U. S. Department of Commerce
National Oceanic and Atmospheric Administration
National Weather Service
National Centers for Environmental Prediction
5200 Auth Road
Camp Springs, MD 20746

Technical Note

User manual and system documentation of WAVEWATCH III ™ version 3.14 †

Hendrik L. Tolman ‡

Environmental Modeling Center
Marine Modeling and Analysis Branch

May 2009

---

† MMAB Contribution No. 276.

‡ e-mail: Hendrik.Tolman@NOAA.gov

This page is intentionally left blank.

Contents

1 Introduction

1.1 About this manual

This is the user manual and system documentation of version 3.14 of the full-spectral third-generation wind-wave model WAVEWATCH III ™. This model has been developed at the Marine Modeling and Analysis Branch (MMAB) of the Environmental Modeling Center (EMC) of the National Centers for Environmental Prediction (NCEP). It is based on WAVEWATCH I and WAVEWATCH II as developed at Delft University of Technology, and NASA Goddard Space Flight Center, respectively. WAVEWATCH III ™ differs from its predecessors in all major aspects; i.e., governing equations, program structure, numerical and physical approaches.

This manual describes the governing equations, numerical approaches, compilation, and running of WAVEWATCH III™. The format of a combined user manual and system documentation has been chosen, to give users the necessary background to upgrade the model according to their own specifications. This approach is becoming more important, as WAVEWATCH III ™ is developing into a wave modeling framework rather than a wave model. By design, a user can apply his or her numerical or physical approaches, and thus develop a new wave model based on the WAVEWATCH III ™ framework. In such an approach, optimization, parallelization, nesting, input and output service programs from the framework can be easily shared between actual models. Whereas this document is intended to be complete and self-contained, this is not the case for all elements in the system documentation. For additional system details, reference is made to the source code, which is fully documented.

The governing equations and numerical approaches used in this model are described in chapters 2 and 3. Running the model is described in chapter 4. Installing WAVEWATCH III ™ is described in chapter 5. Finally, a short system documentation is given in chapter 6. A thorough knowledge of WAVEWATCH III ™ can be obtained by following chapters 2 through 5. A shortcut is to first install the model (chapter 5), and then successively modify input files in example runs (chapter 4).

The present model version (3.14) is the first official release since model version 2.22. Since the latter release massive modifications have been made to the code. Some of the major modifications are:

- Development of the multi-grid or wave model driver ww3_multi. In this driver, the full two-way interaction between all grids is considered at the time step level. The development of this approach is described in Tolman (2007, 2008). Idealized test cases mww3_test_01 through 05, and real life test cases mww3_case_01 through 03 have been added to the standard test cases to demonstrate the capabilities of this approach.
- Source term slots for linear wind input and surf-zone physics have been added. New source term packages have been included such as linear input, surf breaking, WAM4 input and dissipation and bottom scattering. The exact nonlinear interactions routines (WRT) have been upgraded to version 5 of this package.
- To make the development of a multi-grid model possible, a new dynamic data structure was introduced to the model. This has little or no impact on the model user, but it has major consequences for model developers.
- A model version with a continuously moving grid has been developed (model version 3.02, Tolman and Alves, 2005). This model version is intended only for deep water without land masses or currents. As part of this implementation, the calculation of advection time steps in Eq. (3.4) has been revised. To test this new option, a hurricane test ww3_ts3 has been added to the standard test cases.
- In model version 3.04 the track output file format is converted to a sequential file, using a single file server process, and file server options are introduced for the restart files (Tolman, 2003b). In later model version, the option is added to designate some processes to perform output only.
- Model version 3.05 includes expanded GrADS scripts to plot spectra and source terms, adding black and white options, and an alternative Cartesian display format.
- Wind stresses in the model are now described as vectors, paving the way for flux parameterizations that allow for stresses that do not line up with the wind direction. Additional output fields are added. Partitioning of spectral data can now be done for the entire wave

model grid (new output type), and provides partitioned wind sea and swell output fields (publication in preparation).
- With the introduction of the multi-grid model driver, active boundary points can now be placed at the outer points of the grid. Also, grid points can now be defined as inactive rather than land, and an expanded definition of the grid status map allows for drying out and wetting of model grid points.
- In the multi-grid model driver, changes of spectral resolution between grids are now allowed.
- Time steps can now be fractions of seconds, although general time management still propagates the solution by integer seconds.

A note of caution. WAVEWATCH III ™ includes wave-current interactions. The implementation of these interactions has only been tested in idealized test cases. Only limited tests in realistic conditions have been performed.

Up to date information on this model can be found (including bugs and bug fixes) on the WAVEWATCH III ™ web page, and comments, questions and suggestions should be directed to the corresponding E-mail address

http://polar.ncep.noaa.gov/waves/wavewatch
NCEP.EMC.wavewatch@NOAA.gov

We will redirect questions regarding contributions from outside NCEP to the respective authors of the codes.

1.2 Licensing terms

Starting with model version 3.14, WAVEWATCH III ™ is distributed under the following licensing terms:

start of licensing terms

Software, as understood herein, shall be broadly interpreted as being inclusive of algorithms, source code, object code, data bases and related documentation, all of which shall be furnished free of charge to the Licensee.

Corrections, upgrades or enhancements may be furnished and, if furnished, shall also be furnished to the Licensee without charge. NOAA, however, is not required to develop or furnish such corrections, upgrades or enhancements.

NOAA's software, whether that initially furnished or corrections or upgrades, are furnished as is. NOAA furnishes its software without any warranty whatsoever and is not responsible for any direct, indirect or consequential damages that may be incurred by the Licensee. Warranties of merchantability, fitness for any particular purpose, title, and non-infringement, are specifically negated.

The Licensee is not required to develop any software related to the licensed software. However, in the event that the Licensee does so, the Licensee is required to offer same to NOAA for inclusion under the instant licensing terms with NOAA's licensed software along with documentation regarding its principles, use and its advantages. This includes changes to the wave model proper including numerical and physical approaches to wave modeling, and boundary layer parameterizations embedded in the wave model The Licensee is encouraged but not obligated to provide pre-and post processing tools for model input and output. The software required to be offered shall not include additional models to which the wave model may be coupled, such as oceanic or atmospheric circulation models. The software provided by the Licensee shall be consistent with the latest model version available to the Licensee, and interface routines to the software provided shall conform to programming standards as outlined in the model documentation. The software offered to NOAA shall be offered as is, without any warranties whatsoever and without any liability for damages whatsoever. NOAA shall not be required to include a Licensee's software as part of its software. Licensee's offered software shall not include software developed by others.

A Licensee may reproduce sufficient software to satisfy its needs. All copies shall bear the name of the software with any version number as well as replicas of any applied copyright notice, trademark notice, other notices and credit lines. Additionally, if the copies have been modified, e.g. with deletions or additions, this shall be so stated and identified.

All of Licensee's employees who have a need to use the software may have access to the software but only after reading the instant license and stating, in writing, that they have read and understood the license and have agreed to its terms. Licensee is responsible for employing reasonable efforts to assure that only those of its employees that should have access to the software, in

fact, have access.

The Licensee may use the software for any purpose relating to sea state prediction.

No disclosure of any portion of the software, whether by means of a media or verbally, may be made to any third party by the Licensee or the Licensee's employees

The Licensee is responsible for compliance with any applicable export or import control laws of the United States.

end of licensing terms

The software will be distributed through our web site after the Licensee has agreed to the license terms.

1.3 Copyrights and trademarks

1.4 Contributors and acknowledgments

Even in its original development, when I was working on the code mostly on my own on the development of WAVEWATCH III ™, many have contributed to the success of the model. With the expansion of physical and numerical parameterizations available, the list of contributors to this model is growing. I would like to recognize the following contributors (in alphabetic order) :

Henrique Alves (Metocean Engineers, Australia)
: Support of code development while at NCEP, shallow water physics packages.

Fabrice Ardhuin (Service Hydrographique et Océanographique de la Marine, France)
Various physics packages.

Nico Booij (Delft University of Technology, The Netherlands)
Original design of source code pre-processor (`w3adc`), basic method of documentation and other programming habits. Spatially varying wavenumber grid.

Dmitry V. Chalikov (ESSIC, Univ. of Maryland, USA)
Co-author of the Tolman and Chalikov (1996) input and dissipation parameterizations and source code.

Arun Chawla (Science Applications International Corporation, USA)
Support of code development at NCEP, GRIB packing, automated grid generation software (Chawla and Tolman, 2007, 2008).

Barbara Tracy (US Army Corps of Engineers, ERDC-CHL, USA)
Spectral partitioning

Gerbrant Ph. van Vledder (Alkyon Hydraulic Consultancy & Research, NL)
Webb-Resio-Tracy exact nonlinear interaction routines, as well as some of the original service routines.

The development of WAVEWATCH III ™ has been an ongoing process for well over a decade. The development of WAVEWATCH I was entirely funded through my Ph.D. work at Delft University. The development of WAVEWATCH II has been funded entirely through my position as a National Research Council Resident Research Associate at NASA, Goddard Space Flight Center. The initial development of WAVEWATCH III version 1.18 was entirely funded by NOAA/NCEP, with most funding provided by the NOAA High Performance Computing and Communication (HPCC) office. Developments of version 2.22 have been funded similarly through NOAA/NCEP. Since then, funding is provided by NCEP, but also by many partners outside NCEP. Much of the work at NCEP has been performed under contract by SAIC.

I would finally like to thank all users who have reported errors and glitches, or have made suggestions for improvements, particularly those who have beta-tested this and previous model release.

2 Governing equations

2.1 Introduction

Waves or spectral wave components in water with limited depth and non-zero mean currents are generally described using several phase and amplitude parameters. Phase parameters are the wavenumber vector $\mathbf{k}$, the wavenumber k, the direction θ and several frequencies. If effects of mean currents on waves are to be considered, a distinction is made between the relative or intrinsic (radian) frequency σ $(= 2\pi f_r)$, which is observed in a frame of reference moving with the mean current, and the absolute (radian) frequency ω $(= 2\pi f_a)$, which is observed in a fixed frame of reference. The direction θ is by definition perpendicular to the crest of the wave (or spectral component), and equals the direction of $\mathbf{k}$. Generally, scales of variation of depths and currents are assumed to be much larger than those of an individual wave. The quasi-uniform (linear) wave theory then can be applied locally, giving the following dispersion relation and Doppler type equation to interrelate the phase parameters

$$\sigma^2 = gk \tanh kd \,, \tag{2.1}$$

$$\omega = \sigma + \mathbf{k} \cdot \mathbf{U} \,, \tag{2.2}$$

where d is the mean water depth and $\mathbf{U}$ is the (depth- and time- averaged) current velocity. The assumption of slowly varying depths and currents implies a large-scale bathymetry, for which wave diffraction can generally be ignored. The usual definition of $\mathbf{k}$ and ω from the phase function of a wave or wave component implies that the number of wave crests is conserved (see, e.g., Phillips, 1977; Mei, 1983)

$$\frac{\partial \mathbf{k}}{\partial t} + \nabla\omega = 0 \,. \tag{2.3}$$

¿From Eqs. (2.1) through (2.3) the rates of change of the phase parameters can be calculated (e.g., Christoffersen, 1982; Mei, 1983; Tolman, 1990, equations not reproduced here).

For monochromatic waves, the amplitude is described as the amplitude, the wave height, or the wave energy. For irregular wind waves, the (random) variance of the sea surface is described using variance density spectra

(in the wave modeling community usually denoted as energy spectra). The variance spectrum F is a function of all independent phase parameters, i.e., $F(\mathbf{k},\sigma,\omega)$, and furthermore varies in space and time, e.g., $F(\mathbf{k},\sigma,\omega;\mathbf{x},t)$. However, it is usually assumed that the individual spectral components satisfy the linear wave theory (locally), so that Eqs. (2.1) and (2.2) interrelate $\mathbf{k}$, σ and ω. Consequently only two independent phase parameters exist, and the local and instantaneous spectrum becomes two-dimensional. Within WAVEWATCH III ^{TM} the basic spectrum is the wavenumber-direction spectrum $F(k,\theta)$, which has been selected because of its invariance characteristics with respect to physics of wave growth and decay for variable water depths. The output of WAVEWATCH III ^{TM}, however, consists of the more traditional frequency-direction spectrum $F(f_r,\theta)$. The different spectra can be calculated from $F(k,\theta)$ using straightforward Jacobian transformations

$$F(f_r,\theta) = \frac{\partial k}{\partial f_r} F(k,\theta) = \frac{2\pi}{c_g} F(k,\theta) \,, \tag{2.4}$$

$$F(f_a,\theta) = \frac{\partial k}{\partial f_a} F(k,\theta) = \frac{2\pi}{c_g} \left(1 + \frac{\mathbf{k}\cdot\mathbf{U}}{k c_g} \right)^{-1} F(k,\theta) \,, \tag{2.5}$$

$$c_g = \frac{\partial \sigma}{\partial k} = n \frac{\sigma}{k} \;,\; n = \frac{1}{2} + \frac{kd}{\sinh 2kd} \;, \tag{2.6}$$

where c_g is the so-called group velocity. From any of these spectra one-dimensional spectra can be generated by integration over directions, whereas integration over the entire spectrum by definition gives the total variance E (in the wave modeling community usually denoted as the wave energy).

In cases without currents, the variance (energy) of a wave package is a conserved quantity. In cases with currents the energy or variance of a spectral component is no longer conserved, due to the work done by current on the mean momentum transfer of waves (Longuet-Higgins and Stewart, 1961, 1962). In a general sense, however, wave action $A \equiv E/\sigma$ is conserved (e.g., Whitham, 1965; Bretherthon and Garrett, 1968). This makes the wave action density spectrum $N(k,\theta) \equiv F(k,\theta)/\sigma$ the spectrum of choice within the model. Wave propagation then is described by

$$\frac{DN}{Dt} = \frac{S}{\sigma} \,, \tag{2.7}$$

where D/Dt represents the total derivative (moving with a wave component) and S represents the net effect of sources and sinks for the spectrum

F. Because the left side of Eq. (2.7) generally considers linear propagation, effects of nonlinear wave propagation (i.e., wave-wave interactions) arise in S. Propagation and source terms will be discussed separately in the following sections.

2.2 Propagation

In a numerical model, a Eulerian form of the balance equation (2.7) is needed. This balance equation can either be written in the form of a transport equation (with velocities outside the derivatives), or in a conservation form (with velocities inside the derivatives). The former form is valid for the vector wavenumber spectrum $N(\mathbf{k};\mathbf{x},t)$ only, whereas valid equations of the latter form can be derived for arbitrary spectral formulations, as long as the corresponding Jacobian transformation as described above is well behaved (e.g., Tolman and Booij, 1998). Furthermore, the conservation equation conserves total wave energy/action, unlike the transport equation. This is an important feature of an equation when applied in a numerical model. The balance equation for the spectrum $N(k,\theta;\mathbf{x},t)$ as used in WAVEWATCH III ^{TM} is given as (for convenience of notation, the spectrum is henceforth denoted simply as N)

$$\frac{\partial N}{\partial t} + \nabla_x \cdot \dot{\mathbf{x}} N + \frac{\partial}{\partial k}\dot{k} N + \frac{\partial}{\partial \theta}\dot{\theta} N = \frac{S}{\sigma} \,, \tag{2.8}$$

$$\dot{\mathbf{x}} = \mathbf{c}_g + \mathbf{U} \,, \tag{2.9}$$

$$\dot{k} = -\frac{\partial \sigma}{\partial d}\frac{\partial d}{\partial s} - \mathbf{k} \cdot \frac{\partial \mathbf{U}}{\partial s} \,, \tag{2.10}$$

$$\dot{\theta} = -\frac{1}{k}\left[\frac{\partial \sigma}{\partial d}\frac{\partial d}{\partial m} - \mathbf{k} \cdot \frac{\partial \mathbf{U}}{\partial m}\right] \,, \tag{2.11}$$

where $\mathbf{c}_g$ is given by c_g and θ, s is a coordinate in the direction θ and m is a coordinate perpendicular to s. Equation (2.8) is valid for a Cartesian grid. For large-scale applications, this equation is usually transferred to a spherical grid, defined by longitude λ and latitude ϕ, but maintaining the definition of the local variance (i.e., per unit surface, as in WAMDIG, 1988)

$$\frac{\partial N}{\partial t} + \frac{1}{\cos\phi}\frac{\partial}{\partial\phi}\dot{\phi}N\cos\theta + \frac{\partial}{\partial\lambda}\dot{\lambda}N + \frac{\partial}{\partial k}\dot{k}N + \frac{\partial}{\partial\theta}\dot{\theta}_g N = \frac{S}{\sigma}\,, \tag{2.12}$$

$$\dot{\phi} = \frac{c_g\cos\theta + U_\phi}{R}\,, \tag{2.13}$$

$$\dot{\lambda} = \frac{c_g\sin\theta + U_\lambda}{R\cos\phi}\,, \tag{2.14}$$

$$\dot{\theta}_g = \dot{\theta} - \frac{c_g\tan\phi\cos\theta}{R}\,, \tag{2.15}$$

where R is the radius of the earth and U_ϕ and U_λ are current components. Equation (2.15) includes a correction term for propagation along great circles, using a Cartesian definition of θ where $\theta = 0$ corresponds to waves traveling from west to east. WAVEWATCH III ^{TM} can be run on either a Cartesian of spherical grid. Note that unresolved obstacles such as islands can be included in the equations. In WAVEWATCH III ^{TM} this is done at the level of the numerical scheme, as is discussed in section 3.3.3.

2.3 Source terms

2.3.1 General concepts

In deep water, the net source term S is generally considered to consist of three parts, a wind-wave interaction term S_{in}, a nonlinear wave-wave interactions term S_{nl} and a dissipation ('whitecapping') term S_{ds}. The input term S_{in} is dominated by the exponential growth term, and this source term generally describes this dominant process only. For model initialization, and to provide more realistic initial wave growth, and linear input term S_{ln} can also be considered in WAVEWATCH III ^{TM}.

In shallow water additional processes have to be considered, most notably wave-bottom interactions S_{bot} (e.g., Shemdin et al., 1978). In extremely shallow water, depth-induced breaking (S_{db}) and triad wave-wave interactions (S_{tr}) become important. Also available in WAVEWATCH III ^{TM} are source terms for scattering of waves by bottom features (S_{sc}) and a general purpose slot for additional, user defined source terms (S_{xx}

This defines the general source terms used in WAVEWATCH III ^{TM} as

$$S = S_{ln} + S_{in} + S_{nl} + S_{ds} + S_{bot} + S_{db} + S_{tr} + S_{sc} + S_{xx} \; . \tag{2.16}$$

Other source terms are easily added. These source terms are defined for the *energy* spectra. In the model, however, most source terms are directly calculated for the action spectrum. The latter source terms are denoted as $\mathcal{S} \equiv S/\sigma$.

The treatment of the nonlinear interactions defines a third-generation wave model. Therefore, the options for the calculation of S_{nl} will be discussed first in sections 2.3.2 and 2.3.3. S_{in} and S_{ds} represent separate processes, but should be considered as interrelated, because the balance of these two source terms governs the integral growth characteristics of the wave model. Two combinations of these basic source terms are available, those of WAM cycles 1 through 3 (section 2.3.4) and the parameterizations of Tolman and Chalikov (1996) (section 2.3.5) and those of WAM cycle 4[1] (section 2.3.6). Linear input, shallow water source terms or source terms describing special physical processes are considered to be "additional" source terms. Available options are described in Sections 2.3.7 through 2.3.12.

A third-generation wave model effectively integrates the spectrum only up to a cut-off frequency f_{hf} (or wavenumber k_{hf}). Above this frequency a parametric tail is applied (e.g., WAMDIG, 1988)

$$F(f_r,\theta) = F(f_{r,hf},\theta) \left(\frac{f_r}{f_{r,hf}} \right)^{-m} \tag{2.17}$$

which is easily transformed to any other spectrum using the Jacobian transformations as discussed above. For instance, for the present action spectrum, the parametric tail can be expressed as (assuming deep water for the wave components in the tail)

$$N(k,\theta) = N(k_{hf},\theta) \left(\frac{f_r}{f_{r,hf}} \right)^{-m-2} \tag{2.18}$$

The actual values of m and the expressions for $f_{r,hf}$ depend on the source term parameterization used, and will be given below.

[1] including recent updates.

Before actual source term parameterizations are described, the definition of the wind requires some attention. In cases with currents, one can either consider the wind to be defined in a fixed frame of reference, or in a frame of reference moving with the current. Both definitions are available in WAVEWATCH III TM, and can be selected during compilation. The output of the program, however, will always be the wind speed which is not in any way corrected for the current.

2.3.2 Nonlinear interactions (DIA)

Nonlinear wave-wave interactions can be modeled using the discrete interaction approximation (DIA, Hasselmann et al., 1985). This parameterization was originally developed for the spectrum $F(f_r,\theta)$. To assure the conservative nature of S_{nl} for this spectrum (which can be considered as the "final product" of the model), this source term is calculated for $F(f_r,\theta)$ instead of $N(k,\theta)$, using the conversion (2.4).

Resonant nonlinear interactions occur between four wave components (quadruplets) with wavenumber vector $\mathbf{k}_1$ through $\mathbf{k}_4$. In the DIA, it is assumed that $\mathbf{k}_1 = \mathbf{k}_2$. Resonance conditions then require that

$$\left.\begin{array}{rcl} \mathbf{k}_1 + \mathbf{k}_2 & = & \mathbf{k}_3 + \mathbf{k}_4 \\ \sigma_2 & = & \sigma_1 \\ \sigma_3 & = & (1+\lambda_{nl})\sigma_1 \\ \sigma_4 & = & (1-\lambda_{nl})\sigma_1 \end{array}\right\} \quad , \tag{2.19}$$

where λ_{nl} is a constant. For these quadruplets, the contribution δS_{nl} to the interaction for each discrete (f_r,θ) combination of the spectrum corresponding to $\mathbf{k}_1$ is calculated as

$$\begin{pmatrix} \delta S_{nl,1} \\ \delta S_{nl,3} \\ \delta S_{nl,4} \end{pmatrix} = D \begin{pmatrix} -2 \\ 1 \\ 1 \end{pmatrix} C g^{-4} f_{r,1}^{11} \times \left[F_1^2 \left(\frac{F_3}{(1+\lambda_{nl})^4} + \frac{F_4}{(1-\lambda_{nl})^4} \right) - \frac{2F_1F_3F_4}{(1-\lambda_{nl}^2)^4} \right] , \tag{2.20}$$

where $F_1 = F(f_{r,1},\theta_1)$ etc. and $\delta S_{nl,1} = \delta S_{nl}(f_{r,1},\theta_1)$ etc., C is a proportionality constant. The nonlinear interactions are calculated by considering a

	λ_{nl}	C
WAM-3	0.25	$2.78\ 10^7$
Tolman and Chalikov	0.25	$1.00\ 10^7$

Table 2.1: Default constants in DIA for input-dissipation packages.

limited number of combinations (λ_{nl}, C). In practice, only one combination is used. Default values for different source term packages are presented in Table 2.1.

This source term is developed for deep water, using the appropriate dispersion relation in the resonance conditions. For shallow water the expression is scaled by the factor D (still using the deep-water dispersion relation, however)

$$D = 1 + \frac{c_1}{\bar{k}d}\left[1 - c_2\bar{k}d\right] e^{-c_3\bar{k}d} \,. \tag{2.21}$$

Recommended (default) values for the constants are (Hasselmann and Hasselmann, 1985) $c_1 = 5.5$, $c_2 = 5/6$ and $c_3 = 1.25$. The overbar notation denotes straightforward averaging over the spectrum. For an arbitrary parameter z the spectral average is given as

$$\bar{z} = E^{-1} \int_0^{2\pi} \int_0^{\infty} zF(f_r, \theta)\, df_r\, d\theta \,, \tag{2.22}$$

$$E = \int_0^{2\pi} \int_0^{\infty} F(f_r, \theta)\, df_r\, d\theta \,, \tag{2.23}$$

For numerical reasons, however, the mean relative depth is estimated as

$$\bar{k}d = 0.75\hat{k}d \,, \tag{2.24}$$

where $\hat{k}$ is defined as

$$\hat{k} = \left(\overline{1/\sqrt{k}}\right)^{-2} \,. \tag{2.25}$$

The shallow water correction of Eq. (2.21) is valid for intermediate depths only. For this reason the mean relative depth $\bar{k}d$ is not allowed to become

smaller than 0.5 (as in WAM). All above constants can be reset by the user in the input files of the model (see chapter 4).

2.3.3 Nonlinear interactions (WRT) (G. Ph. van Vledder)

The second method for calculating the nonlinear interactions in WAVEWATCH III ™ is the so-called Webb-Resio-Tracy method (WRT), which is based on the original six-dimensional Boltzmann integral formulation of Hasselmann (1962, 1963a,b), and additional considerations by Webb (1978), Tracy and Resio (1982) and Resio and Perrie (1991).

The Boltzmann integral describes the rate of change of action density of a particular wavenumber due to resonant interactions between pairs of four wavenumbers. To interact these wavenumbers must satisfy the following resonance conditions

$$\left.\begin{array}{rcl} \mathbf{k}_1 + \mathbf{k}_2 & = & \mathbf{k}_3 + \mathbf{k}_4 \\ \sigma_1 + \sigma_2 & = & \sigma_3 + \sigma_4 \end{array}\right\} \quad , \tag{2.26}$$

which is a more general version of the resonance conditions (2.19). The rate of change of action density N_1 at wave number $\mathbf{k}_1$ due to all quadruplet interactions involving $\mathbf{k}_1$ is given by

$$\begin{aligned} \frac{\partial N_1}{\partial t} \;=\; & \iiint G\left(\mathbf{k}_1,\mathbf{k}_2,\mathbf{k}_3,\mathbf{k}_4\right)\, \delta\left(\mathbf{k}_1+\mathbf{k}_2-\mathbf{k}_3-\mathbf{k}_4\right)\, \delta\left(\sigma_1+\sigma_2-\sigma_3-\sigma_4\right) \\ & \times \quad \left[N_1 N_3 \left(N_4 - N_2\right) + N_2 N_4 \left(N_3 - N_1\right)\right]\, d\mathbf{k}_2\, d\mathbf{k}_3\, d\mathbf{k}_4 \quad , \end{aligned} \tag{2.27}$$

where the action density N is defined in terms of the wavenumber vector $\mathbf{k}$, $N = N(\mathbf{k})$. The term G is a complicated coupling coefficients for which expressions have been given by Herterich and Hasselmann (1980). In the WRT method a number of transformations are made to remove the delta functions. A key element in the WRT method is to consider the integration space for each $(\mathbf{k}_1, \mathbf{k}_3)$ combination (see Resio and Perrie, 1991)

$$\frac{\partial N_1}{\partial t} = 2 \int T\left(\mathbf{k}_1,\mathbf{k}_3\right)\, d\mathbf{k}_3 \quad , \tag{2.28}$$

in which the function T is given by

$$
\begin{aligned}
T\left(\mathbf{k}_1,\mathbf{k}_3\right) &= \iint G\left(\mathbf{k}_1,\mathbf{k}_2,\mathbf{k}_3,\mathbf{k}_4\right)\ \delta\left(\mathbf{k}_1+\mathbf{k}_2-\mathbf{k}_3-\mathbf{k}_4\right) \\
&\times\ \delta\left(\sigma_1+\sigma_2-\sigma_3-\sigma_4\right)\ \theta\left(\mathbf{k}_1,\mathbf{k}_3,\mathbf{k}_4\right) \\
&\times\ \left[N_1N_3\left(N_4-N_2\right)+N_2N_4\left(N_3-N_1\right)\right]\ d\mathbf{k}_2\,d\mathbf{k}_4\quad , \qquad (2.29)
\end{aligned}
$$

in which

$$
\theta\left(\mathbf{k}_1,\mathbf{k}_3,\mathbf{k}_4\right)=\left\{\begin{array}{lll} 1 & \text{when} & \left|\mathbf{k}_1-\mathbf{k}_3\right|\le\left|\mathbf{k}_1-\mathbf{k}_4\right| \\ 0 & \text{when} & \left|\mathbf{k}_1-\mathbf{k}_3\right|>\left|\mathbf{k}_1-\mathbf{k}_4\right| \end{array}\right. \qquad (2.30)
$$

The delta functions in Eq. (2.29) determine a region in wavenumber space along which the integration should be carried out. The function θ determines a section of the integral which is not defined due to the assumption that $\mathbf{k}_1$ is closer to $\mathbf{k}_3$ than $\mathbf{k}_2$. The crux of the Webb method consists of using a local coordinate system along a so-named locus, that is, the path in $\mathbf{k}$ space given by the resonance conditions for a given combination of $\mathbf{k}_1$ and $\mathbf{k}_3$. To that end the (k_x,k_y) coordinate system is replaced by a (s,n) coordinate system, where s (n) is the tangential (normal) direction along the locus. After some transformations the transfer integral can then be written as a closed line integral along the closed locus

$$
\begin{aligned}
T\left(\mathbf{k}_1,\mathbf{k}_3\right) &= \oint G\left|\frac{\partial W(s,n)}{\partial n}\right|^{-1}\ \theta(\mathbf{k}_1,\mathbf{k}_3,\mathbf{k}_4) \\
&\times\ \left[N_1N_3\left(N_4-N_2\right)+N_2N_4\left(N_3-N_1\right)\right]\ ds\quad , \qquad (2.31)
\end{aligned}
$$

In which G is the coupling coefficient and $|\partial W/\partial n|$ is the gradient term of a function representing the resonance conditions (see Van Vledder, 2000). Numerically, the Boltzmann integral is computed as the finite sum of many line integrals T for all discrete combinations of $\mathbf{k}_1$ and $\mathbf{k}_3$. The line integral (2.31) is solved by dividing the locus in typically 30 pieces, such that the discretized version is given as:

$$
T\left(\mathbf{k}_1,\mathbf{k}_3\right)\approx\sum_{i=1}^{n_s}G(s_i)W(s_i)P(s_i)\,\Delta s_i\quad , \qquad (2.32)
$$

in which $P(s_i)$ is the product term for a given point on the locus, n_s is the number of segments, and s_i is the discrete coordinate along the locus. Finally, the rate of change for a given wavenumber $\mathbf{k}_1$ is given by

$$\frac{\partial N(\mathbf{k}_1)}{\partial t} \approx \sum_{i_{k3}=1}^{n_k} \sum_{i_{\theta 3}=1}^{n_\theta} k_3 T(\mathbf{k}_1, \mathbf{k}_3)\, \Delta k_{i_{k3}}\, \Delta\theta_{i_{\theta 3}} \tag{2.33}$$

where n_k and n_θ are the discrete number of wavenumbers and directions in the calculational grid, respectively. Note that although the spectrum is defined in terms of the vector wavenumber $\mathbf{k}$, the calculational grid in a wave model is more conveniently defined in terms of the absolute wavenumber and wave direction (k, θ) to assure directional isotropy of the calculations. Taking all wave numbers $\mathbf{k}_1$ into account produces the complete source term due to nonlinear quadruplet wave-wave interactions. Details of the efficient computation of a locus for a given combination of the wave numbers $\mathbf{k}_1$ and $\mathbf{k}_3$ can be found in Van Vledder (2000, 2002a,b).

It should be noted that these exact interaction calculations are extremely expensive, typically requiring 10^3 to 10^4 times more computational effort than the DIA. Presently, these calculations can therefore only be made for highly idealized test cases involving a limited spatial grid.

The nonlinear interactions according to the WRT method have been implemented in WAVEWATCH III^{TM} using the portable subroutines developed by Van Vledder (2002b). In this implementation, the computational grid of the WRT method is taken identical to the discrete spectral grid of WAVEWATCH III ^{TM}. In addition, the WRT routines inherit the power of the parametric spectral tail as in the DIA. Choosing a higher resolution than the computational grid of WAVEWATCH III^{TM} for computing the nonlinear interactions is possible in theory, but this does not improve the results and is therefore not implemented.

Because nonlinear quadruplet wave-wave interactions at high frequencies are important, it is recommended to choose the maximum frequency of the wave model about five times the peak frequency of the spectra that are expected to occur in a wave model run. Note that this is important as the spectral grid determines the range of integration in Eq. (2.33). The recommended number of frequencies is about 40, with a frequency increment factor 1.07. The recommended directional resolution for computing the nonlinear interactions is about 10°. For specific purposes other resolutions may be used, and some testing with other resolutions may be needed.

An important feature of most algorithms for the evaluation of the Boltzmann integral is that the integration space can be pre-computed. This is also the case for the subroutine version of the WRT method used in WAVE-

WATCH III ^{TM}. In the initialization phase of the wave model the integration space, consisting of the discretized paths of all loci, together with the interaction coefficients and gradient terms, are computed and stored in a binary data file. For each water depth such a data file is generated and stored in the current directory. The names of these data files consist of a keyword, "quad", followed by the keyword "*xxxx*", with *xxxx* the water depth in meters, or 9999 for deep water. The extension of the binary data file is "bqf" (of Binary Quadruplet File). If a BQF file exists, the program checks if this BQF file has been generated with the proper spectral grid. If this is not the case, the existing BQF file is overwritten with the correct BQF file. During a wave model run with various depths, the optimal BQF is used, by looking at the nearest water depths for which a valid BQF file has been generated. In addition, the result is rescaled using the ratio of the depth scaling factors (2.21) for the target depth and the depth corresponding to the BQF file.

2.3.4 Input and dissipation (WAM-3)

The input and dissipation source terms of WAM cycles 1 through 3 are based on Snyder et al. (1981) and Komen et al. (1984) (see also WAMDIG, 1988). The input source term is given as

$$\mathcal{S}_{in}(k,\theta) = C_{in}\frac{\rho_a}{\rho_w}\max\left[0,\left(\frac{28\,u_*}{c}\cos(\theta-\theta_w)-1\right)\right]\,\sigma\,N(k,\theta)\;, \tag{2.34}$$

$$u_* = u_{10}\sqrt{(0.8+0.065u_{10})10^{-3}}\,, \tag{2.35}$$

where C_{in} is a constant ($C_{in}=0.25$), ρ_a (ρ_w) is the density of air (water), u_* is the wind friction velocity (Charnock, 1955; Wu, 1982), c is the phase velocity σ/k, u_{10} is the wind speed at 10 m above the mean sea level and θ_w is the mean wind direction. The corresponding dissipation term is given as

$$\mathcal{S}_{ds}(k,\theta) = C_{ds}\,\hat{\sigma}\,\frac{k}{\hat{k}}\left(\frac{\hat{\alpha}}{\hat{\alpha}_{PM}}\right)^2 N(k,\theta)\;, \tag{2.36}$$

$$\hat{\sigma} = \left(\overline{\sigma^{-1}}\right)^{-1}\;, \tag{2.37}$$

$$\hat{\alpha} = E\,\hat{k}^2\,g^{-2}\,, \tag{2.38}$$

where C_{ds} is a constant ($C_{ds} = -2.36\,10^{-5}$), $\hat{\alpha}_{PM}$ is the value of $\hat{\alpha}$ for a PM spectrum ($\hat{\alpha}_{PM} = 3.02\,10^{-3}$) and where $\hat{k}$ is given by Eq. (2.25).

The parametric tail [Eqs. (2.17) and (2.18)] corresponding to these source terms is given by[2] $m = 4.5$ and by

$$f_{hf} = \max\left[\, 2.5\,\hat{f}_r \,,\, 4\,f_{r,PM} \,\right] \,, \tag{2.39}$$

$$f_{r,PM} = \frac{g}{28\,u_*} \,, \tag{2.40}$$

where $f_{r,PM}$ is the Pierson and Moskowitz (1964) frequency, estimated from the wind friction velocity u_*. The shape and attachment point of this tail is hardwired to the present model. The tunable parameters C_{in}, C_{ds} and α_{PM} are preset to their default values, but can be redefined by the user in the input files of the model.

2.3.5 Input and dissipation (Tolman and Chalikov)

The source term package of Tolman and Chalikov (1996) consists of the input source term of Chalikov and Belevich (1993) and Chalikov (1995), and two dissipation constituents. The input source term is given as

$$\mathcal{S}_{in}(k,\theta) = \sigma\,\beta\,N(k,\theta) \,, \tag{2.41}$$

where β is a nondimensional wind-wave interaction parameter, which is approximated as

$$10^4\beta = \begin{cases} -a_1\tilde{\sigma}_a^2 - a_2 & , & & \tilde{\sigma}_a & \le -1 \\ a_3\tilde{\sigma}_a(a_4\tilde{\sigma}_a - a_5) - a_6 & , & -1 < & \tilde{\sigma}_a & < \Omega_1/2 \\ (a_4\tilde{\sigma}_a - a_5)\tilde{\sigma}_a & , & \Omega_1/2 < & \tilde{\sigma}_a & < \Omega_1 \\ a_7\tilde{\sigma}_a - a_8 & , & \Omega_1 < & \tilde{\sigma}_a & < \Omega_2 \\ a_9(\tilde{\sigma}_a - 1)^2 + a_{10} & , & \Omega_2 < & \tilde{\sigma}_a & \end{cases} \tag{2.42}$$

where

$$\tilde{\sigma}_a = \frac{\sigma\,u_\lambda}{g}\cos(\theta - \theta_w) \tag{2.43}$$

[2] originally, WAM used $m = 5$, present setting used for consistent limit behavior (e.g., Tolman, 1992).

is the non-dimensional frequency of a spectral component, θ_w is the wind direction and u_λ is the wind velocity at a height equal to the 'apparent' wave length

$$\lambda_a = \frac{2\pi}{k|\cos(\theta-\theta_w)|} \; . \tag{2.44}$$

The parameters $a_1 - a_{10}$ and Ω_1, Ω_2 in Eq. (2.42) depend on the drag coefficient C_λ at the height $z = \lambda_a$:

$$\begin{array}{llll}
\Omega_1 = 1.075 + 75C_\lambda & \Omega_2 = & 1.2 + 300C_\lambda \\
a_1 = 0.25 + 395C_\lambda, & a_3 = & (a_0 - a_2 - a_1)/(a_0 - a_4 + a_5) \\
a_2 = 0.35 + 150C_\lambda, & a_5 = & a_4\Omega_1 \\
a_4 = 0.30 + 300C_\lambda, & a_6 = & a_0(1 - a_3) \\
a_9 = 0.35 + 240C_\lambda, & a_7 = & (a_9(\Omega_2 - 1)^2 + a_{10})/(\Omega_2 - \Omega_1) \\
a_{10} = -0.05 + 470C_\lambda, & a_8 = & a_7\Omega_1 \\
 & a_0 = & 0.25a_5^2/a_4
\end{array} \tag{2.45}$$

The wave model takes the wind u_r at a given reference height z_r as its input, so that u_λ and C_λ need to be derived as part of the parameterization. Excluding a thin surface layer adjusting to the water surface, the mean wind profile is close to logarithmic

$$u_z = \frac{v_*}{\kappa} \ln\left(\frac{z}{z_0}\right) \; , \tag{2.46}$$

where $\kappa = 0.4$ is the Von Kàrmàn constant, and z_0 is the roughness parameter. This equation can be rewritten in terms of the drag coefficient C_r at the reference height z_r as (Chalikov, 1995)

$$C_r = \kappa^2 \left[R - \ln(C)\right]^2 \; , \tag{2.47}$$

where

$$R = \ln\left(\frac{z_r g}{\chi\sqrt{\alpha}u_r^2}\right) \; , \tag{2.48}$$

where $\chi = 0.2$ is a constant, and where α is the conventional nondimensional energy level at high frequencies. An accurate explicit approximation to these implicit relations is given as

$$C_r = 10^{-3}\left(0.021 + \frac{10.4}{R^{1.23} + 1.85}\right) \; . \tag{2.49}$$

The estimation of the drag coefficient thus requires an estimate of the high-frequency energy level α, which could be estimated directly from the wave model. However, the corresponding part of the spectrum is generally not well resolved, tends to be noisy, and is tainted by errors in several source terms. Therefore, α is estimated parametrically as (Janssen, 1989)

$$\alpha = 0.57 \left(\frac{u_*}{c_p} \right)^{3/2} \quad . \tag{2.50}$$

As the latter equation depends on the drag coefficient, Eqs. (2.48) through (2.50) formally need to be solved iteratively. Such iterations are performed during the model initialization, but are not necessary during the actual model run, as u_* generally changes slowly. Note that Eq. (2.50) can be considered as an internal relation to the parameterization of C_r, and can therefore deviate from actual model behavior without loss of generality. In Tolman and Chalikov (1996), C_r is therefore expressed directly in terms of c_p.

Using the definition of the drag coefficient and Eq. (2.46) the roughness parameter z_0 becomes

$$z_0 = z_r \ \exp\left(-\kappa C_r^{-1/2} \right) \ , \tag{2.51}$$

and the wind velocity and drag coefficient at height λ become

$$u_\lambda = u_r \frac{ln(\lambda_a / z_0)}{ln(z_r / z_0)} \tag{2.52}$$

$$C_\lambda = C_r \left(\frac{u_a}{u_\lambda} \right)^2 \tag{2.53}$$

Finally, Eq. (2.50) requires an estimate for the peak frequency f_p. To obtain a consistent estimate of the peak frequency of actively generated waves, even in complex multimodal spectra, this frequency is estimated from the equivalent peak frequency of the positive part of the input source term (see Tolman and Chalikov, 1996)

$$f_{p,i} = \frac{\int\int \ f^{-2} \, c_g^{-1} \ \max\left[\, 0 \, , \, \mathcal{S}_{wind}(k,\theta) \, \right] df \ d\theta}{\int\int \ f^{-3} \, c_g^{-1} \ \max\left[\, 0 \, , \, \mathcal{S}_{wind}(k,\theta) \, \right] df \ d\theta} \, , \tag{2.54}$$

from which the actual peak frequency is estimated as (the tilde identifies nondimensional parameter based on u_* and g)

$$\tilde{f}_p \;=\; 3.6\,10^{-4} \;+\; 0.92\tilde{f}_{p,i} \;-\; 6.3\,10^{-10}\;\tilde{f}_{p,i}^{-3} \;\;. \tag{2.55}$$

All constants in the above equations are defined within the model. The user only defines the reference wind height z_r.

During testing of a global implementation of WAVEWATCH III$^{\rm TM}$ including this source term (Tolman, 2002f), it was found that its swell dissipation due to opposing or weak winds was severely overestimated. To correct this deficiency, a filtered input source term is defined as

$$\mathcal{S}_{i,m} = \left\{ \begin{array}{cccccc} \mathcal{S}_i & \text{for} & \beta \geq 0 & \text{or} & f > 0.8 f_p & \\ X_s \mathcal{S}_i & \text{for} & \beta < 0 & \text{and} & f < 0.6 f_p & \\ \mathcal{X}_s \mathcal{S}_i & \text{for} & \beta < 0 & \text{and} & 06 f_p < f < 0.8 f_p & \end{array} \right. \;\;, \tag{2.56}$$

where f is the frequency, f_p is the peak frequency of the wind sea as computed from the input source term, $\mathcal{S}_i$ is the input source term (2.41), and $0 < X_s < 1$ is a reduction factor for $\mathcal{S}_i$, which is applied to swell with negative β only (defined by the user). $\mathcal{X}_s$ represents a linear reduction of X_s with f_p providing a smooth transition between the original and reduced input.

The drag coefficient that follows from Eq. (2.50) becomes unrealistically high for hurricane strength wind speeds, leading to unrealistically high wave growth rates. To alleviate this, the drag coefficient at the reference height C_r can be capped with a maximum allowed drag coefficient $C_{r,max}$, either as a simple hard limit

$$C_r = \min(C_r, C_{r,max} \;\;, \tag{2.57}$$

or with a smooth transition

$$C_r = C_{r,max} \tanh(C_r / C_{r,max}) \;\;. \tag{2.58}$$

Selection of the capped drag coefficient occurs at the compile stage of the code. The cap level and cap type can be set by the user. Defaults settings are $C_{r,max} = 2.5\,10^{-3}$ and Eq. (2.57).

The corresponding dissipation source term consists of two constituents. The (dominant) low-frequency constituent is based on an analogy with energy dissipation due to turbulence,

$$\mathcal{S}_{ds,l}(k,\theta) = -2\,u_*\,h\,k^2 \phi\,N(k,\theta)\,, \tag{2.59}$$

$$h = 4 \left(\int_0^{2\pi} \int_{f_h}^{\infty} F(f,\theta)\, df\, d\theta \right)^{1/2} . \tag{2.60}$$

$$\phi = b_0 + b_1 \tilde{f}_{p,i} + b_2 \tilde{f}_{p,i}^{-b_3} . \tag{2.61}$$

where h is a mixing scale determined from the high-frequency energy content of the wave field and where ϕ is an empirical function accounting for the development stage of the wave field. The linear part of Eq. (2.61) describes dissipation for growing waves. The nonlinear term has been added to allow for some control over fully grown conditions by defining a minimum value for ϕ ($\phi_{\min}$) for a minimum value of $f_{p,i}$ ($f_{p,i,\min}$). If $\phi_{\min}$ is below the linear curve, b_2 and b_3 are given as

$$b_2 = \tilde{f}_{p,i,\min}^{b_3} \left(\phi_{\min} - b_0 - b_1 \tilde{f}_{p,i,\min} \right) , \tag{2.62}$$

$$b_3 = 8 . \tag{2.63}$$

If $\phi_{\min}$ is above the linear curve, b_2 and b_3 are given as

$$\tilde{f}_a = \frac{\phi_{\min} - b_0}{b_1} \quad , \quad \tilde{f}_b = \max \left\{ \tilde{f}_a - 0.0025 \, , \, \tilde{f}_{p,i,\min} \right\} \quad , \tag{2.64}$$

$$b_2 = \tilde{f}_b^{b_3} \left[\phi_{\min} - b_0 - b_1 \tilde{f}_b \right] , \tag{2.65}$$

$$b_3 = \frac{b_1 \tilde{f}_b}{\phi_{\min} - b_0 - b_1 \tilde{f}_b} . \tag{2.66}$$

The above estimate of b_3 results in $\partial\phi / \partial \tilde{f}_{p,i} = 0$ for $\tilde{f}_{p,i} = \tilde{f}_b$. For $\tilde{f}_{p,i} < \tilde{f}_b$, ϕ is kept constant ($\phi = \phi_{\min}$).

The empirical high-frequency dissipation is defined as

$$\mathcal{S}_{ds,h}(k,\theta) = -a_0 \left(\frac{u_*}{g} \right)^2 f^3 \, \alpha_n^B \, N(k,\theta) \quad , \tag{2.67}$$

$$B = a_1 \left(\frac{f u_*}{g} \right)^{-a_2} \quad ,$$

$$\alpha_n = \frac{\sigma^6}{c_g \, g^2 \alpha_r} \int_0^{2\pi} N(k,\theta)\, d\theta \quad , \tag{2.68}$$

where α_n is Phillips' nondimensional high-frequency energy level normalized with α_r, and where a_0 through a_2 and α_r are empirical constants. This parameterization implies that $m = 5$ in the parametric tail, which has been preset in the model. Note that in the model Eq. (2.68) is solved assuming a deep water dispersion relation, in which case α_n is evaluated as

$$\alpha_n = \frac{2\,k^3}{\alpha_r} F(k) \quad . \tag{2.69}$$

The two constituents of the dissipation source term are combined using a simple linear combination, defined by the frequencies f_1 and f_2.

$$\mathcal{S}_{ds}(k,\theta) = \mathcal{A}\mathcal{S}_{ds,l} + (1-\mathcal{A})\,\mathcal{S}_{ds,h} \;, \tag{2.70}$$

$$\mathcal{A} = \begin{cases} 1 & \text{for} \quad f < f_l \quad , \\ \frac{f-f_2}{f_1-f_2} & \text{for} \quad f_1 \le f < f_2 \quad , \\ 0 & \text{for} \quad f_2 \le f \quad , \end{cases} \tag{2.71}$$

To enhance the smoothness of the model behavior for frequencies near the parametric cut-off f_{hf}, a similar transition zone is used between the prognostic spectrum and the parametric high-frequency tail as in Eq. (2.18)

$$N(k_i,\theta) = (1-\mathcal{B})\,N(k_i,\theta) + \mathcal{B}N(k_{i-1},\theta)\left(\frac{f_i}{f_{i-1}}\right)^{-m-2} \;, \tag{2.72}$$

where i is a discrete wavenumber counter, and where $\mathcal{B}$ is defined similarly to $\mathcal{A}$, ranging from 0 to 1 between f_2 and f_{hf}.

The frequencies defining the transitions and the length scale h are predefined in the model as

$$\left.\begin{array}{rcl} f_{hf} & = & 3.00\,f_{p,i} \\ f_1 & = & 1.75\,f_{p,i} \\ f_2 & = & 2.50\,f_{p,i} \\ f_h & = & 2.00\,f_{p,i} \end{array}\right\} \quad . \tag{2.73}$$

Furthermore, $f_{p,i,\min} = 0.009$ and $\alpha_r = 0.002$ are preset in the model. All other tunable parameters have to be provided by the user. Suggested and default values are given in Table 2.2.

Test results of these source terms in a global model implementation (Tolman, 2002f) suggested that (i) the model tuned in the classical way to fetch-limited growth for stable conditions underestimates deep-ocean wave growth

Tuned to :	a_0	a_1	a_2	b_0	b_1	$\phi_{\min}$
KC stable	4.8	$1.7\,10^{-4}$	2.0	$0.3\,10^{-3}$	0.47	0.003
KC unstable	4.5	$2.3\,10^{-3}$	1.5	$-5.8\,10^{-3}$	0.60	0.003

Table 2.2: Suggested constants in the source term package of Tolman and Chalikov. KC denotes Kahma and Calkoen (1992, 1994). First line represents default model settings.

(a deficiency apparently shared by the WAM model) and that (ii) effects of stability on the growth rate of waves as identified by Kahma and Calkoen (1992, 1994) should be included explicitly in the parameterization of the source terms. Ideally, both problems would be dealt with by theoretical investigation of the source terms. Alternatively, the wind speed u can be replaced by an effective wind speed u_e. In Tolman (2002f) the following effective wind speed is used :

$$\frac{u_e}{u} = \left(\frac{c_o}{1 + C_1 + C_2} \right)^{-1/2} , \tag{2.74}$$

$$C_1 = c_1 \tanh \left[\max(0, f_1 \{ \mathcal{ST} - \mathcal{ST}_o \}) \right] , \tag{2.75}$$

$$C_2 = c_2 \tanh \left[\max(0, f_2 \{ \mathcal{ST} - \mathcal{ST}_o \}) \right] , \tag{2.76}$$

$$\mathcal{ST} = \frac{hg}{u_h^2} \frac{T_a - T_s}{T_0} , \tag{2.77}$$

where $\mathcal{ST}$ is a bulk stability parameter, and T_a, T_s and T_0 are the air, sea and reference temperature, respectively. Furthermore, $f_1 \leq 0$, c_1 and c_2 have opposite signs and $f_2 = f_1 c_1 / c_2$. Following Tolman (2002f), default settings of $c_0 = 1.4$, $c_1 = -0.1$, $c_2 = 0.1$, $f_1 = -150$ and $\mathcal{ST}_o = -0.01$ in combination with the tuning to stable stratification wave growth data ('KC stable' parameter values in Table 2.2) are used. Note that this effective wind speed was derived for winds at 10 m height. The wind correction can be switched off by the user during compilation of the model, and default parameter settings can be redefined by the user in the program input files.

2.3.6 Input and dissipation (WAM-4 and variants) (F. Ardhuin)

The wind-wave interaction source terms used here are based on the wave growth theory of Miles (1957), modified by Janssen (1982). The pressure-slope correlations that give rise to part of the wave generation are parameterized following Janssen (1991). A wave dissipation term due to shear stresses variations in phase with the orbital velocity is added for the swell part of the spectrum, based on the swell decay observations of Ardhuin et al. (2008).

This parameterization was further extended by Abdalla and Bidlot (2002) to take into account a stronger gustiness in unstable atmospheric conditions. This effect is included in the present parameterization and is activated with the `!/STAB3` switch. Efforts have been made to make the present implementation as close as possible to the one in the ECWAM model (Bidlot et al., 2005), in particular the stress lookup tables were verified to be identical. Some minor differences remain which are under investigation. The source term reads (Janssen, 1991)

$$\mathcal{S}_{in}(k,\theta) = \frac{\rho_a}{\rho_w}\frac{\beta_{\max}}{\kappa^2}\mathrm{e}^Z Z^4 \left(\frac{u_\star}{C}\right)^2 \cos^{p_{in}}(\theta-\theta_u)\sigma N\left(k,\theta\right) + S_{out}(k,\theta), \quad (2.78)$$

where ρ_a and ρ_w are the air and water densities, $\beta_{\max}$ is a non-dimensional growth parameter (constant), κ is von Kármán' constant, and p_{in} is a constant that controls the directional distribution of $\mathcal{S}_{in}$. In the present implementation the air/water density ratio ρ_a/ρ_w is constant. We define $Z = \log(\mu)$ where μ is given by Janssen (1991, eq. 16), and corrected for intermediate water depths, so that

$$Z = \log(kz_1) + \kappa/\left[\cos\left(\theta-\theta_u\right)\left(u_\star/C + z_\alpha\right)\right], \quad (2.79)$$

where z_1 is a roughness length modified by the wave-supported stress τ_w, and z_α is a wave age tuning parameter[3]. The roughness z_1 is defined as,

[3]Although this tuning parameter z_α is not well described in WAM-Cycle4 documentation, it has an important effect on wave growth. Essentially it shifts the wave age of the long waves, which typically increases the growth, and even generates waves that travel faster than the wind. This accounts for some gustiness in the wind and should possibly be resolution-dependent. For reference, this parameter was not properly set in early versions of the SWAN model, as discovered by R. Lalbeharry.

$$U_{10} = \frac{u_\star}{\kappa}\log\left(\frac{z_u}{z_1}\right) \tag{2.80}$$

$$z_1 = \alpha_0\frac{\tau}{\sqrt{1-\tau_w/\tau}}, \tag{2.81}$$

where $\tau = u_\star^2$, and z_u is the height at which the wind is specified. These two equations provide an implicit functional dependence of $u_\star$ on U_{10} and τ_w/τ. This relationship is tabulated (Janssen, 1991; Bidlot et al., 2007).

An important part of the parameterization is the calculation of the wave-supported stress τ_w,

$$\tau_w = \left| \int_0^{k_{\max}} \int_0^{2\pi} \frac{\mathcal{S}_{in}(k',\theta)}{C} \left(\cos\theta, \sin\theta\right) \mathrm{d}k'\mathrm{d}\theta + \tau_{\mathrm{hf}}(u_\star,\alpha)\left(\cos\theta_u, \sin\theta_u\right)\right|, \tag{2.82}$$

which includes the resolved part of the spectrum, up to $k_{\max}$, as well as the stress supported by shorter waves, τ_{hf}. Assuming a f^{-X} diagnostic tail beyond the highest frequency, τ_{hf} is given by

$$\begin{aligned}\tau_{\mathrm{hf}}(u_\star,\alpha) &= \frac{u_\star^2}{g^2}\frac{\sigma_{\max}^X 2\pi\sigma}{2\pi C_g(k_{\max})}\int_0^{2\pi} N\left(k_{\max},\theta\right)\max\left\{0,\cos\left(\theta-\theta_u\right)\right\}^3 d\theta \\ &\quad \times\frac{\beta_{\max}}{\kappa^2}\int_{\sigma_{\max}}^{0.05*g/u_\star}\frac{\mathrm{e}^{Z_{\mathrm{hf}}}Z_{\mathrm{hf}}^4}{\sigma^{X-4}}\mathrm{d}\sigma\end{aligned} \tag{2.83}$$

where the second integral is a function of $u_\star$ and the Charnock coefficient α only, which is easily tabulated. In practice the calculation is coded with $X = 5$, and the variable Z_{hf} is defined by,

$$Z_{\mathrm{hf}}(\sigma) = \log(kz_1) + \min\left\{\kappa/\left(u_\star/C + z_\alpha\right), 20\right\}. \tag{2.84}$$

This parameterization is highly sensitive to the spectral level at $k_{\max}$. A higher spectral level will lead to a larger value of $u_\star$ and thus positive feedback on the wind input via z_1. This sensitivity is exacerbated by the sensitivity of the high frequency spectral level to the presence of swell via the dissipation term.

In the present implementation, an optional ad hoc reduction of $u_\star$ is implemented in order to allow a balance with a saturation-based dissipation.

This correction also reduces the drag coefficient at high winds. Essentially, the wind input is reduced for high frequencies and high winds. This is performed by replacing $u_\star$ in eq. (2.78) with $u'_\star(k)$ defined for each frequency as

$$\left(u'_\star\right)^2 = \left| u_\star^2 \left(\cos\theta_u, \sin\theta_u\right) - |s_u| \int_0^k \int_0^{2\pi} \frac{S_{in}\left(k',\theta\right)}{C} \left(\cos\theta, \sin\theta\right) \mathrm{d}k'\mathrm{d}\theta, \right| \tag{2.85}$$

where the sheltering coefficient $|s_u| \sim 1$ can be used to tune the stresses at high winds, which would be largely overestimated for $s_u = 0$. For $s_u > 0$ this sheltering is also applied within the diagnostic tail in eq. (2.83), which requires the estimation of a 3-dimensional look-up table for the high frequency stress, the third parameter being the energy level of the tail.

The swell dissipation part is activated by setting s_0 to a non-zero value. The parameterization of Ardhuin et al. (2008) is chosen with $s_0 = 3$, and is given by

$$\mathcal{S}_{out}\left(k,\theta\right) = -s_5 \frac{\rho_a}{\rho_w} \left\{ 2k\sqrt{2\nu\sigma} \right\} N\left(k,\theta\right), \tag{2.86}$$

if the air-sea boundary layer significant Reynolds number $Re = 2u_{\mathrm{orb},s}H_s/\nu_a$ is smaller than Re_c, and otherwise

$$\mathcal{S}_{out}\left(f,\theta\right) = -\frac{\rho_a}{\rho_w} \left\{ 16 f_e \sigma^2 u_{\mathrm{orb},s}/g \right\} N\left(k,\theta\right). \tag{2.87}$$

The significant surface orbital velocity is defined by

$$u_{\mathrm{orb},s} = 2\left[\iint \sigma^2 \, F(k,\theta) \, dkd\theta \right]^{1/2}. \tag{2.88}$$

The first equation (2.86) is the linear viscous decay by Dore (1978), with ν_a the air viscosity and s_5 is an $O(1)$ tuning parameter. Eq. (2.87) is a parameterization for the nonlinear turbulent decay. When comparing model results to observations, it was found that the model tended to underestimate large swells and overestimate small swells, with regional biases. This defect is likely due, in part, to errors in the generation or non-linear evolution of theses swells. However, it was chosen to adjust f_e as a function of the wind speed and direction,

$$f_e = s_1 f_{e,GM} + [|s_3| + s_2 \cos(\theta - \theta_u)]\, u_\star / u_{\rm orb}, \tag{2.89}$$

where $f_{e,GM}$ is the friction factor given by Grant and Madsen's (1979) theory for rough oscillatory boundary layers without a mean flow, using a roughness length adjusted to r_z times the roughness for the wind z_1. The coefficients s_1 is an $O(1)$ tuning parameter, and the coefficients s_2 and s_3 are two other adjustable parameters for the effect of the wind on the oscillatory air-sea boundary layer. When $s_2 < 0$, wind opposing swells are more dissipated than following swells. Further, if $s_3 > 0$ $\mathcal{S}_{out}$ is applied to the entire spectrum and not just the swell.

Due to the increase in high frequency input compared to WAM3, the dissipation function was adapted by Janssen (1994) from the WAM3 dissipation, and later reshaped by Bidlot et al. (2005). That later modification is referred to as "BAJ".
The generic form of the WAM4 dissipation term is,

$$S_{ds}\left(k,\theta\right)^{\rm WAM} = C_{ds}\overline{\alpha}^2\overline{\sigma}\left[\delta_1\frac{k}{\overline{k}} + \delta_2\left(\frac{k}{\overline{k}}\right)^2\right] N\left(k,\theta\right) \tag{2.90}$$

where C_{ds} is a non-dimensional constant δ_1 and δ_2 are weight parameters,

$$\overline{k} = \left[\frac{\int k^p N\left(k,\theta\right)\mathrm{d}\theta}{\int N\left(k,\theta\right)\mathrm{d}\theta}\right]^{1/p} \tag{2.91}$$

with p a constant power. Similarly the mean frequency is defined as

$$\overline{\sigma} = \left[\frac{\int \sigma^p N\left(k,\theta\right)\mathrm{d}\theta}{\int N\left(k,\theta\right)\mathrm{d}\theta}\right]^{1/p}, \tag{2.92}$$

so that the mean steepness is $\overline{\alpha} = E\overline{k}^2$.

The mean frequency also occurs in the definition of the maximum frequency of prognostic integration of the source terms. Since the definition of that frequency may be different from that of the source term it is defined with another exponent $p_{\rm tail}$.

Unfortunately these parameterizations are sensitive to swell. An increase in swell height typically reduces dissipation at the windsea peak (the first factor is reduced) and increases dissipation at high frequencies (the second

factor is reduced). For $p < 2$, as in the WAM-Cycle 4 and BAJ parameterizations, this sensitivity is much larger than the expected effect of short wave modulation by long waves.

The evidence of a threshold behavior of the wave breaking process, the underestimation of swell dissipation (Tolman, 2002f), the very strong dissipation at high frequency given by eq. (2.90), and the known deficiencies of WAM4 and BAJ source terms in the presence of swell has lead to several new parameterizations. The source term code was thus generalized to allow the use of WAM4, BAJ or others parameterization, via a simple change of the parameters in the namelists SIN3 and SDS3. The overall best parameterization found so far is described by Ardhuin et al. (2008), and is referred to as "ACC350". Modifications and further tuning are likely to further reduce the model errors. Nevertheless, at present, the default values of the namelist parameters correspond to BAJ.

The general form of the dissipation source terms computed with the ST3 switch thus takes the form of a combination of a WAM4-type term and a saturation-based term,

$$S_{\rm ds} = S_{\rm ds}^{\rm SAT} + S_{\rm ds}^{\rm TURB} + C_{\rm lf}\frac{1-S}{2}S_{\rm ds}^{\rm WAM4} + C_{\rm hf}\frac{1+S}{2}S_{\rm ds}^{\rm WAM4}. \tag{2.93}$$

The switch coefficients $C_{\rm lf}$ and $C_{\rm hf}$ allow the switching on and off of either the low (unsaturated) and/or high (saturated) part of the WAM4 dissipation term. The saturation level is given by,

$$S = \tanh\left[10\left(\frac{B'(k,\theta)}{B_r}\right)^{0.5} - B_{r2}\right], \tag{2.94}$$

with the saturation spectrum partially integrated over directions

$$B'(k,\theta) = \int_{\theta-\Delta_\theta}^{\theta+\Delta_\theta} \sigma k^3 A(k,\theta){\rm d}\theta', \tag{2.95}$$

All relevant source term parameters can be set via the namelists SIN3 and SDS3 to yield either the original WAM4 source terms or the BAJ form as modified by Bidlot et al. (2005).

The saturation term is given by,

Par.	WWATCH var.	namelist	WAM4	BAJ	ACC350
z_u	ZWND	SIN3	10.0	10.0	10.0
α_0	ALPHA0	SIN3	0.01	0.0095	0.0095
$\beta_{\max}$	BETAMAX	SIN3	1.2	1.2	1.75
p_{in}	SINTHP	SIN3	2	2	1.7
z_α	ZALP	SIN3	0.0110	0.0110	0.005
s_u	TAUWSHELTER	SIN3	0.0	0.0	-1
s_0	SWELLFPAR	SIN3	0.0	0.0	3
s_1	SWELLF	SIN3	0.0	0.0	0.7
s_2	SWELLF2	SIN3	0.0	0.0	-0.18
s_3	SWELLF3	SIN3	0.0	0.0	-0.15
Re_c	SWELLF4	SIN3	0.0	0.0	100000
s_5	SWELLF5	SIN3	0.0	0.0	1.2
z_r	Z0RAT	SIN3	0.0	0.0	0.04

Table 2.3: Parameter values for WAM4, BAJ and ACC350 source term parameterizations that can be reset via the SIN3 and SDS3 namelist. BAJ is generally better than WAM4, and the ACC350 combination was found to perform better than BAJ in all conditions except very high waves ($H_s > 11$ m, work in progress). However, the default parameters still corresponds to BAJ. Please note that the name of the variables only apply to the namelists. In the source term module the names are slightly different, with a doubled first letter, in order to differentiate the variables from the pointers to these variables, and the SWELLFx are combined in one array SSWELLF.

Par.	WWATCH var.	namelist	WAM4	BAJ	ACC350
$C_{\rm ds}$	SDSC1	SDS3	-4.5	-2.1	0.0
p	WNMEANP	SDS3	-0.5	0.5	0.5
$p_{\rm tail}$	WNMEANPTAIL	SDS3	-0.5	0.5	0.5
δ_1	SDSDELTA1	SDS3	0.5	0.4	0.0
δ_2	SDSDELTA2	SDS3	0.5	0.6	0.0
$C_{\rm ds}^{\rm sat}$	SDSC2	SDS3	0.0	0.0	-2.4×10^{-5}
$C_{\rm lf}$	SDSLF	SDS3	1.0	1.0	0.0
$C_{\rm hf}$	SDSHF	SDS3	1.0	1.0	0.0
Δ_θ	SDSDTH	SDS3	0.0	0.0	70
B_r	SDSBR	SDS3	0.0	0.0	1.2×10^{-3}
B_{r2}	SDSBR2	SDS3	1.0	1.0	0.8
B_0	SDSC4	SDS3	0.0	0.0	1.0
$p^{\rm sat}$	SDSP	SDS3	0.0	0.0	2.0
$C_{\rm turb}$	SDSC5	SDS3	0.0	0.0	0.0
$C_{\rm ds,6}$	SDSC6	SDS3	0.0	0.0	0.25
s_{m0}	SDSBM0	SDS3	1.0	1.0	1.0
s_{m1}	SDSBM1	SDS3	0.0	0.0	0.0
s_{m2}	SDSBM2	SDS3	0.0	0.0	0.0
s_{m3}	SDSBM3	SDS3	0.0	0.0	0.0
s_{m4}	SDSBM4	SDS3	0.0	0.0	0.0

Table 2.4: Parameter values for WAM4, BAJ and ACC350 source term parameterizations that can be reset via the SDS3 namelist. BAJ is generally better than WAM4, and the ACC350 combination was found to perform better than BAJ in all conditions except very high waves ($H_s > 11$ m, work in progress), with a general underestimation of the directional spread at high frequencies. However, the default parameters still corresponds to BAJ. Please note that the name of the variables only apply to the namelists. In the source term module the names are slightly different, with a doubled first letter, in order to differentiate the variables from the pointers to these variables, and the SDSMx are combined in one array SSDSM.

$$S_{\rm ds}^{\rm SAT}(k,\theta) = \sigma C_{\rm ds}^{\rm SAT} \left\{ C_{\rm ds,6} \left[\max\left\{ \frac{B(k)}{B_r P(kD)} - B_0, 0 \right\} \right]^{p^{\rm sat}} + (1 - C_{\rm ds,6}) \left[\max\left\{ \frac{B'(k,\theta)}{B_r P(kD)} - B_0, 0 \right\} \right]^{p^{\rm sat}} \right\} N(k,\theta) . \quad (2.96)$$

This form loosely follows the proposed saturation form of Phillips (1984), with the use of a partially integrated saturation spectrum B', defined by eq. (2.95), and its full integration B instead of Phillips' original non-integrated form. B is defined as

$$B(k) = \int_0^{2\pi} \sigma k^3 A(k,\theta) {\rm d}\theta. \quad (2.97)$$

Using B' is generally consistent with the lower saturation for broader spectra found by Banner et al. (2002), and allows a better fit to observed directional spectra.

This form thus differs from those proposed by Alves et al. (2003) and Van der Westhuysen et al. (2007). The $P(kD)$ factor in the denominator mimics the change in maximum wave steepness from deep to shallow water as given by Miche (1944). The function $P(x) = 1$ if $s_{m0} = 1$ and otherwise, $P(x) = [x/(s_{m1} + s_{m2}x + s_{m3}x^2 + s_{m4}x^3)]^2$. Using this form may lead in a double counting of dissipation when depth-induced breaking is parameterized separately.

The wave-turbulence interaction term of Teixeira and Belcher (2002) and Ardhuin and Jenkins (2006)), is given by

$$S_{\rm ds}^{\rm TURB}(k,\theta) = -2 C_{\rm turb} \sigma \cos(\theta_u - \theta) k \frac{\rho_a u_\star^2}{g \rho_w} N(k,\theta) . \quad (2.98)$$

The coefficient $C_{\rm turb}$ is of order 1 and can be used to adjust for ocean stratification and wave groupiness.

2.3.7 Linear input (Cavaleri and Malanotte-Rizzoli)

A linear input source term is useful to allow for the consistent spin-up of a model from quiescent conditions, and to improve initial wave growth behavior. Available in WAVEWATCH III $^{\rm TM}$ is the parameterization of Cavaleri

and Malanotte-Rizzoli (1981), with a filter for low-frequency energy as introduced by Tolman (1992). The input term can be expressed as

$$\mathcal{S}_{lin}(k,\theta) = 80 \left(\frac{\rho_a}{\rho_w}\right)^2 g^{-2} k^{-1} \max\left[0, u_* \cos(\theta - \theta_w)\right]^4 \; G \quad , \tag{2.99}$$

where ρ_a and ρ_w are the densities of air and water, respectively, and where G is the filter function

$$G = \exp\left[-\left(\frac{f}{f_{filt}}\right)^{-4}\right] \quad . \tag{2.100}$$

In Tolman (1992) the filter frequency f_{filt} was given as the Pierson-Moskowitz frequency f_{PM}, which in turn was estimated as in Eq. (2.40). In the present implementation, the filter can be related to both f_{PM} and the cut-off frequency of the prognostic part of the spectrum f_{hf} as defined in Eq. (2.17)

$$f_{filt} = \max\left[\alpha_{PM} f_{PM}, \alpha_{hf} f_{hf}\right] \quad , \tag{2.101}$$

where the constants α_{PM} and α_{hf} are user-defined. Default values of these constants are set to $\alpha_{PM} = 1$ and $\alpha_{hf} = 0.5$. Addition of the dependency on f_{hf} assures consistent growth behavior at all fetches, without the possibility of low-frequency linear growth to dominate at extremely short fetches.

2.3.8 Bottom friction (JONSWAP)

A simple parameterization of bottom friction is the empirical, linear JONSWAP parameterization (Hasselmann et al., 1973), as used in the WAM model (WAMDIG, 1988). Using the notation of Tolman (1991), this source term can be written as

$$\mathcal{S}_{bot}(k,\theta) = 2\Gamma \, \frac{n - 0.5}{gd} \, N(k,\theta) \, , \tag{2.102}$$

where Γ is an empirical constant, which is estimated as $\Gamma = -0.038 \, \mathrm{m^2 s^{-3}}$ for swell (Hasselmann et al., 1973), and as $\Gamma = -0.067 \, \mathrm{m^2 s^{-3}}$ for wind seas (Bouws and Komen, 1983). n is the ratio of phase velocity to group velocity given by (2.6). The default value for $\Gamma = -0.067$ can be redefined by the user in the model input files.

2.3.9 Surf breaking (Battjes-Janssen) (J. H. G. M. Alves)

The implementation in WAVEWATCH III ™ of depth-induced breaking algorithms is intended to extend the applicability of the model to within shallow water environments, where wave breaking, among other depth-induced transformation processes, becomes important.

The first step in that direction is the inclusion of the approach of Battjes and Janssen (1978, henceforth denoted as BJ78), which is based on the assumption that all waves in a random field exceeding a threshold height, defined as a function of bottom topography parameters, will break. For a random wave field, the fraction of waves satisfying this criterion is determined by a statistical description of surf-zone wave heights (i.e., a Rayleigh-type distribution, truncated at a depth-dependent wave-height maximum).

The bulk rate δ of spectral energy density dissipation of the fraction of breaking waves, as proposed by BJ78, is estimated using an analogy with dissipation in turbulent bores as

$$\delta = 0.25\, Q_b\, f_m\, H_{\max}^2 \quad , \tag{2.103}$$

where Q_b is the fraction of breaking waves in the random field, f_m is the mean frequency and $H_{\max}$ is the maximum individual height a component in the random wave field can reach without breaking (conversely, above which all waves would break). In BJ78 the maximum wave height $H_{\max}$ is defined using a Miche-type criterion (Miche, 1944),

$$\bar{k} H_{\max} = \gamma_M \tanh(\bar{k} d) \quad , \tag{2.104}$$

where γ_M is a constant factor. This approach also removes energy in deep-water waves exceeding a limiting steepness. This can potentially result in double counting of dissipation in deep-water waves. Alternatively, $H_{\max}$ can be defined using a McCowan-type criterion, which consists of simple constant ratio

$$H_{\max} = \gamma\, d \quad , \tag{2.105}$$

where d is the local water depth and γ is a constant derived from field and laboratory observation of breaking waves. This approach will exclusively represent depth-induced breaking. Although more general breaking criteria for $H_{\max}$ as a simple function of local depth exist (e.g., Thornton and

Guza, 1983), it should be noted that the coefficient γ refers to the maximum height of an individual breaking wave within the random field. McCowan (1894) calculated the limiting wave-height-to-depth ratio for a solitary wave propagating on a flat bottom to be 0.78, which is still used presently as a conservative criteria in engineering applications. The average value found by Battjes and Janssen (1978) was $\gamma = 0.73$. More recent analyses of waves propagating over reefs by Nelson (1994, 1997) suggest a ratio of 0.55.

The fraction of breaking waves Q_b is determined in terms of a Rayleigh-type distribution truncated at $H_{\max}$ (i.e., all broken waves have a height equal to H_{max}), which results in the following expression:

$$\frac{1 - Q_b}{-\ln Q_b} = \left(\frac{H_{rms}}{H_{\max}} \right) \quad , \tag{2.106}$$

where H_{rms} is the root-mean-square wave height. In the current implementation, the implicit equation (2.106) is solved for Q_b iteratively. With the assumption that the total spectral energy dissipation δ is distributed over the entire spectrum so that it does not change the spectral shape (Eldeberky and Battjes, 1996) the following depth-induced breaking dissipation source function is obtained

$$S_{db}(k,\theta) = -\alpha \frac{\delta}{E} F(k,\theta) = -0.25\, \alpha\, Q_b\, f_m \frac{H_{\max}^2}{E} F(k,\theta) \quad , \tag{2.107}$$

where E is the total spectral energy, and $\alpha = 1.0$ is a tunable parameter. The user can select between Eqs. (2.104) and (2.105), and adjust γ and α. Defaults are Eq. (2.105), $\gamma = 0.73$ and $\alpha = 1.0$.

2.3.10 Triad interactions

Not yet available.

2.3.11 Bottom scattering (R. Magne and F. Ardhuin)

Waves propagating over a sloping bottom are partially reflected. In the limit of small variation in water depth ΔH with respect to the mean water depth H, the reflection coefficient is proportional to the bottom spectrum

Kreisel (1949) and leads to a redistribution of wave energy in direction. This process may be formulated as a source term, which leads to accurate reflection coefficients when considering the evolution of the spectrum over scales larger than the bottom auto-correlation length, with reasonable accuracy up to $\Delta d/d \simeq 0.6$ (Ardhuin and Magne, 2007). The source term reads,

$$S_{\rm sc}(\mathbf{k}) = \frac{\pi}{2} \int_0^{2\pi} \frac{k'^2 M^2(\mathbf{k},\mathbf{k}')}{\sigma\sigma' \left(k' C'_g + \mathbf{k}\cdot\mathbf{U}\right)} F^B(\mathbf{k}-\mathbf{k}') \left[N(\mathbf{k}') - N(\mathbf{k}')\right] {\rm d}\theta', \quad (2.108)$$

with the coupling coefficient

$$M(\mathbf{k},\mathbf{k}') \simeq M_b(\mathbf{k},\mathbf{k}') = \frac{g\mathbf{k}\cdot\mathbf{k}'}{\cosh(kd)\cosh(k'd)} \quad (2.109)$$

where the effect of bottom-induced current and elevation changes are neglected, as appropriate for low to moderate current velocity relative to the intrinsic phase speed, i.e. $U/C < 0.3$. For larger Froude numbers, in particular in near-blocking conditions, the present implementation is not expected to be accurate. In Eq. (2.108), k and k' are related by the resonance condition, $\omega = \omega'$, i.e. $\sigma + \mathbf{k}\cdot\mathbf{U} = \sigma' + \mathbf{k}'\cdot\mathbf{U}$, where $\mathbf{U}$ is the phase advection velocity (see, e.g., The WISE Group, 2007).

The bottom spectrum $F^B(\mathbf{k})$ is specified in the file bottomspectrum.inp. This spectrum may be determined from multi beam bathymetric data. In the absence of detailed bathymetric data, the sand dune spectrum may be parameterized based on the work of Hino (1968). Recent observations generally confirm the earlier data on sand dune spectra (Ardhuin and Magne, 2007), with a non-dimensional constant spectrum for large k, i.e. $F^B(\mathbf{k}) \sim k^{-4}$.

The bottom spectrum is double-sided for simplicity of calculation and normalized such that the bottom variance (in square meters) is

$$< d^2 >= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F^B(k_x,k_y){\rm d}k_x{\rm d}k_y. \quad (2.110)$$

In the present implementation this bottom spectrum is assumed to be the same at all grid points.

The source term is computed according to different methods depending on the value of the current. For zero current, the interactions only involves waves of the same frequency and the interaction is always the same and linear in terms of the directional spectrum. In this case the interaction is expressed

as a matrix problem, and the interaction matrices are diagonalized as a preprocessing step for a finite number of wavenumber magnitudes (Ardhuin and Herbers, 2002). The cost of this preprocessing increases linearly with the number of discrete wavenumbers.

For non-zero current, the interaction pattern depends on the current magnitude and direction (magnitude only for an isotropic bottom spectrum), and this would increase the overhead cost by at least one order of magnitude. In the present implementation, the interaction integration is recomputed at every source term call.

2.3.12 User-defined source terms

This slot is intended for a source term that is not yet classified in Eq. (2.16). Almost by definition, it cannot be provided here.

2.4 Output parameters

The wave model provides output of the following gridded fields of mean wave parameters. Some of these parameters can also be found in the output for selected points. For activation of the output see section 4.4.5

1) The mean water depth (m).
2) The mean current velocity (vector, m/s).
3) The mean wind speed (vector, m/s). This wind speed is alway the speed as input to the model, i.e., is not corrected for the current speed.
4) The air-sea temperature difference (°C).
5) The friction velocity u_* (scalar). Definition depends on selected source term parameterization (m/s). An alternative vector version of the stresses is available for research (requires user intervention in the code).
6) Significant wave height (m) [see Eq. (2.23)]

$$H_s = 4\sqrt{E}\,. \tag{2.111}$$

7) Mean wave length (m) [see Eq. (2.22)]

$$L_m = 2\pi\overline{k^{-1}}\,. \tag{2.112}$$

8) Mean wave period (s)

$$T_m = 2\pi\overline{\sigma^{-1}}\,. \tag{2.113}$$

9) Mean wave direction (degr., meteorological convention)

$$\theta_m = \mathrm{atan}\left(\frac{b}{a}\right)\,, \tag{2.114}$$

$$a = \int_0^{2\pi}\int_0^{\infty} \cos(\theta) F(\sigma,\theta)\; d\sigma\; d\theta\,, \tag{2.115}$$

$$b = \int_0^{2\pi}\int_0^{\infty} \sin(\theta) F(\sigma,\theta)\; d\sigma\; d\theta\,. \tag{2.116}$$

10) Mean directional spread (degr.; Kuik et al., 1988)

$$\sigma_\theta = \left[2\left\{1-\left(\frac{a^2+b^2}{E^2}\right)^{1/2}\right\}\right]^{1/2}\,, \tag{2.117}$$

11) Peak frequency (Hz), calculated from the one-dimensional frequency spectrum using a parabolic fit around the discrete peak.
12) Peak direction (degr.), defined like the mean direction, using the frequency/wavenumber bin containing of the spectrum $F(k)$ that contains the peak frequency only.
13) Peak frequency of the wind sea part of the spectrum. For WAM-3 input, this is the highest local peak in the one-dimensional spectrum, if this frequency is higher than half the PM frequency and smaller than 0.75 times the maximum discrete frequency (otherwise undefined). For the Tolman and Chalikov input, it is calculated using Eqs. (2.54) and (2.55).
14) Wind sea direction (degr.), defined like the mean direction, using the frequency/wavenumber bin containing the peak frequency of the wind sea only.
15) Wave heights H_s of partitions of the spectrum (see below).

16) Peak (relative) periods of partitions of the spectrum (parabolic fit).
17) Peak wave lengths of partitions of the spectrum (from peak period).
18) Mean direction of partitions of the spectrum.
19) Directional spread of partition of the spectrum Cf. Eq. (2.117).
20) Wind sea fraction of partition of the spectrum (see below).
21) Wind sea fraction of the entire spectrum.
22) Number of partitions found in the spectrum.
23) Average time step in the source term integration (s).
24) Cut-off frequency f_c (Hz, depends on parameterization of input and dissipation).
25) Ice concentration.
26) Water level.
27) Near-bottom rms excursion amplitude

$$a_{b,rms} = \left[2 \iint \frac{1}{\sinh^2 kd} \, F(k,\theta) \, dkd\theta \right]^{1/2} . \tag{2.118}$$

28) Near-bottom rms orbital velocity

$$u_{b,rms} = \left[2 \iint \frac{\sigma^2}{\sinh^2 kd} \, F(k,\theta) \, dkd\theta \right]^{1/2} . \tag{2.119}$$

29) Radiation stresses

$$S_{xx} = \rho_w g \iint \left(n - 0.5 + n \cos^2\theta \right) \, F(k,\theta) \, dkd\theta \, , \tag{2.120}$$

$$S_{xy} = \rho_w g \iint n \sin\theta \cos\theta \, F(k,\theta) \, dkd\theta \, , \tag{2.121}$$

$$S_{yy} = \rho_w g \iint \left(n - 0.5 + n \sin^2\theta \right) \, F(k,\theta) \, dkd\theta \, , \tag{2.122}$$

where

$$n = \frac{1}{2} + \frac{kd}{\sinh 2kd} \, . \tag{2.123}$$

30) Slot for user defined parameter (requires modification of code).
31) Idem.

Output types 15 through 22 are based on partitioning of the spectrum into individual wave fields. The method of Hanson and Phillips (2001) is used, implemented as described in Tracy et al. (2007). With this, a 'wind sea fraction' W is introduced

$$W = E^{-1}\, E|_{U_p>c} \quad , \tag{2.124}$$

where E is the total spectral energy, and $E|_{U_p>c}$ is the energy in the spectrum for which the projected wind speed U_p is larger than the local wave phase velocity $c = \sigma/k$. The latter defines an area in the spectrum under the direct influence of the wind. To allow for nonlinear interactions to shift this boundary to lower frequencies, and subsequently to have fully grown wind seas inside this are, U_p includes a multiplier C_{mult}

$$U_p = C_{mult} U_{10} \cos(\theta - \theta_w) \quad . \tag{2.125}$$

The multiplier can be set by the user. The default value is $C_{mult} = 1.7$

Old output types 13 and 14 have become obsolete with the introduction of the partitioned output types 15 through 22. However, they are retained in the present model release to provide required downward compatibility with model version 1.18 at NCEP. These output types are likely to be retired in upcoming versions of WAVEWATCH III ™.

3 Numerical approaches

3.1 Basic concepts

Equation (2.8) or (2.12) represents the basic equations of the wave model. However, modified versions of these equations are used in the model, where (a) they are solved on a variable wavenumber grid (see below), where (b) a modified versions of these equations are used to properly described dispersion for discretized equations in selected numerical schemes (see section 3.3), and where (c) sub-grid obstacles such as islands are considered (see section 3.3).

If (2.8) or (2.12) is solved directly, an effective reduction of spectral resolution occurs in shallow water (see Tolman and Booij, 1998). This loss of resolution can be avoided if the equation is solved on a variable wavenumber grid, which implicitly incorporates the kinematic wavenumber changes due to shoaling. Such a wavenumber grid corresponds to a spatially and temporally invariant frequency grid (Tolman and Booij, 1998). The corresponding local wavenumber grid can be calculated directly from the invariant frequency grid and the dispersion relation (2.1), and hence becomes a function of the local depth d. To accommodate economical calculations of S_{nl}, a logarithmic frequency grid is adopted,

$$\sigma_{m+1} = X_\sigma \, \sigma_m \, , \tag{3.1}$$

where m is a discrete grid counter in k-space. X_σ is defined by the user in the input files of the program. Traditionally, in most applications of third-generation models $X_\sigma = 1.1$ is used.

The effects of a spatially varying grid will be discussed for the Cartesian equation (2.8) only. Adaptation to the spherical grid is trivial. Denoting the variable wavenumber grid with κ, the balance equation becomes

$$\frac{\partial}{\partial t}\frac{N}{c_g} + \frac{\partial}{\partial x}\frac{\dot{x}N}{c_g} + \frac{\partial}{\partial y}\frac{\dot{y}N}{c_g} + \frac{\partial}{\partial \kappa}\frac{\dot{\kappa}N}{c_g} + \frac{\partial}{\partial \theta}\frac{\dot{\theta}N}{c_g} = 0 \, , \tag{3.2}$$

$$\dot{\kappa}\,\frac{\partial k}{\partial \kappa} = c_g^{-1}\frac{\partial \sigma}{\partial d}\left(\frac{\partial d}{\partial t} + \mathbf{U}\cdot\nabla_x d\right) - \mathbf{k}\cdot\frac{\partial \mathbf{U}}{\partial s} \, . \tag{3.3}$$

Equation (3.2) is solved using a fractional step method, as is commonplace in wave modeling. The first step considers temporal variations of the depth,

and corresponding changes in the wavenumber grid. As is discussed by Tolman and Booij (1998), this step can be invoked sparsely. By splitting off effects of (temporal) water level variations, the grid becomes invariant, and the depth becomes quasi-steady for the remaining fractional steps. Other fractional steps consider spatial propagation, intra-spectral propagation and source terms.

The multiple splitting technique results in a model that can efficiently be vectorized and parallelized at the same time. The time splitting furthermore allows for the use of separate partial or dynamically adjusted time steps in the different fractional steps of the model. WAVEWATCH III ^{TM} makes a distinction between 4 different time steps.

1) The 'global' time step Δt_g, by which the entire solution is propagated in time, and at which intervals input winds and currents are interpolated. This time step is provided by the user, but can be reduced within the model to reach a requested input or output time.
2) The second time step is the time step for spatial propagation. The user supplies a reference maximum propagation time step for the lowest model frequency $\Delta t_{p,r}$, assuming no currents, and no grid motion. For the frequency with counter m, the maximum time step $\Delta t_{p,m}$ is calculated within the model as

$$\Delta t_{p,m} = \frac{\dot{x}_{p,r}}{\dot{x}_{p,m}} \Delta t_{p,r} \; . \tag{3.4}$$

   where $\dot{x}_{p,r}$ is the maximum advection speed for the longest waves without currents or grid motion, and $\dot{x}_{p,m}$ is the actual maximum advection speed (including current) for frequency m. If the propagation time step is smaller than the global time step, the propagation effects are calculated with a number of successive smaller time steps. This generally implies that several partial time steps are used for the lowest frequency, but that the highest frequencies are propagated over the interval Δt_g with a single calculation. The latter results in a significantly more efficient model, particularly if higher-order accurate propagation schemes are used. Note that $\Delta t_{p,m}$ may be defined bigger than Δt_g, and that this has potential impact in model economy for cases with (strong) currents.
3) The third time step is the time step for intra-spectral propagation. For large-scale and deep-water grids this time step can generally be

taken equal to the global time step Δt_g. For shallow water grids, smaller intra-spectral propagation time steps allow for larger effects of refraction within the stability constraints of the scheme. Note that the order of invoking spatial and intra-spectral propagation is alternated to enhance numerical accuracy. If strong refraction o narrow swells occur, this may result in a notable undulation of mean wave parameters. This can be avoided by setting this time step to an even integer fraction of Δt_g.

4) The final time step is the time step for the integration of the source terms, which is dynamically adjusted for each separate grid point and global time step Δt_g (see section 3.5). This results in more accurate calculations for rapidly changing wind and wave conditions, and a more economical integration for slowly varying conditions.

The following sections deal with the separate steps in the fractional step method, model input, ice treatment and boundary data transfer between separate model grids.

3.2 Depth variations in time

Temporal depth variations result in a change of the local wavenumber grid. Because the wavenumber spectrum is invariant with respect to temporal changes of the depth, this corresponds to a simple interpolation of the spectrum from the old grid to the new grid, without changes in the spectral shape. As discussed above, the new grid simply follows from the globally invariant frequency grid, the new water depth d and the dispersion relation (2.1). The time step of updating the water level is generally dictated by physical time scales of water level variations, but not by numerical considerations (Tolman and Booij, 1998).

The interpolation to the new wavenumber grid is performed with a simple conservative interpolation method. In this interpolation the old spectrum is first converted to discrete action densities by multiplication with the spectral bin widths. This discrete action then is redistributed over the new grid cf. a regular linear interpolation. The new discrete actions then are converted into a spectrum by division by the (new) spectral bin widths. The conversion

requires a parametric extension of the original spectrum at high and low frequencies because the old grid generally will not completely cover the new grid. Energy/action in the old spectrum at low wavenumbers that are not resolved by the new grid is simply removed. At low wavenumbers in the new grid that are not resolved by the old grid zero energy/action is assumed. At high wavenumbers in the new grid the usual parametric tail is applied if necessary. The latter correction is rare, as the highest wavenumbers usually correspond to deep water.

In practical applications the grid modification is usually relevant for a small fraction of the grid points only. To avoid unnecessary calculations, the grid is transformed only if the smallest relative depth kd in the discrete spectrum is smaller than 4. Furthermore, the spectrum is interpolated only if the spatial grid point is not covered by ice, and if the largest change of wavenumber is at least $0.05\Delta k$.

3.3 Spatial propagation

Spatial propagation is described by the first terms of Eq. (3.2). For the spherical grid [Eq. (2.12)], the corresponding spatial propagation step becomes

$$\frac{\partial \mathcal{N}}{\partial t} + \frac{\partial}{\partial \phi}\,\dot{\phi}\mathcal{N} + \frac{\partial}{\partial \lambda}\,\dot{\lambda}\mathcal{N} = 0\,, \tag{3.5}$$

where the propagated quantity $\mathcal{N}$ is defined as $\mathcal{N} \equiv N\, c_g^{-1} \cos\phi$. For the Cartesian grid, a similar equation is found propagating $\mathcal{N} \equiv N\, c_g^{-1}$. In this section equations for the more complicated spherical grid are presented only. Conversion to a Cartesian grid is generally a simplification and is trivial.

Equation (3.5) in form is identical to the conventional deep-water propagation equation, but includes effects of both limited depths and currents. At the land-sea boundaries, wave action propagating toward the land is assumed to be absorbed without reflection, and waves propagating away from the coast are assumed to have no energy at the coastline. For so-called 'active boundary points' where boundary conditions are prescribed, a similar approach is used. Action traveling toward such points is absorbed, whereas action at the boundary points is used to estimate action fluxes for components traveling into the model.

Three separate issues arise regarding the spatial propagation. The first is the actual propagation scheme used (section 3.3.1). The second is the occurrence and alleviation of the Garden Sprinkler Effect (GSE) as discussed in section 3.3.2. The third is the inclusion of the effects of unresolved obstacles on wave propagation. Sub-grid treatment of such obstacles is addressed in WAVEWATCH III ^{TM} as part of the numerical propagation scheme, and is discussed in section 3.3.3.

3.3.1 Propagation schemes

A simple and cheap first order upwind scheme has been included, mainly for testing during development of WAVEWATCH III ^{TM}. To assure numerical conservation of action, a flux or control volume formulation is used. The flux between grid points with counters i and $i-1$ in ϕ-space ($\mathcal{F}_{i,-}$) is calculated as

$$\mathcal{F}_{i,-} = \left[\dot{\phi}_b \, \mathcal{N}_u \right]^n_{j,l,m} \; , \tag{3.6}$$

$$\dot{\phi}_b = 0.5 \, \left(\dot{\phi}_{i-1} + \dot{\phi}_i \right)_{j,l,m} \; , \tag{3.7}$$

$$\mathcal{N}_u = \left\{ \begin{array}{ll} \mathcal{N}_{i-1} & \text{for} \quad \dot{\phi}_b \geq 0 \\ \mathcal{N}_i & \text{for} \quad \dot{\phi}_b < 0 \end{array} \right. \; , \tag{3.8}$$

where j, l and m are discrete grid counters in λ-, θ- and k-spaces, respectively, and n is a discrete time step counter. $\dot{\phi}_b$ represents the propagation velocity at the 'cell boundary' between points i and $i-1$, and the subscript u denotes the 'upstream' grid point. At land-sea boundaries, $\dot{\phi}_b$ is replaced by $\dot{\phi}$ at the sea point. Fluxes between points i and $i+1$ ($\mathcal{F}_{i,+}$) are obtained by replacing $i-1$ with i and i with $i+1$. Fluxes in λ-space are calculated similarly, changing the appropriate grid counters and increments. The 'action density' ($\mathcal{N}^{n+1}$) at time $n+1$ is estimated as

$$\mathcal{N}^{n+1}_{i,j,l,m} = \mathcal{N}^n_{i,j,l,m} + \frac{\Delta t}{\Delta \phi} \left[\mathcal{F}_{i,-} - \mathcal{F}_{i,+} \right] + \frac{\Delta t}{\Delta \lambda} \left[\mathcal{F}_{j,-} - \mathcal{F}_{j,+} \right] \; , \tag{3.9}$$

where Δt is the propagation time step, and $\Delta\phi$ and $\Delta\lambda$ are the latitude and longitude increments, respectively. Equations (3.6) through (3.8) with $\mathcal{N} = 0$ on land and applying Eq. (3.9) on sea points only automatically invokes the required boundary conditions.

Note that Eq. (3.9) represents a two-dimensional implementation of the scheme, for which the norm of the actual advection vectors needs to be used in Eq. (3.4). Note furthermore, that this implies a CFL criterion for the full equation, which is generally more stringent than that for a scheme where λ and ϕ propagation are treated separately as in the third order schemes discussed below. For a grid with equal increments in both directions, this results in a maximum time step that is a factor $1/\sqrt{2}$ smaller for the first order scheme than for the third order schemes.

Also available is the QUICKEST scheme (Leonard, 1979; Davis and More, 1982) combined with the ULTIMATE TVD (total variance diminishing) limiter (Leonard, 1991). This is the default propagation scheme for WAVEWATCH III ^{TM}. This scheme is third-order accurate in both space and time, and has been selected based on the extensive intercomparison of higher order finite difference schemes for water quality models performed by (see Cahyono, 1994; Falconer and Cayhono, 1993; Tolman, 1995). This scheme is applied to propagation in longitudinal and latitudinal directions separately, alternating the direction to be treated first.

In the QUICKEST scheme the flux between grid points with counters i and $i-1$ in ϕ-space ($\mathcal{F}_{i,-}$) is calculated as[4]

$$\mathcal{F}_{i,-} = \left[\dot{\phi}_b \, \mathcal{N}_b \right]^n_{j,l,m} \, , \tag{3.10}$$

$$\dot{\phi}_b = 0.5 \, \left(\dot{\phi}_{i-1} + \dot{\phi}_i \right) \, , \tag{3.11}$$

$$\mathcal{N}_b = \frac{1}{2} \left[\, (1+C)\mathcal{N}_{i-1} + (1-C)\mathcal{N}_i \, \right] - \left(\frac{1-C^2}{6} \right) \mathcal{CU} \, \Delta\phi^2, \tag{3.12}$$

$$\mathcal{CU} = \begin{cases} (\, \mathcal{N}_{i-2} - 2\mathcal{N}_{i-1} + \mathcal{N}_i \,) \; \Delta\phi^{-2} & \text{for} \quad \dot{\phi}_b \geq 0 \\ (\, \mathcal{N}_{i-1} - 2\mathcal{N}_i + \mathcal{N}_{i+1} \,) \, \Delta\phi^{-2} & \text{for} \quad \dot{\phi}_b < 0 \end{cases} \, , \tag{3.13}$$

$$C = \frac{\dot{\phi}_b \, \Delta t}{\Delta\phi} \, , \tag{3.14}$$

where $\mathcal{CU}$ is the (upstream) curvature of the action density distribution, and where C is a CFL number including a sign to identify the propagation direction. Like the first order scheme, this scheme gives stable solutions for

[4] Fluxes ($\mathcal{F}_{i,+}$) between grid points with counters $i+1$ and i again are obtained by substituting the appropriate indices.

$|C| \leq 1$. To assure that this scheme does not generate aphysical extrema, it is used in combination with the ULTIMATE limiter. This limiter uses the central, upstream and downstream action density (suffices c, u and d, respectively), which are defined as

$$\begin{array}{lll} \mathcal{N}_c = \mathcal{N}_{i-1} \,, & \mathcal{N}_u = \mathcal{N}_i \,, & \mathcal{N}_d = \mathcal{N}_{i-2} \quad \text{for} \quad \dot{\phi}_b \geq 0 \\ \mathcal{N}_c = \mathcal{N}_i \,, & \mathcal{N}_u = \mathcal{N}_{i-1} \,, & \mathcal{N}_d = \mathcal{N}_{i+1} \quad \text{for} \quad \dot{\phi}_b < 0 \end{array} \quad . \tag{3.15}$$

To assess if the initial state and the solution show similar monotonic or non-monotonic behavior, the normalized action $\tilde{\mathcal{N}}$ is defined

$$\tilde{\mathcal{N}} = \frac{\mathcal{N} - \mathcal{N}_u}{\mathcal{N}_d - \mathcal{N}_u} \,. \tag{3.16}$$

If the initial state is monotonic (i.e., $0 \leq \tilde{\mathcal{N}}_c \leq 1$), the (normalized) action at the cell boundary $\mathcal{N}_b$ is limited to

$$\tilde{\mathcal{N}}_c \leq \tilde{\mathcal{N}}_b \leq 1 \quad , \quad \tilde{\mathcal{N}}_b \leq \tilde{\mathcal{N}}_c \, C^{-1} \quad . \tag{3.17}$$

otherwise

$$\tilde{\mathcal{N}}_b = \tilde{\mathcal{N}}_c \,. \tag{3.18}$$

An alternative scheme is necessary if one of the two grid points adjacent to the cell boundary is on land or represents an active boundary point. In such cases, Eqs. (3.7) and (3.12) are replaced by

$$\dot{\phi}_b = \dot{\phi}_s \,, \tag{3.19}$$

$$\mathcal{N}_b = \mathcal{N}_u \,, \tag{3.20}$$

where the suffix s indicates the (average of) the sea point(s). This boundary condition represents a simple first order upwind scheme, which does not require the limiter (3.15) through (3.18).

The final propagation scheme, similar to Eq. (3.9), becomes

$$\mathcal{N}^{n+1}_{i,j,l,m} = \mathcal{N}^{n}_{i,j,l,m} + \frac{\Delta t}{\Delta \phi} \left[\mathcal{F}_{i,-} - \mathcal{F}_{i,+} \right] \,. \tag{3.21}$$

The scheme for propagation in λ-space is simply obtained by rotating indices and increments in the above equations[5]

[5] The 'soft' boundary treatment as described on page 31 of Tolman (2002e) is no longer available, because it is incompatible with the advanced nesting techniques introduced in model version 3.14.

Note that the ULTIMATE QUICKEST scheme is implemented as alternate one-dimensional schemes, for which the maxima of component advection speeds need to be used in Eq. (3.4). For consistency, the same time steps are always used for λ and ϕ propagation for a given component.

3.3.2 Garden Sprinkler Effect

The ULTIMATE QUICKEST scheme is sufficiently free of numerical diffusion for the so-called 'Garden Sprinkler Effect' (GSE) to occur, i.e., a continuous swell field disintegrates into a set of discrete swell fields due to the discrete description of the spectrum (Booij and Holthuijsen, 1987, Fig. 3c). Several GSE alleviation methods are available in WAVEWATCH III ^{TM}.

The classical GSE alleviation method is given by Booij and Holthuijsen (1987), who derived an alternative propagation equation for the discrete spectrum, including a diffusive correction to account for continuous dispersion in spite of the discrete spectral description. This correction influences spatial propagation only, which for general spatial coordinates (x, y) becomes

$$\frac{\partial \mathcal{N}}{\partial t} + \frac{\partial}{\partial x}\left[\dot{x}\mathcal{N} - D_{xx}\frac{\partial \mathcal{N}}{\partial x}\right] + \frac{\partial}{\partial y}\left[\dot{y}\mathcal{N} - D_{yy}\frac{\partial \mathcal{N}}{\partial y}\right] - 2D_{xy}\frac{\partial^2 \mathcal{N}}{\partial x \partial y} = 0\,, \quad (3.22)$$

$$D_{xx} = D_{ss}\,\cos^2\theta + D_{nn}\,\sin^2\theta\,, \quad (3.23)$$

$$D_{yy} = D_{ss}\,\sin^2\theta + D_{nn}\,\cos^2\theta\,, \quad (3.24)$$

$$D_{xy} = (D_{ss} - D_{nn})\,\cos\theta\,\sin\theta\,, \quad (3.25)$$

$$D_{ss} = (\Delta c_g)^2\,T_s/12\,, \quad (3.26)$$

$$D_{nn} = (c_g \Delta\theta)^2\,T_s/12\,, \quad (3.27)$$

where D_{ss} is the diffusion coefficient in the propagation direction of the discrete wave component, D_{nn} is the diffusion coefficient along the crest of the discrete wave component and T_s is the time elapsed since the generation of the swell. In the present fractional step method the diffusion can be added as a separate step

$$\frac{\partial \mathcal{N}}{\partial t} = \frac{\partial}{\partial x}\left[D_{xx}\frac{\partial \mathcal{N}}{\partial x}\right] + \frac{\partial}{\partial y}\left[D_{yy}\frac{\partial \mathcal{N}}{\partial y}\right] + 2D_{xy}\frac{\partial^2 \mathcal{N}}{\partial x \partial y}\,. \quad (3.28)$$

This equation is incorporated with two simplifications, the justification of which is discussed in Tolman (1995). First, the swell 'age' T_s is kept constant throughout the model (defined by the user, no default value available). Secondly, the diffusion coefficients D_{ss} and D_{nn} are calculated assuming deep water

$$D_{ss} = \left(\left(X_\sigma - 1\right) \frac{\sigma_m}{2k_m} \right)^2 \frac{T_s}{12} \,, \tag{3.29}$$

$$D_{nn} = \left(\frac{\sigma_m}{2k_m} \Delta\theta \right)^2 \frac{T_s}{12} \,, \tag{3.30}$$

where X_σ is defined as in Eq. (3.1). With these two assumptions, the diffusion tensor becomes constant throughout the spatial domain for each separate spectral component.

Equation (3.28) is solved using a forward-time central-space scheme. At the cell interface between points i and $i-1$ in ϕ (x) space, the term in brackets in the first term on the right side of Eq. (3.28) (denoted as $\mathcal{D}_{i,-}$) is estimated as

$$D_{xx} \frac{\partial \mathcal{N}}{\partial x} \approx \mathcal{D}_{i,-} = D_{xx} \left(\frac{\mathcal{N}_i - \mathcal{N}_{i-1}}{\Delta x} \right) \Bigg|_{j,l,m} \,. \tag{3.31}$$

Corresponding values for counters i and $i+1$, and for gradients in λ (y) space again are obtained by rotating indices and increments. If one of the two grid points in located on land, Eq. (3.31) is set to zero. The mixed derivative at the right side of Eq. (3.28) (denoted as $\mathcal{D}_{ij,--}$) is estimated for the grid point i and $i-1$ in x-space and j and $j-1$ in y-space as

$$\mathcal{D}_{ij,--} = D_{xy} \left(\frac{-\mathcal{N}_{i,j} + \mathcal{N}_{i-1,j} + \mathcal{N}_{i,j-1} - \mathcal{N}_{i-1,j-1}}{0.5(\Delta x_j + \Delta x_{j-1})\, \Delta y} \right) \Bigg|_{l,m} \,. \tag{3.32}$$

Note that the increment Δx is a function of y due to the use of the spherical grid. This term is evaluated only if all four grid points considered are sea points, otherwise it is set to zero. Using a forward in time discretization of the first term in Eq. (3.28), and central in space discretizations for the remainder of the first and second term on the right side, the final algorithm becomes

$$\mathcal{N}^{n+1}_{i,j,l,m} = \mathcal{N}^{n}_{i,j,l,m} + \frac{\Delta t}{\Delta x}\left(\mathcal{D}_{i,+} - \mathcal{D}_{i,-}\right) + \frac{\Delta t}{\Delta y}\left(\mathcal{D}_{j,+} - \mathcal{D}_{j,-}\right) + \frac{\Delta t}{4}\left(\mathcal{D}_{ij,--} + \mathcal{D}_{ij,-+} + \mathcal{D}_{ij,+-} + \mathcal{D}_{ij,++}\right) \; . \tag{3.33}$$

Stable solutions are obtained for (e.g., Fletcher, 1988, Part I section 7.1.1)

$$\frac{D_{\max}\,\Delta t}{\min(\Delta x, \Delta y)^2} \leq 0.5 \, , \tag{3.34}$$

where $D_{\max}$ is the maximum value of the diffusion coefficient (typically $D_{\max} = D_{nn}$). Because this stability criterion is a quadratic function of the grid increment, stability can become a serious problem at high latitudes for large scale applications. To avoid that this puts undue constraints on the time step of a model, a corrected swell age $T_{s,c}$ is used

$$T_{s,c} = T_s \min\left\{ 1 \, , \, \left(\frac{\cos(\phi)}{\cos(\phi_c)} \right)^2 \right\} \, , \tag{3.35}$$

where ϕ_c is a cut-off latitude defined by the user.

The above diffusion is needed for swell propagation only, but is not realistic for growing wind seas. In the latter conditions, the ULTIMATE QUICKEST scheme without the dispersion correction is sufficiently smooth to render stable fetch-limited growth curves (Tolman, 1995). To remove minor oscillations, a small isotropic diffusion is used for growing wave components. To assure that this diffusion is small and equivalent for all spectral components, it is calculated from a preset cell Reynolds (or cell Peclet) number $\mathcal{R} = c_g \Delta x D_g^{-1} = 10$, where D_g is the isotropic diffusion for growing components

$$D_g = \frac{c_g \, \min(\Delta x, \Delta y)}{\mathcal{R}} \, , \tag{3.36}$$

The diffusions for swell and for wind seas are combined using a linear combination depending on the nondimensional wind speed or inverse wave age $u_{10} c^{-1} = u_{10} k \sigma^{-1}$ as

$$X_g = \min\left\{ 1 \, , \, \max\left[0 \, , \, 3.3 \left(\frac{k\, u_{10}}{\sigma} \right) - 2.3 \right] \right\} \, , \tag{3.37}$$

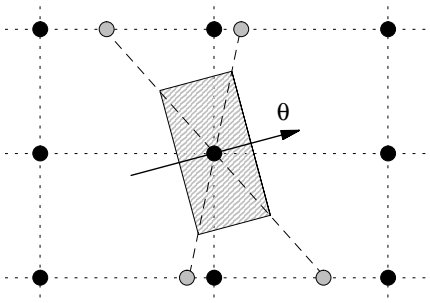

Figure 3.1: Graphical depiction of spatial averaging GSE alleviation technique used here. Solid circles and dotted lines represent the spatial grid. Hatched area represent averaging area to be considered. Corner point values are obtained from the central grid point and the gray points. The latter values are obtained by interpolation from adjacent grid points (from Tolman, 2002a).

$$D_{ss} = X_g D_g + (1 - X_g) D_{ss,p} \,, \tag{3.38}$$

$$D_{nn} = X_g D_g + (1 - X_g) D_{nn,p} \,, \tag{3.39}$$

where the suffix p denotes propagation diffusions as defined in Eqs. (3.29) and (3.30). The constants in Eqs (3.36) and (3.37) are preset in the model.

The major drawback of the above GSE alleviation method is its potential impact on model economy as discussed in relation to Eq. (3.34) and in Tolman (2001, 2002a). For this reason, two additional GSE alleviation methods have been developed for WAVEWATCH III ™.

The first of these two methods, which represents the default for WAVEWATCH III ™, replaces the additional diffusion step (3.28) with a separate fractional step in which direct averaging of the field of energy densities for a given spectral component is considered. The area around each grid point over which the averaging is performed extends in the propagation (**s**) and normal (**n**) directions as

$$\pm\gamma_{a,s}\, \Delta c_g\, \Delta t\, \mathbf{s} \quad , \pm\gamma_{a,n}\, c_g \Delta\theta\, \Delta t\, \mathbf{n} \quad , \tag{3.40}$$

where $\gamma_{a,s}$ and $\gamma_{a,n}$ are tunable constants, the default value of which is set to 1.5. This averaging is graphically depicted in Fig. 3.1. Note that these

values may require some retuning for practical applications, as discussed in Tolman (2002a). Appendix A of the latter paper presents details of the averaging scheme, including conservation considerations. Consistency with the Booij and Holthuijsen (1987) approach furthermore implies that $\gamma_{a,s}$ and $\gamma_{a,n}$ should vary with the spatial grid resolution (see Chawla and Tolman, 2008, Appendix).

Note that this kind of averaging with dominant directions **s** and **n** is similar to the Booij and Holthuijsen (1987) diffusion method, that uses the same main directions. The averaging method, however, never influences the time step, because it is completely separated from the actual propagation. Moreover, if explicit schemes are used with typically $c_g \Delta t / \Delta x < 1$, it is obvious that the averaging over the area as defined in (3.40) will generally require information at directly neighboring spatial grid points only, as in Fig. 3.1. Furthermore, this method does not require high-latitude filtering.

As is illustrated in Tolman (2002a,d), this method gives virtually identical results as the previous method, but does so at slightly lower costs. For high resolution applications, the averaging method may become dramatically more economical.

A third possible GSE alleviation method considers that the advection for a give discrete spectral bin is not unidirectional but divergent (see Tolman, 2002a). An early version of this method was included in model version 1.18. Because this method has not yet been developed to maturity, it is not provided with the present release of WAVEWATCH III ^{TM}.

Finally, the GSE can be alleviated somewhat by assuring that the discrete spectral directions do not coincide with spatial grid lines. This can be achieved by defining the first discrete direction θ_1 as

$$\theta_1 = \alpha_\theta \, \Delta\theta \quad , \tag{3.41}$$

where $-0.5 \leq \alpha_\theta \leq 0.5$ can be defined by the user. Note that setting $\alpha \neq 0$ is beneficial to the first order scheme, but has negligible impact on the third order scheme.

3.3.3 Unresolved obstacles

Even with the original tuning of WAVEWATCH III ^{TM} version 1.15 (Tolman, 2002f), it was clear that unresolved islands groups are a major source of local wave model errors. This was illustrated in some more detail in Tolman (2001,

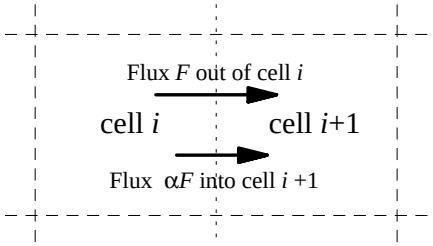

Figure 3.2: Graphical depiction of treatment of unresolved obstacles. Common cell boundary (dotted line) has transparency α. Dashed lines represent other cell boundaries. Numerical flux from left to right.

Fig. 3), and Tolman et al. (2002, Fig. 8). In WAVEWATCH III$^{\rm TM}$, a methodology from the SWAN model (Booij et al., 1999; Holthuijsen et al., 2001) was adopted to apply the effects of unresolved obstacles at the cell boundaries of the spatial grid within the numerical scheme. In this approach, the numerical fluxes between cells through their common boundary are suppressed according to the degree of obstruction provided by the unresolved obstacle. In this approach, the numerical propagation scheme of the ULTIMATE QUICKEST scheme of Eq. (3.21) is modified as

$$\mathcal{N}^{n+1}_{i,j,l,m} = \mathcal{N}^{n}_{i,j,l,m} + \frac{\Delta t}{\Delta \phi} \left[\alpha_{i,-} \mathcal{F}_{i,-} - \alpha_{i,+} \mathcal{F}_{i,+} \right] \; . \tag{3.42}$$

where $\alpha_{i,-}$ and $\alpha_{i,+}$ are 'transparencies' of the corresponding cell boundaries, ranging from 0 (closed boundary) to 1 (no obstructions). For outflow boundaries, transparencies by definition are 1, otherwise energy will artificially accumulate in cells. For inflow boundaries, transparencies less than 1 result in elimination of obstructed energy at the cell boundary. This approach is graphically depicted in Fig. 3.2. Note that a similar approach is easily adopted in the first order scheme (3.9). Note, furthermore, that an alternate obstruction approach with obstructions as a function of the spectral direction θ has been used by Hardy and Young (1996) and Hardy et al. (2000).

Two methods for defining the obstructions are available in the model. The first defines the obstructions directly at the grid boundary. This requires the generation of staggered depth-transparency grids. The second allows the

user to define depths and transparencies at the same grid. In this case, the transparency at the inflow boundary becomes $0.5(1 + \alpha_i)$, and the outflow transparency by definition is 1. To complete the total transparency α_i, the next cell in the flow direction will have an inflow transparency $2\alpha_i/(1 + \alpha_i)$. If consecutive cells are partially obstructed, the product of individual transparencies is applied.

This approach can also be used to continuously model the effects of ice coverage on wave propagation. This will be discussed in section 3.7. Details of the sub-grid treatment of islands and ice can be found in Tolman (2003c). A study of impacts of this approach in large scale wave models is presented in Tolman (2002d, 2003c).

The default setting of WAVEWATCH III ^{TM} is not to include sub-grid modeling of obstacles. Generating obstruction grids can be labor intensive. For this reason, an automated approach for generating bottom and obstruction grids was developed by Chawla and Tolman (2007, 2008). Note that this option does not involve compile-level choices, but is entirely controlled from the grid preprocessor (see following chapter).

3.4 Intra-spectral propagation

The third step of the numerical algorithm considers refraction and residual (current-induced) wavenumber shifts. For both the spherical and Cartesian grid, the equation to be solved in this step becomes

$$\frac{\partial N}{\partial t} + \frac{\partial}{\partial k}\,\dot{k}_g N + \frac{\partial}{\partial \theta}\,\dot{\theta}_g N = 0\,, \tag{3.43}$$

$$\dot{k}_g = \frac{\partial \sigma}{\partial d}\frac{\mathbf{U}\cdot\nabla_x d}{c_g} - \mathbf{k}\cdot\frac{\partial \mathbf{U}}{\partial s}\,. \tag{3.44}$$

where $\dot{k}_g$ is the wavenumber velocity relative to the grid, and $\dot{\theta}_g$ is given by (2.15) and (2.11). This equation does not require boundary conditions in θ-space, as the model by definition uses the full (closed) directional space. In k-space, however, boundary conditions are required. At low wavenumbers, it is assumed that no wave action exists outside the discrete domain. It is therefore assumed that no action enters the model at the discrete low-wavenumber boundary. At the high-wavenumber boundary, transport across

the discrete boundary is calculated assuming a parametric spectral shape as given by Eq. (2.18). The derivatives of the depth as needed in the evaluation of $\dot{\theta}$ are mostly determined using central differences. For points next to land, however, one-sided differences using sea points only are used.

Propagation in θ-space can cause practical problems in an explicit numerical scheme, as the refraction velocity can become extreme for long waves in extremely shallow water. To avoid the need of extremely small time steps due to refraction, the propagation velocity in θ-space (2.11) is filtered with respect to the depth refraction term

$$\dot{\theta} = X_{rd}(\lambda, \phi, k)\dot{\theta}_d + \dot{\theta}_c \,, \tag{3.45}$$

where the indices d and c refer to the depth and current related fraction of the refraction velocity in (2.11). The filter factor X_{rd} is calculated for every wavenumber and location separately, and is determined so that the CFL number for propagation in θ-space due to the *depth* refraction term cannot exceed a pre-set (user defined) value (default 0.7). This corresponds to a reduction of the bottom slope for some low frequency wave components. The effected components are expected to carry little energy because they are in extremely shallow water. Long wave components carrying significant energy are usually traveling toward the coast, where their energy is dissipated anyway.

As with the propagation in physical space, a first order and an ULTIMATE QUICKEST scheme are available. In the first order scheme the fluxes in θ- and k-space are calculated Cf. Eqs. (3.6) through (3.8) (replacing $\mathcal{N}$ with N and rotating the appropriate counters). The complete first order scheme becomes

$$N^{n+1}_{i,j,l,m} = N^{n}_{i,j,l,m} + \frac{\Delta t}{\Delta\theta}\left[\mathcal{F}_{l,-} - \mathcal{F}_{l,+}\right] + \frac{\Delta t}{\Delta k_m}\left[\mathcal{F}_{m,-} - \mathcal{F}_{m,+}\right] \,, \tag{3.46}$$

where $\Delta\phi$ is the directional increment, and Δk_m is the (local) wavenumber increment. The low-wavenumber boundary conditions is applied by taking $\mathcal{F}_{m,-} = 0$ for $m = 1$, and the high wavenumber boundary condition is calculated using the parametric approximation (2.18) for N, extending the discrete grid by one grid point to high wavenumbers.

The ULTIMATE QUICKEST scheme for the θ-space is implemented similar to the scheme for physical space, with the exception that the closed direction space does not require boundary conditions. The variable grid spacing in k-space requires some modifications to the scheme as outlined by (Leonard, 1979, Appendix). Equations (3.10) through (3.14) then become

$$\mathcal{F}_{m,-} = \left[\dot{k}_{g,b}\, N_b \right]_{i,j,l}^{n} , \tag{3.47}$$

$$\dot{k}_{g,b} = 0.5 \left(\dot{k}_{g,m-1} + \dot{k}_{g,m} \right) , \tag{3.48}$$

$$N_b = \frac{1}{2} \left[(1+C)N_{i-1} + (1-C)N_i \right] - \frac{1-C^2}{6}\, \mathcal{CU}\, \Delta k_{m-1/2}^2, \tag{3.49}$$

$$\mathcal{CU} = \begin{cases} \frac{1}{\Delta k_{m-1}} \left[\frac{N_m - N_{m-1}}{\Delta k_{m-1/2}} - \frac{N_{m-1} - N_{m-2}}{\Delta k_{m,-3/2}} \right] & \text{for} \quad \dot{k}_b \geq 0 \\ \frac{1}{\Delta k_m} \left[\frac{N_{m+1} - N_m}{\Delta k_{m+1/2}} - \frac{N_m - N_{m-1}}{\Delta k_{m-1/2}} \right] & \text{for} \quad \dot{k}_b < 0 \end{cases} , \tag{3.50}$$

$$C = \frac{\dot{k}_{g,b}\, \Delta t}{\Delta k_{m-1/2}} , \tag{3.51}$$

where Δk_m is the discrete band or cell width at grid point m, and where $\Delta k_{m-1/2}$ is the distance between grid points with counters m and $m-1$. The ULTIMATE limiter can be applied as in Eqs. (3.15) through (3.18), if the CFL number of Eq. (3.51) is used. At the low- and high-wavenumber boundaries the fluxes again are estimated using a first-order upwind approach, with boundary conditions as above defined for the first-order scheme. The final scheme in k-space becomes

$$N_{i,j,l,m}^{n+1} = N_{i,j,l,m}^{n} + \frac{\Delta t}{\Delta k_m} \left[\mathcal{F}_{m,-} - \mathcal{F}_{m,+} \right] , \tag{3.52}$$

3.5 Source terms

Finally, the source terms are accounted for by solving

$$\frac{\partial N}{\partial t} = \mathcal{S} . \tag{3.53}$$

As in WAM, a semi-implicit integration scheme is used. In this scheme the discrete change of action density ΔN becomes (WAMDIG, 1988)

$$\Delta N(k,\theta) = \frac{\mathcal{S}(k,\theta)}{1-\epsilon D(k,\theta)\Delta t}\,, \tag{3.54}$$

where D represents the diagonal terms of the derivative of $\mathcal{S}$ with respect to N (WAMDIG, 1988, Eqs. 4.1 through 4.10), and where ϵ defines the offset of the scheme. Originally, $\epsilon = 0.5$ was implemented to obtain a second order accurate scheme. Presently, $\epsilon = 1$ is used as it is more appropriate for the large time steps in the equilibrium range of the spectrum (Hargreaves and Annan, 1998, 2001), and as it result in much smoother integration of the spectrum. The change of ϵ has little impact on mean wave parameters, but makes the dynamical time stepping as described below more economical.

The semi-implicit scheme is applied in the framework of a dynamic time-stepping scheme (Tolman, 1992). In this scheme, integration over the global time step Δt_g can be performed in several dynamic time steps Δt_d, depending on the net source term $\mathcal{S}$, a maximum change of action density ΔN_m and the remaining time in the interval Δt_g. For the n^{th} dynamic time step in the integration over the interval Δt_g, Δt_d^n is calculated in three steps as

$$\Delta t_d^n = \min_{f<f_{hf}} \left[\frac{\Delta N_m}{|\mathcal{S}|} \left(1 + \epsilon D \frac{\Delta N_m}{|\mathcal{S}|} \right)^{-1} \right]\,, \tag{3.55}$$

$$\Delta t_d^n = \max\,\left[\,\Delta t_d^n\,,\,\Delta t_{d,\min}\right]\,, \tag{3.56}$$

$$\Delta t_d^n = \min\,\left[\,\Delta t_d^n\,,\,\Delta t_g - \sum_{i=1}^{n-1} \Delta t_d^i\,\right]\,, \tag{3.57}$$

where $\Delta t_{\min}$ is a user-defined minimum time step, which is added to avoid excessively small time steps. The corresponding new spectrum N^n becomes

$$N^n = \max\,\left[\,0\,,\,N^{n-1} + \left(\frac{\mathcal{S}\Delta t_d}{1-\epsilon D\Delta t_d}\right)\,\right]\,. \tag{3.58}$$

The maximum change of action density ΔN_m is determined from a parametric change of action density ΔN_p and a filtered relative change ΔN_r

$$\Delta N_m(k,\theta) = \min\,\left[\,\Delta N_p(k,\theta)\,,\,\Delta N_r(k,\theta)\,\right]\,, \tag{3.59}$$

$$\Delta N_p(k,\theta) = X_p\,\frac{\alpha}{\pi}\,\frac{(2\pi)^4}{g^2}\,\frac{1}{\sigma k^3}\,, \tag{3.60}$$

	X_p	X_r	X_f	$\Delta t_{d,\min}$
WAM equivalent	$\frac{\pi}{24}10^{-3}\Delta t$	$\infty(\geq 1)$	–	Δt_g
suggested	0.1-0.2	0.1-0.2	0.05	$\approx 0.1\Delta t_g$
default setting	0.15	0.10	0.05	–

Table 3.1: User-defined parameters in the source term integration scheme

$$\Delta N_r(k,\theta) = X_r \ \max\ [\ N(k,\theta)\ ,\ N_f\ ]\ , \tag{3.61}$$

$$N_f = \max\ \left[\ \Delta N_p(k_{\max},\theta)\ ,\ X_f\ \max_{\forall k,\theta}\{N(k,\theta)\}\ \right]\ , \tag{3.62}$$

where X_p, X_r and X_f are user-defined constants (see Table 3.1), α is a PM energy level (set to $\alpha = 0.62\,10^{-4}$) and $k_{\max}$ is the maximum discrete wavenumber. The parametric spectral shape in (3.60) corresponds in deep water to the well-known high-frequency shape of the one-dimensional frequency spectrum $F(f) \propto f^{-5}$. The link between the filter level and the maximum parametric change in (3.62) is used to assure that the dynamic time step remains reasonably large in cases with extremely small wave energies. A final safeguard for stability of integration is provided by limiting the discrete change of action density to the maximum parametric change (3.60) in conditions where Eq. (3.56) dictates Δt_d^n. In this case Eq. (3.56) becomes a limiter as in the WAM model. Impacts of limiters are discussed in detail in for instance Hersbach and Janssen (1999, 2001), Hargreaves and Annan (2001) and Tolman (2002c).

The dynamic time step is calculated for each grid point separately, adding additional computational effort only for grid points in which the spectrum is subject to rapid change. The source terms are re-calculated for every dynamic time step.

It is possible to compile WAVEWATCH III ^{TM} without using a linear growth term. In such a case, waves can only grow if some energy is present in the spectrum. In small-scale applications with persistent low wind speeds, wave energy might disappear completely from part of the model. To assure that wave growth can occur when the wind increases, a so-called seeding option is available in WAVEWATCH III ^{TM} (selected during compilation). If the seeding option is selected, the energy level at the seeding frequency

$\sigma_{\rm seed} = \min(\sigma_{\rm max}, 2\pi f_{hf})$ is required to at least contain a minimum action density

$$N_{\rm min}(k_{seed},\theta) = 6.25\,10^{-4}\frac{1}{k_{\rm seed}^3\,\sigma_{\rm seed}}\max\left[\,0.\,,\,\cos^2(\theta-\theta_w)\right] \min\left[\,1\,,\,\max\left(\,0\,,\,\frac{|u_{10}|}{X_{\rm seed}g\sigma_{\rm seed}^{-1}}-1\,\right)\right]\,, \quad (3.63)$$

where $g\sigma_{\rm seed}^{-1}$ approximates the equilibrium wind speed for the highest discrete spectral frequency. This minimum action distribution is aligned with the wind direction, goes to zero for low wind speeds, and is proportional to the integration limiter (3.60) for large wind speeds. $X_{\rm seed} \geq 1$ is a user-defined parameter to shift seeding to higher frequencies. Seeding starts if the wind speed reaches $X_{\rm seed}$ times the equilibrium wind speed for the highest discrete frequency, and reaches its full strength for twice as high wind speeds. The default model settings include the seeding algorithm, with $X_{\rm seed} = 1$.

In model version 3.11, surf-zone physics parameterizations have been introduced. Such physics, particularly depth-induced breaking, operate on much smaller time scales than deep water and limited depth physics outside the surf zone. To assure reasonable behavior for larger time steps, an additional optional limiter has been adopted from the SWAN model, similar to the Miche style maximum wave height in the depth limited wave breaking source term of Eq. (2.104). In this limiter, the maximum wave energy E_m is computed as

$$E_m = \frac{1}{16}[\gamma_{lim}\,\bar{k}\,\tanh(\bar{k}d)]^2 \quad , \quad (3.64)$$

where γ_{lim} is a factor comparable to γ_M in Eq. (2.104), with the caveat that γ_M is representative for an individual wave, whereas γ_{lim} is representative for the significant wave height. If the total spectral energy E is larger than the maximum energy E_m, the limiter is applied by simply rescaling the spectrum by the factor E/E_m, loosely following the argumentation from Eldeberky and Battjes (1996) ad used in section 2.3.9.

This limiter can be switch on or off in the compilation of the model, and γ_{lim} can be adjusted by the user (default $\gamma_{lim} = 0.75$). Note that this limiter should be used as a 'safety valve' only, and hence that it should be less strict than the breaking criterion in the surf-breaking source term, if this source term is modeled explicitly.

3.6 Winds and currents

Model input mainly consists of wind and current fields. Within the model, winds and currents are updated at every time step Δt_g and represent values at the end of the time step considered. Several interpolation methods are available (selected during compilation). By default, the interpolation in time consists of a linear interpolation of the velocity and the direction (turning the wind or current over the smallest angle). The wind speed or current velocity can optionally be corrected to (approximately) conserve the energy instead of the wind velocity. The corresponding correction factor X_u is calculated as

$$X_u = \max \left[1.25 \, , \, \frac{u_{10,rms}}{u_{10,l}} \right] \, , \tag{3.65}$$

where $u_{10,l}$ is the linearly interpolated velocity and $u_{10,rms}$ is the rms interpolated velocity. Finally, winds can optionally be kept constant and changed discontinuously (option not available for current).

Note that the auxiliary programs of WAVEWATCH III$^{\text{TM}}$ include a program to pre-process input fields (see section 4.4.4). This program transfers gridded fields to the grid of the wave model. For winds and currents this program utilizes a bilinear interpolation of vector components. This interpolation can be corrected to (approximately) conserve the velocity or the energy of the wind or the current by utilizing a correction factor similar to Eq. (3.65).

3.7 Ice coverage

Ice covered sea is considered as 'land' in WAVEWATCH III$^{\text{TM}}$, assuming zero wave energy and boundary conditions at ice edges are identical to boundary conditions at shore lines. Grid points are taken out of the calculation if the ice concentration becomes larger than a user-defined concentration. If the ice concentration drops below its critical value, the corresponding grid point is 're-activated'. The spectrum is then initialized with a PM spectrum based on the local wind direction with a peak frequency corresponding to the second-highest discrete frequency in the grid. A small spectrum is used to assure that spectra are realistic, even for shallow coastal points.

The above discontinuous ice treatment represents the default model setting in WAVEWATCH III$^{\text{TM}}$. In the framework of the modeling of unresolved obstacles as discussed in section 3.3.3, a continuous method is also available, as given by Tolman (2003c). In this method, a user-defined critical ice concentration at which obstruction begins ($\epsilon_{c,0}$) and is complete ($\epsilon_{c,n}$) are given (defaults are $\epsilon_{c,0} = \epsilon_{c,n} = 0.5$, i.e., discontinuous treatment of ice). From these critical concentrations, corresponding decay length scales are calculated as,

$$l_0 = \epsilon_{c,0} \min(\Delta x, \Delta y) \quad . \tag{3.66}$$

$$l_n = \epsilon_{c,n} \min(\Delta x, \Delta y) \quad . \tag{3.67}$$

from which cell transparencies in x and y (α_x and α_y, respectively) are calculated as

$$\alpha_x = \begin{cases} 1 & \text{for} \quad \epsilon\Delta x < l_0 \\ 0 & \text{for} \quad \epsilon\Delta x > l_n \\ \frac{l_n - \epsilon\Delta x}{l_n - l_0} & \text{otherwise} \end{cases} \quad , \quad \alpha_y = \begin{cases} 1 & \text{for} \quad \epsilon\Delta y < l_0 \\ 0 & \text{for} \quad \epsilon\Delta y > l_n \\ \frac{l_n - \epsilon\Delta y}{l_n - l_0} & \text{otherwise} \end{cases} \quad . \tag{3.68}$$

Details of this model can be found in Tolman (2003c).

Updating of the ice map within the model takes place at the discrete model time approximately half way in between the valid times of the old and new ice maps. The map will not be updated, if the time stamps of both ice fields are identical.

3.8 Spectral partitioning (B. Tracy)

Figure 3.3 shows an example surface plot of an energy density spectrum at one grid point at a specific time. The amount of energy density at each frequency-direction intersection is shown by this surface. The surface is divided into shaded areas or partitions representing energy from sub-peaks within the spectrum. Figure 3.3 shows four spectral partitions, an area of windsea and three swell trains. The total energy represented by this spectrum can be defined by bulk parameters, such as the significant wave height H_s. The shaded areas, called partitions of the spectrum, show spectral subfeatures that give more information about this grid point's energy situation.

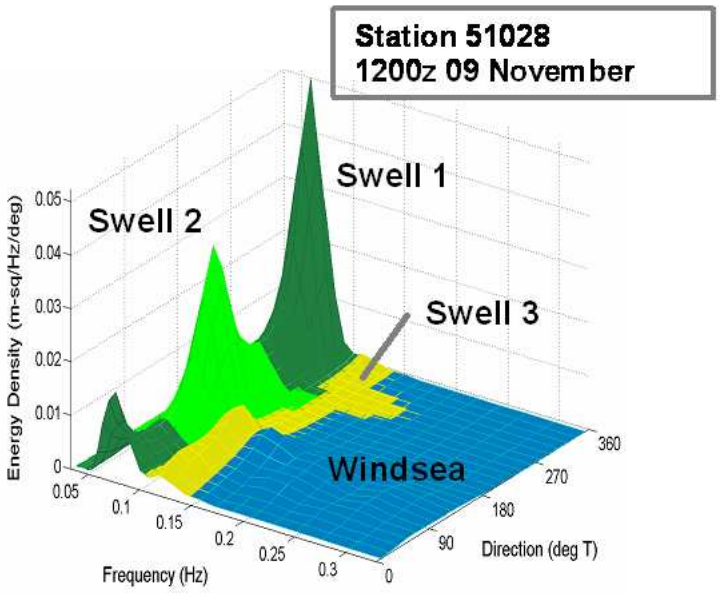

Figure 3.3: Surface plot of an energy density spectrum showing spectral partitions for windsea and three swell trains. This is a snapshot of hindcasted conditions at Christmas Island (NOAA buoy 51028) at 12:00 UTC on November 9, 2000..

WAVEWATCH III ^{TM} has point and field output options available to provide quantitative descriptions of these individual spectral partition such as partition wave height, peak period of partition (parabolic fit), peak wavelength of partition, mean direction of partition, wind-sea fraction of partition (W) using Eq. (2.124), and the number of partitions. In the field output, these parameters correspond to output fields 15 through 22 and can be found in section 2.4.

Since the two-dimensional spectrum in Fig. 3.3 looks like a topological surface, it is logical to apply an image processing partitioning algorithm that treats the spectral surface like a topographical surface. The partitioning shown in Fig. 3.3 is based on a digital image processing watershed algorithm (Vincent and Soille, 1991) first prototyped by Hanson and Jensen (2004) for

the analysis of ocean wave data. The continental divide where everything to the east goes into the Atlantic Ocean and everything to the west goes into the Pacific Ocean is a typical example of a watershed line. The oceans represent minima that determine the watershed line. If the spectral surface is inverted, the spectral peaks become catchments and watershed lines or partition boundaries can be determined using the Vincent and Soille (1991) algorithm. Calculation of parameters for each spectral partition can then be accomplished and wave system analysis as described in Hanson and Phillips (2001) can be applied. Hanson and Jensen (2004) and Hanson et al. (2006) used a MATLAB code to apply the Vincent and Soille (1991) algorithm[6]. This code has been transformed to an efficient FORTRAN routine for use in the version 3.11 of WAVEWATCH III ^{TM}. Coding follows the Vincent and Soille (1991) paper but incorporates an efficient sort routine (O(n)) discussed in Tracy et al. (2006).

3.9 Nesting

Traditionally, wave models only consider one-way nesting, with boundary data from low resolution grids being provided to high resolution grids. This approach has always been available in WAVEWATCH III^{TM}, and is discussed in section 3.9.1. In model version 3.14, a multi-grid wave model driver was introduced, considering full two-way nesting between grids. This approach is discussed in section 3.9.2.

3.9.1 Traditional one-way nesting

The conventional wave model program ww3_shel considers a single wave model grid. This program includes options to transfer boundary conditions from large-scale runs to small-scale runs. Each run can simultaneous accept one data set with boundary conditions, and generate up to 9 data sets with boundary conditions. To assure conservation of wave energy with incompatible depths and currents, the boundary data consists of energy spectra

[6] Now available as XWaves from http://www.WaveForceTechnologies.com, replacing the previous APL WAVES package

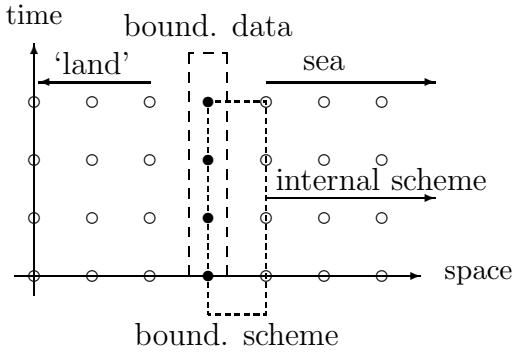

Figure 3.4: Traditional one-way nesting approach as used in ww3_shel. One-dimensional representation in space and time, symbols represent grid points.

$F(\sigma, \theta)$. The data file consists of spectra at grid points of the generating run, and information needed to interpolate spectra at the requested boundary points. The size of the transfer files is thus minimized if the input points for a small-scale run are located on grid lines in the large-scale run. When used as input, the spectra are interpolated in space and time for every global time step Δt_g, using a linear interpolation of spectral components.

The numerical approach for including boundary data in a wave model is illustrated in Fig. 3.4. Active boundary points are assigned in the grid to separate sea points from land points or from otherwise deactivated grid points. Between the active boundary points and sea points, a local boundary scheme is applied (typically first order). In the internal sea points of the model, the selected propagation scheme is used.

Practical aspect of the conventional one-way nesting approach are discussed in more detail in Appendix C.

3.9.2 Two-way nesting

Model version 3.14 introduces the multi-grid or mosaic approach to wave modeling with the introduction of the wave model program ww3_multi (Tolman, 2006, 2007, 2008). In this program, an arbitrary number of grids with arbitrary resolutions is considered, with data exchange between grids at each

relevant model time step. The grids are given a rank number, where lower rank corresponds to lower resolution, and equal rank corresponds to similar resolution (but not necessarily equal resolution). Three types of data transfer between grids are considered. These are

- Transfer of data from lower to higher rank grids.
- Transfer of data from higher to lower rank grids.
- Transfer of data between grids with equal rank.

Data transfer from lower to higher ranked grids is accomplished by providing boundary data to the higher ranked grid, as in the traditional one-way nesting approach described in the previous section and in Fig. 3.4.

When this approach is combined with data transfer from higher to lower rank, a full two way nesting approach is established. In ww3_multi the data at the lower ranked grids is reconciled with the data at the higher ranked grids after the higher ranked grids have 'caught up' in time with the lower ranked grids. Considering that the resolution of the lower ranked grid by definition is lower that the resolution of the higher ranked grid, a natural way to estimate the wave energy in the lower ranked grid $E_{l,i}$ from energy in the higher ranked grid $E_{h,j}$ is

$$E_{l,i} = \sum w_{i,j} W_{h,j} \quad , \tag{3.69}$$

where i and j are grid counters in the two grids, and where $w_{i,j}$ are averaging weights. The weights can be defined consistent with conservation of wave energy as the surface of the grid box j in the higher ranked grid that covers the grid box i in the lower ranked grid, normalized with the surface of the lower ranked grid box i. This is illustrated in Fig. 3.5. To avoid circular reconciliation, grid points in the lower ranked grid that contribute to the boundary data in the higher ranked grid are not updated in this manner.

Overlapping grids with similar rank cannot use the above two-way nesting technique to consistently exchanger data. Instead, all such grids are propagated one time step, after which the grids are reconciled as is illustrated in Fig. 3.6 For grid 1 ($\circ$ in Fig. 3.6) two areas can be distinguished. In area C, the influence of the boundary has propagated into the grid since the last reconciliation. The actual depth of penetration depends on the stencil width of the numerical scheme, and the number of propagation time steps. In areas A

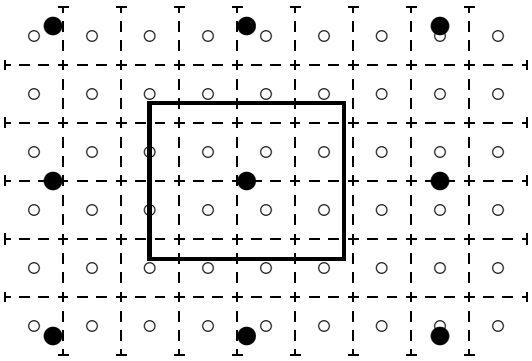

Figure 3.5: Concept for reconciling lower ranked grid with higher ranked grid in two-way nesting approach. ∘ and hashed lines represent the higher ranked grid points and grid boxes, respectively, • and solid lines represent lower ranked grid and central grid box.

---

and B, information from the boundary has not yet penetrated, and this area can be considered as the 'interior' of grid 1. Similarly, area A represents the boundary penetration depth for grid 2 (• in Fig. 3.6) whereas B and C represent the interior of grid 2. A simple and consistent reconciliation between grid 1 and 2 uses data from grid 1 exclusively in area A (interpolating data from grid 1 to grid points in grid 2 as necessary), and uses data from grid 2 exclusively in area C. In area B, where interior parts of both grids overlap, a consistent solution can be found by using weighted averages from both grids. Note that this approach is easily extended to multiple overlapping grids.

Note that for explicit numerical propagation schemes and overlapping grids with identical resolution and coinciding grid points, solutions for overlapping grids and the compatible single grid can be identical, as long as the overlap areas are sufficiently wide.

The two-way nesting techniques in ww3_multi are largely automated. Each grid is prepared individually, with its own preferred time stepping information. Locations where each grid expects to get boundary data are marked as in the one-way nesting approach. All other bookkeeping needed to implement the two-way nesting techniques are automated, although some iterations may be needed to assure that all input boundary points defined in each grid can

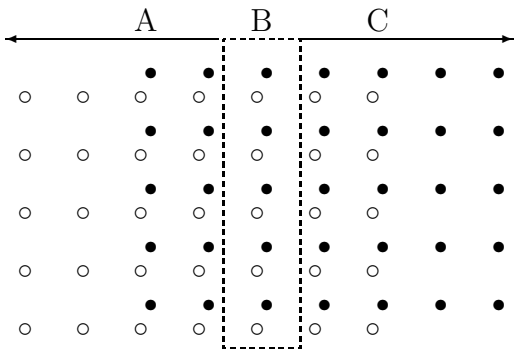

Figure 3.6: Concept for reconciling grids with identical rank and therefore similar resolution. ∘ represents points of grid 1, • represents grid 2.

---

be provided with boundary data from other grids in the multi-grid application. Alternatively, each grid can obtain data from an external data file as in the traditional nesting approach. In the present implementation, each grid has to obtain all boundary data from a single file, of from other grids in the multi-grid application, but cannot receive data from file and grids simultaneously. Details on the management algorithm developed to run all grid simultaneously can be found in Tolman (2007, section 3.4) or Tolman (2008), and will not be reproduced here.

Note that the grids used in ww3_multi do not need to have the same spectral discretization. Spectra are converted on the fly in ww3_multi. Details on the numerical techniques used for this approach can be found in Tolman (2007, section 3.5.5).

Grid generation for multiple grids in such an approach can be cumbersome, and consistency between grids is required for consistent model results. For this reason automated grid generation utilities have been developed by Chawla and Tolman (2007, 2008).

This page is intentionally left blank.

4 Running the wave model

4.1 Program design

The core of WAVEWATCH III ™ is the wave model subroutine. The wave model routine can be called by either a stand-alone program shell or any other program that requires dynamically updated wave data. Two such programs are provided with the WAVEWATCH III ™ release. Auxiliary programs include a grid preprocessor, a program to generate artificial initial conditions, a generic program shell (and a corresponding input pre-processor) and output post-processors. In the discussion of the model below, file names will be identified by the file type font, the contents of a file by the `code` type font and FORTRAN program elements by the FORTRAN type font.

The main wave model routine is W3WAVE. Data files are identified with the file extension .ww3, except in the multi-grid wave model ww3_multi, where the file extension identifies an individual grid. For simplicity, the file extension .ww3 will be used throughout this chapter. A relational diagram including the basic data flow is presented in figure 4.1.

The grid preprocessor writes a model definition file mod_def.ww3 with bottom and obstruction information and parameter values defining the physical and numerical approaches. The wave model requires initial conditions, consisting of a restart file restart.ww3, written by either the wave model itself, or by the initial conditions program. If this file is not available, the wave model will be initialized automatically, depending on the ability of the model to start from calm conditions. If linear growth or spectral seeding is switch on, the model will start from calm conditions ($H_s = 0$), otherwise the initial conditions will consist of a parametric fetch-limited spectrum based on the initial wind field (see the corresponding option in the initial conditions program). The wave model routine (W3WAVE) optionally generates up to 9 restart files restart*n*.ww3, where *n* represents a single digit integer number. The wave model also optionally reads boundary conditions from the file nest.ww3 and generates boundary conditions for consecutive runs in nest*n*.ww3. The model furthermore dumps raw data to the output files out_grd.ww3 , out_pnt.ww3, track_o.ww3 and partition.ww3 (gridded mean wave parameters, spectra at locations, spectra along tracks, and partitioned wave data, respectively). The tracks along which spectra are to be presented is defined in the file

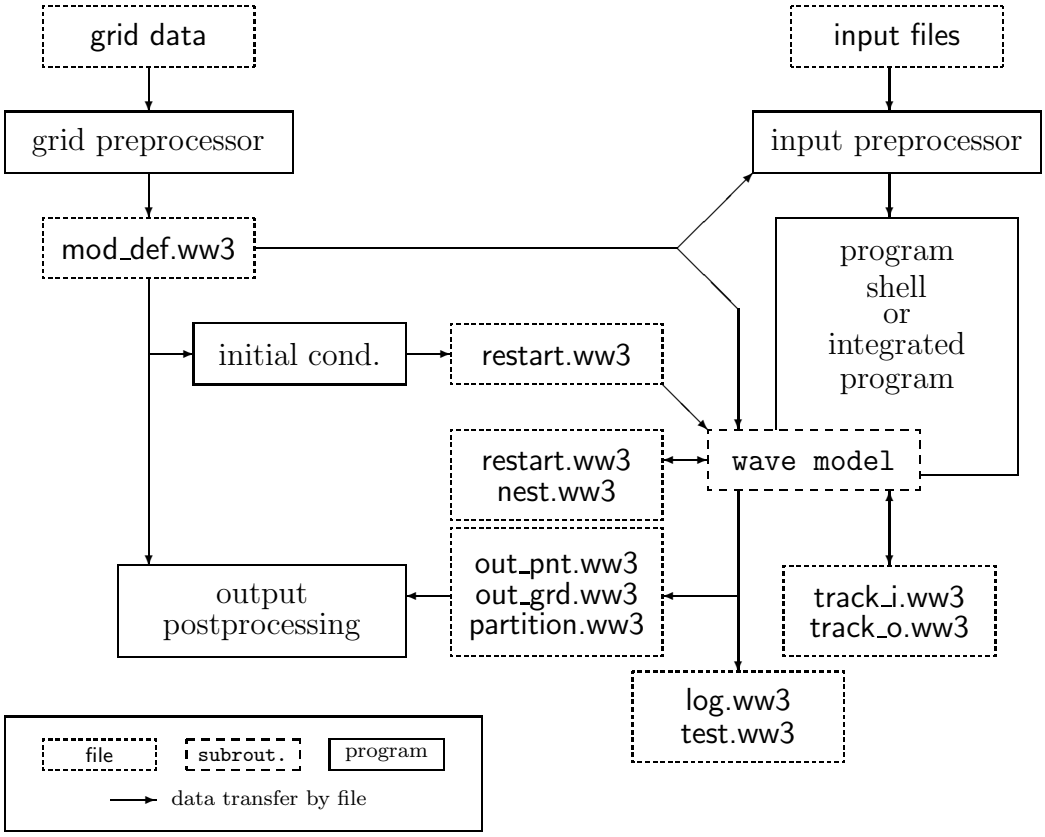

Figure 4.1: Basic program elements and data flow

track_i.ww3. Note that the wave model does not write to standard output, because this would be inconvenient if WAVEWATCH III TM is part of an integrated model. Instead, it maintains its own log file log.ww3 and optionally a test output files test.ww3 for a shared memory version of the model, or test*nnn*.ww3 for distributed memory versions, where *nnn* is the processor number starting with 1. Finally, six output post-processors are available (binary post-processing of raw gridded fields, point output and track output files; GRIB packing of gridded data; post-processing for later GrADS graphical processing of gridded and spectral data). A more detailed description of all program elements and there input files is given below. Note that the source codes of each routine are fully documented. This documentation is an additional source of information about WAVEWATCH III TM.

Files specific to WAVEWATCH III TM are opened by name within the

program. The unit numbers, however, have to be defined by the user[7], guaranteeing the largest possible flexibility for implementation in integrated models.

Next to the wave model subroutine, an initialization routine and an interface routine for data assimilation are provided. The latter routine is intended to be run side by side with the wave model routine. The routine includes a generic interface that provides all necessary model components to perform full spectral data assimilation. This routine is integrated into the generic wave model shell, which is set up to perform time step managements for a wave model with or without data assimilation. The shell also provides a simple yet flexible way to provide the data assimilation scheme with various types of data. Data assimilation has not yet been included in the multi-grid wave model shell.

4.2 The wave model routines

As discussed above, the actual wave model is a subroutine. To run the model, a program shell is needed. WAVEWATCH III ^{TM} is provided with a simple stand-alone shell as will be discussed in section 4.4.5, and with a more complex multi-grid model shell as will be discussed in section 4.4.6. The present section concentrates on the wave model subroutines.

The wave model initialization routine W3INIT performs model initialization. This includes setting up part of the I/O system by defining unit numbers, initializing internal time management, processing the model definition file (mod_def.ww3), processing initial conditions (restart.ww3), preparing model output, and calculating grid-dependent parameters. If the model is compiled for an MPI environment, all necessary communication for both calculations and output are determined and initialized (the model uses persistent MPI communication throughout).

The wave model routine W3WAVE can be called any number of times to propagate the wave field in time after the initialization has taken place. After some initial checks, the subroutine interpolates winds and currents, updates ice concentrations and water levels, propagates the wave field, and applies the selected source terms for a number of time steps. The internal time step

[7] Except for ww3_multi.

			input		output	
step	pass	date time	b w l c i d		g p t r b f	
0	1	1968/06/06 00:00:00	F		X X	
8	1	02:00:00			X	
12	1	03:00:00			X	
16	1	04:00:00			X	
24	1	06:00:00	X		X X	
32	2	08:00:00			X	
36	2	09:00:00			L	
40	2	10:00:00			X	
48	2	12:00:00	X	X	L	L

Figure 4.2: Example action table from file log.ww3.

is defined by the interval for which the calculations are to be performed, and by the requested output times. At the end of the calculations, the routine provides the calling program with the requested fields of wave data. A documentation of the interface of W3WAVE can be found in the source code (w3wavemd.ftn).

Apart from the raw data files as described above, the program maintains a log file log.ww3. This file is opened by W3INIT (contained in W3WAVE in w3wavemd.ftn), which writes some self-explanatory header information to this file. Each consecutive call to W3WAVE adds several lines to an 'action table' in this log file as is shown in Fig. 4.2. The column identified as 'step' shows the discrete time step considered. The column identified as 'pass' identifies the sequence number of the call to W3WAVE; i.e., 3 identifies that this action took place in the third call to W3WAVE. The third column shows the ending time of the time step. In the input and output columns the corresponding actions of the model are shown. An `X` identifies that the input has been updated, or that the output has been performed. An `F` indicates a first field read, and an `L` identifies the last output. The seven input columns identify boundary conditions (`b`), wind fields (`w`), water levels (`l`), current fields (`c`), ice concentrations (`i`), and data for assimilation (`d`), respectively. Note that data assimilation takes place at the end of the time step after the

wave routine call. The six output columns identify gridded output (`g`), point output (`p`), output along tracks (`t`), restart files (`r`), boundary data (`b`), and partitioned spectral data `f`, respectively.

4.3 The data assimilation interface

As discussed above, the wave model subroutine is supplemented with a data assimilation interface routine (W3WDAS in w3wdasmd.ftn). This routine is integrated in the stand-alone shell (see section 4.4.5) to provide time step management of a combined wave model / data assimilation scheme. In this a fairly simple approach is assumed where data assimilation is performed at selected times, while the wave model marches forward in time. In the setup of the shell, the data assimilation is performed after the model has reached the target time, but has not yet produced output. After the data assimilation is performed, the wave model routine is called again only to generate output as requested. Thus, the wave model output for a given time will include the effects of data assimilation for that specific target time.

The generic program shell also processes several types of data to be assimilated, and passes it on to the data assimilation interface routine. All data needs to be preprocessed using the wave model input preprocessor (see section 4.4.4), and will be recognized by the generic shell by file name. Presently, up to three different data files can be used. Tentatively, these could be mean wave parameters, one dimensional spectral data, and two dimensional spectral data, respectively. This is, however, not hardwired to the model and in fact needs to be defined by the user.

Presently, no data assimilation packages are available. User supplied data assimilation schemes can be included in the wave model using the interface routine (W3WDAS in w3wdasmd.ftn), the documentation of which should be sufficient for the necessary programming. Details on how to add user supplied software to the WAVEWATCH III ^{TM} compilation system can be found in the following chapter. NCEP is presently working on wave data assimilation techniques, but presently has no plans to distribute wave data assimilation software.

4.4 Auxiliary programs

4.4.1 General concepts

All auxiliary programs presented here with the exception of the track output post-processor read input from a pre-defined input file. The first character on the first line of the input file will be considered to be the comment character, identifying comment lines in the input file. This comment character has to appear on the first position of input lines to be effective. In all examples in the following sections lines starting with '$' therefore only contain comment. The programs furthermore all write formatted output to the standard output unit.

In the following sections, all available auxiliary programs are described using an example input file with all options included (partially as comment). These files are identical to the distributed example input files. The sections furthermore show the name of the executable program, the program name (as appears in the program statement), the source code file and input and output files and there unit numbers (in brackets behind the file name). Input and output files marked with *are optional. The intermediate files mentioned below are all UNFORMATTED, and are not described in detail here. Each file is written and read by a single routine, to which reference is made for additional documentation.

mod_def.ww3	Subroutine W3IOGR (w3iogrmd.ftn).
out_grd.ww3	Subroutine W3IOGO (w3iogomd.ftn).
out_pnt.ww3	Subroutine W3IOPO (w3iopomd.ftn).
track_o.ww3	Subroutine W3IOTR (w3iotrmd.ftn).
restart.ww3	Subroutine W3IORS (w3iorsmd.ftn).
nest.ww3	Subroutine W3IOBC (w3iobcmd.ftn).
partition.ww3	Subroutine W3IOSF (w3iosfmd.ftn).

Preprocessing and compilation of the programs is discussed in the following two chapters. Examples of test runs of the model are provided with the source code.

4.4.2 The grid preprocessor

Program	:	ww3_grid		(W3GRID)
Code	:	ww3_grid.ftn		
Input	:	ww3_grid.inp	(10)	Formatted input file for program.
		'grid file' *	(user)	File with bottom depths.
		'obstr. file' *	(user)	File with sub-grid obstructions.
		'mask file' *	(user)	File with grid mask.
Output	:	standard out	(6)	Formatted output of program.
		mod_def.ww3	(20)	Model definition file in WAVEWATCH III TM format.
		mask.ww3 *	(20)	Land-sea mask file (switch O2a).
Scratch	:	ww3_grid.scratch	(90)	Formatted scratch file.

Note that bottom and obstruction data may be in same file.

start of example input file

```
$ ------------------------------------------------------------------- $
$ WAVEWATCH III Grid preprocessor input file                          $
$ ------------------------------------------------------------------- $
$ Grid name (C*30, in quotes)
$
  'TEST GRID (GULF OF NOWHERE)   '
$
$ Frequency increment factor and first frequency (Hz) ---------------- $
$ number of frequencies (wavenumbers) and directions, relative offset
$ of first direction in terms of the directional increment [-0.5,0.5].
$ In versions 1.18 and 2.22 of the model this value was by definiton 0,
$ it is added to mitigate the GSE for a first order scheme. Note that
$ this factor is IGNORED in the print plots in ww3_outp.
$
   1.1  0.04118  25  24  0.
$
$ Set model flags ---------------------------------------------------- $
$  - FLDRY         Dry run (input/output only, no calculation).
$  - FLCX, FLCY    Activate X and Y component of propagation.
$  - FLCTH, FLCK   Activate direction and wavenumber shifts.
$  - FLSOU         Activate source terms.
$
   F T T T F T
$
$ Set time steps ----------------------------------------------------- $
```

```
$ - Time step information (this information is always read)
$     maximum global time step, maximum CFL time step for x-y and
$     k-theta, minimum source term time step (all in seconds).
$
    900. 950. 900. 300.
$
$ Start of namelist input section ------------------------------------ $
$   Starting with WAVEWATCH III version 2.00, the tunable parameters
$   for source terms, propagation schemes, and numerics are read using
$   namelists. Any namelist found in the folowing sections up to the
$   end-of-section identifier string (see below) is temporarily written
$   to ww3_grid.scratch, and read from there if necessary. Namelists
$   not needed for the given switch settings will be skipped
$   automatically, and the order of the namelists is immaterial.
$   As an example, namelist input to change SWELLF and ZWND in the
$   Tolman and Chalikov input would be
$
$   &SIN2 SWELLF = 0.1, ZWND = 15. /
$
$ Define constants in source terms ---------------------------------- $
$
$ Stresses - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
$   TC 1996 with cap    : Namelist FLX3
$                           CDMAX  : Maximum allowed CD (cap)
$                           CTYPE  : Cap type :
$                                      0: Discontinuous (default).
$                                      1: Hyperbolic tangent.
$
$ Linear input - - - - - - - - - - - - - - - - - - - - - - - - - - - -
$   Cavaleri and M-R    : Namelist SLN1
$                           CLIN   : Proportionality constant.
$                           RFPM   : Factor for fPM in filter.
$                           RFHF   : Factor for fh in filter.
$
$ Exponential input  - - - - - - - - - - - - - - - - - - - - - - - - -
$   WAM-3               : Namelist SIN1
$                           CINP   : Proportionality constant.
$
$   Tolman and Chalikov : Namelist SIN2
$                           ZWND   : Height of wind (m).
$                           SWELLF : swell factor in (n.nn).
$                           STABSH, STABOF, CNEG, CPOS, FNEG :
$                                     c0, ST0, c1, c2 and f1 in . (n.nn)
$                                     through (2.65) for definition of
$                                     effective wind speed (!/STAB2).
```

```
$   WAM4 and variants  : Namelist SIN3
$                           ZWND    : Height of wind (m).
$                           ALPHA0  : minimum value of Charnock coefficient
$                           Z0MAX   : maximum value of air-side roughness z0
$                           BETAMAX : maximum value of wind-wave coupling
$                           SINTHP  : power of cosine in wind input
$                           ZALP    : wave age shift to account for gustiness
$                       TAUWSHELTER : sheltering of short waves to reduce u_star
$                         SWELLFPAR : choice of swell attenuation formulation
$                                     (1: TC 1996, 3: ACC 2008)
$                          SWELLF : swell attenuation factor
$     Extra parameters for SWELLFPAR=3 only
$                   SWELLF2, SWELLF3 : swell attenuation factors
$                           SWELLF4 : Threshold Reynolds number for ACC2008
$                       SWELLF5 : Relative viscous decay below threshold
$                             Z0RAT : roughness for oscil. flow / mean flow
$
$ Nonlinear interactions - - - - - - - - - - - - - - - - - - - - - - - -
$   Discrete I.A.       : Namelist SNL1
$                           LAMBDA : Lambda in source term.
$                           NLPROP : C in sourc term. NOTE : default
$                                    value depends on other source
$                                    terms selected.
$                           KDCONV : Factor before kd in Eq. (n.nn).
$                           KDMIN, SNLCS1, SNLCS2, SNLCS3 :
$                                    Minimum kd, and constants c1-3
$                                    in depth scaling function.
$   Exact interactions  : Namelist SNL2
$                           IQTYPE : Type of depth treatment
$                                     1 : Deep water
$                                     2 : Deep water / WAM scaling
$                                     3 : Shallow water
$                           TAILNL : Parametric tail power.
$                           NDEPTH : Number of depths in for which
$                                    integration space is established.
$                                    Used for IQTYPE = 3 only
$                         Namelist ANL2
$                           DEPTHS : Array with depths for NDEPTH = 3
$
$ Dissipation  - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
$   WAM-3               : Namelist SDS1
$                           CDIS, APM : As in source term.
$
$   Tolman and Chalikov : Namelist SDS2
$                           SDSA0, SDSA1, SDSA2, SDSB0, SDSB1, PHIMIN :
```

```
$                                  Constants a0, a1, a2, b0, b1 and
$                                  PHImin.
$
$   WAM4 and variants   : Namelist SDS3
$                           SDSC1    : WAM4 Cds coeffient
$                           MNMEANP, WNMEANPTAIL : power of wavenumber
$                                    for mean definitions in Sds and tail
$                           SDSDELTA1, SDSDELTA2 : relative weights
$                                    of k and k^2 parts of WAM4 dissipation
$                           SDSLF, SDSHF : coefficient for activation of
$                              WAM4 dissipation for unsaturated (SDSLF) and
$                               saturated (SDSHF) parts of the spectrum
$                           SDSC2    : Saturation dissipation coefficient
$                           SDSC4    : Value of B0=B/Br for wich Sds is zero
$                           SDSBR    : Threshold Br for saturation
$                           SDSP     : power of (B/Br-B0) in Sds
$                           SDSBR2   : Threshold Br2 for the separation of
$                             WAM4 dissipation in saturated and non-saturated
$                           SDSC5 : coefficient for turbulence dissipation
$                           SDSC6 : Weight for the istropic part of Sds_SAT
$                           SDSDTH: Angular half-width for integration of B
$
$
$ Bottom friction  - - - - - - - - - - - - - - - - - - - - - - - - - - -
$   JONSWAP             : Namelist SBT1
$                           GAMMA   : As it says.
$
$
$ Surf breaking  - - - - - - - - - - - - - - - - - - - - - - - - - - - -
$   Battjes and Janssen : Namelist SDB1
$                           BJALFA  :
$                           BJGAM   :
$                           BJFLAG  :
$
$ Propagation schemes ---------------------------------------------- $
$   First order         : Namelist PRO1
$                           CFLTM  : Maximum CFL number for refraction.
$
$   UQ with diffusion   : Namelist PRO2
$                           CFLTM  : Maximum CFL number for refraction.
$                           DTIME  : Swell age (s) in garden sprinkler
$                                    correction. If 0., all diffusion
$                                    switched off. If small non-zero
$                                    (DEFAULT !!!) only wave growth
$                                    diffusion.
```

```
$                          LATMIN : Maximum latitude used in calc. of
$                                   strength of diffusion for prop.
$
$   UQ with averaging   : Namelist PRO3
$                          CFLTM  : Maximum CFL number for refraction.
$                          WDTHCG : Tuning factor propag. direction.
$                          WDTHTH : Tuning factor normal direction.
$
$ Miscellaneous ------------------------------------------------- $
$   Misc. parameters    : Namelist MISC
$                          CICE0  : Ice concentration cut-off.
$                          CICEN  : Ice concentration cut-off.
$                          PMOVE  : Power p in GSE aleviation for
$                                   moving grids in Eq. (D.4).
$                          XSEED  : Xseed in seeding alg. (!/SEED).
$                          FLAGTR : Indicating presence and type of
$                                   subgrid information :
$                                    0 : No subgrid information.
$                                    1 : Transparancies at cell boun-
$                                        daries between grid points.
$                                    2 : Transp. at cell centers.
$                                    3 : Like 1 with cont. ice.
$                                    4 : Like 2 with cont. ice.
$                          XP, XR, XFILT
$                                   Xp, Xr and Xf for the dynamic
$                                   integration scheme.
$                          IHMAX  : Number of discrete levels in part.
$                          HSPMIN : Minimum Hs in partitioning.
$                          WSM    : Wind speed multiplier in part.
$                          WSC    : Cut of wind sea fraction for
$                                   identifying wind sea in part.
$                          FLC    : Flag for combining wind seas in
$                                   partitioning.
$                          FMICHE : Constant in Miche limiter.
$
$ In the 'Out of the box' test setup we run with sub-grid obstacles
$ and with continuous ice treatment.
$
  &MISC CICE0 = 0.25, CICEN = 0.75, FLAGTR = 4 /
  &FLX3 CDMAX = 3.5E-3 , CTYPE = 0 /
$ &SDB1 BJGAM = 1.26, BJFLAG = .FALSE. /
$
$ Mandatory string to identify end of namelist input section.
$
END OF NAMELISTS
```

```
$
$ Define grid ------------------------------------------------------ $
$ Four records containing :
$  1 NX, NY. As the outer grid lines are always defined as land
$    points, the minimum size is 3x3.
$  2 Grid increments SX, SY (degr.or m) and scaling (division) factor.
$    If NX*SX = 360., latitudinal closure is applied.
$  3 Coordinates of (1,1) (degr.) and scaling (division) factor.
$  4 Limiting bottom depth (m) to discriminate between land and sea
$    points, minimum water depth (m) as allowed in model, unit number
$    of file with bottom depths, scale factor for bottom depths (mult.),
$    IDLA, IDFM, format for formatted read, FROM and filename.
$      IDLA : Layout indicator :
$                   1   : Read line-by-line bottom to top.
$                   2   : Like 1, single read statement.
$                   3   : Read line-by-line top to bottom.
$                   4   : Like 3, single read statement.
$      IDFM : format indicator :
$                   1   : Free format.
$                   2   : Fixed format with above format descriptor.
$                   3   : Unformatted.
$      FROM : file type parameter
$                 'UNIT' : open file by unit number only.
$                 'NAME' : open file by name and assign to unit.
$
$ Example for longitude-latitude grid (switch !/LLG), for Cartesian
$ grid the unit is meters (NOT km).
$
     12     12
      1.     1.     4.
     -1.    -1.     4.
     -0.1 2.50  10  -10. 3 1 '(....)' 'NAME' 'bottom.inp'
$
$ If the above unit number equals 10, the bottom data is read from
$ this file and follows below (no intermediate comment lines allowed).
$
  6 6 6 6 6 6 6 6 6 6 6 6
  6 6 6 5 4 2 0 2 4 5 6 6
  6 6 6 5 4 2 0 2 4 5 6 6
  6 6 6 5 4 2 0 2 4 5 6 6
  6 6 6 5 4 2 0 0 4 5 6 6
  6 6 6 5 4 4 2 2 4 5 6 6
  6 6 6 6 5 5 4 4 5 6 6 6
  6 6 6 6 6 6 5 5 6 6 6 6
  6 6 6 6 6 6 6 6 6 6 6 6
```

```
  6 6 6 6 6 6 6 6 6 6 6 6
  6 6 6 6 6 6 6 6 6 6 6 6
  6 6 6 6 6 6 6 6 6 6 6 6
$
$ If sub-grid information is available as indicated by FLAGTR above,
$ additional input to define this is needed below. In such cases a
$ field of fractional obstructions at or between grid points needs to
$ be supplied.  First the location and format of the data is defined
$ by (as above) :
$  - Unit number of file (can be 10, and/or identical to bottem depth
$    unit), scale factor for fractional obstruction, IDLA, IDFM,
$    format for formatted read, FROM and filename
$
   10 0.2  3 1 '(....)' 'NAME' 'obstr.inp'
$
$ *** NOTE if this unit number is the same as the previous bottom
$     depth unit number, it is assumed that this is the same file
$     without further checks.                                    ***
$
$ If the above unit number equals 10, the bottom data is read from
$ this file and follows below (no intermediate comment lines allowed,
$ except between the two fields).
$
  0 0 0 0 0 0 0 0 0 0 0 0
  0 0 0 0 0 0 0 0 0 0 0 0
  0 0 0 0 0 0 0 0 0 0 0 0
  0 0 0 0 0 0 0 0 0 0 0 0
  0 0 0 0 0 0 0 0 0 0 0 0
  0 0 0 0 0 0 5 0 0 0 0 0
  0 0 0 0 0 0 5 0 0 0 0 0
  0 0 0 0 0 0 4 0 0 0 0 0
  0 0 0 0 0 0 4 0 0 0 0 0
  0 0 0 0 0 0 5 0 0 0 0 0
  0 0 0 0 0 0 5 0 0 0 0 0
  0 0 0 0 0 0 0 0 0 0 0 0
$
  0 0 0 0 0 0 0 0 0 0 0 0
  0 0 0 0 0 0 0 0 0 0 0 0
  0 0 0 0 0 0 0 0 0 0 0 0
  0 0 0 0 0 0 0 0 0 0 0 0
  0 0 0 0 0 0 0 0 5 5 5 0
  0 0 0 0 0 0 0 0 0 0 0 0
  0 0 0 0 0 0 0 0 0 0 0 0
  0 0 0 0 0 0 0 0 0 0 0 0
  0 0 0 0 0 0 0 0 0 0 0 0
```

```
  0 0 0 0 0 0 0 0 0 0 0 0
  0 0 0 0 0 0 0 0 0 0 0 0
  0 0 0 0 0 0 0 0 0 0 0 0
$
$ *** NOTE size of fields is always NX * NY                        ***
$
$ Input boundary points and excluded points --------------------------- $
$    The first line identifies where to get the map data, by unit number
$    IDLA and IDFM, format for formatted read, FROM and filename
$    if FROM = 'PART', then segmented data is read from below, else
$    the data is read from file as with the other inputs (as INTEGER)
$
   10 3 1 '(....)' 'PART' 'mapsta.inp'
$
$ Read the status map from file ( FROM != PART ) ---------------------- $
$
$ 3 3 3 3 3 3 3 3 3 3 3 3
$ 3 2 1 1 1 1 0 1 1 1 1 3
$ 3 2 1 1 1 1 0 1 1 1 1 3
$ 3 2 1 1 1 1 0 1 1 1 1 3
$ 3 2 1 1 1 1 0 0 1 1 1 3
$ 3 2 1 1 1 1 1 1 1 1 1 3
$ 3 2 1 1 1 1 1 1 1 1 1 3
$ 3 2 1 1 1 1 1 1 1 1 1 3
$ 3 2 1 1 1 1 1 1 1 1 1 3
$ 3 2 1 1 1 1 1 1 1 1 1 3
$ 3 2 1 1 1 1 1 1 1 1 1 3
$ 3 3 3 3 3 3 3 3 3 3 3 3
$
$ The legend for the input map is :
$
$    0 : Land point.
$    1 : Regular sea point.
$    2 : Active boundary point.
$    3 : Point excluded from grid.
$
$ Input boundary points from segment data ( FROM = PART ) ------------ $
$   An unlimited number of lines identifying points at which input
$   boundary conditions are to be defined. If the actual input data is
$   not defined in the actual wave model run, the initial conditions
$   will be applied as constant boundary conditions. Each line contains:
$     Discrete grid counters (IX,IY) of the active point and a
$     connect flag. If this flag is true, and the present and previous
$     point are on a grid line or diagonal, all intermediate points
$     are also defined as boundary points.
```

```
$
      2   2   F
      2  11   T
$
$  Close list by defining point (0,0) (mandatory)
$
      0   0   F
$
$ Excluded grid points from segment data ( FROM != PART )
$   First defined as lines, identical to the definition of the input
$   boundary points, and closed the same way.
$
      0   0   F
$
$   Second, define a point in a closed body of sea points to remove
$   the entire body of sea points. Also close by point (0,0)
$
      0   0
$
$ Output boundary points ------------------------------------------- $
$ Output boundary points are defined as a number of straight lines,
$ defined by its starting point (X0,Y0), increments (DX,DY) and number
$ of points. A negative number of points starts a new output file.
$ Note that this data is only generated if requested by the actual
$ program. Example again for spherical grid in degrees. Note, these do
$ not need to be defined for data transfer between grids in te multi
$ grid driver.
$
     1.75  1.50  0.25 -0.10     3
     2.25  1.50 -0.10  0.00    -6
     0.10  0.10  0.10  0.00   -10
$
$  Close list by defining line with 0 points (mandatory)
$
     0.    0.    0.    0.       0
$
$ ------------------------------------------------------------------ $
$ End of input file                                                  $
$ ------------------------------------------------------------------ $
```

end of example input file

4.4.3 The initial conditions program

Program	:	ww3_strt		(W3STRT)
Code	:	ww3_strt.ftn		
Input	:	ww3_strt.inp	(10)	Formatted input file for program.
		mod_def.ww3	(20)	Model definition file.
Output	:	standard out	(6)	Formatted output of program.
		restart.ww3	(20)	Restart file in WAVEWATCH III^{TM} format.

start of example input file

```
$ ------------------------------------------------------------------- $
$ WAVEWATCH III Initial conditions input file                          $
$------------------------------------------------------------------- $
$ type of initial field ITYPE .
$
    1
$
$ ITYPE = 1 ---------------------------------------------------------- $
$ Gaussian in frequency and space, cos type in direction.
$ - fp and spread (Hz), mean direction (degr., oceanographic
$    convention) and cosine power, Xm and spread (degr. or m) Ym and
$    spread (degr. or m), Hmax (m) (Example for lon-lat grid in degr.).
$
$   0.10  0.01  270. 2  1. 0.5 1. 0.5 2.5
    0.10  0.01  270. 2  0. 1000. 1. 1000. 2.5
$   0.10  0.01  270. 2  0. 1000. 1. 1000. 0.01
$   0.10  0.01  270. 2  0. 1000. 1. 1000. 0.
$
$ ITYPE = 2 ---------------------------------------------------------- $
$ JONSWAP spectrum with Hasselmann et al. (1980) direct. distribution.
$ - alfa, peak freq. (Hz), mean direction (degr., oceanographical
$   convention), gamma, sigA, sigB, Xm and spread (degr. or m) Ym and
$   spread (degr. or m)  (Example for lon-lat grid in degr.).
$   alfa, sigA, sigB give default values if less than or equal to 0.
$
$   0.0081  0.1  270.  1.0 0. 0. 1. 100. 1. 100.
$
$ ITYPE = 3 ---------------------------------------------------------- $
$ Fetch-limited JONSWAP
$ - No additional data, the local spectrum is calculated using the
$   local wind speed and direction, using the spatial grid size as
$   fetch, and assuring that the spectrum is within the discrete
```

```
$   frequency range.
$
$ ITYPE = 4 ---------------------------------------------------------- $
$ User-defined spectrum
$ - Scale factor., defaults to 1 if less than or equal 0.
$ - Spectrum F(f,theta) (single read statement)
$
$ -0.1
$  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
$  0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
$  0 1 4 2 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
$  0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
$  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
$  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
$  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
$  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
$  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
$  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
$  0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0
$  0 0 0 0 0 0 0 0 0 0 1 2 3 2 1 1 0 0 0 0 0 0 0 0 0
$  0 0 0 0 0 0 0 0 0 1 3 9 7 5 3 2 1 0 0 0 0 0 0 0 0
$  0 0 0 0 0 0 0 0 0 0 1 3 4 3 2 1 0 0 0 0 0 0 0 0 0
$  0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0
$  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
$  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
$  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
$  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
$  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
$  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
$  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
$  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
$  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
$
$ ITYPE = 5 ---------------------------------------------------------- $
$ Starting from calm conditions.
$ - No additional data.
$
$ -------------------------------------------------------------------- $
$ End of input file                                                    $
$ -------------------------------------------------------------------- $
```

end of example input file

4.4.4 The field preprocessor for the generic shell

Program	:	ww3_prep		(W3PREP)
Code	:	ww3_prep.ftn		
Input	:	ww3_prep.inp	(10)	Formatted input file for program.
		mod_def.ww3	(11)	Model definition file.
		'user input'*	(user)	See example below.
Output	:	standard out	(6)	Formatted output of program.
		level.ww3*	(12)	Water levels file.
		current.ww3*	(12)	Current fields file.
		wind.ww3*	(12)	Wind fields file.
		ice.ww3*	(12)	Ice fields file.
		data0.ww3*	(12)	Assimilation data ('mean').
		data1.ww3*	(12)	Assimilation data ('1-D spectra').
		data2.ww3*	(12)	Assimilation data ('2-D spectra').

start of example input file

```
$ ------------------------------------------------------------------- $
$ WAVEWATCH III Field preprocessor input file                          $
$ ------------------------------------------------------------------- $
$ Mayor types of field and time flag
$   Field types  :  ICE   Ice concentrations.
$                   LEV   Water levels.
$                   WND   Winds.
$                   WNS   Winds (including air-sea temp. dif.)
$                   CUR   Currents.
$                   DAT   Data for assimilation.
$   Format types :  AI    Transfer field 'as is'.
$                   LL    Field defined on regular longitude-latitude
$                         or Cartesian grid.
$                   F1    Arbitrary grid, longitude and latitude of
$                         each grid point given in separate file.
$                   F2    Like F1, composite of 2 fields.
$
$   NOTE : Options F1 and F2 have not yet been set up or tested for the
$          Cartesian version of the model.
$        - Format type not used for field type 'DAT'.
$
$   Time flag    : If true, time is included in file.
$   Header flag  : If true, header is added to file.
$                  (necessary for reading, FALSE is used only for
```

```
$                  incremental generation of a data file.)
$
   'ICE' 'LL' F T
$
$ Additional time input ------------------------------------------------ $
$ If time flag is .FALSE., give time of field in yyyymmdd hhmmss format.
$
   19680606 053000
$
$ Additional input format type 'LL' ---------------------------------- $
$ Grid range (degr. or m) and number of points for axes, respectively.
$ Example for longitude-latitude grid.
$
    -0.25 2.5  15  -0.25  2.5  4
$
$ Additional input format type 'F1' or 'F2' --------------------------- $
$ Three or four additional input lines, to define the file(s) with
$ the grid information :
$ 1) Discrete size of input grid (NXI,NYI).
$ 2) Define type of file using the parameters FROM, IDLA, IDFM (see
$    input for grid preprocessor), and a format
$ 3) Unit number and (dummy) name of first file.
$ 4) Unit number and (dummy) name of second file (F2 only).
$
$ 15  3
$ 'UNIT' 3 1 '(.L.L.)'
$ 10 'll_file.1'
$ 10 'll_file.2'
$
$ Additional input for data ------------------------------------------- $
$ Dimension of data (0,1,2 for mean pars, 1D or 2D spectra), "record
$ length" for data, data value for missing data
$
$  0  4  -999.
$
$ Define data files --------------------------------------------------- $
$ The first input line identifies the file format with FROM, IDLA and
$ IDFM, the second (third) lines give the file unit number and name.
$
  'UNIT' 3 1 '(..T..)' '(..F..)'
  10 'data_file.1'
$ 10 'data_file.2'
$
$ If the above unit numbers are 10, data is read from this file
$ (no intermediate comment lines allowed),
```

```
$ This example is an ice concentration field.
$
   1. 1. 1. 1. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0.
   1. 1. .5 .5 .5 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
   0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
   0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
$
$ This example is mean parameter assimilation data
$ First record gives number of data records, data are read as as
$ individual records of reals with recored length as given above
$
$  3
$ 1.5  1.6 0.70 10.3
$ 1.7  1.5 0.75  9.8
$ 1.9  1.4 0.77 11.1
$
$ ------------------------------------------------------------------- $
$ End of input file                                                   $
$ ------------------------------------------------------------------- $
```

end of example input file

Note that the optional output files are specific to ww3_shel and ww3_multi, but are not processed by the actual wave model routines. These files are consequently not needed if the wave model routines are used in a different shell or in an integrated program. However, the routines reading and writing these files are system-independent and could therefore be used in customized applications of the basic wave model. The reading and writing of these files is performed by the subroutine W3FLDG (w3fldsmd.ftn). For additional documentation and file formats reference if made to this routine.

4.4.5 The generic shell

Program	:	ww3_shel		(W3SHEL)
Code	:	ww3_shel.ftn		
Input	:	ww3_shel.inp	(10)	Formatted input file for program.
		mod_def.ww3	(30)	Model definition file.
		restart.ww3	(30)	Restart file.
		nest.ww3*	(33)	Boundary conditions file.
		level.ww3*	(11)	Water levels file.
		current.ww3*	(12)	Current fields file.
		wind.ww3*	(13)	Wind fields file.
		ice.ww3*	(14)	Ice fields file.
		data0.ww3*	(15)	Assimilation data.
		data1.ww3*	(16)	Assimilation data.
		data2.ww3*	(17)	Assimilation data.
		track_i.ww3*	(22)	Output track information.
Output	:	standard out	(6)	Formatted output of program.
		log.ww3	(20)	Output log of wave model (see section 4.2).
		test.ww3*	(6/21)	Test output of wave model.
		restart*n*.ww3*	(30)	Restart file(s).
		nest*n*.ww3*	(34-42)	Nesting file(s).
		out_grd.ww3*	(31)	Raw output of gridded fields.
		out_pnt.ww3*	(32)	Raw output of spectra.
		track_o.ww3*	(23)	Raw output of spectra along tracks.
Scratch	:	ww3_shel.scratch	(90)	Formatted scratch file.

start of example input file

```
$ -------------------------------------------------------------------- $
$ WAVEWATCH III shell input file                                       $
$ -------------------------------------------------------------------- $
$ Define input to be used with flag for use and flag for definition
$ as a homogeneous field (first three only); eight input lines.
$
   F F     Water levels
   F F     Currents
   T T     Winds
```

```
   T       Ice concentrations
   F       Assimilation data : Mean parameters
   F       Assimilation data : 1-D spectra
   F       Assimilation data : 2-D spectra.
$
$ Time frame of calculations ------------------------------------------ $
$ - Starting time in yyyymmdd hhmmss format.
$ - Ending time in yyyymmdd hhmmss format.
$
   19680606 000000
   19680606 060000
$
$ Define output data --------------------------------------------------- $
$
$ Define output server mode. This is used only in the parallel version
$ of the model. To keep the input file consistent, it is always needed.
$ IOSTYP = 1 is generally recommended. IOSTYP > 2 may be more efficient
$ for massively parallel computations. Only IOSTYP = 0 requires a true
$ parallel file system like GPFS.
$
$    IOSTYP = 0 : No data server processes, direct access output from
$                 each process (requirese true parallel file system).
$             1 : No data server process. All output for each type
$                 performed by process that performes computations too.
$             2 : Last process is reserved for all output, and does no
$                 computing.
$             3 : Multiple dedicated output processes.
$
   1
$
$ Five output types are available (see below). All output types share
$ a similar format for the first input line:
$ - first time in yyyymmdd hhmmss format, output interval (s), and
$   last time in yyyymmdd hhmmss format (all integers).
$ Output is disabled by setting the output interval to 0.
$
$ Type 1 : Fields of mean wave parameters
$          Standard line and line with flags to activate output fields
$          as defined in section 2.4 of the manual. The second line is
$          not supplied if no output is requested.
$                               The raw data file is out_grd.ww3,
$                               see w3iogo.ftn for additional doc.
$
$
   19680606 000000   3600  19680608 000000
```

```
     T T T T T  T T T T T  T T T T T  T T T T T  T T T T T
     T T T T T  T
$
$ Type 2 : Point output
$          Standard line and a number of lines identifying the
$          longitude, latitude and name (C*10) of output points.
$          The list is closed by defining a point with the name
$          'STOPSTRING'. No point info read if no point output is
$          requested (i.e., no 'STOPSTRING' needed).
$          Example for spherical grid.
$                               The raw data file is out_pnt.ww3,
$                               see w3iogo.ftn for additional doc.
$
$   NOTE : Spaces may be included in the name, but this is not
$          advised, because it will break the GrADS utility to
$          plots spectra and source terms, and will make it more
$          diffucult to use point names in data files.
$
   19680606 000000    900  19680608 000000
$
    -0.25 -0.25 'Land      '
     0.0   0.0  'Point_1   '
     2.0   1.0  'Point_2   '
     1.8   2.2  'Point_3   '
     2.1   0.9  'Point_4   '
     5.0   5.0  'Outside   '
$
     0.0   0.0  'STOPSTRING'
$
$ Type 3 : Output along  track.
$          Flag for formatted input file.
$                         The data files are track_i.ww3 and
$                         track_o.ww3, see w3iotr.ftn for ad. doc.
$
   19680606 000000   1800  19680606 013000
     T
$
$ Type 4 : Restart files (no additional data required).
$                               The data file is restartN.ww3, see
$                               w3iors.ftn for additional doc.
$
   19680606 030000      1  19680606 030000
$
$ Type 5 : Boundary data (no additional data required).
$                               The data file is nestN.ww3, see
```

```
$                              w3iobp.ftn for additional doc.
$
   19680606 000000   3600  20010102 000000
$
$ Type 6 : Separated wave field data (dummy for now).
$          First, last step IX and IY, flag for formatted file
$
   19680606 000000   3600  20010102 000000
      0 999 1 0 999 1 T
$
$ Homogeneous field data ------------------------------------------- $
$ Homogeneous fields can be defined by a list of lines containing an ID
$ string 'LEV' 'CUR' 'WND', date and time information (yyyymmdd
$ hhmmss), value (S.I. units), direction (current and wind, oceanogr.
$ convention degrees)) and air-sea temparature difference (degrees C).
$ 'STP' is mandatory stop string.
$ Also defined here are the speed with which the grid is moved
$ continuously, ID string 'MOV', parameters as for 'CUR'.
$
   'LEV' 19680606 010000    1.0
   'CUR' 19680606 073125    2.0    25.
   'WND' 19680606 000000   20.    145.    2.0
   'MOV' 19680606 013000    4.0    25.
   'STP'
$
$ ------------------------------------------------------------------- $
$ End of input file                                                   $
$ ------------------------------------------------------------------- $
```

end of example input file

4.4.6 The multi-grid shell

Program	:	ww3_multi		(W3MLTI)
Code	:	ww3_multi.ftn		
Input	:	ww3_multi.inp	(8)	Input file for multi-grid wave model shel.
Output	:	standard out	(6)	Formatted output of program.
		log.mww3	(9)	Output log of wave model driver.
		test.mww3*	(auto)	Test output of wave model.

This wave model program requires and produces a plethora of input and output files consistent with those of ww3_shel in section 4.4.5, where file extensions .ww3 are replaced by an identifier for a specific grid. Note that all files are opened by name, and that the unit number assignment is dynamic and automatic.

start of example input file

```
$ ------------------------------------------------------------------- $
$ WAVEWATCH III multi-grid model driver input file                    $
$ ------------------------------------------------------------------- $
$
$ ********************************************************************
$ *** NOTE : This is an example file from the mww3_test_05 script ***
$ ***        Unlilke other input example files this one CANNOT    ***
$ ***        be run as an independent interactive run             ***
$ ********************************************************************
$
$ The first input line sets up the general multi-grid model definition
$ by defining the follwing six parameters :
$
$   1) Number of wave model grids.i                         ( NRGRD )
$   2) Number of grids definint input fields.               ( NRINP )
$   3) Flag for using unified point output file.           ( UNIPTS )
$   4) Output server type as in ww3_shel.inp
$   5) Flag for dedicated process for iunified point output.
$   6) Flag for grids sharing dedicated output processes.
$
  3 1 T 1 T T
$
$ ------------------------------------------------------------------- $
$ If there are input data grids defined ( NRINP > 0 ), then these
$ grids are defined first. These grids are defined as if they are wave
```

```
$ model grids using the file mod_def.MODID. Each grid is defined on
$ a separate input line with MODID, and eight input flags identifying
$ the presentce of 1) water levels 2) currents 3) winds 4) ice and
$ 5-7) assimilation data as in the file ww3_shel.inp.
$
  'input' F F T F F F F
$
$ In this example, we need the file mod_def.input to define the grid
$ and the file wind.input to provide the corresponding wind data.
$
$ ------------------------------------------------------------------- $
$ If all point output is gathered in a unified point output file
$ ( UNIPTS = .TRUE. ), then the output spectral grid needs to be
$ defined. Ths information is taken from a wave model grid, and only
$ the spectral definitions from this grid are relevant. Define the
$ name of this grid here
$
  'points'
$
$ In this example, we need the file mod_def.points to define the
$ spectral output grid, and the point output will be written to the
$ file out_pnt.points
$
$ ------------------------------------------------------------------- $
$ Now each actual wave model grid is defined using 13 parameters to be
$ read fom a single line in the file. Each line contains the following
$ parameters
$     1)   Define the grid with the extension of the mod_def file.
$    2-8)  Define the inputs used by the grids with 8 keywords
$          corresponding to the 8 flags defining the input in the
$          input files. Valid keywords are:
$            'no'      : This input is not used.
$            'native'  : This grid has its own input files, e.g. grid
$                        grdX (mod_def.grdX) uses ice.grdX.
$            'MODID'   : Take input from the grid identified by
$                        MODID. In the example below, all grids get
$                        their wind from wind.input (mod_def.input).
$     9)   Rank number of grid (internally sorted and reassigned).
$    10)   Group number (internally reassigned so that different
$          ranks result in different group numbers.
$   11-12) Define fraction of cumminicator (processes) used for this
$          grid.
$    13)   Flag identifying dumping of boundary data used by this
$          grid. If true, the file nest.MODID is generated.
$
```

```
  'grd1'  'no' 'no' 'input' 'no' 'no' 'no' 'no'   1  1  0.00 1.00  F
  'grd2'  'no' 'no' 'input' 'no' 'no' 'no' 'no'   2  1  0.00 1.00  F
  'grd3'  'no' 'no' 'input' 'no' 'no' 'no' 'no'   3  1  0.00 1.00  F
$ 'grd1'  'no' 'no' 'input' 'no' 'no' 'no' 'no'   1  1  0.00 0.50  F
$ 'grd2'  'no' 'no' 'input' 'no' 'no' 'no' 'no'   2  1  0.25 0.75  F
$ 'grd3'  'no' 'no' 'input' 'no' 'no' 'no' 'no'   3  1  0.50 1.00  F
$
$ In this example three grids are used requiring the files
$ mod_def.grdN. All files get ther winds from the grid 'input'
$ defined by mod_def.input, and no other inputs are used. In the lines
$ that are commented out, each grid runs on a part of the pool of
$ processes assigned to the computation.
$
$ ------------------------------------------------------------------- $
$ Starting and ending times for the entire model run
$
   19680606 000000   19680607 000000
$
$ ------------------------------------------------------------------- $
$ Specific multi-scale model settings (single line).
$    Flag for masking computation in two-way nesting (except at
$                                                      output times).
$    Flag for masking at printout time.
$
  T T
$
$ ------------------------------------------------------------------- $
$ Conventional output requests as in ww3_shel.inp.  Will be applied
$ to all grids.
$
   19680606 000000   3600  19680607 000000
  T T T F F  T F F F F  T T F F T  T F T F F  F T F F F  F F F F F  F
$
$ NOTE: Ff UNIPTS = .TRUE. then the point output needs to be defined
$       here and cannot be redifined below.
$
   19680606 000000   3600  19680608 000000
      0.E3     0.E3  'eye       '
      0.E3    50.E3  'mN        '
    -35.E3    35.E3  'mNW       '
    -50.E3     0.E3  'mW        '
    -35.E3   -35.E3  'mSW       '
      0.E3   -50.E3  'mS        '
     35.E3   -35.E3  'mSE       '
     50.E3     0.E3  'mE        '
```

```
      35.E3    35.E3  'mNE       '
       0.E3   100.E3  'aN        '
     -70.E3    70.E3  'aNW       '
    -100.E3     0.E3  'aW        '
     -70.E3   -70.E3  'aSW       '
       0.E3  -100.E3  'aS        '
      70.E3   -70.E3  'aSE       '
     100.E3     0.E3  'aE        '
      70.E3    70.E3  'aNE       '
       0.E3   210.E3  'bN        '
    -150.E3   150.E3  'bNW       '
    -210.E3     0.E3  'bW        '
    -150.E3  -150.E3  'bSW       '
       0.E3  -210.E3  'bS        '
     150.E3  -150.E3  'bSE       '
     210.E3     0.E3  'bE        '
     150.E3   150.E3  'bNE       '
       0.E3   800.E3  'cN        '
    -550.E3   550.E3  'cNW       '
    -800.E3     0.E3  'cW        '
    -550.E3  -550.E3  'cSW       '
       0.E3  -800.E3  'cS        '
     550.E3  -550.E3  'cSE       '
     800.E3     0.E3  'cE        '
     550.E3   550.E3  'cNE       '
       0.E3     0.E3  'STOPSTRING'
$
   19680606 000000      0  19680608 000000
$
   19680606 000000      0  19680608 000000
$
   19680606 000000      0  19680608 000000
$
   19680606 000000      0  19680608 000000
$
$ -------------------------------------------------------------------- $
$ Output requests per grid and type to overwrite general setup
$ as defined above. First record per set is the grid name MODID
$ and the output type number. Then follows the standard time string,
$ and conventional data as per output type. In mww3_test_05 this is
$ not used. Below, one example generating partitioning output for
$ the inner grid is included but commented out.
$
$ 'grd3'  6
$  19680606 000000    900  19680608 000000
```

```
$     0 999 1 0 999 1 T
$
$ -------------------------------------------------------------------- $
$ Mandatory end of outpout requests per grid, identified by output
$ type set to 0.
$
  'the_end'  0
$
$ -------------------------------------------------------------------- $
$ Moving grid data as in ww3_hel.inp. All grids will use same data.
$
  'MOV' 19680606 000000   5.  90.
  'STP'
$
$ -------------------------------------------------------------------- $
$ End of input file                                                    $
$ -------------------------------------------------------------------- $
```

end of example input file

4.4.7 Gridded output post-processor

Program	:	ww3_outf		(W3OUTF)
Code	:	ww3_outf.ftn		
Input	:	ww3_outf.inp	(10)	Input file for gridded output post-processor.
		mod_def.ww3	(20)	Model definition file.
		out_grd.ww3	(20)	Raw gridded output data.
Output	:	standard out	(6)	Formatted output of program.
		... *	(50)	Transfer file.

start of example input file

```
$ ------------------------------------------------------------------- $
$ WAVEWATCH III Grid output post-processing                            $
$------------------------------------------------------------------- $
$ Time, time increment and number of outputs
$
  19680606 060000 10800. 1
$
$ Request flags identifying fields as in ww3_shel input and
$ section 2.4 of the manual.
$
  F F F F F  T F F F F  F F F F F  F F F F F  F F F F F  F F F F F  F
$ F F F F F  F F F F F  F F F F T  T T T T T  T T F F F  F F F F F  F
$ T T T T T  T T T T T  T T T T T  T T T T T  T T T T T  T T T T T  T
$
$ Output type ITYPE [0,1,2,3], and IPART [ 0,...,NOSWLL ]
$
  1  2
$ ------------------------------------------------------------------- $
$ ITYPE = 0, inventory of file.
$            No additional input, the above time range is ignored.
$
$ ------------------------------------------------------------------- $
$ ITYPE = 1, print plots.
$            IX,IY range and stride, flag for automatic scaling to
$            maximum value (otherwise fixed scaling),
$            vector component flag (dummy for scalar quantities),
$
  1 12 1 1 12 1 F F
$
$ ------------------------------------------------------------------- $
$ ITYPE = 2, field statistics.
```

```
$            IX,IY range.
$
$ 1 12 1 12
$
$ ------------------------------------------------------------------- $
$ ITYPE = 3, transfer files.
$            IX, IY range, IDLA and IDFM as in ww3_grid.inp.
$            The additional option IDLA=5 gives ia longitude, lattitude
$            and parameter value(s) per record (defined points only),
$
$ 2 11 2 11 1 2
$
$ For each field and time a new file is generated with the file name
$ ww3.yymmddhh.xxx, where yymmddhh is a conventional time idicator,
$ and xxx is a field identifier. The first record of the file contains
$ a file ID (C*13), the time in yyyymmdd hhmmss format, the lowest,
$ highest and number of longitudes (2R,I), id.  latitudes, the file
$ extension name (C*$), a scale factor (R), a unit identifier (C*10),
$ IDLA, IDFM, a format (C*11) and a number identifying undefined or
$ missing values (land, ice, etc.).  The field follows as defined by
$ IDFM and IDLA, defined as in the grid proprocessor. IDLA=5 is added
$ and gives a set of records containing the longitude, latitude and
$ parameter value. Note that the actual data is written as an integers.
$
$ ------------------------------------------------------------------- $
$ End of input file                                                   $
$ ------------------------------------------------------------------- $
```

end of example input file

The extension of the file name of transfer files for ITYPE = 3 identifies the content of the file. The file extension for each data type is given in Table 4.1 on page 111.

4.4.8 Point output post-processor

Program	:	ww3_outp		(W3OUTP)
Code	:	ww3_outp.ftn		
Input	:	ww3_outp.inp	(10)	Input file for point output post-processor.
		mod_def.ww3	(20)	Model definition file.
		out_pnt.ww3	(20)	Raw point output data.
Output	:	standard out	(6)	Formatted output of program.
		tab*nn*.ww3 *	(*nn*)	Table of mean parameters where *nn* is a two-digit integer.
		. . . *	(user)	Transfer file.

start of example input file

```
$ ------------------------------------------------------------------- $
$ WAVEWATCH III Point output post-processing                          $
$------------------------------------------------------------------- $
$ First output time (yyyymmdd hhmmss), increment of output (s),
$ and number of output times.
$
  19680606 060000  3600.  7
$
$ Points requested ------------------------------------------------- $
$ Define points for which output is to be generated.
$
  1
  2
  3
  4
$ mandatory end of list
 -1
$
$ Output type ITYPE [0,1,2,3]
$
  1
$ ------------------------------------------------------------------- $
$ ITYPE = 0, inventory of file.
$            No additional input, the above time range is ignored.
$
$ ------------------------------------------------------------------- $
$ ITYPE = 1, Spectra.
$          - Sub-type OTYPE :  1 : Print plots.
```

```
$                             2 : Table of 1-D spectra
$                             3 : Transfer file.
$                             4 : Spectral partitioning.
$          - Scaling factors for 1-D and 2-D spectra Negative factor
$            disables, output, factor = 0. gives normalized spectrum.
$          - Unit number for transfer file, also used in table file
$            name.
$          - Flag for unformatted transfer file.
$
    1   0.  0.  33  F
$
$ The transfer file contains records with the following contents.
$
$ - File ID in quotes, number of frequencies, directions and points.
$   grid name in quotes (for unformatted file C*21,3I,C*30).
$ - Bin frequenies in Hz for all bins.
$ - Bin directions in radians for all bins (Oceanographic conv.).
$                                                          -+
$ - Time in yyyymmdd hhmmss format                          | loop
$                                                 -+        |
$ - Point name (C*10), lat, lon, d, U10 and       |  loop    | over
$   direction, current speed and direction       |   over   |
$ - E(f,theta)                                    |  points  | times
$                                                 -+        -+
$
$ The formatted file is readable usign free format throughout.
$ This datat set can be used as input for the bulletin generator
$ w3split.
$
$ ------------------------------------------------------------------- $
$ ITYPE = 2, Tables of (mean) parameter
$          - Sub-type OTYPE :  1 : Depth, current, wind
$                              2 : Mean wave pars.
$                              3 : Nondimensional pars. (U*)
$                              4 : Nondimensional pars. (U10)
$                              5 : 'Validation table'
$                              6 : WMO standard output
$          - Unit number for file, also used in file name.
$
$ 2  33
$
$ If output for one point is requested, a time series table is made,
$ otherwise the file contains a separate tables for each output time.
$
$ ------------------------------------------------------------------- $
```

```
$ ITYPE = 3, Source terms
$          - Sub-type OTYPE :  1 : Print plots.
$                              2 : Table of 1-D S(f).
$                              3 : Table of 1-D inverse time scales
$                                  (1/T = S/F).
$                              4 : Transfer file
$          - Scaling factors for 1-D and 2-D source terms. Negative
$            factor disables print plots, factor = 0. gives normalized
$            print plots.
$          - Unit number for transfer file, also used in table file
$            name.
$          - Flags for spectrum, input, interactions, dissipation,
$            bottom and total source term.
$          - scale ISCALE for OTYPE=2,3
$                              0 : Dimensional.
$                              1 : Nondimensional in terms of U10
$                              2 : Nondimensional in terms of U*
$                             3-5: like 0-2 with f normalized with fp.
$          - Flag for unformatted transfer file.
$
$ 1  0. 0. 50   T T T T T T  0  F
$
$ The transfer file contains records with the following contents.
$
$ - File ID in quotes, nubmer of frequencies, directions and points,
$   flags for spectrum and source terms (C*21, 3I, 6L)
$ - Bin frequenies in Hz for all bins.
$ - Bin directions in radians for all bins (Oceanographic conv.).
$                                                          -+
$ - Time in yyyymmdd hhmmss format                          | loop
$                                                -+         |
$ - Point name (C*10), depth, wind speed and      |  loop    | over
$   direction, current speed and direction       |   over   |
$ - E(f,theta) if requested                       |  points  | times
$ - Sin(f,theta) if requested                     |          |
$ - Snl(f,theta) if requested                     |          |
$ - Sds(f,theta) if requested                     |          |
$ - Sbt(f,theta) if requested                     |          |
$ - Stot(f,theta) if requested                    |          |
$                                                -+         -+
$
$ ------------------------------------------------------------------- $
$ End of input file                                                   $
$ ------------------------------------------------------------------- $
```

end of example input file

The spectral data transfer file generated with ITYPE = 1 and OTYPE = 3 can be converted into a spectral bulletin using w3split (see section 5.2). This program reads the following five records from standard input (no comment lines allowed) :

- Name of output location.
- Identifier for run to be used in table.
- Name of input file.
- Logical identifying UNFORMATTED input file.
- Name of output file.

All above strings are read as characters using free format, and therefore need to be enclosed in quotes.

4.4.9 Track output post-processor

Program	:	ww3_trck		(W3TRCK)
Code	:	ww3_trck.ftn		
Input	:	track_o.ww3	(11)	Raw track output data.
Output	:	standard out	(6)	Formatted output of program.
		track.ww3	(51)	Formatted data file.

This post-processor does not require a formatted input file with program commands. It will simply convert the entire unformatted file to an integer compressed formatted file. The file contains the following header records :

- File identifier (character string of length 34).
- Number of frequencies and directions, first direction and directional increment (radians, oceanographic convention).
- Radian frequencies of each frequency bin.
- Corresponding directional bin size times frequency bin size to obtain discrete energy per bin.

For each output point the following records are printed :

- Date and time in `yyyymmdd hhmmss` format, longitude and latitude in degrees, and a status identifier 'ICE', 'LND' or 'SEA'. The following two records are written only for sea points.
- Water depth in meters, current and wind u and v components in meters per second, friction velocity in meters per second, air-sea temperature difference in degrees centigrade and scale factor for spectrum.
- The entire spectrum in integer packed format (can be read using free format).

4.4.10 GRIB output post-processor

Program	:	ww3_grib		(W3GRIB)
Code	:	ww3_grib.ftn		
Input	:	ww3_grib.inp	(10)	Input file for gridded output post-processor.
		mod_def.ww3	(20)	Model definition file.
		out_grd.ww3	(20)	Raw gridded output data.
Output	:	standard out	(6)	Formatted output of program.
		gribfile	(50)	GRIB file.

start of example input file

```
$ ------------------------------------------------------------------- $
$ WAVEWATCH III Grid output post-processing ( GRIB )                  $
$------------------------------------------------------------------- $
$ Time, time increment and number of outputs.
$
  19680606 000000  3600. 3
$
$ Request flags identifying fields as in ww3_shel input and
$ section 2.4 of the manual.
$
  T T T T T  T T F F T  T T T T T  T T T T T  T T T T T  T T T T T  T
$
$ Aditional info needed for grib file
$   Forecast time, center ID, generating process ID, grid definition
$   and GDS/BMS flag
$
  19680606 010000 7 10 255 192
$
$ ------------------------------------------------------------------- $
$ End of input file                                                   $
$ ------------------------------------------------------------------- $
```

end of example input file

This post-processor packs fields of mean wave parameters in GRIB format, using GRIB version II and NCEP's w3 and bacio library routines, or in GRIB2, using NCEPS's operational package. Additional packing data can be found in Table 4.1 on page 111.

The GRIB packing is performed using the NCEP's GRIB tables as described in NCEP (1998). Because the w3 and bacio routine are not fully

portable, they are not supplied with the code. The user will have to provide corresponding routines. It is suggested that such routines are activated with additional WAVEWATCH III ^{TM} switches in the mandatory switch group containing the 'NOGRB' switch, as if presently the case with the NCEP routines. The GRIB2 packing is performed according to WMO (2001), and is performed with NCEP's standard operational packages.

Table 4.1 shows the KPDS(5) data values for GRIB packing. For the partitioned data, the first number identifies the wind sea, the second number identifies swell. Most data are packed as surface data (KPDS(6) = 0). For the partitioned swell fields, however, consecutive fields are packed at consecutive levels, with the level type indicator set to (KPDS(6) = 241). KPDS(7) identifies the actual level or swell field number.

Table 4.1 shows several KPDS data values for GRIB2 packing. The first number in the table represents LISTSEC0(2), which identifies the discipline type (e.g., oceanography, meteorology, etc.) The second number represents KPDS(1), which identifies the parameter category (e.g., waves, circulation, ice, etc.) within the discipline type. The third number represents KPDS(2), which identifies the actual parameter. For the partitioned data, A/B means A for wind sea and B for swell. Additionally KPDS(10) = 0 for surface data, and KPDS(10) = 241 to pack consecutive swell fields at consecutive levels. KPDS(12) identifies the actual level or swell field number.

Although the above input file contains flags for all 31 output fields of WAVEWATCH III ^{TM}, not all fields can be packed in GRIB. If a parameter is chosen for which GRIB packing is not available, a message will be printed to standard output. Table 4.1 shows which parameter can be packed in GRIB. Note that at NCEP the conversions from GRIB to GRIB2 coincided with the introduction of partitioned wave model output. This required some duplicate definitions in GRIB and some apparent inconsistencies between GRIB and GRIB2 packing.

4.4.11 Gridded output post-processor for GrADS

Program	:	gx_outf		(GXOUTF)
Code	:	gx_outf.ftn		
Input	:	gx_outf.inp	(10)	Input file for gridded output post-processor.
		mod_def.ww3	(20)	Model definition file.
		out_grd.ww3	(20)	Raw gridded output data.
Output	:	standard out	(6)	Formatted output of program.
		ww3.grads	(50)	GrADS data file.
		ww3.ctl	(51)	GrADS control file.

start of example input file

```
$ ------------------------------------------------------------------- $
$ WAVEWATCH III Grid output post-processing ( GrADS )                 $
$-------------------------------------------------------------------- $
$ Time, time increment and number of outputs.
$
  19680606 000000  3600. 25
$
$ Request flags identifying fields as in ww3_shel input and
$ section 2.4 of the manual.
$
  F F T F T  T F F F F  T T F F T  T F T F F  T T F F F  F F F F F  F
$
$ Grid range in discrete counters IXmin,max, IYmin,max, flags for
$ including sea and boundary points in map
$
  0 999 0 999  T T
$
$ NOTE : In the Cartesian grid version of the code, X and Y are
$        converted to longitude and latitude assuming that 1 degree
$        equals 100 km if th maximum of X or Y is larger than 1000km.
$        For maxima between 100 and 1000km 1 degree is assumed to be
$        10km etc. Adjust labels in GrADS scripts accordingly.
$
$ ------------------------------------------------------------------- $
$ End of input file                                                   $
$ ------------------------------------------------------------------- $
```

end of example input file

This post-processor generates input files with gridded model parameters for the Grid Analysis and Display System (GrADS, Doty, 1995). This graphical software can be obtained from http://www.iges.org/grads. Although GrADS can also work with GRIB files, the present preprocessor is preferable, as the data file also gives access to a land-sea-ice map.

4.4.12 Point output post-processor for GrADS

Program	:	gx_outp		(GXOUTP)
Code	:	gx_outp.ftn		
Input	:	gx_outp.inp	(10)	Input file for point output post-processor.
		mod_def.ww3	(20)	Model definition file.
		out_pnt.ww3	(20)	Raw point output data.
Output	:	standard out	(6)	Formatted output of program.
		ww3.spec.grads	(30)	GrADS data file with spectra and source terms.
		ww3.mean.grads	(31)	File with mean wave parameters.
		ww3.spec.ctl	(32)	GrADS control file.

start of example input file

```
$ ------------------------------------------------------------------- $
$ WAVEWATCH III Point output post-processing ( GrADS )                $
$------------------------------------------------------------------- $
$ First output time (yyyymmdd hhmmss), increment of output (s),
$ and number of output times.
$
  19680606 000000  3600. 7
$
$ Points requested ------------------------------------------------- $
$ Define points for which output is to be generated.
$
$ 1
$ 2
$ 3
  4
$ mandatory end of list
 -1
$
$ ------------------------------------------------------------------- $
$   Flags for plotting F, Sin, Snl, Sds, Sbt, Stot
$
  T T T T T T
$
$ NOTE : In the Cartesian grid version of the code, X and Y are
$        converted to km. Use source_xy.gs instead of source.gs
$
$ ------------------------------------------------------------------- $
```

```
$ End of input file                                                  $
$ -------------------------------------------------------------------- $
```

end of example input file

This post-processor is intended to generate data files with which GrADS (see previous section) can plot polar plots of spectra and source terms. To achieve this, spectra and source terms are store as "longitude-latitude" grids. For each output point a different name is generated for the data, typically LOC*nnn*. When the data file is loaded in GrADS, the variable LOC001 will contain a spectral grid for the first requested output point at level 1, the input source term at level 2, etc. For the second output point the data is stored in LOC002 etc. The actual output point names are passed to GrADS through the control file ww3.spec.ctl. Wave heights and environmental data are obtained from ww3.mean.grads The user, however, need not be aware of the details of the GrADS data files and data storage. The GrADS scripts spec.gs, source.gs and 1source.gs are provided to automatically generate spectral plots from the output files of this post-processor.

Note: for the GrADS scripts to work properly, the names of the output points should not contain spaces.

field	description	file extension	GRIB1 data	GRIB2 data
1	depth	.dpt	–	–
2	mean current components	.cur	–	–
3	wind speed	.wnd	32	0,2,1
	wind direction		31	0,2,0
	wind u		33	0,2,2
	wind v		34	0,2,3
4	air-sea temp. dif.	.dt	–	–
5	friction velocity comp.	.ust	–	–
6	wave height H_s	.hs	100	10,0,3
7	mean wave length	.l	–	–
8	mean wave period T_m	.t	103	–
9	mean wave direction θ_m	.dir	101	–
10	directional spread σ	.spr	–	–
11	peak period T_p	.fp	108	10,0,11
12	peak direction θ_p	.dp	107	10,0,10
13	wind sea period T_w	.fpl	110	–
14	wind sea direction θ_w	.dpl	109	–
15	H_s of partition	.phs	102,105	10,0,5/8
16	T_p of partition	.ptp	103,106	10,0,6/9
17	L_p of partition	.plp	–	–
18	θ_m of partition	.pth	101,104	10,0,4/7
19	σ of partition	.psi	–	–
20	wind sea fraction of part.	.pws	–	–
21	total wind sea fraction	.wsf	–	–
22	number of partitions	.pnr	–	–
23	average time step	.dtd	–	–
24	cut-off frequency f_c	.fc	–	–
25	ice coverage	.ice	91	10,2,0
26	water level	.wlv	–	10,3,1
27	near-bottom amplitude	.abr	–	–
28	near-bottom velocity	.ubr	–	–
29	radiation stress	.Sxy	–	–
30	user defined #1	.us1	–	–
31	user defined #2	.us2	–	–

Table 4.1: Field output post processors ancillary data.

This page is intentionally left blank.

5 Installing the wave model

5.1 Introduction

WAVEWATCH III ™ is written in ANSI standard FORTRAN-90, with in essence no machine-dependent elements, so that WAVEWATCH III ™ can be installed without modifications on most platforms. WAVEWATCH III ™ utilizes its own preprocessor to process include files (presently only used for adding NCEP-specific documentation), to select model options at the compile level, and to switch test output on or off. This approach proved to be efficient during the development of WAVEWATCH III ™, but it complicates the installation of WAVEWATCH III ™. To minimize complications, a set of UNIX/Linux scripts is provided to automate the installation in general and the use of the preprocessor in particular. This option is not supported for other operation systems like MS or Mac products. If the code is to be compiled on one of the latter platforms, it is suggested to extract a working code in a UNIX/Linux environment using the utility `w3_source` (see below), and than to port this clean code to the platform of choice.

WARNING

If version 3.14 is implemented as an upgrade to previous versions of WAVEWATCH III ™, please note that this version may not be compatible with previous model versions. It is therefore prudent *NOT* to install the new version of WAVEWATCH III ™ on top of the old version. See Appendix A for suggestions on managing multiple model version.

WARNING

5.2 Installing files

In its packaged version, WAVEWATCH III ™ is contained in several files:

install_wwatch3	The WAVEWATCH III ™ install program.

wwatch3.aux.tar	Archive file containing programs and scripts controlling the compiling and linking of and code management of WAVEWATCH III ™.
wwatch3.ftn.tar	Archive file containing source code.
wwatch3.inp.tar	Archive file containing example input files.
wwatch3.tst.tar	Archive file containing test cases with results.

As the first step of installing WAVEWATCH III ™, these files have to be copied to a work directory on the machine on which WAVEWATCH III™ will be installed. Because this directory will be the 'home' directory of WAVEWATCH III ™, it is suggested that a new directory is created (see also warning in previous section). Furthermore install_wwatch3 has to be made executable by typing

```
chmod 700 install_wwatch3
```

after which the installation of the files is started by typing

```
install_wwatch3
```

WARNING

The install program will ask for a compiler to compile some auxiliary FORTRAN codes used in putting WAVEWATCH III ™ together. Unlike the actual WAVEWATCH III ™ source code, these programs are still written in FORTRAN-77. It is therefore sufficient to point toward the generic FORTRAN-77 compiler on the system. This choice of mixed codes was made to assure that if WAVEWATCH III ™ is put on a UNIX/Linux box only to put the code together, it will not be necessary to buy a FORTRAN-90 compiler.

WARNING

When install_wwatch3 is executed for the first time, it will ask the user to identify the directory in which WAVEWATCH III ™ will be installed. This has to be the directory in which the above five files reside. The program will then print a message that it cannot find the setup file, and ask some questions. The default answer or options are shown in square brackets. After the user has verified that the setup parameters are satisfactorily,

the install program will create the (hidden) setup file .wwatch3.env in the user's home directory. The setup can be modified by rerunning the install program, or by manually editing the setup file .wwatch3.env. The 'home' directory of WAVEWATCH III ™ can only be changed by editing or removing .wwatch3.env.

After the setup file is processed, the install program asks if the user wants to continue with the installation. If the user chooses to continue, the program will look for the four archive files. If a file is found, the program asks if the contents of the file should be installed. If no files are found, the archive files do not reside in the home directory, or the home directory is erroneously defined. To avoid that the install program generates unwanted files and directories in such a case, abort the install program (type Ctrl C), and check the location of the archive files, and the 'home' directory of WAVEWATCH III ™ (see previous paragraph).

After files to be unpacked have been identified, the program will ask if old files should be overwritten automatically. If the user chooses 'n', the program will ask permission to overwrite each file that already exists. Files that contain user specific information, such as compile and link options, will never be replaced by the install program.

As the first step of the actual installation, the install program checks if the following directories exist in the 'home' directory of WAVEWATCH III ™.

arc	Archive directory.
aux	Raw auxiliary programs (source codes etc.).
bin	Executables and shell scripts for compiling and linking.
exe	WAVEWATCH III ™ executables.
ftn	Source code and makefile.
inp	Input files.
mod	Module files.
obj	Object files.
test	Scripts with test cases.
work	Auxiliary work directory.

All these directories are generated by the install program install_wwatch3, except for the archive directory, which is generated by arc_wwatch3 (see below).

If installation of the auxiliary programs is requested (file wwatch3.aux.tar), the install program will first process source codes of auxiliary programs, using the compiler as defined by the user in the setup file. Note that these codes are still in fixed format FORTRAN-77.

w3adc.f	WAVEWATCH III ™ FORTRAN preprocessor.
w3prnt.f	Print files (source codes) including page and line numbers.
w3list.f	Generate a generic source code listing.
w3split.f	Generate spectral bulletin identifying individual wave fields within a spectrum from the spectral output of the point output post-processor (see section 4.4.8).

The above source codes are stored in the directory aux and the executables are stored in the directory bin. A more detailed description of these programs (including instructions on running the executables) can be found in the documentation included in the above source code files. After the compilation of these programs, several UNIX shell scripts and auxiliary files are installed in the bin directory.

ad3	Script to run the preprocessor w3adc and the compile script comp for a given source code file.
ad3_test	Test version of ad3, showing modifications to original source file. This script does not compile code.
all_switches	Generates a list of all w3adc switches present in the source code files.
arc_wwatch3	Program to archive versions of WAVEWATCH III™ in the directory arc.
comp.gen	Generic compiler script. The actual compiler script comp will be copied from this script if it does not exists.
comp.*xxx*	The compiler script comp for a specific hardware-compiler combination.
find_switch	Script to find WAVEWATCH III™ source code files containing compiler switches (or arbitrary strings).
link.gen	Generic linker script. Actual script is link.
link.*xxx*	The link script comp for a specific hardware-compiler combination.
list	Script to print source code listing using w3prnt.
ln3	Script to make symbolic link of source code file to work directory.
make_makefile.sh	Script to generate the of the makefile based on selections in the file switch).

switch.gen	Generic file with preprocessor switches (section 5.4).
w3_clean	Script to clean up work and scratch directories by removing files generated during compilation or test runs.
w3_make	Script to compile and link components of WAVEWATCH III ^{TM} using a makefile.
w3_new	Script to touch correct source code files to account for changes in compiler switches in combination with the makefile.
w3_source	Script to generate a true FORTRAN source code for any of he WAVEWATCH III ^{TM} program elements.

The use of these scripts is explained in section 5.3. After this, several GrADS scripts are installed in the aux directory.

cbarn.gs	Semi-standard GrADS script for displaying color bars.
colorset.gs	Script to define colors used in shading.
map2_n.gs	Script used in propagation test ww3_tp2.n, can be used as template for specialized map plots.
map_s3.gs	Idem for moving grid hurricane test ww3_ts3.
source.gs	Script for composite plot of spectra and source terms (2-D polar or Cartesian plots in color or in black and white).
1source.gs	Script to plot single source term.
spec.gs	Script to plot spectra.
profile.gs	Script to display profiling data generated by ww3_multi.

As the final step of processing wwatch3.aux.tar, some links between directories are established.

If the installation of the source code is requested (file wwatch3.ftn.tar), the install program will copy source code and include files to the directory ftn. All the files considered are discussed in detail in chapter 6.

If the installation of the input files is requested (file wwatch3.inp.tar), the install program will copy the example input files as presented in the previous chapter to the directory inp. Links are make to the work directory work. No other action is taken.

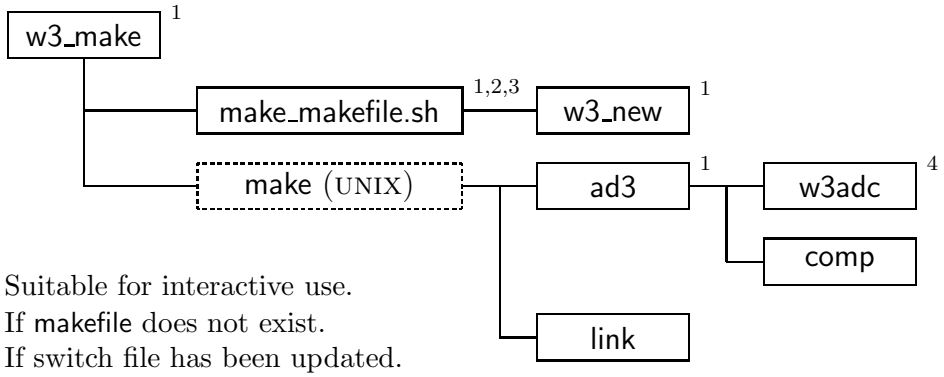

Figure 5.1: General layout of the compiler program w3_make.

If the installation of the test cases is requested (file wwatch3.tst.tar), the install program will copy the test cases to the directory test (see section 5.6). No other action is taken.

Finally, the install program lists manual modifications required by or suggested to the user. These messages are printed only if the compile and link system are installed (file wwatch3.aux.tar).

5.3 Compiling and linking

Compilation of WAVEWATCH III TM is performed using the script w3_make in the bin directory[8]. If this script is used without parameters, all basic programs of WAVEWATCH III TM are compiled. Optionally, names of programs to be compiled can be given as part of the compile command. For instance

```
w3_make ww3_grid ww3_strt
```

will compile the grid preprocessor and the initial conditions program only. w3_make uses several of the scripts described in the previous section. A graphical representation is given in Fig. 5.1.

[8] Note that before running w3_make several user interventions are needed as described in the remainder of this section.

If necessary, the script w3_make uses the scripts make_makefile.sh to generate a makefile. make_makefile.sh generates a list of modules to be linked, based on the program switches in the file switch (see section 5.4), and checks all needed sources for module dependencies. If switches have been changed since the last call to w3_make, w3_new is used to 'touch' relevant source code or to delete relevant object files. After the makefile has been completed, the standard UNIX make utility is used to compile and link the programs. Instead of directly using the FORTRAN compiler, the makefile invokes the preprocessor and compile scripts ad3 and comp, and the link script link. The script ad3 uses the extension of the file name to determine the necessary action. Files with extension .ftn are processed by w3adc, files with extension .f or .f90 are send to the script comp directly. Although a user could try out several of these scripts interactively, he or she generally needs to run w3_make only.

Before a first attempt is made at compiling, user intervention is required in three scripts/files. For convenience of debugging and development, links to these three files are made in the work directory work. The files in the work directory are

comp	Compiler script. This script requires the correct definition of the compiler and its options. Linked to ../bin/comp
link	Linker script. This script requires the correct definition of the linker and its options. Linked to ../bin/link
switch	File containing a list of switches as recognized by the preprocessor w3adc. Linked to ../bin/switch. The file provided with WAVEWATCH III^{TM} should result in a hardware independent code.

WARNING

The auxiliary scripts w3_make etc. us the switch, comp and link files from the ./bin directory under the WAVEWATCH III^{TM} home directory, *NOT* from the local directory.

WARNING

After the appropriate changes have been made, or the appropriate example scripts have been copied in, (parts of) WAVEWATCH III ^{TM} can be compiled and linked. When the program is compiled for the first time, it is suggested to compile program parts one-by-one to avoid lengthy errors messages, and to set up error capturing in comp. A good place to start is compilation of the simple test code CTEST. First go to the directory work and make a link to the source code of this routine by typing

```
ln3 ctest
```

This link is made to facilitate later inclusion of errors to test or set-up error capturing in the script comp. The inner workings of the preprocessor w3adc can be seen by typing the command

```
ad3_test ctest
```

which will show how the actual source code is constructed from ctest.ftn, include files and program switches. Next, the compilation of this subroutine can be tested by typing

```
ad3 ctest 1
```

which invokes both the preprocessor w3adc and the compile script comp. The 1 at the end of this line activates test output. If it is omitted, this command should result in a single line of output, identifying that the routine is being processed. If ad3 works as expected, an object file obj/ctest.o is generated. If requested during the initial set up, a source code and listing file (ctest.f and ctest.l) can be found in the scratch directory. The listing file is also retained if compilation errors are detected by comp. At this time, it is prudent to test error capturing in the script comp by adding errors and warnings to ctest.ftn in the work directory. The error capturing is discussed in some detail in the documentation of comp. After comp has been tested, and the errors in ctest.ftn have been removed, the link to the work directory and the file obj/ctest.o can be deleted.

After a single routine has been compiled successfully, the next step is to try to compile and link an entire program. The grid preprocessor can be compiled by typing

```
w3_make ww3_grid
```

If the compilation appears successful, and if the input files have been installed (see above), the grid preprocessor can be tested by typing

```
ww3_grid
```

in the work directory. If the input files have been installed, a link to the input file ww3_grid.inp will be present in the work directory, and the grid preprocessor will run and send its output to the screen. Output files of the grid preprocessor will appear in the work directory. When a program is compiled for the first time, the operating system might not be able to find the executable. If this occurs, try to type

```
rehash
```

or open a new shell to work from. In this way all separate programs can be compiled and tested. To clean up all temporarily files (such as listings) and data files of the test runs, type

```
w3_clean
```

Note that w3_make only checks the switch file for changes. If the user changes the compile options in the compile and link scripts comp and link, it is advised to force the recompilation of the entire program. This can be achieved by typing

```
w3_new all or w3_new
```

before invoking w3_make. This might also be useful if the compilation is unsuccessful for no apparent reason.

Two additional remarks need to be made regarding parallel versions of the model (OpenMP and MPI versions). First, complications may occur when preparing executables for running in an MPI environment. Such complications are discussed in Appendix D. Secondly, the OpenMP code should be compiled using directives only, i.e., do not use compiler options that automatically thread the code.

5.4 Selecting model options

The file switch in the bin directory contains a set of strings identifying model options to be selected. Many options are available. Of several groups of options it is mandatory to select exactly one. These mandatory switches are described in section 5.4.1. Other switches are optional, and are described in section 5.4.2. Default model setting are identified in section 5.4.3. The order in which the switches appear in switch is arbitrary. How these switches are included in the source code files is described in section 6.2.

5.4.1 Mandatory switches

Of each of the below groups of switches exactly one has to be selected. The first group of switches controls the selection of machine-dependent code. With the introduction of FORTRAN-90 this set of switches should have become obsolete. Problems with some compilers have prompted the retention of the second switch.

F90	FORTRAN-90 style date and time capturing and program abort.
DUM	Dummy to be used if WAVEWATCH III ^{TM} is to be installed on previously untried hardware.

Hardware model (first group) and message passing protocol (second group). Note that these two groups share a switch. This implies that the MPI switch can only be used in combination with the DIST switch.

SHRD	Shared memory model.
DIST	Distributed memory model.

SHRD	Shared memory model, no message passing.
MPI	Message Passing Interface (MPI).

Word length used to determine record length in direct access files

LRB4	4 byte words.
LRB8	8 byte words.

Selection of grid type:

LLG	Spherical grid.
XYG	Cartesian grid.

Selection of propagation schemes:

PR0	No propagation scheme used.
PR1	First order propagation scheme.
PR2	ULTIMATE QUICKEST propagation scheme with Booij and Holthuijsen (1987) dispersion correction.
PR3	ULTIMATE QUICKEST propagation scheme with Tolman (2002a) averaging technique.
PRX	Experimental (user supplied).

Selection of flux computation:

FLX0	No routine used; flux computation included in source terms,
FLX1	Friction velocity according to Eq. (2.35).
FLX2	Friction velocity form Tolman and Chalikov input.
FLX3	Idem, with cap of Eq. (2.57) or (2.58).
FLXX	Experimental (user supplied).

Selection of linear input:

LN0	No linear input.
SEED	Spectral seeding of Eq. (3.63).
LN1	Cavaleri and Malanotte-Rizzoli with filter.
LNX	Experimental (user supplied).

Selection of input and dissipation:

ST0	No input and dissipation used.
ST1	WAM3 source term package.
ST2	Tolman and Chalikov (1996) source term package. See also the optional STAB2 switch.
STAB2	Enable stability correction (2.74) - (2.77) for Tolman and Chalikov (1996) source term package.
ST3	WAM4 and variants source term package.
STAB3	Enable stability correction from Abdalla and Bidlot (2002) for WAM4 source term package.

STX	Experimental (user supplied).

Selection of nonlinear interactions:

NL0	No nonlinear interactions used.
NL1	Discrete interaction approximation (DIA).
NL2	Exact interaction approximation (WRT).
NLX	Experimental (user supplied).

Selection of bottom friction:

BT0	No bottom friction used.
BT1	JONSWAP bottom friction formulation.
BTX	Experimental (user supplied).

Selection depth-induced breaking of :

DB0	No depth-induced breaking used.
DB1	Battjes-Janssen.
DBX	Experimental (user supplied).

Selection of triad interactions:

TR0	No triad interactions used.
TRX	Experimental (user supplied).

Selection of bottom scattering:

BS0	No bottom scattering used.
BSX	Experimental (user supplied).

Selection of supplemental source term:

XX0	No supplemental source term used.
XXX	Experimental (user supplied).

Selection of method of wind interpolation (time):

WNT0	No interpolation.
WNT1	Linear interpolation.
WNT2	Approximately quadratic interpolation.

Selection of method of wind interpolation (space):

WNX0	Vector interpolation.
WNX1	Approximately linear speed interpolation.
WNX2	Approximately quadratic speed interpolation.

Selection of method of current interpolation (time):

CRT1	Linear interpolation.
CRT2	Approximately quadratic interpolation.

Selection of method of current interpolation (space):

CRX0	Vector interpolation
CRX1	Approximate linear speed interpolation.
CRX2	Approximate quadratic speed interpolation.

Switch for user supplied GRIB package.

NOGRB	No package included.
NCEP1	NCEP GRIB1 package for IBM SP.
NCEP2	NCEP GRIB2 package for IBM SP.

5.4.2 Optional switches

All switches below activate model behavior if selected, but do not require particular combinations. The following switches control optional output for WAVEWATCH III ^{TM} programs.

O0	Output of namelists in grid preprocessor.
O1	Output of boundary points in grid preprocessor.
O2	Output of the grid point status map in grid preprocessor.
O2a	Generation of land-sea mask file mask.ww3 in grid preprocessor.
O2b	Output of obstruction map in grid preprocessor.
O2c	Print status map in format as read by ww3_grid.
O3	Additional output in loop over fields in field preprocessor.
O4	Print plot of normalized one-dimensional energy spectrum in initial conditions program.

o5	Id. two-dimensional energy spectrum.
o6	Id. spatial distribution of wave heights (not adapted for distributed memory).
o7	Echo input data for homogeneous fields in generic shell.
o7a	Diagnostic output for output points.
o7b	Idem in ww3_multi.
o8	Filter field output for extremely small wave heights in wave model (useful for some propagation tests).
o9	Assign a negative wave height to negative energy in wave model. Used in testing phase of new propagation schemes.
o10	Identify main elements of multi-grid model extensions in standard output.
o11	Additional log output on management algorithm in log.mww3.
o12	Identify removed boundary points in overlapping grids (center).
o13	Identify removed boundary points in overlapping grids (edge).
o14	Generate log file with buoy data buoy_log.ww3 for output type `ITYPE = 0` in ww3_outp.
o15	Generate log file with time stamps of input data file times.XXX in ww3_prep.

The following switches enable parallelization of the model using OpenMP directives, also known as 'threading'. Note that in the present version of the model, threading and parallelization using the MPI switch cannot be used simultaneously.

OMP0	High level parallelization of calls to source term and propagation subroutines.
OMP1	Parallelization of loops in output and other processing.

Furthermore the following miscellaneous switches are available:

C90	Compiler directives for Cray C90 (vectorization).
DSS0	Switch off frequency dispersion in diffusive dispersion correction.
MGG	Activate GSE alleviation correction in Eq. (E.4).
MGP	Activate propagation correction in Eq. (E.1).
MGW	Apply wind correction in moving grid approach of appendix E.
MLIM	Use Miche-style shallow water limiter of Eq. (3.64).

MPIT	Test output for MPI initializations.
MPRF	Profiling of individual models and nesting in ww3_multi.
NEC	Compiler directives for NEC SX6/SX8 (vectorization).
NCO	Code modifications for operational implementation at NCO (NCEP Central Operations). Mostly changes unit numbers and file names. Not recommended for general use.
NNT	Generate file test_data_*nnn*.ww3 with spectra and nonlinear interactions for training and testing of NNIA.
RWND	Correct wind speed for current velocity.
S	Enable subroutine tracing in the main WAVEWATCH III^{TM} subroutines by activating calls to the subroutine STRACE.
T	Enable test output throughout the program(s).
Tn	Id.
TDYN	Dynamic increment of swell age in diffusive dispersion correction (test cases only).
XW0	Swell diffusion only in ULTIMATE QUICKEST scheme.
XW1	Id. wave growth diffusion only.

5.4.3 Default model settings

The default WAVEWATCH III ^{TM} is defined by the following selection of mandatory and optional switches. Mandatory switch groups for which no option is listed here do not influence model results, and their setting is therefore irrelevant with respect to the definition of a default version of WAVEWATCH III ^{TM}. The default model is defined by using the following switches :

LLG	Longitude-latitude grid.
PR3	Propagation scheme (UQ with averaging).
LN1	Linear input term activated.
ST2	Basic source terms (Tolman and Chalikov, 1996).
STAB2	Switching on retuning and stability correction in basic source terms.
NL1	Nonlinear interactions (DIA).
BT1	Bottom friction (JONSWAP).
DB1	Depth-induced breaking (Battjes-Janssen).
MLIM	Miche-style limiter activated.

TR0	No triad interactions.
BS0	No bottom scattering.
XX0	No additional source term.
WNT1	Wind interpolation (linear).
WNX1	Wind interpolation (linear).
RWND	Define wind as relative to current.
CRT1	Current interpolation (linear).
CRX1	Current interpolation (linear).

The optional switches O8, O9, DSS0, MGP, MGW, TDYN, XW0 and XW1 modify model behavior and are *not* used in the default version of WAVEWATCH III^{TM}. All optional switches not explicitly mentioned in this section do not influence model behavior and their setting is therefore irrelevant for the definition of the default model.

Default parameter settings for all model options have been presented in chapter 2. The model will automatically default to these values, unless they are explicitly overwritten using NAMELIST input provided by the user in the input file for the grid preprocessor (ww3_grid.inp). The only parameter setting for which no default setting is given is the swell age T_s in the propagation option PR2. The value of this parameter is set to 0, and needs to be overwritten by the user to turn the corresponding GSE alleviation method on.

5.5 Modifying the source code

Source code can obviously be modified by editing the source code files in the ftn directory. However, it is usually more convenient to modify source code files from the work directory work. This can be done by generating a link between the ftn and work directories. Such a link can be generated by typing

```
ln3 filename
```

where `filename` is the name of a source code or include file, with or without its proper extension. Working from the work directory is recommended for several reasons. First, the program can be tested from the same directory, because of similar links to the input files. Secondly, links to the relevant switch, compile and link programs are also available in this directory. Third,

it makes it easy to keep track of files which have been changed (i.e., only those files to which links have been created might have been changed), and finally, source codes will not disappear if files (links) are accidentally removed from the work directory.

Modifying source codes is straightforward. Adding new switches to existing subroutines, or adding new modules requires modification of the automated compilation scripts. If a new subroutine is added to an existing module, no modifications are necessary. If a new module is added to WAVEWATCH III™, the following steps are required to include it in the automatic compilation:

1) Add the file name to sections 2.b and c of make_makefile.sh to assure that the file is included in the makefile under the correct conditions.
2) Modify section 3.b of this script accordingly to assure that the proper module dependency is checked. Note that the dependency with the object code is checked, allowing for multiple or inconsistent module names in the file.
3) Run script interactively to assure that makefile is updated.

For details of inclusion, see the actual scripts. Adding a new switch to the compilation systems requires the following actions:

1) Put switch in required source code files.
2) If the switch is part of a new group of switches, add a new 'keyword' to w3_new.
3) Update files to be touched in w3_new if necessary.
4) Update make_makefile.sh with the switch and/or keyword.

These modifications need only be made if the switch selects program parts. For test output etc., it is sufficient to simply add the switch to the source code. Finally, adding an old switch to an additional subroutine requires these actions:

1) Update files to be touched in w3_new.

If WAVEWATCH III™ is modified, it is convenient to maintain copies of previous versions of the code and of the compilation scripts. To simplify this, an archive script (arc_wwatch3) is provided. This script generates tar files that

can be reinstalled by the install program install_wwatch3. The archive files are gathered in the directory arc. The names of the archive files can contain user defined identifiers (if no identifier is used, the name will be identical to the original WAVEWATCH III ^{TM} files). The archive program is invoked by typing

```
arc_wwatch3
```

The interactive input to this script is self-explanatory. An archive file can be re-installed by copying the corresponding tar files to the WAVEWATCH III^{TM} home directory, renaming them to the file names expected by the install program, and running the install program.

5.6 Running test cases

If WAVEWATCH III ^{TM} is installed and compiled successfully, it can be tested by running the different program elements interactively from the work directory. The switch settings in the generic switch file correspond to the activated inputs in the example input files. It should therefore be possible to run all model elements by typing

```
ww3_grid | more
ww3_strt | more
ww3_prep | more
ww3_shel | more
ww3_outf | more
ww3_outp | more
ww3_trck | more
ww3_grib | more
 gx_outf | more
 gx_outp | more
```

where the `more` command is added to allow for on-screen inspection of the output. Note that `ww3_grib` will only provide GRIB output if a user-supplied packing routine is linked in. Note furthermore that no simple interactive test case for ww3_multi is provided. GrADS can then be run from the work directory to generate graphical output for these calculations. All intermediate

output files are placed in the work directory, and can be removed conveniently by typing

```
w3_clean
```

Also available are several test cases in the directory test. Example output files for selected runs are identified with the extension '.tar'. The documentation of each script identifies the preprocessor switches required to run the test case, and example outputs if available. The test cases are using the scratch directory as defined during the installation of WAVEWATCH III ^{TM}, and relevant output files will be saved in the directory test (this can easily be changed in the top of each test script). Hence, running a test case consists of five steps :

1) Look up the required switches in the documentation of the test cases, and set these switches in switch. Do not add optional switches like SEED unless specified explicitly.
2) Compile all programs by executing w3_make.
3) Execute the test script from the test directory test.
4) Check the output files in the test directory.
5) Clean up by executing w3_clean.

Note that most test cases expect that the code is compiled for a single processor using the shared memory model. The OpenMP and MPI versions of the model can also be tested with the standard test cases, and should give identical results. Converting a test case to run in parallel mode requires two modifications to the above list

1) Only the main program ww3_shel or ww3_multi should be compiled with the parallel options (see appendix D).
2) Modify the test script so that only ww3_shel runs in the proper parallel environment. Because this environment is system dependent, this option has not been build into the script.

This page is intentionally left blank.

6 System documentation

6.1 Introduction

In this chapter a brief system documentation is presented. Discussed are the custom preprocessor used by WAVEWATCH III™ (section 6.2), the contents of the different source code files (section 6.3), optimization (section 6.4), and the internal data storage (section 6.5). For a more elaborate documentation, reference is made to the source code itself, which is fully documented.

6.2 The preprocessor

The WAVEWATCH III ™ source code files are not ready to use FORTRAN files; mandatory and optional program options still have to be selected, and test output may be activated[9]. Compile level options are activated using 'switches'. The arbitrary switch 'SWT' is included in the WAVEWATCH III ™ files as comment of the form !/SWT, where the switch name SWT is followed by a space or by a '/'. If a switch is selected, the preprocessor removes the comment characters, thus activating the corresponding source code line. If '/' follows the switch, it is also removed, thus allowing the selective inclusion of hardware-dependent compiler directives etc. The switches are case sensitive, and available switches are presented in section 5.4. Files which contain the switch C/SWT can be found by typing

```
find_switch '!/SWT'
```

A list of all switches included in the WAVEWATCH III ™ files can be obtained by typing

```
all_switches
```

[9] Exceptions are some modules that are not originally part of WAVEWATCH III ™, like the exact interaction modules. Such modules with the extension .f of .f90 bypass the preprocessor and get copied to the work directory with the .f extension.

```
0 1
'w3wavemd.ftn'        'w3wavemd.f'
'F90 LRB4 SHRD NOGRB LLG PR1 ST1 NL2 BT0 WND1 CUR1'
'!docb_w3wavemd.doc'  'w3wavemd.doc'
'!docb_w3wave.doc'    'w3wave.doc'
'!docb_w3init.doc'    'w3init.doc'
'!docb_w3gath.doc'    'w3gath.doc'
'!docb_w3scat.doc'    'w3scat.doc'
'!docb_w3nmin.doc'    'w3nmin.doc'
'!docb_w3mpii.doc'    'w3mpii.doc'
'!docb_w3mpio.doc'    'w3mpio.doc'
```

Figure 6.1: Example input for W3ADC.

Pre-processing is performed by the program `w3adc`. This program is found in the file w3adc.f, which contains a ready to compile FORTRAN source code and a full documentation[10]. Various properties of `w3adc` are set in PARAMETER statements in w3adc.f, i.e., the maximum length of switches, the maximum number of include files, the maximum number of lines in an include file and the line length. `w3adc` reads its 'commands' from standard input. An example input file for `w3adc` is given in figure 6.1. Line-by-line, the input consists of

- → Test indicator and compress indicator
- → File names of the input and output code
- → Switches to be turned on in a single string (see section 5.4)
- → Multiple lines with include string identifying an include file and name of this file.

A test indicator 0 disables test output, and increasing values increase the detail of the test output. A compress indicator 0 leaves the file as is. A compress indicator 1 results in the removal of all comment lines indicated by '!', except for empty switches, i.e., lines starting with '!/'. A compress

[10] Presently still in fixed-format FORTRAN-77.

indicator 2 results in the subsequent removal of all comments. Comment lines are not allowed in this input file. The above input for W3ADC is read using free format. Therefore quotes are needed around strings. Echo and test output is send to the standard output device. To facilitate the use of the preprocessor, several UNIX scripts are provided with WAVEWATCH III ™ as discussed in section 5.3. Note that compiler directives are protected from file compression by defining them using a switch.

Note that the present version of W3ADC assumes that the computer system allows for file identification by name. If this option is not available on the system, the program is easily adapted to read by unit number.

6.3 Program files

The WAVEWATCH III ™ source code files are stored in files with the extension ftn[11]. Starting with version 2.00, the code has been organized in modules. Only the main programs are not packaged in modules. Originally, variables were bundled with the code modules, resulting in a single static data structure. In model version 3.06, a separate dynamical data structure was introduced, allow for the presence of multiple wave grids in a single program, as a preparation for the development of the the multi-grid model driver.

The subroutines contained in the modules are described in some detail below. The relation between the various subroutines is graphically depicted in Figs. 6.2 through 6.6. Three groups of codes are considered. The first are the main wave model subroutine modules, which are generally identified by the file name structure w3*xxxx*md.ftn. These modules are described in section 6.3.1. The second group consists of modules specific to the multi-grid wave model driver, which are generally identified by the file name structure wm*xxxx*md.ftn. These modules are described in section 6.3.2. The final group consists of auxiliary programs and wave model drivers, and is described in section 6.3.4. Section 6.3.3 briefly describes the data assimilation module.

[11] with the exception of some modules provided by others.

6.3.1 Wave model modules

At the core of the wave model are the wave model initialization module and the wave model module.

Main wave model initialization module — w3initmd.ftn

w3init	The initialization routine W3INIT, which prepares the wave model for computations (internal).
w3mpii	MPI initialization (internal).
w3mpio	MPI initialization for I/O (internal).
w3mpip	MPI initialization for I/O (internal, point output only).

Main wave model module — w3wavemd.ftn

w3wave	The actual wave model W3WAVE.
w3gath	Data transpose to gather data for spatial propagation in a single array (internal).
w3scat	Corresponding scatter operation (internal).
w3nmin	Calculate minimum number of sea points per processor (internal).

The main wave model routines and all other subroutines require a data structure to exist. The data structure is contained in the following modules.

Define model grids and parameter settings — w3gdatmd.ftn

w3nmod	Set number of grids to be considered.
w3dimx	Set dimensions for spatial grid and allocate storage.
w3dims	Set dimensions for spectral grid and allocate storage.
w3setg	Set pointers to selected grid.

Dynamic wave data describing sea state — w3wdatmd.ftn

w3ndat	Set number of grids to be considered.
w3dimw	Set dimensions and allocate storage.
w3setw	Set pointers to selected grid.

Auxiliary storage w3adatmd.ftn

w3naux	Set number of grids to be considered.
w3dima, w3dmnl	Set dimensions and allocate storage.
w3seta	Set pointers to selected grid.

Model output w3odatmd.ftn

w3nout	Set number of grids to be considered.
w3dmo2, w3dmo3, w3dmo5	Set dimensions and allocate storage.
w3seto	Set pointers to selected grid.

Model input w3idatmd.ftn

w3ninp	Set number of grids to be considered.
w3dimi	Set dimensions and allocate storage.
w3seti	Set pointers to selected grid.

There are presently four propagation schemes available, as well as a 'slot' for a user supplied propagation routine. The propagation schemes are packaged in the following modules.

Propagation module (first order) w3pro1md.ftn

w3map1	Generation of auxiliary maps.
w3xyp1	Propagation in physical space.
w3ktp1	Propagation in spectral space.

Propagation module (UQ scheme with diffusion) w3pro2md.ftn

w3map2	Generation of auxiliary maps.
w3xyp2	Propagation in physical space.
w3ktp2	Propagation in spectral space.

Propagation module (UQ scheme with averaging) w3pro3md.ftn

w3map3	Generation of auxiliary maps.
w3mapt	Generation of transparency maps.

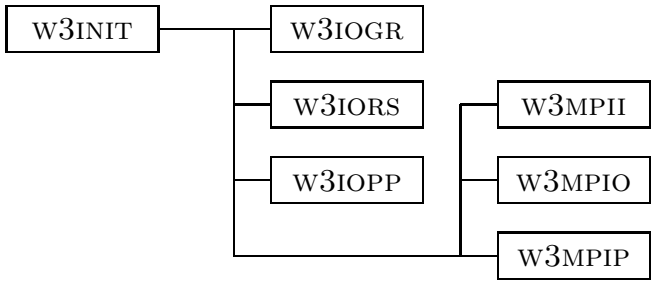

Figure 6.2: Subroutine structure for wave model initialization routine without service routines and data base management routines. Note that W3IOGR on reading data in calls all necessary initialization routines for interpolation tables and phsyics parameterizations.

---

w3xyp3	Propagation in physical space.
w3ktp3	Propagation in spectral space.

Propagation module (generic ULTIMATE QUICKEST) w3uqckmd.ftn

w3qck$n$	Routines performing ULTIMATE QUICKEST scheme in arbitrary spaces (1: regular grid. 2: irregular grid 3: regular grid with obstructions).

Propagation module (slot for user supplied routines) w3proxmd.ftn

w3xypx	Propagation in physical space.
w3ktpx	Propagation in spectral space.

The source term calculation and integration is contained in several modules. The module w3srcemd.ftn manages the general calculation and integration. Additional modules contain the actual source term options.

Source term integration module w3srcemd.ftn

w3srce	Integration of source terms.

Flux (stress) module (Wu, 1980) w3flx1md.ftn

w3flx1	Calculation of stresses.

Flux (stress) module (Tolman and Chalikov) w3flx2md.ftn

w3flx2	Calculation of stresses.

Flux (stress) module (Tolman and Chalikov, capped) w3flx3md.ftn

w3flx3	Calculation of stresses.

Flux (stress) module (slot for user supplied routines) w3flxxmd.ftn

w3flxx	Calculation of stresses.
inflxx	Initialization routine.

Linear input (Cavaleri and Malanotte Rizzoli) w3sln1md.ftn

w3sln1	Calculation S_{lin}.

Linear input (slot for user supplied routines) w3sln1md.ftn

w3slnx	Calculation S_{lin}.
inslnx	Corresponding initialization routine.

Input and dissipation module (dummy version) w3src0md.ftn

w3spr0	Calculation of mean wave parameters (single grid point).

Input and dissipation module (WAM-3) w3src1md.ftn

w3spr1	Calculation of mean wave parameters (single grid point).
w3sin1	Calculation of S_{in}.
w3sds1	Calculation of S_{ds}.

Input and dissipation module (Tolman and Chalikov, 1996) w3src2md.ftn

w3spr2	Calculation of mean wave parameters (single grid point).
w3sin2	Calculation of S_{in}.
w3sds2	Calculation of S_{ds}.

inptab	Generation of the interpolation table for β.
w3beta	Function to calculate β (internal).

Input and dissipation module (WAM-4 and variants, Ardhuin et al., 2008) w3src3md.ftn

w3spr3	Calculation of mean wave parameters (single grid point).
w3sin3	Calculation of S_{in}.
w3sds3	Calculation of S_{ds}.
tabu_stress	Tabulation of wind stress as a function of U_{10} and τ_w
tabu_tauhf	Tabulation of the short waves-supported stress
tabu_tauhf2	Tabulation of the short waves-supported stress with sheltering
tabu_swellft	Tabulation of oscillatory friction factor
calc_ustar	Computes friction velocity using stress table
inptab	Same as in w3src2md.ftn.
kzeone	Pre-calculation for Kelvin functions
kerkei	Calculation of Kelvin functions

Input and dissipation module (slot for user supplied routines) w3srcxmd.ftn

w3sinx	Calculation of S_{in}.
w3sdsx	Calculation of S_{ds}.

Nonlinear interaction module (DIA) w3snl1md.ftn

w3snl1	Calculation of S_{nl}.
insnl1	Initialization for S_{nl}.

Nonlinear interaction module (WRT) w3snl2md.ftn

w3snl2	Interface routine for S_{nl}.
insnl2	Initialization for S_{nl}.

These routines provide the interface to the WRT routines. The WRT routines are provided in the files mod_constants.f90, mod_fileio.f90, mod_xnl4v4.f90, and serv_xnl4v4.f90. For details on these files, see Van Vledder (2002b).

Nonlinear interaction module (slot for user supplied routines) w3snlxmd.ftn

w3snlx	Calculation of S_{nl}.
insnlx	Initialization for S_{nl}.

Bottom friction module (JONSWAP) w3sbt1md.ftn

w3bt1	Calculation of S_{bot}.

Bottom friction module (slot for user supplied routines) w3sbtxmd.ftn

w3sbtx	Calculation of S_{bot}.
insbtx	Initialization of S_{bot}.

Depth induced breaking module (Battjes-Janssen) w3sdb1md.ftn

w3sdb1	Calculation of S_{db}.

Depth induced breaking module (slot for user supplied routines) w3sdbxmd.ftn

w3sdbx	Calculation of S_{db}.
insdbx	Initialization of S_{db}.

Triad interactions module (slot for user supplied routines) w3strxmd.ftn

w3strx	Calculation of S_{tr}.
instrx	Initialization of S_{tr}.

Bottom scattering module (slot for user supplied routines) w3sbsxmd.ftn

w3sbsx	Calculation of S_{bs}.
insbsx	Initialization of S_{bs}.

Module for unclassified source term (slot for user supplied routines) w3sxxxmd.ftn

w3sxxx	Calculation of S_{xx}.
insxxx	Initialization of S_{xx}.

The input fields such as winds and currents are transferred to the model through the parameter list of W3WAVE. The information is processed within W3WAVE by the routines in the following module.

Input update module w3updtmd.ftn

w3ucur	Interpolation in time of current fields.
w3uwnd	Interpolation in time of wind fields.
w3uini	Generate initial conditions from the initial wind field.
w3ubpt	Updating of boundary conditions in nested runs.
w3uice	Updating of the ice coverage.
w3ulev	Updating of water levels.
w3utrn	Updating grid box transparencies.
w3ddxy	Calculation of spatial derivatives of the water depth.
w3dcxy	Calculation of spatial derivatives of the currents.

There are seven types of WAVEWATCH III ™ data files (other than the preprocessed input fields, which are part of the program shall rather than the actual wave model). The corresponding routines are gathered in six modules.

I/O module (mod_def.ww3) w3iogrmd.ftn

w3iogr	Reading and writing of mod_def.ww3.

I/O module (out_grd.ww3) w3iogomd.ftn

w3outg	Calculation of gridded output parameters.
w3iogo	Reading and writing of out_grd.ww3.

I/O module (out_pnt.ww3) w3iopomd.ftn

w3iopp	Processing of requests for point output.
w3iope	Calculating point output data.
w3iopo	Reading and writing of out_pnt.ww3.

I/O module (track_o.ww3) w3iotrmd.ftn

w3iotr	Generate track output in track_o.ww3.

I/O module (restart.ww3) w3iorsmd.ftn

w3iors Reading and writing of restart*n*.ww3.

I/O module (nest.ww3) w3iobcmd.ftn

w3iobc Reading and writing of nest*n*.ww3.

I/O module (partition.ww3) w3iofsmd.ftn

w3iofs Writing of partition.ww3.

To complete the basic wave model, several additional modules are needed. For the actual contents of the service modules see the documentation in the source code files.

constants.ftn	Physical and mathematical constants
w3arrymd.ftn	Array manipulation routines including 'print plot' routines.
w3cspcmd.ftn	Conversion of spectral discretization.
w3dispmd.ftn	Routines to solve the dispersion relation, including interpolation tables.
w3partmd.ftn	Perform spectral partitioning for a single spectrum.
w3servmd.ftn	General service routines.
w3timemd.ftn	Time management routines.

This completes the description of the basic wave model routines. The relation between the initialization routine and other routines is illustrated in Fig. 6.2. A similar relational diagram for the wave model routine is presented in Fig. 6.3.

6.3.2 Multi-grid modules

The multi-grid wave model shel ww3_multi provides a shell around the basic wave model as described in the previous section. This shell manages the side-by-side running of multiple wave model grids, and all communication between the grids. To achieve this various additional modules have been developed. At the core are the initialization, multi-grid model and finalization routines.

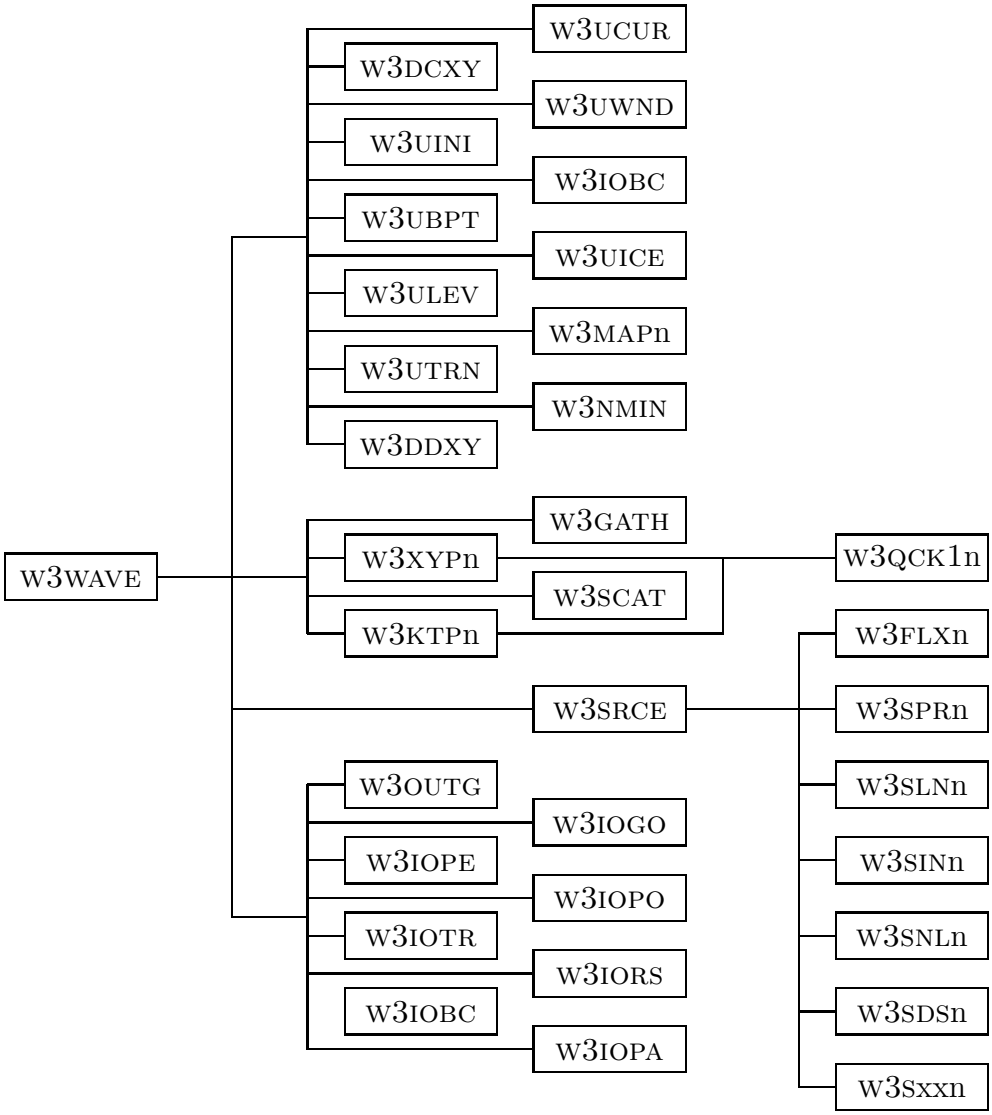

Figure 6.3: Subroutine structure for wave model routine without service routines and without routines managing the data structures. W3SXXN represents all additional shallow water source term routines.

Initialization of multi-grid model wminitmd.ftn

wminit	Multi-grid model initialization.

Running of multi-grid model wmwavemd.ftn

wmwave	Multi-grid model execution.
wmprnt	Printing to log file.
wmbcst	Non-blocking MPI broadcast.
wmwout	Idem.

Finalizing of multi-grid model wmfinlmd.ftn

wmfinl	Multi-grid model finalization.

These routines are designed to become part of a coupled model. For the structure of in particular the actual wave model routine, reference is made to Tolman (2007). The resulting wave model driver ww3_multi consequently becomes extremely simple; it initializes the MPI environment, and then calls the above three modules consecutively.

The main multi-grid wave model routines require an expansion of the data structure used by WAVEWATCH III ^{TM}. Furthermore, main activities are gathered in subroutines in various modules.

Data storage wmmdatmd.ftn

wmndat	Set number of grids to be considered.
wmdimd, wmdimm	Set dimensions and allocate storage.
wmsetm	Set pointers to selected grid.

Determine grid relations wmgridmd.ftn

wmglow	Relations to lower ranked grids.
wmghgh	Relations to higher ranked grids.
wmgeql	Relations between equal ranked grids.
wmrspc	Determine need for spectral conversion between grids.

Update model input wmupdtmd.ftn

wmupdt	General input update routine.
wmupd1	Update input from native files using w3fldsmd.ftn from section 6.3.4.
wmupd2	Update input from pore-defined input grids.
wmupdv	Update vector fields.
wmupds	Update scalar fields.

Perform internal communications wminiomd.ftn

wmiobs	Stage internal boundary data.
wmiobg	Gather internal boundary data.
wmiobf	Finalize WMIOBS (MPI only).
wmiohs	Stage internal high to low rank data.
wmiohg	Gather internal high to low rank data.
wmiohf	Finalize WMIOHS (MPI only).
wmioes	Stage internal data between equal ranked grids.
wmioeg	Gather internal data between equal ranked grids.
wmioef	Finalize WMIOES (MPI only).

Unify point output to single file wmiopomd.ftn

wmiopp	Initialization routine.
wmiopo	Data gather and write routine (using W3IOPO in w3iopomd.ftn).

To complete the multi-grid wave model, one additional service module is needed. For the actual contents of the service module see the documentation in the source code files.

wmunitmd.ftn Dynamic unit number assignment

6.3.3 Data assimilation module

WAVEWATCH III [TM] includes a data assimilation module that can work in conjunction with the main wave model routine, and is integrated in the generic program shell. The module is intended as an interface to a data assimilation package to be provided by the user.

Data assimilation module w3wdasmd.ftn

w3wdas	Data assimilation interface.

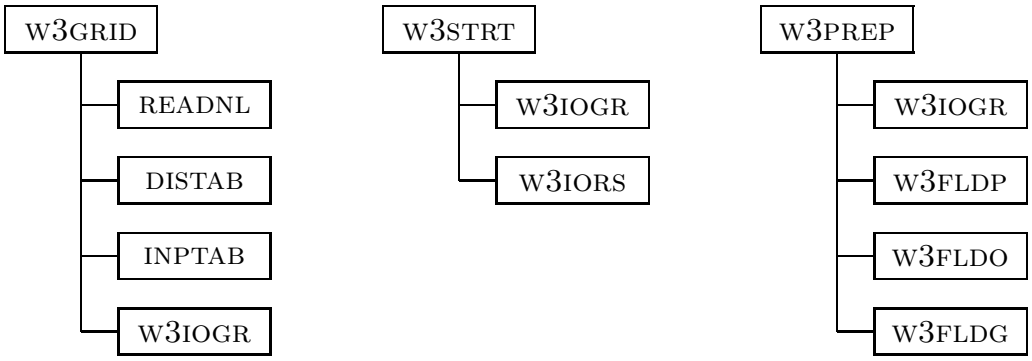

Figure 6.4: Subroutine structure of pre-processors

6.3.4 Auxiliary programs

WAVEWATCH III ^{TM} has several auxiliary pre- and post-processors, and two wave model shells (see section 4.4). These main programs and some additional routines are stored in the following files. Generally, subroutines used only by the programs are stored as internal subroutines with the main program. There is no need for using the module structure in this case. The exception is an additional module w3fldsmd.ftn which deals with the data flow of input fields for the wave model between the field pre-processor and the stand-alone model shell. The latter module does not have any explicit WAVEWATCH III ^{TM} dependencies, and can therefore be integrated in any custom data pre-processor.

Input data file management module w3fldsmd.ftn

w3fldo	Opening and checking of data files for W3SHEL.
w3fldg	Reading and writing of data files for W3SHEL (model input).
w3fldd	Reading and writing of data files for W3SHEL (data assimilation).
w3fldp	Prepare interpolation of input fields from arbitrary grids.
w3fldh	Management of homogeneous input fields in W3SHEL.
w3fldm	Process moving grid data in W3SHEL.

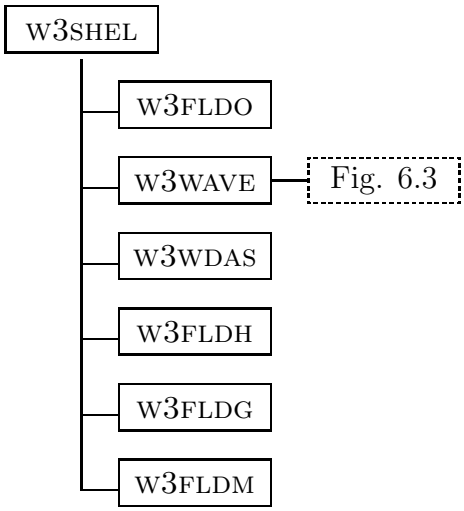

Figure 6.5: Subroutine structure for the generic program shell.

Grid pre-processing program ww3_grid.ftn

w3grid	The grid preprocessor.
readnl	Reading NAMELIST input (internal).

Initial conditions program ww3_strt.ftn

w3strt	The initial conditions program.

Input field pre-processing program ww3_prep.ftn

w3prep	Pre-processor for the input fields for the generic shell.

Generic wave model program ww3_shel.ftn

w3shel	The generic program shell.

Generic wave model program ww3_multi.ftn

w3mlti	The multi-grid program shell.

Gridded data post-processing program ww3_outf.ftn

w3outf	The post-processing program for gridded fields of mean wave parameters.
w3exgo	Actual output routine (internal).

Point post-processing program ww3_outp.ftn

w3outp	The post-processing program output at selected locations.
w3expo	Actual output routine (internal).

Track output post-processing program ww3_trck.ftn

w3trck	Converting unformatted direct access track output file to integer-packed formatted file.

Gridded data post-processing program (GRIB) ww3_grib.ftn

w3grib	The post-processing program for generating GRIB files.
w3exgb	Actual output routine (internal).

Gridded data post-processing program (GrADS) gx_outf.ftn

gxoutf	The post-processing program for converting gridded fields of mean wave parameters to input files for GrADS.
gxexgo	Actual output routine (internal).

Point post-processing program (GrADS) gx_outp.ftn

gxoutp	The post-processing program for converting output at selected locations to input files for GrADS.
gxexpo	Actual output routine (internal).

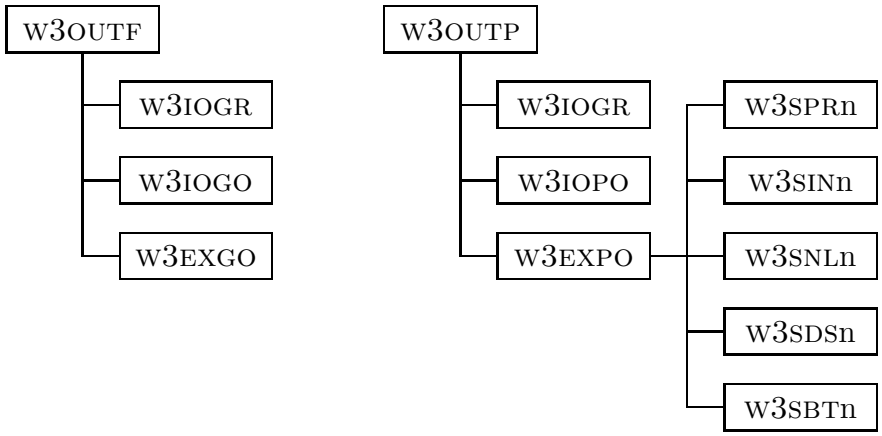

Figure 6.6: Subroutine structure for the postprocessors. The structure of the GRIB postprocessor is similar to W3OUTF, replacing W3OUTF and W3EXGO with W3GRIB and W3EXGB, respectively. The structure of the GrADS postprocessors is similar, replacing W3OUTF, W3OUTP, W3EXGO and W3EXPO with GXOUTF, GXOUTP, GXEXGO and GXEXPO, respectively

6.4 Optimization

The source code of WAVEWATCH III ™ is written in ANSI standard FORTRAN 90, and has been compiled and run on a variety of platforms ranging from PC's to supercomputers.

Optimization for vector computers has been performed by structuring the code in long vector loops where possible. Optimization was originally performed for the Cray YMP and C90. Note that some compiler directives for vectorization have been used. Note also that the vector optimization has not been updated since about 1997, and therefore needs to be revisited if the model is implemented on a vector machine. Vectorization directives are activated by the corresponding preprocessor switch (C90).

Parallelization for shared memory machines using threading has been implemented using standard OpenMP directives. Such parallelization takes place mainly in the loop calling the source term routine W3SRCE and the different propagation routines. OpenMP directives are activated by the corresponding preprocessor switches (OMPn).

Parallelization for distributed memory machines is discussed in some detail in section 6.5.2.

Note that an important part of the optimization is the use of interpolation tables for the solution of the dispersion relation and for the calculation of the wind-wave interaction parameter).

6.5 Internal data storage

The remainder of this chapter will deal with the internal data storage used by WAVEWATCH III ^{TM}. In section 6.5.1 the layout of a single wave model grid as used in ww3_shel is discussed. In section 6.5.2 the parallelization approaches for a single grid are discussed. In section 6.5.3 the simultaneous storage of multiple wave grids is discussed. Finally, the actual wave model variables are described in section 6.6. Note that the code is fully documented, including the variables defining the data storage.

6.5.1 Grids

For convenience and economy of programming, spatial and spectral grids are considered separately. This approach is inspired by the splitting technique described in chapter 3. For spatial propagation, a simple 'rectangular' spatial grid is used, as is illustrated in Fig. 6.7. The grid can either be a Cartesian '(x, y)' grid or a spherical grid. In a spherical grid, the longitudes are denoted throughout the program by the counter IX, and latitudes by the counter IY, and the corresponding grid dimensions (NX,NY). All spatial field arrays are dynamically allocated within the code, corresponding work arrays are usually automatic, to allow for thread-safe code. The closure of the grid in case of a global applications is handled within the model, and does not require user intervention. To simplify the calculation of derivatives of in particular the current, the outer grid points (IX=1,NX, unless the grid is global) and (IY=1,NY) will be considered as land points, inactive points or active boundary points. The minimum grid size therefore is NX=3, NY=3. Input arrays are typically assumed to be of the form

ARRAY(NX,NY) ,

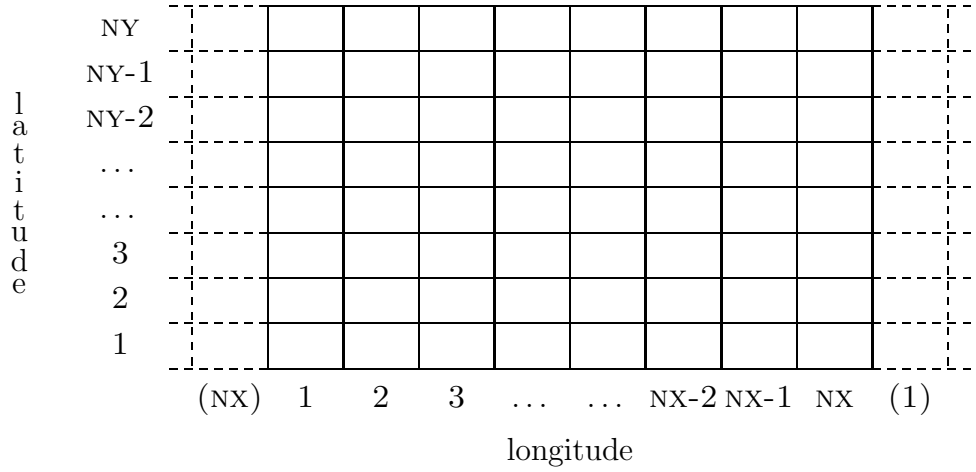

Figure 6.7: Layout of the spatial grid. Grid points are denoted as boxes, dotted boxes denoted repeated collumns for global model applications.

and are read row by row (see also chapter 4). Within the program, however, they are typically stored with rotated indices

$$\text{ARRAY(NY,NX)} \ .$$

This makes it easier to provide global closure, which typically requires extension of the x axis. Furthermore, such two-dimensional array are usually treated as one-dimensional arrays, to increase vector lengths. The array ARRAY, its one-dimensional equivalent VARRAY and IXY are defined as

$$\text{ARRAY(MY,MX)} \ , \ \text{VARRAY(MY*MX)} \ ,$$
$$\text{IXY} = \text{IY} + (\text{IX-1})*\text{MY} \ .$$

Note that this representation of the grid is used *internally* within the model only.

The spectral grid for a given spatial grid point (IX,IY) is defined similarly, using a directional counter ITH and a wavenumber counter IK (Fig. 6.8). The size of the spectral grid is set using dynamic allocation. As with the spatial grid, the internal description of the spectrum A is defined as

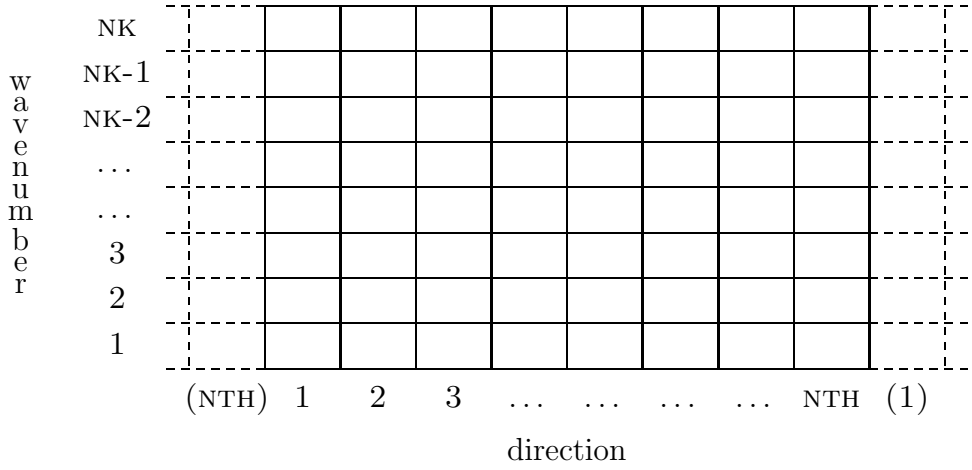

Figure 6.8: Layout of the spectral grid. Dotted boxes denoted repeated collumns for directional closure.

A(NTH,NK) ,

and equivalent one-dimensional arrays are used throughout the program. Inside the model, directions are always Cartesian, $\theta = 0^\circ$ corresponds to propagation from east to west (positive x or IX direction), and $\theta = 90^\circ$ corresponds to propagation from south to north (positive y or IY direction). Output directions use other conventions, as is discussed in chapter 4.

The storage of the wave spectra accounts for the majority of the memory required by the model, because the splitting technique used assures that any part of the model operates on a small subset of the entire wave field. To minimize the amount of memory needed, only spectra for actual sea points are stored. Sea points are here defined as points where spectra are potentially needed. This includes active boundary points, and sea points covered by ice. For archiving purposes, a one-dimensional sea point grid is defined using the counter ISEA. Spectra are then stored as

A(ITH,IK,ISEA) .

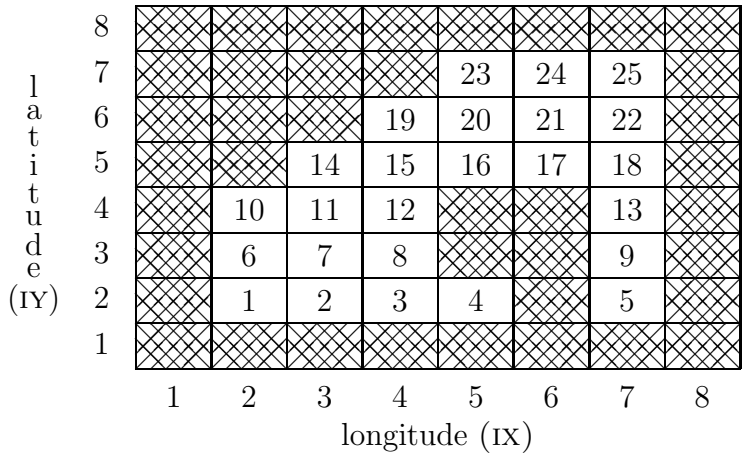

Figure 6.9: An example of the onedimensional storage grid for spectra. Hatched grid boxes denote land points. Numbers within the grid boxes show the grid counter ISEA of the storage grid.

An example of the layout of this storage grid in relation to the full grid of Fig. 6.7 is given in Fig. 6.9. Obviously, the relation between the storage grid and the full spatial grid requires some bookkeeping. For this purpose, two 'maps' MAPFS and MAPSF are defined.

$$\begin{aligned} \text{MAPSF}(\text{ISEA},1) &= \text{IX} \ , \\ \text{MAPSF}(\text{ISEA},2) &= \text{IY} \ , \\ \text{MAPSF}(\text{ISEA},3) &= \text{IXY} \ , \\ \text{MAPFS}(\text{IY},\text{IX}) &= \text{VMAPFS}(\text{IXY}) = \text{ISEA} \ , \end{aligned}$$

where $\text{MAPFS}(\text{IY},\text{IX}) = 0$ for land points. Finally, status maps $\text{MAPSTA}(\text{IY},\text{IX})$ and $\text{MAPST2}(\text{IY},\text{IX})$ are maintained to identify sea, land, active boundary and ice points. MAPSTA represents the main status map for the grid;

$\text{MAPSTA}(\text{IY},\text{IX}) = 0$	for excluded points,
$\text{MAPSTA}(\text{IY},\text{IX}) = 1$	for sea points,
$\text{MAPSTA}(\text{IY},\text{IX}) = 2$	for active boundary points.

Sea points and active boundary point which are not considered in the wave model due to the presence of ice are marked by their corresponding negative status indicator (-1 or -2). MAPST2 contains secondary information. For excluded points MAPSTA)IY,IX) = 0, this map distinguished between land points MAPST2(IY,IX) = 0 and otherwise excluded points MAPST2(IY,IX) = 1. For sea points that are disabled MAPSTA(IY,IX) $<$ 0, consecutive bits in MAPST2 identify the reason for deactivation (bit value 1 indicating deactivation).

bit	identifies
1	Ice coverage
2	Point dried out
3	Land in moving grid or inferred in nesting
4	Masked in two-way nesting

Two additional considerations have been made. First, the two status maps can be collapsed into a single map for storage. To assure that the storage is backward compatible with the previous mode version, the two maps are combined into a single map MAPTMP

$$\text{MAPTMP} = \text{MAPSTA} + 8 * \text{MAPST2}$$

considering that only the first few bits of MAPSTA contain data. the original maps can be recovered as

$$\text{MAPSTA} = \text{MOD} (\text{MAPTMP} + 2 , 8) - 2$$
$$\text{MAPST2} = \text{MAPTMP} - \text{MAPSTA}$$

Second, a single map is used in the graphics output program, to simplify the plotting of the status of grid points. In the graphics files, the map is defined as

map	implies
2	Active boundary point
1	Active sea point
0	Land point (including as identified in MAPST2.
-1	Point covered by ice, but wet.
-2	Dry point, not covered by ice.
-3	Dry point covered by ice.
-4	Point masked in the two-way nesting scheme.
-5	Other disabled point.

Similarly, a single map can be used to simplify processing in the grid preparation program ww3_grid. In this map a distinction is made between points as follows:

map	implies
3	Excluded points
2	Active boundary point
1	Active sea point
0	Land point

6.5.2 Distributed memory concepts.

The general grid structure described in the previous paragraph is used for both shared and distributed memory versions of the model, with some minor differences. For the distributed memory version of the model, not all data is kept at each processor. Instead, each spectrum is kept at a single processor only. The spectra on the storage grid are distributed over the available processors with a constant stride. Because only part of the spectra are stored locally on a given processor, a distinction needs to be made between the above global sea point counter ISEA, and the local sea point counter JSEA. If the actual number of processors used in the computation is NAPROC, and if IAPROC is the processor number ranging form 1 to NAPROC, these parameters are related in the following way

$$\begin{aligned} \text{ISEA} &= \text{IAPROC} + (\text{JSEA-1})\ \text{NAPROC} \ , \\ \text{JSEA} &= 1 + (\text{ISEA-1})\ /\ \text{NAPROC} \ , \\ \text{IAPROC} &= 1 + \text{MOD}(\text{ISEA-1},\text{NAPROC}) \ . \end{aligned}$$

In model version 3.10, a further refinement was introduced. The actual number of processors NAPROC can be smaller than the total number of processors used by the program (NTPROC). Processors where NAPROC < IAPROC ≤ NTPROC are reserved for output processing only.

With this data distribution, source terms and intra-spectral propagation can be calculated at the each given processor without the need for communication between processors. For spatial propagation, however, a data transpose is required where the spectral components (ITH,IK) for all spatial grid points have to be gathered at a single processor. After propagation has been performed, the modified data have to be scattered back to their 'home' processor. Individual spectral components are assigned to specific processors in such a way that the number of partial propagation steps to be performed by each processor is roughly identical. This makes a good load balance possible. The actual algorithm can be found in section 4.d of the subroutine W3INIT (w3initmd.ftn).

The data transpose for the gather operation is implemented in two steps using the Message Passing Interface (MPI) standard (e.g. Gropp et al., 1997). First, values for each spatial grid point for a given spectral bin (ITH,IK) are gathered in a single target processor in a one-dimensional array STORE(ISEA), which then is converted to the full two-dimensional field of spectral components. After propagation has been performed, the transpose for the scatter operation reverses this process, using the same one-dimensional array STORE. Whereas the algorithm for distributing spatial propagation over individual processors assures a global (per time step) load balance, it does not assure that communication is synchronized, because not each calculation at each processor will take the same effort. To avoid that this results in a load imbalance, non-blocking communication has been used. Furthermore, the one-dimensional array STORE(ISEA) is replaced by STORE(ISEA,IBUF), where the added dimension of the array supplies an actively managed buffer space (see W3GATH and W3SCAT in w3wavemd.ftn). These buffers allow that spare clock cycles as may occur during communication can be used for calculation, and that hiding of communication behind calculation will occur if the hardware is capable of doing this. To avoid problems with incompatibilities between FORTRAN and MPI, separate gather and scatter data arrays are used. The buffered data transposes are graphically depicted in Fig. 6.10. More details can be found in Tolman (2002b)

In principle only the storage array A(ITH,IK,JSEA) is influenced by the data distribution. Input fields, maps and output fields of mean wave param-

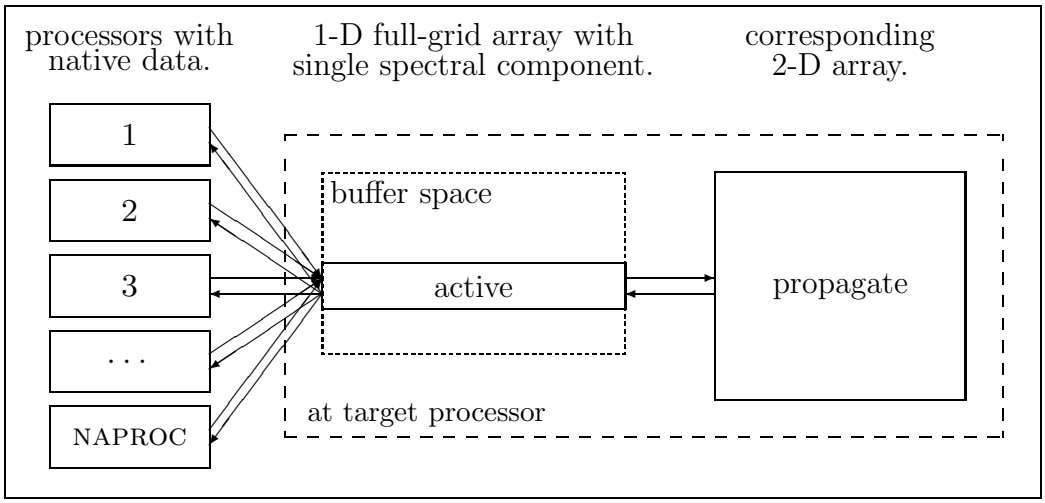

Figure 6.10: Data transpose in distributed memory model version. First, the data is moved from left to right in the figure during the gather operation. After the calculation is performed, the data is moved from right to left in the scatter operation.

---

eters in principle are retained at full resolution at each grid point. Full maps are available at each processor at each phase of the calculation. Input and output fields generally contain pertinent data at the stride NAPROC only.

Distributed memory also requires modifications to the I/O. Input files are read completely by each separate processor. The type of file output is determined by the I/O type indicator IOSTYP

IOSTYP	implies
0	Restart file written from each individual process.
1	Each file written from assigned process.
2	Each file written from a single dedicated output process.
3	Dedicated output processes for each output type.

Note that the restart file is a direct access file, so that each processor can efficiently gather only the locally stored spectra, without the need of reading through the entire file. The restart file is either written by each individual process directly, or all data is funneled through a dedicated processor. The first method requires a parallel file system, the second method is generally applicable.

The present algorithm for data distribution has been chosen for several reasons. First, it results in an automatic and efficient load balancing with respect to the (dynamic) integration of source terms, the exclusion of ice covered grid points, and of intra-spectral propagation. Secondly, the communication by definition becomes independent of the numerical propagation scheme, unlike for the more conventional domain decomposition. In the latter case, only a so-called 'halo' of boundary data needs to be converted to neighboring 'blocks' of grid points. The size of the halo depends on the propagation scheme selected. The main disadvantage of the present data distribution scheme is that the amount of data to be communicated each time step is much larger than for a more conventional domain decomposition, particularly when relatively small numbers of processors are used. On an IBM RS6000 SP, on which the distributed memory version of WAVEWATCH III ™ was tested, the relatively large amount of communication did not constitute a significant part of the overall time of computation, and the model shows excellent scaling behavior for up to O(100) processors (Tolman, 2002b).

6.5.3 Multiple grids

So far, only a single wave model grid has been considered. To make it possible to run several model grids in a single program, a data structure needs to be devised in which all different model grids and internal work arrays for all models are retained simultaneously, with a simple mechanism to choose the actual wave model grid to work on. In order to achieve this, some FORTRAN 90 features (e.g., Metcalf and Reid, 1999) are used in the following way:

1) Define one or more data structures in the model code that contain the model setup and relevant work arrays, using a TYPE declaration.

2) Construct arrays of these data structures, with each element of the array defining a separate model grid.

3) Redefine the basic parameters describing the model such as the number of grid points NX and NY as pointers, and point these to the proper element of the proper data structures to generate instantaneous aliases.

```
!/
!/ Data structures
!/
      TYPE GRID
        INTEGER                :: NX, NY, NSEA
        REAL, POINTER          :: ZB(:)
      END TYPE GRID
!/
!/ Data storage
!/
      TYPE(GRID), TARGET, ALLOCATABLE :: GRIDS(:)
!/
!/ Data aliasses
!/
      INTEGER, POINTER         :: NX, NY, NSEA
      REAL, POINTER            :: ZB(:):
!/
```

Figure 6.11: Example of the data structure declarations used in w3gdatmd.ftn to define multiple spatial grids in the wave model. For simplicity, the example considers only the grid dimensions NX, NY and NSEA, and the bottom depth array ZB.

```
!
      NX      => GRIDS(IMOD)%NX
      NY      => GRIDS(IMOD)%NY
      NSEA    => GRIDS(IMOD)%NSEA
!
      ZB      => GRIDS(IMOD)%ZB
!
```

Figure 6.12: Example of the source code used to activate the pointer aliases in Fig. 6.11 for the model number IMOD.

In this way it is possible to define a multi-model data structure, while keeping the layout of all original variables describing the model unchanged inside the model subroutines. Such a structure and its usage are illustrated in Figs. 6.11 and 6.12 with an example from the actual source code. Note that the pointer arrays like ZB inside the structures are assigned memory as

ALLOCATE GRIDS(IMOD)%ZB(NSEA)

After this statement, the alias pointer ZB again needs to be pointed to the proper element of the structure for this alias to properly point to the newly allocated space. For this reason, the subroutine W3DIMX, which allocates the arrays in this structure, includes at the end a call to the subroutine W3SETX, which in turn sets all pointer aliases for the selected grid. The same is true for other subroutines setting array sizes in other structures.

6.6 Variables in modules

Below, most PUBLIC and PRIVATE variables in modules are described briefly (full documentation can also be found in the source code). The file name of the module is given at the right margin of the start of each list. The second column of each list identifies the type of the variable. I, R, L and C represent integer, real, logical and character, A identifies an array, and P

identifies a PARAMETER declaration. All variables are public, unless marked with *. First, parameter settings in various modules are discussed. After that, the contents of the main storage modules is discussed.

6.6.1 Parameter settings in modules

Several modules has internally used parameter settings. Here only parameter settings that are generally usable or impact model output are presented.

Physical and mathematical constants : constants.ftn

GRAV	RP	Acceleration of gravity g.	(m s^{-2})
DWAT	RP	Density of water.	(kg m^{-3})
DAIR	RP	Density of air.	(kg m^{-3})
PI	RP	π.	
TPI	RP	2π.	
HPI	RP	0.5π.	
TPIINV	RP	$(2\pi)^{-1}$.	
HPIINV	RP	$(0.5\pi)^{-1}$.	
RADE	RP	Conversion factor from radians to degrees.	
DERA	RP	Conversion factor from degrees to radians.	
RADIUS	RP	Radius of the earth.	(m)
G2PI3I	RP	$g^{-2}(2\pi)^{-3}$.	
G1PI1I	RP	$g^{-1}(2\pi)^{-1}$.	

Wave model initialization module : w3initmd.ftn

CRITOS	RP	Critical fraction of resources used for output only (triggers warning output).
WWVER	CP	Version number of the main program.

I/O module (mod_def.ww3) : w3iogrmd.ftn

VERGRD	CP*	Version number of file mod_def.ww3.
IDSTR	CP*	ID string for file.

I/O module (out_grd.ww3) : w3iogomd.ftn

VEROGR	CP*	Version number of file out_grd.ww3.

IDSTR CP* ID string for file.

I/O module (out_pnt.ww3) : w3iopomd.ftn

VEROPT CP* Version number of file out_pnt.ww3.
IDSTR CP* ID string for file.

I/O module (track_o.ww3) : w3iotrmd.ftn

VERTRK CP* Version number of file track_o.ww3.
IDSTRI CP* ID string for file track_i.ww3.

I/O module (restart.ww3) : w3iorsmd.ftn

VERINI CP* Version number of file restart.ww3.
IDSTR CP* ID string for file.

I/O module (nest.ww3) : w3iobcmd.ftn

VERBPT CP* Version number of file nest.ww3.
IDSTR CP* ID string for file.

I/O module (partition.ww3) : w3iosfmd.ftn

VERTRT CP* Version number of file partition.ww3.
IDSTR CP* ID string for file.

Multi-grid model input update : wmupdtmd.ftn

SWPMAX IP Maximum number of extrapolation sweeps allowed to make maps match in conversion from input from input grid to wave model grid.

The service modules contain private variables only, with the exception of the interpolation tables for the solution of the dispersion relation

Solving the dispersion relation : w3dispmd.ftn

NAR1D IP Dimension of interpolation tables.
DFAC RP Maximum nondimensional water depth kd.

ECG1	RA	Table for calculating group velocities from the frequency and the depth.
EWN1	RA	Id. wavenumbers.
N1MAX	I	Largest index in tables.
DSIE	R	Nondimensional frequency increment.

Automatic unit number assignment : wmunitmd.ftn

UNITLW	IP	Lowest unit number to be considered.
UNITHG	IP	Highest unit number to be considered.
INPLOW, INPHGH	IP	Range of input file unit numbers.
OUTLOW, OUTHGH	IP	Range of output file unit numbers.
SCRLOW, SCRHGH	IP	Range of scratch file unit numbers.

Note that the main programs only contain locally defined variables, which need not be documented here in detail.

6.6.2 Data in W3GDATMD.

The module W3GDATMD in the file w3gdatmd.ftn contains the data describing a set of models. First several variables and parameters embedded in the module are defined:

NGRIDS	I	Number of models for which space is available.
NAUXGR	I	Auxiliary grids (model input, unified point output).
IGRID	I	Number of model/grid presently selected.
ISGRD	I	Number of spectral model grid presently selected.
IPARS	I	Number of physical model parameter settings presently selected.
FLAGLL	LP	Flag for spherical grid (otherwise Cartesian).
GRID	T	Data structure containing information of spatial grids.
GRIDS	TA	Array if type GRID and size NGRIDS.
SGRD	T	Data structure containing information of spectral grids.
SGRDS	TA	Array if type SGRD and size NGRIDS.

MPAR	T	Data structure containing all other model parameters in sub structures (see below).
MPARS	TA	Array if type MPAR and size NGRIDS.

For each element of the type GRID, direct aliases are defined with the same name, as illustrated in Figs. 6.11 and 6.12. These variables are:

NX,NY	I	discrete grid dimensions, NX, NY ≥ 3.	
NSEA	I	Number of sea points in grid.	
NSEAL	I	Id. locally stored on present process.	
TRFLAG	I	Type of transparencies used.	
MAPSTA	IA	Grid status map.	
MAPST2	IA	Secondary grid status map.	
MAPSF	IA	Storage grid map.	
MAPFS	IA	Id.	
SX,SY	R	Spatial grid increments.	(° or m)
X0,Y0	R	Lower left point of spatial grid.	(° or m)
DTCFL	R	Maximum CFL number for spatial propagation.	
DTCFLI	R	Id. for intra-spectral propagation.	
DTMAX	R	Maximum overall time step.	
DTMIN	R	Minimum source term integration time step.	
DMIN	R	Minimum water depth.	(m)
CTMAX	R	Maximum CFL number for depth refraction.	
FICE0/N	R	Ice concentration cut-off.	(-)
PFMOVE	R	Power factor in GSE alleviation correction for moving grid.	
ZB	RA	Bottom depths on storage grid.	(m)
CLAT(I)	RA	(Inverse) cosine of latitude.	
CLATS	RA	Id.	
CTHG0	RA	Latitude factor in great circle propagation speed.	
TRNX/Y	RA	Grid box transparencies.	(–)
GINIT	L	Flag for initialization of model/grid.	
GLOBAL	L	Flag for global grid.	
FLDRY	L	Flag for 'dry run' (no calculations).	
FLC*xx*	L	Flags for propagation in all spaces.	
FLSOU	L	Flag for source term integration.	
FLAST	LA	Flags for source term computation per grid point.	
GNAME	C	Grid name.	

FILEXT	C	File extension for raw files for this model/grid.

Similarly, the structure SGRD contains the following variables and/or aliases:

NK	I	Number of discrete spectral wavenumbers, NK ≥ 3.
NK2	I	Extended wavenumber range.
NTH	I	Number of discrete spectral directions, NTH ≥ 4.
NSPEC	I	Number of spectral bins.
MAPWN	IA	Map with discrete wavenumber for the one dimensional description of the spectrum.
MAPTH	IA	Id. for discrete directions.
DTH	R	Spectral directional increment. (rad)
XFR	R	Factor defining discrete frequency increment.
FR1	R	Lowest discrete frequency. (Hz)
FTE	R	Factor in tail integration of total energy.
FTF	R	Id. mean frequency.
FTWN	R	Id. mean wavenumber.
FTTR	R	Id. mean period.
FTWL	R	Id. mean wave length.
FACTIn	R	Auxiliary to calculate discrete frequency from continuous frequency.
FACHFA	R	Factor defining parametric tail for the action spectrum $N(k,\theta)$.
FACHFE	R	Id. for the energy spectrum $F(f)$.
TH	RA	Spectral directions. (rad)
ESIN	RA	$\sin(\theta)$ for discrete spectral directions.
ECOS	RA	Id. $\cos(\theta)$.
ES2	RA	$\sin^2(\theta)$ for entire spectrum.
ESC	RA	Id. $\sin(\theta)\cos(\theta)$.
EC2	RA	Id. $\cos^2(\theta)$.
SIG	RA	Frequencies for discrete wavenumbers. (rad s^{-1})
SIG2	RA	Id. for entire discrete spectrum.
DSIP	RA	Frequency band widths for each wavenumber as used in propagation. (rad s^{-1})
DSII	RA	Id. for spectral integration.
DDEN	RA	Composite band with and conversion to energy for each wavenumber (DDEN = DTH * DSII * SIG). (rad s^{-1})

DDEN2	RA	Id. for entire spectrum.
SINIT	L	Flag for initialization of spectral grid.

The structure MPAR contains addition structures and a single aliased variable:

PINIT	L	Flag for initialization of this structure.
NPARS	T	Structure with numerical parameters for source term integration (structure NPAR).
PROPS	T	Structure with parameters for propagation schemes (structure PROP).
SFLPS	T	Structure with parameters for flux computation (structure SFLP).
SLNPS	T	Structure with parameters linear wind input source term (structure SLNP).
SRCPS	T	Structure with parameters for input and dissipation source term (structure SCRP).
SNLPS	T	Structure with parameters for nonlinear interaction source term (structure SNLP).
SBTPS	T	Structure with parameters for bottom friction source term (structure SBTP).
SDBPS	T	Structure with parameters for depth-induced breaking term (structure SDBP).
STRPS	T	Structure with parameters for triad interaction source term (structure STRP).
SBSPS	T	Structure with parameters for bottom scattering source term (structure SBSP).
SXXPS	T	Structure with parameters for arbitrary additional source term (structure SXXP).

The structure NPAR contains the following alias pointers:

FACP	R	Composite constant for parametric cut-off.
XREL	R	X_r in dynamic integration.
XFLT	R	X_f in dynamic integration.
FXFM	R	First constant in tail.
FXPM	R	Second constant in tail.
XFT	R	Constant for f_2 in tail.
XFC	R	Constant for f_{hf} in tail.

FACSD	R	Seeding constant X_{seed}.	
FHMAX	R	Maximum H_s/d ratio in shallow water limiter.	

The structure PROP contains the following alias pointers. All pointer are activated by the switches on the right.

DTME	R	Swell age in disp. corr.	(!/PR2)
CLATMN	R	Id. minimum cosine of lat.	(!/PR2)
WDCG	R	Factors in width of av. Cg.	(!/PR3)
WDTH	R	Factors in width of av. Th.	(!/PR3)

The structure SFLP contains the following alias pointer, presented as above.

NITTIN	I	Number of iterations for drag calculation.	(!/FLX2, !/FLX3)
CINXSI	R	Constant in parametric description of drag.	(!/FLX2, !/FLX3)
CAP_ID	I	Type of cap applied to C_d.	(!/FLX3)
CD_MAX	R	Maximum value for C_d.	(!/FLX3)

The structure SLNP contains the following alias pointer, presented as above.

SNLC1	R	Proportionality and other constants.	(!/LN1)
FSPM	R	Factor for f_{PM} in filter.	(!/LN1)
FSHF	R	Factor for f_h in filter.	(!/LN1)

The structure SRCP contains the following alias pointers, with the associated switches again displayed on the right.

SINC1	R	Combined constant for input.	(!/ST1)
SDSC1	R	Combined constant for dissipation.	(!/ST1)
ZWIND	R	Height at which the wind is defined. (m)	(!/ST2)
FSWELL	R	Reduction factor of negative input for swell.	(!/ST2)
SHSTAB, OFSTAB, CCNG, CCPS, FFNG, FFPS	R	Factors in effective wind speed.	(!/ST2)
CDSAn	R	Constants in high-freq. dissipation.	(!/ST2)
SDSALN	R	Factor for nondimensional 1-D spectrum.	(!/ST2)
CDSBN	R	Constants in parameterization of ϕ.	(!/ST2)
FPIMIN	R	Minimum value for $f_{p,i}$.	(!/ST2)

XFH	R	Constant for turbulent length scale.		(!/ST2)
XFN	R	Constants in combining low and high frequency dissipation.		(!/ST2)
ZWND	R	Height at which the wind is defined.	(m)	(!/ST3)
AALPHA	R	Minimum value of Charnock coefficient.		(!/ST3)
BBETA	R	Wind-wave growth $\beta_{\max}$ parameter.		(!/ST3)
ZZALP	R	Wave age correction z_α in wind input.		(!/ST3)
TTAUWSHELTER	R	Sheltering coefficient for input to short waves.		(!/ST3)
SSWELLFPAR	I	Choice of negative input parameterization.		(!/ST3)
SSWELLF	R	Reduction factor of negative input for swell.		(!/ST3)
SSWELLF2	R	Extra parameter for negative input.		(!/ST3)
SSDSC1	R	Constant for WAM4-part of dissipation.		(!/ST3)
DDELTA1	R	Weight of k part in WAM4 dissipation.		(!/ST3)
DDELTA2	R	Weight of k^2 part in WAM4 dissipation.		(!/ST3)
SSDSLF	R	WAM4 switch in non-saturated part.		(!/ST3)
SSDSHF	R	WAM4 switch in saturated part of spectrum.		(!/ST3)
SSDSC2	R	Constant for SAT-part of dissipation.		(!/ST3)
SSDSC3	R	Modification parameter for saturation power.		(!/ST3)
SSDSC4	R	Hard relative threshold for saturation.		(!/ST3)
SSDSC5	R	Constant for wave-turbulence term.		(!/ST3)
SSDSBR	R	Saturation threshold for wave breaking.		(!/ST3)
SSDSP	R	Base value of power of saturation.		(!/ST3)
SSDSDTH	R	Directional window for saturation.		(!/ST3)
WWNMEANP	R	Mean wavenumber power for $S_{\rm ds}$.		(!/ST3)
WWNMEANPTAIL	R	Mean wavenumber power for the tail.		(!/ST3)
SSTXFTWN,SSTXFTF,SSTXFTFTAIL	R	Tail coefficients for mean wavenumber.		(!/ST3)

The structure SNLP contains the following alias pointers, presented as above.

SNLC1	R	Scaled proportionality constant.	(!/NL1)
LAM	R	Factor defining quadruplet.	(!/NL1)
KDCON	R	Conversion factor for relative depth.	(!/NL1)
KDMN	R	Minimum relative depth.	(!/NL1)
SNLSN	R	Constants in shallow water factor.	(!/NL1)
IQTPE	I	Type of depth treatment.	(!/NL2)

NDPTHS	I	Number of depth for which integration space needs to be computed.	(!/NL2)
NLTAIL	R	Tail factor for parametric tail.	(!/NL2)
DPTHNL	RA	Depths corresponding to NDPTHS.	(!/NL2)

The structure SBTP contains the following alias pointer, presented as above.

SBTC1	R	Proportionality constant.	(!/BT1)

The structure SDBP contains the following alias pointer, presented as above.

SDBC1	R	Proportionality constant.	(!/DB1)
SDBC2	R	$H_{\max}/d$ ratio.	(!/DB1)
FDONLY	L	Flag for chocking depth only, otherwise use Miche criterion.	(!/DB1)

The structure STRP contains the following alias pointer, presented as above.

DUMMY	R	Placeholder only.	(!/TR0-X)

The structure SBSP contains the following alias pointer, presented as above.

DUMMY	R	Placeholder only.	(!/BS0-X)

The structure SXXP contains the following alias pointer, presented as above.

DUMMY	R	Placeholder only.	(!/XX0-X)

6.6.3 Data in W3WDATMD.

The module W3WDATMD in the file w3wdatmd.ftn contains the dynamic data of the model, that is, spectra and other fields describing the instantaneous wave field. First several variables and parameters embedded in the module are defined:

NWDATA	I	Number of models in array dim.
IWDATA	I	Selected model, initialized at -1.
WDATA	T	Basic data structure.

WDATAS	TA	Array of data structures.

For each element of the type WDATA, direct aliases are defined with the same name, as in W3GDATMD. These variables are:

TIME	IA	Valid time for spectra.
TLEV	IA	Valid time for water levels.
TICE	IA	Valid time for ice.
VA	RA	Storage array for spectra.
WLV	RA	Water levels.
ICE	RA	Ice coverage.
UST	RA	Friction velocities (absolute value).
USTDIR	RA	Friction velocities (direction).
ASF	RA	Stability correction factor.
FPIS	RA	Input peak frequencies.
DINIT	L	Flag for array initialization.
FL_ALL	L	Flag for array initialization to include VA.

6.6.4 Data in W3ADATMD.

The module W3ADATMD in the file w3adatmd.ftn contains data that are used inside the wave model only. Embedded in the module are the following parameters:

NADATA	I	Number of models in array dim.
IADATA	I	Selected model for output, init. at -1.
MPIBUF	IP	Number of buffer arrays for 'hidden' MPI communications (no hiding for MPIBUF = 1).
WADAT	T	Basic data structure.
WADATS	TA	Array of data structures.

For each element of the type WADAT, alias pointers are defined as in the module W3ADATMD. Within the type, several groups of variables are present. The first are the internal grid definition of the model:

CG	RA	Group velocities for all wave model sea points and frequencies.

WN	RA	Idem, wavenumbers.

The second group of parameters consists auxiliary arrays needed to process model input:

CA0-I	RA	Absolute current velocity (initial and inc.) in W3UCUR. (m/s)
CD0-I	RA	Current direction (initial and increment) in W3UCUR. (rad)
UA0-I	RA	Absolute wind speeds (init. and incr.) in W3UWND (m/s)
UD0-I	RA	Wind direction (initial and incr.) in W3UWND (rad)
AS0-I	RA	Stability par. (initial and incr.) in W3UWND (degr)
ATRNX-Y	RA	Actual transparency info.

The third group of parameters consists of internal gridded fields of parameters:

DW	RA	Water depths.
UA	RA	Absolute wind speeds.
UD	RA	Absolute wind direction.
U10	RA	Wind speed used.
U10D	RA	Wind direction used.
AS	RA	Stability parameter.
CX/Y	RA	Current components.
EMN	RA	Mean energy.
FMN	RA	Mean frequency.
WNM	RA	Mean wavenumber.
AMX	RA	Spectral maximum.
CDS	RA	Drag coefficient.
Z0S	RA	Roughness parameter.
HS	RA	Wave height.
WLM	RA	Mean wave length.
TMN	RA	Mean wave period.
THM	RA	Mean wave direction.
THS	RA	Mean directional spread.
FP0	RA	Peak frequency.
THP0	RA	Peak direction.

FP1	RA	Wind sea peak frequency.
THP1	RA	Wind sea peak direction.
DTDYN	RA	Mean dynamic time step (raw).
FCUT	RA	Cut-off frequency for tail.
ABA	RA	Near bottom rms wave excursion amplitude.
ABD	RA	Corresponding direction.
UBA	RA	Near bottom rms wave velocity.
UBD	RA	Corresponding direction.
Sxx	RA	Radiation stress components.
PHS	RA	Wave height of partition of spectrum.
PTP	RA	Peak period of partition of spectrum.
PLP	RA	Peak wave length of partition of spectrum.
PTH	RA	Direction of partition of spectrum.
PSI	RA	Directional spread of partition of spectrum.
PWS	RA	Wind sea fraction of partition of spectrum.
PWST	RA	Wind sea fraction of total spectrum.
PNR	RA	Number of wave fields in partitioning.
DDDx	RA	Spatial derivatives of the depth.
DCxDx	RA	Spatial derivatives of the current.

The fourth group of parameters consists of map data for the first order propagation scheme (!/PR1).

IS0-2	IA	Spectral propagation maps.
FACVX-Y	RA	Spatial propagation factor map.

The fifth group of parameters consists of map data for the third order propagation scheme (!/PR2-4).

NMX-Yn	I	Counters for MAPX2, see W3MAP3.
NMXY	I	Dimension of MAPXY.
NACT1-2	I	Dimension of MAPAXY.
NCENT	I	Dimension of MAPAXY.
MAPX2	IA	Map for prop. in 'x' (longitude) dir.
MAPY2	IA	Idem in y' (latitude) direction.
MAPXY	IA	
MAPAXY	IA	List of active points used in W3QCK1.
MAPCXY	IA	List of central points used in avg.

MAPTH2	IA	Like MAPX2 for refraction (rotated and shifted, see W3KTP3). Like MAPAXY.
MAPWN2	IA	Like MAPX2 for wavenumber shift.
MAPTRN	LA	Map to block out GSE mitigation in proper grid points.

The sixth group of parameters consists variables used by the parameterizations for the nonlinear interactions :

NFR	I	Number of frequencies (NFR = NK).	(!/NL1)
NFRHGH	I	Auxiliary frequency counter.	(!/NL1)
NFRCHG	I	Id.	(!/NL1)
NSPECX-Y	I	Auxiliary spectral counter.	(!/NL1)
IPnn	IA	Spectral address for S_{nl}.	(!/NL1)
IMnn	IA	Id.	(!/NL1)
ICnn	IA	Id.	(!/NL1)
DALn	R	Lambda dependent weight factors.	(!/NL1)
AWGn	R	Interpolation weights for Snl.	(!/NL1)
SWGn	R	Interpolation weights for diag. term.	(!/NL1)
AF11	RA	Scaling array (f^{11}).	(!/NL1)
NLINIT	L	Flag for initialization.	(!/NL1)

The seventh group of parameters consists MPP and MPI variables:

IAPPRO	IA	Processor numbers for propagation calc. for each spectral component.	
MPI_COMM_WAVE, MPI_COMM_WCMP			
	I	mpi communicator for wave model.	(!/MPI)
WW3_FIELD_VEC, WW3_SPEC_VEC			
	I	MPI derived vector types.	(!/MPI)
NRQSG1	I	Number of handles in IRQSG1.	(!/MPI)
NRQSG2	I	Number of handles in IRQSG2.	(!/MPI)
IBFLOC	I	Present active buffer number.	(!/MPI)
ISPLOC	I	Corresponding local spectral bin number (1,NSPLOC,1).	(!/MPI)
NSPLOC	I	Total number of spectral bins for which prop. is performed on present CPU.	(!/MPI)
BSTAT	IA	Status of buffer (size MPIBUF): 0: Inactive.	(!/MPI)

1: A → STORE (active or finished).
2: STORE → A (active or finished).

BISPL	IA	Local spectral bin number for buffer (size MPIBUF). (!/MPI)
IRQSG1	IA	MPI request handles for scatters and gathers to VA (persistent). (!/MPI)
IRQSG2	IA	MPI request handles for gathers and scatters to STORE (persistent). (!/MPI)
GSTORE, SSTORE	RA	Communication buffer (NSEA,MPIBUF). (!/MPI)
SPPNT	RA	Communication buffer (NTH,NK,4).

The final group of parameters consist of all other internal auxiliary parameters that need to be saved between model runs:

ITIME	I	Discrete time step counter.
IPASS	I	Pass counter for log file.
IDLAST	I	Last day ID for log file.
NSEALM	I	Maximum number of local sea points.
ALPHA	RA	Phillips' alpha.
FLCOLD	L	Flag for 'cold start' of model.
FLIWND	L	Flag for initialization of model based on wind.
AINIT	L	Flag for array initialization.
FL_ALL	L	Flag for all or partial array initialization.

6.6.5 Data in W3ODATMD.

The module W3ODATMD in the file w3odatmd.ftn contains the data describing model output. First several variables and parameters embedded in the module are defined:

NOUTP	I	Number of models for which space is available.
IOUTP	I	Number of model/grid presently selected.
IOSTYP	I	Flag for type of output server approach.
NOGRD	IP	Number of output field typed defined.
NOSWLL	IP	Number of swell fields from partitioning to be stored in fields.

NOEXTR	IP	Number of slots for user-defined output fields.
IDOUT	CA	ID strings for output fields.
FNMPRE	CA	File name preamble.
UNDEF	R	Value for undefined parameters in gridded output fields.
UNIPTS	L	Flag for unified point output (multi-grid model).
UPPROC	L	Flag for separate processor for unified point output (multi-grid model).
OUTPUT	T	Data structure containing output information.
OUTPTS	TA	Array of type OUTPUT and size NOUTP.

For each element of the type OUTPUT, alias pointers are defined as in the module W3GDATMD. These variables are:

NDSO	I	General output unit number (log file).
NDSE	I	Error output unit number.
NDST	I	Test output unit number.
SCREEN	I	Unit for 'direct' output.
NTPROC	I	Number of processors (total).
NAPROC	I	Number of processors (computation only).
IAPROC	I	Actual processor number (starting at 1),
NAPLOG	I	Proc. dealing with log output.
NAPOUT	I	Proc. dealing with standard output.
NAPERR	I	Proc. dealing with error output.
NAPFLD	I	Proc. dealing with raw field output.
NAPPNT	I	Proc. dealing with raw point output.
NAPTRK	I	Proc. dealing with track output.
NAPRST	I	Proc. dealing with restart output.
NAPBPT	I	Proc. dealing with boundary output.
NAPPRT	I	Proc. dealing with partitioning output.
TOFRST	IA	Times for first output.
TONEXT	IA	Times for next output.
TOLAST	IA	Times for last output.
TBPI0	IA	Time of first set of input boundary spectra.
TBPIN	IA	Idem second set.
NDS	IA	Array with data set numbers.
DTOUT	RA	Output intervals.
FLOUT	LA	Output flags.
OUT1	T	Structure of type OTYPE1 with data for output type 1 (fields output).

OUT2	T	Idem OTYPE2, output type 2 (point output).
OUT3	T	Idem OTYPE3, output type 3 (track output).
OUT4	T	Idem OTYPE4, output type 4 (boundary data output).
OUT5	T	Idem OTYPE5, output type 5 (restart files).
OUT6	T	Idem OTYPE6, output type 6 (partitioning output).

For each element of the type OTYPE1, alias pointers are defined as:

IPASS1	I	Pass counter in IO routine for file management.
NRQGO	I	Number of handles in IRQGO. (!/MPI)
IRQGO	IA	MPI request handles for field output. (!/MPI)
FLOGRD	LA	Array with flags for output fields.
WRITE1	L	Flag for reading or writing data.

For each element of the type OTYPE2, alias pointers are defined as:

IPASS2	I	Pass counter in IO routine for file management.
NOPTS	I	Number of output points.
NRQPO(2)	I	Number of MPI handles IRQPOn. (!/MPI)
IPTINT	IA	Id. interpolation counters.
IL	IA	Number of land points in interpolation box for output point.
IW	IA	Id. water.
II	IA	Id. ice.
IRQPO1/2	IA	Array with MPI handles. (!/MPI)
PTLOC	RA	Name of output locations.
PTIFAC	RA	Id. weights.
DPO	RA	Interpolated depths.
WAO	RA	Interpolated wind speeds.
WDO	RA	Interpolated wind directions.
ASO	RA	Interpolated air-sea temp. diff.
CAO	RA	Interpolated current speeds.
CDO	RA	Interpolated current directions.
SPCO	RA	Output spectra.
PTNME	CA	Output locations.
GRDID	CA	Originating grid ID.
O2INIT	L	Flag for array initialization.
O2IRQI	L	Flag for array initialization.

For each element of the type OTYPE3, alias pointers are defined as:

IPASS3	I	Pass counter in IO routine for file management.	
IT0PNT	I	Base tag number of MPI communication.	
IT0TRK	I	Base tag number of MPI communication.	
IT0PRT	I	Base tag number of MPI communication.	
NRQTR	I	Number of handles in IRQTR.	(!/MPI)
IRQTR	IA	Array with MPI handles.	(!/MPI)
O3INIT	L	Flag for array initialization.	
STOP	L	Flag for end of output.	
MASKn	LA	Mask arrays for internal use.	

For each element of the type OTYPE4, alias pointers are defined as:

IFILE4	I	File number for output files.	
NBLKRS	I	Number of blocks in communication of spectra.	(!/MPI)
RSBLKS	I	Corresponding block size.	(!/MPI)
NRQRS	I	Number of MPI handles.	(!/MPI)
IRQRS	IA	Array with MPI handles.	(!/MPI)
IRQRSS	IA	Array with MPI handles.	(!/MPI)
VAAUX	IA	Aux. spectra storage.	(!/MPI)

For each element of the type OTYPE5, alias pointers are defined as:

NBI(2)	I	Number of input bound. points.	
NFBPO	I	Number of files for output bound. data.	
NRQBP(2)	I	Number of MPI handles.	(!/MPI)
NBO(2)	IA	Number of output boundary points. per file.	
NDSL	IA	Array with unit numbers.	
IPBPI	IA	Interpolation data for input boundary points.	
ISBPI	IA	Sea point counters for input boundary points.	
IRQBP1/2	IA	Array with MPI handles.	(!/MPI)
X/YBPI	RA	Location of input boundary points.	
RDBPI	RA	Interpolation factors for input boundary points.	
ABPI0/N	RA	Storage of spectra from which to interpolate boundary data.	
BBPI0/N	RA	Id., secondary storage.	
ABPOS	RA	Temporarily storage for output boundary data.	

IPBPO, ISBPO, X/YBPO, RDBPO		
	–	Id. for output boundary points.
O5INIn	L	Flags for array initializations.
FLBPI	L	Flag for input of boundary data.
FLBPO	L	Flag for output of boundary data.
FILER	L	Read flag for file management.
FILEW	L	Write flag for file management.
FILED	L	Dump flag for file management (multi-grid model).
SPCONV	L	Dump flag for file management (multi-grid model).

For each element of the type OTYPE6, alias pointers are defined as:

IPASS6	I	Pass counter for file management.
IHMAX	I	Number of discrete spectral levels.
IX0/N/S	I	First-last-step IX counters.
IY0/N/S	I	Idem IY counters.
HSPMIN	R	Minimum significant height per part.
WSMULT	R	Multiplier for wind sea boundary.
WSCUT	R	Cut-off wind factor for wind seas.
ICPRT	IA	Counters for partitions.
DTPRT	RA	Data from partitions.
FLCOMB	L	Flag for combining wind seas.
FLFORM	L	Flag for (un)formatted output.
O6INIT	L	Flag for array initializations.

6.6.6 Data in W3IDATMD.

The module W3IDATMD in the file w3idatmd.ftn contains the input data for the wave model. First several variables and parameters embedded in the module are defined:

NIDATA	I	Number of models for which space is available.
IIDATA	I	Number of model/grid presently selected.
INPUT	T	Data structure containing output information.
INPUTS	TA	Array of type INPUT and size NIDATA.

For each element of the type INPUT, alias pointers are defined as in the module W3GDATMD. These variables are:

TLN	IA	Time for water level field.	
TC0/N	IA	Times for current fields.	
TW0/N	IA	Times for wind fields.	
TIN	IA	Time for ice field.	
TnN	IA	Time for data types 1-3.	
TDN	IA	Time for next data.	
TG0/N	IA	Times for grid motion data.	
TFN	IA	Array consolidating most above times.	
GA0/N	R	Norm of grid speed vector.	(ms^{-1})
GD0/N	R	Direction of grid speed vector.	(?)
WX0/N, WY0/N			
	RA	Cartesian wind components.	(ms^{-1})
DT0/N	RA	Air-sea temperature differences.	(°C)
CX0/N, CY0/N			
	RA	Cartesian current components	(ms^{-1}).
WLEV	RA	Water level field.	(m)
ICEI	RA	Ice concentrations.	(-)
IINIT	L	Flag for array initialization.	
FLLEV	L	Flag for water level input.	
FLCUR	L	Flag for current input.	
FLWIND	L	Flag for wind input.	
FLICE	L	Flag for ice input.	
FLAGS	LA	Array consolidating the above four flags, as well as four additional data flags.	

6.6.7 Data in WMMDATMD.

The module WMMDATMD in the file wmmdatmd.ftn contains the input data for the wave model. First several variables and parameters embedded in the module are defined:

NMDATA	I	Number of models in array dim.
IMDATA	I	Selected model in data structure.

MDSI	I	Unit number for input file.
MDSO	I	Unit number for output (log file).
MDSS	I	Unit number for output (screen).
MDST	I	Unit number for test output.
MDSE	I	Unit number for error output. These outputs correspond to similar unit numbers as defined per grid, but are used for multi-grid routines only.
MDSP	I	Unit number for profiling.
MDSUP	I	Unit number for unified point output.
MDSF	IA	Unit numbers for input files.
NMPROC	I	Number of processors (for total multi- grid model).
IMPROC	I	Corresponding actual processor number.
NMPLOG, NMPSCR, NMPTST, NMPERR, NMPUPT	I	Processors in NMPROC designated for the above output units numbers.
STIME	IA	Model run starting time.
ETIME	IA	Model run ending time.
TSYNC	IA	Synchronization time for grids.
TMAX	IA	Maximum next time per grid.
TOUTP	IA	Next output time for grids.
TDATA	IA	Time for which data is available.
NRGRD	I	Number of grids.
NRINP	I	Number of input grids.
NRGRP	I	Number of groups.
NMVMAX	I	Number of moving grid data.
GRANK	IA	Rank number for grid.
GRGRP	IA	Group number for grid.
INGRP	IA	Grids in group, element 0 is number.
GRDHGH, GRDEQL, GRDLOW	IA	Dependent grids with higher, same or lower rank number, element 0 is number.
ALLPRC	IA	Map of processors in MPI_COMM_MWAVE for all individual grids.
MODMAP	IA	Map which model is running where in MPI_COMM_MWAVE each group.
GRSTAT	IA	Grid computation status indicator.
DTRES	RA	Residual of time step.

NBI2G	IA	Map cross-referencing how many spectra echo grid provides to boundary cond. for other grids.
RESPEC	LA	Map for need to convert spectra between grids.
BCDUMP	LA	Flag for dumping internal bound. data.
INPMAP	IA	Map for external input grids.
IDINP	CA	Input field identifiers.
CLKT0, CLKINC, CLKMAX		
	I	Global wall clock parameters,
FLGBDI	L	Flag for initialization of boundary distance maps.
FLGHGN	L	Flags for using mask for computations and output for areas of grid overlap.
IFLSTI	LA	Flags for last ice per grid.
IFLSTL	LA	Flags for last level per grid.
MPI_COMM_MWAVE		
	I	MPI communicator. (!/MPI)
MTAGN	I	"Zero" tag number for MPI. (!/MPI)
NBISTA	IA	Status for gathering input boundary data. (!/MPI)
HGHSTA	IA	Status for gathering high resolution data. (!/MPI)
EQLSTA	IA	Status for gath. data for equally ranked grids. (!/MPI)
MDATA	T	Data structure for grid dependent data.
MDATAS	TA	Array of data structures.
BPST	T	Data structure for staging boundary data.
BPSTGE	TA	Array of data structures.
HGST	T	Data structure for staging 2-way nesting data.
HGSTGE	TA	Array of data structures.
EQST	T	Data structure for staging equal grid reconciliation data.
EQSTGE	TA	Array of data structures.

For each element of the type MDATA, alias pointers are defined as in the module W3GDATMD. These variables are:

NBI2S	IA	Source information of boundary input data (grid number and sea counter).
MAPBDI	RA	Map with distances to boundary.
MAPODI	RA	idem, open edges of grids.
NRUPTS	I	Number of unified output points.
UPTMAP	IA	Mapping of unified points to grids.

MAPMSK	IA	Mask corresponding to FLGHGn above.
MINIT, MSKINI, FLDATn		
	L	Flags for array initializations.
FLLSTI	L	Flag for last ice per grid.
FLLSTL	L	Flag for last level per grid.
NMV	I	Number of moving grid data.
TMV	IA	Moving grid times.
AMV	RA	Moving grid velocities.
DMV	RA	Moving grid directions.
RCLD	IA	Record length for data assimilation.
NDT	IA	Number of data for data assimilation.
DATAN	RA	Assimilation data.
MPI_COMM_GRD, MPI_COMM_BCT		
	I	Communicators for grid and broadcast. (!/MPI)
CROOT	I	"root" for MPI_COMM_GRD in MPI_COMM_MWAVE. (!/MPI)
FBCAST	L	Flag for need of broadcasting data to processors that are not in the communicator. (!/MPI)
NRQBPG	I	Number of request handles. (!/MPI)
IRQBPG	IA	Request handles. (!/MPI)
NRQHGG	I	Number of request handles. (!/MPI)
IRQHGG	IA	Request handles. (!/MPI)
NRQEQG	I	Number of request handles. (!/MPI)
IRQEQG	IA	Request handles. (!/MPI)

For each element of the type BPST, alias pointers are defined as in the module W3GDATMD. These variables are:

NRQBPS	I	Number of request handles. (!/MPI)
IRQBPS	IA	Request handles. (!/MPI)
VTIME	IA	Valid time of data.
STIME	IA	Buffer for time for sending. (!/MPI)
SBPI	RA	Spectral data storage.
TSTORE	RA	Spectral data buffer. (!/MPI)
INIT	L	Flag for array allocation.

For each element of the type HGST, alias pointers are defined as in the module W3GDATMD. These variables are:

NRQHGS	I	Number of request handles.	(!/MPI)
IRQHGS	IA	Request handles.	(!/MPI)
NRQOUT	I	Number of local spectra.	(!/MPI)
OUTDAT	IA	Corresponding data.	(!/MPI)
NTOT, NREC, NRC1, NSND, NSN1, NSMX			
	I	Counters for total data, send and received data with and without masking.	
VTIME	IA	Valid time of data.	
LJSEA	IA	Local sea point counters.	
NRAVG	IA	Number of points in averaging.	
IMPSRC	IA	Source processor for data.	
ITAG	IA	Communication tag.	
ISEND	IA	Composite of all data needed for send.	
WGHT	RA	Weights in averaging.	
SHGH	RA	Staging area for spectra.	
TSTORE	RA	Staging area for spectra to be send out.	(!/MPI)
INIT	L	Flag for array allocation.	

For each element of the type EQST, alias pointers are defined as in the module W3GDATMD. These variables are:

NRQEQS	I	Number of request handles.	(!/MPI)
IRQEQS	IA	Request handles.	(!/MPI)
NRQOUT	I	Number of local spectra.	(!/MPI)
OUTDAT	IA	Corresponding data.	(!/MPI)
NTOT, NREC, NSND, NAVMAX			
	I	Counters for total data, send and received data.	
VTIME	IA	Valid time of data.	
I/JSEA	IA	Sea point counters.	
NAVG	IA	Number of spectra in averaging.	
RIP	IA	Processor (receiving).	
RTG	IA	Tag number (receiving).	
SIS,SJS	IA	Sea point counter (sending).	
SI1/2	IA	Storage array counters (sending).	
SIP	IA	Processor (sending).	
STG	IA	Tag (sending).	
SEQL	RA	Staging array.	
WGHT	RA	Weight between grids.	

WAVG	RA	Weight within grid.	
TSTORE	RA	Staging area for spectra to be send out.	(!/MPI)
INIT	L	Flag for array allocation.	

This page is intentionally left blank.

References

Abdalla, S. and J. R. Bidlot, 2002: Wind gustiness and air density effects and other key changes to wave model in CY25R1. Tech. Rep. Memomrandum R60.9/SA/0273, Research Department, ECMWF, Reading, U. K.

Alves, J. H. G. M., M. L. Banner and I. R. Young, 2003: Revisiting the Pierson-Moskowitz asymptotic limits for fully developed wind waves. *J. Phys. Oceanogr.*, **33**, 1301–1323.

Ardhuin, F., B. Chapron and F. Collard, 2008: Ocean swell evolution from distant storms. *Nature Geoscience*.

Ardhuin, F. and T. H. C. Herbers, 2002: Bragg scattering of random surface gravity waves by irregular sea bed topography. *J. Fluid Mech.*, **451**, 1–33.

Ardhuin, F. and A. D. Jenkins, 2006: On the interaction of surface waves and upper ocean turbulence. *J. Phys. Oceanogr.*, **36**(3), 551–557.

Ardhuin, F. and R. Magne, 2007: Current effects on scattering of surface gravity waves by bottom topography. *J. Fluid Mech.*, **576**, 235–264.

Banner, M. L., J. R. Gemmrich and D. M. Farmer, 2002: Multiscale measurement of ocean wave breaking probability. *J. Phys. Oceanogr.*, **32**, 3364–3374.

Battjes, J. A. and J. P. F. M. Janssen, 1978: Energy loss and set-up due to breaking of random waves. in *Proc. 16th Int. Conf. Coastal Eng.*, pp. 569–587. ASCE.

Bidlot, J. R., S. Abdalla and P. A. E. M. Janssen, 2005: A revised formulation for ocean wave dissipation in CY25R1. Tech. Rep. Memorandum R60.9/JB/0516, Research Department, ECMWF, Reading, U. K.

Bidlot, J. R., P. A. E. M. Janssen and S. Abdalla, 2007: A revised formulation of ocean wave dissipation and its model impact. Tech. Rep. Memorandum 509, ECMWF, Reading, U. K.

Booij, N. and L. H. Holthuijsen, 1987: Propagation of ocean waves in discrete spectral wave models. *J. Comput. Physics*, **68**, 307–326.

Booij, N., R. C. Ris and L. H. Holthuijsen, 1999: A third-generation wave model for coastal regions, Part I, model description and validation. *J. Geophys. Res.*, **104**, 7649–7666.

Bouws, E. and G. J. Komen, 1983: On the balance between growth and dissipation in an extreme depth-limited wind-sea in the southern north sea. *J. Phys. Oceanogr.*, **13**, 1653–1658.

Bretherthon, F. P. and C. J. R. Garrett, 1968: Wave trains in inhomogeneous moving media. *Proc. Roy. Soc. London*, **A 302**, 529–554.

Cahyono, 1994: *Three-dimensional numerical modelling of sediment transport processes in non-stratified estuarine and coastal waters*. Ph.D. Thesis, University of Bradford, 315 pp.

Cavaleri, L. and P. Malanotte-Rizzoli, 1981: Wind-wave prediction in shallow water: Theory and applications. *J. Geophys. Res.*, **86**, 10,961–10,973.

Chalikov, D. V., 1995: The parameterization of the wave boundary layer. *J. Phys. Oceanogr.*, **25**, 1333–1349.

Chalikov, D. V. and M. Y. Belevich, 1993: One-dimensional theory of the wave boundary layer. *Bound. Layer Meteor.*, **63**, 65–96.

Charnock, H., 1955: Wind stress on a water surface. *Quart. J. Roy. Meteor. Soc.*, **81**, 639–640.

Chawla, A. and H. L. Tolman, 2007: Automated grid generation for WAVEWATCH III. Tech. Note 254, NOAA/NWS/NCEP/MMAB, 71 pp.

Chawla, A. and H. L. Tolman, 2008: Obstruction grids for spectral wave models. *Ocean Mod.*, **22**, 12–25.

Christoffersen, J. B., 1982: Current depth refraction of dissipative water waves. Series Paper 30, Institute of Hydrodynamics and Hydraulic Engineering, Techn. Univ. Denmark.

Davis, R. W. and E. F. More, 1982: A numerical study of vortex shedding from rectangles. *J. Fluid Mech.*, **116**, 475–506.

Dore, B. D., 1978: Some effects of the air-water interface on gravity waves. *Geophys. Astrophys. Fluid. Dyn.*, **10**, 215–230.

Doty, B., 1995: *The grid analysis and display system GrADS*. COLA, http://www.iges.org/grads.

Eldeberky, Y. and J. A. Battjes, 1996: Spectral modelling of wave breaking: Application to boussinesq equations. *J. Geophys. Res.*, **101**, 1253–1264.

Falconer, R. A. and Cayhono, 1993: Water quality modelling in well mixed estuaries using higher order accurate differencing schemes. in S. S. Y. Wang, editor, *Advances in Hydro- Science and Engineering*, pp. 81–92. CCHE, University of Mississippi.

Fletcher, C. A. J., 1988: *Computational techniques for fluid dynamics, part I and II*. Springer, 409+484 pp.

Grant, W. D. and O. S. Madsen, 1979: Combined wave and current interaction with a rough bottom. *J. Geophys. Res.*, **84**, 1797–1808.

Gropp, W., E. Lusk and A. Skjellum, 1997: *Using MPI: Portable parallel programming with the message-passing interface*. MIT Press, 299 pp.

Hanson, J. L. and R. E. Jensen, 2004: Wave system diagnostics for numerical wave models. in *8 th international workshop on wave hindcasting and*

forecasting, JCOMM Tech. Rep. 29, WMO/TD-No. 1319.

Hanson, J. L. and O. M. Phillips, 2001: Automated analysis of ocean surface directional wave spectra. *J. Atmos. Oceanic Techn.*, **18**, 177–293.

Hanson, J. L., B. A. Tracy, H. L. Tolman and D. Scott, 2006: Pacific hindcast performance evaluation of three numerical wave models. in *9 th international workshop on wave hindcasting and forecasting, JCOMM Tech. Rep. 34*. Paper A2.

Hardy, T. A., L. B. Mason and J. D. McConochie, 2000: A wave model for the Great Barrier Reef. *Ocean Eng.*, **28**, 45–70.

Hardy, T. A. and I. R. Young, 1996: Field study of wave attenuation on an offshore coral reef. *J. Geophys. Res.*, **101**, 14,311–14,326.

Hargreaves, J. C. and J. D. Annan, 1998: Integration of source terms in WAM. in *Proceedings of the 5th International Workshop on Wave Forecasting and Hindcasting*, pp. 128–133.

Hargreaves, J. C. and J. D. Annan, 2001: Comments on improvement of the short fetch behavior in the WAM model. *J. Atmos. Oceanic Techn.*, **18**, 711–715.

Hasselmann, K., 1962: On the non-linear transfer in a gravity wave spectrum, Part 1. General theory. *J. Fluid Mech.*, **12**, 481–500.

Hasselmann, K., 1963a: On the non-linear transfer in a gravity wave spectrum, Part 2, Conservation theory, wave-particle correspondence, irreversibility. *J. Fluid Mech.*, **15**, 273–281.

Hasselmann, K., 1963b: On the non-linear transfer in a gravity wave spectrum, Part 3. Evaluation of energy flux and sea-swell interactions for a Neuman spectrum. *J. Fluid Mech.*, **15**, 385–398.

Hasselmann, K., T. P. Barnett, E. Bouws, H. Carlson, D. E. Cartwright, K. Enke, J. A. Ewing, H. Gienapp, D. E. Hasselmann, P. Kruseman, A. Meerburg, P. Mueller, D. J. Olbers, K. Richter, W. Sell and H. Walden, 1973: Measurements of wind-wave growth and swell decay during the Joint North Sea Wave Project (JONSWAP). *Ergaenzungsheft zur Deutschen Hydrographischen Zeitschrift, Reihe A(8)*, **12**, 95 pp.

Hasselmann, S. and K. Hasselmann, 1985: Computations and parameterizations of the nonlinear energy transfer in a gravity-wave spectrum, Part I: A new method for efficient computations of the exact nonlinear transfer integral. *J. Phys. Oceanogr.*, **15**, 1369–1377.

Hasselmann, S., K. Hasselmann, J. H. Allender and T. P. Barnett, 1985: Computations and parameterizations of the nonlinear energy transfer in a gravity-wave spectrum, Part II: parameterizations of the nonlinear energy

transfer for application in wave models. *J. Phys. Oceanogr.*, **15**, 1378–1391.

Hersbach, H. and P. A. E. M. Janssen, 1999: Improvement of the short fetch behavior in the WAM model. *J. Atmos. Oceanic Techn.*, **16**, 884–892.

Hersbach, H. and P. A. E. M. Janssen, 2001: Reply to comments on "improvement of the short fetch behavior in the WAM model". *J. Atmos. Oceanic Techn.*, **18**, 716–721.

Herterich, K. and K. Hasselmann, 1980: A similarity relation for the nonlinear energy transfer in a finite-depth gravity-wave spectrum. *J. Fluid Mech.*, **97**, 215–224.

Hino, M., 1968: Equilibrium-range spectra of sand waves formed by flowing water. *J. Fluid Mech.*, **34**, 565–573.

Holthuijsen, L. H., N. Booij, R. C. Ris, I. G. Haagsma, A. T. M. M. Kieftenburg and E. E. Kriezi, 2001: *SWAN Cycle III version 40.11 user manual.* Delft University of Technology, Department of Civil Engineering, P.O. Box 5048, 2600 GA Delft, The Netherlands, see http://swan.ct.tudelft.nl.

Janssen, P. A. E. M., 1982: Quasilinear approximation for the spectrum of wind-generated water waves. *J. Fluid Mech.*, **117**, 493–506.

Janssen, P. A. E. M., 1989: Wind-induced stress and the drag of air-flow over sea waves. *J. Phys. Oceanogr.*, **19**, 745–754.

Janssen, P. A. E. M., 1991: Quasi-linear theory of of wind wave generation applied to wave forecasting. *J. Phys. Oceanogr.*, **21**, 1631–1642.

Kahma, K. K. and C. J. Calkoen, 1992: Reconciling discrepancies in the observed growth rates of wind waves. *J. Phys. Oceanogr.*, **22**, 1389–1405.

Kahma, K. K. and C. J. Calkoen, 1994: *Komen et al. (1994).* Chap. II.8 Growth curve observations, pp. 174–182. Cambridge Univ. Press.

Komen, G. J., L. Cavaleri, M. Donelan, K. Hasselmann, S. Hasselmann and P. E. A. M. Janssen, 1994: *Dynamics and modelling of ocean waves.* Cambridge University Press, 532 pp.

Komen, G. J., S. Hasselmann and K. Hasselmann, 1984: On the existence of a fully developed wind-sea spectrum. *J. Phys. Oceanogr.*, **14**, 1271–1285.

Kreisel, G., 1949: Surface waves. *Quart. Journ. Appl. Math.*, pp. 21–44.

Kuik, A. J., G. Ph. Van Vledder and L. Holthuijsen, 1988: A method for the routine analysis of pitch-and-roll buoy wave data. *J. Phys. Oceanogr.*, **18**, 1020–1034.

Leonard, B. P., 1979: A stable and accurate convective modelling procedure based on quadratic upstream interpolation. *Comput. Methods Appl. Mech. Engng.*, **18**, 59–98.

Leonard, B. P., 1991: The ULTIMATE conservative difference scheme

applied to unsteady one-dimensional advection. *Comput. Methods Appl. Mech. Engng.*, **88**, 17–74.

Longuet-Higgins, M. S. and R. W. Stewart, 1961: The changes in amplitude of short gravity waves on steady non-uniform currents. *J. Fluid Mech.*, **10**, 529–549.

Longuet-Higgins, M. S. and R. W. Stewart, 1962: Radiation stress and mass transport in gravity waves, with application to 'surf-beats'. *J. Fluid Mech.*, **10**, 529–549.

McCowan, J., 1894: On the highest wave of permanent type. *Philosophical Magazine*, **38**, 351–358.

Mei, C. C., 1983: *The applied dynamics of ocean surface waves*. Wiley, New York, 740 pp.

Metcalf, M. and J. Reid, 1999: *FORTRAN 90/95 explained, second edition*. Oxford University Press, 341 pp.

Miche, A., 1944: Mouvements ondulatoire de la mer en profondeur croissante ou décroissante. Forme limite de la houle lors de son déferlement. Application aux digues maritimes. Deuxième partie. Mouvements ondulatoires périodiques en profondeur régulièrement décroissante. *Annales des Ponts et Chaussées*, **114**, 131–164, 270–292.

Miles, J. W., 1957: On the generation of surface waves by shear flows. *J. Fluid Mech.*, **3**.

NCEP, 1998: GRIB. Office Note 388, NOAA/NWS/NCEP, Available by anonymous ftp from ftp://nic.fb4.noaa.gov.

Nelson, R. C., 1994: Depth limited wave heights in very flat regions. *Coastal Eng.*, **23**, 43–59.

Nelson, R. C., 1997: Height limits in top down and bottom up wave environments. *Coastal Eng.*, **32**, 247–254.

Phillips, O. M., 1977: *The dynamics of the upper ocean, second edition*. Cambridge Univ. Press, 336 pp.

Phillips, O. M., 1984: On the response of short ocean wave components at a fixed wavenumber to ocean current variations. *J. Phys. Oceanogr.*, **14**, 1425–1433.

Pierson, W. J. and L. Moskowitz, 1964: A proposed spectral form for fully developed wind seas based on the similarity theory of S. A. Kitaigorodskii. *J. Geophys. Res.*, **69**, 5181–5190.

Resio, D. T. and W. Perrie, 1991: A numerical study of nonlinear energy fluxes due to wave-wave interactions. Part 1: Methodology and basic results. *J. Fluid Mech.*, **223**, 609–629.

Shemdin, O., K. Hasselmann, S. V. Hsiao and K. Heterich, 1978: Nonlinear and linear bottom interaction effects in shallow water. in *Turbulent fluxes through the sea surface, wave dynamics and prediction*, pp. 347–365. NATO Conf. Ser. V, Vol 1.

Snyder, R. L., F. W. Dobson, J. A. Elliott and R. B. Long, 1981: Array measurements of atmospheric pressure fluctuations above surface gravity waves. *J. Fluid Mech.*, **102**, 1–59.

Teixeira, M. A. C. and S. E. Belcher, 2002: On the distortion of turbulence by a progressive surface wave. *J. Fluid Mech.*, **458**, 229–267.

The WISE Group, 2007: Wave modelling: the state of the art. *Progress in Oceanograpy*, In press.

Thornton, E. B. and R. T. Guza, 1983: Transformation of wave height distribution. *JGR*, **88**, 5925–5938.

Tolman, H. L., 1990: The influence of unsteady depths and currents of tides on wind wave propagation in shelf seas. *J. Phys. Oceanogr.*, **20**, 1166–1174.

Tolman, H. L., 1991: A third-generation model for wind waves on slowly varying, unsteady and inhomogeneous depths and currents. *J. Phys. Oceanogr.*, **21**, 782–797.

Tolman, H. L., 1992: Effects of numerics on the physics in a third-generation wind-wave model. *J. Phys. Oceanogr.*, **22**, 1095–1111.

Tolman, H. L., 1995: On the selection of propagation schemes for a spectral wind wave model. Office Note 411, NWS/NCEP, 30 pp + figures.

Tolman, H. L., 2001: Improving propagation in ocean wave models. in B. L. Edge and J. M. Hemsley, editors, *4th International Symposium on Ocean Wave Measurement and Analysis*, pp. 507–516. ASCE.

Tolman, H. L., 2002a: Alleviating the garden sprinkler effect in wind wave models. *Ocean Mod.*, **4**, 269–289.

Tolman, H. L., 2002b: Distributed memory concepts in the wave model WAVEWATCH III. *Parallel Computing*, **28**, 35–52.

Tolman, H. L., 2002c: Limiters in third-generation wind wave models. *The Global Atmosphere and Ocean System*, **8**, 67–83.

Tolman, H. L., 2002d: Testing of WAVEWATCH III version 2.22 in NCEP's NWW3 ocean wave model suite. Tech. Note 214, NOAA/NWS/NCEP/OMB, 99 pp.

Tolman, H. L., 2002e: User manual and system documentation of WAVEWATCH III version 2.22. Tech. Note 222, NOAA/NWS/NCEP/MMAB, 133 pp.

Tolman, H. L., 2002f: Validation of WAVEWATCH III version 1.15 for a

global domain. Tech. Note 213, NOAA/NWS/NCEP/OMB, 33 pp.

Tolman, H. L., 2003a: Optimum Discrete Interaction Approximations for wind waves. Part 1: Mapping using inverse modeling. Tech. Note 227, NOAA/NWS/NCEP/MMAB, 57 pp. + Appendices.

Tolman, H. L., 2003b: Running WAVEWATCH III on a linux cluster. Tech. Note 228, NOAA/NWS/NCEP/MMAB, 27 pp.

Tolman, H. L., 2003c: Treatment of unresolved islands and ice in wind wave models. *Ocean Mod.*, **5**, 219–231.

Tolman, H. L., 2006: Toward a third release of WAVEWATCH III; a multi-grid model version. in *9th international workshop on wave hindcasting and forecasting, JCOMM Tech. Rep. 34*. Paper L1.

Tolman, H. L., 2007: Development of a multi-grid version of WAVEWATCH III. Tech. Note 256, NOAA/NWS/NCEP/MMAB, 88 pp. + Appendices.

Tolman, H. L., 2008: A mosaic approach to wind wave modeling. *Ocean Mod.*, **25**, 35–47.

Tolman, H. L. and J. H. G. M. Alves, 2005: Numerical modeling of wind waves generated by tropical cyclones using moving grids. *Ocean Mod.*, **9**, 305–323.

Tolman, H. L., B. Balasubramaniyan, L. D. Burroughs, D. V. Chalikov, Y. Y. Chao, H. S. Chen and V. M. Gerald, 2002: Development and implementation of wind generated ocean surface wave models at NCEP. *Wea. Forecasting*, **17**, 311–333.

Tolman, H. L. and N. Booij, 1998: Modeling wind waves using wavenumber-direction spectra and a variable wavenumber grid. *The Global Atmosphere and Ocean System*, **6**, 295–309.

Tolman, H. L. and D. V. Chalikov, 1996: Source terms in a third-generation wind-wave model. *J. Phys. Oceanogr.*, **26**, 2497–2518.

Tracy, B., E.-M. Devaliere, T. Nicolini, H. L. Tolman and J. L. Hanson, 2007: Wind sea and swell delineation for numerical wave modeling. in *10th international workshop on wave hindcasting and forecasting & coastal hazerd symposium*. Paper P12.

Tracy, B. and D. T. Resio, 1982: Theory and calculation of the nonlinear energy transfer between sea waves in deep water. WES Report 11, US Army Corps of Engineers.

Tracy, F. T., B. Tracy and D. T. Resio, 2006: ERDC MSRC Resource. Tech. Rep. Fall 2006, US Army Corps of Engineers.

Van der Westhuysen, A. J., M. Zijlema and J. A. Battjes, 2007: Saturation-based whitecapping dissipation in swan for deep and shallow water. *Coastal*

Eng., **54**, 151–170.

Van Vledder, G. Ph., 2000: Improved method for obtaining the integration space for the computation of nonlinear quadruplet wave-wave interaction. in *Proceedings of the 6th International Workshop on Wave Forecasting and Hindcasting*, pp. 418–431.

Van Vledder, G. Ph., 2002a: Improved parameterizations of nonlinear four wave interactions for application in operational wave prediction models. Report 151a, Alkyon, The Netherlands.

Van Vledder, G. Ph., 2002b: A subroutine version of the Webb/Resio/Tracy method for the computation of nonlinear quadruplet interactions in a wind-wave spectrum. Report 151b, Alkyon, The Netherlands.

Vincent, L. and P. Soille, 1991: Watersheds in digital spaces: An efficient algorithm based on immersion simulations. *IEEE Transactions of Pattern Analysis and Machine Intelligence*, **13**, 583–598.

WAMDIG, 1988: The WAM model – a third generation ocean wave prediction model. *J. Phys. Oceanogr.*, **18**, 1775–1809.

Webb, D. J., 1978: Non-linear transfers between sea waves. *Deep-Sea Res.*, **25**, 279–298.

Whitham, G. B., 1965: A general approach to linear and non-linear dispersive waves using a Lagrangian. *J. Fluid Mech.*, **22**, 273–283.

WMO, 2001: WMO MANUAL ON CODES VOLUME I - International codes; part B - binary codes. Publication Number 306, WMO.

Wu, J., 1982: Wind-stress coefficients over sea surface from breeze to hurricane. *J. Geophys. Res.*, **87**, 9704–9706.

APPENDICES

This page is intentionally left blank.

A Managing multiple model versions

WARNING

If version 3.14 is implemented as an upgrade to previous versions of WAVEWATCH III ^{TM}, please note that this version may not be compatible with previous model versions. It is therefore prudent *NOT* to install the new version of WAVEWATCH III ^{TM} on top of the old version.

WARNING

When WAVEWATCH III ^{TM} is first installed, the user needs to define a 'home' directory for WAVEWATCH III ^{TM}. This information is stored in the .wwatch3.env in the users home directory, and used by virtually all WAVEWATCH III^{TM} utility scripts. If a new model version is developed or installed, it is prudent to do this in a new directory, to avoid loss of previous work or issues of possible incompatibility of model versions. In order to have the proper scripts work with the proper model version, the user has two basic options.

- Dynamically update the environment file .wwatch3.env to point to the proper directory in which the present work is done.
- Point the environment file .wwatch3.env to a generic directory name like wwatch3, and store various model versions in directories with specific names like wwatch3_3.14 or wwatch3_dev. Then make the generic name wwatch3 a symbolic link to the specific directory to select that directory to work with.

At NCEP, the second method is the method of choice.

This page is intentionally left blank.

B Setting model time steps

Model time steps are set on a grid-by-grid basis and are considered as a part of the model setup in the model definition file mod_def.ww3. This implies that in a multi-grid model set-up (using the model driver ww3_multi) each grid is associated with its own time step setting. In this section some guidance is given for setting time steps for individual grids, and for grids in a mosaic approach. Examples of practical time step setting for practical grids can be found in the individual grids used in the test cases mww3_case_01 through mww3_case_03.

B.1 Individual grids

A basic wave model grid requires the definition of four time steps as is described in section 3.1 on page 42 of this manual. Typically, the first step to consider is the CFL time step for spatial propagation, that is, the second of the four time steps defined in ww3_grid.inp for the grid considered. The critical CFL number C_c that identifies stability of the numerical scheme is defined as [compare Eq. (3.14)]

$$C_c = \frac{c_{g,\max}\Delta t}{\min(\Delta x, \Delta y)} \quad , \tag{B.1}$$

where $c_{g,\max}$ is the maximum group velocity, and Δt, Δx, and Δy are time and space increments. The maximum group velocity is the group velocity for the lowest discrete model frequency. Noting that for a given frequency the largest group velocity occurs in intermediate water depth, this maximum velocity is approximately 1.15 times the deep water group velocity for the lowest discrete spectral frequency. Note that the CFL number formally includes affects of currents [Eq. (2.9)] and grid movement [Eq. (E.1)]. The latter two effects are accounted for internally in the model by adjusting the corresponding minimum time step dynamically depending on the current velocity and the grid movement speed. Hence, the user can define this minimum propagation time step ignoring currents and grid movement. For the schemes used here the critical CFL number is 1.

The second time step to consider is the overall time step (the first time step identified in ww3_grid.inp).For maximum numerical accuracy, this time

step should be set smaller than or equal to the above CFL time step. However, particularly in spherical grids, the critical CFL condition occurs only in a few grid points. In most grid points, CFL numbers will be much smaller. In such grids, accuracy does not suffer significantly if the overall time step is take as 2 to 4 times the critical CFL time steps. Such a setting generally has a major positive impact on model economy. The key to numerical accuracy is the interpretation of the CFL number. this number represents the normalized distance over which information propagates in a single time step. Inaccuracy occurs if information propagates over several grid boxes before source terms are applied. With CFL $\approx$ 1 and the overall time step four times the CFL time step, information will propagate over four grid boxes before source terms are applied. This may lead to model inaccuracies. If, however, the maximum CFL number is 1, but the average CFL number is only 0.25, as is the case even for the lowest frequency in many spherical grids, information only propagates over one grid box in a single overall time step, and no issues with accuracy develop.

An effective overall time step also considers requested time intervals at which model forcing is available, and at which model output is requested. If input and output time steps are multiple integer times the overall time step, a balanced and consistent numerical integration scheme exists, although the model does not require this. Most important in this consideration is reproducibility of results. If input or output time steps are modified so that they are no longer an integer multiple of the overall model time step, then the actual discrete time stepping in the model will be modified by these input and output time steps, and hence an impact on actual model results may be expected. Such an impact may be notable, but is generally very minor.

The third time step to consider is the maximum refraction (and wavenumber shift) time step. For maximum model economy, this time step should be set equal to (or lager then) the overall time step. However, this will alternate the order of spatial and refraction computations for consecutive model time steps, which in cases of strong refraction may lead to a minor oscillation of wave parameter with a period of $2\Delta t$. Such oscillations can be avoided altogether by setting the maximum refraction time step to half the overall time step. Considering the minor cost of the refraction term in the model, this generally has a negligible impact on model economy. The preferred refraction time step is therefore half the overall model time step.

One note of caution is appropriate with setting this time step. To assure numerical stability, the characteristic refraction velocities are filter as in

Eq. (3.45). This filtering suppresses refraction in cases with rapidly changing bottom topography. The impact of this filtering is reduced when the refraction time step is reduced. It is therefore prudent to test a model grid with much smaller intra-spectral model time steps to assess the impact of this filtering.

The final time step to set is the minimum time step for the dynamical source term integration in section 3.5. This is a safety valve to avoid prohibitively small time steps in the source term integration. Depending on the grid increment size this is typically set to 5 to 15s. Note that increasing this time step does not necessarily improve model economy; a larger minimum source term integration time step will increase the spectral noise in the integration, which in turn may *reduce* the average source term integration time step!

B.2 Mosaics of grids

Considerations for time step settings for individual grids making up a mosaic model using ww3_multi are in principle identical to those for individual grids as discussed in the previous section. Additional considerations are:

- Overall time steps for individual grids do not need to 'match' in any way for the management algorithm for the mosaic approach to work properly. However, if identically ranked grids share overall time steps, and if integer ratios between time steps of grids with different ranks are employed, then it will be much easier to follow and predict the working of the management algorithm,

- If two grids with identical rank overlap, then the required width of the overlap area will be defined by the stencil width of the numerical scheme, and the number of times this scheme is called for the longest wave component (ratio of overall time step to maximum CFL time step. Thus, model economy for individual grids will improve with increased overall model time step, but the required overlap of equally ranked grids will then increase, reducing the economy of the mosaic approach.

This page is intentionally left blank.

C Setting up nested runs

C.1 Using ww3_shel

The mechanics of running nested models using the single-grid wave model program ww3_shel in principle is simple. A large scale model produces a file with boundary data, for instance nest1.ww3. This file is then renamed to nest.ww3 and put in the directory in which the nested (small scale) model is run. The small scale model then will automatically process the file and update the boundary conditions as required and available. Setting up the nesting consistently is more involved. A simple step-by-step method is presented here.

1) The first step is to set up the large scale model completely, but without generating boundary data for the nested model(s). Include the proper wind fields, graphical outputs etc. Test this model until you are satisfied that it works properly.

2) Set up the small scale model, for the moment ignoring the boundary conditions. Take into consideration that the boundary conditions ideally should coincide with grid lines in the large scale model to minimize the file size of the boundary data files. Set up this model in the same way as the large scale model, and test it thoroughly.

3) When the small scale model is set up satisfactorily in the above way, the boundary conditions need to be defined. Go into the file ww3_grid.inp for the small scale model, and mark all the intended input boundaries as outlined in the documentation in section 4.4.2. Make sure that the model switch !/O1 is selected in the switch file, and recompile if necessary. Run `ww3_grid` and save the screen output. The output of this program now includes a list of all points that are marked as input boundary points. Also make sure that stored copies of mod_def.ww3 for the small scale model (if any) are properly updated.

4) The next step is to include all the input boundary points in the above list as output boundary points in the large scale model. Keep

the list handy, and go to the file ww3_grid.inp for the large scale model. Add all points of the above list as output boundary points as indicated in the documentation in section 4.4.2. Make sure that all data (an no other data) is sent to a single file, and run `ww3_grid` with the proper input file. This should now give a list of output boundary points that should be consistent with the above list of input boundary points. Note that the order in which the points occur in the list is inconsequential. Again make sure that stored copies of mod_def.ww3 for the large scale model (if any) are properly updated.

5) If there are discrepancies between the two lists of points, iterate between the two previous steps until the list are consistent.

6) The next step is to start to generate the boundary data from the large scale model. This requires the nesting output to be activated in the large scale model. The output is already set up and included in the model definition file (mod_def.ww3) of the large scale model in the above steps. It now needs to be activated by setting the beginning time, time increment and ending time in the input file ww3_shel.inp for the actual model run of the large scale model. This step does not need to be performed if a second or consecutive nest is added. The large scale model will now produce the file with boundary data. If this is the first nest included the output file will be nest1.ww3. This file needs to be saved for use in the small scale model.

7) To include the nesting data in the small scale model, the above boundary data file needs to be renamed to nest.ww3 and needs to be put in the directory from which ww3_shel for the small scale model is run. If the small scale model has properly defined the input boundary points in its definition file mod_def.ww3, it will automatically process the file nest.ww3 and update the boundary data as available. At this point, two additional tests are recommended.

   - When first running the small scale model with the file nest.ww3 present, pay close attention to the output of `ww3_shel` to assure that (i) the program reports that the file nest.ww3 has been processed and has been found OK, and (ii) that no additional

warnings are present regarding incompatible or missing boundary data. Also check the log file log.ww3 to assure that the boundary data are updated at the expected times.

- When all data apparently are processed, it is illustrative and prudent to make a model run of the small scale model where the wind fields are switched off in ww3_shel.inp, and where no restart file restart.ww3 is made available. In such a model run, wave energy can only enter the domain from the boundaries. This is a good test to assure that the boundary data is passed from the large scale model to the small scale model as expected.

Additional nested models can be added in the same way. Adding a second level nest from the small scale model is also done in the same way. The model is presently set up for producing up to 9 files with boundary data per model run. There are no limitations on the number of consecutive ('telescoping') nests.

C.2 Using ww3_multi

Performing two-way nesting in the wave model driver ww3_multi is greatly simplified compared to using the wave model driver ww3_shel, because all data transfer needed is performed internally in the multi-grid wave model routines. A mosaic model system is set up by iteratively going through the following steps.

1) Set up a grid using the ww3_grid utility. Define the grid, its active boundary points and all other model information such as time steps, but *do not* attempt to generate output nesting data for other grids. This will be assessed automatically by the multi-grid wave model routines in ww3_multi. Note that the lowest ranked grid can optionally use active boundary data, either as read from file or to be kept constant during computation. Higher ranked grids will require active boundary point in order to be valid in the mosaic approach,

2) Add this grid as an extra grid to the input file ww3_multi.inp with the appropriate rank number. Running ww3_multi will identify discrepancies between grids and requested boundary data points that

can be resolved iteratively, and other discrepancies between grids. It can be tedious to remove such discrepancies by hand. The grid generation package of Chawla and Tolman (2007, 2008) checks for such discrepancies automatically, and is therefore recommended for grid generation for this version of WAVEWATCH III ™.

Note that grid on which input data fields are defined can be added in a similar way. Note that the use of land-sea masks in oceanic input fields (current, water level and ice) is recommended to assure realistic input values at coastal points.

Generally, lower ranked grids are developed first, although grid of any rank could be added at any time.

D Setting up for distributed machines (MPI)

D.1 Model setup

In order to run WAVEWATCH III ™ on a distributed memory machine using MPI, two requirements need to be met. First, all executables need to be compiled properly. This implies that the codes are compiled with the proper WAVEWATCH III ™ options (switches), and with the proper compiler options. Second, the parallel version of the model needs to be run in a proper parallel environment. This implies that the parallel codes are run on a multi-processor machine, invoking the proper parallel environment on that machine. These two issues are discussed in some detail below.

Of all the WAVEWATCH III ™ programs described in section 4, only three benefit from a parallel implementation with MPI: the actual models ww3_shel and ww3_multi, and the initial conditions program ww3_strt. ww3_strt is typically not used in operational environments, and can generally be run in single processor mode. The main reason for running ww3_strt in multi-processor mode is to reduce its memory requirements. These three codes are the only codes that manipulate all spectra for all grid points simultaneously, and hence require much more memory than all other WAVEWATCH III ™ programs. An added benefit (other than reduced run times) of running these programs in parallel is that the parallel versions of these programs require less memory per processor if the number of processors is increased.

Considering the above, it is sufficient for most implementations on parallel machines to compile only the main programs ww3_shel and ww3_multi with the MPI options. All other WAVEWATCH III ™ programs with the exception of ww3_strt are designed for single-processor use. The latter programs should not be run in a parallel environment, because this will lead to I/O errors in output files. Furthermore, there is no possible gain in run time for these codes in a parallel environment due to their design. Because all programs share subroutines, it is important to assure that this compilation is done correctly, that is, that the subroutines and main programs are compiled with compatible compiler settings. This implies that subroutines that are shared between parallel and non-parallel programs should be compiled individually for each application.

The first step for compiling the MPI version of programs is to assure that the proper compiler and compiler options are used. Examples of this for an IBM system using the xlf compiler, and a Linux system using the Portland compiler can be found in the example comp and link scripts provided with the distribution of WAVEWATCH III ™.

The second step is to invoke the proper compile options (switches) in compiling all parts of WAVEWATCH III ™. Most programs will be compiled for single-processor use. To assure that all subroutines are consistent with the main programs to which they are linked, the compile procedure should be divided into two parts. A simple script that will properly compile all WAVEWATCH III ™ programs is given in Fig. D.1. Alternatively, the commands in the script can be run interactively, while directly editing the switch file when appropriate.

An alternative way of consistently compiling the code it to first extract all necessary subroutines per code using w3_source, then put the sources and the makefile in individual directories, and compile using the make command. In this case the code for ww3_shel and ww3_multi are extracted using the appropriate MPI switches, whereas all other codes are extracted using the switches for the shared memory architecture.

After all codes have been compiled properly, the actual wave models ww3_shell and `ww3_multi` needs to be run in the proper parallel environment. The actual parallel environment depends largely on the computer system used. For instance, on NCEP's IBM systems, the number of processors and the proper environment is set in 'job cards' at the beginning of the script. The code is then directed to the parallel environment by invoking it as

```
poe ww3_shel
```

Conversely, on many Linux types systems, the MPI implementation includes the mpirun command which is typically used in the form

```
mpirun -np $NP ww3_shel
```

where the `-np $NP` option typically requests a number of processes from a resource file (`$NP` is a shell script variable with a numerical value). For details of running parallel codes on your system, please refer to the manual or user support (if available).

```
#!/bin/sh

# Generate appropriate switch file for shared and
# distributed computational environments

  cp switch switch.hold
  sed -e 's/DIST/SHRD/g' \
      -e 's/MPI //g'      switch.hold > switch.shrd
  sed 's/SHRD/DIST MPI/g' switch.hold > switch.MPI

# Make all single processor codes

  cp switch.shrd switch
  w3_make ww3_grid ww3_strt ww3_prep ww3_outf ww3_outp \
          ww3_trck ww3_grib gx_outf gx_outp

# Make all parallel codes

  cp switch.MPI switch
  w3_make ww3_shel ww3_multi

# Go back to a selected switch file

  cp switch.shrd switch
# cp switch.hold switch

# Clean up

  rm -f switch.hold switch.shrd switch.MPI
  w3_clean

# end of script
```

Figure D.1: Simple script to assure proper compilation of all WAVEWATCH III ^{TM} codes in a distributed (MPI) environment. This script assumes that the SHRD switch is selected in the switch file before the script is run.

Note that the as a part of the parallel model setup, I/O options are available to select between parallel and non-parallel file systems (see also Tolman, 2003a).

D.2 Common errors

Some of the most common errors made in attempting to run ww3_shel and ww3_multi under MPI are:

- Running in a parallel environment with a serial code (no MPI in compilation).

  This will result in corrupted data files, because all processes are attempting to write to the same file. This can be identified by the standard output of ww3_shel. The proper parallel version of the code will produce each output line only once. The non-parallel version will produce one copy of each output line for each individual process started.

- You are running in a parallel environment with a serial code (programs other than intended MPI codes).

  This will result in corrupted data files, because all processes are attempting to write to the same file. This can be identified by the standard output of the programs, which will produce multiple copies of each output line.

- ww3_shel or ww3_multi are compiled properly, but not run in a parallel environment.

  On some systems, this will result in automatic failure of the execution of `ww3_shel`. If this does not occur, this can only be traced by using system tools for tracking when and where the code is running.

- During compilation serial and parallel compiled subroutines are mixed.

  This is the most common source of compiling, linking and run time errors of the code. Follow the steps outlined in the previous section to avoid this.

E Moving grids

E.1 General concept

In order to address wave growth issues in rapidly changing, small scale conditions such as hurricanes, an option to add a given continuous advection speed to the grid has been added to the model in model version 3.02. This model version is described in detail in Tolman and Alves (2005). In this appendix, only a cursory description is given.

WARNING

The continuously moving grid version of WAVEWATCH III ™ is only intended for testing wave model properties in highly idealized conditions. This model version should only be used for deep water without mean currents and land masses. Furthermore, to avoid complications with great circle propagation, only Cartesian grid should be used. The option is furthermore implemented only for propagations options PR1 and PR3. Note that this is not checked in the scripts or programs at either the compile or run time level. This option is not described in the body of the manual but in this appendix only, because it is not considered to be a general application.

WARNING

For the above described application Eq. (2.8) can be written as

$$\frac{\partial N}{\partial t} + (\dot{\mathbf{x}} - \mathbf{v}_g) \cdot \nabla_x N = \frac{S}{\sigma} \,, \tag{E.1}$$

where $\mathbf{v}_g$ represents the advection velocity of the grid. This option is selected when compiling the model (see section E.3). A second compile level option allows for adding the grid advection velocity $\mathbf{v}_g$ to the wind field. This allows for a simple method to assure mass conservation of a wind field independent of the actual and instantaneous grid advection velocity. The advection velocity $\mathbf{v}_g$ can vary in time and is provided by the user at the run time of the model (see section E.3).

E.2 Numerical implementation

For the simplified conditions for which Eq. (E.1) is valid, the implementation of the moving grids is trivial if it is considered that this equation is equivalent to

$$\frac{\partial N}{\partial t} + \nabla_x \cdot (\dot{\mathbf{x}} - \mathbf{v}_g)\, N = \frac{S}{\sigma}\,, \tag{E.2}$$

which in turn implies that the advection velocity $\mathbf{v}_g$ can be added directly to $\dot{\mathbf{x}}$ for arbitrary numerical schemes solving Eq. (2.8). Because this influences the net advection velocity, it also influences stability characteristics. This impact has been accounted for automatically by including the moving grid velocity in the calculation of the actual propagation time step in Eq (3.4). Hence, the user need to provide a proper maximum propagation time step representative for $\mathbf{v_g} = \mathbf{0}$ only.

The motion of the grid has an apparent influence on the Garden Sprinkler Effect (GSE), due to the different retention time in the grid of spectral components with identical frequency but different propagation direction. Current GSE alleviation methods tend to be more efficient for younger swells than for older swells. Hence, swells with longer retention time in the moving grid tend to show a more pronounced GSE (see Tolman and Alves, 2005). To mitigate this apparent imbalance in GSE alleviation, Eq. (3.40) is replaced with

$$\pm\gamma_a\gamma_{a,s}\,\Delta c_g\,\Delta t\,\mathbf{s}\quad, \pm\gamma_a\gamma_{a,n}\,c_g\Delta\theta\,\Delta t\,\mathbf{n}\quad, \tag{E.3}$$

$$\gamma_a = \left(\frac{|\dot{\mathbf{x}}|}{|\dot{\mathbf{x}} - \mathbf{v_g}|}\right)^p \tag{E.4}$$

where γ_a is a correction factor accounting for the grid movement, and where the power p is a parameter allows for some tuning. With this modification, the effects of the GSE can be distributed more evenly over the grid by rescaling the amount of smoothing applied with the expected residence time of corresponding spectral component in the moving grid (see Tolman and Alves, 2005).

E.3 Running with moving grids

To switch on the moving of the grid, or the correction of the wind field, two optional switches are added to the WAVEWATCH III ^{TM} source code:

MGP	Apply advection correction for continuous moving grid.
MGW	Apply wind correction for continuous moving grid.
MGG	Apply correction to averaging strength in GSE correction for continuous moving grid.

The advection velocity and direction is input to the shell similar to the input of homogeneous currents (see bottom of file ww3_shel.inp in section 4.4.5), exchanging the keyword 'CUR' with 'MOV'. The advection velocity can be changed in time like all homogeneous input fields. An example of running with a moving grid model is given in test case ww3_ts3.

This page is intentionally left blank.