Optimization of Propeller Performance Characteristics

Ante Čalić, Nenad Vulić, Igor Pavlović, Marko Katalinić

This paper presents the computational tool designed to calculate and optimize the performance characteristics of marine propellers, specifically Wageningen B-series propellers. The tool consists of two main modules: the thrust calculator, which estimates the thrust and efficiency of a propeller based on empirical and experimental data, and the optimization module, which utilizes brute-force and genetic algorithm methods to determine the optimal propeller configurations. By integrating basic input parameters such as ship speed, propeller diameter, pitch, and other essential characteristics, the tool is implemented to search for and find the solution for improving propulsion efficiency. The effectiveness of the tool is demonstrated through case studies on two vessels, a container ship and a special-purpose vessel, where significant improvements in propeller performance were identified. The tool as described in this paper represents a straight forward method for propulsion performance optimization in practical maritime operations.

KEYWORDS
~ Wageningen B-series
~ Propeller efficiency
~ Thrust calculator
~ Open water characteristics

University of Split, Faculty of Maritime Studies, Split, Croatia
e-mail: ipavlovic@pfst.hr
doi 10.7225/toms.v14.n02.s05

Received: 14 Mar 2025 / Revised: 11 Apr 2025 / Accepted: 16 Apr 2025 / Published: 20 Jul 2025

1. INTRODUCTION

One of the key elements in propulsion power prediction of marine vessels with a standard propulsion system is based on the propeller open water characteristics. Open water characteristics are typically determined through model-scale open water tests. Due to differences in flow regimes at varying Reynolds numbers, scaling procedures are necessary to adjust for the discrepancies between model-scale and full-scale characteristics (Carlton, 2019; Bhattacharyya, A. et al, 2015; W. P. A. van Lammeren, J. D. van Manen, 1969; Arapakopoulos et al., 2019; Molland and Hawksley, 1985).

Based on their characteristics, propellers are divided in different series each of different nature and extent. The numerical presented hereafter calculates propeller performance characteristics for Wageningen B-screw Series propellers, as this series is the one among the most commonly used. The purpose of these series in general is to provide specific design diagrams that offer information enabling the user to select the optimal propeller dimensions for real-world ship operations. (Carlton, 2019; Troost, 1951; Ekinci, 2011).

For the purpose of understanding the factors with the most influence on propeller performance, a characteristics calculation tool was created in Python programming language. Performance characteristics calculation tool presented consists of 2 parts: the thrust (T) calculator with the plotter for a specific propeller configuration and an optimizer module that calculates optimal configurations based on desired T.

The Wageningen series is one of the most comprehensive and widely adopted propeller series. Initially introduced by Troost (Troost, 1951) in the set of publications in the late 1940's, it remains commonly known as the 'Troost series' among practitioners. Over time, the series has been expanded to offer a complete range of fixed-pitch, non-ducted propeller designs. Table 1 shows range of parameters for Wageningen B-screw series.

Series	Number of propellers in series	Range of parameters			D (mm)	r_h/R	Cavitation data available	Notes
		Z	A_E/A_O	P/D				
Wageningen B-Series	$\simeq 120$	2-7	0.3-1.05	0.6-1.4	250	0.169	No	Four-bladed propeller has non-constant pitch distance

Table 1. Wageningen B-screw series parameters (Carlton, 2019)

A typical open water diagram for a fixed pitch propeller working in a non-cavitating environment is shown in Figure 1. with the advance coefficient (J) on the abscise and thrust and torque coefficients (KT), (KQ) and efficiency (η0) on the ordinate.

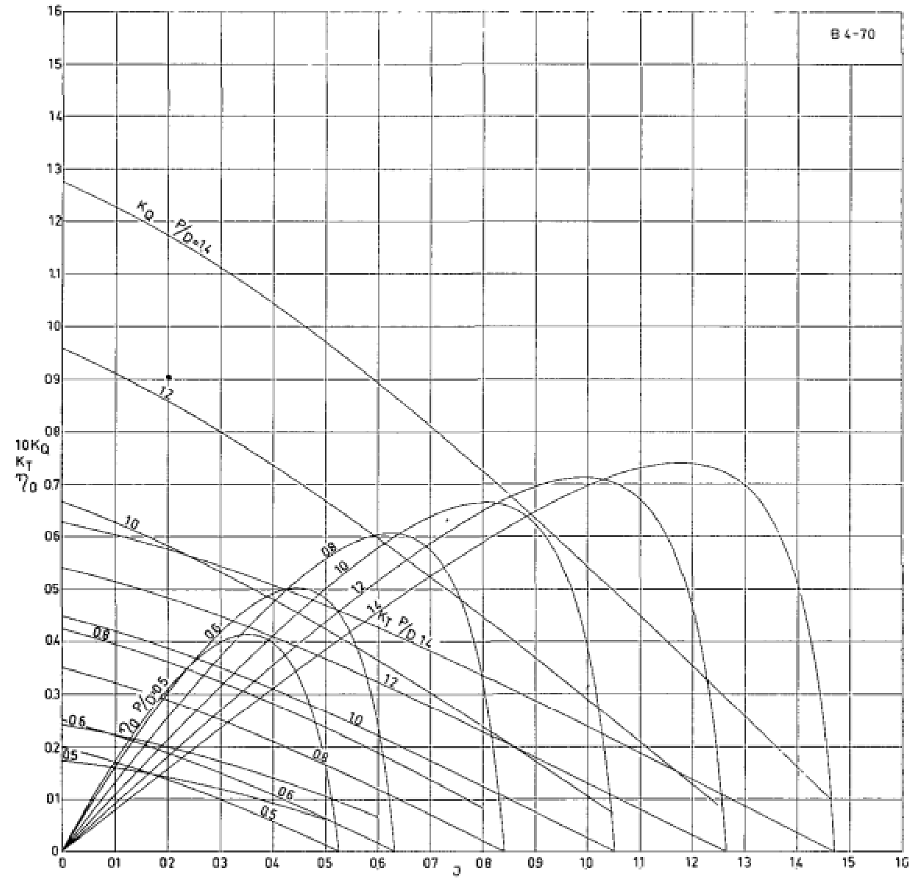

Figure 1. Open-water test results of B 4 – 70 screw series (Bhattacharyya, A. et al, 2015)

2. METHODOLOGY

2.1. Thrust calculator

The calculator module determines T and η_0 using empirical formulas for designing Wageningen B-series propellers. The open-water characteristics of the Wageningen B-series propellers for Reynolds number of 2×10^6 are represented as polynomial functions in equations (6) and (7) (Carlton, 2019). Input data and constants are presented in Table 2.

Input data:		
Ship speed	V_b	kn
Propeller revolutions	n	rpm
Propeller diameter	D	m
Pitch	P	m
Blade area ratio	A_E/A_O	
Number of blades	Z	
Wake fraction coefficient	w	
Thrust deduction coefficient	t	
Constants:		
Fluid density	ρ	kg/m³
Fluid kinematic viscosity	ν	m²/s

Table 2. Input data and constants

The moments and forces generated by the propeller are fundamentally expressed through a set of non-dimensional characteristics, which are universally applicable to a specific geometric configuration (Carlton, 2019). Expressions for polynomials representing the Wageningen B-screw series KT, KQ, J and η0 are shown in equations 1 to 4.

The required blade surface area to minimize cavitation risk can be determined using the expressions provided in Carlton (2019). However, cavitation calculations have been omitted in this research.

$$K_T = \frac{T}{\rho n^2 D^4} \tag{1}$$

$$K_Q = \frac{Q}{\rho n^2 D^5} \tag{2}$$

$$J = \frac{V_a}{nD} \tag{3}$$

$$\eta_O = \frac{K_T}{K_Q} \cdot \frac{J}{2\pi} \tag{4}$$

According to (Oosterveld and van Oossanen, 1975) the range of the series, represented as a matrix of blade number versus blade area ratio, is shown in Table 3, indicating that the series includes approximately 20 different blade area–blade number configurations.

Blade number (Z)	**Blade area ratio A_E/A_O**													
2	0.30													
3		0.35			0.50			0.65			0.80			
4			0.40			0.55			0.70			0.85	1.00	
5				0.45			0.60			0.75				1.05
6					0.50			0.65			0.80			
7						0.55			0.70			0.85		

Table 3. Wageningen B-screw series blade number and blade area correspondence (Carlton, 2019)

Given that input value is ship speed, the advance coefficient (*J*) can be calculated with the following equations:

$$J = V_b \cdot 0.5144 \cdot \frac{1-w}{n \cdot D} \tag{5}$$

and the Reynolds number:

$$Re = \frac{n \cdot D^2}{\nu} \tag{6}$$

From (Carlton, 2019) the open water characteristics of the series are presented as polynomial functions for a Reynolds number of $2 \cdot 10^6$ by equations (5) and (6).

$$K_Q = \sum_{n=1}^{47} \mathrm{Cn}(J)^{\mathrm{Sn}} (P/D)^{\mathrm{tn}}, (A_E/A_O)^{\mathrm{un}} (Z)^{\mathrm{vn}} \tag{7}$$

$$K_T = \sum_{n=1}^{39} \mathrm{Cn}(J)^{\mathrm{Sn}} (P/D)^{\mathrm{tn}}, (A_E/A_O)^{\mathrm{un}} (Z)^{\mathrm{vn}} \tag{8}$$

In (Carlton, 2019) the table of coefficients (table 6.6 on page 102) *s, t, u* and *v* for the K_T and K_Q polynomials that represent the Wageningen B-screw series for a Reynolds number of $2 \cdot 10^6$ was used to calculate the sums of K_T and K_Q. Once the sums of K_T and K_Q are calculated, from equation (1) thrust can be calculated as:

$$T = K_T \cdot (\rho n^2 D^4) \tag{9}$$

To extend this calculation further so that propeller characteristics can be predicted for other Reynolds numbers within the range $2 \cdot 10^6$ to $2 \cdot 10^9$ a set of corrections of the following form was derived:

$$K_T(Re) = K_T(Re = 2 \cdot 10^6) + \Delta\, K_T(Re) \tag{10}$$

$$K_Q(Re) = K_Q(Re = 2 \cdot 10^6) + \Delta\, K_Q(Re) \tag{11}$$

Where ΔK_T and ΔK_Q are described as:

$$\begin{aligned} \Delta K_T = {} & 0.000353485 - 0.00333758\,(A_E/A_O)J^2 - 0.00478125(A_E/A_O)(P/D)J \\ & + 0.000257792(\log Re - 0.301)^2(A_E/A_O)J^2 + 0.0000643192(\log Re - 0.301)(P/D)^6 J^2 \\ & - 0.0000110636(\log Re - 0.301)^2(P/D)^6 J^2 \\ & - 0.0000276305(\log Re - 0.301)^2 Z(A_E/A_O)J^2 \\ & + 0.0000954(\log Re - 0.301)Z(A_E/A_O)(P/D)J \\ & + 0.0000032049(\log Re - 0.301)Z^2(A_E/A_O)(P/D)^3 J \end{aligned} \tag{12}$$

$$\begin{aligned} \Delta K_Q = {} & -0.000591412 + 0.00696898(P/D) - 0.0000666654\,Z(P/D)^6 + 0.0160818(A_E/A_O)^2 \\ & - 0.000938091(\log Re - 0.301)(P/D) - 0.00059593(\log Re - 0.301)(P/D)^2 \\ & + 0.0000782099(\log Re - 0.301)^2(P/D)^2 + 0.0000052199(\log Re - 0.301)Z(A_E/A_O)J^2 \\ & - 0.00000088528(\log Re - 0.301)^2 Z(A_E/A_O)(P/D)J \\ & + 0.0000230171(\log Re - 0.301)Z(P/D)^6 - 0.00000184341(\log Re - 0.301)^2 Z(P/D)^6 \\ & - 0.00400252(\log Re - 0.301)(A_E/A_O)^2 + 0.000220915(\log Re - 0.301)^2(A_E/A_O)^2 \end{aligned} \tag{13}$$

According to equation (7), corrections for the Reynolds numbers in the range $2 \cdot 10^6$ to $2 \cdot 10^9$ need to be applied, considering that the final formula for thrust *T* is:

$$T = [K_T(Re = 2 \cdot 10^6) + \Delta\, K_T(Re)] \cdot (\rho n^2 D^4) \tag{14}$$

The torque *Q* is calculated in the same manner:

$$Q = \left[K_Q(Re = 2 \cdot 10^6) + \Delta\, K_Q(Re)\right] \cdot (\rho n^2 D^5) \tag{15}$$

With the T and Q for desired input calculated the calculation process finished and plotter depicts K_T, K_Q, η_0 and J on a diagram. In Figure 2. an example diagram for input values described in Table 4.

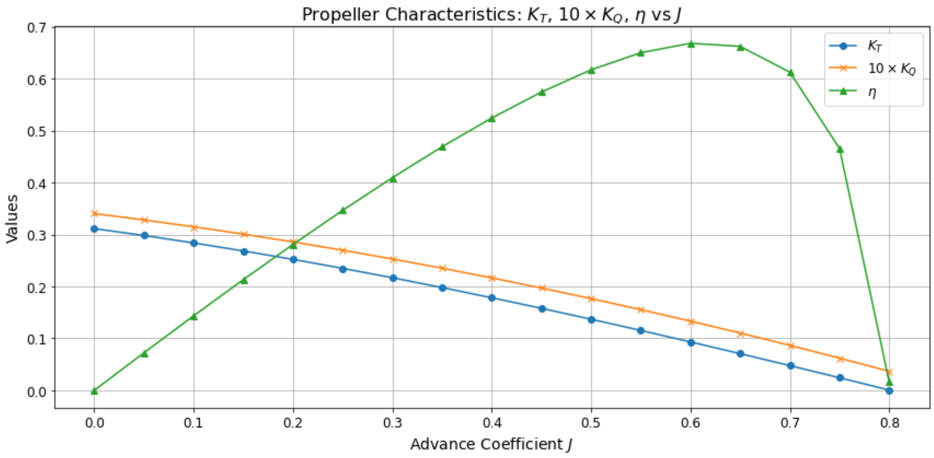

Figure 2. Example of K_T, K_Q, η_0 and J curves

	Unit	Value
V_b	knots	6.5
n	rpm	450
D	m	1.5
P	m	1.095
A_E/A_O		0.6
z		4

Table 4. Example values for T calculation and plot

3. OPTIMIZATION

Optimization has been conducted with two different methods. The first approach was carried out with a brute-force search method and the second with Genetic algorithm (GA) method. Both methods use the same formulations from the thrust calculator but with T as one of the inputs. Optimization tools are programmed to find the desirable T with the best possible η_0. In this paper are these optimizations presented through an example propeller, details of the input data in Table 5.

	Unit	Value
V_b	knots	6.5
n	rpm	1250
A_E/A_O		0.75
z		3
w		0
t		0
Desired T	kN	1.37

Table 5. Example of input data for both optimization methods

3.1. Brute force method

The brute force method guarantees that the best solution will be found if it exists within the search space. This method works in a way that it finds all the combinations of the desired variables and then compares them to check which one meets the criteria and the best solution is presented. It can be concluded that for large search spaces this can often be computationally expensive and inefficient. Despite that, the brute force method is a valuable technique in situations where the search space is reasonable.

Figure 3. shows the code providing the idea for brute force method, specifically the part of the optimization process. Starting from line 66, the other loop iterates over possible diameters of the propeller from 0.1 to 2.0 meters, increasing in steps of 0.01 meters. For each diameter it calculates *J* and *Re* and checks if *Re* is in range $2{\cdot}10^6$ to $2{\cdot}10^9$. In line 70 for each diameter, it iterates over possible pitches in a similar range. After *T* is calculated for different iterations of *D* and *P*, in line 76 condition specifies the threshold for how close the calculated thrust needs to be to the desired thrust for the parameters to be considered acceptable, in this case 0.05 of the desired value. Other combinations are filtered out in this part of code. After this point η_0 (*eta*0) is simply checked to be in realistic bounds and the best η_0 singled out and printed as a solution.

```
66  for D in np.arange(0.1, 2.0, 0.01):
67      J = Vb * 0.5144 * (1 - w) / (n * D)
68      Rn = (n * D**2) / ni
69      if 2e6 <= Rn <= 2e9:
70          for P in np.arange(0.1, 2.0, 0.01):
71              delta_Kt = calculate_delta_Kt(Ae_Ao, P, D, J, Rn, z)
72              delta_Kq = calculate_delta_Kq(Ae_Ao, P, D, Rn, z, J)
73              sum_Kt = sum(C * (J**s) * ((P/D)**t) * (Ae_Ao**u) * (z**v) for C, s, t, u, v in Kt_coeffs)
74              sum_Kq = sum(C * (J**s) * ((P/D)**t) * (Ae_Ao**u) * (z**v) for C, s, t, u, v in Kq_coeffs)
75              T = (sum_Kt + delta_Kt) * (ro * D**4 * n**2)
76              if abs(T - desired_T) < 0.05:  # T deviation tolerance
77                  M = (sum_Kq + delta_Kq) * (ro * D**5 * n**2)
78                  eta0 = J / (2 * math.pi) * ((sum_Kt + delta_Kt) / (sum_Kq + delta_Kq))
79                  if 0.1 <= eta0 <= 0.9 and eta0 > best_eta0:
80                      best_eta0 = eta0
81                      best_params = (D, P, Ae_Ao, T, M, eta0)
```

Figure 3. Brute force source code

3.2. Genetic algorithm method

Genetic algorithms are a class of optimization techniques based on the principles of natural selection and genetics. The algorithm works in a way that it “creates a population” of possible solutions to the problem after which they “evolve” over multiple generations to find more suitable solution (Haupt, 1995).

For this purpose, a Python library “geneticalgoritam” was utilized, the library is distributed for implementing standard and elitist (GA). This package solves combinatorial, continuous and mixed optimization problems with discrete, continuous and mixed variables (*The Python Package Index (PyPI)*, November 2024)(Beasley, Bull and Martin, 1993).

In Figure 4. part of the code from the GA module can be seen. All the preceding calculations are the same as in brute force approach. The first difference observed is in line 94 in the part of a fitness function that evaluates potential solutions. The η_0 (eta0) is defined with the use of “-10“ as a multiplier that directly impacts how the fitness values are calculated. When discussing genetic algorithms, where lower fitness values are often preferable (minimization problem), this negative coefficient inversely scales the efficiency. This means higher efficiencies, which are inherently positive, are transformed into more negative values (Haupt, 1995). The magnitude of 10 determines how sensitive the fitness score is to changes in efficiency. A larger absolute value would make the fitness score more responsive to small changes in efficiency, amplifying its impact on the overall fitness evaluation. The term “80*abs (*T*-desired_*T*)” adds a penalty based on the absolute difference between the calculated thrust (*T*) and the desired thrust (desired_*T*). The coefficient 80 amplifies this difference, indicating that deviations from the desired thrust result in a significant increase in the fitness value, representing a worse outcome.

Bounds are set as in the brute force method, and from line 103 in Figure 4. algorithm parameters are set. The parameter can be fine-tuned in order to achieve the best results. The parameters are as follows: maximum iterations number, population size, mutation probability, elitism ratio, crossover probability and type, parents’ portion.

```
    if 0.1 <= eta0 <= 0.9:
        result = -10 * eta0 + 80 * abs(T - desired_T)
    else:
        result = float('inf')

    return result

#  bounds for D and P
bounds = [(0.1, 2.0), (0.1, 2.0)]

algorithm_param = {
    'max_num_iteration': 1000,
    'population_size': 100,
    'mutation_probability': 0.1,
    'elit_ratio': 0.01,
    'crossover_probability': 0.8,
    'parents_portion': 0.3,
    'crossover_type': 'uniform',
    'max_iteration_without_improv': None
}
```

Figure 4. Genetic algorithm source code

The population size parameter specifies the number of individuals in each generation. A larger population size allows the GA to explore a more diverse set of solutions, however, it also increases computational cost (Haupt, 1995)(Beasley, Bull and Martin, 1993). Mutation probability determines the likelihood that any given gene in an individual will be randomly altered, which helps to avoid local minimums and ensures exploration of the solution space, value of 0.1 means that each gene has a 10% chance of being mutated. The elitism ratio specifies the portion of the top-performing individuals from the current generation that should be kept and carried over directly to the next generation without mutation. This ensures that the best solutions are not lost. Crossover probability and type set the likelihood that recombination will occur between two selected parents to produce new offspring. A high crossover probability, 80% in this case, promotes the mixing of genetic material and the creation of new solutions, type determines the method used to combine the genetic information of two parents during reproduction. "Uniform" crossover treats each gene independently and decides randomly whether it will come from the first parent or the second parent. Parent’s portion parameter defines the fraction of the population that will be selected to produce the next generation. A parent’s portion of 0.3 means that 30% of the best-performing individuals are chosen for reproduction for the next generation. This helps to focus the search on promising areas of the solution space. In figures 5 and 6 objective function and η_0 are shown as plots over iterations/generations.

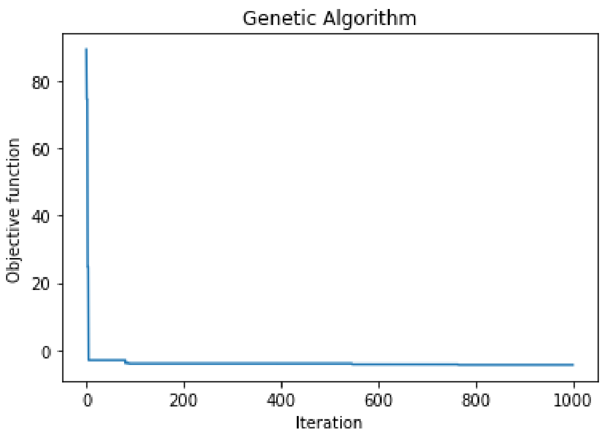

Figure 5. Objective function over iterations

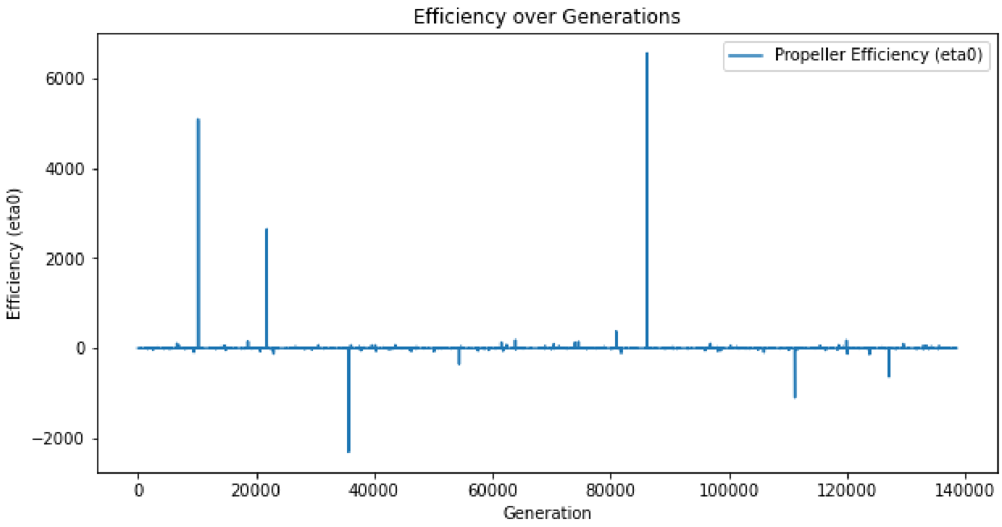

Figure 6. η_0 over generations

In Table 6. results of both optimization methods for an example propeller described in Table 5. have been presented for comparison.

	Unit	Brute force	Genetic algorithm	Numerical difference	Percentage difference
D	m	0.379	0.388	0.009	2.32%
P	m	0.269	0.267	0.002	0.74%
T	kN	1.329	1.377	0.048	3.49%
$\boldsymbol{\eta_0}$		0.512	0.509	0.003	0.59%

Table 6. Example of input data for both optimization methods

From Table 6. it is obvious that both optimization methods showed similar results with only a 0.03 difference in η_0. GA is more precise in T calculation but brute force method calculated *T* sufficiently given the desired thrust for the parameters to be considered acceptable is set to 0.05 in the optimization settings.

4. ANALYSIS

Verification has been conducted on two example vessels for which the propeller data was available. Example one is the container ship 11400 TEU. Propeller and sea trial data appended at the end of this document. Table 7. shows input data for the first vessel. The wake fraction coefficient was unknown for both ships and was excluded from the calculation.

	Unit	Value
$\boldsymbol{V_b}$	knots	27.73
n	rpm	109.5
$\boldsymbol{A_E/A_O}$		0.8409
D	m	8.9
P	m	8.318
z		6
w		0
t		0

Table 7. Container ship 11400 TEU input data

From this data the thrust is calculated to be 1331.94 kN and the efficiency 0.66. The calculated *T* value then becomes the input value for the optimization programs as desired *T,* to be achieved with the highest efficiency. Results of both optimization methods are presented in Table 8.

	Unit	**Brute force**	**Genetic algorithm**	**Numerical difference**	**Percentage difference**	**Installed**	$\frac{Brute\ force - Instaled}{Instaled} \cdot 100$
D	m	8	7.96	0.04	0.5%	8.9	10.1 %
P	m	8.7	8.72	0.02	0.23%	8.3	4%
T	kN	1339.84	1331.86	7.98	0.6%	1331.94*	0.006%
η_0		0.74	0.74	0	0	0.66*	8%

* Values of *T* and η_0 in Installed column are calculated, not measured.

Table 8. Results of optimization processes for the container ship 11400 TEU

The ship in the second example is the general cargo ship 761 GT. The input data for this example is shown in Table 9.

	Unit	**Value**
V_b	knots	11
n	rpm	450
A_E/A_O		0.6
D	m	1.5
P	m	1.095
z		4
w		0
t		0

Table 9. General cargo ship 761 GT input data.

Given the input data *T* is calculated at 36.69 kN with η_0 of 0.635. The same process is repeated as in first example and the results of optimization is shown in Table 10. Optimization has been done on both examples with same settings only different input data and the tolerances for *T.* Since in the first example desired *T* was 1331.94 the tolerance for *T* deviation was ±10 kN while in the second example for a much smaller desired *T* the tolerance was set to ±0.9 kN.

	Unit	Brute force	Genetic algorithm	Numerical difference	Percentage difference	Installed	$\frac{Brute\ force - Instaled}{Instaled} \cdot 100$
D	m	1.48	1.42	0.06	4.05%	1.5	1.3%
P	m	1.11	1.16	0.05	4.31%	1.095	1.3%
T	kN	36.63	36.68	0.05	0.14%	36.69*	0.89%
η_0		0.635	0.631	0.004	0.63%	0.635*	0%

* Values of *T* and η_0 in Installed column are calculated, not measured.

Table 10. Results of optimization processes for the general cargo ship 761 GT

From the results of the second example of optimization it is obvious that the optimization could not find any solutions with the higher η_0. The models provide solutions that vary *T* values within the selected range with the same η_0. In first example the program was calculated the possibility of an 8% increase in propeller efficiency. For the second example it can be concluded that the current propeller is optimal for the desired purpose within the limits of this calculation.

5. CONCLUSION

Performance characteristics calculation tool was developed in Python programming language and presented in this report. The calculation tool uses empirical methods based on Wageningen B-screw series propellers. The tool consists of 2 parts: the thrust calculator and an optimizer module that calculates optimal configurations based on desired thrust. Together this package is intended to be an engineer's "at hand" tool that can be used in research for better understanding of propeller performance under varying conditions.

By integrating empirical models with optimization algorithms like Genetic Algorithms (GAs), this tool provides an efficient means to explore a wide design space and identify optimal propeller geometries for specific operational requirements. Example of this can be in the early-stage design of propulsion systems, where rapid estimation of optimal propeller parameters (e.g., diameter, pitch, and area ratio) can inform decision-making before starting costly and time-consuming CFD simulations or model tests.

From the results obtained from the input data of two example vessels (the container ship 11400 TEU and the general cargo ship 761 GT) comparison was made with regards to installed propellers. In the first example, once calculated the thrust and the efficiency of the current setup, the optimization programs calculated that with a change in propeller diameter and respectively in pitch, efficiency could be improved by 8%. In reality this ship's propeller is probably optimised to a finer degree. Difference calculated here is mostly contributed by the wake fraction coefficient being unknown. In the case of the general cargo ship 761 GT the optimization could not find a configuration with better efficiency suggesting that once the real wake fraction would be input there would still be room for improvement.

Cavitation calculations used to determine blade surface area that is required to keep the risk of cavitation to a minimum have been neglected in this research and left for future development of the program. In order for the optimization programs to find a wider range of solutions it is also suggested for future improvements to incorporate more parameters such as blade area ratio, RPM or number of blades and for more accurate solutions the wake should be known.

CONFLICT OF INTEREST

The authors declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

REFERENCES

Arapakopoulos, A. et al. (2019) 'Parametric models for marine propellers', Ocean Engineering, 192 (September 2021). Available at: https://doi.org/10.1016/j.oceaneng.2019.106595

Beasley, D., Bull, D.R. and Martin, R.R. (1993) 'ORCA-Online Research @ Cardiff an overview of genetic algorithms: Part 1, fundamentals', Genetic Algorithms, 15 (2), pp. 58–69

Bhattacharyya, A. et al. (2015) 'Scale effects on a 4-bladed propeller operating in ducts of different design in open water', Proceedings of the 4th International Symposium on Marine Propulsors (SMP'15) (June 2024). Available at: http://www.marinepropulsors.com/proceedings/2015/WB2-2.pdf

Carlton, J. (2019) Marine propellers and propulsion (4th edn). Burlington: Butterworth-Heinemann

Ekinci, S. (2011) 'A practical approach for design of marine propellers with systematic propeller series', Brodogradnja, 62 (2), pp. 123–129

Haupt, R.L. (1995) 'An introduction to genetic algorithms for electromagnetics', IEEE Antennas and Propagation Magazine, 37 (2), pp. 7–15. Available at: https://doi.org/10.1109/74.382334

Molland, A.F. and Hawksley, G.J. (1985) 'An investigation of propeller performance and machinery applications in wind assisted ships', Journal of Wind Engineering and Industrial Aerodynamics, 20 (1–3), pp. 143–168. Available at: https://doi.org/10.1016/0167-6105(85)90016-9

Oosterveld, M.W.C. and van Oossanen, P. (1975) 'Further computer-analyzed data of the Wageningen B-screw series', International Shipbuilding Progress, 22 (251), pp. 251–262. Available at: https://doi.org/10.3233/isp-1975-2225102

The Python Package Index (PyPI) (2024) geneticalgorithm. Available at: https://pypi.org/project/geneticalgorithm/

Troost, L. (1951) 'Open water test series with modern propeller forms III, two bladed and five bladed propellers - Extension of three and four bladed B-series', Transactions of the North East Coast Institution of Engineers and Shipbuilders (NECIES), 67

van Lammeren, W.P.A., van Manen, J.D. and Oosterveld, M.W.C. (1969) 'The Wageningen B-screw series', Society of Naval Architects and Marine Engineers, 77 (8), p. 43