Parallel computing efficiency of SWAN 40.91

Christo Rautenbach[1, 2, 3, 4], Julia C. Mullarney[4], Karin R. Bryan[4]

[1] Institute for Coastal and Marine Research, Nelson Mandela University, South Africa
[2] Department of Oceanography and Marine Research Institute, University of Cape Town, South Africa
[3] Research and development, MetOcean (a division of the Metrological Service), Raglan, New Zealand
[4] Environmental Research Institute, University of Waikato, Hamilton, New Zealand

Correspondence to: Christo Rautenbach (rautenbachchristo@gmail.com)

Abstract. Effective and accurate ocean and coastal wave predictions are necessary for engineering, safety, and recreational purposes. Refining predictive capabilities is increasingly critical to reduce the uncertainties faced with a changing global wave climatology. Simulating WAves in the Nearshore (SWAN) is a widely used spectral wave modelling tool employed by coastal engineers and scientists, including for operational wave forecasting purposes. Fore- and hindcasts can span hours to decades and a detailed understanding of the computational efficiencies is required to design optimized operational protocols and hindcast scenarios. To date, there exists limited knowledge on the relationship between the size of a SWAN computational domain and the optimal amount of parallel computational threads/ cores required to execute a simulation effectively. To test ~~this~~the scalability, a hindcast cluster of 28 computational threads/ cores (1 node) was used to determine the computation efficiencies of a SWAN model configuration for southern Africa. The model extent and resolution emulate the current operational wave forecasting configuration developed by the South African Weather Service (SAWS). We implemented and compared both OpenMP and the Message Passing Interface (MPI) distributing memory architectures. Three sequential simulations (corresponding to typical grid cell numbers) were compared to various permutations of parallel computations ~~via~~ using the speed-up ratio, time saving ratio and efficiency tests. Generally, a computational node configuration of 6 threads/ cores produced the most effective computational set-up based on wave hindcasts of one-week duration. The use of more than 20 threads/ cores resulted in a decrease in speed-up ratio for the smallest computation domain, owing to the increased sub-domain communication times for limited domain sizes.

Keywords: SWAN, Parallel computing, Forecasting, Hindcasting, South Africa

1 Introduction

The computational efficiency of Met-ocean (Metrological-Ocean) modelling has been the topic of ongoing deliberation for decades. The applications range from long-term atmospheric and ocean hindcast simulations to the fast responding simulations related to operational forecasting. Long-duration simulations are usually associated with climate change related research, with simulation periods of at least 30-years across multiple spatial and temporal resolutions needed to capture key oscillations (Babatunde et al., 2013). Such hindcasts are frequently used by coastal and offshore engineering consultancies for purposes such as those related to infrastructure design (Kamphuis, 2020), or environmental impact assessments (Frihy, 2001; Liu et al., 2013).

Operational (or forecasting) agencies are usually concerned with achieving simulation speeds that would allow them to accurately forewarn their stakeholders of immediate, imminent and upcoming met-ocean hazards. The main stakeholders are usually other governmental agencies (e.g. disaster response or environmental affairs departments), commercial entities and the public. Both atmospheric and marine forecasts share similar numerical schemes that solve the governing equations and thus share a similar need in computational efficiency. Fast simulation times are also required for other forecasting fields such as hydrological dam-break models (e.g. Zhang, et al., (2014)). Significant advancement in operational forecasting can be made by examining the way in which the code interfaces with the computation nodes, and how results are stored during simulation. Numerous operational agencies (both private and public) makes use of Simulating Waves in the Nearshore (SWAN) to predict nearshore wave dynamics (refer to Genseberger & Donners, (2020) for details regarding the SWAN numerical code and solution schemes). These agencies include the South African Weather Service (e.g. Rautenbach, et al., (2020)), MetOcean Solutions (a division of the Metrological Office of New Zealand) (e.g. de Souza, et al., (2020)), the United Kingdom MetOffice (e.g. O'Neill et al., (2016)) and the Norwegian Metrological Service (e.g. Jeuring, et al., (2019). In general, these agencies have substantial computational facilities but nonetheless still face the challenge of optimizing the use of their computational clusters between various models (being executed simultaneously). These models may include atmospheric models (e.g. the Weather Research and Forecasting (WRF) model), Hydrodynamic models (e.g. Regional Ocean Modeling System (ROMS) and the Semi-implicit Cross-scale Hydroscience Integrated System Model (SCHISM)) and spectral waves models (e.g. Wave Watch III (WW3) and SWAN (Holthuijsen, 2007; The SWAN team, 2019b; The SWAN team, 2019)). Holthuijsen, (2007) presents a theoretical background to the spectral wave equations, wave measurement techniques and statistics as well as a concluding chapter ~~linking the book~~the theoretical analysis to the SWAN numerical model. There must also be a balance between hindcast and forecast priorities and client needs. Some of these agencies use a regular grid (instead of irregular grids (e.g. Zhang, et al., (2016)), with nested domains in many of their operational and hindcast projects. Here, we focus only on the computational performance of a structured regular grid (typically implemented for spectral wave models).

Kerr et al., (2013) performed an inter-model comparison of computational efficiencies by comparing SWAN, coupled with ADCIRC, and the NOAA official storm surge forecasting model, Sea, Lake, and Overland Surges from Hurricanes (SLOSH~~),~~); however, did not investigate the optimal thread usage of a single model. Other examples of a coupled wave and storm surge model computational benchmarking experiments include Tanaka, et al, (2011) and Dietrich et al., (2012) ~~for~~ who used a unstructured meshes to simulate waves during Hurricanes Katrina, Rita, Gustav and Ike in the Gulf of Mexico~~an Golf~~. Results from ~~These~~ these models were ~~also~~ present~~ their results~~ed on a log-log scale and their experimental design tested computational thread numbers not easily obtainable by smaller agencies and companies. The latter rather require sequential versus paralleled computational efficiencies using smaller scale efficiency metrics. Genseberger & Donners, (2015), explored the scalability of SWAN using a case study focused on the Wadden Sea in the Netherlands. By investigating the efficiency of both the OpenMP (OMP) and MPI version of the then current SWAN, they found that the OpenMP was more efficient on a single node. They also proposed a hybrid version of SWAN, to combine the strengths of both implementations of SWAN: using OpenMP to more optimally share memory and MPI to distribute memory over the computational nodes.

Here we build on the case study of Genseberger & Donners using results produced in the present study for southern Africa, to answer the following research questions: 1) when using SWAN, is it always better to have as many threads/ cores as possible available to solve the problem at hand? 2) What is the speed-up relationship between number of threads/ cores and computational grid size? 3) At what point (number of threads/ cores) does the domain sub-communications start to make the whole computation less effective? 4) What is the scalability of a rectangular grid, SWAN set-up?

Methodology and background

Details of the model configuration can be found in ~~(~~Rautenbach, et al., (2020~~ (a))~~a) and ~~(~~Rautenbach, et al., (2020~~ (~~b~~)~~). The computational domain (refer to Figure 1) and physics used here were the same as presented in those studies.

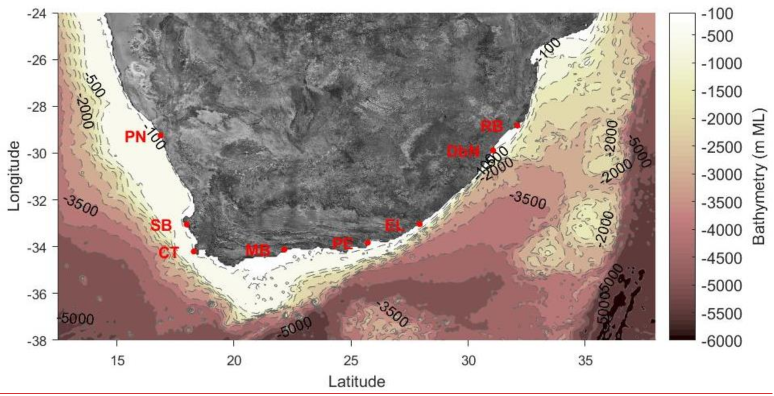


Figure 1: SWAN model extent and associated bathymetry. The location of all the major coastal towns are also provided via acronyms as follows: Port Nolloth (PN), Saldanha Bay (SB), Cape Town (CT), Mossel Bay (MB), Port Elizabeth (PE), East London (EL), Durban (DN) and Richards Bay (RB).

All computations were performed on Intel Xeon Gold E5-2670, 2.3GHz computational nodes. Twenty-eight threads/ cores each with 96 GB RAM were used with~~and~~ 1~~-~~ Gbyte/s inter-thread communication speed. SWAN 40.91 was implemented with the Van der Westhuysen whitecapping formulation (van der Westhuysen, et al., 2007) and Collins bottom friction correlation (Collins, 1972) with a coefficient value of 0.015. Fully spectral wave boundary conditions were extracted from a global Wave Watch III model at 0.5 geographical degree resolution.

Formatted: Centered

Formatted: English (United Kingdom)

Here, the main aim was not the validation of the model ~~was not the main aim~~ but rather ~~the~~ to quantify the relative computational scalabilities, as described at the end of the previous section. However, it should be noted that no nested domains were employed during the present study. Only the parent domain was used as a measure for scalability. The computational extent given in Rautenbach, Barnes, et al., (2020) (a) and Rautenbach, et al., (2020) (b) contains numerous non-wet grid cells that are not included in the computational expense of the current study. In Table 1~~Table 1~~, the size of the computational domain and resolution, together with the labelling convention are given. For clarity, we define the resolutions as low, medium and high, denoted L, M and H, respectively, in the present study (noting that given the domain size, these resolutions would be classified as intermediate to high regional resolution for operational purposes).

Formatted: Font: Not Bold

Table 1: SWAN grid resolution, grid cell numbers and reference labels.

Label	SWAN grid resolution	Computational grid cell number
L	0.1000	31 500
M	0.0625	91 392
H	0.0500	142 800

The test for scalability ability of a model used here was the ability to respond to an increased number of computations with an increasing amount of resources. In the present study these resources are computational threads/ cores. An arbitrary week of computations were performed to assess model performance. Model spin-up was done via a single stationary computation. The rest of the computation was performed using a non-stationary computation using an hourly time-step~~. This~~, which implied wind-wave generation within the model occurred on the timescale of the wind forcing resolution. The grid resolutions used in the present study corresponded to 0.1, 0.0625 and 0.05 geographical degrees. Local bathymetric features were typically resolved through downscaled, rotated, rectangular grids, ~~like~~ following the methodology employed by Rautenbach, et al., (2020) (a). A nested resolution increase of more than 5-times is also not recommended (given that the regional model is nested in the global Wave Watch III output at 0.5 geographical degree resolution, ~~refer to~~ (Rautenbach, et al., 2020) (a). Given these constraints, these resolutions represent realistic and typical SWAN model set-up, for both operational and hindcast scenarios.

The three main metrics for estimating computational efficiency are: the *Speed-up*, *Time saving* and *Efficiency ratios*. A~~The~~ fourth parameter, and arguably the most important, is the *Scalability* and is estimated using the other three parameters as metrics.

The *Speed-up* ratio is given as:

$$S_p = T_1 / T_p \quad (1)$$

where T_1 is the time in seconds it takes for a sequential computation on one thread and T_p is the time a simulation takes with *p* computational threads/ cores (Zhang et al., 2014).

The *Time saving* ratio is given by:

$$T_1 S_p = (T_1 - T_p)/T_1 \quad (2)$$

and the *Efficiency* ratio ~~follow with the same variables definitions~~is defined as:

$$E_p = S_p / p. \quad (3)$$

The Scalability of SWAN was tested based on the Speed-up ratios for the grid resolutions in Table 1~~Table 1~~.

Formatted: Font: Not Bold

Zafari, Larsson, & Tillenius, (2019) recently presented some of the first results investigating the effect of different compilers on the scalability of a shallow water equation solver. Their experiments compared a model compiled with GNU Compiler Collection (gcc) 7.2.0 and linked with OpenMPI and Intel C++ compilers with Intel MPI for relatively small computational problems. Their numerical computation considered models with 600K, 300K and 150K grid cell sizes (what they called matrix size). These computational grid sizes were deemed “small”, but they still acknowledged the significant computational resources required to execute geographical models of this size due to the large number of time steps ~~usually involved~~undertaken to solve these problems.

From a practical point of view, regular SWAN grids will rarely be used in dimensions exceeding the resolutions presented in the previous section. The reason for this statement is twofold: 1) to downscale a spectral wave model from a global resolution to a regional resolution ~~may~~ should not exceed a five-times refinement factor and 2) when reasonably high~~er~~ resolutions are required in the nearshore (to take complex bathymetric features into account), nested domain are preferred. The reasoning will be different for an unstructured grid approach (Dietrich et al., 2012). Given these limitations ~~on~~ with the widely used structured SWAN grid approach, SWAN grids will almost exclusively be deemed as a low spatial computational demand model. Small tasks create a sharp drop in performance via the Intel C++ compiler due to the “work stealing” algorithm, aimed at balancing out the computational load between threads/ cores (Zafari et al., 2019). In this scenario, the threads/ cores compete against each other resulting in an unproductive simulation. In our experiments, each task performed via Intel was approximately 13-times faster but the overall performance was 16-time slower than the equivalent gcgc~~-~~compiled version of the compiled shallow water model presented by Zafari et al. (2019).

Results

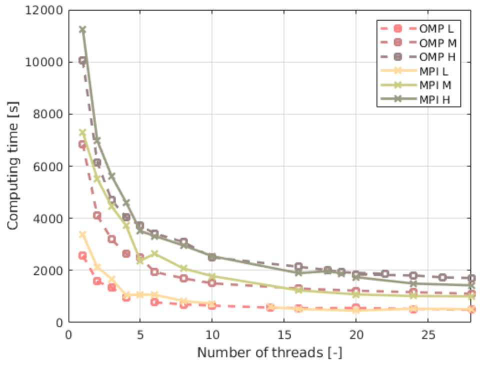
In Figure 221~~Figure 2Figure 1~~, the computational scalability of SWAN is given as a function of number of computational threads/ cores. ~~In~~ Figure 221~~Figure 2Figure 1~~ (a) shows the computational time in seconds and ~~is presented.~~ h~~H~~ere the model resolutions grouped together with not much differentiation between them. These results also highlight the need for performance metrics, like described in the previous section. From Figure 221~~Figure 1~~ (b) the MPI version of SWAN is more efficient for all the computational domain sizes. There is also a clear grouping between OMP and MPI. Figure 221~~Figure 2Figure 1~~ (c) presents the speed-up ratios and clearly indicates that the MPI version of SWAN outperforms the OMP version. The closer the result are to the 1:1 line, the better the scalability.

Formatted: Font: Not Bold

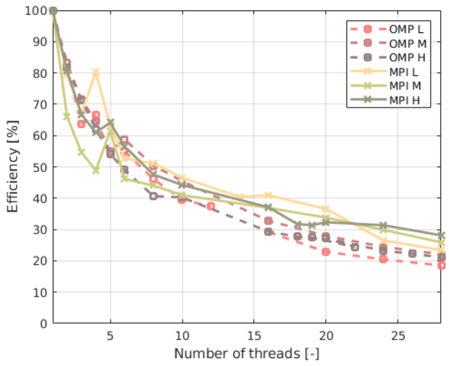
Formatted: Font: Not Bold

Formatted: Font: Not Bold

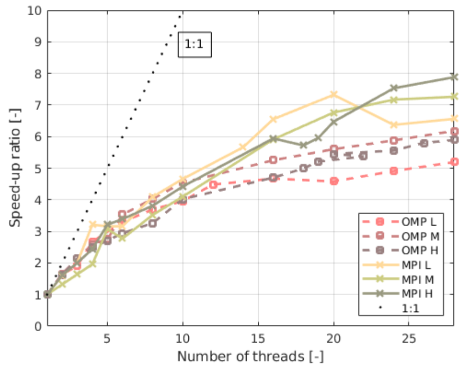
Formatted: Font: Not Bold



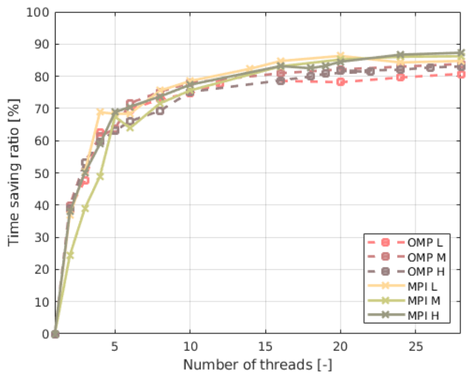
(a)



(b)



(c)



(d)

Figure 22~~1~~: Model performance as a function of the number of computational threads/ cores. (a) Computing time in seconds, (b) Efficiency (Equation (3)), (c) Speed-up ratio (Equation (1)) and (d) the Time saving ratio (Equation (2)).

Near linear speed up is observed for a small number of computational threads/ cores. This result agrees with th~~e results ~~ose reported by Zafari et al., (2019). In Figure 221~~Figure 2Figure 1~~ (d) the ~~same~~ results are expressed ~~obtained~~ via the time saving ratio. In this case, the curves start to asymptote~~Here a clear and district flattening down is observed~~ with thread counts larger than approximately 6.

Formatted: Font: Not Bold

Discussion

The behaviour noted in the results is similar to the dam breaking computational results reported by S. Zhang et al., (2014). Genseberger & Donners, (2020) presents the latest finding on the scalability and benchmarking of SWAN. However, their focus was quantifying the performance of their new hybrid version of SWAN. In their benchmarking experiments (for the Wadden Sea, in the Netherlands), they obtained different results to Figure 221~~Figure 2Figure 1~~ (a), with OMP generally producing faster wall-clock computational times. They also considered the physical distances between computational threads/ cores and found that this parameter has a negligible effect compared to differences between OMP ~~vs~~ and MPI, over an increasing number of threads/ cores. Their benchmarking also differed from the results presented here as they only provided results as a function of node number. Each one of their nodes consisted of 24 threads/ cores. In the present study, the benchmarking of a single node (28 threads/ cores) is evaluated compared with a serial computation on a single thread. For benchmarking, without performance metrics, they found that the wall clock times, for the iterations and not a full simulation, reached a minimum (for large computational domains) at 16 nodes (16 × 24 threads/ cores) for the MPI SWAN and 64 nodes (64 × 24 threads/ cores) for the hybrid SWAN. These results were based on using the Cartesius 2690 v3 (Genseberger and Donners, 2020). With the hybrid SWAN, the optimal wall-clock time turn point, for iterations, increased with increased number of computational cells. All of the reported turn points (optimal points) occurred at node counts well above 4 nodes (4 × 24 threads/ cores). The wall-clock performance estimation of Genseberger & Donners, (2015) did however indicate similar result to those presented in Figure 2~~1~~ (a), with OMP running faster than MPI for small thread/ core counts. For larger thread/ core counts MPI performs better in the present study. This difference in performance is probably related to the particular hardware configuration (Genseberger and Donners, 2015). It must still be noted that with an increased number of nodes, and thus threads/ cores, the total computational time should continue to decrease up until the point where the internal domain decomposition, communication efficiencies, starts to outweigh the gaining of computational power. Based on results of Genseberger & Donners, (2020), we can estimate that, for our node configuration and region of interest, the communication inefficiencies will become dominant at approximately 16 nodes (16 × 24 threads/ cores). One of the possible explanations for the none-perfect speed-up observed in Figure 2~~1~~ (c) is related to the computational domain partition methods used~~. This also closely relates to~~, and the wet and dry (or active and inactive) points definitions in the model. In the present study the dry points were the

Formatted: Font: Not Bold

bathymetry or~~/~~ topography values above Mean Sea Level (MSL). The employed partition method is currently stripwise because of the underlying parallel technique, namely the wavefront method (Genseberger and Donners, 2015; Zijlema, 2005). The stripwise partition is thus potentially not the most effective method~~s~~ to optimize speed-up. In the present study ~~it~~ this partition leads to an optimal point around 6 threads/ cores without losing too much parallel efficiency. In general, increasing the number of threads/ cores will still produce results faster, but in an increasingly inefficient manner. This trend is clear from Figure 221~~Figure 2~~ (c) and (d) where the total computational time (speed-up ratio, time saving ratio) does not scale linearly with the increasing number of threads/ cores. The ideal scenario (linear scalability) would be if the computational results followed the 1:1 line in Figure 221~~Figure 2~~ (c). In Figure 221~~Figure 1~~ (d) the non-linear, flattening ~~down~~ of the time saving ratio is also evident, although ~~it~~the ratio still slightly increases beyond 6 threads/ cores. This result implies the total computational time will marginally decrease with increasing number of threads/ cores. This marginal decrease in computational time could however still make significant differences in the total simulation times when extensive simulation periods are considered.

Formatted: Font: Not Bold, English (United Kingdom)

Formatted: Font: Not Bold, English (United Kingdom)

Formatted: Font: Not Bold, English (United Kingdom)

Conclusion

The present study investigated the scalability of SWAN, a widely used spectral wave model. Three typical wave model resolutions were used for these purposes. Both the OpenMP (OMP) and the Message Passing Interface (MPI) implementations of SWAN were tested. The scalability is presented via three performance metrics: the efficiency, speed-up ratio and the timesaving ratio. The MPI version of SWAN outperformed the OMP version based on all three metrics. The MPI version of SWAN performed best with the largest computational domain resolution, resulting in the highest speed-up ratios. The time saving ratio indicated a decrease after approximately six computational threads/ cores. This result suggests that six threads/ cores are the most effective configuration for executing SWAN. The largest increases in speed-up and efficiency was observed with small thread counts. According to Genseberger & Donners, (2020), computational times decrease up to ~16 nodes (16 × 24 threads/ cores), indicating the wall-clock optimal computational time for their cases study. This result suggests that multiple nodes will be required to reach the optimal wall-clock computational time – even though this turn point might not be the most efficient computational configuration. Ultimately, the efficiencies recommended here can improve operational performance substantially, particularly when implemented over the range of modelling software needed to produce useful metocean forecasts. Future studies might consider investigating the scalability employing a gcc compiler.

Code/Data availability

The open source version of SWAN was run for the purposes of the present study. SWAN maybe be downloaded from here: http://swanmodel.sourceforge.net/. To ensure a compatible version of SWAN remains available, the current, latest version of SWAN is permanently archive here: https://hdl.handle.net/10289/14269. The bathymetry used for the present study may be downloaded here: https://www.gebco.net/ and the wind forcing may be found here: https://climatedataguide.ucar.edu/climate-data/climate-forecast-system-reanalysis-cfsr.

~~The open source version of SWAN was run for the purposes of the present study. SWAN maybe be downloaded from here: http://swanmodel.sourceforge.net/. The bathymetry used for the present study may be downloaded here: https://www.gebco.net/ and the wind forcing may be found here: https://climatedataguide.ucar.edu/climate-data/climate-forecast-system-reanalysis-cfsr.~~

Author contribution

Dr. C. Rautenbach conceptualised the study, executed the experiments and wrote the manuscript. He also secured the publication funding. Dr. J. C. Mullarney and Professor K. R. Bryan reviewed the manuscript.

Competing interests

No conflict of interests.

Funding

This research was funded by the National Research Foundation of South Africa (Grant Numbers: 116359).

References

Babatunde, A., Pascale, B., Sin, C. C., William, C., Peter, C., Fatima, D., Seita, E., Veronika, E., Forest, C., Peter, G., Eric, G., Christian, J., Vladimir, K., Chris, R. and Markku, R.: Evaluation of Climate Models. In: Climate Change 2013: The Physical Science Basis. Contribution of Working Group I to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change, edited by T. F. Stocker, D. Qin, G.-K. Plattner, M. Tignor, S. K. Allen, J. Boschung, A. Nauels, Y. . Xia, V. Bex, and P. M. Midgley, Cambridge University Press, Cambridge, UK., 2013.

Collins, J. I.: Prediction of shallow-water spectra, J. Geophys. Res., 77(15), 2693–2707, doi:10.1029/JC077i015p02693, 1972.

Dietrich, J. C., Tanaka, S., Westerink, J. J., Dawson, C. N., Luettich, R. A., Zijlema, M., Holthuijsen, L. H., Smith, J. M., Westerink, L. G. and Westerink, H. J.: Performance of the Unstructured-Mesh, SWAN+ADCIRC Model in Computing Hurricane Waves and Surge, J. Sci. Comput., 52(2), 468–497, doi:10.1007/s10915-011-9555-6, 2012.

Frihy, O. E.: The necessity of environmental impact assessment (EIA) in implementing coastal projects : lessons learned from the Egyptian Mediterranean Coast, , 44, 489–516, 2001.

Genseberger, M. and Donners, J.: A Hybrid SWAN Version for Fast and Efficient Practical Wave Modelling, Procedia Comput. Sci., 51(1), 1524–1533, doi:10.1016/j.procs.2015.05.342, 2015.

Genseberger, M. and Donners, J.: Hybrid SWAN for Fast and Efficient Practical Wave Modelling - Part 2, vol. 12139, edited by V. V. Krzhizhanovskaya, G. Závodszky, M. H. Lees, J. J. Dongarra, P. M. A. Sloot, S. Brissos, and J. Teixeira, pp. 87–100, Springer International Publishing, Cham., 2020.

Holthuijsen, L. H.: Waves in Oceanic and Coastal Waters, Cambridge University Press., 2007.

Jeuring, J., Knol-kauffman, M. and Sivle, A.: Toward valuable weather and sea-ice services for the marine Arctic : exploring user – producer interfaces of the Norwegian Meteorological, Polar Geogr., 0(0), 1–21, doi:10.1080/1088937X.2019.1679270, 2019.

Kamphuis, J. W.: Introduction to coastal engineering and management - Advanced series on ocean engineering - Volume 48, World scientific publishing Co. Pte. Ltd., Singapore., 2020.

Kerr, P. C., Donahue, A. S., Westerink, J. J., Luettich, R. A., Zheng, L. Y., Weisberg, R. H., Huang, Y., Wang, H. V., Teng, Y., Forrest, D. R., Roland, A., Haase, A. T., Kramer, A. W., Taylor, A. A., Rhome, J. R., Feyen, J. C., Signell, R. P., Hanson, J. L., Hope, M. E., Estes, R. M., Dominguez, R. A., Dunbar, R. P., Semeraro, L. N., Westerink, H. J., Kennedy, A. B., Smith, J. M., Powell, M. D., Cardone, V. J. and Cox, A. T.: U.S. IOOS coastal and ocean modeling testbed: Inter-model evaluation of tides, waves, and hurricane surge in the Gulf of Mexico, J. Geophys. Res. Ocean., 118(10), 5129–5172, doi:10.1002/jgrc.20376, 2013.

Liu, T. K., Sheu, H. Y. and Tseng, C. N.: Environmental impact assessment of seawater desalination plant under the framework of integrated coastal management, Desalination, 326, 10–18, doi:10.1016/j.desal.2013.07.003, 2013.

O'Neill, C., Saulter, A., Williams, J. and Horsburgh, K.: NEMO-surge: Application of atmospheric forcing and surge evaluation. Technical report 619, , (December), 57 [online] Available from: http://www.metoffice.gov.uk/binaries/content/assets/mohippo/pdf/library/frtr_619_2016p.pdf, 2016.

Rautenbach, C., Daniels, T., de Vos, M. and Barnes, M. A.: A coupled wave, tide and storm surge operational forecasting system for South Africa: validation and physical description, Nat. Hazards, doi:10.1007/s11069-020-04042-4, 2020a.

Rautenbach, C., Barnes, M. A., Wang, D. W. and Dykes, J.: Southern African wave model sensitivities and accuracies, J. Mar. Sci. Eng., Under revi, 2020b.

de Souza, J. M. A. C., Couto, P., Soutelino, R. and Roughan, M.: Evaluation of four global ocean reanalysis products for New Zealand waters–A guide for regional ocean modelling, New Zeal. J. Mar. Freshw. Res., 0(0), 1–24, doi:10.1080/00288330.2020.1713179, 2020.

Tanaka, S., Bunya, S., Westerink, J. J., Dawson, C. and Luettich, R. A.: Scalability of an unstructured grid continuous Galerkin based hurricane storm surge model, J. Sci. Comput., 46(3), 329–358, doi:10.1007/s10915-010-9402-1, 2011.

The SWAN team: IMPLEMENTATION MANUAL SWAN Cycle III version 41.31, [online] Available from: http://swanmodel.sourceforge.net/online_doc/swanimp/swanimp.html, 2019a.

The SWAN team: USER MANUAL SWAN Cycle III version 41.3, , 129 [online] Available from: http://www.fluidmechanics.tudelft.nl/swan/index.htmhttp://www.fluidmechanics.tud, 2019b.

The SWAN Team: USER MANUAL SWAN Cycle III version 40.51, Cycle, 2006.

van der Westhuysen, A. J., Zijlema, M. and Battjes, J. A.: Nonlinear saturation-based whitecapping dissipation in SWAN for deep and shallow water, Coast. Eng., 54(2), 151–170, doi:10.1016/j.coastaleng.2006.08.006, 2007.

Zafari, A., Larsson, E. and Tillenius, M.: DuctTeip: An efficient programming model for distributed task-based parallel computing, Parallel Comput., 90, 102582, doi:10.1016/j.parco.2019.102582, 2019.

Zhang, S., Xia, Z., Yuan, R. and Jiang, X.: Parallel computation of a dam-break flow model using OpenMP on a multi-core computer, J. Hydrol., 512, 126–133, doi:10.1016/j.jhydrol.2014.02.035, 2014.

Zhang, Y. J., Ye, F., Stanev, E. V and Grashorn, S.: Seamless cross-scale modeling with SCHISM, Ocean Model., 102, 64–81, doi:10.1016/j.ocemod.2016.05.002, 2016.

Zijlema, M.: Parallelization of a nearshore wind wave model for distributed memory architectures, in Parallel Computational

Fluid Dynamics 2004, pp. 207–214, Elsevier., 2005.