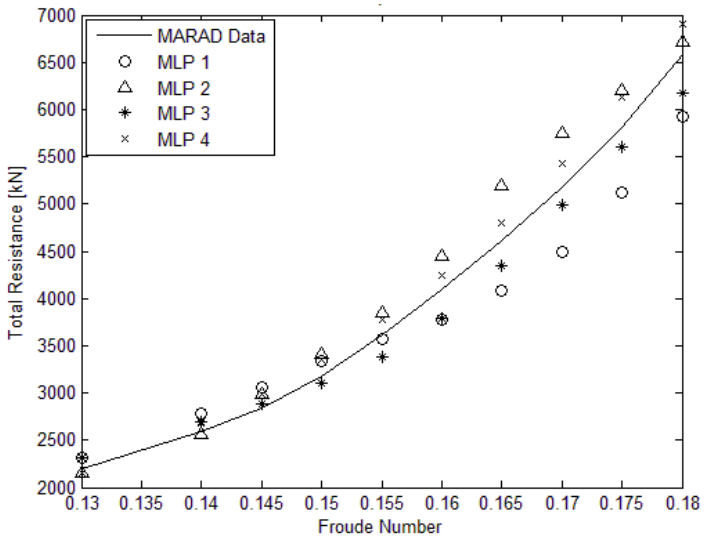


Fig. 43: Total resistance for ship M

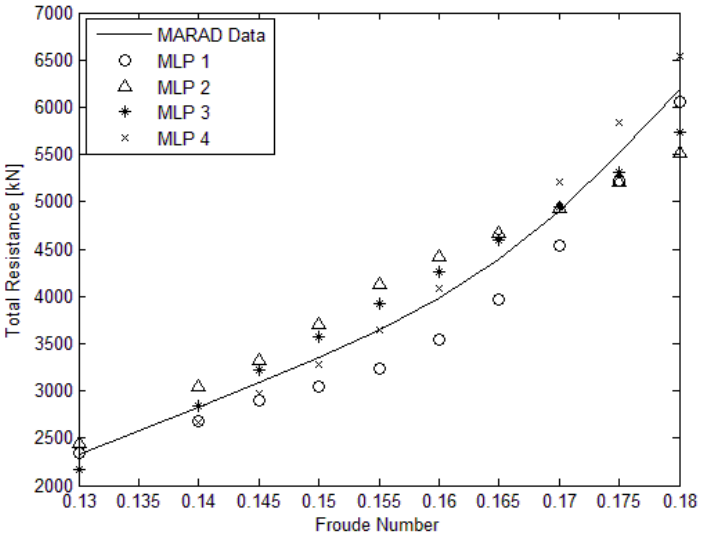


Fig. 44: Total resistance for ship O

3.2 Radial Basis Function Neural Networks

3.2.1 Implementation

The performance of RBF neural networks is affected by the number of neurons that comprise the hidden layer, the selected activation function and learning method. The radial basis activation function that was selected for this study is the Gaussian, which has the standard deviation and its center as parameters, while the K-means algorithm was chosen as the learning method. In order to determine the network that suits better to the problem at hand, several trials have been conducted, changing the number of neurons and the sigma values, i.e. the square root of the standard deviation. The networks were evaluated based on their MSE and R values of the validation set. The

trials were conducted in three stages. In the first stage, the available data was separated into mutually exclusive training (80%) and validation (20%) sets. The configurations of the tested networks during this stage are presented in Table 11. Each network is given a serial number, that corresponds to a combination of a number of hidden neurons and a sigma value. For example, the networks with serial number 3 and 15 have 10 and 70 hidden neurons respectively and a sigma value equal to 1. The networks with serial number 47 and 77 have 20 hidden neurons and a sigma value equal to 0.25 and 0.1 respectively. Observations during these trials regarding the performance indices affected the tested combinations, hence the difference in the number of tested configurations per sigma value. The best MSE and R values per sigma value are also included in this table. The MSE values regarding both datasets and all networks are illustrated in graphical form in Fig. 45.

During trials a relation between the sigma value and the number of the hidden neurons was observed. Assuming a series of networks with different number of hidden neurons and the same sigma value, the MSE value declines as the number of the hidden neurons increases, and for a specific number of neurons it stabilizes. This limit decreases as the sigma value increases. For example, as it can be seen in Fig. 45, for sigma value equal to 0.5 the limit is at 30 neurons, while for sigma equal to 1 is at 50 neurons.

.

Table 11 : Networks of the first stage of trials

Serial No.	No of Hidden neurons	Sigma	Training set		Validation set	
			min MSE	max R	min MSE	max R
1-2	25/30	1.50	0.0160	0.8734	0.0181	0.8700
3-20	10/15/20/25/30/35/40/45/ 50/55/60/65/70/80/90/100/150/ 200/500	1.00	0.0148	0.8834	0.0171	0.8775
21-38	10/15/20/25/30/35/40/45/ 50/55/60/65/70/80/90/100/150/ 200/500	0.50	0.0133	0.8949	0.0152	0.8907
39-40	200/250	0.35	0.0110	0.9133	0.0139	0.9002
41-42	200/250	0.30	0.0068	0.9460	0.0104	0.9252
43-44	200/250	0.27	0.0049	0.9611	0.0079	0.9436
45-66	10/15/20/25/30/35/40/45/ 50/55/60/70/80/90/100/150/200 /250/300/350/400/450	0.25	0.0023	0.9745	0.0056	0.9597
67-68	200/250	0.22	0.0032	0.9747	0.0056	0.9599
69-70	200/250	0.17	0.0029	0.9773	0.0053	0.9618
71-72	200/250	0.15	0.0029	0.9818	0.0054	0.9612
73-74	200/250	0.12	0.0027	0.9788	0.0062	0.9554
75-96	10/15/20/25/30/35/40/45/ 50/55/60/70/80/90/100/150/200 /250/300/350/400/450	0.10	0.0025	0.9807	0.0063	0.9547
97-103	250/300/350/400/450/500/ 550	0.05	0.0023	0.9818	0.0087	0.9378

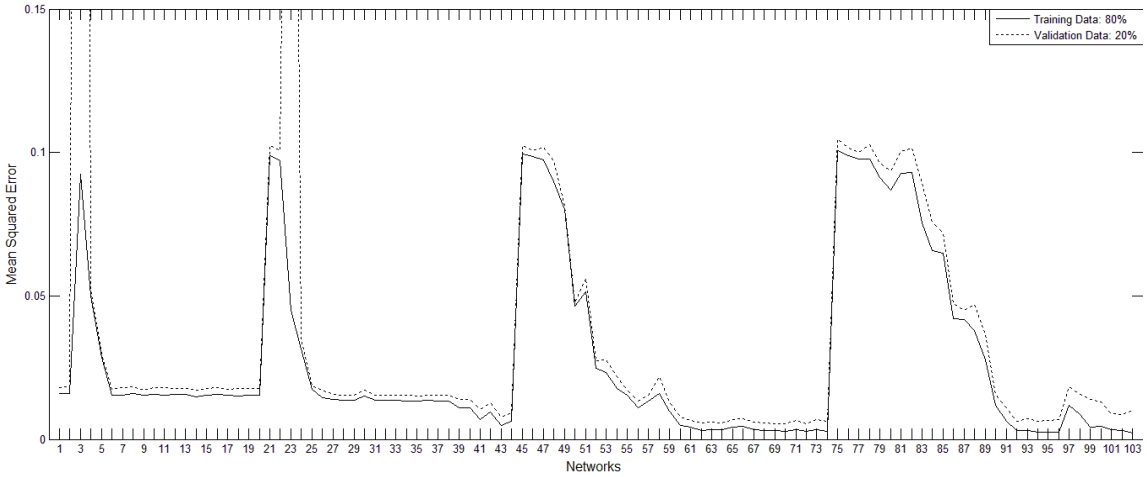


Fig. 45: MSE values of networks – first stage of trials

In the second stage, the six best performing networks of the previews stage were retrained with different division ratio of the available data along with some new configurations. The 85% of the available data were assigned to training set and the 15% to the validation set. The networks that were retrained retain their serial number from the 1st stage with the addition of the letter b. For the additional networks capital letters from A to J are used as code names. The configurations of the networks and their performance characteristics are presented in Table 12.

Table 12 : Networks of the second stage of trials

Code	No of Hidden neurons	Sigma	Training set		Validation set	
			MSE	R	MSE	R
60b	250	0.25	0.0042	0.9757	0.0028	0.9744
62b	350	0.25	0.004	0.9768	0.0027	0.9745
92b	250	0.10	0.0062	0.9637	0.0079	0.9264
94b	350	0.10	0.0034	0.9801	0.0021	0.9802
95b	400	0.10	0.0032	0.9814	0.0018	0.9836
96b	450	0.10	0.0032	0.9815	0.0017	0.9845
A	350	0.35	0.0115	0.9328	0.0100	0.9068
B	350	0.30	0.0113	0.934	0.0101	0.9058
C	350	0.27	0.0084	0.9507	0.0070	0.9353
D	350	0.22	0.0038	0.9775	0.0022	0.9797
E	350	0.17	0.0036	0.9793	0.0021	0.9801
F	350	0.15	0.0034	0.9801	0.0020	0.9817
G	350	0.12	0.0033	0.9805	0.0017	0.9837
H	500	0.10	0.0032	0.9815	0.0017	0.9838
I	550	0.10	0.0031	0.9816	0.0016	0.9849
J	600	0.10	0.0032	0.9815	0.0017	0.9838

The 20 networks from both stages that presented the lower validation MSE were selected for further testing. Taking into account that RBF neural networks are also susceptible to overfitting, the selected networks were also evaluated using data from

thirteen MARAD hullforms (see Table 13). The first five (i.e. ships B, E, H, M and O) belong to the sixteen hulls that were used in experiments, but their data were not included in the training and evaluation data sets. More importantly, as already mentioned, for two of them (ships M and O) their geometric characteristics exceeded the limits of the training data set. The other 8 hulls (i.e. S1 to S8) were created by selecting their geometric properties within the limits of the MARAD series. Their wetted surface was calculated with the use of three diagrams illustrated in ([5]) providing the wetted surface coefficient (C_S) for various values of L/B, B/T and C_B for MARAD-type hullforms. The C_R values for these ships were taken from the diagrams presented in Fig. 5 to

Fig. 9 using linear interpolation when required and their displacement was assumed 350000 tons and equal to that of the 16 MARAD hullforms. The geometric characteristics of the aforementioned ships are presented in the following table.

Table 13: Geometric properties of hullforms used for testing

Ship	L/B	B/T	C_B	WSA [m^2]
B	6	3	0.875	31111
E	5	3	0.850	29356
H	6.5	3	0.850	31877
M	6.5	3.75	0.875	33984
O	5	3.75	0.875	32252
S1	5	3	0.825	30224
S2	5	4.5	0.825	33493
S3	5	3.75	0.825	30224
S4	5	4.5	0.845	31775
S5	5	3.75	0.845	31346
S6	5	3	0.845	28931
S7	5.5	3	0.865	29363
S8	6	3	0.865	29996

As expected, the selected networks performed well for all the ships within the limits of the training dataset (i.e. all ships except M and O). The six networks that provided the best predictions were selected. In an attempt to improve their performance, a new training algorithm was generated and a third stage of trials was carried out. In the third stage of these trials, the six selected networks were retrained using a modified K-means algorithm, and six new networks were developed from this process.

This algorithm was inspired by a training method mentioned in Section 2.4.1 that adds neurons to the hidden layer until an error function is minimized. The new algorithm adds neurons to existent networks that have presented good performance indices. The centers of the RB functions of the new neurons are placed at the positions where the higher errors were observed. As long as the training MSE error of the suggested network is lower than the initial, the algorithm adds one neuron per step, searching for an even lower training MSE value in the interim. This algorithm has been tested in a

series of cases and the derived networks were compared with the original networks as well as with others of the same characteristics trained with the K-means algorithm in order to define its effectiveness and the results were promising. From this algorithm six networks were obtained and their characteristics along with their origin network are given in Table 14. The networks constructed in the third stage presented better validation MSE and R values compared to their origin networks. The decrease of MSE value was between 1% and 14%. Because R values were already very close to 1, the actual improvement may be better observed by looking at the 1-R values, which showed the same reduction (see Table 14). The networks of this stage were further evaluated based on their prediction of C_R for the ships in Table 13.

Table 14 : Networks' properties - third stage of trials

Origin network	Code	No of Hidden neurons	Sigma	Training set		Validation set		Improvement	
				MSE	R	MSE	R	MSE	1-R
39	i	215	0.35	0.0170	0.9152	0.0136	0.9024	2%	2%
40	ii	325	0.35	0.0109	0.9138	0.0137	0.9015	1%	1%
61	iii	213	0.25	0.0035	0.9720	0.0060	0.9572	10%	10%
94b	iv	354	0.10	0.0033	0.9808	0.0018	0.9829	14%	14%
G	v	357	0.12	0.0032	0.9810	0.0016	0.9850	6%	8%
J	vi	606	0.10	0.0032	0.9816	0.0016	0.9848	6%	6%

3.2.2 Discussion and Results

The testing of the RBF neural networks, as already mentioned, was conducted in three stages and 125 RBF neural networks in total were developed. In the first stage, the division ratio of the available data to training and validation set was 80% and 20% respectively and 103 RBF neural networks were developed. In the second stage, 85% of the data were assigned to the training set, 15% to the validation set and 16 networks were created. In the third stage, six new networks were obtained applying a modified K-means training algorithm to the best six networks from the two previous stages and two of them were finally chosen; the networks with code ii and vi (see Table 14). These networks herein denoted as RBF1 and RBF2 respectively were selected because they presented a good fit inside the dataset and the lower errors outside its limits. The first network (RBF1) has 213 hidden neurons, sigma parameter equal to 0.25, and its training set accounted for 80% of the available dataset. The second network (RBF2) has 606 hidden neurons, the value of the sigma parameter is 0.1 and 85% of the data were assigned to its training set. They were used for estimating the C_R and subsequently the resistance of the ships mentioned in Table 13 for a range of Froude numbers. The comparison of the residual resistance coefficient (C_R) predicted by these two neural networks with the actual values according to the MARAD diagrams for the first five ships is illustrated in Fig. 46 to Fig. 50.

The predictions for the first three ships, i.e. the MARAD hulls within the limits of the training set, are quite close to the real values. In particular, for ship B, the results derived by both networks present very small deviations from the target values (see

Fig. 46). For the other two ships, i.e. E and H, the predictions by both networks are relatively poorer, but they are following the trend of the actual values (see Fig. 47 and Fig. 48). As already mentioned, the geometric properties of ships M and O exceeded the range of the training data set. For these ships, the predictions by the two RBF networks are poor, but they are presented for the sake of completeness (Fig. 49, Fig. 50).

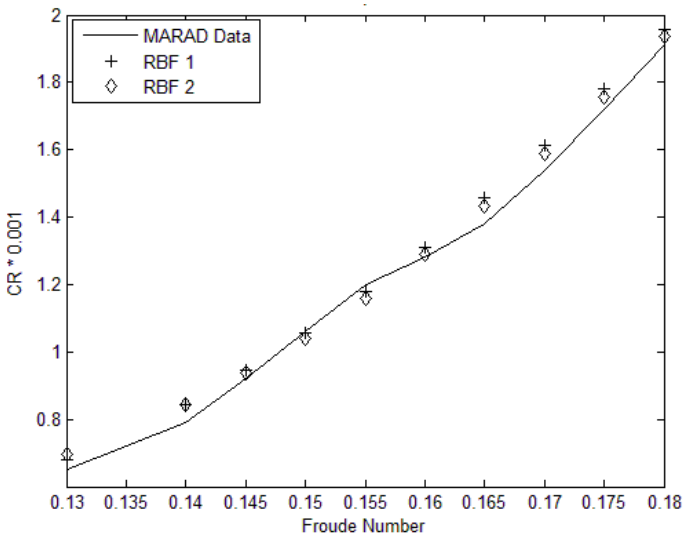


Fig. 46: Residual resistance coefficient for ship B

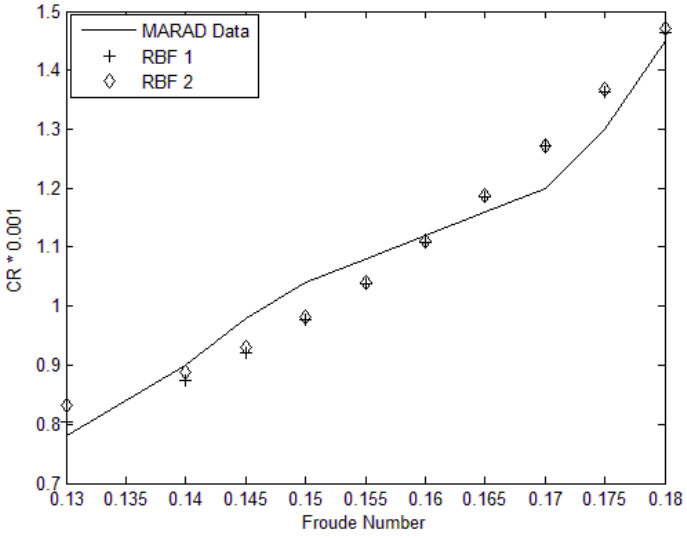


Fig. 47: Residual resistance coefficient for ship E

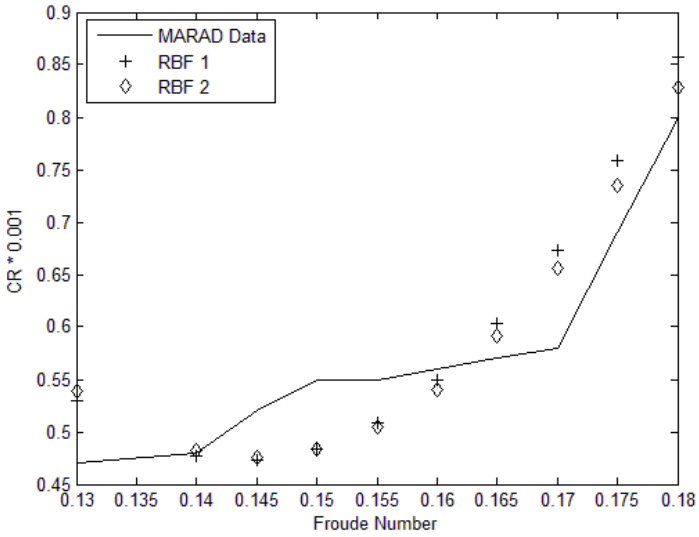


Fig. 48: Residual resistance coefficient for ship H

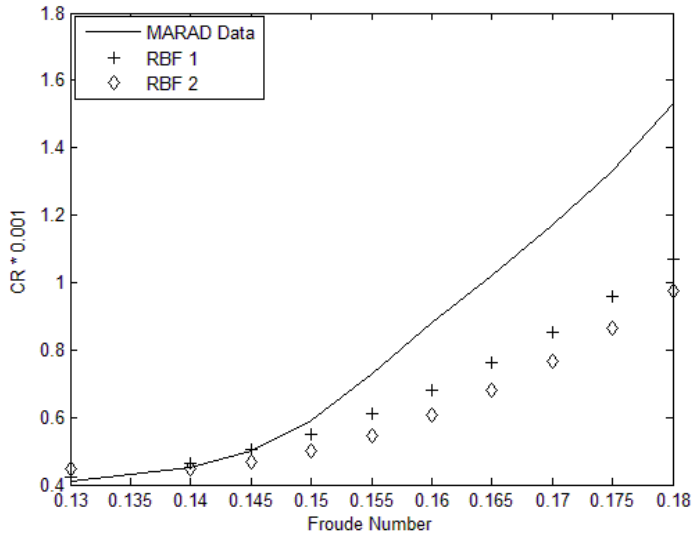


Fig. 49: Residual resistance coefficient for ship M

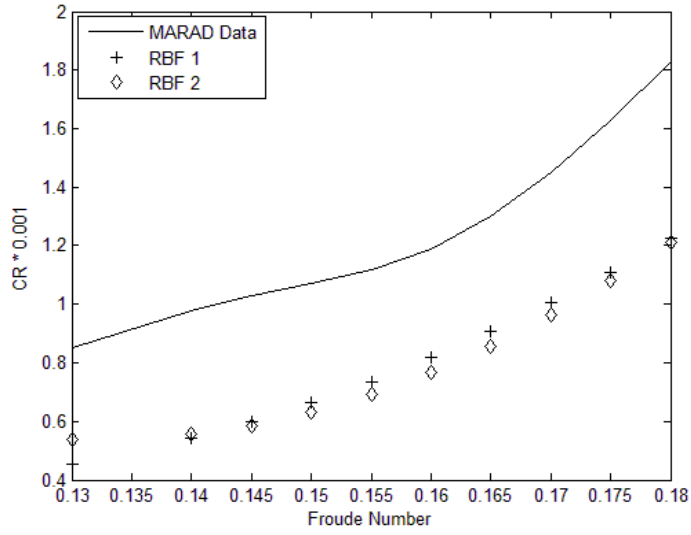


Fig. 50: Residual resistance coefficient for ship O

A comparison of the results for the total resistance of these five ships based on the MARAD data and the network predictions for the C_R coefficient is presented in Fig. 51 to Fig. 55 as well as in Table 16 to Table 19. The predictions for the first three ships with characteristics within the limits of the training dataset (ships B, E, H) present acceptable deviations from the experimental values, in the order of 0.2% to 4.5% for both networks. For the other two ships, exceeding the limits of the dataset, the deviations of the predictions obtained with RBF1 and RBF2 were quite larger. These of RBF1 ranged between 0.1% to 15.4% for ship M and 13.9% to 18.3% for ship O, while of RBF2 were 0.2% to 18.5% for ship M and 13% to 18.6% for ship O.

The predictions of C_R values and resistance of the ships S1 to S8 are presented in tabular form in Table 20 to Table 27. The deviations of the predictions of RBF1 for these ships ranged between 0.0% and 2.9%, while those of RBF2 were from 0.2% to 1.4%.

Table 15: Total resistance prediction of ship B

		MARAD	RBF1		RBF2	
Fn	v [kn]	R_T [kN]	R_T [kN]	diff. [%]	R_T [kN]	diff. [%]
0.130	14.8	2025.4	2052.3	-1.3	2068.9	-2.1
0.140	15.9	2486.3	2545.2	-2.4	2543.8	-2.3
0.145	16.5	2810.3	2838.7	-1.0	2829.0	-0.7
0.150	17.1	3173.3	3168.1	0.2	3150.8	0.7
0.155	17.6	3565.7	3537.0	0.8	3513.1	1.5
0.160	18.2	3904.5	3949.3	-1.1	3919.7	-0.4
0.165	18.8	4294.3	4408.9	-2.7	4375.0	-1.9
0.170	19.4	4804.3	4920.0	-2.4	4883.1	-1.6
0.175	19.9	5385.4	5486.8	-1.9	5448.5	-1.2
0.180	20.5	6027.0	6114.0	-1.4	6075.8	-0.8

Table 16: Total resistance prediction of ship E

		MARAD	RBF1		RBF2	
Fn	v [kn]	R_T [kN]	R_T [kN]	diff. [%]	R_T [kN]	diff. [%]
0.130	14.0	1832.1	1851.7	-1.1	1873.3	-2.2
0.140	15.1	2222.2	2199.9	1.0	2211.8	0.5
0.145	15.6	2455.9	2399.5	2.3	2407.5	2.0
0.150	16.1	2684.7	2618.6	2.5	2623.4	2.3
0.155	16.7	2905.1	2859.0	1.6	2861.6	1.5
0.160	17.2	3136.7	3122.8	0.4	3124.2	0.4
0.165	17.8	3379.9	3412.2	-1.0	3413.7	-1.0
0.170	18.3	3634.8	3729.5	-2.6	3732.2	-2.7
0.175	18.8	3986.7	4076.9	-2.3	4082.2	-2.4
0.180	19.4	4435.6	4457.0	-0.5	4466.1	-0.7
0.185	19.9	4915.8	4872.4	0.9	4886.6	0.6
0.190	20.5	5444.9	5325.8	2.2	5346.3	1.8

Table 17: Total resistance prediction of ship H

		MARAD	RBF1		RBF2	
Fn	v [kn]	R_T [kN]	R_T [kN]	diff. [%]	R_T [kN]	diff. [%]
0.130	15.3	2012.7	2073.0	-3.0	2080.9	-3.4
0.140	16.5	2332.1	2328.2	0.2	2334.9	-0.1
0.145	17.0	2544.9	2485.2	2.3	2489.1	2.2
0.150	17.6	2756.6	2666.6	3.3	2666.5	3.3
0.155	18.2	2936.0	2876.5	2.0	2871.0	2.2
0.160	18.8	3136.2	3119.1	0.5	3106.7	0.9
0.165	19.4	3343.8	3398.7	-1.6	3378.0	-1.0
0.170	20.0	3558.8	3720.3	-4.5	3689.6	-3.7
0.175	20.6	3964.5	4088.8	-3.1	4046.3	-2.1
0.180	21.2	4398.9	4509.4	-2.5	4453.2	-1.2
0.185	21.7	4945.0	4987.6	-0.9	4915.8	0.6
0.190	22.3	5552.4	5529.2	0.4	5439.8	2.0

Table 18: Total resistance prediction of ship M

		MARAD	RBF1		RBF2	
Fn	v [kn]	R_T [kN]	R_T [kN]	diff. [%]	R_T [kN]	diff. [%]
0.130	15.8	2202.5	2218.0	-0.7	2245.9	-2.0
0.140	17.0	2592.1	2612.1	-0.8	2587.5	0.2
0.145	17.6	2844.2	2847.5	-0.1	2795.9	1.7
0.150	18.2	3173.3	3113.1	1.9	3034.6	4.4
0.155	18.8	3608.8	3412.6	5.4	3307.9	8.3
0.160	19.4	4097.9	3750.1	8.5	3620.4	11.7
0.165	20.0	4608.4	4129.8	10.4	3976.7	13.7
0.170	20.6	5177.8	4556.1	12.0	4382.0	15.4
0.175	21.2	5810.8	5033.7	13.4	4841.4	16.7
0.180	21.8	6578.7	5567.5	15.4	5360.5	18.5

Table 19: Total resistance prediction of ship O

		MARAD	RBF1		RBF2	
Fn	v [kn]	R_T [kN]	R_T [kN]	diff. [%]	R_T [kN]	diff. [%]
0.130	14.9	2325.3	1939.7	16.6	2023.0	13.0
0.140	16.1	2830.3	2334.7	17.5	2349.6	17.0
0.145	16.6	3089.9	2565.8	17.0	2550.9	17.4
0.150	17.2	3351.6	2822.6	15.8	2782.4	17.0
0.155	17.8	3640.8	3107.7	14.6	3048.1	16.3
0.160	18.4	3975.5	3424.1	13.9	3352.2	15.7
0.165	18.9	4393.1	3774.6	14.1	3699.0	15.8
0.170	19.5	4905.8	4162.5	15.2	4093.1	16.6
0.175	20.1	5508.8	4590.9	16.7	4539.0	17.6
0.180	20.7	6194.0	5063.2	18.3	5041.6	18.6

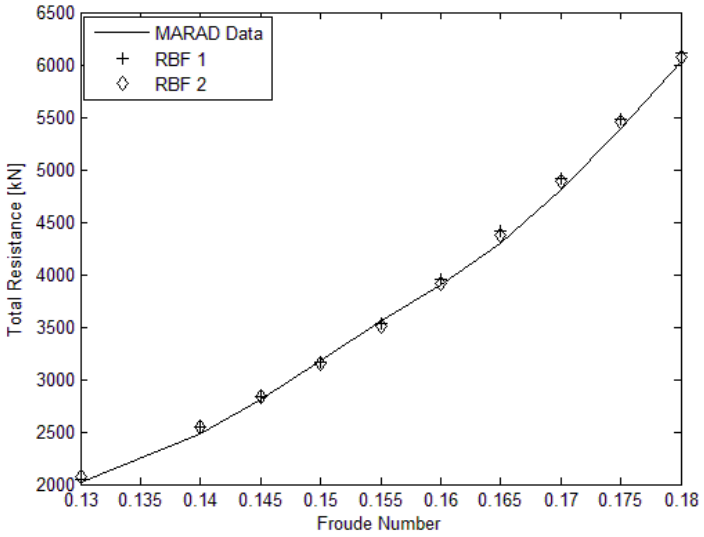


Fig. 51: Total resistance for ship B

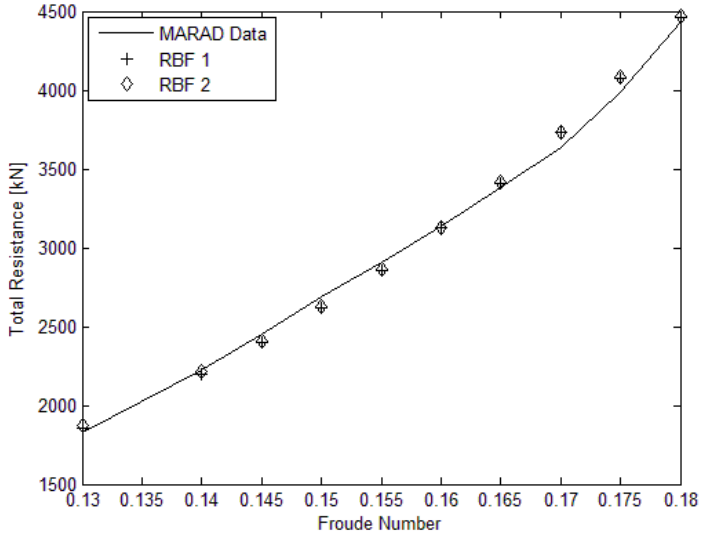


Fig. 52: Total resistance for ship E

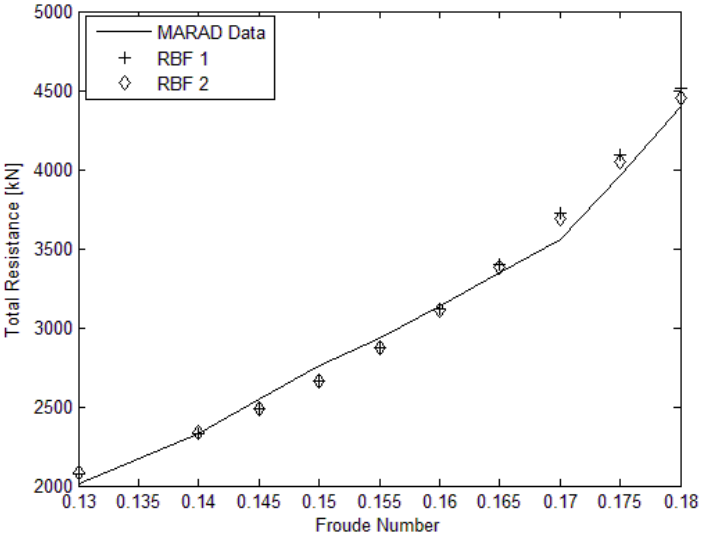


Fig. 53: Total resistance for ship H

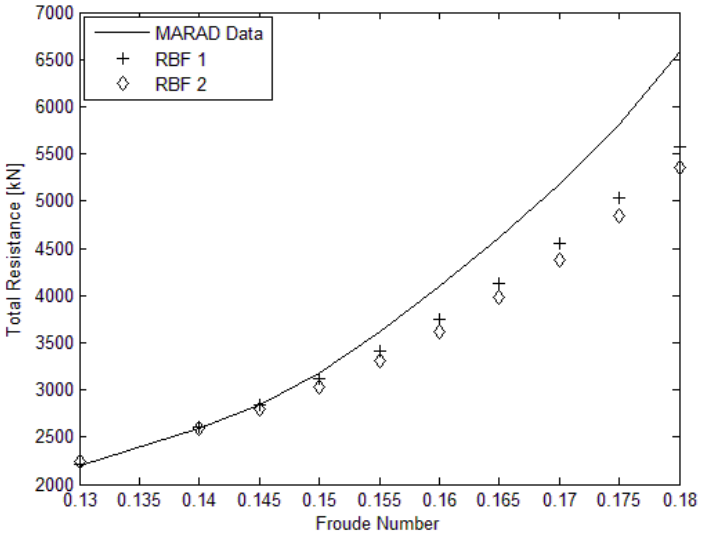


Fig. 54: Total resistance for ship M

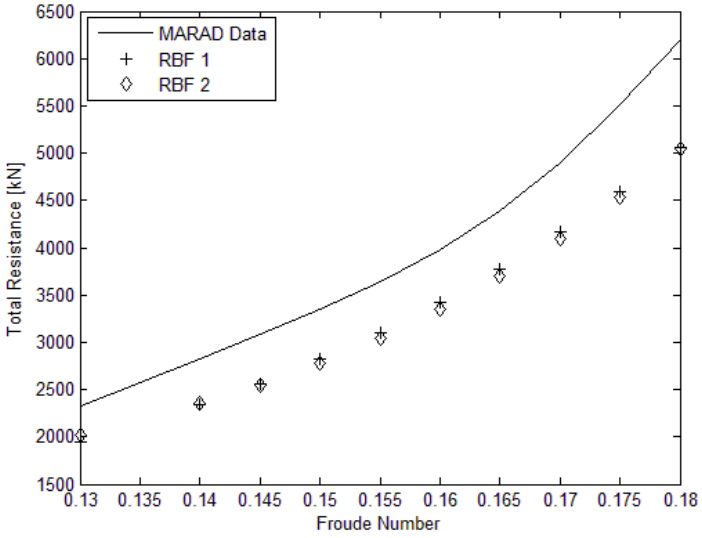


Fig. 55: Total resistance for ship O

Table 20: Total resistance and residual resistance coefficient prediction of ship S1

	MARAD		RBF1			RBF2		
Fn	C_R	R_T [kN]	C_R	R_T [kN]	diff. [%]	C_R	R_T [kN]	diff. [%]
0.1325	0.67	2023.5	0.71	2052.2	-1.4	0.68	2032.4	-0.4
0.1400	0.73	2294.3	0.73	2293.5	0.0	0.72	2292.3	0.1
0.1600	0.88	3173.4	0.86	3152.6	0.7	0.88	3183.2	-0.3
0.1775	1.06	4176.1	1.08	4210.8	-0.8	1.08	4216.8	-1.0
0.1800	1.10	4344.1	1.12	4374.7	-0.7	1.12	4370.5	-0.6

Table 21: Total resistance and residual resistance coefficient prediction of ship S2

	MARAD		RBF1			RBF2		
Fn	C_R	R_T [kN]	C_R	R_T [kN]	diff. [%]	C_R	R_T [kN]	diff. [%]
0.1325	0.54	2281.6	0.47	2212.0	3.0	0.54	2283.9	-0.1
0.1400	0.56	2560.0	0.55	2538.9	0.8	0.56	2552.8	0.3
0.1600	0.67	3492.3	0.74	3603.6	-3.2	0.69	3523.0	-0.9
0.1775	0.91	4739.4	0.92	4745.5	-0.1	0.90	4718.5	0.4
0.1800	0.95	4924.5	0.94	4905.8	0.4	0.94	4901.5	0.5

Table 22: Total resistance and residual resistance coefficient prediction of ship S3

	MARAD		RBF1			RBF2		
Fn	C_R	R_T [kN]	C_R	R_T [kN]	diff. [%]	C_R	R_T [kN]	diff. [%]
0.1325	0.52	1803.8	0.54	1825.9	-1.2	0.51	1801.9	0.1
0.1400	0.54	2021.4	0.56	2040.6	-1.0	0.54	2015.3	0.3
0.1600	0.68	2797.1	0.69	2811.2	-0.5	0.68	2793.4	0.1
0.1775	0.89	3748.2	0.91	3766.4	-0.5	0.90	3759.0	-0.3
0.1800	0.93	3897.1	0.94	3913.7	-0.4	0.94	3906.9	-0.3

Table 23: Total resistance and residual resistance coefficient prediction of ship S4

	MARAD		RBF1			RBF2		
Fn	C_R	R_T [kN]	C_R	R_T [kN]	diff. [%]	C_R	R_T [kN]	diff. [%]
0.1325	0.58	2008.2	0.53	1954.7	2.7	0.60	2027.0	-0.9
0.1400	0.62	2266.7	0.60	2244.1	1.0	0.61	2248.5	0.8
0.1600	0.73	3082.2	0.81	3186.5	-3.4	0.73	3086.2	-0.1
0.1775	0.98	4193.8	0.99	4197.4	-0.1	0.98	4181.7	0.3
0.1800	1.03	4375.9	1.01	4339.6	0.8	1.02	4355.8	0.5

Table 24: Total resistance and residual resistance coefficient prediction of ship S5

	MARAD		RBF1			RBF2		
Fn	C_R	R_T [kN]	C_R	R_T [kN]	diff. [%]	C_R	R_T [kN]	diff. [%]
0.1325	0.54	2004.0	0.53	1993.7	0.5	0.54	2011.1	-0.4
0.1400	0.57	2259.0	0.57	2257.2	0.1	0.57	2251.1	0.3
0.1600	0.78	3207.9	0.78	3210.6	-0.1	0.75	3173.0	1.1
0.1775	1.02	4336.7	1.05	4399.5	-1.4	1.05	4389.4	-1.2
0.1800	1.08	4542.4	1.10	4585.4	-0.9	1.10	4584.2	-0.9

Table 25: Total resistance and residual resistance coefficient prediction of ship S6

	MARAD		RBF1			RBF2		
Fn	C_R	R_T [kN]	C_R	R_T [kN]	diff. [%]	C_R	R_T [kN]	diff. [%]
0.1325	0.79	1865.0	0.80	1871.2	-0.3	0.81	1881.2	-0.9
0.1400	0.87	2137.7	0.85	2116.6	1.0	0.86	2124.2	0.6
0.1600	1.08	3008.8	1.06	2990.0	0.6	1.07	2994.9	0.5
0.1775	1.32	4023.8	1.35	4063.2	-1.0	1.35	4068.6	-1.1
0.1800	1.39	4219.0	1.40	4231.0	-0.3	1.40	4236.2	-0.4

Table 26: Total resistance and residual resistance coefficient prediction of ship S7

	MARAD		RBF1			RBF2		
Fn	C_R	R_T [kN]	C_R	R_T [kN]	diff. [%]	C_R	R_T [kN]	diff. [%]
0.1325	0.83	1967.7	0.82	1959.9	0.4	0.80	1939.9	1.4
0.1400	0.86	2211.8	0.90	2242.5	-1.4	0.86	2210.1	0.1
0.1600	1.21	3278.4	1.18	3240.9	1.1	1.17	3233.1	1.4
0.1775	1.57	4548.0	1.51	4452.5	2.1	1.60	4579.5	-0.7
0.1800	1.66	4780.1	1.56	4641.5	2.9	1.67	4799.7	-0.4

Table 27: Total resistance and residual resistance coefficient prediction of ship S8

	MARAD		RBF1			RBF2		
Fn	C_R	R_T [kN]	C_R	R_T [kN]	diff. [%]	C_R	R_T [kN]	diff. [%]
0.1325	0.54	1846.0	0.52	1835.3	0.6	0.55	1856.1	-0.5
0.1400	0.57	2075.8	0.57	2078.0	-0.1	0.59	2095.2	-0.9
0.1600	0.86	3057.0	0.86	3058.6	-0.1	0.85	3050.5	0.2
0.1775	1.29	4414.5	1.30	4436.4	-0.5	1.27	4381.9	0.7
0.1800	1.38	4667.0	1.38	4668.1	0.0	1.34	4604.8	1.3

3.3 Support Vector Machines

3.3.1 Implementation

The performance of SVMs is mainly affected by the number of the support vectors and the kind of the Kernel function used. The number of support vectors is determined by the algorithm and influenced by ε value. The box constraint C, and the kernel parameters are also important factors. The user is able to select the Kernel function and its parameters, the ε and C values. In order to determine the network that suits better to the problem at hand, several trials have been conducted, changing these values. The networks were evaluated based on their performance indices. The trials were conducted in two stages.

The MATLAB environment, as already mentioned, was selected for the development of SVMs and all the available tools that provides were deployed. In all trials the data were standardized and a 5-fold validation scheme was used. It is noted that 8% of the available data was kept for testing, while the other 92% for training and validation. In the first stage, SVMs using all available kernel functions were created and compared in order to select the one presenting the best results. These functions along with various performance indices calculated by the program are presented in Table 28. Regarding this table's notations, Root Mean Square Error (RMSE) is the square root of MSE and Mean Absolute Error (MAE) is the average of the absolute value of difference between targets and predicted values. The program selected the network that used the Fine Gaussian function as the best one, based on its RMSE. However, the network with the Medium Gaussian function presented lower MAE value. Both of the aforementioned functions are Gaussian kernels with different parameters; the Fine Gaussian has kernel scale equal to 0.5, while for the Medium Gaussian is 2. As a result, the Gaussian kernel was selected. Five values of kernel scales were tested in the second stage: 0.25, 0.5 (i.e. the Fine Gaussian), 1, 1.5, 2 (i.e. the Medium Gaussian). It is worth noting that the ε and C values in the first stage were set to auto for all SVM models. That is ε equal to interquartile range (IQR) and C equal to IQR divided by 1.349 for Gaussian kernels and 1 otherwise. During the second stage, these parameters were also subjected to testing.

Table 28: Performance of SVMs – first stage of trials

Kernel function	Mean value of validation folds			
	RMSE	MAE	MSE	R
Linear	0.19	0.14	0.0361	0.73
Quadratic	0.09	0.06	0.0081	0.94
Cubic	0.06	0.03	0.0036	0.97
Fine Gaussian	0.06	0.03	0.0036	0.97
Medium Gaussian	0.06	0.02	0.0036	0.97
Coarse Gaussian	0.11	0.07	0.0121	0.91

In the second stage, SVMs with Gaussian kernels using 5 kernel scales were trained. Each one of these five cases, was tested with combinations of ε and C values, in order for the best one to be obtained. In this stage, the performance indices regarding the test set were also available. The procedure followed for each case is described herein. Firstly, the ε value is set to auto while the C varies and a series of networks is trained with these characteristics. Subsequently, the C value is set to auto and the ε varies. The C and ε values of the best performing networks according to validation and test MSE are selected, resulting to 1 to 4 combinations for each case depending on possible overlapping. The SVMs with ε value equal to 0.001 produced better results for all kernels and datasets. As a result, for each case two final combinations of C and ε values were selected and tested. The results of this stage are presented in Table 29. For each kernel scale value the performance indices of the best networks with ranging ε and C values are provided in the first two rows. In the third and the fourth row the

characteristics of the networks with the best *C* according to the validation and test MSE respectively are provided. It is noted that a total number of 105 SVMs was developed and evaluated in this stage, and ten of them were selected for further testing. A graphical description of this stage is provided in Fig. 56.

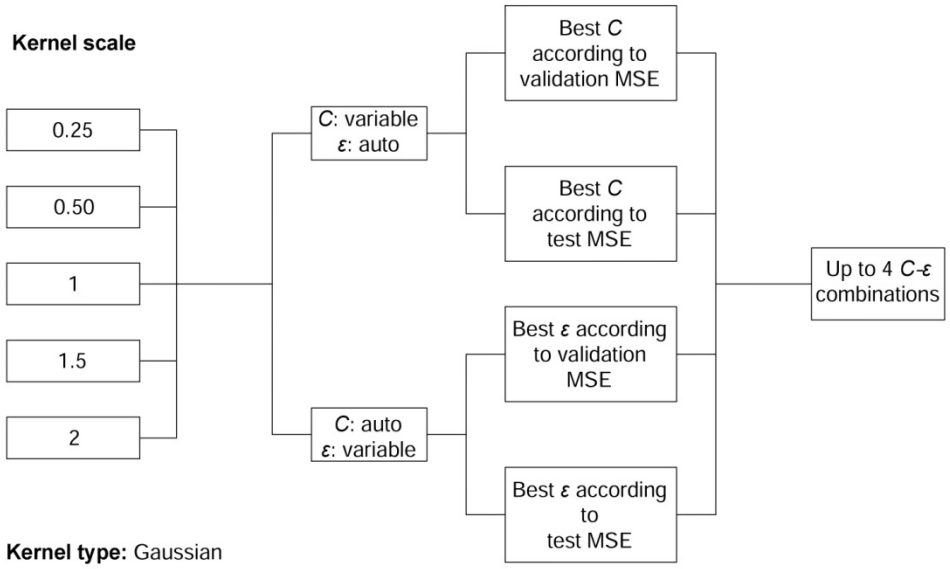


Fig. 56: Graphical description of the second stage of trials

Table 29: MSE and R values of SVMs – second stage of trials

Kernel scale	ε	C	Mean of 5 validation folds		Test Set	
			min MSE	max R	min MSE	max R
0.25	auto	variable	0.0025	0.98	0.001500	0.9864
	variable	auto	0.0016	0.99	0.000440	0.9961
	0.001	1	0.0009	0.99	0.000143	0.9987
	0.001	10	0.0016	0.99	0.000077	0.9993
0.50	auto	variable	0.0010	0.99	0.000869	0.9923
	variable	auto	0.0025	0.99	0.000063	0.9994
	0.001	500	0.0009	0.99	0.000083	0.9993
	0.001	10	0.0025	0.99	0.000007	0.9999
1	auto	variable	0.0025	0.98	0.000589	0.9948
	variable	auto	0.0025	0.98	0.000049	0.9996
	0.001	2000	0.0025	0.97	0.000018	0.9998
	0.001	20	0.0025	0.98	0.000014	0.9999
1.50	auto	variable	0.0036	0.98	0.000486	0.9957
	variable	auto	0.0025	0.98	0.000059	0.9995
	0.001	500	0.0025	0.98	0.000035	0.9997
	0.001	3	0.0025	0.98	0.000033	0.9997
2	auto	variable	0.0036	0.98	0.000441	0.9961
	variable	auto	0.0036	0.98	0.000213	0.9981
	0.001	2000	0.0025	0.98	0.000169	0.9985
	0.001	1500	0.0025	0.98	0.000115	0.9990

The tested values of ε for all kernels were auto, 0.1,0.05,0.01,0.001. The range of C values for each kernel depended on the observed results and are presented in the following table.

Table 30: Tested C values for each kernel case – second stage of trials

Kernel scale	Tested *C* values														
0.25	auto	0.05	0.5	0.75	1	1.25	1.5	2	3	10	20	50	100	200	500
0.50	auto	0.05	0.5	1	2	3	10	20	50	100	200	500	1000	1500	2000
1	auto	0.05	0.5	1	2	3	10	20	50	100	200	500	1000	1500	2000
1.5	auto	0.05	0.5	1	2	3	10	20	50	100	200	500	1000	1500	2000
2	auto	0.5	1	2	3	10	20	50	100	200	500	1000	1500	2000	2500

3.3.2 Discussion and Results

The second stage resulted in two networks for each case of kernel scale, leading to a total number of ten networks. Subsequently, the generalization ability of these networks was evaluated using data from five MARAD hullforms (B,E,H,M and O) listed in Table 5 (chapter 3.1.2). Once again, it is noted that data from these hulls was not included in the training and test data sets used for the training and evaluation of the networks. More importantly, for two of them (ships M and O) their geometric characteristics (i.e. their combination of C_B and B/T values) exceeded the limits of the training data set.

Two networks were finally selected. The first one (denoted SVM1) because it presented the best fit inside the dataset and the second one (denoted SVM2) for its lower errors outside its limits. SVM1 uses Gaussian kernel with kernel scale 1.5, C value equal to 500 and ε equal to 0.001, while the corresponding values for SVM2 are 2,1500 and 0.001. The comparison of the residual resistance coefficient (C_R) in a range of Froude numbers predicted by these two neural networks with the actual values according to the MARAD diagrams for the five ships is illustrated in Fig. 57 to Fig. 61.The predictions for the first three ships, i.e. the MARAD hulls within the limits of the training set, are quite close to the real values (see Fig. 57 - Fig. 59). In particular, these for ship E (Fig. 58) are the best ones compared to those for the other two ships.

As already mentioned, the geometric properties of ships M and O exceeded the range of the training data set. For these ships, the predictions of the two networks are poor. However, those of SVM2 are considerably better than that of SVM1 for both ships. The predictions for ship O are better than those regarding ship M (Fig. 60 and Fig. 61).

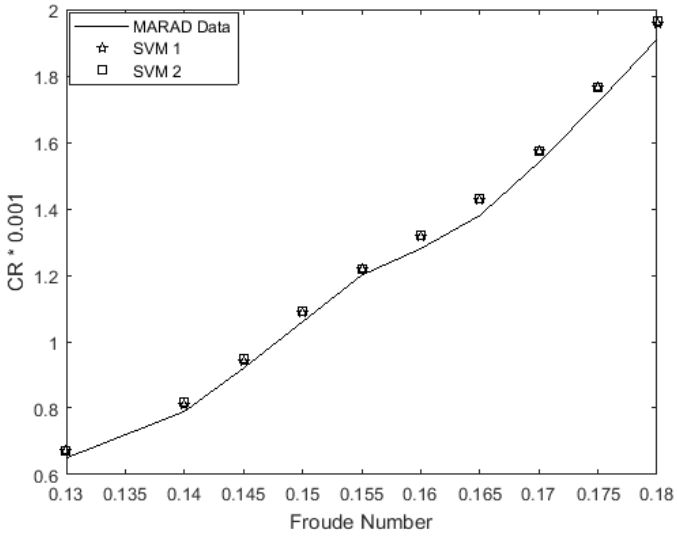


Fig. 57: Residual resistance coefficient for ship B

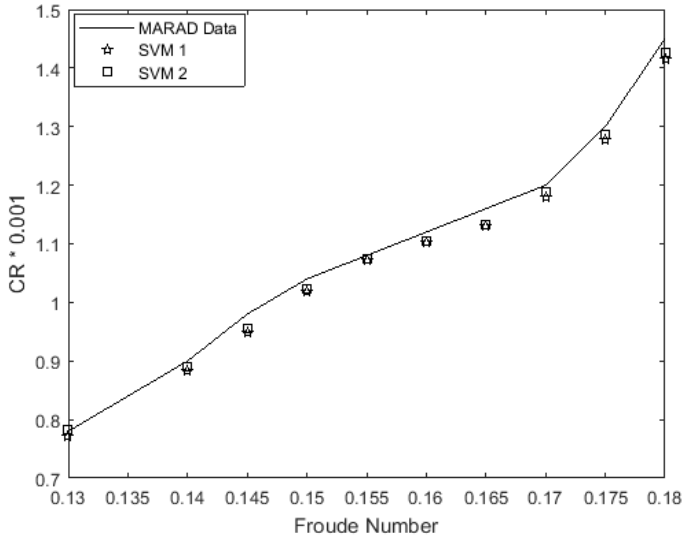


Fig. 58: Residual resistance coefficient for ship E

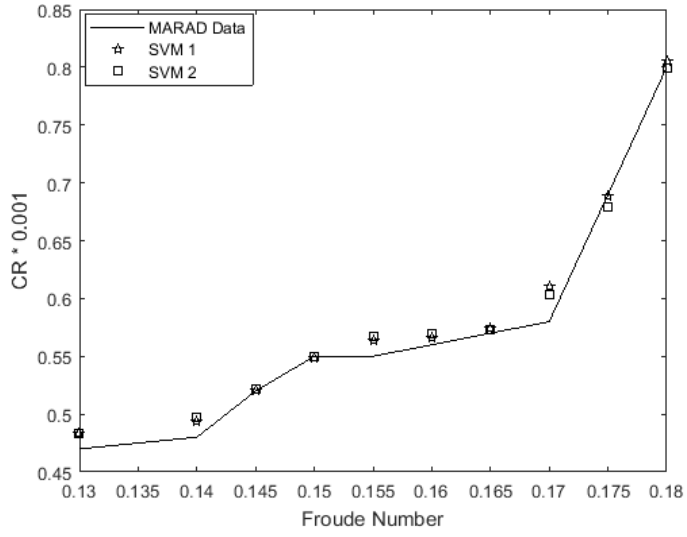


Fig. 59: Residual resistance coefficient for ship H

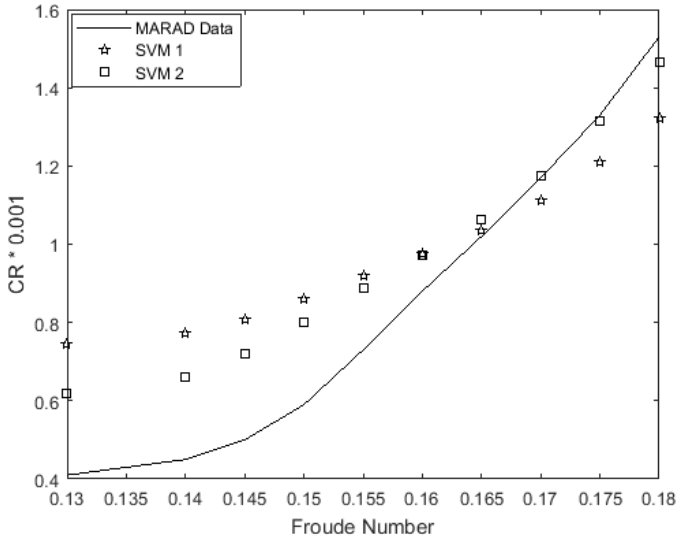


Fig. 60: Residual resistance coefficient for ship M

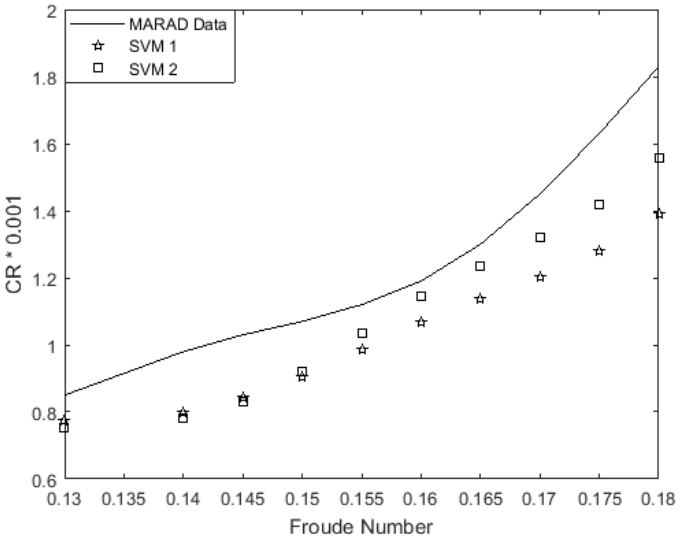


Fig. 61: Residual resistance coefficient for ship O

A comparison of the results for the total resistance of these five ships based on the MARAD data and the network predictions for the C_R coefficient is presented in Fig. 62 to Fig. 66 as well as in Table 31 to Table 35. The predictions for the first three ships with characteristics within the limits of the training dataset (ships B, E, H) present low deviations from the experimental values, in the order of 0.1% to 1.6% for SVM1 and up to 1.7% for SVM2. For the other two ships exceeding the limits of the dataset, the deviations were larger. These of SVM1 ranged between 0.6% to 17.4% for ship M and 3.3% to 13.2% for ship O, while these of SVM2 were lower (0.5% to 11.0% for ship M and 1.7% to 8.3% for ship O).

Table 31: Total resistance prediction of ship B

		MARAD	SVM1		SVM2	
Fn	v [kn]	R_T [kN]	R_T [kN]	diff. [%]	R_T [kN]	diff. [%]
0.130	14.8	2025.4	2045.7	-1.0	2046.0	-1.0
0.140	15.9	2486.3	2509.0	-0.9	2518.4	-1.3
0.145	16.5	2810.3	2833.6	-0.8	2844.1	-1.2
0.150	17.1	3173.3	3209.3	-1.1	3213.3	-1.3
0.155	17.6	3565.7	3587.7	-0.6	3590.0	-0.7
0.160	18.2	3904.5	3956.4	-1.3	3963.4	-1.5
0.165	18.8	4294.3	4361.2	-1.6	4367.2	-1.7
0.170	19.4	4804.3	4860.0	-1.2	4858.2	-1.1
0.175	19.9	5385.4	5458.7	-1.4	5461.8	-1.4
0.180	20.5	6027.0	6110.8	-1.4	6122.3	-1.6

Table 32: Total resistance prediction of ship E

		MARAD	SVM1		SVM2	
Fn	v [kn]	R_T [kN]	R_T [kN]	diff. [%]	R_T [kN]	diff. [%]
0.130	14.0	1832.1	1825.7	0.4	1834.0	-0.1
0.140	15.1	2222.2	2207.2	0.7	2212.4	0.4
0.145	15.6	2455.9	2425.9	1.2	2432.3	1.0
0.150	16.1	2684.7	2661.7	0.9	2667.9	0.6
0.155	16.7	2905.1	2895.8	0.3	2899.0	0.2
0.160	17.2	3136.7	3117.2	0.6	3118.4	0.6
0.165	17.8	3379.9	3341.3	1.1	3345.6	1.0
0.170	18.3	3634.8	3608.5	0.7	3618.6	0.4
0.175	18.8	3986.7	3955.6	0.8	3968.9	0.4
0.180	19.4	4435.6	4384.6	1.1	4399.5	0.8
0.185	19.9	4915.8	4866.4	1.0	4887.7	0.6
0.190	20.5	5444.9	5387.4	1.1	5416.5	0.5

Table 33: Total resistance prediction of ship H

		MARAD	SVM1		SVM2	
Fn	v [kn]	R_T [kN]	R_T [kN]	diff. [%]	R_T [kN]	diff. [%]
0.130	15.3	2012.7	2026.5	-0.7	2025.8	-0.7
0.140	16.5	2332.1	2347.7	-0.7	2352.8	-0.9
0.145	17.0	2544.9	2545.0	0.0	2546.4	-0.1
0.150	17.6	2756.6	2753.9	0.1	2756.3	0.0
0.155	18.2	2936.0	2955.0	-0.6	2960.7	-0.8
0.160	18.8	3136.2	3145.4	-0.3	3151.0	-0.5
0.165	19.4	3343.8	3350.0	-0.2	3348.5	-0.1
0.170	20.0	3558.8	3610.9	-1.5	3599.0	-1.1
0.175	20.6	3964.5	3961.3	0.1	3944.9	0.5
0.180	21.2	4398.9	4408.1	-0.2	4398.1	0.0
0.185	21.7	4945.0	4939.3	0.1	4934.0	0.2
0.190	22.3	5552.4	5550.9	0.0	5518.1	0.6

Table 34: Total resistance prediction of ship M

		MARAD	SVM1		SVM2	
Fn	v [kn]	R_T [kN]	R_T [kN]	diff. [%]	R_T [kN]	diff. [%]
0.130	15.8	2202.5	2586.8	-17.4	2440.4	-10.8
0.140	17.0	2592.1	3020.7	-16.5	2873.7	-10.9
0.145	17.6	2844.2	3283.9	-15.5	3156.9	-11.0
0.150	18.2	3173.3	3585.8	-13.0	3493.2	-10.1
0.155	18.8	3608.8	3917.6	-8.6	3864.8	-7.1
0.160	19.4	4097.9	4266.4	-4.1	4259.3	-3.9
0.165	20.0	4608.4	4638.1	-0.6	4689.1	-1.8
0.170	20.6	5177.8	5061.3	2.2	5188.5	-0.2
0.175	21.2	5810.8	5558.7	4.3	5781.5	0.5
0.180	21.8	6578.7	6116.5	7.0	6438.6	2.1

Table 35: Total resistance prediction of ship O

		MARAD	SVM1		SVM2	
Fn	v [kn]	R_T [kN]	R_T [kN]	diff. [%]	R_T [kN]	diff. [%]
0.130	14.9	2325.3	2249.4	3.3	2229.3	4.1
0.140	16.1	2830.3	2626.3	7.2	2606.4	7.9
0.145	16.6	3089.9	2860.5	7.4	2847.3	7.9
0.150	17.2	3351.6	3135.6	6.4	3157.1	5.8
0.155	17.8	3640.8	3452.9	5.2	3521.1	3.3
0.160	18.4	3975.5	3793.7	4.6	3906.6	1.7
0.165	18.9	4393.1	4136.9	5.8	4292.0	2.3
0.170	19.5	4905.8	4489.2	8.5	4688.4	4.4
0.175	20.1	5508.8	4888.1	11.3	5137.6	6.7
0.180	20.7	6194.0	5373.3	13.2	5682.5	8.3

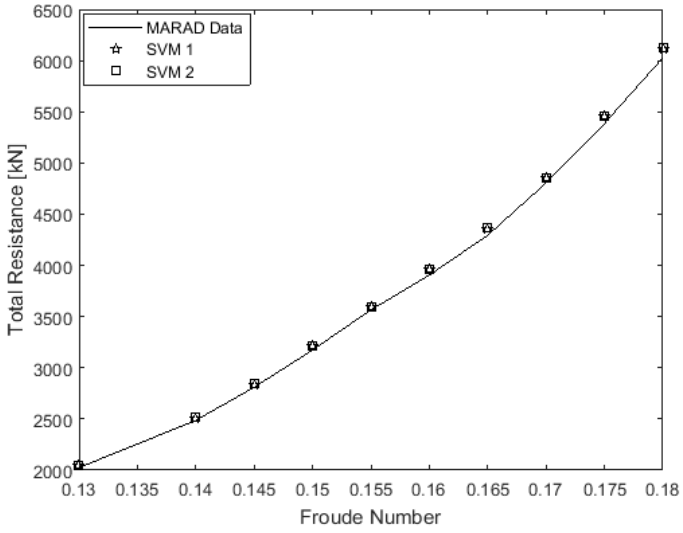


Fig. 62: Total resistance for ship B

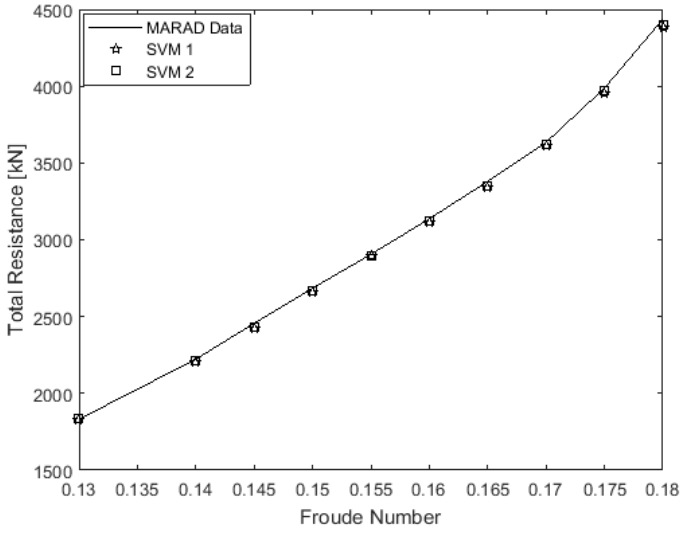


Fig. 63: Total resistance for ship E

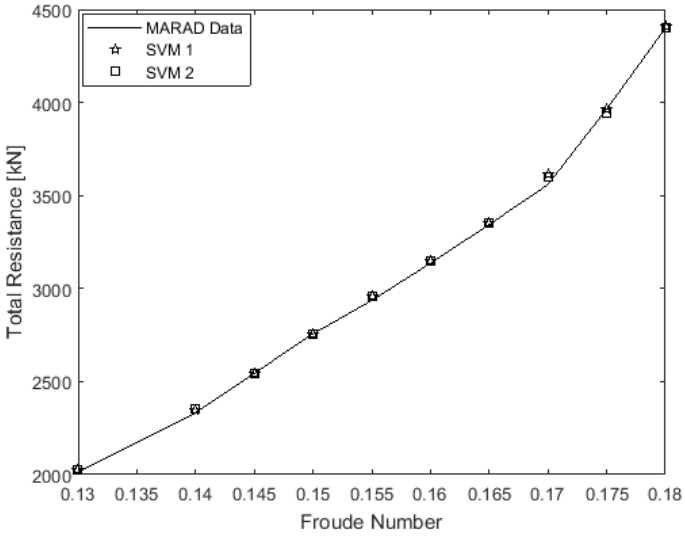


Fig. 64: Total resistance for ship H

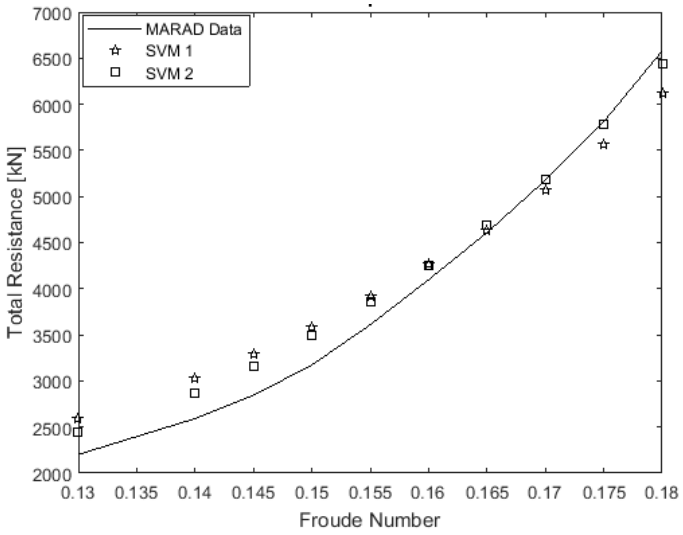


Fig. 65: Total resistance for ship M

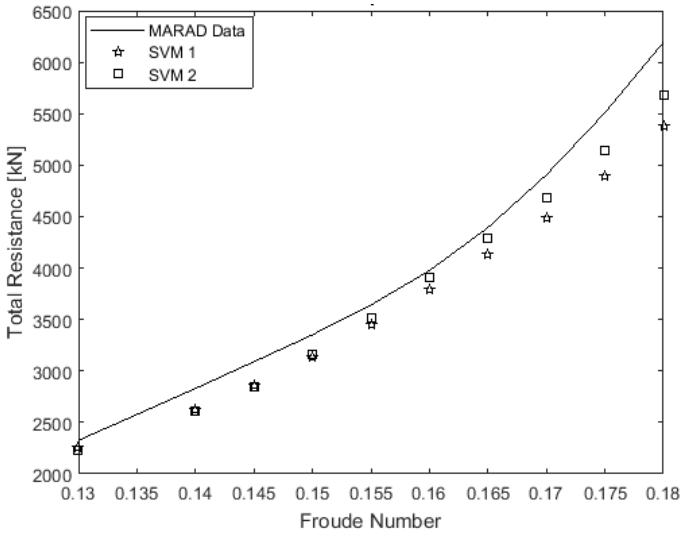


Fig. 66: Total resistance for ship O

3.4 Comparison of selected ANNs

The trials described in the previews chapters resulted in the selection of 8 neural networks; 4 MLPs, 2 RBFs and 2 SVMs. These networks are compared herein based on their predictions of C_R for the five MARAD hullforms (B,E,H,M and O). All networks performed quite well inside the dataset (ships B,E,H), with the best one being SVM1 with observed deviation of predictions from the target values up to 1.6%, followed by SVM2, MLP2 and MLP1. Outside the dataset the best performing network was MLP4, presenting deviations not higher than 6.8% from the desired values, while MLP3 and SVM2 are following with 7.8% and 11% respectively. The performance of RBFs was not superior than that of the other networks, although they presented deviations lower than 5% inside the dataset's limits, which is considered acceptable. An overall comparison of the highest observed deviations between estimations and desired values per ship and network are provided in Fig. 67. Figures

illustrating the predictions of C_R the five ships by the selected networks are also presented in Fig. 68-Fig. 72.

It is worth noting that predictions using the developed artificial neural networks were conducted for Froude numbers up to 0.18, and in some cases up to 0.19, which are quite high when compared to Froude numbers of existing full-form vessels of similar size. However, since the data provided by the MARAD diagrams are extending up to these values, it was decided to use the full range of available data during the training and evaluation of the networks.

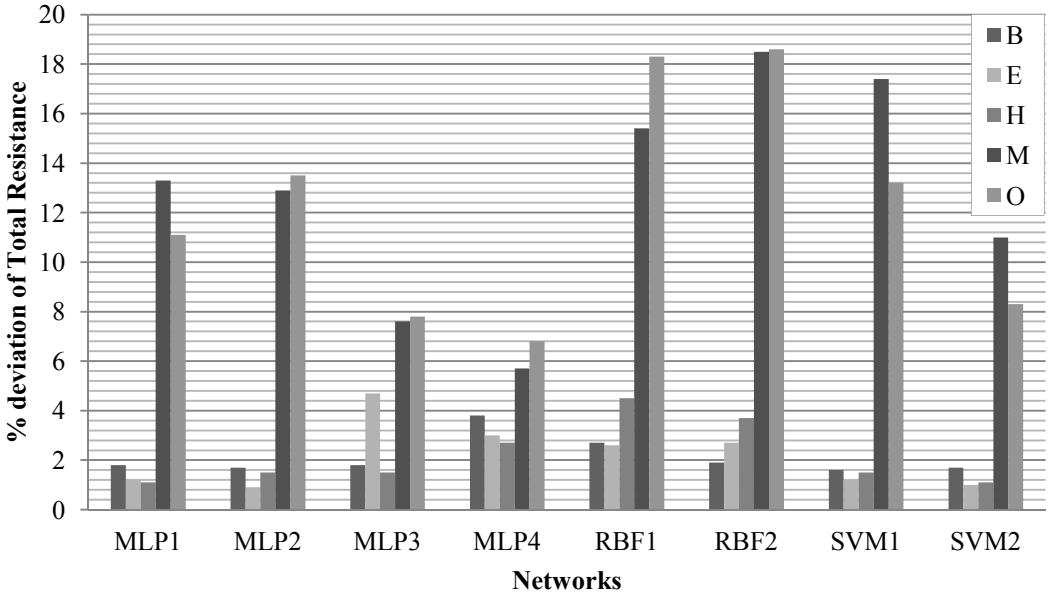


Fig. 67: Higher observed deviation of Total Resistance per ship and network

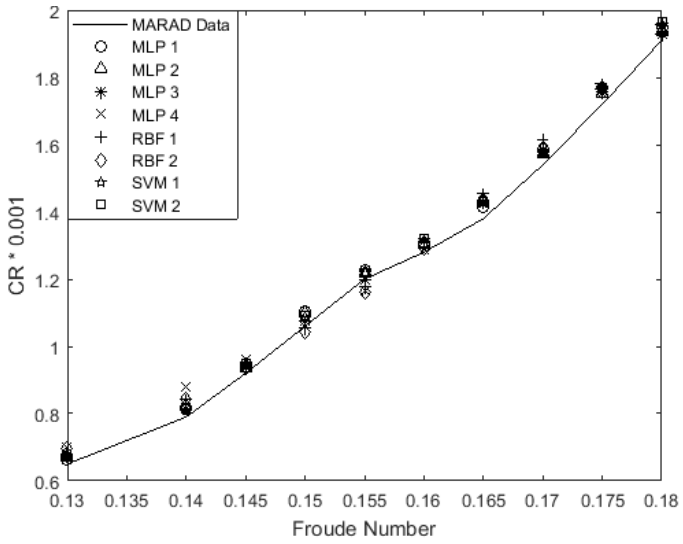


Fig. 68: Residual resistance coefficient for ship B

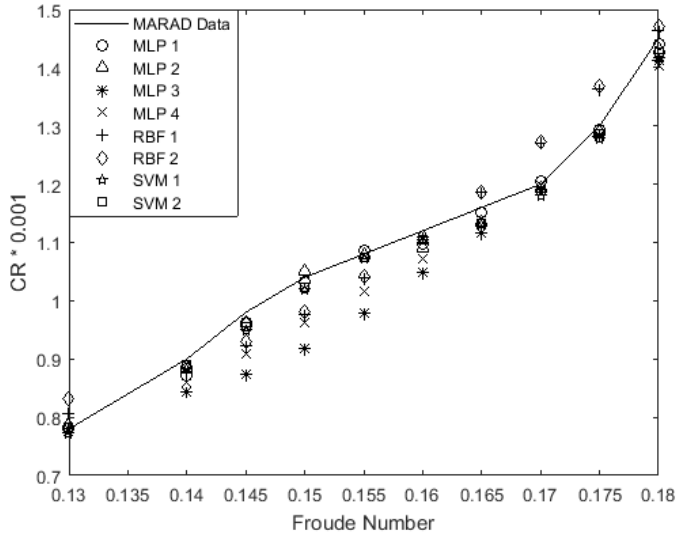


Fig. 69: Residual resistance coefficient for ship E

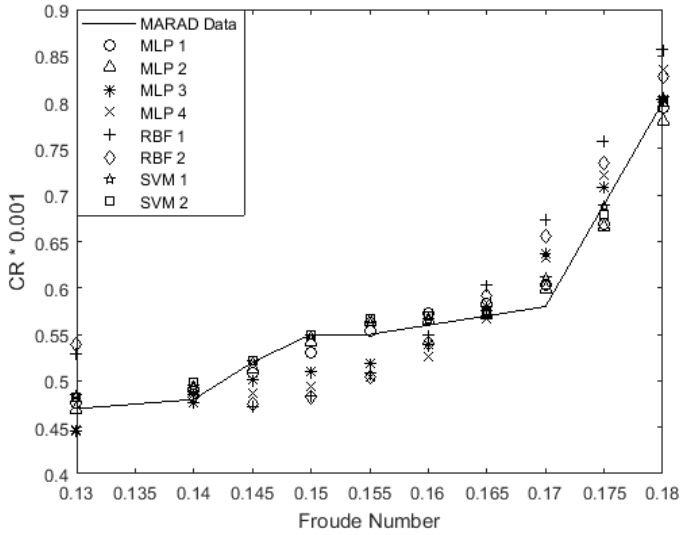


Fig. 70: Residual resistance coefficient for ship H

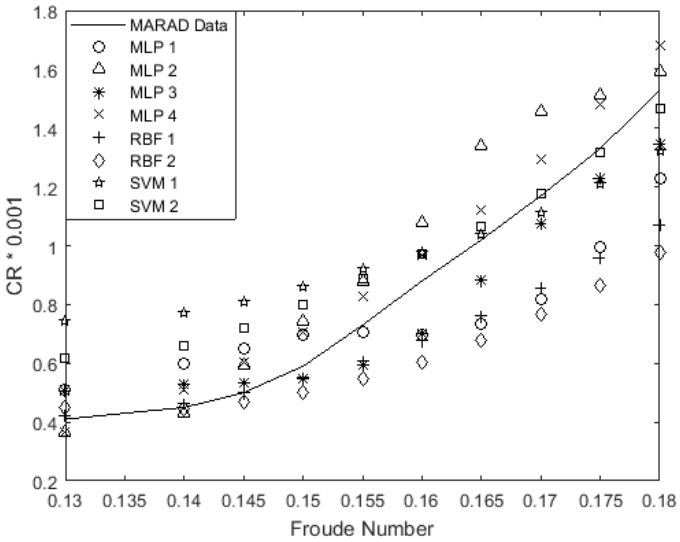


Fig. 71: Residual resistance coefficient for ship M

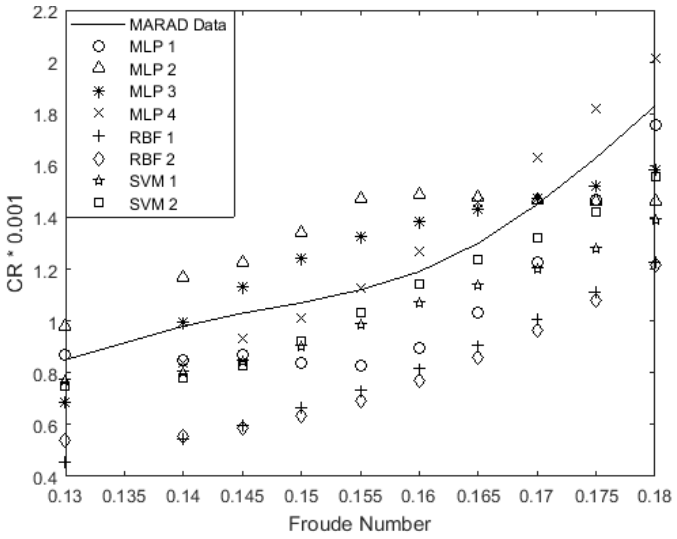


Fig. 72: Residual resistance coefficient for ship O

Conclusion

This thesis investigates the ability of Artificial Neural Networks to predict the resistance of MARAD hullforms. The data for the training and evaluation of the networks were obtained from five diagrams illustrating the residual resistance coefficient (C_R) for the full load condition presented in [5]. A total number of 1893 points were collected from these diagrams and were separated into pairs of input – output data. The input vector consisted of length to beam ratio (L/B), breadth to draft ratio (B/T), block coefficient (C_B) and the Froude number (F_n), while the output was the residual resistance coefficient (C_R).

Multi-layer perceptrons were trained and evaluated in two stages of trials. In the first stage a total number of 288 networks were developed. In this stage the learning function and rate were held constant, while different architectures, training and activations functions were tested. According to their performance on a validation data set, six networks of specific architecture, training and activations functions were selected. In the second set of trials, these six networks were tested in six learning rates and two learning functions while their other characteristics remained constant. Finally, four networks were selected. The first two presented the lower mean squared error on the validation dataset. The other two networks were selected for their possible generalization ability beyond the limits of the training data.

Radial Basis Function networks were developed through a three stages process. In the first stage, 103 networks were trained. The division ratio of the available data to training and validation set was 80% and 20% respectively. In the second stage, the 85% of the data was assigned to the training set, 15% to the validation set and 16 networks were created. In the third stage, six networks were tested, using a modified K-means training algorithm, aiming to enhance their performance. All networks presented relatively high deviations outside the limits of the dataset, and as a result the two networks presenting the lower deviations inside its limits were selected.

Support Vector Machines were trained using a two stages process. In the initial stage, SVMs with six Kernel functions were trained, in order for the most appropriate to be selected. The functions of the two best performing networks were Gaussian with different kernel scale configurations. As a result, in the second stage the Gaussian function was selected and 5 alternative kernel scales were tested. Every kernel scale was tested using 21 combinations of ε and C values. This process resulted in two networks for each kernel scale, leading to a total number of ten networks. Two of these were finally selected. The first one presented the lower validation mean square error inside the dataset and the second one presented the lower deviations outside the dataset, while performing well inside the dataset.

Multi-layer perceptron neural networks and Support Vector Machines proved to be particularly effective in the evaluation of resistance of MARAD Hull forms with geometric characteristics within the limits of training data, with total deviations less than 2%. Some of the networks presented the ability of generalization exceeding the

limits of the training data. The obtained results with the MARAD series indicate that ANNs may be effectively used to provide accurate resistance predictions also for other systematic series. In addition, it might be argued that ANNs could be successfully trained to estimate calm water resistance of selected hullform types, based on the results of systematic calculations using advanced CFD software tools. They might be also used as an efficient means of calculation in other kinds of problems, such us for the prediction of maneuvering or seakeeping characteristics of ships, provided that adequate data for training are available.

References

1. Todd, F.H.,(1963) Series 60 Methodical Experiments with Models of Single-Screw Ships, Report 1712, U.S. Government Printing Office
2. Williams, A. (1969), The SSPA Cargo Liner Series Resistance. SSPA Report No 66. Giitenburg, Statens Skeppsprovningsanstalt
3. Moor, D., Parker, M.N., Pattulo, R. N. M. (1961) The BSRA Methodical Series – An Overall Presentation, Transactions RINA, Volume 103
4. Roseman,D.P., Seibold,F.,(1985) An Introduction to the U.S. Maritime Administration Systematic Series, Marine Technology 221, pp. 329-338
5. Roseman,D.P., Seibold,F.,(1987) The MARAD Systematic Series of full-form ship models, The society of Naval Architects and Marine Engineers
6. Shaher Sabit, A. (1971), A Regression Analysis of the Resistance Results of the BSRA Standard Series, International Shipbuilding Progress, Jan 1971, 3-17
7. Shaher Sabit, A. (1972), An Analysis of the Series 60 Results. Part 1, Analysis of Forms and Resistance Results, International Shipbuilding Progress, March 1972, 81-97.
8. Shaher Sabit, A. (1972), An Analysis of the Series 60 Results. Part 2, Analysis of Propulsion Factors, International Shipbuilding Progress, September 1972, 294-301
9. Shaher Sabit, A. (1976), The SSPA Cargo Liner Series Regression Analysis of the Resistance and Propulsive Coefficients, International Shipbuilding Progress, 23 (263), 213-217.
10. Radojcic, D., Zgradic, A., Kalajdzic, M., Simic, A. (2014), Resistance Prediction for Hard Chine Hulls in the Pre-Planning Regime, Polish Maritime Research 2(82), Vol 21, pp. 9-26.
11. Bojovic, P. (1997), Resistance of AMECRC Systematic Series of High Speed Displacement Hullforms, IV Symposium on High Speed Marine Vehicles, 18-21 March 1997, Naples, Italy.
12. Mason, A., Couser, P., Mason, G., Smith, C.R., von Konsky, B.R. (2005), Optimization of Vessel Resistance using Generic Algorithms and Artificial Neural Networks, COMPIT 2005, Hamburg, Germany pp. 440-454.
13. Koushan, K. (2001), Empirical prediction of ship resistance and wetted surface area using Artificial Neural Network, Proceedings of the Eighth International Symposium on Practical Design of Ships and other Floating Structures, Vol. 1, pp. 501-508.
14. Ortigosa, I., López, R., García, J. (2014), Prediction of Total Resistance Coefficients Using Neural Networks, Journal of Maritime Research, [S.l.], v. 6, n. 3, p. 15-26, ISSN 1697-9133, http://www.jmr.unican.es/index.php/jmr/article/view/119
15. Cepowski, T. (2005), Application of statistical methods and artificial neural networks for approximating ship's roll in beam waves, Polish Maritime Research, No2/2005.
16. Cepowski, T., (2007), Application of artificial neural networks to approximation and identification of sea-keeping performance of a bulk carrier in ballast loading condition, Polish Maritime Research 4(54) Vol. 14, pp. 31-39.
17. Yao, J., Han, D. (2012), RBF Neural Network Evaluation Model for MDO Design of Ship, IPCSIT Vol. 47, pp. 309-312.
18. Martins, P.T., Lobo, V. (2007), Estimating Maneuvering and Sea-keeping characteristics with Neural Networks, IEEE OCEANS 2007-Aberdeen.
19. Pedersen, B.P., Larsen, J. (2009), Prediction of Full-Scale Propulsion Power using Artificial Neural Networks, COMPIT 2009: 8th International Conference on Computer and IT Applications in the Maritime Industries, Budapest, pp. 537-550
20. Bishop, C.M.(1995), Neural Networks for Pattern Recognition, Clarendon Press, Oxford, ISBN-10: 0198538642.
21. Haykin, S. (1999), Neural Networks – A Comprehensive Foundation, Prentice-Hall, Upper Saddle River, ISBN 81-7808-300-0.
22. Politis, G.K.(2011), Ship Resistance and Propulsion, NTUA Publications.
23. Molland, A.F., Turnock S.R., Hudson, D.A. (2011), Ship Resistance and Propulsion. Practical estimation of ship propulsive power, Cambridge University Press, ISBN 978-0-521-76052-2.
24. Yokoo, Koichi, (1966), Systematic Series Model Tests in Japan Concerning the Propulsive Performance of Full Ship Forms, Japan Shipbuilding and Marine Engineering Volume 1, Number 2
25. Muntjewerf, J. J., (1970), Methodical Series Experiments on Cylindrical Bows, Transactions of RINA, Vol. 112, No. 2.

26. Troost, L., Todd, F.H., Hughes, G.(1957), Skin Friction and Turbulence Stimulation, International Towing Tank Conference Proceedings, Spain, Madrid, pp. 136
27. McCulloch, W., Pitts, W. (1943), A logical Calculus of the ideas immanent in nervous activity, Bulletin of Mathematical Biology, Vol. 5, issue 4, pp.115-133.
28. Russell S., Norvig P.,(1995) Artificial intelligence – A modern approach, Prentice-Hall, Upper Saddle River, ISBN 0-13-103805-2.
29. Hebb,D. (1949) The Organization of behavior, John Willy and Sons Inc, ISBN 9780471367277.
30. Rosenblatt,F. (1958) The Perceptron: A Probabilistic Model for Information Storage and Organization in the brain, Cornell Aeronautical Laboratory, Psychological Review, Vol. 65, No. 6, pp. 386–408.
31. Widrow, B., Hoff, M. E., Jr. (1960) Adaptive switching circuits, in 1960 IRE WESCON Convention Record, Part 4, New York: IRE, pp. 96–104.
32. Widrow, B. (1962) Generalization and Information Storage in Networks of Adaline `Neurons', Self-Organizing Systems symposium proceedings, Spartan Books, Washington, pp. 435-461.
33. Minksy, M,Papert,S (1969) Perceptrons: An introduction to computational geometry, Cambridge, Mass: MIT Press, ISBN: 0262130432.
34. Willshaw, D.J., von der Malsburg, C. (1976) How patterned neural connections can be set up by self-organization, Proceedings of the Royal Society of London. Series B, Vol. 194, No. 1117, pp. 431-445.
35. Hopfield, J.J. (1982) Neural networks and physical systems with emergent collective computational abilities, Proceedings of the National Academy Sciences, Vol. 79, pp. 2554-2558.
36. Rojas,R. (1996), Neural Networks – A Systematic Introduction, Berlin: Springer –Verlag, ISBN 3540605053.
37. Kohonen, T. (1982) Self-organized formation of topologically correct feature maps, Biological Cybernetics, Vol. 43, pp. 59-69.
38. Rumelhart, D.E., Hinton, G. E.,Williams, R. J. (1986) Learning Representations by Back Propagating Errors, Nature, Vol. 323, pp. 533-536.
39. Rumelhart, D.E., McClelland, PDP Research Group (1986) Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations, Vol. 1, The MIT Press, ISBN: 9780262181204
40. Broomhead, D.S., Lowe, D. (1988) Multi-Variable Functional Interpolation and Adaptive Networks. Complex Systems, 2, 327-355.
41. Boser, B.E., Guyon, I.M., Vapnik, V.N. (1992) A Training Algorithm for Optimal Margin Classifiers, Proceedings of the 5th annual workshop on Computational Learning Theory, pp. 144-152.
42. Cortes, C., Vapnik, V. (1995) Support-Vector Networks, Machine Learning, Vol. 20, Issue 3, pp. 273-297.
43. Zurada J.M.(1992) Introduction to artificial neural systems, West Publishing Company, St. Paul, ISBN 0-314-93391-3.
44. Krose, B., van der Smagt, P. (1996), An Introduction to Neural Networks, The University of Amsterdam.
45. Snyman, J. A. (2005), Practical Mathematical Optimization, Springer, ISBN 0-387-24348-8.
46. Hagan, M.T., Demuth, H.B., Beale, M.H., De Jesús, O. (2014), Neural Network Design, 2nd Edition, Martin Hagan, ISBN: 0971732116.
47. Levenberg, K. (1944), A method for the solution of certain non-linear problems in least squares, Quarterly of Applied Mathematics, 2, pp. 164-168.
48. Marquardt, D. (1963), An Algorithm for Least-Squares Estimation of Nonlinear Parameters, Journal of the Society for Industrial and Applied Mathematics, Vol. 11, No. 2, pp. 431-441.
49. Beale M.H., Hagan M.T., Demuth H.B. (2015), Neural Network Toolbox-User's Guide, The Mathworks Inc.
50. Powell, M.J.D. (1987), Radial Basis Functions for Multivariable Interpolation: A Review, IMA Conference on Algorithms for Approximation of Functions and Data, Oxford University Press, pp. 143-167.
51. Poggio, T., Girosi, F. (1990) Networks for Approximation and Learning, Proceedings of the IEEE, Vol. 78, No. 9, September 1990.

52. Linde,Y., Buzo, A, Gray, R.M. (1980) An algorithm for vector quantizer design. IEEE Transactions on Communications, Vol. 28, No.1, pp. 84-95.
53. Vapnik, V.N., Lerner, A (1963) Pattern recognition using generalized portraits, Automation and Remote Control, Vol. 24, No. 6, pp. 774-780.
54. Vapnik, V.N., Chervonenkis, A. (1964) A class of algorithms for pattern recognition learning, Automation and Remote Control (in Russian), Vol. 25, Issue 6, pp. 937–945.
55. Drucker, H., Burges, C.J.C., Kaufman, L., Smola, A.,Vapnik, V.N.(1997) Support vector regression machines, Advances in Neural Information Processing Systems 9, MIT Press, pp. 155–161.
56. Müller, K.R., Smola, A., Rätsch, G., Schölkopf, B , Kohlmorgen, J, Vapnik, V.N. (1997) Predicting time series with support vector machines, ICANN'97, Springer Lecture Notes in Computer Science, Vol. 1327, pp. 999–1004, Berlin.
57. Osuna, E.E., Freund, R., Girosi, F. (1997) Support Vector Machines: Training and Applications, MIT, Artificial Intelligence Laboratory and Center of Biological and Computational Learning Department of Brain and Cognitive Sciences, A.I. Memo No. 1602, C.B.C.L. Paper No. 144, March 1997 (http://hdl.handle.net/1721.1/7290).
58. Vapnik, V. N. (2000) The Nature of Statistical Learning Theory, Springer-Verlag, 2nd Edition, ISBN: 0-387-9878-0.
59. Smola, J., Schölkopf, B. (2004) A tutorial on support vector regression, Statistics and Computing, Vol. 14, pp. 199-222.
60. Platt, J. (1999) Fast Training of support vector machines using sequential minimal optimization, Advances in Kernel Methods-Support Vector Machines, Cambridge, pp. 185-208.
61. Statistics and Machine Learning Toolbox User's Guide (2017), The Mathworks Inc.
62. MATLAB 2016b, The MathWorks Inc., Natick, 2016.
63. MATLAB 2017a, The MathWorks Inc., Natick, 2017.
64. Mathworks.com(2017),NeuralNetworkToolbox:https://www.mathworks.com/products/neural-network.html.
65. Mathworks.com(2017), Statistics and Machine Learning Toolbox: https://www.mathworks.com/products/statistics.html.