Agenzia Nazionale per le Nuove Tecnologie,
l'Energia e lo Sviluppo Economico Sostenibile

Ministero dello Sviluppo Economico

RICERCA DI SISTEMA ELETTRICO

Modello di onde per l'area mediterranea

A. Bargagli, A. Carillo, V. Ruggiero, P. Lanucara, G. Sannino

CASPUR
Consorzio interuniversitario
per le Applicazioni di Supercalcolo
Per Università e Ricerca

Report RdS/2011/

MODELLO DI ONDE PER L'AREA MEDITERRANEA

Andrea Bargagli, Adriana Carillo, Gianmaria Sannino (ENEA)
Vittorio Ruggiero, Piero Lanucara (CASPUR)

Settembre 2011

Report Ricerca Sistema Elettrico
Accordo di Programma Ministero dello Sviluppo Economico – ENEA
Area : Produzione di energia elettrica e protezione dell'ambiente
Progetto: Studi e valutazioni sul potenziale energetico delle correnti marine

Responsabile Progetto: Gianmaria Sannino, ENEA

MODELLO DI ONDE PER L'AREA MEDITERRANEA

Sommario

1. INTRODUZIONE

Negli ultimi venti anni i modelli di simulazione del moto ondoso sono diventati degli strumenti indispensabili per le previsioni operative dello stato del mare sia in ambito scientifico che ingegneristico. Tra i vari modelli disponibili, il WAM (*WAve prediction Model*, WAMDI-Group 1988) è il più diffuso. La prima versione del modello è stata realizzata nel 1991. Attualmente il modello è utilizzato da una vasta comunità che comprende più di 150 istituti di ricerca. Il modello è stato aggiornato nel corso degli anni e allo stato attuale la versione 4.5, descritta in Komen et al. 1994 e Guenther et al. 1992, può essere considerata come lo stato dell'arte dei modelli di onde. In questa versione del modello le equazioni di base e la loro discretizzazione numerica sono rimaste fondamentalmente invariate rispetto alla versione del 1991, mentre lo schema di integrazione della funzione sorgente ha subito diverse modifiche (Hersbach - Janssen, 1999, Bidlot, et al., 2005). Altre modifiche hanno riguardato l'ingegnerizzazione e la parallelizzazione del codice originale.

Un altro modello molto diffuso per la simulazione del moto ondoso è lo SWAN (*Simulating WAves Nearshore*). Tuttavia, come sarà evidente dai risultati presentati nei paragrafi 2 e 3, le migliori prestazioni (in termini di tempo di calcolo) ottenute con il WAM sono state determinanti nella scelta finale del WAM come modello da applicare nella valutazione del potenziale energetico del moto ondoso nel bacino mediterraneo.

2. INSTALLAZIONE E VALIDAZIONE DEL MODELLO SWAN

SWAN è un modello di onde di terza generazione che descrive l'evoluzione dello spettro di densità di energia in arbitrarie condizioni di vento, correnti e batimetria. La sua implementazione fisica e gli algoritmi di calcolo numerico sono stati sviluppati e studiati appositamente per superare le tradizionali difficoltà incontrate nell'applicazione di un modello d'onda in zone costiere caratterizzate da scarsa profondità.

Lo SWAN, supportato dal Ministero dei Trasporti, dei Lavori Pubblici e della Gestione Acque dei Paesi Bassi, è stato sviluppato e viene aggiornato continuamente all'interno del Dipartimento di Meccanica dei Fluidi della Facoltà di Ingegneria Civile e Scienze Geologiche della Delft University of Technology. I suoi file sorgenti sono scritti in Fortran, sono di pubblico dominio e quindi scaricabili gratuitamente dal sito della Delft University (www.swan.tudelft.nl).

Il primo passo del lavoro è stato, appunto, l'acquisizione dei file sorgenti che sono stati installati su un'apposita area di lavoro del calcolatore ENEA di Portici (cluster CRESCO, Centro computazionale di RicErca sui Sistemi COmplessi). A questo punto, dopo una lettura della documentazione associata, si è proceduto alla compilazione dei sorgenti per generare il codice eseguibile.

Il passo successivo ha visto l'esecuzione dei quattro casi test messi a disposizione direttamente dagli sviluppatori dello SWAN. In particolare i primi tre (*refraction test*, *slanted current test*, *diffraction test*) sono casi puramente accademici, mentre il quarto, denominato *Haringvliet field case*, simula un caso più realistico; in particolare simula il moto ondoso che interessa un ramo dell'estuario di Rhine situato nel sud-ovest dell'Olanda, con la doppia scelta di descrivere l'estuario con una griglia strutturata oppure una non strutturata.

Lo SWAN è un modello parallelo e consente di sfruttare al meglio le potenzialità dei recenti calcolatori in quanto può essere utilizzato in modalità di calcolo parallelo a memoria condivisa (implementata con OpenMP (OM)), parallela a memoria distribuita (implementata con MPI (MPI)) ed ovviamente in modalità seriale (serial). Tutti i test sono stati eseguiti nelle tre diverse modalità di lavoro. Gli output dei test confrontati con le soluzioni esatte sono risultati corretti.

2.1 Test sui calcolatori ENEA

Per valutare le prestazioni numeriche del codice SWAN in condizioni realistiche è stato messo a punto un esperimento per simulare il moto ondoso nel Mediterraneo. La griglia orizzontale del modello è caratterizzata da una risoluzione orizzontale di 1/16°. Dettagli sulla batimetria usata per l'implementazione mediterranea sono riportati nel paragrafo 5. Il modello SWAN, nell'implementazione mediterranea, è stato eseguito inizialmente nella versione seriale in modo da generare i necessari dati di output da usare durante la fase di validazione di tutte le prove parallele successive.

La macchina di calcolo usata, denominata CRESCO, è installata nella sede ENEA di Portici. CRESCO è costituita da 256 Server IBM HS21 con processori Intel Xeon Quad-Core E5345 e 34 Server IBM 3850M2 con processori Intel Xeon Quad-Core E7430. Tutta l'architettura è interconnessa tramite una rete Infiniband 4X a elevate prestazioni di CISCO che ottimizza la comunicazione tra i nodi di calcolo nell'esecuzione di applicazioni che utilizzano intensivamente il calcolo parallelo distribuito (MPI).

Su questa macchina sono installati tre differenti compilatori Fortran: Intel, PGI e GNU.

Nell'ottica di dover eseguire simulazioni che richiedono grossi tempi di calcolo si è focalizzata l'attenzione sulle due versioni parallele dello SWAN. La prima operazione è stata quella di compilare lo SWAN con ciascuno dei tre compilatori nella versione parallela OMP ed eseguirlo misurando i tempi di esecuzione per realizzare un confronto diretto. A questo fine si è sfruttata la possibilità offerta dallo SWAN di implementare l'opzione **timg** che fornisce, a conclusione della simulazione, un report in cui sono indicati il tempo totale di esecuzione e nel dettaglio il tempo speso nell'eseguire le varie parti del codice.

Per il test di partenza ciascuno dei compilatori è stato utilizzato con le opzioni di ottimizzazione di *default*. Tuttavia, con il compilatore PGI, il run non terminava correttamente. Attivate le procedure di *debugging* è emerso che il comportamento anomalo era causato dall'istruzione **!$OMP FLUSH (LLOCK),** contenuta nel file **swancom1.f**.

Questa istruzione OpenMP assicura la visibilità della variabile **LLOCK** a tutti i processi che partecipano al calcolo; è dunque un'istruzione di sincronizzazione del codice e consente di mantenere la correttezza dei risultati della simulazione stessa.

Tuttavia questa istruzione OpenMP non è correttamente gestita dal compilatore PGI e di conseguenza il codice entra in una sorta di *loop* infinito. La soluzione di abbassare il livello di ottimizzazione del codice non è risultata soddisfacente in quanto penalizza le prestazioni. Per risolvere il problema e poter quindi usare il compilatore PGI con le opzioni di ottimizzazione di *default*, nella stessa *routine* si è sostituito l'istruzione **LOGICAL, ALLOCATABLE :: LLOCK(:,:)**,

con quella equivalente

LOGICAL,VOLATILE, ALLOCATABLE :: LLOCK(:,:), che istruisce il compilatore a non ottimizzare quelle parti di codice che contengono riferimenti a questa variabile.

A questo punto, utilizzando la versione WAM implementata per il Mediterraneo, è stata eseguita una simulazione della durata di 1 giorno utilizzando 8 processi (1 processo per *core*).

Intel	PGI	GNU
1287	1985	3094

Tabella 1: Tempi di esecuzione per simulare 1 giorno utilizzando il paradigma OMP con 8 *cores*

Nella Tabella 1 vengono riportati i tempi ottenuti misurati in secondi (unità di misura usata anche nel seguito) per i tre compilatori. Successivamente sono state modificate le opzioni di compilazione variando il livello di ottimizzazione, tuttavia i risultati non si discostano sostanzialmente da quelli indicati nella Tabella 1.

Per lo stesso modello, sempre utilizzando 8 *cores* per simulare 1 giorno, è stato compilato ed eseguito SWAN nella versione parallela MPI:

Intel	PGI	GNU
2252	2913	6264

Tabella 2: Tempi di esecuzione per simulare 1 giorno utilizzando il paradigma MPI con 8 *cores*

Nella Tabella 2 vengono riportati i tempi ottenuti per i tre compilatori. Anche in questo caso modificare il livello di ottimizzazione rispetto al *default* non ha portato sostanziali variazioni nei tempi indicati nella tabella.

Dall'analisi delle tabelle e da quanto detto in precedenza appare evidente che:

• Il compilatore Intel è quello che fornisce le migliori prestazioni in termini di tempi di esecuzione.

• Modificare il livello di ottimizzazione rispetto a quello di *default* non migliora le prestazioni.

• A parità di numero di *core*s la versione OMP risulta più efficiente di quella MPI.

A questo punto, avendo compilato il codice mediante il compilatore Intel è stata eseguita un'analisi della scalabilità. Di seguito nella Tabella 3 si riportano i tempi registrati per simulare 1 giorno con le versioni OMP ed MPI al variare del numero di *cores*. Dalla Tabella risulta che:

• la versione OMP scala discretamente nel passaggio da 16 a 24 *cores*, mentre le prestazioni peggiorano nettamente nell'esecuzione a 32 *cores*.

• La versione MPI ha il tempo di esecuzione che diminuisce solo leggermente all'aumentare dei *cores* di esecuzione e questo si traduce in una scarsa scalabilità.

• Ancora, a parità di numero di *cores,* la versione OMP risulta più efficiente di quella MPI.

Core	OMP	MPI
16	1009	2394
24	827	2212
32	1257	2135

Tabella 3: Tempi di esecuzione per simulare 1 giorno mediante le versioni OMP ed MPI al variare del numero dei *cores*

Mediante Scalasca (www.scalasca.org), un *tool* che misura e analizza le prestazioni a *runtime* dei codici paralleli, è stato possibile procedere ad un'analisi più dettagliata delle due versioni parallele compilate con Intel. Scalasca ha tra le sue funzionalità (vedi Appendice B) quella di permettere di visualizzare un report in cui le *routines* contenute nel programma sono sostanzialmente suddivise in tre gruppi:

- **ANY** quelle che compongono il programma;
- **OMP** (**MPI**) quelle che contengono al loro interno istruzioni di parallelizzazione;
- **COM** quelle che interagiscono con quelle che contengono istruzioni di parallelizzazione;
- **USR** quelle che sono coinvolte in operazioni puramente locali al processo.

Per ciascun gruppo viene fornito inoltre:

• **max_tbc:** il massimo delle richieste della capacità del *trace buffer* per ciascun processo espressa in *bytes*;

• **time**: il tempo totale di esecuzione come somma del tempo di esecuzione dei singoli *cores*;

• **%** la percentuale del tempo di esecuzione rispetto a quello totale.

Per la versione OMP con 8 *cores* si ottiene:

flt type	max_tbc	time	%	region
ANY	15393475848	7383.42	100	ALL
OMP	1172422920	182.57	2.47	OMP
COM	4656	24.93	0.34	COM
USR	14565265608	7175.93	97.19	USR

da cui si può vedere come la somma del carico computazionale locale sui *cores* (USR) risulta circa del 97% e questo dato permette di avere, come visto, una buona efficienza.

Per la versione MPI si ottiene:

flt type	max_tbc	time	%	region
ANY	15159173232	8533.29	100	ALL
MPI	9258888	3656.44	42.85	MPI
COM	9257904	56.43	0.66	COM
USR	15140656392	4818.66	56.47	USR

da cui risulta, viceversa, che per il run eseguito nella versione MPI, circa il 43% del tempo totale di esecuzione viene speso per le istruzioni MPI. Questo spiega la scarsa efficienza della versione MPI rispetto a quella OpenMP.

Per valutare la scalabilità della versione OMP, sono state effettuate simulazioni della durata di 1 giorno utilizzando un numero crescente di *cores*. In particolare, nella Tabella 4 è riportato solo il valore percentuale delle tre diverse parti del codice secondo la divisione riportata dal *tool* Scalasca.

Core	OMP	COM	USR
16	7.53	0.94	91.53
24	8.06	0.52	91.42
32	46.28	0.18	53.54

Tabella 4: Percentuale di tempo impiegato da SWAN, compilato in versione OMP, per eseguire le tre regioni principali del codice al variare del numero dei *cores* utilizzati. I valori si riferiscono alla simulazione della durata di 1 giorno.

I risultati ottenuti eseguendo la versione SWAN compilata in modalità MPI per una simulazione della stessa durata, sono riportati nella Tabella 5.

Core	OMP	COM	USR
16	47.35	0.64	52.01
24	51.80	0.62	47.58
32	54.52	0.62	44.96

Tabella 5: Percentuale di tempo impiegato da SWAN, compilato in versione MPI, per eseguire le tre regioni principali del codice al variare del numero dei *cores* utilizzati. I valori si riferiscono alla simulazione della durata di 1 giorno.

Per verificare la stabilità dei risultati ottenuti, sono state eseguite ulteriori simulazioni della durata di 7 giorni. Nella Tabella 6 e Tabella 7 riportiamo i dati ottenuti mediante Scalasca.

Core	OMP	COM	USR
16	9.23	0.90	89.87
24	9.19	0.59	90.22
32	51.49	0.38	48.13

Tabella 6: Percentuale di tempo impiegato da SWAN, compilato in versione OMP, per eseguire le tre regioni principali del codice al variare del numero dei *cores* utilizzati. I valori si riferiscono alla simulazione della durata di 7 giorni.

Core	MPI	COM	USR
16	47.69	0.63	51.68
24	52.27	0.62	47.11
32	54.66	0.63	44.71

Tabella 7: Percentuale di tempo impiegato da SWAN, compilato in versione MPI, per eseguire le tre regioni principali del codice al variare del numero dei *cores* utilizzati. I valori si riferiscono alla simulazione della durata di 7 giorni.

Dal confronto della Tabella 4 con la Tabella 6, emerge che aumentando il numero dei giorni di simulazione, ossia aumentando il numero di operazioni da eseguire, si nota un leggero peggioramento delle prestazioni, più netto all'aumentare dei *cores* impiegati. Tale comportamento è da ascrivere alla fase di inizializzazione di SWAN, che è sempre eseguita in modalità seriale, e che quindi non rientra nel computo del tempo necessario a svolgere operazioni OMP.

Dal confronto della Tabella 5 con la Tabella 7 risulta evidente che all'aumentare della durata della simulazione i contributi percentuali delle singole parti rimangono quasi invariati. Tale comportamento si spiega con il fatto che la percentuale di tempo spesa da SWAN per svolgere le funzioni MPI rappresenta una percentuale alta rispetto al tempo speso per il calcolo. La frazione MPI è talmente alta che il tempo speso per la fase di inizializzazione è trascurabile.

Nella Tabella 4 e nella Tabella 6 si nota il grosso aumento della parte OMP (in particolare nella Tabella 6 va dal 9.19 % al 51.49 %) nel passaggio da 24 a 32 *cores* che porta ad un degrado della scalabilità.

Mediante il *tool* Scalasca è possibile visualizzare anche il dettaglio del tempo speso per le diverse istruzioni del codice. In particolare nella prima colonna della Tabella seguente è indicata la zona di appartenenza, nella seconda il max_tbc, nella terza il tempo di esecuzione totale (somma di quello dei singoli *cores*), nella quarta la relativa percentuale rispetto al tempo totale, nella quinta l'istruzione, la *routine* e la riga a cui si riferiscono le informazioni. Di seguito sono riportati i tempi di esecuzione delle istruzioni OMP maggiormente dispendiose, per la simulazione eseguita con 24 *cores*:

flt type	max_tbc	time	%	command@routine:row
OMP	40317388104	4085.61	3.20	!$omp flush @swancom1.f:1634
OMP	848174976	233.12	0.18	!$omp flush @swancom1.f:1676
OMP	848174976	173.49	0.14	!$omp flush @swancom1.f:1674
OMP	848174976	140.99	0.11	!$omp flush @swancom1.f:1636
OMP	115776	6004.35	4.70	!$omp do @swancom1.f:1578

e quelli relativi alla stessa versione eseguita con 32 *cores*:

flt type	max_tbc	time	%	command@routine:row
OMP	101702023200	65668.56	24.76	!$omp flush @swancom1.f:1634
OMP	616854528	263.82	0.10	!$omp flush @swancom1.f:1676
OMP	616854528	182.82	0.07	!$omp flush @swancom1.f:1674
OMP	616854528	163.08	0.06	!$omp flush @swancom1.f:1636
OMP	115776	64603.30	24.36	!$omp do @swancom1.f:1578

Dal confronto delle due si può notare come nel passaggio da 24 a 32 *cores* aumenti il tempo di esecuzione dell'istruzione **!$omp flush** contenuta nella riga 1634 della routine **swancom1.f**. Poiché questa è un'istruzione di sincronizzazione questo tempo è chiaramente sottratto al calcolo con il risultato, come visto, di degradare la scalabilità del codice.

Un identico confronto è possibile farlo per la versione MPI, dove per il caso a 24 *cores* si ottiene:

flt type	max_tbc	time	%	MPI_command
MPI	33600000	61.95	0.09	MPI_Send
MPI	30912000	28213.39	41.19	MPI_Recv
MPI	80920	1727.24	2.52	MPI_Barrier
MPI	80640	5470.82	7.99	MPI_Allreduce
MPI	43760	299.19	0.44	MPI_Bcast
MPI	80	0.09	0.00	MPI_Gather
MPI	80	0.69	0.00	MPI_Gatherv
MPI	72	0.00	0.00	MPI_Comm_size
MPI	24	0.00	0.00	MPI_Initialized

mentre per il caso a 32 *cores* si ottiene:

flt type	max_tbc	time	%	MPI_command
MPI	33600000	100.96	0.14	MPI_Send
MPI	30912000	30754.67	42.42	MPI_Recv
MPI	80920	2373.14	3.27	MPI_Barrier
MPI	80640	5975.06	8.24	MPI_Allreduce
MPI	43760	386.21	0.53	MPI_Bcast
MPI	80	0.09	0.00	MPI_Gather
MPI	80	1.06	0.00	MPI_Gatherv
MPI	72	0.00	0.00	MPI_Comm_size
MPI	24	0.00	0.00	MPI_Initialized
MPI	24	0.02	0.00	MPI_Finalize
MPI	24	37.06	0.05	MPI_Init
MPI	24	0.00	0.00	MPI_Comm_rank

Dall'analisi delle tue tabelle precedenti si nota che la maggior parte del tempo viene speso da SWAN nell'esecuzione delle comunicazioni (punto-punto e collettive) e come questo tempo tenda a crescere leggermente all'aumentare del numero di *core*s sui quali si esegue la simulazione.

Per quanto visto finora le migliori prestazioni, dal punto di vista del tempo di esecuzione, si ottengono con la versione OMP eseguita con 24 *cores*. Lo stesso esperimento della durata di 1 giorno, nella versione OMP compilata per Intel, è stato eseguito anche su altri calcolatori presenti in ENEA:

• **cresco-fpga** fornita di Dual-Core AMD Opteron 8222 con cpu di 3000.262 Mhz su cui è montata una scheda FPGA. Il tempo di esecuzione su 8 *cores*, il massimo numero disponibile su un singolo nodo, è stato di circa 4083 secondi.

• **crescoc** fornita di Six-Core AMD Opteron 2427 con cpu di 2211.354 Mhz. Il tempo registrato girando su 12 *cores*, il massimo numero disponibile su un nodo, è stato di circa 1175 secondi.

Tuttavia, in questo caso sono state ottenute prestazioni peggiori rispetto alla macchina CRESCO si è deciso di usare quest'ultima come macchina su cui eseguire i run.

3. INSTALLAZIONE E VALIDAZIONE DEL MODELLO WAM

Il WAM è anch'esso un modello di simulazione di onda di terza generazione che risolve l'equazione di trasporto dell'onda esplicitamente. Il WAM consente di eseguire simulazioni su griglie a scala regionale o globale e con una risoluzione, in principio, arbitraria nello spazio e nel tempo su una griglia cartesiana. I sorgenti sono ben documentati, scritti in Fortran e, contrariamente allo SWAN, utilizza esclusivamente MPI per la distribuzione del carico di lavoro su una architettura parallela di tipo *cluster*.

3.1 Test sui calcolatori ENEA

Rispetto a SWAN, la struttura modulare di WAM ha semplificato di molto la procedura di installazione del modello. I files eseguibili vanno posti nella sottodirectory **abs** della *directory* di *root* e ciascun eseguibile è responsabile di una specifica azione (ad esempio preproc è relativa alla fase di preprocessing del modello, etc.).

Grazie all'esperienza maturata durante la fase di installazione di SWAN sui calcolatori ENEA, si è preferito utilizzare sin da subito il compilatore Intel in ambiente OpenMPI (implementazione MPI-2 sviluppata e mantenuta da un consorzio di Università, centri di ricerca ed industrie).

Nella sottodirectory **mk** sono disponibili i *makefiles* che consentono di compilare, linkare e finalmente creare gli eseguibili di cui al punto precedente; questa operazione ha comportato una parziale modifica e ristrutturazione di alcuni files del WAM.

Nella sottodirectory **jobs** sono disponibili i *template* relativi agli *script* di sottomissione dei vari eseguibili (preproc, wam, ...) sulle macchine di calcolo; questi *script* sono stati modificati per renderli operativi nell'ambiente di lavoro CRESCO.

Una prima batteria di test ha riguardato quelli disponibili nella sottodirectory jobs; a titolo di esempio lo script **jpreproc** è divenuto jpreproc.mise, jwam jwam.mise, etc. Tutti gli *script*, preparati sulla base di quelli predisposti dagli sviluppatori del WAM, sono andati a buon fine. L'unico *script* che utilizza MPI (jwam.mise) ha riportato parimenti risultati confrontabili con quelli di riferimento.

A questo punto sono stati realizzati gli *script* per i run di interesse; in particolare è stato implementato l'esperimento relativo alla simulazione del moto ondoso nel bacino mediterraneo seguendo la stessa procedura di implementazione utilizzata per SWAN.

Al fine di verificare la scalabilità del modello WAM in configurazione mediterranea, come per il modello SWAN, sono stati eseguiti diversi esperimenti variando il numero di *cores*. L'analisi delle prestazioni ha indicato che WAM risulta possedere un'ottima scalabilità, quasi lineare fino a 50 *cores*. In particolare è stato verificato che WAM impiega per simulare 1 giorno con 32 *cores* 630 secondi. Questo valore corrisponde a circa un quarto del tempo impiegato da SWAN, compilato in modalità MPI, e circa alla metà rispetto alla versione SWAN compilata in modalità OMP.

Relativamente ai tempi di calcolo, WAM risulta essere il modello migliore. Tuttavia la scelta di un modello è dettata principalmente dalla capacità che questo ha nel riprodurre i dati osservati. Per questo motivo sono stati condotti dei test per verificare i risultati ottenuti contemporaneamente con SWAN e WAM. Dal confronto è emerso che entrambi i modelli mostrano un ottimo accordo tra le variabili che descrivono i campi di onda simulati per il Mediterraneo. A parità di risultati fisici, si è deciso di utilizzare il modello che ha mostrato le migliori prestazioni computazionali.

4. MODELLO WAM

4.1 Formulazione matematica

Il vento genera onde irregolari in ampiezza e periodo, comunque le proprietà statistiche della superficie marina quali altezza media, periodo medio e direzione media dell'onda cambiano più lentamente delle effettive strutture dinamiche quali altezza della superficie marina e corrente marina istantanee. E' possibile pertanto, descrivere lo stato del mare in termini delle sopra citate quantità medie e della loro variazione nel tempo e nello spazio. Lo stato del mare è quindi rappresentabile come una sommatoria di vari contributi ondulatori:

$$\eta(t) = \sum_i \mathbf{a}_i \cos(\sigma_i t + \alpha_i),$$

dove η è l'elevazione del mare rispetto ad un livello medio, $\mathbf{a}_i$ è l'ampiezza dell'onda, σ_i è la frequenza radiante relativa e α_i è la fase del componente ondulatorio *i*-esimo. Queste quantità medie ($\mathbf{a}_i$, σ_i, α_i) sono a loro volta funzione della posizione geografica e del tempo (ad una scala maggiore del periodo medio) . Nel caso la corrente marina media $\vec{u}$ sia non nulla occorre introdurre la frequenza radiante assoluta:

$$\omega_i = \sigma_i + \vec{\boldsymbol{k}}_i \cdot \vec{u}$$

con $\vec{k}_i$ il numero d'onda della componente *i*-esima.
La relazione tra frequenza relativa e numero d'onda è detta relazione di dispersione e, nel caso di approssimazioni lineari, è data da:

$$\sigma^2 = g\boldsymbol{k}\tanh(\boldsymbol{kD})$$

dove *g* è l'accelerazione di gravità e $\boldsymbol{D}$ la profondità del mare.

La variabile fondamentale in un modello d'onda come WAM è rappresentata dallo spettro di densità di energia, quantità che è strettamente connessa allo spettro di densità di varianza, infatti:
$E = \rho_w g\langle\eta^2\rangle$ dove $\langle\eta^2\rangle$ è la densità di varianza perciò le due quantità E e $\langle\eta^2\rangle$ sono differenti solo per un fattore che è ritenuto costante $(\rho_w g)$.
La densità di varianza $\langle\eta^2\rangle$ è la funzione di auto-correlazione $C(0)$ dell'elevazione η :

$$C(\tau) = \langle\eta(t)\eta(t+\tau)\rangle,$$

cioè $\langle\eta^2\rangle = C(0) = \int_0^{+\infty} E(\omega)d_\omega$ con $\omega > 0$ la frequenza in Hertz. A sua volta $E(\omega)$ è la densità di energia alla frequenza ω indipendentemente dalla direzione di propagazione cioè:

$$E(\omega) = \int_0^{2\pi} E(\omega,\theta)d_\theta,$$

con θ indicante l'angolo della direzione perpendicolare alle creste d'onda.

Definiamo lo spettro dell'energia a un specifico numero d'onda $\vec{\boldsymbol{k}} = \begin{pmatrix} k_x \\ k_y \end{pmatrix}$, nella posizione geografica $\vec{\mathbf{x}} = \begin{pmatrix} x \\ y \end{pmatrix}$ e al tempo t come $\boldsymbol{F}(\boldsymbol{k}, \mathbf{x}, t)$, questa corrisponderà alla distribuzione della densità di varianza $\langle \eta^2 \rangle$ per lo specifico numero d'onda $\vec{\boldsymbol{k}}$ nella medesima posizione e tempo. Si può definire la densità di azione $\boldsymbol{N}$ come:

$$\boldsymbol{N} = g\, \boldsymbol{F}/\sigma$$

per ogni valore del vettore posizione $\vec{\mathbf{x}}$ e del tempo t.
In condizioni in cui la corrente $\vec{\boldsymbol{u}}$ e la profondità **D** variano lentamente in spazio e tempo, la densità di azione $\boldsymbol{N}$ è un invariante adiabatico, per cui lo scopo del modello previsionale per lo stato del mare WAM è quello di prevedere il valore di $\boldsymbol{N}(\boldsymbol{k}, \mathbf{x}, t)$ cioè la densità di azione per ogni punto di coordinate geografiche $\vec{\mathbf{x}}$, tempo t e vettore numero d'onda $\vec{\boldsymbol{k}}$.
Utilizzando il valore della densità di azione $\boldsymbol{N}$ si possono facilmente ricavare l'energia d'onda $E = \sigma \boldsymbol{N}$ e il momento dell'onda $\vec{P} = \vec{\boldsymbol{k}}\, \boldsymbol{N}$.
Quindi la relazione per il bilancio energetico permette di esprimere la tendenza di $\boldsymbol{N}$:

$$\frac{\partial \boldsymbol{N}}{\partial t} + \nabla_{\vec{x}} \cdot \left[\left(\overrightarrow{\boldsymbol{c_g}} + \vec{\boldsymbol{u}} \right) \boldsymbol{N} \right] + \left(\frac{\partial \vec{\boldsymbol{k}}}{\partial t} \right) \cdot \nabla_{\vec{k}} [\boldsymbol{N}] = \frac{S_{tot}}{\sigma} \qquad (1)$$

dove l'operatore $\nabla_{\vec{x}} = \left\{ \frac{\partial}{\partial x}, \frac{\partial}{\partial y} \right\}$ è il gradiente spaziale, l'operatore $\nabla_{\vec{k}} = \left\{ \frac{\partial}{\partial k_x}, \frac{\partial}{\partial k_y} \right\}$ è il gradiente rispetto alle componenti del vettore numero d'onda $\vec{\boldsymbol{k}}$ e infine $\overrightarrow{\boldsymbol{c_g}}$ è la velocità di gruppo ovvero la velocità di propagazione dell'energia d'onda:

$$\overrightarrow{\boldsymbol{c_g}} = \begin{pmatrix} \boldsymbol{c_x} \\ \boldsymbol{c_y} \end{pmatrix} = \nabla_{\vec{k}}(\boldsymbol{\omega}) = \begin{pmatrix} \frac{\partial \omega}{\partial k_x} \\ \frac{\partial w}{\partial k_y} \end{pmatrix} = \frac{1}{2} \left(1 + \frac{2|\vec{\boldsymbol{k}}| d}{\boldsymbol{sinh}\,(2|\vec{\boldsymbol{k}}| \boldsymbol{D})} \right) \frac{\sigma \vec{\boldsymbol{k}}}{|\vec{\boldsymbol{k}}|^2}.$$

Il secondo termine a sinistra della relazione (1) rappresenta l'avvezione dell'energia nel dominio geografico causata sia della velocità di gruppo sia dalla corrente $\vec{\boldsymbol{u}}$.
Il terzo termine a sinistra della relazione (1) rappresenta la diffrazione ovvero l'avvezione dell'energia nel dominio spettrale causata dalla velocità spettrale $\frac{\partial \vec{\boldsymbol{k}}}{\partial t}$; questo termine è nullo se il vettore numero d'onda $\vec{\boldsymbol{k}}$ si mantiene costante nel tempo.
Introducendo le coordinate sferiche longitudine λ e latitudine φ è possibile definire le nuove velocità $\boldsymbol{c_\lambda}$ e $\boldsymbol{c_\varphi}$ come:

$$\frac{d\lambda}{dt} = \boldsymbol{c_\lambda} = \frac{1}{\mathrm{R}\cos\varphi} \left[\frac{1}{2} \left(1 + \frac{2|\vec{\boldsymbol{k}}| \boldsymbol{D}}{\sinh\left(2|\vec{\boldsymbol{k}}| \boldsymbol{D}\right)} \right) \frac{\sigma \vec{\boldsymbol{k}} \cos \boldsymbol{\theta}}{\boldsymbol{k^2}} + \mathrm{u}_\lambda \right]$$

$$\frac{d\varphi}{dt} = \boldsymbol{c_\varphi} = \frac{1}{\mathrm{R}} \left[\frac{1}{2} \left(1 + \frac{2|\vec{\boldsymbol{k}}| \boldsymbol{D}}{\sinh\left(2|\vec{\boldsymbol{k}}| \boldsymbol{D}\right)} \right) \frac{\sigma \vec{\boldsymbol{k}} \sin \boldsymbol{\theta}}{\boldsymbol{k^2}} + \mathrm{u}_\varphi \right]$$

Utilizzando queste espressioni il termine avvettivo della relazione (1) può essere riscritto in termini di coordinate sferiche nella forma seguente:

$$\nabla_{\vec{x}} \cdot \left[\left(\overrightarrow{\boldsymbol{c_g}} + \vec{\boldsymbol{u}} \right) \boldsymbol{N} \right] = \frac{\partial \boldsymbol{c_\lambda} \boldsymbol{N}}{\partial \lambda} + \frac{1}{\cos\varphi} \frac{\partial \boldsymbol{c_\sigma} \cos\varphi \boldsymbol{N}}{\partial \varphi}.$$

Se il termine a destra della relazione (1) fosse nullo avremmo condizioni effettivamente adiabatiche ovvero la variazione dei parametri d'onda è determinata solo da dinamiche avvettive. In generale, invece, il termine S_{tot} rappresenta la somma delle forzanti che agiscono alla formazione delle onde marine e costituisce il termine più importante al fine di valutare correttamente la dinamica delle onde superficiali.
In particolare S_{tot} è parametrizzato come sommatoria di vari termini che rappresentano l'effetto delle varie forzanti che causano l'aumento oppure la diminuire dell'altezza d'onda:

$$S_{tot} = S_{in} + S_{nonlin} + S_{dissip}$$

Il termine S_{in} parametrizza l'effetto del vento il termine S_{dissip} parametrizza i vari contributi dissipativi e S_{nonlin} parametrizza il contributo delle interazioni non lineari tra onde di differente numero d'onda.

4.2 Parametrizzazione del termine di forzante dovuto al vento

Il termine forzante S_{in} dovuto al vento ha sempre rappresentato una notevole difficoltà sia per una corretta valutazione teorica sia come parametrizzazione. Il semplice esame sperimentale di serie temporali di elevazione del livello del mare, rilevabili tramite l'osservazione con boe o altimetri, non è molto utile, perché all'effetto del vento si sommano gli effetti dissipativi e le interazioni non lineari fra onde con differente numero d'onda.
La spiegazione comunemente accettata sulla dinamica attraverso cui il vento produce una crescita d'onda è quella del lavoro che la pressione atmosferica produce sulla superficie del mare. In questo contesto è trascurato l'effetto della turbolenza sulla dinamica delle onde di gravità, ma alcuni ricercatori ritengono che questo processo sia fondamentale. Essi ipotizzano che, dato un profilo di vento $U = U_0(z)$, possa essere definita una altezza critica z_c, dove la velocità del vento eguaglia la velocità $\boldsymbol{c}$ delle onde di gravità (cioè $\mathbf{c} = U_0(z_c)$). In questo caso è possibile un trasferimento di energia tra le onde di gravità e il profilo medio di vento e con questa ipotesi il tasso di crescita dell'onda è proporzionale alla curvatura del profilo di vento ed il termine forzante S_{in} assume la seguente formulazione:

$$S_{in} = \boldsymbol{\gamma}\, N$$

dove $\boldsymbol{\gamma}$ rappresenta il tasso di crescita dell'energia di onda.

Secondo Miles (1957) il tasso di crescita dell'onda dipende da solo due parametri:

$$\chi = u_* \left(\frac{u_*}{\mathbf{c}}\right) \cos(\theta - \phi) \qquad e \qquad \Omega_m = g\frac{z_0}{u_*^2}$$

dove u_* è la *friction-velocity*, z_0 è il parametro di rugosità dell'aria quando interagisce con la superficie marina, $\mathbf{c}$ è la velocità di fase dell'onda, ϕ è la direzione del vento e θ la direzione di propagazione dell'onda.
Il parametro profilo Ω_m agisce sulla crescita d'onda rappresentando l'effetto della rugosità che però a sua volta dipende dallo stato del mare.
Janssen(1991) propose la semplice parametrizzazione:

$$\boldsymbol{\gamma} = \omega \in \boldsymbol{\beta}\, \chi^2,$$

dove ω è la frequenza angolare, ϵ è il rapporto tra la densità dell'aria e quella dell'acqua e $\boldsymbol{\beta}$ è il cosiddetto parametro di Miles. $\boldsymbol{\beta}$ è definito da una formulazione che include l'altezza critica adimensionale $\mu = |\mathbf{k}| z_c$, dove $|\mathbf{k}|$ è il modulo del vettore numero d'onda:

$$\boldsymbol{\beta} = \left(\beta_m/_{\kappa^2}\right) \mu \, (\ln \mu)^4 \quad \text{con } \mu \leq 1,$$

dove κ è la costante di Von Karman e β_m una costante empirica, quindi l'unico parametro non ancora definito, cioè la altezza critica μ, può essere definito ancora in termini dei due parametri usati da Miles cioé Ω_m e χ :

$$\mu = \left(u_*/_{\kappa \mathbf{c}}\right)^2 \Omega_m \, exp\{\kappa/_{\chi}\}.$$

Con queste relazioni potrebbe essere calcolato $\boldsymbol{\gamma}$ (il tasso di crescita dell'energia d'onda per effetto del vento) se la rugosità z_0 fosse tenuta costante, ma questa dipende fortemente dallo stato del mare pertanto occorre considerare questa dipendenza usando la relazione di Charnock: $z_0 = |\boldsymbol{\tau}| \frac{\boldsymbol{\alpha}}{g}$ dove $|\boldsymbol{\tau}|$ è il modulo dello stress cinematico del vento e $\boldsymbol{\alpha}$ è il parametro di Charnock .

A sua volta il parametro di Charnock $\boldsymbol{\alpha}$ dipende dello stato del mare e questa dipendenza può essere espressa in funzione dello stress del vento $|\boldsymbol{\tau}|$ e dello stress delle onde $|\boldsymbol{\tau}_w|$:

$$\boldsymbol{\alpha} = \frac{\hat{\alpha}}{\sqrt{1 - \frac{|\tau_w|}{|\tau|}}},$$

dove $\hat{\boldsymbol{\alpha}} = 0.01$ e gli stress $\boldsymbol{\tau}$ e $\boldsymbol{\tau}_w$ sono così formulati:

$$\vec{\boldsymbol{\tau}} = \left(\frac{\kappa \overrightarrow{U_0}(z_{obs})}{\mathbf{ln}(z_{obs}/z_0)}\right)^2 \qquad \text{e} \qquad \overrightarrow{\boldsymbol{\tau}_w} = \frac{g}{\epsilon} \int \vec{\boldsymbol{k}} \boldsymbol{\gamma} \, N \, d\omega \, d\theta$$

z_{obs} rappresenta l'altezza al di sopra del livello del mare dove sono disponibili le misure di vento. Quando lo stress dell'aria $\boldsymbol{\tau}$ e quello delle onde $\boldsymbol{\tau}_w$ assumono moduli dello stesso ordine il parametro di Charnock $\boldsymbol{\alpha}$ assume valori molto grandi che si riflettono in una elevata rugosità e quindi l'efficienza di trasferimento di momento dal vento alle onde si incrementa in modo significativo.

La procedura per il calcolo di $\boldsymbol{\gamma}$ è iterativa poiché questo dipende dal parametro di Charnock $\boldsymbol{\alpha}$ che a sua volta dipende dallo stress d'onda $\boldsymbol{\tau}_w$che è funzione del valore di $\boldsymbol{\gamma}$ alla iterazione precedente.

4.3 Termini dissipativi

Il termine sorgente S_{dissip} è dovuto in realtà alla somma di differenti contributi.

Il valore dello stress d'onda $\boldsymbol{\tau}_w$ dipende, in modo particolare, dai modi ad alta frequenza pertanto è necessaria una corretta descrizione dei termini dissipativi a questa scala, che descrivano al meglio il principale fenomeno alla base di questa dissipazione ad alta frequenza cioè quello del *whitecapping* ovvero la rottura della cresta d'onda sotto l'azione del vento.

Con questo scopo il termine dissipativo dovuto al vento, è stato riscritto da Janssen (1989) per tener conto della sua parametrizzazione del termine di crescita $\boldsymbol{\gamma}$:

$$S_{ds} = C_{ds} \langle \omega \rangle (\langle \mathbf{k} \rangle^2 \mathbf{m_0})^2 \left[(1 - \delta) \frac{|\mathbf{k}|}{\langle \mathbf{k} \rangle} + \delta \left(\frac{|\mathbf{k}|}{\langle \mathbf{k} \rangle} \right)^2 \right] N,$$